Solving the bus planning problem

5.1 Introduction

At Amsterdam Airport Schiphol (AAS) one can distinguish two types of stands where aircraft can be assigned to. There are the regular gates that are equipped with a passenger air bridge, and there are the so-called remote stands which do not have a bridge connection to the terminal. When an aircraft is assigned to a stand with a passenger air bridge, the passengers can (dis)embark the aircraft via this bridge. On the other hand, when an aircraft is assigned to a remote stand, the passengers will need to be transported to or from the terminal building. Since it is not an option that passengers just walk from the remote stand to the terminal building or vice versa, buses are used to transport them.

Since there are a limited number of buses available we need to determine which bus will drive from where to where at which time. This problem we will refer to as the bus planning problem.

When we look at the characteristics of the bus planning problem, one can
see similarities with some other problems. One related problem is the vehicle scheduling problem of buses in public transport. In both problems we need to determine which buses will drive what rides. A difference between the two problems is the fact that with bus planning at AAS only short rides (in the order of minutes) need to be assigned to a bus, while in the case of public transport the rides are complete lines from beginning to end and they take much longer (generally in the order of one hour or more). Furthermore, at an airport the distances are very limited and generally the time needed to drive from the end point of one ride to the start point of the next ride is very little. This means that we can be very flexible with regards to creating combinations of rides that need to be driven by one bus. This in contrast to the vehicle scheduling problem in public transport, where the time needed to drive from the end point of one ride to the start point of the next ride can be significant.

The problem also has similarities with various problems in the field of Automated Guided Vehicles (AGV). For example, the problem of routing AGV in a container terminal is related to bus planning at an airport because in both cases the goal is to assign rides to a vehicle. One major practical difference between bus planning and routing AGV is the fact that AGV have no drivers and thus there is no need for mandatory (lunch)-breaks as required by law. Le-Anh and de Koster (2006) give a review of the design and control of AGV systems. They not only discuss the online case, but also give an overview of the offline scheduling of AGV. This offline scheduling problem is similar to bus planning, since in both problems we know everything in advance and need to create a schedule subject to certain criteria. They indicate that the offline scheduling problem of AGV is similar to the Pickup and Delivery Problem with Time Windows (PDPTW).

More recently, Ropke and Cordeau (2006) proposed an algorithm for solving the PDPTW that is based on a column generation approach. Though the bus planning problem at an airport has some similarities to the PDPTW, one major difference is that at an airport, the buses must drive straight from the pickup point to the delivery point and cannot pick up more passengers in the meantime and thus the time windows are very strict. Furthermore, the same remark made earlier regarding the limitation on the distances also holds for the comparison with the PDPTW. So bus planning in this way is a special version of the PDPTW.
5.2 Problem description

As mentioned in the previous section, platform buses are used to transport the passengers to or from the aircraft standing on remote stands. For this transportation two types of buses are used at AAS. The first type of bus is called the *peak bus*, which are standard buses that are also used for city trips in public transportation. Peak buses are capable of transporting up to 50 persons at once.

The second type of bus is the *Cobus*. It is a bus that is specifically designed for use at airports. Compared to the peak bus it has several advantages, a clear one is that it is larger and can transport up to 70 persons at once. Other advantages are that the bus has a lower floor (thus giving easy access) and it has doors on both sides. This latter advantage is particularly useful, since it will not matter from which direction an aircraft at the platform is approached.

Every season the planners at AAS look at the planned flights and estimate the number of buses needed in each 15 minute time interval of a day for the complete upcoming season. This information is then sent to the Haagse Tram Maatschappij (HTM) bus company and they will create *shifts* for buses and drivers such that they fulfill the capacity requirements for each of the 15 minute intervals. The shifts have four attributes, namely a *starting time*, an *ending time*, a *type of bus* that drives the shift, and the *number* of buses within the shift. Each bus within a shift is driven by one driver.

The planners at AAS don’t have a direct influence on these shifts and they must use the shift information as input for their planning. Looking at the type of bus driving the shifts, one can see that the majority of the shifts at AAS are driven by the Cobuses.

Considering passengers, we can see that for departing flights the passengers enter the bus at the bus gate at the terminal. The bus then drives to the aircraft at the platform after which the passengers disembark the bus to board the aircraft. The combination of these three events we call a *trip*. For arriving flights one can likewise define trips consisting of these three phases, but then the origin is the aircraft and the destination is the bus gate. Furthermore, the time needed for the passengers to embark or disembark the bus is set to five minutes, independent of the number of passengers and the type of bus.
Since the number of passengers of each flight is known (the airlines should provide this to AAS), one can determine the number of trips that is needed to transport all passengers to or from the aircraft. Due to the fact that the number of passengers that can be transported with a Cobus differs from the number of passengers that can be transported with a peak bus, sometimes not all trips actually need to be served. An example of such a situation would be the transportation of 60 passengers: we either need two peak buses for this, or just one Cobus. If we create two trips but one of these trips is driven by a Cobus, the second trip is redundant because all passengers already could be transported with the one Cobus.

After running preliminary experiments we decided that in the case where there are trips for a departing flight that are possibly redundant, depending on whether the other trips for the departing flight are driven by Cobuses, we will only allow Cobuses to drive the trips for this departing flight. The reason for deciding this is that the majority of the buses that are available at AAS are Cobuses. Furthermore, this situation does not occur often. One explanation for this is a preference at AAS to send a Cobus as the first bus to an aircraft in case of a departing flight. This means that for departing flights only situations where the number of passengers is between 120 (one Cobus and one peak bus) and 140 (two Cobuses) would result in a possibly redundant trip because we need to send at least two buses, but possibly three. These numbers of passengers are relatively high and such flights are seldomly assigned to remote stands.

The way the trips are generated is prescribed by AAS and is different for arriving and departing aircraft. For an arriving aircraft it is required that there is exactly the number of buses needed to transport all passengers that need to disembark after the aircraft has come to a full stop at the remote stand.

For a departing aircraft the creation of trips is different. One major difference is that regardless of the number of passengers there must always be at least two trips for a departing flight. Furthermore, in contrast to the trips for arriving aircraft, not all trips start at the same time. The last trip must be finished with disembarking the passengers five minutes before the actual departure of the flight. The trip before the last trip must be finished with disembarking the passengers 10 minutes before the actual departure of the flight, etc. Since we know the time it takes for embarking and disembarking the passengers in the bus and the time it takes to drive from the bus station to the remote stand, we can calculate at what time a trip should start, given its end time.
Our goal is to find a planning for the buses that is as robust as possible, where robust means that having a small change in arrival or departure times of flights during the actual day itself does not imply a lot of replanning. A possible measure is to look at the idle time between pairs of trips that are consecutively assigned to the same bus. The idle time between two trips $t$ and $t'$ that are assigned consecutively to the same bus with trip $t$ being the first trip, is defined as follows:

\[
\text{idle time}(t, t') = T_{t'}^{\text{start}} - T_t^{\text{end}} - \text{driving time}(t, t'),
\]

where $T_{t'}^{\text{start}}$ and $T_t^{\text{end}}$ denote the start and the end time respectively of a trip $t$. By subtracting the “driving time” from the end location of the first trip to the start location of the second trip, we clearly obtain a nett idle time between the two trips.

