An empirical study of planning and scheduling interactions in the road passenger transportation domain*

Javier Murillo and Beatriz Lopez
University of Girona, Campus Montilivi, edifici P4, 17071 Girona
{jmurillo,blopez}@eia.udg.es
http://aeds.udg.es

Abstract

In this paper we investigate the interactions between planning and scheduling in a specific scenario, road passenger transportation. The problem consists of finding the best assignment of available drivers (resources) to a set of requested services (actions) given a cost function and subject to the constraints provided by administrations. We have considered two approaches. In the first approach we deal with planning and scheduling as a single solution step. In the second approach, we consider planning and scheduling in separate steps. The results obtained from the experimentation show, in this particular case, that keeping planning and scheduling separate, achieves a better performance.

Introduction

On the basis that practical problems have been difficult to solve using either planning or scheduling techniques in an isolated way (Smith, Frank, & Jonsson 2000), there is a current trend to integrate both paradigms (Barták & Rudová 2001).

Planning deals with actions, and scheduling with resources; however, the selection of one action involves the consumption of some resources. As illustrated in (Gallab, Nau, & Traverso 2004), when you move a robot from position l to position l', you are changing the state of the robot in both an absolute and a relative way. The absolute change is induced by the new position of the robot after performing the movement (what planning analyzes). The relative change consists of slightingly decreasing the energy level of the robot (what scheduling deals with). How much the energy decreases depends on the roughness of the terrain, and the soil characteristics, among other factors. So, the selection of the next position of the robot (planning) impacts on the resource management (scheduling) and vice versa: improving a constraint on a scheduling problem impacts on the activity selection (planning) (Bresina *et al.* 2002).

One of the recent solutions proposed to overcome planning and scheduling interactions is to define an integrated framework. For example, in (Castillo *et al.* 2006) a hierarchical task network (HTN) is proposed, while in (Haslum & Geffner 2001) propose a heuristic search procedure to deal with planning and time and resource constraints. A different approach is (Ai-Chang *et al.*), in which a set of plans are stored in a temporal network in order to deal with constraints.

However, it is not necessarily true that an integrated framework is the only solution when dealing with a new problem in which both actions and time and resources are involved. An analysis of the benefits and limitations of keeping planning and scheduling separate is required.

This has been our case. We are dealing with the road passenger transportation problem, in which we need to optimize the driver's driving time, while assigning sequences of actions to them (drivers' duties). Two different approaches are possible: planning and scheduling in separated steps, and a single step for the entire process. In this paper we experimentally analyze both approaches. From the study we conclude that in our particular problem, even though some interactions between planning and scheduling exist, better results are obtained if we isolate the two steps.

This paper is organized as follows. First, we provide a description of the problem. Then we present both alternatives, namely, the service approach (integration) and the journey approach (isolation). We continue by providing the results of our experimentation. And we end the paper with some conclusions and a discussion.

Problem description

In the road passenger transportation problem we are presented with a set of resources, drivers $D = \{d_1, \ldots, d_n\}$, and a set of tasks, (services) $S = \{s_1, \ldots, s_m\}$, to be performed using the resources. The problem consists of finding the *best* assignment of drivers to services given a cost function and subject to the constraints and preferences provided by administrations (local, national or European). We are dealing, then, with a *constraint optimization problem* in which we are trying to minimize the driver's costs, both in time and distance. The solution of the problem should be a complete allocation of the drivers activities that covers the complete set of services. This includes which sequence of actions each driver should perform (planning outcomes), and when (schedule outcomes). Thus, the components of the problem are: drivers, services, constraints and the cost function.

*This research project has been partially funded by the Spanish MEC project TIN2004-06354-C02-02 and DURSI AGAUR SGR 00296.

First, each **driver** is characterized by a basic cost imposed by his contract, a cost per kilometer, a cost per time unit, a starting point and a final point (often the same), and the number of hours that he/she has driven during the last two weeks.

Second, each **service** is characterized by the start location and the final location (where the service is), the start time and the final time (when). From this information, it is possible to compute the spatial distance between both locations, which in turn can also be measured with the time units required by a driver to cover it. There are two kinds of services to be considered: requested and intervening. Requested services are the ones that customers have applied for, while intervening services are those that require moving the driver from the end location of a service to the start location of the next service assigned to him. Analogously, if the first service that the driver performs is not in to his/her current initial place, the corresponding amount of time needed to drive from the initial place to the first service should also be computed in the scheduling. This is similar to the ending location, if the final service is different than the driver's ending place. Requested services are the data of the problem, while intervening services are generated while working towards the solution.

