Depth-Bound Heuristics and Iterative-Deepening Search Algorithms in Classical Planning

Bachelor's Thesis Presentation
Florian Spiess, 13 June 2017

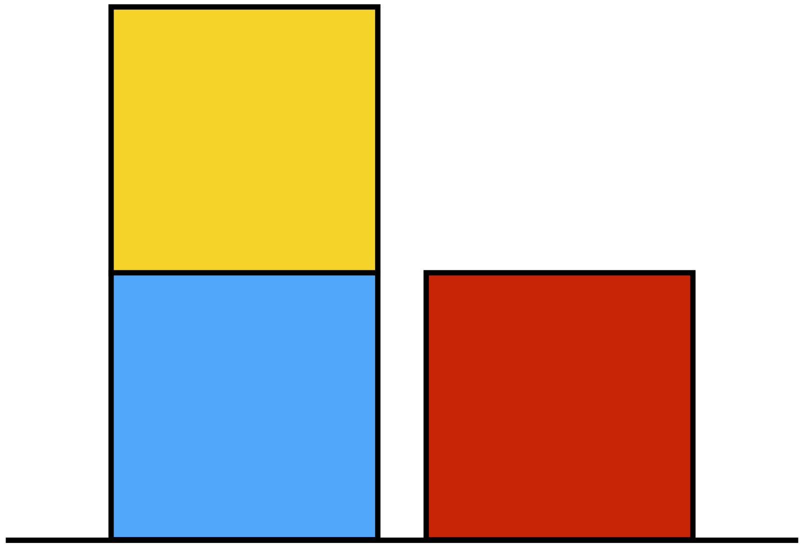
Departement of Mathematics and Computer Science
Artificial Intelligence

Classical Planning

- Goal: Find series of actions from initial to goal state
- Static, deterministic, fully observable, discrete, single-agent problems
- E.g.:
    - Shortest package delivery route
    - Stacking blocks

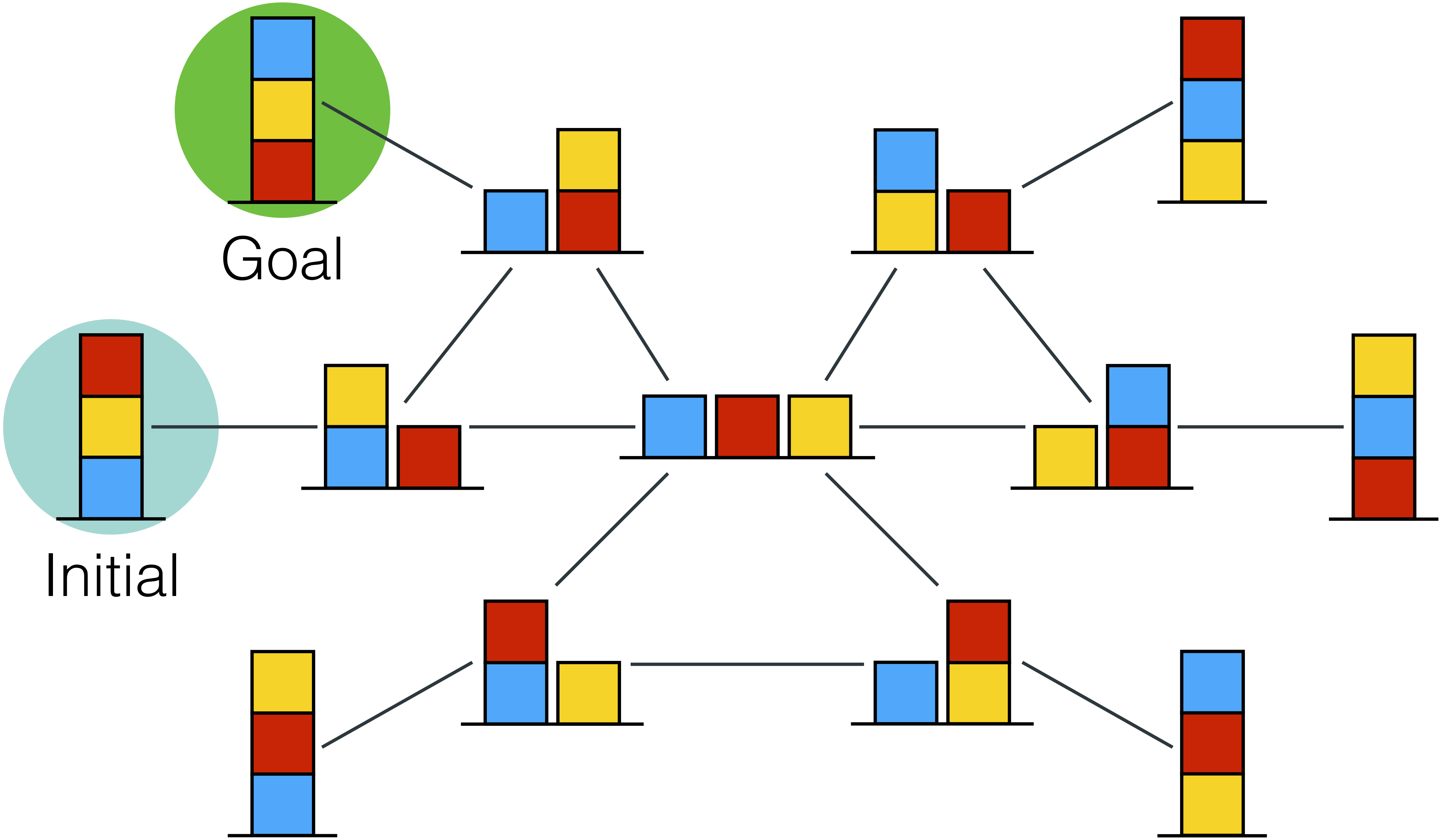
Blocks World

- Goal: Stack blocks in a certain order
- Only move one block at a time
- Only move blocks at the top of stacks



