Proceedings of the 16th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, The Netherlands, October 6-9, 2013

An Intelligent Carpooling App for a Green Social Solution to Traffic and Parking Congestions

Oussama Dakroub, Carl Michael Boukhater, Fayez Lahoud, Mariette Awad, Hassan Artail
Department of Electrical and Computer Engineering
American University of Beirut
Beirut, Lebanon
Emails: {cab14, ohd01, fal00, mariette.awad, hartail}@aub.edu.lb

***Abstract* —Because existing public transportation infrastructure cannot be adapted in a timely manner to address the daunting traffic and parking congestion in urban environments, researchers are investigating social solutions, such as carpooling, where a driver and one or more passengers having semi-common routes share a private vehicle. Although many carpooling systems have been proposed, most of them lack various levels of automation, functionality, practicality, and solution quality. While Genetic Algorithms (GAs) have been successfully adopted for solving combinatorial optimization problems, their use is highly uncommon in carpooling problems. Motivated to propose a solution for the many to many carpooling scenario, we present in this paper a GA with a customized fitness function that searches for the solution with minimal travel distance, efficient ride matching, timely arrival, and maximum fairness while taking into account the riding preferences of the carpoolers. The computational results and simulations based on real user data show the merits of the proposed method and motivate follow up research.**

Keywords— carpooling; genetic algorithm; intelligent transportation

I. Introduction

The constant population and economical growth has caused an enormous increase in the number of private cars in cities worldwide. This phenomenon has lead to traffic congestions, parking problems, inordinate fuel consumption, and excessive pollution. While the average capacity in a car is 4 passengers, cars are often observed with one rider. In fact 78% of Americans drive alone to work [1]. Because existing public transportation systems cannot be adapted in a timely manner or without major capital investments to address the growing needs of populations, developing social solutions, such as carpooling, where a driver and one or more passengers having totally or partly common routes share a private vehicle would be a green as well as a cost effective public solution to the daunting problem of traffic congestions. Carpooling stands out as an effective and social approach to exploit available transportation resources, i.e., fill the empty seats in private vehicles. It allows people to share a ride for similar departure and destination locations.

Figures 1, 2, and 3 show some relevant statistical trends compiled from [2]. Carpooling presents many benefits. From a financial perspective, it saves money on gas, parking fees, and wear and tear on the vehicle. With an increase in the adoption of carpooling, roads would become considerably less congested, and parking spots would become more available. Moreover, carpooling is very eco-friendly. If four people share one ride, the amount of fuel used to transport these individuals would roughly be reduced by a factor of 4. As can be seen in Fig. 3, highway and off-highway vehicles contribute to roughly 80% of the total carbon monoxide emissions. Additionally, there are also some social benefits to carpooling: less stress from riding, new friendships can be formed during ride time, and riders can read or snooze before reaching their destinations. Also, carpooling would enhance a sense of responsibility with those who tend to be late as they would become more dependable and accountable to the other commuters. Finally, leaving a car at home allows other family members to use it if the need arises. Unfortunately, with all the aforementioned benefits, carpoolers hardly constitute 15% out of all car drivers according to Fig. 2.

Carpooling problems are classified as either a Daily Carpooling Problem (DCPP) or a Long-term Carpooling Problem (LTCPP) [3]. In the first class of problems, each day a set of users declare themselves as drivers for that day. The

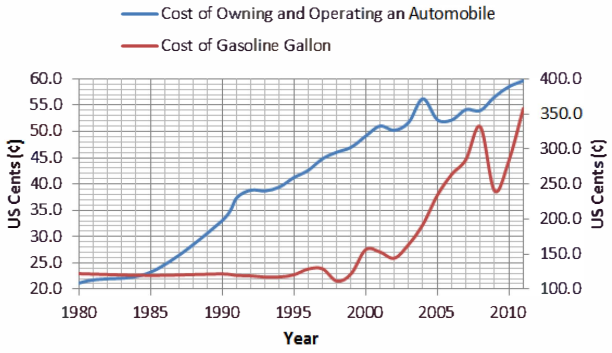

Fig 1. The average costs of owning and operating an automobile and of a US gasoline gallon.

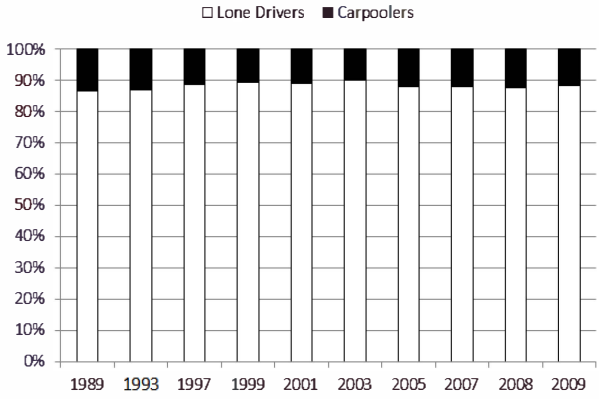

Fig 2. Percentages of lone drivers and carpoolers among people using automobiles to commute to work.

