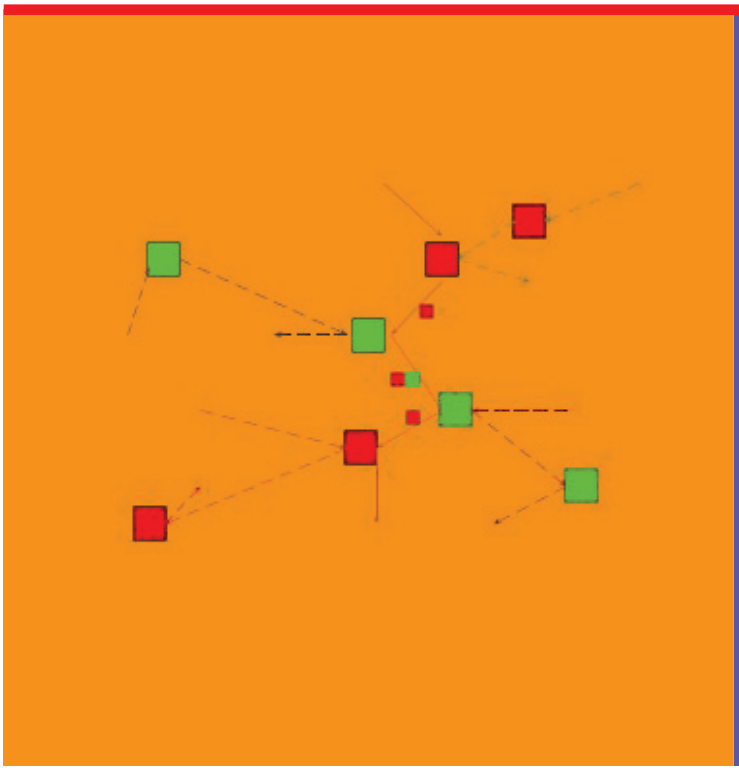
Route planning for airport personnel transporting passengers with reduced mobility

Reinhardt, Line Blander; Clausen, Tommy; Pisinger, David

Publication date:
2010

Document Version
Publisher's PDF, also known as Version of record

Route planning for airport personnel transporting passengers with reduced mobility



Report 17.2010

DTU Management Engineering

Line Blander Reinhardt
Tommy Clausen
David Pisinger
September 2010

DTU Management Engineering
Department of Management Engineering

Route planning for airport personnel transporting passengers with reduced mobility

Line Blander Reinhardt Tommy Clausen David Pisinger

September 17, 2010

Abstract

Major airports have an average throughput of more than 100,000 passengers per day, some of which will need special assistance. The largest airports have a daily average throughput of more than 500 passengers with reduced mobility. A significant number of people and busses are assigned to provide transportation for the passengers with reduced mobility. It is often necessary for a passenger with reduced mobility to use several different modes of transport during their journey through the airport. Synchronization occurs at the locations where transport modes are changed as to not leave passengers unattended. A description of the problem together with a mathematical model is presented. The objective is to maximize the quality of service by scheduling as many of the passengers as possible, while ensuring a smooth transport with short waiting times. A simulated annealing based heuristic for solving the problem is presented. The algorithm makes use of an abstract representation of a candidate solution which in each step is transformed to an actual schedule by use of a greedy heuristic. Local search is performed on the abstract representation using advanced neighborhoods which modify large parts of the candidate solution. Computational results are reported showing that the algorithm is able to find good solutions within a couple of minutes, making the algorithm applicable for dynamic scheduling. Moreover high-quality solutions can be obtained by running the algorithm for 15 minutes.

1 Introduction

At the 31st biggest airport London Gatwick there was a throughput of 32 million passengers in 2009 according to [2]. London Gatwick reports [1] that around 900 passengers with reduced mobility arrived each day.

Such passengers may be passengers returning from vacation with an injury, elderly or weak passengers, blind and deaf passengers, and passengers with other disabilities. In the remaining of this paper we will refer to passengers needing assistance as passengers with reduced mobility (PRM). The support provided for the PRMs may be dedicated transport through the airport, and assistance at boarding. When assisting PRMs through an airport the PRM is picked up at the arriving location, e.g. check-in or gate of arrival, and delivered at the destination location, e.g. arrival hall or gate of departure. It is a service requirement that the PRM is not left alone during the journey from start to

end. It may also be possible to assist more than one PRM at a time depending on whether the PRM is in a wheelchair or how well they are able to walk and orient themselves.

In the case studied the objective is to optimize quality of service given the personnel available. Optimizing quality of service is in our case, given a fixed set of personnel and transport objects, to minimize the number of PRMs not delivered and to minimize the total unnecessary travel time used on the journeys. We view the problem of assisting PRMs as a dial-a-ride problem (DARP), which is a generalization of the pickup and delivery problem (PDP). For more details on the dial-a-ride problem (DARP) definition see Cordeau and Laporte [5].

The dial-a-ride Problem (DARP) is NP-hard by reduction from the Hamiltonian cycle problem (Baugh et al. [3]). Normally, the DARP is defined with time windows at either pickup or delivery, but not both, see [5] and [12]. Even though Cordeau and Laporte in [5] argue that having time windows at both ends may be too restricting for the planning, Jaw et al. [12] show that given a pickup time window and a limit on the passenger travel time an implicit time window is imposed on the delivery. Jaw et al. [12] found that explicit definition of the delivery time window improved their algorithm. In the considered problem there can be explicit time windows at both pick up and delivery.

Airports often have several terminals and the transport between the terminals is at the studied airport done in special buses solely for PRMs. Such buses will have a specific location for picking up PRMs at each terminal. Moreover, for aircrafts not located at a gate, the PRM will be transported in a special bus between the gate and the aircraft. Therefore, the pickup and delivery of a PRM is represented as a number of pickup and delivery segments. The airport and airlines require that the PRMs are not left alone at any point during their journey through the airport, and the PRMs are required to be in their assigned flight seat at a fixed pre-specified time before departure. However, the PRM may be left alone for a while before boarding at the departing terminal in a supervised area.

Between each pick up and delivery of a PRM the transport object delivering the PRM must meet the transport object picking up the PRM. This vehicle synchronization is called a temporal dependency, therefore the problem is a dial-a-ride problem with temporal dependencies (DARPTD). The concept of synchronization in routing was used by Ioachim et al. [11] for the fleet assignment problem and later expanded to the more general temporal dependencies by Dohn el al. [9]. In pickup and delivery problems the similar problem of cross docking has been considered, which has a transfer of goods between vehicles at the synchronized points. The pickup and delivery with cross docking is used in supply chain and planning city logistics systems [4], [7]. Pickup and delivery with cross docking was studied by Wen et al. [15]. In the cross docking problems the cross docking is optional for the vehicles. This is not the case in the problem of assisting PRMs at an airport, as the cross-docking points for each PRM are known and fixed. The synchronization constraints and the objective separates the routing of transport objects transporting PRMs in airports from the rich pickup and delivery problem described in [13].

In this paper we have constructed a local search heuristic for the specific problem based on simulated annealing. The algorithm makes use of an abstract representation, which is transformed to an actual schedule by use of a greedy heuristic. Local search is performed on the abstract representation using large

neighborhoods. In each iteration the resulting candidate solution is evaluated, and accepted according to the standard criteria in simulated annealing. Computational results are reported showing that the algorithm is able to construct high-quality solutions in 15 minutes.

