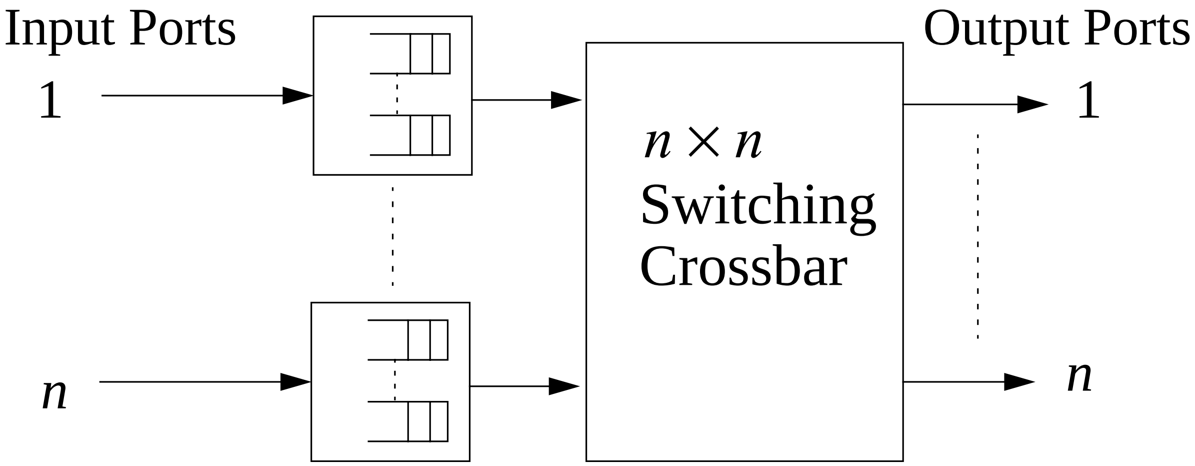
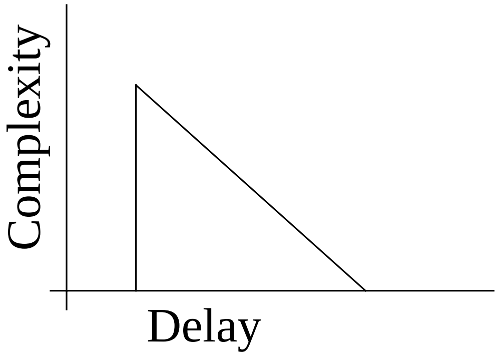
# Tradeoffs in Delay Guarantees and Computation Complexity for $N \times N$ Packet Switches



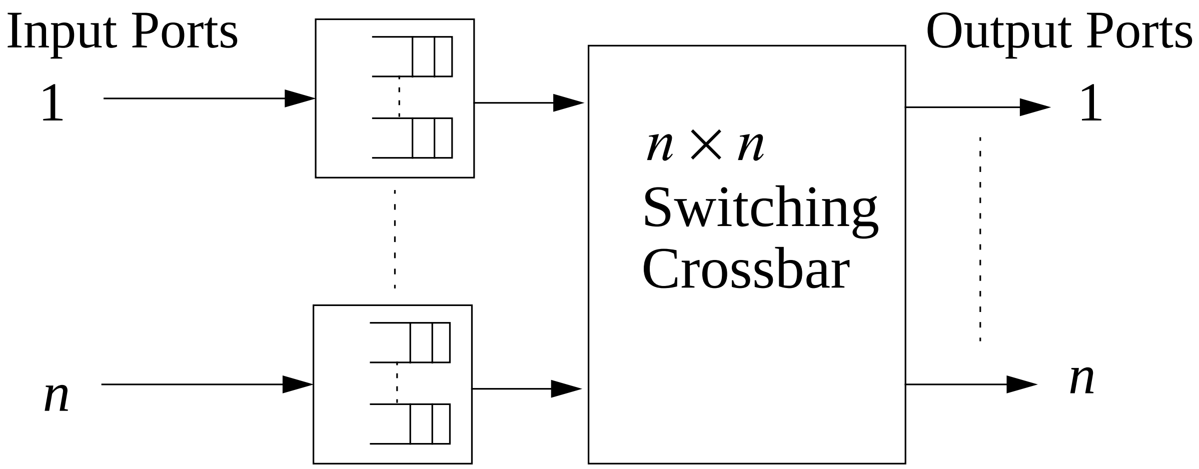


Michael J. Neely
MIT -- LIDS
mjneely@mit.edu

Eytan Modiano (MIT -- LIDS)
Charlie Rohrs (MIT -- LIDS)

M.J. Neely, E. Modiano, and C.E. Rohrs, "Tradeoffs in Delay Guarantees and Computation Complexity for N x N Packet Switches" Proceedings of the Conference on Information Sciences and Systems, Princeton: March 2002.

