Towards an Effective Framework Combining Planning and Scheduling [Extended Abstract]*

Andrii Nyporko[1,2], **Lukáš Chrpa**[2]

[1] Faculty of Electrical Engineering, Czech Technical University in Prague
[2] Institute of Informatics, Robotics and Cybernetics, Czech Technical University in Prague
nyporand@cvut.cz, chrpaluk@cvut.cz

Introduction

Automated Planning and Scheduling are complementary fields of study that are usually pursued separately. Whereas planning deals with finding sequences of actions achieving a specified goal, scheduling deals with the problem of allocating activities (or jobs) to limited resources. Looking from the scheduling side, activities (or jobs) as well as their (partial) ordering have to be (usually) explicitly specified up front. Scheduling with alternatives that allows choosing which activities to schedule has been studied in recent past (Čapek, Šůcha, and Hanzálek 2012; Hanzálek, Čapek, and Šůcha 2017). However, it still requires an explicit specification of alternatives and ordering of activities that, on the other hand, is inherent to planning. Temporal planning (Fox and Long 2003) offers a machinery for incorporating scheduling into planning but this has to be done explicitly. One of the frameworks that effectively combine planning and scheduling is NASA's EUROPA that is used in systems for space and planetary exploration (see e.g. (Muscettola et al. 1998)). Ruml, Do, and Fromherz (2005) aim at on-line planning and scheduling in manufacturing and represent activities like "planning" actions, which is, perhaps, the closest work to our proposal.

This extended abstract formalises the concept of "combined" planning and scheduling tasks, where we introduce two types of activities – production and maintenance – represented similarly to actions in planning that are then scheduled on (limited) resources. We propose a high-level idea how these tasks can be compiled to classical planning tasks. Note that classical planning domains from the International Planning Competition – Schedule or Woodworking – are examples of such "combined" tasks. Our idea is evaluated on tasks concerning scheduling activities on reconfigurable machines (Borgo et al. 2016; Vahedi-Nouri et al. 2022).

Problem Specification

We can describe the environment by a set of *state variables* V, where each variable $v \in V$ has its own domain $D(v)$. An *assignment* of a variable $v \in V$ is a pair (v, val), where $val \in D(v)$. A (partial) *variable assignment* p over V is a set of assignments of individual variables from V and $p[v]$ represents a value of v in p (or $p[v]$ is undefined if v in not part of p). Let $vars(p)$ denote the set of variables that p considers. A *state* is a complete variable assignment over V. We say that a (partial) variable assignment q *holds* in a (partial) variable assignment p, denoted as $p \models q$, if and only if for each v being a part of q it is the case that $q[v] = p[v]$.

Then, we define a set of attributes L that determine configuration of *resources* R (e.g. machines). A *resource* $r \in R$ has its *attributes*, denoted as $attr(r) \subseteq L$.

We consider two types of activities, namely *production activities* and *maintenance activities*, where the former represent "jobs" that have to be run on resources while the latter change attributes of the resources. A *production activity* is a quadruple $a = (dur(a), attr(a), pre(a), \mathit{eff}(a))$, where $dur(a)$ represents the duration of a, $attr(a) \subseteq L$ represents a set of attributes that a requires, and $pre(a)$ and $\mathit{eff}(a)$ are partial variable assignments over V representing a's precondition and effects, respectively. Production activities produce specified "products" that might be required by other (production) activities or might be goals.

A *maintenance activity* is a quadruple $m = (dur(m), res(m), rem(m), add(m))$, where $dur(a)$ represents the duration of m, $res(m) \subseteq R$ represents a set of resources on which m can be performed, $rem(m) \subseteq L$ and $add(m) \subseteq L$ are sets of attributes that are removed or added, respectively.

We define a *planning-scheduling* task (or PS task, for short) as a tuple $P = (V, L, A, R, M, I, G)$, where V is a set of state variables, L is a set of attributes, A is a set of production activities, R is a set of resources (each with initial attributes), M is a set of maintenance activities, I is an initial state (over V) and G is a partial assignment over V representing the goal.

Let $\pi = \{(t_1, x_1, r_1), \ldots, (t_n, x_n, r_n)\}$ be a set of triples (timestamp,activity,resource), where $x_i \in A \cup M$ and $r_i \in R$ $(1 \leq i \leq n)$. We say that π is *consistent* if for each x_i, x_j $(i \neq j)$ it is the case that if $[t_i, t_i + dur(x_i)]$ and $[t_j, t_j + dur(x_j)]$ are not disjoint, then $r_i \neq r_j$ and if $x_i, x_j \in A$, then $vars(x_i) \cap vars(x_j) = \emptyset$. The *state trajectory* respecting I and π is a sequence of states defined recursively such that $s(0) = I$ and for each $v \in V$,

*This research is supported by Czech Science Foundation (project no. 23-05575S) and by Grant Agency of the Czech Technical University (project no. SGS23/091/OHK3/1T/37)

$s(t+1)[v] = \mathit{eff}(x_i)[v]$ if there exists $x_i \in A$ such that $t+1 = t_i + dur(x_i)$, or $s(t+1)[v] = s(t)[v]$ otherwise. Analogously, for each resource $r \in R$ we define a *configuration trajectory* as a sequence of attributes starting with the initial attributes $attr(r)(0)$ and where $attr(r)(t+1) = (attr(r)(t) \setminus rem(x_i)) \cup add(x_i)$ if there exists $x_i \in M$ such that $r_i = r$ and $t_i + dur(x_i) = t+1$, or $attr(r)(t+1) = attr(r)(t)$ otherwise. We say that a maintenance activity $m \in M$ is *applicable* on a resource r in time t iff $rem(m) \subseteq attr(r)(t)$. We also say that a production activity $a \in A$ is *applicable* on a resource r in time t iff $s(t) \models pre(a)$ and $attr(a) \subseteq attr(r)(t)$. We say that π is a *solution* of a PS-task P iff π is consistent, for each (t_i, x_i, r_i) it is the case that x_i is applicable on r_i in t_i, and $s(\max_{i=1}^{n}(t_i + dur(x_i))) \models G$.

