Finite-Time Information-Theoretic Bounds in Queueing Control

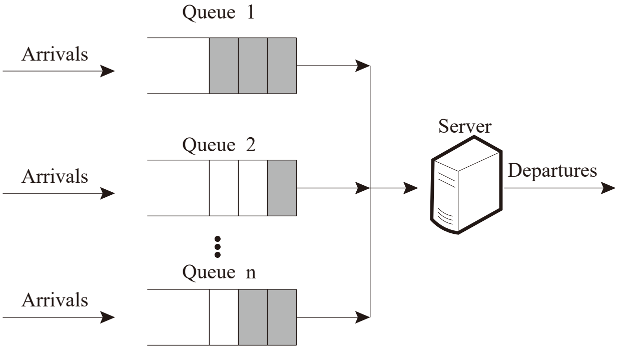
Yunbei Xu

with Yujie Liu and Vincent Y. F. Tan

National University of Singapore

Oct. 2025

The Scheduling Problem in Queueing



Deciding which queues to serve at each time step to optimize performance metrics like delay, waiting time, or throughput

Why Finite-Time Queue Scheduling is Critical

Figure: NVIDIA's open-sourced KAI Scheduler

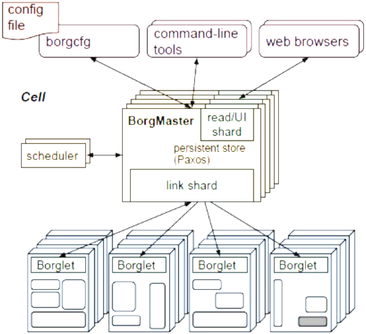


Figure: The high-level architecture of Google Borg

Bursty, non-stationary workload, millisecond-level lease granularity

Question: Can we speak about fundamental limits at time T (not just steady state), and design policies that hit those limits?

MaxWeight Policy

De facto policy: MaxWeight (Back-pressure) policy

- **Asymptotic (steady-state)** guarantees: throughput-optimality (stability for all rates in the interior of the capacity region), diffusion optimality in heavy traffic for many settings
  - [Tassiulas & Ephremides, 2002], [Stolyar, 2004], [Mandelbaum & Stolyar, 2004], [Dai & Lin, 2008]
- **Limitations** of MaxWeight in **finite-time** regime
  - MaxWeight tends to pick extreme schedules, and transient backlogs can grow large before averaging effects [Shah & Wischik, 2006], [Bramson, D'Auria, Walton 2021] (validated in our experiments)

Gap: little is known theoretically about its parameter-dependent performance and fundamental limitations in non-asymptotic settings

Central Questions

- How can we formulate a finite-time language (minimax framework) for the scheduling problem in queueing systems?
- Within this framework, what is the minimum achievable queue length by time T?
- Can MaxWeight attain this minimum?
- If not, what alternative scheduling policies can possibly achieve it, and under what conditions?

Key Answers

- **Finite-time information-theoretic lower bound:** first minimax framework for the scheduling problem in queueing; fundamental limit for any scheduling policy
- **MaxWeight is not minimax-optimal:** In finite-time regime, its backlog exceeds the lower bound by a geometry-dependent factor
- **Introducing** $\mathrm{LyapOpt}$ **policy:** Minimizes the *full* Lyapunov drift (first- and second-order terms). $\mathrm{LyapOpt}$ matches the lower bound up to absolute constants
- **Extensive simulations:** $\mathrm{LyapOpt}$ consistently outperforms MaxWeight across a wide range of scenarios

Bridging Queueing Control and Learning Theory

- **Not just regret.** Most learning results study *regret* when the model is unknown (RL). Here we ask: even if the model is known (oracle DP), what is the best achievable performance in finite time under randomness?
- **Queueing as structured DP.** Single-hop SPNs give a clean DP testbed. We build an **information-theoretic** toolkit for *finite-horizon* analysis (instead of only steady-state/asymptotic results).
- **Fundamental limits, which motivate algorithms.** We prove *minimax lower bounds* that hold for *any* policy, and design policies that match them—giving sharp benchmarks for short-horizon control.

