Restless Bandits, Weakly Coupled MDPs and LP Relaxations

Nicolas Gast

joint work with our two students Kimang Khun, Chen Yan,
co-supervised with Bruno Gaujal

Inria

Data-driven Queueing Challenges – September, 2022

Motivation problem 1: Applicant screening problem

N applicants, T rounds of interview.
Each round: you can interview up to αN candidates.
Goal: maximize the expected quality of candidates.

Motivation problem 1: Applicant screening problem

N applicants, T rounds of interview.
Each round: you can interview up to αN candidates.
Goal: maximize the expected quality of candidates.

Each candidate has an (unknown) quality p_n.

- Result of an interview: Bernoulli(p_n)

Goal: find the βN highest p_n.

Possible heuristics:

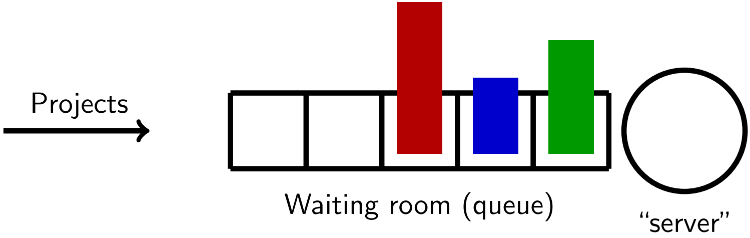
- Greedy (exploitation)?
- Random (exploration)?

Motivation problem 2: What to work on?

Job Scheduling

Motivation problem 2: What to work on?

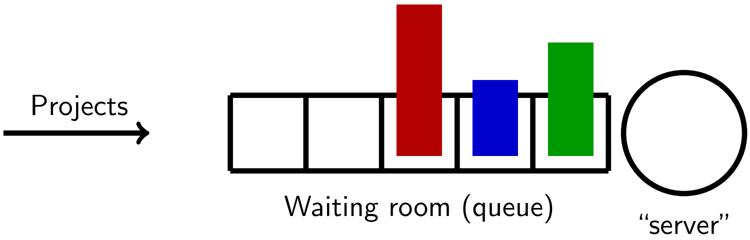
Job Scheduling



- Examples: research projects, tasks allocations, electric vehicle charging, wireless scheduling,...
- We allow preemption (preempt-resume).

Motivation problem 2: What to work on?

Job Scheduling



- Examples: research projects, tasks allocations, electric vehicle charging, wireless scheduling,...
- We allow preemption (preempt-resume).

Possible heuristic: SRPT ((Shortest Remaining Processing Time).)

- "Strongly optimal" [Schrage, 1966] if you know the project durations and you want to minimize the waiting time

Can we do better?

These two problems are restless bandit problems.

- PSPACE hard in general.
- Greedy, random, SRPT,... are in general not optimal.

Can we do better?

These two problems are restless bandit problems.

- PSPACE hard in general.
- Greedy, random, SRPT,... are in general not optimal.

We construct policies by assuming independence.

- Asymptotically optimal policies.
- LP-based (= computationally efficient)

Outline

Outline

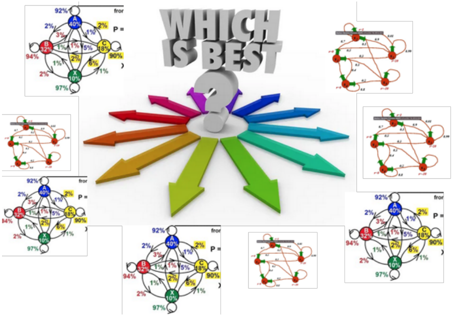
Finite-horizon restless bandits

Subsidy, infinite-horizon and index policies

How to Compute Indices: A Sub-Cubic Algorithm

Conclusion

Reslless Markov bandit problem



A decision maker faces N arms.
Time is discrete.

- Each arm is a 2-action MDP (passive / active)
- Controller can *activate* $\alpha N < N$ arms each time.

Policy : you observe the states. Which ones do you activate?

For simplicity, we consider statistically identical bandits

An arm is a 2-action MDP: if in state s:

- Activation: earn $r(s, active)$, jump to $s' \sim \mathbf{P}\left[s' \mid s, active\right]$.
- Passive: earn $r(s, passive)$, jump to $s' \sim \mathbf{P}\left[s' \mid s, passive\right]$.

Transition are independent *given the actions*.

For simplicity, we consider statistically identical bandits

An arm is a 2-action MDP: if in state s:

- Activation: earn $r(s, active)$, jump to $s' \sim \mathbf{P}\left[s' \mid s, active\right]$.
- Passive: earn $r(s, passive)$, jump to $s' \sim \mathbf{P}\left[s' \mid s, passive\right]$.

Transition are independent *given the actions*.

