

# Stability Properties of Constrained Queueing Systems



- Leandros Tassiulas and Anthony Ephemides, "Stability Properties of Constrained Queueing Systems and Scheduling Policies for Maximum Throughput in Multihop Radio Networks", IEEE Trans on Automatic Control, vol. 37, no. 12, December 1992.

## Outline

---

- Motivation
- Intent of work
- Network model
- Conditions for stability
- Maximum throughput policy
- Applications

## Motivation

---

- In a multihop radio network, what is the theoretical maximum throughput?
- Can we reach it? How?
- What properties do maximum throughput schedules have?

## Intent of work

---

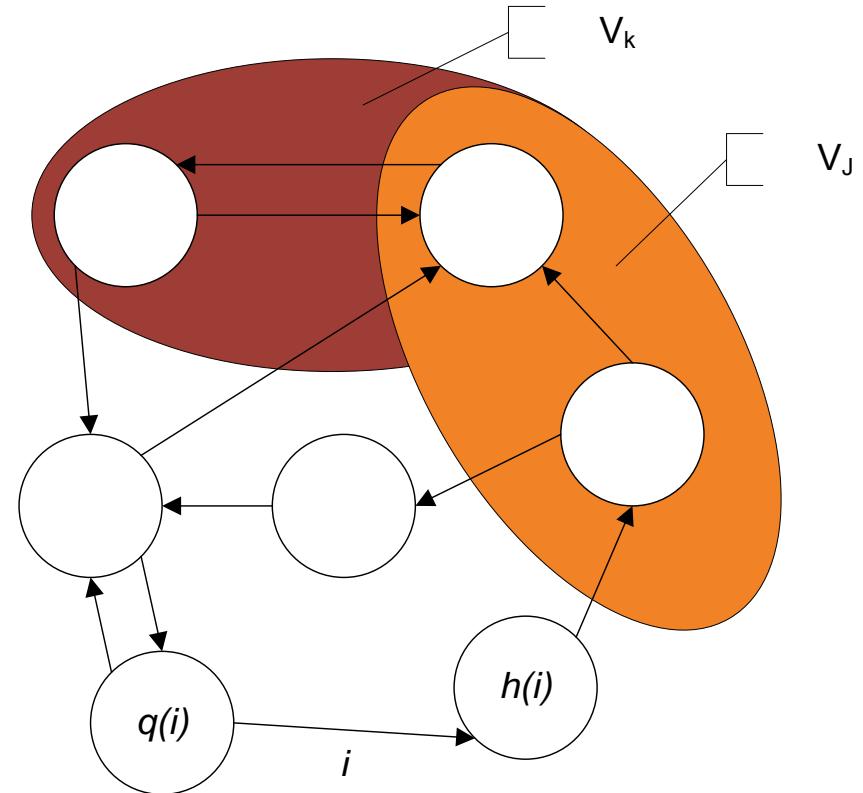
- Prove that a scheduling policy which maintains stable queue lengths has maximum throughput in a multihop radio network
- Show that there exists an optimal policy which, for all traffic rates for which it is possible, is stable
- Describe this optimal policy

## Network model: Assumptions

- Time is discretized into slots
- All transmissions take the same amount of time
- Link conditions are *a priori* determinable
- Centralized decision-making – state of network fully known
- Infinite buffer sizes
- Stationary arrival and service rates
- All messages are deliverable – effectively, network is strongly connected

