Mobility Allowance Shuttle Transit (MAST) Services: MIP formulation and strengthening with logic constraints

Luca Quadrifoglio

Zachry Department of Civil Engineering
Texas A&M University
lquadrifoglio@civil.tamu.edu

Maged M. Dessouky and Fernando Ordóñez

Daniel J. Epstein Department of Industrial and Systems Engineering
University of Southern California

We study a hybrid transportation system referred to as Mobility Allowance Shuttle Transit (MAST) where vehicles may deviate from a fixed path consisting of a few mandatory checkpoints to serve demand distributed within a proper service area. In this paper we propose a Mixed Integer Programming (MIP) formulation for the static scheduling problem of a MAST type system. Since the problem is NP Hard, we develop sets of logic cuts, by using reasonable assumptions on passengers' behavior. The purpose of these constraints is to speed up the search for optimality by removing inefficient solutions from the original feasible region. Experiments show the effectiveness of the developed inequalities, achieving a reduction up to 90% of the CPU solving time for some of the instances.

Summary[1]

We study a hybrid transportation system referred to as Mobility Allowance Shuttle Transit (MAST) where vehicles may deviate from a fixed path consisting of a few mandatory checkpoints to serve demand distributed within a proper service area. A MAST system is described by a set of vehicles driving along a base fixed-route and serving a specific geographic area. The base route can be laid out around a loop or between two terminals. Vehicles must stop at a set of checkpoints along the main path. The checkpoints are conveniently located at major transfer points or high density demand zones, are relatively far from each other and have fixed departure times. Given a proper amount of slack time, vehicles are allowed to deviate from the fixed path to serve (pick-up and/or drop-off) customers at their desired locations, as long as they are within a service area.

The MAST system considered consists of a single vehicle, associated with a predefined schedule along a fixed-route consisting of C checkpoints. A trip *r* is defined as a portion of the schedule beginning at one of the terminals and ending at the other one after visiting all the intermediate checkpoints. The service area is represented by a rectangular region defined by L×W, where L (on the x axis) is the distance between terminals 1 and C and W/2 (on the y axis) is the maximum allowable deviation from the main route in either side (see **Error! Reference source not found.**).

The demand is defined by a set of requests. Each request is defined by pick up/drop off service stops and a ready time for pick up. The MAST service can respond to four different types of requests: pick up (P) and drop off (D) at the checkpoints; non checkpoint pick up (NP) and drop off (ND), representing customers picked up/dropped off at any location within the service area. A certain amount of slack time between any consecutive pair of checkpoints is needed in order to allow deviations to serve NP or ND requests. There are consequently four different possible types of customers' requests: PD ("Regular"), pick

[1] This is a summary of the following paper: Quadrifoglio L., Dessouky M., Ordóñez F., "Mobility Allowance Shuttle Transit services: MIP formulation and strengthening with logic constraints", *European Journal of Operational Research*, 2008, 185, 481–494

up and drop off at the checkpoints; PND (“Hybrid”), pick up at the checkpoint, drop off not at the checkpoint; NPD (“Hybrid”), pick up not at the checkpoint, drop off at the checkpoint; NPND (“Random”), pick up and drop off not at the checkpoints. In this paper we consider a static scenario in which all the demand is known in advance. We also assume one customer per request, no vehicle capacity constraint and a deterministic environment.

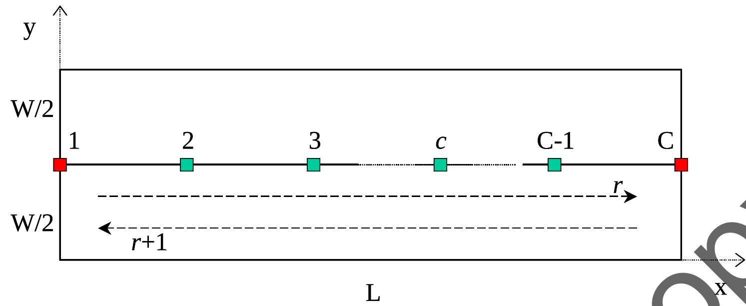

Fig. 1. MAST system

The MAST scheduling problem can be formulated as a mixed integer linear program:

$$\min \quad \boldsymbol{\omega}_1 \left(\sum_{(i,j)\in\mathbf{A}} \boldsymbol{\delta}_{i,j} x_{i,j} \right) \Big/ \mathbf{v} + \boldsymbol{\omega}_2 \left(\sum_{k\in\mathbf{K}} (d_k - p_k) \right) + \boldsymbol{\omega}_3 \left(\sum_{k\in\mathbf{K}} (p_k - \boldsymbol{\tau}_k) \right) \qquad \text{Objective Function}$$

Subject to:

$$\begin{aligned}
&\sum_i x_{i,j} = 1 && \forall j \in N/\{1\} \\
&\sum_j x_{i,j} = 1 && \forall i \in N/\{TC\} \\
&t_i = \theta_i && \forall i \in N_0 \\
&p_k = t_{pu(k)} && \forall k \in K/K_{PND} \\
&d_k = \bar{t}_{do(k)} && \forall k \in K/K_{NPD} \\
&\sum_{r\in HRD(k)} z_{k,r} = 1 && \forall k \in K_{HYB}
\end{aligned}$$

$$\begin{aligned}
&p_k \geq t_{pu(k,r)} - M(1-z_{k,r}) && \forall k \in K_{PND}, \forall r \in HRD(k) \\
&p_k \leq t_{pu(k,r)} + M(1-z_{k,r}) && \forall k \in K_{PND}, \forall r \in HRD(k) \\
&d_k \geq \bar{t}_{do(k,r)} - M(1-z_{k,r}) && \forall k \in K_{NPD}, \forall r \in HRD(k) \\
&d_k \leq \bar{t}_{do(k,r)} + M(1-z_{k,r}) && \forall k \in K_{NPD}, \forall r \in HRD(k) \\
&p_k \geq \tau_k && \forall k \in K \\
&d_k \geq p_k && \forall k \in K \\
&\bar{t}_j \geq t_i + x_{i,j}\delta_{i,j}/v - M(1-x_{i,j}) && \forall (i,j) \in A
\end{aligned}$$

SETS of NODES
N_0 = checkpoints
N_n = non-checkpoints
$N = N_0 \cup N_n$

SETS of ARCS
A = all arcs

SETS of CUSTOMERS
K_{PD} = PD requests
K_{PND} = PND requests
K_{NPD} = NPD requests
K_{NPND} = NPND requests
$K_{HYB} = K_{PND} \cup K_{NPD}$
$K = K_{PD} \cup K_{HYB} \cup K_{NPND}$
$pu(k) \in N, \forall k \in K/K_{PND}$ = pick-up of k
$do(k) \in N, \forall k \in K/K_{NPD}$ = drop-off of k
$pu(k,r) \in N, \forall k \in K_{PND}$ = pick-ups of k
$do(k,r) \in N, \forall k \in K_{NPD}$ = drop-offs of k

SETS of TRIPS
RD = all trips
$HRD(k) \subset RD, \forall k \in K_{HYB}$ = feasible trips of k

VARIABLES
$\boldsymbol{x_{i,j} = \{0, 1\}, \forall (i, j) \in A}$
$t_i, \forall i \in N$ = departure time from i
, $\forall i \in N/\{1\}$ = arrival time at i
$p_k, \forall k \in K$ = pick-up time of request k
$d_k, \forall k \in K$ = drop-off time of request k
$\boldsymbol{z_{k,r} = \{0,1\}, \forall k \in K_{HYB}, \forall r \in HRD(k)}$

PARAMETERS
$\delta_{i,j}, \forall (i, j) \in A$ = distance from i to j
v, vehicle speed
b_i, boarding/disembarking time at i
$\theta_i, \forall i \in N_0$ = departure times from checkpoints
$\tau_k, \forall k \in K$ = ready time of request k
$\omega_1/\omega_2/\omega_3$ = objective function weights

Where $x_{i,j}$ indicates whether an arc (i,j) is used ($x_{i,j} = 1$) or not ($x_{i,j} = 0$) and $z_{k,r}$ indicates whether the checkpoint stop of the hybrid request k (a pick-up if $k \in K_{PND}$ or a drop-off if $k \in K_{NPD}$) is scheduled in trip r, $\forall r \in RD$.

