# International Conference on Networks, Games, Control and Optimization (NetGCoop)

The next edition will be held in Bilbao from 08/10/2025 to 10/10/2025

## Save the date!

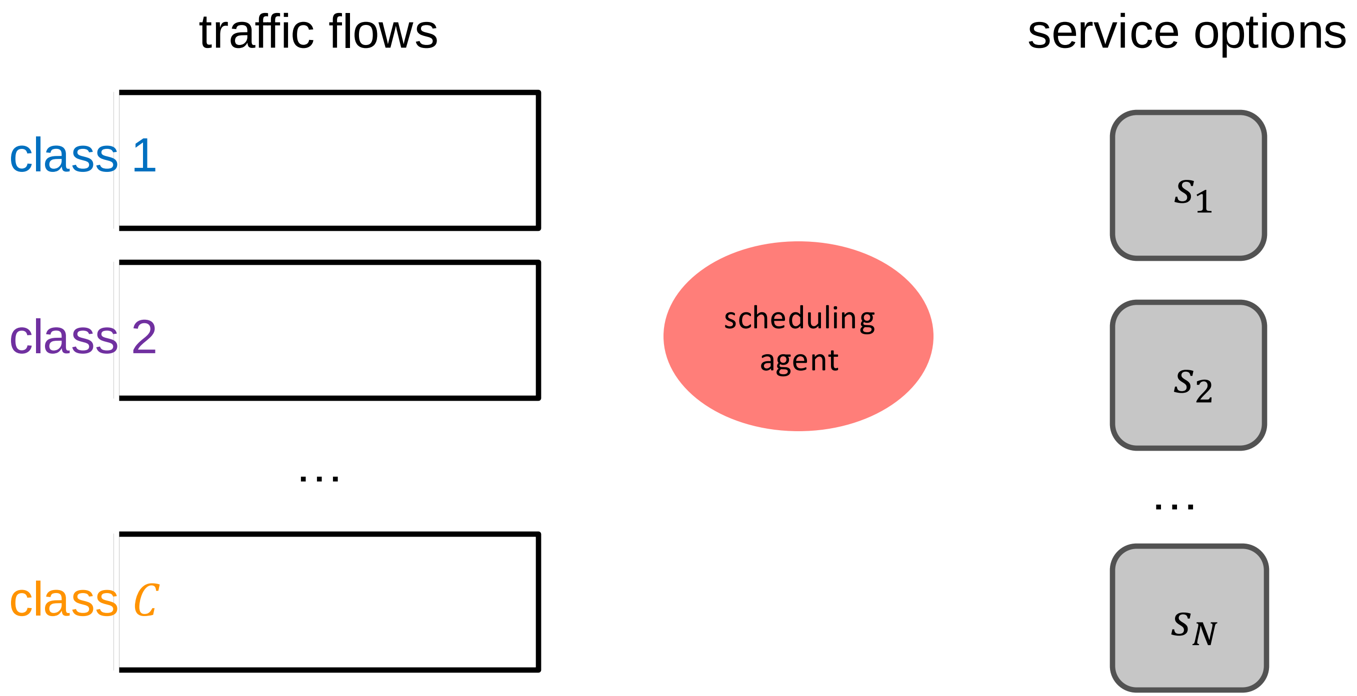
More information soon on the web site:
https://netgcoop2025.univ-avignon.fr/

# Stability and performance of multi-class queueing systems with unknown service rates: A scheduling-based approach

**Elene ANTON (Université de Pau et des Pays de l'Adour - UPPA)**
**Joint work with Sem Borst (TU/e)**

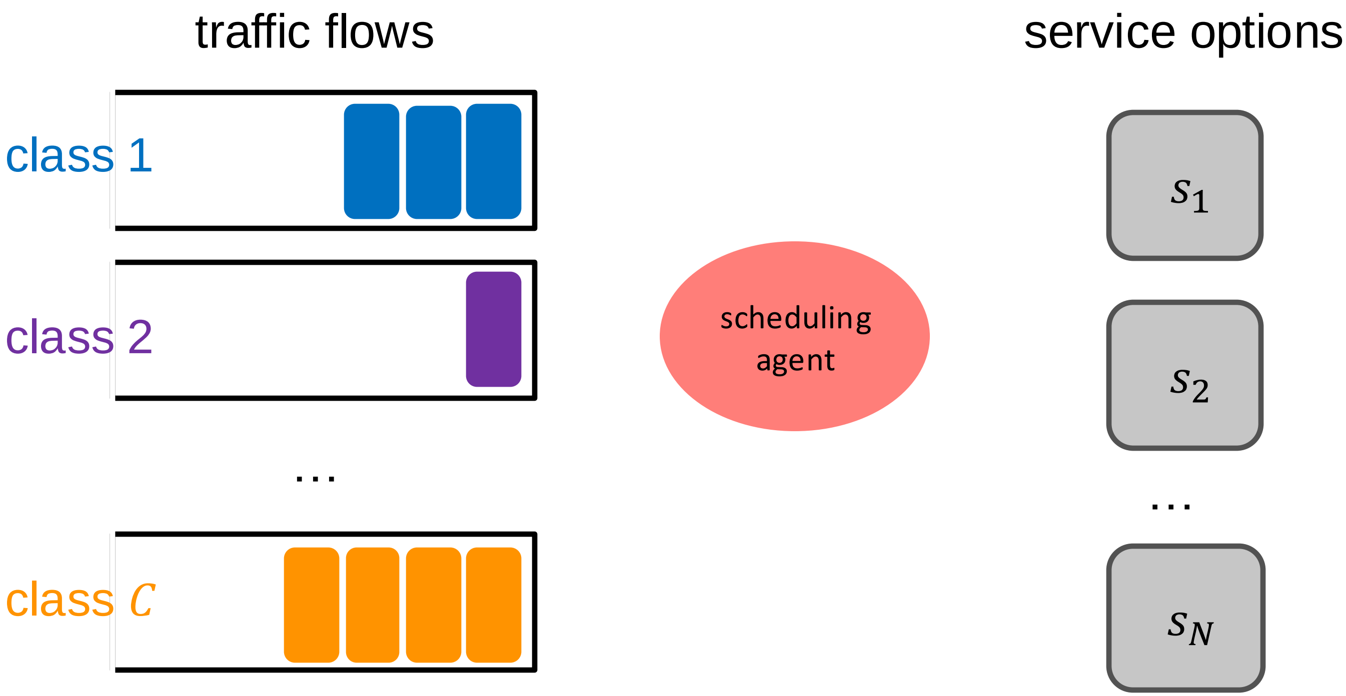
13ème Atelier en Évaluation des Performances , 2-4 Décembre 2024, Toulouse, France