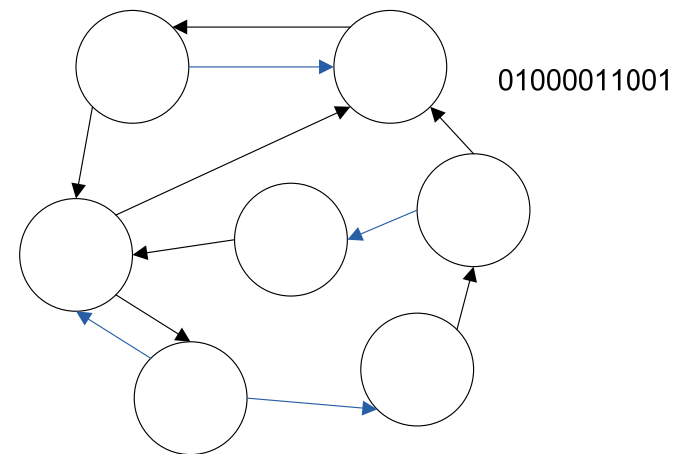
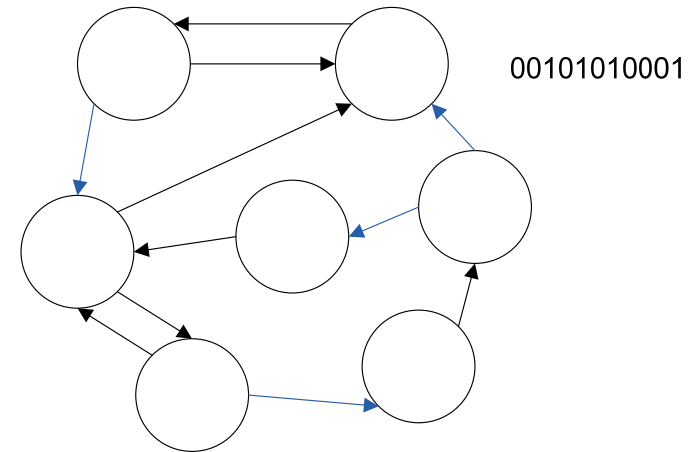
## Network topology

- **Queues** connected by **servers** (aka nodes connected by links)
- Each **customer** (packet) is of a particular **class**
- Each class has a particular **destination**
- The set of queues that are the destination of class  $j$  is denoted by  $V_j$



## Interference model

- At each time step, a set of links which do not interfere with each other may be used - this is called an *activation set* or *activation vector*
- The activation vector is a binary vector
- The complete set of allowable activation vectors is called the *constraint set (S)*



## System state evolution

- As customers are sent through links, the number of each type of customer in each queue will change
- For each class  $j$ , at each time  $t$ , there exists a vector  $\mathbf{X}^j$  describing the number of class  $j$  customers in each queue
- This equation describes how  $\mathbf{X}^j$  evolves over time:

$$\mathbf{X}^j(t+1) = \mathbf{X}^j(t) + \mathbf{R}^j \mathbf{M}(t+1) \mathbf{E}^j(t+1) + \mathbf{A}^j(t+1)$$

## System state evolution: State vector

$$\mathbf{X}^j(t+1) = \mathbf{X}^j(t) + \mathbf{R}^j \mathbf{M}(t+1) \mathbf{E}^j(t+1) + \mathbf{A}^j(t+1)$$

- $\mathbf{X}^j(t) = (X_{lj}: l = 1, \dots, L)$  is the vector of queue lengths of class  $j$  at the end of slot  $t$
- It contains all the state in the system

## System state evolution: Routing matrix

$$\mathbf{X}^j(t+1) = \mathbf{X}^j(t) + \mathbf{R}^j \mathbf{M}(t+1) \mathbf{E}^j(t+1) + \mathbf{A}^j(t+1)$$

- $\mathbf{R}^j$  is the routing matrix of class  $j$
- Purely determined by network topology
- An  $L \times N$  matrix that reflects the connectivity of the queues among themselves and with the destination nodes
- The element of  $\mathbf{R}^j$  in its  $l$ th row and  $i$ th column is:

$$r_{li}^j = \begin{cases} 1, & \text{if } h(i) = l \text{ and queue } l \text{ is not connected} \\ & \text{with the destination node of class } j \\ -1, & \text{if } q(i) = l \\ 0 & \text{otherwise.} \end{cases}$$

