Algorithms for MDP Planning

Shivaram Kalyanakrishnan

Department of Computer Science and Engineering
Indian Institute of Technology Bombay
shivaram@cse.iitb.ac.in

August 2018
Overview

1. Value Iteration

2. Linear Programming

3. Policy Iteration
   Policy Improvement Theorem

4. Complexity of algorithms
Overview

1. Value Iteration

2. Linear Programming

3. Policy Iteration
   Policy Improvement Theorem

4. Complexity of algorithms
Value Iteration

$V_0 \leftarrow$ Arbitrary, element-wise bounded, $n$-length vector. $t \leftarrow 0$.

Repeat:

For $s \in S$:

$$V_{t+1}(s) \leftarrow \max_{a \in A} \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V_t(s')).$$

$t \leftarrow t + 1$.

Until $V_t \approx V_{t-1}$ (up to machine precision).
Value Iteration

\[ V_0 \leftarrow \text{Arbitrary, element-wise bounded, n-length vector.} \quad t \leftarrow 0. \]

Repeat:

For \( s \in S \):

\[ V_{t+1}(s) \leftarrow \max_{a \in A} \sum_{s' \in S} T(s, a, s') \left( R(s, a, s') + \gamma V_t(s') \right). \]

\( t \leftarrow t + 1. \)

Until \( V_t \approx V_{t-1} \) (up to machine precision).

Convergence to \( V^* \) guaranteed using a max-norm contraction argument.
Overview

1. Value Iteration

2. Linear Programming

3. Policy Iteration
   Policy Improvement Theorem

4. Complexity of algorithms
Minimise \( \sum_{s \in S} V(s) \)

subject to \( V(s) \geq \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V(s')) , \forall s \in S, \forall a \in A. \)
Linear Programming

Minimise \[ \sum_{s \in S} V(s) \]
subject to \[ V(s) \geq \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V(s')) , \forall s \in S, \forall a \in A. \]

Let \(|S| = n\) and \(|A| = k\).
Minimise \[ \sum_{s \in S} V(s) \]
subject to \[ V(s) \geq \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V(s')) , \forall s \in S, \forall a \in A. \]

Let \(|S| = n\) and \(|A| = k\).

\(n\) variables, \(nk\) constraints.
Minimise \[ \sum_{s \in S} V(s) \]
subject to \[ V(s) \geq \sum_{s' \in S} T(s, a, s') (R(s, a, s') + \gamma V(s')) , \forall s \in S, \forall a \in A. \]

Let \(|S| = n\) and \(|A| = k\).

\(n\) variables, \(nk\) constraints.

Can also be posed as dual with \(nk\) variables and \(n\) constraints.
Overview

1. Value Iteration

2. Linear Programming

3. Policy Iteration
   - Policy Improvement Theorem

4. Complexity of algorithms
Policy Improvement
Policy Improvement
Policy Improvement

\[ Q^\pi(s_3, \square) \leq Q^\pi(s_3, \blacksquare) \]
Policy Improvement

Improvable states
Policy Improvement

π

Improving actions

Improvable states

S1  S2  S3  S4  S5  S6  S7  S8
Given $\pi$, pick one or more improvable states, and in them, switch to an arbitrary improving action. Let the resulting policy be $\pi'$. 
Given $\pi$, pick one or more improvable states, and in them, switch to an arbitrary improving action. Let the resulting policy be $\pi'$.
Policy Improvement

Given $\pi$, 
Pick one or more improvable states, and in them, 
Switch to an arbitrary improving action. 
Let the resulting policy be $\pi'$.

Policy Improvement Theorem:
(1) If $\pi$ has no improvable states, then it is optimal, else
(2) if $\pi'$ is obtained as above, then
\[ \forall s \in S : V^{\pi'}(s) \geq V^{\pi}(s) \] and 
\[ \exists s \in S : V^{\pi'}(s) > V^{\pi}(s). \]
Policy Improvement

Given $\pi$, pick one or more improvable states, and in them, switch to an arbitrary improving action. Let the resulting policy be $\pi'$.

