A Collaborative Multiagent Taxi-Dispatch System

Kiam Tian Seow, Nam Hai Dang and Der-Horng Lee

***Abstract*— This paper presents a novel multiagent approach to automating taxi dispatch that services current bookings in a distributed fashion. The existing system in use by a taxi operator in Singapore and elsewhere attempts 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 satisfaction. To realize the system, a multiagent architecture is proposed, populated with software collaborative agents that can actively negotiate on behalf of taxi drivers in groups of size N for available customer bookings. Theoretically, an analysis of the boundary and optimal multiagent taxi-dispatch situations is presented along with a discussion of their implications. Experimentally, the operational efficiency of the existing and proposed dispatch systems was evaluated through computer simulations. The empirical results, obtained for a 1000-strong taxi fleet over a discrete range of N, show that the proposed system can dispatch taxis with reduction in customer waiting and empty taxi cruising times of up to 33.1% and 26.3%, respectively; and up to 41.8% and 41.2% reduction when a simple negotiation speedup heuristic was applied.**

***Note to Practitioners*— With the liberalization of the taxi industry in Singapore and elsewhere, 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. Our research is motivated by the increasing 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 customer waiting and empty taxi cruising times, when both leverage on real-time traffic information for shortest-time path computation over a road network as proposed in [2]. Additionally, NTuCab dispatch can be implemented on an existing technological infrastructure, providing the opportunities to harness the existing power of multiple intelligent transportation systems technologies. A more efficient dispatch system can help maintain a higher standard of customer service vis-à-vis human driver satisfaction, especially when the demand for taxi service is manageable for the fleet size. In future research, 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 available taxi supply in real-time.**

A preliminary conference version of this paper appeared in [1].

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

N.H. Dang is with EON Experience Lab, EON Reality Pte Ltd, Republic of Singapore 079909. dang0001@ntu.edu.sg.

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

***Index Terms*— Taxi Dispatch, Service Automation, Automated Software Agents, Intelligent Paratransit Transportation.**

I. Introduction

Taxis are a convenient means of paratransit 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 dispatch system; the system utilizes a Global Positioning System (GPS) to automatically locate taxis in real-time [3], [4].

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 [2]. 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-served 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 will 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 computed using real-time traffic information [2], 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-served basis. So the dispatcher would have to attend to request 1 first,

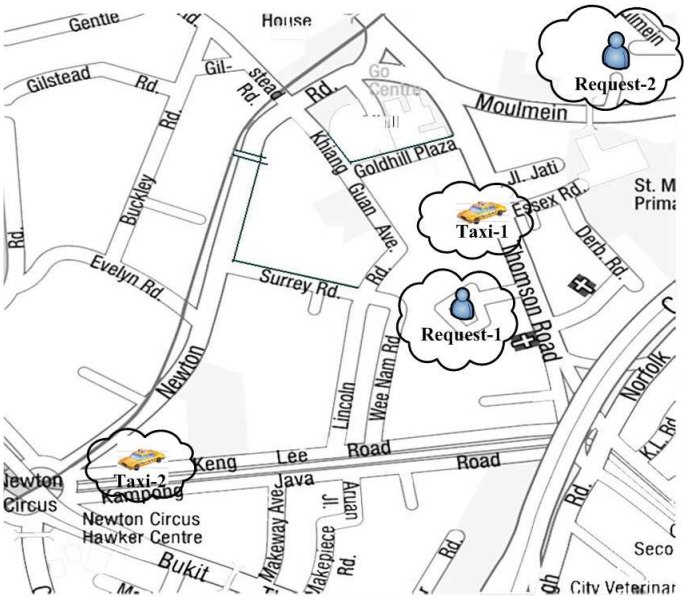


Fig. 1. A taxi dispatch scenario

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 satisfaction. And it must be implementable on an existing technological infrastructure. With this philosophy in mind, we propose a multiagent approach deploying *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 [5], several taxi agents can collaboratively search for an assignment solution that they jointly agree.[1]

