First-order Models for Sequential Decision-making

Insights, Caveats, and Tricks-of-the-Trade

Scott Sanner

NICTA

THE AUSTRALIAN NATIONAL UNIVERSITY

Warning

- This talk is about:

~~Statistical~~ Relational ~~Learning~~

Probabilistic Planning

{ SRL, ILP, Human expert } provide the model

- But wait, don't leave…
  - Overlap with and extensions of lifted inference
    - FOPI / FOVE
  - In the end, we learn in order to make decisions!

Talk Summary

- Relational models are natural for many sequential decision-making problems

- But most sequential planners ground the relational model
  - thus throwing away all relational structure

- Potential gains with lifted relational solutions
  - interpretability
  - time and space efficiency

    ➔ scalability

Focus of this talk

Talk Outline

- First-order MDPs (FOMDPs)
  - "a first model" for first-order sequential decision theory
  - basic ideas and highlights
- Caveats of FOMDPs and workarounds
  - limitations and enhancements
  - tricks-of-the-trade
- Extensions
  - factored FOMDP
  - first-order POMDP

First-order MDPs without the BS

Bunch of Symbols

Scott Sanner

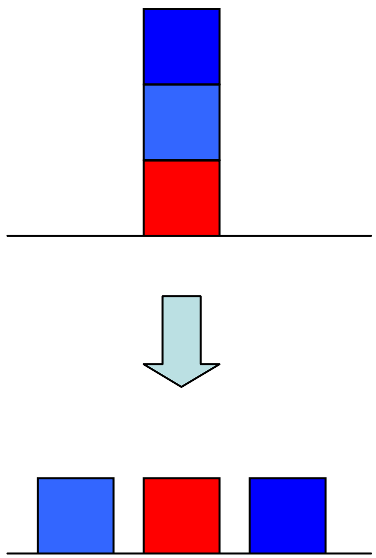
NICTA

THE AUSTRALIAN NATIONAL UNIVERSITY

Relational Planning Languages

- Common languages:
  - (P)STRIPS
  - (P)PDDL
    - more expressive than STRIPS
    - for example, *universal* and *conditional* effects:

```
(:action  put-all-blue-blocks-on-table
          :parameters  ( )
          :precondition  ( )
          :effect (forall (?b)
                     (when (Blue ?b)
                           (not (OnTable ?b)))))
```

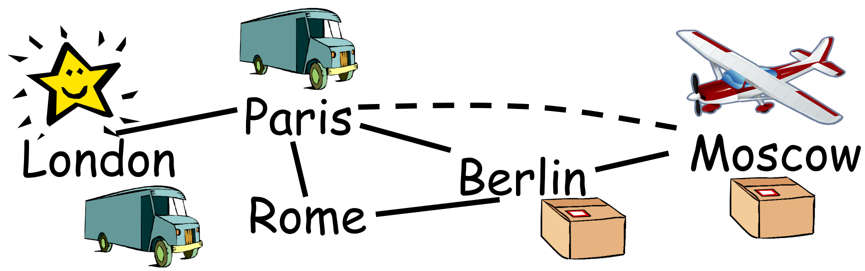


  - General Game Playing (GGP)
    - one or more agents

How to Solve?

- Relational planning *problem*:

Logistics:



(:action load-box-on-truck-in-city
:parameters (?b - box ?t - truck ?c – city)
:precondition (and (BIn ?b ?c) (TIn ?t ?c))
:effect (and (On ?b ?t) (not (BIn ?b ?c))))

- Solve *ground problem* for *each domain instance*?
  - 3 trucks:  2 planes:  3 boxes: 

- Or solve lifted specification for *all* domains at once?

Case Statement

- $\langle S, A, T, R \rangle$ for FOMDPs defined in terms of **case**
  - e.g., **reward case** in *Logistics* FOMDP:

$$rCase(s) = \begin{array}{|c|c|} \hline \exists b.\ BoxIn(b, paris, s) & 1 \\ \hline \neg\exists b.\ BoxIn(b, paris, s) & 0 \\ \hline \end{array}$$

Quantified formulae

Disjoint partitioning of world states

Case Operations

- **Operators:** Define unary & binary case operations
  - E.g., can take "cross-sum" $\oplus$ (or $\otimes$, $\ominus$) of cases…

ϕ	10
$\neg\phi$	20

$\oplus$

ψ	3
$\neg\psi$	4

=

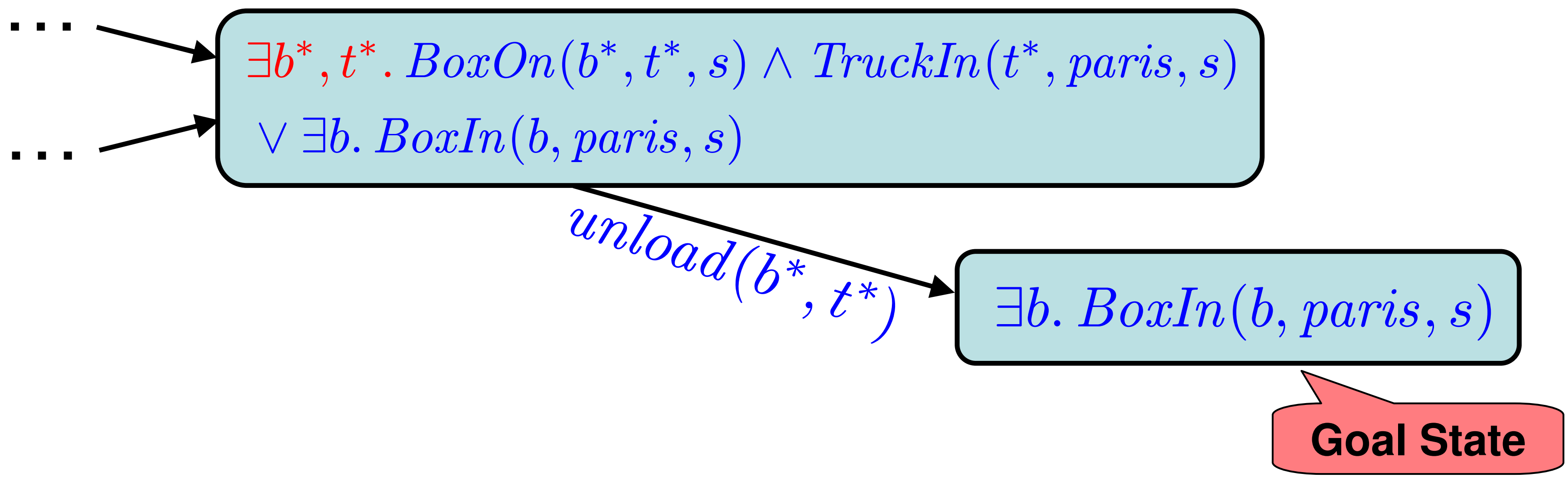
$\phi \wedge \psi$	13
~~$\phi \wedge \neg\psi$~~	~~14~~
$\neg\phi \wedge \psi$	23
$\neg\phi \wedge \neg\psi$	24

Inconsistent partitions should be removed for **compactness**

First-order Regression Planning

[Reiter's Default Solution to Frame Problem]

