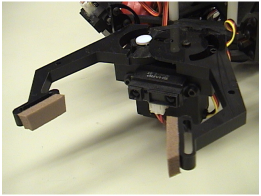
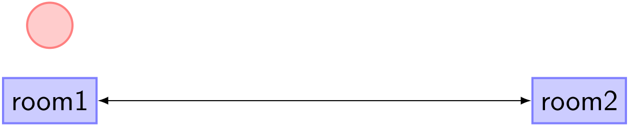
Fluent Merging for Classical Planning Problems

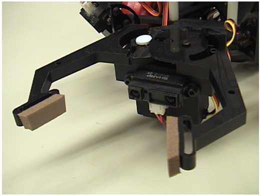
Jendrik Seipp Malte Helmert

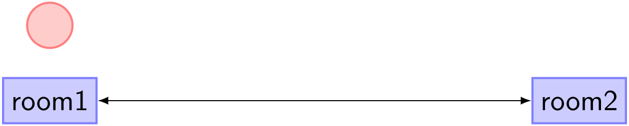
Albert-Ludwigs-Universität Freiburg, Germany

ICAPS 2011 Workshop on
Knowledge Engineering for Planning and Scheduling
June 12th, 2011

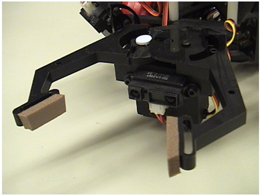


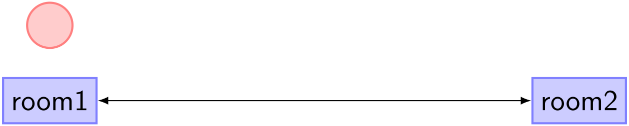

room1
room2





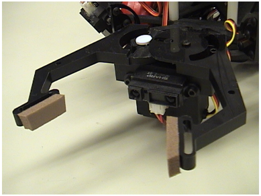
- in(ball, room1) $\in$ {True, False}
- in(ball, room2) $\in$ {True, False}
- carry(ball, arm) $\in$ {True, False}

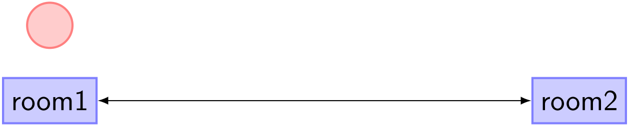




- in(ball, room1) $\in$ {True, False}
- in(ball, room2) $\in$ {True, False}
- carry(ball, arm) $\in$ {True, False}

- ball_pos $\in$ {in(ball, room1), in(ball, room2), carry(ball, arm)}





- in(ball, room1) $\in$ {True, False}
- in(ball, room2) $\in$ {True, False}
- carry(ball, arm) $\in$ {True, False}

- ball_pos $\in$ {in(ball, room1), in(ball, room2), carry(ball, arm)}
- robby_pos $\in$ {robby-in(room1), robby-in(room2)}
- state_arm $\in$ {free(arm), full(arm)}

Background

- Paper by van den Briel, Kambhampati and Vossen at ICAPS 2007 Heuristics workshop
- Mutex groups in Fast Downward

Fast Downward Planning System

- Translation
- Knowledge compilation
- Search

Fast Downward Planning System

- Translation
- Fluent Merging
- Knowledge compilation
- Search

Merging two variables

Definition (SAS$^+$ planning task)

$$\Pi = \langle \mathcal{V}, \mathcal{O}, s_0, s_\star \rangle$$

- Merging also generalized for conditional effects.

Variables

- $\mathcal{V} = \{\text{ball\_pos, robby\_pos, arm}\}$
  - $\mathcal{D}_{\text{ball\_pos}} = \{\text{in(ball, room1)}, \text{in(ball, room2)}, \text{carry(ball, arm)}\}$
  - $\mathcal{D}_{\text{robby\_pos}} = \{\text{robby-in(room1)}, \text{robby-in(room2)}\}$
  - $\mathcal{D}_{\text{state\_arm}} = \{\text{free(arm)}, \text{full(arm)}\}$

Variables

- $\mathcal{V} = \{\text{ball\_pos, robby\_pos, arm}\}$
  - $\mathcal{D}_{\text{ball\_pos}} = \{\text{in(ball, room1)}, \text{in(ball, room2)}, \text{carry(ball, arm)}\}$
  - $\mathcal{D}_{\text{robby\_pos}} = \{\text{robby-in(room1)}, \text{robby-in(room2)}\}$
  - $\mathcal{D}_{\text{state\_arm}} = \{\text{free(arm)}, \text{full(arm)}\}$
- New variable: $\text{ball\_pos} \otimes \text{state\_arm}$

$\mathcal{D}_{\text{ball\_pos} \otimes \text{state\_arm}}$

$\text{in(ball, room1)} \otimes \text{free(arm)}$	$\text{in(ball, room1)} \otimes \text{full(arm)}$
$\text{in(ball, room2)} \otimes \text{free(arm)}$	$\text{in(ball, room2)} \otimes \text{full(arm)}$
$\text{carry(ball, arm)} \otimes \text{free(arm)}$	$\text{carry(ball, arm)} \otimes \text{full(arm)}$

