Bounded Branching and Modalities in Non-Deterministic Planning

Blai Bonet

Departamento de Computación

Universidad Simón Bolívar

Caracas, Venezuela

Introduction

- We consider variations on the task of deciding the existence of solutions for non-deterministic planning problems:
  - Bounds in the number of branch points in a plan
  - Extensions of the description language with modal formulae
- The first applies to planning problems with complete and partial information; the first treatment of this problem appears to be [Meuleau & Smith, 2003]
- The second variation only applies to the case of planning problems with partial information

Goals of This Talk

- Make an overview of (some) known results about complexity of planning
- Motivate the relevance of proposed variations
- Make an overview of the new complexity results
- Won't go over proofs, yet will give some hints

Outline

- Planning with Complete Information
  - Classical (deterministic) planning
  - Non-deterministic planning (aka contingent planning)
  - Conformant and plans with bounded branching
- Planning with Partial Information
  - Contingent planning
  - Conformant and plans with bounded branching
- Two Special Cases

Background: Deterministic Models

- Understood in terms of:
  - a discrete and finite state space S
  - an initial state $s_0 \in S$
  - a non-empty set of goal states $G \subseteq S$
  - actions $A(s) \subseteq A$ applicable in each state s
  - a function that maps states and actions into states $f(a,s) \in S$

- **Solutions:** sequences $(a_0, \ldots, a_n)$ of actions that "transform" s_0 into a goal state

Background: Description Language

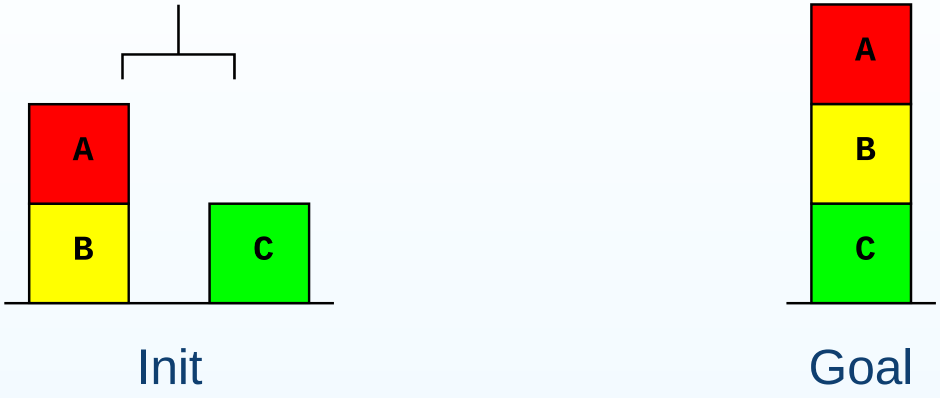
- Propositional language used to **compactly** describe the transition function and the applicable actions
- States are valuations to propositional symbols
- We use an action language similar to that in [Rintanen, 2004]:
  - Actions are pairs $\langle prec, effect \rangle$
  - $prec$ is a propositional formula used to define $A(s)$
  - Effects include atomic effects, conditional effects and conjunctions
- Initial state defined by the set I of propositions that hold true
- Goal states defined by a propositional formula Φ_G

Example – Blocksworld (Deterministic)

Init Goal

- Propositions:
  - Blocks' positions: $\{$`on-table(B)`,`on(A,B)`,`on-table(C)`$\}$
  - Others: $\{$`clear(A)`,`clear(C)`,`empty-hand`$\}$

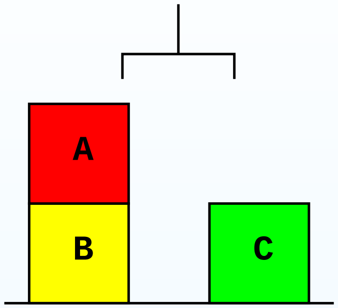
Example – Blocksworld (Deterministic)



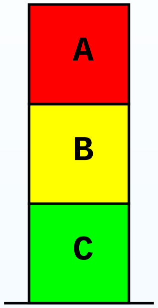
- Propositions:
  - Blocks' positions: {on-table(B), on(A,B), on-table(C)}
  - Others: {clear(A), clear(C), empty-hand}

