Probabilistic planning with full observability

- Several algorithms:
  1. dynamic programming (finite horizons)
  2. value iteration (discounted rewards, infinite horizon)
  3. policy iteration (discounted rewards, infinite horizon)
- Some of these algorithms can be easily implemented without explicitly representing the state space (e.g. by using algebraic decision diagrams ADDs).

Optimal plans over a finite horizon

- Simple algorithm based on dynamic programming. Idea:
  1. Value of a state at last stage N is the best immediate reward.
  2. Value of a state at earlier stages are obtained from values for later stages.
- No discounting is needed.
- Efficiency degrades with long horizons.

Optimal plans over a finite horizon

The optimum values $v_t(s)$ for states $s \in S$ at time $t \in \{1, \dots, N\}$ fulfill the following equations. ($A(s)$ = actions applicable in s.)

$$v_N(s) = \max_{a \in A(s)} R(s,a)$$

$$v_i(s) = \max_{a \in A(s)} \left(R(s,a) + \sum_{s' \in S} p(s'|s,a) v_{i+1}(s') \right) \text{ for } i \in \{1, \dots, N-1$$

Optimal plans over a finite horizon: plans

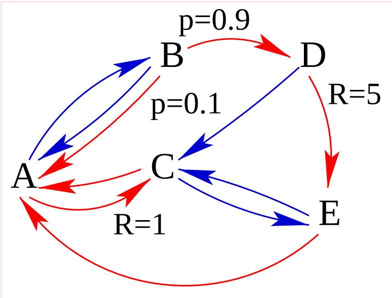
Action for state $s \in S$ at time t is $\pi(s,t)$ defined by

$$\pi(s,N) = \arg\max_{a \in A(s)} R(s,a)$$

$$\pi(s,i) = \arg\max_{a \in A(s)} \left(R(s,a) + \sum_{s' \in S} p(s'|s,a) v_{i+1}(s') \right) \text{ for } i \in \{1, \dots, N$$

If plan execution is *actually* unbounded, best action is $\pi(s,1)$! (*receding-horizon control*)

An example for demonstrating the algorithms

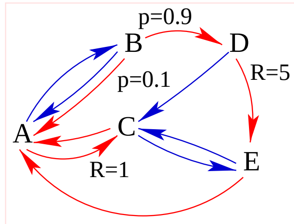


$$\begin{bmatrix} 0 & 0 & 1.0 & 0 & 0 \\ 0.1 & 0 & 0 & 0.9 & 0 \\ 1.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 \\ 1.0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1.0 & 0 & 0 & 0 \\ 1.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 \\ 0 & 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 & 0 \end{bmatrix}$$

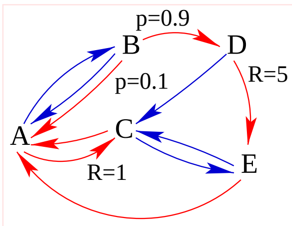
Total rewards with finite horizon: example

i	$v_i(A)$	$v_i(B)$	$v_i(C)$	$v_i(D)$	$v_i(E)$
9	1.0	0.0	0.0	5.0	0.0
8					
7					
6					
5					
4					
3					
2					
1					



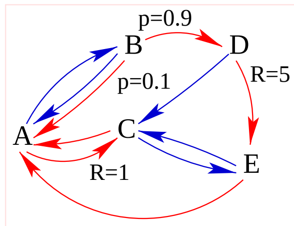
Total rewards with finite horizon: example

i	$v_i(A)$	$v_i(B)$	$v_i(C)$	$v_i(D)$	$v_i(E)$
9	1.0	0.0	0.0	5.0	0.0
8	1.0	4.6	1.0	5.0	1.0
7					
6					
5					
4					
3					
2					
1					



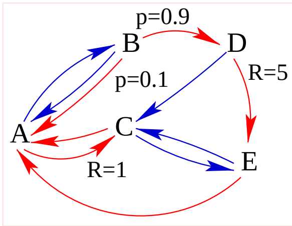
Total rewards with finite horizon: example

i	$v_i(A)$	$v_i(B)$	$v_i(C)$	$v_i(D)$	$v_i(E)$
9	1.0	0.0	0.0	5.0	0.0
8	1.0	4.6	1.0	5.0	1.0
7	4.6	4.6	1.0	6.0	1.0
6					
5					
4					
3					
2					
1					



