# Distributed Directory System: A Healthcare Use Case for Rural Areas

Alethia Hume*, Fausto Giunchiglia* and Luca Cernuzzi†
*Department of Information Engineering and Computer Science
University of Trento, Trento, Italy
Email: hume, fausto@disi.unitn.it
http://www.disi.unitn.it
† Departamento de Ingeniería Electrónica e Informática
Universidad Católica "Nuestra Señora de la Asunción", Asuncion, Paraguay
Email: lcernuzz@uca.edu.py
http://www.uca.edu.py

***Abstract*—The digital content of users is commonly organised in local directories representing entities from the real world (e.g., people, locations, organisations, and events). Different representations can show different "versions", using different names to refer to the same real world entity (e.g., George Lombardi, Lombardi G., Dr. Lombardi). Although the data in these directories are related and can even complement each other, there are no formal links connecting them and allowing users to share and search across them. In this work we propose a *Distributed Directory System*, applied to *A Healthcare Use Case for Rural Areas* that allows peers: (i) to maintain full control over their data; and (ii) to find different versions of an entity based on any name that is used in the network to refer to it. We evaluate the approach in networks of different sizes using PlanetLab and we show promising results in terms of scalability.**

***Index Terms*—Distributed Directory, Entity, P2P, Name-based Search.**

## I. Introducción

Vemos Internet como una red de usuarios – de aquí en adelante llamados pares – es decir, una red P2P que organiza su contenido en directorios, los cuales almacenan representaciones digitales de sus propias versiones de *entidades* que existen en el mundo real. Las entidades pueden ser de diferentes tipos, por ejemplo, (e.g., persona, lugar, evento y otras), tienen un nombre, y son descritas a través de atributos (e.g., latitud-longitud, tamaño, fecha de nacimiento) que son diferentes para distintos tipos de entidades [1]. Diferentes versiones de una entidad pueden representar diferentes puntos de vista, mostrando distintos aspectos de la entidad o el mismo aspecto con diferentes niveles de detalle. En cierto modo, las representaciones locales de los pares pueden ser vistas como partes de la información relativa a una entidad en particular y que son almacenadas de manera distribuida en la red.

En esta red, los diferentes directorios contienen datos relacionados y, en cierta medida, pueden complementarse unos a otros. Uno de los problemas que nos impide hacer uso de la relación entre estos datos es la falta de enlaces que conecten a los directorios locales de los pares. Una iniciativa que tiene como objetivo conectar datos relacionados en la web ha sido la de Linked Data[1], la cual ha permitido vincular conjuntos de datos importantes, tales como dbpedia, Freebase, DBLP, ACM, y otros. Sin embargo, este enfoque deja fuera de la web semántica a los usuarios individuales, es decir a los simples pares normales, y los datos de sus directorios locales que pueden estar almacenados en dispositivos personales como por ejemplo, ordenadores, portátiles, teléfonos inteligentes, PDAs, etc. En este trabajo proponemos un directorio distribuido que construye los enlaces de conexión entre los directorios locales a este nivel, es decir, el nivel de simples pares. Es importante tener en cuenta que todo el directorio puede ser visto como otro conjunto de datos, lo que podría ser considerado como un nodo en el gráfico de Linked Data. Esto significa que ambos enfoques no resultan mutuamente excluyentes sino por el contrario podrían nutrirse el uno del otro. De esta manera, el directorio se convertiría en el puente que permite a los simples pares participar como parte de la Web Semántica en lugar de actuar sólo como consumidores de ella.

Al igual que en cualquier directorio, un par normalmente identifica y distingue una entidad de las demás por medio de nombres, los cuales desempeñan un papel diferente al de los demás atributos, ya que estos se constituyen en identificadores en lugar de descripciones [2]. Por ejemplo, George Lombardi, Trento, Italia, Universidad de Trento son nombres que se refieren a una persona, una ciudad, un país, y una universidad respectivamente. Los valores de otros tipos de atributos tienen un significado que se puede entender, por ejemplo, mediante la asignación a conceptos de una base de conocimientos, como WordNet[2]. Los nombres, por otro lado, son cadenas de caracteres que se comportan de manera similar a las palabras claves. Las entidades del mundo real pueden ser llamadas por varios nombres, como consecuencia de variaciones y errores. Entonces, el conjunto de nombres utilizados en distintas representaciones locales para identificar a la misma entidad del mundo real pueden ser diferentes, al mismo tiempo que los conjuntos de nombres usados para identificar diferentes

[1] http://linkeddata.org/
[2] http://wordnet.princeton.edu/

978-1-4673-9143-6/15/$31.00 

entidades del mundo real pueden superponerse.

El enfoque que proponemos para un *Distributed Directory System (DDS)* incorpora la noción de una entidad del mundo real descrita por diferentes representaciones locales de los pares[3]. Esta noción se utiliza para organizar las referencias a las representaciones locales con el fin de permitir la búsqueda de toda la información disponible sobre las entidades descritas en la red. Nuestro sistema ofrece dos características principales:

- En primer lugar, toma en consideración el hecho de que múltiples nombres, posiblemente distintos, puedan ser utilizados para identificar la misma entidad del mundo real (e.g., George Lombardi vs. G. Lombardi and Italy vs. Italia).
- En segundo lugar, permite a los pares tener el control sobre la privacidad de sus datos debido a que el directorio almacena únicamente los nombres de la entidad y un enlace a la representación local de la misma.

Como resultado, cualquier nombre que es usado en alguna representación local para identificar a una entidad puede ser utilizado para buscar las diferentes versiones de esa entidad que se encuentren almacenadas en la red de pares.

Este documento esta organizado de la siguiente forma, la Sección II presenta un caso de uso que tiene como objetivo motivar la discusión considerando un mundo compuesto de directorios con datos relacionados. En la Sección III se formalizan las nociones básicas que conectan los diferentes directorios, mientras que en la Sección IV se discute el problema de la coincidencia de nombres que surge al vincular diferentes directorios. A continuación, se propone un directorio distribuido de entidades en la Sección V y los algoritmos para realizar la búsqueda en tal directorio se introducen en la Sección VI. Los detalles referentes a la implementación y evaluación son discutidos en la Sección VII. Por último, los trabajos relacionados se discuten en la Sección VIII y las conclusiones se presentan en la Sección IX.

## II. Caso de Uso: Directorios de Información Enfocados en el Area de la Salud

Hoy en día, la mayor parte de nuestros datos se encuentran organizados en directorios. Un antiguo y conocido ejemplo es el directorio telefónico, utilizado para organizar las direcciones y los números de teléfono de personas y empresas. Otras formas más recientes de directorios se pueden encontrar, por ejemplo, en las listas de contactos, directorios de productos y servicios, directorios de documentos, directorios de eventos (i.e., calendarios o agendas) que se utilizan en dispositivos actuales (e.g., ordenadores, PDAs, teléfonos inteligentes) para organizar localmente la información referente a entidades del interés de sus usuarios. Por otra parte, los datos de diferentes directorios, posiblemente de diferentes pares, pueden estar relacionados. Diferentes pares que planifican asistir al mismo evento pueden almacenar sus propias representaciones locales de dicho evento, o diferentes proveedores de un producto o un servicio podrían almacenar sus propias descripciones locales de dicho producto/servicio.