## System state evolution: Service matrix

$$\mathbf{X}^j(t+1) = \mathbf{X}^j(t) + \mathbf{R}^j \mathbf{M}(t+1) \mathbf{E}^j(t+1) + \mathbf{A}^j(t+1)$$

- At a given time step  $t$ , a binary variable  $M_i(t)$  indicates whether a customer served by server  $i$  completes service
- n.b.  $M_i(t)$  is defined even if a server isn't activated during  $t$
- $\mathbf{M}(t)$  is an  $N \times N$  diagonal matrix, the  $i$ th element of which is equal to  $M_i(t)$
- $\mathbf{A}^j(t) = (A_{lj}(t): l = 1, \dots, L)$  is a vector with its  $l$ th element  $A_{lj}(t)$  being equal to the number of customers of class  $j$  arriving at queue  $l$  during slot  $t$

## System state evolution: Single-class activation vector

$$\mathbf{X}^j(t+1) = \mathbf{X}^j(t) + \mathbf{R}^j \mathbf{M}(t+1) \mathbf{E}^j(t+1) + \mathbf{A}^j(t+1)$$

- A binary variable  $E_{ij}(t)$  indicates whether server  $i$  is activated during time slot  $t$
- $E_{ij}(t) = 1$  indicates that server  $i$  is activated and serves a customer of class  $j$
- The vector  $\mathbf{E}^j(t) = (E_{ij}(t): i = 1, \dots, N)$  indicates which servers are serving customers of class  $j$  this time slot
- $E_{ij}(t)$  are selected by the scheduling policy

## Multiclass activation vectors

- The vector  $\mathbf{E}(t) = (E_{ij}(t): i = 1, \dots, N; j = 1, \dots, J)$  indicates which class each server serves, if any, at slot  $t$
- Recall: an activation vector must be a valid transmission set. Thus,
- A binary vector  $\mathbf{e} = (e_{ij}: i = 1, \dots, N; j = 1, \dots, J)$  is a *multiclass activation vector* if the vectors  $\mathbf{e}^j = (e_{ij}: i = 1, \dots, N)$  are such that:

$$\sum_{j=1}^J \mathbf{e}^j \in \mathcal{S}$$

## System state evolution: Customer arrival

$$\mathbf{X}^j(t+1) = \mathbf{X}^j(t) + \mathbf{R}^j \mathbf{M}(t+1) \mathbf{E}^j(t+1) + \mathbf{A}^j(t+1)$$

- $\mathbf{A}^j(t) = (A_{lj}(t): l = 1, \dots, L)$  is a vector with its  $l$ th element  $A_{lj}(t)$  being equal to the number of customers of class  $j$  arriving at queue  $l$  during slot  $t$

## Random transmission and traffic models

- The sequences  $M_i(t)$ ,  $A_{lj}(t)$ , for all  $i, l, j$  are independent and identically distributed (i.i.d.)
- This means that they:
  - Don't change over time
  - Have no memory
- Note that they may be different between values of  $i, l, j$
- Additionally, the arrival process has a finite variance

## Transmission and traffic

- $E[M_i(t)] = m_i$  is equivalent to normalized bandwidth for server  $i$ 
  - $m_i$  is the proportion of time slots in which a transmission will successfully complete
- $E[A_{lj}(t)] = a_{lj}$  is equivalent to normalized demand by class  $j$  at queue  $l$ 
  - $a_{lj}$  is the proportion of time slots in which a customer of class  $j$  will show up at queue  $l$

## Activation policies

- An activation rule  $g$  maps system states onto multiclass activation vectors
- Activation rules have the property that no servers are activated for non-existent customers
- An activation policy  $\pi$  consists of a sequence of activation rules  $g_t$ . If all such  $g_t$  are identical, then the policy is said to be stationary

## Markov chain model of queue length

---

- Queue length is a Markov chain,  $\mathbf{X}$
- This is true because of the assumptions made regarding arrival and transmission rates

