CS 4700: Foundations of Artificial Intelligence

Spring 2020
Prof. Haym Hirsh

Lecture 23
April 8, 2020

ONLINE

Backup plans:

If this Zoom meeting ends prematurely,
five-minute break, check Piazza

Other announcements:

Please fill in survey on Canvas

Other announcements:

First quiz: Thu Apr 9 12:00pm

Topic: Uninformed search

24 hour window for submission

Further details out shortly after lecture

Next Karma Lecture

Thursday, at 11:40am
https://cornell.zoom.us/j/276979194

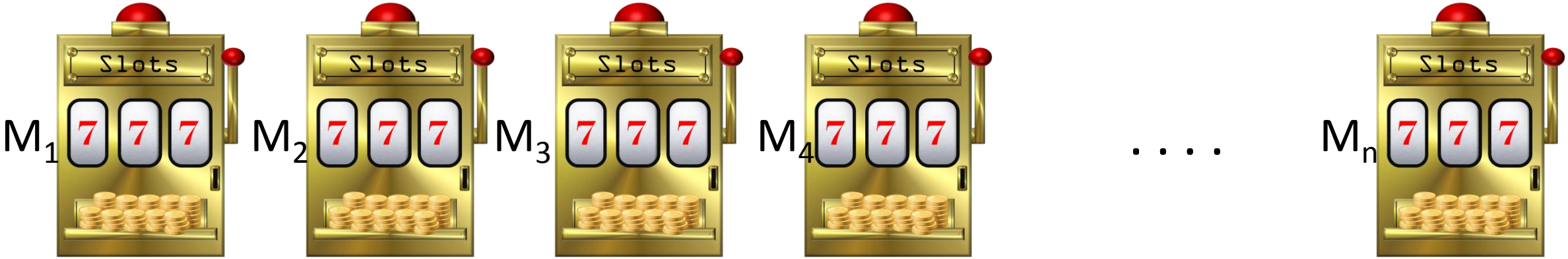
"Efficient Machine Learning via Data Summarization"

Baharan Mirzasoleiman, Stanford University

Multi-Armed Bandit

What strategy do I use to pick a sequence of a_i?

a_1 a_2 a_3 a_4 a_n



R_1 R_2 R_3 R_4 R_n

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

- Pick the arm with largest UCB(M_i) instead of $\hat{R}_i$

$$\text{UCB}(M_i) = \hat{R}_i + \frac{g(N)}{\sqrt{N_i}}$$

where

$\hat{R}_i$ = average reward for i so far

N = total number of pulls made so far

N_i = total number of pulls of M_i so far

$g(N) = c\sqrt{\ln N}$ $\qquad$ $g(N) = \sqrt{2\ \log(1 + N \log^2 N)}$

g(N) should go up more slowly than $\sqrt{N_i}$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

- Pick the arm with largest UCB(M_i) instead of $\hat{R}_i$

$$\text{UCB}(M_i) = \hat{R}_i + \frac{g(N)}{\sqrt{N_i}}$$

where

$\hat{R}_i$ = average reward for i so far = $\frac{\text{Sum}_i}{N_i}$

N = total number of pulls made so far

N_i = total number of pulls of M_i so far

$g(N) = c\sqrt{\ln N}$ $\qquad$ $g(N) = \sqrt{2\ \log(1 + N \log^2 N)}$

g(N) should go up more slowly than $\sqrt{N_i}$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Algorithm:

For i ← 1 to n { $\mathrm{Sum}_i \leftarrow R(\mathrm{arm}_i)$; $N_i \leftarrow 1$ }; $N \leftarrow n$ /* Initialization */

Loop Forever

best ← $\underset{1 \le i \le n}{\mathrm{argmax}} \left[\frac{\mathrm{Sum}_i}{N_i} + \frac{g(N)}{\sqrt{N_i}} \right]$

pull arm a_{best} and get reward r

$\mathrm{Sum}_{best} \leftarrow \mathrm{Sum}_{best} + r$; $N_{best} \leftarrow N_{best} + 1$; $N \leftarrow N+1$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Algorithm:

For i ← 1 to n { $\mathrm{Sum}_i \leftarrow R(\mathrm{arm}_i)$; $N_i \leftarrow 1$ }; $N \leftarrow n$ /* Initialization */

Loop Forever

$$\mathrm{best} \leftarrow \underset{1 \le i \le n}{\mathrm{argmax}} \left[\frac{\mathrm{Sum}_i}{N_i} + \frac{g(N)}{\sqrt{N_i}} \right]$$

pull arm a_{best} and get reward r

$\mathrm{Sum}_{best} \leftarrow \mathrm{Sum}_{best} + r$; $N_{best} \leftarrow N_{best} + 1$; $N \leftarrow N+1$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Algorithm:

For i ← 1 to n { $\mathrm{Sum}_i \leftarrow R(\mathrm{arm}_i)$; $N_i \leftarrow 1$ }; $N \leftarrow n$ /* Initialization */

Loop Forever

$$\text{best} \leftarrow \underset{1 \le i \le n}{\operatorname{argmax}} \left[\frac{\mathrm{Sum}_i}{N_i} + \frac{g(N)}{\sqrt{N_i}} \right] \quad \hat{R}_i$$

pull arm a_{best} and get reward r

$\mathrm{Sum}_{best} \leftarrow \mathrm{Sum}_{best} + r$; $N_{best} \leftarrow N_{best} + 1$; $N \leftarrow N+1$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Algorithm:

Pull each arm once

For i ← 1 to n { $\mathrm{Sum}_i \leftarrow R(\mathrm{arm}_i)$; $N_i \leftarrow 1$ }; $N \leftarrow n$ /* Initialization */

Loop Forever

best ← $\underset{1 \le i \le n}{\mathrm{argmax}} \left[\frac{\mathrm{Sum}_i}{N_i} + \frac{g(N)}{\sqrt{N_i}} \right]$ $\hat{R}_i$

pull arm a_{best} and get reward r

$\mathrm{Sum}_{best} \leftarrow \mathrm{Sum}_{best} + r$; $N_{best} \leftarrow N_{best} + 1$; $N \leftarrow N+1$

Multi-Armed Bandits:
Upper Confidence Bound (UCB) Heuristic

Reminder: this is a form of MDP

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Reminder: this is a form of MDP

$$\Pi^{UCB}(s) = a_{\underset{1 \le i \le n}{\mathrm{argmax}}} \left[\frac{\mathrm{Sum}_i}{N_i} + \frac{g(N)}{\sqrt{N_i}} \right]$$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Instead of cumulative discounted reward, average reward for N steps

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Instead of cumulative discounted reward, average reward for N steps

Expected average reward for a policy Π: $\mu_{\mathrm{N}}^{\Pi} = \mathrm{E}_{\Pi}\left[\frac{\sum_{i=1}^{\mathrm{n}} \mathrm{Sum}_i}{\mathrm{N}}\right]$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Instead of cumulative discounted reward, average reward for N steps

Expected average reward for a policy Π: $\mu_{\mathrm{N}}^{\Pi} = \mathrm{E}_{\Pi}\left[\frac{\sum_{i=1}^{\mathrm{n}} \mathrm{Sum}_i}{\mathrm{N}}\right]$

Expected average reward for always picking optimal arm: $\mu^{\mathrm{best}} = \max_i \mathrm{R}_i$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Expected average reward for a policy Π: μ_{N}^{Π}

Expected average reward for optimal arm: μ^{best}

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}}$ - $N\mu_{\mathrm{N}}^{\Pi}$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Expected average reward for a policy Π: μ_{N}^{Π}

Expected average reward for optimal arm: μ^{best}

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}}$ - $N\mu_{\mathrm{N}}^{\Pi}$

How much exploration costs you

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}}$ - $N\mu_{\mathrm{N}}^{\Pi}$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}}$ - $N\mu_{\mathrm{N}}^{\Pi}$

Known Result: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = \Omega(\log(\mathrm{N}))$

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log\left(1 + \mathrm{N}\log^2 \mathrm{N}\right)}$ or c$\sqrt{\ln(\mathrm{N})}$?