Total rewards with finite horizon: example

i	$v_i(A)$	$v_i(B)$	$v_i(C)$	$v_i(D)$	$v_i(E)$
9	1.0	0.0	0.0	5.0	0.0
8	1.0	4.6	1.0	5.0	1.0
7	4.6	4.6	1.0	6.0	1.0
6	4.6	5.86	4.6	6.0	4.6
5	5.86	5.86	4.6	9.6	4.6
4	5.86	9.23	5.86	9.6	5.86
3	9.23	9.23	5.86	10.86	5.86
2	9.23	10.70	9.23	10.86	9.23
1	10.70	10.70	9.23	14.23	9.23



Optimality / Bellman equations (infinite horizon)

Values $v(s)$ of states $s \in S$ are the discounted sum of the expected rewards obtained by choosing the best possible actions in s and in its successors.

$$v(s) = \max_{a \in A(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v(s') \right) \quad (1)$$

λ is the discount constant: $0 < \lambda < 1$.

Plan evaluation by solving linear equations

Given a plan π, its value under discounted rewards with discount constant λ satisfies the following equation for every $s \in S$.

$$v(s) = R(s,\pi(s)) + \sum_{s' \in S} \lambda p(s'|s,\pi(s)) v(s') \quad (2)$$

This yields a system of $|S|$ linear equations and $|S|$ unknowns. The solution of these equations gives the value of the plan in each state.

Plan evaluation: example

Consider the plan

$$\pi(A) = R, \pi(B) = R, \pi(C) = B, \pi(D) = R, \pi(E) = B$$

$$\begin{aligned}
v_\pi(A) &= R(A,R) + 0\lambda v_\pi(A) + 0\lambda v_\pi(B) + 1\lambda v_\pi(C) + 0\lambda v_\pi(D) + 0\lambda v_\pi(E) \\
v_\pi(B) &= R(B,R) + 0.1\lambda v_\pi(A) + 0\lambda v_\pi(B) + 0\lambda v_\pi(C) + 0.9\lambda v_\pi(D) + 0\lambda v_\pi(E) \\
v_\pi(C) &= R(C,B) + 0\lambda v_\pi(A) + 0\lambda v_\pi(B) + 0\lambda v_\pi(C) + 0\lambda v_\pi(D) + 1\lambda v_\pi(E) \\
v_\pi(D) &= R(D,R) + 0\lambda v_\pi(A) + 0\lambda v_\pi(B) + 0\lambda v_\pi(C) + 0\lambda v_\pi(D) + 1\lambda v_\pi(E) \\
v_\pi(E) &= R(E,B) + 0\lambda v_\pi(A) + 0\lambda v_\pi(B) + 1\lambda v_\pi(C) + 0\lambda v_\pi(D) + 0\lambda v_\pi(E)
\end{aligned}$$

$$
\begin{array}{rcl}
v_\pi(A) & = & 1 \quad +\lambda v_\pi(C) \\
v_\pi(B) & = & 0 + 0.1\lambda v_\pi(A) \quad +0.9\lambda v_\pi(D) \\
v_\pi(C) & = & 0 \quad +\lambda v_\pi(E) \\
v_\pi(D) & = & 5 \quad +\lambda v_\pi(E) \\
v_\pi(E) & = & 0 \quad +\lambda v_\pi(C)
\end{array}
$$

$$
\begin{array}{rrrrrcl}
v_\pi(A) & & -\lambda v_\pi(C) & & & = & 1 \\
-0.1\lambda v_\pi(A) & +v_\pi(B) & & -0.9\lambda v_\pi(D) & & = & 0 \\
 & & v_\pi(C) & & -\lambda v_\pi(E) & = & 0 \\
 & & & v_\pi(D) & -\lambda v_\pi(E) & = & 5 \\
 & & -\lambda v_\pi(C) & & +v_\pi(E) & = & 0
\end{array}
$$

Solving with $\lambda = 0.5$ we get

$$
\begin{array}{lllllcl}
v_\pi(A) & & & & & = & 1 \\
 & v_\pi(B) & & & & = & 2.3 \\
 & & v_\pi(C) & & & = & 0 \\
 & & & v_\pi(D) & & = & 5 \\
 & & & & v_\pi(E) & = & 0
\end{array}
$$

This is the value function of the plan.