Third, there are several **constraints** regarding driving time which are quite complex in our transport domain. Our problem is related to coach transportation and timetables are as strict as with other forms of transportation (train, buses); what is important is accomplishing the driving time regulated by law. An example of such a constraint is the following: if passengers are children (like on a school trip), then the maximum continuous driving time is 2 hours; in other cases up to 4 hours are allowed. For the sake of simplicity, we consider only four simple constraints in a first approach to the problem (see (López 2005) for a complete list)[1]. The description of the constraints is as follows:

Overlapping. A driver cannot be assigned to two different services with overlapping times. In addition, a driver assigned to a service that ends at time t and location l cannot be assigned to another service that starts at time $t+1$, unless the location of the new service is the same (l). That is, the overlapping constraint also takes care of the transportation of the driver from the end location of a service to the start location of the next service to be performed.

Maximum driving time. By effective driving time we mean that in addition to the driving time dedicated to the services requested, the driving time required for the intervening services should also be computed.

Maximum journey length. Between one service and the next one, drivers can enjoy free time in which no driving activities are being performed. Then, the total of driving time plus free time cannot be over the maximum journey length allowed.

Maximum driving time per two-weeks. There are some regulations regarding driver resting time. One of them describes that the maximum driving time per two weeks cannot be over 90 hours.

Finally, a cost function is required to measure the quality of the solutions, that is, the cost of the allocation of services to drivers. The goal is to find the allocation with the lowest cost, subject to the constraints.

In fact, we can find several works in the literature regarding crew scheduling problems, which are known to be particular cases of the set covering problem (Hoffman & Padberg 2000). Crew scheduling, however, has often been studied in regular services (Loureno, Paixao, & Portugal 2001; Pezzella & Faggioli 1997; Cantillon & Pesendorfer 2006). Indeed, it is important to distinguish between scheduling and two other related problems: vehicle scheduling and the rostering problem (dealing with the rotating shifts of the crews). But also in these latter scenarios, only regular services with strict timetables are studied (see, for example, (Esclapes 2000)). Regular services focus on strict arrival times, while our problem concerns compliance with laws regarding the driving interval times.

The alternatives

In order to investigate the interactions between planning and scheduling, two different approaches have been considered: the service approach and the journey approach (see Figure 1). The service approach maintains planning and scheduling in a single step. That is, given a set of requested services, the driver's schedules are generated, and in them intervening services are also considered. The journey approach splits the solution process into two steps. The first step consists of a planning process, in which a set of possible plans are generated. Each plan consists of a set of compatible journeys. Each journey is the sequence of services (both requested and intervening) that can be assigned to a driver. The outcome of the planning step consists of a set of plans sorted according to their cost. The second step of the journey approach is a scheduling process in which the journeys of each plan are assigned to drivers. This step is repeated for each plan. Both approaches have been modeled following a branch and bound[2] approach and, in each case, particular heuristic and estimating cost functions have been defined.

The service approach

In the service approach, we assign drivers to the services following the branch and bound method: when a solution is found, the branch and bound method computes its cost which is compared with the best solution found so far (upper bound). If the cost of the partial solution is higher, the algorithm backtracks, pruning the subtree below it (Dechter 2003). In addition, if the cost of the partial solution is not higher, but it is possible to estimate its final cost, and the estimation goes over the upper bound, it is also pruned.

[1] We are simplifying our problem in order to illustrate planning and scheduling, but our ultimate goal is to deal with the complexity of the full problem.

[2] We have experimentally tested other methods, but branch and bound allow the manipulation of complex constraints more easily. In addition, branch and bound guarantees that the optimal solution is found.

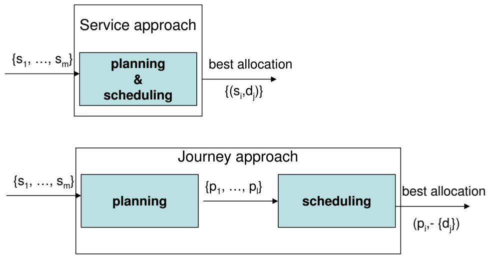

Figure 1: Studied alternatives.

In the following, the formalization of the problem according to the service approach, its modeling and an example are given.

Problem formalization.

Definition 1. A *service* is a tuple

$$s_i =< sl_i, fl_i, st_i, ft_i >$$

where sl_i is the start location, fl_i the final location, st_i the initial time, and ft_i the final time ($st_i < ft_i$).

