On Bidirectional Heuristic Search in Classical Planning: An Analysis of BAE*

Kilian Hu[1] David Speck[2]

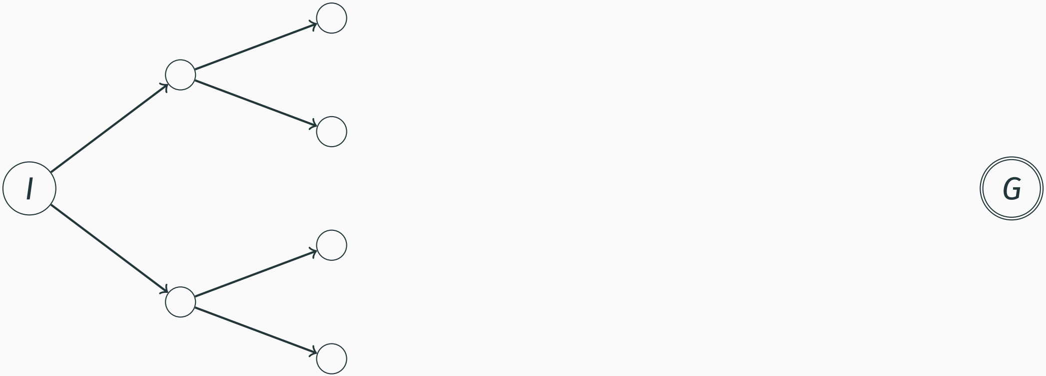
[1]University of Freiburg

[2]Linköping University

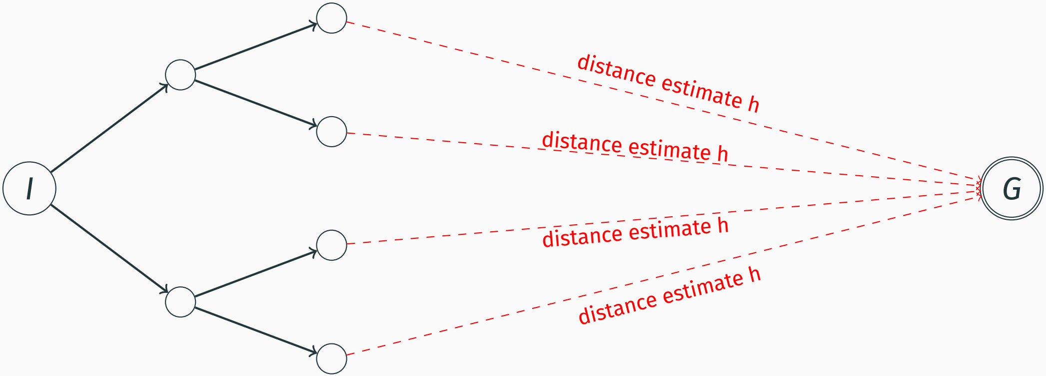
LINKÖPING UNIVERSITY

Motivation

Motivation



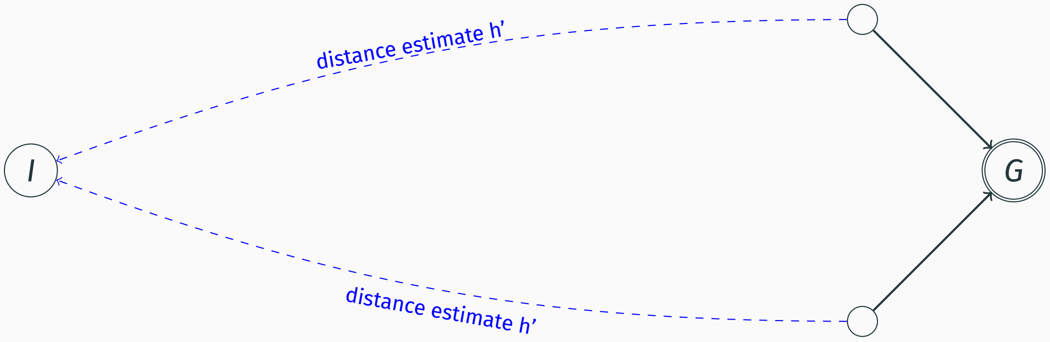
Motivation



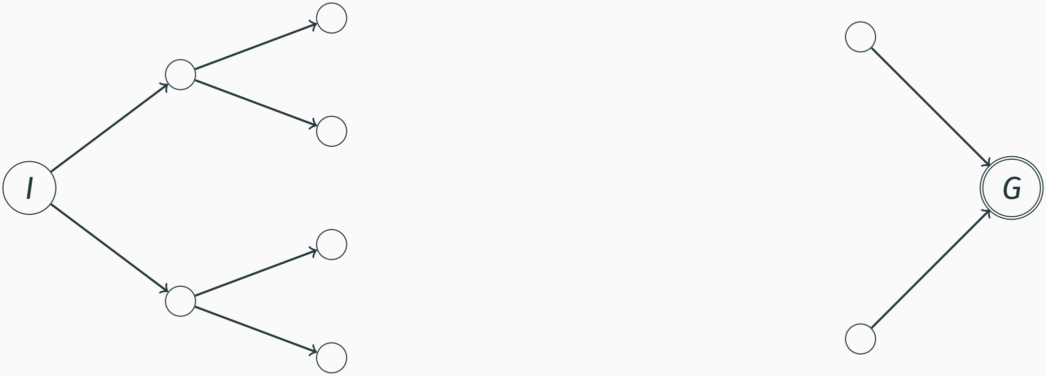
I
G
distance estimate h
distance estimate h
distance estimate h
distance estimate h