# 1. MODEL DESCRIPTION



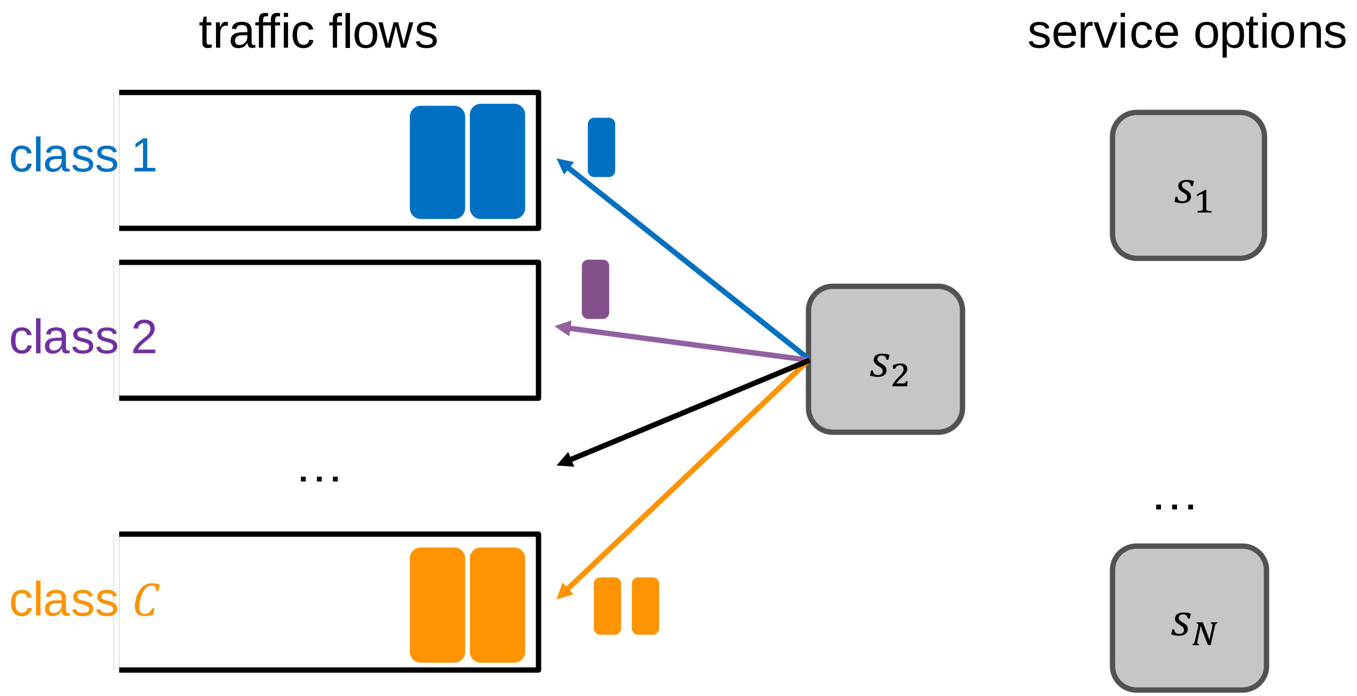

- $C$ classes of trafic flows.
- $N$ mutually exclusive service options/modes.
- Input-queued system.
- Real-world examples : channel/frequency selection in wireless communications (e.g. WiFi networks or cognitive radio systems), …

# 1. MODEL DESCRIPTION




- Time slotted operation.
- The number of jobs that arrive per class is independent across time slots.

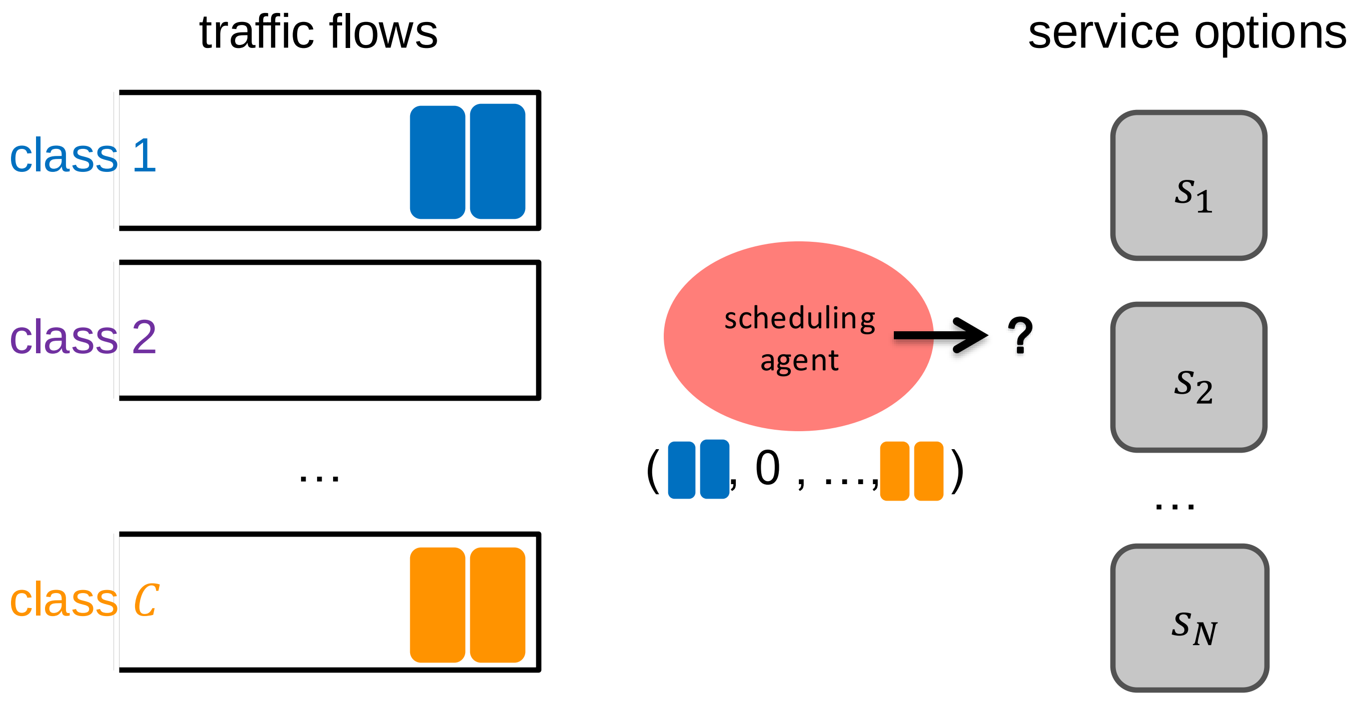
# 1. MODEL DESCRIPTION




Each time slot,

- a single service mode/option can be selected,
- when service mode/option $s$ is selected in time slot $t$, (up to) $R_{c,s}(t)$ class-$c$ jobs will be served
- neither realizations nor statistics of $R_{c,s}(t)$ are known to scheduling agent

# 1. MODEL DESCRIPTION

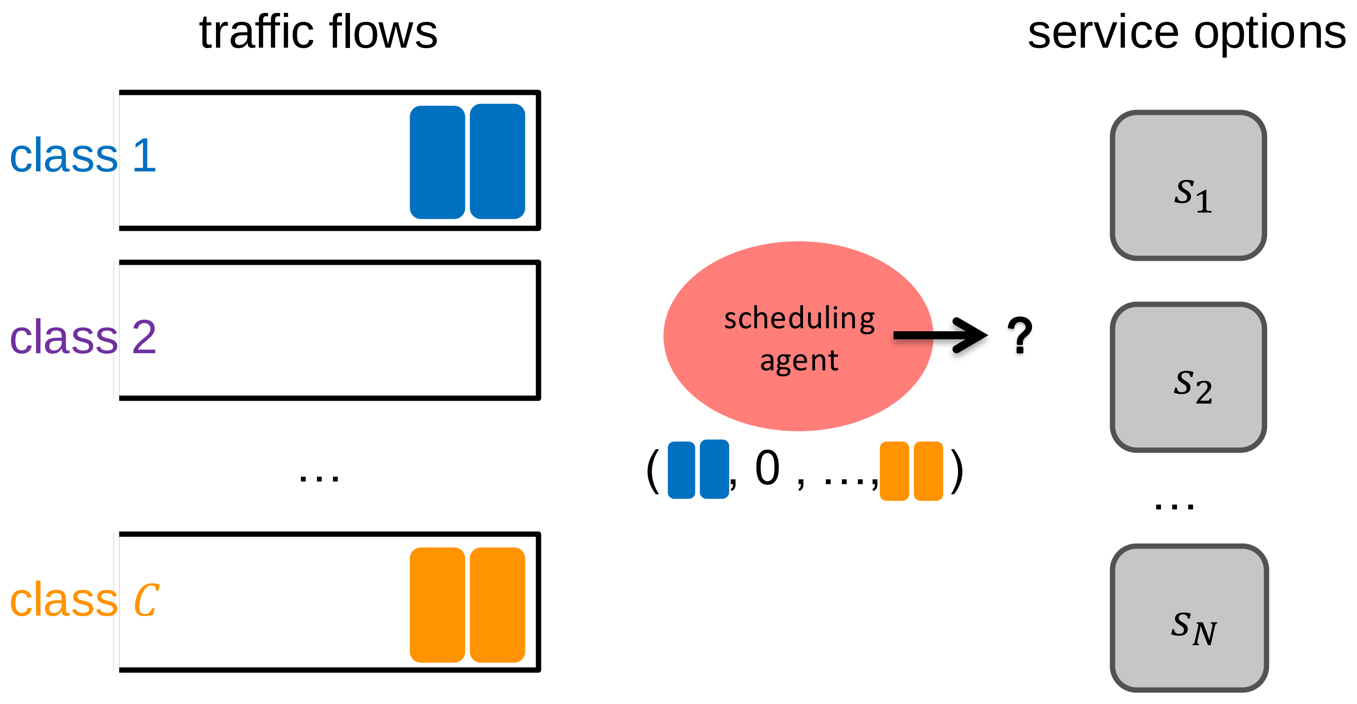



Scheduling agent :

