Generating Shared Transportation Plans Under Varying Preferences: Ridesharing Models and Mechanisms

Ece Kamar [1]
kamar@eecs.harvard.edu
Eric Horvitz
horvitz@microsoft.com

March 2009

Technical Report
MSR-TR-2009-2011

We introduce and test computational methods that provide a principled approach to ridesharing. We address several challenges with collaboration and coordination among self-interested people aimed at minimizing the cost of transportation and the impact of travel on the environment. The work investigates the problem of applying mechanism design ideas to a dynamic, real-life problem, and evaluates different VCG-like payment schemes in terms of their computational efficiency, budget-balance, incentive compatibility and strategy-proofness in this domain. The Agent-based Carpooling (ABC) methodology employs planning, optimization, and payment mechanisms that provide fair and efficient solutions to the rideshare collaboration challenge. We review the behavior of a working ABC prototype that learns about destinations and preferences from GPS traces and calendars, and considers time, gas, and cognitive costs. ABC identifies beneficial ridesharing plans by performing cost-benefit analyses. The system provides incentives to collaborate in ridesharing by introducing a VCG-based payment mechanism. The ABC methods and prototype have been applied to precomputed, scheduled contexts as well as dynamic settings where requests are handled on the fly, and evaluated on hundreds of real-life GPS traces collected from a community of commuters. The evaluations show promise for reducing the number of vehicles on the road, thus reducing CO2 emissions and fuel expenditures.

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://www.research.microsoft.com

[1] Ece Kamar performed this research during an internship at Microsoft Research.

1 Introduction

Learning and reasoning methods hold promise for addressing challenges to the environment and climate. According to a recent study, transportation is the source of 33% of CO_2 emissions [6]. The study also provides an estimate that computing and communication technology applied to individual transportation could reduce CO_2 emissions by 70-90 MMT in 2020 and generate gross fuel savings of $20-50 billion. Ridesharing has been proposed as a promising means for reducing the cost of transportation and the impact of travel on the environment. In this work we describe an effort to develop principles of coordination that provide a formalized approach to ideal ridesharing that aims at bringing people together in collaborative rideshare plans. Taking a broader perspective, the domain serves as a challenging and representative arena for exploring methods that allow self-interested parties to collaborate effectively in joint plans.

We shall review our approach to the rideshare challenge via bringing together plan optimization with payment mechanisms that provide fair and efficient solutions. We present an adaptive and dynamic rideshare system that considers people's preferences to construct personalized joint plans and to calculate fair payments. An agent participating in a rideshare incurs multiple costs, including time costs for lengthening or delaying a trip, fuel costs for adding new waypoints and cognitive costs for driving. We review how the ridesharing system learns about trip requests and preferences from GPS traces and calendars, and explore the methods that allow the system to take into consideration time, gas, and cognitive costs. The optimization component for constructing collaborative rideshare plans solves computationally expensive optimization problems. We investigate the use of different VCG-based payment schemes in a dynamic setting, and consider computational efficiency, budget-balance, incentive compatibility and strategy-proofness. The methods have been explored within a prototype and experimental platform called the Agent-Based Carpool (ABC) system. We describe our evaluation of multiple aspects of the system using hundreds of real-life GPS traces collected from a community of commuters over five years. The system has been tested with scheduled contexts, as well as dynamic settings in which agents can stochastically enter the system. We explore future models of transportation, and discuss real-world considerations that may impact the performance of the ABC system.

Ridesharing domain allows us to examine the design and application of coordination mechanisms to a dynamic domain where agents have different preferences. Our coordination mechanism generalizes to settings where agents need to collaborate on joint plans to minimize their cumulative cost, and obtain fair payments as incentives. We provide empirical evaluations of the mechanisms on real-life data, which highlight the challenges and tradeoffs that arise from the coordination of self-interested agents in real-life domains. Our mechanism respects the privacy of users, prioritizes fairness and user happiness, and promotes truthful behavior.

There has been growing interest in the use of the web and computing methods in assisting ridesharing. A number of online rideshare services[1] offer users varied experiences from simple to complicated. Nuride, Zimride, Craiglist and mailing groups provide social networks where users can meet and manually arrange carpools. More sophisticated ridesharing services such as iCarpool and CarpoolWorld help users with online trip matching. These services also motivate users to carpool with mile-based rewards, or interfaces that allow users to negotiate on payments. The systems typically serve as a platform to bring users together, rather than as an active mechanism that generates rideshare plans and provides fair payments. In distinction to prior efforts, we employ optimization and mechanism design techniques to explore ridesharing via the challenge of providing personalized collaboration and coordination plans for self-interested parties in a dynamic environment.

Coalescing rational agents into groups of participants in rideshare plans is similar to the *initial-commitment decision problem* (ICDP) proposed by [12]. ICDP determines the set of tasks that an agent needs to commit to in a collaboration. The ABC system is responsible for assigning agents to specific tasks as drivers or passengers. The methods employed in ABC system are an extension to prior work on ICDP as a payment mechanism is included, which provides a rationale and incentives for self-interested agents to collaborate.

Several previous studies on set-cover optimization problems focus on mechanism design for cost sharing [5, 15]. The cost sharing problem focuses on dividing the cost of a service among self-interested agents in a fair manner, where the cost is independent of agents' preferences. Several approximately efficient truthful mechanisms are presented for cost sharing in that work. The optimization used in ABC makes use of greedy optimization procedures similar to the approach taken in the earlier set-cover optimization efforts. However, the payment mechanisms employed in the past are not suitable for collaboration among self-interested agents. We don't have a distinction between service providers and receivers, and the cost is not independent of agents' preferences.

Mechanism design has been applied to the coordination of self-interested robots in sequential decision-making scenarios [2]. In this work, we apply similar ideas to a dynamic optimization problem, where both the joint plan and payments are calculated based on people's preferences, and present detailed analysis of the optimization and payment mechanisms with respect to the computational issues with real-life data.

Song&Regan presents a design for a collaborative carrier network for the truckload tracking industry [20]. That work contains cost-benefit analysis for the truckload domain, but does not provide a detailed analysis of the optimization and payment mechanisms nor address the computational issues with real-life data as way we do in this paper.

In the next section, we describe the architecture of the ABC ridesharing system, focusing on the user modeling component. Section 3 presents the optimization mechanism. We then provide the payment mechanism, and discuss

[1] www.nuride.com, www.carpoolworld.com, www.zimride.com, www.icarpool.com, sfbay.craigslist.org/sfo/rid/

the tradeoffs that arise by applying different payment schemes to ridesharing. Section 5 presents the empirical evaluation of our ridesharing system on real-life dataset. Section 6 explores real-world considerations for our ridesharing domain.

2 Methodology and Architecture

Computing ideal ridesharing plans is a challenging problem as the solution must be adaptable to varied and dynamic preferences of self-interested agents, must provide compelling and fair incentives, and must be easy to use. The ABC system addresses these challenges to create personalized rideshare plans while minimizing the cumulative cost of transportation. The system has three main components that embody separate but interrelated reasoning methodologies: a user-modeling component that accesses and represents the preferences of agents, an optimization component that generates rideshare plans, and a payment component that provides incentives to agents to collaborate.

The user-modeling component is responsible for identifying the preferences of agents about their desired trips, and for passing the preferences into the optimization and payment components. Before the start of the optimization, the user-modeling component gathers information about agents' individual commute plans, including their origin, destination, timing of a trip, and preferences about a return trip. A destination analyzer accesses or guesses the intended destination of a mobile user via direct input of the destination, deviation of recurrent commute patterns from a sequence of GPS trails, or via a dynamic probabilistic inference conditioned on a partial trajectory [14]. To perform cost-benefit analysis of a ridesharing plan, the user-modeling component models agent-specific costs for driving, delaying a trip, diverting an ideal route to pick up or drop off other agents, and changing stop points. Dynamically capturing these costs are crucial for the success of the ridesharing system, as the system needs to adapt to different and changing preferences of the agents. For example, an agent may be willing to wait and pick up other agents on the way when the cost of time is low, but not on a rainy day when time cost is high. The user-modeling component also receives user input on the per mile cost of gas and maintenance.