- Use regression to back-chain through actions



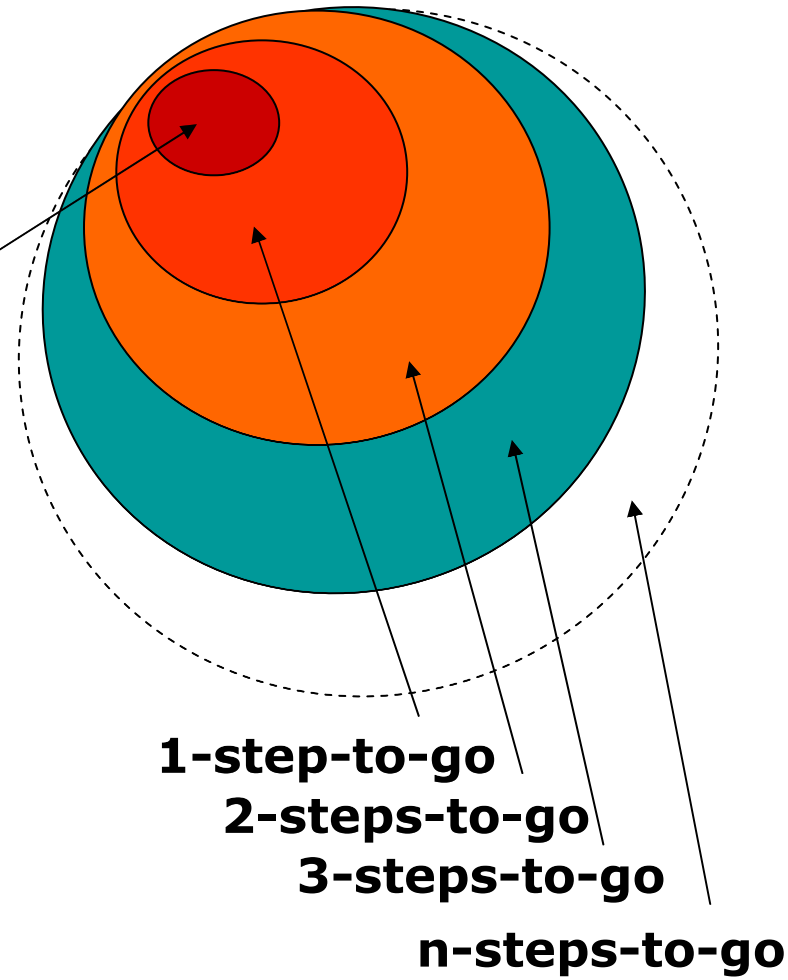
- Use $\exists$ to tell if valid action instantiation exists

First-order Regression Planning

- Define *abstract* goal / reward, e.g.,

$$\exists b.\ BoxIn(b, paris, s)$$

- Can take **expectation** over deterministic outcomes for **decision-theoretic regression**



Symbolic Dynamic Programming

- What value if 0-stages-to-go?
  - Obviously $V^0(s) = rCase(s)$
- What value if 1-stage-to-go?
  - We know Q-value for each action (regress + ∃ quantification)

$$V^1(s) = \quad max_s \left\{ \begin{array}{l} \begin{array}{|l|l|} \hline \varphi_1 & 9 \\ \hline \varphi_2 & 0 \\ \hline \end{array} = Q^1(s, load(b,t)\,) \\ \\ \begin{array}{|l|l|} \hline \varphi_3 & 3 \\ \hline \varphi_4 & 1 \\ \hline \end{array} = Q^1(s, unload(b,t)\,) \end{array} \right.$$

Symbolic Dynamic Programming

- What value if 0-stages-to-go?
  - Obviously $V^0(s) = rCase(s)$
- What value if 1-stage-to-go?
  - We know Q-value for each action (regress + ∃ quantification)

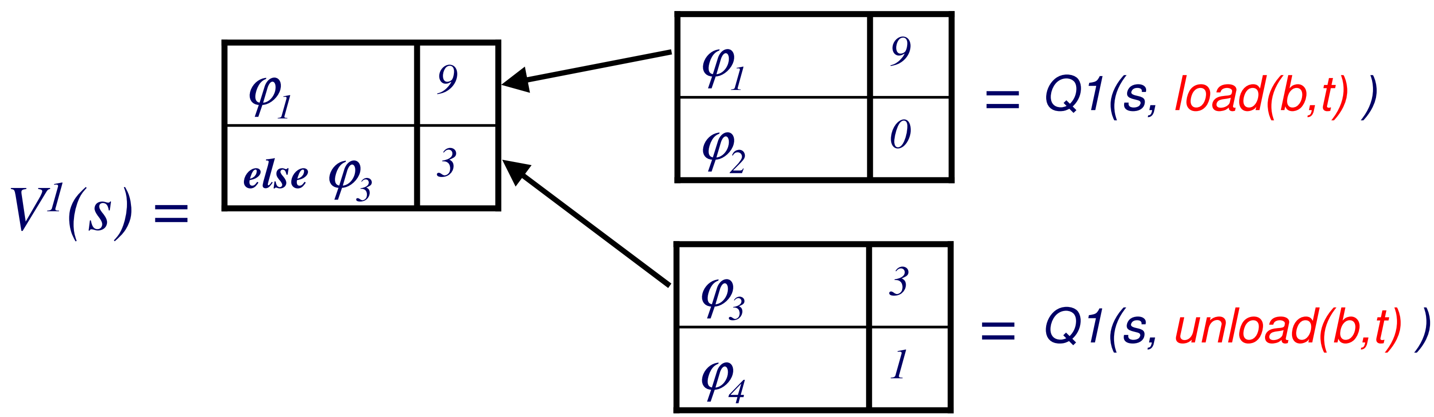
$$V^1(s) = \begin{array}{|c|c|}\hline \varphi_1 & 9 \\ \hline\end{array}$$

$$\begin{array}{|c|c|}\hline \varphi_1 & 9 \\ \hline \varphi_2 & 0 \\ \hline\end{array} = Q^1(s, load(b,t))$$

$$\begin{array}{|c|c|}\hline \varphi_3 & 3 \\ \hline \varphi_4 & 1 \\ \hline\end{array} = Q^1(s, unload(b,t))$$

Symbolic Dynamic Programming

- What value if 0-stages-to-go?
  - Obviously $V^0(s) = rCase(s)$
- What value if 1-stage-to-go?
  - We know Q-value for each action (regress + ∃ quantification)



Symbolic Dynamic Programming

- What value if 0-stages-to-go?
  - Obviously $V^0(s) = rCase(s)$
- What value if 1-stage-to-go?
  - We know Q-value for each action (regress + ∃ quantification)

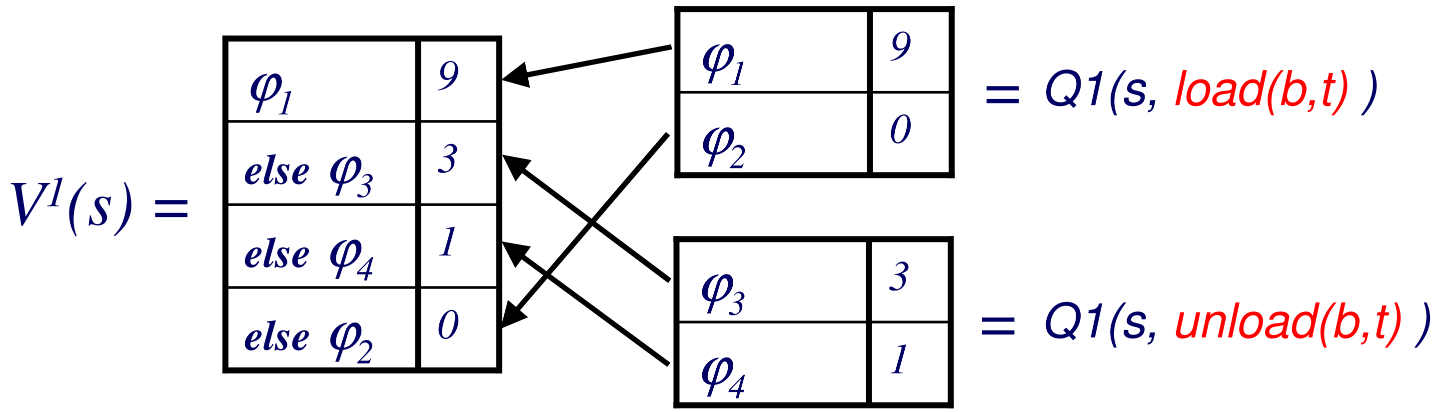
$$V^1(s) = \begin{array}{|l|l|}\hline \varphi_1 & 9 \\ \hline \textbf{else } \varphi_3 & 3 \\ \hline \textbf{else } \varphi_4 & 1 \\ \hline \end{array}$$

