Contrastando reconstrucciones con herramientas computacionales: una aplicación a la cladística

Ariel Jonathan Roffé

Director: Santiago Ginnobili

Codirector: Pablo Lorenzano

Consejero de estudios: Eduardo Barrio

Tesis de Doctorado en Filosofía

Universidad de Buenos Aires

Agradecimientos

Esta tesis es el fruto de cinco arduos años de estudios e investigaciones doctorales, y otros tantos de formación universitaria de grado. Tales esfuerzos no fueron puramente individuales. Hay un sinnúmero de personas e instituciones que me brindaron apoyo (de diversos tipos) y sin las cuales recorrer este camino hubiese sido imposible. A ellas y ellos deseo agradecer brevemente en este comienzo de la tesis.

Agradezco, en primer lugar, a mis padres. No solo por el soporte económico que me brindaron durante todos estos años (que fue muy importante), sino también por el afectivo, por haber creído en mí y haber confiado en todas mis decisiones. Su apoyo constante e incondicional fue sin dudas el factor más importante para poder haber llegado a concluir esta tesis.

En segundo, a mis directores. Santiago es la persona que mayor influencia tuvo sobre mi formación intelectual. Su trabajo como director, y mi admiración por él, se ven reflejados en este trabajo. La elección de los problemas, mi consideración sobre la importancia de abordarlos y los marcos conceptuales usados para solucionarlos son todos fruto de sus enseñanzas. A Pablo, por su generosidad y por haberme integrado en los diversos espacios en los que participa. Su conocimiento erudito sobre la historia de la filosofía de la ciencia y de la biología, y sobre el estructuralismo metateórico fueron una aporte invaluable para la elaboración de esta tesis.

En tercer lugar, deseo agradecer a los miembros de todos los grupos de investigación de los que participo, los cuales me brindaron amplias oportunidades de discusión de ideas (propias y ajenas), instancias de aprendizaje en ambientes intelectualmente estimulantes y grandes amistades. Agradezco por ello a los miembros del grupo ANFIBIO, liderado por Santiago, especialmente a Andrea, Carlos, Ezequiel y Federico, quienes pacientemente escucharon, criticaron y mejoraron absolutamente todos mis trabajos a lo largo de estos años (algunos de los cuales quedaron integrados en la tesis). Agradezco también al grupo dirigido por Pablo, por ayudarme a mejorar mi comprensión del estructuralismo metateórico, así como al Buenos Aires Logic Group y a su director (y mí consejero de estudios) Eduardo, que me brindaron un espacio de formación metodológica que considero fundamental para un profesional en la academia.

En cuarto lugar, a mis amigos Anahí, Claudio, Joaquín, Mauro, Pedro, Ramona y Soledad, porque no solo de *negotium* se vive. Por último, cabe destacar que el presente trabajo fue desarrollado con una beca doctoral otorgada por el CONICET.

Índice

Capítulo 1. Introducción

El proyecto de elaborar una metaciencia o ciencia de la ciencia tiene una larga tradición en filosofía de la ciencia. Desde sus inicios a principios del siglo XX con los empiristas lógicos, la filosofía de la ciencia buscó (junto con otras disciplinas) explicar diversos aspectos del quehacer científico (Carnap, 1934; Hahn et al., [1929] 2002). En particular, una preocupación central ha sido la de elaborar una metateoría, esto es, una teoría acerca de las teorías científicas.

Algunos historiadores que estudiaron al Círculo de Viena han sostenido recientemente que los empiristas lógicos veían a la metateoría como dividida en dos grandes partes: por un lado estaría la lógica de la ciencia y por otro la pragmática de la ciencia (Lorenzano, 2011; Uebel, 2013, 2015). En este proyecto bipartito, la lógica de la ciencia consistiría en el análisis formal de las teorías (en sus aspectos sintáctico y semántico), utilizando las herramientas de la lógica matemática, para esclarecer el contenido cognitivo de las teorías. Esto no solo tendría consecuencias prácticas inmediatas (como el mejoramiento de la educación y la comunicación científicas), sino que además permitiría delimitar el ámbito de otro tipo de discusiones que, bajo esta concepción de la filosofía de la ciencia, caerían bajo la dimensión pragmática de la ciencia. Entre ellas se encontrarían, por ejemplo, cuestiones (prácticas o éticas) que, a pesar de no tener contenido cognitivo, podían tener influencias (positivas o negativas) sobre el curso de los acontecimientos (Frank, 1951; Hahn et al., [1929] 2002).

Como es bien sabido, el empirismo lógico como programa de investigación fracasó y fue abandonado. Esto se debió, en parte, a dificultades internas a su marco teórico. Por ejemplo, sus proponentes nunca pudieron aplicar de manera precisa el criterio de teoricidad que sostenían (véanse Hanson, 1958; Hempel, 1970; Kuhn, 1962; Lewis, 1970; Putnam, 1962; Sneed, 1971). Por otro lado, el fracaso del empirismo lógico se debió también a que el contexto político de la Guerra Fría no era favorable al tipo de objetivos políticos que los empiristas lógicos perseguían como meta última de sus estudios (Howard, 2003, 2009; McCumber, 1996; Reisch, 2005; véanse también Douglas, 2010; Fehr & Plaisance, 2010 para más sobre la revalorización actual de los objetivos políticos del Círculo).

En muchos casos, esto derivó en intentos de hacer filosofía de la ciencia sin un marco metateórico por detrás que la sustente. Sin embargo, en dichos casos, usualmente se terminó apelando (implícita o explícitamente) a las concepciones del empirismo lógico, en el momento en

que se necesitaban utilizar términos metateóricos. Por otra parte, otras corrientes de pensamiento posteriores han retomado el estudio de la estructura lógica de las teorías (Balzer et al., [1987] 2012; Kitcher, 1993; van Fraassen, 1989, entre otros), sin duda influenciadas por la metateoría de los empiristas lógicos, aunque proponiendo aparatos metateóricos que difieren de aquella en muchos sentidos sustanciales.

El presente trabajo de investigación se inscribe dentro de la corriente del estructuralismo metateórico (Balzer et al., [1987] 2012; Díez et al., 2002; Sneed, 1971). La metateoría estructuralista ha sido adoptada y utilizada por investigadores provenientes de diversos lugares del mundo, desde Alemania, España y Estados Unidos, hasta diversos países de Latinoamérica (Argentina, Brasil, México, Colombia), en donde ha tenido un gran desarrollo. A diferencia de la metateoría clásica de los empiristas lógicos, el marco estructuralista ha sido aplicado exitosamente a la reconstrucción de teorías de diversos ámbitos de la ciencia, incluyendo la física y astronomía (Balzer et al., [1987] 2012; Carman, 2010; Lastiri, 2012; P. Lorenzano et al., 2007; Schmidt, 2014; Sneed, 1971), la química y la bioquímica (Alleva, Díez, & Federico, 2017; Caamaño, 2009; Donolo, Federico, & Lorenzano, 2007; Falguera & Donato-Rodríguez, 2016; C. Lorenzano, 2002; O'Lery, 2012), la biología (Balzer & Lorenzano, 2000; Bernabé, 2018; Blanco, 2012; Díaz & Lorenzano, 2017; J. Díez & Lorenzano, 2013; Ginnobili, 2016, 2018; P. Lorenzano, 2002; Mendez & Casanueva, 2006), la sociología y lingüística (Abreu, 2012, 2014; Gonzalo & Balzer, 2012; Peris-Viñé, 2011), entre otros.

Parte de la ventaja de trabajar dentro de un marco metateórico que está genuinamente constituido como un programa de investigación o paradigma es que es posible hacer "ciencia normal" en su interior. Esto es, dado que las bases del programa metateórico se encuentran relativamente asentadas (aunque podrían ser abandonadas en algún momento, como la historia de la ciencia atestigua que ha ocurrido con una gran cantidad de programas que se creían bien establecidos), es posible —al menos durante el tiempo que programa dure— aplicar esas bases para extender los alcances del programa y así construir acumulativamente sobre el conocimiento ya existente. Con cada nueva reconstrucción de una teoría utilizando las herramientas del estructuralismo se enriquece el campo de aplicación de este. Por otro lado, el hecho de trabajar dentro de un programa no implica que los postulados metateóricos no puedan alterarse o extenderse. Como ocurre con toda otra teoría, las aplicaciones en nuevos lugares muchas veces

sugieren desarrollos novedosos, y (como se verá más adelante) esta tesis apunta a ser especialmente rica en ese sentido.

Uno de los objetivos centrales de la tesis es proveer herramientas computacionales que permitan testear reconstrucciones hechas al seno de esta metateoría. La relevancia de este punto debería ser inmediatamente clara para cualquiera que posea una actitud científica hacia los estudios metateóricos o metacientíficos. Hasta el presente, el testeo de reconstrucciones era llevado a cabo mayormente "a ojo". Es decir, a pesar de que el estructuralismo tiene un alto grado de desarrollo conceptual y una rica variedad de aplicaciones a casos, no se contaba hasta el momento con ningún modo sistemático y exacto de testear la adecuación de esas aplicaciones (ni, por tanto, para testear a la metateoría misma, ya que las teorías —incluyendo las teorías acerca de las teorías— se testean a través de sus aplicaciones). En cualquier otra área de la ciencia, este estado de cosas sería considerado altamente problemático, y no hay razón para pensar que el estudio (meta)científico de las teorías científicas debería estar exceptuado del testeo riguroso de sus productos (meta)teóricos. Por tanto, proporcionar tanto procedimientos como herramientas para llevar esto a cabo es relevante para volver al estudio de las teorías más plenamente científico.

Con respecto al uso de herramientas computacionales en el estructuralismo metateórico, pueden citarse algunos escritos en donde se intenta trazar alguna conexión entre estas dos áreas. Por ejemplo, un artículo (Wajnberg, Corruble, Ganascia, & Moulines, 2004) en el cual se proponen heurísticas basadas en dicha metateoría para automatizar tareas en el ámbito del *descubrimiento* científico y la postulación de regularidades empíricamente adecuadas; otro caso es un libro (en alemán, no traducido a otros idiomas) acerca de teorías en ciencias sociales (Manhart, 1995). En el ámbito de las aplicaciones o programas de computación específicamente diseñados para lidiar con reconstrucciones estructuralistas no ha habido, sin embargo, un desarrollo sistemático. En ese sentido, al incluir no solo una novedosa propuesta teórica sino también una aplicación computacional, el presente proyecto llena un vacío importante en esta literatura.

La herramienta computacional que introduciré es un programa de computación, llamado Reconstructor, el cual está disponible para ser descargado gratuitamente en mi sitio web (https://sites.google.com/view/ariel-roffe/software; se utilizará aquí la versión 2.0). Gran parte del esfuerzo realizado para la elaboración de la presente tesis, y gran parte de su contenido, se encuentran fuera de la tesis: están en miles y miles de líneas de código de ese programa. Espero que el lector pueda apreciar ese hecho y esfuerzo, y sepa disculpar el tono mayormente expositivo

de algunos de los capítulos (especialmente del capítulo 4, en donde se presenta toda la variedad de funcionalidades que el programa incluye).

El otro objetivo central de la tesis es presentar una reconstrucción estructuralista de una teoría particular, a saber, la cladística (Hennig, 1950, 1966; Kitching, Forey, Humphries, & Williams, 1998; Nelson & Platnick, 1981; Wiley & Lieberman, 2011), una teoría proveniente de la sistemática biológica. El análisis de esta teoría es relevante en cuanto a la ampliación del campo de aplicaciones de la metateoría ya que, en la literatura de filosofía de la biología, se ha prestado mucha mayor atención a áreas como la microevolución y a teorías como la genética de poblaciones, que a otras igualmente importantes como la sistemática, y el estudio de las homologías y la ancestría común.[1]

Asimismo, la reconstrucción me permitirá contribuir a la solución de diversos problemas conceptuales que surgieron al interior de la sistemática. Según Kuhn, los científicos se preocupan por la filosofía mayormente en períodos de crisis reconocida (Kuhn, 1970, p. 88), más aun cuando están en una disputa paradigmática con otros científicos. Afortunadamente (para mí), desde los años 60 del siglo XX hasta la actualidad la sistemática ha sido un campo de batalla continuo entre distintos programas de investigación. Quizás por ello los sistemáticos suelen ser especialmente receptivos a la filosofía dentro de la comunidad científica. Autores como Popper, Hull, Ghiselin, Brady y Sober son ampliamente leídos y citados. Hull y Ghiselin incluso han formado parte de sociedades científicas y comités editoriales de revistas del área. Y no solo eso, sino que muchos sistemáticos se han vuelto estudiosos de la filosofía y han producido textos filosóficos de notable calidad. Una parte no despreciable de la literatura citada en la presente tesis consiste en textos filosóficos escritos por sistemáticos y publicados en revistas especializadas de sistemática. Esta posibilidad de intercambio fructífero con los científicos practicantes de la disciplina constituye uno de los aspectos más interesantes de trabajar en el área de la filosofía de la sistemática. Adicionalmente, la matematización y computarización de la sistemática en este período derivó en diferentes estudios formales de la cladística, por parte de sistemáticos, de matemáticos y de personas provenientes de las ciencias de la computación. La reconstrucción propuesta en el capítulo 3 edifica sobre lo construido por estos autores.

[1] Compárense, por ejemplo, el número de artículos listados en PhilPapers bajo las categorías *Homology* (96) y *Phylogenetic Inference* (43) con las categorías de *Natural Selection* (462) y *Fitness* (173).

Los dos grandes objetivos de la tesis se vinculan ya que la reconstrucción de la cladística será utilizada como caso de aplicación de la herramienta computacional, ilustrando de ese modo la fertilidad de este último. La aplicación de las herramientas computacionales al caso de estudio sugerirá, a su vez, nuevos desarrollos para la propia metateoría. Dos ejemplos de ello serán una elucidación alternativa de la noción de método de determinación y la utilización de modelos incompletos para representar casos de falta de información. Con respecto al primero de estos puntos, se brindará una elucidación de aquella noción basada en el concepto de algoritmo (en lugar de entenderla como una clase de modelos, como es estándar actualmente en el estructuralismo). Se argumentará que ello tiene varias ventajas, una de las cuales es justamente la posibilidad de ejecutar métodos de determinación desde Reconstructor para automatizar la tarea de extraer predicciones teóricas. En cuanto a los modelos incompletos, se mostrará como Reconstructor es capaz de utilizarlos, junto con una semántica trivaluada no-clásica, para evaluar la satisfacción de las leyes en situaciones de información incompleta. Si bien las lógicas no clásicas del tipo que presentaré ya habían sido utilizadas con propósitos similares en otras áreas (véanse Cobreros et al., 2014; Priest, 2008), nunca habían sido aplicadas en detalle en el marco de la filosofía de la ciencia y la metateoría.

La estructura de la presente tesis se divide en tres partes. La parte I, que consta de dos capítulos, introduce la teoría que conformará el caso de estudio de la tesis, a saber, la cladística. En el capítulo 2 se hace una presentación de la cladística, resaltando algunos de sus antecedentes, la historia de su surgimiento y las escuelas rivales con las que se enfrentó y continúa enfrentando hoy. Se hace además una presentación informal del contenido de la teoría, partiendo desde la ancestría común darwiniana, pasando por la obra de Hennig y sus aportes fundamentales, y mencionando finalmente algunos desarrollos contemporáneos de este programa de investigación.

El capítulo 3 expone mi reconstrucción formal de esta teoría. Para ello, se introducen a la vez algunas nociones del estructuralismo metateórico y se formalizan conceptos filogenéticos como los de árbol filogenético, caracteres y estados, asignaciones de caracteres a los nodos del árbol, largo de un cladograma, etc. Una breve subsección discute el estatus fáctico de la cladística a la luz de lo que será presentado como sus leyes fundamentales. Se sostiene que la teoría es difícil de contrastar en casos reales porque, en general, los eventos evolutivos relevantes ocurrieron en el pasado remoto, de modo que no hay modo independiente de saber cuál es el árbol real ni los estados

reales de los ancestros. Por ese motivo, la cladística es en general usada para retrodecir fenómenos. Sin embargo, se examinan algunos casos en los que la teoría fue efectivamente contrastada con filogenias experimentales. Finalmente, se introducen algunos conceptos adicionales que, si bien no juegan un rol en las aserciones fácticas centrales de la teoría, son ampliamente usados, y que por tanto resulta valioso expresar formalmente con mayor claridad. Entre ellos destacan los distintos tipos de filetismo entre grupos y los tipos de morfismos entre estados de caracteres.

La segunda parte, que contiene dos capítulos, introducirá la metodología para contrastar reconstrucciones formales, el software en el que dicha metodología puede ser llevada a cabo (capítulo 4) y la aplicación de este software a la teoría objeto (capítulo 5). El primero de estos capítulos introduce el programa Reconstructor. Se presentan los diversos módulos, su funcionalidad y su modo de uso, ejemplificando cada uno de ellos con una teoría cuasi-ficticia pero sencilla (una simplificación de la mecánica de Descartes). Por otro lado, la metodología para contrastar reconstrucciones se divide en tres tipos de test distintos. El primer tipo compara el *output* del programa para ciertas operaciones (la satisfacción de oraciones formales en un modelo, la aplicación de métodos de determinación, etc.) con las contrapartes informales que estos procedimientos pretenden representar (la satisfacción de las leyes por parte de las aplicaciones, las mediciones y predicciones de valores de conceptos, etc.) para ver que la reconstrucción "se comporte igual" que la teoría objeto original. El segundo tipo de test funciona por coherencia interna (p.e. el resultado de la ejecución de los métodos determinación debe satisfacer la formalización de las leyes). Por último, un tercer método se basa en la utilización de métodos de determinación automáticos, en los que Reconstructor infiere automáticamente el valor de ciertos conceptos a partir de las leyes. Esto será explicado en mayor detalle en el capítulo 4. En todos los casos, se ejemplifica como Reconstructor es capaz de llevar a cabo estos procedimientos.

En el capítulo 5 se aplica la herramienta introducida en el capítulo 4 a la reconstrucción presentada en el capítulo 3. Se muestra, en primer lugar, cómo cargar la reconstrucción en el programa axioma por axioma. Se discuten algunos de los desafíos y ajustes que fue necesario hacer para llevar esto a cabo, tanto por motivos de computabilidad como de que el lenguaje formal utilizado corrientemente por los estructuralistas no suele ser totalmente explícito. Se muestra además cómo representar distintas aplicaciones posibles de la cladística como modelos, y cómo la reconstrucción propuesta, cargada en el programa, pasa todos los tests mencionados en el párrafo anterior.

Finalmente, una tercera parte, que abarca dos capítulos, extraerá algunas consecuencias del análisis anterior (capítulo 6), y brindará algunas conclusiones finales, examinando el trabajo que aún resta por hacer (capítulo 7). Las consecuencias del capítulo 6 se dividen en dos tipos. Por un lado, se extraen conclusiones para la metateoría estructuralista (sección 6.1), centradas especialmente en la noción de *método de determinación*. Se propondrá aquí la elucidación alternativa de la esta noción, mencionada arriba. Por otro lado, las conclusiones para la cladística se centran en las consecuencias que la reconstrucción formal tiene para debates conceptuales y filosóficos en su interior, especialmente aquellos relacionados con el concepto de homología. Se examinará el problema clásico de la circularidad en torno a este concepto y cómo este reaparece en el marco de la cladística. Se mostrará además cómo, históricamente, el debate sobre las homologías se dio en el marco de la discusión sobre el cladismo de patrón. Las últimas subsecciones de 6.2. defienden la concepción sobre las homologías de los cladistas de patrón de acusaciones (de naturaleza filosófica) acerca de la supuesta relación que presuponen entre teoría y observación.

PARTE I - La teoría objeto

Capítulo 2. La cladística

En este capítulo se hace una presentación de la cladística, resaltando algunos de sus antecedentes, la historia de su surgimiento y las escuelas rivales con las que se enfrentó y continúa enfrentando hoy. Se hace además una presentación informal del contenido de la teoría. El objetivo es introducir al lector a la teoría que será objeto de análisis en los capítulos posteriores. De ese modo, si bien tal introducción será dada con algún nivel de detalle, algunos temas más complejos y algún detalle histórico quedarán por fuera del alcance de esta tesis. Para presentaciones históricas más completas de la teoría pueden verse Hull (1988) y Rieppel (2016). En cuanto a su contenido, pueden consultarse diversos manuales en la literatura, como ser, Kitching et al. (1998), Nelson y Platnick (1981), Wiley y Lieberman (2011), entre otros.

La estructura del capítulo será la siguiente. En la primera sección, introduciré algunas consideraciones sobre la ancestría común darwiniana, su *explanandum* (las homologías) y el modo como da cuenta de ellas. Esto es relevante ya que tanto las homologías como la ancestría común forman parte del marco conceptual de la cladística contemporánea (como parte de su *explanandum* y *explanans*, respectivamente). La sección siguiente realiza un breve raconto histórico del surgimiento de la cladística y establece una distinción importante entre la propuesta cladista en taxonomía y su método de reconstrucción filogenética. En tercer lugar, se introducen los contenidos del método de inferencia filogenética en algún detalle. Por último, se extraen algunas conclusiones.

2.1. La ancestría común darwiniana

2.1.1. El marco conceptual darwiniano

El entramado conceptual propuesto por Darwin en *El origen de las especies* (1859, [1872] 1992) involucra diferentes elementos, que no siempre han sido claramente diferenciados. Estos elementos son, en distinta medida, conceptualmente independientes entre sí, y tuvieron diferente grado de aceptación y diferente desarrollo en la historia de la biología (pre- y post-darwiniana). Entre ellos pueden mencionarse a:

- Evolución

La evolución es la tesis de que las especies cambian. Esto es, de que los descendientes de un individuo *I*, perteneciente a una especie *E*, pueden experimentar modificaciones en grado tal que deba dejar de considerárselos como individuos de la misma especie que la original. Darwin no fue, por supuesto, el primer evolucionista. Por ejemplo, autores como Lamarck (1809) ya habían propuesto que las especies evolucionaban aunque por medio de diferentes mecanismos y manteniendo la idea aristotélica de la escala natural.

- Selección natural

La selección natural era para Darwin el principal (aunque no el único) *mecanismo* evolutivo. Es decir, la selección explica *cómo* o a través de qué proceso los individuos se modifican a lo largo de las generaciones produciendo el cambio evolutivo.

Una característica importante de la selección natural es que produce cambios *adaptativos* en la progenie. Es decir, el *explanandum* de la teoría de la selección natural es la presencia de rasgos adaptativos. Cabe notar que el concepto de adaptación que propuso Darwin[2] es relativo a un ambiente. Es decir, un rasgo es adaptativo cuando permite a los organismos que lo portan resolver un problema ambiental de una manera efectiva (un modo de operacionalizar esto es a través de los llamados modelos de optimalidad, véanse Ginnobili & Roffé, 2017; Roffé & Ginnobili, 2019a). De ese modo, un rasgo que es adaptativo para un organismo en un ambiente podría ser ineficiente, o incluso perjudicial, en otro ambiente. En consecuencia, no hay, para Darwin, un patrón objetivo mediante el cual comparar el grado de "perfección" de especies que viven en ambientes distintos, ni por tanto una escala lineal que permita ordenar a los seres vivos desde los organismos inferiores hasta (presumiblemente) el ser humano en la cima.

El modo en el que la teoría de la selección natural explica la presencia de rasgos adaptativos es a través de la iteración de un proceso que opera en dos generaciones (Ginnobili, 2014, 2018, llama selección natural ahistórica a este proceso, e histórica a su iteración). El principio o ley que gobierna este proceso ahistórico (a veces llamado el Principio de la selección natural) afirma algo como lo siguiente:

[2] Y que, por otro lado, solapa en muchos casos con el concepto de adaptación que presuponían los teólogos naturales británicos como Paley (1809); véase Ginnobili (2013) para más sobre este punto.

> *Los organismos que poseen un rasgo que desarrolla una función de manera más eficiente que organismos que no lo poseen son más aptos, y tienden a tener, en consecuencia, si el rasgo es heredable, un mayor éxito reproductivo diferencial que aquellos organismos de la población que no portan el rasgo.* (Ginnobili, 2016, p. 18, traducción propia)

Un ejemplo paradigmático de aplicación de este principio sería el caso del cuello de las jirafas (Darwin [1872] 1992, pp. 177-178). Este rasgo es adaptativo, ya que permite a las jirafas solucionar un problema ambiental con el que se enfrentan en su hábitat: que las hojas de las acacias de las cuales se alimentan se encuentran a gran altura. La explicación seleccionista de cómo la población de jirafas adquirió este rasgo sería la siguiente. En alguna generación anterior, por azar, algún subconjunto de la población de jirafas nació con un cuello un poco más largo que las demás. Estas jirafas podían alimentarse mejor de las ramas altas (i.e. desarrollaban una función, la alimentación, de manera más efectiva), lo cual les permitía sobrevivir más (i.e. eran más aptas) y por tanto tener un mayor éxito reproductivo que las jirafas de cuellos más cortos. Por tanto, dado que el largo del cuello es un rasgo heredable, el porcentaje de jirafas con cuello largo aumentó en la progenie de esta población de jirafas, en esa generación. Si se itera este proceso a lo largo de muchas generaciones se culmina con una población de jirafas de cuellos largos.

- Origen común

Si bien la selección natural es la innovación conceptual de Darwin más conocida, y aquella con la que más frecuentemente se lo asocia, no fue la única que este autor propuso, ni la que consideraba la más importante. En efecto, Darwin también propuso que toda la rica diversidad de los seres vivos que observamos en la actualidad deriva de un único ancestro común. De ese modo, la historia de la vida en la tierra podría representarse por medio de un diagrama con forma de árbol: partiendo de una única raíz (una única especie ancestral), diversas ramas fueron surgiendo en diferentes direcciones (i.e. la misma especie original dio lugar a diferentes especies descendientes), las cuales a su vez continuaron ramificándose, etc.

El modo en el que una especie progenitora puede dar lugar a dos (o más) especies descendientes se relaciona, según Darwin, con la selección natural. Si una población de organismos de una especie queda dividida en dos (p.e. por el surgimiento de alguna barrera geográfica en la región que la población ocupaba) y luego tales poblaciones son sujetas a presiones selectivas diferentes entonces, al evolucionar por selección para adaptarse a los diferentes ambientes, las

poblaciones comenzarán a divergir. Este proceso daría lugar primero a variedades distintas de la misma especie y, con el paso del tiempo, a especies (e incluso géneros, familias, etc.) distintas.[3]

Si bien la selección natural y el origen común se relacionan en Darwin de este modo (la selección brinda el mecanismo de especiación), como se dijo anteriormente, las dos tesis son conceptualmente independientes. Es decir, si bien la selección es compatible con el origen común, también lo es con varios orígenes separados para diferentes especies actuales. Por otra parte, el origen común también es conceptualmente compatible con otros mecanismos evolutivos no seleccionistas (p.e. el uso y desuso lamarckianos). Adicionalmente, la selección por sí sola no brinda evidencia a favor de ninguna de estas dos alternativas (origen común y orígenes separados).[4] La evidencia para el origen común proviene de otro ámbito, de la morfología comparada y el estudio de las homologías (Blanco, 2012). En la subsección siguiente se introducen estos puntos en mayor detalle.

2.1.2. Las homologías y el origen común

Para comprender los motivos por los que Darwin postula el origen común es conveniente comenzar exponiendo lo que la tesis pretende explicar, es decir, aquello de lo que pretende dar cuenta. Lo que el origen común explica no son los rasgos adaptativos, sino los rasgos homólogos. En palabras de Darwin:

> *Hemos visto que los miembros de una misma clase, independientemente de sus costumbres, se parecen en el plan general de su organización. Esta semejanza se expresa frecuentemente por el término* unidad de tipo *o diciendo que las diversas partes y órganos son homólogos en las distintas especies de la clase. Todo el asunto se comprende con la denominación general de Morfología. Es ésta una de las partes más interesantes de la historia natural, y casi puede decirse que es su verdadera esencia. ¿Qué puede haber más*

[3] Los ejemplos paradigmáticos de especiación que introduce Darwin (p.e. los pinzones de Galápagos) siguen este patrón explicativo. Sin embargo, cuando Darwin teoriza acerca del mecanismo de especiación apela no a la selección direccional actuando independientemente en dos poblaciones, sino al llamado "principio de divergencia" —i.e. la idea de que, en una misma población, es ventajoso alejarse de la media poblacional ya que de ese modo se ocupan nichos en los que hay menos competencia (Darwin, 1859, 1872; Roffé & Ginnobili, 2019b)

[4] Si bien puede brindar evidencia de que las diferentes especies de pinzones de las Galápagos provienen de un ancestro común, no ocurre lo mismo con especies más lejanas entre sí (p.e. un pinzón y un elefante).

curioso que el que la mano del hombre, hecha para coger; la del topo, hecha para minar; la pata del caballo, la aleta de la marsopa y el ala del murciélago, estén todas construidas según el mismo patrón y encierren huesos semejantes en las mismas posiciones relativas? (Darwin, [1872] 1992, pp. 569-570, énfasis en original).

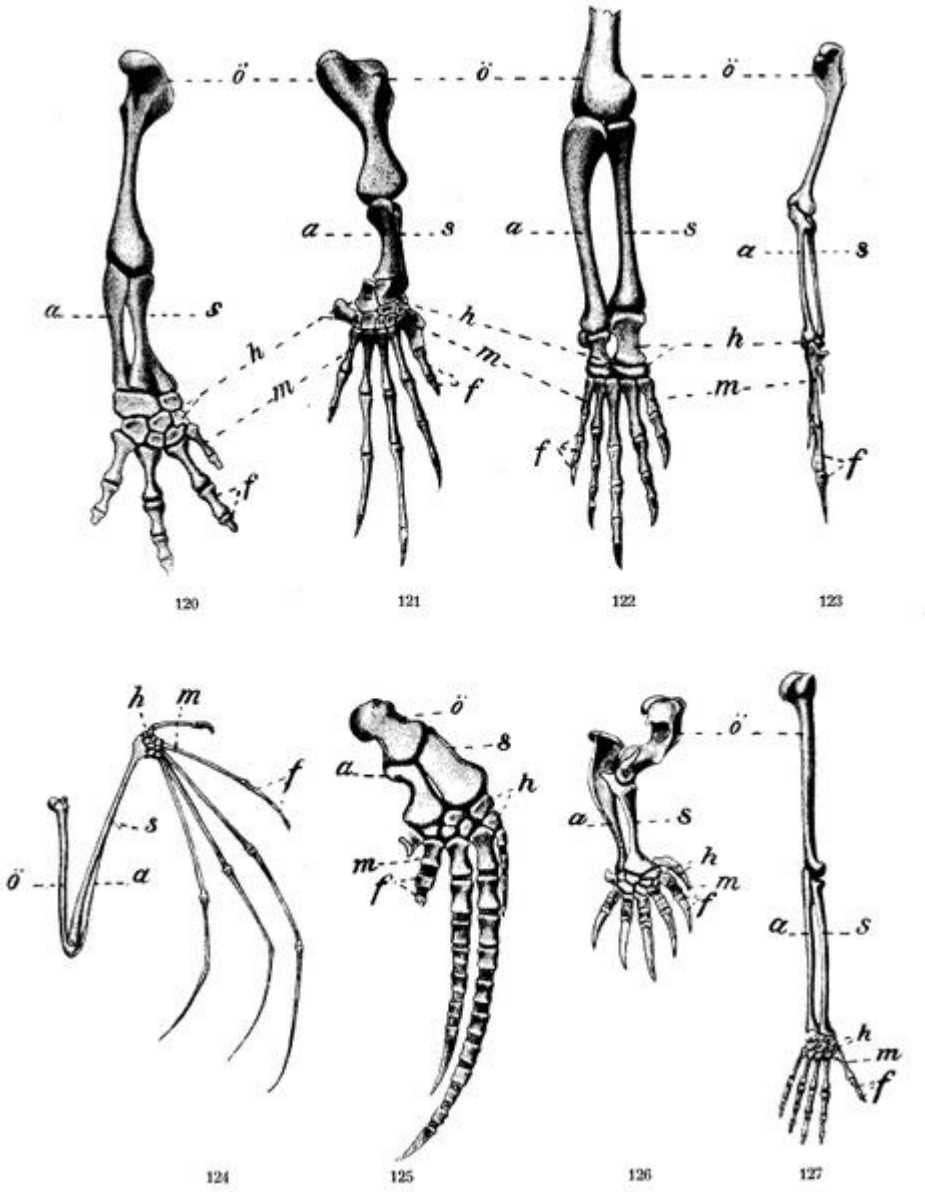


Fig. 2.1. Homología en las extremidades de distintas especies de vertebrados: salamandra (120), tortuga marina (121), cocodrilo (122), ave (123), murciélago (124), ballena (125), topo (126), humano (127). Fuente: https://commons.wikimedia.org/wiki/File:Arm_skeleton_comparative_NF_0102.5-2.png

Otros ejemplos que brinda el autor de este tipo de rasgos son los siguientes:

> *Vemos esta misma gran ley en la construcción de los órganos bucales de los insectos: ¿qué puede haber más diferente que la proboscis espiral, inmensamente larga, de un esfíngido; la de una abeja o una chinche, curiosamente plegada, y los grandes órganos masticadores de un coleóptero? Sin embargo, todos estos órganos, que sirven para fines tan diferentes, están formados por modificaciones infinitamente numerosas de un labio superior, mandíbulas y dos pares de maxilas. La misma ley rige la construcción de los órganos bucales y patas de los crustáceos. Lo mismo ocurre en las flores de las plantas.* (Darwin, [1872] 1992, pp. 570-571).

El concepto de homología y los métodos para reconocerlas habían sido propuestos previamente por Geoffroy Saint-Hilarie (1830, quien usaba el término "analogías filosóficas" para referirse a ellas; véase Caponi, 2015) y luego teorizado por Richard Owen (1848, 1849), entre otros. Para Owen, la presencia de este tipo de similaridades estructurales (p.e. en las extremidades de los vertebrados) se explicaba apelando a que estaban construidas a partir del mismo *arquetipo*. Adicionalmente, Owen concebía al arquetipo como una idea en la mente divina, de modo que el conocimiento de los arquetipos brindaba un entendimiento de algo semejante a los "planos arquitectónicos" que Dios había utilizado para crear a los seres vivos (la causa formal en términos aristotélicos).

Lo que Darwin hizo fue encarnar al arquetipo de Owen: este sería un ancestro (véase la cita siguiente). De ese modo, la homología entre la extremidad de los vertebrados se explicaría no por compartir algún plano de construcción abstracto, sino porque el ancestro de todos ellos poseía una versión de tal extremidad. Según las necesidades impuestas por el ambiente la extremidad se habría ido modificando de distinto modo en cada rama del árbol evolutivo, preservando, sin embargo, la estructura general. Nuevamente cabe citar en alguna extensión a Darwin, quien explica la idea con suma claridad:

> *La explicación es bastante sencilla, dentro de la teoría de la selección de ligeras modificaciones sucesivas, por ser cada modificación provechosa en algún modo a la forma modificada; pero que afectan a veces, por correlación, a otras partes del organismo. En cambios de esta naturaleza habrá poca o ninguna tendencia a la variación de los planes primitivos o a transposición de las partes. Los huesos de un miembro pudieron acortarse y aplastarse en cualquier medida, y ser envueltos al mismo tiempo por una membrana gruesa para servir como una aleta; o en una membrana palmeada pudieron todos o algunos huesos*

> *alargarse hasta cualquier dimensión, creciendo la membrana que los une de manera que sirviese de ala; y, sin embargo, todas estas modificaciones no tenderían a alterar el armazón de huesos o la conexión relativa de las partes.*
>
> *Si suponemos que un remoto antepasado —el arquetipo, como puede llamársele— de todos los mamíferos, aves y reptiles tuvo sus miembros construidos según el plan actual, cualquiera que fuese el fin para el que sirviesen, podemos desde luego comprender toda la significación de la construcción homóloga de los miembros en toda la clase.* (Darwin, [1872] 1992, pp. 571-572).

Cabe aclarar sin embargo que, dada la evidencia con la que contaba, Darwin fue prudente en su postulación de un único origen común. Lo limitó con seguridad a cuatro o cinco especies dentro de los animales y a un número similar para las plantas, y solo especuló acerca de un único origen común entre todos los organismos. En sus palabras:

> *Cabe preguntarse hasta donde hago extensiva la doctrina de la modificación de las especies. (...) [N]o puedo dudar de que la teoría de la descendencia con modificación comprende todos los miembros de una misma clase o de un mismo reino. Creo que los animales descienden, a lo sumo, de sólo cuatro o cinco progenitores y las plantas de un número igual o menor. La analogía me llevaría a dar un paso más. O sea, a creer que todos los animales y plantas descienden de un solo prototipo.* (Darwin, [1872] 1992, pp. 631-632)

2.1.3 Darwin y la taxonomía

El reconocimiento de que el origen común explica la presencia de homologías tenía, para Darwin, profundas implicancias sobre la taxonomía. Las implicancias apuntaban, según este pensador, no tanto al modo de construir una taxonomía, sino más bien al modo de entenderla o interpretarla. Lo que los naturalistas habían estado haciendo, sin saberlo, era agrupar a los organismos con un criterio genealógico:

> *Todas las anteriores reglas y medios y dificultades en la clasificación pueden explicarse, si no me engaño mucho, admitiendo que el sistema natural está fundado en la descendencia*

> *con modificación; que los caracteres que los naturalistas consideran como demostrativos de verdadera afinidad entre dos o más especies son los que han sido heredados de un antepasado común, pues toda clasificación verdadera es genealógica; que la comunidad de descendencia es el lazo oculto que los naturalistas han estado buscando inconscientemente, y no un plan ignorado de creación o el enunciado de proposiciones generales al juntar y separar simplemente objetos más o menos semejantes.* (Darwin, [1872] 1992, pp. 551-552).

Cabe recordar además la explicación darwiniana de la especiación. Puesto que Darwin creía que las variedades, si continúan evolucionando independientemente, se convierten eventualmente en especies distintas, y luego en géneros, familias, etc. distintas, una clasificación de los organismos en esas categorías debería reflejar esas relaciones de ancestría.

Sin embargo, una vez más, la conexión entre el reconocimiento de la ancestría común y la idea de una taxonomía basada en las relaciones filogenéticas no es una de implicación lógica. En el período en el que Darwin publicó las sucesivas ediciones de *El origen* no se contaba aun con un método sistemático para identificar las relaciones filogenéticas a partir de las distribuciones "observadas" de homologías. De ese modo, algunos de su colegas (p.e. Huxley) aceptaron la primera propuesta (la ancestría común) pero rechazaron la segunda (la de basar la taxonomía en las relaciones de ancestría), sobre la base de que la descendencia "no era cognoscible con suficiente certeza como para permitir que ese tipo de consideraciones tengan lugar en algo tan importante como la taxonomía" (Hull, 1988, p. 98, traducción propia).[5]

La aparición de tal tipo de herramientas sistemáticas tuvo que esperar a la segunda mitad del siglo XX con el surgimiento de la cladística (entre otras). Como se verá en la siguiente sección, un motivo importante por el que el programa cladista tuvo éxito en taxonomía con su propuesta de basar la taxonomía *exclusivamente* en las relaciones de hermandad entre grupos fue que propuso un método para retrodecir con exactitud esas relaciones, así como para resolver los conflictos entre los agrupamientos incompatibles que sugerían diferentes caracteres homólogos. Tal método o procedimiento es descripto en las secciones siguientes.

[5] Hull (1988, p. 373) relata las posteriores discusiones, completamente anacrónicas, acerca de si debía considerarse a Darwin un cladista o un evolucionista en taxonomía.

2.2. El surgimiento de la cladística

2.2.1. Periodización

El nacimiento del programa cladista puede datarse en 1950 con la publicación del libro *Grundzüge einer Theorie der phylogenetischen Systematik* del entomólogo alemán Willi Hennig. En tal obra, Hennig propuso algunos de los pilares de lo que sería el programa de investigación cladista (cabe notar que Hennig no usa nunca el término "cladista", véase más adelante). Entre esos pilares figuran la idea de representar únicamente la relación de hermandad entre grupos en las clasificaciones, así como un método para inferir esa relación (los cuáles se explicarán en los apartados siguientes).

Sin embargo, inicialmente, la publicación de este libro tuvo mínimo impacto en la comunidad de los sistemáticos. Hull (1988, pp. 131-132) hipotetiza que esto se debió a varios factores. Para comenzar, se debió a que el libro estaba escrito en idioma alemán, siendo que muchos investigadores de los Estados Unidos (el epicentro de la sistemática en ese período) no hablaban ese lenguaje. Adicionalmente, el estilo de escritura de Hennig era difícil de seguir, incluso para un hablante nativo. Por otra parte, Hull describe a Hennig como una persona tímida y no muy al tanto de lo que estaba ocurriendo en el ámbito estadounidense, en donde la taxonomía numérica (o escuela fenetista) y la escuela evolucionista estaban teniendo ya grandes disputas. Por estos motivos, Hull describe el encuentro entre Hennig y Sokal (el padre de la taxonomía numérica) en 1964 como poco fructífero.

En 1965, Ernst Mayr (uno de los principales proponentes y defensores de la escuela evolucionista en taxonomía, quien podía leer alemán) publicó un artículo criticando tanto a la taxonomía numérica como al enfoque de Hennig (Mayr, 1965). Fue Mayr, en aquel artículo, quien designó "escuela fenetista" y "escuela cladista" a sus rivales. Hennig había llamado "sistemática filogenética" a su enfoque; sin embargo, a Mayr le parecía confundente tal designación, ya que consideraba que su escuela (la evolucionista) también prestaba atención a la filogenia. Lo distintivo del cladismo, según Mayr, es que prestaba atención únicamente a la división de linajes (cladogénesis) y ninguna a la evolución interna de los linajes una vez separados (anagénesis), un proceso que consideraba igualmente importante en la evolución; de ahí el nombre "enfoque cladista" (*cladistic approach*). Según Hull, inicialmente, los cladistas estaban descontentos con tal designación pero luego terminaron adoptándola (Hull, 1988, p. 133).

Ese mismo año, y 15 años después de la publicación de su libro, Hennig publicó un resumen de los contenidos del libro en inglés (Hennig, 1965). La traducción completa del libro al inglés (en realidad, de una revisión del libro), titulada *Phylogenetic Systematics*, tuvo que esperar un año más (Hennig, 1966). Hull (1988, p. 134) relata que los traductores al inglés (Davis y Zangerl) tuvieron mucha dificultad con la prosa de Hennig y que (posiblemente por ese motivo) decidieron no solo traducir sino también editar fuertemente los contenidos del libro, eliminando lo que veían como pasajes repetitivos, simplificando oraciones y clarificando las ideas. Sin embargo, los traductores estaban formados en la tradición de la morfología idealista alemana (el principal enemigo de Hennig en su libro), de modo que su formación puede haber introducido algún sesgo o distorsión en la edición del libro. El propio Hennig no pudo colaborar demasiado con la traducción ya que en ese momento se encontraba en proceso de escapar desde Alemania del Este hacia Alemania del Oeste.

En consonancia con esto, la influencia directa de la traducción al inglés del libro de Hennig tampoco fue un factor causal relevante en la formación de un programa de investigación cladista, como cabría esperar *a priori*. En cambio, las ideas de Hennig se difundieron inicialmente a través de una monografía del entomólogo sueco Lars Brundin (1966), la cual aplicaba los principios hennigianos al estudio de la distribución geográfica de la familia de los quironómidos (*Chironomidae*). Esta monografía fue leída por un joven investigador estadounidense, Gareth Nelson, en una visita a Estocolmo. Nelson quedó inicialmente muy impresionado por la monografía, y discusiones posteriores con Brundin terminaron convenciéndolo de convertirse al cladismo (los posteriores encuentros entre Nelson y Brundin están narrados en Hull, 1988, p. 144)

Nelson se convertiría más adelante en el campeón de la causa cladista en los Estados Unidos desde el *American Museum of Natural History*. Desde su llegada al museo en 1967 se ocupó primero de convencer a sus colegas (entre ellos Patterson, Greenwood, Bonde, Schaeffer y Rosen) y luego de formar a varios estudiantes de doctorado (Eldredge, Cracraft, Gaffney, Platnick y Wiley, inicialmente) en el programa cladista. Este grupo funcionó de manera cohesiva, promoviendo sus ideas, respondiendo de manera coordinada a críticas externas y tomando el control de la edición de asociaciones y revistas científicas del área (p.e. Nelson fue editor de *Systematic Zoology* —hoy *Systematic Biology*—, posiblemente la revista de sistemática teórica más importante). Según Hull (1988) el accionar coordinado de este grupo fue instrumental en la posterior "victoria" del programa cladista frente a sus rivales fenetistas y evolucionistas.

Por otra parte, el desarrollo matemático (y luego computacional) de la cladística fue obra principalmente de James Farris, un ex colega de Sokal en Nueva York (véanse, p.e. Farris, 1969, 1970; Kluge & Farris, 1969 para algunas de las contribuciones de Farris). Inicialmente Farris parecía pertenecer al campo de la escuela de taxonomía numérica, aunque sus publicaciones versaban en su mayoría sobre métodos matemáticos en inferencia filogenética (cabe notar que el propio Sokal había hecho contribuciones en esta área, véase p.e. Camin & Sokal, 1965). Sin embargo, Farris terminó alineándose totalmente con los investigadores del *American Museum*, convirtiéndose en uno de los más ardientes defensores del programa cladista.

Desde los años 60 y 70 del siglo XX hasta la actualidad, el programa cladista continuó desarrollándose y aplicándose a cada vez más casos. Se han propuesto diversas mejoras y ampliaciones al método para inferir las relaciones de hermandad entre grupos. Entre ellas cabe mencionar a: (*i*) métodos de enraizado de árboles (Maddison, Donoghue, & Maddison, 1984; Nelson, 1978; Nixon & Carpenter, 1993, 1996); (*ii*) medidas de soporte de grupos (Bremer, 1988; Felsenstein, 1985) —para testear la robustez de los resultados obtenidos—; (*iii*) modos de representar la similaridad y el conflicto entre árboles (Adams, 1972; Schuh & Polhemus, 1980); (*iv*) medidas de ajuste de caracteres a árboles (Farris, 1989; Kluge & Farris, 1969); (*v*) métodos de pesado de caracteres (Farris, 1969; Goloboff, 1993b, 2014; Neff, 1986); (*vi*) métodos para la determinación dinámica de las homologías (Ramírez, 2007; Wheeler, 1996, 1999, 2003a, 2003b); y por último, e importantemente, (*vii*) programas de computación capaces de llevar a cabo análisis cladísticos (Felsenstein, 1993; Goloboff, 1994; Goloboff & Catalano, 2016; Goloboff, Farris, & Nixon, 2008; Wilgenbusch & Swofford, 2003). Algunos de estos puntos serán explicados en las secciones 2.3 y 2.4. Cabe destacar también que la Argentina ha sido un foco importante de desarrollo del programa cladista, principalmente a través de los aportes de Pablo Goloboff y de sus discípulos y colegas.

Por último, cabe mencionar que si bien la escuela cladista resultó victoriosa en su disputa frente a las escuelas fenetista y evolucionista, se enfrentó posteriormente (y continúa enfrentándose hoy) a otros programas rivales, particularmente en el dominio de los métodos utilizados para inferir las filogenias. Los cladistas suelen identificarse con los métodos de tipo de parsimonia (Quinn, 2017), explicados en la sección siguiente. Los métodos probabilísticos de inferencia filogenética, máxima verosimilitud y análisis bayesiano (Felsenstein, 1981; Huelsenbeck, Ronquist, Nielsen, & Bollback, 2001), han surgido como alternativas al método de parsimonia, y han ganado mucho

terreno en la actualidad. No me ocuparé en esta tesis de introducir, reconstruir ni evaluar estos métodos probabilísticos (que son de una complejidad conceptual y matemática mayor a la de los métodos de parsimonia), ya que quedan por fuera del alcance de mis objetivos.

Como puede apreciarse a partir de los párrafos anteriores, la cladística como programa de investigación o paradigma en sistemática involucra tesis o propuestas en diferentes planos. En particular, deben distinguirse las tesis o propuestas en taxonomía (en adelante, llamaré cladonomía a este conjunto de propuestas) de los métodos de inferencia filogenética (que llamaré cladística). Esta distinción (que en ocasiones se traza utilizando otros términos) es utilizada principalmente por los rivales de la escuela cladista en taxonomía, ya que en ocasiones desean aceptar la cladística como método legítimo de inferencia filogenética pero rechazar que las clasificaciones deban estar basadas únicamente en información extraída a partir de sus productos (p.e. véanse Ashlock, 1974; Aubert, 2015, pp. 2-3). En los dos subapartados siguientes se introducen ambas propuestas (cladonomía y cladística) por separado.

2.2.2. La cladonomía

La propuesta cladista en taxonomía puede comprenderse más fácilmente si se atiende al contexto en el que surgió, y a los puntos de acuerdo y desacuerdo con las concepciones a las cuales se enfrentó.

Por una parte, el programa cladista acordaba con la taxonomía numérica (o escuela fenetista)[6] en que se debían introducir métodos precisos en la taxonomía, utilizando herramientas matemáticas y computacionales. En esto se oponían a los métodos tradicionales, que se basaban frecuentemente en la "intuición" de los expertos acerca de lo que era relevante, y en criterios poco claros y explícitos para agrupar organismos, frecuentemente distintos en grupos distintos. Adicionalmente, compartían la idea de que se debían utilizar muchos rasgos y no unos pocos, como ocurría tradicionalmente, a fin de evitar sesgos y falsos positivos de asociación. Es decir, en resumen, compartían la idea de que la taxonomía debía volverse más "objetiva", sus métodos más replicables, y en general "más ciencia y menos arte".

[6] El programa fenetista estaba representado principalmente por Sokal, Sneath, Michener, Ehrlich y Rohlf; véase Sokal y Sneath (1963) para un manual escrito en esta tradición; Hull (1988), para un resumen histórico.

Por otra parte, la escuela fenetista creía que una taxonomía debía reflejar solamente la similaridad total (*overall similarity*) entre taxa, y no el curso de la evolución (al cual veían como conocimiento "teórico"[7]). En este punto los cladistas acordaban más con la escuela evolucionista, ya que creían (al igual que Darwin) que una taxonomía debía reflejar la filogenia. La principal diferencia de los cladistas con la escuela evolucionista[8] consistía en que los primeros pretendían representar *únicamente* a la relación de hermandad entre grupos en las clasificaciones. Esta es una relación triádica, que se cumple respecto de tres individuos (especies o taxa de nivel superior) A, B y C cuando A y B comparten un ancestro que no es un ancestro de C. Visto en forma de árbol, eso se daría cuando ocurre lo siguiente:

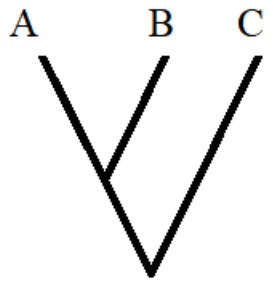


Fig. 2.2. Relación de hermandad ((A, B), C) en forma de árbol. Lo único que importa en este diagrama es la topología del árbol. Los ejes horizontal y vertical no representan nada.

Es decir, lo que los cladistas pretendían representar en sus clasificaciones era *únicamente* el orden de división de los linajes. Esta relación triádica quedaría representada en una taxonomía del siguiente modo:

Familia 1

Género 1

Especie A

Especie B

[7] No es del todo claro qué querían decir con esto. A veces parecen presuponer algo semejante a la distinción teórico-observacional, mientras que otras parecen decir solamente que no había métodos de inferencia filogenética confiables en la época (p.e. véase Sokal, 1962).

[8] Cuyos principales representantes eran algunos de los creadores de la Síntesis evolutiva, como Mayr y Simpson. Véase Simpson (1961) como obra de referencia del pensamiento taxonómico de esta escuela en la época.

Género 2

Especie C

El criterio para pasar de una topología a una clasificación (y viceversa) es que especies que pertenecen al mismo grupo (del nivel taxonómico que sea) compartan un ancestro común que no es un ancestro de ninguna otra especie que esté por fuera del grupo. De ese modo, todo taxón válido será monofilético —lo cual para los cladistas quería decir que incluye a *todos* y *solo* los descendientes de algún ancestro. Por ejemplo, dada la topología anterior, el grupo (A, C) no sería válido ya que excluiría a B, lo cual sería interpretado como afirmando que A y C comparten un ancestro que no es un ancestro de B, afirmación que sería falsa.

Los evolucionistas, por otra parte, pretendían que las clasificaciones reflejaran más información que esta. En particular, dado que veían a la anagénesis como un proceso igualmente importante en la evolución, pretendían reflejar también las relaciones ancestro-descendiente efectivas, así como la similaridad morfológica entre especies. Supóngase, por ejemplo, que el curso efectivo de la evolución de tres especies hubiese sido el siguiente:

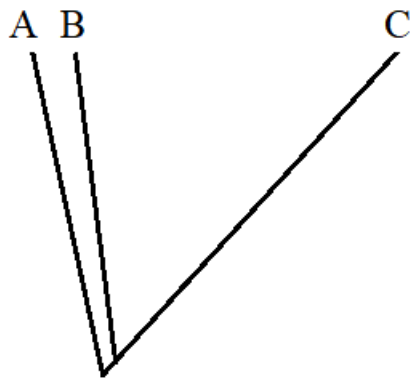


Fig. 2.3. Diagrama representando las relaciones de ancestría (topología del árbol), la proximidad morfológica (eje horizontal), y el tiempo de divergencia (eje vertical) de tres especies.

Aquí el eje horizontal representa a la similaridad morfológica y la topología del árbol al orden de diversificación. El eje vertical representa al tiempo. Así, en este caso, la división entre los linajes A y (B,C) habría ocurrido próxima a la división entre B y C (las tres especies habrían estado evolucionando independientemente por más tiempo); además, C habría divergido más del fenotipo

ancestral, mientras que A y B se habrían mantenido próximos a él. Por estos motivos, un evolucionista podría sostener, por ejemplo, que (A, B) debía conformar un grupo taxonómico que excluya a C (un taxón que para un cladista no sería monofilético). De ese modo, la clasificación se vería igual que la anterior a pesar de que el orden de diversificación es distinto.

Los motivos por los cuales los cladistas pretendían representar en sus clasificaciones únicamente el orden de diversificación y no a relaciones como la ancestría efectiva o la similaridad morfológica son de distinto orden (Hull, 1979). Por un lado estaban los motivos relacionados con los dispositivos representacionales con los que se contaba (particularmente en taxonomía). En particular, el sistema jerárquico linneano no permite representar múltiples criterios simultáneamente (p.e. el orden de divergencia de linajes y la similaridad morfológica). Esto se debe a que ese sistema cuenta únicamente con la subordinación de unos grupos a otros como mecanismo representacional.

Considérese, por ejemplo, a la topología de la figura 2.2 como un escenario evolutivo (interprétese al eje vertical como tiempo y al eje horizontal como similaridad morfológica). Dado que aquí las morfologías están uniformemente distribuidas en el morfoespacio y que la división de linajes ocurrió a intervalos regulares, un evolucionista podría decidir agrupar en base al orden de división de linajes, produciendo la clasificación ((A, B), C). Sin embargo, en el caso de la figura 2.3, dada la morfología muy derivada de C, un taxónomo evolucionista podría pensar que la taxonomía tendría mejor poder predictivo / mayor contenido informacional agrupando a B con A que a B con C, produciendo nuevamente la clasificación ((A, B), C). Así, el agrupamiento ((A, B), C) tendría una lectura ambigua. Podría representar o bien un caso en donde A y B comparten un ancestro que no es un ancestro de C (como en la figura 2.2) o bien un caso en el que B y C comparten un ancestro que no es un ancestro de A, pero en donde A y B son morfológicamente más cercanos (como en la figura 2.3). Es decir, no hay una correspondencia uno a uno entre una clasificación dada y la información evolutiva que se usó en su construcción, de modo que no puede extraerse con seguridad ninguna información de la primera. En otras palabras, la subordinación en una jerarquía linneana debe representar siempre la misma relación para que tal jerarquía pueda ser leída de manera no ambigua.

Todas las escuelas taxonómicas acordaban en que una clasificación debe tener "poder predictivo" y/o constituir un "sistema de almacenamiento de información". Es decir, que ver el lugar en el que un taxón está ubicado en una clasificación debería darnos algún tipo de información

acerca de ese taxón. Para los cladistas, un principio fundamental a tener en cuenta era que la información que se introduce en la elaboración de la taxonomía debe luego poder ser extraída de ella de manera no ambigua. Querer incluir más información en una taxonomía que contiene mecanismos representacionales limitados puede tener el efecto inverso al deseado: volverla menos informativa (para un punto similar relacionado con la ancestría efectiva en lugar de la similaridad morfológica véase Nelson, 1973).

Por supuesto que una alternativa sería continuar sosteniendo que una taxonomía debe contener información tanto de la relación de hermandad entre grupos como de la ancestría efectiva y la similaridad morfológica, y que si el sistema linneano no permite representar estas tres (y quizás otras) relaciones de manera no ambigua entonces hay que reemplazarlo o modificarlo —de hecho los propios cladistas intentaron divisar métodos representacionales adicionales para agregar a la taxonomía linneana (véase nuevamente Nelson, 1973, como un ejemplo). Aquí es donde entran otro tipo de consideraciones, más bien metodológicas o epistemológicas, para quedarse únicamente con la relación indicada arriba. Por ejemplo, respecto de la relación de ancestría efectiva, se sostuvo que (*i*) en una distribución de rasgos la ancestría efectiva (A es un ancestro de B) es imposible de distinguir de una en donde A y B tienen un ancestro común C, pero ni A ni B es ancestro del otro (ambas situaciones se verían igual en una distribución de homologías); y (*ii*) que dado el enorme número de especies que debe haber existido y las condiciones poco frecuentes bajo las que ocurre la fosilización la probabilidad de que un taxa fósil t_1 sea efectivamente un ancestro de otro taxa t_2 del análisis y no una rama extinta que se desprendió del linaje que llevo a t_2 es muy pequeña (Hull, 1979, p. 429).

Tendré algo más que decir sobre estos debates en el capítulo 6. En lo que resta de esta sección me encargaré de introducir el otro componente central del programa cladista, su método para inferir las relaciones filogenéticas.

2.2.3. La cladística

La posición cladista en taxonomía (agrupar únicamente en base a la relación de hermandad entre grupos) se impuso porque, conjuntamente con ella, los cladistas propusieron un único método, claro y operacionalmente aplicable, para determinar tal relación: la cladística. Como explica Hull:

> *Varios de los asistentes [taxónomos numéricos, en una conferencia que tuvo lugar en 1981] presentaron artículos extremadamente sofisticados mostrando la amplia variedad de técnicas formales que existían para agrupar organismos en taxa. Cada técnica tenía sus fortalezas y debilidades. Ninguna era claramente superior. (...) Los cladistas, por el contrario, presentaron un y sólo un método, que afirmaban que era preferible bajo todo aspecto. Para los estudiantes presentes que estaban luchando para producir una tesis en un período limitado de tiempo, la elección era obvia.* (Hull, 1988, p. 256)

La idea es, a primera vista, sencilla. Como se dijo, las homologías eran interpretadas, desde Darwin, como indicativas de la ancestría común. Ahora bien, en general, las homologías se nos presentan en la naturaleza bajo patrones anidados. Por ejemplo, considérese la siguiente distribución de dos caracteres en cuatro especies de arañas del género *Monapia* (datos tomados de Ramírez, 2003):

Taxón / Carácter	Fila de ojos anterior	Mancha negra longitudinal ventral en el abdomen
(1) *Monapia tandil*	Recta	Ausente
(2) *Monapia fierro*	Recta	Presente
(3) *Monapia carolina*	Recta	Presente
(4) *Monapia dilaticollis*	Recurvada	Ausente

Tabla 2.1. Ejemplo de una distribución de dos caracteres en cuatro especies, tomada de Ramírez (2003).

La idea básica es interpretar a este patrón como evidencia de que (1), (2) y (3) comparten un ancestro, que ganó una fila anterior recta, y que no es un ancestro de (4). A la vez, el segundo carácter indica que (2) y (3) comparten un ancestro, que ganó la mancha negra, y que no es un ancestro de (1) y (4).[9] De ese modo, los patrones anidados de homologías permiten resolver el orden de diversificación de los linajes —en este caso, lo que sugieren es la clasificación (4, (1, (3, 2))).

[9] Para simplificar el ejemplo, se está asumiendo que la fila recurvada y la ausencia de mancha son estados ancestrales, y la fila recta y la presencia de mancha derivados, pero ello no tiene que ser necesariamente el caso. Si se asumiera que la fila recta y la mancha son ancestrales y los otros estados derivados, entonces la interpretación filogenética sería distinta. Por ejemplo, en el caso de los ojos, la interpretación sería que el ancestro común de las 4 especies poseía la

Sin embargo, existe una complicación importante. Los patrones anidados de homologías raramente son perfectamente compatibles entre sí en casos interesantes. Por ejemplo, en el caso recién presentado el siguiente carácter también es parte de la matriz original.

Taxón / Carácter	Fila de ojos anterior	Mancha negra longitudinal ventral en el abdomen	Dentículos en la porción prolateral de C2
(1) *Monapia tandil*	Recta	Ausente	Presente
(2) *Monapia fierro*	Recta	Presente	Presente
(3) *Monapia carolina*	Recta	Presente	Ausente
(4) *Monapia dilaticollis*	Recurvada	Ausente	Ausente

Tabla 2.2. Mismo ejemplo de la Tabla 2.1, con carácter adicional.

Nótese que ahora el tercer carácter es compatible con el primero pero conflictivo con el segundo, ya que sugiere agrupar a (2) con (1) en lugar de con (3). Es decir, el primer y el tercer carácter sugieren el agrupamiento (4, (3, (1, 2))). La pregunta se torna, entonces, cómo acomodar la información incompatible provista por los distintos caracteres.

La solución de la cladística consiste en tomar muchos caracteres en consideración (cientos o miles) y preservar al agrupamiento (i.e. inferir como la historia evolutiva correcta) que mejor es soportado por la evidencia total. Por esto se entiende preservar el árbol que requiere postular el menor número de cambios entre estados en total, considerando todos los caracteres. En la sección siguiente, se expone en detalle el método cladístico y el modo en el que esta solución funciona.

2.3. Presentación informal de la teoría

fila recta, y que (4) la perdió, en lugar de interpretar que el ancestro de las cuatro no la poseía, y el ancestro de (1), (2) y (3) la ganó; para más sobre este punto véase la sección 2.3.5.

En esta sección se introduce el contenido de la cladística en tanto método de inferencia filogenética, de modo informal, utilizando el mismo lenguaje que se puede encontrar en un manual de biología (tal como los citados al inicio de este capítulo).

2.3.1. El *explanandum*: La matriz de datos

Lo que la cladística busca explicar es, como se dijo, una distribución "observada" (previamente determinada) de homologías en un conjunto de especies. Usualmente, esta distribución se presenta en la forma de una matriz de datos, que contiene a los taxa bajo consideración como filas y a los caracteres como columnas. Las celdas indican el estado que posee el taxón de la fila correspondiente para el carácter de la columna correspondiente (tabla 2.3).

	C_1	**C_2**	**C_3**
T_1	0	1	0
T_2	0	0	0
T_3	1	0	0
T_4	1	1	1

Tabla 2.3. Matriz de datos de ejemplo, que contiene 4 taxa y 3 caracteres.

Así, por ejemplo, la segunda columna de la primera fila indica que el taxón T_1 posee el estado 1 para el carácter C_2. Usualmente, los estados para un carácter se codifican mediante números. El número 1 puede significar así diferentes estados en diferentes caracteres (para un ejemplo, véase a continuación). La matriz recién dada es un ejemplo muy simplificado, a fines didácticos, del modo en que se vería una matriz de datos real. Los análisis reales generalmente contienen más de 20 taxa y un número aún mayor de caracteres. Un ejemplo de una matriz real puede verse (entre infinidad de otros lugares) en Ramírez (2003), cuyo objetivo era estudiar la filogenia de la subfamilia de arañas *Amaurobioidinae*. Una porción de la matriz puede verse en la figura 2.4.

TABLE 1
(*Continued*)

	0123456789	1111111111 0123456789	2222222222 0123456789	3333333333 0123456789	4444444444 0123456789	5555555555 0123456789	6666666666 0123456789	7777777777 0123456789	8888888888 0123456789	9999999999 0123456789
Amaurobioides africana	1000000110	1100010100	2100000000	0011010000	?011000101	00010?0100	1000000111	0000020001	000?1???00	0?0?101100
Axyracrus elegans	110000a010	1210010100	1100000000	0011000000	0011000101	00010?0000	1000100110	0000020?0	??????????	??0?101j00
Aysenia elongata	1?00001110	1010011???	21?0??1010	001???0?0?	??????????	??????????	??????????	??????????	??????????	??????????
Aysenia segestrioides	1000001110	1110010100	1100001000	0011000000	?011000101	00010?0001	1000000110	0000020001	000?000000	0?0?111000
Aysenia araucana	1000001110	1110010100	f100001000	0011000000	?011000101	00010?0001	1000000110	0000020001	000?000000	0?0?111000
Aysenia cylindrica	1?00001110	111?011100	2100001000	001??0000?	?011000101	00010?0000	10000001a0	0000020001	000?000000	0?0?101000
Aysenoides parvus	110000a011	1210011100	1100001000	0011000000	?011000111	00010?0000	1000000110	0?00020010	??????????	??0?101400
Aysenoides terricola	1?00001010	1110011100	2100001000	0011000000	?011000101	00011?0000	1000010111	0000021001	0000100000	0?0?101400
Aysenoides colecole	1100001011	1210010000	2100001000	0011000000	?011000101	00011?0000	1000010111	0?00021001	0000100000	0?0?101400
Gayenna americana	1100000011	0001030000	1100000000	0011000000	000?000100	0000010000	10010101g?	0000010002	1000200000	0000101600
Gayennoides molles	1000000011	0100011000	1100000000	0011000000	?00?000100	0010010000	100001014?	0000010002	0000200000	0002101600
Gayennoides losvilos	1000000010	0200011000	2000000000	0011000000	?00?000100	0010010000	100001014?	0000010002	0000200000	0002101600
Arachosia praesignis	1000000010	0a01030100	1100000000	00111001a?	?00?000100	00000?0000	100001013?	110000?001	0000200000	000?101600
Arachosia honesta	1100000010	0001130100	d100000000	0011100011	?00?000100	00000?0000	100001013?	11000??001	0100200000	000?101600
Arachosia bergi	1100000010	0101011100	1100000000	00111001aa	000?000100	00000?0000	100001013?	110001?001	0100200000	0000101600
Sanogasta pehuenche	1000000010	0100010000	1100000000	0011000000	?00?000100	0000010000	100001013?	1100000001	0000100000	000?101600
Sanogasta approximata	1000000010	0100011100	1100000000	0011000000	000?000100	100?010000	100001013?	1100000001	0001100000	000?101600
Sanogasta maculosa	1000000010	0100011100	1100000000	0011000000	000?000100	0000010000	100001013?	11000?0002	0000200000	000?101600
Sanogasta maculatipes	1000000010	0100010100	1100000000	0011000000	000?000100	0000000000	100010013?	1200000002	0000200000	000?101600
Sanogasta alticola	1000000010	0100010100	1100000000	0011000000	?00?000100	0000000000	100010013?	1200000002	0000200000	000?101600
Sanogasta mandibularis	1000000010	0100010a00	2101000000	0011000000	?00?000100	0000000000	100010013?	1200000002	0000200000	000?101600
Sanogasta puma	000000a000	0100010100	1100000000	0010000000	000?000100	0000010000	100001013?	1100000002	0000200000	000?101600
Sanogasta tenuis	000000a000	0100030100	1100000100	0010000000	?00?000100	0000010000	100001013?	1100000002	0000200000	000?101600
Sanogasta x-signata	1000000010	0100010100	1101000000	0011000000	?00?000100	0000010000	100?01013?	1100000002	0?01200000	000?101600
Sanogasta minuta	1000000010	0100011100	1100000000	0011000000	?00?000100	0000010000	100001013?	11000?0002	0000200000	000?101600
Sanogasta backhauseni	1000000010	0100021100	1100000000	0011000000	?00?000100	0000010000	100001013?	1100000002	0000200000	000?101600
Tomopisthes varius	1000000010	0100011000	1100000000	0011000000	000?000100	0000001000	100101014?	0000010002	1c00100011	0002101600
Tomopisthes horrendus	1000000010	0100110000	1100000000	0011100000	000?000100	000000?000	100101014?	0000010002	1001100011	0000101600
Tomopisthes pusillus	1000000011	0200001000	1100000000	0011000000	000?000100	000000?000	10010101g?	0000010002	1c01100011	0000101600
Philisca puconensis	1000000011	0100001100	1100000000	0011000000	?00?000100	000000?000	100101114?	0000010002	1001100000	0002101600
Philisca ornata	1a00000010	0100031000	2?10110000	0010000000	?00?000100	000000?000	100001112?	0000010002	10001000a0	0101101600

Fig. 2.4. Ejemplo de una porción (aproximadamente un sexto de los taxa) de una matriz de datos real (tomada de Ramírez, 2003)

En esta matriz los caracteres son morfológicos. Los caracteres 0 a 5 tienen que ver con el color y el patrón corporal, 6 a 8 con el caparazón, 9 a 16 con los ojos, etc. Por ejemplo, la primera columna (el carácter 0) representa al patrón corporal —sus estados son (0) color uniforme; (1) con parches contrastantes—; así, de la matriz se lee que *Sanogasta Pehuenche* posee un patrón de coloración con parches contrastantes, mientras que *Sanogasta Puma* posee un patrón uniforme. Por otro lado, el carácter 1 representa la ausencia (0) o presencia (1) de una mancha oscura longitudinal ventral en el abdomen (Ramírez, 2003, p. 15).

Si bien en la mayoría de los análisis clásicos en cladística los caracteres más usados eran morfológicos es posible usar otros tipos de caracteres. Se han usado, por ejemplo, caracteres comportamentales y ontogenéticos. Un tipo de dato que es muy frecuentemente utilizado hoy, más aún que los morfológicos, son los datos de secuencias (de nucleótidos o de aminoácidos). Ello se debe a que, por un lado, con las técnicas modernas de secuenciación los datos de secuencias son

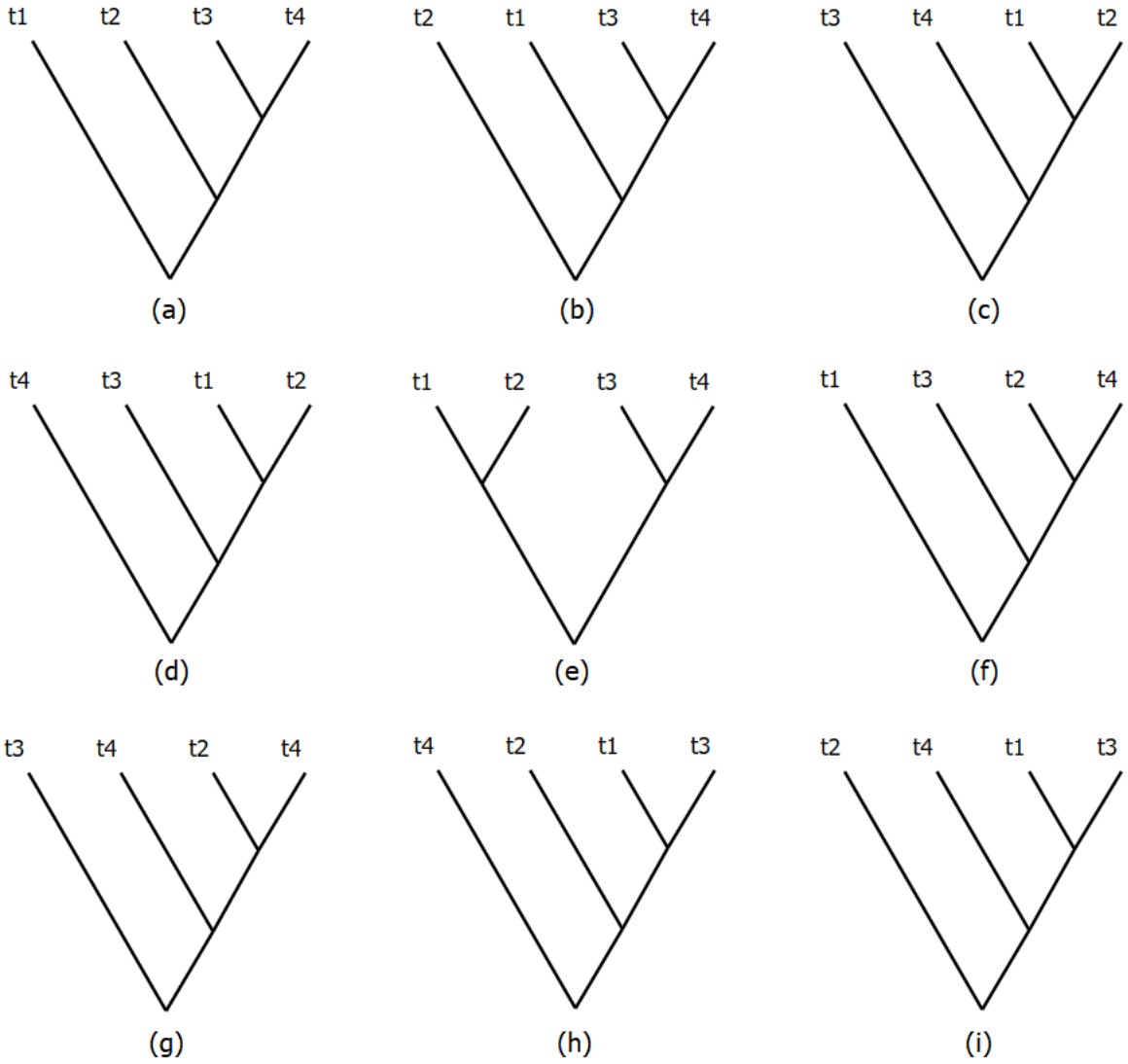
comparativamente mucho más fáciles de obtener en volúmenes grandes, además de que existen grandes bases de datos online (como GenBank) que contienen datos de secuencias utilizadas anteriormente en otros estudios y que pueden ser reutilizadas. Por otro lado, los procedimientos para obtener una secuencia a partir de un ejemplar son más explícitos que aquellos utilizados para codificar la morfología, lo cual hace que a veces sean vistas como menos "subjetivas". (Para una defensa de la necesidad de no descuidar la morfología en esta época en la que se ha vuelto prevalente el uso de datos moleculares véase Giribet, 2015).

