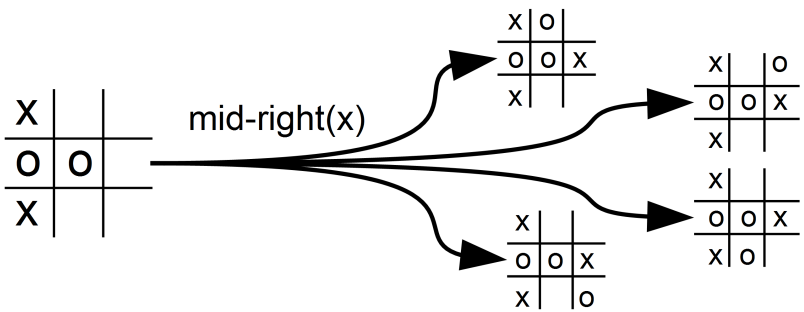
Leveraging FOND Planning Technology to Solve Multi-Agent Planning Problems

Christian Muise, Paolo Felli, Tim Miller,
Adrian R. Pearce, Liz Sonenberg

Department of Computing and Information Systems, University of Melbourne
{christian.muise,paolo.felli,tmiller,adrianrp,l.sonenberg}@unimelb.edu.au

Multi-agent Planning as FOND: Key Points



- Perspectival view on multi-agent planning
- Leverage the power of modern FOND planning
- Seamlessly handle a mix of agnostic, collaborative, and combative agents

Outline

Outline

FOND Planning

Planning Task: $\langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$

- $\mathcal{F}$: Finite set of fluents.
- $\mathcal{G}$: Conjunction of atoms over $\mathcal{F}$ that determines the goal.
- $\mathcal{I}$: Complete setting of the fluents for the initial state.
- $\mathcal{A}$: Set of (possibly non-deterministic) actions.

FOND Planning

Planning Task: $\langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$

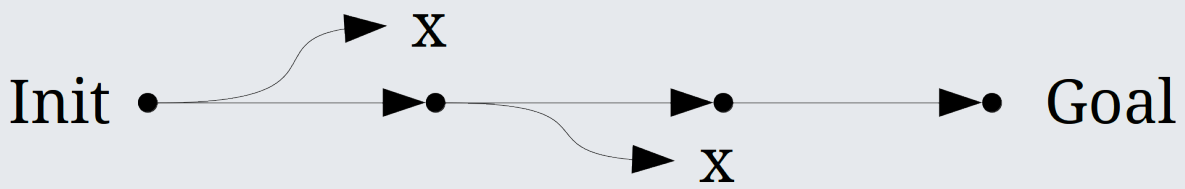
- $\mathcal{F}$: Finite set of fluents.
- $\mathcal{G}$: Conjunction of atoms over $\mathcal{F}$ that determines the goal.
- $\mathcal{I}$: Complete setting of the fluents for the initial state.
- $\mathcal{A}$: Set of (possibly non-deterministic) actions.

Actions

- Pre_a is the set of fluents that must be true to execute a.
- Eff_a is a finite set of *non-deterministic effects*.
- A non-deterministic effect consists of a set of fluents.

