

Multi-Armed Bandits

Aditya Mahajan Ravi Agarwal

EECS 558: Stochastic Control

December 9, 2003

Outline

Multi Armed Bandits

- Bandit Process – Gittin's Index
- Multi Armed Bandit Problem

Forwards Induction

- Why does it work?
- Queuing Models

Multiple Plays

- Sufficiency Conditions
- Concave Envelopes

Bandit Processes

Model : Single Armed Bandits

- A Markov decision process
- Countable State Space \mathcal{S}
- Action Space $\mathcal{U} = \{0,1\}$
 - Freezing Control and Continuation Control
- Time Invariant Rewards $R_t(X(t), U(t)) = R(X(t), U(t))$
 $R(i,0) = 0 \quad R(i,1) = R(i)$

Expected Rewards

Expected Discounted Reward

Total Expected Discounted Reward under any policy $\pi \in \mathcal{G}$

$$V^\pi(i) = \mathbf{E}^\pi \left\{ \sum_{t=0}^{\infty} \beta^t R(X(t), U(t)) \mid X(0) = i \right\}$$

Deterministic Stationary Markov Policies

- Divide the state \mathcal{S} into **Stopping Set** \mathcal{S}_0 and **Continuation Set** \mathcal{S}_1
- **Stopping Rule** — Random stopping time τ

Gittin's Index

The Expected Reward per unit Discounted Time under τ

$$\nu(i, \tau) = \frac{\mathbf{E} \left\{ \sum_{t=0}^{\tau-1} \beta^t R(X(t)) \mid X(0) = i \right\}}{\mathbf{E} \left\{ \sum_{t=0}^{\tau-1} \beta^t \mid X(0) = i \right\}}$$

Gittin's Index

The Gittin's Index is

$$\nu(i) \triangleq \sup_{\tau > 0} \nu(i, \tau)$$

Supremum achieved for

$$\tau(i) = \min \{ t : \nu(X(t)) < \nu(i) \}$$

Gittin's Index

The Expected Reward per unit Discounted Time under τ

$$\nu(i, \tau) = \frac{\mathbf{E} \left\{ \sum_{t=0}^{\tau-1} \beta^t R(X(t)) \mid X(0) = i \right\}}{\mathbf{E} \left\{ \sum_{t=0}^{\tau-1} \beta^t \mid X(0) = i \right\}}$$

Gittin's Index

The **Gittin's Index** is

$$\nu(i) \triangleq \sup_{\tau > 0} \nu(i, \tau)$$

Supremum achieved for

$$\tau(i) = \min \{ t : \nu(X(t)) < \nu(i) \}$$

Gittin's Index

The Expected Reward per unit Discounted Time under τ

$$\nu(i, \tau) = \frac{\mathbf{E} \left\{ \sum_{t=0}^{\tau-1} \beta^t R(X(t)) \mid X(0) = i \right\}}{\mathbf{E} \left\{ \sum_{t=0}^{\tau-1} \beta^t \mid X(0) = i \right\}}$$

Gittin's Index

The **Gittin's Index** is

$$\nu(i) \triangleq \sup_{\tau > 0} \nu(i, \tau)$$

Supremum achieved for

$$\tau(i) = \min \{ t : \nu(X(t)) < \nu(i) \}$$

Multi Armed Bandits

Model

- Family of N alternative bandit processes
- Countable State Space
State $\mathbf{X}(t) = (X_1(t), X_2(t), \dots, X_N(t))$
- Action Space $\mathcal{U} = \{1, 2, \dots, N\}$
- States of all other processes remain frozen
- Total Reward at time t , $\tilde{R}(t) = R_{n(t)}(X_{n(t)}(t)) = R_n(i)$

Objective

Find π to Maximize Total Expected Discounted Reward

$$V^\pi(\mathbf{i}) = \mathbf{E}^\pi \left\{ \sum_{t=0}^{\infty} \beta^t \tilde{R}(t) \mid \mathbf{X}(0) = \mathbf{i} \right\}$$

Multi Armed Bandits

Model

- Family of N alternative bandit processes
- Countable State Space
State $\mathbf{X}(t) = (X_1(t), X_2(t), \dots, X_N(t))$
- Action Space $\mathcal{U} = \{1, 2, \dots, N\}$
- States of all other processes remain frozen
- Total Reward at time t , $\tilde{R}(t) = R_{n(t)}(X_{n(t)}(t)) = R_n(i)$

Objective

Find π to Maximize Total Expected Discounted Reward

$$V^\pi(\mathbf{i}) = \mathbf{E}^\pi \left\{ \sum_{t=0}^{\infty} \beta^t \tilde{R}(t) \mid \mathbf{X}(0) = \mathbf{i} \right\}$$