## Conditions for stability

- The system is stable if the queue length process  $X$  reaches steady state and does not go to infinity
- The queue length process will reach steady state if:
  - All recurrent states are positive recurrent (i.e. there are a finite number of recurrent states)
  - The probability of hitting a recurrent state is 1
- If the Markov chain is irreducible (i.e., it is possible to get from any state to any other state), then this is equivalent to ergodicity of  $X$

## Reducible Markov chain

- The state space of the chain is partitioned into the sets  $T$ ,  $R_1, R_2, \dots$
- All  $R_j$  are closed sets of communicating states
- $T$  contains all transient states
- For any  $\mathbf{x}$  in  $T$  assume that  $\mathbf{X}(0) = \mathbf{x}$  and consider the time at which the chain hits one of the sets  $R_j$  for the first time:

$$\tau_x = \begin{cases} \infty, & \text{if } \mathbf{X}(t) \in T, \forall t > 0 \\ \min\{t > 0 : \mathbf{X}(t) \notin T\}, & \text{otherwise} \end{cases}$$

## Definition of stability for reducible Markov chain

- The system is stable if for the queue length process we have

$$P(\tau_y < \infty) = 1 \quad \forall \mathbf{y} \in T$$

- and all states  $x \in \bigcup_{j=1}^{\infty} R_j$  are positive recurrent

## Stability of reducible Markov chains, cont'd

Consider a Markov chain  $X(t)$  with state space  $\mathfrak{X}$ . If there exists a lower bounded real function  $V : \mathfrak{X} \rightarrow \mathbf{R}$ , an  $\varepsilon > 0$  and a finite subset  $\mathfrak{X}_0$  of  $\mathfrak{X}$  such that :

$$E[V(\mathbf{X}(t+1)) - V(\mathbf{X}(t)) \mid \mathbf{X}(t) = \mathbf{y}] \leq -\varepsilon \quad \text{if } y \notin \mathfrak{X}_0$$

$$E[V(\mathbf{X}(t+1)) \mid \mathbf{X}(t) = \mathbf{y}] < \infty \quad \text{if } y \in \mathfrak{X}_0$$

then for the time  $\tau_x$  we have  $P(\tau_x < \infty) = 1 \quad \forall x \in T$