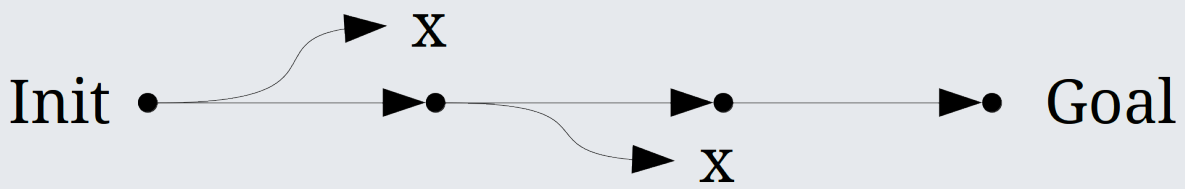
FOND Solutions and Representation

Weak Plan

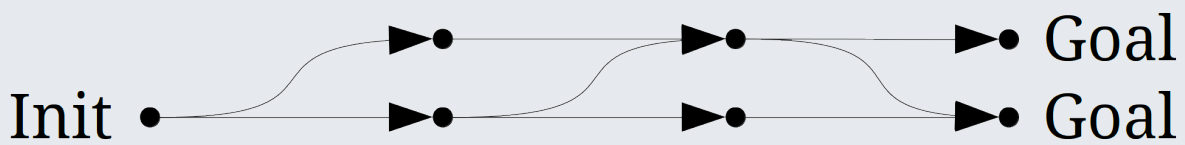


FOND Solutions and Representation

Weak Plan

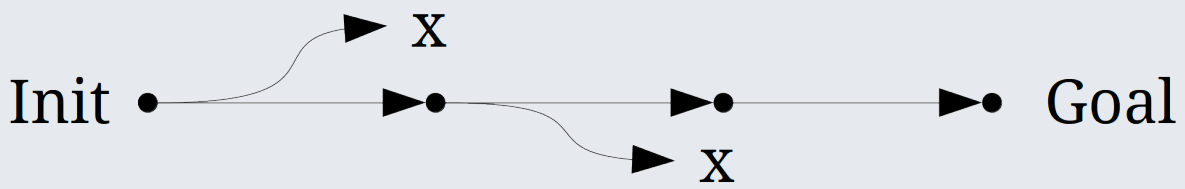


Strong Plan

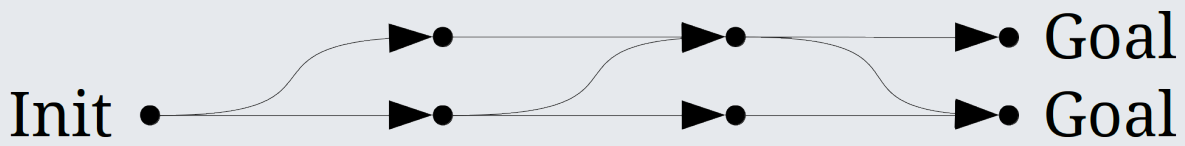


FOND Solutions and Representation

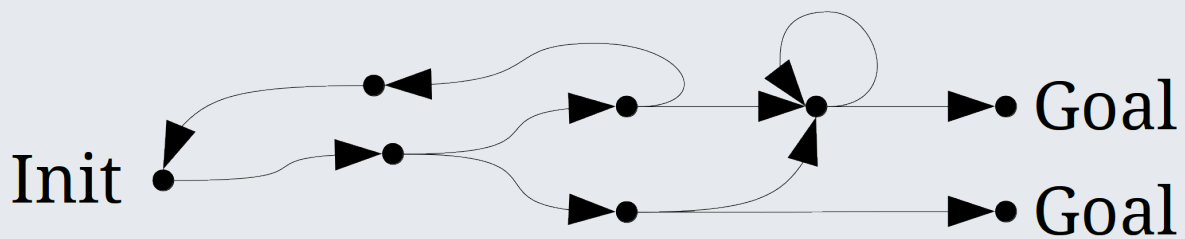
Weak Plan



Strong Plan



Strong Cyclic Plan

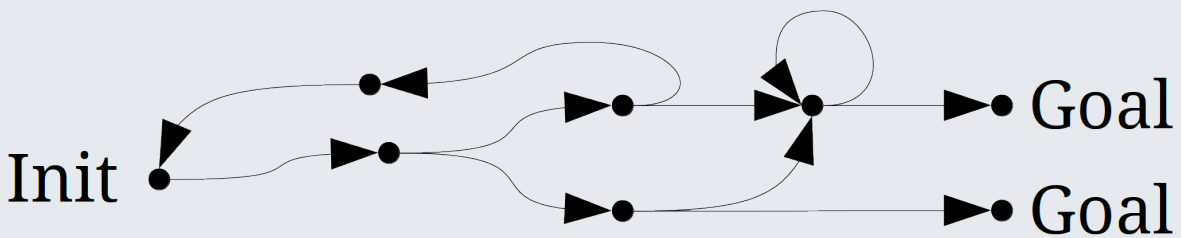


FOND Solutions and Representation

Policy

$P(s)$: Given a set of triples of the form $\langle p, a, c \rangle$ where p is a partial state, a an action, and c a cost, return the action a in state s from the pair with the lowest cost c such that $s \models p$.