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}}$ - $N\mu_{\mathrm{N}}^{\Pi}$

Known Result: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = \Omega(\log(\mathrm{N}))$

Your expected regret will grow at least logarithmically with N

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log(1 + \mathrm{N}\log^2 \mathrm{N})}$ or c$\sqrt{\ln(\mathrm{N})}$?

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}} - N\mu_{\mathrm{N}}^{\Pi}$

Known Result: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = \Omega(\log(\mathrm{N}))$

$\mathrm{regret}_{\mathrm{N}}^{\mathrm{UCB}(g(\mathrm{N}))} = O(\log(\mathrm{N}))$

Your expected regret will grow at least logarithmically with N

UCB with these g(N) functions has regret that grows at worst logarithmically with N

Multi-Armed Bandits: Upper Confidence Bound (UCB) Heuristic

Why g(N) = $\sqrt{2\ \log(1 + \mathrm{N}\log^2 \mathrm{N})}$ or c$\sqrt{\ln(\mathrm{N})}$?

Regret for a policy: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = N\mu^{\mathrm{best}} - N\mu_{\mathrm{N}}^{\Pi}$

Known Result: $\mathrm{regret}_{\mathrm{N}}^{\Pi} = \Omega(\log(\mathrm{N}))$

$\mathrm{regret}_{\mathrm{N}}^{\mathrm{UCB}(g(\mathrm{N}))} = O(\log(\mathrm{N}))$

Your expected regret will grow at least logarithmically with N

UCB with these g(N) functions has regret that grows at worst logarithmically with N

They are "optimal"

Monte Carlo Tree Search (MCTS)

Application of multi-armed bandits

Monte Carlo Tree Search (MCTS)
Application of multi-armed bandits

(Section 5.4)

Timeline of Key Ideas in Game Tree Search

1948	Alan Turing	Look ahead and use an evaluation function
1950	Claude Shannon	Game tree search
1956	John McCarthy	Alpha-beta pruning
1959	Arthur Samuel	Learn evaluation function (Reinforcement learning)

…

1997: Deep Blue defeats Gary Kasparov (3½–2½)

Game tree search with alpha-beta pruning
plus lots of enhancements

Go?