978-1-4799-2914-613/$31.00 ©2013 IEEE

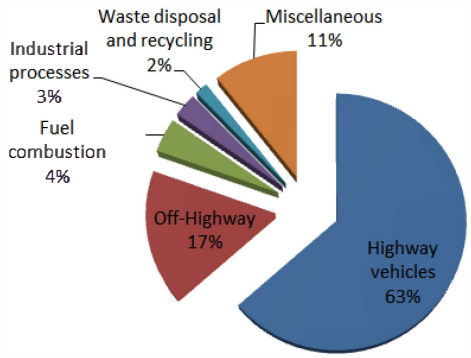

Fig 3. The various factors contributing to carbon monoxide emissions

challenge is to assign passengers to these drivers to ensure the lowest cost routes while respecting the users' schedules. The LTCPP is more challenging, as it is NP-Complete [4]. Users in this problem can be drivers on certain days and passengers on others. It is up for the system to effectively and fairly assign roles for the different days of carpooling. As before, the aim is to assign passengers to drivers and minimize the individual and total travel distance, while respecting users' schedules and maintaining overall fairness.

Existing carpooling approaches rely on direct communication and arrangements between users who know each other or social websites designed for this purpose. With the aim of developing a more complete and practical carpooling system that mitigates traffic and parking congestions, this paper proposes an automated carpooling system based on a genetic algorithm (GA) with a customized fitness function. The system exploits all existing resources to ensure an easy and low-cost implementation. An application running on a 3G/GPS enabled smartphone provides communication between every subscriber and the main system server to offer functionalities such as user registration, ride scheduling and synchronization, and driver tracking. The system will ensure minimized travel distance, efficient ride matching, timely arrival of all users, and fairness in driver selection.

The remaining of this paper is organized as follows: Section II presents related research, while Section III introduces the proposed model. Section IV describes the GA behind the scheduler, while Section V contains the computational and simulation results. Finally, Section VI concludes the paper and discusses future work.

II. Literature Review

Several research projects have tackled the carpooling problem and proposed various solutions. Starting with the work in [5], a distributed algorithm was proposed to map the driver having the earliest departure time to his destination and one or more passengers through a low cost path. The process is repeated until the pool of drivers becomes empty. While this algorithm is simple and adapts easily to newcomers, the solution is suboptimal, since the highest percentage of picked up passengers was reported to be 80%. Also, the fairness component was neglected.

Another solution is based on the Dijkstra Algorithm [6]. The network of users is subdivided into small areas centered on a driver. A check is done on each passenger to see if a car with empty seats passes near him or her, and the assignment is performed incrementally. This solution has a fast runtime as compared to other carpooling solutions, but it is not globally optimal, and thus it will not be fair when considering incremental driven distances.

The Adaptive Genetic Algorithm is another studied approach to solve the LTCPP with a little knowledge about the search space [7]. The GA chromosomes contain pools of several users, half of which are inserted using a greedy insertion method while the other half is randomly inserted. Drawbacks of this approach are the sacrifice of individual fairness to reach a near-optimal solution and the difficulty of adaptation to sudden changes in the system. Yet, another proposed solution is the clustering ant colony algorithm proposed in [8]. The constraints posed by the model are used as preferences (pheromones). The ants trace a path depending on these preferences and other attractiveness formulas. The algorithm is self-adaptive, includes preferences using numerical weights for each, and takes into account both time and distance costs. Near-optimal results can be achieved if the preferences and attractiveness formulas are suitable for the problem. As with most proposed approaches, the main disadvantage is lack of fairness.

In addition to the above, a Lagrangian relaxation method was proposed based on a network flow technique that models the drivers and passengers' routes and schedules [9]. The complexity of the problem increases for large networks and number of variables plus constraints. Lagrangian relaxation finds a lower bound for the problem, which associates a great cost to every unmet constraint. The upper bound for the optimal solution ensures fairness for the drivers over time. Reported results show that this method is fast and efficient for small populations, but it degrades quickly as the population grows larger. Another drawback is the traveling cost that was reduced to monetary cost.

In [10] and [11] the many-to-one scheme is discussed, and is easier to handle than the many-to-many problem that our work deals with. In [10], two methods based on two integer-programming formulations are proposed. The heuristic method yields a valid upper bound, which allows a feasible solution out of the Lagrangian lower bound solution. The exact method combines three lower bounds derived from the problem. In [11], two algorithms are presented: DCA solves the carpooling problem, while DCA-Branch and Bound globalizes the obtained solution.