To achieve a robust schedule we want to ensure that all idle times between pairs of trips that are consecutively assigned to the same bus are as big as possible. To model this within a cost function we chose to use the cost function $c^B$ based on the one we used in Chapter 4 for the gate assignment problem:

\[
c^B(\text{idle}) = 1000(\arctan(-0.21 \times \text{idle}) + \frac{\pi}{2}),
\]

where idle is the idle time between two trips.

Driving times are provided by AAS in minutes and are based on reference points. In this system, each of the platforms is represented by one reference point. This means that the driving time from one end of a platform to another end of the same platform would take zero minutes of driving time, since the whole platform uses the same reference point. To overcome this problem of zero minute driving times and to build in a bit of robustness in the short driving times, all driving times are set to at least five minutes. These five minutes are sufficient to deal also with the fact that buses are not allowed to drive freely everywhere, but must give right to aircraft and sometimes must wait till they can drive behind a just parked aircraft, till engines are switched off, etc.

Using the above cost function we ensure that very short idle times get penalized very strongly, while bigger idle times get a lower cost. We did not make use of a so-called cut-off value (i.e. a threshold beyond which the cost will not decrease anymore if the idle time becomes bigger), since using such a cut-off value would introduce symmetry in the model. This is not preferable because it makes the resulting ILP problems more difficult to solve.
CHAPTER 5. SOLVING THE BUS PLANNING PROBLEM

From discussions with the planners at AAS, we learned that in the case of a departing flight, there is a preference to let the first trip to the aircraft be driven by a Cobus. This is not a mandatory rule, but in case of choice the planners prefer this. To model this, we use an additional penalty cost in case the first trip to a departing aircraft is driven by a peak bus.

5.3 Problem formulation

Similar to the gate assignment problem in Chapter 4 we have multiple ways of formulating the bus planning problem as an Integer Linear Program (ILP). One standard approach is to introduce binary variables indicating whether a bus serves one trip after another trip (this order is needed to calculate the cost because it is based on the time between two trips) but, as for gate assignment, this approach would result in a model that is way too large to handle.

To cope with this problem we will use an ILP formulation similar to the formulation we used in Chapter 4 for the gate assignment problem.

With the gate assignment problem we created gate plans that consist of flights that are to be assigned to the same physical gate. Now we will introduce bus plans, which consist of a set of trips that will be driven by one bus. Similar to letting gate plans correspond to gate types, we let bus plans correspond to shifts. For a given bus plan to be feasible we require that:

- all trips present in the bus plan can be realized within the start and end times of the associated shift;
- no two trips are conflicting (i.e. there must be enough time to drive from the destination of one trip to the origin of the next trip).

For the shifts that last longer than 4.5 hours (so-called long shifts), we need an extra constraint on the set of trips being driven. Whenever a driver has a shift that lasts more than 4.5 hours, he is required by law to have a 45 minute break that does not start within 1.5 hours of the shift start time and does not end within 1.5 hours of the shift end time. So somewhere around the middle of the shift, a 45 minute period of time should be reserved for a break. These 45
minutes do not include the time it takes the driver to drive from the last trip before the break to the canteen and from the canteen to the start of the first trip after the break.

One problem that is introduced by arriving flights with many passengers is that several trips are needed, all with exactly the same origin and destination. This results in symmetry in the model that we would rather not have. To resolve this situation we introduce so-called *supertrips*. A supertrip is like a normal trip, except that instead of having to be driven by exactly one bus, it must be driven by $q$ buses, where $q$ is determined by the number of passengers to be transported. So instead of having $q$ normal trips that all need to be driven once, we now have one supertrip that needs to be driven exactly $q$ times.

As for the gate assignment problem in Chapter 4 we formulate the bus planning problem as an ILP that uses this notion of bus plans in the following way. Suppose that we are given the complete set of all possible bus plans. To solve the problem, we have to select a subset from this complete set such that each trip is present in exactly one of the selected bus plans. We must ensure that we have as many bus plans selected for a given shift as there are buses within that shift. Furthermore, we can determine the cost $c^B_j$ of a bus plan $j$ by summing the cost of all successive pairs of trips in the bus plan. The objective now is to minimize the total cost of the set of the selected bus plans.

Introduce a binary variable $y_j$ for each bus plan $j$ as follows:

$$y_j = \begin{cases} 1 & \text{if bus plan } j \text{ is selected;} \\ 0 & \text{otherwise.} \end{cases}$$

To ensure each trip is driven the correct number of times, we add the following constraints:

$$UAT_t + \sum_{j=1}^{M} h_{tj} y_j = S_t,$$

where

$$h_{tj} = \begin{cases} 1 & \text{if trip } t \text{ is present in bus plan } j; \\ 0 & \text{otherwise,} \end{cases}$$

$M$ denotes the total number of available bus plans, and $S_t$ denotes the number of times trip $t$ needs to be covered, which is equal to one for all normal trips and for a supertrip it is equal to the number of trips that the supertrip is composed
out. Finally, UAT\(_{t}\) (unassigned trip) allows for trip \(t\) being unassigned in case not enough buses are present to drive all trips at a certain time. To ensure trips are assigned whenever enough buses are present, the UAT\(_{t}\) variable for each trip \(t\) gets a very high cost coefficient \(R_t\) in the objective function. The value of this cost coefficient has been determined through preliminary experiments.

To ensure we select as many bus plans for each shift as there are buses present for that shift we add the following constraint for each shift \(b\):

\[
\sum_{j=1}^{M} f_{jb} y_j = T_b, (5.2)
\]

where

\[
f_{jb} = \begin{cases} 
1 & \text{if bus plan } j \text{ corresponds to shift } b; \\
0 & \text{otherwise},
\end{cases}
\]

and \(T_b\) denotes the number of buses shift \(b\) consists of.

Now the complete model is as follows:

\[
\text{Minimize } \sum_{j=1}^{M} c_j B_j y_j + \sum_{t=1}^{T} R_t \text{UAT}_t
\]

subject to:

\[
\text{UAT}_t + \sum_{j=1}^{M} h_{tj} y_j = S_t \text{ for } t = 1, \ldots, T \quad (5.3)
\]

\[
\sum_{j=1}^{M} f_{jb} y_j = T_b \text{ for } b = 1, \ldots, B \quad (5.4)
\]

\[
y_j \in \{0, 1\} \text{ for } j = 1, \ldots, n. \quad (5.5)
\]

### 5.3.1 Pricing problem

We solve the model for the bus planning problem by means of column generation. Since the total number of possible bus plans is enormous, we work with a small subset of the possible columns. For this subset we solve the above model, which gives us a dual multiplier \(\phi_t\) for the constraint 5.3 corresponding to trip \(t\) and a
dual multiplier $\omega_b$ for the constraint 5.4 corresponding to shift $b$. The reduced cost of a bus plan $j$ for shift $b$ is equal to
\[
c_j = c_j - \sum_{t=1}^{T} h_{tj} \phi_t - \omega_b.
\]

After solving the above model with the subset of the bus plans, we need to determine whether there exists a bus plan with negative reduced costs. It is not possible to check the reduced cost of all bus plans, again because the total number of possible bus plans is enormous. In the pricing problem we want to find the bus plan with minimum reduced cost. As soon as we have found the minimum there are two options: the bus plan has negative reduced cost, meaning that adding this bus plan to the model might decrease the objective value. Or, the bus plan has reduced cost $\geq 0$, meaning that adding this bus plan will not decrease the objective value and thus the optimum is found.