Motivation

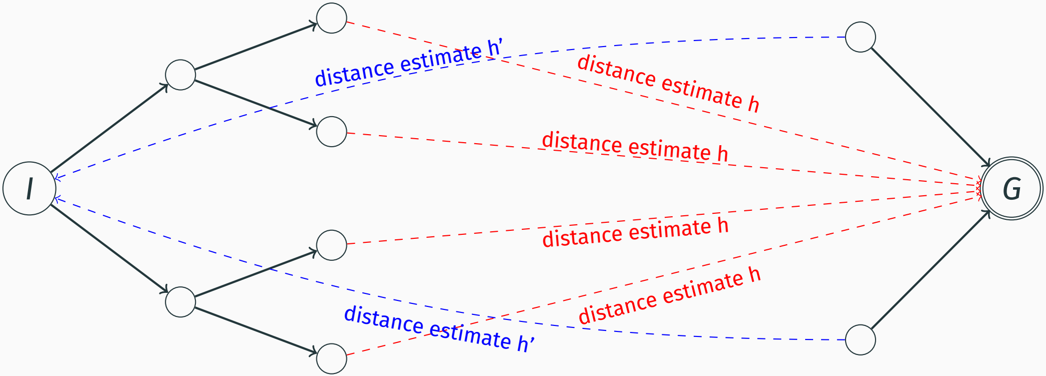
Motivation



Motivation

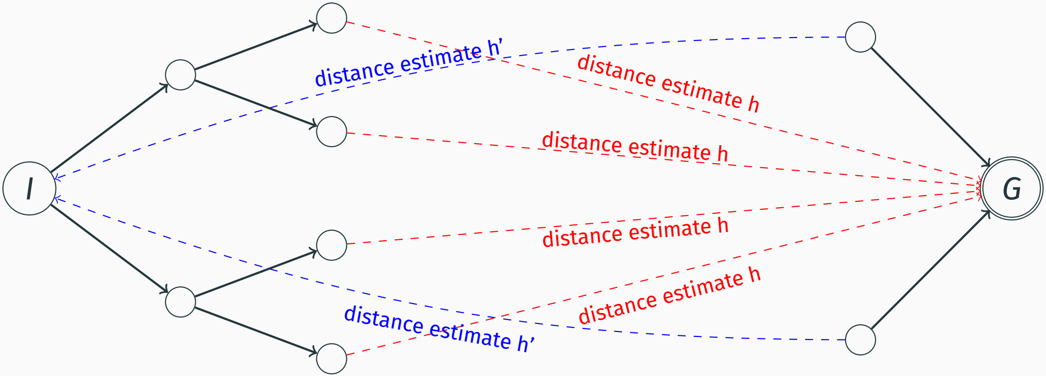


Motivation



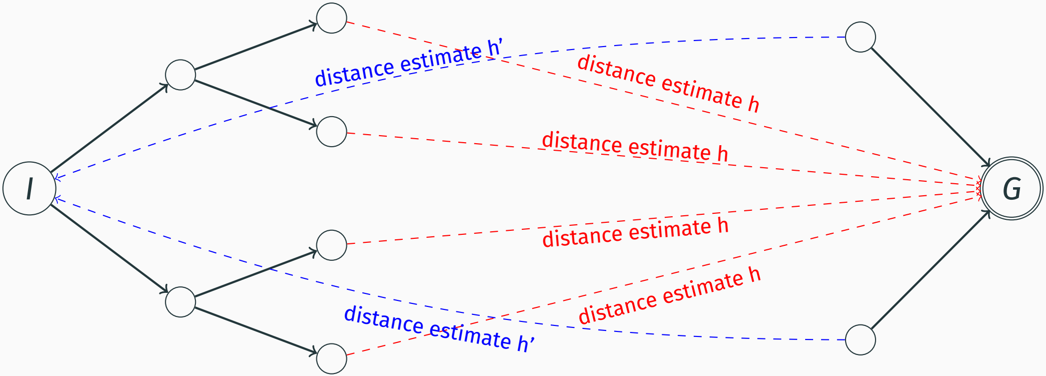
I
G
distance estimate h'
distance estimate h
distance estimate h
distance estimate h
distance estimate h
distance estimate h'

Motivation



- Recent years: new strong bidirectional search algorithms

Motivation



- Recent years: new strong bidirectional search algorithms

⇝ Can we transfer these works to classical planning?

Heuristic Search

- Most prominent algorithm: A* (backward A^*_b)
  - Explores the state space unidirectionally
- Bidirectional heuristic search performs two searches simultaneously
  - Can outperform unidirectional search
  - Generally considered inefficient [BK15]

Heuristic Search

- Most prominent algorithm: A* (backward A_b^*)
  - Explores the state space unidirectionally
- Bidirectional heuristic search performs two searches simultaneously
  - Can outperform unidirectional search
  - Generally considered inefficient [BK15]

Rekindled interest in bidirectional heuristic search

- Improved theoretical understanding [ECS+17]
- Development of new successful algorithms
  - NBS [ECS+17]
  - BAE* [Sad13, ARB20]

Heuristic Search

- Most prominent algorithm: A* (backward A_b^*)
  - Explores the state space unidirectionally
- Bidirectional heuristic search performs two searches simultaneously
  - Can outperform unidirectional search
  - Generally considered inefficient [BK15]

Rekindled interest in bidirectional heuristic search

- Improved theoretical understanding [ECS+17]