Blocks World

State Space



Heuristics

- Approximate goal distance
- Require time to construct / calculate

Goal

- Depth-bound heuristics
- Evaluate with iterative-deepening search algorithms
- Implementation in Fast Downward

Merge-and-Shrink

- Constructs abstract state space
- Calculates heuristic value in abstract state space

Merge-and-Shrink

State Space Representation

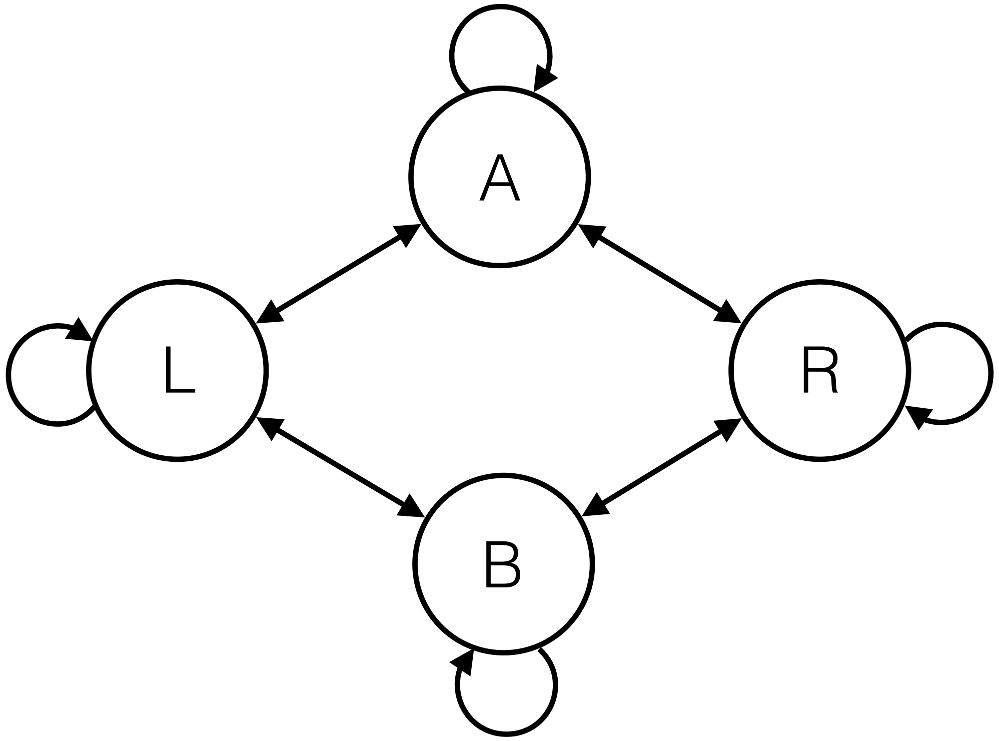
- States can be represented as lists of variables
- E.g. Logistics with one package, two trucks:
  - Package —> Left
  - Truck A —> Right
  - Truck B —> Left

Merge-and-Shrink

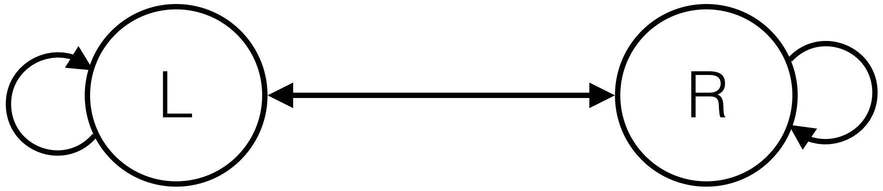
Projection

- Only considers state change of one variable
- E.g. projection onto:

Package



Truck A

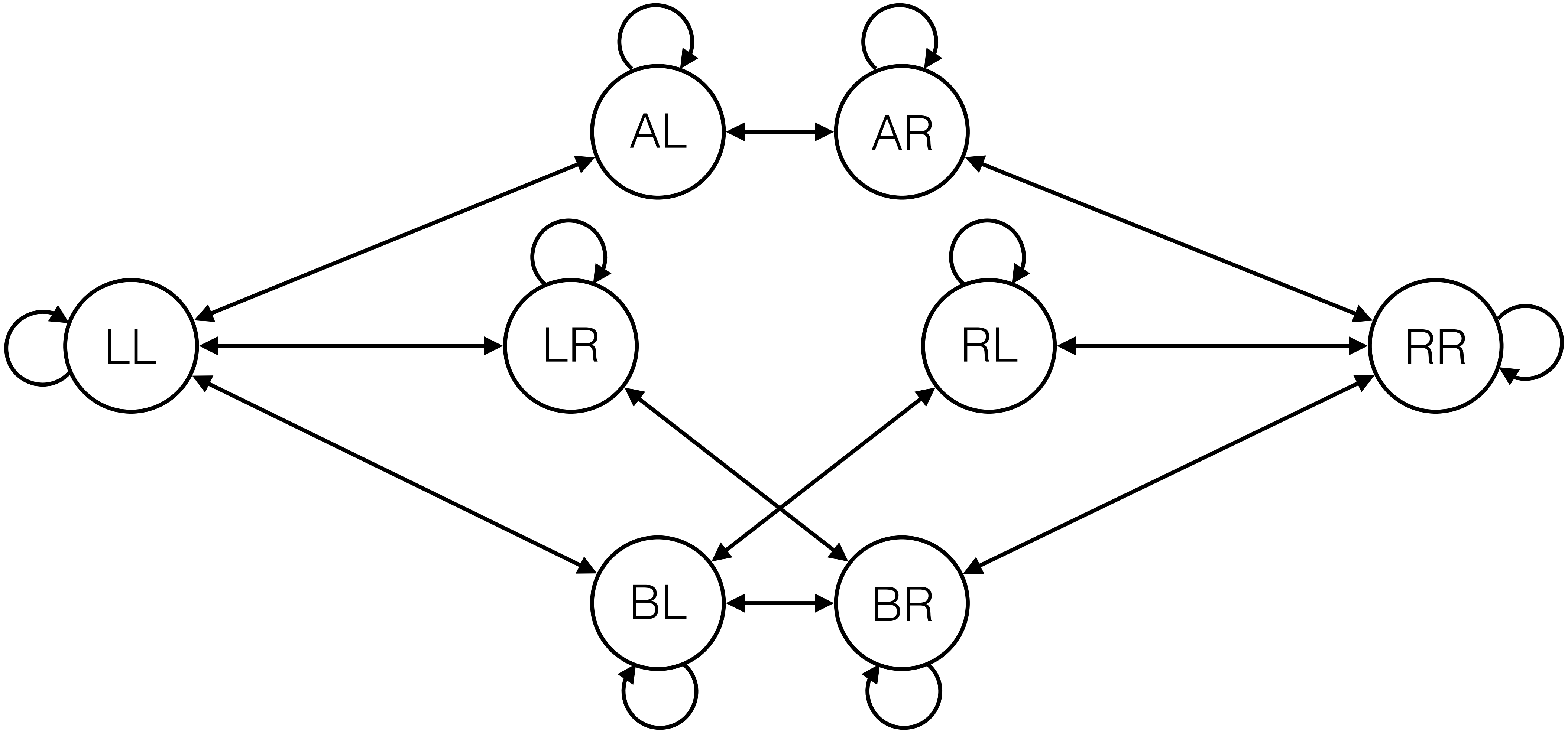


Merge-and-Shrink

Merge

- Merge through synchronized product

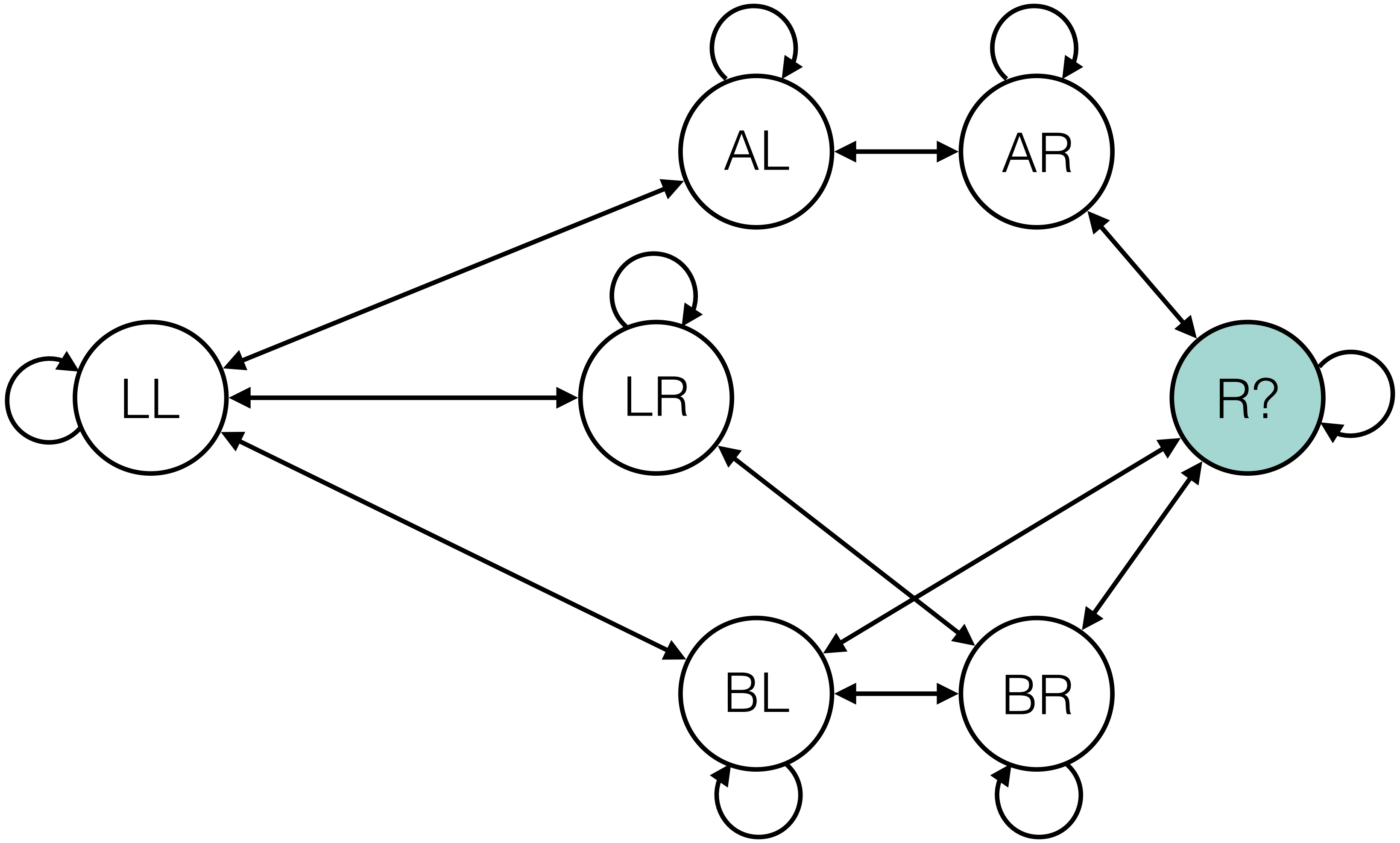
- E.g. merge of projections on Package and Truck A:



Merge-and-Shrink

Shrink

- Combine states to reduce size



Merge-and-Shrink

Modification

- Prune abstract states with cost > *f*-bound
  - Reduce construction time
  - Increase heuristic accuracy

