Integrating Planning and Scheduling

A. Garrido, F. Barber

{agarridot, fbarber}@dsic.upv.es

Dpto. Sistemas Informáticos y Computación

Universidad Politécnica de Valencia

Camino de Vera s/n, 46071 Valencia (Spain)

Abbreviated title: Integrating Planning and Scheduling

January 10, 2001

Abstract

Planning and scheduling research is becoming an increasingly interesting topic in the Artificial Intelligence area, because of its immediate application to real problems. Although the last few years have seen dramatic advances in planning systems, they have not seen the same advances in the methods of solving planning and scheduling problems. In this paper, an intuitive way of integrating independent planning and scheduling processes is presented which achieves better performance in the process of solving planning and scheduling problems. The integrated system has the advantage of obtaining a final plan which is executable and optimal. Moreover, the system discards any partial plan as soon as the plan becomes invalid, improving its performance. Our experience demonstrates the validity of this system for tackling planning and scheduling problems.

1 Introduction

For many years, there has been a huge gap between planning and scheduling processes within the Artificial Intelligence area (Smith et al., 2000). Research in planning was focused on problems with multiple levels of action selection, where the only constraints were precedence constraints. Moreover, classical planners assumed a simple discrete model of time that did not take into account shared resources, metric constraints and optimisation criteria in the plan execution. In fact, many planning systems ignored the optimisation of the solution: *'optimisation was generally assumed to be unimportant'* (Smith et al., 2000). Therefore, after the planning stage, a scheduler had to decide which resources to actually allocate. This scheduler was considered as a knowledge-based scheduling process (Dorn and Froeschl, 1993), where the operations accomplishing state transitions were already formulated by the planning process. The scheduler had to reason on actions, resources and time, guaranteeing the feasibility of the planned actions according to the available resources and, it could also optimise these actions according to an evaluation function such as action cost, due time, resource usage, etc. Hence, the final result is the schedule –obtained by the scheduler– of the plan –obtained by the planner–, showing the temporal assignment of actions (steps) and the resources to be used (Sauer, 2000). Consequently, a lot of effort has been put into managing resource scheduling in an efficient way once planning is complete (Bäckström, 1998; Srivastava and Kambhampati, 1999). However, this sequential approach of planning and scheduling is not always adequate for solving real complex problems with a huge number of constraints. Most real problems are overconstrained problems and, therefore, it is necessary to manage a set of soft constraints that should be satisfied but may be relaxed (Sauer, 2000; Tsang, 1993). Although some attempts at integrating planning and scheduling have been carried out (Gervasio and DeJong, 1992; Muscettola, 1994), the most widely used approach is the temporal planning approach such as O-Plan (Currie and Tate, 1991), TOSCA (Beck, 1993) and IxTeT (Ghallab and Laruelle, 1994). In these temporal planning approaches, the gap between planning and scheduling was almost nil: an indistinguishable mixture of planning and scheduling was performed during the solving process. However, temporal planners could only manage the simple constraints which appear in planning environments, i.e., qualitative constraints that define precedence constraints (causal links) among actions of the same plan. These qualitative constraints can represent the nonsimultaneous use of a resource. Hence, if two actions need the same resource, an execution order must be imposed in order to execute one action before the other one. Although the temporal module of a planner can manage these constraints, it is not capable enough to represent all the problem constraints which can appear in planning and scheduling problems. However, many problems that could not be solved by other approaches are now solved by means of the integration of planning and scheduling. For instance, the utility of the integrated system of planning and scheduling is clearly demonstrated with HSTS (Muscettola, 1994). This system unifies both processes in order to solve planning and scheduling tasks for the *Hubble Space Telescope* on space missions. Therefore, a special scheduling module to manage more complex metric constraints becomes indispensable.

In section 2, we examine the problem of planning and scheduling, with its main elements and imposed constraints. In section 3, we present a historical perspective of the alternative approaches used for solving planning and scheduling problems. The three main approaches that have been described are the sequential approach (Srivastava and Kambhampati, 1999, 1999b; Yang, 1997), the temporal planning approach (Currie and Tate, 1991; El-Kholy and Richards, 1996; Ghallab and Laruelle, 1994; Smith and Weld, 1999) and the integrated approach (Garrido et al., 2000; Muscettola, 1994). This section surveys the relationship between planning and scheduling processes, discussing some related work. In section 4, we also provide a model to integrate planning and scheduling, where we explain the structure, the requirements of planning and scheduling processes and the behaviour of the integrated system using a simple example. Although this approach is the most flexible (Long et al., 2000), there is still no definitive approach that is able to integrate the management of all kinds of planning and scheduling problems. The aim of the proposed approach is to take advantage of planning and scheduling processes of any kind in order to discard unfeasible plans while they are being generated and thus improve global performance. The planning system can be based both on POCL (*Partial Order Causal Link*) (Weld, 1994) and graph-based approaches (Blum and Furst, 1997). The scheduling system could also be based on *CSP* (*Constraint Satisfaction Problem*) techniques (Dorn and Froeschl, 1993; Tsang, 1993). However, the most appropriate techniques are the ones based on closure techniques (Barber, 2000; Dechter et al., 1991). The conclusions are presented in section 5.

2 Problem of Planning and Scheduling

A typical planning problem is defined by the triple $\mathcal{PP} = \{\mathcal{I}_s, \mathcal{O}, \mathcal{F}_s\}$, where:

$\mathcal{I}_s$ = Set of initial facts which form the initial situation of the problem.

$\mathcal{O}$ = Set of operators defined in the domain of the problem.

$\mathcal{F}_s$ = Set of final facts which form the final situation of the problem.

The goal of the planning process is to transform the initial facts into the final facts by the proper instantiation of the existing operators. These operators must be instantiated, forming the plan (as a partial or total sequence of actions), with the initial information or with the information deduced by the application of other operators. The typical constraints that appear in a solving process of planning are mainly related to the execution order of the planned actions. This way, the only constraints that appear during the planning process are simple ordering constraints: causal links and the constraints which appear in solving planning conflicts and threats (Weld, 1994). However, all these constraints can be transformed into single *before* or *after* ordering relations which represent easily manageable qualitative constraints.