Intelligent agents and multiagent approaches are gaining increasing prominence in automation research, and have been successfully formulated or applied in the domains of manufacturing (e.g., [6]), warehousing (e.g., [7]), product family design (e.g., [8]) and hierarchical decision-making protocols (e.g., [9]), to name a few. Although using multiple automated agents is also not new in intelligent service transportation, as seen in transport logistics [10] and route guidance [11], 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 acceptance or refusal 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 systems (ITS) technologies in, for example, vehicle routing [12], automatic vehicle location [13], mobile phone location determination [14] and palmtop-based navigation [15], exploiting the huge investments already made in the internet, wireless communication and mobile devices, as well as 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 and solution 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

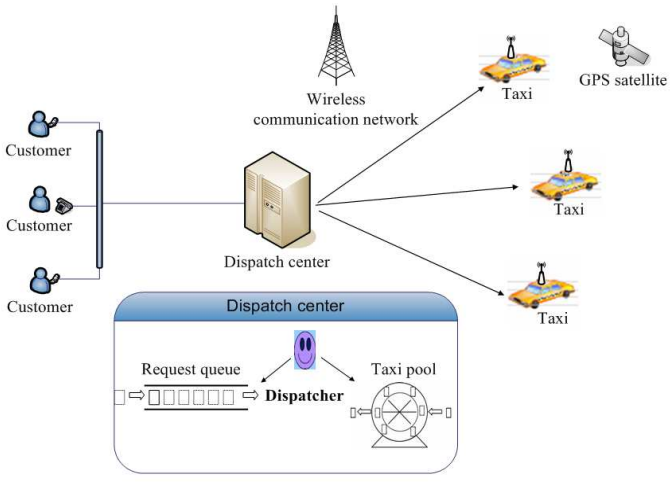
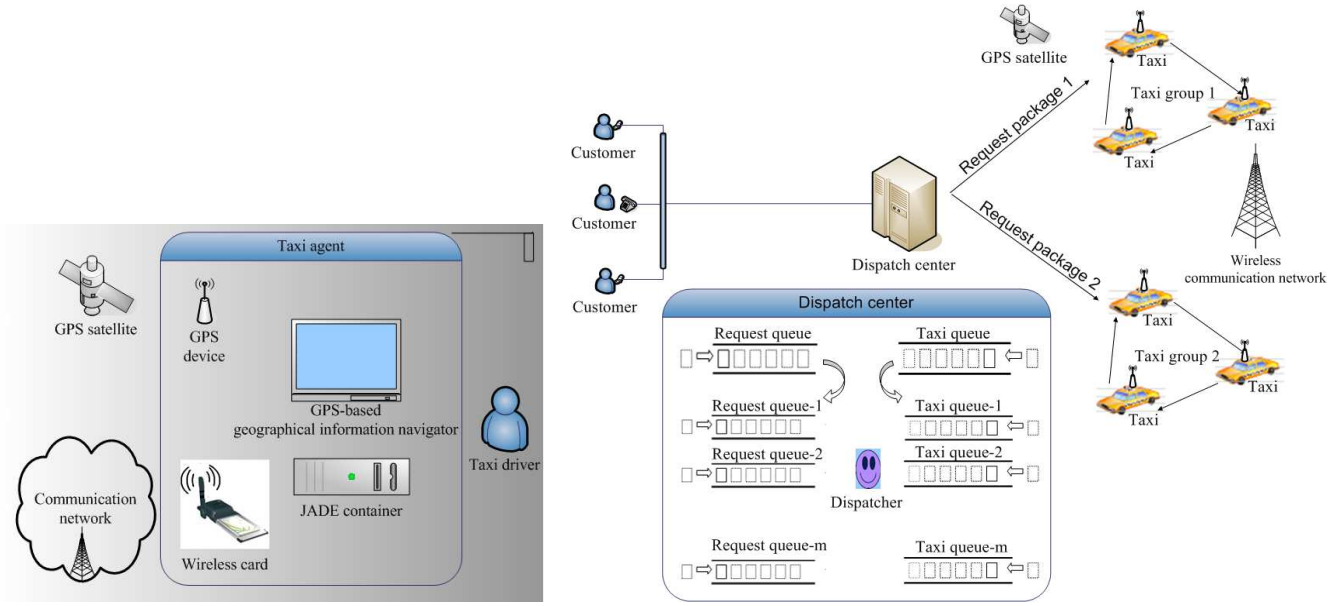


