Procesamiento del Lenguaje Natural, num. 29 (2002), pp.55-62 recibido 27-05-2002; aceptado 01-07-2002

Selección automática del vocabulario definidor en un diccionario explicativo *

Alexander Gelbukh y **Grigori Sidorov**
Laboratorio de Lenguaje Natural,
Centro de Investigación en Computación (CIC),
Instituto Politécnico Nacional (IPN),
Av. Juan de Dios Bátiz, CP 07738, Zacatenco, México DF
{gelbukh, sidorov}@cic.ipn.mx, www.gelbukh.com

Resumen: Un diccionario semántico computacional empleado para los sistemas de la inferencia lógica e inteligencia artificial no debe contener ciclos en el sistema de definiciones. Incluso en un diccionario orientado al lector humano los ciclos muy cortos no son deseables. Sin embargo, los diccionarios explicativos tradicionales los contienen. Se presenta un algoritmo para la detección de tales ciclos y para la selección de un conjunto mínimo de las palabras primitivas a través de las cuales se pueden definir todas las demás palabras. Se describe una herramienta que ayuda al lexicógrafo a elegir tales palabras y corregir los defectos en el diccionario, relacionados con los ciclos en las definiciones.
Palabras clave: lexicografía computacional, diccionario semántico, definición, primitivos semánticos.

Abstract: A computational semantic dictionary to be used in the logical inference and artificial intelligence systems should not contain cycles in its definition system. Even in a dictionary oriented to the human reader, short cycles are undesirable. However, the traditional explanatory dictionaries do contain them. We present an algorithm for detecting such cycles and selecting a minimal set of the primitive words through which all other words can be defined. Also, we describe a tool that helps the lexicographer to choose such primitive words and to correct the defects in the dictionary related with the cycles in the definitions.
Keywords: computational lexicography, semantic dictionary, definition, semantic primitives.

1 Introducción

La manera natural para la construcción de un diccionario semántico orientado a los sistemas computacionales de la inferencia lógica e inteligencia artificial, es la definición de unas palabras a través de otras ya conocidas.

Por ejemplo, en matemáticas los términos se definen a través de otros de tal manera que en cualquier definición se puede sustituir cualquier término (digamos, *bisectriz*) por su definición (*línea que divide el ángulo en partes iguales*) sin alterar el sentido. El sistema de definiciones se construye de tal manera que al repetir este proceso iterativamente, se llega a una definición larga que consiste sólo de los términos que se llaman primitivos, tales como *punto* y *línea*.

Éstos últimos no tienen ninguna definición dentro de este sistema lógico, porque si la tuvieran se causarían círculos viciosos: el *punto* se definiría a través del mismo término, lo que se considera un problema grave para el razonamiento lógico. Entonces, cualquier sistema de definiciones lógicas sin círculos viciosos tiene que usar las palabras primitivas no definidas en este sistema. Este hecho es muy simple demostrar matemáticamente usando el modelo de grafo descrito más adelante en este artículo.

En la teoría lexicográfica también se reconoce que todas las palabras se deben definir usando unas pocas palabras primitivas, aunque

* Trabajo realizado con el apoyo parcial del Gobierno de México (CONACyT y SNI) y CGEPI-IPN, Mexico. Expresamos nuestro más cordial agradecimiento a Graeme Hirst y Ted Pedersen por sus útiles consejos y discusión.

hay controversias en las opiniones sobre el número de las primitivas necesarias, desde unas 60 (Wierzbicka 1980, 1996) hasta unos miles (Apresjan 1974, 1995).

En la lexicografía práctica se reconoce que un diccionario debe usar un número reducido de las palabras en las definiciones. Por ejemplo, el *Longman dictionary of contemporary English* usa en sus definiciones sólo las palabras del vocabulario definidor (*defining vocabulary*) acotado conocido como *Longman defining vocabulary* –alrededor de dos mil palabras.

En cuanto a los ciclos, es intuitivamente obvio que incluso en un diccionario orientado al lector humano (y mucho más en uno orientado a los sistemas de razonamiento automático) un conjunto cíclico de definiciones como:

gallina: hembra de gallo.
gallo: macho de gallina.

abeja: insecto que segrega miel.
miel: sustancia que producen las abejas.

convenio: pacto, acuerdo.
acuerdo: pacto, tratado.
tratado: convenio.

es un defecto en el sistema de definiciones, ya que equivale a decir *gallina es una hembra del macho de gallina*, lo que no ayuda a entender qué es una gallina sin saberlo de antemano.