The planning stage involves a huge selection in a vast domain of possible actions to plan (Smith et al., 2000; Yang, 1997). Consequently, planning systems typically simplify the planning and scheduling problem which implies dealing with unreal situations. For instance, these systems consider actions of the same duration (typically, instantaneous actions), they do not explicitly take into account resource availability, effect persistence, etc. Concretely, dealing with a scheduling problem implies to work with additional elements, which are: a set of constraints over the problem (*hard* and *soft* constraints), a set of available resources, and an evaluation function to assess the quality of each plan in order to discern among several plans. Thereafter, these problems require certain temporal features which impose more constraints to be satisfied and a much more complex planning process. These temporal needs are related to actions, problem states and resources:

- Noninstantaneous duration of actions (Barret et al., 1996; Blum and Furst, 1997; Ghallab et al., 1998): Planning systems assume that all actions have the same duration and they are instantaneous (they are considered atomic and discrete). However, this assumption is not true in real problems because the actions carry out different tasks, which means differing durations (Smith and Weld, 1999). Moreover, depending on the resource to be used, the duration of the actions may change (Beck and Fox, 2000).

- Action effect persistence (Barret et al., 1996; Blum and Furst, 1997): Although planning systems work with infinite persistence (an effect is held until another action removes it), the effects do not always behave this way in real situations. For instance, in the action of boiling some water, the effect of *'hot water'* has finite persistence. In these cases, it becomes necessary to add explicit constraints to represent this kind of persistence.

- Ordering constraints in action abstract contexts (Aylett et al., 2000): Although planners can manage the execution order of the actions (causal links), this kind of order is only qualitative and quite simple. Nevertheless, in more abstract macro-actions, more complex qualitative constraints, such as *overlaps*, *starts*, *finishes*, *equal*, etc., and metric constraints among actions can appear (Dechter et al., 1991).
- Temporal constraints over problem states (Smith and Weld, 1999): Planning systems only deal with obtaining the sequence of actions to achieve the goal. However, they do not consider the instant of time when each action is going to be executed or when a state is achieved. Moreover, constraints over the execution of the entire plan, such as due dates which appear in real problems, are also not considered.
- Shared resource management: Many of the conflicts among actions that appear during the planning process are due to the nonsimultaneous use of shared resources. Since usual planners do not have the necessary knowledge to manage resource availability, they use artificial strategies (such as mutual exclusion relationships, *mutexes* in Graphplan (Blum and Furst, 1997)) to avoid the nonsimultaneous use of each resource. However, explicit resource management becomes necessary to guarantee the allocation of resources (taking into account constraints on their availability and usage) and to allow us to optimise their usage. For instance, it may be necessary to allocate the resources to be used only during the morning because it implies a lower cost or because they are available only during that period of time.

Besides guaranteeing the plan correctness (on the sequence of actions), it is necessary to guarantee its executability (satisfying all the problem constraints and resource availability) and its optimality (optimising due times or costs). Therefore, it is necessary to incorporate a new module into the planners. This module manages all the temporal constraints which appear in a planning and scheduling problem (in an explicit way). This new module can be part of the planning process or can be treated as a separate process (forming the scheduling process). However, our opinion is that the management of these temporal constraints can not be included as part of the planner because implied constraints are usually quite complex. In contrast, we think it is necessary to integrate a specific scheduling module which forms an integrated architecture of planning and scheduling, as we present in section 4.

The most frequent constraints in scheduling are disjunctive constraints (due to mutual exclusions of shared resources), which involve a huge number of disjunctive alternatives. Despite this huge number, only a few alternatives commonly involve a feasible order. Moreover, the scheduling problem must deal with the nonsimultaneous allocation of limited shared resources to a set of actions through time. All the imposed constraints (both from planning and scheduling) can be managed in a similar way by means of qualitative and metric (quantitative) disjunctive constraints (Dechter et al., 1991). Qualitative constraints impose a relative order of execution between two temporal objects, which can represent time points such as the beginning or ending of actions, duration intervals, etc. In contrast, metric constraints fix an absolute bound of the temporal distance between temporal objects.

Thus, the scheduling problem can be formulated as a *TCSP* (*Temporal Constraint Satisfaction Problem*) (Dechter et al., 1991) using a constraint-based representation that implies the twofold additional problems (Tsang, 1995):

- Feasibility problem: The final solution must accomplish all the problem constraints. In addition, another objective to achieve is the optimisation of some evaluation function according to several criteria such as cost, due time, etc.

- Overconstrained problem: Some planning and scheduling problems work with a vast number of constraints. For instance, due to the fact that the available resources are limited, the task of determining an execution order may be a very complex task. Therefore, it is necessary to guarantee only a sub-set of the whole set of problem disjunctive constraints.

Once all the problem constraints that characterise the planning and scheduling problem are known, they must be modelled by a definition language. Although many domain definition languages have appeared in the literature, one of the most currently used is *PDDL* (*Planning Domain Definition Language* (Ghallab et al., 1998)). This UCPOP-based language supports STRIPS-based operators, hierarchical actions, conditional effects, universal quantifiers and many other features, which unfortunately are not commonly used in real planners. However, this language has some drawbacks which are related to the typical elements used in the most common scheduling problems. Consequently, it is necessary to extend this language in order to be able to define some temporal features such as action durations, resources, effect persistence and metric constraints on resources, on actions and on the entire plan.

3 Relation between Planning and Scheduling Processes. Related Work

In this section, we study the relation between planning and scheduling processes. Historically, the first two approaches were the sequential approach of planning and scheduling and the temporal planning approach. Currently, there are many research lines interested in the integration of planning and scheduling in order to allow us to solve real problems in a more efficient way. Hence, the last approach presented is the integration of planning and scheduling, which is detailed in next section.

3.1 Sequential Approach of Planning and Scheduling

This approach has been widely used due to its simplicity because *'it is much clearer conceptually to draw a line between planning and scheduling; planning precedes scheduling'* (Yang, 1997). In this approach, the two processes are independent with no relation between them during their execution (*scheduling as a post-planning phase* (Srivastava and Kambhampati, 1999, 1999b)). Hence, the solving process is divided into the two following stages:

- First stage: *planning*. The planner obtains the plan by the application of the operators defined in the domain. The only constraints present at this stage are the precedence constraints.
- Second stage: *scheduling*. After planning is complete, the scheduler allocates the needed resources for the entire plan and validates all the problem constraints, both implicit (nonsimultaneous resource use) and explicit (action durations, effect persistence and due times).