Time is an important resource and is one of the major factors influencing the cost of different commute plans. The user-modeling component employs a probabilistic time-cost model. The model considers as input the time of day, day of week, and sets of attributes about agents' commitments drawn from an online appointment book. The probabilistic model for the cost of time is learned from user annotated training data via a machine-learning procedure based on Bayesian structure search. See [10, 11] for more details about the machine learning and reasoning about the cost of time in different settings. For the current state of each of the agents, the user modeling component constructs a time-cost function T to estimate the cost of the time spent travelling between the start time (t_s) and end time (t_e) of the trip, and the additional cost for delaying the start time of a rideshare trip from the start time t_s^o to t_s of an independent trip. T is captured with respect to the nearest deadlines drawn from the agent's calendar. Given that the set of calendar items fall between $[t_s, t_e]$ is M, $m \in M$ is a calendar item, the start time of m is t_s^m, the end time of m is t_e^m, c_n is the minute time cost for travelling, c_m is the additional cost for missing a minute of m, c_d is the minute cost for delay; T is defined as,

$$\begin{aligned} T(t_s, t_e) = \quad & ((t_e - t_s) \times c_n) + (|t_s - t_s^o| \times c_d) \\ & +(\sum_{m \in M} \phi(m, t_s, t_e) \times c_m) \\ \phi(m, t_s, t_e) = \quad & \min(t_e^m, t_e) - \max(t_s^m, t_s) \end{aligned}$$

The user modeling component is also responsible for acquiring agent preferences about collaborating with different agents with respect to social networks that they belong to, or their characteristics. The cost model presented in Section 3.2 to model personal costs of users for different rideshares can be extended accordingly to incorporate these preferences.

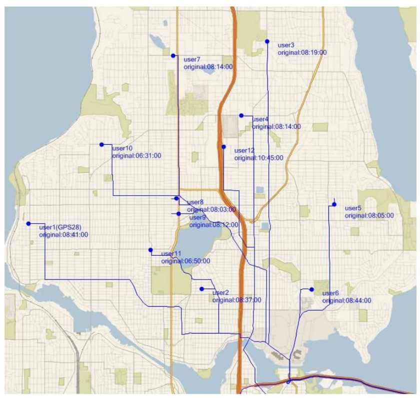
3 Rideshare Optimization

The optimization component groups agents together and generates a collection of rideshare plans that maximizes the efficiency of transportation. The component acquires private user preferences from the user modeling component (e.g., time cost, destination, preferences), combines these preferences with more global contexts to capture the collaborative value of a rideshare plan. The optimization does not reveal any private information about the agents, except the final attributes of the collaborative rideshare plan, which are only revealed to the other members of the rideshare group. The optimization component has two properties that make it difficult for agents to find out about other agents in the system and thus collude in the mechanism; the component combines multiple user preferences and contextual factors to determine the best possible plan, agents do not get to know about other rideshare plans that they are not involved in.

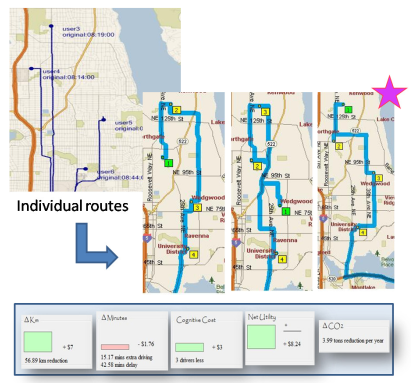
The optimization component takes in the set of individual desired commute plans as inputs and solves two difficult optimization problems to generate a collection of collaborative rideshare plans. The two optimizations are: (1) generating rideshare plans for groups of agents and (2) clustering agents into rideshare groups (see Figure 1).

3.1 A Rideshare Plan

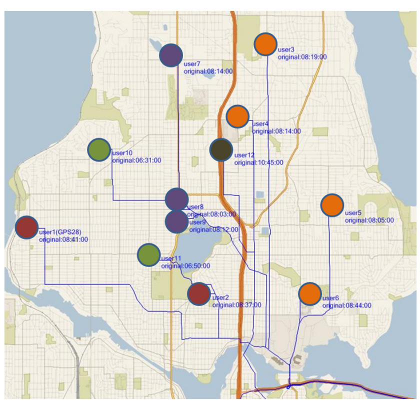
One of the advantages of having a personalized and adaptive ridesharing system is choosing the most convenient plan for a collaborative group of agents among all possible plans, with respect to their preferences. Choosing the best possible rideshare plan is a large search problem where the system explores possible trip start times, stop orderings, stop locations, trip durations, and possible routes



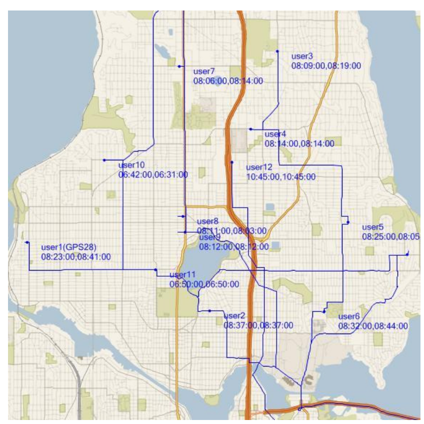
(a) Input: Set of individual commute plans



(b) Rideshare plan optimization



(c) Rideshare group optimization



(d) Output: A collection of collaborative rideshare plans

Figure 1: ABC rideshare optimization steps

among stop points to generate a plan with highest possible cumulative value. Let P be the set of all agents in rideshare system, $S \subseteq P$ a rideshare group, $\mathcal{C}(\mathcal{S})$ the universe of all possible rideshare plans for S. A rideshare plan $C_i \in \mathcal{C}(\mathcal{S})$ is defined by the following attributes:

- $S = \{p_h, \dots, p_q\}$, the set of agents of the rideshare group;
- $p_d \in S$, the assigned driver for the rideshare plan C_i; $S_{-d} = S \setminus \{p_d\}$;
- $\mathcal{L}_{-d} = \{\ell_{h,s}, \ell_{h,e}, \dots, \ell_{q,s}, \ell_{q,e}\}$, the set of start\end (stop) locations of agents in S_{-d}, where p_i's start location is $\ell_{i,s}$, the end location is $\ell_{i,e}$. For all $p_i \in S_{-d}$, $\ell_{i,s}$ and $\ell_{i,e}$ are located in a radius of $\ell^o_{i,s}$ and $\ell^o_{i,e}$ – the initial start\end locations for p_i's individual commute plan. $\mathcal{L}$, the complete set of start\end locations, is the combination of $\mathcal{L}_{-d}$ with the start\end locations of p_d: $\mathcal{L} = \mathcal{L}_{-d} \bigcup \{\ell_{d,s}, \ell_{d,e}\}$, where $\ell_{d,s} = \ell^o_{d,s}$, $\ell_{d,e} = \ell^o_{d,e}$.
- Θ_{-d}, the commute chain excluding p_d, is any ordering of $\mathcal{L}_{-d}$ such that for all $p_i \in S_{-d}$, $index(\ell_{i,s}) < index(\ell_{i,e})$ (i.e., any agent's start location precedes the end location in Θ_{-d}). $\Theta = \ell_{d,s} \circ Q_{-d} \circ \ell_{d,e}$ is the commute chain for S.
- t_s, the start time of the rideshare plan. $t(l)$, the scheduled time of stop location l is defined as below, where $\Delta t(\ell_j, \ell_{j+1})$ is the estimated travel duration betwen two consecutive stop locations ℓ_j, $\ell_{j+1} \in \Theta_{-d}$:

$$t(\ell) = \begin{cases} t_s & \ell = \ell_{d,s} \\ t_s + \displaystyle\sum_{\ell_j, \ell_{j+1} : i < index(\ell)} \Delta t(\ell_j, \ell_{j+1}) & otherwise \end{cases}$$

3.2 Value of Rideshare Plans

Although reduction in gas costs and personal goals of reducing CO_2 emissions from vehicles are considered to be the motivation for bringing self-interested agents to collaborate in rideshare plans, the additional time and travel required for adding new stops to a trip, or having fewer numbers of agents driving in heavy traffic can play an important role in the willingness of agents to participate. We define a personal inconvenience cost that captures several agent-specific cost factors. The personal inconvenience factors are composed to yield the total cost of a rideshare plan.

