Learning to Combine Admissible Heuristics Under Bounded Time

C. Domshlak E. Karpas S. Markovitch

Faculty of Industrial Engineering and Management

Faculty of Computer Science
Technion

20.9.2009 - ICAPS - Learning on Planning and Learning

Outline

Motivation

- Domain independent optimal planning
  - A^* + admissible heuristic (almost always)
  - Which heuristic to use?
- Sample results:

Domain	h_{LA}	$h_{\text{LM-CUT}}$
airport	25	**38**
freecell	**28**	15

Combining Admissible Heuristics

- Why use only one heuristic?
- Simplest combination method: $\max_h$
- Sample results:

Domain	h_{LA}	$h_{\text{LM-CUT}}$	$\max_h$
airport	25	**38**	36
freecell	**28**	15	22

- Other combination methods exist (additive heuristics, additive/disjunctive, ...)

Combining Admissible Heuristics (2)

- The problem with $\max_h$
  - We need to compute many heuristic functions
  - The heuristic value is the result of only one computation
  - Some computation is wasted
- Possible solution: learn a classifier which predicts which heuristic will be the “winner”

Informative vs. Fast Heuristics

- Sometimes spending a lot of time to compute the most informed heuristic is not the best thing to do

Outline

Theoretical Model - Which Heuristic to Compute When?

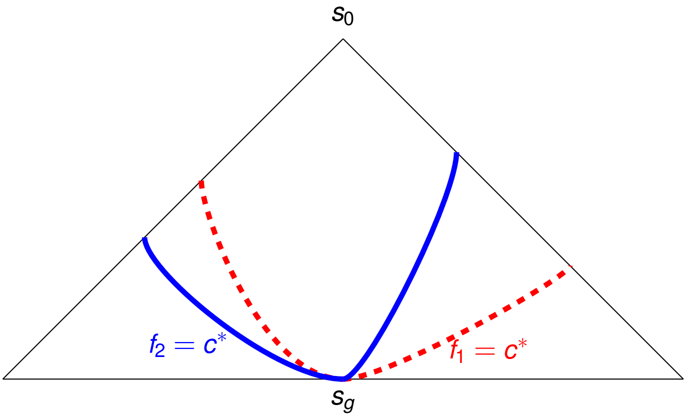
Assumptions

- State space is a tree
- Single goal state
- Uniform cost actions
- Constant branching factor b
- Perfect knowledge

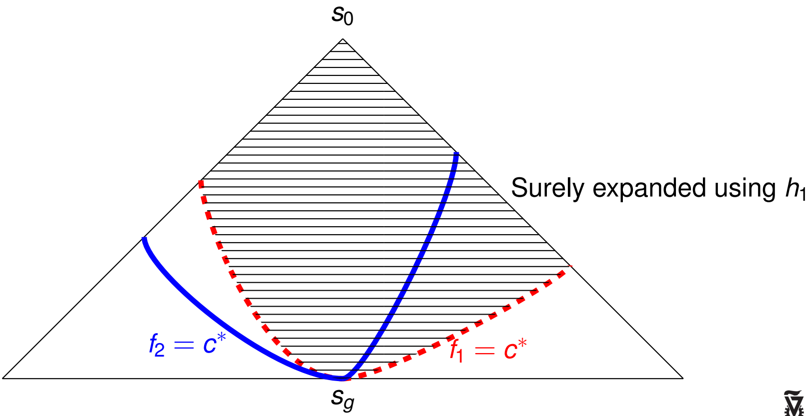
Two heuristics: h_1 and h_2

- Consistent
- Evaluating h_i takes time t_i

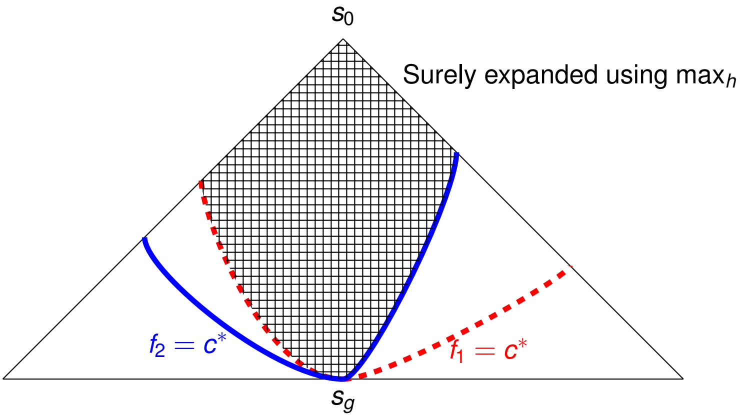
Theoretical Model - Which Heuristic to Compute When?



