Hierarchical POMDP Planning and Execution

Joelle Pineau

Machine Learning Lunch

November 20, 2000

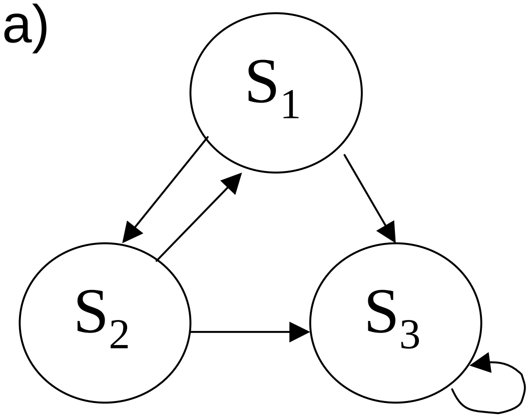
Carnegie Mellon

Partially Observable MDP

- POMDPs are characterized by:
  - States: $s \in S$
  - Actions: $a \in A$
  - Observations: $o \in O$

  - Transition probabilities: $T(s,a,s')=Pr(s'|s,a)$
  - Observation probabilities: $T(o,a,s')=Pr(o|s,a)$
  - Rewards: $R(s,a)$

  - Beliefs: $b(s_t)=Pr(s_t|o_t,a_t,\ldots,o_0,a_0)$



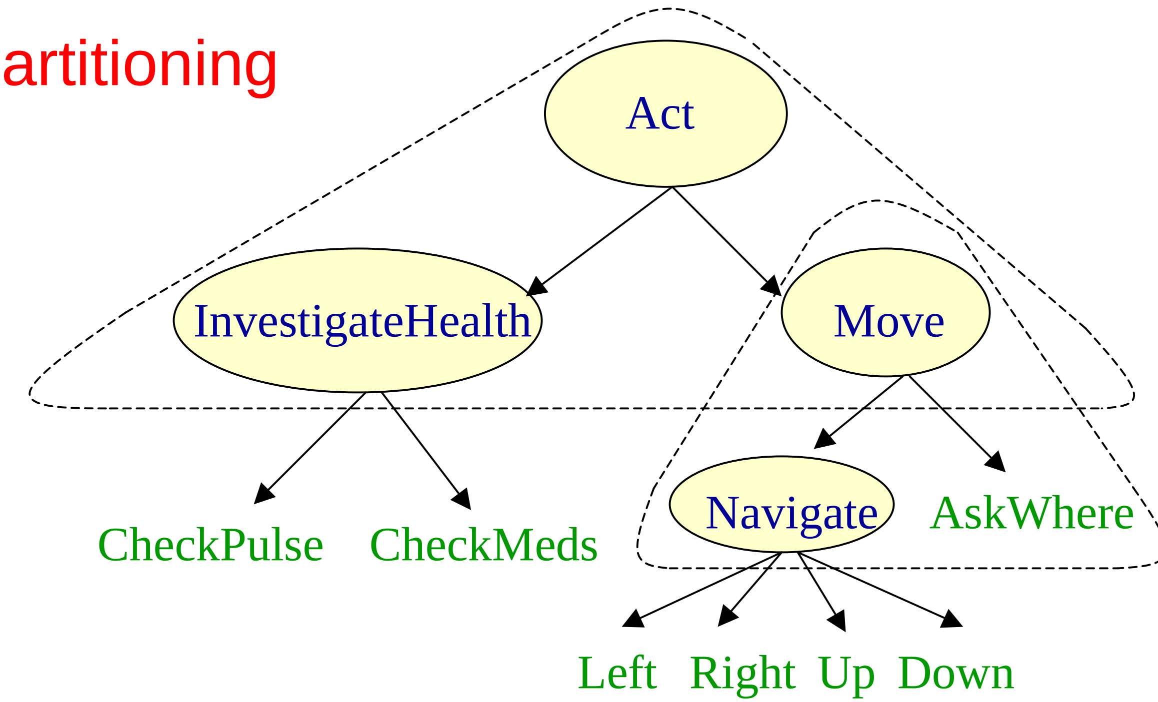
The problem

- How can we find good policies for complex POMDPs?

- Is there a principled way to provide near-optimal policies?