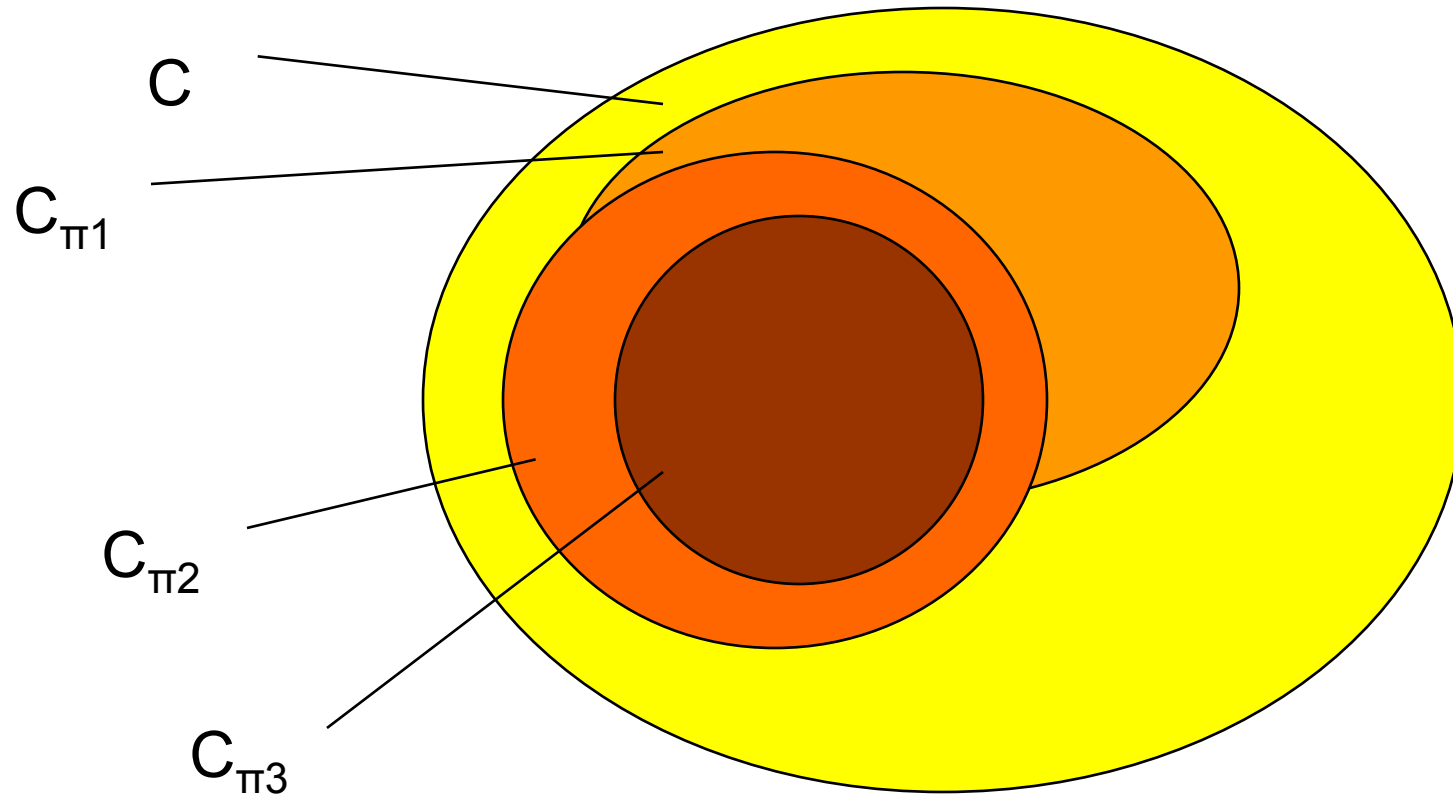
and all states  $x \in \bigcup_{j=1}^{\infty} R_j$  are positive recurrent.

- A proof is not given, but the authors note that it is a “trivial extension of Foster’s criteria for irreducible chains”
- That this guarantees stability can also be seen intuitively
- Still need to prove that such a lower-bounded function exists, for any given chain

## Scheduling for maximum throughput

- Want the system to be able to handle a wide range of arrival rates – high throughput included
- Denote the arrival rate of class  $j$  to queue  $l$ ,  $E[A_{lj}(t)] = a_{lj}$ ;  $\mathbf{a} = (a_{lj}: l = 1, \dots, L; j = 1, \dots, J)$
- The stability region  $C_\pi$  of policy  $\pi$  is the set of such vectors  $\mathbf{a}$  for which the system is stable under  $\pi$
- The stability region  $C$  is the union of all such sets
- A policy  $\pi_1$  dominates policy  $\pi_2$  if the system is always stable under  $\pi_1$  when it is stable under  $\pi_2$

## Stability region diagram



## Maximum throughput policy: Skeleton

---

1. Select a weight for each server which is proportional to the expected overall amount of progress which it makes when activated
2. Select an activation vector such that the sum of these weights is maximized
3. Turn off any server which is serving a queue with fewer customers than there are servers serving that queue

## Maximum throughput policy: Progress

- Given:
  - A server  $i$  transmits from queue  $q(i)$  to queue  $h(i)$
  - The set of queues connected to the destination of class  $j = V_j$
- The weight of each server is defined as follows:

$$D_{ij}(t) = \begin{cases} (X_{q(i)j}(t-1) - X_{h(i)}(t-1))m_i, & \text{if } h(i) \notin V_j \\ X_{q(i)j}(t-1)m_i, & \text{if } h(i) \in V_j \end{cases}$$

