Logic programming applied to genome evolution in cancer*

A. Dal Palù[1], A. Dovier[2], A. Formisano[3], A. Policriti[2], and E. Pontelli[4]

[1] Dipartimento di Matematica e Informatica, Università degli Studi di Parma, Italy
[2] Dipartimento di Scienze Matematiche, Informatiche e Fisiche
Università degli Studi di Udine, Italy
[3] Dipartimento di Matematica e Informatica, Università degli Studi di Perugia, Italy
[4] Department of Computer Science, New Mexico State University

Abstract. As often observed in the literature, cancer evolution follows a path that is unique to each patient; therefore, classical analysis based on the identification of typical mutations, provides little insight in the understanding of the general rules that drive cancer genesis and evolution. Recent genome sequencing pipelines allow researchers to retrieve rich genetic and epigenetic information from sampled tissues. Analyzing and comparing the evolution of cancer cells for each patient over a large time span can provide some accurate information and relationships. This paper presents a project for a logic programming based analysis that processes time-related genomic information.

Keywords: Cancer evolution, Genome analysis, ASP

1 Introduction

Modern sequencing techniques applied to genomic studies are now capable of producing high-throughput data related to specific individuals. With fast and inexpensive methods, it is possible to retrieve accurate information about a DNA sequence, its methylation (used for epigenetic studies), histones modifications, and gene and protein expression. The process can be repeatedly applied to the same sample over years, for instance, before and after a set of pharmacological therapies. The evolution of an organism and/or a specific sample of cells at genomic scale can be tracked when observing such biological properties. The cancer cells include features such as fast changing genome and cross combination of different offsprings of tumoral cells.

The classical theory of gene mutation, used since the 70s, defines the cancer evolution as a Darwinian process, where the cells compete for survival and the mutations accumulated over time may produce the insurgence of a tumor. However, the search for specific markers and pathways did not produce a clear understanding for many cases. More flexible models could capture the large variability of DNA mutations observed in the same type of tumors among patients.

* The work is partially supported by INdAM GNCS 2016 project.

Compared to previous models, where a simple gene mutation was assumed during cancer evolution, new data allows a more precise investigation and suggests new models based on evolution principles. In particular, the temporal dimension is taken into account in the genomic and epigenomic analysis [29]. This novel paradigm is reflected in the growing literature on *Cancer genome evolution*[18]: this research direction considers the genetic material as a global and detailed source of information. The changes among cells generations during the development of a tumor can be tracked and explained by looking at the global properties over time.

The goal of our study is to employ *Answer Set Programming (ASP)* [26, 24] to model new mining techniques, that search for relevant time-dependent relationships. In particular, differently from classical algorithms, where statistical analysis is used to identify strong peaks over a noise threshold, we focus on mixing evolutionary analysis and mutation analysis. The combination of the two techniques allows us to produce a rich and flexible model. The use of logic programming helps in the definition of a declarative model that merges two distinct aspects: the haplotype identification problem and phylogenetic reconstruction. The literature has already offered separate logic programming models of these two problems. In our case, the evolution of cancer genome can provide uniform input to both problems, namely the search for descriptors of mutations that are correlated over time.

Along with the modeling of this novel perspective, another challenge is the size of the data to be analyzed, requiring the use of modern ASP solving technologies and motivating the exploration of novel resolution models, such as those based on the use of parallel programming techniques (e.g., GPU programming, as recently explored in [27, 5, 7, 2, 3]). This paper provides a preliminary report describing the activities of an ongoing GNCS-2016 project, focused on the analysis of genome evolution in cancer, and outlining the potential of this research.

2 Background

We assume that the reader is familiar with Answer Set Programming (see, e.g., [26]). In this section, we briefly introduce the formalization of two well-known problems in bioinformatics. The first problem is the *haplotype inference* problem [16], i.e., the problem of identifying the minimal set of mutations that explain those observed on a population-wide genome sequencing. The second problem considered is the classical problem of *phylogenetic inference*: the reconstruction of a tree that summarizes the mutations over time for a set of species.