$$\begin{array}{|l|l|}\hline \varphi_1 & 9 \\ \hline \varphi_2 & 0 \\ \hline \end{array} = Q1(s, load(b,t))$$

$$\begin{array}{|l|l|}\hline \varphi_3 & 3 \\ \hline \varphi_4 & 1 \\ \hline \end{array} = Q1(s, unload(b,t))$$

Symbolic Dynamic Programming

- What value if 0-stages-to-go?
  - Obviously $V^0(s) = rCase(s)$
- What value if 1-stage-to-go?
  - We know Q-value for each action (regress + ∃ quantification)



- Value iteration: [Boutilier, Reiter, & Price, IJCAI-01]
  - Obtain V^{n+1} from V^n until $(V^{n+1} \ominus V^n) < \varepsilon$

Ex: First Two Steps of SDP

$$V^0(s) = R(s) = \begin{array}{|l|} \hline \exists b.BoxIn(b, paris, s) \quad : 10 \\ \hline \neg\exists b.BoxIn(b, paris, s) : \quad 0 \\ \hline \end{array}$$

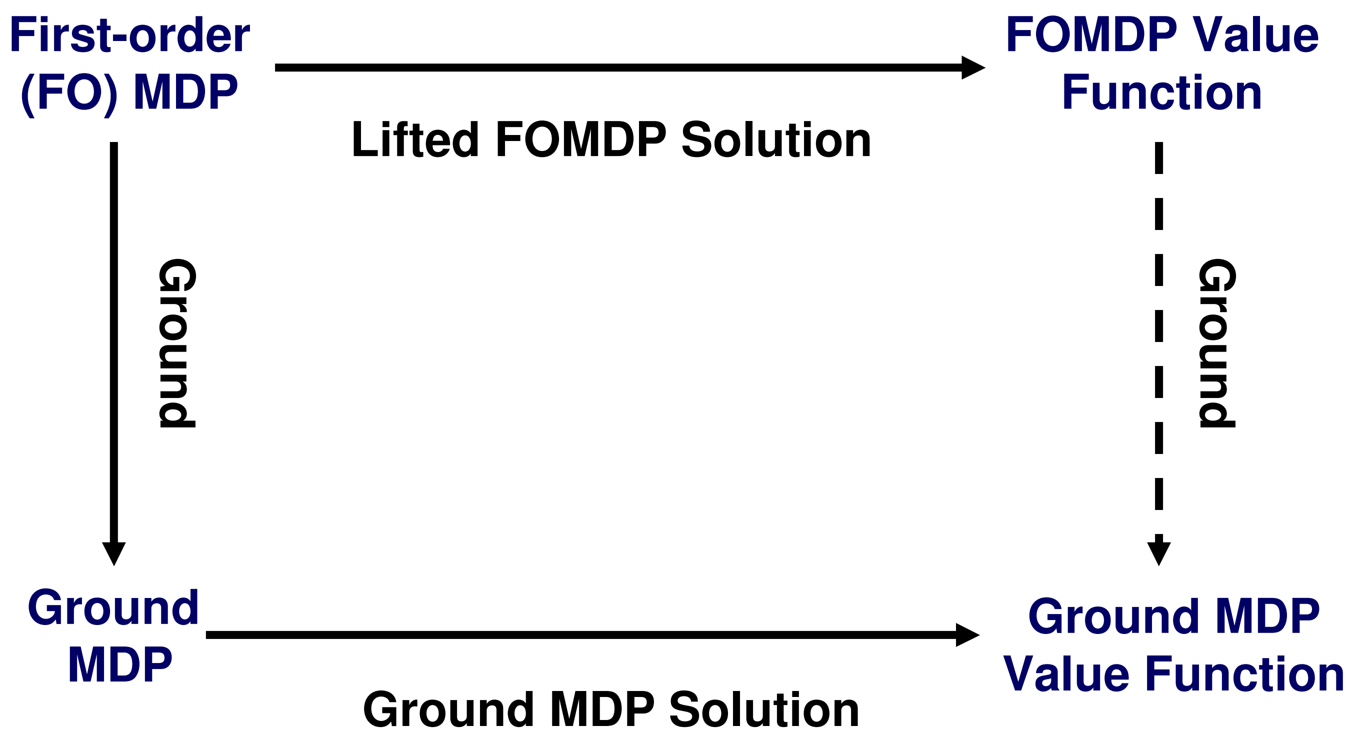
$$V^1(s) \ = \ \begin{array}{|l|} \hline \exists b.BoxIn(b, paris, s) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad : 19.0 \\ \hline \neg\text{``} \wedge \exists b, t.TruckIn(t, paris, s) \wedge BoxOn(b, t, s) : \quad 8.1 \\ \hline \neg\text{``} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad : \quad 0.0 \\ \hline \end{array}$$

$$V^2(s) \ = \ \begin{array}{|l|} \hline \exists b.BoxIn(b, paris, s) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad : 26.1 \\ \hline \neg\text{``} \wedge \exists b, t.TruckIn(t, paris, s) \wedge BoxOn(b, t, s) : 15.4 \\ \hline \neg\text{``} \wedge \exists b, c, t.BoxOn(b, t, s) \wedge TruckIn(t, c, s) \quad : \quad 7.3 \\ \hline \neg\text{``} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad : \quad 0.0 \\ \hline \end{array}$$

Correctness of SDP

[Boutilier, Reiter, & Price, IJCAI-01]

- **Show SDP for FOMDPs is correct w.r.t. ground MDP:**



Where are we?

- Loosely defined the FOMDP

- Described how it is possible to find a lifted solution independent of domain size
  - Exploits state abstraction
  - Exploits action abstraction (∃)

First-order MDPs

Caveats & Workarounds

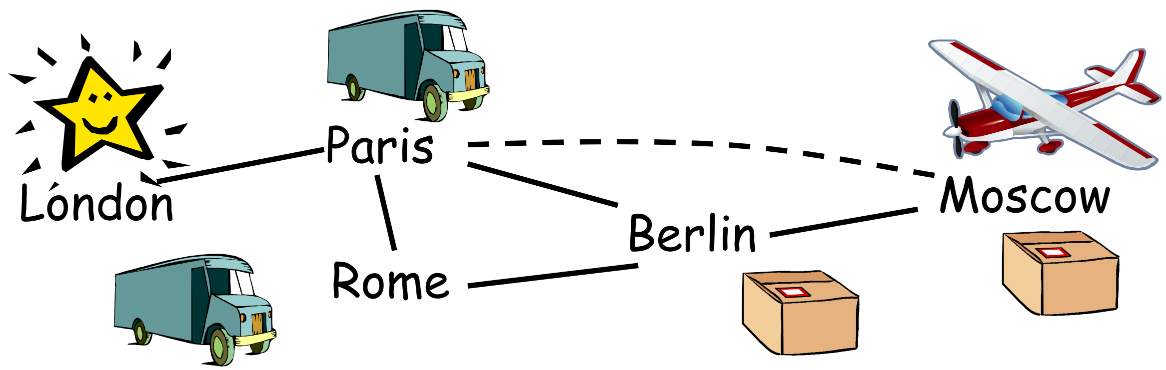
Scott Sanner

NICTA

THE AUSTRALIAN NATIONAL UNIVERSITY

Caveats of First-order Planning I

- Many problems have topologies
  - e.g, reachability constraints in logistics



- If topology not fixed *a priori*…
  - first-order solution must consider ∞ topologies
    - e.g., if *Moscow* reachable from *Rome* in five steps…
  - in general case, leads to ∞ values / policy

Fix your topology!