In [12], an integrated system is presented to organize the carpooling services using the web, GIS, and SMS. The carpooling problem itself is solved by an optimization algorithm, which solves the routing problem heuristically. Nevertheless, their definition of carpooling services is based on the idea that car-owners share rides to the same destination. Users without cars and multiple destinations are excluded in their carpooling services.

One attempt to solve the LTCPP is based on saving functions [13]. Automatic and heuristic data processing routines were developed by the authors to allow efficient matching of rides for passengers and riders. According to the published results, this approach yields more than 50 percent

average savings in car traveled distances as compared to having no carpooling system.

Another attempt at solving the LTCPP is the Multi-Matching System [14], where an optimized matching model was devised to intelligently match the passengers and the riders in the carpooling scheme. The core algorithm is based on Lagrangian relaxation with sub-gradient methods, and focuses on the fairness aspect of the carpooling problem by considering driving distances along with the frequency of being a driver when computing fairness.

Another work that addressed fairness is that of [15], whose research introduces the fair share concept. Their algorithm determined the set of drivers among a group of carpooling participants for a particular day while attempting to secure fairness. In spite of the fact that they accounted for the fairness aspect in carpooling, they do not consider the distance traveled or the participants' schedules.

Differently from the previously mentioned research, our work puts forward an efficient solution that builds on the strong points and improves the insufficiently addressed issues, such as fairness and user preferences, which are major criteria that affect the willingness of a user to participate in a carpooling system. Our solution allows for the participation of users who do not own cars and for the riders' preferences, which is another disregarded issue in previous works we do account for. Our solution insures that all participants secure a ride if the user requests are feasible, given the speed limits and available roads, and if the number of strictly-riders that participate in the system is low.

III. Proposed Model

This section introduces the server and client sides of the proposed system and provides the definition and the mathematical formulation necessary for the understanding of the LTCPP. As stated earlier, the model serves a scheme that allows for multiple origins and multiple destinations. Fig. 4 illustrates the system architecture.

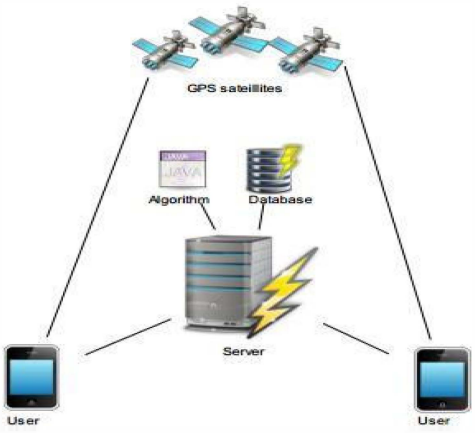

Fig 4. Carpooling system architecture

A. Server

The server side of the system has two main functionalities: it runs the carpooling scheduler and stores all users and rides information. After collecting all the needed information from the different subscribers, the scheduler is daily allocated a six hours time window to run starting 10PM. We assume that no change in schedules occurs after this time. As discussed earlier, matching passengers and drivers is done and all the rides are determined for the upcoming day. The server then communicates all the rides to the respective drivers and passengers. As for the server database, implemented using SQLite, it stores information such as usernames and hashed passwords of the subscribers, users' profiles and preferences, and history of all rides. The database model is shown in Fig. 5.

B. Client

The client side consists of a smartphone application developed for the Android OS. It is the user's portal to the system. The application allows a new user to register providing all the required information such as full name, date of birth, gender, and car details. At login in, a main screen appears allowing a user to add rides or view her/his status. To add a ride, a user specifies the origin and destination by typing the name of the location or using the map to pinpoint it. Departure and arrival times are checked for feasibility against Google Maps before user's requests are sent to the server. Viewing the status of existing ride displays information about the ride timing and details about the driver and the route to be taken. The smartphone needs a 3G connectivity to exchange data with the server, as well as a GPS capability to feed its coordinates into the system. Screenshots of the client application are shown in Fig. 6.

C. Scheduler

To ensure that all participants secure a ride, we assume that the number of cars on a certain day is enough to accommodate all the users who wish to carpool on that day. The model also necessitates that all participants provide their requested rides beforehand. This data includes the car capacity, the origin and destination of the user for each trip, the desired departure and arrival times, and personal preferences. These preferences include desired number of users to ride with, smoking permission, and a blacklist. In addition to the aforementioned user data, other metrics are needed to run the model. These are: the geographical coordinates of all the users and destinations, and the distance and travel time between every two nodes of the network.

