2011 14th International IEEE Conference on
Intelligent Transportation Systems
Washington, DC, USA. October 5-7, 2011

Scheduling Multiple Vehicle Mobility Allowance Shuttle Transit (m-MAST) Services

Wei Lu, Lu Lu and Luca Quadrifoglio

Abstract— **The mobility allowance shuttle transit (MAST) system is an innovative hybrid transit system in which vehicles are allowed to deviate from a fixed route to serve flexible demand. In this paper, we develop an insertion heuristic that economically inserts the deviation requests into the schedule for multiple-vehicle MAST system, which has never been addressed in the literature. The proposed heuristic is validated and evaluated by a set of simulations performed at different demand levels and with different control parameters. By comparing its performance versus the optimal solutions, the effectiveness and quality of the heuristic is confirmed. Compared to its single-vehicle counterpart, the multiple-vehicle MAST prevails in terms of rejection rate, passenger waiting time and overall objective function, among other performance indices.**

I. INTRODUCTION

Public transit services are divided into two broad categories: fixed-route transit (FRT) and demand responsive transit (DRT). The FRT systems are thought to be cost-efficient because of their ride-sharing attribute and sufficient loading capacity. But they are considered by the general public to be inconvenient since the fixed stops and schedule are not able to meet individual passenger's desire. This inherent lack of flexibility is the most significant constraint of fixed-route transit. The DRT systems are much more flexible to offer door-to-door pick-up and drop-off services. They have been operated in quite a few cities and working as an effective type of flexible transit service especially within low-density residential areas, such as examples in Denver (CO), Raleigh (NC), Akron (OH), Tacoma (WA), Sarasota (FL), Portland (OR) and Winnipeg (Canada) [1]. However, the associated high cost prevents the DRT to be deployed as a general transit service. Thus, transit agencies are faced with increasing demand for improved and extended DRT service and a combination of these two types of transit systems is needed to provide a relatively cost-efficient and flexible transit type.

The mobility allowance shuttle transit (MAST) is an innovative concept that combines the cost-efficient operability of traditional FRT with the flexibility of DRT systems. It allows transit vehicles to deviate from a fixed route consisting of a few mandatory checkpoints to serve on demand customers within a predetermined service area, and thus can be both affordable and convenient enough to attract the general public. For the MAST system, the fixed route can be either a loop or a line between two terminals. The checkpoints are usually located at major transfer stops or high demand zones and are relatively far from each other. A hard constraint of the MAST system is the scheduled departure time from checkpoints.

The design and operations of the MAST system has attracted considerable attention in recent years. Quadrifoglio et al. [2] evaluated the performance of MAST systems in terms of serving capability and longitudinal velocity. Their results indicate that some basic parameters are helpful in designing the MAST system such as slack time and headway. Quadrifoglio et al. later developed an insertion heuristic scheduling algorithm to address a large amount of demand dynamically [3]. In [4], Quadrifoglio and Dessouky carried out a set of simulations to show the sensitivity analysis for the performance of the insertion heuristic algorithm and the capability of the system over different shapes of service area. Zhao and Dessouky [5] studied the optimal service capacity for the MAST system. Although these studies investigated the design and operations of the MAST system from various aspects, they are all for the single-vehicle MAST system.

Since the MAST system is a special case of the pickup and delivery problem (PDP), it can be modeled as a mixed integer program (MIP). The PDP has been extensively studied and many of the exact algorithms are based on integer programming techniques [6]–[10]. Other exact algorithms include dynamic programming. Psaraftis used dynamic programming to solve the single-vehicle DARP [11] and its variant with time windows [12]. Both algorithms has a time complexity of $O(N^2 3^N)$ (N for customers), and can solve an instance of N up to 20 in a meaningful time.

Since the optimization problem of PDP is known to be strongly $\mathcal{NP}$-hard [13], researchers have been studying on heuristic approaches to solve PDP with large instances in a reasonable (polynomial) time, while not compromising the quality of solution too much. Along these approaches, insertion heuristics are the most popular because they can provide meaningfully good results in very fast running time, thus are capable of handling problems with large instances. Another reason that justifies insertion heuristics in practice is that they can be easily implemented in dynamic environments [14] since they usually insert requests into the existed schedule in a cost-efficient manner. Savelsbergh and Sol [15] gave a complete review on pickup and delivery problem and discussed the several variants of the problem in terms of different optimization objectives, time-constraints, and fleet sizes. Both exact algorithms based on mathematical modeling and heuristics were reviewed. Some recent efforts in insertion heuristics includes Lu and Dessouky's [16]. A major disadvantage of the insertion heuristics is usually it's

W. Lu, L. Lu and L. Quadrifoglio are with Zachry Department of Civil Engineering, Texas A&M University, College Station, TX 77843-3136, USA lquadrifoglio@civil.tamu.edu

hard to bound its performance. Another disadvantage is its myopic and greedy approach for current optimum at each time step without having an overview of all the request. The insertion heuristic controlled by "usable slack time" resolved this issue efficiently [3].

II. Systems Definition and Model Description

A. Definition of MAST System

A general MAST system has a fleet of vehicles serving a set of customers' requests (see [3] for definition). With a fixed pre-defined time headway, vehicles travel along a fixed-route line (back and forth between two terminals or along a loop) which consists of an ordered set of checkpoints associated with predefined departure times, which serve as tight constraints.

