Learning Heuristic Functions in Classical Planning

Master Thesis

Cedric Geissmann

March 15, 2016

Natural Science Faculty of the University of Basel
Department of Mathematics and Computer Science
Artificial Intelligence
http://ai.cs.unibas.ch

UNI
BASEL

Outline

1. Background
   - Planning
   - Machine Learning
   - Artificial Neural Networks
2. Learning a Heuristic Function
   - Search Strategy
   - Walk Strategy
   - Prediction Strategy
3. Results
   - Search Strategy
   - Walk Strategy
   - Prediction Strategy
   - Discussion
4. Conclusion and Future Work

Background

Planning

Planning task:

- Variables
- States assign values to variables
  - Initial state s_0
  - Goal states $s_\star$
- Operators have preconditions and effects

Heuristic Search

- Heuristic search in state space
- Heuristic function
  - Estimates cost to nearest goal
- Use heuristic function to guide the search towards the goal

Machine Learning

Machine Learning ...

...is the task of learning from data and make predictions on data.

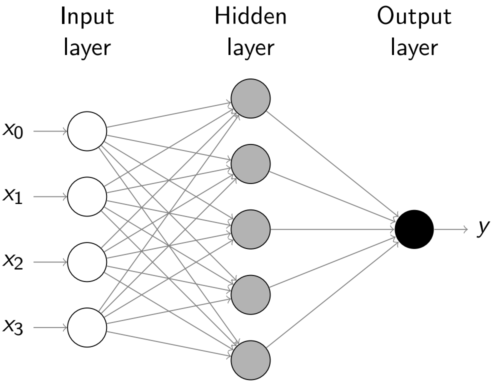
- Set of input values $\vec{x}$
- Unknown target function $f^\star : \vec{x} \rightarrow \vec{y}$
- Set of function hypotheses $F = \{f \mid f : \vec{x} \rightarrow \vec{y}\}$

Goal

Find a function f that approximates $f^\star$ the best.

Artificial Neural Network

- Inspired by the brain
- Can predict data
- Needs to be trained



From Regression to Heuristic Function

- Regression
  - Find a function $f : \vec{x} \rightarrow \vec{y}$
- Heuristic function
  - Function $h : \vec{v} \rightarrow h_{val}$

$f = h$

- $\vec{x} = \vec{v}$
- $\vec{y} = h_{val}$

- $\vec{x}$ use variable values of any state s
- $\vec{y}$ use distance from state s to the goal

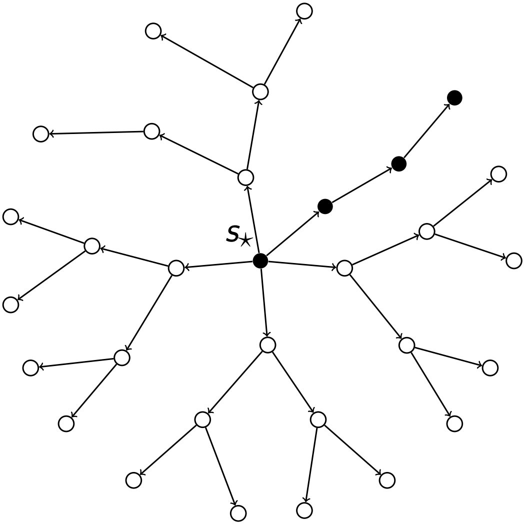
Learning a Heuristic Function

Learning a Heuristic Function

- Use ANN as heuristic function
- Train ANN with *back-propagation*
- Generate training set for *back-propagation*
  - Search Strategy
  - Walk Strategy
  - Prediction Strategy