This paper is organized as follows. Section 2 contains a detailed problem description to ensure a thorough understanding of the operational process. In Section 3 a mathematical model of the problem is presented. Section 4 presents the solution methods used. Section 5 contains the specifications of the data instances received. In Section 6 the tuning of the parameters for simulated annealing us described. Section 7 contains the results of the solution method applied to the real-life instances received. In Section 8 there is a discussion on the results and future work.

2 Problem Description

We will term the considered problem the *Airport passenger with reduced mobility transport problem* (APRMTP). The APRMTP has been defined in cooperation with a service company providing the assistance for PRMs at a major transit airport. The company does not deliver service in the entire airport but in the majority of the airport. The company has 120 employees assisting between 300 and 500 PRMs through the airport each day. The employees have a prespecified working area such as a specific terminal, driving between terminals bus stop locations or driving between aircrafts and gates. A worker assigned to one area may not move into another area. Therefore, the journey of the PRM is split up into a pickup and delivery for each of these areas. We call the pick up and delivery in a specific area for a segment and the ordered set of segments of a given PRM for a journey. On average there are three segments per PRM, given the 300 to 500 PRMs each shift we get a total of between 900 and 1500 pick up and delivery segments. This also includes assistance when boarding, which we have represented as a pickup and delivery request with special conditions. We will in the remainder of this paper refer to the boarding assistance as embarkment. The employee assigned to an embarkment cannot go to another location between pickup and delivery of the embarkment segment. However, an employee may assist as many PRMs embarking on to the same flight as their capacity allows.

It should be noted that all of the pickup and delivery locations for every segment of the journey are predetermined.

As mentioned in the previous section, the PRM may not be left alone except at special supervised areas located in the departing terminal. This means that the employee delivering the PRM to a bus must wait with the PRM for the bus to arrive at the bus stop before being able to initiate the next task. The bus also has to wait for the employee to come and pick up the delivered PRM before continuing the route.

The company wants to make sure that they deliver the best service possible with the given number of employees and the current employees working area assignments. The PRMs are split into two categories: Those who are prebooked and those who are immediate. The prebooked PRMs ordered the service when purchasing the ticket or at least days in advance. Immediate PRMs requests the service at check-in, and therefore may only be known half an hour in advance for PRMs arriving on flights or at check-in for passengers checking in at the airport.

It is not always possible for the company to assist all PRMs and in such cases the prebooked PRMs have higher priority. The company also wishes to minimize unnecessary time the PRM spends on the journey segments. Unnecessary time could be time spent waiting to be picked up by the employee working in the area of the succeeding segment or extra travel time caused by picking up or delivering other PRMs before being delivered. Note, that the time spent at the supervised area of the departing terminal is not included in the service evaluation. The unnecessary time spent on the segments we call excess time. The problem is then to route the employees on foot or in their assigned vehicle so that the service quality is maximized. Clearly, when minimizing the overall traveling time there is a risk of having a few PRMs with very large traveling times. Therefore, it is important to limit the traveling times of the different segments so that the journey never becomes very unsatisfactory.

Many additional constraints concerning the pick-up and delivery times, assistance at embarkment and transport to and from aircrafts not located at a gate, are imposed by the airport and airlines. Such constraints are

- an arriving PRM must be picked up exactly upon arrival
- a terminal transfer takes place on bus between the bus stop locations of the terminals
- the PRM may not be left unsupervised
- the PRM must not use more than 30 minutes of excess time on each segment
- embarkment takes 20 minutes and can not start earlier than 60 minutes before departure.
- the PRM must be seated in the plane exactly 20 minutes before departure.

The last two items mean that embarkment can not start later than 40 minutes before departure and even earlier when plane is parked away from gate.

Note, that the person assisting the PRM through embarkment will not be able to leave the PRM until 20 minutes before departure if the aircraft is located at gate or before the PRM is picked up by special vehicle if the aircraft is located away from gate. In the latter case the special vehicle cannot leave the PRM before 20 minutes before departure.

The different transportation forms such as vehicles and assistance on foot have different capacities. Moreover the PRMs are assigned a volume depending on their disability. For example it is very hard for one person to push two wheelchairs however, two hearing or sight impaired persons can be assisted by the same employee.

2.1 Example of a journey of an PRM

The most complex example of a journey is the case where a PRM makes a transit from an arriving aircraft not located at the gate to an departing aircraft in another terminal not located at the gate. In such a case the segments of the journey are as follows (see also Figure 1)

1. The PRM is picked up by a vehicle at the arriving aircraft on the exact time of arrival (it is a requirement that the PRMs are picked up exactly upon arrival) and delivered at gate.

2. The PRM is picked up at the gate by an employee. The bus delivering the PRM in segment 1 cannot leave before the PRM is picked up. The PRM is delivered at the bus stop for the special inter terminal busses.

3. The PRM is picked up by a special inter terminal bus at the bus stop and delivered to the bus stop at the terminal of the departing flight. Again the employee of segment 2 cannot leave before the bus arrives for pickup.

4. The PRM is picked up by an employee and delivered to the gate of departing flight. Again the bus in segment 3 cannot leave before the employee arrives.

5. The PRM is assisted by an employee through embarkment at gate. This task takes 20 minutes. For the time between segments 4 and 5 the PRM can be left at a special supervised lounge.

6. The PRM is picked up at the gate by a vehicle and delivered to the plane exactly 20 minutes before the departure time of the flight. Again the employee of segment 5 cannot leave before the vehicle arrives for pickup.

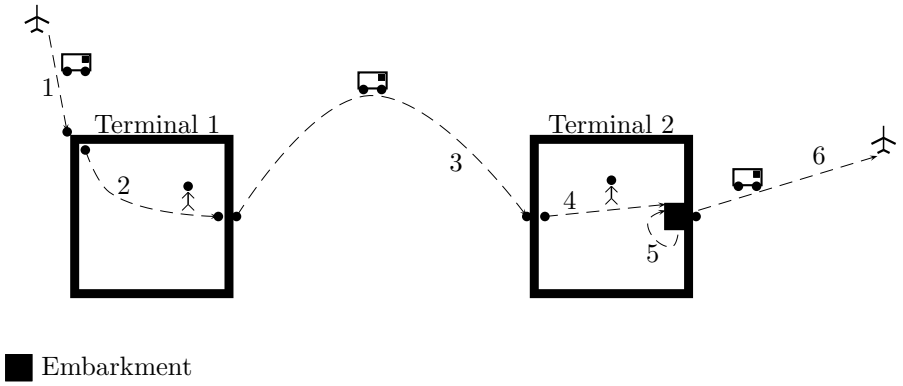


Figure 1: Illustration of a journey with six segments: (1) The PRM is picked up by a vehicle at the arriving aircraft and brought to the gate, (2) The PRM is brought to the inter terminal bus. (3) The PRM is transported by a inter terminal bus from Terminal 1 to Terminal 2, (4) The PRM is brought to the gate of departing flight, (5) The PRM is assisted through embarkment at gate, (6) The PRM is delivered to the aircraft.

We say that such a journey has six segments. All transit journeys are formulated as an ordered subset of these segments. Non-transit PRMs are either picked up or delivered to a public area in the same terminal as respectively the arriving or departing flight.

The algorithm is used in a dynamic setting, where immediate PRMs arrive continuously and disruptions in the daily plan such as flight delays frequently occurs. Therefore, the company desires to receive a solution to the problem within a couple of minutes. We do not consider robustness and break times. However, robustness can be obtained by introducing buffer time in the time to get from one location to another and by altering the set of employees available and breaks maybe included by splitting the shift of an employee into several shifts.