Un ejemplo de la mencionada tendencia creciente a utilizar datos de secuencias puede verse en Ceccarelli et al. (2019). En este estudio, del que Ramírez es coautor, se estudió la misma subfamilia de arañas que en Ramírez (2003). Sin embargo, la matriz de datos constó aquí de 4 genes, con un total de 2.179 sitios, de los cuales 746 eran informativos para el método de parsimonia (i.e. eran variables y satisfacían ciertos requisitos formales adicionales; véase Ceccarelli et al., 2018, SI1, tabla S6). Por otro lado, también es frecuente encontrar matrices combinadas en las que hay tanto caracteres morfológicos como de secuencias. Por ejemplo, Labarque, Soto, Ramírez, y Arnedo (2015) realizaron un análisis con una matriz combinada del mismo grupo de arañas. Esto suele ser útil en casos en donde la información genética es difícil o imposible de conseguir (p.e. cuando se incluyen taxa fósiles en el análisis).

2.3.2. Cladogramas

Lo que explica la distribución de homologías (i.e. la matriz de datos) es, como se dijo, el patrón histórico de diversificación de las especies. Dado un conjunto finito de especies hay también un número finito de posibles patrones de diversificación. Estos patrones son usualmente representados como árboles, llamados cladogramas, en donde las ramas representan a la relación de ancestría inmediata y los nodos internos representan ancestros hipotéticos.[10] En los cladogramas los nodos internos se dividen siempre en dos, y no ocurre nunca una mezcla de linajes (i.e. dos nodos internos que apuntan al mismo nodo posterior, como podría ocurrir en casos de especiación por hibridación). En la figura 2.5 se encuentran representados los 15 cladogramas posibles para los 4 taxones de la tabla 2.3.

[10] Para algunos debates sobre lo que representa cada elemento de un cladograma véase el capítulo 6.


t1 t2 t3 t4
(a)
t2 t1 t3 t4
(b)
t3 t4 t1 t2
(c)
t4 t3 t1 t2
(d)
t1 t2 t3 t4
(e)
t1 t3 t2 t4
(f)
t3 t4 t2 t4
(g)
t4 t2 t1 t3
(h)
t2 t4 t1 t3
(i)

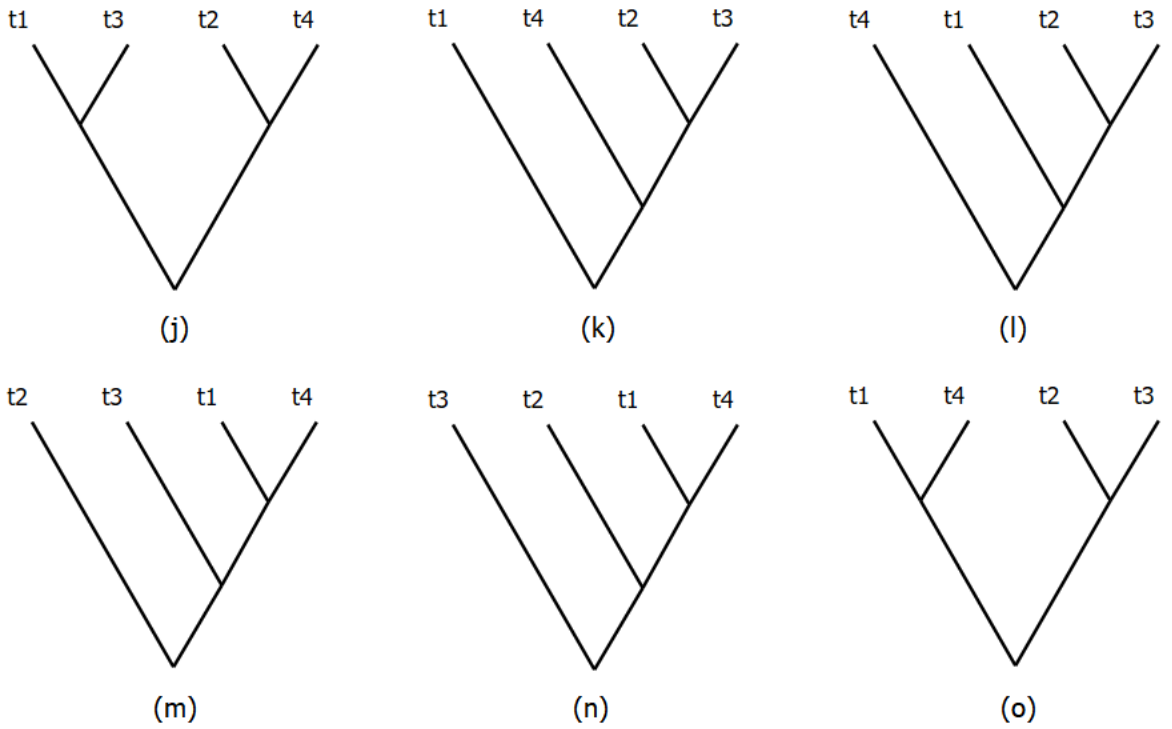


Fig. 2.5. Los 15 cladogramas posibles para 4 taxa.

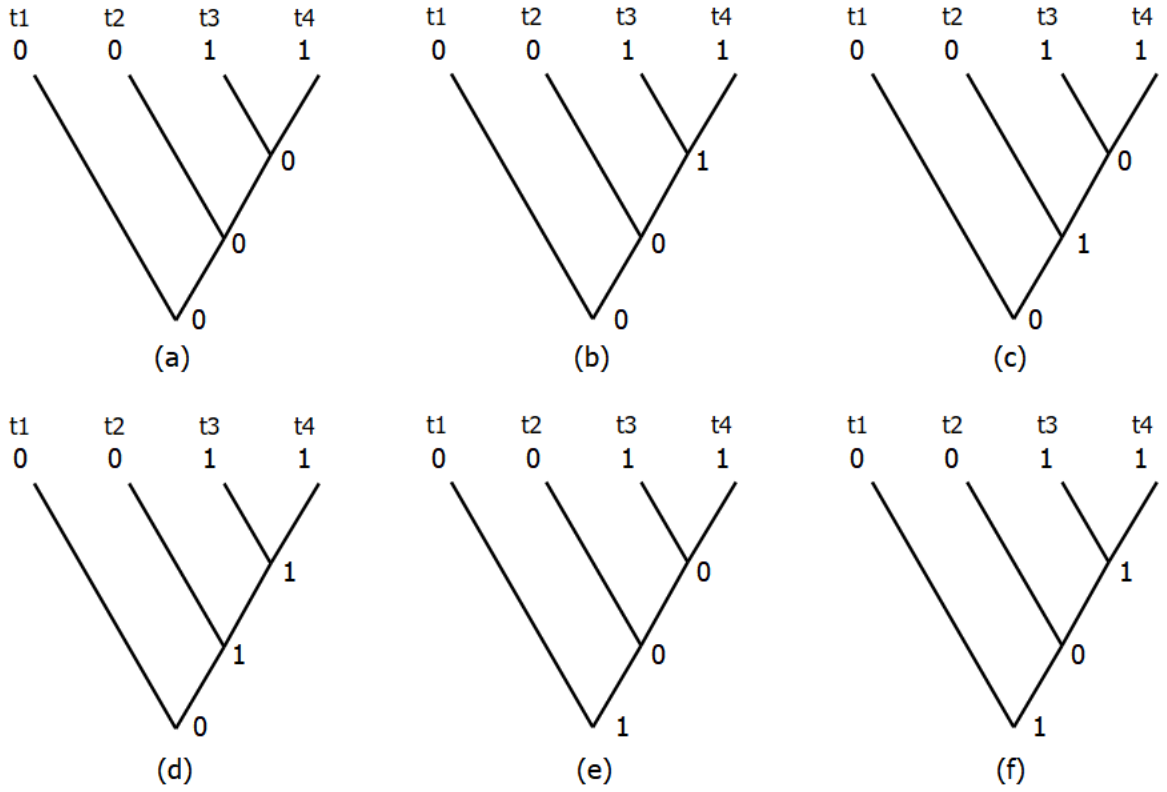
Cabe notar que el número de cladogramas posibles crece de manera extremadamente rápida con el aumento del número de taxa. Por ejemplo, con 5 taxa el número de cladogramas asciende ya a 106, con 9 taxa a 2.027.025, y con 20 taxa el número se eleva a 8.200.794.532.637.891.559.375 cladogramas (una derivación de la fórmula para realizar este cálculo puede encontrarse en Felsenstein, 1978). El análisis cladista debe elegir a uno de estos árboles como aquel que representa a la historia filogenética real. Esto se realiza comparando a los árboles según un criterio de optimalidad (que se describe a continuación). Dado que el número de árboles crece de manera tan rápida se torna imposible computacionalmente recorrer uno a uno todos los posibles cladogramas y medir su grado de optimalidad cuando el número de taxa es mayor a (aproximadamente) 20. De este modo, los análisis reales suelen utilizar métodos "heurísticos", que dan una solución aproximada al problema de encontrar el árbol óptimo, que puede no ser la globalmente óptima.[11]

[11] Como se verá en la siguiente sección, el criterio de optimalidad usado (la variable a minimizar) es el largo de un cladograma (la mínima cantidad de transformaciones entre estados sobre las ramas del árbol que es necesario postular para dar cuenta de los estados de los taxa terminales). De ese modo, que los métodos heurísticos no necesariamente alcancen la solución globalmente óptima quiere decir que no necesariamente encuentran el árbol filogenético de menor largo entre todos los posibles. Por otro lado, existen métodos "exactos" (que garantizan llegar a la solución globalmente óptima) y que no presuponen recorrer todos los árboles filogenéticos, p.e. el método de *branch-and-bound*. Sin embargo, incluso con estos métodos optimizados se torna imposible realizar búsquedas exactas con más de 20 taxa.

Dado que me interesan más los aspectos conceptuales que los computacionales no entraré en mayores detalles acerca de estos métodos heurísticos aquí.

2.3.3. Largo de un cladograma I

Para elegir un árbol, cada carácter es mapeado en cada árbol para ver cuántos cambios evolutivos deben postularse para explicar su distribución observada. Más precisamente, optimizar un carácter sobre un árbol significa asignar sus estados a los nodos internos del árbol de modo tal de minimizar el número de transformaciones entre estados. Por ejemplo, en el árbol (a) de la figura 2.5 los estados del carácter C_1 pueden asignarse a los nodos internos de los siguientes 8 modos (para los nodos terminales, i.e., los taxa bajo estudio, el estado del carácter está dado en la matriz de datos, de modo que son iguales en todas las asignaciones).



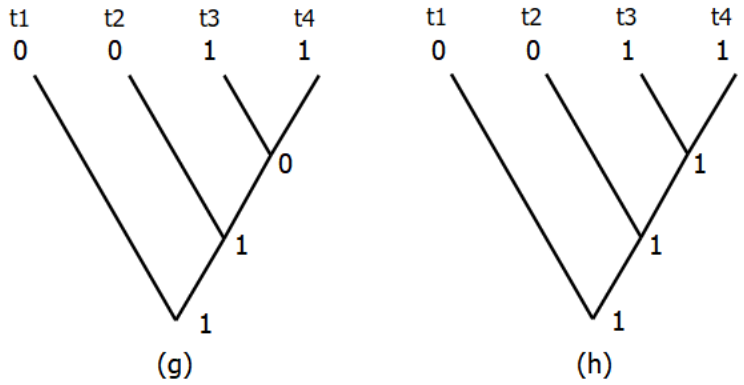


Fig. 2.6. Posibles asignaciones del carácter C_1 en el árbol (a).

Nótese que la asignación (b) requiere postular un solo cambio evolutivo para acomodar el carácter C_1 en este árbol, mientras que el resto de las asignaciones requiere al menos dos (por ejemplo, la asignación (a) requiere dos cambios, desde al ancestro de t_3 y t_4 hacia cada uno de sus descendientes). De ese modo, queda establecido que el carácter C_1 puede ser acomodado en este árbol postulando un solo cambio.[12] El *largo* del árbol (a) es la sumatoria del mínimo número de cambios necesario para explicar cada uno de los caracteres de la matriz. Así, si se realiza este mismo procedimiento con el resto de los caracteres de la matriz, puede verse que tal árbol posee un largo de 4 (el carácter C_2 se acomoda con un mínimo de 2 cambios, mientras que C_3 con 1 cambio). La cladística busca encontrar al árbol o cladograma de menor largo (i.e. aquel que requiere, en total, el menor número de cambios evolutivos para explicar las distribuciones observadas de caracteres).

Esto brinda un criterio para resolver los conflictos entre patrones anidados incompatibles de homologías, como el que se ilustró en la subsección 2.2.3. Sencillamente se prefieren aquellos patrones que están confirmados por la mayor cantidad de otros patrones. A este método de inferencia filogenética se lo ha llamado el método de *parsimonia* porque asume (o *prima facie* parece asumir, esto será discutido más adelante) que la explicación más simple (la que requiere postular menos eventos evolutivos de transformación de caracteres) es la correcta. Es decir, parece

[12] En adelante llamaré *asignación* a secas a una asignación de estados a los nodos internos, y asignación óptima u optimización a una asignación que minimiza el largo del árbol. Nuevamente, existen algoritmos que permiten determinar la mínima cantidad de pasos necesaria para optimizar un carácter en un árbol que no requieren probar toda posible asignación. Bajo ciertas circunstancias usuales (p.e. caracteres no ordenados o linealmente ordenados, véase más adelante) los algoritmos de Fitch (1971) y Farris (1970) son relativamente rápidos, de modo que el número de caracteres involucrado no suele constituir una limitación computacional para el análisis, a diferencia del número de taxa.

asumirse que en la evolución es menos "costoso" (o más frecuente) conservar un carácter en su estado actual que modificarlo, en un linaje.

La caracterización simplificada del método cladístico expuesta hasta este punto se enfrenta, en realidad, con varias complicaciones adicionales. Para comenzar, si se mide el largo de los otros 14 árboles de la figura 2.5, puede verse que los cladogramas (b) a (e) y (k) a (o) poseen igualmente un largo de 4 (i.e. pueden acomodarse los tres caracteres C_1- C_3 postulando el mismo número de transformaciones; los restantes 5 árboles, (f) a (j), poseen un largo de 5). La siguiente figura ilustra este hecho con los árboles (a), (e) y (n).

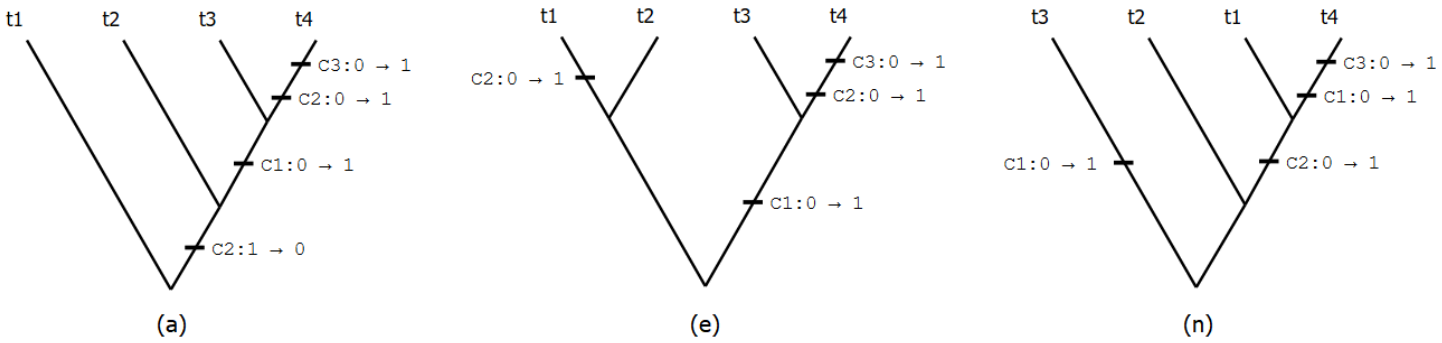


Fig. 2.7. Cladogramas (a), (e) y (n) de la figura 2.5, con todos los caracteres optimizados. Se indican los cambios en los caracteres en las ramas en lugar de los estados sobre los nodos internos (la presentación es equivalente, puede inferirse una cosa de la otra). Se observa que los tres árboles requieren postular 4 cambios en total para dar cuenta de los tres caracteres de la matriz de datos (i.e. todos tienen un largo de 4).

Es decir, en este caso, el análisis no permite determinar un único árbol filogenético óptimo, sino que encuentra varios. Esto es frecuente en análisis reales. Por ejemplo, Labarque et al. 2015, mencionado anteriormente, encontraron 1.944 árboles óptimos,[13] con un largo de 6.629. En ese caso, lo que se postula es que la historia evolutiva real debe estar entre los árboles óptimos seleccionados por el análisis.

Un modo de representar gráficamente una situación en la que distintos árboles pueden ser el correcto es por medio del árbol de *consenso estricto* entre los árboles en cuestión. En el consenso estricto se colapsan los nodos divergentes entre los árboles input, y se mantienen solo los grupos

[13] Entre un número astronómicamente grande de árboles posibles (véase más arriba). En casos reales el porcentaje de árboles óptimos sobre los árboles posibles sí suele ser mucho menor que en el ejemplo simplificado recién descripto.

para los cuales hay acuerdo entre todos ellos. Por ejemplo, un árbol de consenso estricto entre los árboles (a) y (f) de la figura 2.5 se vería de este modo:

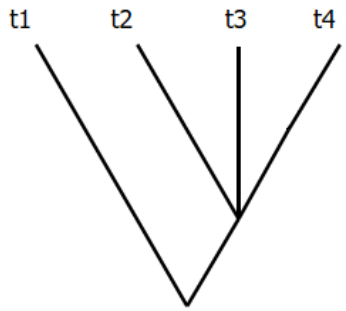


Fig. 2.8. Consenso estricto entre los árboles (a) y (f) de la figura 2.5. Ambos árboles contienen al grupo (t_2, t_3, t_4), con t_1 como grupo hermano, de modo que eso se refleja en el árbol de consenso. Sin embargo, difieren en el modo de resolver la relación entre t_2, t_3 y t_4, la cual, por tanto, aparece colapsada.

Aquí la tricotomía en el nodo que lleva al grupo (t_2, t_3, t_4) no indica una especiación simultánea en tres linajes, sino el desconocimiento del orden de separación binario entre esos tres linajes (es lo que se llama una "politomía débil"). Una porción de un árbol de consenso estricto real (Labarque et al. 2015, figura S2) se ve como sigue:

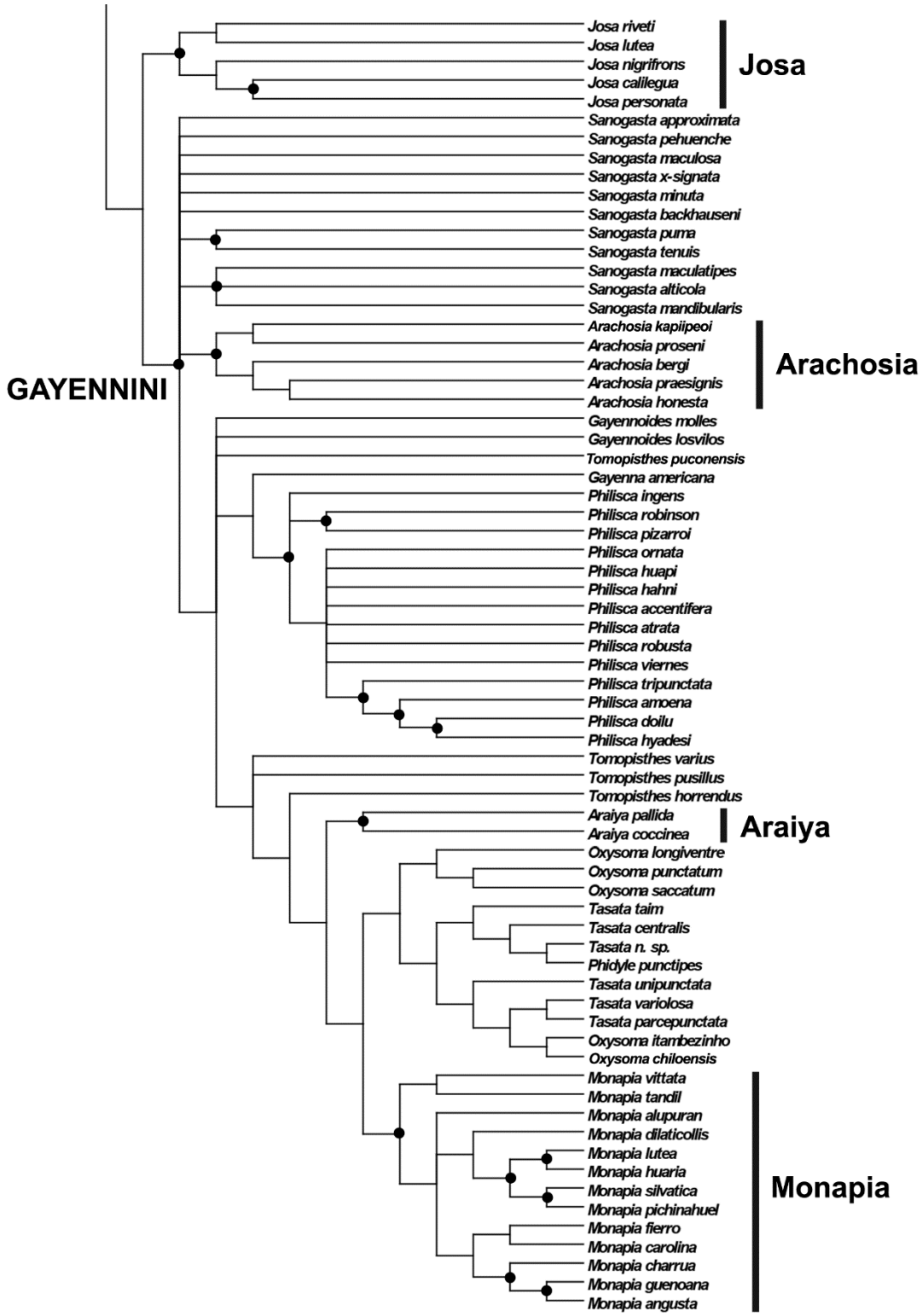


Fig. 2.9. Parte del árbol de consenso estricto, en el análisis de datos morfológicos, presentado en Labarque et al. (2015), figura S2.

Si se observan nodos como el etiquetado con GAYENNINI se verá que hay muchas instancias de nodos internos colapsados.

2.3.4 Matrices de costo y largo de un cladograma II

Otra complicación adicional surge del siguiente hecho. Hasta el momento se asumió que toda transformación de un estado a otro en cualquier carácter tiene el mismo costo o peso (i.e. toda transformación suma 1 al largo del cladograma). Sin embargo, esto no tiene por qué ser el caso. En ocasiones, se desea que una transformación en un carácter sea más costosa que una transformación en otro carácter, p.e. que sea más costoso ganar o perder una estructura morfológica compleja que una mancha de color superficial. Por otra parte, incluso dentro de un carácter, algunas transformaciones pueden establecerse como más costosas que otras, p.e. puede quererse que una transversión cueste más que una transición en un carácter de ADN, y/o que una inserción o deleción cueste más que una sustitución (reflejando el hecho de que las primeras ocurren con menor frecuencia en la naturaleza que las segundas).

El modo de lograr ambas cosas es asignando una matriz de costos a cada carácter, que indica cuán costoso es pasar de cada estado del carácter a cada otro estado. Por ejemplo, para el caso ficticio considerado anteriormente (tabla 2.3), las matrices de costo para los caracteres C_1-C_3 podrían verse del siguiente modo:

C_1	**0**	**1**
0	0	3
1	1	0

C_2	**0**	**1**
0	0	2
1	2	0

C_3	**0**	**1**
0	0	1
1	1	0

Tablas 2.4, 2.5 y 2.6. Ejemplos de matrices de costo alternativas.

Aquí C_3 se comporta igual que antes (todo cambio de estado entre dos nodos adyacentes añade 1 al largo del cladograma). Sin embargo, en este caso, se diría que C_2 pesa el doble que C_3 ya que dos transformaciones en C_3 serían equivalentes a una en C_2, i.e. el método prefiere un árbol con 3 cambios en C_3 a uno con 2 en C_2. Esto muestra que no es estrictamente correcto afirmar que la

cladística siempre elige el árbol con el menor *número* de transformaciones evolutivas postuladas.[14] En el caso de C_1, nótese que *dentro suyo* algunas transformaciones son más costosas que otras (p.e. pasar del estado 0 al estado 1 es más costoso que pasar de 1 a 0). En todos los casos, solo los costos *relativos* importan (cuánto cuesta una transformación contra el trasfondo de las otras), no los valores absolutos. Es decir, las unidades no importan.

La elección de una matriz de costos para los caracteres puede afectar al modo en el que ellos son optimizados sobre los árboles, así como el largo de los cladogramas. De ese modo, esta elección puede afectar a la cuestión de cuáles son los cladogramas óptimos. Por ejemplo, con las matrices de costo de las tablas 2.4 a 2.6, los árboles (e) y (n) —que antes eran ambos óptimos con un largo de 4, véase la figura 2.7— poseen ahora diferente largo, 6 y 5 respectivamente. Las optimizaciones correspondientes se verían como sigue:

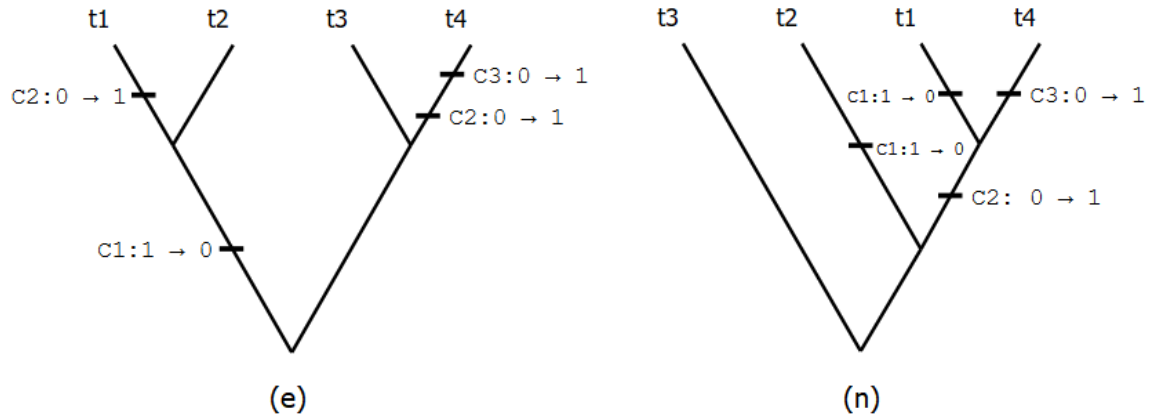


Fig. 2.8. Cladogramas (e) y (n) con todos los caracteres optimizados bajo las matrices de costo de las tablas 2.4 a 2.6.

2.3.5 Enraizamiento

[14] Asignar pesos diferenciales a caracteres desde el comienzo del análisis sería en realidad una de las formas posibles de pesar caracteres. Para otros métodos de pesado, véanse Farris (1969), Goloboff (1993). No profundizaré aquí sobre estos otros métodos, ni sobre las discusiones acerca de cuál método de pesado es más adecuado.

Un punto conceptual importante concierne a la raíz de los cladogramas. Puede notarse que los 15 cladogramas posibles para 4 taxones corresponden a los distintos modos de enraizar los siguientes 3 árboles no enraizados:

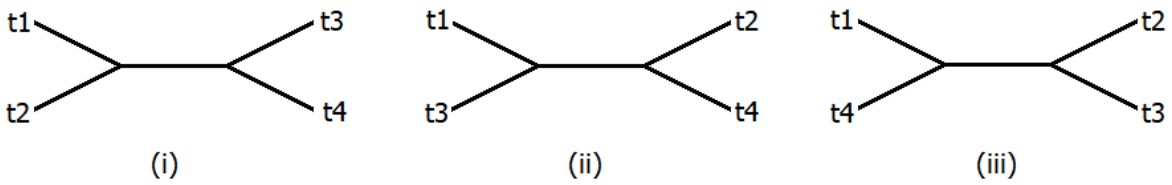


Fig. 2.9. Los 3 árboles no enraizados posibles con 4 taxa.

Por ejemplo, el cladograma (a) de la figura 2.5 es el resultado de tomar el árbol (i) de esta figura y enraizarlo en la línea que tiene a t_1 como nodo terminal, mientras que el cladograma (e) es el que surge de enraizar este mismo árbol en la línea del medio.

Los árboles no-enraizados son importantes por diversos motivos. Uno de ellos es que cuando todas las matrices de costo son simétricas (i.e. cuando todo cambio cuesta lo mismo que su inverso) entonces todos sus enraizamientos posibles tienen el mismo largo. Este resultado está matemáticamente probado (p.e. véase Semple y Steel, 2003, p. 85). Sin embargo, un modo intuitivo de verlo es el siguiente. Tómese un árbol con todos los caracteres optimizados (p.e. alguno de los de la figura 2.7). Al quitar la raíz del árbol y ponerla en otro lado todas las marcas indicando eventos de cambio evolutivo pueden mantenerse en su lugar para explicar la distribución de estados de la matriz —solo que algunos de esos cambios tendrán ahora la dirección reversa (p.e. figura 2.10). Sin embargo, dado que todo cambio y su reverso tienen el mismo costo el largo se mantendrá idéntico.

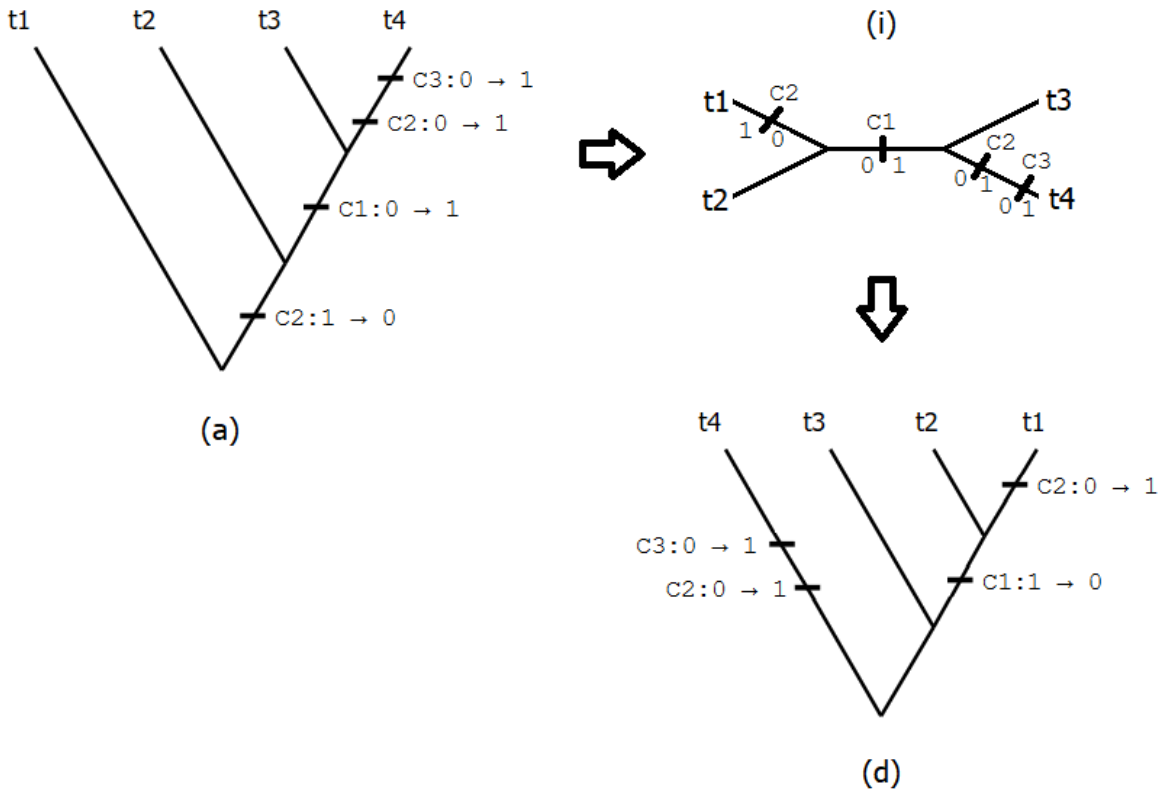


Fig. 2.10. Ejemplo de operación de re-enraizado de un árbol. Esquina superior izquierda: árbol (a) optimizado (véase figura 2.7a); esquina superior derecha: árbol no enraizado (i), surge de quitar el nodo raíz a (a) y borrar las direcciones de las líneas; esquina inferior derecha: árbol (i) enraizado en la línea que tiene a t_4 como terminal (el resultado es el árbol (d) de la figura 2.5). Se observa como las optimizaciones se mantienen y el largo es idéntico entre (a) y (d).

De hecho, en el primer análisis que se realizó en esta sección (subsección 2.3.3.) los 10 árboles óptimos consistían en todos los enraizamientos posibles de los árboles no enraizados (i) y (iii).

Este hecho (que se pueda re-enraizar sin alterar el largo) tiene diversas implicancias para la cladística. Por un lado, en caso de matrices simétricas (que son la gran mayoría en análisis reales), existe una ventaja computacional: no es necesario calcular el largo de cada árbol posible, sino que basta con calcular los largos de sus contrapartes no enraizadas, que son siempre menos en cantidad, y luego enraizarlos para obtener el conjunto de los árboles óptimos (de hecho, los programas de computación operan con árboles no enraizados al hacer búsquedas). Por otro lado, sin embargo, el hecho de que sea posible re-enraizar en cualquier rama es muy problemático, ya que implica que nunca se encontrará un único cladograma óptimo. Peor aún, dado que es posible re-enraizar un árbol en todas sus ramas sin alterar el largo siempre habrá un árbol óptimo que rompa la monofilia

del grupo que esa rama especificaba. En otras palabras, el consenso estricto de los árboles óptimos no tendrá *ningún* grupo más que el conjunto entero de los taxa.

Por estos motivos, desde muy temprano, los cladistas se ocuparon de proponer métodos para seleccionar el lugar adecuado para poner la raíz en un cladograma no enraizado (llamados métodos de *rooting* o enraizamiento). El método estándar aceptado actualmente fue propuesto por Nixon y Carpenter (1993, 1996) y consiste en lo siguiente.[15] En cualquier análisis, además del grupo cuya filogenia se intenta dilucidar (llamado *ingroup* o grupo interno), se debe incluir en la matriz de datos a al menos dos taxa que se hipotetice que están fuera de ese grupo, es decir, que no comparten al ancestro común más reciente del *ingroup* (este segundo grupo es llamado el *outgroup* o grupo externo). Si, en el análisis, el cladograma óptimo no enraizado contiene alguna rama tal que enraizarlo en ese punto vuelve al ingroup un grupo monofilético, entonces el árbol se enraíza allí. Caso contrario, se descarta la hipótesis de monofilia del *ingroup* respecto del *outgroup*, y se repite el análisis con otra elección de *outgroup*.

Como ejemplo, considérese la siguiente matriz con 3 taxa + 2 outgroups, y 3 caracteres:

	C_1	**C_2**	**C_3**
T_1	0	0	1
T_2	1	0	1
T_3	1	0	1
O_1	0	1	0
O_2	0	1	0

Tabla 2.7. Matriz de datos de ejemplo, con 3 taxa en el ingroup + 2 en el outgroup, y 3 caracteres.

Para esta matriz hay un único árbol no enraizado óptimo, que se ve como sigue:

[15] De hecho, el método ya había sido usado anteriormente, pero se volvió el método estándar a partir de la publicación de estos dos artículos.

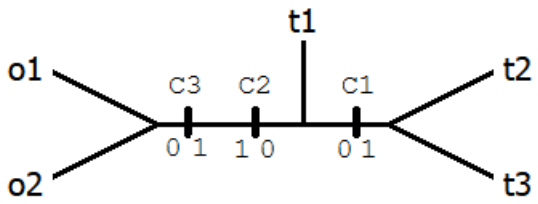


Fig. 2.11. Cladograma no enraizado óptimo para la matriz de datos de la tabla 2.7.

Hay tres ramas en este árbol no enraizado en las que poner la raíz volvería al ingroup un grupo monofilético. La raíz se sitúa, pues, en cualquiera de esas tres ramas, dado que los tres enraizamientos producen la misma topología (t_1, (t_2, t_3)) para el *ingroup*, que es lo único que interesa (adicionalmente, las optimizaciones también serán idénticas, cambiando solo la dirección de los cambios, como se ilustró arriba).

2.3.6. Otras nociones útiles

Si bien lo expuesto hasta aquí basta para comprender el funcionamiento de la cladística, existen algunos términos técnicos específicos que vale la pena introducir (y luego reconstruir), ya que son parte del vocabulario usual de la sistemática. Estos términos refieren a los tipos de relaciones filéticas entre grupos (monofilia, parafilia, polifilia) y de morfismos de caracteres (apomorfía, plesiomorfía, etc.).

Se dice que un grupo es monofilético (según un árbol) cuando incluye a un ancestro y *todos* y *solo* sus descendientes. Como se dijo anteriormente, la propuesta cladista en taxonomía sostenía que los grupos de una clasificación debían consistir solo en grupos monofiléticos (según el árbol filogenético real, al cual se accede a través de un análisis filogenético). Los grupos no monofiléticos son designados o bien polifiléticos o bien parafiléticos. Si bien hay acuerdo entre los cladistas en torno al significado de monofilia (no lo había, en cambio, entre los cladistas y los evolucionistas), la distinción entre grupos polifiléticos y parafiléticos no es tan clara, habiendo distintas propuestas de definición en la literatura (Farris, 1974; Hennig, 1966; Nelson, 1971).

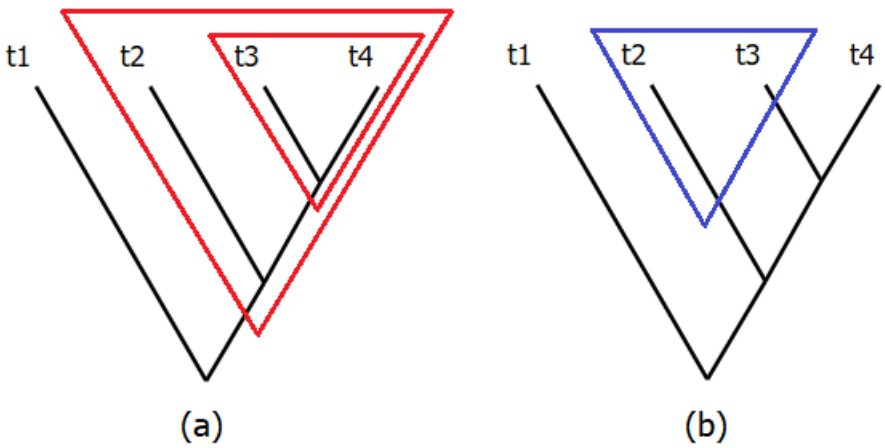


Fig. 2.12 (a) Ejemplo de dos grupos monofiléticos (en rojo); **(b)** Ejemplo de un grupo no-monofilético (parafilético o polifilético, en azul).

Por otro lado, se dice que un estado de un carácter es plesiomórfico para un taxón (del grupo de estudio o un ancestro hipotético de un cladograma) cuando su ancestro inmediato posee el mismo estado para ese carácter. En caso contrario, se dice que el carácter es apomórfico (i.e. es nuevo respecto de su antecesor inmediato). Las simplesiomorfías y sinapomorfías son usualmente definidas como plesiomorfías y apomorfías compartidas entre dos o más taxones, respectivamente (p.e. Kitching et al., 1998, p. 3). Como veremos más adelante, estas dos últimas definiciones no son del todo precisas ni satisfactorias. Me encargaré de clarificar estas nociones en el capítulo siguiente.

Por último, cabe mencionar que existen aplicaciones de la cladística por fuera de la biología. Ella es utilizada para la inferencia filogenética, por ejemplo, en la stemmatología (la reconstrucción de manuscritos ancestrales a partir de las copias que llegaron a la actualidad; p.e. Salemans, 1996) y la evolución de lenguajes (Dunn, 2010).

2.4. Conclusiones

En este capítulo, he introducido la teoría que será objeto de análisis en los capítulos siguientes. Se realizó primero una introducción a la tesis del origen común darwiniano y su *explanandum*, las cuales se encuentran presupuestas en la cladística. En segundo lugar, se hizo una presentación

histórica del programa de investigación cladista y se distinguió entre sus dos componentes, el clasificatorio (la cladonomía) y el método de inferencia filogenética (la cladística). Por último, en la tercera sección, se introdujo el contenido de la cladística.

En el capítulo siguiente se presentará una reconstrucción formal de la cladística, junto con una introducción al aparato formal metateórico que utilizaré para ello (el estructuralismo). En la segunda parte de la tesis, se introduce un programa de computación cuyo fin es contrastar reconstrucciones formales. La reconstrucción presentada en el capítulo siguiente, así como la comprensión pre-formal de la teoría ganada en este capítulo, servirán de insumos para ejemplificar el uso del programa con un caso real.

Capítulo 3. Una reconstrucción estructuralista de la cladística

En este capítulo se ofrece una reconstrucción formal de la cladística, utilizando el marco conceptual del estructuralismo metateórico. Esta reconstrucción funcionará luego como *input*, en el capítulo 5, para el programa de computación presentado en el 4, a fin de ilustrar la fertilidad y aplicabilidad de ese programa. Adicionalmente, la reconstrucción tendrá por sí sola algunas consecuencias filosóficas interesantes, que serán tratadas en el capítulo 6.

La estructura del capítulo será la siguiente. En la primera sección se hace un breve resumen de otros tratamientos formales que se han hecho de la cladística en el área de la matemática filogenética. Las siguientes dos secciones introducen las nociones estructuralistas de axiomas impropios y clase de los modelos potenciales, y leyes fundamentales y clase de los modelos a secas, respectivamente. Conjuntamente con estas presentaciones se dan los axiomas impropios y las leyes fundamentales de la cladística. La cuarta sección elucida formalmente otros conceptos (grupo monofilético, tipos de morfismo de caracteres) que no forman parte de las leyes pero aun así son importantes en la práctica. En la quinta sección se introducen las nociones estructuralistas de especialización, T-teoricidad y condición de ligadura, y se identifica cuáles son estas en la cladística. Por último, se hace una introducción a la noción de método de determinación y a su elucidación estándar en el marco del estructuralismo, y se da un ejemplo (informal) de un método de determinación, que no parece fácilmente acomodable a aquella elucidación estándar.

Las subsecciones en donde se realiza la presentación del estructuralismo no entrarán en todo el detalle matemático de una presentación de manual de la metateoría (para eso, véase Balzer et al., [1987] 2012), sino que trataré al material con un nivel de complejidad y abstracción (espero) más didáctico, pero a la vez suficiente para mis propósitos.

3.1. Tratamientos formales previos de la cladística

Existe actualmente un campo de estudio dedicado al estudio matemático de la filogenética, en el que participan tanto biólogos como matemáticos. El presente capítulo construye sobre lo edificado en esa área. Sin embargo, la matematización de la cladística llevada a cabo allí fue elaborada sin un aparato conceptual metateórico explícito por detrás, que permita entender a la teoría cladística

y a diversas características suyas como análogas a otras teorías en otros ámbitos de la ciencia. Por otra parte, los objetivos de estos autores tenían usualmente que ver más con probar teoremas y establecer resultados de complejidad algorítmica que con precisar el lenguaje para identificar las aserciones fácticas de la teoría y discutir cuestiones conceptuales que la rodean. Los objetivos de la formalización en esta tesis son más cercanos a los segundos que a los primeros (aunque, por supuesto, algunos teoremas podrán probarse a partir de los axiomas identificados). Por tanto, la terminología usada (y por lo tanto las pruebas) diferirán en alguna medida de aquellos trabajos previos. Donde sea relevante, indicaré la equivalencia entre mis nociones y las de otros autores.

Ampliando sobre lo anterior, la literatura en matemática filogenética trata principalmente con los siguientes problemas. En primer lugar, dado que las búsquedas de árboles suelen ser computacionalmente intensivas (de hecho, con más de 20 taxones, el número de árboles posibles es tan grande —véase Felsenstein 1978— que no pueden hacerse búsquedas exactas), buena parte de estos análisis están apuntados a encontrar algoritmos eficientes y/o a establecer resultados de computabilidad para los ya existentes (p.e. Farris, 1970). En unos pocos casos, cierto aparato formal fue propuesto para clarificar el significado de algunos conceptos, como los de polifilia y parafilia (Farris, 1974), homología (Jardine, 1967) y para representar a los cladogramas explícitamente como grafos (Martin, Blackburn, & Wiley, 2010). En otros, el formalismo matemático tiene que ver con la introducción de métricas (p.e. de distancia entre árboles en Robinson & Foulds, 1981). Finalmente, en raras ocasiones, los sistemáticos han intentado ofrecer axiomatizaciones de la cladística (Aubert, 2015; Brower, 2000; Farris et al., 1970). Una parte importante de trabajo realizado en todas estas áreas puede encontrarse compilado en dos manuales recientes (Semple & Steel, 2003; Steel, 2016). En cuanto a la parte computacional, existen diversos software que hacen búsquedas de árboles (Goloboff & Catalano, 2016; Wilgenbusch & Swofford, 2003; son dos de los más conocidos).

Si bien toda esta tradición existe, no hay aún una reconstrucción formal completa, que establezca claramente el lenguaje, las restricciones que rigen sobre este (en terminología estructuralista, los axiomas impropios), que dé una definición precisa de todos los conceptos definidos, y que precise cuáles son las leyes fundamentales y especiales (acerca de las leyes en biología, véase Lorenzano, 2014-2015). Esto último es especialmente importante para entender a la cladística como una teoría empírica y no meramente como un método (aunque las teorías suelen implicar metodologías en sus condiciones de aplicabilidad). En ese sentido, una formulación clara

de las leyes fácticas puede ayudar a comprender cómo se contrasta una teoría como esta. Al analizar esta teoría desde la metateoría propuesta se pondrán en relieve ciertas cuestiones que probablemente serían pasadas por alto de otro modo (p.e. el estatus de T-teoricidad de los conceptos —lo cual será utilizado luego en el capítulo 6, en la discusión sobre las homologías— y las condiciones de ligadura). Por otra parte, una reconstrucción que apunta a la clarificación conceptual y al reconocimiento de las afirmaciones fácticas de la teoría puede ser útil posteriormente para la comunicación y enseñanza de la teoría.

3.2. Axiomas impropios

3.2.1. Los axiomas impropios en el estructuralismo

Los axiomas impropios establecen cuántos y cuáles son los conceptos de la teoría, su tipo lógico y les atribuyen algunas propiedades formales, sin expresar las afirmaciones fácticas fundamentales de la teoría. Presentar los axiomas impropios es básicamente establecer el lenguaje o (mejor dicho) el marco conceptual de la teoría. Los axiomas impropios de una teoría determinan una clase de modelos que los estructuralistas llaman *modelos potenciales* (y que en lógica a veces se denominan interpretaciones), notada como $\mathbf{M_p}$. El nombre se debe a que los modelos potenciales son aquellos respecto de los cuales tiene sentido preguntarse si satisfacen las leyes fácticas, ya que contienen el número y el tipo adecuado de conceptos. Las leyes fácticas se agregarán luego como axiomas sobre (i.e. como una extensión de) los axiomas impropios, restringiendo la clase de los $\mathbf{M_p}$ a una nueva clase **M** de modelos actuales (i.e. $\mathbf{M} \subseteq \mathbf{M_p}$). Cada modelo potencial representa a una aplicación posible de la teoría (que puede ser exitosa o no), mientras que los modelos actuales representan a aquellas aplicaciones posibles de la teoría que además satisfacen las leyes fácticas.

Como recién se dijo, en los axiomas impropios a veces se establecen restricciones sobre los modelos que van más allá de la pura especificación del lenguaje. Por ejemplo, respecto de una relación triádica *T*, los siguientes dos son axiomas impropios típicos:

(*i*) $T \subseteq D_1 \times D_2$ [donde D_1 y D_2 son dominios]

(*ii*) *T* es reflexiva, simétrica y transitiva

Si bien (*i*) establece que *T* es una relación diádica (aunque nos da ya sobre qué conjuntos), (*ii*) nos da propiedades de *T* que van más allá de la pura especificación del lenguaje. No hay un criterio general, preciso y tajante que determine cuáles axiomas deben formar parte de los impropios y cuáles de los propios (véase Balzer et al., [1987] 2012 pp. 55-64). Sin embargo, en general, cuando se trabaja con casos específicos no resulta complicado distinguir entre las aserciones fácticas fundamentales y aquellas que constituyen el marco conceptual.

A lo largo de esta sección se presentan los axiomas impropios de la cladística con explicaciones informales acerca de su significado, de manera más didáctica pero menos ordenada y sistemática. Una presentación completa y más ordenada de los modelos potenciales de la teoría se encuentra en la última subsección (3.2.7). Las definiciones y caracterizaciones de funciones se presentarán como parte de los axiomas impropios, para que las leyes fundamentales contengan solo las aserciones fácticas fundamentales de la teoría.

3.2.2. Taxa terminales y árboles filogenéticos

El dominio de aplicación de la teoría (el conjunto de taxa cuyas relaciones de cercanía filogenética se desea inferir) puede ser caracterizado formalmente a partir de un conjunto *T*:

T ($=\{t_1, t_2, t_3, \ldots, t_n\}$) es un conjunto finito que contiene al menos tres elementos.

Los árboles filogenéticos o cladogramas, con todos los posibles patrones de diversificación entre estos taxa, estarán contenidos en otro conjunto *A*:

$A = \{A_i / A_i$ es un árbol filogenético$\}$

Cada árbol filogenético será representado formalmente como un grafo (una versión de los grafos basados en nodos [*node-based trees*] de Martin et al., 2010). Habrá para ello un conjunto finito de nodos *N*, que incluye a los nodos terminales e internos —por lo que $T \subset N$. A su vez, llamaré *I* al conjunto de los nodos internos ($I \stackrel{\text{def}}{=} N - T$). Dado que todo cladograma para un mismo conjunto de

terminales tiene el mismo número de nodos internos (véase Steel, 2016, p. 10, lema 1.4) no será necesario definir un conjunto de nodos internos diferente para cada cladograma. De ese modo, cada cladograma $A_i \in A$ puede definirse del siguiente modo. A_i es un cladograma sii $A_i = \langle N, D(A_i)\rangle$, tal que:

(i) $\quad D(A_i) \subseteq N^2$

$D(A_i)$ representa a la relación de descendencia entre nodos según A_i. $\langle n_1, n_2\rangle \in D(A_i)$ significa que, según el cladograma A_i, n_2 es un descendiente inmediato de n_1. Que esto sea una relación significa que todo A_i será un grafo *dirigido* sobre el conjunto de vértices N.

El modo de dibujar un cladograma A_i a partir de su especificación formal es el usual en teoría de grafos: se representará a los nodos como puntos, y a los ejes (la relación de descendencia entre dos nodos) como líneas que los unen. Seguiré la convención biológica de representar a los ejes como teniendo una dirección hacia arriba (o hacia la derecha, cuando el cladograma se presenta girado 90°). Por ejemplo, el cladograma A_x

$$N = \{t_1, t_2, t_3, t_4, n_1, n_2, n_3\}$$
$$D(A_x) = \{\langle n_1, n_2\rangle, \langle n_1, t_1\rangle, \langle n_2, n_3\rangle, \langle n_2, t_4\rangle, \langle n_3, t_2\rangle, \langle n_3, t_3\rangle\}$$

puede dibujarse del siguiente modo:

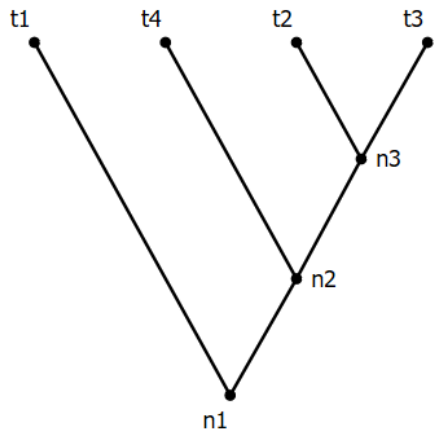


Fig. 3.1. Ilustración del árbol filogenético (*qua* grafo) especificado anteriormente.

Un camino (en inglés *path*) entre dos nodos n_x y n_y es una secuencia de nodos $\langle n_x, n_{x1}, n_{x2} \ldots, n_y \rangle$ tal que para todo par de nodos consecutivos n_{xi} y n_{xi+1}, $\langle n_{xi}, n_{xi+1} \rangle \in D(A_i)$. Por ejemplo, en el árbol recién dado, $\langle n_1, n_2, t_4 \rangle$ es un camino.

Con esta terminología en mente, es posible (y necesario) poner algunas restricciones adicionales sobre $D(A_i)$, ya que no todo grafo dirigido sobre N constituirá un cladograma:

(*ii*) $D(A_i)$ es acíclica. Formalmente:

$\nexists n_{x1} \in N$, $\exists n_{x2}, \ldots, n_{xj} \in N$ tal que;

$\langle n_{x1}, n_{x2} \rangle \in D(A_i)$ y … y $\langle n_{xj-1}, n_{xj} \rangle \in D(A_i)$ y $\langle n_{xj}, n_{x1} \rangle \in D(A_i)$

[Esto es, no hay ningún camino que salga desde un nodo y termine en el mismo nodo]

(*iii*) $D(A_i)$ es enraizada. Formalmente:

$\exists! n_x \in N$ tal que $\forall n_y \in N$:

(*iv*.1) si $n_y \neq n_x$, $\langle n_y, n_x \rangle \notin D(A_i)$

(*iv*.2) $\exists n_{x1}, \ldots, n_{xj} \in N$ ($\langle n_x, n_{x1} \rangle \in D(A_i)$ y $\langle n_{x1}, n_{x2} \rangle \in D(A_i)$ y … y $\langle n_{xj}, n_y \rangle \in D(A_i)$)

Nótese que $\exists! x\, \varphi(x)$ es la notación lógica usual para "existe un *y solo un* individuo que satisface φ" (i.e. $\exists! x\, \varphi(x) \stackrel{\text{def}}{=} \exists x\, \varphi(x) \wedge \forall z$ si $\varphi(z)$ entonces $z = x$). (iv.1) Establece que existe un único nodo que no es descendiente de ningún otro nodo (llamado nodo basal o raíz del cladograma); mientras que (iv.2) afirma que existe un camino entre la raíz y todo otro nodo del cladograma.

(*iv*) Cada nodo interno tiene exactamente dos descendientes:

$\forall n \in I$, $\exists n_{x1}, n_{x2} \in N$ tal que:

(*v*.1) $n_{x1} \neq n_{x2}$

(*v*.2) $\langle n, n_{x1} \rangle \in D(A_i)$ y $\langle n_x, n_{x2} \rangle \in D(A_i)$

(*v*.3) $\forall n_{x3} \in N$ (si $n_{x1} \neq n_{x3} \neq n_{x2}$ entonces $\langle n_x, n_{x3} \rangle \notin D(A_i)$)

(*v*) Los taxa T bajo consideración son nodos terminales:

$\forall t \in T$ ($\nexists n \in N$ tal que $\langle t, n \rangle \in D(A_i)$

(*vi*) $D(A_i)$ no tiene fusiones (*mergings*), i.e. dos nodos distintos no pueden tener el mismo descendiente (biológicamente, esto se puede interpretar como afirmando que no hay especiación por hibridación):

$\forall n \in N \nexists n_x \in N \exists n_y \in N (\langle n_x, n\rangle \in D(A_i)$ y $\langle n_y, n\rangle \in D(A_i)$ y $n_x \neq n_y)$

3.2.3. Isomorfismo entre cladogramas

Nótese que lo que nos interesa es el patrón de diversificación de los taxa terminales. Los nodos internos pueden intercambiarse entre sí en un árbol sin modificar este patrón. Por ejemplo, los siguientes dos grafos representan el mismo escenario evolutivo:

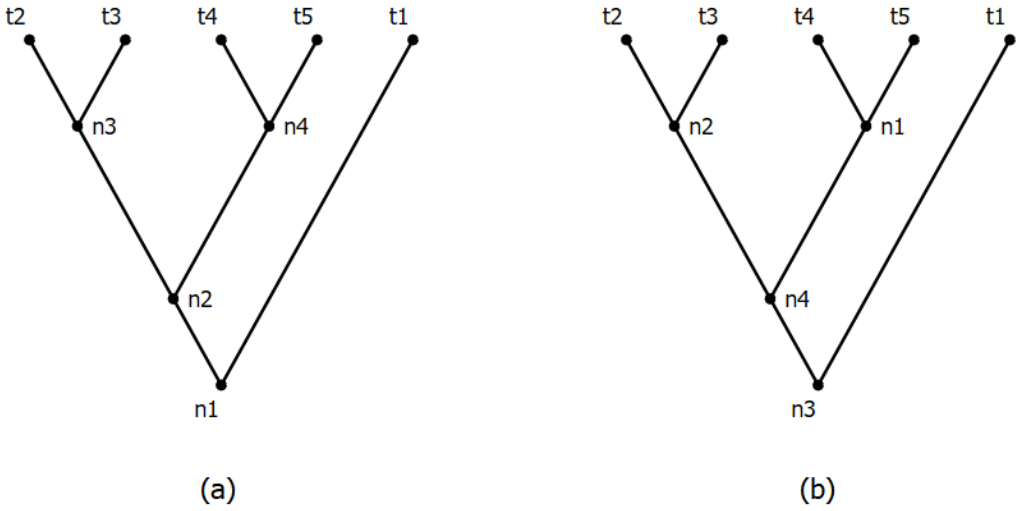


Fig. 3.2. Árboles filogenéticos distintos (*qua* grafos) que representan la misma relación de hermandad entre grupos, ya que lo único que importa es el orden de diversificación de linajes de los taxa terminales (y no los nombres que se da a los nodos internos o a los ancestros hipotéticos).

Diremos pues que dos cladogramas A_i y A_j son *isomórficos* (notado como $A_i \cong A_j$) si y solo si existe una biyección *iso*: $N \rightarrow N$ tal que:

(*i*) si $n_x \in T$, entonces $iso(n_x) = n_x$ (los taxa terminales quedan idénticos)

(*ii*) si $\langle n_x, n_y\rangle \in D(A_i)$, con $n_x, n_y \in N$, entonces $\langle iso(n_x), iso(n_y)\rangle \in D(A_j)$

Bajo esta definición, los cladogramas de la figura 3.2 son isomórficos, ya que existe una función biyectiva *iso* que asigna a los nodos del primero los siguientes nodos del segundo:

$iso(t_1) = t_1$; $iso(t_2) = t_2$; $iso(t_3) = t_3$; $iso(t_4) = t_4$; $iso(t_5) = t_5$;
$iso(n_1) = n_3$; $iso(n_2) = n_4$; $iso(n_3) = n_2$; $iso(n_4) = n_1$

Nótese que la noción de isomorfismo para cladogramas es distinta a la noción de isomorfismo para grafos en general (es una noción más restringida, ya que incluye a la condición (*i*)). Ello se debe a que, por ejemplo, no deseamos que el siguiente cladograma sea isomórfico con los dos anteriores, ya que ilustra un patrón de diversificación diferente:

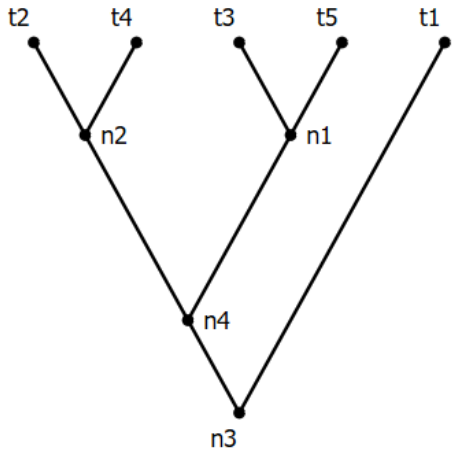


Fig. 3.3. Árbol no isomórfico con los dos anteriores (bajo la noción recién definida, sí isomórfico bajo la noción usual de isomorfismo para grafos).

(sin la condición (*i*) arriba lo serían). En adelante, se asume que *A* contiene al conjunto de todos los cladogramas *hasta el isomorfismo*.

3.2.4. Jerarquías

Las jerarquías (Semple & Steel, 2003, pp. 50-53; Steel, 2016, pp. 18-20) constituyen otro modo de representar árboles filogenéticos, las cuales, como veremos, tienen varias propiedades matemáticas y computacionales deseables. Formalmente, una jerarquía es un conjunto, tal que cada elemento suyo se corresponde con un nodo del árbol. Los nodos son también representados como conjuntos (i.e. la jerarquía es un conjunto de conjuntos), que contiene a los terminales a los cuales tal nodo lleva a través de algún camino. Así, por ejemplo, el árbol de la figura 3.2 puede representarse como una jerarquía del siguiente modo:

$$\{\{t_1, t_2, t_3, t_4, t_5\}, \{t_2, t_3, t_4, t_5\}, \{t_2, t_3\}, \{t_4, t_5\}, \{t_1\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}\}$$

El elemento de la izquierda corresponde al nodo raíz, el siguiente al nodo n_2 (de la figura 3.2a) los dos siguientes a los nodos n_3 y n_4 (nuevamente, de 3.2a) respectivamente, y los últimos cinco a los nodos terminales. De ese modo, la jerarquía correspondiente a un árbol puede ser visto como el conjunto de los grupos monofiléticos pertenecientes a ese árbol.

Formalmente, el conjunto J de las jerarquías puede definirse como el de los $x \subseteq \wp(\wp(T))$ tales que:

(i) $T \in x$
(ii) $\forall t \in T\ (\{t\} \in x)$
(iii) $\emptyset \notin x$
(iv) $\forall y_1 \in x\ \forall y_2 \in x\ (y_1 \cap y_2 \in \{y_1, y_2, \emptyset\})$
(v) $|x| = 2T - 1$

La prueba de la correspondencia entre jerarquías y árboles filogenéticos puede encontrarse en Semple y Steel (2003, pp. 52-53) y en Steel (2016, pp. 18-19). Dado que cada árbol se corresponde con una y solo una jerarquía, introduciré una función j: $A \rightarrow J$, tal que, dado un árbol, devuelve su jerarquía correspondiente. Esto se logra definiendo $j(A_i) = \{X \in \wp(T) / \exists n \in N$ (hay un camino de n a cada miembro de X y no hay ningún camino de n a un nodo que no sea miembro de X)$\}$. De manera más precisa, puede definirse al conjunto P de caminos como $P = \{\langle n_x, n_{x1}, n_{x2} \ldots, n_y\rangle$ / para todo par de nodos consecutivos n_{xi} y n_{xi+1}, $\langle n_{xi}, n_{xi+1}\rangle \in D(A_i)\}$, y a la función j como:

$j(A_i) = \{X \in \wp(T) / \exists n \in N\ (\forall x \in X, \exists p \in P\ (p = \langle n, \ldots, x\rangle)$ y
$\forall x \in N - X, \neg\exists p \in P\ (p = \langle n, \ldots, x\rangle))\}$.

3.2.5. La matriz de datos (caracteres y estados) y las asignaciones de estados a los nodos internos de un árbol

El dominio de los caracteres constará de un conjunto finito y no-vacío $C = \{C_1, \ldots, C_m\}$. Cada carácter C_i será a su vez un conjunto de estados —i.e. $C_i = \{c_{i,1}, c_{i,2}, \ldots, c_{i,j}\}$. Adoptaré la convención usual de codificar a los estados para cada carácter por medio de números naturales. Con ello no se quiere implicar que todo carácter está ordenado (véase después). Otro punto importante es que los caracteres no comparten estados, es decir, $\bigcap C = \emptyset$.

Adicionalmente, tendremos una serie de funciones de asignación para mapear estados de los caracteres a nodos. La función d_T asignará estados a los nodos terminales (está dada como input del análisis cladista), es decir: d_T: $T \times C \rightarrow \bigcup C$. En otras palabras, los conceptos T, C y d_T representan formalmente a la matriz de datos.

Contaremos además con un conjunto $\delta = \{d_1, \ldots, d_n\}$ de funciones de asignación completas, es decir, que mapean estados de caracteres a todo miembro de N, incluyendo a los nodos internos (i.e. para toda $d_i \in N$, d_i: $N \times C \rightarrow \bigcup C$). Para ambos tipos de funciones vale el requisito de que $d_T(t, C_i) \in C_i$ y $d_i(n, C_i) \in C_i$; es decir, dado un nodo y un carácter, ambas devuelven un estado de ese carácter y no de otro. δ es un conjunto de funciones y no una función particular ya que el estado de los nodos internos no está dado como un *input* en una aplicación de la teoría; en cambio, tal asignación puede realizarse de diversos modos. Sin embargo, todas las posibles asignaciones tienen que respetar la asignación (sí dada) de los nodos terminales. Por lo tanto, un requisito adicional para los miembros de δ es que:

$\forall d_i \in \delta, \forall t \in T, \forall C_i \in C\ (d_i(t, C_i) = d_T(t, C_i)))$

Un ejemplo de una función d_T de asignación para un conjunto T de 4 taxa terminales ($T = \{t_1, t_2, t_3, t_4\}$), dados 2 caracteres con 2 estados cada uno ($C_1 = \{c_{1,1}, c_{1,2}\}$ y $C_2 = \{c_{2,1}, c_{2,2}\}$) sería la siguiente:

$d_T(t_1, C_1) = c_{1,1}$ $\quad$ $d_T(t_1, C_2) = c_{2,1}$

$d_T(t_2, C_1) = c_{1,1}$ $\quad$ $d_T(t_2, C_2) = c_{2,2}$

$d_T(t_3, C_1) = c_{1,2}$ $\quad$ $d_T(t_3, C_2) = c_{2,2}$

$d_T(t_4, C_1) = c_{1,1}$ $\quad$ $d_T(t_4, C_2) = c_{2,2}$

Esta información representa formalmente a la siguiente matriz de datos:

Taxa / Carácter	C_1	C_2
t_1	1	1
t_2	1	2
t_3	2	2
t_4	1	2

Tabla 3.4. Matriz de datos representada por la función d_T de arriba.

Una función de asignación completa d_x basada en d_T podría verse así:

$d_x(t_1, C_1) = c_{1,1}$ $\quad$ $d_T(t_1, C_2) = c_{2,1}$

$d_x(t_2, C_1) = c_{1,1}$ $\quad$ $d_x(t_2, C_2) = c_{2,2}$

$d_x(t_3, C_1) = c_{1,2}$ $\quad$ $d_x(t_3, C_2) = c_{2,2}$

$d_x(t_4, C_1) = c_{1,1}$ $\quad$ $d_x(t_4, C_2) = c_{2,2}$

$d_x(n_1, C_1) = c_{1,2}$ $\quad$ $d_x(n_1, C_2) = c_{2,2}$

$d_x(n_2, C_1) = c_{1,2}$ $\quad$ $d_x(n_2, C_2) = c_{2,1}$

$d_x(n_3, C_1) = c_{1,1}$ $\quad$ $d_x(n_3, C_2) = c_{2,1}$

Esta función de asignación podría luego ser dibujada sobre dos árboles distintos del siguiente modo:

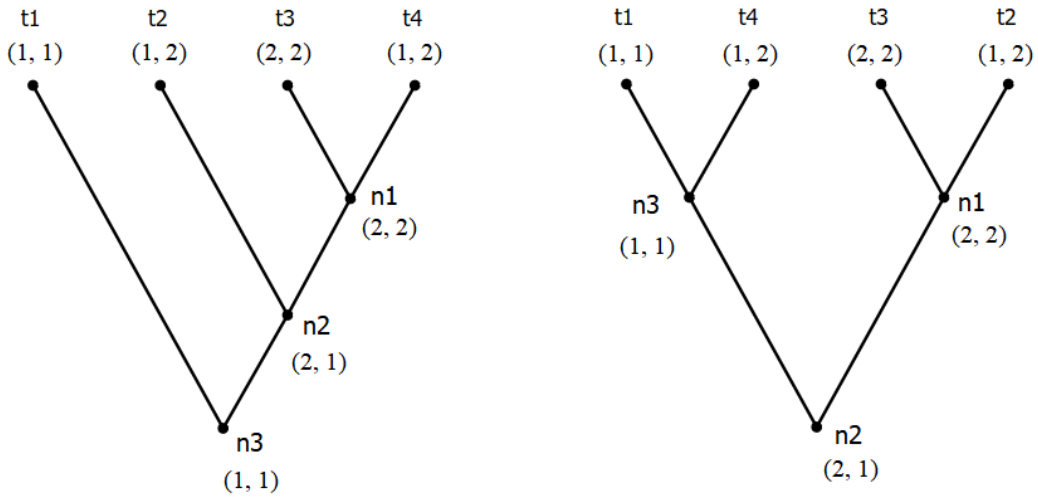


Fig. 3.4. Función de asignación d_x dibujada sobre dos árboles.

Esta no será la asignación óptima para ninguno de los dos árboles (i.e. la que minimiza el largo), sino que, en ambos casos, esta será otra (u otras) función del conjunto δ.

Nótese que dados $|T|$ taxa (y por lo tanto $|T| - 1$ nodos internos) y un carácter con s estados, cada uno de los $|T| - 1$ nodos internos puede asumir cualquiera de los s estados. Es decir, para un carácter habrá $s^{|T|-1}$ asignaciones posibles. El número total de funciones de asignación surge de multiplicar el número de asignaciones posibles para cada carácter:

$$|\delta| = \prod_{i=1}^{|C|} |C_i|^{|T|-1}$$

Este número crece de manera muy rápida con el número de taxa y de caracteres incluidos en el análisis. Sin embargo, como se explicó en el capítulo anterior, existen métodos para computar la asignación óptima (u óptimas) para un árbol, que no requieren computar el largo del cladograma bajo cada una de ellas. Este punto será importante en la sección 3.6.

3.2.6. Matrices de costo y largo de un cladograma

Las matrices de costo serán representadas por una única función *$*, que toma como argumentos a dos estados de un carácter y devuelve un número natural. Formalmente:[16]

$$\$: \bigcup C \times \bigcup C \rightarrow \mathbb{N}$$

Así, el costo evolutivo de pasar del estado $c_{i,j}$ al estado $c_{i,k}$ en el carácter C_i quedará representado por $\$(c_{i,j}, c_{i,k})$. En el tipo de aplicaciones que consideraré en esta tesis, el costo de mantenerse en el mismo estado es siempre 0. Sin embargo, a fin de mantenerme en un plano de mayor generalidad, no introduciré este requisito como axioma.

A continuación, se definen dos funciones binarias, *w* y l_1, que devuelven el peso de un eje y el largo de un cladograma, respectivamente, ambas bajo una asignación. La función *w* devuelve el peso de un eje en un cladograma, bajo una asignación (i.e. lo que ese eje añade al largo del cladograma). Informalmente, el peso de un eje es igual a la suma de los costos de las transformaciones que ocurren entre los nodos que ese eje une. Formalmente:

$$w(\langle n_j, n_k \rangle, d_x) = \sum_{Ci \in C} \$(d_x(n_j, C_i), d_x(n_k, C_i))$$

De modo un poco más intuitivo, si $\$(c_{i,j}, c_{i,k})$ representa el costo de pasar del estado *j* al estado *k* en el carácter C_i, $\$(d_x(n_y, C_i), d_x(n_z, C_i))$ representará el costo de pasar de un nodo a otro para ese carácter, bajo una asignación d_x. *w* simplemente suma tal costo para todos los caracteres presentes en *C*.

El largo de un cladograma A_x bajo una asignación d_m es simplemente la suma de los pesos de todos sus ejes. Es decir:

$$l_1(A_i, d_m) = \sum_{\langle x,y \rangle \in D(A_i)} w(\langle x, y \rangle, d_m)$$

[16] Nótese que así caracterizados el dominio y el codominio, esta será una función parcial ya que habrá argumentos (p.e. dos estados que no pertenecen al mismo carácter) para los cuales no habrá ningún valor asignado. Podría modificarse el dominio y codominio para que *$* sea total, pero ello complejizaría el axioma. Por simplicidad se lo deja de esta forma.

El largo a secas de un cladograma (l_2) quedará definido como el mínimo largo de ese árbol bajo toda asignación posible. Es decir:

$$l_2(A_i) = min(\{l_1(A_i, d_j) / d_j \in \delta\})$$

El conjunto de los árboles óptimos *AO* estará definido como el conjunto de los árboles de largo$_2$ mínimo. Formalmente:

$$AO = \{A_i \in A / l_2(A_i) \in min(\{l_2(A_k) / A_k \in A\})\}.$$

Por otra parte, será útil para lo que sigue contar con la noción de asignación óptima para un árbol. Para ello, puede introducirse una función δ_O, definida del siguiente modo:

$$\delta_O(A_i) = \{d_j \in \delta / l_1(A_i, d_j) = l_2(A_i)\}.$$

Es decir, dado un árbol, δ_O devuelve el conjunto de las asignaciones de estados a nodos que arrojan el mínimo largo para ese árbol.

Por último, puede definirse un conjunto *AOj*, de jerarquías óptimas, que será simplemente el conjunto de las jerarquías que corresponden a los árboles óptimos:

$$AOj = \{J_i \in J / \exists A_k \in \text{AO}\ (j(A_k) = J_i)\}.$$

En la subsección siguiente se presentan todos los axiomas dados hasta este punto de manera más ordenada, y en un formato estructuralista usual.

3.2.7. Los axiomas impropios de **CLAD**

En esta subsección se presentan los axiomas impropios de la cladística en formato estructuralista estándar. Por comodidad (y por motivos que se verán más claros más adelante en la tesis) incluiré a las definiciones como axiomas impropios.

