Uninformed search

Christos Dimitrakakis

March 6, 2024

Outline

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y \rangle \in h} c(\langle x, y \rangle)$

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y \rangle \in h} c(\langle x, y \rangle)$
- We can equivalently consider state-action costs $c(s, a)$.

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y \rangle \in h} c(\langle x, y \rangle)$
- We can equivalently consider state-action costs $c(s, a)$.
- A policy π specifies a path $x_1, \ldots$ with $x_{k+1} = \tau(x_k, \pi(x_k))$

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y \rangle \in h} c(\langle x, y \rangle)$
- We can equivalently consider state-action costs $c(s, a)$.
- A policy π specifies a path $x_1, \ldots$ with $x_{k+1} = \tau(x_k, \pi(x_k))$
- Following a policy π from state $x_1 = x$ has a total cost $C^\pi(x_1) = \sum_{k=1}^{t} c(x_k, \pi(x_k))$.

The shortest path problem

- Traversing arc $\langle x, y\rangle$ incurs costs $c(\langle x, y\rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y\rangle \in h} c(\langle x, y\rangle)$
- We can equivalently consider state-action costs $c(s, a)$.
- A policy π specifies a path $x_1, \ldots$ with $x_{k+1} = \tau(x_k, \pi(x_k))$
- Following a policy π from state $x_1 = x$ has a total cost $C^\pi(x_1) = \sum_{k=1}^{t} c(x_k, \pi(x_k))$.

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y \rangle \in h} c(\langle x, y \rangle)$
- We can equivalently consider state-action costs $c(s, a)$.
- A policy π specifies a path $x_1, \ldots$ with $x_{k+1} = \tau(x_k, \pi(x_k))$
- Following a policy π from state $x_1 = x$ has a total cost $C^\pi(x_1) = \sum_{k=1}^{t} c(x_k, \pi(x_k))$.

The shortest path problem

- Input: start nodes X and goal nodes Y and edge costs $c : A \to \mathbb{R}$.
- Output: Find a path h from X to Y so that $C(h) \leq C(h')$ for all h'

The shortest path problem

- Traversing arc $\langle x, y \rangle$ incurs costs $c(\langle x, y \rangle)$
- Following a path h has a total cost $C(h) = \sum_{\langle x,y \rangle \in h} c(\langle x, y \rangle)$
- We can equivalently consider state-action costs $c(s, a)$.
- A policy π specifies a path $x_1, \ldots$ with $x_{k+1} = \tau(x_k, \pi(x_k))$
- Following a policy π from state $x_1 = x$ has a total cost $C^\pi(x_1) = \sum_{k=1}^{t} c(x_k, \pi(x_k))$.

The shortest path problem

- Input: start nodes X and goal nodes Y and edge costs $c : A \to \mathbb{R}$.
- Output: Find a path h from X to Y so that $C(h) \leq C(h')$ for all h'

Notes

- If the path/policy does not reach a goal, the cost is infinite.
- We can maximise rewards instead of minimising costs.

Formalising the shortest path problem

The cost from state x of a policy that reaches a goal is

$$C^\pi(s) \triangleq \sum_{i=1}^{\infty} c[s_t, \pi(s_t)], \qquad s_{t+1} = \tau[s_t, \pi(s_t)], \quad s_1 = s$$

where for every $s \in Y$, $c(s, a) = 0$ and $\tau(s, a) = s$ for all actions.

- We can calculate this recursively (from the goal state)

$$\begin{aligned} C^\pi(s) &= \sum_{i=1}^{\infty} c[s_t, \pi(s_t)] && (1) \\ &= c[s, \pi(s)] + \sum_{i=2}^{\infty} c[s_t, \pi(s_t)] && (2) \\ &= c[s, \pi(s)] + C^\pi\{\tau[s, \pi(s)]\}. && (3) \end{aligned}$$

- The same idea applies for the shortest path

$$C^*(s) \triangleq \min_\pi C^\pi(s) = \min_a \{c[s, a] + C^*[\tau(s, a)]\}. \qquad (4)$$

The shortest path algorithm: backward search

Shortest path algorithm

Input: Goal states Y, starting state x.
Set $C(s) = 0$ for all states $s \in Y$, $F_0 = Y$.
for $t = 0, 1, \ldots$ **do**
  for $s' \in F_t$ **do**
    $\pi(s) = \arg\min_a c(s, a) + C(\tau(s, a))$
    $C(s) = \min_a c(s, a) + C(\tau(s, a))$
  end for
  $F_{t+1} = \texttt{parent}(F_t)$.
  if $F_{t+1} = \emptyset$ or $x \in F_t$ **then**
    return π, C
  end if
end for

Algorithm idea

- Start from goal states
- Go back one step each time, adding the cost.
- Stop whenever there are no more states to go back to, or if we reach the start state.

Optimality proof

Theorem

$C(s) = C^*(s)$

Proof

- If $s \in Y$, then $C(s) = 0 = C^*(s)$.
- For any other $s', s = \texttt{parent}(s')$: we will show that: if $C(s') \leq C^*(s')$ then $C(s) \leq C^*(s)$.

$$
\begin{aligned}
C(s) &= \min_a \{c(s,a) + C(\tau(s,a))\} && \text{(by definition)} \\
&\leq \min_a \{c(s,a) + C^*(\tau(s,a))\} && \text{(by induction)} \\
&\leq \min_a \left\{c(s,a) + C^{\pi'}(\tau(s,a))\right\}, \qquad \forall \pi' && \text{(by optimality)} \\
&\leq C^{\pi}(s), \qquad \forall \pi.
\end{aligned}
$$

For the optimal policy π^*, $C^{\pi^*}(s) = C^*(s)$, so $C(s) \leq C^*(s)$. Finally,

$$C^*(s) \leq C^{\pi}(s) = C(s) \geq C^*(s),$$

since $C^{\pi}(s) = C(s)$ for the policy returned by the algorithm.