UNIVERSIDAD DE SANTIAGO DE CHILE

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

Estrategias de planificación para motores de búsqueda verticales

Danilo Fernando Bustos Pérez

Profesor Guía: Dra. Carolina Bonacic Castro
Profesor Co-guía: Dr. Mauricio Marín Caihuán

Trabajo de Titulacíon presentado en conformidad a los requisitos para obtener el Título de Ingeniero Civil Informático

SANTIAGO DE CHILE
2013

AGRADECIMIENTOS

Dedicado a ... .

RESUMEN

Resumen en Castellano

Palabras Claves: keyword1, keyword2 .

ABSTRACT

Resumen en Inglés

Keywords: keyword1, keyword2 .

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

1.1 ANTECEDENTES Y MOTIVACIÓN

La Web 2.0 o "Web Social" se caracteriza por ser una Web centrada en el usuario, la participación, la interactividad y la colaboración (**?**). *Youtube*[1], *Facebook*[2], *Flickr*[3] ahora son compañeros cotidianos y habituales en el mundo tecnológico que nos toca vivir. Antes de este cambio cultural, la Web 1.0 estaba centrada principalmente en el contenido, el cual era actualizado unilateralmente por usuarios expertos; era principalmente comunicación de una sola vía. En cambio, la Web 2.0 elimina las barreras entre productores y consumidores de información, creando canales de doble vía, empoderando al usuario final como creador activo de información. Este cambio produjo una verdadera revolución social del uso de las tecnologías en la red. Sea dicho de paso, esta revolución se vio facilitada por la incorporación de la tecnología de la "sindicación" la cual permite retroalimentación en línea sobre la información de productores y consumidores. En definitiva la Web 2.0 es una forma distinta de usar la Web como canal de comunicación (**?**) que permite a los usuarios finales, además de publicar, interactuar con los distintos contenidos de la Web (textual, multimedia, etc) de diversas formas.

El masivo uso de las aplicaciones de la Web 2.0 ha poblado Internet de una enorme cantidad de información, por este motivo, y para ayudar a los usuarios en la búsqueda de información, se desarrollan dos tipos de sistemas: los motores de búsqueda y sistemas de

[1] http://www.youtube.com/
[2] https://www.facebook.com/
[3] http://www.flickr.com/

recomendación (en adelante SR). Estos últimos entregan información filtrada de acuerdo al comportamiento de otros usuarios (meta-información), de las preferencias propias, ajenas y de los atributos de la información buscada. El comportamiento de los usuarios puede ser modelado como un conjunto de interacciones de valor colaborativo que permiten obtener información sobre la colaboración entre usuarios explícita o implícita. Estas interacciones pueden ser de distintos tipos (**?**), e incluyen actividades como etiquetar (*tagging*), comentar (commenting), valorizar (*rating*) entre otros.

El área de investigación de los SR nace a mediados de los 90', cuando *GroupLens*[4] publica el primer artículo, que se basa en el filtrado colaborativo para generar recomendaciones de noticias (**?**). En el estado del arte (**?**) se plantea que esta área se ha mantenido en expansión debido a que provee un gran número de aplicaciones prácticas que ayudan a los usuarios a encontrar información de utilidad.

En la actualidad existen varios SR, enfocados a distintos dominios de aplicación. Algunos ejemplos son sitios como *MovieLens*[5] y *Netflix*[6] para recomendación de peliculas; *Google News*[7] para recomendación de noticias; *LastFm*[8] para recomendación de música, y *Amazon*[9] para recomendación de *e-commerce*, entre otras. En este mismo ámbito y con el objetivo de mejorar su SR, *Netflix* creó un concurso llamado *"Netflix Prize"*[10] que premiaba con un millón de dólares a quienes lograran mejorar su algoritmo de recomendación basado en la métrica **RMSE** (error cuadrático medio). Este concurso motivó la investigación en el área promoviendo nuevas formas de abordar el problema de recomendación basada en *rating*.