Policy Improvement Theorem:
(1) If $\pi$ has no improvable states, then it is optimal, else
(2) if $\pi'$ is obtained as above, then

$$\forall s \in S : V^{\pi'}(s) \geq V^\pi(s) \text{ and } \exists s \in S : V^{\pi'}(s) > V^\pi(s).$$
Definitions and Basic Facts

For $X : S \to \mathbb{R}$ and $Y : S \to \mathbb{R}$, we define $X \succeq Y$ if $\forall s \in S : X(s) \geq Y(s)$, and we define $X \succ Y$ if $X \succeq Y$ and $\exists s \in S : X(s) > Y(s)$. 
Definitions and Basic Facts

For $X : S \rightarrow \mathbb{R}$ and $Y : S \rightarrow \mathbb{R}$, we define $X \succeq Y$ if $\forall s \in S : X(s) \geq Y(s)$, and we define $X \succ Y$ if $X \succeq Y$ and $\exists s \in S : X(s) > Y(s)$.

For policies $\pi_1, \pi_2 \in \Pi$, we define $\pi_1 \succeq \pi_2$ if $V^{\pi_1} \succeq V^{\pi_2}$, and we define $\pi_1 \succ \pi_2$ if $V^{\pi_1} \succ V^{\pi_2}$. 
Definitions and Basic Facts

- For $X : S \to \mathbb{R}$ and $Y : S \to \mathbb{R}$, we define $X \succeq Y$ if $\forall s \in S : X(s) \geq Y(s)$, and we define $X \succ Y$ if $X \succeq Y$ and $\exists s \in S : X(s) > Y(s)$.

  For policies $\pi_1, \pi_2 \in \Pi$, we define $\pi_1 \succeq \pi_2$ if $V^{\pi_1} \succeq V^{\pi_2}$, and we define $\pi_1 \succ \pi_2$ if $V^{\pi_1} \succ V^{\pi_2}$.

- **Bellman Operator.** For $\pi \in \Pi$, we define $B^{\pi} : (S \to \mathbb{R}) \to (S \to \mathbb{R})$ as follows: for $X : S \to \mathbb{R}$ and $\forall s \in S$,

$$
(B^{\pi}(X))(s) \overset{\text{def}}{=} \sum_{s' \in S} T(s, \pi(s), s') \left(R(s, \pi(s), s') + \gamma X(s')\right).
$$
Definitions and Basic Facts

For $X : S \to \mathbb{R}$ and $Y : S \to \mathbb{R}$, we define $X \succeq Y$ if $\forall s \in S : X(s) \geq Y(s)$, and we define $X \succ Y$ if $X \succeq Y$ and $\exists s \in S : X(s) > Y(s)$.

For policies $\pi_1, \pi_2 \in \Pi$, we define $\pi_1 \succeq \pi_2$ if $V^{\pi_1} \succeq V^{\pi_2}$, and we define $\pi_1 \succ \pi_2$ if $V^{\pi_1} \succ V^{\pi_2}$.

**Bellman Operator.** For $\pi \in \Pi$, we define $B^\pi : (S \to \mathbb{R}) \to (S \to \mathbb{R})$ as follows:

For $X : S \to \mathbb{R}$ and $\forall s \in S$,

$$(B^\pi(X))(s) \overset{\text{def}}{=} \sum_{s' \in S} T(s, \pi(s), s') (R(s, \pi(s), s') + \gamma X(s')) .$$

**Fact 1.** For $\pi \in \Pi$, $X : S \to \mathbb{R}$, and $Y : S \to \mathbb{R}$:

if $X \succeq Y$, then $B^\pi(X) \succeq B^\pi(Y)$. 
Definitions and Basic Facts

- For \( X : S \rightarrow \mathbb{R} \) and \( Y : S \rightarrow \mathbb{R} \), we define \( X \succeq Y \) if \( \forall s \in S : X(s) \geq Y(s) \), and we define \( X \succ Y \) if \( X \succeq Y \) and \( \exists s \in S : X(s) > Y(s) \).

