Learning-Augmented Dynamic Power Management with Multiple States via New Ski Rental Bounds

A. Antoniadis, C. Coester, M. Eliáš, A. Polak, B. Simon

December 13, 2022

NeurIPS 2021

slides from M. Eliáš

Dynamic power management (DPM)

Machine with multiple sleep states:

server.png

S-states.png

Dynamic power management (DPM)

Machine with multiple sleep states:

server.png

S-states.png

Deeper sleep → higher wake-up cost

Dynamic power management (DPM)

Machine with multiple sleep states:

server.png

S-states.png

Deeper sleep → higher wake-up cost

Input: sequence of idle periods

Our task: choose sleep states during each idle period

Dynamic power management (DPM)

Machine with multiple sleep states:

server.png

S-states.png

Deeper sleep → higher wake-up cost

Input: sequence of idle periods

Our task: choose sleep states during each idle period

- duration is not known in advance!

Dynamic power management: previous works

Algorithms:

- Irani, Shukla, and Gupta (2003)
- Lotker, Patt-Shamir, and Rawitz (2012)

Predicting lengths of the idle periods:

- Benini, Bogliolo, and Micheli (2000)
- Chung, Benini, Bogliolo, Lu, and Micheli (2002)

Heuristics:

- Helmbold, Long, Sconyers, and Sherrod (2000)
- Lim, Sharma, Tak, and Das (2011)

Learning-augmented algorithms

- introduced by Lykouris and Vassilvitskii (2018)

Algorithms receive predictions

- e.g. learned from past data

Learning-augmented algorithms

- introduced by Lykouris and Vassilvitskii (2018)

Algorithms receive predictions

- e.g. learned from past data

Three desired properties:

Consistency

Close-to-optimal performance with close-to-accurate predictions.

Learning-augmented algorithms

- introduced by Lykouris and Vassilvitskii (2018)

Algorithms receive predictions

- e.g. learned from past data

Three desired properties:

Consistency

Close-to-optimal performance with close-to-accurate predictions.

Robustness

Strong guarantees also with incorrect predictions.

Learning-augmented algorithms

- introduced by Lykouris and Vassilvitskii (2018)

Algorithms receive predictions

- e.g. learned from past data

Three desired properties:

Consistency

Close-to-optimal performance with close-to-accurate predictions.

Robustness

Strong guarantees also with incorrect predictions.

Smoothness

Performance deteriorates smoothly with the prediction error.

Predictions for DPM

At the beginning of idle period i:

- prediction τ_i of the length ℓ_i of the idle period i

Predictions for DPM

At the beginning of idle period i:

- prediction τ_i of the length ℓ_i of the idle period i

Prediction error in ith idle period

$$\eta_i = \alpha \cdot |\tau_i - \ell_i|,$$

- α = power consumption of the active state

Predictions for DPM

At the beginning of idle period i:

- prediction τ_i of the length ℓ_i of the idle period i

Prediction error in ith idle period

$$\eta_i = \alpha \cdot |\tau_i - \ell_i|,$$

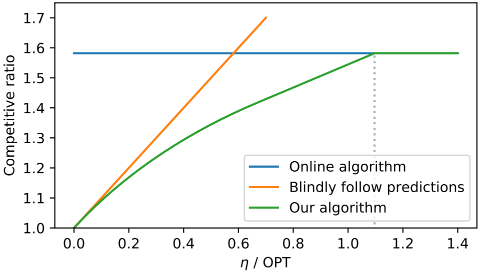
- α = power consumption of the active state