3 Mathematical formulation

When describing the model it is important to bear in mind that the journey of each PRM is a set of pick up and deliveries called segments. This means that a PRM is picked up and delivered if all the segments of the journey are handled. Therefore, we must for each PRM make sure that all their segments are assigned before we consider them delivered. As common in dial-a-ride problems with a heterogeneous fleet each segment is represented by a pickup vertex and a delivery vertex specific to that segment. There is a location inside each area where the employees start and end their shift.

3.1 Graph representation

There is as mentioned earlier a vertex for each origin and destination of a journey segment. The location of all the vertices in a journey are pre determined. Since we already know which transport group is assigned to each vertex we can generate a disjoint graph for each transport group. Each graph has its own depot where the transport objects starts and end their work. These graphs are only "virtually" connected by the connection between segments of a journey. For each terminal we have a directed graph connecting the vertices that must be serviced by foot personnel working in the given terminal. The busses transporting PRMs between terminals have a directed graph of their pick up and delivery vertices. The vehicles transporting PRMs from gates to airplanes have a directed graph connecting their pickup and delivery vertices. Connections between pick up and delivery vertices, which are infeasible due to their time windows, are removed from the graph.

3.2 Mathematical model

Given the following sets:

K	The set of transport objects. Contains all vehicles and persons on foot
R	The set of segments. Contains all the segments of all the journeys
R_p	The set of segments. Contains all the segments for PRM p
B	The set of prebooked PRMs
C	The set of all PRMs

F	The set of departing flights
V	The set of pick up and delivery points/vertices
$V*$	The set of pickup and delivery vertices and depots/bases
V_f	The set of embarkment vertices for flight $f \in F$
P	The set of pickup vertices
D	The set of delivery vertices
E	The set of edges connecting the elements in $V*$
λ_p	The set of vertex pairs (i, j) where i is the delivery vertex of the segment right before the segment with pickup vertex at j on a journey for PRM p
δ	The set of vertex pairs that must be synchronized for handover

Each segment has a start o_d and a destination t_d and the set V is all of the different o_d and t_d vertices. Each work area has a starting point v_0 and an end point v_e representing the location, where the transport objects start and end the day, by two vertices.

We define the following parameters:

M_b	The penalty for not transporting a prebooked PRM
M_n	The penalty for not transporting an immediate PRM
$t_{o_d t_d}$	The minimum time needed to deliver segment $d \in R$
t_{ij}^k	The minimum time it takes to go from i to j on transport k
l'_j	The change in load at vertex $j \in V$
H	The maximum excess time allowed to be used on a segment. Here $H = 30$
C_k	The capacity of transport object $k \in K$
M	A big constant being at least as large as the shift length
M_s	A big constant larger than the largest number of segments in a PRM
M_l	A big constant at least as large as the biggest capacity plus the largest volume possible for a PRM
a_i	The release time at vertex $i \in V*$
b_i	The due time at vertex $i \in V*$

We use the following variables:

s_i^k	the time when transport object k leaves vertex i
ϕ_p	An indicator variable indicating if a PRM $p \in C$ has a segment not assigned. ϕ_p is 0 if all segments of p are assigned and 1 otherwise
x_{ij}^k	An indicator variable indicating if the edge (i, j) is used by object k. x_{ijk} is 1 if the edge is used by k and 0 otherwise
l_i^k	the load on transport object k when leaving vertex i

As objective we have chosen an linear weighted combination of assigning as many PRMs as possible and minimizing the total excess time the PRMs spend on their journey:

$$\min \quad \sum_{d \in R} \left(\left(\sum_{k \in K} s_{t_d k} - s^k_{o_d} \right) - t_{o_d t_d} \right) + \sum_{p \in B} (1 - \phi_p) * M_b + \sum_{p \in C \setminus B} (1 - \phi_p) * M_n \quad (1)$$

s.t.

(P and D)	$\sum_{i \in V} x^k_{i o_d} - \sum_{i \in V} x^k_{i t_d} = 0$	$j \in R,\, k \in K$	(2)
(balance)	$\sum_{j \in V} x^k_{ij} - \sum_{j \in V} x^k_{ji} = 0$	$i \in V,\, k \in K$	(3)
(start)	$\sum_{j \in V} x^k_{v_0 j} = 1$	$k \in K$	(4)
(end point)	$\sum_{j \in V} x^k_{j v_e} = 1$	$k \in K$	(5)
(P $\rightarrow$ D)	$s^k_{t_d} - s^k_{o_d} \geq 0$	$k \in K,\, d \in R$	(6)
(Complete)	$M_s \phi_p + \sum_{d \in R_p} (1 - x^k_{o_d j}) \geq 0$	$k \in K\, p \in C$	(7)
(Timelimit)	$s^k_{t_d} - s^k_{o_d} - t^k_{o_d t_d} \leq H$	$k \in K,\, d \in R$	(8)
(Connect)	$s^k_i + t^k_{ij} + M(x^k_{ij} - 1) \leq s^k_j$	$k \in K,\, (ij) \in E$	(9)
(Handover)	$\sum_{k \in K} s^k_i = \sum_{k \in K} s^k_j$	$(i,j) \in \delta$	(10)
(Journey)	$\sum_{k \in K} s^k_i \leq \sum_{k \in K} s^k_j$	$(ij) \in \lambda_p$	(11)
(Release)	$a_i \leq s^k_i + a_i(1 - \sum_{j \in V*} x^k_{ij})$	$i \in V*,\, k \in K$	(12)
(Due)	$b_i \geq s^k_i + b_i(1 - \sum_{j \in V*} x^k_{ji})$	$i \in V*,\, k \in K$	(13)
(Load)	$l^k_i + l'_j - M_l(x^k_{ij} - 1) \leq l^k_j$	$(ij) \in E,\, k \in K$	(14)
(Capacity)	$l^k_i \leq C_k$	$i \in V,\, k \in K$	(15)
(Emb)	$\sum_{k \in K} x^k_{ij} = 0$	$j \in V \setminus V_f, i \in P \cap V_f, f \in F$	(16)
(Emb load)	$\sum_{j \in V \setminus V_f} (C_k x^k_{ij} - l^k_j) \geq 0$	$k \in K, i \in D \cup V_f, f \in F$	(17)
(Variables)	$x^k_{ij} \in \{0,1\}$	$k \in K,\, (ij) \in E$	(18)
	$\phi_p \in \{0,1\}$	$p \in C$	(19)
	$s^k_i \in \mathbb{R}^+_0$	$i \in V \cup \{p_k\},\, k \in K$	(20)
	$l^k_i \in \mathbb{Z}^+_0$	$i \in V,\, k \in K$	(21)

The objective function (1) sums all the excess time used on the segments and adds a penalty if a PRM is not delivered. The penalty depends on whether PRM p is prebooked ($p \in B$) or immediate ($p \in C \setminus B$). Constraints (2) ensure that for each segment any PRM picked up is also delivered. Constraints (3) ensure that transport objects leaves all pickup or delivery vertices they enter. Constraints (4) ensure that all transport objects leaves their base. Constraints (5) ensure that all transport objects return to their base. Constraints (6) ensure that on each segment a PRM is picked up before it is delivered. Constraints (7) ensure that any PRM with at least one segment not assigned generates exactly one penalty in the objective. Constraints (8) ensure that the excess

time used on segment d does not exceed H. Constraints (9) ensure that if an edge (i, j) is used the time vertex j is visited is greater than the time vertex i is visited plus the travel time on edge (i, j). Constraints (10) ensure that delivering transport object meets pickup transport object for at delivery pickup handover vertex pair in δ. Constraints (11) ensure that the segments of the journey are completed in the right order. Constraints (12) and (13) ensure that the segments are started and ended within their given time window. Constraints (14) ensure that the load is updated when a PRM is picked up or delivered. Note that since load is increased for any a pickup vertex in V then constraints (14) together with constraints (9) ensure a connected route. This is true under the general assumption that the transport from pickup to the delivery point is always greater than zero. Constraints (15) ensure that the capacity is not exceeded. Constraints (16) and (17) enforce the embarkment conditions of only starting embarkment tasks on the same flight before completing a embarkment segment. Constraints (16) only allow edges going from a pickup vertex of an embarkment segment to vertices of embarkment on the same flight. Constraints (17) ensure that when using and edge between an embarkment vertex and any vertex not belonging to an embarkment request for the same flight the load must be zero.

