Aalto University
School of Electrical
Engineering

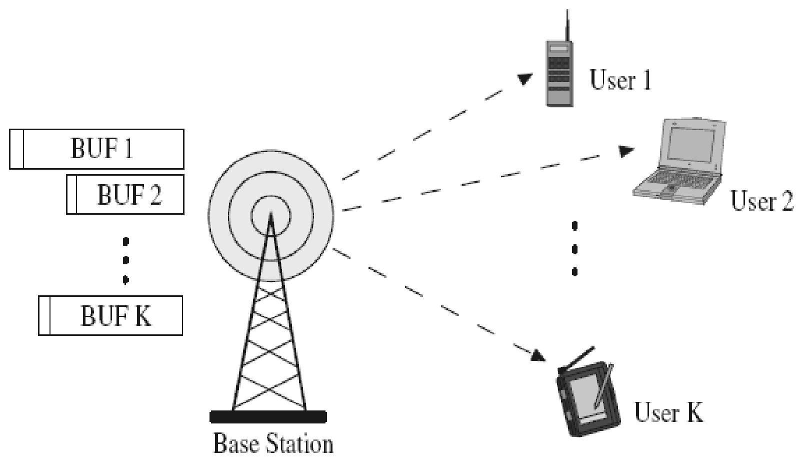
Combining optimally opportunistic and size-based scheduling in scalable queues

Samuli Aalto, Aleksi Penttinen, Pasi Lassila, Prajwal Osti
Aalto University, Finland

INFORMS APS
6–8 July 2011
Stockholm, Sweden

Research problem

- Downlink data transmission in a wireless cellular system
- Traffic = elastic flows
  - file transfers using TCP
- Scheduling decisions in each time slot
  - time scale of milliseconds
- Traffic dynamics in a much longer time scale
  - time scale of seconds/minutes
- Optimal time-slot-level scheduler for flow-level performance?



Source: Hu et al. (2004)

Flow-level performance

- Performance is expressed as flow-level delay
  - Mean flow delay describes
    how long, on the average, it takes to transfer a file

- Importance of the time scale
  - Users do not care about time-slot or packet-level delays,
    but the flow-level delay, i.e., the total time to transmit a file

Time-slot-level schedulers

- Channel-aware schedulers
  - Channel conditions varying randomly for each user
  - Scheduling based on channel information
  - Scheduler may prefer users with a good channel
  - Opportunistic scheduling
  - Examples: MR, PF
- Size-based schedulers
  - Scheduling based on flow size information
  - Scheduler may prefer users with a short flow
  - Example: SRPT
  - Schrage (1968): SRPT optimal in the M/G/1 queue

Fundamental trade-off

- Opportunistic scheduling
  - Aggregate mean service rate increases with the number of users (opportunistic gain, multiuser diversity gain)
  - However, a user with a long remaining service requirement blocks the other users
- SRPT
  - The number of flows is reduced most efficiently
  - However, opportunistic gain is lost due to suboptimal channel (later on also due to a smaller number of flows)

Combining opportunistic and size-based scheduling

- Tsybakov (2003)
  - Dynamic programming approach (time-slot scale)
- Hu et al. (2004)
  - Heuristic approach: TAOS (time-slot scale)
- Lassila and Aalto (2008)
  - Another heuristic approach: SRPT-P (time-slot scale)
- Ayesta et al. (2010)
  - Age-based information, Markovian system (time-slot scale)
- Sadiq and de Veciana (2010)
  - Time-scale separation (flow scale)
  - Transient system
  - Optimality result for nested polymatroids
  - Cf. optimality of SRPT-FM, Raj et al. (2004)

Time-scale separation: From the time-slot scale to the flow scale

- $R(t) = (R_1(t),\ldots, R_k(t))$ = rate vector in time slot t
- $R_i(t)$ = instantaneous rate of user i
- Assume: $R_i(t)$ is a stationary and ergodic process
- Assume: Scheduling policy $\pi \in \Pi_k$ is stationary
- Define: The long-term throughput for user i:

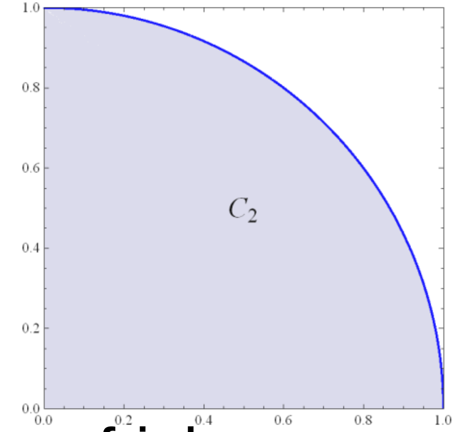
$$\theta_i^{\pi} = \sum_{\mathbf{r}} r_i \, p_i^{\pi}(\mathbf{r}) P\{R(t) = \mathbf{r}\}$$