The transit demand is defined by a set of requests. Each request consists of pick-up/drop-off locations and a ready time for pick-up. There are four possible types of customers:

- PD (Regular): pick-up and drop-off at a checkpoint
- PND (Hybrid): pick-up at a checkpoint and drop-off at a random point
- NPD (Hybrid): pick-up at a random point and drop-off at a checkpoint
- NPND (Random): pick-up and drop-off at random points

Readers can refer to Section VI for a summary of notation used in this paper. To give an example, we consider $k = 4$ customers (see Table I) with their corresponding pick-up and drop-off stops according to the network in Fig. 1, describing a simple single-vehicle MAST (1-MAST) system with $TC = TC_0 = 3$ checkpoints in $N_0 = \{1^0, 2^0, 3^0\}$, two random pick-up stops in $N_{n^+} = \{4^+, 5^+\}$, two random drop-off requests in $N_{n^-} = \{6^-, 7^-\}$, and one vehicle in vehicle set $V = \{1\}$. Let $N = N_0 \cup N_{n^+} \cup N_{n^-}$. The network is almost a complete graph, excluding the arcs violating the conditions described above, namely $(2, 1)$, $(3, 2)$, $(3, 1)$ and $(1, 3)$ that violate the predetermined sequence of checkpoints $(1 \to 2 \to 3)$ and $(6, 4)$, $(2, 5)$, $(7, 1)$ that violate the pick-up before drop-off precedence for each request. In addition, since checkpoints 1 and 3 represent the beginning and the end of the service, there are no arcs to 1 and no arcs from 3.

When another vehicle is added into the system (i.e., $V = \{1, 2\}$), the number of checkpoints stays the same ($TC_0 = 3$) but we need to double TC ($TC = 2 \times TC_0 = 6$) because the checkpoint are visited by different vehicles at

TABLE I
Sample Set of Requests

k	$ps(k)$	$ds(k)$
1	4^+	6^-
2	1^0	7^-
3	5^+	2^0
4	1^0	3^0

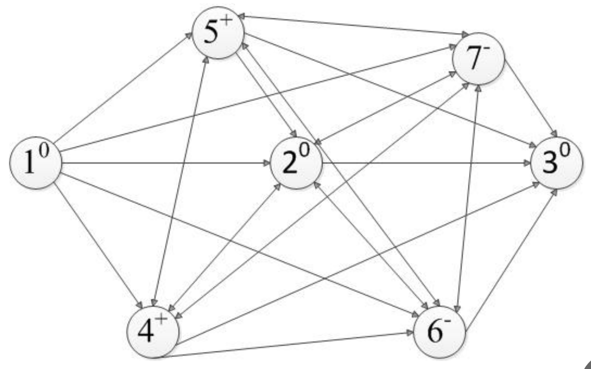

Fig. 1. Sample 1-MAST network

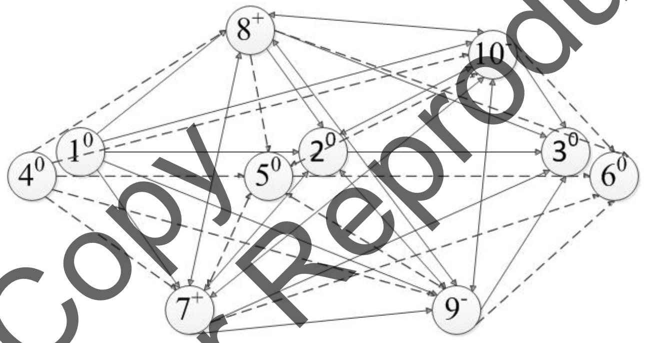

Fig. 2. Sample 2-MAST network

different times. As a result, the arcs nearly double even if the request set remains the same (see Fig. 2). Note that the solid arcs are legal arcs for vehicle 1, while dash arcs are for vehicle 2. The nodes 1^0 & 4^0 (so as 2^0 & 5^0 and 3^0 & 6^0) are essentially the same node geographically, but in the perspective of scheduling they're not, since they are visited by different vehicles at different times. For the graph, the same aforementioned precedence and time constraints still apply in a m-MAST system. Besides, no arcs between nodes representing checkpoints visited by different vehicles (such as $(1 \to 4)$ and $(1 \to 5)$) are allowed.

To formally introduce the multiple-vehicle MAST problem, we first present some definitions.

