A Planning-Graph Heuristic for Forward-Chaining Adversarial Planning

Pascal Bercher and Robert Mattmüller

Introduction

Given

- Fully observable discrete planning problem
- Two agents, protagonist and antagonist
- Reachability goal for protagonist

Desired

- Winning strategy for the protagonist (= strong plan, not necessarily conformant)

Technically

- STRIPS-style state and action encoding
- Players taking turns

First Step

Reduction to evaluation of AND/OR graph over physical states.

Algorithmic Alternatives

- Symbolic regression search (cf. MBP [CPRT03])
- Heuristically guided explicit-state progression search [BG01]

Approach used here

Variant of AO* algorithm [Nil80].

How to initialize leaf node cost estimates?

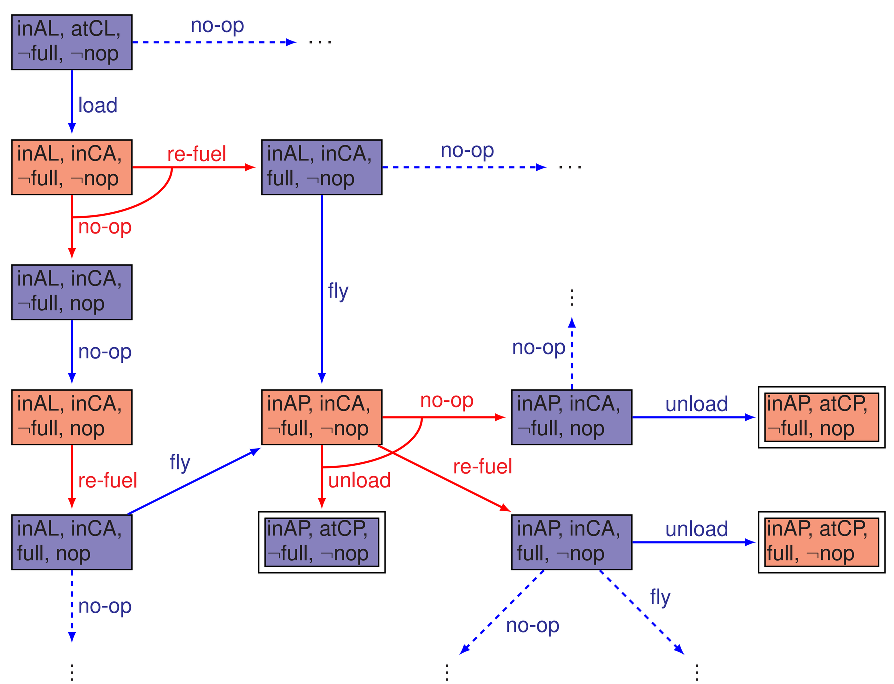
Variant of FF heuristic [HN01].

Example: Airplane Domain

Problem

- Logistics-like problem: transport a package from London to Paris by plane.
- Protagonist and antagonist have different capabilities:
  - The protagonist can:
    - load (package into plane)
    - fly (from city to city if tank is full, emptying it)
    - unload (package from plane)
    - no-op (do nothing)
  - The antagonist can:
    - unload (package from plane)
    - re-fuel (if tank is empty)
    - no-op (do nothing; fairness condition: not only no-ops)
- Antagonist wants to sabotage task, e.g.,
  - by unloading packages at the wrong places or
  - by refusing to re-fuel when necessary.

AND/OR Graph and Solution



AO* Search

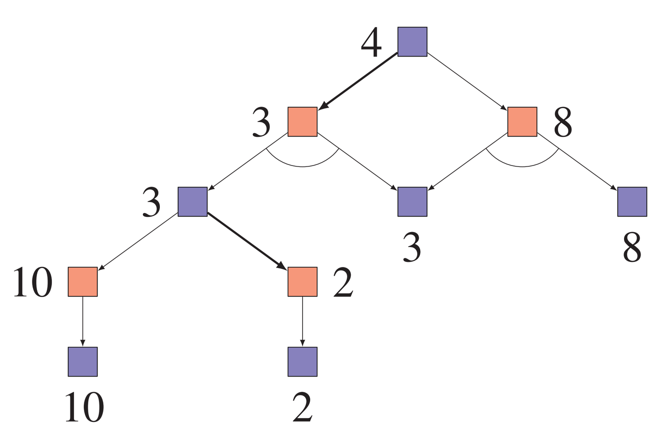
Algorithm [Nil80]

- Start with initial state.
- While not finished do:
  - Extract most promising subgraph by tracing marked connectors from initial state.
  - Choose unexpanded leaf node in subgraph and expand it.
  - Initialize cost estimates of new nodes.
  - Propagate cost estimates and winner information upward and update marked connectors.

