Aplicación de la teoría de agentes al modelo de Grafos para la detección de patrones en textos

Fernando Carlos Federico
Facultad de Ingeniería
Universidad de Buenos Aires
fenandofederico1984@yahoo.com.ar

Juan M. Ale
Facultad de Ingeniería
Universidad de Buenos Aires
ale@acm.org

Resumen
Text mining puede ser definido como el descubrimiento de conocimiento en grandes colecciones de documentos. Se asocia principalmente al descubrimiento de patrones interesantes como clusters, asociaciones, desviaciones, similitudes, y diferencias. Por otro lado, los Attributed Relational Graphs (ARG) se definen como una extensión de los grafos ordinarios asociando atributos discretos o reales a sus vértices y arcos. El uso de los atributos permite a los ARG ser posibles de no sólo modelar estructuras topológicas de una entidad sino también sus propiedades no estructurales, que usualmente se pueden representar como vectores. Estas características hacen a esta herramienta un elemento útil a la hora de realizar búsqueda de patrones. Es por ello que, en este trabajo se define un algoritmo basado en grafos para la detección de patrones de textos. Debido a que el volumen de información que se debe procesar es grande, dicho algoritmo contempla la aplicación del modelo de agentes para controlar de manera dinámica el espacio de búsqueda y, en consecuencia, reducir los tiempos de procesamiento de los textos.

Palabras clave: Text Mining, Attributed Relational Graphs, Agentes.

1. Introducción

El motivo de este trabajo se encuentra fundamentado en tres temas pilares, Text Mining, los Attibuted Relational Graphs y el concepto de agentes.

Text mining es un área de investigación emergente que puede ser caracterizada como el descubrimiento de conocimiento en grandes colecciones de documentos, combinando métodos de aprendizaje con métodos de procesamiento de textos. Está asociado principalmente al descubrimiento de patrones interesantes como clusters, asociaciones, desviaciones, similitudes, y diferencias [4, 5,13]. Asimismo, la detección de patrones en textos es una herramienta útil en aplicaciones para reconocimiento inteligente de caracteres, sistemas de compresión de texto, traducciones automáticas, y aplicaciones similares en las que un sistema debe elegir el siguiente elemento (letra, palabra, fonema, etc...) de entre una lista de posibles candidatos [3].

Por otro lado, los Attributed Relational Graphs (ARG) se definen como una extensión de los grafos ordinarios asociando atributos discretos o reales a sus vértices y arcos. El uso de los atributos permite a los ARG ser posibles de no sólo modelar estructuras topológicas de una entidad sino también sus propiedades no estructurales, que usualmente se pueden representar como vectores [17]. Estas características hacen a esta herramienta un elemento útil a la hora de detectar de patrones [14, 15, 16].

Finalmente, existe una gran variedad de definiciones de agentes dependiendo del uso que el autor le haya dado al término. Según la definición de agente de IBM, los agentes son entidades de

software que ejecutan un conjunto de operaciones en nombre de un usuario u otro programa con cierto grado de independencia o autonomía, utilizando algún conocimiento o representación de las metas y deseos del usuario [6]. Sin embargo, investigadores de la Universidad de Indiana forman parte de una comunidad que apoya el enfoque basado en agentes cuyo conocimiento y capacidad deductiva sea limitado. Dicho enfoque propone que un conjunto de agentes simples permite la ejecución de un sistema inteligente de forma más sencilla [2].

En este contexto, el presente trabajo define un modelo genérico basado en los ARG para la búsqueda de patrones, con la incorporación de la teoría de agentes para la reducción de caminos irrelevantes en el grafo. Este comportamiento se modeliza mediante la metodología BDI (Belief, Desire and Intentions) [10], que permite representar el accionar de un agente a partir del conocimiento que posea, las metas y las intenciones del usuario. En resumen, se precisa un algoritmo para detección de patrones que considera algunas características no contempladas en trabajos anteriores (*ver sección 1.2*), a saber:

1. Flexibilidad en el tipo de patrón que se desea detectar.
2. Control del volumen del grafo mediante la reducción de caminos irrelevantes.
3. Velocidad en la detección de patrones.

Se presenta un modelo genérico y fácilmente adaptable al tipo de texto que se desee procesar. Un campo de aplicación posible es el de la clasificación de páginas Web a partir de los patrones de textos que se detecten en ellas (Proyecto de Web Semántica) adaptando el modelo a, por ejemplo, el idioma del sitio y longitud del texto.