4 Solution method

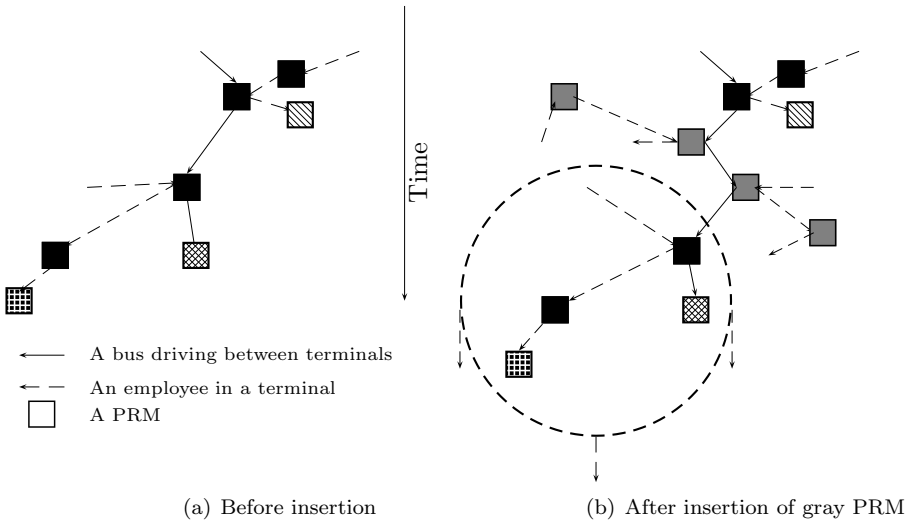
The solution method we present is a greedy insertion heuristic combined with simulated annealing.

In the survey by Cordeau and Laporte [5] from 2007 a list of some of the methods used for the dial-a-ride problem with multiple vehicles is provided. In this list the only exact methods are a branch and cut method optimizing on vehicle travel cost by Cordeau [6] and an improvement on this method by Ropke et al. [14]. The exact method has been tested on a maximum 96 requests and 8 vehicles, which was solved in 71 minutes.

The dial-a-ride problems are usually solved by heuristics as the problems are often real-life problems. Real life problems generally contains some additional constraints, which can be complicating and the objective varies. Moreover in real-life there can be constraints or desires not defined in the problem, which arises after the problem definition. Due to this an optimal solution might actually not be the best solution for the users.

Since the problem covered here is a dial-a-ride problem with complicating synchronization and embarkment constraints it is natural to consider heuristic solution methods. Moreover since the requirement is to solve instances with between 900 and 1500 requests within 2 minutes a heuristic method seems to be the only option.

Jaw et al. [12] in 1986 reports finding a good solution to their dial-a-ride problem on an instance with 2617 requests and 28 vehicles using an insertion sort method. Given the machines available in 1986 the solution is found quickly and the method would on the machines available today satisfy the solution time requirement. Other heuristic methods that are able to solve dial-a-ride problems with a large number of requests are Diana and Dessouky [8] using regret insertion solving problems with a 1000 requests and Xiang et al. [16] using a local search heuristic solving problems with 2000 requests.



(a) Before insertion (b) After insertion of gray PRM

Figure 2: A section of the routes when inserting the gray PRM. The part of the graph inside the dashed circle is moved to a later time because of propagated delays caused by the insertion of the gray PRM. The patterned PRMs are the next PRMs to be handled by the employee or bus.

The synchronization constraints present in the airport PRM transport problem (APRMTP) do add some complexity to the generation of feasible solutions and the calculation of the objective value. When a segment is inserted in a route it may have influence on not only the travel times of the later segments in the route, but also the other segments of the PRM and the segments of their routes and so forth. This means that every time a segment end and start time is changed it may generate a cascade of changes on related segments. Therefore, when checking for feasibility one may, in worst case, have to evaluate the feasibility of all the segments in the problem . The same is true when calculating the objective as the insertion of a segment may influence the travel time on all remaining segments in all the routes. However, together with a constraint on the maximum excess time allowed on each segment it may also constrain the problem significantly. An example of this is that the synchronization constraint reduces the number of possible feasible solutions.

We have constructed an insertion heuristic for the initial solution, which is described in the next section and later used in a simulated annealing scheme to improve solutions. The greedy insertion heuristic will given an ordered list lead to a deterministic solution found with a search for best insertion spot while the simulated annealing broadens the search by randomly selecting a neighborhood from a large neighborhood space and accepting solutions with a worse objective value. This method is similar to GRASP [10] as there at each iteration is a greedy construction. However, to avoid the in this case hard task of evaluating candidates we have a fixed order. The routes in the constructed solution are in the APRMTP very interdependent and therefore it would be very difficult for a local search to find better solutions using neighborhoods on the constructed solution. Instead of performing a local search on the constructed solution as done in GRASP we perform the local search on the fixed candidate lists given for the previous solution. Because of this reverse execution of the GRASP structure the algorithm presented could be described as a reversed GRASP.

Figure 2 shows how PRM segments are moved when inserting a PRM segment into the route represented by the solid line.

4.1 Insertion heuristic

To quickly find a feasible solution we have created a greedy insertion heuristic (GIH). The heuristic takes two lists of PRMs one containing the prebooked PRMs and one containing the immediate PRMs. The insertion heuristic inserts first the PRMs in the prebooked list then the PRMs in the immediate list by going through the list in sequential order. The reason for this is that it is very important for the service provider to serve the prebooked PRMs.

We have sorted the lists by earliest pickup time of the PRM, starting at the earliest. For each PRM the segments are inserted in the order they appear in the journey and the next PRM is not inserted before all segments of the previous PRM are inserted.

For each segment only the set of transport objects working in the graph containing the given segment are investigated for an insertion place. The segment is inserted in the place and transport object where it creates the least increase in the total excess time. We only allow an insertion to push the time of the other segments forward. Therefore, when checking for feasibility we only need to go through the segments with larger delivery times.

Usually when minimizing the route cost as in pickup and delivery problems it is possible to calculate the new objective by the difference in the time introduced by the insertion and removal however, in our case we did not do this as we found it too complicating when minimizing excess time and number of undelivered PRMs especially when there are propagating delays induced by the synchronization constraints.