Theoretical Model - Which Heuristic to Compute When?

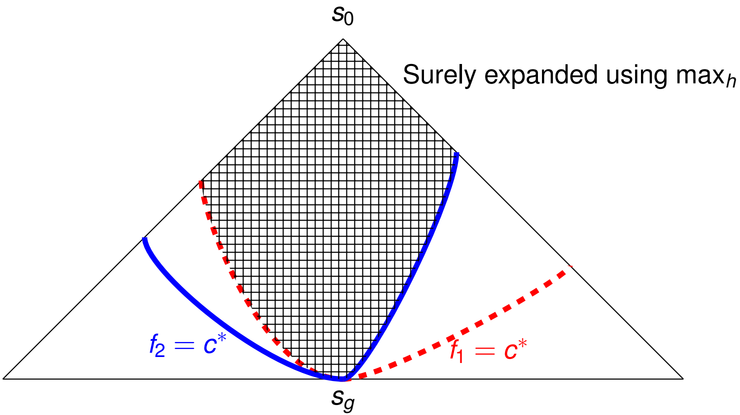


Theoretical Model - Which Heuristic to Compute When?

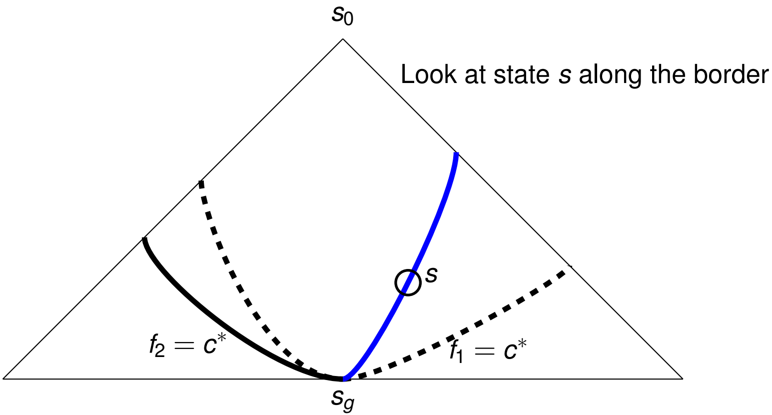


Theoretical Model - Which Heuristic to Compute When?

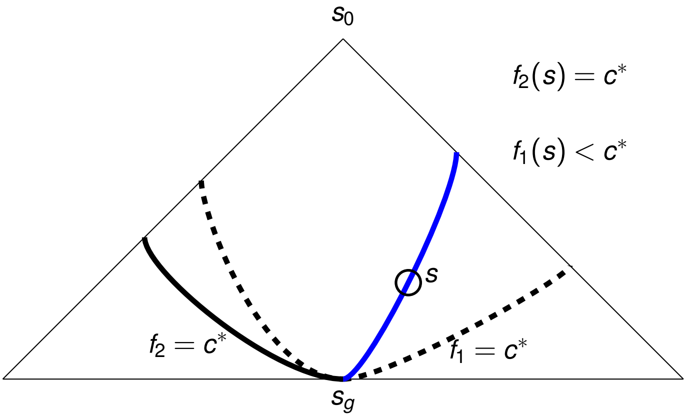
Best decision - just expand, don't evaluate



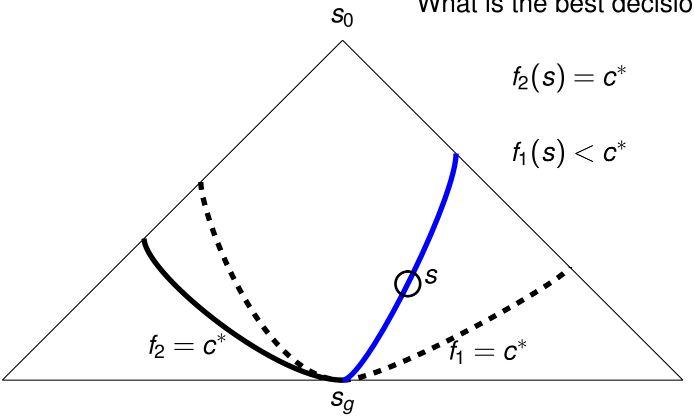
Theoretical Model - Which Heuristic to Compute When?