The N arms are *statistically identical* and we denote by:

$$X_s^{(N)}(t) = \frac{1}{N}\{\#\text{ arms in state } s \text{ at time } t\}.$$

$$Y_{s,a}^{(N)}(t) = \frac{1}{N}\{\#\text{arms in state } s \text{ for which action } a \text{ is taken at } t\}.$$

Restless bandits are difficult to solve

A admissible policy is a sequence of functions $\pi_t : \mathcal{X} \to \mathcal{Y}$ such that $Y^{(N)}(t) = \pi_t(X^{(N)}(t))$ is feasible with respect to $X^{(N)}(t)$, and

$$\sum_s a Y_{s,a}^{(N)}(t) \leq \alpha \qquad\qquad (\textit{activation constraint})$$

Restless bandits are difficult to solve

A admissible policy is a sequence of functions $\pi_t : \mathcal{X} \to \mathcal{Y}$ such that $Y^{(N)}(t) = \pi_t(X^{(N)}(t))$ is feasible with respect to $X^{(N)}(t)$, and

$$\sum_s aY^{(N)}_{s,a}(t) \leq \alpha \qquad\qquad (\textit{activation constraint})$$

Theorem (Papadimitriou, Tsitsiklis 1994)

Finding the best admissible policy is PSPACE-complete.

(harder than NP-hard).

The LP-relaxation

Replace the activation constraint $\sum_s a Y_{s,a}^{(N)}(t) \leq \alpha$ by

$$\sum_s a \mathbb{E}\left[Y_{s,a}^{(N)}(t)\right] \leq \alpha.$$

The LP-relaxation

Replace the activation constraint $\sum_s aY_{s,a}^{(N)}(t) \leq \alpha$ by

$$\sum_s a\mathbb{E}\left[Y_{s,a}^{(N)}(t)\right] \leq \alpha.$$

Lemma

Finding the optimal allocation can be solved by an LP.

The LP-relaxation

Replace the activation constraint $\sum_s aY_{s,a}^{(N)}(t) \leq \alpha$ by

$$\sum_s a\mathbb{E}\left[Y_{s,a}^{(N)}(t)\right] \leq \alpha.$$

Lemma

Finding the optimal allocation can be solved by an LP.

Proof. Let $y_{s,a}(t) := \mathbb{E}\left[Y_{s,a}^{(N)}(t)\right]$ and $x_s(t) = \mathbb{E}\left[X_s^{(N)}(t)\right]$.

$$\begin{aligned}
\max \sum_{s,a,t} & y_{s,a}(t) r_{s,a} \\
\text{s.t.} \quad & \sum_a a\, y_{s,a}(t) \leq \alpha N && \forall t \\
& \sum_{s,a} y_{s,a}(t) p(s' \mid s, a) = x_{s'}(t+1) && \forall t, s' \\
& \sum_a y_{s',a}(t) = x_{s'}(t).
\end{aligned}$$

How can we use the LP to build a policy for the original problem?

$\sum_s a Y_{s,a}^{(N)}(t) \leq \alpha$		$\sum_s a \mathbb{E}\left[Y_{s,a}^{(N)}(t) \right] \leq \alpha$
Original problem (Hard)	$\longrightarrow$	LP relaxation (Easy)
V_N^*	$\leq$	V_{rel}^*

How can we use the LP to build a policy for the original problem?

$$\sum_s a Y_{s,a}^{(N)}(t) \leq \alpha \qquad \sum_s a \mathbb{E}\left[Y_{s,a}^{(N)}(t) \right] \leq \alpha$$

Original problem (Hard) $\longrightarrow$ LP relaxation (Easy)

$$V_N^* \leq V_{rel}^*$$

$$V_{rel}^* \downarrow x^*, y^*$$

How can we use the LP to build a policy for the original problem?

$$\sum_s a Y_{s,a}^{(N)}(t) \le \alpha \qquad\qquad \sum_s a \mathbb{E}\left[Y_{s,a}^{(N)}(t)\right] \le \alpha$$

Original problem (Hard) $\longrightarrow$ LP relaxation (Easy)

$$V_N^* \le V_{rel}^*$$

$$V_N^* \wr V_N^\pi$$

V_N^π —— Can we build π that is close to optimal? —— x^*, y^* (obtained from V_{rel}^*)

Subject of (G, Gaujal, Yan 2021), (Frazier et al 2020), (Brown and Smith 2019)

LP-relaxation vs Original problem

Relaxed problem: Optimal sequence $x_s^*(t)$, $y_{s,a}^*(t)$.

LP-relaxation vs Original problem

Relaxed problem: Optimal sequence $x_s^*(t)$, $y_{s,a}^*(t)$.
Original problem: Sequence $\pi_t : \mathcal{X} \to \mathcal{Y}$.

- A policy is LP-compatible if $\pi_t(x^*) = y^*$.

LP-relaxation vs Original problem

Relaxed problem: Optimal sequence $x_s^*(t)$, $y_{s,a}^*(t)$.
Original problem: Sequence $\pi_t : \mathcal{X} \to \mathcal{Y}$.

- A policy is LP-compatible if $\pi_t(x^*) = y^*$.