Strong Cyclic Plan



First-Person Multi-agent Planning

Key Idea We are just a single agent operating in a multi-agent world. Planning should *not* be omnipotent, but rather conditioned on what others may do.

First-Person Multi-agent Planning

Key Idea We are just a single agent operating in a multi-agent world. Planning should *not* be omnipotent, but rather conditioned on what others may do.

First-Person MAP Problem $\langle \overrightarrow{ag}, App, \Pi \rangle$:

- $\overrightarrow{ag}$: Sequence of agents to act in the world
- $i \in \overrightarrow{ag}, App(i)$: Set of actions agent i can execute. $App(s, i) \subseteq App(i)$ are those also applicable in state s

We control agent $me \in \overrightarrow{ag}$, but must react to anything the other agents may do when it is their turn to "play"

Solution: Same as FOND!!

Outline

Outline

2. Approach
   - (modified) PRP Algorithm
   - Plausible Outcomes

Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**

return P;

Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**
    $Open = \{\mathcal{I}\}$; $Seen = \{\}$;
    while $Open \neq \emptyset$ **do**

    ProcessDeadends();
return P;

Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**
    $Open = \{\mathcal{I}\}$; $Seen = \{\}$;
    while $Open \neq \emptyset$ **do**
        $s = Open$.pop();
        if $\neg$SATISFIES$(s, \mathcal{G}) \wedge s \notin Seen$ **then**
    PROCESSDEADENDS();
return P;

Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**
    $Open = \{\mathcal{I}\}$; $Seen = \{\}$;
    while $Open \neq \emptyset$ **do**
        $s = Open$.pop();
        if $\neg$SATISFIES$(s, \mathcal{G}) \wedge s \notin Seen$ **then**
            $Seen$.add(s);
    PROCESSDEADENDS();
return P;

Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**
    $Open = \{\mathcal{I}\}$; $Seen = \{\}$;
    while $Open \neq \emptyset$ **do**
        $s = Open$.pop();
        if $\neg$SATISFIES$(s, \mathcal{G}) \wedge s \notin Seen$ **then**
            $Seen$.add(s);
            if $P(s)$ is undefined **then**
                GENPLANPAIRS$(\langle \mathcal{F}, \mathcal{G}, s, \mathcal{A} \rangle, P)$;
    PROCESSDEADENDS();
return P;

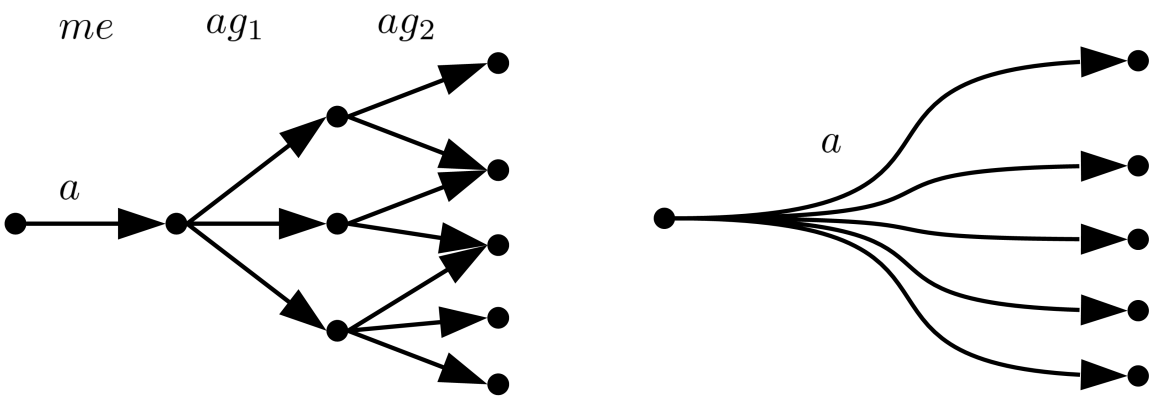
Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**
    $Open = \{\mathcal{I}\}$; $Seen = \{\}$;
    while $Open \neq \emptyset$ **do**
        $s = Open$.pop();
        if $\neg$SATISFIES$(s, \mathcal{G}) \wedge s \notin Seen$ **then**
            $Seen$.add(s);
            if $P(s)$ is undefined **then**
                GENPLANPAIRS$(\langle \mathcal{F}, \mathcal{G}, s, \mathcal{A} \rangle, P)$;
            if $P(s)$ is defined **then**
                for $s' \in$ **GenerateSuccessors**$(s, P(s))$ **do**
                    $Open$.add(s');
    PROCESSDEADENDS();
return P;

