Using Intelligent Collaborative Agents for Automating Distributed Taxi Dispatch

Kiam Tian Seow, Nam Hai Dang and Der Horng Lee

***Abstract*— This paper presents the study of a novel approach towards an automated taxi dispatch system that handles current bookings in a distributed fashion. Existing systems in use by taxi operators in Singapore attempt to increase customer satisfaction locally, by sequentially dispatching nearby taxis to service customers. The proposed dispatch system attempts to increase customer satisfaction more globally, by concurrently dispatching multiple taxis to the same number of customers in the same geographical region, and vis-à-vis human driver and dispatcher satisfaction. To realize the system, we propose a multiagent architecture, populated with collaborative taxi agents that can actively negotiate on behalf of taxi drivers in groups of size N for available customer bookings. The dispatch and operational efficiency of the existing and proposed dispatch systems were evaluated through computer simulations on MITSIMLab, an existing simulation-based laboratory originally developed for evaluating traffic management system designs at the operational level. The empirical results, obtained for a 1000-strong taxi fleet over a discrete range of N, show that the proposed system can dispatch taxis up to over 50% faster, with up to 41.8% and 41.2% reduction in customer waiting time and empty taxi cruising time, respectively. A more efficient dispatch system can help maintain a higher standard of customer service vis-à-vis human driver and dispatcher satisfaction, especially when the demand for taxi service is manageable for the fleet size.**

***Note to Practitioners*— With the liberalization of the taxi industry in Singapore, keener competition among taxi operators has emerged. We believe the taxi operator that leads the competition will be the one with the best automated taxi dispatch system, offering the highest cost productivity and customer satisfaction. However, recent statistics on customer satisfaction, as reported in the news (e.g., [1]), question the efficiency of existing taxi dispatch systems. While many factors might have influenced and contributed to this outcome, our research is motivated by the need for better automated approaches to match customer service requests and taxis, whose arrival and availability, respectively, might be sporadic or not known *a priori*. In this paper, we propose a novel multiagent system, called NTuCab dispatch, to automate taxi dispatch in a distributed fashion. Our experiments for a 1000-strong taxi fleet show that NTuCab dispatch can outperform existing centralized dispatch, in terms of reduction in dispatch, customer waiting and empty taxi cruising times. Additionally, NTuCab dispatch can be implemented on an existing technological infrastructure, providing the opportunities to harness the existing power of multiple intelligent transportation technologies. In future research, prior to a large-scale implementation, we will need to add more features and investigate their effectiveness towards achieving overall efficiency, including techniques to influence and better match the physical distributivity between service demand and non-empty taxi supply in real-time.**

***Index Terms*— Taxi Dispatch Operation, Collaborative Agents, Intelligent Public Service Transportation.**

I. Introduction

Taxis are a convenient means of public transport in many countries, including Singapore. In providing quality customer service, fast and efficient fleet dispatching is essential. In Singapore, online dispatch of available taxis to current customer bookings is done with the aid of a satellite-based taxi dispatching system; the system utilizes a Global Positioning System (GPS) to automatically locate taxis in real-time [2].

In handling *current* taxi bookings, the major focus of taxi dispatch systems has been primarily on reaching individual customers in the shortest time possible to enhance customer satisfaction [3]. This means reaching the customers via the shortest real-time paths possible. However, merely increasing *individual* customer satisfaction, as is the current practice, is a *local* endeavor, in that it entails assigning the nearest taxi to a customer prioritized in a first come first serve queue, without considering the effects of the assignment on other awaiting customers in the request queue.

In a real world scenario, there are often multiple taxi service requests (current demand) and multiple taxis available (current supply) in a given time window. To improve taxi fleet service performance, ideally, we should simultaneously and optimally assign taxis to service all customer bookings that are made within the time window. This is a challenging problem confronting current taxi dispatch systems. Practically, one feasible approach is to effectively group these customer bookings and then efficiently assign each group to the same number of available taxis. One method that we propose along this vein will be described later. The key purpose is to focus on *group average* customer satisfaction instead of a prioritized individual's. This is a more *global* endeavor that would need to consider the mutual assignment exchange effects among the taxis for the concurrently awaiting customers, namely, '*Would (group) total customer waiting time shorten if two taxis are allowed to exchange their currently assigned bookings*?' The motivation is that, by increasing *group average* customer satisfaction, overall, more customers can be satisfied.

To elaborate, consider a scenario of two available taxis in the vicinity of two taxi (service) requests, as depicted in Fig. 1. The shortest-time path to reach a request location can be

K.T. Seow is with the Division of Computing Systems, School Computer Engineering, Nanyang Technological University, Republic of Singapore 639798. asktseow@ntu.edu.sg.

N.H. Dang is with the Division of Computing Systems, School Computer Engineering, Nanyang Technological University, Republic of Singapore 639798. dang0001@ntu.edu.sg.

D.H. Lee is with the Department of Civil Engineering, The National University of Singapore, Republic of Singapore 117576. cveleedh@nus.edu.sg.

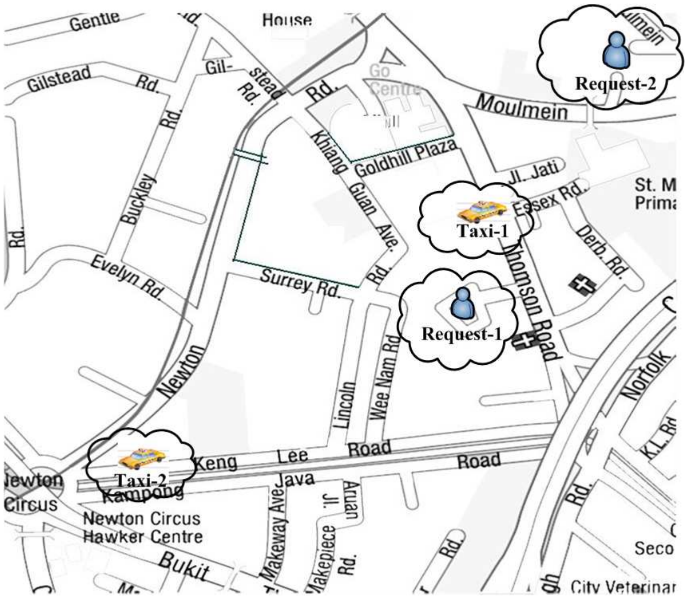


Fig. 1. A taxi dispatch scenario

computed using real-time traffic conditions [3], but for the convenience of illustration, we shall assume that a shorter distance path is also a shorter real-time path. Say, request 1 is initiated before request 2 within a small time window. Under the current practice, requests are allocated different taxis, one request at a time on a customer first come first serve basis. So the dispatcher would have to attend to request 1 first, and assign taxi 1 to service it since taxi 1 is nearer to the request than taxi 2; this leaves taxi 2 to be assigned to the remaining request 2. However, one can see that if we could allow the taxis to exchange their assigned requests, the group average distance (or time) taken by a taxi, and hence the (group average) customer waiting time, would be shorter. This more balanced allocation is often possible if the two requests could be considered *concurrently* for taxi assignment, exploiting the mutual assignment effects in attempting to minimize the total real-time travel period of the taxis in picking up the customers. And the allocation is practically efficient only if it can be done without excessive communication between the taxi dispatch center and the taxis. To the best of our knowledge, current taxi dispatch systems do not support the kind of concurrent taxi assignment envisaged, and cannot therefore exploit this real world scenario to full advantage as needed for improving fleet performance.

In our opinion, to support such concurrent taxi assignment for current taxi bookings necessitates a novel approach to fleet dispatch operation. The new approach should not only aim at increasing average customer satisfaction, but also vis-à-vis average driver and dispatcher satisfaction. And it must be implementable on an existing technological infrastructure. With this philosophy in mind, we propose one where *there are collaborative taxi agents that can, on behalf of taxi drivers, cooperatively negotiate to decide among themselves their different assignments, from among the multiple taxi requests initiated within a time window*. A taxi agent is an active software entity residing in an in-vehicle computing unit of a taxi. By cooperative negotiation [4], several taxi agents can collaboratively search for an assignment solution that they jointly agree.[1]

Although using multiple computing agents is not new in intelligent service transportation [5], we realize that it might be a radically new ideology to deploy them for the specific purpose of taxi dispatch. For, the new idea entails the software ('agents') localized in the in-vehicle computing units to *collaborate* and play a more active role in consensus decision-making, rather than just passively presenting a new request from, and relaying the human driver's request accept-or-refuse decision to the taxi dispatch center. However, in our opinion, investigating this idea is timely, since a multiagent approach will invariably provide a set-up to harness the existing power of multiple intelligent transportation technologies in, for example, vehicle routing [6], automatic vehicle location [7], mobile phone location determination [8], and palmtop-based navigation [9], exploiting the huge investments already made in the internet, wireless communication and mobile devices, GPS-based location, geographic and traffic information systems.