Sin embargo, estas definiciones son ejemplos reales (un poco simplificados) del Diccionario Explicativo del ruso de Ozhegov (uno de los más usados; primer ejemplo) y el Diccionario Anaya de la Lengua Española.

Ahora bien, ¿se puede convertir un diccionario explicativo tradicional en un sistema de definiciones lógicas para el razonamiento automático en los programas de inteligencia artificial? De nuestra discusión es claro que para esto, en primer lugar, se requiere detectar y eliminar los círculos viciosos en las definiciones.

En algunos casos esto se puede lograr cambiando manualmente las definiciones. Sin embargo, un paso inevitable en este proceso es declarar algunas palabras primitivas, es decir, no definidas dentro de este sistema lógico, de tal manera que todas las demás palabras se definan a través de éstas, ya sea directamente o indirectamente. Conviene que este conjunto definidor sea lo menor posible.

En este artículo presentamos una solución basada en los métodos de la lexicografía computacional (Saint-Dizier y Viegas, 1995; Cheng-ming, 1996; Richardson, 1997; Rigau, 1998). Específicamente, hemos desarrollado una herramienta que permite al lexicógrafo detectar los círculos viciosos en el diccionario y elegir el conjunto definidor. La herramienta se basa en un algoritmo que automáticamente genera una variante del conjunto definidor mínimo (aunque no el menor posible; la diferencia se explicará en la sección 4.1).

En el resto del artículo, primero describimos la herramienta y después explicamos brevemente el algoritmo mencionado. Para esto, definimos la estructura de datos que usamos para la investigación del diccionario. Después describimos el algoritmo, la metodología experimental y los resultados de nuestros experimentos con un diccionario real. Finalmente, mencionamos las tareas futuras y formulamos las conclusiones.

2 La herramienta

Basándonos en el algoritmo que se describirá más adelante, hemos desarrollado una herramienta que permita al lexicógrafo investigar la estructura del diccionario con el fin de detectar y corregir los círculos viciosos cortos (lo que es útil incluso para los diccionarios orientados al lector humano) o todos. Lo último se hace eligiendo un vocabulario definidor con ciertas características (es necesario para la compilación de los diccionarios orientados a los sistemas de inferencia lógica automática).

La herramienta proporciona la siguiente información:

- Muestra varias características de la palabra, tales como su frecuencia en las definiciones del diccionario, el tamaño de su propia definición, el largo mínimo del ciclo en que está involucrada, etc.
- Genera diferentes conjuntos definidores mínimos permitiéndole al usuario seleccionar los parámetros del algoritmo de su generación.
- Permita al lexicógrafo cambiar manualmente el conjunto generado y verifica que el conjunto cambiado todavía es un conjunto definidor y que es mínimo.
- Permita al lexicógrafo cambiar las definiciones de algunas palabras e investigar el impacto a los conjuntos definidores que se generan.
- Dado una lista de las palabras que el lexicógrafo quiere que sean *no* primitivas, verifica si existe algún conjunto definidor que

no las contiene. Éste existe siempre y cuando las palabras elegidas no formen círculos viciosos. Si así es, genera una o varias variantes de tal conjunto. Si no es así, muestra los círculos, lo que ayuda a eliminar de la lista las palabras que los causan.

- Dado una lista de las palabras que el lexicógrafo quiere que *sí* sean definidoras, genera uno o varios conjuntos definidores que contengan estas palabras. Si tal conjunto definidor no puede ser mínimo, sugiere eliminar ciertas palabras de la lista.

Dado un conjunto definidor mínimo, la herramienta puede:

- Para una palabra no primitiva, mostrar su definición expandida a las palabras definidoras, es decir, la que consiste sólo de las palabras definidoras.
- Para una palabra primitiva, mostrar los ciclos (más cortos o todos) que su definición actual causa en el diccionario.

La interpretación de los resultados se discuta en la sección 6, donde se menciona que en un buen diccionario, el conjunto definidor no debe tener muchas palabras con frecuencia baja. Lo que ayuda al lexicógrafo detectar y corregir ciertos defectos cambiando las definiciones.

