Concurrent Planning by Decomposition[1]

L. Sebastia and **E. Onaindia** and **E. Marzal** [2]

Abstract. One approach to reduce the inherent complexity of planning is problem decomposition. This paper introduces a new decomposition technique in STRIPS domains which is based on the idea of landmark. Landmarks are ordered and grouped in different sequential sets named intermediate goals (IG), being each IG a sub-goal to solve. Then, we build the initial state (IS) corresponding to each IG, such that each pair (IS, IG) is viewed as an independent problem. This way a multiprocessor system can be used to solve these problems concurrently.

1 Introduction

Planning is a difficult task. One of the techniques that can be used to reduce this difficulty is **problem decomposition**, that is, decomposing the planning problem into smaller problems, solving these problems and combining the obtained solutions. This approach brings three advantages: (1) the obtained problems are simpler than the original problem, and, consequently, are easier to solve, (2) a concurrent resolution (in a multiprocessor system) of these problems may report important time savings and (3) the same new techniques used in general planning can also be used to solve these problems.

In this paper, we present a new decomposition technique for STRIPS domains. This technique first studies the structure of the planning problem $P = \langle \mathcal{A}, \mathcal{I}, \mathcal{G} \rangle$[3] to extract a set of landmarks, that is, literals that must be true in any solution plan. These landmarks are grouped into sequential intermediate goal sets IG_i, where each IG_i defines the goal of a new subproblem. Then, we compute an intermediate initial state IS_i for each new subproblem. These problems $P_i = \langle \mathcal{A}, IS_{i-1}, IG_i \rangle$ can be solved by any planner to obtain a plan Pl_i. The final solution plan for the original problem is obtained by simply concatenating the plans Pl_i: $Pl = Pl_1 \circ Pl_2 \circ \ldots \circ Pl_{n-1} \circ Pl_n$.

2 Overview of our decomposition technique

In order to explain our decomposition technique, we introduce the example shown in Figure 1. In this problem, the goal is to transport some crates from one location to another to obtain the configuration specified in the goal.

Our decomposition approach is based on the concept of landmark and on the orders that can be established between them ([6], [5]).

Definition 1 *A* **landmark** *is defined as a literal that is true at some point in all solution plans.*

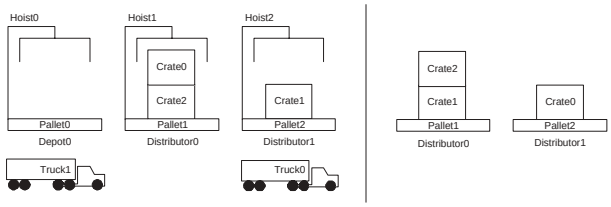

Figure 1. Example on the depots domain [2].

Definition 2 *There is a* **necessary** *order between two landmarks l and l' if l is a prerequisite for achieving l'. A* **reasonable** *order between l and l' states that it is reasonable to achieve l first, because otherwise l' would be achieved twice in the plan.*

Definition 3 *A* **LG** *is a graph (N, E) where N is the set of landmarks and E represents the necessary and reasonable orders among landmarks. We represent as $l \leq l'$ the existence of an edge between two landmarks l and l' in the LG.*

We can identify the following landmarks in our example, among others: (clear Pallet2), (lifting Hoist2 Crate0) and (on Crate0 Pallet2). We can establish the following necessary orders between these landmarks: (clear Pallet2) $<_n$ (on Crate0 Pallet2) and (lifting Hoist2 Crate0) $<_n$ (on Crate0 Pallet2). From these necessary orders we can conclude that: (clear Pallet2) $<_r$ (lifting Hoist2 Crate0).

Once the LG for a planning problem is built, our decomposition technique uses this graph to compute the intermediate goals.

Definition 4 *An* **intermediate goal (IG)** *is a set of landmarks that must all be true at the same point in a solution plan.*

Each IG must accomplish the following properties:

Property 1 *All the literals in an IG must be consistent*[4] *between each other.*

Property 2 *A literal l belongs to an IG if and only if all its predecessor nodes in the $LG(N, E)$ have been visited*[5] *before l.*

Property 3 *A visited literal l is propagated to the following IGs if l belongs to $\mathcal{G}$ or until a successor literal l' in the $LG(N, E)$ is visited.*

In our example, the first IG is: {(clear Crate2), (lifting Hoist1 Crate0), (lifting Hoist2 Crate1), (clear Pallet2)}. The next IGs are computed by

[1] This work has been partially funded by the Spanish government CICYT projects TIC2002-04146-C05-04, TIC2001-4936 and DPI2001-2094-C03-03 and by the UPV project 20020681.

[2] Technical University of Valencia. e-mail: {lstarin, onaindia, emarzal}@dsic.upv.es

[3] $\mathcal{A}$ is the set of actions, $\mathcal{I}$ is the initial state and $\mathcal{G}$ is the goal.

[4] Two literals are said to be consistent when they can simultaneously belong to the same correct planning state.

[5] A literal is said to have been visited if it has been included in a previous IG.

going through the *LG(N,E)* and grouping together those landmarks that fulfil the given properties.

Once a planning problem is decomposed into a set of n ordered IGs, our aim is to compute an intermediate state for each new subproblem so that they can be solved concurrently.

Definition 5 *An* **intermediate state (IS_i)** *for P_{i+1} is a complete world state which will contain the literals from IG_i plus some other literals necessary to complete the state: $IS_i \supseteq IG_i$.*

Consequently, we have to find the set of literals to add to IG_i such that makes IS_i be the closest consistent state to IG_i (according to IS_{i-1}). In order to do so, we use the concept of **property space**[1].