Prediction error during the whole instance

$$\eta = \sum_{i=1}^{n} \eta_i$$

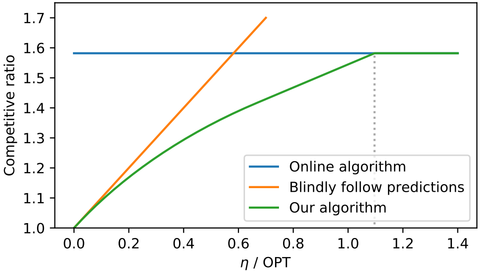
Our results

Algorithm for DPM with a theoretical guarantee



Our results

Algorithm for DPM with a theoretical guarantee

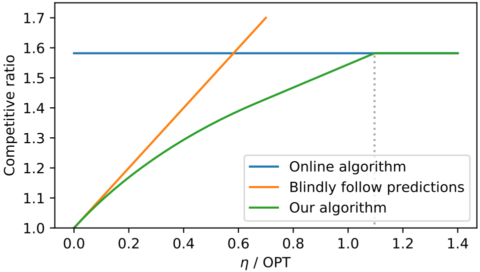


New bounds for ski rental

- performance as a function of prediction error
- optimal trade-off: consistency vs. error dependence

Our results

Algorithm for DPM with a theoretical guarantee



New bounds for ski rental

- performance as a function of prediction error
- optimal trade-off: consistency vs. error dependence

Experimental evaluation

Outline of the technical part

1. ski rental: optimal consistency/error dependence trade-off
2. reduction from DPM to ski rental
3. finding the best trade-off online

Ski rental

ski_rental2.jpg

Fundamental problem:

- see Phillips, Westbrook (1999) and Irani, Karlin (1996)
- implicit in many problems in Online Optimization
- appears in Online Learning with switching costs

Ski rental

During a ski season of unknown length:

- each day we decide whether to
  - rent the skis for one more day paying α, or
  - buy the skis paying cost β

Ski rental

During a ski season of unknown length:

- each day we decide whether to
  - rent the skis for one more day paying α, or
  - buy the skis paying cost β

Corresponds to the following DPM:

- single idle period (of unknown length) and two states:
  - active state: power consumption α, wake-up cost $\mathbf{0}$
  - sleep state: power consumption $\mathbf{0}$, wake-up cost β

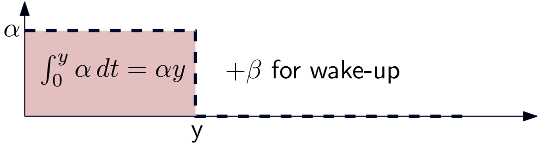
Ski rental

During a ski season of unknown length:

- each day we decide whether to
  - rent the skis for one more day paying α, or
  - buy the skis paying cost β

Corresponds to the following DPM:

- single idle period (of unknown length) and two states:
  - active state: power consumption α, wake-up cost $\mathbf{0}$
  - sleep state: power consumption $\mathbf{0}$, wake-up cost β
- cost if switching to the sleep state at time y:



Ski rental: previous work

Competitive ratio

$$\max \frac{\text{cost}(ALG)}{\text{cost}(OPT)}$$

Competitive ratio

$$\max \frac{\mathrm{cost}(ALG)}{\mathrm{cost}(OPT)}$$

Without predictions:

- 2-competitive deterministic (folklore)
- $\frac{e}{e-1}$-competitive randomized (Karlin et al. '90)

Ski rental: previous work

Competitive ratio

$$\max \frac{\text{cost}(ALG)}{\text{cost}(OPT)}$$

Without predictions:

- 2-competitive deterministic (folklore)
- $\frac{e}{e-1}$-competitive randomized (Karlin et al. '90)

Learning augmented:

- Purohit, Svitkina, Kumar (2018)
- deterministic algorithm with a trade-off parameter $\lambda \in (0, 1)$
- correct prediction: $(1 + \lambda)$-competitive **(consistency)**
- wrong prediction: $(1 + \frac{1}{\lambda})$-competitive **(robustness)**
- randomized algorithm with an improved trade-off
- Gollapudi, Panigrahi '20, Wei, Zhang '20, Angelopoulos et al. '20

Trade-off between consistency and error dependence

Performance as a function of prediction error η

- algorithm is (ρ, μ)-competitive, if

$$\text{cost}(ALG) \leq \rho \cdot \text{cost}(OPT) + \mu \cdot \eta$$

Trade-off between consistency and error dependence

Performance as a function of prediction error η

- algorithm is (ρ, μ)-competitive, if

$$\text{cost}(ALG) \leq \rho \cdot \text{cost}(OPT) + \mu \cdot \eta$$

Examples:

- algorithm which follows the prediction blindly:

$$\rho = 1, \quad \mu = 1$$

Trade-off between consistency and error dependence

Performance as a function of prediction error η

- algorithm is (ρ, μ)-competitive, if

$$\text{cost}(ALG) \leq \rho \cdot \text{cost}(OPT) + \mu \cdot \eta$$

Examples:

- algorithm which follows the prediction blindly:

$$\rho = 1, \quad \mu = 1$$

- standard online algorithms which do not use predictions:

$\rho = 2,\ \mu = 0$ deterministic; $\quad \rho = \frac{e}{e-1},\ \mu = 0$ randomized

Trade-off between consistency and error dependence

Performance as a function of prediction error η

- algorithm is (ρ, μ)-competitive, if

$$\text{cost}(ALG) \leq \rho \cdot \text{cost}(OPT) + \mu \cdot \eta$$

Examples:

- algorithm which follows the prediction blindly:

$$\rho = 1, \quad \mu = 1$$

- standard online algorithms which do not use predictions:

$$\rho = 2,\ \mu = 0 \text{ deterministic}; \quad \rho = \tfrac{e}{e-1},\ \mu = 0 \text{ randomized}$$

- (ρ, μ)-competitiveness of Purohit et al. (2018)
  - either $\mu \geq 1$,
  - or $\mu = 0$ and $\rho \geq \frac{e}{e-1}$

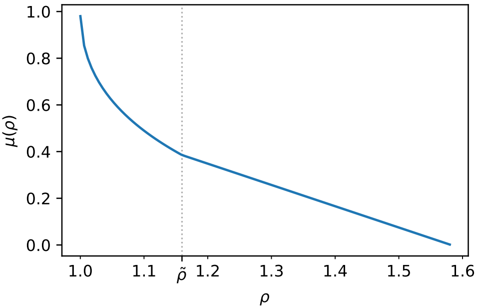
Our new bound for ski rental

Theorem (optimal trade-off between ρ and μ):

- there is a (ρ, μ)-competitive algorithm for ski rental, where

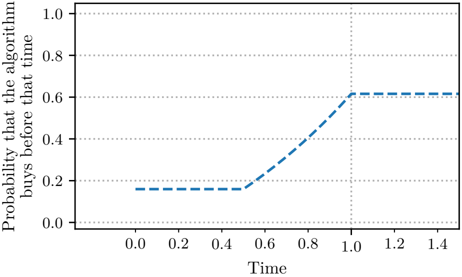
$$\rho \in \left[1, \frac{e}{e-1}\right] \text{ and } \mu = \mu(\rho) = \max\left\{\frac{1 - \rho\frac{e-1}{e}}{\ln 2}, \rho(1-T)e^{-T}\right\}$$

- $T \in [0, 1]$ is the solution to $T^2 e^{-T} = 1 - \frac{1}{\rho}$



Randomized algorithm

- determined by a CDF F of buying until a given time

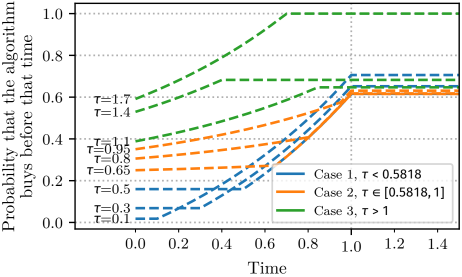


- sample $p \sim U([0, 1])$, buy at the first time t s.t. $F(t) \geq p$.

Our algorithm for ski rental

Randomized algorithm

- determined by a CDF F of buying until a given time

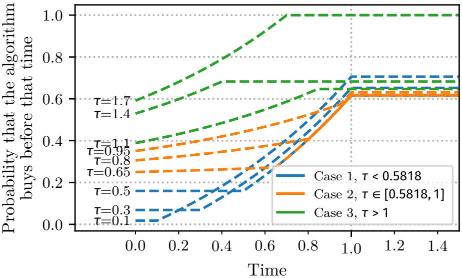


- sample $p \sim U([0,1])$, buy at the first time t s.t. $F(t) \geq p$.