Computing Strong Cyclic Solutions

Input: FOND planning task $\Pi = \langle \mathcal{F}, \mathcal{G}, \mathcal{I}, \mathcal{A} \rangle$
Output: Partial policy P
Initialize policy P
while P changes **do**
    $Open = \{\mathcal{I}\}$; $Seen = \{\}$;
    while $Open \neq \emptyset$ **do**
        $s = Open$.pop();
        if $\neg$SATISFIES$(s, \mathcal{G}) \wedge s \notin Seen$ **then**
            $Seen$.add(s);
            if $P(s)$ is undefined **then**
                GENPLANPAIRS$(\langle \mathcal{F}, \mathcal{G}, s, \mathcal{A} \rangle, P)$;
            if $P(s)$ is defined **then**
                for $s' \in$ **GenerateSuccessors**$(s, P(s))$ **do**
                    $Open$.add(s');
    PROCESSDEADENDS();
return P;

Reinterpreting Multi-agent Actions



GenerateSuccessors

Input: State s, action a
Output: Set of successor states S

$S = \{Prog(s, a, e) \mid e \in \mathrm{Eff}_a\}$;
for $i = 1 \ldots |\overrightarrow{ag}|$ **do**
    $S' = \emptyset$;
    for $s' \in S$ **do**
        for $a' \in App(s', \overrightarrow{ag}[i])$ **do**
            $S' = S' \cup \{Prog(s', a', e) \mid e \in \mathrm{Eff}_{a'}\}$;
    $S = S'$;
return S;

GenerateSuccessors

Input: State s, action a
Output: Set of successor states S

$S = \{Prog(s, a, e) \mid e \in \mathrm{Eff}_a\}$;
for $i = 1 \ldots |\overrightarrow{ag}|$ **do**
    $S' = \emptyset$;
    for $s' \in S$ **do**
        for $a' \in \mathbf{App}(s', \overrightarrow{ag}[i])$ **do**
            $S' = S' \cup \{Prog(s', a', e) \mid e \in \mathrm{Eff}_{a'}\}$;
    $S = S'$;
return S;

Outline

2. Approach
   - (modified) PRP Algorithm
   - Plausible Outcomes

Restricting Non-determinism via Plausibility

Key Idea If we know the goal of some agents, then we should only consider their *plausible* actions.

Restricting Non-determinism via Plausibility

Key Idea If we know the goal of some agents, then we should only consider their *plausible* actions.

Plausibility Function

Top k actions from $App(s, ag)$ according to $Score$:

$$Score(s, a, ag, \mathcal{G}) = \min_{e \in \mathrm{Eff}_a} h_{FF}(Prog(s, a, e), \mathcal{G}(ag))$$

Restricting Non-determinism via Plausibility

Key Idea If we know the goal of some agents, then we should only consider their *plausible* actions.

Plausibility Function

Top k actions from $App(s, ag)$ according to $Score$:

$$Score(s, a, ag, \mathcal{G}) = \min_{e \in \mathrm{Eff}_a} h_{FF}(Prog(s, a, e), \mathcal{G}(ag))$$

Generalized Plausibility We can substitute any heuristic in place of h_{FF}. Further, we can consider *any* reasonable notion of plausibility: UCT sampling, nested ToM, etc.

GenerateSuccessors

Input: State s, action a
Output: Set of successor states S

$S = \{Prog(s, a, e) \mid e \in \mathrm{Eff}_a\}$;
for $i = 1 \ldots |\overrightarrow{ag}|$ **do**
    $S' = \emptyset$;
    for $s' \in S$ **do**
        for $a' \in \mathbf{App}(s', \overrightarrow{ag}[i])$ **do**
            $S' = S' \cup \{Prog(s', a', e) \mid e \in \mathrm{Eff}_{a'}\}$;
    $S = S'$;
return S;

