Heterogenous Scheduling and Rostering
(extended abstract)

T. Sjöland, e-mail:`thomas.sjoland@sics.se`
M. Aronsson*, P. Holmberg, P. Kreuger and S. Lindblom
Swedish Institute of Computer Science, Box 1263, SE-164 29 Kista
(*) Industrilogik L4i AB, Box 21024, SE-100 31 Stockholm
June 21, 2000

Abstract

Reasoning with time is a natural activity for instance in the context of scheduling. We are interested in finding interfacing principles to combine several constraint solvers to allow users, for instance reasoning agents, to have a uniform view of the combined constraint store of heterogenous solvers, while allowing efficiently traceable interactions.

We have designed and implemented a *distributed model for planning and scheduling of transports* in a railway company which is usable also for other production planning problems.

Our model emphasises the use of an agent for the coordination of distinct subproblems occurring in railway scheduling such as for instance the allocation of track resources to transports, the allocation of vehicles to transports, and the allocation of staff to perform the transportation tasks.

Abstraction of constraint stores is an important technique used to enable translations between heterogenous solvers and to improve performance of the overall combined problem of producing a railway schedule. By using abstractions of relevant representations we achieve a hierarchical system which by focussing attention on different levels of abstraction can achieve an acceptable solution faster than previous techniques.

Further articles on this topic [SjKrAsDa98, KrArHoLi2000a, Holm2000, KrArHoLi2000b] give more details.

keywords Railway scheduling, rostering ,abstract constraints, heterogenous constraints

1 Scheduling and Allocation in Transportation

We study the coordination of scheduling and resource allocation in railway transport systems.

Our software for planning of track allocations and engine rosters can be understood as a support system for a manager responsible of the planning task. A fundamental property is that planning steps can be iterated since the input and output data has the same format. This allows successive planning steps where new orders are added to those that have already been given resources, even though this approach excludes the finding of optimal plans/schedules.

This type of problem solving strategy where a sub-problem is partially solved in order to constrain the planning space for a later step in the process is sometimes named *iterative* problem solving.

We currently build an interactive tool that contains specialized solvers for sub-problems and mechanisms to subdivide, abstract and refine problem specifications. In a general perspective the fundamental mechanisms we thereby implement can be the core of a framework for decision support systems for a large class of complex technical systems.

The platform we use consists of SICStus Prolog for efficient constraint solvers and CPLEX for OR-based solvers and Mozart-Oz [Mozart] used for coordination of the solvers.

2 The Railway Scheduling Problem

Railway scheduling is normally treated with various techniques from operations research [surveyrail]. Recently real world scheduling and allocation problems have been successfully modeled and solved with constraint programming techniques [KrCaOlSjAs97a, Si96]. Since some of these problems are NP-hard in themselves and interact in complex ways, problem decomposition is necessary to handle real world problem sizes.

We attack the problem of coordinating the planning processes for a number of resources required to perform transport services in a railway net. We thereby mean diminishing the total cost for using these resources.

The three most important of these resources are *track positions*, *engines* and *traveling staff*.

A railway network is a graph where the nodes are places where significant operations can occur. The operations are for instance meeting between trains using the same part of the net in different directions, overtaking of slower trains by faster ones, waiting for safe security distances to be achieved, unloading of cargo, reassignment of engines, cars and staff etc. It is common that the nodes are classified in different categories, for instance only a subset of the nodes allow reassignment of engines, some nodes allow engines and cars to be parked, some do not. There is also internal state in the nodes, which for some sub-problems can be abstracted away, such as information about exactly which tracks are connected and how they can be used. The arcs are either directed or undirected, perhaps the same arc may be undirected at certain time intervals and directed otherwise, for instance when the net is repaired. There are various constraints that must be fulfilled. For the purpose of this paper we are concerned only with constraints expressing allowed departure and arrival times, and allowed waiting times between arrival and departure and constraints that express the use of a resource.

3 Representations

We use a specification language for specifying *train orders*, a set of *trip specifications* over a rail network. The language allows the expression of more or less exact time specifications for instance for departure and arrival times and for stops at arbitrary places in the route of a train. The language allows the specification of all the information required by the different planning components, including recurrent (for instance hourly, daily or weekly) trips.

A rail network is represented as three interrelated sets:

- **Locations:** Meeting places, some allowing turning e.g. of engines
- **Tracks:** directed and undirected
- **Routes:** predetermined sequences of track traversals

A *location* is characterised by a unique identifier, the set of tracks adjoined to it and the set of routes starting, ending at it and passing through it. Meeting places contain in addition resource and capacity limitations on the location for instance maximum number of trains simultaneously waiting at it. It also encodes local limits on waiting durations for trains passing through it.

A *track* is represented by a unique identifier, the locations it connects, a length and the maximum speed at which different types of trains may traverse it. Tracks (links) are either directed (only movements in one direction are allowed) or non-directed. More than one track can connect two nodes[1].

Our network representation contains appr 330 links (tracks) and 270 nodes (meeting locations) with, for each station, an integer representing the available track resources, that is the maximum number of trains allowed to be simultaneously waiting at the station. We express the constraints on stations representing limits for track resources as cumulative constraints [AB93].

A route finally, is represented by a unique identifier and an ordered sequence of locations. The problem of generating suitable routes for a given trip demand on a network, consisting only of locations and tracks, is not currently addressed in our work.

We use the following terminology for some elementary relations: Tracks connect locations. Routes use individual tracks. Locations are served by routes.

4 Problem Representation - Plans

4.1 Trips

Trips model individual trains traversing the network. A trip is represented by a unique identifier, the route it follows, an engine resource requirement, a location resource requirement and speed parameters used to determine traversal durations for the tracks in its route. Trips are said to pass locations, traverse tracks and follow routes.

4.2 Tasks

A *task* represents the traversal of a track by an individual trip. With each task we associate a unique identifier and the departure time and waiting duration at the origin of the traversal. We associate also with each task a traversal duration and another duration we call headway.

[1] In the case of three or more tracks between two nodes, the train order must express which route (and thereby which tracks) is to be used for a train. In our net this case does not occur.

- The *traversal duration* represents the minimum time distance between the departures of any two trips traversing a track in opposite directions

- The *headway* is the minimum duration after a trip departs from or arrives at a location in the network before a following trip running in the same direction on the same track may depart from or arrive at that location

These parameters are currently determined from the length of the track and the speed parameters of the track and the trip. In a more realistic setting the determination of suitable headways is of course a non-trivial practical problem.

4.3 Plans and Schedules

We use the notion *plan* to denote a (set of) schedule(s) in terms of a given network and a specification of a required set of train trips. This specification allows us to limit the arrival, departure and waiting durations for arbitrary locations in the route of a *trip*.

A *schedule* is the result of assigning values to the time points and durations associated with each task in a plan. The scheduling problem consists in finding such an assignment. A constraint representation of the scheduling problem is generated from the plan as a set of constraints over a set of finite domain variables and constants representing respectively the departure time, waiting duration, traversal duration and headway associated with each task.

5 Track Allocation as a Constrained Packing Problem

The problem to generate an efficient but safe schedule can partly be understood as a packing problem in the time dimension of polygons in the 3D-space that is determined by the net and by time. The *channels* that represent the gap in space-time in which the trains can be located are not allowed to overlap. Also the trains must have reached a station before they can depart for the next station, just as they should not stay for too long a time before they proceed towards the next destination.

Testing whether a plan is safe means to determine that none of the contained channels has any common surface and that all other constraints are satisfied.

The *scheduling problem* can be viewed as an allocation of trips to time slots on tracks such that safety distances are upheld when two trips use a track simultaneously and so that the constraints on the use of the nodes are upheld. We refer to [KrCaOlSjAs97a] or [KCO$^+$97] for more details of the finite domain constraint model used in our prototype scheduler.

6 Scheduling

A scheduling problem contains of a number of *tasks* with restrictions on *start times*, *stop times* and *time extension*. Often the tasks are *partially ordered* into sequences. A totally ordered subset of tasks is often called a *job*. Each task *covers* or uses one or more resources during certain time intervals.

Resources can in general model widely separated entities.

For instance:

- processing equipment in a production process

- staff or vehicles in a transport net

- network resources such as track links, shunting yards and buffer areas

In the railway industry the most important resources are tracks , vehicles and staff.

To arrange the tasks so that no limitations in resources are violated is called *to schedule* the tasks and it is in general a very difficult computational problem. Nevertheless the many practical applications for methods in this area make it fairly well studied.

Many of the best approaches to solving scheduling problems during the latest ten years have introduced as global constraints in constraint programming system [CP89], [CP94], [MS94], [BP95], [CL96].

6.1 Job-shop Scheduling

The simple case where the network consists only of double tracks can be modelled as a job-shop scheduling (JSS) problem [Bak74, CP89, CP90, BP95] viewing train trips as jobs and tracks as resources. Each train trip traversing a track represents a task where the traversal duration is taken as the task extension.

Since in such a model each track traversal requires exclusive use of the track resource it is necessary to keep traversal durations and hence individual tracks short to achieve reasonable schedules. While this model corresponds closely to the real network with individual track segments delimited by signals and sensors, it results in very large scheduling problems (in terms of the number of tasks).

In order to generate tactical plans it is often sufficient to abstract from this model by collapsing individual track segments into aggregate tracks connecting nodes consisting of only meeting points, splits and merges in the network. In this case, instead of taking the traversal duration as the task extension, we use instead the notion of headway which models a minimum (time) distance between two individual trips traversing an aggregate track consisting of several individual segments.

Note that headway does not correspond directly to the task duration in a job shop scheduling problem since the actual interval between the departure times for any two trips departing on a track depends on the difference in speed between the trips. This follows from the fact that the headway has to be preserved at both endpoints of the traversal and means that in general (that is if all speeds are different) we have to consider for each task as many time extensions as there are other tasks to be scheduled on the track resource.

Thus the trip scheduling problem can be viewed as a job shop scheduling problem with release times and either sequence dependent durations [BRR88, BT96] or sequence dependent setup times with fixed durations [LM72, WW77, So90, JD95, Sim95]. Here we ignore the distinction between such models and simply refer to this sequence dependent time extension as the task extension.

Plans based on the headway abstraction are realisable in a more detailed model as long as the headway exceeds the longest traversal duration for an individual segment in the aggregate track resource but, of course, some schedules in the more detailed model will not have a corresponding schedule in the abstract one.

A fully operational system would have to take into account also several levels of warnings that the signal system issues before actually stopping a running train. The determination of a practically useful headway for each track and type of trip will have to take also these parameters into account.

One further complication arises when part of the network consists of tracks used in both directions (single tracks) as is the case in a large part of the Swedish rail network. In the model outlined above this case is handled by considering the time extension of a trip traversing a track in relation to any other traversing the track in the opposite direction to be the full traversal duration.

The scheduling problem that we consider can be concisely stated as:

- Schedule a set of train trips over a fixed network of predetermined routes where trains travel through a network with mixed double and single tracks connecting nodes where trains can meet and overtake
- Maintain reasonable bounds on waiting and makespan

7 Rostering of Engines

Rostering is the problem of assigning resources (such as engines) to perform the train orders. The modelling in [DHKK97] is interesting background information.

We summarize some points that have arisen in a dialog with engine planning experts of SJ (the Swedish Rail) [Fra97, Fra98, SJ98, SJ98]:

- A *train chain* is a least unbreakable activity (mission) for an engine. A train chain can contain several train numbers but is performed by one and the same traffic unit (including the engine). The train chain defines the route on which the traffic unit is run, while the contained train numbers define departure and arrival times at turn/end points.
- An *engine roster* is a sequence of services (in this case train chains) to be performed in a particular order. Often it is also demanded that the rosters are closed circuits and that they can be performed repeatedly a number of times. The engines are planned to drive repeated (cyclical) routes, also called circuits.
- One critical aspect on the rosters is the connection and disconnection of cars, *turning*. The least time for this operation, *the turning time*, usually varies between 15 and 30 minutes, depending on for instance properties of the station on which the turn occurs.
- A rostering plan will also contain a number of generated orders for empty trains which are to be added to the original train orders. Transportation of *empty trains* are planned train chains which are being performed by traffic units which only contain engines without cargo. These are used if the engine is needed at some other location than where an ordinary train chain has been finished. This is of course costly and should not be used if not necessary. A variant is to let engines be part of an already planned train chain which in general should be preferred. In the case of engines this kind of a train chain is named a *multiple engine*. The empty trains necessary to create connected engine rosters should also be generated and scheduled as a result of the rostering planning phase. The allocation of engines should ideally (in order to minimize costs) utilize the existing set of engines in such a way that empty trips and waiting times at turns is minimized[2].
- *Type week* - the plan for a week, that is trains that recur regularly during a number of weeks. The rosters for engines are not currently described with type weeks.
- The cost measures for the rostering plan are for instance accumulated waiting time for all rosters. Another demand is that the rosters should be approximately equally long so that the engines are used efficiently. This demand can also be formulated as a demand on waiting times for both the time an engine spends waiting to start its first trip and the time that passes from the end of

[2]In strategic planning it is an important goal to minimize the number of engines needed to perform a typical set of plans.

its last trip until the time point when the last roster is finished. The cost for transfers of engines varies depending on which type of train is being handled.