Definition 1 (m-MAST route)**.** *An m-MAST route Rt_v for vehicle v is a directed route through a subset $N_v \subset N$ such that:*

1) *Rt_v starts in $1 + (v - 1) \times TC_0$.*
2) *$\{1 + (v - 1) \times TC_0, ..., v \times TC_0\} \subset N_v$.*
3) *$(ps(k) \cup ds(k)) \cap N_v = \varnothing$ or $(ps(k) \cup ds(k)) \cap N_v = ps(k) \cup ds(k)$ for all $k \in K$.*
4) *If $ps(k) \cup ds(k) \subseteq N_v$, then $ps(k)$ is visited before $ds(k)$.*
5) *Vehicle v visits each location in N_v exactly once.*
6) *Precedence constraint of pick-up and drop-off is not violated.*
7) *Departure times at checkpoints $\{1 + (v - 1) \times TC_0, ..., v \times TC_0\}$ are complied with.*
8) *Rt_v ends in $v \times TC_0$.*

Definition 2 (m-MAST plan)**.** *An m-MAST plan is a set of routes $\mathcal{RT} = \{Rt_v | v \in V\}$ such that:*

1) *Rt_v is a m-MAST route for vehicle v, for each $v \in V$.*
2) *$\{N_v | v \in V\}$ is a partition of N.*

Define $f(\mathcal{RT})$ as the price of plan $\mathcal{RT}$ corresponding to a certain objective function f. Then we define the m-MAST problem as:

$$min\{f(\mathcal{RT}|\mathcal{RT} \text{ is a m-MAST plan})\}$$

Particularly in this paper f is a combination of operation cost and dissatisfaction of customers, defined by:

$$w_1 \times M/v_{speed} + w_2 \times RT \times |K| + w_3 \times WT \times |K|$$

where w_1, w_2, and w_3 are the weights and M represents the total miles driven by the vehicles, v_{speed} is the speed of vehicles, RT is the average ride time per customer, WT the average waiting time per customer from the ready time to the pick-up time, and K is the set of customers. This definition of the f allows optimizing in terms of both the vehicle variable cost (first term) and the service level (the last two terms); modifying the weights accordingly, we can emphasize one factor over the others as needed.

Definition 3 (m-MAST problem)**.** *An optimization problem m-MAST is a 4-tuple $< I_Q, S_Q, f_Q, opt_Q >$, where:*

- *I_Q: the set of all MAST graphs G*
- *S_Q: the set of all m-MAST plans of the graph G*
- *f_Q: $f(\mathcal{RT})$ is the price of m-MAST plan $\mathcal{RT}$ of G*
- *opt_Q: min.*

Theorem 1. *m-MAST problem is $\mathcal{NP}$-hard in the strong sense.*

Proof: We prove by showing that pickup and delivery problem (PDP) [15], which is known to be strongly $\mathcal{NP}$-hard [13], is reducible to m-MAST. Given an instance of PDP, we can construct an instance of m-MAST by relaxing the constraints on departure times at checkpoints, i.e., setting the departure times to be a time window $[0, \infty]$. In this way a solution to the constructed m-MAST corresponds to the original PDP. So solving PDP is no harder than solving m-MAST. Since the reduction can certainly be done in polynomial time $O(|N|)$, we have proved that m-MAST is strongly $\mathcal{NP}$-hard.

MAST system handles the requests differently. Since PD requests use the service like a fixed-route line, there's no need of booking for them. The other types of requests need to make reservations to schedule their non-checkpoint service stops. We consider the following two assumptions in the multi-vehicle MAST problem: 1) the scenario is dynamic so customers may book their rides at any time before or during the service; and 2) each request only has one customer.

B. Model Description

Specifically, the multi-vehicle MAST system (m-MAST) analyzed in this paper consists of a set of vehicles V with predefined schedules along a fixed-route of C checkpoints

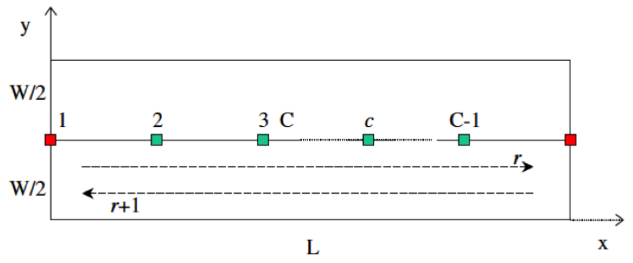

Fig. 3. Multi-Vehicle MAST System

($i = 1, 2, ..., C$). These checkpoints include two terminals ($i = 1$ and $i = C$) and the remaining $C - 2$ intermediate checkpoints. A rectangular service area is considered in this study as shown in Fig. 3, where L is the distance between the two terminals and $W/2$ is the maximum allowable deviation distance on each side of the fixed-route. A trip r is defined as a portion of the schedule beginning at one of the terminals and ending at the other after traversing all the intermediate checkpoints. Each vehicle perform R trips back and forth between the two terminals (see Fig. 3). Since the end terminal of a trip r corresponds to the start terminal of the following trip $r + 1$, the total number of stops at the checkpoints of one vehicle is $TC_0 = (C-1)R+1$, and the total number of stops at the checkpoints of all vehicles is $TC = TC_0 \times |V| = [(C-1)R+1] \times |V|$. Hence, the initial schedules array is represented by an ordered sequence of stops $s = 1, ..., TC$.

We identify the checkpoints with $s = 1, ..., TC$, and the non-checkpoint customers requests (NP or ND) with $s = TC + 1, ..., TS$, where TS represents the current total number of stops. Each stop s of vehicle v has an associated departure and arrival times respectively identified by $t_{s,v}$ and $t'_{s,v}$. As mentioned, the scheduled departure times $t_{s,v}$ of the checkpoints ($\forall s \le TC$) are fixed and assumed to be constraints of the system, which can not be violated, while the $t_{s,v}$ of the non-checkpoint stops ($\forall s > TC$) and all the $t'_{s,v}$ are variables of the system.