Our model can be viewed as a graph $G = (PD \cup SR, A)$, where:

- PD is the set of potential drivers who own cars and can participate as drivers. Each potential driver $pd \in PD$ is associated with an origin and a destination
- SR is the set of strictly-riders who don't have cars and can only participate as passengers. Each strictly rider $sr \in SR$ is associated with an origin and a destination
- $A = \{arc(i, j) \mid i, j \in (PD \cup SR)\}$ is the set of arcs connecting any two different nodes from PD and SR. Each arc has a distance cost, and a time cost to be discussed later.

An example ride is illustrated in Fig. 7 below.

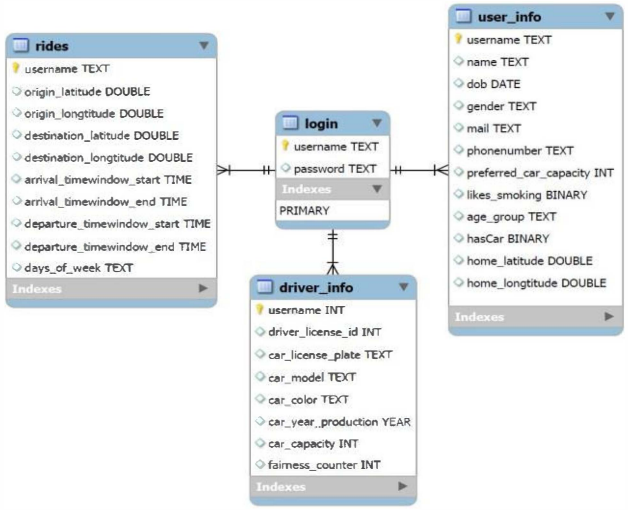

Fig 5. Server database model

Fig 6. Client application screens

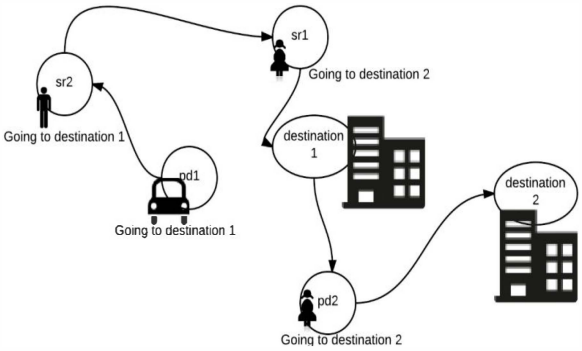

Fig 7. Illustration of a many-to-many car-pooling scenario

Defining a ride R of n users where $n \leq Q$, the vehicle's maximum capacity, each ride will have a driver *pd* and some passengers from the pool $PD \cup SR$. The driver starts the ride from his/her origin, picks up the passengers and drops them off to their destination(s), following the least cost path. The driver's and the passengers' departure and arrival times should be respected. The total cost for this ride will be the sum of the penalties incurred by every passenger, as will be further clarified in the following section.

Given all the users' requirement and constraints, the model aims to solve the LTCPP by minimizing the number of operating cars, assigning the role of driver or passenger for every *pd* while conserving individual fairness, associating passengers with each vehicle, and finding the least cost route for every ride. The objective function, which helps achieve all these goals, is now discussed.

Assuming the following variables:

- OC_i: origin coordinates of user i
- DC_i: destination coordinates of user i
- $T_{D(i)}$: departure time window for user i range $[t_{LD(i)}, t_{HD(i)}]$
- $T_{A(i)}$: arrival time window for user i of range $[t_{LA(i)}, t_{HA(i)}]$
- C_i: fairness counter for user i ($i \in PD$), incremented by one if i drives, and decremented by one if i does not drive
- Q_i: user i's maximum car capacity

The evaluation of the quality of a ride R is based on the following four costs:

1) Distance Cost

- D_k: original distance driven for the driver of car k
- W_j: waypoint j that car k has to pass through (a passenger's origin or destination)
- D_k': extra distance travelled by the driver when in pool k:

$$D'_k = \sum_{j \in k} d\left(W_j, W_{j+1}\right) - D_k$$

- d_k: ratio representing extra distance driven by the driver of pool k

$$f(D) = d_k = \frac{D_k'}{D_k} \quad (1)$$

2) Time Cost

- $T_{A(i)}'$: the time of arrival of passenger *i* at his destination
  $T'_{A(i)} = T_{D(i)} + \sum_{j \in k} d(W_j, W_{j+1}) \times V_{avg} \times \tau \times s$
  - V_{avg} being an average speed depending on the road type (e.g. 80km/h for highways; 40km/h for side roads)
  - τ is the traffic condition that is affected by time of day, weather conditions, and special events
  - s is a security rounding factor to ensure timely arrival
- $f(T'_{A(i)})$: arrival time penalty for passenger i

$$f(T'_{A(i)}) = \begin{cases} 0 \text{ if } T'_{A(i)} \in T_{A(i)} \\ \text{very high (e.g., 1000) if } T'_{A(i)} > T_{HA(i)} \\ 0.5 \text{ if } t_{LA(i)} - 5\,\text{min} < T'_{A(i)} < t_{LA(i)} \end{cases} \quad (2)$$

3) Fairness Cost

- $f(C_i)$: fairness penalty for user *i*

$$f(C_i) = K^{(C_i + k)} \quad (3)$$

The fairness cost will account for the number of days and the amount of extra distance a user drives.

