UAI 2011

Symbolic Dynamic Programming for Discrete and Continuous State MDPs

Scott Sanner

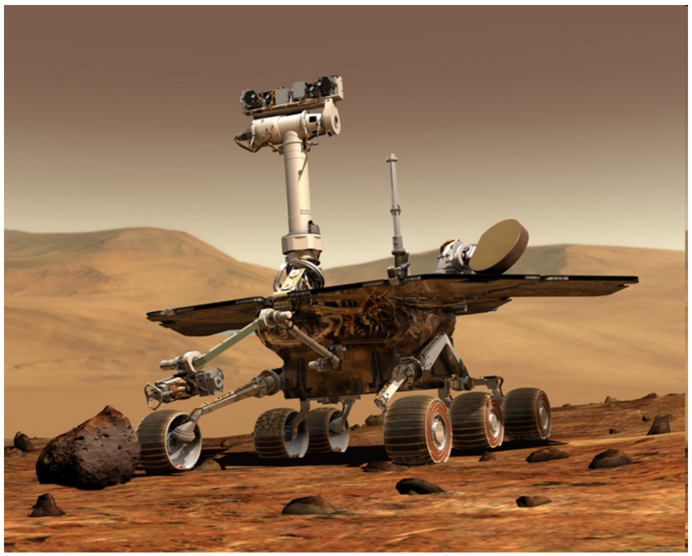
Karina Valdivia Delgado
Leliane Nunes de Barros

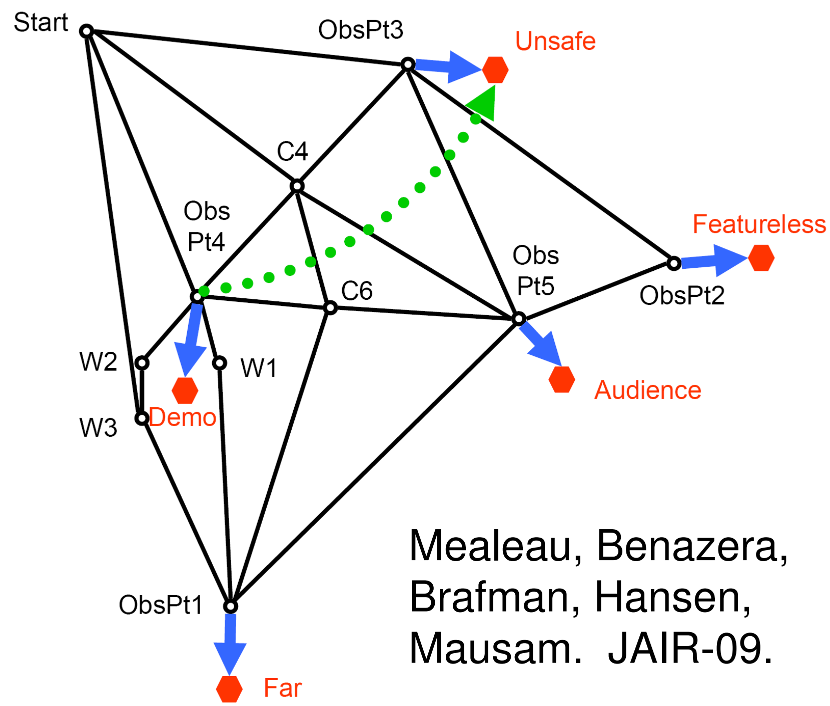
NICTA

THE AUSTRALIAN NATIONAL UNIVERSITY

Universidade de São Paulo

Continuous Domains: Mars Rovers





Mealeau, Benazera, Brafman, Hansen, Mausam. JAIR-09.

- Continuous state
  - Time (t), Energy (e), Robot position (x,y,θ)
- Closed-form exact solution?
  - Currently only if 1D or piecewise rectilinear solution exists ☹

Our work: exact for multidimensional nonlinear domains!

Previous Work

Discrete and Continuous (DC-)MDPs

- Mixed discrete / continuous state

$$(\vec{b}, \vec{x}) = (b_1, \ldots, b_n, x_1, \ldots, x_m) \in \{0,1\}^n \times \mathbb{R}^m$$

- Discrete action set $a \in \mathcal{A}$
- DBN factored transition model

$$P(\vec{b}', \vec{x}' | \vec{b}, \vec{x}, a) =$$

$$\underbrace{\left(\prod_{i=1}^{n} P(b_i' | \vec{b}, \vec{x}, a) \right)}_{\text{discrete}} \underbrace{\left(\prod_{j=1}^{m} P(x_j' | \vec{b}, \vec{b}', \vec{x}, a) \right)}_{continuous}$$

- Arbitrary action-dependent reward

$$R_a(\vec{b}, \vec{x}) = x_1 + x_2$$

Value Iteration for DC-MDPs

- Value of policy in state is expected sum of rewards
- Want optimal value $V^{h,*}$ over horizons $h \in 0..H$
  - Implicitly provides optimal horizon-dependent policy
- Compute inductively via **Value Iteration** for $h \in 0..H$
  - **Regression step:**

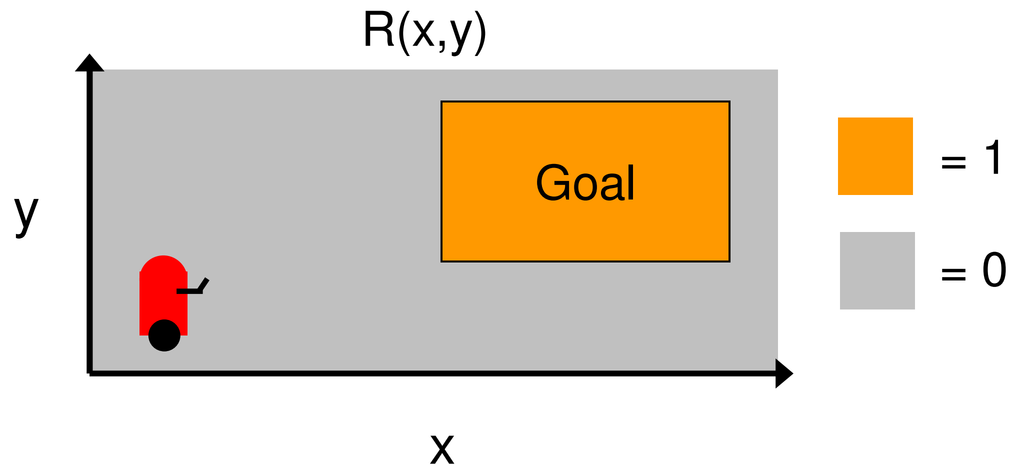
$$Q_a^{h+1}(\vec{b},\vec{x}) = R_a(\vec{b},\vec{x}) + \gamma \cdot$$

$$\sum_{\vec{b}'} \int_{\vec{x}'} \left(\prod_{i=1}^{n} P(b_i'|\vec{b},\vec{x},a) \prod_{j=1}^{m} P(x_j'|\vec{b},\vec{b}',\vec{x},a) \right) V^h(\vec{b}',\vec{x}')d\vec{x}'$$