Different from single-vehicle MAST, the optimization of multi-vehicle MAST is a more restricted problem in which three types of decisions have to be made: 1). Assignment: the solution has to assign requests to vehicles in a way the objective function is minimized. 2). Routing: the solution has to specify routes for vehicles so that the total miles are small. 3). Scheduling: the solution has to give schedules (sequence) to pick up and drop off customers so that the waiting and ride times are small.

The objective of this research is to develop an insertion heuristic algorithm that efficiently inserts the random requests (deviations) into the schedule in an economical manner. The algorithm is expected to reasonably approximate the optimality of the problem in a polynomial time so that the real-time dynamic operation of the service is possible. In another perspective, let $\alpha(s)$ represent the current position of stop s in the schedule, $\forall s$. The problem defined by a m-MAST system is to determine the indices $\alpha(s)$ and the departure/arrival times $t_{s,v}$, $\forall s > TC, \forall v$, and $t'_{s,v}$, $\forall s$, in order to minimize objective function while not violating the checkpoints fixed departure times $t_{s,v}$, $\forall s \leq TC, \forall v$, and the customers precedence constraints. The problem is solved by means of an cheapest insertion heuristic algorithm described in the following section.

It is assumed that vehicles are homogeneous, following checkpoints back and forth with a predefined time headway. And the vehicles have unlimited capacity. This simplifies the mathematical problem, without compromising adherence to reality, as even small vehicles will almost never be filled up to capacity, due to much more restrictive time constraints.

III. ALGORITHM DESCRIPTION

A. Control Parameters

1) Usable Slack Time (st): Slack time is a crucial resource in MAST system for vehicles to deviate from the main route to serve NP and ND requests. The initial slack time of vehicle v between two consecutive checkpoints s and $s+1$, $st^{(0)}_{s,s+1,v}$, is defined by

$$st^{(0)}_{s,s+1,v} = t_{s+1,v} - t_{s,v} - d_{s,s+1}/v_{speed} - h_{s+1}, \quad \forall s = 1, ..., TC-1 \tag{1}$$

where v_{speed} is the vehicle speed, h_{s+1} the time allowed at stop $s+1$ for passengers boardings and dis-embarkments, and $d_{s,s+1}$ the distance between s and $s+1$. As more pick ups and drop offs occur off the base route, the current slack time $st_{s,s+1,v}$ available is progressively reduced. Initially,

$$st_{s,s+1,v} = st^{(0)}_{s,s+1,v} \quad \forall s = 1, ..., TC-1 \tag{2}$$

At time t_{now}, usable slack time $st^{u}_{s,s+1,v}$ of vehicle v between vehicle s and $s+1$ is defined as:

$$st^{u}_{s,s+1,v} = \begin{cases} \pi^{(0)}_{s,s+1} st^{(0)}_{s,s+1,v}, & \text{for } t_{now} < t_{s,v} \\ [1 + (\pi^{(0)}_{s,s+1} - 1)(1 - \frac{t_{now} - t_{s,v}}{t_{s+1,v} - t_{s,v}})] st^{(0)}_{s,s+1,v} \\ , \quad \text{for } t_{s,v} \leq t_{now} \leq t_{s+1,v} \\ st^{(0)}_{s,s+1,v} & \text{for } t_{now} > t_{s+1,v} \end{cases} \tag{3}$$

where $\pi^{(0)}_{s,s+1}$ is the parameter controlling the usage of slack time, the lower it is set, the more slack time is reserved for future insertions.

Setting $\pi^{(0)}_{s,s+1}$ too low would prevent the algorithm from working properly. From [3], we know that the minimum value of $\pi^{(0)}_{s,s+1}$ should be:

$$\pi^{(0)min}_{s,s+1} = (W/v_{speed} + h_q)/st^{(0)}_{s,s+1,v} \tag{4}$$

2) Backtracking Distance (bd): The vehicles could drive back and forth with respect to the direction of a trip r while serving customers in the service area, having a negative impact on the customers already onboard, which may perceive this behavior as an additional delay. Therefore, we limit the amount of backtracking in the schedule. The backtracking distance indicates how much the vehicle drives backwards on the x-axis while moving between two consecutive stops to either pick up or drop off a customer at a non-checkpoint stop with respect to the direction of the current trip. Denote backtracking distance between stop a and b by $bd_{a,b}$. We define the control parameter $BACK > 0$ that is the maximum allowable backtracking distance that the vehicle can ride between any two consecutive stops. Clearly, with $BACK \geq L$, any backtracking is allowed.

B. Feasibility

While evaluating a customer request, the algorithm needs to determine the feasibility of the insertion of a new stop $s = q$ between any two consecutive stops a and b already scheduled. The extra time needed for the insertion is computed as follows:

$$\Delta t_{a,q,b} = (d_{a,q} + d_{q,b} - d_{a,b})/v_{speed} - h_q \tag{5}$$

Let m and $m+1$ be the checkpoints before and after stops a and b in the schedule. It is feasible to insert q between a and b if the following conditions hold:

$$\begin{cases} \Delta t_{a,q,b} \leq min(st_{m,m+1,v}, st^{u}_{m,m+1,v}) \\ bd_{a,q} \leq BACK \\ bd_{q,b} \leq BACK \end{cases} \tag{6}$$

C. Cost Function

For each feasible insertion of a stop q, the algorithm computes the following quantities along with $\Delta t_{a,q,b}$:

- ΔRT: the sum over all passengers of the extra ride time, including the ride time of the customer requesting the insertion.
- ΔWT: the sum over all passengers of the extra waiting time.