The above formulation is sufficient to find the optimal solution of the problem, but it is ineffective in the sense that it includes many feasible inefficient solutions and thus has a weak LP relaxation.

A way to speed up the search for optimality is the development of constraints and their addition to the math program formulation. These constraints are called valid if they reduce the dimensions of the relaxed feasible region, but all integer feasible solutions of the original model are not touched. The ideal purpose of these constraints is to produce the convex hull of the integer feasible solutions which would allow LP algorithms to solve the problem much faster. Another category of constraints, the so called "logic cuts", have the purpose to eliminate some integer feasible solutions that are provably suboptimal. Thus, they can not be considered valid, but they can be indeed very effective. They may significantly shrink the feasible region, even by some orders of magnitude, and they allow improving the quality of the LP relaxation bound, considerably speeding up the reduction of the optimality gap throughout the iterations of the solver. As a result, they can be extremely beneficial in reducing the CPU time in the search for optimality.

In this paper we develop and add "logic cuts" to strengthen the above MAST formulation. The underlying concept behind all the developed inequalities is that hybrid customers will be choosing their P or D checkpoints as close as possible to their corresponding ND or NP stop, once these are placed in the schedule. More formally, we can state (proofs in the full paper) the following Propositions 1 and 2:

Proposition 1. A necessary condition for optimality is that NPD customers must disembark the vehicle at the first occurrence of their D checkpoint following their scheduled NP pick-up stop.

Proposition 2. If $\omega_2 > \omega_3$, a necessary condition for optimality is that PND customers must board the vehicle at the last occurrence of their P checkpoint prior to their scheduled ND drop-off stop.

Although the logic behind the above Propositions may seem obvious to a human mind, it is not explicitly stated in the formulation and the solver would still consider several feasible but inefficient solutions (violating the above Propositions) as possible candidates while searching for optimality. Therefore, based on the above Propositions, we develop three different groups of valid inequalities to add to the formulation.

Group #1: The first group of inequalities is developed by directly applying Propositions 1 and 2. They include constraints linking the z variables to the t variables (departure times) of non checkpoint stops of hybrid requests and constraints linking the z variables to some of the x variables. An example is

$t_{do(k)} < z_{k,r}\theta_j + M(1-z_{k,r})$, with $j = pu(k,r+1)$, $\forall k \in K_{PND}$, $\forall r \in RD/\{R\}$

Group #2: A second group of inequalities includes constraints linking z and x variables by making use of Propositions 1 and 2 along with the ready times τ of the requests. An example is

$\tau_{q(i)} + \delta_{i,j} + b_j \le z_{k,r}\theta_j + M(2-z_{k,r}-x_{do(k),i})$,
with $i = pu(q(i))$, $j = pu(k,r+1)$, $\forall k \in K_{PND}$, $\forall r \in RD/\{R\}$, $\forall (do(k),i) \in A_n$

Group #3: A third group of inequalities links z and x variables by applying the results from the Propositions to pairs of hybrid requests. An example is

$z_{h,s}\theta_i - z_{k,r}\theta_j < M(3-z_{h,s}-z_{k,r}-x_{do(k),do(h)})$,
with $i = pu(h,s)$, $j = pu(k,r+1)$, $\forall k,h \in K_{PND}$, $\forall r \in RD/\{R\}$, $\forall s \in RD$

Experimental results on several instances (which we are omitting in this summary, but are explained in details in the full paper) show the effectiveness of the developed inequalities, which are able to reduce the CPU solution time by up to more than **90%** for some cases. Specifically, Group "#1" provide the best overall results that always effective, followed in general by Group "#2" and Group "#3", which are not always effective. The synergistic effect of including all the cuts together further reduces the CPU solution time in many cases. We provide the result for one case in the following **Table 1.**:

Table 1.