  - **Maximization step:**

$$V_{h+1} = \max_{a \in A} Q_a^{h+1}(\vec{b},\vec{x})$$

Exact Solutions to DC-MDPs: Domain

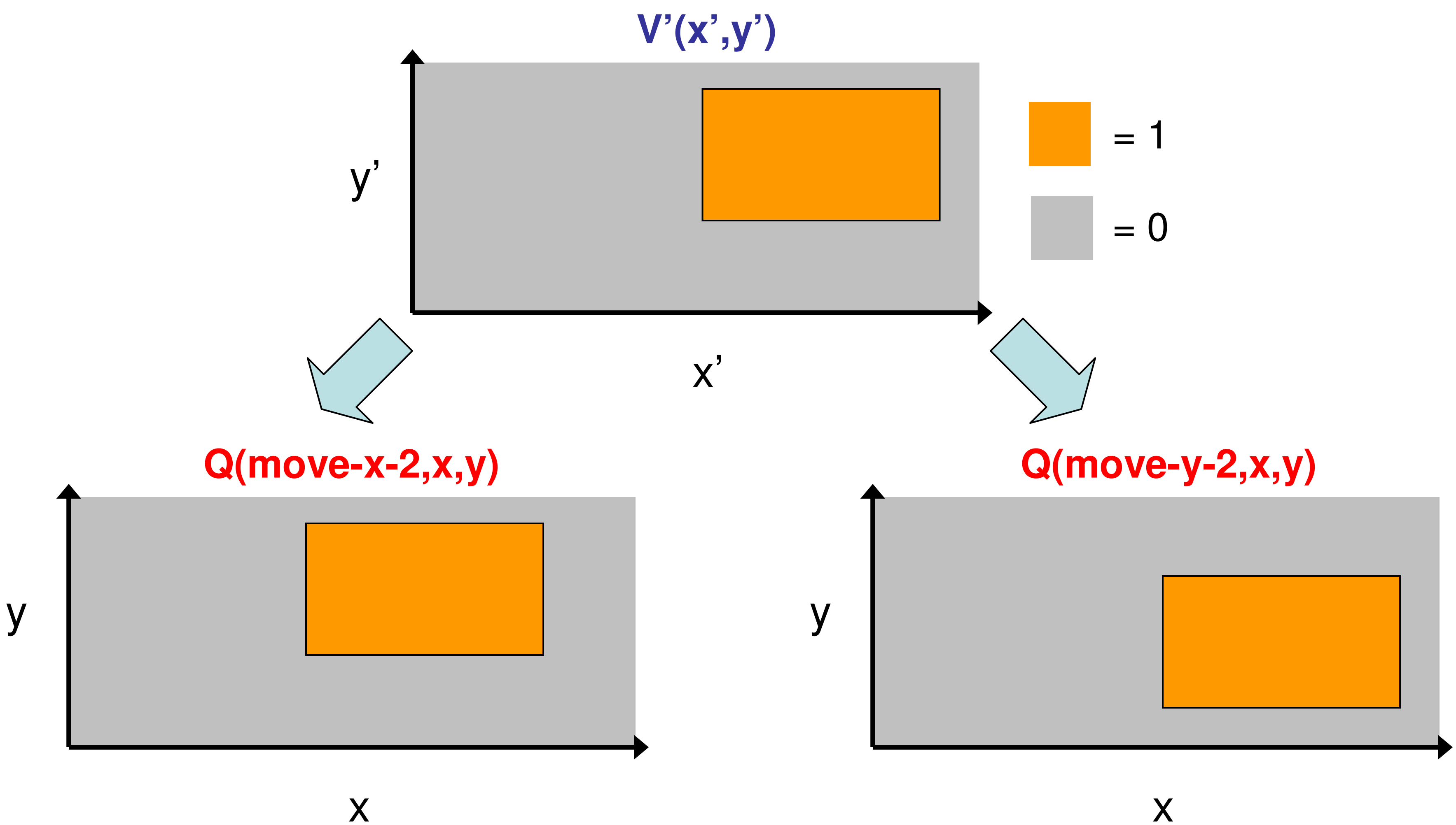


- 2-D Navigation
- State: (x,y) $\in \mathbb{R}^2$
- Actions:
  - move-x-2
    - x' = x + 2
    - y' = y
  - move-y-2
    - x' = x
    - y' = y + 2
- Reward:
  - R(x,y) = I[(x > 5) ^ (x < 10) ^ (y > 2) ^ (y < 5)]

Assumptions:
1. Continuous transitions are deterministic and linear
2. Discrete transitions can be stochastic
3. Reward is piecewise rectilinear

Exact Solutions to DC-MDPs: Regression

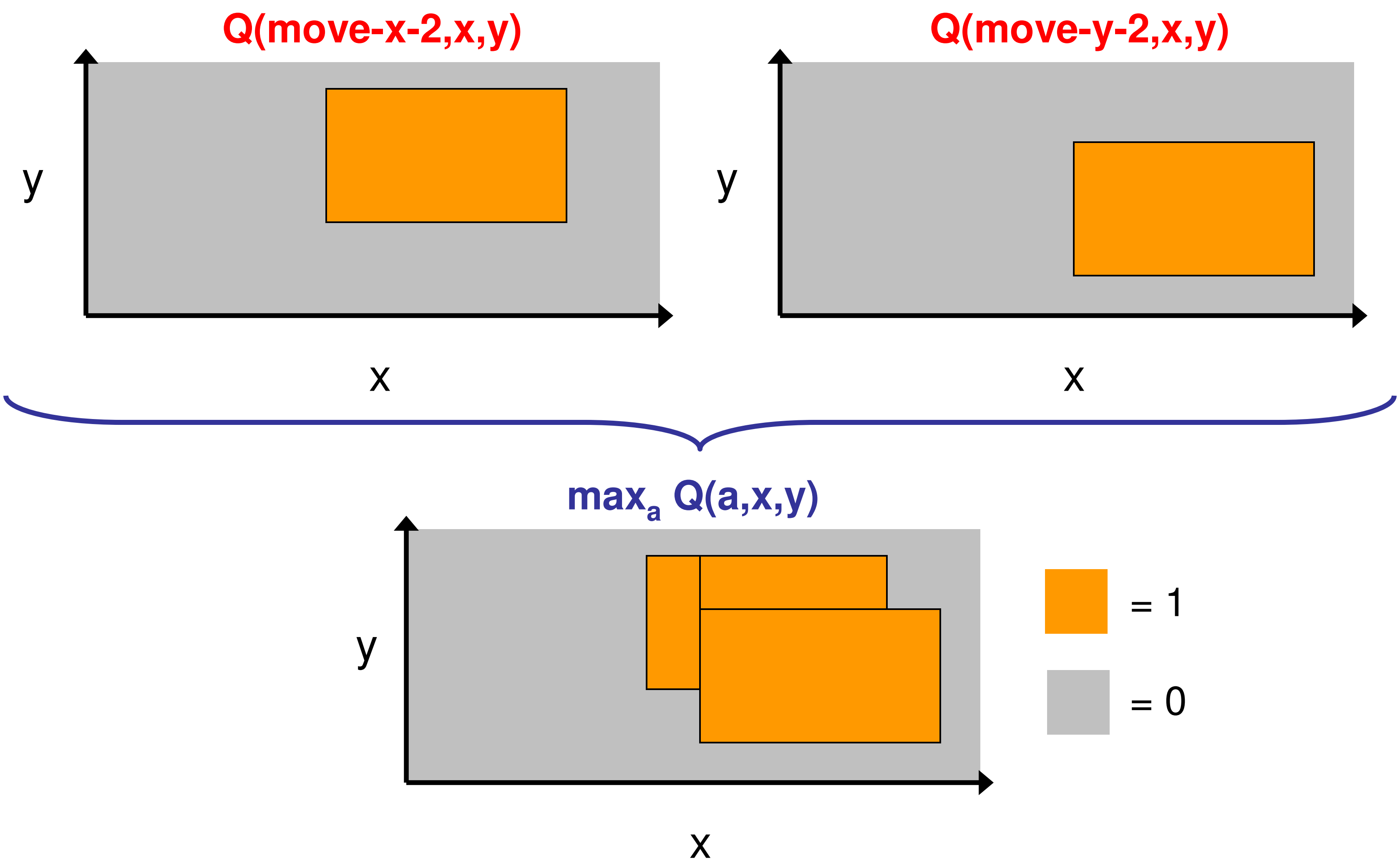
- Continuous regression is just translation of "pieces"