Theoretical Model - Which Heuristic to Compute When?



Theoretical Model - Which Heuristic to Compute When?



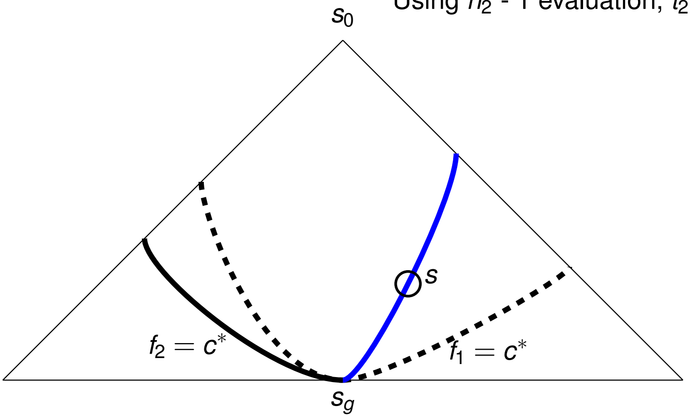
What is the best decision?

$f_2(s) = c^*$

$f_1(s) < c^*$

Theoretical Model - Which Heuristic to Compute When?

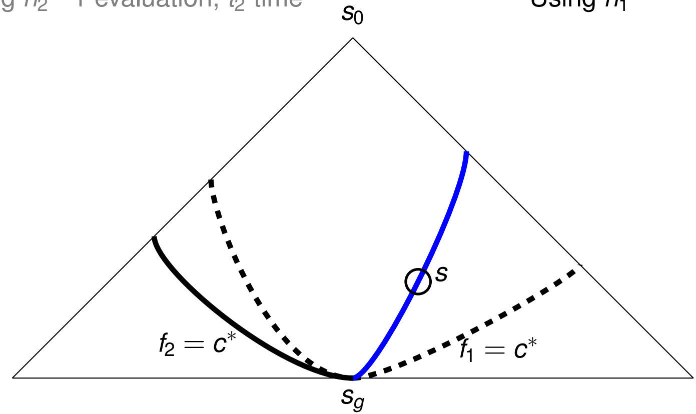
Using h_2 - 1 evaluation, t_2 time



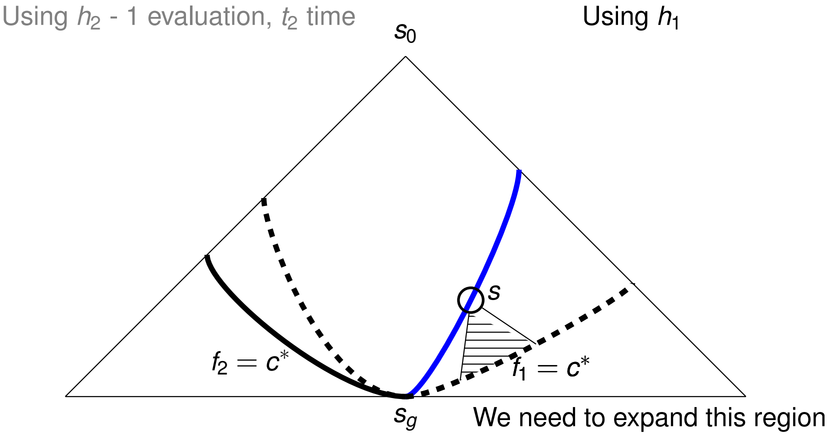
Theoretical Model - Which Heuristic to Compute When?

Using h_2 - 1 evaluation, t_2 time

Using h_1



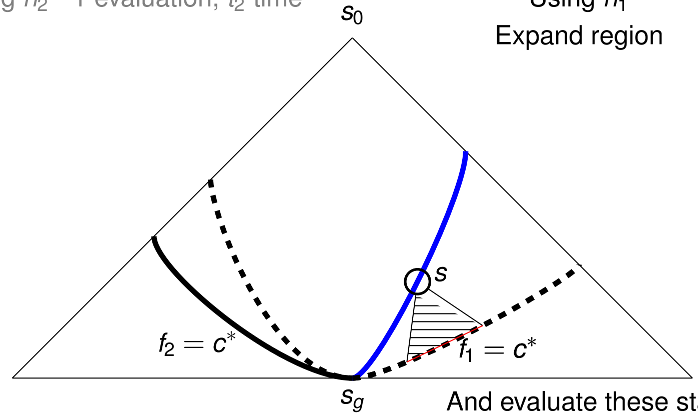
Theoretical Model - Which Heuristic to Compute When?