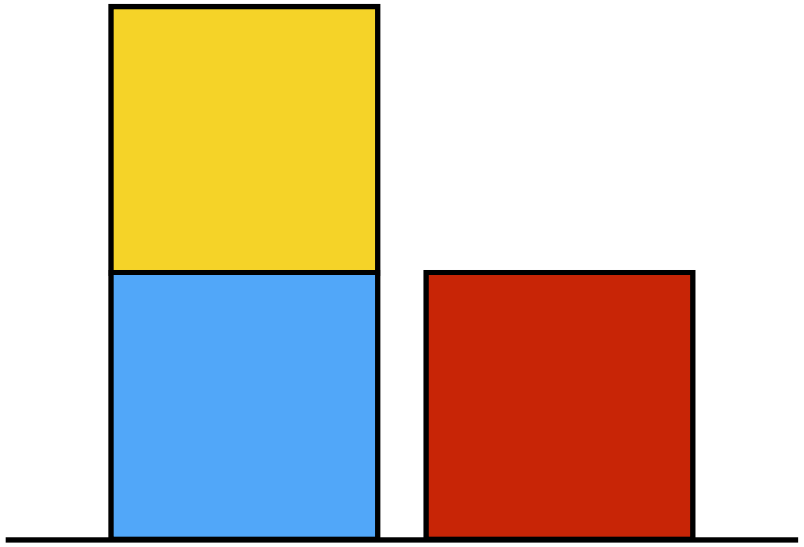
Landmark Cut

State Space Representation

- States can be represented as set of propositions

- E.g. Blocks world:

    - state = {Y-on-B, B-on-F, R-on-F}



Landmark Cut

Delete Free Planning Task

- Acquired proposition cannot be lost
- E.g.:
  {Y-on-F, B-on-F, R-on-F}

  — move Yellow onto Blue —>

  {Y-on-F, B-on-F, R-on-F, Y-on-B}

Landmark Cut

- Estimates the minimum cost of a delete free plan
- Iteratively sums costs of required actions

Landmark Cut

Modification

- Stop calculation once sum of costs > *f*-bound
    - Reduce calculation time

IDA* Search

- Iterative-deepening A*
- Tree search
- Explores paths until *f* > *f*-bound
- Restarts with increased *f*-bound
- No open list
- No closed list —> low memory usage

IDA* Search

Implementation

- Successor generation requires closed list in Fast Downward
- With closed list
- With duplicate detection

IDBFA* Search

- Iterative-deepening breadth-first A*
- A* search but prunes nodes with *f* > *f*-bound
- No solution —> increase *f*-bound

Breadth-First Heuristic Search

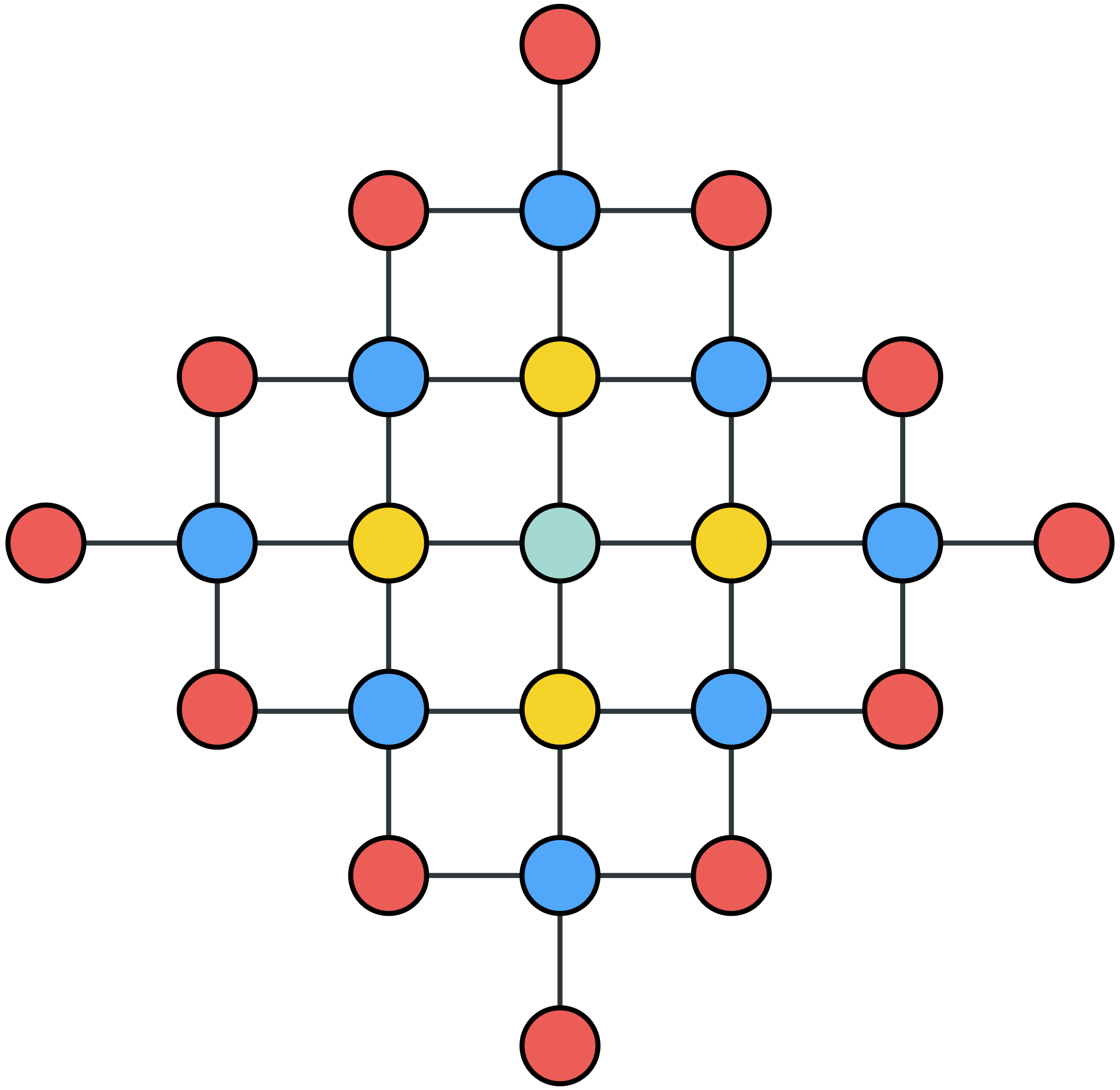
- Store explored nodes —> High space complexity
- Only search frontier required to find goal

  —> Delete visited nodes

  - No duplicate detection!
  - No solution path!