Feng *et al*, UAI-04

Exact Solutions to DC-MDPs: Maximization

- Q-value maximization yields piecewise rectilinear solution



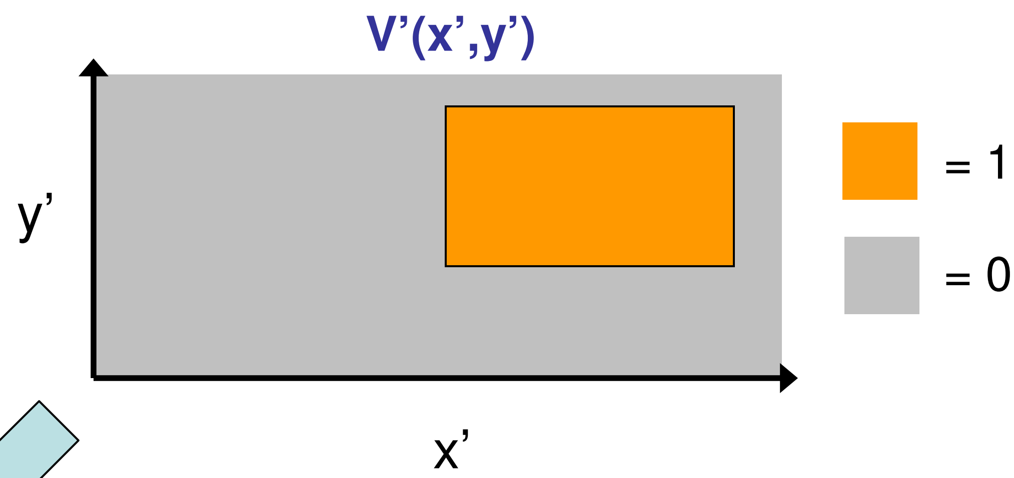
Previous Work Limitations I

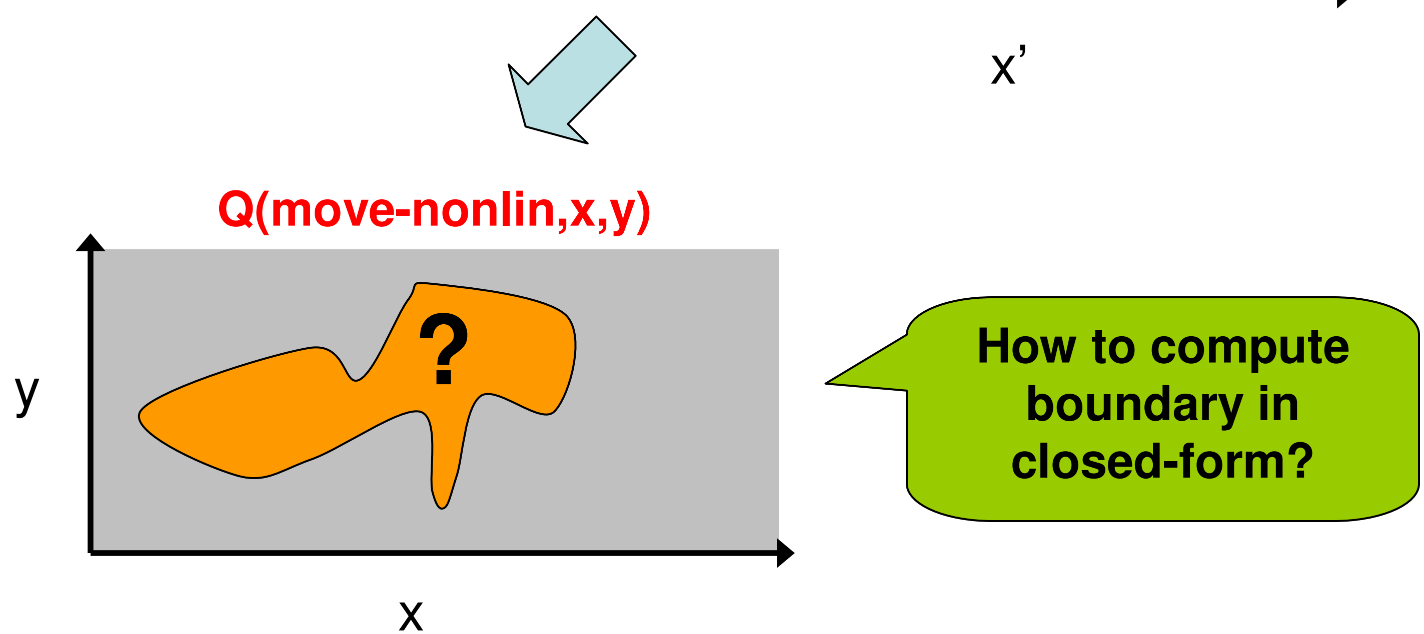
- Exact regression when transitions nonlinear?

Action move-nonlin:

– $x' = x^3y + y^2$

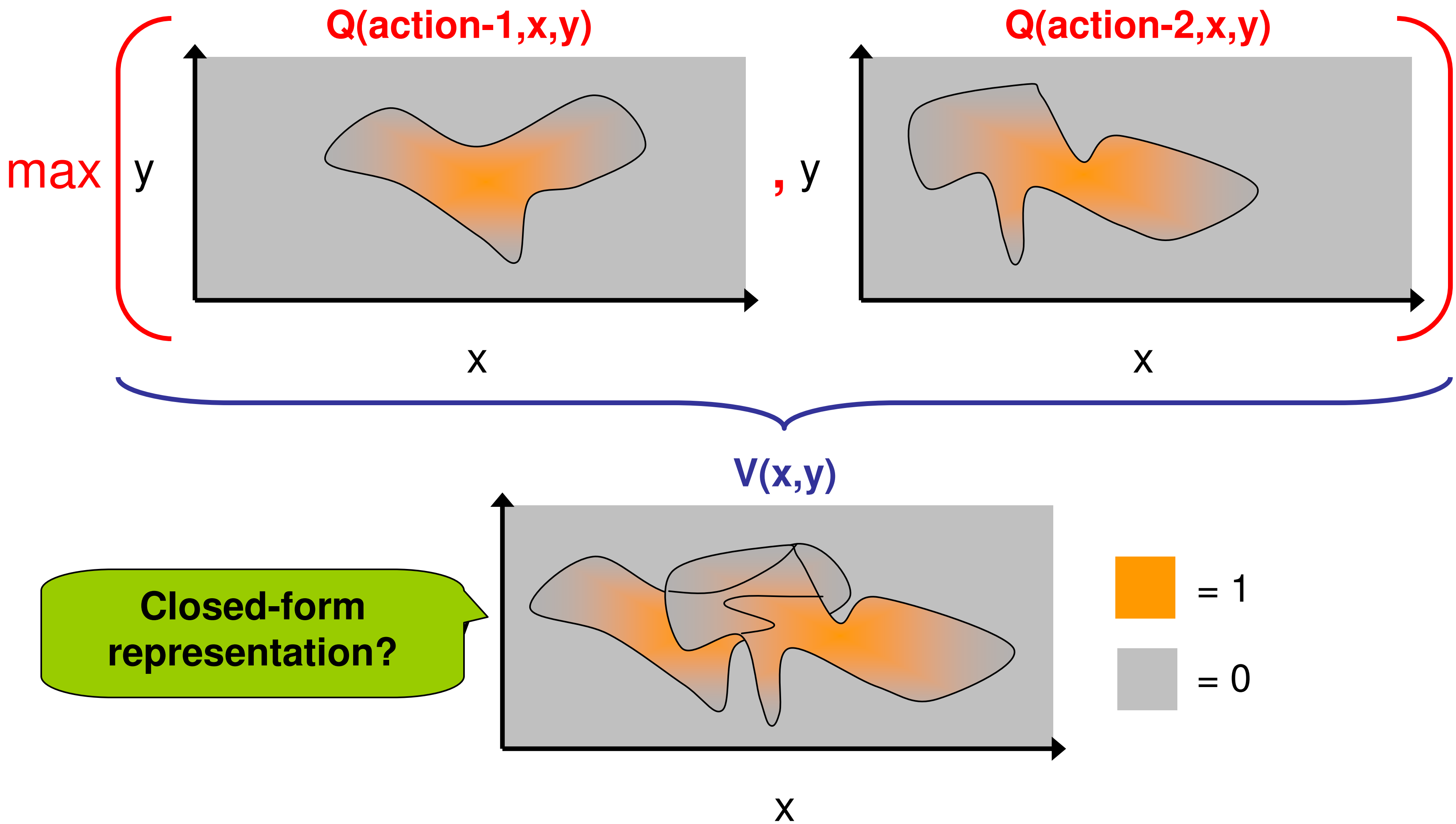
– $y' = y * \log(x^2y)$





Previous Work Limitations II

- max(.,.) when reward/value arbitrary piecewise?