- Ordinary trains - least cost
- Transportation of engines as part of an ordinary train, middle
- Empty trains (engines) travelling alone, highest cost

The fixed cost for using an engine in a roster is also a parameter in the cost function.

In a planning phase for the network we consider there are 3900 train chains and up to 1500 possible trips of empty trains, that is 5400 train chains. As an estimate the planner must consider up to about 500 000 possible turnings.

The current state-of-the art planning systems are based on OR-techniques [DHKK97] and have the following performance: - 2 min, full scale engine rostering plan without considering engine types - 5 hours, plan considering engine types

This is due to a difference in complexity of the two formulations of the problem as discussed in [Löb98].

7.1 Rosters

Some types of tasks can be understood as cyclic processes or flows. In a simple vehicle problem the movement of the vehicles can be considered as cyclic paths in a graph. How hard it is to solve such problems depends strongly on the conditions being posed on the arcs of the graph.

If the condition can be determined locally the problem can be modeled as a *network flow* [AMO93]. For such problems there exist very efficient optimizing solvers (see below section on network flows). Such techniques are used in practice in the generation of engine rosters [DHKK97].

If an arc in the graph represents a *turning* of an engine from one trip to another the model is a classical network flow only if the decision whether the arc satisfies the model can be made locally, that is independent of other tasks (trips) than the two that the arc connects. This in practice rarely the case. In production planning for the railway industry two important rostering problems occur that do not have this property.

- The problem to generate engine rosters without a fixed schedule
- the problem to generate cycles of a determined "length" for instance working periods with daily rest periods

A lot of the work that has been done in the project have focused on characterizing these problems. The following references put the problem in context [BG81], [Sav85], [GA86], [Sol86], [Sol87], [SD88], [PR93], [TP93]. We believe that mechanisms to solve problems of this type can be of great value to decision support systems in a wide variety of domains.

7.2 Set Partitioning

The problem to determine the rosters can be formulated as set partitioning with additional constraint, for instance regarding time [Sid]. The set of train orders should be divided in as many disjunct subsets as there are engines. Further conditions on each subset of trains is that it should constitute a connected roster where turnings are only allowed on a subset of all the meeting locations (the turning locations) and the order between trains is implied by the roster.

How the rostering problem can be formulated as a constraint problem is discussed for instance by Simonis [Sid]. We have implemented a rostering algorithm for engines in the same spirit which handles a medium size problem. This corresponds to the state-of-the-art in this area [Si96].

It should be noted that the "set-partitioning" abstraction in itself doesn't give any support for the generation of suitable empty trains or multiple trains. How these shall be generated with the above sketched approach is not yet determined. In [Sid] Simonis solves the problem for empty trains. Note that in the model an empty train can be added anywhere in a given roster and that we in order to find an optimal rostering plan need a cost function which considers both the total waiting time and costs of empty trains.

7.3 The Approach of the TUFF-project to Engine Rostering

In order to investigate the possibilities to allocate engine and track resources at the same time we specified and implemented an engine rostering algorithm using constraints. This algorithm allocates a determined or minimal number of engines to a train order so that all trips can in principle be performed within the given time interval and the trips are ordered in a connected sequence for each engine.

A rostering plan constrains the earlier train order by limiting the possible arrival and departure times for end stations significantly. Our current approach is to in a first phase specialize train orders to orders of rosters containing train orders with more precise demands on departure times and fixed traversal times. We associate to each train a predecessor or successor so that the set of trains is divided into disjunct sequences (rosters). With each train is also associated a waiting time before the next train in the roster departs. As a part of the problem we should also determine which trains start and stop each roster and conditions imposed on the locations at which this happens.

The result of a rostering plan is then in a second phase used to sharpen the input for the track allocation algorithm. The train planner verifies whether the rostering plan can be instantiated to a track allocation plan. If this is not successful, the initiative returns to the rostering planner which generates a new proposal. In any case an economic evaluation of the generated track allocation plan and overly costly parts of the plan are identified. For the purpose of eliminating these a bidding protocol between the two planners might be useful.

7.4 Representation of Rosters

Represent a roster as an ordered sequence of turnings. A turning is defined by a turning location V, an arriving train A with an interval for the allowed arrival time points, a departing train B with an interval for departure time and a variable Vt for the turning time (identical with the waiting time at the end station for A) the value of which is given by the difference between departure time and arrival time. The turning time variable Vt is limited downwards by a least turning time (defined by the station and the engine type) and upwards by requirements given by the turning location, individual rosters and the whole plan.

The engine allocation problem is determine B for all turnings given A and a set of constraints on Vt. B is to be selected among the trains that are scheduled for departures after A's arrival time at V. This set is only partly determined by a train order since the order between arrivals and departures at a meeting location is partly the result of the work performed by the track allocation planner.

What the rostering planner needs to do is thereby to partly determine (partially order) departures and arrivals at the turning locations, that is to possibly specialize the train order. Given such a specialized and extended train order a rostering plan with a reasonable waiting time can often be generated. If not, the train order is relaxed to make this possible.

We are now facing two problems:

- To generate a partial order of arrivals and departures for turning locations without generating a complete schedule.
- Given such a partial order, to compute an engine roster plan with reasonable limits on the total waiting time, roster lengths and empty trains.

An approach to generate (enumerate) the rostering plan could be to breadth-first (in parallel) add as successor at the end of each roster the train whose earliest departure time from the turning location is closest after its earliest arrival time. Limits on waiting times and time from the last arrival in each roster to the last in the whole plan can be assigned so that the length of the rosters (in time) is limited downwards.

8 Basic technologies

8.1 Constraint programming

Constraint programming (CP) [vH89], [Tsa93], [Sim96b] is based on the idea of an abstract space of statements (restrictions or conditions), called a *constraint space*.

Some of these statements can be understood as fully determined. Take for instance the claim that a given train will depart from Avesta 15.05 thursday April 15th in the year 2001. Other statements are less exact for instance that Ovako needs to transport between 320 and 380 kilotons of steel from Hofors and Hellefors to Malmö next year. Both these statements can be represented as conditions in a constraint programming system. We name them *constraints* or restrictions on the value space for the variables contained by the statement.

It is of course non-trivial to determine in a space of such statements how for instance these two statements are related given some mathematical model of a planning problem but under certain circumstances it is possible to make calculations with such abstract objects and to determine for instance logical consistency, that is to decide whether the statements are possibly both true.

It is also possible to compute one or several *witnesses*, that is assignments of values to all variables in a space. This is called to *enumerate the search space* for a given problem and in general it contains the utilisation of a search procedure. Such witnesses can in the production planning domain for instance be concrete production plans. If many such plans are generated they can be compared with respect to different measures of *cost*. Finding the, in some sence, best plan is modeled as an optimisation problem in the constraint system.

Much of the search when enumerating the search space for a given problem can often be eliminated with a technique named *constraint propagation*. This means briefly that each statement that is not completely determined is considered as a temporarily interrupted computation which can be made to interact with other similar concurrent computations. Computations are interrupted when the information that is needed to determine a value is missing but they continue once the information is later available. A constraint programming system can be seen as a set of parallel processes which communicate, interact and are synchronised via shared variables in a dataflow graph.

Whether it is possible to compute solutions for a given problem or not is to a large extent depending on the expressive power of the language being used to express statements about the problem. Formulating a mathematical model of some real process in such a language is in general a very hard problem. In spite of this there have been good results by using constraint techniques to model and solve known hard planning problems, for instance many classical scheduling and resource allocation problems.

Consider the example with a departure time. If it is known we would like to represent it as a number. Assume now that it is not known, but that there is nevertheless some information available about it, for instance that it must be between 11 and 12 some given day and that it must follow after the arrival time for some other trip.

This is a very strong restriction of the value for the departure time. Yet it contains an ambiguity which separates it in a fundamental way from the totally determined one.

A number of such statements can for instance express the relation between this departure time and other departure times (or arrivals) and can be said to represent a specification of a production plan. The more determined it is, the closer it is to a finished plan with totally determined times.

It is possible to construct a production plan by successively adding more and more information to a space containing conditions (a constraint space).

8.1.1 Finite domains

The constraint programming systems that have been most actively developed the last ten years are those that built on *finite domains*. In such a system each variable can take on values from a finite set of discrete values. This type of variable is natural to use to model discrete entities such as the number of engines or staff that have been allocated to a given task. They are, however, unnecessarily restrictive when the modelling concerns values that can be assumed to vary over continuous domains (with an infinite number of possible values), for instance time.

Techniques from operations analysis, for instance linear programming (LP) and integer programming (IP) handles models efficiently where most of the variables are continuous and where a simple and well defined cost function well captures the "goodness" of different solutions to a given problem. In contrast, the techniques that have been developed in constraint programming, using finite domains, work well also when a majority of the variables model naturally discrete entities, when the cost function is hard to determine and when the model contains complicated (for instance non-linear) conditions. In this way these two classes of techniques can be said to complement each other.

8.1.2 Global constraints

The first type of constraint that was studied in constraint programming was constraints that relate one or two variables, for instance $<, \leq, \neq, =$ etc. In contrast to these simple binary constraints the focus has in recent years more and more been on complicated constraints between an unlimited number of variables. Exemples of such constraints are such that relate variables with the value of a linear sum or that maintain pairwise disequality of an arbitrarily large set of variables.

Such constraints can in principle often be encoded in terms of a set of simpler binary constraints which semantically have the same meaning. This is rarely practical, however, since an efficient solution is often too computationally complex to be realised by simply consider the variables in a pairwise fashion. The expression "global constraints" for this type of constraints was introduced in [BC94] and refers to arguments that can be made over a multitude of variables related with a non-binary condition.

Global constraints are abstractions of more complicated properties of problems and enables computations on a more detailed model. Many times methods from operations analysis or algorithm theory, which operate on graphs, can efficiently and naturally be integrated into a constraint programming system as global constraints. This is an active and very promising research area in constraint programming.

Among the examples of global reasoning that have succesfully been introduced into constraint programming systems are a number of fundamental scheduling mechanisms.

For a systematic description of a large number of global constraints see [Bel00].

8.2 OR-techniques

We should compare constraint programming with traditional optimization techniques that have been developed in operations analysis. These techniques are often based on massive computations where input are given in the form of a complete problem description. These methods are very efficient when the model of the real problem suits well into some wellknown class of problems and the problem is pure, that is independent of a context which is hard to describe or too complex. They are therefore often less suitable in an early stadium of the planning process when many parameters are yet unknown.

On the other hand they have a given place once various side constraints are eliminated with other methods and when comparison of results of strategic choices in well delimited sub-problems is desired. Many of these methods can be seen as planning primitives, methods that can be used to investigate properties of the problem.

Constraint techniques, on the other hand, offer the possibility to maintain a dynamically changing space of statements that represent all currently possible (sub)-plans given the constraints from customer demands, resource limitations and cost considerations. We can choose to optimize parts of the plan late and incrementally and leave the rest of the plan only partially determined. Another advantage is that it is possible to incorporate techniques from operations analysis into the constraint paradigm in the form of global constraints. This makes constraint programming an "*integrative*" project, where techniques from different areas are collected and made available to modelling experts without requiring from them a detailed algorithmic competence. This we see as one of the big advantages with constraint technology. Some of the techniques we see an immediate use for in the production planning domain are discussed below.

8.2.1 Linear programming

Linear programming is used to model optimizing problems where the cost function can be expressed as a linear sum over continuous variables and where the conditions are linear inequalities over these.

