11.7 PLANNING WITH PARTIALLY INSTANTIATED OPERATORS

The version of POP in Figure 11.13 outlines the algorithm, but leaves some details unspecified. In particular, it does not deal with variable binding constraints. For the most part, all this entails is being diligent about keeping track of binding lists and unifying the right expressions at the right time. The implementation techniques of Chapter 10 are applicable here.

There is one substantive decision to make: in RESOLVE-THREATS, should an operator that has the effect, say, $\neg At(x)$ be considered a threat to the condition $At(Home)$? Currently we can distinguish between threats and non-threats, but this is a **possible threat**. There are three main approaches to dealing with possible threats:

POSSIBLE THREAT

- **Resolve now with an equality constraint:** Modify RESOLVE-THREATS so that it resolves all possible threats as soon as they are recognized. For example, when the planner chooses the operator that has the effect $\neg At(x)$, it would add a binding such as $x = HWS$ to make sure it does not threaten $At(Home)$.
- **Resolve now with an inequality constraint:** Extend the language of variable binding constraints to allow the constraint $x \neq Home$. This has the advantage of being a lower commitment—it does not require an arbitrary choice for the value of x—but it is a little more complicated to implement, because the unification routines we have used so far all deal with equalities, not inequalities.
- **Resolve later:** The third possibility is to ignore possible threats, and only deal with them when they become *necessary* threats. That is, RESOLVE-THREATS would not consider $\neg At(x)$, to be a threat to $At(Home)$. But if the constraint $x = Home$ were ever added to the plan, then the threat would be resolved (by promotion or demotion). This approach has the advantage of being low commitment, but has the disadvantage of making it harder to decide if a plan is a solution.

Figure 11.14 shows an implementation of the changes to CHOOSE-OPERATOR, along with the changes to RESOLVE-THREATS that are necessary for the third approach. It is certainly possible (and advisable) to do some bookkeeping so that RESOLVE-THREATS will not need to go through a triply nested loop on each call.

When partially instantiated operators appear in plans, the criterion for solutions needs to be refined somewhat. In our earlier definition (page 349), we were concerned mainly with the question of partial ordering; a solution was defined as a partial plan such that all linearizations are guaranteed to achieve the goal. With partially instantiated operators, we also need to ensure that all instantiations will achieve the goal. We therefore extend the definition of achievement for a step in a plan as follows:

> A step S_i **achieves** a precondition c of the step S_j if (1) $S_i \prec S_j$ and S_i has an effect that necessarily unifies with c; and (2) there is no step S_k such that $S_i \prec S_k \prec S_j$ in some linearization of the plan, and S_k has an effect that possibly unifies with $\neg c$.

The POP algorithm can be seen as constructing a proof that each precondition of the goal step is achieved. CHOOSE-OPERATOR comes up with the S_i that achieves (1), and RESOLVE-THREATS makes sure that (2) is satisfied by promoting or demoting possible threats. The tricky part is that