Consideremos un ejemplo en el área de la salud que involucra a proveedores y usuarios de servicios de salud principalmente en las áreas rurales de países en vías de desarrollo. Este caso de uso incluye a organizaciones como farmacias y clínicas, ya sean publicas o privadas, así como al personal de salud, incluyendo doctores, enfermeras, etc, y a los pacientes. En zonas rurales es común encontrar ciudades medianas rodeadas de distintas comunidades más pequeñas que muchas veces se encuentran semi-aisladas, sobre todo en materia de información sobre los servicios, por ejemplo, de salud a los que sus pobladores pueden acceder localmente o en comunidades vecinas. Dentro de este contexto existe una necesidad real de publicar y compartir información sobre dichos servicios, es decir, construir redes de información sobre servicios de salud.

Por un lado tenemos el caso de las farmacias que necesitan compartir información acerca de la disponibilidad de medicamentos (i.e., directorio de medicamentos) con, *(i)* los pacientes para que ellos sepan donde encontraran lo que les fue prescrito por el doctor, *(ii)* otras farmacias para coordinar cuando necesita re-abastecerse, y *(iii)* los doctores quienes podrían estar interesados en saber cuál es la disponibilidad de la farmacia local (o la más cercana) antes de prescribir una medicina a sus pacientes. Por otro lado tenemos también a diferentes clínicas, quienes necesitan compartir información relativa al personal de cada clínica (i.e., directorio de profesionales), la disponibilidad de los mismos (i.e., agenda), y como se puede contactar con ellos (i.e., lista de contactos). Similar información necesitan también compartir los mismos doctores directamente con los pacientes que quieran consultar con ellos de forma privada (i.e, en el consultorio privado de los doctores). La información sobre los profesionales de la salud pueden ser muy útil para los pacientes que quieran planificar una visita medica. Un paciente podría tomar la decisión de trasladarse hasta una clínica en una comunidad vecina si encuentra que el médico con el que quiere tratarse tiene mayor disponibilidad allí o si se encuentra disponible en un día/hora que le resulta más conveniente.

Veamos en detalle el ejemplo de la Figura 1 que en la parte superior muestra la lista de profesionales de diferentes pares, las cuales pueden ser vistas como directorios locales de personas. Los diferentes pares de esta red pueden almacenar distinta información sobre las personas descritas en sus directorios locales (por ejemplo, disponibilidad asociadas a diferentes lugares, números de teléfono, direcciones de correo electrónico, y otros), mostrando diferentes horarios y lugares donde dichos profesionales prestan servicio así como la maneras de ponerse en contacto con ellos. Por ejemplo, supongamos que $p_1$ es una clínica en la que el Dr. George Lombardi trabaja. Es así que $p_1$ tiene almacenada en su lista de profesionales la información relativa a la agenda del doctor con la disponibilidad del mismo en los días y durante el horario en que se encuentra en la clínica. En la descripción que dicha clínica tiene almacenada sobre el Dr. Lombardi también se

[3]Una versión preliminar y reducida de este trabajo fue presentada como póster en [3]

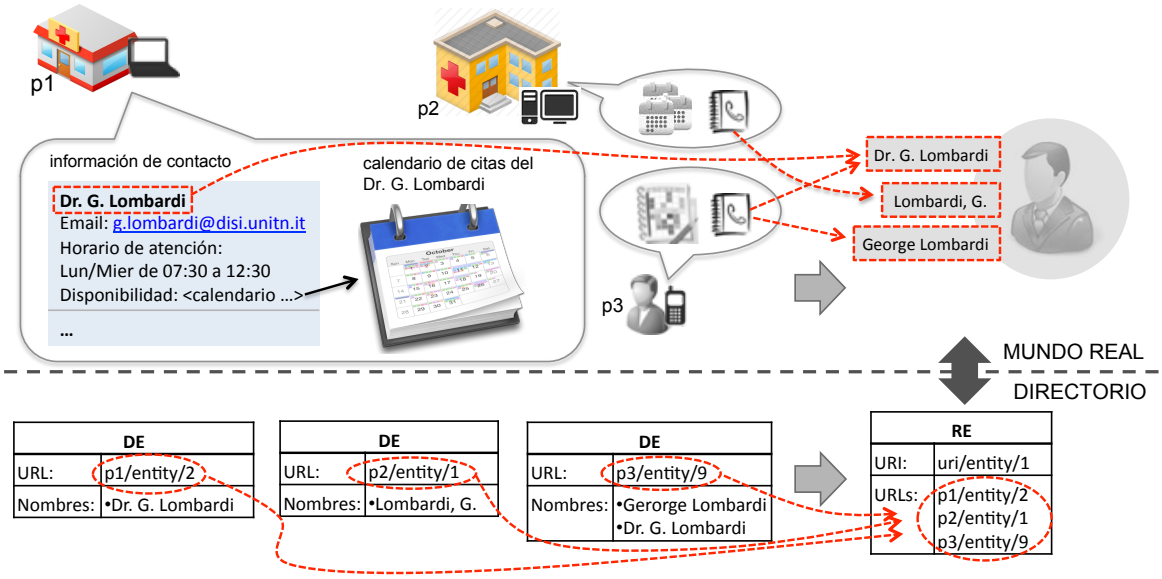


Fig. 1. Ejemplo de listas de contacto

incluye información que permite ponerse en contacto con él, por ejemplo, numero de oficina, número teléfono, etc. Otra clínica $p_2$ ubicada en una comunidad vecina y en donde también trabaja el Dr. Lombardi, tendría que tener información complementaría a la anterior, referente a la disponibilidad del mismo y su información de contacto en ésta clínica, por ejemplo, su dirección de correo electrónico, número de oficina y teléfono en dicha clínica, etc. El par $p_3$ representa al mismo Dr. Lombardi quien podría tener almacenada información sobre si mismo, por ejemplo, la dirección de su consultorio privado, disponibilidad para recibir pacientes, así como su domicilio, número de teléfono, su agenda privada, entre otros.

Ahora supongamos que un paciente, al que llamamos $p_4$, oye hablar del Dr. Lombardi y quiere consultar con él. Podemos ver que:

1) En primer lugar, la información que $p_4$ necesita está distribuida en la red y su problema se traduce en encontrar dónde están almacenadas las diferentes piezas.
2) En segundo lugar, los diferentes pares pueden llamar a la misma persona usando diferentes nombres, por ejemplo, Dr. Lombardi, George Lombardi, G. Lombardi. En nuestro ejemplo, esto significa que $p_4$ necesita estar seguro de que los demás pares (es decir, $p_1$, $p_2$ y $p_3$) se están refiriendo a la misma persona.
3) En tercer lugar, la información puede cambiar con el tiempo. Los números de teléfonos a través de los cuales se lo puede contactar cambiarán si cambia de lugar de trabajo, y su disponibilidad cambiará a medida que vaya dando nuevas citas a sus pacientes.
4) Por último, la privacidad y la sensibilidad de la información deben ser consideradas, el número de teléfono y la dirección del domicilio del Dr. Lombardi son datos más sensibles en comparación con el el número de teléfono de las clínicas o el consultorio donde trabaja. Como consecuencia, $p_3$ no compartirá dicha información con todo el mundo.

