The ComeWithMe System for Searching and Ranking Activity-Based Carpooling Rides

Vinicius Monteiro de Lira
CIN-UFPE, Recife, Brazil
vcml@cin.ufpe.br
ISTI-CNR, Pisa, Italy
lira@isti.cnr.it

Chiara Renso
ISTI-CNR, Pisa, Italy
renso@isti.cnr.it

Raffaele Perego
ISTI-CNR, Pisa, Italy
perego@isti.cnr.it

Salvatore Rinzivillo
ISTI-CNR, Pisa, Italy
rinzivillo@isti.cnr.it

Valeria Cesario Times
CIN–UFPE, Recife, Brazil
vct@cin.ufpe.br

ABSTRACT

ComeWithMe is an activity oriented carpooling service that enlarges the candidate destinations of a ride request by considering alternative places where the desired activity can be performed. It is based on the observation that individuals often move towards a place to perform an activity while the activity is often not strictly associated with a single place, as one may go for shopping or eating to many different locations. Activity-oriented carpooling hugely increases the number of rides matching a query, thus introducing requirements on system responsiveness and ranking effectiveness that are not common to traditional carpooling services. The demoed system implements the ComeWithMe service in almost its entirety, and includes the back-end and a user-friendly mobile application for smartphones aimed at achieving users' acceptance and usability.

Keywords

Carpooling; rides search; ranking

1. INTRODUCTION

The increasing of the vehicle fleet in major cities tends to worsen traffic congestion that in turn has negative effects on the environment due to the emission of pollutants. Traffic congestion becomes a factor of decision in the people's lifestyle: when planning their daily activities people consider not only the destination location, the time and the transportation facilities, but also the possible delay and stress due to traffic jams. Increasingly, people give priority to the traffic issue at the expenses of their preferences, sometimes changing the intended destination and/or the travelling time depending on the traffic forecasts.

On the other hand, economic development in large cities has increased the offers of venues where to carry on activities, allowing the users to choose among several alternative locations. For example, a person may choose among many different restaurants where to go for lunch, or different supermarkets where to go shopping. Additionally, each individual has a personal attitude to discover and try many different alternative locations [2]. ComeWithMe[1] exploits these attitudes offering a carpooling service that intercepts the willingness of users to delay/anticipate their movements and possibly change their final destination.

Previous work showed that ComeWithMe can boost the ride possibilities up to 80% [3, 5], thus providing a sustainable way to use the fleet of cars that performs daily routines in a city. A direct consequence of this huge increase is that when we apply this method to large cities with many users, each carpooling request can match hundreds of possible rides offers to several destinations at different temporal slots. It is clear that effectively ranking these offers is crucial for the passenger acceptance and the success of the system.

This paper presents a system to efficiently collect, index, retrieve and rank the carpooling rides offered by users. The demo presents the main features of the ComeWithMe prototype that manages both drivers (offering rides) and passengers (requesting rides), and matches the requests with offers with the objective of maximizing the passenger acceptance rate. The system we present tackles the scalability issue by exploiting an inverted index that optimizes query processing, while usability is achieved by exploiting the knowledge mined from drivers trajectory data collected from the smart-phone sensors, and by providing a user-friendly mobile interface running on smart-phones.

We show the ComeWithMe system for the driver and passenger profiles. The driver offers rides based on scheduled trips or the analysis of his travel history (routine trips) and also receives the requests of rides from the passengers. The passengers query the system by selecting the desired destination, the time and by setting the flexibility preferences. The system firstly expands the query to get alternative destinations while preserving the intended activity, then searches the possible rides offered by drivers and finally ranks the retrieved rides based on passengers preferences. From here, the user can select a ride and visualize its details, check the route on the map and send the request to the driver.

Carpooling is quite used today and many systems are available to users. We can recall Blablacar.com, CarpoolWorld.com, Djengo.be and Carpooling.com. These systems have all similar functionalities, such as the search for commuting rides (travelling between two cities) and regular rides

[1] http://comewithme.isti.cnr.it

SIGIR '16, July 17-21, 2016, Pisa, Italy
© 2016 ACM. ISBN 978-1-4503-4069-4/16/07...$15.00
DOI: http://dx.doi.org/10.1145/2911451.2911459

(home to work). Part of them also provide search for rides for groups of users (groups in school, events or companies). In summary, almost all of them can be seen as complementary to ComeWithMe since they focus on fixed destinations, while our system is based on the idea that some destinations are flexible.