A model for the cost of personal inconvenience combines the time cost with gas and cognitive costs to estimate the cost of an agent becoming associated with a trip. The user modeling component provides the probabilistic time-cost function, $T_i(t_s, t_e)$. The fuel cost in dollars for one mile is represented as c_g. The inconvenience model combines the input from the user modeling component with traffic prediction services and public contexts (e.g., daily events that may affect the traffic) to construct a cognitive cost model for an agent. $CC_i(\ell_s, \ell_e)$ represents the predicted cognitive cost of p_i for driving between the given stops. The optimization engine makes calls to Microsoft Mappoint services to estimate the travel duration. $\Delta t(\ell_i, \ell_j)$ represents the duration of travel between stops ℓ_i and ℓ_j, whereas $\Delta d(\ell_i, \ell_j)$ represents the distance to be travelled between these stops.

The initial inconvenience cost for agent p_i, $PC^o(p_i)$, represents the cost for an agent following the individual trip that would be created between initial start\end locations of p_i in the absence of ridesharing.

$$\begin{aligned} PC^o(p_i) &= T_i(t^o_{i,s}, t^o_{i,e}) + \Delta d(\ell^o_{i,s}, \ell^o_{i,e}) \times c_g + \\ &\quad CC_i(\ell^o_{i,s}, \ell^o_{i,e}) \\ t^o_{i,e} &= t^o_{i,s} + \Delta t(\ell^o_{i,s}, \ell^o_{i,e}) \end{aligned}$$

where the start time of the individual trip is $t^o_{i,s}$.

A gas and cognitive cost is incurred if an agent is assigned as the driver in a given trip. $\ell_j, \ell_{j+1} \in L$ are consecutive stop locations in commute chain Q. The inconvenience cost of the driver for rideshare plan C, $PC(p_d, C)$, is calculated as follows,

$$\begin{aligned} PC(p_d, C) &= T_d(t(\ell^o_{d,s}), t(\ell^o_{d,e})) + \\ &\quad \sum_{\ell_j, \ell_{j+1} \in Q} (\Delta d(\ell_j, \ell_{j+1}) \times c_g + CC_d(\ell_j, \ell_{j+1})) \end{aligned}$$

The passengers of a rideshare are only subject to time costs for the duration between their scheduled start and end locations. The inconvenience cost of a passenger $p_i \in S_{-d}$, $PC(p_i)$ is:

$$PC(p_i, C) = T_i(t(\ell_{i,s}), t(\ell_{i,e}))$$

$v_i(C)$ represents the value of agent p_i for rideshare plan C.

$$v_i(C) = PC^o(p_i) - PC(p_i, C)$$

The value of a rideshare plan, $V(C)$, represents the value of agents in rideshare plan C for switching to collaborative plan C from their individual plans. $V(C)$ is calculated as,

$$V(C) = (\sum_{p_i \in S} v_i(C))$$

3.3 Rideshare Plan Optimization as Search

Rideshare plan optimization seeks to identify a rideshare plan for a group of agents S with the highest combined value. This optimization problem is a search problem over the universe of rideshare plans $\mathcal{C}(\mathcal{S})$ available for S, where the search dimensions of $\mathcal{C}(\mathcal{S})$ are the set of possible commute chains, set of possible stop locations for the passengers, trip start times and potential routings between stop points. The optimization engine performs geospatial search over the feasible paths that satisfy the constraints of a rideshare plan for S. Given the start\end locations of the assigned driver, the engine computes updated routes by adding potential passenger stop points as waypoints and performing A^* search. The set of potential passenger stop points are selected from a radius around the original stop points of the passenger. The magnitude of the radius is limited by the maximum distance the passenger is willing to diverge from the original stop location to make the trips more efficient. The engine searches the start time of the rideshare plan that minimizes the total cost.

The rideshare plan optimizer selects the plan $C^*(S)$ that offers the maximum total value to agent set S, among all possible plans $\mathcal{C}(\mathcal{S})$. It provides $C^*(S)$ to the rideshare group optimizer.

$$C^*(S) = argmax_{C_j \in \mathcal{C}(\mathcal{S})} V(C_j)$$

3.4 Rideshare Group Assignment as Set Cover

Given a set of agents P in the rideshare system, the rideshare group optimization finds the set of subset of P that covers all agents in P by offering the highest cumulative value. Thus, this optimization is identical to the well-known NP-hard set-cover problem.

Let us consider a set of agents, $P = \{p_1, \ldots, p_n\}$ willing to collaborate in a rideshare system. k is the capacity of a single vehicle, thus the maximum size of a collaborative rideshare group. A set cover for $SC_i = \{S_m, \ldots, S_n\}$ for agent set P is a set of subsets of P, such that for all subsets S_j; $|S_j| \leq k$, $\bigcup_{S_j \in SC_i} S_j = P$, and for any $S_j, S_k \in SC_i$ $S_j \bigcap S_k = \emptyset$. Thus, a set cover SC_i in rideshare system represents a collection of rideshare groups, and their best possible rideshare plans that cover all agents in the ridesharing system without exceeding the capacity of a transportation vehicle. $\mathcal{SC}(\mathcal{P}) = \{SC_1, \ldots, SC_r\}$ is defined to be the universe of all set covers for set of agents P.

We define a valuation function, $V(S_j)$, which corresponds to the value generated by the best possible rideshare plan for bringing agents S_j together. The value of a set cover SC_i, which is also a collective rideshare plan for P is:

$$\begin{aligned} V(S_j) &= \begin{cases} 0 & |S_j| \leq 1 \\ V(C^*(S_j)) & otherwise \end{cases} \\ V(SC_i) &= \sum_{S_j \in SC_i} V(S_j) \end{aligned}$$

A set-cover solver returns the optimal set cover $SC^* \in \mathcal{SC}(\mathcal{P})$ such that $SC^* = argmax_{SC_i \in \mathcal{SC}(\mathcal{P})} V(SC_i)$. However, optimal set cover solver takes exponential time in practice. The two characteristics of the ridesharing domain make the optimal solution infeasible to apply. The dynamic nature of the domain requires the optimization to run efficiently, because agents may unexpectedly arrive, leave, or change preferences which may result in running the optimization multiple times. In addition to the NP-hard complexity of the set cover, the optimization calls expensive online traffic prediction and routing services to evaluate the value of each set cover which makes the optimization more expensive. As a solution, we use an approximate, greedy set-cover algorithm to generate the rideshare groups [15, 21].

The rideshare optimization system ensures that no rideshare group is worse off by engaging in the process. The rideshare group generator includes single-item subsets as well as rideshare groups in the set-cover optimization, thus selects individual (initial) trips for some of the agents rather than assigning them into carpools should no beneficial rideshare plan be available. Thus, any rideshare group generated by the optimizers offers non-negative cumulative utility to the agents. However, ensuring non-negative utility does not guarantees individual rationality or fairness between agents in the rideshare system. The system may incur additional costs to the assigned driver for a group while generating benefit for the other passengers. The next section investigates payment mechanisms that can fairly divide the collaborative benefit generated by the rideshare optimization component.