To solve the pricing problem we introduce a Directed Acyclic Graph (DAG) $G_b = (V,E)$ for every shift $b$. We add a vertex to $G_b$ if trip $t$ is within the start and end time of shift $b$. We add an edge between two vertices if the two corresponding trips are not conflicting. Now a path through the graph $G_b$ represents a feasible bus plan for shift $b$ and vice versa, all feasible bus plans for shift $b$ can be represented by a path through the graph $G_b$.

The above only holds for shifts which do not require a mandatory break. In case we have a long shift, we have to provide a way of putting a break somewhere in the middle of the shift. We can do this by adding break vertices to the graphs corresponding to shifts which last longer than 4.5 hours in the following way:

- Add a break vertex between two trips which are far away in time, such that there is time for a break of at least 45 minutes and the driving time to and from the break location,

- Add a break vertex between the source vertex and any vertex of which the corresponding trip has a start time that is at least 135 minutes (i.e. 90 minutes buffer and 45 minutes break) later than the start time of the shift. This indicates a driver starts his shift and does nothing and then starts his break,
Add a break vertex between any vertex of which the corresponding trip has an end time that is least 135 minutes earlier than the end time of the shift and the sink vertex. This indicates a driver has his break and after that does not drive any trips anymore for the duration of his shift.

Finally, we also add the extreme situation in which a driver starts the shift, and is not assigned to any trip for the complete duration of the shift and only is assigned to have a break within the given limits. This is accomplished by adding a break vertex between the source and sink vertex, with an edge from the source to the break and one from the break to the sink.

With these break vertices added we can represent any feasible bus plan for a long-shift $b$ with a path through the graph $G_b$, but now not every path through the graph corresponds to a feasible bus plan (i.e. paths that contain no break vertex or more than one break vertex do not correspond to feasible bus plans). We will address this issue while solving the pricing problem and ensure that only paths corresponding to feasible bus plans are considered.

One note to be made regarding the breaks is that with the above approach we do not plan the exact start time and end time of a break but we give a time window in which the break must be held. An example of this time window can be seen in Figure 5.1 where the 45 minute break must take place somewhere in the depicted time window.

We now want to put cost on each of the edges in such a way that a path through the graph does not only correspond to a feasible bus plan, but also that the total cost of the path is equal to the reduced cost of the corresponding bus plan. When looking at the reduced cost of a bus plan, the contribution of trip $t$ if it is followed by trip $t'$ is exactly:
5.3. PROBLEM FORMULATION

$C^B(\text{idle}(t, t')) - \phi_t$, where $\phi_t$ is the dual multiplier corresponding to the trip constraint (5.1) corresponding to trip $t$. In case of a possible break between trip $t$ and $t'$, denoted by the presence of a secondary path from $t$ to $t'$ through a break vertex the cost of the edge from vertex $t$ to the break vertex will be equal to the cost of the edge from vertex $t$ to vertex $t'$. The edge from the break vertex to vertex $t'$ will have cost 0. This cost distribution is depicted in Figure 5.2.

Furthermore, the cost of all outgoing edges from the source vertex will get cost $-\omega_b$, where $\omega_b$ is the dual multiplier corresponding to the constraint (5.2) for shift $b$.

In case of a shift without a break, a path through the graph corresponds to a feasible bus plan and the total cost of the path is equal to the reduced cost of the bus plan. In case of a shift that needs a mandatory break, any path through the graph that contains exactly one break vertex also corresponds to a feasible bus plan and again the total cost of the path is equal to the reduced cost of the bus plan. Finding the bus plan with minimum reduced cost for a given shift $b$ now comes down to finding the shortest source-to-sink path in the graph $G_b$. For the shifts that have a duration of less than 4.5 hours, solving the shortest path problem is straight-forward. Since we have a DAG with topological order this can be done in $O(V + E)$ time (cf. Cormen et al. (2001)). For the shifts that have a duration longer than 4.5 hours, and therefore require a mandatory break, determining the bus plan with minimum reduced cost takes a bit more work. Instead of having a dynamic programming problem with only one state variable per vertex for holding the cost of the minimum cost path to this vertex, we also need a second state variable for holding the cost of the minimum cost path with a break to this vertex. Furthermore, we also need to keep two references to the predecessors: one for the case a break has been taken already and one for the...
case no break has been taken.

Additionally, we must make two minor modifications to the behavior of updating the cost of the successor vertices in the dynamic programming algorithm:

- Do not update the minimum cost value with a break of any successor that is a break vertex because it is not possible to have more than one break vertex in a path.

- When the current vertex is a break vertex, only update the minimum cost value with a break of the successors with the minimum cost value without a break of the current vertex. Recall from Figure 5.2 that the cost of the outgoing edges of a break vertex is set to 0.

The required modifications will not change the complexity of the algorithm because we only add a constant number of additional operations per vertex. So the running time of the algorithm for finding the shortest path with breaks also is $O(V + E)$.

If our shortest path algorithm cannot find a new bus plan with negative reduced costs for none of the bus shifts, the value of the LP-relaxation cannot be improved anymore and this means that the LP-relaxation has been solved to optimality.

### 5.3.2 Solving the ILP

Since the solution to the LP-relaxation might be fractional, we have to ensure that we end up with an integral solution. We achieve this by reinserting the integrality constraints and solve the problem with the set of columns generated during the column generation as an ILP. It turns out that using only the set of columns generated during the column generation is often too restrictive for the ILP, in the sense that this results in quite bad integral solutions, which come at the expense of big running times.

As in the previous chapter we resolve this difficulty by creating a set of extra columns during the column generation process. These extra columns are put in a column pool and are generated in the following way:
5.3. PROBLEM FORMULATION

- For each trip-vertex belonging to the path with minimum cost:
  
  - Take the vertex out of the graph
  - Solve the shortest path problem again
  - Put the bus plan corresponding to the new shortest path in the column pool
  - Reinsert the vertex again in the graph.

Note that when solving the shortest path after one of the vertices has been removed, we do not require the total cost of the shortest path to be negative (i.e. we do not require negative reduced cost for these additional columns).

By generating the columns this way they are not completely random, but quite related to the columns needed for solving the LP-relaxation to optimality.

After the column generation is finished, all unique columns from the column pool are added as candidates to the ILP-formulation of the problem. After these columns have been added, the problem has become considerably larger in number of variables. Fortunately, the set of columns is now less restrictive, enabling the solver to find feasible, good integral solutions easier.