ComeWithMe has been introduced in [3] where an algorithm for the spatio-temporal reallocation has been proposed to test the system potential. This demo paper presents the implementation of the whole prototype system, based on scalable and efficient information retrieval techniques.

2. SYSTEM ARCHITECTURE

ComeWithMe is designed to answer efficiently and effectively activity-oriented carpooling queries.

Efficiency and scalability are needed since the possibility of reallocating space and time boosts the rides opportunities at a high rate. For example, the activity "eating Italian food" may result in hundreds of different destinations in a large city, which raises to thousands with a less specific activity is chosen, e.g., "eat at a restaurant". In general, the number of possible rides tends to exponentially increase with the generality of the activity performed. The additional flexibility in the management of the pickup-up points and the temporal constraints further increments the complexity of query answering.

Besides expanding the user query on the basis of the intended activity and then retrieving all possible rides, ComeWithMe ranks the candidate rides according to the user's context and preferences. A number of features affect ranking such as: user current location and schedule, her willingness to walk to reach the pick-up or destination points, her adaptability to anticipate or delay the preferred departure and arrival time. The personalization of ranking criteria is important to improve the quality of experience perceived by the user and increase the probability of the ride acceptance.

The architecture of ComeWithMe is illustrated in Figure 1. As we can see, it is organized into three main components detailed in the following subsections: the *Trip Manager* to handle trips data, the *Ride Search Engine* to answer activity-oriented queries, and the *Mobile app* to interact with the users.

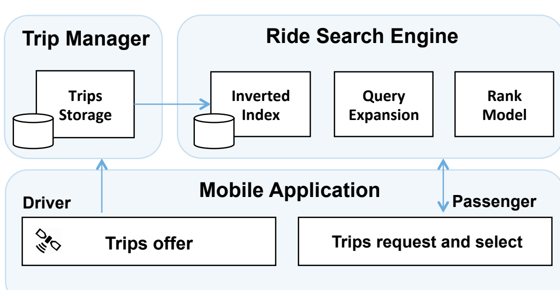

Figure 1: The architecture of the ComeWithMe system with three main components.

2.1 Trip Manager

This module is in charge of managing the drivers mobility data to model the ride offers. The region of interest covered by the carpooling service is simply modeled with a 2D spatial grid, and the path followed by each trip represented by the time-ordered sequence of cells visited [3]. The resolution of the grid is tuned to balance the number of activities per cell with the spatial accuracy of the rides.

The trips are associated with metadata specifying: the unique identifiers of the driver (*driverid*) and the trip (*tripid*), the date and the time t at which the trip will start, and the *ordered sequence* of cells visited along with their visiting time. Furthermore, each trip is annotated with additional information stating if the trip is routinely, and, if so, with its repetition pattern *(e.g., working days, week-end, Mondays, etc).* An example of the conversion from the geospatial representation of the trip is exemplified in Figure 2.

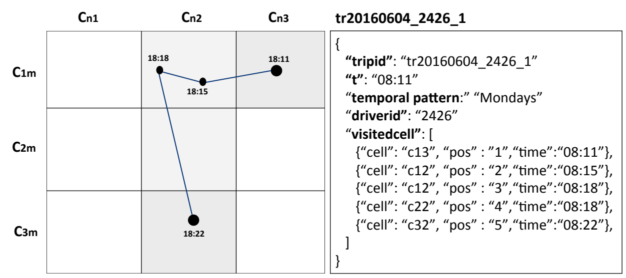

Figure 2: Example of trip representation.

ComeWithMe currently supports two different sources of trips: *routine trips* mined from historical GPS data collected by the ComeWithMe system, and *scheduled trips* explicitly offered by users through the mobile application.