Theorem (G., Gaujal, Yan 2021)

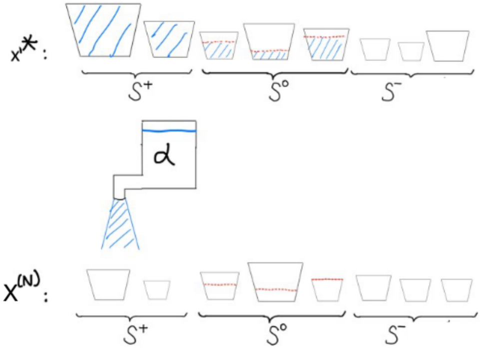
- *A continuous policy is LP-compatible iff* $\lim_{N\to\infty} V_N^\pi = V_{rel}^*$.
- *A locally linear LP-compatible policy satisfies*

$$|V_N^\pi - V_{rel}^*| \leq C_1 e^{-C_2 N}.$$

Proof (sketch). If π is continuous: $\lim_{N\to\infty} X_\pi^{(N)}(t) = x_\pi(t)$ and therefore $\lim_{N\to\infty} V_N^\pi = V_{rel}^\pi$.

How to build an LP-compatible policy: water-filling

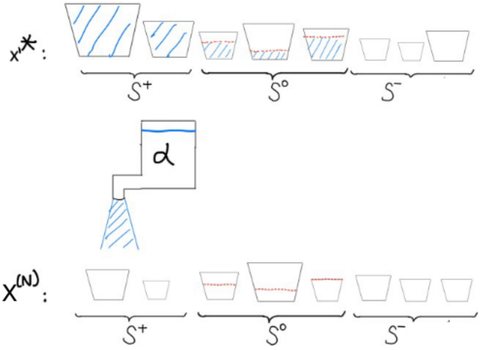
LP-compatible: find a function $\pi : \mathcal{X} \to \mathcal{Y}$ such that $\pi(x^*) = y^*$.



where:
- $S^+ = \{s : y_{s,0}(t) = 0\}$
- $S^- = \{s : y_{s,1}(t) = 0\}$
- $S^0 = \{s : y_{s,0}(t) > 0 \wedge y_{s,1}(t) > 0\}$

How to build an LP-compatible policy: water-filling

LP-compatible: find a function $\pi : \mathcal{X} \to \mathcal{Y}$ such that $\pi(x^*) = y^*$.



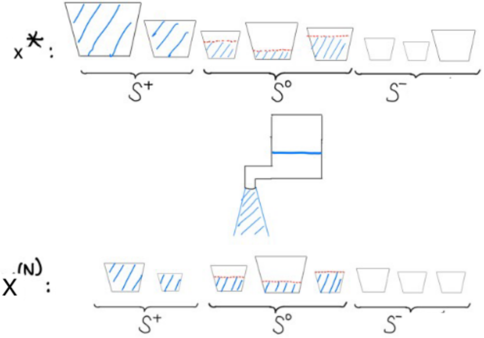
- Priority to S^+

where:

- $S^+ = \{s : y_{s,0}(t) = 0\}$
- $S^- = \{s : y_{s,1}(t) = 0\}$
- $S^0 = \{s : y_{s,0}(t) > 0 \wedge y_{s,1}(t) > 0\}$

How to build an LP-compatible policy: water-filling

LP-compatible: find a function $\pi : \mathcal{X} \to \mathcal{Y}$ such that $\pi(x^*) = y^*$.

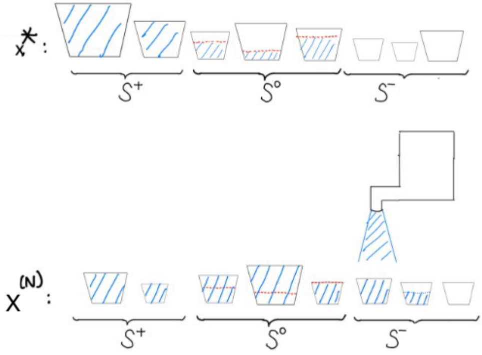


- Priority to S^+
- Second priority to S^0 up to $y^*_{s,1}$.

where:
- $S^+ = \{s : y_{s,0}(t) = 0\}$
- $S^- = \{s : y_{s,1}(t) = 0\}$
- $S^0 = \{s : y_{s,0}(t) > 0 \wedge y_{s,1}(t) > 0\}$

How to build an LP-compatible policy: water-filling

LP-compatible: find a function $\pi : \mathcal{X} \to \mathcal{Y}$ such that $\pi(x^*) = y^*$.