ASP is particularly suited to the modeling and resolution of these classes of problems, because of its flexibility in the modeling phase, its elaboration tolerance, and the fast prototyping cycle. In the literature, there are examples of ASP encoding of the haplotyping problem [10] and phylogenetic tree reconstruction problem [25, 9] (along with other uses of ASP to support phylogenetic data, e.g., to support complex queries on phylogenetic repositories [4]). These problems have also been addressed using alternative logic-based and constraint-based

paradigms—the readers are referred to, e.g., [1, 28] for additional references. However there are no applications nor combinations of these techniques in the study of genome evolution in cancer.

2.1 Phylogenetic Inference

Phylogenies are artifacts that describe the relationships among entities (e.g., proteins or genomes) derived from a process of evolution. We often refer to the entities studied in a phylogeny as *taxonomic units (TUs)* or *taxa*.

The field of *Phylogenetics* developed from the domain of biology as a powerful instrument to investigate similarities and differences among entities as a result of an evolutionary process. Evolutionary theory provides a powerful framework for comparative biology, by converting similarities and differences into events reflecting causal processes. As such, evolutionary-based methods provide more reliable answers than the traditional similarity-based methods, as they employ a theory (of evolution) to describe changes instead of relying on simple pattern matching. Indeed, evolutionary analyses have become the norm in a variety of areas of biological analysis. Evolutionary methods have proved successful, not merely in addressing issues of interest to evolutionary biologists, but in regard to practical problems of structural and functional inference [32]. Evolutionary inference of pairing interactions determining ribosomal RNA structure [35] is a clear case in which progress was made by the preferential use of an evolutionary inference method, even when direct (but expensive and imprecise) experimental alternatives were available. Eisen and others [31, 8] have shown how an explicitly evolutionary approach to protein "function" assignment eliminates certain categories of error that arise from gene duplication and loss, unequal rates of evolution, and inadequate sampling. Other inference problems that have been addressed through evolutionary methods include studies of implications of SNPs in the human population [31], identification of specificity-determining sites [14], inference of interactions between sites in proteins [34], interactions between proteins [33], and inferences of categories of sets of genes that have undergone adaptive evolution in recent history [23].

Phylogenetic analysis has also found applications in domains that are outside of the realm of biology; for example, a rich literature has explored the evolution of languages (e.g., [12, 30, 6]). The definitions and techniques employed are the same; of course the notion of "observable property" can be different. Starting from genes one notices differences using string matching algorithms. But differences (to be analyzed and explained) can be more macroscopic such as the presence/absence of a tail in an animal or the way one say "father" in a language.

Modeling. Let us consider the problem of *phylogenetic tree reconstruction*, namely: given a set of data characterizing the entities being studied (e.g., species, genes, languages), we wish to identify a phylogeny that accurately describes the evolutionary lineages among the given entities. We start with the notion of phylogenetic tree and then we give the notion of compatibility of characters.

A *phylogenetic tree* (or simply a *phylogeny*) is typically a labeled binary tree $(V, E, L, \mathcal{T}, \mathcal{L})$ where:

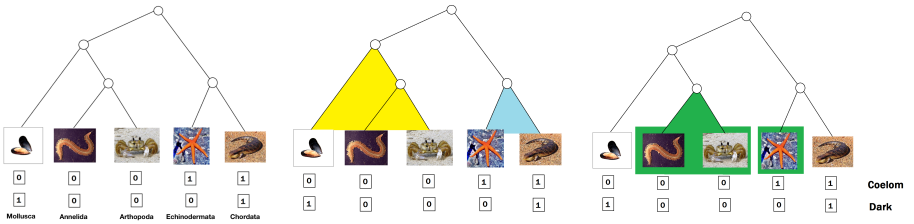

Fig. 1. A Sample Phylogeny (left), compatible (center–Coelom) and incompatible (right–Dark) characters