Solutions of PS-tasks can be optimised for a given objective function that might be, for example, makespan, or total cost of activities (might depend on resources and starting/finishing time).

Towards Compiling PS Tasks to Classical Planning

The high-level idea how to represent a PS task as a planning task is to merge the "planning" environment with the resource-specific attributes. Both types of activities can be represented as actions. Temporal aspects of the problem can be represented by "timestamp" objects alongside with predicates responsible for arithmetic and logical operations. For each resource, we keep track of the timestamp in which the last scheduled activity (so far) finished. On top of that, we have to assure that (implicit) ordering of (production) activities, given by the fact that one activity achieves a precondition for another activity, respect time constraints by keeping track in which timestamp a certain fact has been achieved. Then, we specify "synchronising" actions that synchronises timestamp for an activity as maximum of timestamps of its precondition facts and the current timestamp of the machine on which the activity is planned to be scheduled.

We would like to note that in contrast to temporal planning (PDDL-based), such an approach allows for straightforward incorporation of time-based constraints (e.g. deadlines). On top of that, classical planning is often leveraged to solve temporal planning tasks (in PDDL 2.1 semantics) (Celorrio, Jonsson, and Palacios 2015).

Reconfigurable Machines: A Case Study

For our preliminary evaluation of our idea we considered production planning of *Reconfigurable Machines* that are becoming more widespread (Borgo et al. 2016). Production activities require a specific (machine) configuration that can be changed by maintenance activities. Some production activities require an input from another production activity. As planners we considered FastDownward (Helmert 2006) with FF and hcea heuristics, k-BFWS (Lipovetzky and Geffner 2017) and Powerlifted (B. Corrêa and Seipp 2022) that reasons over lifted representations. Timeout is 3600 seconds.

The results in Table 1 show that we can solve tasks with 140 activities in several seconds by the lifted planner (as grounding seems to struggle with timestamp objects). Specialized approaches are capable of solving more complex problems (Vahedi-Nouri et al. 2022), e.g. 110x16x7 in 892 seconds. In spite of encouraging results achieved by the Powerlifted planner, for more complex cases we might still need more effective means to reason with timestamp objects, which we plan for future work.

AxMxC	*FastDownward*	*BFWS*	*Powerlifted*
40x2x4	44.2(3.7)	25.8(3.05)	0.08
80x2x4	305(108)	576(23.6)	0.5
100x2x4	434.85(126)	2551.2(70.09)	0.8
120x2x4	1027(446)	TIMEOUT	1.5
140x2x4	TIMEOUT	TIMEOUT	4.9

Table 1: Planning time (in seconds), where search time is shown in brackets. AxMxC stands for the number of production activities, machines and configurations, respectively.

References

B. Corrêa, A.; and Seipp, J. 2022. Best-First Width Search for Lifted Classical Planning. In *the International Conference on Automated Planning and Scheduling*, 11–15.

Borgo, S.; Cesta, A.; Orlandini, A.; and Umbrico, A. 2016. A Planning-Based Architecture for a Reconfigurable Manufacturing System. In *the International Conference on Automated Planning and Scheduling*, 358–366.

Celorrio, S. J.; Jonsson, A.; and Palacios, H. 2015. Temporal Planning With Required Concurrency Using Classical Planning. In *the Twenty-Fifth International Conference on Automated Planning and Scheduling, ICAPS 2015*, 129–137.

Fox, M.; and Long, D. 2003. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. *J. Artif. Intell. Res.*, 20: 61–124.

Hanzálek, Z.; Čapek, R.; and Šůcha, P. 2017. Total Setup Time Minimisation in Production Scheduling with Alternatives. In *Industrial Applications of Holonic and Multi-Agent Systems*, 11–23. Cham: Springer International Publishing.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26: 191–246.

Lipovetzky, N.; and Geffner, H. 2017. A Polynomial Planning Algorithm That Beats LAMA and FF. In *the International Conference on Automated Planning and Scheduling, ICAPS 2017*, volume 27, 195–199.

Muscettola, N.; Nayak, P. P.; Pell, B.; and Williams, B. C. 1998. Remote Agent: To Boldly Go Where No AI System Has Gone Before. *Artif. Intell.*, 103(1-2): 5–47.

Ruml, W.; Do, M.; and Fromherz, M. 2005. On-line Planning and Scheduling for High-speed Manufacturing. In *the 15th International Conference on Automated Planning and Scheduling, ICAPS 2005*, 30–39.

Vahedi-Nouri, B.; Tavakkoli-Moghaddam, R.; Hanzálek, Z.; and Dolgui, A. 2022. Workforce planning and production scheduling in a reconfigurable manufacturing system facing the COVID-19 pandemic. *Journal of Manufacturing Systems*, 63: 563–574.

Čapek, R.; Šůcha, P.; and Hanzálek, Z. 2012. Production scheduling with alternative process plans. *European Journal of Operational Research*, 217(2): 300–311.