Proposed Approach

- Exploit structure in the problem domain.
- What type of structure?
  - Action set partitioning

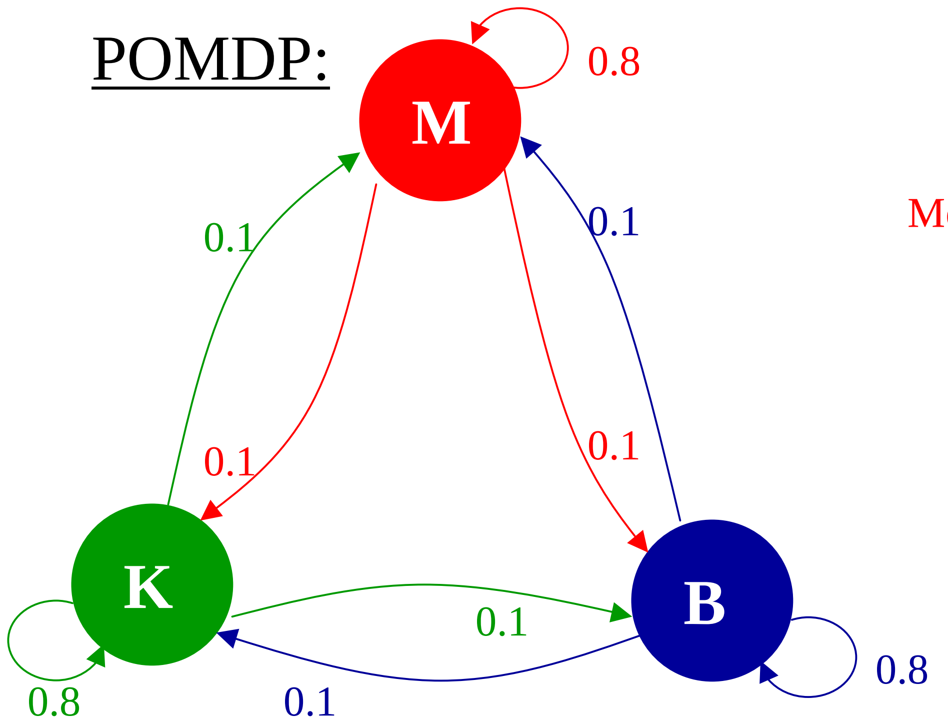


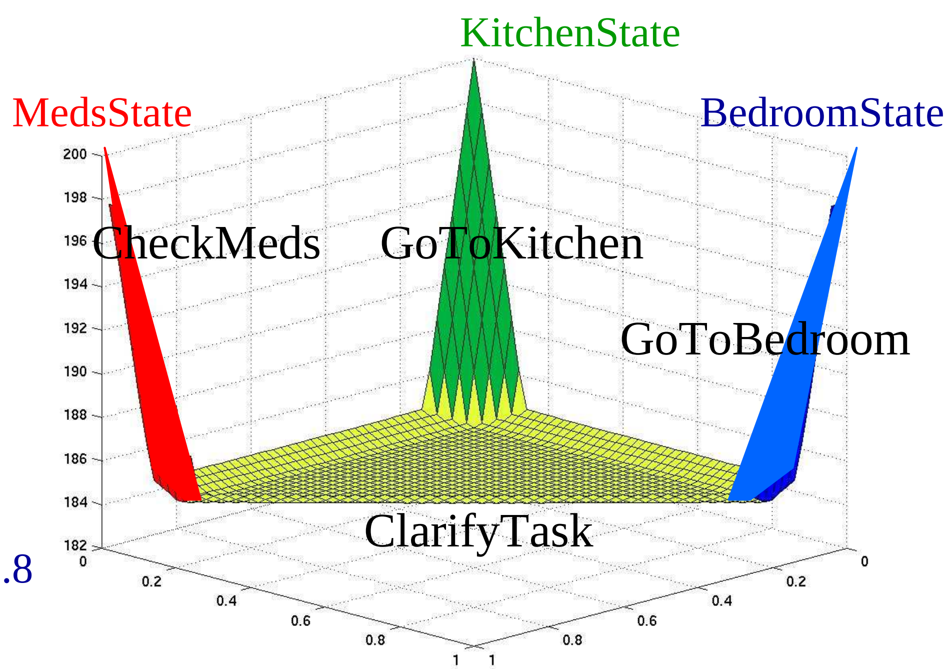
Hierarchical POMDP Planning

- What do we start with?
  - A full POMDP model: $\{S_o, A_o, O_o, M_o\}$.
  - An action set partitioning graph.
- Key idea:
  - Break the problem into many “related” POMDPs.
  - Each smaller POMDP has only a subset of A_o.

