Solving Multiagent Planning Problems with Concurrent Conditional Effects

Daniel Furelos-Blanco[1] and Anders Jonsson[2]

[1]Imperial College London [2]Universitat Pompeu Fabra

Motivation

What is concurrent multiagent planning?

- Agents collaborate to solve a problem.
- Collaboration = joint actions executed by multiple agents at once.

What is the challenge?

- The number of joint actions is worst-case exponential in the number of agents.
- Few planners handle concurrency.

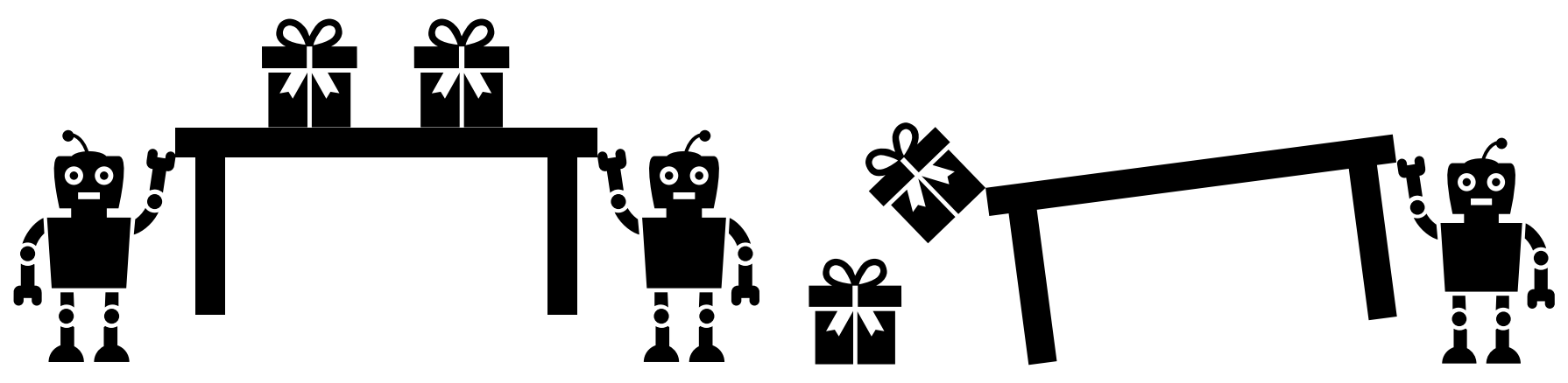

Figure 1: The TableMover [1] domain: if only one agent lifts the table, the blocks fall.

Proposed Approach

Solve multiagent planning problems involving concurrency by translating them into **classical planning**.

Planning Formalisms

A **classical planning** problem is defined as

$$\Pi = \langle F, A, I, G \rangle$$

where:

- F is a set of fluents,
- A is a set of atomic actions,
- $I \subseteq F$ is an initial state, and $G \subseteq F$ is a goal condition.

A **multiagent planning** (MAP) problem is defined as

$$\Pi = \langle N, F, \{A^i\}_{i \in N}, I, G \rangle$$

where $N = \{1, \ldots, N\}$ is the agent set, and A^i is the action set of agent $i \in N$.

Concurrency Constraints

- Formulations in [1, 2] use actions as **fluents**.
  - **Positive concurrency:** action a^1 has a^2 as precondition (must be done together).
  - **Negative concurrency:** action a^1 has $\neg a^2$ as precondition (cannot be done together).
- **Effects** of an action a^1 can be **conditioned** to the simultaneous execution of another action a^2.
- Each agent contributes **at most once** to the joint action.

```
(:action lift-side
 :agent ?a - agent
 :parameters (?s - side)
 :precondition (and (at-side ?a ?s)
   (down ?s) (handempty ?a)
   (forall (?a2 - agent ?s2 - side)
     (not(lower-side ?a2 ?s2)))
 )
 :effect (and (not (down ?s))
   (up ?s) (lifting ?a ?s)
   (not (handempty ?a ?s))
...
```

```
...
(forall
  (?b - block ?r - room ?s2 - side)
  (when
    (and (inroom Table ?r)
      (on-table ?b) (down ?s2)
      (forall (?a2 - agent)
        (not (lift-side ?a2 ?s2))))
    (and (on-floor ?b) (inroom ?b ?r)
      (not (on-table ?b))))))
)
```

Figure 2: TableMover's `lift-side` action using Kovacs notation (concurrency constraints).