- Development of new successful algorithms
  - NBS [ECS+17]
  - BAE* [Sad13, ARB20]
    - ⇝ Excellent performance (compared to more complex algorithms)
    - ⇝ Promising candidate for planning

BAE* in a Nutshell

- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics

BAE* in a Nutshell

- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
  - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$

BAE* in a Nutshell

- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
    - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$

- $g_F(n) = 3$

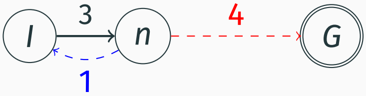
BAE* in a Nutshell

- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
  - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$

- $g_F(n) = 3$
- $h_F(n) = 4$
- $f_F(n) = 7$

BAE* in a Nutshell

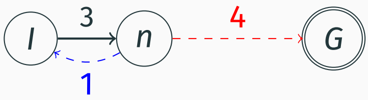
- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
  - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$



- $g_F(n) = 3$
- $h_F(n) = 4$
- $f_F(n) = 7$
- $h_B(n) = 1$

BAE* in a Nutshell

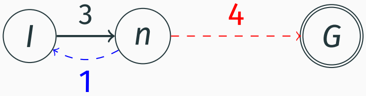
- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
  - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$



- $g_F(n) = 3$
- $h_F(n) = 4$
- $f_F(n) = 7$
- $h_B(n) = 1$
- $d_F(n) = 3 - 1 = 2$

BAE* in a Nutshell

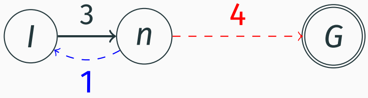
- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
  - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$



- $g_F(n) = 3$
- $h_F(n) = 4$
- $f_F(n) = 7$
- $h_B(n) = 1$
- $d_F(n) = 3 - 1 = 2$
- $b_F(x) = 7 + 2 = 9$