- observes the global state of the queues,
- can infer service rates $R_{c,s}(t)$ from evolution of queue lengths, but does not have any advance knowledge of realizations or underlying statistics

→ in stark contrast to conventional assumptions.

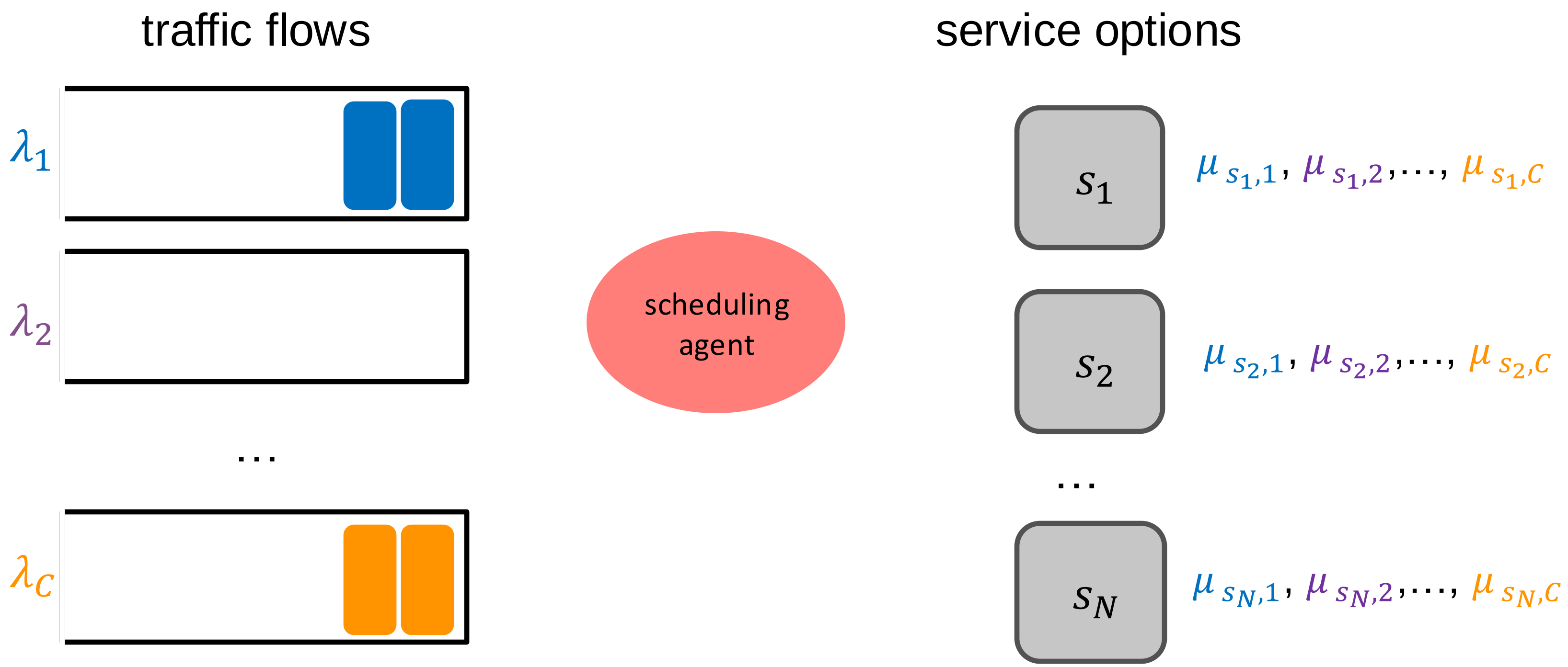
# 1. MODEL DESCRIPTION




**Objective:**
Design scheduling algorithm that

- achieves maximum stability (throughput optimality), and
- provides (near-)optimal response times.

# 1. MODEL DESCRIPTION




For analysis purposes, we assume that

- the number of jobs that arrive per time slot and class ~ Geometric with mean $\lambda_c$ for class $c$.
- the number of served jobs of class $c$ at service option $s$ ~ Geometric with mean $\mu_{s,c}$.

# 2. STABILITY REGION

**Stability region:** Given the set of mean arrival rates $(\lambda_1, \lambda_2, \dots, \lambda_C)$ and mean service rates $\mu_s = (\mu_{s,1}, \mu_{s,2}, \dots, \mu_{s,C})$ for service option $s$. There exists a vector

$$(\overline{\lambda_1}, \overline{\lambda_2}, \dots, \overline{\lambda_C}) \in convex\ hull\ (\mu_1, \mu_2, \dots, \mu_N)$$

such that $(\lambda_1, \lambda_2, \dots, \lambda_C) < (\overline{\lambda_1}, \overline{\lambda_2}, \dots, \overline{\lambda_C})$ component-wise.

# 2. STABILITY REGION

**Stability region:** Given the set of mean arrival rates $(\lambda_1, \lambda_2, \dots, \lambda_C)$ and mean service rates $\mu_s = (\mu_{s,1}, \mu_{s,2}, \dots, \mu_{s,C})$ for service option $s$. There exists a vector

$$(\overline{\lambda_1}, \overline{\lambda_2}, \dots, \overline{\lambda_C}) \in convex\ hull\ (\mu_1, \mu_2, \dots, \mu_N)$$

such that $(\lambda_1, \lambda_2, \dots, \lambda_C) < (\overline{\lambda_1}, \overline{\lambda_2}, \dots, \overline{\lambda_C})$ component-wise.

**The above stability condition …**

- is **necessary** for all algorithms that do not have advance knowledge of the realizations of the service rates $R_{c,s}(t)$ and
- **sufficient** for the algorithm that we will propose.

→ **maximum stability** for our algorithm.

