On the Identification and Use of Hierarchical Resources in Planning and Scheduling

Bernd Schattenberg and **Susanne Biundo**
Department of Artificial Intelligence
University of Ulm, Germany
Email: `firstName.lastName@informatik.uni-ulm.de`

Abstract

Many real-world planning applications have to deal with resource allocation problems, and so does planning in the domain of crisis management assistance. In order to support resource allocation in these kind of applications, we present a new approach to the integration of scheduling capabilities and planning. The proposed methodology relies on a hybrid planner, which combines action and state abstraction by integrating hierarchical task network (HTN) planning and state based partial order causal link (POCL) planning into a common framework. We extend the abstraction mechanism of the planner to different kinds of abstraction for resources, namely subsumption, approximation, qualification, and aggregation.

We show how these abstractions can be used when modeling the domain and how reasoning about resources can be performed in a flexible way, namely by merging opportunistic planning and scheduling strategies.

Introduction

Many real-world planning problems, like those in the domain of crisis management support, are challenging in two dimensions: They demand huge computational and representational capabilities of the supporting system. Their solutions typically consist of very large courses of action, which make considerable use of all kinds of resources, ranging from limited time and building material to manpower and supplies. In our ongoing project we deal with scenarios taken from the mission of the German *Technisches Hilfswerk* (THW) – a governmental disaster relief organization – within the flood disaster at the river *Oder* in July 1997.

From a planning perspective, these kind of problems show two aspects: to find suitable courses of action to reach a specified goal state from a given initial situation and to allocate resources, i.e. the assignment of resources and time to the actions in order to guarantee the success and efficiency of the mission.

In the past, these two aspects of the planning task were mostly considered to be two different kinds of problems, called *planning* and *scheduling*, one performed after the other. Newer approaches take into account, that there are many real-world domains where the two problem solving phases interact to a huge extent, and so neither can be reasonably carried out without knowledge about the progress of the other. For those classes of problems we agree with the arguments of authors like (Laborie & Ghallab 1995a) and completely integrate the scheduling steps in the plan generation phase by regarding resource manipulation as plan modification steps. Adopting this view, the planning system can perform its generation procedure in a least commitment and, as we will show, opportunistic manner. The ultimate goal of our effort is a framework that is flexible enough to handle all nuances of planning problems from pure *planning* to pure *scheduling*, and that is easy to model and promises to be reasonably efficient in finding a solution.

In order to meet this challenge, the work in this paper extends the approach in (Biundo & Schattenberg 2001), which integrates the action abstracting *hierarchical task network (HTN) planning* paradigm and state abstracting operator based techniques (*POCL*). We add resource manipulating expressions to the preconditions and effects of tasks and integrate reasoning about resources on different levels of abstraction. To this end, we identify four types of resource abstraction, namely:

Subsumption defines one resource to be a specialization of another;

Approximation relaxes upper and lower bounds for numerical values;

Qualification implements the transfer from symbolic to numerical values;

Aggregation groups components to super-structures.

We adapt these concepts to our formal framework in (Biundo & Schattenberg 2001) by introducing appropriate decomposition axioms for resources.

The planning strategy generates plans with resource information in a *least commitment* way. We define scheduling steps as additional plan modification operations, which interleave with the planning process in all aspects: task decomposition, task insertion, closing of preconditions, and conflict detection and resolution. The proposed approach can opportunistically and smoothly move between the scheduling and the hybrid planning paradigm. It allows for an informed merging of planning and scheduling, where both processes can share all the knowledge about the planning task.

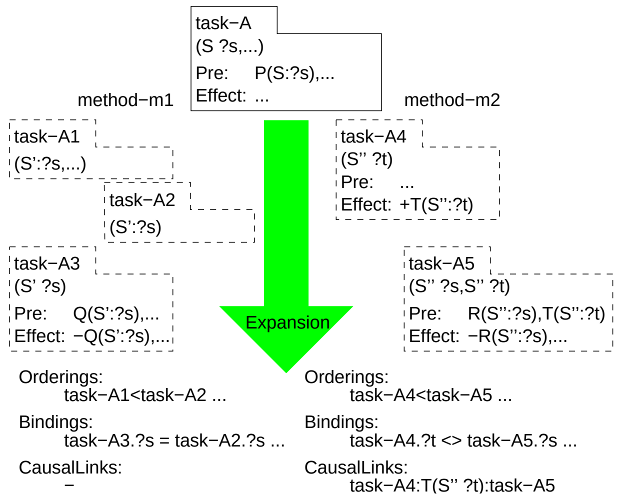


Figure 1: An abstract task with two methods for expansion.

The rest of the paper is organized as follows. First, we introduce the hybrid planning approach which we will extend in this paper. We then present different types of resource abstractions we have identified and define them in terms of our framework. After that, we describe how the hierarchical resource information is exploited in our hybrid planning system and how the additional plan modification steps are integrated in the planning strategy. The paper ends with an overview over related approaches and some concluding remarks.

The Hybrid Planning Framework

Our approach is based on a planning methodology which integrates HTN planning and operator-based techniques (Biundo & Schattenberg 2001). It takes the notion of complex or *abstract* and *primitive tasks* from the HTN paradigm (Erol, Hendler, & Nau 1994, e.g.), in which abstract tasks are stepwise refined into *networks* of primitive ones (plan fragments) using so-called *methods*, which thereby relate the different levels of action abstraction. In contrast to classical HTN planners, our approach provides all tasks with preconditions and effects to facilitate reasoning about causal interactions even on the most abstract levels of plan refinement. Figure 1 shows an abstract task A which has a variable $?s$ of sort S in its signature and which contains the unary predicate $\mathrm{P}(S{:}?s)$ as its precondition. The rest of the precondition formula and the effects are omitted. For the expansion of A we defined two methods, which decompose it into the two networks shown. For example, the one on the left side in Figure 1 contains three subtasks, each of which carrying a variable of sort S' in its signature. These plan fragments carry constraint sets that constitute an ordering on the subtasks and define the variable bindings among the subtasks and the super-task. Not shown in the figure are further domain constraints.