Definition 2. A *driver* is a tuple

$$d_i =< bc_i, kmc_i, sl_i, fl_i, hw_i, hc_i >$$

where bc_i is the basic cost, kmc_i is the cost per kilometer, sl_i is the start location, fl_i is the final location (often $sl_i = fl_i$), hw_i are the hours accumulated over two weeks, and hc_i is the cost per time unit.

Regarding time, we consider half an hour to be the time unit, and we assume that the distance covered in that time is 45 kilometers. This assumption is used to compute the duration of intervening services.

Definition 3. Given two services, s_i and s_j, with $ft_i < stj$, an *intervening service* between s_i and s_j is defined as a tuple

$$s_{i-j} =< sl_{i-j}, fl_{i-j}, st_{i-j}, ft_{i-j} >$$

where sl_{i-j} is the start location (with $sl_{i-j} = fl_i$), fl_{i-j} the final location (with $fl_{i-j} = sl_j$), st_{i-j} the initial time, and ft_{i-j} the final time, with $st_{i-j} > ft_i$ and $ft_{i-j} < st_j$.

Given a set of services S, and a set of drivers D, a total number of intervening services k could be required. Let I be the set of such intervening services. Then,

Definition 4. An *allocation based on services* is a tuple

$$A_i =< (s_1, d_{i_1}), (s_2, d_{i_2}), ...(s_l, d_{i_l}) >$$

where $s_i \in S \cup I$, $d_j \in D$, and in which all constraints are satisfied. Furthermore, $\bigcup_{s_i \in (A_i \setminus I)} = S$, that is, all requested services are covered, and $\bigcap_{s_i \in (A_i \setminus I)} = \emptyset$, that is, no service is repeated.

Constraints have been described above and are the key issues in the different modeling obtained from the techniques.

The *cost function* that measures the individual cost of a driver i in an allocation A_k is the following:

$$cost(A_k, d_i) = bc_i + \frac{(dist(A_k, d_i) * kmc_i)}{\alpha} + (h(A_k, d_i) * hc_i)\beta \quad (1)$$

where $dist(A_k, d_i)$ is the distance covered by the driver in the A_k allocation measured in kilometers, $h(A_k, d_i)$ is the journey of the driver in the A_k allocation (including non-occupied time) and α and β are parameters of the cost function. This cost function tries to make kilometers and hours, which have different scales, comparable (kilometers are usually defined in [0,100] while hours in [0,24]). After several tries, we have set $\alpha = 10.0$ and $\beta = 7.0$. Finally, note that we are assuming a constant speed of 45km/time unit (1 time unit = 0.5 hours).

The *cost function* is the function that measures the cost of an allocation A_k and is defined as the addition of the individual drivers $cost(A_k, d_i)$, that is,

$$C(A_k) = \sum_{i \in A_k} cost(A_k, d_i). \quad (2)$$

The road passenger transportation problem consists of finding the allocation that minimizes the cost:

$$argmin_{\forall i}(C(A_i)) \quad (3)$$

subject to the above constraints.

Problem modeling. According to a CSP formulation, the model of the problem is based on variables, domain constraints and a cost function. The variables of the problem are the services, the domains are drivers. The main difficulty of using the branch and bound method was to define the appropriate function to estimate the cost of a partial solution in order to prune the search space and provide an answer in a reasonable time. This estimation function should take into account the remaining assignments to be performed, which depends on both the requested services and the intervening services. This function can underestimate the real cost, but never overestimate it, in order to assure that we are not pruning optimal solutions. The estimated cost function F^e has been defined as the sum of the individual estimation cost f^e of the remaining services, R; that is,

$$F^e(R) = \sum_{s_i \in R} f^e(s_i) \quad (4)$$

We can easily define an estimation function $f^e(s_i)$ by computing the cost of the s_i services according to the minimum driver cost. But a more accurate estimator should include intervening services. If we analyze the position of the drivers at the time when service s_i should be deployed,we can estimate the intervening services required both for going to the start location of the service and for returning from the end location. Therefore, we can calculate the estimation of a service with the following expression:

$$f^e(s) = dur(s) * cost_{min_h} + dist(s) * cost_{min_{km}} \quad (5)$$

where $dur(s)$ is the time required to perform the service, including the time required to go to and return from the initial and end locations respectively; $dist(s)$ is the distance in kilometers required to deploy the service, including go and return distances; $cost_{min_h}$ is the minimum cost per hour, and $cost_{min_{km}}$ is the minimum cost per kilometer.