Caveats of First-order Planning II

- $\forall$ Rewards

identical goals ➔ lifted goal decomposition

$$R(s) = \boxed{\begin{array}{l|r} \forall b, c.\, Dst(b,c) \rightarrow BoxIn(b,c,s) & : 1 \\ \hline \neg \text{“} & : 0 \end{array}}$$

- Value function must distinguish ∞ cases

$$V^t(s) = \boxed{\begin{array}{lcr} \forall b, c.\, Dst(b,c) \rightarrow BoxIn(b,c,s) & : & 1 \\ \hline \text{One box not at destination} & : & \gamma \\ \hline \text{Two boxes not at destination} & : & \gamma^2 \\ \hline \vdots & : & \vdots \\ \hline t-1 \text{ boxes not at destination} & : & \gamma^{t-1} \end{array}}$$

- Policy will also likely be ∞
  - Some notable exceptions (*put all blocks on table*)

Caveats of First-order Planning III

- Unreachable States
  - (P)PDDL domains often under-constrained
    - *BlocksWorld:* 2 blocks cannot be on a 3^{rd} block
    - *Logistics:* 1 box cannot be in 2 cities at once
  - But nowhere are these constraints encoded!
- If no background theory to restrict legal states
  - First-order planning must solve for *all* states
  - Where most are illegal!
- But if initial states known…

First-order Real-time DP

- Simulate trials and do DP at every **visited state**
  - We know Q-value for each action (regress + ∃ quantification)

$$V^1(s) = \max_s \left\{ \begin{array}{l} \begin{array}{|c|c|} \hline \varphi_1 & 9 \\ \hline \cancel{\varphi_2} & \cancel{0} \\ \hline \end{array} = Q^1(s,\ load(b,t)\ ) \\ \\ \begin{array}{|c|c|} \hline \cancel{\varphi_3} & \cancel{3} \\ \hline \varphi_4 & 1 \\ \hline \end{array} = Q^1(s,\ unload(b,t)\ ) \end{array} \right.$$

- Lifted Symbolic DP at every step
- Ground evaluation to find partitions for current state
  - no theorem proving for consistency checking!
- Restrict lifted value function to reachable states

Caveats of First-order Planning IV

- Value function may grow very large

$$V^t(s) = \begin{array}{|lcr|} \hline \phi_1 & : & 100.0 \\ \hline \phi_2 & : & 10.0 \\ \hline \phi_3 & : & 2.5 \\ \hline \vdots & \vdots & \vdots \\ \hline \phi_{10000} & : & 1.1 \\ \hline \end{array}$$

- Need compact data structures and / or approximations…

First-order ADDs

- Want to compactly represent:

case =

$\exists x.[A(x) \vee \forall y.A(x) \wedge B(x) \wedge \neg A(y)]$	1
$\neg$ "	0

- Push down quantifiers, expose prop. structure:

$$[\exists x.A(x)] \vee ([\exists x.A(x) \wedge B(x)] \wedge [\forall y.\neg A(y)])$$

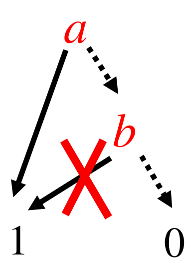
Var	*Var* $\Leftrightarrow$ *FOL KB*
a	$\equiv [\exists x.A(x)]$
b	$\equiv [\exists x.A(x) \wedge B(x)]$

case =

$a \vee (b \wedge \neg a)$	1
$\neg$ "	0

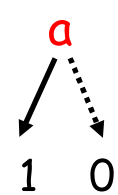
- Convert to first-order ADD

case =



First-order CSI!

=



FO-ADDs are Compact!

- Lifted reward for *Logistics*

$rCase(s) = \boxed{\exists b.\ BIn(b, Paris, s)}$

10 0

- Lifted value function for *Logistics*

$vCase(s) = \boxed{\exists b.\ BIn(b, Paris, s)}$

$100 : \ noop$ $\boxed{\exists b, t. TIn(t, Paris, s) \wedge On(b, t, s)}$

$89 : \ unload(b, t)$ $\boxed{\exists b, t.\ On(b, t, s)}$

$80 : \ drive(t, Paris)$ $\boxed{\exists b, c.\ BoxIn(b, c, s) \wedge \exists t. TIn(t, c, s)}$

$72 : \ load(b, t)$ ...

But sometimes you need to approximate…

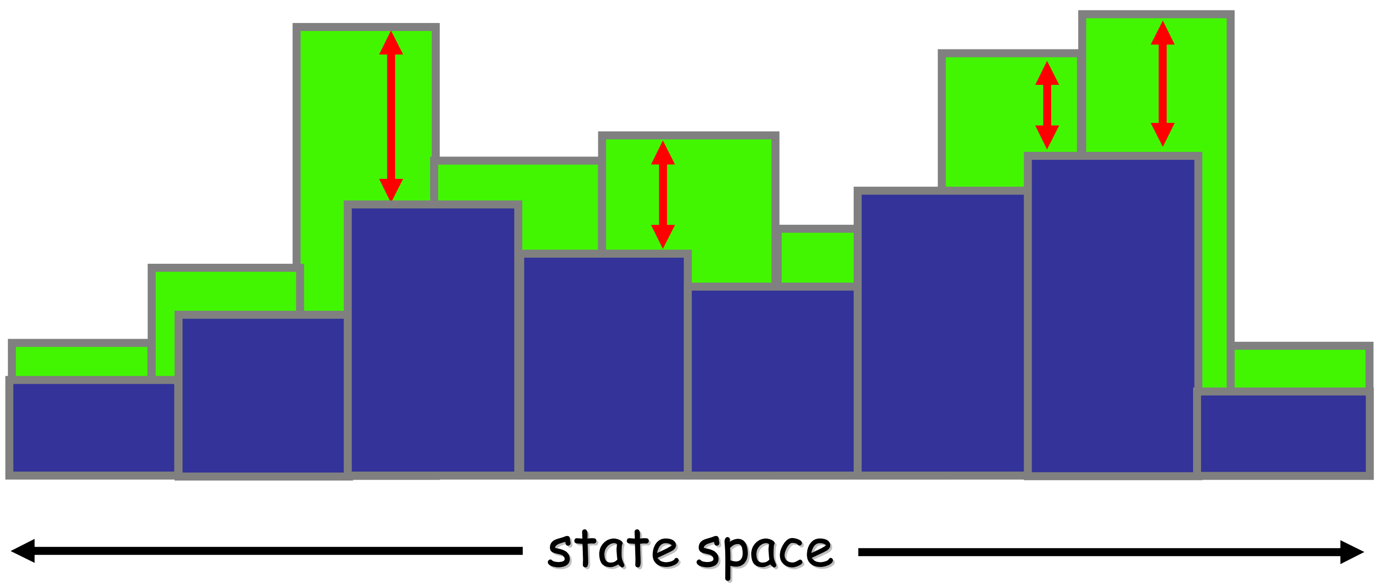
Approximate FOMDP Solutions via LP

- (SanBout, UAI-05/06) **FOALP** / **FOAPI:** Generalize approximate LP and policy iteration (PI) solutions
  - **First-order linear program:**