Finally, the cost function is defined as:

$$COST = w_1 \times \Delta t_{a,q,b} + w_2 \times \Delta RT + w_3 \times \Delta WT \tag{7}$$

D. Buckets

Considering the schedule's array, Each checkpoint c is scheduled to be visited by each vehicle a number of times, with different stop indices $s(r,c,v)$ (stop index of the r-th occurrence of checkpoint c in the schedule of vehicle v), depending on the fleet size and how many trips R are planned.

For each intermediate checkpoint $c = 2, ..., C-1$ and each $v \in V$ the indices $s(k,c,v)$, which identify them in the schedule, are computed by the following sequence:

$$\begin{aligned} s(r,c,v) =& 1 + (C-1)(r-1) \\ &+ \frac{(C-1) + (-1)^r[(C-1) - 2(c-1)]}{2} \\ &+ (v-1)TC_0 \quad \forall r = 1, ..., R, \forall v = 1, ..., |V| \end{aligned} \tag{8}$$

For the terminal checkpoints 1 and C, since their frequency of occurrence is halved, the sequences are the following:

$$s(r,1,v) = 1 + 2(C-1)(r-1) + (v-1)TC_0, \quad \forall r = 1, ..., 1 + \lfloor R/2 \rfloor, \forall v = 1, ...|V| \tag{9}$$

$$s(r, C, v) = C + 2(C-1)(r-1) + (v-1)TC_0, \quad \forall r = 1, ..., 1 + \lceil R/2 \rceil, \forall v = 1, ...|V| \tag{10}$$

Definition 4. *For every checkpoint c and every $v \in V$, a bucket of c and v is a portion of the schedule delimited by two successive occurrences of c by the same vehicle, namely all the stops s in the current schedule's array such that $\alpha[s(r, c, v)] \leq \alpha(s) < \alpha[s(r+1, c, v)]$ for any allowable r, as described in Eq. 8 - Eq. 10.*

The buckets' definition for $NPND$-type customers needs to be slightly revised. We characterize the sequence representing the occurrences of any terminal checkpoint ($c = 1$ or C):

$$s(r, 1 \text{ or } C, v) = 1 + (C-1)(r-1) + (v-1)TC_0 \quad \forall r = 1, ..., R+1, \forall v = 1, ..., |V| \tag{11}$$

For $NPND - type$ customers, a bucket represents all the stops s such that $\alpha[s(r, 1 \text{ or C}, v)] \leq \alpha(s) < \alpha[s(r + 1, 1 \text{ or C}, v)]$ for any allowable r as described in Eq. 11

E. Assignment and Insertion Procedure

1) PD type: PD-type requests do not need any insertion procedure, since both pick-up and drop-off points are checkpoints and they are already part of the schedule. However assignment procedure is needed. The assignment is made by choosing the vehicle traveling as the desired direction of the customer that can provides the earliest pickup time .

2) PND type: PND-type customers need to have their ND stop q inserted in the schedule. The algorithm checks for insertions feasibility in the buckets of the P checkpoint. Since the ND stop cannot be scheduled before P, the first bucket to be examined is the one starting with the first occurrence of P following the current position of the vehicle, that is the bucket delimited by $s(k, P, v)$ and $s(k' + 1, P, v)$, with $(k', v) = min_{k,v} s(k, P, v)$, s.t. $t_{s(k,P,v)} \geq t_{now}$. Among the feasible insertions between all pairs of consecutive stops a, b in the first bucket of all the vehicles, the algorithm selects the one with the minimum COST and then stops. The customer is therefore scheduled to be picked up at $s(k, P, v)$ and dropped off at the ND inserted stop q. If no feasible insertions are found in the first bucket, the algorithm repeats the procedure in the second bucket, assuming that the customer will be picked up at the beginning of it corresponding to the following occurrence of P, that is $s(k' + 1, P, v)$. This process is repeated bucket by bucket until at least one feasible insertion is found.

3) NPD type: NPD-type customers need to have their NP stop q inserted in the schedule. The algorithm runs in a very similar way except for changing the insertion from ND to NP.

4) NPND type: A NPND-type customer requires the insertion of two new stops q and q'; therefore, the insertion procedure will be performed by a $O(|V| \cdot |TS|^2)$ procedure, where $|TS|$ is the total number of stops. It means that for each feasible insertion of the NP stop q, the algorithm checks feasibility for the ND stop q'. A NPND feasibility is granted when both NP and ND insertions are simultaneously feasible. The search for NPND feasibility is performed with the additional constraint of having q scheduled before q'.