- Define: The (opportunistic) capacity region:

$$C_k = \{(\theta_1^{\pi},\ldots,\theta_k^{\pi}) \in \Re_+^k : \pi \in \Pi_k\}$$

Part 1
Optimal scheduling in scalable queues

Scalable queue



- Service system where the service capacity is self-scalable depending on the current number of jobs
- When there are k jobs with sizes

$$s_1 \geq \ldots \geq s_k$$

choose a rate vector

$$\mathbf{c}_k = (c_{k1}, \ldots, c_{kk}) \in C_k$$

and serve job i with rate c_{ki}

- Assume: Capacity regions C_k compact and symmetric

Optimal scheduling problem (transient system)

- Assume that there are n jobs in the system at time 0
- What is the optimal way to make the system empty?
- Objective: Minimize the mean delay (or flow time)
- Define: Flow time (or total completion time) for policy ϕ

$$T^{\phi} = \sum_{i=1}^{n} t_i^{\phi}$$

where t_i is the completion time of job i

- Define: Operating policies

$$\Phi_n = \{\phi = (\mathbf{c}_1, \ldots, \mathbf{c}_n) : \mathbf{c}_k \in C_k \text{ for all } k\}$$

Trivial case: One job

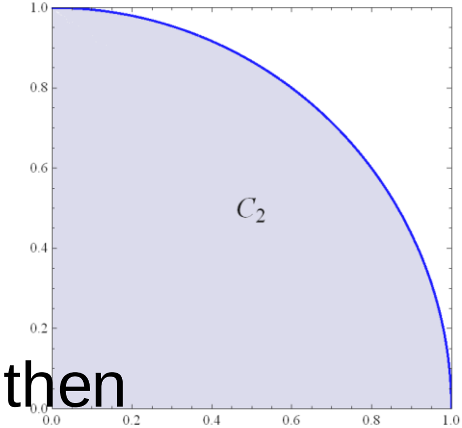
- Define:

$$G_1^* = \frac{1}{c_1^*}, \qquad c_1^* = \max_{c_1 \in C_1} c_1$$

- Now

$$T^* = \min_{\phi \in \Phi_1} T^{\phi} = s_1 G_1^*, \quad \phi^* = (\mathbf{c}_1^*)$$

Simple case: Two jobs



- If job 2 (i.e., the shorter one) completes first, then

$$T^{\phi} = 2\frac{s_2}{c_{22}} + (s_1 - \frac{s_2}{c_{22}} c_{21}) \frac{1}{c_1^*} = \frac{s_2}{c_{22}} (2 - \frac{c_{21}}{c_1^*}) + \frac{s_1}{c_1^*}$$

- Otherwise

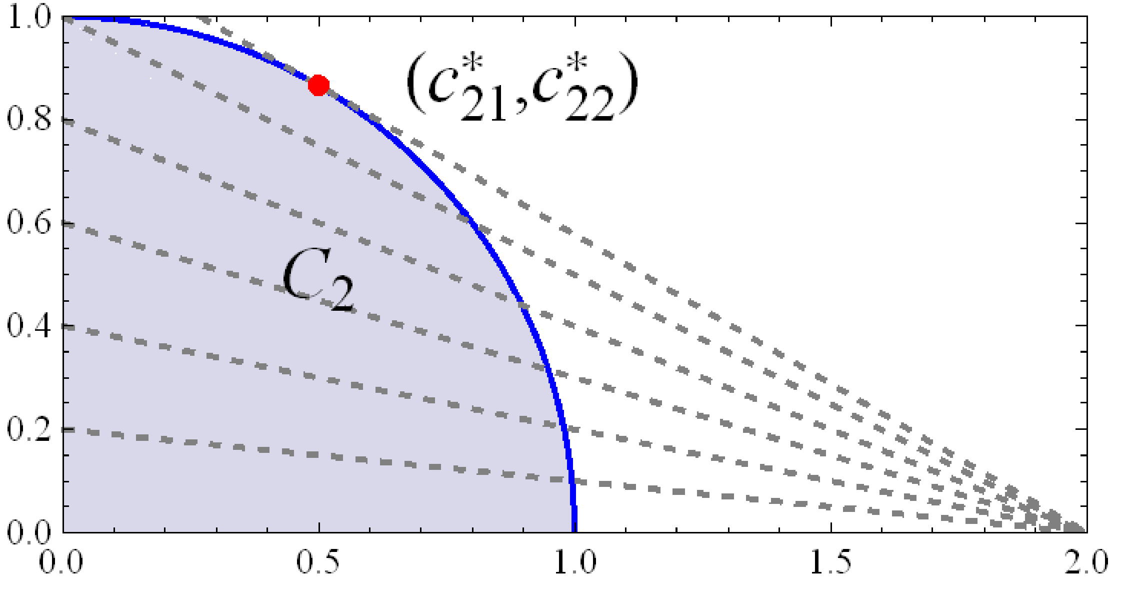
$$T^{\phi} = 2\frac{s_1}{c_{21}} + (s_2 - \frac{s_1}{c_{21}} c_{22}) \frac{1}{c_1^*} = \frac{s_1}{c_{21}} (2 - \frac{c_{22}}{c_1^*}) + \frac{s_2}{c_1^*}$$

- Let us minimize (a function not depending on sizes!)

