Joint NCC & IIS Bull., Comp. Science, 16 (2001), 57–65

Solving problems of resource constraint satisfaction with Time-EX

J. D. Hoffman, D. Inishev

In the present paper, we examine some issues related to the development of Time-EX (an intelligent scheduling and project management system) as an application of the method of subdefinite models (SD-models). Extension of SD-models with dynamic elements is briefly discussed, along with their implementation via the tools for knowledge representation and processing developed in the Russian Research Institute for Artificial Intelligence (RRIAI). An outline of a partial solution to the problem using the current version of Time-EX is presented.

Introduction

The connection between the method of subdefinite models [1–3] and constraint programming (CP) has been comprehensively analyzed and described in [8, 9, 12] by researchers of RRIAI.

Among the CP methods, the "tolerance propagation method" proposed by E. Hyvonen, [16], is the closest to subdefinite calculations. All the CP methods, however, have common limitations that are partially overcome by SD-models [6].

Let us examine the traditional problem of Constraint Satisfaction (CS). In the most general form, the problem is stated as follows.

Given a number of constraints $R_i(x_1, x_2, \ldots, x_n)$, $i = 1, k$ defined over the variables $x_1, x_2, \ldots, x_n$ with domains $X_1, X_2, \ldots, X_n$, we need to find the sets of values $\langle a_1, a_2, \ldots, a_n \rangle$ $(a_i \in X_i)$ that satisfy all of the constraints simultaneously.

In a similar manner, we can state the same problem in terms of SD-models, which introduce a new basic notion of "subdefinite" values. For every variable, its "subdefinite" value is an estimate for its true value based on the information available at the moment. This value is intermediate between fully known, or precise value and fully unknown, or indefinite one; it may be refined as new information becomes available.

The apparatus of SD-mathematics is described in many publications (mostly in [1–3], [8, 9, 12] and some others). We are interested in a specific application: project scheduling and planning. This problem has been studied in RRIAI in the framework of the Time-EX project. The results have been published in [4–7] and some other papers.

Subdefinite scheduling and dynamic SD-models

In Time-EX, the scheduling problem is represented as a subdefinite extension of the network time model. This model is perfect for description of a static scheduling problem.

A subdefinite schedule is described by the model $M = (X, R)$, where X is a set of t-intervals representing the execution time for the tasks to be scheduled, and R is a set of relations (constraints) linking the t-intervals of individual tasks. All of the relations (constraints) in the implemented version of the system are conjunctive (like the "succession" dependence).

In the subdefinite extension of the time model, all objects are replaced by the corresponding subdefinite objects: a subdefinite t-point and distance are represented by an interval $[a, b]$, where $a \leq b$, while a subdefinite t-interval is represented by $X = \{x, y, d\}$, where $d = y - x + 1, x \leq y$. The variables x and y are start and finish of a subdefinite t-interval and d is its duration. The variables x and y are subdefinite t-points and d is a subdefinite distance. The relations over subdefinite t-intervals are subdefinite extensions of respective relations of the t-model.

In this case the subdefinite approach is very effective. Firstly, the embedded engine implementing subdefinite calculations immediately checks the existence of a solution. Secondly, if the solution exists, it allows one to eliminate at once the contradiction between the minimal length of the schedule and availability of reserves without introducing additional parameters. If the system is inconsistent, the Time-EX technology allows one to modify the model so that it will have a solution.

The problem becomes significantly more complicated in the case of resource planning, compared to pure scheduling. In this case it is not enough to declare all the variables and constraints in advance, because once the resources have been assigned to project tasks, it is necessary to modify the model and introduce new constraints. Thus, we have the problem of introducing dynamic elements into the model.

No general solution to the problem of dynamic SD-models has been proposed so far ([12–14]). Only some specialized approaches exist that produce partial solutions to specific problems. Some of them will be described below.

We say that an SD-model is dynamic if it is possible to modify the constraint network during computation [12]. The simplest modification is addition of a new constraint. This may be needed in the following cases:

- if the solution is too indefinite and cannot satisfy the user;
- when we want to satisfy a disjunction of constraints;
- when we want to satisfy a conditional constraint;

- when we are modeling processes that change in time.