A solution to previous limitations:

Symbolic Dynamic Programming (SDP)

n.b., motivated by SDP from Boutilier *et al* (IJCAI-01) but here continuous instead of relational

SDP uses Symbolic *Case* Representation

$$f = \begin{cases} \phi_1 : & f_1 \\ \vdots & \vdots \\ \phi_k : & f_k \end{cases}$$

$$P(x'|x,y) = \delta\left(x' - \begin{cases} (x < xy^2) \wedge (x > y) : & x + y \\ (x \geq xy^2) \vee (x \leq y) : & (x - y)^2 + 1 \end{cases}\right)$$

Deterministic transitions represented by δ over (conditional) equation

Logical combinations of inequalities of arbitrary expressions

Arbitrary expressions

Case Operations: ⊕, ⊗

$$\begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases} \oplus \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} = \qquad ?$$

Case Operations: ⊕, ⊗

$$\begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases} \oplus \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} = \begin{cases} \phi_1 \wedge \psi_1 : & f_1 + g_1 \\ \phi_1 \wedge \psi_2 : & f_1 + g_2 \\ \phi_2 \wedge \psi_1 : & f_2 + g_1 \\ \phi_2 \wedge \psi_2 : & f_2 + g_2 \end{cases}$$

- Similarly for ⊗
  - Expressions trivially closed under +, *
- What about max?
  - $\max(f_1, g_1)$ not pure arithmetic expression ☹

Case Operations: max

$$\max\left(\begin{cases}\phi_1: & f_1\\ \phi_2: & f_2\end{cases},\begin{cases}\psi_1: & g_1\\ \psi_2: & g_2\end{cases}\right) = \qquad ?$$

Case Operations: max

$$\max\left(\begin{cases}\phi_1: & f_1\\ \phi_2: & f_2\end{cases}, \begin{cases}\psi_1: & g_1\\ \psi_2: & g_2\end{cases}\right) = \begin{cases}\phi_1 \wedge \psi_1 \wedge f_1 > g_1: & f_1\\ \phi_1 \wedge \psi_1 \wedge f_1 \leq g_1: & g_1\\ \phi_1 \wedge \psi_2 \wedge f_1 > g_2: & f_1\\ \phi_1 \wedge \psi_2 \wedge f_1 \leq g_2: & g_2\\ \phi_2 \wedge \psi_1 \wedge f_2 > g_1: & f_2\\ \phi_2 \wedge \psi_1 \wedge f_2 \leq g_1: & g_1\\ \phi_2 \wedge \psi_2 \wedge f_2 > g_2: & f_2\\ \phi_2 \wedge \psi_2 \wedge f_2 \leq g_2: & g_2\end{cases}$$

Key point: still in case form!

Size blowup? We'll get to that…

Symbolic Dynamic Programming

- In a nutshell
  - $R(.)$, $P(.|.)$ defined as case statements
  - Value iteration uses case operations
    - $\oplus$, $\otimes$, max
  - Then provably:
    - $V^h(.)$ is also in case form for all horizons h!
- Only "tricky part" is continuous regression

SDP Regression Step

- **Binary variables** b_i
  - Factored $\sum_{bi \in \{0,1\}}$ (e.g, SPUDD: Hoey *et al*, UAI-99)

- **Continuous variables** x_j
  - $\int_x \delta[x - y] f(x) dx = f(y)$ triggers symbolic *substitution*, so

$$\int_{x_j'} \delta[x_j' - g(\vec{x})] V' dx_j' \; = \; V'\{x_j'/g(\vec{x})\}$$

  - e.g.,

$$\int_{x_1'} \delta[x_1' - (x_1^2 + 1)] \left(\begin{cases} \underline{x_1'} < 2 : & \underline{x_1'} \\ \underline{x_1'} \geq 2 : & \underline{x_1'^2} \end{cases} \right) dx_1' \; = \; \begin{cases} \underline{x_1^2 + 1} < 2 : & \underline{x_1^2 + 1} \\ \underline{x_1^2 + 1} \geq 2 : & \underline{(x_1^2 + 1)^2} \end{cases}$$

  - If g is *case*: need *conditional substitution*, see paper

Case $\rightarrow$ XADD

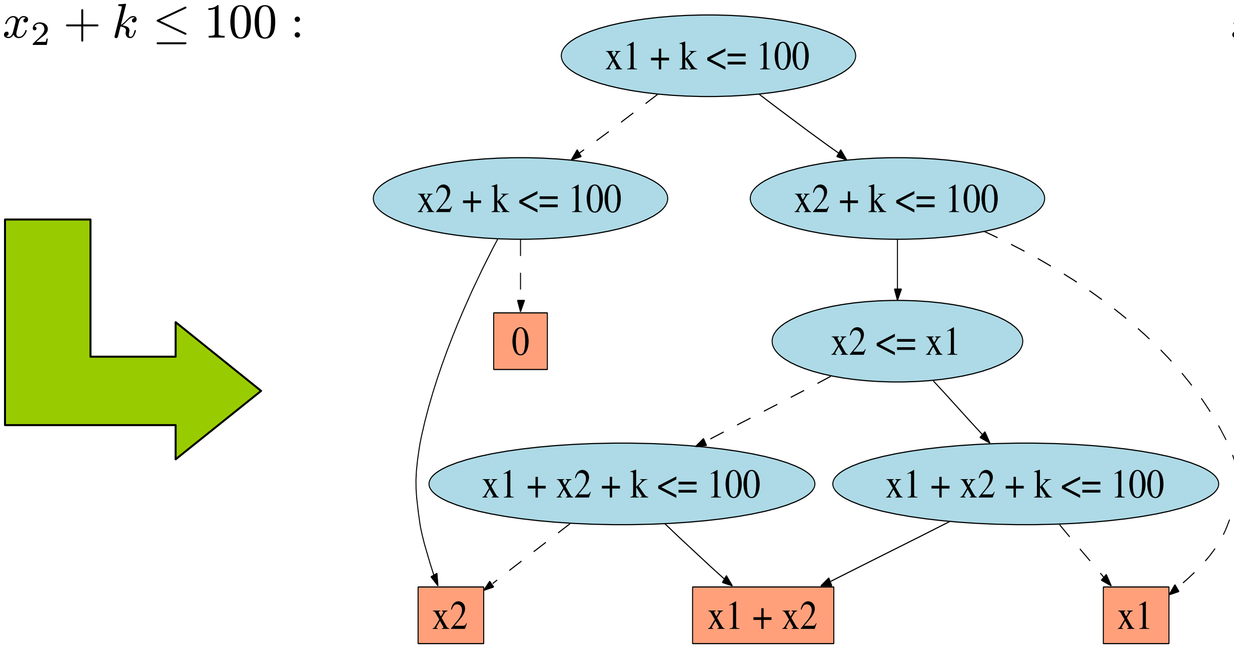
SDP needs an efficient data structure for