El resto del trabajo está organizado de la siguiente manera. En la segunda sección, se define formalmente el grafo que modeliza a un texto para la detección de patrones.. Con el fin de reducir el espacio de búsqueda y agilizar el proceso de detección de patrones, se presenta, en la tercera sección, un método para la selección de los arcos más relevantes del nodo. Para poder llevar adelante dicha tarea mientras se procesa el texto, se consideró a cada uno de los nodos del grafo como un agente cuya meta es la de maximizar el promedio del peso de los arcos que parten de él, dicha modelización se encuentra plasmada en la cuarta sección. La quinta sección presenta un estudio de performance comparando la detección de patrones mediante la utilización de agentes y sin ella. En la sexta sección se presentan las conclusiones y líneas de trabajos futuros.

1.2 Trabajos Relacionados

Existen trabajos en la bibliografía actual que utilizan la representación de textos mediante grafos. Por ejemplo, en (Wei Jin, Rohini Srihari [7]) se propone un método basado en grafos para capturar la estructura y la semántica del documento de forma más efectiva. Básicamente el modelo se basa en un grafo que pondera las relaciones entre los elementos del documento que analiza y extrae conclusiones estructurales a partir de ellas.

En (Tomita,Nakawatase, Ishii [8]), el articulo se concentra principalmente en determinar los algoritmos para obtener subgrafos a partir del grafo original y calcular la similaridad entre ellos. Por otro lado, en (Tomita,Nakawatase, Ishii [3]) se define una arquitectura para representación de paginas Web mediante grafos y su posterior almacenamiento para el descubrimiento de conocimiento en la Web.

Finalmente, existen publicaciones, como (Tsuyoshi Kitani[11]) que basan su trabajo en un idioma en particular (en el caso de [11], el Japonés).

La diferencia de este trabajo con los otros es que se presenta un modelo genérico para la búsqueda de patrones en textos que permite al usuario definir mediante una función peso el tipo de

patrón que desea buscar y que realiza un control sobre el volumen del grafo para obtener mayor velocidad de procesamiento. Es decir, lo determinado en este artículo es independiente del objetivo que tenga el usuario, el idioma o las características del procesamiento que se le quiera dar al texto.

2. Definición del Modelo

2.1 Definición del grafo

Los grafos son estructuras útiles para esquematizar relaciones entre elementos. Es debido a esto que, para la modelización del texto, se utiliza un grafo donde los gramas sean nodos y la relación entre ellos, arcos.

Un grafo es una tripla ordenada $\langle V(G), E(G), \psi_G \rangle$, y la correcta formalización del grafo depende de la definición de la misma. Para ello se exponen las siguientes definiciones.

Definición 1: *Sea* Σ *un alfabeto cualquiera, entonces V (G) es el espacio definido por el producto cartesiano* $V : \Sigma x \Re$.

Acorde a esto, sea un nodo $v \in V$, éste se puede representar como el vector $v = [\alpha, f]$, siendo α un elemento del alfabeto Σ (un grama) y $f \in \Re$ la frecuencia de aparición del grama, dada por la ecuación:

$$f = \frac{N_\alpha}{N_t} \qquad [1]$$

Donde N_α es el número instancias de α (apariciones en el texto), y N_t el número total de instancias en el texto.

Definición 2: *E(G) es el espacio definido por el producto cartesiano* $E : VxVx\Re$.

A partir de esta definición se interpreta que, sea $e \in E, e$ se puede representar como $e = [v_1, v_2, w]$, siendo $v_1, v_2 \in V$ y $w \in \Re$, el peso del arco.

Finalmente, sean $e \in E, u, v \in V$ y $\psi_G(e) = uv$ se dice que e une a u y v, y éstos son extremos de e. En nuestro cado, esta función puede leerse como "*u precede a v en el texto*".

ψ_G cumple las siguientes propiedades:

1) Inyectividad:

$e_1, e_2 \in E$, $u, v \in V, \psi_G(e_1) = \psi_G(e_2) \leftrightarrow e_1 = e_2$

2) Grafo Dirigido:

Sean $e_1, e_2 \in E, u, v \in V, \psi_G(e_1) = uv$ y $\psi_G(e_2) = vu \rightarrow \psi_G(e_1) \neq \psi_G(e_2)$

Ya definida la tripla, solo resta formalizar la definición del grafo que se utilizará a lo largo del documento:

Definición 3 (grafo del modelo): *se define* $H = [V_H, E_H, \psi_G]$ *como un grafo dirigido ponderado, donde* $V_H \subset V$, $E_H \subset E$.

2.2. Función peso ($\xi(d)$)

Los patrones en la red se detectan a partir de los valores de peso de los arcos del grafo. Este valor es modificado según la distancia que se observa entre dos gramas de un texto. La función peso

es la que pondera dicha distancia. No se ha incluido la determinación de una expresión analítica para esta función, debido a que ésta se encuentra sujeta al problema que se desea solucionar.