4 Mechanism Design for Rideshare

Drivers of carpools usually bear additional commute costs for adding new way-points to their original plans to generate value for the collaboration. As the ABC system focuses on bringing self-interested agents together, we need a payment mechanism that distributes the value generated by the ridesharing plan fairly among agents. The payment mechanism motivates the agents to participate in the plan.

As stated by the impossibility theorem, no exchange mechanism can be efficient, budget-balanced and individually rational [16]. Moreover, expensive payment calculations may not be feasible for a dynamic system. We shall present our initial VCG-based payment mechanism, and then explore the tradeoffs with applying the mechanism within the ABC prototype in terms of efficiency, computational complexity, budget-balance, and individual rationality.

4.1 VCG Payments for ABC

Sharing costs for fuel among agents is a simple but widely used payment mechanism in ridesharing. However this simple payment scheme is not suitable for a personalized ridesharing system, because this payment scheme does not consider varying user preferences in payment calculation. Using such a payment scheme in ABC would make the system vulnerable to deceptive reporting of needs by individual agents targeted at making carpool plans more suitable for their preferences.

ABC's payment mechanism distributes VCG-based payments to promote truthful behavior, to ensure fairness and the ultimate sustainability of the system, while maximizing total value of the collaboration [23, 8, 3].

The valuation of agent p_i for a collective rideshare plan SC is $v_i(SC \in \mathcal{SC}(\mathcal{P})) = v_i(C^*(S))$, given that p_i is involved in rideshare plan $C^*(S) \in SC$. Let ρ_i be p_i's payment to the system, $v_i(SC) - \rho_i$ represents p_i's utility. The

VCG payments for ABC system are calculated as below, given that V^*_{-i} is the collaborative value of the collection of rideshare plans (SC^*) to all agents except p_i, $(V_{-i})^*$ is the value of the collection of rideshare plans when p_i is excluded from the ABC system:

$$\rho_i = (V_{-i})^* - V^*_{-i}$$

If the carpool policy calculated by the optimization component is optimal, the VCG payment mechanism is efficient – its output maximizes social value, is individual-rational – all agents have positive utility by participating, and strategy-proof – truth-telling is a dominant strategy.

The VCG payment component does not overburden the agents by inquiring about the utility of each potential rideshare assignment. Instead, valuations are generated by the system based on acquired preferences.

4.2 Tradeoffs on VCG Based Payments

Applying VCG payments to ridesharing optimization faces several challenges. First, the VCG payment mechanism is not budget-balanced, and may return a loss. Secondly, calculating VCG payments in a dynamic mechanism is computationally expensive. Third, VCG mechanisms require the computation of optimal outcomes to ensure truthfulness. The ABC system calculates VCG-based payments based on an approximate optimization of rideshare assignments and routes. Thus, agents are not necessarily incented to be truthful [17].

We modify the VCG payment scheme to adapt it to the dynamic requirements of ABC the problem. To simplify the analysis, we make the assumption that removing one agent from a carpool group does not affect the rideshare allocation of agents outside of that group. We calculate local VCG-based payments, which computes the VCG payment of agent p_i only among the agents that share the same carpool as p_i. This assumption makes payment calculations significantly more efficient, as carpool optimizations for payment calculations are done over a small subset of all agents.

We tested the local VCG-based payment scheme on a large dataset of GPS trails that we describe in more detail in Section 5. The experimental results show that value distribution with local payments maintains 99.7% to 100% of individual-rationality among agents with varying fuel and time costs. However, the evaluation highlighted the prospect of incurring a deficit with VCG-based payments. In our study, we found that the system pays drivers more than it collects from the passengers. To sustain the carpooling system with local VCG-based payments, the system needs to distribute 55% to 79% of the cumulative value generated with carpools back to agents as payments. The deficit of the system grows proportional to the average time costs of the agents, as it gets harder to bring self-interested agents together when time cost is high.

Given the challenge with balancing the budget, we experimented with another VCG-centric scheme. Previous work presents a threshold-based mechanism that enforces budget-balance as a hard constraint on payment calculation

[19]. We modified the local VCG-based payment scheme with the threshold rule specified by Parkes et al., to eliminate deficit. $\Delta_{vick,i}$ represents the non-negative portion of VCG payments which is called Vickery discount.

$$\Delta_{vick,i} = V^* - (V_{-i})^*$$

where V^* is the cumulative value of rideshare plans.

For some parameter $C \geq 0$, we redefine threshold discounts $\Delta^t_{vick,i}$, and payments p_i as given below. With linear programming, we calculate the threshold parameter C that eradicates the deficit, and use this parameter to calculate threshold-based payments.

$$\begin{aligned} \Delta^t_{vick,i} &= max(0, \Delta_{vick,i} - C) \\ \rho^t_i &= v_i(SC^*) - \Delta^t_{vick,i} \end{aligned}$$

Studies with the real-world commute dataset using the local VCG-based payments with the threshold rule demonstrated that the revised mechanism was able to eliminate the deficit for a range of time and fuel cost values. The mechanism did not negatively influence the individual rationality nor the efficiency of the ABC system.

With threshold-based payments and suboptimal outcomes, our mechanism is not guaranteed to be truthful. Investigating the effect of using the local payments and threshold on the truthfulness of agents requires a deeper level of analysis on the system. Parkes et al., states that the threshold-based payment scheme has better incentive properties than other rules proposed in their work. We believe that the payment scheme is hard to manipulate by bounded-rational agents given the incomplete information available to agents about other agents and the indirect affect of an agents preferences on outcomes. To provide additional motivation for truthfulness, the implementation of the ABC payment mechanism can be further enriched with second chance mechanism as proposed by Nisan&Ronen [17].

5 Real-World Trip Dataset and Studies

The ABC prototype provides options for offline, batch optimizations and for real-time simulations of incoming ride requests based on the dynamic queuing and assignment of carpools. Statistics are maintained on multiple dimensions of cost and savings. The system also provides visualizations of paths on a city map. We ran studies based on the driving trip data gathered from 215 subjects over a 5 year period [13]. These subjects voluntarily placed GPS receivers with logging in their cars over several weeks. Nearly all the subjects live in the Seattle, WA USA area. The GPS receivers were programmed to record GPS data only when the users are in motion. The dataset contains a total of 1,434,308 (latitude, longitude) points for an average of 6,671 points per participant.

As the goal of this research is generating carpool plans for daily commutes of users, we segmented the dataset into discrete trips. We identified any two consecutive GPS points either 5 minutes or more than 7 km apart as two separate trips. The trips that are shorter than a threshold are eliminated, which resulted in 7,377 individual trips. For each user, we selected a pair of morning and evening trips that appear to capture daily commute patterns of the users by having the following properties: (1) the regularity of the commutes on trip data of the user, (2) minimum divergence of the selected commutes from a round trip. 215 morning\evening commute patterns were extracted for 215 users in dataset with an average duration of 26 mins for morning, 29 mins for evening, and average distance of 21km for morning and 24 km for the evening.

The commute patterns extracted from the trip dataset is used to test the ABC prototype. We shall present our evaluation of the ABC prototype on batch optimization of scheduled morning/evening commutes, and then extend our studies to explore the performance of the prototype on future models of transportation. Next, we explore the dynamic version of the ABC prototype on real-time simulations. We conclude with an extension of ABC system that considers park&ride lots in optimization.

5.1 Studies on Scheduled ABC

The scheduled model of ridesharing gets the morning\evening commute patterns of individual users as its input, and optimizes both patterns simultaneously to generate the best possible combination of morning\evening rideshares. The scheduled ABC prototype assumes that the morning\evening commuter plans of users are known at the time of optimization and they do not change during the day. The optimization sets as a hard constraint that the morning driver set has to be identical to the evening river set, as all users drove in the morning has to drive back in the evening. However, the allocation of passengers are flexible and may change between morning\evening commutes.