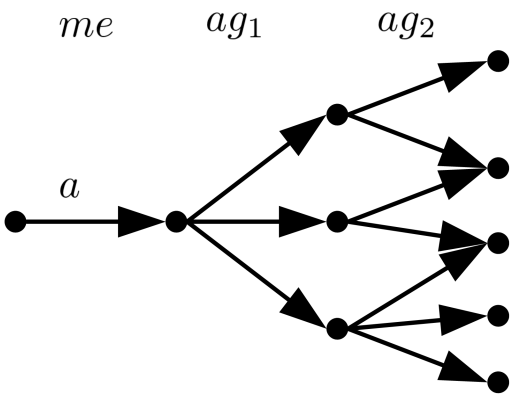
GenerateSuccessors

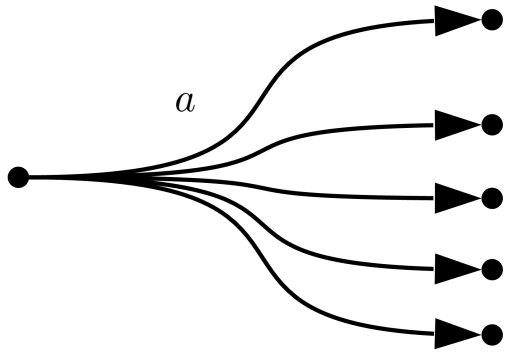
Input: State s, action a
Output: Set of successor states S

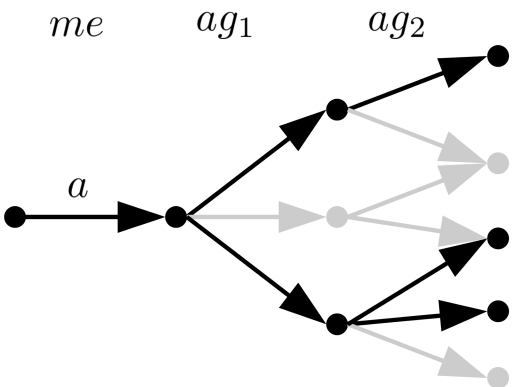
$S = \{Prog(s, a, e) \mid e \in \mathrm{Eff}_a\}$;
for $i = 1 \ldots |\overrightarrow{ag}|$ **do**
    $S' = \emptyset$;
    for $s' \in S$ **do**
        for $a' \in \mathbf{Plausible}(s', \overrightarrow{ag}[i])$ **do**
            $S' = S' \cup \{Prog(s', a', e) \mid e \in \mathrm{Eff}_{a'}\}$;
    $S = S'$;
return S;

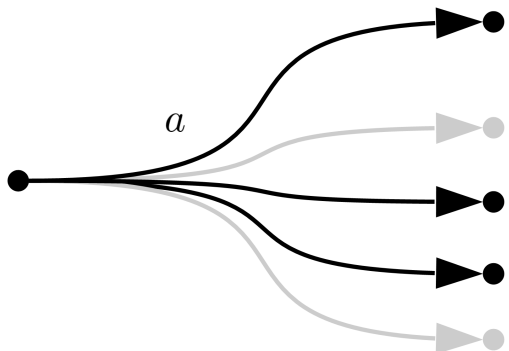
$$Plausible(s', \overrightarrow{ag}[i]) \subseteq App(s', \overrightarrow{ag}[i])$$

Reinterpreting Multi-agent Actions









Outline

Setup

Behaviour of "Others"

- Not reasonable for agents to follow the same plausibility function (overfitting) or random moves (uninformative).
- **Compromise**: Other agents use 1000 Monte-Carlo roll-outs per action to see which is most appealing given their own goal.

Setup

Behaviour of "Others"

- Not reasonable for agents to follow the same plausibility function (overfitting) or random moves (uninformative).
- **Compromise**: Other agents use 1000 Monte-Carlo roll-outs per action to see which is most appealing given their own goal.

Metrics

Policy size:	The number of partial state-action pairs in the best found policy.
Success rate:	The percentage of the 1000 trials that ended in success for the planning agent.
Planning time:	The number of seconds required to compute the policy (after 30m, best policy is used).

Tic-Tac-Toe

p1 Open board, goal to draw.

p2 Same as **p1**, not using noops.

p3 Open board, goal to win.

p4 Bottom corners played, goal to win.