Compilation

Divide simulation of a joint action in three different phases:

- **Action selection:** check preconditions of constituent atomic actions.
- **Action application:** apply effects of constituent atomic actions.
- **Resetting:** reset auxiliary fluents.

The resulting number of actions is **polynomial**, not exponential:

$$|A'| = 3\sum_{i \in N} |A^i| + 4.$$

Extension

Joint actions with bounded size C:

- At most C agents can act at a time.
- Purpose: reduce branching factor.
- The number of actions is still polynomial:

$$|A'| = (2C + 1) \sum_{i \in N} |A^i| + 4.$$

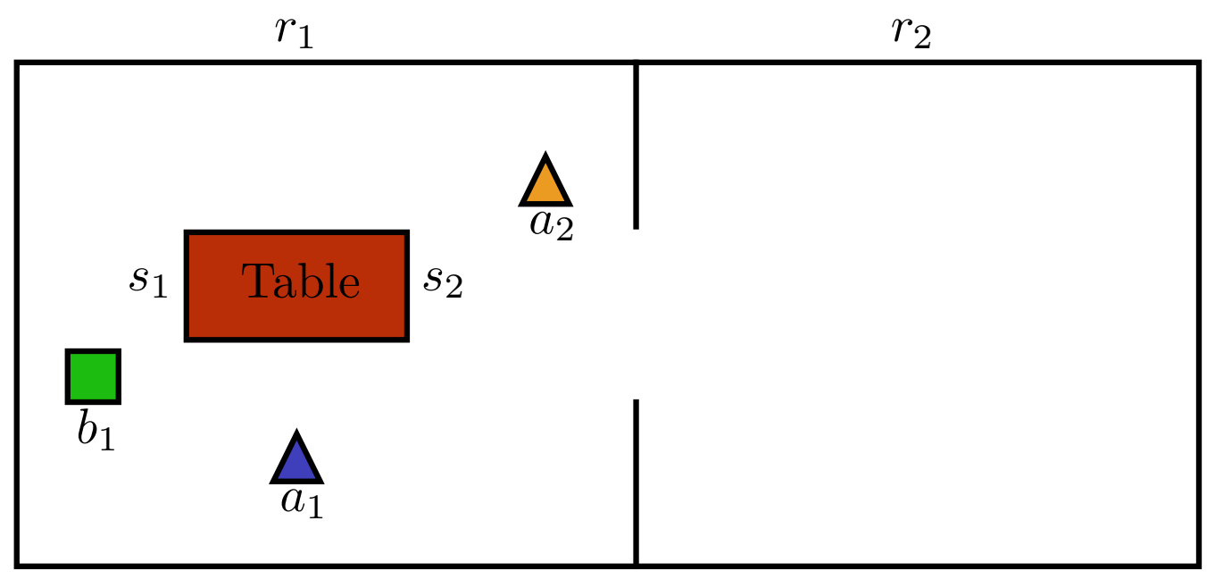

Figure 3: TableMover instance example.