Case:	**B1a**	**TS=10: R=2; $\lvert K_{PD}\rvert$=1; $\lvert K_{PND}\rvert$=2; $\lvert K_{NPD}\rvert$=1; $\lvert K_{NPND}\rvert$=1**										
cuts	*var*	*bin*	*lin*	*con*	*sec*	*n*	*i*	*rel*	*opt*	*ub*	*lb*	*gap*
none	67	43	24	85	0.04	64	403	81.2	114.7	/	/	0.0%
#1	67	43	24	91	0.03	27	221	81.8	114.7	/	/	0.0%
#2	67	43	24	87	0.04	50	324	81.2	114.7	/	/	0.0%
#3	67	43	24	85	0.04	64	403	81.2	114.7	/	/	0.0%
all	67	43	24	93	0.03	25	217	81.8	114.7	/	/	0.0%
Case:	**B1b**	**TS=15: R=4; $\lvert K_{PD}\rvert$=1; $\lvert K_{PND}\rvert$=2; $\lvert K_{NPD}\rvert$=2; $\lvert K_{NPND}\rvert$=1**										
cuts	*var*	*bin*	*lin*	*con*	*sec*	*n*	*10^3 i*	*rel*	*opt*	*ub*	*lb*	*gap*
none	124	89	35	156	0.56	695	7.91	105.8	164.9	/	/	0.0%
#1	123	88	35	199	0.19	126	1.39	105.8	164.9	/	/	0.0%
#2	124	89	35	188	0.50	643	5.46	105.8	164.9	/	/	0.0%
#3	124	89	35	256	0.62	815	7.25	105.8	164.9	/	/	0.0%
all	123	88	35	309	0.25	89	1.55	105.8	164.9	/	/	0.0%
Case:	**B1c**	**TS=20: R=4; $\lvert K_{PD}\rvert$=1; $\lvert K_{PND}\rvert$=5; $\lvert K_{NPD}\rvert$=4; $\lvert K_{NPND}\rvert$=1**										
cuts	*var*	*bin*	*lin*	*con*	*sec*	*10^3 n*	*10^6 i*	*rel*	*opt*	*ub*	*lb*	*gap*
none	247	197	50	299	619.0	723.3	5.58	132.8	217.8	/	/	0.0%
#1	244	195	49	351	49.0	60.7	0.47	132.8	217.8	/	/	0.0%
#2	247	197	50	400	355.7	319.9	3.33	132.8	217.8	/	/	0.0%
#3	247	197	50	639	508.1	460.2	4.03	132.8	217.8	/	/	0.0%
all	244	195	49	742	32.0	27.2	0.31	132.8	217.8	/	/	0.0%
Case:	**B1d**	**TS=25: R=4; $\lvert K_{PD}\rvert$=2; $\lvert K_{PND}\rvert$=6; $\lvert K_{NPD}\rvert$=6; $\lvert K_{NPND}\rvert$=2**										
cuts	*var*	*bin*	*lin*	*con*	*sec*	*10^6 n*	*10^6 i*	*rel*	*opt*	*ub*	*lb*	*gap*
none	398	336	62	452	36,000	20.2	249	193.0	?	312.8	293.0	6.3%
#1	398	336	62	506	36,000	17.5	235	193.0	?	312.8	304.4	2.7%
#2	398	336	62	552	36,000	17.0	246	193.0	?	312.8	293.4	6.2%
#3	397	335	62	590	36,000	14.4	215	193.0	?	312.8	295.6	5.5%
all	397	335	62	744	36,000	15.3	219	193.0	?	312.8	299.8	4.1%

Where TS is the total number of stops in the network, R is the number of trips, we solved the same instance without adding any groups of inequalities (“none”), adding only one group at a time (“#1”, “#2” or “#3”) or adding all the groups together (“all”). For each run we show the size of the problem solved: total variables (“var”), divided into binary (“bin”) and linear (“lin”) and total number of constraints (“con”). The following columns show the time to reach optimality in seconds (“sec”), the number of nodes visited in the branch and bound tree (“n”), the number of simplex iterations performed (“i”), the relaxed optimal value (“rel”) and the real optimum (“opt”).