Informe
Proyecto

Recuperación de Información y Recomendaciones en la Web

2018

Grupo 1

Juan Carriquiry

Giovani Rondán

Joaquín Scocozza

1. Introducción

El presente documento tiene como objetivo presentar el proyecto realizado para la asignatura Recuperación de Información y Recomendaciones en la Web.

El mismo consiste en la implementación de una solución alternativa a la búsqueda de viajes económicos o de corta duración, para distintas ciudades establecidas en el sistema.

2. Descripción del Problema

El problema a resolver consiste en la obtención de los conjuntos de viajes más económicos para viajar desde un origen dado hacia un determinado destino, combinando diferentes medios de transporte. Para ello, además de la implementación de un sistema que implemente esta solución, es necesario lograr una obtención de datos válidos, para los posibles medios de transporte a ofrecer.

La motivación de esta problemática se fundamenta en el hecho de que en la actualidad no existe un sistema similar, que permita combinar medios de transporte y que funcione con una cantidad considerable de empresas para los diferentes medios de transporte.

Además, es de interés para la solución resultante, que el sistema brinde resultados en un tiempo de respuesta razonable. Es decir, en un máximo de 4 o 5 segundos, que es en promedio del tiempo de respuesta que ofrecen los sitios similares para este tipo de consultas.

3. Enfoque de la solución

3.1. Aplicación Web

Para la solución del presente proyecto, se optó por implementar una aplicación web, donde los usuarios del sistema (en principio sin requerir registro previo), puedan acceder al sitio web y consultar por los viajes que deseen.

En la solución realizada, el sistema ofrece una pantalla principal, donde los usuarios pueden seleccionar un origen y destino (dentro de las opciones), la fecha en la que deseen viajar y agregar filtros extra a su búsqueda.

3.2 Módulo para la Carga de Datos

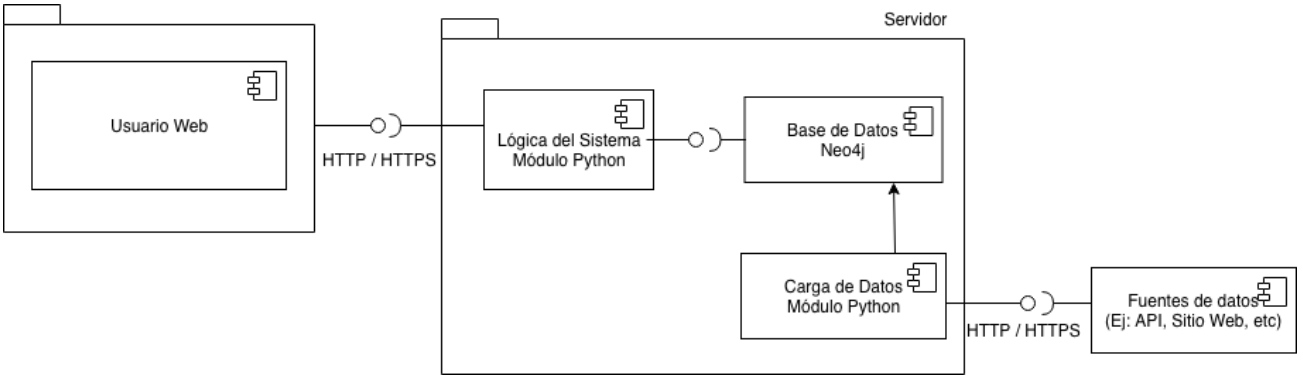
Además, la solución que se llevó a cabo, brinda un módulo extra para la carga de datos, donde el administrador del sistema puede configurar las diferentes fuentes de información para la obtención de los datos de viajes.

En principio, este módulo ofrece la carga de datos para viajes de ómnibus para ciudades de Uruguay (Web scraping) y para viajes entre las diferentes ciudades de Uruguay utilizando Uber como transporte (API de Uber).

Como configuración adicional, el administrador puede configurar también los distintos orígenes y destinos manejados por la aplicación.

4. Diseño

El sistema en cuestión se encuentra organizado de la siguiente manera:



Donde se destacan:

- ❏ Una interfaz de usuario (indicada por usuario web).
- ❏ Un componente representando el servidor, que a su vez se compone de la base de datos utilizada (Neo4j), la lógica del sistema (módulo python para la comunicación con la interfaz de usuario) y un módulo adicional para cargar los datos en la base de datos (módulo python).
- ❏ Finalmente, como parte del sistema, se indican también las Fuentes de Datos (por ejemplo sitios web), ya que los mismos forman parte esencial en el sistema, brindando la información de los viajes a ofrecer.