Our preliminary experiments showed that the integrality gap between the solution value of the LP-relaxation and the final integral solution was really small. Similar to solving the final ILP for the gate assignment problem in Chapter 4, we can solve the present ILP in two stages. First we assume a small integrality gap by supplying an upper bound to the solver which will prune all nodes with a relaxation value above it (i.e. it behaves as if an integral solution with the given value has been found already). Due to this tight upper bound, we can also have the solver put more emphasis on finding integral solutions, instead of proving the optimum. Setting the emphasis on integral solutions will result in different branching schemes and node selections being used by the solver.

If this tight upper bound leads to an infeasible ILP problem, we will remove the upper bound and solve the problem again. After preliminary experiments, the upper bound was set such that for all instances the ILP problem was still feasible.
CHAPTER 5. SOLVING THE BUS PLANNING PROBLEM

<table>
<thead>
<tr>
<th>Instance</th>
<th># Flights</th>
<th># trips</th>
<th># supertrips</th>
<th># Shifts</th>
<th># buses</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>198</td>
<td>299</td>
<td>49</td>
<td>14</td>
<td>50</td>
</tr>
<tr>
<td>2</td>
<td>230</td>
<td>346</td>
<td>41</td>
<td>21</td>
<td>63</td>
</tr>
<tr>
<td>3</td>
<td>248</td>
<td>372</td>
<td>41</td>
<td>23</td>
<td>61</td>
</tr>
<tr>
<td>4</td>
<td>252</td>
<td>379</td>
<td>31</td>
<td>24</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>253</td>
<td>380</td>
<td>31</td>
<td>19</td>
<td>64</td>
</tr>
<tr>
<td>6</td>
<td>202</td>
<td>305</td>
<td>17</td>
<td>23</td>
<td>62</td>
</tr>
<tr>
<td>7</td>
<td>227</td>
<td>343</td>
<td>43</td>
<td>20</td>
<td>62</td>
</tr>
<tr>
<td>8</td>
<td>212</td>
<td>292</td>
<td>25</td>
<td>28</td>
<td>113</td>
</tr>
<tr>
<td>9</td>
<td>261</td>
<td>391</td>
<td>37</td>
<td>24</td>
<td>61</td>
</tr>
<tr>
<td>10</td>
<td>249</td>
<td>378</td>
<td>23</td>
<td>24</td>
<td>64</td>
</tr>
<tr>
<td>11</td>
<td>250</td>
<td>378</td>
<td>28</td>
<td>21</td>
<td>64</td>
</tr>
<tr>
<td>12</td>
<td>249</td>
<td>377</td>
<td>35</td>
<td>26</td>
<td>62</td>
</tr>
<tr>
<td>13</td>
<td>254</td>
<td>382</td>
<td>54</td>
<td>21</td>
<td>62</td>
</tr>
<tr>
<td>14</td>
<td>199</td>
<td>303</td>
<td>36</td>
<td>13</td>
<td>50</td>
</tr>
<tr>
<td>15</td>
<td>222</td>
<td>332</td>
<td>32</td>
<td>21</td>
<td>63</td>
</tr>
<tr>
<td>16</td>
<td>251</td>
<td>378</td>
<td>40</td>
<td>22</td>
<td>61</td>
</tr>
<tr>
<td>17</td>
<td>260</td>
<td>390</td>
<td>21</td>
<td>24</td>
<td>64</td>
</tr>
<tr>
<td>18</td>
<td>253</td>
<td>382</td>
<td>29</td>
<td>23</td>
<td>64</td>
</tr>
<tr>
<td>19</td>
<td>263</td>
<td>394</td>
<td>35</td>
<td>23</td>
<td>62</td>
</tr>
<tr>
<td>20</td>
<td>253</td>
<td>382</td>
<td>58</td>
<td>20</td>
<td>62</td>
</tr>
<tr>
<td>21</td>
<td>192</td>
<td>290</td>
<td>50</td>
<td>14</td>
<td>50</td>
</tr>
<tr>
<td>22</td>
<td>214</td>
<td>318</td>
<td>42</td>
<td>21</td>
<td>63</td>
</tr>
<tr>
<td>23</td>
<td>245</td>
<td>369</td>
<td>41</td>
<td>26</td>
<td>61</td>
</tr>
<tr>
<td>24</td>
<td>230</td>
<td>344</td>
<td>16</td>
<td>25</td>
<td>64</td>
</tr>
<tr>
<td>25</td>
<td>250</td>
<td>373</td>
<td>22</td>
<td>21</td>
<td>64</td>
</tr>
<tr>
<td>26</td>
<td>252</td>
<td>380</td>
<td>33</td>
<td>24</td>
<td>62</td>
</tr>
<tr>
<td>27</td>
<td>257</td>
<td>387</td>
<td>43</td>
<td>22</td>
<td>62</td>
</tr>
<tr>
<td>28</td>
<td>192</td>
<td>287</td>
<td>38</td>
<td>15</td>
<td>50</td>
</tr>
<tr>
<td>29</td>
<td>199</td>
<td>302</td>
<td>24</td>
<td>20</td>
<td>63</td>
</tr>
<tr>
<td>30</td>
<td>228</td>
<td>342</td>
<td>27</td>
<td>26</td>
<td>61</td>
</tr>
</tbody>
</table>

Table 5.1: Instance sizes.

5.4 Computational experiments

For testing our solution method for the bus planning problem we have implemented the algorithm in C++. The computer on which we ran the experiments was equipped with an Intel Pentium 4 3.00 GHz processor and 1 GB of RAM. For solving the LPs and ILPs we made use of the Concert Technology interface to CPLEX 9.1.3 (ILOG, 2005).

For testing our algorithm, AAS provided us with all data regarding buses and flights for one complete month. These data concerned the real arrival and departure times of the flights as they were known at the end of the day. Hence, we cannot compare our results fully to the results of the approach currently in use at AAS, which uses the expected data. Another complicating factor is that
the current approach works by planning everything with a rolling time horizon of three hours ahead, while our solution method considers the whole day.

In Table 5.1 the sizes of the instances provided by AAS are given. The column with the number of supertrips gives the number out of the total number of trips which represent supertrips (i.e. trips that need to be covered more than once). In the column with the number of trips the supertrips are counted as one trip. Almost all of the supertrips consist of trips that need to be covered just twice.

**The effect of the improvements separately**

To avoid symmetry in the model, we introduced supertrips. To investigate the effect of using supertrips, we solved the instances not only with supertrips enabled, but also with supertrips disabled (i.e. every supertrip is replaced by $S_t$ identical separate trips). Furthermore, we also investigate the effect of column deletion (CD), which is removing columns from the model every given number of iterations.

In Table 5.2 and Table 5.3 we present the different results regarding solving the LP in case supertrips are disabled and enabled respectively.

As in solving the gate assignment problem in Chapter 4 we also make use of column deletion while solving the bus planning problem. We experimented with different values for the parameters of the column deletion and based on these experiments we used the following values for the parameters: every 30 iterations we determine the average of the reduced cost of the column added in the previous iteration and remove all columns from the model that have reduced cost larger than $-0.75$ times this average.

For supertrips disabled and enabled, we solved the problem both without and with column deletion and, we present for all cases the number of iterations it took for the column generation to find the optimum, as well as the number of columns that were present in the model when the column generation was finished.