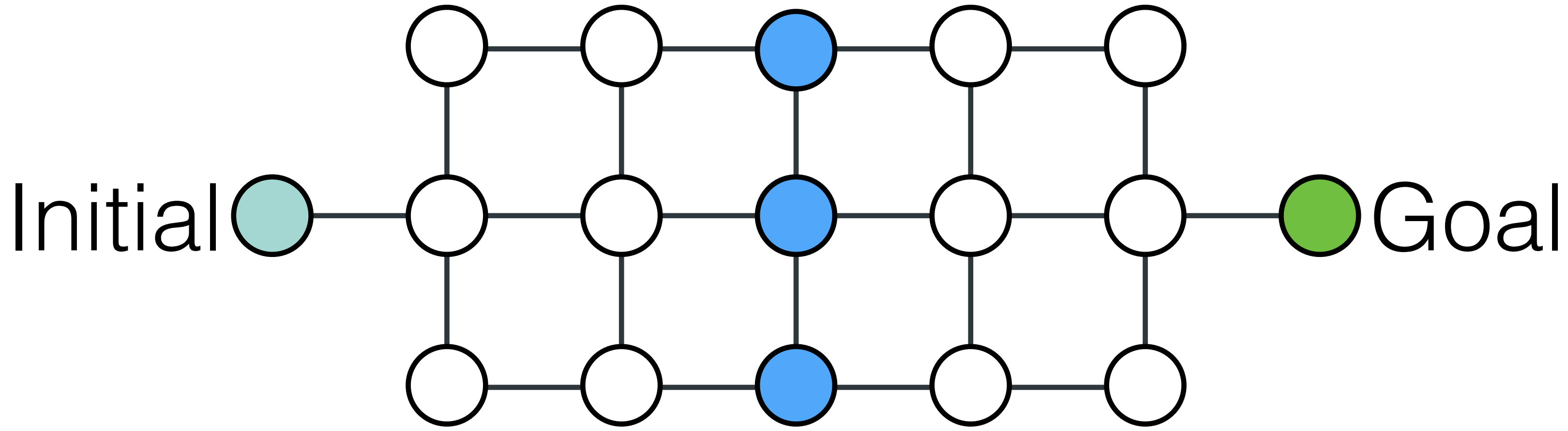
Breadth-First Heuristic Search

- Breadth-first search explores nodes in ‘depth-layers’