The rest of this paper is organized as follows. Section II describes the basic architectures of both the existing and proposed taxi dispatch systems, with emphasis on the core problem of the latter. Section III presents and discusses a microscopic simulation study comparing the proposed and current dispatch approaches. Section IV concludes the paper and points to some future work.

II. Taxi Dispatch Systems

A. Current Taxi Dispatch System

Fig. 2 depicts the architecture of a current taxi dispatch system in use by a taxi operator in Singapore. Incoming taxi service requests are queued on a first come first serve basis at the dispatch center. For each customer (online booking) request, available taxis in the vicinity of the customer location are considered, and a taxi among them is dispatched to service it, upon the human driver accepting the booking job. An efficient way is to assign a nearby taxi that can traverse the shortest-time path to the customer location, computed using real-time traffic conditions [3].

B. Proposed Taxi Dispatch System

As suggested in the introduction, multiagent collaboration might leverage on the improved dispatch method [3] by exploiting the mutual assignment exchange effects among multiple taxis awaiting multiple requests - a scenario that is not uncommon. Towards this end, an infrastructure[2] to support a taxi agent capable of collaboration is proposed, as shown in Fig. 3(a). And, to effectively utilize such software agents for taxi dispatch, we deploy them in a multiagent architecture, proposed as depicted in Fig. 3(b).

[1] At this juncture, a general notion of negotiation suffices; it will become clearer in Section II-B, where a specific automated negotiation mechanism for taxi agents is introduced, with its concrete details elaborated in Appendix I.

[2] We use JADE [10] as the (Java) agent development environment for agent software implementation; in principle, any other agent development package may be used.

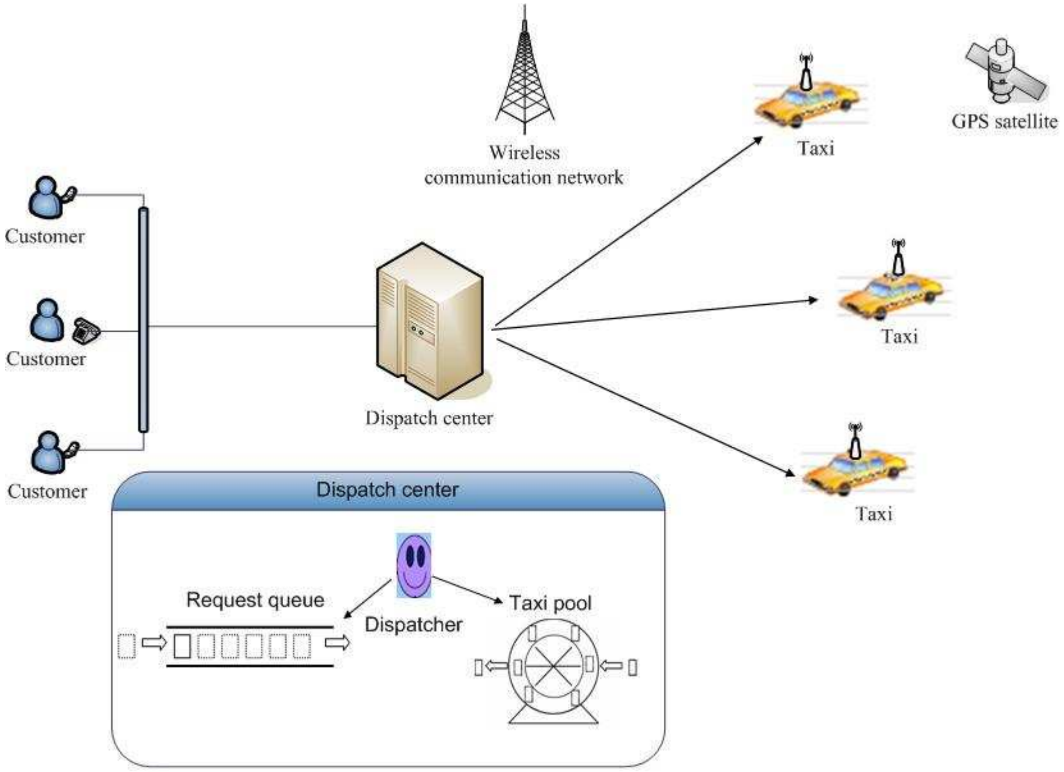


Fig. 2. Current state-of-the-art taxi dispatch system: Centralized architecture

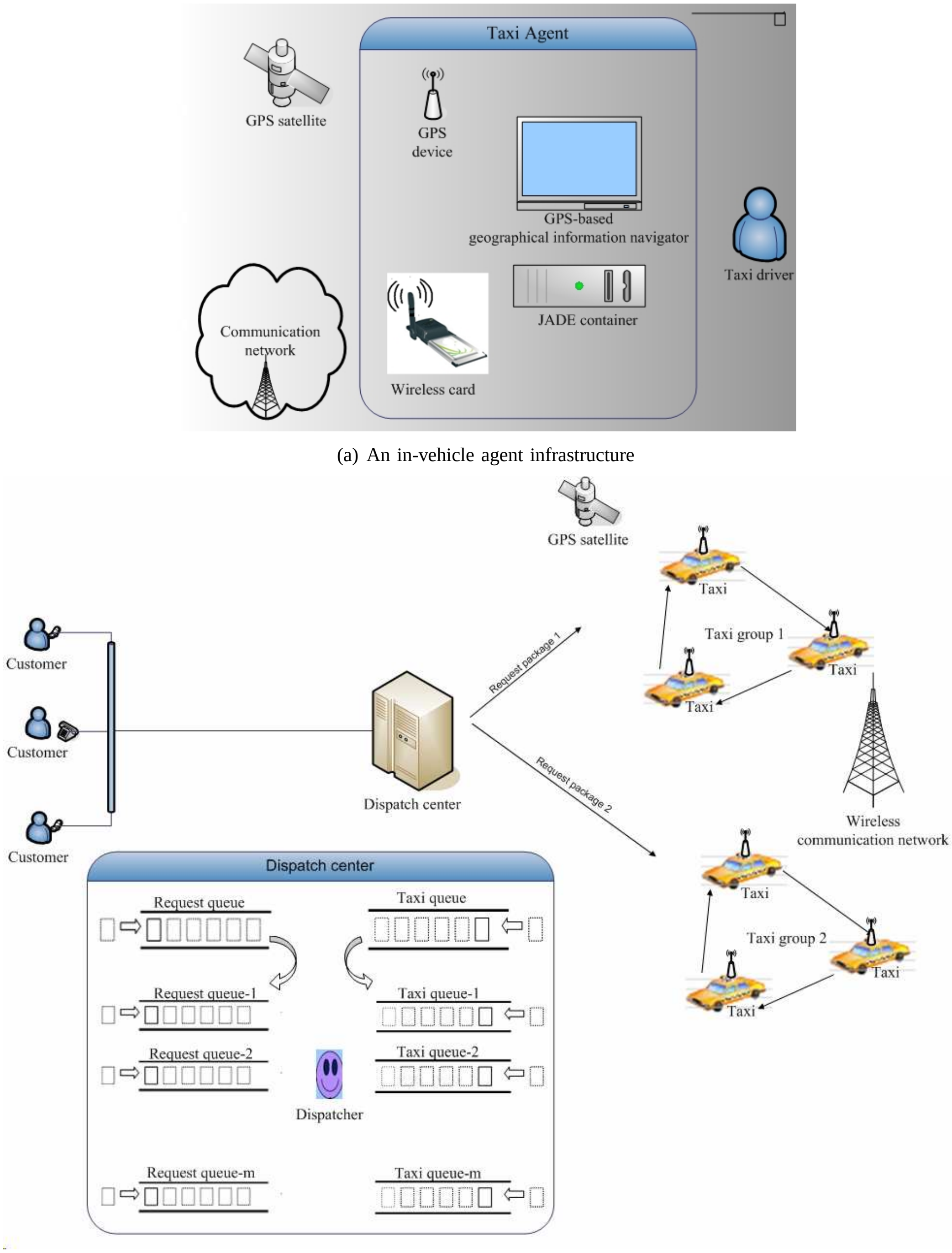
1) Multiagent Taxi Dispatch Architecture: The architecture assumes that a geographical road network is partitioned into m logical areas of taxi operation, $m \geq 1$. This partition is made known to all taxi agents. At the dispatch center, available taxis and booking requests in a logical area are recorded in a taxi queue and a request queue, respectively. The area and corresponding queues are identified by the same index i, $1 \leq i \leq m$.