Optimal Policy

Optimal Policy

From Dynamic Programming, the solution is given by

$$\mathbf{V}^\pi = \mathbf{R}^\pi + \beta \mathbf{P}^\pi \mathbf{V}^\pi$$

Gittin's Index Rule

At each time choose process with the highest Gittin's Index for continuation

$$u(t) = \arg \max_{n \in \mathcal{U}} \nu_n(i_n)$$

► Details

Advantage

- Index rule does not depend on state of other processes
- Forwards Induction Policy
 - Computationally less intensive than backwards induction

Optimal Policy

Optimal Policy

From Dynamic Programming, the solution is given by

$$\mathbf{V}^\pi = \mathbf{R}^\pi + \beta \mathbf{P}^\pi \mathbf{V}^\pi$$

Gittin's Index Rule

At each time choose process with the highest Gittin's Index for continuation

$$u(t) = \arg \max_{n \in \mathcal{U}} \nu_n(i_n)$$

► Details

Advantage

- Index rule does not depend on state of other processes
- Forwards Induction Policy
 - Computationally less intensive than backwards induction

Optimal Policy

Optimal Policy

From Dynamic Programming, the solution is given by

$$\mathbf{V}^\pi = \mathbf{R}^\pi + \beta \mathbf{P}^\pi \mathbf{V}^\pi$$

Gittin's Index Rule

At each time choose process with the highest Gittin's Index for continuation

$$u(t) = \arg \max_{n \in \mathcal{U}} \nu_n(i_n)$$

► Details

Advantage

- Index rule does not depend on state of other processes
- Forwards Induction Policy
Computationally less intensive than backwards induction

Backwards Induction vs. Forwards Induction

Backwards Induction

- Works on **the Principle of Optimality**
- If optimal policy from certain stage (or time) is known, we can extend this policy to an optimal policy starting one stage earlier

Forwards Induction

- Randomized n -step look-ahead policy
- For a given policy, maximize over all stopping times τ
- Second maximization over all policies

Backwards Induction vs. Forwards Induction

Backwards Induction

- Works on **the Principle of Optimality**
- If optimal policy from certain stage (or time) is known, we can extend this policy to an optimal policy starting one stage earlier

Forwards Induction

- Randomized **s -step look ahead policy**
- For a given policy, maximize over all stopping times τ .
- Second maximization over all policies

Why Forwards Induction works

Does Forwards Induction always work?

- Not optimal for all Markov decision processes
- For some policies, lower initial reward can be compensated by higher gains later on

Why Forwards Induction works in this case?

- When playing one arm, other arms remain frozen
- Decisions at each stage are not irrevocable
- There is no later compensation for not choosing the arm having highest rate of reward

Why Forwards Induction works

Does Forwards Induction always work?

- Not optimal for all Markov decision processes
- For some policies, lower initial reward can be compensated by higher gains later on

Why Forwards Induction works in this case?

- When playing one arm, other arms remain frozen
- Decisions at each stage are not irrevocable
- There is no later compensation for not choosing the arm having highest rate of reward

Why Forwards Induction works

Does Forwards Induction always work?

- Not optimal for all Markov decision processes
- For some policies, lower initial reward can be compensated by higher gains later on

Why Forwards Induction works in this case?

- When playing one arm, other arms remain frozen
- **Decisions at each stage are not irrevocable**
- There is no later compensation for not choosing the arm having highest rate of reward

Why Forwards Induction works

Does Forwards Induction always work?

- Not optimal for all Markov decision processes
- For some policies, lower initial reward can be compensated by higher gains later on

Why Forwards Induction works in this case?

- When playing one arm, other arms remain frozen
- **Decisions at each stage are not irrevocable**
- There is no later compensation for not choosing the arm having highest rate of reward

Multi-armed Bandit Process in Queueing

Resource Allocation Problem

- N Parallel Queues, with holding cost but no arrivals
- Single Server, pre-emptive memoryless service
- Minimize total expected discounted holding cost

Variations

- Arrivals
- Switching Costs
- Connectivity of Servers and Queues
- Multiple Servers

Multi-armed Bandit Process in Queueing

Resource Allocation Problem

- N Parallel Queues, with holding cost but no arrivals
- Single Server, pre-emptive memoryless service
- Minimize total expected discounted holding cost

Variations

- Arrivals
- Switching Costs
- Connectivity of Servers and Queues
- Multiple Servers

Multiple Armed Bandits with Multiple Plays

Model

- Family of N alternative bandit processes
- Countable State Space
State $\mathbf{X}(t) = (X_1(t), X_2(t), \dots, X_N(t))$
- M arms chosen to be played at each time
- Total Reward at time t , $\tilde{R}(t)$ is the sum of the individual rewards

Objective

Find π to Maximize Total Expected Discounted Reward

$$V^\pi(\mathbf{i}) = \mathbf{E}^\pi \left\{ \sum_{t=0}^{\infty} \beta^t \tilde{R}(t) \mid \mathbf{X}(0) = \mathbf{i} \right\}$$

Multiple Armed Bandits with Multiple Plays

Model

- Family of N alternative bandit processes
- Countable State Space
State $\mathbf{X}(t) = (X_1(t), X_2(t), \dots, X_N(t))$
- M arms chosen to be played at each time
- Total Reward at time t , $\tilde{R}(t)$ is the sum of the individual rewards

Objective

Find π to Maximize Total Expected Discounted Reward

$$V^\pi(\mathbf{i}) = \mathbf{E}^\pi \left\{ \sum_{t=0}^{\infty} \beta^t \tilde{R}(t) \mid \mathbf{X}(0) = \mathbf{i} \right\}$$

Optimality of Gittin's Index

Is Gittin's Index Optimal?

