Search Strategies for Scheduling Problems with Optional Activities

Roman Barták

Charles University, Faculty of Mathematics and Physics
Malostranské nám. 2/25, 118 00 Praha 1, Czech Republic
roman.bartak@mff.cuni.cz

Abstract

Scheduling problems typically deal with sequencing and resource allocation while the problem of activity selection is assumed to be resolved before scheduling. To increase flexibility of the scheduling systems in complex environments, it is possible to include decisions about other alternatives, for example about alternative process routes, in the scheduling problem. This could be realized by assuming optional activities in the problem. The paper studies search strategies for solving scheduling problems with optional activities – the search procedure assumes both sequencing decisions and selection among optional activities.

Introduction

Traditional scheduling problems deal with allocating known activities to scarce resources, such as machines, over time. The set of activities to be scheduled is known in advance. Optional activities modeling alternative process routes were introduced to scheduling by Beck and Fox (1999). The idea is that each activity is annotated by a Boolean validity variable which indicates presence of the activity in the schedule. These validity variables can participate in explicit logical constraints describing possible relationships between existences of activities as for example described in ILOG Manufacturing Scheduling Library (Nuijten et al. 2003). In (Barták and Čepek 2007) we proposed implicit logical relations between the optional activities targeted to modeling alternative processes. Now, the scheduling problem with optional activities consists of selection of a subset of activities and allocating them to time in such a way that the logical, temporal, and resource constraints are satisfied. Though there exist techniques to include optional activities in resource constraints (Vilím et al. 2005) we are not aware about search strategies that involve decisions about optional activities.

In this paper we propose a branching method for scheduling problems with optional activities. This method integrates sequencing decisions as well as decisions about alternative processes. We believe that this method will give better-quality results in comparison with the traditional approach where both decisions are resolved separately.

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Traditional Branching Strategies

The traditional scheduling problem "reduces" to deciding the start time of each activity. Basically, there are two search strategies: deciding the start times directly (SST – Set a Start Time) or precedence constraint posting (PCP) followed by the assignment of start time variables.

SST strategy is a strong commitment strategy – already allocated activity cannot be shifted later in time to satisfy some constraint, it is necessary to backtrack to the decision point and to select the appropriate time there. This seems too rigid for many scheduling problems.

PCP strategy decides first the order of all activities in each resource and in the second stage, allocates the activities to time. Frequently, the second stage – time allocation – can be realized by backtrack-free search if the polynomial constraint propagation guarantees global consistency of temporal constraints (which is, for example, the case of simple temporal networks). In the first stage, we can take a pair of not-yet ordered activities A and B and the search procedure branches over the options whether A is before B or not. This branching scheme requires $O(n^2)$ choices to be resolved during search because each pair among the n activities should be ordered. In (Baptiste et al., 1995) a different branching scheme for activity ordering is studied. Rather than deciding about the order of two not-yet ordered activities, we can decide about the first activity in the resource – either some activity is allocated first in the resource or not. This branching strategy uses $O(n)$ choice points during search.

Branching with Optional Activities

When optional activities are assumed, the sequencing decisions must be accompanied by the selection of activities. This could be done separately which corresponds to the traditional approach "plan first, schedule next" that is currently used to solve such problems. Because scheduling decisions strongly influence selection of activities and vice versa (for example a fully occupied resource may force alternative routing such as outsourcing for another process), we propose an integrated branching strategy for deciding the position of activity in the set of activities. We try to allocate the selected activity in the first position in the resource and if this is not possible we try the second position etc. until its position is found.