- The leaves L represent the taxonomic units being compared;
- The internal nodes $V \setminus L$ represent the (hypothetical) ancestral units; in rare cases, the internal nodes correspond to concrete entities (e.g., fossils);
- The edges E of the tree describe evolutionary relationships; the structure of the edges describe the processes that hypothetically led to the evolution of the TUs, e.g., biological processes of *speciation*, *gene duplication*, and *gene loss*;
- Commonly, each TU is described by a collection of finite domain properties, referred to as *characters*. In the formalization, $\mathcal{T} = (C, D, f)$ is the description of such properties, where
  - $C = \{c_1, \ldots, c_k\}$ is a finite set of characters;
  - $D = (D_{c_1}, \ldots, D_{c_k})$ associates a finite domain to each character;
  - $f : L \times C \to \bigcup_{c \in C} D_c$ is a function that provides the value of each character for each TU being studied.
- We are often interested in the length of the branches of a phylogeny and/or the assignment of *dates* to the internal nodes of the phylogeny; if this feature is present, then we will describe it as a function $\mathcal{L} : E \to \mathbb{R}^+$.

Whenever we do not have information about the length of the branches, we omit the component $\mathcal{L}$ from the description of the phylogeny.

For presentation simplicity, we focus on one example with macroscopic observable properties. Fig. 1 (left) shows a phylogenetic tree for the TUs $L = \{\mathit{Mollusca}, \mathit{Annelida}, \mathit{Arthopoda}, \mathit{Echinodermata}, \mathit{Chordata}\}$. In this example, the set of characters is $C = \{\mathit{Coelom}, \mathit{Dark}\}$—*Coelom* denotes the presence/absence of coelom (a body cavity between the intestine and the body walls), while *Dark* denotes the phenotypical character of having dark color. In this example, these are both binary characters, i.e., $D_{\mathit{Coelom}} = D_{\mathit{Dark}} = \{0, 1\}$. The function f describing the five TUs is given by the table underneath each TU in the figure—e.g., $f(\mathit{Annelida}, \mathit{Coelom}) = 0$ and $f(\mathit{Annelida}, \mathit{Dark}) = 0$.

The key point in the phylogenetic tree reconstruction problem is how to define what does it mean to "accurately describe", i.e., what measure of accuracy is used to compare plausible trees. A variety of measures have been proposed, and various phylogenetic reconstruction methods have been proposed based on the specific measure being used to assess quality of the phylogeny. A common method

used in deriving phylogenies is based on the idea of *character compatibility*—a principle derived from Le Quesne's idea of uniquely derived characters [21, 22].

The intuitive idea of compatibility is as follows: a character c is compatible with a phylogeny if the TUs that present the same value for such character are connected by a subtree within the phylogeny. More formally, given a phylogeny $\mathcal{P} = (V, E, L, \mathcal{T}, \mathcal{L})$, with $\mathcal{T} = (C, D, f)$, a character $c \in C$ is compatible with $\mathcal{P}$ if there is a mapping $h_c : V \to D_c$ such that:

- For each $t \in L$ we have that $h_c(t) = f(t, c)$;
- For each $i \in D_c$, the projection of the graph (V, E) on the set of nodes $V_i^c = \{t \in V \mid h_c(t) = i\}$ has a subgraph that has V_i^c as nodes and it is a rooted tree.

A character that is not compatible with a phylogeny $\mathcal{P}$ is said to be *incompatible*. The above (sub-tree) requirement implicitly states that when a character changes (during evolution) it never goes back to the previous value. This is referred to as the *Camin-Sokal* requirement; moreover, it also accounts for the requirement that the "change" occurs in a unique place, known as the *Dollo* requirement.

In the example of Fig. 1, the character *Coelom* is compatible with the given phylogeny—as shown in Fig. 1(middle). On the other hand, the character *Dark* is not compatible with this phylogeny (as shown in Fig. 1(right)).

The goal, in phylogeny reconstruction, is to determine a phylogeny that maximizes the number of characters that are compatible with it. This problem has been often referred to as the *k-incompatibility problem* [11]. Formally, the k-incompatibility problem is the problem of deciding, given a set L of TUs, a character description $\mathcal{T} = (C, D, f)$ of L, and an integer $n \in \mathbb{N}$, whether there is a phylogeny $(V, E, L, \mathcal{T})$ that has at most k incompatible characters.

2.2 Haplotype Inference

The differences between two organisms of the same species are derived from differences in some peculiar points of their DNA sequences. We present here the problem of reconstructing the connection between a set of *diploid* organisms (such as humans), given some information about such specific DNA locations.