This way, the plan obtained by the planner must be instantiated in time. The scheduler checks the feasibility of each entire plan (with the overload that it supposes). This is done so that the plan which is obtained after the two previous stages is executable. The main advantage of this approach is the possibility of obtaining an abstract plan, which is independent of the available resources and can be reusable. This abstract plan will be instantiated into several situations, over the same domain, according to the available resources for each situation (Srivastava and Kambhampati, 1999b).

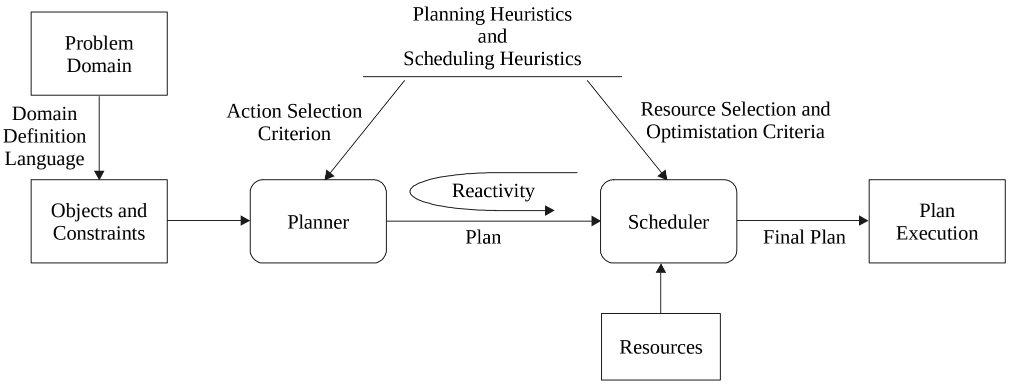
Problem Domain
Domain Definition Language
Objects and Constraints
Planning Heuristics and Scheduling Heuristics
Action Selection Criterion
Resource Selection and Optimistation Criteria
Planner
Reactivity
Plan
Scheduler
Final Plan
Plan Execution
Resources

Figure 1: Sequential approach of planning and scheduling

Figure 1 shows the problem domain which defines the objects and problem constraints by means of the domain definition language. Then, the planner and the scheduler processes obtain the final plan which will be executed at the execution stage. Nonintegrated heuristics for planning and scheduling are used to select the action to plan and the resource to schedule respectively (Sadeh and Fox, 1996), according to separate optimisation criteria. A *reactivity* stage occurs when the scheduler detects any inconsistency in the plan obtained by the planner. This reactivity stage will repair the schedule of the plan trying to maintain a correct feasible plan over time (Sauer, 2000; Smith, 1995). Hence, the twofold disadvantages of this approach which make unfeasible for real complex problems are:

- Lack of a global optimisation criterion: The two processes work in an independent way. For this reason, each process does not know the behaviour of the other process and they do not cooperate to obtain a better final plan. Moreover, they do not use a global heuristic criterion in order to obtain an optimal plan.
- Overload on planning and scheduling processes: The plan obtained by the planner may not be

feasible. Since no problem constraints are checked during the plan generation, the plan may be unfeasible because it violates some constraint (for instance, there are not enough resources). Hence, a new stage of planning (planner-scheduler reactivity in Figure 1) must be carried out discarding the current plan. In consequence, it might be necessary to successively perform several planning and scheduling processes to obtain a feasible plan.

3.2 Temporal Planning Approach

This approach arose as an attempt to include a simple temporal reasoning module, which performs the scheduling process, in the planning process. Planners of this kind are called temporal planners. For instance, O-Plan, IxTeT, parcPLAN (El-Kholy and Richards, 1996) and Temporal Graphplan (Smith and Weld, 1999). Temporal planners are able to deal with temporal information and/or to reason on resources. They usually manage temporal information by means of an explicit representation and by using *TPN* (*Time Point Network*) and *TCN* (*Temporal Constraint Network*) algorithms to represent time constraints on time points. Thus, these planners can manage qualitative and metric constraints.

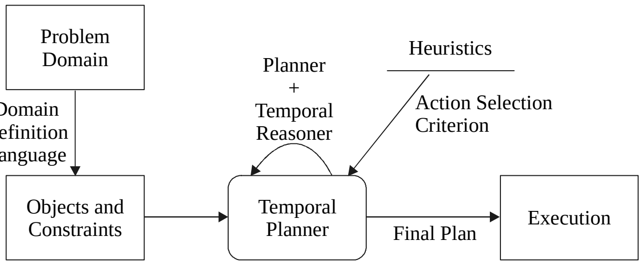

Figure 2: Temporal planning approach

The temporal planning approach is shown in Figure 2. The temporal planner is formed by a traditional planner plus a temporal constraint reasoner that performs the temporal reasoning. The temporal reasoner instantiates actions in time from causal relations and allocates the resources according to their use and availability. The heuristics are mainly used to select the actions. The optimisation criterion is generally assumed to be unimportant (simply finding a plan is good enough) and the temporal reasoner does not optimise the plan. Therefore, although the final plan obtained by the temporal planner is an executable plan, it may not be optimal because neither cost nor optimisation criteria have been used. This approach has two important drawbacks:

- The temporal reasoning of the temporal planner is adequate but limited. The management of complex temporal constraints (mainly the disjunctive ones) on plans, actions and shared resources is a very difficult task because there is no specific time manager. Furthermore, temporal planners do not take into account the temporal knowledge provided by a specific scheduling process. Consequently, it is difficult to perform optimisation tasks on the plan. This way, the final plan will be executable but it might not be optimal nor efficient.

- It becomes difficult to determine when the system is planning or scheduling. For instance, *'it is easy to see that O-Plan works, but it is difficult to see why'* (Bäckström, 1998). Thus, the whole process is a mixture of planning and scheduling and, therefore, it is much more difficult to define common heuristic criteria to improve the performance of the system.

3.3 Integrated Approach of Planning and Scheduling

This approach integrates a process of planning and scheduling to avoid the drawbacks of the previous approaches. This approach constitutes one of the most active research topics, mainly due to its flexibility in real problems, as proposed in (Gervasio and DeJong, 1992; Muscettola, 1994). Although this approach is more widely studied in the next section, briefly, here are its main features:

- Planning and scheduling processes cooperate as partial stages during the problem solving process. Hence, the integrated system can take advantage of both processes, overlapping the planning of actions with their allocation in time (by the scheduling process). This way, the system performance is improved because it only manages feasible plans excluding any unfeasible plan from the search space (Garrido et al., 2000).
- Since the integrated system is formed by a planner and a scheduler, we can apply traditional heuristic criteria to each process to improve its performance and the optimality of the final plan and schedule (Sadeh and Fox, 1996). For instance, techniques to diminish the search space in the planner, and more efficient criteria to perform a better optimisation of the obtained schedule in the scheduler. Furthermore, new optimisation criteria can be applied to improve the performance of the integrated system which takes into account the criteria of both processes.