- Actions:
  - **unstack(A,B)**:
    ⟨empty-hand ∧ clear(A) ∧ on(A,B), holding(A) ∧ clear(B) ∧ ¬on(A,B)⟩
  - **pick(A)**: ⟨empty-hand ∧ clear(A) ∧ on-table(A), holding(A) ∧ ¬on-table(A)⟩
  - **stack(A,B)**: ⟨holding(A) ∧ clear(B), empty-hand ∧ on(A,B) ∧ ¬holding(A)⟩
  - **drop(A)**: ⟨holding(A), empty-hand ∧ on-table(A) ∧ ¬holding(A)⟩

Example – Blocksworld (Deterministic)



Init



Goal

- Propositions:
  - Blocks' positions: $\{$`on-table(B)`, `on(A,B)`, `on-table(C)`$\}$
  - Others: $\{$`clear(A)`, `clear(C)`, `empty-hand`$\}$
- Actions:
  - **`unstack(A,B)`**:
    $\langle$`empty-hand` $\wedge$ `clear(A)` $\wedge$ `on(A,B)`, `holding(A)` $\wedge$ `clear(B)` $\wedge\neg$`on(A,B)`$\rangle$
  - **`pick(A)`**: $\langle$`empty-hand` $\wedge$ `clear(A)` $\wedge$ `on-table(A)`, `holding(A)` $\wedge\neg$`on-table(A)`$\rangle$
  - **`stack(A,B)`**: $\langle$`holding(A)` $\wedge$ `clear(B)`, `empty-hand` $\wedge$ `on(A,B)` $\wedge\neg$`holding(A)`$\rangle$
  - **`drop(A)`**: $\langle$`holding(A)`, `empty-hand` $\wedge$ `on-table(A)` $\wedge\neg$`holding(A)`$\rangle$
- Plan: (`unstack(A,B)`, `drop(A)`, `pick(B)`, `stack(B,C)`, `pick(A)`, `stack(A,B)`)

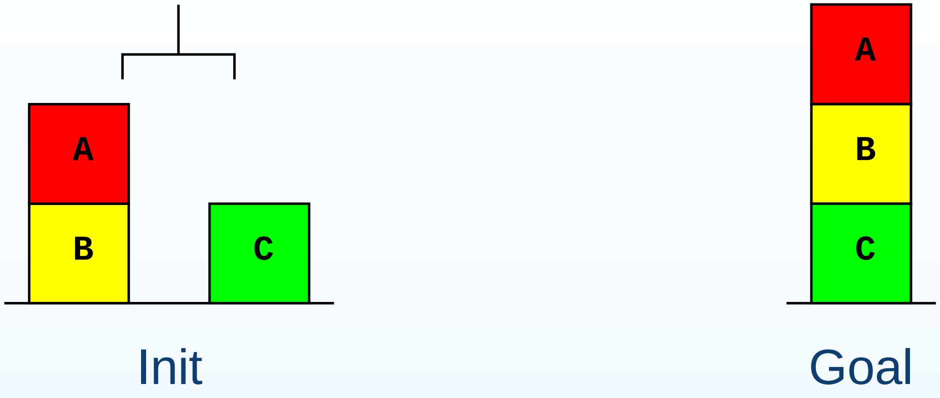
Non-Deterministic Planning with Complete Information

- Non-deterministic planning deals with problems where actions might have more than one outcome (non-deterministic actions)
- After the application of an action, the agent **observes the state** of the system and chooses next action
- This is a **branch point** in the plan!
- Another possibility is to apply a **sequence of actions blindly**, make a single observation at the end, and then choose next sequence of actions
- This is also a branch point in the plan!
- It is a natural to ask whether there exist plans of bounded branching

Non-Deterministic Models

- As deterministic models but transition function maps states and actions into **sets of states** $F(a,s) \subseteq S$
- There can be more than one initial state described by formula Φ_I
- Description language extended with non-deterministic effects
- Solutions cannot be sequences of actions!
- Solutions are tree-like structures called **contingent plans**