- Decisions not irrevocable
- In general, Gittin's Index not optimal (Ishikida (1992))

Sufficiency Condition for Optimality of Gittin's Index

Pandelis and Teneketzis (1999) For each realization ω , for any $i \neq j$ and k, l such that $v_i(\tau_i(k, \omega), \omega) > v_j(\tau_j(l, \omega), \omega)$ we have

$$v_i(\tau_i(k, \omega), \omega)(1 - \beta) \geq v_j(\tau_j(l, \omega), \omega)$$

Optimality of Gittin's Index

Is Gittin's Index Optimal?

- Decisions not irrevocable
- In general, Gittin's Index not optimal (Ishikida (1992))

Sufficiency Condition for Optimality of Gittin's Index

Pandelis and Teneketzis (1999) For each realization ω , for any $i \neq j$ and k, l such that $v_i(\tau_i(k, \omega), \omega) > v_j(\tau_j(l, \omega), \omega)$ we have

$$v_i(\tau_i(k, \omega), \omega)(1 - \beta) \geq v_j(\tau_j(l, \omega), \omega)$$

Concave Envelopes

The concave envelope $\{\bar{X}_i(l)\}_{l=0}^{\infty}$ of $\{X_i(l)\}_{l=0}^{\infty}$ is given by

$$\bar{Z}_i(l) = \min_{s \leq l} \nu_i(s), \quad l = 0, 1, 2, \dots$$

Properties

- The reward obtained under any policy π is not decreased when the original reward processes are replaced with their concave envelopes.
- The reward obtained under Gittin's Index Rule does not change when the reward processes are replaced by their concave envelopes.

Concave Envelopes

The concave envelope $\{\bar{X}_i(l)\}_{l=0}^{\infty}$ of $\{X_i(l)\}_{l=0}^{\infty}$ is given by

$$\bar{Z}_i(l) = \min_{s \leq l} \nu_i(s), \quad l = 0, 1, 2, \dots$$

Properties

- The reward obtained under any policy π is not decreased when the original reward processes are replaced with their complex envelopes.
- The reward obtained under **Gittin's Index Rule** does not change when the reward processes are replaced by their concave envelopes.

Concave Envelopes

The concave envelope $\{\bar{X}_i(l)\}_{l=0}^{\infty}$ of $\{X_i(l)\}_{l=0}^{\infty}$ is given by

$$\bar{Z}_i(l) = \min_{s \leq l} \nu_i(s), \quad l = 0, 1, 2, \dots$$

Properties

- The reward obtained under any policy π is not decreased when the original reward processes are replaced with their complex envelopes.
- The reward obtained under [Gittin's Index Rule](#) does not change when the reward processes are replaced by their concave envelopes.

Solving Gittin's Index for a Single Arm

Standard Arm

- Bandit Process X
- Standard Process with fixed reward γ

$$V(i) = \max \left\{ R(i) + \beta \sum P_{i,j} V(j), \gamma + \beta V(i) \right\}$$

Retirement Option

- Bandit Process X
- Option of retiring with a fixed reward M

$$(1) \quad V(i, M) = \max \left\{ R(i) + \beta \sum P_{i,j} V(j, M), M \right\}$$

Solution of Retirement Option

Optimal Policies

- $V(i, M)$ is a non-decreasing convex function of M
- Optimal Policies partition \mathcal{S} into three sets

Strict Continuation Set $\mathcal{C}_M = \{i : V(i, M) > M\}$

Strict Stopping Set $\mathcal{S}_M = \{i : M > R(i) + \beta \sum P_{i,j} V(j, M)\}$

Indifferent States $\mathcal{I}_M = \{i : M = R(i) + \beta \sum P_{i,j} V(j, M)\}$

- Optimal Value Function

$$V(i, M) = \mathbf{E} \left\{ \sum_{t=0}^{\tau(i, M)-1} \beta^t R(X(t)) + \beta^{\tau(i, M)} M \mid X(0) = i \right\}$$

Gittin's Index

Relation between two problems

$$M(i) = \sup\{M : i \in \mathcal{C}_M\} = \inf\{M : V(i, M) = M\}$$

$$\gamma(i) = (1 - \beta)M(i)$$

$$\nu(i) = \gamma(i)$$

◀ Gittin's Index