Manipulación de conocimiento en Sistemas Multi-Agentes por medio de espacios de tuplas

Luciano H. Tamargo, Marcelo A. Falappa, Alejandro García

Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
Laboratorio de Investigación y Desarrollo de Inteligencia Artificial
Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur
Av. Alem 1253, (B8000CPB) Bahía Blanca, Argentina
Tel: (0291) 459-5135 / Fax: (0291) 459-5136
Email: {lt,maf,ajg}@cs.uns.edu.ar

Resumen

Esta línea de investigación tiene como objetivo analizar el intercambio de conocimiento en sistemas multi-agente utilizando áreas de conocimiento compartidas. El proyecto involucra la extensión del modelo Linda con el objetivo de lograr adaptarlo a un entorno distribuido en un sistemas multi-agente con agentes con conocimiento. En este trabajo, en primer lugar se describe el conjunto de operaciones sobre espacios de tuplas que hemos propuesto en [1]. En base a estas operaciones se propondrá una extensión y finalmente se describirá el trabajo a futuro a realizar en esta línea de investigación.

1. Introducción

En un sistema multi-agente los agentes deben poder intercambiar información, ya sea en un entorno colaborativo o basado en negociación. Este intercambio puede hacerse por medio de pasaje de mensajes o utilizando áreas de conocimiento compartidas. Este trabajo tiene como objetivo analizar el intercambio de conocimiento en sistemas multi-agente por medio de áreas de conocimiento compartidas distribuidas. La importancia del mismo radica en que, disponer de un modelo para compartir conocimiento, es un aspecto a resolver en sistemas multi-agente con agentes deliberativos. Por lo tanto, sería interesante contar con un modelo para el mantenimiento de bases de conocimiento que permita abstraerse de los aspectos de implementación.

El modelo Linda [2, 3, 4, 5], es un modelo de memoria compartida originalmente definido para proveer comunicación y sincronización en programas con paralelismo. Utiliza un espacio de memoria compartida llamado *"espacio de tuplas"* (ET), sobre el cual propone seis operaciones. Agregando estas operaciones del ET a un lenguaje base se genera un dialecto de programación paralela. Los ETs son una abstracción a partir de la cual los procesos cooperan y se comunican. Estos pueden verse como áreas de memoria compartidas asociativas referenciadas que no requieren hardware subyacente para el área de memoria física donde residen. Es importante notar que los ETs difieren de las relaciones que son propuestas en el modelo de datos relacional [6], debido a que las relaciones sólo aceptan tuplas que tienen formato (las mismas deben tener la misma aridad, y sus atributos deben ser tipados y mantener un orden); sin embargo, los ETs aceptan tuplas sin formato.

Financiado parcialmente por CONICET (PIP 5050), Universidad Nacional del Sur (PGI 24/ZN11) y Agencia Nacional de Promoción Científica y Tecnológica (PICT 2002 Nro 13096).

Linda provee cuatro operaciones básicas de espacio de tuplas, out, in , rd y $eval$, y dos variantes adicionales inp y rdp. La operación **out(t)** causa que la tupla t sea agregada al ET; **in(t)** causa que alguna tupla s que concuerde con la tupla-molde t sea retirada del ET (lectura destructiva), y si no se encuentra una tupla *s* que coincida cuando *in(t)* se ejecuta, la ejecución del proceso se suspende hasta que una tupla que coincida esté disponible (es una primitiva bloqueante); **rd(t)** es igual a *in(t)*, excepto que la tupla que coincide no es retirada y permanece en el ET. Las versiones predicativas de *in* y *rd*, **inp** y **rdp** respectivamente, intentan localizar una tupla que coincida, y retornan 0 si la búsqueda falla, en caso contrario retornan 1. La operación **eval(t)** es igual a *out(t)*, excepto que t es evaluado después (en paralelo) en vez de antes de ser ingresado en el ET.

En la década del 90 surgieron variantes al modelo, teniendo en cuenta múltiples ETs distribuidos en forma remota, buscando mejorar aspectos que conciernen a los ambientes distribuidos y paralelos. Estas variantes fueron propuestas en trabajos como [7, 8, 9, 10, 11, 12], y consisten en permitir la existencia de **varios espacios de tuplas**. Esto brinda la posibilidad de que un sistema multi-agente pueda trabajar con más de un área de conocimiento permitiendo descentralizar el sistema, lo que provoca que el conocimiento sea compartido de manera distribuida.