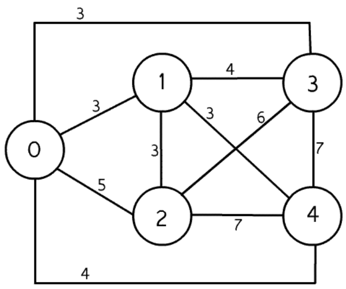

Figure 2: Localities of the case example.

$dur(s)$ is computed as follows:

$$dur(s) = t_{go} + t_{serv} + t_{ret} \quad (6)$$

where t_{serv} is the duration of the service, t_{go} is the minimum time for going to the initial position of the service, and t_{ret} is the minimum time required to return. Analogously, $dist(s)$ is defined as follows:

$$dist(s) = km_{go} + km_{serv} + km_{ret} \quad (7)$$

where km_{serv} is the number of kilometers of the service, km_{go} are the kilometers for go and km_{ret} are the kilometers for return.

Regarding heuristics, we have used the following:

- sorting variables: services have been ordered according to their initial start time
- sorting values: drivers to be assigned to each variable have been ordered according to their costs; so drivers with lower costs (basic, per hour, and per kilometer) are tried first
- sorting constraints: overlapping constraint, driving time, journey length and cumulated driving time

Example. Suppose that there are five different locations in which the road passenger transportation company works. The distance in time units between each location is shown in Figure 2. The establishment of the time distances is performed according to our relationship between time units and kilometers (half an hour, 45 km). For example, the distance between locality 2 and 1 is 3 units of time (3*0.5=1,5 hours), which is equivalent to 135 km (3*45).

Regarding drivers, let us suppose the ones provided in Table 1. For a given day, the services requested from the company are the ones shown in Table 2.

The following constraints are considered:

- maximum driving time (MDT): 22 time units
- maximum journey length (MJ): 30 time units
- maximum driving time per two weeks (MTB): 180 time units

The optimal solution to this problem is the one provided in the Gantt diagram of Figure 3. Requested services are in dark, while intervening services are in grey. Note also that there are some gaps in the drivers' journeys corresponding to non-occupied hours. Some drivers do not participate in the allocation because either their cost is higher or their cumulated times do not allow it. Table 3 gives the cumulated times per driver together with the cost of the solution. DT is the driving time, JL is the journey length, FT is the free time, HW is the week's cumulated time, and S the services assigned to the driver. Observe that only requested services are given, but all drivers have an intervening service when required according to the Gantt diagram (see Figure 3). Note also that driver 9 has an intervening service for moving from the end location of service 5 to the starting location of service 8.

To illustrate the interactions between planning and scheduling, suppose that one of the constraints is changed. For example, the maximum journey length is 25 time units instead of 30. Then, it is clear that additional drivers should

Driver	bc	kmc	sl	fl	hw	hc
d_1	20	0.6	0	0	100	0.6
d_2	30	0.7	0	0	90	0.4
d_3	30	0.4	0	0	70	0.7
d_4	25	0.6	0	0	65	0.6
d_5	15	0.6	0	0	70	0.5
d_6	15	0.5	0	0	100	0.4
d_7	15	0.7	0	0	180	0.5
d_8	13	0.5	0	0	165	0.5
d_9	16	0.6	0	0	100	0.5
d_{10}	13	0.6	0	0	110	0.6
d_{11}	16	0.5	0	0	58	0.7

Table 1: Available drivers.

Service	sl	fl	st	ft
s_1	1	2	19	22
s_2	2	3	5	11
s_3	2	4	10	17
s_4	4	0	25	29
s_5	3	1	13	17
s_6	4	3	7	14
s_7	3	1	21	25
s_8	2	3	20	26
s_9	2	3	10	16
s_{10}	3	4	15	22
s_{11}	1	4	25	28

Table 2: Services requested.

Driver	DT	JL	FT	HW	S	Cost
d_2	22	27	5	117	s_9, s_7, s_{11}	174.9
d_5	16	16	0	86	s_3	114.2
d_6	22	26	4	126	s_6, s_{10}, s_4	137.3
d_8	14	14	0	179	s_2	93.5
d_9	19	19	0	119	s_5, s_8	133.8
d_{10}	11	11	0	121	s_1	88.9
Total cost						742.6

Table 3: Solution of the example

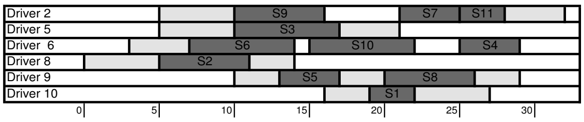

Figure 3: Gantt diagram of the optimal solution

Driver	Services
d_2	s_2, s_5
d_5	s_3
d_6	s_9, s_7
d_8	s_8
d_9	s_{10}, s_4
d_{10}	s_1, s_{11}
d_{11}	s_6

Table 4: Solution of the example with MJ=25.

be required to develop the services; and, as a consequence, the duties of each driver are also different, as illustrated in Table 4.

