A New Context-Based θ-Subsumption Algorithm

Olga Skvortsova

International Center for Computational Logic, Technische Universität Dresden,
`skvortsova@iccl.tu-dresden.de`

1 Introduction

θ-subsumption is a decidable but incomplete approximation of logical implication, important to inductive logic programming, theorem proving, and most surprisingly, to AI planning. This work is motivated by the area of AI planning which currently lacks the efficient symbolic inference algorithms. These algorithms perform computations of successor and predecessor states of a given state wrt. a given action, where both states and actions are first-order entities [1]. In AI planning, the θ-subsumption problem for states Z_1 and Z_2 is a problem of whether there exists a substitution θ such that $Z_1\theta \subseteq Z_2$, where states Z_1 and Z_2 are sets of literals. θ-subsumption is used as a consequence relation for the decision of whether a state covers the preconditions of an action as well as a redundancy test for detecting which states can be removed from the state space.

In general, θ-subsumption is NP-complete [2]. One approach to cope with the NP-completeness of θ-subsumption is deterministic subsumption. A clause is said to be determinate if there is an ordering of literals, such that in each step there is a literal which has exactly one match that is consistent with the previously matched literals [3]. However, in practice, there may be only few literals, or none at all, that can be matched deterministically. Recently, in [3], it was developed another approach, which we refer to as literal context, LITCON, for short, to cope with the complexity of θ-subsumption. The authors propose to reduce the number of matching candidates for each literal by using the contextual information. The method is based on the idea that literals may only be matched to those literals that possess the same relations up to an arbitrary depth in a clause. As result, a certain superset of determinate clauses can be tested for subsumption in polynomial time.

Unfortunately, as it was shown in [4], LITCON does not scale very well up to large depth. Because in some planning problems, the size of state descriptions can be relatively large, it might be necessary to compute the contextual information for large values of the depth parameter. Therefore, we are strongly interested in a technique that scales better than LITCON. In this paper, we present an approach, referred to as object context, OBJCON, for short, which demonstrates better computational behaviour than LITCON. Based on the idea of OBJCON, we develop a new θ-subsumption algorithm and implement it in our planning system FLUCAP [1].

2 Object Context

In general, a literal l in a state Z_1 can be matched with several literals in a state Z_2, that are referred to as matching candidates of l. LITCON is based on the idea that literals in

Z_1 can be only matched to those literals in Z_2, the context of which include the context of the literals in Z_1 [3]. The context is given by occurrences of identical objects (variables *Vars* (Z) and constants *Const* (Z)) or chains of such occurrences and is defined up to some fixed depth. In effect, matching candidates that do not meet the above context condition can be effortlessly pruned. In most cases, such pruning results in deterministic subsumption, thereby considerably extending the tractable class of states.

The computation of the context itself is dramatically affected by the depth parameter: The larger the depth is, the longer the chains of objects' occurrences are, and thus, more effort should be devoted to build them. Unfortunately, LitCon does not scale very well up to large depth [4]. For example, consider a state $Z = \{on(X, Y), on(Y, table), r(X), b(Y), h(X), h(Y), w(X), d(Y), f(X)\}$ that can be informally read as: A block X is on the block Y which is on the table, and both blocks enjoy various properties, like e.g., color (red r, blue b) or weight (heavy h). Z contains nine literals and only three objects. In LitCon, the context should be computed for each of nine literals in order to keep track of all occurrences of identical objects. What if we were to compute the context for each object instead? In our running example, we would need to perform computations only three times, in this case.

In this paper, we propose a more efficient approach, referred to as ObjCon, for computing the contextual information and incorporate it into a new context-based θ-subsumption algorithm. More formally, we build the object occurrence graph $\mathcal{G}_Z = (V, E, \ell)$ for a state Z, where vertices are objects of Z, denoted as *Obj* (Z) and edges $E = \{(o_1, \pi_1, l, \pi_2, o_2) | l(t_1, \dots, t_n) \in Z \wedge o_1 = t_{\pi_1} \wedge o_2 = t_{\pi_2}\}$ with $o_1, o_2 \in$ *Obj* (Z), $l(t_1, \dots, t_n)$ being a literal and π_1, π_2 being positions of objects o_1, o_2 in l. The labeling function $\ell(o) = \{l | l(o) \in Z\}$ associates each object o with a unary literal name l this object belongs to. The object occurrence graph for the state Z from our running example will contain three vertices X, Y and $table$ with labels $\{r, h, w, f\}$, $\{b, h, d\}$ and $\{\}$, resp., and two edges $(X, 1, on, 2, Y)$ and $(Y, 1, on, 2, table)$.