3 La estructura de datos

Para el funcionamiento del algoritmo, así como para las definiciones y discusiones matemáticas, representamos el diccionario como un grafo dirigido (Evens, 1988; Fellbaum, 1990; Kozima y Furugori 1993).

Los vértices de este grafo son las palabras que se mencionan en el diccionario –tanto las palabras encabezado como las que se usan en las definiciones. Si la misma palabra ocurre en diferentes contextos, se cuenta como el mismo vértice.

Las flechas del grafo se definen como sigue: la flecha desde la palabra v_1 hasta la palabra v_2 significa que en la definición de la palabra v_1 ocurre la palabra v_2. Las palabras que no se definen en el diccionario no tienen las flechas salientes, y las que no se usan en las definiciones de otras palabras, no tienen las flechas entrantes.

Nótese que hay diferentes maneras de considerar que dos ocurrencias textuales corresponden a «la misma palabra»: 1) cuando coinciden como cadenas de letras, 2) por el lema, 3) por la raíz común, 4) por el significado específico en el cual se usan en el contexto dado, etc. En la sección 5 se dan más detalles sobre los dos métodos que aplicamos (Grafo 1 –por el lema– y Grafo 2 –por significado).

4 El algoritmo

Esta sección se orienta al lector interesado en los aspectos matemáticos y técnicos del algoritmo. El lector interesado sólo en las aplicaciones y resultados, puede omitirla.

Desde el punto de vista matemático, el problema y su solución son los siguientes[1].

4.1 Definiciones

Sea $\mathbf{G} = \{\mathbf{V}, \mathbf{F}\}$ un grafo dirigido (digrafo) definido por los conjuntos $\mathbf{V}$ de N vértices y $\mathbf{F} \subseteq \mathbf{V} \times \mathbf{V}$ de flechas. Por un ciclo en este grafo, entenderemos un ciclo dirigido.

Sea un subconjunto $\mathbf{P} \subseteq \mathbf{V}$ un *conjunto definidor* si cualquier ciclo en el grafo $\mathbf{G}$ contiene un vértice de $\mathbf{P}$. En otras palabras, si el grafo $\mathbf{G} = \mathbf{G} - \mathbf{P} \stackrel{\text{def}}{=} \{\mathbf{V}, \mathbf{F}\}$, donde $\mathbf{V} = \mathbf{V} \setminus \mathbf{P}$ y $\mathbf{F} = \mathbf{F} \cap (\mathbf{V} \times \mathbf{V})$, no tiene ciclos. Llamaremos los vértices $p \in \mathbf{P}$ los definidores.

Un conjunto definidor $\mathbf{P} \subseteq \mathbf{V}$ es *mínimo* si ningún subconjunto $\mathbf{P} \subset \mathbf{P}$ es definidor. Es decir, para cada vértice $p \in \mathbf{P}$, existe un ciclo en $\mathbf{G}$ que contiene p y no contiene ningún otro vértice de $\mathbf{P}$.

El conjunto definidor mínimo no tiene que ser el menor (el que se contiene en todos los conjuntos definidores; tal subconjunto usualmente no existe) ni siquiera del tamaño menor posible: como es muy fácil mostrar con ejemplos y como también será claro de nuestro algoritmo, en el mismo grafo pueden existir muchos conjuntos definidores mínimos de tamaños diferentes.

El algoritmo que aquí presentamos resuelva el siguiente problema: *dado un grafo dirigido* $\mathbf{G}$, *encontrar un subconjunto definidor* $\mathbf{P}$ *mínimo, aunque no del tamaño menor posible.*

El algoritmo encuentra una de los muchísimos posibles conjuntos definidores mínimos. La selección de la variante se puede controlar usando un ordenamiento σ que ordena los números de 1 a N en una secuencia $\sigma(1), ..., \sigma(N)$.

[1] En este artículo no es nuestra tarea presentar un algoritmo del costo computacional óptimo. Aunque existe toda una clase de algoritmos con mejor complejidad –los llamados dinámicos– son más complejos y difíciles de entender y realizar (Demetrescu y Italiano. 2000).