The search for NPND feasibility is performed in at most two consecutive buckets meaning that when checking for NP insertion feasibility in bucket i and $i+1$, the algorithm looks for ND insertion feasibility only in bucket i and $i + 1$. For each of the vehicles, the algorithm starts checking the NPND feasibility in the first bucket delimited by the current position of the bus (x_b, y_b) and the end of the current trip r. This is the first occurrence in the schedule of one of the terminal checkpoints $s = 1$ or $s = C$, namely $s(k', 1 \text{ or } C, v) = min_{k,v} s(k, 1 \text{ or } C, v)$, s.t. $t_{s(k,1 \text{ or } C,v)} \geq t_{now}$. Among all the feasible NPND insertions in the first bucket, the algorithm selects the one with the minimum COST. If no NPND feasibility is found, the algorithm will then check pairs of two consecutive buckets at a time, increasing the checking-range by one bucket at each step (buckets 1/2, then buckets 2/3, . . . , i/(i + 1), etc.). While checking buckets i/i + 1, we already know that NPND insertion is infeasible in bucket i Therefore, while NP insertion feasibility needs to be considered in both buckets (since NPND insertion infeasibility in bucket i does not prevent NP insertion to be feasible in i), ND insertion needs to be checked only in bucket i + 1. The procedure will continue until at least one NPND feasible insertion is found.

5) Rejection Policy: The general assumption while performing the insertion procedure is a no-rejection policy from both the MAST service and the customers. Thus, the algorithm attempts to insert the customer requests checking, if necessary, the whole existing schedule of all the vehicles bucket by bucket. So generally pending requests will not be rejected, rather be postponed. However, in a static environment, where the trips of service are very small, requests are more likely to be rejected. But this still happens at only very small probabilities.

F. Update Time Windows

The algorithm provides customers at the time of the request with time windows for their pick-up and drop-off locations. Assuming the customer is assigned to vehicle

v, the earliest departure time $et_{q,v}$ from q is computed as follows:

$$et_{q,v} = t_{a,v} + d_{a,q}/v_{speed} + h_q \tag{12}$$

where $t_{a,v}$ represents the current departure time from stop a of vehicle v. Also the departure time of q is initialized likewise:

$$t_{q,v} = t_{a,v} + d_{a,q}/v + h_q = et_{q,v} \tag{13}$$

It can be easily shown that $et_{q,v}$ is a lower bound for any further updates of $t_{q,v}$. The algorithm then computes the latest departure time from q, $lt_{q,v}$, as follows:

$$lt_{q,v} = et_{q,v} + st_{m,m+1,v} \tag{14}$$

Quadrifoglio et al. [3] proved that $lt_{q,v}$ is an upper bound for $t_{q,v}$ by a contradiction argument. Therefore, Eq. 14 says that future possible insertions between m and q will delay $t_{q,v}$ to a maximum total amount of time bounded by the currently available slack time.

In a similar fashion, the earliest and latest arrival times, $et'_{q,v}$ and $lt'_{q,v}$, are computed. As a result, the customer, once accepted, is provided with $et_{q,v}, lt_{q,v}, et'_{q,v}$, and $lt'_{q,v}$, being aware that their actual times $t_{q,v}$ and $t'_{q,v}$ will be bounded by these values:

$$et_{q,v} \le t_{q,v} \le lt_{q,v} \tag{15}$$
$$et'_{q,v} \le t'_{q,v} \le lt'_{q,v} \tag{16}$$

While a P request has $et_{P,v} = t_{P,v} = lt_{P,v}$ because the departure time from a checkpoint is a constant in a MAST system, a D request will have $et'_{D,v} \le t'_{D,v} \le lt'_{D,v}$. Clearly, NP and ND requests will also have $et_{NP,v} \le t_{NP,v} \le lt_{NP,v}$ and $et'_{ND,v} \le t'_{ND,v} \le lt'_{ND,v}$.

G. Overall Approach

The overall approach is described by Algorithm 1. The overall time complexity of the algorithm is $O(T \cdot |N| \cdot |V|)$, where T is the overall service horizon, $|N|$ is the total number of customers, $|V|$ is the total number of vehicles.

Algorithm 1 Overall Scheme

```
for $t = service\_start$ to $service\_end$ do
    if If customer request $i$ received then
        for each vehicle $v$ ($v = 1, 2, ..., |V|$) do
            while No feasible insertion found do
                1. Check the current bucket for the feasible insertion spots for customer $i$'s NP or ND request.
                2. Go to check the next bucket
            end while
            Record $sub\_mincost(v)$, the min. cost of $v$
        end for
        if at least one vehicle have feasible insertion spot then
            1. assign customer i to the v with minimum $sub\_mincost(v)$ for $\forall v = 1, 2, ..., |V|$
            2. customer i accepted
        else
            customer i rejected
        end if
    end if
end for
```

IV. Experimental Results

The target of this section is to show that the proposed insertion heuristic can be used as an efficient scheduling tool for m-MAST systems. Two series of experiments are conducted. First, after tuning the control parameters, 2-MAST system and 1-MAST system are compared to confirm the potential of m-MAST to handle heavy demand. Then the algorithm is compared to optimality obtained through solving the mixed integer program of MAST using $CPLEX^{©}$.