Routine trips. ComeWithMe mobile app allows users to collect personal GPS data on a volunteer basis. These data are sent to the trip manager and stored in a spatial database. By applying the clustering method proposed in [4], we infer the drivers routine trips that are suggested as carpooling opportunities. The drivers have the option to decide whether to offer these rides to users.

Scheduled trips. These trips are made explicitly available by the user through the ComeWithMe mobile application specifying the path and the time of a specific planned trip she wants to share.

Additional functionalities provided by the Trip Manager module include the interactions with the mobile application, the requests of deletion of previously published trips, and the update of routine trips on a daily basis.

2.2 The Ride Search Engine

The main task of the *ride search engine* is to answer ComeWithMe passengers' queries by providing lists of ride opportunities ranked according to the user context and preferences. This task is accomplished by mean of three submodules: *Query Expansion*, *Inverted Index* and *Ranking Model*.

Query Expansion. This module boosts the possibilities of car rides by exploiting a query expansion technique. The use of query expansion generally increases recall and it is widely adopted in many application fields [1]. The queries in our context are carpooling requests expressing the passenger's intention to move to a venue to perform an activity. Given the destination *Point of Interest* (PoI) specified by the passenger, the query is automatically expanded with places related to the same activity by using a hierarchical thesaurus (an example is shown in Figure 3). The specific PoIs are the narrowest terms, while the intermediate layers represent different activities abstraction levels and thus possible query generalizations. For example, looking at Figure 3, when a passenger requests as destination "Da Gino", we see that it is an Italian Restaurant and expanding the query over Italian restaurants we have "Ristorante Giannino" as an alternative destination. Abstracting again up to "Eating" we have all the venues where they serve food corresponding to "Pizzeria", "Japanese restaurants", etc. The more we expand the query to broader terms, the more rides possibilities the passenger can select from the driver offers.

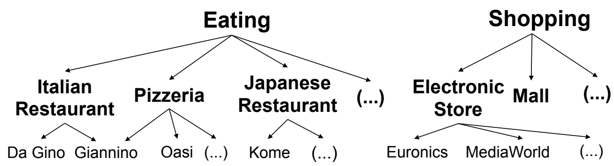

Figure 3: Structure of the thesaurus.

Each venue in the thesaurus is associated with a cell of the spatial grid indicating its location. Analogously, user queries are coded with the cells representing the pick-up area, the destination place, and a set of other cells representing alternative destinations. An example of the expansion process is illustrated in Figure 4: the destination PoI "Da Gino" is expanded with other possible venues (and cells) where the passenger can perform the activity "Eating".

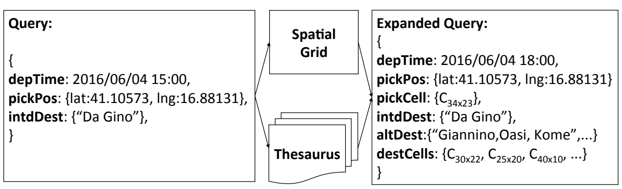

Figure 4: A query expansion example having Italian restaurant "Da Gino " as destination place is expanded into a list of cell containing alternative places for eating.

Inverted Index. Given the pick-up and the destination cells, and a time-window, ComeWithMe exploits an inverted index to retrieve the trips that cross these cells within the specified time window.

The index aims at speeding up query processing by improving the capability of ComeWithMe to handle an increasing number of ride opportunities, thus making the system scalable. We have a vocabulary entry for each cell of the spatial grid. In turn, each cell is associated with a list of postings, where each postings stores the *tripid* and the estimation of the time at which the car will cross the given cell. To speed up query processing, postings are organized into *buckets* where each bucket contains the postings referred to a specific coarse time window. Within each bucket, the postings are ordered by *tripid*. The organization in buckets avoids the scanning of the complete postings list when searching for trips, while the postings ordering within buckets allows to perform efficiently postings intersection operations.

To retrieve the candidate rides for a user query, ComeWithMe uses the pick-up cell, the time window for the ride (hint for accessing the right buckets), and the possible destination cells. The queries have the following semantics:

$$(pickCell\ AND\ (destCell_1\ OR\ destCell_2\ OR\ \cdots OR\ destCell_n))$$

Trips matching the above Boolean query are retrieved from the inverted index and the ones not respecting the correct order between pickup and destination cells, or the user-specified time constraints, are filtered out from the results list.