The DNA of diploid organisms is organized in pairs of not completely identical copies of *chromosomes*. The sequence of nucleotides from a single copy is called *haplotype*, while the conflation of the two copies constitutes a *genotype*. Each person inherits one of the two haplotypes from each parent. The most common variation between two haplotypes is a difference in a single nucleotide. Using statistical analysis within a population, it is possible to describe and analyze the typical points where these mutations occur. Each of such differences is called a *Single Nucleotide Polymorphism (SNP)*. In other words, a SNP is a single nucleotide site, in the DNA sequence, where more than one type of nucleotide (usually two) occur with a non-negligible population frequency. We refer to such sites as *alleles*.

Considering a specific genotype, a SNP site where the two haplotypes have the same nucleotide is called an *homozygous* site, while it is *heterozygous* otherwise. Research has confirmed that SNPs are the most common and predominant

form of genetic variation in DNA. Moreover, SNPs can be linked to specific traits of individuals and with their phenotypic variations within their population. Consequently, haplotype information in general, and SNPs in particular, are relevant in several contexts, such as, for instance, in the study and diagnosis of genetic diseases, in forensic applications, etc. This makes the identification of the haplotype structure of individuals, as well as the common part within a population, of crucial importance. In practice, biological experiments are used to collect genotype data instead of haplotype data, mainly due to cost or technological limitations. To overcome such limitations, accurate computational methods for inferring haplotype information from genotype data have been developed during the last decades (for a review, the reader is referred to [17, 15, 16]).

Modeling. The haplotype inference problem can be formulated as follows. First, we apply an abstraction and represent genotypes and haplotypes by focusing on the collection of ambiguous SNPs sites in a population. Moreover, let us denote, for each site, the two possible alleles using 0 and 1, respectively. Hence, an haplotype will be represented by a sequence of n components taken from $\{0, 1\}$. Each genotype g, being a conflation of two (partially) different haplotypes h_1 and h_2, will be represented as a sequence of n elements taken from $\{0, 1, 2\}$, such that 0 and 1 are used for its *homozygous* sites, while 2 is used for the *heterozygous* sites. More specifically, following [20], let us define the conflation operation $g = h_1 \oplus h_2$ as follows:

$$g[i] = \begin{cases} h_1[i] & \text{if } h_1[i] = h_2[i] \\ 2 & \text{otherwise} \end{cases}$$

where $g[i]$ denotes the i^{th} element of the sequence g, for $i = 1, \ldots, n$.

We say that a genotype g is *resolved* by a pair of haplotypes h_1 and h_2 if $g = h_1 \oplus h_2$. A set H of haplotypes *explains* a given set G of genotypes, if for each $g \in G$ there exists a pair of haplotypes $h_1, h_2 \in H$ such that $g = h_1 \oplus h_2$.

Given a set G of m genotypes, the *haplotype inference problem* consists of determining a set H of haplotypes that explains G. The cardinality of H is bound by $2m$ but, in principle, each genotype having $k \leq n$ ambiguous sites, can be explained by 2^{k-1} different pairs of haplotypes. For instance, the singleton $G = \{212\}$ (i.e., $k = 2$) can be explained in two ways, namely by choosing $H = \{011, 110\}$ or $H = \{010, 111\}$ (see also Fig. 2). Hence, in general, there might be an exponential number of explanations for a given set G. All of them are, from the combinatorial point of view, "equivalent" and a blind algorithm—not exploiting any biological insights—may result in inaccurate, i.e., biologically improbable, solutions. What is needed is a genetic model of haplotype evolution to guide the algorithm in identifying the "right" solution(s).

