INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Decision Making in Multiagent Settings: Team Decision Making

Matthijs Spaan[§] Christopher Amato[*] Shlomo Zilberstein[*]

[§]Institute for Systems and Robotics, IST, Lisbon, Portugal

[*]University of Massachusetts Amherst, MA, USA

AAMAS10 Tutorial, May 10, 2010

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Tutorial outline of Part 2

Overlapping material with Part 1 is indicated like *this*.

1. Introduction
2. Models
   - *Single-agent MDPs and POMDPs*
   - Decentralized POMDPs
   - Subclasses and complexity
3. Algorithms
   - Exact algorithms
   - Approximation methods
   - Specialized algorithms for subclasses
4. Problem domains and software tools
5. Wrapup

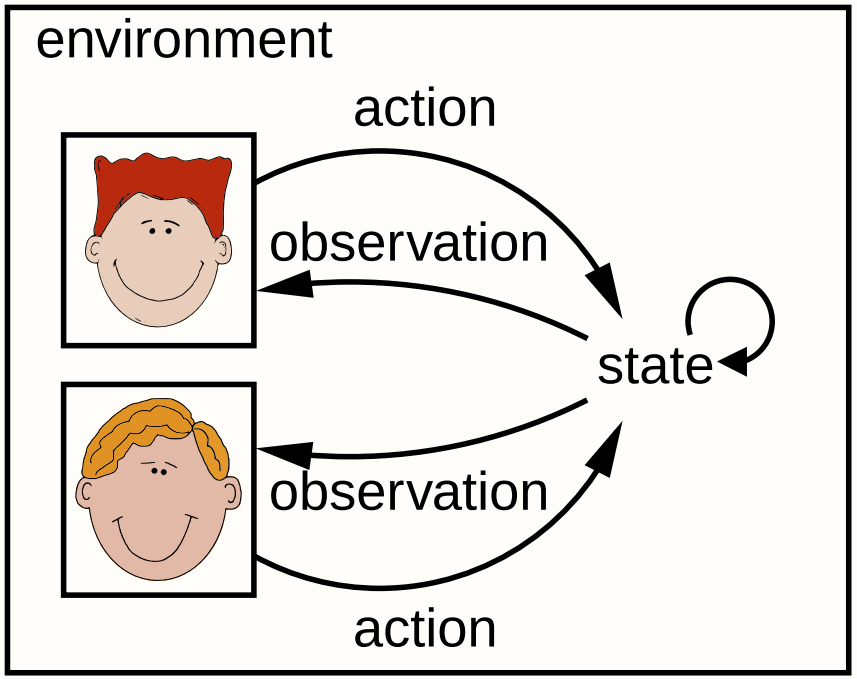
INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Introduction

Introduction

- AI: develop intelligent agents.
- Cooperating multiagent systems.
- Problem: planning how to act.
- Joint payoff but decentralized actions and observations.



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

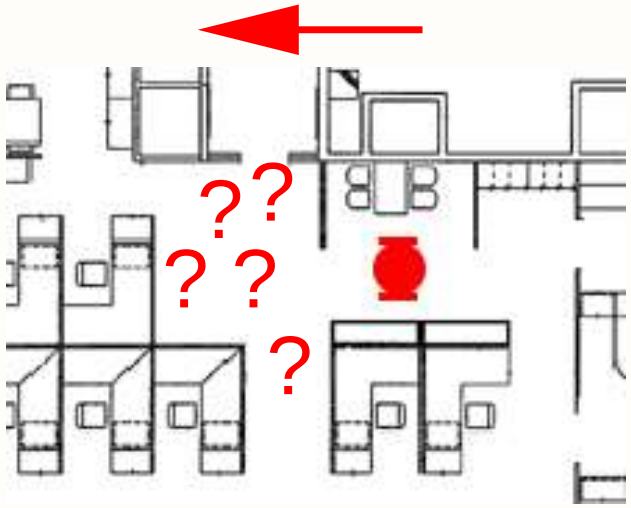
Related previous work

- Group decision theory in economics, team theory (Marschak, 1955; Papadimitriou and Tsitsiklis, 1982)
- Decentralized detection (Tsitsiklis and Athans, 1985; Tsitsiklis, 1988)
- Optimization of decentralized systems in operations research (Witsenhausen, 1971; Sandell et al., 1978)
- Communication strategies (Varaiya and Walrand, 1978; Xuan et al., 2001; Pynadath and Tambe, 2002)
- Approximation algorithms (Peshkin et al., 2000; Guestrin et al., 2002; Nair et al., 2003; Emery-Montemerlo et al., 2004)

Decision-theoretic planning

- Decision-theoretic planning tackles uncertainty in sensing and acting in a principled way.

- We need to model:
  - each agent's actions
  - their sensors
  - their environment
  - their task



- Popular for single-agent planning under uncertainty (MDPs, POMDPs).

Decision-theoretic planning

Assumptions:

- Sequential decisions: problems are formulated as a sequence of discrete "independent" decisions.
- Markovian environment: the state at time t depends only on the events at time $t-1$.
- Stochastic models: the uncertainty about the outcome of actions and sensing can be accurately captured.
- Objective encoding: the overall objective can be encoded using cumulative (discounted) rewards over time steps.

Multiagent planning problems

Aspects:

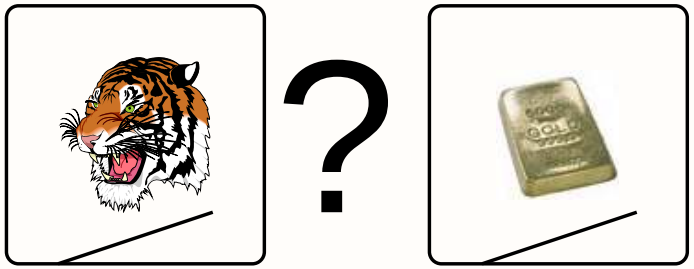
- on-line vs. off-line
- centralized vs. distributed
  - planning
  - execution
- cooperative vs. self-interested
- observability
- communication

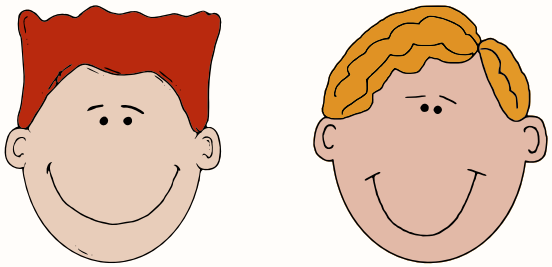
INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Example: The DEC-Tiger problem

- A toy problem: decentralized tiger (Nair et al., 2003).
- Opening correct door: both receive treasure.
- Opening wrong door: both get attacked by a tiger.
- Agents can open a door, or listen.
- Two noisy observations: hear tiger left or right.
- Don’t know the other’s actions or observations.



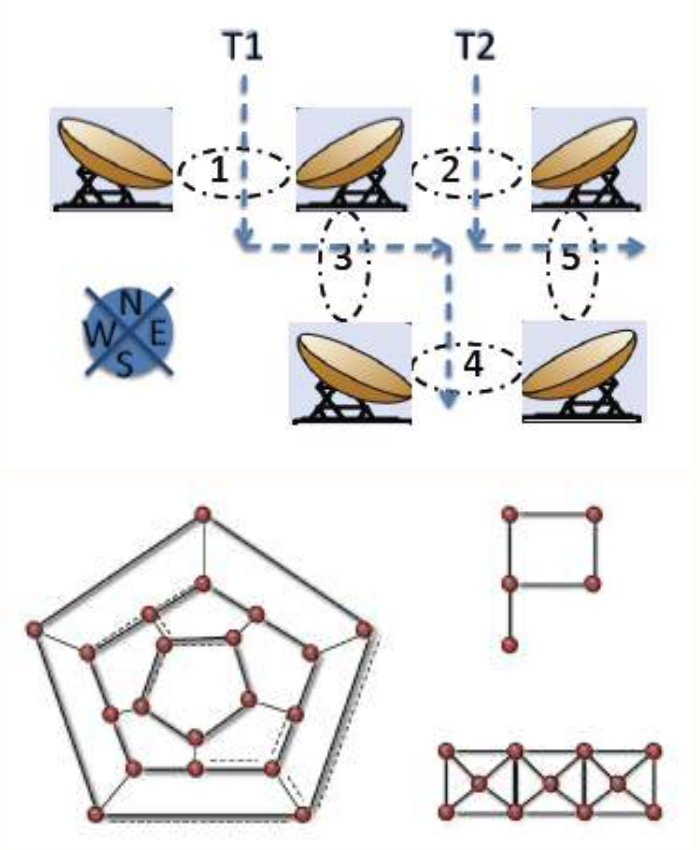


INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Example: Sensor network problems

- Sensor networks for
  - Target tracking (Nair et al., 2005; Kumar and Zilberstein, 2009a)
  - Weather phenomena (Kumar and Zilberstein, 2009b)
- Two or more cooperating sensors.



Application domains

Possible application domains:

- Multi-robot coordination
  - Space exploration rovers (Zilberstein et al., 2002)
  - Helicopter flights (Pynadath and Tambe, 2002)
  - Navigation (Emery-Montemerlo et al., 2005; Spaan and Melo, 2008)
- Load balancing for decentralized queues (Cogill et al., 2004)
- Multi-access broadcast channels (Ooi and Wornell, 1996)
- Network routing (Peshkin and Savova, 2002)
- Sensor network management (Nair et al., 2005)

Models

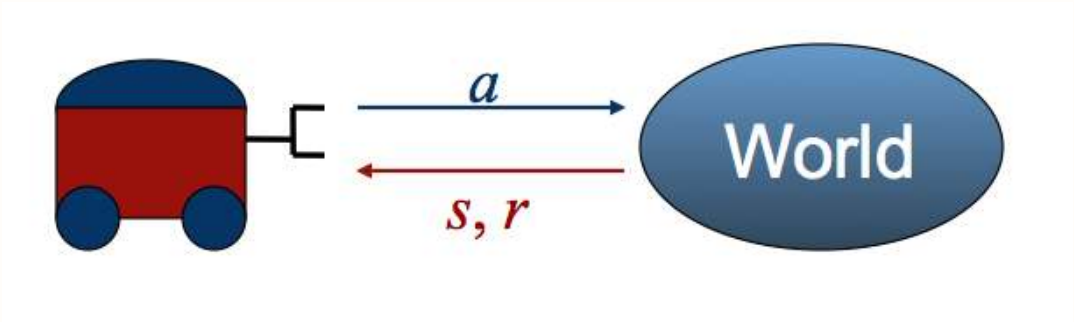
INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Models outline

1. *MDPs: single agent, fully observable*
2. *POMDPs: single agent, partially observable*
3. Decentralized POMDPs
4. Complexity results
5. DEC-POMDP subclasses

Markov decision processes



- A model of sequential decision-making developed in operations research in the 1950's.
- Allows reasoning about actions with uncertain outcomes.
- MDPs have been adopted by the AI community as a framework for:
  - ► Decision-theoretic planning (e.g., Dean et al., 1995)
  - ► Reinforcement learning (e.g., Barto et al., 1995)

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Definition of Markov decision processes

Definition 1. *A Markov decision process (MDP) is a tuple* $\langle S, A, P, R \rangle$ *where*

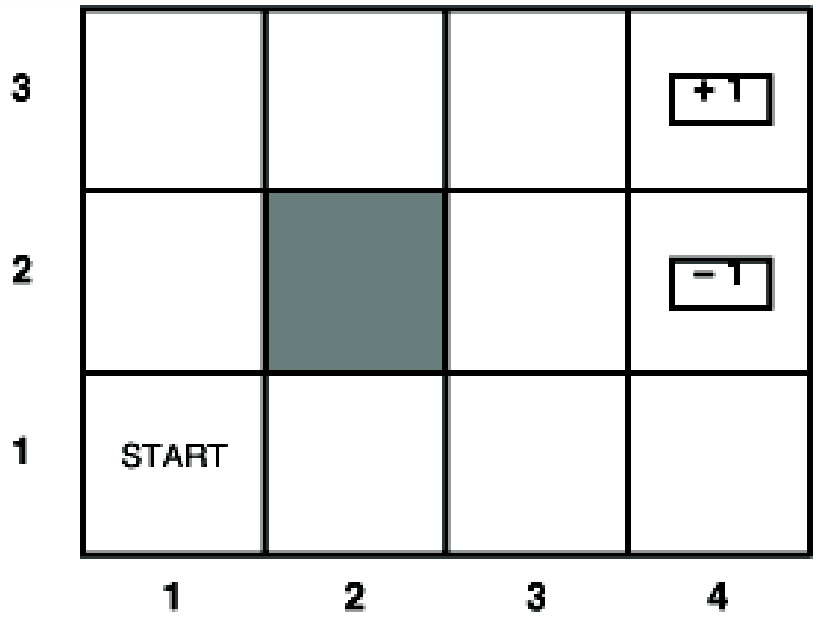
- S *is a finite set of states, with distinguished initial state* s_0*.*
- A *is a finite set of actions*
- $P : S \times A \rightarrow \Delta S$ *is a Markovian transition function.* $P(s'|s, a)$ *denotes the probability that taking action* a *in state* s *will result in a transition to state* s'*.*
- $R : A \times S \rightarrow \mathbb{R}$ *is a reward function.* $R(a, s')$ *denotes the reward obtained when action* a *is taken and a state transition to* s' *occurs.*

The Markov assumption:

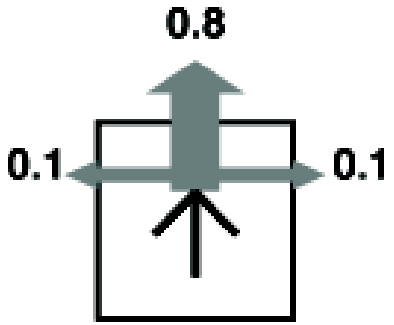
$$P(s_t|s_{t-1}, s_{t-2}, \ldots, s_0, a) = P(s_t|s_{t-1}, a)$$

Example of a Markov decision process

A simple 4×3 grid environment (Russell and Norvig, 2003)



(a)



(b)

Partially observable MDPs

After each action, the agent receives an observation o that provides partial information about the underlying state s.

Definition 2. *A partially observable MDP (POMDP) is a tuple* $\langle S, A, P, \Omega, O, R \rangle$ *where*

- S, A, P, *and* R *are the same as for MDP*
- Ω *is a finite set of observations*
- $O : A \times S \rightarrow \Delta\Omega$ *is an observation function.* $O(o|a, s')$ *denotes the probability of observing* o *when action* a *is taken and and a state transition to* s' *occurs.*

Performance criteria

- How to combine rewards over multiple time steps or histories?
- The assumption that the agent's preference over histories depends only on the current state allows only two possible ways to define utilities of histories:
  1. $V([s_0, a_1, s_1, a_2, s_2, \ldots]) = R(a_1, s_1) + R(a_2, s_2) + \ldots$
  2. $V([s_0, a_1, s_1, a_2, s_2, \ldots]) = R(a_1, s_1) + \gamma R(a_2, s_2) + \ldots$
- Finite-horizon problems involve a fixed number of steps, h.
- Best action in each state may depend on the number of steps left (nonstationary).
- Infinite-horizon policies depend only on the current state (stationary).
- Finite-horizon problems can be solved by adding the number of steps left to the state.

Policies and value functions

- A policy π is a mapping from states to actions.
- The value function for a finite-horizon MDP:

$$V^{\pi}(s_0) = E\Big[\sum_{t=0}^{h-1} R(\pi(s_t), s_{t+1})\Big].$$

- The value function for an infinite-horizon MDP:

$$V^{\pi}(s_0) = E\Big[\sum_{t=0}^{\infty} \gamma^t R(\pi(s_t), s_{t+1})\Big].$$

The Bellman equation

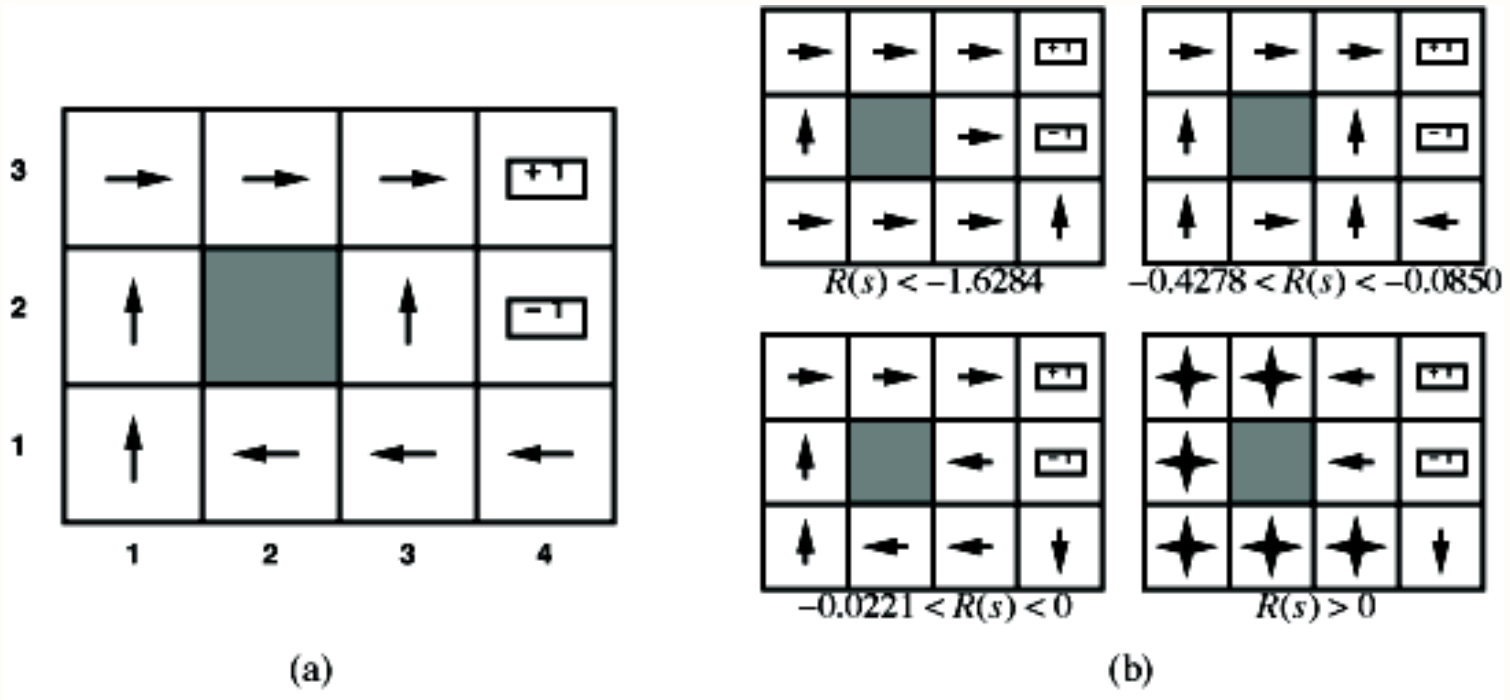
- Optimal policy defined by:

$$\pi^*(s) = \arg\max_a \sum_{s'} p(s'|s,a)V(s')$$

$$V(s) = R(s) + \gamma \max_a \sum_{s'} p(s'|s,a)V(s')$$

- In these equations, the reward function depends only on state, but this can be easily generalized.
- Can be solved using dynamic programming (Bellman, 1957)

Examples of optimal policies



(a) An optimal policy for the stochastic environment with $R(s) = -0.04$ for all nonterminal states. (b) Optimal policies for four different ranges of $R(s)$. (Russell and Norvig, 2003)

Value iteration for MDPs

Algorithm 1: Value iteration (Bellman, 1957)

input : MDP problem, convergence parameter ε

output: A policy that is ε-optimal for all states

begin

  Initialize V'

  repeat

    $V \leftarrow V'$

    for *each state* s **do**

      $V'(s) \leftarrow R(s) + \gamma \max_a \sum_{s'} p(s'|s,a) V(s')$

  until *CloseEnough*(V, V')

  return Greedy policy with respect to V'

end

Algorithm 2: Policy iteration (Howard, 1960)

input : MDP problem
output: An optimal policy
begin
  Initialize π'
  repeat
    $\pi \leftarrow \pi'$
    $V \leftarrow \text{ValueDetermination}(\pi)$
    for *each state* s **do**
      $\pi'(s) \leftarrow \arg\max_a \sum_{s'} p(s'|s,a)V(s')$
  until $\pi = \pi'$
  return π'
end

Value determination

Value determination of a policy π can be implemented using:
value iteration:

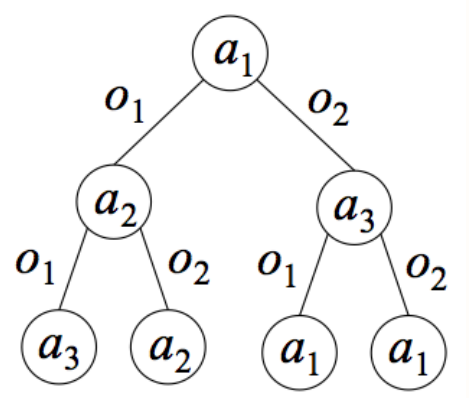
$$V'(s) \leftarrow R(s) + \gamma \sum_{s'} p(s'|s, \pi(s)) V(s')$$

or by solving a set of n linear equations:

$$V(s) = R(s) + \gamma \sum_{s'} p(s'|s, \pi(s)) V(s')$$

Solving POMDPs

- Need to act based on partial observations
- Finite-horizon problems: A policy can be represented as a mapping from observation sequences to actions, $\pi : \Omega^* \rightarrow A$
- This can be summarized using a policy tree:



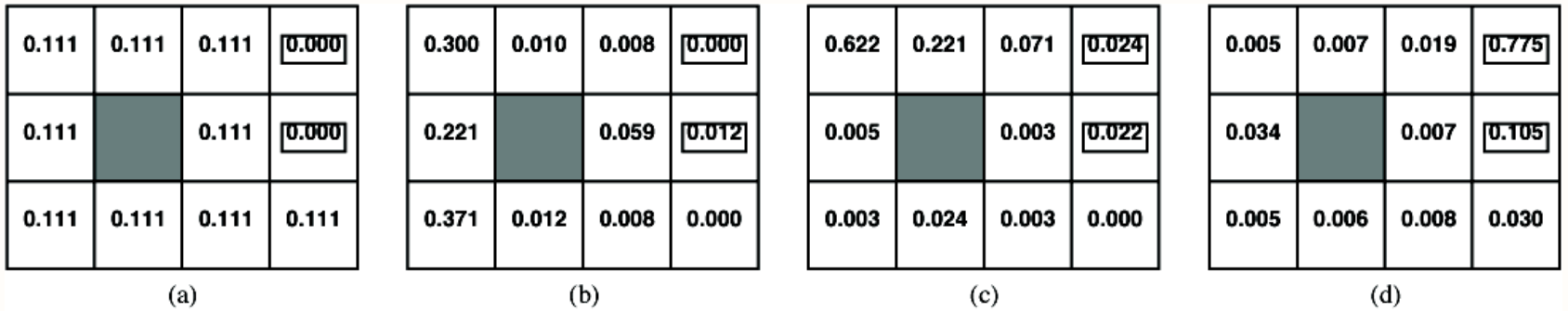
- Infinite-horizon problems: Solution can be represented as a finite-state controller with nodes labelled with actions and transitions labelled with observations
- Use of stochastic controllers versus deterministic

Belief states

A belief state is a probability distribution over states that can summarize the knowledge of the agent at a given point.

$$b(s_t) = Pr(s_t = s | s_0, a_1, o_1, a_2, o_2, \ldots, a_{t-1}, o_{t-1})$$

Example: Consider the 4×3 domain with no observation.