Fig. 2. Current state-of-the-art taxi dispatch system: Centralized architecture

Fig. 2 depicts the basic architecture of a current taxi dispatch system in use by a taxi operator in Singapore and elsewhere. Incoming taxi service requests are queued on a first-come, first-served basis at the dispatch center. For each customer

[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 and elaborated.



(a) An in-vehicle agent infrastructure

(b) Collaborative multiagent architecture

Fig. 3. Proposed *N*TuCab dispatch system: Non-centralized

(online booking) request, available taxis in the vicinity of the customer pick-up location are considered, and a taxi among them is dispatched to service it, upon the human driver acknowledging acceptance of the booking job. It is empirically confirmed that an efficient way of dispatch is to assign a nearby taxi that can traverse the shortest-time path to the customer pick-up location, computed using real-time traffic information over a road network [2].

In general, it is not feasible to compute the (locally) shortest-time path for each of a possibly large number of available taxis nearby a customer location, since determining such a path requires a considerably significant amount of computation time. Thus, it has been assumed [2] that a taxi with the shortest-time path is found among a limited number (say, 20) in the vicinity that are the nearest in terms of their shortest straight-line distances to the customer location .

As a remark, besides [2] and a related but different effort that empirically examines the performance improvement of using GPS in taxi dispatch operations [3], there is apparently little work in the published literature on automated taxi dispatch. This is despite its importance and commercial interest.[2] Henceforth, for comparison purposes, we shall assume that the existing (state-of-the-art) taxi dispatch system available employs the centralized architecture of Fig. 2 and applies the dispatch method proposed in [2]. In our opinion, this system should compare favorably with the centralized dispatch system provided by a market leader. According to the website (`www.cordic.com`), the commercial system associates every taxi service request with a taxi queue according to its customer pick-up location, and then offers it to the first taxi in that queue.

[2]Examples of commercial companies delivering automated taxi dispatch solutions include Cordic, DDS Digital Dispatch and Mobisoft.

B. Proposed Taxi Dispatch System

As suggested in the introduction, multiagent collaboration might leverage on the improved dispatch method [2] by exploiting the mutual assignment exchange effects among multiple taxis awaiting multiple requests - a scenario that is not uncommon. Towards this end, an infrastructure[3] 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) 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) it is not in negotiation, (ii) the taxi has entered (and is currently in) the new area,[4] and (iii) the passenger

[3] We recommend JADE [16] as the agent technology development package for MA^3-LM agent software implementation. In principle, however, any other agent development package that supports communication may be used.

[4]That an area i is *new* means that, operationally, the taxi agent has not notified the dispatch center to insert its taxi record into the taxi queue i (that otherwise does not contain the record).

has just alighted from the taxi or the taxi is empty, with no committed taxi request to service next. 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 taxi queues, taking the information from a common taxi queue that is 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, into the taxi queue y.

b) **DMT-B**: Insert the records of taxi bookings into the respective request queues, taking the information from a common request queue that is continually accepting customer taxi bookings. If a taxi request is from an area i, insert the record, on a first-come, first-served basis, into the request queue i.

c) **DMT-C**: Delete the records of a taxi and the negotiated assignment that the taxi has accepted; and communicate the service information of the taxi to the assigned customer. 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 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) [17]. 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

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

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 [18]. Using agent modeling ideas, recent research has developed agent mechanisms [19], [20] 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 [18]. In essence, these taxi agents can compute and leverage on the possible overall A-QoS increments, achievable through properly reassigning requests among themselves.