The results of the scheduled rideshare system are evaluated in terms of the the *efficiency on number of commutes* (i.e., the reduction ratio on total number of commutes), *efficiency on number of commutes* (i.e., the reduction ratio on total cost) and the reduction on CO_2 emissions. The system is tested with varying fuel costs (i.e., from \$0.035/mile to \$0.14/mile) and varying average time costs (i.e., from \$0/hour to \$9.6/hour).

Figure 2 compares the individual commute plans with the collection of rideshare plans generated by the system. The thinner blue color on main highways indicates the positive effect of ridesharing on the morning commute traffic in the Seattle region. When the fuel cost is set to \$0.07/mile[2], and average time cost is set to \$4.8/hour, ABC system is able to achieve 41% efficiency on number of commutes, 14% efficiency on total cost of transportation which results in 84.16 tons of CO_2 reduction per year.

To investigate the influence of the cost of fuel on the efficiency of the rideshare optimization, we tested ABC over a range of fuel costs. As shown in Figure 3,

[2]http://www.commutesolutions.org/calc.htm states \$0.07/mile to be the per mile cost of driving.

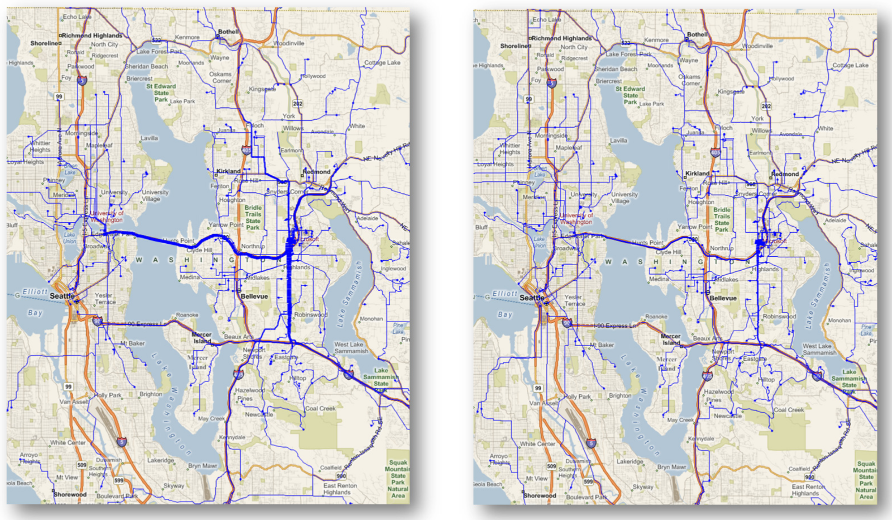


Figure 2: Seattle area map displaying the commute routes of study participants (darker blue color represents more crowded routes) Left: Morning trips without ABC system, Right: Morning trips with ABC system

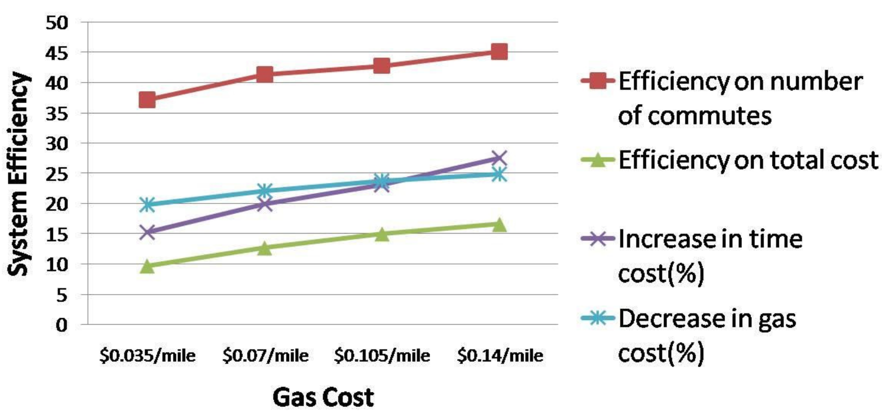


Figure 3: Effect of fuel cost on the efficiency of ABC system

the efficiency of the carpooling system on both the number of commutes and the total cost increases significantly with increases in the cost of fuel. These results indicate that increasing fuel costs can provide increasing incentives for agents to collaborate, and we expect the willingness of agents to carpool to grow as fuel costs increase. Consequently, the reduction on CO_2 emissions increases 25% as gas costs increases from 0.035/mile to \$0.14/mile.

We investigated the influence of changes in the cost of time on the efficiency of the rideshare system by varying the average time costs of users as shown in Figure 4. As the costs of time increase, the efficiency of the optimization with regard to the number of commutes and total costs incurred drops significantly The reduction on CO_2 emissions drop 29.6%. Increasing time costs reduce the incentive of agents to collaborate in ridesharing.

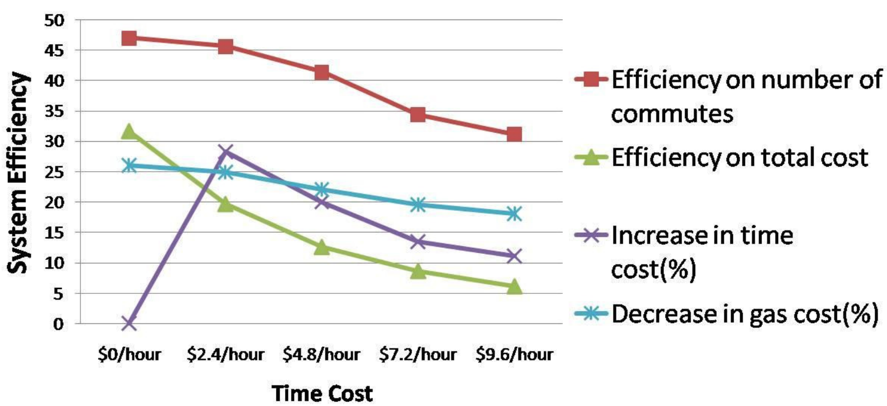


Figure 4: Influence of the average time cost on the efficiency of ABC planning.

The results on both time and fuel costs indicate how the rideshare methods can adapt to changing preferences among agent. The decision-theoretic optimization component is able to balance the costs of carpooling (i.e., increasing time costs) with its benefits (i.e., decreasing fuel costs, cognitive costs). As shown in Figure 4, the system is able to compress the time component of the overall cost of transportation despite increasing per minute time costs of users.

To simulate the effect of increasing the number of agents in the system, we populated commute patterns with randomly created artificial commute patterns. The artificial commute patterns are generated by pairing randomly selected start/end points from the trips dataset, with trip start times taken from a Gaussian distribution representing the start times of the commute patterns in the data. As displayed in Figure 5, the efficiency of the system grows as the logarithm of the number of the agents in the system. With more agents, the system is more likely to find better matches for the users. Therefore, the performance ridesharing system is expected to improve with increasing numbers of users.

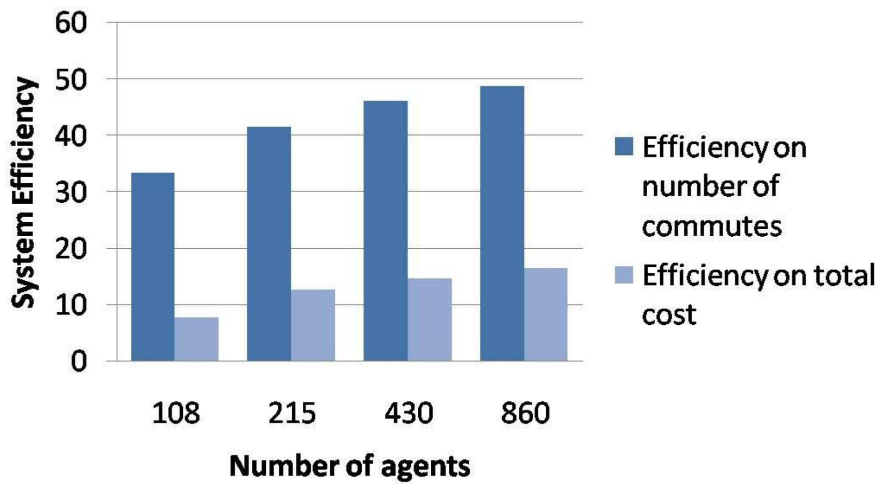


Figure 5: Effect of the size of agent set on the efficiency of ABC planning.

