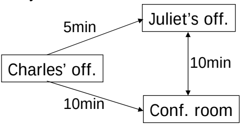
# Ch 17 - Making Complex Decisions

**Outline**

- Sequential Decision Problems
- Markov Decision Processes
- Optimal policy
- Value Iteration

# Example: Finding Juliet

- A robot, Romeo, is in Charles' office and must deliver a letter to Juliet
- Juliet is either in her office, or in the conference room. Without other prior knowledge, each possibility has probability 0.5

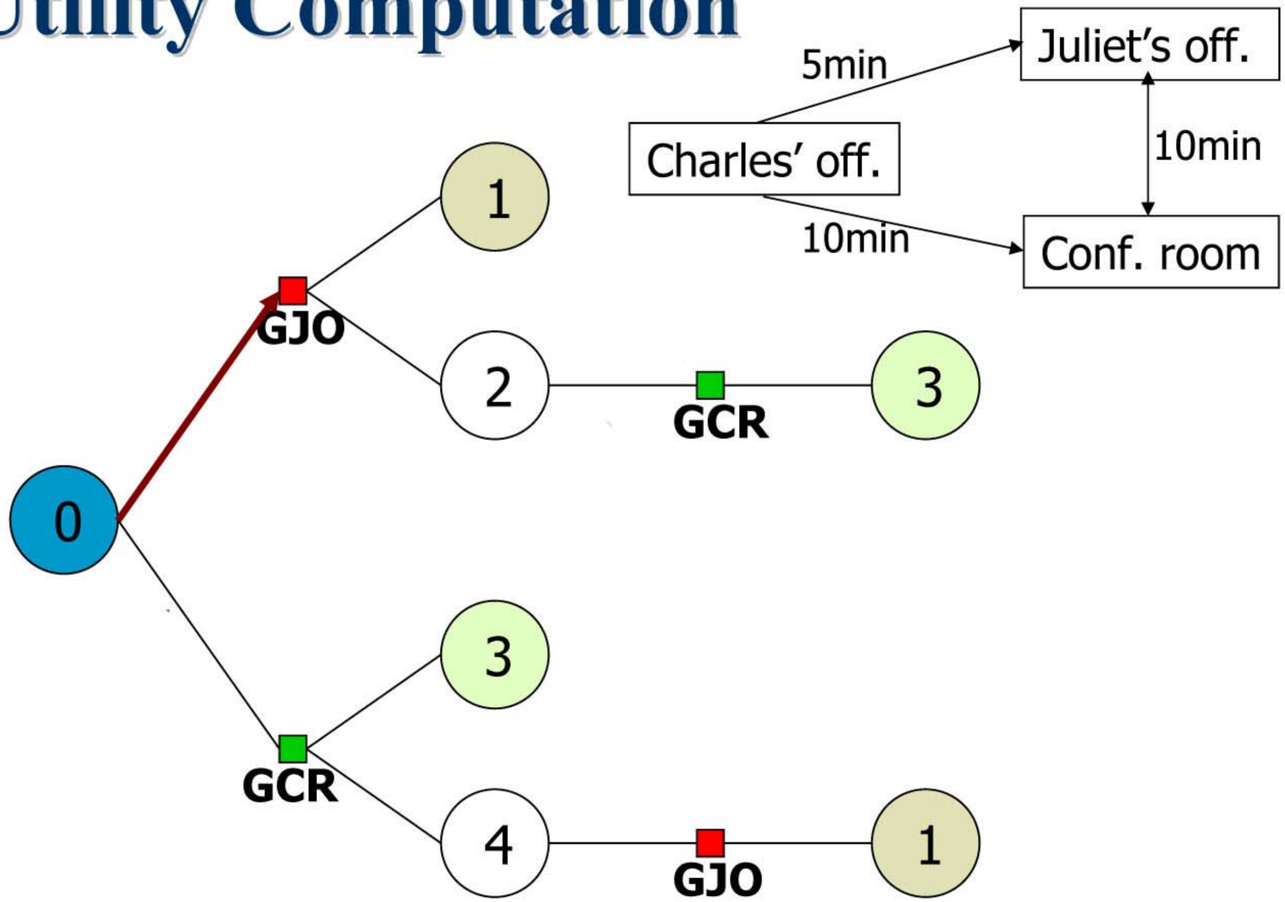