One important development for CLAP is a decentralized agent algorithm called MA^3-LM [20], 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 propose the first potential real-world service-automation

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. All the agents begin negotiation with an initial selection made under an arbitrary permutation of Π (1) (the one-to-one mapping). For taxi dispatch, every taxi agent in group $\mathcal{A}$ negotiates with the other agents in the same group over the requests in $\mathcal{O}$. The purpose is to decide which two agents are to exchange their current request selections in every consecutive round. This is done by collaborative reasoning per round, during which every agent determines its individual request exchange *desires* and resulting exchange *intention* (its best desire), based on its computed *beliefs* of exchange alternatives that may increase the group's total A-QoS, S_{tot} (2).

Definition 1 (Belief Set B_i*):* Given that an agent $a_i \in \mathcal{A}$'s current request 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 request selection $r \in \mathsf{B}_i$ that *may* lead to increase in total A-QoS S_{tot} (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 request 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 request 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 $r^i \in \mathcal{O}$, 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 request 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 request 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 net exchange 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 request 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. In so doing, all the agents perform local mediation to arbitrate their intentions. This involves incremental arbitration of intention as the better intention (i.e., the one with higher net exchange gain) - between a taxi agent's own and the intention it received - is passed by the agent to another in a circular fashion, before finally reaching a taxi agent that is concurrently and dynamically entrusted the role of a request selection-exchange initiator for a particular negotiation round [20]. This process differs from that in the original mechanism called MA^3 [19], which uses the same BDI reasoning model but has an extra agent dedicated to receiving and arbitrating all agent intentions. The negotiation process will terminate following a negotiation round when all agents have no (more) intention to exchange request selections and so submit nil intentions, discovered through arbitration by local mediation.

The relevant algorithmic details for MA^3-LM agents [20] based on the formal description above is generically summarized in [1, p. 1050]. MA^3-LM agents always terminate with an often highly efficient but not necessarily optimal solution after a finite number of negotiation rounds [19], [20].

To reduce the number of negotiation rounds in the negotiation process, an assignment initialization heuristic (labelled H-Max) can be applied prior to negotiation, the details of which are documented in the appendix. And to significantly reduce the average communication time per negotiation round in practice, a faster version called non-redundant BDI reasoning discussed in [21, p. 685] for MA^3 can be easily adapted to MA^3-LM agents.

In our current work, 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 MA^3-LM agent $a_i \in \mathcal{A}$ represents) to travel from its designated or current location to the pick-up location of a customer (who initiated the request $r_j \in \mathcal{O}$); the negation is so that the MA^3-LM agents, in maximizing the total A-QoS S_{tot} (2), are minimizing the total travel time of their taxis. This real-time travel period is that computed over a directed road network with real-time traffic information [2]. The smaller this time value is, the nearer to the customer the taxi is said to be.

3) Decentralized versus Centralized - A Discussion: The *N*TuCab dispatch system architecture deliberately avoids using a centralized (single-agent) algorithm [18] for assigning requests to taxi agents, thus removing a processing bottleneck and a crucial 'single point of failure'. The A-QoS information is already distributed among the taxi agents, so it is wasteful to have to replicate by centralized information gathering or computation, required if single-agent assignment processing is used instead. In decentralized MA^3-LM processing, multiple taxi agents share their local A-QoS information only as needed when they negotiate for a request. In practice, the proposed architecture can even allow these MA^3-LM taxi agents to flexibly respond on-the-fly (i.e., during negotiation) to A-QoS data invalidation, by their independently correcting their local

A-QoS values as necessary before the start of each negotiation round. Such data invalidation can occur due to, say, sudden changes detected in road traffic conditions. In this practical situation, the centralized algorithm, on the contrary, would have to recompute from scratch the assignment solution, after data validation through recomputing or re-gathering the A-QoS values from the agents involved; and this could possibly incur even higher processing and communication overheads. Finally, the proposed system can also be readily extended to degrade more gracefully when some of the taxis or taxi agents are out of service temporarily.

4) Theoretical Boundary & 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.

	r_0	r_1	r_2
a_0	x_h	x_h	x_h
a_1	y_h	y_h	y_h
a_2	z_h	z_h	z_h

(a) For **DT-CC**

	r_0	r_1	r_2
a_0	x_v	y_v	z_v
a_1	x_v	y_v	z_v
a_2	x_v	y_v	z_v

(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 $r_j \in \mathcal{O}$.