La función peso también juega un papel preponderante en el número de arcos y nodos de la red, debido a que la ponderación de la distancia es utilizada por el mecanismo de selección para la eliminación de arcos *[sección 3]*.

Debido a que el propósito de esta sección es presentar el objetivo de la función peso, no se define una expresión analítica de la misma. Sin embargo, se expresan las siguientes restricciones

$$\xi(d) > 0 \forall d \in \Re^+ \qquad [2]$$
$$\exists \xi(d) \forall d \in N^+$$

Es ministerio de quién implemente el modelo que se presenta en este capítulo el definir la expresión analítica de $\xi(d)$ acorde a los objetivos que desee cumplir.

2.3. Actualización del grafo

Conforme se recorre un texto para la detección de patrones, el grafo que modeliza dicho documento debe cambiar. La actualización del grafo se puede dar por dos causas:

a) La aparición de un nuevo grama en el texto.
b) La aparición de un grama preexistente.

2.3.1 Aparición de un nuevo grama en el texto

Sea $H^n = (V_H^n, E_H^n)$ un estado n del grafo y $v \in V$ un nuevo vértice, la actualización del grafo se lleva adelante mediante la siguiente función:

$$\Lambda(v) = \bigcup_{v_i \in V_H^n} (v, v_i, t(x_i)) \qquad [3]$$

Siendo t la función definida como $t : \Re^k \rightarrow \Re$

$$t(x_i) = \sum_{j=1}^{k} \xi(x_{ij}) \qquad [4]$$

x_i: vector de distancias entre el grama v y v_i

Debido a que este modelo teórico esta diseñado para una aplicación práctica, se debe hacer mención al valor de k. Dicho número, representa la cantidad de instancias v_i anteriores a v. No obstante, al implementar en un software el algoritmo de actualización, se encuentra inmanejable, en cuestiones de performance y memoria, el almacenamiento de dichas instancias. Es debido a esto que, se introduce al modelo el concepto de *ventana de visualización de gramas* (σ). Dicha *ventana* determina el número de gramas que se deben considerar previos a una actualización del grafo.

Finalmente se define la instancia posterior a la actualización como,

$$H^{n+1} = (V_H^{n+1}, E_H^{n+1}) \qquad [5]$$

Donde,

$$V_H^{n+1} = V_H^n \cup v \qquad \text{y} \qquad E_H^{n+1} = E_H^n \cup \Lambda(v) \qquad [6]$$

2.3.2. Aparición de un grama preexistente (ANE)

Cada vez que aparezca un nodo ya presente en el grafo, se llevará adelante la operación de actualización de pesos que formalmente se define como:

$$\text{Sea } v \in V_H^n, \forall e \in \bigcup_{vi}^{+}, w(e) = w(e) + t(x_{vi}) \quad [1] \; [7]$$

Donde,

x_{vi}: vector de distancias entre dos nodos.

A igual que en la sección anterior, es necesario remarcar que la definición presentada sólo es válida en un marco teórico, debido a que dicha actualización resulta viable computacionalmente sólo si se tiene en cuenta la *ventana de visualización de gramas* que limita el tiempo y memoria necesarios para implementar el algoritmo.

3. Método de selección

3.1. Objetivo

El método de selección que se presenta en este apartado tiene como finalidad reducir el espacio de búsqueda de los patrones de gramas. Este algoritmo permite eliminar aquellos arcos que posean un peso bajo con respecto a la sumatoria total de los pesos del conjunto de arcos salientes del nodo. De esta forma, se eliminan caminos innecesarios y se obtiene un grafo de menor tamaño que facilita el proceso de búsqueda de patrones. Formalmente, el objetivo del algoritmo se puede definir de la siguiente manera:

Definición 4. *Sea* $H = (V, E)$ *un grafo ponderado con función de peso* $w : E \to \Re$ *, encontrar un subgrafo* $H' = (V', E') / \forall v \in V', w(v) > \alpha$ *, donde* α *es una constante del problema.*

3.2. Algoritmo y Estructura de Datos

Se define, en principio, el conjunto de arcos $A = \{a_1, a_2, ..., a_n\}$ y supongamos $A = \bigcup_v^+$. Cada arco de A será evaluado con respecto al conjunto según la función probabilística:

$$\overline{w}^k{}_i = \frac{w^k(a_i)}{\sum_{a_j \in A} w^k(a_j)} \qquad [8]$$

Donde, $w^k(a_j)$ es el peso del arco a_j en una instancia *k* del nodo *v*.