(a) The initial probability distribution for the agent's location

(b) after moving *Left* five times

(c) after moving *Up* five times

(d) after moving *Right* five times

Bayesian updating of beliefs

- When action a is taken in belief state $b(s)$ and o is observed, the new belief $b'(s')$ can be calculated using Bayes' rule:

$$b'(s') = Pr(s'|b,a,o) = \frac{O(o|a,s')\sum_s b(s)P(s'|s,a)}{Pr(o|a,b)}$$

- The probability of the observation can be computed by summing over all possible s'

$$\begin{aligned} Pr(o|a,b) &= \sum_{s'} Pr(o|a,s',b)Pr(s'|a,b) \\ &= \sum_{s'} O(o|a,s')Pr(s'|a,b) \\ &= \sum_{s'} O(o|a,s') \sum_s b(s)P(s'|s,a) \end{aligned}$$

Belief state transition model

We can now define a new "belief-state MDP" with the following transition model:

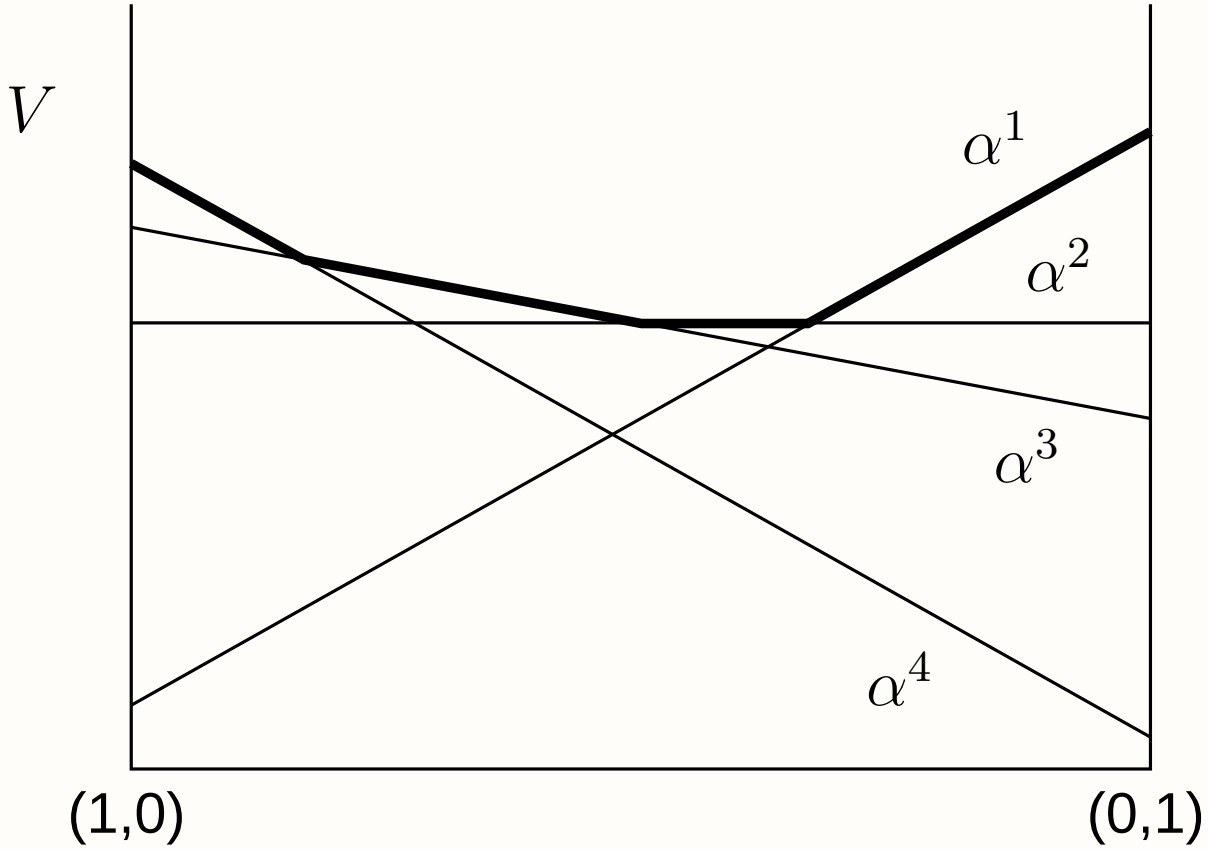
$$\begin{aligned} Pr(b'|b,a) &= \sum_o Pr(b'|o,a,b)Pr(o|a,b) \\ &= \sum_o Pr(b'|o,a,b) \sum_{s'} O(o|a,s') \sum_s b(s)P(s'|s,a) \end{aligned}$$

And the following reward function:

$$\rho(b) = \sum_s b(s)R(s)$$

Solving POMDPs

The optimal value function of a (finite-horizon) POMDP is piecewise linear and convex: $V(b) = \max_\alpha b \cdot \alpha$.



INSTITUTO SUPERIOR TÉCNICO

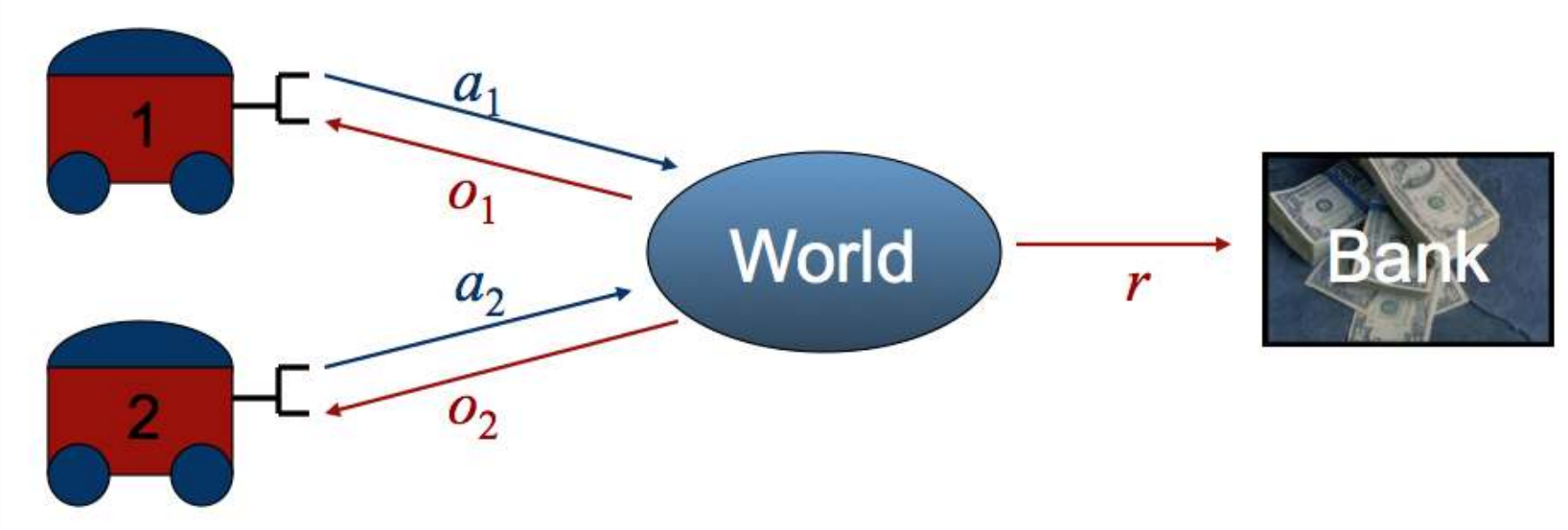
University *of* Massachusetts Amherst

POMDP methods

- Optimal
  - Enumerate and prune (Monahan, 1982; Zhang and Liu, 1996; Hansen, 1998b)
  - Search for witness points (Sondik, 1971; Cassandra et al., 1994)
- Heuristic
  - Most likely state (Cassandra et al., 1996)
  - Q_{MDP} (Littman et al., 1995)
- Approximate
  - Grid-based approximations (Lovejoy, 1991; Brafman, 1997; Zhou and Hansen, 2001; Bonet, 2002)
  - Optimizing finite-state controllers (Poupart and Boutilier, 2004; Amato et al., 2007)
  - Branch-and-bound search (Satia and Lave, 1973; Hansen, 1998a)
  - Point-based techniques (Pineau et al., 2003; Spaan and Vlassis, 2005)

Decentralized POMDPs

Now we consider a group of agents that control the environment jointly.



Each agent receives a separate partial observation.
The agents try to optimize a single reward function.

DEC-POMDP

Definition 3. *A decentralized partially observable MDP (DEC-POMDP) is a tuple* $\langle I, S, \{A_i\}, P, \{\Omega_i\}, O, R, h\rangle$ *where*

- I *is a finite set of agents indexed* $1, \ldots, n$.
- S *is a finite set of states, with distinguished initial state* s_0.
- A_i *is a finite set of actions available to agent* i, *and* $\vec{A} = \otimes_{i \in I} A_i$ *is the set of joint actions.*
- $P : S \times \vec{A} \to \Delta S$ *is a Markovian transition function.* $P(s'|s, \vec{a})$ *denotes the probability that after taking joint action* $\vec{a}$ *in state* s *a transition to state* s' *occurs.*
- Ω_i *is a finite set of observations available to agent* i, *and* $\vec{\Omega} = \otimes_{i \in I} \Omega_i$ *is the set of joint observations.*
- $O : \vec{A} \times S \to \Delta\vec{\Omega}$ *is an observation function.* $O(\vec{o}|\vec{a}, s')$ *denotes the probability of observing joint observation* $\vec{o}$ *given that joint action* $\vec{a}$ *was taken and led to state* s'.
- $R : \vec{A} \times S \to \mathbb{R}$ *is a reward function.* $R(\vec{a}, s')$ *denotes the reward obtained after joint action* $\vec{a}$ *was taken and a state transition to* s' *occurred.*
- *If the DEC-POMDP has a finite horizon, that horizon is represented by a positive integer* h.

Partially observable stochastic games

Definition 4. *A partially observable stochastic game (POSG) is a tuple* $\langle I, S, \{A_i\}, P, \{\Omega_i\}, O, \{R_i\}, h\rangle$ *where*

- *All the components except the reward function are the same as in a DEC-POMDP*
- *Each agent has an individual reward function:* $R_i : A_i \times S \to \mathbb{R}$. $R_i(a_i, s')$ *denotes the reward obtained after action* a_i *was taken by agent* i *and a state transition to* s' *occurred.*

This is the self-interested version of the DEC-POMDP model

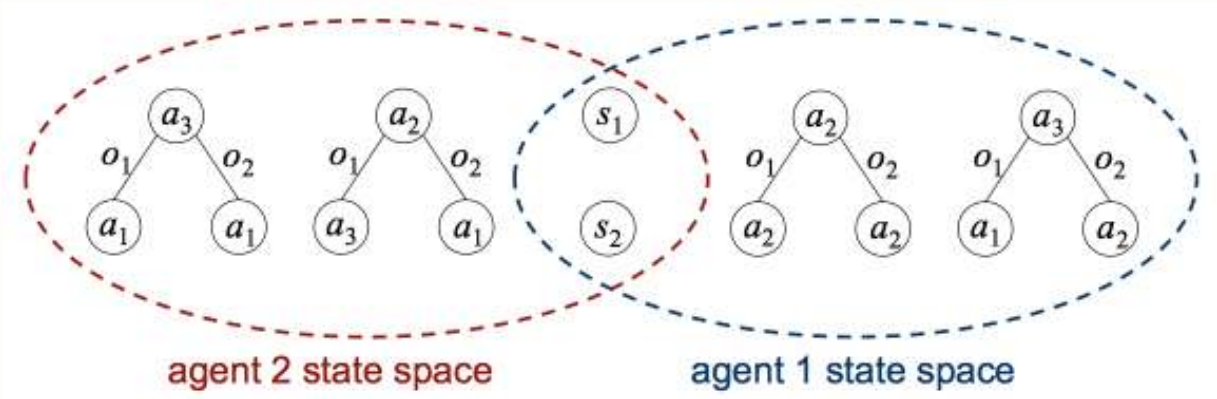
Policies for DEC-POMDPs

Definition 5. *A **local policy** for agent i, π_i, is a mapping from local histories of observations $\vec{\mathbf{o}}^i = (o_{i_1} \cdots o_{i_t})$ over Ω_i to actions in A_i, $\pi_i : \Omega_i^* \rightarrow A_i$.*

Definition 6. *A **joint policy**, $\pi = \langle \pi_1, \ldots, \pi_n \rangle$, is a tuple of local policies, one for each agent.*

Other forms of policy representations:

- Mapping from (generalized) belief states to actions:



- Mapping from internal memory states to actions (FSCs)

Value functions for DEC-POMDPs

Definition 7. *The value of a joint policy π for a finite-horizon DEC-POMDP with initial state s_0 is:*

$$V^{\pi}(s_0) = E\Big[\sum_{t=0}^{h-1} R(\vec{a}_t, s_t) | s_0, \pi\Big].$$

Definition 8. *The value of a joint policy π for an infinite-horizon DEC-POMDP with initial state s_0 and discount factor $\gamma \in [0, 1)$ is:*

$$V^{\pi}(s_0) = E\Big[\sum_{t=0}^{\infty} \gamma^t R(\vec{a}_t, s_t) | s_0, \pi\Big].$$

Modeling communication

Definition 9. *A decentralized partially observable Markov decision process with communication (DEC-POMDP-COM) is a tuple* $\langle I, S, \{A_i\}, P, \{\Omega_i\}, O, \Sigma, C_\Sigma, R, h\rangle$ *where:*

- I, S, $\{A_i\}$, P, $\{\Omega_i\}$, O, *and* h *are defined as in the DEC-POMDP.*
- Σ *is the alphabet of communication messages.* $\sigma_i \in \Sigma$ *is an atomic message sent by agent* i*, and* $\vec{\sigma} = \langle \sigma_1, \ldots, \sigma_n \rangle$ *is a joint message, i.e. a tuple of all messages sent by the agents in one time step. A special message belonging to* Σ *is the null message,* ε_σ*, which is sent by an agent that does not want to transmit anything to the others. Agents incur no cost for sending a null message.*
- C_Σ *is the cost of transmitting an atomic message.* $C_\Sigma : \Sigma \to \mathbb{R}$, $C_\Sigma(\varepsilon_\sigma) = 0$.
- *R is the reward function.* $R(\vec{a}, s', \vec{\sigma})$ *represents the reward obtained by all agents together, when they execute the joint action* $\vec{a}$*, a state transition to* s' *occurs, and the joint message* $\vec{\sigma}$ *is sent.*

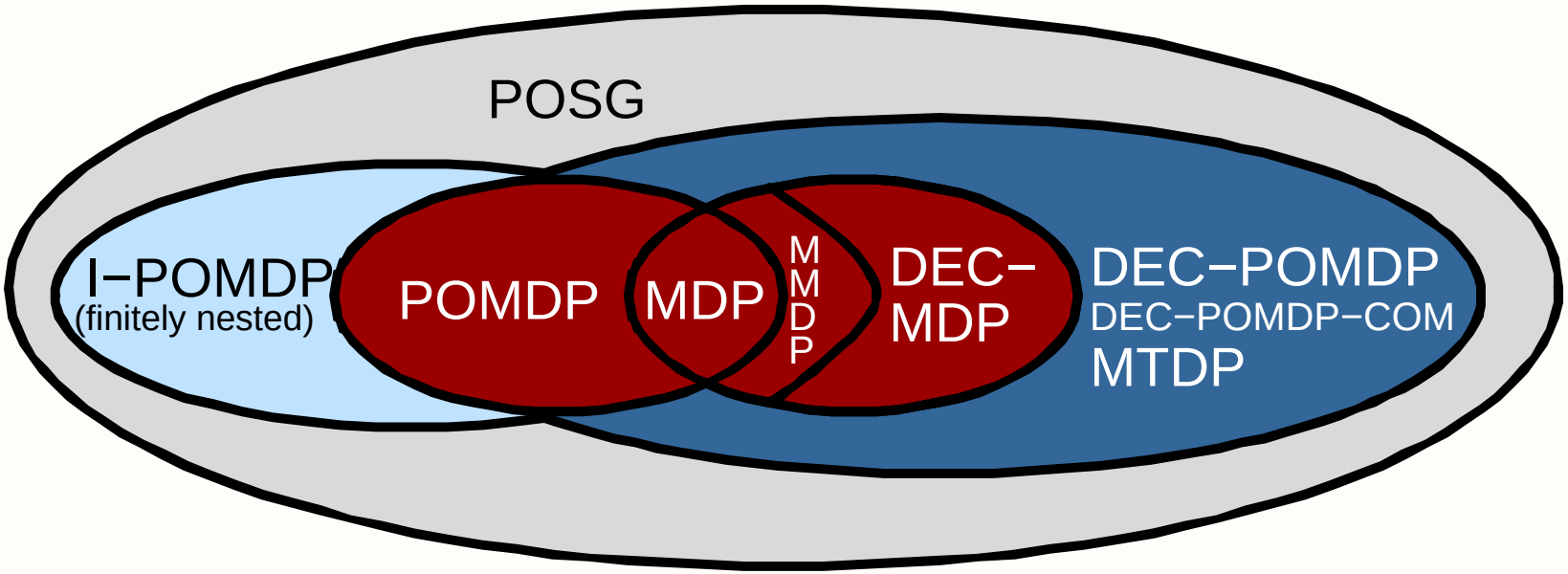
Interactive POMDPs

- Interactive POMDPs (I-POMDPs) extend state space with behavorial models of other agents (Gmytrasiewicz and Doshi, 2005).
- Agents maintain beliefs over physical and models of others.
  - ► Recursive modeling.
- When assuming a finite nesting, beliefs and value functions can be computed (approximately).
- Finitely nested I-POMDPs can be solved as a set of POMDPs.

INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Relationships among the models



Previous complexity results

Finite Horizon

MDP	P-complete (if $h < \|S\|$)	(Papadimitriou and Tsitsiklis, 1987)
POMDP	PSPACE-complete (if $h < \|S\|$)	(Papadimitriou and Tsitsiklis, 1987)

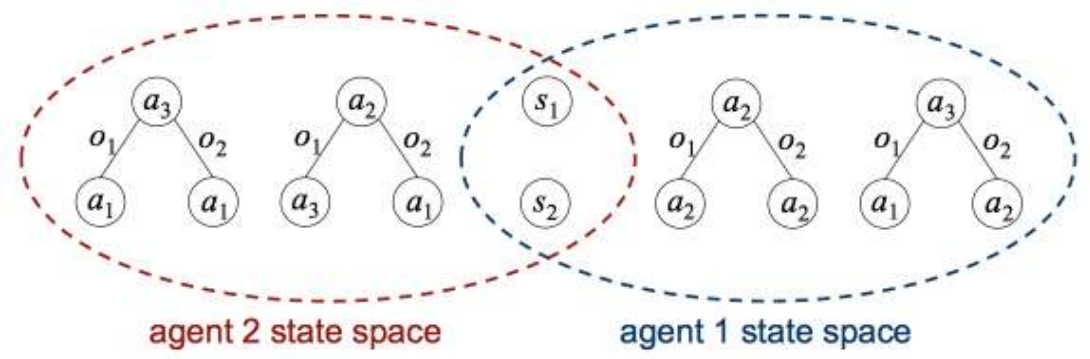
Infinite Horizon Discounted

MDP	P-complete	(Papadimitriou and Tsitsiklis, 1987)
POMDP	undecidable	(Madani et al., 1999)

DEC-POMDPs complexity

Intuition

- Agents must consider the choices of all others in addition to the state and action uncertainty present in POMDPs.
- This makes DEC-POMDPs much harder to solve (NEXP-complete).
- Solvable in nondeterminstic exponential time: Can guess a solution in exponential time and transform the DEC-POMDP into an exponentially bigger belief state MDP.
- NEXP-hardness: Reduction from tiling problem (each agent must place a tile based on local information and the result must be consistent).



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Upper bound for DEC-POMDPs

Theorem 1. *Finite-horizon DEC-POMDPs are in nondeterministic exponential time.*

Proof: The following process shows that a non-deterministic Turing machine can solve any instance of a DEC-POMDP$_n$ in at most exponential time.

1. Guess a joint policy and write it down in exponential time. This is possible, because a joint policy consists of n mappings from observation histories to actions. Since $h \leq |S|$, the number of possible histories is exponentially bounded by the problem description.
2. The DEC-POMDP together with the guessed joint policy can be viewed as an exponentially bigger POMDP using n-tuples of observations and actions.
3. In exponential time, convert all the observation sequences into a belief state.
4. In exponential time, compute transition probabilities and expected rewards for an exponentially bigger belief state MDP.
5. This MDP can be solved in polynomial time, which is exponential in the original problem description.

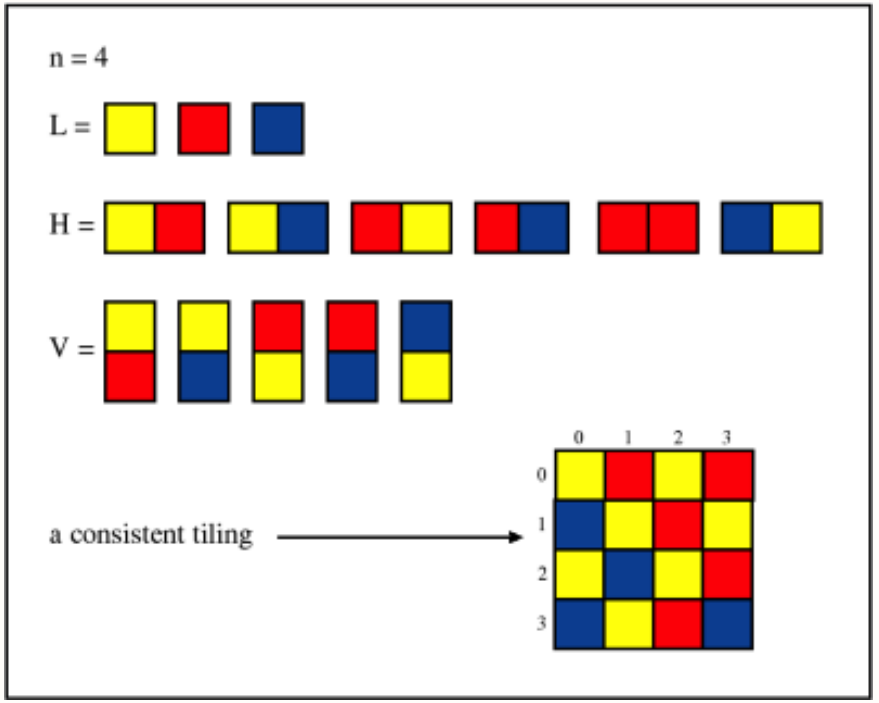
Thus, there is an accepting computation path in the non-deterministic machine if and only if there is a joint policy that can achieve reward K.