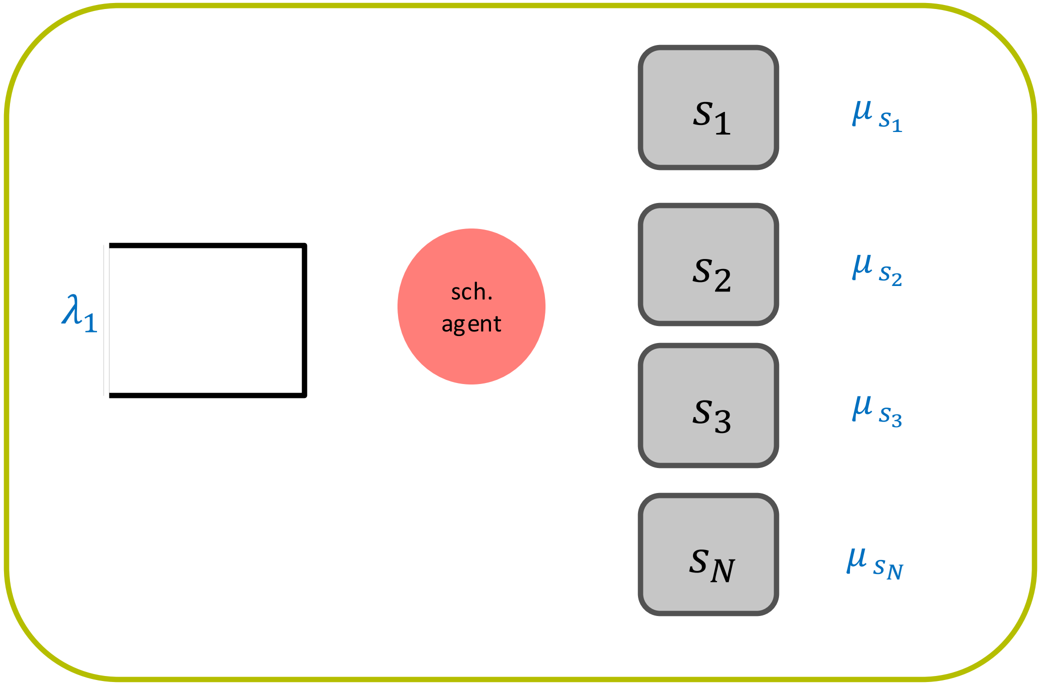
**The above stability condition …**

- is **not** necessary in case of scheduling algorithms that do have advance knowledge of the realizations of service rates (channel-aware, `opportunistic').

# 3. STABILITY REGION SINGLE CLASS

**Proposition :** Consider the system with a single traffic class and $N$ service options with service rates $\mu_s$ for service option $s$. The system is stable if the following holds:

$$\lambda_1 < \max\{\mu_1, \mu_2, \dots, \mu_N\}.$$



# 3. STABILITY REGION SINGLE CLASS

**Proposition :** Consider the system with a single traffic class and $N$ service options with service rates $\mu_s$ for service option $s$. The system is stable if the following holds:

$$\lambda_1 < \max\{\mu_1, \mu_2, \dots, \mu_N\}.$$

**Algorithm :**

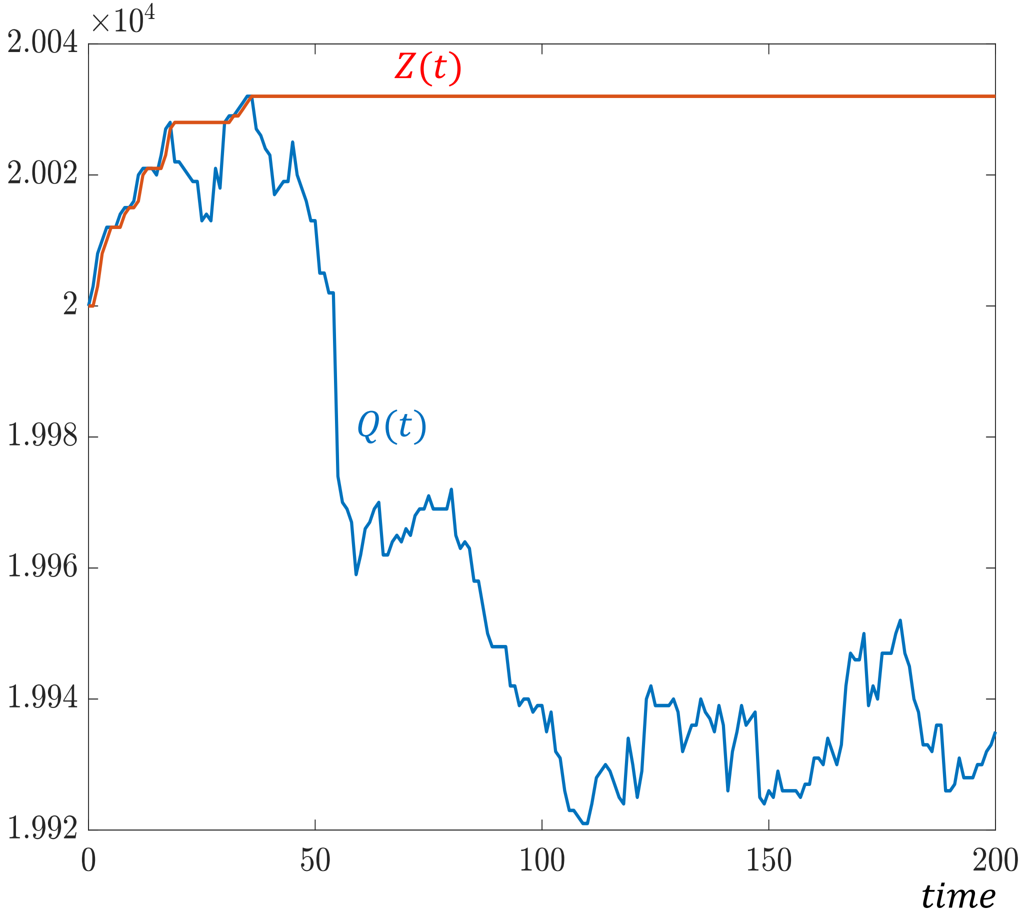
- $Q(t) =$ the number of jobs in queue at time $t$ with

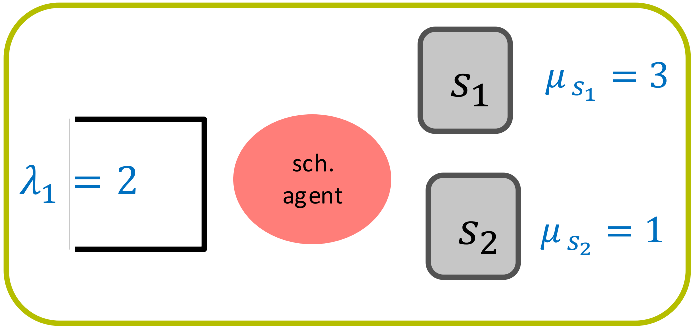
$$Q(t) = Q(t-1) + A(t) - R_{S(t)}(t),$$

  and $A(t)$ and $R_{S(t)}(t)$, number of arrivals and departures when the service option is $S(t)$.