We can see that that the effect of using column deletion on the number of iterations needed goes both ways: there are some instances for which the use
of column deletion decreases the total number of iterations needed, but there are also instances where the use of column deletion increases the number of iterations.

The use of column deletion does yield a significant reduction in the number of variables present in the model when the optimum is reached; reductions up to 80% can be seen. This massive reduction results in a shorter time needed for re-solving the restricted master problem after adding the new columns in each iteration of the column generation process.

The fact that the models in each iteration are smaller in size, meaning they take less time to be solved, results in the decrease in running times as seen in Table 5.2 and Table 5.3. Even in the case where more column generation iterations are needed to solve the LP-relaxation, the total time needed for solving the LP-relaxation decreases due to the size of the models in each iteration.

**Default CPLEX versus CPLEX with improvements**

In Table 5.4 the results of solving the problem with both the use of supertrips and the use of column deletion is compared to solving the problem without these two improvements and is created by combining Table 5.2 and Table 5.3. We refer to case with the two improvements as *Enhanced* and the case without them as *Default*.

When we look at Table 5.4 we can see that the combination of using column deletion and using supertrips yields a significant speedup. The minimum improvement encountered is a speedup of a factor 1.49, while the average speedup is 2.39. Furthermore, when looking at the speedup achieved compared to the size of the running time of the original problem, we can see that there is a trend of the speedup increasing with increasing running times of the original problems.

In Table 5.5 we present information regarding the resulting ILP problems. We compare the time needed for solving the ILP both without and with the upper bound and we can see that although guessing a relative tight upper bound on the ILP solution does not always yield a decrease in the time needed for solving the ILP problem, it enables us to solve all the ILP models within two minutes. Without using the tight upper bound approach two of the instances are not
5.4. **COMPUTATIONAL EXPERIMENTS**

<table>
<thead>
<tr>
<th>Instance</th>
<th>Iterations</th>
<th>Columns in model</th>
<th>Time (s) for LP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No CD</td>
<td>With CD</td>
<td>No CD</td>
</tr>
<tr>
<td>1</td>
<td>568</td>
<td>621</td>
<td>6023</td>
</tr>
<tr>
<td>2</td>
<td>459</td>
<td>428</td>
<td>7664</td>
</tr>
<tr>
<td>3</td>
<td>534</td>
<td>559</td>
<td>8975</td>
</tr>
<tr>
<td>4</td>
<td>549</td>
<td>598</td>
<td>9279</td>
</tr>
<tr>
<td>5</td>
<td>610</td>
<td>684</td>
<td>8794</td>
</tr>
<tr>
<td>6</td>
<td>306</td>
<td>301</td>
<td>5431</td>
</tr>
<tr>
<td>7</td>
<td>525</td>
<td>560</td>
<td>7609</td>
</tr>
<tr>
<td>8</td>
<td>375</td>
<td>420</td>
<td>7833</td>
</tr>
<tr>
<td>9</td>
<td>584</td>
<td>570</td>
<td>9431</td>
</tr>
<tr>
<td>10</td>
<td>469</td>
<td>489</td>
<td>8780</td>
</tr>
<tr>
<td>11</td>
<td>594</td>
<td>563</td>
<td>9321</td>
</tr>
<tr>
<td>12</td>
<td>535</td>
<td>486</td>
<td>9008</td>
</tr>
<tr>
<td>13</td>
<td>685</td>
<td>635</td>
<td>9671</td>
</tr>
<tr>
<td>14</td>
<td>663</td>
<td>709</td>
<td>6178</td>
</tr>
<tr>
<td>15</td>
<td>345</td>
<td>373</td>
<td>6041</td>
</tr>
<tr>
<td>16</td>
<td>542</td>
<td>538</td>
<td>8391</td>
</tr>
<tr>
<td>17</td>
<td>630</td>
<td>589</td>
<td>10243</td>
</tr>
<tr>
<td>18</td>
<td>475</td>
<td>511</td>
<td>9020</td>
</tr>
<tr>
<td>19</td>
<td>529</td>
<td>640</td>
<td>9157</td>
</tr>
<tr>
<td>20</td>
<td>586</td>
<td>585</td>
<td>8842</td>
</tr>
<tr>
<td>21</td>
<td>568</td>
<td>602</td>
<td>5778</td>
</tr>
<tr>
<td>22</td>
<td>421</td>
<td>425</td>
<td>7296</td>
</tr>
<tr>
<td>23</td>
<td>509</td>
<td>518</td>
<td>9701</td>
</tr>
<tr>
<td>24</td>
<td>449</td>
<td>478</td>
<td>8157</td>
</tr>
<tr>
<td>25</td>
<td>578</td>
<td>536</td>
<td>9272</td>
</tr>
<tr>
<td>26</td>
<td>527</td>
<td>544</td>
<td>9375</td>
</tr>
<tr>
<td>27</td>
<td>670</td>
<td>738</td>
<td>9637</td>
</tr>
<tr>
<td>28</td>
<td>586</td>
<td>606</td>
<td>6246</td>
</tr>
<tr>
<td>29</td>
<td>435</td>
<td>456</td>
<td>6462</td>
</tr>
<tr>
<td>30</td>
<td>427</td>
<td>418</td>
<td>7763</td>
</tr>
</tbody>
</table>

**Table 5.2:** Overview results LP without supertrips, both without and with column deletion.

Software solvable within a set time-limit of one hour.

Looking at the instances that are solved by both the default and the enhanced settings, we can see that the time needed by the default settings often is less than the time needed when the enhanced settings are used. This can be explained by the fact that with the enhanced settings a considerable amount of time is spent by CPLEX on pre-processing the root-node of the branch-and-bound tree. Due to the aggressive settings supplied to CPLEX for cut-generation, solving the LP-relaxation of the root-node and adding the cuts takes quite some time.

After the rootnode has been solved, the upper bound limit is obtained by multiplying the value of the solved rootnode with the required maximum integrality.
CHAPTER 5. SOLVING THE BUS PLANNING PROBLEM