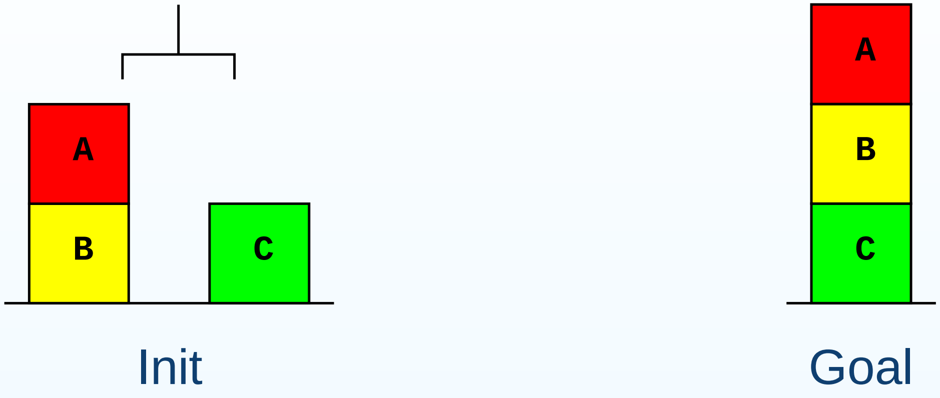
Example – Blocksworld (Non-Deterministic)



- New Action:
  - **`unstack(A,B)`**:

$$\langle \texttt{empty-hand} \wedge \texttt{clear(A)} \wedge \texttt{on(A,B)},$$
$$\texttt{holding(A)} \wedge \texttt{clear(B)} \wedge \neg\texttt{on(A,B)} \oplus$$
$$\texttt{clear(B)} \wedge \texttt{on-table(A)} \wedge \neg\texttt{on(A,B)}\rangle$$

Example – Blocksworld (Non-Deterministic)



- New Action:
  - **unstack(A,B)**:

$$\langle \texttt{empty-hand} \land \texttt{clear(A)} \land \texttt{on(A,B)},$$
$$\texttt{holding(A)} \land \texttt{clear(B)} \land \neg\texttt{on(A,B)} \oplus$$
$$\texttt{clear(B)} \land \texttt{on-table(A)} \land \neg\texttt{on(A,B)}\rangle$$

- Contingent Plan:

$$\texttt{unstack(A,B)} \begin{cases} \texttt{drop(A)},\texttt{pick(B)},\texttt{stack(B,C)},\texttt{pick(A)},\texttt{stack(A,B)} \\ \texttt{pick(B)},\texttt{stack(B,C)},\texttt{pick(A)},\texttt{stack(A,B)} \end{cases}$$

Complexity of Deterministic and Non-Deterministic Planning

- PLAN-DET is PSPACE-complete [Bylander, 1994]
- Deciding existence of solution for a contingent problem with full observability (i.e. PLAN-FO-CONT) is EXPTIME-complete
- Shown by [Rintanen, 2004] using Alternating TMs with polynomial space bound

Problem	Complete for	Reference
PLAN-DET	PSPACE	[Bylander, 1994]
PLAN-FO-CONT	EXPTIME	[Rintanen, 2004]

Complexity of Deterministic Planning

- The existence of a plan can be decided with the non-deterministic program:

    1. Let $counter := 0$
    2. Let $state := I$
    3. If $state \models \Phi_G$, then ACCEPT
    4. Choose applicable action a in $state$
    5. Let $state := f(a, state)$
    6. Let $counter := counter + 1$
    7. If $counter = 2^{|P|}$, then REJECT
    8. Goto 3

- Therefore, PLAN-DET is in NPSPACE = PSPACE

- The fact than PLAN-DET is PSPACE-hard was shown in [Bylander, 1994] with a direct simulation of DTMs with polynomial space bound

Conformant Planning

- Let's consider actions of the form:
  - **drop(A):** $\langle \mathit{true}, (\texttt{holding(A)} \rhd \texttt{empty-hand} \wedge \texttt{on-table(A)} \wedge \neg\texttt{holding(A)})\rangle$

  in which the **precondition** has been moved into a **conditional effect**

Conformant Planning

- Let's consider actions of the form:
  - **drop(A)**: $\langle \mathit{true}, (\texttt{holding(A)} \rhd \texttt{empty-hand} \wedge \texttt{on-table(A)} \wedge \neg\texttt{holding(A)})\rangle$

  in which the **precondition** has been moved into a **conditional effect**

- It's not hard to show that the plan:

  pick(A),drop(A),pick(B),drop(B),pick(C),drop(C),pick(A),drop(A),
  pick(B),drop(B),pick(C),drop(C),pick(B),stack(B,C),pick(A),stack(A,B)

  achieves the goal (i.e. A on B on C) **no matter what's the initial situation**

Conformant Planning

- Let's consider actions of the form:
  - **drop(A):** $\langle true, (\texttt{holding(A)} \rhd \texttt{empty-hand} \wedge \texttt{on-table(A)} \wedge \neg\texttt{holding(A)})\rangle$

  in which the **precondition** has been moved into a **conditional effect**

- It's not hard to show that the plan:

  pick(A), drop(A), pick(B), drop(B), pick(C), drop(C), pick(A), drop(A),
  pick(B), drop(B), pick(C), drop(C), pick(B), stack(B,C), pick(A), stack(A,B)

  achieves the goal (i.e. A on B on C) **no matter what's the initial situation**

- This plan is called **conformant** [Goldman & Boddy, 1996; Smith & Weld, 1998]