A partir de la posibilidad de que puedan coexistir varios ETs en un mismo sistema multi-agente, surgen naturalmente, nuevas operaciones sobre ETs. Esto es, operaciones que tengan como dominio y rango ETs, lo cual hace al modelo aún más interesante y poderoso. Pero se debe notar que, el hecho de agregar más operaciones que se aplican sobre ETs hace que el modelo pierda *ortogonalidad*, obligando al usuario a pensar en más operaciones. Sin embargo, nosotros consideramos que estas nuevas operaciones le agregan expresividad, y permiten que el modelo se aproxime, aún más, a los objetivos del trabajo, ya que permite que los agentes tengan la posibilidad de manipular el conocimiento en forma masiva. Estas nuevas operaciones se definen en la siguiente sección.

2. Operaciones que tienen como dominio y rango ETs

Ante la posibilidad de que en un sistema multi-agente puedan coexistir varios espacios de tuplas, resulta interesante disponer de operaciones que tengan como dominio y rango ETs, como se ha visto en los trabajos [8, 13] donde se sugieren las operaciones *collect* y *copy-collect*, respectivamente. A diferencia de estas dos, en esta sección se incluyen y ejemplifican operaciones para unir dos ETs, obtener los elementos comunes y calcular la diferencia, entre otras. Estas operaciones las hemos definido en [1] y se incluye en este artículo un resumen de ellas para que el mismo sea autocontenido. No obstante, en el presente artículo se incluye una nueva definición de la operación que permite realizar la diferencia de espacios de tupla. Esto es, ahora contamos con dos tipos de diferencias, *diferencia existencial* y *diferencia sin réplica*. Todas estas operaciones, en algún sentido recuerdan a las operaciones del álgebra relacional propuestas en [6]. Sin embargo, difieren de éstas ya que en el álgebra relacional las relaciones están formateadas y, como se mencionó con anterioridad, los espacios de tuplas no. Es importante destacar que los espacios de tuplas pueden contener tuplas repetidas [14]. Por lo tanto, las operaciones que se sugerirán tendrán variantes que admiten tuplas replicadas, las cuales se asemejan a las operaciones multi-set del álgebra relacional extendida propuestas en [15]. En el mismo sentido que las operaciones del álgebra relacional, estas difieren de las propuestas en este trabajo, ya que las relaciones sólo admiten tuplas con formato.

Dentro de un ET, el orden de las tuplas no será considerado relevante. Por lo tanto, en las operaciones que se describen a continuación, cuando hay varias tuplas repetidas, y se debe decidir cual de ellas será parte de la solución, la elección se realizará de manera no determinística.

2.1. Operaciones para unión de espacios de tuplas

Estas operaciones permiten, por ejemplo, que un agente que está compartiendo conocimiento en diferentes ETs con distintos agentes, logre juntar en un único ET todo su conocimiento, permitiéndole de esta manera, poder trabajar con su conocimiento global sin alterar el conocimiento de los demás agentes.

Definición 1: **(Unión total)** Dados dos espacios de tuplas ET_1 y ET_2 la union total $ET_1 \uplus ET_2$ es un espacio de tuplas que contiene todas las tuplas que pertenecen a ET_1 más todas las tuplas que pertenecen a ET_2 [1].

Definición 2: **(Unión sin réplicas)** Dados dos espacios de tuplas ET_1 y ET_2 la union sin réplicas $ET_1 \bigcup ET_2$ es un espacio de tuplas que contiene las tuplas que pertenecen a ET_1 más las tuplas que pertenecen a ET_2, sin considerar repeticiones de tuplas [1].

A continuación, se muestra un ejemplo en el cual se podrá notar con claridad la diferencia que existe entre ambas operaciones. Por cuestiones de simplicidad, las tuplas de los ejemplos serán representadas por letras proposicionales *a*, *b*, *c* y *d*, debido a que por el momento sólo interesa el comportamiento de las operaciones y no como unifican las tuplas durante el proceso de la operación. En este trabajo, un espacio de tuplas ET_1 que contiene las tuplas *a*, *a* y *c* se denotará ET_1 = *[a,a,c]*. Para la implementación de éstas operaciones se debe proveer la definición de igualdad entre tuplas. Como en este trabajo en los ejemplos se utilizarán letras proposicionales para representar tuplas, dos tuplas son iguales si son representadas por la misma letra proposicional.