- The robot's goal is to minimize the time spent in transit

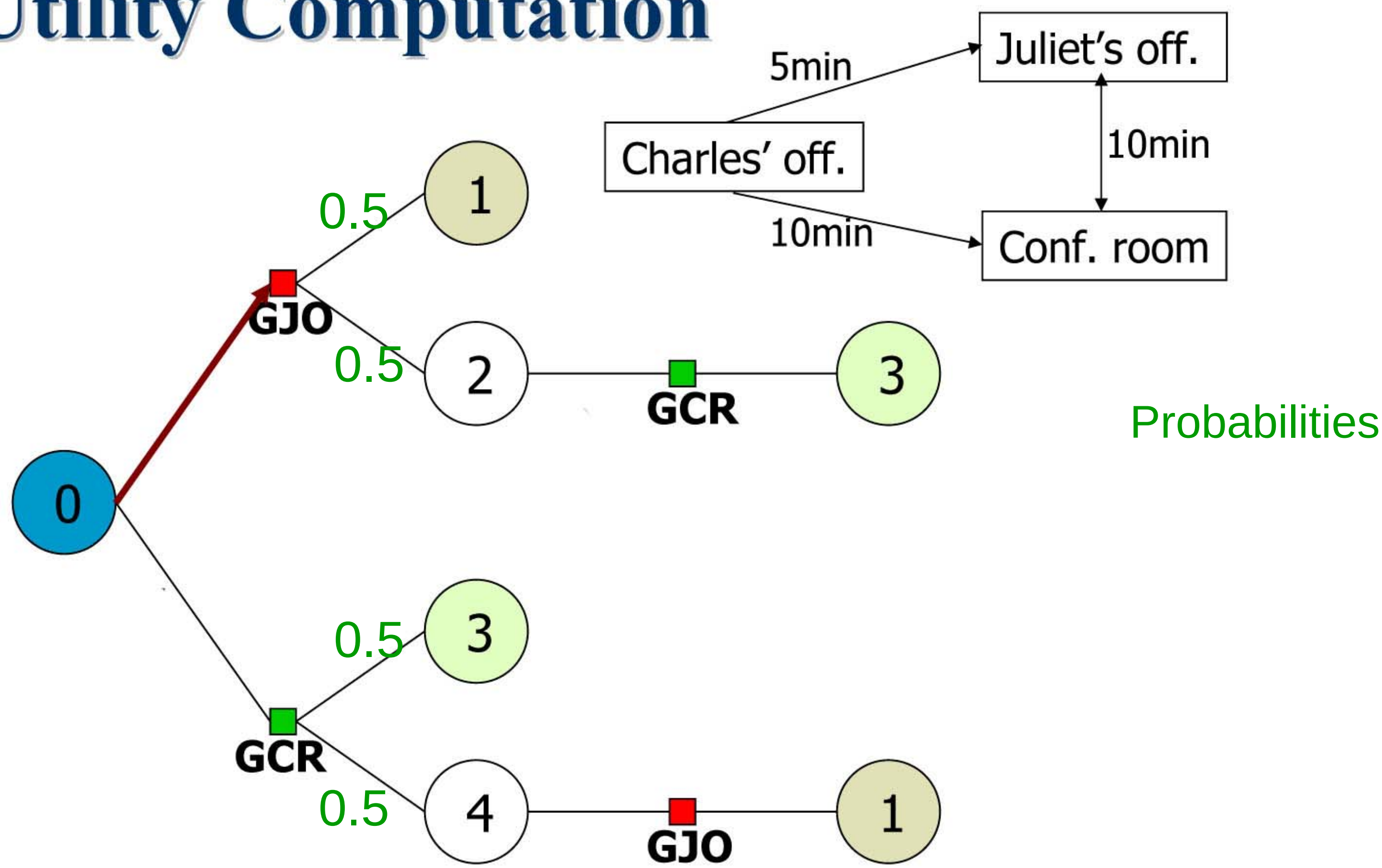
# Example: Finding Juliet

- States are:
  - S0: Romeo in Charles' office
  - S1: Romeo in Juliet's office and Juliet here
  - S2: Romeo in Juliet's office and Juliet not here
  - S3: Romeo in conference room and Juliet here
  - S4: Romeo in conference room and Juliet not here
- Actions are:
  - GJO (go to Juliet's office)
  - GCR (go to conference room)