Let us first introduce some notions. We call the activity *invalid* if its validity variable is set to false (invalid activities are not present in the solution). Similarly, we call activity *valid* if its validity variable is set to true (valid activities are present in the solution). *Non-invalid* activities are activities that are not invalid – they are either valid or their validity status is not yet decided (they may become either valid or invalid). We say that the activity is *allocated* if its position in the resource is known. By knowing the position of the activity we mean that all valid activities allocated to the resource before the given activity are known. Our branching scheme is based on selecting a non-invalid activity whose position is not known and finding the position of the activity. Basically, we are allocating the activities from left to right (from past to future). Among all non-invalid, not yet allocated activities, we select the activity with the earliest possible start time, ties broken by preferring the activity with the smallest duration and the smallest latest completion time. Let us call this activity A and let XA be the set of all other non-invalid, not yet allocated activities that share the same unary resource with A. There are two levels of branching for activity A. First, we branch between making the activity A valid and making the activity A invalid (if the activity is already valid the second branch is ignored). The second branching level is only for valid activity A and it is the allocation of the activity, that is, finding the position of the activity. For each activity B in XA that is still non-invalid (it may happen that some activities in XA become invalid due to propagation of previous decisions), we first try to allocate A before B, that is we introduce a precedence constraint between A and B. The alternative choice is making the activity B valid and allocating it before A (by posting a precedence constraint). This way we find the position of A in the resource because we know all valid activates that appear before A (all other activities are either invalid or positioned after A). Figure 1 gives an example of the search tree for XA = {B, C, D}.

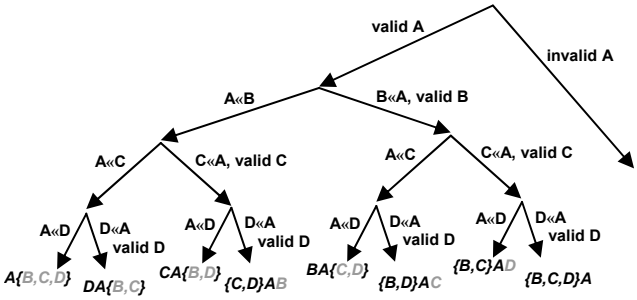

Figure 1. Allocation process for activity A (possible sequences are in leafs, activities in parentheses are not-yet ordered, grey activities are not-yet decided to be valid).

If activity A is allocated or decided to be invalid the above process is repeated for the remaining non-invalid, not yet allocated activities until all activities are either invalid or valid and allocated. Finally, we assign the start times by using the earliest possible start time for each valid activity.

Conclusions

The paper describes a first step towards scheduling strategies that integrate decisions about activity sequencing with selection of activities to be scheduled. Such integration produces better-quality plans than plans obtained by solving both problems separately (because of higher flexibility in the decisions) and we also believe that it is computationally viable.

There are other possible branching strategies for the sequencing of optional activities. The important issue is to propose a strategy that explores all alternatives and does not explore any alternative twice. This may seem easy, but one must realize that if the activity is invalid then its position in the resource is irrelevant. Hence if we try several positions of the invalid activity in the schedule (this may happen, if the position of the activity is decided and then the activity becomes invalid) then we are wasting time because they are all equivalent regarding the final schedule. The proposed strategy guarantees that it explores all possible sequences of valid activities (all possible schedules of the resource) and no sequence is explored twice. We are working on a formal proof of this claim that is based on the following idea. For each valid activity, we explore all its positions in the resource, that is, all possible valid predecessors are tried. For any pair of activities A and B such that A is being allocated first in this pair we explore the following possible options: valid A is before valid B, valid A is before invalid B, valid B is before valid A, A is invalid and each of these options is tried once. Later if B is being allocated then its position to A is already determined or A is invalid so no ordering constraints between A and B are explored.

Acknowledgments. The research is supported by the Czech Science Foundation under the contract no. 201/07/0205.

References

Baptiste, P.; Le Pape, C.; Nuijten, W. 1995. Constraint-Based Optimization and Approximation for Job-Shop Scheduling. *Proceedings of the AAAI-SIGMAN Workshop on Intelligent Manufacturing Systems*, IJCAI-95.

Barták R., Čepek O. 2007. Nested Temporal Networks with Alternatives. In *Papers from the 2007 AAAI Workshop on Spatial and Temporal Reasoning* (TR WS-07-12), pp. 1-8, AAAI Press.

Beck, J.Ch. and Fox, M.S. 1999. Scheduling Alternative Activities. *Proceedings of the National Conference on Artificial Intelligence* (AAAI), pp. 680-687. AAAI Press.

Nuijten, W.; Bousonville, T.; Focacci, F.; Godard, D.; Le Pape, C. 2003. MaScLib: Problem description and test bed design, http://www2.ilog.com/masclib

Vilím P., Barták R., Čepek, O. 2005. Extension of O(n log n) filtering algorithms for the unary resource constraint to optional activities. *Constraints*, Volume 10, Number 4, pp. 403-425. Springer Verlag.