Related Works: Key Areas

- **Finite-horizon analyses in queueing.**
  - Challenging even for simple $M/M/1$. [Abate & Whitt, 1987]
  - Convergence-to-steady-state via coupling/spectral methods—not finite-time backlog with explicit scaling. [Robert, 2013; Gamarnik & Goldberg, 2013]
  - Results for specific policies/topologies, e.g., JSQ; do not cover general scheduling. [Luczak & McDiarmid, 2006; Ma & Maguluri, 2025]
- **Parameter learning in queueing (unknown rates, partial feedback).**
  - *Queueing regret.* [Krishnasamy, Sen, Johari, Shakkottai, 2021]; [Stahlbuhk, Shrade, Modiano, 2021]; [Freund, Lykouris, Weng, 2023]
  - *Time-averaged queue length.* [Yang, Srikant, Ying, 2023]
  - *Adversarial stability (AQT).* Focus on universal stability/delay, not time-T backlog scaling. [Borodin, Kleinberg, Raghavan, Sudan, Williamson, 2001]

Related Works: Key Areas (Cont.)

- **Lower bounds for structured DP.**
  - Queueing control is computationally hard (curse of dimensionality). [Papadimitriou & Tsitsiklis, 1987]
  - One work on delay lower bound: $G/D/1$ queue. [Gupta & Shroff, 2009]
- **Drift-method limitations and alternatives.**
  - Classical Lyapunov drift targets stability/steady-state performance. [Eryilmaz & Srikant, 2012; Maguluri & Srikant, 2016]
  - Drift-plus-penalty is a steady-state tradeoff framework. [Neely, 2010]
  - These are largely *asymptotic* and *first-order*; they do not directly yield *finite-time minimax* bounds with explicit parameter dependence.

Outline

- **Part 1: Problem Setup & Minimax Framework**
- **Part 2: General Lower Bounds**
- **Part 3: Finite-Time Performance Guarantees**
- **Part 4: Experiments**

Goal in mind: Explain a finite-time, parameter-explicit theory for single-hop scheduling; show a gap for MaxWeight; present a policy (LyapOpt) that matches a minimax lower bound.

Part 1: Problem Setup & Minimax Framework

Problem Setup: Single-Hop SPN

Discrete-time single-hop SPN with n parallel queues.

- $Q(t)$: Queue length vector at time t.
- $A(t)$: Arrival vector at time t.
  $\lambda(t) = \mathbb{E}[A(t)]$: Mean arrival rate vector.
- Scheduling set $\mathcal{D}_t \subseteq \mathbb{R}_+^n$: Each $D(t) \in \mathcal{D}_t$ is a "schedule" (jobs departing).

Queueing Dynamics

$$Q(t+1) = \max\{Q(t) - D(t), \mathbf{0}\} + A(t+1), \quad t \geq 0$$

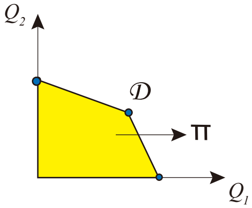
with $Q(0) = \mathbf{0}$.

Problem Setup: Arrival Processes

- **Adversarial Arrivals and Departure Sets:** $\{A(t), \mathcal{D}_t\}_{t \geq 0}$ chosen by an adversary, potentially with arbitrary dependencies.
- **Stochastic Arrivals and Fixed Departure Set (special case):** $\{A(t)\}_{t \geq 0}$ is i.i.d. with mean λ. $\mathcal{D}_t \equiv \mathcal{D}$ fixed.

Definition (Capacity region)

$\Pi_t = \{\gamma \in \mathbb{R}_+^n : \gamma \leq d, \text{ for some } d \in \text{conv}(\mathcal{D}_t)\}$.



Assumption

$\lambda(t) \in \rho\Pi$ for all $t \geq 0$, $\rho \in (0, 1]$.

Problem Setup: Policy

History: $\mathcal{H}_t = \{(\mathcal{D}_0, D(0), A(1)), \ldots, (\mathcal{D}_{t-1}, D(t-1), A(t)), \mathcal{D}_t\}$.

Policy

A *policy* $\Phi = \{\phi_t\}_{t \geq 0}$. $\phi_t : \mathcal{H}_t \rightarrow$ Probability distribution over $\mathcal{D}_t$

- $D(t) \in \mathcal{D}_t$ is chosen according to the distribution $\phi_t(\mathcal{H}_t)$.