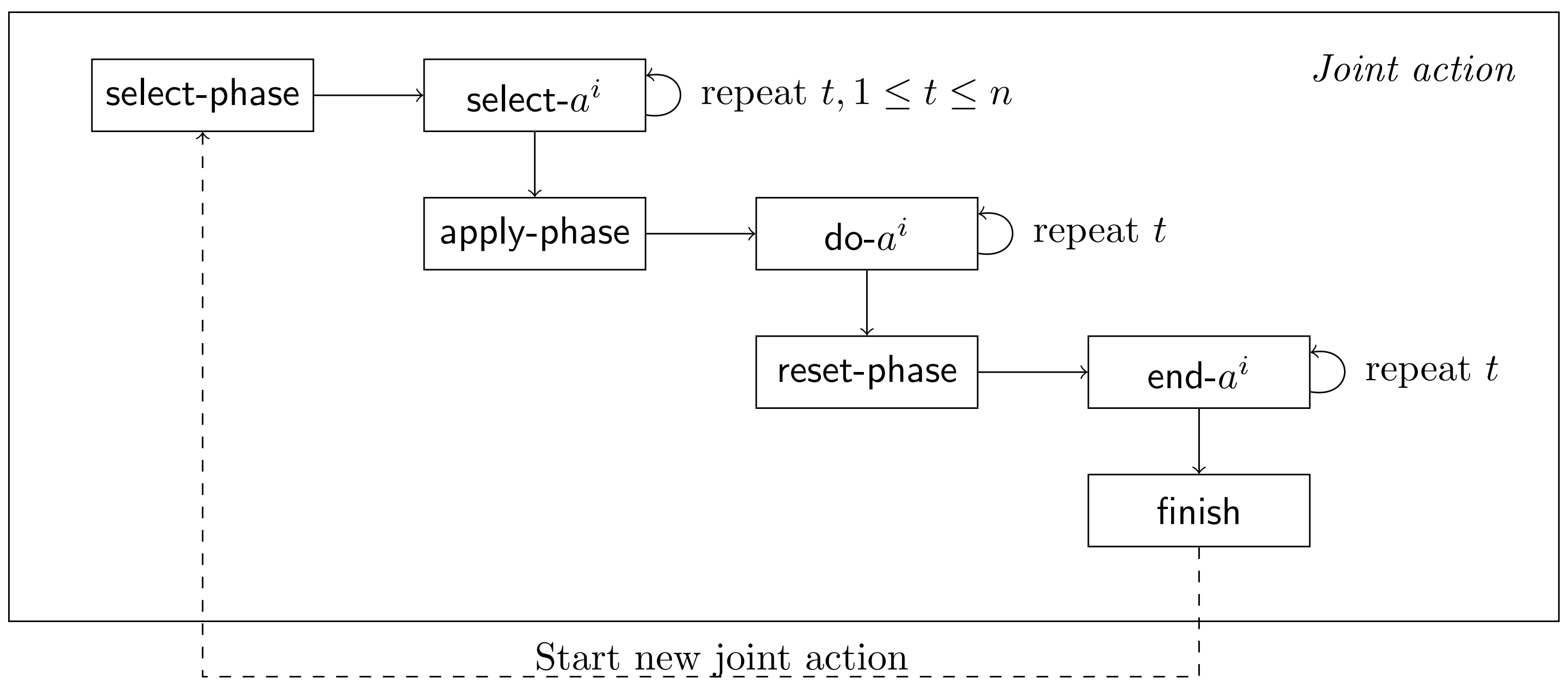

Figure 4: Compilation of each multiagent action into a classical action.

Multiagent plan

```
1 (to-table a1 r1 s2)(pickup-floor a2 b1 r1)
2 (putdown-table a2 b1 r1)
3 (to-table a2 r1 s1)
4 (lift-side a1 s2)(lift-side a2 s1)
5 (move-table a1 r1 r2 s2)(move-table a2 r1 r2 s1)
6 (lower-side a1 s2)
```

Classical plan (1st joint action)

```
 1 (select-phase )
 2 (select-to-table a1 r1 s2)
 3 (select-pickup-floor a2 b1 r1)
 4 (apply-phase )
 5 (do-pickup-floor a2 b1 r1)
 6 (do-to-table a1 r1 s2)
 7 (reset-phase )
 8 (end-to-table a1 r1 s2)
 9 (end-pickup-floor a2 b1 r1)
10 (finish )
```

Figure 5: How a multiagent plan is represented using our approach for problem in Figure 3.