A. Performance Measures and System Parameters

The performance measures of interests include:

- WT: Average waiting time (the difference between actual pick-up time and request time) per customer
- RT: Average ride time (the difference between drop-off time and pick-up time) per customer
- M: Total mileage traveled by vehicles
- Rej. Rate: Rejection rate shows the percentage of customers that are not accepted.
- Z: The objective function Z in this section is a slightly different from Eq. 7. Instead, it is defined as Eq. 17, indicating total operation cost and service level of MAST. Note N is the total number of customers.

$$Z = w_1 \times M/v_{speed} + w_2 \times RT \times N + w_3 \times WT \times N \tag{17}$$

A summary of the parameters that are used in the experiments and the customer types distribution are shown in Table II. It is assumed that the checkpoint requests (P and D) are uniformly distributed among the C checkpoints and that non-checkpoint requests (NP and ND) are uniformly distributed in the service area.

B. 2-MAST vs. 1-MAST

The performance of 2-MAST and 1-MAST are compared under their tuned control parameter settings (see [3] for

TABLE II

SYSTEM PARAMETERS

L	10 miles
W	1 mile
C	3
v_{speed}	25 miles/h
$b_s(s = 1, ..., TS)$	18 sec
t_v	50 min
$w_1/w_2/w_3$	0.4/0.4/0.2
$\|PD\|/\|PND\|/\|NPD\|/\|NPND\|$	10%/40%/40%/10%

details of proper parameter setting for 1-MAST). The best configuration for 2-MAST according to Z (and all other performance measures) is found by setting $BACK = 0.2$ miles, and $\pi^{(0)}_{s,s+1} = 0.3$. The detailed process of tuning the parameters is omitted in this paper. The experiments are conducted under three different levels of demand θ. The simulation is run for 50 hours (equivalently, $R = 60$). The results are summarized in Table III.

The following observation can be made: 1). 2-MAST provides a service of WT nearly half of that of 1-MAST, which is certainly more attractable to customers. RT and Rej. Rate also has a better value. 2). Although M value of 2-MAST is larger than 1-MAST due to the increase in fleet size, the overall objective value Z of 2-MAST is still significantly better than that of 1-MAST, indicating the advantage of 2-MAST.

C. Heuristic vs. Optimality

Although the worst-case analysis of approximation scheme is of theoretical interests, it becomes intractable in this research because of the existence of complicated time constraints and weighted combination of objective function. As a result we conduct several numerical experiments based on random generated demand to evaluate the performance of the algorithm. In this section, the results produced by the proposed heuristic is compared with the optimal results by solving the integer program using $CPLEX^{©}$, a commercial solver.

The results of two different settings of $R = 6$ and $R = 4$ are shown in Table IV and Table V. Note that the demand is represented by $(|PD|, |PND|, |NPD|, |NPND|)$.

Based on the results of Table 3 and Table 4, the following observations can be made: 1). From the ratio of apx/opt we can see, the performance of the heuristic is reasonably good. 2). The ratio is reasonably stable which isn't growing intractably big as the demand increases.

V. CONCLUSION AND FUTURE RESEARCH

In this paper, we first give the formal definition of the optimization problem of scheduling m-MAST and give the proof of its $\mathcal{NP}$-hardness. Then we develop an $O(T \cdot |N| \cdot |V|)$ insertion heuristic for scheduling m-MAST service. The algorithm allows customers to place a request, and once accepted, provides them with time windows for both pick-up and drop-off points. The algorithm makes effective use of control parameters to reduce the consumption of slack time and enhance the algorithm performance. We resort to experiments to evaluate the algorithm. The results of simulations show the efficacy of the algorithm and its optimal control parameter setting and demonstrate that the algorithm can be used as an effective method to schedule m-MAST service. By comparing the performance of 2-MAST and 1-MAST, the potential of m-MAST to provide a more attractable service and a better overall operation cost is shown. In addition, a comparison versus optimality values computed by $CPLEX^{©}$ in a static scenario shows that the results obtained by the heuristic are not far from optimum.

VI. NOTATION

Requests:

- K = set of requests
- $ps(k) \in N_{n^+}$ = pick-up stop of random request k
- $ds(k) \in N_{n^-}$ = drop-off stop of random request k

Sets of Nodes:

- N_0 = checkpoints
- N_{n^+}/N_{n^-} = random pick-up/drop-off stops
- $N = N_0 \cup N_{n^+} \cup N_{n^-}$

Parameters:

- R = number of trips
- C = number of checkpoints
- V = set of vehicles
- v_{speed} = vehicle speed
- $TC_0 = (C - 1) \times R + 1$ = number of checkpoint stops of one vehicle
- $TC = TC_0 \times |V|$ = total number of stops at checkpoints in the schedule
- $d_{i,j}$ = rectilinear distance between i and j, $\forall i, j \in N$
- $w_1/w_2/w_3$ = objective function weights

Variables:

- s = stop index
- TS = current total number of stops, including checkpoints and non-checkpoints
- $t_{s,v}$ = scheduled departure time at stop s of vehicle v
- $t'_{s,v}$ = scheduled arrival time at stop s of vehicle v
- $et_{q,v}$ = earliest departure time from stop q of vehicle v

TABLE III

2-MAST VS. 1-MAST

	2-MAST					1-MAST				
θ (customers per hour)	Z	WT	RT	M	Rej. Rate	Z	WT	RT	M	Rej. Rate
15	10946.9	25.25	19.11	1487.8	0%	14965.1	51.92	21.35	862.0	0.4%
20	14814.4	26.09	20.24	1564.0	0%	20610.9	52.19	23.57	921.8	0.7%
25	17882.3	25.39	19.95	1626.9	0%	27533.2	58.64	24.36	969.9	0.88%