The formal semantics of our methodology rely on work on logic based planning (Stephan & Biundo 1996). Based

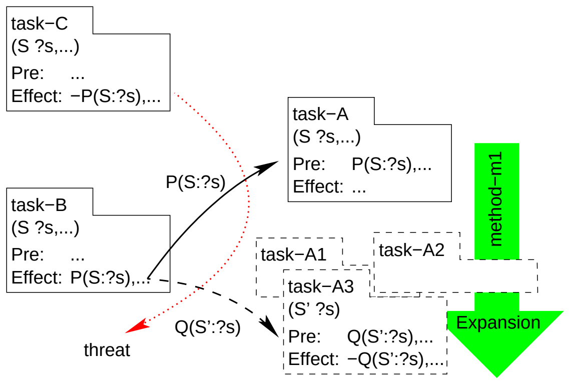


Figure 2: Refinement of the causal structure along the expansion hierarchy.

on this framework we introduce sort hierarchies on domain objects and so-called *decomposition axioms*. These axioms, together with the sort and sub-sort definitions, impose a hierarchy on the relations and objects in the domain. For the examples in Figures 1 and 2 we assume, that S' and S'' are two sub-sorts of S, and that the following decomposition axiom is defined:

$$\mathrm{P}(S : ?s) \leftrightarrow \mathrm{Q}(S' : ?s) \vee \mathrm{R}(S'' : ?s)$$

These hierarchies are used to relate pre- and postconditions of tasks on different levels of abstraction, thereby enabling the definition of so-called *legal decompositions of tasks* (cf. (Biundo & Schattenberg 2001)).

However, this mechanism can not only be used to justify expansion schemas but also enables the system to keep track and reason about the causal interactions in a plan, even if its tasks are on different levels of abstraction. As an example, Figure 2 shows one expansion step, using the definitions above: A task B establishes the condition $\mathrm{P}(S{:}?s)$ for task A, denoted by a causal link, like it is used in state based POCL planning. Another task C threatens this link. It has $\neg\mathrm{P}(S{:}?s)$ as an effect and the ordering relation allows C to be executed between B and A. Let us assume that the system ignores the threat in this situation and decides to expand the abstract task A, using the first method. Justified by the decomposition axiom and sort hierarchy as defined above, the planner passes the established causal link from A to its subtask A-3 and specializes the link's annotation accordingly to $\mathrm{Q}(S'{:}?s)$. The important point is, that after the expansion step the – now abstract – threat could still be detected by taking the decomposition axioms into account.

This means, we do not loose any causal information during refinement and we therefore can "safely" decompose any abstract task. Furthermore, expansion can be used to split up and solve abstract threats, because when conflicting abstract tasks cannot be ordered and non-codesignation does not work, an overlapping of the expanded networks may result in a consistent plan.

The mechanism can also be used to identify condition establishers or the need to insert a new task, independent from the level of abstraction of the tasks involved. The planning strategy is therefore free to decide when and how to close an open precondition, i.e. the methodology allows for a flexible least commitment planning strategy ranging from pure HTN to a pure POCL style of planning.

The Role of Abstraction for Resources

When looking at planning domains like crisis management in a flood disaster scenario, a large number of resources can be identified, which play an important role in finding a solution for a given planning problem. For example, when securing a dike, thousands of sand sacks have to be prepared and installed, several types of specialized vehicles fortify constructions or build new ones – assisted by workers on the dykes and divers in the water. Furthermore, various kinds of tools and materials have to be organized, including power supplies for the illumination of the working area at night, for example.

Within this setting, human planners can use or combine four different kinds of resource abstraction, each of which will be presented in the following sections: *subsumption*, *approximation*, *qualification*, and *aggregation*. These resource abstractions impose hierarchies on resources, which enable an integration of scheduling capabilities in the hybrid planning approach.

Temporal projection on these resources is done by computing so-called *resource profiles* in the fashion of (Drabble & Tate 1994). Each profile represents cumulated resource capacities according to the consuming and producing tasks in the current plan. If at some point in time some profiles' values get below zero, the plan necessarily over-consumes the associated resources at this level of abstraction and the system has to decide on repairing steps.

The examples in the following sections deal almost exclusively with allocating symbolic resources in the preconditions of the tasks. If not stated otherwise the presented techniques can be utilized for numerical resources as well. Furthermore, the mechanism works in the same way for post-conditions of tasks, viz. for consumption, production and de-allocation.

Subsumption

Subsumption is expressed by defining one type of resource to be a specialization of the other, i.e. by a sort hierarchy. Figure 3 shows an example of a sort hierarchy on resources, which play a role in the example domain. Typically, there exist several types of transportation units, therefore we have an abstract transportation task allocating one unit of an abstract resource called (THW-) *Unit* – indicated by the dashed dark arrow. After one expansion step however, the task is specialized into a more concrete way of transportation. For example, the planner chooses a method that decomposes the abstract task into a network which contains the shipping transportation task. According to its signature and a decomposition axiom that looks like

$$\begin{aligned}&\text{At}(\text{Unit: } u, \text{Location: } \textit{from}) \leftrightarrow \\ &\quad [\ \ \text{Standing-at}(\text{Vehicle: } u, \text{Location: } \textit{from}, \text{Road: } r) \vee \\ &\qquad \text{Aircraft-at}(\text{Aircraft: } u, \text{Location: } \textit{from}, \text{Height: } h) \vee \\ &\qquad \text{Boat-at}(\text{Boat: } u, \text{Location: } \textit{from}, \text{Water-street: } w) \vee \ldots],\end{aligned}$$

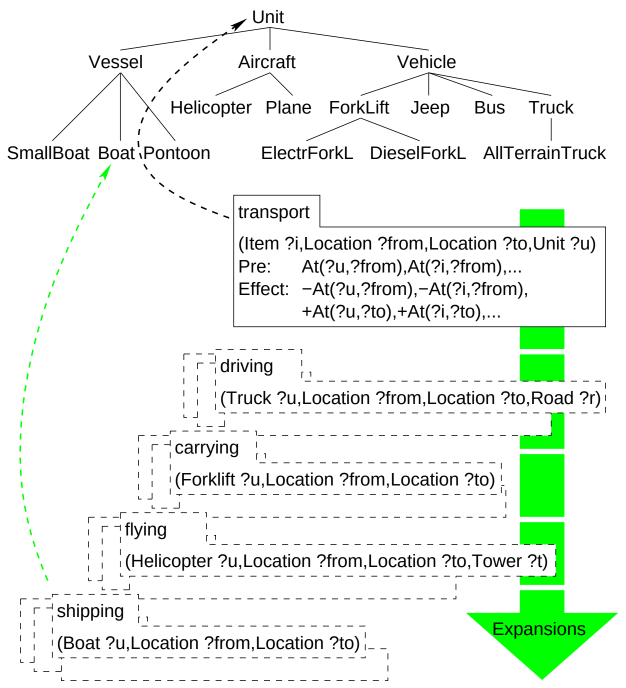

Unit
Vessel
Aircraft
Vehicle
Helicopter
Plane
ForkLift
Jeep
Bus
Truck
SmallBoat
Boat
Pontoon
ElectrForkL
DieselForkL
AllTerrainTruck
transport
(Item ?i,Location ?from,Location ?to,Unit ?u)
Pre: At(?u,?from),At(?i,?from),...
Effect: −At(?u,?from),−At(?i,?from), +At(?u,?to),+At(?i,?to),...
driving
(Truck ?u,Location ?from,Location ?to,Road ?r)
carrying
(Forklift ?u,Location ?from,Location ?to)
flying
(Helicopter ?u,Location ?from,Location ?to,Tower ?t)
shipping
(Boat ?u,Location ?from,Location ?to)
Expansions