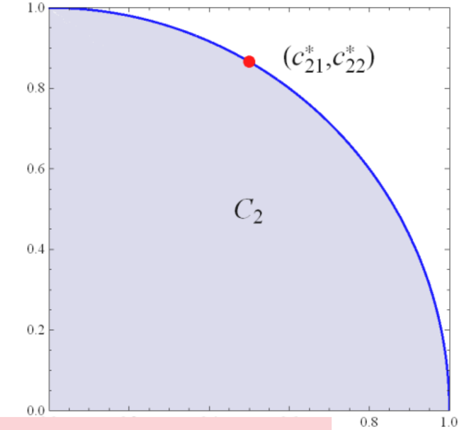
$$g(\mathbf{c}_2) = \frac{1}{c_{22}} (2 - \frac{c_{21}}{c_1^*}), \quad \mathbf{c}_2 \in C_2$$

Simple case: Two jobs (cont.)

- Geometric interpretation



Simple case: Two jobs (cont.)



- Define:

$$G_2^* = g(\mathbf{c}_2^*) = \min_{\mathbf{c}_2 \in C_2} g(\mathbf{c}_2)$$

- Result: Now **if**

$$G_1^* < G_2^*$$

then (due to the symmetry property!)

$$T^* = \min_{\phi \in \Phi_2} T^{\phi} = s_2 G_2^* + s_1 G_1^*, \ \phi^* = (\mathbf{c}_1^*, \mathbf{c}_2^*), \ c_{21}^* \le c_{22}^*$$

Simple case: Two jobs (cont.)

- Justification:

$$
\begin{aligned}
T^{\phi} &\geq \min\{s_2 g(c_{21}, c_{22}) + s_1 G_1^*, s_1 g(c_{22}, c_{21}) + s_2 G_1^*\} \\
&\geq \min\{s_2 G_2^* + s_1 G_1^*, s_1 G_2^* + s_2 G_1^*\} \\
&= s_2 G_2^* + s_1 G_1^* \qquad [\text{since } G_2^* > G_1^*] \\
T^{\phi^*} &= s_2 g(c_{21}^*, c_{22}^*) + s_1 G_1^* \qquad [\text{since } c_{22}^* \geq c_{21}^*] \\
&= s_2 G_2^* + s_1 G_1^*
\end{aligned}
$$

Simple case: Two jobs (cont.)

- Required additional result:

$$\frac{1}{c_{22}^*}\left(2-\frac{c_{21}^*}{c_1^*}\right) \le \frac{1}{c_{21}^*}\left(2-\frac{c_{22}^*}{c_1^*}\right) \Leftrightarrow$$

$$c_{21}^*\left(2-\frac{c_{21}^*}{c_1^*}\right) \le c_{22}^*\left(2-\frac{c_{22}^*}{c_1^*}\right) \Leftrightarrow$$

$$(c_{22}^* - c_{21}^*)\left(2-\frac{c_{21}^*}{c_1^*}-\frac{c_{22}^*}{c_1^*}\right) \ge 0 \Leftrightarrow$$

$$c_{22}^*(c_{22}^* - c_{21}^*)(G_2^* - G_1^*) \ge 0$$

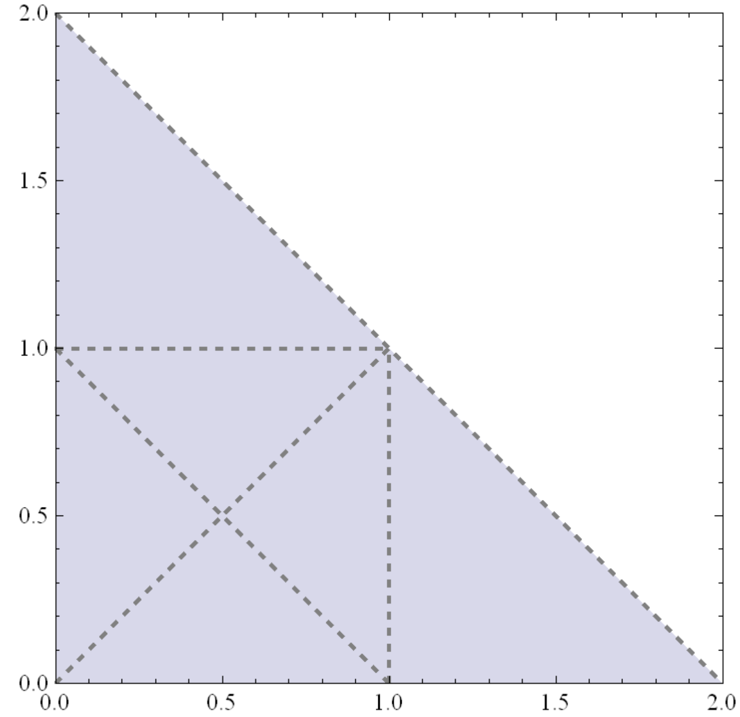
Simple case: Two jobs (cont.)