The journey approach

When, adopting the journey view, two main steps should be performed:

1. planning: journey and plan generation
2. scheduling: repeat the branch and bound method for each plan to allocate journeys of the plan to the available drivers

In the following section, the formalization of the problem according to the journey approach, the modeling of the planning and scheduling approach, and an illustrative example are given.

Problem formalization. In the journey approach, we need to include the definition of a journey, a plan, and allocation based on journeys.

Definition 5. A *journey* is subsets of non overlapping services

$$j_i = \{s_1, ..., s_p\}$$

where $s_j \in S \cup I$, in which the maximum driving time and maximum journey length constraints are satisfied.

Two journeys j_i and j_k can have common services, $j_i \cap j_k \neq \emptyset$. For this reason, some journeys are incompatible. Then,

Definition 6. A *plan* p_k is a set of compatible journeys, such that all services are covered and none are repeated $p_k = \{j_{k1}, ..., j_{kn}\}$.

Definition 7. An *allocation based on journeys* is a tuple

$$A_i =< (j_1, d_{i_1}), (j_2, d_{i_2}), ...(j_l, d_{i_l}) >$$

where j_k is a journey, all journeys belong to the same plan, $d_k \in D$, $S \subset \bigcup_k j_k$ (all services are covered) and in which all constraints are satisfied.

The cost function is the same as 2.

Planning modeling. The planning step includes both journey and plan generation. In this step we consider the overlapping constraint, the maximum journey length constraint and the maximum driving time.

The outcome of this planning step is the set $P = \{p_1, ..., p_n\}$ of all possible plans. Plans are sorted according to their estimated cost. The estimating function is based on the definition of a cost matrix C, of p*n dimension (rows are journeys, columns are drivers), in which an element $c_{i,j}$ represents the cost of journey j_i when the driver d_j is assigned[3].

Then, given a plan $p_i = \{j_{i_1}, ..., j_{i_k}\}$, the estimating function is defined as follows:

$$f(p_i) = \sum_{j_x \in p_i} min_{y \in D}(c_{x,y}) \quad (8)$$

See more details of the estimating function in (Murillo 2006).

Scheduling modeling. The scheduling step consists of assigning drivers to the journeys of plans. First, an allocation for the first plan p_1 of P and its corresponding cost $C(p_1)$ are computed. This cost is set as their upper bound α. Then, all the successive plans p_i of P are analyzed while $f(p_i) < \alpha$ (see the algorithmic schema in Figure 4).

A single allocation problem is modeled according to a constraint-based approach. Variables represent journeys, domains are drivers, and, at most, drivers can be assigned once to a variable. Constraints are related to journeys. Note, however, that it is not necessary to take into account the overlapping constraint, since this constraint has been taken into account in the journey generation preprocess. The same happens with the maximum driving time and maximum journey length constraints. The unique constraint handled in this step is the maximum driving time per two weeks.

Let $\alpha = \infty$
for each p_i in P **do**
  if $f(p_i) < \alpha$ **then**
    find best allocation A_i for p_i
    if $cost(A_i) < \alpha$ **then**
      solution = A_i
      $\alpha = cost(A_i)$
    end-if
  end-if
end-for

Figure 4: Algorithm pseudocode for the scheduling step.

To prune the search space we use the same estimating function of equation 8, taking into account that instead of a complete plan, we estimate the cost of the remaining journeys of a plan. In the assignment step, the following heuristics are also applied:

[3]When a driver cannot perform the journey, this cost is set to ∞.

- variables (journeys inside a plan) have been sorted according to their cost: the cheapest one first
- values (drivers) are also sorted according to their cost (minimum first)
- since there is a single constraint, no sorting is required for them

Example. The different services required for the example of Table 2 can be organized in 40 journeys as shown in Table 5. In general, with m services, $2^m - 1$ journeys can be generated corresponding to all possible combinations, but some of them are pruned according to the overlapping and journey length constraints as well as the intervening services. In our case, with $m = 11$ we obtain 40 journeys, which is a number less than the analytically one required, $40 << 2^{11} - 1$.