5.2 Studies on Future Models of Transportation

In this section, we focus on future models of transportation in which vehicles are considered to be shared resources that are allocated according to dynamic needs of users (For a popular example of shared vehicles, see [1]). We will show that these transportation models facilitate more efficient and collaborative transportation plans for individual users in comparison to traditional ownership-based model of transportation when incorporated into the ABC prototype. We present two transportation models of shared vehicles; limited zipcars and unlimited zipcars, which introduce different levels of constraints to the ABC optimization, thus result in varying levels of efficiencies.

The limited zipcar model sets as a condition that every shared vehicle driven during morning commute needs to be driven back in the evening commute, but not necessarily by the same driver. Thus, the limited zipcar model relaxes the constraint of the traditional scheduled model by allowing both passenger and driver sets to change between morning and evening commutes, but keeps the shared vehicle set constant between morning and evening commutes. The unlimited zipcar model further relaxes the constraints of the limited zipcar model, and allows vehicle set to change between morning and evening commutes. The unlimited zipzar model is an upper bound on the value that can be provided by the ridesharing optimization, as this model assumes an unlimited supply of shared vehicles and thus minimizes the set of constraint on the optimization.

Figure 6 compares the efficiencies of the future models of transportation with the traditional ownership-based model. As shown in the figure, the efficiency of the carpooling system on both the number of commutes and the total cost increases as the transportation model changes from ownership based to

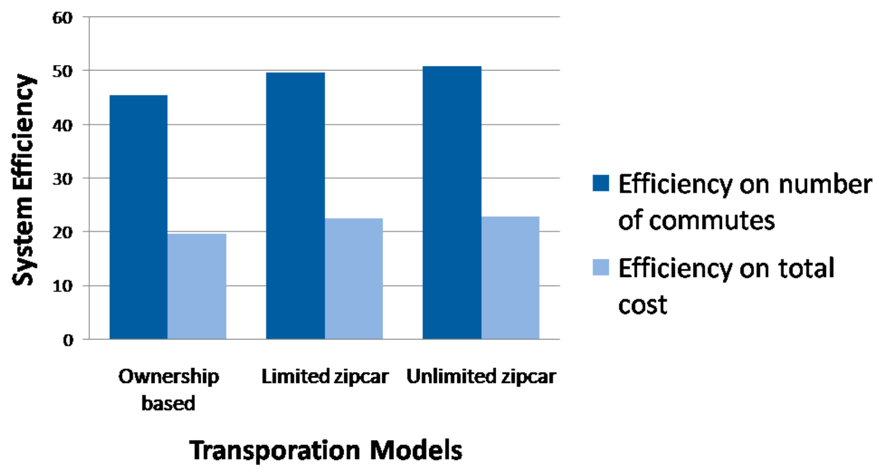


Figure 6: Efficiency of ABC planning with varying transportation models.

limited zipcar and to unlimited zipcar. These results indicate that relaxing the constraints on transportation models can improve the efficiency of carpooling significantly.

5.3 Studies on Dynamic ABC

Dynamic ABC is an extension of the ABC prototype to a dynamic architecture that can handle commute requests on the fly, thus provides users flexibility to add, update or remove commute requests. The dynamic ABC prototype utilizes an online myopic optimization to assign an upcoming commute request either as a passenger or a driver to a carpool, or as an individual commute if there are no beneficial carpools available. The carpool plans are updated dynamically as more commute requests are introduced to the system. The myopic optimization assigns a user to a carpool plan only if doing so improves the combined value of the users. Thus, the system guarantees that every update of the carpool plans increases the cumulative efficiency of the plans. A commute plan with a passed start time is removed from the optimization, and its local VCG payments are calculated within the carpool group according to the mechanism presented in Section 4.2.

Figure 7 displays an instance of the dynamic ABC prototype. The activity window shows the commute requests received at the current time, and the plans that are just started being executed. The waiting window lists the commute plans that are created by the system, but have not being carried out yet. The ABC interface displays the economic analysis of the updates that are performed on the set of commute plans, in terms of the gas, time, cognitive costs and CO_2 emissions. The map interface displays ongoing rideshares in green, ongoing individual trips in red, recently generated rideshares in blue and recently generated

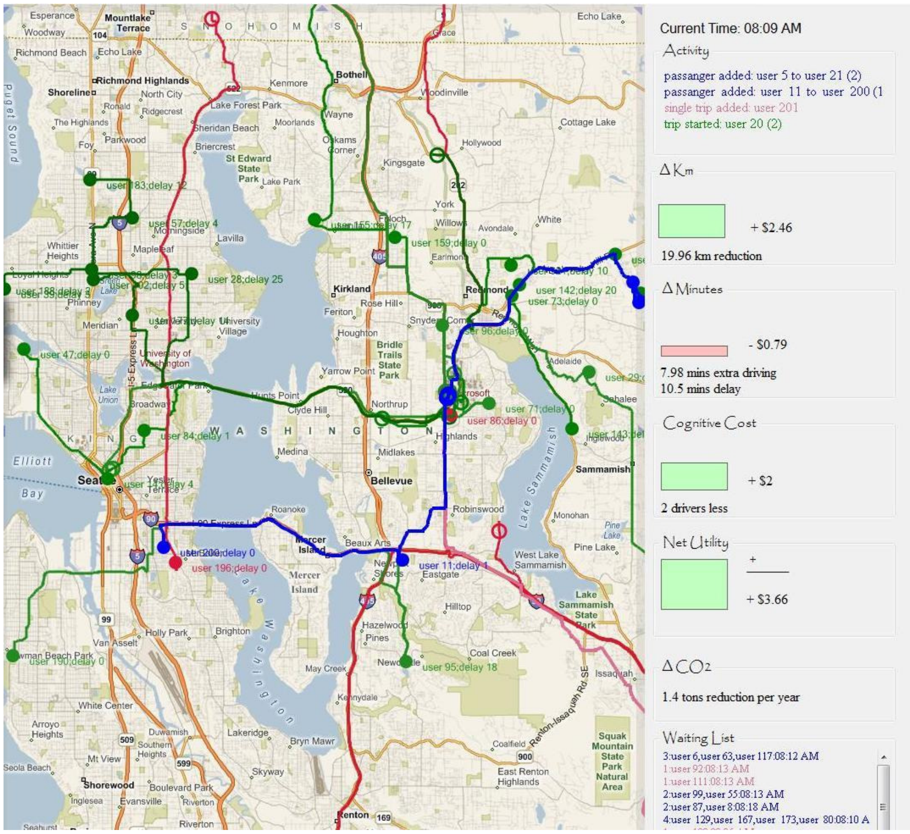


Figure 7: Interface for Dynamic ABC Prototype

individual trips in pink.

The dynamic ABC prototype presents a more realistic implementation of ridesharing with dynamically arriving and leaving users. To test the dynamic ABC prototype, the commute patterns of individual users are sorted with respect to their initial start time, and given as input to the prototype one by one a particular amount of time before the initial start time (i.e., the notification interval). The performance of the dynamic prototype with varying notification intervals is compared with the scheduled prototype in Figure 8. These results show that the dynamic prototype is able to generate up to %86.5 efficiency on total cost when compared with the scheduled prototype, and improve its efficiency on number of commutes by %9.4. We observe that the performance of the dynamic prototype increases as the notification interval grows, as the system is able to pair users with more appropriate carpool options when it has a larger set of candidates to consider.