<table>
<thead>
<tr>
<th>Instance</th>
<th>Iterations No CD</th>
<th>Iterations With CD</th>
<th>Columns in model No CD</th>
<th>Columns in model With CD</th>
<th>Time (s) for LP No CD</th>
<th>Time (s) for LP With CD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>523</td>
<td>586</td>
<td>5325</td>
<td>1050</td>
<td>57.6</td>
<td>49.2</td>
</tr>
<tr>
<td>2</td>
<td>377</td>
<td>404</td>
<td>6334</td>
<td>1521</td>
<td>90.9</td>
<td>69.8</td>
</tr>
<tr>
<td>3</td>
<td>501</td>
<td>481</td>
<td>8187</td>
<td>1648</td>
<td>175.2</td>
<td>99.0</td>
</tr>
<tr>
<td>4</td>
<td>521</td>
<td>518</td>
<td>8779</td>
<td>1959</td>
<td>171.0</td>
<td>111.5</td>
</tr>
<tr>
<td>5</td>
<td>533</td>
<td>545</td>
<td>7752</td>
<td>1787</td>
<td>152.0</td>
<td>105.4</td>
</tr>
<tr>
<td>6</td>
<td>271</td>
<td>329</td>
<td>4927</td>
<td>1136</td>
<td>37.3</td>
<td>32.5</td>
</tr>
<tr>
<td>7</td>
<td>482</td>
<td>490</td>
<td>6834</td>
<td>1438</td>
<td>107.1</td>
<td>70.1</td>
</tr>
<tr>
<td>8</td>
<td>360</td>
<td>371</td>
<td>7317</td>
<td>2157</td>
<td>49.9</td>
<td>45.8</td>
</tr>
<tr>
<td>10</td>
<td>527</td>
<td>523</td>
<td>8352</td>
<td>1778</td>
<td>208.9</td>
<td>119.4</td>
</tr>
<tr>
<td>11</td>
<td>411</td>
<td>483</td>
<td>7794</td>
<td>1807</td>
<td>148.7</td>
<td>114.9</td>
</tr>
<tr>
<td>12</td>
<td>528</td>
<td>530</td>
<td>8353</td>
<td>1788</td>
<td>182.3</td>
<td>111.3</td>
</tr>
<tr>
<td>13</td>
<td>516</td>
<td>463</td>
<td>8127</td>
<td>1678</td>
<td>300.2</td>
<td>109.4</td>
</tr>
<tr>
<td>14</td>
<td>588</td>
<td>577</td>
<td>8163</td>
<td>1533</td>
<td>302.7</td>
<td>116.2</td>
</tr>
<tr>
<td>15</td>
<td>639</td>
<td>655</td>
<td>5750</td>
<td>1121</td>
<td>68.6</td>
<td>48.6</td>
</tr>
<tr>
<td>16</td>
<td>336</td>
<td>377</td>
<td>5600</td>
<td>1273</td>
<td>69.6</td>
<td>58.6</td>
</tr>
<tr>
<td>17</td>
<td>465</td>
<td>516</td>
<td>7458</td>
<td>1701</td>
<td>145.6</td>
<td>107.4</td>
</tr>
<tr>
<td>18</td>
<td>604</td>
<td>564</td>
<td>9771</td>
<td>1700</td>
<td>266.0</td>
<td>140.3</td>
</tr>
<tr>
<td>19</td>
<td>445</td>
<td>462</td>
<td>8135</td>
<td>1873</td>
<td>170.8</td>
<td>116.1</td>
</tr>
<tr>
<td>20</td>
<td>508</td>
<td>509</td>
<td>8363</td>
<td>1670</td>
<td>211.0</td>
<td>127.2</td>
</tr>
<tr>
<td>21</td>
<td>500</td>
<td>539</td>
<td>7624</td>
<td>1487</td>
<td>171.7</td>
<td>101.1</td>
</tr>
<tr>
<td>22</td>
<td>483</td>
<td>514</td>
<td>4843</td>
<td>1100</td>
<td>43.1</td>
<td>33.9</td>
</tr>
<tr>
<td>23</td>
<td>364</td>
<td>367</td>
<td>6149</td>
<td>1251</td>
<td>67.1</td>
<td>50.4</td>
</tr>
<tr>
<td>24</td>
<td>476</td>
<td>470</td>
<td>8923</td>
<td>1577</td>
<td>228.0</td>
<td>103.2</td>
</tr>
<tr>
<td>25</td>
<td>446</td>
<td>461</td>
<td>7937</td>
<td>1615</td>
<td>125.8</td>
<td>71.3</td>
</tr>
<tr>
<td>26</td>
<td>529</td>
<td>532</td>
<td>8620</td>
<td>1851</td>
<td>257.3</td>
<td>109.3</td>
</tr>
<tr>
<td>27</td>
<td>524</td>
<td>493</td>
<td>8887</td>
<td>1584</td>
<td>237.1</td>
<td>121.4</td>
</tr>
<tr>
<td>28</td>
<td>581</td>
<td>599</td>
<td>8317</td>
<td>1573</td>
<td>283.3</td>
<td>123.2</td>
</tr>
<tr>
<td>29</td>
<td>512</td>
<td>554</td>
<td>5453</td>
<td>1295</td>
<td>53.0</td>
<td>41.3</td>
</tr>
<tr>
<td>30</td>
<td>419</td>
<td>426</td>
<td>6115</td>
<td>1375</td>
<td>61.2</td>
<td>42.4</td>
</tr>
<tr>
<td>31</td>
<td>409</td>
<td>372</td>
<td>7337</td>
<td>1568</td>
<td>98.2</td>
<td>62.9</td>
</tr>
</tbody>
</table>

Table 5.3: Overview results LP with supertrips, both with and without column deletion.

gap and the settings for cut-generation are set to default again.

Furthermore, we set the emphasis for the solver to finding integral solutions because all integral solutions will fall within the given tight integrality gap.

In Table 5.6 we present all information regarding the best settings for solving the problem. We can see from the table that the time needed for solving the pricing problems is about 50% of the total time needed for solving the LP-relaxation.

In the time needed for solving the pricing problems also the time needed for creating the additional bus plans for the column pool is incorporated. The creation of these additional bus plans accounts for roughly 70% of the time
### 5.4. COMPUTATIONAL EXPERIMENTS

| Instance | Iterations | | Columns in model | | Time (s) for LP | |
|----------|------------|----------------|-------------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| 1        | 568        | 586           | 6023             | 1050            | 103.6           | 49.2            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 2        | 459        | 404           | 7664             | 1521            | 172.4           | 69.8            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 3        | 534        | 481           | 8975             | 1648            | 241.7           | 99.0            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 4        | 549        | 518           | 9279             | 1959            | 246.2           | 111.5           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 5        | 610        | 545           | 8794             | 1787            | 236.2           | 105.4           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 6        | 306        | 329           | 5431             | 1136            | 51.6            | 32.5            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 7        | 525        | 490           | 7609             | 1438            | 168.6           | 70.1            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 8        | 375        | 371           | 7833             | 2157            | 68.0            | 45.8            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 10       | 584        | 523           | 9431             | 1778            | 292.5           | 119.4           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 11       | 469        | 483           | 8780             | 1807            | 214.4           | 114.9           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 12       | 594        | 530           | 9321             | 1788            | 311.0           | 111.3           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 13       | 535        | 463           | 9008             | 1678            | 319.5           | 109.4           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 14       | 685        | 577           | 9671             | 1533            | 439.4           | 116.2           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 15       | 663        | 655           | 6178             | 1121            | 97.6            | 48.6            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 16       | 345        | 377           | 6041             | 1273            | 100.1           | 58.6            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 17       | 542        | 516           | 8391             | 1701            | 212.2           | 107.4           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 18       | 630        | 564           | 10243            | 1700            | 305.0           | 140.3           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 19       | 475        | 462           | 9020             | 1873            | 246.1           | 116.1           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 20       | 529        | 509           | 9157             | 1670            | 304.1           | 127.2           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 21       | 586        | 539           | 8842             | 1487            | 301.8           | 101.1           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 22       | 568        | 514           | 5778             | 1100            | 79.1            | 33.9            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 23       | 421        | 367           | 7296             | 1251            | 122.8           | 50.4            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 24       | 509        | 470           | 9701             | 1577            | 318.2           | 103.2           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 25       | 449        | 461           | 8157             | 1615            | 145.0           | 71.3            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 26       | 578        | 532           | 9272             | 1851            | 267.2           | 109.3           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 27       | 527        | 493           | 9375             | 1584            | 281.6           | 121.4           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 28       | 670        | 599           | 9637             | 1573            | 575.9           | 123.2           |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 29       | 586        | 554           | 6246             | 1295            | 88.2            | 41.3            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 30       | 435        | 426           | 6462             | 1375            | 81.0            | 42.4            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |
| 31       | 427        | 372           | 7763             | 1568            | 130.7           | 62.9            |                  |                  |                 |                 |                 |                 |                 |                 |                 |                 |