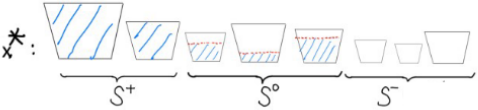


- Priority to S^+
- Second priority to S^0 up to $y^*_{s,1}$.
- Fill with the rest.

where:
- $S^+ = \{s : y_{s,0}(t) = 0\}$
- $S^- = \{s : y_{s,1}(t) = 0\}$
- $S^0 = \{s : y_{s,0}(t) > 0 \wedge y_{s,1}(t) > 0\}$

Existence of an LP-compatible policy

Non-degenerate = for all t: $|S^0(t)| \geq 1$.
Rankable = for all t: $S^0(t) \leq 1$.



Theorem

- *For any problem, there exists an LP-compatible policy.*
- *If the problem is non-degenerate, then there exists a locally linear LP-compatible policy.*
- *If the problem is rankable, there exists a strict priority policy that is LP-compatible.*

Illustration: Applicant screening problem

Figure from (Brown Smith 2020)

Candidates with prior quality Beta(1,1), Interview budget $\alpha = 0.25$

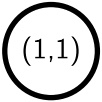
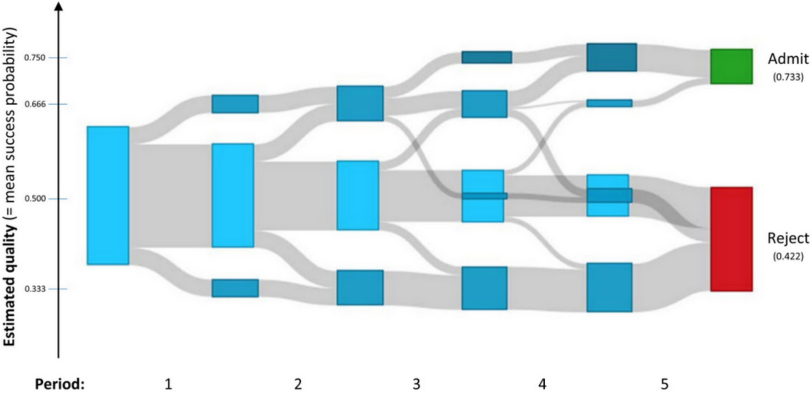


Illustration: Applicant screening problem

Figure from (Brown Smith 2020)

Candidates with prior quality Beta(1,1), Interview budget α=0.25

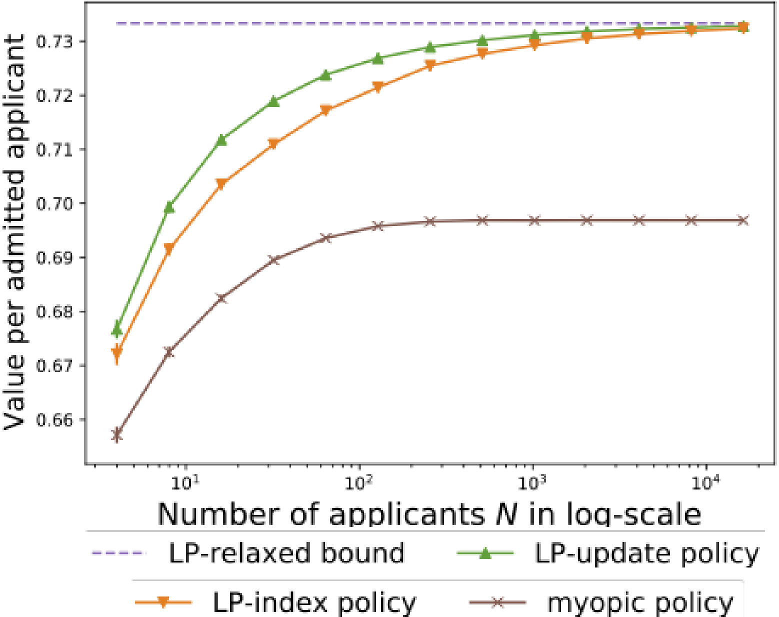


Example: optimal relaxed solution: after two interviews:

- $x^*_{(1,1)} = 2x^*_{(2,1)} = 2x^*_{(1,2)} = 0.5$.
- $y^*_{(2,1),interview} = y^*_{(1,1),interview} = 0.125$.

Optimality

LP-index = policy from (Brown Smith 2020), LP-update = update LP solution.



Outline

Subsidy and indexability

Main idea: use a Lagrangian decomposition and replace the constraint $\sum \mathbb{E}[aY_{s,a}] = \alpha$ by a subsidy $a\mu$ to action a.

An arm is a 2-action MDP: if in state s:

- Activation: earn $r(s, active)$, jump to $j \sim \mathbf{P}[j \mid s, active]$.
- Passive: earn $r(s, passive)$, jump to $j \sim \mathbf{P}[j \mid s, passive]$.

