TÉCNICA DE MINERÍA DE DATOS PARA LA ADAPTACIÓN DE SITIOS WEB

E. Lazcorreta[a], F. Botella[a], A. Fernández-Caballero[b], P. González[b]

[a] Centro de Investigación Operativa, Universidad Miguel Hernández de Elche, Avda.Universidad s/n, 03202 Elche, España
e-correo: {enrique, federico}@umh.es

[b] Laboratorio de Interacción con el Usuario e Ingeniería del Software (LoUISE), Universidad de Castilla-La Mancha, Campus Universitario, 02071 Albacete, España
e-correo: {caballer, pgonzalez}@info-ab.uclm.es

Resumen

Las técnicas de Minería de Datos (DM) surgen con la necesidad de obtener información de grandes repositorios de datos. Entre las técnicas más populares están las que buscan reglas de asociación entre los distintos ítems de una "cesta de la compra", que en nuestra investigación son las páginas solicitadas en una visita de un usuario a un sitio web (sesión). Aplicando el principio de coherencia, que postula que el contenido de las páginas visitadas en una misma sesión guarda cierta coherencia, podemos usar las reglas de asociación para hacer predicciones sobre las páginas que desea visitar el usuario, predicciones que el servidor puede presentar como *sugerencias de navegación al usuario*. Si además consideramos que cada usuario está interesado en un número "pequeño" de contenidos del sitio web podemos reducir considerablemente las dimensiones del repositorio para aplicar dichas técnicas en tiempo real y hacer sugerencias de navegación personalizadas al usuario anónimo.

Metodología

Son muchos los sistemas de recomendación existentes en la literatura científica: Analog, WebWatcher, WebPersonalize, SpeedTracer, PageGather, Suggest, … La mayoría de ellos se basan en dos componentes: uno que opera off-line aplicando técnicas de MD sobre grandes repositorios de datos, y otro que opera on-line utilizando los resultados obtenidos en la fase off-line para mostrar las sugerencias al usuario. La justificación de esta separación está en el excesivo consumo de tiempo y recursos en la aplicación de técnicas de MD, dado que son muchos y muy heterogéneos los datos a procesar, y su actualización se realiza generalmente en las horas en que el servidor tiene menor trabajo.

Estas características pueden restar actualidad y granularidad a las sugerencias obtenidas. La inclusión de sugerencias a páginas nuevas del sitio se pospone hasta que su popularidad alcanza el soporte mínimo o bien hasta que se actualicen los resultados en la fase off-line del servicio. Se proporcionan a todos los usuarios las mismas sugerencias, la inclusión de los usuarios en clusters permite operar con un número de datos reducido pero ignora las diferentes necesidades de usuarios que solicitan la misma página. Con nuestro trabajo no pretendemos sustituir sino complementar las sugerencias emitidas por un sistema de recomendación, por lo que partimos de sistemas que ya tienen un repositorio con las sesiones de usuarios, un nuevo repositorio de sesiones "activas" es también explotado y sólo se ha de actualizar el repositorio histórico periódicamente y sin uso de técnicas de MD.

Hemos elegido el algoritmo Apriori (Agrawall et al., 1994, véase Algoritmo 1) por su sencillez. En su implementación hemos reunido en una sola función la unión y poda para dotarlo de mayor eficiencia y usamos una relajación de la poda que reduce notablemente el tiempo de búsqueda de candidatos respecto al algoritmo original (véanse Algoritmo 2, 3 y 4).

La metodología usada es la siguiente:

1. El usuario solicita la página X.
2. El servidor identifica al usuario, toma del repositorio las sesiones pertenecientes al usuario y las guarda en memoria principal reordenando sus ítems según su código numérico y sin repetición, habiendo previamente asignado el código 0 a la página X.
3. Se aplica Apriori con relajación de poda a los conjuntos de accesos en memoria principal. La reducción de datos a procesar, su homogeneidad y la lectura desde memoria principal posibilitan la aplicación de este algoritmo en tiempo real.
4. El servidor devuelve al usuario la página X con las sugerencias más frecuentes de las primeras ramas del árbol L obtenido en el paso anterior y filtradas convenientemente con información de la estructura del sitio web.