Lower bound for DEC-POMDPs

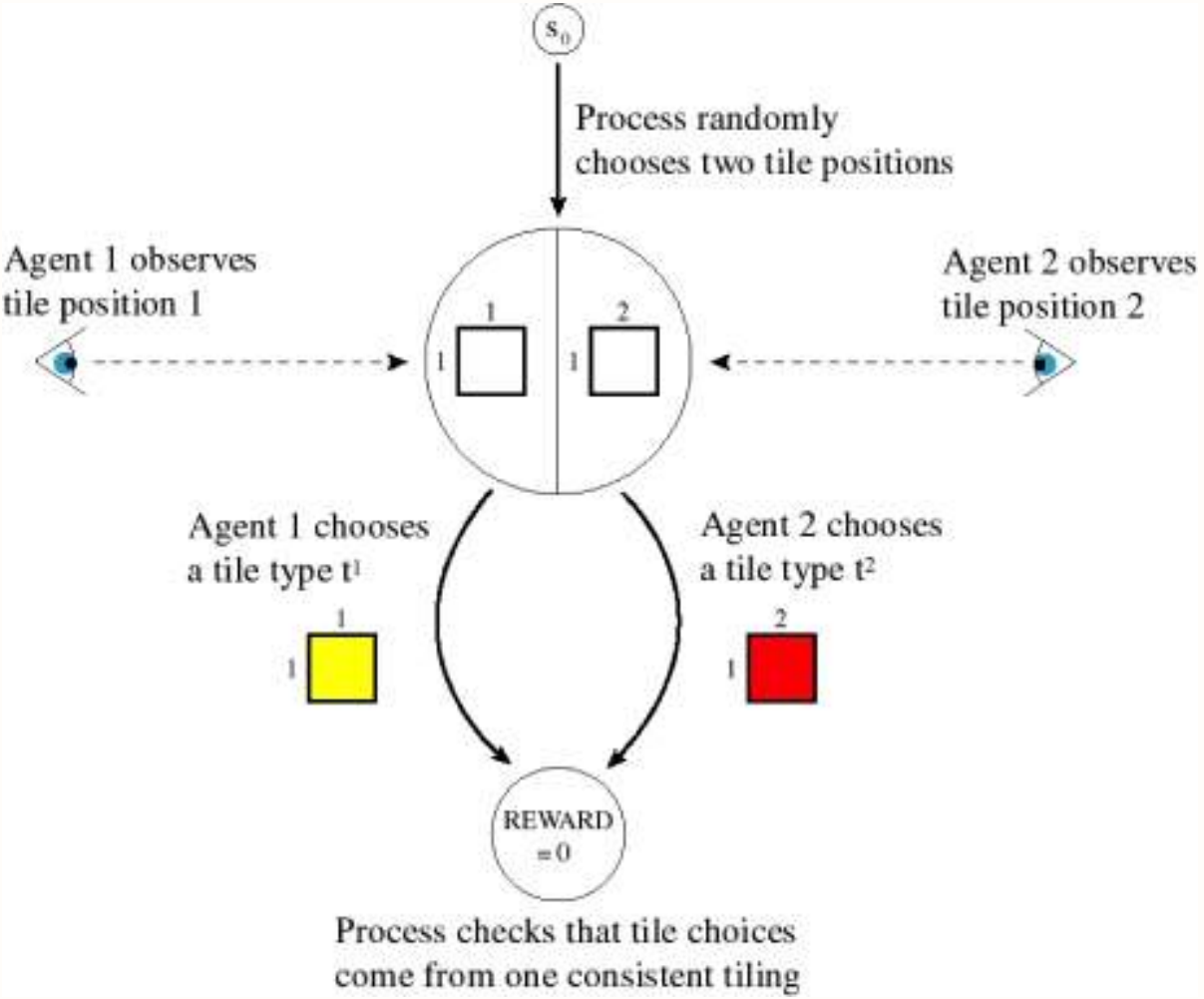
Theorem 2. *(Bernstein et al., 2002) Two-agent finite-horizon DEC-POMDPs are NEXP-hard.*

- Thus provably intractable (unlike POMDP)
- Probably doubly exponential (unlike POMDP)

Proof: By reduction from TILING



Proof of hardness



$\exists$ policy with expected reward 0 $\Leftrightarrow$ $\exists$ consistent tiling

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Proof of hardness

- Naive approach has a state for every pair of tile positions (exponential in size of instance!)
- Luckily, we need only remember info about relationship between positions in the state
- Generate positions bit-by-bit, and only remember key information:
  - Are they equal?
  - Are they horizontally adjacent?
  - Are they vertically adjacent?
- Q.E.D.

Joint observability

Definition 10. *Joint full observability $\equiv$ collective observability. A DEC-POMDP is jointly fully observable if the n-tuple of observations made by all the agents uniquely determine the current global state. That is, if $O(\vec{o}|\vec{a}, s') > 0$ then $P(s'|\vec{o}) = 1$.*

Definition 11. *A decentralized Markov decision process (**DEC-MDP**) is a DEC-POMDP with joint full observability.*

A stronger result: The problem is NEXP-hard even when the state is jointly observed! That is, two-agent finite-horizon DEC-MDPs are NEXP-hard.

Classes of DEC-POMDPs

Definition 12. *A **factored** n**-agent DEC-MDP** is a DEC-MDP for which the world state can be factored into $n+1$ components, $S = S_0 \times S_1 \times \ldots \times S_n$.*

Definition 13. *A factored, n-agent DEC-MDP is said to be **locally fully observable** if each agent observes its own state component*

$$\forall o_i \, \exists \hat{s}_i : Pr(\hat{s}_i | o_i) = 1.$$

Definition 14. *Local state/observation/action $\hat{s}_i \in S_i \times S_0$ is referred to as the local state, $a_i \in A_i$ as the local action, and $o_i \in \Omega_i$ as the local observation for agent i.*

Definition 15. ***Full observability*** $\equiv$ ***individual observability***. *A DEC-POMDP is fully observable if there exists a mapping for each agent i, $f_i : \Omega_i \rightarrow S$ such that whenever $O(\vec{o}|s, \vec{a}, s')$ is non-zero then $f_i(o_i) = s'$.*

Classes of DEC-POMDPs

Definition 16. ***MMDP*** *A multi-agent Markov decision process is a DEC-POMDP with full observability (Boutilier, 1996).*

Definition 17. *A factored, n-agent DEC-MDP is said to be* ***transition independent*** *if there exists P_0 through P_n such that*

$$Pr(s'_i|(s_0,\ldots,s_n),\vec{a},(s'_1,\ldots,s'_{i-1},s'_{i+1},\ldots,s'_n)) = \begin{cases} P_0(s'_0|s_0) & i=0 \\ P_i(s'_i|\hat{s}_i,a_i,s'_0) & 1\leq i\leq n \end{cases} \quad (1)$$

Definition 18. *A factored, n-agent DEC-MDP is said to be* ***observation independent*** *if there exists O_1 through O_n such that:*

$$Pr(o_i|(s_0,\ldots,s_n),\vec{a},(s'_0,\ldots,s'_n),(o_1,\ldots,o_{i-1},o_{i+1},\ldots,o_n)) = Pr(o_i,\hat{s}_i,a_i,\hat{s}_i{}').$$

Classes of DEC-POMDPs

Definition 19. *A factored, n-agent DEC-MDP is said to be **reward independent** if there exist f and R_1 through R_n such that*

$$R((s_0, \ldots, s_n), \vec{a}, (s'_0, \ldots, s'_n)) = f(R_1(\hat{s}_1, a_1, \hat{s}'_1), \ldots, R_n(\hat{s}_n, a_n, \hat{s}'_n)) \quad (2)$$

and

$$R_i(\hat{s}_i, a_i, \hat{s}'_i) \leq R_i(\hat{s}_i, a'_i, \hat{s}''_i) \Leftrightarrow$$
$$f(R_1 \ldots R_i(\hat{s}_i, a_i, \hat{s}'_i) \ldots R_n) \leq f(R_1 \ldots R_i(\hat{s}_i, a'_i, \hat{s}''_i) \ldots R_n) \quad (3)$$

Theorem 3. *If a DEC-MDP has independent observations and transitions, then the DEC-MDP is locally fully observable.*

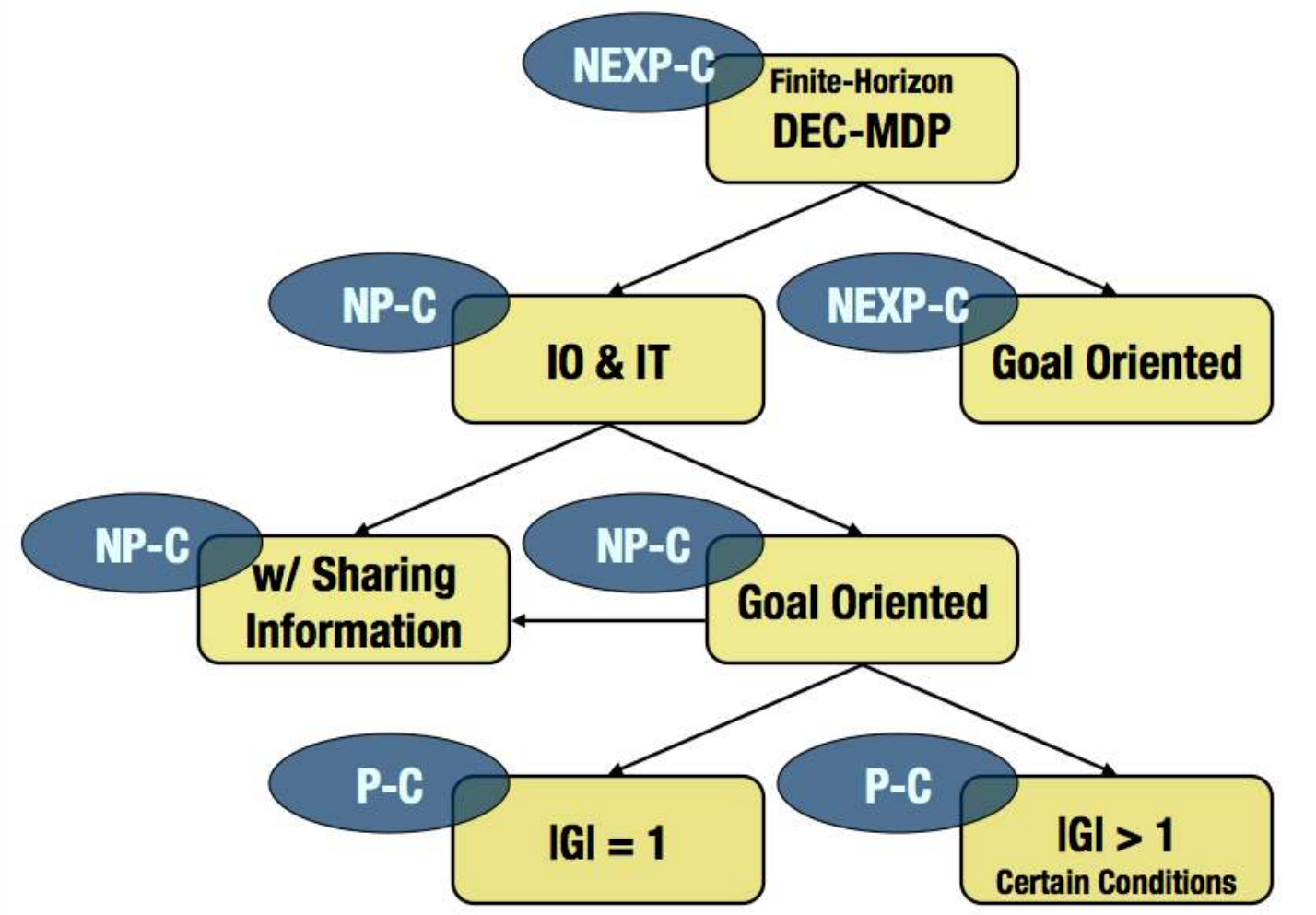
INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Observability, communication and complexity

Observability	General Communication	Free Communication
Full	MMDP (P-complete)	MMDP (P-complete)
Joint Full	DEC-MDP (NEXP-complete)	MMDP (P-complete)
Partial	DEC-POMDP (NEXP-complete)	MPOMDP (PSPACE-complete)

More complexity results



(Goldman and Zilberstein, 2004)

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Algorithms

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Algorithms outline

1. Optimal solutions
2. Bottom-up algorithms
3. Top-down algorithms
4. Other finite-horizon algorithms
5. Infinite-horizon algorithms
6. Algorithms for subclasses

INSTITUTO
SUPERIOR
TÉCNICO

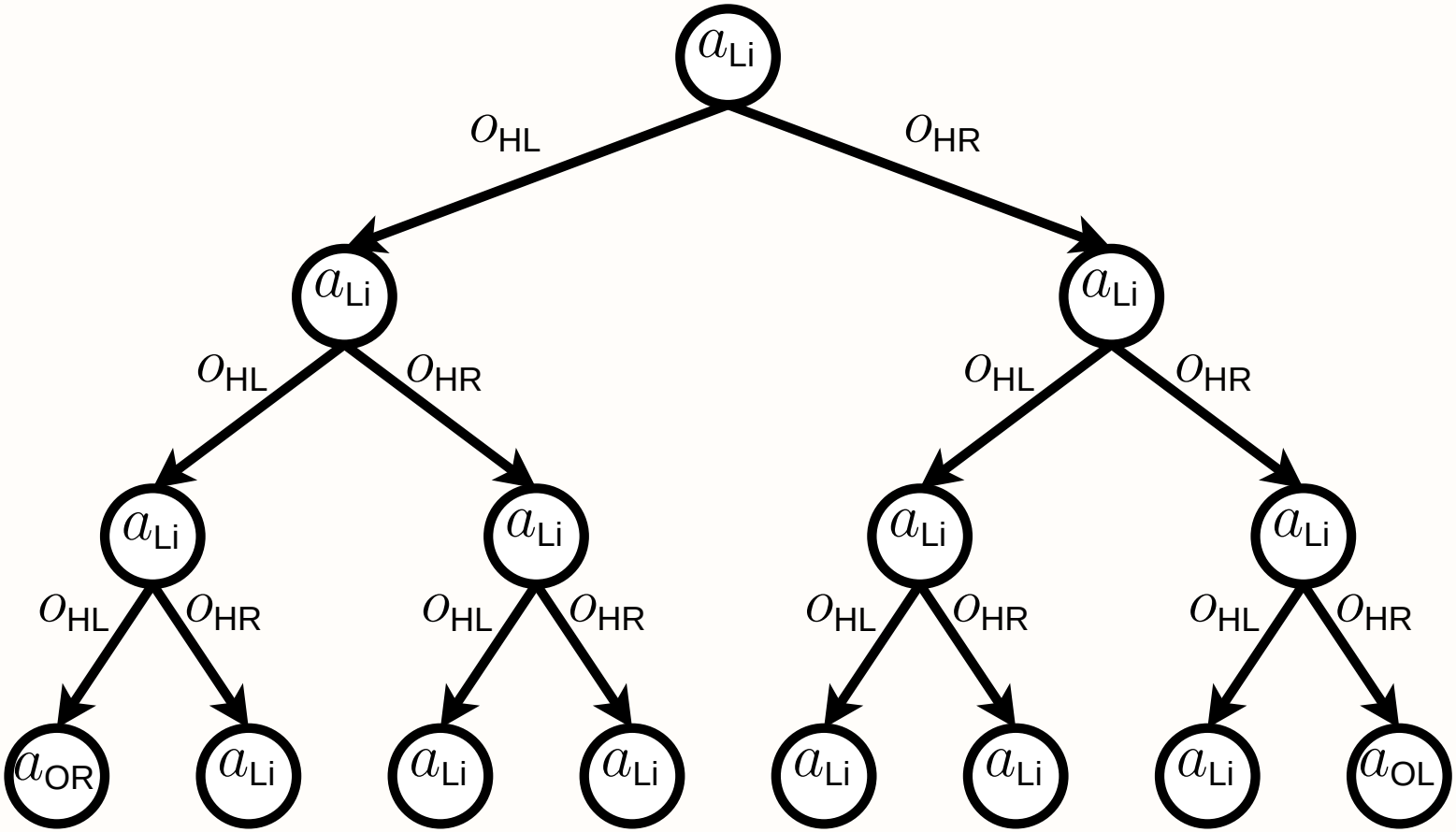
University *of*
Massachusetts
Amherst

Optimal DEC-POMDP solutions

- Reminder
  - ▶ No beliefs over states available.
  - ▶ No piecewise linear convex value functions.
- MDP/POMDP algorithms do not transfer directly. . .
  - ▶ . . . but ideas do transfer.

Optimal DEC-POMDP solutions

Example: optimal policy for Dec-Tiger, $h = 4$ (shown for 1 agent)



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

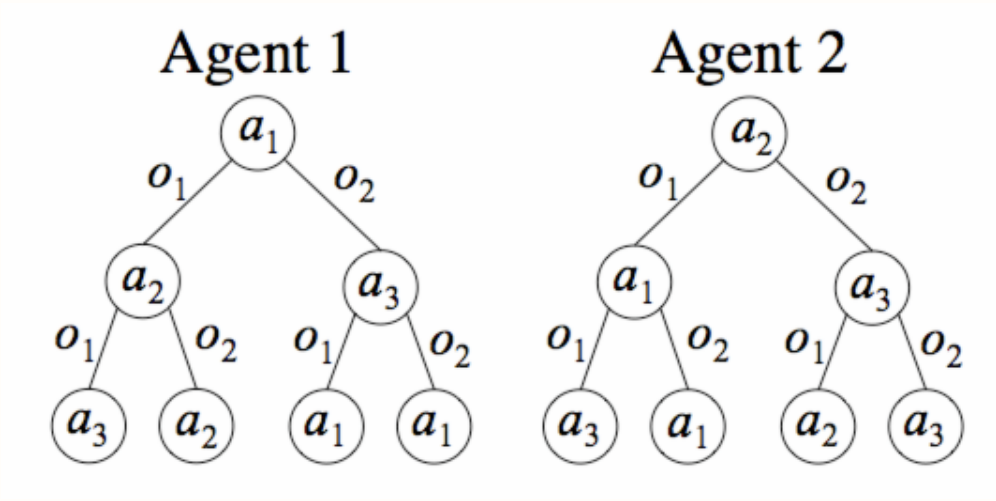
Axes

- Algorithms can be classified according along various axes:
  - Optimal vs. Approximate
  - General Model vs. Subclasses
  - Infinite horizon vs. Finite horizon
- Axes will be indicated as follows:

Optimal / Approximate, General / Subclasses, Infinite / Finite horizon

Bottom up approaches

- Build the policies up for each agent simultaneously
- Begin on the last step (single action) and continue until the first
- When done, choose highest value set of trees for any initial state



INSTITUTO SUPERIOR TÉCNICO

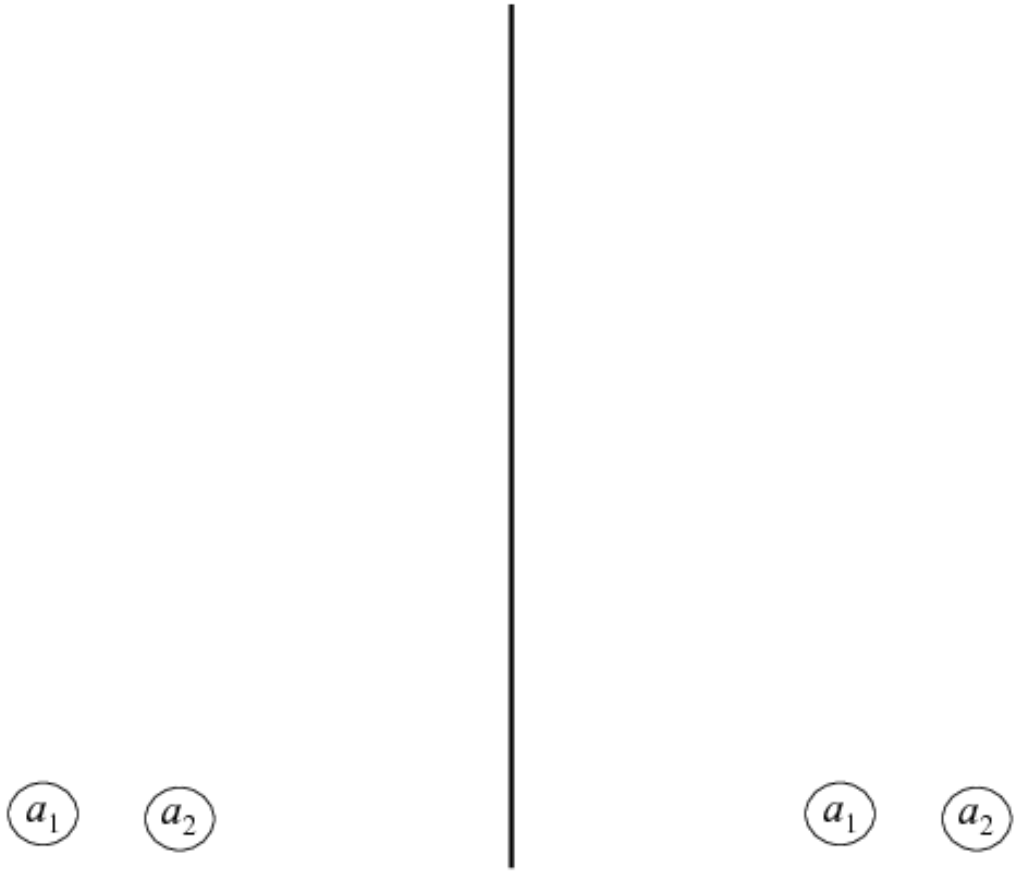
University *of* Massachusetts Amherst

Exhaustive search

- Construct all possible policies for the set of agents
- Do this in a bottom up fashion
- Trivially includes an optimal set of trees when finished

Optimal, General, Finite-horizon

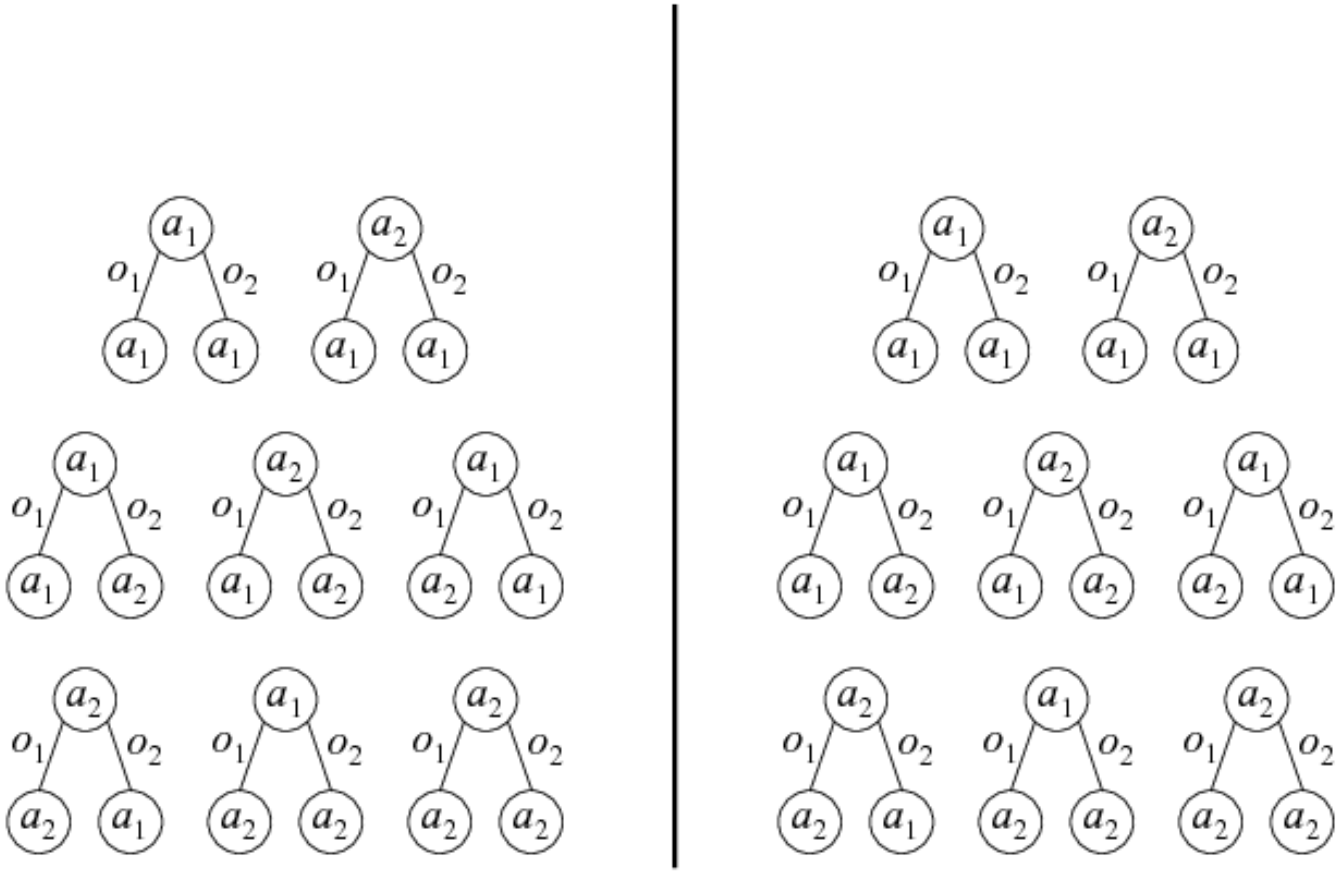
Exhaustive search example (2 agents)



