Pattern-Database Heuristics for Partially Observable Nondeterministic Planning

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Manuela Ortlieb and Robert Mattmüller

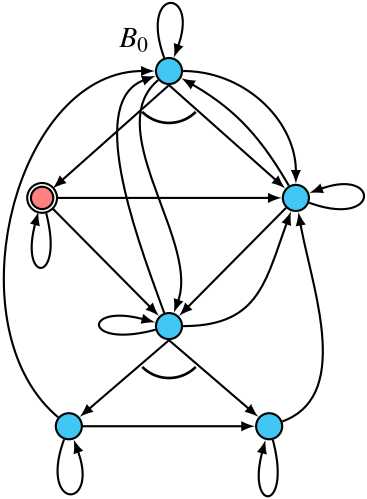
Research Group Foundations of Artificial Intelligence
Department of Computer Science
University of Freiburg, Germany

September 19th, 2013

WHAT: POND Planning

Partially observable nondeterministic (POND) planning:

- Given:
    - state variables
    - nondeterministic and sensing actions
    - inital state description
    - goal description

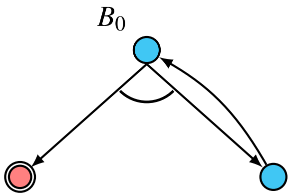


WHAT: POND Planning

Partially observable nondeterministic (POND) planning:

- Given:
  - state variables
  - nondeterministic and sensing actions
  - inital state description
  - goal description
- Wanted:
  - mapping from belief states to actions
  - to reach goal state

⇝ "strong cyclic plan"

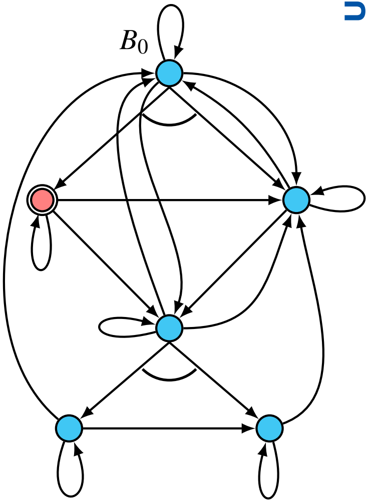


WHY: Advance Offline Planning

- Goal: Model realistic features of planning tasks like
  - nondeterminism and
  - partial observability
- Purpose:
  - Generate complete plan offline.
  - Avoid replanning during plan execution.
- Approach:
  - Do not reinvent the wheel.
  - Benefit from research on heuristics in classical planning.

Algorithmic approach:

- Progression search in belief space
- for a strong cyclic plan
- guided by distance heuristic

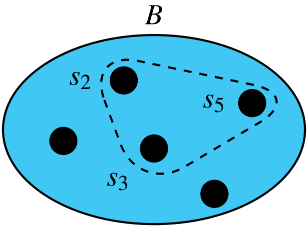


Research Question

Domain-independent distance heuristic for belief states?

Option 1: "Simplify"

1. Apply classical planning heuristic to individual world states.
2. Aggregate h-values over belief state.



⇝ Pros and Cons:

- ✓ easy to do
- ✗ sampling unclear
- ✗ aggregation unclear
- ✗ informativeness?

Aggregation:

$$h_{\mathsf{B}}(B) = h(s_2) + h(s_3) + h(s_5)$$

or

$$h_{\mathsf{B}}(B) = \max\{h(s_2), h(s_3), h(s_5)\}$$

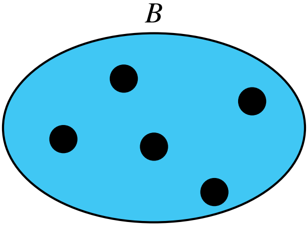
or …

Research Question

Domain-independent distance heuristic for belief states?

Option 2: “Lift”

- Lift definitions of heuristics to POND setting and define heuristic for belief states directly.



Compute $h_{\mathrm{B}}(B)$ “directly”.