    $\Rightarrow$ *imposing policy constraint*
- But why?
  - POMDP: exponential run-time per value iteration $O(|A|\Gamma_{n-1}^{|O|})$

Example





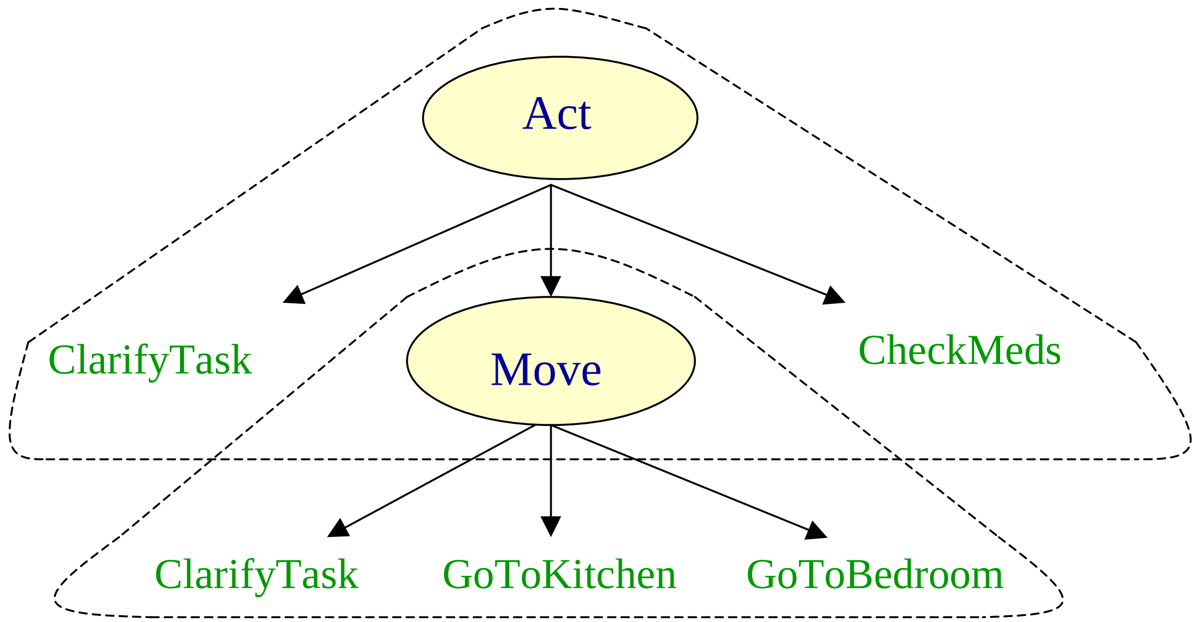
S_o = {Meds, Kitchen, Bedroom}

A_o = {ClarifyTask, CheckMeds, GoToKitchen, GoToBedroom}

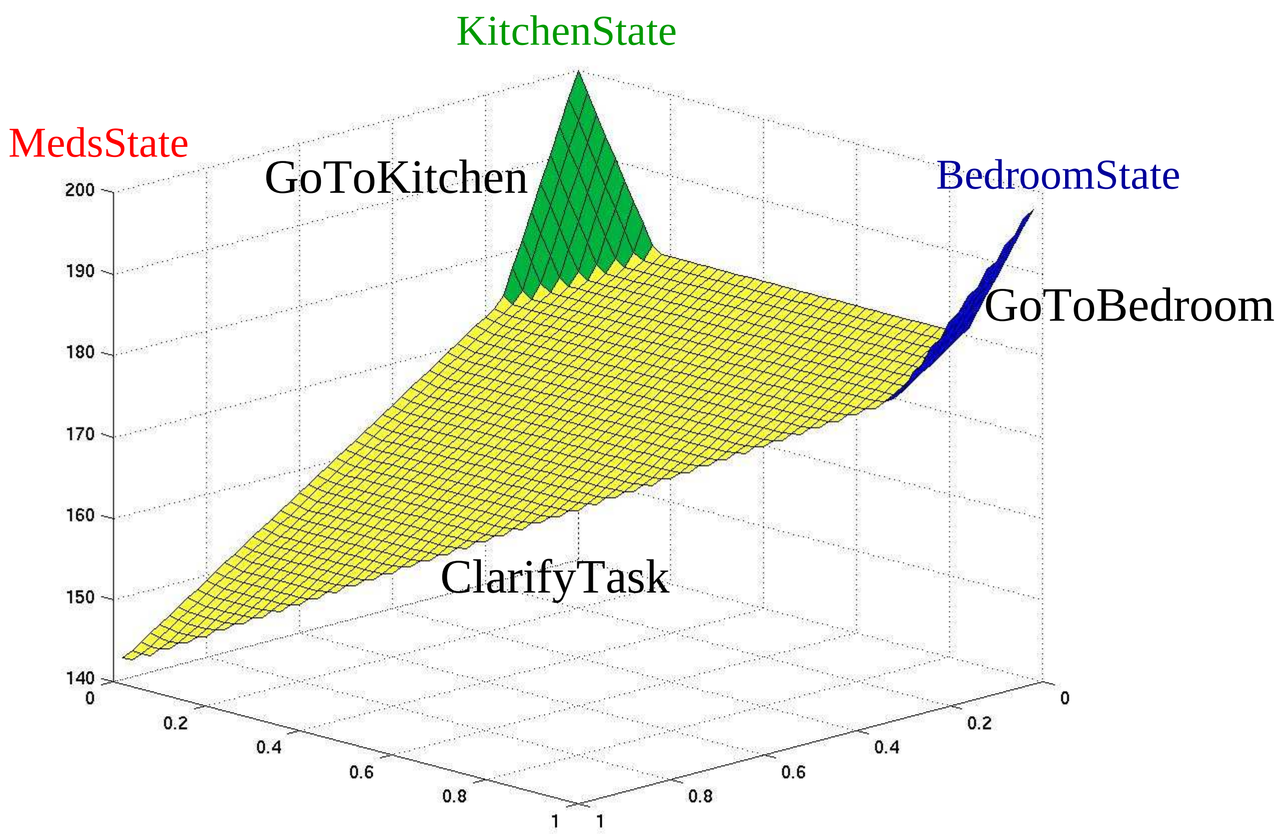
O_o = {Noise, Meds, Kitchen, Bedroom}

Hierarchical POMDP