A Packet Switch:



Input Ports

1

$n$

$n \times n$
Switching
Crossbar

Output Ports

1

$n$

Notation:

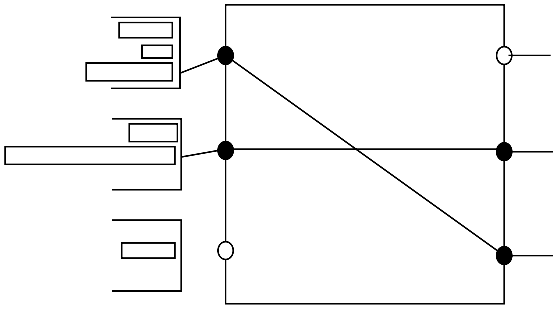
$N_{ij}(t)$ = Number of packets in queue $(i,j)$ at time $t$.

$A_{ij}(t)$ = Number of new arrivals to queue $(i,j)$ at time $t$.

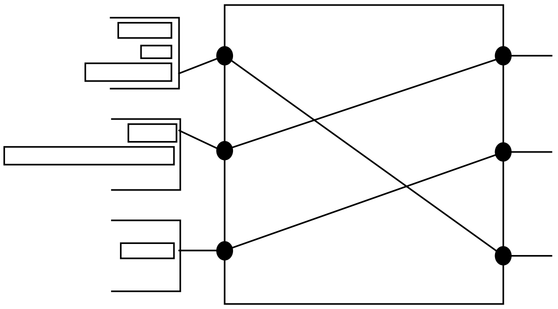
$C_{ij}(t)$ = Connection Decision for queue $(i,j)$ at time $t$.

Crossbar Constraint:

$(C_{ij}(t)) \in$ {Permuation Matrices} = $\{M_1, M_2, ..., M_{n!}\}$



(input 3 blocked)



(Maximum matching--
but large queue unserved)

# Precedents for $n \times n$ Switching:

Maximum Weight Match (MWM) gives 100% Thruput:

N. McKeown, V. Anantharam, J. Walrand (*Infocom* 1996)
L. Tassiulas and A. Ephremides (*Trans. on Aut. Contr.* 1992)

Delay Guarantee of MWM is *O(n)*:

E. Leonardi, M. Mellia, F. Neri, M. Ajmone Marson
(*Infocom* 2001)

***But MWM => $O(n^3)$ Complexity every timeslot***

Linear Complexity Algorithm for 100% Thruput:

L. Tassiulas (*Infocom* 1998) -- No Delay analysis
(Can be shown $avg.\ delay \leq O(n!)$)

Other Methods for Decreasing Complexity for 100% Thruput:

“A Practical Algorithm to Achieve 100% Thruput...”
A. Mekkittikul, N. McKeown (*Infocom* 1998)

“Stable Algorithms for Input Queued Switches”
D. Shah (*Allerton* 2001)

“Analysis of Sched. Algs. that Provide 100% Thruput...”
I. Keslassy and N. McKeown (*Allerton* 2001)