Los SR tradicionales se basan en dos dimensiones: el usuario y el ítem para calcular una función de utilidad (**?**). Básicamente se construyen perfiles de usuario e ítem, como también se

[4]http://www.grouplens.org/
[5]http://movielens.umn.edu/login
[6]www.netflix.com
[7]news.google.com
[8]www.last.fm
[9]www.amazon.com
[10]http://www.netflixprize.com

modelan las interacciones de los usuarios hacia los ítems. Sin embargo, en el desarrollo de la investigación en el área se están modelando nuevas dimensiones con el objetivo de mejorar el proceso de recomendación (**?**). Ejemplos de estas dimensiones son el prestigio y afinidad entre usuarios para medir la confiabilidad de la recomendación (**?**). La geolocalización (**?**) basándose en la ubicación física y/o virtual del usuario. Otro conjunto de dimensiones importantes se basan en el contexto de interacción del usuario mejorando así el proceso de recomendación (**?**). Luego estos sistemas se han sobre-especializado con el objetivo de entregar recomendaciones de mayor relevancia para el usuario final.

En las empresas se hace cada vez más necesario contar con SR para aumentar las probabilidades de selección de sus productos y/o servicios por parte de sus clientes. Cada empresa cuenta con registros de las interacciones de sus clientes en diversos repositorios de datos que suelen ser transversales a los sistemas de la empresa. Además, existe la información proveniente de las aplicaciones basadas en la Web 2.0, que proveen API's para consultar información de sus clientes, como también aplicaciones innovadoras de las empresas que se basan en la ideología de la Web 2.0.

En el área de investigación de los SR es difícil reproducir y extender los resultados obtenidos en las distintas publicaciones (**?**). Por lo tanto, se debe re-implementar los SR para un dominio particular donde éstos son usados. En este contexto, han aparecido trabajos que intentan aportar estándares en la representación, construcción y validación de SR que permitan la construcción sistemática de SR. Por ejemplo en *Lenskit* (**?**) se provee un *framework* extensible y mantenible para la construcción y evaluación de SR basados en filtrado colaborativo. Por otro lado en *Synergy* (**?**) se plantea un modelo de datos que representa las distintas interacciones que realizan los usuarios con los ítems. **?** plantea un *framework* que busca organizar y estandarizar el área de los sistemas de recomendación en dos aspectos: un modelo de representación de datos, y una arquitectura para proveer servicios de recomendación en una red social.

1.2 DESCRIPCIÓN DEL PROBLEMA

La representación y construcción de SR son fuente de los siguientes desafíos:

1. La Web 2.0 proporciona nuevas formas de interacción para los usuarios. Luego, existen dificultades para modelar múltiples interacciones para que estas sean usadas por los SR (**?**).

2. Los SR para obtener mejores resultados sobre un dominio específico se acoplan demasiado a la data (**?**). Esto provoca una sobre-especialización del SR, dificultando la posibilidad de reutilizar la solución en otro dominio de aplicación.

3. Un SR puede degradar sus resultados en el tiempo debido a cambios en la data referente a la interacción de los usuarios por lo que debería ser re-entrenado o cambiado por otro.

4. Las mejoras obtenidas en los algoritmos son difíciles de generalizar para todos los usuarios del sistema, por ejemplo algunos usuarios pueden usar un campo muy reducido de opciones de interacción y otros más.

Un ingeniero a cargo de realizar un SR espera contar con estándares y patrones para la construcción, por lo tanto no suele resolverlo de manera *ad-hoc* como lo hace un investigador del área de los SR. En efecto, un investigador tiende a especializarse para resolver problemas científicos y no se preocupa de proveer estándares para que su solución sea generalizable y/o reutilizable.

En definitiva, el problema de esta tesis está relacionada con la generalización, contextualización y transferencia de los SR. Esto puede expresarse en la siguiente pregunta que guía este trabajo de tesis: ¿Es posible contar con un *framework* que permita representar y construir SR de distinta complejidad en diversos dominios de aplicación?