Example: Linear program

Maximize

$$c_1x_1 + \cdots + c_nx_n$$

where

$$a_{11}x + \cdots + a_{1n}x_n \leq b_1$$

$$\vdots$$

$$a_{m1}x_1 + \cdots + a_{mn}x_{n \leq b_m}$$

and

$$x_1 \geq 0, \ldots x_n \geq 0$$

	LP (linear progr.)	IP (integer progr.)	CP (constraint progr.)
variable domains	continuous	integer/mixed	integer (finite set), intervals
cost function	optimization	optimization, search	domain limitation
scalability	polynomial compl.	certain problem classes	good for decision problems
search	SIMPLEX	problem dependence	spec. method integration

Table 1: Some properties of a selection of techniques

Simplex Simplex is the oldest and most well known algorithm to solve this type of problem. Despite that Simplex in most applications has very good complexity properties (polynomial time complexity) there is for certain classes of problems specialised algorithms which are even more efficient. An example of such an algorithm is *network flow optimization* (see below).

The main disadvantage with Simplex is that it works only if all the conditions are linear and the variables are continuous. The latter requirement excludes all disjunctive conditions (decision problems) where the value of a function depends on a boolean variable. The research field studies the problems that occur when the first demand is loosened. In OR it is called non-linear programming and will not be further covered in this overview. The problems that can be formulated when variables are allowed to vary over discrete values are studied in the area of integer programming. Sometimes problems where both linear and integer variables occur, are called *mixed*.

Network flows Network flows is a collecting concept for many types of linear programming algorithms which handle optimization of flows in networks. The area is well researched and many problems are classified. Three important sub-problems in network flow optimization are *shortest path*, *maximal flow* and *minimal cost*. When computing the shortest path the shortest path between some or all points in a network is determined. Maximal flows encode problems where the flow between pairs of points in the network should be maximized. In minimal cost problems costs are assigned for the flow between different points and the cost of sending flows from one or more points in the network to one or more destinations has to be minimized.

Since this area is large, the algorithms are fairly dissimilar. For minimal cost problems specialised variants of Simplex are often used.

8.2.2 Integer programming

Many times the introduction of integer variables can be avoided by using more or less advanced modelling tricks that encode a problem with integer demands as a linear problem. This demands, however, often a comparatively specialised mathematical competence and is far from always possible.

If integer conditions are allowed, a significantly more free modelling is achieved, but instead search has to be introduced in the algorithms. The search mechanisms that have been developed to solve this type of problem use in a direct way the cost function and solve in each step (iteration) *linear relaxations* (for parts) of the real problem.

A great many heuristic methods have been developed to make the search converge faster for certain classes of problems. Given a concrete problem, determining whether it can be modeled as an instance of one of these well studied problem classes also requires specialized mathematical competence.

Lagrange-relaxation Lagrange-relaxation is a technique which can be seen as systematically reformulate integer demands as parameters in the cost function in a corresponding linear program. These parameters are then adjusted by solving the relaxed problem with respect to the parametrized cost function. In each step the parameters are adjusted based on the results from the preceding iteration.

Lagrange-relaxation has successfully been used to solve for instance very large vehicle rostering problems [Löb98] and it is also one of the most important techniques to solve pairing problems (see below). For a general introduction to Lagrange-relaxation see [Ree95b]. This reference [Ree95a] contains in addition descriptions of a number of so called local search mechanisms which are not treated in this overview article.

Pairing algorithms Most optimizing systems for staff planning in the transport sector (that is for travelling staff) work in two steps:

1. Produce a number of possible circuits, that is jobs containing tasks (legs) that can be executed in order by for instance one person; each circuit should also (sometimes by adding passive transports) describe a cycle in the track net graph, that is, for instance starting and ending in the same location

2. Solve an optimizing problem that contains in choosing among the above generated circuits a subset such that all tasks are a part of at least one cycle and so that a global cost is minimized

Step two is from a mathematical point of view simple to formulate, even if the size of the problems can many times make them hard to solve. In order to reach as good results as possible one must therefore in step one generate as many candidates as can be handled computationally in step two. It is however in practice often impossible to consider all possible circuits, so the selection that is being made is of utmost importance for the result.

Step two can simplified be described as constructing a matrix, with the tasks as rows and the in step one generated circuits as columns. In the meeting points between the tasks and circuits there is the value 1 if the task is a part of the circuit, and 0 otherwise. Furthermore it is required that each row shall sum up to at least the number of persons needed to perform the job. The task for the optimization algorithm is now to assign boolean (0/1) values so that the cost is minimized.

There are special algorithms for 0/1-matrices that are very efficient and which can handle large data sets.

Step 1 is not as simple, and it is here that the different commercial solvers differ. This is the step that is named *pairing*. It is important that a good selection of alternative circuits is generated in this step, since these are the only candidates considered in the search for solutions.

The circuits should also satisfy conditions that encode laws and union agreements, which are often hard to represent in a correct and efficient way. It is far from clear that agreements and legislation is mathematically consistent and there is always room for interpretations that often vary locally. Since agreements also change with time, it is important that they are represented in such a way that they are easy to maintain.

In addition a number of heuristically motivated generation conditions are represented and cost parameters, which limit the choice of circuits to those that are "reasonable".

These two sets of conditions are then used in different methods to generate candidates for circuits for step two. Two main mathods are used in this context: *Integer programming* with *Lagrange-relaxation* and *column generation* (see for instance [DSD84], [RS94]). This is still an active area of research within OR.

8.3 Agents, Concurrency and Distribution

By subdividing a planning system into possibly physically separated subproblem solvers and specifying protocols for the interactions between the different parts of the system a distributed software model is achieved. Such models are nowadays often named *agent models*, especially if the different parts have a certain degree of independence. One reason to use this design principle is to clarify the communication between different actors (internal and external) and different organisational units. Another is to reduce the inherent complexity of producing a train plan.

Parallel or concurrent search and distributed solving of constraint satisfaction problems might speed up the finding of the solution compared to solving the problems one at a time or in a centralised manner[paropt, SG96].

We have investigated techniques to partition large complex problems into smaller and less composite ones where communication between subproblem solvers is limited to for instance information about contributions to a globally defined cost [KrArHoLi2000b].

8.4 Negotiation Protocols

We have investigated interfacing principles to combine several spaces to allow planning agents, to have a uniform view of a heterogeneous set of spaces, while allowing efficiently controllable and reliably traceable interactions. We relate constraint-based descriptions of the spaces with different levels of detail and formally relate the different levels to each other.

The agent modeling approach discussed above necessitates definition of *negotiation protocols*. Each planning agent produces a local plan. After the local plans have been produced, the computational agents start negotiating with each other. The negotiation is conducted with respect to a globally defined cost function covering the entire planning problem. After having exchanged tasks the computational agents can produce an improved plan. This process is iterated until the negotiation process halts or is interrupted. The resulting total plan, given through the combined local plans, converges in this process towards an optimum. Typically the items exchanged in the protocol contains some form of plan abstractions (or description) for the relevant subplans representing limitations, cost estimates etc.

8.5 Support for Agent Models in Mozart-Oz

One goal of our project is to develop base technology in the form of software infrastructure enabling a uniform access model of the information in a set of spaces. From the perspective of the user (or a control program) the different spaces should appear as a single one. The distribution is to be transparent since the access model abstracts away from the full details of the network and differences in data representation. Furthermore the access model offers a common information model and a uniform language for access and modification of the spaces.

9 Modelling Train Planning with an Agent Model

We wish to define a suitable agent model expressing a division of the problem solving in sub-problems solved (partly) independently.

The rostering planner for engines, together with the track allocation planner and their interfaces form a minimal agent model and the procedure for interaction between these is a rather trivial negotiation protocol.

Other approaches that might prove relevant concern for instance auction models for allocation of track capacity [IsNi97].

9.1 Alternative Agent Models

In general it is possible to make a division into agents in several alternative dimensions. The choice of division is controlled partly by performance demands on the planning and partly by organizational demands on the architecture of the system (for instance distributed use). In the case where the purpose is to considerably increase the performance by running the planning on several computers it is required that the problem can be divided in such a way that parallel computation is enabled.

In an agent model where a plan manager is responsible for the total plan cost optimization, the different agents could be made more independent and the generated plan could possibly be the result of an iterative process where the agents estimate the costs that occur in the frame of their respective areas of responsibilities and generate a set of internal bids (plan suggestions) converging towards a globally optimal cost.

9.2 Possible Problem Division

Some different approaches to subdivision of the planning/scheduling problem that might be useful are listed:

- **Regional** Regional planners can be agents that schedule the local plans on a regional level while a national planner takes care of the common plan for the long distance trains[3].
- **Train types** Divide the system based on the different types of trains.
- **Customer** Divide the system based on the different responsibilities for the service salesmen, that is different groups of customers.
- **Type week** Divide the planning in different time intervals with for instance a planning agent per type week. Since this partitioning gives largely independent partial problems, at least for the scheduling problem, we might have here an interesting possibility for parallelization.
- **Organizational** Different planners are used for scheduling, engine rostering and staff rostering. The agent architecture could be separated based on these different subsystems.

A table indicates some different subdivisions in the timetabling problem for the swedish railroad.

Partial domains	tracks	engines	staff
X2000	X	X	X
Intercity	X	X	X
Interregional	X	X	X
Goods-national	X	X	X
Goods-regional	X	X	X
...	X	X	X

9.3 An Agent Model Based on Organizational Considerations

Figure 1 shows the agent model we currently consider.

The agent model divides the system as follows:

[3]This method is currently implemented for the track allocation planner in an early prototype form.

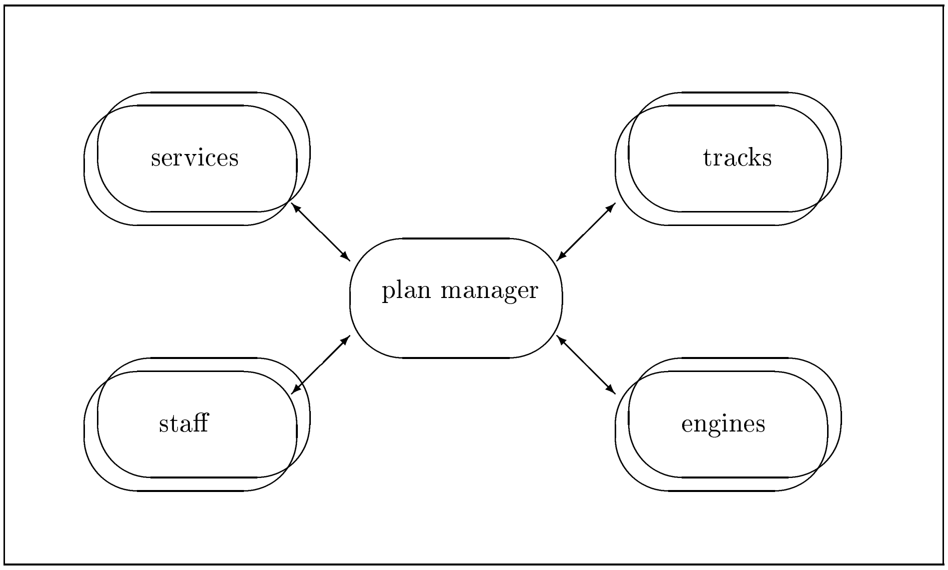

Figure 1: Agent model based on organization.

- The *service producers* generate train orders and negotiate contracts with customers
- The *track allocation planners* supply track resources (generate time table)
- The *engine planners* supply engines (generate engine rosters)
- The *staff planners* supply staff (generate staff rosters)
- The *plan manager* coordinates the negotiations and maintain the total plan

The different types of agents (except the plan manager) might exist in several *clones* each responsible for some sub-problem of the given type.

9.4 Protocols and Interactions

In order to make good suggestions concerning the coordination problem we need to understand:

- The underlying problem domains
- How (partial) solutions in the respective domains affect costs and opportunities in the others
- The "protocols" that are currently used in the manual process
- How the suggested plans are iterated today and how they are expected to look in five years
- What the software infrastructure will look in five years

9.5 Formalising the Protocols

Some of the work items remaining to formulate are

- Abstract descriptions of the different problem domains which are detailed enough to allow reasoning about proposed partial plans

- Representations of specifications and partial solutions in the respective domains
- Mechanisms to generate specialized specifications given abstract descriptions of the planning situation for the different domains
- Relevant and computationally reasonable cost functions for the respective domains and globally for the coordination agent

9.6 Coordination mechanisms

Most more complex activities demand co-ordination of several more or less heterogenous resources. For really large enterprises the problem must in addition for each resource be divided into smaller sub-problems in order that these shall be of a manageable size.