$\rightsquigarrow$ Pros and Cons:

✗ less straightforward

✓ no sampling issue

✓ no aggregation issue

✓ more informative?

Research Question

Remark:

Bryce et al. (2006): "lifted" computation of $h(B)$ for RPG approach using labeled uncertainty graph (LUG).

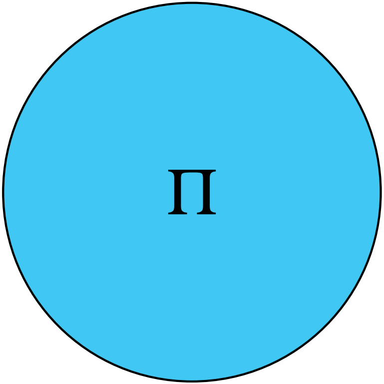
Showed superiority over a "simplifying" approach (sample, compute h^{RPG}, aggregate).

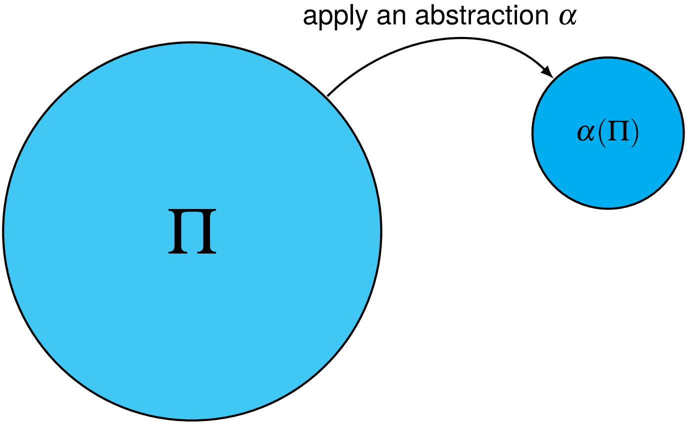
This work:

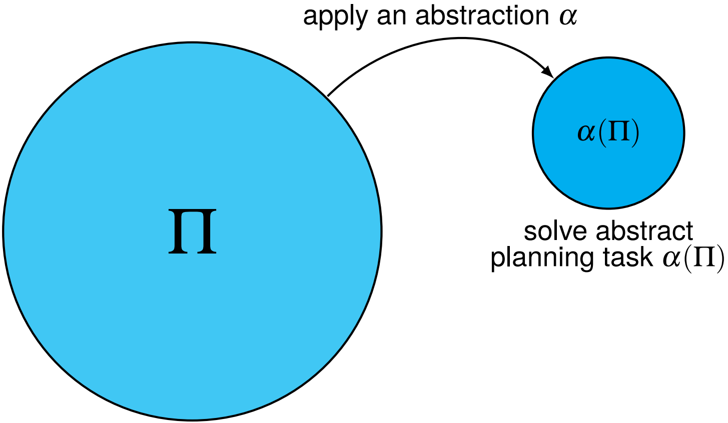
Comparison of "lifted" and "simplifying" approach for pattern-database heuristic.

	"lift"		"simplify"
RPG	LUG	$\succ$	SG
	[Bryce et al., 2006]		
PDB	?	$\succ \prec \sim$?	?
	[this work]		