1.3 OBJETIVOS Y SOLUCIÓN PROPUESTA

A continuación se presenta el objetivo general del proyecto que se concreta con el cumplimiento de los objetivos específicos. Para finalmente declarar los alcances del trabajo de tesis.

1.3.1 Objetivo General

Construir un *framework* que permita la representación y construcción de SR.

1.3.2 Objetivos Específicos

1. Sistematizar el proceso de construcción de SR desde la literatura actual del área.
2. Caracterizar las dimensiones emergentes para los SR.
3. Diseñar un modelo orientado a eventos de representación de datos para SR.
4. Diseñar un conjunto de operaciones para la obtención de datos desde el modelo de datos propuesto.
5. Diseño e implementación de una herramienta para construir SR.
6. Construir SR's basados en el *framework* propuesto.

7. Publicar los resultados de la investigación en una revista de la especialidad.

1.3.3 Alcances

El trabajo de tesis propuesto tiene los siguientes alcances:

- Dada la amplia variedad de SR que se encuentra en la literatura; la solución aunque es genérica se ejemplifica sólo a los SR de filtrado colaborativo *user-user* y de *tagging social* basados en técnicas de *clustering*.
- No es concluyente en la mejora de rendimientos, sino solamente en la eficacia del *framework* propuesto.
- Sólo se validará con el desarrollo de algoritmos que trabajen exclusivamente con *tags* y *ratings* como dos tipos de eventos colaborativos posibles.
- Este producto de software sólo contempla la construcción de un API y una implementación de esta para la construcción de los SR.
- Tanto el API y la implementación serán escritas en *Java Standard Edition 7*.
- Este producto de software no dispondrá extensiones *Windows* ni *Unix*, *Linux*. Sólo podrá ejecutar en la máquina virtual *Java* disponible en estos sistemas operativos.
- No se contempla establecer los procedimientos de evaluación de algoritmos dentro de la herramienta construida.

1.3.4 Solución propuesta

Este proyecto de investigación será del tipo investigación aplicada **I+D** (Investigación + Desarrollo):

1. **Investigación**: se propondrá un modelo que permita la representación de datos basándose en un framework proveniente del área de los sistemas colaborativos y un método de construcción de SR que se encuentren en el estado del arte del área.

2. **Desarrollo**: se construirá una API basada en el modelo y método propuesto.

Finalmente se construirán dos SR usando el API anterior para probar la eficacia del modelo propuesto.

1.3.5 Características de la solución

Las características de la solución propuesta son las siguientes:

- Una especificación o API[11] extensible basada en patrones de diseño que aseguren la flexibilidad requerida basada en un bajo acoplamiento, mantenibilidad y alta cohesión.

- Se construirá una herramienta basada en el API propuesta que permita la representación de datos basado en un método para construir SR.

- La representación de datos se realizará bajo un esquema orientado a eventos que permita modelar el comportamiento de los usuarios en la Web 2.0. Dada esta forma

[11] *Application Programming Interface*

de representación se construirá un conjunto de operaciones para la adquisición de datos del modelo como entrada para los SR.

- La construcción de algoritmos se realizará caracterizando el proceso de recomendación.
- Prueba empírica con la herramienta de software mediante la construcción de SR, que expresen la utilidad del modelo propuesto.

1.3.6 Propósito de la solución

Los propósitos de la solución son los siguientes:

- Proveer un modelo de representación de datos extensible. Para la concreción de lo anterior, se utilizará un *framework* conceptual del área de los sistemas colaborativos denominado *3-Ontology* propuesto por (**?**) que permite modelar las interacciones dentro de un ambiente colaborativo.
- Un método que permita la construcción de SR del estado del arte.
- Proveer al área de los SR una herramienta de uso personal o empresarial extensible basada en el modelo de representación de datos y el método de construcción.

1.4 METODOLOGÍA Y HERRAMIENTAS DE DESARROLLO

1.4.1 Metodología

La metodología usada para este trabajo como se explicó anteriormente se basa en un proyecto de investigación aplicada I+D.