Subsidy and indexability

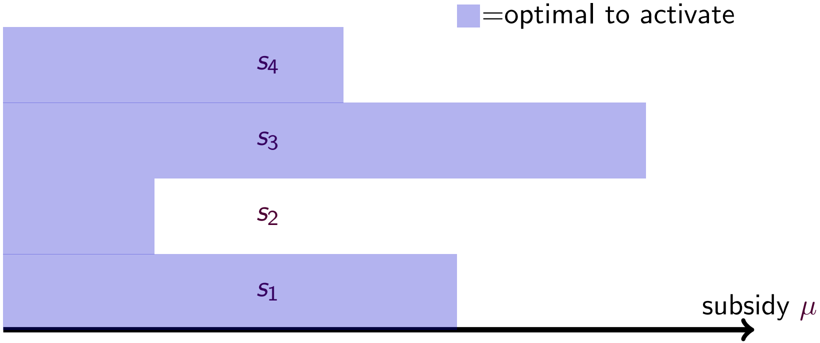
Main idea: use a Lagrangian decomposition and replace the constraint $\sum \mathbb{E}\left[aY_{s,a}\right] = \alpha$ by a subsidy $a\mu$ to action a.

An arm is a 2-action MDP: if in state s:

- Activation: earn $r(s, active) + \mu$, jump to $j \sim \mathbf{P}\left[j \mid s, active\right]$.
- Passive: earn $r(s, passive)$, jump to $j \sim \mathbf{P}\left[j \mid s, passive\right]$.

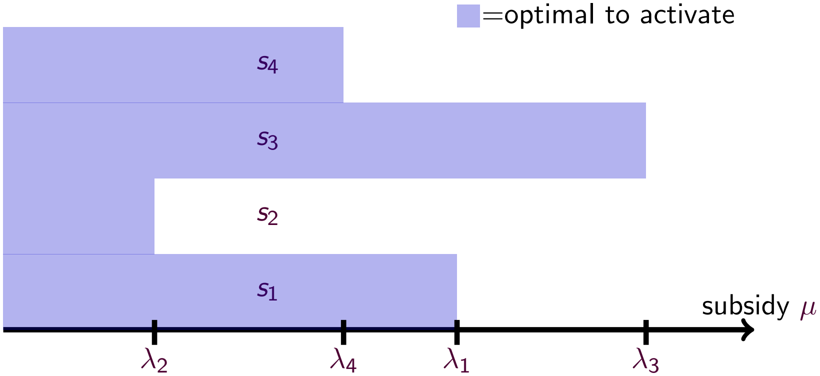
In general, the "optimal" subsidy depends on α.

What is the subsidy and who should we activate?



[1] Verloop. Asymptotically optimal priority policies for indexable and nonindexable restless bandits. (2016) Annals of Applied Probability.

What is the subsidy and who should we activate?



If the subsidy for which "activation" is optimal for s is $(-\infty, \lambda(s)]$, then the state is indexable and $\lambda(s)$ is its Whittle index.

- Activate arms by decreasing order of Whittle index.

Whittle policy is[1] LP-compatible for infinite horizon.

[1] Verloop. Asymptotically optimal priority policies for indexable and nonindexable restless bandits. (2016) Annals of Applied Probability.

Asymptotic optimality of Whittle index

Theorem (Weber Weiss 90s, Verloop 2016)

For infinite horizon, an LP-compatible policy is asymptotically optimal in under the "global attractor condition".

(G. Gaujal Yan 2021) *Holds with exponential rate for non-degenerate problems.*

Illustration with a stochastic scheduling problem

Example: two jobs of sizes X and Y with:

- $X = 10 - \varepsilon$
- $Y = \begin{cases} 2 & \text{proba } 1/2 \\ 18 & \text{proba } 1/2 \end{cases}$

Illustration with a stochastic scheduling problem

Example: two jobs of sizes X and Y with:

- $X = 10 - \varepsilon$
- $Y = \begin{cases} 2 & \text{proba } 1/2 \\ 18 & \text{proba } 1/2 \end{cases}$

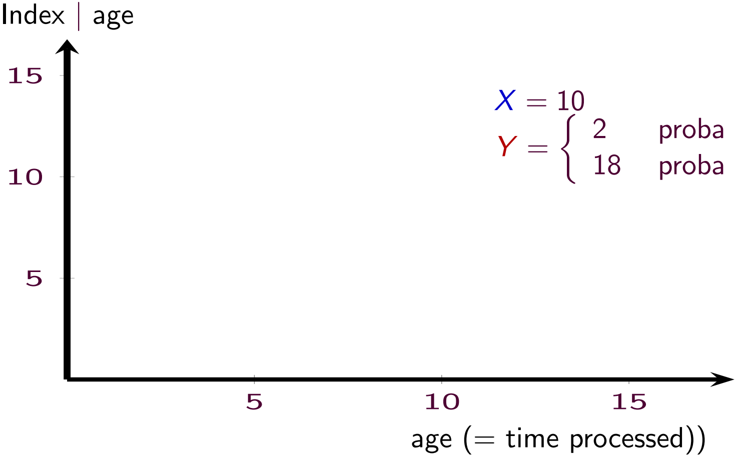
For this model: Gittins (=Whittle) index policy is optimal:

- serve job with smallest index first.

