Utilizing Problem Structure in Planning: A Local Search Approach

Jörg Hoffmann

Institut für Informatik
Georges-Köhler-Allee, Geb. 52
79110 Freiburg, Germany
hoffmann@informatik.uni-freiburg.de

Abstract. The book deals with general problem solving in the form given by the STRIPS planning formalism. Planning systems read in a STRIPS problem description, and then perform the search for a solution fully automatically, i.e., without any knowledge about the problem other than the transition rules that generate the search space. Starting from an existing approach to planning as heuristic search, I develop a novel system based on local search with a refined heuristic function. The system, named *FF* (short for Fast-Forward), achieves dramatic runtime improvements over all previous systems in a wide range of benchmark domains. A subsequent investigation determines the reasons for this performance: as it turns out, most of the benchmarks share certain kinds of problem structure. These structures imply certain characteristic qualities of FF's heuristic function, which in turn explain FF's behavior. I discuss the consequences of my findings on ways of benchmarking in planning.

1 Planning

Planning is a form of general problem solving. It has been one of the main topics of interest in AI from the outset. The task in the field is to design problem solving techniques that can be applied to arbitrary problem domains and that thus model, or imitate, one of the distinguishing qualities of humans: the ability to reason about unseen domains. The overall goal in the field can be formulated as making such general techniques competitive, within realistic application domains, with specialized solvers. General problems in this context are formulated by means of an *initial state*, a *goal*, and a set of *actions*. In the simplest and most wide-spread formalism, STRIPS, states are sets of propositional facts (those that are true in the state at hand). An action can be *applied* in a state if its *precondition* – a set of facts – is true in the state. When applying the action, its *positive effects* (a set of facts) are added into the state, while its *negative effects* (also a set of facts) are deleted. A *plan* is then a sequence of actions that transforms the initial state into a goal state – a state that satisfies the goal (in the STRIPS case, a conjunction of facts).

The benchmark problems traditionally used for evaluation in planning are simplified formal adoptions of real-world problems, like e.g. transportation tasks (objects must be transported to different locations by means of vehicles) or construction tasks (complex objects must be assembled from their parts). Related planning tasks are *instances* of the same *planning domain*. A planning domain constitutes infinitely many instances. The book considers 20 of the most frequently used planning domains. These domains cover the band width of the traditional planning benchmarks.

2 A Local Search Approach

Blai Bonet and Héctor Geffner [2] have recently proposed an approach to planning as heuristic search, implemented in the *HSP* system. Search takes place in the (forward) *state space*, i.e., in the set of all states that are reachable from the initial state through action applications. Search is controlled by a *heuristic function* that, for each search state s, estimates the state's *goal distance* (the number of action applications needed to reach the nearest goal state from s). To obtain such a heuristic estimation the problem is *relaxed*, i.e., simplified. The difficulty of solving the relaxed problem (starting from s) then gives an estimate of how difficult it is to solve the real problem (starting from s). The relaxation proposed by Bonet and Geffner is to ignore the negative effects of all actions. The goal distance estimate for a state s is then the length of a shortest relaxed plan for s. I will henceforth refer to the length of such a shortest relaxed plan with $h^+(s)$. Computing the h^+ function is NP-hard so Bonet and Geffner have proposed an approximation technique (which I do not describe in detail here). The resulting heuristic estimates are used in a standard *hill-climbing* search: starting in the initial state, iteratively look at all direct successors of the current state and pick one with lowest heuristic value. The FF system features the following three modifications.

1. I introduce a refined heuristic function. The h^+ approximation used in HSP completely ignores *positive interactions* between different facts and actions. In a number of domains this leads to considerable overestimation of the real goal distance. I therefore develop a new h^+ approximation that can take account of positive interactions. The technique explicitly solves the relaxed problem in each search state, using a Graphplan-style [1] approach. The length of the relaxed plans (which can take advantage of positive interactions) forms FF's heuristic function.
2. The second algorithmic technique is a novel local search scheme, combining local and systematic search. The search scheme is motivated by the observation that, in a number of benchmark domains, there often *are* states with strictly better heuristic value near the current search state, but not among its *immediate* successors. Thus the idea is, when facing a state s during a hill-climbing search, to perform a systematic lookahead for a (possibly indirect) successor state s' with strictly better heuristic value, $h(s') < h(s)$. The lookahead is implemented via a standard breadth-first search.
3. The third algorithmic technique prunes the search space. I observe that the relaxed plans often contain exactly those actions that must in fact be used in the respective search states. FF thus considers only those successors of a search state that are generated by actions contained in the state's relaxed plan.