Action Partitioning:



Local Value Function and Policy - *Move* Controller



Modeling Abstract Actions

Problem: *Need parameters for abstract action Move*

Solution: *Use the local policy of corresponding low-level controller*

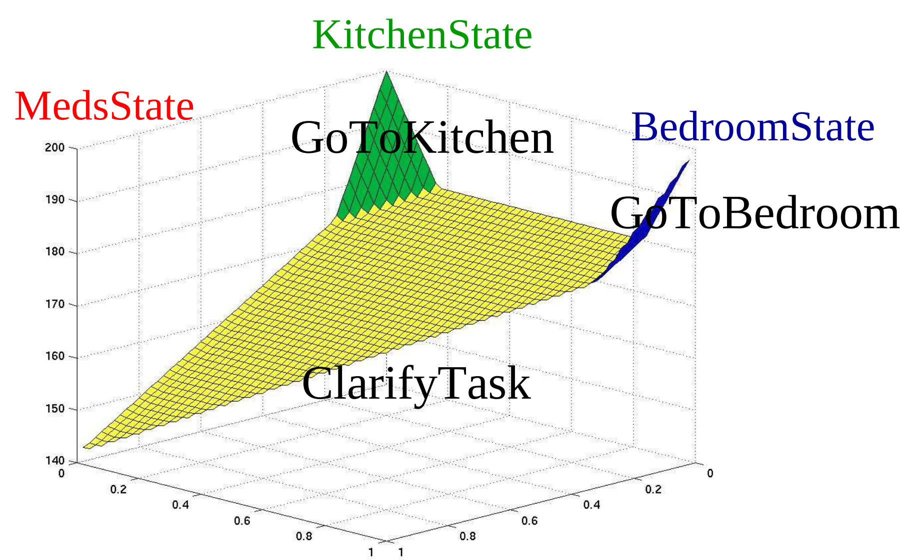
General form: $\Pr(s_j \mid s_i, a_k^{abstract}) = \Pr(s_j \mid s_i, \mathrm{Policy}(a_k^{abstract}, s_i))$

