Future Directions for First-Order Decision-Theoretic Planning

Research Proposal

Scott Sanner

ssanner@cs.toronto.edu

MDP Overview

- **MDPs** are ***de facto*** **standard model** for **decision-theoretic planning** problems
- But, traditional **enum. state models** are **inadequate** for **representation / inference**
- Thus, **MDP research** has focused on:
  - **Algorithms** that **exploit MDP structure**
  - **MDP language extensions** for succinct models

FOMDP Overview

- Addressing both issues, **first-order MDPs (FOMDPs) introduced** (BRP, 2001)
- Allows **relational MDPs (RMDPs)** to be **solved independently of ground domain**
- But, this **level of abstraction** has its **costs:**
  - **Theorem proving** required for **compactness**
  - **No upper bound** on **optimal value fn size!**

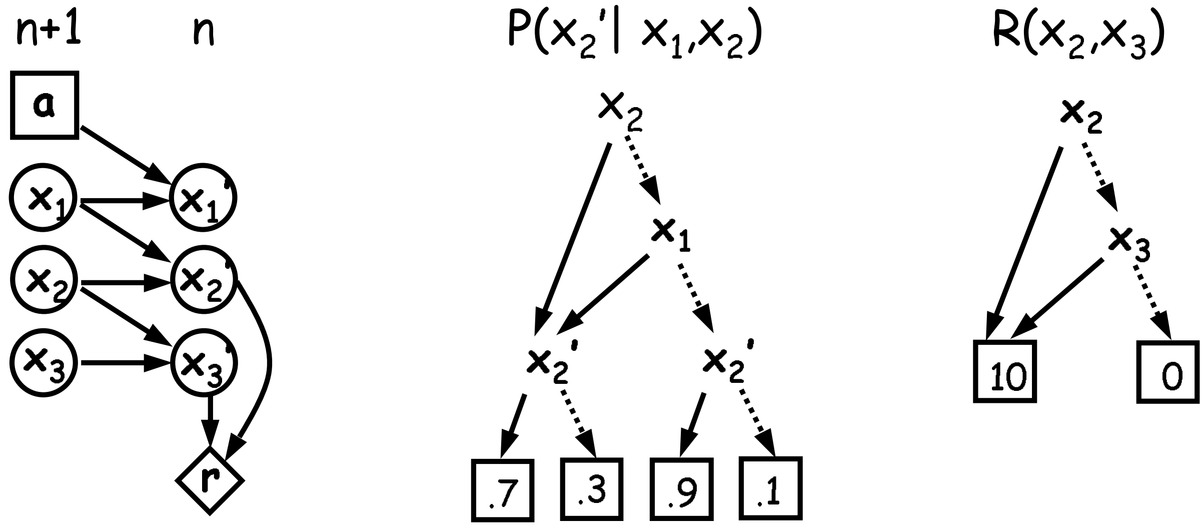
Current and Future Directions

More research needed to make **MDPs** and **FOMDPs practical** for **realistic applications:**

- **Structure exploitation in algorithms**:
  - Exploiting **structure for exact/approx. solutions**
  - Exploiting **structure** in **basis function approaches**
- **Modeling language extensions**:
  - Sum/count aggregators
  - Explicit quantity
  - Topological structure
  - Program constraints
  - Concurrent actions

1a) Exploiting CSI in Factored MDPs

- Use **ADDs** to **exploit CSI** in **factored MDP** model:



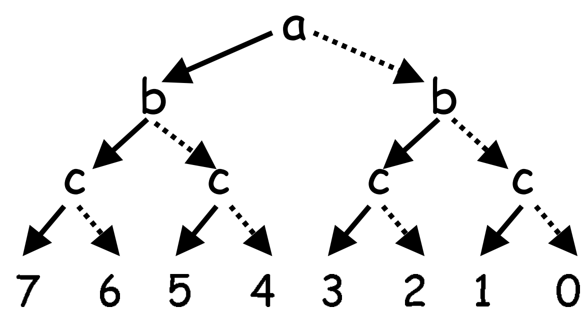
- **Value iteration (VI)** for **factored MDPs**:
  - $V^{n+1}(x_1 \ldots x_i) = R(x_1 \ldots x_i) + \gamma \cdot \max_a \sum_{x1' \ldots xi'} \prod_{F1 \ldots Fi} P_1(x_1' |_{\ldots} a) \ldots Pi(x_i' |_{\ldots} a)\, V^n(x_1' \ldots x_i')$

BKGD

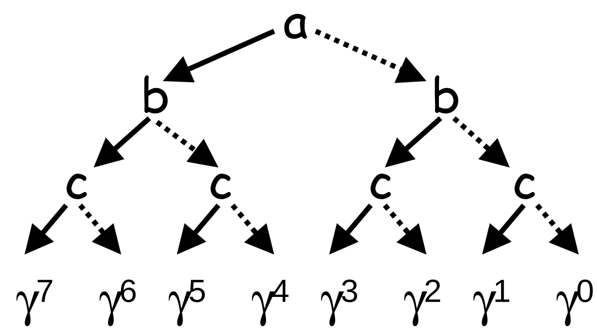
- **SPUDD** (HSHB, 1999): **ADD-based VI**

1a) Is CSI enough for MDPs?