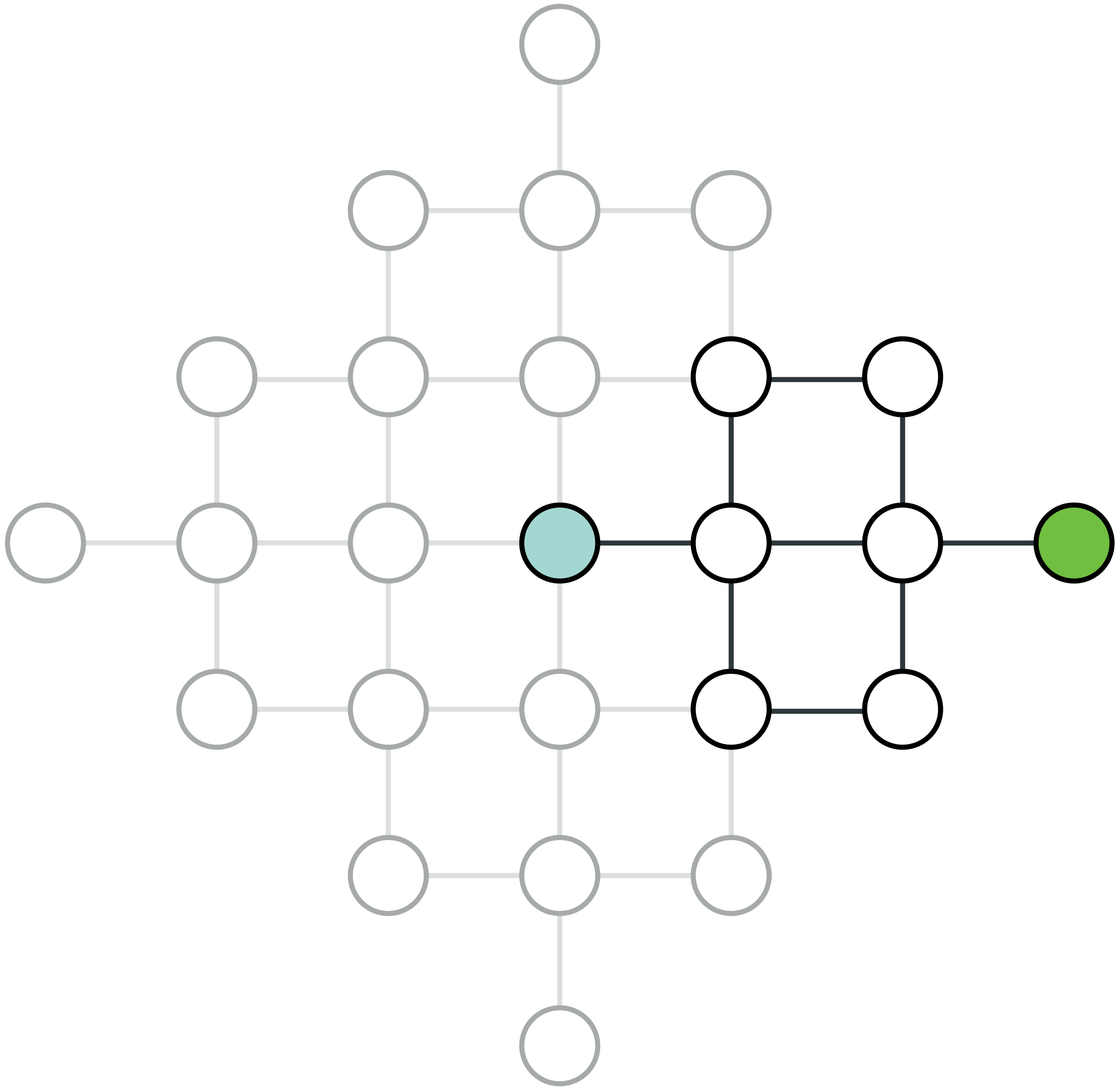
Breadth-First Heuristic Search

- Save one intermediate layer
- Recursively solve problems



Breadth-First Heuristic Search

- Nodes pruned with *f*-bound



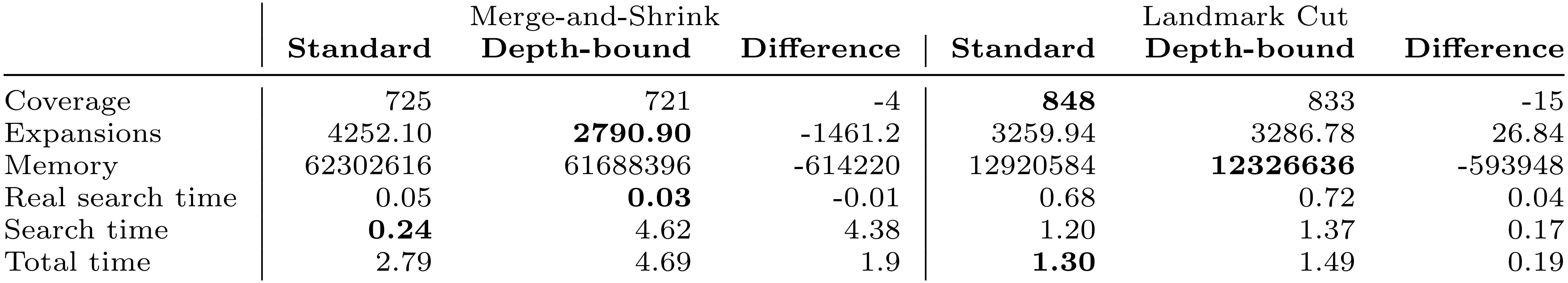
Evaluation

- Experiments on 1667 Tasks
  (from 57 domains)

- IDBFHS on subset of 160 Tasks
  (from 6 unit-cost, undirected graph domains)

Results

IDA* Comparison

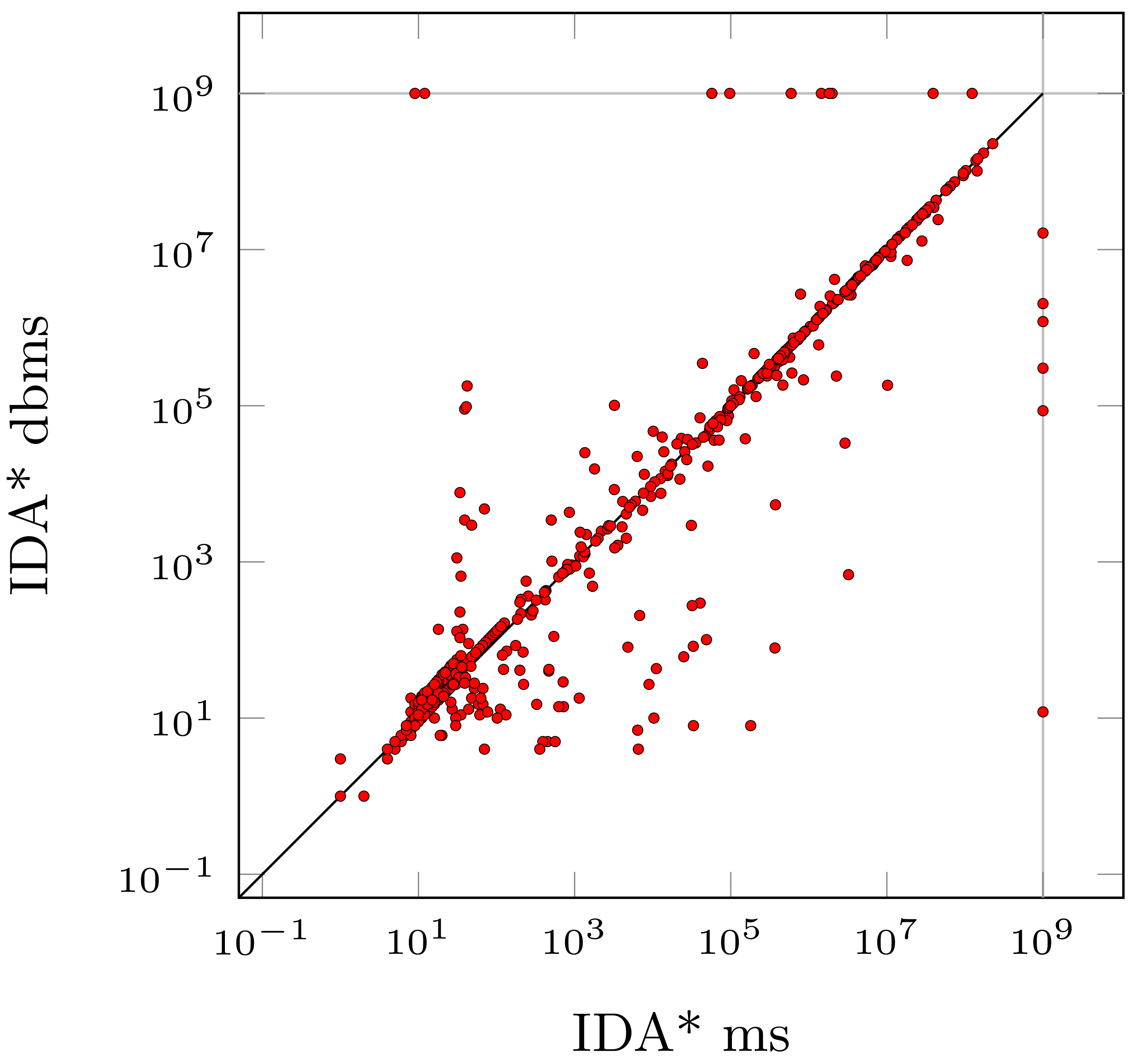


		Merge-and-Shrink			Landmark Cut	
	Standard	**Depth-bound**	**Difference**	**Standard**	**Depth-bound**	**Difference**
Coverage	725	721	-4	**848**	833	-15
Expansions	4252.10	**2790.90**	-1461.2	3259.94	3286.78	26.84
Memory	62302616	61688396	-614220	12920584	**12326636**	-593948
Real search time	0.05	**0.03**	-0.01	0.68	0.72	0.04
Search time	**0.24**	4.62	4.38	1.20	1.37	0.17
Total time	2.79	4.69	1.9	**1.30**	1.49	0.19