Theoretical Model - Which Heuristic to Compute When?

Using h_2 - 1 evaluation, t_2 time

Using h_1
Expand region

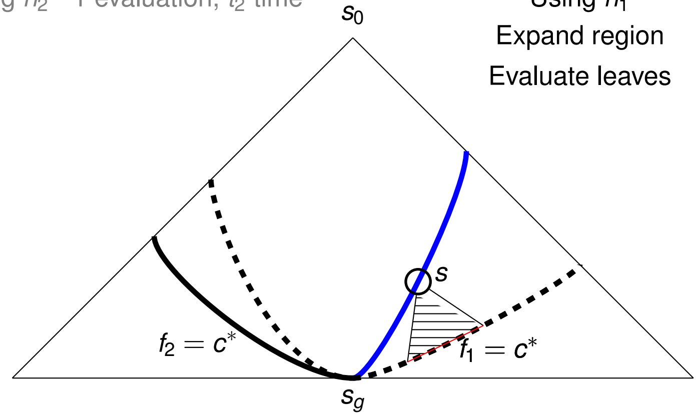


And evaluate these states

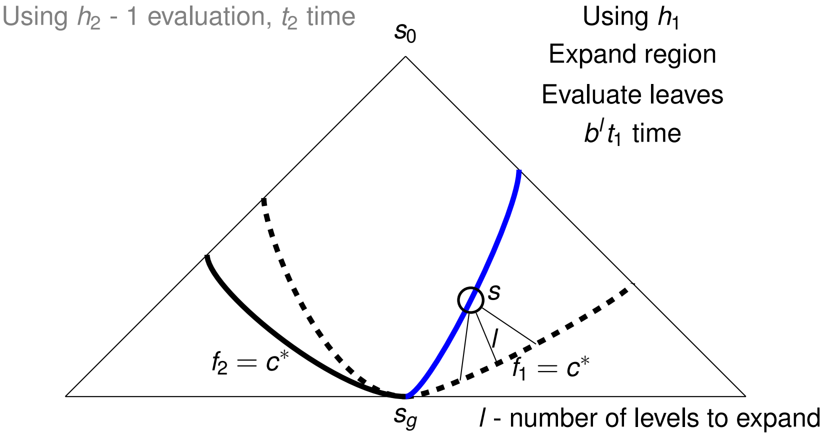
Theoretical Model - Which Heuristic to Compute When?

Using h_2 - 1 evaluation, t_2 time

Using h_1
Expand region
Evaluate leaves



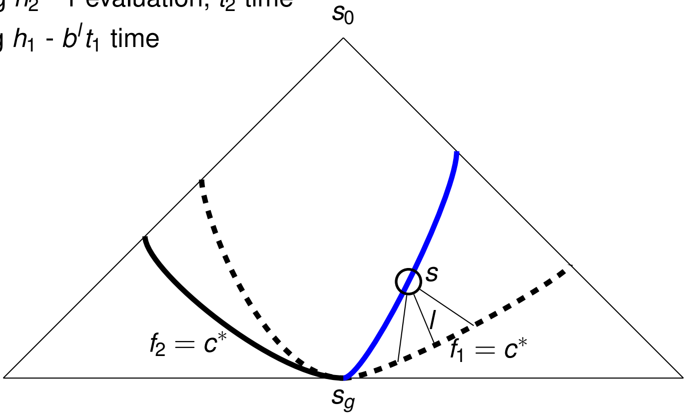
Theoretical Model - Which Heuristic to Compute When?



Theoretical Model - Which Heuristic to Compute When?

Using h_2 - 1 evaluation, t_2 time

Using h_1 - $b^l t_1$ time

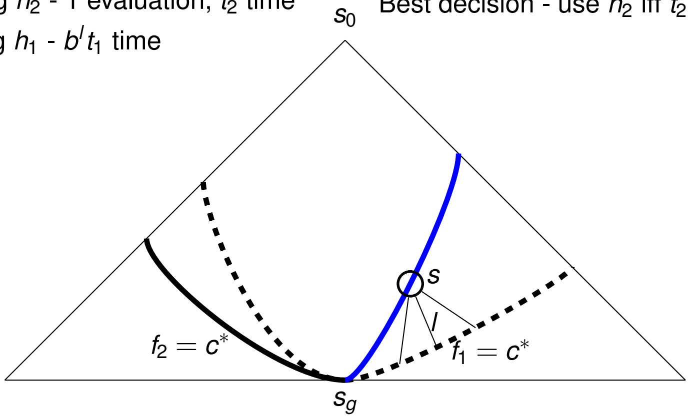


Theoretical Model - Which Heuristic to Compute When?