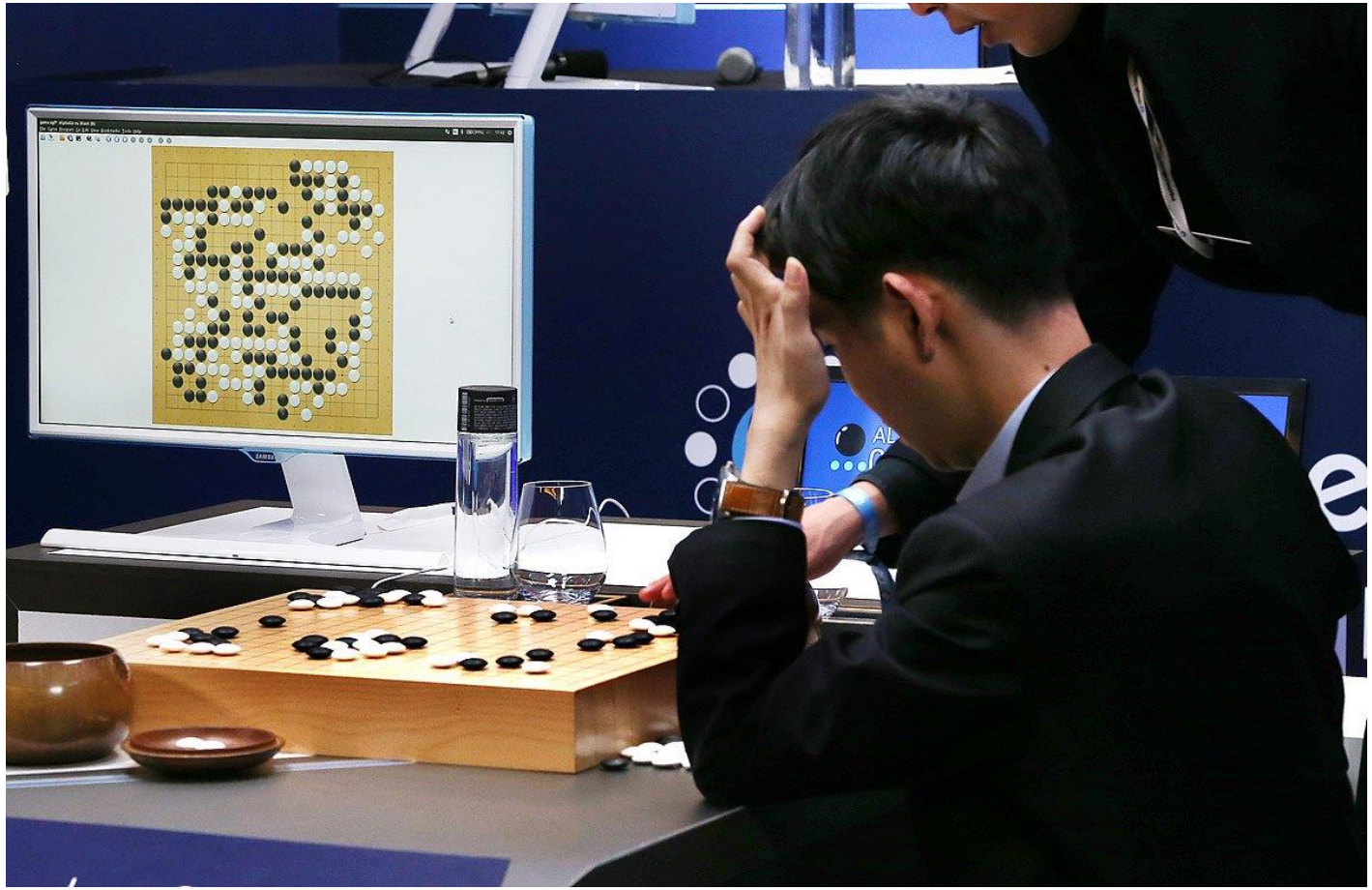
Branching factor is in the 100s

Evaluation function is difficult because payoff may be very far away

Needed new ideas

1983	Bruce Ballard	Lookahead for probabilistic moves
1987	Bruce Abramson	Evaluation by expected outcome (repeated simulation)
1992	Gerald Tesauro	TD-Gammon (reinforcement learning, self-play)
1992	Bernd Brugmann	Monte Carlo Go (simulated annealing)
1999	U of Alberta	Simulation in Poker
1999	Matt Ginsberg	Simulation in Bridge
2002	Brian Sheppard	Simulation in Scrabble
2006	Levente Kocsis and Csaba Szepesvár	Multi-armed bandits for Monte-Carlo tree search

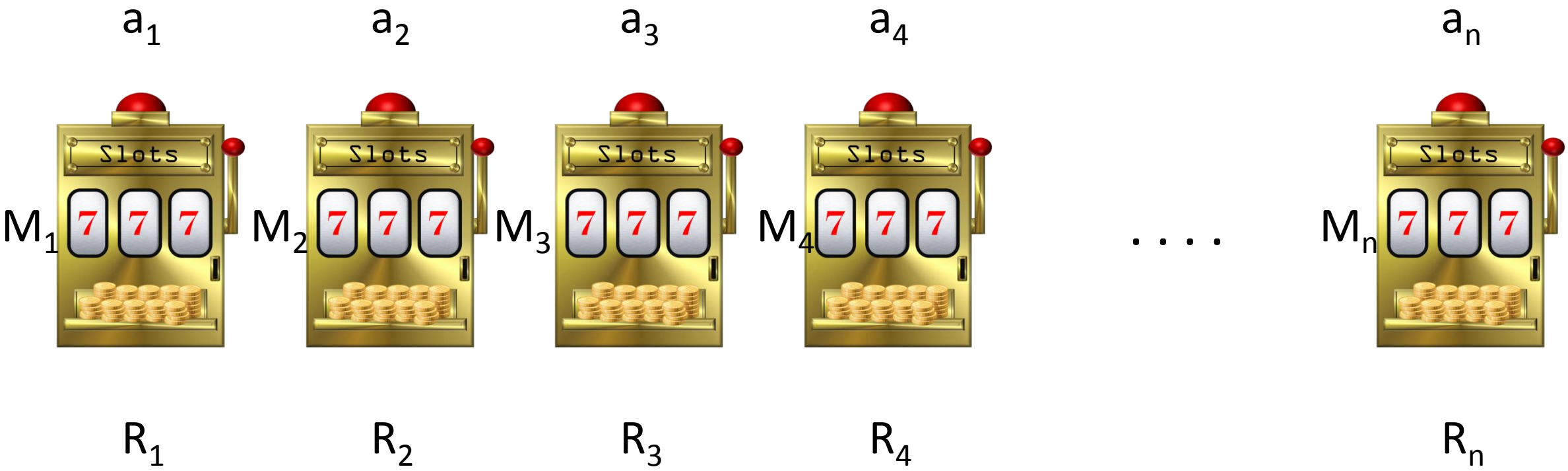
2016: AlphaGo defeats Lee Sedol (4-1)