Luego, se especifica $[\overline{w}_{\min}; \overline{w}_{\max}]$ como el rango de valores posibles para $\overline{w}_i$. Es decir, se considera que todo arcos cuyo $\overline{w}$ sea menor a $\overline{w}_{\min}$ no debe ser tomado en cuenta y por ende puede ser eliminado. Mientras que todo arco que tenga un $\overline{w}$ mayor a $\overline{w}_{\max}$ puede ser interpretado como una relación fuerte entre gramas y en consecuencia un patrón.

Cada arco de A es dispuesto en un vector *V* de q posiciones según su $\overline{w}_j$ de la siguiente forma:

[1] Se define como $\bigcup_v^+$ al conjunto de arcos salientes de v.

$$a_j \in V(i) \leftrightarrow \left(\overline{w}_{\min} + (i-1)\left(\frac{\overline{w}_{\max} - \overline{w}_{\min}}{q} \right) \right) < \overline{w}^k{}_j < \left(\overline{w}_{\min} + i.\left(\frac{\overline{w}_{\max} - \overline{w}_{\min}}{q} \right) \right); 1 \le i < q$$

ó

$$a_j \in V(q) \leftrightarrow \left(\overline{w}_{\min} + (q-1)\left(\frac{\overline{w}_{\max} - \overline{w}_{\min}}{q} \right) \right) < \overline{w}^k{}_j \qquad [9]$$

Cada vez que se genera una instancia del nodo v (aparece el grama en el texto) se actualizan todos los pesos de los arcos [*ver sección 3.3.a y 3.3.b*], y en consecuencia los $\overline{w}^k{}_j$. Esto puede generar que, los arcos activados incrementen su posición en el vector. Debido a la presencia de la ventana de visualización de gramas [*ver sección 2.3.a*] cada vez que sucede una instancia del nodo v pueden existir algunos arcos que no se activen y por ende su posición en el vector tiende a disminuir. Es decir, cada vez que aparezca un grama en el texto se debe evaluar a cada uno de los arcos del nodo para verificar su correcta posición en el vector. Debido a que esto es costoso en tiempo, se decidió implementar la siguiente solución: Sucedido un tiempo ε [*ver sección 3.3*] el nivel inferior se limpiará de arcos (se eliminan) y éste pasará a ser el nivel superior del vector.

Por otro lado, cada posición del vector tiene lo que se denomina una zona segura, todo elemento perteneciente a esa zona no descenderá de nivel cuando pase el tiempo ε. La pertenencia de un elemento del vector a la zona de seguridad se explicará en la sección 3.4.

Finalmente, se debe mencionar que el valor de q, como así también, el rango $[\overline{w}_{\min}; \overline{w}_{\max}]$ son parámetros del algoritmo, en consecuencia deben ser definidos por quién lo implemente.

3.3. Definición del tiempo ε

Cada vez que se activa un nodo del grafo (instancia *k*) existe un conjunto de gramas en la ventana de visualización. Cuando se actualizan los pesos de los arcos salientes del nodo sólo se tienen en cuenta dichos gramas. Las distancias existentes entre el nodo y cada uno de ellos determinan un conjunto de incrementos de pesos que denominamos P^k_{vv}.

$$P^k_{vv} = \{p^k_1, p^k_2, \ldots, p^k_\sigma\}$$
$$P^{k+}_{vv} = \sum_{i=1}^{\sigma} p^k_i \qquad [10]$$

Por otro lado, cada nodo *v* del grafo tiene asociado un conjunto de arcos $A = \bigcup_v^+ = \{a_1, a_2, \ldots, a_n\}$. Cada elemento de este conjunto será comparado mediante la expresión *8.* Si tomamos en cuenta dicha expresión y si no hay aumento del peso del arco a_i se observa que:

$$\overline{w}^{k+1}_i = \frac{w^k(a_i)}{\sum_{a_j \in A} w^k(a_j) + P^{(k+1)+}_{vv}} \qquad [11]$$

Si suponemos $P^k_{vv} = P_{vv}$ constante, luego de η instancias se tiene:

$$\overline{w}^{k+\eta}_i = \frac{w^k(a_i)}{\sum_{a_j \in A} w^k(a_j) + \eta * P_{vv}} \qquad [12]$$

Finalmente, otro elemento a tener en cuenta es la posición del arco en el vector V. Como se mencionó en la sección *3.2*, el arco a_i se encuentra en el vector *V* según la forma *9*, para el estudio de ε se debe formalizar los limites de cada posición del vector. Dicho valor es definido de la siguiente manera:

$$\Delta = \Delta(\mathrm{m}) = \left(\frac{\overline{w}_{\max} - \overline{w}_{\min}}{q} \right) \qquad [14]$$

Ya explicitados todos los elementos necesarios, se estudia cuántas instancias del nodo deben ocurrir para que un arco que se encuentre en un nivel determinado del vector disminuya su posición.

