Combining Heuristics and Transition Classifiers in Classical Planning

Farid Musayev[1] Dominik Drexler[1] Daniel Gnad[1, 2] Jendrik Seipp[1]

[1]Linköping University [2]Heidelberg University

Summary

Summary

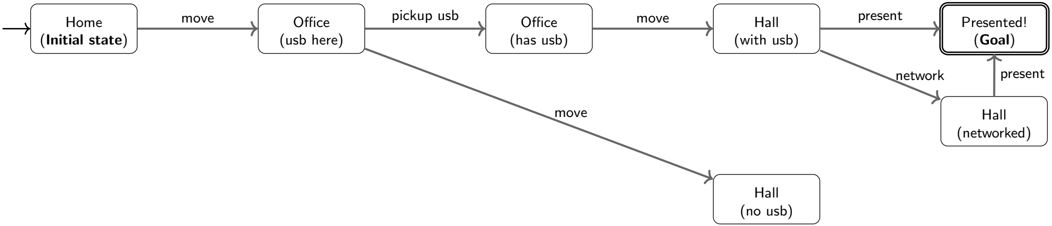
- **Learning for Classical Planning:**
  - Learning-based planners often rely solely on learned heuristics or policies.
  - To date, no purely learning-based method consistently outperforms state-of-the-art heuristic planners.

Summary

- **Learning for Classical Planning:**
  - ▷ Learning-based planners often rely solely on learned heuristics or policies.
  - ▷ To date, no purely learning-based method consistently outperforms state-of-the-art heuristic planners.
- **Contribution**:
  - ▷ Combine strengths of both paradigms, learning and planning.
  - ▷ A simple approach to learn transition classifiers, a domain general knowledge.
  - ▷ A detailed study of how to combine learned classifiers with heuristic search methods.

Classical Planning

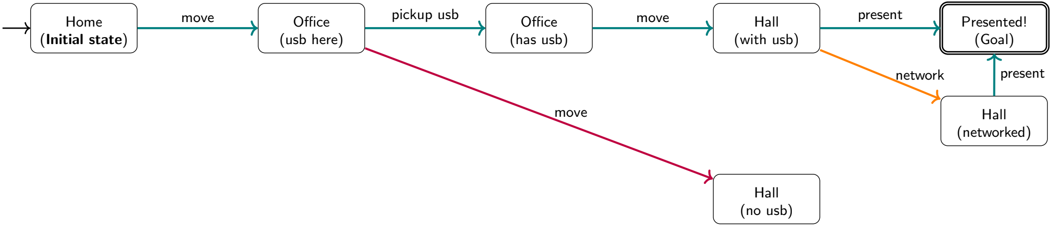
Classical Planning — Example (Perish or Present[1])



[1] Inspired by "Chaotic Researcher" domain from Gnad et al.

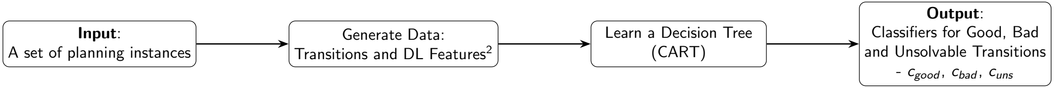
What Do We Learn?

Learning Transition Classifiers



Good transition decreases the distance to the goal state.
Bad transition increases or does not change the distance to the goal state.
Unsolvable transition leads to a state from which the goal state can never be reached.

Learning Pipeline



[2] Francès, Bonet, and Geffner, "Learning General Planning Policies from Small Examples Without Supervision".

How Do We Plan?

Combining Heuristics and Classifiers

We combine h^{FF}(a state-of-the-art non-learned heuristic[3]) with c_{good}, c_{bad}, and c_{uns}:

[3] Hoffmann and Nebel, “The FF Planning System: Fast Plan Generation Through Heuristic Search”.

[4] Yoon, Fern, and Givan, “Learning Control Knowledge for Forward Search Planning”.

Combining Heuristics and Classifiers

We combine h^{FF}(a state-of-the-art non-learned heuristic[3]) with c_{good}, c_{bad}, and c_{uns}:

- **Heuristic-first:**
  - ▷ Use $h^{\mathrm{FF}}$to guide Greedy Best-First Search (GBFS).
  - ▷ Break ties using c_{good}, c_{bad}, and c_{uns}.

[3] Hoffmann and Nebel, "The FF Planning System: Fast Plan Generation Through Heuristic Search".