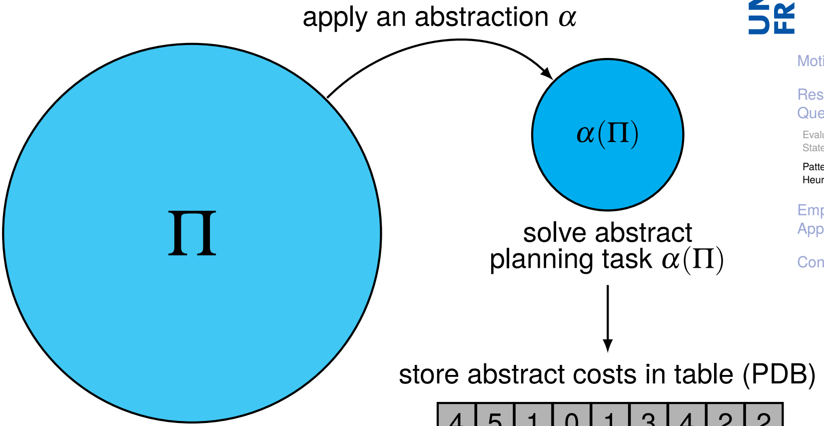
Pattern-Database Heuristics




apply an abstraction α
Π
$\alpha(\Pi)$

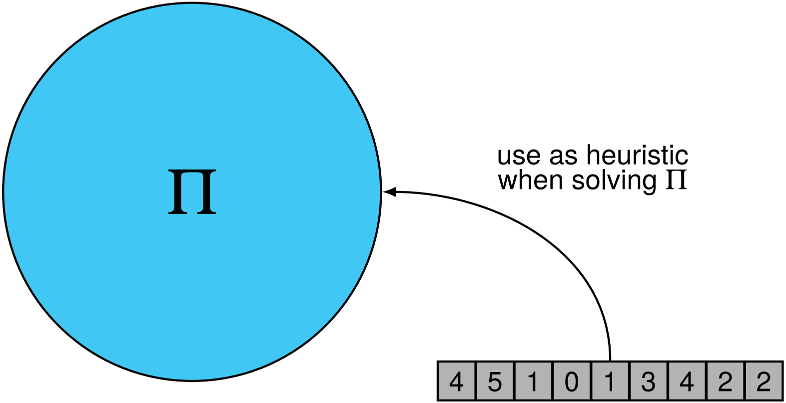

apply an abstraction α
Π
$\alpha(\Pi)$
solve abstract planning task $\alpha(\Pi)$

Pattern-Database Heuristics



4	5	1	0	1	3	4	2	2

Pattern-Database Heuristics



Pattern-Database Heuristics

Full vs. partial observability:

- **Full observability:**
  abstract state space “only” exponential in pattern size
  ⇒ larger patterns possible
  - ✓ much of the state structure taken into account
  - ✗ (un)observability not taken into account
- **Partial observability:**
  abstract state space doubly exponential in pattern size
  ⇒ only smaller patterns possible
  - ✗ less of the state structure taken into account
  - ✓ (un)observability taken into account

Question:

How to deal with this tradeoff?

Empirical Approach

Question:
In abstraction, should we assume

- full observability (option 1) or
- partial observability (option 2)?

In abstraction, should we assume

- deterministic actions (option 1) or
- nondeterministic actions (option 2)?

Way to investigate this tradeoff:
purely empirical

Empirical Approach

Implementation and comparison of three variants of POND PDB heuristic:

Variant	Observability	Determinization	Abstract problem type	Abstract goal distances	Sampling, aggregation?
FO-Det (“simplify everything”)	full	yes	classical	optimistic	yes
FO-NDet (“simplify observation”)	full	no	FOND	expected	yes
PO-NDet (“simplify nothing”)	partial	no	POND	expected	no

Implementation Details

- Strong cyclic POND planner using variant of LAO* [Hansen and Zilberstein, 2001]
- Guided by FO-Det, FO-NDet and PO-NDet PDB heuristics
- Canonical heuristic function, iPDB [Haslum et al., 2007]
- Symbolic BDD representation of belief states and transitions
- Sampling of world states from belief states uniformly with replacement
- 4GB memory limit, 30 minute time limit per run

Benchmark Domains

- FIRST-RESPONDERS adapted to requiring some active sensing
- BLOCKSWORLD adapted to requiring some active sensing
- CANADIAN-TRAVELER-PROBLEM without probabilities and with unit edge costs

Belief State Sampling and Aggregation

For FO-(N)Det:

- How many world states to sample from belief states?
  $\rightsquigarrow$ experiment with
  - 1
  - 5
  - 10
  - 15
  - “all”
- How to aggregate values?
  $\rightsquigarrow$ experiment with
  - maximization
  - summation