Es importante señalar que, en el contexto de países en vías de desarrollo, la construcción de un sistema centralizado global que conecte y mantenga el tipo de información como la descrita en esta sección requeriría de una importante inversión económica y tecnológica. A diferencia de otras áreas, en las zonas rurales es común que los pobladores no cuenten con un acceso estable, o de alta velocidad, a internet. Debido a la limitación en cuanto a los recursos del estado es también común que no existan portales basados en la web que permitan la integración de dicha información. Como consecuencia la inversión necesaria supera ampliamente la disponibilidad de recursos de los principales actores interesados. Teniendo en cuenta estos factores, nuestro enfoque propone un *modelo para la construcción de un directorio distribuido* utilizando la limitada tecnología disponible, es decir, ordenadores de bajo costo, teléfonos móviles, computadores portátiles, y otros dispositivos que podrían estar conectados a través de conexiones a internet de baja velocidad, intranets, etc.

## III. Conectando Directorios

Definimos un Directorio de Entidades que formaliza los vínculos entre los datos de diferentes directorios a través de la distinción entre una Entidad Digital ($DE$) y una Entidad del Mundo Real ($RE$). Una $DE$ se define como una representación digital local de una entidad que existe en el mundo real. Se utiliza un $URL$ (localizador de recursos uniforme) –siglas en inglés de *uniform resource locator*– con el fin de identificar unívocamente una $DE$ y puede ser también utilizado para obtener la descripción local completa de la entidad correspondiente, es decir, su descripción basada en atributos. También consideramos un conjunto de nombres $\{N\}$ como identificadores que son legibles por humanos, son

utilizados en una $DE$ para referirse a una $RE$ y lo distinguen de los demás. Formalmente,

$$DE = \langle URL, \{N\} \rangle \quad (1)$$

Por otro lado, una $RE$ representa a la entidad del mundo real y se modela como un conjunto de $DEs$. Utilizamos un $URI$ (identificador de recursos uniforme) –siglas en inglés de *uniform resource identifier*– para identificar de forma única cada $RE$. Formalmente,

$$RE = \langle URI, \{URL\} \rangle \quad (2)$$

donde $\{URL\}$ es un conjunto no vacío de identificadores de diferentes $DEs$ que describen $RE$. Como consecuencia de la composición de estas definiciones podemos ver que varios conjuntos de nombres se asignan a una $RE$ a través de las definiciones $DE$ de diferentes pares que describen la misma $RE$.

En la segunda parte de la Figura 1 (parte inferior) mostramos como el ejemplo de la Sección II se puede formalizar usando estas nociones (i.e., $DEs$ and $REs$). Podemos ver un mapeo de *uno-a-uno* entre la $RE$ de un directorio de entidades y la persona representada en diferentes listas de contactos. Además, vemos que una entrada de una lista de contactos se traduce en una $DE$ en el directorio (i.e., también en este caso existe un mapeo de *uno-a-uno*). Hay una relación de *uno-a-muchos* entre $REs$ y $DEs$, lo que demuestra que cada entrada individual en una lista de contactos corresponde a una persona pero una persona puede ser descrita en muchas entradas diferentes (posiblemente de diferentes pares). Por último, la relación entre los $Nombres$ y las $REs$ introduce un problema de coincidencia de nombre que se discute con más detalles en la siguiente sección.

Es importante tener en cuenta que estas nociones permiten la separación entre "que" está siendo representado y "donde" está siendo representado. Esta separación es necesaria para modelar los problemas indicados en los puntos 1 y 2 correspondientes al ejemplo de la Sección II. Las $DEs$ modelan las diferentes piezas de información que $p_4$ necesita y sus $URLs$ indican donde están almacenadas. La $RE$ modela el enlace que conecta diferentes $DEs$ y su $URI$ identifica lo que representa. Con relación a el punto 2, podemos ver que diferentes conjuntos de nombres son asignados en las $DE$, lo cual modela el echo de que $p_1$, $p_2$ and $p_3$ pueden definir los diferentes nombres que utilizan para llamar a una entidad.

Por otra parte, la distinción entre las dos nociones ($DE$ y $RE$) también proporciona la herramienta para hacer frente a las cuestiones introducidas por los otros dos punto (i.e., los puntos 3 y 4 de la Sección II). El dinamismo de la información referente a las entidades y la privacidad de los datos locales se limitan a afectar a las $DEs$. De esta manera, cuando la dirección de correo electrónico del Dr. Lombardi cambia (ver Figura 1), $p_3$ (el medico) actualiza su representación local (i.e., la $DE$). La definición de la $RE$ correspondiente no se ve afectada por esta actualización, no obstante, la información disponible en la red P2P sobre el Dr. George Lombardi se actualiza. Del mismo modo, el control de acceso se puede implementar directamente sobre los datos asociados a cada representación de una $DE$ individual sin que esto afecte a las definiciones de $REs$. Cabe mencionar que dicha implementación de control de acceso está fuera del alcance de este documento, pero se invita a los lectores interesados a ver por ejemplo [4].

## IV. Coincidencia de Nombres

Los nombres son identificadores legibles por humanos que sirven al propósito de distinguir una entidad de las demás. Pueden ser definidos como etiquetas compuestas por una combinación de palabras, números y símbolos [2]. En el contexto de nuestro directorio de entidades, definimos el conjunto de nombres que identifican a una $RE$ como la unión de los nombres usados en las diferentes $DEs$ de los pares que representan localmente a dicha $RE$. Los nombres son distintos a otros atributos, ya que juegan el papel de palabras claves y no tienen un significado que pueda ser descrito asociándolos a conceptos de una base de conocimientos. Como tal, los nombres pueden sufrir de diferentes tipos de variaciones. Tomando en cuenta los resultados del estudio realizado en [5], podemos distinguir entre los siguientes tipos:

- **Formato.** Las variaciones de formato son altamente dependientes del tipo de entidad y afectan principalmente a los nombres de personas. Este tipo incluye la variación del orden en el cual las palabras del nombre pueden ser escritas (e.g., *George Lombardi* and *Lombardi, George*) y las múltiples abreviaciones que pueden existir para el mismo nombre completo (e.g., *Giulio Augusto Lombardi* puede ser abreviado como *G. A. Lombardi*, *Giulio A. Lombardi* y otros). Es también importante notar que la abreviación de un nombre puede ser una referencia válida para nombres diferentes (e.g., *G. Lombardi* es válido para referirse a *George Lombardi* pero también para referirse a *Giulio Lombardi*).
- **Traducciones completas.** Los nombres a veces cambian completamente cuando son escritos en un idioma diferente (e.g., *Trento* en Italiano, *Trient* en Alemán or *Trent* en Inglés).
- **Traducción parcial.** En otros casos solamente una parte del nombre es traducida cuando se escribe en otro idioma. Este es el caso cuando el nombre está compuesto por sustantivos comunes y propios, donde el sustantivo común es llamado desencadenante en [5] y es la única parte que se ve afectada por la traducción (e.g., *University of* Trento vs. *Universidad de* Trento).
- **Errores de ortografía.** Los nombres podrían estar mal escritos, ya sea en la definición de una $DE$ o en la especificación de una consulta (i.e., solicitud de búsqueda). Los errores pueden ser consecuencia de una variación en la puntuación, el uso de mayusculas, la utilización de espacios, así como debido a omisiones, sustituciones o variaciones fonéticas (e.g., *Fasuto* vs. *Fausto*, *G Lombardi* vs. *G. Lombardi*).
- **Seudónimos.** Las entidades también tienen seudónimos que no son (necesariamente) variaciones de nombres sino

mas bien nombres alternativos de la entidad, los cuales pueden ser definidos (y usados) en diferentes contextos. Este es el caso de algunos apodos arbitrarios que a veces son utilizados por los pares para referirse a una $DE$ (e.g., *Fede* es frecuentemente usado como apodo para *Federico* o *Federica* y *El rey del Rock & Roll* es un apodo común para referiste a *Elvis Presley*).