$x = \langle A, C, I, J, N, T, AO, AOj, A_R, \delta, \delta_O, \$, d_R, d_T, j, l_1, l_2, w\rangle$ es una cladística potencial ($x \in$ **$\mathbf{M_p}$(CLAD)**) si y solo si:

(1) T es un conjunto finito tal que $|T| > 3$ [taxa terminales]

(2) N es un conjunto finito tal que $T \subset N$ [nodos de los cladogramas]

(3) $I = N - T$ [nodos internos]

(4) $A = \{A_i / A_i = \langle N, D(A_i)\rangle\}$ que satisfacen los siguientes requisitos hasta el isomorfismo:

(4.1) $D(A_i) \subseteq N^2$ [relación binaria de descendencia inmediata]

(4.2) $\nexists n_{x1} \in N, \exists n_{x2}, \ldots, n_{xj} \in N$ tal que $\langle n_{x1}, n_{x2}\rangle \in D(A_i)$ y … y $\langle n_{xj-1}, n_{xj}\rangle \in D(A_i)$ y $\langle n_{xj}, n_{x1}\rangle \in D(A_i)$ [$D(A_i)$ es acíclica]

(4.3) $\exists! n_x \in N$ tal que $\forall n_y \in N$ (

si $n_y \neq n_x$, $\langle n_y, n_x\rangle \notin D(A_i)$ y

$\exists n_{x1}, \ldots, n_{xj} \in N$ ($\langle n_x, n_{x1}\rangle \in D(A_i)$ y $\langle n_{x1}, n_{x2}\rangle \in D(A_i)$ y … y $\langle n_{xj}, n_y\rangle \in D(A_i)$))

[$D(A_i)$ es enraizada]

(4.4) $\forall n \in I, \exists n_{x1}, n_{x2} \in N$ tal que

($n_{x1} \neq n_{x2}$ y $\langle n, n_{x1}\rangle \in D(A_i)$ y $\langle n_x, n_{x2}\rangle \in D(A_i)$ y

$\forall n_{x3} \in N$ (si $n_{x1} \neq n_{x3} \neq n_{x2}$ entonces $\langle n_x, n_{x3}\rangle \notin D(A_i)$))

[Cada nodo interno tiene exactamente dos descendientes]

(4.5) $\forall t \in T$ ($\nexists n \in N$ tal que $\langle t, n\rangle \in D(A_i)$ [Los taxa bajo consideración son nodos terminales]

(4.6) $\forall n \in N \nexists n_x \in N \exists n_y \in N$ ($\langle n_x, n\rangle \in D(A_i)$ y $\langle n_y, n\rangle \in D(A_i)$ y $n_x \neq n_y$) [No ocurren fusiones en el árbol]

(5) $A_R \in A$ [A_R es un árbol particular, el árbol real, véase la sección siguiente]

(6) J es el conjunto [jerarquías] de los $x \subseteq \wp(\wp(T))$ tales que:

(5.1) $T \in x$

(5.2) $\forall t \in T$ ($\{t\} \in x$)

(5.3) $\emptyset \notin x$

(5.4) $\forall y_1 \in x \; \forall y_2 \in x$ ($y_1 \cap y_2 \in \{y_1, y_2, \emptyset\}$)

(5.5) $|x| = 2T - 1$

(7) C ($= \{C_1, \ldots, C_m\}$) es un conjunto finito, no-vacío tal que cada C_i ($= \{c_{i,1}, c_{i,2}, \ldots, c_{i,j}\}$) es un conjunto finito, no vacío. [caracteres y estados]

(8) $\bigcap C = \emptyset$ [Los caracteres no comparten estados]

(9) d_T: $T \times C \rightarrow \bigcup C$, tal que $d_T(t, C_i) \in C_i$ [asignación para terminales]

(10) δ (= $\{d_1, \ldots, d_n\}$) es un conjunto finito, no-vacío [de funciones de asignación completas], tal que $\forall d_i \in N$ (d_i: $N \times C \rightarrow \bigcup C$ y $d_i(n, C_i) \in C_i$)

(11) $\delta_R \in \delta$ [δ_R es una asignación particular, la asignación real, véase la sección siguiente]

(12) $\forall d_i \in \delta$, $\forall t \in T$, $\forall C_i \in C$ ($d_i(t, C_i) = d_T(t, C_i)$)) [Las asignaciones completas respetan las asignaciones para terminales]

(13) $\$$: $\bigcup C \times \bigcup C \rightarrow \mathbb{N}$ [función parcial de costos]

(14) w: $N^2 \times \delta \rightarrow \mathbb{N}$, tal que $w(\langle n_j, n_k \rangle, d_x) = \sum_{Ci \in C} \$(d_x(n_j, C_i), d_x(n_k, C_i))$ [peso de un eje]

(15) l_1: $A \times \delta \rightarrow \mathbb{N}$, tal que $l_1(A_i, d_m) = \sum_{\langle x,y \rangle \in D(A_i)} w(\langle x, y \rangle, d_m)$ [largo de un cladograma bajo una asignación]

(16) l_2: $A \rightarrow \mathbb{N}$ tal que $l_2(A_i) = min(\{l_1(A_i, d_j) / d_j \in \delta\})$ [largo de un cladograma]

(**17**) $AO = \{A_i / l_2(A_i) \in min(\{l_2(A_k) / A_k \in A\})\}$ [árboles óptimos]

(18) δ_O: $A \rightarrow \delta$, tal que $\delta_O(A_i) = \{d_j \in \delta / l_1(A_i, d_j) = l_2(A_i)\}$[asignaciones óptimas para un árbol]

(19) j: $A \rightarrow J$, tal que $j(A_i) = \{X \in \wp(T) / \exists n \in N$ ($\forall x \in X$, $\exists p \in P$ ($p = \langle n, \ldots, x \rangle$) y $\forall x \in N - X$, $\neg \exists p \in P$ ($p = \langle n, \ldots, x \rangle$))$\}$ [jerarquía correspondiente a un árbol]

(20) $AOj = \{J_i \in J / \exists A_k \in$ AO ($j(A_k) = J_i$)$\}$ [jerarquías óptimas]

3.3. Leyes fundamentales

3.3.1. La noción de ley fundamental

Como se explicó arriba (subsección 3.2.1), una primera característica de las leyes de la teoría es que, a diferencia de los axiomas impropios, establecen las restricciones fácticas que una aplicación (i.e. modelo) de la teoría debe satisfacer. En el estructuralismo existe además una distinción entre leyes *fundamentales* y leyes *especiales*. La idea general es que las leyes fundamentales establecen restricciones sustantivas (aunque a veces relativamente débiles) sobre todas las aplicaciones de la

teoría, mientras que las leyes especiales imponen restricciones adicionales a los de la ley fundamental pero solo sobre un subconjunto de casos (i.e. condicionalmente a que se cumpla cierto antecedente).[17]

Un ejemplo de ello puede verse en la mecánica cartesiana (una teoría que será presentada en mayor detalle en el capítulo 4). En esta teoría, que se ocupa de cómo (a qué velocidades) se mueven un conjunto de cuerpos que pueden chocar entre sí, la ley fundamental afirma que la velocidad total presente en un sistema cerrado de cuerpos se mantiene idéntica en todo tiempo. Esta ley vale siempre en la teoría, independientemente de que los cuerpos bajo estudio choquen o no, y de las velocidades iniciales a las que choquen. De ese modo, si en un sistema de dos cuerpos estos chocan con velocidades iniciales de v_x y v_y, la ley fundamental solo nos dice que la suma de las velocidades finales será idéntica a la suma inicial, pero no las velocidades concretas que cada cuerpo tendrá luego del choque. Eso se establece en las leyes especiales. En estas, se establecen restricciones adicionales para distintos tipos de choques (p.e. si los cuerpos vienen inicialmente con velocidades opuestas pasa cierta cosa, si uno está quieto y el otro lo choca pasa otra, etc.) que permiten predecir unívocamente las velocidades finales.

Nuevamente, no hay un conjunto preciso de condiciones necesarias y suficientes para distinguir entre leyes fundamentales y leyes especiales. Los estructuralistas hablan más bien de condiciones necesarias débiles o síntomas. Algunos síntomas de que un enunciado es una ley fundamental son (un resumen ampliado de ellos puede encontrarse en Lorenzano, 2014-2015):

(*a*) Carácter sinóptico o arracimado: Las leyes fundamentales suelen contener a todos (o a la mayoría) de los conceptos de la teoría.

(*b*) Valen en todas las aplicaciones intencionales de la teoría (en esto se distinguirían de las leyes especiales, que pretenden aplicarse solo a algunas aplicaciones de la teoría).

[17] Estrictamente hablando, en el estructuralismo, la ley fundamental es la clase de **M** de los modelos, no los axiomas propios que se usan para caracterizar tal clase. Sin embargo, puede hablarse en un sentido derivado de los axiomas propios como leyes, en tanto aquellos que determinan a la ley en sentido estricto. De hecho, como se verá a continuación, algunos de los requisitos que usualmente se formulan para que algo sea una ley fundamental (p.e. ser sinóptica) solo tienen sentido en tanto predicadas de enunciados, no de clases de modelos. En adelante, por comodidad, continuaré cometiendo la pequeña imprecisión conceptual de tratar a los axiomas como leyes (ya que ello no afectará los puntos conceptuales que deseo establecer).

(*c*) Carácter cuasi-vacuo o empíricamente irrestricto: ponen restricciones débiles sobre lo que no puede ocurrir, de modo que suelen resistir la refutación cuando son tomadas aisladamente.

(*d*) Papel sistematizador: tienen un alto poder unificador al permitir tratar distintos *tipos* de sistemas bajo un mismo marco conceptual. Funcionan como guía para la formulación de leyes especiales (Moulines, 1978).

(*e*) Fuerza modal: Las leyes no son meras generalizaciones accidentales sino que soportan enunciados contrafácticos. Esto queda elucidado por medio de la idea de que toda aplicación pretendida ("real" o imaginaria) debe satisfacer la ley fundamental *junto con sus especializaciones* —i.e. las leyes fundamentales son necesarias *en su dominio de aplicación*.

A continuación, se presentan lo que se identificaron como las leyes fundamentales de la cladística, y se discute en qué medida ellas satisfacen los síntomas (*a*) y (*b*). Los restantes tres serán discutidos posteriormente, en la subsección 3.5.1.

3.3.2. Las leyes fundamentales de **CLAD**

Recuérdese que lo que la cladística busca inferir es el árbol filogenético que dio lugar (y que explica) la distribución observada de caracteres y estados. Para hacerlo postula no solo un árbol como óptimo, sino también una asignación de estados para los ancestros hipotéticos (los nodos internos). Una aplicación de esta teoría será considerada exitosa cuando: (*i*) el árbol inferido como el óptimo coincide con el árbol real (o bien, si hay muchos árboles óptimos, cuando el real está entre los óptimos); y (*ii*) cuando la distribución real de los caracteres de los ancestros coincide con la optimización de los caracteres sobre aquel árbol inferido (si hay más de una optimización posible, cuando la real es alguna de ellas).

Para poder expresar las afirmaciones fácticas (*i*) y (*ii*) recién mencionadas, es necesario introducir a la reconstrucción los conceptos de árbol real y distribución de caracteres real. Lo haré simplemente a través de dos constantes de individuo, A_R y d_R, bajo el único requisito formal de que

$A_R \in A$ y $d_R \in \delta_O$. De ese modo, (*i*) y (*ii*) pueden expresarse formalmente simplemente diciendo que:

(*i*) $A_R \in AO$
(*ii*) $d_R \in \delta_O(A_R)$

Esto expresa que (*i*) el patrón de diversificación real de las especies está entre los árboles óptimos; y (*ii*) la historia real de evolución de los caracteres está entre las asignaciones de estados óptimas para el árbol real, según la cladística.

Recuérdese además que, cuando hay más de un cladograma óptimo, los cladistas suelen presentar la información contenida en todos los árboles de menor largo a través de un árbol de consenso estricto. La noción de consenso estricto es más fácilmente formulable formalmente utilizando a las jerarquías. El consenso estricto de un conjunto de árboles (*qua* jerarquías) puede ser representado sencillamente como el conjunto intersección de ese conjunto (Semple & Steel, 2013, p. 53; Steel, 2016, p. 21). Por ejemplo, dados los árboles A_i y A_j, representados como jerarquías:

$j(A_i) = \{\{t_1, t_2, t_3, t_4, t_5\}, \{t_2, t_3, t_4, t_5\}, \{t_2, t_3\}, \{t_4, t_5\}, \{t_1\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}\}$
$j(A_j) = \{\{t_1, t_2, t_3, t_4, t_5\}, \{t_2, t_3, t_4, t_5\}, \{t_2, t_3, t_4\}, \{t_2, t_3\}, \{t_1\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}\}$

Graficables del siguiente modo (figura 3.5, a, b, respectivamente):

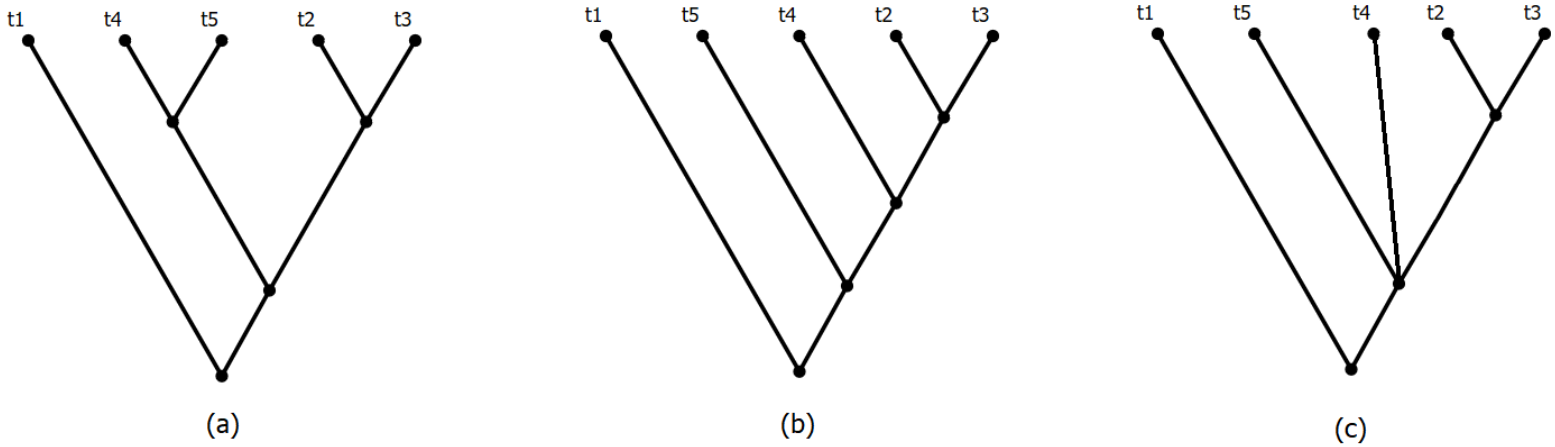


Fig. 3.5. Dos árboles y su consenso estricto

El consenso estricto, *qua* jerarquía, puede ser presentado como:

$$j(A_i) \cap j(A_j) = \{\{t_1, t_2, t_3, t_4, t_5\}, \{t_2, t_3, t_4, t_5\}, \{t_2, t_3\}, \{t_1\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}\}$$

El cual, nuevamente, contiene el conjunto de los terminales a los que lleva cada nodo del árbol de la figura 3.5c.

Por otro lado, diré que una jerarquía $j(A_i)$ refina a un árbol de consenso así definido cuando el consenso es un subconjunto del árbol (Steel, 2016, p. 21). Por ejemplo, en el caso de arriba, vale tanto que $j(A_i) \cap j(A_j) \subseteq j(A_i)$ como que $j(A_i) \cap j(A_j) \subseteq j(A_j)$. Inicialmente había considerado que la condición (*i*) podía expresarse equivalentemente diciendo que:

$$(i') \bigcap AOj \subseteq j(A_R)$$

Sin embargo, esta afirmación es más débil. El motivo se explicará al final del capítulo 5.

3.3.3. Presentación estructuralista de las leyes fundamentales

La presentación estructuralista de las leyes fundamentales se ve como sigue:

x es una cladística actual ($x \in \mathbf{M}(\mathbf{CLAD})$) si y solo si:

(1) $x \in \mathbf{M_p}(\mathbf{CLAD})$

(2) $A_R \in AO$

(3) $d_R \in \delta_O(A_R)$

Puede verse claramente que estas dos leyes (tomadas conjuntamente) exhiben el síntoma (*b*) que es típico de las leyes fundamentales. Toda aplicación exitosa de la cladística será una en la que los árboles óptimos contienen al árbol real, y en la que la optimización de los caracteres identifica

correctamente los estados de los ancestros. Si alguna de ellas falla, entonces la aplicación de la cladística sería considerada no-exitosa.[18]

El caso del primer síntoma es menos claro. Las leyes fundamentales parecen, a primera vista, incluir solo unos pocos conceptos de la teoría. Nótese, sin embargo, que *AO* y δ_O son conceptos definidos. Por ejemplo, *AO* está definido a partir de *A* y l_2. A su vez, l_2 está caracterizado a partir de *A*, δ y l_1; l_1 de otros conceptos, etc. Gráficamente:

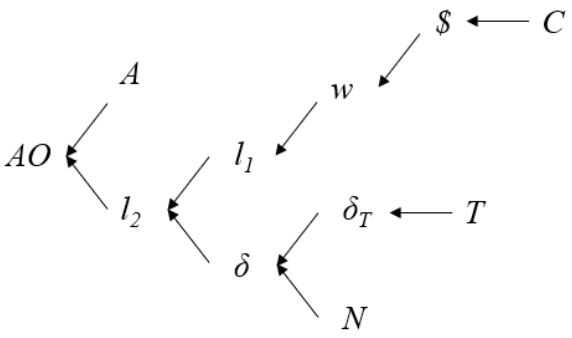


Fig. 3.6. Grafo ilustrando qué conceptos están definidos o caracterizados a partir de cuáles otros. Algunas flechas fueron omitidas para mejorar la legibilidad (p.e. l_1 y l_2 también incluyen a *A* en su caracterización, l_1 y *w* incluyen a δ, etc.)

Si se realiza este ejercicio con todos los conceptos que figuran en las dos leyes puede verse que, en realidad, los dos axiomas dados contienen (conjuntamente) a la gran mayoría de los conceptos de la teoría. Es decir, si bien puede sostenerse que estrictamente hablando las leyes no exhiben el requisito (*a*) —ya que (*a*) es un requisito sintáctico, y no hay aquí una única ley que contiene explícitamente a (casi) todos los conceptos de la teoría— sí lo exhiben en espíritu. Esto se debe a que tomados conjuntamente los axiomas propios que determinan a la clase de los modelos establecen conexiones / restricciones en las interpretaciones para todos los términos de la teoría.[19]

[18] El caso de la segunda (**(3)** en la presentación estructuralista) es un poco más debatible. En los inicios del programa cladista el énfasis estaba puesto más sobre la identificación de la topología correcta que sobre las reconstrucciones de estados ancestrales. Sin embargo, actualmente, el segundo de estos propósitos es considerado tan importante como el primero. Adicionalmente, las contrastaciones explícitas de la cladística midieron el grado de ajuste de las asignaciones de estados a ancestros a los estados reales de esos ancestros (véase la sección siguiente).

[19] De todos modos, cabe notar que el propio Moulines (1991, pp. 233-234) discute casos de teorías en donde ocurren cosas similares, —p.e. en la mecánica relativista del continuo y en la electrodinámica, en las reconstrucciones ofrecidas por Bartelborth (1988)— en donde las diversas leyes fundamentales "no parecen poder reformularse como leyes sinópticas de manera plausible y natural" (Moulines, 1991, p. 234). Habría aquí, sin embargo, dos diferencias con esos casos. Por un lado, las dos leyes fundamentales de **CLAD** parecen dos aspectos o partes de una misma ley, mientras que en aquellos parecerían ser leyes distintas e independientes. Por otro lado, en el presente caso (incluso tomadas

3.3.4. El estatus fáctico de la cladística

En esta subsección ofreceré algunas consideraciones adicionales acerca del estatus de las leyes fundamentales recién presentadas (especialmente de la primera, las consideraciones en torno a la segunda son idénticas), lo cual abonará a una comprensión de por qué la reconstrucción es adecuada. Recuérdese que la primera ley fundamental afirma (informalmente) que el cladograma más parsimonioso (i.e. el de menor largo, o uno de los de menor largo si hay más de uno) coincide con el patrón de diversificación real de las especies. A la pregunta de cómo se justifica la aceptación de esta ley puede responderse de (al menos) los tres siguientes modos.

En primer lugar estaría la idea de que la justificación de la cladística está basada en la simplicidad. Por detrás de ello podría haber un principio metafísico *a priori* de que la realidad es simple (una posición que, en mi conocimiento, nadie sostuvo), o bien un principio epistemológico de que las explicaciones más simples son preferibles debido a nuestras limitaciones cognitivas (véase Sober, 1988, capítulo 2).

Esta posición no parece muy razonable. Independientemente de que no es claro por qué un árbol que requiere postular más convergencias sería menos simple que otro que requiera menos (no estaría claro qué noción de simplicidad está en juego), puede afirmarse lo siguiente. La apelación a la simplicidad en ciencias se realiza en general cuando las dos hipótesis en comparación son empíricamente equivalentes. En ese caso, un posible factor de decisión entre ellas es su simplicidad (como sea que se la entienda). Sin embargo, a pesar de que todos los árboles (bajo alguna asignación de caracteres) son capaces de dar cuenta de la distribución observada de homologías, no son empíricamente equivalentes en cuanto a la historia evolutiva que postulan para dar cuenta de ella. Si bien tal historia evolutiva (el patrón de diversificación) no suele ser "observable", ya que los eventos de especiación relevantes suelen estar en el pasado remoto, sí existen casos en donde es posible acceder a ese patrón independientemente de la cladística (véase más abajo).

conjuntamente) las leyes contienen/restringen a todos los conceptos a través de conceptos definidos —no conteniendo *explícitamente* a todo concepto de la teoría. Podría argumentarse sin embargo que, al ser eliminables los conceptos definibles, las leyes podrían reformularse (de modo mucho más engorroso) para contener explícitamente a todo concepto de la teoría.

Otra posición posible consiste en sostener que las leyes fundamentales están justificadas *teóricamente*, dado lo que sabemos acerca de la evolución a través de teorías independientes, pero no empíricamente (i.e. no las podemos testear directamente). Por ejemplo, en el ámbito de la genética, sabemos que las tasas de sustitución de nucleótidos en poblaciones suelen ser bajas, y por lo tanto que es menos probable que una especie ancestral con cierto estado para un carácter (p.e. un nucleótido en una posición del genoma) haya dado lugar a dos especies con un estado diferente al original, y en donde tal estado haya surgido dos veces independientemente, en lugar de una vez en algún otro ancestro común (posterior) de ambas. Así, por ejemplo, algunos autores y autoras consideran que aplicar la cladística es hacer una inferencia a la mejor explicación (Quinn, 2016) o que el método de parsimonia es el que minimiza las hipótesis *ad hoc* de homoplasia, dado que la hipótesis nula o explicación por *default* de la similaridad homológica es la herencia del rasgo a partir de un ancestro común (Farris, 1983).

La posición que defenderé aquí es que las leyes fundamentales recién presentadas son leyes fácticas, que poseen el mismo estatus (fáctico) que cualquier otra ley fundamental de cualquier otra teoría (p.e. $F = m \times a$ en la mecánica clásica). Es decir, que se las debe aceptar o rechazar por motivos empíricos, según el éxito aplicativo que tengan. La cladística no tiene nada de peculiar en esto. Nuevamente, el mayor obstáculo para aceptar esta posición (y una razón por la que algunos autores sostuvieron alguna de las anteriores) parecería ser que un test empírico de la cladística requeriría conocimiento independiente del patrón de diversificación real para establecer si este efectivamente se encuentra entre los que la cladística determina como los árboles óptimos. Sin embargo, en la mayoría de los casos, la historia real no es cognoscible por haber ocurrido hace mucho tiempo.

Existen, sin embargo, casos en donde sí es posible obtener esa información. Algunos son los estudios llevados a cabo en el área de la filogenética experimental. Un ejemplo de este tipo de enfoque se encuentra en Hillis et al. (1992).[20] El objetivo de estos autores no era solamente testear a la cladística, sino más bien realizar un experimento crucial que pudiese decidir entre diversos métodos de inferencia filogenética (la cladística entre ellos). Para ello, generaron una filogenia en el laboratorio, que usaron para comparar cuál de los métodos la reconstruía mejor. Dado que generar una filogenia con un grado interesante de diferenciación entre los individuos requiere del

[20] Otro modo de tener un conocimiento independiente de la filogenia sería a través de simulaciones (Huelsenbeck, 1995), pero es más dudoso que pueda hablarse de un test fáctico de la cladística en tal caso (Hillis et al. 1992, p. 589-590; 1993, p. 90)

paso de muchas generaciones los autores usaron un virus (el bacteriófago T7) que tiene generaciones muy cortas, elevando además las tasas de mutación al criarlos en un ambiente con mutágenos. Así, partiendo de una población original, fueron dividiéndola en dos cada ciertos intervalos de tiempo, para obtener la siguiente topología (que conocían previamente a utilizar los métodos):

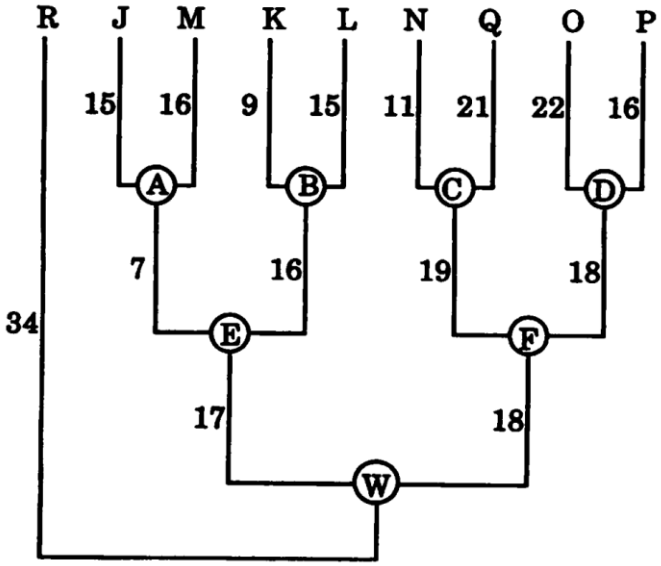


Fig. 3.7 Filogenia conocida del bacteriófago T7, tomada de Hillis et al., 1992, p. 590.

Luego secuenciaron parte de los genomas y los usaron como matriz de datos para los diferentes métodos. Nótese que con 8 taxa terminales (el noveno, R, es un *outgroup*) hay 135.135 topologías posibles, con lo cual la probabilidad de dar con la correcta por azar es mínima.

Los resultados fueron los siguientes. De los métodos utilizados (5 en total, incluyendo parsimonia) todos dieron con la topología correcta. En este sentido, el experimento fracasó *qua* experimento crucial,[21] pero fue exitoso *qua* contrastación de la cladística. En cuanto a la segunda ley fundamental, la cladística logró reconstruir correctamente 1369 de los 1404 estados de los nodos ancestrales (un 97,5%, aproximadamente), incorrectamente 18 (1,3%) y ambiguamente 20

[21] Sober (1993) objetó que el experimento no servía para comparar métodos ya que la filogenia que diseñaron los autores (p.e. una con largos de rama idénticos) era "fácil" para todas las metodologías testeadas —esto era conocido a partir de simulaciones. Los autores respondieron que este experimento era solo una primera aproximación a lo que podía hacerse con filogenias experimentales, y que una contrastación más exhaustiva debía probar con otras topologías (Hillis et al., 1993, p. 91).

(1,4%) —no había comparación aquí con los otros métodos, ya que la cladística era el único que permitía reconstruir los estados de los ancestros.[22]

En suma, la hipótesis de que la evolución procede parsimoniosamente, haciendo que el curso real de la filogenia sea el de menor largo, es una hipótesis fáctica, que puede contrastarse cuando se tiene acceso independiente al patrón de diversificación real. Esto último ocurre en casos como los de filogenias experimentales. Hay que aclarar, sin embargo, que a pesar de las declaraciones iniciales de Hillis et al. (1992) la filogenética experimental sigue siendo un campo relativamente chico, no habiendo demasiados trabajos publicados con este tipo de enfoque. Según Oakley (2009), parte importante de la razón estriba en que generar una filogenia experimental consume mucho más tiempo y recursos que hacer una simulación. Adicionalmente, la discusión actual en torno al mejor método de inferencia filogenético ha disminuido en intensidad. En cambio, la mayoría de los sistemáticos emplean varios métodos a la vez, considerando como más sólidos a los resultados (p.e. a los grupos) que se obtienen en todos ellos. Otra razón por la que las filogenias experimentales no se popularizaron es que los tiempos generacionales requeridos para hacer análisis interesantes obligan a usar siempre el mismo tipo de organismos (virus, y quizás bacterias). A partir de todo esto, es posible concluir que en la gran mayoría de los casos de aplicación reales la cladística no es contrastada, sino que se la usa para retrodecir la historia evolutiva.

3.4. Otros conceptos importantes de CLAD

En esta sección se introducen algunos conceptos adicionales que, si bien no juegan un rol en las aserciones fácticas centrales de la teoría, son ampliamente usados y resulta valioso expresarlos formalmente con mayor claridad. Entre ellos destacan la noción de grupo monofilético y los tipos de morfismos entre estados de caracteres.

Para comenzar, considérese la noción de *grupo monofilético*. En la práctica, esta noción es ambigua. Para comenzar, puede hablarse de grupos monofiléticos *según un árbol* o de grupos monofiléticos a secas (i.e. "reales"). En consecuencia, la noción formal de monofilia que definiré será relativa a un árbol, en el segundo caso quedando relativizada al árbol real A_R. Por otro lado, a

[22] Por supuesto, los 18 casos de estimaciones incorrectas no implican que la contrastación haya fracasado rotundamente, dado que toda teoría se aplica con cierto grado de aproximación / margen de error.

veces se habla de grupos monofiléticos como conjuntos de nodos que incluyen a un ancestro y a todos y solo sus descendientes (esta es la definición de manual, p.e. Wiley & Lieberman, 2011, p. 9), mientras que otras veces se incluye solo a las especies actuales de aquellos grupos (es decir, a aquellas especies actuales que tienen un ancestro común que no es ancestro de ninguna otra especie bajo estudio). Definiré dos nociones, M_1 y M_2, para elucidar a estas dos nociones usadas en la literatura. Ambas serán funciones, tomando como argumento un árbol y devolviendo un conjunto de conjuntos de nodos (solo terminales en el segundo caso). Es decir, M_1: $A \rightarrow \wp(N)$ y M_2: $A \rightarrow \wp(T)$.

M_2 (la monofilia para terminales) es simplemente la función j. Como se dijo más arriba, la jerarquía correspondiente a un árbol (i.e. $j(A_i)$) contiene a los grupos monofiléticos (en el primer sentido) para ese árbol. Es decir a los conjuntos de terminales para los cuales existe un nodo interno que lleva a todos ellos (y solo ellos) a través de algún camino. La definición de M_1 es idéntica, modificando T por N:

$$M_1(A_i) = \{X \in \wp(N) \,/\, \exists n \in N\ (\forall x \in X,\ \exists p \in P\ (p = \langle n, \ldots, x\rangle) \wedge \forall x \in N - X,\ \neg\exists p \in P\ (p = \langle n, \ldots, x\rangle))\}$$

Pasando a los tipos de morfismos de caracteres, las nociones de apomorfismo y plesiomorfismo pueden definirse formalmente del siguiente modo. Como se explicó en el capítulo anterior, un estado de un carácter es apomórfico en un nodo cuando su ancestro inmediato posee un estado distinto para el mismo carácter. Si un nodo n_x en un árbol A_i no es el nodo raíz, los axiomas impropios 4.3 y 4.6 implican que hay un único nodo n_y tal que $\langle n_y, n_x\rangle \in D(A_i)$ — n_y siendo el ancestro inmediato de n_x según A_i. Por tanto, si A_i es un árbol, d_j una optimización y $d_j(n_x, C_k) = c_{k,m}$ entonces diremos que $c_{k,m}$ es apomórfico para n_x (en A_i, bajo d_j) si y solo si $d_j(n_y, C_k) \neq c_{k,m}$ para el n_y tal que $\langle n_y, n_x\rangle \in D(A_i)$. En caso contrario, el estado es plesiomórfico (i.e. si $d_j(n_y, C_k) = c_{k,m}$).

La noción de sinapomorfía es un poco más compleja. Las sinapomorfías no pueden ser simplemente apomorfías compartidas, como a veces se afirma. Es decir, si n_{x1} y n_{x2} son descendientes inmediatos de un nodo n_y, y ambos comparten la apomorfía $c_{k,m}$, ello significa que el caso (a) en la siguiente figura es lo que ocurre:

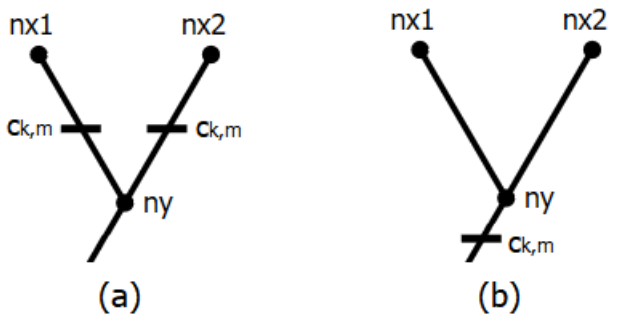


Fig. 3.8. (a) Apomorfía compartida por n_{x1} y n_{x2}; **(b)** Sinapomorfía de n_{x1} y n_{x2}.

cuando lo que se desea expresar con la noción de sinapomorfía es más bien el caso (b). Es decir, lo que es compartido por n_{x1} y n_{x2} es una apomorfía *del ancestro*, no propia. Adicionalmente, la sinapomorfía se predica usualmente de grupos de terminales. Por ejemplo, en siguiente árbol y bajo la siguiente optimización, se diría que el estado 1 de C_1 es una sinapomorfía de $\{t_2, t_3, t_4, t_5\}$, mientras que el estado 1 de C_2 es una sinapomorfía de $\{t_2, t_3, t_4\}$

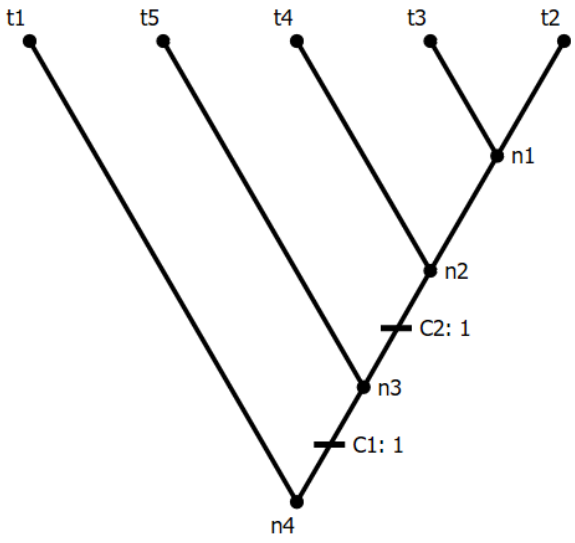


Fig. 3.9. Dos sinapomorfías de grupos

De ese modo, la definición de sinapomorfía aplicará a grupos de terminales más que a nodos individuales del cladograma. Diremos que un estado de un carácter es sinapomórfico para un grupo de terminales (en un árbol y bajo una asignación de caracteres) cuando el estado es apomórfico

para su ancestro común más reciente. Nótese que, nuevamente, la definición será inaplicable cuando el ancestro común más reciente sea el nodo raíz.

Formalmente, puede definirse al ancestro común más reciente de un conjunto X de terminales (notado $acmr(X)$) como el nodo n_x tal que:

(i) existe un camino desde n_x hasta todo miembro de X

(ii) para todo nodo $n_y \neq n_x$, si n_y satisface (i) entonces existe un camino desde n_y hasta n_x

De ese modo, un estado es sinapomórfico para un grupo X si y solo si es apomórfico para $acmr(X)$. Podría agregarse algún requisito adicional a esta definición, como ser, que además todos los terminales de X compartan el estado en cuestión, que (además de lo anterior) no ocurran reversiones (p.e. el ancestro común más reciente del grupo posee $c_{k,m}$ y en algún camino, el estado cambia de $c_{k,m}$ a $c_{k,n}$, y luego vuelve a cambiar a $c_{k,m}$ antes de llegar a ningún terminal). Si estos requisitos adicionales son exigidos por los sistemáticos al hablar de sinapomorfía no es claro, ya que la noción informal de sinapomorfía es relativamente vaga.

Del mismo modo, diré que un estado de un carácter es simplesiomórfico para un grupo X si y solo si es plesiomórfico para $acmr(X)$. Estas dos definiciones son precisas y permiten decidir sobre los casos confusos, como el siguiente (que contiene un único carácter, con estados 0 y 1, mapeado sobre el árbol):

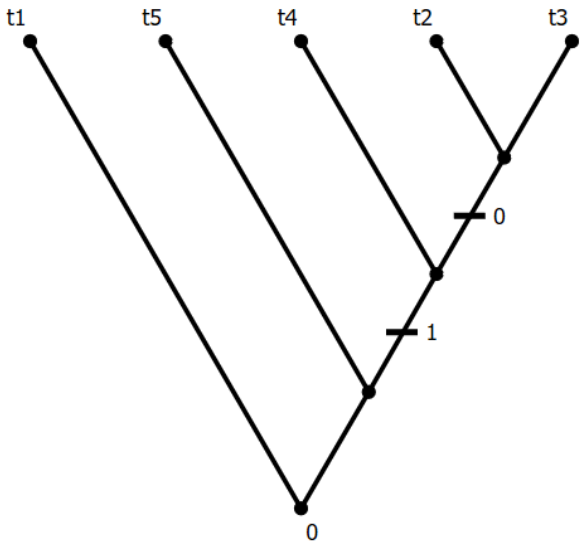


Fig. 3.10. Ejemplo intuitivamente problemático desde el punto de vista de la sinapomorfía.

En este caso, mis definiciones dirían que el estado 0 es simplesiomórfico para {t_2, t_3, t_4, t_5} (a pesar de que uno de esos taxa, t_4, no exhibe ese estado) y sinapomórfico para {t_2, t_3}.

Cabe reiterar, por último, que las nociones de apomorfía, plesiomorfía, sinapomorfía y simplesiomorfía son relativas a un árbol y una asignación de estados cualquiera, que no necesariamente son el árbol óptimo y la asignación óptima para ese árbol. Los sistemáticos toman como significativas solamente a aquellas apomorfías, etc. del árbol y la asignación óptima. Si hay más de un árbol óptimo, se toman como informativas solamente a aquellas apomorfías, etc. de grupos que aparecen en todos los árboles. Del mismo modo, si para un árbol existe más de una asignación óptima, se toman como informativas solo a aquellas apomorfías, etc. que aparecen en todas ellas (i.e. no se tienen en cuenta optimizaciones ambiguas de nodos, en el sentido discutido en la sección 3.6.1.).

3.5. Especializaciones, T-teoricidad y condiciones de ligadura

3.5.1. Especializaciones

La noción de especialización del estructuralismo ya fue introducida en la sección 3.3.1 (un tratamiento formal más preciso puede encontrase en Balzer et al., [1987] 2012, pp. 224-234). A modo de resumen de lo dicho, la idea es que las leyes especiales agregan contenido empírico (i.e. restricciones sobre la clase de modelos) para distintos tipos de casos de aplicación de la ley fundamental. La posesión de especializaciones es lo que da un carácter unificador a las teorías, en el sentido de poder tratar bajo un mismo marco conceptual a fenómenos de distinto tipo (piénsese en objetos en caída libre, trayectorias de planetas, etc. en el caso de la mecánica clásica). Si bien **CLAD** es unificadora en el sentido de que es aplicable a distintos tipos de datos (genéticos, morfológicos, comportamentales) y niveles biológicos, se la aplica siempre igual.

De ese modo, dado que no posee especializaciones, no es sorprendente que las leyes fundamentales de **CLAD** no satisfagan los requisitos (*c*) y (*d*) de la sección 3.3.1 (ser cuasi-vacuas y funcionar como guía para la formulación de leyes especiales). Estos dos síntomas son típicos de las leyes fundamentales *de teorías que poseen especializaciones*, en donde las leyes fundamentales

postulan restricciones débiles que son luego fortalecidas en las leyes especiales.[23] Por otro lado, al no poseer especializaciones, los criterios para identificar si se cumplen las condiciones (*b*) y (*e*) (valer para toda aplicación intencional y poseer fuerza modal) colapsan en lo primero —ya que (*e*) indicaba que una ley fundamental posee fuerza modal cuando vale en toda aplicación pretendida junto con sus especializaciones. De ese modo, ya que (como se dijo) (*b*) es satisfecha, también lo es (*e*).

Cabría preguntarse, en cambio, si **CLAD** misma no es una especialización de alguna otra teoría. Por ejemplo, Goloboff (1993b) introdujo el método de inferencia filogenética llamado "pesos implicados", que es idéntico a la cladística estándar salvo por el modo de computar el largo de los árboles. En pesos implicados el largo de un árbol también se mide a partir de la sumatoria de los costos de acomodar cada carácter. Pero estos costos son distintos (al costo de acomodar un carácter en un árbol se lo llama "distorsión" en pesos implicados). Para medir la distorsión de un carácter se utiliza una función convexa, que hace que agregar un paso a un carácter que ya tiene mucha homoplasia agregue poco a la distorsión, mientras que agregar un paso a uno con poca homoplasia agregue mucho. La idea es disminuir el error surgido de la selección inicial de caracteres, haciendo que caracteres poco confiables terminen pesando poco en el cómputo del largo.

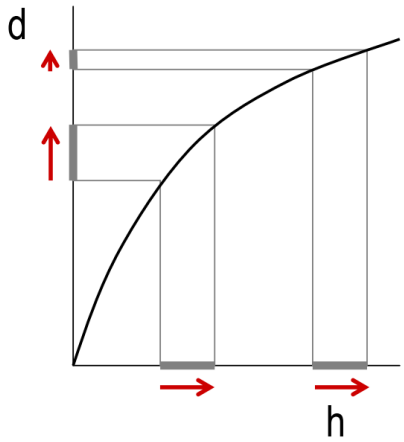


Fig. 3.11. Función convexa de distorsión (*d*) en base a la cantidad de homoplasia (*h*) de un carácter.

[23] De hecho cabe preguntarse si tiene sentido aplicar la etiqueta de "ley *fundamental*" (en lugar de "ley" a secas) a un axioma de una teoría que no contiene leyes especiales. Sea cual fuere el caso, sigue siendo interesante ver en qué medida se exhiben los síntomas usuales de las leyes fundamentales en una teoría sin leyes especiales.

La función matemática usada es $d = h / (h + k)$, en donde h es el peso de ese carácter en el árbol (la cantidad de cambios que presupone) y k es una constante. Por ejemplo, si $k = 2$, agregar una transformación a un carácter que se acomoda perfectamente en la filogenia (con un h inicial de 0) adiciona 1/3 (= 2/6) a la distorsión. Agregarle otro paso hace que su distorsión sea 2/4 (= 3/6), es decir, agrega solo 1/6 a la distorsión (y así sucesivamente).[24] Típicamente los sistemáticos que emplean este método prueban con varios valores de k y se quedan con los resultados que más se repiten en los sucesivos análisis.

Así dado que la estructura (los conceptos) y las leyes son idénticas, excepto por las definiciones de w, l_1 y l_2, podría pensarse que tanto pesos implicados como parsimonia son especializaciones de algún elemento teórico más general, en el que se especifican de distinto modo las funciones de optimización. Sin embargo, en una inspección más cercana muestra que esto no es el caso: parsimonia y pesos implicados son teorías *rivales*. Tienen exactamente al mismo dominio de aplicaciones pretendidas, no son restricciones de una ley más abstracta operando para distintos tipos de casos del dominio de aplicaciones pretendidas.

3.5.2. T-teoricidad

El problema de la teoricidad (cuáles son y cómo adquieren semántica empírica los términos teóricos) es un problema clásico de la filosofía de la ciencia. Es ampliamente sabido que la distinción teórico/observacional de la concepción estándar de las teorías científicas (véase Hempel, 1958 para un resumen) se encuentra con muchas dificultades. Posteriormente a la caída del empirismo lógico como programa de investigación, ha habido intentos por reformular el criterio de teoricidad que este había propuesto (p.e. Hempel, 1970). Los estructuralistas han propuesto una de tales reformulaciones (Balzer et al., [1987] 2012, pp. 98–128; Sneed, 1971).

Su noción de teoricidad descansa sobre tres puntos fundamentales. En primer lugar, por diferentes motivos se deja de lado la cuestión de la observabilidad, adoptando una distinción entre conceptos teóricos y no-teóricos (véanse Bar-Hillel, 1970; Hempel, 1970; Lewis, 1970; Putnam, 1962). En segundo lugar, esta distinción teórico/no-teórico no es absoluta, sino que está relativizada a teorías. Esto es, un concepto puede ser teórico en una teoría y no-teórico en otra. Y tercero, los

[24] Un refinamiento del aparato matemático usado en pesos implicados puede encontrarse en Goloboff (2014).

estructuralistas dicen que un concepto *C* es T-teórico (teórico en una teoría T) si y solo si todo método de determinación de *C* presupone T; en cambio es T-no-teórico si y solo si no es T-teórico (i.e. si existe un modo de determinar *C* que es independiente de T). La noción de método de determinación será introducida con mayor detalle en la sección siguiente, pero basta aquí con decir lo siguiente. Determinar un concepto significa averiguar su extensión.[25] Una determinación de un concepto *C* presupone una teoría T cuando uno aplica las leyes de T para encontrar la extensión de *C* —o mejor dicho, cuando la determinación involucra asumir que T es verdadera, lo cual ocurre cuando se aplican las leyes (axiomas propios) de T para encontrar el valor de *C*.

Un ejemplo sería el siguiente. La filogenia real (A_R en la reconstrucción) sería **CLAD**-no-teórica, ya que es posible (en algunos casos al menos, como se argumentó en 3.3.4) conocerla sin aplicar la cladística. Por supuesto también es posible conocerla (total o parcialmente) a partir de una aplicación de **CLAD**. Una contrastación de la cladística, del estilo de las comentadas en 3.3.4, consiste justamente en determinar A_R tanto teórica como no-teóricamente y ver si los resultados coinciden (i.e. comparar la filogenia conocida independientemente con la topología *output* de la cladística). En ese sentido, la filogenia real pertenece a la "base empírica" de **CLAD**, ya que puede usarse para contrastarla. Sin embargo, que pertenezca a la base empírica no significa que sea observacional en el sentido de los empiristas lógicos (i.e. que esté libre de *toda* teoría) sino solo que está libre de la teoría para la cual es no-teórica, pudiendo estar cargada de *otras* teorías.

Si se observan los modelos potenciales (el lenguaje) de **CLAD**, hay algunos conceptos que son claramente no-teóricos: *T*, *C*, *$* y d_T. Todos ellos funcionan como *input* de un análisis cladista. Sin tenerlos previa e independientemente establecidos el análisis no puede ni siquiera comenzar. Adicionalmente, como se argumentó en 3.3.4, A_R y d_R serían **CLAD**-no-teóricos.

Por otro lado hay una serie de conceptos cuyo estatus de teoricidad es a primera vista más dudoso. Ellos son *A*, *I*, *J*, *N*, *AO*, *AOj*, δ, δ_O, *j*, l_1, l_2 y *w*. Parecería, por ejemplo, que determinar el conjunto de los árboles filogenéticos *A* involucra usar los axiomas impropios de **CLAD** y por tanto que *A* es **CLAD**-teórico. Sin embargo, la mayoría de estos son simplemente conceptos definidos. Ninguno presupone las leyes fundamentales. Es decir, determinarlos no implica asumir que **CLAD**

[25] Para una constante de individuo esto es averiguar el individuo del dominio que la constante denota; para conceptos cualitativos (representados por letras de predicado unarias) el conjunto de elementos del dominio; para relaciones *n*-arias las *n*-tuplas del dominio que la componen; para funciones *n*-arias los pares ordenados de una *n*-tupla y un valor. También es posible hablar de determinación en casos en donde no se determina la extensión completa del concepto sino, por ejemplo, el valor de una función para un argumento.

es verdadera o empíricamente adecuada. Por ejemplo, dado el conjunto de terminales *T*, *A* queda automáticamente determinado (es el espacio de todos los árboles posibles). Del mismo modo, dados los caracteres *C* y la asignación de estados para terminales d_T, el conjunto de las funciones de asignación completas δ queda automáticamente determinado, aplicando su definición (es el espacio de todas las asignaciones posibles); dada la matriz de costos *$* y lo anterior, la función *w* queda determinada (es el peso de cada eje posible), etc. Ninguna de estas determinaciones implica asumir que el árbol y la asignación reales coinciden con los óptimos de **CLAD** (como sí lo asume la determinación teórica del árbol real A_R, comentada arriba).[26] Por tanto, puede concluirse que **CLAD** no tiene conceptos T-teóricos.

Esta característica (que **CLAD** no tenga conceptos teóricos) tampoco debería ser sorprendente, ya que no tiene especializaciones. Como se dijo arriba, en general, las teorías que contienen concepto T-teóricos suelen tenerlos débilmente restrictos en las leyes fundamentales, encontrándose las restricciones más sustantivas en las especializaciones. Esto es lo que suele dar a las leyes fundamentales el carácter cuasi-vacuo mencionado arriba. Si bien es sencillo encontrar casos de teorías sin conceptos teóricos y con especializaciones (p.e. la teoría de la selección natural reconstruida por Ginnobili, 2018; véase Ginnobili & Carman, 2016 para más sobre este punto) no lo es encontrar lo inverso, casos de teorías sin especializaciones y con conceptos T-teóricos.[27]

De ese modo, la presentación estructuralista de los modelos parciales de **CLAD** (la clase de los modelos determinada por la especificación del lenguaje *empírico* de la teoría) sería simplemente $\mathbf{M_{pp}}(\mathbf{CLAD}) = \mathbf{M_p}(\mathbf{CLAD})$.

Otras cuestiones en torno a la T-teoricidad de **CLAD** serán discutidas en la sección 2 del capítulo 6.

3.5.3. Condiciones de ligadura

Las condiciones de ligadura constituyen otro tipo de restricción sobre las interpretaciones posibles de la teoría. Los axiomas impropios, las leyes fundamentales y las leyes especiales son

[26] Algunos de los métodos de determinación de CLAD serán de hecho reconstruidos (en un lenguaje de programación) en el capítulo 5. Ahí se verá aún más claramente que su aplicación no requiere presuponer a las leyes fundamentales.
[27] Aunque esto es meramente una generalización empírica, no veo motivos *a priori* que impliquen que ello es conceptualmente imposible.

proporcionadas a través de axiomas que deben ser satisfechos por cada modelo individualmente. En cambio, las condiciones de ligadura imponen restricciones sobre la aceptabilidad *conjunta* de modelos. Es decir, desde un punto de vista sintáctico, pueden pensarse como axiomas formulados en un meta-nivel que cuantifican sobre los modelos de la teoría.

Un ejemplo típico son las condiciones de ligadura de igualdad, que establecen que, bajo ciertas condiciones, un mismo término en dos interpretaciones distintas debe tener la misma denotación. Un caso de ello sería la condición de que un mismo cuerpo en un mismo momento debe poseer la misma masa, si tal cuerpo está presente en dos aplicaciones distintas de la mecánica clásica (p.e. la masa del planeta Tierra cuando es usada para calcular la trayectoria de dos objetos en caída libre).

La condición de ligadura más importante para **CLAD** se relaciona con el método de *outgroups* de enraizamiento de una topología (Nixon & Carpenter, 1993, 1996, véase la sección 2.3.5). Recuérdese que este método consiste en incluir al menos dos *outgroups* (taxa externos al grupo de interés) en un análisis filogenético, tal que se crea que el grupo de interés (o *ingroup*) es monofilético con respecto a los *outgroups*. Si algún enraizamiento del árbol no-enraizado de menor largo da como resultado un *ingroup* monofilético, entonces tal árbol se enraíza en aquella rama. Si ello no pasa, entonces se descarta la hipótesis de monofilia del *ingroup* y se eligen otros *outgroups*.

Lo que está por detrás de esto es la siguiente idea. Si un análisis cladístico dictamina que las especies A y B son más cercanas entre sí que con C en el árbol inferido como el real, entonces ese patrón de diversificación debe mantenerse en otros análisis. Si ello no es el caso (p.e. figura 3.12), entonces alguna de las aplicaciones infirió un árbol incorrecto.

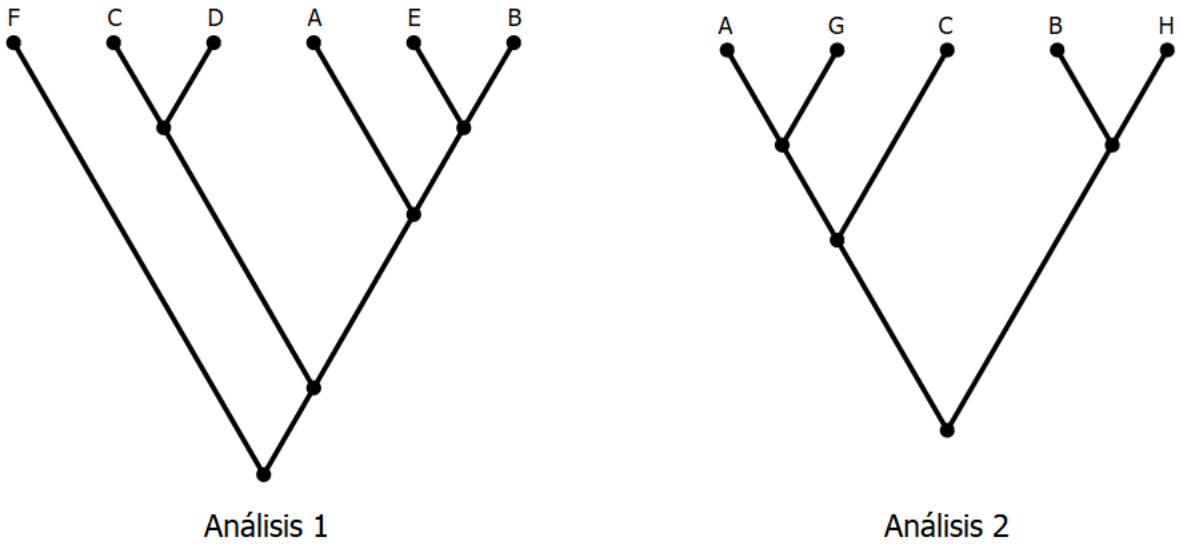


Fig. 3.12. Árboles incompatibles de análisis distintos.

Esta condición de ligadura es más fácilmente formulable formalmente usando la noción de jerarquía. Que A y B son más cercanos entre sí que con C en una jerarquía J_i puede expresarse diciendo que existe algún conjunto Z tal que A ∈ Z, B ∈ Z y C ∉ Z. En el caso de la figura 3.12, los árboles son incompatibles porque el conjunto {A, B, E} (que incluye a A y B pero excluye a C) pertenece a la jerarquía correspondiente al primero, mientras que el conjunto {A, C, G} (que incluye a A y C pero excluye a B) pertenece a la jerarquía correspondiente al segundo. Esto es, el primero afirma que A y B son filogenéticamente más cercanos entre sí que con C, mientras que el segundo afirma que A y C son filogenéticamente más cercanos entre sí que con B.

De ese modo, la condición de ligadura quedaría formulada del siguiente modo:

> **C_1(CLAD)**: $X \in \mathbf{C_1}$ si y solo si $\emptyset \neq X \subseteq \mathbf{M_p}$**(CLAD)** y para toda $x, y \in X$, y todos t_1, t_2 y t_3: si $\{t_1, t_2, t_3\} \subseteq T_x \cap T_y$ y existe un Z tal que ($\{t_1, t_2\} \subseteq$ Z y $t_3 \notin$ Z y Z $\in j(A_R)_x$) entonces, existe un W tal que ($\{t_1, t_2\} \subseteq$ W y $t_3 \notin$ W y W $\in j(A_R)_y$)

Informalmente, si t_1, t_2 y t_3 pertenecen al conjunto de los terminales de dos modelos potenciales distintos, y t_1 y t_2 son más cercanos entre sí que con t_3 según el árbol real del primer modelo (i.e. la jerarquía correspondiente al árbol real contiene un conjunto que incluye a t_1 y t_2 y excluye a t_3),

entonces t_1 y t_2 también deben ser más cercanos entre sí que con t_3 en el árbol real del segundo modelo (i.e. su jerarquía correspondiente debe incluir un conjunto que haga lo mismo).

Una segunda condición de ligadura afirma que si un mismo taxón pertenece a dos análisis distintos, entonces si un mismo carácter está incluido en ambos, el taxón debe poseer el mismo estado para ese carácter en los dos análisis. Formalmente:

> **C**$_2$(**CLAD**): $X \in$ **C**$_2$ si y solo si $\emptyset \neq X \subseteq$ **M**$_\mathbf{p}$(**CLAD**) y para toda $x, y \in X$, todo t y todo c: si $t \subseteq T_x \cap T_y$ y $c \subseteq C_x \cap C_y$ entonces $d_T(t, c)_x = d_T(t, c)_y$.

Nótese que, en ambos casos, las condiciones de ligadura recién formuladas operan sobre / restringen a conceptos **CLAD**-no-teóricos (y no podría ser de otro modo, ya que la teoría no contiene conceptos T-teóricos). Usualmente las condiciones de ligadura operan sobre conceptos T-teóricos (p.e. la masa en la mecánica newtoniana). Cabe preguntarse entonces si estas restricciones son efectivamente restricciones *de **CLAD*** o si están presupuestas por ella (siendo parte de alguna teoría subyacente). Por mor de la completud, se las deja aquí como parte de la reconstrucción. Sería un resultado interesante que una teoría T contuviese condiciones de ligadura sobre conceptos T-no-teóricos.

Considerando las dos condiciones de ligadura anteriores, la condición de ligadura global de **CLAD** queda entonces expresada como:

> **GC**(**CLAD**) = **C**$_1 \cup$ **C**$_2$

A partir de todo lo dicho, el elemento teórico básico de **CLAD** puede formularse del siguiente modo:

> **K**(**CLAD**) $\coloneqq \langle$**M**$_\mathbf{p}$(**CLAD**), **M**(**CLAD**), **M**$_\mathbf{pp}$(**CLAD**), **GC**(**CLAD**)$\rangle$
> **I**(**CLAD**) $\subseteq$ **M**$_\mathbf{pp}$(**CLAD**)
> **T**(**CLAD**) $\coloneqq \langle$**K**(**CLAD**), **I**(**CLAD**)$\rangle$

3.6. Métodos de determinación

La noción de *método de determinación* es sumamente importante para la concepción estructuralista de las teorías. Como se expuso en la sección anterior, La T-teoricidad de un concepto se establece en base a la existencia de métodos de determinación que no presupongan las leyes. Adicionalmente, en ocasiones se dice que los métodos de determinación constituyen el sentido (fregeano) de los términos científicos, y que son parte fundamental de su significado (Moulines, 1998). Por otro lado, los métodos de determinación son sumamente importantes para los propios científicos. Buena parte de la práctica (de desarrollo teórico) de los científicos consiste en buscar más y mejores métodos de determinación para los conceptos de las teorías que usan.

La idea presistemática por detrás de este concepto es, como se explicó anteriormente, que un método de determinación consiste en un modo sistemático de averiguar la extensión de un concepto. Esta noción requiere de una elucidación más precisa. En la primera subsección se da la elucidación estructuralista estándar de ella, la cual puede encontrarse en Balzer et al. ([1987] 2012, pp. 113–119). En el capítulo 6 se proveerá una elucidación alternativa.

3.6.1. Elucidación estructuralista estándar

En la elucidación estándar, un método de determinación consiste en una clase de modelos. Más precisamente, un método de determinación es una clase de modelos que tienen la estructura:

$$\langle D_1, \ldots, D_n, R_1, \ldots, R_m, f_1, \ldots, f_k, t \rangle$$

En donde $D_1, \ldots, D_n$ son dominios, $R_1, \ldots, R_m$ son relaciones, $f_1, \ldots, f_k$ son funciones y t es el concepto a determinar. Es decir, cada modelo de esta clase será un modelo potencial de alguna teoría (que puede ser la teoría que se está reconstruyendo u otra, cuando el método de determinación es T-no-teórico). Adicionalmente, los modelos de esta clase deben satisfacer dos condiciones. Por un lado, los valores de t deben depender sistemáticamente de los valores de $D_1, \ldots, f_k$; por otro, los valores de t deben quedar unívocamente determinados por los valores de $D_1, \ldots, f_k$. Es decir, conocer las extensiones de los dominios, relaciones y funciones debe permitir encontrar

unívocamente el valor de *t*, en muchos casos. Esto suele expresarse diciendo que todos los modelos de la clase satisfacen alguna oración ϕ, tal que si un modelo de la clase satisface $\phi(D_1, \ldots, D_n, R_1, \ldots, R_m, f_1, \ldots, f_k, t)$ y $\phi(D_1, \ldots, D_n, R_1, \ldots, R_m, f_1, \ldots, f_k, t')$, entonces $t = t'$.

Lo dicho en la sección 3.5.2 puede expresarse ahora con mayor claridad. Un método de determinación M presupone una teoría T si y solo si todo miembro de M es un modelo actual de T. Esto implica, como se dijo arriba, que la determinación o medición llevada a cabo (informalmente hablando, al aplicar la oración ϕ para obtener el valor de *t*) satisface las leyes (axiomas impropios, leyes fundamentales y especializaciones) de T.

En el capítulo 4 se verá que el programa de computación que se introduce en la presente tesis trata de un modo distinto al estándar a los métodos de determinación —como algoritmos en lugar de clases de modelos. En el capítulo 6 me encargaré de elucidar más precisamente la noción de método de determinación implementada en el software presentado en el capítulo 4. Esta divergencia respecto del tratamiento estándar se debe a distintos motivos (que serán explorados luego). Pero uno de ellos consiste en que algunos métodos de determinación de **CLAD** (o más bien de cosas que informalmente identificaríamos como métodos de determinación) parecen quedar mejor o más naturalmente reconstruidos como algoritmos que como clases de modelos —p.e. las exposiciones estándar los presentan como algoritmos, y no es claro el modo en el que se los podría reformular como clases de modelos. En la subsección siguiente se brinda un ejemplo de un método tal.

3.6.2. Ejemplo de un métodos de determinación de **CLAD**

Como se explicó anteriormente, el largo de un árbol ($l_2(A_i)$) está definido como el mínimo costo de acomodar a todos los caracteres bajo toda asignación posible de estados a los nodos ancestrales (i.e. el mínimo $l_2(A_i, \delta_x)$ para todo $\delta_x \in \delta$). En el ejemplo de la sección 2.3.3, ello se determinó simplemente probando toda combinación posible de asignaciones, y midiendo el largo_1 resultante. Sin embargo, como se notó arriba (sección 3.2.5) la cantidad de asignaciones posibles crece extremadamente rápido con el número de taxa y de caracteres. De ese modo, probar todas las asignaciones posibles se vuelve rápidamente ineficiente (e incluso imposible) computacionalmente. Los cladistas, sin embargo, divisaron métodos para encontrar las

asignaciones óptimas (y por tanto para medir el largo de un cladograma) que no involucran revisar toda asignación posible.

El método más general, propuesto por Sankoff (1975), funciona para cualquier conjunto de caracteres y matrices de costo. Pero, aunque sigue siendo mucho más rápido que la prueba de toda asignación posible, aun así suele ser computacionalmente lento. En cambio, los algoritmos propuestos por Fitch (1971) y Farris (1970) (este último mejorado luego por Goloboff, 1993a) son mucho más rápidos, aunque funcionan solo con ciertos tipos de matrices de costo. A continuación se expone el método de Fitch, que permite optimizar un carácter sobre un árbol bajo costos iguales (i.e. cuando pasar de un estado del carácter a otro cuesta lo mismo para todo par de estados diferentes, y cuesta 0 mantenerse en el mismo estado).[28] La razón para elegir este método y no los otros es que es conceptualmente más simple, pero toda conclusión que se extraiga posteriormente para este método podría extraerse también para los otros.

El método consiste en lo siguiente. Dado un árbol y una asignación de estados para los nodos terminales (i.e. dado un $A_i \in A$ y d_T), se realizan dos "pasadas" sobre el árbol, una hacia abajo y otra hacia arriba, que van determinando secuencialmente los estados de los ancestros. Por ejemplo, supóngase que el árbol y la asignación para terminales son los siguientes (utilizando el mismo ejemplo de la sección 2.3.3):

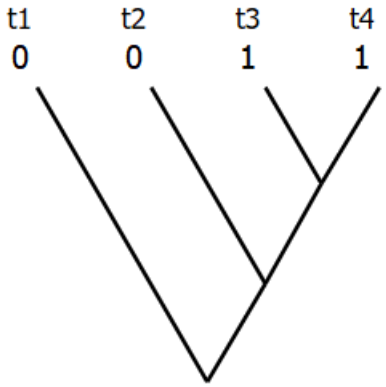


Fig. 3.13 Árbol a optimizar

[28] Si bien esta condición parece muy restrictiva, vale aclarar que la gran mayoría de los caracteres en análisis reales son tratados de este modo.

La pasada hacia abajo consiste en iterar el siguiente proceso hasta que la asignación de todo nodo del cladograma se encuentre determinada (nótese que, en la pasada hacia abajo, las asignaciones posibles para un carácter con dos estados 0 y 1 serán 0, 1 y {0, 1}):

a) Tómese un nodo n_x tal que la asignación para sus dos descendientes n_y y n_z se encuentre determinada.
b) Si n_y y n_z comparten algún estado (p.e. si n_y es 0 y n_z es {0, 1}) entonces asígnese a n_x el estado que comparten.
c) En caso contrario, asígnese a n_x el conjunto de todos los estados de sus descendientes.

La secuencia de asignaciones en la pasada hacia abajo del árbol dado se verá entonces como sigue:

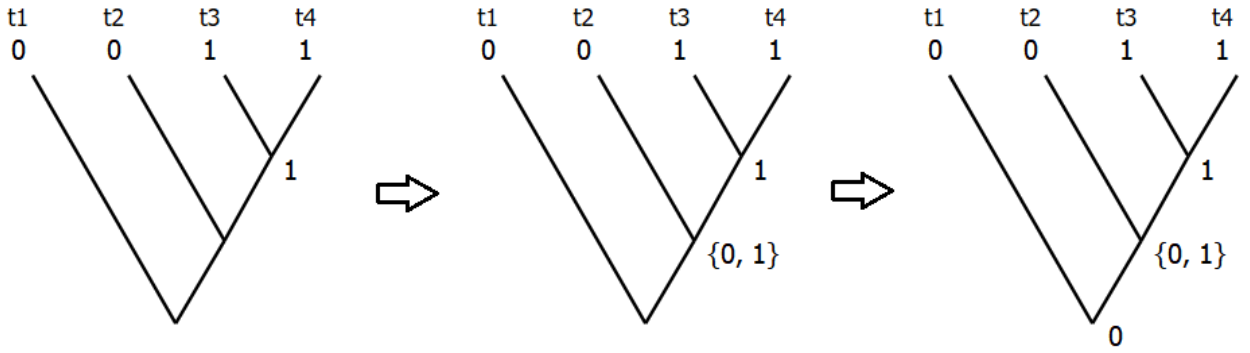


Fig. 3.14. Pasada hacia abajo del algoritmo de Fitch, para el árbol de la figura 3.13.

La interpretación del estado {0, 1} es como una ambigüedad, es decir, un caso en donde tanto el estado 0 como el estado 1 son posibles. El largo del cladograma para este carácter queda establecido ya en la pasada hacia abajo, como la cantidad de veces que fue necesario aplicar la cláusula c). Así, el largo del cladograma se determina como la sumatoria de la cantidad de veces que se aplicó c) para todo carácter de la matriz. En consecuencia, los programas de inferencia filogenética, al hacer una búsqueda de árboles (en donde lo único que interesa es conocer el largo del árbol en cuestión), realizan solo la pasada hacia abajo.

La pasada hacia arriba sirve para resolver (donde es posible) las ambigüedades en la optimización. Para ello se parte del nodo raíz, y se visitan secuencialmente los nodos internos

adyacentes que no hayan sido visitados aun, hasta recorrer todos los nodos internos, haciendo lo siguiente:

a) Si la optimización del nodo no es ambigua, se pasa al nodo siguiente
b) Si la optimización es ambigua (i.e. hay más de un estado posible), se prueban todos los estados del conjunto y se observa cuántas transformaciones implican para los tres nodos adyacentes (su ancestro y sus dos descendientes). Se eligen los estados que minimizan la cantidad de transformaciones (que puede ser uno o más de uno del conjunto).

En el caso de la figura 3.14, el único nodo interno ambiguo es el intermedio. Nótese que el ancestro posee el estado 0 y sus descendientes los estados 0 y 1. Si se pone un 0 en ese nodo, entonces habrá ocurrido solo una transformación, hacia el 1 de la derecha. En cambio, si se pone un 1, habrán ocurrido dos, una desde el ancestro hacia él, y otra desde él hacia el descendiente de la izquierda. En consecuencia, la ambigüedad se resuelve en este caso hacia un 0. Nótese que la asignación coincide con la que se había obtenido en el capítulo 2, probando todas las posibles.

Una reconstrucción de este método de determinación será dada en el capítulo 5.

3.7. Conclusiones

En este capítulo, se ha presentado una reconstrucción formal estructuralista de la cladística. Se mencionaron, en primer lugar, algunos tratamientos formales previos en el área de la matemática filogenética, sobre los que la presente reconstrucción edifica.

En segundo lugar, se introdujeron las nociones de axioma impropio y modelos potenciales del estructuralismo y se dieron estos para la cladística.

Luego se presentó la noción de ley fundamental (y la correspondiente clase **M** de los modelos) y se establecieron las leyes fundamentales de la cladística. Estas afirman, resumidamente, que el árbol y la asignación de caracteres encontrados como óptimos en el análisis coinciden con las reales. Se argumentó luego que el estatus de estos enunciados es el de leyes fácticas, y se ilustró este punto con uno de los modos en los que ellos fueron contrastados, utilizando filogenias experimentales.

A continuación se elucidaron formalmente otras nociones que, si bien no forman parte de la formulación de las leyes, son importantes y frecuentemente usadas por los sistemáticos, de modo que tiene valor precisarlas. Ellas son las nociones de grupo monofilético y los tipos de morfismo (apomorfismo, plesiomorfismo, sinapomorfismo y simplesiomorfismo) de caracteres.

En quinto lugar, se discutió la posibilidad de identificar a las especializaciones, la T-teoricidad de los conceptos y las condiciones de ligadura de la teoría. La ausencia de especializaciones explica por qué fallan dos de los síntomas usuales que exhiben las leyes fundamentales. En cuanto a las condiciones de ligadura, se defendió que el método de enraizamiento a través de la inclusión de *outgroups* es en realidad una constricción sobre modelos tomados conjuntamente (i.e. una condición de ligadura), y no una constricción sobre cada modelo tomado individualmente (i.e. una ley fundamental o especial).

Por último, se introdujo la noción presistemática de método de determinación del estructuralismo así como su elucidación estándar (que apela a clases de modelos), para luego dar un ejemplo de algo que (informalmente) parece un método de determinación de **CLAD**, pero que es difícil de acomodar a tal elucidación estándar.

Los desarrollos introducidos en este capítulo servirán de insumo para capítulos posteriores. En el capítulo siguiente se presentará un programa de computación diseñado para ayudar a contrastar reconstrucciones formales de teorías. En el capítulo 5 se aplica el programa a la contrastación de la reconstrucción introducida en este capítulo, brindando así un caso paradigmático y exitoso de aplicación del programa e ilustrando su fertilidad y adecuación.

Por otro lado, la reconstrucción tendrá por si misma algunas consecuencias interesantes. Por ejemplo, el estatus de T-teoricidad de los conceptos de **CLAD**, discutido en este capítulo, tendrá consecuencias más adelante para la discusión sobre la circularidad en torno al concepto de homología (capítulo 6).

PARTE II - Testeo de la reconstrucción de la cladística mediante el software Reconstructor

Capítulo 4. Reconstructor y la contrastación de reconstrucciones

En este capítulo se presenta una metodología para contrastar reconstrucciones formales de teorías, así como un software para llevarla a cabo. El software, llamado Reconstructor, puede descargarse gratuitamente desde mi sitio web (https://sites.google.com/view/ariel-roffe/software). Se utilizará aquí la versión 2.0.

La metodología consistirá en varios procedimientos, cuyo objetivo, a grandes rasgos, es establecer si la reconstrucción "se comporta igual" que la teoría objeto siendo reconstruida. Me encargaré de volver a esto más preciso a lo largo del capítulo, pero un ejemplo de lo que se quiere decir con ello (i.e. un ejemplo de uno de los procedimientos) es el siguiente. Como se dijo en el capítulo 3, en la metateoría estructuralista, las aplicaciones potenciales de teorías se representan como modelos (en el sentido lógico / formal del término). En ese sentido, es posible decir que una reconstrucción se comporta igual que la teoría objeto cuando los modelos que representan a aplicaciones paradigmáticamente exitosas de la teoría satisfacen la formalización de las leyes. Del mismo modo, si informalmente se cree que una aplicación no es exitosa, entonces el modelo correspondiente deberá no satisfacer la formalización de las leyes.

El modo de llevar a cabo esta metodología en Reconstructor consistirá en cargar la reconstrucción formal de la teoría en el programa (introduciendo el lenguaje formal, las leyes, los métodos de determinación, etc.) y sus aplicaciones (como modelos). Una vez hecho esto, el programa permite realizar diversas operaciones sobre ese *input*, como ser, determinar si los modelos satisfacen las oraciones formales dadas como leyes. De ese modo, la contrastación se efectúa comparando el *output* de Reconstructor con la comprensión informal del usuario del resultado que deberían dar tales operaciones. Se asume para ello que el usuario tiene una comprensión adecuada previa de la teoría objeto.

La estructura de este capítulo será la siguiente. En la primera sección se dan los primeros pasos con el programa, detallando la sintaxis del lenguaje formal que puede utilizarse para cargar axiomas (impropios, leyes fundamentales y especializaciones). En la segunda sección se explica cómo cargar modelos, y cómo consultar por la denotación de términos y la satisfacción de oraciones en ellos. A partir de esto, se introduce el primer método de contrastación de reconstrucciones. La sección 4.3 trata con el modo cargar condiciones de ligadura y evaluarlas en conjuntos de modelos.

La sección 4.4 detalla la semántica no-clásica que utiliza Reconstructor para evaluar oraciones en modelos que contienen fallas de denotación. Las secciones 4.5 y 4.6 introducen el modo como Reconstructor trata a los métodos de determinación (manuales y automáticos, respectivamente). Por último, en 4.7 y 4.8 se ofrecen algunas conclusiones y se brinda un apéndice con una lista de términos por default y abreviaturas de axiomas en Reconstructor.

Para lograr todo esto, se presentará una teoría objeto de juguete como ejemplo (una simplificación de la mecánica de Descartes) y se mostrará cómo realizar todo lo anterior con ella. El archivo con la teoría reconstruida se presenta como material suplementario. Una aplicación del software a una teoría real (la cladística) será ofrecida en el próximo capítulo. Por último, cabe mencionar que este capítulo expande sobre los resultados ya publicados en Roffé (2019b).

4.1. Lenguaje, axiomas y especializaciones

En esta sección se comienza a explicar el funcionamiento del programa Reconstructor. Se muestra cómo especificar el lenguaje formal con el que se va a trabajar, y se introduce la sintaxis que el programa acepta para términos, fórmulas y oraciones. A su vez, se da una lista de términos (de relación y de función) que el software acepta como parte de su lenguaje de base, los cuales serán siempre interpretados de manera estándar en todo modelo (véase la sección 4.2). Se muestra además cómo cargar axiomas (impropios y leyes fundamentales) y especializaciones, cómo consultar al programa si una cadena de símbolos es un término, una fórmula bien formada y una oración del lenguaje, y cómo obtener una representación gráfica de la red teórica.