- **ADDs** exploit **CSI**, but more structure beyond CSI
- **Example 1: Additive** reward/utility **functions**
  - $R(a,b,c) = R(a) + R(b) + R(c)$
    $= 4a + 2b + c$



- **Example 2: Multiplicative** value **functions**
  - $V(a,b,c) = V(a) \cdot V(b) \cdot V(c)$
    $= \gamma^{(4a + 2b + c)}$

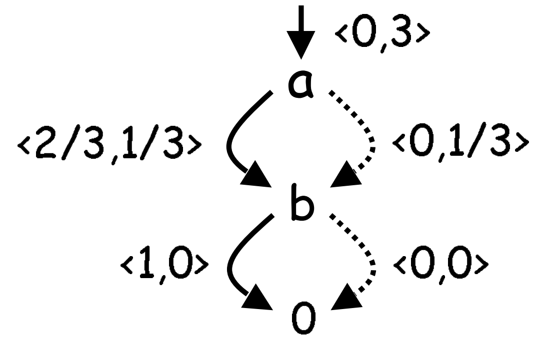


1a) Exploiting CSI/Add/Mult in MDPs

PREV

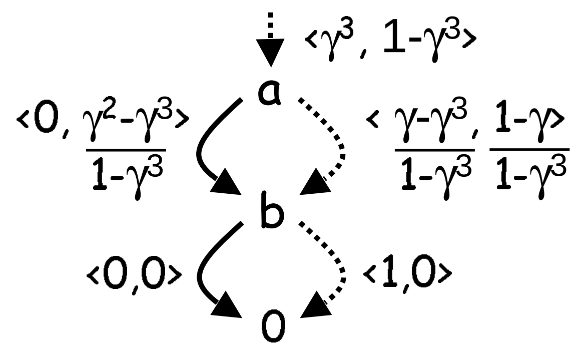
- **Replace ADDs** with **Affine ADDs** (SM, 2005)
- **Example 1: Additive** reward/utility **functions**

  - $R(a,b) = R(a) + R(b)$
    $= 2a + b$



- **Example 2: Multiplicative** value **functions**

  - $V(a,b) = V(a) \cdot V(b)$
    $= \gamma^{(2a + b)};\ \gamma < 1$



- Up to **exp→lin time/space reduct., never worse**!

1a) Exploiting Prop. Structure in FOMDPs

- **FOMDP operations** use **case statements**, e.g.

$\exists x\, A(x)$	10
$\neg \exists x\, A(x)$	20

$\oplus$

$\exists y\, A(y)$	1
$\neg \exists y\, A(y)$	2

=

$\exists x\, A(x) \wedge \exists y\, A(y)$	11
~~$\exists x\, A(x) \wedge \neg \exists y\, A(y)$~~	~~12~~
~~$\neg \exists x\, A(x) \wedge \exists y\, A(y)$~~	~~21~~
$\neg \exists x\, A(x) \wedge \neg \exists y\, A(y)$	22

- **Problem: Case ops** yield **redundant formulae**

CURR

- **Solution: Extract prop. struct. & simplify**, e.g.