Goal: Minimize cumulative queue length

Minimax Criteria

- Standard approach in statistics [Wald, 1945], optimization [Nemirovsky & Yudin, 1978], and machine learning to study finite-sample (finite-horizon) difficulty.
- When no exact limit is known, quantify the best achievable performance via the $\inf_{\Phi} \sup_{\mathcal{M}}$ criterion.
- Traditionally concerning regret due to model uncertainty and partial feedback; extends to queueing control and (stochastic) DP oracle here.
- First minimax formulation for finite-time fundamental limits of scheduling policies.

Minimax Criteria: Performance Metrics

- **Total Queue Length:** captures the overall system backlog

$$\mathbb{E}\left[\sum_{i=1}^{n} Q_i(T)\right]$$

This is a (stochastic) Dynamic Programming problem:

Objective at t depends on all past decisions.

Minimax Criteria: Model Classes

Model Classes: Arrival Process and Scheduling Set

General class $\mathcal{M}^{\rho}(C, B)$:

$$\Big\{ (A(\cdot), \{\mathcal{D}_t\}) : \lambda(t) \in \rho \mathcal{D}_t, \frac{1}{n}\sum_{i=1}^{n} \mathsf{Var}(A_i(t)) \le C^2, \ \forall t \ge 0; \quad \frac{1}{n}\sum_{i=1}^{n} d_i^2 \le B^2, \ \forall d \in \mathcal{D} \Big\},$$

- $\rho \in (0, 1]$: Traffic intensity. $\rho \to 1$ is "heavy traffic"
- $C \ge 0$: Arrival variability; can generalize to random departure as well.
- $B > 0$: Scheduling set diameter.

Minimax Criteria: Fundamental Lower Bound

Goal: find the fundamental minimax lower bound at time T:

$$\inf_{\Phi} \sup_{(A(\cdot),\{\mathcal{D}_t\})\in\mathcal{M}^{\rho}(C,B)} \mathbb{E}_{\Phi,(A(\cdot),\mathcal{D})}\left[\sum_{i=1}^{n} Q_i(T)\right].$$

- First-ever minimax formulation for finite-time fundamental limitations of queueing control
- Offers a principled approach to quantifying the hardness of structured dynamic scheduling problems

Part 2: General Lower Bounds

Theorem: Lower Bounds – Fundamental Limit

Theorem (**General Lower Bounds**)

For any scheduling policy, and for arrival processes and scheduling sets within the model class $\mathcal{M}^{\rho}(C, B)$, the following lower bound holds:
*For all $T > c_0 \left(\frac{B^2}{nC^2} + \frac{nC^2}{B^2} + 1 \right)$, there is a **unified lower bound** that covers both the heavy-traffic ($\rho \to 1$) and interior ($\rho \in (0, 1)$) regimes:*

$$\inf_{\Phi} \sup_{\mathcal{M}^{\rho}(C,B)} \mathbb{E}\left[\sum_{i=1}^{n} Q_i(T)\right] \geq c_1 \min\left\{ nC\sqrt{T-1}, \frac{nC^2}{B(1-\rho)} \right\} + \sqrt{n}\,\rho B,$$

where c_0 and c_1 are positive absolute constants.

General Lower Bounds: Proof Blueprint

Proof sketch (high level).

1. **Reduce DP to partial sums.** Lower bound the queueing recursion by a functional of partial sums (i.e., finite sums of random variables). In the *oracle DP* (known parameters), use *stochastic-process* lower bounds rather than statistical tools (Le Cam/Fano).
2. **Deviation via Gaussian / random walk.** Couple to a Gaussian or random walk, combine a sharp proxy bound with an approximation error. Heavy traffic ($\rho \to 1$): mean-zero $\sqrt{T}$-type lower bound. Interior ($\rho \in (0,1)$): negatively drifted walk with $(1-\rho)^{-1}$ behavior.
3. **Gaussian-to-general approximation error.** Control the approximation error via strong-approximation techniques, e.g., the Komlós–Major–Tusnády (KMT) coupling, which provides uniform (in time) coupling with quantifiable error terms.

Implications of Lower Bounds

- Bounds explicitly quantify **scaling with**:
  - Time horizon T
  - Variance parameter of arrival C (no B for heavy-traffic $\rho \to 1$)
  - Number of queues n
  - In interior cases, constant scaling with $\frac{1}{1-\rho}$
- **Fundamental Benchmark:** No policy can guarantee better than $nC\sqrt{T}$ (heavy-traffic) or $\frac{nC^2}{B(1-\rho)}$ (interior) scaling in finite horizon.
- **Next Step:** Introduce a novel algorithm that matches this lower bound by explicitly optimizing both first- and second-order Lyapunov terms, addressing the identified gap.