Variables

- $\mathcal{V} = \{\text{ball\_pos, robby\_pos, arm}\}$
  - $\mathcal{D}_{\text{ball\_pos}} = \{\text{in(ball, room1)}, \text{in(ball, room2)}, \text{carry(ball, arm)}\}$
  - $\mathcal{D}_{\text{robby\_pos}} = \{\text{robby-in(room1)}, \text{robby-in(room2)}\}$
  - $\mathcal{D}_{\text{state\_arm}} = \{\text{free(arm)}, \text{full(arm)}\}$
- New variable: $\text{ball\_pos} \otimes \text{state\_arm}$

$\mathcal{D}_{\text{ball\_pos} \otimes \text{state\_arm}}$

$\text{in(ball, room1)} \otimes \text{free(arm)}$	$\text{in(ball, room1)} \otimes \text{full(arm)}$
$\text{in(ball, room2)} \otimes \text{free(arm)}$	$\text{in(ball, room2)} \otimes \text{full(arm)}$
	$\text{carry(ball, arm)} \otimes \text{full(arm)}$

Variables

- $\mathcal{V} = \{\text{ball\_pos, robby\_pos, arm}\}$
  - $\mathcal{D}_{\text{ball\_pos}} = \{\text{in(ball, room1)}, \text{in(ball, room2)}, \text{carry(ball, arm)}\}$
  - $\mathcal{D}_{\text{robby\_pos}} = \{\text{robby-in(room1)}, \text{robby-in(room2)}\}$
  - $\mathcal{D}_{\text{state\_arm}} = \{\text{free(arm)}, \text{full(arm)}\}$
- New variable: $\text{ball\_pos} \otimes \text{state\_arm}$

$\mathcal{D}_{\text{ball\_pos}\otimes\text{state\_arm}}$

$\text{in(ball, room1)} \otimes \text{free(arm)}$
$\text{in(ball, room2)} \otimes \text{free(arm)}$
$\text{carry(ball, arm)} \otimes \text{full(arm)}$

Operator 1

- move-room1-room2 = $\langle\{\text{robby-in(room1)}\},$ $\{\text{robby-in(room2)}\}\rangle$

Operator 2

- pick-ball-in-room1 = $\langle\{$robby-in(room1), in(ball, room1), free(arm)$\}$, $\{$carry(ball, arm), full(arm)$\}\rangle$

Operator 2

- pick-ball-in-room1 = $\langle\{\text{robby-in(room1)}, \text{in(ball, room1)} \otimes \text{free(arm)}\},$
  $\{\text{carry(ball, arm)} \otimes \text{full(arm)}\}\rangle$

Operator 3

- drop-ball-in-room1 = $\langle\{$robby-in(room1), carry(ball, arm)$\}$, $\{$in(ball, room1), free(arm)$\}\rangle$

Operator 3

- drop-ball-in-room1 $=$ $\langle\{\text{robby-in(room1)}, \text{carry(ball, arm)}\}, \{\text{in(ball, room1)} \otimes \text{free(arm)}\}\rangle$

Operator 3

- drop-ball-in-room1-with-full(arm) $=$
  $\langle\{$robby-in(room1), carry(ball, arm) $\otimes$ full(arm)$\}$,
  $\{$in(ball, room1) $\otimes$ free(arm)$\}\rangle$

Initial state

- $s_0 = \text{robby-in(room1)} \wedge \text{in(ball, room1)} \wedge \text{free(arm)}$

Initial state

- $s_0 = \text{robby-in(room1)} \wedge \text{in(ball, room1)} \otimes \text{free(arm)}$

Goal

- $s_\star = \text{in(ball, room2)}$
- $\text{in(ball, room2)} \otimes \text{free(arm)} \rightarrow \text{in(ball, room2)}$?

Goal

- $s_\star = \text{in}(\text{ball}, \text{room2})$
- $\text{in}(\text{ball}, \text{room2}) \otimes \text{free}(\text{arm}) \rightarrow \text{in}(\text{ball}, \text{room2})$?
- $\text{pseudo-op} = \langle \{\text{in}(\text{ball}, \text{room2}) \otimes \text{free}(\text{arm})\}, \{\text{in}(\text{ball}, \text{room2})\} \rangle$
- $\mathcal{D}_{\text{ball\_pos} \otimes \text{state\_arm}} \leftarrow \mathcal{D}_{\text{ball\_pos} \otimes \text{state\_arm}} \cup \{\text{in}(\text{ball}, \text{room2})\}$

Why is Fluent Merging interesting for KEPS?