4) Preferences Cost

- P_i: set of preferences of user *i* with their relative priority, e.g., (Smoking, 1), (Only 3 passengers, 3).
- Cost (preference) = $\begin{cases} 0 \text{ if met} \\ priority \text{ if unmet} \end{cases}$

$$f(P) = \sum_{i \in k} \sum_{j \in P_i} cost(j) \quad (4)$$

The total preference cost is the summation of the preference costs of every user in the ride. The smaller the sum of these four costs, the higher the quality of the ride *R*. Hence, the objective function (*OF*) of the model is:

$$OF = f(D) + \sum_{i \in k} f(T(i)) + f(C_i) + f(P) \quad (5)$$

IV. Genetic Algorithm for Carpooling

A. Overview

GAs are adaptive search algorithms based on natural selection and survival of the fittest concept. A GA uses a population of individuals that undergoes solution selection under the influence of mutation and/or crossover operators. A fitness function is then used to evaluate individuals, and the survivability of each individual depends on its fitness, which fits well the car-pooling problem. A standard GA follows the workflow described in Fig. 8.

The algorithm that is used in the system maintains the general structure of a GA but has been modified to improve convergence – which is sometimes an issue with standard GAs. GA convergence time is decreased due to an initial solution that roughly selects the early population instead of a randomly initialized one. Since one of the objectives is to minimize the number of drivers, and a population has a fixed number of drivers, the GA will be run in parallel over different population sizes, and each thread will be checked for survivability.

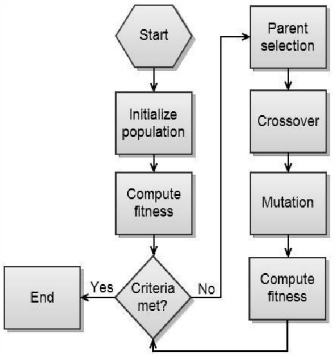

Fig 8. Genetic Algorithm workflow

B. Initial Solution

The initial solution algorithm is a fairly simple algorithm that forms the starting point of the GA. The purpose is to decrease the GA convergence time.

The main components of this algorithm are the arrival and departure times of the user and the extra distance driven by the driver. The initial solution will produce a set of solutions with a varied number of drivers for each solution. Initially, the drivers are randomly selected from *PD*. To assign passengers to vehicles, the algorithm performs three checks. The first one is the existence of at least one empty seat in the driver's car. The second is the evaluation of the ratio of the extra distance to be driven if the pickup takes place to the original distance to be driven without pickup. This ratio must be less than or equal to the average of the ratios computed for all pairs of drivers and unassigned passengers. The third check is if the pickup does not affect the driver's desired arrival time. If the last check does not succeed for the unassigned passengers, the offset from the driver's arrival time (time difference between the calculated time of arrival and the desired time of arrival as inputted by the user) is evaluated. It must be smaller or equal to the average of the offsets computed for all pairs of drivers and unassigned passengers.

C. Population Encoding

Each chromosome will represent a ride and the genes will be the car occupants. The set of all chromosomes belonging to a thread represents a solution, i.e., the number of possible solutions examined is equal to the number of threads running in parallel. Within a thread and after every evolution, the genes will be swapped and/or mutated to obtain better rides. The number of drivers will then be bound

978-1-4799-2914-613/$31.00 ©2013 IEEE

to the number of chromosomes in the population. A sample chromosome is shown in Fig. 9.

Fig 9. Sample chromosome

A driver can be alone or share his/her car with one or more passengers, up to the maximum car capacity. The passengers, as listed in Fig. 8, are not in the order of pickup. Instead, the path is computed using the origin and destination coordinates of each user in the chromosome, and fitness evaluations are based on these calculations.

D. Fitness and Operators

The fitness of a chromosome should reflect the quality of the ride and its probability to be the final solution. Therefore, it should include measures about distance, travel time, and fairness. Hence, the fitness function (*FF*) is proposed to be:

$$FF = \begin{cases} 50 \text{ if solo driver} \\ 100 - \left(f(D) + \sum_{i \in k} f(T(i)) + f(C_i) + f(P) \right) \text{otherwise} \end{cases} \quad (6)$$

The *FF* attributes different weights to the four costs:

1) $f(D)$ ranges between 0 and 100; any value greater than 100 is clamped.