The current prototype implements a myopic optimization that only considers the current set of commute plans of users, without reasoning about their

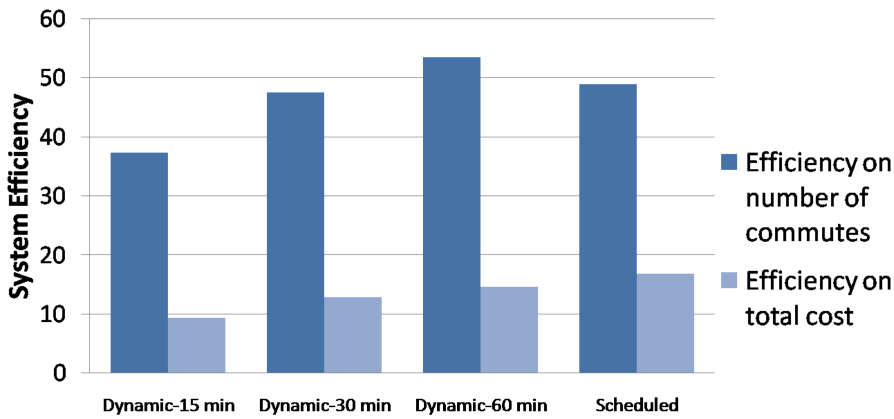


Figure 8: Effect of notification interval on the efficiency of Dynamic ABC system

future commute plans. This simplification may be problematic if the system assigns users as passengers in the morning commute but fails to assign them to carpools in the return trip. To remedy this problem, the dynamic ABC prototype implements the unlimited zipcar model, and assumes that there is an unlimited supply of vehicles for users to use whenever a carpool is unavailable without paying extra cost. In a real-life application of this system, this unrealistic assumption can be satisfied by a backup system such as a taxi or a shuttle service. This myopic system can be improved in future work with non-myopic optimization that can predict the future demand on carpools, and by consequently incorporating the cost of the backup service into the optimization.

Implementing the ABC system as a continuous and dynamic mechanism introduces new challenges in terms of payment calculations. Although, we apply local VCG-payments and calculate the payments within a carpool group, the payment mechanism might get more vulnerable to deceptions from users as they get to continuously interact with the system. In a dynamic mechanism, it is no longer possible to calculate a threshold parameter with linear programming, thus threshold-based payments are only budget-balanced in expectation. In future work, the payment mechanism of the dynamic ABC prototype can benefit from work on online mechanism design literature to adopt itself to these challenges [18, 7].

5.4 Studies on Park&Ride Lots

Park&Ride lots in Seattle area offer a new and efficient way of commuting by combining personal and public transportation. These lots are public and free parking lots distributed through Seattle region by Washington State Department of Transportation. They serve as stop points for public transportation and

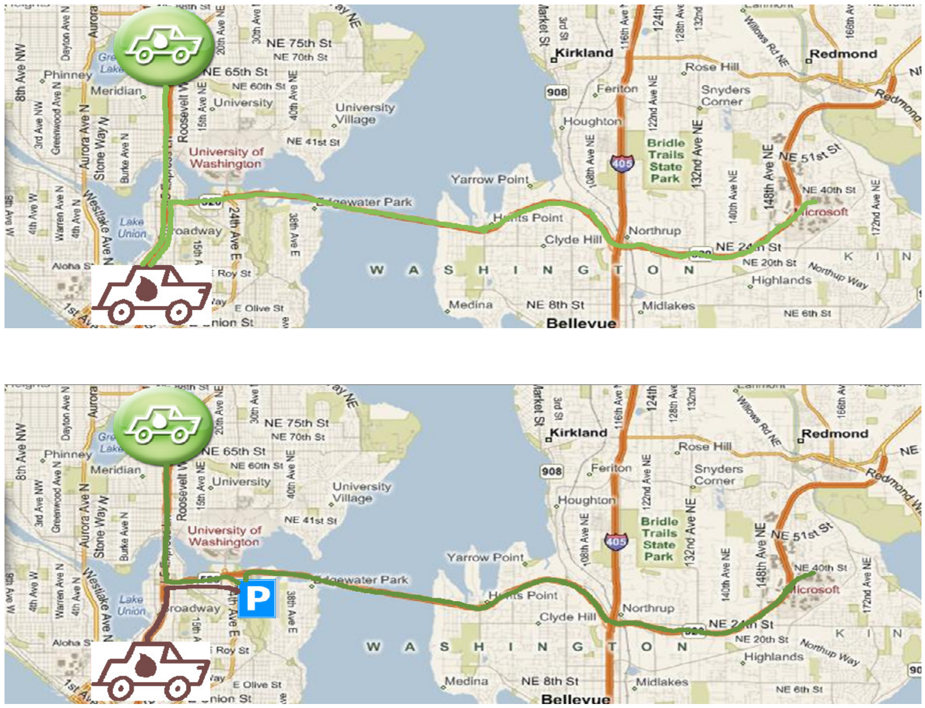
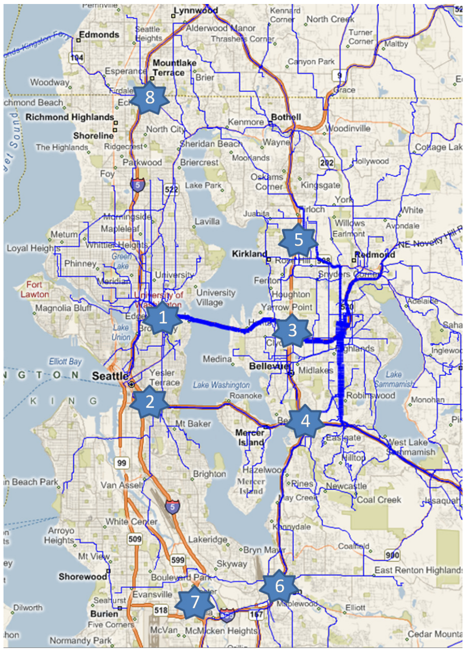


Figure 9: Up: Rideshare plan generated by the default system without considering park&ride lots. Green car picks up the maroon passenger and travels to final destination. Down: Rideshare plan generated after considering park&ride lots. The green and maroon users meet at the park&ride lot, green user drives both users to the final destination.

meeting locations for commuters. In this section, we modify our ridesharing optimization to consider the park&ride lots as waypoints at which passengers of a rideshare can come together with the driver, and form a more efficient rideshare plan. For each passenger of the rideshare, the modified optimization trades off the cost of the driver for picking up the passenger from his/her original start location, and the cumulative cost of the driver and the passenger for getting to a park&ride lot to meet. The optimization adds a stop at the park&ride lot to pick up a passenger, if doing so has a higher cumulative benefit for the rideshare group (See Figure 9 for an example of this analysis). Considering the park&ride lots makes the Rideshare plan optimization more complex as the optimization needs to consider all combinations of pick-up locations (including park&ride lots) for all passengers. It is important to note that, if a rideshare plan includes picking up a passenger from a park&ride lot, the cost of the plan takes into account both the drivers and the passengers individual costs for getting to the park&ride lot.



	Total Commutes	Total Cost
No park&ride	110	923.17
Location 1	108	919.27
Location 2	109	922.35
Location 3	110	923.27
Location 4	110	923.17
Location 5	108	922.44
Location 6	110	921.07
Location 7	110	921.75
Location 8	109	922.93

Figure 10: Left: Eight park&ride lots selected on Seattle region. Right: Table listing the effect of each park&ride lot on the efficiency of ridesharing. No park&ride represents the default model that does not consider park&ride lots.

The table in Figure 10 presents an initial investigation of the effect of park&ride lots on the efficiency of ridesharing for the MSMLS dataset. Due to practicality reasons, this empirical investigation makes two limiting assumptions (1) the optimization considers one of the 8 park&ride lots given in Figure 10 at a given time, (2) the optimization assumes that either all passengers and the driver meets at the park&ride lot, or all passengers are picked up from their original start locations. The results show that 6 of the park&ride lots considered at the experiment improve the efficiency of ridesharing on total cost, 4 of the locations improve efficiency on total number of commutes. In our future studies, we will focus on extending the optimization algorithm to consider multiple park&ride lots and possible combinations of park&ride and original start locations for passengers. We believe that these extensions will improve the effect of park&ride lots on the efficiency of ridesharing further. In future work, our optimization engine can be utilized as a guide to determine ideal park&ride placements with respect to the cumulative cost of transportation.