Initially, all participating taxis are in one of the designated areas of operation, and the taxi queues at the dispatch center are updated accordingly. The 'housekeeping' communication protocol supporting the dispatch operations can then be prescribed as follows:

1) A taxi agent(see Fig. 3(a)) performs the following mandatory tasks:
    a) Announce the *availability of its taxi in a new area of operation* to the dispatch center when (i) the taxi has left its last recorded area and is currently in the new area, and (ii) the passenger has just alighted from the taxi or the taxi is empty. Also does so if its already assigned customer is found to have cancelled the booking.
    b) Negotiate on behalf of its human driver for a booking job when it receives from the dispatch center, a request package containing a group of bookings and the ad hoc group agent members with whom it will collaborate with.
    c) Inform the dispatch center of the taxi driver's decision to accept or refuse its negotiated assignment.
2) The dispatch center performs the following mandatory housekeeping tasks:
    a) **DMT-A**: Update the availability of taxis in the respective queues from a common taxi queue continually updated in response to notification by the taxi agents. When a taxi has left an area x and entered a new area y, update by deleting the taxi record from the taxi queue x, and inserting the record, in a first in first out fashion, in the taxi queue y.
    b) **DMT-B**: Insert the records of taxi bookings continually accepted in a common request queue, on a first come first serve (FCFS) basis, into the respective request queues. If a taxi request is from an area i, insert the record, in a FCFS fashion, in the request queue i.
    c) **DMT-C**: Delete the records of a taxi and the negotiated assignment that the taxi has accepted. Penalize by placing the record of a taxi that has refused to accept its negotiated assignment at the tail of the same taxi queue at the time of update; and compensate by placing that of the refused assignment at the head of the respective request queue.

Under the proposed architecture, the dispatch center carries out the following operations per cycle:

1) Distribute pending requests to available taxis for each logical area i.
    Take a group of N requests from the head of the request



(a) An in-vehicle agent infrastructure

(b) Non-centralized collaborative multiagent architecture

Fig. 3. Proposed *N*TuCab dispatch system

queue i and send it to an *ad hoc* group of taxis assembled from the head of the taxi queue i. The size of N of the last group assembled may vary depending on the number of pending requests and available taxis. Do so until the the taxi or request queue is empty. (As soon as a group of bookings is received, the taxi agents concerned will asynchronously carry out intra-group negotiation over the N requests.)

2) Do housekeeping and await taxi assignment decisions.

   a) Do housekeeping tasks **DMT-A** and **DMT-B** as needed till notification is received about completion of a group collaboration; following which there is a pre-specified period when every agent in the group submits the human driver's decision of either accepting or refusing its negotiated booking.

   b) Do housekeeping task **DMT-C**.

3) Continue with Step (2) if not all the ad hoc taxi groups across all the logical areas have completed negotiation, else do tasks **DMT-A** and **DMT-B** as needed and proceed to next cycle of dispatch operations.

We call the resulting system the ***N*-T**axi Gro**U**p **C**oll**AB**orative (*N*TuCab) dispatch system.

2) Core Problem & Solution: Within this architecture, the core issue in multiagent taxi dispatch is a linear assignment problem (LAP) [11]. The problem is concerned with efficiently assigning every taxi agent with a different taxi request. The efficiency (or optimality) of the concurrent allocation is measured either in terms of minimizing total cost or maximizing total service quality.

To elaborate formally, consider a group of taxi agents $\mathcal{A} = \{a_0, a_1, \cdots, a_{N-1}\}$ of size $N \geq 2$, and a group of different taxi service-requests $\mathcal{O} = \{r_0, r_1, \cdots, r_{N-1}\}$ of size N. The A-QoS (*application quality-of-service*) that an agent $a \in \mathcal{A}$ can offer for each request is defined by $d[a, r]$ for all $r \in \mathcal{O}$. Then our core objective of taxi dispatch is to find, for an $N \times N$ LAP, the particular (total) assignment solution

$$\begin{aligned} \Pi : \mathcal{A} \to \mathcal{O} \text{ such that for } a_i, a_j \in \mathcal{A}, \\ i \neq j \text{ implies } \; \Pi(a_i) \neq \Pi(a_j) \end{aligned} \tag{1}$$

a one-to-one mapping of agents to requests that (ideally) maximizes the total A-QoS S_{tot},

$$S_{tot} = \sum_{i=0}^{|\mathcal{A}|-1} d[a_i, \Pi(a_i)] \tag{2}$$

$\Pi(a) \in \mathcal{O}$ refers to a request selection by agent $a \in \mathcal{A}$ (under an arbitrary permutation of Π); and $\max\{S_{tot}(2)\}$ defines the (ideal) optimal social goal of the agents. An assignment or allocation set corresponds to one permutation of Π (1), and can also be equivalently represented as containing elements of the form $(a, \Pi(a)) \in \mathcal{A} \times \mathcal{O}$.

The generic LAP has been extensively researched in the Operations Research literature [12]. Using agent modeling ideas, recent research has developed agent mechanisms [13], [14] for a multiagent version, termed collaborative LAP (CLAP), where knowledge about the LAP is distributed among the agents, such that every agent $a \in \mathcal{A}$ initially only has its own local information, $d[a, r]$ for all $r \in \mathcal{O}$. The automated mechanisms developed enable collaborative taxi agents to cooperatively negotiate for different requests *by* themselves, in contrast to a centralized algorithm deciding *for* them as in [12]. In essence, these taxi agents can compute and leverage on the possible overall A-QoS increments, gained through reassigning requests among themselves.

One important development for CLAP is a decentralized agent algorithm called MA^3-LM [14], which is well-suited and adapted for use with the proposed taxi dispatch architecture. The reasons for considering MA^3-LM are: (i) the core problem in collaborative taxi dispatch - taxi online assignment or allocation - can clearly be treated as a CLAP, (ii) MA^3-LM is decentralized and so maps directly onto our proposed multiagent architecture, and (iii) being computationally simple and easy to understand, one of our research contributions is to provide the first potential real-world application of MA^3-LM, along with an investigation of its applicability and performance.

The MA^3-LM agents divide their collaborative reasoning process into negotiation rounds. Details of their reasoning per round are documented in Appendix I. To speed up the negotiation process, an assignment initialization heuristic (labelled H-Max) is applied prior to negotiation; the details of which are documented in Appendix II.

3) Theoretical Boundary and Optimal Situations: The physical locations of taxis and customers are significant information for taxi dispatch. They are in general not known *a priori*. However, two theoretical *boundary* situations about their relative locations can be identified:

1) **DT-CC:** Geographically distributed taxis for concentrated customers, as when dispatching taxis to a taxi stand.
2) **CT-DC:** Geographically concentrated taxis for distributed customers, as when dispatching taxis from a depot.

$$\begin{array}{c|ccc} & r_0 & r_1 & r_2 \\ \hline a_0 & x_h & x_h & x_h \\ \hline a_1 & y_h & y_h & y_h \\ \hline a_2 & z_h & z_h & z_h \end{array}$$

(a) For **DT-CC**

$$\begin{array}{c|ccc} & r_0 & r_1 & r_2 \\ \hline a_0 & x_v & y_v & z_v \\ \hline a_1 & x_v & y_v & z_v \\ \hline a_2 & x_v & y_v & z_v \end{array}$$

(b) For **CT-DC**

Fig. 4. Types of CLAP for the boundary situations, shown for $N = 3$

Being 'geographically concentrated' means that the taxis or customers are in close physical proximity of one another. Theoretically, we shall assume that they are at the same location point. Following, the **DT-CC** and **CT-DC** situations result in taxi agents facing the types of CLAP as depicted in Fig. 4, where each row of matrix entries are the respective A-QoS values $d[a_i, r_j]$ as computed by an agent $a_i \in \mathcal{A}$ for every request $r_j \in \mathcal{O}$. The A-QoS data entry $d[a_i, r_j] < 0$ denotes the negation of the shortest (planned) real-time for a taxi (that the agent $a_i \in \mathcal{A}$ represents) to travel from its designated location to the pick-up location of a customer (who initiated the request $r_j \in \mathcal{O}$). This real-time travel period is that computed over a directed road network with real-time traffic information [3]. The smaller this time value is, the nearer to the customer the taxi is said to be.

In between the two boundary situations, we also identify a situation, labelled **OM-TC**, where the geographical distributions between the taxis and the service requests are optimally matched, such that every taxi agent $a \in \mathcal{A}$ has a different request $r \in \mathcal{O}$ that is *the nearest* to its taxi location; all its other requests are not as near. An example of situation **OM-TC** is given in Fig. 5, in which the values shown each indicates the largest A-QoS (or least negative travel time) of an agent $a_i \in \mathcal{A}$ for a different request.

$$\begin{array}{c|ccc} & r_0 & r_1 & r_2 \\ \hline a_0 & - & - & x_0^2 \\ \hline a_1 & - & x_1^1 & - \\ \hline a_2 & x_0^0 & - & - \end{array}$$

Fig. 5. A CLAP instance of **OM-TC**, with $N = 3$

Following, we present two propositions on MA^3-LM in the situations identified. To understand the proofs, the reader is referred to Appendix I for the formal definitions of Belief (B), Desire (D) and Intention (I), and other features of MA^3-LM. For the proof of Proposition 2, the reader is also referred to Appendix II for details of the speedup heuristic, H-Max.