2) $f(T(i))$ has a value of 0 for a solution in the time window, and a value of 100 for exceeding the time limit. This gives the same importance for the time constraints and gives a very low fitness for a chromosome that doesn't respect constraints.

3) $f(C_i)$ is an exponent of a constant. The greater the number of days a user has to drive, the faster this function grows.

4) $f(P)$ is a small integer reflecting the priority that a user contributes to his preferences, e.g., priority of 3 for prohibiting smoking.

GA standard operators have also been modified and one was created as shown below to better suit the requirements of the car-pooling model:

1) Crossover operator: given two chromosomes, two random indexes are computed to determine the crossover positions, and then swap the contents of these chromosomes. A check is performed to test if the first position is still a driver. The change will be committed only in case the resulting two chromosomes are better than their parents. Fig. 10 illustrates this.

2) Mutation operator: The mutation is actually a swap but between two different chromosomes because a user cannot be completely removed from the solution. Therefore, the mutation exchanges two genes between different chromosomes.

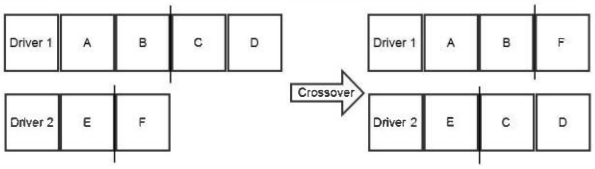

Fig 10. Genetic Algorithm workflow Sample crossover operation

As for the crossover operator, a check on the driver is made, and the same operations apply. Afterwards, the resulting two new chromosomes are evaluated against their older versions and if the sum of their new fitness is greater than their old one, the mutation is kept and the population is updated. This is illustrated in Fig. 11.

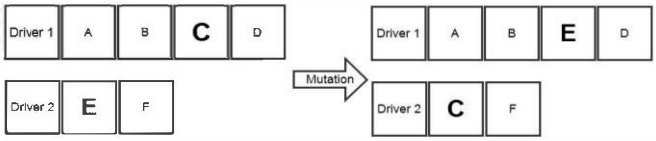

Fig. 11. Sample mutation operation between two chromosomes

3) DriverSwitch operator: This operator was been added because the driver in the chromosome might not be the best driver to take the passengers of this ride, and one of the passengers could hold a higher fitness for this chromosome. This operator randomly switches the driver in a chromosome with another potential driver from the genes. If the resulting chromosome turns out to be better than the previous one, changes will be saved; otherwise, they will not be retained. Fig. 12 shows this "within chromosome swapping".

Fig. 12. Sample DriverSwitch operation in a chromosome

V. Simulation Results

In order to test the proposed model on real world scenarios, we performed multiple tests using real user data consisting of 119 students wishing to carpool to four different university campuses in Lebanon. To build the model and the solution algorithm, the Java language was used. The tests were performed on an Intel Core i7-3632QM CPU @ 2.20GHz and 6 GB of RAM in the environment of Microsoft Windows 8.

A. Carpooling Survey

The survey was answered by 150 adults, mostly Lebanese students including age range. The results showed that 89.33% are willing to participate in an efficient carpooling system. Moreover, they are willing to pay an average of $2.38 per ride. Regarding preferences, 50.6% of the users wish to participate in non-smoking rides, 24% prefer to carpool with people of the same age group, and 75.86% desire a maximum car capacity of four individuals, including the driver. The survey results also showed that 55.33% do not mind leaving between 15 to 60 minutes prior to their planned departure time while 41.33% prefer not to exceed the 15 minutes time window. When asked about late arrivals,

45.33% of the users do not accept arriving late while 41.33% can tolerate up to 15 minutes in lateness. The preceding results are used to set different parameters in the algorithm, such as the relative priority of a user preference.

B. *Advantage of the Initial Solution Algorithm*

This test is run with the full population consisting of 119 users and four destinations. Users are students who wished to commute in the morning to their respective universities, and have provided their actual schedules and constraints. Case A runs the proposed model as described in this paper. Case B runs the GA without initializing the population using the initial solution algorithm, i.e. using a standard GA for original solution encoding. The results are summarized in Table I, where:

- Fitness: fitness score of the chromosome as shown in (6).
- Increase in distance (%): % of extra driving distance incurred by the driver when picking up the passengers.
- # Late arrivals: # of users who did not arrive on time.
- Fairness: fairness score as calculated by the fairness cost function shown in (3).
- # Used cars: # of operating cars, i.e., the number of rides.
- # Solo drivers: # of cars occupied solely by the driver

The results confirm the usefulness of adding the proposed custom initial solution to the GA. Case A exhibits a higher fitness score. The number of operating cars is 32 versus 39 for case B. Also, no driver rides alone in the scenario of case A, whereas there are 5 solo drivers in the other. Also, as shown in Fig. 13, case A converges much faster than case B. The average fitness value of 85.1 is reached in 286 iterations for case A, as compared to 83.8 in 453 iterations for B.