- Fluent Merging as an attempt to show that the underlying representation is not set in stone

Fluent Selection

- Random variables
- Number of mutexes
- Minimize total domain size
- Heavily connected variables in causal graph
- Two-cycle pairs in causal graph
- Goal variables
- Minimize number of operators

Experiments - Settings

- 5 merges, only variable pairs
- Worse performance with bigger values
- 30 minutes, 2 GB memory
- Greedy best-first search with deferred evaluation and h^{cea} (Helmert and Geffner, 2008)

Experiments - Results

Domain	**no-merge**	**rand**	**mutex**	**size**	**conn**	**cycles**	**goals**	**ops**
depot (22)	17	11	14	12	**15**	**15**	13	14
freecell (80)	78	75	**77**	76	72	72	57	37
pathways (30)	15	14	<u>**16**</u>	<u>**17**</u>	14	14	13	15
pipes-nt (50)	38	5	8	**16**	14	14	9	**16**
pipes-t (50)	24	9	3	**17**	11	8	9	15
rovers (40)	34	31	34	<u>**35**</u>	34	34	34	24
schedule (150)	60	58	59	59	54	52	39	**60**
tpp (30)	28	20	**24**	**24**	22	**24**	23	16
trucks (30)	17	15	14	**16**	14	14	**16**	6
...	...				...			
Total (880)	709	616	625	**660**	619	608	583	548

- Each method best in at least one domain
- No method comes close to reference

Same object method

- First-order PDDL representation
- Examples:
  - $\mathcal{D}_v = \{\text{painted}(\text{chair1}), \text{not-painted}(\text{chair1})\}$

Same object method

- First-order PDDL representation
- Examples:
  - $\mathcal{D}_v = \{\text{painted}(\text{chair1}), \text{not-painted}(\text{chair1})\}$

Same object method

- First-order PDDL representation
- Examples:
  - $\mathcal{D}_v = \{\text{painted(chair1)}, \text{not-painted(chair1)}\}$
  - $\mathcal{D}_u = \{\text{at(c2 loc1)}, \text{at(c2 loc2)}, \text{at(c2 loc3)}\}$

Same object method

- First-order PDDL representation
- Examples:
  - $\mathcal{D}_v = \{\text{painted}(\text{chair1}), \text{not-painted}(\text{chair1})\}$
  - $\mathcal{D}_u = \{\text{at}(\text{c2 loc1}), \text{at}(\text{c2 loc2}), \text{at}(\text{c2 loc3})\}$

Same object method

- First-order PDDL representation
- Examples:
  - $\mathcal{D}_v = \{\text{painted(chair1)}, \text{not-painted(chair1)}\}$
  - $\mathcal{D}_u = \{\text{at(c2 loc1)}, \text{at(c2 loc2)}, \text{at(c2 loc3)}\}$
- Merge only variables that speak about the same object

Experiments - Settings

- Discouraging results with optimal configurations
- Greedy best-first search with deferred evaluation and
  - h^{cea}: Context-enhanced additive heuristic (Helmert and Geffner, 2008)
  - h^{CG}: Causal graph heuristic (Helmert 2004)
  - h^{FF}: FF/additive heuristic (Hoffmann and Nebel, 2001)

Experiments - Results

Domain	Merges h^{FF} 0	2	5	10	15	20	30
depot (22)	19	18	19	**20**	**20**	**20**	**20**
freecell (80)	76	**80**	78	77	79	78	75
miconic (150)	**150**	**150**	**150**	**150**	**150**	80	80
pprinter (30)	**23**	22	22	22	22	22	22
pipes-nt (50)	**43**	41	42	42	**43**	42	42
pipes-t (50)	38	**39**	38	37	**39**	37	37
rovers (40)	**40**	**40**	**40**	**40**	**40**	**40**	37
schedule (150)	**150**	149	149	149	149	149	148
sokoban-sat (30)	24	28	**29**	28	28	28	28
storage (30)	**20**	**20**	**20**	**20**	19	19	19
trucks (30)	**19**	17	17	18	18	18	18
wood-sat (30)	**29**	**29**	28	28	28	28	**29**
...	...						
Total (908)	820	822	821	820	**824**	750	744

Mutex threshold

- Suggested by reviewer
- $\frac{|\mathcal{D}_{a \otimes b}|}{|\mathcal{D}_a| \cdot |\mathcal{D}_b|} < x$?
- For gripper example: $\frac{(3 \cdot 2) - 3}{3 \cdot 2} = 0.5$

Mutex threshold - Experiments

		70%			80%			90%		
# Merges →	0	2	5	10	2	5	10	2	5	10
depot (22)	17	18	17	18	19	17	18	**20**	18	19
freecell (80)	**76**	**76**	**76**	75	**76**	**76**	75	**76**	**76**	**76**
trucks-strips (30)	18	**21**	17	17	**21**	17	17	**21**	17	17
Total (132)	111	115	110	110	116	110	110	**117**	111	112

Table: Greedy best-first search with deferred evaluation and h^{FF}

- Other domains: No mutexes or no change compared to h^{FF}

Future Work

- Inspect impact on heuristics in detail
- Fluent merging with boolean fluents
- Use automatic parameter configuration methods

Summary

- First general implementation and experimental evaluation
- Improvements in some domains
- Find out which and how many fluents to merge