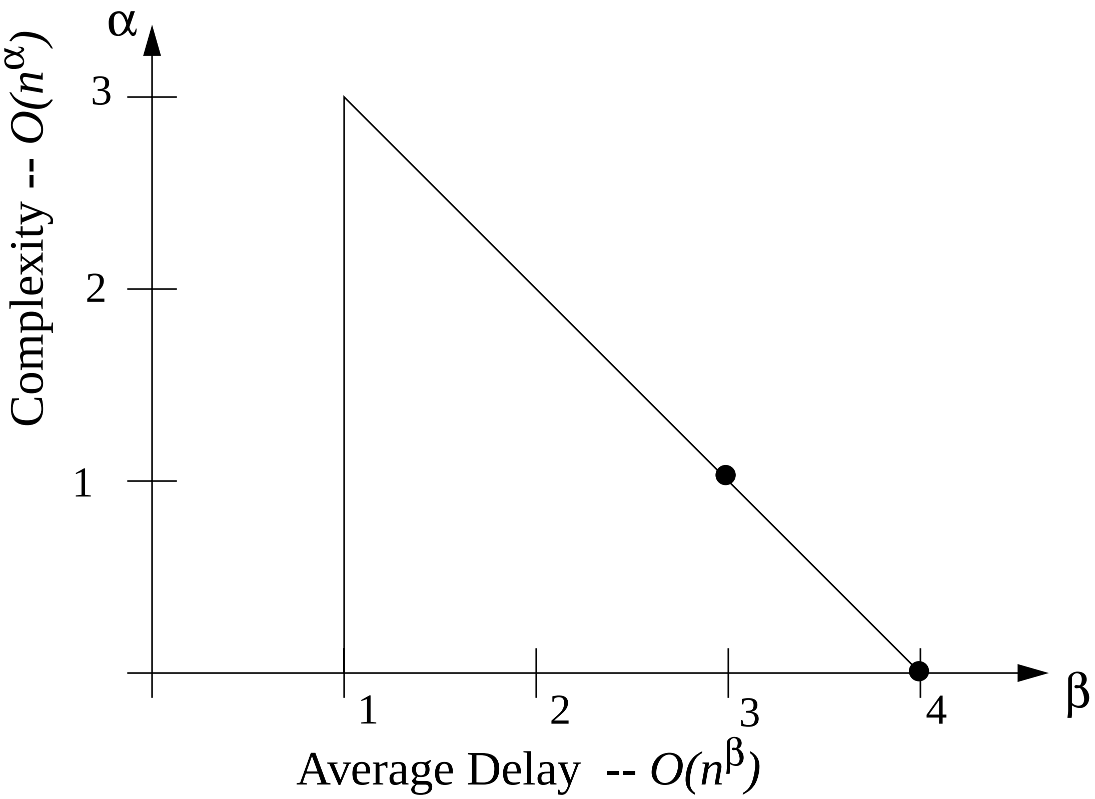
(Focus on Stability/100% thruput--Don’t examine on delay)

*Objective of this talk*: Keeping 100% Thruput, design a scheduling strategy with reduced computation complexity while ensuring polynomial delay bounds.

Let $\alpha \in (0, 3]$. We describe class of policies $\pi_\alpha$:

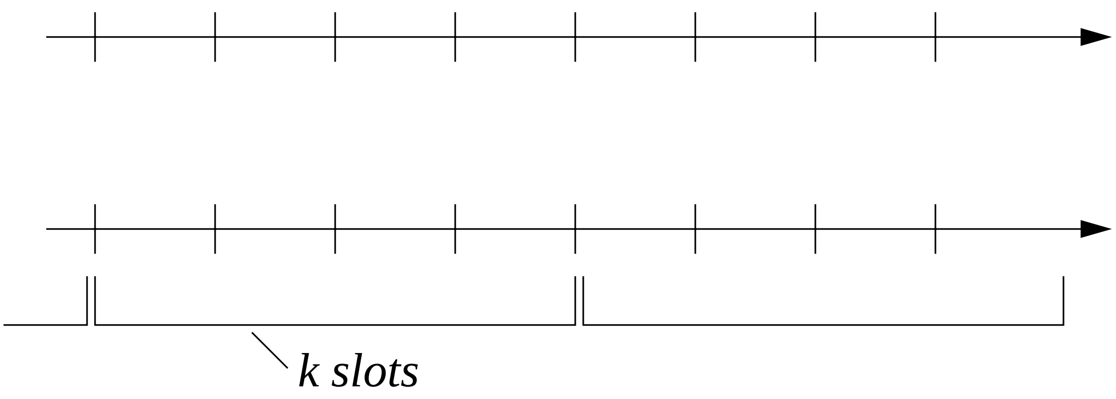
-Per-Timeslot Computation Complexity: $O(n^\alpha)$

-Average Delay: $O(n^{4-\alpha})$



## The Idea is Quite Simple:

-Use a modified version of MWM.
-Rather than compute a match every timeslot, we allow *k* timeslots for the computation.
-Switching Configuration held fixed for *k* slots while the next computation proceeds--using (estimated) weights equal to the queue sizes seen at the beginning of *k* slot interval.