Para realizar todo esto será conveniente contar con un ejemplo sencillo de una teoría a representar. Utilizaré para ello una simplificación de la mecánica del choque de Descartes ([1644] 1982, [1677] 2004; en adelante MCAR). Una reconstrucción formal más completa de esta teoría puede encontrarse en Lorenzano et al. (2007, con correcciones posteriores en Lorenzano et al. 2008, 2010). En este capítulo no importará tanto la adecuación de la reconstrucción ya que pretendo utilizarla solo como medio para ilustrar la funcionalidad del programa. La aplicación de Reconstructor a una teoría real (la cladística) será llevada a cabo en el capítulo siguiente.

4.1.1. Una teoría objeto de ejemplo

MCAR fue introducida por Descartes para dar cuenta del modo en el que los cuerpos se mueven y colisionan en el espacio. Más específicamente, lo que Descartes buscaba explicar era la velocidad a la que un conjunto de cuerpos se movía en el espacio en distintos momentos.[29] El tiempo es tratado como una colección discreta de instantes. Algunos de los cuerpos pueden chocar con otros cuerpos del conjunto entre dos instantes cualquiera. Esto es, las colisiones mismas son tratadas como una caja negra. Todo lo que a Descartes le importa son las velocidades que los cuerpos tienen antes y después de que la colisión haya tenido lugar.

Para dar cuenta de estos fenómenos, Descartes propone una serie de generalizaciones de diferente alcance. En el nivel más general (lo que consideraremos como la ley fundamental), MCAR postula que la sumatoria de las rapideces (*speeds*, esto es, las velocidades haciendo abstracción de la dirección del movimiento) de los cuerpos se mantiene constante en todo tiempo.[30] Esto es, en un choque puede "transferirse" algo de rapidez de un cuerpo a otro, pero mientras el sistema se mantenga cerrado (mientras ningún cuerpo choque con un cuerpo externo a los cuerpos siendo considerados) la cantidad total se mantiene constante. Como especializaciones (leyes que restringen la clase de los modelos poniendo restricciones adicionales sobre las interpretaciones de los conceptos, véase 3.5.1) consideraré las siguientes dos. Por un lado, si un cuerpo no choca con ningún otro entre dos momentos sucesivos, entonces mantiene su velocidad constante. Por otro, si un cuerpo choca con otro entonces ambos "rebotan" con velocidades opuestas a las que traían.[31]

Formalmente, todo esto puede representarse del siguiente modo. El lenguaje de MCAR, $\mathscr{L}_{\mathbf{MCAR}}$ puede presentarse como $\langle B, T, \Leftarrow, CL, v\rangle$, en donde B y T son conjuntos (un conjunto de cuerpos y de instantes de tiempo, respectivamente), $\Leftarrow$ es una relación binaria entre dos instantes de tiempo (indicará que el primer instante es anterior al segundo), CL es una relación triádica entre dos cuerpos y un instante de tiempo (indicará que ambos cuerpos chocan inmediatamente antes del

[29] Por simplicidad, consideraré al espacio como unidimensional. De ese modo, las velocidades positivas indicarán movimiento de izquierda a derecha, mientras que las velocidades negativas representarán movimiento de derecha a izquierda. Las rapideces serán los módulos de las velocidades. Nótese que estoy usando rapidez en un sentido distinto al que lo usan Lorenzano et al. (2008, 2010).

[30] En la versión de Descartes, lo que se mantiene constante es el producto de las velocidades y los volúmenes de los cuerpos. Para simplificar, asumiré que todos los cuerpos considerados tienen el mismo volumen, y por lo tanto que el concepto de volumen puede obviarse de las leyes.

[31] Nuevamente, esto es una simplificación. En la teoría real, esto solo ocurre cuando las velocidades iniciales eran opuestas. Para otros pares de velocidades iniciales, Descartes presenta otras leyes especiales. Aquí se hará caso omiso de esto, considerando que todos los choques pueden tratarse con la segunda ley especial mencionada.

instante dado) y v es un símbolo de función binario, representando a la función velocidad de un cuerpo en un instante de tiempo.

Con este lenguaje, los axiomas impropios de MCAR (es decir, $\mathbf{M_p}$(**MCAR**)) pueden presentarse como sigue:

(IMP1) $B \neq \emptyset$

(IMP2) $T \neq \emptyset$

(IMP3) $\Leftarrow \subseteq T \times T$, y es irreflexiva, antisimétrica y transitiva

(IMP4) $CL \subseteq B \times B \times T$

(IMP5) $\forall b_x \in B, \forall b_y \in B, \forall t_i \in T$ (si $CL(b_x, b_y, t_i)$ entonces $CL(b_y, b_x, t_i)$)

(IMP6) v: $B \times T \rightarrow \mathbb{R}$

Mientras que las leyes fundamentales (i.e. **M**(**MCAR**)) pueden formularse del siguiente modo:

(LF) $\forall t_i \in T, \forall t_j \in T, \sum_{b \in B} |v(b,t_i)| = \sum_{b \in B} |v(b,t_j)|$

(ESP1) $\forall b_x \in B, \forall t_i \in T,$

si $\nexists b_y \in B$ tal que $CL(b_x, b_y, t_i)$ entonces $v(b_x, t_{i+1}) = v(b_x, t_i)$

(ESP2) $\forall b_x \in B, \forall b_y \in B, \forall t_i \in T,$

si $CL(b_x, b_y, t_i)$ entonces $(v(b_x, t_{i+1}) = -v(b_x, t_i) \wedge v(b_y, t_{i+1}) = -v(b_y, t_i))$

En las subsecciones siguientes se muestra como cargar todos estos datos en Reconstructor.

4.1.2. Lenguaje formal

En esta subsección y las siguientes se muestra cómo cargar **MCAR** en Reconstructor desde cero. Para ello, el primer paso es abrir el programa e ir al menú **File > New theory**. Al hacer esto, se habilitará la pestaña **Language**. En este punto se debe introducir un nombre para la teoría y la cantidad de conceptos que esta contendrá (luego pueden agregarse o quitarse conceptos, de ser necesario). Al presionar **Ok**, aparecerá el número de campos correspondiente al número de

conceptos indicado. Se debe completar cuál será el término usado para denotar tal concepto, el tipo del concepto (dominio, constante, relación o función) y (de corresponder) la aridad. Al hacer click en el segundo **Ok**, se habilitará el resto de las pestañas y ya se podrán cargar axiomas, modelos, etc. Un ejemplo de cómo se ve el programa tras la carga del lenguaje de **MCAR** está en la figura 4.1.

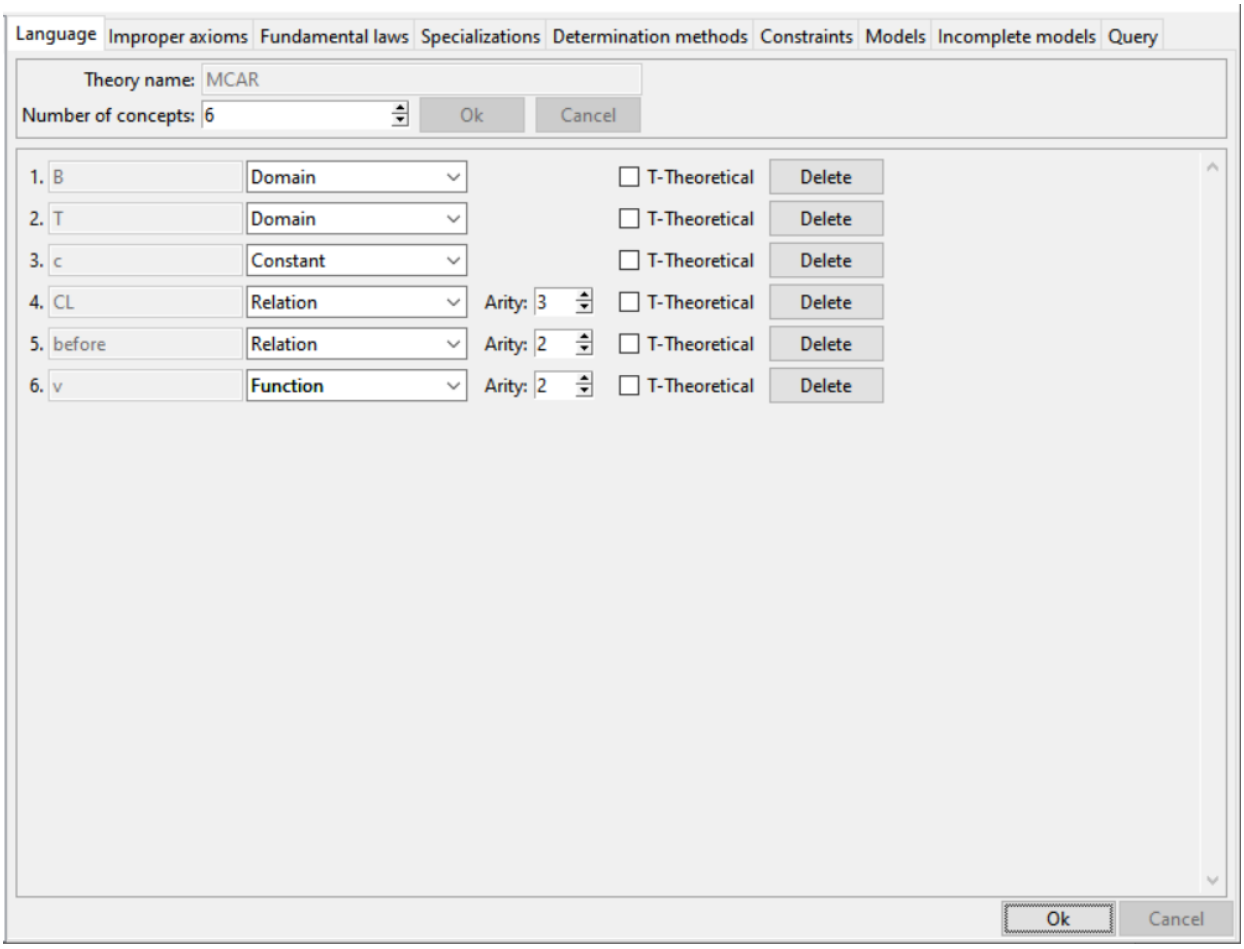


Fig. 4.15. Carga del lenguaje de **MCAR**. Se agregó una constante de individuo *c*, que no forma parte del lenguaje de la teoría, pero que me será útil para ilustrar algunas funciones del programa. Más adelante se agregarán más conceptos (en el archivo suplementario ya se encuentran todos ellos).

También es posible decirle al programa qué conceptos son T-teóricos en la teoría. Esto solo afectará el modo en el que la reconstrucción se imprime a un archivo de LaTeX (una funcionalidad en la que no entraré aquí, de modo que dejaré estos campos en blanco).

4.1.3. La sintaxis de Reconstructor

El siguiente paso en la carga de la reconstrucción es la carga de los axiomas impropios. Esto puede hacerse desde la pestaña **Improper Axioms**. Allí, simplemente se le da un nombre al axioma en el campo superior y se carga el axioma en el inferior (p.e. figura 4.4 abajo). En esta subsección se detalla la sintaxis del lenguaje formal que Reconstructor utiliza para la carga de axiomas.

El programa utiliza un lenguaje base de primer orden, que contiene (entre otras cosas, véase a continuación) a la teoría de conjuntos y la aritmética de los números reales. Es decir, el lenguaje incluye ítems de lenguaje (p.e. el numeral "1", la constante "∅", el símbolo de función "+" y las relaciones "<" y "∈") que luego serán interpretadas de modo estándar en todo modelo. El lenguaje fue elaborado de modo tal que las fórmulas sean sencillas de escribir con un teclado QUERTY. Las fórmulas dadas como input pueden ser:

1. Atómicas

Las fórmulas atómicas pueden ser o bien "Falsum" (la fórmula que siempre obtiene valuación 0), o bien "Verum" (siempre valuación 1), o bien una relación *n*-aria seguida de *n* términos —entre paréntesis y separados por comas. Reconstructor es sensible a mayúsculas y minúsculas (p.e. "falsum" no será considerada una fórmula bien formada).

Las relaciones pueden ser las especificadas por el usuario o relaciones por default. En el primer caso, por ejemplo, *before*(*c*, *c*) y *CL*(*c*, *c*, *c*) serán fórmulas bien formadas, mientras que cosas como *CLccc* y *CL*(*c*, *c*) no lo serán (a la primera le faltan paréntesis y comas separando los términos, la segunda tiene una aridad incorrecta). En el segundo caso, Reconstructor incluye algunas relaciones por default, que son las siguientes:

- "=": la relación binaria de igualdad.
- "<", "<=", ">" y ">=": las relaciones numéricas binarias menor, mayor, menor o igual y mayor o igual.
- "in": la relación conjuntista de pertenencia (∈).
- "subset" y "psubset": las relaciones conjuntistas de inclusión (⊆) e inclusión propia (⊂).

Las atómicas que contienen una relación por default se escriben en notación infija. De ese modo, "*c* = *c*", "*c* <= *c*" y "*c* subset *c*" serán fórmulas bien formadas (nótese que no llevan paréntesis), mientras que "=(*c*, *c*)", "<=(*c*, *c*)" y "subset(*c*, *c*)" no lo serán. En cambio, las atómicas con relaciones del usuario van siempre en notación prefija (como se detalló arriba).

Es posible chequear si algo es una fórmula bien formada para Reconstructor desde el módulo **Query** (la última pestaña de la derecha). Para ello, en la lista desplegable **Item** elíjase **Sentence**, escríbase la cadena de símbolos en la caja de la derecha, y luego elíjase **Formula?** o **Sentence?** (para fórmulas con y sin variables libres) en la lista **Query** de abajo y presiónese el botón **Submit**. Los resultados de la consulta aparecen impresos en la caja de arriba (figura 4.2).

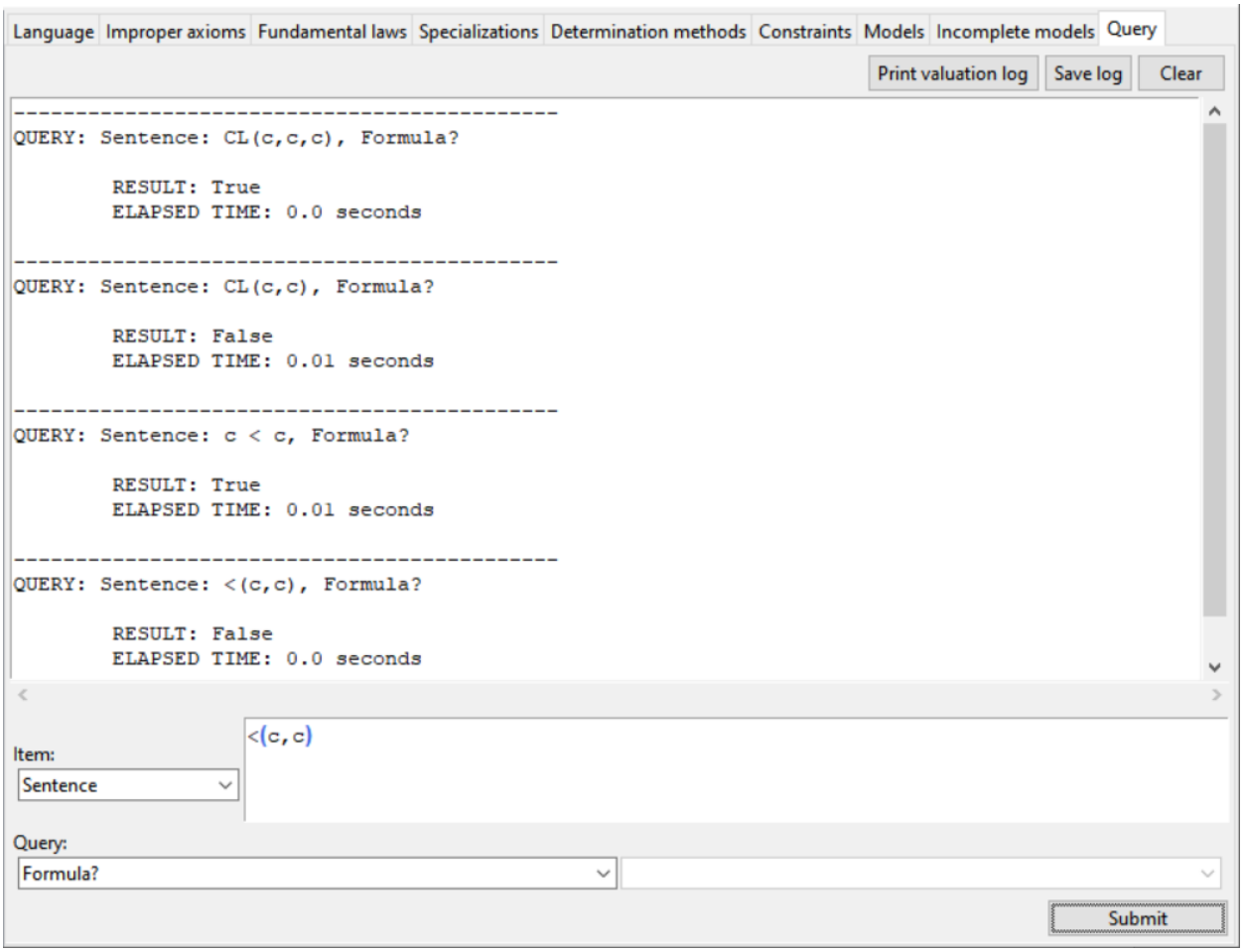


Fig. 4.16. Chequeando si una cadena de símbolos es una fórmula bien formada.

Respecto de las cadenas de símbolos aceptadas como términos, estas pueden ser:

- Variables: “x”, “y” o “z”, o bien alguna de ellas seguida por un número natural (p.e. “x1”, “y234”, “z49”).
- Constantes de individuo, que pueden ser (entre otras):
    - Especificadas por el usuario (*c* en el caso anterior). Reconstructor también trata a los predicados y símbolos de función sueltos como términos (que denotan a sus interpretaciones, las cuales son conjuntos). Así, en **MCAR**, expresiones como *v* y *CL* también son términos.
    - Numerales enteros (“1”, “2”, “3”, …) y racionales (“5.21”, “6.5232”)
    - Las constantes “NAT”, “INT” y “RLS”, que referirán al conjunto de los naturales, enteros y reales, respectivamente.
- Símbolos de función *n*-arios seguidos de *n* términos (entre paréntesis y separados con comas). Nuevamente, pueden ser de diversos tipos:
    - Especificados por el usuario (p.e. *v*(*c*, *c*))
    - Por default. Reconstructor incluye una rica variedad de funciones por default. Algunas pocas se escriben en notación infija (p.e. para la suma “1 + 1” es un término, mientras que “+(1, 1)” no lo es), aunque la mayoría se escribe en notación prefija (p.e. “gcd(x, y)” denotará el máximo común divisor entre dos números *x* e *y*).

El listado completo de términos que Reconstructor acepta por default (así como su tipo, aridad y notación) se encuentra detallado en el Apéndice al final del capítulo (sección 4.8). Al igual que con las fórmulas, es posible chequear si algo es un término correcto desde el módulo **Query**, seleccionando **Term?** en la segunda lista desplegable (figura 4.3).

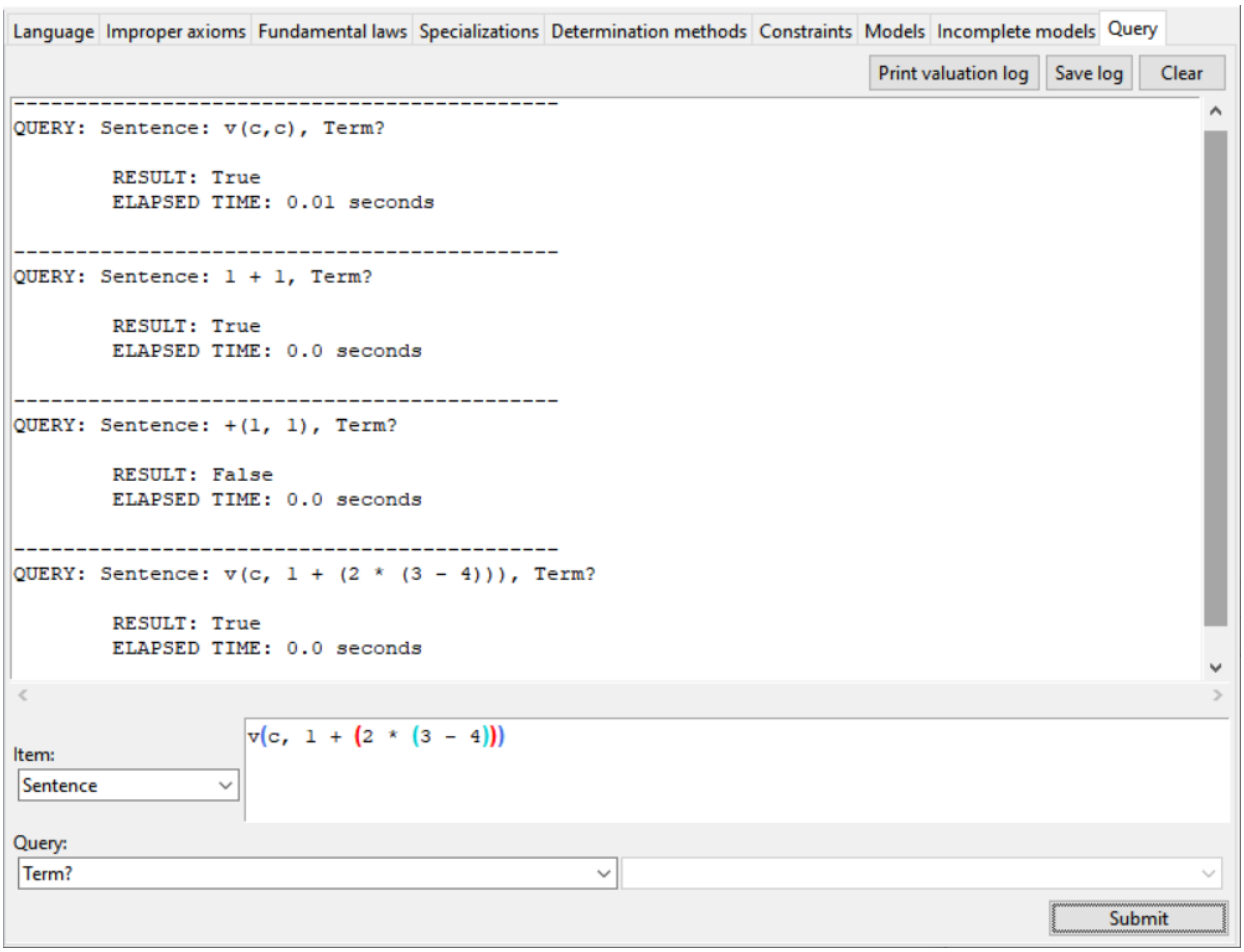


Fig. 4.17. Chequeando si una cadena de símbolos es un término correcto.

2. Fórmula con conectiva unaria

Si "ϕ" es una fórmula bien formada, entonces "¬ϕ" y "°ϕ" también lo son. De ese modo, por ejemplo, "¬Falsum", "¬¬Falsum" y "¬°¬°¬*CL*(*c*, *c*, *c*)" son fórmulas bien formadas de **MCAR**. El operador "°" proviene de la lógica LFI1, y significa que la fórmula que sigue tiene un valor determinado (véase la sección 4.4).

3. Fórmula con conectiva binaria

Si "ϕ" y "ψ" son fórmulas bien formadas, entonces "(ϕ & ψ)", "(ϕ or ψ)", "(ϕ then ψ)" y "(ϕ iff ψ)" también lo son. Para el condicional, el programa también acepta "(if ϕ then ψ)". Los paréntesis externos son necesarios excepto cuando son los externos (existe una excepción más, véase el caso de los cuantificadores). De ese modo, el programa aceptará tanto "(Falsum & (Falsum or Verum))" como "Falsum & (Falsum or Verum)" como fórmulas.

4. Fórmula cuantificada

Por razones de computabilidad, los cuantificadores en Reconstructor deben estar siempre acotados. Si "ϕ" es una fórmula, "x" es una variable y "T" es un término, entonces "forall x in T (ϕ)" y "exists x in T (ϕ)" son fórmulas. En cambio, "forall x ϕ" y "exists x (ϕ)" no lo son, ya que no están acotadas. Cuando la fórmula ϕ lleva paréntesis externos (p.e. cuando su constante lógica principal es una conectiva binaria) no es necesario repetir paréntesis. De ese modo, Reconstructor aceptará tanto "forall x in T ((Falsum & Falsum))" como "forall x in T (Falsum & Falsum)".

Otro punto importante al introducir fórmulas son los espacios en blanco. Por ejemplo si se introduce "forall x in T(Falsum)", dado que falta un espacio entre T y el paréntesis siguiente, Reconstructor interpretará esto como un intento de aplicar T al argumento Falsum, lo cual no es lo que el usuario desea. Presionar la tecla F1 hará que los espacios se resalten en amarillo, lo cual facilita la detección de espacios de más o de menos.

Por último, y en relación con los cuantificadores, Reconstructor permite también introducir fórmulas de la forma "!exists x in T (ϕ)". Estas fórmulas son tomadas como afirmando que existe un *y solo un* x tal que ϕ. Es decir, al introducir una fórmula de esta forma, el programa carga como axioma la oración "exists x in T (ϕ & forall y in T (if [x/y]ϕ then y = x))", en donde [x/y]ϕ es el resultado de sustituir x por y en ϕ.

4.1.4 Carga de axiomas impropios

Con este vocabulario, los axiomas impropios de **MCAR** dados arriba pueden cargarse en Reconstructor del siguiente modo:

Nombre	**Axioma impropio**	**Traducción al lenguaje de Reconstructor**
IMP1	$B \neq \emptyset$	¬B = nullset
IMP2	$T \neq \emptyset$	¬T = nullset

IMP3	$\Leftarrow \subseteq T \times T$, y es irreflexiva, antisimétrica y transitiva	(3.1) before subset cartp(T, T) (3.2) antireflexive(before, T) (3.3) antisymmetric(before, T) (3.4) transitive(before, T)
IMP4	$CL \subseteq B \times B \times T$	CL subset cartp(B, B, T)
IMP5	$\forall b_x \in B, \forall b_y \in B, \forall t_i \in T$ (si $CL(b_x, b_y, t_i)$ entonces $CL(b_y, b_x, t_i)$)	forall x1 in B (forall x2 in B (forall y in T (if CL(x1,x2,y) then CL(x2,x1,y))))
IMP6	$v: B \times T \rightarrow \mathbb{R}$	func(v, B, T, RLS)

Tabla 4.5. Traducción de los axiomas impropios al lenguaje de Reconstructor

Como puede notarse en IMP3 en las traducciones de 3.2, 3.3, 3.4, Reconstructor incluye modos abreviados de introducir ciertas oraciones formales. Por ejemplo, al introducir "antireflexive(before,T)" como axioma, el programa carga en realidad la oración "forall x in T (¬before(x,x))". La lista completa de abreviaturas para axiomas impropios se encuentra en el apéndice de la subsección 4.8.2.

En el caso de "func(v, B, T, RLS)" (IMP6), el programa carga una oración formal más compleja, que expresa cuáles son los tipos de los argumentos y del valor, más el requisito de unicidad (a ningún argumento de la función se le asigna más de un valor). Cabe notar que el requisito de existencia para funciones (a todo miembro del dominio se le asigna al menos un valor) no es impuesto por la expresión "func(…)". Esto se debe a dos motivos. El primero es que, en algunos casos, se quiere permitir que ocurran fallas de denotación en una función; es decir, que haya argumentos para los cuales la función no devuelve ningún valor (que representarán casos de falta de información, véase la sección 4.2.2). El segundo es que el requisito de unicidad debe cuantificar sobre el conjunto del dominio,[32] pero este puede ser un conjunto infinito (p.e. los números naturales). Sin embargo, a pesar de que oraciones de la forma "forall x in NAT (ϕ)" son consideradas fórmulas bien formadas, arrojarán un error cuando se desee evaluarlas en un modelo (nuevamente, por motivos de computabilidad, ya que Reconstructor utiliza una semántica

[32] El requisito de existencia afirma que para todo argumento posible *del dominio* la función devuelve un valor. Formalmente $\forall x \in D\ (\exists y \in COD\ (f(x) = y))$.

sustitucional). De todos modos, si se desea agregar el requisito de existencia, esto puede hacerse manualmente con las herramientas del lenguaje de base.

4.1.5. La ley fundamental

Para la carga de la ley fundamental, las sumatorias y productorias (SUM y PROD) tienen la siguiente sintaxis:

> "SUM([variable especial], [cota inf], [cota sup], [término])" o
> "SUM([variable especial], [cota conjuntista], [término])"
>
> "PROD([variable especial], [cota inf], [cota sup], [término])" o
> "PROD([variable especial], [cota conjuntista], [término])"

Ambas funciones llevan como primer argumento una variable especial (i, necesariamente seguida de un número natural). Si se proveen 4 argumentos a la función (primera forma para ambas SUM y PROD), la variable especial asumirá secuencialmente los valores que van desde la cota inferior hasta la cota superior (ambas cotas deben ser números naturales, y el salto es de a 1 número). Si se proveen 3 argumentos, el programa asume que la cota es un conjunto, y la variable especial asumirá como valor los elementos del conjunto. El término del final puede contener a la variable especial. De ese modo, "SUM(i1, 1, 3, i1)" (interpretado como $\sum_{i_1=1}^{3} i_1$) denotará 6, mientras que "SUM(i2, {1, 2}, SUM(i1, 1, 3, i1))" (interpretado como $\sum_{i_2 \in \{1,2\}} \sum_{i_1=1}^{3} i_1$) denotará 36.

Así, la ley fundamental puede cargarse del modo siguiente (donde "mod" es la función módulo):

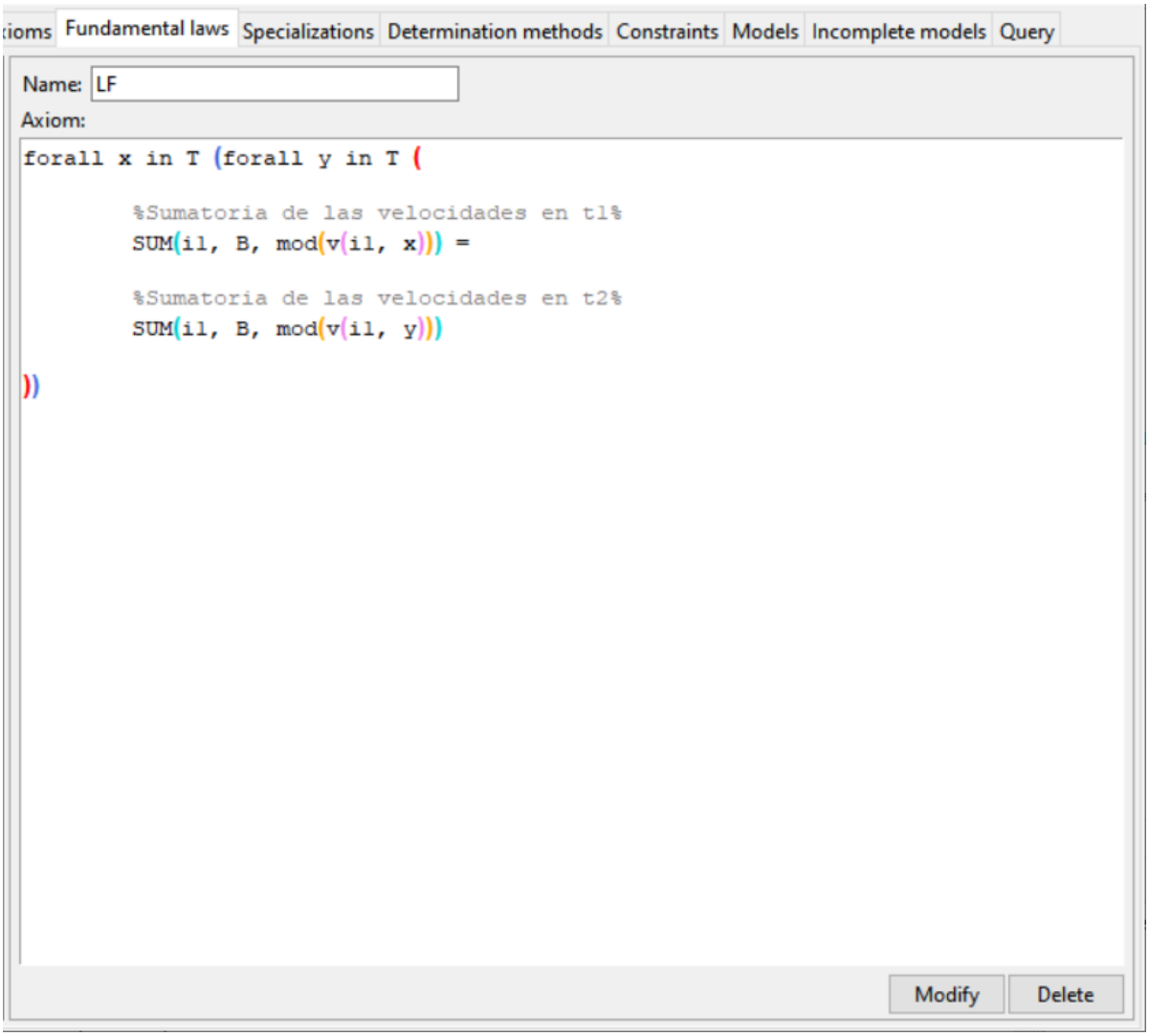


Fig. 4.18. Ley fundamental de **MCAR** cargada en Reconstructor

Puede verse en la imagen que el programa contiene varias funcionalidades para volver más legibles a las fórmulas complejas. En primer lugar, ignora los enters y tabs al introducir axiomas. Además, los paréntesis de apertura y de cierre que se corresponden entre sí se colorean automáticamente igual. En tercer lugar, parar el cursor a la izquierda de un paréntesis de apertura hará que tanto él como su paréntesis de cierre se resalten en gris. Por último, Reconstructor permite ingresar comentarios (entre símbolos "%") que serán ignorados luego.

4.1.6. Especializaciones

La carga de las especializaciones es similar y puede realizarse desde el módulo correspondiente. Existen, sin embargo, algunas ligeras diferencias: al cargar una especialización, debe decirse de qué elemento teórico lo es y con cuántos axiomas cuenta (ya que una misma especialización puede contener más de un axioma; en el caso de **MCAR** ambas contienen solo uno). Indicado esto, el lenguaje formal utilizado es exactamente el mismo. Por ejemplo, la primera especialización de **MCAR** se ve del siguiente modo:

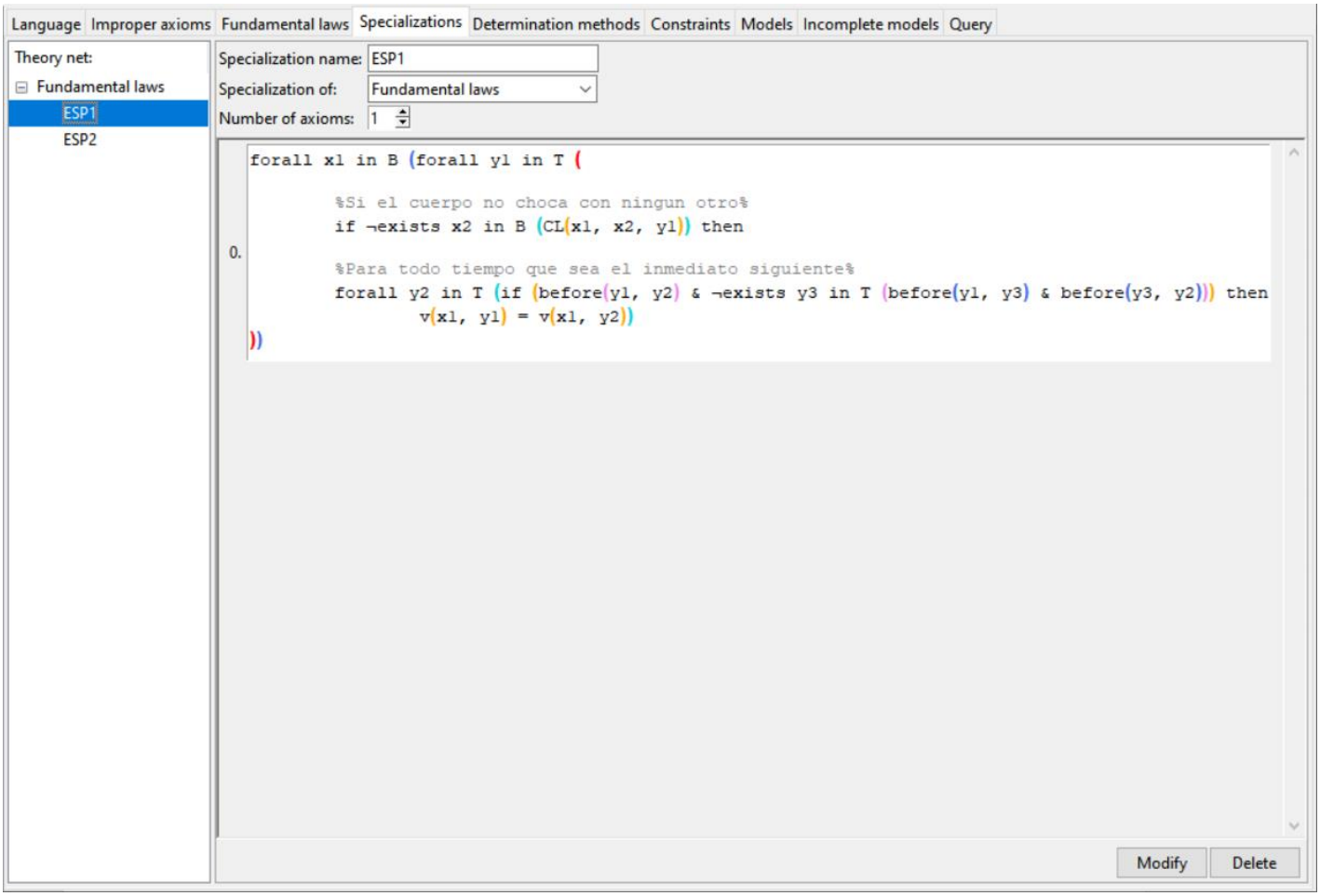


Fig. 4.19. Primera especialización de **MCAR** cargada en Reconstructor

En este caso fue necesario hacer ajustes respecto de la oración formal dada en la sección anterior. Esto se debe a que la oración formal anterior utilizaba una abreviatura que, estrictamente hablando, no era correcta: contenía al término t_{i+1} en el consecuente del condicional. Lo mismo ocurre con la segunda especialización.

Una vez cargadas ambas, es posible pedirle al programa que dibuje la red teórica desde el menú **Visualization > Draw theory net**, la cual, en el presente caso sencillo, se ve del como en la figura 4.6a. Redes teóricas más complejas tendrán visualizaciones más interesantes (véase 4.6b para un ejemplo).

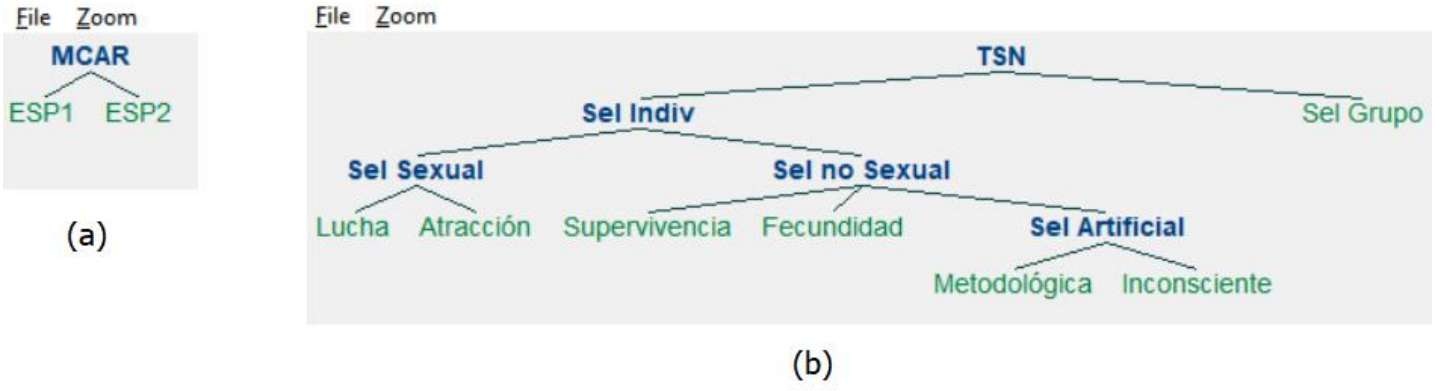


Fig. 4.20. (a) Red teórica de **MCAR** en Reconstructor; **(b)** Red teórica de la teoría de la selección natural (Ginnobili, 2018, p. 95 simplificada)

4.2. Modelos y el primer método de contrastación

En esta sección se muestra cómo cargar modelos al programa. Los modelos pueden ser de dos tipos, completos e incompletos (i.e. puede fallar la denotación de algún término en ellos). Se expone cómo consultar al programa por la denotación de un término en un modelo, por si una oración dada es satisfecha por un modelo y por si un modelo satisface todos los axiomas propios, leyes fundamentales y especializaciones cargadas anteriormente.

En base a esto, se introduce el primer procedimiento de testeo de una reconstrucción formal: los modelos correspondientes a aplicaciones paradigmáticamente exitosas de la teoría deben satisfacer la formalización de las leyes, mientras que modelos correspondientes a aplicaciones no exitosas deben hacer falsas a alguna ley (a aquella/s que informalmente se piensa que fallan en tal aplicación). Se ejemplifica con algunos casos sencillos.

4.2.1 Carga de modelos

Los modelos (potenciales) completos pueden cargarse desde la pestaña **Models**. Luego de elegido un nombre para el modelo en el campo **Model name**, debe introducirse en el recuadro abajo la interpretación para cada término del lenguaje primitivo, en el campo correspondiente. Como ejemplo, tómese la aplicación de **MCAR** que contiene dos cuerpos (b_1 y b_2) y dos tiempos (t_1 y t_2), tales que b_1 y b_2 chocan antes de t_1, con velocidades de 2 y −2, y resultan en las velocidades −2 y 2 (un caso de aplicación potencial exitosa). En Reconstructor, este modelo se vería como sigue:

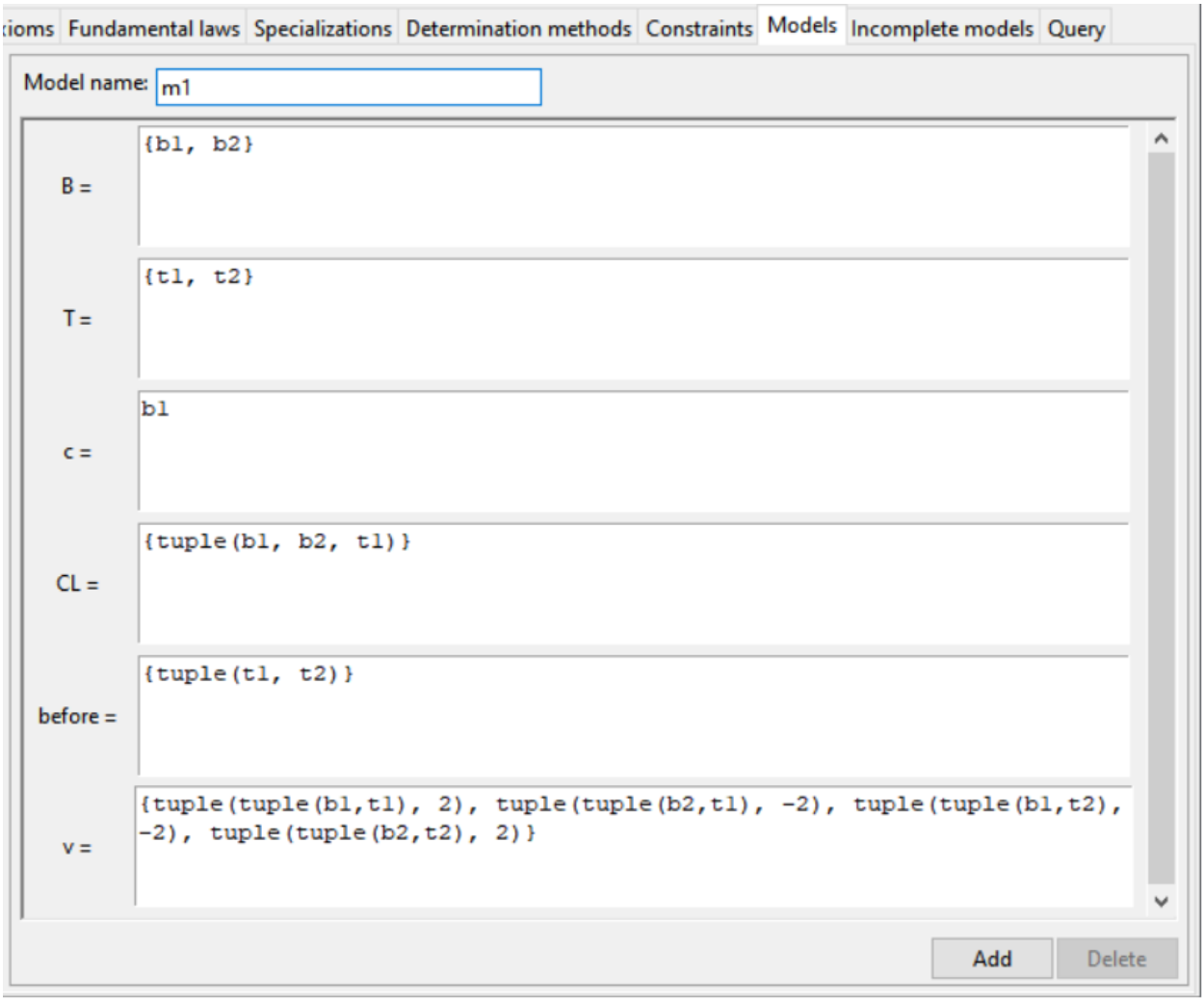


Fig. 4.21. Modelo de **MCAR** cargado en Reconstructor

La sintaxis para la carga de modelos opera como sigue:

- Los dominios deben ser conjuntos de objetos. En Reconstructor (en todos los módulos) es posible usar tanto la sintaxis "{c1, c2, …}" como "set(c1, c2, …)" para conjuntos. Los objetos miembros de los conjuntos pueden ser ellos mismos conjuntos, tuplas (especificadas con "tuple(…)"), números o strings (e.g. "d1"). Las últimas son interpretadas como tipos de objeto propios de la teoría y no como objetos de alguna clase por default.
- Las constantes deben ser objetos de alguno de los dominios, o conjuntos/tuplas construidas a partir de objetos de los dominios.
- Las interpretaciones de relaciones *n*-arias son conjuntos de *n*-tuplas, como es estándar. Cada *n*-tupla debe contener como elementos a objetos de los dominios, propios o auxiliares/por default (o conjuntos/tuplas construidos a partir de los dominios).
- Las funciones pueden especificarse de dos modos:
    - El que se muestra en la imagen, que coincide con un modo estándar de interpretar una función. Es decir, una interpretación de un símbolo de función es un conjunto de pares ordenados. Cada par tiene como primer elemento a una tupla (con los argumentos) y como segundo a un objeto (con el valor de la función para esos argumentos).
    - Un modo menos engorroso de cargar interpretaciones de funciones es usar la sintaxis "$f(a_1, \ldots, a_n) = v_1, f(a_j, \ldots, a_k) = v_2$", en donde f es el término para la función, los a_i son argumentos, y los v_i valores. Nótese que si se introduce la interpretación de la función de este modo, Reconstructor la guardará en memoria del primer modo. Así, si se actualiza la pestaña, la interpretación de la función se verá como arriba.

Como puede verse, desde la pestaña **Models**, todas las interpretaciones que son conjuntos se cargan por extensión, listando a los elementos del conjunto. Esto puede ser engorroso cuando los conjuntos son grandes. Reconstructor permite simular una definición por comprensión desde el módulo de métodos de determinación (véase la sección 4.5).

Una vez cargado un modelo es posible consultar a Reconstructor por la denotación de un término en él, desde el módulo **Query**. Allí, debe seleccionarse la opción **Model** en la primer lista desplegable, escribirse el nombre del modelo a la derecha, seleccionar **Denotation?** en la lista inferior izquierda y brindarse el término a evaluar a su derecha. Esta última lista desplegable trae precargados por default los términos primitivos de la teoría, pero puede escribirse en ella para

consultar por otros términos (primitivos, por default, o términos complejos que tienen ambos). Por ejemplo:

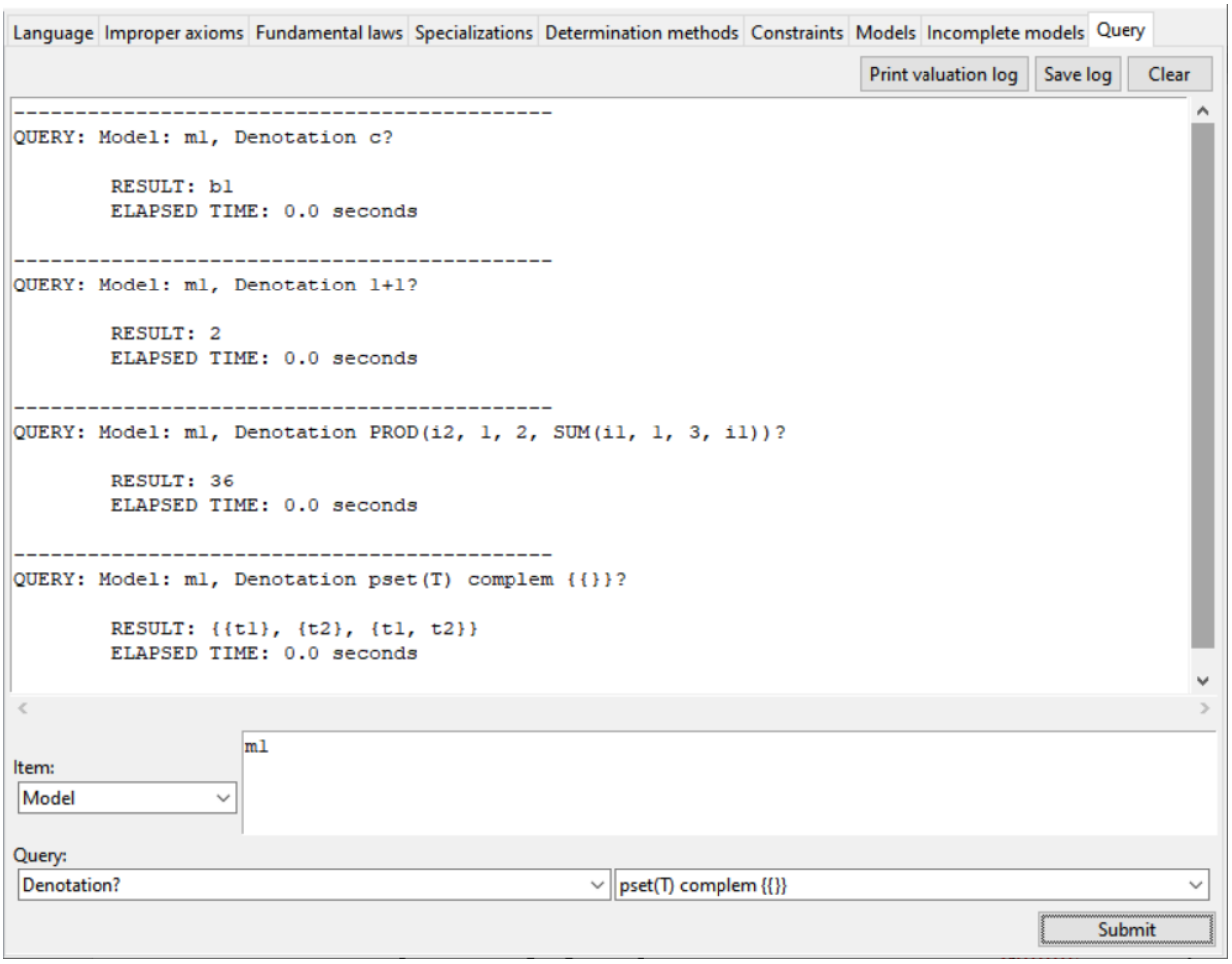


Fig. 4.22. Consultas por la denotación de términos.

4.2.2. Modelos incompletos

Otra funcionalidad que será útil luego es la posibilidad de cargar modelos incompletos. Los modelos incompletos son aquellos en donde uno o más términos contienen fallas de denotación. Esto es, una constante de individuo puede fallar en denotar, una *n*-tupla de elementos puede no pertenecer ni a la extensión ni a la anti-extensión de una relación y un argumento de una función puede no tener asignado un valor.

Este tipo de modelos pueden cargarse desde la pestaña **Incomplete models**.[33] Un ejemplo de un modelo de este tipo es el siguiente:

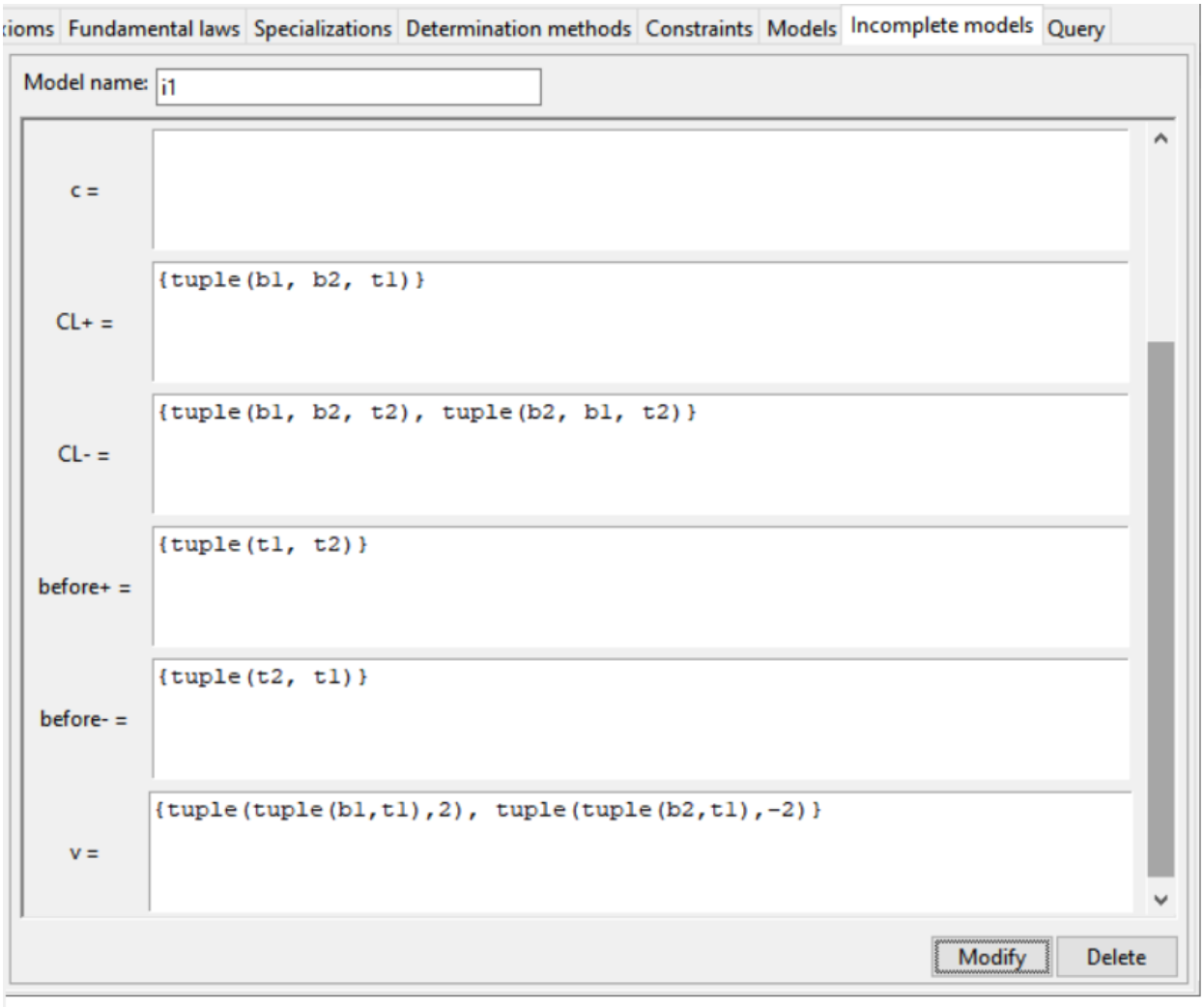


Fig. 4.23. Ejemplo de carga de modelo incompleto en **MCAR**. *B* y *T* (que no se ven en la captura) son iguales que en la figura 4.7.

En este caso, se ilustran los tres tipos de indeterminación: la constante *c* no recibió ninguna interpretación, la tupla $\langle b_2, b_1, t_1 \rangle$ (entre otras) no pertenece ni a la extensión ni a la anti-extensión de *CL* y las velocidades no están cargadas para t_2. Si se consulta a Reconstructor por la denotación de expresiones como "*c*" el resultado será indeterminado [*indeterminate*]. El modo en el que el

[33] De hecho, la pestaña **Models** también acepta el primer y el tercer tipo de indeterminación.

programa valúa las oraciones que contienen términos con fallas de denotación será explicado en la sección 4.4.

4.2.3 Valuaciones y el primer método de contrastación de reconstrucciones

Una de las funciones centrales de Reconstructor es, como se adelantó anteriormente, chequear si una oración formal es satisfecha por un modelo dado. Una vez cargado un modelo, puede chequearse si este satisface una oración desde el módulo **Query**. Existen dos modos de hacer esto: (i) elegir **Sentence** en la primer lista desplegable, escribir la oración, luego elegir **Satisfied by?** y por último dar el nombre del modelo, en la última lista; y (ii) elegir **Model** en la primer lista desplegable, escribir el nombre del modelo, y luego elegir **Satisfies?** y el tipo de axioma (precargado) que se desea evaluar. En la figura 4.10 se ilustra el segundo método.

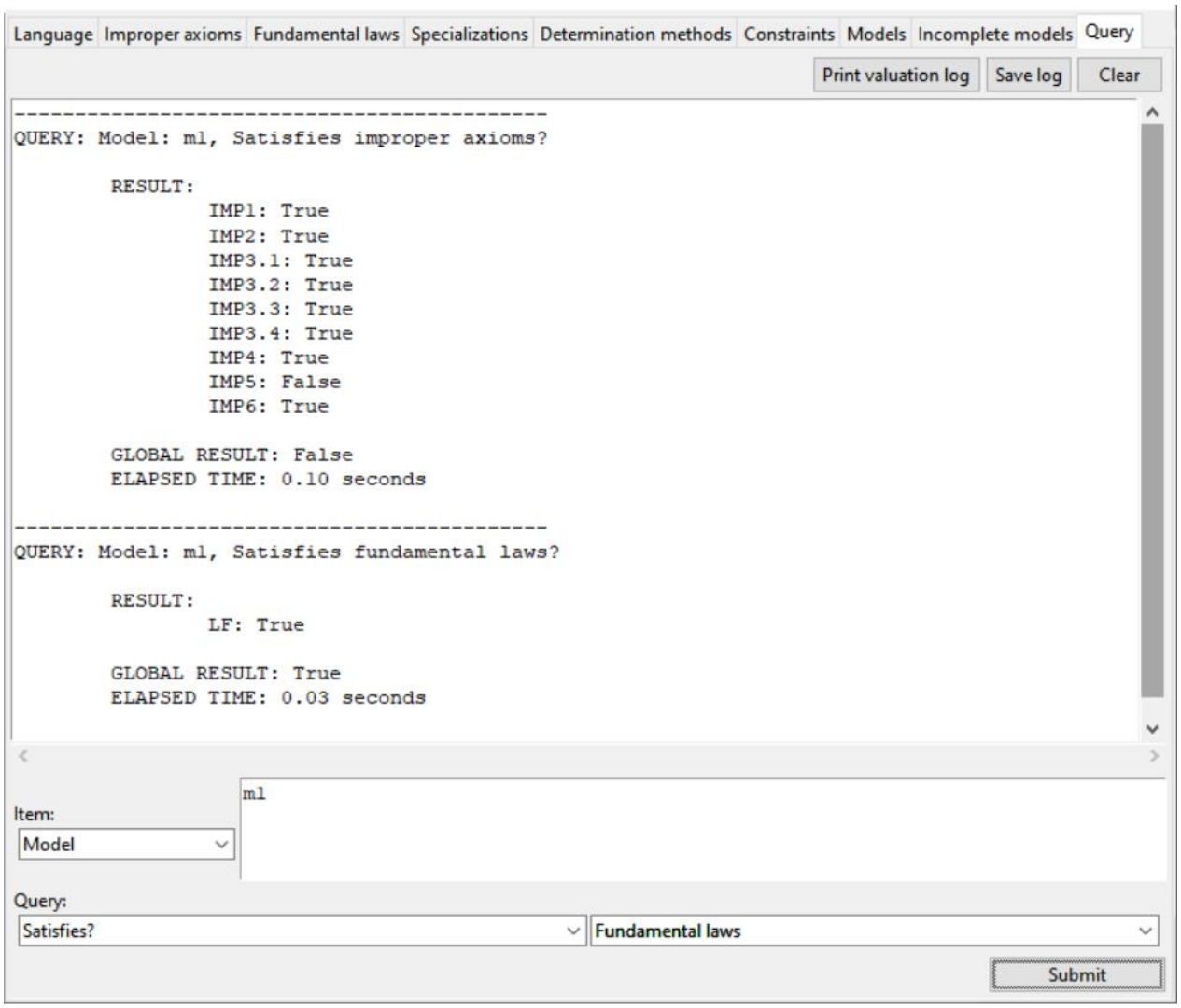


Fig. 4.24. Consultas sobre la valuación de los axiomas impropios y la ley fundamental en m_1. La valuación de las especializaciones puede consultarse de modo idéntico, seleccionando **Specializations** en la última lista desplegable.

Como se dijo anteriormente, el modelo m_1 pretendía representar a un caso de aplicación exitoso de la teoría. Sin embargo, se observa que Reconstructor afirma que uno de los axiomas impropios (IMP5) es falso. Como se adelantó, esto mostraría que la reconstrucción no es adecuada ya que (y en esto consiste el test) modelos que traducen aplicaciones paradigmáticamente exitosas de teorías deben satisfacer las leyes (y viceversa para aplicaciones no exitosas).

Pero Reconstructor no se detiene ahí. Es posible investigar el motivo por el que la contrastación falló. Cuando el valor dado por el programa y las expectativas del usuario no coinciden puede afirmarse que ocurrió una de las tres siguientes cosas:

i. Alguno de los axiomas (en este caso, el axioma impropio 5) fue incorrectamente formalizado.
ii. El modelo no es una traducción adecuada de la situación empírica descripta.
iii. Lo que la comunidad científica considera un modelo exitoso en realidad no lo es.

Para estos casos, en donde las expectativas y el *output* no coinciden, Reconstructor incluye otra herramienta útil para determinar cuál de esas tres opciones es el caso, que consiste en la posibilidad de imprimir el log de valuaciones (i.e. el proceso de razonamiento interno por el que Reconstructor llegó a afirmar que las valuaciones eran esas). Esto se hace oprimiendo el botón **Print valuation log**. Esto abrirá un cuadro de diálogo para guardar el log como un archivo .txt que contiene el log de la última valuación (o conjunto de valuaciones) que se le pidió evaluar. En este caso, para el axioma IMP5, el log dice lo siguiente:

```
109 ------------------------------------------------
110 VALUATION: ['subset(CL,cartp(B,B,T))']
111
112 v(['subset(CL,cartp(B,B,T))']) = 1.0
113
114 ------------------------------------------------
115 VALUATION: ['forall', 'x1', 'in', 'B', ['forall', 'x2', 'in', 'B', ['forall', 'y', 'in', 'T', [['CL(x1,x2,y)'],
    'then', ['CL(x2,x1,y)']]]]]
116
117 v(['CL(c,c,$0)']) = 0.0
118 v([['CL(c,c,$0)'], 'then', ['CL(c,c,$0)']]) = 1.0
119 v(['CL(c,c,$1)']) = 0.0
120 v([['CL(c,c,$1)'], 'then', ['CL(c,c,$1)']]) = 1.0
121 v(['forall', 'y', 'in', 'T', [['CL(c,c,y)'], 'then', ['CL(c,c,y)']]]) = 1.0
122 v(['CL(c,$2,$0)']) = 1.0
123 v(['CL($2,c,$0)']) = 0.0
124 v([['CL(c,$2,$0)'], 'then', ['CL($2,c,$0)']]) = 0.0
125 v(['forall', 'y', 'in', 'T', [['CL(c,$2,y)'], 'then', ['CL($2,c,y)']]]) = 0.0
126 v(['forall', 'x2', 'in', 'B', ['forall', 'y', 'in', 'T', [['CL(c,x2,y)'], 'then', ['CL(x2,c,y)']]]]) = 0.0
127 v(['forall', 'x1', 'in', 'B', ['forall', 'x2', 'in', 'B', ['forall', 'y', 'in', 'T', [['CL(x1,x2,y)'], 'then',
    ['CL(x2,x1,y)']]]]]) = 0.0
128
129 ------------------------------------------------
```

Fig. 4.25. Log de valuaciones de Reconstructor. Las fórmulas se muestran aquí en el formato interno que el programa usa para almacenarlas y evaluarlas.

Dado que Reconstructor utiliza una semántica sustitucional (por limitaciones computacionales, ya que los modelos son siempre finitos) puede verse que asignó un nombre a aquellos elementos de los dominios que no tenían una constante asignada ($0 es t_1, $1 es t_2 y $2 es b_2, estas denotaciones pueden consultarse desde el módulo **Query** si se desea).

Recuérdese que el axioma 5 afirmaba que si b_x choca con b_y en t_i, entonces b_y choca con b_x en t_i (i.e. la relación de choque es simétrica en las primeras dos posiciones). El programa dictaminó que la oración es falsa porque:

- $CL(b_1, b_2, t1)$ es verdadera y $CL(b_2, b_1, t1)$ es falsa (líneas 122 y 123 del log)
- Por tanto, $CL(b_1, b_2, t1) \rightarrow CL(b_2, b_1, t1)$ es falsa (línea 124)
- Por tanto, la cuantificación universal sobre las tres posiciones es falsa (líneas 125 a 127)

Es decir, lo que falló es que $\langle b_2, b_1, t1 \rangle$ no está en la interpretación de *CL*, estando en el caso (ii) anterior (la situación empírica fue incorrectamente cargada como modelo). Si se agrega esta tupla a la denotación de *CL* (en el archivo suplementario, ello se hace en un modelo m_2) se obtiene que todos los axiomas impropios son verdaderos y que el test es exitoso.

Todo esto ilustra otro punto importante. Como el holismo de la contrastación nos enseña, un test de una hipótesis nunca lo es de la hipótesis aislada, sino que siempre existen algunas presuposiciones que se testean conjuntamente con la hipótesis deseada. Estas incluyen condiciones iniciales, cláusulas *ceteris paribus*, etc.; incluso, según las versiones más radicales, un test puede incluir partes de la matemática y de la lógica como parte del conocimiento de background asumido. Si bien el objetivo de este test es determinar si las leyes fueron correctamente formalizadas (i.e. la hipótesis sería que las leyes fueron correctamente formalizadas), como todo otro test de toda otra teoría, también incluye ciertas presuposiciones. En el presente caso, (ii) y (iii) de arriba son algunas de ellas. De ese modo, al igual que en todo otro ámbito científico, el fallo del test no implica automáticamente (i.e. lógicamente) que la hipótesis sea falsa. Nuevamente, el caso recién considerado ilustra este punto.

Un test más completo de una teoría utilizando este método consistiría en diseñar distintos tipos de situaciones (i.e. modelos) en las que leyes deberían ser satisfechas, así como varios en los que distintas combinaciones de leyes deberían no valer. En el segundo caso, el programa debería arrojar que las leyes particulares que se cree que no valen en tal caso son las que son falsas. En **MCAR** esto incluiría diseñar casos que satisfagan (y no satisfagan) la otra ley especial (la que lidia con cuerpos que no chocan), casos con más dos cuerpos y más de dos instantes de tiempo, etc. Si en todos estos casos la reconstrucción "se comporta igual" que su contraparte informal, entonces

puede afirmarse que la formalización de las leyes es correcta (asumiendo que los otros presupuestos se satisfacen también).

4.3. Condiciones de ligadura

Como se expuso en el capítulo 3, las condiciones de ligadura son constricciones que actúan no sobre cada modelo individualmente, sino sobre la aceptabilidad conjunta de varios modelos. En otras palabras, no es un modelo aislado el que satisfará o no una condición de ligadura, sino que debe evaluarse esto en un conjunto de modelos. Por otro lado, las condiciones de ligadura se formulan en un lenguaje formal, pero desde un metanivel, e incluyen a veces términos que no son términos del vocabulario formal de la teoría. En esta sección, se muestra como todas estas características se encuentran incorporadas en Reconstructor. Para ello, se introducen dos condiciones de ligadura a la reconstrucción formal siendo usada de ejemplo (**MCAR**).

4.3.1. Condición de igualdad de la velocidad

Una primera condición de ligadura que se introducirá en **MCAR** afirma que si un mismo cuerpo está presente en dos aplicaciones (modelos) distintos de la teoría, entonces en un mismo tiempo, el cuerpo debe poseer la misma velocidad. En una reconstrucción estructuralista estándar, esto podría formularse del siguiente modo:

> **C_1(MCAR)**: $X \in \mathbf{C_1}$ si y solo si $\emptyset \neq X \subseteq M_p$(MCAR) y para todo $x, y \in X$,
> todo b y todo t: si $b \in B_x \cap B_y$ y $t \in T_x \cap T_y$ entonces, $v_x(b, t) = v_y(b, t)$

Las condiciones de ligadura pueden ser ingresadas a Reconstructor desde la pestaña **Constraints**. En esta pestaña, la parte del medio actúa de modo similar las pestañas **Improper axioms** y **Fundamental laws**. Es decir, se elige un nombre para la condición y se la carga en el campo inferior derecho. El software incluye otros dos términos por default (que solo pueden ser usados

desde este módulo, ya que pertenecen al metalenguaje), con el objetivo de poder cargar condiciones de ligadura. Ellos son:

- "modelset": es una constante de individuo, que referirá al conjunto de los modelos en los que se está evaluando el constraint (la variable *X* arriba).
- "denotation": función diádica, "denotation(x, y)" refiere a la denotación del término *x* en el modelo *y*.

Con ayuda de estos dos términos por default la condición **C₁** puede cargarse del siguiente modo:

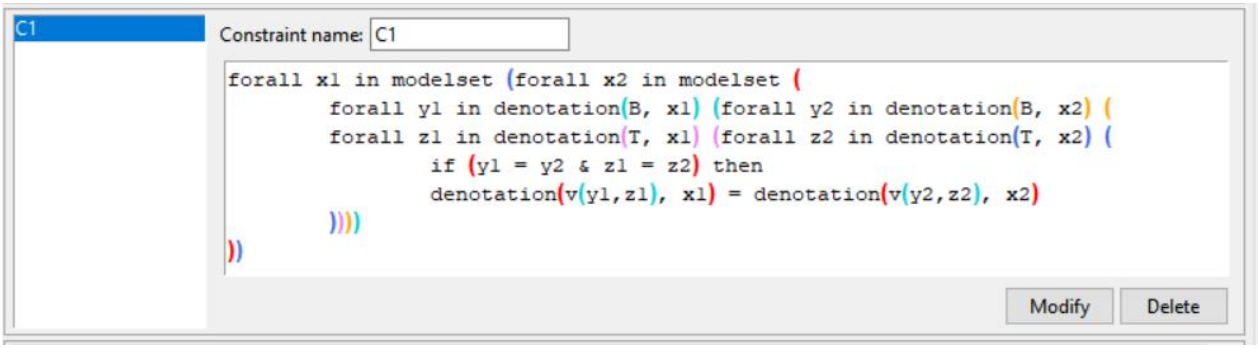


Fig. 4.26. Carga de la condición de ligadura **C₁** en Reconstructor

Como se observa, es necesario hacer alguna traducción entre el lenguaje formal y el lenguaje de Reconstructor (p.e. porque *b* y *t* están cuantificadas sin cota en la oración original). Sin embargo, la sintaxis de Reconstructor es al menos tan transparente y legible como la original.

Para evaluar si una condición de ligadura es satisfecha por un conjunto de modelos, se debe ir al módulo **Query**. Para ilustrar esta funcionalidad agregaré dos modelos, m_3 y m_4, el primero de los cuales es compatible con m_2 (introducido anteriormente) y el segundo no lo es (contiene una velocidad distinta para b_1 y b_2 en t_2) —aunque ambos satisfacen las leyes individualmente, como puede observarse en la primera consulta de la figura 4.13.[34]

[34] La opción **Theory** en la tercer lista desplegable, para la opción **Model set**, testea si el conjunto de modelos dado satisface todos los axiomas (impropios, leyes, especializaciones y condiciones de ligadura) de la teoría.