5. Implementación

5.1 - Aplicación web y servidor

Para la implementación del sistema, se utilizó como base el lenguaje de programación Python [1], el cual permite la implementación de la lógica por parte del servidor y posee diferentes librerías que facilitan la implementación de los demás módulos del sistema.

Para la comunicación de la interfaz de usuario con el servidor, se optó por utilizar la librería de Python Bottle [2], la cual se define como un micro framework rápido y liviano para el desarrollo web con Python. Desde la interfaz de usuario, alcanza con enviar el pedido de búsqueda al servidor, con los parámetros deseados, y manejar la respuesta utilizando Bottle.

5.2 - Persistencia

Para la persistencia de la información de los distintos viajes y posibles ciudades de origen/destino, se optó por utilizar la base de datos Neo4j [3], la cual se caracteriza por ser una base de datos no relacional, orientada a grafos, cuyo potencial se aprecia principalmente en este tipo de realidades (grafos), donde las consultas se realizan sobre aristas ponderadas (viajes con precio y duración) para determinados nodos (origen y destino dados).

En particular, Neo4j se ofrecía como una alternativa muy interesante para los integrantes del equipo, por ser una tecnología diferente a las comúnmente utilizadas para proyectos de facultad y por su potencial para proyectos de gran porte, donde deben manejarse cantidades muy grandes de datos.

5.3 - Carga de Datos

Para la obtención de los datos, el módulo de carga implementado ofrece dos posibilidades:

1. Web scraping

   En particular, para la carga de viajes de ómnibus, se decidió utilizar la técnica de web scraping, implementada con la librería de Python BeautifulSoup4 [4].
   Dicha librería ofrece la posibilidad de extraer datos desde distintos sitios webs, de manera sencilla y performante. Esta funcionalidad fue implementada en un módulo especial del servidor que permite configurar más sitios web y parámetros para los mismos.

   Como fuente de información se obtuvieron los diferentes viajes del sitio web de Urubus [5], el cual posee datos para todas las combinaciones de ómnibus del Uruguay, lo cual nos permitió centralizar el problema de la obtención de datos a un solo sitio web.

2. API Uber

   Como medio de transporte alternativo a los ómnibus, se obtuvo información de viajes de Uber para las distintas ciudades del sistema, mediante la API [6] que ofrece su sitio para desarrolladores.

   En particular se utilizó el método *'get_price_estimates()'*, que ofrece una estimación para el precio del viaje solicitado (origen y destino se ingresan indicando latitud y longitud de los mismos).

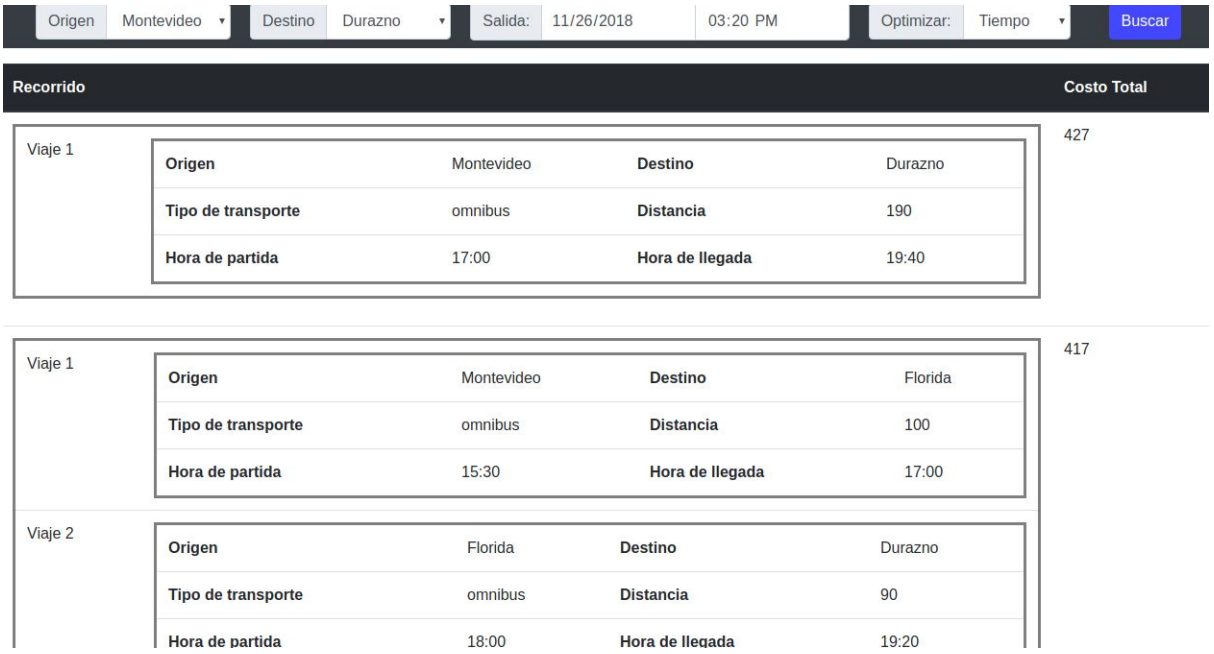
El módulo para la carga de datos, cuenta con un componente principal *'load_data.py',* que se encarga de invocar los métodos para la carga de viajes de ómnibus y de uber, y guardarlos localmente como archivo .json.
Una vez se obtienen los archivos .json con la información de los viajes, el sistema puede cargarlos en la base de datos (utilizando un script para la carga) y así tenerlos en cuenta como respuesta a las consultas de los usuario, este script toma la información de los json generados previamente, lo cual le da la posibilidad al sistema de en un futuro ingresar nuevos datos provisto por otro sistema con el simple requerimiento que respete el formato del json.