Figure 3: Abstraction of resources by building a subsumption hierarchy.

one unit of the more concrete resource *Boat* has to be allocated (dashed light arrow), which is consistent with the subsumption defined above.

Reasoning within this hierarchy is quite similar to the mechanism involved in dealing with symbolic causal interactions in our framework for hybrid planning. As every subresource qualifies for being allocated instead of one of its super-sorts, usage profiles have to take into account that

1. every allocation of a resource belonging to a sub-sort implies an allocation of a resource of the respective super-sort, and
2. every allocation of a resource belonging to a super-sort which has not been specialized yet, implies a *possible* allocation of a resource for every sub-sort.

As we will see later on, the second argument can be used to reveal and resolve conflicts in possibly over-consuming plans.

Approximation

Numerical values for resources and their manipulation operations may be estimated on the more abstract level of a plan. Durations for tasks are often approximated, as their exact time consumption cannot be calculated precisely until the very concrete plan refinement level has been reached with all necessary tasks inserted in the plan (cf. Figure 4). It shows possible refinements of the transportation task together with an informal account of the respective durations. Here, the overall time interval for the abstract transportation task for example over-estimates all of its possible refinements. The

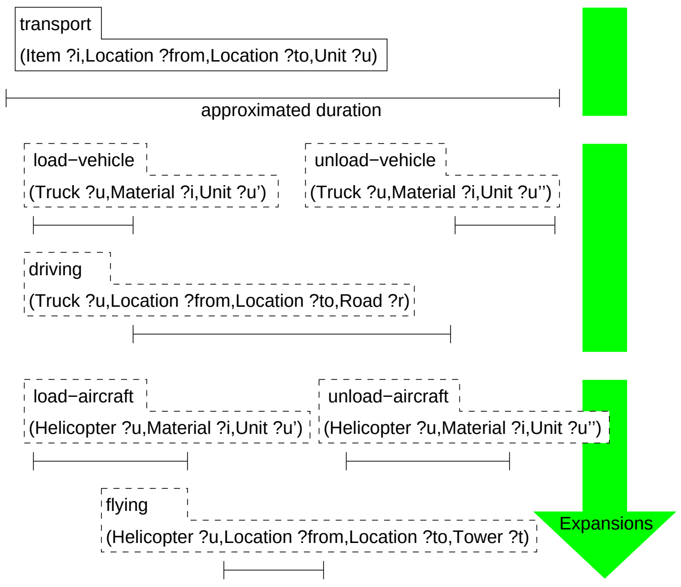

transport
(Item ?i,Location ?from,Location ?to,Unit ?u)
approximated duration
load−vehicle
(Truck ?u,Material ?i,Unit ?u')
unload−vehicle
(Truck ?u,Material ?i,Unit ?u'')
driving
(Truck ?u,Location ?from,Location ?to,Road ?r)
load−aircraft
(Helicopter ?u,Material ?i,Unit ?u')
unload−aircraft
(Helicopter ?u,Material ?i,Unit ?u'')
flying
(Helicopter ?u,Location ?from,Location ?to,Tower ?t)
Expansions

Figure 4: Abstraction of resources by approximation. The abstract task over-estimates the duration of every expansion.

sum of the durations for the expansion which contains the transportation by helicopter and its loading operations is in particular smaller than that of the tasks involved in delivery by truck.

The estimation of duration parameters can be modeled through intervals with lower and upper bounds ranging from zero to infinite. Infinite upper and zero lower bounds represent open intervals for the concepts “at least” and “at most”, respectively. Approximation facilitates a very natural way of changing the view on numerical values from one level of abstraction to the next. In a situation like in Figure 4, we would like to model, for example, that the abstract transportation takes at least four hours, the loading procedure at most ten minutes, and so on. However, we note, that by this kind of (numerical) approximation we may sacrifice an important monotonicity property, viz. it does not guarantee to predictably over-estimate increasing resource manipulations and under-estimate decreasing ones along the task refinements. Therefore, it cannot be used to cut the search space as efficiently as desirable: For example, a resource over-consumption may turn out harmless after another refinement step if production and consumption are not consistently over- and under-estimated, respectively. Another point here is the accuracy – or in some sense admissibility – of the approximating function. The more precise the estimations are, the better can resource information guide search.

On the other hand, we can use the method definitions to check for inconsistently over- and under-estimating approximation hierarchies off-line. Similar calculations can guide the scheduling process by suggesting to assign interval restrictions, when particular expansions are ruled out by the planning process: When according to Figure 4 the system has unsuccessfully tried to use the transporting method using trucks, the abstract time estimation can be reduced.

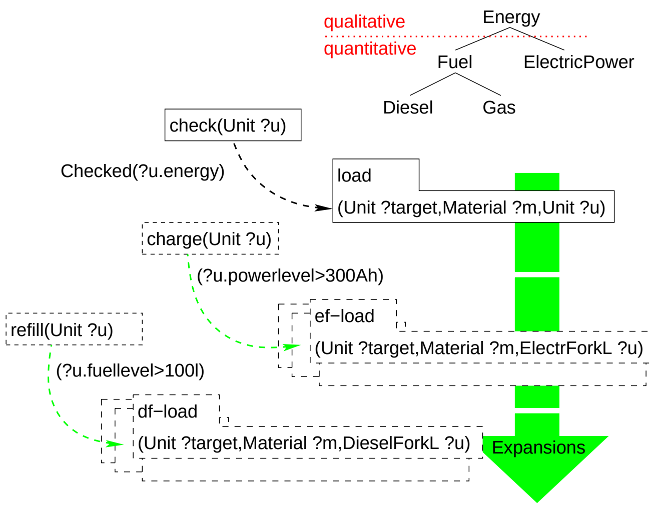


Figure 5: *Energy* as abstraction of resources by qualifying the numerical resources *Fuel* and *Electric Power*.