Part 3: Finite-Time Performance Guarantees

Optimal Lyapunov Policy ($\mathrm{LyapOpt}$)

Recall: $Q(t+1) = \max\{Q(t) - D(t), \mathbf{0}\} + A(t+1)$

$\mathrm{LyapOpt}$ policy

At each time t, select $D(t)$ as the solution to:

$$D(t) \in \operatorname*{argmin}_{d \in \mathcal{D}} \sum_{i=1}^{n} \left(\max\{Q_i(t) - d_i, 0\}\right)^2$$

- Minimizes a surrogate of the Lyapunov ($V(x) = \|x\|_2^2$) drift: $\Delta V(t) = \mathbb{E}[V(Q(t+1) - A(t+1)) - V(Q(t) - A(t)) \mid \mathcal{H}_t]$.
- **Novelty:** Optimizes the full one-step Lyapunov drift, including second-order terms.
- **Stability:** $\mathrm{LyapOpt}$ is throughput optimal (i.e., stable in interior regime). Smaller Lyapunov drift than MaxWeight $\Rightarrow$ positive recurrence in one line.

General Lyapunov Drift Analysis

- For any policy, arrival processes and scheduling sets in $\mathcal{M}(C, B)$:
  One-step Lyapunov drift:

$$\Delta V(t) \le f(Q(t), D(t)) + r(Q(t), A(t+1))$$

$$\text{where } f(Q(t), d) = \mathbb{E}\left[\underbrace{2\sum_{i=1}^{n} Q_i(t)\big(\lambda_i(t) - d_i\big)}_{\text{first-order term}} + \underbrace{\sum_{i=1}^{n} \big(d_i^2 - \lambda_i(t)^2\big)}_{\text{second-order term}} \,\Bigg|\, \mathcal{H}_t \right]$$

and $\mathbb{E}[r(Q(t), A(t+1))] = \sum_{i=1}^{n} \mathrm{Var}(A_i(t+1))$.

- Summing over time and applying Jensen's and Cauchy–Schwarz inequalities:

$$\mathbb{E}\left[\sum_{i=1}^{n} Q_i(T) \right] \le n \sqrt{ \sum_{t=1}^{T-1} \mathbb{E}[f(Q(t), D(t))]/n + (T-1)C^2 + \sum_{i=1}^{n} \mathbb{E}[A_i(T)]}.$$

Theorem: Finite-Time Performance of LyapOpt

Theorem (Finite-Time Performance of the LyapOpt Policy)

Within $\mathcal{M}^\rho(C, B)$, if $\lambda(t) \in \mathcal{D}_t$ for all $t \geq 0$, the LyapOpt *policy achieves:*

$$\mathbb{E}\Big[\sum_{i=1}^{n} Q_i(T)\Big] \leq nC\sqrt{T-1} + \sum_{i=1}^{n} \mathbb{E}[A_i(T)]$$

- When $\lambda(t) \in \mathcal{D}_t$, LyapOpt perfectly matches the arrival rate and achieves the fundamental lower bound (up to a constant factor), establishing its finite-time optimality.

MaxWeight Policy

MaxWeight policy

Selects schedules $D^{\mathsf{MaxWeight}}(t)$ according to:

$$D^{\mathsf{MaxWeight}}(t) \in \underset{d \in \mathcal{D}}{\operatorname{argmax}} \langle Q(t), d \rangle$$

- Optimizes the first-order Lyapunov term, prioritizing queues with larger backlogs.

Theorem: Upper Bound of MaxWeight Policy

$$f(Q(t), d) = \mathbb{E}\left[\underbrace{2\sum_{i=1}^{n} Q_i(t)\big(\lambda_i(t) - d_i\big)}_{\text{first-order term}} + \underbrace{\sum_{i=1}^{n}\big(d_i^2 - \lambda_i(t)^2\big)}_{\text{second-order term}} \;\middle|\; \mathcal{H}_t \right]$$

Theorem (Upper Bound of MaxWeight Policy)

Under $\mathcal{M}^\rho(C, B)$, the MaxWeight policy satisfies

$$\mathbb{E}\left[\sum_{i=1}^{n} Q_i(T)\right] \le n\sqrt{(B^2 + C^2)(T-1)} + \sum_{i=1}^{n} \mathbb{E}[A_i(T)].$$