GIH(P_1, P_2)

```
1:  for each p PRM first in list P1 then the list P2  do
2:    for each segment s in p do
3:      for each employee e serving s  do
4:        for each vertex v1,v2 in e in the possible time interval for s do
5:          if load and time feasible insert s_start before v1 and s_end before v2
            then
6:            calculate total excess time for all inserted segments;
7:          end if
8:        end for
9:        Select s1 and s2 where the least excess time is generated;
10:     end for
11:     if insertion was not possible then
12:       Delete already inserted segments of p;
13:       Register p as not inserted;
14:     else
15:       insert s in the e where the least excess time is generated;
16:     end if
17:   end for
18: end for
```

In the pseudo code of **GIH** the lines 1 and 2 contributes to the complexity

of **GIH** with the total number of requests $|R|$, which we here call n. The combinations that occur in line 3 and 4 can be n^2. Checking for feasibility in line 5 is done by a depth first search which at most goes through $2n$ vertices updating their times and the edge load. The calculation of total excess time of all inserted segments in line 6 is done by adding up excess time of all inserted segments, which is at most n. Therefore, the asymptotic running time of the greedy insertion sort is $O(n^4)$.

However, in the test cases provided by the company the time windows are quite tight and not all employees are available in the area of a given segment therefore there are often only a few locations where feasibility is actually checked and excess time calculated.

4.2 Simulated annealing

A Simulated annealing algorithm using the two lists of PRMs as an abstract representation was implemented. The initial solution is the greedy insertion heuristic on the two lists of PRMs, sorted by earliest possible start time. At each iteration in the simulated annealing algorithm a number of moves takes place to obtain a candidate solution x from the current solution x'.

The moves are made in the two PRM lists, which are then converted to a solution by the insertion heuristic. The moves are as follows:

- moving a not assigned PRM a random number of places forward in their respective lists
- to swap the place of two PRMs randomly selected within the same list.

Note, that the prebooked PRMs and immediate PRMs will always remain in their respective lists. The lists are when the moves are completed converted by the greedy insertion heuristic into a schedule.

Let the objective function be defined as $f(x)$ for a solution x. If $f(x) \leq f(x')$ then x is accepted. Otherwise we accept the solution with probability:

$$\exp^{f(x')-f(x)/T} \tag{22}$$

Details on the selected values for the temperature T will be covered in Section 6 on tunings, The temperature is decreased by a selected factor at each iteration. This decreasing factor is also called the cooling rate. The large neighborhood described and the acceptance probability allow the algorithm to escape local minima.

5 Data Instance and other parameter values

We were from the service company given a list of almost 5000 PRM request with information about the type and the position of the origin and destination. A travel time between the locations for the different transport forms was generated from this information. We have received 12 test cases from the company covering dates September 20 to October 1, 2009. These data sets contain between 374 and 555 PRMs each day. Some of the PRMs in the data set were removed before running the tests due to corrupted data for the PRM or that too little time was available for the PRMs journey so that a solution transporting the PRM can not

exist. This resulted in sets of between 353 and 495 PRMs. The data sets each had a set of employees for the given day and their assigned terminal or vehicle. The number of employees assigned on a day was around 120. The employees were assigned to 6 different terminals and 2 different bus types. For each bus type there is a type specific area of operation.

The capacities of the employees has been settled with the ground handling company as:

- Employee assisting inside terminal has capacity 4
- Bus between terminals has capacity 24
- Bus between gate and aircraft has capacity 18

For each PRM there is given a start time, an end time, a start location, an end location and a PRM type. There where six different types of PRMs in the data sets for each of the types we have assigned a volume as follows:

WCHC Cannot walk or stand. Needs wheel chair. Volume 3

WCHS Cannot walk up or down stairs. Volume 2

WCHR Cannot walk long distances. Volume 2

BLND Passenger is blind. Volume 2

DEAF Passenger is deaf. Volume 2

ASS Passenger cannot orient them selves. Volume 2

From the PRM data and the rules given by the company for the journey we generated the segments for each PRM given the arrival and departure location of the PRM. For each shift between 900 and 1500 segments where generated.

6 Tuning

Since our insertion algorithm has a complexity of $O(n^4)$, and the time allowed to solve the overall problem is limited to a few minutes, the number of iterations, which can be investigated in simulated annealing given the size of the problem instances is limited to a few hundreds. Hence, frequently the problem cases contain more PRMs than iterations performed in simulated annealing. Therefore, we consider the possibility for making several moves at each iteration. The moves are relocations in the list and thus doing more moves does not influence the running time significantly. However, the neighborhood becomes much larger and the previous solutions may be ruined by a large number of moves.

We first test the combination of different number of moves with different cooling rates given an fixed initial temperature. From the tuning tests a good combination of cooling rate and number of moves is selected and used in the test of the different possibilities for the initial temperature. The tuning is done of a solution time of 2 minutes as this is the requirement for the solution time given by the users.

The initial temperature is adjusted so that the probability of selecting a solution, which is exactly t percent greater than the initial solution is 50%.

For finding the best combination of the cooling rate and the number of moves, we have fixed the initial temperature so that a solution 5% worse than the initial solution must initially be accepted with 50% probability. This means that given an initial solution x and the temperature parameter of t then the initial temperature T is calculated as follows:

$$T = -tx/\log(0.5) \tag{23}$$

We made the choice of including the initial objective value in the generation of the initial temperature so that the variance in the size of the different problems does not influence the acceptance rate.

The three test cases used for tuning were randomly selected from the data sets received. Note that the maximum excess time on a segment $H = 30$ for all tests runs as this generally matches the requirement of the airports.

Case	prms	deleted	time	init NAP	init NAI	init sol
20090920	353	21	120	0	4	3566
20091001	474	45	120	1	1	4401
20090926	374	27	120	0	0	1782

Table 1: The test cases used for tuning, with the results from the greedy insertion heuristic.

In Table 1 we report the characteristics of each data instance and the values of the initial solution constructed with the greedy insertion heuristic on the lists, where the PRMs are sorted by earliest arrival time. The name of the test case is given in column one. In column two the number of PRMs in the test case is provided and in column three the number of PRMs that was deleted from the initial data set due to corrupted data. This means that column two and column three gives the number of PRMs in the initial data set. In column four the time in seconds allowed for simulated annealing is given. In all of the tuning cases we have used a solution time limit of 2 minutes. Column five reports the number of unassigned prebooked PRMs (NAP) in the initial solution and column six reports the number of unassigned immediate PRMs (NAI). In column seven the value of the initial solution is reported.

In Table 2 we have fixed the temperature parameter $t = 0.05$. The tuning is done for each of the test instances described in Table 1. For each test case and each combination of cooling rate and number of moves given in respectively column two and three the algorithm is run ten times and the average solution of the ten runs is reported in column four. The average number of iterations completed within the two minutes each run lasted is reported in column five. The standard deviation of the solutions is reported in column six. The average number of unassigned prebooked PRMs and immediate PRMs is reported in respectively column seven and eight. In column nine the best solution of the ten runs is given. The gap reported in column ten is the percent wise gap between the initial solution and the average solution found by simulated annealing calculated as:

$$gap = \frac{initsolution - averagesolution}{initsolution} \cdot 100$$

Note that by using the average solution reported in column three the gap represents the expected improvement for a single 2 minute run of the simulated annealing algorithm.