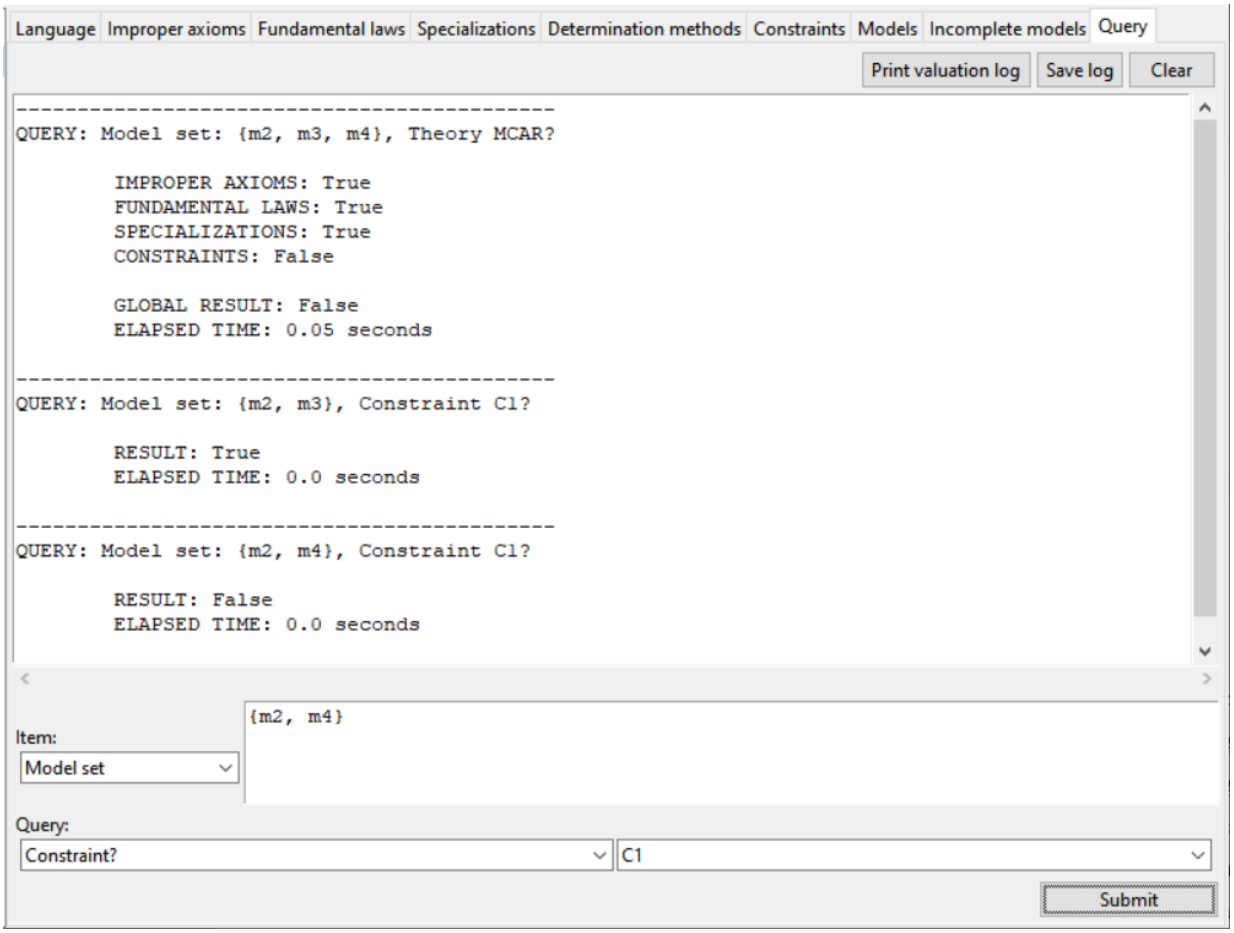


Fig. 4.27. Consultas sobre la satisfacción de condiciones de ligadura por un conjunto de modelos.

Al igual que en la sección 4.2.3., un modo de contrastar que una formalización de una condición de ligadura sea adecuada es comparando los resultados que pre-formalmente se esperarían con el *output* del programa, para consultas sobre la verdad o falsedad de la condición en un conjunto de modelos.

4.3.2. Condición de extensividad del volumen

Para ilustrar una funcionalidad adicional (la posibilidad de agregar lenguaje metateórico) será conveniente agregar un concepto más al lenguaje objeto de la teoría —lo haré a través de la función unaria *q*, que representa al volumen de un cuerpo (puede agregarse a Reconstructor desde la pestaña **Language**, como se ilustró anteriormente). La condición de ligadura nueva afirmará que si un

cuerpo b_3 en un modelo y es el resultado de concatenar dos cuerpos b_1 y b_2 de otro modelo x, entonces el volumen de b_3 es igual a la suma de los volúmenes de b_1 y b_2 (este es otro tipo de condición de ligadura típico de las teorías; p.e. existe uno similar para la masa en la mecánica newtoniana del choque, véase Balzer et al., [1987] 2012, p. 153). En formato estructuralista, esto se expresaría como sigue:

C_2 es una condición de ligadura de extensividad del volumen en MCAR si y solo si existe • tal que:

- $\bullet \subseteq (\mathbf{B} \times \mathbf{B} \times \mathbf{B})$, con $\mathbf{B} = \{B_x / x \in M_p(\text{MCAR})\}$
- Para toda X, $X \in \mathbf{C_2}$ si solo si $\emptyset \neq X \subseteq M_p(\text{MCAR})$ y para todo $x, y \in X$, todo b, b' y b'':

  si $b, b' \in B_x$ y $b'' \in B_y$ y $\bullet(b, b', b'')$, entonces, $q_y(b'') = q_x(b) + q_x(b')$

Esta condición de ligadura presupone la existencia de un término • del metalenguaje, tal que $\bullet(b, b', b'')$ representa a la afirmación de que b'' es una concatenación de b y b'. Asimismo, evaluar si esta condición se satisface en un conjunto de modelos presupone contar con una interpretación para este término metateórico. Ambas cosas pueden cargarse desde la pestaña **Constraints**, en la parte superior (que funciona de modo similar a la pestaña **Language**) e inferior (que funciona de manera similar a la pestaña **Models**).

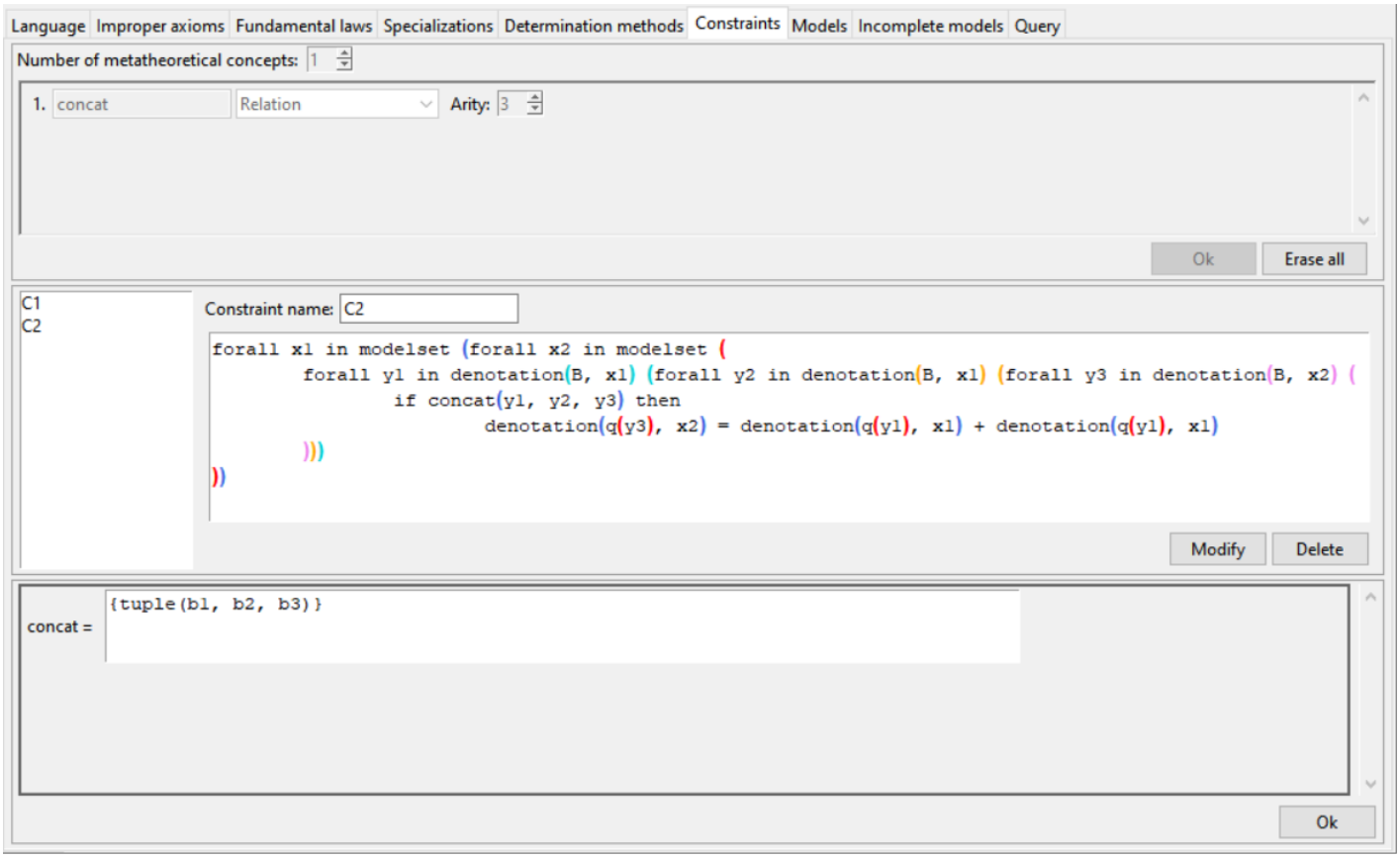


Fig. 4.28. Carga de lenguaje metateórico y de una interpretación para este lenguaje, desde la pestaña **Constraints**. El término metateórico • fue cargado como "concat", para que sea más sencillo de escribir con un teclado QUERTY. La parte del medio de la imagen contiene la formalización de $\mathbf{C_2}$.

Una vez cargada esta condición de ligadura, puede testearse la corrección de la formalización (expuesta en el campo del medio en la figura 4.14) insertando dos nuevos modelos, m_5 y m_6, que contengan un cuerpo b_3. Para ilustrar, se da a b_1 y b_2 un volumen de 1 en m_2 y a b_3 un volumen de 2 y 3 en m_5 y m_6, respectivamente.

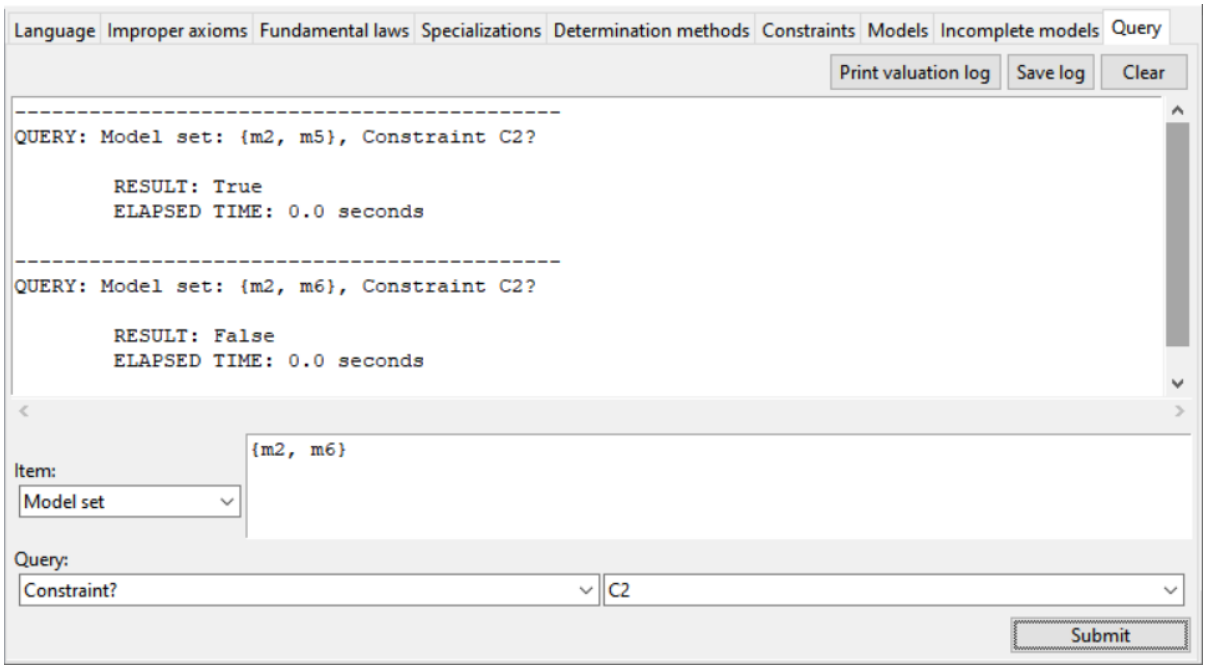


Fig. 4.29. Contrastación de la adecuación de la formalización de $\mathbf{C_2}$.

4.4. La semántica paracompleta de Reconstructor

Como se expuso en la sección 4.2.2., Reconstructor permite cargar modelos incompletos, que contienen fallas de denotación. Para evaluar oraciones en un modelo de este tipo, el programa emplea una semántica no-clásica, trivaluada y paracompleta.

En cuanto al uso de lógicas no clásicas en el marco del estructuralismo metateórico no existe literatura previa disponible. Sí bien hay usos previos de lógicas trivaluadas paracompletas para representar casos de falta de información (véanse Cobreros, Égré, Ripley, & van Rooij, 2014; Gottwald, 2017; Kleene, 1952; Łukasiewicz, 1920 —republicado en inglés en Łukasiewicz, 1970—; Nolt, 2018; Priest, 2008), ellas no fueron aplicadas detalladamente a contextos científicos, y menos aún en el marco de una metateoría formal. Hacer esto tiene pleno sentido ya que existen casos en la práctica científica en las que un investigador no conoce de antemano las denotaciones de todos los conceptos en una aplicación (modelo) dada, y que pueden ser fructíferamente representados como casos de fallas de denotación. Esto ocurre, por ejemplo, previamente a hacer una predicción o a realizar cierta observación.

En esta sección se describe este aparato semántico en detalle. Esto no solo agregará otro uso interesante para las lógicas trivaluadas paracompletas (véase Cobreros, Égré, Ripley, & van Rooij, 2014 para un resumen de los usos actuales) y un uso novedoso de lógicas no clásicas en metateorías formales, sino que además me permitirá introducir un nuevo modo de contrastar una reconstrucción.

4.4.1. Atómicas y conectivas

Para explicar el modo en el que Reconstructor se comporta con modelos incompletos, cargaré, en primer lugar el siguiente modelo incompleto (bajo el nombre i_2 en el archivo suplementario).

$i_2 = \langle B_{i2}, T_{i2}, c_{i2}, before_{i2}, CL_{i2}, q_{i2}, v_{i2}\rangle$
$B_{i2} = \{b_1, b_2\}$
$T_{i2} = \{t_1, t_2\}$
c_{i2} = [falla de denotación]
$before_{i2}^{+} = \{\langle t_1, t_2\rangle\}$
$before_{i2}^{-} = \{\langle t_1, t_1\rangle, \langle t_2, t_1\rangle, \langle t_2, t_2\rangle\}$
$CL_{i2}^{+} = \{\langle b_1, b_2, t_1\rangle, \langle b_2, b_1, t_1\rangle\}$
$CL_{i2}^{-} = \{\langle b_1, b_1, t_1\rangle, \langle b_2, b_2, t_1\rangle\}$
$q_{i2} = \{\langle\langle b_1\rangle, 1\rangle, \langle\langle b_2\rangle, 1\rangle\}$
$v_{i2} = \{\langle\langle b_1, t_1\rangle, 5\rangle, \langle\langle b_2, t_1\rangle, -5\rangle\}$

Con este modelo, es posible ejemplificar el modo como Reconstructor se comporta en casos de fallas de denotación. Para comenzar, si una atómica contiene un término cuya denotación falla, entonces la atómica recibirá valor 0.5 (indeterminado; véase la figura 4.16). Caso contrario, la cláusula clásica para atómicas es usada.

Para oraciones que contienen una conectiva como su símbolo principal, la situación es un poco más compleja. No se quiere que toda fórmula que contiene una subfórmula con valor 0.5 tenga a su vez valor 0.5. La interpretación intuitiva del valor 0.5 para una oración es que la oración en cuestión tiene un valor de verdad determinado (es decir, es o bien verdadera o bien falsa), pero

que no conocemos por alguna limitación empírica o epistémica. Esto contrasta con interpretaciones del valor 0.5 como la ausencia de valor de verdad (p.e. para oraciones que puedan ser consideradas vacías de significado, como oraciones gramaticalmente mal formadas —la lógica trivaluada de Bochvar (1938) permite lidiar más adecuadamente con ese tipo de casos).

En su lugar, considérese la oración "*CL*(*c*, *c*, *c*) ∨ Verum". En i_2, no conocemos el valor de verdad del disyunto de la izquierda (i.e. "*CL*(*c*, *c*, *c*)" recibe valor 0.5, véase arriba) dado que la constante *c* falla en denotar. Sin embargo, esto no importa a fin de evaluar la oración completa ya que basta con que un disyunto sea verdadero para que una disyunción lo sea, y el segundo disyunto lo es. En otras palabras, o bien "*CL*(*c*, *c*, *c*)" es verdadera o bien es falsa. Pero en ambos casos "*CL*(*c*, *c*, *c*) ∨ Verum" será verdadera, con lo cual nuestro desconocimiento del valor de verdad de "*CL*(*c*, *c*, *c*)" no afecta a nuestro conocimiento del valor de la oración completa. Por otra parte, la oración "*CL*(*c*, *c*, *c*) ∧ Verum" sí debería recibir el valor 0.5, ya que si "*CL*(*c*, *c*, *c*)" fuese verdadera la oración recibiría el valor 1, mientras que si fuese falsa, la oración recibiría el valor 0; por tanto, dado que no se conoce el valor de "*CL*(*c*, *c*, *c*)", no hay modo de saber si la oración completa es verdadera o falsa.

Extender esta línea de razonamiento al resto de las conectivas clásicas da como resultado las siguientes tablas de verdad, que corresponden con las matrices de la lógica K_3 (introducidas por Kleene, 1952):

¬	
1	0
0.5	0.5
0	1

∨	**1**	**0.5**	**0**
1	1	1	1
0.5	1	0.5	0.5
0	1	0.5	0

∧	**1**	**0.5**	**0**
1	1	0.5	0
0.5	0.5	0.5	0
0	0	0	0

→	**1**	**0.5**	**0**
1	1	0.5	0
0.5	1	0.5	0.5
0	1	1	1

Tablas 4.6 a 4.5. Tablas de verdad para las conectivas.

De este modo, Reconstructor otorga los siguientes resultados a consultas desde el modelo **Query**:

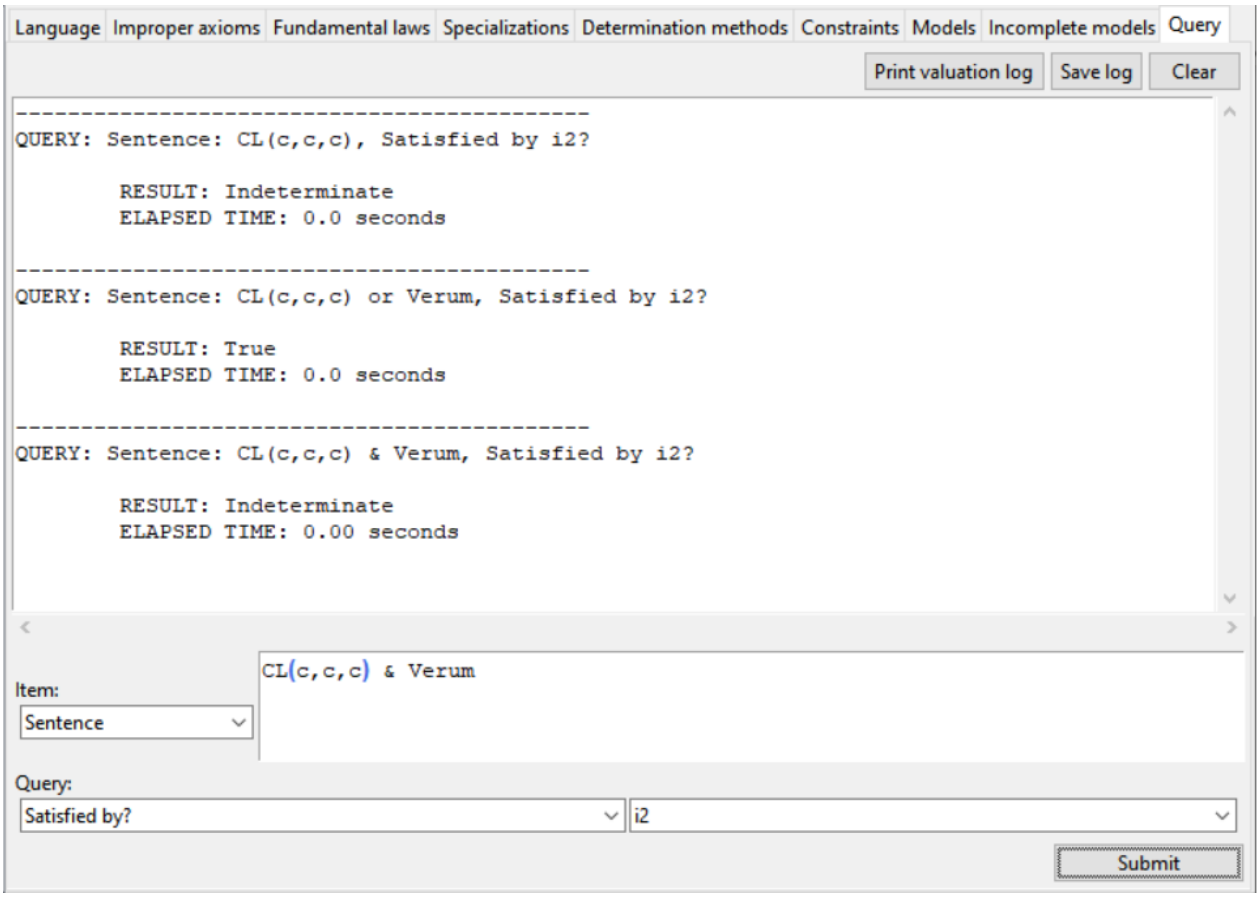


Fig. 4.30. Resultados de consultas de satisfacción de oraciones en modelos incompletos, parte 1.

Por último, nótese que Reconstructor está diseñado para consultar acerca del valor de verdad de una oración en un modelo dado. No incluye una definición de consecuencia lógica (no chequea si una inferencia es válida en el lenguaje objeto). Por tanto, aunque utiliza las *matrices* de K_3, estrictamente hablando, no implementa a la *lógica* K_3. Por supuesto que, dadas estas cláusulas semánticas, el programa *sí infiere* el valor de verdad que una oración tendrá en un modelo. Por ejemplo, sí se le pide dar $v_m(\neg\phi)$ y está dado en m que $v_m(\phi) = 0$ entonces, dado que una de las cláusulas semánticas afirma que "si $v_m(\phi) = 0$ entonces $v_m(\neg\phi) = 1$", Reconstructor inferirá (utilizando un *modus ponens*) que $v_m(\neg\phi) = 1$. La lógica que Reconstructor implementa para hacer este tipo de inferencias metateóricas (semánticas) es la lógica clásica. Esto es usual en enfoques no clásicos, en donde la metateoría suele ser clásica.

4.4.2. Cuantificadores

Los cuantificadores son evaluados de manera estándar en Reconstructor, como la conjunción (para el universal) y disyunción (para el existencial) de todas sus instancias. Sin embargo, es necesario hacer algunas cualificaciones. Recuérdese que el lenguaje de base del programa incluye (entre otras cosas) a la aritmética, y con ello a términos como los numerales, que son siempre interpretados de modo estándar en todo modelo (completo e incompleto). Es decir, todo modelo de Reconstructor es en realidad infinito (ya que contiene a $\mathbb{N}$ como dominio auxiliar), aunque esto no se vea explícitamente desde a pestaña **Models**. Por tanto, por motivos obvios de computabilidad, los cuantificadores deben estar siempre acotados a conjuntos finitos (véase 4.3.1). Esto introduce una complicación adicional, y es que el término de la cota de un cuantificador puede fallar en denotar.

Versiones previas del programa lidiaban con este problema simplemente dando valor 0.5 a una oración cuantificada cuya cota fallaba en denotar. Sin embargo, ello dejaba como indeterminadas a oraciones que deberían tener un valor determinado. Por ejemplo, la oración "$\forall x$ in $\{c\}$ (Verum)" debería ser verdadera en i_2, dado que sin importar lo que c denote, la oración será verdadera de todos modos. Para corregir esto, se utilizó la siguiente estrategia. Cada vez que la denotación de la cota es indeterminada, las variables libres ligadas por el cuantificador son sustituidas por una constante especial "*indeterminum*". Toda atómica que contiene a esta constante recibe valor 0.5. Así, "$\forall x$ in $\{c\}$ (Verum)" recibe valor 1, mientras que "$\forall x$ in $\{c\}$ ($q(x) = 1$)" recibe valor 0.5 (a pesar de que q está determinada para todo elemento de su dominio) —véase la figura 4.17.

Por otra parte, nótese que "$\forall x \in A$ (ϕ)" es una abreviación de "$\forall x$ ($x \in A \rightarrow \phi$)", mientras que "$\exists x \in A$ (ϕ)" lo es de "$\exists x$ ($x \in A \wedge \phi$)". Sin embargo, en caso de cota con falla de denotación el término A puede denotar cualquier cosa, *incluyendo al conjunto vacío*. Pero si A fuese el conjunto vacío, "$\forall x$ ($x \in A \rightarrow \phi$)" sería verdadera independientemente de la valuación de ϕ (ya que el antecedente del condicional sería falso para toda sustitución de x), mientras que "$\exists x$ ($x \in A \wedge \phi$)" sería falsa (ya que el primer conyunto sería siempre falso). Por ello, las fórmulas de la forma "$\forall x \in A$ (ϕ)" en donde la denotación de A falla nunca reciben valor 0 (dado que para algunas denotaciones posibles serían verdaderas), mientras que fórmulas de la forma "$\exists x \in A$ (ϕ)" nunca reciben valor 1 (por motivos análogos). Esto da como resultado el último caso de la figura 4.17:

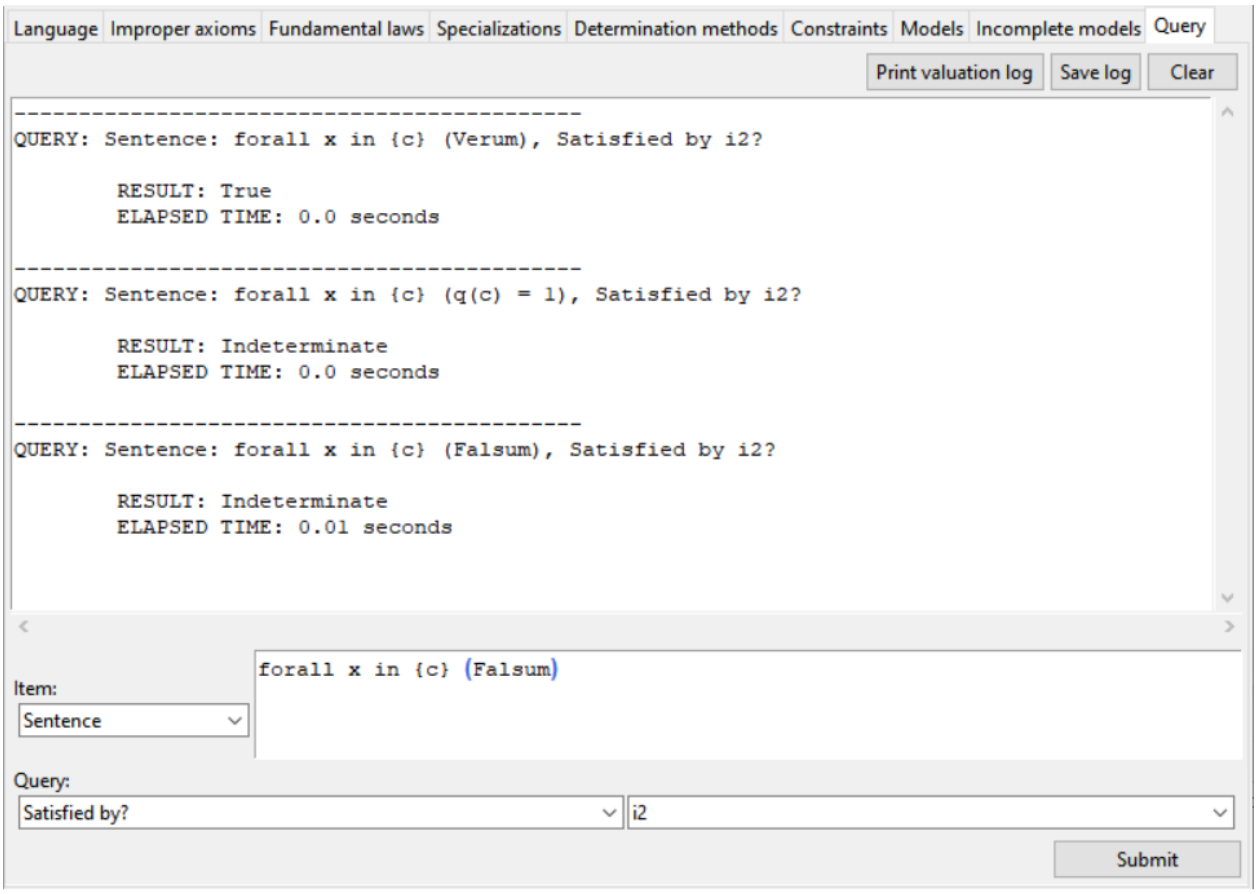


Fig. 4.31. Resultados de consultas de satisfacción de oraciones en modelos incompletos, parte 2.

4.4.3. Recaptura

El aparato semántico introducido hasta este punto, aunque es mayormente adecuado para los propósitos filosóficos para los que Reconstructor fue diseñado, tiene un potencial problema. Este consiste en que algunas oraciones reciben valor indeterminado cuando deberían ser verdaderas (o falsas). Un ejemplo claro de ello es la oración "$q(c) = 1 \vee \neg q(c) = 1$" en i_2, que debería ser verdadera ya que cualquiera sea la denotación de c, la oración será verdadera. Sin embargo, dado que "$q(c) = 1$" es indeterminada, también lo serán su negación y la disyunción entre ambas, de acuerdo a las tablas de K_3. Lo mismo ocurriría con cualquier otra fórmula que sea una tautología clásica que contenga atómicas indeterminadas, así como con oraciones que son teoremas de la teoría, y que por tanto deberían ser verdaderas en todo modelo suyo, aunque algunos términos fallen en denotar.

Hay dos modos posibles de lidiar con este problema, el primero basado en lógicas libres y el segundo en métodos de recaptura. Se exponen en ese orden a continuación, y se explica por qué se prefirió el segundo método al primero en la implementación de Reconstructor.

El primer enfoque posible, basado en lógicas libres y supervaluaciones (véase Nolt, 2018), consistiría en lo siguiente. Para evaluar una oración con un término *b* que falla en denotar, asígnese todo posible objeto del dominio a *b* y véase la valuación de la oración resultante. Si todas las asignaciones son verdaderas, entonces la oración original también lo es. Si todas son falsas, la oración original es falsa. En caso contrario, la oración recibe un valor indeterminado. La razón por la que esto no se implementó en Reconstructor es obvia: de hecho computar una valuación de este modo sería imposible ya que, como se dijo, todo modelo es infinito (*b* podría ser miembro de algún conjunto auxiliar como los números naturales). Por tanto, asignar toda denotación posible a *b* requeriría evaluar la oración con infinitas sustituciones de *b*.

El segundo método se basa en la idea de recaptura (Tajer, 2018; Teijeiro, 2018). La idea aquí será "marcar" con una constante del lenguaje objeto que se está en un contexto clásico (i.e. de valores de verdad no indeterminados para una fórmula). La constante en cuestión será el operador de determinación (presente en lógicas como LFI1, véase Carnielli & Coniglio, 2016), ° en Reconstructor, cuya tabla de verdad es:

°	
1	0
0	0.5
1	1

Tabla 4.6. Tabla de verdad para el operador de determinación.

Lo que esta constante hace es devolver el valor 1 cuando la oración que sigue tiene un valor determinado, y 0 cuando tiene valor indeterminado. Es decir, °ϕ afirma que ϕ tiene un valor determinado. De ese modo, permite afirmar cosas del estilo "si el valor de la oración ϕ es determinado, entonces ψ". Así, si bien "$q(c) = 1 \vee \neg q(c) = 1$" continuará teniendo valor indeterminado, al menos puede expresarse la oración "$^\circ q(c) = 1 \rightarrow (q(c) = 1 \vee \neg q(c) = 1)$", que será verdadera en todo modelo. En el caso de los teoremas de la teoría, para verificar que valen también en modelos incompletos, lo que podría hacerse (aunque sería engorroso) es formular la oración

“°(∧Ax) → T”, en donde ∧Ax es la conjunción de los axiomas de la teoría y T el teorema en cuestión.

4.5. Métodos de determinación manuales

Como se expuso en el capítulo 3, de acuerdo a la metateoría estructuralista, una teoría es más que un conjunto de axiomas. Las teorías también contienen otros tipos de elementos. Uno de esos otros, las condiciones de ligadura, ya fueron tratados en este capítulo. En esta sección se expone como Reconstructor lidia con otro tipo de elemento: los métodos de determinación. Esto lo hace a través del módulo **Determination methods**, uno de los más útiles del software. Los métodos de determinación en Reconstructor pueden ser manuales (i.e. cargados por el usuario, esta sección) o automáticos (i.e. inferidos por el programa a partir de las leyes, sección siguiente). Como se verá a continuación, la carga de métodos manuales presupone algunos conocimientos de programación, no así con los métodos automáticos de la sección siguiente.

4.5.1. Carga de métodos: un método T-teórico

Como se presentó en el capítulo 3, un método de determinación es (preteóricamente) un modo sistemático de hallar la extensión de un concepto. Reconstructor elucida e implementa esta idea a través de la noción de *algoritmo*. Como se vio anteriormente, esta no es la elucidación estándar de dicha noción (la cual apela a clases de modelos). Me extenderé sobre las consecuencias conceptuales de este punto en el capítulo 6. Aquí me centraré en mostrar su implementación en el software.

En el programa, un método de determinación es un algoritmo, esto es, una secuencia ordenada de pasos a seguir, que no requieren creatividad, y que, dado cierto *input*, da como resultado la extensión deseada del concepto. Los algoritmos deben escribirse en un lenguaje de programación, en este caso Python 3. Esto se debe a que es el lenguaje en el que Reconstructor mismo está programado (i.e. la descarga misma del programa incluye a un intérprete de Python). Por otro lado, Python como lenguaje tiene algunas ventajas: tiene una sintaxis relativamente

transparente y es fácil de aprender (comparativamente con otros lenguajes, al menos). La ventaja de cargar a los métodos de determinación como algoritmos es que los algoritmos pueden luego ser ejecutados por una computadora. De ese modo, los métodos de determinación servirán para completar las denotaciones de modelos incompletos.

El modo más sencillo de introducir esta funcionalidad es a través de un ejemplo. Se muestra a continuación cómo cargar un método de MCAR que aplica la especialización ESP2, para determinar las velocidades de dos cuerpos que chocan. Informalmente, el input de este método consiste en los conceptos *B*, *T*, *before*, *CL* y *v* (las interpretaciones de estos términos en un modelo). Es decir, es necesario conocer de antemano cuál es el conjunto de cuerpos, de tiempos, la relación de precedencia temporal, la relación de choque y las velocidades (iniciales). Dado todo esto, el algoritmo operará como sigue:

Inputs: *B*, *T*, *before*, *CL*, *v* [sus interpretaciones en un modelo particular]
1. Para todo instante de tiempo x_1 en T:
    2. Si hay un tiempo x_2 sucesor en *T*:
    3. Para todo cuerpo y_1 en *B*:
        4. Para todo cuerpo y_2 en *B*:
            5. Si $\langle y_1, y_2, x_1\rangle \in CL$:
            6. Obtener las velocidades $v(y_1, x_1)$ y $v(y_2, x_1)$
            7. Si ambas están determinadas:
            8. Agregar $v(y_1, x_2) = -v(y_1, x_1)$ a *v*
            9. Agregar $v(y_2, x_2) = -v(y_2, x_1)$ a *v*
10. Devolver *v*

Las instrucciones en un algoritmo se ejecutarán secuencialmente, una a una. En algunos casos, la ejecución puede saltar desde un paso hacia otro que está más atrás. Por ejemplo, los bloques indentados representan aquí partes de código que pueden ejecutarse más de una vez, por estar en el contexto de un *loop* (en este caso, un cuantificador universal). Por ejemplo, si en el modelo dado $B = \{b_1, b_2\}$, entonces se ejecutarán las instrucciones 4-9 primero con $y_1 = b_1$ y luego con $y_1 = b_2$. En cada una de estas dos ejecuciones, los pasos 5-9 se ejecutarán primero con $y_2 = b_1$ y luego con $y_2 = b_2$ (ya que hay otro cuantificador universal en el paso 4.). El paso 2. está simplificado arriba.

Para que el algoritmo no requiera ningún tipo de creatividad deben darse instrucciones más específicas (véase a continuación).

Para cargar este algoritmo en Reconstructor, se debe ir a la pestaña **Determination methods**. Desde allí, en el sector del medio a la izquierda, se elige primero un nombre para el método y se agregan los conceptos input a la lista. Al oprimir el botón >>, se habilitará el campo de la derecha, y se completarán automáticamente algunas cosas.

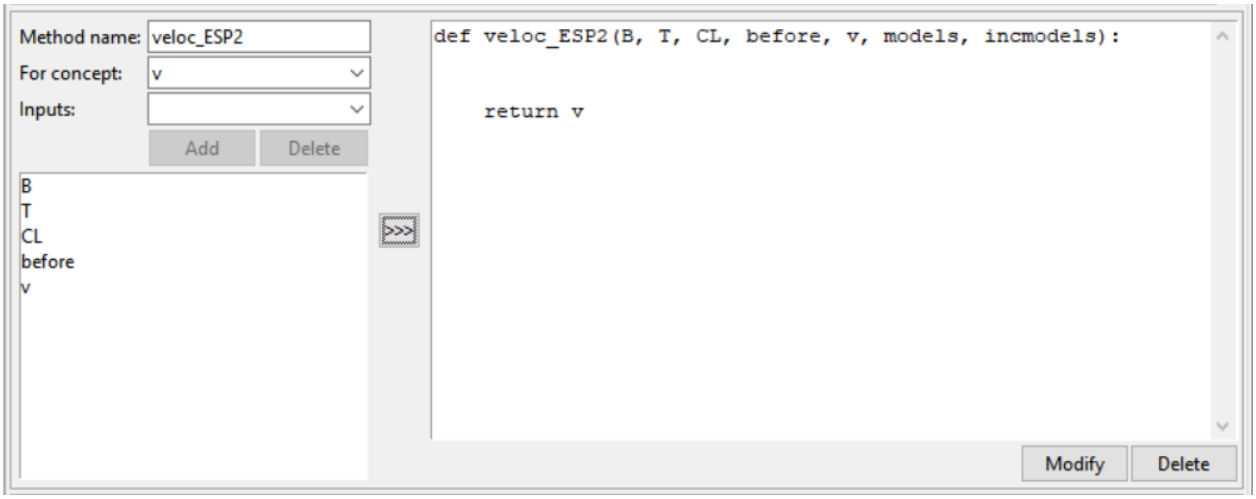


Fig. 4.32. Pasos iniciales para la carga de un método de determinación

Como puede verse, en Reconstructor, los métodos de determinación son funciones de Python. Como argumentos, estas funciones tienen a los conceptos del modelo que se dan como input. Dos constantes especiales, "models" e "incmodels", contienen a todos los modelos completos e incompletos (para la carga de métodos que usan alguna condición de ligadura, véase la subsección 4.5.4.). El valor a devolver debe ser el concepto para el cual el método está diseñado. Teniendo todo esto en cuenta, el algoritmo anterior se traduce en el lenguaje de Python a lo siguiente:

```
def veloc_ESP2(B, T, before, CL, v, models, incmodels):
    v = set(v)
    for x1 in T:
        # Para cada tiempo x1, chequear si hay un tiempo x2 sucesor
        x2 = None
        for z in T:
            if (x1, z) in before:
```

```
                instante_siguiente = True
                for x3 in T:
                    if (x1, x3) in before and (x3, z) in before:
                        instante_siguiente = False
                if instante_siguiente:
                    x2 = z
                    break
        # Si hay:
        if x2 is not None:
            # Para todo par de cuerpos
            for y1 in B:
                for y2 in B:
                    # Si chocan:
                    if (y1, y2, x1) in CL:
                        # Obtener las velocidades iniciales
                        v_y1 = None
                        v_y2 = None
                        for v_argument in v:
                            if v_argument[0] == (y1, x1):
                                v_y1 = v_argument[1]
                            if v_argument[0] == (y2, x1):
                                v_y2 = v_argument[1]

                        # Si ambas están determinadas, agregar las velocidades finales
                        if v_y1 is not None and v_y2 is not None:
                            v.add(((y1, x2), -v_y1))
                            v.add(((y2, x2), -v_y2))
    return v
```

Para el lector conocedor de Python, hay algunos puntos a destacar en el código de arriba (en caso de desconocimiento, el lector puede saltar directamente a la subsección siguiente). Para comenzar, los modelos en Reconstructor son diccionarios de Python, en donde las *keys* son los términos y los *values* sus interpretaciones. Las interpretaciones (los *values*) siguen el mismo formato que en la pestaña **Models**: las interpretaciones de constantes son objetos (números, conjuntos, o strings para los objetos propios de la teoría como b_1), las de predicados monádicos son conjuntos de objetos,

las de predicados *n*-ádicos conjuntos de *n*-tuplas de objetos y las de funciones *n*-ádicas pares ordenados, en donde el primer elemento es una *n*-tupla.

Las interpretaciones de los predicados y funciones (así como cualquier conjunto dentro de una interpretación) son *frozensets*, no *sets*; del mismo modo, todas las tuplas de elementos son objetos de la clase *tuple*, no *list*. De ese modo, la primera línea del código transforma *v* a *set*, ya que se quiere agregar elementos a *v* (las dos líneas antes de la última). Si el objeto del *return statement* es un objeto de la clase *set* (como en este caso) el programa lo convertirá automáticamente en un *frozenset* (véase la figura 4.20 abajo).

En segundo lugar, los términos que fallan en denotar tendrán asignada la constante *None* como *value*. De ahí que el chequeo de si las velocidades iniciales están determinadas se realice como en la cuarta línea de abajo hacia arriba.

Adicionalmente, cabe notar que, en modelos incompletos, las interpretaciones de predicados son pares ordenados, el primer elemento siendo la extensión (un conjunto) y el segundo la anti-extensión (otro conjunto). De ese modo, el código de arriba solo funcionará si se lo ejecuta en un modelo completo. Para que funcionase también en modelos incompletos, debería complejizárselo (p.e. en líneas como "`if (x1, z) in before:`" debería chequearse primero si *before* es un objeto de tipo *frozenset*, en cuyo caso se ejecuta lo anterior, o si es un objeto de tipo *tuple*, en cuyo caso el código a ejecutar sería "`if (x1, z) in before[0]:`"). Por mor de la legibilidad, se lo deja de este modo aquí.

Por último, ya que no hay consola ni por tanto función *print*, para ver el contenido de una variable puede usarse el método *messagebox.showinfo*([título de la ventana], [mensaje]), que abrirá un cuadro de diálogo con el contenido dado como mensaje. Esto puede resultar útil si el resultado de ejecutar el método (subsección siguiente) no es el esperado.

4.5.2. Ejecución de métodos de determinación

Una vez definido un método de determinación, es posible ejecutarlo tomando a un modelo como *input*. Esto se realiza desde la parte inferior de la pestaña **Determination methods**. Para ilustrar esta funcionalidad se agrega al archivo suplementario un modelo m_7, que contiene dos cuerpos b_1 y b_2, y tres tiempos t_1, t_2 y t_3. Los cuerpos chocan en t_1 con velocidades de 3 y –3, y luego vuelven

a chocar en t_2 con las velocidades resultantes del primer choque (para que la situación sea plausible, imagínese que los cuerpos son dos bolillas rodando sobre una pista circular). Sus velocidades en t_2 y en t_3 no están determinadas (recuérdese que los modelos completos permiten este tipo de falla de denotación, véase la nota al pie 33). Para ejecutar el método recién cargado en este modelo se deben llenar los datos de la izquierda (método manual, el nombre del método y el nombre del modelo) y oprimir el botón **Determine**.

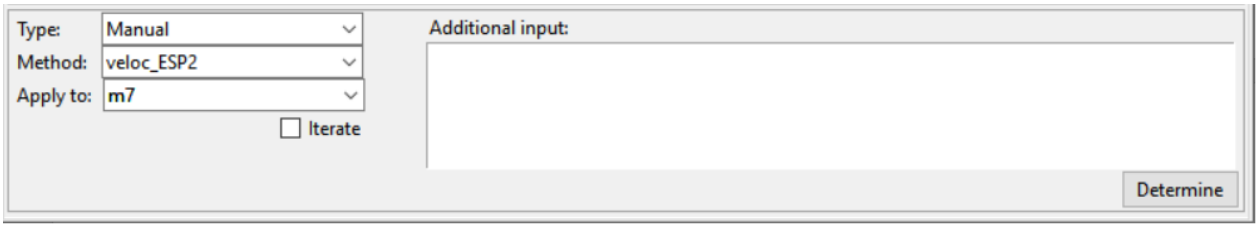


Fig. 4.33. Ventana para la ejecución de un método de determinación.

Tras la ejecución, el programa devolverá o bien un mensaje de error (si hubo algún error en la ejecución) o bien un mensaje con el resultado de la determinación, preguntando si se desea guardar el resultado en el modelo. En este caso, el mensaje se ve así:

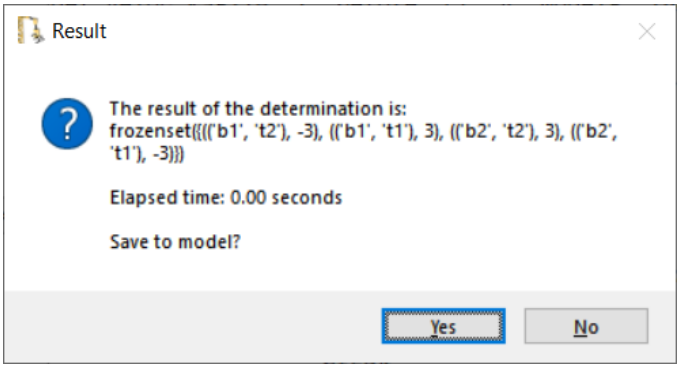


Fig. 4.34. Resultado de la ejecución del método anterior en m_7.

Nótese que el programa encontró las velocidades para t_2. Si se guarda el resultado en el modelo y se vuelve a ejecutar el mismo método, se encontrarán las velocidades para t_3.

Activar la opción **Iterate** antes de ejecutar un método (véase la figura 4.19) hará que el programa continúe ejecutando el método hasta que la denotación del concepto a determinar no se

modifique. Por ejemplo, en este caso, activar esta opción hará que el programa encuentre tanto los valores de t_3 como los de t_2 en una única ejecución.

4.5.3. El segundo método de testeo de reconstrucciones

El procedimiento realizado en la sección anterior es el análogo formal a la realización de una predicción. Se partió de una situación con información parcialmente incompleta o desconocida y aplicando las leyes de la teoría (o más precisamente, un algoritmo basado en esas leyes) se obtuvieron los valores que se desconocían. De ese modo, el primer método de testeo de reconstrucciones (subsección 4.2.3) puede adaptarse para chequear la corrección de la formalización de un método de determinación como un algoritmo. Es decir, puede concluirse que un método de determinación quedó bien formalizado si las predicciones que se realizan con la contraparte informal del algoritmo dan el mismo resultado que las que arroja Reconstructor al ejecutar ese algoritmo.

Adicionalmente, la posibilidad de completar modelos utilizando métodos T-teóricos habilita un segundo tipo de chequeo de una reconstrucción, que funciona por coherencia interna. El punto de partida es que puede evaluarse la satisfacción de las leyes (desde el modelo **Query**) de modelos completados con un método de determinación. El segundo método consistiría entonces en chequear que los modelos así completados satisfagan las leyes. Si no lo hacen, entonces o bien el método o bien las leyes están mal formalizados (o bien hay una incoherencia en la teoría objeto, pero se está asumiendo que ese no es el caso).

4.5.4. Métodos T-no-teóricos y métodos basados en condiciones de ligadura

Por último en esta sección, se introducen algunas funcionalidades adicionales de los métodos manuales que los métodos automáticos de la sección siguiente actualmente no poseen. Ellos son la posibilidad de cargar métodos no-teóricos (que presuponen lenguaje externo a la teoría) y la posibilidad de usar métodos basados en condiciones de ligadura.

Para lo primero, el ejemplo a considerar será la determinación de la velocidad de un cuerpo en MCAR a partir de la distancia recorrida en cierto tiempo. Más precisamente, la velocidad de un cuerpo en t_x (que en MCAR representa un instante de tiempo) puede medirse tomando algún intervalo de duración $2j$, $[t_{x-j}, t_{x+j}]$ (donde t_{x-j} debería ser posterior al instante anterior a t_x en el modelo de MCAR; y t_{x+j} deberá ser anterior al siguiente instante en el modelo), y midiendo la diferencia en la posición inicial y final de un cuerpo en ese intervalo. Si representamos a la posición inicial y final como dos coordenadas p_i y p_f sobre un eje,[35] el método consistiría en aplicar la ecuación $v(b_y, t_x) = (p_f - p_i) / 2j$. Este método es MCAR-no-teórico ya que la determinación de la velocidad de un cuerpo de este modo no presupone que la sumatoria de las velocidades de los cuerpos se mantiene constante (i.e. que la ley fundamental vale).

Nótese que p_i, p_f y j no fueron explícitamente ingresados como términos de MCAR en la pestaña **Language** (aunque quizás podría sostenerse que términos como "posición" están implícitos en la función velocidad). Sin embargo, pueden ser cargados como *input* adicional (primero ingresándolos en la lista de *inputs*, escribiendo sobre la lista desplegable **Inputs** y luego dando sus interpretaciones al momento de la ejecución en el cuadro **Additional input**).

[35] Recuérdese que estamos tratando al espacio como unidimensional en MCAR, por simplicidad. Véase la nota al pie 1.

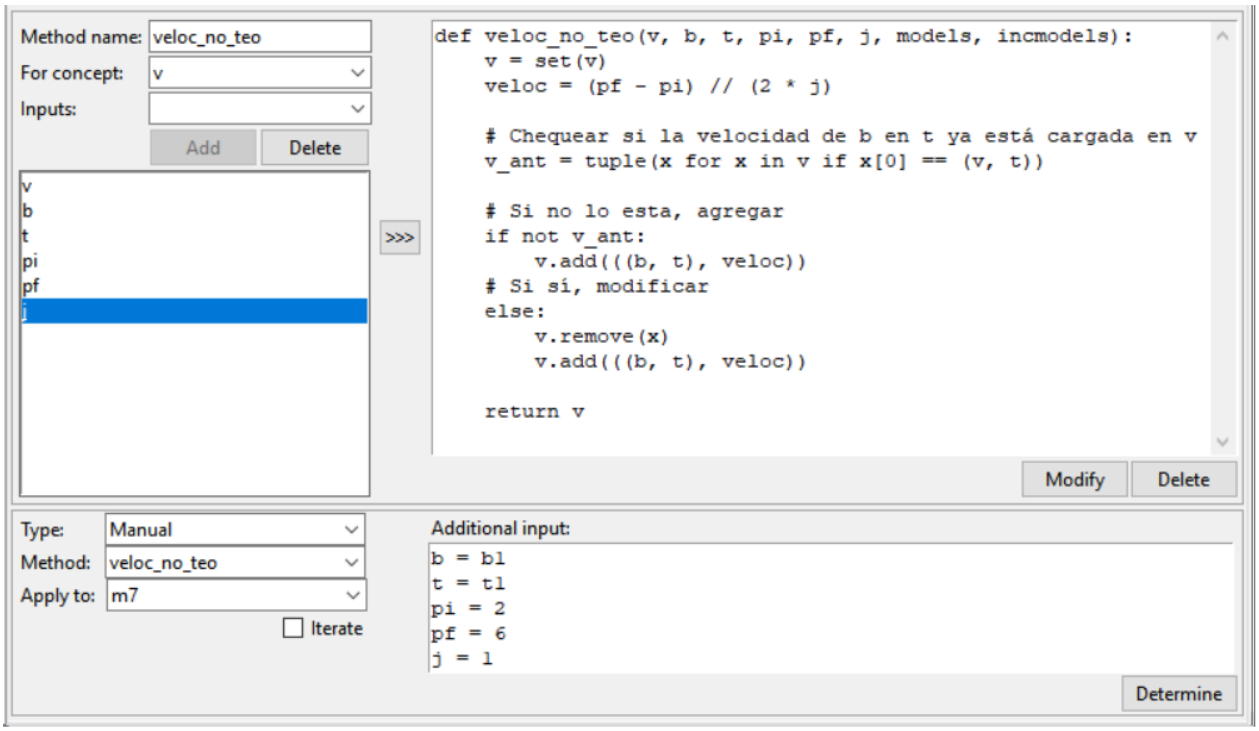


Fig. 4.35. Carga y ejecución de un método de determinación MCAR-no-teórico.

Las interpretaciones del input adicional al momento de ejecutar deben ser provistas en el siguiente formato:

- Cada interpretación va en una línea separada (se debe poner un Enter entre ellas)
- Cada línea debe tener la forma "[término] = [interpretación]"
- Las interpretaciones van en el mismo formato que en la pestaña **Models**.

La ejecución del método no-teórico de la figura 4.21, con las interpretaciones provistas en esa figura, dan a b_1 una velocidad de 2 en t_2.

La carga de métodos T-no-teóricos brinda una posibilidad adicional. Comparar el output de un método de determinación teórico (manual o automático) con el de uno no-teórico equivale a realizar (formalmente) un test de la teoría objeto. Es decir, un test de una teoría consiste en comparar los resultados de la aplicación de un método T-no-teórico con uno T-teórico para el mismo concepto (véase el capítulo 6 para más sobre este punto). Así, si ya se tiene confianza en la formalización de los métodos de determinación Reconstructor puede funcionar como un contrastador automático de la teoría objeto.

La segunda posibilidad adicional es la de cargar un método basado en una condición de ligadura. Esto es posible ya que, como se dijo arriba, los algoritmos reciben siempre a dos variables "models" e "incmodels", que contienen a los modelos completos e incompletos, como parte del *input*. En mayor detalle, estas dos variables son diccionarios de Python, que contienen como *keys* a los nombres de los modelos (*strings*) y como *values* a los modelos mismos (que, como se dijo, son también *dicts*). Adicionalmente, una de ellas contendrá a la *key* "actual", que contendrá al modelo en el que se está ejecutando el método (si el modelo es completo, tal key estará en la variable "models", caso contrario en "incmodels"). De ese modo, el algoritmo puede "ver" lo que hay en modelos distintos al que se está completando. Así, un método que utiliza la condición de ligadura $\mathbf{C_1}$ (igual velocidad en un mismo tiempo) se vería como sigue:

```
def veloc_condic_lig(B, T, v, models, incmodels):
    v = set(v)
    for m in models.values():
        # Para todo modelo distinto del actual
        if m != models['actual']:
            # Para todo cuerpo y todo tiempo del modelo actual
            for b in B:
                for t in T:
                    # Si el modelo actual no tiene v(b,t)
                    if not tuple(x for x in v if x[0] == (b, t)):
                        # Ver si está en el otro modelo
                         v_bt_m = tuple(x for x in m['v'] if x[0] == (b, t))
                         if v_bt_m:
                            # Si está, agregar a v y salir de la función
                            v.add(((b, t), v_bt_m[0][1]))
                            return v
    return v
```

Al ejecutar este método en m_7, lo que hace es completar las velocidades faltantes en base a la información contenida en otros modelos (completos, el algoritmo puede adaptarse fácilmente a buscar también en los incompletos). Nuevamente, puede usarse la función **Iterate** para que el programa extraiga toda la información posible en una única ejecución.

4.6. Métodos de determinación automáticos

Como se expuso en la sección anterior, la utilización de métodos de determinación manuales presupone conocimientos de programación en Python. En esta sección se introducen los métodos de determinación automáticos, los cuales pueden ejecutarse para completar un modelo sin necesidad de proveer explícitamente un algoritmo para ello (y por tanto, sin presuponer tales conocimientos). En cambio, los algoritmos correspondientes son "inferidos" (en un sentido amplio) por el programa a partir de las condiciones de verdad de los axiomas de la teoría.

Para ello, se muestra primero cómo pueden ejecutarse los métodos automáticos, para luego pasar a explicar el funcionamiento interno del programa en la ejecución de tales métodos. Ellos serán caracterizados matemáticamente a partir de una generalización del aparato matemático introducido por Kripke (1975), diseñado originalmente para lidiar con las paradojas semánticas resultantes de introducir un predicado de verdad transparente a un lenguaje que contiene autorreferencia. Esta generalización extiende tal construcción para permitir la determinación de la extensión y anti-extensión de cualquier ítem del lenguaje objeto, no solo del predicado de verdad.

4.6.1. Idea general y utilización de métodos de determinación automáticos

Ejecutar un método de determinación automático en Reconstructor es muy sencillo. En la parte inferior de la pestaña **Determination methods**, selecciónese **Type: Automatic**, y luego el nombre de la ley (axioma impropio, ley fundamental o especialización) y del modelo (los métodos automáticos funcionan solo con modelos incompletos, más abajo se explicará por qué), y presiónese **Determine**. Esto hará que Reconstructor aplique la ley al modelo e intente encontrar denotaciones nuevas o diferentes a las ya cargadas. Más precisamente, lo que el programa hará será establecer si, dadas las interpretaciones ya cargadas y la oración formal dadas como *input*, la última constriñe a las interpretaciones de algún (o más de un) término de modo tal que haya un único valor posible para él. En caso de que lo haya, le asignará al término esa interpretación. Llamaré a esto *imponer* [*enforce*] la oración en el modelo.

Un ejemplo sencillo, para ilustrar este punto, sería el siguiente. Supóngase que se trabaja con un lenguaje formal que contiene un dominio *D*, dos constantes c_1 y c_2, y una relación binaria *R*. Supóngase además que, en un modelo *m*, el dominio consta de los objetos $\{o_1, o_2\}$ (nombrados por c_1 y c_2, respectivamente) y que *R* es como sigue:

$$R_m^+ = \{\langle o_1, o_2\rangle\}$$
$$R_m^- = \{\langle o_2, o_1\rangle\}$$

En este modelo, las oraciones $R(c_1, c_1)$ y $R(c_2, c_2)$ recibirán valor indeterminado. Si a un método automático se le da este modelo y la oración formal "$\forall x \in D\ (R(x, x))$", entonces el método completará el modelo introduciendo $\langle o_1, o_1\rangle$ y $\langle o_2, o_2\rangle$ a R_m^+, ya que esta es la única denotación posible para *R* que hace verdadera a la oración. En cambio, si le pide imponer la oración formal "$\exists x \in D\ (R(x, x))$", el método no agregará ninguna denotación nueva, ya que la denotación de *R* que satisface la oración no es unívoca.

Un ejemplo más interesante proviene de aplicar la especialización ESP2 de MCAR al modelo incompleto i_2 introducido arriba (que contiene solo las velocidades para t_1 para dos cuerpos, así como la información de que chocan en t_1). Ello da como resultado las velocidades para t_2 (el lector puede hacer la prueba por su cuenta desde el archivo suplementario).

A continuación se explica el modo en el que se logró que Reconstructor haga este tipo de inferencias automáticas. Para ello, comenzaré con una breve subsección introductoria al aparato kripkeano de puntos fijos, para luego mostrar cómo este puede generalizarse y cómo tal generalización está implementada en Reconstructor.

4.6.2. La teoría de la verdad de Kripke

En esta subsección se describen algunas de las motivaciones y problemas originales, así como el aparato matemático introducido por Kripke, para lidiar con ellos. La descripción será breve y al punto dado que ello será suficiente para mis propósitos. El lector que desee profundizar en estos temas puede consultar Kripke (1975) y Barrio (2014).

El problema original consiste en que, en un lenguaje formal que contiene autorreferencia (p.e. cualquier lenguaje que contenga a la aritmética de Robinson), si la lógica es clásica, no puede agregarse un predicado de verdad transparente al lenguaje (i.e. uno que satisfaga el esquema-axioma T: $\phi \leftrightarrow Tr(\text{`}\phi\text{'})$).[36] Esto se debe a que el lema de diagonalización de Gödel permitiría probar lo siguiente como teorema $\vdash_{R+T} \lambda \leftrightarrow \neg Tr(\text{`}\lambda\text{'})$, es decir, que hay una oración λ que es equivalente a la negación de la verdad de sí misma (en términos más sencillos λ afirmaría "yo no soy verdadera", por eso es llamada la *oración del mentiroso*). Sin embargo, esto genera una paradoja. En términos semánticos, no puede darse una asignación de valor de verdad consistente a λ, ya que (*i*) Si λ es verdadera, entonces (por la instancia de diagonalización) $\neg Tr(\text{`}\lambda\text{'})$ también debe serlo, y por tanto $Tr(\text{`}\lambda\text{'})$ debe ser falsa; pero (por la instancia correspondiente del esquema-T) ello implica que λ es falsa. En cambio, si (*ii*) λ es falsa, entonces (por la instancia de diagonalización) $\neg Tr(\text{`}\lambda\text{'})$ también debe serlo, y por tanto $Tr(\text{`}\lambda\text{'})$ debe ser verdadera, lo cual (por la instancia del esquema-T) implica que λ también debe serlo.

Según Kripke, el problema con oraciones como la del mentiroso es que no están *fundadas*. El valor de verdad de oraciones como $Tr(\text{`}Tr(\text{`}1+1=2\text{'})\text{'})$ depende, en última instancia, de hechos que son independientes del predicado de verdad y su interpretación (p.e. en este caso, la verdad de esta oración depende de la verdad de la oración 1 + 1 = 2). En cambio, la oración del mentiroso no dependería de ningún "hecho" tal. No hay nada "en el mundo" (i.e. en la teoría de base a la cual se agrega el predicado de verdad) que fije el valor de verdad de λ.

El modo de Kripke de lidiar con esto consiste en partir de un modelo de base, que tiene las interpretaciones de los términos de la teoría de base ya determinadas, pero que contiene una extensión y una anti-extensión vacías para el predicado de verdad, y luego ir construyendo esta extensión y anti-extensión en pasos sucesivos. Por ejemplo, considérese el caso de la aritmética. En el modelo de base (llámese m_0) ocurre que:

(*a*) $v_{m0}(1 + 1 = 2) = 1$

(*b*) $v_{m0}(1 + 2 = 5) = 0$

[36] También puede formularse este requisito por medio de dos reglas, T-in y T-out, que afirman $\phi \therefore Tr(\text{`}\phi\text{'})$ y $Tr(\text{`}\phi\text{'}) \therefore \phi$, respectivamente.

Pero dado que el predicado de verdad tiene una extensión y una anti-extensión vacías, también será el caso que:

(*c*) $v_{m0}(Tr(\text{`}1 + 1 = 2\text{'})) = 0.5$
(*d*) $v_{m0}(Tr(\text{`}1 + 2 = 5\text{'})) = 0.5$
(*e*) $v_{m0}(Tr(\text{`}Tr(\text{`}1 + 1 = 2\text{'})\text{'})) = 0.5$
(*f*) $v_{m0}(Tr(\text{`}Tr(\text{`}Tr(\text{`}1 + 1 = 2\text{'})\text{'})\text{'})) = 0.5$

La idea consiste en ir generando una serie de modelos sucesivos que contienen extensiones y anti-extensiones cada vez más completas para el predicado de verdad. Para ello, hay una regla de revisión, que permite generar una nueva interpretación para el predicado de verdad dada una interpretación en un modelo previo. La regla afirma que:

(R1) Si $v_{mi}(\phi) = 1$ entonces 'ϕ' $\in I_{mi+1}(Tr)^{+}$
(R2) Si $v_{mi}(\phi) = 0$ entonces 'ϕ' $\in I_{mi+1}(Tr)^{-}$

Es decir, el nombre de toda oración verdadera en un modelo *i* entra en la extensión del predicado de verdad en el modelo *i*+1, y lo mismo con toda oración falsa y la anti-extensión. Así, dado que en m_0 1 + 1 = 2 es verdadera y 1 + 2 = 5 es falsa, '1 + 1 = 2' entrará en la extensión del predicado de verdad en m_1, mientras que '1 + 2 = 5' entrará en la anti-extensión. Así, en m_1 ocurre que:

(*c*) $v_{m1}(Tr(\text{`}1 + 1 = 2\text{'})) = 1$
(*d*) $v_{m1}(Tr(\text{`}1 + 2 = 5\text{'})) = 0$
(*e*) $v_{m1}(Tr(\text{`}Tr(\text{`}1 + 1 = 2\text{'})\text{'})) = 0.5$
(*f*) $v_{m1}(Tr(\text{`}Tr(\text{`}Tr(\text{`}1 + 1 = 2\text{'})\text{'})\text{'})) = 0.5$

Nótese que, en m_1, (*e*) sigue teniendo valor indeterminado ya que el código '*Tr*('1 + 1 = 2')' no está ni en la extensión ni en la anti-extensión del predicado de verdad. Sin embargo, dado que *Tr*('1 + 1 = 2') es verdadera en m_1, por la regla de revisión, ese código estará en la extensión de *Tr* en m_2, y por tanto:

(*c*) $v_{m2}(Tr(\text{`}1 + 1 = 2\text{'})) = 1$

(*d*) $v_{m2}(Tr(\text{`}1 + 2 = 5\text{'})) = 0$

(*e*) $v_{m2}(Tr(\text{`}Tr(\text{`}1 + 1 = 2\text{'})\text{'})) = 1$

(*f*) $v_{m2}(Tr(\text{`}Tr(\text{`}Tr(\text{`}1 + 1 = 2\text{'})\text{'})\text{'})) = 0.5$

De ese modo, la extensión y la anti-extensión del predicado de verdad se van completando progresivamente, pero siempre abarcando únicamente a oraciones fundadas. El proceso no se detiene tras ningún número finito de pasos, pero sí puede probarse que existe un cardinal transfinito α, tal que aplicar la regla de revisión a m_α da como resultado a m_α mismo. Esto es lo que se denomina un *punto fijo*, un modelo tal que revisarlo no resulta en ninguna modificación. En todo punto fijo toda oración fundada tiene un valor de verdad determinado.

En la siguiente subsección se muestra como generalizar este aparato matemático para permitir completar las denotaciones de términos diferentes al predicado de verdad.

4.6.3. Una generalización del aparato kripkeano

Un modo ilustrativo de entender la extensión que propondré en esta subsección es comenzar preguntándose por qué Kripke propone las reglas (R1) y (R2) de la sección anterior como reglas de revisión, y no otras. La respuesta es que ello se debe a que se desea que toda instancia del esquema-T sea verdadera en el modelo final. Esto es, en mi terminología anterior, a que se desea *imponer* el esquema-T sobre el modelo de base m_0. Así, la regla (R1) resulta de que:

- Para imponer un bicondicional (i.e. para volver verdadero a algo de la forma A ↔ B en un modelo *m*), es necesario hacer que $v_m(A) = v_m(B)$. Si $v_m(A) = 1$ y $v_m(B) = 0.5$ entonces, dado que 0.5 es un estado menos informativo que 1, se debe hacer que $v_m(B) = 1$ (i.e. imponer B)
- Si B es una atómica (p.e. algo de la forma $P(c)$), imponerla implica hacer que $I(c) \in I(P)^+$.

Dado que las oraciones B son de la forma $Tr(\text{`A'})$ y que el predicado de verdad se encuentra inicialmente indeterminado en m_0, estas consideraciones nos dan como resultado (R1). Este modo de razonar nos sirve heurísticamente para pensar en cómo deberían ser las reglas que permitan

imponer oraciones diferentes al esquema-T (i.e. que usen otras oraciones como reglas de revisión). Por ejemplo, si se desease imponer algo de la forma A ∧ B, debería imponerse tanto A como B (i.e. asegurar que v_m(A) = 1 y v_m(B) = 1, en lugar de meramente asegurar que v_m(A) = v_m(B), como ocurre para un bicondicional).

Una generalización de esta idea puede ser presentada introduciendo una función recursiva (metateórica) *e*, que representa a la función de imposición [*enforcement*]. Esta función tomará como argumentos a una oración ϕ y un modelo *m,* y devolverá a otro modelo *m*', que puede poseer alguna/s denotaciones completadas y/o modificadas para hacer verdadera a ϕ. Las cláusulas para ello son las siguientes:

(e_1) Si ϕ es una atómica de la forma $R(t_1, \ldots, t_n)$ entonces $e(\phi, m)$ implica que $\langle I(t_1), \ldots, I(t_n)\rangle \in I_{mi+1}(R)^+$

(e_2) $e(\phi \wedge \psi, m_i) = e(\psi, e(\phi, m_i))$

(e_3) $e(\phi \rightarrow \psi, m_i) = \begin{cases} e(\psi, m_i) & \text{si } v_{mi}(\phi) = 1 \\ e(\neg\phi, m_i) & \text{si } v_{mi}(\phi) = 0.5 \text{ y } v_{mi}(\psi) = 0 \\ m_i & \text{sino} \end{cases}$

(e_4) $e(\phi \vee \psi, m_i) = e(\neg\phi \rightarrow \psi, m_i)$

(e_5) $e(\phi \leftrightarrow \psi, m_i) = e((\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi), m_i)$

(e_6) $e(\forall x \in t\ \phi(x), m_i) = e(\bigwedge\phi(c), m_i)$ para toda constante *c* que denote un objeto de I(*t*)

(e_7) $e(\exists x \in t\ \phi(x), m_i) = e(\bigvee\phi(c), m_i)$ para toda constante *c* que denote un objeto de I(*t*)

(e_8) Si ϕ es una atómica negada de la forma $\neg R(t_1, \ldots, t_n)$ entonces $e(\phi, m)$ implica que $\langle I(t_1), \ldots, I(t_n)\rangle \in I_{mi+1}(R)^-$

(e_9) $e(\neg\neg\phi, m_i) = e(\phi, m_i)$

(e_{10}) $e(\neg(\phi \wedge \psi), m_i) = e(\neg\phi \vee \neg\psi), m_i)$

(e_{11}) $e(\neg(\phi \vee \psi), m_i) = e(\neg\phi \wedge \neg\psi), m_i)$

(e_{12}) $e(\neg(\phi \rightarrow \psi), m_i) = e(\phi \wedge \neg\psi, m_i)$

(e_{13}) $e(\neg(\phi \leftrightarrow \psi), m_i) = e((\neg\phi \rightarrow \psi) \wedge (\neg\psi \rightarrow \phi), m_i)$

(e_{14}) $e(\neg\forall x \in t\ \phi(x), m_i) = e(\exists x \in t\ \neg\phi(x), m_i)$

(e_{15}) $e(\neg\exists x \in t\ \phi(x), m_i) = e(\exists x \in t\ \neg\phi(x), m_i)$

Nótese que (e_8)-(e_{15}) son cláusulas para las negaciones de (e_1)-(e_7) (más la cláusula para la doble negación). Las cláusulas base de la función recursiva serían (e_1) y (e_8). Respecto de las cláusulas (e_6) y (e_7), $\phi(x)$ representa a una oración ϕ que tiene a x como variable libre, y $\phi(c)$ a la oración que resulta de sustituir las ocurrencias libres de x por la constante c. Cabe aclarar que, en el contexto en que me interesa aplicar este aparato, el término t siempre denotará un conjunto finito (dado que, como se expuso, en Reconstructor los cuantificadores siempre están acotados a conjuntos finitos), y por lo tanto las oraciones $\bigwedge\phi(c)$ y $\bigvee\phi(c)$ serán finitarias y bien formadas en un cálculo de primer orden.[37]

A modo de ejemplo, considérese el siguiente ejemplo sencillo. Tómese un lenguaje con un dominio D, dos constantes de individuo c_1 y c_2, un predicado monádico P y un predicado diádico R. Sea m un modelo incompleto tal que:

$m = \langle D_m, c_m, P_m^+, P_m^-, R_m^+, R_m^- \rangle$
$D_m = \{o_1, o_2\}$
$c_{1m} = o_1$
$c_{2m} = o_1$
$P_m^+ = \{\}$
$P_m^- = \{\}$
$R_m^+ = \{\}$
$R_m^- = \{\}$

Y supóngase que se pide $e(\neg P(c_1) \wedge \forall x \in D\,(R(x, x)), m)$. Lo que ocurrirá es lo siguiente.

- Dado que la oración es conjuntiva, por la cláusula (e_2), e primero impondrá $\forall x \in D\,(R(x, x))$ sobre m, y luego impondrá $\neg P(c_1)$ sobre el resultado m'.
- Por la cláusula (e_6), imponer $\forall x \in D\,(R(x, x))$ sobre m es igual a imponer $(R(c_1, c_1) \wedge R(c_2, c_2))$ sobre m.
- Nuevamente por la cláusula (e_2), esto equivale a imponer $R(c_2, c_2)$ sobre m, y luego $R(c_1, c_1)$ sobre el resultado m''.

[37] En caso de que se desee aplicar este tipo de enfoque independientemente de este contexto, estas clausulas pueden ser fácilmente reformuladas en términos asignacionales estándar.

- Imponer $R(c_2, c_2)$ sobre m significa, por la cláusula (e_1), tomar m y agregar $\langle o_2, o_2 \rangle$ a $I(R)^+$ para obtener m''.
- Imponer $R(c_1, c_1)$ sobre m'' significa, por la cláusula (e_1), tomar m'' y agregar $\langle o_1, o_1 \rangle$ a $I(R)^+$ para obtener m'.
- Imponer $\neg P(c_1)$ sobre m' significa, por la cláusula (e_2), tomar m' y agregar o_1 a $I(P)^-$ para obtener m'''.

De ese modo, el modelo final que devuelve $e(\neg P(c_1) \wedge \forall x \in D\ (R(x, x)), m)$ es como sigue:

$m''' = \langle D_{m'''}, c_{m'''}, P_{m'''^+}, P_{m'''^-}, R_{m'''^+}, R_{m'''^-} \rangle$

$D_{m'''} = \{o_1, o_2\}$

$c_{1m'''} = o_1$

$c_{2m'''} = o_1$

$P_{m'''^+} = \{\}$

$P_{m'''^-} = \{o_1\}$

$R_{m'''^+} = \{\langle o_1, o_1 \rangle, \langle o_2, o_2 \rangle\}$

$R_{m'''^-} = \{\}$

Es decir, una única llamada a e logró completar parcialmente las denotaciones de P y de R.

4.6.4. Extensiones monótonas y no-monótonas

Las cláusulas (e_1)-(e_{15}) no especifican un valor de verdad inicial para la oración ϕ a imponer en el modelo m. Por ejemplo, es posible imponer la oración $\neg R(c_1, c_1) \wedge P(c_1)$ sobre el modelo m''' recién obtenido (en el que ambos conyuntos de la oración son falsos). En ese caso, de acuerdo a las cláusulas, o_1 saldrá de P^- y entrará en P^+, y $\langle o_1, o_1 \rangle$ saldrá de R^+ y entrará en R^-, en el resultado final. Diré que una imposición de una oración ϕ en un modelo m (llámese al resultado de esta operación m*) es *monótona* cuando ninguna oración ψ es tal que $v_m(\psi) \neq v_{m*}(\psi)$ y ambas $v_m(\psi)$ y

$v_{m*}(\psi)$ son valores de verdad determinados (0 o 1).[38] Es decir, cuando ninguna oración pasa de ser verdadera a ser falsa (o viceversa). En una extensión monótona solamente se agrega información nueva, previamente desconocida, mientras que en una no-monótona se modifica información ya conocida.