Vars: w_i; $i \leq k$

Minimize: $f(w_i)$

Subject to: $case_1(w,s) \geq case_2(w,s)$; $\forall s$



Constraint Generation for FO-LPs

- Example constraint:

$$0 \geq w_1 \cdot \begin{array}{|c|c|}\hline \phi(s) & 3 \\ \hline \neg\phi(s) & 4 \\ \hline \end{array} \oplus w_2 \cdot \begin{array}{|c|c|}\hline \varphi(s) & 10 \\ \hline \neg\varphi(s) & 20 \\ \hline \end{array} \; ; \forall s$$

- Only finite *distinct* constraints… but still many

$\phi(s) \wedge \varphi(s)$	$0 \geq 3w_1 + 10w_2$
$\neg\phi(s) \wedge \varphi(s)$	$0 \geq 4w_1 + 10w_2$
$\phi(s) \wedge \neg\varphi(s)$	$0 \geq 3w_1 + 20w_2$
$\neg\phi(s) \wedge \neg\varphi(s)$	$0 \geq 4w_1 + 20w_2$

- Solve LP via constraint generation
  - Efficiently find max violated constraints
  - Generalize variable elimination to *relation elimination!*

Relation Elimination for Constraint Gen.

$$0 \geq \max_s \left(\begin{array}{|ll|}\hline \forall b,c.\ Dst(b,c) \supset BoxIn(b,c,s) & : 10 \\ \hline \neg `` & : 0 \\ \hline \end{array} \oplus \begin{array}{|ll|}\hline \exists b,c.\ Dst(b,c) \wedge \neg BoxIn(b,c,s) : & w_1 \\ \hline \neg `` & -w_1 \\ \hline \end{array} \oplus \begin{array}{|ll|}\hline \exists t,c.\, TruckIn(t,c,s) & : w_2 \\ \hline \neg `` & : 0 \\ \hline \end{array} \right)$$

$$w_1 = 2 \; and \; w_2 = 1$$

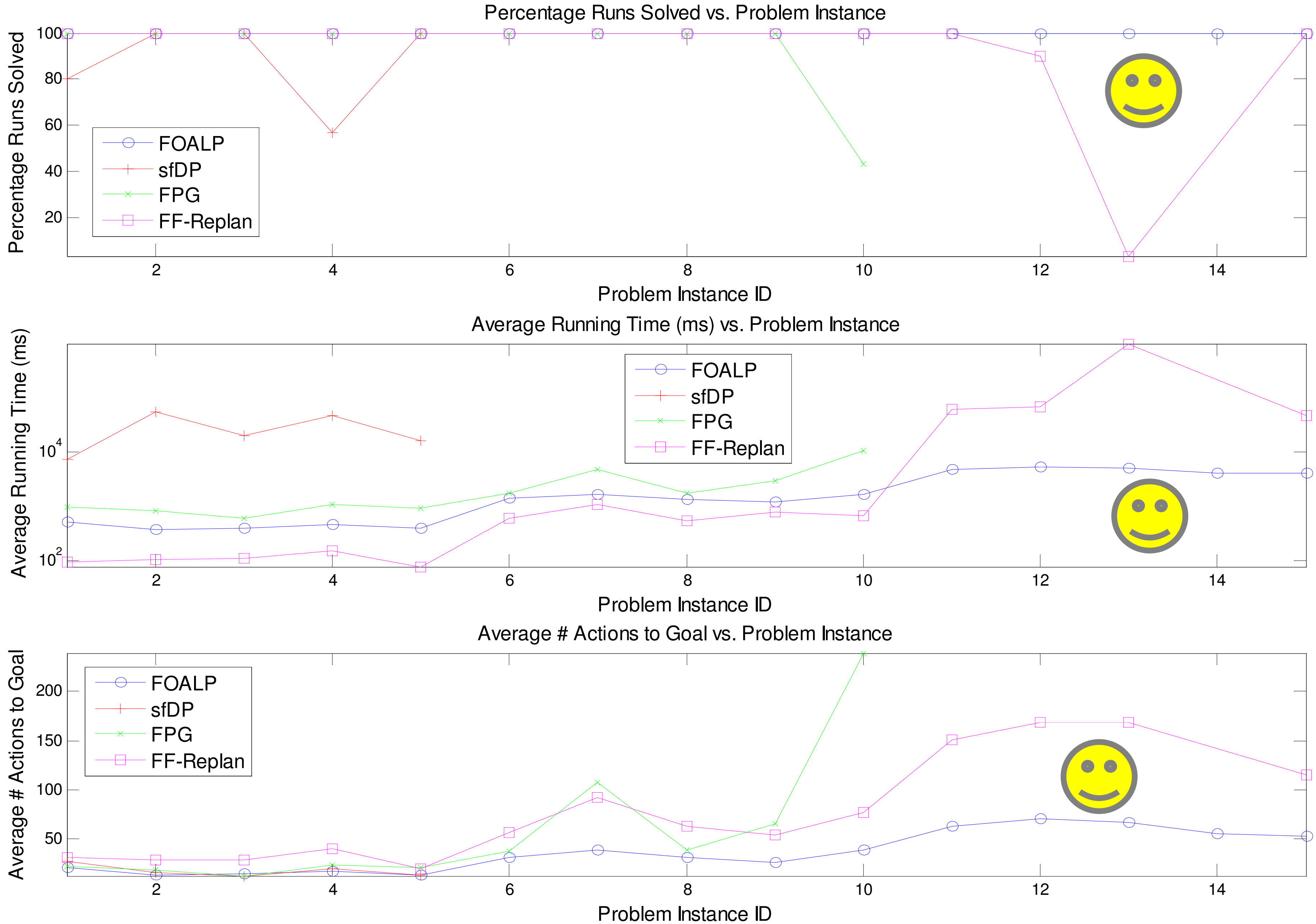
$$0 \geq \max_s \left(\begin{array}{|ll|}\hline \{\neg Dst(b,c) \vee BoxIn(b,c,s)\} & : 10 \\ \hline \{Dst(c_1,c_2), \neg BoxIn(c_1,c_2,s)\} : & 0 \\ \hline \end{array} \oplus \begin{array}{|ll|}\hline \{Dst(c_3,c_4), \neg BoxIn(c_3,c_4,s)\} : & 2 \\ \hline \{\neg Dst(b,c) \vee BoxIn(b,c,s)\} & : -2 \\ \hline \end{array} \oplus \begin{array}{|ll|}\hline \{TruckIn(c_5,c_6,s)\} & : 1 \\ \hline \{\neg TruckIn(t,c,s)\} & : 0 \\ \hline \end{array} \right)$$