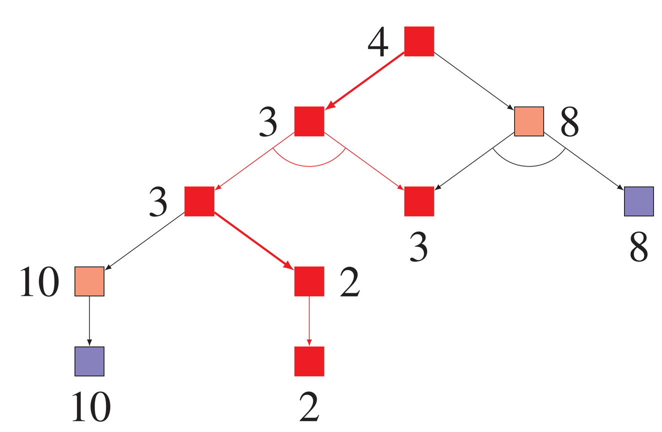
Details

- Graph search with duplicate elimination
- Approximative updates of cost estimates

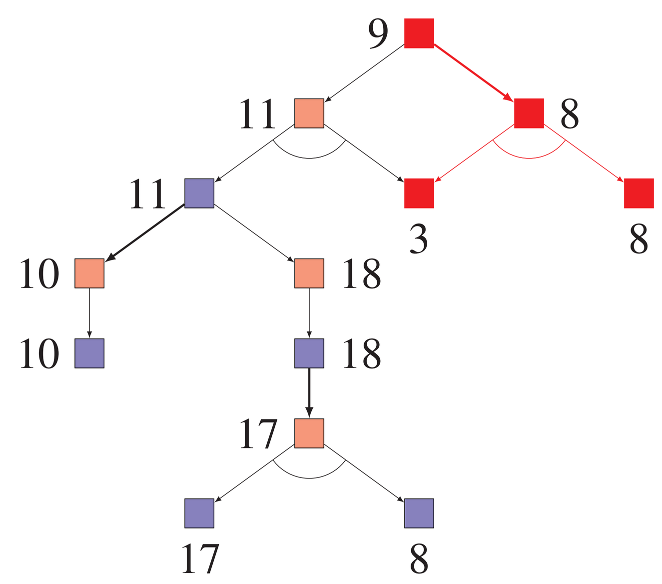
Example: (1) Before Expansion



Example: (2) Most Promising Subgraph



Example: (3) After Expansion and Update



Heuristic

Motivation

- Why: We need to initialize the cost estimates at new leaf nodes in an informed way.
- Constraint: Heuristic should be domain-independent.
- How: Adapt heuristics from classical planning to the adversarial setting. Here: FF heuristic.

FF Heuristic [HN01]

- Generate relaxed problem (no delete lists).
- Build relaxed planning graph.
- Extract (non-optimal) relaxed plan and return its length.

Adaption

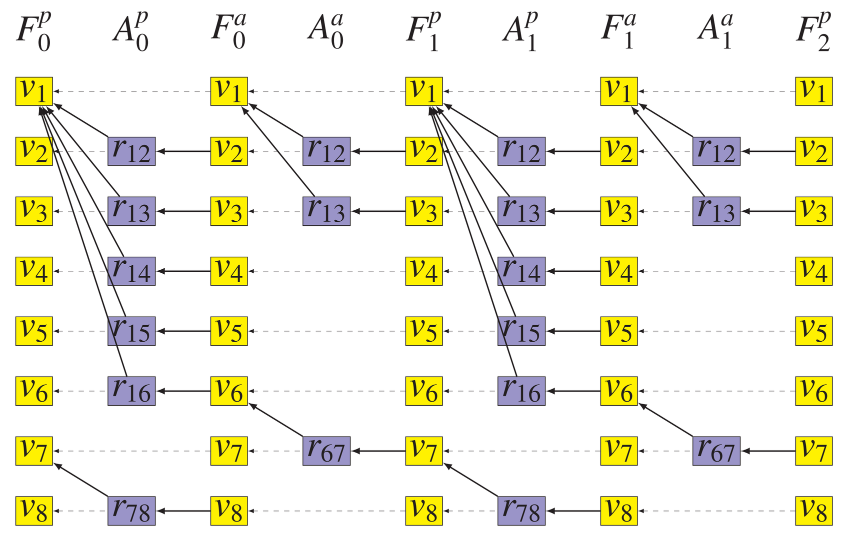
- Distinct proposition and action layers for protagonist and antagonist.
- Distinguish between relaxed actions controlled by protagonist and antagonist.