- compact, minimal case representation
- efficient case operations

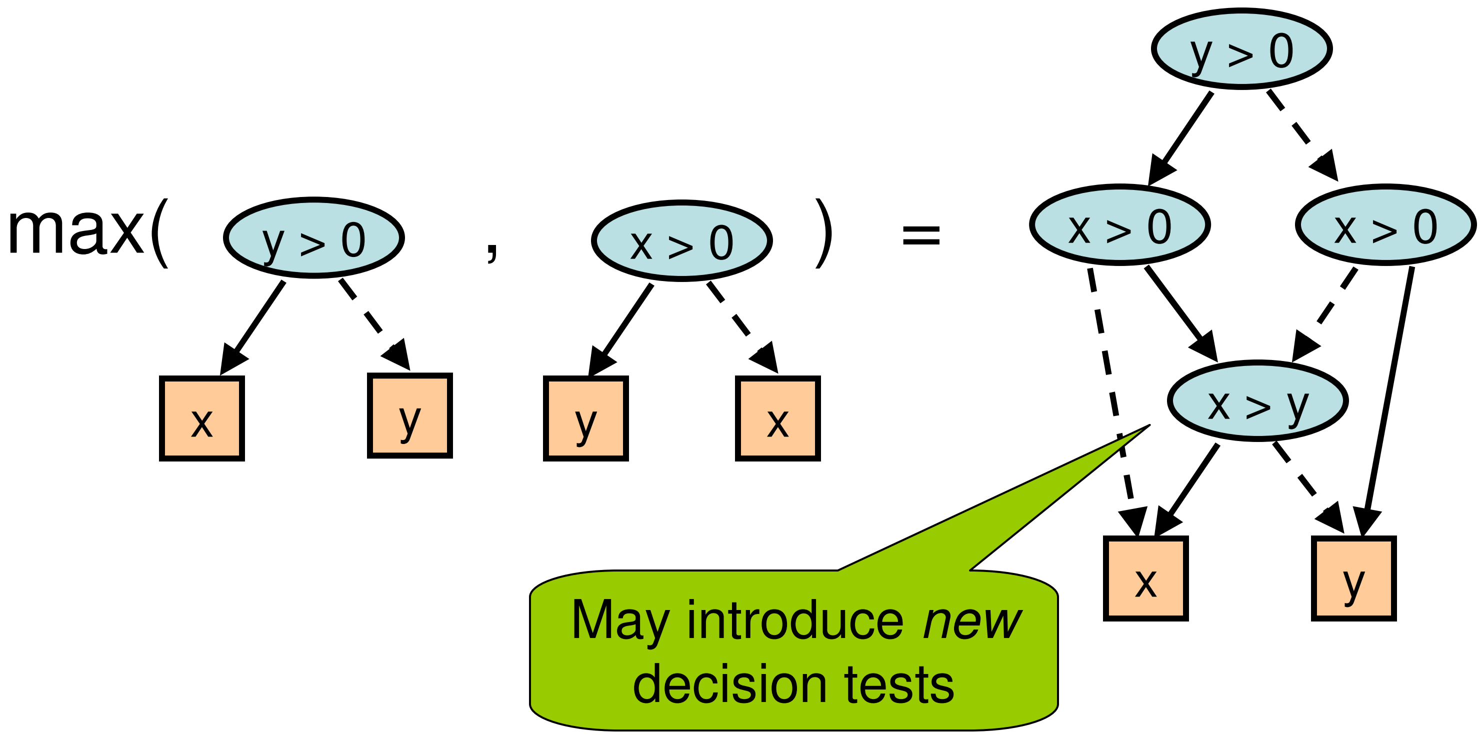
XADDs

- Extended ADD representation of case statements

$$V = \begin{cases} x_1 + k > 100 \wedge x_2 + k > 100 : & 0 \\ x_1 + k > 100 \wedge x_2 + k \leq 100 : & x_2 \\ x_1 + k \leq 100 \wedge x_2 + k > 100 : & x_1 \\ x_1 + x_2 + k > 100 \wedge x_1 + k \leq 100 \wedge x_2 + k \leq 100 \wedge x_2 > x_1 : & x_2 \\ x_1 + x_2 + k > 100 \wedge x_1 + k \leq 100 \wedge x_2 + k \leq 100 \wedge x_2 \leq x_1 : & x_1 \\ x_1 + x_2 + k \leq 100 : & x_1 + x_2 \end{cases}$$



XADD Maximization

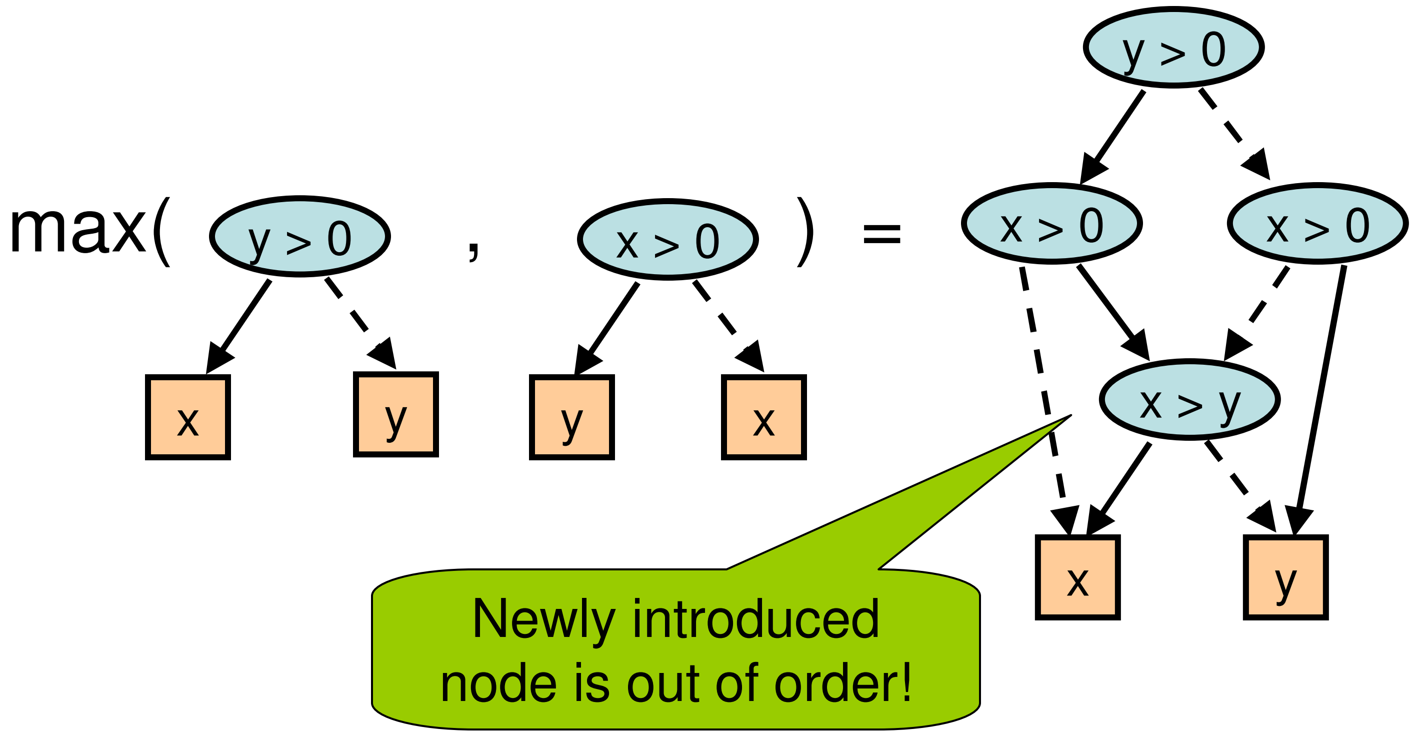


Maintaining XADD Orderings I

- Max may get variables out of order

Decision ordering (root→leaf)