**Table 5.4:** Overview results LP of default approach and enhanced approach (i.e. with supertrips and column deletion).

needed for solving the pricing problems. Due to its structure the creation of these additional columns could be separated over multiple computers or threads because their creation only depends on the graph and a set of dual multipliers.

Since in most cases the time needed for solving the LP-relaxation is larger than the time needed for solving the resulting ILP, a considerable improvement of the total running times can be achieved by parallelizing the solving of the pricing problems and also the creation of the additional columns.

To investigate the effect on the time needed for solving both the LP-relaxation and the ILP, we plot the respective running times against the number of trips.
### Table 5.5: The ILP information for both the default settings and enhanced settings with a tight upper bound.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Number of rows</th>
<th>Number of columns</th>
<th>Percentage Non-Zero</th>
<th>Time (s) Default</th>
<th>Time (s) Enhanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>313</td>
<td>40615</td>
<td>3.38</td>
<td>51.9</td>
<td>31.1</td>
</tr>
<tr>
<td>2</td>
<td>367</td>
<td>49384</td>
<td>2.98</td>
<td>5.6</td>
<td>13.0</td>
</tr>
<tr>
<td>3</td>
<td>395</td>
<td>58404</td>
<td>2.68</td>
<td>70.3</td>
<td>32.3</td>
</tr>
<tr>
<td>4</td>
<td>403</td>
<td>67855</td>
<td>2.65</td>
<td>5.4</td>
<td>17.9</td>
</tr>
<tr>
<td>5</td>
<td>399</td>
<td>58095</td>
<td>2.77</td>
<td>6.5</td>
<td>10.5</td>
</tr>
<tr>
<td>6</td>
<td>328</td>
<td>36047</td>
<td>3.1</td>
<td>1.2</td>
<td>3.9</td>
</tr>
<tr>
<td>7</td>
<td>363</td>
<td>51173</td>
<td>3.03</td>
<td>3.5</td>
<td>8.4</td>
</tr>
<tr>
<td>8</td>
<td>320</td>
<td>46378</td>
<td>2.69</td>
<td>1.8</td>
<td>4.6</td>
</tr>
<tr>
<td>10</td>
<td>415</td>
<td>62555</td>
<td>2.63</td>
<td>8.6</td>
<td>22.9</td>
</tr>
<tr>
<td>11</td>
<td>402</td>
<td>65530</td>
<td>2.69</td>
<td>*</td>
<td>54.7</td>
</tr>
<tr>
<td>12</td>
<td>399</td>
<td>64603</td>
<td>2.76</td>
<td>4.6</td>
<td>11.5</td>
</tr>
<tr>
<td>13</td>
<td>403</td>
<td>61269</td>
<td>2.73</td>
<td>5.1</td>
<td>9.2</td>
</tr>
<tr>
<td>14</td>
<td>403</td>
<td>61517</td>
<td>2.76</td>
<td>4.3</td>
<td>9.0</td>
</tr>
<tr>
<td>15</td>
<td>316</td>
<td>41331</td>
<td>3.19</td>
<td>3.4</td>
<td>9.4</td>
</tr>
<tr>
<td>16</td>
<td>353</td>
<td>47087</td>
<td>3</td>
<td>*</td>
<td>101.7</td>
</tr>
<tr>
<td>17</td>
<td>400</td>
<td>57743</td>
<td>2.7</td>
<td>39.5</td>
<td>35.5</td>
</tr>
<tr>
<td>18</td>
<td>414</td>
<td>71978</td>
<td>2.64</td>
<td>12.0</td>
<td>27.5</td>
</tr>
<tr>
<td>19</td>
<td>405</td>
<td>68003</td>
<td>2.77</td>
<td>7.7</td>
<td>18.7</td>
</tr>
<tr>
<td>20</td>
<td>417</td>
<td>65201</td>
<td>2.69</td>
<td>4.7</td>
<td>11.1</td>
</tr>
<tr>
<td>21</td>
<td>402</td>
<td>58176</td>
<td>2.79</td>
<td>2.6</td>
<td>6.2</td>
</tr>
<tr>
<td>22</td>
<td>304</td>
<td>33664</td>
<td>3.21</td>
<td>4.0</td>
<td>15.4</td>
</tr>
<tr>
<td>23</td>
<td>339</td>
<td>44892</td>
<td>3.06</td>
<td>3.1</td>
<td>5.2</td>
</tr>
<tr>
<td>24</td>
<td>395</td>
<td>65459</td>
<td>2.67</td>
<td>6.3</td>
<td>9.7</td>
</tr>
<tr>
<td>25</td>
<td>369</td>
<td>54702</td>
<td>2.73</td>
<td>2.2</td>
<td>5.9</td>
</tr>
<tr>
<td>26</td>
<td>394</td>
<td>65128</td>
<td>2.75</td>
<td>6.5</td>
<td>13.1</td>
</tr>
<tr>
<td>27</td>
<td>404</td>
<td>67885</td>
<td>2.71</td>
<td>5.0</td>
<td>9.7</td>
</tr>
<tr>
<td>28</td>
<td>409</td>
<td>62702</td>
<td>2.71</td>
<td>6.2</td>
<td>14.7</td>
</tr>
<tr>
<td>29</td>
<td>302</td>
<td>38544</td>
<td>3.25</td>
<td>2.3</td>
<td>7.0</td>
</tr>
<tr>
<td>30</td>
<td>322</td>
<td>42764</td>
<td>3</td>
<td>2.9</td>
<td>6.8</td>
</tr>
<tr>
<td>31</td>
<td>368</td>
<td>49768</td>
<td>2.69</td>
<td>2.7</td>
<td>6.5</td>
</tr>
</tbody>
</table>