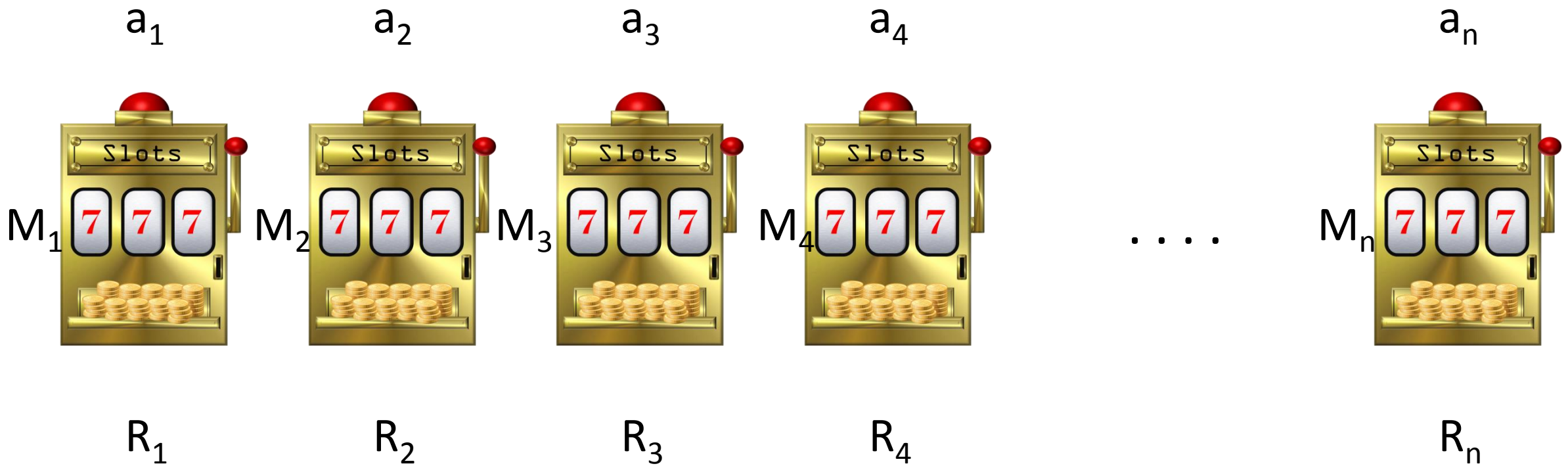
Key ideas of Monte Carlo Tree Search:

1. View move selection as a multi-armed bandit problem

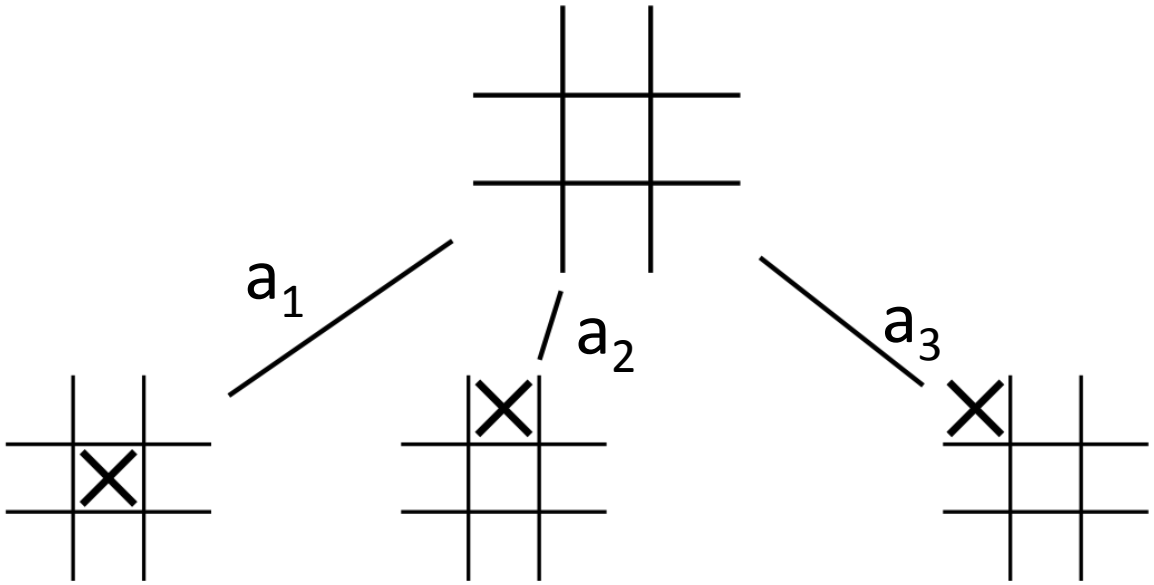
Multi-Armed Bandit



Multi-Armed Bandit for Game Tree Search



Multi-Armed Bandit for Game Tree Search

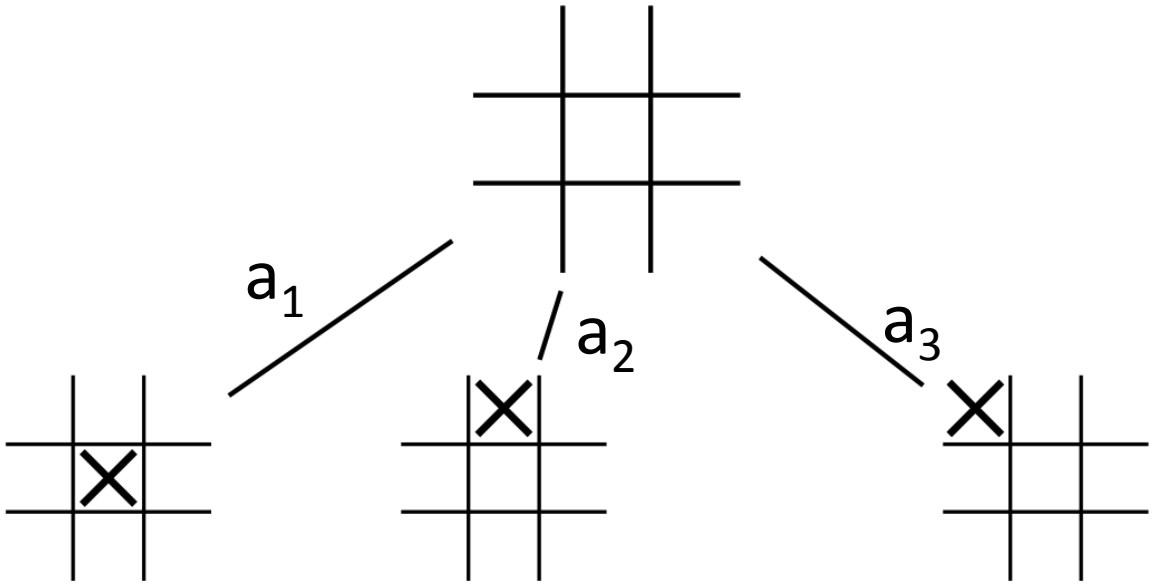


What move should I try?

Key ideas of Monte Carlo Tree Search:

1. View move selection as a multi-armed bandit problem
2. Evaluate moves by simulating games

Multi-Armed Bandit for Game Tree Search



What move should I try on each simulated game?