In the next section, the integrated approach of planning and scheduling, including its architecture, structure and behaviour, is presented in a more detailed way.

4 Architecture of the Integrated System

As we have seen above, the main aim of an integrated system of planning and scheduling is to guarantee its executability during the plan generation, satisfying all the problem constraints in an optimal way. In order to delve deeper into the integrated system, we will study its main features and the necessary requirements for the planning and scheduling processes in this section.

4.1 Structure of the Integrated System

The structure of the integrated system is shown in Figure 3. In this system, the planner tackles the necessary actions to achieve the goal whereas the scheduler deals with the validation of each constraint imposed by the planned actions. In addition, the planner and the scheduler can take advantage of common heuristics such as the action selection and the selection of the most suitable resources. Hence, when the solving process terminates, an executable optimal plan is obtained.

The integrated system (see Figure 3) is formed by a control module that manages the behaviour of the planner and scheduler. Besides managing the interactivity of the solving process, the control module manages the global data and the common heuristics and optimisation criteria. The planning and scheduling processes share data structures with the common information (problem objects, constraints, actions, resources, etc.) in a special kind of data structure which is similar to a *blackboard* model. On the one hand, the planner must access each piece of data which is related to actions, their order, initial situation and goals. On the other hand, the scheduler must keep all the information which is related to allocation and nonsimultaneous resource usage, temporal constraints and metric order among actions. In addition to this common information, both processes have a local memory to store the information that only they need, such as local assignments, internal structures, etc.

4.2 Behaviour of the Integrated System

The solving process of planning and scheduling starts after defining all the problem objects. The control module manages the interaction between the planner and the scheduler (Figure 4) asking the planner for a set of actions (forming a partial plan) that satisfies some of the problem goals. The planner indicates what the resource necessity for the planned actions is and the new action orders which are established. When the control module has the partial plans provided by the planner, the

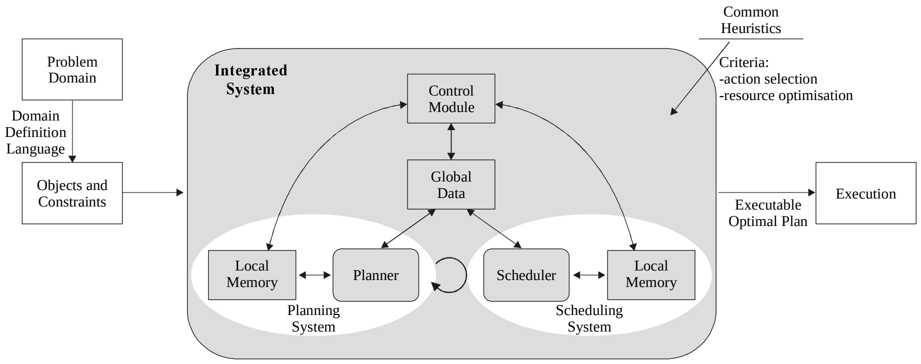

Figure 3: Structure of the integrated system of planning and scheduling

control module selects one to be validated by the scheduler. Thus, the scheduler carries out the resource allocation and the constraint checking. These resources may be defined in the action or determined by another action which has already been planned (for instance, that action must be executed on a particular resource because it belongs to a macro-action which uses that resource). If the scheduler carries out the resource allocation, it will allocate the resources according to the resource availability, the previous constraints and the optimisation criteria.

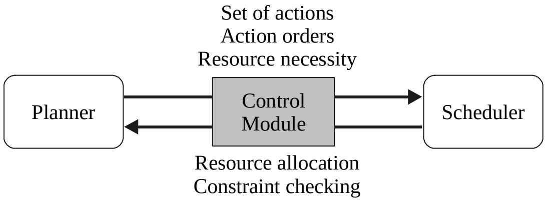

Figure 4: Interaction between the planner and the scheduler managed by the control module

A strong communication between the two processes is performed during the plan generation by means of the shared data structures as illustrated in Figures 3 and 4. Communication between the planner and the scheduler must occur when:

- A new action (or an existing action in a POCL approach) is planned to solve a precondition. In this case, the scheduler must update the order among the actions according to the new causal links. Moreover, the scheduler must allocate the needed resources (according to resource optimisation criteria) and update the ordering constraints among actions which use the same resources.
- A new order among actions is established due to the resolution of a planning conflict (*threats* in POCL approaches). As in the previous case, the scheduler must update the order among the actions.

In these cases, the planner informs the scheduler about the action to be planned and in which alternative planning context it has been planned. Obviously, resources in actions of a same planning context must not be simultaneously used.

Once the behaviour of the integrated system has been introduced, we will show how both processes must behave. Since the two processes have traditionally been executed in a separate way, it becomes necessary to include several requirements in both processes (mainly to the scheduling system) to adapt their behaviours for the simultaneous integrated execution.

4.2.1 Requirements for the Planning System

Traditional planners obtain the plan by including actions and relating them according to precedence relations established in causal links. However, instead of assembling these actions to build the plan, it is more appropriate to obtain the plan in a hierarchical way. This way, the planner obtains a high-level objective that will be broken down into simpler actions. Hence, the planning process is divided into two levels. The former deals with an abstract plan, which is formed by a sequence of partially ordered macro-actions. Then, the latter refines this abstract plan. Although the best planning technique for an integrated system would be *HTN* (*Hierarchical Transition Network* (Nau et al., 1995)), other planning approaches based on POCL or graph-based planning would also be possible. Macro-actions indicate which actions should be used to obtain a specific goal. For this reason, it is important for the domain definition language to be able to define macro-actions or hierarchical actions such as *PDDL*.

The planner should be able to manage several plans due to the interactivity of the solving process in an integrated system. Each alternative plan represents a different planning context that can solve the current problem in an alternative way. Currently, planners based on POCL such as the traditional UCPOP and graph-based planners such as the popular Graphplan can do it. Both approaches offer the possibility of maintaining several alternative plans in their respective search space or planning graph.