Conformant Planning

- Let's consider actions of the form:
  - **drop(A):** $\langle true, (\texttt{holding(A)} \rhd \texttt{empty-hand} \wedge \texttt{on-table(A)} \wedge \neg\texttt{holding(A)})\rangle$

  in which the **precondition** has been moved into a **conditional effect**

- It's not hard to show that the plan:

  pick(A), drop(A), pick(B), drop(B), pick(C), drop(C), pick(A), drop(A),

  pick(B), drop(B), pick(C), drop(C), pick(B), stack(B,C), pick(A), stack(A,B)

  achieves the goal (i.e. A on B on C) **no matter what's the initial situation**

- This plan is called **conformant** [Goldman & Boddy, 1996; Smith & Weld, 1998]

- **A conformant plan is a no-branch plan for a non-deterministic planning problem with full observability!!**

Complexity of Conformant Planning

- Checking the existence of a conformant plan (i.e. PLAN-FO-CONF) is EXPSPACE-complete
- Shown by [Haslum & Jonsson, 1999] using Regular Expressions with Exponentiation and Non-deterministic Finite Automata with Counters

Problem	Complete for	Reference
PLAN-DET	PSPACE	[Bylander, 1994]
PLAN-FO-CONT	EXPTIME	[Rintanen, 2004]
PLAN-FO-CONF	EXPSPACE	[Haslum & Jonsson, 1999]

Complexity of Conformant Planning

- The existence of a plan can be decided with the non-deterministic program:

    1. Let $counter := 0$
    2. Let $bel := \{s : s \models \Phi_I\}$
    3. If $s \models \Phi_G$ for all $s \in bel$, then ACCEPT
    4. Choose applicable action a in bel
    5. Let $bel := \cup\{F(a,s) : s \in bel\}$
    6. Let $counter := counter + 1$
    7. If $counter = 2^{2^{|P|}}$, then REJECT
    8. Goto 3

- Therefore, PLAN-FO-CONF is in NEXPSPACE = EXPSPACE

- The fact than PLAN-FO-CONF is EXPPSPACE-hard was shown in [Haslum & Jonsson, 1999]

Haslum & Jonsson Proof (Sketch)

- Let M be a non-deterministic TM with exponential space bound and $w \in \Sigma^*$
- Build **Regular Expression with Exponentiation** (REE) $\alpha = \alpha(w)$ such that

$$\alpha = \Sigma^* \iff w \notin L(M)$$

- Therefore, $\alpha = \Sigma^*$ is EXPSPACE-complete [Hopcroft & Ullman, 1979]

Haslum & Jonsson Proof (Sketch)

- Let M be a non-deterministic TM with exponential space bound and $w \in \Sigma^*$
- Build **Regular Expression with Exponentiation** (REE) $\alpha = \alpha(w)$ such that

$$\alpha = \Sigma^* \iff w \notin L(M)$$

- Therefore, $\alpha = \Sigma^*$ is EXPSPACE-complete [Hopcroft & Ullman, 1979]
- Build NFAC N such that $\sigma \in \alpha$ iff $\sigma \in N$ ($|N|$ is polynomial in $|\alpha|$)
- Build a conformant planning problem P such that:

$$\begin{aligned} P \text{ has solution} &\iff \exists\sigma : \sigma \notin L(N) \\ &\iff \exists\sigma : \sigma \notin \alpha \\ &\iff \alpha \neq \Sigma^* \end{aligned}$$

- This shows that PLAN-FO-CONF is co-EXPSPACE-hard = EXPSPACE-hard

Plans of Bounded Branching

- Contingent and conformant planning are **extreme** points of a discrete yet infinite range of solution forms:
  - Conformant = No branch
  - Contingent = Unbounded branch
- In the middle, we can think of plans with no more than k branches