1. Formar el conjunto de los definidores $\mathbf{P} = \varnothing$ y de los no definidores $\mathbf{V} = \varnothing$.
2. Paso opcional: depuración del grafo, véase más adelante el algoritmo B. En este paso se pueden agregar elementos al **P** y eliminar vértices (y sus arcos) del **G**. A continuación se supone que **V**, **F** y N son definidos por el grafo **G** ya depurado.
3. Para $i = 1, ..., N$ repetir:
4. Seleccionar $\sigma(i)$-ésimo vértice $v \in \mathbf{V}$.
5. Verificar si v se puede agregar al grafo sin causar ciclos. Para esto:
6. Para cada par de flechas w, $u \in \mathbf{V}$ tal que $(w \to v)$, $(v \to u) \in \mathbf{F}$ repetir:
7. Verificar que $w \notin \mathbf{A}_u$.
8. Si una de estas pruebas falla, agregar v a **P**.
9. En el caso contrario, agregarlo a **G** . Para esto:
10. Agregar v a **V** .
11. Formar $\mathbf{A}_v = \cup \mathbf{A}_u$, donde la unión es por las flechas $u \in \mathbf{V}$ tales que $(v \to u) \in \mathbf{F}$.
12. Para cada flecha $w \in \mathbf{V}$, $(w \to v) \in \mathbf{F}$ repetir:
13. Para cada vértice $q \in \mathbf{V}$ tal que $q = w$ ó $w \in \mathbf{A}_q$ repetir:
14. Agregar $\{v\} \cup \mathbf{A}_v$ a $\mathbf{A}_q$.

Figura 1: El algoritmo A.

Por ejemplo: 3, 5, 2, 4, 1 es un ordenamiento para $N = 5$.

Los conjuntos **P** generados basándose en diferentes ordenamientos σ son usualmente diferentes. Como se verá del algoritmo, el sentido del ordenamiento es el siguiente: los primeros vértices tienden ser no definidores y los últimos tienden entrar en **P**. Específicamente, el primer vértice en el ordenamiento siempre no es definidor (si no tiene un lazo[2]).

Este ordenamiento se puede definir de antemano, o bien generar el siguiente número en cada paso del algoritmo de entre los números todavía no usados según alguna estrategia (nosotros usamos este método).

4.2 Funcionamiento

Ahora bien, dado **G** y σ, el algoritmo funciona como sigue. Se construye, paso a paso, un subgrafo acíclico $\mathbf{G} \subseteq \mathbf{G}$. Al inicio, es vacío. Se construye insertándole uno por uno –en el dado orden σ– los vértices de **G** (y los arcos que los conectan con los vértices ya insertados). En cada paso, **G** se mantiene acíclico: si el vértice a insertar le causa ciclos, se considera un definidor y no se inserta en **G** . Al terminar el proceso, se tiene el conjunto definidor **P**, que por su construcción es mínimo (porque cada $p \in \mathbf{P}$ tiene ciclos incluso en un subgrafo del **G**).

El paso computacionalmente más costoso del algoritmo es la verificación de que el nuevo vértice no le causaría ciclos al **G** . Para esto, para cada vértice $v_i \in \mathbf{V}$ el algoritmo mantiene un conjunto $\mathbf{A}_i$ de vértices alcanzables en **G** desde v_i, diciéndose del vértice u que es alcanzable desde v si existe un camino dirigido en el grafo desde v hacia u. El algoritmo aprovecha que es una relación transitiva.

También el algoritmo mantiene el conjunto **V** de los vértices ya incluidas en el **G** y el conjunto **P**. Al terminar el algoritmo, **P** será un conjunto definidor mínimo. Véase el algoritmo detallado en la Figura 1.

Dado que el tamaño de una definición (el número de flechas salientes) en un diccionario es limitado, el algoritmo tiene complejidad cuadrática en N, siendo las operaciones computacionalmente más pesadas los pasos 7 y 14 (la demostración es fuera del alcance de este artículo). Dado el gran tamaño del diccionario ($N = 30$ mil en nuestro caso), empleamos un paso adicional (paso 2) para disminuir el tamaño del grafo antes de la aplicación del algoritmo, como se describe en la siguiente subsección.