- Equivalent condition:

$$G_2^* > G_1^* \quad \Leftrightarrow$$

$$c_{21} + c_{22} < 2 \cdot c_1^*$$

- Suffient condition: nested capacity regions
- Note: However, capacity regions are not required to be nested



General case: n jobs

- Define (recursively):

$$G_k^* = \min_{\mathbf{c}_k \in C_k} g_k(\mathbf{c}_k), \quad g_k(\mathbf{c}_k) = \frac{1}{c_{kk}}\left(k - \sum_{i=1}^{k-1} c_{ki} G_i^*\right)$$

- Theorem 1: **If**

$$G_1^* < \ldots < G_n^*$$

then

$$T^* = \min_{\phi \in \Phi_n} T^\phi = \sum_{k=1}^{n} s_k G_k^*, \quad \phi^* = (\mathbf{c}_1^*, \ldots, \mathbf{c}_n^*)$$

General case: n jobs (cont.)

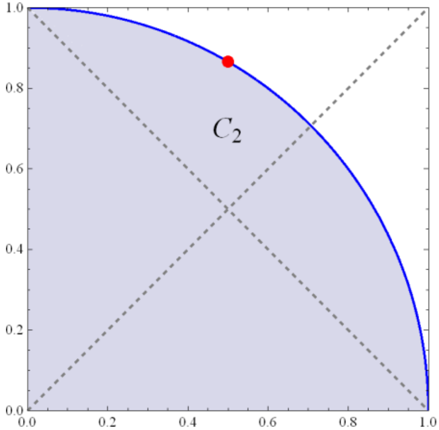
- In addition,

$$c^*_{k1} \leq \ldots \leq c^*_{kk} \quad \text{for all } k$$

- Thus, the optimal policy applies the SRPT-FM principle:
  - the shortest job is served with the highest rate,
  - the second shortest job is served with the second highest rate,
  - etc.

- Note also that the optimal rate vector does not depend on the absolute sizes (only on their order)

Example: Alpha-ball



- Let $\alpha \geq 1$ and consider capacity regions

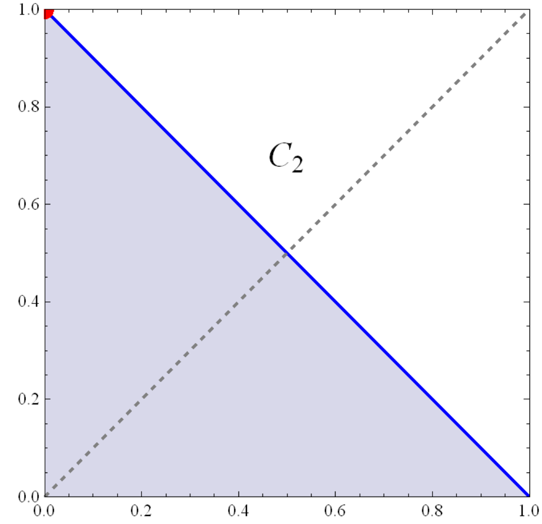
$$C_k = \{\mathbf{c}_k \geq 0 : \sum_{j=1}^{k} c_{kj}^{\alpha} \leq 1\}$$

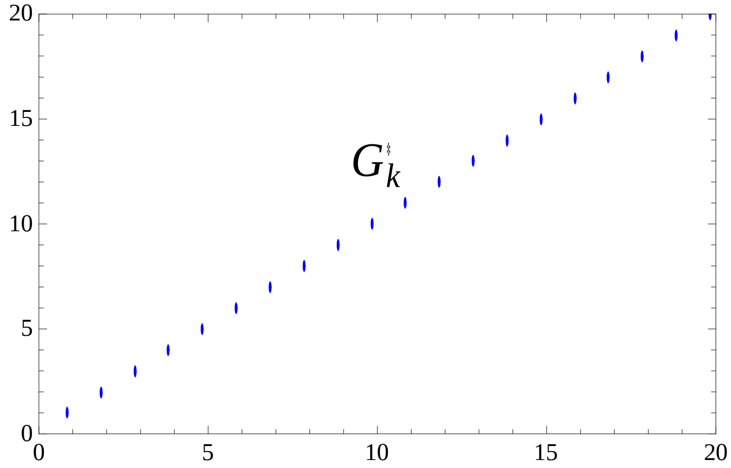
- Now

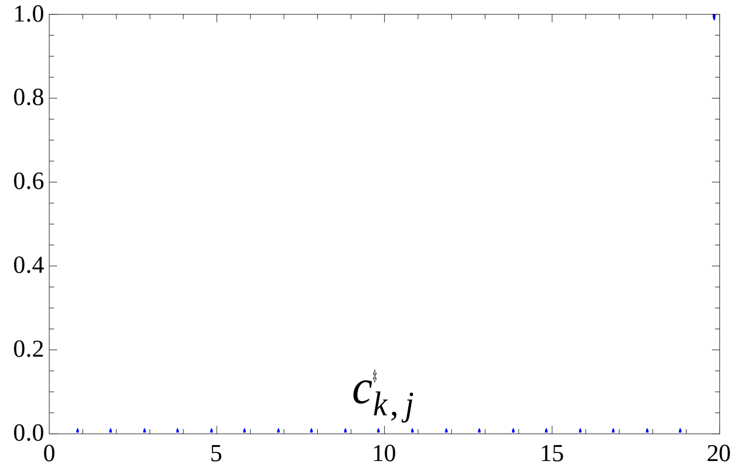
$$G_k^* = \left(k^{\frac{\alpha}{\alpha-1}} - (k-1)^{\frac{\alpha}{\alpha-1}} \right)^{\frac{\alpha-1}{\alpha}} \quad \text{(increasing in } k\text{)}$$

$$c_{kj}^* = \left(\frac{G_j^*}{k} \right)^{\frac{1}{\alpha-1}} \quad \text{(increasing in } j\text{)}$$