Ranking Model. We have already discussed how the query expansion consistently increases the possibilities of rides and, consequently, how crucial it is to propose to the user the rides that she is willing to accept.

The ranking score of candidate rides is thus computed as a linear combination of a set of features, mainly derived from the flexibility preferences the passenger can set through the mobile app: 1) a temporal tolerance indicating the delay of the departure time of the trip respect to the preferred time indicated in the query; 2) a temporal tolerance on the possible anticipation of the trip respect to the indicated preferred time; 3) a spatial tolerance indicating how much the passenger is willing to walk to reach the pick-up point and/or the destination location. Other information considered in the computation of the ride score include the trip duration and the semantic similarity between the actual destination of the ride and the one specified in the query. Intuitively, the duration of the trip should not be too long respect to the duration of the fastest of all the possible rides. On the other hand, the destination venue should be, in order of preference: close to the PoI chosen in the query; another PoI in the same thesaurus category (e.g., a different Italian restaurant when the requested venue was an Italian restaurant); a PoI in the more abstract category of the thesaurus.

The set of candidate rides retrieved from the index are thus ranked by considering:

1. the *distance* that the passenger is supposed to walk to reach the pick-up point and/or the destination location from the drop-off point. This distance is computed as the sum of the distance between the user location and the pick-up point, plus the distance between the drop off point and the destination. The obtained value is normalized with respect to the maximum distance the specific passenger declared to be willing to walk;

2. the *anticipation* of the trip respect to the desired pick-up time. This time is normalized with respect to the maximum time tolerance set in the user preferences;

3. the *delay* of the trip respect to the desired pick-up time (normalized as the previous feature);

4. the normalized *duration* of the trip. This feature has the value 1 when the duration is the minimum among all the retrieved rides, while it has the value 0 when the duration is close to the maximum duration of all the candidate rides;

5. the *semantic similarity* between the preferred destination PoI in the user query and the alternative PoI reached by the current ride. This value is set to 1 when the ride reaches the preferred destination, while it smoothly decreases to 0 as we consider PoIs in broader categories of the thesaurus.

For the first three features lower is better, while for the last two the opposite holds. Candidate rides are ranked using a linear combination of these five features where each feature is associated with a weight $\omega_i \in [-1, 1]$. Since no golden standard is available to optimize weights, the current prototype uses a uniform weighting schema $[-\frac{1}{5}, -\frac{1}{5}, -\frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$. The effect in the result list of manually changing these weights will be shown during the demo.

2.3 The Mobile Application

ComeWithMe has two different profiles of users: the driver and the passenger. The passenger interface allows the user to request rides for a given destination, as shown in Figure 5. Once the required information is filled in, the user can submit the query and see the ranked list of rides offer.

In Figure 5, we see on the left our query example representing a user asking for a ride in the city of Pisa to go to the "Bella Napoli" pizzeria located in "via del Borghetto". The query specifies also the temporal tolerance (delay 30 minutes or anticipate 30 minutes) from the desired departure time at 19.00 and the spatial tolerance indicating the maximum distance the passenger is willing to walk (up to 600 meters).

ComeWithMe returns, for each query, a ranked list of rides where the best options are shown on the top. In our test dataset, during the specified temporal window (from 18:30 to 19:30), we have a total of 276 routine trips, 23 of which spatially matches the query from the pickup point to at least one "Pizzeria" among the 121 in the dataset. Since each trip can pass through many cells where "pizzeria" places are located, the ride search engine retrieves and ranks a total of 156 rides to "pizzeria" destinations (see Figure 5 on the right). Observing the results of the query, we notice that the first two rides are to the intended pizzeria "Bella Napoli", while other destination are "Panuozzo" and "La Greppia".