The principle of processing in the first and second cases is practically the same. Each SD-variable can be represented in the form of a disjunction. The addition of new constraints is regarded as a new state of the model in which the constraint satisfaction process is started. If the result is consistent, then it is saved; if other solutions are not required, then computation stops. If other solutions are required, then we backtrack to the original model, introduce new constraints, and repeat the CS cycle. If the result is unsatisfactory or contradictory, we return to the preceding state, enter new constraints and perform another computation cycle. Thus, the constraint satisfaction process for changing the constraint network is reduced to implementation of backtracking.

Other approaches to the development of dynamic computational models exist, for example, structural constraint satisfaction, constraint logic programming, etc.[1] The use of object-oriented approach and knowledge processing techniques is very promising (in particular, the research performed in RRIAI – NeMo, Semp-TAO, and TAO projects).

The model of knowledge representation in Semp-TAO [14, 15] combines the main tools of knowledge processing: frames, semantic networks, production rules, as well as SD-models and constraint programming techniques based on the object-oriented approach. The Semp-TAO technology is based on the notion of a local computational model, that is, a model with static definition of relations and monotonic refinement of values during recalculation. These models are linked by means of semantic networks and production rules defined over the networks. The networks consist of objects that may be arbitrary entities of the subject domain. A semantic network may contain compound objects, or frames, linked with binary relations. The use of objects allows one to structure the model more clearly and to represent it in the form of a functional network that links the slots of various objects and ensures recalculation and modification of the values of related slots of all objects in the semantic network.

Objects with identical properties are combined into a class. Properties of the class define the behavior of the object identified by a set of constraints. Classes may inherit properties of other classes; multiple inheritance is possible. An important feature of this technology is that the slots of objects may have subdefinite values that can be refined by constraints defined on the slots. The set of constraints of all objects and relations represented in a semantic network constitutes a global functional network, activated for every modification of the values of the objects' slots.

[1] We can note, for example, the work in which a semi-dynamic resource-centric scheduling model using slots is represented [11].

Thus, a computational model supports recalculation and the refinement of the values of slots in objects linked by relations. In such a model, static local computational models are integrated in a single global model. This global model is structured and has explicitly defined local submodels, and also can change its structure dynamically.

So, we can draw a conclusion that the technological environment for knowledge representation and processing ensures implementation of dynamic computational models, and the use of the object-oriented approach makes them flexible, powerful and simple.

Time-EX and resource planning problems

In the previous version of Time-EX, a model of schedules with conjunctive constraints [4–7] has been implemented.

It is known that in real problems of project management the available resources are often limited.

If a plan is interpreted as a vector $x = \{x_1, \ldots, x_n\} \in R^n$, then the admissible domain may be represented as the intersection of two domains:

$R^n \subset M_1$, a domain of resource possibilities of the project;

M_2, a domain of restrictions on the parameters of the project plan.

A plan is admissible only if it belongs to the intersection of the sets M_1 and M_2, i.e. $x \in M_1 \cap M_2$.

Most real projects deal with resource constraints, and so the problem of schedule refinement under resource restriction is crucial for project management. One important element of resource planning is *resource leveling*. When the time intervals of tasks competing for a non-shared resource intersect, the resource overload may hamper execution of the project (and may lead to the threat of frustration of the project). The introduction of additional resource, in order to increase overall productivity of the plan execution, is not always possible (resource availability restrictions) or desirable (cost restrictions). Thus, the only solution may be resource leveling, i.e. the automatic replacement of original relations between competing tasks by the "nonconcurrency" relation.

$Nonconcurrency(t_1, t_2) \Leftrightarrow Succession(t_1, t_2) \vee Succession(t_2, t_1)$,
where t_1, t_2 are time intervals.

But this relation is a disjunctive one, i.e. a disjunction of constraints. In such cases, as indicated above, the constraint satisfaction process leads to the necessity of introducing dynamic elements in the SD-model.

Some special approaches are applicable that allow one to solve the given problem partially. We shall consider one of them.

Resource leveling can be reduced to replacement of the original relations

between tasks that compete for the resources by the succession relation with lag $iR \geq 0$. Here, the order of succession will be defined by a priority assigned by the user.

Thus, the process of resource leveling may be carried out as follows:

1. Finding the interval of resource overload and the tasks that use the overloaded resource.
2. Introduction of the succession relation $F < S$ with lag $iR \geq 0$ instead of the initial relations in the model. Even if the tasks are not related, the succession relations are introduced.
3. After modification, the model is recalculated (refined), which can make the system inconsistent. In this case one more schedule optimization cycle, using a well-known technique, is necessary.

We should note here the following difficulties that can arise as a result of resource leveling.