En caso de que una extensión sea no-monótona Reconstructor mostrará una advertencia y preguntará si se desea guardar el resultado como un modelo nuevo. Por ejemplo, si se carga un modelo incompleto i_3 que contiene dos cuerpos b_1 y b_2 que chocan con velocidades iniciales 1 y −1, y rebotan con velocidades −2 y 2, imponer la segunda especialización resultará en una extensión no monótona, ya que esa especialización constriñe la interpretación de las velocidades en t_2 a −1 y 1:

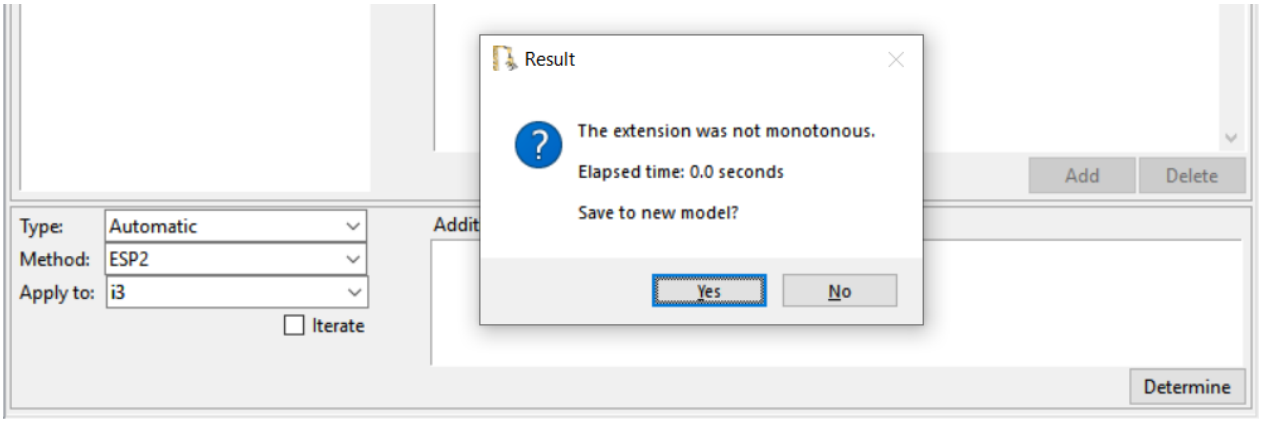


Fig. 4.36. Advertencia en Reconstructor tras una extensión no monótona

En el contexto de imponer una ley empírica sobre un modelo que representa una situación empírica inicial, una falla de monotonía puede indicar o bien un error en la información inicial cargada en el modelo, o bien un error en la formalización de una ley, o bien la falsedad de las leyes de la teoría. De ese modo (si se asume que lo primero no es el caso y se sabe que lo tercero no lo es), esto indica un tercer modo posible de contrastar una reconstrucción. A saber, la imposición de las leyes a los modelos incompletos (a través de los métodos automáticos) debe resultar en extensiones monótonas.

[38] Ya que este aparato permite que ocurran estas cosas, también puede verse como una generalización de aquel presentado por Herzberger, Gupta y Belnap (conocido como la teoría de la revisión de la verdad [*revisión theory of truth*], véase Kremer, 2016), a una lógica no clásica (con más de dos valores de verdad).

Nótese por último que, en caso de imposiciones no monótonas, el orden en que se escriban las subfórmulas de la oración ϕ puede alterar el resultado final. Por ejemplo, $e(\psi \wedge \neg\psi, m)$ dará un resultado distinto a $e(\neg\psi \wedge \psi, m)$, ya que la cláusula ($e_2$) impone primero la fórmula de la derecha y luego la de la izquierda. Sin embargo, dado que (en este contexto) una imposición no monótona señala que algo salió mal, es indiferente cuál de los dos resultados sea el final, en tanto y en cuanto el programa advierta que un problema ocurrió (ya que la ocurrencia del problema indica que no se debe confiar en el resultado final de la determinación).

4.6.5. Implementación en Reconstructor

Por último, es relevante discutir algunos de los detalles de la implementación del aparato matemático anterior en Reconstructor.

En primer lugar, la opción **Iterate** también funciona para los métodos automáticos. En este caso, el efecto será que el programa continúe aplicando la regla de revisión al modelo inicial hasta que el modelo final se estabilice (i.e. hasta que aplicarle nuevamente la regla de como resultado a él mismo). En terminología kripkeana, lo que esto hace es *buscar un punto fijo*.

En segundo lugar, es posible explicar ahora por qué los métodos automáticos funcionan solo con modelos incompletos. Para ello, supóngase que, en el ejemplo de la sección 4.6.3. se pide imponer $P(c_1)$ sobre el primer modelo, m. Por las cláusulas, ello hará que $o_1 \in \mathrm{I}(P)^+$, de modo que la nueva extensión de P será $\{o_1\}$. Ahora bien, dado que los modelos completos en Reconstructor solo poseen extensión y no anti-extensión (o, mejor dicho, Reconstructor asume que la anti-extensión es el complemento de la extensión), determinar en un modelo completo que $\mathrm{I}(P) = \{o_1\}$ implicaría afirmar que $\neg P(c_2)$ es verdadera, algo que claramente no se desea.

En tercer lugar, el lenguaje que el usuario puede cargar como propio incluye la posibilidad de cargar símbolos de función, que se interpretan de modo distinto a los predicados. De ese modo, es necesario agregar una cláusula más a las ya dadas para lidiar con esto:

(e_{16}) Si ϕ es una atómica de la forma $f(t_1, \ldots, t_n) = b$ entonces $e(\phi, m)$ implica $\langle\langle \mathrm{I}(t_1), \ldots, \mathrm{I}(t_n)\rangle, \mathrm{I}(b)\rangle \in \mathrm{I}_{mi+1}(f)$

En cuarto lugar, otro punto importante es que el lenguaje de base de Reconstructor incluye una serie de términos (constantes, relaciones y funciones) por default, que siempre son interpretadas de modo estándar en todo modelo. Las interpretaciones de estos términos no pueden modificarse en el curso de una revisión. Por ejemplo, si se pide a Reconstructor imponer la oración "1 = 2" en un modelo el resultado será que no hace nada.

Tres de las relaciones por default introducen además cláusulas adicionales para la función de imposición:

(e_{17}) $e(t_1 = t_2, m)$ implica $t_{1m+1} = t_{2m}$, si t_1 no comienza con un símbolo de función (caso contrario, entra en la cláusula (e_{16})), ni es un término por default (caso contrario, $t_{2m+1} = t_{1m}$, si t_2 no comienza con un símbolo de función).

(e_{18}) $e(t_1 \in t_2, m)$ implica $t_{1m} \in t_{2m+1}$, si t_{2m+1} es un conjunto.

(e_{19}) $e(\neg t_1 \in t_2, m)$ implica $t_{1m} \notin t_{2m+1}$, si t_{2m+1} es un conjunto (es decir, si t_{1m} pertenecía a la denotación de t_{2m}, sale de ella en t_{2m+1}).

(e_{20}) $e(t_1 \subseteq t_2, m) = e(t_3 \in t_2, m)$ para todo t_3 que denota a un elemento de t_1 en m.

El resto de las relaciones por default (o sus negaciones) no tienen cláusulas asociadas porque no determinan un valor unívoco para el resultado. Por ejemplo "$\neg t_1 = t_2$" no fija una interpretación única para t_1 ni para t_2. En las cláusulas (e_{18}) y (e_{19}) el comportamiento es ligeramente distinto (aunque similar) si t_2 es un símbolo de función aplicado (p.e. $e(1 \in f(2), m)$ introducirá al número 1 en el segundo elemento de la tupla que representa a ese argumento y ese valor en I(f)).

Por último, puede verse a este aparato matemático (y a su implementación computacional) como un nuevo tipo de enfoque en el área del razonamiento automatizado. En esta área, los enfoques suelen ser sintácticos, i.e. trabajar en teoría de la prueba, no en teoría de modelos. En ese sentido, los métodos automáticos de Reconstructor pueden ser vistos como el equivalente semántico a un probador automático de teoremas (*ATPs* en su sigla usual). Sin embargo, dado que estos métodos están en una fase inicial (mientras que algunos *ATP*s tienen décadas de desarrollo ya) este equivalente semántico tiene algunas debilidades en el modo en que maneja a la matemática estándar. Por ejemplo, si se pide a Reconstructor imponer una oración como "$c_1 = 1 + 1$", asignará la denotación 2 a c_1 en el modelo siguiente. Sin embargo, si se pide imponer "$c_1 + 1 = 3$" no hará nada. Esto se debe a que $e(c_1 + 1 = 3, m)$ contiene una igualdad con dos términos por default de

ambos lados (la función suma y el numeral 3), que Reconstructor no puede modificar. Es decir, Reconstructor no despeja la ecuación, sino que se guía únicamente por la forma sintáctica de la oración a imponer. Versiones futuras de este módulo deberían tener mejores formas de lidiar con estos problemas (que, por otro lado, solo surgen cuando las oraciones involucran términos matemáticos por default, para lógica pura la implementación en Reconstructor funciona sin problemas).

Tendré algunas cosas más que decir sobre la comparación entre este y otros enfoques posibles en la conclusión de la tesis.

4.7. Conclusiones

En este capítulo he presentado un programa de computación, llamado Reconstructor, capaz de tomar una reconstrucción formal como *input* y de realizar distintas operaciones sobre ese *input*. Entre ellas, se han destacado las capacidades de (*i*) determinar si algo es un término, una fórmula y una oración bien formadas; (*ii*) consultar por la denotación de un término en un modelo; (*iii*) establecer si las leyes (axiomas impropios, leyes fundamentales y especializaciones) son satisfechas por un modelo dado; (*iv*) establecer si una condición de ligadura es satisfecha por un conjunto de modelos; (*v*) determinar el valor de verdad de oraciones con fallas de denotación para algún término; y (*vi*) completar modelos incompletos utilizando métodos de determinación manuales (cargados como algoritmos) y automáticos (inferidos a partir de las condiciones de verdad de las leyes).

Todas estas operaciones permiten llevar a cabo diversos procedimientos, también propuestos a lo largo del capítulo, para contrastar la reconstrucción formal brindada como *input*. Ellos consisten en (*a*) contrastar la formalización de las leyes y las condiciones de ligadura comparando el valor de verdad esperado informalmente de ellas en una aplicación con el valor de verdad de su formalización en el modelo (o conjunto de modelos) que representa a esa aplicación; (*b*) chequear que el resultado de aplicar un método de determinación sobre un modelo satisfaga posteriormente las leyes de la teoría; y (*c*) observar si la extensión de un modelo incompleto imponiendo las leyes de la teoría sobre él es monótona.

Cabe destacar que todas estas son condiciones necesarias pero no suficientes para considerar adecuada a una reconstrucción. Por ejemplo, en el caso de (*a*), una formalización de una ley debe no solo tener las mismas condiciones de verdad que su contraparte informal, sino que además debe tener una forma (sintáctica) al menos similar a ella. De otro modo, cualquier formalización lógicamente equivalente de una ley empírica que posea las mismas condiciones de verdad que ella sería adecuada, a pesar de que nadie reconocería a muchas de esas formulaciones como formalizaciones adecuadas de la ley original. Esto es patente cuando se introducen subfórmulas irrelevantes. Por ejemplo, una ley que quedaría adecuadamente formalizada bajo la forma "$\forall x$ ($\phi \rightarrow \psi$)" podría ser cargada como "$\neg\exists x$ ($\neg(\neg\phi \vee \neg\neg\psi)) \wedge \top$", satisfaciendo (*a*). Adicionalmente, la adecuación de una reconstrucción también exige cosas como que las derivaciones usuales (informales) de teoremas sean reproducibles formalmente. Diré algo más sobre este punto en la conclusión de la tesis. Por otro lado, para evitar confusiones, cabe aclarar que Reconstructor permite automatizar el proceso de contrastar una reconstrucción *ya dada como input*, no así el proceso de elaborar una reconstrucción.

Por último, cabe destacar que el programa aquí presentado puede tener otros usos independientes del presentado aquí como su objetivo principal. Por ejemplo, es posible que pueda ser utilizado de manera fructífera para la enseñanza de la lógica y/o de la matemática (complementándose con otras herramientas pedagógicas propuestas por el autor, como ser, el sistema TAUT —https://www.taut-logic.com—; véanse también Abramovich, 2013; Fierro, 2015). Un ejemplo sería la enseñanza de la teoría de modelos y la semántica formal. La posibilidad de definir un lenguaje formal, y de cargar interpretaciones y pedir denotaciones para términos ese lenguaje puede ayudar a comprender lo que es un modelo, su estructura y el modo en que se debe interpretar cada tipo de ítem lingüístico. Del mismo modo, la posibilidad de evaluar oraciones y de chequear el log de valuaciones puede ser útil para comprender las cláusulas de las funciones de valuación.

4.8. Apéndice

4.8.1 Lista términos por default

Símbolo	Tipo	Aridad	Notación	Denotación y ejemplo de uso
+	Función	2	Infija	**Suma.** "1 + 1" denotará 2; "1 + (2 + 3)" denotará 6
-	Función	2	Infija	**Resta.** "2 - 1" denotará 1; "0 - (3 - 2)" denotará −1
*	Función	2	Infija	**Multiplicación.** "1 * 1" denotará 1; "2 * (2 * 3)" denotará 12
/	Función	2	Infija	**División.** "4 / 2" denotará 2 "1 / (6 / 3)" denotará 0.5
^	Función	2	Infija	**Potenciación.** "2 ^ 3" denotará 8
mod	Función	1	Prefija	**Módulo.** "mod(0 - 10)" denotará 10
sqrt	Función	1	Prefija	**Raíz cuadrada.** "sqrt(4)" denotará 2
fact	Función	1	Prefija	**Factorial.** "fact(4)" denotará 24
gcd	Función	2	Prefija	**Denominador común mayor.** "gcd(16, 36)" denotará 4
log	Función	2	Prefija	**Logaritmo.** log(*x*, *y*) denota al logaritmo de *x* en base *y*. P.e. "log(8, 2)" denotará 3
union	Función	2	Infija	**Unión.** "{1, 2} union {2, 3}" denotará "{1, 2, 3}"
inters	Función	2	Infija	**Intersección.** "{1, 2} inters {2, 3}" denotará "{2}"
complem	Función	2	Infija	**Complemento.** "{1, 2, 3} complem {2, 3}" denotará {1}
cartp	Función	$n \geq 2$	Prefija	**Producto cartesiano.** "cartp({1, 2}, {3, 4})" denotará {⟨1, 3⟩, ⟨1, 4⟩, ⟨2, 3⟩, ⟨2, 4⟩}, mientras que "cartp({1, 2}, {3, 4}, {5, 6})" denotará {⟨1, 3, 5⟩, ⟨1, 3, 6⟩, ⟨1, 4, 5⟩, ⟨1, 4, 6⟩, …}
pset	Función	1	Prefija	**Conjunto potencia.** "pset(1, 2)" denotará {{}, {1}, {2}, {1, 2}}
Uset	Función	1	Prefija	**Conjunto unión.** "Uset({{1, 2}, {3, 4}})" denotará {1, 2, 3, 4}. El argumento debe ser un conjunto de conjuntos.

Iset	Función	1	Prefija	**Conjunto intersección.** "Iset({{1, 2}, {2, 4}})" denotará {2}. El argumento debe ser un conjunto de conjuntos.
card	Función	1	Prefija	**Cardinalidad.** "card({11, 12, 13})" denotará 3
nullset	Constante	-	-	**Conjunto vacío.** Denota {}
set	Función	*n*	Prefija	**Conjunto.** Toma *n* elementos y devuelve un conjunto que los contiene a todos. "set(1, 2, 3)" denotará {1, 2, 3}. Una notación alternativa para "set(*x*, *y*, *z*, …)" es "{*x*, *y*, *z*, …}"
tuple	Función	*n*	Prefija	**Tupla.** Toma *n* elementos y devuelve una tupla que contiene los contiene, en el orden en que fueron pasados como argumentos. "tuple(2, 1, 3)" denotará ⟨2, 1, 3⟩.
setcompr	Función	2	Prefija	**Comprensión de conjuntos.** Toma un término con una variable libre y un conjunto o tupla, y devuelve al conjunto de elementos que surgen de reemplazar a la variable libre del primer término con los elementos del segundo. P.e. "setcompr(1 + x, {1, 2})" denotará {2, 3}.
tuplecompr	Función	2	Prefija	**Comprensión de tuplas.** Idéntico a setcompr pero devuelve una tupla. Si el segundo argumento es un conjunto, lo ordenará arbitrariamente.
SUM	Función	4	Prefija	**Sumatoria.** Explicado en la sección 4.1.5. "SUM(i1, 1, 3, i1+1)" denotará 9
PROD	Función	4	Prefija	**Productoria.** Explicado en la sección 4.1.5. "PROD(i1, 1, 3, i1+1)" denotará 24
NAT	Constante	-	-	**Conjunto de los números naturales.** Solo puede ser usado como segundo argumento de un "in" – p.e. "1 in NAT"

INT	Constante	-	-	**Conjunto de los números enteros.** Solo puede ser usado como segundo argumento de un "in" – p.e. "1 in INT"
RLS	Constante	-	-	**Conjunto de los números reales.** Solo puede ser usado como segundo argumento de un "in" – p.e. "1.23 in RLS"
Id	Función	2	Prefija	**Función identidad.** Toma una tupla y un número *i* natural y devuelve lo que la tupla contiene en la posición *i*. P.e. "Id(⟨5, 6, 7⟩, 2)" denotará 6.
sample	Función	3	Prefija	**Muestreo.** Toma un conjunto, un tamaño de muestra y "r" / "nr" [con reemplazo / sin reemplazo] y devuelve una muestra al azar con las características deseadas. P.e. "sample({1,2,3}, 2, nr)" puede devolver {1, 2}, {2, 3} o {1, 3}.
randint	Función	2	Prefija	**Entero al azar.** Tomá dos enteros y devuelve un entero al azar entre ellos. P.e. "randint(1, 3)" puede devolver 1, 2 o 3.
randfloat	Función	2	Prefija	**Racional al azar.** Idéntica a randint pero toma dos números racionales y devuelve un número racional.
choice	Función	1	Prefija	**Elemento al azar.** Toma un conjunto o tupla y devuelve un elemento al azar suyo. P.e. "choice({1,2,3})" puede devolver 1, 2 o 3.
range	Función	2	Prefija	**Rango.** Toma dos enteros y devuelve una tupla con los enteros entre ellos (incluidos). P.e. "range(1,4)" denotará ⟨1, 2, 3, 4⟩. Si el primer entero es mayor al segundo devuelve una tupla vacía.
min	Función	*n*	Prefija	**Mínimo.** Toma *n* números (o un conjunto de números) y devuelve el mínimo número. P.e. "min(6, 1, 4)" y "min({6, 1, 4})" denotan 1.
max	Función	*n*	Prefija	**Máximo.** Idéntico a min pero devuelve el máximo.
sin	Función	1	Prefija	**Seno.** "sin(90)" denotará 1.

cos	Función	1	Prefija	**Coseno.** "cos(180)" denotará −1.
tan	Función	1	Prefija	**Tangente.** "tan(45)" denota 0.9999999999999999 (Python aproxima en algunos casos).
arcsin	Función	1	Prefija	**Arcoseno.** Uso similar a las anteriores.
arccos	Función	1	Prefija	**Arcocoseno.** Uso similar a las anteriores.
arctan	Función	1	Prefija	**Arcotangente.** Uso similar a las anteriores.
radians	Función	1	Prefija	**Radianes.** Pasa un número en grados a radianes. P.e. "radians(180)" denota 3.141592653589793 (una aproximación de π)
degrees	Función	1	Prefija	**Grados.** Pasa un número en radianes a grados. P.e. "degrees(Pi)" denota 180.
e_constant	Constante	-	-	**Constante *e*.** Aproximación. Denota 2.718281828459045.
Pi	Constante	-	-	**Constante π.** Aproximación. Denota 3.141592653589793.
modelset	Constante	-	-	**Conjunto de modelos.** Utilizable solo en el módulo de condiciones de ligadura. Explicado en la sección 4.3.1.
denotation	Función	2	Prefija	**Denotación.** Utilizable solo en el módulo de condiciones de ligadura. Explicada en la sección 4.3.1.
indeterminum	Constante	-	-	**Constante de indeterminación.** Siempre tiene una denotación indeterminada.

4.8.2 Abreviaturas para axiomas impropios

Abreviatura	Expresa	Oración completa
reflexive(R,T)	La relación binaria R es reflexiva con respecto a T	forall x in T (R(x, x))

symmetric(R,T)	La relación binaria R es simétrica con respecto a T	forall x in T (forall y in T (if R(x, y) then R(y, x)))
transitive(R,T)	La relación binaria R es transitiva con respecto a T	forall x in T (forall y in T (forall z in T (if (R(x, y) & R(y, z)) then R(x, z))))
antireflexive(R,T)	La relación binaria R es anti-reflexiva con respecto a T	forall x in T (¬R(x, x))
antisymmetric(R,T)	La relación binaria R es anti-simétrica con respecto a T	forall x in T (forall y in T (if R(x, y) then ¬R(y, x)))
antitransirive(R,T)	La relación binaria R es anti-transitiva con respecto a T	forall x in T (forall y in T (forall z in T (if (R(x, y) & R(y, z)) then ¬R(x, z))))
func(*f*, D_1, …, D_n, C)	*f* es una función de D_1, …, D_n (dominios) a C (codominio) que cumple unicidad	forall x in f ((Id(x,2) in C & (Id(Id(x, 1), 1) in D_1 & … & Id(Id(x,1), n) in D_n)) & forall z1 in f (forall z2 in f (Id(z1, 1) = Id(z2, 1) then Id(z1, 2) = Id(z2, 2))))
inyective(*f*)	La función *f* es inyectiva	forall x1 in f (forall x2 in f (Id(x1, 2) = Id(x2, 2) then Id(x1, 1) = Id(x2, 1)))

Capítulo 5. Aplicación del programa Reconstructor a la teoría objeto

En este capítulo se aplica la metodología de contrastación a través de la herramienta computacional (ambas introducidas en el capítulo anterior) a la reconstrucción de la cladística presentada en el capítulo 3. Con ese fin, el presente capítulo estará estructurado de la siguiente manera. La sección 5.1 muestra cómo cargar los axiomas de **CLAD** en Reconstructor, resaltando los casos en donde fue necesario hacer alguna modificación al axioma formal. En 5.2 se introducen las aplicaciones que se considerarán para la contrastación, primero informalmente y luego formalmente como modelos. Se comienzan a mostrar, además, los resultados de la aplicación de los métodos de contrastación de reconstrucciones introducidos en el capítulo anterior a lo cargado en 5.1 y 5.2 (axiomas y modelos). Luego, en la sección 5.3, se muestra cómo cargar las condiciones de ligadura y cómo testear la adecuación de su formalización con ciertos conjuntos de modelos. La sección 5.4 formaliza y carga al software un método manual para calcular los largos de los árboles filogenéticos basado en el método de optimización de Fitch (capítulo 3, subsección 3.6.2). Por último, en 5.5, se brindan algunas conclusiones.

5.1. Los axiomas

Si bien la sintaxis de Reconstructor está diseñada para ser lo más parecida posible al lenguaje formal que utilizan habitualmente los lógicos contemporáneos y los estructuralistas la carga de axiomas en Reconstructor no es una tarea completamente automática, sino que puede requerir algunos ajustes. Esto se debe a dos motivos principales.

En primer lugar, a que al presentar una reconstrucción formal por escrito se utilizan atajos o convenciones notacionales que, si bien se comprenden fácilmente al leerse, resultan en expresiones que estrictamente hablando no son oraciones bien formadas del lenguaje. Ejemplos de ellos son el uso de los puntos suspensivos (p.e. expresiones de la forma $\exists x_1 \ldots \exists x_n \, \phi$) o la cuantificación sobre subíndices (p.e. expresiones de la forma $\exists x \, d_x(t_1) = t_2$). Es decir, si bien el lenguaje formal utilizado típicamente en las reconstrucciones es mucho más preciso que el lenguaje natural, aun así se suelen utilizar recursos lingüísticos cuya sintaxis y/o semántica no se encuentra

del todo especificada. Reconstructor obliga al usuario a utilizar el lenguaje formal de manera totalmente precisa.

El segundo motivo obedece a cuestiones de computabilidad y a la complejidad de la teoría tomada como objeto de estudio. En la cladística el número de árboles y de asignaciones posibles aumenta de modo explosivo con el aumento del número de taxa y de caracteres, de modo que chequear si un modelo satisface los axiomas puede volverse intensivo cuando los dominios en el modelo se vuelven grandes (lo cual ocurre para números relativamente bajos de terminales y de caracteres). Además, Reconstructor es un generalista, no está optimizado para correr a la cladística particularmente, a diferencia de los programas específicos diseñados por sistemáticos (p.e. Goloboff & Catalano, 2016), sino que su objetivo es contrastar reconstrucciones de cualquier ámbito de la ciencia. Esto me forzará, por un lado, a considerar aplicaciones de la cladística no demasiado complejas (p.e. las aplicaciones que consideraré en este capítulo contienen solo 4 taxa terminales y 2 caracteres). Esto no solo reducirá los tiempos de computación sino que hará más fácil de comprender al proceso de contrastación. Agregar más taxa o más caracteres al análisis no agregaría nada novedoso conceptualmente. Por otro lado, incluso teniendo esto en cuenta, algunos axiomas serán debilitados para que su satisfacción en un modelo sea menos intensiva de calcular.

A continuación, procedo a mostrar cómo cada axioma de la teoría quedó expresado en Reconstructor. Me detendré a explicar adicionalmente la traducción solamente en casos en donde esta no sea trivial —i.e. donde haya sido necesario hacer algún ajuste de los recién mencionados. Los axiomas cargados en Reconstructor pueden visualizarse en el programa, abriendo el archivo suplementario CLAD.theory. Sugiero fuertemente que el lector interesado los vea de este modo, ya que, como se ilustró en el capítulo anterior, el software incluye funcionalidades como el coloreado automático de paréntesis para ayudar con la visualización.

Axioma impropio	**En Reconstructor**
(1) T es un conjunto finito tal que $\|T\| \geq 3$	card(T) >= 3
(2) N es un conjunto finito tal que $T \subset N$	T psubset N
(3) $I = N - T$	I = N complem T

Nótese que en Reconstructor no es necesario expresar que los conjuntos dados son finitos, ya que las interpretaciones de términos primitivos en todo modelo del programa son necesariamente finitas.

El axioma 4 es la definición de cladograma. Recuérdese que en la reconstrucción un cladograma A_i es un grafo, que consta de un conjunto de nodos N y una relación binaria $D(A_i)$ — i.e. un conjunto de pares ordenados de N. Esto será idéntico en Reconstructor, pero por comodidad se considerará a D una función que toma un árbol como argumento y devuelve el conjunto de sus ejes. Esto me evitará tener que agregar al lenguaje un término de relación por cada árbol filogenético. Para constreñir a la función D para que devuelva a los ejes de cada árbol, agregaré el axioma:

(4.0a) forall x in A (D(x) = Id(x, 2))

en donde Id(x, y) denota por default a la función que toma una tupla x y un índice numérico y, y devuelve lo que la tupla contiene en la posición y.

El resto de los puntos de (4) también requieren alguna modificación. Por ejemplo, (4.2) afirma $\nexists n_{x1} \in N, \exists n_{x2}, \ldots, n_{xj} \in N$ tal que $\langle n_{x1}, n_{x2}\rangle \in D(A_i)$ y … y $\langle n_{xj-1}, n_{xj}\rangle \in D(A_i)$ y $\langle n_{xj}, n_{x1}\rangle \in D(A_i)$. Esto expresa que $D(A_i)$ es acíclica, esto es, que no hay un camino que parta de un nodo y termine en el mismo nodo. Pero este axioma contiene con al cuantificador $\exists n_{x2}, \ldots, n_{xj} \in N$ el cual, como se dijo, no está estrictamente bien formado gramaticalmente (los puntos suspensivos son un atajo). Para sortear esta dificultad, se introduce al lenguaje una segunda función D_2, que para un árbol devuelve su relación de ancestría *no inmediata*. Esto es, si $\langle n_x, n_y\rangle \in D(A_i)$ y $\langle n_y, n_z\rangle \in D(A_i)$ entonces $\langle n_x, n_z\rangle \in D_2(A_i)$ (además de los otros dos pares). Así, por ejemplo, el requisito de aciclicidad quedará expresado como $\forall x \in N, \langle x, x\rangle \notin D_2(A_i)$.

La caracterización de D_2 será dada por medio de otro axioma nuevo:

(4.0b) forall x in A (D(x) subset D2(x) & forall y1 in D2(x) (forall y2 in D2(x) (
    if Id(y1, 2) = Id(y2, 1) then tuple(Id(y1, 1), Id(y2, 2)) in D2(x))))

El primer conyunto expresa que todo elemento de $D(A_i)$ (la relación de ancestría inmediata) está presente en $D_2(A_i)$. El segundo conyunto dice que, para dos elementos de $D_2(A_i)$ (i.e. dos pares de

nodos), si el segundo elemento del primer par es igual al primer elemento del segundo par, entonces el par formado por el primer elemento del primer par y el segundo elemento del segundo están en $D_2(A_i)$ (esto permite expresar que si $\langle n_x, n_y\rangle \in D(A_i)$ y $\langle n_y, n_z\rangle \in D(A_i)$ entonces $\langle n_x, n_z\rangle \in D_2(A_i)$ sin un triple cuantificador sobre N, lo cual sería computacionalmente más ineficiente).

Habiendo introducido D y D_2, ya es posible introducir todos los puntos del axioma 4 en Reconstructor, del siguiente modo.

Axioma impropio	**En Reconstructor**
(4.1) $D(A_i) \subseteq N^2$	forall x in A (D(x) subset cartp(N, N))
(4.2) $\nexists n_{x1} \in N, \exists n_{x2}, \ldots, n_{xj} \in N$ tal que $\langle n_{x1}, n_{x2}\rangle \in D(A_i)$ y … y $\langle n_{xj-1}, n_{xj}\rangle \in D(A_i)$ y $\langle n_{xj}, n_{x1}\rangle \in D(A_i)$ [***D*(*A*$_i$) es acíclica**]	forall x in A (forall y in N (¬tuple(y, y) in D2(x))) [expresada por medio de D_2]
(4.3) $\exists! n_x \in N$ tal que $\forall n_y \in N$ (si $n_y \neq n_x$, $\langle n_y, n_x\rangle \notin D(A_i)$ y $\exists n_{x1}, \ldots, n_{xj} \in N$ ($\langle n_x, n_{x1}\rangle \in D(A_i)$ y $\langle n_{x1}, n_{x2}\rangle \in D(A_i)$ y … y $\langle n_{xj}, n_y\rangle \in D(A_i)$)) [***D*(*A*$_i$) es enraizada**]	forall x in A (exists y in I (forall z in N (if ¬y = z then tuple(y, z) in D2(x)))) [expresada por medio de D_2]
(4.4) $\forall n \in I, \exists n_{x1}, n_{x2} \in N$ tal que ($n_{x1} \neq n_{x2}$ y $\langle n, n_{x1}\rangle \in D(A_i)$ y $\langle n_x, n_{x2}\rangle \in D(A_i)$ y $\forall n_{x3} \in N$ (si $n_{x1} \neq n_{x3} \neq n_{x2}$ entonces $\langle n_x, n_{x3}\rangle \notin D(A_i)$)) [**Cada nodo interno tiene exactamente dos descendientes**]	forall x in A (forall y1 in Id(x,2) (!exists y2 in Id(x, 2) (Id(y1, 1) = Id(y2, 1) & ¬y2 = y1))) [Versión computacionalmente más eficiente (aunque un poco más débil). Afirma que para todo par ancestro-descendiente en $D(A_i)$ hay un y solo un par con el mismo primer elemento (ancestro) y distinto segundo elemento (descendiente)]
(4.5) $\forall t \in T$ ($\nexists n \in N$ tal que $\langle t, n\rangle \in D(A_i)$ **[Los taxa bajo consideración son nodos terminales]**	forall x in A (forall y in T (¬exists z in N (tuple(y, z) in D(x))))
(4.6) $\forall n \in N\ \forall n_x \in N\ \forall n_y \in N$ (si $\langle n_x, n\rangle \in D(A_i)$ y $\langle n_y, n\rangle \in D(A_i)$ entonces $n_x = n_y$) **[No ocurren fusiones en el árbol]**	forall x in A (forall y1 in Id(x,2) (forall y2 in Id(x,2) (if Id(y1, 2) = Id(y2, 2) then y1 = y2)))

	[Versión equivalente y computacionalmente más eficiente. Afirma que para toda dupla de pares ancestro-descendiente en $D(A_i)$, si el descendiente es el mismo, entonces el ancestro es el mismo]

Continuando con la carga de axiomas impropios, los siguientes pueden expresarse del siguiente modo:

Axioma impropio	**En Reconstructor**
(5) $A_R \in A$	AR in A
(6) J es el conjunto [jerarquías] de los $x \subseteq \wp(\wp(T))$ tales que: **(6.1)** $T \in x$ **(6.2)** $\forall t \in T\ (\{t\} \in x)$ **(6.3)** $\emptyset \notin x$ **(6.4)** $\forall y_1 \in x\ \forall y_2 \in x\ (y_1 \cap y_2 \in \{y_1, y_2, \emptyset\})$ **(6.5)** $\lvert x\rvert = 2T - 1$ **[Jerarquías]**	forall x in J (((T in x & forall y in T ({y} in x)) & ((¬nullset in x & card(x)=(2*card(T))-1) & forall z1 in x (forall z2 in x (z1 inters z2 in {z1, z2, nullset}))))) [versión más débil pero computacionalmente más eficiente, ya que no requiere cuantificar sobre un doble conjunto potencia]
(7) C (= $\{C_1, \ldots, C_m\}$) es un conjunto finito, no-vacío tal que cada C_i (= $\{c_{i,1}, c_{i,2}, \ldots, c_{i,j}\}$) es un conjunto finito, no vacío. [**caracteres y estados**]	¬C = nullset & forall x in C (card(x) > 0) [el segundo conyunto expresa que los elementos de C son conjuntos — Reconstructor devolverá un error al evaluar este axioma si no lo son]
(8) $\bigcap C = \emptyset$ [**Los caracteres no comparten estados**]	ISet(C) = nullset

A estos axiomas les siguen la caracterización de las funciones que asignan estados a nodos. En los casos en donde un mismo axioma expresaba tanto el dominio y el codominio como la expresión matemática que los relacionaba, se cargaron dos axiomas distintos en Reconstructor, separando ambos componentes. En el caso de δ (= $\{d_1, \ldots, d_n\}$), en lugar de tener un conjunto de funciones binarias, se cargó una única función d triádica, cuyo tercer argumento es el subíndice i que

correspondía a una función d_i particular (así, por ejemplo $d_1(x, y)$ es $d(x, y, 1)$) —nuevamente, para que no sea necesario tener en el lenguaje un símbolo de función por cada asignación.

Axioma impropio	**En Reconstructor**
(9) d_T: $T \times C \rightarrow \bigcup C$, tal que $d_T(t, C_i) \in C_i$ **[asignación para terminales]**	func(dT, T, C, USet(C))
	forall x in T (forall y in C (dT(x, y) in y))
(10) δ (= $\{d_1, \ldots, d_n\}$) es un conjunto finito, no-vacío, tal que $\forall d_i \in N$ (d_i: $N \times C \rightarrow \bigcup C$ y $d_i(n, C_i) \in C_i$) **[funciones de asignación completas]**	forall x in d (Id(x, 2) in USet(C) & ((Id(Id(x,1), 1) in N & Id(Id(x,1), 2) in C) & Id(Id(x,1), 3) in NAT)) (*)
	forall z in setcompr(Id(Id(x1, 1), 3), d) (forall x in N (forall y in C (d(x, y, z) in y))) (**)
(11) $\delta_R \in \delta$	dR in setcompr(Id(Id(x1, 1), 3), d)
(12) $\forall d_i \in \delta$, $\forall t \in T$, $\forall C_i \in C$ ($d_i(t, C_i) = d_T(t, C_i)$)) [**Las asignaciones completas respetan las asignaciones para terminales**]	forall z in setcompr(Id(Id(x1, 1), 3), d) (forall x in T (forall y in C (d(x,y,z) = dT(x,y))))

(*) Este axioma es similar a "func(N, C, NAT, USet(C)))" pero sin el requisito de unicidad. Ello se debe a que incluso en modelos sencillos como el que se introducirá en la sección siguiente (4 taxa y 2 caracteres) *d* es relativamente grande (p.e. su interpretación en ese modelo tiene 896 elementos) y la cuantificación anidada que exige el requisito de unicidad toma mucho tiempo de evaluar.

(**) Otro punto a destacar es que la cota del primer cuantificador devuelve el conjunto de los índices que están como tercer argumento (i.e. el conjunto de los índices de las funciones de asignación). Esto se debe a que en Reconstructor la expresión "setcompr($f(x)$, t)", en donde $f(x)$ es un término con una variable libre y t un término que denota a un conjunto, devuelve el conjunto de los elementos que surgen de reemplazar a la variable libre x por los elementos de t. Por otro lado,

dado que *d* es una función triádica, sus elementos tienen la forma ⟨⟨*x*, *y*, *z*⟩, *w*⟩. Por lo tanto, dado un elemento *e* = ⟨⟨*x*, *y*, *z*⟩, *w*⟩, Id(*e*, 1) = ⟨*x*, *y*, *z*⟩ y por tanto Id(Id(*e*, 1), 3) = Id(⟨*x*, *y*, *z*⟩, 3) = *z*. Ambas cosas implican que la expresión setcompr(Id(Id(x1, 1), 3), d) devolverá al conjunto de las cosas que hay en la tercera posición de los argumentos de *d*, i.e. los índices. Esta cota se repetirá en otros axiomas (p.e. en (11) y (12)).

A continuación siguen las funciones de costos, pesos y largos de estados de caracteres y cladogramas. En los casos en donde las funciones tomaban una función de asignación como argumento (p.e. *w*: $N^2 \times \delta \rightarrow \mathbb{N}$), tal argumento fue sustituido por el índice que representa a esa función en la nueva función triádica *d*.

Axioma impropio	**En Reconstructor**
(13) $\$: \cup C \times \cup C \rightarrow \mathbb{N}$ [**función parcial de costos**]	func(cost, USet(C), USet(C), NAT)
(14) $w: N^2 \times \delta \rightarrow \mathbb{N}$, tal que $w(\langle n_j, n_k \rangle, d_x) = \sum_{Ci \in C} \$(d_x(n_j, C_i), d_x(n_k, C_i))$ [**peso de un eje**]	forall x in w (Id(x, 2) in NAT & (Id(Id(x,1), 1) in cartp(N, N) & Id(Id(x,1), 2) in NAT)) [ídem a 10.1, equivale a func sin unicidad, por motivos computacionales]
	forall x1 in I (forall x2 in N (forall z in setcompr(Id(Id(y, 1), 3), d) (w(tuple(x1, x2), z) = SUM(i1, C, cost(d(x1,i1,z), d(x2,i1,z))))))
(15) $l_1: A \times \delta \rightarrow \mathbb{N}$, tal que $l_1(A_i, d_m) = \sum_{\langle x,y \rangle \in D(A_i)} w(\langle x, y \rangle, d_m)$ [**largo de un cladograma bajo una asignación**]	forall x in l1 (Id(x, 2) in NAT & (Id(Id(x,1), 1) in A & Id(Id(x,1), 2) in NAT)) [ídem a 10.1, equivale a func sin unicidad, por motivos computacionales]
	forall x in A (forall z in setcompr(Id(Id(x1, 1), 3), d) (l1(x,z) = SUM(i1, D(x), w(i1,z))))
(16) $l_2: A \rightarrow \mathbb{N}$ tal que $l_2(A_i) = min(\{l_1(A_i, d_j) / d_j \in \delta\})$ [**largo de un cladograma**]	func(l2, A, NAT)
	forall x in A (l2(x) = min(setcompr(l1(x,y), setcompr(Id(Id(x1, 1), 3), d)))) (*)

(*) En (16.2) la segunda comprensión es igual que la anterior, devolviendo el conjunto IND de los índices presentes en *d*. De ese modo, (16.2) afirma que $l_2(A_i) = \min(\text{setcompr}(l_1(A_i, y), \text{IND}))$ para todo árbol A_i. La comprensión que figura en esta expresión devuelve al conjunto de los largos$_1$ bajo todas la asignaciones posibles.

Los axiomas impropios restantes definen al conjunto de los árboles, asignaciones y jerarquías óptimas, y definen a la función para pasar de un árbol (*qua* grafo) a una jerarquía.

Axioma impropio	**En Reconstructor**
(**17**) $AO = \{A_i / l_2(A_i) \in min(\{l_2(A_k) / A_k \in A\})\}$ [**árboles óptimos**]	forall x in A (x in AO iff forall y in A (l2(x) <= l2(y)))
(18) δ_O: $A \rightarrow \delta$, tal que $\delta_O(A_i) = \{d_j \in \delta / l_1(A_i, d_j) = l_2(A_i)\}$ [**asignaciones óptimas para un árbol**]	forall x in A (forall y in setcompr(Id(Id(x1, 1), 3), d) (y in dO(x) iff l1(x,y) = l2(x)))
(19) j: $A \rightarrow J$, tal que $j(A_i) = \{X \in \wp(T) / \lvert X\rvert = 1$ o $\exists n \in N\ (\forall x \in X, \exists p \in P\ (p = \langle n, \ldots, x\rangle)$ y $\forall y \in T - X, \neg\exists p \in P\ (p = \langle n, \ldots, y\rangle))\}$ [**jerarquía correspondiente a un árbol**]	func(j, A, J) forall x in A (forall y in pset(T) complem {{}} ((card(y) = 1 or exists z in N (forall y1 in y (tuple(z, y1) in D2(x)) & forall y2 in T complem y (¬tuple(z, y2) in D2(x)))) then y in j(x))) (*)
(20) $AOj = \{J_i \in J / \exists A_k \in$ AO $(j(A_k) = J_i)\}$ [**jerarquías óptimas**]	forall x in A (if x in AO then j(x) in AOj) (*)

(*) Las traducciones de los axiomas 19.2 y 20 son más débiles que los originales (son condicionales en lugar de bicondicionales), también por motivos computacionales, pero diferentes a los anteriores: es para que puedan ser usados como métodos de determinación automáticos (véase la sección siguiente).

Por último, las leyes fundamentales son sencillas de cargar en Reconstructor:

Ley fundamental	En Reconstructor
(2) $A_R \in AO$	AR in AO
(3) $d_R \in \delta_O(A_R)$	dR in dO(AR)

5.2. Los modelos y (algunos) métodos de determinación

En esta sección se describen primero (informalmente) una serie de aplicaciones (potenciales, algunas exitosas y otras no) de la cladística, así como su representación formal como modelos. Luego, se exponen los resultados del primer método de contrastación de reconstrucciones aplicado a estos modelos y a los axiomas introducidos en el programa en la sección anterior.

5.2.1. Las aplicaciones a considerar

En esta subsección se describen las aplicaciones (informales) que luego serán cargadas como modelos en el programa en la subsección siguiente. Como se dijo, las aplicaciones serán sencillas (i.e. pocos taxa y caracteres), por dos motivos. En primer lugar, porque considerar aplicaciones más complejas no agregaría nada conceptualmente interesante. En segundo, porque el número de árboles, asignaciones, etc. crece de manera muy rápida con el aumento en el número de taxa y de caracteres, y se vuelve computacionalmente intensivo chequear la satisfacción de los axiomas en un modelo. A modo ilustrativo, considérese que en los modelos que representarán a las aplicaciones introducidas en esta sección (que contienen solo 4 taxa y 2 caracteres), las cardinalidades de *d*, *w* y l_1 son 896, 1344 y 960, respectivamente —y que en ocasiones hay que cuantificar, a veces de manera anidada, sobre ellos.

Tendremos entonces la siguiente matriz de datos:

Taxa / Carácter	C_1	C_2
t_1	0	0
t_2	0	0
t_3	0	1
t_4	1	1

Tabla 5.7. Matriz de datos a considerar

Para estos 4 taxa habrá por tanto 15 árboles posibles (véase el capítulo 2, figura 2.5) y un único árbol no enraizado óptimo, de largo 2, que se ve como sigue:

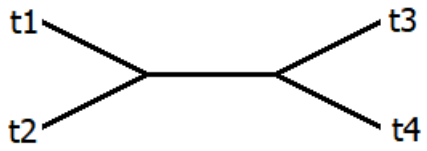


Fig. 5.37. Árbol no enraizado óptimo para la matriz de datos de la tabla 5.1.

Esto significa que habrá 5 de los 15 árboles que poseen el largo mínimo de 2 (que corresponden a los 5 ejes en los que se puede enraizar el árbol de la figura 5.1). En notación parentética, ellos son:

a) $((t_1, t_2), (t_3, t_4))$
b) $(t_1, (t_2, (t_3, t_4)))$
c) $(t_2, (t_1, (t_3, t_4)))$
d) $(t_3, (t_4, (t_1, t_2)))$
e) $(t_4, (t_3, (t_1, t_2)))$

Consideraré una primera aplicación que no posee al árbol a) en el conjunto de los óptimos, la cual debería, por tanto, no satisfacer a los axiomas impropios. Es decir, su representación como modelo contendrá al número y tipo de conceptos adecuado, pero no satisfará al axioma impropio (**17**) —la definición del conjunto *AO* como el de los árboles de menor largo—, no siendo por tanto un modelo

potencial de la teoría.[39] Una segunda aplicación satisfará todos los axiomas impropios (tendrá a los 5 árboles anteriores, con largo 2, como los óptimos) pero no a la ley fundamental (2) —es decir, será un modelo potencial pero no un modelo actual. Para ello, supondré que el árbol real no es ninguno de esos cinco, sino el árbol ((t_1, t_3), (t_2, t_4)). Es decir, supondré que la filogenia real no coincide con ninguno de los árboles óptimos que arroja la cladística, y por tanto este caso debería ser un falsador (o un *puzzle*) de la teoría, al no satisfacer (2). Una tercera aplicación satisfará la ley (2) pero no la ley (3). Es decir, la filogenia real estará entre las óptimas según la cladística, pero la evolución real de los caracteres será distinta a lo que la cladística retrodice (i.e. a las asignaciones de estados a los nodos internos para el árbol óptimo). Para ello, tendrá al árbol a) como el real, pero la asignación real asignará 0 a los tres nodos internos para ambos caracteres. Así, por ejemplo, el ancestro del grupo (t_3, t_4) poseerá el estado 0 para el carácter C_2, habiendo ocurrido una convergencia en la descendencia, al cambiar ambos independientemente a 1. Por último, una cuarta aplicación satisfará ambas leyes fundamentales: tendrá al árbol a) como el real y a la asignación (t_1, t_2, t_3, t_4): 0, (t_1, t_2): 0, (t_3, t_4): 1 como la real (para el carácter C_2). Resumiendo:

Aplicación	**Axiomas impropios**	**Ley (2)**	**Ley (3)**
a_1	No	-	-
a_2	Sí	No	-
a_3	Sí	Sí	No
a_4	Sí	Sí	Sí

Tabla 8. Cuadro que resume lo que debería satisfacer y no satisfacer cada una de las aplicaciones introducidas en el párrafo anterior. Las celdas con - significan solo que no me preocupará el valor de ese axioma en el modelo correspondiente (de hecho quedarán indeterminadas, véase la sección siguiente).

5.2.2. Carga de modelos

En esta subsección y la siguiente se muestra como cargar las aplicaciones a_1-a_4 de la subsección anterior como cuatro modelos m_1-m_4. Como se dijo arriba, incluso en este caso simple, algunos

[39] Testear la adecuación de los axiomas impropios presupone cargar estructuras que *no* los satisfacen. Puede sonar extraño a un estructuralista llamar aplicación a algo que no es un modelo potencial de la teoría. Si el lector prefiriese otro término no tendría problemas con ello.

conceptos poseen extensiones relativamente grandes. Por ello, sería sumamente engorroso cargar por extensión (i.e. a mano) las denotaciones de todos los conceptos. Por ello, mostraré a continuación cómo cargar paso a paso un modelo de **CLAD** introduciendo unos pocos conceptos a mano y luego usando una combinación de métodos manuales y automáticos para llenar las denotaciones del resto. Dado que utilizaré métodos automáticos para cargar algunas denotaciones, trabajaré desde el módulo de modelos incompletos (en este caso ello no hace diferencia porque el lenguaje primitivo **CLAD** no contiene relaciones). Esta dinámica de carga de modelos es específica para **CLAD**, aunque puede funcionar como ilustración del modo en que se puede trabajar con otras teorías desde Reconstructor.

Comenzaré con un modelo incompleto i_{00}, en el cual se cargarán a mano los conceptos T, C, dT, cost, N e I. Nótese que los primeros tres representan a la matriz de datos y el cuarto a las matrices de costos. Estos son los conceptos que no pueden determinarse aplicando las leyes de la teoría, sino que cualquier aplicación los presupone (véase el capítulo 3). Los últimos dos representan a los nodos (totales e internos) de los cladogramas —podrían cargarse usando métodos, pero ya que las aplicaciones consideradas son simples y los nodos son pocos, los cargaré a mano por comodidad. Así, el modelo i_{00} se ve del siguiente modo:

$C_{i00} = \{\{c_{00}, c_{01}\}, \{c_{10}, c_{11}\}\}$

$T_{i00} = \{t_1, t_2, t_3, t_4\}$

$N_{i00} = \{t_1, t_2, t_3, t_4, n_1, n_2, n_3\}$

$I_{i00} = \{n_1, n_2, n_3\}$

$d_{Ti00} = \{\langle\langle t_1, \{c_{00}, c_{01}\}\rangle, c_{00}\rangle, \langle\langle t_1, \{c_{10}, c_{11}\}\rangle, c_{10}\rangle, \langle\langle t_2, \{c_{00}, c_{01}\}\rangle, c_{00}\rangle, \langle\langle t_2, \{c_{10}, c_{11}\}\rangle, c_{10}\rangle,$
$\langle\langle t_3, \{c_{00}, c_{01}\}\rangle, c_{00}\rangle, \langle\langle t_3, \{c_{10}, c_{11}\}\rangle, c_{11}\rangle, \langle\langle t_4, \{c_{00}, c_{01}\}\rangle, c_{01}\rangle, \langle\langle t_4, \{c_{10}, c_{11}\}\rangle, c_{11}\rangle\}$

$cost_{i00} = \{\langle\langle c_{00}, c_{00}\rangle, 0\rangle, \langle\langle c_{00}, c_{01}\rangle, 1\rangle, \langle\langle c_{01}, c_{00}\rangle, 1\rangle, \langle\langle c_{01}, c_{01}\rangle, 0\rangle, \langle\langle c_{10}, c_{10}\rangle, 0\rangle, \langle\langle c_{10}, c_{11}\rangle,$
$1\rangle, \langle\langle c_{11}, c_{10}\rangle, 1\rangle, \langle\langle c_{11}, c_{11}\rangle, 0\rangle\}$

El resto de los conceptos se encuentran indeterminados en i_{00}, y los iré cargando secuencialmente aplicando métodos. Para construir el conjunto de los árboles filogenéticos, lo más sencillo es construir primero las jerarquías y luego construir los árboles a partir de las jerarquías. Para ello, se introdujeron dos métodos manuales, *build_J* y *build_A* en el archivo suplementario. No reproduciré aquí el código para estos métodos (que es propio y puede verse en el tal archivo), basta con decir

que, una vez ejecutados sobre i_{00} se obtienen las jerarquías y los árboles (están cargados en el archivo suplementario en un modelo i_{01}):

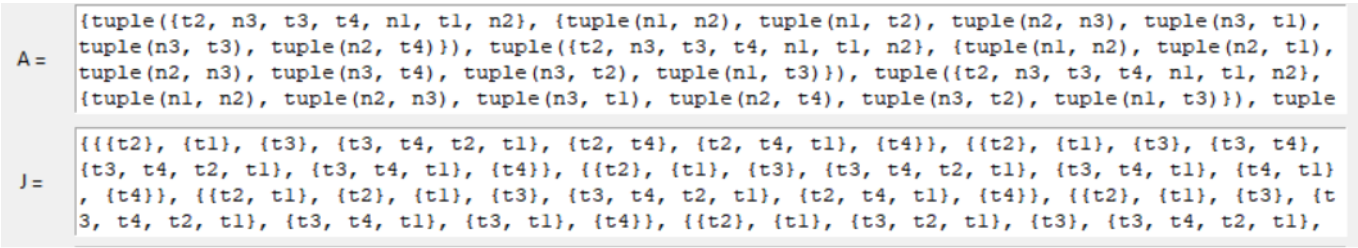


Fig. 5.38. Árboles filogenéticos cargados en el modelo i_{01} de Reconstructor, tras la ejecución de los métodos de determinación manuales *build_J* y *build_A*.

Los siguientes dos conceptos a cargar son D y D_2 (que en el archivo suplementario son dos funciones primitivas, las de descendencia inmediata y no inmediata según un árbol). D puede cargarse simplemente ejecutando el axioma (4.0a) como método automático. El caso de D_2 es un poco más complejo. El axioma (4.0b) puede usarse como método automático para determinarla. Sin embargo esto presupone un paso previo. Ese axioma tiene como consecuente del condicional a la afirmación "tuple(Id(y1, 1), Id(y2, 2)) in D2(x)". Pero esto solo agregará la tupla correspondiente a $D_2(x)$ si $D_2(x)$ ya es un conjunto (recuérdese que la cláusula (e_{18}) de la función de imposición decía que "$e(t_1 \in t_2, m)$ implica $t_{1m} \in t_{2m+1}$, **si t_{2m+1} es un conjunto**"; véase la sección 4.6.5. del capítulo 4). Por ese motivo, un primer método manual *begin_D2* hará que para todo árbol A_i, $D_2(A_i) = \{\}$, para luego poder correr (4.0b) como método automático.[40] (4.0b) tiene otra característica interesante. El axioma completo afirma que:

> $\forall x$ in A ($D(x)$ subset $D_2(x)$ & $\forall y_1$ in $D_2(x)$, $\forall y_2$ in $D_2(x)$ (si Id(y_1, 2) = Id(y_2, 1) entonces $\langle$Id(y_1, 1), Id(y_1, 2)$\rangle$ in $D_2(x)$))

Si se ejecuta este método de determinación una vez, dado que $D_2(x)$ es el conjunto vacío para todo $x \in A$, ello hará que (por el primer conyunto y la cláusula (e_{20}) —para subconjuntos— de la función de imposición) todo elemento de $D(x)$ esté en $D_2(x)$. Y dado que en el modelo inicial $D_2(x)$ es vacía,

[40] Dado que el método automático modificará el valor de $D_2(A_i)$, el programa reconocerá a la aplicación de (4.0b) como método automático como una extensión no monótona. Esto, sin embargo es comportamiento esperado y no representa una falla del método de determinación en este caso.

los siguientes dos cuantificadores recorrerán el conjunto vacío, y por tanto, no agregarán nada a $D_2(x)$. Pero si se vuelve a correr (4.0b) sobre el resultado la parte de transitividad comenzará a agregar nuevas denotaciones. Por ejemplo, si $D(x)$ posee las tuplas $\langle n_1, n_2\rangle$, $\langle n_2, n_3\rangle$, y $\langle n_3, t_1\rangle$ el condicional en el axioma agregará $\langle n_1, n_3\rangle$ y $\langle n_2, t_1\rangle$ a $D(x)$ —pero no, $\langle n_1, t_1\rangle$ aún. Para esto último habría que volver a correr (4.0b) sobre el resultado anterior. Por estos motivos, (4.0b) debe ejecutarse chequeando la opción **Iterate** en Reconstructor, para que encuentre la denotación completa en una única ejecución. El resultado de ejecutar estos tres métodos está guardado en el archivo suplementario como un modelo i_{02}.

A continuación, para construir *j* (la función de correspondencia entre árboles y jerarquías) se utiliza la misma estrategia. El axioma (19.2) —el que se usará como método automático— posee como consecuente del condicional la expresión "*y* in $j(x)$" (donde *y* es un conjunto de taxa terminales y *x* un árbol), con lo cual es necesario que $j(A_i)$ sea un conjunto para que el método automático le agregue elementos. Un método manual *begin_j* hace que $j(A_i) = \{\}$ para todo árbol A_i. El resultado de aplicar estos dos métodos está guardado como un modelo i_{03}.

El siguiente concepto a cargar es *d*, el conjunto de todas las asignaciones posibles de estados a nodos, terminales e internos. Los elementos de esta función surgen de una combinatoria matemática sobre los caracteres y los nodos. Para cargarla, se introdujo un método de determinación manual, *build_d*. Nuevamente, el código es propio y no se lo reproducirá aquí (ya que no tiene mayor interés filosófico/conceptual), pero puede verse en el archivo suplementario. El resultado de ejecutarlo está guardado como un modelo incompleto i_{04}.

Ya cargados los árboles, las asignaciones y la función de costos, es posible ahora determinar los pesos y largos de los cladogramas (funciones *w*, l_1 y l_2). Ello puede hacerse fácilmente ejecutando los axiomas (14.2), (15.2) y (16.2) como métodos automáticos. El resultado se encuentra guardado como un modelo i_{05}.

Restan unos pocos conceptos. Los siguientes son *AO*, *AOj* y d_O (los árboles y jerarquías óptimas, y la función que devuelve el conjunto de optimizaciones óptimas para un árbol). Para *AO*, el axioma relevante es (17) $\forall x \in A\ (x \in AO$ sii $\forall y \in A\ (l_2(x) \leq l_2(y)))$. Nuevamente, dado que una parte del axioma es $x \in AO$, *AO* debe ser un conjunto para que el método automático correspondiente le agregue algún valor (algo semejante ocurre con el axioma (20) y AO_j). Por tanto, para usar (17) y (20) como métodos automáticos se deben cargar inicialmente $AO = \{\}$ y $AO_j = \{\}$. Con d_O y el axioma (18) ocurre algo similar, pero dado que d_O es una función y no una constante,

se emplea la estrategia anterior de usar un método manual para agregar al conjunto vacío como valor de todo argumento, para luego usar el método automático correspondiente. El resultado de agregar las denotaciones de estos tres conceptos está cargado en i_{06}.

Puede observarse que la denotación de *AO* tras aplicar estos métodos es:

$\{\langle\{n_3, t_4, t_2, t_1, t_3, n_2, n_1\}, \{\langle n_2, n_3\rangle, \langle n_3, t_1\rangle, \langle n_1, t_4\rangle, \langle n_2, t_3\rangle, \langle n_3, t_2\rangle, \langle n_1, n_2\rangle\}\rangle,$
$\langle\{n_3, t_4, t_2, t_1, t_3, n_2, n_1\}, \{\langle n_2, n_3\rangle, \langle n_3, t_4\rangle, \langle n_2, t_1\rangle, \langle n_3, t_3\rangle, \langle n_1, n_2\rangle, \langle n_1, t_2\rangle\}\rangle,$
$\langle\{n_3, t_4, t_2, t_1, t_3, n_2, n_1\}, \{\langle n_3, t_1\rangle, \langle n_2, t_3\rangle, \langle n_1, n_3\rangle, \langle n_3, t_2\rangle, \langle n_1, n_2\rangle, \langle n_2, t_4\rangle\}\rangle,$
$\langle\{n_3, t_4, t_2, t_1, t_3, n_2, n_1\}, \{\langle n_2, n_3\rangle, \langle n_3, t_4\rangle, \langle n_2, t_2\rangle, \langle n_3, t_3\rangle, \langle n_1, n_2\rangle, \langle n_1, t_1\rangle\}\rangle,$
$\langle\{n_3, t_4, t_2, t_1, t_3, n_2, n_1\}, \{\langle n_2, n_3\rangle, \langle n_3, t_1\rangle, \langle n_3, t_2\rangle, \langle n_1, t_3\rangle, \langle n_1, n_2\rangle, \langle n_2, t_4\rangle\}\rangle\}$

Estos 5 árboles corresponden a los árboles e), c), a), b) y d) de la subsección anterior, respectivamente. Esto ya brinda un primer indicio de que los métodos manuales y automáticos cargados y ejecutados hasta el momento son adecuados.

5.2.3. Resultados de la contrastación de **CLAD**, parte I

El modelo m_1, que representa a la aplicación a_1 (la que no posee el árbol a) y por lo tanto no debe satisfacer los axiomas impropios) puede ya cargarse como modelo a partir de i_{06}, simplemente quitando ese árbol de *AO*. Si se evalúa la satisfacción de los axiomas impropios en m_1 se obtiene el siguiente resultado:

```
-----------------------------------------
QUERY: Model: m1, Satisfies improper axioms?

        RESULT:
                (01): True
                (02): True
                (03): True
                (04.0a): True
                (04.0b): True
                (04.1): True
                (04.2): True
                (04.3): True
                (04.4): True
                (04.5): True
                (04.6): True
                (05): Indeterminate
                (06): True
                (07): True
                (08): True
                (09.1): True
                (09.2): True
                (10.1): True
                (10.2): True
                (11): Indeterminate
                (12): True
                (13): True
                (14.1): True
                (14.2): True
                (15.1): True
                (15.2): True
                (16.1): True
                (16.2): True
                (17): False
                (18): True
                (19.1): True
                (19.2): True
                (20): True

        GLOBAL RESULT: False
        ELAPSED TIME: 18.41 seconds
```

Fig. 5.39. Resultados de la contrastación de **CLAD** con m_1.

Puede verse que el resultado global es falso, lo cual es lo que se deseaba, pero además que el axioma particular que falló es (17), la definición de *AO*. Los dos axiomas con resultado indeterminado son las caracterizaciones de *AR* y d_R (el árbol y la asignación reales) cuya denotación no se cargó aun en los modelos.

La carga de estos dos conceptos (sobre la base de i_{06}) permitirá cargar los modelos m_2-m_4. Para m_2, el modelo que no satisface la ley fundamental (2) al tener como árbol real a ((t_1, t_3), (t_2, t_4)), basta con cargar (la traducción a grafo de) este árbol como la denotación de *AR*. El resultado es el que se esperaba: el modelo satisface ahora los axiomas impropios, y devuelve el siguiente resultado para la consulta sobre la satisfacción de las leyes fundamentales:

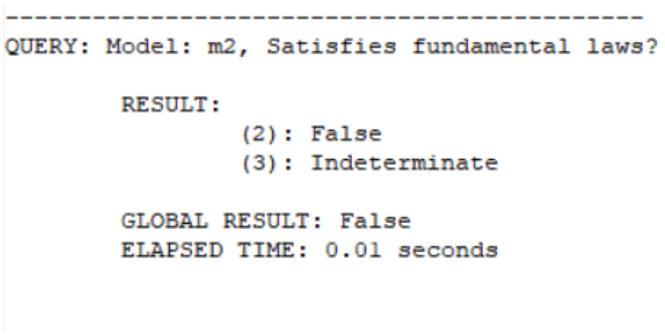


Fig. 5.40 Resultados de la contrastación de **CLAD** con m_2.

El modelo m_3 puede cargarse de un modo semejante, eligiendo al árbol a) como el real (A_R), y a la asignación que elige c00 y c10 para todo nodo interno como la real (d_R) —que resulta ser la que lleva el índice 43.

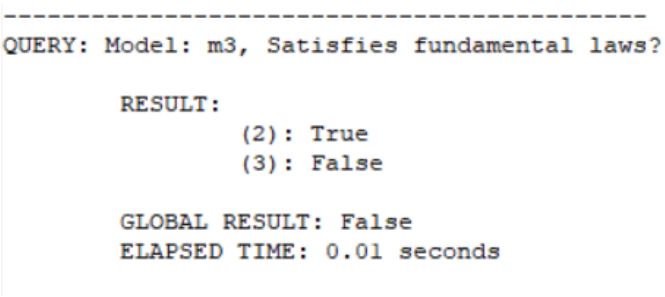


Fig. 5.41 Resultados de la contrastación de **CLAD** con m_3.

Por último, el modelo m_4 puede cargarse modificando d_R en m_3 para que posea una de las asignaciones óptimas según la cladística para el árbol a). Existen dos asignaciones óptimas para este árbol, que son las siguientes:

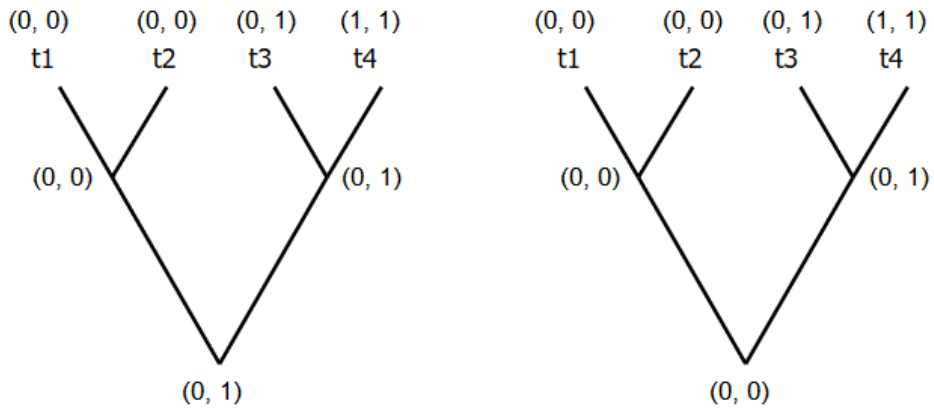


Fig. 5.42 Asignaciones óptimas para el árbol a).

Ellas corresponden con las asignaciones número 33 y 35 del modelo i_{06}, respectivamente. Cargamos entonces d_{Om4} = 33, y ocurre lo siguiente:

```
-------------------------------------------
QUERY: Model: m4, Satisfies fundamental laws?

        RESULT:
                (2): True
                (3): True

        GLOBAL RESULT: True
        ELAPSED TIME: 0.01 seconds
```

Fig. 5.43 Resultados de la contrastación de **CLAD** con m_4.

Lo cual es exactamente lo que se deseaba. En ese sentido, se desprende de los resultados obtenidos en esta sección que la reconstrucción de la cladística introducida en el capítulo 3 pasó exitosamente una primera contrastación, que corresponde al primer método de contrastación de teorías introducido en el capítulo 4.

Adicionalmente, puede verse que en m_4 las denotaciones de los conceptos son las esperadas. Por ejemplo, como se dijo, *AO* corresponde con lo que informalmente se consideró que eran los árboles óptimos. Por otro lado, consultas como las siguientes también dan los resultados esperados:

```
-----------------------------------------
QUERY: Model: m4, Denotation I2(AR)?

        RESULT: 2
        ELAPSED TIME: 0.0 seconds

-----------------------------------------
QUERY: Model: m4, Denotation dO(AR)?

        RESULT: {33, 35}
        ELAPSED TIME: 0.0 seconds

-----------------------------------------
QUERY: Sentence: j(AR) in AOj, Satisfied by m4?

        RESULT: True
        ELAPSED TIME: 0.0 seconds

-----------------------------------------
QUERY: Sentence: ISet(AOj) subset j(AR), Satisfied by m4?

        RESULT: True
        ELAPSED TIME: 0.0 seconds
```

Fig. 5.44 Resultados de otras consultas en m_4.

Esto abona a que los métodos manuales utilizados en la subsección anterior para la carga de los modelos fueron correctamente reconstruidos (el segundo método de contrastación de reconstrucciones del capítulo 4; tendré más que decir sobre este en la sección 5.4), ya que las denotaciones de los conceptos en el modelo (el *output* de esos métodos) coincide con su contraparte informal.

5.3. Las condiciones de ligadura

En esta sección se muestra cómo cargar las condiciones de ligadura de **CLAD** y cómo utilizar el primer método de testeo de reconstrucciones con ellas. A diferencia de lo que ocurría en la segunda condición de ligadura de MCAR (capitulo 5, sección 4.3.2), las condiciones de ligadura de **CLAD** no presuponen lenguaje metateórico nuevo. De ese modo, pueden cargarse en Reconstructor del modo siguiente.

Condición de ligadura	En Reconstructor

(**C1**) $X \in \mathbf{C_1}$ si y solo si $\emptyset \neq X \subseteq \mathbf{M_p(CLAD)}$ y para toda $x, y \in X$, y todos t_1, t_2 y t_3: si $\{t_1, t_2, t_3\} \subseteq T_x \cap T_y$ y existe un Z tal que ($\{t_1, t_2\} \subseteq$ Z y $t_3 \notin$ Z y Z $\in j(A_R)_x$) entonces, existe un W tal que ($\{t_1, t_2\} \subseteq$ W y $t_3 \notin$ W y W $\in j(A_R)_y$) [**La cercanía filogenética se preserva**]	forall x1 in modelset (forall x2 in modelset (forall y1 in denotation(T, x1) (forall y2 in denotation(T, x1) (forall y3 in denotation(T, x1) (if {y1, y2, y3} subset denotation(T, x2) then (if exists z1 in denotation(j(AR), x1) ({y1, y2} subset z1 & ¬y3 in z1) then exists z2 in denotation(j(AR), x2) ({y1, y2} subset z2 & ¬y3 in z2)))))))
(**C2**) $X \in \mathbf{C_2}$ si y solo si $\emptyset \neq X \subseteq \mathbf{M_p(CLAD)}$ y para toda $x, y \in X$, todo t y todo c: si $t \subseteq T_x \cap T_y$ y $c \subseteq C_x \cap C_y$ entonces $d_T(t, c)_x = d_T(t, c)_y$ [**Las asignaciones de estados a terminales deben ser idénticas**]	forall x1 in modelset (forall x2 in modelset (forall y in denotation(T, x1) (forall z in denotation(C, x1) (if (y in denotation(T, x2) & z in denotation(C, x2)) then denotation(dT(y, z), x1) = denotation(dT(y, z), x2)))))

Para contrastar la adecuación de estos dos axiomas, se introducen cuatro modelos nuevos (n_1-n_4), que representan a cuatro aplicaciones tales que:

- n_1 y n_2 contienen 4 taxa cada uno (t_1, t_2, t_3 y t_5), los tres primeros estando presentes en los análisis de la sección anterior (particularmente, en m_4). n_1 y m_4 deberían satisfacer **C₁**: en m_4 el árbol real es ((t_1, t_2), (t_3, t_4)), mientras que en n_1 es (t_3, (t_5, (t_1, t_2))), un árbol compatible en cuanto a las relaciones de t_1, t_2 y t_3, los terminales compartidos. Por otro lado, n_2 y m_4 no deberían satisfacer **C₁**: el árbol real según n_2 es (t_1, (t_5, (t_2, t_3))), el cual es incompatible respecto a las relaciones de t_1, t_2 y t_3.
- n_3 y n_4 contienen 3 taxa cada uno (t_1, t_2 y t_3), y un carácter (C_0). En n_3, la asignación d_T para los terminales y el carácter C_0 coincide con las de m_4 (i.e. {n_3, m_4} debería satisfacer la condición de ligadura **C₂**). En cambio, n_4 asigna $d_T(t_1, \{c_{00}, c_{01}\}) = c_{01}$ (cuando m_4 asignaba c_{00}). Por lo tanto, {n_4, m_4} debería no satisfacer la condición de ligadura **C₂**.

Una vez cargado todo esto en Reconstructor, puede verificarse que los resultados son los esperados en todos los casos:

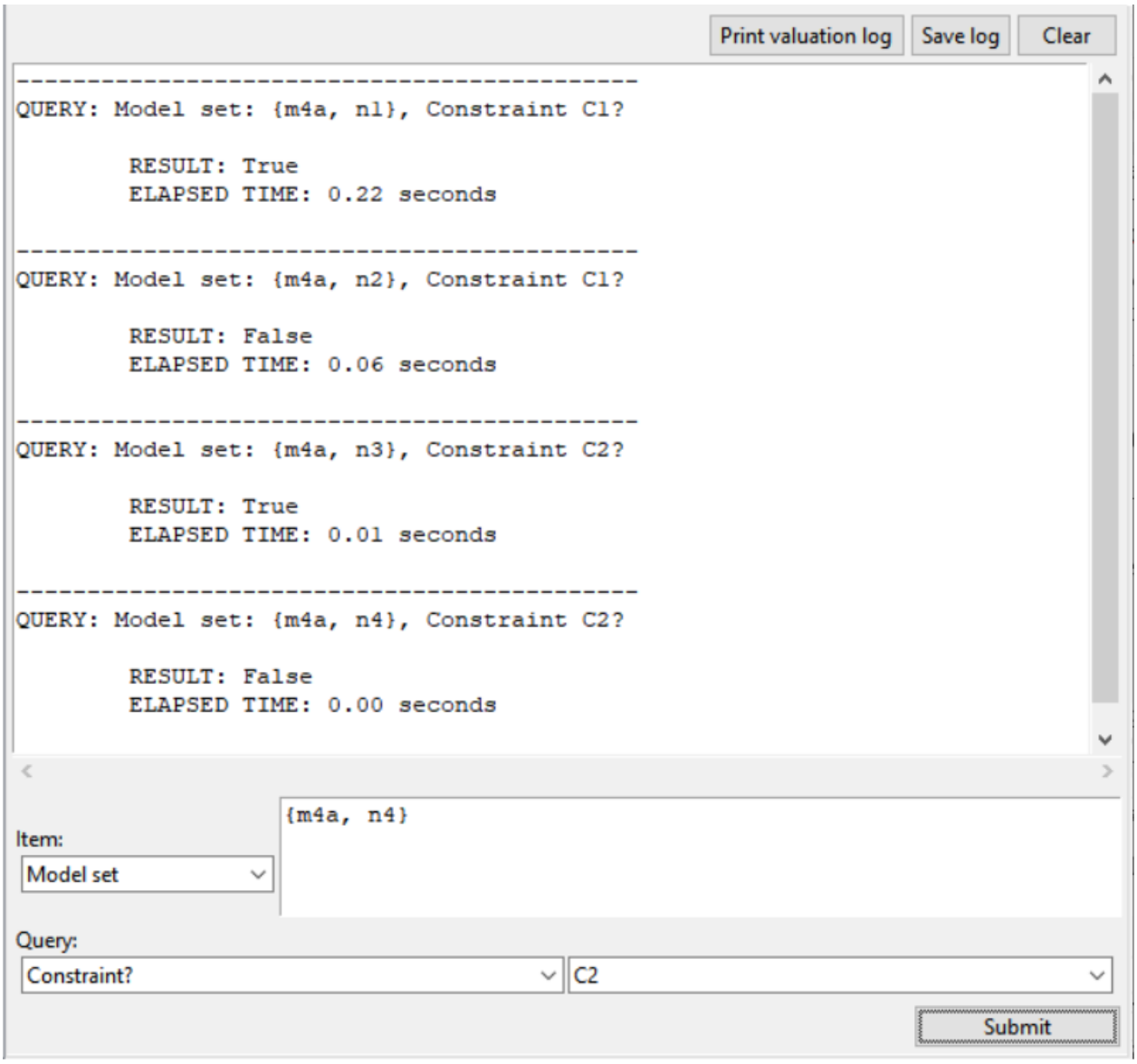


Fig. 5.45 Resultados de la contrastación de **CLAD** con las condiciones de ligadura $\mathbf{C_1}$ y $\mathbf{C_2}$ (el modelo m_{4a} es idéntico a m_4; está copiado al módulo de modelos completos, ya que las condiciones de ligadura funcionan solo con modelos completos)

5.4. Otros métodos de determinación manuales

En esta sección se muestra cómo cargar un método de determinación manual, que utiliza el algoritmo de optimización de Fitch (capítulo 3, sección 3.6.2) para hacer búsquedas de largos de

árboles sin la necesidad de calcular el largo de un árbol bajo toda asignación posible de estados a los nodos internos. Se mostrará que esto tiene la ventaja de ser computacionalmente mucho más eficiente, razón por la cual tiene una importancia mayúscula en la práctica de los sistemáticos.