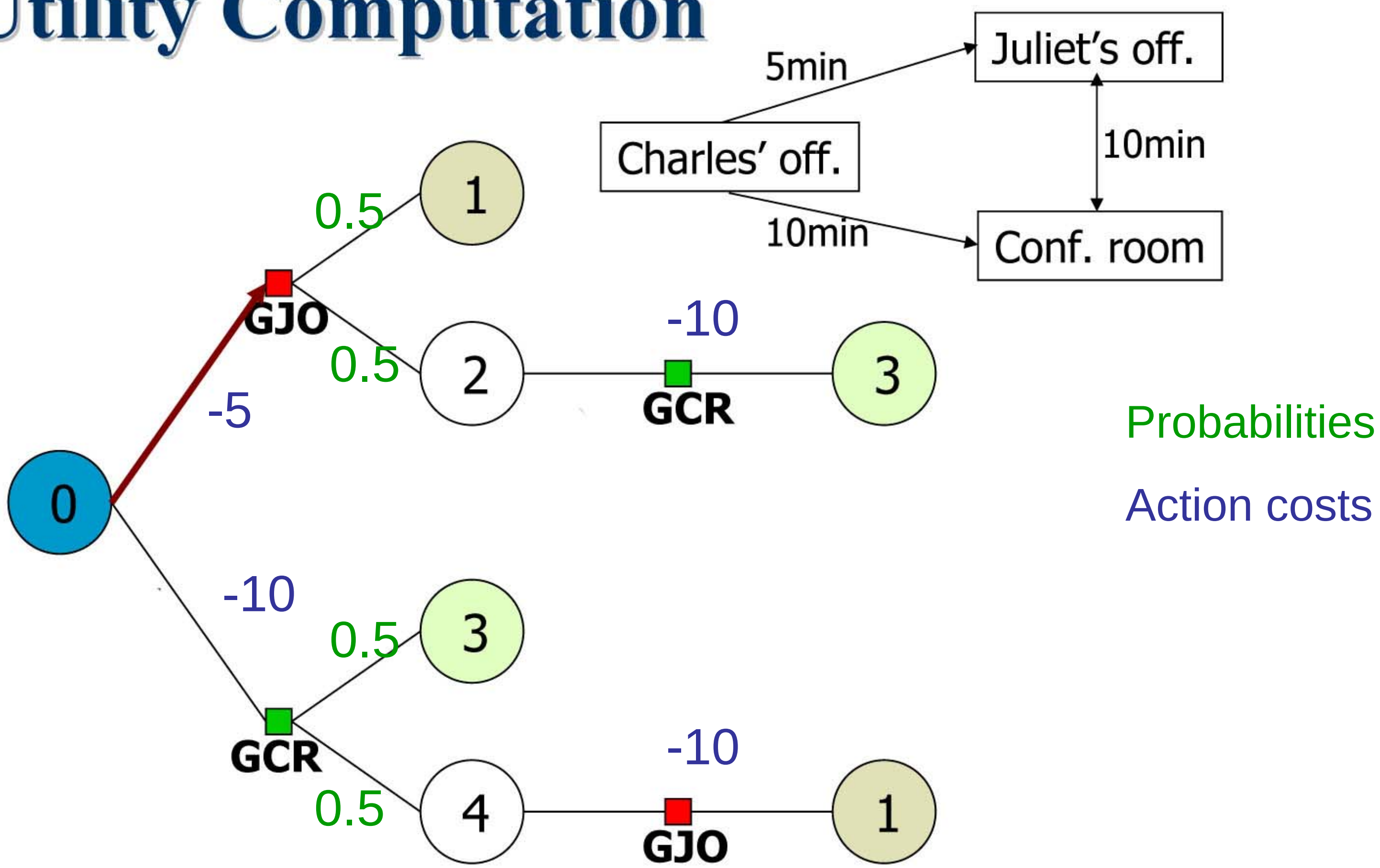
# Utility Computation



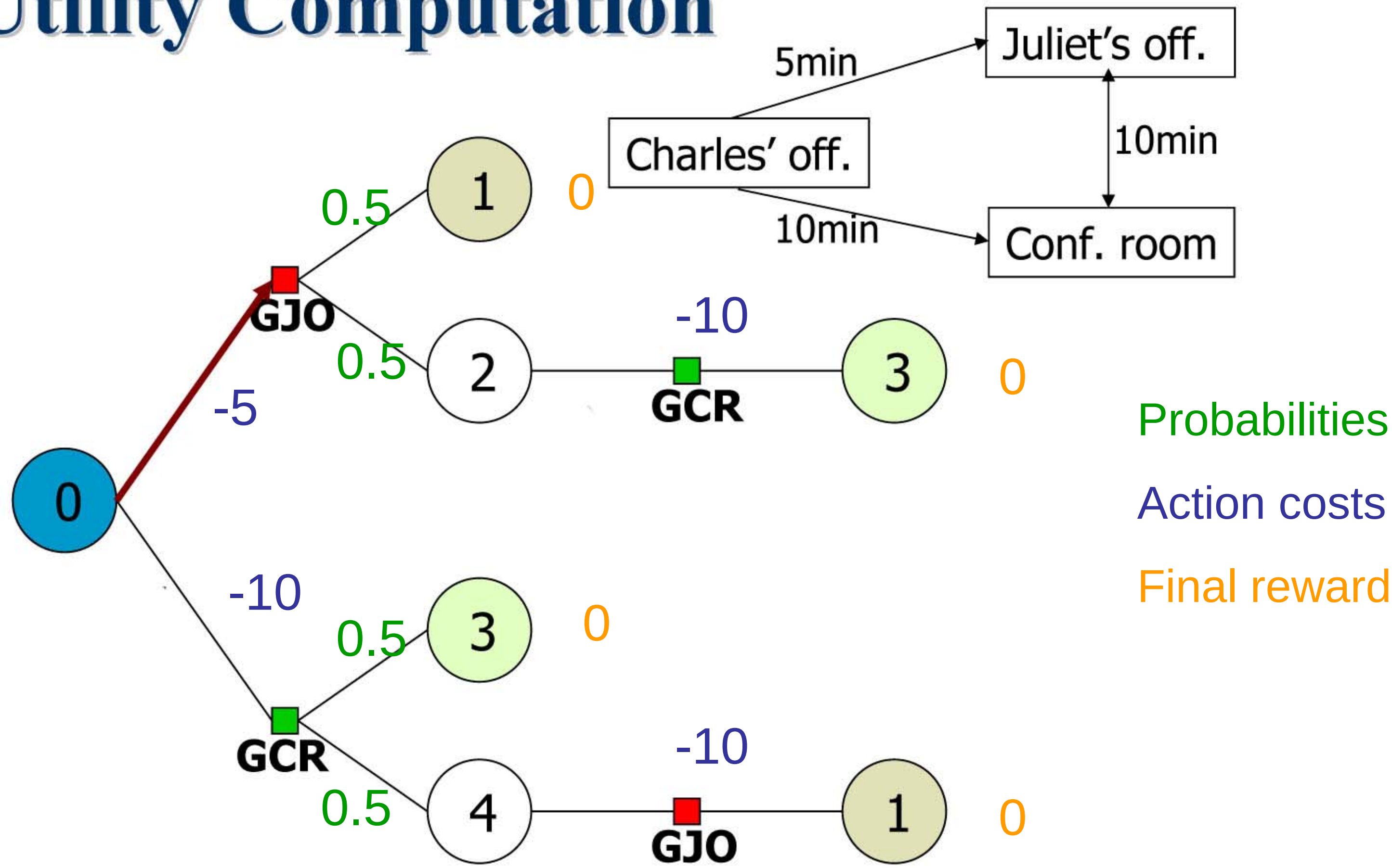
# Utility Computation



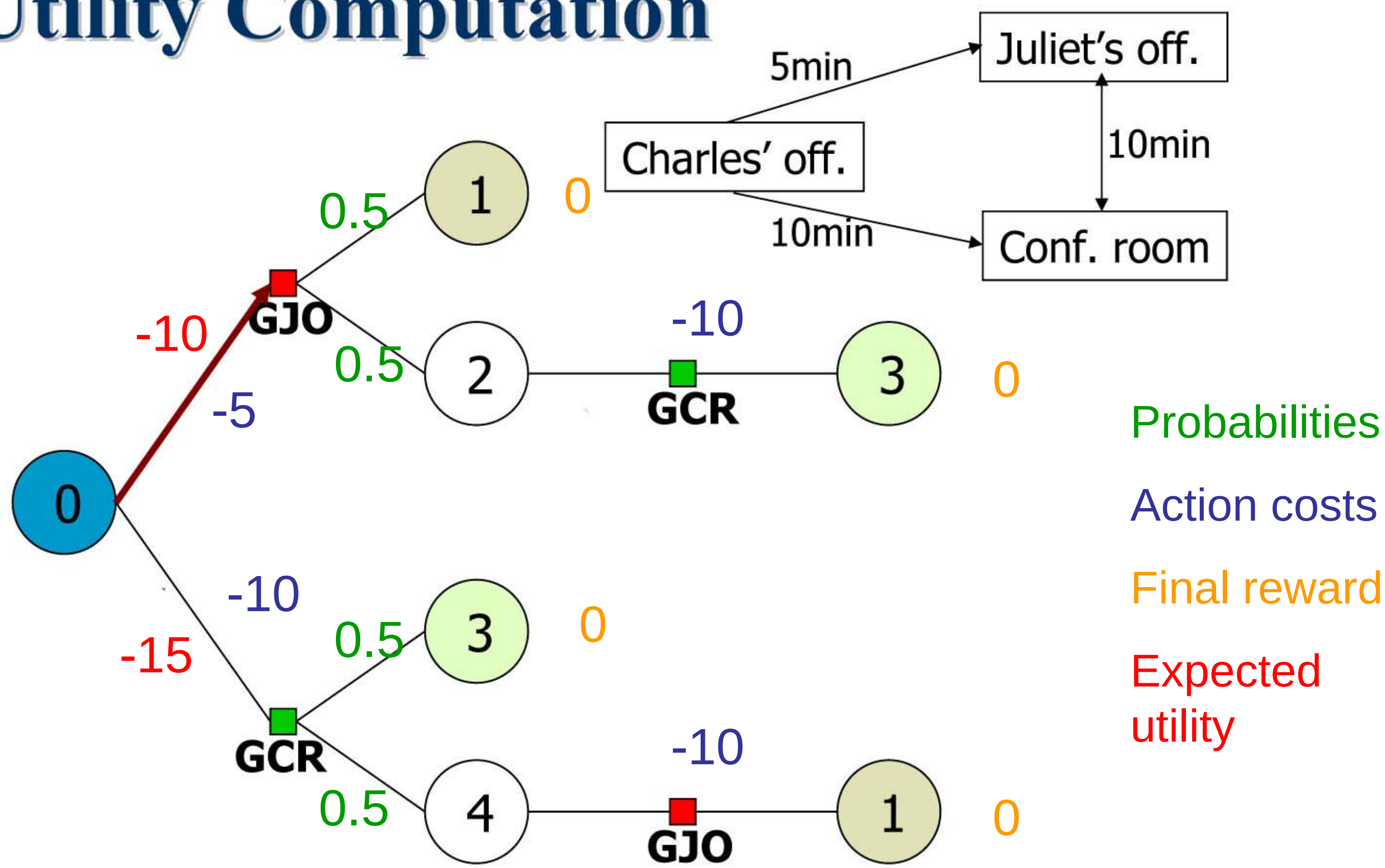
# Utility Computation



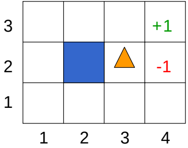
# Utility Computation



# Utility Computation

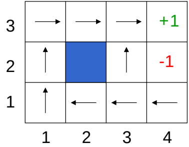


# Another example




- The robot needs to recharge its batteries
- [4,3] provides power supply
- [4,2] is a sand area from which the robot cannot escape
- [4,3] or [4,2] are terminal states
- Reward of a terminal state: +1 or -1
- Reward of a non-terminal state: -1/25