Proposition 1: The MA^3-LM taxi-agents in situations **DT-CC** and **CT-DC** will incur one negotiation round to reach an optimal (or a maximum) total A-QoS assignment.

Proof: We sketch the proof as follows: In either situation, in the first round of negotiation, every MA^3-LM taxi-agent will yield a nil intention (Definition 3) - the termination condition for MA^3-LM. In the former situation, it is due always to an empty belief set (Definition 1) computed by every agent. In the latter, it is due always to an empty desire set (Definition 2) computed by every agent except one, which yields a nil intention due always to an empty belief set. The first round is thus the last negotiation round. Finally, the total A-QoS assignment solution reached must be optimal when negotiation terminates, since, in either situation, it is the same total value regardless of the request selections [under Π (1)]. Hence the proposition. ■

In other words, in situations **DT-CC** and **CT-DC**, MA^3-LM taxi-agents incur the minimum number of negotiation rounds [13], [14].

Proposition 2: The MA^3-LM taxi-agents using H-Max in situation **OM-TC** will incur a total of two negotiation rounds to reach an optimal total A-QoS assignment.

Proof: We sketch the proof as follows: In situation **OM-TC**, immediately after assignment initialization through H-Max, every MA^3-LM taxi-agent [under Π (1)] will have selected or reselected a different request that is the nearest to its taxi location. This initialization process is considered to incur the first negotiation round. In the second round of negotiation that follows, every MA^3-LM taxi-agent will yield a nil intention - the termination condition for MA^3-LM - due always to an empty belief set computed by the agent. The second round is thus the last negotiation round. Finally, since every agent will have selected its nearest request [under Π (1)] when negotiation terminates, it follows that the total A-QoS assignment solution reached is optimal (or maximum). Hence the proposition. ■

III. Simulations & Performance Evaluation

To study the comparative performance of the proposed *N*TuCab dispatch system and an existing centralized system (see Section II-A), we conducted microscopic computer simulations on MITSIMLab [15], [16], simulating taxi operations in a selected ITS-managed urban road network of reasonable complexity. We focused on two performance measures: (i) dispatch efficiency and (ii) operational efficiency. The former for both the systems was investigated in terms of dispatch completion time; and the latter, in terms of customer waiting time versus empty cruising time. In our simulations, *dispatch completion time* is measured as the period for a complete cycle of dispatch operations; *customer waiting time* is measured from the moment a customer raises a request to the moment an assigned taxi arrives to pick up the customer; *empty taxi cruising time* is measured from the moment it is available to the moment it accepts (or commits to service) a negotiated assignment.

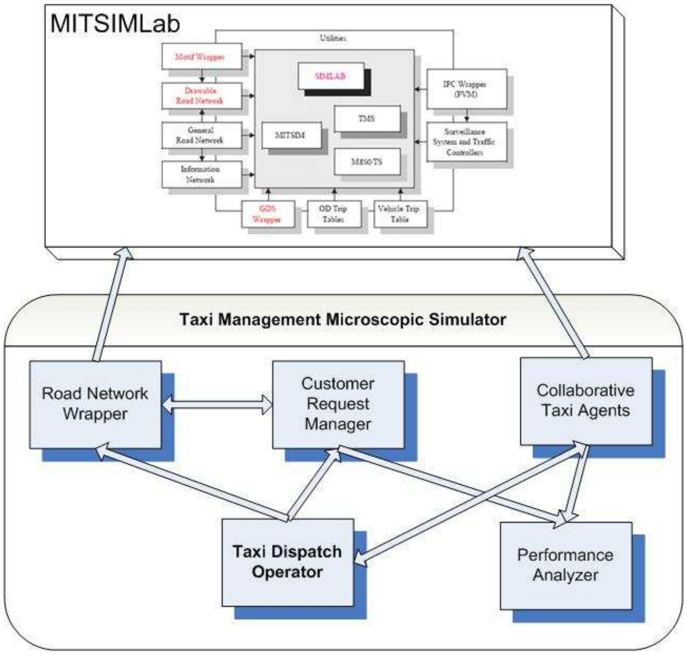


Fig. 6. The TM2S-MITSIMLab simulation model

A. Experimental Scope & Investigation

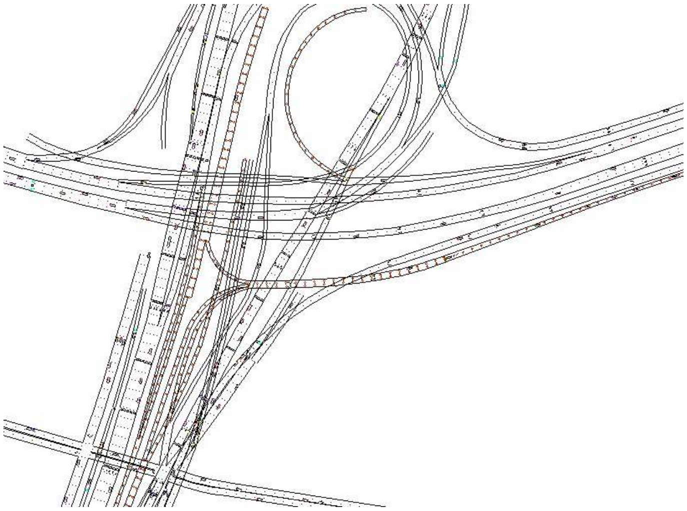


Fig. 7. The urban road network model used

The experiments were performed on MITSIMLab [15], [16], through a Taxi Management Microscopic Simulator (TM2S) developed for this research study. The overall software architecture of the simulator is shown in Fig. 6. The TM2S module wraps around the MITSIMLab to simulate the real time activities of a dispatch operator and the associated negotiation processes of a network of collaborative taxi agents, in accordance to either the centralized or the proposed *N*TuCab dispatch system (see Section II), but confined to one logical area of operation. The urban road network model used for all the experiments is shown Fig. 7; it covers a physical area of about $15km \times 10km$, and was built using a road network editor [16]. In providing the required traffic information for dispatch operation, an abstracted road network is also built over the MITSIMLab-based urban road network model.

For both the centralized and proposed systems, we computed the travel times using real-time traffic information as proposed in [3]. The expected travel time for a taxi to reach a customer's pick-up location (the negation of which determines an A-QoS data) was calculated using the router-choice model mechanism in the MITSIMLab network model; the time computation considered (i) current traffic conditions, (ii) delay and regulation of turning movements at road intersections and (iii) possible penalties for certain designated links (e.g., freeway bias).

In the MITSIMLab environment, the taxis move randomly in the road network. Upon the taxi agents in TM2S deciding and accepting their negotiated assignment, each corresponding taxi moves from its current location to the assigned customer's pick up location, and then to the customer's destination.

In principle, both the existing centralized and *N*TuCab dispatch systems should operate for any taxi fleet size. For our simulations over the TM2S-MITSIMLab model, a taxi fleet size of 1000 was simulated for a one-hour duration. The taxi fleet is about 30 % of the traffic volume that can be simulated on MITSIMLab, and constitutes a reasonable traffic composition in the urban setting considered.

To evaluate *operational efficiency*, we carried out experiments for a range of hourly demand rates (defined by taxi bookings per taxi per hour). For each demand rate, simulated with incoming requests generated by the request manager, the *customer waiting* and *empty taxi cruising* times were recorded for centralized dispatch, and collaborative agent dispatch for several group sizes $N \in \{5, 10, 15, 20\}$.

To evaluate *dispatch efficiency*, we did the simulations for a range of request queue sizes, each as generated by the request manager, and recorded the dispatch completion times.

The experiments were repeated for collaborative agent dispatch with speed-up initialization heuristic incorporated (see Appendix II).

The simulation data (raw dispatch completion, customer waiting and empty cruising times) gathered from the dispatch operator and taxi agents were saved into a text file for off line performance analysis.

B. Analysis of Numerical Results

1) On Dispatch Efficiency: Fig. 8 shows that, in centralized dispatch, the dispatch completion time increases sharply with the request queue size. In contrast, under *N*TuCab dispatch, for each group size N, it remains relatively constant as the queue size increases, due to the advantage of concurrency in multiple intra-group negotiations. But as N increases, the dispatch completion time of *N*TuCab also increases due to the increasingly longer group negotiation.