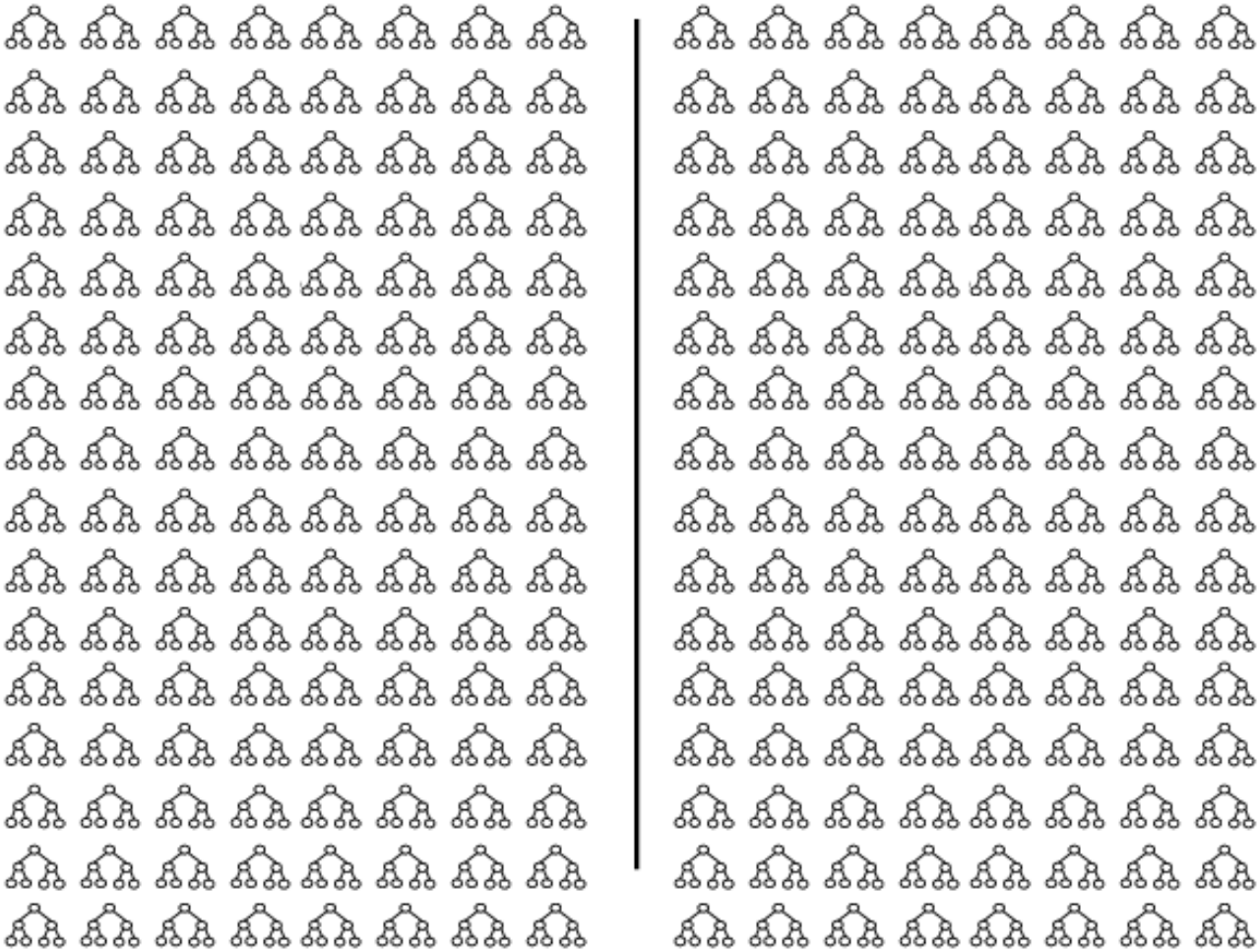
INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Exhaustive search example (2 agents)



Exhaustive search example (2 agents)



INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Exhaustive search summary

- Can find an optimal set of trees
- Number of each agent's trees grows exponentially at each step
- Many trees will not contribute to an optimal solution
- Can we reduce the number of trees we consider?

Dynamic programming

Add a pruning step to exhaustive search

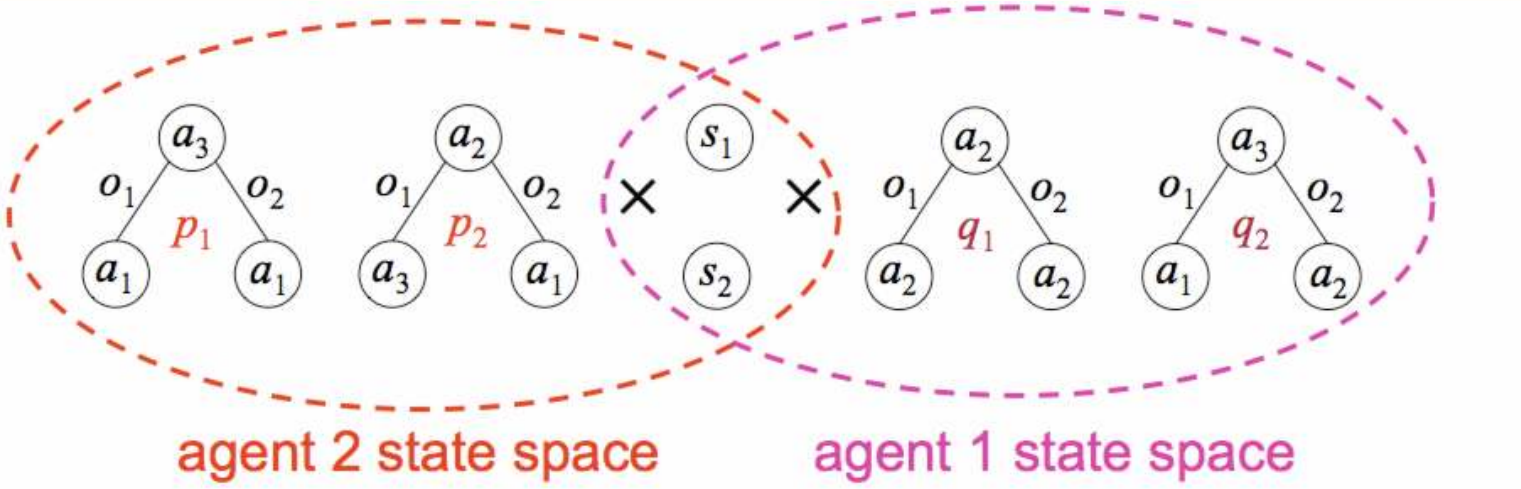
- At each step, remove trees that will never be part of optimal solution
- Uses linear programming (LP) to prune these dominated trees
- Ensures any policy that can contribute to optimal is not removed

(Hansen et al., 2004)

Optimal, General, Finite-horizon

Pruning trees

- Prune over multiagent belief space (policies of the other agents and states of the system)
- Retains optimal policy for any belief state



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Linear program for pruning a tree

Variables: $\epsilon, x(\hat{q}_i)$

Objective: Maximize ϵ

Improvement constraints:

$$\forall s, q_{-i} \quad V(s, q_i, q_{-i}) + \epsilon \leq \sum_{\hat{q}_i} x(\hat{q}_i) V(s, \hat{q}_i, q_{-i})$$

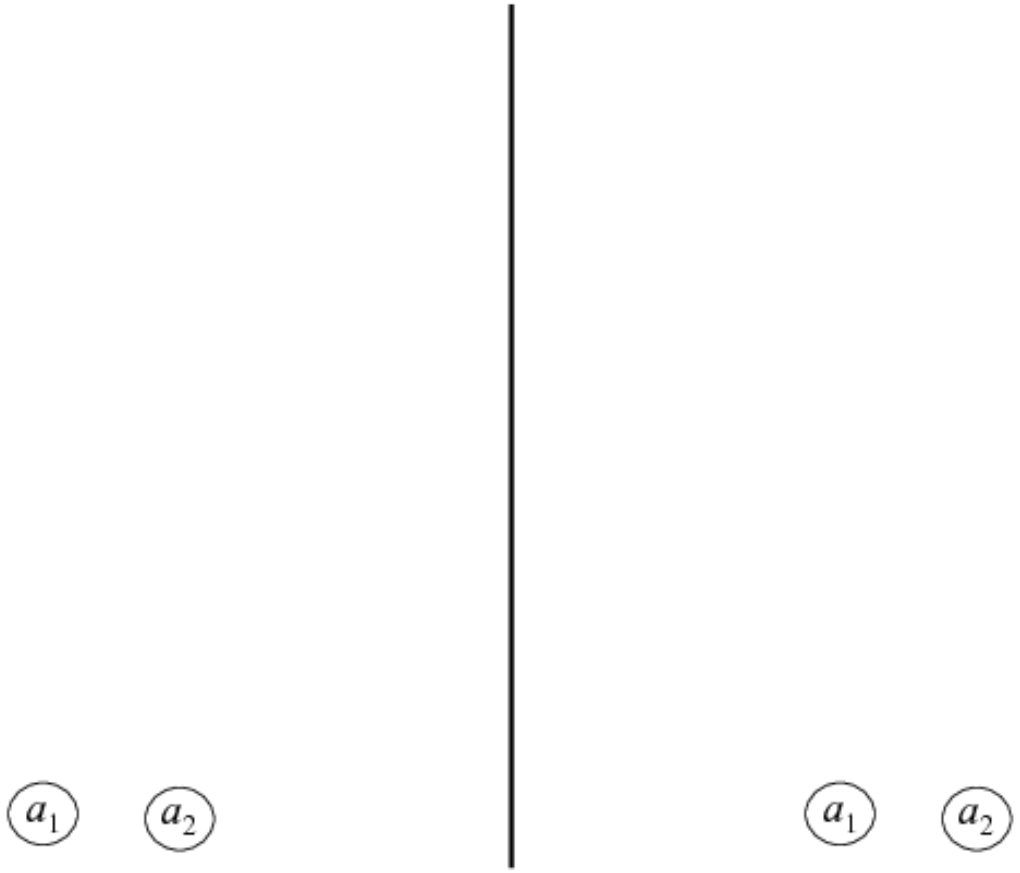
Probability constraints: $\sum_{\hat{q}_i} x(\hat{q}_i) = 1, \qquad \forall \hat{q}_i \quad x(\hat{q}_i) \geq 0$

Prune tree q_i if there is a distribution of other trees $x(\hat{q}_i)$ that has a higher value for all system states and trees of the other agents q_{-i}.

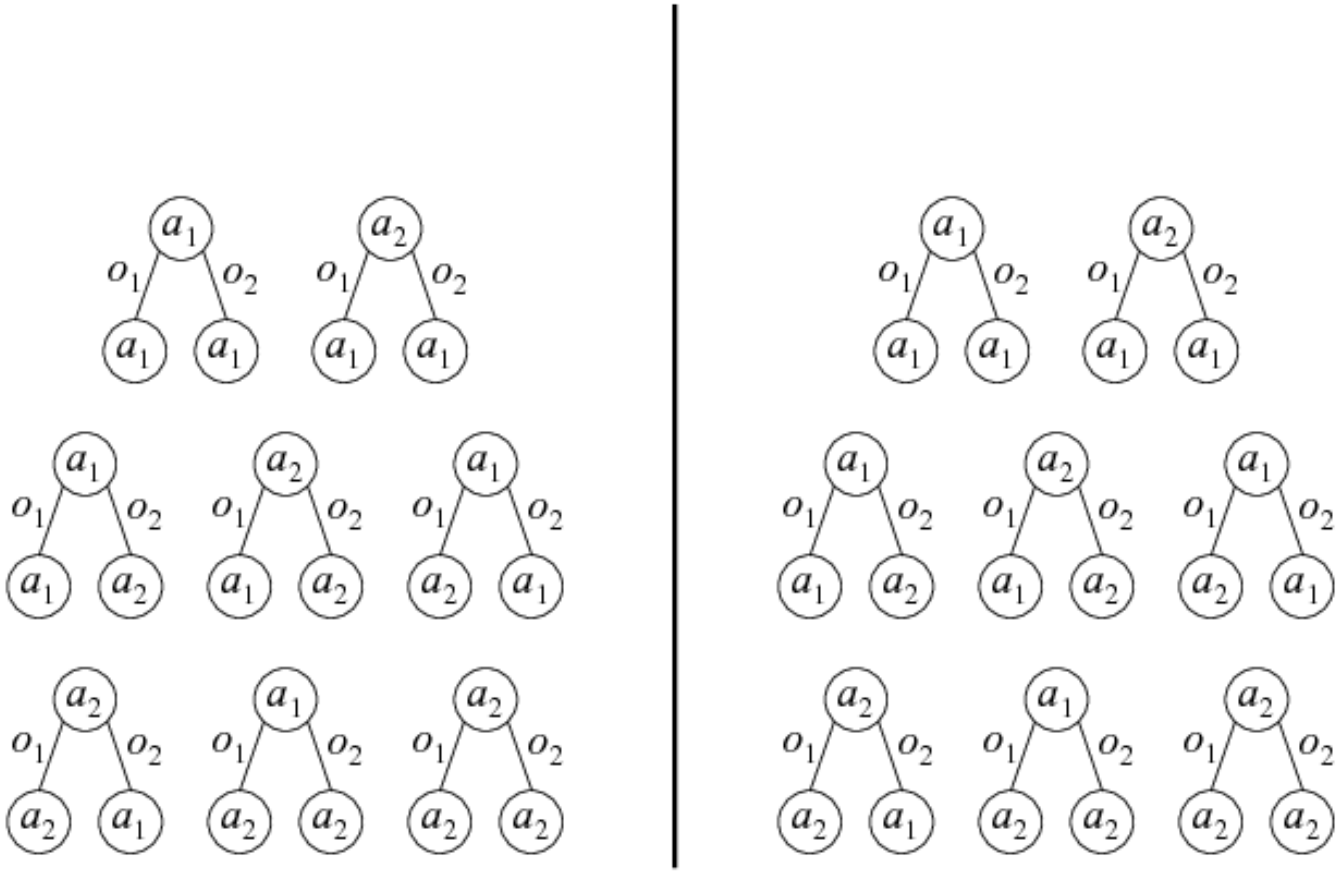
INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Dynamic programming example (2 agents)



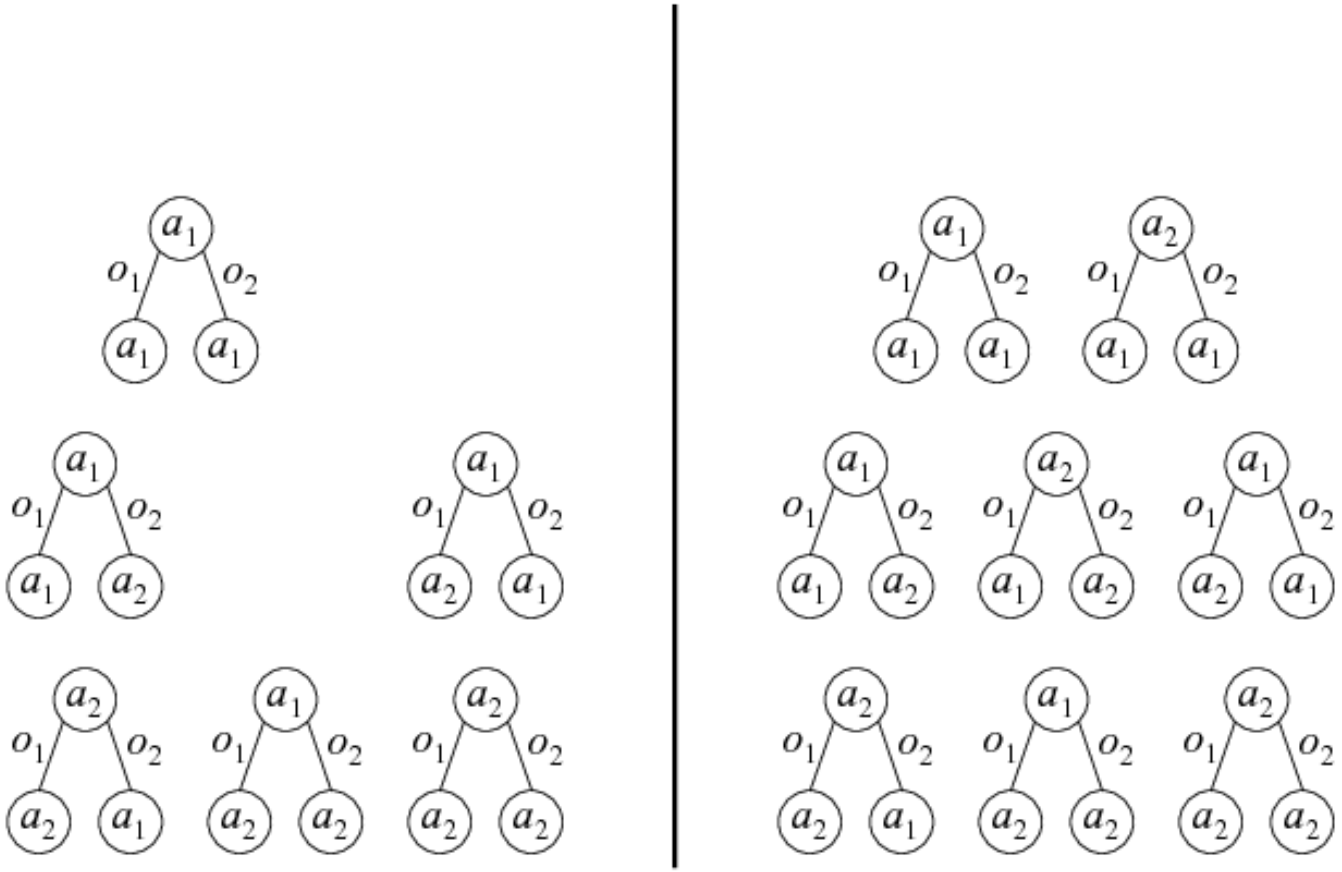
Dynamic programming example (2 agents)



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

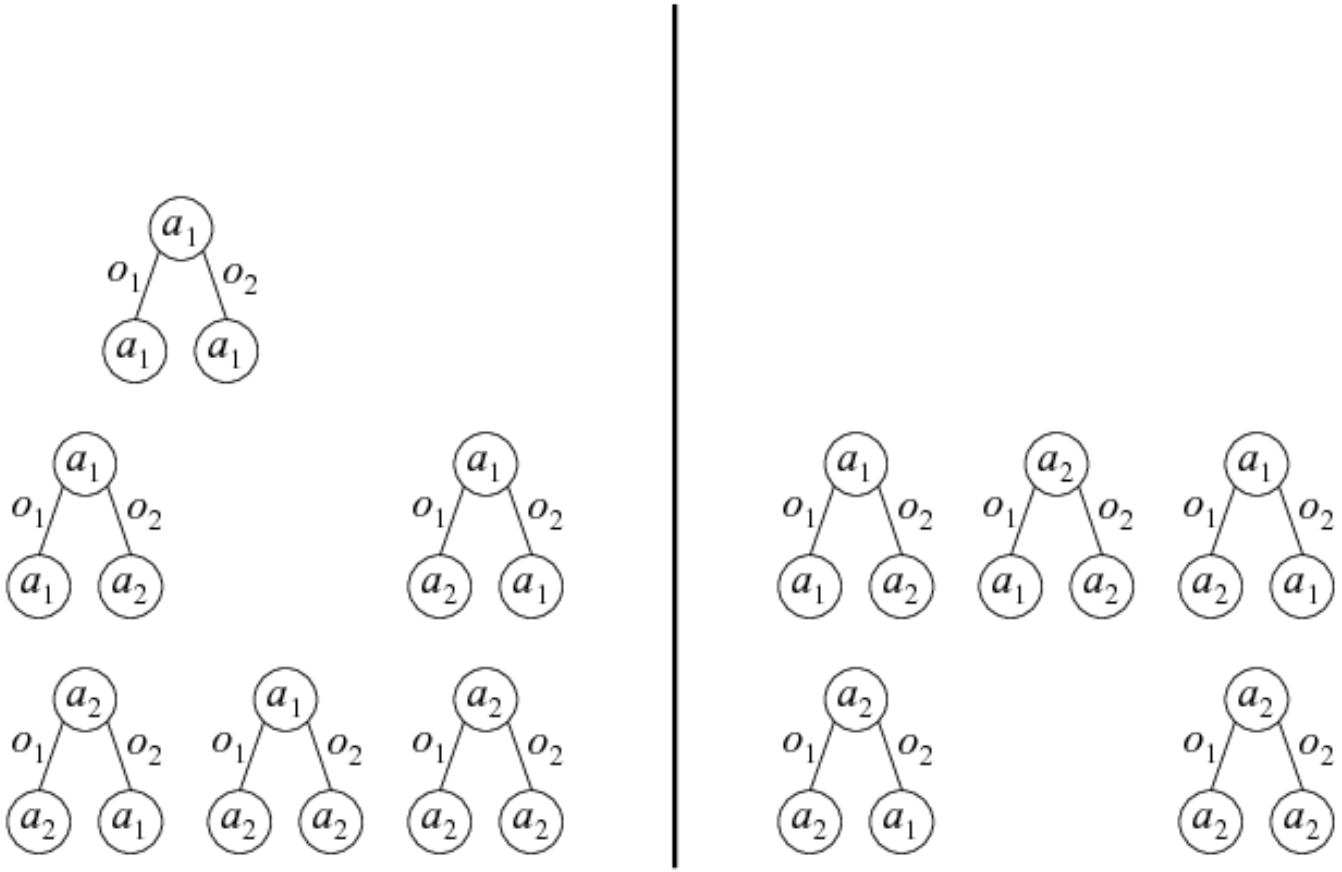
Dynamic programming example (2 agents)



INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

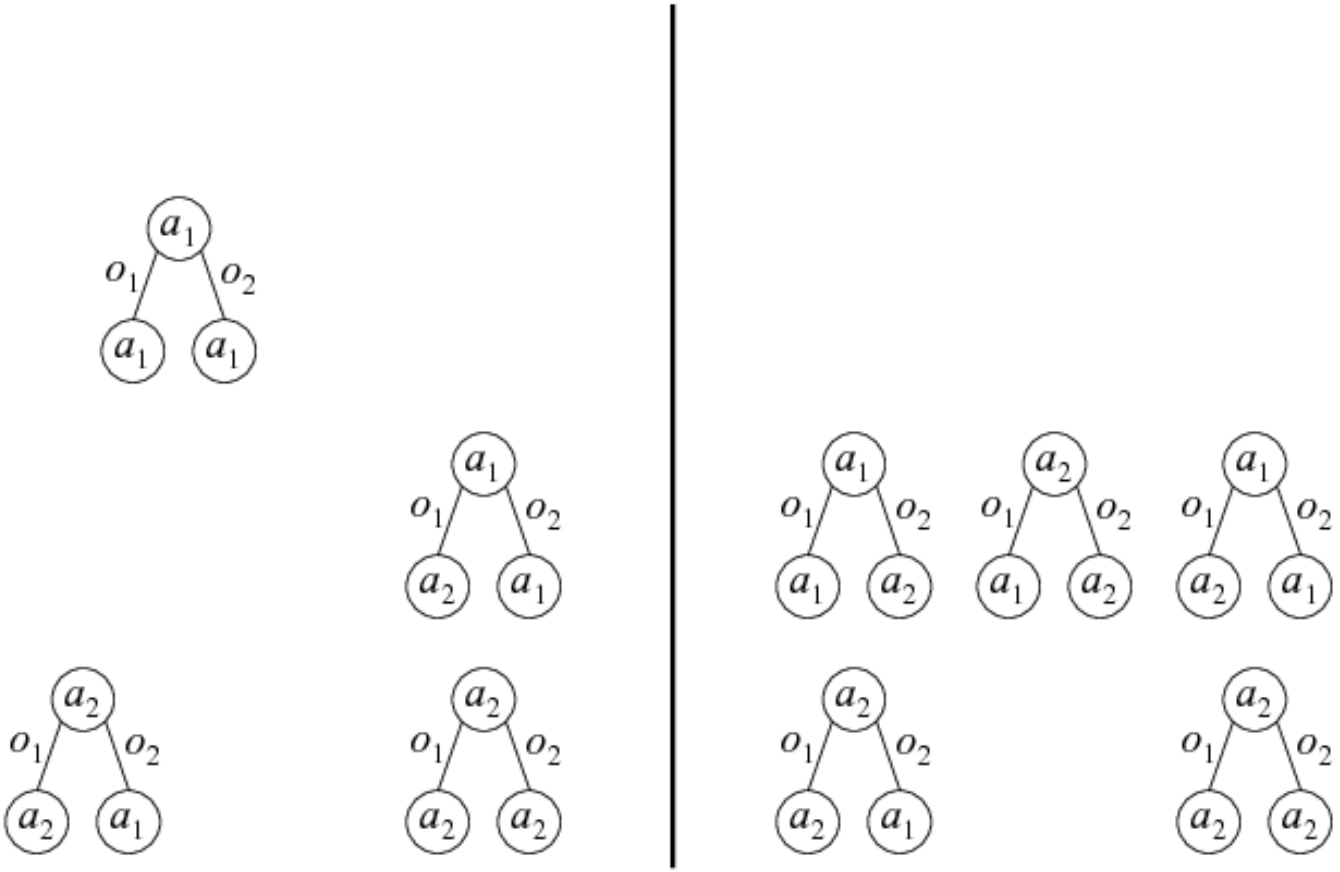
Dynamic programming example (2 agents)



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

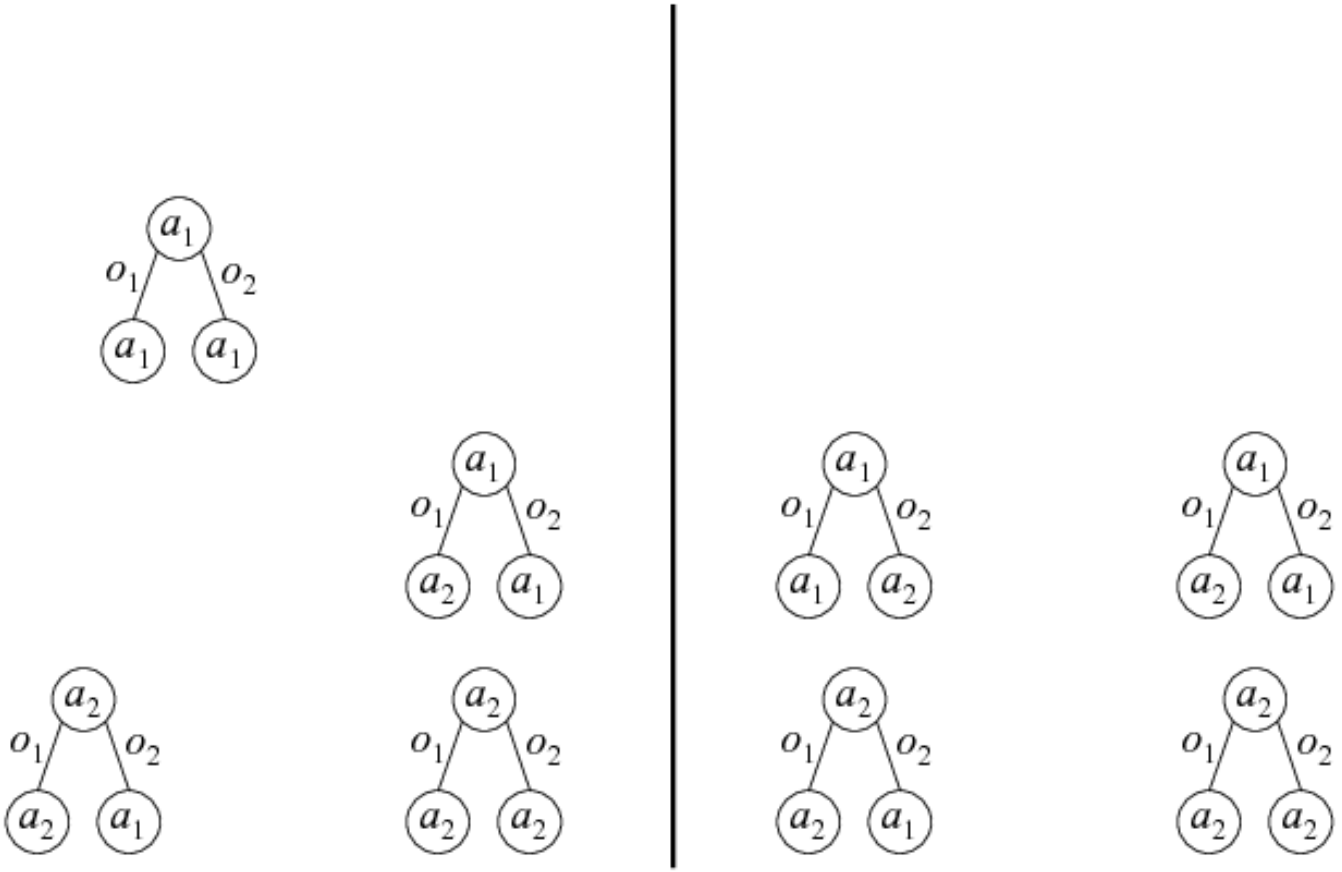
Dynamic programming example (2 agents)



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

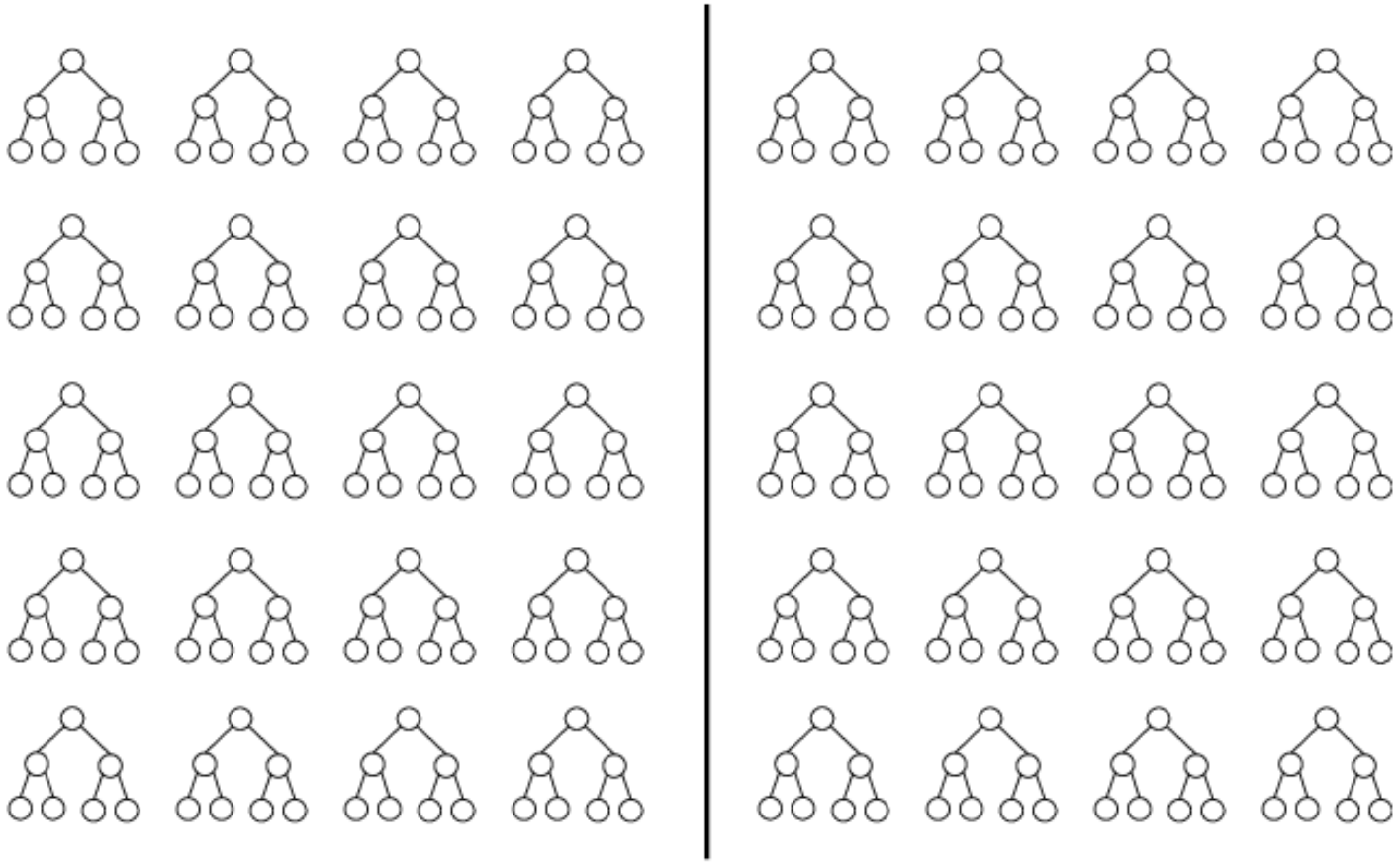
Dynamic programming example (2 agents)



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Dynamic programming example (2 agents)



INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Dynamic programming summary

- Will produce optimal solution for any initial state
- Improves scalability of exhaustive backups with pruning at each step
- Still limited to small problems
- For POSGs, iterative removal of very weakly dominated strategies

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Approximate bottom-up algorithms

Optimal algorithms have theoretical value, but are generally intractable in practice

- Approximate bottom up algorithms can provide higher value and solve larger problems
  - ▶ Joint Equilibrium Search for Policies (JESP)
  - ▶ Memory Bounded Dynamic Programming (MBDP) (will be discussed later)

Approximate, General, Finite-horizon

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Joint Equilibrium Search for Policies (JESP)

Instead of exhaustive search, find best response

Algorithm 3: JESP (Nair et al., 2003)

Start with policy for each agent

while *not converged* **do**

  for $i = 1$ *to* n **do**

    Fix other agent policies

    Find a best response policy for agent i

Approximate, General, Finite-horizon

JESP summary

- Find a locally optimal set of policies
- Worst case complexity is the same as exhaustive search, but in practice is much faster
- Can also incorporate dynamic programming to speed up finding best responses
  - Fix policies of other agents
  - Generate reachable belief states from initial state b_0
  - Build up policies from last step to first
  - At each step, choose subtrees that maximize value at reachable belief states

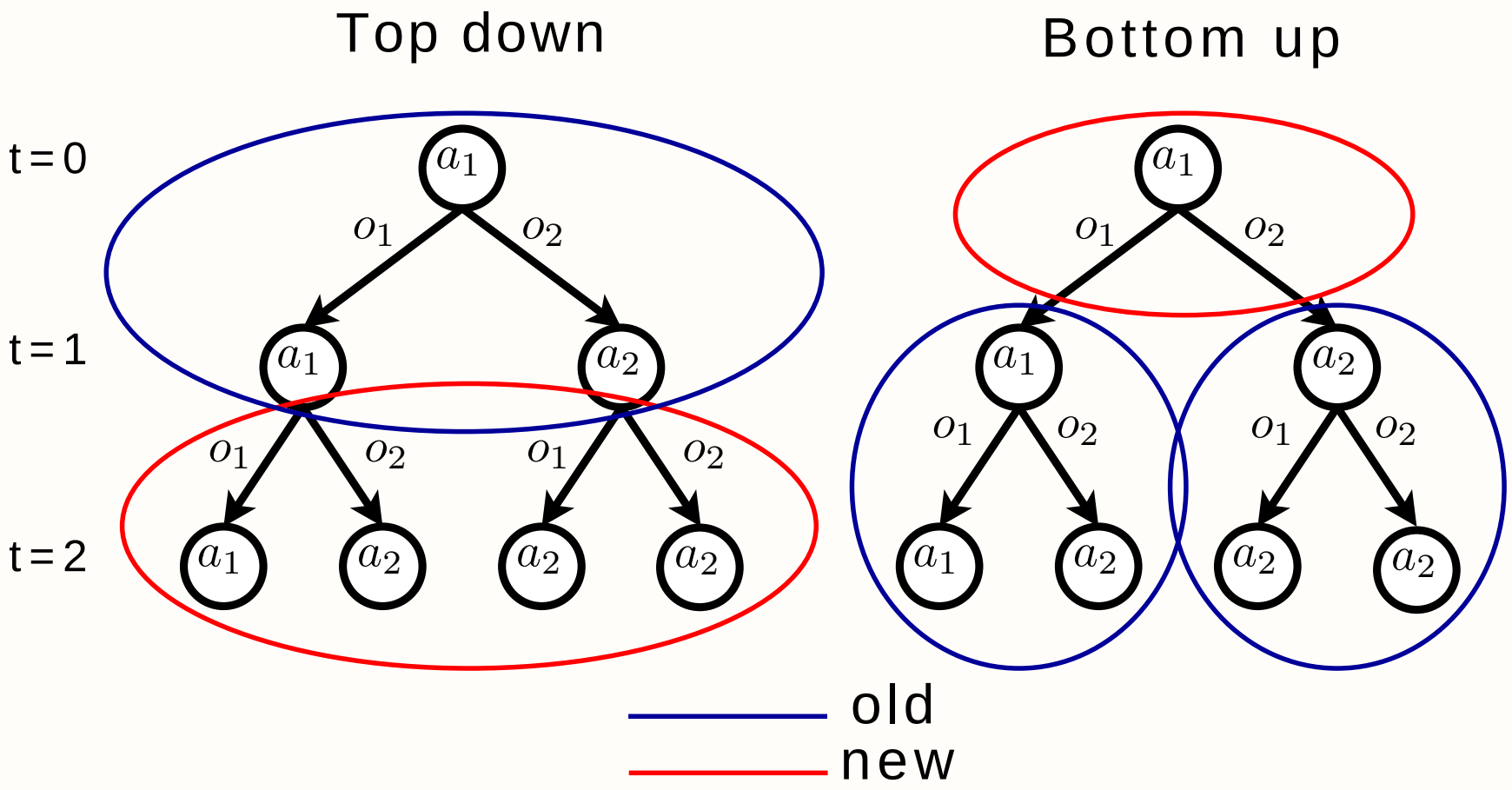
Bottom up summary

- Build agents polices simultaneously from the last step until the first
- Typically build policies for any initial state
- Optimal algorithms are not very scalable
- Approximate algorithms use locally optimal solutions and top-down heuristics (discussed later)
- Improve scalability by
  - Compressing policies at each step (Boularias and Chaib-draa, 2008)
  - Not generating tree sets exhaustively (Amato et al., 2009)

INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Top down approaches



Multiagent A*

- MAA*: top-down heuristic policy search (Szer et al., 2005).
  - Requires an admissable heuristic function.
  - A*-like search over partially specified joint policies:

$$\varphi^t = (\delta^0, \delta^1, \ldots, \delta^{t-1}),$$

$$\delta^t = (\delta_0^t, \ldots, \delta_n^t) \qquad \delta_i^t : \vec{\mathcal{O}}_i^t \rightarrow A_i$$

- Heuristic value for φ^t:

$$\underbrace{\widehat{V}(\varphi^t)}_{F} = \underbrace{V^{0\ldots t-1}(\varphi^t)}_{G} + \underbrace{\widehat{V}^{t\ldots h-1}}_{H}$$

  - If $\widehat{V}^{t\ldots h-1}$ is admissible (overestimation), so is $\widehat{V}(\varphi^t)$.