- For policies \( \pi_1, \pi_2 \in \Pi \), we define \( \pi_1 \succeq \pi_2 \) if \( V^{\pi_1} \succeq V^{\pi_2} \), and we define \( \pi_1 \succ \pi_2 \) if \( V^{\pi_1} \succ V^{\pi_2} \).

- **Bellman Operator.** For \( \pi \in \Pi \), we define \( B^\pi : (S \rightarrow \mathbb{R}) \rightarrow (S \rightarrow \mathbb{R}) \) as follows:
  
  for \( X : S \rightarrow \mathbb{R} \) and \( \forall s \in S \),
  
  \[
  (B^\pi(X))(s) \overset{\text{def}}{=} \sum_{s' \in S} T(s, \pi(s), s') \left( R(s, \pi(s), s') + \gamma X(s') \right).
  \]

- **Fact 1.** For \( \pi \in \Pi \), \( X : S \rightarrow \mathbb{R} \), and \( Y : S \rightarrow \mathbb{R} \):
  
  if \( X \succeq Y \), then \( B^\pi(X) \succeq B^\pi(Y) \).

- **Fact 2.** For \( \pi \in \Pi \) and \( X : S \rightarrow \mathbb{R} \):
  
  \[
  \lim_{l \rightarrow \infty} (B^\pi)^l(X) = V^\pi. \text{ (from Banach’s FP Theorem)}
  \]
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi$, $\forall s \in S$: $B^{\pi'}(V^\pi)(s) = Q^\pi(s, \pi'(s))$. 
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi, \forall s \in S: B^{\pi'}(V^\pi)(s) = Q^\pi(s, \pi'(s))$.

$\pi$ has no improvable states

$\implies \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi)$
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi$, $\forall s \in S$: $B^{\pi'}(V^\pi)(s) = Q^\pi(s, \pi'(s))$.

$\pi$ has no improvable states

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^{\pi'}(V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi)$
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi$, $\forall s \in S$: $B^{\pi'}(V^\pi)(s) = Q^\pi(s, \pi'(s))$.

$\pi$ has no improvable states

\[\Rightarrow \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi)\]
\[\Rightarrow \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi)\]
\[\Rightarrow \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi) \succeq \cdots \succeq \lim_{l \to \infty} (B^{\pi'})^l(V^\pi)\]
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi, \forall s \in S$: $B_{\pi'}^{\prime}(V^{\pi})(s) = Q^{\pi}(s, \pi'(s))$.

$\pi$ has no improvable states

$$\implies \forall \pi' \in \Pi : V^{\pi} \succeq B_{\pi'}^{\prime}(V^{\pi})$$

$$\implies \forall \pi' \in \Pi : V^{\pi} \succeq B_{\pi'}^{\prime}(V^{\pi}) \succeq (B_{\pi'}^{\prime})^2(V^{\pi})$$

$$\implies \forall \pi' \in \Pi : V^{\pi} \succeq B_{\pi'}^{\prime}(V^{\pi}) \succeq (B_{\pi'}^{\prime})^2(V^{\pi}) \succeq \cdots \succeq \lim_{l \to \infty} (B_{\pi'}^{\prime})^l(V^{\pi})$$

$$\implies \forall \pi' \in \Pi : V^{\pi} \succeq V^{\pi'}.$$
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi$, $\forall s \in S$: $B^\pi (V^\pi)(s) = Q^\pi (s, \pi'(s))$. 

$\pi$ has no improvable states

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^\pi (V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^\pi (V^\pi) \succeq (B^\pi')^2 (V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^\pi (V^\pi) \succeq (B^\pi')^2 (V^\pi) \succeq \cdots \succeq \lim_{l \to \infty} (B^\pi')^l (V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq V^\pi'$. 

$\pi$ has improvable states and policy improvement yields $\pi'$. 
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi$, $\forall s \in S$: $B^{\pi'}(V^\pi)(s) = Q^\pi(s, \pi'(s))$.