If the resource leveling is carried out consecutively, then every leveling and reallocation step for one of the resources can lead to conflicts on the other resource, and so on. Thus, for sequential leveling the solution is reduced to exhaustive search, and for large project dimension it becomes practically unsolvable.

On the other hand, if we simultaneously introduce complete information about overloads during the resource assignment, then we can face a problem of working with a very large model. In this case we have to swap in the elements of the constraint set consecutively, which can also lead to exhaustive search.

In any case, the next step of schedule refinement (model calculation under new constraints) may show that the system has become inconsistent and requires the duration of the whole project to be increased. This situation is clear; it is taken into account in most professional scheduling systems.

Thus, the approach here described implements the method of processing the disjunctive and conditional constraints by means of backtracking as described above.

The approach that implements state-of-the-art methods of knowledge representation and processing is more promising. It can be illustrated using the elements of the technological environment Semp-TAO designed in RRIAI.

In Semp-TAO, the model of a schedule is represented by object classes, such as task, milestone, resource, etc. Each object is characterized by the values of its attributes (slots). The values of the slots can be of various data types; in our case such values can be task durations, intervals of t-points, etc. Objects may have dependences (or constraints) defined on the values

of the object slots. These dependences allow automatic refinement of the model's parameters when necessary.

The dependences between the tasks are defined in the form of *binary relations*. Once we have a set of such relations, we can manipulate the parameters of local models without changing its structure. The relations may contain constraints as well, and this allows us to refine the object parameters. As an example, we can describe the succession relation as follows:

relation *AFTER(task1, task2 : TASK)*
constraints
T1 : task1.start¿=task2.finish;
end;
relation *AFTER(task1, task2 : TASK)*
constraints
T1 : task1.start¿=task2.finish;
end;

Other object classes of the scheduling problem may be described in the knowledge representation and processing language (KRPL) as follows:

class *TASK*
Name : string;
Start : integer (0..200);
Finish : integer (0..200);
Duration : integer (0..200);
constraints
T1: duration = finish – start + 1;
end;
class *RESOURCE*
Name : string;
Quantity : integer (0..500);
end;
class *RES_TASK*
Taskname: string;
Resname : string;
Quantity : integer (0..500);
end;
relation *INCLUDE(task1, task2 : TASK)*
constraints
T1 : task1.start¡=task2.start;
T2 : task1.finish¿=task2.finish;
end;

The calculation process consists of two phases:

- model composition,
- calculation, schedule refinement and optimization.

In the first phase the objects of tasks, milestones, and resources are created. Next, the relations that define the project hierarchy (breakdown) and the sequence of task execution are defined. Once the tasks are linked by the relations, their parameters are recalculated automatically. As a result, we have the functional network that represents the global model of the project plan.

To compose the plan, we activate the production rules that change the structure of the preliminary network. The network obtained in the first step is an intermediate version of the plan if it is not inconsistent. It is saved as a source version. In the process of application of the rules, the parameters may become inconsistent, which means that optimization is impossible without changing the network parameters. In this case the system will return to the source version.

An example of a production rule may be the rule that finds a resource overload interval and the tasks using the overloaded resource. The rule is applied by means of the pattern search. The pattern of the rule will be intersection of the time intervals of the tasks that use the same resource. The result of the rule execution will be resource leveling by adding the succession relation between these tasks, for example, in the order of decreasing priority.

In the scheduling and resource-planning problem, it is natural to interpret the functional network as a structural PERT diagram of a project that reflected the project's organization structure. The local model connected to objects can describe not only single tasks but also the whole stages with an arbitrary internal structure.

Here two inverse processes are possible: decomposition of a global model (disassembly) and integration of local models (assembly).

The possibility of structured planning makes resource planning more meaningful due to the following assumptions:

- the resources are not arbitrarily assigned, but according to a certain organization structure of the project; for example, a separate team may perform each subproject, and the equipment needed for this is assigned in advance as unshared resource;
- resource leveling is implemented according to the plan structure, as in the stage of composing a plan, i.e., in each subproject, leveling is implemented independently, and in the integrated plan the load leveling is implemented only for those resources that are assigned to

task groups, for example, resources of type "management" assigned to the whole subproject.

Thus, application of the knowledge representation and processing methods allows quick implementation of intelligent planning systems with a possibility of automatic dynamic resource leveling, introduction of automatic network optimization and working with several project configurations differing both in the structure and in the number of tasks and relations.