In the first step of the solution, we obtain a total of 1018 plans, as a result of the compatible combination of the 40 journeys. In general, a total of $2^k - 1$ plans can be obtained, k being the number of journeys. But due to journey incompatibility, a lower number of plans are effectively generated.

The solution to the problem according to the journey approach is the plan $p = \{j_1, j_2, j_{19}, j_{30}, j_{31}, j_{32}\}$ with the following journey allocation: $< (j_1, d_2), (j_2, d_6), (j_{19}, d_9), (j_{30}, d_{10}), (j_{31}, d_8), (j_{32}, d_5) >$. This solution has obviously the same cost as in the service case example.

As in the service approach, let us suppose that the maximum journey length changes, from 30 time units to 25. Then, a different lower number of plans is generated, and the optimal scheduling is the following: $< (j'_2, d_6)$, (j'_{10}, d_2), (j'_{14}, d_5), (j'_{11}, d_{10}), (j'_1, d_9), (j'_{19}, d_8), $(j'_{17}, c_{11}) >$ where

$$\begin{aligned} j'_2 &= \{s_9, s_7\} \\ j'_{10} &= \{s_2, s_5\} \\ j'_{14} &= \{s_3\} \\ j'_{11} &= \{s_1, s_{11}\} \\ j'_1 &= \{s_{10}, s_4\} \\ j'_{19} &= \{s_8\} \\ j'_{17} &= \{s_6\} \end{aligned}$$

In the case that the reduced constraint is the maximum driving time per two weeks, the generated plans are the same, but additional resources will be required in the scheduling step.

Analytical alternative comparison

Given a set of n drivers and m services, the service approach has a computational complexity of

$$n^m \qquad (9)$$

Regarding the journey approach, we need to analyze three components of the complexity:

1. Journey generation: $nj = (2^m - 1)$
2. Plan generation: $np = 2^{nj} - 1 = 2^{(2^m - 1)} - 1$
3. Allocation generation: $np * \frac{n!}{(n-njp-1)!}$, where njp is the number of journeys per plan. In the worst case, $njp = m$.

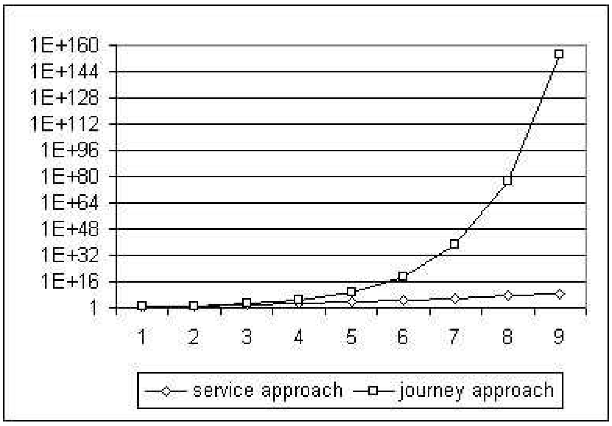

Figure 5: Analytical complexity comparison (logarithmic scale).

Of course, the latter component is the dominant one. In an extensive form, it is the following:

$$(2^{(2^m-1)} - 1) * \frac{n!}{(n - m - 1)!} \qquad (10)$$

In order to compare the 9 and 10 expressions, we can suppose $n = m$. Then, we have the following complexity for both approaches:

$$\begin{array}{ll} service & n^n \\ journey & (2^{(2^n-1)} - 1) * n!) \end{array}$$

Figure 5 shows the corresponding plots.

Even though the analytical complexity in the journey approach is higher than in the service approach, their experimental performance is better in the former, as shown in the next section.

Experimentation

In order to experimentally analyze the different techniques, up to 70 examples have been generated with different complexities. The first example has a single service and a single driver; the second example two services and two drivers; and so on until the 70th example. The data corresponding to service and drivers have been generated randomly for each example. In this sense, the complexity of the 70th example is greater than in a real case of the application we are dealing with.

In the graphic of Figure 6 we can see the execution time in milliseconds of the service and journey approaches. This execution time includes the planning and the scheduling time in the journey approach (the scheduling time is close to 0). The service approach can solve the problem in a reasonable time in up to 11 services while the journey approach can solve it in up to 20 services.