[4] Yoon, Fern, and Givan, "Learning Control Knowledge for Forward Search Planning".

Combining Heuristics and Classifiers

We combine h^{FF}(a state-of-the-art non-learned heuristic[3]) with c_{good}, c_{bad}, and c_{uns}:

- **Heuristic-first:**
  - Use $h^{\mathrm{FF}}$to guide Greedy Best-First Search (GBFS).
  - Break ties using c_{good}, c_{bad}, and c_{uns}.
- **Alternating** between heuristics and classifiers:
  - Store states in two open lists: one sorted by $h^{\mathrm{FF}}$and another by c_{good}.
  - Expand states in a dual-queue approach.

[3] Hoffmann and Nebel, "The FF Planning System: Fast Plan Generation Through Heuristic Search".

[4] Yoon, Fern, and Givan, "Learning Control Knowledge for Forward Search Planning".

Combining Heuristics and Classifiers

We combine h^{FF}(a state-of-the-art non-learned heuristic[3]) with c_{good}, c_{bad}, and c_{uns}:

- **Heuristic-first:**
  - Use $h^{\mathrm{FF}}$to guide Greedy Best-First Search (GBFS).
  - Break ties using c_{good}, c_{bad}, and c_{uns}.
- **Alternating** between heuristics and classifiers:
  - Store states in two open lists: one sorted by $h^{\mathrm{FF}}$and another by c_{good}.
  - Expand states in a dual-queue approach.
- **Classifier-first:**
  - Use c_{good} to guide lookahead[4] search.
  - Store states expanded along trajectory in the open list sorted by h^{FF}.
  - If no c_{good}-compatible transition is found, pick the next state from the open list.

[3] Hoffmann and Nebel, "The FF Planning System: Fast Plan Generation Through Heuristic Search".

[4] Yoon, Fern, and Givan, "Learning Control Knowledge for Forward Search Planning".

Combining Heuristics and Classifiers

We combine h^{FF}(a state-of-the-art non-learned heuristic[3]) with c_{good}, c_{bad}, and c_{uns}:

- **Heuristic-first:**
  - Use $h^{\mathrm{FF}}$to guide Greedy Best-First Search (GBFS).
  - Break ties using c_{good}, c_{bad}, and c_{uns}.
- **Alternating** between heuristics and classifiers:
  - Store states in two open lists: one sorted by $h^{\mathrm{FF}}$and another by c_{good}.
  - Expand states in a dual-queue approach.
- **Classifier-first:**
  - Use c_{good} to guide lookahead[4] search.
  - Store states expanded along trajectory in the open list sorted by h^{FF}.
  - If no c_{good}-compatible transition is found, pick the next state from the open list.

Enhance all combinations with **unsolvability** detection:

- Compute c_{uns}-compatible transitions and sort resulting successor states in the open list.

---

[3] Hoffmann and Nebel, "The FF Planning System: Fast Plan Generation Through Heuristic Search".

[4] Yoon, Fern, and Givan, "Learning Control Knowledge for Forward Search Planning".

What is an Effective Combination?

What is an Effective Combination?

Domain	Heuristic-first	+ uns	Alternating	+ uns	Classifier-first	+ uns
Blocksworld (90)	30	30	28	28	**39**	**39**
Childsnack (90)	25	26	20	23	33	**34**
Ferry (90)	67	67	65	65	68	**69**
Floortile (90)	12	31	22	32	17	**33**
Miconic (90)	88	89	89	88	**90**	**90**
Rovers (90)	31	31	38	38	**47**	**47**
Satellite (90)	**67**	66	66	65	**67**	**67**
Sokoban (90)	**36**	34	34	34	**36**	35
Spanner (90)	30	30	30	30	**56**	49
Transport (90)	45	44	52	53	**57**	**57**
Sum (900)	431	448	444	456	510	**520**

Number of solved tasks per domain and algorithm on the IPC 2023 Learning Track instances.