Qualification

In some cases, approximation may not make much sense when building a task hierarchy. For the abstraction of quantitative, numerical values, the modelers may want to use qualitative, symbolic terms. An example: There are many ways for loading material on a transportation unit, and all of them consume a certain (maybe approximated) numerical amount of energy. The two gadgets for performing the tasks are electrical fork lifts at urban supply centers and fork lifts driven by diesel fuel on site. As their two energy sources do not reasonably correlate in their consumption to allow for a good estimation, we abstract from concrete numbers or intervals and speak of *energy* which has to be checked or re-filled before the loading task can be performed.

Figure 5 shows such a situation: The resource hierarchy on top implements qualification (see the dotted line) by defining *Energy* as the super-resource of *Fuel* and *Electric Power* with *Fuel* subsuming diesel fuel and gas. The abstract loading task on the right hand side shows the logical symbol *energy* of sort *Energy* in its precondition: We assume that energy has to be checked beforehand by an appropriate checking task, which carries a classical postcondition for adding the formula to the state description (denoted by the dark dashed arrow).

When the check task gets refined, for example into networks containing electrical or diesel driven fork lifts the resource is specialized by the qualification and subsumption hierarchies, together with a decomposition axiom[1]:

$$\text{Checked}(\text{Energy: } u.energy) \leftrightarrow$$
$$[\quad u.power\text{-}level > 300 \ \vee \ u.fuel\text{-}level > 100 \ \vee \ldots]$$

[1]The integration of numerical reasoning into the formal framework underlying our planning approach will be done following respective techniques developed in the areas of automated reasoning and theorem proving. The description of this integration is beyond the scope of this paper, however, and will be done elsewhere.

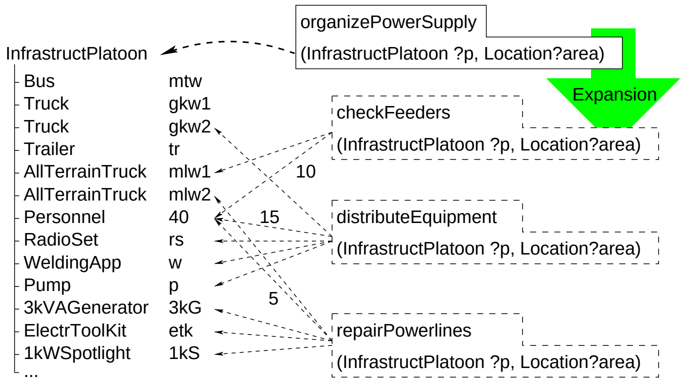


Figure 6: Abstraction of resources by aggregating components.

This results in a numerical quantity of three hundred ampere-hours electricity and one hundred liters diesel fuel needed, respectively, like it is shown by the light dashed arrows in the figure. The preparation tasks are also made more concrete (labeled `ef-load` and `df-load`) and provide now an appropriate energy resource.

The computation of profiles which involve qualified resources

1. propagates allocations of the not yet specialized qualitative resource through all profiles of the quantitative subresources. In the example above, 300*l* of diesel fuel *and* the given amount of electricity are allocated, until the task is refined. And it reversely
2. allocates the symbolic resource according to the qualification hierarchy for every numeric allocation.

The rationale behind this sort of hierarchical relation is to *determine* a symbolic causal structure on the abstract level, and to refine it into numerical quantities as soon as the planner decided which expansion schemas are appropriate.

Aggregation

Aggregation is a structural abstraction, used for resources which can be decomposed into more or less independent components. Many of them can be found in our scenario: THW units are typically organized in platoons. They consist of a small bus for the transportation of persons, two heavy trucks for carrying various gadgets, one mission specialized vehicle, and the personnel for operating the platoon. For example, if supporting measures for broken down energy or drinking water are needed, a so-called infrastructure platoon is ordered, which has the necessary equipment mounted on its trucks. On site, the platoon can be split up so that the power lines are repaired independently from handing out additional equipment to the relievers, etc.

Figure 6 shows an abstract task for organizing power supply, which allocates one of the infrastructure platoons (see dashed arrow). The components of the platoon are distributed among the tasks in the expansion like it is depicted with the small arrows: For example, checking the feeders in the area needs one of the all terrain trucks (ATTruck) and ten people working. This distribution is covered by decomposition axioms like the following one:

$$\begin{aligned}
&\text{Available}(\text{InfrastructPlatoon: } p, \text{Location: } \textit{area}) \leftrightarrow \\
&\;[\ \text{Standing-at}(\text{Bus: } \textit{mtw}, \text{Location: } \textit{area}, \text{Road: } r) \wedge \\
&\quad \text{Standing-at}(\text{ATTruck: } \textit{mlw1}, \text{Location: } \textit{area}, \text{Road: } r) \wedge \\
&\quad \text{Operational}(\text{RadioSet: } \textit{rs}, \text{Location: } \textit{area}) \wedge \\
&\quad = (\text{Personnel}, 40) \wedge \ldots \\
&\quad \vee \ldots\ ]
\end{aligned}$$

Reasoning about aggregated resources is similar to reasoning about subsumed ones. In order to project resource usage the system has to consider that in every profile

1. the allocation of aggregates implies an allocation for each of its components, and
2. every allocation of a component implies a possible allocation of an object of the aggregate sort, unless the component is assigned to an already allocated aggregate.

A consequence of the second point is that as soon as a task allocates a resource, which is a component of an aggregate sort, the profile for the component sort is updated first. Then the system has to check, whether the component can be consistently assigned to any instantiated aggregate in the current plan. If this is not the case, an other aggregate is allocated. Consistency of sharing components is regulated by domain axioms for the aggregate(d) sorts. In terms of the example, the above can be read as: If some concrete task wants to allocate one spotlight, and no platoon is present at the appropriate location, an infrastructure platoon has to be ordered.

Planning with Hierarchical Resources

Relying on the representation concepts for abstraction in planning with resources which we have defined so far, we will now present the utilization of the abstraction hierarchies in our planning framework.