Plans of Bounded Branching

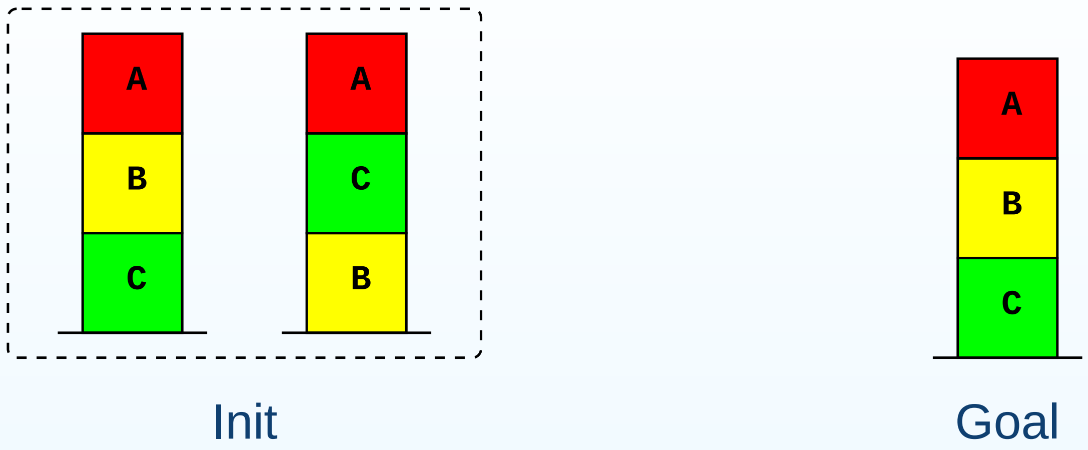
- Contingent and conformant planning are **extreme** points of a discrete yet infinite range of solution forms:
  - Conformant = No branch
  - Contingent = Unbounded branch
- In the middle, we can think of plans with no more than k branches
- Checking the existence of a contingent plan with at most k branches (i.e. PLAN-FO-CONT-k) is EXPSPACE-complete
- Proof similar to Haslum & Jonsson's for conformant planning

Problem	Complete for	Reference
PLAN-DET	PSPACE	[Bylander, 1994]
PLAN-FO-CONT	EXPTIME	[Rintanen, 2004]
PLAN-FO-CONF	EXPSPACE	[Haslum & Jonsson, 1999]
PLAN-FO-CONT-k	EXPSPACE	**New**

Problems with Partial Information

- Arise when the agent **cannot fully observe the state of the system**
- The agent receives some information after the execution of an action:
  - Full (the state is revealed)
  - Partial (e.g. the truth value of a proposition is revealed)
  - Null
- After the feedback is received, the agent chooses the next action
- This is a branch-point in the plan!

Example – Blocksworld (Partial Information)



- Observables: $Z = \{\texttt{clear(A)}, \texttt{clear(B)}, \texttt{clear(C)}\}$
- Current Knowledge: Block A is clear
- Contingent Plan:

$$\texttt{pick(A)} \begin{cases} \texttt{stack(A,B)} \\ \texttt{drop(A)}, \texttt{pick(C)}, \texttt{drop(C)}, \texttt{pick(B)}, \texttt{stack(B,C)}, \texttt{pick(A)}, \texttt{stack(A,B)} \end{cases}$$

Another Example – Game of Mastermind

- A simple game played by a codemaker and codebreaker:
    - Codemaker chooses a **secret code** at the beginning
    - Codebreaker must **discover** the code by making **guesses**
- Each guess answered with two tokens of information:
    - the number of matches in the guess
    - the number of “near” matches in the guess