En el ámbito de la investigación se realizará un estudio exploratorio sobre la representación de datos en los SR y su relación con un modelo particular de representación del contexto de trabajo en los sistemas colaborativos. Para lograr este objetivo se modelará un *framework* para SR. Por lo tanto, el tipo de investigación que se llevará a cabo será de índole exploratorio donde el objetivo es examinar un tema o problema de investigación poco estudiado o que no ha sido abordado antes (**?**). Se considera exploratorio ya que se ha encontrado poca literatura que aborde la representación de datos en SR. A partir de esto se contemplan las siguientes etapas:

- Revisión del estado del arte de SR para realizar una categorización.
- Revisión del *framework* conceptual *3-Ontology* para especificar los conceptos básicos del área que serán usados en la solución.
- Modelamiento del proceso de generación de SR.
- Modelamiento de dimensiones emergentes de SR.
- Comparación con otros *framework*.
- Evidenciar la completitud del *framework* mediante la representación teórica de diversos tipos de SR.

- Dar evidencia empírica del *framework* mediante la construcción de SR de filtrado colaborativo.

La metodología de desarrollo del software usada se basa en la filosofía ágil de **SCRUM**[12]. Sin embargo dado que el trabajo fue realizado de forma personal, no se especificaron los roles de la metodología. Se justifica una metodología ágil ya que los requerimientos de la herramienta usada varían durante el desarrollo *framework*. Luego, se tiene la flexibilidad necesaria para abordar cambios de requerimientos a lo largo del desarrollo. La documentación que se genera es simple y básica:

- Documentación del código mediante *Javadocs*[13].
- Diseño de clases.
- Modelo de componentes: especifica las interacciones de los distintos componentes.
- Diagrama de módulos.

1.4.2 Herramientas de desarrollo

Las herramientas que fueron usadas para la realización de este trabajo de tesis son las siguientes:

- Sistema operativo: *Linux* con su distribución *Ubuntu 12.04*.
- *Java SE Development kit 7*[14]: se utiliza para el desarrollo del software. Se justifica su uso por ser un lenguaje orientado a objetos de alto nivel que permite una representación

[12]https://www.scrum.org/
[13]http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html
[14]http://www.oracle.com/technetwork/java/javase/overview/index.htmltwork/java/javase/overview/index.html

legible del *framework* construido. Además, posee una comunidad bastante amplia y gran número de librerías que facilitan el desarrollo.

- *TeXstudio*[15]: herramienta para la construcción de documentos en **LaTeX**.
- *Eclipse Kepler*[16]: el **IDE** *eclipse* en su versión *kepler* que es usada para desarrollo de aplicaciones estándar de *Java*.
- *DIA*[17]: para el desarrollo de los diagramas subyacentes al código y los modelos construidos.
- *Notebook* personal: *Toshiba L505D. AMD Turion(tm) II Dual-Core Mobile M520 × 2.* 4 GB de memoria RAM.
- *Gradle*[18]: herramienta que permite el control del ciclo de vida del software, desde su construcción al despliegue. Adopta los estándares propuestos por *Apache Maven* como estructura de directorios y manejo de dependencias.
- *Git*[19]: herramienta para el control de versiones del código fuente y documentos asociados al trabajo de tesis.
- *BitBucket*[20]: repositorio remoto de *Git* donde se administra el control de versiones.
- Servidor *Huelen* del Departamento de Ingeniería Informática, **USACH**.

[15] texstudio.sourceforge.net
[16] http://www.eclipse.org/kepler/
[17] http://dia-installer.de/
[18] http://www.gradle.org/
[19] http://git-scm.com/
[20] http://bitbucket.org/

1.5 RESULTADOS OBTENIDOS

La representación de SR basado en el *framework* conceptual *3-Ontology* provee una forma de situar las interacciones de los usuarios hacia los ítems dentro de aplicaciones colaborativas como son los SR. Por lo tanto, se asume que las interacciones tienen un contexto que les da un sentido espacial, temporal y social. La herramienta construida RBOX 2.0 se basa en el modelo propuesto y cumple con las propiedades sistemáticas de mantenibilidad, flexibilidad, reusabilidad y escalabilidad. La eficacia del modelo se valida mediante la construcción de un SR de filtrado colaborativo *user-user* para *Movielens*, y otro SR de *Tagging Social* bajo un *dataset* propietario de una empresa de noticias. En cada SR construido se muestra la correspondencia del dominio de aplicación al modelo propuesto.