- **Utility of a history**: sum of rewards of traversed states
- Goal: Maximize expected reward by ???

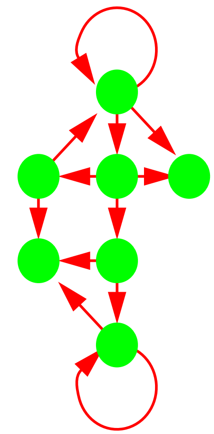
# Optimal Policy




- A policy P is a complete mapping from states to actions
- The optimal policy P* is the one that has the greatest expected reward among all policies.
- What if a terminal state is never reached?

# Stochastic Automata with Utilities

A *Markov Decision Process* (MDP) model contains:

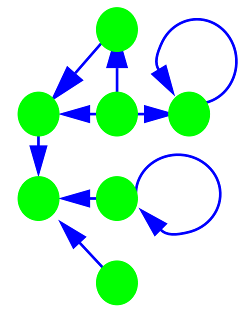


- A set of possible world states *S*
- A set of possible actions *A*
- A real valued reward function *R(s,a)*
- A description *T* of each action's effects in each state.

We assume the Markov Property: *the effects of an action taken in a state depend only on that state and not on the prior history.*

# Stochastic Automata with Utilities

A *Markov Decision Process* (MDP) model contains:



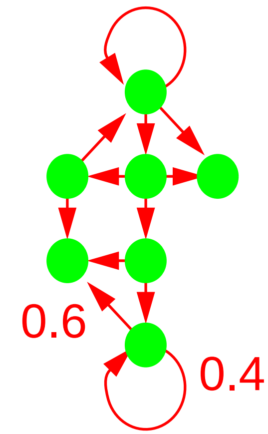
- A set of possible world states *S*
- A set of possible actions *A*
- A real valued reward function *R(s)*
- A description *T* of each action's effects in each state.

We assume the Markov Property: *the effects of an action taken in a state depend only on that state and not on the prior history.*

# Representing Actions

Deterministic Actions:

- $T: S \times A \rightarrow S$ For each state and action we specify a new state.



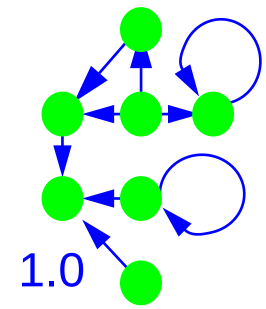

Stochastic Actions:

- $T: S \times A \rightarrow Prob(S)$ For each state and action we specify a probability distribution over next states. Represents the distribution $P(s' \mid s, a)$.

# Representing Actions

Deterministic Actions:

- $T: S \times A \rightarrow S$ For each state and action we specify a new state.