	N	p1	p2	p3	p4
Success rate (%)	1	0	1	45	48
	2	0	2	47	100
	3	0	4	47	100
	4	16	3	47	100
	∞	100	100	79	100
	Rnd	39	81	52	100
Policy size	1	42	47	14	7
	2	88	107	26	15
	3	121	260	19	15
	4	871	250	22	15
	∞	1358	651	69	15
	Rnd	827	606	27	12
Planning time (s)	1	5	1	0.01	0.01
	2	155	11	0.26	0.01
	3	658	40	0.50	0.01
	4	978	39	7.34	0.01
	∞	1765	114	39.32	0.01
	Rnd	30m	130	0.01	0.01

Blocksworld

	N	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
Success rate (%)	1	38	63	31	20	0	4	0	19	12	11
	2	62	66	61	73	17	22	47	81	18	39
	3	92	97	90	96	49	62	81	90	84	76
	4	100	100	100	100	95	70	89	100	98	86
	∞	100	100	100	100	100	68	85	100	100	100
	Rnd	100	100	100	100	100	68	88	100	100	100
Policy size	1	32	27	31	111	49	25	68	63	43	30
	2	48	77	78	316	141	99	175	217	73	83
	3	125	114	157	576	298	246	445	459	126	164
	4	337	236	593	932	757	973	892	830	593	526
	∞	550	286	631	1113	1149	785	987	699	867	818
	Rnd	586	444	671	1045	1076	662	1006	763	745	818
Planning time (s)	1	0.01	0.01	0.01	0.02	0.02	0.01	0.02	0.02	0.02	0.01
	2	0.02	0.04	0.04	0.14	0.06	0.04	0.08	0.12	0.02	0.04
	3	0.06	0.06	0.06	0.32	0.24	0.16	0.56	0.36	0.04	0.10
	4	0.24	0.14	0.62	1.02	1.10	1.08	0.98	0.90	0.68	0.54
	∞	0.14	0.06	0.26	0.44	0.46	0.30	0.40	0.22	0.32	0.48
	Rnd	0.20	0.12	0.28	0.44	0.44	0.28	0.44	0.26	0.28	0.36

Sokoban

	N	p1	p2	p3	p4	p5
Success rate (%)	1	100	99	97	67	28
	2	100	98	96	98	100
	3	100	98	96	97	100
	4	100	✘	✘	99	100
	∞	100	✘	✘	✘	✘
	Rnd	100	71	48	37	✘
Policy size	1	11	27	27	138	906
	2	11	26	31	5990	7891
	3	11	26	31	10952	10223
	4	11	✘	✘	10312	9270
	∞	11	✘	✘	✘	✘
	Rnd	11	11	11	11	✘
Planning time (s)	1	0.06	0.08	0.08	0.46	195
	2	0.06	0.08	0.10	30m	30m
	3	0.06	0. 10	0.12	30m	30m
	4	0.06	✘	✘	30m	30m
	∞	0.06	✘	✘	✘	✘
	Rnd	0.08	0.06	0.08	0.08	✘

Outline

Considerations of Efficiency

The value of doing nothing

Under what situations does it pay off to model noop actions for the other agents?

Considerations of Efficiency

The value of doing nothing

Under what situations does it pay off to model noop actions for the other agents?

Issues with FOND as a black box

What prevents us from casting the First-Person MAP problem using a traditional FOND encoding / planner?

Considerations of Interpretation

Winning -vs- not losing

Is it better to win most of the time with some chance of losing, or to never lose but draw often?

Considerations of Interpretation

Winning -vs- not losing

Is it better to win most of the time with some chance of losing, or to never lose but draw often?

(Un)Fair non-determinism

How damaging is the assumption of fairness in MAP? (e.g., unfair adversaries, livelocks, etc)

Outline

Summary

- Addressed a complex and understudied form of multi-agent planning problems
- Employed state-of-the-art FOND planning techniques to compute compact solutions
- Introduced a mechanism for reducing the amount of non-determinism based on a model of the other agents

Future Work

- Gracefully move from very plausible to all outcomes
- Explore richer notions for a plausibility function
- Experiment on General Game Playing domains

Any Questions?

Demo, benchmarks, code, and slides available at:
`http://www.haz.ca/research/map-as-fond/`

THE UNIVERSITY OF MELBOURNE