The dispatch completion time of centralized dispatch is significantly shorter than that of *N*TuCab dispatch when the demand queue size is small (50) or for $N = 20$. However, because of the relative independence on demand queue size, *N*TuCab dispatch tends to outperform centralized dispatch with a shorter dispatch completion time when the queue size increases, with the smaller N-group sizes starting to outperform from a smaller queue size. For instance, $N = 5$ starts outperforming centralized dispatch from queue size 100 whereas $N = 15$ starts from queue size 500; the only exception is $N = 20$, which could not do better than centralized dispatch [see Table I(a)].

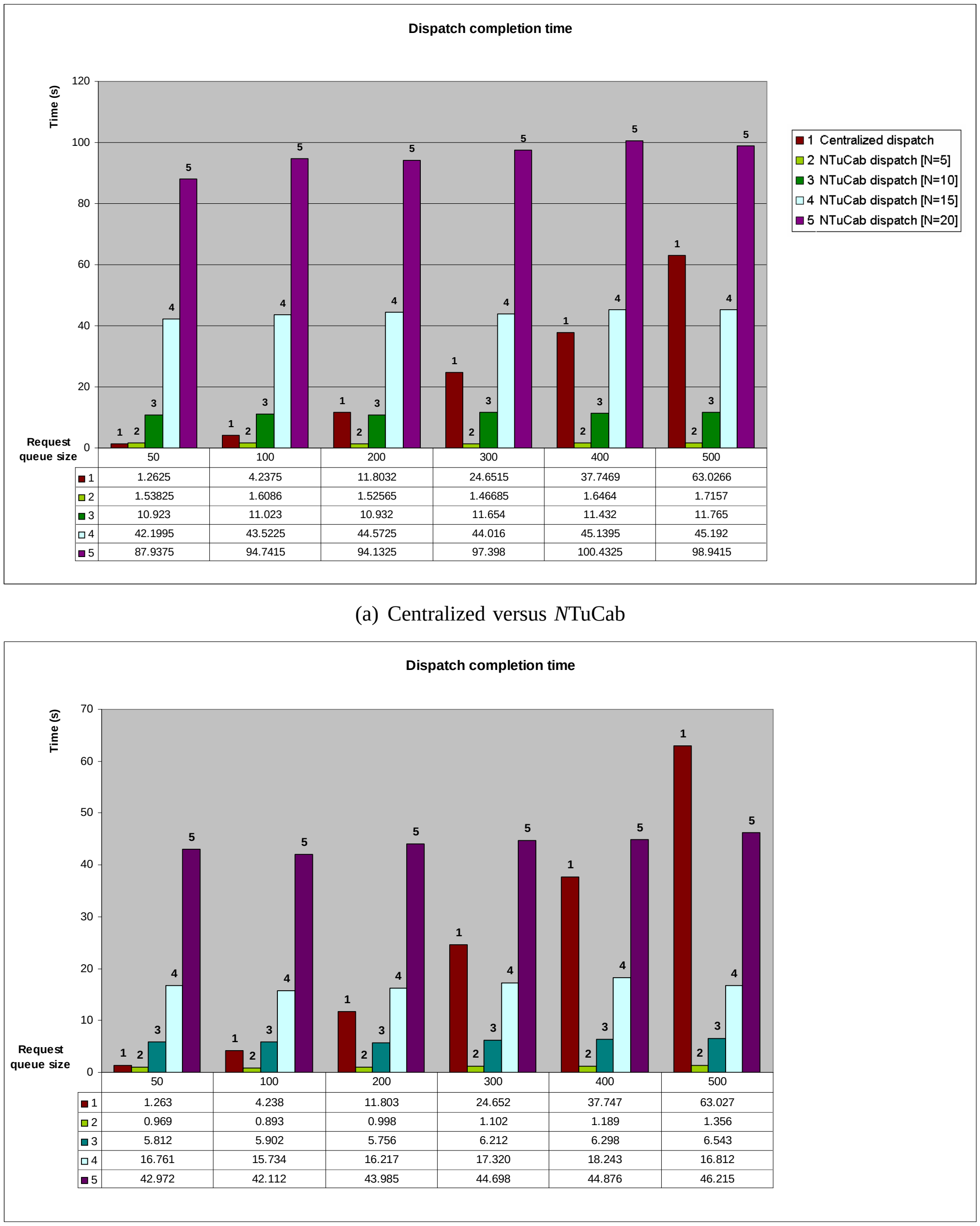
Incorporating the speedup heuristic, the negotiation time, and hence the dispatch completion time, decreases at every demand rate and for all N-group sizes considered, with improvement (or reduction) of over 50% at queue size 100 for $N = 5$, and is maintained at bigger queue sizes from some N-group size onwards [see Table I(b)]. Comparing Figs. 8(a) and 8(b), we observe that with the speedup heuristic, a larger N can outperform centralized dispatch starting from a smaller demand queue size. For instance, $N = 5$ now starts outperforming centralized dispatch from queue size 50 (instead of 100) whereas $N = 15$ now starts from queue size 300 (instead of 500); and at each queue size, we can set a N-group size that outperforms centralized dispatch [see Table I(b)].

2) On Operational Efficiency: Fig. 9 shows that under *N*TuCab dispatch, at each demand rate and for a small $N = 5$, the customer waiting time is longer. This is due to less efficient assignments as some better positioned taxis for the requests might not be in the taxi groups that negotiated for them. As N increases, initially, the customer waiting time becomes shorter due to increase in grouping efficiency, but at $N = 20$, especially at higher demand rates, it starts increasing due to offset to the gains in grouping efficiency, namely, longer negotiation time for a bigger N, and service demand outstripping taxi supply at higher demand rates.

Comparing Figs. 9(a) and 9(b), we observe that incorporating the speedup heuristic mitigates the problem of negotiation time, shortening customer waiting time for all demand rates and N-group sizes.

Fig. 10 shows that, regardless of the group size N, under *N*TuCab dispatch, as the demand rate increases, empty cruising time shortens. This implies that taxis roam less frequently without customers onboard. The empty cruising time converges approximately to the N-group negotiation time, with the roaming time without request negotiation approaching zero. Comparing Figs. 10(a) and 10(b), we observe that with the speedup heuristic, group negotiation, and hence empty cruising time, shortens for all demand rates and N-group sizes considered.

At each demand rate (except when it is 1 for $N = 5$, and without applying speedup heuristic), *N*TuCab dispatch outperforms centralized dispatch, with good improvements in customer waiting time of up to 33.1% at demand rate 4 for $N = 15$ [see Table II(a)], and up to 41.8% at demand rate 2.5 for $N = 20$, when the speedup heuristic is used [see Table II(b)]; and with good improvements in empty cruising time of up to 26.3% at demand rate 1.5 for $N = 20$ [see Table III(a)], and up to 41.2% at demand rate 4 for $N = 20$, when the speedup heuristic is used [see Table III(b)].



(a) Centralized versus *N*TuCab

(b) Centralized versus *N*TuCab with H-Max

Fig. 8. Dispatch efficiency

TABLE I

DISPATCH COMPLETION TIME: IMPROVEMENTS (IN %) OF *N*TUCAB OVER CENTRALIZED DISPATCH

(a) Without H-Max

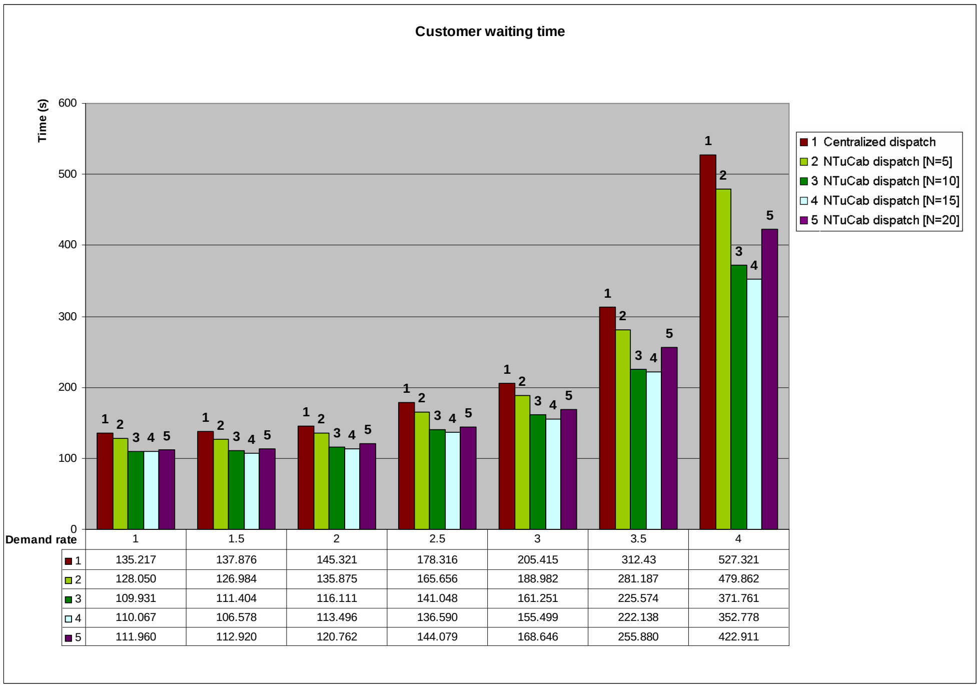
	Demand queue size					
N	50	100	200	300	400	500
5	-21.8	62.0	87.1	94.0	95.6	97.3
10	-765.2	-160.1	7.4	52.7	69.7	81.3
15	-3242.5	-927.1	-277.6	-78.6	-19.6	28.3
20	-6865.3	-2135.8	-697.5	-295.1	-166.1	-57.0