Las variaciones de nombre, junto con la definición de $DE$ presentada anteriormente, muestran que la relación entre los nombres y las $DEs$ es del tipo *muchos-a-muchos*. A su vez, esto lleva a un problema de coincidencia de nombre cuando tenemos la intención de buscar una entidad basándonos en su nombre [2]. Este problema, en el contexto de un directorio de entidades, se puede descomponer en:

1) *El problema de mapear nombres dentro de la red:* Un nombre usado en una $DE$ puede ser una variación de un nombre usado en otra $DE$ que representa la misma $RE$. Necesitamos tener en cuenta todos los diferentes nombres (incluyendo variaciones) usados en la red para identificar una $RE$ y mapearlos a todos los diferentes $DEs$ que describen la $RE$. En el ejemplo de la Figura 1, si el usuario está buscando una entidad con el nombre *"George Lombardi"*, el directorio debería retornar todas las $DEs$ (i.e, *p1/entity/2*, *p2/entity/1* and *p3/entity/9*) que representan las diferentes versiones de *uri/entity/1* en lugar de retornar solo aquella que usa dicho nombre (i.e., *p3/entity/9*).
2) *El problema de mapear una consulta con los nombres usados en la red:* Este caso considera una consulta que contiene nombres que son desconocidos para el directorio de entidades, pero que son variaciones de uno o más nombres conocidos dentro de la red. Decimos que un nombre es desconocido para el directorio si no existe ninguna $DE$ en la red que utilice dicho nombre para identificar una $RE$. El ejemplo más sencillo es el de una consulta que contiene un nombre mal escrito con respecto a las $DEs$ del directorio. En el ejemplo de la Figura 1, si el usuario inserta la consulta *"Goerge Lombardi"*, la búsqueda debería ser capaz de encontrar a *"George Lombardi"* como una posible repuesta (i.e., una entidad candidata a ser una respuesta a la pregunta).

## V. Un Sistema de Directorio Distribuido

En este estudio proponemos un Sistema de Directorio Distribuido (DDS – siglas en inglés de *Distributed Directory System*) que organiza la información referente a entidades incorporando las nociones de $RE$ y $DE$ que fueron presentadas en la Sección III. Estas nociones permiten la separación del problema de encontrar las $DEs$ que representan diferentes versiones de una $RE$ del problema de encontrar $REs$ que son identificadas con múltiples nombres. El enfoque que proponemos aprovecha esta separación mediante la construcción de dos índices diferentes, uno para hacer frente a cada problema.

Un ***DEindex*** es creado para conectar $REs$ (i.e., $URIs$) a $DEs$ (i.e., $URLs$) y puede ser definido formalmente como,

$$DEindex = \{RE \rightarrow DE \mid \nexists RE' \rightarrow DE \in DEindex \; s.t., RE' \neq RE\} \quad (3)$$

Como se puede ver, este indice codifica una relación *uno-a-muchos* que existe entre $REs$ y $DEs$ debido a que diferentes $REs$ no pueden corresponder a la misma $DE$.

Por otro lado, un ***REindex*** es creado para conectar los nombres que son dados (en representaciones locales) a las $REs$ (i.e., $URIs$). Si llamamos $\{N^{DE}\}$ al conjunto de nombres de una entidad digital $DE$, entonces el ***REindex*** puede ser definido formalmente como,

$$REindex = \{N \rightarrow RE \mid \exists RE \rightarrow DE \in DEindex \; s.t., N \in \{N^{DE}\}\} \quad (4)$$

Podemos ver que este indice codifica una relación *muchos-a-muchos* entre los $Nombres$ y las $REs$ por que la única restricción establecida en la definición está relacionada con la existencia de una representación local que de "soporte" a dicha asignación.

A continuación discutimos con más detalle como se realizan la publicación, el mantenimiento y la búsqueda de entidades en el ***DDS***:

La *publicación y eliminación de* $DEs$ son los dos eventos principales que modifican el $DDS$ al afectar el contenido de los índices definidos anteriormente. La publicación de una $DE$ afecta a los dos indices de forma directa. En primer lugar, la $DE$ es asociada a la $RE$ que representa mediante la adición de la relación correspondiente (i.e., $RE \rightarrow DE$) al $DEindex$. En segundo lugar, las relaciones $N_i^{DE} \rightarrow RE$, entre cada nombre $N_i^{DE}$ en $\{N^{DE}\}$ y la $RE$ que está asociada a la $DE$, se agregan al $REindex$. Con el fin de hacer esto, asumimos que el par almacena en caché localmente el identificador (i.e., el $URI$) de la $RE$ que es representada por su $DE$[4]. Por otro lado, cuando una $DE$ es eliminada de la red, sólo el $DEindex$ se ve directamente afectado. La misma asignación $RE \rightarrow DE$ que se agrega cuando una $DE$ es publicada, luego se remueve del $DEindex$ cuando el par elimina dicha $DE$. Con respecto al $REindex$, decimos que no se ve directamente afectado porque las asignaciones de nombres no pueden ser removidas sin antes haber comprobado que ya no son válidos para identificar a la $RE$ correspondiente. Dicha verificación se discute como parte del mantenimiento del ***DDS***.

El *mantenimiento* del ***DDS*** se realiza a través de controles periódicos sobre los índices con el fin de detectar y eliminar las entradas que ya no son válidas. En el $DEindex$, una entrada puede considerarse inválida si contiene una asignación a una $DE$ que ha estado inaccesible durante mucho tiempo. Para detectar esta situación, a cada entrada se adjunta una marca de tiempo que indica la última vez en que la $DE$ estaba accesible. Esta marca de tiempo se actualiza en cada control

[4]Hay que tener en cuenta que la identificación inicial de la $RE$ descrita por una $DE$ constituye un problema de gestión de identidades y esta fuera del alcance de este trabajo. Véase por ejemplo [6], [7]

periódico. Cuando la $DE$ no es accesible, el intervalo entre la última vez que estaba accesible y la hora actual se verifica. La entrada correspondiente se elimina del $DEindex$ si tal intervalo excede un umbral dado. Una entrada del $REindex$, por otro lado, se considera inválida si contiene una asignación que no cumple con la restricción establecida por la definición de dicho índice, es decir, en la ecuación 4. Esto significa que una asociación entre $N$ y $RE$ tiene que ser removida del $REindex$ cuando ya no hay $DEs$ en la red que utilicen el nombre $N$ para referirse a tal $RE$. En otras palabras, cuando ninguna de las entidades disponibles proporcionan soporte a dicha asignación.