C. *Sparse versus Clustered Distribution of Users*

To study the effect of users' geographical distribution on the carpooling scheme, a second test was conducted using two extreme cases: a very sparse population and a highly clustered one. Although real life data show that populations in general do not fall into either of these categories, but somewhere in between, it is important to check that the model produces acceptable results irrespective of population distribution. Table II shows the results of two cases:

- Case A is for a population of 8 clusters of 20 users each.
- Case B is for a very sparse population of 160 users.

Both scenarios produced very good results: the average fitness of case A is 84.4 and that of case B is 79.7. In both scenarios, the average increase in driven distance does not exceed 10%. Moreover, all the users' schedules are respected and no late arrivals occurred. The number of used cars is 41, which translates to 4 users/car on average. These test cases reflect well the advantages of carpooling: the car occupancy is increased to 4 while the cost incurred on the drivers is acceptable. Another advantage of the proposed approach is that it allows for a global solution rather than a strictly local one. This concept is illustrated in Fig. 14, which is a representation of a ride taken from case A. The nodes and path illustrations were done using Google Maps JavaScript API v3 [16]. In Fig. 15, driver A picks up passenger B from his cluster. Then on his way to destination E, A picks up passengers C and D from a second cluster before dropping everyone off at the final destination.

TABLE I. IMPACT OF INITIAL SOLUTION

Metrics	Cases	
	A (with initial solution)	***B (without initial solution)***
Min/Max/Avg Fitness	71.4/98.8/85.1	50/98.9/83.8
Min/Max/Avg Increase in Distance (%)	0/28.6/7.2	0/14.3/4.1
Min/Max/Avg Fairness	0/1.49/0.93	0/1.47/0.68
# Late Arrivals	0	0
# Used Cars	32	39
# Solo Drivers	0	5

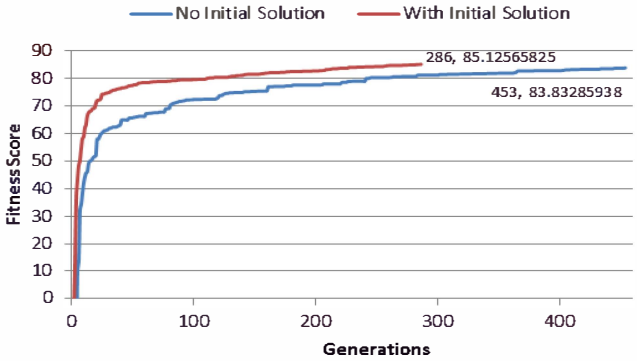

Fig 13. Average fitness vs number of generations for cases A and B

TABLE II. RESULTS FOR DIFFERENT POPULATION DISTRIBUTIONS

Metrics	Cases	
	A (clustered)	***B (sparse)***
Min/Max/Avg Fitness	50/99/84.4	50/97.4/79.7
Min/Max/Avg Increase in Distance (%)	0/23/6.4	0/35.7/9.9
Min/Max/Avg Fairness	0/5.01/2.34	0/1.51/0.89
# Late Arrivals	0	0
# Used Cars	41	41
# Solo Drivers	1	1

D. *Impact of Carpooler Riding Preferences*

The aim of this test is to assess the effectiveness of including the preferences cost function in the evaluation of the solution quality. The population is the same as that of the first test: 119 users going to four different destinations. Case A has a fitness function, which does not account for user preferences whereas case B has the complete fitness function shown in (6). The results are illustrated in Table III.

Three new metrics are shown in this test, these are:

- Smoking (%): percentage of users whose smoking preferences are not met.
- Age Group (%): percentage of users whose age group preferences are not met.
- Car Capacity (%): percentage of users whose car capacity preferences are not met.

Fig 14. Pick-ups from two clusters to a common destination

Based on the survey results, the preferences were given different priorities, which specify their impact on the solution quality. Smoking has the highest priority (3), followed by car capacity (2) and age group (1). When preferences were taken into consideration, only 11.77% of the users had their smoking preferences unmet, versus 21.01% for case A. Moreover, 16.8% were unsatisfied with the age group, versus 26.05% for case A. The car capacity numbers are close: 11.43% for case A versus 18.75% for case B. This is because there are three additional operating cars in case A amongst which one solo driver. This decreases the average number of riders per car in scenario A, which is reflected in a lower percentage of unmet car capacity preferences.