(b) With H-Max

	Demand queue size					
N	50	100	200	300	400	500
5	23.3	**78.9**	**91.5**	**95.5**	**96.9**	**97.8**
10	-360.3	-39.3	**51.2**	**74.8**	**83.3**	**89.6**
15	-1227.6	-271.3	-37.4	29.7	**51.7**	**73.3**
20	-3303.7	-893.8	-272.7	-81.3	-18.9	26.7

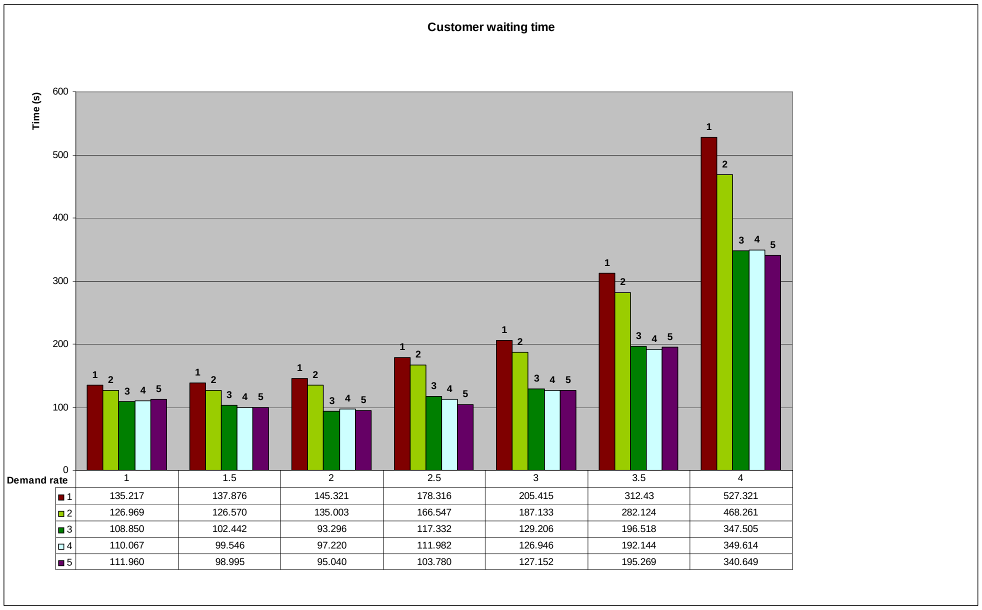
IV. CONCLUSIONS

The new idea of multiagent collaborative assignment of current bookings for automating distributed taxi dispatch is introduced in this paper. The proposed *N*TuCab dispatch system realizes the idea using MA^3-LM through a proposed multiagent architecture (see Section II-B). Using TM2S-MITSIMLab simulations on an urban road network model (Fig. 7), we evaluated the performance of the *N*TuCab dispatch system,



Demand rate	1	1.5	2	2.5	3	3.5	4
■ 1	135.217	137.876	145.321	178.316	205.415	312.43	527.321
■ 2	128.050	126.984	135.875	165.656	188.982	281.187	479.862
■ 3	109.931	111.404	116.111	141.048	161.251	225.574	371.761
□ 4	110.067	106.578	113.496	136.590	155.499	222.138	352.778
■ 5	111.960	112.920	120.762	144.079	168.646	255.880	422.911

(a) Centralized versus *N*TuCab



Demand rate	1	1.5	2	2.5	3	3.5	4
■ 1	135.217	137.876	145.321	178.316	205.415	312.43	527.321
■ 2	126.969	126.570	135.003	166.547	187.133	282.124	468.261
■ 3	108.850	102.442	93.296	117.332	129.206	196.518	347.505
□ 4	110.067	99.546	97.220	111.982	126.946	192.144	349.614
■ 5	111.960	98.995	95.040	103.780	127.152	195.269	340.649

(b) Centralized versus *N*TuCab with H-Max

Fig. 9. Operational efficiency: Customer waiting time

TABLE II

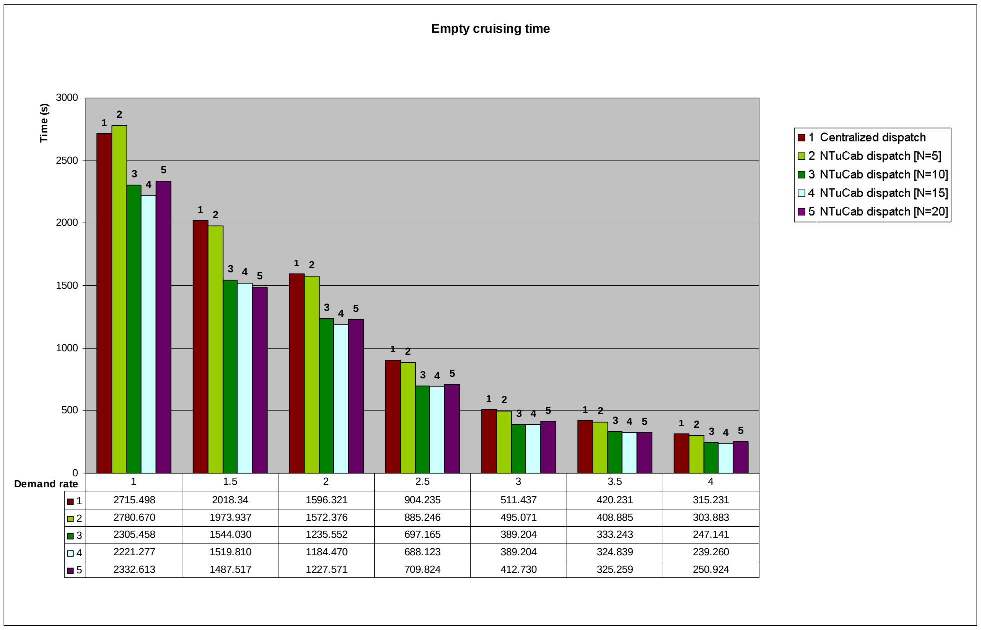
CUSTOMER WAITING TIME: IMPROVEMENTS (IN %) OF *N*TUCAB OVER CENTRALIZED DISPATCH

(a) Without H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	5.3	7.9	6.5	7.1	8.0	10.0	9.0
10	18.7	19.2	20.1	20.9	21.5	27.8	29.5
15	18.6	22.7	21.9	23.4	24.3	28.9	33.1
20	17.2	18.1	16.9	19.2	17.9	18.1	19.8

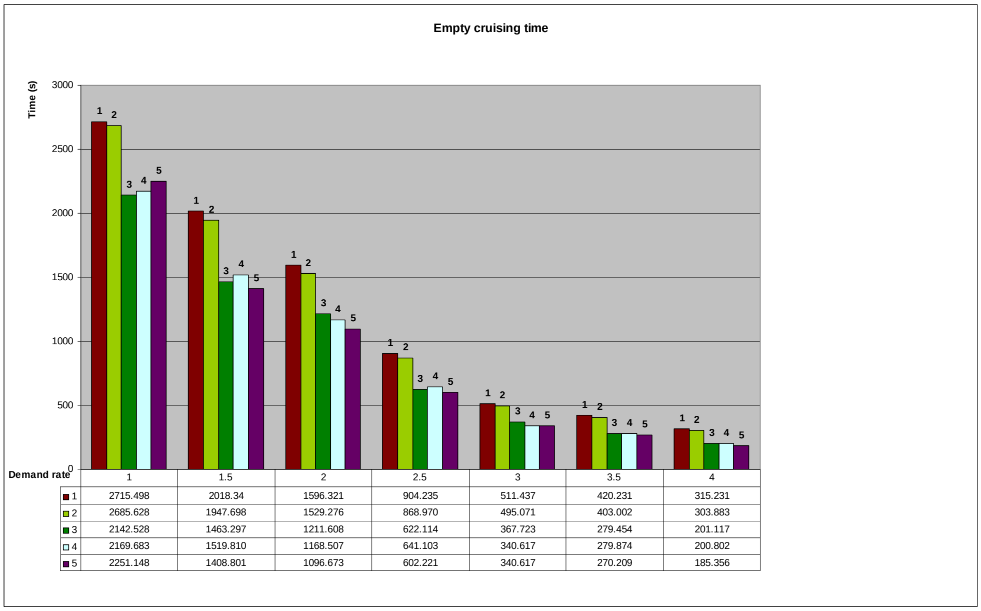
(b) With H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	6.1	8.2	7.1	6.6	8.9	9.7	11.2
10	19.5	25.7	35.8	34.2	37.1	37.1	34.1
15	18.6	27.8	33.1	37.2	38.2	38.5	33.7
20	17.2	28.2	34.6	**41.8**	38.1	37.5	35.4