Ejemplo 1 *Considere los siguientes espacios de tuplas:* ET_1 *= [a,b,b,a,d] y* ET_2 *= [c,a,b,c,b].*

$ET_1 \uplus ET_2$ *= [a,b,b,a,d,c,a,b,c,b]* $\quad$ $ET_1 \bigcup ET_2$ *= [a,b,d,c]*

$ET_1 \uplus ET_1$ *= [a,b,b,a,d,a,b,b,a,d]* $\quad$ $ET_1 \bigcup ET_1$ *= [a,b,d]*

Aquí se puede observar que el resultado de la operación $ET_1 \uplus ET_2$ *mantiene las repeticiones de los dos espacios de tuplas, mientras que la operación* $ET_1 \bigcup ET_2$ *elimina toda repetición. También se puede observar que la operación "unión sin réplica" brinda la posibilidad de retornar el contenido sin réplicas de un ET realizando* $ET_1 \bigcup ET_1$*. Mientras que* $ET_1 \uplus ET_1$ *duplica el contenido del ET.*

2.2. Operaciones para intersección de espacios de tuplas

Estas operaciones permiten, por ejemplo, que un agente que está compartiendo conocimiento en diferentes ETs con distintos agentes, logre obtener en un único ET el conocimiento que comparte en común con los diferentes grupos de agentes.

Definición 3: **(Intersección total)** Dados dos espacios de tuplas ET_1 y ET_2 la intersección total $ET_1 \pitchfork ET_2$ es un espacio de tuplas que contiene todas las tuplas que pertenecen tanto a ET_1 como a ET_2 [1].

Definición 4: **(Intersección sin réplicas)** Dados dos espacios de tuplas ET_1 y ET_2 la intersección sin réplicas $ET_1 \bigcap ET_2$ es un espacio de tuplas que contiene tuplas que pertenecen tanto a ET_1 como a ET_2, sin considerar repeticiones de tuplas [1].

Ejemplo 2 *Considere los siguientes espacios de tuplas:* ET_1 *= [a,b,b,a,d] y* ET_2 *= [c,a,b,c,b].*

$ET_1 \pitchfork ET_2$ *= [a,b,b]* $\quad$ $ET_1 \bigcap ET_2$ *= [a,b]*

$ET_1 \pitchfork ET_1$ *= [a,b,b,a,d]* $\quad$ $ET_1 \bigcap ET_1$ *= [a,b,d]*

Aquí se puede observar que el resultado de la operación $ET_1 \pitchfork ET_2$ *admite tuplas repetidas en el caso de que existan en la intersección, mientras que la operación* $ET_1 \bigcap ET_2$ *elimina toda repetición existente en la intersección. Además se puede observar que al igual que en la "unión sin réplicas", la "intersección sin réplicas" brinda la posibilidad de retornar el contenido sin réplicas de un ET realizando* $ET_1 \bigcap ET_1$*. Esto es,* $ET_1 \bigcap ET_1 \equiv ET_1 \bigcup ET_1$*.*

2.3. Operaciones para diferencia de espacios de tuplas

Estas operaciones, por ejemplo, brindan la posibilidad de eliminar conocimiento de un espacio en base al conocimiento almacenado en otro. Esto genera que no se realice trabajo redundante. Es decir, supongamos que dos grupos de agentes se encargan de hacer lo mismo en base a información extraída de ETs diferentes, si estos ETs tienen tuplas que están en ambos y hacen referencia a un mismo trabajo, sería interesante sacarlas de alguno de los ETs para que el trabajo no sea duplicado. A continuación, se definen dos versiones nuevas de la operación que realiza diferencia de ETs vista en [1].

Definición 5: **(Diferencia existencial)** Dados dos espacios de tuplas ET_1 y ET_2 la diferencia existencial $ET_1 \ -_e \ ET_2$ es un espacio de tuplas cuyo contenido resulta de quitar (en forma existencial) de ET_1 aquellas tuplas que pertenecen a ET_2.

Definición 6: **(Diferencia sin réplicas)** Dados dos espacios de tuplas ET_1 y ET_2 la diferencia sin réplicas $ET_1 \ - \ ET_2$ es un espacio de tuplas cuyo contenido resulta de quitar de ET_1 aquellas tuplas (y sus respectivas réplicas) que pertenecen a ET_2.

Ejemplo 3 *Considere los siguientes espacios de tuplas:* ET_1 *= [a,b,b,a,d] y* ET_2 *= [c,a,b,c,b].*

$ET_1 -_e ET_2$ *= [a, d]* $\qquad$ $ET_1 -_e ET_1$ *= [ ]*
$ET_1 - ET_2$ *= [d]* $\qquad$ $ET_1 - ET_1$ *= [ ]*

Los casos $ET_1 -_e ET_1$ *y* $ET_1 - ET_1$ *permiten notar que si se aplica la "diferencia existencial" o la "diferencia sin réplicas" a un espacio de tuplas con si mismo se obtiene un espacio de tuplas vacío. En* $ET_1 - ET_2$ *se puede notar que a* ET_1 *se le quitaron todas las tuplas que están en* ET_2 *y también sus respectivas réplicas.*

2.4. Diferencia simétrica

Esta operación permite, por ejemplo, obtener en un espacio de tuplas el conocimiento que no tienen en común los espacios de tuplas que son operandos de la misma.