*: Not Solvable within one hour
### 5.4. COMPUTATIONAL EXPERIMENTS

<table>
<thead>
<tr>
<th>Instance</th>
<th>Number of iterations</th>
<th>Size of pool</th>
<th>Columns taken out</th>
<th>Time (s) needed for solving</th>
<th>Pricing</th>
<th>LP</th>
<th>ILP</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>586</td>
<td>50189</td>
<td>4800</td>
<td>25.9</td>
<td>49.2</td>
<td>31.1</td>
<td>87.2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>404</td>
<td>59287</td>
<td>5205</td>
<td>37.8</td>
<td>69.8</td>
<td>13.0</td>
<td>93.3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>481</td>
<td>68237</td>
<td>6631</td>
<td>44.7</td>
<td>99.0</td>
<td>32.3</td>
<td>146.2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>518</td>
<td>79710</td>
<td>7465</td>
<td>58.6</td>
<td>111.5</td>
<td>17.9</td>
<td>146.3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>545</td>
<td>71700</td>
<td>6376</td>
<td>56.3</td>
<td>105.4</td>
<td>10.5</td>
<td>131.1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>329</td>
<td>41757</td>
<td>4233</td>
<td>17.1</td>
<td>32.5</td>
<td>3.9</td>
<td>43.7</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>490</td>
<td>62626</td>
<td>5734</td>
<td>37.7</td>
<td>70.1</td>
<td>8.4</td>
<td>91.0</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>371</td>
<td>45355</td>
<td>5746</td>
<td>26.5</td>
<td>45.8</td>
<td>4.6</td>
<td>59.0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>523</td>
<td>75863</td>
<td>6963</td>
<td>58.2</td>
<td>119.4</td>
<td>22.9</td>
<td>158.9</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>483</td>
<td>75460</td>
<td>7190</td>
<td>58.3</td>
<td>114.9</td>
<td>54.7</td>
<td>186.0</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>530</td>
<td>77342</td>
<td>6996</td>
<td>60.3</td>
<td>111.3</td>
<td>11.5</td>
<td>138.8</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>463</td>
<td>71605</td>
<td>6586</td>
<td>53.9</td>
<td>109.4</td>
<td>9.2</td>
<td>135.1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>577</td>
<td>75736</td>
<td>6897</td>
<td>57.9</td>
<td>116.2</td>
<td>9.0</td>
<td>141.2</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>655</td>
<td>51075</td>
<td>5161</td>
<td>25.1</td>
<td>48.6</td>
<td>9.4</td>
<td>65.0</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>377</td>
<td>53986</td>
<td>5220</td>
<td>30.7</td>
<td>58.6</td>
<td>101.7</td>
<td>169.8</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>516</td>
<td>70275</td>
<td>6529</td>
<td>53.7</td>
<td>107.4</td>
<td>35.5</td>
<td>158.2</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>564</td>
<td>85648</td>
<td>8189</td>
<td>71.0</td>
<td>140.3</td>
<td>27.5</td>
<td>185.9</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>462</td>
<td>79072</td>
<td>6877</td>
<td>59.6</td>
<td>116.1</td>
<td>18.7</td>
<td>151.8</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>509</td>
<td>78166</td>
<td>7110</td>
<td>67.0</td>
<td>127.2</td>
<td>11.1</td>
<td>156.6</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>539</td>
<td>72740</td>
<td>6441</td>
<td>54.6</td>
<td>101.1</td>
<td>6.2</td>
<td>123.1</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>514</td>
<td>42103</td>
<td>4168</td>
<td>18.4</td>
<td>33.9</td>
<td>15.4</td>
<td>55.4</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>367</td>
<td>52971</td>
<td>5177</td>
<td>26.5</td>
<td>50.4</td>
<td>5.2</td>
<td>64.5</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>470</td>
<td>78872</td>
<td>7729</td>
<td>47.3</td>
<td>103.2</td>
<td>9.7</td>
<td>129.2</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>461</td>
<td>64228</td>
<td>6808</td>
<td>37.8</td>
<td>71.3</td>
<td>5.9</td>
<td>90.3</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>532</td>
<td>76408</td>
<td>6978</td>
<td>56.0</td>
<td>109.3</td>
<td>13.1</td>
<td>138.6</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>493</td>
<td>78797</td>
<td>7530</td>
<td>61.2</td>
<td>121.4</td>
<td>9.7</td>
<td>148.0</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>599</td>
<td>76563</td>
<td>6946</td>
<td>60.4</td>
<td>123.2</td>
<td>14.7</td>
<td>154.5</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>554</td>
<td>47765</td>
<td>4695</td>
<td>20.3</td>
<td>41.3</td>
<td>7.0</td>
<td>54.5</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>426</td>
<td>47560</td>
<td>4954</td>
<td>21.4</td>
<td>42.4</td>
<td>6.8</td>
<td>57.2</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>372</td>
<td>56164</td>
<td>5847</td>
<td>29.4</td>
<td>62.9</td>
<td>6.5</td>
<td>81.6</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.6: All information with regards to the optimal approach.

Solving the ILP is influenced in a positive way when the symmetry is removed with using the supertrips.

The effect of using supertrips simplifies the problem in two ways: first of all the symmetry is removed, secondly the number of trips is decreased. While solving the LP-relaxation benefits more from the decrease in the number of trips, solving the resulting ILP benefits more from the fact that the symmetry is removed.
5.5 Conclusion

We have presented an ILP representation for the bus planning problem at AAS and developed a new method for solving the problem. We have also written an implementation to test our algorithm and we were able to solve the schedule for any given day in a matter of minutes. Contrary to the current implementation at AAS, which makes use of a rolling horizon of 3 hours, we are able to plan the complete day, which enables us to have a much better overview while planning for example the mandatory breaks. Furthermore, considering the full day enables us to investigate the combination of the bus planning problem with the gate planning problem, which we will consider in Chapter 6.

In this chapter we looked at creating a robust schedule for a complete day of the bus planning problem. We assumed the arrival time and departure time of all flights to be static. One interesting idea that arose is to investigate the possi-
Figure 5.4: Time needed to solve the ILP in relation to number of trips.
abilities of also using this approach for the operational planning when changes in the arrival times and departure times occur. In the case of operational planning the schedule could be solved again every time a conflict introducing change (i.e. a situation in which one bus would have to drive two trips at the same time) in the data is presented by the Central Information System Schiphol (CISS).

In operational planning there is less time available for solving the bus planning problem than in day-ahead planning. The running times are a bit too large for our approach to be used as is for the operational planning. Hence, it does not allow for making a complete planning from scratch every time a conflict introducing change is presented. We may reduce the total time needed for solving the problem by making use of parallel programming, but this alone will not be sufficient.

In the case of a flight having a delay, we do not want to have to solve the problem from scratch again; we would like to reuse as much information from previous solutions as possible. One possibility is to keep all the bus plans in memory and when a flight gets a delay, update the trips corresponding to this flight in all bus plans to reflect the delay. After this we can remove all bus plans that became invalid due to trips overlapping with breaks or the end of a shift. In case a bus plan became invalid because of two trips overlapping after the delay of one of them, we could also create two new bus plans from this one invalid bus plan. Each of the two new bus plans would contain only one of the two trips that caused the original bus plan to become invalid. This way, the whole process of re-solving the problem would have a hot-start instead of starting all the way from scratch. The idea is that this would allow the problem to be solved in a much shorter time than when solved from scratch.

Currently we are investigating this possibility by comparing the suggested approach with a simple first-come-first-serve heuristic (which forms also the basis of the program currently in use at AAS for solving the bus planning problem) in a simulation study.