testcase	coolrate	moves	average sol	av ite	stdD	av NAP	av NAI	best sol	gap
20090920	0.5	4	2655.2	326.5	179	0	2.3	2491	25.5%
	0.8	4	2732.2	325.9	187	0	2.5	2457	23.4%
	0.9	4	2720.2	326.7	242	0	2.4	2426	23.7%
	0.95	4	2753.5	324.5	201	0	2.5	2511	22.8%
	0.99	4	2912.2	322.1	239	0	2.5	2605	18.3%
	0.5	12	2495.8	315.8	57	0	2.0	2396	30.0%
	0.8	12	2496.3	316.7	58	0	2.0	2396	30.0%
	0.9	12	2510.4	315.5	34	0	2.0	2461	29.6%
	0.95	12	2500.0	310.1	55	0	2.0	2431	29.9%
	0.99	12	2609.3	306.3	52	0	2.0	2541	26.8%
	0.5	20	2521.5	306.1	79	0	2.0	2351	29.3%
	0.8	20	2505.6	309.9	47	0	2.0	2418	29.7%
	0.9	20	2515.2	308.8	33	0	2.0	2450	29.5%
	0.95	20	2495.0	301.2	53	0	2.0	2437	30.0%
	0.99	20	2564.3	298.7	46	0	2.0	2464	28.1%
20091001	0.5	4	2691.7	125.3	198	0	0.4	2461	38.8%
	0.8	4	2796.7	125.4	130	0	0.8	2521	36.5%
	0.9	4	2794.8	123.2	120	0	0.5	2543	36.5%
	0.95	4	2737.5	120.0	187	0	0.2	2319	37.8%
	0.99	4	3149.8	117.7	205	0	0.7	2869	28.4%
	0.5	12	2838.9	120.8	111	0	0.3	2736	35.5%
	0.8	12	2742.0	115.1	161	0	0.1	2510	37.7%
	0.9	12	2723.3	114.4	190	0	0.3	2472	38.1%
	0.95	12	2749.2	107.7	126	0	0.3	2598	37.5%
	0.99	12	2893.6	104.1	130	0	0.2	2677	34.3%
	0.5	20	2891.6	113.2	131	0	0.1	2789	34.3%
	0.8	20	2834.0	110.4	141	0	0.2	2635	35.6%
	0.9	20	2820.9	106.1	184	0	0.3	2459	35.9%
	0.95	20	2831.5	103.3	136	0	0.4	2644	35.7%
	0.99	20	3048.1	100.0	129	0	0.6	2842	28.1%
20090926	0.5	4	1442.6	255.1	40	0	0	1378	19.0%
	0.8	4	1427.9	256.5	50	0	0	1368	19.9%
	0.9	4	1428.5	254.8	41	0	0	1352	19.8%
	0.95	4	1480.8	254.8	48	0	0	1395	16.9%
	0.99	4	1563.0	250.1	80	0	0	1437	12.3%
	0.5	12	1482.0	248.0	50	0	0	1425	16.8%
	0.8	12	1484.3	245.1	42	0	0	1420	16.7%
	0.9	12	1431.5	245.2	52	0	0	1322	19.7%
	0.95	12	1491.7	241.6	70	0	0	1380	16.3%
	0.99	12	1508.8	236.1	78	0	0	1399	15.3%
	0.5	20	1508.4	243.0	50	0	0	1400	15.4%
	0.8	20	1496.7	240.2	57	0	0	1424	16.0%
	0.9	20	1490.5	237.1	73	0	0	1370	16.4%
	0.95	20	1471.1	234.5	55	0	0	1347	17.4%
	0.99	20	1520.9	228.9	55	0	0	1422	14.7%

Table 2: Tuning cooling rate and number of moves

Analyzing the results in Table 2 using ranking of the gap for each of the three data sets and comparing their ranks we have chosen the value 0.9 for the cooling rate and 12 moves at each iteration. The selected values for cooling rate and number of moves are used in the tuning test on values for the initial temperature shown in Table 3.

Case	temperature	av best sol	av it	stdD	av NAP	av NAI	best sol	gap
20090920	1%	2490.3	317.0	53	0	2.0	2379	30.2%
	5%	2510.4	315.5	34	0	2.0	2461	29.6%
	10%	2585.3	313.4	146	0	2.2	2454	27.5%
	15%	2542.9	312.6	36	0	2.0	2462	28.7%
20091001	1%	2812.2	107.0	175	0	0.4	2470	36.1%
	5%	2723.3	114.4	190	0	0.3	2472	38.1%
	10%	2717.9	109.2	130	0	0.2	2538	38.2%
	15%	2774.5	108.8	234	0	0.3	2428	37.0%
20090926	1%	1470.3	248.3	64	0	0	1387	17.5%
	5%	1431.5	245.2	52	0	0	1322	19.7%
	10%	1473.8	241.2	54	0	0	1370	17.3%
	15%	1486.4	242.0	44	0	0	1384	16.6%

Table 3: Tuning initial temperature

The columns in Table 3 are structured the same way as Table 2, except that the columns reporting cooling rate and moves in Table 2 are replaced by a single temperature column in Table 3. The results in Table 3 show that the best initial solution is obtained bychoosing the point where a solution will be accepted with 50% probability as 5% under the initial value. These values will be used for the tests in Section 7. The values found in this section for the simulated annealing determines the intensification and diversification of the heuristic. The high number of moves at each iteration and the acceptance probability ensures

diversification while the cooling rate generates an intensification. The start temperature determines the start diversification generated by the acceptance probability.

7 Test Results

We have received 12 test cases from the service company covering dates September 20 to October 1, 2009. Unfortunately, the company does not have records on how the PRMs actually where schedule for those test cases, as the system mainly takes care of registering the PRMs arriving, what segments needs to be completed and which employees are available for completing them. Hence, we cannot directly compare our schedules with the historic data.

It should be noted that in the data sets a little more than half of the PRMs are prebooked. Three of the twelve data sets has been used for tuning, however, we still include them in this test section. The test are run on a computer with a 64 bit Intel Xeon 2.67 GHz CPU.

All the simulated annealing tests reported in this section has been run with:

- Temperature: we use the factor $t = 0.05$ to adjust the temperature T according to (23).
- Cooling rate: 0.9
- Moves at each iteration: 12

Note that temperature is calculated given an initial solution x and the temperature parameter of t as described in equation (23).

In Table 4 we report the results of running the greedy insertion heuristic on the data set where the PRMs are sorted by earliest arrival time for all 12 data sets. For each data set the excess time allowed on a segment is again $H = 30$ minutes corresponding to airport requirements.

In column one the name of the data set is described by the date of the shift. We report how many PRMs in the given data set we had to remove due to corrupted data for the PRM or that too little time was given for the PRMs journey so that a solution transporting the PRM can not exists. Note that the number of PRMs reported in column two is the number of PRMS after the removal of the deleted PRMs reported in column three from the initial set. The number of prebooked passengers not assigned (NAP) and not assigned immediate passengers (NAI) in the initial solution is reported in respectively column four and five. Finally the initial solution retrieved from the greedy insertion heuristic on the PRM lists sorted by earliest arrival time is reported in column six.

From Table 4 it can be seen that for 9 of the 12 instances there is no prebooked PRM rejected. The number of rejected immediate PRMs is also quite low compared to the total number of PRMs when solving the problem using just the greedy insertion heuristic.

The results in Table 4 are used for evaluating the test results of the simulated annealing algorithm presented in Table 5.

In Table 5 we have for each data set tested the simulated annealing for 120 seconds (2 minutes), 300 seconds (5 minutes) and 3600 seconds (1 hour) as given in column three. For the tests allowed 120 second and 300 seconds each test

Case	prms	prms deleted	NAP	NAI	sol
20090920	353	21	0	4	3566
20090921	391	35	7	5	11430
20090922	428	23	0	5	5162
20090923	361	42	0	0	2001
20090924	432	33	0	3	4062
20090925	438	46	0	2	3500
20090926	374	27	0	0	1782
20090927	397	32	0	2	3095
20090928	378	23	0	0	2208
20090929	419	37	0	1	3259
20090930	495	60	7	9	13443
20091001	474	45	1	1	4401

Table 4: The results of the insertion heuristic run on the different data sets. Note, that this is also the initial solution used by the simulated annealing heuristic.

