Figure 11.8 A hierarchical planning algorithm that uses angelic semantics to identify and commit to high-level plans that work while avoiding high-level plans that don't. The predicate MAKING-PROGRESS checks to make sure that we aren't stuck in an infinite regression of refinements. At top level, call ANGELIC-SEARCH with [Act] as the initialPlan.

The ability to commit to or reject high-level plans can give ANGELIC-SEARCH a significant computational advantage over HIERARCHICAL-SEARCH, which in turn may have a large advantage over plain old BREADTH-FIRST-SEARCH. Consider, for example, cleaning up a large vacuum world consisting of rectangular rooms connected by narrow corridors. It makes sense to have an HLA for Navigate (as shown in Figure 11.4) and one for CleanWholeRoom. (Cleaning the room could be implemented with the repeated application of another HLA to clean each row.) Since there are five actions in this domain, the cost for BREADTH-FIRST-SEARCH grows as $5^d$, where $d$ is the length of the shortest solution (roughly twice the total number of squares); the algorithm cannot manage even two $2 \times 2$ rooms. HIERARCHICAL-SEARCH is more efficient, but still suffers from exponential growth because it tries all ways of cleaning that are consistent with the hierarchy. ANGELIC-SEARCH scales approximately linearly in the number of squares—it commits to a good high-level se-
In this section we extend planning to handle partially observable, nondeterministic, and unknown environments. Chapter 4 extended search in similar ways, and the methods here are also similar: sensorless planning (also known as conformant planning) for environments with no observations; contingency planning for partially observable and nondeterministic environments; and online planning and replanning for unknown environments.

While the basic concepts are the same as in Chapter 4, there are also significant differences. These arise because planners deal with factored representations rather than atomic representations. This affects the way we represent the agent’s capability for action and observation and the way we represent belief states—the sets of possible physical states the agent might be in—for unobservable and partially observable environments. We can also take advantage of many of the domain-independent methods given in Chapter 10 for calculating search heuristics.

Consider this problem: given a chair and a table, the goal is to have them match—have the same color. In the initial state we have two cans of paint, but the colors of the paint and the furniture are unknown. Only the table is initially in the agent’s field of view:

\[
\text{Init(} \text{Object( Table) } \land \text{Object( Chair) } \land \text{Can}(C_1) \land \text{Can}(C_2) \land \text{InView( Table)})
\]

\[
\text{Goal( Color } x, e) \land \text{A Color( Table, r))}
\]

There are two actions: removing the lid from a paint can and painting an object using the paint from an open can. The action schemas are straightforward, with one exception: we now allow preconditions and effects to contain variables that are not part of the action’s variable