So the journey approach shows a better performance than the service approach. From our understanding, this is due to three main reasons. First, in the service approach we

$j_1 = \{9,7,11\}$	$j_2 = \{6,10,4\}$	$j_3 = \{6,7,11\}$	$j_4 = \{5,7,11\}$	$j_5 = \{5,1,11\}$
$j_6 = \{2,10,4\}$	$j_7 = \{2,5,4\}$	$j_8 = \{10,11\}$	$j_9 = \{10,4\}$	$j_{10} = \{9,11\}$
$j_{11} = \{9,7\}$	$j_{12} = \{9,4\}$	$j_{13} = \{7,11\}$	$j_{14} = \{6,11\}$	$j_{15} = \{6,10\}$
$j_{16} = \{6,7\}$	$j_{17} = \{6,4\}$	$j_{18} = \{5,11\}$	$j_{19} = \{5,8\}$	$j_{20} = \{5,7\}$
$j_{21} = \{5,4\}$	$j_{22} = \{5,1\}$	$j_{23} = \{3,11\}$	$j_{24} = \{3,4\}$	$j_{25} = \{2,10\}$
$j_{26} = \{2,7\}$	$j_{27} = \{2,5\}$	$j_{28} = \{2,4\}$	$j_{29} = \{1,11\}$	$j_{30} = \{1\}$
$j_{31} = \{2\}$	$j_{32} = \{3\}$	$j_{33} = \{4\}$	$j_{34} = \{5\}$	$j_{35} = \{6\}$
$j_{36} = \{7\}$	$j_{37} = \{8\}$	$j_{38} = \{9\}$	$j_{39} = \{10\}$	$j_{40} = \{11\}$

Table 5: Journeys of the case example

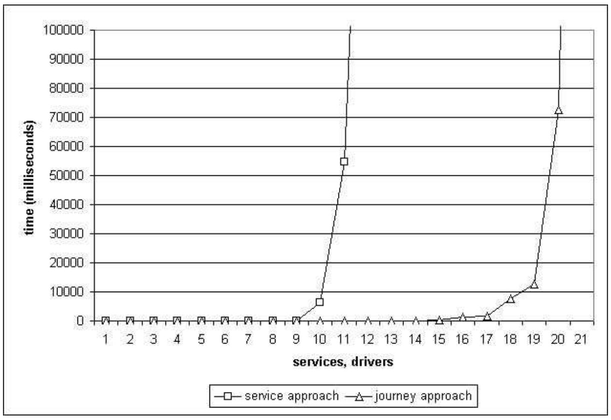

Figure 6: Execution time in service and journey approach

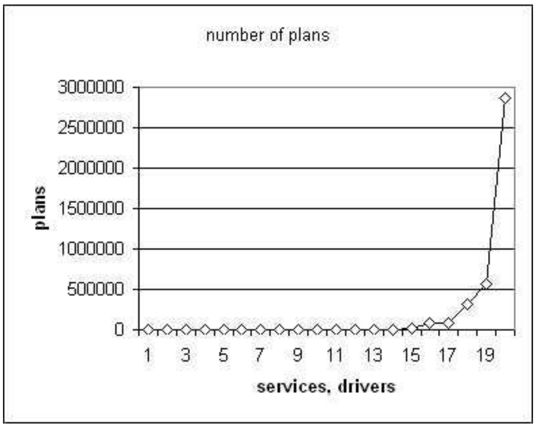

Figure 7: Number of plans generated.

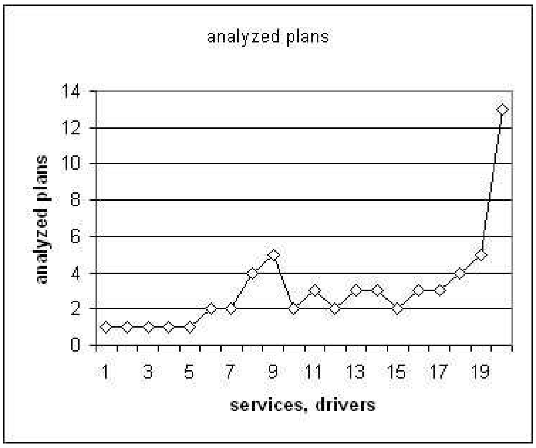

Figure 8: Number of plans analyzed.

are dealing with more constraints at once than in the journey approach, in which we consider some constraints in the planning step and some others in the scheduling step. In our problem, even though constraints can be quite complex they have few inter-dependencies, and they can be split and treated locally. Dealing whit all constraints together can have a multiplicative impact on the performance, as the results show in the service approach.

Second, in the journey approach the number of alternatives in the search space is also reduced. A driver can only be assigned once, so when he is assigned this value, he is pruned for the remaining assignments.

