Artificial Intelligence 246 (2017) 1–21

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Model-lite planning: Case-based vs. model-based approaches

Hankz Hankui Zhuo [a,*], Subbarao Kambhampati [b]

[a] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
[b] Department of Computer Science and Engineering, Arizona State University, United States

ARTICLE INFO

Article history:
Received 2 July 2015
Received in revised form 17 January 2017
Accepted 19 January 2017
Available online 25 January 2017

Keywords:
AI planning
Case-based planning
Action model learning

ABSTRACT

There is increasing awareness in the planning community that depending on complete models impedes the applicability of planning technology in many real world domains where the burden of specifying complete domain models is too high. In this paper, we consider the problem of generating robust and accurate plans, when the agent only has access to incomplete domain models, supplanted by a set of successful plan cases. We will develop two classes of approaches – one case-based and the other model-based. ML−CBP is a case-based approach that leverages the incomplete model and the plan cases to solve a new problem directly by affecting case-level transfer. RIM is a model-based approach that uses the incomplete model and the plan cases to first learn a more complete model. This model contains both primitive actions as well as macro-operators that are derived from the plan cases. The learned model is then used in conjunction with an off-the-shelf planner to solve new problems. We present a comprehensive evaluation of the two approaches, both to characterize their relative tradeoffs, and to quantify their advances over existing approaches.

1. Introduction

Most work in planning assumes that complete domain models are given as input in order to synthesize plans. However, there is increasing awareness that building domain models at any level of completeness presents steep challenges for domain creators. Indeed, recent work in web-service composition (cf. [5,23]) and work-flow management (cf. [6]) suggest that dependence on complete models can well be the real bottle-neck inhibiting applications of current planning technology.

There has thus been interest in the so-called "model-lite" planning approaches (cf. [29]) that aim to synthesize plans even in the presence of incomplete domain models. The premise here is that while complete models cannot be guaranteed, it is often possible for the domain experts to put together reasonable but incomplete models to generate plans. The challenge then is to work with these incomplete domain models, and yet produce plans that have a high chance of success with respect to the "complete" (but unknown) domain model. Producing such robust plans is, of course, only possible if the planner has access to additional sources of knowledge besides the incomplete domain model.

In this paper, we look at the case where, in addition to the incomplete model, the agent has access to plan cases that were successful in the past. It is not difficult to collect successful plan cases in real-world applications. For example, operating systems, such as Linux, can easily collect successful plan cases by recording the log information when users operating the system, as mentioned by Yang et al. [52]. As another real-world application, in intelligent factory, the control center

* Corresponding author.
E-mail addresses: zhuohank@mail.sysu.edu.cn (H.H. Zhuo), rao@asu.edu (S. Kambhampati).

http://dx.doi.org/10.1016/j.artint.2017.01.004
0004-3702/

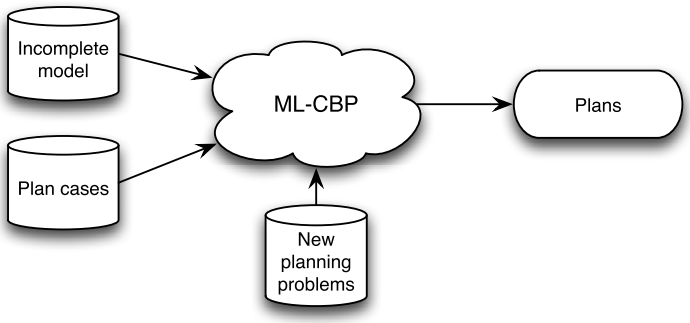

Fig. 1. The framework of ML-CBP.

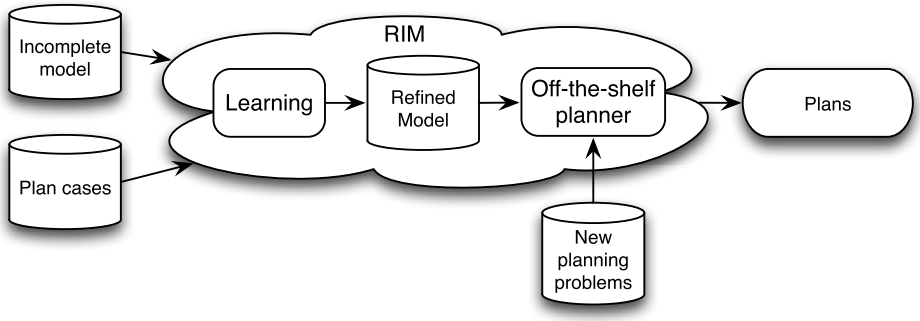

Fig. 2. The framework of RIM.

can collect plan cases by recording instructions distributed to mechanical equipments and/or workers, or using sensors to observe actions executed by mechanical equipments and/or workers. On the other hand, we consider that the preconditions and effects specified in the incomplete domain model are composed of human's common knowledge. For example, Executing an action of "deleting a file" generally requires a precondition specifying that "the file exists", resulting in "the file being deleted" (as an effect of the action of "deleting"). This common knowledge is generally correct and not difficult to collect. Here we define a plan case by a triple, i.e., an initial state s_0, a goal g and an action sequence that transits s_0 to g. These cases can be seen as providing additional knowledge of the domain over and above the incomplete domain theory. Specifically, they indicate that the true model must correspond to an extension of the incomplete model that will allow the plan cases to execute successfully. Such cases can be exploited in either case-based or model-based approaches. In the case-based approaches, the new problems are solved by directly using cases, while in the later, the cases are first compiled into a (refined) action model, which is then passed on to an off-the-shelf planner. The aim of this paper is to develop and compare both case-based and model-based approaches. We are interested, in particular, in how the number of available cases and the degree of incompleteness of the input model affect the relative performance of the two types of approaches.

At first blush, it would seem as if we can directly use off-the-shelf case-based approaches (such as OAKPlan [44]) or action-model-learning approaches (such as ARMS [60,52]). This however turns out not to be the case. As we shall elaborate below, the existing case-based approaches assume availability of the complete domain model (and focus on efficiency rather than robustness). Most of existing model-learning approaches, in contrast, assume that only plan cases are available, and thus do not exploit the availability of a partial/incomplete model. As we shall see, such a partial model can be leveraged to come up with models that include macro-actions in addition to primitive ones.

Accordingly, in this paper, we develop and present novel case-based as well as model-based approaches. For the case-based approach, we take a two stage process. First, we use the incomplete domain model to synthesize a "skeletal" plan. Next, with the skeletal plan in hand, we "mine" the case library for fragments of plans that can be spliced into the skeletal plan to increase its correctness. The plan improved this way is returned as the best-guess solution to the original problem. We call this approach ML-CBP, which stands for **M**odel-**L**ite **C**ase-**B**ased **P**lanning, a previous version of which can be found from [57]. The framework of ML-CBP is shown in Fig. 1. Plan cases are exploited to produce plan solutions for planning problems directly in ML-CBP.

For the model-based approach we develop a framework called RIM (which stands for **R**efining **I**ncomplete planning domain **M**odels through plan cases, a previous version of which can be found from [58]. The framework of RIM is shown in Fig. 2) that extends the ARMS family of methods for learning models from plan cases [60,52] so that they can benefit from both the incomplete model, and the macro-operators extracted from the plan cases. In the first phase, we mine candidate macros mined from the plan cases, and in the second phase we learn precondition/effect models both for the primitive actions and the macro actions. Finally we use the refined model to do planning (where the planner is biased