Using h_2 - 1 evaluation, t_2 time

Using h_1 - $b^l t_1$ time

Best decision - use h_2 iff $t_2 < b^l t_1$



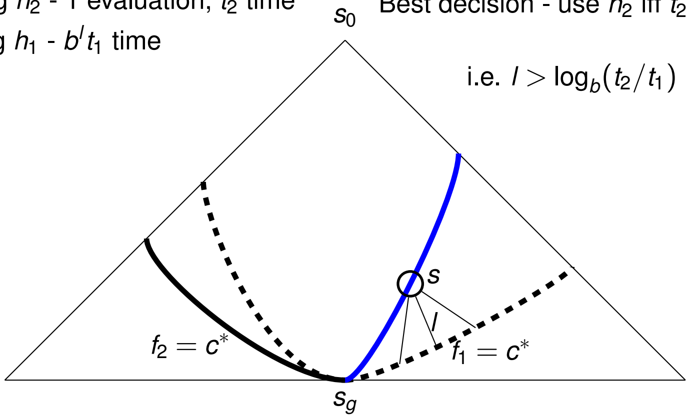
Theoretical Model - Which Heuristic to Compute When?

Using h_2 - 1 evaluation, t_2 time

Using h_1 - $b^l t_1$ time

Best decision - use h_2 iff $t_2 < b^l t_1$

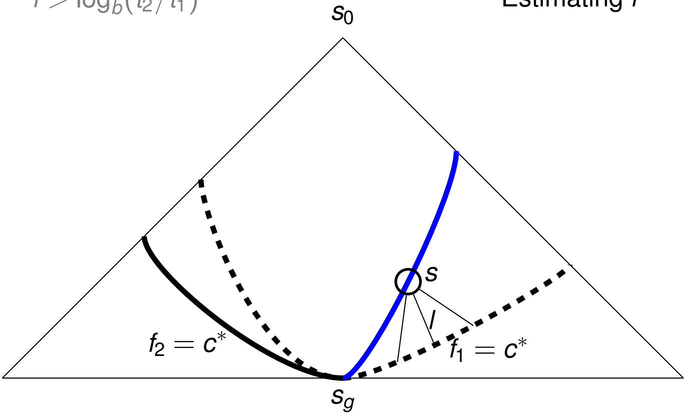
i.e. $l > \log_b(t_2/t_1)$



Theoretical Model - Which Heuristic to Compute When?

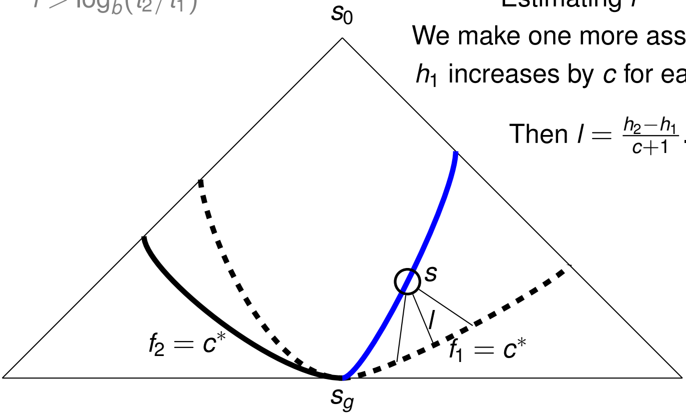
$l > \log_b(t_2/t_1)$

Estimating l



Theoretical Model - Which Heuristic to Compute When?

$l > \log_b(t_2/t_1)$



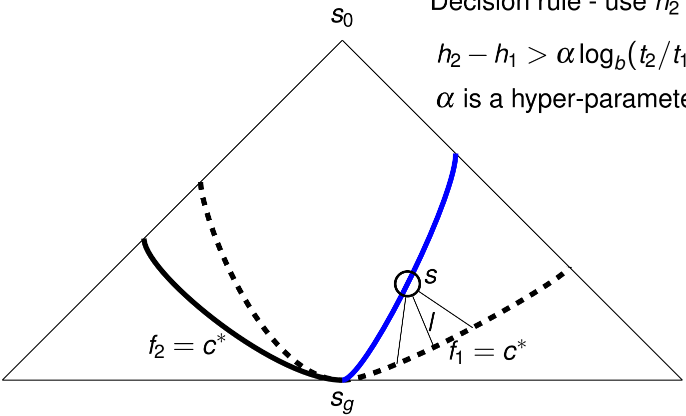
Estimating l

We make one more assumption:

h_1 increases by c for each level.