Demand rate	1	1.5	2	2.5	3	3.5	4
■ 1	2715.498	2018.34	1596.321	904.235	511.437	420.231	315.231
■ 2	2780.670	1973.937	1572.376	885.246	495.071	408.885	303.883
■ 3	2305.458	1544.030	1235.552	697.165	389.204	333.243	247.141
□ 4	2221.277	1519.810	1184.470	688.123	389.204	324.839	239.260
■ 5	2332.613	1487.517	1227.571	709.824	412.730	325.259	250.924

(a) Centralized versus *N*TuCab



Demand rate	1	1.5	2	2.5	3	3.5	4
■ 1	2715.498	2018.34	1596.321	904.235	511.437	420.231	315.231
■ 2	2685.628	1947.698	1529.276	868.970	495.071	403.002	303.883
■ 3	2142.528	1463.297	1211.608	622.114	367.723	279.454	201.117
□ 4	2169.683	1519.810	1168.507	641.103	340.617	279.874	200.802
■ 5	2251.148	1408.801	1096.673	602.221	340.617	270.209	185.356

(b) Centralized versus *N*TuCab with H-Max

Fig. 10. Operational efficiency: Empty cruising time

TABLE III

EMPTY CRUISING TIME: IMPROVEMENTS (IN %) OF *N*TUCAB OVER CENTRALIZED DISPATCH

(a) Without H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	-2.4	2.2	1.5	2.1	3.2	2.7	3.6
10	15.1	23.5	22.6	22.9	23.9	20.7	21.6
15	18.2	24.7	25.8	23.9	23.9	22.7	24.1
20	14.1	26.3	23.1	21.5	19.3	22.6	20.4

(b) With H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	1.1	3.5	4.2	3.9	3.2	4.1	3.6
10	21.1	27.5	24.1	31.2	28.1	33.5	36.2
15	20.1	24.7	26.8	29.1	33.4	33.4	36.3
20	17.1	30.2	31.3	33.4	33.4	35.7	**41.2**

and showed that, even on a basic infrastructure (Fig. 3(b)), the distributed multiagent system approach is promising in terms of significant improvements in both dispatch and operational efficiency over the existing centralized approach.

To approximate the ideal situation **OM-TC** (see Proposition 2) in real-time so as better manage the overall efficiency of *N*TuCab dispatch, future work will include investigating the following issues:

1) Intelligent taxi roaming: Using historical service demand data might lead to better advisory for available taxi drivers to roam more intelligently to match the distributivity of service demand.
2) *N*TuCab dispatch queue pre-processing: Re-grouping based on physical proximity among groups of requests and taxis in the respective queues, prior to the start of every dispatch cycle, might better match the geographic distributivity of service demand to that of the available taxis.

The impetus is to influence and better match the physical distributivity between service demand and non-empty taxi supply in real-time.

In conclusion, in computing the shortest-time paths using real-time traffic conditions [3], the proposed *N*TuCab dispatch system achieves high efficiency, particularly in limiting customer waiting time, provided the demand for taxi service is manageable for a fleet size.

As a final note, the work presented not only provides a complementary and distributed approach to taxi dispatch, but also serves as the first service-automation application of the new agent algorithm MA^3-LM.

Appendix I
Review of MA^3-LM Mechanism [14]

The MA^3-LM mechanism is said to enable $N \geq 2$ agents in group $\mathcal{A}$ to cooperatively negotiate for objects in group $\mathcal{O}$ of the same size, and solve the CLAP. This section reviews the model formalization of the B, D and I knowledge components in an arbitrary negotiation round, as well as the decentralized mediation component. It should be clear that the BDI component definitions below, which are A-QoS data dependent, can be naturally interpreted as an agent's *beliefs*, *desires* and *intentions*. In these definitions, the current object selections of all agents refer to those made under an arbitrary permutation of Π (a one-to-one mapping required by CLAP, as defined in Section II-B.2).

Definition 1 (Belief Set B_i): Given that an agent $a_i \in \mathcal{A}$'s current object selection is $r^i \in \mathcal{O}$. Then its (current) belief set B_i is given by

$$\mathsf{B}_i = \{r \in \mathcal{O} \mid d[a_i, r] > d[a_i, r^i]\} \tag{3}$$

If $\mathsf{B}_i \neq \emptyset$, this means that agent $a_i \in \mathcal{A}$ has at least one alternative object selection $r \in \mathsf{B}_i$ that *may* lead to increase in total A-QoS (see Section II-B.2) when made in exchange with an agent whose current selection is $r \in \mathcal{O}$.

Definition 2 (Desire Set D_i): Given that an agent $a_i \in \mathcal{A}$'s current object selection is $r^i \in \mathcal{O}$ and its belief set is B_i, $\mathsf{B}_i \neq \emptyset$. An arbitrary agent $a_j \in \mathcal{A}$ whose current object selection is $r^j \in \mathcal{O}$ is said to accept agent $a_i \in \mathcal{A}$'s beliefs B_i if $r^j \in \mathsf{B}_i$. To generate the desired exchange options or desires D_i, agent $a_i \in \mathcal{A}$ broadcasts its beliefs B_i and current selection $r^i \in R$, and an arbitrary agent $a_j \in \mathcal{A}$ who accepts the beliefs would respond with a pair of A-QoS values $d[a_j, r^j]$ and $d[a_j, r^i]$, so that for each of the $|\mathsf{B}_i|$ responses received, the corresponding object exchange option $[(a_i, r^j), (a_j, r^i), \rho] \in \mathsf{D}_i$ (i.e., is agent $a_i \in \mathcal{A}$'s desire) if $\rho > 0$, where ρ is defined by

$$\rho = -d[a_i, r^i] + d[a_i, r^j] - d[a_j, r^j] + d[a_j, r^i] \tag{4}$$

If $\rho > 0$, it means that there is a net exchange gain if agent $a_i \in \mathcal{A}$ gives up its current selection $r^i \in \mathcal{O}$ and selects $r^j \in \mathcal{O}$, and in exchange, agent $a_j \in \mathcal{A}$ gives up its current selection $r^j \in \mathcal{O}$ and selects $r^i \in \mathcal{O}$. Thus, any desire $d \in \mathsf{D}_i$, when carried out, will definitely lead to an increase in total A-QoS without violating Π. Quite naturally, it provides the motivation for agent $a_i \in \mathcal{A}$ to want to exchange its current object selection.

Definition 3 (Intention I_i): Given that an agent $a_i \in \mathcal{A}$'s desire set is D_i, $\mathsf{D}_i \neq \emptyset$. Then, agent $a_i \in \mathcal{A}$'s intention I_i is given by

$$\begin{aligned} I_i = [(a_i, r^j), (a_j, r^i), \rho] \in \mathsf{D}_i, &\text{ for which} \\ \rho = \max\{\rho' \mid [-, -, \rho'] \in \mathsf{D}_i\ \}& \end{aligned} \tag{5}$$

Agent $a_i \in \mathcal{A}$'s decisive stance or intention has to be I_i since it is viewed as the best exchange option (in terms of incremental social gain from the agent's perspective) that the agent can propose. It is said to have no intention if either $\mathsf{B}_i = \emptyset$ or $\mathsf{D}_i = \emptyset$, in which case $I_i = nil$, where $nil = [-, -, 0]$.

All the agents' exchange intentions (or the lack thereof communicated as a nil intention) $I_i \in \mathsf{I}$ would need to undergo arbitration to decide which two agents to proceed with the object exchange, before a negotiation round is concluded, and the next round begins. Essentially, an intention $I = [-, -, \rho]$ with the highest exchange gain $\rho > 0$, i.e., one that contributes to the highest increase (in total A-QoS) if carried out, would need to be selected.

The negotiation process will terminate following a negotiation round when all agents have no (more) intention to exchange objects and so submit nil intentions, discovered through arbitration.