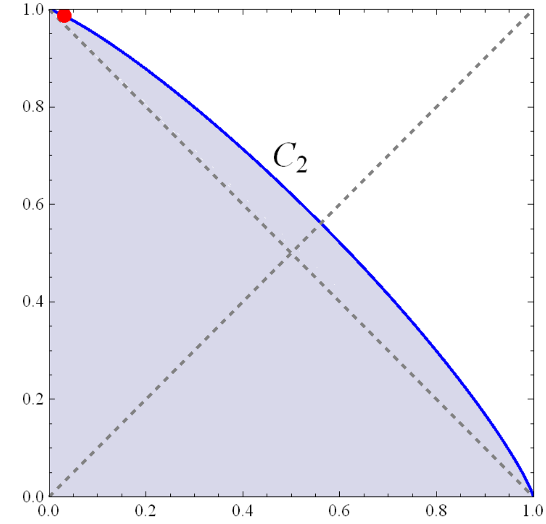
Alpha = 1.0
(single-server queue)



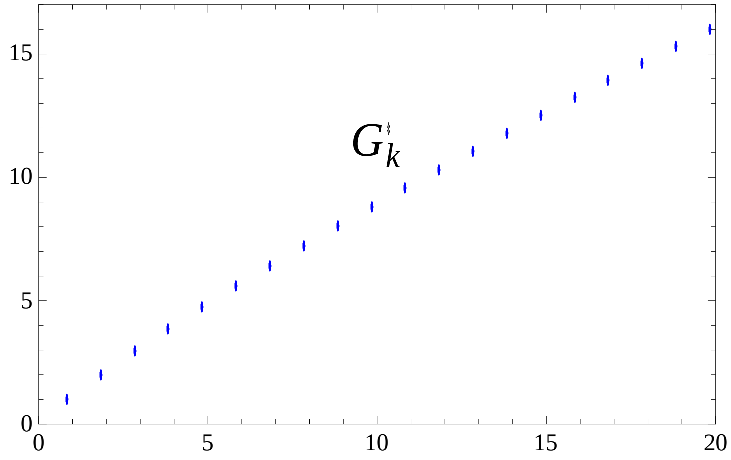




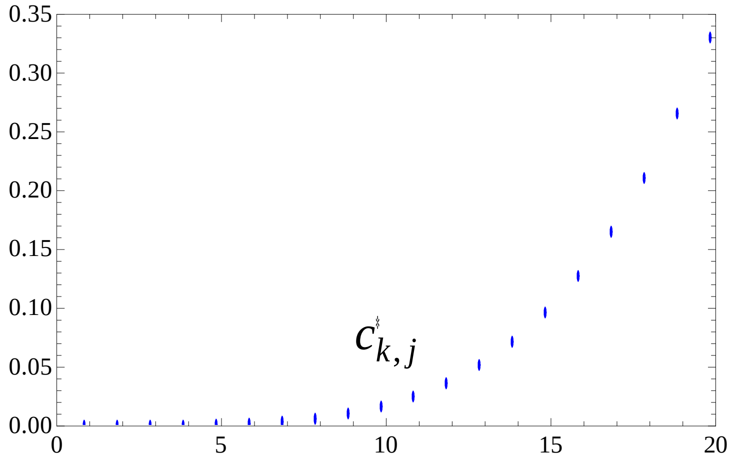
Alpha = 1.2



1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
C_2

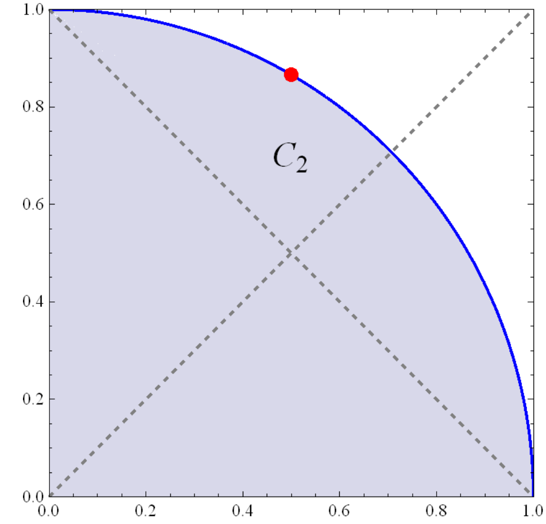


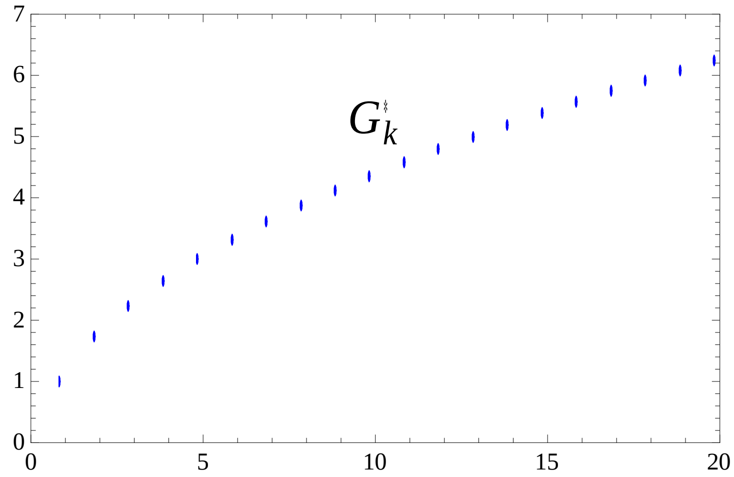
15
10
5
0
0
5
10
15
20
$G_k^{\ddagger}$

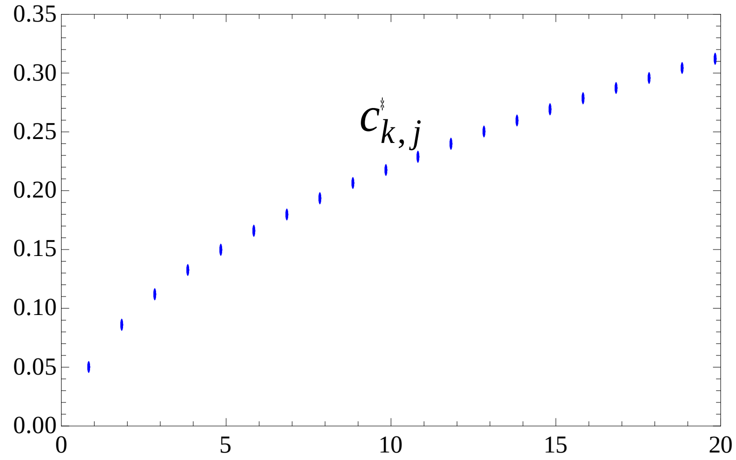


0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
5
10
15
20
$c_{k,j}^{\ddagger}$