- Depth-bound heuristics have lower coverage
- Depth-bound heuristics are slower
- Depth-bound M&S requires fewer expansions

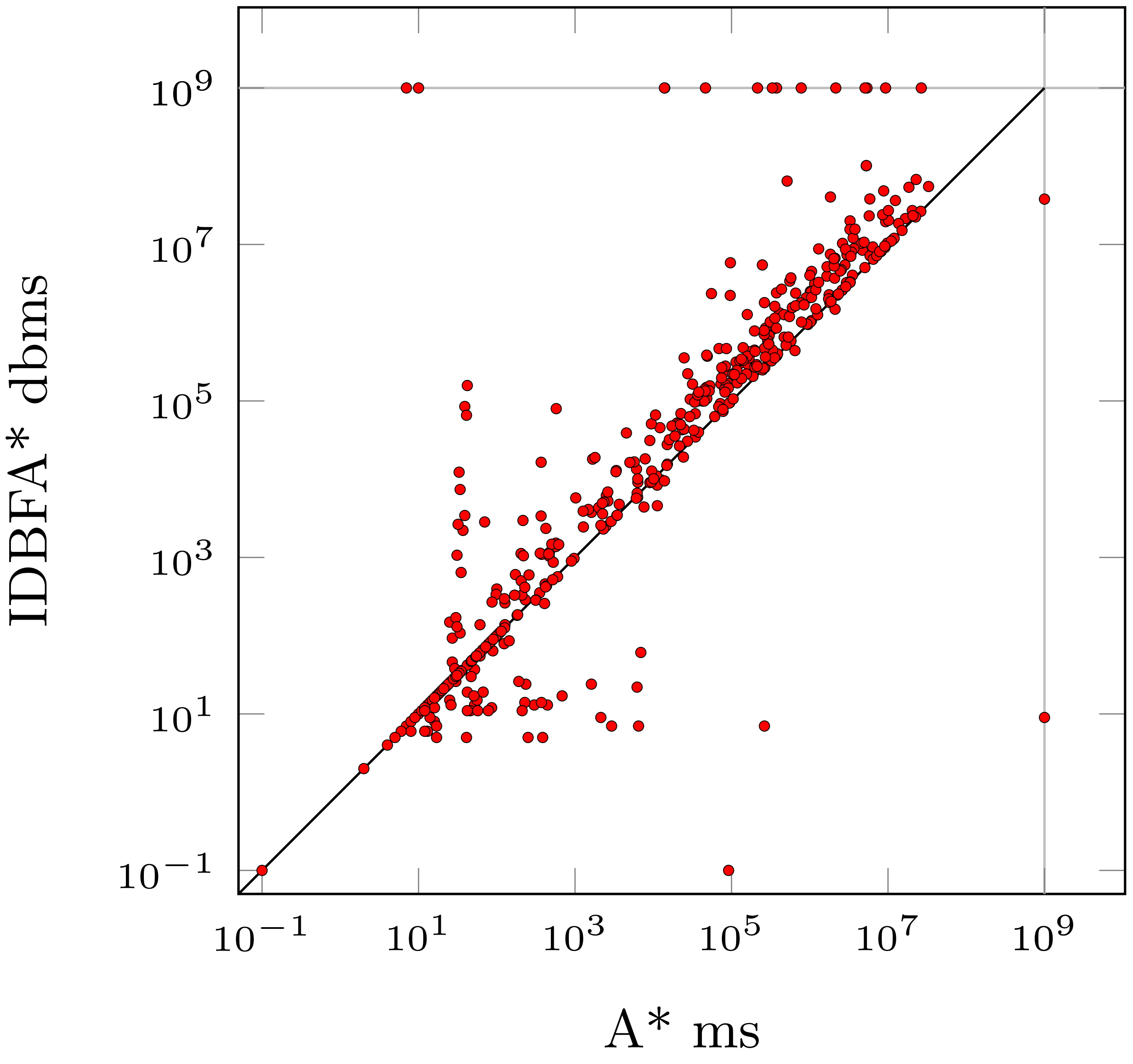
Results

Expansions



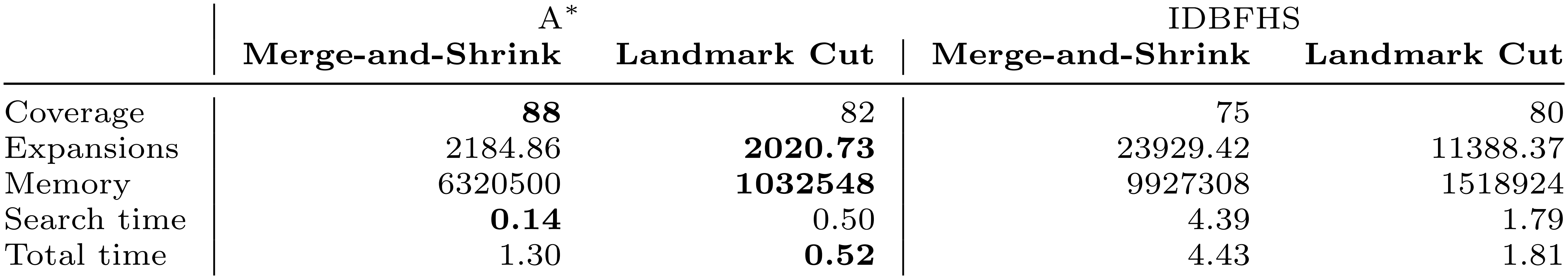
Results

Expansions



Results

A* and IDBFHS



	A*		IDBFHS	
	Merge-and-Shrink	**Landmark Cut**	**Merge-and-Shrink**	**Landmark Cut**
Coverage	**88**	82	75	80
Expansions	2184.86	**2020.73**	23929.42	11388.37
Memory	6320500	**1032548**	9927308	1518924
Search time	**0.14**	0.50	4.39	1.79
Total time	1.30	**0.52**	4.43	1.81