Prop Var	FOL Mapping
a	$\exists x\, A(x)$
b	$\exists x\, B(x)$

➔

a	10
$\neg a$	20

$\oplus$

a	1
$\neg a$	2

=

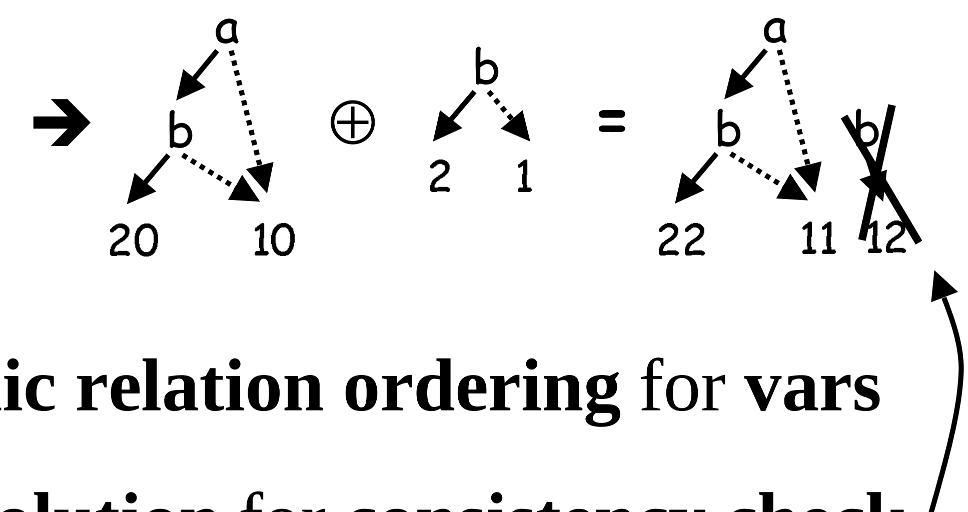
a	11
$\neg a$	22

1a) Exploiting CSI/Add/Mult in FOMDPs

CURR

- **Propositional mapping** also enables **extension of case statements** to **first-order (affine) ADDs**

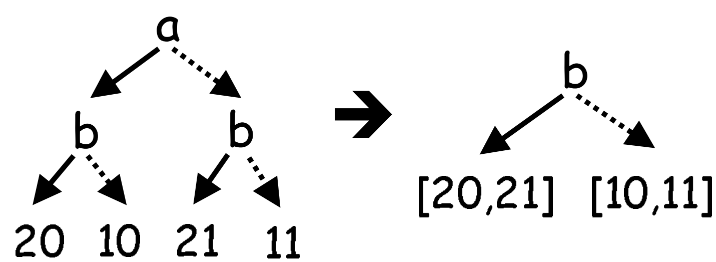
Prop Var	FOL Mapping
a	$\exists x\ A(x)$
b	$\exists x\ A(x) \wedge B(x)$



- Use **lexicographic relation ordering** for **vars**
- **Use ordered resolution** for **consistency check**
- **Replace FOMDP case and ops** with **FO(A)ADD** $\Rightarrow$ **exploit logical, add,** and **mult structure!**

1a) Structured Approximation Solutions

PREV
- **APRICODD** (SHB, 2000): **Approx. VI w/ ADDs**
  - At each VI step, prune value fn & replace w/ range
  - Err. contracts on VI
  - Can still converge!



FUTR
- Extend **APRICODD** to **AADDs** for **MDPs**
  - Prune nodes that minimize $\max(\ |F(v, \mathbf{X}) - F(\neg v, \mathbf{X})|\ )$
  - Can perform explicit merges in node cache, or reduce precision at terminal (more difficult for AADDs)

FUTR
- Extend **APRICODD** to **FO(A)ADDs** for **FOMDPs**
  - Direct extension, or can we exploit structure better?

1b) Structured FO Basis Fn Solutions

- Represent **value fn** as **linear comb.** of **basis fns:**

$V(s) = w_1 \cdot$

∃b,c BIn(b,c,s)	1
¬ ∃b,c BIn(b,c,s)	0

$\oplus\ w_2 \cdot$

∃t,c TIn(t,c,s)	1
¬ ∃t,c TIn(t,c,s)	0

- Reduces **MDP solution** to **finding good weights**

PREV

- **FOALP** (SB, 2005): **Approx. LP for FOMDPs**
  - Formulate as **optimization of LP** w/ **FO constraints**
  - Use a **relational variant** of **var elim** to **efficiently find max violated constraint** for **constraint generation**
  - **Projection** of **value fn** onto **weights obviates need** for **simplification**, only need to do **consistency checking!**

1b) More FO Basis Fn Research

FUTR

- **FO Approximate Policy Iteration (FOAPI):**
  - **API** typically yields **lower error** than **ALP**
  - **Generalize API error bounds** to **FOAPI:**
    - API has much tighter err. bounds than ALP!

