CFDP: an approach to Cost-Optimal Planning based on FDP

Stéphane Grandcolas et Cyril Pain-Barre
LSIS – UMR CNRS 6168
Domaine Universitaire de Saint-Jérôme
Avenue Escadrille Normandie-Niemen
13397 MARSEILLE CEDEX 20 France
{stephane.grandcolas,cyril.pain-barre}@lsis.org

CFDP is a planning system based on the paradigm of planning as constraint satisfaction, that searches for cost-optimal plans. Basically it is an *Iterative Deepening Depth-First Search* procedure (Korf 1985) like its predecessor FDP (Grandcolas & Pain-Barre 2007) (FDP produces optimal sequential plans). CFDP works directly on a structure related to Graphplan's planning graph: given a fixed bound on the length of the plan, the structure is incrementally build. Each time the structure is extended, a search for cost-optimal sequential plans is made. The process stops either if it can be proved that searching for longer plans will only produce plans of worse quality, or if the given bound is reached and no plan has been found.

The search procedure is not complete since if no solution is found it cannot prove that there is no solution. If a solution is found, in most cases the procedure provides a cost-optimal plan and the proof of its optimality. In some cases a solution is found with no assurance of its optimality (this problem is due to the fixed bound and can easily be corrected).

Overview

CFDP is based on the FDP planning system (Grandcolas & Pain-Barre 2007), which competes at IPC-5. FDP searches optimal sequential plans (that is plans for which the number of actions is minimal). It implements an Iterative Deepening Depth First Search procedure, and uses structures similar to Graphplan planning graphs. Its main characteristics are:

- a powerfull mecanism to maintain consistency and propagate removals, that ressembles CSP arc-consistency procedures,
- the detection of redundant sequences of actions so as these sequences are considered only once; the idea is to force consecutive actions which are *independent* to respect a given ordering,
- an evaluation of the reachability of the goals which permits early dead ends detection,
- the memorization of invalid states together with their distances to the goals, so as to avoid redundant searches.

FDP performs successive searches for sequential plans, increasing their lengths step by step. Then the first plan which is discovered is optimal in the number of actions. Searching for a plan of length k consists in a depth first search, starting from the initial state. FDP supports different decompositions like splitting the actions at a given step. However, the search process is basically the enumeration of the consistent action sequences that can be applied in the initial state.

We encourage interested readers to refer to the paper (Grandcolas & Pain-Barre 2007) for details on the FDP system.

This approach is not well suited for the search of cost optimal plans, excepted in the particular case where all actions have the same cost. The main search strategy has been modified to satisfy this new objective.

CFDP search process

CFDP main process consists in searching sequential plans of increasing lengths until no better plan can be found. The problem is to find a criterion that ensures that, beyond a given length, plans have increasing costs. The approach that we propose is based on the evaluation of minimal bounds for the costs of plans terminations.

The system operates in three phases:

phase 1 a sequential plan is searched with the standard FDP search procedure; unsuccessfull states are memorized during the search.

phase 2 the sequences of length less than a given bound, which terminate valid plans are enumerated ; the minimal costs of these sequences are memorized for each length.

phase 3 valid plans of increasing lengths are searched using FDP ; this phase stops when dead ends are all caused by the violation of the *cost constraint*.

Notes phase 1.

The objective is to verify that there exists at least one solution (within the given lentgth bound). Furthermore a lower bound and an upper bound on respectively the length and the cost of an optimal solution are provided. For this purpose we use FDP standard search: it consists in an Iterative Deepening scheme with a Depth First Search procedure. The first solution which is discovered is optimal in the number of actions. Then its length is minimal and its cost is an upper bound for the cost of a cost-optimal solution.

Many techniques are used in FDP in order to speed up the search. The memorization of the invalid states is an important one: during the search, each time the procedure demonstrates that it is not possible to reach the goals from the current state S in k steps, the couple (S, k) is memorized in a hash table. Afterwards, each time the situation (S, k) is rediscovered, the search is aborted.

Notes phase 2.

In this phase goals only are instanciated, the initial state is completely undefined. An iterative deepening depth first search is used to enumerate all the sequences of lengths less than a given bound l_{max} that achieve the goals. Minimal costs for each length are memorized during the search. Since the initial state is undefined it is more efficient to perform a backward search. Minimal costs help to prune the search in phase 3.

Notes phase 3.

In this phase the FDP search procedure is used in a special way: in the FDP structure the initial state only is instanciated, the final state is left undefined. The FDP structure is extended step by step until it can be proved that no longer plan of cost less than the current minimal cost exists (or until a fixed bound is reached, but if a plan has been found in phase 1 then this bound should be ∞). Since the goals are not instanciated, dead ends are caused either by the violation of the *cost constraint*, or by the detection of goals unreachability. Failures caused by the propagation of removals should not occur since we use a forward search and the final state is uninstantiated.

The **Cost constraint.** The *cost constraint* is a dynamic constraint involving the current minimal cost C_{min} (that is the minimal cost of the plans which have been encountered in phase 1 or since the beginning of the search in phase 3): each time a better plan is discovered the constraint is updated. This constraint helps the search procedure to discard partial plans which cannot be extended to plans with costs less than C_{min}. For a given partial plan P it evaluates the minimal cost of the remaining steps to the final state (calculated in phase 2). This evaluation, added to the cost of the sequence P, gives a minimal bound for the cost of a plan beginning with P. If this bound is not less than C_{min} then the search with P is aborted.

Termination. When searching for plans of length k, if all dead ends are caused by the violation of the cost constraint, then searching for plans of length more than k will do so. This is due to the monotonicity of the evaluation of sequences costs. Given a sequence P, if the evaluation procedure returns a minimal cost c for any k steps sequence which begins with P, then it will return a minimal cost greater or equal to c for sequences beginning with P of length greater than k.

Finally, remark that a good minimal bound at the very beginning of the search can improve drastically the process. Besides, the phase 3 search is not very efficient since the final state is completely undefined, especially at the beginning of the process when there are no solution plans. Then in most cases it is preferable to use the length and the cost of the solution produced by the phase 1 as a basis for the phase 3 search.

Conclusion

We have presented a planner based on FDP search procedure. Given a fixed upper bound on the number of actions, the system search cost-optimal plans.

The main scheme is a 3 phases process: phase 1 and phase 2 produce information to speed up the forward search procedure used in phase 3.

We have not yet compared our approach with other existing systems.

References

Grandcolas, S., and Pain-Barre, C. 2007. Filtering, decomposition and search space reduction for optimal sequential planning. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07)*, 993–998.

Korf, R. 1985. Macro-operators: A weak method for learning. *Artificial Intelligence* 26(1):35–77.