La *Búsqueda* en el $DDS$ puede realizarse usando dos tipos diferentes de identificadores, $URIs$ y $nombres$. En este contexto, cuando se tiene como entrada un $URI$ significa que la $RE$ objetivo ha sido plena y únicamente identificada. Por lo tanto, el objetivo de la búsqueda es el de obtener todas las diferentes representaciones (i.e., las $DEs$) de la $RE$. Por otro lado, en una búsqueda basada en nombres el objetivo es el de encontrar a las $RE$ que son candidatas a ser la respuesta correcta, como consecuencia de la relación *muchos-a-muchos* entre nombres y $REs$. Luego de que las $REs$ candidatas han sido encontradas, podemos usar la búsqueda basada en $URI$ para obtener las diferentes representaciones de las mismas. En lo que sigue, la búsqueda por nombres se discute con más detalles, mientras que la búsqueda basada en $URIs$ se incluye como parte ella.

Una consulta se define formalmente como $Q = \{N^Q\}$, donde $\{N^Q\}$ es el conjunto no vacío de nombres que son usados para identificar una $RE$ objetivo. Entonces, el problema de buscar entidades en base a sus nombres puede ser visto como la recuperación de $REs$ que son descritas en la red por al menos una $DE$, de manera tal que la intersección entre $\{N^{DE}\}$ y $\{N^Q\}$ no está vacía. Esta definición incluye coincidencias parciales entre $\{N^{DE}\}$ y $\{N^Q\}$ con el fin de permitir la búsqueda de una $RE$ a partir de cualquiera de los nombres asignados a ella en diferentes $DE$. A su vez, esto se puede traducir en la definición formal de la Respuesta de Consulta $RC$ de la siguiente manera:

$$\begin{aligned} RC = \{\langle RE, \{DE\}\rangle \mid \\ \exists N' \in \{N^Q\} : N' \rightarrow RE \ \in \ REindex \\ \wedge \ \forall DE' \in \{DE\} : RE \rightarrow DE' \in DEindex\} \end{aligned} \quad (5)$$

Como mencionamos antes, esta respuesta se construye en dos pasos. Los algoritmos que realizan dichos pasos se presentan en la Sección VI.

## VI. Algoritmos

En esta sección, asumimos que los índices ofrecen interfaces de llamadas APIs –siglas en inglés de *Application Programming Interface*– no bloqueantes (permitiendo la paralelización de las búsquedas en los índices), lo que significa que una llamada a la función $GET$ retorna inmediatamente una referencia a un objeto que se llenará con los resultados de las operaciones de búsqueda de índice.

**Algorithm 1** Estructuras de Datos Globales

```
1: REAnswer : ⟨isComplete, name, reAnsValues⟩
2: DEAnswer : ⟨isComplete, URI, deAnsValues⟩
3: isComplete : boolean    ▷ TRUE cuando la operación de
   búsqueda en el indice finaliza
4: reAnsValues : NULL OR {URI} OR {URL} OR {{URI}
   ∪ {URL}}
5: deAnsValues : {URL}    ▷ Conjunto no vacío de URLs
```

En el Algoritmo 1, definimos las estructuras de datos globales, que están estrictamente relacionadas con los índices. Dichas estructuras se utilizan en las distintas funciones implicadas en la búsqueda. Usamos la sentencia *for all* (linea 6 en el Algoritmo 2 y linea 8 en el Algoritmo 3) para denotar la ejecución concurrente de las sentencias que se encuentran en su cuerpo (i.e., linea 7 en el Algoritmo 2 y lineas 9 a 24 en el Algoritmo 3).

**Algorithm 2** Buscar Entidad

```
1: function SEARCHENTITY(names : {name}) : {⟨RE,
   relevance⟩}
2:     REs : {⟨RE, relevance⟩}    ▷ Almacena los resultados
3:     RE : ⟨URI, {URL}⟩ ▷ {URL}.size == 1 cuando URI
   == NULL
4:     relevance : integer
5:     REs := {}
6:     for all name ∈ names do              ▷ Hilos paralelos
7:         HANDLEREANSWER(GetREindex(name), REs)
8:     end for
9:     return REs
10: end function
```

La función Buscar Entidad se presenta en el Algoritmo 2 y es el principal punto de entrada para la búsqueda basada en nombres. Esta función recibe un conjunto de nombres como consulta y devuelve un conjunto de $REs$ candidatas de acuerdo con las restricciones dadas en la ecuación 5. Para medir la relevancia de cada $RE$ candidata, contamos el número de nombres de la consulta que coinciden con los nombres asociados a la $RE$. Dicho valor de relevancia se asocia a cada $RE$ candidata y se incluye en el conjunto de resultados. En la línea 7, el primer paso de la búsqueda por nombres se inicia con la llamada a la función $GetREindex$ del $REindex$. El objeto devuelto por la función se pasa al manejador correspondiente, el cual sabe cómo procesarlo.

El Algoritmo 3 muestra la función $HandleREAnswer$, que se encarga de procesar los valores recuperados del $REindex$. Podemos ver en las líneas 4 a 6 el bucle que espera hasta que se complete la respuesta. Luego, en la línea 8, iniciamos un hilo de ejecución para procesar cada valor recuperado. Un valor recuperado del $REindex$ representa una $RE$, puede ser un $URI$ o un $URL$ (véase las linea 4 del Algoritmo 1). En el primer caso, se dice que la identidad de la $RE$ es conocida. La instancia correspondiente es creada (línea 10 en el Algoritmo 3) con el identificador global y un

(hasta ahora) conjunto vacío de $DEs$. En el último caso, el $URL$ identifica a una $RE$ que no tiene identificador global y se asume que sólo hay una $DE$ que la describe (línea 18 en el Algoritmo 3).

**Algorithm 3** Manejador de Entidades Reales

```
1: function HANDLEREANSWER(reAnswer : REAnswer, REs : {⟨RE, relevance⟩})
2:     waitingTime : integer
3:     waitingTime := 5 ▷ Tiempo de espera parametrizable
4:     while reAnswer.isComplete = FALSE do
5:         WAITMS(waitingTime)          ▷ Especificado en milisegundos
6:     end while
7:     if reAnswer.reAnsValues ≠ NULL then
8:         for all reAnsValue ∈ reAnswer.reAnsValues do ▷ Hilos paralelos
9:             if ISURI(reAnsValue) then
10:                rEntity := ⟨reAnsValue,{}⟩
11:                if rEntity ∈ REs then
12:                    RELEVANCERE++(REs, rEntity)
13:                else
14:                    ADD(REs,⟨rEntity,1⟩)
15:                    HANDLEDEANSWER(GetDEindex(reAnsValue), REs)
16:                end if
17:            else
18:                rEntity := ⟨NULL,{reAnsValue}⟩
19:                if rEntity ∈ REs then
20:                    RELEVANCERE++(REs, rEntity)
21:                else
22:                    ADD(REs, ⟨rEntity,1⟩)
23:                end if
24:            end if
25:        end for
26:    end if
27: end function
```

En las líneas 11 y 19, comprobamos si la $RE$ ya está incluida en el conjunto de resultados. Si es así, llamamos a la función *relevanceRE++*, que incrementa el valor de la relevancia asociada a la $RE$. De lo contrario, añadimos la $RE$ al conjunto de resultados con un valor de relevancia inicial establecido en 1 (lineas 14 y 22). En este punto, si estamos en el caso de una $RE$ con identificador global (es decir, con un $URI$), el segundo paso de la búsqueda se inicia con la llamada a la función $GetDEindex$ del $DEindex$ (véase la línea 15). El objeto devuelto por dicha función se pasa a la función $HandleDEAnswer$, que se encarga de procesarlo.