Scheduling in the Presence of Planning

The main difference between a pure scheduling problem and an integrated one is that not all actions are known in the initial schedule. In the case of a hybrid planning problem the actions in the schedule are typically not on the same level of abstraction. In addition to that, tasks may be inserted at any stage of the plan generation process. Most heuristic-based scheduling systems do not work in such an environment, because their techniques over-constraint the partial solutions. An example would be shifting operations to minimize idle time between two actions: if not all activities are present in the schedule, the modifications might rule out the insertion of necessary preparation activities. We propose, that an integrated scheduler should develop its solution in a way comparable to that of the planner: in a *least commitment* strategy, first maintaining the partial solution consistent and then trying to refine it.

Therefore, we support the view of authors like (Laborie & Ghallab 1995a) who developed the IxTeT temporal planning system: Scheduling can be seen as a stepwise opportunistic schedule modification (or: refinement). A scheduler which follows this paradigm has mainly the choice between steps like:

- modify the ordering of the steps by inserting additional ordering constraints or by assigning time slots and narrowing time intervals respectively, and
- assign resources to plan variables in tasks.

We typically find this techniques in repair based scheduling systems like (Chien *et al.* 2000). Furthermore, these two kinds of manipulations correspond to the first two of the following plan modification options the hybrid planning system (Biundo & Schattenberg 2001) has:

- modify the task ordering by adding ordering constraints,
- assign values to plan variables, and
- modify the plan's causal structure by adding causal links, inserting new tasks or by expanding an abstract task into one of its methods task networks.

Motivated by the fact that plan generation should interact with the scheduling process in a way that allows for a well informed processing on both sides, we merge the scheduling steps above more or less unchanged as additional plan modification steps into a least commitment algorithm. That is to say, we replace purely explorative plan generation steps by resource sensitive schedule/plan modifications.

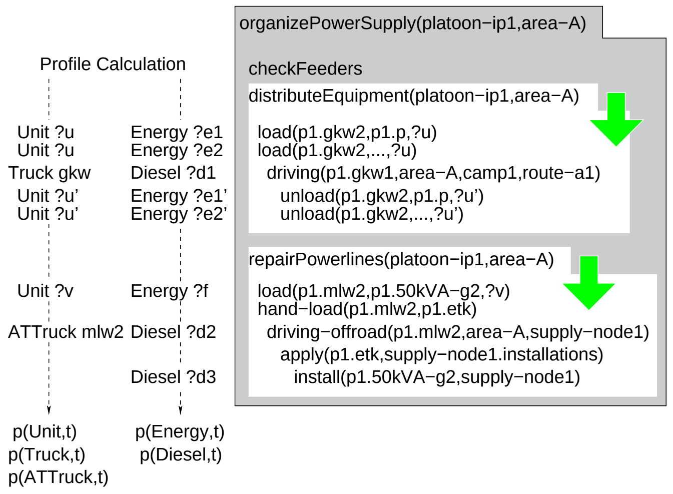
An Integrated Algorithm

The integrated algorithm we propose can be outlined as follows:

1. Check for causal and resource over-allocation threats, and open preconditions (including unassigned resources) in the current plan.
2. If there are none, and all tasks in the plan are primitive, then return the current plan as a result.
3. Try to resolve the first threat by one of the strategies:
   (a) add ordering constraints / separate time intervals for *promotion* or *demotion*;
   (b) add variable constraints for variable *(non-) codesignation* or *restrict* a conflicting (resource) variable to a more special sort;
   (c) *expand* any of the conflicting tasks or *insert* a task to adjust resource profiles (if this is not necessarily leading to a cycle).

   This (nondeterministic) choice is performed systematically under profile reasoning and is a backtracking point for the search. The algorithm gets the modified plan as a new input, but if repairing is not possible or no strategy is left, then a failure is returned.
4. If there are open preconditions, unassigned plan variables or abstract tasks in the current plan, perform one of the following steps:
   (a) *identify* and link a suitable establisher or *insert* and link a task to close the precondition;
   (b) *expand* one of the abstract tasks;
   (c) *narrow* the range for resources.

   These choices are again a backtracking point and subject to a planning strategy.



Profile Calculation
Unit ?u
Unit ?u
Truck gkw
Unit ?u'
Unit ?u'
Unit ?v
ATTruck mlw2
Energy ?e1
Energy ?e2
Diesel ?d1
Energy ?e1'
Energy ?e2'
Energy ?f
Diesel ?d2
Diesel ?d3
p(Unit,t)
p(Truck,t)
p(ATTruck,t)
p(Energy,t)
p(Diesel,t)
organizePowerSupply(platoon−ip1,area−A)
checkFeeders
distributeEquipment(platoon−ip1,area−A)
load(p1.gkw2,p1.p,?u)
load(p1.gkw2,...,?u)
driving(p1.gkw1,area−A,camp1,route−a1)
unload(p1.gkw2,p1.p,?u')
unload(p1.gkw2,...,?u')
repairPowerlines(platoon−ip1,area−A)
load(p1.mlw2,p1.50kVA−g2,?v)
hand−load(p1.mlw2,p1.etk)
driving−offroad(p1.mlw2,area−A,supply−node1)
apply(p1.etk,supply−node1.installations)
install(p1.50kVA−g2,supply−node1)

Figure 7: Relevant resource profiles after the expansion of an abstract task.

Example

The following example describes how the presented algorithm is used in the integrated planning and scheduling process.

We start with an initial situation including a task which uses an infrastructure platoon (cf. Figure 6) as an aggregated resource. Figure 7 shows a possible expansion of the abstract task for caring about power supplies. The shaded area represents the task network which replaces the abstract task in the current plan. Most constraint sets and other tasks are omitted for sake of simplicity. We will focus on the two subtasks `distributeEquipment` and `repairPowerLines`: the first one is supposed to transport some supplies to a local camp, the second one installs one of the larger power generators at a presumably broken supply node. The light areas indicate the expansions of the two subtasks. The indent depth indicates a partial ordering of subtasks. For example, the implementing method of `distributeEquipment` starts with two loading tasks followed by a transportation respectively driving step and the unloading operations at the end. We skipped the refinement of the transportation (cf. Figure 3), which is forced by the variable binding constraints of the surrounding network to be instantiated with one of the `Truck` units, and hence no other expansion method (flying, etc.) can be used.