VI. Conclusion

In this paper, a new automated carpooling system is proposed. The objective is to develop an intelligent and reliable transportation solution that reduces travel costs, traffic and parking congestions, and pollution while securing fairness, satisfying carpoolers' preferences, and incurring minimal costs among all participants. The presented approach is based on a customized GA coupled with a structured initial solution that takes into consideration preferences of the carpoolers. Preliminary tests have been performed to assess the efficiency of the model in different scenarios. The simulation results based on real data show that the model provides a high quality solution in a reasonable amount of time. Implementing a user rating system and tracking driver reliability and safety will be the subject of future work. Future research will also involve enhancements to the algorithm so that it can be run in a pseudo real time manner to allow for sudden changes in schedules.

TABLE III. Test Results for Different Fitness Functions

Metrics	Cases	
	A (no preferences)	*B (with preferences)*
Min/Max/Avg Fitness	50/96.62/85.6	71.4/98.8/85.1
Min/Max/Avg Increase in Distance (%)	0/20.2/4.8	0/28.6/7.2
Min/Max/Avg Fairness	0/1.45/0.82	0/1.49/0.93
# Late Arrivals	0	0
# Used Cars	35	32
# Solo Drivers	1	0
Smoking (%)	21.01	11.77
Age Group (%)	26.05	16.8
Car Capacity (%)	11.43	18.75

References

[1] *Carpooling trends in Canada and abroad.* Mar. 2009. http://www.tc.gc.ca/media/documents/programs/cs73e-carpooling.pdf (accessed Nov. 11, 2012).

[2] Online:http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/publications/national_transportation_statistics/index.html. (2013)

[3] Varrentrapp, K., Maniezzo, V., Stutzle, T.: The Long Term Carpooling Problem: On the Soundness of the Problem Formulation and Proof of NP-completeness. Technische Universitat Darmstadt (2002)

[4] Maniezzo, V., Carbonaro, A., Hildmann, H.: An ANTS heuristic for the long-term carpooling problem. In: New Optimization Techniques in Engineering, pp. 411–430 (2004)

[5] Son, Ta Anh, Le Thi Hoai An, Pham Dinh Tao, and Djamel Khadraoui. "A Distributed Algorithm Solving Multiobjective Dynamic Carpooling Problem." *International Conference on Computer & Information Science.* 2012.

[6] Sghair, Manel, Hayfa Zgaya, Slim Hammandi, and Christian Tahon. "A Distributed Dijkstra's Algorithm For The Implementation Of A Real Time Carpooling Service With An Optimized Aspect On Siblings." *IEEE Annual Conference on Intelligent Transportation Systems.* Madeira Island, Portugal, 2010.

[7] Guo, Yuhan - Goncalves, Gilles - Hsu, Tienté. *A Multi-agent Based Self-adaptive Genertic Algorithm for the Long-term Carpooling Problem.* Springer Science Business Media B.V. 2012, 2011.

[8] Guo, Yuhan, Gilles Goncalves, and Tienté Hsu. "A Clustering Ant Colony Algorithm for the Long-term Carpooling Problem." *International conference on swarm interlligence.* Lille, France, 2011.

[9] Yan, Shangyao, Chun-Ying Chen, and Yu-Fang Lin. "A Model With a Heuristic Algorithm for Solving the Long-term Many-to-Many Carpooling Problem." *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*, 2011.

[10] Baldacci R., Maniezzo V. and Mingozzi A.: An Exact Method for the Carpooling Problem Based on Lagrangean Column Generation. Oper. Res. 52(3) (June 2004) 422-439

[11] Ta Anh Son, Le T hi Hoai An, Gerald Arnould, Djamel Khadraoui and Pharn Dinh Tao: Solving Carpooling Problem using DCA, Global Information Infrastructure Symposium (GIIS), Danang 4-6 Aug. 2011, pp.I-6.

[12] Calvo R. W., Luigi F. L., Haastrup P. and Maniezzo V: A distributed geographic information system for the daily carpooling problem, Computers and Operations Research, 31 (2004) 2263-2278

[13] E. Ferrari, R. Manzini. The carpooling problem: heuristic algorithms based on savings functions. Journal of Advanced Transportation, 37(3):243-272, 2003.

[14] Y. Lin. Matching Model and Heuristic Algorithm for Fairness in the Carpool Problem. Doctoral Thesis of National Central University, Taiwan, 2009.

[15] Williams, & Fagin. (1983). A Fair Carpool Scheduling Algorithm. IBM Journal of Research, 133-139.

[16] Google maps api family. (n.d.). Retrieved from h https://developers.google.com/map