BAE* in a Nutshell

- Bidirectional A* with Error [Sad13, ARB20]
- Performs two A*-like searches
- Exploits inaccuracy of consistent heuristics
- Priority function: $b_x(n) = f_x(n) + d_x(n)$
  - Diff-value: $d_x(n) = g_x(n) - h_{\overline{x}}(n)$
- Termination: lower bound $\geq$ upper bound
  - Lower: $\frac{bMin_f + bMin_b}{2}$
  - Upper: cost of best solution found so far



- $g_F(n) = 3$
- $h_F(n) = 4$
- $f_F(n) = 7$
- $h_B(n) = 1$
- $d_F(n) = 3 - 1 = 2$
- $b_F(x) = 7 + 2 = 9$

Bidirectional Search in Planning

Algorithms like BAE* come with some challenges

- Forward search is easy
- Backward search is tricky
  - Multiple goal states (sometimes exponentially many)
  - Regressing a state with an action $\rightsquigarrow$ multiple predecessors

Bidirectional Search in Planning

Algorithms like BAE* come with some challenges

- Forward search is easy
- Backward search is tricky
  - Multiple goal states (sometimes exponentially many)
  - Regressing a state with an action $\rightsquigarrow$ multiple predecessors

Backward search in Planning

- Generation of a reversed task [Sud13, GB13, ABFF13]
- Reversed SAS+ tasks often use partial states
  - Adjustment of the heuristic computations
  - Non-trivial intersection check of search frontiers

Bidirectional Search in Planning

Algorithms like BAE* come with some challenges

- Forward search is easy
- Backward search is tricky
  - Multiple goal states (sometimes exponentially many)
  - Regressing a state with an action ⇝ multiple predecessors

Backward search in Planning

- Generation of a reversed task [Sud13, GB13, ABFF13]
- Reversed SAS+ tasks often use partial states
  - Adjustment of the heuristic computations
  - Non-trivial intersection check of search frontiers

⇝ Here: Reversed task with an explicit state representation

Reversed Task Generation

Initial states

- Goal states of the actual task
- Add initial states directly to open list
  - Alternative: single dummy initial state
  - Can negatively affect heuristics

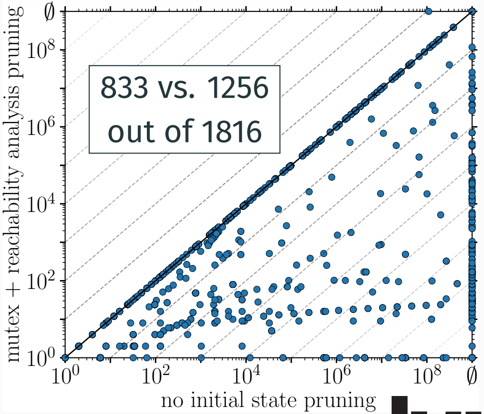
LINKÖPING UNIVERSITY

Reversed Task Generation

Initial states

- Goal states of the actual task
- Add initial states directly to open list
    - Alternative: single dummy initial state
    - Can negatively affect heuristics
- Reducing the number of initial states

Number of initial states

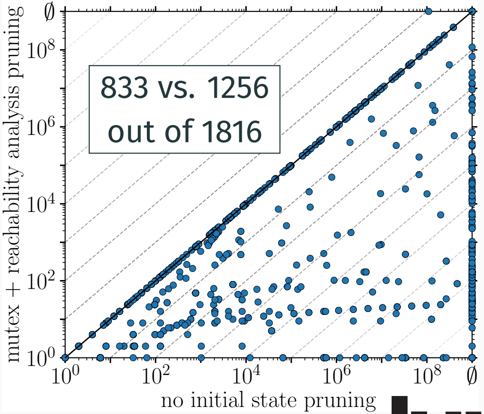


Reversed Task Generation

Initial states

- Goal states of the actual task
- Add initial states directly to open list
  - Alternative: single dummy initial state
  - Can negatively affect heuristics
- Reducing the number of initial states

Number of initial states



Reversed Operators

- Multiple reversed actions for an action
- In practice often unproblematic
  - 1.44 reversed actions per action

Empirical Evaluation

- Implemented A_b^* and BAE* in Fast Downward [Hel06]
  - Backward search based on the reversed task
- 3 different consistent heuristics
  - Max heuristic $h^{\max}$ [BG01]
  - Incremental pattern database heuristic h^{iPDB} [HBH+07]
  - Diverse potentials heuristic h^{pot} [SPH15]
- Planning tasks from optimal track of IPCs
- 30 min and 4 GB memory for each task