Case	# runs	time (s)	av sol	av it	stdD	av NAP	av NAI	best sol	gap
20090920	10	120	2504.6	279.5	61	0	2.0	2423	29.8%
20090920	10	300	2426.9	767.6	52	0	2.0	2361	31.9%
20090920	3	3600	2223.3	7812.3	20	0	2.0	2197	37.7%
20090921	10	120	9026.0	209.0	369	5.6	3.6	8227	21.0%
20090921	10	300	8658.3	499.4	430	5.6	3.0	8112	24.2%
20090921	3	3600	7713.7	4063.3	663	5.3	2.3	7205	31.5%
20090922	10	120	3233.5	131.1	315	0	1.3	2922	37.4%
20090922	10	300	2838.9	317.8	85	0	1.0	2736	45.0%
20090922	3	3600	2508.0	2348.3	74	0	1.0	2425	51.4%
20090923	10	120	1659.4	244.0	50	0	0	1589	17.1 %
20090923	10	300	1550.1	592.1	45	0	0	1464	22.5%
20090923	3	3600	1419.0	4420.7	34	0	0	1375	29.3%
20090924	10	120	2822.5	141.5	142	0	1.1	2670	30.6%
20090924	10	300	2578.8	309.7	85	0	1.0	2415	36.5%
20090924	3	3600	2197.3	2716.7	24	0	1.0	2163	45.9%
20090925	10	120	2051.8	150.4	60	0	0	1935	41.4%
20090925	10	300	1974.0	367.2	38	0	0	1910	43.6%
20090925	3	3600	1749.3	4252.3	13	0	0	1732	50.0%
20090926	10	120	1481.0	246.0	39	0	0	1422	16.9%
20090926	10	300	1351.9	597.3	53	0	0	1303	21.1%
20090926	3	3600	1139.0	5854.3	28	0	0	1103	36.1%
20090927	10	120	1899.4	212.7	34	0	0	1852	38.6%
20090927	10	300	1770.9	516.7	70	0	0	1672	42.8%
20090927	3	3600	1512.3	5927.3	29	0	0	1477	51.1%
20090928	10	120	1623.8	224.2	49	0	0	1533	26.5%
20090928	10	300	1512.9	544.8	46	0	0	1445	31.5%
20090928	3	3600	1329.3	5586.3	36	0	0	1281	39.8%
20090929	10	120	2221.3	123.8	85	0	0	2043	31.8%
20090929	10	300	2102.3	302.1	48	0	0	2003	35.5%
20090929	3	3600	1855.7	2826.0	47	0	0	1800	43.1%
20090930	10	120	7351.9	103.4	1478	2.8	5.0	4051	45.3%
20090930	10	300	5142.8	250.1	1264	1.3	3.3	3992	61.7%
20090930	3	3600	3524.3	2141.0	338	1.0	0.7	3247	73.9 %
20091001	10	120	2777.1	112.1	168	0	0.3	2554	36.9%
20091001	10	300	2470.0	270.0	162	0	0.1	2230	43.9%
20091001	3	3600	2062.3	1925.3	17	0	0	2038	53.1%

Table 5: The results of simulated annealing running for two minutes, five minutes and one hour.

is repeated ten times. However, for the tests running 3600 seconds each test is only repeated three times as given in column two.

The temperature, cooling rate and number of moves are set to the selected values and the excess time allowed on each segment is set to $H = 30$ (minutes).

In column four the average of the solution for the runs is reported. The average number of iterations of the runs and the standard deviation of the solutions is reported in column five and six. In column seven and eight respectively the average number of unassigned prebooked PRMs and unassigned immediate PRMs are reported. The best solutions of all the runs is reported in column nine and the gap between the average solution and the initial solution from Table 4 is reported in column ten. The gap is in Table 5 calculated as described in Section 6. The results in Table 5 show that after running simulated annealing for two minutes the initial greedy heuristic solution is significantly improved. Note, that more solution time improves the solution quality in all cases and in some cases this improvement is significant. However, the improvements achieved from the initial solution in the first five minutes is in all cases greater than the improve-

ments achieved in the following fifty five minutes. The improvement achieved in the first two minutes from the initial solution is greater in all cases except one (20090926) than the improvement achieved in the following fifty eight minutes. This indicates that the algorithm is good at finding improvements early in the algorithm and therefore works well with the requirement of solutions within a couple of minutes.

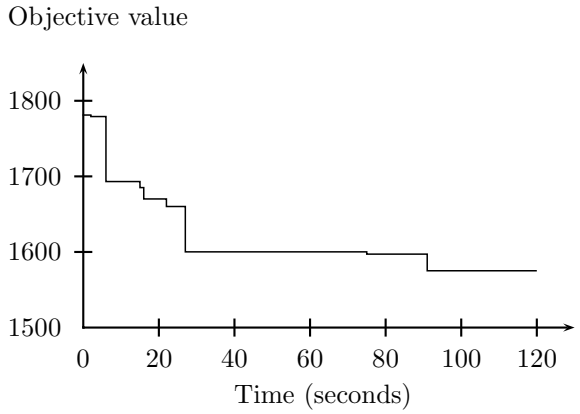


Figure 3: The solution values over time for test case 20090926

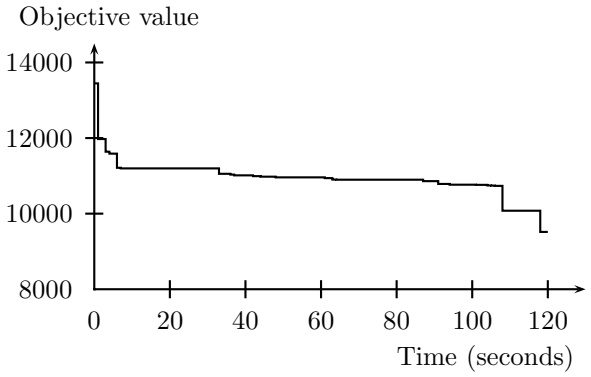


Figure 4: The solution values over time for test case 20090930

Since the simulated annealing algorithm is tuned for runs of 2 minutes it is obvious that the acceptance of worse solutions will occur in the first 2 minutes. To show the improvement of solution value for the required solution time we have in graphs 3 and 4 shown the runs for only the first 2 minutes.