4.2.2 Requirements for the Scheduling System

Traditionally, scheduling systems have been based on different techniques, mainly in the areas of Operational Research and Artificial Intelligence (Dorn and Froeschl, 1993; Sauer, 2000; Smith, 1992): Knowledge-Based Systems, Evolutionary Programming and genetic algorithms, Agent Models, Neural Networks, etc. In addition, in the Artificial Intelligence area, many schedulers are also based on Constraint Programming and *CSP* (*Constraint Satisfaction Problem*) techniques (Kumar, 1992; Wallace, 1996). In these schedulers, the entire set of problem constraints is known in advance, so the aim of the schedulers is to obtain a solution which satisfies the temporal constraint satisfaction problem. However, if the set of constraints is modified (when new actions are planned), the previous solution may become invalid. Hence, much computational effort is spent on obtaining a solution which can be useless after asserting new constraints. Although it is clear that incremental *CSP* methods might be used here (Tsang, 1993), the system does not need the solution to the problem in each step: it only needs to assure the consistency of all the constraints in each current partial plan. Therefore, the most appropriate techniques for solving *TCSPs* of this kind are the closure techniques, which consist of propagating a temporal constraint network in order to guarantee different levels of consistency (Dechter et al., 1991; Dechter, 1992). Hence, the closure process propagates each new constraint asserted into the system and determines whether the constraint network is still consistent (i.e., a feasible plan, which satisfies all the problem and resource usage constraints). This indicates the utility of the exploitation of constraint propagation techniques in scheduling systems, which involve combinatorial problems to carry out resource allocation (Beck and Fox, 2000; Wallace, 1996).

In general, the requirements for a scheduling process in an integrated system are the following:

- The scheduler must behave incrementally. An integrated system of planning and scheduling is a dynamic system because constraints are progressively known during the generation of each partial plan. As the planner plans actions, it provides new precedence and resource usage constraints to the scheduler which validates them. This is the main property that makes the closure techniques the most appropriate techniques because they behave in a dynamic way –*dynamic* or *interactive scheduling* (Garrido et al., 2000; Sauer, 2000).

- The scheduler must manage constraints from several planning contexts –*contextual scheduling* as in (Garrido et al., 2000)– because the planner can manage several plans. Each planning context has constraints that must not interfere with other planning context constraints. Thus, the scheduler will validate the constraints taking into account the plan they belong to.

- Some evaluation function to assess the quality of each plan must be used. Thus, the scheduler should be able to manage several optimisation criteria such as cost, execution times, deadlines, which are used in real scenarios (Aylett et al., 2000; Wallace, 1996).

4.2.3 Functionality Through an Example

We will introduce a simple example to explain the functionality of the integrated system. We will use as example a modified version of the well-known ferry problem (Barret et al., 1996). Although the example is very simple, it is complex enough to reveal the main deficiencies of the sequential approach of planning and scheduling and the temporal planning approach studied in section 3.

The objective of the problem is to transport two vehicles (*V1* and *V2*) from one side of a river to the other one by using a ferry (see Figure 5). However, we have added a lorry that can also transport the vehicles. The resources are the ferry, the lorry and a new drawbridge, which has been added to the problem and crosses the river. The drawbridge must be up in order for the ferry to sail under it, and it must be down in order for the lorry to drive over it. Thus, the drawbridge is a shared resource which must be nonsimultaneously *used* by both the ferry and the lorry.

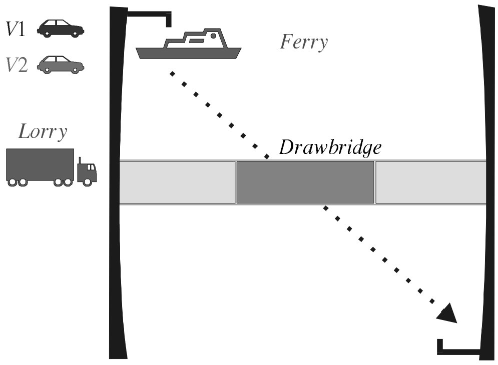

Figure 5: Ferry example. The elements are the ferry, the lorry, the drawbridge and the two vehicles

The domain actions are defined in Figure 6 in accordance with the extended *PDDL* language used in our approach. The actions have temporal and optimisation features such as their duration and cost, respectively, besides their parameters, preconditions and effects. Therefore, this example contains all the features that can appear in a real planning and scheduling problem.

As can be seen in Figure 6, the actions `load-ferry`, `unload-ferry`, `load-lorry` and `unload-lorry` have a duration and cost which are so small that they are not taken into account. Hence, the only difference (concerning duration and cost) between transporting each vehicle by ferry or lorry happens in actions `sail-ferry` and `drive-lorry`. The duration of the action `sail-ferry` is 60 and its cost is 20. The duration of the action `drive-lorry` is 20 and its cost is 50. The resource nonsimultaneously shared by two actions is the drawbridge. Moreover, let us suppose there exists a temporal constraint over the plan's deadline which fixes the deadline at 75. Hence, the plan must be executed before 75 on the time line. Obviously, the four alternative plans that a planner could obtain are:

- Both vehicles *V1* and *V2* are transported by ferry (*plan 1*). The necessary time for this plan is 180 ($60*3$) and its cost is 60 ($20*3$).
- *V1* is transported by ferry and *V2* is transported by lorry (*plan 2*). The necessary time for this plan is 80 (60+20) and its cost is 70 (20+50). The same time and cost is for the plan which transports *V1* on lorry and *V2* on ferry (*plan 3*).
- Both vehicles *V1* and *V2* are transported by lorry (*plan 4*). The necessary time for this plan is 60 ($20*3$) and its cost is 150 ($50*3$).