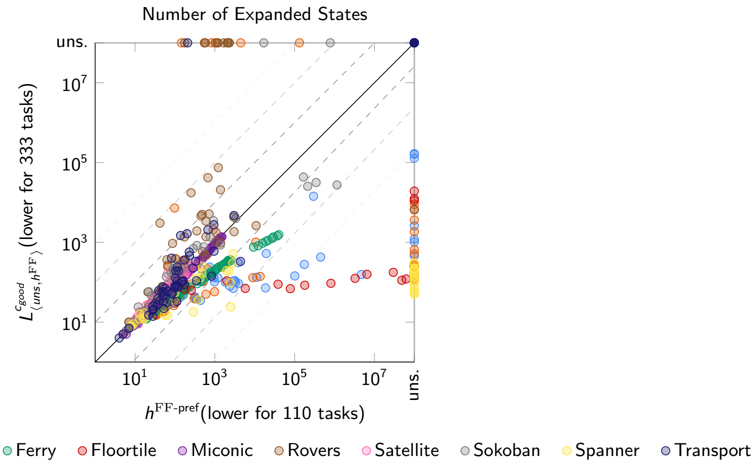
Are Transition Classifiers Helpful?

Are Transition Classifiers Helpful?

Domain	h^{FF}	$h^{\mathrm{FF\text{-}pref}}$	$L^{c_{good}}_{\langle uns, h^{\mathrm{FF}} \rangle}$
Blocksworld (90)	28	31	**39**
Childsnack (90)	22	33	**34**
Ferry (90)	**69**	**69**	**69**
Floortile (90)	11	15	**33**
Miconic (90)	**90**	**90**	**90**
Rovers (90)	32	**57**	47
Satellite (90)	64	**67**	**67**
Sokoban (90)	36	**37**	35
Spanner (90)	30	30	**49**
Transport (90)	38	**58**	57
Sum (900)	420	487	**520**

Number of solved tasks per domain and algorithm on the IPC 2023 Learning Track instances.

Are Transition Classifiers Helpful?



Comparison to State of the Art

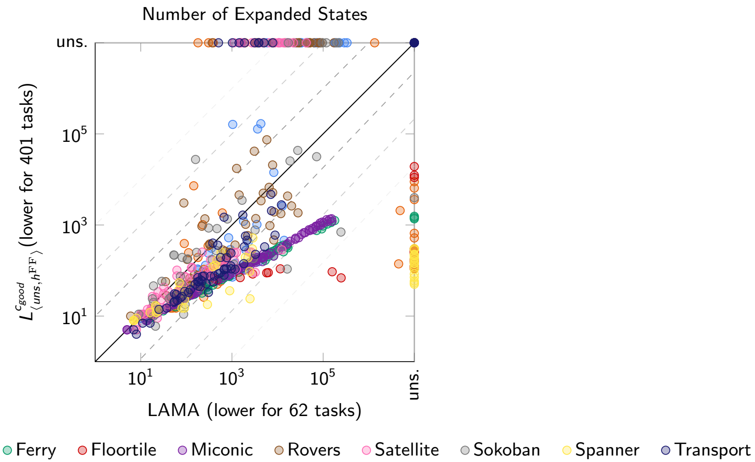
Comparison to State of the Art

Domain	LAMA[5]	$L^{c_{good}}_{\langle uns, h^{\mathrm{FF}} \rangle}$
Blocksworld (90)	**60**	39
Childsnack (90)	**35**	34
Ferry (90)	66	**69**
Floortile (90)	11	**33**
Miconic (90)	**90**	**90**
Rovers (90)	**68**	47
Satellite (90)	**89**	67
Sokoban (90)	**40**	35
Spanner (90)	30	**49**
Transport (90)	**66**	57
Sum (900)	**555**	520

Number of solved tasks per domain and algorithm on the IPC 2023 Learning Track instances.

[5] Richter and Westphal, "The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks".

Comparison to State of the Art



Conclusions

Conclusions

- A single decision tree that simultaneously captures three classes of transitions.
- A detailed study of how to combine heuristic search methods with the learned classifiers.
- Classifier-based lookahead search using h^{FF} and unsolvability is an effective combination.
- The strengths of combining techniques from learning and planning.

References

References I

- Francès, Guillem, Blai Bonet, and Hector Geffner. "Learning General Planning Policies from Small Examples Without Supervision". In: *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021)*. Ed. by Kevin Leyton-Brown and Mausam. AAAI Press, 2021, pp. 11801–11808.
- Hoffmann, Jörg and Bernhard Nebel. "The FF Planning System: Fast Plan Generation Through Heuristic Search". In: *Journal of Artificial Intelligence Research* 14 (2001), pp. 253–302.
- Richter, Silvia and Matthias Westphal. "The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks". In: *Journal of Artificial Intelligence Research* 39 (2010), pp. 127–177.
- Yoon, Sungwook, Alan Fern, and Robert Givan. "Learning Control Knowledge for Forward Search Planning". In: *Journal of Machine Learning Research* 9 (2008), pp. 683–718.