Sea τ el valor que debe disminuir el arco a_i para bajar un nivel en el vector y si se supone P_{vv}^k constante se tiene:

$$\tau = \overline{w}_i^k - \overline{w}_i^{k+\eta} \qquad [15]$$

Luego,

$$\eta = \frac{\left(\sum_{a_j \in A} w^k(a_j)\right)\tau}{P_{vv}\left(\overline{w}_i^k - \tau\right)} \qquad [16]$$

Si el arco a_i se encuentra en la posición *m* del vector, entonces τ puede escribirse de la siguiente manera:

$$\varepsilon = \overline{w}_i^k - \theta(\mathrm{m}-1) \qquad [17]$$

Luego,

$$\eta = \frac{\left(\sum_{a_j \in A} w^k(a_j)\right)\ (\%\ w^k - \theta(\mathrm{m}-1))}{P_{vv}\ \theta(\mathrm{m}-1)} \qquad [18]$$

Como norma, se decide esperar lo máximo posible antes de disminuir a un arco de posición. Por los tanto, se elige

$$\theta(m-1) = \overline{w}_{\min}$$
$$\overline{w}_i^k - \theta(m-1) = \Delta$$

Por último se define α_i como la relación entre la sumatoria de los pesos de los arcos eliminados y el total de pesos asociados al nodo.

$$\alpha_i = \frac{\sum_{\Gamma k} w^k(a_i)}{W_j^i} \qquad [19]$$

Donde Γk es el conjunto de pesos de los arcos eliminados en la instancia k.

Así, para la primera iteración:

$$\sum_{a_j \in A} w^1(a_j) = P_{vv}, \text{ y } \alpha_1 = 0 \text{ (no se elimina ningún arco)}$$

Luego,

$$\varepsilon_1 = \frac{\Delta}{w_{\min}}$$

Para la segunda iteración se tiene:

$$\sum_{a_j \in A} w^2(a_j) = \varepsilon_1 P_{vv} - \alpha_1 \varepsilon_1 P_{vv}$$

(Todo lo acumulado en la primera iteración menos lo eliminado)

Por lo tanto,

$$\varepsilon_2 = \varepsilon_1^2 (1 - \alpha_1)$$

Siguiendo el mismo razonamiento, por inducción se tiene:

$$\varepsilon_1 = \frac{\Delta}{w_{min}}$$

$$\varepsilon_2 = \varepsilon_1^{\,2}\,(1-\alpha_1)$$

$$\varepsilon_n = \varepsilon_{n-1}\,(1+(1-\alpha_{n-1})\,\varepsilon_1)$$

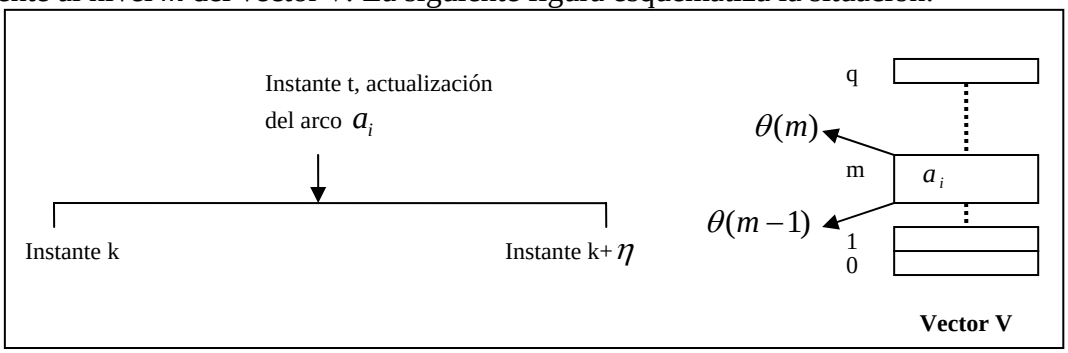
[20]

3.4. Zona de seguridad

Cada posición del vector *V* posee lo que se denomina una zona de seguridad. Todo arco que pertenezca a dicha zona se llama *"arco seguro"*, sin embargo aquel arco que no se encuentre en una zona de seguridad se designa como *"no seguro"*. El manejo de las zonas de seguridad se rige mediante las siguientes reglas:

1. Todo arco que se encuentre en una zona de seguridad no desciende de nivel cuando se eliminan los arcos de la última posición de *V*.
2. Cada vez que se produce una eliminación del nivel inferior todo arco perteneciente una zona de seguridad pierde la condición de *"arco seguro"*, es decir, sale de la zona segura.
3. Siempre que se produzca una eliminación del nivel inferior, toda zona segura se completará con los arcos *"no seguros"* del nivel inmediatamente superior.