Limitation of MaxWeight (and drift-based methods)

- Ignores second-order effects
- Always selects extreme points, fails to adapt to arrival-rate geometry

MaxWeight Lower Bound (Dimension 2)

Proposition

There exists a family of instances with $\rho \in (0,1]$, $B \geq 3\sqrt{2}$ and $1 \leq C \leq B$, for which the expected total queue lengths under MaxWeight satisfy:

$$\lim_{\rho \to 1} \sup_{\mathcal{M}^{\rho}(C,B)} \mathbb{E}\Big[\sum_{i=1}^{2} Q_i^{\mathrm{MaxWeight}}(T)\Big] \geq \frac{C\sqrt{T}}{2\sqrt{2e\pi}} + \frac{BT^{\frac{1}{3}}}{4\sqrt{3}}, \quad T \geq \frac{B^2}{C^2} + \frac{C^2}{B^2} + 9.$$

In particular, there exists an instance in $M^1(0,B)$ with $B \geq 3\sqrt{2}$ and

$$\mathbb{E}\Big[\sum_{i=1}^{2} Q_i^{\mathsf{LyapOpt}}(T)\Big] = 2 - \frac{1}{\sqrt{2}\,B}, \quad T \geq 1,$$

$$\mathbb{E}\Big[\sum_{i=1}^{2} Q_i^{\mathrm{MaxWeight}}(T)\Big] \geq \frac{\sqrt{BT}}{2\sqrt{2e\pi}}, \quad \left\lceil \frac{2B^2}{\sqrt{2}B - 1} \right\rceil \leq T \leq \left\lceil (\frac{\sqrt{2}B}{2} - 1)^3 \right\rceil.$$

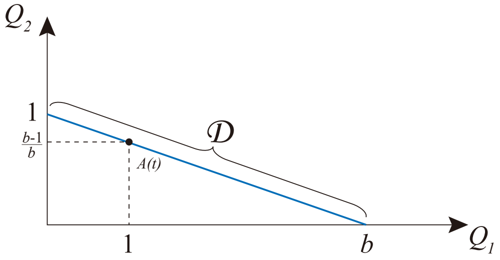
MaxWeight Lower Bound (Dimension 2)

Construction: Consider the scheduling set and arrivals

$$\mathcal{D} = \{d \in \mathbb{R}^2 : d = x(b,0) + (1-x)(0,1), 0 \le x \le 1\},$$

$$A(t) = (1, (b-1)/b) \text{ for all } t \ge 1, \text{ and } A(0) = (1, (b-1)/b - \varepsilon),$$

with $b = \sqrt{2}B$



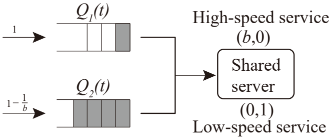
MaxWeight Lower Bound (Dimension 2)

Proof Sketch:

- **MaxWeight's Selection Rule:**
  - MaxWeight always choose extreme points $(0, 1)$ or $(b, 0)$;
  - At each time t, MaxWeight selects $(b, 0)$ unless $\frac{Q_2(t)}{Q_1(t)} \geq b$.
- **Queue Dynamics under MaxWeight:**
  - $Q_2(t)$ accumulates to b before the first use of $(0, 1)$.
  - Using $(0, 1)$ increases $Q_1(t)$ to 2.
  - To use $(0, 1)$ again, $Q_2(t)$ must build up to $2b$.
  - This alternating pattern causes $Q_2(t)$ to grow at rate $\sqrt{bT}$ over a finite horizon T.
- **Contrast with $\mathrm{LyapOpt}$:**
  - Always chooses the "true arrival" schedule $(1, (b-1)/b)$.
  - Maintains constant queue lengths $O(1)$.

MaxWeight Lower Bound (Dimension 2)

Common in wireless networks and data centers.



MaxWeight Behavior:

- Over-prioritizes Q_1 via extreme-point selection.
- $Q_2(t)$ builds up due to limited service.
- Highlights MaxWeight's finite-time inefficiency.