La idea será diseñar un método manual para el concepto l_2 que tome cada árbol, y la asignación d_T como input, y le asigne un largo haciendo solo la pasada hacia abajo del algoritmo de Fitch (1971). Formalmente (ahora en el lenguaje de Python) esto queda representado como sigue:

```
def l2_Fitch(A, C, T, I, dT, models, incmodels):
    from copy import deepcopy
    l2 = set()
    starting_optimizations = {x: {} for x in C}
    for x in dT:
        starting_optimizations[x[0][1]][x[0][0]] = frozenset({x[1]})
    # Hasta ahora tenemos starting = {C1: {t1: {0}, t2: {1}, ...}, C2 = {...}}

    for a in A:
        visited_nodes = set(T)
        unvisited_nodes = set(I)
        optimizations = deepcopy(starting_optimizations)
        tree_length = 0

        while unvisited_nodes:
            for node in unvisited_nodes:
                descendants = [x[1] for x in a[1] if x[0] == node]
                if set(descendants) < visited_nodes:
                    for char in C:
                        descendant1_state = optimizations[char][descendants[0]]
                        descendant2_state = optimizations[char][descendants[1]]

                        if descendant1_state & descendant2_state:
                            optimizations[char][node] = descendant1_state &
                                                        descendant2_state
                        else:
                            optimizations[char][node] = descendant1_state |
```

```
                                                        descendant2_state
                        tree_length += 1

                    visited_nodes.add(node)
                    unvisited_nodes.remove(node)
                    break
        l2.add(((a,), tree_length))

    return l2
```

Para contrastar la codificación de este método de determinación pueden usarse tanto el primero como el segundo métodos de contrastación de reconstrucciones introducidos en el capítulo anterior. En el primero, se chequea la coincidencia entre la aplicación de una ley / método formalizados y su contraparte informal. El segundo es un test por coherencia, y consiste en ver si el resultado de aplicar los métodos manuales a un modelo satisface los axiomas (formalizados) de la teoría. Dado que sabemos que el modelo m_4 satisface todos los axiomas (propios e impropios) y que la denotación de l_2 coincide con su contraparte informal, ambos tests se pueden realizar conjuntamente, aplicando el método manual a m_4 y viendo si la nueva denotación de l_2 coincide con la anterior. Esto quedó hecho en el modelo m_{4b} (en los modelos completos) en el archivo suplementario, nuevamente con un resultado positivo.

Adicionalmente, puede chequearse que este método manual toma significativamente menos tiempo para encontrar los largos$_2$ de todos los árboles filogenéticos que el procedimiento delineado en 5.2. Lo primero le toma a Reconstructor aproximadamente 0,7 segundos (para encontrar l_2 a partir de A, C, T, I y d_T), mientras que lo segundo (determinar d, w, l_1 y l_2 como se indicó en la sección anterior) toma aproximadamente unos 9,48 segundos (0,62 + 3,43 + 3,32 + 2,11 para cada concepto, respectivamente, en la corrida de prueba que se realizó) —es decir, es más de 10 veces más rápido (y el método manual recién introducido podría optimizarse aún más). Este tipo de mejora en la eficiencia computacional es extremadamente importante en la práctica sistemática, ya que (como se dijo) las aplicaciones reales contienen números mucho más grandes de taxa y de caracteres, lo cual trae aparejado números astronómicamente grandes de árboles y de asignaciones posibles. En esos casos, las diferencias en la eficiencia se vuelven aún mayores (véase la sección 6.1.4. para un cálculo sobre el número de pasos algorítmicos requeridos en un análisis más grande). De hecho, una parte importante del desarrollo teórico (pasado y presente) de la cladística consiste

en encontrar algoritmos que implementen funciones de búsqueda más eficientes. El software introducido en esta tesis permite reflejar o reconstruir también este aspecto de la práctica científica.

5.5. Conclusiones

En este capítulo se ilustró la fertilidad y aplicabilidad de los métodos de contrastación y del software que permite llevarlos a cabo (Reconstructor) con un caso de aplicación real (la cladística). Para ello, las secciones anteriores ejemplificaron en detalle cómo se cargaron los axiomas, los modelos, las condiciones de ligadura y los métodos manuales a fin de introducir la reconstrucción en el programa. También se mostraron los resultados de la aplicación de los métodos de contrastación de reconstrucciones aplicados a estos elementos.

Cabe resaltar que la versión de la reconstrucción presentada en el capítulo 3 e introducida en el software en este capítulo pasó exitosamente todos los tests, pero que esto esconde algunos fracasos previos. Es decir, la propia carga de la reconstrucción en el programa me permitió detectar y arreglar errores en versiones previas de la reconstrucción. En este sentido, creo que la utilidad del software estriba no solo en permitir usuario testear una reconstrucción completa ya realizada, sino en funcionar como una ayuda en el proceso mismo de reconstruir una teoría.

Un ejemplo ilustrativo de un fallo previo, corregido gracias a la utilización del software, es el siguiente. Inicialmente había considerado una formulación alternativa de la ley fundamental (2), que afirmaba que el árbol real refina al consenso estricto de los árboles del conjunto *AO*. Formalmente, esto sería: $\bigcap AO \subseteq A_R$. Sin embargo, el resultado de la contrastación de la reconstrucción con m_2 daba como resultado que la ley (2) así formulada era verdadera (cuando debía ser falsa). Tras investigar el log de valuaciones descubrí el problema (y de hecho aprendí algo que no sabía): el consenso estricto de un conjunto de árboles contiene menos información que los árboles que se usaron para construirlo. Por ejemplo, el consenso estricto entre los árboles (jerarquías) {{A, B, C}, {A, B}, {A}, {B}, {C}} y {{A, B, C}, {A, C}, {A}, {B}, {C}} consiste en su intersección, {{A, B, C}, {A}, {B}, {C}}. Sin embargo, el árbol {{A, B, C}, {B, C}, {A}, {B}, {C}} también refina a este consenso estricto, aunque no haya estado entre los árboles originales usados para su construcción. Por ese motivo, el árbol real de m_2 refinaba al consenso

estricto de los árboles óptimos del conjunto *AO* aunque no estaba entre esos árboles. La versión actual de la ley fundamental (2) corrige ese problema.

PARTE III - Consecuencias del análisis anterior

Capítulo 6. Consecuencias

En este capítulo se extraen algunas consecuencias del análisis presentado en los capítulos 2 a 5. Se brindan, por un lado, algunas extensiones y posibles modificaciones al aparato metateórico estructuralista, con foco en la noción de método de determinación (sección 6.1). Por otro lado, se extraen algunas consecuencias para el debate sobre la noción de homología y la circularidad en el marco de la cladística (sección 6.2).

6.1. Consecuencias sobre el estructuralismo: la noción de método de determinación

En esta sección se presenta una elucidación alternativa de la noción de método de determinación, basada en el concepto de algoritmo. Adicionalmente, se brinda una elucidación de la noción de *ejecución* de un algoritmo, entendiéndola como una sucesión de modelos, y se analiza cómo esto posibilita una distinción entre métodos T-teóricos y T-no-teóricos (esto se comprenderá mejor más adelante). A pesar de haber brindado una elucidación alternativa, se sostiene que tanto la elucidación clásica como la alternativa pueden ser útiles en diferentes contextos. La mayor ventaja de la elucidación alternativa propia que brindo aquí es que los métodos *qua* algoritmos pueden cargarse en el programa Reconstructor y ejecutarse para completar un modelo. Por último, se sostiene que los métodos de determinación automáticos en Reconstructor son una herramienta útil para diagnosticar qué conceptos son T-teóricos en un segundo sentido que los estructuralistas suelen reconocer (esto es, determinables a través de la teoría, véase más adelante).

6.1.1. Introducción

Como se expuso en la sección 3.6 del capítulo 3, la noción de método de determinación es sumamente importante para la metateoría estructuralista. Como se mencionó en esa sección, la distinción de T-teoricidad y el significado (en tanto sentido fregeano) de los conceptos están

relacionados con esta noción. Por otro lado, como se mostró en el capítulo anterior, la búsqueda de métodos de determinación más eficientes es sumamente importante para la práctica de los propios científicos (o al menos lo es para los sistemáticos).

La elucidación estructuralista estándar fue presentada en la subsección 3.6.1, y consiste en identificar a estos con clases de modelos que satisfacen cierta oración ϕ, en donde ϕ contiene a un término *t* a determinar cuyo valor queda unívoca y sistemáticamente determinado dados los valores de los demás términos que ϕ contiene. En la subsección siguiente (3.6.2) se presentó lo que podría considerarse un caso paradigmático de un método de determinación (a ser reconstruido), la optimización de Fitch. Una reconstrucción de este método fue brindada en el capítulo 5, sección 5.4, aunque no bajo la concepción estándar de método de determinación. En cambio, fue introducido a Reconstructor como un algoritmo.[41] En este capítulo se expande sobre esta idea, proponiendo una elucidación alternativa de la noción de método de determinación, e identificando a estos con algoritmos más que con clases de modelos.

Considerar a los métodos de determinación como algoritmos tiene la ventaja inmediata de permitir que estos sean cargados en Reconstructor y aplicados sobre modelos incompletos para llenar las denotaciones faltantes. Esto, a su vez, permite testear de modo más directo la adecuación de la formalización de los métodos. O bien, si ya se tiene confianza en la reconstrucción de los métodos, permite hacer predicciones automatizadas desde Reconstructor.

Por otro lado, proponer una elucidación alternativa conlleva al menos dos desafíos. En primer lugar, esta debe ser tan precisa como la original (la exactitud es, según Carnap, uno de los criterios bajo los cuales se debe evaluar una elucidación conceptual; véase Carnap, 1950). Además, la noción estándar permite dar un criterio claro para establecer si un método presupone o no una teoría T, y de ese modo distinguir entre conceptos T-teóricos y T-no-teóricos, mientras que no es inmediatamente claro cómo hacer esto con la noción alternativa. En las dos subsecciones siguientes me encargaré de precisar la noción alternativa, mostrando cómo pueden pasarse estos dos desafíos.

6.1.2. La noción de algoritmo

[41] Aunque esto no quita que pueda darse una descripción matemática más abstracta de este procedimiento en la que quede representado como una clase de modelos.

En cuando a la exactitud, la noción de algoritmo puede ser elucidada con distintos grados de precisión. A nivel menos preciso y más informal, un algoritmo es un método efectivo para calcular el valor de una función. Un método efectivo es una secuencia de instrucciones (oraciones imperativas) que:

- Es finita.
- No requieren ningún tipo de creatividad para ser ejecutadas.
- Aplicada a un caso de su ámbito de aplicación, termina en un tiempo finito.
- Aplicada a un caso de su ámbito de aplicación, devuelve el valor correcto de la función para los argumentos dados como *input*.

Así, volviendo a la noción de método de determinación, la función sería una que toma como *input* ciertos conceptos, y devuelve un valor del concepto a determinar; el método de determinación quedaría identificado con el algoritmo que permite calcular el valor de esta función. Que el algoritmo calcule el valor *de una función*, la cual debe (por ser función) cumplir con el requisito de unicidad, implica que el algoritmo devolverá un valor unívoco para un conjunto de argumentos *input* (esto es análogo a uno de los requisitos de la elucidación estándar). Por otra parte, que el método sea identificado con el algoritmo para calcular el valor de la función y no con la función misma implica que el valor de la función debe ser *computable* —algo que no estaba en la elucidación estándar pero sí está en la práctica científica.[42]

Si bien la elucidación anterior es relativamente vaga e imprecisa (p.e. ¿cuándo un paso requiere creatividad? ¿cuál es la sintaxis aceptable para las oraciones imperativas? etc.), los algoritmos se suelen escribir lenguajes de programación, que tienen sintaxis claras y especificadas, y para los cuales la noción de creatividad no es problemática (lo no-creativo es lo que puede procesar una computadora, y que viene incorporado al lenguaje a través de los términos por *default* del lenguaje de programación). Para algunos ejemplos de algoritmos escritos en el lenguaje de programación Python véanse los capítulos 4 y 5.

[42] Esto último puede ser discutible, debería buscarse si existe un caso de un método de determinación *usado en la práctica científica* tal que el término *t* quede unívocamente determinado por una oración $\phi(D_1, \ldots, D_n, R_1, \ldots, R_m, t)$ y que encontrar el valor de *t* que hace verdadera a ϕ no sea una función computable. En tal caso, es posible que, en este punto, la elucidación estándar sea superior a la alternativa que estoy proponiendo. En caso contrario, sostendría que poner requisitos más fuertes es una ventaja de la elucidación alternativa.

Existen, por otro lado, elucidaciones más precisas de las nociones de algoritmo, computabilidad, etc. provenientes de las ciencias de la computación, por ejemplo aquellas que apelan a la noción de máquina de Turing o a la recursividad (Boolos, Burgess, & Jeffrey, 2007). Sin embargo, especificar algoritmos bajo estas elucidaciones más precisas suele ser mucho más engorroso y menos práctico (p.e. diseñar una máquina de Turing es similar a programar en lenguaje máquina, o en algún lenguaje de muy bajo nivel). Aun así, a fines teóricos o conceptuales, basta con saber que las nociones más precisas existen y que todo algoritmo dado en un lenguaje de programación de alto nivel se puede traducir a un algoritmo especificado de modo más preciso (p.e. a una tabla de estados de una máquina de Turing).

Nuevamente citando a Carnap (1950), otro de los criterios para evaluar la adecuación de una elucidación es la fertilidad. Si bien las nociones de las ciencias de la computación son muy precisas, sería muy poco fructífero intentar reconstruir métodos usando esas nociones —serían muy engorrosas y poco clarificadoras de la estructura de las teorías y de la práctica científica real. En cambio, como se dijo, los lenguajes de programación suelen tener especificaciones suficientemente claras para su sintaxis y resulta mucho más fructífero representar métodos formulándolos en ellos (p.e. como se mostró en los capítulos anteriores, ello permite ejecutarlos en una computadora). De ese modo, en adelante asumiré que los métodos de determinación pueden darse en algún lenguaje de programación (o bien en pseudocódigo fácilmente adaptable a cualquier lenguaje).

En la sección siguiente desarrollaré una noción (propia) de ejecución de un algoritmo como una secuencia de modelos, que me permitirá responder al segundo desafío planteado anteriormente: permitir distinguir entre métodos T-teóricos y T-no-teóricos.

6.1.3. Ejecución de un algoritmo y T-teoricidad

Para responder al segundo desafío, introduciré un aparato formal para elucidar a la noción de una *ejecución* de un algoritmo, que consistirá en pensarla como una secuencia de modelos (completos o incompletos). Cada modelo representará al estado informacional del sistema en un momento dado. El modelo inicial representa a los parámetros dados como *input* al algoritmo. Cada instrucción del algoritmo puede modificar alguno de esos parámetros, generando un modelo nuevo. Considérese el siguiente ejemplo para iluminar esta idea. El algoritmo para determinar las

velocidades en un tiempo sucesor en la mecánica cartesiana (dado en la subsección 4.5.1) era como sigue:

Inputs: *B*, *T*, *before*, *CL*, *v*

1. Para todo instante de tiempo x_1 en T:
    2. Si hay un tiempo x_2 sucesor en *T*:
    3. Para todo cuerpo y_1 en *B*:
        4. Para todo cuerpo y_2 en *B*:
            5. Si $\langle y_1, y_2, x_1\rangle \in CL$:
            6. Obtener las velocidades $v(y_1, x_1)$ y $v(y_2, x_1)$
            7. Si ambas están determinadas:
            8. Agregar $v(y_1, x_2) = -v(y_1, x_1)$ a *v*
            9. Agregar $v(y_2, x_2) = -v(y_2, x_1)$ a *v*
10. Devolver *v*

Considérese ahora una ejecución de este algoritmo en un modelo inicial *m* que contiene dos cuerpos b_1 y b_2 y dos tiempos t_1 y t_2. Los cuerpos chocan en t_1 con velocidades de 5 y –5. Es decir, el modelo inicial se ve como sigue:

$m = \langle B_m, T_m, before_m, CL_m, v_m\rangle$

$B_m = \{b_1, b_2\}$

$T_m = \{t_1, t_2\}$

$before_m = \{\langle t_1, t_2\rangle\}$

$CL_m = \{\langle b_1, b_2, t_1\rangle, \langle b_2, b_1, t_1\rangle\}$

$v_m = \{\langle\langle b_1, t_1\rangle, 5\rangle, \langle\langle b_2, t_1\rangle, -5\rangle\}$

Este sería el estado informacional dado como *input* al algoritmo / método de determinación. La ejecución del algoritmo comienza por la primera instrucción. La variable libre T en ella es reemplazada por la denotación de T en el modelo inicial. Así, la instrucción dice:

1. Para todo x_1 en $\{t_1, t_2\}$

El bloque de código siguiente se ejecutará, por tanto, instanciando a la variable x_1 con t_1, en primer lugar, e instanciando a x_1 con t_2, en segundo. En la primera de estas instanciaciones, la segunda instrucción chequea si hay un tiempo sucesor a t_1 en m. Al encontrarlo (ya que $\langle t_1, t_2\rangle \in before_m$) ejecuta el bloque de código siguiente.[43] Ninguna de estas dos instrucciones cambió aún ninguna denotación en el modelo. Es fácil, sin embargo, ver que si se continua la ejecución se llegará eventualmente a las instrucciones (instanciadas)

8. Agregar $v(b_1, t_2) = -5$ a v
9. Agregar $v(b_2, t_2) = 5$ a v

La instrucción 8. modifica el modelo (genera un nuevo modelo m') haciendo que:

$$v_{m'} = \{\langle\langle b_1, t_1\rangle, 5\rangle, \langle\langle b_2, t_1\rangle, -5\rangle, \langle\langle b_1, t_2\rangle, -5\rangle\}$$

mientras que 9. genera el modelo m'':

$$m'' = \langle B_{m''}, T_{m''}, before_{m''}, CL_{m''}, v_{m''}\rangle$$
$$B_{m''} = \{b_1, b_2\}$$
$$T_{m''} = \{t_1, t_2\}$$
$$before_{m''} = \{\langle t_1, t_2\rangle\}$$
$$CL_{m''} = \{\langle b_1, b_2, t_1\rangle, \langle b_2, b_1, t_1\rangle\}$$
$$v_{m''} = \{\langle\langle b_1, t_1\rangle, 5\rangle, \langle\langle b_2, t_1\rangle, -5\rangle, \langle\langle b_1, t_2\rangle, -5\rangle, \langle\langle b_2, t_2\rangle, 5\rangle\}$$

Así, la ejecución de este algoritmo sobre m puede representarse como la secuencia $\langle m, m', m''\rangle$. Con esto en mente, es posible definir cuándo un método de determinación (*qua* algoritmo) presupone una teoría T del siguiente modo: un método de determinación presupone una teoría T si

[43] Nótese que, así como están formuladas, las instrucciones 1 y 2 (y otras en el algoritmo) no son estrictamente oraciones imperativas (de hecho, ni siquiera son oraciones). Se las podría reformular, sin embargo, como oraciones imperativas (p.e. "si (…) ejecutar el siguiente bloque de código"). Por otro lado, este tipo de recursos (condicionales, *loops*) son típicos de los lenguajes de programación de alto nivel. Si traduce el algoritmo a un lenguaje de más bajo nivel, todas las instrucciones quedarían representadas como oraciones imperativas.

y solo si, para toda ejecución (secuencia de modelos), el último modelo es un modelo de T.[44] Esta definición es además bastante cercana en espíritu a la definición estándar de presuposición (presentada en 3.6.1).[45] Adicionalmente (y de manera idéntica a la elucidación estándar), un concepto será T-teórico cuando todo método de determinación suyo presupone T (bajo la nueva noción de método y de presuposición).

En las siguientes subsecciones se mencionan algunas ventajas de la elucidación alternativa recién propuesta frente a la clásica.

6.1.4. Ventajas de la elucidación alternativa

Una primera ventaja de la elucidación alternativa ya fue mencionada y ejemplificada anteriormente: la posibilidad de ejecutar métodos para realizar predicciones/determinaciones automatizadas a partir de la reconstrucción. Esta ventaja es especialmente importante si (como es el caso en la presente tesis) se pone énfasis en la contrastación de reconstrucciones. Con ese objetivo, se vuelve importante que la reconstrucción formal funcione como / sea capaz de realizar todo lo que realiza la teoría original. Esto es más fuerte que la idea de que la reconstrucción debe ser meramente una representación formal / abstracta adecuada de una teoría empírica. Si lo que se busca es aplicabilidad (y especialmente aplicabilidad automatizada), la posibilidad de ejecutar algoritmos en una computadora es una ventaja importante.[46]

[44] También sería posible definir esta noción poniendo el requisito más débil de que algún modelo de la secuencia sea un modelo de T, pero no necesariamente el último. Esto equivaldría a pedir que las leyes de la teoría se usen en la ejecución del método, pero dejaría abierta la posibilidad de que el propio método las invalide luego. Esto último, sin embargo, daría como resultado métodos que son teóricamente inadecuados, de modo que la posibilidad de que ello ocurra no me preocupa demasiado. Es decir, cómo se comporta este criterio para métodos inadecuados es irrelevante.

[45] Adicionalmente, si se quisiera mantenerse en un espíritu semanticista, podría identificarse al método de determinación con la clase de las secuencias de modelos que constituyen sus posibles ejecuciones, en lugar de con la secuencia de oraciones imperativas —a mi entender esto sería equivalente (y posiblemente menos claro y fácil de entender).

[46] Cuando hablo aquí de aplicabilidad automatizada no me refiero a la posibilidad de aplicar automáticamente la teoría de manera completa. Esto podría requerir cosas como, por ejemplo, aplicar teorías previas para determinar las extensiones de los conceptos del *explanandum*, o decidir bajo qué rama de la red teórica intentar subsumir un modelo potencial. Una reconstrucción de una teoría T no necesariamente incluirá algoritmos para realizar estas tareas (aunque puede incluirlos, p.e. Reconstructor incluye la posibilidad de introducir métodos de determinación T-no-teóricos, véase el capítulo 4). En cambio, aquí me refiero puramente a la posibilidad de, una vez cargado un método de determinación, aplicar ese método para encontrar la extensión de un concepto en un modelo (completo o incompleto) pre-cargado.

Una segunda ventaja está relacionada con la ya mencionada. La metateoría estructuralista contiene un segundo criterio de T-teoricidad (distinto e independiente del desarrollado en 3.5.2) según el cual un concepto es T-teórico cuando se lo puede determinar usando T (a diferencia del criterio estándar y más usado según el cual un concepto es T-teórico cuando *no* se lo puede determinar *sin* T). Para evitar confusiones, me referiré a esta segunda noción como T-determinabilidad. Nótese que todo concepto T-teórico es T-determinable, pero que un concepto T-no-teórico puede también ser T-determinable (tener métodos de determinación T-no-teóricos y T-teóricos). De hecho, los conceptos T-no-teóricos y T-determinables son aquellos que permiten contrastar a las teorías, dado que una teoría se contrasta determinando a un mismo concepto T-teórica y T-no-teóricamente y viendo si los resultados coinciden (Ginnobili & Roffé, 2018). La noción de un método como un algoritmo y una ejecución como una secuencia de modelos es útil aquí debido a los métodos automáticos de Reconstructor. Recuérdese que lo que estos hacen es interpretar a las leyes como oraciones imperativas, y construir una secuencia de modelos en donde las denotaciones de algunos conceptos están completadas o modificadas. Es decir, los métodos automáticos de Reconstructor entran dentro de la caracterización alternativa de método de determinación. Uniendo ambas cosas, la noción alternativa de método y la implementación de métodos automáticos en Reconstructor permiten automatizar el criterio de T-determinabilidad: si las leyes fundamentales o las especializaciones usadas como métodos automáticos permiten determinar un concepto entonces ese concepto es T-determinable. Esto, sin embargo, brinda una condición suficiente pero no necesaria para la T-determinabilidad, debido a las limitaciones de la implementación computacional de los métodos automáticos (no siempre que existe un valor unívoco para un concepto el método automático correspondiente lo encuentra, véase la subsección 4.6.5).[47]

La segunda ventaja ya fue sugerida en la sección 5.4, y consiste en que este modo de considerar a los métodos de determinación permite capturar mejor cierta parte de la práctica científica: la búsqueda de mejoras en la eficiencia computacional de los métodos. Esto es especialmente importante en áreas de la ciencia (como la sistemática) en las que los problemas son combinatoriamente explosivos, en donde el número de opciones a considerar o de pasos a ejecutar

[47] Otro motivo para sostener que los métodos automáticos de Reconstructor dan condiciones suficientes pero no necesarias para la T-determinabilidad es que las inferencias realizadas en las determinaciones T-teóricas no siempre son deductivas. Puede ser posible que, en ocasiones, la inferencia de la extensión de un término sea abductiva y/o probabilística (en cuyo caso Reconstructor no las hará).

en un algoritmo que resuelve cierto problema crece muy rápidamente cuando los *inputs* crecen (i.e. tienen una complejidad de tipo NP o superior, véase Goldreich, 2010).

El algoritmo de Fitch para computar el largo de un árbol es un ejemplo de una mejora de este tipo. El algoritmo que se desprende inmediatamente de las leyes de **CLAD** para calcular el largo$_2$ de un árbol mide el largo$_1$ bajo toda asignación posible. Con $|C|$ caracteres y $|T|$ terminales hay $\prod_{i=1}^{|C|} |C_i|^{|T|-1}$ asignaciones posibles (véase la sección 3.2.5), y el cálculo del largo$_1$ visita una vez cada rama por árbol (cada árbol tiene $|N| - 1$ ramas) para cada asignación. De modo que este algoritmo requiere aproximadamente $|N-1| \times \prod_{i=1}^{|C|} |C_i|^{|T|-1}$ cálculos para computar el largo de un árbol. En cambio, el algoritmo de Fitch visita una vez cada nodo interno por cada carácter, de modo que requiere solo $|C| \times (|T| - 1)$ pasos. Con 20 taxa y 20 caracteres de dos estados cada uno (un número relativamente bajo para un análisis real) la diferencia es de $9{,}6 \times 10^{115}$ vs. 380 cálculos para obtener el largo de un árbol.

Este tipo de consideraciones permite pensar en un nuevo tipo de desarrollo teórico (diacrónico) que una teoría puede experimentar. Algunos de los modos usualmente reconocidos (Moulines, 2011) son:

- **Emergencia o cristalización:** emergencia gradual de una nueva teoría desde (en términos kuhnianos) un período pre-paradigmático
- **Evolución teórica:** modificación no-esencial de una teoría, a través de la adición o eliminación de especializaciones. Lo que ocurriría en la ciencia normal kuhniana.
- **Incorporación o incrustación:** Las aplicaciones intencionales de una teoría pasan a ser aplicaciones de otra, que la incluye (p.e. la mecánica del choque en la mecánica newtoniana). Los casos de reducción interteórica entrarían aquí.
- **Suplantación:** Un cambio de paradigma kuhniano. Reemplazo de una red teórica por otra distinta que subsume (aproximadamente) a al menos algunas de las mismas aplicaciones intencionales.

Las consideraciones precedentes permiten reconocer un nuevo subtipo de desarrollo por evolución teórica, que llamaré *mejora en la eficiencia de métodos*. Este consiste justamente en la

incorporación de un nuevo método de determinación más eficiente que los anteriores. En el caso de la cladística, estas mejoras en la eficiencia consistieron en una reducción de la complejidad algorítmica (p.e. en los algoritmos de optimización de caracteres). Este sería un cambio relativamente pequeño en comparación con los anteriores, en el sentido de que las mejoras en la eficiencia no ponen restricciones fácticas adicionales (aunque si pueden ser clave para que casos previamente intratables computacionalmente se vuelvan aplicaciones exitosas, es decir, pueden ser muy importantes en otros sentidos). De hecho buena parte de la ciencia normal en sistemática consistió en este tipo de desarrollo. Nuevamente, dado que la combinatoria de árboles, asignaciones, etc. aumenta muy rápidamente con el número de taxa y de caracteres, encontrar algoritmos eficientes fue históricamente muy importante para volver a la cladística aplicable a números interesantes de taxa y caracteres. Para ilustrar este punto nótese que, de los 5 artículos más citados de la revista *Systematic Biology* (anteriormente *Systematic Zoology*), la revista más importante del área, 4 son resultados sobre algoritmos más eficientes para computar ciertos problemas (el quinto es la descripción de un programa de computación que realiza búsquedas bayesianas de árboles). Ellos tienen entre 5.000 y 13.000 citas, siendo el artículo de Fitch de 1971 el cuarto, con 5.715 citas (fuente: https://www.lens.org/).

El caso de Fitch ilustra otro punto importante: el algoritmo requiere *inputs* de cierto tipo para ser aplicable (en particular, matrices de costo uniformes), que no son características de toda aplicación de la cladística. Es decir, el método de Fitch aplica a un subconjunto de los casos del dominio de aplicación de **CLAD**. En esto se parece a una especialización de la teoría, aunque no es exactamente eso ya que el método de Fitch no pone restricciones fácticas adicionales (no restringe la clase de los modelos) —es decir, llega al mismo resultado que la búsqueda menos eficiente usando el algoritmo que se desprende de las leyes. Así, es posible acuñar otra noción para hablar acerca de la relación de mejorar la eficiencia para un subconjunto de casos, que llamaré *restricción algorítmica*. Formalmente, si α y α' son dos algoritmos (secuencias de oraciones imperativas), que se aplican a los conjuntos de aplicaciones pretendidas $I(\alpha)$ e $I(\alpha')$, diré que α' es una restricción algorítmica de α si y solo si $I(\alpha') \subseteq I(\alpha)$ y $O(\alpha) < O(\alpha')$ ($O(\ldots)$ es notación estándar en ciencias de la computación para hablar sobre la complejidad algorítmica).

6.1.5. Conclusiones

En esta sección se propuso una elucidación alternativa de la noción de método de determinación. Esta elucidación alternativa identifica a los métodos de determinación con algoritmos, en lugar de clases de modelos. Se identificaron dos posibles desventajas de esta elucidación frente a la estándar, a saber, la posible mayor imprecisión y la falta de un criterio de T-teoricidad basada en ellos. Se dio una respuesta a estos dos puntos precisando la noción de algoritmo y la de ejecución de un algoritmo. Se argumentó que la noción alternativa tiene algunas ventajas sobre la clásica, entre ellas: la posibilidad de ejecutar algoritmos (métodos de determinación) en Reconstructor para testear la reconstrucción de los métodos y/o obtener predicciones automatizadas a partir de la reconstrucción; la posibilidad de establecer de manera automatizada la T-determinabilidad de los conceptos aplicando los métodos automáticos de Reconstructor (los cuales conforman a la noción alternativa); y la posibilidad de distinguir un nuevo tipo de desarrollo teórico (la mejora en la eficiencia de métodos) y de una relación similar en un aspecto a la especialización, para métodos de determinación (la restricción algorítmica) que jugaron un rol muy importante en la historia de la sistemática, permitiendo hacer un análisis más fino del desarrollo diacrónico de la ciencia.

Aun así, afirmar que esta noción debería reemplazar a la anterior de método de determinación es demasiado fuerte. La aplicación de la noción nueva se realizó en un único caso (dos si se cuenta a MCAR en el capítulo 4), resta aún ver si ella es aplicable y fértil en tanto aplicada a otras teorías.

6.2. Consecuencias sobre la cladística: la noción de homología

En esta sección se extraen conclusiones más específicamente de la reconstrucción de la cladística presentada en el capítulo 3, sobre algunas cuestiones metateóricas de la sistemática. Existen muchas discusiones metateóricas para las cuales la reconstrucción podría tener consecuencias. Me centraré aquí solo en una de ellas, para la cual la reconstrucción presentada tiene consecuencias más o menos

inmediatas y claras.[48] Ella concierne al debate clásico en torno a la circularidad en torno a la noción de homología y a cómo este problema reaparece en el marco de la cladística.

Procederé como sigue. En las secciones 6.2.1 y 6.2.2 se introduce el problema de la circularidad en la determinación de las homologías, primero de modo clásico y luego dentro del marco de la cladística. Luego, se expone una solución usual a este problema y se la sitúa en el marco de la discusión más amplia acerca del cladismo de patrón (6.2.3, 6.2.4 y 6.2.5). Por último, se consideran algunas críticas a esta solución y se responde a ellas apelando a la noción de T-teoricidad del estructuralismo metateórico (6.2.5 y 6.2.6). Finalmente, se extraen algunas conclusiones.

6.2.1. El problema clásico

Como se presentó en el capítulo 2 (sección 2.1.2), el *explanandum* de la teoría del origen común de Darwin consiste en la presencia de homologías. El término "homología" era usado por Darwin en el sentido oweniano, de rasgos que tienen un cierto parecido estructural.[49] Esto es, rasgos que son similares en la cantidad y disposición de las partes, la composición de sus elementos, etc. Así, buscaba dar una explicación alternativa a este fenómeno ya conocido (la presencia de homologías). La explicación anterior consistía en la postulación de un tipo ideal, que en algunas versiones era entendido como una idea en la mente de Dios. Según Darwin, en cambio, la presencia de homologías se debe a que ambas especies actuales habrían heredado el rasgo de un ancestro común. El rasgo podría modificarse independientemente en cada linaje en base a los diferentes problemas ambientales que estos enfrentaron, pero habría una tendencia a mantener la estructura básica (ya que la selección natural opera modificando los rasgos existentes). De ese modo, puede afirmarse que (al menos para Darwin originalmente):

[48] El análisis presentado anteriormente tiene, sin embargo, otras consecuencias interesantes (aunque menos directas), en las que no entraré aquí por motivos de pertinencia. Algunos de esos resultados se encuentran en prensa en Roffé (en prensa).

[49] Al menos en la primera edición de *El origen* lo usa siempre de ese modo. En la última edición comenta, respecto de la terminología propuesta por Lankester (1870) —quien divide a las homologías en homoplasias y homogenias— que las homoplasias de Lankester son lo que él llama analogías, quizás implicaturando que las homologías coinciden con las homogenias de aquel autor (los rasgos derivados de un ancestro común), aunque no es del todo claro que esa implicatura esté presente (véase Darwin [1872] 1992, pp. 574-575). De todos modos, en la edición de 1872, la gran mayoría de las veces usa el término homología del modo clásico. Agradezco a Daniel Blanco por discusiones sobre este punto.

(*i*) La herencia de un rasgo a partir de un ancestro común *explica* la presencia de homologías en su progenie

Ahora bien, con el desarrollo de la biología contemporánea en el siglo XX, muchos autores consideraron que ocurrió un cambio conceptual en la noción de homología y su significado. En particular, es común la posición según la cual afirmar que dos rasgos son homólogos *significa* que son derivados a partir de un ancestro común. Para estos autores, la noción de homología está *definida* a partir de la ancestría común (Futuyma, 2005; Mayr, 1982; Pearson, 2010; Simpson, 1961, entre muchos otros). Por ejemplo, Mayr afirma que:

> *El término 'homólogo' existía antes de 1859* [el año de publicación de El origen de las especies] *pero adquirió el significado aceptado actualmente cuando Darwin estableció la teoría del origen común. En esta teoría, la definición biológicamente más significativa de 'homólogo' es: 'una característica en dos o más taxa es homóloga cuando es derivada de la misma (o una correspondiente) característica en un ancestro común'. ¿Cuál es la naturaleza de la evidencia que puede usarse para demostrar la probable homología en un caso dado? Hay un conjunto de criterios tales (como la posición de una estructura en relación con otras), pero es completamente confundente incluir esa evidencia en la definición de 'homólogo'.* (Mayr, 1982, p. 45, traducción propia)

Un ejemplo más reciente puede encontrarse en el manual de biología evolutiva de Futuyma, en el cual se sostiene que "un carácter (o estado de un carácter) es definido como homólogo en dos especies si ha sido derivado de su ancestro común" (Futuyma, 2005, p. 49). Al igual que Mayr, Futuyma afirma que los criterios clásicos Owenianos son útiles para *hipotetizar* la homología, es decir, que funcionan en todo caso como heurísticas para el descubrimiento de homologías, pero que la contrastación de estas hipótesis se realiza construyendo árboles filogenéticos y determinando la evolución de los caracteres. De manera semejante, Cristopher Pearson (en un trabajo que se discutirá en mayor detalle más adelante) afirma en el marco de la discusión entre cladistas de patrón y cladistas filogenéticos que:

> *Contrariamente al enfoque tradicional* [la noción Oweniana de homología]*, gran parte de la biología contemporánea, así como de la filosofía de la biología, interpreta a la homología como un concepto histórico. De acuerdo a este enfoque los rasgos son homólogos solo en caso de que sean derivados del mismo rasgo en un ancestro común. (…) Como los conceptos de "abuela" o "adaptación", entonces, el de homología tiene una dimensión histórica. Uno simplemente no puede entender qué es que dos rasgos sean homólogos sin prestar atención a las relaciones evolutivas de los taxa que portan los rasgos en cuestión.* (Pearson, 2010, pp. 483-484, traducción propia).

De ese modo, tenemos un segundo principio que afirma que:

> (*ii*) Las homologías están *definidas* como aquellos rasgos derivados de un ancestro común

Si se toma (*i*) y se reemplaza al concepto de homología por su *definiens*, lo que queda es que:

> (*i*') La herencia de un rasgo a partir de un ancestro común explica la presencia de rasgos derivados del ancestro, en su progenie

Una afirmación claramente circular (véanse Blanco, 2012; Jardine, 1967; Patterson, 1982; Wiley & Lieberman, 2011, p. 117, entre otros, para formulaciones similares de este problema). Si se cree que la explicación darwiniana de (lo que Owen llamaba) homologías no es circular ni trivial entonces o bien (*i*) o bien (*ii*) deben abandonarse o modificarse.

En la sección siguiente se examina cómo este problema reaparece en el marco de la cladística contemporánea. Pero antes de pasar a ello es interesante notar que el caso del concepto de adaptación traído a colación por Pearson es relevante ya que presenta muchas similaridades con el de homología. Se ha afirmado, por ejemplo, que la explicación de la presencia de adaptaciones por parte de la teoría de la selección natural es circular porque a la vez (*i*b) la selección natural explica la adquisición de adaptaciones y (*ii*b) las adaptaciones son (por definición) aquellos rasgos surgidos por selección natural. Así, reemplazando al concepto de adaptación en (*i*b) por su *definiens*, se obtiene que la selección natural explica la adquisición de rasgos surgidos por selección natural. Adicionalmente, también es cierto que Darwin propone la selección natural para explicar un fenómeno previamente conocido y explicado de otro modo. Por ejemplo, los teólogos naturales

británicos explicaban la presencia de adaptaciones apelando a un diseñador (aunque hay cierta discusión acerca de si el concepto de adaptación es el mismo en Darwin que en la teología natural, véanse Blanco, 2008; Caponi, 2011; Ginnobili, 2013). Por otro lado, algunos biólogos y filósofos contemporáneos creen que el concepto pre-darwiniano de adaptación no forma parte de la biología actual sino que habría sido reemplazado por (*ii*b), como afirma Pearson con respecto a las homologías. Más adelante tendré algo más que decir sobre este paralelismo.

6.2.2. Homología y circularidad en el marco de la cladística

Para comprender cómo ambas patas del dilema resurgen en el marco de la cladística, considérese lo siguiente. Si la noción de homología es entendida o definida a partir de la ancestría común, entonces la aplicación de la cladística (u otros métodos de inferencia filogenética) es fundamental para establecer cuáles rasgos son homólogos (como afirman Mayr y Futuyma). La cladística puede usarse para determinar esto ya que: (1) permite inferir cuál es el árbol filogenético real, y (2) dado un árbol, permite inferir los estados de los caracteres de los ancestros hipotéticos.

Nótese que una asignación de estados a los ancestros hipotéticos ((2) en el párrafo anterior) implica un esquema de homologías (en el sentido filogenético). Por ejemplo, tómese el árbol y las optimizaciones (b) y (c) de la figura 2.6 del capítulo 2:

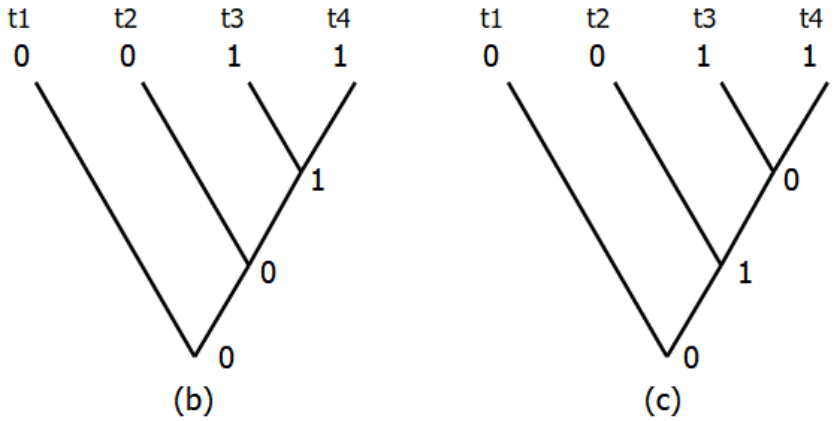


Fig. 6.46. Dos asignaciones posibles para un árbol con 4 taxa terminales

Según la primera asignación, el estado 1 es homólogo entre t_3 y t_4, ya que ambos lo heredaron de su ancestro común. Lo mismo ocurre con el estado 0 para t_1 y t_2. En cambio, en la segunda asignación tanto el estado 1 para t_3 y t_4 como el estado 0 para t_1 y t_2 son homoplásicos (1 es una convergencia entre t_3 y t_4, y 0 es una reversión en t_2). De ese modo, algunos cladistas han considerado que la noción de homología es equivalente a la de sinapomorfía —o bien a la de sinapomorfía + simplesiomorfía (Nixon & Carpenter, 2012; Patterson, 1982; Richter, 2017). Dado que la cladística permite elegir (basándose en un mismo criterio de optimalidad, que es el largo del árbol) tanto el árbol como la asignación óptima, puede determinar cuál esquema de homologías es el real.[50] En conclusión hasta el momento, si se adopta la definición filogenética de homología, entonces la determinación de las homologías presupone la aplicación de la cladística (i.e. es parte del *output* de un análisis filogenético).

La otra pata del dilema surge de considerar a la matriz de datos de la cladística (uno de los *inputs* del análisis filogenético). Existen múltiples, quizás incluso infinitas, maneras de dividir a un organismo en rasgos o características. Pero no todo rasgo posible de un ser vivo se considera útil a fin de inferir una filogenia. Por ejemplo, los rasgos meramente *análogos* (i.e. similares en la función, como ser, las alas de un pájaro y las de un insecto) no son indicativos de herencia a partir de un ancestro común, sino más bien de que los organismos en cuestión enfrentaron problemas adaptativos similares y encontraron soluciones similares. Es decir, su presencia compartida se explica por selección natural (operando de manera semejante pero independiente en ambos linajes) y no por herencia a partir de un ancestro común. En otras palabras, los caracteres y estados de la matriz de datos deben elegirse con cuidado. Matrices de datos con caracteres espurios resultarán en agrupamientos y árboles espurios (Mishler, 2005; Rieppel & Kearney, 2002). Uno de los requisitos que se suele poner para la inclusión de un carácter en una matriz de datos cladista es, como se dijo en el capítulo 2, que sea una homología. Esto implica dos cosas distintas: (*a*) que todos los estados del carácter formen una serie de transformación. Por ejemplo, los estados 0, 1 y 2 del carácter C_1 tienen que ser *estados alternativos de la misma característica*, y en ese sentido todos homólogos entre sí. Este requisito se suele formular diciendo que todos los estados de un mismo carácter deben ser homólogos *transformacionales*.[51] Y por otro lado, (*b*) que, dentro de un

[50] Por supuesto que, dado que a veces hay más de un árbol y/o de una asignación óptimas, la determinación del esquema de homologías puede no ser unívoca. Aun así, incluso en esos casos, puede determinarse que algunos caracteres y estados son homólogos,

[51] Para los caracteres tipo ausencia/presencia esto suele ser un poco más dudoso.

carácter, los estados codificados con el mismo número sean homólogos entre sí y no homólogos con estados codificados con un número distinto. A esto a veces se lo denomina homología *táxica*.[52]

(*a*) y (*b*) implican que el esquema de homologías entre los caracteres debe ser conocido para elaborar una matriz de datos. Y ya que esta matriz funciona como *input* para un análisis cladista (el análisis no puede ni siquiera comenzar sin ella), el esquema de homologías debe ser conocido antes de realizar el análisis. Así, la circularidad surge en el marco de la cladística porque el esquema de homologías está a la vez presupuesto (como *input* en la matriz de datos) y es parte del *output* (como esquema de sinapomorfías y simplesiomorfías) del análisis cladista. De ese modo, en el marco de la cladística la circularidad no solo es conceptual sino también operacional. La pregunta es cómo operacionalizar o elaborar una matriz de datos para ser usada en un análisis si la construcción de la matriz presupone el conocimiento de los resultados del análisis mismo.

6.2.3. Una posible solución

Una propuesta de solución a este problema fue popularizada por de Pinna (1991, pp. 372–375) — aunque ya había sido propuesta antes en otros términos (como de Pinna mismo nota), véanse p.e. Lankester (1870), Patterson (1988) y Rieppel (1988). Ella consiste en distinguir dos sentidos o conceptos distintos del término homología, ambos presentes en la cladística. Por un lado, lo que de Pinna llamó homologías *primarias* son las homologías en el sentido clásico, aquellos rasgos que tienen un parecido estructural. La matriz de datos contendría homologías primarias, lo cual evitaría la circularidad porque la determinación de las homologías primarias no presupondría conocimientos sobre la filogenia y la evolución de los caracteres. Contrariamente, su determinación apelaría a los criterios de correspondencia topográfica, composición, etc. usados por los morfólogos (propuestos originalmente por Saint-Hilaire y Owen y sintetizados por Remane, 1952; en el contexto del problema clásico, por fuera de la cladística, autores como Blanco, 2012 proponen soluciones análogas).[53]

[52] La terminología de homología táxica y transformacional se debe a Patterson (1982), y a veces es usada de manera confusa en la literatura.

[53] En el caso de secuencias (de nucleótidos o aminoácidos) se utiliza el alineamiento para el establecimiento de homologías primarias entre bases (Giribet & Wheeler, 1999, p. 132), el cual puede pensarse como basado en la topografía y la composición. Por motivos de pertinencia, no entraré en detalles acerca del alineamiento de secuencias aquí. Para más sobre este punto véase Roffé (en prensa).

Por otro lado, el concepto de homología *secundaria* sería el concepto filogenético de rasgo en dos descendientes heredado a partir de un ancestro común. Lo que una aplicación la cladística permite determinar como parte de su *output* es un esquema de homologías secundarias. Los árboles óptimos encontrados en un análisis cladista suelen contener algo de homoplasia (en aplicaciones reales, ello ocurre prácticamente siempre), lo cual implica que la cladística permite mostrar que algunas homologías primarias (táxicas) no son homologías secundarias —o bien, como nota de Pinna, que una homología primaria de un grupo a veces se divide en homologías secundarias de distintos subgrupos (de Pinna, 1991, p. 374). Es decir, otra razón para pensar que el concepto de homología primaria y secundaria son distintos es que no coinciden extensionalmente.

Nótese que esta solución es semejante a la propuesta por algunos autores para el problema con el concepto de adaptación. Por ejemplo, Ginnobili (2010, 2018, pp. 29–30) distingue dos conceptos de adaptación presentes en el marco de la teoría de la selección natural. Por un lado, está la adaptación como "la perfección de estructura y coadaptación" (i.e. la posesión de rasgos que cumplen funciones de maneras altamente efectivas), el *explanandum* de la teoría, compartido por autores que proponían teorías distintas para dar cuenta de él (como ocurre con las homologías primarias, que son el *explanandum* de la cladística y que eran compartidas por Owen y los morfólogos idealistas). Por otro lado, estaría la adaptación como rasgo cuya presencia se explica por selección natural, es decir, como caso del otro concepto de adaptación que puede ser tratado exitosamente con la teoría de la selección natural (i.e. como ocurre con las homologías secundarias y la cladística). La similaridad entre los dos problemas y las dos soluciones (la circularidad con adaptación y homología) sugiere que la confusión entre un caso del *explanandum* de una teoría y la aplicación teórica bajo la que se lo subsume (i.e. entre un modelo parcial y el correspondiente modelo actual en jerga estructuralista) es recurrente en ciencias empíricas.

La discusión de la solución introducida en esta sección se dio (al menos parcialmente) en el marco de una discusión más amplia sobre el rol de la teoría evolutiva en la cladística. Esta discusión más amplia se dio entre los llamados cladistas de patrón o cladistas transformados y algunos de sus críticos. En la subsección siguiente se introduce en mayor detalle la posición de los cladistas de patrón, para luego, en la subsección siguiente, revisar algunas de las críticas a la solución introducida en esta sección.

6.2.4. El cladismo de patrón

El debate sobre el cladismo de patrón tuvo lugar especialmente a fines de los 70 y durante los años 80 del siglo XX, disminuyendo luego en intensidad (aunque hasta hoy siguen apareciendo publicaciones sobre la temática, véanse por ejemplo Brower, 2019; Pearson, 2010, 2018; Roffé et al., 2018). La posición de los cladistas de patrón era compleja e involucraba muchos puntos distintos, pero el hilo común consistía en sostener, en general, que la teoría evolutiva no jugaba ningún rol (o jugaba un rol disminuido respecto de lo que se creía) en la cladística. Para comprender por qué afirmaban esto considérense dos de las objeciones clásicas que recibió la cladística.

En primer lugar, se encuentra la objeción (ya mencionada en el capítulo 2) de que los taxa fósiles pueden ser ancestros de algún taxón muestreado actual. Sin embargo, la cladística pone siempre a los taxa muestreados (actuales y fósiles) en los nodos terminales, nunca en los nodos internos. En otras palabras, si un taxón fósil A es ancestro de otros dos taxa B y C, la cladística lo situará como el grupo hermano en lugar de como el ancestro de B y C. En segundo lugar, en la cladística los nodos internos se dividen siempre en dos, pero la especiación no necesariamente es dicotómica. Puede ocurrir tanto la especiación por anagénesis (en la que una especie ancestral da lugar a una única especie descendiente) o bien por cladogénesis pero en más de dos especies hijas (una "politomía dura", en contraste a una politomía suave que indica solamente ignorancia del orden de resolución dicotómico).

Respecto de la ancestría efectiva entre taxa muestreados, algunos cladistas defendieron que la relación ancestro-descendiente efectiva es incognoscible (Nelson, 1972). Considérense los siguientes dos posibles árboles para tres taxa muestreados A, B y C (y un ancestro hipotético B'):

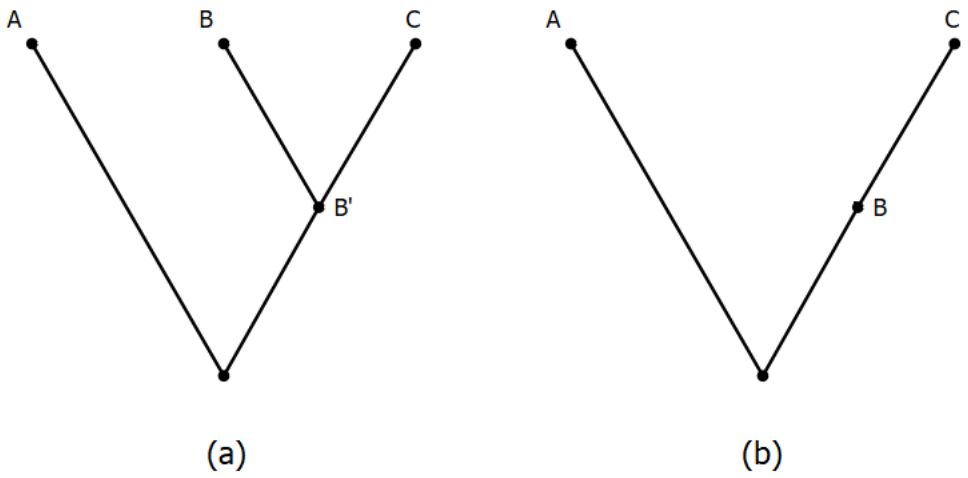


Fig. 6.47. Dos posibles árboles evolutivos. Figura redibujada a partir de la figura 2 de Engelman & Wiley (1977).

En este caso se sostuvo que una autapomorfía (un carácter derivado único) en B puede ser más costosa en el cladograma 2(b) que en el cladograma 2(a), y que por tanto solo se puede situar a un taxón como ancestro de otro en caso de que no posea autapomorfías. Sin embargo, este caso también sería compatible con 2(a). Engelmann y Wiley (1977) y Platnick (1977) caracterizan a esta situación como una en donde la hipótesis de ancestría efectiva puede ser refutada pero no confirmada (o en el vocabulario popperiano que utilizan, corroborada), dado que cualquier caso confirmatorio de 2(a) también lo es de 2(b). En otras palabras, puede darse evidencia en contra de la ancestría efectiva pero no a favor. Del mismo modo, estos autores sostienen que el registro estratigráfico puede refutar una hipótesis de ancestría efectiva pero no confirmarla. Por ejemplo, si B aparece en una capa superior a A en el registro fósil (o si B es un taxón actual) entonces no puede ser el ancestro de A. Sin embargo, si aparece en una capa inferior ello es compatible tanto con 2(a) como con 2(b).

Ahora bien, según estos autores, si la situación (b) de la figura 2 es incognoscible entonces también debe serlo (a) (Hull, 1979; Nelson, 1973). En otras palabras, si las distribuciones de rasgos se ven iguales en ambos casos, y no hay ninguna evidencia independiente que nos permita decidir entre ambas hipótesis, entonces no puede afirmarse que una de las dos relaciones es cognoscible y la otra no. Por tanto, si se interpreta a las líneas de un cladograma como indicando la ancestría efectiva, el requerimiento de que todos los taxa muestreados vayan en las puntas parece problemático.

Existen varias posibles respuestas a este problema. Por un lado, es posible sostener que casi todos los taxa fósiles poseen alguna autapomorfía, y que por tanto en la práctica el problema no surge, ya que todas las hipótesis de ancestría efectiva son menos parsimoniosas (Tattersall & Eldredge, 1977, p. 207).[54] Otra posible respuesta es que las condiciones de fosilización son raras, y que se habrían preservado muy pocas especies ancestrales respecto de la enorme variedad de especies extintas que deben haber existido. Así, la probabilidad de encontrar un fósil que sea un ancestro efectivo de otro taxón sería muy baja (Hull, 1979, p. 429).

Respecto de la especiación por anagénesis, nuevamente, hay diferentes respuestas posibles. Algunos autores apelan a hipótesis sobre la frecuencia con la que habrían ocurrido distintos procesos evolutivos. Por ejemplo, Cracraft (1974) considera que "cualquier intento de reconstruir la filogenia necesariamente refleja la concepción previa del investigador acerca de cómo el proceso evolutivo tuvo lugar" (p. 71). Así, por ejemplo, defiende la dicotomía en los cladogramas sobre la base de la defensa del modelo de especiación alopátrica frente al del gradualismo filético. Otra respuesta tradicional (p.e. Wiley, 1979) apela al concepto de especie de Hennig (1966), según el cual las especies son comunidades reproductivas. En la anagénesis, en la que un mismo linaje cambia fenotípica y genotípicamente pero sin perder su unidad reproductiva no habría, por tanto especiación. La especiación ocurriría solo cuando la unidad reproductiva se pierde, i.e. cuando hay cladogénesis. Así, una especie puede cambiar fenotípicamente sin perder su identidad.[55]

En cambio, según los cladistas de patrón, ambas objeciones surgen de entender mal a la naturaleza y el estatus de la cladística, y las respuestas recién dadas se comprometen *innecesariamente* con asunciones acerca de los procesos evolutivos. Para ellos, tanto el hecho de que los taxa muestreados vayan situados en las hojas del árbol como que los nodos internos se dividan en dos no involucran presuposiciones acerca del proceso evolutivo. Para justificar esto

[54] Incluso una posición más débil, que consistiría en sostener que un ancestro puede hipotetizarse cuando el largo de rama que lleva al taxón en cuestión es corto —i.e. cuando hay pocos cambios entre la optimización del nodo y el taxón (Ramírez, comunicación personal) se encuentra con problemas, ya que asume que la cantidad de cambios en una rama es representativa del tiempo transcurrido. Esto, a la vez, asume que la tasa de cambio es uniforme en toda rama y todo carácter, algo que pocos sistemáticos estarían dispuestos a aceptar. De hecho, los métodos probabilísticos de inferencia filogenética (máxima verosimilitud y análisis bayesiano) modelan este aspecto explícitamente, y ello suele citarse como una ventaja de esos enfoques frente a los enfoques tipo parsimonia.

[55] Estas ideas son compatibles (y fueron una inspiración para) los biólogos y filósofos que sostuvieron la tesis de que las especies son individuos en lugar de clases (Ghiselin, 1974; Hull, 1978) —i.e. que son linajes espaciotemporalmente situados cuyas propiedades pueden cambiar. Para estos autores, el evolucionismo es la tesis de que *las especies mismas* cambian. Si las especies cambian entonces no hay ningún conjunto de características fenotípicas o genotípicas esenciales que las definan. En cambio, el mero hecho de que los descendientes (cercanos o remotos) de un individuo de una especie E_1 pueden pertenecer a una especie distinta E_2 es, según estos autores, compatible con el esencialismo.

introducen una distinción entre cladogramas y árboles evolutivos (la distinción proviene de un manuscrito no publicado de Nelson, cuyas ideas se divulgaron inicialmente en Tattersall & Eldredge, 1977). Los árboles evolutivos reflejan el curso efectivo de la evolución mientras que los cladogramas son en realidad compatibles con muchos árboles evolutivos distintos.

Para ejemplificar, tómese el caso de la figura 2 arriba. Según Nelson (1973), ambos árboles evolutivos serían representados por un mismo cladograma, que se vería como sigue:

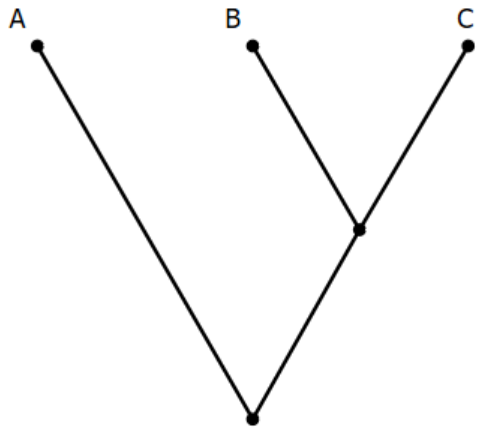


Fig. 6.48. Cladograma para los dos árboles de la figura 2 según Nelson.

Nótese que este cladograma es isomórfico al árbol filogenético de la figura 6.2(a). Sin embargo, según este autor, la interpretación del grafo es diferente. En este caso, el cladograma no representa al curso efectivo de la evolución. Que dos nodos sean flechados por un nodo interno significaría solamente que ese nodo interno es un ancestro de ambos nodos. La figura 6.2(b) se ajusta a esta lectura del cladograma de la figura 6.3, ya que B y C tienen efectivamente un ancestro que no es ancestro de A —marcado con D en la figura 6.4, y A, B y C tienen un ancestro común (la raíz del árbol).

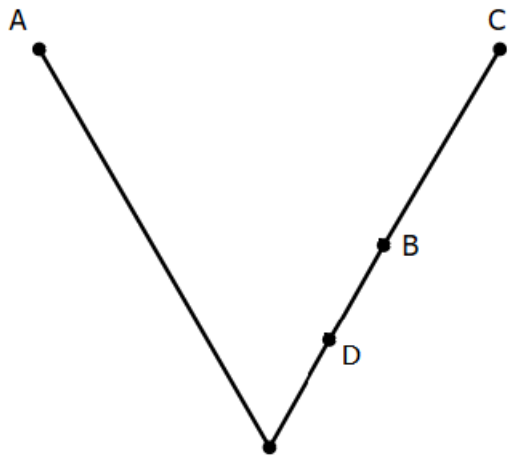


Fig. 6.49. Árbol filogenético 6.2(b) con un ancestro agregado en el linaje que lleva a B. Este ancestro estaría representado en el cladograma de la figura 6.3 por el nodo interno que lleva a B y C.

Así, el *cladograma* de la figura 6.3 es compatible con ambas hipótesis evolutivas de la figura 6.2 —y por tanto es más débil que el *árbol evolutivo* de la figura 6.2(a).[56] Parte de esta confusión habría surgido por confundir a la relación de hermandad entre grupos con el patrón efectivo de diversificación. La primera sería en realidad más débil, e incluiría como una de sus posibilidades a la ancestría efectiva entre taxa del grupo, además de la cladogénesis (Platnick, 1977, p. 439).

En cuanto a la especiación por anagénesis (tratado en Platnick, 1979, pp. 541–542). Supóngase que el curso real de la evolución de un linaje (i.e. el árbol filogenético) es el siguiente (figura 6.5a):

[56] La cantidad de árboles filogenéticos con los que un mismo cladograma sería compatible variaba de autor en autor. Por ejemplo, Wiley (1979) reconoce solo 6 (basándose en su rechazo de la especiación por anagénesis), mientras que Cracraft (1974) identifica 13 y Platnick (1977), 22.

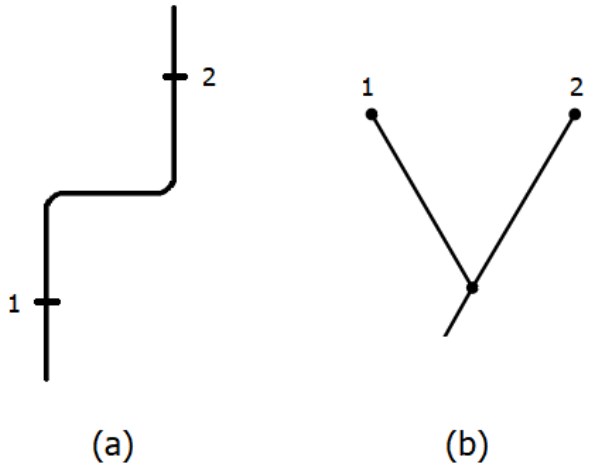


Fig. 6.50. (a) Curso evolutivo con especiación por anagénesis (el eje x representa a la morfología); **(b)** cladograma para el curso evolutivo (a). Redibujada a partir de la figura 2 de Platnick (1979).

en donde los puntos 1- y 2- indican los lugares en los que se muestrearon especímenes. Según Platnick, el resultado de codificar los caracteres y estados de estos especímenes en una matriz y de aplicarles la cladística daría como resultado el cladograma de la figura 6.5b. Nótese que el mismo cladograma surgiría de un árbol evolutivo real que se viera como el de 6.5b. Así, nuevamente, un cladograma sería en realidad compatible con muchos árboles filogenéticos distintos.

Esto no significa, sin embargo, que los cladistas de patrón defiendan la existencia de la especiación por anagénesis. El punto es más bien que llegar al cladograma 6.5b es compatible tanto con defender a la especiación por anagénesis como con sostener el concepto de Hennig de especie (i.e. con sostener que los especímenes muestreados en 1- y 2- pertenecen a la misma especie o a especies distintas). Comprometerse con una u otra posición resulta *innecesario* para la cladística (Platnick, 1979, p. 542).

No comprometerse con un concepto de especie y/o un modelo de especiación tiene una implicancia adicional: los nodos terminales de un cladograma no necesariamente representan especies, ya que dos terminales distintos pueden pertenecer a la misma especie. Del mismo modo, el árbol de la figura 6.6a sería representado como un cladograma del modo de 6.6b:

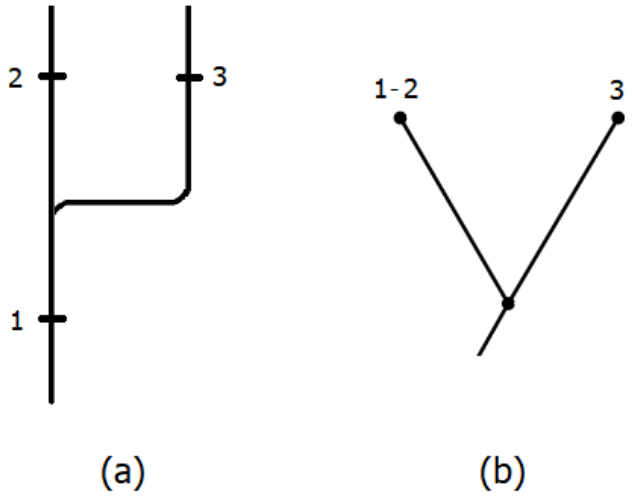


Fig. 6.51. (a) Árbol filogenético con especiación por cladogénesis, pero en donde uno de los linajes sucesores se mantiene morfológicamente idéntico; **(a)** Cladograma para el árbol 6.6a. Figura adaptada a partir de la figura 2 de Platnick (1979).

dado que los especímenes 1- y 2- serían morfológicamente idénticos, en desconocimiento de la historia evolutiva real, se los identificaría como el mismo terminal. De ese modo, e inversamente a lo anterior, un mismo terminal puede en realidad contener a individuos de especies distintas bajo el criterio Hennigiano de especie. Todo esto implica que los nodos no representan especies (actuales y ancestrales) sino meramente a conjuntos de rasgos, y por tanto los cladogramas ilustran solamente a la evolución de los rasgos, y no de las especies. La historia evolutiva de los rasgos sería compatible con (y más fácilmente cognoscible que) las distintas historias evolutivas para las especies.

Por otro lado, dejando de lado el modo en que se *interpreta* un cladograma (lo que serían los axiomas de interpretación de mi formalismo), la *construcción* del conjunto de los cladogramas óptimos a partir de la matriz de datos (i.e. el modo matemático de optimizar caracteres y de medir el largo, i.e. la formulación puramente formal de los axiomas impropios) sería idéntica. Por este motivo Platnick afirma que todo estos principios representan una transformación limitada en la cladística, ya que "los métodos [de Hennig] para analizar datos y construir clasificaciones a partir de ellos se mantienen esencialmente inalterados. Como mucho, lo que cambia es el modo en que esas conclusiones son justificadas" (Platnick, 1979, p. 538, traducción propia). En realidad, teniendo en cuenta a la teoría completa y no solo a la definición del concepto *AO* en mi reconstrucción (axioma 17, la caracterización de los árboles óptimos como los de menor $largo_2$), la

posición recién delineada sí implica algunos cambios. Para comenzar, si se permite que el árbol real sea algo de la forma de la figura 5a, entonces el axioma impropio (5), que afirma que $A_R \in A$ (el árbol real está entre los cladogramas), debe reemplazarse por uno más débil, que permita ese tipo de historia filogenética (i.e. A_R será simplemente un grafo dirigido acíclico y conectado, que contiene a *T* entre sus nodos). Adicionalmente, la ley fundamental (2) debe debilitarse, ya que el árbol real no necesariamente estará entre los cladogramas óptimos sino que tiene que ser *compatible* con alguno de ellos (en donde habría que definir precisamente la noción de compatibilidad). Esto implica, contra Platnick, que la posición de los cladistas de patrón sí tiene algunas implicancias metodológicas, p.e. en los modos en los que se puede testear la cladística (sección 3.3.4).

Cabe notar, sin embargo, que algunos cladistas de patrón fueron aún más lejos que esto. No solo consideraron que la aserción fáctica central implícita en la teoría era más débil, sino que directamente no la consideraban parte de la teoría. La cladística no sería más que un método para detectar patrones en la naturaleza (esquemas anidados de sinapomorfías) —una suerte de Orden natural que los sistemáticos pre-darwinianos también habrían estudiado.[57] Por ejemplo, Platnick (1979, p. 28) establece "el primer principio de la cladística: que la naturaleza está ordenada en un único patrón especificable. Admito que esto no es una teoría científica; no puede ser testeada". Por supuesto, ese patrón sería *luego* explicado en base a la evolución (de caracteres o especies), pero la explicación misma no formaría parte de la cladística. En otras palabras, esto equivaldría a tomar mi reconstrucción de la cladística y quitarle las leyes fundamentales. La cladística no sería más que un modo de determinar al conjunto *AO*, i.e. de elaborar un conjunto de grafos que ilustran la inclusión anidada de rasgos, sin hacer ninguna afirmación fáctica acerca de cómo se relaciona esta distribución con algún proceso histórico. Esto haría que la cladística deje de ser una teoría empírica, ya que dejaría de hacer afirmaciones fácticas. Quizás esta discusión sea más bien terminológica y se pueda resolver reconociendo una cladística en sentido restringido (la definición de cladograma

[57] De todos modos, dado que una sinapomorfía es un carácter *derivado* compartido, no está claro si la afirmación de que la cladística detecta patrones anidados de sinapomorfías está libre (y/o si pretende estar libre) de teoría evolutiva. Ello depende, por supuesto de cómo se interprete el concepto de "rasgo derivado". La interpretación estándar sería la de un rasgo que surge evolutivamente de otro, con lo cual esta afirmación meramente reiteraría que la cladística se ocupa de la evolución de rasgos y no de taxa. En cambio, y en contra de lo que afirma Platnick, parecería que una interpretación predarwiniana (derivado como avanzado/complejo) no cuadra del todo bien con la cladística, ya que una sinapomorfía puede consistir en la simplificación de un rasgo. De hecho, los sistemáticos pre-darwinianos (e incluso algunos darwinianos, como la escuela evolucionista) no distinguían entre sinapomorfías y simplesiomorfías a la hora de agrupar taxa —por eso reconocían taxa (como Aves o Reptilia) que, desde un punto de vista Hennigiano, no son monofiléticos.

óptimo) y una en sentido amplio (la teoría, que sí hace una aserción fáctica adicional acerca de ellos).

La posición de los cladistas de patrón involucraba algunas sutilezas y tesis adicionales, particularmente en torno a la relación entre clasificación e inferencia filogenética, en las que no entraré aquí. En lo que sigue me centraré en su posición con respecto a la noción de homología, y en algunas críticas que esta posición recibió. Cabe notar, antes de avanzar, que a lo largo de la presente sección la reconstrucción ya fue utilizada para clarificar la posición general de los cladistas de patrón y las implicancias que esta tiene, en términos de los elementos de la reconstrucción que habría que modificar si ella fuese adecuada.

6.2.5. El cladismo de patrón y la noción de homología

Dado el espíritu de la propuesta de los cladistas de patrón, que consiste (en general) en restarle importancia a la teoría evolutiva en la identificación del conjunto de los cladogramas óptimos, es fácil ver que su posición con respecto a las homologías se alinea mejor con un concepto clásico de homología. Como afirma Pearson:

> *Los cladistas de patrón tienden a enfatizar una concepción tradicional sobre las homologías, solidificada en el trabajo de Richard Owen, quien caracterizó a los homólogos como "el mismo rasgo bajo una variedad de formas y funciones". En la base de la caracterización tradicional de homología está la noción de rasgo tipo, donde ese tipo es identificado por una correspondencia morfológica entre sus diferentes instanciaciones.* (Pearson, *2010*, pp. 483-484, traducción propia).

Por ejemplo, Brady (otro filósofo que participó del debate por el cladismo de patrón, en este caso a favor de tal posición) se basó en la diferencia entre explicar y definir para sostener que la ancestría común hace lo primero pero no lo segundo con respecto a las homologías (a las cuales entendía de modo Oweniano). Según Brady (1985), Darwin habría tenido esto en claro al utilizar el concepto de homología igual que sus predecesores (sin cambios sustanciales en su significado) y afirmando solo que su teoría las explica mejor. De hecho, esta base empírica común habría sido el motivo por el que la propuesta darwiniana en sistemática tuvo tanto éxito entre sus contemporáneos.

Más en general, Brady defiende que la detección de patrones es previa (histórica y lógicamente) a la explicación causal de esos patrones:

> *Nosotros no postulamos, o no deberíamos postular, explicaciones o teorías sobre procesos antes de descubrir un orden particular en las apariencias al cual la teoría se dirige. Después de todo, no percibimos la causalidad de modo directo en ningún sentido, pero sí percibimos a los efectos, y no tenemos otra guía para la operación de ningún poder causal hasta que encontramos una regularidad o patrón entre esos efectos. Ciertamente, parecería que es nuestra habilidad de encontrar lo general entre lo particular lo que nos motiva a buscar una causa generativa en primer lugar. (...) Si perdemos la distinción entre la detección de un patrón y su explicación por una hipótesis sobre procesos, perdemos la razón para nuestras investigaciones, no solo histórica sino también lógicamente.* (Brady, 1985, p. 125, traducción propia).

En cambio, Mayr y los autores que definen a las homologías a partir de la ancestría confunden al *explanandum* con el *explanans*, una estrategia que Brady consideró *self-defeating* (1985, p. 117) y como recomendando que "la observación sea un artefacto de la propia teoría" (p. 124). Por otro lado, respecto de los ataques a los cladistas de patrón (e.g. por parte de Beatty, 1982) agrega que estos autores son justamente los que están intentando retener el estatus empírico de la sistemática al basar la sistemática en "los patrones derivados de la observación (en lugar de sus correspondientes explicaciones)" (Brady, 1985, p. 125).