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 the least negative travel time) of an agent $a_i \in \mathcal{A}$ for a different request.

	r_0	r_1	r_2
a_0	$-$	$-$	x_0^2
a_1	$-$	x_1^1	$-$
a_2	x_0^0	$-$	$-$

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

Following, we present two propositions on MA^3-LM in the situations identified. The proofs entail an understanding of the formal definitions of Belief (B), Desire (D) and Intention (I), and the termination condition of MA^3-LM. For the proof of Proposition 2, the reader is also referred to the appendix 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 maximal) total A-QoS assignment which is arbitrary.

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, any collaborative assignment reached is associated with the same set of A-QoS values, $\{d[a, \Pi(a)] \mid a \in \mathcal{A}\}$, and thus it has 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 will incur the minimum number of negotiation rounds [19], [20]. Despite the quick decision, negotiation is not needed in such situations since every assignment is optimal.

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 maximal). Hence the proposition. ■

Proposition 2 suggests that situation **OM-TC** characterizes the ideal condition to be in, where a relatively quick negotiation will result in an assignment that is both locally and globally optimal. By local optimality, every taxi agent $a \in \mathcal{A}$ selects a request $r \in \mathcal{O}$ for which the A-QoS $d[a, r]$ among all requests in $\mathcal{O}$ is the largest; and by global optimality, we mean that the total A-QoS of the assignment is maximized.

From the above analysis and discussion, we infer that if the working scenarios are exclusively **DT-CC** or **CT-DC**, as could be reasonably assumed in the past, the proposed taxi dispatch approach (see Fig. 3) might only perform as well as the current approach (see Fig. 2). However, in a situation which is in between **DT-CC** or **CT-DC** and **OM-TC**, we envisage that the proposed distributed approach can often significantly