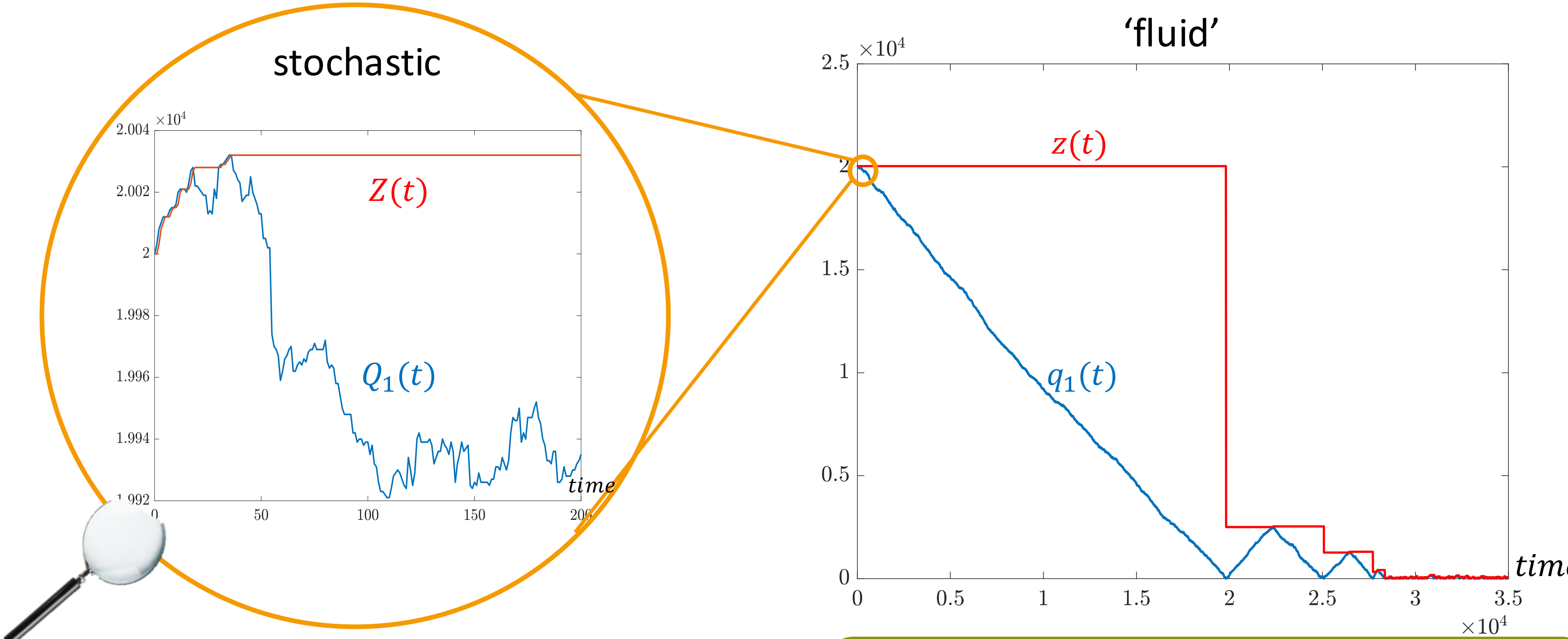
- Fix $Z(0) = Q(0)$ the threshold value : for every $t > 0$:
  - If $Z(t-1) \geq Q(t-1)$ :
    $Z(t) = Z(t-1)$ and $S(t) = S(t-1)$.
  - If $Z(t-1) < Q(t-1)$ or $\mathrm{Q(t)} = 0$:
    $Z(t) = \frac{Z(t-1)}{2}$ and $S(t) \sim Unif(serv.opt)$

# 3. STABILITY REGION SINGLE CLASS



# 3. STABILITY REGION SINGLE CLASS




**Challenges in the stochastic process:**

- The number of times to sample a feasible service option with 'good' trajectory is unbounded, as well as the amount that the threshold value increases.

**Improvement in the fluid limit:**

- Vanish in the limit.

**Challenges in fluid limit:**

- Even if the fluid hits 0 once, then will increase again, but this is bounded.

# 4. STABILITY REGION TWO CLASSES

**Proposition :** Consider the system with a single traffic class with arrival rates $(\lambda_1, \lambda_2)$, and $N$ service options with service rates $\mu_s = (\mu_{s,1}, \mu_{s,2})$ for service option $s$. There exists a vector

$$\left(\overline{\lambda_1}, \overline{\lambda_2}\right) \in convex\ hull\ (\mu_1, \mu_2, \dots, \mu_N)$$

such that $(\lambda_1, \lambda_2) < \left(\overline{\lambda_1}, \overline{\lambda_2}\right)$ component-wise.

# 4. THRESHOLD BASED SCHEDULING ALGORITHM

**Queue dynamics :**

- $Q_c(t)$ = the number of class $c$ jobs in queue at time $t$ with

$$Q_c(t) = Q_c(t-1) + A_c(t) - R_{c,S(t)}(t),$$

and $A_c(t)$ and $R_{c,S(t)}(t)$, number of arrivals and departures of class $c$ when the service option is $S(t)$.

- $L(t) = \sum_{c=1}^{C} Q_c^2(t)$.

- $Z(t)$ = the threshold value at time $t$, with fix $Z(0) = L(0) = \sum_{c=1}^{C} Q_c^2(0)$.

- $(X_1(t), X_2(t))$ = the queue length per class of the threshold value a time $t$, with fix $(X_1(0), X_2(0)) = (Q_1(0), Q_2(0))$.

# 4. THRESHOLD BASED SCHEDULING ALGORITHM

**Algorithm :**

- At each time slot $t > 1$:
  - If $Z(t-1) \geq L(t)$:

    keep going : $Z(t) = Z(t-1)$ ,
    $(X_1(t), X_2(t)) = (X_1(t-1), X_2(t-1))$,
    $S(t) = S(t-1)$.

  - If $Z(t-1) < L(t)$ :
    Update: $Z(t) = L(t)$
    $(X_1(t), X_2(t)) = (Q_1(t), Q_2(t))$
    $S(t) \sim Unif(service\ option)$

    - fix $\sigma$small lower bound for $Z(t)$.

  - If $min_{c \in C}\{Q_c(t)\} = 0$ and $max_{c \in C}\{Q_c(t)\} \geq X_{argmax_{c \in C}\{Q_c(t)\}}(t)$ :
    Update: $Z(t) = L(t)$
    $(X_1(t), X_2(t)) = (Q_1(t), Q_2(t))$

# 4. THRESHOLD BASED SCHEDULING ALGORITHM

## Algorithm :

- At each time slot $t > 1$:
  - If $Z(t-1) \geq L(t)$:

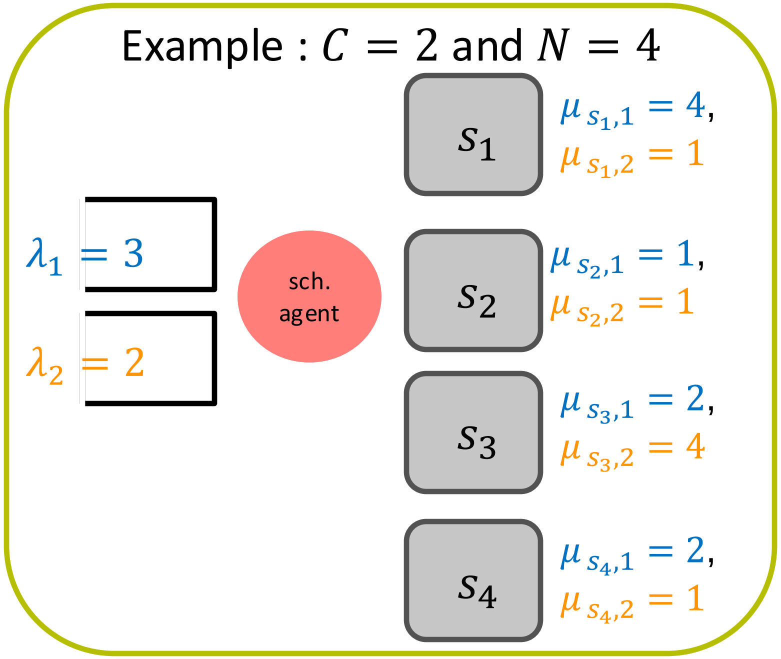
    keep going : $Z(t) = Z(t-1)$ ,
    $(X_1(t), X_2(t)) = (X_1(t-1), X_2(t-1))$,
    $S(t) = S(t-1)$.