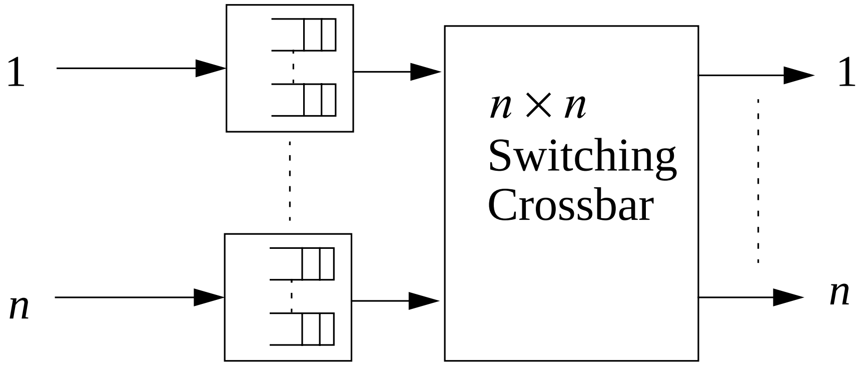


Clearly this reduces per-timeslot computation complexity at the expense of:

-Imposing fixed switch schedule for *k* slots
-Using out-of-date queue backlog information

The trick is to show this maintains stability, and ensures average delay held within *O(nk)*.

# Stability Region of an $n \times n$ Switch:



Notation:

$N_{ij}(t)$ = Number of packets in queue $(i,j)$ at time $t$.

$A_{ij}(t)$ = Number of new arrivals to queue $(i,j)$ at time $t$.

$C_{ij}(t)$ = Connection Decision for queue $(i,j)$ at time $t$.

Dynamics:

$$N_{ij}(t+1) = max(N_{ij}(t) - C_{ij}(t), 0) + A_{ij}(t)$$

Controls ($C_{ij}(t)$) limited to
Permutation Matrices $\{M_1, M_2, ..., M_{n!}\}$

Input rates: $$\lambda_{ij} = \lim_{t \to \infty} \frac{1}{t} \sum_{\tau=1}^{t} A_{ij}(\tau)$$

Well Known Stability Region of arrival rates:

$$\Omega = \left\{ (\lambda_{ij}) \middle| \sum_i \lambda_{ij} \leq 1, \sum_j \lambda_{ij} \leq 1 \right\}$$

$$\Omega = \left\{ (\lambda_{ij}) \middle| \sum_i \lambda_{ij} \le 1, \sum_j \lambda_{ij} \le 1 \right\}$$

<u>Quick demonstration of stability region</u>:

1. $(\lambda_{ij}) \notin \Omega$ => One of the ineq. constraints violated

=> System unstable. ❑

2. Simple proof that $(\lambda_{ij})$ *strictly interior* to $\Omega$ is sufficient:

a. *Birkhoff-Von Neumann Theorem:*

Convex Hull$\{M_1, M_2, ..., M_{n!}\}$ = *Dominant Face of* $\Omega$

$$= \Omega \in \left\{ (\lambda_{ij}) \middle| \sum_i \lambda_{ij} = 1, \sum_j \lambda_{ij} = 1 \right\}.$$

b. Suppose rates $(\lambda_{ij})$ are fully known and *strictly interior* to $\Omega$:

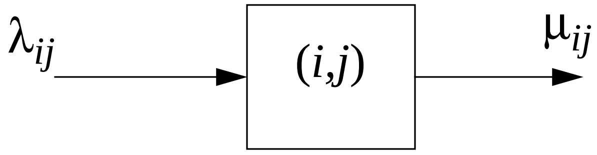
=> There are rates $(\mu_{ij}) \in$ *Dominant face of* $\Omega$

such that $(\lambda_{ij}) < (\mu_{ij})$.

=> $(\mu_{ij}) = p_1 M_1 + p_2 M_2 + ... + p_{n!} M_{n!}$ (*where* $\sum p_i = 1$).

Scheduling Strategy:

Every timeslot, choose matrix $M_i$ with probability $p_i$.