The object context ObjCon(o, Z, d) of depth $d > 0$ is defined for each object o of a state Z as a chain of labels: $\ell(o) \xrightarrow{\pi_1^1 \cdot f^1 \cdot \pi_2^1} \ell(o_1) \xrightarrow{\pi_1^2 \cdot f^2 \cdot \pi_2^2} \dots \xrightarrow{\pi_1^d \cdot f^d \cdot \pi_2^d} \ell(o_d) \in$ ObjCon(o, Z, d) *iff* $o \xrightarrow{\pi_1^1 \cdot f^1 \cdot \pi_2^1} o_1 \xrightarrow{\pi_1^2 \cdot f^2 \cdot \pi_2^2} \dots \xrightarrow{\pi_1^d \cdot f^d \cdot \pi_2^d} o_d$ is a path in $\mathcal{G}_Z$ of length d starting at o. In our running example, ObjCon$(X, Z, 1)$ of depth 1 of the variable X in Z contains one chain $\{\{r, h, w, f\} \xrightarrow{1 \cdot on \cdot 2} \{b, h, d\}\}$.

Following the ideas of [3], we define the embedding of object contexts for states Z_1 and Z_2, which serves as a pruning condition for reducing the space of matching candidates for Z_1 and Z_2. Briefly, let $OC_1 =$ObjCon(o_1, Z_1, d), $OC_2 =$ObjCon(o_2, Z_2, d). Then OC_1 is embedded in OC_2, written $OC_1 \preccurlyeq OC_2$, *iff* for every chain of labels in OC_1 there exists a chain of labels in OC_2 which preserves the positions of objects in literals and the labels for each object in OC_1 are included in the respective labels in OC_2 up to the depth d. Finally, if ObjCon$(X, Z_1, d) \not\preccurlyeq$ ObjCon(o, Z_2, d) then there exists no θ such that $Z_1 \mu \theta \subseteq Z_2$, where $\mu = \{X \mapsto o\}$. In other words, a variable X in Z_1 cannot be matched against an object o in Z_2 within a globally consistent match, if the variable's context cannot be embedded in the object's context. Therefore, the substitutions that meet the above condition can be effortlessly pruned from the search space. Due to the lack of space, further formal results are omitted here.

algorithm	`BW100`	`BW125`	`BW150`	`BW175`	`BW200`	`BW250`	`BW300`	`BW350`	`BW400`	`BW450`
AllTheta										
d=2	2085	2951	4745	3921	–	–	–	–	–	–
d=3	365	611	1285	834	1815	3513	–	–	–	–
d=4	**117**	**162**	**320**	**172**	**597**	**1264**	5791	–	–	–
d=5	589	713	1015	1050	3421	5182	**2783**	**3914**	–	–
FluCaP										
d=2	54	490	–	–	–	–	–	–	–	–
d=3	13	15	5391	3718	–	–	–	–	–	–
d=4	4	83	1768	972	4236	5017	–	–	–	–
d=5	**3**	**5**	362	11	981	1249	3769	5351	–	–
d=6	3	6	**19**	**10**	**28**	713	1115	2018	2517	–
d=7	5	7	22	14	37	**59**	553	942	**102**	–
d=8	12	15	40	25	78	115	**94**	**71**	163	–
d=9	35	40	99	69	255	395	145	186	605	**618**
d=10	148	124	365	254	1053	–	516	770	3445	4529

Table 1. Comparison between AllTheta and FluCaP. Average timing results in milliseconds for one subsumption test for several instances `BWX` of Blocksworld problems, where `X` stands for the number of blocks in a problem. A dash means that the algorithm did not finish within 100 minutes. The best results are marked in bold.

Table 1 depicts the comparison timing results between the LitCon-based subsumption reasoner, referred to as AllTheta, and its ObjCon-based opponent, referred to as FluCaP. The results were obtained using RedHat Linux running on a 2.4GHz Pentium IV machine with 2GB of RAM. We demonstrate the advantages of exploiting the object-based context information on problems that stem from the extended version of the classical Blocksworld planning scenario. For each problem, there have been done 1000 subsumption tests. The time limit of 100 minutes has been allocated. The results show that FluCaP scales better than AllTheta on large problems. E.g., AllTheta could solve problems of size up to 350 blocks only. Whereas FluCaP easily scales further. We believe that it happens because FluCaP is less sensitive to the growth of the depth parameter. Under the condition that the number of objects in a state is strictly less than the number of literals and other parameters are fixed, the amount of object-based context information is strictly less than the amount of the literal-based context information. Moreover, FluCaP requires two orders of magnitude less time than AllTheta.

References

1. Hölldobler, S., Karabaev, E., Skvortsova, O.: FluCaP: A heuristic search planner for first-order MDPs. JAIR (2006) To appear.
2. Kapur, D., Narendran, P.: NP-completeness of the set unification and matching problems. In: CADE. (1986)
3. Scheffer, T., Herbrich, R., Wysotzki, F.: Efficient θ-subsumption based on graph algorithms. In: ILP Workshop. (1996)
4. Karabaev, E., Rammé, G., Skvortsova, O.: Efficient symbolic reasoning for first-order MDPs. In: ECAI'2006 Workshop. (2006)