Then $l = \frac{h_2 - h_1}{c+1}$.

Theoretical Model - Which Heuristic to Compute When?



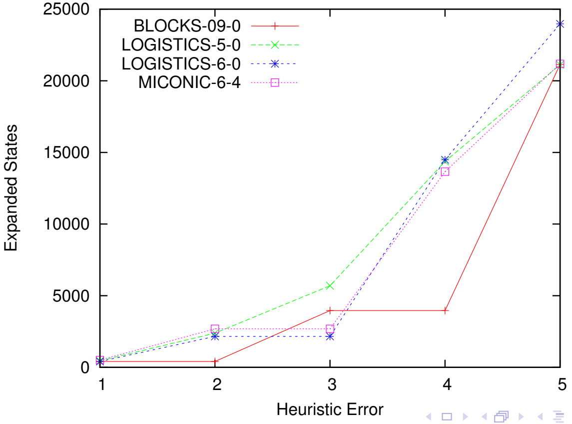
Decision rule - use h_2 iff

$$h_2 - h_1 > \alpha \log_b(t_2/t_1)$$

α is a hyper-parameter

Justifying the Rule

- The decision rule derived from the model can be justified by some empirical results (Helmert and Röger, 2008)



Outline

Dealing with Model Assumptions

Assumptions

- State space is a tree
- Single goal state
- Uniform cost actions
- Constant branching factor b
- Perfect knowledge

Two heuristics: h_1 and h_2

- Consistent
- Evaluating h_i takes time t_i

Dealing with Model Assumptions

Assumptions

- State space is a tree - doesn't change the rule
- Single goal state - doesn't change the rule
- Uniform cost actions - doesn't change the rule
- Constant branching factor b
- Perfect knowledge

Two heuristics: h_1 and h_2

- Consistent - doesn't change the rule
- Evaluating h_i takes time t_i

Dealing with Model Assumptions

Assumptions

- State space is a tree - doesn't change the rule
- Single goal state - doesn't change the rule
- Uniform cost actions - doesn't change the rule
- Constant branching factor b - estimate
- Perfect knowledge

Two heuristics: h_1 and h_2

- Consistent - doesn't change the rule
- Evaluating h_i takes time t_i - estimate

Dealing with Model Assumptions

Assumptions

- State space is a tree - doesn't change the rule
- Single goal state - doesn't change the rule
- Uniform cost actions - doesn't change the rule
- Constant branching factor b - estimate
- Perfect knowledge - use decision rule at every state

Two heuristics: h_1 and h_2

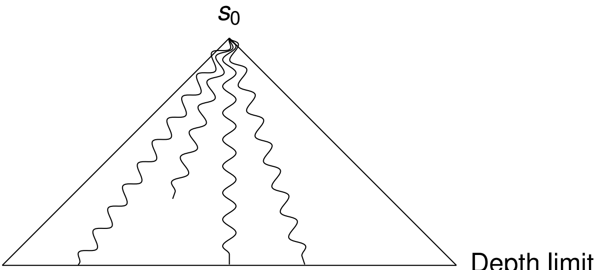
- Consistent - doesn't change the rule
- Evaluating h_i takes time t_i - estimate

Learning

- Collecting training examples
- Labeling training examples
- Generating features
- Building a classifier

Collecting Training Examples

- State space is sampled using stochastic hill climbing “probes”
  - Depth limit = $2 * h(s_0)$
  - Probabiity of expanding successor $s = 1/h(s)$
- All *generated* states are added to the training set
- Probing stops when enough training examples are collected



Labeling Training Examples

- b, t_1, t_2 are estimated from the collected examples
- $h_2 - h_1$ is calculated for each state
- Each example is labeled by h_2 iff $h_2 - h_1 > \alpha \log_b(t_2/t_1)$

- WLOG $t_2 > t_1$ - the choice is always whether to evaluate the more expensive heuristic

Generating Features

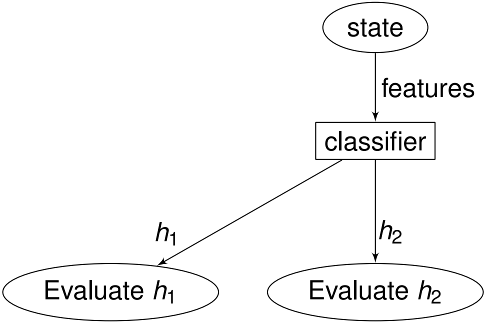
- We use the simplest features - values of state variables
- Better features will probably lead to better results