$\pi$ has no improvable states

$\implies \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi)$

$\implies \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi)$

$\implies \forall \pi' \in \Pi : V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi) \succeq \cdots \succeq \lim_{l \to \infty} (B^{\pi'})^l(V^\pi)$

$\implies \forall \pi' \in \Pi : V^\pi \succeq V^{\pi'}$.

$\pi$ has improvable states and policy improvement yields $\pi'$

$\implies B^{\pi'}(V^\pi) \succ V^\pi$
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi$, $\forall s \in S$: $B^{\pi'}(V^\pi)(s) = Q^\pi(s, \pi'(s))$.

$\pi$ has no improvable states

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^{\pi'}(V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq B^{\pi'}(V^\pi) \succeq (B^{\pi'})^2(V^\pi) \succeq \cdots \succeq \lim_{l \to \infty}(B^{\pi'})^l(V^\pi)$

$\implies \forall \pi' \in \Pi: V^\pi \succeq V^{\pi'}$.

$\pi$ has improvable states and policy improvement yields $\pi'$

$\implies B^{\pi'}(V^\pi) \succ V^\pi$

$\implies (B^{\pi'})^2(V^\pi) \succeq B^{\pi'}(V^\pi) \succ V^\pi$
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi, \forall s \in S: B^\pi' (V^\pi)(s) = Q^\pi (s, \pi'(s))$.

$\pi$ has no improvable states

$$\implies \forall \pi' \in \Pi : V^\pi \succeq B^\pi'(V^\pi)$$

$$\implies \forall \pi' \in \Pi : V^\pi \succeq B^\pi'(V^\pi) \succeq (B^\pi')^2(V^\pi)$$

$$\implies \forall \pi' \in \Pi : V^\pi \succeq B^\pi'(V^\pi) \succeq (B^\pi')^2(V^\pi) \succeq \cdots \succeq \lim_{l \to \infty} (B^\pi')^l(V^\pi)$$

$$\implies \forall \pi' \in \Pi : V^\pi \succeq V^{\pi'}.$$ 

$\pi$ has improvable states and policy improvement yields $\pi'$

$$\implies B^\pi'(V^\pi) \succ V^\pi$$

$$\implies (B^\pi')^2(V^\pi) \succeq B^\pi'(V^\pi) \succ V^\pi$$

$$\implies \lim_{l \to \infty} (B^\pi')^l(V^\pi) \succeq \cdots \succeq (B^\pi')^2(V^\pi) \succeq B^\pi'(V^\pi) \succ V^\pi$$
Proof of Policy Improvement Theorem

Observe that for $\pi, \pi' \in \Pi, \forall s \in S: B^{\pi'}(V^{\pi})(s) = Q^{\pi}(s, \pi'(s))$.

$\pi$ has no improvable states

\[
\Rightarrow \ \forall \pi' \in \Pi : V^{\pi} \succeq B^{\pi'}(V^{\pi})
\Rightarrow \ \forall \pi' \in \Pi : V^{\pi} \succeq B^{\pi'}(V^{\pi}) \succeq (B^{\pi'})^2(V^{\pi})
\Rightarrow \ \forall \pi' \in \Pi : V^{\pi} \succeq B^{\pi'}(V^{\pi}) \succeq (B^{\pi'})^2(V^{\pi}) \succeq \cdots \succeq \lim_{l \to \infty} (B^{\pi'})^l(V^{\pi})
\Rightarrow \ \forall \pi' \in \Pi : V^{\pi} \succeq V^{\pi'}.
\]

$\pi$ has improvable states and policy improvement yields $\pi'$

\[
\Rightarrow \ B^{\pi'}(V^{\pi}) \succ V^{\pi}
\Rightarrow \ (B^{\pi'})^2(V^{\pi}) \succeq B^{\pi'}(V^{\pi}) \succ V^{\pi}
\Rightarrow \ \lim_{l \to \infty} (B^{\pi'})^l(V^{\pi}) \succeq \cdots \succeq (B^{\pi'})^2(V^{\pi}) \succeq B^{\pi'}(V^{\pi}) \succ V^{\pi}
\Rightarrow \ V^{\pi'} \succ V^{\pi}.
\]
Policy Iteration Algorithm

\[ \pi \leftarrow \text{Arbitrary policy.} \]