Another Example – Game of Mastermind

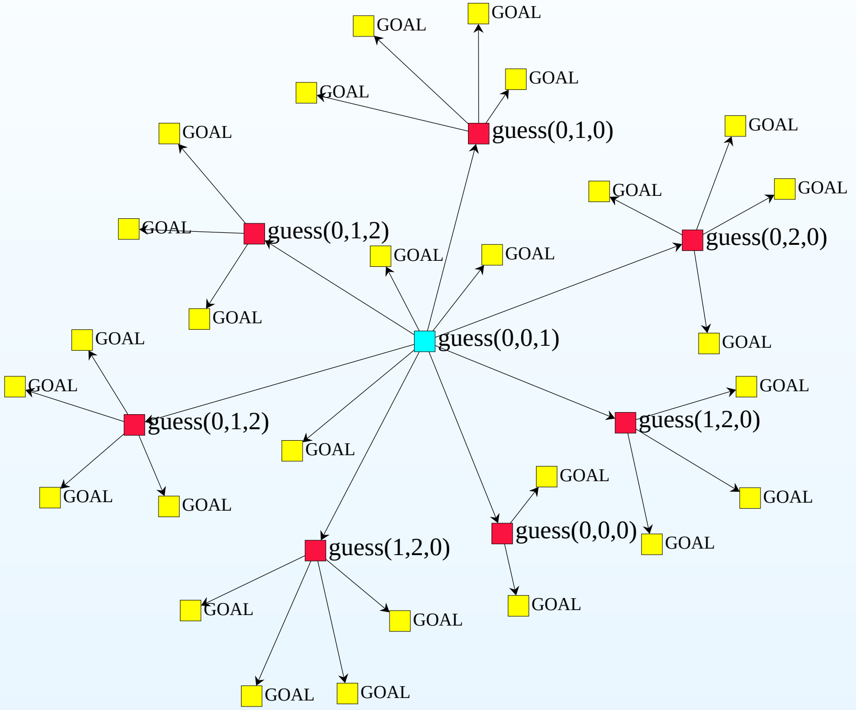
- A simple game played by a codemaker and codebreaker:
  - Codemaker chooses a **secret code** at the beginning
  - Codebreaker must **discover** the code by making **guesses**
- Each guess answered with two tokens of information:
  - the number of matches in the guess
  - the number of “near” matches in the guess
- The dynamics of the game can be modeled as a non-deterministic planning problem with partial information (the secret code is unknown)

Another Example – Game of Mastermind

- A simple game played by a codemaker and codebreaker:
  - Codemaker chooses a **secret code** at the beginning
  - Codebreaker must **discover** the code by making **guesses**
- Each guess answered with two tokens of information:
  - the number of matches in the guess
  - the number of “near” matches in the guess
- The dynamics of the game can be modeled as a non-deterministic planning problem with partial information (the secret code is unknown)
- However, the **goal of the game** (which is to know the secret code) **cannot be expressed in the language**
- A **modal formula** is needed to represent such a goal!!

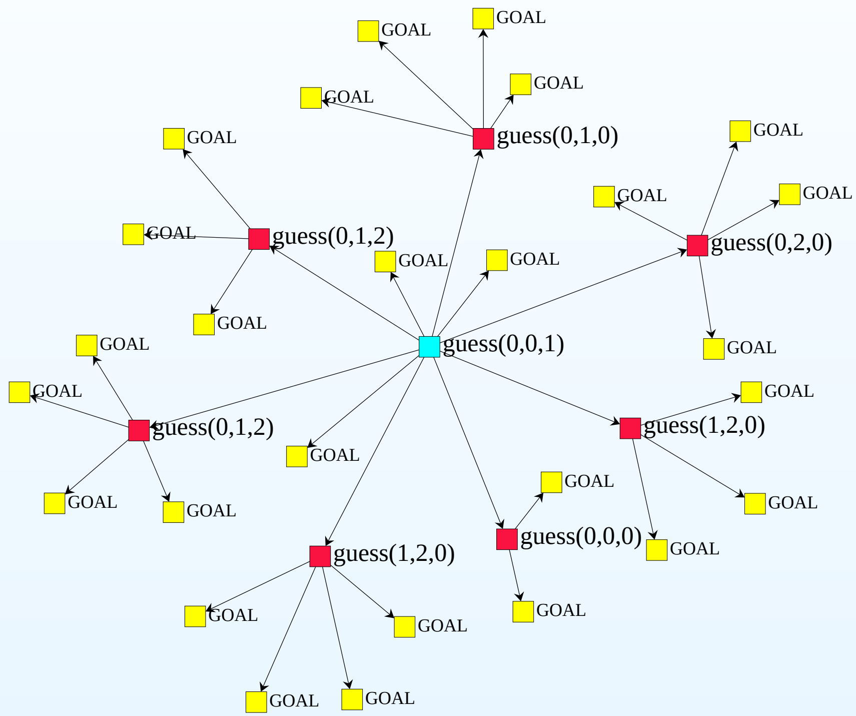
Mastermind: 3 colors, 3 pegs

- Contingent Plan:



Mastermind: 3 colors, 3 pegs

- Contingent Plan:



- We can also compute a **conformant plan** for this task!
- The following plan discovers the secret code no matter what's its value

$$\text{guess}(2,0,0)\ ,\ \text{guess}(2,1,0)\ ,\ \text{guess}(2,2,1)\ .$$

Complexity of Planning with Partial Information

- Checking the existence of a contingent plan (i.e. PLAN-PO-CONT) is 2EXPTIME-complete
- Shown by [Rintanen, 2004] using Alternating TMs with **exponential** space bound
- Checking the existence of conformant plans for partially observable problems with modal formulae is 2EXPSPACE-complete
- Shown using automatas with counters of double exponential capacity
- Checking the existence of plans with bounded number of branchs has same complexity of the conformant task

Problem	Complete for	Reference
PLAN-PO-CONT	2EXPTIME	[Rintanen, 2004]
PLAN-PO-CONF	2EXPSPACE	**New**
PLAN-PO-CONT-k	2EXPSPACE	**New**

Complexity of Conformant Planning with Partial Information

- The existence of a plan can be decided with the non-deterministic program:

  1. Let $counter := 0$
  2. Let $B := \{\{s : s \models \Phi_I\}\}$
  3. If $s \models \Phi_G$ for all $s \in b$ and $b \in B$, then ACCEPT
  4. Choose applicable action a in B
  5. Let $B := \cup_{b \in B} \cup_z \{b_a^z\}$
  6. Let $counter := counter + 1$
  7. If $counter = 2^{2^{2^{|P|}}}$, then REJECT
  8. Goto 3

- Therefore, PLAN-PO-CONF is in N2EXPSPACE = 2EXPSPACE

Lower Bound (Hardness)

- Let M be an NTM with **double** exponential space bound and $w \in \Sigma^*$
- Build REE α such that $\alpha = \Sigma^*$ iff $w \notin L(M)$ (size of α exponential)
- Build NFAC N (size of N w/o counting counter capacity is polynomial)
- Build then a conformant planning problem P such that P has a solution iff $\alpha \neq \Sigma^*$
- In order to get the reduction, need to show how to compactly encode double-exponential capacity counters with a planning problem of polynomial size

Encoding of Counters (Idea)

- The value of counter $c_{2^n-1} \cdots c_0$ of double exponential capacity can be represented as the set of **bits equal to 1**; i.e. $\{i : c_i = 1\}$
- The size of such set might be exponential, however each member of the set can be encoded as a subset from set of polynomial-number of bits!

Encoding of Counters (Idea)

- The value of counter $c_{2^n-1}\cdots c_0$ of double exponential capacity can be represented as the set of **bits equal to 1**; i.e. $\{i : c_i = 1\}$
- The size of such set might be exponential, however each member of the set can be encoded as a subset from set of polynomial-number of bits!
- Examples:

$$\begin{aligned} 18 &= 00010010 = \{1,4\} = \{\{0\},\{2\}\} \\ 129 &= 10000001 = \{0,7\} = \{\{\},\{0,1,2\}\} \end{aligned}$$

Encoding of Counters (Idea)

- The value of counter $c_{2^n-1}\cdots c_0$ of double exponential capacity can be represented as the set of **bits equal to 1**; i.e. $\{i : c_i = 1\}$
- The size of such set might be exponential, however each member of the set can be encoded as a subset from set of polynomial-number of bits!
- Examples:

$$\begin{aligned} 18 &= 00010010 = \{1,4\} = \{\{0\},\{2\}\} \\ 129 &= 10000001 = \{0,7\} = \{\{\},\{0,1,2\}\} \end{aligned}$$

- Multiple counters can be encoded using cross-products of propositional symbols

Two Special Cases

- Existence of plans of bounded branching of **polynomial** length (either full or partial observable case):
  - Can be done with QBFs!
  - Indeed, checking the existence of a plan with at most k branches is in Σ^p_{2k+4}

Two Special Cases

- Existence of plans of bounded branching of **polynomial** length (either full or partial observable case):
  - Can be done with QBFs!
  - Indeed, checking the existence of a plan with at most k branches is in Σ^p_{2k+4}
- Existence of conformant plans for partially observable problems **without** modal formulae is EXPSPACE-complete

Summary

- Considered two variations on the existence of plans:
  - Plans of bounded branching for full and partially observable problems
  - Extension of description language with modalities for planning with incomplete information
- Analyzed and derived tight bounds on the complexity of novel decision problems