In MA^3 [13], the role of arbitration is handled through a dedicated agent in a distributed setting. MA^3-LM [14] uses the same BDI negotiation model but does away with the need for an explicit arbitration agent, through a local mediation procedure among MA^3-LM agents that effectively carries out the arbitration role.

In what follows, we also refer to agent $a_i \in \mathcal{A}$ by its unique identification number i, $0 \leq i \leq N - 1$. In MA^3-LM, each agent $a_i \in \mathcal{A}$ initially selects an object $r \in \mathcal{O}$ according to (a permutation of) Π. Upon activation, all agents begin negotiation concurrently.

The generic BDI assignment (reasoning) mechanism of an agent in an arbitrary round of negotiation can now be described as follows:

MA^3-LM : Decentralized Agent $a_i \in \mathcal{A}$

1) If agent believes that there are alternative object selections which may lead to increase in total A-QoS, it would, based on its (local) beliefs, generate the desired exchange options or desires, from which the best option will be chosen as its intention.
2) Agent invokes LM_Com(.).
3) Concurrent with Step 1 and Step 2, it responds to any requesting agent whose beliefs it accepts, by sending to the requesting

```
     agent the A-QoS values as required for computing the requesting
     agent's desire.
```

LM_Com(.) in Step 2 realizes the mediation procedure which follows next.

Let $\mathbb{Z}_+$ denote the set of positive integers, $\mathbb{N} = \{0, 1, 2, \cdots, N-1\}$ and $i \in \mathbb{N}$ be the unique identification (id) number of an agent.

Define $X\_Init : \mathbb{Z}_+ \rightarrow \mathbb{N}$, a deterministic function with $X\_Init(n) = m$ read as 'object exchange initiator for negotiation round n is agent m'. This is a common function used by all agents $a_i \in \mathcal{A}$ for dynamic assignment of the initiation role. An example of $X\_Init(n)$ is $(n \mod N)$. Initially, $n = 1$, and is incremented upon the start of every successive round.

MA^3-LM : LM_Com(.) for Agent $a_i \in \mathcal{A}$

```
1) If $i = X\_Init(n)$, it
   a) sends its intention (or the lack thereof) to agent $(i+1) \mod N$;
   b) receives a mediated intention (or the lack thereof) from agent $(N+i-1) \mod N$.
      i) If there is no mediated intention, it terminates the negotiation by informing all agents to quit.
      ii) Otherwise,
         A) if it is one of the two agents associated with the final mediated intention, it
            • will change its object selection as intended, and instruct the other agent to proceed with the object selection change;
            • receives acknowledgement of object exchange made as instructed (from the other agent concerned), before informing all agents to proceed to next round of negotiation;
         B) else it
            • instructs, according to the final mediated intention, the two agents concerned to proceed with the object exchange;
            • receives acknowledgement of object exchange made as instructed (from the two agents concerned), before informing all agents to proceed to next round of negotiation.
2) Else it
   a) first receives a mediated intention (or the lack thereof) from agent $(N+i-1) \mod N$;
   b) then mediates by comparing it with its own and selecting one intention, which it sends to agent $(i+1) \mod N$; and
   c) will change its object selection if instructed (and then acknowledges the agent, say $a$, instructing the change) and/or proceeds to next round of negotiation or quit when as instructed by agent $a$.
```

Important properties of MA^3-LM include it not incurring higher reasoning (i.e., computation and communication) complexity than MA^3, and always reaching a final solution in at least one round of negotiation [13], [14].

Appendix II
Assignment Initialization Heuristic

The negotiation speed of MA^3-LM depends on the initial assignment. In attempting to hasten the negotiation, a simple heuristic (labelled H-Max) is proposed and presented herein. Intuitively, the heuristic attempts to set the initial (assignment) solution, with each agent selecting, as far as it is possible, a self-optimal but different object to begin negotiation with. Logically, it can be said to contribute one negotiation round but the overall process can become faster. The details are as follows:

Heuristic H-Max*:* One of the agents in group $\mathcal{A}$ is designated as an (initialization) arbitration agent, and this fact is made known to all agents at the outset. Every agent $a_i \in \mathcal{A}$ selects and proposes to the arbitration agent an object $r_{m_i} \in \mathcal{O}$ that satisfies the following:

$$d[a_i, r_{m_i}] \quad = \quad \max\{d[a_i, r] | \ r \in \mathcal{O}\ \} \tag{6}$$

The arbitration agent, upon receiving all such proposals, will do the following:

1) For each object $r \in \mathcal{O}$,
   - it will approve an agent's proposal (i.e., object selection) if it is the only agent selecting object $r \in \mathcal{O}$;
   - if two or more agents propose to select object $r \in \mathcal{O}$, then it will approve the proposal by an agent that offers the highest A-QoS value among them.
2) Assume that F agents, $F \geq 1$, have had their proposals approved. Then if $(N - F) \geq 1$, it will arbitrarily allocate the remaining $(N - F)$ agents, each with a different object from the $(N - F)$ unselected objects.

Acknowledgements

The authors would like to thank the researchers of the Intelligent Transportation Systems Program at the Massachusetts Institute of Technology (MIT), USA, for making freely available the MITSIMLab simulation software (at `http://web.mit.edu/its/mitsimlab.html`, as of Jan 2007) that we adapted as the environment for our experiments.

The authors also acknowledge the technical support of Mrs Siew Lai Ng-Goh, Irene from the Parallel and Distributed Computing Center (PDCC), School of Computer Engineering, Nanyang Technological University (NTU), Singapore, in installing the MITSIMLab simulator on a Unix workstation.

References

[1] P. H. Paul Chan, *Needed: Hard look at taxi efficiency*. Today Newspaper, MediaCorp Press, Singapore, March 2, 2007.

[2] Z. Liao, "Real-time taxi dispatching using global positioning systems," *Communications of the ACM*, vol. 46, no. 5, pp. 81–83, May 2003.

[3] D. H. Lee, H. Wang, R. L. Cheu, and S. H. Teo, "A taxi dispatch system based on current demands and real-time traffic information," *Transportation Research Record*, vol. 1882, pp. 193–200, 2004.

[4] J. S. Rosenschein and G. Zlotkin, *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. Cambridge, M.A, USA: The MIT Press, 1994.

[5] R. A. Belecheanu, S. Munroe, M. Luck, T. Payne, T. Miller, P. McBurney, and M. Pechoucek, "Commercial applications of agents: Lessons, experiences and challenges," in *Proceedings of the Fifth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'06)*, Hakodate, Japan, July 2006, pp. 1549–1555.

[6] S. Kim, M. E. Lewis, and I. Chelsea C. White, "Optimal vehicle routing with real-time traffic information," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 178–188, June 2005.

[7] N. Papadoglou and E. Stipidis, "Investigation for a global AVL system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 2, no. 3, pp. 121–126, September 2001.

[8] Y. Zhao, “Mobile phone location determination and its impact on intelligent transportation systems,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 1, pp. 55–64, March 2000.

[9] V. de Nitto Personé and V. Grassi, “Performance analysis of caching and prefetching strategies for palmtop-based navigational tools,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 1, pp. 23–34, March 2003.

[10] F. Bellifemine, A. Poggi, and G. Rimassa, “JADE: a FIPA2000 compliant agent development environment,” in *AGENTS’01: Proceedings of the Fifth International Conference on Autonomous Agents*. New York, NY, USA: ACM Press, 2001, pp. 216–217, software downloadable from http://jade.tilab.com/ as of Dec 2006.

[11] H. W. Kuhn, “The Hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, vol. 2, no. -, pp. 83–97, 1955.

[12] R. E. Burkard and E. Cela, “Linear assignment problems and extensions,” in *Handbook of Combinatorial Optimization, Vol.4*, P. M. Pardalos and D. Z. Du, Eds. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999, pp. 75–149.

[13] K. T. Seow and K. Y. How, “Collaborative assignment : A multiagent negotiation approach using BDI concepts,” in *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS’02)*. Palazzo Re Enzo, Bologna, Italy: ACM Press, July 2002, pp. 256–263, extended version to appear as a 2006 Technical Report (# TR-IIS-06-013) archived at the Institute of Information Science, Academia Sinica, Taiwan.

[14] K. T. Seow and K. M. Sim, “Decentralized assignment reasoning using collaborative local mediation,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 11, pp. 1576–1580, November 2006.

[15] Q. Yang, H. N. Koutsopoulos, and M. E. Ben-Akiva, “A simulation laboratory for evaluating dynamic traffic management systems,” *Transportation Research Record*, vol. 1710, pp. 122–130, 2000.

[16] M. Ben-Akiva, H. N. Koutsopoulos, and Q. Yang, “User’s Guide for MITSIMlab and Road Network Editor (RNE),” ITS Program at the Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Boston, USA, Technical Report Draft, November 2002, 116 pages.