Finalmente, el Algoritmo 4 muestra cómo se manejan los valores recuperados del $DEindex$. En primer lugar, esperamos hasta que se completa la respuesta (véase el bucle de la línea 4 a la línea 6) y luego los valores se utilizan para actualizar el conjunto de resultados. Hay que tener en cuenta que la función *addDE2RE* toma la clave (i.e., el $URI$) para identificar, dentro del conjunto de resultados, la $RE$ que tiene

**Algorithm 4** Manejador de Entidades Digitales

```
1: function HANDLEDEANSWER(deAnswer : DEAnswer, REs : {⟨RE, relevance⟩})
2:     waitingTime : integer
3:     waitingTime := 5
4:     while deAnswer.isComplete = FALSE do
5:         WAITMS(waitingTime)
6:     end while
7:     ADDDE2RE(REs, deAnswer.key, deAnswer.deAnsValues)
8: end function
```

que ser actualizada. Luego, los valores (i.e., los $URLs$) son asociados a tales $RE$ con el fin de completar la $RC$. Decimos que esta función (llamada en la linea 7 del Algoritmo 4) agrega $DEs$ a una determinada $RE$ de un conjunto dado.

## VII. Implementación y Evaluación

Implementamos el directorio distribuido sobre una red P2P, donde la distribución de los índices se realiza utilizando una tabla hash distribuida (en inglés, Distributed Hash Table, DHT) Las DHTs[5] permiten, a los nodos que participan en la red, almacenar y recuperar pares de clave y valor. En particular, usamos TomP2P[6], una librería de DHT avanzada que extiende las funcionalidades básicas de las DHTs. La librería soporta el almacenamiento de múltiples valores mapeados a la misma clave y la distinción de múltiple indices con diferentes dominios. Los distintos indices pueden ser vistos como diferentes DHTs, es decir, una para el $DEindex$ y otra para el $REindex$.

Estamos interesados en la evaluación del enfoque bajo condiciones de red realistas y queremos medir hasta qué punto el rendimiento disminuye cuando el tamaño de la red crece, es decir, la escalabilidad de nuestro enfoque. Dicho rendimiento se mide aquí en términos del tiempo que tarda el sistema para procesar una consulta. Usamos *PlanetLab*[7] como banco de pruebas, porque creemos que nos provee de las condiciones de red realistas que necesitamos. PlanetLab ofrece una red de computadoras, nodos, que se distribuyen en todo el mundo, se conectan entre sí a través de Internet y están disponibles para fines de investigación. Llevamos a cabo las evaluaciones en redes de 50, 100 y 150 pares; y usamos datos extraídos de las transacciones de la Conferencia Internacional Conjunta sobre Inteligencia Artificial[8] para generar los conjuntos de datos. En particular, utilizamos los títulos de publicaciones, los nombres de los autores y nombres de lugares relacionados con la conferencia.

Cada conjunto de datos se produce mediante la generación de triplas de $\langle Name, URI, URL\rangle$. Los nombres y los $URIs$ son replicados con el fin de simular diferentes $REs$ con el mismo nombre y diferentes pares que almacenan $DEs$

[5]http://en.wikipedia.org/wiki/Distributed_hash_table
[6]http://www.tomp2p.net/
[7]https://www.planet-lab.eu/
[8]IJCAI (siglas en inglés de *Joint Conference on Artificial Intelligence*) http://ijcai.org/

describiendo a la misma $RE$. Llamémosle $p_n$ a la popularidad de un nombre $n$ (i.e., al numero de $REs$ que son llamadas $n$) y $p_{we}$ a la popularidad de una $RE$ (i.e., al numero de $DEs$ que representan a la $RE$) En primer lugar, por cada nombre $n$, generamos $p_n$ cantidad de triplas con el mismo nombre (diferentes $URI$ y $URL$). En segundo lugar, para cada $URI$, generamos $p_{we}$ cantidad de triplas con el mismo nombre y $URI$ pero con diferentes $URLs$. Los valores de popularidad $p_n$ y $p_{we}$ siguen una distribución Zipf[9], lo que significa que hay una larga cola de nombres y $REs$ con baja popularidad. La distribución de ambos valores de popularidad son independientes, lo que significa que una $RE$ popular no necesariamente tiene un nombre popular y viceversa. Asumimos que la base de entidades locales de cada par contiene, en promedio, 2.000 $DEs$. Tenemos en total alrededor de 100.000, 200.000 y 300.000 $DEs$. El conjunto de consultas establecido para cada par se genera seleccionando aleatoriamente un conjunto de 1.400 nombres del conjunto inicial de nombres de entidades.

Durante la evaluación, primero indexamos el conjunto de datos de acuerdo al tamaño de red correspondiente y luego los pares comienzan el proceso de evaluación de búsqueda pseudo-simultáneamente. En este proceso, cada par realiza los siguientes pasos: (i) toma una consulta del conjunto de consultas, (ii) ejecuta la búsqueda, (iii) mide y registra el tiempo que toma al sistema responder a la consulta, (iv) espera un intervalo de tiempo aleatorio (entre 1 y 3 segundos), y (v) vuelva al paso (i). Estos pasos se repiten hasta llegar al final de la serie de consultas. Una vez que todos los compañeros terminan el proceso de búsqueda, se calcula el promedio (al nivel de toda la red) de tiempo que toma la resolución de una consulta. Mostramos los resultados para los diferentes tamaños de red en la tabla I. Como se puede observar, los valores de los tiempos medios de consulta son estables con el crecimiento de la red y creemos que este es un resultado prometedor en cuanto a la escalabilidad del directorio. Por otro lado, en comparación a los sistemas de recuperación de información (en general), los tiempos medios para la búsqueda todavía son altos.

TABLE I
TIEMPO DE CONSULTA PROMEDIO

| Tamaño de la red | 50 pares | 100 pares | 150 pares |
|---|---|---|---|
| Tiempo medio de consulta (en seg.) | 2.77 | 2.75 | 2.61 |

Con el fin de tener una mejor comprensión de los tiempos de consulta que contribuyen a estos promedios, se analiza la distribución del tiempo de consulta en las diferentes redes. En la Figura 2 se muestran los resultados de este análisis, donde podemos ver que también la distribución del tiempo de consulta es estable con respecto al crecimiento de la red. También en la Figura 2 podemos notar que más del 55% de las consultas son en realidad respondidas en menos de un segundo, mientras que en casi 70% de los casos la respuesta llega en menos de 2 segundos (que es menor que el tiempo promedio). Mas aún, sólo el 9% de las consultas toman más de 5 segundos.

[9]http://en.wikipedia.org/wiki/Zipf's_law

Es importante señalar que los resultados se retornan una vez que la respuesta a la consulta está completa, es decir, una vez que todos los hilos de ejecución implicados en la consulta han terminado. Esto significa que una sola búsqueda lenta es suficiente para retrasar el cálculo de una respuesta a una consulta y por lo tanto aumentar el tiempo de consulta. Asimismo, sabemos que los pares particularmente lentos pueden generar este tipo de problema cuando la búsqueda tiene que pasar a través de ellos. Creemos que, en el panorama general, la capacidad de ampliación del enfoque es un resultado prometedor e importante. Por otro lado, hay algunas técnicas que permiten guardar en caché resultados anteriores o evitar el enrutamiento a través de pares lentos (véase, por ejemplo [8]) que pueden ser implementados para reducir el efecto de los pares lentos en el tiempo de consulta.