Several approaches have been proposed, relying on the implicit or explicit adoption of assumptions reflecting general properties of an underlying genetic model. We focus on one of such formulations, namely *parsimony*. The main underlying idea is the application of a variant of *Ockham's principle of parsimony:* the *minimum-cardinality* possible set H of haplotypes is the one to be chosen as explanation for a given set of genotypes G. For instance the set G in Fig. 2

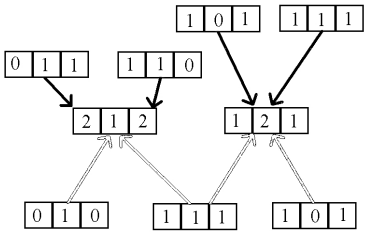

Fig. 2. The set $G = \{212, 121\}$ and two possible explanations

admits two explanations. The one at the bottom, i.e., $\{010, 111, 101\}$, is preferable by the parsimony principle. In this formulation, the haplotype inference problem has been shown in [20] to be APX-hard, through a reduction from the node-covering problem.

3 Methods

The basic idea is to use ASP to model the genome analysis. In particular, as first approximation of the problem, we focus on mutations that took place in specific locations of the DNA (Single Nucleotide Polymorphism). These mutations are tracked at different moments in time for the same individual and tissue, opposed to traditional techniques that search for these mutations across a large set of individuals. Since the data is enriched by time information, it is possible to integrate haplotype search with phylogenetic structure of tumoral fingerprints. In fact, cell offspring relationships are strongly related to an evolutionary tree for species. In our case, it is possible to model different snapshots of the genome at different points in time, and correlate mutations over time as in the classical phylogenetic inference. The algorithms for the construction of a phylogenetic tree need to be modified to capture the evolutionary properties of the various genomes collected from the same patient. Similar approaches have appeared in the literature (e.g., [13]), though not based on logic programming. The goal is to use the combination of haplotyping and phylogenetic tree reconstruction to reconstruct the mutations over time, and provide an evolutionary map of cancer haplotypes. The ASP framework allows us to prototype the models and have a fast feedback about their quality.

3.1 Modeling

The *evolutionary haplotype inference problem* can be formulated by extending the formalization presented for the haplotype inference problem. We define a linear timeline $T = t_0, t_1, \ldots, t_{k-1}$, whose time-steps are associated to each input genotype. Formally a *timed genotype* is a pair (g, t_i) made of a genotype g and a time-step $t_i \in T$. A *timed haplotype* is a haplotype associated to a time step: formally (h, t_i) where h is a haplotype and $t_i \in T$.

We say that a timed genotype (g, t_i) is *resolved* by a pair of timed haplotypes (h_1, t_j) and (h_2, t_k) if $g = h_1 \oplus h_2$, $t_i \geq t_j$ and $t_i \geq t_k$. A set H of timed haplotypes *explains* a given set G of timed genotypes, if for each $g \in G$ there exists a pair of timed haplotypes such that they resolve g.

We need to introduce the notion of haplotype *persistence*: given a set H of timed haplotypes, $(h, t_i) \in H$ is persistent if for any t_j, such that $t_i \leq t_j$, $(h, t_j) \in H$. In other words, persistent haplotypes in H are defined at specific time-steps and they will explain any timed genotypes at times greater or equal to t_i. A set H of timed haplotypes is persistent if every haplotype in H is persistent.

The last notion we introduce is the *preference* over two persistent sets $H_1 \preceq H_2$. Intuitively, we prefer timed haplotypes that appear as late as possible: this reflects the fact that the occurrence of an haplotype cannot be delayed anymore and therefore captures some relevant properties in the timed genomes (e.g., consequences of a therapy). On the other hand, any haplotype at a certain time t_i can be introduced at previous time-steps, without violating any properties. Therefore, a preference that captures the late occurrence of haplotypes reflects a more accurate characterization of the set H. Note that any solution for the original haplotype inference problem can be extended to a timed haplotype solution by adding the time step t_0 to each haplotype.

Formally, given two persistent haplotypes $(h, t_i) \in H_1$ and $(h, t_j) \in H_2$, we say that $(h, t_i) \preceq (h, t_j)$ if $t_i \geq t_j$. We extend the preference to persistent haplotype sets: $H_1 \preceq H_2$ reflects the fact that the set H_1 is preferred to H_2, namely there is no pair $(h, t_i) \in H_1$ and $(h, t_j) \in H_2$ such that $(h, t_i) \not\preceq (h, t_j)$.

Given a set G of timed genotypes, the *evolutionary haplotype inference problem* consists of determining sets H of persistent timed haplotypes that explain G such that there is no other solution $H_1 \preceq H$.