Running a job costs 1€/sec and you can stop anytime. If you finish the job, you earn x. Gittins index = smallest x so that you running or stopping is equivalent.

Illustration with a stochastic scheduling problem

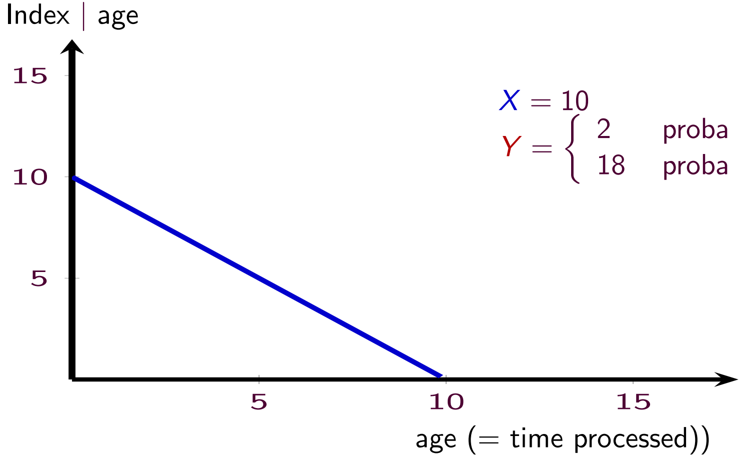
Running a job costs 1€/sec and you can stop anytime. If you finish the job, you earn x. Gittins index = smallest x so that you running or stopping is equivalent.



$$X = 10$$
$$Y = \begin{cases} 2 & \text{proba } 1/2 \\ 18 & \text{proba } 1/2 \end{cases}$$

Illustration with a stochastic scheduling problem

Running a job costs 1€/sec and you can stop anytime. If you finish the job, you earn x. Gittins index = smallest x so that you running or stopping is equivalent.

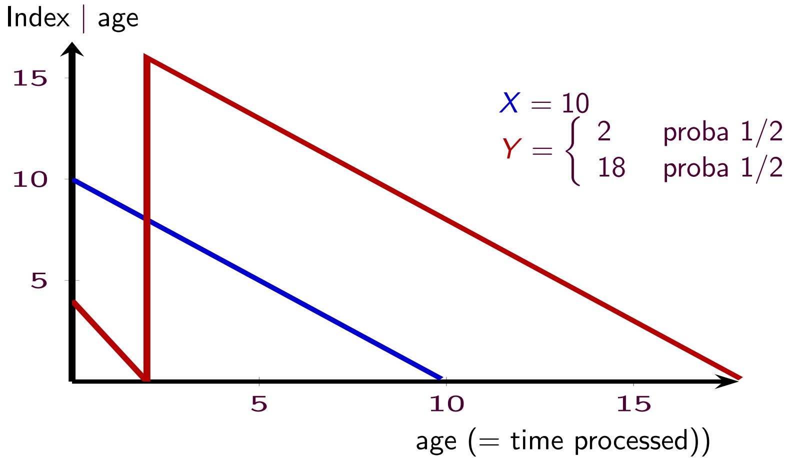


$X = 10$

$$Y = \begin{cases} 2 & \text{proba } 1/2 \\ 18 & \text{proba } 1/2 \end{cases}$$

Illustration with a stochastic scheduling problem

Running a job costs 1€/sec and you can stop anytime. If you finish the job, you earn x. Gittins index = smallest x so that you running or stopping is equivalent.



Scheduling: Are all related problems closed since the 90s?

Scheduling: Are all related problems closed since the 90s?

(Recently) closed questions:

- Can we analyze the performance of this Gittins index? (SOAP: [Scully et al 2018]).
- Multi-server? (Close to optimal: [Grosof et al 2019]).

Scheduling: Are all related problems closed since the 90s?

(Recently) closed questions:

- Can we analyze the performance of this Gittins index? (SOAP: [Scully et al 2018]).
- Multi-server? (Close to optimal: [Grosof et al 2019]).
- At which speed do Whittle index become optimal? (exponentially fast in most cases [G,Gaujal,Yan 2021])
- Can we define index for finite-horizon problems? [Hu and Frazier 2019-20, Brown Smith 2020, Gaujal, Yan 2022].
- Can we leverage indexable to problem to obtain better learning algorithms? (No regret learning.[G,G,Khun, 2021])

Scheduling: Are all related problems closed since the 90s?