Example:

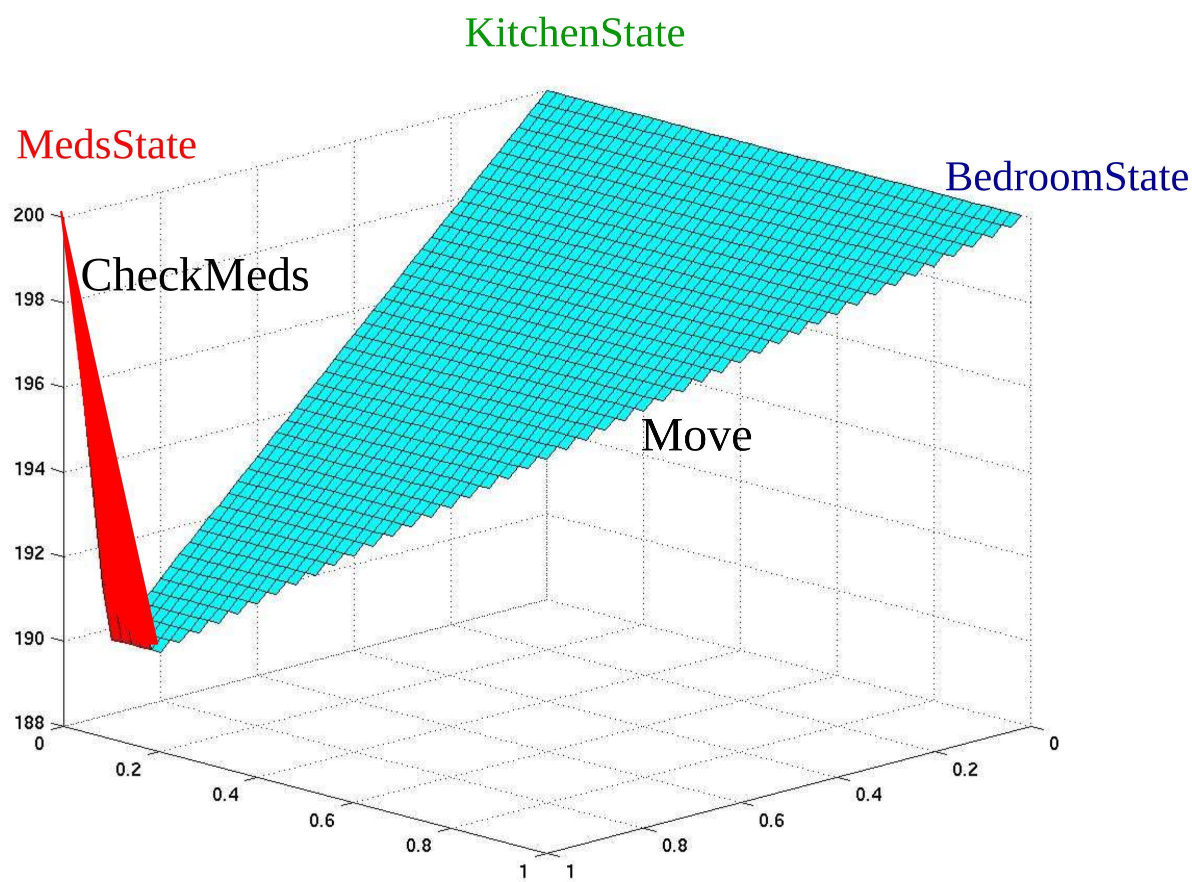
$\Pr(s_j \mid \text{MedsState}, \text{Move})$

