Improving Local Search for Resource-Constrained Planning

Hootan Nakhost, Joerg Hoffmann, Martin Müller

Improving Local Search for Resource-Constrained Planning

Hootan Nakhost
University of Alberta, Edmonton, Canada
nakhost@ualberta.ca

Jörg Hoffmann
INRIA, Nancy, France
joerg.hoffmann@inria.fr

Martin Müller
University of Alberta, Edmonton, Canada
mmueller@ualberta.ca

Resource-Constrained Planning

The need to economize limited resources such as energy, fuel, money, and/or time, is ubiquitous in many of the classical applications of planning. Although planning with resources is a long-standing topic, the progress made in solving such problems has been limited. *Relaxation heuristics* solve a relaxed planning problem in which all "negative effects" of the operators, such as resource consumption, are ignored. Almost all state of the art heuristic functions for planning are based on such relaxations and therefore *completely ignore the need to economize resources*. To address this difficulty, one could try to develop better-suited heuristics. We consider the alternative of developing better-suited search methods.

Let C denote the ratio between the amount of available resources and the minimum amount required to solve an instance. Planning problems become intuively "harder" as C approaches 1. (Hoffmann et al. 2007) observed that otherwise very successful planners perform very poorly when C is close to 1. The research started in this current work attempts to improve this sad situation.

Examples of resource-based planning benchmarks from the International Planning Competitions (IPC) are *Mystery* and *Mprime* from IPC'98, which encode transportation with fuel consumption. *Trucks* in IPC'08 features time as a resource, since actions take time and trucks have strict delivery deadlines. *Satellite* in IPC'02 features fuel consumption as a resource; however, resources are missing in the STRIPS version of this domain. Several other IPC domains feature resource consumption not as a hard constraint, but only as an optimization criterion, which is ignored by most satisficing planners.

Resource constrainedness C is not a controlled quantity in these IPC benchmarks. In most cases, C is not even known. A well-suited banchmark must provide a range of instances with different values of C, while keeping all other settings the same. The main empirical basis of our investigation is the simple transportation domain of Hoffmann et al (2007), called *NoMystery* here. In *NoMystery*, a truck moves in a weighted graph; a set of packages must be transported between nodes; actions move along edges, and load/unload packages; each move consumes the edge weight in fuel. A problem generator allows to control (fuel) constrainedness C. The generator creates a random connected undirected graph with n nodes, and adds k packages with random origins and destinations. The edge weights are uniformly drawn between 1 and 25. A domain-specific branch-and-bound procedure computes the minimum required amount of fuel M, and the initial fuel supply is set to $\lfloor C * M \rfloor$. Our experiments use 5 random instances with $n = 12$ and $k = 12$, and let C range in $\{1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 2.0\}$ for a total of 35 test cases. This setup assures that results across different values of C are exclusively due to the level of resource constrainedness.

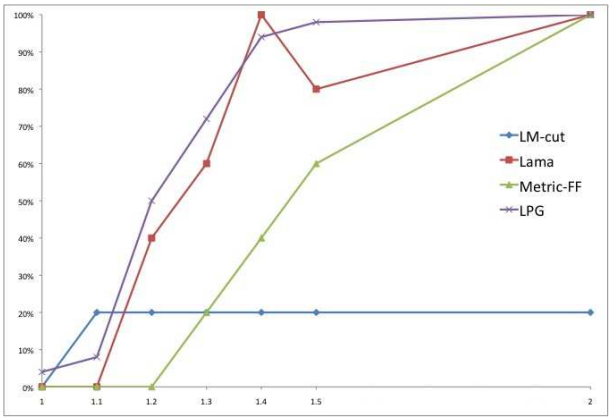

Figure 1: Average coverage of state-of-the-art planners when varying resource constrainedness C. 2 GB memory, 40 minutes time limit. computation in case of a time-out.

Results for a set of current leading planners are depicted in Figure 1. All the satisficing planners excel when C is large, but degrade to be rather useless as C approaches 1.

Adding Smart Restarts to the Arvand Planner

Arvand (Nakhost and Müller 2009) is a stochastic planner that combines random sampling with heuristic evaluation. Arvand can benefit from good heuristic values, but still, to some extent, overcome misleading ones thanks to the speed of random sampling.

Starting at the initial state, at each "search step" Arvand choses its next state from a number of states sampled in a

local neighborhood. Each sample is obtained by running a bounded random walk, a length-limited sequence of randomly selected actions. Only the endpoints of the walks are evaluated using h^{FF}, the FF heuristic (Hoffmann and Nebel 2001). Arvand transitions to an endpoint whose heuristic value is minimal. Since the expensive heuristic evaluation is computed only at endpoints, Arvand can perform a large number of random walks compared to a planner that evaluates after each action. Arvand continues performing search steps in this fashion, until either a goal state is reached or the search gets stuck: either the heuristic value does not improve over several search steps, or all random walk endpoints s are dead-ends where either no actions are applicable or $h^{FF}(s) = \infty$. If the search gets stuck it restarts from the initial state, beginning a new "search episode".