Our algorithm for ski rental

Randomized algorithm

- determined by a CDF F of buying until a given time



- sample $p \sim U([0,1])$, buy at the first time t s.t. $F(t) \geq p$.

Necessary condition for (ρ, μ)-competitiveness

- for any prediction τ and time t:

$$\text{renting cost} + \text{buying cost} \leq \rho \cdot \min\{1, t\} + \mu \cdot |t - \tau|$$

- obstacle: $|t - \tau|$ is not monotone

Multi-state DPM

- power consumptions $\alpha_0 > \alpha_1 > \cdots > \alpha_k = 0$
- wake-up costs $0 = \beta_0 < \beta_1 < \cdots < \beta_k$

Multi-state DPM

- power consumptions $\alpha_0 > \alpha_1 > \cdots > \alpha_k = 0$
- wake-up costs $0 = \beta_0 < \beta_1 < \cdots < \beta_k$

Construct k instances of ski rental:

- $j = 1, \dots, k$: rental cost $\alpha_{j-1} - \alpha_j$, buying cost $\beta_j - \beta_{j-1}$
- switch from state $j - 1$ to j using the ski rental algorithm

Ski rental → multistate DPM

Multi-state DPM

- power consumptions $\alpha_0 > \alpha_1 > \cdots > \alpha_k = 0$
- wake-up costs $0 = \beta_0 < \beta_1 < \cdots < \beta_k$

Construct k instances of ski rental:

- $j = 1, \ldots, k$: rental cost $\alpha_{j-1} - \alpha_j$, buying cost $\beta_j - \beta_{j-1}$
- switch from state $j - 1$ to j using the ski rental algorithm

What if switch $j \to j + 1$ happens before $j - 1 \to j$?

Ski rental → multistate DPM

Multi-state DPM

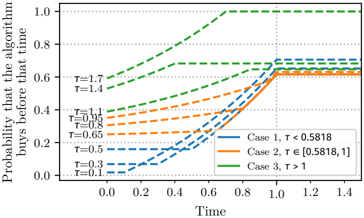
- power consumptions $\alpha_0 > \alpha_1 > \cdots > \alpha_k = 0$
- wake-up costs $0 = \beta_0 < \beta_1 < \cdots < \beta_k$

Construct k instances of ski rental:

- $j = 1, \ldots, k$: rental cost $\alpha_{j-1} - \alpha_j$, buying cost $\beta_j - \beta_{j-1}$
- switch from state $j - 1$ to j using the ski rental algorithm

What if switch $j \to j + 1$ happens before $j - 1 \to j$?

- we need to make ski rental algorithm monotone



Multi-state DPM

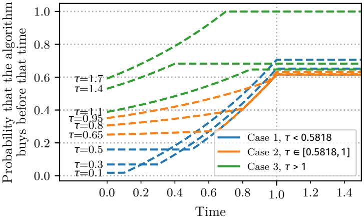
- power consumptions $\alpha_0 > \alpha_1 > \cdots > \alpha_k = 0$
- wake-up costs $0 = \beta_0 < \beta_1 < \cdots < \beta_k$

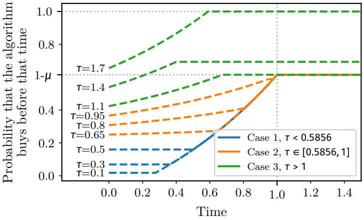
Construct k instances of ski rental:

- $j = 1, \ldots, k$: rental cost $\alpha_{j-1} - \alpha_j$, buying cost $\beta_j - \beta_{j-1}$
- switch from state $j-1$ to j using the ski rental algorithm

What if switch $j \to j+1$ happens before $j-1 \to j$?