1.6 ORGANIZACIÓN DEL DOCUMENTO

En el Capítulo 2 se presenta un marco teórico donde se exponen los fundamentos del trabajo como son los conceptos básicos de los SR, características emergentes de los SR dentro del contexto de la información social y el *framework* conceptual *3-Ontology*. A continuación, en el Capítulo 3 se realiza la definición del modelo propuesto comenzando con una revisión de los trabajos relacionados para luego definir formalmente el modelo. En el Capítulo 4 se presenta el diseño e implementación de RBOX 2.0, la herramienta de software basada en el modelo de representación propuesto. Luego en el Capítulo 5 se presentan dos casos de estudio donde se valida la eficacia del modelo propuesto. En el Capítulo 6 se presentan las conclusiones, para

finalmente terminar con las referencias del trabajo.

CAPÍTULO 2. MARCO TEÓRICO

En este capítulo se exponen los conceptos teóricos del presente trabajo de tesis. Rellenar al final

2.1 MOTORES DE BÚSQUEDA VERTICALES

A medida que pasa el tiempo y la Web sigue creciendo, los motores de búsqueda se convierten en una herramienta cada vez más importante para los usuarios. Estas máquinas ayudan a los usuarios a buscar contenido dentro de la Web, puesto que conocen en qué páginas de la Web aparecen qué palabras. Sin un buscador los usuarios estarían obligados a conocer los localizadores de recursos uniformes (URL) de cada uno de los sitios a visitar. Además, los motores de búsquedas en cierto modo conectan la Web, ya que existe un gran número de páginas Web que no tienen referencia desde otras páginas, siendo el único modo de acceder a ellas a través de un motor de búsqueda.

Un motor de búsqueda está construído por diversos componentes, y su arquitectura típica la podemos ver en la Figura 2.1. Existe un proceso denominado *crawling*, éste posee una tabla con las páginas Web iniciales en las que se extrae el contenido de cada una de ellas. A medida que el *crawler* comienza a encontrar enlaces a otras páginas Web, la tabla de páginas a visitar crece. El contenido que se extrae en el procedimiento de *crawling* es enviado al proceso de indexamiento, este se encarga de crear un índice de las páginas visitadas por el *crawler*.

Dado el volúmen de datos involucrado en el procesamiento, se debe tener una estructura