## VIII. TRABAJOS RELACIONADOS

El enfoque que introducimos en este estudio está relacionado con los enfoques que se encargan de la gestión de información de entidades en una red P2P. Más específicamente, nuestro enfoque se ocupa de la indexar y buscar de forma distribuida a las entidades en función de sus identificadores. Hasta donde sabemos, no existen enfoques que se encargan de la integración de dichas áreas, es decir, que se centran en la búsqueda de entidades en una red P2P. Existen sin embargo, algunos enfoques relacionados dentro de ambas áreas y en esta sección analizamos aquellos que consideramos más relevantes.

Por un lado encontramos algunos enfoques que concentran la atención en la definición de modelos y estructuras para la representación de las entidades [1]. Por otro lado, en [7] se propone un sistema de gestión de nombres de entidades (ENS - siglas en inglés de *entity name system*) con el fin de brindar apoyo a la generación y re-utilización de identificadores de entidades globales únicos, a través de diferentes repositorios RDF[10]. En este enfoque, el repositorio local de un simple usuario no se considera como una fuente de datos y los usuarios necesitan un permiso de acceso especial a fin de contribuir a la definición de las entidades. Como un primer paso para hacer frente a la búsqueda, el trabajo presentado en [9] propone un modelo que analiza la especificación de la consulta y realiza la desambiguación del tipo de entidad deseado. En [10], se distinguen aquellas entidades que tienen un nombre, las cuales se extraen mediante el análisis de consultas basado en la coincidencia sintáctica de patrones. Estos enfoques no abordan directamente la búsqueda, pero sus resultados son relevantes para la definición del directorio propuesto en este estudio.

Otros enfoques que se ocupan de la búsqueda siguiendo una perspectiva centrada en entidades se pueden encontrar en la literatura [11], [12], [13]. En [11], [13], se proponen motores de búsqueda de entidades; en [12] se usan reglas heurísticas para identificar entidades dentro de una colección de documentos; y un servicio para encontrar documentos que contienen

[10]Marco de Descripción de Recursos, RDF –siglas en inglés de *Resource Description Framework*– http://es.wikipedia.org/wiki/Resource_Description_Framework

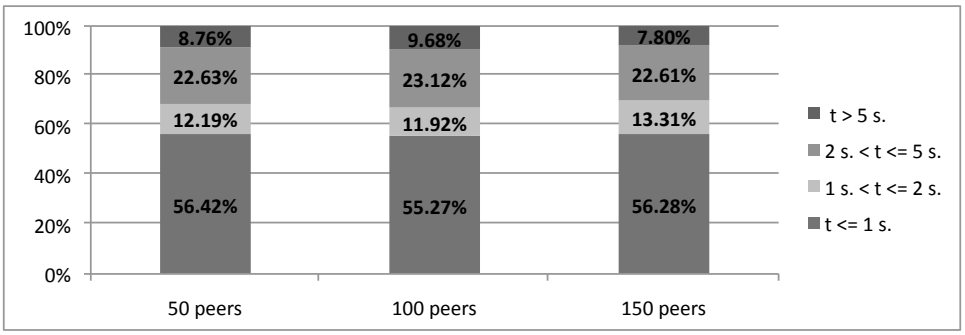


Fig. 2. Tiempo de consulta en las diferentes redes

declaraciones sobre recursos particulares se proporciona en Sindice [14]. La mayoría de estos enfoques recogen datos de múltiples fuentes en la web pero no tienen en cuenta la distribución a nivel de simples usuarios individuales (es decir, una red P2P). En particular, [13] agrega automáticamente las descripciones de diferentes fuentes y permite la posterior navegación a través de entidades vinculadas. La distribución se considera en términos de grupos de equipos que permiten el procesamiento en paralelo y el almacenamiento escalable pero la búsqueda se mantiene centralizada (i.e., los índices son construidos de forma centralizada). A diferencia de estos enfoques, nuestro enfoque realiza una búsqueda distribuida en una red P2P y permite a los usuarios mantener sus datos localmente.

Existen también enfoques P2P, que se ocupan de la búsqueda distribuida, pero no están basados en entidades [15], [16]. Estos se clasifican principalmente como enfoques estructurados y no estructurados. Las primeras redes no estructuradas (por ejemplo., Gnutella[11]) tienen problemas de escalabilidad debido al número de mensajes generados y no pueden garantizar que todos los posibles resultados serán encontrados. Otros enfoques utilizan técnicas de agrupamiento [17], [18], [19], [20], [21], su objetivo es encontrar el mejor grupo para responder a una consulta y luego enviar la consulta a los pares dentro de dicho grupo. Nuestro enfoque puede encontrar todas las respuestas disponibles y ha mostrado resultados prometedores en términos de escalabilidad.

Podemos encontrar también enfoques más estructurados que tienen como objetivo garantizar la localización del contenido compartido en la red (por ejemplo, CAN [22], Chord [23] Pastry [24] y Tapestry [25]). Ellos almacenan pares de $\langle clave, valor \rangle$ en una DHT (tabla hash distribuida), permitiendo luego la recuperación de un valor asociado con una clave dada. Otros enfoques permiten la búsqueda basada en múltiples palabras claves usando DHTs pero esto puede ser muy costoso en términos del almacenamiento necesario y el tráfico generado (por ejemplo [26]). En algunos casos se combina el uso de estructuras jerárquicas con técnicas de agrupamiento usando la estructura de DHTs [27], [28], [29], [30]. En general, los enfoques P2P proveen las técnicas necesarias para construir la solución que proponemos. La novedad de nuestro enfoque está en el dominio de aplicación de tales técnicas.

[11]http://en.wikipedia.org/wiki/Gnutella

## IX. Conclusiones

En este documento hemos presentado un directorio distribuido de entidades que introduce las nociones de $DE$ (Entidad Digital) y $RE$ (Entidad Real) con el fin de vincular los directorios locales de diferentes pares. El directorio proporciona servicios de búsqueda basados en identificadores de entidades. En particular, hemos presentado los algoritmos para la búsqueda de entidades basada en nombres. Hemos discutido el problema de coincidencia de nombres que aparece como consecuencia de la relación *muchos-a-muchos* entre los nombres y las $REs$. Luego, hemos mostramos que por su diseño, nuestro directorio se ocupa del problema de la búsqueda de nombres dentro de la red (es decir, la primera parte del problema de coincidencia de nombre).

Los datos de los compañeros se almacenan localmente, sólo los identificadores y los enlaces a las representaciones locales están indexados. Esta infraestructura permite la implementación de mecanismos de control de acceso en las representaciones locales con el fin de hacer frente a los problemas de privacidad. Al mismo tiempo, los cambios realizados por los pares en sus representaciones locales, están disponibles en el directorio de manera directa. Los índices se distribuyen utilizando una tabla hash distribuida (DHT), pero la definición del directorio es independiente de la implementación de DHT subyacente.

