Heterogenous Scheduling and Rostering
(short technical paper)

M. Aronsson*, P. Kreuger and T. Sjöland
Swedish Institute of Computer Science, Box 1263, SE-164 29 Kista
(*) Industrilogik L4i AB, Box 21024, SE-100 31 Stockholm
January 2, 2001

Abstract

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 sub-problems 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.

Using constraints as the key technology, we are interested in finding interfacing principles to combine several solvers to allow users, for instance reasoning agents, to have a uniform view of the combined constraint store of heterogenous solvers.

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.

keywords Railway scheduling, rostering, abstract constraints

1 Scheduling and Allocation in Transportation

The problem we are interested in is diminishing the total cost for performing transport services in a railway net. Resources can in general model widely separated entities, for instance processing equipment in a production process, and in a transportation network track links, shunting yards, buffer areas, etc. We are in particular interested in the problem of coordinating the planning processes involved in time-tabling and allocation of resources in railway nets. The three most important resources there are *track positions*, *engines* and *traveling staff*.

A scheduling problem consists of a number of *tasks* utilizing resources. A task has restrictions on *start time*, *stop time* 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.

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 an assignment such that no limitations in resources are violated. This is in general a very difficult computational problem [FS90]. Nevertheless the many practical applications for methods in this area make it fairly well studied.

A constraint representation of the scheduling problem is generated from the plan as a set of finite domain constraints. We refer to [KrCaOlSjAs97a] or [KCO$^+$97] for more details of the finite domain constraint model used in our prototype scheduler.

1.1 Railway networks

A railway network is a graph where the nodes (*locations*) are places where significant operations can occur and the arcs are *tracks*. 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. 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. The arcs are either directed or undirected. *routes* are predefined paths in the network.

1.2 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.

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].

The scheduling problem that we consider can be concisely stated thus: 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 while maintaining reasonable bounds on waiting and makespan.

2 Our Project

Our software for scheduling trips 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.

One goal of our project is to enable a uniform access model of the information in a set of spaces which might use e.g. different constraint models. From the perspective of the user (or a control program) the different spaces should appear as a single one. Furthermore the access model should offer a common information model and a uniform language for access and modification of the spaces.

We currently build an interactive tool that contains specialized solvers for sub-problems and mechanisms to subdivide, abstract and refine problem specifications. The platform we use consists of SICStus Prolog for efficient constraint solvers and CPLEX for OR-based solvers and Mozart-Oz [Mozart] for coordination of the solvers.

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

2.1 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 and for stops at arbitrary places in the route of a train, and the specification of all the information required by the different planning components, including recurrent (for instance hourly, daily or weekly) trips.

Our network representation contains for each station a maximum number of trains allowed to be simultaneously waiting at the station. We express this limit as a cumulative constraint [AB93].

Trips model individual trains traversing the network. Suitable routes for a given trip are given by the user.

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* representing the minimum time distance between the departures of any two trips traversing a track in opposite directions and another duration called *headway*, 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.

A *plan* denotes a (set of) schedule(s) in terms of a given network and a specification of a required set of train trips. This specification allows limiting the arrival, departure and waiting durations for arbitrary locations in the route of a *trip*.

2.2 Agents, Concurrency and Distribution

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

By subdividing a planning system into possibly physically separated sub-problem 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].

Techniques to partition large complex problems into smaller and less composite ones where communication between sub-problem solvers is limited is discussed in [KrArHoLi2000b].

2.2.1 Interactivity and distribution

Planning of the complicated and large activities of a supplier of transport services must for many reasons use manual decisions at decisive points. A decision support system is not intended to replace the experts who make these decisions with machines, just to release them of the burden of routine decisions and to supply as good a decision background as possible.

It might be advantageous to decentralize and to distribute the planning process, but 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.

2.2.2 Negotiation Protocols

The agent modeling approach 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 sub-plans representing limitations, cost estimates etc.

2.3 Agent Model Based on Organisation

The agent model we currently consider divides the system as follows: The *service producers* generate train orders and negotiate contracts with customers, *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) and 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.

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 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, but with limited knowledge of the detailed models of the others, jointly build a working tactical plan.

3 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].

3.1 Abstract Constraint Solving for Scheduling

Abstraction of constraint models can be used also for efficiency. A particular technical idea 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 is upheld we guarantee that a solution in the abstract domain corresponds to some 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.

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].

3.2 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. When co-ordinating these resource problems the need arises to move between these different levels in a uniform and unambiguous way.

When dividing such a resource problem into smaller parts based for instance on 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 deal with 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", is simpler to solve than the problem that would have arisen from the original problem descriptions.

A third example is the case when the problem is simplified by 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 it is often desirable that the abstractions used meet certain properties, for instance that the existence of an abstract solution *guarantees* the existence of at least one concrete solution.

The approach we currently work with is one that builds on a concept of task that is hierarchically organised. Each primitive mission and job is represented as a *mission structure* with an unambiguously determined sub-structure. A problem structure is represented partly by such a mission structure, partly by a (partial) order relation on the individual tasks and their parts and finally partly by 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 it is in itself a problem structure.

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 the coordinator and the components for planning of vehicles and tracks respectively. 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.

3.3 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. 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 to use fewer time points in the intervals described by the finite domain variables.

A second suggestion is to 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.

We need to guarantee that the traversal time parameter for an arc in the abstract network covers the longest possible time between the two nodes in the concrete network. This is of course doable [1] but it might prohibit the finding of a schedule if the abstraction is applied on parts of the network where there is much traffic.

3.4 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.

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

[1] 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.

which is then translated into constraints for the time points in the coordination protocol.

3.5 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.

In order to make it convenient to use strategies we use a planning language. Such a language should offer 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.

4 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.

4.1 Process industry

In processing industries such as, for example, 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[TACIT], SICS in 1997-98 performed a study of the process flow containing furnaces. In cooperation with planning expertise from OVAKO, SICS developed a decision support system for this flow of materials. The tool reads data from the processing units and 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*[2] 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.

[2] 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.

4.2 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 of tools, textiles and vehicles[DaCh95] etc.

4.3 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 pointing to how computer technology can be used to significantly improve the information management and calculations 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, weather 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). Early on in the project SICS 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.

During the period 1988 to 1995 SICS has developed demonstration software for conferences and for industrial and governmental parties and given seminars.

The French construction company Bouygues is involving some of the top researchers in constraint programming [CK92], [Cas97].

4.4 Transport service production

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 successfully approached with both OR and with constraints, 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.

Similar problems occur in planning of vehicle and staff resources for public transport systems such as subways, trams and busses. 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.

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-ordinate 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 sub-contractors 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.

In the EU-project CHIC-II [CHI98] studies have been performed concerning the problem of allocating 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, given a set of roughly scheduled tasks. Different alternative approaches have been tried and compared within the project.

5 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 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 the coordination problem to us. The cooperation with the development teams for Mozart-Oz team and for SICStus/Quintus Prolog 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.

[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.

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

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

[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.

[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.

[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.

[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.

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

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

[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.

[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.

[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

[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*

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

[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.

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

[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.

[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.

[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.

[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.

[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.

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

[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.)

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

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

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

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

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

[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.