And third, dealing with the planning and the scheduling separately facilitates the definition of more specific heuristics and specialized estimated cost functions. Particularly, in the journey approach we can estimate the intervening services cost more accurately.

The better performance with the journey approach, however, is not gratuitous. First of all, the modeling effort is higher. And second, we need to store all the plans in memory. In Figure 7 we can see the number of plans generated in the journey approach. Only a very small part of them are analyzed (see Figure 8); the rest are pruned for the estimating function. Memory requirements, however, can be improved by storing only potential plans. Our current work is focused on that.

Conclusions

Even though there has been recent interest in developing integrated frameworks for planning and scheduling, there is no common agreement on whether such integration would be practical or not (Bresina *et al.* 2002; Smith, Frank, & Jonsson 2000). In this paper we have analyzed the road passenger transportation problem from both points of view: keeping planning and scheduling separate, and merging the two approaches. We have solved the problem in both approaches by means of branch and bound method. The results show that, for our particular application, keeping planning and scheduling separate achieves a better performance. The sequentiality of planning and scheduling filters out some of the constraints in the scheduling process, making the latter easier. Moreover, the domain of variables in the assignment step is reduced.

In addition, considering planning and scheduling separately, we can define specific heuristics and cost functions that significantly reduce the search space. So, even though complexity analysis indicates that the integration framework has lower complexity than the isolated one, the experimental results tell us the opposite.

As a drawback, the journey approach requires far more memory than the service approach to store plans. However, this is an issue that can be overcome in the near future.

The results obtained are particular to our problem, in which complex constraints are tackled, but there are few interdependencies between them. From the results obtained, we will continue working in planning and scheduling as separate steps in our particular problem. We know that our results can hardly be extended to other domains, but we maintain the claim that keeping planning and scheduling separate is still a good solution for some kinds of problems.

References

Ai-Chang, M.; Bresina, J.; Charest, L.; Chase, A.; jung Hsu, J. C.; Jonsson, A. K.; Kanefsky, B.; Morris, P.; Rajan, K.; Yglesias, J.; Chafin, B.; Dias, W.; and Maldague, P. APGEN: Mixed-initiative planning and scheduling for the mars exploration rover mission. *IEEE Intelligent Systems* 19(1):8–12.

Barták, R., and Rudová, H. 2001. Integrated modelling for planning, scheduling, and timetabling problems. *Proceedings of the Twentieth Workshop of the UK Planning and Scheduling Special Interest Group (PLANSIG), Edinbourgh* 19–31.

Bresina, J.; Dearden, R.; Meuleau, N.; Ramakrishnan, S.; and Washington, R. 2002. Planning under continuous time and resource uncertainty: a challenge for AI. *Proc. 19 th Conference on Uncertainty in AI.*

Cantillon, E., and Pesendorfer, M. 2006. *Auctioning bus routes: The London experience*. MIT Press. To appear.

Castillo, L.; Fdez-Olivares, J.; García-Pérez, O.; and Palao, F. 2006. Efficiently handling temporal knowledge in an htn planner. *Proc. ICAPS.*

Dechter, R. 2003. *Constraint Processing*. Morgan Kaufmann Publishers.

Esclapes, C. 2000. *Asignación de condugtores a jornadas de trabajo en empresas de transporte colectivo*. Ph.D. Dissertation, Technical University of Catalonia.

Gallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: Theory and Practice*. Elsevier - Morgan Kaufmann Publishers.

Haslum, P., and Geffner, H. 2001. Heuristic planning with time and resources. *Proc. IJCAI-01 Workshop on Planning with Resources.*

Hoffman, K., and Padberg, M. 2000. Set covering, packing and partitioning problems. *Encyclopedia of Optimization.*

López, B. 2005. Time control in road passenger transportation. problem description and formalization. Research Report IIiA 05-04, University of Girona.

Loureno, H. R.; Paixao, J. P.; and Portugal, R. 2001. Metaheuristics for the bus-driver scheduling problem. *Transportation Science* 3(35):331–343.

Murillo, J. 2006. Road passenger transportation: study and resolution with optimization techniques. Msc, University of Girona. In Catalan.

Pezzella, F., and Faggioli, E. 1997. Solving large set covering problems for crew scheduling. *Sociedad de Estadística e Investigación Operativa* 5(1):41–61.

Smith, D.; Frank, J.; and Jonsson, A. 2000. Bridging the gap between planning and scheduling. *the Knowledge Engineering Review* 15(1):61–94.