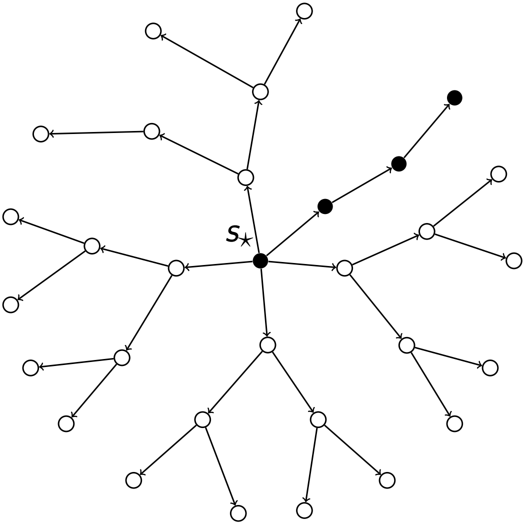
Generate the Training Set with Search Strategy

- Chose random goal state $s_\star$
- Perform random walk starting at $s_\star$
- Search from random walk endpoint
- Add every state from the solution path to the training set



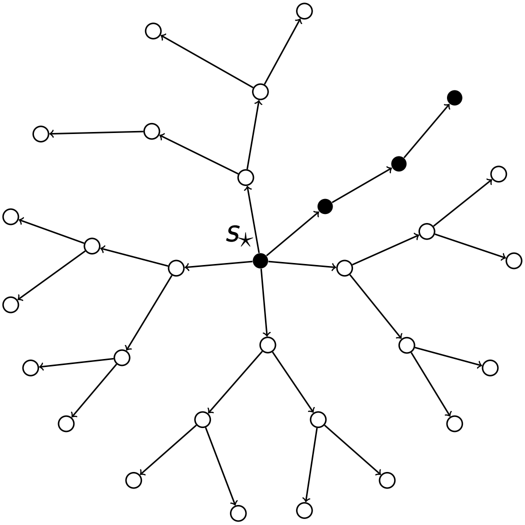
Generate the Training Set with Walk Strategy

- Chose random goal state $s_\star$
- Perform random walk starting at $s_\star$
- Add every state on the random walk to the training set



Generate the Training Set with Prediction Strategy

- Chose random goal state $s_\star$
- Perform random walk starting at $s_\star$
- Add every state on the random walk to the training set
- Use a *solution cost predictor* to estimate distance



Results

Baseline

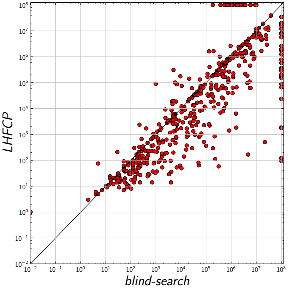
Learning Heuristic Functions in Classical Planning (LHFCP)

	blind-search	**LHFCP**	h^{FF}
Coverage	680	693	1308
Expansions	53369.73	24990.02	282.75
Search time	1.31	1.62	0.25
Total time	1.35	8.66	0.27

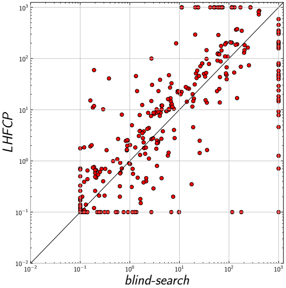
- Slightly better than *blind-search*
- High **total time**
- Higher **search time** than *blind-search*
- Worse than h^{FF}

Performance against *blind-search*

Expansions:

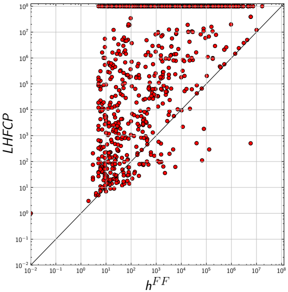


Search time:

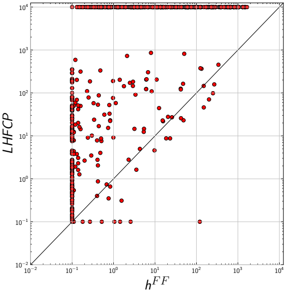


Performance against h^{FF}

Expansions:



Search time:



Expansions Gripper

	blind-search	**LHFCP**	h^{FF}
prob01.pddl	253	110	123
prob02.pddl	1853	993	1413
prob03.pddl	11773	10648	10735
prob04.pddl	68605	65716	66585
prob05.pddl	376829	370598	373347
prob06.pddl	1982461	1913835	1976941
prob07.pddl	10092541	10051648	10084311

