Review: FF The Fast-Forward Planning System

http://www.cs.toronto.edu/~sheila/2542/w06/readings/ffplan01.pdf

Overview

Fast-Forward (FF) improved upon Heuristic Search Planner (HSP) with improved heuristics, search, and pruning. The results were significant improvements on a majority of the planning benchmarks in both time and plan length. FF was the most successful automatic planner in the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS'00) planning systems competition.

Heuristics

Instead of HSP's weighting of goals, based on the weights of facts, FF implements a relaxed version of GRAPHPLAN while ignoring delete lists. It first builds the planning graph until all goals are reached, and then backtracks by working on all goals missing in each state and adding actions which generate a missing goal. For each action it adds the preconditions as goals in the previous state, and moves to the previous state, working backwards.

Search

Instead of the random hill climbing used by HSP, FF implements an enforced form of hill climbing. FF evaluates the heuristic value of all direct successor states to the current state in a breadth-first style. If none of the those states has a better heuristic value then it moves to the successors of those successor states, and so on until a better heuristic value is found at state S'. It then adds the path to S' to the plan, and starts the process again from S'.

Pruning

FF prunes the search tree by looking for "helpful actions". Helpful actions are those applicable actions that add at least one goal at the lowest layer of the relaxed solution. When it finds helpful actions FF only explores those branches of the search tree. If enforced hill climbing using this pruning technique fails to find a solution, then FF simply switches to a complete weighted A* algorithm.

Conclusion

In the author's study comparing FF's performance with HSP, he ran benchmarks against HSP with different combinations of FF's heuristic, search, and pruning features turned on or off to try to measure the impact of each improvement over HSP separately.

FF's estimates improve run-time performance in about half of the planning benchmarks when using only FF's heuristics and leaving FF's search and pruning turned off. FF also generates shorter plan lengths in many more cases when helpful actions are used. When both enforced hill climbing and helpful actions are used, FF is faster in 16 out of 20 domains, and often finds better solutions. Helpful action pruning improves runtime performance significantly in the majority of cases regardless of whether FF's search and pruning are used. In the case where helpful actions are used with enforced hill climbing and HSP's heuristic there are rare cases where FF will find a significantly longer plan.

Review: The LAMA Planner Using Landmark Counting in Heuristic Search

https://arxiv.org/ftp/arxiv/papers/1401/1401.3839.pdf

Overview

LAMA is a classical planning system based on heuristic forward search, which improved upon and was influenced by HSP, FF, and Fast Downward. Its core feature is the use of a pseudo-heuristic derived from landmarks, propositional formulas that must be true in every solution of a planning task. LAMA showed best performance among all planners in the sequential satisficing track of the International Planning Competition 2008.

Translation

The purpose of the translator is to transform the planner PDDL into a multi-valued state representation similar to the SAS+ formalism. LAMA performs translation from binary to finite-domain variables, so instead of "parcel p1 is at location A" LAMA uses state variables like "the location of car c1" or "the state of parcel p1". The translator component of LAMA is very similar to that of Fast Downward with minor modifications.

Knowledge Compilation

Using the multi-valued task representation generated by the translator, LAMA builds a number of data structures which facilitate landmark generation and search. For example, domain transition graphs (DTGs) encode for each finite-domain state variable the ways in which it may change its value through actions. LAMA also creates "successor generators" and "axiom evaluators", which are data structures for efficiently determining the set of applicable actions in a given state and evaluating the values of derived state variables.

Search

LAMA's search attempts to find a plan using heuristic search with some enhancements, such as the use of preferred operators (similar to helpful actions in FF) and deferred heuristic evaluation, which mitigates the impact of large branching factors. Deferred heuristic evaluation means that states are not evaluated upon generation, but upon expansion. LAMA first runs a greedy best-first search, aimed at finding a solution as quickly as possible. Once a plan is found, it searches for progressively better solutions by running a series of weighted A$*$ searches with decreasing weight. The cost of the best known solution is used for pruning the search, while decreasing the weight makes the search progressively less greedy, trading speed for solution quality. The best solution found in the available time is then reported.

Conclusion

LAMA is one of the best performing of the current classical planning systems, winning the 2008, 2011 International Planning Competition (IPC) for the sequential satisfying track. In 2014 it was not entered, but would have been 12th out of 21 planners if it competed.

Review: LLAMA: Learning LAMA

http://www.cs.colostate.edu/~ipc2014/ipcl2014description-llama.pdf

Overview

LAMA is a classical planning system based on heuristic forward search, which improved upon and was influenced by HSP, FF, and Fast Downward. However LAMA's heuristics ignore the real cost of the actions and return instead an estimate of the plan length to the goal. The main advantage of these heuristics compared with real-cost heuristics is that they solve a greater number of problems, but heuristics that predict the real cost should find solutions of better quality.
To increase the effectiveness of real-cost heuristics and reduce the impact of their drawbacks without losing quality, LLAMA uses machine learning techniques to automatically obtain good combinations of several heuristics per domain.

Training

Training is domain-dependent, and generates models specific to each domain. A set of simple problems for each domain is generated, and ideal solutions are used to generate labeled data for training the model for each domain. The model is used to determine the configuration of heuristics used for each domain.

The ideal solutions used for training were generated using Fast Downward Stone Soup (FDSS), LAMA11, and Multi-Heuristic First Solution (MHFS).

LLAMA uses linear regression for training. Linear Regression (LR) models are linear functions that minimize the sum of squared residuals of the model. Several other non-linear regression models were evaluated, but linear regression proved sufficient and was noticeably faster.

Heuristics Suite

Several heuristics are used to generate estimates of distance and cost to goals from states. These are the Additive heuristic (Add), the Blind heuristic, the Causal graph heuristic (CG), the Context-enhanced additive heuristic (CEA), the Fast Forward heuristic (FF), the Goal count heuristic, the Landmark count heuristic (LM-Count), the Landmark-cut heuristic (LM-Cut), and the Max heuristic. Attribute selection is used to eliminate correlated heuristics for particular domains. Following training, a linear combination using learned weights defines the domain dependent heuristic, along with an alternation multiple queue that uses the heuristics selected during the learning process (ASH), instead of using the learned model.

Conclusion

The evolution of planners over the last 17 years has led through Fast Forward and Fast Downward to LAMA and now LLAMA. With each generation, there has been gradual improvement. Two of the main directions in planning currently are using learning to create domain-specific planners, and using ensembles or portfolios of planners to select planners which perform best in particular domains. LLAMA takes the first approach, learning the effectiveness of a suite of possible heuristics and applying a linear combination of the effective heuristics for each domain. Planners such as MIPlan take the second approach, and also achieve good results.