Although *plan 4* is the highest-priced plan, it is the only feasible one because of the deadline. Here, we can see the main drawbacks of the sequential and temporal planning approaches seen in section 3. For instance, the planning and scheduling stages should be repeated in the sequential approach until finding *plan 4*, and no scheduling heuristics to optimise each partial plan would be

```
(:action load-ferry
  :parameters (?ferry ?vehicle ?place)
  :precondition (and (tferry ?ferry)
                     (tvehicle ?vehicle)
                     (tplace ?place)
                     (free ?ferry)
                     (at-ferry ?place)
                     (at-vehicle ?place))
  :effect (and (on-vehicle ?vehicle ?ferry)
               (not (at-vehicle ?place))
               (not (free ?ferry)))
  :resources (?ferry)
  :duration 0
  :cost 0)
(:action unload-ferry
  :parameters (?ferry ?vehicle ?place)
  :precondition (and (tferry ?ferry)
                     (tvehicle ?vehicle)
                     (tplace ?place)
                     (at-ferry ?place))
  :effect (and (at-vehicle ?vehicle ?place)
               (free ?ferry)
               (not (on-vehicle ?vehicle ?ferry)))
  :resources (?ferry)
  :duration 0
  :cost 0)
(:action sail-ferry
  :parameters (?ferry ?vehicle ?from ?to)
  :precondition (and (tferry ?ferry)
                     (tvehicle ?vehicle)
                     (tplace ?from)
                     (tplace ?to)
                     (at-ferry ?from))
  :effect (and (at-ferry ?to)
               (not (at-ferry ?from)))
  :resources (?ferry bridge)
  :duration 60
  :cost 20)
```

```
(:action load-lorry
  :parameters (?lorry ?vehicle ?place)
  :precondition (and (tlorry ?lorry)
                     (tvehicle ?vehicle)
                     (tplace ?place)
                     (free ?lorry)
                     (at-lorry ?place)
                     (at-vehicle ?place))
  :effect (and (on-vehicle ?vehicle ?lorry)
               (not (at-vehicle ?place))
               (not (free ?lorry)))
  :resources (?lorry)
  :duration 0
  :cost 0)
(:action unload-lorry
  :parameters (?lorry ?vehicle ?place)
  :precondition (and (tlorry ?lorry)
                     (tvehicle ?vehicle)
                     (tplace ?place)
                     (at-lorry ?place))
  :effect (and (at-vehicle ?vehicle ?place)
               (free ?lorry)
               (not (on-vehicle ?vehicle ?lorry)))
  :resources (?lorry)
  :duration 0
  :cost 0)
(:action drive-lorry
  :parameters (?lorry ?vehicle ?from ?to)
  :precondition (and (tlorry ?lorry)
                     (tvehicle ?vehicle)
                     (tplace ?from)
                     (tplace ?to)
                     (at-lorry ?from))
  :effect (and (at-lorry ?to)
               (not (at-lorry ?from)))
  :resources (?lorry bridge)
  :duration 20
  :cost 50)
```

Figure 6: Definition of the domain actions

used in the temporal planning approach.

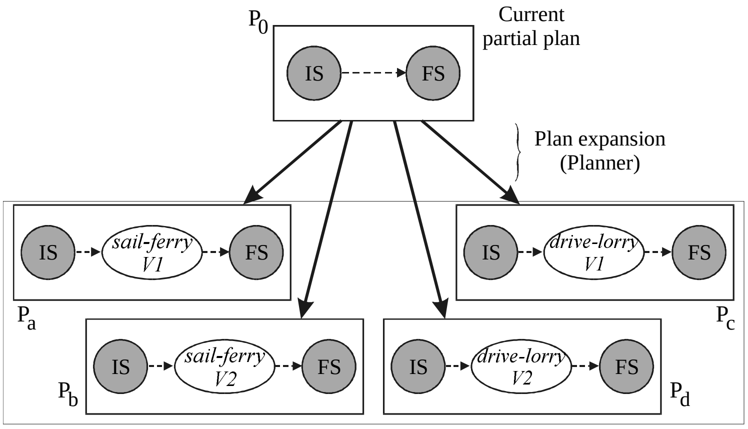

Figure 7: Plan expansion starting from the current partial plan P_0

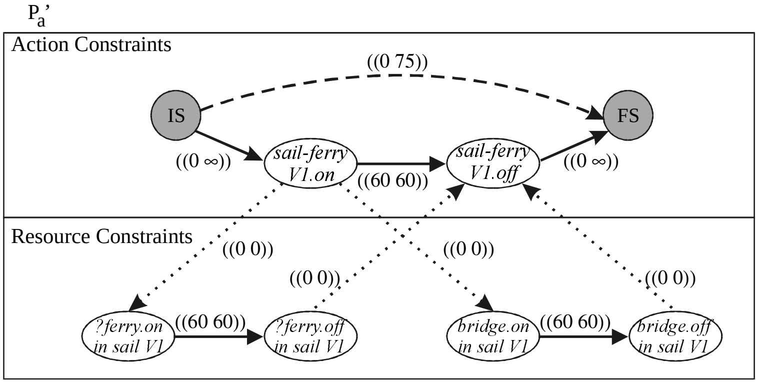

Figure 8: Resource allocation and constraint checking of the plan P_a (P_a' generation)

Now, we will see how this example is solved by the integrated system. Let P_0 be a current partial plan. In Figure 7, the plan P_0 is only formed by two fictitious actions which represent the initial situation (`IS`) and the final situation (`FS`). Figure 7 shows us this initial plan and the plan expansion carried out by the planner. Starting from P_0, the rest of partial plans which will achieve the problem goals are expanded in the following way:

1. The planner informs the control module about the alternative partial plans (P_a, P_b, P_c and P_d) that add new actions (`sail-ferry V1`, `sail-ferry V2`, `drive-lorry V1` and `drive-lorry V2`, respectively) to achieve the goal; all the permutations to transport *V1* and *V2* (Figure 7).
2. The control module selects the partial plan with which to continue calculating the most favourable partial plan (according to integrated optimisation criteria) to be expanded next. In our example, the plan P_a is selected by the control module because it is the lowest-priced partial plan.
3. The scheduler validates the temporal constraints and allocates the resources (only if necessary) of the plan selected by the control module (P_a) as can be seen in Figure 8. In our example, the selected plan adds a new action (`sail-ferry V1`) which imposes new constraints to validate (utilisation of the ferry and the drawbridge) in the temporal constraint network. Figure 8 shows how the scheduler manages the constraints among actions and resources.
4. After validating the plan (P_a), a new partial plan which satisfies all the constraints is obtained (P_a'). The new plan (P_a') may be more constrained because the scheduler carries out a more precise reasoning on time and resources than the planner does (Figure 8).
5. Although the obtained plan (P_a') is still an incomplete plan, it is an executable and (eventually) optimal plan because all the constraints have been validated by the scheduler. This way, the system only manages partial plans which are feasible.
6. Then, the control module asks the planner for new plans that achieve other problem goals. For instance, successor plans of P_a' are P_{a1}' and P_{a2}' (see Figure 9). Next, the control module

again selects the plan to be validated by the scheduler. In this example, if the optimisation criterion used is the cost criterion, the selected plan will be P_{a1}'. Since actions `sail-ferry V1` and `sail-ferry V2` use the same ferry and drawbridge in a nonsimultaneous way, the scheduler will inform that the plan P_{a1}' is unfeasible because the deadline (temporal constraint *'((0 75))'* in Figure 8) has been violated. Clearly, it is impossible to sequentially execute two 60-duration actions in a 75-duration period. Therefore, this partial plan is immediately discarded and another alternative plan must be found. However, since the alternative plan P_{a2}' is also unfeasible, the control module must select another of the alternative plans found in Figure 7. Hence, the process is repeated until the control module detects (through the planner) that all the problem goals have been satisfied.