Definición 7: **(Diferencia simétrica)** Dados dos espacios de tuplas ET_1 y ET_2 la diferencia simétrica $ET_1 \sim ET_2$ es un espacio de tuplas que contiene todas las tuplas que pertenecen a ET_1 y no pertenecen a ET_2, más todas las tuplas que pertenecen a ET_2 y no pertenecen a ET_1 [1].

Ejemplo 4 *Considere los siguientes espacios de tuplas:* ET_1 *= [a,b,b,a,d] y* ET_2 *= [c,a,b,c,b].*

$ET_1 \sim ET_2$ *= [a, d, c, c]* $\qquad$ $ET_1 \sim ET_1$ *= [ ]*

El caso $ET_1 \sim ET_1$ *permite notar que si se aplica la "diferencia simétrica" a un espacio de tuplas con si mismo se obtiene un espacio de tuplas vacío.*

3. Trabajo relacionado, conclusiones y trabajo a futuro

Esta línea de investigación tiene como objetivo analizar el intercambio de conocimiento en sistemas multi-agentes utilizando áreas de conocimiento compartidas. Como un primer paso hacia este objetivo, en [1] hemos propuesto extender el modelo Linda definiendo nuevas operaciones que permiten que los agentes puedan manipular en forma masiva el conocimiento. Esto es, operaciones que tienen como dominio y rango ETs, lo cual hace al modelo aún más interesante y poderoso. En este artículo, además, se modificó la definición de la operación *diferencia* vista en [1].

Aunque el hecho de agregar más operaciones que se aplican sobre ETs hace que el modelo pierda *ortogonalidad*, consideramos que estas nuevas operaciones le agregan expresibidad cuando Linda se

aplica a un sistema multi-agente. Estas operaciones surgieron en forma natural a partir del hecho de que puedan existir varios espacios de tuplas en un mismo sistema, como lo proponen los trabajos [7, 8, 13, 9].

Como trabajo a futuro se planea definir diferentes propiedades sobre las operaciones propuestas. En base a estas propiedades se intentará buscar un conjunto mínimo de operaciones que le brinden a los programadores las herramientas necesarias para programar agentes que manipulan su conocimiento en forma masiva. Además se intentará combinar estas operaciones con nociones relacionas a la revisión de creencias para obtener versiones consolidadas de las mismas.

Referencias

[1] L. H. Tamargo, A. J. García, and M. A. Falappa. Reformulación del modelo linda para compartir conocimiento en sistemas multi-agente. *CACIC 2006*, pages 1801–1812, 2006.

[2] N. Carriero and D. Gelernter. Linda in context. *Comm. of the ACM*, 32(4):444–458, 1989.

[3] N. Carriero and D. Gelernter. Capitulo 3: Linda. In *How to write parallel programs*, 1992.

[4] Paolo Ciancarini. Coordination Languages as Software Integrators of Multiagent architectures. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 21(4):314–335, 1995.

[5] Paolo Ciancarini. Coordinating Multiagent Applications on the WWW: A Reference Architecture. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 24(5):362–374, May 1998.

[6] E. F. Codd. A relational model of data for large shared data banks. *Commun. ACM*, 13(6):377–387, 1970.

[7] K.K. Jensen. Toward a multiple tuple spaces. In *PhD thesis, Aalbrog University, Departament of Mathematics and Computer Science*, 1993.

[8] P. Butcher, A. Wood, and M. Atkins. Global synchronisation in Linda. *Concurrency: Practice and Experience*, 6(6):505–516, 1994.

[9] A. Rowstron and A. Wood. Bonita: a set of tuple space primitives for distributed coordination. In *Proc. HICSS30, Sw Track*, pages 379–388, Hawaii, 1997. IEEE Computer Society Press.

[10] David Gelernter. Multiple tuple spaces in linda. In *PARLE '89: Proceedings of the Parallel Architectures and Languages Europe, Volume II: Parallel Languages*, pages 20–27, London, UK, 1989. Springer-Verlag.

[11] Susanne Christine Hupfer. Melinda: Linda with multiple tuple space. 1990.

[12] Brian Nielsen and Tom Slrensen. PhD thesis.

[13] A. Rowstron, A. Douglas, and A. Wood. Copycollect: A new primitive for the linda model, 1996.

[14] Tanenbaum. Chapter 1 and 6. In *Distributed Operating Systems*, 1995.

[15] Paul W. P. J. Grefen and Rolf A. de By. A multi-set extended relational algebra - a formal approach to a practical issue. In *ICDE*, pages 80–88, 1994.