This model introduces only time information to available genotype. It is possible to extend it to other facts that are annotated with the samples. For example, the clinical condition of the patient can provide information about therapies and other physiological parameters. The timed genotypes can be enriched by a tuple of properties that could help in the comparison between solutions of evolutionary haplotype inference problem for different patients. This information can be retrieved from public/controlled access databases (see, e.g., the cancer genome atlas cancergenome.nih.gov).

4 Conclusion

In this work-in-progress paper, we briefly discussed the initial modeling of the evolutionary haplotype inference problem; the problem is tied to investigation of genome evolution in cancer (e.g., as result of pharmacological interventions). The problem is combinatorial in nature, and suitable for modeling and analysis using logic programming techniques. The project is in its infancy and will proceed through the integration of the proposed haplotype inference with techniques to reconstruct an associate evolutionary tree (with techniques borrowed from phylogenetic analysis).

References

1. P. Barahona, L. Krippahl, and O. Perriquet. Bioinformatics: A challenge to constraint programming. *Hybrid Optimization*, 45:463–487, 2011.
2. Federico Campeotto, Agostino Dovier, Ferdinando Fioretto, and Enrico Pontelli. A GPU implementation of large neighborhood search for solving constraint optimization problems. In Proc of *ECAI* 2014 - 21st European Conference on Artificial Intelligence, volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 189–194. IOS Press, 2014.
3. Federico Campeotto, Agostino Dovier, and Enrico Pontelli. A declarative concurrent system for protein structure prediction on GPU. *J. Exp. Theor. Artif. Intell.*, 27(5):503–541, 2015.
4. B. Chisham, B. Wright, T. Le, T. Son, and E. Pontelli. CDAO-Store: Ontology-driven Data Integration for Phylogenetic Analysis. *BMC Bioinformatics*, 12:98, 2011.
5. Alessandro Dal Palù, Agostino Dovier, Andrea Formisano, and Enrico Pontelli. CUD@SAT: SAT solving on GPUs. *J. Exp. Theor. Artif. Intell.*, 27(3):293–316, 2015.
6. A.J. Dobson. Lexicostatistical grouping. *Anthropological Linguistics*, 11:216–221, 1969.
7. Agostino Dovier, Andrea Formisano, Enrico Pontelli, and Flavio Vella. A GPU implementation of the ASP computation. In Marco Gavanelli and John H. Reppy, editors, *Proc of Practical Aspects of Declarative Languages - 18th International Symposium, PADL 2016,*, volume 9585 of *Lecture Notes in Computer Science*, pages 30–47. Springer, 2016.
8. J.A. Eisen and P.C. Hanawalt. A phylogenomic study of DNA repair genes, proteins, and processes. *Mutation Research - DNA Repair*, 435(3):171–213, 1999.
9. Esra Erdem. Applications of answer set programming in phylogenetic systematics. In Marcello Balduccini and Tran Cao Son, editors, *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, volume 6565 of *Lecture Notes in Computer Science*, pages 415–431. Springer, 2011.
10. Esra Erdem, Ozan Erdem, and Ferhan Türe. HAPLO-ASP: Haplotype inference using answer set programming. In Esra Erdem, Fangzhen Lin, and Torsten Schaub, editors, *Logic Programming and Nonmonotonic Reasoning, 10th International Conference, LPNMR 2009, Potsdam, Germany, September 14-18, 2009. Proceedings*, volume 5753 of *Lecture Notes in Computer Science*, pages 573–578. Springer, 2009.
11. G.F. Estabrook. Ancestor-descendant relations and incompatible data: Motivation for research in discrete mathematics. In *Mathematical Hierarchies and Biology*, volume 27 of *DIMAS Series in Discrete Mathematics*, pages 1–28. American Mathematical Society, 1997.
12. H.A. Gleason. Counting and calculating for historical reconstruction. *Anthropological Linguistics*, 1:22–32, 1959.
13. C. Greenman et al. Estimation of rearrangement phylogeny for cancer genomes. *Genome Res.*, 22(2):346–361, 2012.
14. T. Gruber. Toward principles for the design of ontologies used for knowedge sharing. *International Journal of Human Computer Studies*, 43(5-6), 1995.
15. Dan Gusfield. An overview of combinatorial methods for haplotype inference. In Istrail et al. [19], pages 9–25.
16. Dan Gusfield and Steven Hecht Orzack. Haplotype inference. In Srinivas Aluru, editor, *Handbook of Computational Molecular Biology*, Computer & Information Science, chapter 18. Chapman & Hall/CRC, 2005.