  - If $Z(t-1) < L(t)$ :
    Update: $Z(t) = L(t)$
    $(X_1(t), X_2(t)) = (Q_1(t), Q_2(t))$
    $S(t) \sim Unif(service\ option)$

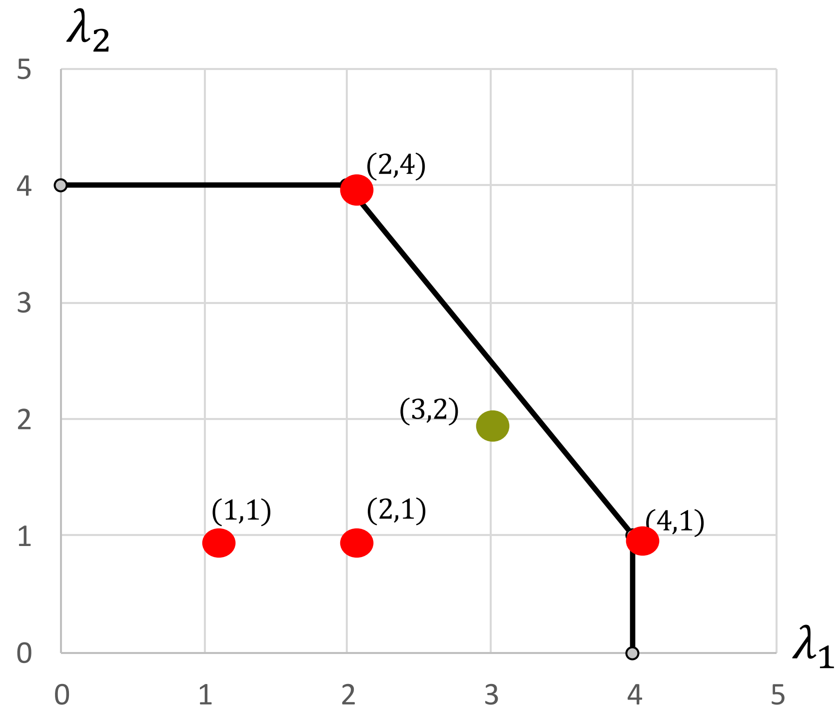
    - fix $\sigma$small lower bound for $Z(t)$.

  - $min_{c \in C}\{Q_c(t)\} = 0$ and $max_{c \in C}\{Q_c(t)\} \geq X_{argmax_{c \in C}\{Q_c(t)\}}(t)$ :
    Update: $Z(t) = L(t)$
    $(X_1(t), X_2(t)) = (Q_1(t), Q_2(t))$

# 4. THRESHOLD BASED SCHEDULING ALGORITHM : EXP

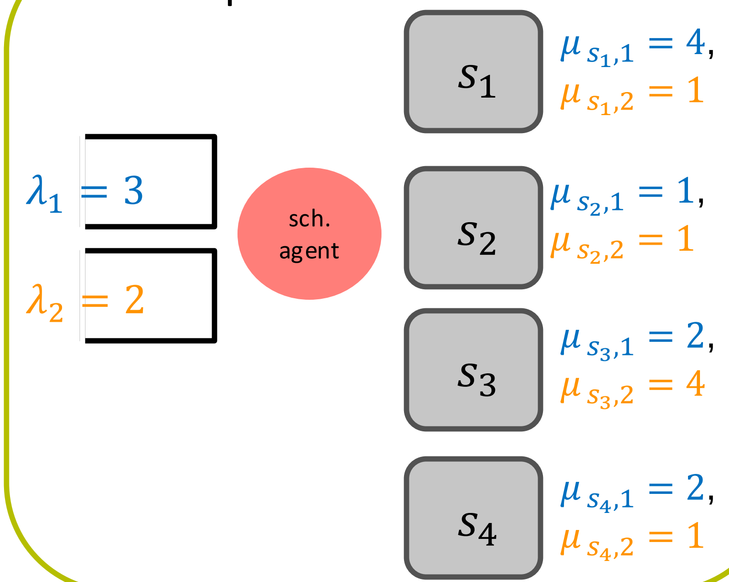



Stability region:



# 4. THRESHOLD BASED SCHEDULING ALGORITHM : EXP

Example : $C = 2$ and $N = 4$

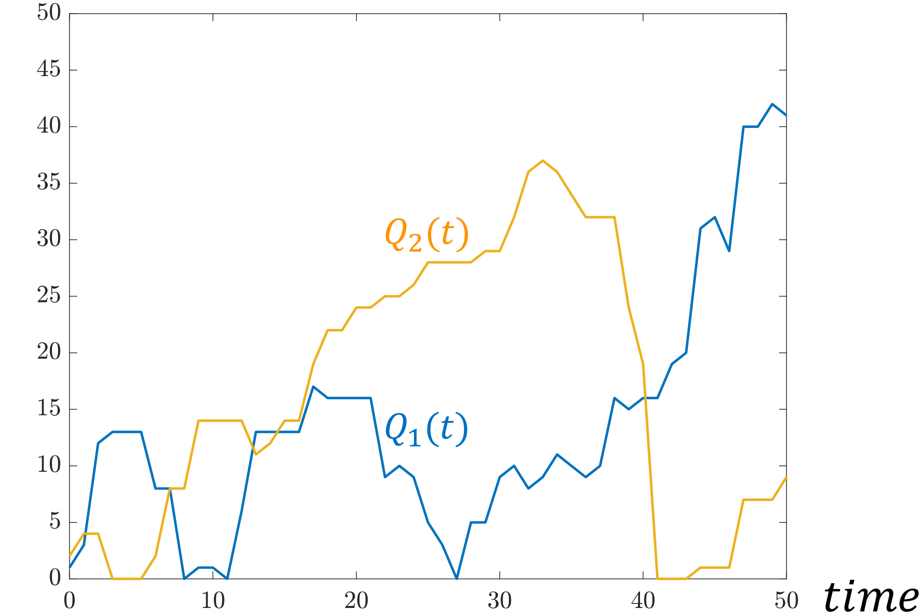


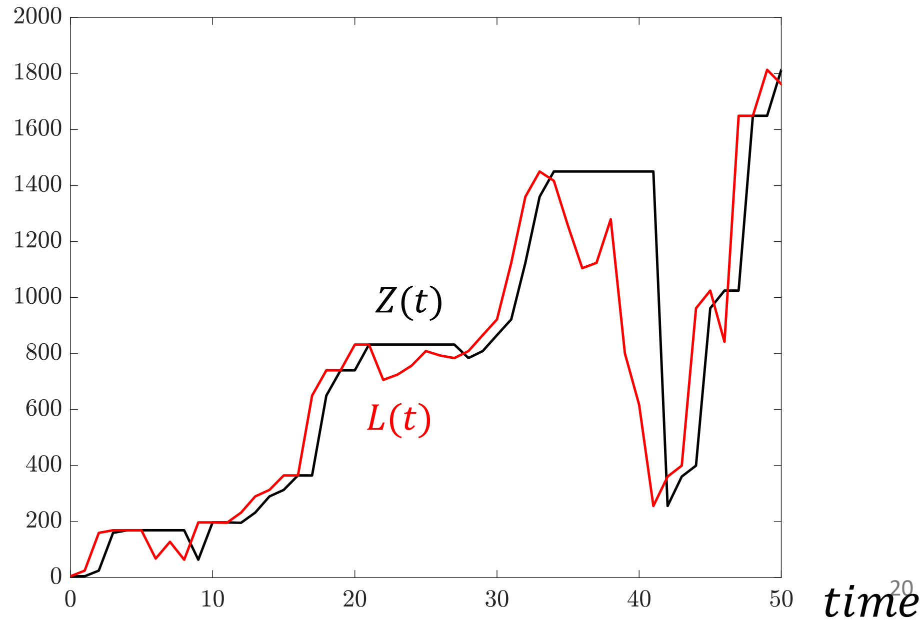

**Sec. of service options :**

**time**

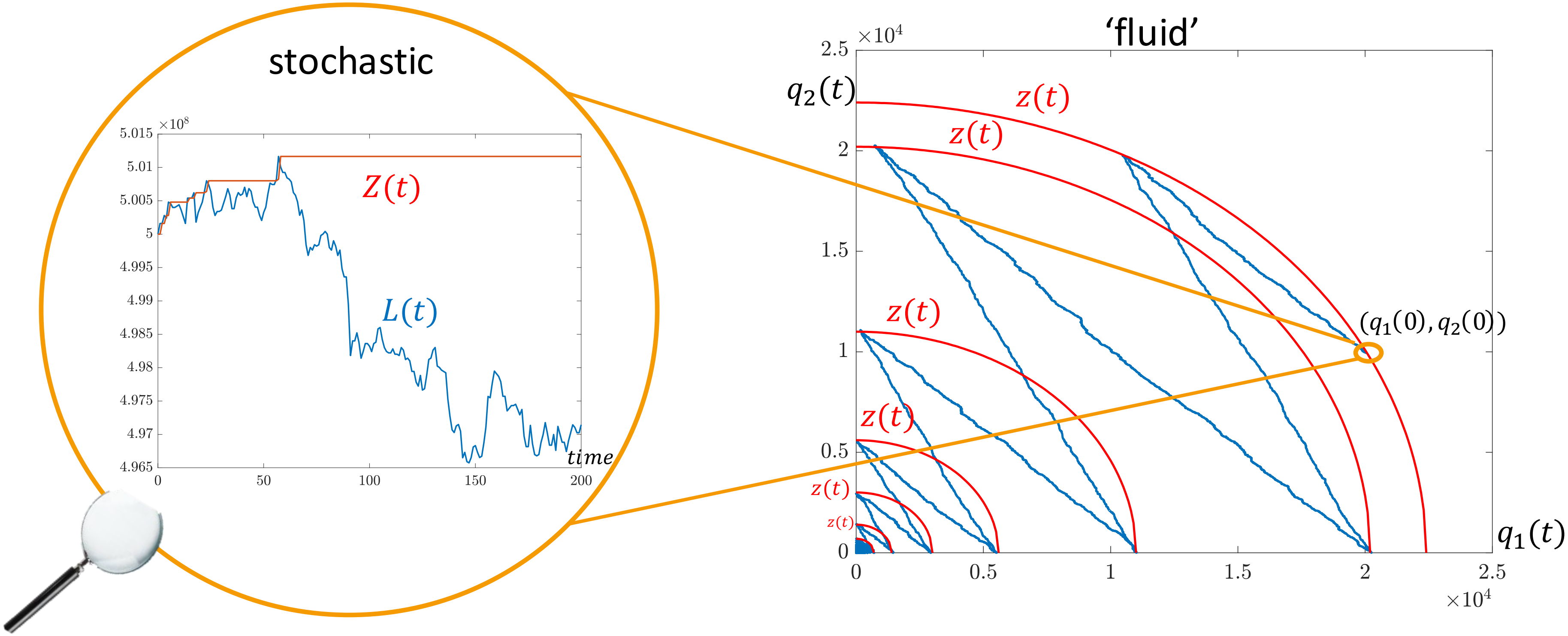
| 0 | 1 | 2 | 3 | 4 | 5 | 6 - 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 2 | 4 | 3 | 4 | 4 | 1 | 4 | 3 | 3 | 2 | 2 |

| 18 | 19 | 20 | 21 | 22 | 23-27 | … |
|---|---|---|---|---|---|---|
| 2 | 2 | 1 | 2 | 2 | 1 | … |



# 4. STABILITY REGION TWO CLASSES



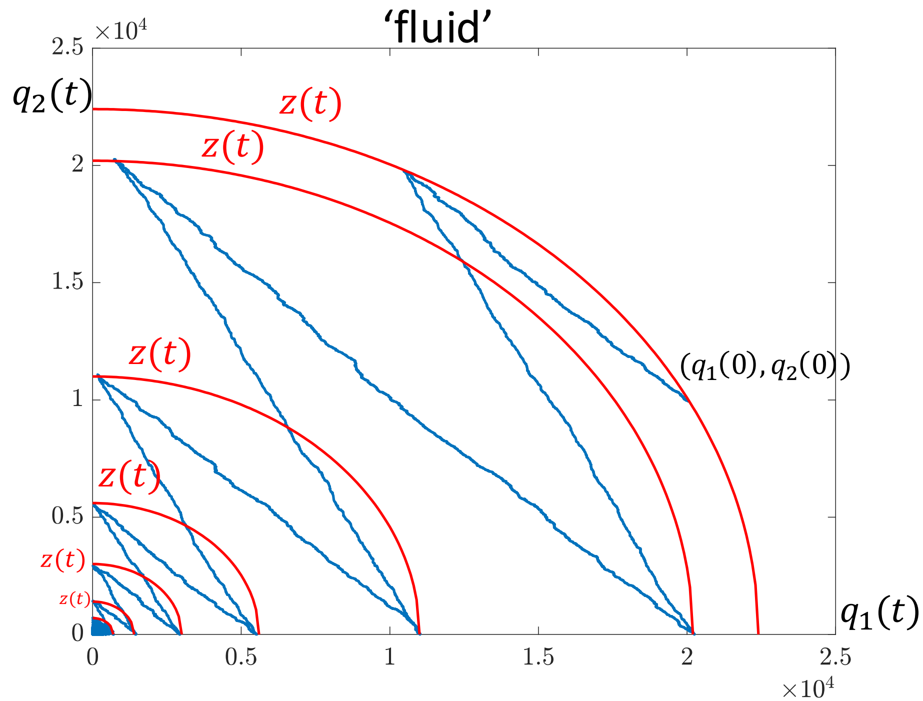

**Challenges in the stochastic process:**

- The number of times to sample a feasible service option with 'good' trajectory is unbounded, as well as the amount that the threshold value increases.

**Improvement in the fluid limit:**

- Vanish in the limit.

# 4. STABILITY REGION TWO CLASSES




**Additional challenges in the fluid:**

- The feasible service option is dependent of the per class number of jobs.
- An unpredictable number of 'good' trajectories before the one where z$(t)$ is reduced.

**BUT :**

- There is at least one feasible service option per pair $(q_1, q_2)$.
- The fluid limit lives inside the area determined by z$(t)$ , and this eventually shrinks.

# 5. CONCLUSION AND FUTURE RESEARCH

**Conclusions :**

- We provide an algorithm that is maximally stable, for system that do not have any advance knowledge of the service rates.
- However, `lazy' scheduling by randomly resampling without really acting as long as things seem to move in the right direction.

**Future work :**

- Improve (response time) performance, by using an active learning algorithm that learns to sample the best combination of service options.

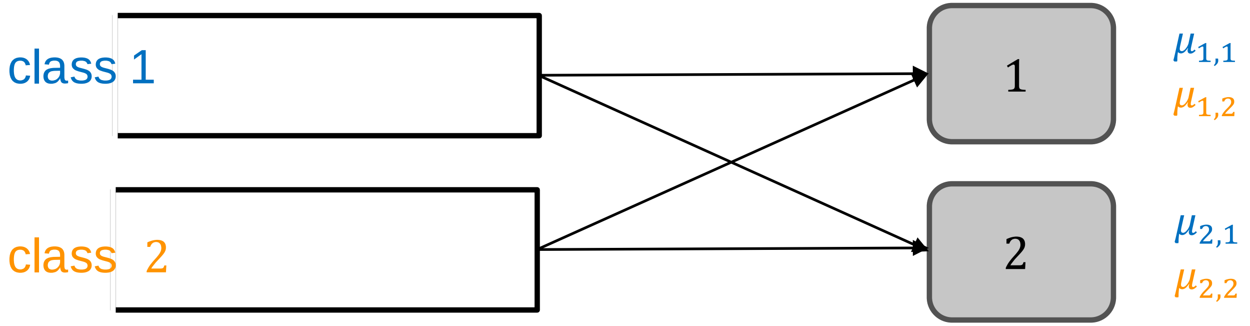
# 5. CONCLUSION AND FUTURE RESEARCH

THANK YOU!