On the left, the preparation of the profile calculation is depicted by showing some of the allocated resources at the current level of plan refinement: The profile for abstract THW units over time –denoted by *p*(Unit,*t*)– can identify two disjoint time points for *Unit* allocations in the network of `distributeEquipment`, and one in that of `repairPowerlines`. Codesignation constraints in the first task network assign the same object to the two loading tasks, and the same to be used in the unloading procedure respectively: this suggests ordering constraints among the loading and unloading tasks (cf. step 1 and 3a of the algorithm). The resource reasoning takes this into account by as-

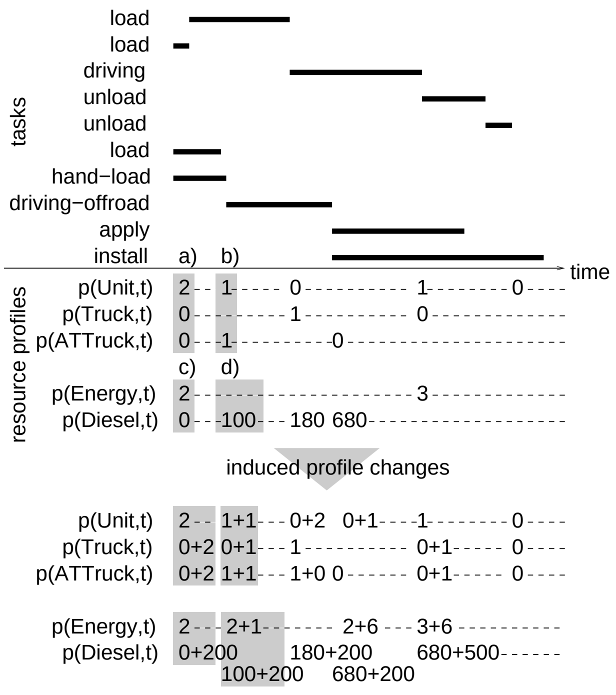


Figure 8: Resource profiles for the example, cf. Figure 7.

suming the allocation of at most two units of *Unit*: two at the beginning of the loading tasks and a third at the beginning of the unloading, which can be re-allocated after the loading procedure releases it. For each loading operation, energy checks are implied, as each of them allocates the symbolic resource *Energy* (cf. Figure 5). In addition, *Diesel* is consumed for the transporting trucks and the power generator, and the trucks itself have to be allocated during driving.

The profile projections for the different resources involved are shown in Figure 8. Their calculation follows straight forward from the task definitions. Please note, that the upper three resources are de-allocated after use, while the others are not, viz. they are *consumed*.

In addition to these explicit profile manipulations, there are implicit manipulations induced according to the abstraction hierarchy. They are reflected in changes of related resource profiles. Figure 8 also shows the changed profiles according to the resource hierarchies. We will focus on the points labeled from *a* to *d*.

The modifications in *a* and *b* are made according to the resource sort hierarchy (*subsumption*). The two *Unit* objects at the first time point could potentially be restricted to the sub-sorts *Truck* or that of the all terrain vehicles *ATTruck* (other sub-sorts are omitted). At the second time point in *b* the system propagates the de-allocation of the abstract resource into the sub-resource, while the allocation of the *ATTruck* unit is reflected in the super-sort.

c describes the propagation of a qualified resource: The abstract loading tasks need in the beginning two *Energy* sorted objects. According to the qualification, each of them can be specialized into 100 liters of diesel fuel (gas and electrical power are omitted). The allocation of another 180 liters of diesel by Trucks implies further *Energy* consumption on the abstract level.

For the next plan modification step, we assume that the capacity for the *Diesel* resource is exceeded, this means, that step 1 of the algorithm detects the over-consumption. The conflict resolution strategies for manipulating the ordering cannot resolve the problem, as there are for example no suitable production tasks in the current plan. A combination of steps 3*b* and *c* can be used to guide the planner into choosing to expand one of the loading tasks into an `ef-load` task, which consumes electric power. The rationale behind this is to restrict the sort of the resource to a non-conflicting sub-sort – as a guideline, the system compares the number of concrete allocations and potential ones, so the other *Energy* sub-sorts qualify for being restricted to.

After that, we assume that an over-allocation occurs on the abstract level for *Electric Fork Lift*, because only one is present at the site and the two loading tasks still not necessarily ask for the same object. The algorithm resolves this conflict by executing step 3*b* (assigning the same resource) and in a later iteration step 3*a* (ordering the steps). We note, that especially in this last conflict resolution the insertion of a producing or de-allocating task could be recommended by the system, as well as performing step *b* after *a*. However, the current strategy of the system always prefers to try to resolve the bottleneck with the tasks given in the current plan, as this often leads to acceptable solutions. This issues will of course be tackled as part of the future work.

Some remarks on the example: Our use of hierarchies in resource reasoning can vary in the same way, the HTN principle is used in our planning engine. If the user has knowledge about the hierarchical relations between the resources she/he can implement it in the methods, sort-hierarchies, decomposition axioms, etc. If not, the algorithms still works with only concrete resource modifications. Another issue is the dimension of planning versus scheduling: If no resource information is given, the system works identically to the hybrid planning approach in (Biundo & Schattenberg 2001). On the other hand, the HTN paradigm allows for an arbitrary initial task network – for example, one which represents a causally complete plan – and no other task or method definitions: the system works then as a pure scheduler.

Related Work

The importance of resources for many practical planning problems like our emergency scenario has led to a variety of approaches that perform resource reasoning during plan generation (Wilkins 1988). In this overview we focus on planning systems and techniques that are motivated in such a context and therefore omit pure planning and pure scheduling systems.

The basic idea underlying most planners is to use resource information for pruning the search space, viz. to rule out plans, which do not allow for a consistent resource allocation. Most existing systems incorporate a constraint-based resource management. Instead of using more generic constraint satisfaction problem solvers, we often find techniques which determine for each specific resource the manipulating operators and subdivide them into consuming and producing actions in order to project potential points of over-consumption of the resource. We find HTN planners like SIPE-2 (Wilkins 1999) that detect necessarily over-consuming plans and SHOP (Nau *et al.* 2001), which allows for resource computations along the task reduction schemas programmed by the modeler. Graph-based systems can be extended, like it has been done with *resource time maps* for IPP (Koehler 1998), or SAT-based approaches (Rintanen & Jungholt 2000), which try to calculate effects of parallel resource manipulation steps.

The members of this first category of systems are keeping book of the consequences of plan generation steps on the balance of availability versus consumption of quantities over time. However, this kind of reasoning does only detect the most necessary backtracking points. It is difficult at least to guide the plan generation process itself. This motivates methodologies, which try to get more information out of the resource analysis and into the planning process.