Results

Domain	N	Coverage					Time (s.)					Makespan					# Grounded actions ($\times 10^3$)				
		2	4	∞	CJR	SB	2	4	∞	CJR	SB	2	4	∞	CJR	SB	2	4	∞	CJR	SB
Maze	20	**13**	8	6	11	9	361.5	444.2	**145.6**	195.1	216.1	47.2	22.0	**11.7**	77.3	67.7	41.7	69.3	**27.9**	156.8	108.2
$a = 10$	10	**8**	6	5	7	6	250.2	575.6	**170.4**	228.4	323.1	48.3	25.0	**12.2**	79.6	69.8	39.9	67.4	**26.1**	119.3	102.1
$a = 15$	10	**5**	2	1	4	3	**539.5**	-	-	-	-	**45.4**	-	-	-	-	43.9	71.8	**30.0**	194.3	115.1
BoxPushing	20	9	15	16	-	**18**	**5.2**	36.4	143.3	-	305.8	**11.2**	11.3	12.9	-	20.5	3.5	5.7	2.5	-	**2.0**
$a = 2$	10	9	9	9	-	**10**	**5.2**	7.6	6.0	-	158.9	**11.2**	11.9	11.3	-	18.4	1.8	3.2	**1.1**	-	1.2
$a = 4$	10	0	6	7	-	**8**	-	**79.7**	319.9	-	489.5	-	**10.5**	15	-	23.1	5.2	8.2	3.8	-	**2.9**
TableMover	24	**15**	12	**15**	-	-	**263.4**	336.7	341.1	-	-	**58.7**	59.0	61.5	-	-	7.4	13.1	**4.6**	-	-
$a = 2$	12	10	10	**11**	-	-	**103.9**	226.6	214.7	-	-	63.5	**62.0**	64.5	-	-	3.4	6.1	**2.0**	-	-
$a = 4$	12	**5**	2	4	-	-	**582.4**	-	-	-	-	**49.0**	-	-	-	-	11.5	20.1	**7.2**	-	-
Workshop	20	**15**	13	13	-	-	134.3	301.4	**52.5**	-	-	35.7	37.0	**32.5**	-	-	18.0	31.0	**11.5**	-	-
$a = 4$	10	**8**	**8**	**8**	-	-	42.8	263.3	**37.1**	-	-	**37.3**	43.9	**37.3**	-	-	7.7	13.6	**4.8**	-	-
$a = 8$	10	**7**	5	5	-	-	238.8	362.3	**77.1**	-	-	33.9	26.0	**24.8**	-	-	28.2	48.3	**18.1**	-	-

- Unbounded compilation (∞) has the highest coverage.
- Compilation $C = 2$ is fast but cannot solve problems involving > 2 agents.
- Our approach can solve a wider range of problems than CJR [3] and SB [4].

Scalability

#Agents	# Grounded actions		Time (s.)	
	Naive	∞	Naive	∞
2	48	100	0.1	0.2
4	992	260	0.5	0.2
6	31248	484	53.9	0.4
8	-	772	-	0.5
10	-	1124	-	0.8
50	-	21604	-	42.0
100	-	83204	-	289.9

Table 1: Scalability comparison of our compilation with the naive one in the Maze domain.

Conclusions

- Sound and complete method for compiling MAPs into classical planning problems.
- The number of resulting actions is polynomial in the MAP description.
- Handles concurrency constraints including conditional effects.
- Solves problems out of reach of previous approaches.

References

[1] Craig Boutilier and Ronen I. Brafman.
Partial-Order Planning with Concurrent Interacting Actions.
J. Artif. Intell. Res. (JAIR), 14, 2001.

[2] Daniel L Kovacs.
A Multi-Agent Extension of PDDL3.1.
In *Proc. Workshop on the International Planning Competition*, 2012.

[3] Matthew Crosby, Anders Jonsson, and Michael Rovatsos.
A Single-Agent Approach to Multiagent Planning.
In *Proc. ECAI 2014*.

[4] Shashank Shekhar and Ronen I. Brafman.
Representing and Planning with Interacting Actions and Privacy.
In *Proc. ICAPS 2018*.

Acknowledgements

This work has been supported by the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502). Anders Jonsson is partially supported by the grants TIN2015-67959 and PCIN-2017-082 of the Spanish Ministry of Science.

Contact Information

- Software: https://github.com/aig-upf/universal-pddl-parser-multiagent
- Email: d.furelos-blanco18@imperial.ac.uk