To handle problems with different degrees of detail is necessary if both the costs that occur locally and the possible profits by co-ordinate the shared-resource usage of the local activities are considered. We need an abstract (less fine-grained) model to estimate the total costs at the strategic and central level and concrete and detailed local cost measures for each sub-problem. The aim is to limit the local choices so that they at least meet the elementary demands on the central level.

9.6.1 Interactivity and distribution

Planning of so complicated and large activities as those of a major supplier of transport services must for many reasons be done through manual decisions in decisive points. A decision support system is not intended to replace the expert who makes these decisions with a machine, just to release her of the burden of routine decisions and to supply as good a decision background as possible.

Many times it is also for organisatorical and political reasons good to decentralize and to distribute the planning process. In spite of this there are also large gains to be made in the co-ordination of decisions that clearly affect several parts of the activity. One obvious example is shared resources such as infrastructure.

The assumption we work with is in many ways in line with these observations. We work with planning components that can communicate suggested solutions and local costs and that can work with different levels of abstraction. With this approach it is natural that a user inspects the results and controls the direction in which the system should proceed, which methods to apply etc.

Furthermore each problem solving component is implemented as a separate process (thread) which using primitives of our implementation language [Smo95],[Mozart] can also be distributed to physically separated locations in a computer network. This enables the decision process to be distributed and that users with different areas of responsibility together, but with limited knowledge of the detailed models of the others, and according to common (agreed or centrally dictated) criteria build a working tactical plan.

The experimental systems we work with today are for natural reasons rather far away from the realisation of such a vision but we are convinced that this type of system given a high enough amount of work and expertise are in principle possible to develop.

10 Sharing vs. Abstraction of Constraint Spaces

Solving constraints in a distributed fashion and coordinating separate solvers are related but different problems [SG96]. In distributed constraint solving the problem is essentially to distribute a constraint space over several computers, while coordinated solvers may or may not use distribution to solve separate problems. In the case of coordination there is a need to relate the constraint formulations used in the different solvers, thus enabling a coordinating agent to guide the sub-solvers. Bidding strategies [FiMuPiSch95] similar to contract nets [Sm80] should be expressed with explicit protocols where constraints or abstract descriptions thereof are communicated. The abstraction of constraint expressions from different spaces must guarantee that important information is not lost.

11 Abstraction for Reducing Complexity

Constraint solving is often limited by a prohibitively large search space, for instance by the enumeration of many redundant variants.

The use of meta-operations such as constraint relaxation, redundant modeling [CLW96] etc. could be utilized to improve the search behaviors of a solver. Another approach is to use abstraction to speed up search problems. This has been investigated in AI for general search problems [HPZ95, HMZM96].

11.1 Abstract Constraint Solving for Scheduling

Abstraction of constraint models can be used also for efficiency. A particular technical idea that comes naturally in our application is that of *Abstract Constraint Solving*. By constructing a suitable abstract constraint problem we get a constraint problem over a smaller domain than that of the immediate (concrete) formulation of the constraint problem.

By defining the abstraction so that a safeness relation (for non-solutions) is upheld we guarantee that a solution in the abstract domain corresponds to a concrete solution.

This methodology should generate feasible solutions faster than by solving the concrete problems directly, although optimal solutions that would be found with a more detailed larger search space might be lost in the process.

11.2 Abstraction Hierarchy

The safeness relation gives as expected a hierarchy of constraint problems (and domains) with increasingly less detail, and thus likely faster execution but possibly poorer plans higher up in the hierarchy. By switching attention dynamically between different representation of the problem based on some heuristics it might be possible to keep a suitable quality of the plans while speeding up the solution process. This technique is known as widening/narrowing in abstract interpretation [Cousot].

11.3 Plan abstraction and -concretisation

A production planning problem can be divided into several relatively independent dimensions. One is the division based on resource types as discussed elsewhere in this report. Co-ordination of the plans for these resources demand tasks at several simultaneous abstraction levels. So for instance are the problem the jobs in the vehicle problem. The legs of the staff rostering problem can be built by track slots but are in themselves parts of the trips. When co-ordinating these resouce problems the need arises to in a uniform and unambiguous way move between these different levels.

When dividing such a resource problem into smaller parts based on for instance regions or type of traffic the need arises to merge sub-plans generated locally with incomplete knowledge about the resource conflicts that occur when they are to be co-ordinated. One way to attack such a problem is to abstract from the given solutions to a common problem description that keeps as much as possible of the locally good properties of the sub-plans. In order for this to be meaningful, it is then obvious that such an abstract "solution", seen as a problem, is simpler to solve than the problem that would have occurred if the original problem descriptions had been joint to begin with.

A third example is the case when the problem is simplified by for instance combining simple resources to more complex ones in order to reduce the total set of tasks in a given problem.

In particular in the last of these cases one often wants that the abstractions used shall meet certain properties, for instance that the existence of an abstract solution *guarantees* the existence of at least one conceret solution.

These examples aim to illustrate the need for an unabiguous and extensive formalisation of structures and mappings that can be used to model problems of this kind.

The approach we currently work with is one that build on a concept of task that is hierarcically organised.

Each primitive mission and job is represented as a *mission structure* with a unambiguously determined sub-structure. A problem structure is represented by partly such a mission structure, partly a (partial) order relation on the individual tasks and their parts and finally of a formal representation of the resources used by the tasks. *Abstractions* and *concretisations* are defined as mappings on problem structures.

Note that a solution for one problem in this context can be considered as a specific type of concretisation of a problem and in itself is a problem structure.

In todays prototype system this approach is only partially implemented. An example of a functioning problem structure is the *trip sets* that are the most important component in the protocol used to send problems and solutions between components for vehicle planning, track planning and the coordinator. Abstraction occurs for instance when the vehicle component ignores much of the sub-structure that occurs in the track component which has to take into consideration all track resources and station resources.

11.4 The Abstraction of Time in the Scheduling Problem

Time points in the constraints are represented as finite domain variables in our scheduler [KCO+97]. This means that the scheduling problem is discretized. Since different parts of the network might have different loads there is a possibility of generating a large number of redundant solutions with this approach. This is a problem in itself and for reaching an optimal solution various propagation methods and search heuristics might be applied. We are however primarily interested in feasible solutions since the makespan of the schedule could be controlled by a global parameter of the maximum time allowed for the total solution.

Our first suggestion for abstraction is fairly straight forward: use fewer timepoints in the intervals described by the finite domain variables.

A second orthogonal suggestion: use a simpler network such that only those nodes where significant operations such as overturns, meetings and reallocation of engines or staff are allowed, are taken into account. A later phase can generate times for stops at intermediate nodes.

11.5 Soundness of the Abstraction

In order for an abstract constraint problem to be a solution a (negative) safeness criterion must hold between the set of solutions described by the concrete constraint C and its concretization g, and abstraction a.

$$s \notin C \rightarrow s \notin g(a(C))$$

This is understood as a statement that any tuple s that is *not* a solution to the concrete problem C is also not a solution to the concretisation of the abstract problem $g(a(C))$.

The equivalent logical reformulation

$$s \in g(a(C)) \rightarrow s \in C$$

gives the immediate property that we want.

In the case of using fewer timepoints this obviously holds as long as the timepoints occurring in the abstract problem are also in the concrete formulation.

In the case of using only a subset of the nodes the problem becomes a bit more subtle. Consider first a concrete example:

$$a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e$$

where the trip is associated with departure and arrival times for each of the nodes in such a way that the constraints hold.

Consider now an abstracted formulation of the constraint for the transportation between a and e:

$$a \longrightarrow c \longrightarrow e$$

We need to guarantee that the traversal time parameter for the arc between nodes a and c in the abstract network cover the longest possible time between a and c in the concrete network. This is of course doable [4] but might prohibit the finding of a schedule if the abstraction is applied on parts of the network where there is a lot of traffic.

11.6 Abstraction of Rosters in the Allocation Problem

For the set partitioning problem of allocating train orders to engines it could be useful to abstract away the explicit reasoning about time-points and rather use only the partial order between train orders implied by the geometry of the net.

Consider a partially ordered set of train orders $O = \{x_0 \xrightarrow{*} y_0, x_n \xrightarrow{*} y_n\}$ between nodes.

Partition the train orders over a set of engines $V = \{E_1 \ldots E_m\}$ in such a way that the partial order over O is guaranteed.

Here the important information is the order, rather than the exact timepoints of the turnings connecting the train orders into a roster.

The partial order can of course be expressed in terms of constraints over departure and arrival times, as indicated above, but it might be profitable to consider a different representation for the rostering problem solver, the result of which is then translated into constraints for the time points in the coordination protocol. It is desired that such translation of representations can be expressed as safe abstractions (and concretizations).

[4] Here we disregard the complication that particular constraints on locations dismissed in the abstraction, such as cumulative constraints on the number of co-located trains at such locations at any time point, might be hard to maintain.

11.7 Planning strategies

The above described abstraction- and concretization mechanisms can be combined in complex patterns, called strategies. Such strategies can be used both to enable planning of bigger problems that are possible to handle as a unit and to reduce the required time for the planning by dividing the problem into smaller pieces.

One strategy can be used for instance to co-ordinate the results of several track resource planning sessions. This is interesting for instance if remote trains travel through a geographical area where also local traffic occurs, and can also be used to join together solutions for track plannings of geographically closely related areas. Strategies need also not be limited to abstractions and concretizations, but can furthermore also be used to control the co-ordination between the different sub-problems.

11.7.1 Planning language

In order to make it practically possible to use strategies it is required that a strategy can be easily be defined. It should also be saved and reused. This becomes possible with the aid of a planning language.

Such a language should offer possibilities to use mechanisms for abstraction and concretisation. It should also support the use of conditions, so that if one part of a strategy is failing a new attempt is made to solve this part of the problem in another way. Finally a strategy that is entered as an expression is the language need to be stored and again be read into the system.

It is worth noting that a planning language is not just useful for solution of the problem at hand in the transport sector but that it can simplify the solution of many types of larger planning problems.

12 Applicability in other domains

The techniques that are described in this report have applications in a broad spectrum of activities in modern society. The overview below lists a selection of such activities.

12.1 Process industry

12.1.1 Steel manufacturing

In the steel industry there are a number of intricate planning problems around material flows and scheduling of processing units. At OVAKO Steel in Hofors, Sweden, within the EU-project TACIT, SICS in 1997-98 performed a study of the process flow containing furnaces. The purpose of the project was to show that constraint technology is mature enough to be used in real industrial applications. This problem consisted of 2-3 relatively well defined resource problems.

SICS developed, in cooperation with planning expertise from OVAKO, a decision support system for this flow of materials. The tool reads data from the processing units and after a step where certain parameters can be changed, it helps the manual generation of running schedules for furnaces, and certain transport equipment. The size of the problem and demand for short response times made it impossible to use optimisation techniques. Instead the problem was solved with an *eager*[5] algorithm that acted upon a constraint programming model. With this approach we could find plans of a high quality for up to two 24-hour periods within the time range of 30 to 60 seconds.

[5]Without search.

An advanced interface was developed to represent the results in the form of Gantt-schemata and other graphical tools. Alternative plans (based on different manual decisions) can be stored and compared manually and considering different cost measures.

12.1.2 Paper industry and (bio-)chemical industry

In other process industries, for instance in paper production, petrochemical industry and fuel production there are similar problems as in the OVAKO domain. Of course there are also other problems. No activity of this complexity can be modeled according to a “blueprint” model.

This speaks for a technology that allows rich heterogenous models and in itself can incorporate both general and specialised problem solution methods for sub-problem solution.

Constraint programming seem to be a promising candidate for such a technology and ought to be the basis of a general method to model a very large class of problems.

12.2 Manufacturing industry

The TACIT-project was an international co-operation that contained also parts in France and Hungary. The hungarian partner developed a system for scheduling of machine resources for plastic components. In this case the size of the problem was such that a traditional search couldn’t be done directly in the constraint programming system. The system is not optimizing but some constraints on the cost parameters makes the plans generated by the system of good quality.

This problem is similar to many that are found in manufacturing, for instance tool manufacturing, textile manufacturing, vehicle manufacturing etc.