As it turns out, these modifications are very effective in the benchmarks. Running FF against HSP, and against one representative of each previous approach to planning, I find that FF demonstrates much better asymptotic runtime behavior (exponentially, in 7 of the 15 domains) than any of the other systems. The quality (number of actions) of the found plans is comparable. Similar observations were made in the *2nd International Planning Competition* (carried out alongside AIPS-2000): in the 5 benchmark domains used there, FF demonstrated superior performance and was the only fully-automated planner to be awarded the title *Group A Distinguished Performance Planning System* (also winning two first prizes sponsored by *Celcorp* and *Schindler Lifts Inc*). Considering these results, and the fact that FF differs from HSP only by the three techniques outlined above, an important question is which of these techniques are responsible for the performance gain. I clarify this issue by an empirical analysis: the new and old techniques are evaluated in all eight possible combinations, and the analysis of (statistically) significant performance improvements shows that the new search scheme, and its interaction with the pruning technique, tend to be most important in the 20 domains looked at.

3 Problem Structure

The only way a reasoning technique can be "fast" in solving an intractable problem is by exploiting the structure of the specific problem instances it is evaluated on. So what is the problem structure that makes FF so fast in the traditional planning benchmarks? I answer this question through an investigation of the *local search topology* of these benchmarks. The local search topology of an example instance is given by the set of search states and their heuristic values. Intuitively, when interpreting the heuristic values as heights what we get is a search space surface. A local search procedure then moves along this surface, trying to get to a state with height 0. The effect of local search topology on the performance of local search has been investigated, e.g., in the context of satisfiability testing (SAT) [3]. For the first time, these techniques are now applied to AI planning. Looking at the properties of the h^+ function (which both HSP and FF approximate), I prove the following.

1. In 15 of the 20 domains the state spaces do not contain any *unrecognized dead ends*, i.e., states from which the goal is unreachable but for which there is a relaxed plan.
2. In 13 of these 15 domains the state spaces do not contain any *local minima*, i.e., regions that are no solutions but whose neighbors all have a higher heuristic value.
3. In 8 of these 13 domains there is a constant upper bound on the *maximal exit distance*; the maximal exit distance in a state space is defined as the distance to a state with better heuristic value, maximized over all states that do not lie on a local minimum.

These are obviously very advantageous qualities of a heuristic function. In fact it is easy to see that FF's local search scheme, if the heuristic function has all these three qualities, finds a goal state after looking at polynomially many states. One of the 8 domains where the h^+ function has the three qualities is the *Logistics* domain, which is the classical transportation benchmark in planning. Through the process of proving, I also identify what the characteristic problem structures are that cause the heuristic quality. They are the following.

1. In 14 of the 20 domains all actions are either (at least) invertible, or have irrelevant negative effects and static positive effects (implying that they do not need to be inverted in order to avoid dead ends).
2. In 10 of these 14 domains all actions have the property that, if they start a shortest real solution from a state, they also start a shortest relaxed solution from that state.
3. In 8 of these 10 domains one part of the actions have negative effects that are no longer needed once the action has been applied on a shortest solution path, while at most a constant number of the other actions must be applied consecutively on such a path.

The first pattern of structure implies the non-existence of dead ends. The second and first structures together imply the non-existence of local minima under h^+. With the third pattern of structure on top of that, the maximal exit distance under h^+ is bound by a constant. Beyond these general implications, I identify one domain where all dead ends are recognized, as well as three more domains where there are no local minima under h^+ due to somewhat more complicated reasons. By an empirical investigation that takes samples from the state spaces of larger example instances, I show that the qualities of the h^+ function are largely preserved by the approximative heuristic function that is used in FF (remember that h^+ is hard to compute).

Considering these results, one important question is how relevant all of this is in practice: are the observed problem structures, and the resulting performance achievements, a phenomenon that occurs in real-world applications of planning, or are they only a phenomenon that occurs in the, perhaps unrealistically simple, traditional planning benchmarks? Probably this question is best answered by designing new planning benchmarks that are as realistic as possible. And this is precisely what Stefan Edelkamp and the author are trying to do as the organizers of the next international planning competition to be carried out alongside ICAPS-04.

Acknowledgments

I would like to thank my PhD supervisor, Bernhard Nebel, for his continued support.

References

1. Avrim L. Blum and Merrick L. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90(1-2):279–298, 1997.

2. Blai Bonet and Héctor Geffner. Planning as heuristic search. *Artificial Intelligence*, 129(1–2):5–33, 2001.
3. Jeremy Frank, Peter Cheeseman, and John Stutz. When gravity fails: Local search topology. *Journal of Artificial Intelligence Research*, 7:249–281, 1997.