$$0 \geq \max_s \left(\begin{array}{|ll|}\hline \{\cancel{\neg Dst(b,c) \vee BoxIn(b,c,s)}, Dst(c_3,c_4), \cancel{\neg BoxIn(c_3,c_4,s)}, \neg Dst(c_3,c_4), \emptyset \ \} : & 12 \\ \hline \{\ \cancel{\neg Dst(b,c) \vee BoxIn(b,c,s)}\} & : \ 8 \\ \hline \{Dst(c_1,c_2), Dst(c_3,c_4), \cancel{\neg BoxIn(c_1,c_2,s)}, \cancel{\neg BoxIn(c_3,c_4,s)}\ \} & : \ 2 \\ \hline \{\cancel{\neg Dst(b,c) \vee BoxIn(b,c,s)}, Dst(c_1,c_2), \cancel{\neg BoxIn(c_1,c_2,s)}, \emptyset \ \} & : -2 \\ \hline \end{array} \oplus \begin{array}{|ll|}\hline \{TruckIn(c_5,c_6,s)\} & : 1 \\ \hline \{\neg TruckIn(t,c,s)\} & : 0 \\ \hline \end{array} \right)$$

$$0 \geq \max_s \left(\boxed{\{\ \} : 8} \oplus \begin{array}{|ll|}\hline \{TruckIn(c_5,c_6,s)\} & : 1 \\ \hline \{\neg TruckIn(t,c,s)\} & : 0 \\ \hline \end{array} \right)$$

$$0 \geq 10 + -w_1 + w_2$$

International Prob. Planning Comp.: FOALP 2nd



First-order MDPs

Related Work & Remarks

Scott Sanner

NICTA

THE AUSTRALIAN NATIONAL UNIVERSITY

Related Purely Deductive Approaches

- Value Iteration:
  - **ReBel algorithm** *(Kersting, van Otterlo, De Raedt, ICML-04)*
  - **FOVIA algorithm for fluent calculus** *(Karabaev & Skvortsova, UAI-05)*
  - **First-order decision diagrams (FODDs)** *(Wang, Joshi, Khardon, IJCAI-07; JK, ICAPS-08; WJK, JAIR-08)*

Computationally attractive for FOMDP subset

Elegant FO-DD theory

- Approximate Linear Programming (ALP)
  - **First-order ALP (FOALP)** *(Sanner & Boutilier, UAI-05; SB, AIJ-08)*

Introduces first-order LP, AIJ article best reference

- Policy Iteration
  - **Approximate policy iteration (FOAPI)** *(Sanner & Boutilier, UAI-06)*
  - **Modified policy iteration with FODDs** *(Wang, Joshi, & Khardon, UAI-07)*
- Factored FOMDPs – FOMDP extension
  - **Factored SDP and Factored FOALP** *(Sanner & Boutilier, ICAPS-07)*

FOMDP subsumes all previous representations;

But FOMDP not enough for PPDDL, need at least *factored FOMDP*

Related Inductive Appoaches

First-order inductive MDP approaches

- Relational RL work by *Driessens, Dzeroski, De Raedt*
- Numerous works by *Yoon, Fern, and Givan*
- UAI-04 paper by *Gretton and Thiebaux*
- Recent ICML-08 non-parametric policy gradient work by *Kersting and Driessens*

- **Requires simulating grounded MDPs**
- **No performance bounds for *all* ground MDPs**

Induction vs. Analytical Derivation

- The average human has 1 testicle

- In an inductive setting, need to ensure you have the right hypothesis space (and inductive bias) for generalization

- Not an issue for SDP / FOMDPs
  - Guaranteed to **derive** ε-optimal lifted policy

FOMDP Conclusions

FOMDPs are lifted MDP model

– Use case notation and regression
– Symbolic dynamic programming = lifted DP

– **Exploit state & action abstraction for MDPs**
– **Exact or approximate bounds for *all* ground MDPs**

