Tuning Search Heuristics for Classical Planning with Macro Actions

I. Murugeswari and **N. S. Narayanaswamy**
Department of Computer Science and Engineering
Indian Institute of Technology Madras, India
email: eswari@cse.iitm.ac.in and swamy@cse.iitm.ac.in

Abstract

This paper proposes a new approach to improve domain independent heuristic state space search planners for classical planning by tuning the search heuristics using macro actions of length two extracted from sample plans. This idea is implemented in the planner AltAlt and the new planner Macro-AltAlt is tested on the domains introduced for the learning track of the International Planning Competition (IPC-2008). The performance of Macro-AltAlt measured by the length of the plan found and the number of states explored to find the plan is compared with that of AltAlt.

Introduction

Domain independent heuristic state space search planners for classical planning (Ghallab, Nau, and Traverso 2004) work by searching the state space with the guidance of a heuristic function. We explore the possibility of improving the search process by tuning the heuristic values of states using domain specific knowledge gained through experience in solving problems in a domain. We do this by learning macro actions and their frequencies in a domain independent manner from plans for sample problems in a domain. A macro action is defined as a sequence of actions in which consecutive actions work on a common object in the problem. A sequence of actions occurring repeatedly in plans for different problems in a domain indicates that it is characteristic of plans for problems in the underlying domain to have these actions together in the same order. Therefore, it might be useful to bias the search to give preference to actions that form a highly frequent macro action learnt from experience. This can be done by appropriately adjusting the heuristic values of states based on the frequency of the macro action that generates it.

Related Work

Improving domain independent planners with domain specific details and identifying useful macro actions for aiding problem solvers have been approached in ways different from that of ours. STAN (Long and Fox 1999), an implementation of graphplan employs a preprocessing module TIM (Type Inference Module) that performs domain analysis for identifying domain features. MacroFF (Botea et al. 2005), a forward state space search planner extracts macro actions through domain analysis and extends the domain operator set with macro actions. Jeremy Frank (Frank 2007) has talked about using data mining techniques for identifying macro actions. Richard E. Korf (Korf E. 1985) and Glenn A. Iba (Iba A. 1989) have used macro actions in general problem solving by abstracting them into domain operators.

Our Work

We have implemented our idea of using macro actions to improve search in AltAlt (Nigenda, Nguyen, and Kambhampati 2000). The new planner Macro-AltAlt is organized as a system with two components: A Learner and a Planner as shown in Figure-1.

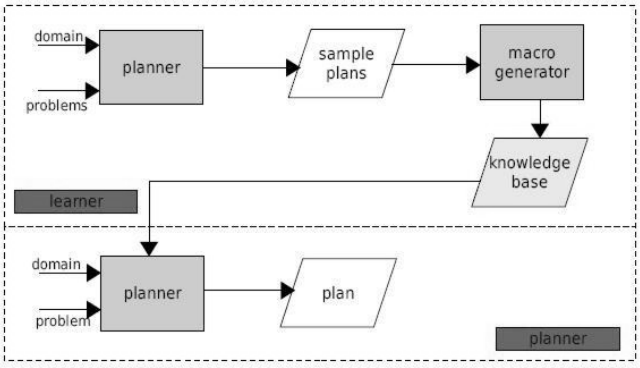

Figure 1: Structure of Macro-AltAlt

Learner The Learner is an off-line component that is run before the planner. It learns macro actions in a domain independent manner from a set of sample plans. The set of plans can come from any source: a domain expert or a planning program. We have made a few implementation level changes to AltAlt primarily to reduce memory requirements. We have used the modified version of AltAlt to generate this set. A macro extractor module reads this set and creates

a knowledge base of macro actions using the apriori (Han and Kamber 2006) data mining algorithm. The steps in the macro action extraction algorithm are given below:

1. Create a list of actions (considering only the action name) in the plan and determine the frequency of each action.
2. Associate a list with each action and save the position(s) of its occurrence(s) in the plan.
3. For each action in the list created in step-1 do:
   (a) For each entry in the associated occurrence list created in step-2 do:
      i. If this action and the consecutive action in the plan have a common argument then add the pair of actions to the list of macro actions (if it is not in the list) and update the frequency.
4. Group the macro actions according to the first action in the pair. Rank the macro actions in a group in the ascending order of their frequencies.