17. Bjarni V. Halldórsson, Vineet Bafna, Nathan Edwards, Ross Lippert, Shibu Yooseph, and Sorin Istrail. A survey of computational methods for determining haplotypes. In Istrail et al. [19], pages 26–47.
18. S Horne, C Ye, B Abdallah, G Liu, and H Heng. Cancer genome evolution. *Transl Cancer Res*, 4(3):303–313, 2015.
19. Sorin Istrail, Michael S. Waterman, and Andrew G. Clark, editors. *Computational Methods for SNPs and Haplotype Inference, DIMACS/RECOMB Satellite Workshop, Piscataway, NJ, USA, November 21-22, 2002, Revised Papers*, volume 2983 of *Lecture Notes in Computer Science*. Springer, 2004.
20. Giuseppe Lancia, Maria Cristina Pinotti, and Romeo Rizzi. Haplotyping populations by pure parsimony: Complexity of exact and approximation algorithms. *INFORMS Journal on Computing*, 16(4):348–359, 2004.
21. W.J. Le Quesne. A Method of Selection of Characters in Numerical Taxonomy. *Syst. Zool.*, 18:201–205, 1969.
22. W.J. Le Quesne. Further Studies Based on the Uniquely Derived Character Concept. *Syst. Zool.*, 21:281–288, 1972.
23. D.A. Liberles and M.L. Wayne. Tracking adaptive evolutionary events in genomic sequences. *Genome Biol.*, 3(6), 2002.
24. Victor W. Marek and Miroslaw Truszczyński. Stable models and an alternative logic programming paradigm. In *The Logic Programming Paradigm*, pages 375–398. Springer Verlag, 1999.
25. N. Moore and P. Prosser. The ultrametric constraint and its application to phylogenetics. *J. Artif. Intell. Res.*, 32:901–938, 2008.
26. Ilkka Niemelä. Logic programs with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):241–273, 1999.
27. J. D. Owens et al. GPU computing. *Proceedings of the IEEE*, 96(5):879–899, 2008.
28. A. Dal Palù, A. Dovier, A. Formisano, and E. Pontelli. Exploring Life through Logic Programming: Answer Set Programming in Bioinformatics. In Michael Kifer and Annie Liu, editors, *Declarative Logic Programming: Theory, Systems, and Applications*. Springer, To appear, available as TR-CS-NMSU-2014-10-24 New Mexico State University.
29. O. Podlaha, M. Riester, S. De, and F. Michor. Evolution of the cancer genome. *Trends Genet.*, 28(4):155–163, 2012.
30. D. Ringe, T.Warnow, and A. Taylor. Indo-European and computational cladistics. *Transactions of the Philological Society*, 100(1):59–129, 2002.
31. E.A. Stone and A. Sidow. Physicochemical constraint violation by missense substitutions mediates impairment of protein function and disease severity. *Genome Res.*, 15(7):978–986, 2005.
32. J.L. Thorne. Models of protein sequence evolution and their applications. *Curr. Opin. Genet. Dev.*, 10(6):602–605, 2000.
33. E.R. Tillier, L. Biro, G. Li, and D. Tillo. Codep: maximizing co-evolutionary interdependencies to discover interacting proteins. *Proteins*, 63(4):822–831, 2006.
34. E.R. Tillier and T.W. Lui. Using multiple interdependency to separate functional from phylogenetic correlations in protein alignments. *Bioinformatics*, 19(6):750–755, 2003.
35. C.R. Woese and N.R. Pace. Probing RNA Structure, Function, and History by Comparative Analysis. In *The RNA World*, pages 91–117. Cold Spring Harbor Laboratory Press, 1993.