4.3 Depuración inicial del grafo

Para disminuir el tamaño de los datos a procesar, se pueden observar los siguientes hechos:

- Los vértices que no tienen las flechas entrantes tienen que estar en **P**. Estos vértices –aunque son pocos– se pueden de antemano agregar al **P** y quitar de **G**.
- Los vértices v con un lazo $(v \to v) \in \mathbf{F}$ también tienen que estar en **P** pues no pueden estar en **G** .
- Los vértices que tienen sólo flechas entrantes o sólo flechas salientes, no pueden estar en **P** (recuérdese que **P** es un conjunto mínimo). Entonces, estos vértices –en nuestro

[2] Un lazo es un arco que va desde un nodo al mismo nodo, o sea, un ciclo del largo 1.

```
1.  Para cada vértice con lazo o sin flechas entrantes, repetir:
2.      Agregarlo al P y eliminarlo del G.
3.  Mientras los siguientes pasos cambian G repetir:
4.      Para cada vértice que no tiene flechas salientes, o entrantes repetir:
5.          Eliminarlo de G considerándolo no definidor.
```

Figura 2: El algoritmo B.

caso resultaron 20 mil– se pueden de antemano quitar de **G** considerándolos no definidores.

La eliminación de un vértice puede hacer que otros vértices pierden todas sus flechas entrantes o salientes. El algoritmo que remueve los vértices redundantes del grafo se muestra en la Figura 2.

El grafo **G** que se obtiene después de la depuración, satisface las siguientes condiciones:

- No tiene lazos.
- Cada vértice tiene tanto las flechas entrantes como salientes.

En tal grafo, para cada vértice v existe un conjunto definidor mínimo **P** tal que $v \notin \mathbf{P}$. Más generalmente, para cada conjunto compatible $\mathbf{Q} \subseteq \mathbf{V}$ (diciéndose del conjunto **Q** compatible si no existen ciclos en **G** que contengan únicamente los vértices del **Q**) existe un conjunto definidor mínimo **P** tal que $\mathbf{Q} \cap \mathbf{P} = \varnothing$. Esto se comprueba por la aplicación del algoritmo A con un ordenamiento σ que empieza con los elementos del **Q**.

Entonces, en tal grafo la propiedad de ser definidor no es propia del vértice sino sólo es relativa a la selección de un conjunto **P**. Es decir, ningún vértice –salvo los vértices con lazos y los que ya no tuvieron ninguna definición en el diccionario inicial– no es un definidor «de por sí», o bien, que entre en cualquier conjunto definidor.

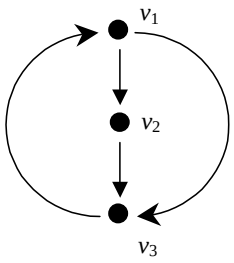

Figura 3: Contraejemplo.

Lo contrario no es cierto: en el grafo depurado con el algoritmo B todavía pueden existir los vértices que no entran en ningún conjunto definidor mínimo, véase la Figura 3, el vértice v_2. Nosotros no tenemos ningún algoritmo rápido para detectar tales vértices. Sin embargo, nuestros experimentos con el diccionario real mostraron que en este diccionario los vértices de este tipo, si existen, no constituyen más de 20% del grafo depurado, pues encontramos los conjuntos definidores mínimos cuya unión cubre unos 80% del grafo. Entonces, detección previa de tales vértices no contribuiría significativamente en el rendimiento del algoritmo.

5 La metodología experimental

Como se explica en la sección 4.1, el comportamiento del algoritmo depende del ordenamiento de los vértices del grafo. Nosotros no sabemos ningún algoritmo que encuentre el conjunto del tamaño menor posible. Probamos los siguientes ordenamientos.

Método 1: *aleatorio, uniforme*. Usamos el ordenamiento uniformamente aleatorio.

Método 2: *por frecuencias*. Ordenamos los vértices por la frecuencia de su uso en las definiciones del mismo diccionario, desde menor a mayor. Entonces, los vértices con menor frecuencia tendieron que entrar en **G** y los con mayor frecuencia tendieron ser definidores y entrar en **P**. Esperamos que con esta heurística, **P** sería menor porque los vértices que lo forman rompen más ciclos en **G**.

Método 3: *aleatorio, por frecuencias*. Este método es una combinación de los métodos 1 y 2. Usamos el ordenamiento aleatorio, pero con las probabilidades en función inversa a las frecuencias. Esperamos que alguna alteración del ordenamiento rígido del método 2 produciría un conjunto menor.