Building a Classifier

- We use the Naive Bayes classifier
  - Very fast
  - Incremental
  - Provides probability distribution for classification

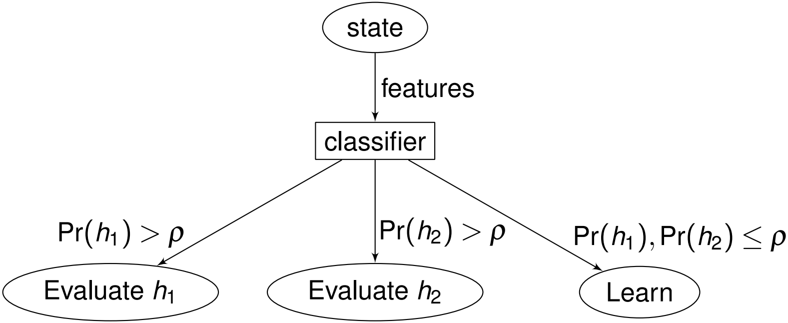
Using the classifier

State Evaluation



Using the classifier

State Evaluation



Final Remarks

- This is an active online learning scheme
- Using $\textsc{sas}^+$ helps, because it reduces dependence between state variables
- This approach can be easily extended to multiple heuristics
  - Learn a classifier for each pair
  - Decide which heuristic to use by voting

Outline

Evaluation

- We used two state of the art heuristics
  - $h_{\text{LM-CUT}}$ - Helmert and Domshlak 2009
  - h_{LA} - Karpas and Domshlak 2009
- Parameters
  - $\alpha = 1$ - decision rule bias
  - $\rho = 0.6$ - confidence threshold
  - Training set size = 100

Results - Solved Problems

Domain	h_{LA}	$h_{\text{LM-CUT}}$	$\max_h$	rnd_h	sel_h
airport	25	38	36	29	36
blocks	20	28	28	28	28
depots	7	7	7	7	7
driverlog	14	14	14	14	14
freecell	28	15	22	15	28
grid	2	2	2	2	2
gripper	6	6	6	6	6
logistics-2000	19	20	20	20	20
logistics-98	5	6	6	5	6
miconic	140	140	140	140	140
mprime	21	25	25	19	25
mystery	13	17	17	14	17
openstacks	7	7	7	7	7
pathways	4	5	5	4	5
psr-small	48	49	48	48	48
pw-notankage	16	17	17	17	17
pw-tankange	9	11	11	10	11
rovers	6	7	7	6	7
satellite	7	8	9	7	9
tpp	6	6	6	6	6
trucks	7	10	9	7	9
zenotravel	9	12	12	10	12
Total	419	450	454	421	**460**