La evaluación de la búsqueda se realizó en redes de 50, 100, y 150 pares, utilizando PlanetLab. Hemos presentado el promedio de tiempo de consulta (como una medida del rendimiento) para diferentes tamaños de red, así como la distribución de los tiempos de consulta. Los resultados pueden ser considerados prometedores en términos de escalabilidad debido a que el rendimiento se mantiene estable con el crecimiento de la red.

Como parte de los trabajos futuros, estamos interesados en la integración de enfoques que hagan frente al problema de coincidencia de la consulta con los nombres usados en la red (i.e., la segunda parte del problema de coincidencia de nombres). Adicionalmente, queremos entender mejor los

diferentes elementos que influyen en el rendimiento de la búsqueda de forma a encontrar y poner en práctica técnicas que ayuden a reducir los tiempos de consulta.

## REFERENCES


[1] B. Bazzanella, J. A. Chaudhry, Themis Palpanas, and H. Stoermer, "Towards a General Entity Representation Model," *5th Workshop on SWAP*, 2008. [Online]. Available: http://disi.unitn.it/~themis/publications/swap08.pdf

[2] G. Holloway and M. Dunkerley, *The Math, Myth and Magic of Name Search and Matching*, 5th ed. Search Software America, 2004.

[3] F. Giunchiglia and A. Hume, "A distributed entity directory," in *The Semantic Web: ESWC 2013 Satellite Events*, ser. Lecture Notes in Computer Science, P. Cimiano, M. Fernández, V. Lopez, S. Schlobach, and J. Völker, Eds. Springer Berlin Heidelberg, 2013, vol. 7955, pp. 291–292. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-41242-4_47

[4] F. Giunchiglia, R. Zhang, and B. Crispo, "Relbac: Relation based access control," in *Proceedings of the 2008 Fourth International Conference on Semantics, Knowledge and Grid*, ser. SKG '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 3–11. [Online]. Available: http://dx.doi.org/10.1109/SKG.2008.76

[5] E. Bignotti, "Semantic name matching," Master's thesis, University of Trento, 2012.

[6] A. Hogan, A. Zimmermann, J. Umbrich, A. Polleres, and S. Decker, "Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora," *JWS: Science, Services and Agents on the World Wide Web*, vol. 10, 2012. [Online]. Available: http://www.websemanticsjournal.org/index.php/ps/article/view/224

[7] P. Bouquet, H. Stoermer, C. Niederee, and A. Maña, "Entity name system: The back-bone of an open and scalable web of data," in *Proceedings of the 2nd IEEE ICSC*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 554–561. [Online]. Available: http://portal.acm.org/citation.cfm?id=1446294.1446425

[8] S. Rhea, B.-G. Chun, J. Kubiatowicz, and S. Shenker, "Fixing the embarrassing slowness of opendht on planetlab," in *Proc. of the 2nd conference on Real, Large Distributed Systems*, ser. WORLDS'05, Berkeley, CA, USA, 2005, pp. 25–30. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251522.1251527

[9] B. Bazzanella, H. Stoermer, and P. Bouquet, "Searching for individual entities: a query analysis," University of Trento, Tech. Rep., 2009.

[10] M. Paşca, "Weakly-supervised discovery of named entities using web search queries," in *Proceedings of the sixteenth ACM conference on CIKM '07*. New York, NY, USA: ACM, 2007, pp. 683–690. [Online]. Available: http://doi.acm.org/10.1145/1321440.1321536

[11] T. Cheng and K. C.-C. Chang, "Entity search engine: Towards agile best-effort information integration over the web," in *CIDR 2007*, 2007, pp. 108–113.

[12] G. Hu, J. Liu, H. Li, Y. Cao, J.-Y. Nie, and J. Gao, "A supervised learning approach to entity search," in *AIRS'06*, ser. LNCS, 2006, vol. 4182, pp. 54–66. [Online]. Available: http://dx.doi.org/10.1007/11880592_5

[13] A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, and S. Decker, "Searching and browsing linked data with swse: The semantic web search engine," *JWS: Science, Services and Agents on the World Wide Web*, vol. 9, no. 4, pp. 365 – 401, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570826811000473

[14] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello, "Sindice. com: a document-oriented lookup index for open linked data," *International Journal of Metadata, Semantics and Ontologies*, vol. 3, no. 1, pp. 37–52, 2008.

[15] J. Risson and T. Moors, "Survey of research towards robust peer-to-peer networks: Search methods," *Computer Networks*, vol. 50, pp. 3485–3521, 2006. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2006.02.001

[16] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *IEEE Communications Surveys and Tutorials*, vol. 7, pp. 72–93, 2005.

[17] M. Bawa, G. Manku, and P. Raghavan, "Sets: Search enhanced by topic segmentation," in *Proceedings of ACM SIGIR Conference*, 2003, pp. 306–313.

[18] E. Cohen, H. Kaplan, and A. Fiat, "Associative search in peer to peer networks: Harnessing latent semantics," in *Proceedings of IEEE INFOCOM*, 2003.

[19] K. Spripanidkulchai, B. Maggs, and H. Zhang, "Efficient content location using interest-based locality in peer-to-peer systems," in *Proceedings of IEEE INFOCOM*, vol. 3, 2003, pp. 2166–2176.

[20] A. Crespo and H. Garcia-Molina, "Semantic overlay networks for p2p systems," Stanford University, Tech. Rep., 2002.

[21] S. Joseph, "Neurogrid: Semantically routing queries in peer-to-peer networks," in *Proc. Intl. Workshop on Peer-to-Peer Computing*, 2002, pp. 202–214.

[22] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proc. of SIGCOMM'01*. NY, USA: ACM, 2001, pp. 161–172. [Online]. Available: http://doi.acm.org/10.1145/383059.383072

[23] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proc. of SIGCOMM'01*. NY, USA: ACM, 2001, pp. 149–160.

[24] P. Druschel and A. Rowstron, "Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proc.of ACM SIGCOM*, 2001.

[25] B. Zhao, L. Huang, J. Stribling, S. Rhea, a.D. Joseph, and J. Kubiatowicz, "Tapestry: A Resilient Global-Scale Overlay for Service Deployment," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 41–53, Jan. 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1258114

[26] J. Li, B. Thau, L. Joseph, M. Hellerstein, and M. F. Kaashoek, "On the feasibility of peer-to-peer web indexing and search," in *IPTPS'03*, 2003.

[27] P. Ganesan, K. Gummadi, and H. Garcia-Molina, "Canon in g major: designing dhts with hierarchical structure," in *ICDCS'04*, 2004, pp. 263 – 272.

[28] D. Janakiram, F. Giunchiglia, H. Haridas, and U. Kharkevich, "Two-layered architecture for peer-to-peer concept search," in *4th Int. Sem Search Workshop*, 2011.

[29] O. Papapetrou, W. Siberski, and W. Nejdl, "Pcir: Combining dhts and peer clusters for efficient full-text p2p indexing," *Computer Networks*, vol. 54, no. 12, pp. 2019–2040, 2010.

[30] L. Garcés-Erice, E. W. Biersack, P. Felber, K. W. Ross, and G. Urvoy-Keller, "Hierarchical peer-to-peer systems," in *Euro-Par*, 2003, pp. 1230–1239.