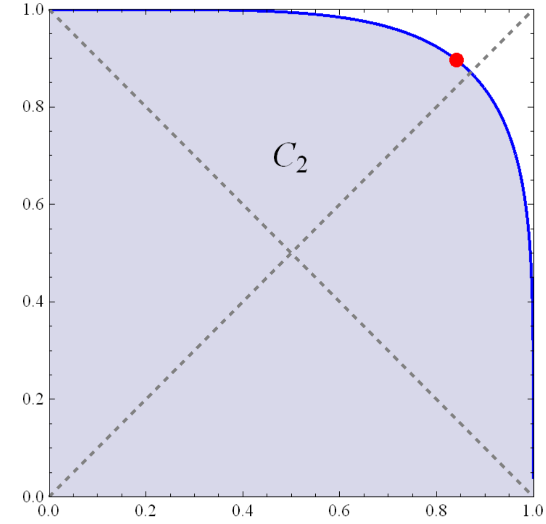
Alpha = 2.0

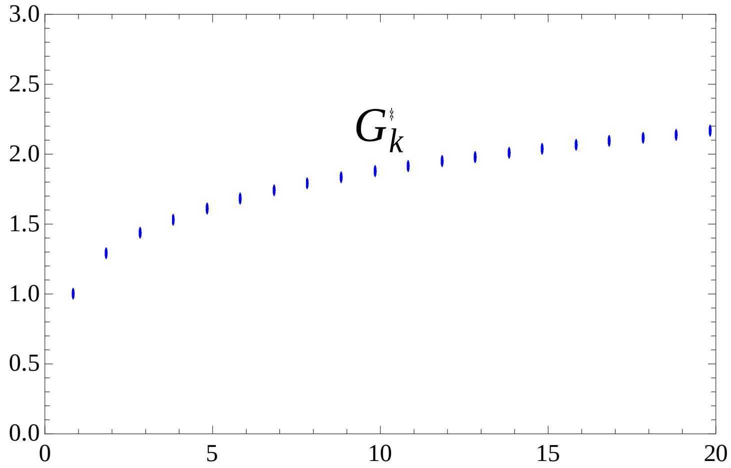


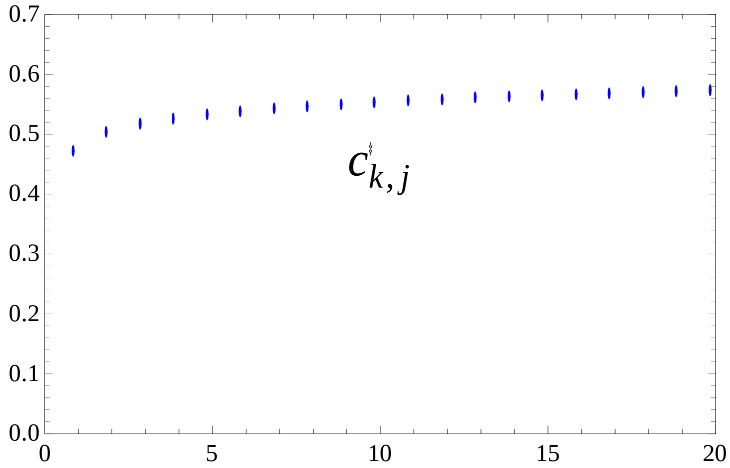




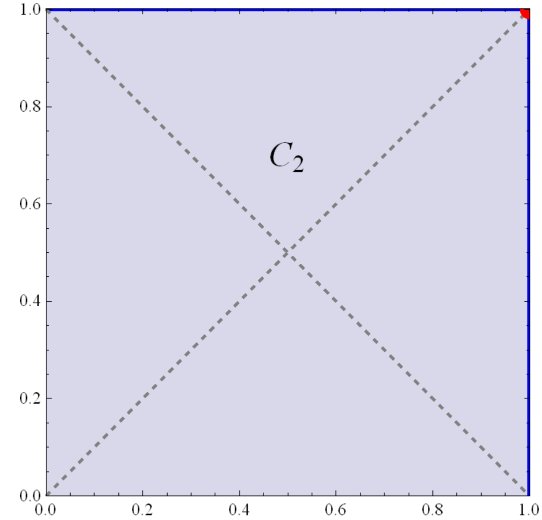
Alpha = 5.0

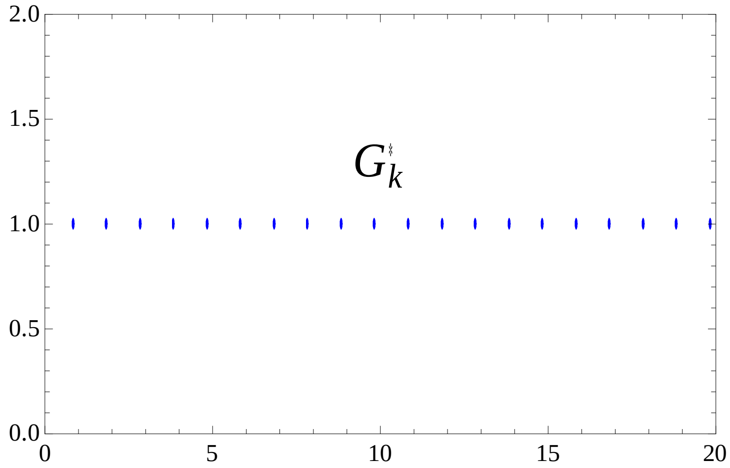

C_2
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

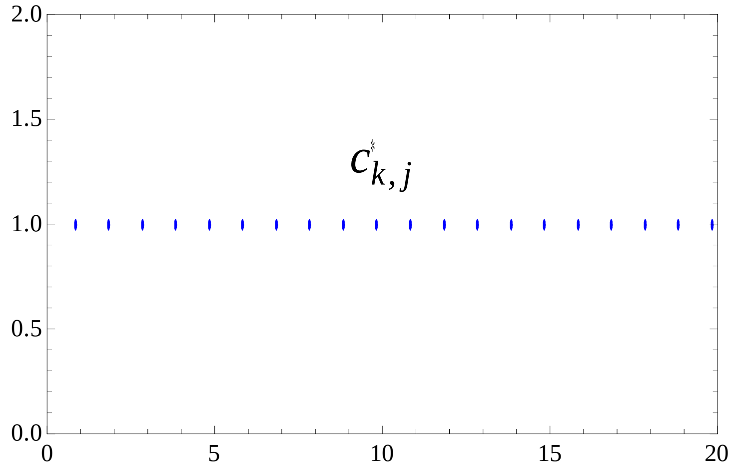

$G_k^\natural$
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
5
10
15
20


$c_{k,j}^\natural$
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
5
10
15
20

Alpha = infinite (infinite-server queue)







Summary of Part 1

- Assumptions:
  - Abstract capacity regions (time-scale separation)
  - Transient system
- Results:
  - Optimality result for compact and symmetric capacity regions
    - includes nested polymatroids (cf. Sadiq and de Veciana (2010))
    - requires an implicit condition related to capacity regions
  - Optimal rate vectors for each phase
    - applying the SRPT-FM principle
- Open questions:
  - Is it possible to make the implicit condition explicit?
  - Is it possible to implement the optimal policy at time-slot scale?

Part 2
Optimal time-slot-level scheduler for the wireless cellular system

Time-slot-level model

- $R(t) = (R_1(t), \ldots, R_k(t))$ = rate vector in time slot t
- $R_i(t)$ = instantaneous rate of user i
- Assume: $R_i(t)$ is a stationary and ergodic process taking values in a finite set
- Assume: Processes $R_i(t)$ are IID

Time-slot-level schedulers

- Assume: Scheduling policy $\pi \in \Pi_k$ is stationary
- Define: The long-term throughput for user i:

$$\theta_i^\pi = \sum_{\mathbf{r}} r_i p_i^\pi(\mathbf{r}) P\{R(t) = \mathbf{r}\}$$