12.3 Building and construction industry

SICS’s first contact with this type of problem was in the framework of improving the working conditions for local managers and other actors on construction sites, especially in construction of buildings. In several projects a demonstration environment was created pointed to how computer technology can be used to significantly improve the information possibilities and calculation possibilities for e. g. local managers. The problem domain is characterised by a great sensitivity for changes in supply and demand in the market, and lead times have been constantly reduced which increases the pressure on the actors to be able to handle fast changes while keeping committed delivery deadlines. It is also common with disturbances during the construction work, partly stemming from changed orders but also “surprises” in ground conditions, wheather and other factors.

During the last few years a standardised product model has been developed within the business branch. This is a necessary component for a computerized planning support tool. Drawings for a building are then replaced with a model in this standard format, which is delivered from an architect and which is taken to be input to a planning system (or to other types of systems, for instance offering calculus systems). SICS early pointed to the need for product models in order to create a platform upon which support systems could be implemented.

Today project planning systems are used in this business area, but a complete coverage from product data models as a background for planning systems is still missing. There is ongoing activity in the business to create a better connection between data from construction tools and planning systems, for instance Skanska and NCC have programs to achieve better software connections between their different support systems.

SICS has during the period 1988 to 1995 developed demonstration software, demonstrated these in different conferences, to industrial and other actors and given seminars, and also participated in development projects at Skanska (under the so called 3T-program).

The french construction company Bouygues is involving some of the top researchers in constraint programming [CK92], [CL94], [CL95b], [CL95a], [CL96], [CL97], [Cas97].

12.3.1 Transport optimization

Studies have in the EU-project CHIC-II [CHI98] been performed concerning the problem of given a set of roughly scheduled tasks allocate vehicle resources based on "your own" vehicle park with low and predictable costs and a more extensive "outer" supplier of vehicle resources where the pricing is assumed to be higher. Different alternative approaches have been tried and compared within the project.

12.3.2 Volume management

In the steering group of the ACOOR-project the representantives of NCC presented a volume management which occurs when constructing roads. A study of this problem seems as a promising candidate to test the techniques that have been developed in a rather different domain. We hope to come to a better understanding of this problem through discussions with NCC.

12.4 Transport service production

12.4.1 Trackbound transports

Trackbound transports contain many more problems than those we have specifically worked with in our projects so far. One example is flow optimization of wagon resources where OR-techniques have been succesfully applied [KL96], [Löb98], [Job98]. Other problems are for instance to generate trip specifications given a statistical flow measure. These problems probably differ between transportation of goods and of people. Such analyses can be used in biddings.

12.4.2 Road transports

So called travelling problems are among the hardest and at the same time best studied areas within the theory for algorithms. These problems are in general very hard to solve but many specific problems have been succesfully been approached with both OR- and with constraint techniques, as well as with local search and with specialized heuristics. Examples are allocation of vehicle resources to perform given point-to-point-transports from one or more depots. If the problem also contains the determining of exact times within fixed time windows the problem is technically much like the engine rostering problem in ACOOR.

12.4.3 Common person transportation systems

Similar problems occur in planning of vehiche and staff resources for public transport systems such as subways, trams and busses. Also these problems are in their pure form well studied within OR but also here there are sometimes complex side constraints and co-ordination problems that OR-techniques are badly suited to deal with.

12.4.4 Air traffic planning

In planning of air traffic optimising systems are used to a relatively high degree in order to cover the flying vehicle and staff resources. The company Carmen Systems based in Gothenburg, Sweden, is a market leader in staff planning systems for airlines and they are also active in the railway industry. Other problems are less well researched, such as minimization of fuel costs given local wheather conditions and late decisions on the route.

12.4.5 Co-ordination of material flows in JIT systems

Co-ordination problems of a similar nature as those occurring in the railway industry occur also when several more or less independent production units shall co-ordinatre is the production of complex products. Such problems are well known from for instance the car industry. The problem is to as much as possible reduce the storage of expensive semi-products and components. The method to buy components from subcontractors and to hold minimal storages drastically reduces the cost of production but it also tends to increase the sensitivity of the systems to disturbances. A certain co-ordination of production plans is therefore often necessary to uphold security of delivery.

13 Conclusion

We have specified (loosely) the problems of scheduling and allocation in railway planning and outlined some considerations relevant to the design of an model with a coordination agent. We also discussed a methodology called *abstract constraint solving* and indicated its possible use in the scheduling and rostering problems.

14 Acknowledgements

The research reported in this paper is a part of a study of the train planning domain currently done in cooperation with the Swedish State Railways (SJ). One of our motivations behind this cooperation is the opportunity to evaluate the feasibility of constraint and constraint logic programming [Hen89] techniques for various problems in their domain. We would like to thank Jolanta Drott and her group at the Swedish State Railways (SJ) for presenting us with the problem and for good cooperation within the TUFF series of projects. The cooperation with the Mozart-Oz teams at SICS, Catholic University of Louvain and at DFKI and University of Saarbrücken and the SICStus/Quintus Prolog development team of SICS/Uppsala is invaluable.

References

[AB93] Aggoun, A., Beldiceanu, N. "Extending CHIP in order to Solve Complex Scheduling and Placement Problems". In *Mathematical Computer Modelling* 17(7): pp. 57-73, Pergamon Press Ltd. 1993.

[AHD97] E. Andersson, C. Hjorring, and J. Drott. Crew scheduling for railways with Carmen systems. Technical report, Carmen Systems AB, and Swedish State Railways, Stab Tågplanering, Stockholm, Sweden, Jul 1997.

[AC91] Applegate, D., Cook, W. "A Computational Study of the Job-Shop Scheduling Problem". In *ORSA Journal of computing*, 3(2): 149-156, 1991.

[Ass80] A. A. Assad. Models for rail transportation. *Transportation Research Part B*, 14:205–220, 1980.

[ABBG83] A. Assad, M. Ball, L. Bodin, and B. Golden. Routing and scheduling of vehicles and crews. *Computers and Operations Research*, 10(2):63–211, 1983.

[AMO93] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows.* Prentice Hall, 1993.

[KrArHoLi2000a] Aronsson M., Kreuger P., Lindblom S. and Holmberg P. "ACOOR Rapport 1 – TUFF: Systemöversikt och arkitektur". SICS Technical Report T2000/06 (in Swedish)

[Bak74] Baker, K.R. *Introduction to sequencing and scheduling* Wiley & Sons, 1974.

[BS86] E. K. Baker and J. R. Schaffer. Solution improvement heuristics for the vehicle routing and scheduling problem with time window constraints. *American Journal of Mathematical and Management Sciences*, 6:261–300, 1986.

[BN95] S. Balasubramanian and D. H. Norrie. A multi-agent intelligent design system integrating manufacturing and shop-floor control. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems*, San Francisco, California, 1995. AAAI Press/The MIT Press.

[BMMN95] M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors. *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science.* North-Holland, 1995.

[BP95] Baptiste, P., Pape, C.L. "A Theoretical and Experimental Comparison of Constraint Propagation Techniques for Disjunctive Scheduling". In the *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* Montreal, Quebec, pp. 400-606, 1995.

[BLPN95] P. Baptiste, C. Le Pape, and W. Nuijten. Incorporating efficient operations research algorithms in constraint based scheduling. In *Proceedings of the first international joint workshop on artificial intelligence and operations research*, Timberline Lodge, Oregon, 1995.

[BB94] P. Barth and A. Bockmayr. Finite domain and cutting plane techniques in CLP(*PB*). Technical report, Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany, 1994.

[BB97] P. Barth and A. Bockmayr. PLAM ProLog and Algebraic Modeling. In *Proceedings of Practical Applications of Prolog*, 1997.

[Bar56] T. E. Bartlet. An algorithm for the minimum number of transport units to maintain a fixed schedule. *ORSA Journal of Computing*, 1956.

[BKC95] G. Baues, P. Kay, and P. Charlier. Constraint based resource allocation for airline crew management. Technical report, COSYTEC SA, 4, rue Jean Rostrand; F-91893 Orsay Cedex; France, 1995.

[BF94] J. C. Beck and M. S. Fox. Supply chain coordination via mediated constraint relaxation. In *Proceedings of the first Canadian workshop on distributed artificial intelligence*, May 15 1994.

[BKS94] C. Beckstein, G. Kraetzschmar, and Schneeberger. Distributed plan maintenance for scheduling and execution. In C. Bckstrm and E. Sandevall, editors, *Current Trends in AI Planning*, pages 74–86. IOS Press, 1994.

[Bel00] Beldiceanu, N. Global constraints as graph properties on structured networks of elementary constraints of the same type. Research R:00:01, SICS, 02 2000.

[BC94] Beldiceanu, N., Contejean, E. "Introducing Global Constraints in CHIP". In *Mathematical Computer Modelling* 20(12): 97–123, Pergamon Press Ltd. 1994.

[BL89] A. S. Belen'kij and E. V. Levner. Scheduling models and methods in optimal freight transportation planning. *Automat. Remote Control*, 50(1):1–56, 1989.

[BB74] E. Beltrami and L Bodin. Networks and vehicle routing for municipal waste collection. *Networks*, 4:65–94, 1974.

[BCG87] A. A. Bertossi, P. Carraresi, and G. Gallo. On some matching problems arising in vehicle scheduling models. *Networks*, 17:271–281, 1987.

[BvR93] D.J. Bertsimas and G. van Ryzin. Stochastic and dynamic vehicle routing in the euclidean plane with multiple capacitated vehicles. *Operations Research*, 41(1):60–76, 1993.

[BRR88] Bianco, L., Ricciardelli, S., Rinaldi, G., Sassano, A. "Scheduling tasks with sequence-dependent processing times". In *Naval Research Logistics* 35:177–184, 1988.

[BG81] L. Bodin and B. Golden. Classification in vehicle routing and scheduling. *Networks*, 11(97–108), 1981.

[BDD99] P. Boizumault, P. David, and H. Djelab. A repair algorithm for allocating resources in mobile telephone network. In *Proceedings of the conference on Practical Applications of Constraint Technology and Logic Programming (PACLP)*, 1999.

[BH80] U. Bokinge and D. Hasselström. Improved vehicle scheduling in public transport through systematic changes in the time-table. *European Journal of Operational Research*, 5:388–395, 1980.

[BT96] P. Brucker and O. Thiele. "A branch & bound method for the general-shop problem with sequence dependent setup-times". In *OR Spektrum*, 18:145-161, 1996.

[BrLiNoNi96] Brännlund U., Lindberg P.O., Nõu A., Nilsson J.E. "Railway Timetabling Using Lagrangian Relaxation", November 1996

[BuWiZi97] Bussieck M.R., Winter T., Zimmermann U.T. "Discrete Optimization in Public Rail Transport". In *Mathematical Programming* 79 (1997) 415-444

[CTV+97] A. Caprara, P. Toth, D. Vigo, M. Fischetti, and Guida P. L. Algorithms for railway crew management. *Mathematical Programming*, 79:125–141, 1997.

[CP89] Carlier, J., Pinson, E. "An Algorithm for Solving the Job-Shop Scheduling Problem". In *Management Science* 35(2): 164-176, 1989.

[CP90] Carlier, J., Pinson, E. "A Practical Use of Jackson's Preemptive Schedule for Solving the Job-Shop Problem". In *Annals of Operations Research*, 26: 269-287, 1990.

[CP94] Carlier, J., Pinson, E. "Adjustments of Heads and Tails for the Job-Shop Scheduling Problem". In the *European Journal of Operational Research*, 78:146-161, 1994.

[TACIT] Carlsson, M., Eriksson, H., Kreuger, P., Åström, E. *TACIT Project deliverable D2.2: Modelling of Ovako Steel Application*, (SICS), 1997.

[CG84] P. Carraresi and G. Gallo. Network models for vehicle and crew scheduling. *European Journal of Operational Research*, 16:139–151, 1984.

[Cas97] Y. Caseau. Using constraint propagation for complex scheduling problems: managing size, complex resources and travel. In G. Smolka, editor, *CP'97 – Principles and Practice of Constraint Programming*, volume 1330 of *LNCS*, Linz, Austria, Oct/Nov 1997. Springer-Verlag.

[CL95] Caseau, Y, Laburthe, F. "Improved CLP Scheduling with task intervals". In Van Hentenryck, P. (ed.) *Logic Programming, Proceedings of the Eleventh International Conference on Logic Programming.* Santa Margherita Ligure, Italy, MIT Press 1994. `http://www.ens.fr/ laburthe/papers.html#iclp94`

[CK92] Y. Caseau and P. Kopperstein. A cooperative-architecture expert system for solving large time/travel assignment problems. In *Proceedings of DEXA'92 (International Conference on Databases and Expert Systems Applications)*, 1992.