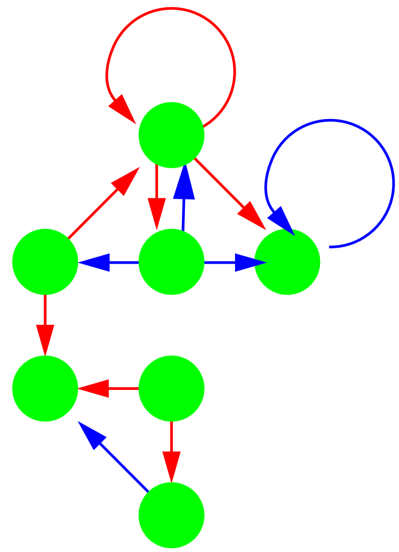



Stochastic Actions:

- $T: S \times A \rightarrow Prob(S)$ For each state and action we specify a probability distribution over next states. Represents the distribution $P(s' \mid s, a)$.

# Representing Solutions

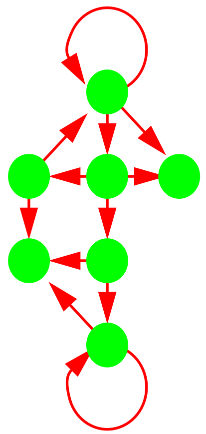
A *policy* $\pi$ is a mapping from $S$ to $A$



# Following a Policy

Following a policy $\pi$:

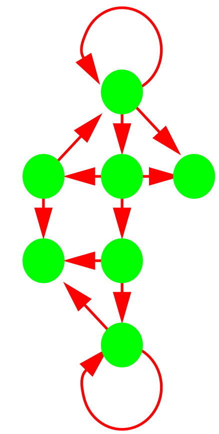
1. Determine the current state $s$
2. Execute action $\pi(s)$
3. Goto step 1.



Assumes full observability: the new state resulting from executing an action will be known to the system

# Evaluating a Policy

How good is a policy $\pi$ in a state $s$ ?



For deterministic actions just total the rewards obtained... but result may be infinite.

For stochastic actions, instead *expected total reward* obtained–again typically yields infinite value.

How do we compare policies of infinite value?

# Objective Functions

An objective function maps infinite sequences of rewards to single real numbers (representing utility)

Options:

1. Set a finite horizon and just total the reward
2. Discounting to prefer earlier rewards
3. Average reward rate in the limit

Discounting is perhaps the most analytically tractable and most widely studied approach

# Discounting

A reward *n* steps away is discounted by $\gamma^n$ for discount rate $0 < \gamma < 1$.

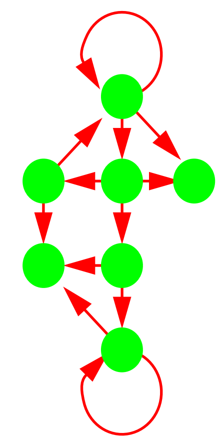
- models mortality: you may die at any moment
- models preference for shorter solutions
- a smoothed out version of limited horizon lookahead

We use *cumulative discounted reward* as our objective

(**Max value** $\le M + \gamma \cdot M + \gamma^2 \cdot M + \ldots = \frac{1}{1-\gamma} \cdot M$ )

# Value Functions

A value function $V_{\pi} : S \rightarrow \Re$ represents the expected objective value obtained following policy $\pi$ from each state in $S$.



Value functions partially order the policies,

- but at least one optimal policy exists, and
- all optimal policies have the same value function, $V^*$

# Bellman Equations

- Bellman equations give a recursive definition of the optimal expected reward

$$V^*(s) = R(s) + \max_a \gamma \sum_{s'} P(s' \mid s,a) V^*(s')$$

- If we can compute V*, we can easily find an optimal policy
  - o Choose an action with maximum expected reward

# Value Iteration

initialize V(s) to all 0

repeat until (change in V is small)

for each state s do

$$V'(s) := R(s) + \max_a \gamma \sum_{s'} P(s' \mid s, a) V(s')$$

end for

V:=V′

end repeat

# Demo

# Advanced Topics

- Explicit search through space of policies (policy iteration)
- Partial observations - POMDP
- Handling large state spaces
  - o Factored state and value function representations
  - o Approximate representations
    - Tile coding, neural networks, ...
- Simultaneously learning & solving an MDP or POMDP (reinforcement learning)