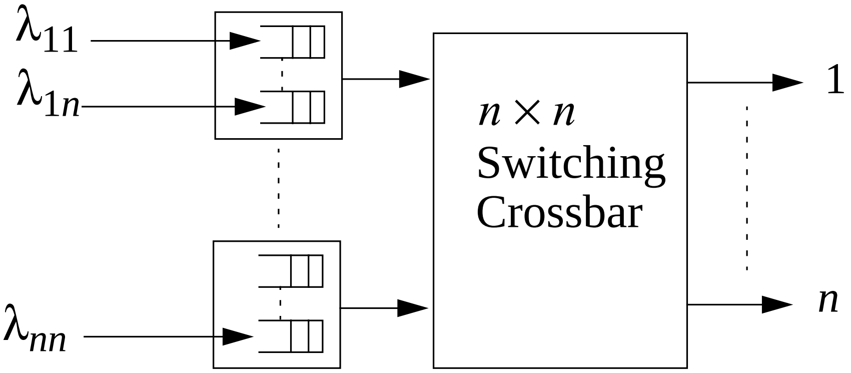


G/G/1 queue, Geometric service rate $\mu_{ij}$ (and $\lambda_{ij}<\mu_{ij}$).❑

(This stabilizing algorithm works when all rates ($\lambda_{ij}$) are known in advance).

Useful to examine Delay of this stabilizing algorithm:

Suppose inputs are Poisson with uniform rates $\lambda_{ij} = \lambda < 1/n$,

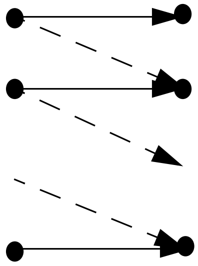


Choose $\mu_{ij}$ = 1/n for all $(i,j)$ (all permutations equally likely).
Then the loading on each queue is $\rho = \lambda n < 1$.

1. Slotted M/G/1 queue with Geometric Service time:

$$Delay = \overline{W} = \frac{n - 1/2}{1 - \rho} + 1. \quad \text{(which is } O(n)\text{)}$$

2. Periodic Schedule: (cycle thru the $n$ cyclic permutations)

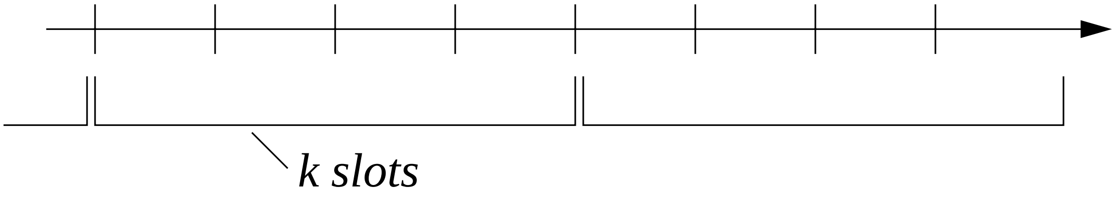


$$Delay = \overline{W} = \frac{n}{2(1 - \rho)} + 1.$$

(still $O(n)$)

## Algorithm for unknown rates $\lambda_{ij}$:

Original MWM: Every timeslot, use occupancies ($N_{ij}(t)$) to choose Permutation $C_{ij}(t)$ to maximize $\sum N_{ij}(t) C_{ij}(t)$.



*k*-slot dynamics:

$$N_{ij}(t+k) \le max(N_{ij}(t) - kC_{ij}(t), 0) + \sum_{r=0}^{k-1} A_{ij}(t+r)$$

While computing a MWM, it will be implemented *k*-slots later, and we don't know the true $N_{ij}(t)$ values to use.

Use *Estimate* $\tilde{N}_{ij}(t)$:

$\tilde{N}_{ij}(t)$ = Number of packets currently in queue (*i*, *j*) that did not arrive during the past *k*-slot interval (*t*-*k*, *t*].

Note that $N_{ij}(t+k) = max(N_{ij}(t) - kC_{ij}(t), 0)$ (★)

*Scheduling Policy* $\pi_k$:

-Configure Switch Matrix ($C_{ij}(t)$) to the permutation calculated by the prev. MWM computation, and keep fixed during [*t*, *t*+*k*).