Método 4: *por votación aleatoria*. En este método, generamos 20 diferentes conjuntos definidores mínimos $\mathbf{P}_i$ con el método 1, y para cada vértice, contamos el número de los conjuntos $\mathbf{P}_i$ en los cuáles éste entra, así asociando con cada vértice un peso entre 0 y 20. Enumeramos primero los vértices con el peso 0 usando sus frecuencias como en el método 2, y después los que entraron, en el orden inverso de su peso, desde 1 a 20. Esperamos que los que entraron en un mayor número de los conjuntos $\mathbf{P}_i$ eran

Grafo	En total	No definidores	Depurado
Lexemas	30725	20366	**10359**
Significados	60818	47802	**13016**

Tabla 1: Número de vértices en los grafos.

Grafo	Método 1. Aleatoriamente, uniformamente	Método 2. Por frecuencias	Método 3. Aleatoriamente, por frecuencias	Método 4. Por votación aleatoria
Lexemas	2789, $s = 25$	2302	2770	**2246**
Significados	2266, $s = 28$	1955	2257	**1913**

Tabla 2: Número de definidores, con diferentes algoritmos.

los «mejores» definidores y debieron entrar en el conjunto que buscamos.

Cabe mencionar, que unos 80% de los vértices entraron por lo menos en un conjunto $\mathbf{P}_i$.

Hicimos dos experimentos con dos grafos diferentes, obteniendo variantes de conjuntos compuestos por los elementos de naturaleza diferente.

Grafo 1: *lexemas*. En este experimento, consideramos un vértice del grafo un lexema, es decir, una palabra normalizada morfológicamente: *piensa*, *pensó*, *pensaríamos* se contaron como el nodo `pensar`. No aplicamos ninguna resolución de ambigüedad, asignando las cadenas ambiguas a varios nodos. En este método, las definiciones de todos los significados de la palabra –o, en su caso, de todos los homónimos de un lema– se consideraron como una sola definición. Es decir, `gato` fue una unidad de consideración definida como ‘animal doméstico o bien herramienta para la reparación de coches .

Consecuentemente, los elementos primitivos encontrados fueron lexemas sin distinguir sus significados. Lo que simplifica el procedimiento pero limita la interpretación semántica del conjunto encontrado.

Grafo 2: *significados*. En este experimento, consideramos un vértice del grafo un significado específico de palabra, por ejemplo: $\texttt{gato}_{1a}$ ‘animal doméstico , $\texttt{gato}_{1b}$ ‘tipo de animales salvajes o domésticos , $\texttt{gato}_{2a}$ ‘herramienta , etc. Para desambiguar los significados de las palabras que forman las definiciones, empleamos un etiquetador (*tagger*) para la normalización morfológica[3] y desambiguación de la categoría gramatical, y después utilizamos un algoritmo parecido al de Lesk (Sidorov y Gelbukh, 2001) para desambiguar el significado de la palabra en el contexto.

Consecuentemente, los elementos primitivos encontrados fueron significados específicos de las palabras, siendo $\texttt{gato}_{1a}$ ‘animal y $\texttt{gato}_{2a}$ ‘herramienta elementos distintos. Lo que justifica más el referir a los elementos encontrados como primitivas *semánticas*.

En ambos casos, sólo se consideraron las palabras significativas, es decir, no se consideran las preposiciones, conjunciones, verbos auxiliares, etc. Nótese que en el Grafo 2, el número de las flechas es exactamente el mismo que el número de las palabras significativas en las definiciones del diccionario, mientras que en el Grafo 1 este número es ligeramente mayor por la ambigüedad léxica.

Los resultados de estos experimentos y su discusión se presenten a continuación.

6 Los resultados y discusión

Para nuestros experimentos usamos el Diccionario de la Lengua Española del grupo Anaya. Este diccionario contiene 30971 artículos, de los cuales sólo 30725 correspondan a las palabras significativas divididas entre 60818 significados específicos.

La aplicación del algoritmo de depuración –algoritmo B– a las dos variantes del grafo redujo cada uno de éstas a unos 10 mil vértices, véase tabla 1. A groso modo, se debe a que unas 20 mil palabras no se usan en las definiciones de otras palabras.