6. Funcionalidades y uso

En el sistema se destacan dos grandes funcionalidades, dirigidas a dos clases de usuario diferentes.

La primera y más importante, es la posibilidad de buscar entre las diferentes combinaciones de origen y destino, la combinación de viajes que permite llegar al destino de la manera más económica posible.

Además, dentro de esta funcionalidad se permite optar por el tipo de resultado, es decir, se permite elegir si se desea que los viajes resultantes sean para la combinación más económica para llegar al destino elegido, ó si los viajes deben ofrecer la combinación con menor duración para llegar a destino.



Como segunda funcionalidad, el sistema ofrece el módulo de carga de datos, el cual se destaca por ser configurable para cargar datos de viajes para otras páginas web (utilizando web scraping con BeautifulSoup4) o desde otro método de carga a configurar.

La carga de datos se encuentra estructurada por medio de transporte (actualmente bus y uber), lo cual le permite escalar fácilmente a otros medios de transporte.

Cabe destacar que la funcionalidad en cuestión, es controlada por el módulo load_data.py que se encarga de invocar los métodos de carga para los diferentes medios de transportes y persistirlos localmente como archivo .json.

7. Evaluación y resultados

Para corroborar la correctitud del sistema se ejecutaron casos de prueba sobre los distintos componentes que conforman el mismo. A continuación se detalla las características de las pruebas realizadas.

7.1 Verificación de datos obtenidos

Debido a las características del sistema, es necesario verificar que los datos obtenidos desde las diferentes fuentes se correspondan con la realidad.

Datos de ómnibus

Para los datos de los ómnibus en primer lugar se verificó que los datos pertenecientes a la página Urubus se correspondan a los datos de las agencias que proveen el servicio.
Esto se realizó de forma manual consultando para un subgrupo de las agencias de viajes, si los datos que se obtienen al realizar una consulta sobre la página de Urubus son los mismos que en sus propias páginas.

Se llegó al resultado de que para todas las consultas realizadas, los datos (horarios de los viajes y el precio asociado) de Urubus relevantes para el sistema desarrollado son correctos.
A su vez se verificó que los datos recabados por el sistema desde la página Urubus sean válidos. Para esto se compararon los datos procesados por el script de carga con los de Urubus y al igual que en el caso anterior se concluyó que los mismos son correctos.

Datos de Uber

Como se describió en las secciones anteriores, los datos del servicio de viajes Uber fueron obtenidos por medio de la API oficial de la empresa. Esto simplifica las tareas de verificación dado que la fuente de datos es confiable.
Para corroborar el correcto funcionamiento de la API, se compararon los resultados obtenidos consultando a la aplicación Uber para dispositivos Android.

7.2 Verificación del algoritmo de búsqueda

Debido al uso de la base Neo4j, la verificación sobre las combinaciones de viajes retornados por el sistema se reduce a comprobar el funcionamiento de la consulta ejecutada para obtener los viajes.
Para esto se diseñó una base de datos de prueba reducida conformada por 5 nodos y 3 aristas entre cada nodo, y se calculó de forma manual la serie de caminos viables de menor costos entre los distintos nodos. Luego se compararon estos caminos con los resultados obtenidos por la consulta que ejecuta el sistema para obtener la combinación de viajes entre dos ciudades.

7.3 Datos obtenidos

Se detalla en la **Tabla 1** la cantidad de datos almacenados en el sistema.