[CL94] Y. Caseau and F. Laburthe. Improved CLP scheduling with task intervals. In P. van Hentenryck, editor, *Proceedings of the eleventh International Conference on Logic Programming*, volume 78, Santa Margherita Ligure, Italy, 1994. MIT Press.

[CL95a] Y. Caseau and F. Laburthe. Cumulative scheduling with task intervals. Technical report, Laboratoire d'Informatique de l'Ecole Normale Supérieure LIENS, Département de Mathématiques ed d'Informatique, 45 rue d'Ulm, 75232 Paris Cedex 05, France, 1995.

[CL95b] Y. Caseau and F. Laburthe. Disjunctive scheduling with task intervals. Technical Report 95-25, Laboratoire d'Informatique de l'Ecole Normale Supérieure LIENS, Département de Mathématiques ed d'Informatique, 45 rue d'Ulm, 75232 Paris Cedex 05, France, 1995.

[CL96] Y. Caseau and F. Laburthe. Improving branch and bound for jobshop scheduling with constraint propagation. Technical report, Laboratoire d'Informatique de l'Ecole Normale Supérieure LIENS, Département de Mathématiques ed d'Informatique, 45 rue d'Ulm, 75232 Paris Cedex 05, France, 1996.

[CL97] Y. Caseau and F Laburthe. Solving small TSP with constraints. In L. Naish, editor, *Proceedings of the Joint International Conference and Symposium on Logic Programming*. MIT Press, 1997.

[paropt] Censor Y., Zenios, A.S. *Parallel Optimization. Theory, Algorithms and Applications*. Oxford University Press. ISBN 0-19-510062-X

[Čer82] V. Černy. A termodynamical approach to the traveling salesman problem: An efficient simulation algorithm. Preprint, Institute of Physics and Biophysics, Comenius University, Bratislava, 1982. See also J. of Optim. Theory Appl. 45 (1985) 41–51.

[CFGG95] A. Chamard, A. Fischler, S.-B. Guinaudeau, and A. Guillaud. CHIC Lessons on CLP Methodology. Technical report, Project deliverable D2.1.2.3 in Esprit Project: EP5291, Constraint Handling in Industry and Commerce (CHIC), Jan 1995.

[CLW96] Cheng, A B.M.W., Lee, J.H.M. and Wu, J.C.K. Speeding up constraint propagation by redundant modeling. In *Second International Conference on Principles and Practice of Constraint Programming CP'96*, volume 1118 of *LNCS*, pages 91–103, Cambridge, Massachusetts, USA, Aug 1996. Springer-Verlag. Available in `http://www.cse.cuhk.edu.hk/ isl/dPub2.html#Li`.

[CHI98] CHIC-II partners. Homepage for the chic-ii project with on line report archive. Web, 1998.

[CCL+96] C. K. Chiu, C. M. Chou, J. H. M. Lee, H. F. Leung, and Y. W. Leung. A constraint-based interactive train rescheduling tool. pages 104–118, 1996.

[CFN95] B. Y. Choueiry, B. Faltings, and G. Noubir. Abstraction methods for resource allocation. Technical report, Swiss Federal Institute of Technology (EPFL), IN-Ecublens, CH-1015 Lausanne, Switzerland, 1995.

[CF94a] B. Y. Choueiry and B. Faltings. Interactive resource allocation by problem decomposition and temporal abstraction. In C Bckstrm and E. Sandevall, editors, *Current trends in AI Planning: EWSP'93-2:nd European Workshop on Planning*, Frontiers in AI and Applications. IOS Press, Amsterdam, The Netherlands, 1994.

[CF94b] B. Y. Choueiry and B. Faltings. Interactive resource allocation by problem decomposition and temporal abstraction (long version). Technical Report TR-94/43, Laboratoire d'Intelligence Artificielle, Départment d'Informatique, EPFL, Ecublens, CH-1015 Lausanne, Switzerland, Jan 1994.

[ChStKaAs94] Christodoulou N., Stefanitsis E., Kaltsas E., Assimakopoulos V. "A Constraint Logic Programming Approach to the Vehicle-Fleet Scheduling Problem". In *Proc. of Practical Applications of Prolog PAP'94*

[CCC95] J. Chu-Carrol and S. Carberry. Communication for conflict resolution in multi-agent collaborative planning. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems*, San Francisco, California, 1995. AAAI Press/The MIT Press.

[CSKA94] N. Christodoulou, D. Stefanitsis, E. Kaltsas, and V. Assimakopoulos. A constraint logic programming approach to the vehicle-fleet scheduling problem. In *Proceedings of Practical Applications of Prolog*, 1994.

[Coh95] P. R. Cohen. Communicative actions for artificial intelligence. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems*, San Francisco, California, 1995. AAAI Press/The MIT Press.

[surveyrail] Cordeau J.-F., Toth, P. and Vigo D. *A Survey of Optimization Models for Train Routing and Scheduling.* February 23, 1998. *to be published in Transportation Science*, Nov? 1998.

[CGL95] J.-F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. publication 95-76, Centre de recherche sur les transports, 1995.

[Cousot] Cousot, P. "Draft Program Analysis Position Statement: The Abstract Interpretation Perspective". In *ACM Workshop on Strategic Directions in Computing Research*, MIT Laboratory for Computer Science, Cambridge, Mass. USA June 14-15, 1966

[Cr95] Crabtree I.B. "Resource Scheduling - comparing simulated annealing with constraint programming". In *Practical Applications of Constraint Technology PACT'95*

[Cra87] T. G. Crainic. Rail tactical planning: Issues, models and tools. In L. Bianco and A. La Bella, editors, *Freight Transport Planning and Logistics*, volume 317 of *Lecture Notes in Economics and Mathematical Systems*, pages 463–509. Springer, 1987.

[Cra96] J. M. Crawford. An approach to resource constrained project scheduling. Technical report, CIRL, 1269 University of Oregon, Eugene, OR 97403-1269, 1996.

[CB94] J. M. Crawford and A. B. Baker. Experimental results on the applications of satisfiability algorithms to scheduling problems. Technical report, Computational Intelligence Research Laboratory, 1269 University of Oregon, Eugene, OR 97403-1269, 1994.

[DLMM95] Dalfiume A., Lamma E., Mello P., Milano M. "A Constraint Logic Programming Application to a Distributed Train Scheduling Problem". In *Proceedings of the conference on the practical applications of prolog*, pages 163–182, 1995.

[DaCh95] David J.-M., Chew T.-L. "Constraint-based applications in production planning; examples from the automotive industry". In *Practical Applications of Constraint Technology PACT 95*

[DaSm83] Davis R., Smith R.G. "Negotiation as a Metaphor for Distributed Problem Solving". In *Journal of Artificial Intelligence* 20 (1983) 63–109

[DT94] M. Del'Amico and M. Trubian. Applying tabu search to the job-shop scheduling problem. *Annals of Operations Research*, 41:231–252, 1994.

[DDS90] M. Desrochers, J. Desrosiers, and M. Solomon. Using column generation to solve the vehicle routing problem with time windows. *Operations Research*, 18(5):411–419, 1990.

[DDS92] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operations Research*, 40(2):342–354, 1992.

[DSD84] J. Desrosiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14:545–565, 1984.

[DDSS95] J. Desrosiers, Y. Dumas, M. M. Solomon, and F. Soumis. *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, chapter Time Constrained Routing and Scheduling, pages 35–139. North-Holland, 1995.

[DH98] J. Drott and I. Hermansson. Sj:s integrerade produktionsprocess. Technical report, PROFS-projektet, SJ, Aug 18 1998.

[DHKK97] J. Drott, E. Hasselberg, N. Kohl, and M. Kremer. A planning system for locomotive scheduling. Technical report, Swedish State Railways, Stab Tågplanering, Stockholm, Sweden, and Carmen Systems AB, Jul 1997.

[Dun95] T. Duncan. Neighborhood search and the vehicle routing problem. Technical Report AIAI-TR-189, AIAI, The University of Edinburgh, 80 South Bridge, Edinburgh EH1 1HN, Scotland UK, 1995.

[dVT91] F. du Verdier and J. P. Tsang. Un raisonnement spatial par propagation de constraintes. In *11iemes Journees Internationales: Les Systemes Experts et leurs Applications*, pages 297–314, Avignon, 1991.

[EFW98] P. S. Eaton, E. C. Freuder, and R. J. Wallace. Constraints and agents — confronting ignorance. *AI magazine*, Summer 1998.

[FMP95a] K. Fisher, J. P. Müller, and M. Pischel. Cooperative transport scheduling: an applications domain for DAI. Technical Report RR–95–01, German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, 1995.

[FMP95b] K. Fisher, J. P. Müller, and M. Pischel. Unified control in a layered agent architecture. Technical Report RR–95–05, German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, 1995.

[FiMuPiSch95] Fischer K., Müller J.P., Pischel M., Schier D. *A Model For Cooperative Transportation Scheduling* German Center for Artificial Intelligence (DFKI), Saarbrücken, Tyskland. (1995)

[FMPS95] K. Fisher, J. P. Müller, M. Pischel, and D. Schier. A model for cooperative transportation scheduling. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems*, San Francisco, California, 1995. AAAI Press/The MIT Press.

[FBF+76] M. Florian, G. Bushell, J. Ferland, G. Guerin, and L. Nastansky. The engine scheduling problem in a railway network. *INFOR*, 14(2):121–138, June 1976.

[FHW91] M. A. Forbes, J. N. Holt, and A. M. Watts. Exact solution of locomotive scheduling problems. *Journal of the Operational Research Society*, 42(10):825–831, 1991.

[FHW94] M. A. Forbes, J. N. Holt, and A. M. Watts. An exact algorithm for multiple depot bus scheduling. *European Journal of Operational Research*, 72:115–124, 1994.

[FS90] M. S. Fox and N. Sadeh. Why is scheduling difficult? In *Proceedings of the European Conference on Artificial Intelligence*, pages 754–765, 1990.

[FBH98] Nils Franzén, Per Brand, and Seif Haridi. Dynamic topology. Technical Report NUTEK's program for complex technical systems (KTS), Projekt Distributed Management Systems, Nr: P10-537-1, SICS/NUTEK, 1998.

[Fra97] T. Franzén. Personal communication. SJ, Aug 1997.

[Fra98] T. Franzén. Personal communication. SJ, Oct 1998.

[FrSa95] Fromherz M., Saraswat V. "Model-Based Computing: Constricting Constraint-Based Software for Electro-Mechanical Systems (extended abstract)". In *Practical Applications of Constraint Technology PACT 95*

[FB97] Tom Frühwirth and P. Brisset. Optimal planning of digital cordless telecommunications. In *Proceedings of the conference of Practical Applications of Constraint Technology (PACT)*, 1997.

[Ger97] C. Gervet. Ongoing research in chic-2: Decision making under uncertainty. In *ERCIM Workshop on Constraints, in conjunction with CP'97*, 1997.

[Glo89] Glover, F. "Tabu search - Part I" In *ORSA Journal of Computing*, 1:190-206, 1989.

[GK94] M. R. Genesereth and S. P. Ketchpel. Software agents. *Communications of the ACM*, 37(7):48–53, 1994.

[Glo89] F. Glover. Tabu search — Part 1. *ORSA Journal of computing*, 1:190–206, 1989.

[GA86] B. L. Golden and A. A. Assad. Vehicle routing with time-window constraints: Algorithmic solutions. *American Journal of Mathematical and Management Sciences*, 6, 1986.

[GJ96] H.-J. Goltz and U. John. Methods for solving practical problems of job-shop scheduling modeled in clp(*FD*). In *Proceedings of Practical Applications of Constraint Technology (PACT'96).* PA, Apr 1996.

[Gos93] V. Gosselin. Train scheduling using constraint programming techniques. In *13th conference on AI, expert systems and natural language*, Avignon, 1993.

[Gu92] J. Gu. Efficient local search for very large-scale satisfiability problems. *SigArt Bulletin*, 3(1):8–12, 1992.

[HaCh93] Hammer M., Champy J. *Reengineering the Corporation* HarperBusiness, 1993

[Ham96] J. R. Hammerton. Delivering scheduling solutions for a changing business word. In *Computers in Railways V, Volume 1: Railway Systems and management*, pages 467–477. Computational Mechanics Publications, COMRECO Rail Ltd, UK (TrainPlan), 1996.