6 Real-World Considerations

This paper focuses on computational methods for efficient and fair ridesharing. However sustaining the success of a live ridesharing mechanism requires thinking about social factors and contingencies that may affect the mechanism. This section highlights and discusses major real-world considerations related to ridesharing.

Providing fair and satisfactory incentives is crucial for the success of the ridesharing mechanism. The users may bear additional costs for adopting their commutes with others, and they must be satisfied with the compensation they receive from the system. The payment component of the ABC system is based on monetary payments. In mechanism design literature monetary payments are mostly used in electronic commerce [22]. When it comes to providing incentives in dynamic mechanisms that manage services (e.g., bandwidth management, p2p services), previous work usually offers non-monetary incentives that affects the quality of service (e.g., improving data quality, increasing the bandwidth) [4]. Possible non-monetary payments for the ridesharing domain might be better parking spots or using fast-lines in highways. However, non-monetary payments may not compensate for extra gas or time costs drivers may bear. On the other hand, burdening users with monetary payments result in hesitancy to join the system. We believe that future user studies on the ABC system will provide more insight about human reaction to different types of incentives and payments, and result in more successful designs for payment mechanisms.

The current design of the ABC system is fully autonomous in the sense that the rideshare plans and payments are computed and dictated to users, under the assumption that human users completely agree with the decisions of the system. Nevertheless, the system might benefit from giving some control to users, especially if monetary payments are involved. Users might get notified about rideshare plans before they plans are finalized, or the system might ask for an approval before committing to a high payment. Sharing control with the user might be particularly important during the trial period of the system to generate trust, when users do not completely understand the way ABC system works. Moreover, user input might be beneficial to better understand the preferences of users (e.g., time and cognitive costs). Incorporating a mixed-initiative component that trades off the cost of interrupting the user with the benefit might improve the performance of the system without overburdening users [9].

When people get involved in the ABC system in real life, we expect that there might be contingencies in which some users willingly or unexpectedly fail to obey their commitments. A driver may fail to pick up passengers, or a passenger may not show up. We refer to these users as deviators. To deter users from failing their commitments, ABC payment component has a punishment module that determines how much a user needs to pay in case of failing a commitment. If a rideshare plan fails, the system runs the optimization component again by excluding the deviator, constructs updated rideshare plans, and notifies users. If no rideshare is available for some of the users, the system backs up to taxi/shuttle services. The punishment of a deviator is the difference of the utilities of all users excluding the deviator between the original and the updated rideshare plans. The deviator pays the additional burden on all other users for failing the commitment.

Success of the ABC system is likely to depend on multiple social and psychological considerations. Although the focus this work is not on these issues, it is important to note that they are crucial for the wide deployment of this system. The system can significantly benefit from social networks, trusted organizations and organizational membership to generate rideshare groups that users are comfortable with. It might be possible to design special incentives that depend on the economics within organizations. The architecture of the ABC system can be further improved with addition of a reputation mechanism that helps to distinguish reliable users from deviators.

7 Summary and Conclusions

We reviewed research on reasoning and optimization for generating ridesharing plans. We explored the problem as an agent collaboration challenge and developed extensions to prior work on coordination among multiple agents and market-based incentives to solve key challenges. We constructed a prototype and explored the performance of the system with a dataset of real-world trips collected over five years. In ongoing work, we are investigating new applications of the tools, including the use of the methods to inform such decisions as to where to locate park and ride facilities and several practical issues with the deployment of a dynamic version of the system in a running online service that serves a city region. We are also exploring extensions to ABC system to account real-world considerations. Beyond the methods and results, we hope that other researchers will be energized about the opportunity to apply inference, optimization, and market mechanisms to address challenges with the environment.

References

[1] www.zipcar.com.

[2] R. Cavallo, D.C. Parkes, and S. Singh. Optimal Coordination of Loosely-Coupled Self-Interested Robots.

[3] E.H. Clarke. Multipart pricing of public goods. *Public Choice*, 1971.

[4] RK Dash, NR Jennings, and DC Parkes. Computational-mechanism design: A call to arms. *IEEE intelligent systems*, 18(6):40–47, 2003.

[5] N.R. Devanur, M. Mihail, and V.V. Vazirani. Strategyproof cost-sharing mechanisms for set cover and facility location games. *Decision Support Systems*, 39(1):11–22, 2005.

[6] Global e Sustainability Initiative. SMART 2020: Enabling the low carbon economy in the information age. 2008.

[7] Eric Friedman and David C. Parkes. Pricing WiFi at Starbucks– Issues in online mechanism design. In *Fourth ACM Conf. on Electronic Commerce (EC'03)*, pages 240–241, 2003.

[8] T. Groves. Incentives in Teams. *Econometrica*, 1973.

[9] E. Horvitz, P. Koch, and J. Apacible. BusyBody: creating and fielding personalized models of the cost of interruption. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 507–510. ACM New York, NY, USA, 2004.

[10] E. Horvitz, P. Koch, R. Sarin, J. Apacible, and M. Subramani. Bayesphone: Precomputation of Context-Sensitive Policies for Inquiry and Action in Mobile Devices. *LECTURE NOTES IN COMPUTER SCIENCE*, 3538:251, 2005.

[11] E. Horvitz, P. Koch, and M. Subramani. Mobile Opportunistic Planning: Methods and Models. *Lecture Notes in Computer Science*, 2007.

[12] L. Hunsberger and B.J. Grosz. A combinatorial auction for collaborative planning. In *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS-2000)*, pages 151–158, 2000.

[13] J. Krumm and E. Horvitz. The Microsoft Multiperson Location Survey. Technical report, Microsoft Research Technical Report MSR-TR-2005-103, Microsoft Research, 2005.

[14] J. Krumm and E. Horvitz. Predestination: Inferring Destinations from Partial Trajectories. *LECTURE NOTES IN COMPUTER SCIENCE*, 4206:243, 2006.

[15] X. Li, Z. Sun, and W. Wang. Cost sharing and strategyproof mechanisms for set cover games. In *Proceedings of the 22nd International Symposium on Theoretical Aspects of Computer Science*. Springer, 2005.

[16] R.B. Myerson and M.A. Satterthwaite. Efficient Mechanisms for Bilateral Trading'. *INTERNATIONAL LIBRARY OF CRITICAL WRITINGS IN ECONOMICS*, 113:399–415, 2000.

[17] N. Nisan and A. Ronen. Computationally Feasible VCG Mechanisms. *Journal of Artificial Intelligence Research*, 29:19–47, 2007.

[18] David C. Parkes. Online mechanisms. In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay Vazirani, editors, *Algorithmic Game Theory*, chapter 16. Cambridge University Press, 2007.

[19] D.C. Parkes, J. Kalagnanam, and M. Eso. Achieving budget-balance with Vickrey-based payment schemes in combinatorial exchanges. *IBM Research Report RC*, 22218.

[20] J. Song and A.C. Regan. An Auction Based Collaborative Carrier Network. *Transportation Research, Part E, under review*, 2003.

[21] Z. Sun, X.Y. Li, W. Wang, and X. Chu. Mechanism design for set cover games when elements are agents. In *Proc 1st Intl Conf Algorithmic Applications in Management (AAIM)*. Springer, 2005.

[22] H.R. Varian. Economic mechanism design for computerized agents. In *USENIX workshop on Electronic Commerce*, pages 13–21, 1995.

[23] W. Vickrey. Counterspeculation, Auctions, and Competitive Sealed Tenders. *The Journal of Finance*, 1961.