- x > y
- y > 0
- x > 0

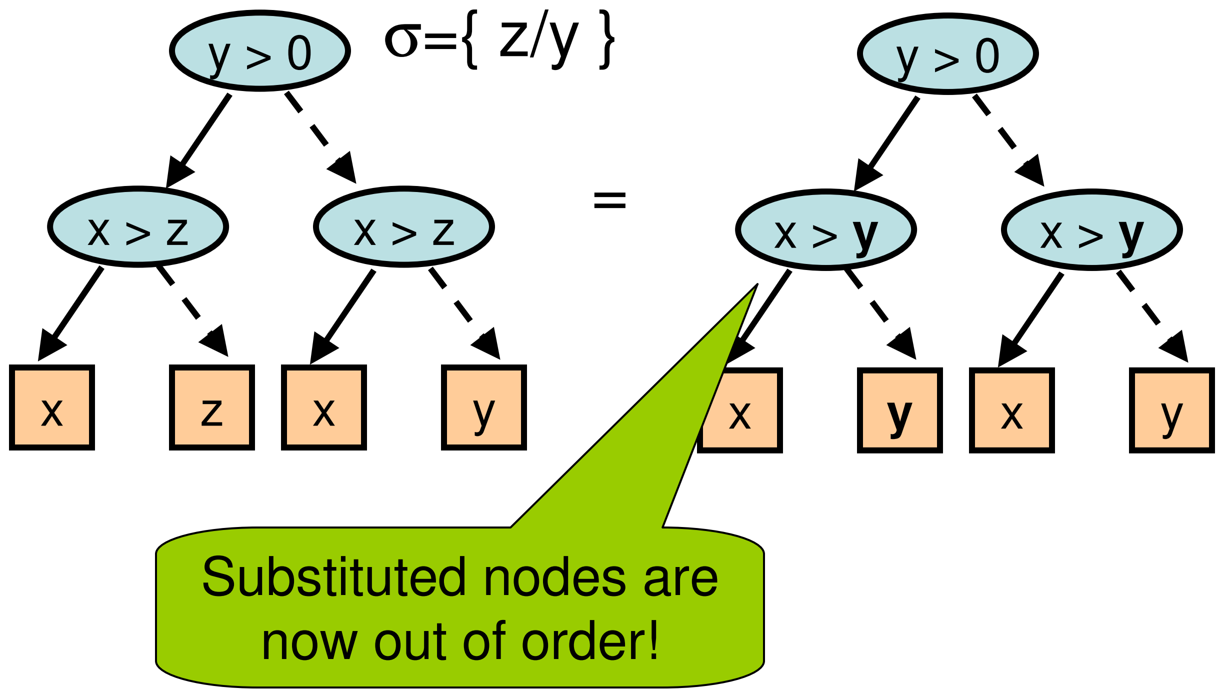


Maintaining XADD Orderings II

- Substitution may get vars out of order

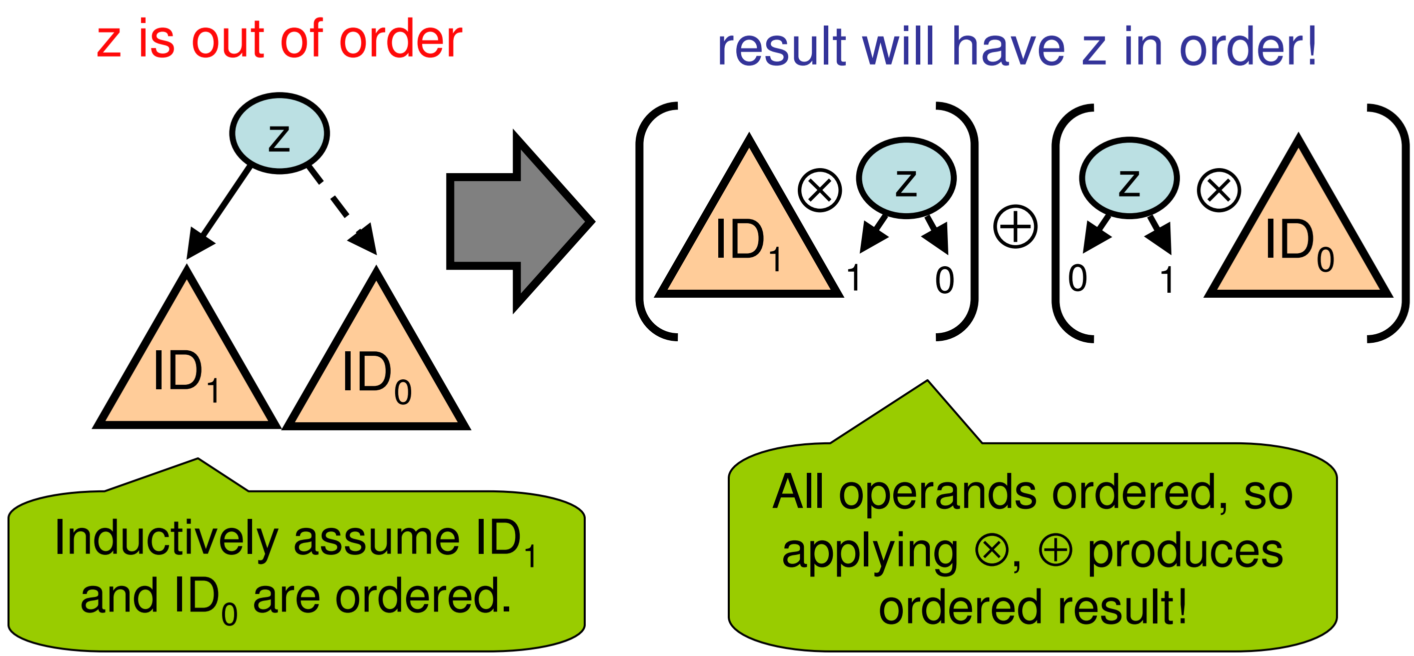
Decision ordering ($root \rightarrow leaf$):