- *LHFCP* performs better on domain **gripper** than h^{FF}
- Also on the domain **psr-small**

Random Walk Length

	rwl-20	**default**	**rwl-100**
Coverage	687	693	690
Expansions	29221.89	29205.89	27534.29
Search time	1.87	1.84	1.79
Total time	9.32	9.49	10.02
Training set size	396.76	457.66	519.24

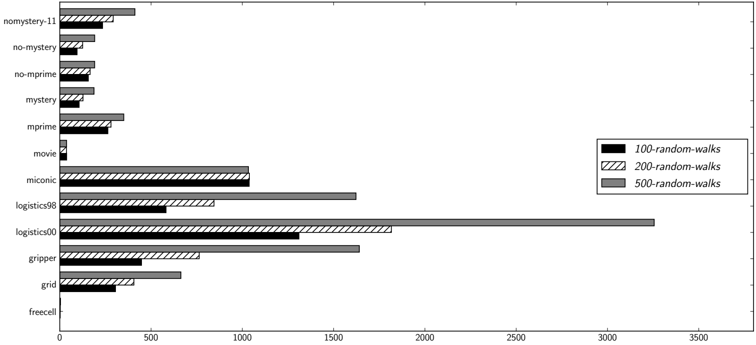
- Longer random walks produce bigger **training set**
- Longer random walks produce slightly better heuristic
- Longer random walks need more **setup time**

Number of Random Walks

	nrw-100	default	nrw-500
Coverage	701	693	675
Expansions	26726.12	25701.05	24337.35
Search time	1.76	1.73	1.70
Total time	7.79	8.95	11.79
Training set size	311.93	466.80	912.01

- More random walks produce bigger **training set**
- More random walks produce slightly better heuristic
- More random walks need more **setup time**

Training Set Size for Number of Random Walks



- Not possible to generate **training set** on each domain
- Size of **training set** does not scale linear on every domain

ANN-Topology

	topo-n-20-1	**default**	**topo-n-100-1**
Coverage	679	693	689
Expansions	35034.57	28690.82	26111.88
Search time	1.60	1.81	2.31
Total time	7.86	9.43	12.05
Training set size	454.60	453.10	452.80

- More neurons produce slightly better heuristic
- More neurons need more **setup time**
- More neurons need more **search time**

Different Initial Heuristic Functions h_0

	h_0-blind-search	h_0-FF	h_0-ipdb	h_0-lm-cut
Coverage	683	705	604	699
Expansions	20814.37	19795.51	21012.24	20664.96
Search time	1.48	1.39	1.41	1.40
Total time	10.79	5.53	51.03	6.46
Training set size	181.35	295.55	242.87	234.74

- *FF* produces the strongest heuristic
- *blind-search* has the smallest **training set**
- *ipdb* has highest **setup time**

Walk Strategy

	search-strategy	**walk-strategy**
Coverage	693	692
Expansions	25367.74	25153.76
Search time	1.65	1.61
Total time	8.80	7.82
Training set size	453.09	1850.41

- Both approaches perform about the same
- Lower **total time** with *walk-strategy*
- Bigger **training set** with *walk-strategy*

Prediction Strategy

	search-strategy	**prediction-strategy**
Coverage	49	19
Expansions	21.21	53.91
Search time	0.10	0.10
Total time	0.36	33.87
Training set size	274.89	760.42

- Only executed on domain **psr-small**
- Lower **coverage** with *prediction-strategy*
- Higher **total time** with *prediction-strategy*
- Bigger **training set** with *prediction-strategy*

Discussion

- Random walk length affects training set size
- Number of random walks affects training set size
- ANN-topology affect total time
- Initial heuristic should have small setup time
- Search can be omitted
- Solution cost predictor too resource hungry

Conclusion and Future Work

Conclusion

- Conclusion
  - Has high setup time
  - Depends on many parameters
  - Inverse operators
  - Domain-independence not possible
  - Can be better than h^{FF} on some domains
- Future Work
  - Adopt parameters to current problem
  - Use other features
  - Use other machine learning techniques

Question?