En esta sección se define cuánto debe ser el valor del incremento del peso de un arco para entrar en una zona de seguridad. Para ello se sigue el siguiente razonamiento, supongamos que en un instante *k* se definen las η iteraciones que se deben esperar para realizar una nueva eliminación de arcos. Dicho valor se calcula para aquellos arcos que no sean activados en todo el período, es decir, no incrementen su valor peso. Sin embargo, no todos los arcos cumplen con ésta condición, es por ello que suponemos un instante t menor a η donde se produce la activación de un nodo a_i perteneciente al nivel *m* del vector *V*. La siguiente figura esquematiza la situación:



Para que un elemento del vector entre en una zona de seguridad su relación de peso con el total en el instante k+η ($\overline{w}_i^{k+\eta}$) debe ser mayor al límite inferior del nivel donde se encuentra, es decir:

$$\theta(m-1) \le \frac{w^{t-1} + I^t}{W_j^{t-1} + P_{ww} + (\eta - t)\,P_{ww}} \qquad \textit{[21]}$$

Siendo

$$W_j^{\,t-1} = \sum_{a_i \in A} w^{\,t-1}(a_i)$$

$w^{t-1} = w^{t-1}(a_i), a_i \in A$

I^t, el incremento del arco en el instante t

Definimos entonces el valor del incremento para entrar en la zona de seguridad:

$$\theta(m-1) \leq \frac{w^{t-1} + I^t}{W_j^k + t\,P_{vv} + (\eta - t)\,P_{vv}}$$

$$\theta(m-1) \leq \frac{w^{t-1} + I^t}{W_j^k + \eta\,P_{vv}}$$

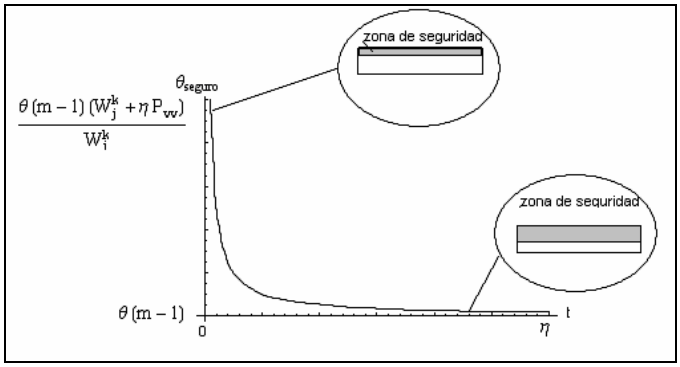
$$\theta(m-1)\,(W_j^k + \eta\,P_{vv}) - w^{t-1} \leq I^t \qquad [22]$$

Es decir, todo incremento en el peso de un arco que cumpla con la condición *22* provocará un pasaje del mismo a la zona de seguridad del nivel. Finalmente encontramos una expresión para la zona de seguridad de un nivel (θ_{seguro}):

$$\frac{w^{t-1} + I^t}{W_j^t} \geq \theta_{seguro} \qquad [23]$$

$$\frac{\theta(m-1)\,(W_j^k + \eta\,P_{vv})}{(W_j^k + t\,P_{vv})} \geq \theta_{seguro}$$

Se puede observar que el tamaño de la zona de seguridad es directamente proporcional al instante t, es decir, acorde suceden instancias del nodo, es más sencillo para un arco entrar a la zona de seguridad.



Se puede ver en la grafica anterior que cada posición del vector posee una zona de seguridad distinta, sin embargo, la variación del tamaño de la misma posee la misma curva independientemente del nivel de *V*.

4. Modelo de Agente

Desde el punto de vista global, el grafo puede entenderse como un conjunto de agentes simples interconectados entre sí. Con el término simples nos referimos a que cada nodo no lleva a cabo un grupo complejo de acciones sino que actúa a partir de primitivas sencillas que permiten, a escala global, desarrollar una actividad compleja. El accionar de cada vértice es independiente de resto por lo que, los nodos, no comparten información sobre el medio que los circunda. La comunicación

interagentes se lleva a cabo sólo una vez terminado el procesamiento del texto. En éste caso, los nodos se comunican entre sí el conjunto de arcos "fuertes" que tienen en posesión.

El funcionamiento interno de los agentes se encuentra explicitado en los párrafos siguientes, donde se especifican las creencias, metas y planes de los mismos.

Creencias: Cada nodo del grafo tiene una visión distinta de su entorno. Es decir, un conjunto diferente de creencias. Es por ello que las variables que aquí se mencionan se encuentran asociadas a un nodo en particular, o lo que es lo mismo, no tienen carácter global en el grafo.