Optimal, General, Finite-horizon

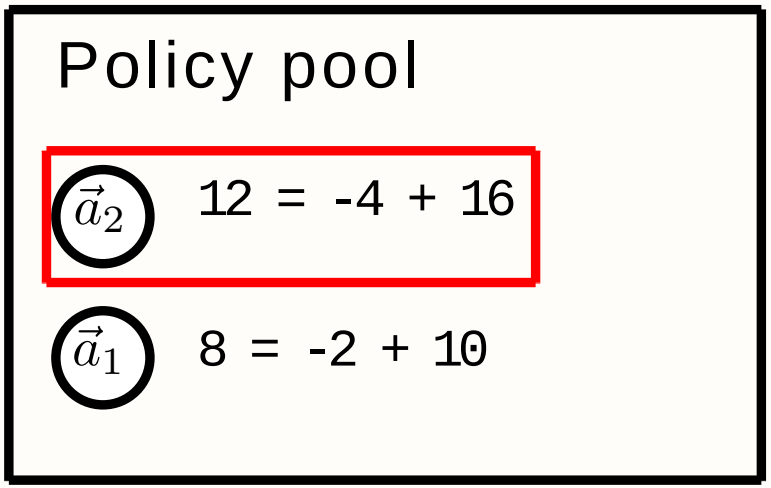
Multiagent A*: example

Policy pool

$\vec{a}_2$ 12 = -4 + 16

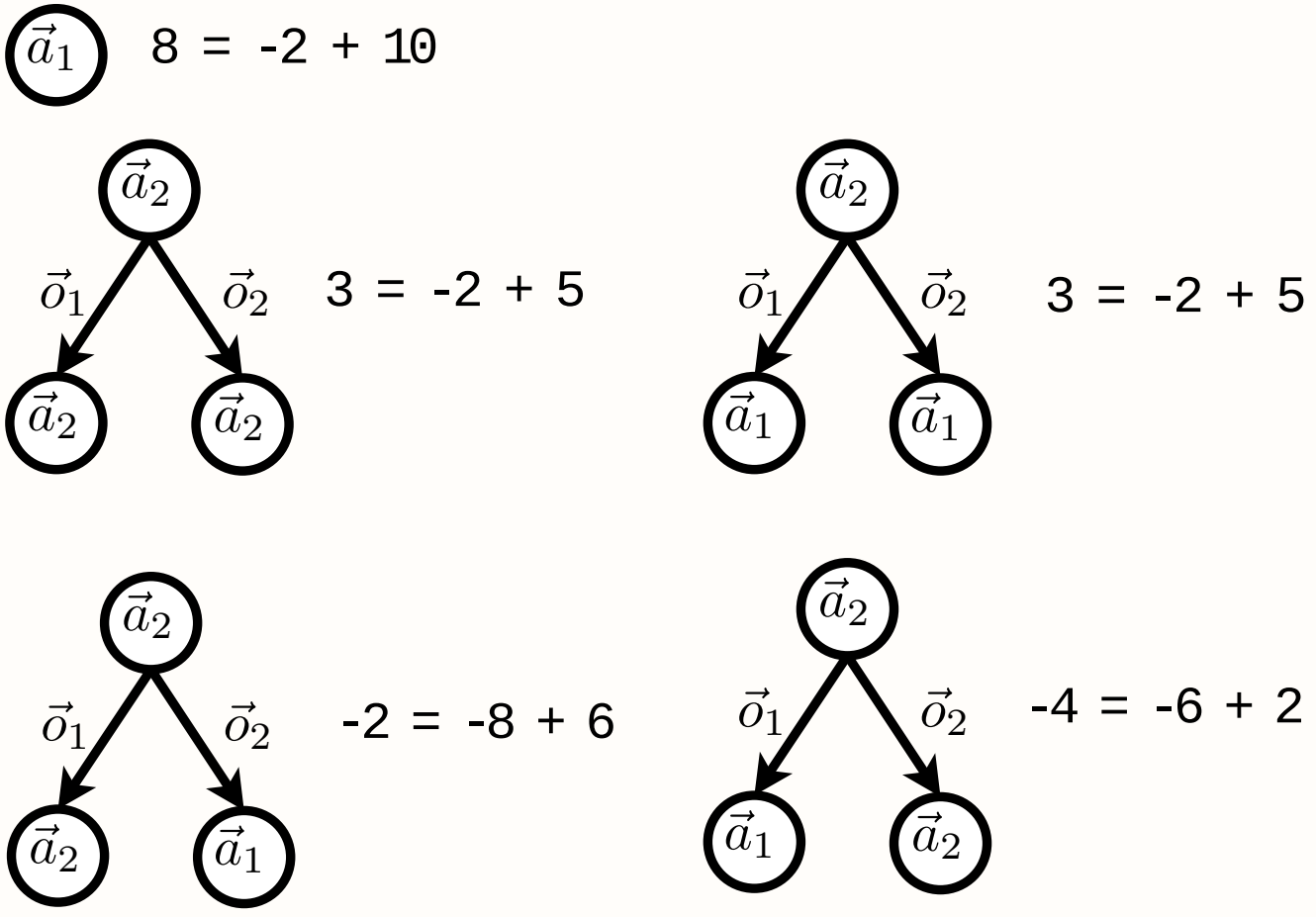
$\vec{a}_1$ 8 = -2 + 10

Multiagent A*: example



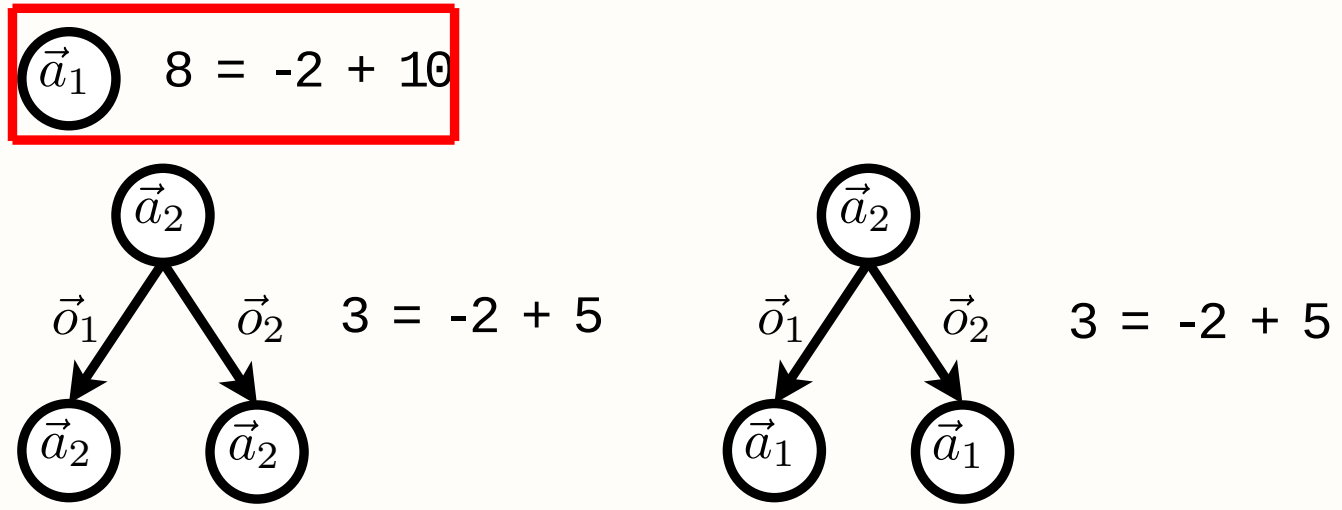
Multiagent A*: example

Policy pool

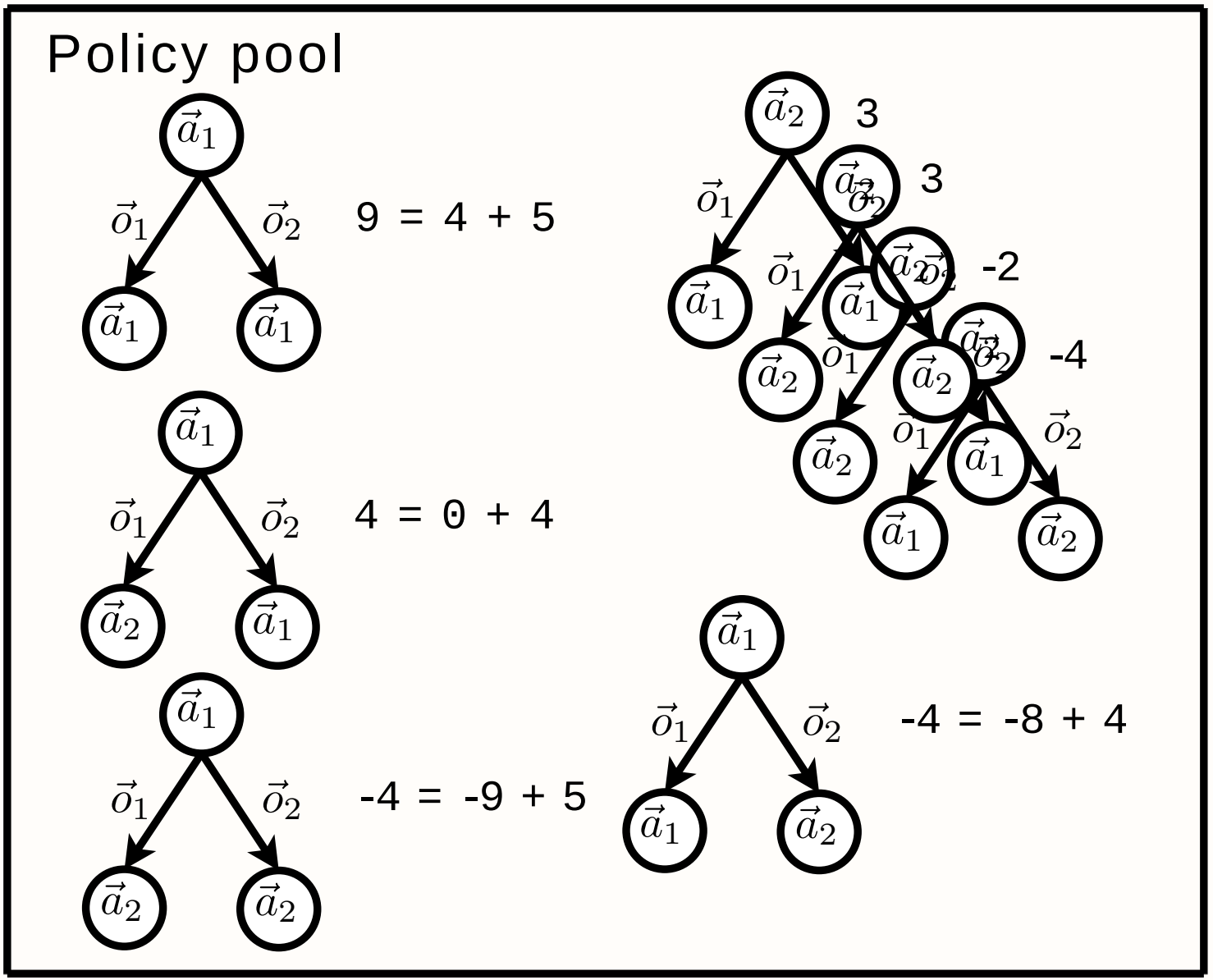


Multiagent A*: example

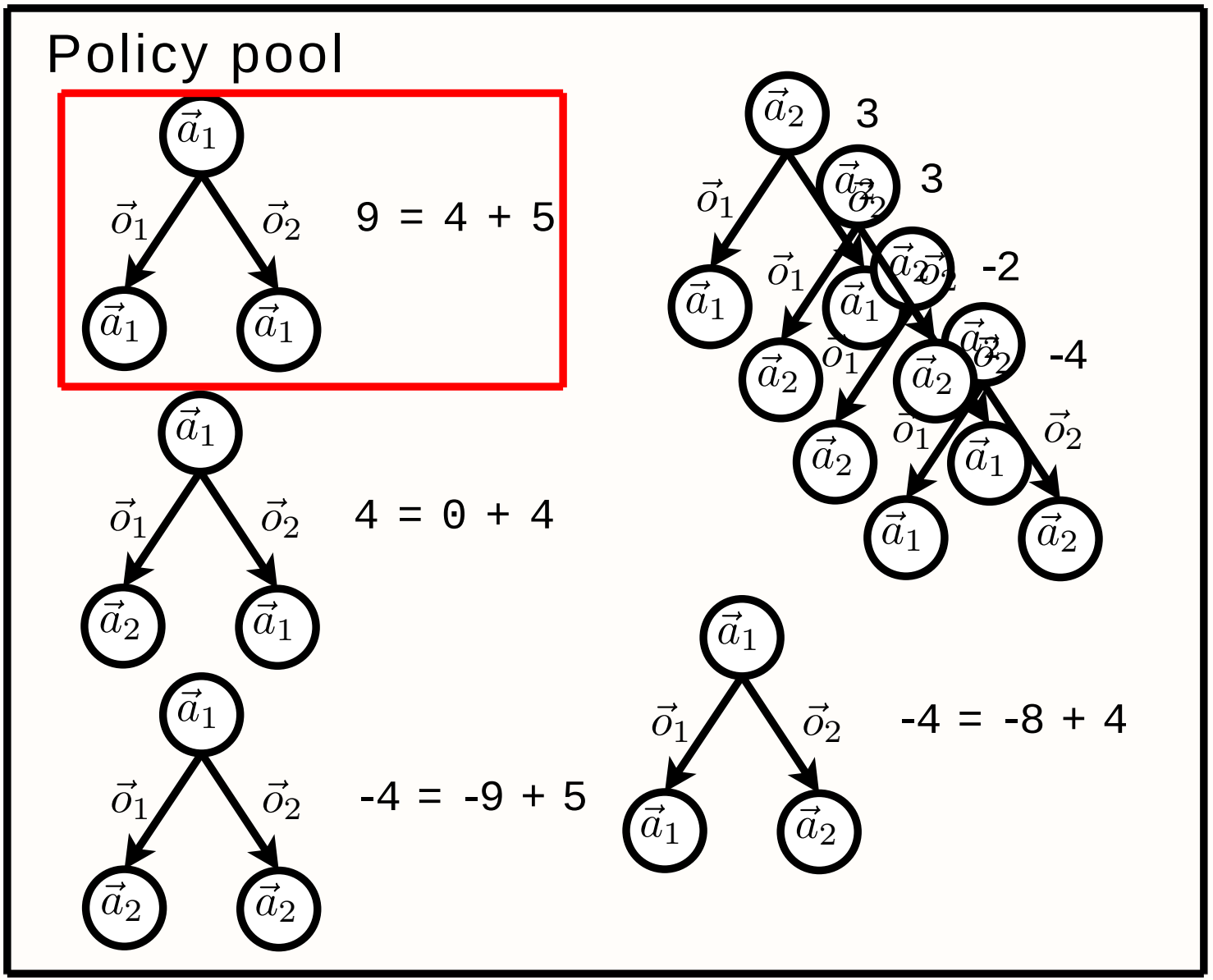
Policy pool



Multiagent A*: example



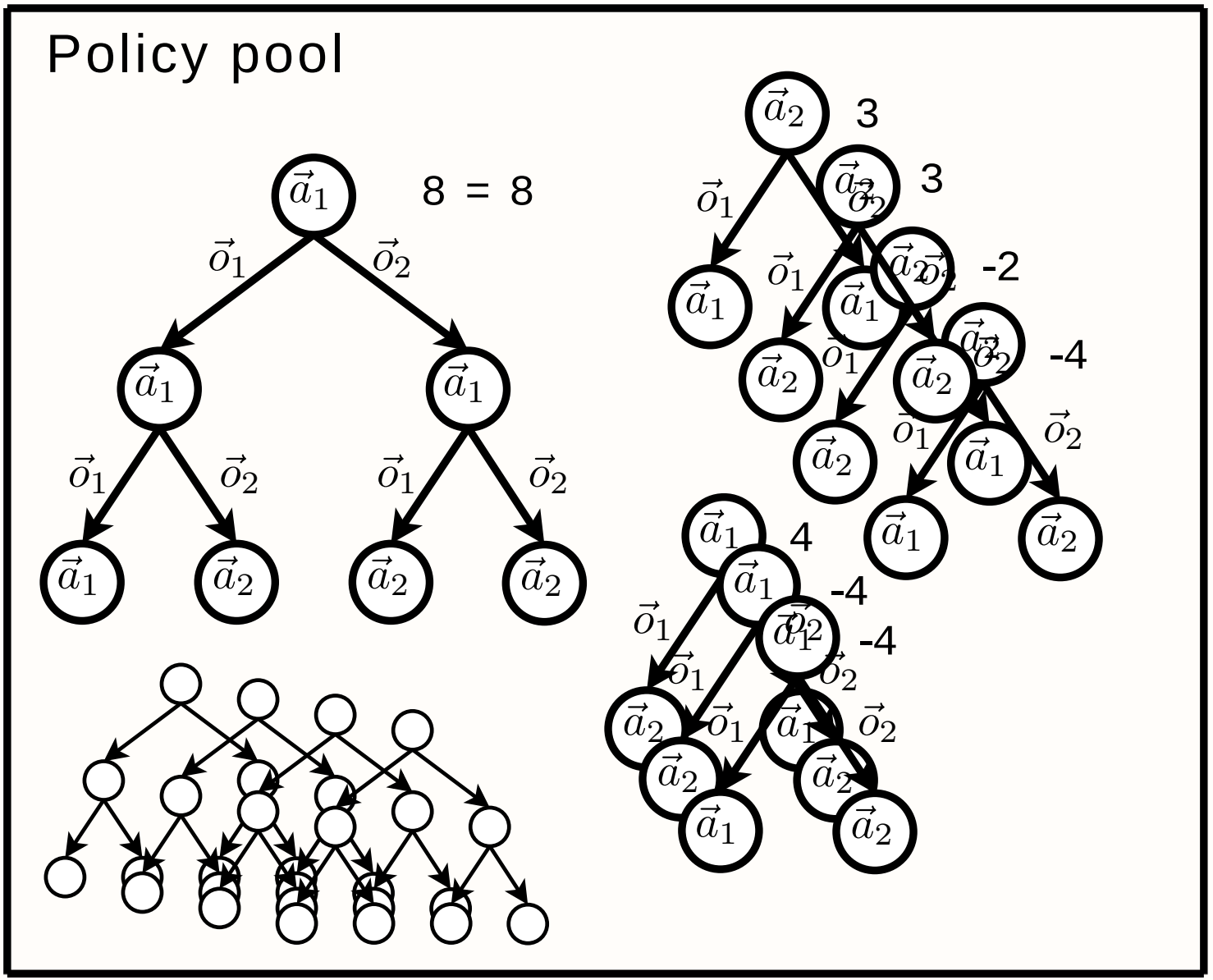
Multiagent A*: example



INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Multiagent A*: example



Heuristic functions

- MAA* and other algorithms need heuristic functions $\widehat{V}^{t\ldots h-1}$.
- Can be defined by solving simplified problem settings.

$$\widehat{V}^t(\varphi^t) = \sum_{\vec{\boldsymbol{\theta}}^t \in \vec{\boldsymbol{\Theta}}^t_{\varphi^t}} P(\vec{\boldsymbol{\theta}}^t | b^0) Q(\vec{\boldsymbol{\theta}}^t, \varphi^t(\vec{\boldsymbol{\theta}}^t)),$$

where $\vec{\theta}_i^t = (a_i^0, o_i^1, \ldots, a_i^{t-1}, o_i^t)$ is an action-observation history.

Heuristic functions

- Q_{MDP} (Littman et al., 1995):

$$\widehat{Q}_{\mathsf{M}}(\vec{\boldsymbol{\theta}}^t, \vec{a}) = \sum_{s \in \mathcal{S}} Q_{\mathsf{M}}^{t,*}(s, \vec{a}) P(s|\vec{\boldsymbol{\theta}}^t),$$

  where $Q_{\mathsf{M}}^{t,*}(s, \vec{a})$ is a non-stationary solution to the underlying MDP.

  - Cheap to compute, and feasible for high horizons.
  - Assumes centralized observations and full observability $\rightarrow$ loose bound.

Heuristic functions

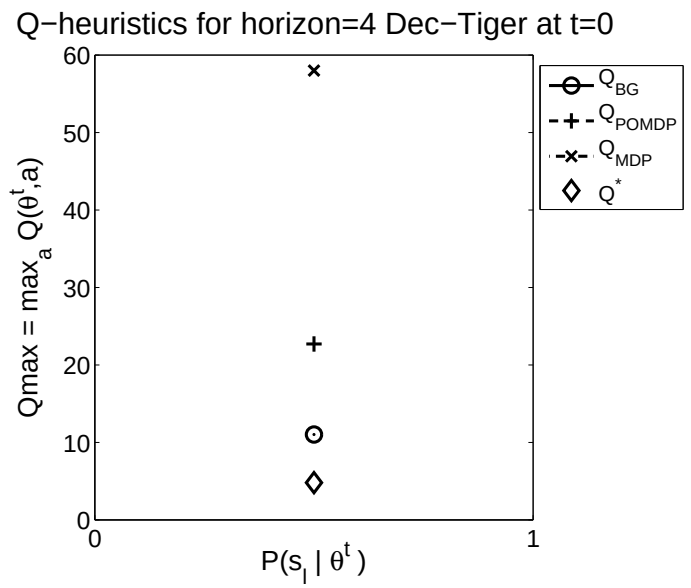
- $\mathrm{Q}_{\mathrm{POMDP}}$ (Szer et al., 2005; Roth et al., 2005):

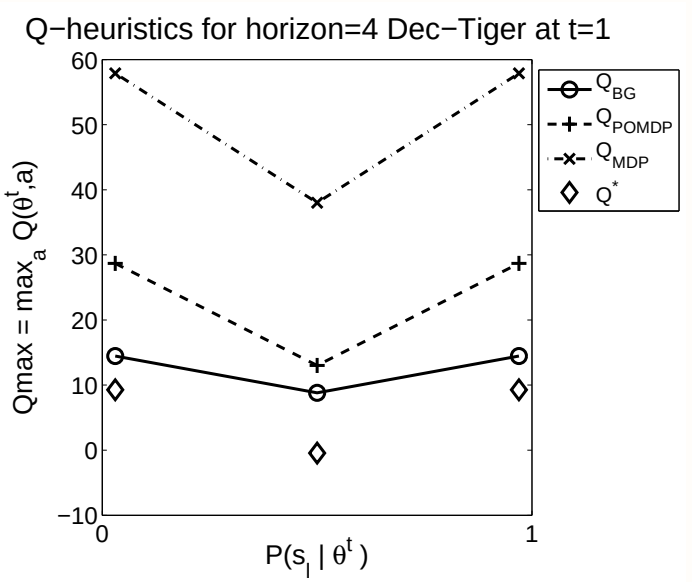
$$\widehat{Q}_{\mathrm{P}}(\vec{\boldsymbol{\theta}}^t, \vec{a}) = Q^*_{\mathrm{P}}(b^{\vec{\boldsymbol{\theta}}^t}, \vec{a}),$$

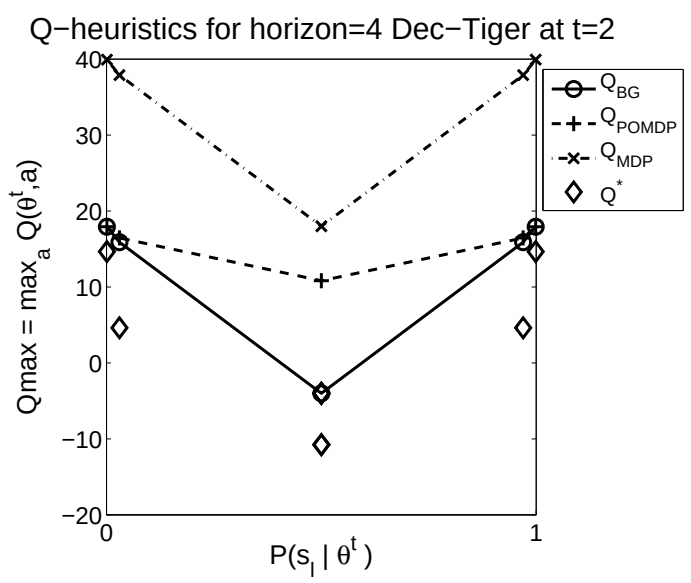
  where $Q^*_{\mathrm{P}}(b^{\vec{\boldsymbol{\theta}}^t}, \vec{a})$ is a solution to the underlying POMDP, i.e., the "belief MDP".
  - ▶ Assumes centralized observations.
  - ▶ Tighter heuristic than $\mathrm{Q}_{\mathrm{MDP}}$, but exponential in h.
- Q_{BG} (Oliehoek and Vlassis, 2007): assumes centralized observations, but delayed by 1 step.
- Hierarchy of upper bounds (Oliehoek et al., 2008b):

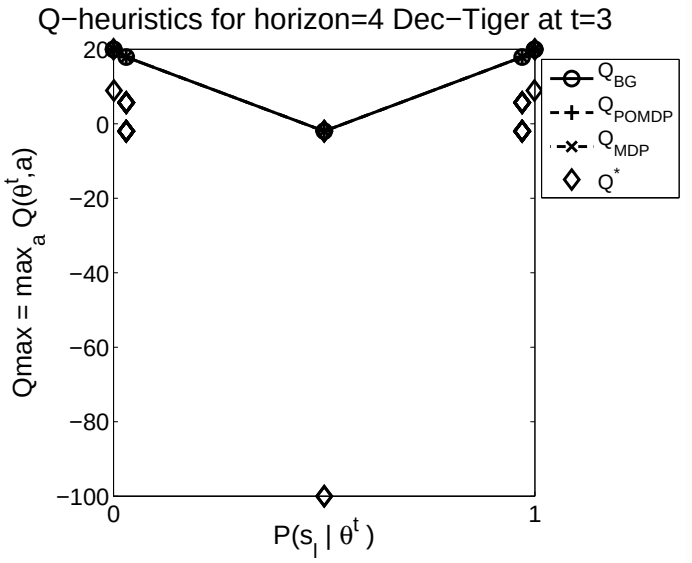
$$Q^* \leq \mathrm{Q}_{\mathrm{BG}} \leq \mathrm{Q}_{\mathrm{POMDP}} \leq \mathrm{Q}_{\mathrm{MDP}}.$$

Heuristic functions: example DEC-Tiger $h = 4$









INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

DEC-POMDPs as series of Bayesian Games

- DEC-POMDPs can be approximated by series of Bayesian Games (Emery-Montemerlo et al., 2004).

Definition 20. *Bayesian Game for a time step* t

- $\boldsymbol{\Theta} = \times_i \Theta_i$ *is the set of joint types. . .*
- *. . . over which a probability function* $P(\boldsymbol{\Theta})$ *is specified*
- $u_i : \boldsymbol{\Theta} \times \mathcal{A} \rightarrow \mathbb{R}$ *— payoff function.*

- Action-observation history $\vec{\theta}_i^t = (a_i^0, o_i^1, \ldots, a_i^{t-1}, o_i^t)$
- BG for time step t of a DEC-POMDP:
  - Types are action-observation histories $\Theta_i \equiv \vec{\Theta}_i^t$
  - Given the past joint policy $\varphi^t = (\delta^0, \ldots, \delta^{t-1})$, probabilities $P(\boldsymbol{\Theta})$ are known.