$= \Pr(s_j \mid \text{MedsState}, \text{ClarifyTask})$

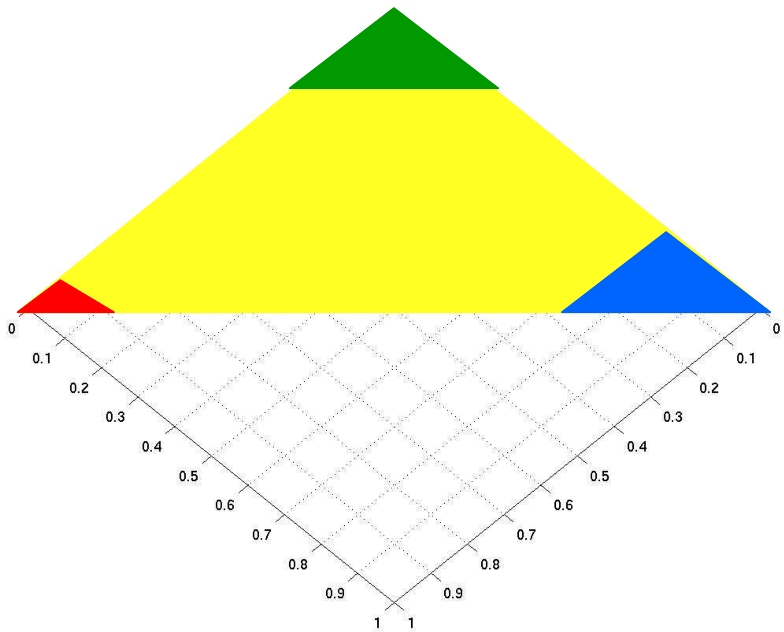
Policy (Move,s_i):



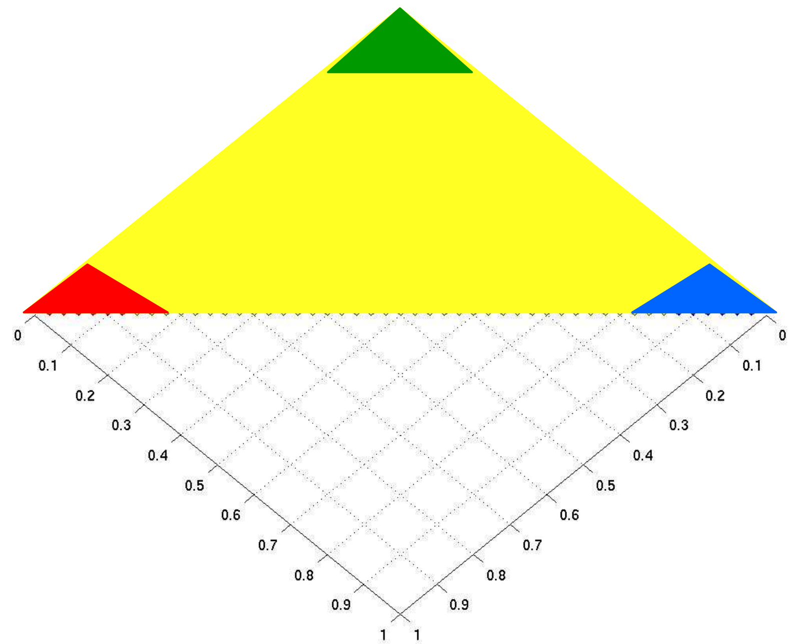
Local Value Function and Policy - *Act* Controller



Comparing Policies



Hierarchical Policy:



Optimal Policy:

= ClarifyTask   = CheckMeds   = GoToKitchen   = GoToBedroom

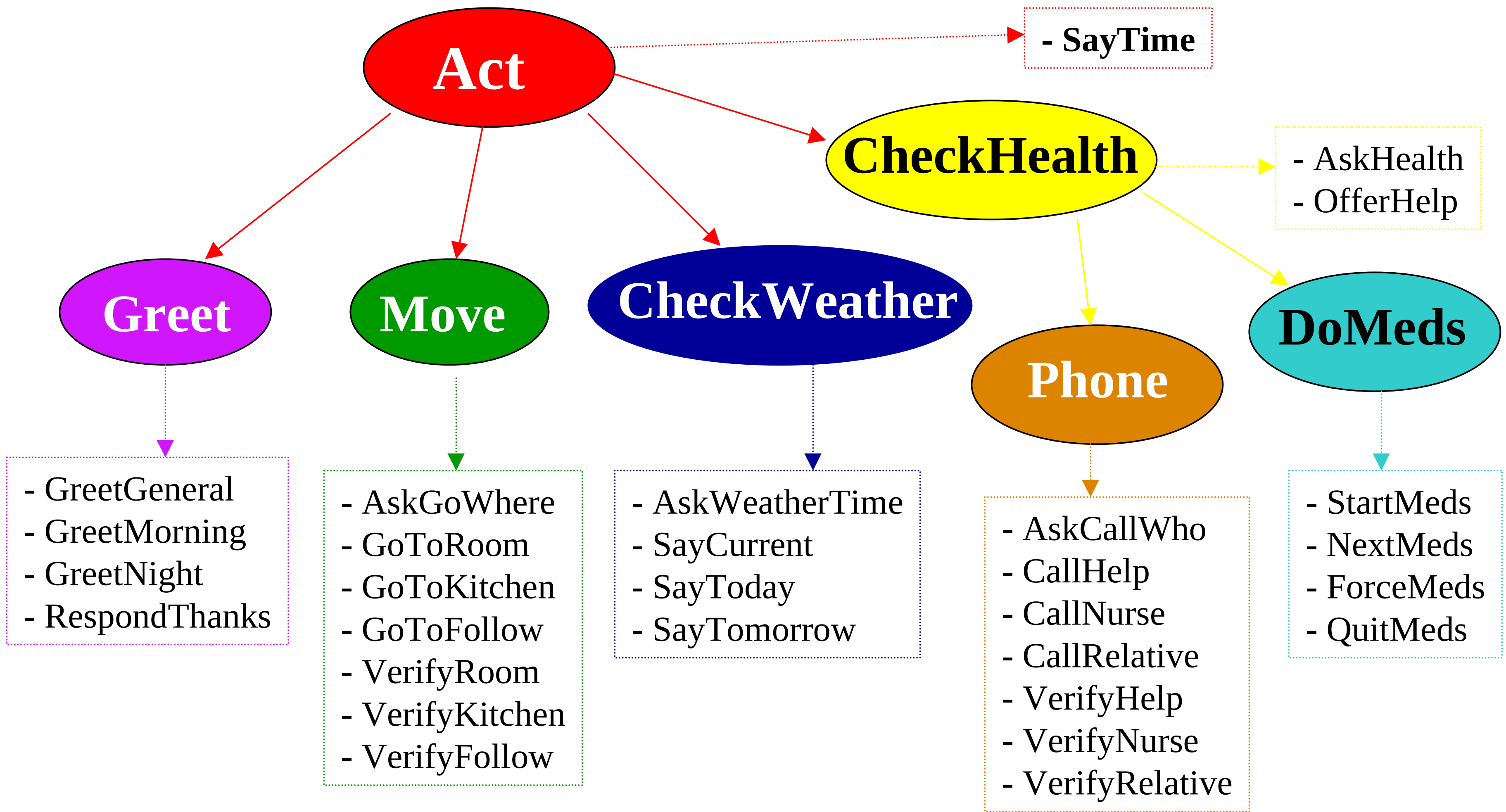
Bounding the value of the approximation

- Value function of top-level controller is an *upper-bound* on the value of the approximation.
  - Why? We were optimistic when modeling the abstract action.

- Similarly, we can find a *lower-bound.*
  - How? We can assume "worst-case" view when modeling the abstract action.

➔ *If we partition the action set differently, we will get different bounds.*

A real dialogue management example



Results:

	MDP	H-POMDP	POMDP
Navigation Problem: |S|=11, |A|=6, |O|=6			
CPU Time (secs):	0.000654	2.84	1119.93
Average Reward:	0.0	12.2	12.5
Dialogue Problem: |S|=20, |A|=30, |O|=27			
CPU Time (secs):	6.46	77.99	>24hrs
Average Reward:	53.33	64.43	
%Correct actions:	80.0	93.2	

Final words

- We presented:
  - a general framework to exploit structure in POMDPs;

- Future work:
  - automatic generation of good action partitioning;
  - conditions for additional observation abstraction;
  - bigger problems!