Descripción	Cantidad	Atributos
Ciudades	19	-Id -Nombre -Latitud -Longitud
Viajes de Uber	30	-Id Ciudad origen -Ciudad destino -Precio -Duración
Viajes de ómnibus	2279	-Id -Ciudad origen -Ciudad destino -Precio -Fecha salida -Fecha llegada -Hora salida -Hora llegada

Tabla 1: Cantidad de datos almacenados

Se almacenaron 19 ciudades en el sistema que se corresponden a las 19 capitales departamentales de Uruguay, estas ciudades se guardaron como nodos en la base Neo4j.

A su vez se obtuvieron datos de viajes para 30 combinaciones de ciudades para el medio de transporte Uber. Es importante destacar que tanto la aplicación para dispositivos móviles como la API tiene una restricción por la que solo permite consultar por viajes con distancia menores a 100 millas. El número de viajes también se ve reducido debido a que Uber no cuenta con servicio para todas las ciudades que forma parte del sistema.

La mayor cantidad de viajes pertenecen a los datos obtenidos para los ómnibus, esto se debe a que para este medio de transporte es necesario almacenar todas las frecuencias horarias disponibles entre dos ciudades.

8. Conclusiones

Al analizar el trabajo realizado se resalta el potencial del sistema implementado para facilitarle al usuario final la planificación de sus viajes combinando diferentes medios de transporte.

Se cumplió con el objetivo preestablecido de recabar datos de las distintas agencias de ómnibus de Uruguay consolidando una base de datos de gran valor para su explotación.

Por otra parte el número de viajes recabados para Uber es inferior al planificado, debido a que la empresa Uber no brinda el servicio en todas las ciudades pertenecientes al sistema. Se prevé que esta limitante no existiría en otras zonas geográficas desde las que se pueden recabar datos, como Estados Unidos donde Uber brinda servicios en la mayor parte de su territorio.

Se demostró que es factible implementar un sistema capaz de soportar diversos medios de transporte y un número superior de ciudades al presentado en el sistema desarrollado, esto se deduce de la performance del sistema ante la cantidad de datos almacenados en esta versión.

Por otra parte se considera que al incorporar otros medios de transporte, como pueden ser aviones, barcos y trenes, y destinos de otras ciudades del mundo; el sistema pasaría a ser más atractivo para el usuario final, facilitando la planificación de viajes al exterior ya sea por trabajo o vacaciones.

Se considera que el proyecto realizado, tiene como ventaja principal la posibilidad de escalar. Es decir, eventualmente podrían añadir más ciudades o crecer en cuanto a fuentes de información y/o métodos de extracción de datos.

A pesar de cumplirse con la realización del sistema en términos de requisitos, se considera que esta primer versión del sistema no ofrece grandes novedades respecto a otros sistemas existentes, ya que cuenta con pocas variantes en cuanto a medios de transporte y viajes disponibles.

9. Trabajo Futuro

Dadas las características del sistema desarrollado es posible planificar nuevas versiones del mismo que incorporen nuevas funcionalidades.

El primer punto a destacar es la posibilidad de incrementar los datos almacenados por el sistema para permitirle al usuario final una mayor selección de destinos y medios de transporte.

Por otra parte resulta interesante analizar la posibilidad de incorporar nuevos métodos de búsqueda a los dos permitidos actualmente, a modo de ejemplo se puede incorporar la opción de seleccionar las combinaciones de viajes más económicas que no sobrepasen una medida de tiempo ingresada por el usuario, o las combinaciones de viajes que mejor balance tengan entre el costo asociado y la duración.

También es posible implementar una solución basada en los sistemas de información geográfica que permita la incorporación de un mapa donde el usuario pueda marcar cuales son el origen y el destino del viaje y el sistema represente en el mapa la trayectoria de las diferentes combinaciones.

Otra mejora a realizar sería permitir consultas que consideren múltiples destinos, obteniendo como respuesta la mejor forma de recorrer todas las ciudades a visitar en función de las preferencias ingresadas por el usuario. A su vez se puede fijar el orden de visita de las ciudades y el tiempo de estadía en cada ciudad para obtener como resultado la sucesión de viajes a realizar para visitar todos los destinos preestablecidos.

10. Referencias

A continuación se presentan las referencias utilizadas a lo largo del proyecto:

[1] - Python: https://docs.python.org/2/
[2] - Bottle: https://bottlepy.org/docs/dev/
[3] - Neo4j: https://neo4j.com/
[4] - BeautifulSoup4: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
[5] - Urubus: https://www.urubus.com.uy/es/?lang=es
[6] - API Uber: https://developer.uber.com/docs