DEC-POMDPs as series of Bayesian Games

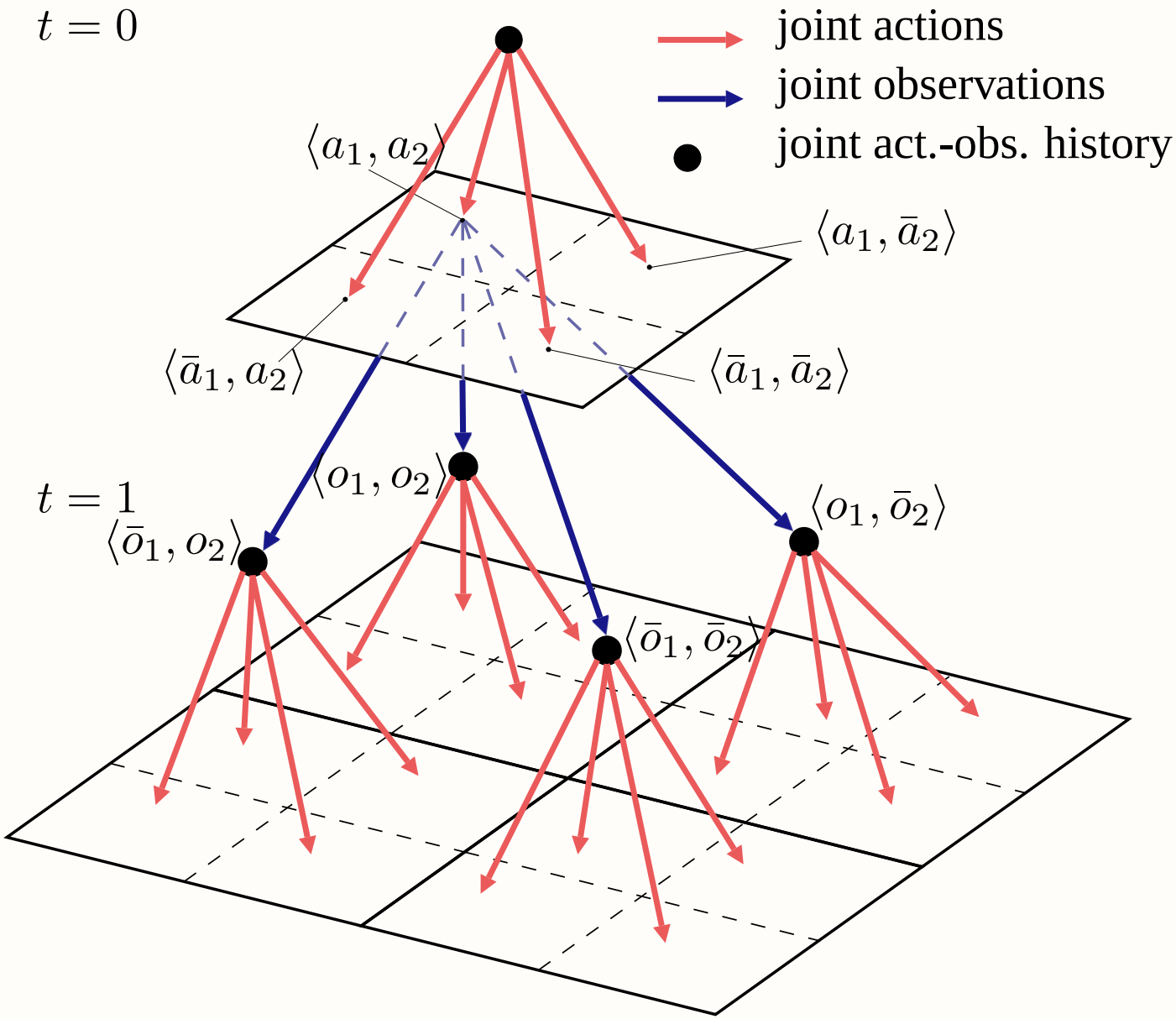
$\vec{\theta}_1^{t=0}$	$\vec{\theta}_2^{t=0}$	$()$	
		a_2	$\bar{a}_2$
$()$	a_1	$+2.75$	-4.1
	$\bar{a}_1$	-0.9	$+0.3$

$\vec{\theta}_1^{t=1}$	$\vec{\theta}_2^{t=1}$	(a_2, o_2)		$(a_2, \bar{o}_2)$		...
		a_2	$\bar{a}_2$	a_2	$\bar{a}_2$	
(a_1, o_1)	a_1	-0.3	$+0.6$	-0.6	$+4.0$	...
	$\bar{a}_1$	-0.6	$+2.0$	-1.3	$+3.6$	...
$(a_1, \bar{o}_1)$	a_1	$+3.1$	$+4.4$	-1.9	$+1.0$	...
	$\bar{a}_1$	$+1.1$	-2.9	$+2.0$	-0.4	
$(\bar{a}_1, o_1)$	a_1	-0.4	-0.9	-0.5	-1.0	...
	$\bar{a}_1$	-0.9	-4.5	-1.0	$+3.5$	...
$(\bar{a}_1, \bar{o}_1)$	...	...	...	...	...	...

INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

DEC-POMDPs as series of Bayesian Games



INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Bayesian Game Approximation

- Idea: instead of doing A* search, only keep the best joint policy φ^t when solving each Bayesian game (Emery-Montemerlo et al., 2004).
- Or: keep the k best (k-GMAA*, forward sweep policy computation Oliehoek et al., 2008b).
- Choice of heuristics: Q_{MDP}, Q_{POMDP}, Q_{BG}.
- Choice of BG solvers:
  - Brute force search (exact)
  - Branch and bound (Oliehoek et al., next Friday, session 23)
  - Alternating Maximization (approximate)

Approximate, General, Finite-horizon

Generalized MAA*

Algorithm 4: GMAA* (Oliehoek et al., 2008b)

Max. lower bound $\underline{v}^\star \leftarrow -\infty$

Policy pool $\mathtt{P} \leftarrow \{\varphi^0 = ()\}$

repeat

  $\varphi^t \leftarrow \mathtt{Select}(\mathtt{P})$

  $\Phi_{\mathtt{Next}} \leftarrow \text{ConstructAndSolveBG}(\varphi^t, b^0)$

  if Φ_{Next} *contains full policies* $\Pi_{Next} \subseteq \Phi_{Next}$ **then**

    $\pi' \leftarrow \arg\max_{\pi \in \Pi_{\mathtt{Next}}} V(\pi)$

    if $V(\pi') > \underline{v}^\star$ **then**

      $\underline{\mathrm{v}}^\star \leftarrow V(\pi')$ `found new lower bound`

      $\pi^\star \leftarrow \pi'$

      $\mathtt{P} \leftarrow \{\varphi \in \mathtt{P} \mid \widehat{V}(\varphi) > \underline{\mathrm{v}}^\star\}$ `prune P`

    $\Phi_{\mathtt{Next}} \leftarrow \Phi_{\mathtt{Next}} \setminus \Pi_{\mathtt{Next}}$ `remove full policies`

  $\mathtt{P} \leftarrow (\mathtt{P} \setminus \varphi^t) \cup \{\varphi \in \Phi_{\mathtt{Next}} \mid \widehat{V}(\varphi) > \underline{\mathrm{v}}^\star\}$

until *P is empty*

Optimal or Approximate, General, Finite-horizon

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Lossless Clustering of Histories

- Idea: if two individual histories induce the same distribution over states and over other agents' histories, they are equivalent and can be clustered (Oliehoek et al., 2009).

$$\forall_{\vec{\boldsymbol{\theta}}_{\neq i}} \quad P(\vec{\boldsymbol{\theta}}_{\neq i}|\vec{\theta}_{i,a}) = P(\vec{\boldsymbol{\theta}}_{\neq i}|\vec{\theta}_{i,b})$$

$$\forall_{\vec{\boldsymbol{\theta}}_{\neq i}} \forall_s \quad P(s|\vec{\boldsymbol{\theta}}_{\neq i}, \vec{\theta}_{i,a}) = P(s|\vec{\boldsymbol{\theta}}_{\neq i}, \vec{\theta}_{i,b})$$

- Lossless clustering, independent of heuristic, but problem dependent.
- Clustering is bootstrapped: algorithms only deal with clustered Bayesian games.
- Large increase in scalability of optimal solvers.

Optimal and Approximate, General, Finite-horizon

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Top down summary

- Top down methods can exploit initial belief state, and various heuristics.
- GMAA* generalizes MAA* and the work of Emery-Montemerlo et al. (2004).
- Without clustering Bayesian games grow exponentially in time horizon.

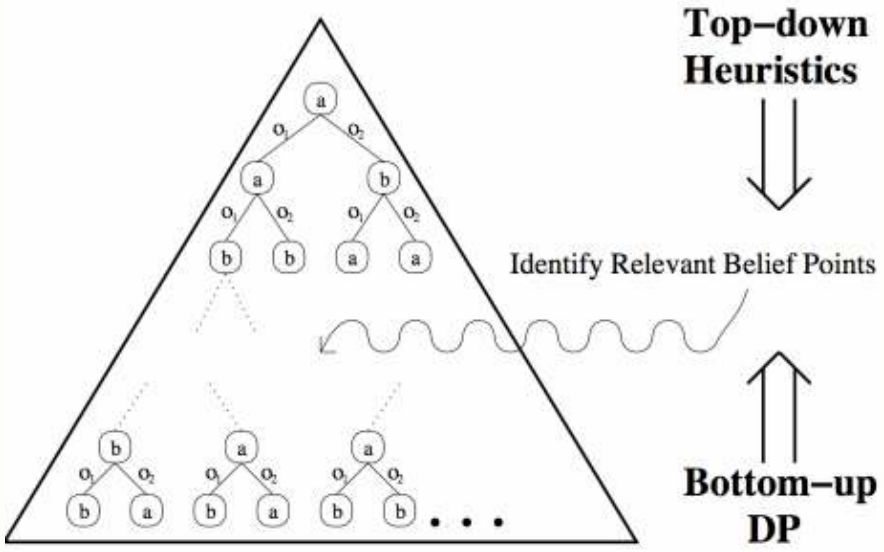
INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Memory Bounded Dynamic Programming (MBDP)

- Do not keep all policies at each step of dynamic programming
- Keep a fixed number for each agent *maxTrees*
- Select these by using heuristic solutions from initial state
- Combines top down and bottom up approaches

(Seuken and Zilberstein, 2007b)



Approximate, General, Finite-horizon

MBDP algorithm

Algorithm 5: MBDP (Seuken and Zilberstein, 2007b)

Start with a one-step policy for each agent

for $t = h$ *to* 1 **do**

  Backup each agent's policy

  for $k = 1$ *to* maxTrees **do**

    Compute heuristic policy and resulting belief state b

    Choose best set of trees starting at b

Select best set of trees for initial state b_0

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

MBDP summary

- Linear complexity in problem horizon
- Exponential in the number of observations
- Performs well in practice (often with very small *maxTrees*)
- Can be difficult to choose correct *maxTrees*

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Extensions of MBDP

- IMBDP: Limit the number of observations used based on probability at each belief (Seuken and Zilberstein, 2007a)
- MBDP-OC: compress observations based on the value produced (Carlin and Zilberstein, 2008)
- PBIP: heuristic search to find best trees rather than exhaustive (Dibangoye et al., 2009)
- PBIP-IPG: extends PBIP by limiting the possible states (Amato et al., 2009)

Other finite-horizon approaches

- Mixed integer linear programming (MILP) (Aras et al., 2007)
  - Represent each agent's policy in sequence form (instead of as a tree).
  - Solve as a combinatorial optimization problem (MILP).

Optimal, General, Finite-horizon

- Direct Cross-Entropy policy search (DICE) (Oliehoek et al., 2008a)
  - Randomized (sampling-based) algorithm using combinatorial optimization.
  - Applies Cross-Entropy method to Dec-POMDPs.
  - Scales well wrt number of agents.

Approximate, General, Finite-horizon

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Summary of finite-horizon algorithms

Optimal finite-horizon algorithms:

- Exhaustive
- Dynamic Programming (Hansen et al., 2004)
- MAA* (Szer et al., 2005)
- MILP (Aras et al., 2007)
- Lossless policy space compression
  (Boularias and Chaib-draa, 2008)
- GMAA*-Cluster (Oliehoek et al., 2009)

Summary of finite-horizon algorithms

Approximate finite-horizon algorithms:

- JESP (Nair et al., 2003)
- kGMAA* (Oliehoek et al., 2008b)
- MBDP (Seuken and Zilberstein, 2007b)
  - IMBDP (Seuken and Zilberstein, 2007a)
  - MBDP-OC (Carlin and Zilberstein, 2008)
  - PBIP (Dibangoye et al., 2009)
  - PBIP-IPG (Amato et al., 2009)
- DICE (Oliehoek et al., 2008a)

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Comparison of optimal finite-horizon algorithms

Comparison of published results:

- MAA*, GMAA*-Cluster (Oliehoek et al., 2009)
- (Boularias and Chaib-draa, 2008)
- DP (taken from Boularias and Chaib-draa, 2008)
- MILP (Aras et al., 2007)

DEC-Tiger (Q_{BG})

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)	$T_{Boularias}$(s)	T_{DP}(s)	T_{MILP}(s)
2	-4.0000	≤ 0.01	≤ 0.01	0.17	0.20	
3	5.1908	0.02	≤ 0.01	1.79	2.29	3.5
4	4.8028	3,069.4	1.50	534.90		72
5	7.0265	–	130.82			

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Comparison of optimal finite-horizon algorithms

BroadcastChannel (Q_{MDP})

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)	$T_{Boularias}$(s)	T_{DP}(s)	T_{MILP}(s)
2	2.0000	≤ 0.01	≤ 0.01	0.14	0.12	
3	2.9900	≤ 0.01	≤ 0.01	0.36	0.46	0.84
4	3.8900	3.22	≤ 0.01	4.59	17.59	10.2
5	4.7900	–	≤ 0.01			25
6	5.6900	–	≤ 0.01			
7	6.5900	–	≤ 0.01			
8	7.4900	–	≤ 0.01			
9	8.3900	–	≤ 0.01			
10	9.2900	–	≤ 0.01			
15	13.7900	–	≤ 0.01			
20	18.3132	–	0.08			
25	22.8815	–	1.67			

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Comparison of optimal finite-horizon algorithms

GridSmall (Q_{BG})

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)
2	0.9100	≤ 0.01	≤ 0.01
3	1.5504	4.21	0.71
4	2.2416	–	30.17

Cooperative Box Pushing ($\mathsf{Q}_{\mathsf{MDP}}$)

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)
2	17.6000	0.05	≤ 0.01
3	66.0810	–	4.55

Hotel 1 (Q_{BG})

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)
2	9.5000	≤ 0.01	0.02
3	15.7047	–	0.07
4	20.1125	–	1.37

Recycling Robots ($\mathsf{Q}_{\mathsf{MDP}}$)

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)
2	6.8000	≤ 0.01	≤ 0.01
3	9.7647	0.02	≤ 0.01
4	11.7264	23052.5	0.02
5	13.7643	–	0.10
10	21.2006	–	4.92
15	25.5940	–	81.46

FireFighting $\langle n_h = 3, n_f = 3 \rangle$ (Q_{BG})

h	V^*	T_{GMAA^*}(s)	T_{clus}(s)
2	-4.3825	0.03	0.03
3	-5.7370	0.91	0.70
4	-6.5789	5605.3	5823.5

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Comparison of approximate finite-horizon algorithms

DEC-Tiger (7 *maxTrees*)

$|S| = 2$, $|A_i| = 3$, $|\Omega_i| = 2$

h	JESP	MBDP
2	-4.00	-4.00
3	-6.00	5.19
10	–	13.49
100	–	93.24
1000	–	819.01
10000	–	7930.68
100000	–	78252.18

Cooperative Box Pushing (3 *maxTrees*,*maxObs*)

$|S| = 100$, $|A_i| = 4$, $|\Omega_i| = 5$

h	MBDP	IMBDP	MBDP-OC
5	–	79.1	72.3
10	–	90.9	103.9
20	–	96.0	149.8
50	–	80.8	278.7
100	–	72.8	503.8

Published results from

- JESP (Nair et al., 2003)
- MBDP (Seuken and Zilberstein, 2007b)
- IMBDP and MBDP-OC (Carlin and Zilberstein, 2008)

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Comparison of approximate finite-horizon algorithms

Cooperative Box Pushing, $|S| = 100$, $|A_i| = 4$, $|\Omega_i| = 5$

h	MBDP	PBIP	PBIP-IPG	Value
10	–	46s	11s	103.22
100	–	536s	181s	598.40
1000	–	5068s	2147s	5707.59
2000	–	10107s	4437s	11392.03

Stochastic Mars Rover, $|S| = 256$, $|A_i| = 6$, $|\Omega_i| = 8$

h	MBDP	PBIP	PBIP-IPG	Value
2	–	106s	19s	5.80
3	–	–	71s	9.38
5	–	–	301s	12.66
10	–	–	976s	21.18
20	–	–	14947s	37.81

Results from (Amato et al., 2009)

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Infinite-horizon only algorithms

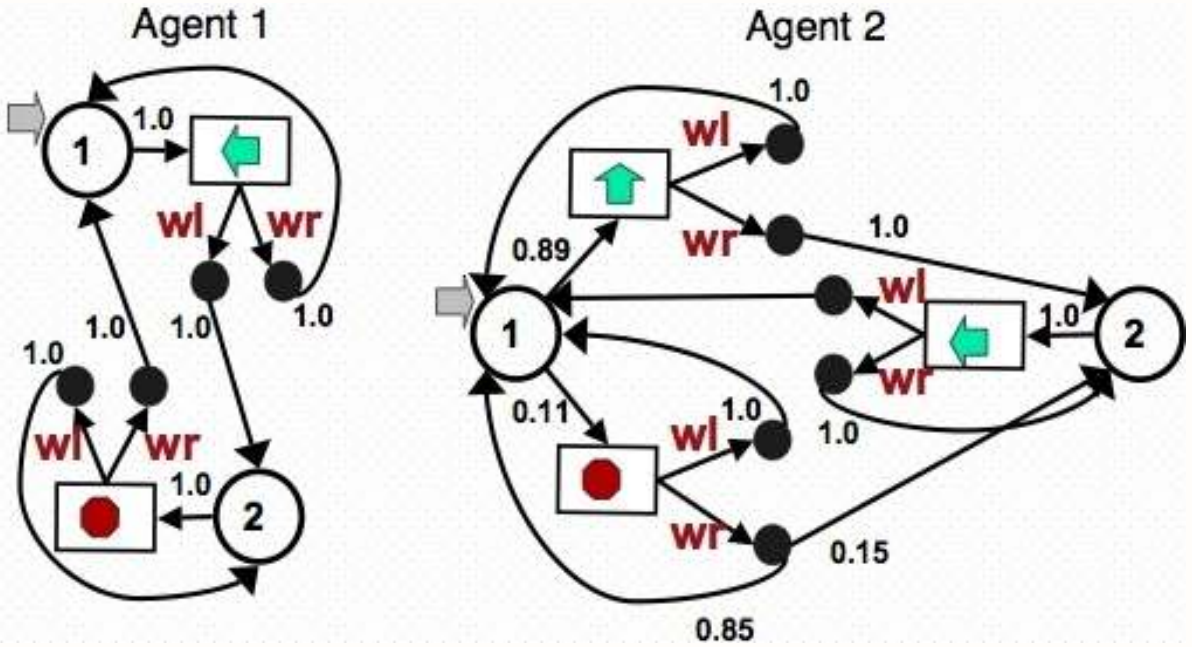
A large enough horizon can be used to approximate an infinite-horizon solution, but this is neither efficient nor compact

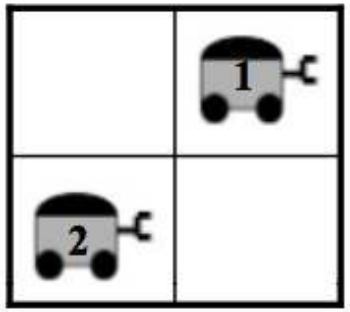
Specialized infinite-horizon solutions have also been developed:

- Policy Iteration (PI)
- Best-First Search (BFS)
- Bounded Policy Iteration for DEC-POMDPS (DEC-BPI)
- Nonlinear Programming (NLP)

Using controllers for infinite-horizon policies

Example: Two agents meeting in a grid





- Periodic policies
- Inherently infinite-horizon
- Randomness can reduce memory limitations

- Nodes define actions
- Transitions based on observations seen

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Policy iteration for DEC-POMDPs

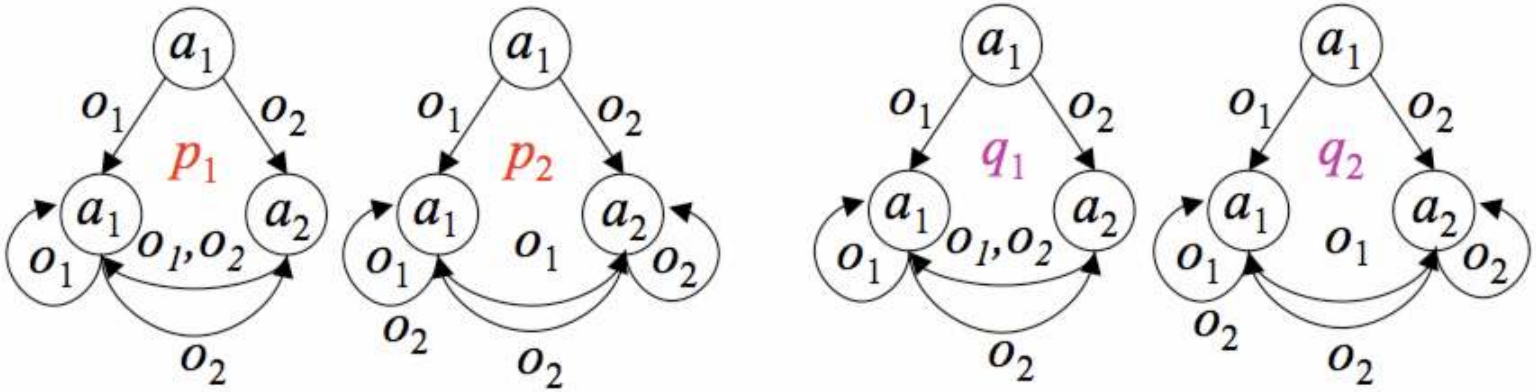
- Dynamic programming with controllers
- Start with a one node controller and build a larger controller at each step
- Stop when discount factor makes further value change less then ϵ
- Pruning merges nodes and creates a stochastic controller as one node may be dominated by a distribution
- Can produce ϵ-optimal controllers for any initial state

(Bernstein et al., 2009)

Optimal, General, Infinite-horizon

Policy iteration for DEC-POMDPs

- Optimal for any initial state
- Intractable for all but the smallest problems
- Does not use known initial state information
  - ► Thus, retains many nodes that are unnecessary for a given start state



Optimal, General, Infinite-horizon

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Approximate infinite-horizon approaches

- Use a fixed-size controller
- Want best value for the given size
- How can the action selection and node transition parameters be set?
  - ▶ Deterministic approaches: using heuristic search methods
  - ▶ Stochastic approaches: using continuous optimization

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Best-first search (BFS)

- Search through space of deterministic action selection and node transition parameters
- Produces optimal fixed-size deterministic controllers
- High search time limits this to very small controllers (< 3 nodes)

(Szer and Charpillet, 2005)

Approximate, General, Infinite-horizon

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Bounded policy iteration (DEC-BPI)

- Improve the controller over a series of steps until value converges
- Alternate between improvement and evaluation
- Improvement

  Use a linear program to determine if a nodes parameters can be changed, while fixing the rest of the controller and other agent policies

  Improved nodes must have better value for all states and nodes of the other agents (multiagent belief space)
- Evaluation: Update the value of all nodes in the agent's controller
- Can solve much larger controller than BFS, but value is low due to lack of start state info and LP

(Bernstein et al., 2005)

Approximate, General, Infinite-horizon

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Nonlinear programming (NLP)

- Optimal fixed-size representation
- Improve and evaluate all in one step by setting value as a parameter
- This requires nonlinear constraints to ensure correct value
- Uses start state info
- Globally optimal solution is intractable, but locally optimal solvers can produce better quality than LP
- Even locally optimal approaches cannot solve large controllers

(Amato et al., 2007)