Example

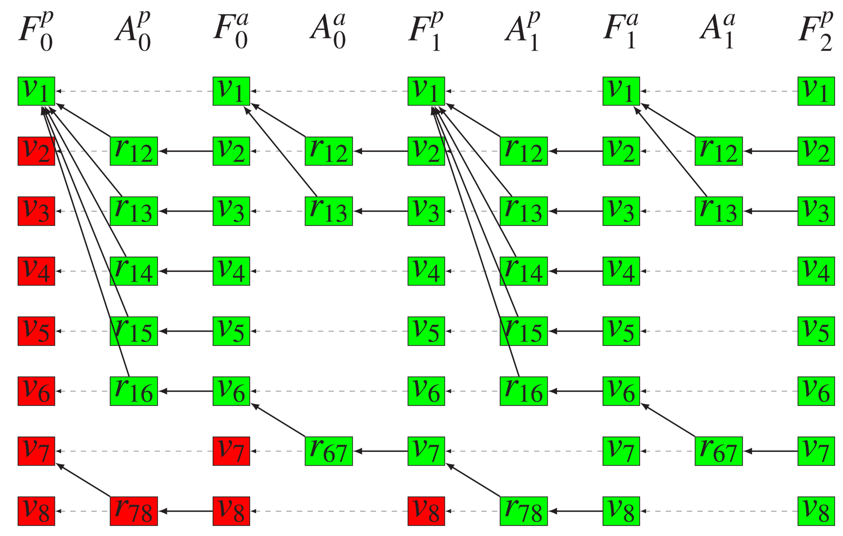
- Variables: $v_1, \ldots, v_8$
- Rules in relaxed problem: $\{r_{12}, r_{13}, r_{14}, r_{15}, r_{16}, r_{67}, r_{78}\}$ where $r_{ij} = \langle v_i \rightarrow v_j \rangle$
- Rules controlled by protagonist: $\{r_{12}, r_{13}, r_{14}, r_{15}, r_{16}, r_{78}\}$
- Rules controlled by antagonist: $\{r_{12}, r_{13}, r_{67}\}$
- Current state: $\{v_1\}$
- Goal: $\{v_1, \ldots, v_8\}$

Heuristic (ctd.)

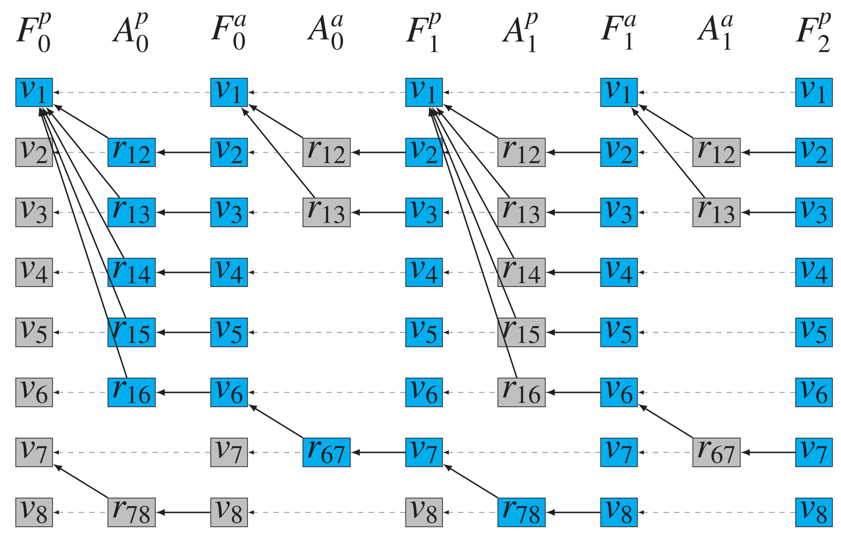
Construction of Relaxed Planning Graph



Applicable Relaxed Operators



Extraction of Relaxed Plan



Restructuring the Relaxed Plan

- Selected rules for protagonist: $\{r_{12}, r_{13}, r_{14}, r_{15}, r_{16}, r_{78}\}$
- Selected rule(s) for antagonist: $\{r_{67}\}$
- Rules, that only belong to the protagonist: $\{r_{14}, r_{15}, r_{16}, r_{78}\}$ (4 rules)
- Rules, that only belong to the antagonist: $\{r_{67}\}$ (1 rule)
- Rules, that belong to both players: $\{r_{12}, r_{13}\}$ (2 rules)