Definition 6 *A* **property** *pr_i is a predicate pr subscripted by a number between 1 and the arity of that predicate.*

Definition 7 *A* **property state** *(PS) is a conjunction of properties such that all the properties in the set must hold for an object in a particular problem state.*

Definition 8 *A* **property space** *is (in short) the set of PSs that specify all the possible states of an object.*

A PS for a crate in the depots domain is $[in_1]$, which indicates that, in a valid state, a crate can be in a truck. The complete property space for crates is: $\{[in_1], [clear_1, on_1, at_1], [on_2, on_1, at_1], [lifting_2]\}$.

In order to build IS_i, we use the previous IS (IS_{i-1}) and the previous IG (IG_i). Then, we select the most appropriate PS for each object in the domain in this particular state and we instantiate it in order to obtain the literals that will be contained in IS_i.

Assuming that IS_{i-1}=$\{$(clear Pallet1), (at Pallet1 Distributor0), (at Hoist1 Distributor0), (lifting Hoist1 Crate1)$\}$ and IG_i=$\{$(on Crate1 Pallet1)$\}$, the most appropriate PS for Crate1 is $[clear_1, on_1, at_1]$. Therefore, the literals referring to Crate1 that will be included in IS_i are $\{$(clear Crate1), (on Crate1 Pallet1), (at Crate1 Distributor0)$\}$.

3 Experiments

In this section, we show a comparison between the results obtained with and without our decomposition technique when solving problems from four domains with three state-of-the-art planners that took part in the last IPC: FF[4], LPG[6] [3] and VHPOP[7].

The problems obtained through the decomposition process can also be solved sequentially (row *TS*). In this case, it is not necessary to compute the intermediate states. *TC* has been computed as follows (T_{decomp} is the time used for decomposing a problem, $T_{problems}$ is the time used to solve all the subproblems sequentially and $T_{max\ pr}$ is the time used to solve the largest problem):

$$TC = T_{decomp} + Max\left(\frac{T_{problems}}{n}, T_{max\ pr}\right)$$

The most outstanding result is that the three planners are able to solve more problems when they are executed with our decomposition technique. Therefore, we can affirm that our technique decomposes the original problem into simpler problems. On the other hand, though solving a decomposed problem usually implies a lack of quality in the solutions (due to the problem is not considered as a whole), our results show that the order in which landmarks are included in the IGs allows to preserve the quality of the solutions and to obtain shorter plans in many cases. With respect to execution time, we can observe that our approach allows planners to solve some problems in a shorter time. As a conclusion, we can affirm that the use of our decomposition technique and the concurrent resolution of the obtained problems can report significant time savings in many cases, which are more remarkable with more time-consuming planners.

[6] Results for LPG have been obtained as the results in average of 5 executions.

		FF		LPG		VHPOP	
		Orig.	Dec.	Orig.	Dec.	Orig.	Dec.
Blocks 102 pr.	S	77	102	48	86	6	84
	L	78.2	80.9	174.2	166.5	9	9
	TS	2.1	16.2	19.5	11.6	12.2	0.2
	TC		14.3		5.2		0.1
Log 77 pr.	S	77	77	77	77	71	77
	L	121.7	119.6	147.4	136.1	115.6	114.1
	TS	1.9	1.9	0.7	2.7	47.1	95.8
	TC		1.6		1.7		17.9
Elev 150 pr.	S	150	150	150	150	52	150
	L	51.1	51.1	55.6	49.2	20.6	19.6
	TS	0.1	2.5	0.1	7.5	7.8	2.6
	TC		1.5		1.7		0.3
Depot 20 pr.	S	17	20	20	20	2	18
	L	48.8	44.1	65.9	50.2	-	-
	TS	8.2	1.1	8.3	5.1	5.8	0.3
	TC		0.5		4.8		0.2
Total solved		371	397	305	351	131	339

Table 1. Comparison between resolution with and without our decomposition technique (time in secs). Col Orig.: results obtained when solving the original problems. Col Dec.: results obtained when using decomposition. Row *S*: number of solved problems. Row *L*: number of actions in average. Row *TS*: execution time in average in the sequential resolution. Row *TC*: execution time in average in the concurrent resolution.

4 Conclusions

In this paper we have introduced a new decomposition technique for planning problems in STRIPS domains. This technique is fully operative and, as the empirical results show, it can provide important benefits in the resolution of planning problems. Moreover, it is based on well-funded concepts on domain analysis in planning such as the extraction and ordering of landmarks and the notion of property spaces.

As for the further work, we are working on extending this work to the decomposition of temporal planning problems.

REFERENCES

[1] M. Fox and D. Long, 'The automatic inference of state invariants in TIM', *Journal of Artificial Intelligence Research*, **9**, 367–421, (1998).

[2] M. Fox and D. Long. Domains and results of the third international planning competition, 2002. http://www.dur.ac.uk/d.p.long/competition.html.

[3] A. Gerevini and I. Serina, 'Lpg: a planner based on local search for planning graphs', in *AIPS'02*. AAAI Press, (2002).

[4] J. Hoffmann, 'Extending ff to numerical state variables', in *ECAI'02*, pp. 571–575. IOS Press, Amsterdam, (2002).

[5] J. Hoffmann, J. Porteous, and L. Sebastia, 'Ordered landmarks', *Journal of Artificial Intelligence Research (submitted)*, (2004).

[6] J. Porteous, L. Sebastia, and J. Hoffmann, 'On the extraction, ordering, and usage of landmarks in planning', in *ECP'01.*, (2001).

[7] H. Younes and R. Simmons, 'On the role of ground actions in refinement planning', in *Procs of the 6th Int. Conf. on AI Planning and Scheduling (AIPS'02)*. AAAI Press, (2002).