En principio, las variables asociadas al conocimiento del agente pueden diferenciarse *en variables de definición de frecuencias* y *variables de peso*. Las primeras están relacionadas al manejo del tiempo mientras que las segundas se distinguen por su utilidad en la comparación de arcos.

Existen, básicamente, dos variables de frecuencia. Por un lado, la variable λ_v que define el número de instancias del nodo *v* y ε_i que especifica el número de de instancias que se debe aguardar para poder realizar una eliminación de arcos. El cálculo de dicha variable se desarrolló anteriormente en la sección *3.3*.

En lo concerniente a los arcos, cada vértice del grafo tiene conocimiento del conjunto de arcos salientes de él. A partir de este conjunto se pueden definir el segundo tipo de variables, las variables de peso.

Se define como W_v^k a la sumatoria pesos de las instancias de $\bigcup_v^+$:

$$W_v^k = \sum_{a_i \in \bigcup_v^+} w^k(a_i) \qquad [24]$$

Existe, por otro lado, otra variable de peso introducida en la sección *3.2*, particularmente mediante la ecuación *8*. La variable $\overline{w}_v^k$ depende de la instancia del nodo y es la utilizada para la comparación de los arcos.

Es necesario destacar que todo elemento perteneciente a $\bigcup_v^+$ se encuentra en dispuesto en el vector *V* según lo especificado en la sección *3.2*.

Finalmente, cada nodo desconoce absolutamente el estado de su nodo vecino. Sólo finalizado el procesamiento los nodos se comunicarán unos con otros proveyendo a su vecino con los arcos de mayor peso.

Deseos: A diferencia de las creencias, todos los nodos del grafo poseen la misma meta. El objetivo principalmente radica en la maximización del promedio de los pesos que salen de él.

Formalmente,

$$\max\left(\frac{W_v^k}{card\left(\bigcup_v^+\right)}\right)$$

Se puede observar que una buena heurística para el cumplimiento de este objetivo radica en la selección de las mejores instancias de $\bigcup_v^+$. De esta manera se busca eliminar los caminos irrelevantes del grafo y poder, por lo tanto, detectar los patrones del texto de forma más rápida.

Plan: El plan de cada se encuentra regido principalmente por sus variables de frecuencia. El procedimiento básico se resume en la eliminación de arcos irrelevantes según lo determinado en el apartado del proceso de selección *[sección 3]*.

Cada instancia del nodo es determinada por la aparición en el texto del grama que representa. Siempre que esto ocurra se incrementara en uno a la variable λ_v y mientras que ésta sea distinta de ε_i solo se procederá a la actualización de los pesos de los arcos.

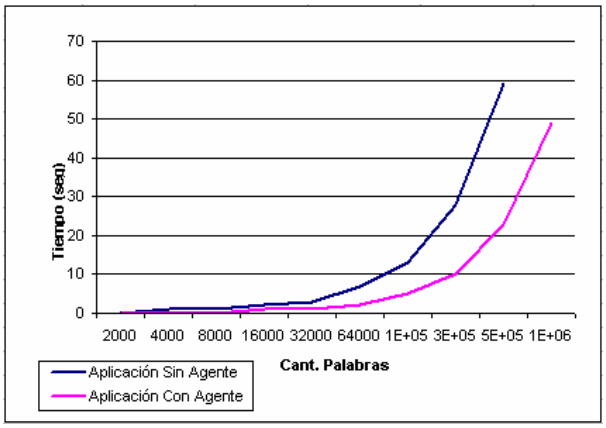
En el caso de que λ_v sea igual a ε_i se procede a eliminar a todos los elementos de la última posición del vector *V*.

Una vez finalizado el documento a procesar, cada nodo del grafo comunicará sus relaciones fuertes al resto para la determinación de los patrones existentes.

5. Caso de Estudio

Para comprobar la efectividad de la aplicación de agentes al grafo se implementó un software basado en lo desarrollado en los capítulos anteriores. Se llevaron adelante dos corridas del mismo, una con la aplicación de agentes y la otra sin ella. En dichas corridas se midieron los tiempos de ejecución y se obtuvieron los resultados graficado en el cuadro. Allí se puede observar que la implementación de agentes al modelo de grafos para la detección de patrones permite una mejora promedio de un 63% en los tiempos.

Cant. Palabras	***Sin Agente***	***Agente***	***Mejora***
2000	0	0	0
4000	1	0	100%
8000	1	0	100%
16000	2	1	50%
32000	3	1	67%
64000	7	2	71%
128000	13	5	62%
256000	28	10	64%
512000	59	23	61%
1024000	-----	49	



Nota: Las pruebas fueron realizadas en un ordenador Pentium 4 2.4 GHz con 512 Mb de memoria. Los textos para realizar las pruebas fueron obtenidos de www.gutenberg.org . Consisten de novelas y cuentos cortos en formato txt idioma ingles.