- we need to make ski rental algorithm monotone



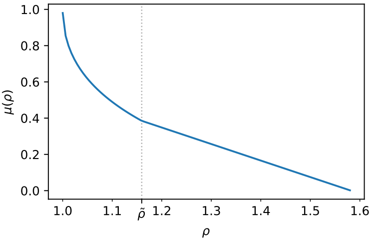


What ρ and μ are the best?

What we have so far:

- given a parameter $\rho \in [1, \frac{e}{e-1}]$, there is *ALG* for DPM with

$$\text{cost}(ALG) \leq \rho \cdot \text{cost}(OPT) + \mu(\rho) \cdot \eta$$

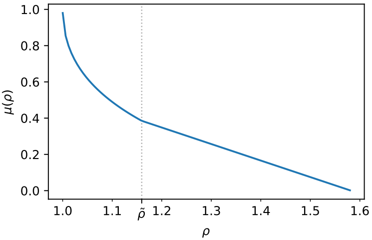


What ρ and μ are the best?

What we have so far:

- given a parameter $\rho \in [1, \frac{e}{e-1}]$, there is ALG for DPM with

$$\text{cost}(ALG) \leq \rho \cdot \text{cost}(OPT) + \mu(\rho) \cdot \eta$$



No good choice of ρ and μ a priori

- $\mu(\rho) > 0$: ALG has unbounded cost with large η
- $\mu(\rho) = 0$: ALG ignores predictions completely

What ρ and μ are the best?

If we knew the prediction error in advance

- for an instance with a total error $\boldsymbol{\eta}$:

$$\rho^* = \arg\min_{\rho} \left\{ \rho \cdot \text{cost}(OPT) + \mu(\rho) \cdot \eta \right\}$$

What ρ and μ are the best?

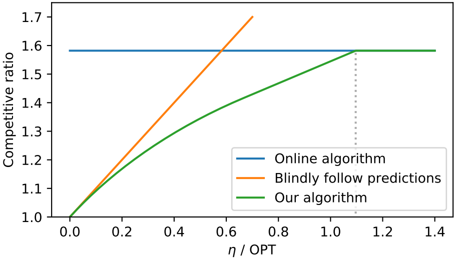
If we knew the prediction error in advance

- for an instance with a total error η:

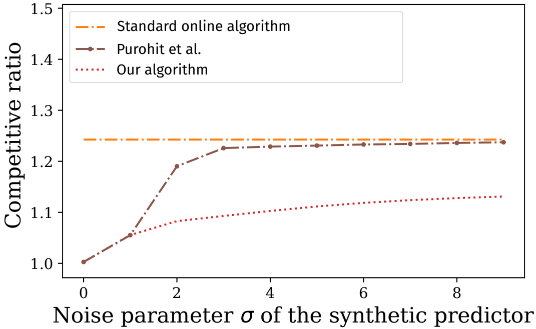
$$\rho^* = \arg\min_{\rho} \left\{ \rho \cdot \mathrm{cost}(OPT) + \mu(\rho) \cdot \eta \right\}$$

We choose ρ using Online Learning

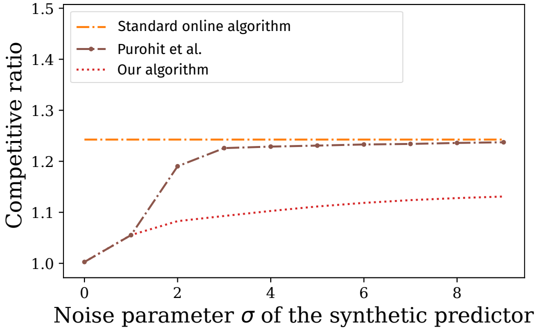
$$\mathrm{cost}(ALG) \leq (1+\epsilon)\Big(\rho^* \cdot \mathrm{cost}(OPT) + \mu(\rho^*) \cdot \eta\Big) + O\left(\frac{\beta}{\epsilon} \log \frac{1}{\epsilon}\right)$$



Experimental results



Experimental results



Thank you for your attention!