Precompiling Hierarchical Domains for Automated Planning

David Winer
School of Computing
The University of Utah
Salt Lake City, UT
drwiner@cs.utah.edu

August 11, 2017

Preliminary Information

PyDPOCL is a prototype decompositional partial-order causal-link (POCL) planner for plan-space refinement problem solving with hierarchical knowledge.

I used PyDPOCL to evaluate the efficiency gain of hierarchical planning with precompiled ground abstract steps. I evaluate using four different conditions with 8 planning problems for a single planning domain. more analyzes to come.

Plan Definitions

A **planning problem** is a tuple $\langle \Lambda_p, \Lambda_a, \Omega, I, G \rangle$ where Λ_p are primitive STRIPS-style [Fikes and Nilsson1972] operators with typed variable arguments, precondition literals, and effect literals, λ_a are abstract operators containing descriptions of partial hierarchical knowledge about how to decompose the action's operation, Ω is a set of typed objects, I is a complete state specification, and G is a set of goal conditions.

A **plan** is a tuple $\langle S, B, O, L, D \rangle$ where S is a set of **steps**, ground action instances, such that objects in Ω substitute consistent-typed argument variables, where if s is a step, then $eff(s)$ is the set of effects and $pre(s)$ is the set of preconditions of s, B is a set of **bindings** between objects and step arguments between preconditions and effects, O is a set of **ordering pairs** over S of the form $s \prec t$ where $s, t \in S$ indicating that s precedes t, where an ordering path $s \rightarrow t$ indicates a path along ordering pairs from s to t, L is a set of **causal links** over S of the form $s \xrightarrow{p} t$ indicating that condition p in $eff(s) \cap pre(t)$ is protected and that $s \prec t$, and D is a set of **decompositional links** over pairs of steps of the form $s \lhd t$ indicating when a step t is added to the plan to meet a specification of an abstract step s's subplan criteria.

Flaw Types

A **flaw** of a plan indicates a sub-goal for solving the planning problem. There are two kinds of flaws, **open conditions** and **threatened causal links**. A **candidate** for an open condition $\langle s_{need}, p \rangle$ is a step s in the plan s.t. $p \in eff(s)$ and for which there is no ordering path $s_{need} s$. A **risk** for an open condition $\langle s_{need}, p \rangle$ is a step s in the plan s.t. $\neg p \in s$ and there is no ordering path $s_{need} \rightarrow s$. A causal link $s \xrightarrow{p} t$ in a plan is threatened by a step $s_{threat} \in S$ just when there are is no ordering path $s_{threat} \rightarrow s$, no ordering path $t \rightarrow s_{threat}$, and $\neg p \in eff(s_{threat})$.

- s Static Flaw, statically True
- i Init Flaw, initially True
- t Threatened Causal Link Flaw (TCLF)
- u Unsafe Flaw, exists some risk
- r Reusable Flaw, exists some candidate

n NonReusable, does not exist some candidate

Another type of flaw is a **decomposition flaw** which indicates that an abstract step's effects are not yet satisfied by sub-steps.

Flaw Repair Strategies

An open condition of the form $\langle s_{need}, p\rangle$ in a plan is repaired by either adding a step s s.t. $p \in eff(s)$, ordering $s \prec s_{need}$, and causal link $s \xrightarrow{p} s_{need}$.

a Add Step

r Reuse Step

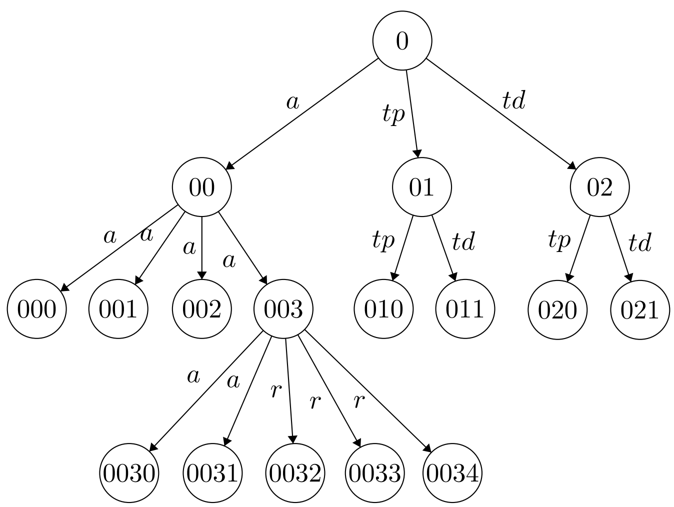
tp resolve TCLF with promotion

td resolve TCLF with demotion

Plan Trace

A **plan trace** is a sequence of $[f]x[r]$ triples where f is a flaw type, x is a number referring to the choice of repair, and r is a type of flaw repair strategy.