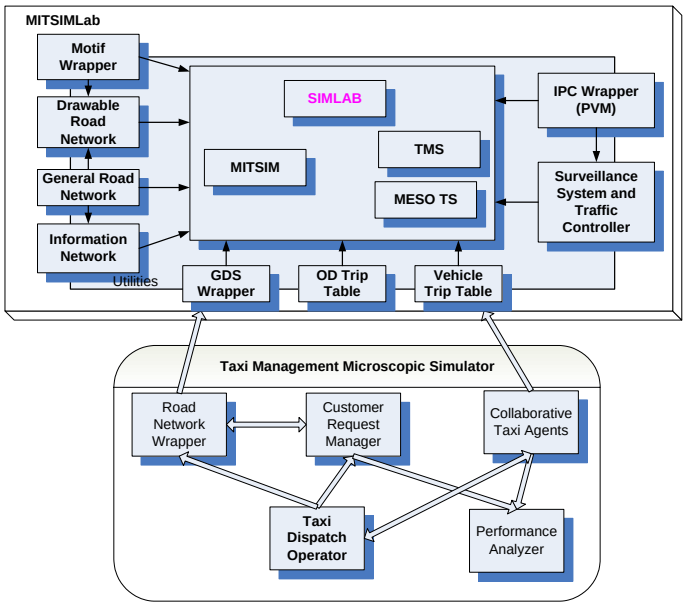


Fig. 6. The TM2S-MITSIMLab simulation model

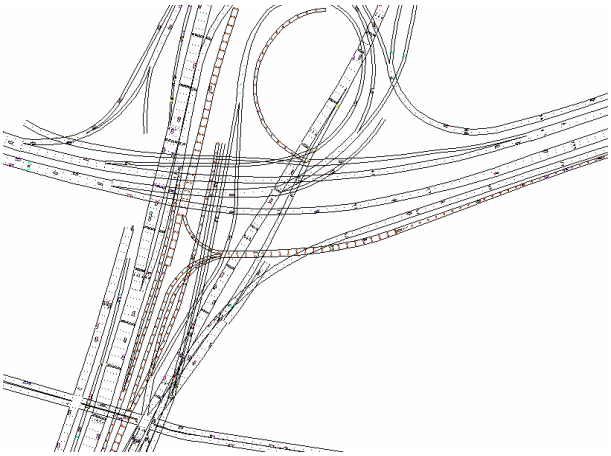


Fig. 7. The urban road network model used

outperform the existing centralized one. With the pervasive use of mobile phones and the associated technologies in cities like Singapore, where one can conveniently book a taxi from almost anywhere, the demand by individual customers for taxi service has set to become more geographically distributed. This emerging real-world scenario, along with the induced necessity for empty taxis to roam about more frequently, means that such in-between situations could become a common occurrence, and it is no longer acceptable to simply assume situation **DT-CC** or **CT-DC**. All these are suggestive of a better performance using our proposed approach, and this is confirmed by simulations.

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, simulating taxi operations in a selected ITS-managed urban road network of reasonable complexity. We focused on operational efficiency. For both the systems, this was investigated in terms of customer waiting time versus empty cruising time. In our simulations, *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.

A. Experimental Scope & Investigation

The experiments were performed on MITSIMLab [22], [23], [24], 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 MITSIMLab simulation software (available from `web.mit.edu/its/mitsimlab.html`) is an existing simulation-based laboratory originally developed for evaluating traffic management system designs at the operational level. It was adapted as the environment for all the experiments, along with a urban road network model built by MITSIMLab developers using a road network editor [23]. The urban road network model used is shown in Fig. 7; it covers a physical area of about $15km \times 10km$. In providing the required real-time traffic information for dispatch operation, an abstracted road network was also built over the MITSIMLab-based urban road network model.

The TM2S module wraps around the MITSIMLab to simulate the real time activities of a dispatch operator and the associated MA^3-LM 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 TM2S module assumes that the taxi agents negotiate over a high-speed wireless communication network. In calculating the multiagent negotiation time, the module abstracts away the underlying detailed communication costs and instead estimates the total negotiation time based on the number of negotiation rounds taken and a conservative estimate of $0.2N$ seconds per round for an N-group of taxi agents. The estimate is the longest average time per negotiation round that we found at $N = 20$, when we experimented separately for groups of $N \in \{5, 10, 15, 20\}$, by running a Java program of MA^3-LM implemented on a multiagent platform called JADE [16] in a local area network (emulating the high-speed communication infrastructure), and without applying the speedup heuristic, H-Max.

In the MITSIMLab environment, the taxis move randomly in the road network. The human drivers of available taxis are assumed to always accept and service any taxi request negotiated by their agent. Upon the taxi agents in TM2S 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.

For both the centralized and proposed systems, we computed the travel times using real-time traffic information as proposed in [2]. As defined, the A-QoS $d[a, r] < 0$ denotes

the negation of the expected shortest travel time for a taxi (represented by an agent $a \in \mathcal{A}$) to move from its current location to the pick-up location of a customer (who initiated the pending request $r \in \mathcal{O}$); since taking the shortest-time path is a time-efficient way to service taxi requests [2]. The A-QoS formula used is

$$d[a, r] = -\min_{\text{all } i}\{T_i\} \tag{6}$$

T_i denotes the expected real-time for the taxi to travel on an arbitrary path i through the road network connecting the two locations. The expected shortest travel time $\min_{\text{all } i}\{T_i\}$ was calculated using the route choice model provided by the traffic flow simulator module of MITSIMLab. 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). Along with the determination of the associated shortest-time path for a taxi, it is essentially a vehicle route planning problem, solved using the techniques developed for MITSIMLab [24, Appendix B].

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.

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 $N \in \{5, 10, 15, 20\}$. The experiments were repeated for collaborative agent dispatch with speed-up initialization heuristic incorporated (see Appendix).

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

B. Analysis of Numerical Results

Table I 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 for a bigger $N = 20$, 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 Tables I(a) and I(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.