- $x > y$
- $y > 0$
- $x > z$



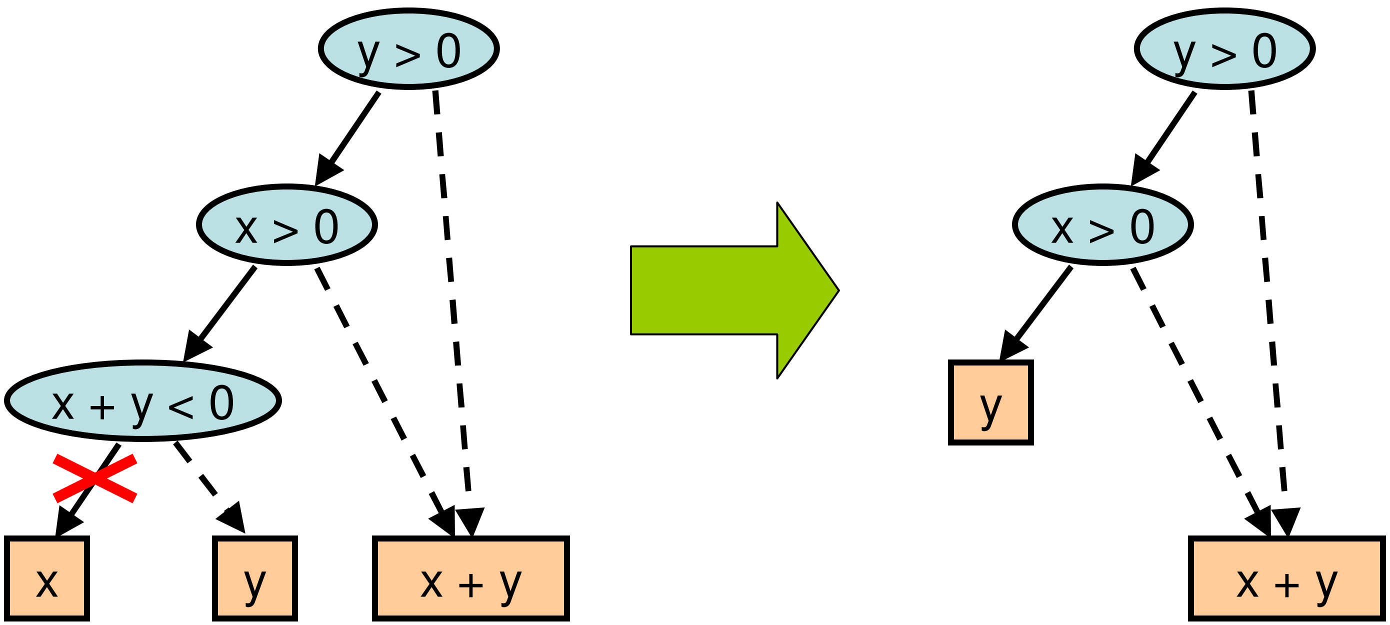
Correcting XADD Ordering

- Build *ordered* XADD from *unordered* XADD



Similar to Penberthy & Weld, AAAI-94

XADD Pruning



Node unreachable – $x + y < 0$ always false if $x > 0$ & $y > 0$

If **linear**, can detect with feasibility checker of LP solver & prune

Empirical Results

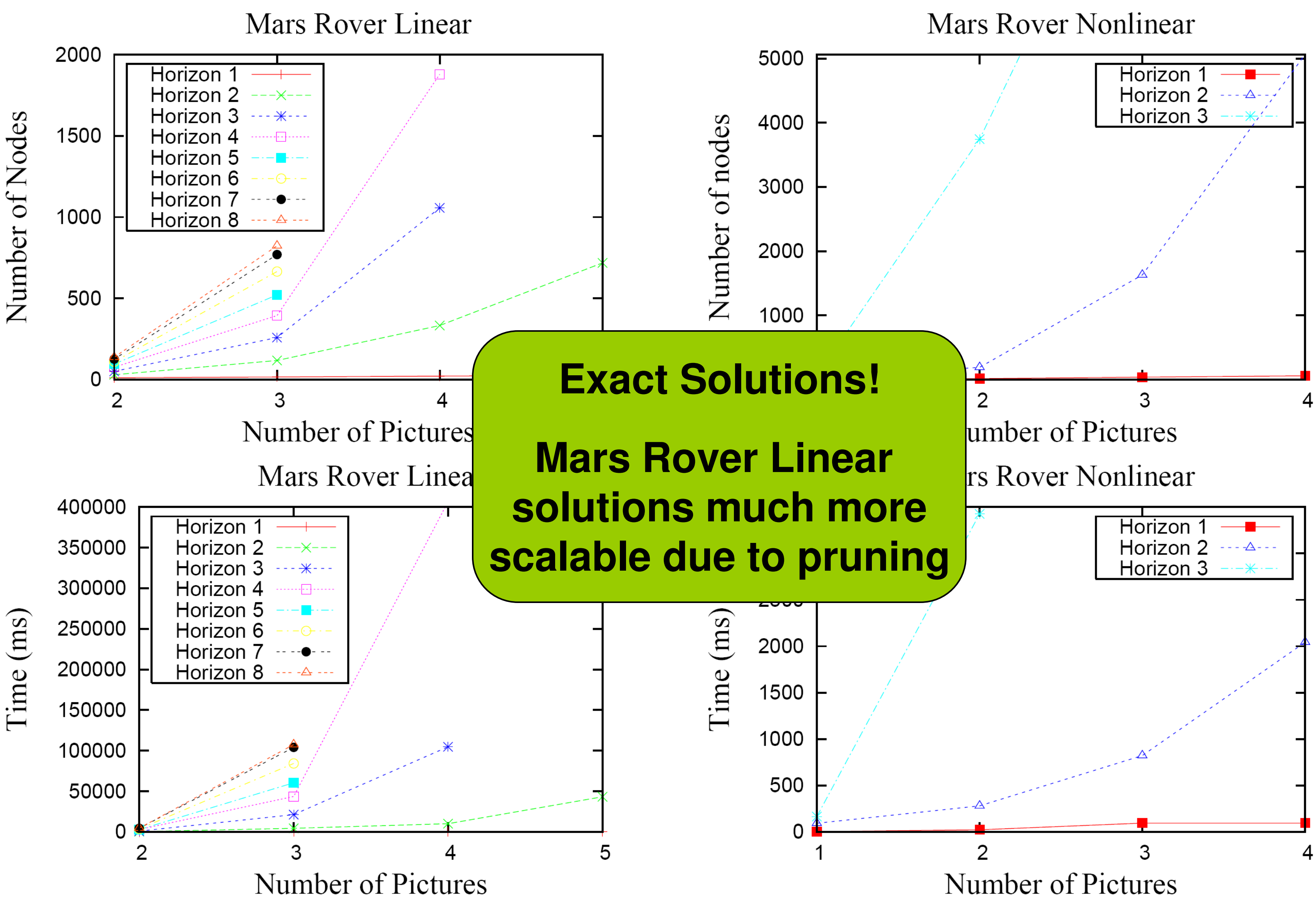
Problem Domains

- Knapsack problem (high-dimensional toy problem)
  - Transfer continuous resources to knapsack
    - Subject to capacity constraints
    - Reward for amount transferred
  - Can solve for optimal ∞-horizon solution

- Mars Rover (variants of Bresina *et al*, UAI-02)
  - Linear
    - take pictures with linear time/energy constraints
  - Nonlinear
    - move to target (x,y) position, taking pictures along way
    - reward is truncated quadratic

- All problem domains / code online:
  - http://code.google.com/p/xadd-inference/

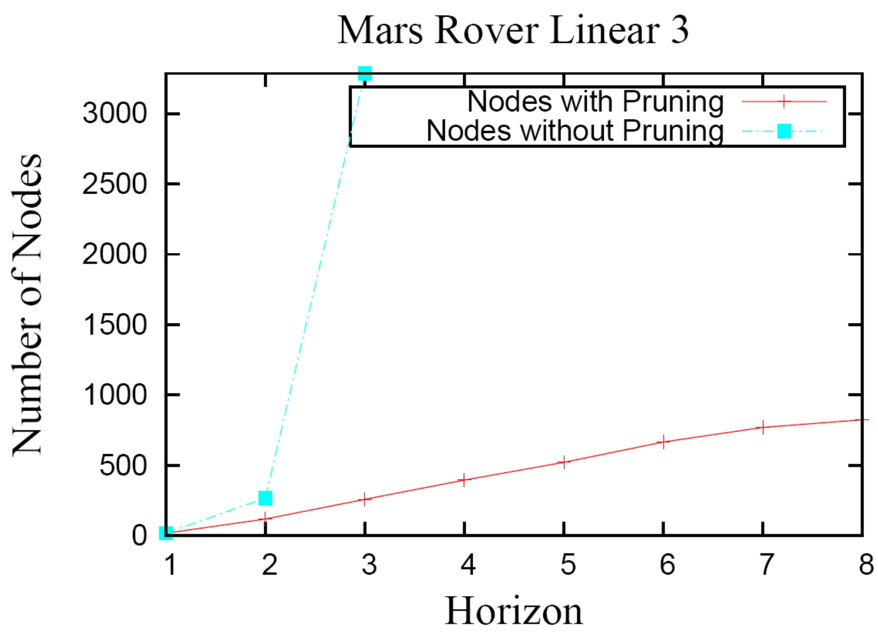
Results: Time and Space for Mars Rover

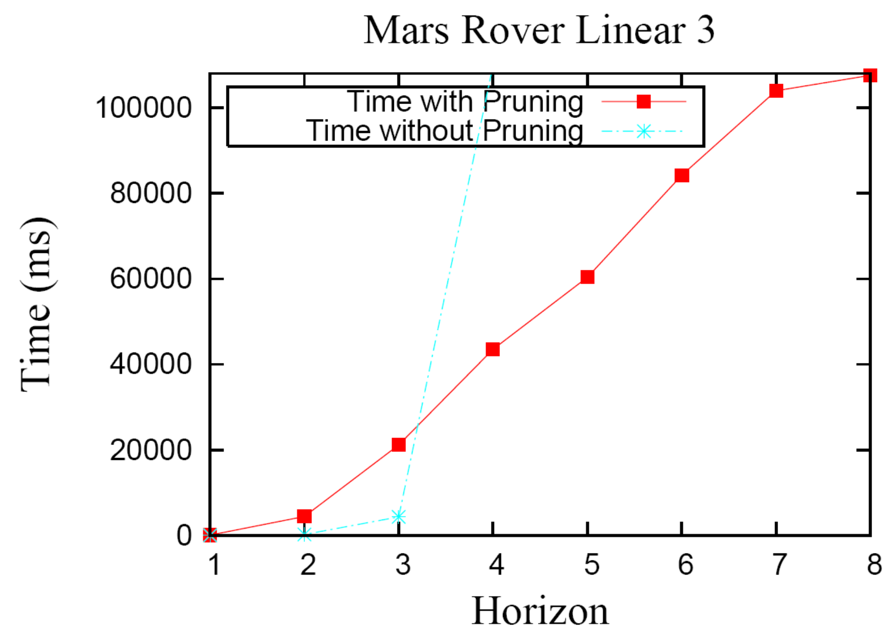


Exact Solutions!