The sequence $[i]0[a]$ $[t]1[td]$ indicates that in this trace, an Init Flaw was selected and the flaw was repaired by adding an action (option 0), and then a TCLF was selected and repaired with demotion (option 1). Below is a tree representing a fictional search, where the last digit of the node represents the option chosen, and the edge label represents the flaw repair strategy.



Decomposition

If an abstract step is added to the plan, then its subplan is .. [borrow terminology from Domain Recipe theory]

References

[Fikes and Nilsson1972] Fikes, R. E., and Nilsson, N. J. 1972. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2(3):189–208.

Table 1: Evaluation of travel domain with 8 different planning problems.
PO = primitive only
DA1 = decomp, add 1 for all (including substeps)
DJA1 = decomp, just add 1 (not for substeps)
Dinc = decomp, incentivizing heuristic (abstracts cost less, no cost for substeps)
$Plan.cost+ = max\_height^2 + 1 - action\_height$
RT = Runtime
Expanded = Nodes expanded
Visited = Nodes visited
BT = Branches Terminated
Max Step = Highest decomposition level of abstract step
TO = Timed out (5 minutes).

		p 1	p 2	p 3	p 4	p 5	p 6	p 7	p 8
	ground steps, PO	15	28	28	28	52	52	21	166
	ground steps, D	63	124	124	124	268	268	93	934
RT	PO	**0.09**	**0.11**	**0.12**	**1.24**	**3.15**	**0.18**	**0.09**	**5.89**
	DA1	0.31	0.35	0.22	4.23	11.1	0.54	0.27	12.02
	DJA1	0.82	0.76	0.81	3.36	10.37	1.03	0.27	15
	Dinc	1.12	1.17	1.36	*TO*	*TO*	1.6	1.62	*TO*
Expanded	PO	12	12	11	73	141	11	9	115
	DA1	38	13	12	138	162	11	**8**	121
	DJA1	12	**8**	7	**66**	**111**	9	**8**	79
	Dinc	**8**	**8**	**8**	NA	NA	**8**	**8**	NA
Visited	PO	25	30	28	173	477	40	22	963
	DA1	49	47	37	420	1044	76	44	1366
	DJA1	71	57	55	309	752	83	44	933
	Dinc	64	66	66	NA	NA	92	91	NA
BT	PO	0	0	2	41	69	2	0	31
	DA1	0	1	0	48	76	2	0	31
	DJA1	18	0	0	27	64	0	0	17
	Dinc	0	0	0	NA	NA	0	0	NA
Max Step	PO	0	0	0	0	0	0	0	0
	DA1	0	0	0	0	0	0	0	0
	DJA1	1	1	1	1	1	1	0	1
	Dinc	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**

Table 2: Travel 1 problem, generate 8 plans per condition

nth plan	data type	PO	DA1	DAJ1	Dinc
1	runtime	0.06	0.31	0.76	1.14
1	nodes expanded	8	12	9	8
1	nodes visited	18	49	57	64
1	branches terminated	0	0	0	0
1	max step level	0	0	1	2
2	runtime	0.17	0.53	1.91	1.78
2	nodes expanded	16	19	18	13
2	nodes visited	33	71	102	94
2	branches terminated	4	4	18	0
2	max step level	0	0	1	2
3	runtime	0.65	2.21	3.27	2.69
3	nodes expanded	41	60	29	20
3	nodes visited	89	226	170	147
3	branches terminated	20	25	36	0
3	max step level	0	0	1	2
4	runtime	0.80	2.51	4.14	3.22
4	nodes expanded	44	63	37	24
4	nodes visited	89	234	213	175
4	branches terminated	35	40	48	0
4	max step level	0	0	1	2
5	runtime	1.28	3.56	5.31	46.86
5	nodes expanded	67	80	57	189
5	nodes visited	135	310	287	865
5	branches terminated	56	58	68	312
5	max step level	0	0	0	1
6	runtime	1.59	5.04	63.83	47.17
6	nodes expanded	80	114	427	193
6	nodes visited	162	432	1721	884
6	branches terminated	67	79	904	312
6	max step level	0	0	1	1
7	runtime	2.68	5.08	63.87	47.72
7	nodes expanded	112	116	429	199
7	nodes visited	224	436	1725	926
7	branches terminated	120	80	905	312
7	max step level	0	1	0	1
8	runtime	2.95	5.16	89.31	47.96
8	nodes expanded	121	118	531	202
8	nodes visited	241	440	2195	943
8	branches terminated	132	82	1262	312
8	max step level	0	0	2	1