\textbf{While } \pi \text{ has improvable states:} \\
\quad \pi \leftarrow \text{PolicyImprovement}(\pi).
Policy Iteration Algorithm

\[ \pi \leftarrow \text{Arbitrary policy.} \]

**While** \( \pi \) has improvable states:

\[ \pi \leftarrow \text{PolicyImprovement}(\pi). \]

Policy Iteration Algorithm

\[ \pi \leftarrow \text{Arbitrary policy.} \]

\textbf{While} \( \pi \) has improvable states:

\[ \pi \leftarrow \text{PolicyImprovement}(\pi). \]
\[ \pi \leftarrow \text{Arbitrary policy.} \]

**While** \( \pi \) has improvable states:

\[ \pi \leftarrow \text{PolicyImprovement}(\pi). \]
Policy Iteration Algorithm

\[ \pi \leftarrow \text{Arbitrary policy.} \]

**While** \( \pi \) has improvable states:

\[ \pi \leftarrow \text{PolicyImprovement}(\pi). \]
$\pi \leftarrow$ Arbitrary policy.

**While** $\pi$ has improvable states:

$\pi \leftarrow$ PolicyImprovement($\pi$).
Policy Iteration Algorithm

\[ \pi \leftarrow \text{Arbitrary policy.} \]

\textbf{While} \( \pi \) has improvable states:

\[ \pi \leftarrow \text{PolicyImprovement}(\pi). \]
Overview

1. Value Iteration

2. Linear Programming

3. Policy Iteration
   Policy Improvement Theorem

4. Complexity of algorithms
Overview

1. Value Iteration

2. Linear Programming

3. Policy Iteration
   - Policy Improvement Theorem

4. Complexity of algorithms *(not a part of course syllabus!)*
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \], where \( B \) is the number of bits used to represent the MDP.
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \], where \( B \) is the number of bits used to represent the MDP.
  Not a strong bound.
Weak and Strong Running-time Bounds

■ Computation model: Infinite precision arithmetic (or Real RAM) model.

■ Upper Bound for Value Iteration [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \], where \( B \) is the number of bits used to represent the MDP. Not a strong bound.

■ Strong bounds depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma, \) etc.).
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \]
  where \( B \) is the number of bits used to represent the MDP.
  Not a strong bound.

- Strong bounds depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma \), etc.).
  Is there a strong upper bound on the complexity of policy evaluation?
Weak and Strong Running-time Bounds

■ Computation model: Infinite precision arithmetic (or Real RAM) model.

■ Upper Bound for Value Iteration [LDK95]: $\text{poly}(n, k, B, \frac{1}{1-\gamma})$, where $B$ is the number of bits used to represent the MDP. Not a strong bound.

■ Strong bounds depend solely on $n$ and $k$ (no dependence on $B, \gamma$, etc.).
Is there a strong upper bound on the complexity of policy evaluation? $O(n^2k + n^3)$. 
Weak and Strong Running-time Bounds

- Computation model: **Infinite precision arithmetic** (or **Real RAM**) model.

- **Upper Bound for Value Iteration** [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \], where \( B \) is the number of bits used to represent the MDP. Not a **strong** bound.

- **Strong bounds** depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma, \) etc.).

  - Is there a strong upper bound on the complexity of **policy evaluation**? \( O(n^2k + n^3) \).
  - Can you give a strong bound on the running time of MDP planning?
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \]
  where \( B \) is the number of bits used to represent the MDP.
  Not a strong bound.

- Strong bounds depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma, \) etc.).
  Is there a strong upper bound on the complexity of policy evaluation? \( O(n^2 k + n^3) \).
  Can you give a strong bound on the running time of MDP planning? \( \text{poly}(n, k) \cdot k^n \).
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \( \text{poly}(n, k, B, \frac{1}{1-\gamma}) \), where \( B \) is the number of bits used to represent the MDP. Not a strong bound.