The new technique we added in *ArvandSR* are *smart restarts*: it maintains a pool of the p "best" previous search episodes, that ultimately reached the smallest heuristic values) and restarts from a random state along those search trajectories, not from the initial state. In this way, *ArvandSR* balances exploitation of previous runs against the risk of repeating previous mistakes.

Experiments

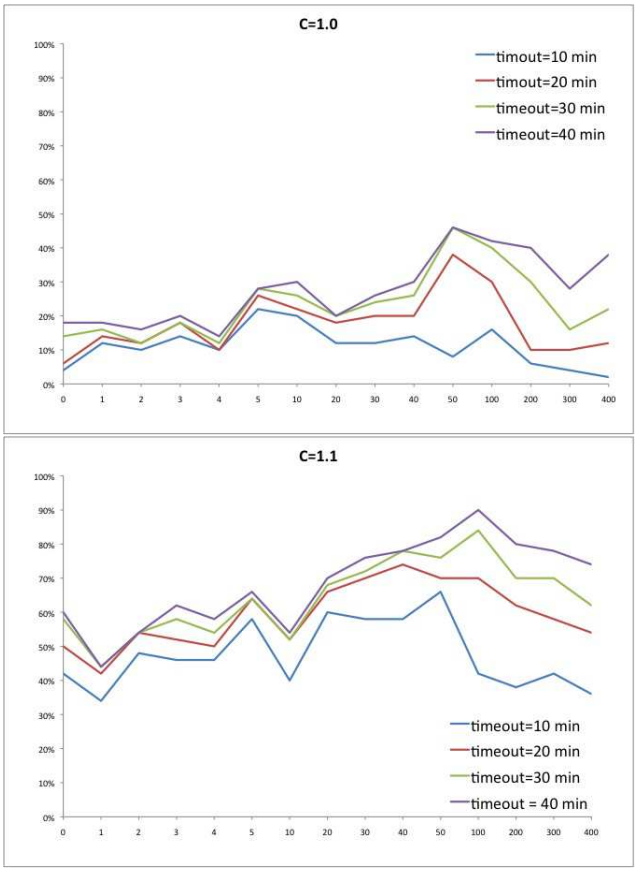

Figure 2: Coverage of *ArvandSR* when varying p (on the x-axis) and timeout; for $C = 1.0$ (a) and $C = 1.1$ (b).

Large-scale experiments with *ArvandSR* in NoMystery modify three relevant parameters: the constrainedness of resources C; the size p of the pool for smart restarts; and a parameter w controlling the trade-off between Arvand's two search strategies MDA and MHA (Nakhost and Müller 2009). Detailed experiments and analyses are given in the accompanying technical report (Nakhost, Hoffmann, and Müller 2010). The data in Figure 2 and in the technical report clearly shows that:

1. **Original Arvand already significantly outperforms the state of the art for tightly constrained resources.** With $p = 0$, at the runtime cut-off 40 minutes used in Figure 1, coverage for $C = 1.0$ is 18% compared to 4% for LPG, and for $C = 1.1$ it is 60% compared to 8% for LPG.
2. **Smart restarts can significantly improve Arvand for tightly constrained resources.** With $p > 0$, coverage for $C = 1.0$ improves to 46% at $p = 50$, and for $C = 1.1$ it reaches 90% at $p = 100$.
3. **The benefit of smart restarts tends to grow as C tends to 1.** This becomes clear when extending Figure 2 with larger C values (Nakhost, Hoffmann, and Müller 2010).

Cross-checking these results on a subset of IPC benchmarks with a resource or puzzle nature shows that in these domains, smart restarts rarely hurt, and sometimes help significantly (Nakhost, Hoffmann, and Müller 2010).

Conclusions and Future Work

Economizing limited resources is an important situation to be considered for automated planning. Our experiments, extending those of Hoffmann et al (2007), clearly show that current state-of-the-art planners are very bad at doing so. We propose local search as a potential way out, and have shown that a simple improvement to Arvand, smart restarts, can already make a big difference. This result is encouraging, yet is only the first step in comprehensively addressing planning under severe resource constraints. Some important open points are:

- Experiment with additional resource-constrained planning domains, equipped with to-be-developed generators allowing to control C like in NoMystery.
- Automatically configure the search, through a static or dynamic scheme for controlling the pool size p.
- Explore other enhancements to local search, such as UCT for prioritizing states in the pool.

References

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *J. Artificial Intelligence Research* 14:253–302.

Hoffmann, J.; Kautz, H.; Gomes, C.; and Selman, B. 2007. SAT encodings of state-space reachability problems in numeric domains. In *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, 1918–1923.

Nakhost, H., and Müller, M. 2009. Monte-Carlo exploration for deterministic planning. In *Proc. 21st International Joint Conference on Artificial Intelligence (IJCAI-09)*, 1766–1771.

Nakhost, H.; Hoffmann, J.; and Müller, M. 2010. Improving local search for resource-constrained planning. Technical Report TR 10-02, Dept. of Computing Science, University of Alberta, Edmonton, Alberta, Canada.