(Recently) closed questions:

- Can we analyze the performance of this Gittins index? (SOAP: [Scully et al 2018]).
- Multi-server? (Close to optimal: [Grosof et al 2019]).
- At which speed do Whittle index become optimal? (exponentially fast in most cases [G,Gaujal,Yan 2021])
- Can we define index for finite-horizon problems? [Hu and Frazier 2019-20, Brown Smith 2020, Gaujal, Yan 2022].
- Can we leverage indexable to problem to obtain better learning algorithms? (No regret learning.[G,G,Khun, 2021])
- Are Whittle index hard to compute? [G, G. Khun, 2022]

Outline

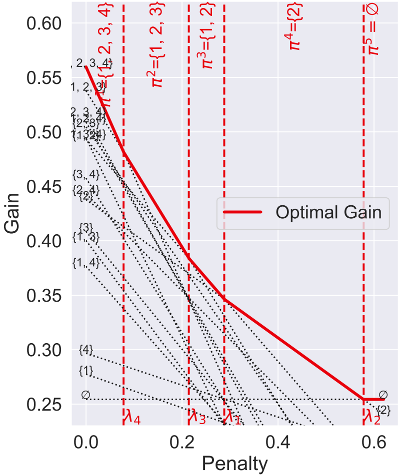
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \ldots S\}$
- $g_\pi(x) =$ value for subsidy x.
- $\pi^*(x) = \arg\max_{\pi} g_\pi(x)$.

General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \dots S\}$
- $g_\pi(x) =$ value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.



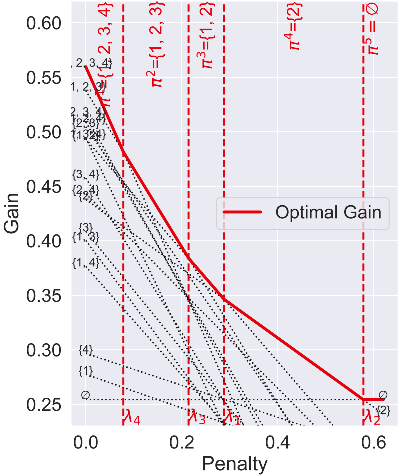
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \dots S\}$
- $g_\pi(x)$ = value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.

Facts:

1. $g_\pi(x)$ is linear in x.



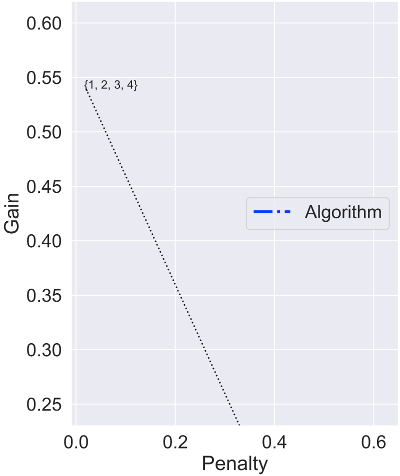
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \dots S\}$
- $g_\pi(x)$ = value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.

Facts:

1. $g_\pi(x)$ is linear in x.
2. $\pi_*(-\infty) = \{1 \dots S\}$.



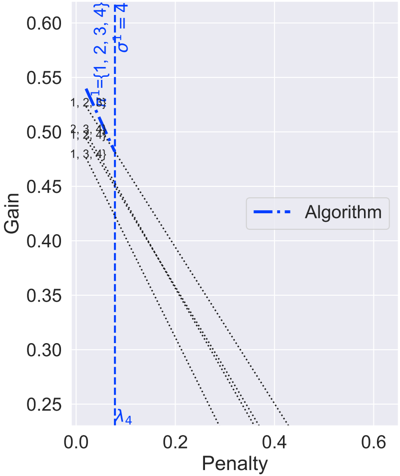
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \dots S\}$
- $g_\pi(x)$ = value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.

Facts:

1. $g_\pi(x)$ is linear in x.
2. $\pi_*(-\infty) = \{1 \dots S\}$.
3. Computing $g_{\pi\setminus\{i\}}(x)$ from $g_\pi(x)$ is easy.



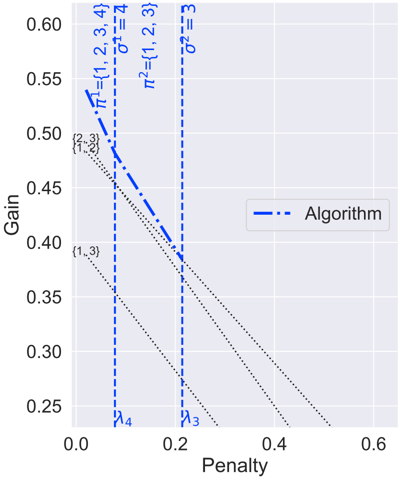
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \dots S\}$
- $g_\pi(x)$ = value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.

Facts:

1. $g_\pi(x)$ is linear in x.
2. $\pi_*(-\infty) = \{1 \dots S\}$.
3. Computing $g_{\pi \setminus \{i\}}(x)$ from $g_\pi(x)$ is easy.



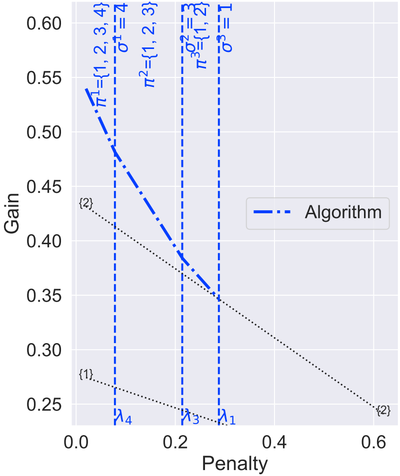
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \ldots S\}$
- $g_\pi(x)$ = value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.