TABLE I

OPERATIONAL EFFICIENCY OF *N*TUCAB AND CENTRALIZED DISPATCH: CUSTOMER WAITING TIME (IN S)

(a) Without H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	128.1	127.0	135.9	165.7	189.0	281.2	479.9
10	109.9	111.4	116.1	141.0	161.3	225.6	371.8
15	110.1	106.6	113.5	136.6	155.5	222.1	352.8
20	112.0	112.9	120.8	144.1	168.6	255.9	422.9
Ⓒ	135.2	137.9	145.3	178.3	205.4	312.4	527.3

(b) With H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	127.0	126.6	135.0	166.5	187.1	282.1	468.3
10	108.9	102.4	93.3	117.3	129.2	196.5	347.5
15	110.1	99.5	97.2	112.0	126.9	192.1	349.6
20	112.0	99.0	95.0	103.8	127.1	195.3	340.6

Ⓒ: Centralized dispatch

TABLE II

CUSTOMER WAITING TIME: REDUCTIONS (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

Table III 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 Tables III(a) and III(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 reductions 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 reductions in empty cruising time of up to 26.3% at demand rate 1.5 for $N = 20$ [see Table IV(a)], and up to 41.2% at demand rate 4 for $N = 20$, when the speedup heuristic is used [see Table IV(b)].

TABLE III
OPERATIONAL EFFICIENCY OF *N*TUCAB AND CENTRALIZED DISPATCH: EMPTY CRUISING TIME (IN S)

(a) Without H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	2780.7	1973.9	1572.4	885.2	495.1	408.9	303.9
10	2305.5	1544.0	1235.6	697.2	389.2	333.2	247.1
15	2221.3	1519.8	1184.5	688.1	389.2	324.8	239.3
20	2332.6	1487.5	1227.6	709.8	412.7	325.3	250.9
Ⓒ	2715.5	2018.3	1596.3	904.2	511.4	420.2	315.2

(b) With H-Max

	Demand rate						
N	1	1.5	2	2.5	3	3.5	4
5	2685.6	1947.7	1529.3	869.0	495.1	403.0	303.9
10	2142.5	1463.3	1211.6	622.1	367.7	279.5	201.1
15	2169.7	1519.8	1168.5	641.1	340.6	279.9	200.8
20	2251.1	1408.8	1096.7	602.2	340.6	270.2	185.4

Ⓒ: Centralized dispatch

TABLE IV
EMPTY CRUISING TIME: REDUCTIONS (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**

IV. CONCLUSION

For automating distributed taxi dispatch, this paper has introduced the new idea of multiagent collaborative assignment of current bookings. The proposed *N*TuCab dispatch system realizes the idea using MA^3-LM through a proposed multiagent architecture (see Section II-B). Theoretical boundary (**DT-CC** and **CT-DC**) and optimal (**OM-TC**) situations are analyzed (see Propositions 1 and 2, respectively) and discussed. Using TM2S-MITSIMLab simulations on an urban road network model (Fig. 7), we evaluated the performance of the *N*TuCab dispatch system, and showed that, even on a basic infrastructure [Fig. 3(b)], the distributed multiagent system approach is promising in terms of significant improvements in operational efficiency over an existing centralized approach (see Section II-A). We also showed that incorporating a simple negotiation speedup heuristic (see Appendix) could raise the efficiency further.

To approximate more towards the ideal situation **OM-TC** in real-time so as to better manage the overall efficiency of *N*TuCab dispatch, future work includes 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 requests.
2) *N*TuCab dispatch queue pre-processing: Re-grouping based on physical proximity between requests and taxis in the respective queues, prior to the start of every dispatch cycle, might better match the geographic distributivity of service requests to that of the available taxis.

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

In conclusion, leveraging on the shortest-time paths computed using real-time traffic information [2], the proposed *N*TuCab dispatch system can potentially achieve high efficiency, particularly in limiting customer waiting time provided the demand for taxi service is manageable for a fleet size. Explicated in this paper are, in our opinion, important theoretical and empirical insights about our proposed multiagent approach. These fundamental insights would serve as a base reference in further research and development on automating distributed taxi dispatch, including the formulation and investigation of new A-QoS formulas to incorporate human drivers' preferences.