- IDBFHS completed fewer tasks than A*
- IDBFHS had higher peak memory

Conclusion

- Depth-bound LM-cut not enough time gain
- Depth-bound M&S slower because of construction
- Depth-bound M&S more accurate for easy tasks

Future Work

- Algorithm determines task complexity:
  - Simple: use depth-bound M&S
  - Complex: use unbound M&S
- Increase M&S depth-bound in greater steps

Thank you for your attention!

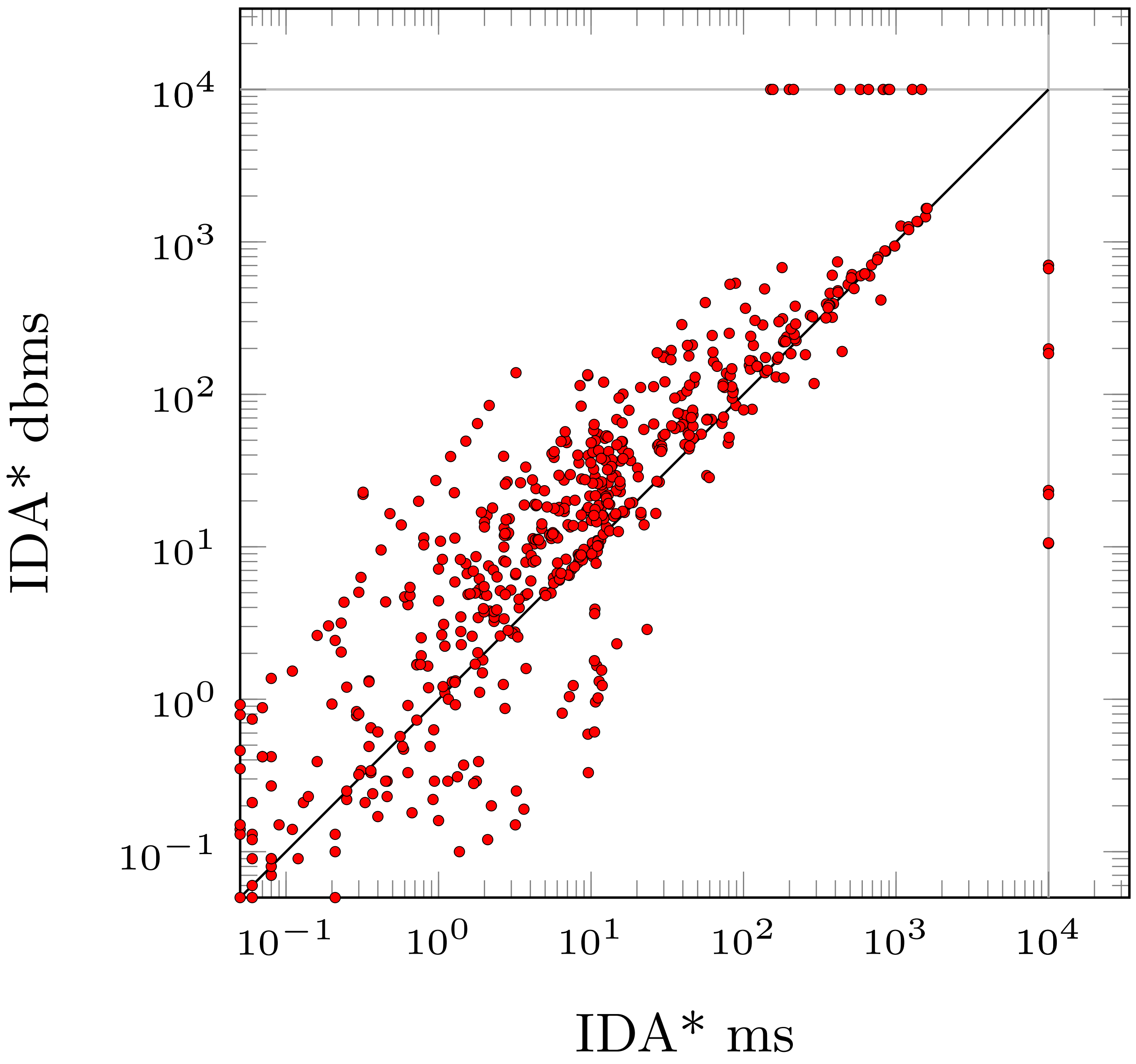
Results

Summary

	A*		IDA*				IDBFA*	
	ms	**lmcut**	**ms**	**lmcut**	**dbms**	**dblmcut**	**dbms**	**dblmcut**
Coverage	745	**882**	725	848	721	833	728	840
Expansions	1822.21	**1301.20**	3939.90	3088.52	2587.65	3113.72	2389.86	3079.64
Memory	63368336	21006000	53595072	9802372	52926128	**9409960**	60730232	20403740
Search time	**0.13**	0.60	0.22	1.12	4.46	1.28	4.76	1.33
Total time	2.01	**0.65**	2.68	1.22	4.53	1.40	5.07	1.45

Results

Total Time


10^4
10^3
10^2
10^1
10^0
10^{-1}
IDA* dbms
10^{-1}
10^0
10^1
10^2
10^3
10^4
IDA* ms