The Value Iteration algorithm

- Value Iteration is the simplest algorithm for finding plans that are close to optimal (discounted rewards, infinite horizon).
- Idea:

1. Start with an arbitrary value function.
2. Compute better and better approximations of the optimal value function.
3. From a good approximation construct a plan.

The Value Iteration algorithm: convergence

- Optimal value function is never reached.
- Plans extracted from a very-close-to-optimal value function *are typically* optimal.
- Important parameter ϵ:

  Algorithm terminates when change in the value function was smaller than $\frac{\epsilon(1-\lambda)}{2\lambda}$.

  Then distance from optimal value function is smaller than ϵ.

The Value Iteration algorithm: definition

1. $n := 0$
2. Assign initial values to $v^0(s)$ for all $s \in S$.
3.

$$v^{n+1}(s) := \max_{a \in A(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v^n(s') \right) \text{ for every } s \in S$$

If $|v^{n+1}(s) - v^n(s)| < \frac{\epsilon(1-\lambda)}{2\lambda}$ for all $s \in S$ then go to step 4.

Otherwise, set $n := n + 1$ and go to step 3.

4. Construct a plan: for every $s \in S$

$$\pi(s) := \arg\max_{a \in A(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v^{n+1}(s') \right)$$

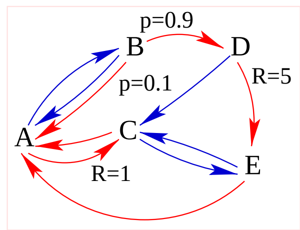
The Value Iteration algorithm: properties

THEOREM Let v_π be the value function of the plan produced by the value iteration algorithm, and let v^* be the value function of the optimal plan(s). Then $|v^*(s) - v_\pi(s)| \leq \epsilon$ for all $s \in S$.

Under full observability there is never a trade-off between the values of two states: if the optimal value for state s_1 is r_1 and the optimal value for state s_2 is r_2, then there is one plan that achieves these both.

Value iteration: example ($\lambda = 0.6$)

i	$v_i(A)$	$v_i(B)$	$v_i(C)$	$v_i(D)$	$v_i(E)$
0	0.000	0.000	0.000	0.000	0.000
1	1.000	0.000	0.000	5.000	0.000
2	1.000	2.760	0.600	5.000	0.600
3	1.656	2.760	0.600	5.360	0.600
4	1.656	2.994	0.994	5.360	0.994
5	1.796	2.994	0.994	5.596	0.994
6	1.796	3.130	1.078	5.596	1.078
7	1.878	3.130	1.078	5.647	1.078
8	1.878	3.162	1.127	5.647	1.127
⋮					
19	1.912	3.186	1.147	5.688	1.147
20	1.912	3.186	1.147	5.688	1.147



The Policy Iteration algorithm

- Finds optimal plans.
- Slightly more complicated to implement: each iteration consists of
  - evaluation of the value of the best plan so far, and
  - improving the plan.
- Number of iterations is small.
- Runtimes often higher than with value iteration.

The Policy Iteration algorithm: definition

1. Assign $n := 0$.
2. Let π^0 be any mapping from states $s \in S$ to actions in $A(s)$.
3. Compute the values $v^n(s)$ of all $s \in S$ under π^n.
4. Let $\pi^{n+1}(s) = \arg\max_{a \in A(s)} \left(R(s,a) + \sum_{s' \in S} \lambda p(s'|s,a) v^n(s')\right)$.
5. Assign $n := n + 1$.
6. If $n = 1$ or $v^n \neq v^{n-1}$ then go to 3.

The Policy Iteration algorithm: properties

THEOREM The policy iteration algorithm terminates after a finite number of steps and returns an optimal plan.

PROOF IDEA: There is only a finite number of different plans, and at each step a properly better plan is found or the algorithm terminates.

The number of iterations needed for finding an ϵ-optimal plan by policy iteration is never higher than the number of iterations needed by value iteration.

Policy iteration: example

itr.	$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$	$\pi(E)$	$v_\pi(A)$	$v_\pi(B)$	$v_\pi(C)$	$v_\pi(D)$	$v_\pi(E)$
1	R	R	R	R	R	1.56	3.09	0.93	5.56	0.93
2	B	R	R	R	R	1.91	3.18	1.14	5.68	1.14