Factored FOMDP

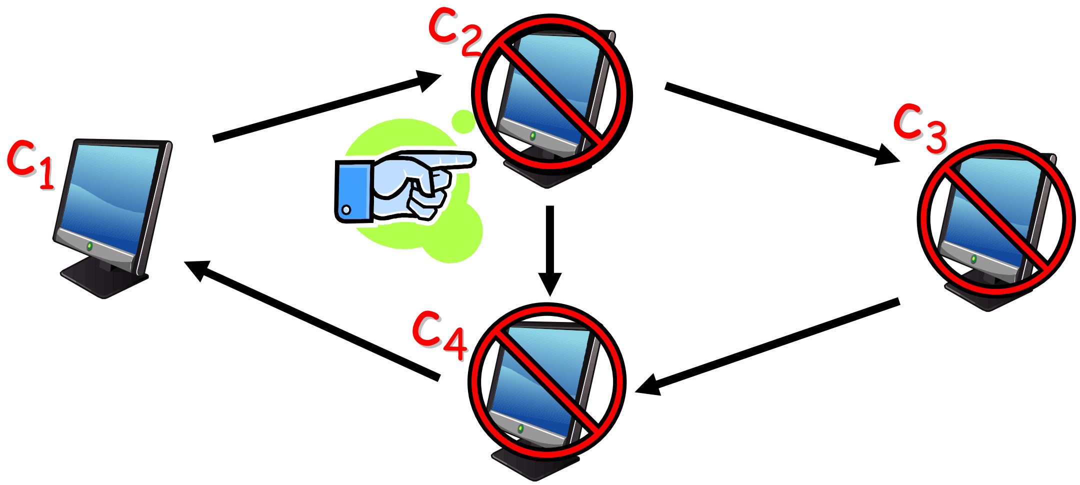
Scott Sanner

NICTA

THE AUSTRALIAN NATIONAL UNIVERSITY

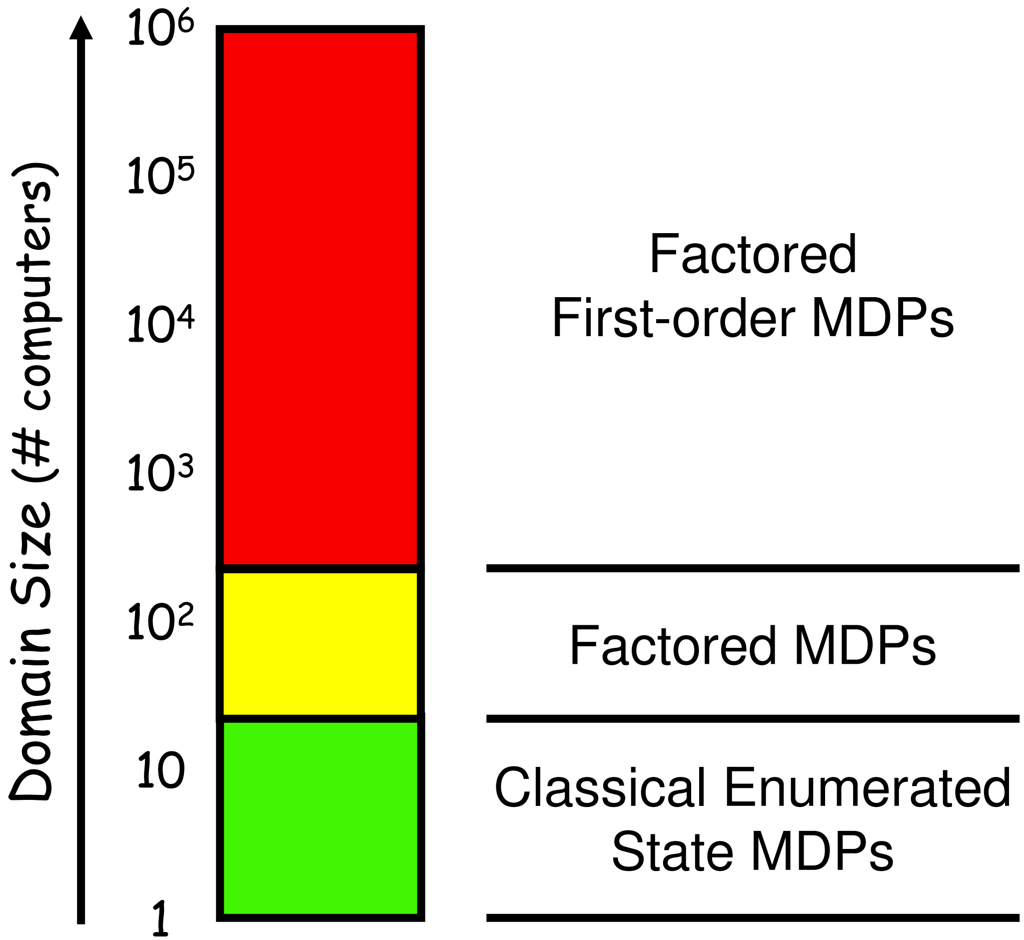
Motivating Example: SysAdmin MDP

- Have n computers $C = \{c_1, \ldots, c_n\}$ in a network
- State: each computer c_i is either "up" or "down"
- Transition: computer is "up" proportional to its state and # upstream connections that are "up"



- Action: manually reboot one computer
- Reward: +1 for every "up" computer

How to Solve SysAdmin?



- First-order MDPs cannot represent SysAdmin domain independently

  ⇒ need factored first-order MDP!
- State-of-the-art (factored) MDP solutions can scale only to ~140 computers

Factored FOMDPs: Additive Reward

- **SysAdmin reward scales with domain size:**

$$rCase(s) = \begin{array}{|l|c|}\hline Run(c_1,s) & 1 \\ \hline \neg\, Run(c_1,s) & 0 \\ \hline \end{array} \oplus \cdots \oplus \begin{array}{|l|c|}\hline Run(c_n,s) & 1 \\ \hline \neg\, Run(c_n,s) & 0 \\ \hline \end{array}$$

- **Beyond expressive power of current FOMDP**

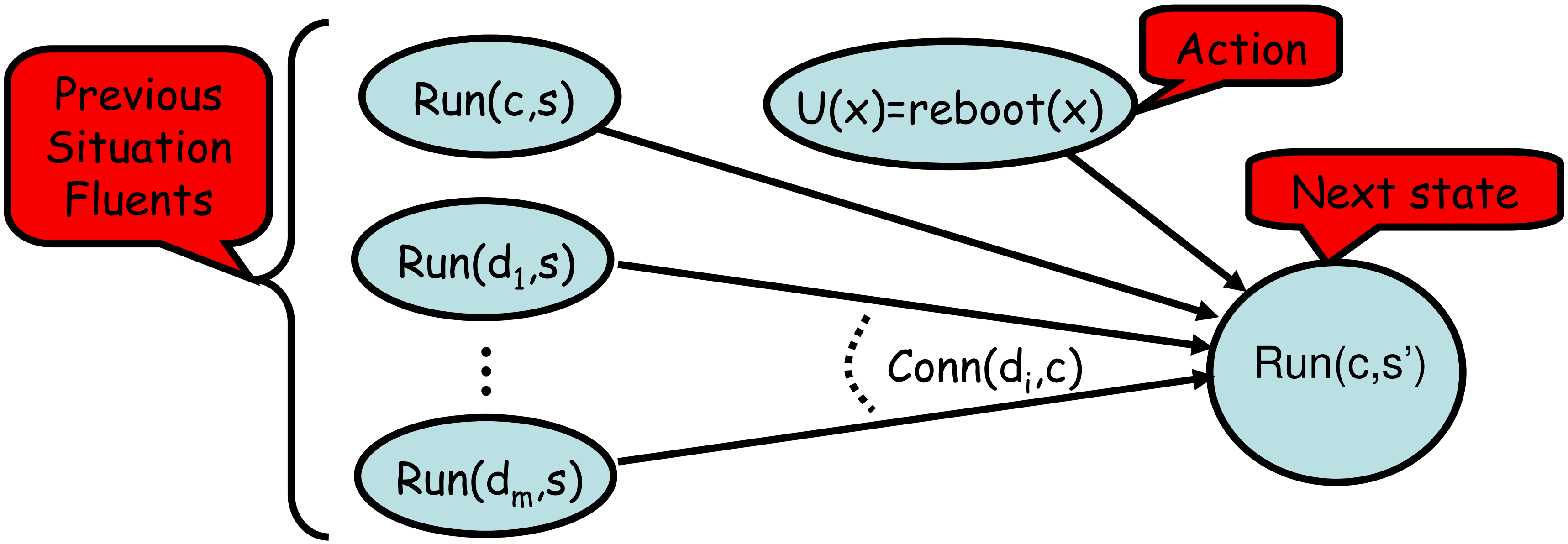
- **Need language extension for $\sum$ aggregator:**

$$rCase(s) = \sum_{c \in C} \begin{array}{|l|c|}\hline Run(c,s) & 1 \\ \hline \neg\, Run(c,s) & 0 \\ \hline \end{array}$$

  – Semantics is just the expanded $\oplus$

Factored FOMDPs: Factored Transition

- Need a relational DBN



- Need a joint distribution over indefinite # objects

$P(\text{Run}(c_1,s'),\ldots,\text{Run}(c_n,s') \mid \text{reboot}(x)) = \prod_{c\in C}$

$x=c$	1
$x\neq c \wedge \text{Run}(c)$	.95
$x\neq c \wedge \neg\,\text{Run}(c)$	.05

Factored FOMDP Solution

- A FOMDP with indefinite sums and products of case statements
  - More expressive formalism than FOPI / FOVE because case statements can be quantified
- For FO-ALP for SysAdmin, introduced two new elimination techniques for constraint generation
  - Linear elimination
  - Existential elimination
    ➔ log(n) computation of SysAdmin ALP solution
  - See ICAPS-07 / my thesis for details

Existential Elimination

- **Need to compute:** max $\exists x\ \Sigma_c$ [case (c, x)]
  - where case(c, x) = $\begin{array}{|l|r|}\hline x = c & 10 \\ \hline x \neq c \wedge \ldots & 9 \\ \hline x \neq c \wedge \ldots & 0 \\ \hline \end{array}$
- **Introduce:**
  - $\sum_c eCase(c,s) = \sum_c \begin{array}{|ll|}\hline b(c) \supset b(next(c)) & : \quad 0 \\ \hline b(c) \wedge \neg b(next(c)) & : \ -\infty \\ \hline \end{array}$
  - $b(c_1)$, $b(c_2)$, b(c3), …, $b(c_{n-1})$, $b(c_n)$
  - $\perp$ $\perp$ T T T T
- **Replace:** $(x = c) \equiv \neg b(c) \wedge b(next(c))$
- **Final constraint:**

$$0 \geq \max_s \sum_c \left[case_1(c,s) \oplus .. \oplus case_p(c,s) \oplus eCase(c,s)\right]$$