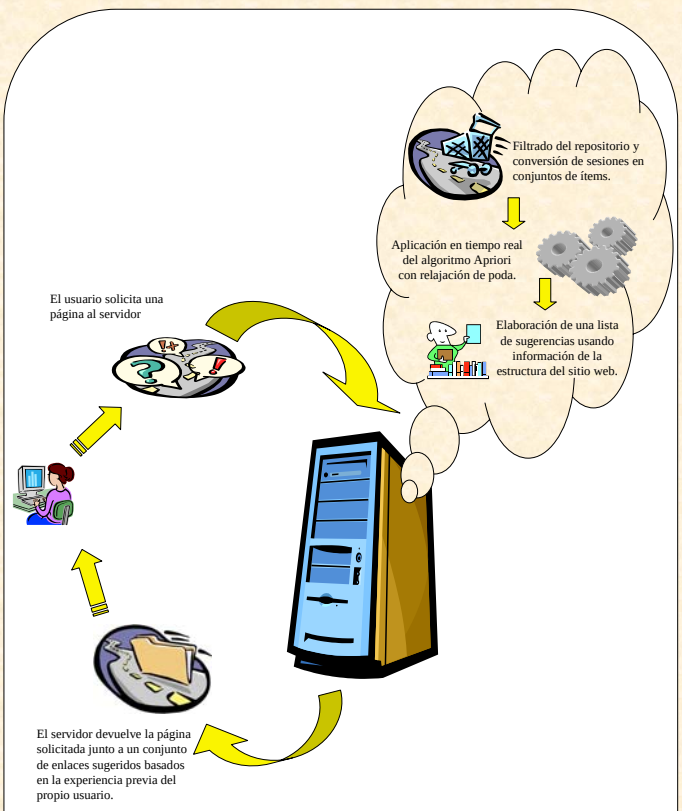

Figura 1: Personalización de un sitio web basado en la aplicación on-line de técnicas de DM.

```
L1 = {large 1-itemsets};
for (k = 2; Lk-1 ≠ ∅; k++) do begin
   Ck = apriori-gen(Lk-1);
   forall transactions t ∈ D do begin
      Ct = subset(Ck, t);
      forall candidates c ∈ Ct do
         c.count++;
   end
   Lk = {c ∈ Ck / c.count ≥ minsup}
end
Answer = ∪ Lk
```

Algoritmo 1: Algoritmo apriori

```
insert into Ck
select p.item1, …, p.itemk-1, q.itemk-1
from Lk-1 p, Lk-1 q
where p.item1 = q.item1, …, p.itemk-2 = q.itemk-2, p.itemk-1 < q.itemk-1;
```

Algoritmo 2: Función apriori-gen, unión

El algoritmo 4 sustituye en nuestra implementación a los algoritmos 2 y 3 originales.

La unión y poda reunidas en una única función ahorran muchas operaciones de inserción y eliminación de elementos en el vector dinámico C_k.

La relajación de la poda ahorra muchas operaciones de búsqueda.

```
forall itemsets do
   forall (k-1)-subsets s of c do
      if (s ∉ Lk-1) then
         delete c from Ck;
```

Algoritmo 3: Función apriori-gen, poda

```
insert into Ck
select c = {p.item1, …, p.itemk-1, q.itemk-1}
from Lk-1 p, Lk-1 q
where (p.item1 = q.item1, …, p.itemk-2 = q.itemk-2, p.itemk-1 < q.itemk-1)
      and
      (p.itemk-1, q.itemk-1) ∈ L2;
```

Algoritmo 4: Función apriori-gen con unión y poda relajada

Experimentación

Para probar la eficiencia del algoritmo Apriori modificado se prepocesaron las solicitudes hechas a un servidor web durante los 22 primeros días del mes de julio de 2004. En la lectura del fichero log se observaron 2.977.380 solicitudes de recursos, 445.560 de las cuales se correspondían con 2.158 páginas visitadas en el sitio web. Se definió *sesión de usuario* como la secuencia de solicitudes realizada desde una misma IP, de modo que entre una petición y la siguiente no transcurran más de 15 minutos y entre la primera y la última solicitud no transcurran más de 8 horas. Se obtuvieron 30.559 sesiones realizadas desde 11.391 direcciones IP, lo que constituyó el repositorio histórico de sesiones.

Tiempo (milisegundos)	Frecuencia	%
t < 15	9.079	79.70
15 ≤ t < 30	2.305	20.24
30 ≤ t < 45	4	0.04
45 ≤ t < 60	2	0.02
t ≥ 60	1	0.01

Tabla 1: Tiempos de ejecución del algoritmo

En la tabla 1 se muestran los tiempos empleados en la ejecución del algoritmo para los 11.391 usuarios de nuestro repositorio. En el 99.94% de los casos se requieren menos de 30 milisegundos para obtener información sobre las asociaciones más frecuentes entre las páginas visitadas por el usuario, información que utilizará el servidor para generar on-line los enlaces sugeridos. Los 3 procesos más lentos se corresponden con sesiones realizadas desde alguna red con dirección IP única, lo que proporciona largas y heterogéneas sesiones. Esta situación es predecible en el filtrado del repositorio histórico por lo que se puede adoptar rápidamente una estrategia que reduzca el tiempo necesario para su análisis, también es factible guardar en el repositorio de direcciones IP's una variable booleana que indique si es factible el análisis en tiempo real de las sesiones procedentes de cada IP y, en el caso de no ser factible, ajustar los parámetros del algoritmo para lograr la velocidad de proceso adecuada.

Conclusiones y trabajo futuro

Con este trabajo se muestra que las técnicas de MD se pueden aplicar en tiempo real maximizando la eficiencia computacional de los algoritmos usados y filtrando los repositorios de datos en subconjuntos que aporten conocimiento sobre una parte de su contenido.

La incorporación de información de la estructura de enlaces del sitio web, además de proporcionar medidas cuantitativas de la eficiencia del sistema del recomendación puede mejorar la usabilidad del sitio si filtramos el conjunto inicial de sugerencias obtenido.

La fusión de nuestros resultados con los obtenidos con los métodos clásicos de predicción puede aportar mayor calidad a las sugerencias emitidas por el servicio. Esta es una de las líneas en que seguirá nuestra investigación.

Referencias

Agrawal, R. & Srikant, R., 1994. Fast Algorithms for Mining Association Rules. In 20th VLDB Conference.

Lee, J.H. & Shiu, W.K., 2001. An adaptive website system to improve efficiency with web mining techniques. Advanced Engineering Informatics, v. 18, n. 3, pp 129-142, julio.

Perkowitz, M. & Etzioni, O., 1999. Adaptive Web Sites: Conceptual Cluster Mining. In Proc. of IJCAI'99, pp 264-269.

Silvestri, F., Baraglia, R. & Palmerini, P., 2004. On-line Generation of Suggestions for Web Users. In Proc. of ICIT: Coding and Computing.