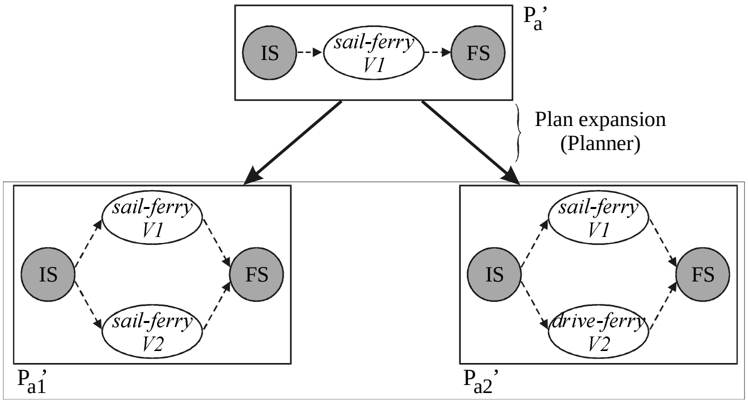

Figure 9: Expansion of successor plans of P_a'

As we have noted in the previous execution of the integrated system, the control module activates both the processes of planning and scheduling, by selecting the partial plan to be processed.

In this example, we have demonstrated the improvements that the integrated approach can obtain. Although the proposed approach is valid in more complex problems, we have chosen this simple one in order to make the execution shown above easier.

To date, our work on the implementation of this integrated approach has been mainly focused on the validation of the presented integration, more than on obtaining an efficient implementation with code optimisation. We have verified the correctness of the system and the final plan through the cooperative behaviour of the planner and the scheduler. Based on our experience, the system presented here seems to be valid for dealing with real problems of planning and scheduling. Comparison between our integrated approach and other planning approaches is quite difficult because currently there are no similar approaches. Planning and scheduling approaches solve different problems in different ways; some deal only with actions, others deal with actions and time, and our approach deals with actions, time and resources. For instance, the sequential approach performs the solving process in a completely different way, whereas the temporal planning approach is not able to manage the same constraints that are managed by the integrated approach.

5 Discussion and Conclusions

Traditionally, planning technology has had an important impact in solving real problems, mainly because of the successful application of special-purpose algorithms, such as schedulers (Dorn and Froeschl, 1993; Smith, 1992; Wallace, 1996). Currently, dealing with planning and scheduling systems is a topic of great interest due to their immediate application to real problems. The integrated approach here presented could be successfully applied in multiple real scenarios, as proposed in (Wallace, 1996), which would take advantage of it. Nowadays, we are working on the problem which imposes the process of cutting stock and stock management (Onaindía et al., 1998). In that problem, there exists a huge number of resources (cranes, lorries, raw products, staff, etc.) and many actions to plan over them. Moreover, we attempt to tackle a logistics problem in a transport company (Garrido and Onaindía, 1999). This problem imposes a set of constraints to satisfy (transport the customers' goods, shipping due times, constraints on the capacity of the vehicles and the branches, etc.) on a set of limited resources (fleet of vehicles, branches, staff, etc.). The evaluation function assesses the economical cost of each plan. Hence, the objective is to obtain a feasible plan of delivery routes and vehicles which satisfies the customer demands. This problem imposes a huge number of alternative actions to plan and of resources to be used. Therefore, an integrated approach with enough knowledge on tasks of planning and scheduling is necessary. In addition, above problems present an important challenge in our approach: reactivity (Sauer, 2000; Smith, 1995). The problem of creating an optimal plan is rather a predictive problem in a deterministic and stable environment. However, during the execution of the plan it is indispensable to take into account a reactive component, because unexpected external events may occur. For instance, a vehicle or itinerary may become unavailable, which may become invalid the obtained plan of delivery routes. In that case, our system must repair the plan according to the new constraints imposed in the problem. Clearly, the system will try to conserve as much as possible of the obtained plan in order to find an alternative plan almost immediately.

This paper has presented an integrated approach for planning and scheduling, which is able to deal with planning and scheduling problems in a more flexible and efficient way than classical approaches. We have detailed the structure of the integrated system, mainly with the control module and global data, which is shared by two processes during the solving process.

We have also presented the behaviour of the integrated system, comparing it to the rest of approaches. The integrated approach presented here works in an incremental way, where the validation stage of the plan is carried out while it is being generated. It implies new requirements, mainly for the scheduling process. The scheduling process should only guarantee the consistency of each partial plan. Therefore, closure techniques are more adequate than traditional *CSP* techniques because they accomplish two of the main requirements for the scheduling process; *incremental* and *contextual scheduling*. Closure techniques can incrementally guarantee several levels of consistency of a temporal constraint network. Each time the planner plans an action in a planning context, its corresponding constraints are verified by the scheduler. Moreover, since the scheduler deals with the constraints imposed during the planning stage, it is necessary to manage the temporal constraints which are related to actions, resources and problem states.

The most interesting contributions of this paper are the advantages that can be obtained as a result of the application of the integrated system. Basically, they are:

- Any conflict, inconsistency or constraint violation in a partial plan is detected as soon as it appears. Since this incident implies a nonfeasible plan, this plan is immediately discarded. Therefore, only the possible feasible plans are processed during the search process, which improves the performance of the solving process.
- The system can manage more complex constraints than temporal planners because the integrated system uses a specific scheduling process. Moreover, many heuristics and optimisation criteria (on the planner and the scheduler) can be defined at different levels. On one hand, many traditional heuristics are applicable to the separate processes of planning and scheduling. On the other hand, some new heuristics can be applied to the integrated system and to the control module. These criteria improve the efficiency of the solving process and the quality (optimality) of the final plan.
- The final plan is feasible, executable and optimal according to how the partial plans are

generated. The scheduler validates the feasibility of the plan after each planning stage. This way, each alternative plan of the search space is executable because it satisfies all the problem constraints and resource availability.