The graphs in Figure 5 and 6 show the development in solution values for a single 1 hour run of respectively test instance 20090926 and 20090930. For the test instance 20090926, Figure 5 show that after running simulated annealing for 15 minutes the improvements to the solutions become seldom and insignificant, for the test instance 20090930 this is reduced to 10 minutes. This means that the significant reductions occurs early in the simulated annealing and very good

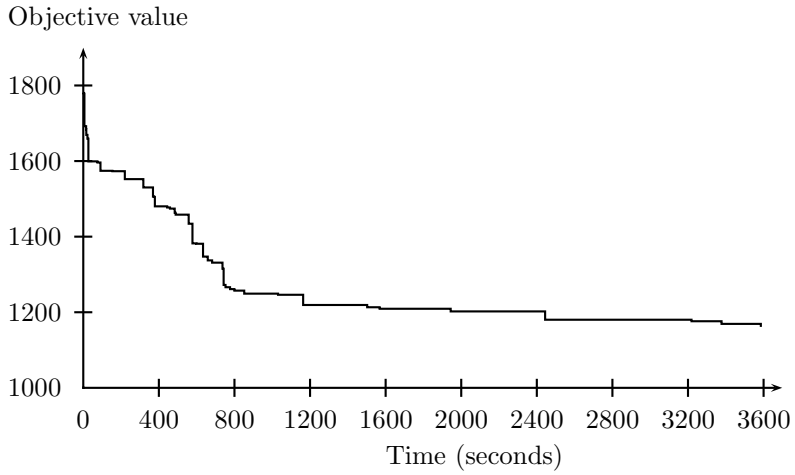


Figure 5: The solution values over time for test case 20090926

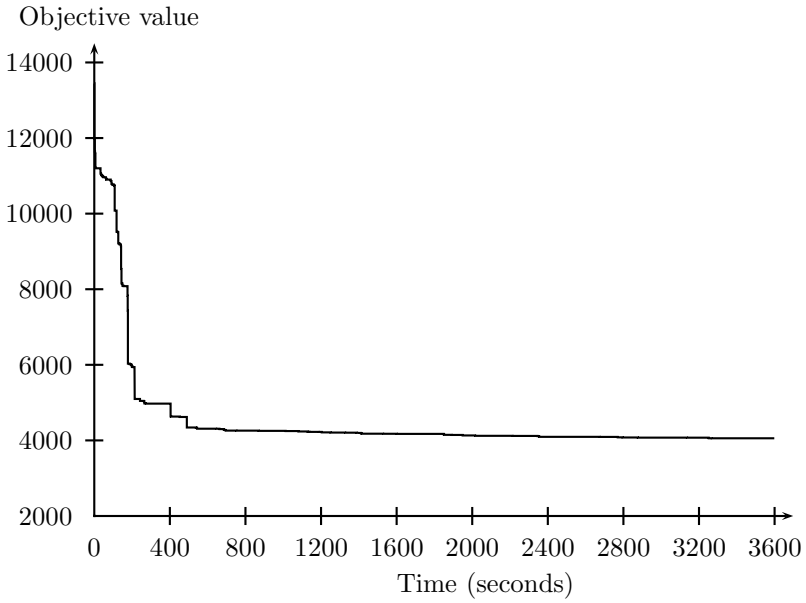


Figure 6: The solution values over time for test case 20090930

quality solutions can be found after 10–15 minutes. In our case the solution time required is quite limited however, if significant solution times where permitted making a random restart of the algorithm with a new ordering of the lists may allow for searches in other areas of the search space. However, since the neighborhood used is very large it is easier for the algorithm to escape local minima.

8 Conclusion

We have presented a model for the airport PRM transport problem (APRMTP) and developed a heuristic with promising results even when the running time is two minutes as required by the service company. Moreover the number of rejected PRMs is very low also in the initial greedy insertion heuristic solutions.

Although the problem has been defined incorporation with a specific handling company, we believe that the developed model is sufficiently general to cover most airports in the world.

The program developed seems to work very well with the short solution time constraint as the big improvements are obtained within the first few minutes.

From the tests and the solution graphs it is clear that increasing the solution time a little could in some cases give significant improvements. In stead of increasing the solution time such improvements could also be achieved by increasing computational power or if possible algorithmic improvements such as parallelization of the program.

The next step in the academic sense would be to test different strategies on the problem presented and compare the solutions quality and to reduce computation time for the greedy insertion heuristic. In the application sense the next step would be to in cooperate this method at the users and to see if the use of our plan can improve their daily service.

We have assumed that the personnel and the location of the personnel is fixed, but the short solution times used by the developed algorithm makes it possible to use local search to test whether relocation of some personnel may lead to a higher service level. Since the number of passengers arriving at the airport is increasing the algorithm can be used to find when an increase in personnel is needed to deliver acceptable service to the increasing volumes of PRMs arriving.

Acknowledgments

The authors wish to thank Torben Barth and Berit Løfstedt for valuable comments. The authors also wish to thank Jacob Colding for presenting and clarifying the problem to us.

References

[1] Gatwick managing directors report, 2009.

[2] The world's top 50 airports. *Air Transport World*, 47:51, 2010.

[3] J. W. Baugh, G. K. R. Kakivaya, and J. R. Stone. Intractability of the dial-a-ride problem and a multiobjective solution using simulated annealing. *Engineering Optimization*, 30:91–123, 1998.

[4] P. Chen, Y. Guo, A. Lim, and B. Rodrigues. Multiple crossdocks with inventory and time windows. *Computers & Operations Research*, 33:43–63, 2006.

[5] J.-F. Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54:573–586, 2006.

[6] J.-F. Cordeau and G. Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153:29–46, 2007.

[7] T. G. Crainic, N. Ricciardi, and G. Storchi. Models for evaluating and planning city logistics systems. *Transportation Science*, 43:432–454, 2009.

[8] M. Diana and M. M. Dessouky. A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Transportation Research Part B*, 38:539–557, 2004.

[9] A. Dohn, M. S. Rasmussen, and J. Larsen. The vehicle routing problem with time windows and temporal dependencies. Technical report, DTU Management, The Technical University of Denmark, 2009.

[10] T. Feo and M. Resende. Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6:109–133, 1995.

[11] I. Ioachim, J. Desrosiers, F. Soumis, and N. Belanger. Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, 119:75–90, 1999.

[12] J. Jaw, A. Odoni, N. Psaraftis, and H. M. Nigel. A heuristic algorithm for the multi-vehicle advance request deal-a-ride problem with time windows. *Transportation Research part B*, 20:243–257, 1986.

[13] D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34:2403–2435, 2007.

[14] S. Ropke, J.-F. Cordeau, and G. Laporte. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks*, 49:258–272, 2007.

[15] M. Wen, J. Larsen, J.-F. Cordeau, and G. Laporte. Vehicle routing with cross-docking. *Journal of the Operational Research Society*, 60:1708–1718, 2009.

[16] Z. Xiang, C. Chu, and H. Chen. A fast heuristic for solving a large-scale dial-a-ride problem under complex constraints. *European Journal of Operational Research*, 174:1117–1139, 2006.

Major airports have an average throughput of more than 100,000 passengers per day, some of which will need special assistance. The largest airports have a daily average throughput of more than 500 passengers with reduced mobility. A significant number of people and busses are assigned to provide transportation for the passengers with reduced mobility. It is often necessary for a passenger with reduced mobility to use several different modes of transport during their journey through the airport. Synchronization occurs at the locations where transport modes are changed as to not leave passengers unattended. A description of the problem together with a mathematical model is presented. The objective is to maximize the quality of service by scheduling as many of the passengers as possible, while ensuring a smooth transport with short waiting times. A simulated annealing based heuristic for solving the problem is presented.

The algorithm makes use of an abstract representation of a candidate solution which in each step is transformed to an actual schedule by use of a greedy heuristic. Local search is performed on the abstract representation using advanced neighborhoods which modify large parts of the candidate solution. Computational results are reported showing that the algorithm is able to find good solutions within a couple of minutes, making the algorithm applicable for dynamic scheduling. Moreover high-quality solutions can be obtained by running the algorithm for 15 minutes.

ISBN 978-87-92706-00-3

DTU Management Engineering
Department of Management Engineering
Technical University of Denmark

Produktionstorvet
Building 424
DK-2800 Kongens Lyngby
Denmark
Tel. +45 45 25 48 00
Fax +45 45 93 34 35

www.man.dtu.dk