[Mozart] Haridi, S., Van Roy, P., Brand, P., Schulte, C. “Programming Languages for Distributed Applications”. Invited paper in *New Generation Computing*, Vol. 16, No.3, pp 223-261, 1998. Ohmsa Ltd. and Springer Verlag., Tokyo.

[HG95] W. D. Harvey and M. L. Ginsberg. Limited discrepancy search. In *Proceedings of the International joint conference on artificial intelligence*, pages 607–613, 1995.

[Her95] Herold A. “Experiences in Constraint Programming from the CHIC User Group”. In *Practical Applications of Constraint Technology PACT 95*

[Holm2000] Holmberg P. “The Use of Abstractions to Solve Large Scheduling Problems”. Examensarbete, Dept. of Teleinformatics, KTH Kista, Sweden Examiner: Haridi S. Advisor: Kreuger P.

[HJL96] K. Holmberg, M. Joborn, and J. T. Lundgren. A model for distribution of empty freight cars. Technical Report LiTH-MAT-R-1996-11, Department of Mathematics, Lindköping University, SE-581 83, Linköping, Sweden, May 1996.

[HMZM96] R. C. Holte, T. Mkadmi, R. M. Zimmer, and A. J. MacDonald. Speeding up problem solving by abstraction: a graph oriented approach. *Artificial Intelligence*, 85:321–361, 1996.

[HPZ95] R. C. Holte, M. B. Perez, and A.J. Zimmer, R. M. MacDonald. Hierarchical a*: Searching abstraction hierarchies efficiently. 1995.

[HLV97] J. A. Hoogeveen, J. K. Lenstra, and S. L. Velde. Sequencing and scheduling: An annoted bibliography. Technical report, Department of Mathematics and Computing Science, Eindhoven University of Tech, 1997.

[HS] M. Huhns and M. Singh. *Global Information Management via Local Autonomous Agents.* Microelectronics and Computer Technology Corporation,Research Division, 3500 West Balcones Center Drive, Austin, TX 78759-5398.

[Huh98] M. N. Huhns. Decentralized logistics using autonomous agent technology. Technical report, Center for Information Technology, College of Engineering, University of South Carolina, 1998.

[IsNi97] Isacson G., Nilsson J.-E. “An Experimental Comparison of First- and Second- Price Auctions - The Case of Track Capacity Allocation”. In *Fifth International Conference on Competition and Ownership in Land Passenger Transport*, Leeds, England, maj 1997.

[Job95] Joborn M. *Empty Freight Car Distribution at Swedish Railways - Analysis and Optimization Modeling.* Lic. Thesis Division of Optimization, Dept. of Mathematics, Linköping University, 1995.

[JD95] Jordan, C., Drex, A.L. “A comparison of constraint and mixed-integer programming solvers for batch sequencing with sequence-dependent setups” In *ORSA Journal on Computing* 7: 160-165, 1995.

[JIH95] W. K. Jackson, M. Irgens, and W. S. Havens. A re-examination of limited discrepancy search. Technical report, Intelligent Systems Lab, Centre for Systems Science, Simon Fraser University, Burnaby, B.C., Canada V5A 1S6, 1995.

[Job98] M. Joborn. Empty freight car distribution at swedish state railways. Lic. thesis, Department of Mathematics, University of Lindköping, Sweden, 1998.

[Joh96] T. Johansson. Riktlinjer för tidtabellsarbetet. Internal Report SJF 613, Utgåva 4, SJ Stab Tågplanering, 920 - 42 10, +46 31 10 42 10, May 1996.

[Kab95] F. Kabanza. Synchronizing multi-agent plans using temporal logic specifications. In V. Lesser, editor, *Proceedings of the First International Conference on Multi-Agent Systems*, San Francisco, California, 1995. AAAI Press/The MIT Press.

[Ken78] J.L. Kennington. A survey of linear cost multi-commodity network flows. *Operations Research*, 26(2):209–236, 1978.

[KG94] T. Khedro and M. R. Genesereth. Modeling multi-agent cooperation as distributed constraint satisfaction problem solving. In A. Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 249–253. John Wiley & Sons, Ltd, 1994.

[KGV83] Kirkpatrick, S., Gelatt, C., Vecchi M. "Optimization by Simulated Annealing". In *Science*, 220:671-680, 1983.

[KL96] A. Kokott and A. Löbel. Lagrangian relaxations and sub-gradient methods for multiple-vehicle scheduling problems. Preprint SC 96-22, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, 1996.

[Kra95] P.T. Kraay, D. R.and Harker. Real-time scheduling of freight railroads. *Transportation Research Part B*, 29(3):213–229, 1995.

[KrArHoLi2000b] Kreuger P., Aronsson M., Holmberg P. and Lindblom S. "ACOOR Rapport 2 – Översikt av tekniker och metoder". SICS Technical Report T2000/07 (in Swedish)

[KrCaOlSjAs97a] Kreuger P., Carlsson M., Olsson J., Sjöland T., Åström E. "The TUFF Train Scheduler". In proc. *Workshop on Constraint Environments* of International Logic Programming Symposium 1997, NY, USA. G. Puebla (coord.)

[KCO+97] P. Kreuger, M. Carlsson, J. Olsson, T. Sjöland, and E. Åström. The tuff train scheduler – Trip scheduling on single track networks. In *The proceedings of the Workshop on Industrial Constraint-Directed Scheduling held at the Third International Conference on Principles and Practice of Constraint Programming*, Schloss Hagenberg, Linz, Austria, 1997. Ed. A. Davenport.

[KRZ97] L. Kroon, H. Romeijn, and P. J. Zwaneveld. Routing trains through railway stations: complexity issue. *European Journal of Operational Research*, 98:485–498, 1997.

[KZ] L. G. Kroon and P. J. Zwaneveld. Routing trains through railway stations including shunting decisions. pages 438–443.

[Kum92] V. Kumar. Algorithms for constraint-satisfaction problems: A survey. *AI Magazine*, 1, 1992.

[Lec96] Leconte, M. "A bounds-based reduction scheme for constraints of difference". In *Constraint-96, Second International Workshop on Constraint-based Reasoning*, Key West, Florida, 1996.

[Lec96] M. Leconte. A bounds-based reduction scheme for constraints of difference. In *Proceedings of Second international workshop on constraint-based reasoning*, Key West, Florida, 1996.

[LLP97] C. Y. Lee, L. Lei, and M. Pinedo. Scheduling: Theory and applications. *Annals of Operations Research*, 70, 1997.

[LB94] J. Lenting and P. Braspenning. An all-pay auction approach to reallocation. In A. Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 249–253. John Wiley & Sons, Ltd, 1994.

[LM72] Lockett, A.G., Muhlemann, A.P. "A scheduling problem involving sequence dependent changeover times". In *Operations Research* 20: 895-902, 1972.

[LMM97] E. Lamma, P. Mello, and M. Milano. A distributed constraint-based scheduler. *Artificial Intelligence in Engineering*, 11:91–105, 1997.

[Löb96] A. Löbel. Solving the LP relaxations of large-scale multiple-depot vehicle scheduling problems exactly. Preprint SC 96-26, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, Dec 1996. A revised version exists as Löb97b.

[Löb97a] A. Löbel. Solving large-scale multiple-depot vehicle scheduling problems. Technical Report SC 97-17, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, Mar 1997.

[Löb97b] A. Löbel. Vehicle scheduling in public transit and lagrangean pricing. Preprint SC 96-26, Konrad-Zuse-Zentrum für Informationstechnik, Berlin, Dec 1997. This is a revised version of Löb96.

[Löb98] A. Löbel. *Optimal Vehicle Scheduling in Public Transit*. PhD thesis, TU Berlin, 1998. Shaker-Verlag, Aachen.

[LP83] T. Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, C–32(2):108–120, 1983.

[LP90] C. Le Pape. A combination of centralized and distributed methods for multi-agent planning and scheduling. *IEEE*, pages 488–493, 1990.

[LP94] C. Le Pape. Implementation of resource constraints in ILOG SCHEDULE: A library for the development of constraint-based scheduling systems. *Intelligent Systems Engineering*, 3(2):55–66, 1994.

[Mal98] C. Malm. Personal communication. Swedish State Railway, Feb 1998.

[Ma92] Martin C. *Logistics and Supply Chain Management* Financial Times Pitman Publishing, 1992

[Meh84] K. Mehlhorn. *Multi-dimensional Searching and Computational Geometry*, volume 3, pages 26–37. Springer-Verlag, 1984.

[Min84] M Minoux. A polynomial algorithm for minimum quadratic cost flow problems. *European Journal of operations research*, 18:377–387, 1984.

[Min89] M. Minoux. Network synthesis and optimum network design problems: Models, solution methods and applications. *Networks*, 19:313–360, 1989.

[MM98] E. M. Morgado and J. P. Martins. CREWS_NS – scheduling train crews in the Netherlands. *AI Magazine*, pages 25–38, Spring 1998.

[MuPoSchWu95] Müller M., Popov K., Schulte C., Würtz J. *Constraint Programming in Oz.* DFKI, Saarbrücken, Tyskland, 1995

[MuWu95] Müller M., Würtz J. *Finite Domain Programming in Oz.* DFKI, Saarbrücken, Tyskland, 1995.

[MP94] J. P. Müller and M. Pischel. An architecture for dynamically interacting agents. *International Journal of Intelligent and Cooperative Informations Systems*, 3(1):25–45, 1994.

[MS94] P. Martin and D. B. Shmoys. A new approach to computing optimal schedules for the job-shop scheduling problem. pages 389–403, 1994.

[MW84] T.L. Magnanti and R.T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55, 1984.

[MW97] T. Müller and J. Würtz. *The Constraint Propagator Interface of DFKI Oz.* German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, Dec 1997.

[OlKrErSjAs96] Olsson J., Krueger P., Eriksson H., Sjöland T., and Åström E. *TUFF förstudie - Tågplanering För Framtiden.* KTM Project Report 96.

[OlRoWe97] Olve N.-G., Roy J., Wetter M. *Balanced Scorecard*, Liber Ekonomi, 1997.

[PYF+92] M. Palaniappan, N. Yankelovich, G. Fitzmaurice, A. Loomis, B. Haan, J. Coombs, and N. Meyrowitz. The envoy framework: An open architecture for agents. *ACM Transactions on Information Systems*, 10(3):233–264, 1992.

[PS82] Papadimitriou, C.H., Steiglitz K. *Combinatorial Optimization: Algorithms and Complexity* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[PG96] G. Pesant and M Gendreau. A view of local search in constraint programming. In *Second International Conference on Principles and Practice of Constraint Programming CP'96*, volume 1118 of *LNCS*, pages 91–103, Cambridge, Massachusetts, USA, Aug 1996. Springer-Verlag.

[PGPR] G. Pesant, M. Gendreau, J.-Y. Potvin, and J.-M. Rousseau. An exact logic programming algorithm for the traveling salesman problem with time windows. *To appear in Transportation Science.*

[PGR97] G. Pesant, M. Gendreau, and J.-M. Rousseau. GENIUS-CP: A generic single-vehicle routing algorithm. In G. Smolka, editor, *CP'97 – Principles and Practice of Constraint Programming*, volume 1330 of *LNCS*, Linz, Austria, Oct/Nov 1997. Springer-Verlag.

[PS95] A. Podelski and G. Smolka. Operational semantics of constraint logic programs with coroutining. In *Proceedings of the International Conference of Logic Programming (ICLP'95)*, Kanagawa, Japan, Jun 1995. MIT Press.

[PR93] J.-Y. Potvin and J.-M. Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66:331–340, 1993.

[PR95] J.-Y. Potvin and J.-M. Rousseau. An exchange heuristic for routing problems with time windows. *Journal of the Operational Research Society*, 46(12):1433–1446, 1995.

[PJO95] W. B. Powell, P. Jaillet, and A. Odoni. *Network Routing*, volume 8 of *Handbooks in Operations Research and Management Science*, chapter Stochastic and Dynamic Networks and Routing, pages 141–295. North-Holland, 1995.

[PS85] F. P. Preparata and M. I. Shamos. *Computational Geometry*, chapter 2, pages 72–77 and 351. Springer-Verlag, 1985.