FUTR

- **Additional research** for **FOALP / FOAPI:**
  - Use of **FO(A)ADD data structures**
  - Can we **automatically generate basis fns?**
  - **Techniques** for **reducing approx. error:**
    - Partition relevance reweighting (FOALP)
    - Bellman error-directed on-line search

2a) Sum/Count Aggregators

- Often, **reward scales** with **domain size:**

SysAdmin Domain: $R(s) = \Sigma_c$

running(c,s)	1
¬ running(c,s)	0

- **Cannot repr.** in current **FOMDP formalism!**
- Need **sum/count aggregator language extension**

CURR

- **One solution approach: extension of FOALP**
  - **Basis fns** w/ **aggregators scale** w/ **domain size**
  - **Caveat:** leads to a **FO LP** with ∞ **constraints**
  - But, **solve over-constrained LP,** then **relax active constraints**
  - **Scalable, near-optimal solution** on **SysAdmin**

2b) Explicit Quantity

- Often, we want to **represent quantity explicitly:**
  - hasWater(Tank-A,25), hasMileage(Car-1,12.34)
- Fortunately, **explicit quantity** is **easy to specify** in **first-order action theories**, e.g.
  - hasWater(t,q,do(a,s)) ≡

    hasWater(t,q-y,s) ∧ a=fill(y) ∧ y ≤ 20 ∨

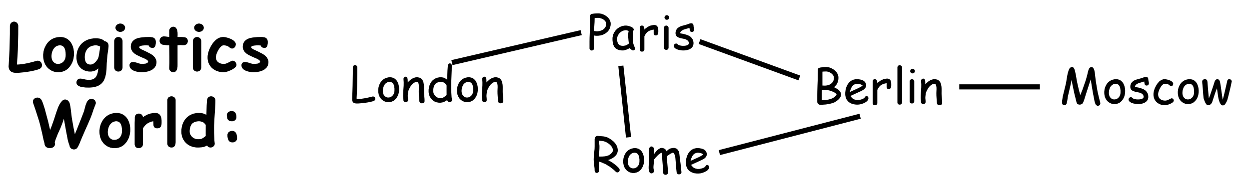
    hasWater(t,q+y,s) ∧ a=drain(y) ∨

    hasWater(t,q,s) ∧ (¬∃y. a=fill(y)∧y≤20)∧¬∃y. a=drain(y)
- Can **apply standard solution techniques** (1a,1b)
- **Problem: simplification/inconsistency** detection with **arithmetic functions & inequalities**
- ⇒ Need to **identify practical inference rules**

FUTR

2c) Topological Structure

- **Many problems have underlying topology**, e.g.



- **Waste of computation** to rely on **MDP inference** to **perform graph-theoretic operations**

FUTR

- Ideally, want to **compile out topological content:**
  - **Precompute stochastic shortest paths** between all node pairs
  - Use a combo of **macro-actions** and **lookup tables** during **regression/max** of actions

2d) Program Constraints

FUTR

- Have **policy constraints** in form of a **program**
- **Goal: make opt. decision** at **non-det. choice pts**
- **Solution: Generalize HAM model** (PR, 1998) to **FOMDPs** with **GOLOG program constraints**

2e) Concurrent Actions

FUTR

- Most **real-world problems** consist of **actions executable in parallel**
- How to **deal** with **action interactions?**
  - **Factored action effects**
  - **Basis function techniques**, e.g. (GKGK, 2003)

Summary of Research Plan

- **Current directions to complete:**
  - (1a) Exact FOMDP solutions with FO(A)ADDs
  - (2a) Sum/Count aggregators
- **Future directions**:
  - (1a) Approx. MDP solutions with AADDs
  - (1a) Approx. FOMDP solutions with FO(A)ADDs
  - (1b) FOAPI and FOALP/FOAPI enhancements
  - (2b) Explicit quantity
  - (2c) Topological structure
  - (2d) Program constraints
  - (2e) Concurrent actions

Bibliography

(BRP, 2001) C. Boutilier, R. Reiter, and B. Price. (2001). *Symbolic Dynamic Programming for First-order MDPs*. IJCAI-01.

(HSHB, 1999) J. Hoey, R. St. Aubin, A. Hu, and C. Boutilier. (1999). *SPUDD: Stochastic Planning using Decision Diagrams*. UAI-99.

(SM, 2005) S. Sanner and D. McAllester. (2005). *Affine Algebraic Decision Diagrams (AADDs) and their Application to Structured Probabilistic Inference*. IJCAI-05.

(SHB, 2000) R. St. Aubin, J. Hoey, and C. Boutilier. (2000). *APRICODD: Approximate Policy Construction using Decision Diagrams*. NIPS-00.

(SB, 2005) S. Sanner and C. Boutilier. (2005). *Approximate Linear Programming for First-order MDPs*. UAI-05.

(PR, 1998) R. Parr and S. Russell. (1998). Reinforcement Learning with Hierarchies of Machines. NIPS-98.

(GKGK, 2003) C. Guestrin, D. Koller, C. Gearhart, and N. Kanodia. (2003). Generalizing Plans to New Environments in Relational MDPs. IJCAI-03.