O-Plan (Tate, Drabble, & Kirby 1994) performs an optimistic and a pessimistic estimation of each resource profile (Drabble & Tate 1994). If the optimistic profile gets below zero, i.e. if all consumption steps are performed as late as possible (Drabble & Kirby 1991) allocating the minimal quantity possible and all production steps are performed as early as possible producing as much as possible, and there is still a point in time in the plan where the capacity is exceeded, then this plan cannot be repaired and search has to backtrack. Furthermore, O-Plan can introduce constraints to evade potentially conflicting plans. Based on this architecture with its constraint managers (Beck & Tate 1995), the TOSCA scheduling system performs an opportunistic search (Beck 1993).

A similar architecture is implemented in (Garrido, Salido, & Barber 2000), where separate constraint-based planning and scheduling modules share a common memory. This system develops several potential solutions in parallel using a strategy of stepwise constraint refinement. Compared to the proposed approach, this non-hierarchical form of planning faces the problem of combinatorial explosion at least in larger applications, where typically many potential solutions exist and numerical resources are involved.

In complete contrast to our approach is that of (Srivastava & Kambhampati 2000), where planning and scheduling for symbolic resources are viewed as two processes that have to be completely separated. Therefore, planning is performed on a relaxed problem level where no resource information is available and the resulting plan is given to a scheduler, which performs the necessary resource allocation. The rationale behind this approach is preparing a causal operator skeleton, that can be filled by a very efficient scheduling module. This framework is defined for planning problems, in which always the plan with the fewest steps is cost-optimal and resource information is not needed to guide the search. In our domain however, there are typically situations, where some goods have to be produced on demand. In these situations, the system falls back to a planner-only configuration.

*parc*PLAN (El-Kholy & Richards 1996) also performs a pre-planning phase: the durations of the plan steps are minimized to determine the necessary overlapping actions and the minimal resource capacities. It introduces meta-variables to represent potential interval overlappings. Their values are manipulated in the planning process according to optimistic and pessimistic estimations: necessary operations are performed because of over-consumptions, more opportunistic ones according to heuristics, which aim at avoiding backtracking. This procedure induces ordering modifications, namely setting and excluding interval overlaps. The approach does not guide the plan generation process by resource demands.

HSTS (Muscettola 1994) is the prototype for a controller of the space telescope *Hubble* and schedules observation sequences on two levels of detail keeping reconfiguration time for the observation devices minimal. Conceptually closely related to it is the mission planning module for the autonomous space probe *Deep Space One* (Muscettola *et al.* 1998). Both systems are constraint-based and generate schedules for parameterized plan fragments, which have to obey larger sets of mission critical constraints. Latest developments in the context of this kind of control software for autonomous spacecraft, the ASPEN platform (Chien *et al.* 2000), follow the paradigm of iterative repair of flawed schedules. The focus of this systems is clearly on scheduling and less on plan generation. However, (Clement *et al.* 2001) builds on it for an HTN planning approach which uses summarized resource information in tasks (for summarized symbolic conditions and their usage in the planning process see (Clement & Durfee 1999)). This form of abstraction recursively deduces bounds for local minima and maxima of resources in abstract tasks from information about resource allocation within more primitive tasks, thereby taking into account possibly overlapping execution of tasks. The methodology can be compared to the presented, hence user specified *approximation* abstraction, so that we believe that this reasoning mechanism can in principle be integrated in the presented approach for cases in which no approximated resource usage is specified. But we note, that the method of summarized information relies on completely specified "classical" task networks in which no additional tasks appear.

In (Castillo, Fdez-Olivares, & González 2000) we find a related approach to hybrid planning which makes use of object aggregations as justifications for expansion schemas (Castillo, Fdez-Olivares, & González 2001). This is in some sense similar to our view on aggregated resources, although we do perform scheduling operations when dealing with this kind of abstraction.

The IxTeT temporal planning system (Laborie & Ghallab 1995a) integrates scheduling by using temporally qualified expressions throughout the representation formalism. They represent state transitions and state persistences of the planning domain. The authors share our view of opportunistic scheduling as additional plan modification steps which can be interleaved with other planning steps: closing open or unachieved preconditions, resolving (resource) conflicts, and adding constraints to evade bottlenecks. The approach is provided with very efficient algorithms. To determine potentially conflicting tasks in a partial plan (known as *min-*

imal critical set computation) (Laborie & Ghallab 1995b) for example, or a least commitment search which is able to quantify the level of commitment in every modification step. Another important feature is the dynamic construction of a resource hierarchy based on condition analysis in the current partial plan (Garcia & Laborie 1996). The hierarchy represents a partial order on the "importance" of the resources for plan causality, and with that the order in which the different resources should be addressed by the reasoning process. This technique will be considered to be integrated into our proposed approach.

Conclusion

We have shown how resource reasoning can be integrated into a hybrid planning approach. On the representational side we identified four possibilities for resource abstraction and motivated their usage in a realistic domain. The least commitment approach of the underlying planning approach allows for merging in scheduling operations for a stepwise plan modification under resource constraints.

The system is part of a larger architecture to give system support for crisis management. Future work includes identifying suitable search strategies, the examination of precompilations out of resource profiles in task networks, and the handling of multi-criteria optimization problems.

References

[Beck & Tate 1995] Beck, H., and Tate, A. 1995. Open planning, scheduling and constraint management architectures. *British Telecommunication's Technical Journal*. Special Issue on Resource Management.

[Beck 1993] Beck, H. 1993. The management of job-shop scheduling constraints in TOSCA. In *NSF Workshop on Intelligent and Dynamic Scheduling for Manufacturing Systems*, 2–14.

[Biundo & Schattenberg 2001] Biundo, S., and Schattenberg, B. 2001. From abstract crisis to concrete relief – a preliminary report on combining state abstraction and HTN planning. In Cesta, A., and Borrajo, D., eds., *Proceedings of the 6th European Conference on Planning (ECP-01)*, Lecture Notes in Computer Science. Toledo, Spain: Springer Verlag, Berlin, Heidelberg, New York, etc.

[Castillo, Fdez-Olivares, & González 2000] Castillo, L.; Fdez-Olivares, J.; and González, A. 2000. A hybrid hierarchical/operator-based planning approach for the design of control programs. In Sauer, J., and Köhler, J., eds., *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-2000)Workshop on New Results in Planning, Scheduling and Design*.

[Castillo, Fdez-Olivares, & González 2001] Castillo, L.; Fdez-Olivares, J.; and González, A. 2001. On the adequacy of hierarchical planning characteristics for real-world problem solving. In Cesta, A., and Borrajo, D., eds., *Proceedings of the 6th European Conference on Planning (ECP-01)*, Lecture Notes in Computer Science. Toledo, Spain: Springer Verlag, Berlin, Heidelberg, New York, etc.