[PL95] Puget, J.-F., Leconte, M. "Beyond the Glass Box: Constraints as Objects". In Lloyd, J. (ed.) *Proceedings of International Logic Programming Symposium (ILPS95)* pp. 513-527, Portland 1995.

[Pug92] J.-F. Puget. *Object Oriented Constraint Programming for Transportation Problems.* ILOG SA, Gentilly Cedex, France, 1992.

[RG76] A. S. Rao and M. P. Georgeff. Embedded multi-agent and distributed reasoning research at the australian artificial intelligence institute. *AISBQ*, pages 25–27, 1976.

[Ree95a] C. Reeves, editor. *Modern Heuristic Techniques for Combinatorial Optimization.* McGraw-Hill International (UK) Ltd., 1995.

[Ree95b] C. Reeves, editor. *Modern Heuristic Techniques for Combinatorial Optimization*, chapter Chapter 6: Lagrangean Relaxation. McGraw-Hill International (UK) Ltd., 1995.

[Reg94] Régin, J.C. "A filtering algorithm for constraints of difference in CSPs" In *AAAI-94, Proceedings of the Twelfth National Conference on Artificial Intelligence.* 1994.

[RS94] C. C. Ribeiro and F. Soumis. A columns generation approach to the multiple-depot vehicle scheduling problem. *Operations Research*, 42(1):41–52, 1994.

[Ri66] Risberg, T. "Simuleringsteknik på Spåret". In *Modern Datateknik* 1966:3, s29–33

[Ros84] J. B. Rosenberg. Geographical data structures compared: A study of data structures supporting region queries. *IEEE Transactions on Computer-Aided Design*, 4(1), January 1984.

[RSL77] D. Rosenkrantz, R. Sterns, and P. Lewis. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal of Computing*, 6:563–581, 1977.

[Sam89] H. Samet. *The Design and Analysis of Spatial Data Structures*, chapter 3, pages 200–213. Addison-Wesley, 1989.

[Sav85] M. W. P. Savelsbergh. Local search in routing problems with time windows. *Annals of Operations Research*, 4:285–305, 1985.

[Sch98a] V. Scholtz. Knowledge-based locomotive planning for the swedish railway. Master's thesis, Institut für Informatik, Universität Stuttgart and Swedish Institute of Computer Science (SICS), Nov 1998.

[Sch96] Schulte C. *Open Programming in Oz.* DFKI , Saarbrücken, Tyskland, 1996.

[Sch98b] C Schulte. *Finite Domain Constraint Programming in Oz: A Tutorial.* German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, Feb 1998.

[SK92] Selman, B., Kautz, H. "An empirical study of Greedy Local Search for Satisfiability Problems" In *Proc. AAAI-92*, pp. 440-446, 1992.

[Sim95] Simonis, H. *Modelling Machine Set-up Time in CHIP* Technical Report COSY/TR, 1995.

[Si96] Simonis H. "A Problem Classification Scheme for Finite Domain Constraint Solving". In *CP'96 applications workshop*, COSYTEC SA, Orsay, France.

[Sib] Simonis H. *Calculating Lower Bounds on a Resource Scheduling Problem.* COSYTEC SA, Orsay, France.

[SC95] H. Simonis and T. Cornelissens. Modeling producer/consumer constraints. Technical report, COSYTEC SA, 4, rue Jean Rostrand; F-91893 Orsay Cedex; France, 1995.

[Sid] Simonis H. *The Use of Exclusion Constraints to Handle Location Continuity Conditions.* COSYTEC SA, Orsay, France.

[Sie] Simonis H. *Modeling Machine Set-up Time in CHIP.* COSYTEC SA, Orsay, France.

[SCK96] H. Simonis, P. Charlier, and P. Kay. TACT — an integrated transportation problem solved with CHIP. Technical report, COSYTEC SA, 4, rue Jean Rostrand; F-91893 Orsay Cedex; France, 1996.

[Sim95a] H. Simonis. A problem classification scheme for finite domain constraint solving. Technical report, COSYTEC SA, 4, rue Jean Rostrand; F-91893 Orsay Cedex; France, 1995.

[Sim95b] H. Simonis. The use of exclusion constraints to handle location continuity. Technical report, COSYTEC SA, 4, rue Jean Rostrand; F-91893 Orsay Cedex; France, 3 1995.

[Sim96a] H. Simonis. Calculating lower bounds on a resource scheduling problem. In *Proceedings of the workshop on applications at the international conference of Constraint programming CP'96*, 1996.

[Sim96b] H. Simonis. Constraint logic programming. Technical report, COSYTEC SA, 1996.

[Sim96c] H. Simonis. Modeling machine set-up time in CHIP. In *Proceedings of the workshop on applications at the international conference of Constraint programming CP'96*, 1996.

[Sim96d] H. Simonis. Tutorial on scheduling and planning with constraint programming. given at University of Uppsala, Feb 1996.

[SJ98] Personal communication with Christer Malm at the Swedish State Railways (SJ), February 1998.

[SjKrAsDa98] Sjöland T., Kreuger P., Åström E. and Danielsson P. "Coordination of Scheduling and Allocation Agents (extended abstract)". Presented in *Second Intnl. Workshop on Timed Concurrent Constraint Programming* 1998, Nice, France. F. de Boer and M. Gabbrielli (coord.)

[Sm80] Smith R.G. "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver". In *Readings In Artificial Intelligence*, pp357–366. (1980)

[Smo95b] Smolka G. *A Survey of Oz.* DFKI , Saarbrücken, Tyskland, 1995

[Smo95] Smolka G. "The Oz Programming Model" In van Leeuwen, J, ed. *Computer Science Today.* Lecture Notes in Computer Science volume 1000: 324-343. Springer Verlag, Berlin, 1995.

[ST95] Smolka, G., Treinen, R. eds. *DFKI Oz Documentation Series.* Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany 1995.

[So90] So, K.T. "Some heuristics for scheduling jobs on parallel machines with setups" In *Management Science* 36: 467-475, 1990.

[SD88] M. M. Solomon and J. Desrosiers. Time window constrained routing and scheduling problems. *Transportation Science*, 22(1):1–13, 1988.

[SG96] G. Solotorevsky and E. Gudes. "Algorithms for Solving Distributed Constraint Satisfaction Problems". In *Proceedings of AIPS-96*, 1996.

[SJ98] SJ Godstransportdivision, SE-10550 Stockholm. *Frakthandboken*, 1998.

[SK92] B. Selman and H. Kautz. An empirical study of greedy local search for satisfiability problems. In *Proceedings of the national conference on artificial Intelligence AAAI-92*, pages 440–446, 1992.

[SKÅD98] T. Sjöland, P. Kreuger, E. Åström, and P. Danielsson. Coordination of scheduling and allocation agents (extended abstract). In Frank de Boer and Maurizio Gabbrielli, editors, *Invited presentation in 2nd International Workshop on Constraint Programming for Time Critical Applications COTIC*, Nice, France, Aug 1998.

[Smi80] R. G. Smith. The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, C-29(12):1104–1113, 1980.

[Smo95] G. Smolka. The Oz programming model. In J. van Leeuwen, editor, *Computer science today*, volume 1000 of *Lecture notes in computer science*, pages 324–343. Springer verlag, 1995.

[So90] K. T. So. Some heuristics for scheduling jobs on parallel machines with setups. *Management Science*, 36:467–475, 1990.

[Sol86] M. M. Solomon. On the worst-case performance of some heuristics for the vehicle routing and scheduling problem with time window constraints. *Networks*, 16:161–174, 1986.

[Sol87] M. M. Solomon. Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research*, 35(2):254–265, March-April 1987.

[SRSF91] K. P. Sycara, S. F. Roth, N. Sadeh, and M. S. Fox. Resource allocation in distributed factory. *IEEE Expert*, Feb 1991.

[ST+95] G. Smolka, R. Treinen, et al. *DFKI Oz documentation series*. German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, 66123 Saarbrücken, Germany, 1995.

[Sun92] U. Sundin. Chapter 11 — assignment and scheduling. In *The CommonKADS Library*. 1992.

[TAT91] T. Tokuyama, T. Asano, and S. Tsukiyama. A dynamic algorithm for placing rectangles without overlapping. *Journal of Information Processing*, 14(1), 1991.

[TBG+95] E. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin. A new neighborhood structure for the vehicle routing problem with time windows. Publication CRT-95-66, Centre de recherche sur les transports, Université de Montréal, 1995.

[TP93] P.M. Thompson and H.N. Psaraftis. Cyclic transfer algorithms for multi-vehicle routing and scheduling problems. *Operations Research*, 41(5):935–946, 1993.

[Tsa93] E. Tsang. *Foundations of Constraint Satisfaction*. Academic Press, 1993.

[VAL95] R. J. M. Vaessens, E. H. L. Aarts, and J. K. Lenstra. A local search template. Technical report, 1995.

[VAL96] R. J. M. Vaessens, E. H. L. Aarts, and J. K. Lenstra. Job shop scheduling by local search. Technical report, 1996.

[Wa95] Wallace M. "Introducing the Practical Applications of Constraints". In *Practical Applications of Constraint Technology PACT 95*

[Wal97] Walser, J. "Solving Linear Pseudo-Boolean Constraint Problems with Local Search" In *Proc. AAAI-97, 1997.*

[Wal97] J. Walser. Solving linear pseudo-boolean constraint problems with local search. In *Proceedings of the national conference on artificial Intelligence AAAI-97*, 1997.

[vDP89] H. van Dyke Parunak. Manufacturing experience with the contract net. In L. Gasser and M. Huhns, editors, *Distributed Artificial Intelligence: Volume 2*. Morgan Kaufman Publishers, 1989.

[Hen89] Van Hentenryck, P. *Constraint Satisfaction in Logic Programming. Programming Logic Series.* The MIT Press, Cambridge, MA, 1989.

[vH89] P. van Hentenryck. *Constraint satisfaction in logic programming.* Programming logic series. The MIT press, Cambridge, MA, 1989.

[vH99] P. van Hentenryck. *The OPL Optimization Programming Language.* The MIT press, Cambridge, MA, 1999.

[VMT94b] Van Marcke K., Tubbax B. *SKAI: A Knowledge Based Environment for Scheduling Traction Equipment and Personnel.* KT Memo 94-2

[VMT94a] Van Marcke K. *Knowledge Based Scheduling with the SKAI Environment.* KT Memo 94-01

[VMT93] Van Marcke K. *Knowledge Based Scheduling: A Case Study.* KT Memo 93-3, Knowledge Technologies n.v.

[WW77] White, C.H., Wilson, R.C. "Sequence dependent set-up times and job sequencing" *International Journal of Production Research* 15, pp. 191-202, 1977.

[Wri89] M.B. Wright. Applying stochastic algorithms to a locomotive scheduling problem. *Journal of the Operational Research Society*, 40(2):187–192, 1989.

[Wu96] Würtz J. "Oz Scheduler: A workbench for Scheduling Problems". In *IEEE International conference on Tools with Artificial Intelligence (ICTAI'96)*, 1996.

[Wür97] J. Würtz. Constraint-based scheduling in Oz. In U. Zimmermann, U. Derigs, W. Gaul, R. Möhrig, and K.-P. Schuster, editors, *Operations Research Proceedings 1996*, pages 218–223. Springer-Verlag, Berlin, Heidelberg, New York, 1997.

[XSS92] Xiong, Y., Sadeh, N., Sycara, K. "Intelligent Backtracking Techniques for Job-Shop Scheduling" In the *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, 1992.

[XSS92] Y. Xiong, N Sadeh, and K. Sycara. Intelligent backtracking techniques for job-shop scheduling. In *Proceedings of the third international conference on principles of knowledge representation and reasoning*, 1992.

[Zho97] Zhou, J. "A Permutation-Based Approach for Solving the Job-Shop Problem", In *Constraints*, 2:185-213, 1997.

[Zho97] J. Zhou. A permutation-based approach for solving the job-shop problem. *Constraints*, 2:185–213, 1997.

[ZKA96] P. J. Zwaneveld, L. G. Kroon, and H. W. Ambergen. A decision support system for routing trains through railway stations. In *Computers in Railways NN: ?*, pages 217–226. Computational Mechanics Publications, 1996.

[ZKR+96] P. J. Zwaneveld, L. G. Kroon, H. E. Romeijn, M. Solomon, S. Dauzère-Pérès, S. P. M. van Hoesel, and H. W. Ambergen. Routing trains through railway stations: Model formulation and algorithms. *Transportation Science*, 30(3):181–194, 1996.