- Strong bounds depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma \), etc.).
  Is there a strong upper bound on the complexity of policy evaluation? \( O(n^2 k + n^3) \).
  Can you give a strong bound on the running time of MDP planning? \( \text{poly}(n, k) \cdot k^n \).

- Bounds for Linear Programming-type approaches to MDP planning:
  \( \text{poly}(n, k, B) \) [K80, K84].
  \( \text{poly}(n, k) \cdot \exp(O(\sqrt{n\log(n)})) \) (Expected) [MSW96].
  \( \text{poly}(n, k) \cdot k^{0.6834n} \) [GK17].
Weak and Strong Running-time Bounds

- **Computation model:** Infinite precision arithmetic (or Real RAM) model.

- **Upper Bound for Value Iteration** [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \], where \( B \) is the number of bits used to represent the MDP. Not a strong bound.

- **Strong bounds** depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma \), etc.).

  Is there a strong upper bound on the complexity of **policy evaluation**? \( O(n^2k + n^3) \).

  Can you give a strong bound on the running time of MDP planning? \( \text{poly}(n, k) \cdot k^n \).

- **Bounds for Linear Programming-type approaches to MDP planning:**
  \( \text{poly}(n, k, B) \) [K80, K84].
  \( \text{poly}(n, k) \cdot \exp(O(\sqrt{n\log(n)})) \) (Expected) [MSW96].
  \( \text{poly}(n, k) \cdot k^{0.6834n} \) [GK17].
  \( \text{poly}(n, k) \) for deterministic MDPs [MTZ10, PY13].
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \[ \text{poly}(n, k, B, \frac{1}{1-\gamma}) \]
  where \( B \) is the number of bits used to represent the MDP. Not a strong bound.

- Strong bounds depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma, \) etc.).
  Is there a strong upper bound on the complexity of policy evaluation? \( O(n^2k + n^3) \).
  Can you give a strong bound on the running time of MDP planning? \( \text{poly}(n, k) \cdot k^n \).

- Bounds for Linear Programming-type approaches to MDP planning:
  \[ \text{poly}(n, k, B) \] [K80, K84].
  \[ \text{poly}(n, k) \cdot \exp(O(\sqrt{n\log(n)})) \] (Expected) [MSW96].
  \[ \text{poly}(n, k) \cdot k^{0.6834n} \] [GK17].
  \[ \text{poly}(n, k) \] for deterministic MDPs [MTZ10, PY13].

- Complexity of Policy Iteration trivially upper-bounded by \( \text{poly}(n, k) \cdot k^n \).
Weak and Strong Running-time Bounds

- Computation model: Infinite precision arithmetic (or Real RAM) model.

- Upper Bound for Value Iteration [LDK95]:
  \( \text{poly}(n, k, B, \frac{1}{1-\gamma}) \), where \( B \) is the number of bits used to represent the MDP.
  Not a strong bound.

- Strong bounds depend solely on \( n \) and \( k \) (no dependence on \( B, \gamma \), etc.).
  Is there a strong upper bound on the complexity of policy evaluation? \( O(n^2k + n^3) \).
  Can you give a strong bound on the running time of MDP planning? \( \text{poly}(n, k) \cdot k^n \).

- Bounds for Linear Programming-type approaches to MDP planning:
  \( \text{poly}(n, k, B) \) [K80, K84].
  \( \text{poly}(n, k) \cdot \exp(O(\sqrt{n\log(n)})) \) (Expected) [MSW96].
  \( \text{poly}(n, k) \cdot k^{0.6834n} \) [GK17].
  \( \text{poly}(n, k) \) for deterministic MDPs [MTZ10, PY13].