Coverage

	h^{max}			h^{iPDB}			h^{pot}		
	A*	A^*_b	BAE*	A*	A^*_b	BAE*	A*	A^*_b	BAE*
Sum (1816)	**847**	558	706	**1009**	732	805	**989**	732	841

LINKÖPING UNIVERSITY

Coverage

	h^{max}			h^{iPDB}			h^{pot}		
	A*	A_b^*	BAE*	A*	A_b^*	BAE*	A*	A_b^*	BAE*
blocks (35)	21	22	**30**	28	**32**	31	28	28	**30**
ged (20)	15	15	**20**	19	13	**20**	19	13	**20**
termes (20)	10	8	**15**	13	12	**16**	12	11	**16**
	...			...			...		
Sum (1816)	**847**	558	706	**1009**	732	805	**989**	732	841

Coverage – Possible to Generate Reversed Task

	$h^{\max}$			h^{iPDB}			h^{pot}		
	A*	A^*_b	BAE*	A*	A^*_b	BAE*	A*	A^*_b	BAE*
Sum (1256)	**719**	558	706	**853**	732	805	**848**	732	841

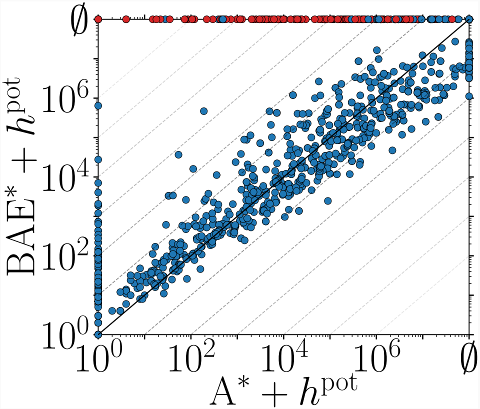
Coverage – Possible to Generate Reversed Task

	$h^{\max}$			h^{iPDB}			h^{pot}		
	A*	A^*_b	BAE*	A*	A^*_b	BAE*	A*	A^*_b	BAE*
Sum (1256)	**719**	558	706	**853**	732	805	**848**	732	841

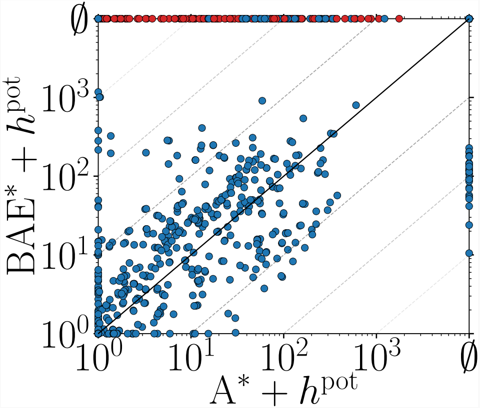
How do these algorithms perform in detail?

BAE* vs. A*

Node Expansions

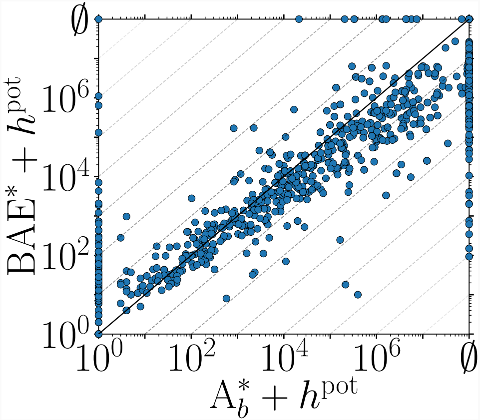


Runtime (seconds)

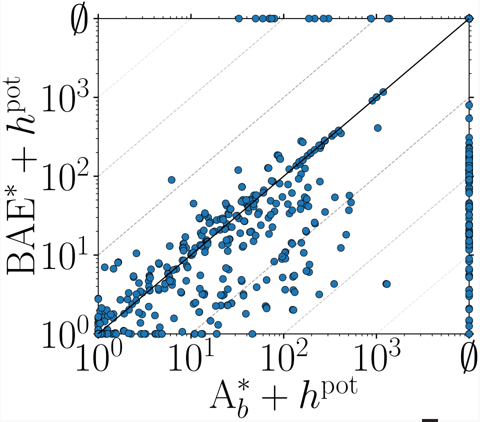


BAE* vs. Backward A*

Node Expansions



Runtime (seconds)