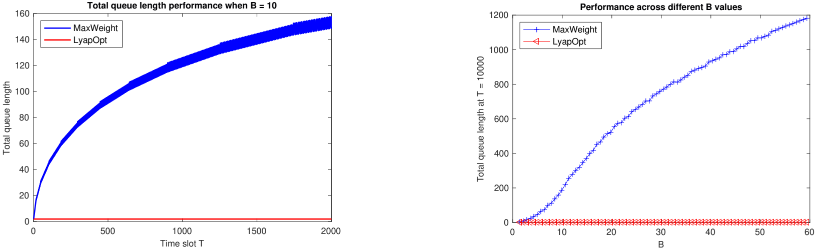
LyapOpt Contrast:

- Adapts to arrival asymmetry.
- Prevents backlog in this special case.

Part 4: Experiments

MaxWeight Lower Bound (Dimension 2)

Empirical Validation: The $\sqrt{BT}$ gap is substantial for practical T and B



(a) Total queue length when $B = 10$

(b) Total queue length across different B

Figure: Performance comparison of MaxWeight and LyapOpt policies versus B

Experiments with More Queues

Experimental Setup

- **Scheduling Set $\mathcal{D}$:** $10n$ integer vectors uniformly sampled from $[1, 10]^n$.
- **Arrival Rates:** 2000 vectors sampled from the boundary of the capacity region Π.
- **Arrival Distributions:** Binomial with variance 1, matching the sampled arrival rates.
- **Simulation:** 1000 time slots, averaged over 100 runs per scenario.

Experiments with More Queues

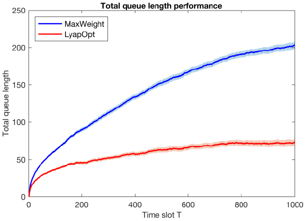
- ratio $= \frac{\text{Total Queue Length } (\mathrm{LyapOpt}) \text{ at } t = 1000}{\text{Total Queue Length (MaxWeight) at } t = 1000}$

Table: Proportion of scenarios with ratio below 1, 0.9, and 0.5

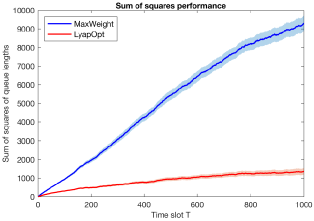
Number of Queues (n)	ratio ≤ 1	ratio ≤ 0.9	ratio ≤ 0.5
2	84.7%	25.9%	0%
3	97.5%	54.1%	36.3%
4	99.9%	78.5%	46.1%
5	100%	67.0%	31.3%
6	97.4%	71.3%	26.5%
7	100%	90.0%	45.9%
8	100%	80.7%	35.9%

- $\mathrm{LyapOpt}$ achieves consistently better performance than MaxWeight for $n = 2$ to 8.
- Significant improvements observed in many cases (ratio ≤ 0.5).

Representative Case Study ($n = 8$ Queues)



(a) Total queue length.



(b) Squared queue length.

Figure: Finite-time comparison of MaxWeight and LyapOpt policies ($n = 8$).

- Both policies show $\sqrt{T}$ growth in total queue length and linear T growth in squared queue length.
- $\mathrm{LyapOpt}$ yields lower total queue length and better balance.

Summary

- Exposed a **finite-time gap** between MaxWeight and the minimax lower bound.
- Proposed $\mathrm{LyapOpt}$, a **second-order** Lyapunov policy that *closes this gap*.
- Theory & simulations: $\mathrm{LyapOpt}$ yields **shorter queues** than MaxWeight over finite horizons.
- Clarifies the **limitations of drift-based (first-order) methods** in transient regimes.
- **Complements steady-state** analyses: revealing interesting finite-time, parameter-dependent phenomena (e.g., geometric structure and second-order effects).

Future Directions

- **Multi-hop networks.** Extend lower bounds and $\mathrm{LyapOpt}$-style policies beyond single hop.
- **Model geometry.** In $G/G/1$, the feasible set $\mathcal{D}$ is a *weighted simplex*; refined analysis to common decision sets.
- **Unknown parameters.**
    - Unknown *arrival rates*: seamlessly covered by our work.
    - n-queue, m-server systems with *unknown service rates*: requires **UCB-type exploration** with backlog-aware exploitation.
- **Computation.** MaxWeight solves a *linear* problem over $\mathcal{D}$ (often LP / min-cut–max-flow); $\mathrm{LyapOpt}$ involves a *quadratic* objective over $\mathcal{D}$—design fast approximations and oracle reductions.