- Complexity of Policy Iteration trivially upper-bounded by \( \text{poly}(n, k) \cdot k^n \).
  Is it more efficient than that?
Switching Strategies and Bounds for Policy Iteration

<table>
<thead>
<tr>
<th>PI Variant</th>
<th>Type</th>
<th>$k = 2$</th>
<th>General $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Howard’s (“all switch”) PI [H60, MS99]</td>
<td>Deterministic</td>
<td>$O\left(\frac{2^n}{n}\right)$</td>
<td>$O\left(\frac{k^n}{n}\right)$</td>
</tr>
<tr>
<td>Mansour and Singh’s Randomised PI [MS99]</td>
<td>Randomised</td>
<td>$1.7172^n$</td>
<td>$\approx O\left(\left(\frac{k}{2}\right)^n\right)$</td>
</tr>
</tbody>
</table>
### Upper bounds on number of iterations

<table>
<thead>
<tr>
<th>PI Variant</th>
<th>Type</th>
<th>$k = 2$</th>
<th>General $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Howard’s (“all switch”) PI</td>
<td>Deterministic</td>
<td>$O\left(\frac{2^n}{n}\right)$</td>
<td>$O\left(\frac{k^n}{n}\right)$</td>
</tr>
<tr>
<td>Mansour and Singh’s Randomised PI</td>
<td>Randomised</td>
<td>$1.7172^n$</td>
<td>$\approx O\left(\left(\frac{k}{2}\right)^n\right)$</td>
</tr>
<tr>
<td>Batch-switching PI (BSPI) [KMG16a]</td>
<td>Deterministic</td>
<td>$1.6479^n$</td>
<td>–</td>
</tr>
<tr>
<td>Recursive BSPI [GK17]</td>
<td>Deterministic</td>
<td>–</td>
<td>$k^{0.7207n}$</td>
</tr>
<tr>
<td>Recursive Simple PI [KMG16b]</td>
<td>Randomised</td>
<td>–</td>
<td>$(2 + \ln(k - 1))^n$</td>
</tr>
</tbody>
</table>
### Upper bounds on number of iterations

<table>
<thead>
<tr>
<th>PI Variant</th>
<th>Type</th>
<th>$k = 2$</th>
<th>General $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Howard’s (“all switch”) PI</td>
<td>Deterministic</td>
<td>$O\left(\frac{2^n}{n}\right)$</td>
<td>$O\left(\frac{k^n}{n}\right)$</td>
</tr>
<tr>
<td>[H60, MS99]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mansour and Singh’s Randomised PI</td>
<td>Randomised</td>
<td>$1.7172^n$</td>
<td>$\approx O\left(\left(\frac{k}{2}\right)^n\right)$</td>
</tr>
<tr>
<td>[MS99]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Batch-switching PI (BSPI)</td>
<td>Deterministic</td>
<td>$1.6479^n$</td>
<td>–</td>
</tr>
<tr>
<td>[KMG16a]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recursive BSPI</td>
<td>Deterministic</td>
<td>–</td>
<td>$k^{0.7207n}$</td>
</tr>
<tr>
<td>[GK17]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recursive Simple PI</td>
<td>Randomised</td>
<td>–</td>
<td>$(2 + \ln(k - 1))^n$</td>
</tr>
<tr>
<td>[KMG16b]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Lower bounds on number of iterations

$\Omega\left(\frac{2^n}{n}\right)$  
Howard’s PI on $n$-state MDPs with $\Theta(n)$ actions per state [F10, HGD12].
### Upper bounds on number of iterations

<table>
<thead>
<tr>
<th>PI Variant</th>
<th>Type</th>
<th>$k = 2$</th>
<th>General $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Howard’s (“all switch”) PI [H60, MS99]</td>
<td>Deterministic</td>
<td>$O\left( \frac{2^n}{n} \right)$</td>
<td>$O\left( \frac{k^n}{n} \right)$</td>
</tr>
<tr>
<td>Mansour and Singh’s Randomised PI [MS99]</td>
<td>Randomised</td>
<td>$1.7172^n$</td>
<td>$\approx O\left( \left( \frac{k}{2} \right)^n \right)$</td>
</tr>
<tr>
<td>Batch-switching PI (BSPI) [KMG16a]</td>
<td>Deterministic</td>
<td>$1.6479^n$</td>
<td>–</td>
</tr>
<tr>
<td>Recursive BSPI [GK17]</td>
<td>Deterministic</td>
<td>–</td>
<td>$k^{0.7207n}$</td>
</tr>
<tr>
<td>Recursive Simple PI [KMG16b]</td>
<td>Randomised</td>
<td>–</td>
<td>$(2 + \ln(k - 1))^n$</td>
</tr>
</tbody>
</table>