LINKÖPING UNIVERSITY

Complementary Strengths

	Single			Oracle		
	A^*	A^*_b	BAE^*	$\{A^*, A^*_b\}$	$\{A^*, BAE^*\}$	$\{A^*, A^*_b, BAE^*\}$
$h^{\max}$	847	558	706	848	**878**	**878**
h^{iPDB}	1009	732	805	1026	1022	**1031**
h^{pot}	989	732	841	1007	1019	**1030**

Complementary Strengths

	Single			Oracle			Simple Classifier $\mathcal{C}$
	A*	A^*_b	BAE*	{A*, A^*_b}	{A*, BAE*}	{A*, A^*_b, BAE*}	{A*, BAE*}
h^{max}	847	558	706	848	878	878	**856**
h^{iPDB}	**1009**	732	805	1026	1022	1031	986
h^{pot}	989	732	841	1007	1019	1030	**995**

$\mathcal{C}$: Use BAE* if the reversed task satisfies

- #initial states $\leq$ 100 and
- generation time $\leq$ 1sec

$\rightsquigarrow$ 20% of instances

Conclusion

- Analysis of bidirectional heuristic search for classical planning
    - In particular BAE*
- Reversed tasks with explicit state representation
    - Mutexes + reachability analysis pruning to reduce #initial states

Conclusion

- Analysis of bidirectional heuristic search for classical planning
    - In particular BAE*
- Reversed tasks with explicit state representation
    - Mutexes + reachability analysis pruning to reduce #initial states
- BAE* performs better than A* in multiple domains
- Analysis shows the merits of BAE*, A* and A_b^*

Conclusion

- Analysis of bidirectional heuristic search for classical planning
    - In particular BAE*
- Reversed tasks with explicit state representation
    - Mutexes + reachability analysis pruning to reduce #initial states
- BAE* performs better than A* in multiple domains
- Analysis shows the merits of BAE*, A* and A_b^*

Future Work

- More compact representations of a reversed task
    - Allow partial states in the backward search
- Other bidirectional heuristic search algorithms

- Vidal Alcázar, Daniel Borrajo, Susana Fernández, and Raquel Fuentetaja, *Revisiting regression in planning*, Proc. IJCAI 2013, 2013, pp. 2254–2260.
- Vidal Alcázar, Patricia J. Riddle, and Mike Barley, *A unifying view on individual bounds and heuristic inaccuracies in bidirectional search*, Proc. AAAI 2020, 2020, pp. 2327–2334.
- Blai Bonet and Héctor Geffner, *Planning as heuristic search*, AIJ **129** (2001), no. 1, 5–33.
- Joseph Kelly Barker and Richard E. Korf, *Limitations of front-to-end bidirectional heuristic search*, Proc. AAAI 2015, 2015, pp. 1086–1092.

- Jürgen Eckerle, Jingwei Chen, Nathan R. Sturtevant, Sandra Zilles, and Robert C. Holte, *Sufficient conditions for node expansion in bidirectional heuristic search*, Proc. ICAPS 2017, 2017, pp. 79–87.
- Héctor Geffner and Blai Bonet, *A concise introduction to models and methods for automated planning*, vol. 7, Synthesis Lectures on Artificial Intelligence and Machine Learning, no. 2, Morgan & Claypool, 2013.
- Patrik Haslum, Adi Botea, Malte Helmert, Blai Bonet, and Sven Koenig, *Domain-independent construction of pattern database heuristics for cost-optimal planning*, Proc. AAAI 2007, 2007, pp. 1007–1012.
- Malte Helmert, *The Fast Downward planning system*, JAIR **26** (2006), 191–246.

- Samir K. Sadhukhan, *Bidirectional heuristic search based on error estimate*, CSI Journal of Computing **2** (2013), no. 1–2, S1:57–S1:64.
- Jendrik Seipp, Florian Pommerening, and Malte Helmert, *New optimization functions for potential heuristics*, Proc. ICAPS 2015, 2015, pp. 193–201.
- Martin Suda, *Duality in STRIPS planning*, arXiv:1304.0897v1 [cs.AI], 2013.