# 5. TWO CLASSES: EXAMPLE X-MODEL

X-model*: 2 traffic classes with 2 service options where either:

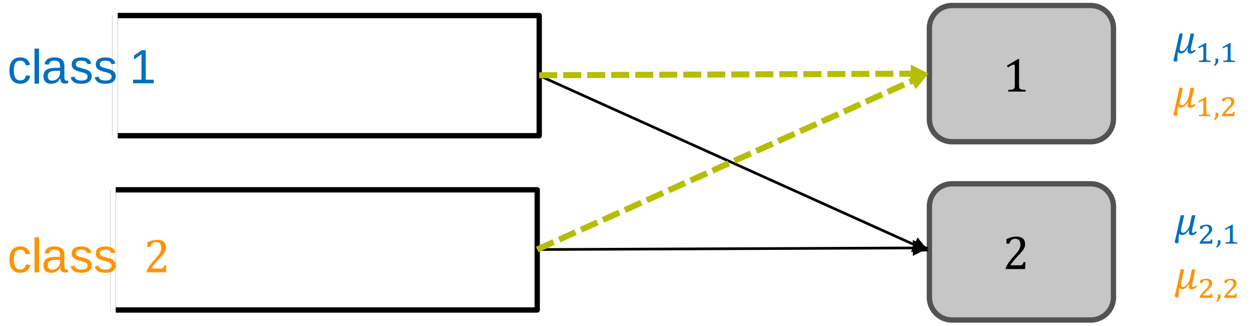
- server 1 (server 2) serves both classes simultaneously or,
- server 1 and server 2 both serve the same class




* Yuan Zhong, *Instability and stability of parameter agnostic policies in parallel server systems, Performance 2023.*

# 5. TWO CLASSES: EXAMPLE X-MODEL

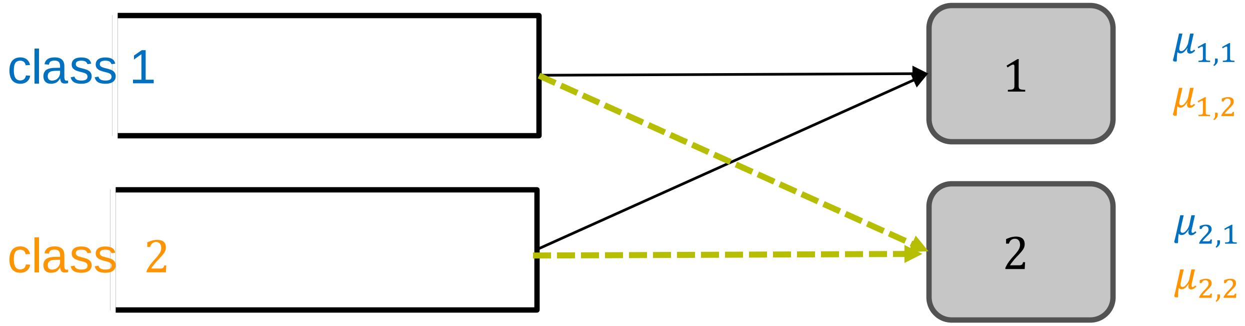
X-model*: 2 traffic classes with 2 service options where either:
- server 1 (server 2) serves both classes simultaneously or,
- server 1 and server 2 both serve the same class



# 5. TWO CLASSES: EXAMPLE X-MODEL

X-model*: 2 traffic classes with 2 service options where either:

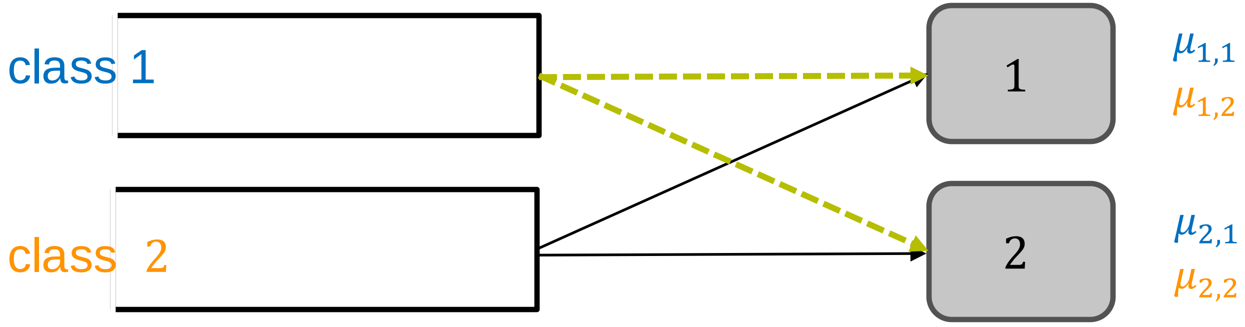
- server 1 (server 2) serves both classes simultaneously or,
- server 1 and server 2 both serve the same class



# 5. TWO CLASSES: EXAMPLE X-MODEL

X-model*: 2 traffic classes with 2 service options where either:

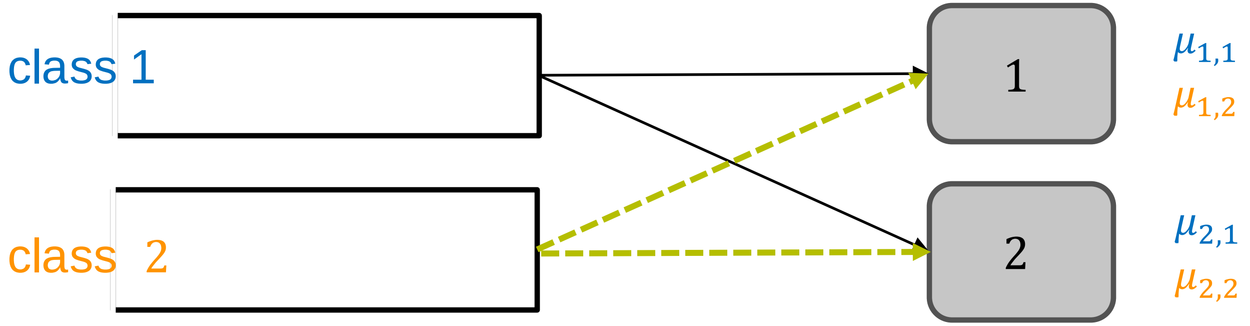
- server 1 (server 2) serves both classes simultaneously or,
- server 1 and server 2 both serve the same class



# 5. TWO CLASSES: EXAMPLE X-MODEL

X-model*: 2 traffic classes with 2 service options where either:

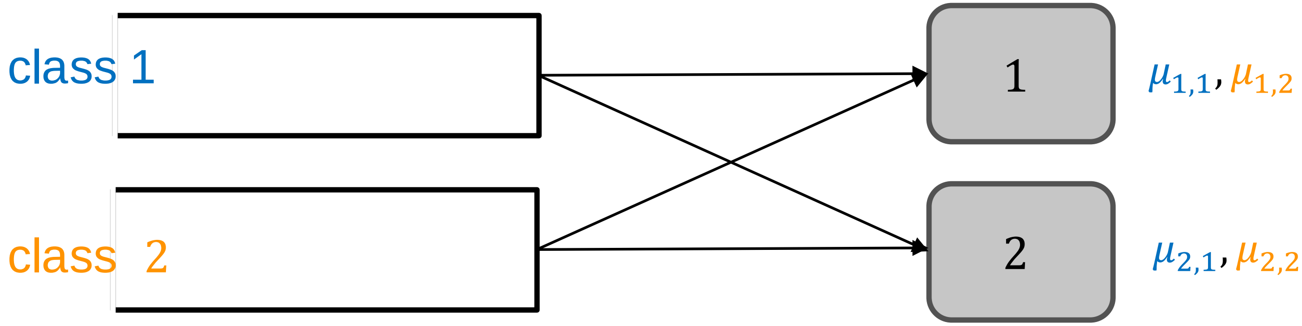
- server 1 (server 2) serves both classes simultaneously or,
- server 1 and server 2 both serve the same class



# 5. TWO CLASSES: EXAMPLE X-MODEL

X-model*: 2 traffic classes with 2 service options where either:

- server 1 (server 2) serves both classes simultaneously or,
- server 1 and server 2 both serve the same class




Can be also modeled by our model: 2 traffic classes with 4 service options