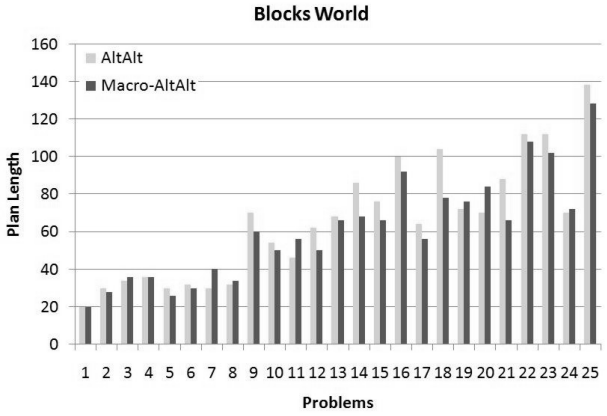

Figure 2: Macro-AltAlt vs AltAlt (Plan Length)

Planner The Planner is the improved version of AltAlt. It takes the knowledge base of the domain as an additional input. AltAlt performs best first search. In each step, the best node among the unexpanded nodes is expanded. The steps in the algorithm to tune the heuristic value are given below:

1. Let *s* be the current node. Generate the successors of *s* and compute the heuristic value of each successor.
2. Offset the heuristic value of each successor by the rank of the macro action $\langle a, x \rangle$ where *a* is the action that generated *s* and *x* is the action that generated the successor node.
3. Consider the new heuristic values of the successor nodes while choosing the best node for expansion in the next step.

Experiments and Observations

We have tested Macro-AltAlt on a set of 25 problems on the following five domains: Blocks World, Matching Blocks World, Gold Miner, Sokoban and N-Puzzle. Macro-AltAlt is compared with AltAlt in the following performance measures: Plan Length and States Explored. Out of the five domains, Macro-AltAlt performs well in the Blocks World domain as shown in the plots in Figure-2 and Figure-3. The results indicate a correlation between the ratios of the lengths of plans found by the two planners and the ratios of the number of states explored by them. This has to be studied further.

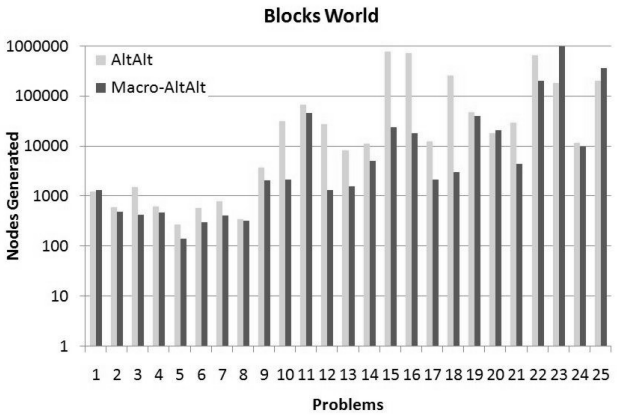

Figure 3: Macro-AltAlt vs AltAlt (Nodes Generated)

Acknowledgements

We are thankful to Dr. Deepak Khemani and Dr. B. Ravindran in the Department of Computer Science and Engineering, Indian Institute of Technology Madras, for their advice and support.

References

Botea, A.; Enzenberger, M.; Muller, M.; and Schaffer, J. 2005. Macro-ff: Improving ai planning with automatically learned macro-operators. *Journal of Artificial Intelligence Research 24* 581–621.

Frank, J. 2007. Using data mining to enhance automated planning and scheduling. *Computational Intelligence and Data Mining, 2007* 251–260.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning - theory and practice*. Morgan Kaufmann.

Han, J., and Kamber, M. 2006. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2^{nd} edition.

Iba A., G. 1989. A heuristic approach to the discovery of macro-operators. *Machine Learning 3* 285–317.

Korf E., R. 1985. Macro-operators: A weak method for learning. *Artificial Intelligence 26* 35–77.

Long, D., and Fox, M. 1999. Efficient implementation of the plan graph in stan. *Journal of Artificial Intelligence Research 10* 87–115.

Nigenda, R. S.; Nguyen, X.; and Kambhampati, S. 2000. Altalt: Combining the advantages of graphplan and heuristic state search. Technical report, In KBCS-00.