In spite of the performance improvements that can be obtained with the integrated system, the complexity of planning and scheduling processes still remains very high. Therefore, it is not irrelevant in an integrated approach to study both processes in a separate way in order to improve their performance and the performance of the integrated system (Bäckström, 1998).

References

Aylett, R.S., G.J. Petley, P.W.H. Chung, B. Chen, and D.W. Edwards. 2000. AI planning: Solutions for real world problems. *Knowledge-Based Systems*, 13:61-69.

Bäckström, C. 1998. Computational aspects of reordering plans. *Journal of Artificial Intelligence Research*, 9:99-137.

Barber, F. 2000. Reasoning on complex disjunctive temporal constraints. *Journal of Artificial Intelligence Research*, 12:35-86.

Barret, A., D. Christianson, M. Friedman, K. Golden, S Penberthy, Y. Sun, and D. Weld. 1996. UCPOP V4.0 User's manual. Repport technique TR 93-09-06d, Dept. of Computer Science and Engineering, University of Washington, Seattle, WA.

Beck, H. 1993. TOSCA: A novel approach to the management of job-shop scheduling constraints. In *Proc. 9th CIM-Europe Annual Conference. Realising CIM's industrial potential*, 138-149.

Beck, J.C., and M.S. Fox. 2000. Constraint-directed techniques for scheduling alternative activities. *Artificial Intelligence*, 121:211-250.

Blum, A.L., and M.L. Furst. 1997. Fast planning through planning graph analysis. *Artificial Intelligence*, 90:281-300.

Currie, K., and A. Tate. 1991. O-Plan: The open planning architecture. *Artificial Intelligence*, 52(1):49-86.

Dechter, R., I. Meiri, and J. Pearl. 1991. Temporal constraint networks. *Artificial Intelligence*, 49:61-95.

Dechter, R. 1992. From local to global consistency. *Artificial Intelligence*, 55:87-107.

Dorn, J., and K. Froeschl. 1993. *Scheduling of production processes.* Ellis Horwood (Series in Articial Intelligence).

El-Kholy, A., and B. Richards. 1996. Temporal and resource reasoning in planning: the parcPLAN approach. In *Proc. 12th European Conference on Artificial Intelligence (ECAI-96)*, 614-618.

Garrido, A., and E. Onaindía. 1999. Un Algoritmo para la Optimización de Rutas de Transporte. In *Proc. 8th Conferencia Española para la Inteligencia Artificial*, 2(4):1-8.

Garrido, A., M.A. Salido, and F. Barber. 2000. Scheduling in a planning environment. In *Proc. 14th Workshop on New Results in Planning, Scheduling and Design of ECAI-2000*, 36-43.

Gervasio, M., and G. DeJong. 1992. A completable approach to integrating planning and scheduling. In Morgan Kaufmann, editeur, *Proc. 1st Int. Conf. (AIPS-92). Artificial Intelligence Planning Systems*, 275-276.

Ghallab, M., and H. Laruelle. 1994. Representation and control in IxTeT, a temporal planner. In *Proc. 2nd Int. Conf. on AI Planning Systems*, 61-67.

Ghallab, M., A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins. 1998. PDDL - The planning domain denition language. AIPS-98 planning competition committee: ftp://ftp.cs.yale.edu/pub/mcdermott/software/pddl.tar.gz.

Kumar, V. 1992. Algorithms for constraint satisfaction problems: A survey. *AI Magazine*, 13(1):32-44.

Long, D., M. Fox, L. Sebastia, and A. Coddington. 2000. An examination of resources in planning. In *Proc. Workshop of the UK Planning and Scheduling Special Interest Group.*

Muscettola, N. 1994. HSTS: Integrating planning and scheduling. In M. Zweben and M. Fox, editeurs, *Intelligent Scheduling*, 169-212. Morgan Kaufmann, San Mateo, CA.

Nau, D., S. Gupta, and W. Regli. 1995. Artificial intelligence planning versus manufacturing-operation planning: A case study. In *Proc. 14th Int. Joint Conf. on AI*, 1670-1676.

Onaindía, E., F. Barber, V. Botti, C. Carrascosa, M.A. Hernández, and M. Rebollo. 1998. A progressive heuristic search algorithm for the cutting stock problem. *Lecture Notes in Artificial Intelligence (1416)* edited by Springer-Verlag, 25-35.

Sadeh, N.M., and M.S. Fox. 1996. Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. *Artificial Intelligence*, 86:1-41.

Sauer, J. 2000. Knowledge-based systems techniques and applications in scheduling. In *Knowledge-based systems techniques and applications*, 4:1293-1325.

Smith, S.F. 1992. Knowledge-based production management: Approaches, results, and prospects. *Production Planning and Control*, 3(4).

Smith, S.F. 1995. *Reactive scheduling systems.* Kluwer Publishing.

Smith, D.E., and D.S. Weld. 1999. Temporal planning with mutual exclusion reasoning. In *Proc. 16th Int. Joint Conf. on AI (IJCAI-99).*

Smith, D.E., J. Frank, and A.K. Jónsson. 2000. Bridging the gap between planning and scheduling. *Knowledge Engineering Review*, 15(1).

Srivastava, B., and S. Kambhampati. 1999. Efficient planning through separate resource scheduling. In *Proc. AAAI Spring Symp. on Search Strategy under Uncertain and Incomplete Information.*

Srivastava, B., and S. Kambhampati. 1999b. Scaling up planning by teasing out resource scheduling. In *Proc. European Conference of Planning.*

Tsang, E. 1993. *Foundations of constraint satisfaction.* Academic Press.

Tsang, E. 1995. AIP scheduling techniques - A comparative study. *British Telecom Technology Journal*, 13(1):16-28.

Wallace, M. 1996. Practical applications of constraint programming. *Constraints*, 1:139-168.

Weld, D. 1994. An introduction to least commitment planning. *AI Magazine*, 15(4):93-123.

Yang, Q. 1997. *Intelligent planning. A decomposition and abstraction based approach.* Springer-Verlag, Berlin, Heidelberg.