APPENDIX

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 presented herein. Intuitively, the heuristic attempts to set the initial (assignment) solution, with each agent $a \in \mathcal{A}$ selecting, as far as it is possible, a self-optimal but different request $r \in \mathcal{O}$ to begin negotiation with. Logically, it can be said to contribute one negotiation round but the overall process can become faster. The details, with $|\mathcal{A}| = |\mathcal{O}| = N \geq 2$, are as follows:

Heuristic H-Max*:* One of the agents in group $\mathcal{A}$ is designated as the speedup initialization (SI) agent, and this fact is made known to all agents at the outset. Every agent $a \in \mathcal{A}$ selects and proposes to the SI agent a request $p \in \mathcal{O}$ that satisfies the following:

$$d[a,p] \quad = \quad \max\{d[a,r] |\ r \in \mathcal{O}\ \} \tag{7}$$

The SI agent, upon receiving all such proposals (including its own), will evaluate them as follows. For each request $r \in \mathcal{O}$:

- It will approve an agent's proposal (i.e., request selection) if it is the only agent selecting request $r \in \mathcal{O}$.
- If two or more agents select the same request $r \in \mathcal{O}$ in their proposal, it will approve the proposal by an agent that offers the highest A-QoS value among them.

Assume that $F \geq 1$ agents have had their proposals approved. Then following the proposal evaluation, if $(N - F) \geq 1$, the SI agent will arbitrarily allocate each of the remaining $(N - F)$ agents with a different request taken from the $(N - F)$ unselected requests.

ACKNOWLEDGEMENT

The authors 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, including the installation of the MITSIMLab simulator software on a Unix workstation.

REFERENCES

[1] K. T. Seow, N. H. Dang, and D.-H. Lee, "Towards an automated multiagent taxi-dispatch system," in *Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE'07)*, Scottsdale, Arizona, USA, 2007, pp. 1045–1050.

[2] 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.

[3] Z. Liao, "Taxi dispatching via global positioning systems," *IEEE Transactions on Engineering Management*, vol. 48, no. 3, pp. 342–347, August 2001.

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

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

[6] X. Zhao and Y.-J. Son, "Penalty function-based two-level hybrid shop floor control system," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 2, pp. 220–232, April 2007.

[7] B.-I. Kim, S. S. Heragu, R. J. Graves, and A. S. Onge, "A hybrid scheduling and control system architecture for warehouse management," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 6, pp. 991–1001, December 2003.

[8] S. K. Moon, J. Park, T. W. Simpson, and S. R. T. Kumara, "A dynamic multiagent system based on a negotiation mechanism for product family design," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 2, pp. 234–244, April 2008.

[9] D. Garg and Y. Narahari, "Mechanism design for single leader stackelberg problems and application to procurement auction design," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 3, pp. 377–393, July 2008.

[10] K. Dorer and M. Calisti, "An adaptive solution to dynamic transport optimization," in *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'05): Industry Track*, Utrecht, The Netherlands, July 2005, pp. 45–51.

[11] J. L. Alder and V. J. Blue, "A cooperative multi-agent transportation management and route guidance system," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 5-6, pp. 433–454, October-December 2002.

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] F. L. Bellifemine, G. Caire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*. (Wiley Series in Agent Technology) John Wiley and Sons, Inc., New York, April 2007, JADE software platform available at `jade.tilab.com`.

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

[18] 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.

[19] 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.

[20] 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.

[21] K. T. Seow, K. M. Sim, and Y. C. Kwek, "Coalition formation for resource co-allocation using BDI assignment agents," *IEEE Transactions on Systems, Man and Cybernetics: Part C*, vol. 37, no. 4, pp. 682–693, July 2007.

[22] 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.

[23] 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.

[24] Q. Yang, "A Simulation Laboratory for Evaluation of Dynamic Traffic Management Systems," Department of Civil and Environmental Engineering, Massachusetts Institute of Technology, Boston, USA, Ph.D Thesis, June 1997, 193 pages.