Linear Elimination

$c_1 \longrightarrow \cdots \; c_i \; \cdots \longrightarrow c_n$

- **Need to compute:** $r(n) = \max_{c2 \ldots cn} \Sigma_{i=1 \ldots n}\, case(c_i, c_{i+1})$
  - where $case(c_i, c_{i+1}, s) =$

c_i	c_{i+1}	
$\bot$	$\bot$	1
$\bot$	$\top$	-5
$\top$	$\bot$	-5
$\top$	$\top$	0

  - $r(2) = \max\, c_2$

c_1	c_2	
$\bot$	$\bot$	1
$\bot$	$\top$	-5
$\top$	$\bot$	-5
$\top$	$\top$	0

+

c_2	c_3	
$\bot$	$\bot$	1
$\bot$	$\top$	-5
$\top$	$\bot$	-5
$\top$	$\top$	0

=

c_1	c_3	
$\bot$	$\bot$	2
$\bot$	$\top$	-4
$\top$	$\bot$	-4
$\top$	$\top$	0

  - $r(4) = \max\, c_2, c_3, c_4$

c_1	c_3	
$\bot$	$\bot$	2
$\bot$	$\top$	-4
$\top$	$\bot$	-4
$\top$	$\top$	0

+

c_3	c_5	
$\bot$	$\bot$	2
$\bot$	$\top$	-4
$\top$	$\bot$	-4
$\top$	$\top$	0

=

c_1	c_5	
$\bot$	$\bot$	4
$\bot$	$\top$	-2
$\top$	$\bot$	-2
$\top$	$\top$	0

- **Computation of** $r(n)$ **takes** $O(\log(n))$ **!**

Other Factored FOMDPs: Logistics

- Boxes fall off trucks with probability .1

vCase(s) =

$\exists b. BoxIn(b, paris, s)$	$: 10$
$\neg$“ $\wedge \{\exists b. [(t = t^* \wedge BoxOn(b, t^*, s) \wedge TruckIn(t^*, paris, s))$ $\vee (\exists t. BoxOn(b, t, s) \wedge TruckIn(t, paris, s)) \vee TruckIn(t, paris, s)]\}$	$: 10 - p$
$\neg$“ $\wedge \exists b. [t = t^* \wedge BoxOn(b, t^*, s) \wedge TruckIn(t^*, paris, s)]$	$: 9$
$\neg$“ $\wedge \exists b. [\exists t. BoxOn(b, t, s) \wedge TruckIn(t, paris, s)]$	$: 10 - 10p$
$\neg$“	$: 0$

$$p = 0.9^{|\{\langle b_i, t_j \rangle | b_i \in Box \wedge b_i \neq b^*. \, BoxOn(b_i, t_j, s) \wedge TruckIn(t_j, paris, s)\}|}$$

- Compact factored FOMDP case statements require symbolic values
  - One of original intentions of SDP

First-order POMDPs

A Proposal

Scott Sanner

NICTA

ANU

THE AUSTRALIAN NATIONAL UNIVERSITY

Kristian Kersting

IAIS

Fraunhofer

Institut
Intelligente Analyse- und
Informationssysteme

Building on previous work by Wang, Khardon and Schmolze

Goals of FO-POMDPs

- Advantages of FOMDP
  - **Exploit state & action abstraction for MDPs**
  - **Exact or approximate bounds for *all* ground MDPs**
- Additional advantage of FO-POMDP
  - **∞ relational observation space**
  - **Solution derives only the *relevant* observations**

Why are FO-POMDPs better?

- One DP Backup in "flat" POMDPs:

$$O(|A||\Gamma|^{|Z|})$$

- A *large observation space* *|Z|* *will kill you*
  - Factored POMDPs attack this problem
  - But cannot handle ∞ relational spaces

FO-POMDP Value and Belief State Representation

$$V^0(s) = R(s) = \boxed{\begin{array}{l|r} \exists b. BoxIn(b, paris, s) & : 10 \\ \hline \neg\exists b. BoxIn(b, paris, s) & : 0 \end{array}} \bullet \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}$$

$$V^1(s) = \boxed{\begin{array}{lr} \exists b. BoxIn(b, paris, s) & : 19.0 \\ \hline \neg\text{``} \wedge \exists b, t. TruckIn(t, paris, s) \wedge BoxOn(b, t, s) & : 8.1 \\ \hline \neg\text{``} & : 0.0 \end{array}} \bullet \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix}$$

$$V^2(s) = \boxed{\begin{array}{lr} \exists b. BoxIn(b, paris, s) & : 26.1 \\ \hline \neg\text{``} \wedge \exists b, t. TruckIn(t, paris, s) \wedge BoxOn(b, t, s) & : 15.4 \\ \hline \neg\text{``} \wedge \exists b, c, t. BoxOn(b, t, s) \wedge TruckIn(t, c, s) & : 7.3 \\ \hline \neg\text{``} & : 0.0 \end{array}} \bullet \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{pmatrix}$$

Key Assumption

- Have *disjoint* **state** and **observation** relations
  - State relations known, but hidden
  - Observation relations fully observed
  - **Connection** is **stochastic observation** actions:
    - Break down into deterministic observation actions
    - Probability distribution over actions

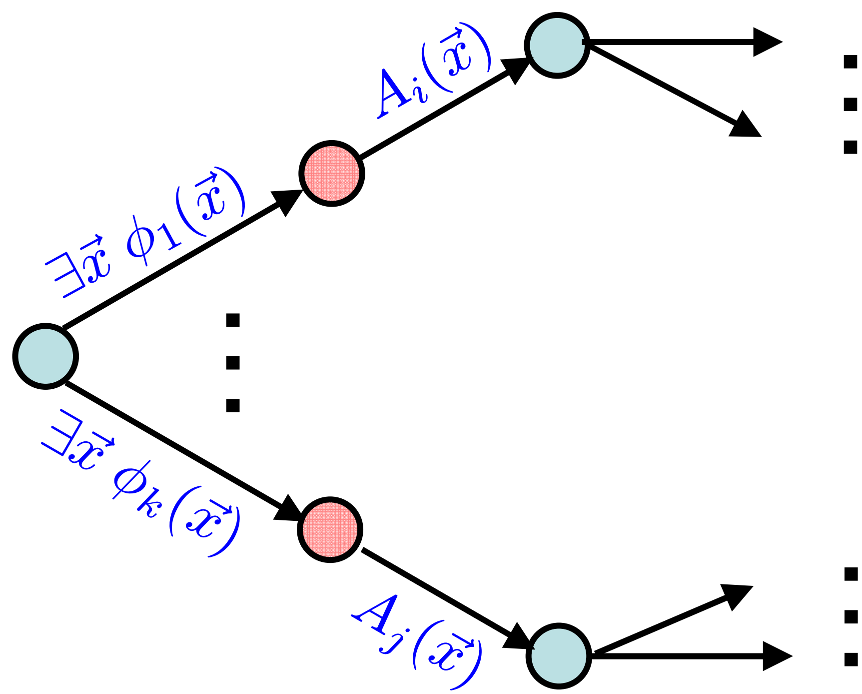
Derive relevant observations for a state partition.

- Objects (terms) fully observed
  - Object identity ambiguity handled by uncertainty on term equality relation

FO-POMDP Policy Tree

- Each α-vector corresponds to FO-strategy
  - Observations at ◯ *partition observation space*
  - Actions at ◯ are *derived from observations*

$$\alpha(s) = \begin{array}{|c|}\hline \psi_1(s) : 10 \\ \hline \psi_2(s) : \;\; 0 \\ \hline\end{array} \cdot \begin{bmatrix} p_1 \\ p_2 \end{bmatrix}$$



Summary

- FOMDPs
  - Basic representation and SDP algorithm
  - Caveats and workarounds
- Extensions
  - Factored FOMDP
  - FO-POMDP
  - No Factored FO-POMDP yet ☺

Take-home Message

- Relational languages compactly capture many sequential decision-making models
  - Rewards
  - Stochastic action theories
- If you have a model (or can learn it)
  - Analytically derive domain-independent solutions (or approximations thereof)
  - Can inform relational RL
  - Exchange techniques with (extensions of) FOPI / FOVE