6. Conclusión y Trabajos Futuros

El método para detección de patrones en textos propuesto permite al usuario definir el tipo de patrón que desea buscar mediante la determinación analítica de la función peso (*sección 2.2*). Esto hace al algoritmo altamente flexible debido a que, mediante la definición de dicha función, se pueden detectar patrones en distintos textos sin importar cuestiones como por ejemplo la longitud o el idioma. Asimismo, la inclusión de la teoría de agentes y el método de selección presentado permite reducir el volumen del grafo disminuyendo los tiempos de procesamiento. Esta cualidad permite que el modelo sea implementado en programas donde se requiera un tiempo de procesamiento corto, como por ejemplo, un explorador de Internet para la clasificación de las páginas Web que muestre.

Entre los trabajos futuros planteados, se encuentran los siguientes:

- Determinación de expresiones analíticas para la función peso dependientes del idioma y el objeto de búsqueda.
- Adaptación del modelo presentado para textos de gran volumen.

- Determinación de métodos de aprendizaje para la eliminación de nodos del grafo poco relevantes.

7. Referencias

[1] Don Adjeroh, Amar Mukherjee, Tim Bell, Matt Powell, Nan Zhang, Proceedings of the Data Compression Conference (DCC '02) , Pattern Matching in BWT-transformed Text, 445, 2002, ISSN:1068-0314

[2] Cristobal Baray y Kyle Wagner Where Do Intelligent Agents Come From?. Crossroad, ACM. 1999.

[3] Christopher D. Manning, Hinrich Schütze, Foundations of Statistical Natural Language Processing, MIT Press: 1999. ISBN 0262133601.

[4] Ciravegna et al., Ed. (2001), Proc. of the 17Th International Joint Conference on Artificial Intelligence (IJCAI-2001), Workshop of Adaptive Text Mining, Seattle, WA, 2001.

[5] Feldman, Ed. (1999), Proc. of The 16th International Joint Conference on Artificial Intelligence (IJCAI-1999), Workshop on Text Mining: Foundations, Techniques and Applications, Stockholm, Sweden, 1999.

[6] Stan Franklin y Art Graesser, “Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents”, Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996.

[7] Wei Jin, Rohini Srihari, Symposium on Applied Computing archive Proceedings of the 2007 ACM symposium on Applied computing table of contents, Graph-based text representation and knowledge discovery, 807 - 811 ,2007, ISBN: 1-59593-480-4

[8] Junji T., Hidekazu N., Megumi I., Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters WWW Alt. '04,Mayo 2004, ACM Press

[9] Junji T., Hidekazu N., Megumi I., Proceedings of the thirteenth ACM international conference on Information and knowledge management CIKM '04, Noviembre 2004, ACM Press

[10] David Kinny, Michael George, Anand Rao, Proceedings of the 7th European workshop on Modelling autonomous agents in a multi-agent world, A Methodology and Modelling Technique for Systems of BDI Agents, 56-71, 1996, ISBN:3-540-60852-4

[11] Tsuyoshi Kitani, Yoshio Eriguchi, Masami Hara, Journal of Artificial Intelligence Research 2, Pattern Matching and Discourse Processing in Information Extraction from Japanese Text,89-110,1994

[12] M. Marko, M. A. Porter, A. Probst, C. Gershenson, A. Das, 2002, Transforming the World Wide Web into a Complexity-Based Semantic Network, http://arxiv.org/html/cs/0205080 (25/05/2007)

[13] Mladenic, Ed. (2000), Proc. of the Sixth International Conference on Knowledge Discovery and Data Mining, Workshop on Text Mining, Boston, MA, 2000.

[14] -. "Subgraph error correcting isomorphisms for syntactic pattern recognition," IEEE Trans. Sysr., Man, Cybern., vol. SMC-13, no. I , pp. 48-62, Jan. 1983.

[15] W. H. Tsai y K. S. Fu. "Error-correcting isomorphisms of attributed relational graphs for pattern analysis," IEEE Trans. Syst., Man, Cyhern.. vol. SMC-9, pp. 757-768, Dec. 1979.

[16] A. K. C. Wong y S . W. Lu and M. Rioux, "Recognition and shape synthesis of 3-D objects based on attributed hypergraphs," IEEE Trum. Purrern Anal. Machine Intell., vol. PAMI-1 1, no. 3, pp. 279- 289. Sept. 1989.

[17] Dong-Qing Zhang, Shih-Fu Chang. Stochastic Attributed Relational Graph Matching for Image Near-Duplicate Detection, DVMM Technical Report #206-2004-6, Department of Electrical Engineering, Columbia University