The passenger can select a ride from the ranked list and visualize some information about the driver and other details about the ride (e.g. the pickup address and time, the destination place, the estimated arrival, etc). Once the user selected and confirmed a ride, ComeWithMe notifies the driver about the upcoming request. Symmetrically, as shown in Figure 6, a driver can see the list of passenger requests and she can select one to visualize the details. From the details interface the driver can accept or decline the request, she can call the passenger, start a chat and visualize the trajectory on the map.

3. CONCLUSIONS

ComeWithMe is a carpooling system offering alternative destinations, thus boosting the number of available rides and requiring scalability besides efficiency and efficacy. The proposed demoed prototype has been implemented to meet these features showing promising results.

We are extending this prototype to support real usage features. For example, we plan to add the handling of the driver and passenger reputation, the management of groups of users and the optimization of the ranking function to maximize user satisfaction. As ongoing work, we are studying how to evaluate the system, and particularly the ranking function, by exploiting historical datasets. We also plan to conduct tailored user studies to further validate the acceptance of the system.

Acknowledgements. This work was partially supported by the EC H2020 Program INFRAIA-1-2014-2015 SoBigData: Social Mining & Big Data Ecosystem (654024).

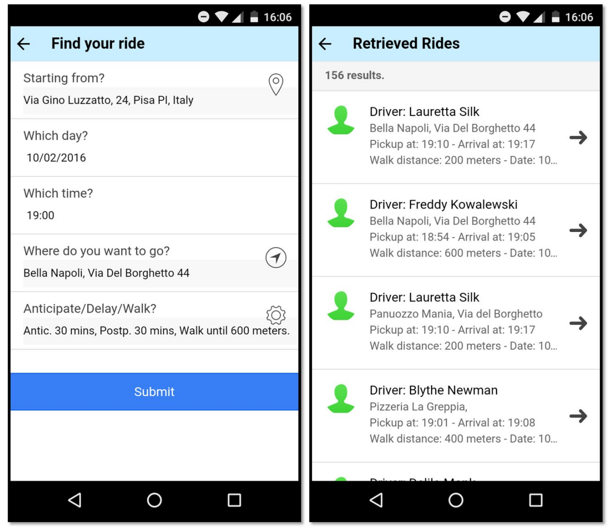

Figure 5: Passenger Interface.

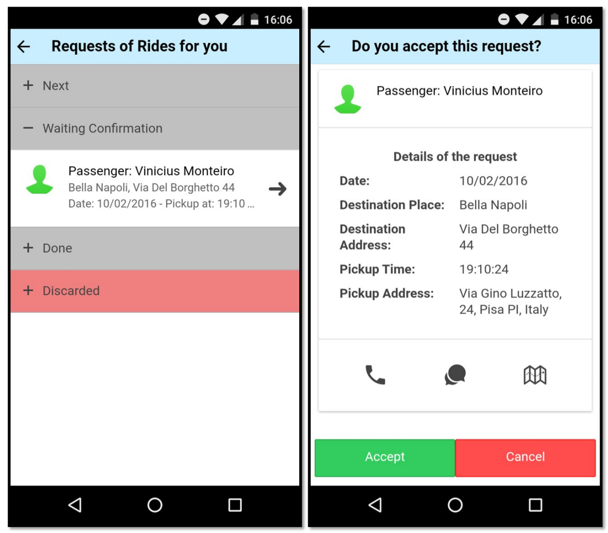

Figure 6: Driver Interface.

4. REFERENCES

[1] C. Carpineto and G. Romano. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1):1:1–1:50, 2012.

[2] V. M. de Lira, S. Rinzivillo, C. Renso, V. C. Times, and P. C. A. R. Tedesco. Investigating semantic regularity of human mobility lifestyle. In *IDEAS 2014, Portugal, July 7-9*, 2014.

[3] V. Monteiro De Lira, V. C. Times, C. Renso, and S. Rinzivillo. ComeWithMe: An activity-oriented carpooling approach. In *IEEE 18th International Conference on Intelligent Transportation Systems*, 2015.

[4] R. Trasarti, F. Pinelli, M. Nanni, and F. Giannotti. Mining mobility user profiles for car pooling. In *Proceedings of the 17th ACM SIGKDD*. ACM, 2011.

[5] L. J. Young. A match.com for carpooling? http://goo.gl/pwTCVb.