### Lower bounds on number of iterations

- $\Omega(2^{n/7})$ Howard’s PI on $n$-state MDPs with $\Theta(n)$ actions per state [F10, HGD12].
- $\Omega(2^{n/2})$ Simple PI on $n$-state, 2-action MDPs [MC94].
### Upper bounds on number of iterations

<table>
<thead>
<tr>
<th>PI Variant</th>
<th>Type</th>
<th>$k = 2$</th>
<th>General $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Howard’s (“all switch”) PI [H60, MS99]</td>
<td>Deterministic</td>
<td>$O \left( \frac{2^n}{n} \right)$</td>
<td>$O \left( \frac{k^n}{n} \right)$</td>
</tr>
<tr>
<td>Mansour and Singh’s Randomised PI [MS99]</td>
<td>Randomised</td>
<td>$1.7172^n$</td>
<td>$\approx O \left( \left( \frac{k}{2} \right)^n \right)$</td>
</tr>
<tr>
<td>Batch-switching PI (BSPI) [KMG16a]</td>
<td>Deterministic</td>
<td>$1.6479^n$</td>
<td>–</td>
</tr>
<tr>
<td>Recursive BSPI [GK17]</td>
<td>Deterministic</td>
<td>–</td>
<td>$k^{0.7207n}$</td>
</tr>
<tr>
<td>Recursive Simple PI [KMG16b]</td>
<td>Randomised</td>
<td>–</td>
<td>$(2 + \ln(k - 1))^n$</td>
</tr>
</tbody>
</table>

### Lower bounds on number of iterations

- $\Omega(2^{n/7})$  Howard’s PI on $n$-state MDPs with $\Theta(n)$ actions per state [F10, HGD12].
- $\Omega(2^{n/2})$  Simple PI on $n$-state, 2-action MDPs [MC94].
- $\Omega(n)$       Howard’s PI on $n$-state, 2-action MDPs [HZ10].
Open Problems

- Is the complexity of Howard’s PI on 2-action MDPs upper-bounded by the Fibonacci sequence ($\approx 1.6181^n$)?
Open Problems

- Is the complexity of Howard’s PI on 2-action MDPs upper-bounded by the Fibonacci sequence ($\approx 1.6181^n$)?

- Is Howard’s PI the most efficient among deterministic PI algorithms (worst case over all MDPs)?
Open Problems

- Is the complexity of Howard’s PI on 2-action MDPs upper-bounded by the Fibonacci sequence ($\approx 1.6181^n$)?

- Is Howard’s PI the most efficient among deterministic PI algorithms (worst case over all MDPs)?

- Is there a super-linear lower bound on the iterations taken by Howard’s PI on 2-action MDPs?
Open Problems

- Is the complexity of Howard’s PI on 2-action MDPs upper-bounded by the Fibonacci sequence ($\approx 1.6181^n$)?

- Is Howard’s PI the most efficient among deterministic PI algorithms (worst case over all MDPs)?

- Is there a super-linear lower bound on the iterations taken by Howard’s PI on 2-action MDPs?

- Is (Howard’s) PI strongly polynomial on deterministic MDPs?
Open Problems

- Is the complexity of Howard’s PI on 2-action MDPs upper-bounded by the Fibonacci sequence ($\approx 1.6181^n$)?

- Is Howard’s PI the most efficient among deterministic PI algorithms (worst case over all MDPs)?

- Is there a super-linear lower bound on the iterations taken by Howard’s PI on 2-action MDPs?

- Is (Howard’s) PI strongly polynomial on deterministic MDPs?

- Is there a strongly polynomial algorithm for MDP planning?
References and Additional Reading


References and Additional Reading