Approximate, General, Infinite-horizon

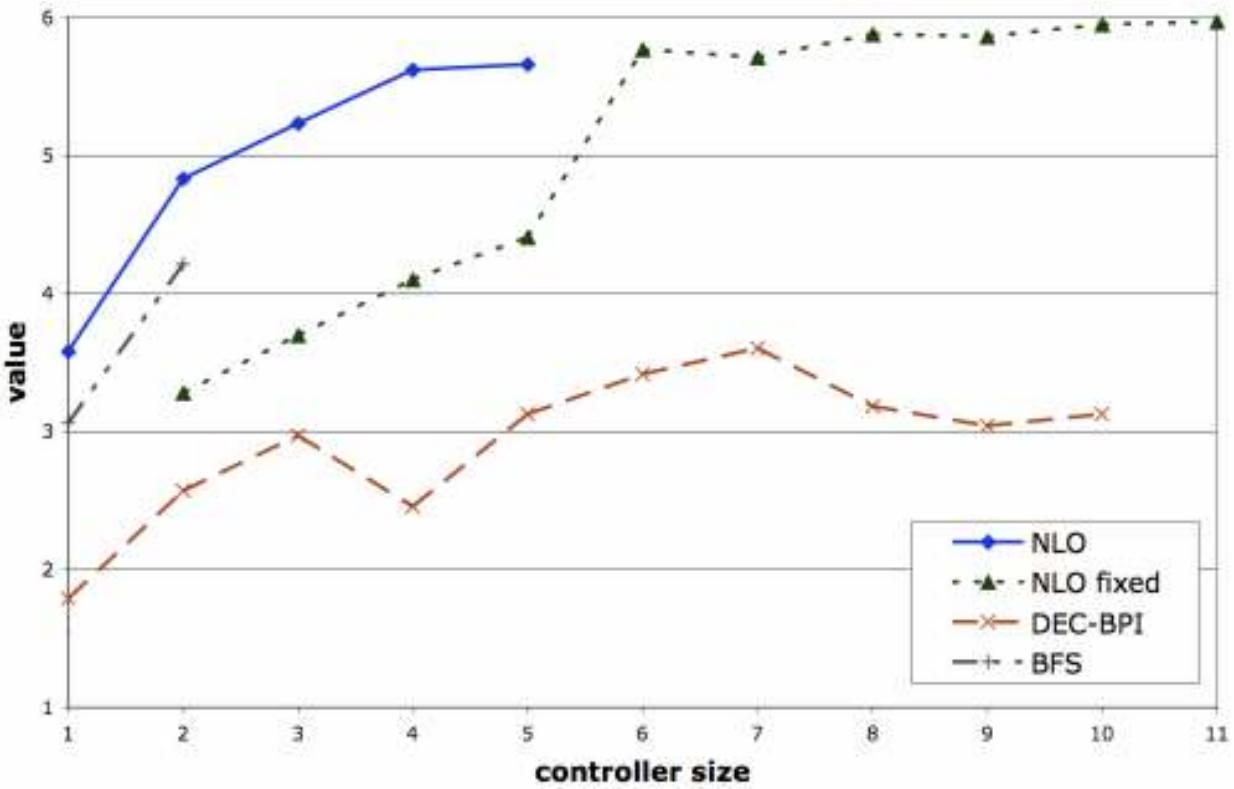
Summary of infinite-horizon algorithms

Optimal algorithm can only solve very small problems

Approximate algorithms can outperform policy iteration because they are more scalable

NLP generally outperforms others, but more scalability is needed

GridSmall: 16 states, 4 actions, 2 obs



Policy Iteration: 3.7 with 80 nodes in 821s before running out of memory

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Summary of infinite-horizon algorithms

Cooperative Box Pushing: 100 states, 4 actions, 5 obs

	Value				Time			
Size	NLP	NLP fix	DEC-BPI	BFS	NLP	NLP fix	DEC-BPI	BFS
1	-1.580	n/a	-10.367	-2	20	n/a	26	1696
2	31.971	-6.252	3.293	–	115	18	579	–
3	46.282	5.097	9.442	–	683	27	4094	–
4	50.640	18.780	7.894	–	5176	44	11324	–
5	–	53.132	14.762	–	–	92	27492	–
6	–	73.247	–	–	–	143	–	–
7	–	80.469	–	–	–	256	–	–

Policy Iteration: 12.84 with 9 nodes in 209s before running out of memory

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Algorithms using communication

- Analysis of possible communication models and complexity results (Pynadath and Tambe, 2002)
- Myopic communication in transition independent Dec-MDPs (Becker et al., 2009)
- Reasoning about run-time communication decisions (Nair et al., 2004; Roth et al., 2005)
- Exploiting factored representations (Roth et al., 2007)
- Stochastically delayed communication (Spaan et al., 2008)

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Algorithms for DEC-POMDP subclasses

- General idea: less powerful models allow for more scalability.
- E.g., independence assumptions:
  - ▶ DEC-MDPs with transition and observation independence.
  - ▶ ND-POMDPs
- Assuming structure in the models
  - ▶ Factored DEC-POMDPs

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Subclasses: DEC-MDPs

Motivation: agents may have limited interactions with each other (e.g. Mars rovers etc.)

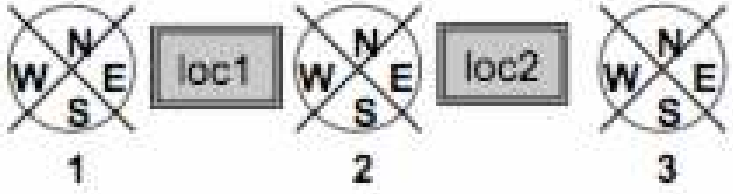
- Independent transitions and observations and special joint reward structure
  - ► Coverage set algorithm: NP-Complete (Becker et al., 2004b)
  - ► Bilinear programming: More efficient anytime approach (Petrik and Zilberstein, 2009)
- Independent observations and event- driven interactions
  - ► Exponential in number of interactions (Becker et al., 2004a)

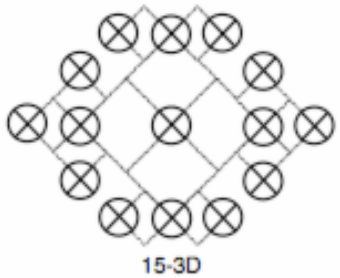
Optimal, Subclass, Finite-horizon

Subclasses: ND-POMDPs

Motivation: to scale up to many agents, exploit locality of interaction (decomposable rewards) (Nair et al., 2005)

- GOA (optimal) and LID-JESP (approximate) (Nair et al., 2005)
- SPIDER (Varakantham et al., 2007)
- FANS (approximate) (Marecki et al., 2008)
- CBDP (approximate) (Kumar and Zilberstein, 2009a)



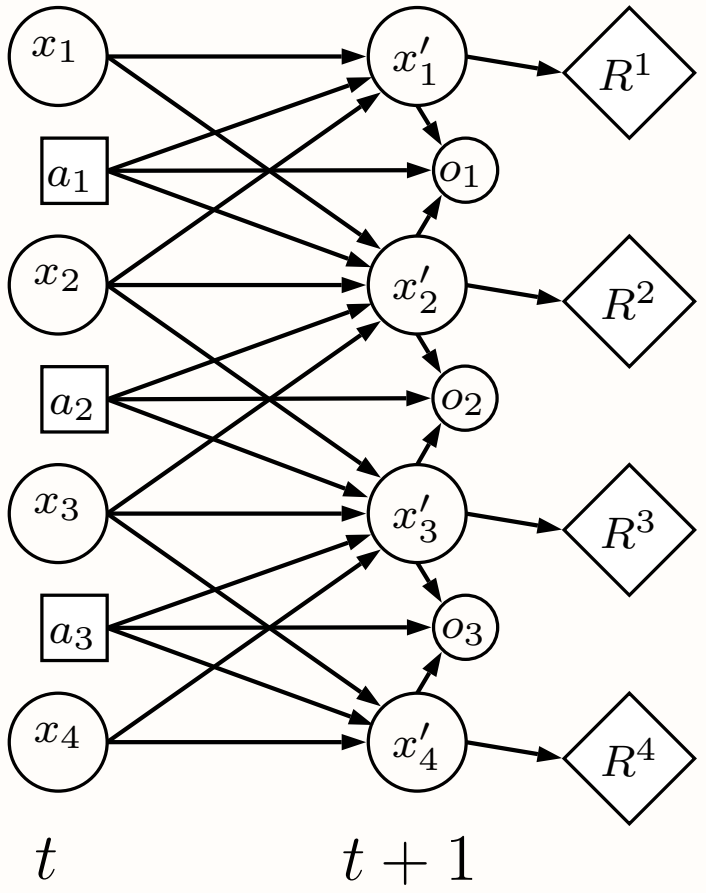


Optimal and Approximate, Subclass, Finite-horizon

Subclasses: Factored DEC-POMDPs

Motivation: exploit locality of interaction, but no strict independence assumptions.

- More general and powerful than ND-POMDPs.
- Less scalable (non-stationary interaction graph).
- GMAA* has been extended to Factored DEC-POMDPs (Oliehoek et al., 2008c).



Optimal and Approximate, Subclass, Finite-horizon

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Problem domains and software tools

- An overview of the existing benchmark problems.
- Description of available software.

Benchmark problems

Some benchmark problems:

- DEC-Tiger (Nair et al., 2003)
- BroadcastChannel (Hansen et al., 2004)
- Meeting on a grid (Bernstein et al., 2005)
- Cooperative Box Pushing (Seuken and Zilberstein, 2007a)
- Recycling Robots (Amato et al., 2007)
- FireFighting (Oliehoek et al., 2008b)
- Sensor network problems (Nair et al., 2005; Kumar and Zilberstein, 2009a,b)

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

Software

- The MADP toolbox aims to provide a software platform for research in decision-theoretic multiagent planning (Spaan and Oliehoek, 2008).
- Main features:
  - A uniform representation for several popular multiagent models.
  - A parser for a file format for discrete Dec-POMDPs.
  - Shared functionality for planning algorithms.
  - Implementation of several Dec-POMDP planners.
- Released as free software, with special attention to the extensibility of the toolbox.
- Provides benchmark problems.

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Problem specification

```
agents: 2
discount: 1
values: reward
states: tiger-left tiger-right
start:
uniform
actions:
listen open-left open-right
listen open-left open-right
observations:
hear-left hear-right
hear-left hear-right
```

Problem specification (1)

```
# Transitions
T: * :
uniform
T: listen listen :
identity
# Observations
O: * :
uniform
O: listen listen : tiger-left : hear-left hear-left : 0.7225
O: listen listen : tiger-left : hear-left hear-right : 0.1275
[...]
O: listen listen : tiger-right : hear-left hear-left : 0.0225
# Rewards
R: listen listen: * : * : * : -2
R: open-left open-left : tiger-left : * : * : -50
[...]
R: open-left listen: tiger-right : * : * : 9
```

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Example program

```
#include "ProblemDecTiger.h"
#include "JESPExhaustivePlanner.h"
int main()
{
    ProblemDecTiger dectiger;
    JESPExhaustivePlanner jesp(3,&dectiger);
    jesp.Plan();
    std::cout << jesp.GetExpectedReward() << std::endl;
    std::cout << jesp.GetJointPolicy()->SoftPrint() << std::endl;
    return(0);
}
```

INSTITUTO SUPERIOR TÉCNICO

University of Massachusetts Amherst

Program output

```
src/examples>  ./decTigerJESP
Value computed for DecTiger horizon 3: 5.19081
Policy computed:
JointPolicyPureVector index 120340 depth 999999
Policy for agent 0 (index 55):
Oempty,  --> a00:Listen
Oempty, o00:HearLeft,  --> a00:Listen
Oempty, o01:HearRight,  --> a00:Listen
Oempty, o00:HearLeft, o00:HearLeft,  --> a02:OpenRight
Oempty, o00:HearLeft, o01:HearRight,  --> a00:Listen
Oempty, o01:HearRight, o00:HearLeft,  --> a00:Listen
Oempty, o01:HearRight, o01:HearRight,  --> a01:OpenLeft
Policy for agent 1 (index 55):
Oempty,  --> a10:Listen
Oempty, o10:HearLeft,  --> a10:Listen
Oempty, o11:HearRight,  --> a10:Listen
Oempty, o10:HearLeft, o10:HearLeft,  --> a12:OpenRight
Oempty, o10:HearLeft, o11:HearRight,  --> a10:Listen
Oempty, o11:HearRight, o10:HearLeft,  --> a10:Listen
Oempty, o11:HearRight, o11:HearRight,  --> a11:OpenLeft
```

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Back to some fundamental questions

- Is decentralized decision making under uncertainty significantly harder than solving POMDPs? Why?
- What features of the problem domain affect the complexity and how?
- Is optimal dynamic programming possible?
- Can dynamic programming be made practical? How?
- Is it beneficial to treat communication as a separate type of action?
- How can we exploit the locality of agent interaction to develop more scalable algorithms?

INSTITUTO
SUPERIOR
TÉCNICO

University *of*
Massachusetts
Amherst

Acknowledgments

This work was partially supported by Fundação para a Ciência e a Tecnologia (ISR/IST pluriannual funding) through the through the PIDDAC Program funds and was supported by project PTDC/EEA-ACR/73266/2006. This work was also supported by the Air Force Office of Scientific Research under Grant No. FA9550-08-1-0181 and by the National Science Foundation under Grant No. IIS-0812149.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

C. Amato, D. S. Bernstein, and S. Zilberstein. Solving POMDPs Using Quadratically Constrained Linear Programs. In *Proc. of Uncertainty in Artificial Intelligence*, 2007.

C. Amato, D. S. Bernstein, and S. Zilberstein. Optimizing memory-bounded controllers for decentralized POMDPs. In *Proc. Int. Joint Conf. on Artificial Intelligence*, 2007.

C. Amato, J. Dibangoye, and S. Zilberstein. Incremental policy generation for finite-horizon DEC-POMDPs. In *Int. Conf. on Automated Planning and Scheduling*, 2009.

R. Aras, A. Dutech, and F. Charpillet. Mixed integer linear programming for exact finite-horizon planning in decentralized POMDPs. In *Int. Conf. on Automated Planning and Scheduling*, 2007.

A. G. Barto, S. J. Bradtke, and S. P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1-2):81–138, 1995.

R. Becker, S. Zilberstein, and V. Lesser. Decentralized Markov decision processes with event-driven interactions. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2004a.

R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman. Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research*, 22: 423–455, 2004b.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

R. Becker, A. Carlin, V. Lesser, and S. Zilberstein. Analyzing myopic approaches for multi-agent communications. *Computational Intelligence*, 25(1):31–50, 2009.

R. Bellman. *Dynamic programming*. Princeton University Press, 1957.

D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

D. S. Bernstein, E. A. Hansen, and S. Zilberstein. Bounded policy iteration for decentralized POMDPs. In *Proc. Int. Joint Conf. on Artificial Intelligence*, 2005.

D. S. Bernstein, C. Amato, E. A. Hansen, and S. Zilberstein. Policy iteration for decentralized control of Markov decision processes. *Journal of Artificial Intelligence Research*, 34(89–132), 2009.

B. Bonet. An epsilon-optimal grid-based algorithm for partially observable Markov decision processes. In *International Conference on Machine Learning*, 2002.

A. Boularias and B. Chaib-draa. Exact dynamic programming for decentralized POMDPs with lossless policy compression. In *Int. Conf. on Automated Planning and Scheduling*, 2008.

C. Boutilier. Planning, learning and coordination in multiagent decision processes. In *Theoretical Aspects of Rationality and Knowledge*, 1996.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

R. I. Brafman. A heuristic variable grid solution method for POMDPs. In *Proc. of the National Conference on Artificial Intelligence*, 1997.

A. Carlin and S. Zilberstein. Value-based observation compression for DEC-POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2008.

A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proc. of the National Conference on Artificial Intelligence*, 1994.

A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien. Acting under uncertainty: Discrete Bayesian models for mobile robot navigation. In *Proc. of International Conference on Intelligent Robots and Systems*, 1996.

R. Cogill, M. Rotkowitz, B. V. Roy, and S. Lall. An approximate dynamic programming approach to decentralized control of stochastic systems. In *Proceedings of the 2004 Allerton Conference on Communication, Control, and Computing*, 2004.

T. Dean, L. Kaelbling, J. Kirman, and A. Nicholson. Planning under time constraints in stochastic domains. *Artificial Intelligence*, 76(1–2):35–74, 1995.

J. S. Dibangoye, A.-I. Mouaddib, and B. Chaib-draa. Point-based incremental pruning heuristic for solving finite-horizon DEC-POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2009.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Approximate solutions for partially observable stochastic games with common payoffs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2004.

P. J. Gmytrasiewicz and P. Doshi. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24:49–79, 2005.

R. Emery-Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Game theoretic control for robot teams. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005.

C. V. Goldman and S. Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *Journal of Artificial Intelligence Research*, 22:143–174, 2004.

C. Guestrin, D. Koller, and R. Parr. Multiagent planning with factored MDPs. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

E. A. Hansen. *Finite-memory control of partially observable systems*. PhD thesis, University of Massachusetts, Amherst, 1998a.

E. A. Hansen. Solving POMDPs by searching in policy space. In *Proc. of Uncertainty in Artificial Intelligence*, 1998b.

E. A. Hansen, D. Bernstein, and S. Zilberstein. Dynamic programming for partially observable stochastic games. In *Proc. of the National Conference on Artificial Intelligence*, 2004.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press and John Wiley & Sons, Inc., 1960.

A. Kumar and S. Zilberstein. Constraint-based dynamic programming for decentralized pomdps with structured interactions. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2009a.

A. Kumar and S. Zilberstein. Event-detecting multi-agent mdps: Complexity and constant-factor approximation. In *Proc. Int. Joint Conf. on Artificial Intelligence*, 2009b.

M. L. Littman, A. R. Cassandra, and L. P. Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning*, 1995.

W. S. Lovejoy. Computationally feasible bounds for partially observed Markov decision processes. *Operations Research*, 39(1):162–175, 1991.

O. Madani, S. Hanks, and A. Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *Proc. of the National Conference on Artificial Intelligence*, Orlando, Florida, July 1999.

J. Marecki, T. Gupta, P. Varakantham, M. Tambe, and M. Yokoo. Not all agents are equal: Scaling up distributed POMDPs for agent networks. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2008.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

J. Marschak. Elements for a theory of teams. *Management Science*, 1(2):127–137, 1955.

G. E. Monahan. A survey of partially observable Markov decision processes: theory, models and algorithms. *Management Science*, 28(1), Jan. 1982.

R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *Proc. Int. Joint Conf. on Artificial Intelligence*, 2003.

R. Nair, M. Tambe, M. Roth, and M. Yokoo. Communication for improving policy computation in distributed POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2004.

R. Nair, P. Varakantham, M. Tambe, and M. Yokoo. Networked distributed POMDPs: A synthesis of distributed constraint optimization and POMDPs. In *Proc. of the National Conference on Artificial Intelligence*, 2005.

F. A. Oliehoek and N. Vlassis. Q-value functions for decentralized POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, May 2007.

F. A. Oliehoek, J. F. Kooi, and N. Vlassis. The cross-entropy method for policy search in decentralized POMDPs. *Informatica*, 32:341–357, 2008a.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

F. A. Oliehoek, M. T. J. Spaan, and N. Vlassis. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research*, 32:289–353, 2008b.

F. A. Oliehoek, M. T. J. Spaan, S. Whiteson, and N. Vlassis. Exploiting locality of interaction in factored Dec-POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 517–524, 2008c.

F. A. Oliehoek, S. Whiteson, and M. T. J. Spaan. Lossless clustering of histories in decentralized POMDPs. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, May 2009.

J. M. Ooi and G. W. Wornell. Decentralized control of a multiple access broadcast channel: Performance bounds. In *Proc. of the 35th Conference on Decision and Control*, 1996.

C. H. Papadimitriou and J. N. Tsitsiklis. On the complexity of designing distributed protocols. *Information and Control*, 53(3):211–218, 1982.

C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

L. Peshkin and V. Savova. Reinforcement learning for adaptive routing. In *Proc. of the Int. Joint Conf. on Neural Networks*, 2002.

L. Peshkin, K.-E. Kim, N. Meuleau, and L. P. Kaelbling. Learning to cooperate via policy search. In *Proc. of Uncertainty in Artificial Intelligence*, 2000.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

M. Petrik and S. Zilberstein. A bilinear programming approach for multiagent planning. *Journal of Artificial Intelligence Research*, 35:235–274, 2009.

J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *Proc. Int. Joint Conf. on Artificial Intelligence*, 2003.

P. Poupart and C. Boutilier. Bounded finite state controllers. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

D. V. Pynadath and M. Tambe. The communicative multiagent team decision problem: Analyzing teamwork theories and models. *Journal of Artificial Intelligence Research*, 16:389–423, 2002.

M. Roth, R. Simmons, and M. Veloso. Reasoning about joint beliefs for execution-time communication decisions. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2005.

M. Roth, R. Simmons, and M. Veloso. Exploiting factored representations for decentralized execution in multi-agent teams. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2007.

S. J. Russell and P. Norvig. *Artificial Intelligence: a modern approach*. Prentice Hall, 2nd edition, 2003.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

J. Sandell, N., P. Varaiya, M. Athans, and M. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Transactions on Automatic Control*, 23(2):108–128, Apr 1978.

J. K. Satia and R. E. Lave. Markovian decision processes with probabilistic observation of states. *Management Science*, 20(1):1–13, 1973.

S. Seuken and S. Zilberstein. Improved memory-bounded dynamic programming for decentralized POMDPs. In *Proc. of Uncertainty in Artificial Intelligence*, July 2007a.

S. Seuken and S. Zilberstein. Memory-bounded dynamic programming for DEC-POMDPs. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 2009–2015, 2007b.

E. J. Sondik. *The optimal control of partially observable Markov processes*. PhD thesis, Stanford University, 1971.

M. T. J. Spaan and F. S. Melo. Interaction-driven Markov games for decentralized multiagent planning under uncertainty. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, pages 525–532, 2008.

M. T. J. Spaan and F. A. Oliehoek. The MultiAgent Decision Process toolbox: software for decision-theoretic planning in multiagent systems. In *Multi-agent Sequential Decision Making in Uncertain Domains*, 2008. Workshop at AAMAS08.

M. T. J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

M. T. J. Spaan, F. A. Oliehoek, and N. Vlassis. Multiagent planning under uncertainty with stochastic communication delays. In *Int. Conf. on Automated Planning and Scheduling*, pages 338–345, 2008.

D. Szer and F. Charpillet. An optimal best-first search algorithm for solving infinite horizon DEC-POMDPs. In *European Conference on Machine Learning*, 2005.

D. Szer, F. Charpillet, and S. Zilberstein. MAA*: A heuristic search algorithm for solving decentralized POMDPs. In *Proc. of Uncertainty in Artificial Intelligence*, 2005.

J. Tsitsiklis. Decentralized detection by a large number of sensors. *Mathematics of Control, Signals and Systems*, 1(2):167–182, 1988.

J. Tsitsiklis and M. Athans. On the complexity of decentralized decision making and detection problems. *IEEE Transactions on Automatic Control*, 30(5):440–446, 1985.

P. Varaiya and J. Walrand. On delayed sharing patterns. *IEEE Transactions on Automatic Control*, 23(3):443–445, 1978.

P. Varakantham, J. Marecki, Y. Yabu, M. Tambe, and M. Yokoo. Letting loose a SPIDER on a network of POMDPs: Generating quality guaranteed policies. In *Proc. of Int. Joint Conference on Autonomous Agents and Multi Agent Systems*, 2007.

H. Witsenhausen. Separation of estimation and control for discrete time systems. *Proceedings of the IEEE*, 59(11):1557–1566, Nov. 1971.

INSTITUTO SUPERIOR TÉCNICO

University *of* Massachusetts Amherst

References

P. Xuan, V. Lesser, and S. Zilberstein. Communication decisions in multi-agent cooperation: Model and experiments. In *Proc. of the Fifth Int. Conference on Autonomous Agents*, 2001.

N. L. Zhang and W. Liu. Planning in stochastic domains: problem characteristics and approximations. Technical Report HKUST-CS96-31, Department of Computer Science, The Hong Kong University of Science and Technology, 1996.

R. Zhou and E. A. Hansen. An improved grid-based approximation algorithm for POMDPs. In *Proc. Int. Joint Conf. on Artificial Intelligence*, 2001.

S. Zilberstein, R. Washington, D. Bernstein, and A. Mouaddib. Decision-theoretic control of planetary rovers. In *Plan-Based control of Robotic Agents*, volume 2466 of *LNAI*, pages 270–289. Springer, 2002.