[3] El analizador morfológico que utilizamos fue basado en el corpus LEXESP desarrollado en la Universidad Politécnica de Cataluña y amablemente puesto a nuestra disposición por el Dr. Horacio Rodríguez.

Un resultado inesperado fue que este núcleo de unos 10 mil vértices es muy fuertemente interconectado: prácticamente de cualquier palabra del diccionario están alcanzables todas las demás palabras de este núcleo. La tarea de la selección del conjunto definidor en un grafo tan fuertemente interconectado es computacionalmente difícil.

A estos dos conjuntos de 10 mil palabras, aplicamos el algoritmo A con las 4 variantes de ordenamiento. Los resultados se presentan en la tabla 2, mostrándose los tamaños de los conjuntos definidores obtenidos.

Para el método 1, se muestra el promedio de los 20 experimentos y la desviación cuadrática promedia *s* (los 67% de los casos desvían del promedio no mas que en *s* y 99% no más que en 3*s*). Atrae atención que la desviación es muy baja, lo que significa que con el método 1, los tamaños de los conjuntos obtenidos son diferentes pero muy parecidos.

Para el método 3, sólo mostramos el resultado de un experimento. Fue sorprendente que los resultados con el método 3 casi no diferían de los obtenidos con el método 1, a pesar de que las probabilidades en este caso correspondieron a las frecuencias del método 2.

Éste último mostró un muy buen desempeño produciendo los conjuntos definidores mucho menores. Sin embargo, el método 4 produjo los menores conjuntos que hemos obtenido.

Aunque creemos que con métodos más sofisticados, se puede obtener los conjuntos aun menores, no esperamos que el tamaño mínimo del conjunto definidor sea mucho menor que los que hemos obtenido, de aproximadamente 2000 palabras. Esto se debe a las siguientes consideraciones lingüísticas: según la opinión común, se supone que 2 mil es el número de las palabras suficientes para definir todas las demás palabras del vocabulario general. Éste es el tamaño del vocabulario definidor de Longman. Según nuestro conocimiento, éste es el número de los hieróglifos en el vocabulario chino básico. Creemos que el hecho de que el tamaño de nuestro conjunto definidor tan exactamente corresponde a la cifra esperada –2 mil– es muy significativo.

Consideremos unos ejemplos de las palabras elegidas con el Método 4, Grafo 1. Las 20 palabras con la mayor frecuencia de las flechas entrantes (las mejores) son:

> *cosa, persona, acción, hacer, efecto, tener, parte, no, conjunto, dar, forma, cierto, cuerpo, relativo, nombre, poder, uno, formar, producir, animal, común, general, determinado, poner, estado, tiempo, decir, planta, obra, etc.*

Son buenos candidatos a los primitivos semánticos.

Otras palabras entraron en el conjunto definidor porque están involucradas en ciclos cortos, aunque tienen una frecuencia baja (las peopres):

> *almuerzo, almuédano, almanaque, alinear, algarroba, alarmar, ahíto, etc.*

Estas palabras son buenos indicadores de la necesidad de cambiar las definiciones en el diccionario para que se excluyen de la lista de las definidoras.

Finalmente, algunas palabras tienen que estar en cualquier conjunto definidor pues tienen lazos. Éstas indican o bien algún problema con el algoritmo de la identificación de las palabras, o bien una definición errónea en el diccionario; encontramos 47 casos:

> *ático, borgoña, lapón*, etc.

Es interesante investigar el largo de los ciclos en los cuales estarían involucradas las palabras definidores si fuesen insertadas en el diccionario (es decir, estos ciclos consisten sólo de las palabras no primitivas). Ejemplos de tales ciclos son:

1: *ático* → *ático*
2: *premura* → *prisa* → *premura*
3: *grano* → *cereal* → *centeno* → *grano*,

etc. En nuestro ejemplo los largos de tales ciclos se distribuyeron como sigue:

L	n	L	n	L	n
1	47	7	53	13	19
2	1496	8	58	14	9
3	177	9	45	15	8
4	67	10	38	16	12
5	47	11	29	17	11
6	72	12	32	18	3

donde *L* es el largo del ciclo más corto causado por la palabra dada y *n* es el número de las palabras con tales ciclos. Sólo mostramos aquí los primeros 18 elementos. El largo máximo del ciclo fue 52.