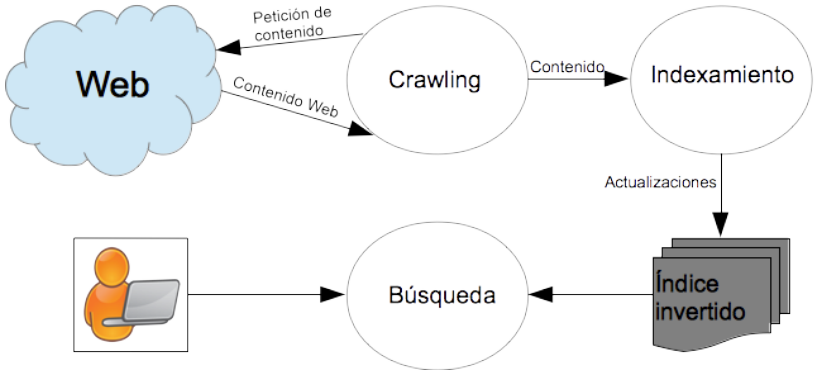

FIGURA 2.1: Arquitectura típica de un motor de búsqueda

de datos que permita encontrar cuáles páginas contienen las palabras presentes en la búsqueda, todo esto dentro de un período de tiempo aceptable. El índice invertido es una estructura de datos que contiene una lista con todas las palabras que el proceso de *crawling* ha visto, y asociada a cada palabra se tiene una lista de todas las páginas Web donde ésta palabra aparece mencionada. El motor de búsqueda construye esta estructura con el objetivo de acelerar el proceso de las búsquedas que llegan al sistema. El proceso de búsqueda es el encargado de recibir la consulta (query), generar un *ranking* de las páginas Web que contienen las palabras de la consulta y finalmente generar una respuesta. Las diversas formas de calcular la relevancia de una página Web será explicado en secciones posteriores.

En un motor de búsqueda se pueden encontrar diversos servicios tales como (a) cálculo de las mejores páginas Web para una cierta consulta; (b) construcción de la página Web con los resultados de la consulta; (c) publicidad relacionada con las *queries*; (e) sugerencia de *queries*; entre muchos otros servicios.

Lo que se hace hoy en día es agrupar computadores para procesar una consulta y producir la respuesta de ésta. Este conjunto de computadores recibe el nombre de *cluster*.

La diferencia entre un motor de búsqueda vertical y uno general, es que el primero se centra solo en un contenido específico de la Web. El *crawler* también debe extraer solo el contenido de aquellas páginas Web que están dentro del dominio permitido. Al ser un dominio acotado,

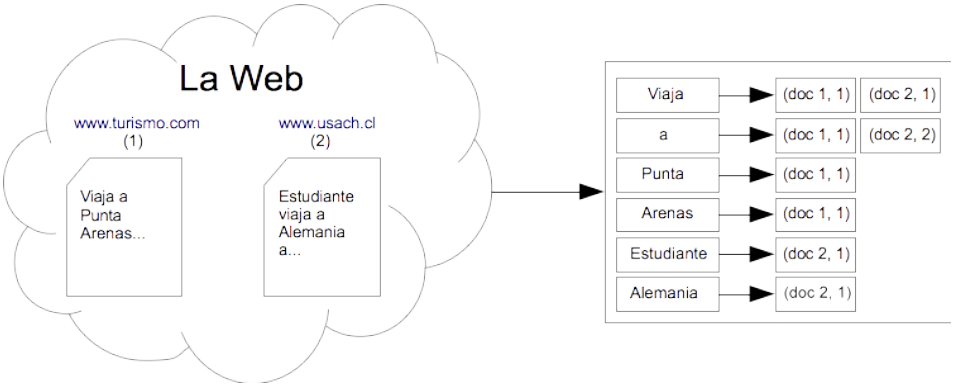

FIGURA 2.2: Índice invertido

las páginas Web a procesar serán menos y por tanto la lista de los términos del índice invertido serán eventualmente de menor tamaño. Sin embargo, en un motor de búsqueda vertical las actualizaciones al índice invertido ocurren con mayor frecuencia.

2.2 ÍNDICE INVERTIDO

Es una estructura de datos que contiene todos los términos (palabras) encontrados. A cada uno de los términos, está asociado una lista de punteros a los documentos (páginas Web) que contienen dicho término. Además se almacena información que permita realizar el *ranking* de las respuestas a las consultas (queries) que llegan al sistema, por ejemplo, el número de veces que aparece el término en el documento. Esta lista recibe el nombre de lista invertida.

Para construir un índice invertido se debe procesar cada palabra que existe en un documento, registrando su posición y la cantidad de veces que éste se repite. Cuando se procesa el término con la información asociada correspondiente, se almacena en el índice invertido (ver Figura 2.2).

El tamaño del índice invertido crece rápido y eventualmente la memoria RAM se agotará antes de procesar toda la colección de documentos. Cuando la memoria RAM se agota, se almacena en disco el índice parcial hasta aquel momento, se libera la memoria y se continúa con el proceso. Además, se debe hacer un *merge* de los índices parciales uniéndo las listas invertidas de cada uno de los términos involucrados.