Belief State Sampling

Dom		FO-Det						FO-NDet					
		max			sum			max			sum		
	n	cov	exp	time	cov	exp	time	cov	exp	time	cov	exp	time
FR	1	42	13835	995	41	13835	1357	40	11084	1125	40	11084	1077
(75)	5	54	6161	291	58	3644	156	58	6599	855	60	4868	206
	10	56	12194	755	62	2716	162	55	11097	494	64	3338	117
	15	51	11267	579	62	4481	320	56	11420	631	65	4998	341
	all	54	11085	395	32	27048	1900	59	9810	309	31	12751	665
BW	1	12	3573	24	12	3573	46	14	4024	49	14	4024	76
(30)	5	14	2766	50	12	2214	34	13	2647	52	13	3261	89
	10	13	2509	34	14	1863	37	12	1699	25	12	3532	77
	15	14	1922	31	14	1796	33	12	1271	25	13	2495	60
	all	13	2392	22	14	1618	16	14	2731	61	12	3007	49
CTP	1	26	751	28	26	751	31	26	728	29	26	728	32
(46)	5	26	494	76	26	460	79	26	507	74	26	488	86
	10	26	560	154	26	428	143	26	561	147	26	391	121
	15	26	518	196	26	401	195	26	523	202	26	408	198
	all	0	—	—	0	—	—	0	—	—	0	—	—

Pattern Selection

For PO-NDet:

We experiment with three variants of pattern selection:

- Configuration "steps 0":
  - no pattern collection search
  - collection of singleton patterns for goal variables
- Configuration "pop mip0.5":
  - assume partial observability during pattern collection search
  - use minimal improvement threshold of 0.5
- Configuration "fop mip0.5":
  - assume full observability during pattern collection search
  - use minimal improvement threshold of 0.5

Pattern Selection

	PO-NDet											
Dom	steps 0				pop mip0.5				fop mip0.5			
	cov	exp	stm	ttm	cov	exp	stm	ttm	cov	exp	stm	ttm
FR	40	25278	3079	3111	70	5887	218	1058	73	5819	262	588
BW	13	6560	630	644	12	5343	423	673	12	6902	779	866
CTP	26	526	9	15	23	461	4	862	26	480	5	314
OVERALL	79	32364	3718	3770	105	11691	645	2593	111	13201	1046	1768

Internal Comparison

Comparison of best configurations of FO-Det, FO-NDet, and PO-NDet side by side to determine overall best PDB configuration.

Dom	FO-Det sum15 mip0.5				FO-NDet sum15 mip0.5				PO-NDet fop mip0.5			
	cov	exp	stm	ttm	cov	exp	stm	ttm	cov	exp	stm	ttm
FR	70	40159	9330	10320	72	28938	9140	11327	73	26414	3851	6095
BW	14	1796	33	85	13	2558	59	113	12	1670	19	78
CTP	26	607	281	849	26	607	270	1004	26	630	7	923
OVERALL	110	42562	9644	11254	111	32103	9469	12444	111	28714	3877	7096

External Comparison

Just to put PDBs in context: external comparison to FF heuristic [Hoffmann and Nebel, 2001] and blind heuristic.

Dom	blind			FF			PO-NDet fop mip0.5			
	cov	exp	stm=ttm	cov	exp	stm=ttm	cov	exp	stm	ttm
FR	16	18716	1337	47	4381	239	73	662	12	95
BW	6	15937	488	15	241	20	12	276	2	37
CTP	13	36124	2128	16	13714	735	26	152	1	88
OVERALL	35	70777	3954	78	18336	993	111	1090	16	219

Conclusion

- For PDBs: best to represent nondeterminism and partial observability in abstraction, i.e.
  - do not determinize abstract problem,
  - do not introduce full observability in abstract problem.

	"lift"		"simplify"
RPG	LUG	$\succ$	SG
	[Bryce et al., 2006]		
PDB	PO-NDet	$\succeq$	FO-(N)Det
	[this work]		

- With PDBs, even more straightforward than with LUG.