Facts:

1. $g_\pi(x)$ is linear in x.
2. $\pi_*(-\infty) = \{1 \ldots S\}$.
3. Computing $g_{\pi \setminus \{i\}}(x)$ from $g_\pi(x)$ is easy.



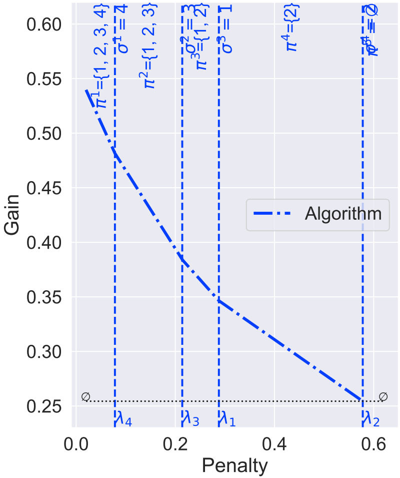
General idea: compute the index by increasing order

- A policy is $\pi \subset \{1 \dots S\}$
- $g_\pi(x)$ = value for subsidy x.
- $\pi^*(x) = \arg\max_\pi g_\pi(x)$.

We want to find the inflection points of the red curve.

Facts:

1. $g_\pi(x)$ is linear in x.
2. $\pi_*(-\infty) = \{1 \dots S\}$.
3. Computing $g_{\pi \setminus \{i\}}(x)$ from $g_\pi(x)$ is easy.



Why are the facts true?

- $g_\pi(x) = (A^\pi)^{-1}(r + x\pi)$

Why are the facts true?

- $g_\pi(x) = (A^\pi)^{-1}(r + x\pi)$
- If subsidy is $-\infty$ for all states, then we should not activate.

Why are the facts true?

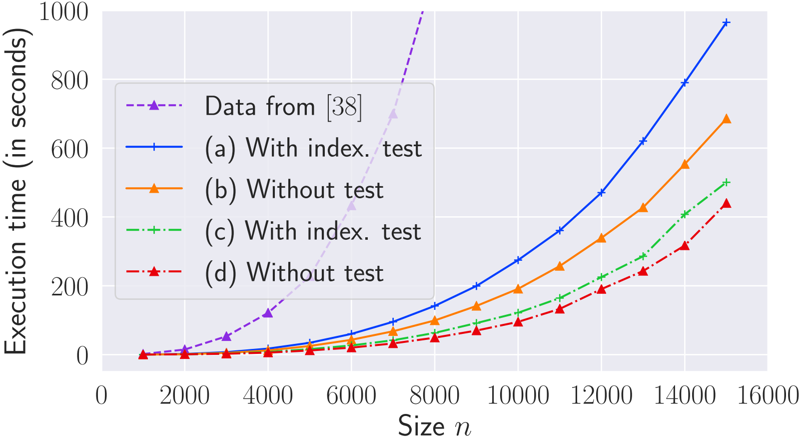
- $g_\pi(x) = (A^\pi)^{-1}(r + x\pi)$
- If subsidy is $-\infty$ for all states, then we should not activate.
- Sherman-Morisson formula: Let A be an invertible matrix, u and v vectors $1D$ such that $1 + v^T A^{-1} u \neq 0$. Then:

$$\left(A + uv^T\right)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1} u}.$$

By using fast matrix multiplication, we can compute Whittle indices in $O(S^{2.53})$ operations (conjectured to be at least n^3 in a 2016 paper).

Simulation result

https://pypi.org/project/markovianbandit-pkg/



Outline

Finite-horizon restless bandits

Subsidy, infinite-horizon and index policies

How to Compute Indices: A Sub-Cubic Algorithm

Conclusion

Conclusion

Index policies / LP relaxation are efficient ways to share resources among tasks.

- Idea: it allows you to compute a "right price" for each resource. You can then activate the cheapest up to your budget.

This scales and performs very well in practice.

- This talk: Optimality of index and computation of index.
- Open questions: learning, continuous state-spaces.

Conclusion

Index policies / LP relaxation are efficient ways to share resources among tasks.

- Idea: it allows you to compute a "right price" for each resource. You can then activate the cheapest up to your budget.

This scales and performs very well in practice.

- This talk: Optimality of index and computation of index.
- Open questions: learning, continuous state-spaces.

http://polaris.imag.fr/nicolas.gast/

- *Computing Whittle (and Gittins) Index in Subcubic Time*, G. Gaujal, Khun https://arxiv.org/abs/2203.05207
- *LP-based policies for restless bandits: necessary and sufficient conditions for (exponentially fast) asymptotic optimality*. G. Gaujal Yan. https://arxiv.org/abs/2106.10067