Mars Rover Linear solutions much more scalable due to pruning

Results: XADD Pruning vs. No Pruning





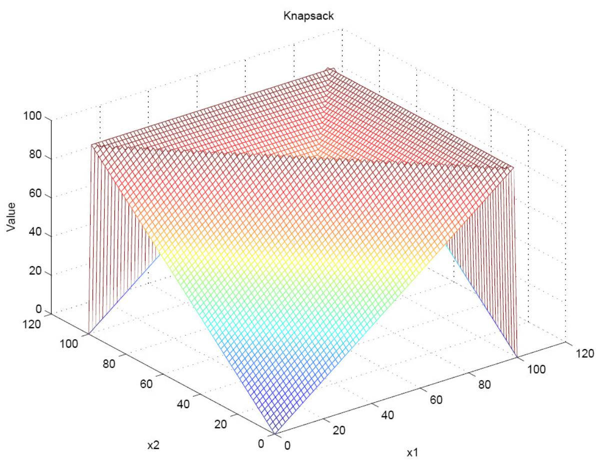
Summary:

- **without pruning: superlinear vs. horizon**
- **with pruning: linear vs. horizon**

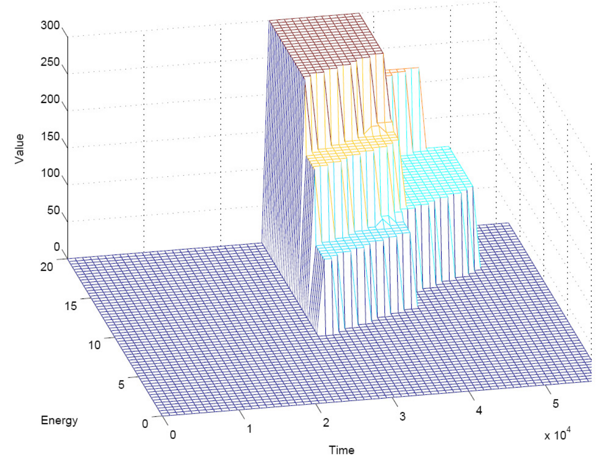
Worth the effort to prune!

Obligatory 3D Value Function Gallery

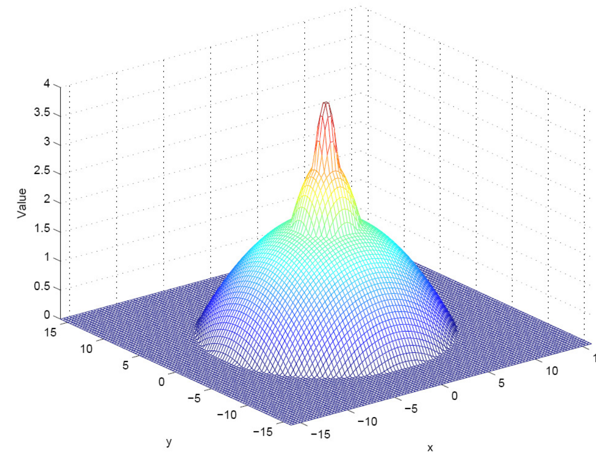
Knapsack



Mars Rover Linear



Mars Rover Nonlinear



Exact value functions in case form:

- **linear & nonlinear piecewise boundaries!**
- **nonlinear function surfaces!**

Conclusions

- First exact, closed-form solutions to subset of **multidimensional, nonlinear DC-MDPs**
- Key insights
  - Symbolic *case* representation
  - DP in terms of case $\otimes$, $\oplus$, max
  - $\int \delta$ triggers (conditional) substitution
- Need compact case, efficient operations
  - Case $\rightarrow$ Extended ADD (XADD)
  - $\otimes$, $\oplus$ technique for efficient decision reordering
  - Advantages of pruning

Future Work

- Efficiency
  - XADD pruning in nonlinear case
  - XADD Approximation?
    - Extend APRICODD (St-Aubin *et al,* NIPS-00)
- Expressivity
  - Full continuous stochastic extension
    - Currently, continuous transitions are mixture of δ's
    - Ideally want Gaussian noise, etc.
  - Continuous actions?
  - Partial observability?

Thank you!

Questions?

Extra Slides

XADD: Details

- The XADD is an ADD allowing
  - Arbitrary expressions at leaves
  - Arbitrary expression inequalities at decision nodes
    - If expressions polynomial, decisions & leaves have canonical form
    - Enforce ordering on all decision tests: $(x < y)$ before $(x < 3)$
- Operations same as XADD
  - But leaf operations may produce XADDs themselves!
    - May also require introduction of new decisions
    - E.g., maximization
- Can introduce support for substitution
  - Needed for SDP regression

1D: Boyan and Littman (1999)

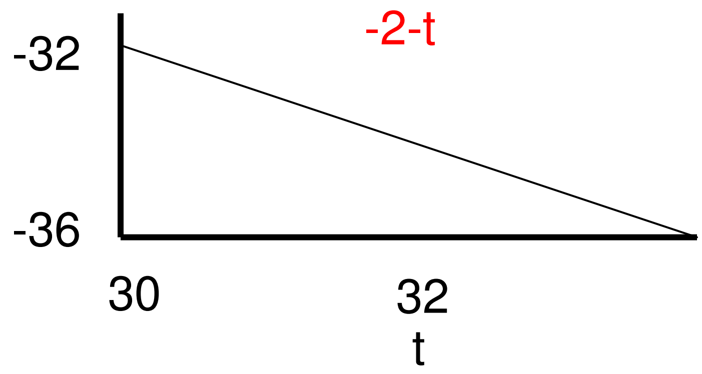
- Exact Solutions to Time-dependent MDPs
  - Assume actions / transitions as follows

Action = bus:

t' = t + 30

s' = office

R(s,t) = -2 – t + 20*I[s'=office]

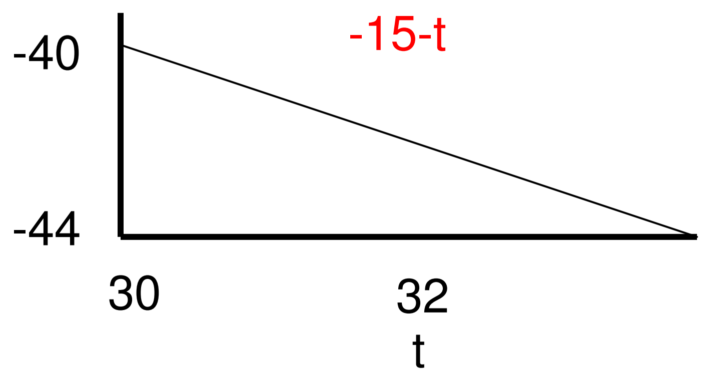


Action = taxi:

t' = t + 10

s' = office

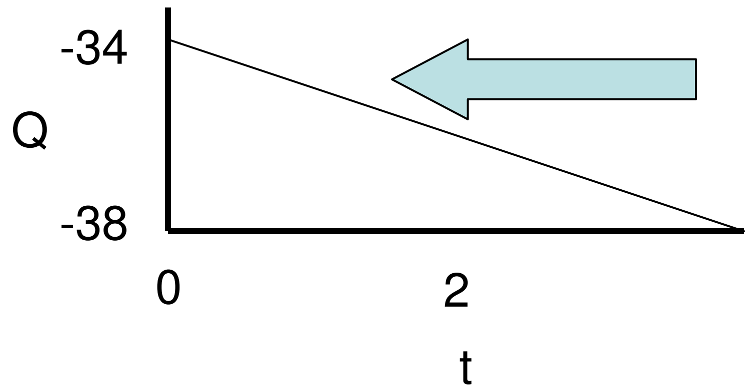
R(s,t) = -15 - t + 20*I[s'=office]



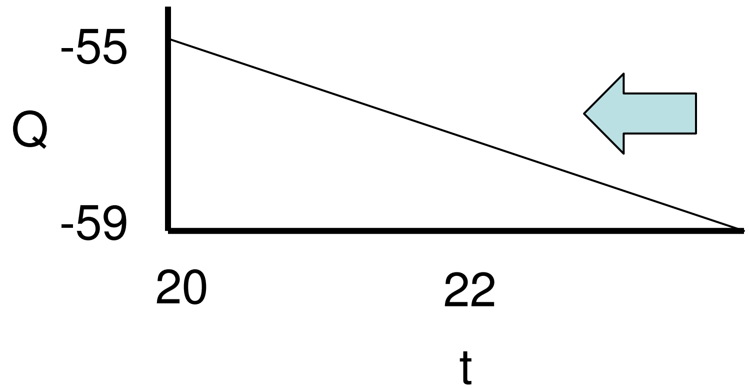
1D: Boyan and Littman (1999)

- Continuous transitions are δ-functions
  - Regressions just sums & translations of value

Q(a=bus,s=office,t)



Q(a=taxi,s=office,t)



1D: Boyan and Littman (1999)

- Value max is just piecewise partitioning

max(Q(a=bus,s=office,t), Q(a=taxi,s=office,t))