Conclusion

We present here some problems that are now under investigation within Time-EX, an environment project of intelligent planning. One more step has been done towards an intelligent project management system that will satisfy all standard management requirements and provide knowledge representation and processing tools for this subject domain.

In the future, the Time-EX technology will be developed in the following directions:

- Increase in inference efficiency and extension of the engine to process complex dependences between time parameters and resources, in particular, dynamic and disjunctive ones. We intend to use new technological environments designed in RRIAI.
- Integration with other intelligent systems (for example, subdefinite financial planning) based on knowledge representation and processing tools and methods.

References

[1] Narin'yani A. S., Sub-definite Set — a Formal Model of Uncompletely Specified Aggregate, Proc. of the Intern. Symposium on Fuzzy Set and Possibility Theory", Acapulco, Mexico, 1980

[2] Narin'yani A. S., Sub-definiteness and Basic Means of knowledge representation, In "Computers and Artificial Intelligence", vol.2, N5, Bratislava, 1983

[3] Nariny'ani A. S., Subdefiniteness in knowledge representation and processing System, "Tehnicheskaya Kibernetika", N 5, 1986 (in Russian)

[4] Narin'yani A. S., Borde S. B., Ivanov D. A., Sub-Definite Mathematics and Novel Scheduling Technology Programs, In: "Artificial Intelligence in Engineering", v.11, N1, February, 1996

[5] Nariny'any A. S., Sedreeva G. O., Sedreev S. V., Frolov S. A., New Generation of the scheduling technology, in "Representation and processing of partially defined knowledge", Russian Research Institute for Artificial Intelligent, Moscow-Novosibirsk, 1996 (in Russian)

[6] Narin'yani A. S., Hoffman J. D., Inishev D., Banasyukevich D., Time-EX as an Application of Constraint Programming Based on Subdefinite Models, Proceedings of the 2000 ERCIM / Compulog Net, Workshop on Constraints, Padova, Italy, June 19-21, 2000

[7] Narin'yani A. S., Hoffman J. D., Inishev D., Banasyukevich D., Intelligence technology of subdefinite scheduling and project management, Proc. II-nd Conference Complex Systems: Control and Modeling Problems (CSCMP-2000), Samara, 2000 (in Russian)

[8] Semenov A. L., Shvetsov I. E., Telerman V. V., Constraint Programming Based on Subdefinite Models and its Applications, ILPS'95 Post-conference Workshop on Interval Constraints, USA, Portland, Oregon, 1995

[9] Ushakov D. M., Telerman V. V., Subdefinite Models as a Variety of Constraint Programming, Proc. of the Int. Conf.: Tools of Artificial Intelligence (ICTAI'96), Toulouse, 1996

[10] Borde S. B., Ponkin S. A., Salychev M. V., Subdefiniteness and calendar scheduling, In Proceedings of the East–West Conference on Artificial Intelligence EWAIC-93, Moscow, pp. 315–318, 1993

[11] Roman Bartak, A Slot Representation of the Resource-Centric Models for Scheduling Problems, Proceedings of the 2000 ERCIM / Compulog Net, Workshop on Constraints, Padova, Italy, June 19-21, 2000

[12] Telerman V., Ushakov D., Data Types in Subdefinite Models. In: Jacques Calmet and others (eds.), Art. Intell. and Symbolic Mathematical Computation, Lecture Notes in Computer Science; Vol. 1138, Springer, (1996), pp. 305–319.

[13] Vazhev I. V., Semenov A. L., Interval Constraints Propagation with dynamic change of data types, in "Representation and processing of partially defined knowledge", Russian Research Institute for Artificial Intelligent, Moscow–Novosibirsk, 1996 (in Russian)

[14] Zagorulko Yu. A., Popov I. G., Object-Oriented Language for Knowledge Representation Using Dynamic Set of Constraints, Knowledge-Based Software Engineering, P.Navrat, H.Ueno (eds), (Proc. 3rd Joint Conf., Smolenice, Slovakia), Amsterdam: IOSPress, 1998, P.124–131.

[15] Zagorulko Yu. A., Popov I. G., A Software Environment based on an Integrated Knowledge Representation Model, Perspectives of System Informatics (Proc. of Andrei Ershov Second International Conference PSI'96), Novosibirsk, June 25-28, 1996, P.300-304.

[16] Hyvonen E. Constraint reasoning based on Interval arithmetic: the tolerance propagation approach, Artificial Intelligence, v.58, p.71–112, 1992.