TABLE IV

HEURISTIC VS. OPTIMALITY, R=6

		Heuristic				Optimal			
Demand	apx/opt	obj.	M	RT	WT	obj.	M	RT	WT
$(2, 5, 5, 2)$	1.06	284.8	129.1	196.4	411.5	267.5	125.5	163.2	409.0
$(2, 6, 6, 2)$	1.09	292.1	131.8	214.9	398.2	268.5	126.1	166.7	403.8
$(2, 7, 7, 2)$	1.10	322.3	129.4	254.8	480.6	291.7	125.0	213.4	431.7
$(2, 8, 8, 2)$	1.13	364.8	140.8	324.9	498.3	321.5	128.6	281.6	426.8
$(2, 9, 9, 2)$	1.11	395.2	134.5	312.3	705.9	354.2	125.7	236.6	694.7

TABLE V

HEURISTIC VS. OPTIMALITY, R=4

		Heuristic				Optimal			
Demand	apx/opt	obj.	M	RT	WT	obj.	M	RT	WT
$(2, 3, 3, 2)$	1.04	224.6	88.9	223.8	248.5	215.1	86.1	193.5	275.4
$(2, 4, 4, 2)$	1.13	215.6	86.0	196.2	272.7	190.3	81.6	151.2	257.5
$(2, 5, 5, 2)$	1.14	275.7	90.6	203.8	536.0	242.5	89.5	176.6	429.5
$(2, 6, 6, 2)$	1.04	242.7	92.5	213.7	342.1	232.7	83.8	193.5	374.4
$(2, 7, 7, 2)$	1.12	284.3	88.7	255.2	485.2	252.8	83.0	213.2	439.0

- $et'_{q,v}$ = earliest arrival time from stop q of vehicle v
- $lt_{q,v}$ = latest departure time from stop q of vehicle v
- $lt'_{q,v}$ = latest arrival time from stop q of vehicle v

REFERENCES

[1] D. Koffman, "Operational experiences with flexible transit services: a synthesis of transit practice," Transportation Research Board, Washington DC, TCRP Synthesis 53, 2004.

[2] L. Quadrifoglio, R. W. Hall, and M. M. Dessouky, "Performance and design of mobility allowance shuttle transit services: Bounds on the maximum longitudinal velocity," *Transportation Science*, vol. 40, no. 3, pp. 351–363, 2006.

[3] L. Quadrifoglio, M. Dessouky, and K. Palmer, "An insertion heuristic for scheduling mobility allowance shuttle transit (mast) services," *Journal of Scheduling*, vol. 10, pp. 25–40, 2007.

[4] L. Quadrifoglio and M. M. Dessouky, "Sensitivity analyses over the service area for mobility allowance shuttle transit (mast) services," in *Computer-aided Systems in Public Transport*, ser. Lecture Notes in Economics and Mathematical Systems, 2008, vol. 600, pp. 419–432.

[5] J. Zhao and M. Dessouky, "Service capacity design problems for mobility allowance shuttle transit systems," *Transportation Research Part B: Methodological*, vol. 42, no. 2, pp. 135 – 146, 2008.

[6] J.-F. Cordeau, "A branch-and-cut algorithm for the dial-a-ride problem," *Operations Research*, vol. 54, no. 3, pp. 573–586, 2006.

[7] Q. Lu and M. Dessouky, "An exact algorithm for the multiple vehicle pickup and delivery problem," *Transportation Science*, vol. 38, pp. 503–514, November 2004.

[8] C. E. Corts, M. Matamala, and C. Contardo, "The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method," *European Journal of Operational Research*, vol. 200, no. 3, pp. 711 – 724, 2010.

[9] G. Berbeglia, J.-F. Cordeau, and G. Laporte, "Dynamic pickup and delivery problems," *European Journal of Operational Research*, vol. 202, no. 1, pp. 8 – 15, 2010.

[10] L. Quadrifoglio, M. M. Dessouky, and F. Ordez, "Mobility allowance shuttle transit (mast) services: Mip formulation and strengthening with logic constraints," *European Journal of Operational Research*, vol. 185, no. 2, pp. 481 – 494, 2008.

[11] H. N. Psaraftis, "A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem," *Transportation Science*, vol. 14, no. 2, pp. 130–154, 1980.

[12] ——, "An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows," *Transportation Science*, vol. 17, no. 3, pp. 351–357, 1983.

[13] J. K. Lenstra and A. H. G. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

[14] A. M. Campbell and M. Savelsbergh, "Efficient insertion heuristics for vehicle routing and scheduling problems," *Transportation Science*, vol. 38, no. 3, pp. 369–378, 2004.

[15] M. W. P. Savelsbergh and M. Sol, "The general pickup and delivery problem," *Transportation Science*, vol. 29, no. 1, pp. 17–29, 1995.

[16] Q. Lu and M. M. Dessouky, "A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows," *European Journal of Operational Research*, vol. 175, no. 2, pp. 672 – 687, 2006.