7 El trabajo futuro

Este artículo presenta los resultados preliminares de nuestra investigación. Las tareas princi-

pales a investigar se agrupan en dos clases de problemas:

- El problema lingüístico: la interpretación lingüística de los resultados y
- El problema técnico: el diseño del algoritmo que encuentre un conjunto **P** óptimo en un sentido dado –técnico y/o lingüístico–.

Con mayor detalle, las tareas futuras específicas son las siguientes:

- Dar una interpretación lingüística clara al conjunto **P** obtenido.
- Elaborar unos criterios lingüísticos que permitirían preferencias en el proceso de inclusión de las palabras en el conjunto definidor (preferir que una palabra sea o no sea primitiva).
- Mejorar el costo computacional del algoritmo para resolver el problema formulado en la sección 4.1, usando un algoritmo dinámico (Demetrescu y Italiano, 2000).
- Desarrollar un algoritmo –sea exacto o aproximado– para la construcción de un conjunto **P** del tamaño menor posible.

Sobre esta última tarea, una de las ideas técnicas que vamos a probar es un algoritmo genético para la construcción del conjunto **P** del tamaño menor posible, en el modo similar al método 4 de la sección 5.

8 Conclusiones

Hemos presentado un método para la selección en un diccionario explicativo, del conjunto mínimo de las palabras a través de las cuales se pueden definir todas las demás palabras en este diccionario, llamado conjunto definidor.

La construcción de tal conjunto se necesita para la conversión del diccionario tradicional en un diccionario semántico computacional orientado a los sistemas de razonamiento lógico automático, siendo un rasgo de tales sistemas lógicas que no se permiten los círculos viciosos en las definiciones.

Nuestro método permitió la construcción de una herramienta que detecta los problemas y defectos en las definiciones del diccionario relacionados con la presencia de los círculos, y ayuda al lexicógrafo a corregirlos.

Para la investigación futura se queda la interpretación lingüística del hecho de que en el diccionario con el cual experimentamos encontramos casi exactamente el número esperado de las palabras primitivas –dos mil.

Referencias

Apresjan, J. D. 1974. Regular polysemy, *Linguistics*. 142: 5–32.

Apresjan, J. D. 1995. *Selected works* (in Russian). V 1, 472 p., V 2, 768 p. Moscow.

Cheng-ming, Guo (ed.). 1996. *Machine Tractable Dictionaries: Design and Construction*, Ablex Publishing Corporation.

Demetrescu, C., G.F. Italiano. 2000. Fully Dynamic Transitive Closure: Breaking Through the $O(n^2)$ Barrier, *Proc. of the 41st Annual IEEE Symposium on Foundations of Computer Science* (FOCS'00), Redondo Beach, Los Angeles, CA. November 12–14.

Evens, M. N. (ed.). 1988. *Relational models of lexicon: Representing knowledge in semantic network*. Cambridge: Cambridge University Press.

Fellbaum, C. 1990. The English verb lexicon as a semantic net. *International Journal of Lexicography* 3: 278–301.

Kozima, H. and Furugori, T. 1993. Similarity between words computed by spreading activation on an English dictionary. *Proc. of the 6th conference of the European chapter of ACL*, pp. 232–239.

Richardson, S. 1997. *Determining Similarity and Inferring Relations in a Lexical Knowledge Base*. Ph.D. thesis, The City University of New York, 187 pp.

Rigau, G. 1998. Automatic *Acquisition of Lexical knowledge from MRDs*. PhD Thesis, Departament de Llenguatges i Sistemes Informàtics, Universitat Politécnica de Cataluña, Barcelona.

Saint-Dizier, P., Viegas, E. (eds.). 1995. *Computational lexical semantics*. Cambridge: Cambridge University Press, 447 p.

Sidorov, G., A. Gelbukh. 2001. Word sense disambiguation in a Spanish explanatory dictionary. *Proc. of TALN-2001*, Tours, France, July 2–5, pp. 398–402.

Wierzbicka, A. 1980. *Lingua Mentalis: The semantics of natural language*. New York: Academic Press.

Wierzbicka, A. 1996. *Semantics: Primes and Universals*. Oxford: Oxford University Press.