Table 3: Plan Traces of solutions with different approaches

c	n	Travel 1 - Plan trace of solution of n'th plan
PO	1	[n]1[a] [n]0[a] [i]1[r] [i]2[r] [n]0[a] [i]1[r] [t]0[tp] [u]2[r]
PO	2	[n]0[a] [n]1[a] [i]1[r] [i]2[r] [u]0[a] [n]0[a] [i]1[r] [t]0[tp] [u]2[r]
PO	3	[n]1[a] [n]0[a] [i]1[r] [i]2[r] [n]0[a] [i]0[a] [t]1[td] [t]0[tp] [u]2[r] [u]2[r] [r]0[a] [i]1[r] [t]0[tp] [t]1[td] [u]2[r]
PO	4	[n]1[a] [n]0[a] [i]1[r] [i]1[a] [i]1[r] [t]1[td] [u]0[a] [i]1[r] [i]2[r] [t]1[td] [n]0[a] [i]1[r] [t]0[tp] [t]0[tp] [t]0[tp] [u]2[r]
PO	5	[n]1[a] [n]0[a] [i]1[r] [i]0[a] [i]1[r] [n]0[a] [i]1[r] [t]0[tp] [u]2[r] [n]1[a] [i]1[r] [i]2[r] [t]1[td] [u]0[a]
PO	6	[n]1[a] [n]1[a] [t]1[td] [r]1[a] [t]1[td] [u]0[a] [i]1[r] [i]2[r] [n]0[a] [i]1[r] [t]0[tp] [u]3[r] [t]0[tp] [n]1[r] [n]1[r]
PO	7	[n]0[a] [n]1[a] [i]1[r] [i]1[a] [i]1[r] [u]0[a] [n]0[a] [i]1[r] [i]2[r] [t]1[td] [n]0[a] [i]1[r] [t]0[tp] [u]2[r]
PO	8	[n]0[a] [n]1[a] [i]1[r] [i]2[r] [u]0[a] [n]0[a] [i]0[a] [t]1[td] [t]0[tp] [u]2[r] [u]2[r] [r]0[a] [i]1[r] [t]0[tp] [t]1[td] [u]2[r]
DA1	1	[n]1[a] [n]1[a] [i]1[r] [i]10[r] [n]0[a] [i]1[r] [t]0[tp] [u]2[r]
DA1	2	[n]3[a] [n]0[a] [i]1[r] [i]10[r] [u]0[a] [n]0[a] [i]1[r] [t]0[tp] [u]2[r]
DA1	3	[n]1[a] [n]1[a] [i]1[r] [i]10[r] [n]0[a] [i]0[a] [t]1[td] [t]0[tp] [u]2[r] [u]2[r] [r]0[a] [i]1[r] [t]1[td] [t]0[tp] [u]2[r]
DA1	4	[n]1[a] [n]1[a] [i]1[r] [i]2[a] [i]1[r] [t]1[td] [u]1[a] [i]1[r] [i]10[r] [t]1[td] [n]0[a] [i]1[r] [t]0[tp] [t]0[tp] [t]0[tp] [u]2[r]
DA1	5	[n]1[a] [n]1[a] [i]1[r] [i]4[a] [i]1[r] [n]0[a] [i]1[r] [t]0[tp] [u]2[r] [n]0[a] [i]1[r] [i]10[r] [t]1[td] [u]0[a]
DA1	6	[n]1[a] [n]0[a] [t]1[td] [r]1[a] [t]1[td] [u]1[a] [i]1[r] [i]10[r] [n]0[a] [i]1[r] [t]0[tp] [u]3[r] [t]0[tp] [n]1[r] [n]1[r]
DA1	7	[n]9[a] [n]1[a] [i]1[r] [i]10[r] [n]0[a] [i]1[r] [t]0[tp] [u]2[r] [n]10[r]
DA1	8	[n]5[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r]
DJA1	1	[n]7[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r]
DJA1	2	[n]7[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r]
DJA1	3	[n]5[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r]
DJA1	4	[n]5[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r]
DJA1	5	[n]1[a] [n]1[a] [i]1[r] [i]10[r] [n]0[a] [i]1[r] [t]0[tp] [u]2[r]
DJA1	6	[n]4[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
DJA1	7	[n]4[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
DJA1	8	[n]2[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
Dinc	1	[n]4[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
Dinc	2	[n]4[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
Dinc	3	[n]2[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
Dinc	4	[n]2[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r] [n]10[r]
Dinc	5	[n]7[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r]
Dinc	6	[n]7[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r]
Dinc	7	[n]5[a] [i]10[r] [i]1[r] [i]10[r] [i]1[r] [u]4[r] [n]10[r]
Dinc	8	[n]5[a] [i]10[r] [i]1[r] [i]11[r] [i]1[r] [u]4[r] [n]10[r]