Si bien la posición de Brady parece razonable, y encaja bien con la distinción entre homología primaria y secundaria introducida en 6.2.3 (aunque este autor no consideraría a las segundas homologías), el lenguaje metateórico inadecuado que utilizó para defender su posición lo convirtió en blanco de críticas por parte de otros autores —y esto no es un caso aislado sino que es representativo de lo que ocurrió en el debate por el cladismo de patrón con otros autores.[58] Por

[58] En el caso de Brady en particular, parece ser que sí defendía algo semejante a la distinción teórico-observacional (véase Ebach & Williams, 2019). Otros cladistas de patrón estaban menos comprometidos con esta distinción, p.e. véase la cita de Platnick abajo. Por otro lado, cabe aclarar que el uso de lenguaje metateórico inadecuado no fue el único motivo por el que surgieron estos malentendidos. En los 70s y 80s la disputa con la escuela fenetista estaba en su punto álgido (Hull, 1988), con lo cual cualquier cosa que sonara (aunque sea ligeramente) a las tesis fenetistas iba a ser atacado. Adicionalmente, en ese momento, los creacionistas tomaron fuera de contexto algunas afirmaciones de los cladistas de patrón para argumentar que muchos biólogos contemporáneos eran anti-evolucionistas (lo cual, por supuesto, no era cierto). Nada de esto, sin embargo, explica críticas como la de Pearson (2010), formuladas mucho tiempo después.

ejemplo, tanto Hull como Pearson interpretaron a los cladistas de patrón como sosteniendo algo semejante a la distinción teórico-observacional de los empiristas lógicos, y como intentando basar a la sistemática en enunciados puramente observacionales acerca de la similaridad entre rasgos —y de ahí a la acusación de fenetistas había solo un paso (p.e. Hull 1988, pp. 236, 239). Por ejemplo, Pearson afirma que:

> *Para los cladistas de patrón, el error de convertir al concepto de homología en un concepto histórico yace en intercambiar una base empírica independiente de teoría por una circular, para la clasificación. La circularidad surge porque se supone que la homología es evidencia para ciertas relaciones evolutivas entre taxa. Pero si la homología incluye conceptualmente a la relación histórica entre taxa y sus rasgos, la evidencia que la homología provee para las relaciones evolutivas es circular (…). No se puede culpar a los cladistas de patrón por su objetivo de evitar la circularidad en sus sistemas de clasificación, y su apelación a la neutralidad teórica es ciertamente entendible. Sin embargo, la concepción de la práctica científica como siendo neutral con respecto la teoría es, en el mejor de los casos, una visión minoritaria entre los filósofos de la ciencia contemporáneos. Para apreciar este punto, vale la pena explicitar que la visión de la ciencia de los cladistas de patrón encaja perfectamente con los elementos centrales de las caracterizaciones positivista y post-positivista tempranas de la ciencia.* (Pearson, 2010, pp. 484-485, traducción propia).

En contra de esta supuesta visión de la ciencia basada en la concepción clásica de teoría, Pearson expone los (en este punto también clásicos) argumentos Hansonianos en contra de la distinción teórico-observacional (Hanson, 1958). Así, según Pearson, el punto de Brady de que definir a las homologías basándose en la ancestría haría que "la observación sea un artefacto de la propia teoría" no constituiría un problema, sino que sería parte del normal funcionamiento de la ciencia. Dado que la observación está cargada de teoría, no hay una base firme y sólida de enunciados (morfológicos, en este caso) puramente observacionales sobre los cuales edificar el resto del conocimiento científico (sistemático) —lo cual es lo que supuestamente pretendían hacer los cladistas de patrón en su discusión de las homologías. En cambio, la posición hansoniana sería más cercana a la idea de Hennig (1966) de la iluminación recíproca, según la cual avances en un área arrojan luz sobre otra, la cual puede volver a arrojar luz sobre la inicial, creando un círculo virtuoso.

Para ilustrar este punto con un caso concreto, Pearson considera un artículo de Smith y Wheeler (2006) en el que los autores habrían utilizado información filogenética para determinar que una estructura encontrada en un conjunto de peces era homóloga a una glándula de veneno en otros peces, al momento de codificar la matriz de datos.

En la siguiente subsección se profundiza sobre la cuestión de la relación entre teoría y observación, y se defiende a la posición de los cladistas de patrón (entre otros, incluyendo a quienes sostienen la distinción entre homología primaria y secundaria) de este tipo de críticas.

6.2.5. Respuesta a las críticas: el estatus de T-teoricidad de las homologías primarias

Como se sugirió en la sección anterior, parte de las críticas que recibieron los cladistas de patrón —y más en general, la posición que sostiene que las afirmaciones (i) y (ii) de la sección 6.2.1 no pueden ser ambas verdaderas a la vez— estuvieron basadas en el lenguaje metateórico inadecuado que estos autores utilizaron para exponer sus ideas. En esta subsección se muestra cómo pueden reformularse los argumentos de estos autores usando un lenguaje más apropiado, evitando así objeciones como las de Pearson y Hull. Me basaré para ello en la distinción estructuralista de T-teoricidad y en la reconstrucción presentada en el capítulo 3. Parte de los resultados de esta subsección se encuentran ya publicados en Roffé, Ginnobili & Blanco (2018), aquí se amplía y elabora sobre ellos.

Como se explicó en el capítulo 3, y se retomó en la sección anterior, la distinción de T-teoricidad del estructuralismo deja de lado la cuestión de la observabilidad, centrándose en cambio en los modos en los que pueden determinarse los conceptos. Un concepto será T-teórico cuando todos sus métodos de determinación presuponen T, y T-no-teórico cuando tenga algún método de determinación que no presuponga T. Los conceptos de la "base empírica" de una teoría T son precisamente sus conceptos T-no-teóricos. Esto se debe a que son aquellos para los cuales se puede obtener una determinación (medición, si el concepto es cuantitativo) independiente de la propia teoría, la cual puede compararse con la determinación teórica del mismo concepto para establecer si la teoría hizo la predicción correcta.

La posición de Brady de que el *explanandum* de una teoría no puede estar definido en la propia teoría (i.e. la teoría para la cual es *explanandum* no puede dar condiciones suficientes *y*

necesarias para su determinación), bajo pena de circularidad, puede expresarse en un vocabulario estructuralista diciendo que el *explanandum* de las teorías es formulado utilizando conceptos no-teóricos para las propias teorías. Formalmente, esto está expresado en el aparato estructuralista en el hecho de que el conjunto **I** (de las aplicaciones intencionales) es un subconjunto de $\mathbf{M_{pp}}$ (los modelos parciales, las subestructuras de la teoría que no contienen conceptos T-teóricos). Así, la posición de los cladistas de patrón en torno a las homologías puede reformularse diciendo que el concepto de homología (homología primaria en los términos que utiliza de Pinna), que constituye el *explanandum* de la cladística, es **CLAD**-no-teórico.

Ahora bien, la distribución de homologías (primarias) a ser explicada es aquella presente en la matriz de datos. En la reconstrucción formal del capítulo 3, la matriz está representada por los conceptos T, C y d_T, todos los cuales, como se dijo, son efectivamente T-no-teóricos. Es obvio que el conjunto de taxa elegidos es previo e independiente al análisis filogenético que se haga de ellos. En el caso de los caracteres y asignaciones a terminales (C y d_T), para morfología, se suelen usar los criterios clásicos Owenianos (topografía, composición, formas intermedias, etc.), mientras que para datos moleculares se utiliza el alineamiento múltiple de secuencias. Ninguno de estos procedimientos de determinación presupone un análisis filogenético previo, y todos pueden realizarse con especies nuevas, para las cuales la filogenia previa no existe.

En conclusión, la idea de que la cladística evita la circularidad porque su *input* (una distribución de homologías) es independiente de la teoría evolutiva puede formularse más precisamente afirmando que ella es T-no-teórica, en lugar de apelar a la terminología referida a la observabilidad. Dado que la determinación T-no-teórica de un concepto puede presuponer otras teorías (que no son T), se evitan las críticas de corte Hansoniano sobre la inexistencia de conceptos puramente observacionales.

El propio Pearson parece reconocer este punto, al afirmar:

> *[Otro potencial malentendido] concierne al posible rechazo del cladismo de patrón sobre la base de una afirmación poco especifica de que la observación siempre está cargada de teoría. (…) Esta idea es superficial hasta el punto de convertir al cladismo de patrón en un hombre de paja. Los cladistas de patrón no necesitan sentirse amenazados por argumentos que muestren que la observación está cargada de teoría, dado que no es la teoría en general lo que buscan purgar de la taxonomía cladística, sino sólo la teoría evolutiva. (…) Con respecto a la cuestión central de la homología, los cladistas de patrón probablemente*

> *aceptarán la idea de que las observaciones están informadas por consideraciones teóricas provenientes de, por ejemplo, la morfología funcional. Los patrones en la naturaleza serán reconocidos como patrones sólo si el observador está armado con la teoría relevante para reconocerlos como patrones. (…) Lo que está en juego, entonces, no es si las observaciones acerca de patrones que determinan a las homologías están cargadas de teoría, sino sólo si las observaciones de homologías están inextricablemente ligadas a la teoría evolutiva en particular.* (Pearson, 2010, p. 486)

Sin embargo, aun reconociendo esto, Pearson intenta sostener su punto anterior (que el concepto de homología primaria está cargado de teoría evolutiva) afirmando que en ocasiones *sí es la teoría evolutiva* la que permite a los sistemáticos reconocer homologías en la construcción de la matriz de datos. Mostrar esto es precisamente la función que cumple el caso Smith y Wheeler en el argumento de Pearson.

En mi reformulación de la discusión, el punto de Pearson sería que si bien se argumentó que las homologías primarias son **CLAD**-no-teóricas —y por lo tanto que pueden determinarse sin presuponer la propia cladística— no se mostró que *no* puedan determinarse presuponiendo esta teoría. Es decir, Pearson continúa sosteniendo que el concepto de homología primaria está cargado de teoría evolutiva porque puede usarse a la cladística para determinarlo.

Sin embargo, en estos términos, es claro que este punto es irrelevante. Cuando se dice que los conceptos de la base empírica de una teoría T están "libres de T", no es porque *no pueda* determinárselos *usando T* (esto suele ser posible, y es precisamente lo que ocurre en una contrastación de la teoría, véase arriba) sino porque *puede* determinárselos *sin usar T*. En otras palabras, los conceptos que *no* están libres de T son los conceptos T-teóricos, ya que *todas* sus determinaciones presuponen hacer uso de T. En los términos que se introdujeron en la sección 6.1, lo que importa es la T-teoricidad y no la T-determinabilidad.

Para ilustrar este punto considérese una discusión imaginaria entre dos grupos de físicos clásicos, llámense "mecánicos mecanicistas" y "mecánicos de patrón" (Roffé et al., 2018, p. 7). Los mecanicistas de patrón argumentan que el concepto de aceleración está libre de la mecánica clásica, mientras que los mecánicos mecanicistas afirman que esto no es el caso. El segundo grupo presenta como argumento contra el primero un caso en el que las aceleraciones son determinadas usando las leyes de la mecánica clásica. Sería claro en este caso que tal argumento es inadecuado. Si la posición de los mecánicos de patrón tenía algún sentido, la afirmación de que las aceleraciones

están libres de mecánica clásica no era acerca de la *imposibilidad* de determinarlas *con* los recursos de esa teoría, sino acerca de la *posibilidad* de hacerlo *sin* esos recursos. El hecho de que esto pueda hacerse es lo que permite a la mecánica clásica explicar las aceleraciones de partículas de un modo no circular. Si el objetivo de argumentar que las aceleraciones están libres de teoría era mostrar cómo la mecánica clásica las explica de manera no-circular, entonces la segunda afirmación (la más débil) es suficiente. Del mismo modo, el hecho de que las distribuciones de homologías primarias puedan determinarse independientemente de la filogenética es lo que permite a esta última explicar la distribución "observada" de homologías sin caer en circularidad.

Las consideraciones precedentes son suficientes para dar una respuesta conceptual al punto que intenta establecer Pearson. Sin embargo, puede decirse algo más, ya que el caso que este autor toma como ejemplo no muestra lo que el autor pretende. Es decir, ni siquiera es cierto que en el artículo de Smith y Wheeler el conocimiento previo de la filogenia haya sido el factor determinante para establecer que el tejido mencionado era efectivamente una glándula de veneno. En cambio, estos autores afirman que:

> *Nuestro examen de seis especies de estos tres géneros indica que los surcos anterolaterales están presentes en las seis especies, pero faltan las glándulas venenosas conspicuas asociadas a estos surcos. Sin embargo, el margen caudal de sus espinas de la aleta tiene un tejido glandular conspicuo (…) que difiere significativamente del típico tejido muscular que se encuentra en el margen posterior de las espinas en la mayoría de los peces no venenosos (…). Identificamos tentativamente esta estructura como una glándula de veneno, en espera de un estudio más profundo. (Smith & Wheeler, 2006, p. 213)*

Es decir, en última instancia, los investigadores usaron el criterio clásico de *composición* para establecer la homología.

En su respuesta a Roffé, Ginnobili y Blanco, Pearson (2018) concede a estos autores que las homologías primarias pueden determinarse independientemente de la cladística, y que ello permite a la teoría explicarlas de manera no-circular (en el plano operacional al menos). Sin embargo, este autor introduce una distinción entre el significado de un concepto y sus métodos de determinación. Así, según él, lo que explica a un concepto (o a la atribución de ese concepto a un conjunto de objetos) puede formar parte del significado del concepto mismo (Pearson, 2018, p. 3). En ese sentido, sostiene que su objetivo era mostrar que los cladistas de patrón no deberían sentirse

amenazados por el cambio conceptual que tuvo lugar en el siglo XX (y que toma como un hecho) de pasar a considerar que las homologías están definidas a partir de la ancestría (aquello que las explica) — en sus palabras, que "no existe una motivación racional" para rechazar este cambio.

Hay varios problemas con esta respuesta. En primer lugar, asume como punto de partida el hecho de que hay un único concepto de homología definido en términos filogenéticos, siendo que eso es justamente parte de lo que estaba en discusión en el debate. Es decir, no está claro que sean los cladistas de patrón quienes tengan que ofrecer una justificación para no adoptar un único concepto filogenético de homología, y no sus contrincantes para adoptarlo.

En segundo lugar, y más sustancialmente, hay un punto en el que acordamos: el significado es un fenómeno complejo, e introducir una teoría nueva puede modificar el significado de los conceptos de su *explanandum* —p.e. porque al agregarles métodos de determinación T-teóricos modifica sus sentidos. En otras palabras, no se niega que alguna forma de holismo (moderado) del significado ocurra. Sin embargo, parece haber aún una "motivación racional" para sostener que el concepto de homología primaria está libre de ancestría *en el mismo sentido en que el concepto de aceleración está libre de fuerzas*. Por supuesto, también ahí uno podría apelar al holismo y sostener que el concepto de aceleración no está libre de fuerzas, dado que la mecánica clásica agrega métodos de determinación para la aceleración que utilizan el concepto de fuerza. En cambio, si lo que Pearson pretende es sostener una posición holista más extrema, en la que ningún concepto está libre de ningún otro concepto, entonces el punto parecería poco interesante y ciertamente no sería lo que estaba siendo discutido en el caso del cladismo de patrón.

Por último, si bien los métodos de determinación para un concepto pueden no agotar su significado, está claro que una definición de un concepto sí brinda condiciones necesarias y suficientes para la determinación de los conceptos que definen. Pero en el presente caso la definición filogenética de homología no da condiciones necesarias para la inclusión de un conjunto de rasgos en una matriz de datos cladista. Si ese fuese el caso (si todo estado compartido de un carácter en una matriz de datos fuese derivado de un ancestro común), entonces el *output* de un análisis cladista no debería contener homoplasia. Pero dado que esto suele ser el caso, tal requisito no es una condición necesaria. En otras palabras, los conceptos de homología primaria y secundaria reconocidos por de Pinna no pueden colapsarse en un único concepto, definido del mismo modo, ya que sus extensiones difieren. En conclusión, independientemente de cómo se caracterice el fenómeno complejo del significado, y del rol que jueguen los métodos de determinación en tal

elucidación, parece claro que los cladistas de patrón tienen razón al afirmar que la noción de homología usada en la construcción de la matriz de datos no está definida como "rasgos derivados de un ancestro común" (homología secundaria en de Pinna).

6.2.6. Conclusiones

En esta sección se extrajeron algunas consecuencias del aparato metateórico estructuralista y de su aplicación a la cladística, tratados en el capítulo 3 de la presente tesis, a la discusión sobre la circularidad en el concepto de homología en el marco de la cladística. Para ello, en primer lugar, se introdujeron el problema clásico de la circularidad y su reaparición en el marco de la cladística en las subsecciones 6.2.1 y 6.2.2, así como una de las soluciones estándar (la distinción entre homología primaria y secundaria) en 6.2.3. Luego, se situó a la discusión acerca de las homologías en el contexto más amplio de la discusión por el cladismo de patrón, y se examinaron algunas críticas a la posición de los cladistas de patrón en torno a las homologías. Se mostró que estas críticas estaban al menos parcialmente motivadas por el lenguaje metateórico inadecuado que esos autores utilizaron para expresar su posición. Se argumentó que las objeciones de corte Hansoniano (desarrolladas más extensamente por Pearson y Hull), que atacaban a la noción de homología en la matriz de datos (i.e. homología primaria) como presuponiendo a la distinción teórico-observacional pueden evitarse si se apela a la noción de T-teoricidad del estructuralismo metateórico. Adicionalmente, se brindó apoyo a la tesis de que las homologías (primarias) son efectivamente **CLAD**-no-teóricas sobre la base de la reconstrucción expuesta en el capítulo 3. Por último, se consideró la respuesta de Pearson a estos puntos, concluyendo que no es adecuada y por tanto que no altera el punto.

Capítulo 7. Conclusiones

En el presente capítulo se realizan dos tareas. En primer lugar, se hace un repaso de los resultados obtenidos a lo largo de la tesis, a fin de ofrecer una visión sinóptica o de conjunto de lo desarrollado a lo largo de los capítulos anteriores. La segunda parte del capítulo menciona algunos problemas abiertos y/o caminos por donde se puede continuar la presente investigación.

7.1. Resumen de los resultados obtenidos

En el primer capítulo de la presente tesis se introdujo el marco general y los objetivos centrales de la tesis. Estos consistieron principalmente en los siguientes dos:

> (*i*) Proveer herramientas computacionales que permitan testear reconstrucciones estructuralistas de teorías. Esto a la vez tenía como objetivo último permitir contrastar a la metateoría misma —ya que toda teoría (incluyendo las teorías acerca de las teorías) se contrastan a través de sus aplicaciones. Esto haría, por otra parte, que la metodología de trabajo en metateoría se acerque al modo usual en el que se trabaja en ciencias empíricas.
> (*ii*) El segundo objetivo consistió en presentar una reconstrucción estructuralista de la cladística. Esto es relevante para la metateoría, al permitir ampliar su dominio de aplicaciones exitosas a áreas como la sistemática, que no habían sido exploradas formalmente desde el estructuralismo (y que fueron relativamente menos tratadas en la filosofía de la biología en general). Por otro lado, proveer esta reconstrucción tenía como objetivo contribuir a resolver debates conceptuales al interior de la sistemática, como ser, la cuestión de la circularidad en torno al concepto de homología.

La parte I de la tesis lidió principalmente con el objetivo (*ii*). En el capítulo 2 se introdujo la cladística. Para volver más didáctica la exposición, se comenzó introduciendo algunos de los antecedentes de esta teoría. En primer lugar, se consideró la explicación darwiniana de las homologías a partir de la ancestría común y las implicancias que ello tenía para ese autor en el terreno de la taxonomía. Se presentó además una periodización y una caracterización breve del

contexto en el que surgió la cladística, así como de las diferencias entre esta escuela y sus principales escuelas opositoras (el fenetismo y el evolucionismo) —distinguiendo entre componentes taxonómicos y filogenéticos (i.e. cladonomía y cladística). Esta introducción histórica me permitió presentar un modo más claro que el usual el contenido de la teoría y sus conceptos centrales, en el resto del capítulo.

En el capítulo 3 se presentó la reconstrucción estructuralista de la teoría introducida en el capítulo anterior. Para ello, se repasaron en primer lugar otros tratamientos formales previos de la cladística, mencionando la literatura previa sobre la que la reconstrucción presentada en el capítulo edifica. La sección 3.2 introdujo la noción de axioma impropio (y modelo potencial) en el estructuralismo y presentó los (numerosos) axiomas impropios de la cladística, mostrando cómo es posible formalizar cada uno de los conceptos introducidos en el capítulo anterior. En la sección 3.3 se introdujo la noción de ley fundamental estructuralista, así como las leyes fundamentales de la cladística. Esta presentación me permitió dar una discusión acerca del estatus fáctico de la cladística. Se defendió que las leyes fundamentales presentadas son leyes fácticas, que poseen el mismo estatus (fáctico) que cualquier otra ley fundamental de cualquier otra teoría. Es decir, que se la acepta o rechaza por motivos empíricos, según el éxito aplicativo que tenga. Se sostuvo que a pesar de que en la mayoría de los casos reales de aplicación no hay posibilidad de contrastación (ya que los eventos evolutivos siendo caracterizados ocurrieron en un pasado remoto), hay posibilidad de contrastación en algunos casos, por ejemplo, aquellos que involucran filogenias experimentales. La sección 3.4 formalizó algunos conceptos adicionales (grupo monofilético, tipos de morfismos entre estados) que son ampliamente usados y resulta valioso elucidar, a pesar de que no juegan un rol en las aserciones fácticas centrales de la teoría. En 3.5 se introdujeron las nociones estructuralistas de especialización, T-teoricidad y condición de ligadura, y se las intentó aplicar a la reconstrucción de la cladística. Se sostuvo, así, que la cladística no tiene especializaciones ni conceptos T-teóricos, y se reconocieron dos condiciones de ligadura (una de las cuales se relaciona con el método de enraizamiento de árboles presentado en el capítulo 2). Por último, en 3.6, se expuso la elucidación estándar de la noción de método de determinación y se presentó un modo de optimizar caracteres sobre un árbol que parecería ser la contraparte informal de un método de determinación de la cladística (y que es reconstruido luego, en el capítulo 5).

La parte II de la tesis vuelve sobre el objetivo (*i*), introduciendo los procedimientos y la herramienta computacional para contrastar reconstrucciones (capítulo 4) y ejemplificando su uso con una reconstrucción de una teoría real (la cladística, capítulo 5).

El primero de estos capítulos presenta conjuntamente los procedimientos de testeo y la herramienta computacional para llevarlos a cabo: el programa Reconstructor. Para ello se procedió mostrando, en primer lugar, cómo cargar el lenguaje, los axiomas impropios, leyes y especializaciones de una reconstrucción (i.e. la sintaxis aceptable para oraciones del lenguaje formal del programa), utilizando para ello una reconstrucción de una versión simplificada de la mecánica de Descartes (MCAR, la cual es usada a lo largo de todo el capítulo para ilustrar las diversas funcionalidades de Reconstructor). Se mostró cómo consultar al programa si una cadena de símbolos es un término, una fórmula o una oración bien formadas del lenguaje de MCAR. Se expuso a continuación el modo de cargar modelos (completos e incompletos), y en base a ello el modo de pedir la denotación de un término y el valor de verdad de una oración formal en alguno de esos modelos. A partir de estas funcionalidades se introdujo el primer método de contrastación de reconstrucciones: la coincidencia en el comportamiento (p.e. en el valor de verdad) entre una parte de la reconstrucción formal y su contraparte informal. En el caso de las leyes y los modelos, esto significa que los modelos que representan a aplicaciones paradigmáticamente exitosas de la teoría objeto (i.e. que informalmente se considera que cumplen las leyes) deben satisfacer la formalización de esas leyes, y viceversa para las no-exitosas. Se argumentó que una contrastación exhaustiva utilizando este método debería introducir aplicaciones (modelos) que informalmente satisfagan diferentes combinaciones de leyes. Se mostró además cómo visualizar los procesos de razonamiento internos del programa, a fin de facilitar la detección de errores en la formalización de las leyes y/o en la carga de modelos (nuevamente, ejemplificando con MCAR). La sección 4.3. realizó todo lo anterior con las condiciones de ligadura, mientras que 4.4 especificó la semántica paracompleta que Reconstructor utiliza para evaluar oraciones en modelos que contienen fallas de denotación, incluyendo la semántica del operador °, el cual puede ser utilizado para "recapturar" la semántica clásica.

Las secciones 4.5 y 4.6 presentaron el novedoso modo en que Reconstructor trata a los métodos de determinación (al cual se da sustento teórico en el capítulo 6). 4.5 lidia con los métodos de determinación manuales, en los que un procedimiento de determinación de un concepto es cargado a mano por el usuario como un algoritmo, en el lenguaje de programación Python. Se

mostró a continuación cómo es posible ejecutar un método en un modelo, a fin de completar y/o modificar las extensiones de los conceptos de ese modelo. Lo precedente me permitió proponer un segundo método de contrastación de reconstrucciones, basado en un criterio de coherencia interna: los modelos completados usando métodos de determinación manuales deben satisfacer la formalización de las leyes. Por último, en esta sección, se expusieron algunas funcionalidades adicionales del programa, que permiten cargar métodos T-no-teóricos (con input adicional) y métodos basados en condiciones de ligadura. En la sección 4.6 se introdujeron los métodos de determinación automáticos, especialmente útiles para aquellos usuarios del programa que no tienen manejo de lenguajes de programación. Estos métodos infieren las denotaciones de los conceptos de la teoría a partir de las leyes y la información previa disponible en el modelo. Para ello, se expuso cómo Reconstructor implementa una generalización (propia) del aparato kripkeano de puntos fijos, que permite imponer oraciones en modelos distintas al esquema-T, y así completar las extensiones de conceptos distintos al predicado de verdad. Un tercer método de contrastación de reconstrucciones se basa en la idea de que las extensiones de modelos producidas por métodos automáticos deben ser conservativas.

El capítulo 5 ilustra cómo los dos objetivos centrales de la tesis se vinculan. En este capítulo la reconstrucción de la cladística presentada en el capítulo 3 (y relacionada principalmente con el objetivo (*ii*)) fue contrastada, utilizándola como caso de aplicación del programa Reconstructor (el principal aporte a (*i*)). Esto me permitió a la vez defender la adecuación de la reconstrucción presentada, así como ilustrar la fertilidad y aplicabilidad del programa a casos reales. El capítulo comenzó brindando las traducciones de los axiomas (impropios y propios) desde el lenguaje formal del capítulo 3 al lenguaje de Reconstructor. El lenguaje de Reconstructor está diseñado para ser lo más parecido posible al lenguaje formal estándar, de modo que la mayoría de las traducciones son sencillas de realizar. Se comentaron, sin embargo, los casos en donde fue necesario realizar alguna modificación, sea porque el lenguaje formal estándar contiene atajos o por motivos de computabilidad. A continuación, se dieron las cuatro aplicaciones a ser utilizadas para llevar a cabo los procedimientos de contrastación introducidos en el capítulo 4. Estas cuatro aplicaciones son tales que (informalmente) satisfacen distintos conjuntos de leyes. Se mostró luego cómo cargarlas al programa como modelos. Dado que en las aplicaciones de la cladística la combinatoria de posibilidades hace que algunos conceptos tengan extensiones relativamente grandes, sería muy engorroso cargar la denotación de cada concepto a mano, por extensión. Por ello, se mostró cómo

puede usarse una combinación de métodos manuales y automáticos para cargar conceptos tales como el conjunto de los árboles filogenéticos, el conjunto de todas las asignaciones posibles, los largos de cada cladograma, etc. Con esto cargado fue posible llevar a cabo los primeros dos procedimientos de contrastación de reconstrucciones, ambos con resultados exitosos (los resultados de los métodos y las valuaciones de las oraciones dan en todos los casos los resultados esperados). Algo idéntico ocurre con las condiciones de ligadura, examinadas en 5.3. Por último, en 5.4, se muestra cómo cargar el método de determinación para optimizar caracteres (y por tanto calcular el largo de un árbol) con el algoritmo de Fitch, introducido al final del capítulo 3. Se mostró que utilizar este método aumenta más de 10 veces la eficiencia computacional en el ejemplo sencillo introducido, y ello se ejemplificó así el motivo por el que los sistemáticos prestan tanta atención a encontrar más y mejores algoritmos.

Por último, el capítulo 6 extrae algunas consecuencias adicionales del análisis llevado a cabo en los capítulos anteriores (especialmente 3-5). Las consecuencias se dividen en dos tipos, para el estructuralismo metateórico y para las discusiones al interior de la sistemática. Las primeras (sección 6.1) conciernen particularmente a la noción de método de determinación. Se introdujo una elucidación alternativa a la estándar de esta noción, identificándolos con algoritmos en lugar de clases de modelos —un ejemplo de un método reconstruido como un algoritmo ya había sido dado en el capítulo 5. Asimismo, se propuso una elucidación de la noción de ejecución de un algoritmo como una secuencia de modelos, lo cual me permitió distinguir entre conceptos T-teóricos y T-no-teóricos utilizando la elucidación alternativa de método de determinación. La idea aquí fue sostener que un concepto es T-teórico si y solo si todas sus ejecuciones son tales que el último modelo de la secuencia satisface las leyes (el criterio es, así, relativamente parecido al estándar). Se comentaron, en la subsección 6.1.4, algunas ventajas adicionales de la elucidación alternativa propuesta. Ejemplos son la automatización parcial del criterio de T-determinabilidad, la introducción de un nuevo tipo de desarrollo diacrónico basado en la mejora de la eficiencia de métodos y la propuesta de una nueva noción de restricción algorítmica.

El segundo conjunto de consecuencias concierne a la noción de homología y a la supuesta circularidad que existiría en torno a este concepto. La sección 6.2 comienza exponiendo el problema clásico de la circularidad, que surgiría porque la ancestría común a la vez explica y se usa para definir a la noción de homología, para luego exponer cómo este problema reaparece en el marco de la cladística (la matriz de datos contiene homologías y es parte del *input* del análisis, y a

la vez un esquema de homologías sería cognoscible solo a partir del *output* de un análisis filogenético). Se consideró luego a una solución estándar, la distinción entre dos conceptos de homología distintos (primaria y secundaria en la terminología usada por de Pinna, 1991) y la discusión en torno al primero de estos conceptos que se dio al interior del debate por el cladismo de patrón (cuya posición general fue resumida en 6.2.4). Más particularmente, se expusieron las objeciones de autores como Hull (1988) y Pearson (2010) a la posición cladista de patrón en torno a las homologías, basadas en la crítica a la distinción teórico-observacional. Se ofreció una respuesta a estas críticas, sobre la base del criterio de T-teoricidad del estructuralismo metateórico. Se argumentó que la crítica mencionada estaba basada no en la incorrección de la distinción entre homología primaria y secundaria sino en el vocabulario metateórico inadecuado utilizado por los cladistas de patrón para introducirla. Por último, se examinó la respuesta de Pearson (2018) a estos argumentos (ya publicados en Roffé, Ginnobili & Blanco, 2018).

7.2. Trabajo por realizar

Considero que los resultados obtenidos en la presente tesis son significativos y constituyen un aporte original a la metateoría estructuralista y a la discusión en filosofía de la sistemática en torno a la cladística. A su vez, ellos abren nuevos interrogantes y posibles líneas de investigación, tanto en lo relativo a la contrastación de reconstrucciones como en lo relativo a aspectos conceptuales de la cladística.

Respecto de lo primero, otro modo interesante de contrastar una reconstrucción, no considerado en la presente tesis, sería a partir de los teoremas que la teoría permite inferir. Es decir, los resultados que se suelen "demostrar" (informalmente) en las exposiciones estándar de las teorías deben ser derivables de los axiomas propuestos (un ejemplo de ello sería el equilibrio de Hardy-Weinberg, usualmente demostrado en los capítulos iniciales de cualquier manual de genética de poblaciones, y probado como teorema en Roffé, 2019). Este criterio resultaría más difícil de automatizar, aunque la utilización de probadores automáticos de teoremas (p.e. aquellos implementados en Isabelle/HOL, véase Nipkow et al., 2002) promete llevar a resultados interesantes. Otro aspecto interesante de estos probadores es que funcionan con lógicas de orden superior, capaces de formalizar la semántica de otras lógicas objeto (esto se conoce como el

enfoque SSE, por *shallow semantical embeddings*, véanse Benzmüller, 2019; Kirchner et al., 2019). De ese modo, quizás sería posible realizar cálculos similares a los que hace Reconstructor (evaluar oraciones, e incluso obtener predicciones automatizadas) bajo un enfoque sintáctico (de teoría de la prueba), en lugar de semántico (de teoría de modelos, como utiliza Reconstructor). Una comparación de las ventajas y desventajas de ambos enfoques sería deseable, aunque eso requeriría previamente el desarrollo de las herramientas presentadas en esta tesis utilizando ese otro software y enfoque (lo cual no es una tarea trivial, y requiere conocimientos del cálculo lambda y de lenguajes de programación funcionales, con los que no cuento actualmente). Espero, en un futuro cercano, poder abocarme a esta tarea.

Por otra parte, las discusiones en filosofía de la sistemática tienen una extensión mucho más amplia y no se agotan en la cuestión de las homologías, el aspecto tratado en mayor profundidad en esta tesis (incluso el tema de la homología no se agota en el presente análisis, véase por ejemplo Roffé, en prensa, para otros resultados filosóficos recientes obtenidos por el autor). Como se dijo en la introducción, la sistemática fue un campo de batalla constante entre distintos programas de investigación desde la segunda mitad del siglo XX, y eso llevó a que los propios sistemáticos hayan discutido temas filosóficos en relativa extensión. Para comenzar, existen otros métodos de inferencia filogenética actualmente en boga, diferentes al método de parsimonia tratado en esta tesis (p.e. máxima verosimilitud y análisis bayesiano, véanse Felsenstein, 1981; Huelsenbeck et al., 2001). También existen modificaciones y ampliaciones al método de parsimonia clásico no consideradas en esta tesis (véanse por ejemplo Goloboff, 1993; Wheeler et al., 2006). Puede resultar interesante para un análisis de la estructura lógica de la filogenética reconstruir estos otros métodos / teorías y compararlos con la presente reconstrucción. Incluso dentro de la cladística estándar hay numerosas discusiones que no fueron tratadas aquí, y para las cuales la reconstrucción presentada podría tener consecuencias interesantes. Algunos ejemplos son la discusión sobre el pesado de caracteres (Kluge, 1997; Neff, 1986); la aplicabilidad de la filosofía de Popper a la cladística (de Queiroz, 2014; Hull, 1999; Rieppel, 2003, 2008; Vogt, 2014a, 2014b, entre muchos otros) —Hull (1988) examina algunos de los motivos históricos por los que Popper se convirtió en el referente filosófico de muchos cladistas—; la discusión sobre el estatus y la definición de las especies Ghiselin, 1974; Hull, 1978; Wilkins, 2009 entre muchos otros), etc. También espero, en un futuro cercano, poder extender los resultados obtenidos en la presente tesis a (al menos) algunas de esas discusiones.

En general, espero que el presente trabajo contribuya no solo a clarificar discusiones metateóricas y a mejorar el aparato metateórico mismo (con propuestas conceptuales y con herramientas computacionales) sino también a que los metacientíficos se sientan atraídos hacia una visión y una actitud más científicas hacia su propia disciplina. Como afirmaron los empiristas lógicos en lo que probablemente sea el documento fundacional de la disciplina (el Manifiesto del Círculo de Viena): a pesar de que la concepción científica del mundo "pudiera parecer a primera vista (…) un punto de consideración puramente teórico" tiene, sin embargo "una conexión interna" con "los esfuerzos hacia una nueva organización de las relaciones económicas y sociales, hacia la unión de la humanidad, hacia la renovación de la escuela y la educación" (Hahn et al., [1929] 2002, p. 111). Sin pretender que la presente tesis lleve directamente a mejoras en esos aspectos de la vida comunitaria, sí espero que permita fomentar y promover la actitud empirista que aquellos autores veían como ligada al desarrollo y al bienestar de la sociedad.

Referencias bibliográficas

Abramovich, S. (2013). Computers in Mathematics Education: An Introduction. *Computers in the Schools*, *30*(1–2), 4–11. doi: 10.1080/07380569.2013.765305

Abreu, C. (2012). La teoría de los grupos de referencia. *Agora: papeles de Filosofía*, *31*(2), 278–309.

Abreu, C. (2014). Análisis estructuralista de la teoría de la anomia. *Metatheoria – Revista de Filosofía e Historia de la Ciencia*, *4*(2), 9–22.

Adams, E. N. (1972). Consensus Techniques and the Comparison of Taxonomic Trees. *Systematic Zoology*, *21*(4), 390–397. doi: 10.2307/2412432

Alleva, K., Díez, J., & Federico, L. (2017). Models, theory structure and mechanisms in biochemistry: The case of allosterism. *Studies in History and Philosophy of Science Part C: Studies in History and Philosophy of Biological and Biomedical Sciences*, *63*, 1–14. doi: 10.1016/j.shpsc.2017.03.004

Ashlock, P. D. (1974). The Uses of Cladistics. *Annual Review of Ecology and Systematics*, *5*(1), 81–99. doi: 10.1146/annurev.es.05.110174.000501

Aubert, D. (2015). A Formal Analysis of Phylogenetic Terminology: Towards a Reconsideration of the Current Paradigm in Systematics. *Phytoneuron*, *2015–66*, 1–54.

Balzer, W., & Lorenzano, P. (2000). The Logical Structure of Classical Genetics. *Journal for General Philosophy of Science*, *31*(2), 243–266. Retrieved from JSTOR.

Balzer, W., Moulines, C. U., & Sneed, J. D. (2012). *Una arquitectónica para la ciencia. El programa estructuralista*. Bernal: Universidad Nacional de Quilmes.

Bar-Hillel, Y. (1970). *Neorealism vs. Neopositivism. A Neo-Pseudo Issue*. Jerusalem: The Magnes Press, The Hebrew University.

Barrio, E. (Ed.). (2014). *La lógica de la verdad*. Buenos Aires: Eudeba.

Bartelborth, T. (1988). *Eine logische Rekonstruktion der klassischen Elektrodynamik*. P. Lang.

Beatty, J. (1982). Classes and Cladists. *Systematic Zoology*, *31*(1), 25–34.

Benzmüller, C. (2019). Universal (meta-)logical reasoning: Recent successes. *Science of Computer Programming*, *172*, 48–62. doi: 10.1016/j.scico.2018.10.008

Bernabé, F. N. (2018). Cerebros, hormonas y filosofía de la ciencia: La polémica de los cerebros tipo a la luz de la elucidación de la Hipótesis Organizacional Activacional. In S. Seno

Chibeni, L. Zaterka, J. Ahumada, D. Letzen, C. C. Silva, L. Al-Chueyr Pereira Martins, & A. P. Oliveira Pereira de Morais Brito (Eds.), *Filosofía e Historia de la Ciencia en el Cono Sur: Selección de trabajos del X Encuentro de la Asociación de Filosofía e Historia de la Ciencia del Cono Sur* (pp. 138–148). Córdoba, Argentina: Universidad Nacional de Córdoba.

Blanco, D. (2008). La naturaleza de las adaptaciones en la teología natural británica: Análisis historiográfico y consecuencias metateóricas. *Ludus Vitalis, XVI*(30), 3–26.

Blanco, D. (2012). Primera aproximación estructuralista a la Teoría del Origen en Común. *Ágora*, *31*(2), 171-194.

Bochvar, D. A. (1938). Ob odnom trechznacnom iscislenii i ego primenenii k analizu paradoksov klassiceskogo rassirennogo funkcional'nogo iscislenija. *Matematiceskij Sbornik*, *4*, 287–308.

Boolos, G. S., Burgess, J. P., & Jeffrey, R. C. (2007). *Computability and Logic* (5th ed.). Cambridge University Press.

Brady, R. (1985). On the Independence of Systematics. *Cladistics*, *1*(2), 113–126. doi: 10.1111/j.1096-0031.1985.tb00416.x

Bremer, K. (1988). The Limits of Amino Acid Sequence Data in Angiosperm Phylogenetic Reconstruction. *Evolution*, *42*(4), 795–803. doi: 10.2307/2408870

Brower, A. V. Z. (2000). Evolution is not a necessary assumption of cladistics. *Cladistics*, *16*, 143–154.

Brower, Andrew V. Z. (2019). Background knowledge: The assumptions of pattern cladistics. *Cladistics*, *35*(6), 717–731. doi: 10.1111/cla.12379

Brundin, L. (1966). *Transantarctic Relationships and Their Significance: As Evidenced by Chironomid Midges. With a Monograph of the Subfamilies Podonominae and Aphroteniinae and the Austral Heptagyiae*. Almqvist & Wiksell.

Caamaño, M. (2009). A Structural Analysis of the Phlogiston Case. *Erkenntnis*, *70*(3), 331–364. doi: 10.1007/s10670-008-9141-y

Camin, J. H., & Sokal, R. R. (1965). A Method for Deducing Branching Sequences in Phylogeny. *Evolution*, *19*(3), 311–326. doi: 10.2307/2406441

Caponi, G. (2011). *La segunda agenda darwiniana. Contribución preliminar a una historia del programa adaptacionista.* México: Centro de estudios filosóficos, políticos y sociales Vicente Lombardo Toledano.

Caponi, G. (2015). El impacto de la filosofía anatómica de Étienne Geoffoy Saint-Hilaire em el desarrollo de la historia natural. *Gavagai - Revista Interdisciplinar de Humanidades*, *2*(2), 9–31. doi: 10.36661/2358-0666.2015n2.8931

Carman, C. C. (2010). La Refutabilidad Del Sistema de Epiciclos y Deferentes de Ptolomeo. *Principia*, *14*(2), 211–240.

Carnap, R. (1934). On the Character of Philosophic Problems. *Philosophy of Science*, *1*(1), 5–19. doi: 10.1086/286302

Carnap, R. (1950). *Logical Foundations of Probability*. Chicago: University of Chicago Press.

Carnielli, W., & Coniglio, M. E. (2016). *Paraconsistent Logic: Consistency, Contradiction and Negation*. doi: 10.1007/978-3-319-33205-5

Ceccarelli, F. S., Koch, N. M., Soto, E. M., Barone, M. L., Arnedo, M. A., & Ramirez, M. J. (2018). *Data from: The grass was greener: Repeated evolution of specialized morphologies and habitat shifts in ghost spiders following grassland expansion in South America*. doi: 10.5061/dryad.20257

Ceccarelli, F. S., Koch, N. M., Soto, E. M., Barone, M. L., Arnedo, M. A., & Ramírez, M. J. (2019). The Grass was Greener: Repeated Evolution of Specialized Morphologies and Habitat Shifts in Ghost Spiders Following Grassland Expansion in South America. *Systematic Biology*, *68*(1), 63–77. doi: 10.1093/sysbio/syy028

Cobreros, P., Égré, P., Ripley, D., & van Rooij, R. (2014). Foreword: Three-valued logics and their applications. *Journal of Applied Non-Classical Logics*, *24*(1–2), 1–11. doi: 10.1080/11663081.2014.909631

Cracraft, J. (1974). Phylogenetic Models and Classification. *Systematic Biology*, *23*(1), 71–90. doi: 10.1093/sysbio/23.1.71

Darwin, C. (1859). *On the origin of species by means of natural selection*. London: John Murray.

Darwin, C. (1872). *The origin of species, 6th ed.* London: John Murray.

de Pinna, M. C. C. (1991). Concepts and Tests of Homology in the Cladistic Paradigm. *Cladistics*, *7*(4), 367–394. doi: 10.1111/j.1096-0031.1991.tb00045.x

de Queiroz, K. (2014). Popperian Corroboration and Phylogenetics. *Systematic Biology*, *63*(6), 1018–1022. doi: 10.1093/sysbio/syu064

Descartes, R. (1982). *Principles of Philosophy: Translated, with Explanatory Notes* (V. R. Miller & R. Miller, Trans.). Dordrecht, Holland: Reidel.

Descartes, R. (2004). *Descartes: The World and Other Writings* (S. Gaukroger, Trans.). Cambridge; New York: Cambridge University Press.

Díaz, M., & Lorenzano, P. (2017). La red teórica de la dinámica de poblaciones. *Scientiae Studia*, *15*(2), 307–342.

Díez, J. A., Lorenzano, P., & Avila del Palacio, A. (Eds.). (2002). *Desarrollos actuales de la metateoría estructuralista: Problemas y discusiones*. Bernal: Universidad Nacional de Quilmes.

Díez, J., & Lorenzano, P. (2013). Who Got What Wrong? Fodor and Piattelli on Darwin: Guiding Principles and Explanatory Models in Natural Selection. *Erkenntnis*, *78*(5), 1143–1175. doi: 10.1007/s10670-012-9414-3

Donolo, A., Federico, L., & Lorenzano, P. (2007). La teoría de la bioquímica metabólica y sus aplicaciones propuestas. In L. A.-C. P. Martins, M. E. B. Prestes, & R. de A. Martins (Eds.), *Filosofia e história da biologia* (Vol. 2, pp. 39–59). San Pablo: Fundo Mackenzie de Pesquisa.

Douglas, H. (2010). Engagement for Progress: Applied Philosophy of Science in Context. *Synthese*, *177*(3), 317–335.

Dunn, M. (2010). Language phylogenies. In *The Routledge Handbook of Historical Linguistics* (pp. 190–211). doi: 10.4324/9781315794013.ch7

Ebach, M. C., & Williams, D. M. (2019). Ronald Brady and the cladists. *Cladistics*, 1–9. doi: 10.1111/cla.12397

Engelmann, G. F., & Wiley, E. O. (1977). The Place of Ancestor-Descendant Relationships in Phylogeny Reconstruction. *Systematic Zoology*, *26*(1), 1–11. doi: 10.2307/2412861

Falguera, J. L., & Donato-Rodríguez, X. (2016). Flogisto versus oxígeno: Una nueva reconstrucción y su fundamentación histórica. *Crítica: Revista Hispanoamericana de Filosofía*, *48*(142), 87–116.

Farris, J. S. (1969). A Successive Approximations Approach to Character Weighting. *Systematic Zoology*, *18*(4), 374–385. doi: 10.2307/2412182

Farris, J. S. (1970). Methods for Computing Wagner Trees. *Systematic Zoology*, *19*(1), 83–92. doi: 10.2307/2412028

Farris, J. S. (1974). Formal Definitions of Parapliyly and Polyphyly. *Systematic Zoology*, *23*. doi: 10.1093/sysbio/23.4.548

Farris, J. S. (1983). The Logical Basis of Phylogenetic Analysis. In N. Platnick & V. A. Funk (Eds.), *Advances in Cladistics*: *Vol. II* (pp. 7–36). New York: Columbia University Press.

Farris, J. S. (1989). The Retention Index and the Rescaled Consistency Index. *Cladistics*, *5*(4), 417–419. doi: 10.1111/j.1096-0031.1989.tb00573.x

Farris, J. S., Kluge, A., & EcKardt, M. (1970). A Numerical Approach to Phylogenetic Systematics. *Systematic Zoology*, *19*. doi: 10.2307/2412452

Fehr, C., & Plaisance, K. S. (2010). Socially relevant philosophy of science: An introduction. *Synthese*, *177*(3), 301–316. doi: 10.1007/s11229-010-9855-7

Felsenstein, J. (1978). The Number of Evolutionary Trees. *Systematic Zoology*, *27*(1), 27–33. doi: 10.2307/2412810

Felsenstein, J. (1981). Evolutionary trees from DNA sequences: A maximum likelihood approach. *Journal of Molecular Evolution*, *17*(6), 368–376. doi: 10.1007/BF01734359

Felsenstein, J. (1985). Confidence Limits on Phylogenies With a Molecular Clock. *Systematic Biology*, *34*(2), 152–161. doi: 10.2307/sysbio/34.2.152

Felsenstein, J. (1993). *PHYLIP (Phylogeny Inference Package), Version 3.5c*.

Fierro, G. M. (2015). Software educativo CHAKANA: Una propuesta para desarrollar la inteligencia lógico-matemática de estudiantes universitarios. *AtoZ: novas práticas em informação e conhecimento*, *4*(2), 103–107. doi: 10.5380/atoz.v4i2.43691

Fitch, W. M. (1971). Toward Defining the Course of Evolution: Minimum Change for a Specific Tree Topology. *Systematic Zoology*, *20*(4), 406–416. doi: 10.2307/2412116

Frank, P. (1951). Einstein, Mach, and Logical Positivism. In P. Schilpp (Ed.), *Albert Einstein: Philosopher-Scientist* (pp. 270–286). Mjg Books.

Futuyma, D. J. (2005). *Evolution*. Sunderland, Massachusetts: Sinauer Associates Inc.

Ghiselin, M. T. (1974). A Radical Solution to the Species Problem. *Systematic Zoology*, *23*, 536–44.

Ginnobili, S. (2010). La teoría de la selección natural darwiniana. *Theoria*, *25*(1), 37–58.

Ginnobili, S. (2013). La utilidad de las flores: El movimiento del diseño inteligente y la biología contemporánea. *Filosofia e História Da Biologia*, *8*(2), 341–359.

Ginnobili, S. (2014). Explicaciones seleccionistas históricas y ahistóricas. *Ludus Vitalis*, *33*(41), 21–41.

Ginnobili, S. (2016). Missing concepts in natural selection theory reconstructions. *History and Philosophy of the Life Sciences*, *38*(8), 1–33. doi: 10.1007/s40656-016-0109-y

Ginnobili, S. (2018). *La Teoría de la Selección Natural. Una Exploración Metacientífica*. Bernal: Universidad Nacional de Quilmes.

Ginnobili, S., & Carman, C. C. (2016). Explicar y contrastar. *Crítica – Revista Hispanoamericana de Filosofía*, *48*(142), 57–86.

Ginnobili, S., & Roffé, A. (2017). Dos usos de los modelos de optimalidad en las explicaciones por selección natural. *Metatheoria – Revista de Filosofía e Historia de la Ciencia*, *8*(1), 43–55.

Ginnobili, S., & Roffé, A. J. (2018). *Conceptos T-contrastacionales: Interacciones entre los dos criterios de teoricidad*. Presented at the XI Encuentro Iberoamericano sobre Metateoría Estructuralista, Buenos Aires, Argentina.

Giribet, G. (2015). Morphology should not be forgotten in the era of genomics–a phylogenetic perspective. *Zoologischer Anzeiger - A Journal of Comparative Zoology*, *256*, 96–103. doi: 10.1016/j.jcz.2015.01.003

Giribet, G., & Wheeler, W. C. (1999). On Gaps. *Molecular Phylogenetics and Evolution*, *13*(1), 132–143. doi: 10.1006/mpev.1999.0643

Goldreich, O. (2010). *P, NP, and NP-Completeness: The Basics of Computational Complexity*. Cambridge: Cambridge University Press.

Goloboff, P. A. (1993a). Character Optimization and Calculation of Tree Lengths. *Cladistics*, *9*(4), 433–436. doi: 10.1111/j.1096-0031.1993.tb00236.x

Goloboff, P. A. (1993b). Estimating Character Weights During Tree Search. *Cladistics*, *9*(1), 83–91. doi: 10.1111/j.1096-0031.1993.tb00209.x

Goloboff, P. A. (1994). *NONA: a tree searching program*.

Goloboff, P. A. (2014). Extended implied weighting. *Cladistics*, *30*(3), 260–272. doi: 10.1111/cla.12047

Goloboff, P. A., & Catalano, S. A. (2016). TNT version 1.5, including a full implementation of phylogenetic morphometrics. *Cladistics*, *32*(3), 221–238. doi: 10.1111/cla.12160

Goloboff, P. A., Farris, J. S., & Nixon, K. C. (2008). TNT, a free program for phylogenetic analysis. *Cladistics*, *24*(5), 774–786. doi: 10.1111/j.1096-0031.2008.00217.x

Gonzalo, A., & Balzer, W. (2012). A Reconstruction of the "Classical" Linguistic Transformational Theory CLT. *Metatheoria – Revista de Filosofía e Historia de La Ciencia*, *2*(2), 25–49.

Gottwald, S. (2017). Many-Valued Logic. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Winter 2017). Retrieved from https://plato.stanford.edu/archives/win2017/entries/logic-manyvalued/

Hahn, H., Neurath, O., & Carnap, R. (1929). La concepción científica del mundo: El Círculo de Viena. *REDES*, *9*(18), 103–149.

Hanson, N. R. (1958). *Patterns of discovery: An inquiry into the conceptual foundations of science*. Cambridge: Cambridge University Press.

Hempel, C. G. (1958). The Theoretician's Dilemma. In H. Feigl, M. Scriven, & G. Maxwell (Eds.), *Minnesota Studies in the Philosophy of Science* (Vol. 2). Minneapolis: University of Minnesota Press.

Hempel, C. G. (1970). On the "Standard Conception" of Scientific Theories. In M. Radner & S. Winokur (Eds.), *Minnesota Studies in the Philosophy of Science*: *Vol. IV* (pp. 142–163). Minneapolis: University of Minnesota Press.

Hennig, W. (1950). *Grundzuge einer Theorie der phylogenetischen Systematik*. Deutscher Sentralverlag.

Hennig, W. (1965). Phylogenetic Systematics. *Annual Review of Entomology*, *10*(1), 97–116. doi: 10.1146/annurev.en.10.010165.000525

Hennig, W. (1966). *Phylogenetic Systematics* (D. D. Davis & R. Zangerl, Trans.). Illinois: University of Illinois Press.

Hillis, D. M., Bull, J. J., White, M. E., Badgett, M. R., & Molineux, I. J. (1992). Experimental phylogenetics: Generation of a known phylogeny. *Science (New York, N.Y.)*, *255*(5044), 589–592. doi: 10.1126/science.1736360

Hillis, D. M., Bull, J. J., White, M. E., Badgett, M. R., & Molineux, I. J. (1993). Experimental Approaches to Phylogenetic Analysis. *Systematic Biology*, *42*(1), 90–92. doi: 10.2307/2992559

Howard, D. (2003). Two Left Turns Make a Right: On the Curious Political Career of North American Philosophy of Science at Midcentury. In *Logical Empiricism in North America*. University of Minnesota Press.

Howard, D. (2009). Better Red Than Dead: Putting an End to the Social Irrelevance of Postwar Philosophy of Science. *Science and Education*, *18*(2), 199–220.

Huelsenbeck, J. P. (1995). Performance of Phylogenetic Methods in Simulation. *Systematic Biology*, *44*(1), 17–48. doi: 10.1093/sysbio/44.1.17

Huelsenbeck, J. P., Ronquist, F., Nielsen, R., & Bollback, J. P. (2001). Bayesian Inference of Phylogeny and Its Impact on Evolutionary Biology. *Science*, *294*(5550), 2310–2314. doi: 10.1126/science.1065889

Hull, D. (1978). A Matter of Individuality. *Philosophy of Science*, *45*(3), 335–360. doi: 10.1086/288811

Hull, D. (1979). The Limits of Cladism. *Systematic Biology*, *28*(4), 416–440. doi: 10.2307/sysbio/28.4.416

Hull, D. (1988). *Science as a Process*. Chicago: The University of Chicago Press.

Hull, D. L. (1999). The Use and Abuse of Sir Karl Popper. *Biology and Philosophy*, *14*(4), 481.

Jardine, N. (1967). The Concept of Homology in Biology. *The British Journal for the Philosophy of Science*, *18*(2), 125–139.

Kirchner, D., Benzmüller, C., & Zalta, E. N. (n.d.). Mechanizing Principia Logico-Metaphysica in Functional Type-Theory. *The Review of Symbolic Logic*, 1–13. doi: 10.1017/S1755020319000297

Kitcher, P. (1993). *The advancement of science: Science without legend, objectivity without illusions*. New York ; Oxford: Oxford University Press.

Kitching, I. J., Forey, P. L., Humphries, C. J., & Williams, D. M. (1998). *Cladistics: The Theory and Practice of Parsimony Analysis* (2nd edition). Oxford ; New York: Oxford University Press.

Kleene, S. C. (1952). *Introduction to metamathematics*. Princeton: Van Nostrand.

Kluge, A. G. (1997). Sophisticated falsification and research cycles: Consequences for differential character weighting in phylogenetic systematics. *Zoologica Scripta*, *26*(4), 349–360. doi: 10.1111/j.1463-6409.1997.tb00424.x

Kluge, A. G., & Farris, J. S. (1969). Quantitative Phyletics and the Evolution of Anurans. *Systematic Zoology*, *18*(1), 1–32. doi: 10.2307/2412407

Kremer, P. (2016). The Revision Theory of Truth. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Winter 2016). Retrieved from https://plato.stanford.edu/archives/win2016/entriesruth-revision/

Kripke, S. (1975). Outline of a Theory of Truth. *Journal of Philosophy*, *72*(19), 690–716. doi: 10.2307/2024634

Kuhn, T. S. (1962). *The structure of scientific revolutions*. Chicago ; London: University of Chicago Press.

Kuhn, T. S. (1970). *The structure of scientific revolutions* (2nd ed.). Chicago, London: University of Chicago Press.

Labarque, F. M., Soto, E. M., Ramírez, M. J., & Arnedo, M. A. (2015). Chasing ghosts: The phylogeny of Amaurobioidinae ghost spiders (Araneae, Anyphaenidae). *Zoologica Scripta*, *44*(5), 550–561. doi: 10.1111/zsc.12119

Lamarck. (1809). *Philosophie zoologique*. Bruxelles: Culture et Civilisation.

Lankester, E. R. (1870). On the use of the term homology in modern zoology, and the distinction between homogenetic and homoplastic agreements. *The Annals and Magazine of Natural History; Zoology, Botany, and Geology*, *6*, 34–43. doi: 10.1080/00222937008696201

Lastiri, M. (2012). Aplicaciones intencionales de la mecánica cuántica. *Agora: papeles de Filosofía*, *31*(2), 271–285.

Lewis, D. (1970). How to Define Theoretical Terms. *The Journal of Philosophy*, *67*(13), 427–446.

Lorenzano, C. (2002). Una reconstrucción estructural de la bioquímica. In J. A. Diez & P. Lorenzano (Eds.), *Desarrollos actuales de la metateoría estructuralista: Problemas y discusiones* (pp. 209–230). Quilmes: Universidad Na-cional de Quilmes/Universidad Autónoma de Zacatecas/Universidad Rovira i Virgili.

Lorenzano, P. (2002). La teoría del gen y la red teórica de la genética. In Jose Díez & P. Lorenzano (Eds.), *Desarrollos actuales de la metateoría estructuralista: Problemas y discusiones* (pp. 285–330). Bernal: Universidad Nacional de Quilmes.

Lorenzano, P. (2011). La Filosofía de la Ciencia y El Lenguaje: Relaciones Cambiantes, Alcances y Límites. *Arbor*, *187*(747), 69–80. doi: 10.3989/arbor.2011.747n1008

Lorenzano, P. (2014). Principios-guía y leyes fundamentales en la metateoría estructuralista. *Cuadernos Del Sur*, *43–44*, 35–74.

Lorenzano, P., Blanco, D., Carman, C. C., Donolo, A., Federico, L., Ginnobili, S., … Onaha, M. E. (2007). La mecánica de René Descartes. In P. García & L. Salvatico (Eds.), *Selección de trabajos de las XVII jornadas* (Vol. 13, pp. 309–316). Córdoba: Universidad Nacional de Córdoba.

Lorenzano, P., Blanco, D., Carman, C. C., Donolo, A., Federico, L., Ginnobili, S., … Onaha, M. E. (2008). Cartesiómetro o de cómo aplicar la mecánica cartesiana. In H. Faas & H. Severgnini (Eds.), *Selección de trabajos de las XVIII jornadas* (Vol. 14, pp. 303–309). Córdoba: Universidad Nacional de Córdoba.

Lorenzano, P., Blanco, D., Carman, C. C., Donolo, A., Federico, L., Ginnobili, S., … Onaha, M. E. (2010). El cartesiómetro: Una propuesta de aplicación consistente de las leyes del movimiento de Descartes. In R. Martins, L. Lewowicz, J. Ferreira, C. Silva, & L. Pereira Martins (Eds.), *Filosofia e história da ciência no Cone Sul. Seleção de trabalhos do 6º Encontro* (pp. 493–504). Campinas: Associação de Filosofia e História da Ciência do Cone Sul (AFHIC).

Łukasiewicz, J. (1920). O logice trójwartościowej. *Studia Filozoficzne*, *5*, 170–171.

Łukasiewicz, J. (1970). *Selected Works*. Amsterdam: North-Holland Publishing Company.

Maddison, W. P., Donoghue, M. J., & Maddison, D. R. (1984). Outgroup Analysis and Parsimony. *Systematic Zoology*, *33*(1), 83–103. doi: 10.2307/2413134

Manhart, K. (1995). *KI-Modelle in den Sozialwissenschaften: Logische Struktur und wissensbasierte Systeme von Balancetheorien*. Oldenbourg.

Martin, J., Blackburn, D., & Wiley, E. O. (2010). Are node-based and stem-based clades equivalent? Insights from graph theory. *PLOS Currents Tree of Life*. doi: 10.1371/currents.RRN1196

Mayr, E. (1982). *The Growth of Biological Thought*. Cambridge, MA: Harvard University Press.

Mayr, Ernst. (1965). Numerical Phenetics and Taxonomic Theory. *Systematic Zoology*, *14*(2), 73–97. doi: 10.2307/2411730

McCumber, J. (1996). Time in the Ditch: American Philosophy and the McCarthy Era. *Diacritics*, *26*(1), 33–49.

Mendez, D., & Casanueva, M. (2006). A reconstruction of Darwin's pangenesis in a graph format. In G. Ernst & K. G. Niebergall (Eds.), *Philosophie der Wissenschaft – Wissenschaft der Philosophie – Festschrift für C. Ulises Moulines zum 60. Geburstag* (pp. 157–164). Mentis: Paderborn.

Mishler, B. D. (2005). The logic of the data matrix in phylogenetic analysis. In *Parsimony, Phylogeny, and Genomics* (pp. 57–70). Oxford: Oxford University Press.

Moulines, C. U. (1978). Cuantificadores existenciales y principios-guía en las teorías físicas. *Crítica: Revista Hispanoamericana de Filosofía*, *10*(29), 59–88. Retrieved from JSTOR.

Moulines, C. U. (1991). *Pluralidad y recursión*. Madrid: Alianza Universidad.

Moulines, C. U. (1998). Esbozo de Ontoepistemosemántica. *Theoria: Revista de Teoría, Historia y Fundamentos de La Ciencia*, *13*(1), 141–159.

Moulines, C. U. (2011). Cuatro tipos de desarrollo teórico en las ciencias empíricas. *Metatheoria – Revista de Filosofía e Historia de la Ciencia*, *1*(2), 11–27.

Neff, N. A. (1986). A Rational Basis for a Priori Character Weighting. *Systematic Biology*, *35*(1), 110–123. doi: 10.1093/sysbio/35.1.110

Nelson, G. (1971). Paraphyly and Polyphyly: Redefinitions. *Systematic Biology*, *20*(4), 471–472. doi: 10.1093/sysbio/20.4.471

Nelson, G. (1972). Comments on Hennig's "Phylogenetic Systematics" and Its Influence on Ichthyology. *Systematic Zoology*, *21*(4), 364–374. doi: 10.2307/2412429

Nelson, G. (1973). Classification as an Expression of Phylogenetic Relationships. *Systematic Biology*, *22*(4), 344–359. doi: 10.2307/2412943

Nelson, G. (1978). Ontogeny, Phylogeny, Paleontology, and the Biogenetic Law. *Systematic Biology*, *27*(3), 324–345. doi: 10.2307/2412883

Nelson, G., & Platnick, N. (1981). *Systematics and Biogeography, Cladistics and Vicariance*. New York: Columbia University Press.

Nipkow, T., Paulson, L. C., & Wenzel, M. (2002). *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Retrieved from https://www.springer.com/gp/book/9783540433767

Nixon, K. C., & Carpenter, J. M. (1993). On Outgroups. *Cladistics*, *9*(4), 413–426. doi: 10.1111/j.1096-0031.1993.tb00234.x

Nixon, K. C., & Carpenter, J. M. (1996). On Simultaneous Analysis. *Cladistics*, *12*(3), 221–241. doi: 10.1111/j.1096-0031.1996.tb00010.x

Nixon, K. C., & Carpenter, J. M. (2012). On homology. *Cladistics*, *28*(2), 160–169. doi: 10.1111/j.1096-0031.2011.00371.x

Nolt, J. (2018). Free Logic. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2018). Retrieved from https://plato.stanford.edu/archives/fall2018/entries/logic-free/

Oakley, T. (2009). A Critique of Experimental Phylogenetics. In T. Garland & M. R. Rose (Eds.), *Experimental Evolution: Concepts, Methods, and Applications of Selection Experiments* (pp. 659–669). doi: 10.1525/california/9780520247666.001.0001

O'Lery, M. (2012). Análisis estructuralista de la teoría de radicales libres y su vínculo con la bioquímica de óxido-reducción. *Ágora. Papeles de Filosofía*, *31*(2), 251–270.

Owen, R. (1848). *On the Archetype and Homologies of the Vertebrate Skeleton*. author.

Owen, R. (1849). *On the nature of Limbs*. John Van Voorst.

Paley, W. (1809). *Natural Theology* (12th ed.). London: J. Faulder.

Patterson, C. (1988). Homology in classical and molecular biology. *Molecular Biology and Evolution*, *5*(6), 603–625. doi: 10.1093/oxfordjournals.molbev.a040523

Patterson, Colin. (1982). Morphological characters and homology. In K. A. Joysey & A. E. Friday (Eds.), *Problems of Phylogenetic Reconstruction* (pp. 21–74). London: Academic Press Inc.

Pearson, C. H. (2010). Pattern Cladism, Homology, and Theory-Neutrality. *History and Philosophy of the Life Sciences*, *32*(4), 475–492.

Pearson, C. H. (2018). Theoricity and homology: A reply to Roffe, Ginnobili, and Blanco. *History and Philosophy of the Life Sciences*, *40*(4), 62. doi: 10.1007/s40656-018-0226-x

Peris-Viñé, L. M. (2011). Actual Models of the Chomsky Grammar. *Metatheoria – Revista de Filosofía e Historia de La Ciencia*, *1*(2), 195–225.

Platnick, N. I. (1977). Cladograms, Phylogenetic Trees, and Hypothesis Testing. *Systematic Biology*, *26*(4), 438–442. doi: 10.1093/sysbio/26.4.438

Platnick, N. I. (1979). Philosophy and the Transformation of Cladistics. *Systematic Zoology*, *28*(4), 537–546. doi: 10.2307/sysbio/28.4.537

Priest, G. (2008). *An Introduction to Non-Classical Logic: From If to Is*. Cambridge University Press.

Putnam, H. (1962). What Theories are Not. In E. Nagel P. Suppes and A. Tarski (Ed.), *Logic, Methodology and Philosophy of Science*. Stanford: Stanford University Press.

Quinn, A. (2016). Phylogenetic Inference to the Best Explanation and the Bad Lot Argument. *Synthese*, *193*(9).

Quinn, A. (2017). When is a Cladist Not a Cladist? *Biology and Philosophy*, *32*(4), 581–598. doi: 10.1007/s10539-017-9577-z

Ramírez, M. J. (2003). The spider subfamily Amaurobioidinae (Araneae, Anyphaenidae): A phylogenetic revision at the generic level. no. 277. *Bulletin of the AMNH*, *277*, 1–262.

Ramírez, M. J. (2007). Homology as a parsimony problem: A dynamic homology approach for morphological data. *Cladistics*, *23*(6), 588–612. doi: 10.1111/j.1096-0031.2007.00162.x

Reisch, G. A. (2005). *How the Cold War transformed Philosophy of Science. To the Icy Slopes of Logic*. New York: University of Cambridge.

Remane, A. (1952). *Die Grundlagen des natürlichen Systems der vergleichenden Anatomie und der Phylogenetik*. Leipzig: Geest & Portig.

Richter, S. (2017). Homology and synapomorphy-symplesiomorphy—Neither synonymous nor equivalent but different perspectives on the same phenomenon. *Cladistics*, *33*(5), 540–544. doi: 10.1111/cla.12180

Rieppel, O. (1988). *Fundamentals of comparative biology*. Basel, Boston, Berlin: Birkhäuser Verlag.

Rieppel, O. (2003). Popper and systematics. *Systematic Biology*, *52*(2), 259–271.

Rieppel, O. (2008). Re-Writing Popper's Philosophy of Science for Systematics. *History and Philosophy of the Life Sciences*, *30*(3–4), 293–316.

Rieppel, O. (2016). *Phylogenetic Systematics: Haeckel to Hennig*. CRC Press.

Rieppel, O., & Kearney, M. (2002). Similarity. *Biological Journal of the Linnean Society*, *75*(1), 59–82. doi: 10.1046/j.1095-8312.2002.00006.x

Robinson, D. F., & Foulds, L. R. (1981). Comparison of phylogenetic trees. *Mathematical Biosciences*, *53*(1), 131–147. doi: 10.1016/0025-5564(81)90043-2

Roffé, A. J. (2019a). Drift as Constitutive: Conclusions From a Formal Reconstruction of Population Genetics. *History and Philosophy of the Life Sciences*, *41*(4), 55. doi: 10.1007/s40656-019-0294-6

Roffé, A. J. (2019b). Reconstructor: A computer program that uses three-valued logics to represent lack of information in empirical scientific contexts. *Journal of Applied Non-Classical Logics*. doi: 10.1080/11663081.2019.1703467

Roffé, A. J., & Ginnobili, S. (2019a). El estatus metateórico de ZFEL. *Humanities Journal of Valparaiso*, *14*, 57–73. doi: 10.22370/rhv2019iss14pp57-73

Roffé, A. J., & Ginnobili, S. (2019b). Optimality Models and the Propensity Interpretation of Fitness. *Acta Biotheoretica*. doi: 10.1007/s10441-019-09369-5

Roffé, A. J., Ginnobili, S., & Blanco, D. (2018). Theoricity, observation and homology: A response to Pearson. *History and Philosophy of the Life Sciences*, *40*(3), 42. doi: 10.1007/s40656-018-0208-z

Saint-Hilaire, É. G. (1830). *Principes de philosophie zoologique*. Pichon et Didier.

Salemans, B. J. P. (1996). Cladistics or the Resurrection of the Method of Lachmann. In P. van Reenen & M. van Mulken (Eds.), *Studies in Stemmatology* (pp. 3–70; By J. Dyk). doi: 10.1075/z.79.03sal

Sankoff, D. (1975). Minimal Mutation Trees of Sequences. *SIAM Journal on Applied Mathematics*, *28*(1), 35–42. doi: 10.1137/0128004

Schmidt, H.-J. (2014). Structuralism in Physics. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Winter 2014). Retrieved from https://plato.stanford.edu/archives/win2014/entries/physics-structuralism/

Schuh, R. T., & Polhemus, J. T. (1980). Analysis of Taxonomic Congruence among Morphological, Ecological, and Biogeographic Data Sets for the Leptopodomorpha (Hemiptera). *Systematic Biology*, *29*(1), 1–26. doi: 10.1093/sysbio/29.1.1

Semple, C., & Steel, M. (2003). *Phylogenetics*. Oxford University Press.

Simpson, G. G. (1961). *Principles of Animal Taxonomy*. New York: Columbia University Press.

Smith, W. L., & Wheeler, W. C. (2006). Venom evolution widespread in fishes: A phylogenetic road map for the bioprospecting of piscine venoms. *The Journal of Heredity*, *97*(3), 206–217. doi: 10.1093/jhered/esj034

Sneed, J. D. (1971). *The Logical Structure of Mathematical Physics*. Dordrecht-Holland: Reidel.

Sober, E. (1988). *Reconstructing The Past: Parsimony, Evolution, and Inference*. MIT Press.

Sober, E. (1993). Experimental Tests of Phylogenetic Inference Methods. *Systematic Biology*, *42*(1), 85–89. doi: 10.2307/2992558

Sokal, R. R. (1962). Typology and empiricism in taxonomy. *Journal of Theoretical Biology*, *3*(2), 230–267. doi: 10.1016/S0022-5193(62)80016-2

Sokal, R. R., & Sneath, P. H. A. (1963). *Principles of Numerical Taxonomy*. W. H. Freeman.

Steel, M. (2016). *Phylogeny: Discrete and Random Processes in Evolution*. Philadelphia: SIAM-Society for Industrial and Applied Mathematics.

Tajer, D. (2018). LFIs and methods of classical recapture. *Logic Journal of the IGPL*, 1–10. doi: 10.1093/jigpal/jzy060

Tattersall, I., & Eldredge, N. (1977). Fact, Theory, and Fantasy in Human Paleontology. *American Scientist*, *65*, 204–211.

Teijeiro, P. (2018). Subvaluationism and classical recapture. *Logic Journal of the IGPL*, 1–13. doi: 10.1093/jigpal/jzy062

Uebel, T. (2013). Pragmatics in Carnap and Morris and the Bipartite Metatheory Conception. *Erkenntnis*, *78*(3), 523–546. doi: 10.1007/s10670-011-9352-5

Uebel, T. (2015). Three Challenges to the Complementarity of the Logic and the Pragmatics of Science. *Studies in History and Philosophy of Science Part A*, *53*, 23–32. doi: 10.1016/j.shpsa.2015.05.002

van Fraassen, B. C. (1989). *Laws and Symmetry*. Oxford University Press.

Vogt, L. (2014a). Popper and phylogenetics, a misguided rendezvous. *Australian Systematic Botany*, *27*(2), 85–94. doi: 10.1071/SB14025

Vogt, L. (2014b). Why phylogeneticists should care less about Popper's falsificationism. *Cladistics*, *30*(1), 1–4. doi: 10.1111/cla.12026

Wajnberg, C.-D., Corruble, V., Ganascia, J.-G., & Moulines, C. U. (2004). A Structuralist Approach Towards Computational Scientific Discovery. In E. Suzuki & S. Arikawa (Eds.), *Discovery Science* (pp. 412–419). Springer Berlin Heidelberg.

Wheeler, W. C. (1996). Optimization Alignment: The End of Multiple Sequence Alignment in Phylogenetics? *Cladistics*, *12*(1), 1–9. doi: 10.1111/j.1096-0031.1996.tb00189.x

Wheeler, W. C. (1999). Fixed Character States and the Optimization of Molecular Sequence Data. *Cladistics*, *15*(4), 379–385. doi: 10.1006/clad.1999.0115

Wheeler, W. C. (2003a). Iterative pass optimization of sequence data. *Cladistics*, *19*(3), 254–260. doi: 10.1111/j.1096-0031.2003.tb00368.x

Wheeler, W. C. (2003b). Search-based optimization. *Cladistics*, *19*(4), 348–355. doi: 10.1111/j.1096-0031.2003.tb00378.x

Wheeler, W. C., Aagesen, L., Arango, C. P., Faivovich, J., Grant, T., D'Haese, C., … Giribet, G. (2006). *Dynamic Homology and Phylogenetic Systematics: A Unified Approach Using Poy*. American Museum of Natural History.

Wiley, E. O. (1979). Cladograms and Phylogenetic Trees. *Systematic Zoology*, *28*(1), 88–92. doi: 10.2307/2413003

Wiley, E. O., & Lieberman, B. S. (2011). *Phylogenetics: Theory and Practice of Phylogenetic Systematics*. John Wiley & Sons.

Wilgenbusch, J. C., & Swofford, D. (2003). Inferring Evolutionary Trees with PAUP*. *Current Protocols in Bioinformatics*, *00*(1), 6.4.1-6.4.28. doi: 10.1002/0471250953.bi0604s00

Wilkins, J. S. (2009). *Defining Species: A Sourcebook from Antiquity to Today*. Peter Lang.