Optimistic Restructuring

- Antagonist helps as much as he can.
- Possible plan: $\langle r_{16}, r_{67}, r_{78}, r_{12}, r_{14}, r_{13}, r_{15} \rangle$
- Heuristic: 7.

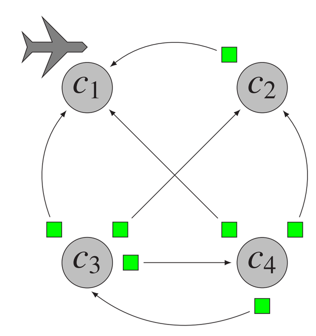
Pessimistic Restructuring

- Antagonist plays as few rules as possible.
- Possible plan: $\langle r_{16}, r_{67}, r_{78}, nop, r_{12}, nop, r_{13}, nop, r_{14}, nop, r_{15} \rangle$
- Heuristic: 11.

Experiments and Results

Problems: Airplane Domain

- Instances of the airplane domain with different delivery graphs.
- Between three and four cities, between one and ten packages.
- Example: four cities, seven packages (see Fig. to the right).



Results

	BFS		AO* + h_{FF}		AO* + $h_{opt. adv. FF}$		AO* + $h_{pes. adv. FF}$		MBP		
ℓ/p	time	nodes	time	nodes	time	nodes	time	nodes	pre	search	BDD
2/1	0.014	44	0.046	37	0.023	37	0.022	37	0.064	0.004	14822
2/2	0.048	152	0.064	96	0.088	96	0.077	84	0.384	0.084	290495
3/3	0.354	2106	0.311	1131	0.571	1106	0.226	285	4.128	3.668	166012
3/4	0.870	8211	0.696	2766	0.781	2499	0.538	1053	39.890	82.073	654147
3/5	5.556	43785	2.599	12676	3.019	11644	0.672	1836	–	–	–
3/6	87.691	237264	12.421	61154	11.896	54469	2.526	10333	–	–	–
4/6	–	–	203.678	408768	37.762	129362	3.973	14115	–	–	–
4/7	–	–	756.138	1006666	131.505	341093	1.375	4262	–	–	–
4/8	–	–	–	–	–	–	29.129	100263	–	–	–
4/9	–	–	–	–	–	–	129.305	361899	–	–	–
4/10	–	–	–	–	–	–	–	–	–	–	–

ℓ : #locations, p : #packages, BDD: #BDD nodes, red: worst, blue: best

Conclusion

- Domain-independent heuristics promising approach to conditional/adversarial planning.
- Explicit-state progression competitive with symbolic regression.
- Potential application in General Game Playing [GLP05].

Future Work

- Assessment of other domain-independent heuristics in conditional setting.
- More Benchmarks: Conditional planning problems from IPC non-deterministic track.

References

- B. Bonet and H. Geffner. Planning as Heuristic Search. *Artif. Intell.*, 129(1-2):5–33, 2001.
- A. Cimatti, M. Pistore, M. Roveri, and P. Traverso. Weak, Strong, and Strong Cyclic Planning via Symbolic Model Checking. *Artif. Intell.*, 147(1–2):35–84, 2003.
- M. R. Genesereth, N. Love, and B. Pell. General Game Playing: Overview of the AAAI Competition. *AI Magazine*, 26(2):62–72, 2005.
- J. Hoffmann and B. Nebel. The FF Planning System: Fast Plan Generation Through Heuristic Search. *JAIR*, 14:253–302, 2001.
- N. J. Nilsson. Principles of Artificial Intelligence. Springer-Verlag, 1980.

Research Group on the Foundations of Artificial Intelligence, Department of Computer Science, University of Freiburg, Germany Mail: {bercherp,mattmuel}@informatik.uni-freiburg.de