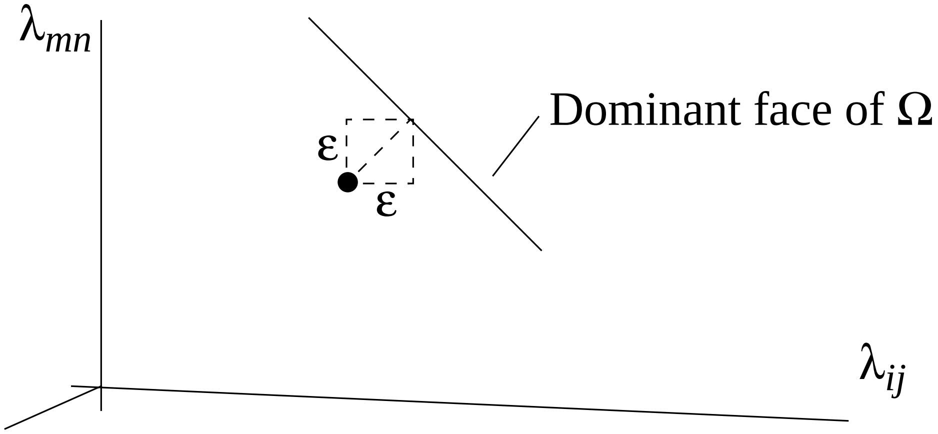
-Calculate $\tilde{N}_{ij}(t+k)$ using (★). Use these values in an MWM computation during the interval [*t*, *t*+*k*). ❑

## Delay Analysis:

Assume arrivals $A_{ij}(t)$ are *iid* every timeslot.

$\lambda_{ij} = E[A_{ij}(t)]$ (But these $\lambda_{ij}$'s are unknown to the controller).

Assume the rates are strictly interior to $\Omega$, and let $\varepsilon > 0$ be the largest value such that $(\lambda_{ij} + \varepsilon) \in \Omega$.



Let $\vec{\varepsilon} = (\varepsilon, \varepsilon, \ldots, \varepsilon)$. Then $\|\vec{\varepsilon}\|^2 = n^2\varepsilon^2$.

Define $d = \|\vec{\varepsilon}\| = n\varepsilon$ = "distance from $(\lambda_{ij})$ to boundary of capacity region $\Omega$."

Theorem: For any integer $k>0$, the policy $\pi_k$ has per-timeslot computation complexity $O(n^3/k)$ and ensures an average delay of $O(nk)$.

(Hence, if $\alpha \in (0, 3]$, choosing $k = \lceil n^{3-\alpha} \rceil$ yields complexity $O(n^{\alpha})$ and delay $O(n^{4-\alpha})$. )

Lyapunov Drift Argument:

Define a Lyapunov Function $L(\underline{N}) = \sum_{i,\,j} N_{ij}^2$.

$$Drift(t) = E[L(\underline{N}(t+1)) - L(\underline{N}(t)) | \underline{N}(t)]$$

<u>Drift Theorem</u>: Suppose there are non-negative values $\gamma_{ij}$ and a value $B$ such that:

$$Drift(t) \le B - \sum_{ij} \gamma_{ij} N_{ij}$$

Then steady state queue occupancies exist with finite mean, satisfying:

$$\sum_{ij} \gamma_{ij} \overline{N}_{ij} \le B \qquad ❑$$

(Proof uses a simple telescoping series argument similar to the delay theorem given in [Leonardi, Mellia, Neri, Marson *Infocom* ‘01]).

----------------------------------------------------------------

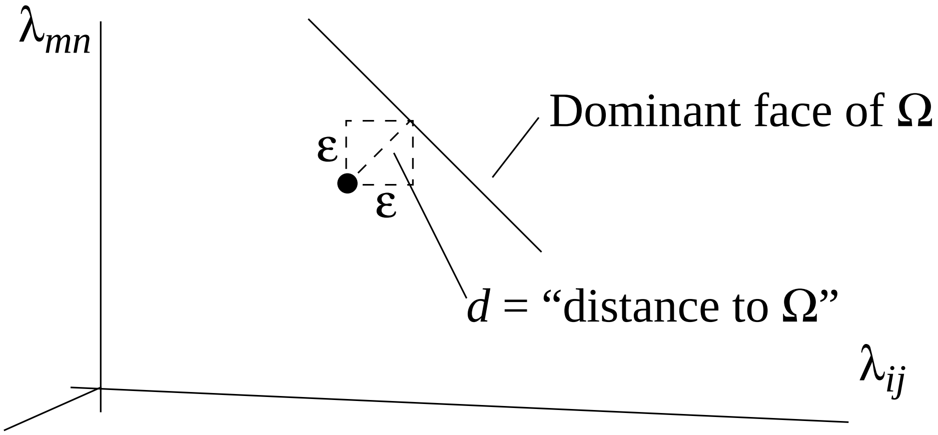
For our *k*-slot problem:

$$Drift(t) \le 2k^2 n - 2k \sum_{ij} N(t)(E[c_{ij} | \underline{N}] - E[A_{ij}])$$

Resulting Delay for any ($\lambda_{ij}$) matrix a distance $d$ from capacity region $\Omega$:

$$\overline{W} \leq \frac{kn}{\lambda_{av} d} + k$$

(where $\lambda_{av} = \frac{1}{n}\sum_{ij}\lambda_{ij}$ = average rate on an input port).



# Robustness to Input Rate Changes:

Previous analysis assumed *iid* packet arrivals every timeslot.

Consider arbitrary changes in the rate matrix ($\lambda_{ij}$).

-Rate matrix is $\underline{\lambda}^{(1)}$ for a certain duration of time.

-Rate matrix changes to $\underline{\lambda}^{(2)}$.

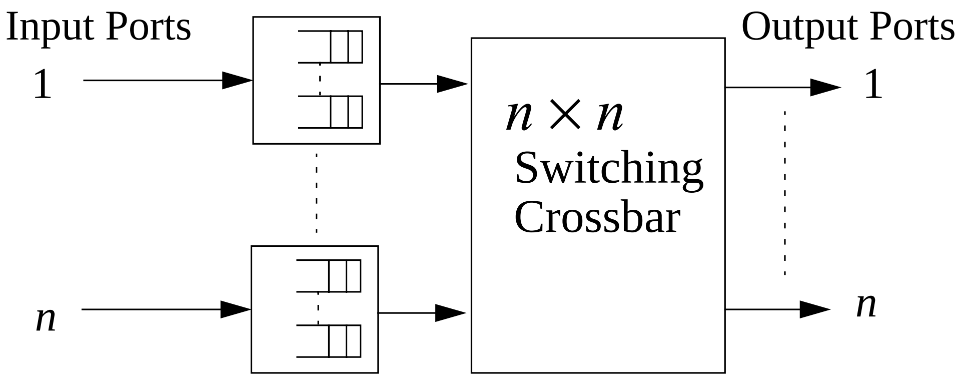
Formally modeled by a time-varying input distribution $f_t(\underline{a})$ on the arrival matrix ($a_{ij}(t)$) at each timestep *t*.

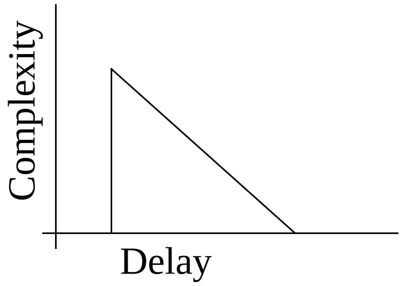
This change in input rates is reflected in the backlog that builds up in the queues of the system. Because the policy $\pi_k$ bases decisions on the size of the queues, it reacts smoothly to such changes in the input statistics:

$$\lim_{\tau \to \infty} \sup \frac{1}{\tau} \sum_{t=0}^{\tau-1} \left( \sum_{i,j} \overline{N}_{ij}(t) \right) \le \frac{kn^2}{d} + kn$$

(Little’s Theorem gives *O(n)* delay).

# Conclusions / Extensions





Algorithm $\pi^{\alpha}$ offering: Complexity $O(n^{a})$
Delay $O(n^{4-a})$

1. Useful for reducing complexity at expense of increasing delay.

2. Furthermore, algorithm naturally applies in situations where physical constraints require slower switching speeds (Optical switches, systems with crossbar electronics operating at speeds slower than input/output line rate).

3. Heuristic Improvements:
-Dynamic link weights.
-Enable switching matrix to switch to an alternate configuration in middle of a k-slot interval.
-Variable slot lenghts: MWM takes 50 slots, then 15...