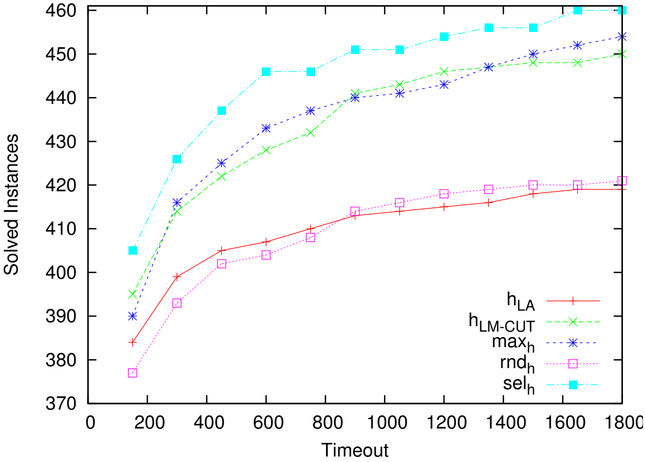
Results - Expanded States

Domain	h_{LA}	$h_{\text{LM-CUT}}$	$\max_h$	rnd_h	sel_h
airport (25)	20.41 (1.48)	1 (1)	1 (1)	1.09 (1.03)	1.16 (1)
blocks (20)	10.87 (4.1)	1 (1)	1 (1)	1.25 (1.24)	1.15 (1.01)
depots (7)	17.08 (16.7)	1 (1)	1 (1)	2.59 (2.35)	1.9 (1.02)
driverlog (14)	14.95 (9.46)	1.08 (1)	1 (1)	2.2 (2.2)	2.27 (1.44)
freecell (15)	1.24 (1.11)	188.57 (22.5)	1 (1)	17.47 (2.29)	1.88 (1.37)
grid (2)	3.38 (3.38)	1.02 (1.02)	1 (1)	1.47 (1.47)	3.32 (3.32)
gripper (6)	1 (1)	1.05 (1.01)	1 (1)	1.01 (1)	1 (1)
logistics-2000 (19)	1 (1)	1 (1)	1 (1)	1 (1)	1 (1)
logistics-98 (5)	8.43 (4.78)	1 (1)	1 (1)	1.66 (1.68)	4.67 (1.8)
miconic (140)	1 (1)	1 (1)	1 (1)	1 (1)	1 (1)
mprime (19)	415.47 (6.4)	3.44 (1)	1 (1)	27.06 (2.31)	6.34 (1.13)
mystery (12)	144.33 (1.06)	1.3 (1)	1 (1)	12.08 (1.29)	3.85 (1)
openstacks (7)	1.07 (1.1)	2.32 (2.29)	1 (1)	1.18 (1.16)	1.1 (1.12)
pathways (4)	251.87 (187.08)	1 (1)	1 (1)	15.19 (10.16)	1 (1)
psr-small (48)	1.36 (1.16)	1 (1)	1 (1)	1.07 (1.03)	1.22 (1.01)
pw-notankage (16)	5.95 (3.25)	1.49 (1)	1 (1)	1.71 (1.53)	1.48 (1.11)
pw-tankange (9)	2.16 (2.03)	1.47 (1.19)	1 (1)	1.31 (1.38)	1.11 (1.02)
rovers (6)	86.2 (5.53)	1 (1)	1 (1)	5.37 (1.06)	1.42 (1.04)
satellite (7)	78.71 (30.88)	1.01 (1)	1 (1)	15.16 (2.68)	2.11 (1.05)
tpp (6)	55.39 (1)	1 (1)	1 (1)	3.42 (1)	1.35 (1)
trucks (7)	77.4 (53.6)	1.01 (1)	1 (1)	7.31 (5.11)	1.03 (1)
zenotravel (9)	24.04 (4.82)	1 (1)	1 (1)	3.61 (2.16)	2.06 (1.17)
Overall	35.41 (1.02)	8.16 (1)	**1 (1)**	3.97 (1)	1.61 (1)
Average (Domain)	55.61 (15.54)	9.76 (2.05)	**1 (1)**	5.69 (2.1)	1.97 (1.21)

Results - Time

Domain	h_{LA}	$h_{\text{LM-CUT}}$	$\max_h$	rnd_h	sel_h
airport (25)	125.96	35.36	73.80	54.78	68.44
blocks (20)	66.01	3.71	6.39	6.44	5.59
depots (7)	196.91	65.99	103.26	155.14	94.36
driverlog (14)	66.67	110.87	86.04	120.84	81.31
freecell (15)	6.04	249.28	23.93	44.22	9.25
grid (2)	12.05	33.78	44.27	38.3	40.26
gripper (6)	71.6	106.48	264.79	161.98	77.07
logistics-2000 (19)	73.32	152.27	255.36	153.89	79.17
logistics-98 (5)	18.84	24.11	29.55	28.69	24.43
miconic (140)	2.03	8.04	10.08	5.67	7.65
mprime (19)	17.52	17.9	15.68	111.48	8
mystery (12)	7.55	1.61	2.03	57.93	2.49
openstacks (7)	15.93	72.3	75.83	52.69	17.11
pathways (4)	5.38	0.08	0.14	1.15	0.18
psr-small (48)	3.55	4.05	7.92	5.73	4.87
pw-notankage (16)	48.8	71.34	71.49	73.92	59
pw-tankange (9)	211.43	173.61	189.89	172.99	130.98
rovers (6)	122.7	5.23	8.79	45.72	7.97
satellite (7)	46.22	3.47	4.51	21.95	3.58
tpp (6)	108.54	14.36	5.9	56.32	5.69
trucks (7)	238.85	11.69	16.48	39.64	15.56
zenotravel (9)	9.84	0.91	1.33	8.27	1.28
Average (Problem)	39.65	38.59	41.39	42.6	**24.53**
Average (Domain)	67.08	53.02	58.97	64.44	**33.83**

Anytime Behavior



Thank You

- Thank You