2.3 ESTRATEGIAS DE EVALUACIÓN DE QUERIES

Existen dos principales estrategias para encontrar los documentos y calcular sus respectivos puntajes de una determinada *query*. Estas son (a) *term-at-a-time* (TAAT) y (b) *document-at-a-time* (DAAT).

2.3.1 TAAT

Este tipo de estrategia procesa los términos de las *queries* uno a uno y acumula el puntaje parcial de los documentos. Las listas invertidas asociadas a un término son procesadas secuencialmente, esto significa que los documentos presente en la lista invertida del término t_i, obtienen un puntaje parcial antes de comenzar el procesamiento del término t_{i+1}. La secuencialidad en este caso es con respecto a los términos contenidos en la *query*.

2.3.2 DAAT

En este tipo de estrategias se valúa la contribución de todos los términos de la query con respecto a un documento antes de evaluar el siguiente documento. Las listas invertidas de cada término de la *query* son procesadas en paralelo, de modo que el puntaje del documento d_j se calcula considerando todos los términos de la query al mismo tiempo. Una vez que se obtiene el puntaje del documento d_j para la *query* completa, se procede al procesamiento del documento d_{j+1}.

2.3.3 Consideraciones

Las estrategias TAAT son mayormente usadas en sistemas de recuperación de información (IR), como son los motores de búsqueda. Cuando se tiene un índice invertido pequeño, las estrategias TAAT rinden adecuadamente, sin embargo cuando los índices invertidos son de gran tamaño las estrategias TAAT poseen dos grandes ventajas: (a) Requieren menor cantidad de memoria para su ejecución, ya que el puntaje parcial por documento no necesita ser guardado y (b) Explotan el paralismo de entrada y salida (I/O) más eficientemente procesando las listas invertidas en diferentes discos simultáneamente.

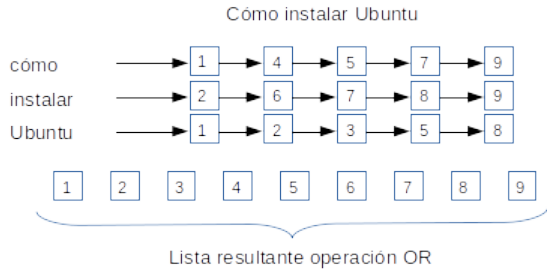

FIGURA 2.3: Operación OR

2.4 OPERACIONES SOBRE LISTAS INVERTIDAS

Cuando una *query* llega al motor de búsqueda, cada término tiene asociado una lista con todos los documentos en los cuales aparece dicho término. El sistema debe decidir qué documentos se analizarán para obtener la respuesa con el conjunto de los K mejores. A continuación se presentan las diferentes formas de operar los documentos pertenecientes a las listas invertidas de una *query*.

2.4.1 OR

Este operador toma las listas invertidas de cada uno de los términos de la *query* y ejecuta la disyunción entre ellas. El resultado de este operador es una lista invertida con todos los documentos que contengan al menos un término de la query. Finalmente, esta lista invertida se ocupará para obtener los mejores K documentos. Un ejemplo sencillo se muestra en la FIGURA 2.3.

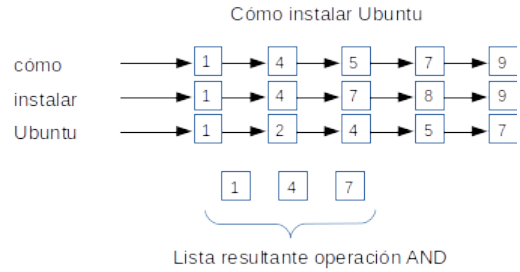

FIGURA 2.4: Operación AND

2.4.2 AND

Este operador ejecuta la conjunción entre las listas invertidas de los términos de una *query*. Se obtiene una lista invertida con los documentos que contengan todos los términos de la *query*. Se debe notar que aquí se obtiene una lista resultante de menor que la obtenida en el operador OR (Ver FIGURA 2.4).

2.4.3 WAND

CAPÍTULO 3. CONCLUSIONES

REFERENCIAS