[Chien *et al.* 2000] Chien, S.; Rabideau, G.; Knight, R.; Sherwood, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barrett, T.; Stebbins, G.; and Tran, D. 2000. ASPEN - Automated planning and scheduling for space mission operations. In *6th International Symposium on Space missions Operations and Ground Data Systems (SpaceOps 2000)*.

[Clement & Durfee 1999] Clement, B. J., and Durfee, E. H. 1999. Theory for coordinating concurrent hierarchical planning agents using summary information. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99); Proceedings of the 11th Conference on Innovative Applications of Artificial Intelligence*, 495–502. Menlo Park, Cal.: AAAI/MIT Press.

[Clement *et al.* 2001] Clement, B. J.; Barrett, A. C.; Rabideau, G. R.; and Durfee, E. H. 2001. Using abstraction in planning and scheduling. In Cesta, A., and Borrajo, D., eds., *Proceedings of the 6th European Conference on Planning (ECP-01)*, Lecture Notes in Computer Science. Toledo, Spain: Springer Verlag, Berlin, Heidelberg, New York, etc.

[Drabble & Kirby 1991] Drabble, B., and Kirby, R. 1991. Associating A.I. planner entities with an underlying time point network. In Hertzberg, J., ed., *Proceedings of the 1st European Workshop on AI Planning (EWSP-91)*, volume 522 of *Lecture Notes in Computer Science*, 27–38. St. Augustin, Germany: Springer Verlag, Berlin, Heidelberg, New York, etc.

[Drabble & Tate 1994] Drabble, B., and Tate, A. 1994. The use of optimistic and pessimistic resource profiles to inform search in an activity based planner. In Hammond, K., ed., *Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems (AIPS-94)*, 243–248. University of Chicago, Illinois, USA: AAAI Press, Menlo Park, California, USA.

[El-Kholy & Richards 1996] El-Kholy, A., and Richards, B. 1996. Temporal and resource reasoning in planning: The parcPLAN approach. In Wahlster, W., ed., *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI-96)*, 614–618. Budapest, Hungary: John Wiley & Sons, Chichester, UK.

[Erol, Hendler, & Nau 1994] Erol, K.; Hendler, J.; and Nau, D. S. 1994. UMCP: A sound and complete procedure for hierarchical task network planning. In Hammond, K., ed., *Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems (AIPS-94)*, 88–96. University of Chicago, Illinois, USA: AAAI Press, Menlo Park, California, USA.

[Garcia & Laborie 1996] Garcia, F., and Laborie, P. 1996. Hierarchisation of the seach space in temporal planning. In Ghallab, M., and Milani, A., eds., *New Directions in AI Planning, Proceedings of the 3rd European Workshop on AI Planning (EWSP-95)*, volume 31 of *Frontiers in Artificial Intelligence*, 217–232. Assisi, Italy: IOS Press, Amsterdam, Oxford, Washington DC, Tokyo.

[Garrido, Salido, & Barber 2000] Garrido, A.; Salido, M. A.; and Barber, F. 2000. Scheduling in a planning environment. In Sauer, J., and Köhler, J., eds., *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI-2000)Workshop on New Results in Planning, Scheduling and Design*, 36–43.

[Koehler 1998] Koehler, J. 1998. Planning under resource constraints. In Prade, H., ed., *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, 489–493. Brighton, UK: John Wiley & Sons, Chichester, UK.

[Laborie & Ghallab 1995a] Laborie, P., and Ghallab, M. 1995a. IxTeT: an integrated approach for plan generation and scheduling. In *ETFA-95*, 485–495.

[Laborie & Ghallab 1995b] Laborie, P., and Ghallab, M. 1995b. Planning with sharable resource constraints. In Mellish, C. S., ed., *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1643–1651. Montreal, Cananda: Morgan Kaufmann, San Francisco, CA.

[Muscettola *et al.* 1998] Muscettola, N.; Nayak, P. P.; Pell, B.; and Williams, B. C. 1998. Remote agent: to go boldly where no AI system has gone before. *Artificial Intelligence* 103(1–2):5–47.

[Muscettola 1994] Muscettola, N. 1994. HSTS: Integrating planning and scheduling. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling*. San Mateo, CA: Morgan Kaufmann, San Francisco, CA. 169–212. ISBN 1-55860-260-7.

[Nau *et al.* 2001] Nau, D.; Munoz-Avila, H.; Cao, Y.; Lotem, A.; and Mitchell, S. 2001. Total-order planning with partially ordered subtasks. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*.

[Rintanen & Jungholt 2000] Rintanen, J., and Jungholt, H. 2000. Numeric state variables in constraint-based planning. In Biundo, S., and Fox, M., eds., *Recent Advances in AI Planning, Proceedings of the 5th European Conference on Planning (ECP-99)*, volume 1809 of *Lecture Notes in Computer Science*, 109–121. Durham, United Kingdom: Springer Verlag, Berlin, Heidelberg, New York, etc.

[Srivastava & Kambhampati 2000] Srivastava, B., and Kambhampati, S. 2000. Scaling up planning by teasing out resource scheduling. In Biundo, S., and Fox, M., eds., *Recent Advances in AI Planning, Proceedings of the 5th European Conference on Planning (ECP-99)*, volume 1809 of *Lecture Notes in Computer Science*, 172–186. Durham, United Kingdom: Springer Verlag, Berlin, Heidelberg, New York, etc.

[Stephan & Biundo 1996] Stephan, W., and Biundo, S. 1996. Deduction-based refinement planning. In Drabble, B., ed., *Proceedings of the 3rd International Conference on Artificial Intelligence Planning Systems (AIPS-96)*, 213–220. Edinburgh, Scotland: AAAI Press, Menlo Park, California, USA.

[Tate, Drabble, & Kirby 1994] Tate, A.; Drabble, B.; and Kirby, R. 1994. O-Plan2: An architecture for command, planning and control. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling*. San Mateo, CA: Morgan Kaufmann, San Francisco, CA. ISBN 1-55860-260-7.

[Wilkins 1988] Wilkins, D. E. 1988. *Practical Planning: Extending the AI Planning Paradigm*. San Mateo, California: Morgan Kaufmann. ISBN 0-934613-94-X.

[Wilkins 1999] Wilkins, D. E. 1999. *Using the SIPE-2 Planning System: A Manual for SIPE-2, Version 6.1*. SRI International Artificial Intelligence Center, 333 Ravenswood Ave., Menlo Park, California 94025.