- Define: The (opportunistic) capacity region:

$$C_k = \{(\theta_1^\pi, \ldots, \theta_k^\pi) \in \Re_+^k : \pi \in \Pi_k\}$$

- Note: Capacity regions are compact and symmetric

Weight-based schedulers

- Define: Weight-based scheduler $\pi \in \Pi_k$ allocates time slot t to user $i*$ for which

$$w_{i*}R_{i*}(t) = \max_i w_i R_i(t)$$

  – where w_i are the weights related to the scheduler

- Example: MR (which is the same as PF in our case)
  – $w_i = 1$ for all i

Connection between the two time scales

- Proposition 1:

$$E[\max_i w_i R_i(t)] = \max_{\mathbf{c}_k \in C_k} \sum_i w_i c_{ki}$$

  – Proof is straightforward:

$$\begin{aligned}\max_{\mathbf{c}_k} \sum_i w_i c_{ki} &= \max_\pi \sum_{\mathbf{r}} \sum_i w_i r_i p_i^\pi(\mathbf{r}) P\{R(t) = \mathbf{r}\} \\ &= \sum_{\mathbf{r}} (\max_i w_i r_i) P\{R(t) = \mathbf{r}\} \\ &= E[\max_i w_i R_i(t)]\end{aligned}$$

Recall the optimal scheduling problem (transient system)

- Assume that there are n jobs in the system at time 0
- What is the optimal way to make the system empty?
- Objective: Minimize the mean delay (or flow time)
- Define: Flow time (or total completion time) for policy ϕ

$$T^{\phi} = \sum_{i=1}^{n} t_i^{\phi}$$

where t_i is the completion time of job i

- Define: Operating policies

$$\Phi_n = \{\phi = (\mathbf{c}_1, \ldots, \mathbf{c}_n) : \mathbf{c}_k \in C_k \text{ for all } k\}$$

Aalto University
School of Electrical
Engineering

Recall the recursion for G* (based on the flow-level model)

- Define (recursively):

$$G_k^* = \min_{\mathbf{c}_k \in C_k} g_k(\mathbf{c}_k), \quad g_k(\mathbf{c}_k) = \frac{1}{c_{kk}} \left(k - \sum_{i=1}^{k-1} c_{ki} G_i^* \right)$$

- Open problem 1: Is it possible to show that in our case

$$G_1^* < \ldots < G_n^*$$

- Open problem 2: If so, how to implement the optimal operating policy with a the time-slot-scale scheduler:

$$\theta_i^{\pi_k^*} = c_{ki}^* \quad \text{for all } k, i$$

Key property

- Proposition 2:

$$E[\max_i G_i^* R_i] = \max_{\mathbf{c}_k \in C_k} \sum_i G_i^* c_{ki} = \sum_i G_i^* c_{ki}^* = k$$

  - Proof by induction

Alternative recursion for G* (based on the time-slot-level model)

- Define (recursively):

$$f_k(a) = \int_0^\infty \left(1 - P\{aR_k \le r\} \prod_{i=1}^{k-1} P\{G_i^* R_i \le r\}\right) dr$$

$$G_k^* = f_k^{-1}(k) \quad \text{(well-defined since } f_k \text{ increasing)}$$

  – Based on the equation:

$$E[\max_{i=1,\ldots,k} G_i^* R_i] = \int_0^\infty \left(1 - \prod_{i=1}^{k} P\{G_i^* R_i \le r\}\right) dr = k$$

Key result

- Proposition 3:

$$G_1^* < \ldots < G_n^*$$

  - Proof by induction
  - Idea briefly on the following slide
- Corollary: Solution of the optimal scheduling problem

$$T^* = \min_{\phi \in \Phi_n} T^{\phi} = \sum_{k=1}^{n} s_k G_k^*, \quad \phi^* = (\mathbf{c}_1^*, \ldots, \mathbf{c}_n^*)$$

$$c_{k1}^* \le \ldots \le c_{kk}^* \ \text{ for all } k$$

Idea of the proof

- Define:

$$X_k = \max_{i=1,\dots,k} G_i^* R_i$$

$$h_{k+1}(a) = E[(aR_{k+1} - X_k) \cdot 1_{\{aR_{k+1} > X_k\}}]$$

- Easily: $h_{k+1}(a)$ is non-decreasing and satisfies

$$h_{k+1}(G_{k+1}^*) = E[X_{k+1} - X_k] = (k+1) - k = 1$$

- It remains to show that

$$h_{k+1}(G_k^*) < 1$$

Optimal time-slot-level scheduler for flow-level performance

- Theorem 2: The optimal operating policy ϕ^* can be implemented by a sequence of weight-based schedulers π_k defined by weight vectors

$$\mathbf{w}_k = (G_1^*, \ldots, G_k^*)$$

  - Proof based on Propositions 1 and 2

- Summary: The optimal time-slot-level scheduler allocates time slot t to user $i*$ for which

$$G_{i*}^* R_{i*}(t) = \max_i G_i^* R_i(t)$$

Related reading

- S. Aalto, A. Penttinen, P. Lassila and P. Osti,
  On the optimal trade-off between SRPT and opportunistic scheduling, in *ACM SIGMETRICS* 2011

- S. Aalto, A. Penttinen, P. Lassila and P. Osti,
  Optimal size-based opportunistic scheduler for wireless systems, submitted, 2011

The End