## Implications of this policy

---

- Queue lengths for each class tend to be equalized
- Higher priority is given to queues which are “backed up”
- No considerations of QoS
- Requires solving an optimization problem that is NP in the general case; however, it is in P for several networking cases

## Characterization of the stability region

---

- Will characterize the stability region by defining a set  $C'$
- Will show that  $C' \subset C \subset \bar{C}'$ , where  $\bar{C}'$  is the closure of  $C'$
- The definition of  $C'$  involves deterministic flows in the network

## Characterization with deterministic flows

---

- Assume that the system is stable under policy  $\pi$
- Operates in steady state
- Let  $f_{ij}$  be the rate at which customers of class  $j$  are served by server  $i$
- Since we're at steady state, we should be satisfying flow conservation

## **a**-admissible flow vectors

- Let  $\mathbf{a}^j = (a_{lj}: l = 1, \dots, L)$  be the arrival rate vector for class  $j$
- A vector  $\mathbf{f}^j = (f_{ij}: i = 1, \dots, N)$  that consists of non-negative numbers and satisfies the flow conservation equations is called an **a**-admissible flow vector for class  $j$
- A vector  $\mathbf{f} = (f_{ij}: i = 1, \dots, N; j = 1, \dots, J)$  for which all corresponding  $\mathbf{f}^j$  satisfy this requirement is an **a**-admissible multicommodity flow vector

$$\mathbf{a}^j = -\mathbf{R}^j \mathbf{f}^j$$

- Let  $F_a$  be the set of all  $\mathbf{a}$ -admissible multicommodity flow vectors
- Define the total flow vector  $\mathbf{f} = \sum \mathbf{f}_j$
- The set  $C'$  is defined:

$$C' = \left\{ \begin{array}{l} \mathbf{a} : \text{there exists } \mathbf{f} \in F_a, c \in \text{co}(S) \text{ such that for the corresponding} \\ f \text{ we have } m_i^{-1} f < c_i \text{ if } f_i > 0 \text{ and } f_i = 0 \text{ if } c_i = 0 \end{array} \right.$$

- $\text{co}(S)$  is the convex hull of the constraint set  $S$
- i.e.  $\text{co}(S)$  is the set of average per-server attempted transmissions we can make

Closure of  $C'$ 

- The closure of  $C'$  is defined as follows:

$$\overline{C'} = \left\{ \mathbf{a} : \text{there exists an } \mathbf{f} \in F_a, \text{ and a } \mathbf{c} \in co(S), \right. \\ \left. \text{such that } \mathbf{M}^{-1} \mathbf{f} \leq \mathbf{c} \right\}$$

## Optimality of $\pi_0$

1. Under policy  $\pi_0$  the system is stable for  $\forall \mathbf{a} \in C'$

$$C' \subset C_{\pi_0}$$

2. If  $\mathbf{a} \in (\overline{C'})^c$ , then the system is unstable for any policy.

3. By definition,  $C_{\pi_0} \subset C$ , and from 2 we have  $C' \subset C_{\pi_0} \subset C$

4. From 1 and 2, we have  $C \subset \overline{C'} \subset \overline{C_{\pi_0}}$

5. Therefore, the set  $C'$  characterizes the system stability region in the sense

$$C' \subset C \subset \overline{C'}$$

and for the stability region of policy  $\pi_0$  we have

$$C' \subset C_{\pi_0} \subset C \subset C_{\pi_0}$$

## Time complexity

---

- Unfortunately, determining whether an arrival and service rate vector pair are in  $C$  is an NP-hard problem.
- Additionally, so is finding the correct activation set to use in each time slot
- However, if secondary interference is tolerated, then it is solvable in polynomial time

## Non-stationary policies

---

- These are policies with different values of  $g_t$  at different time slots  $t$
- However, they don't result in a larger stability region
- Thus, there is little point in using them for the systems modeled here

## Practical applications: Multihop radio networks

- Conflict constraints:
  - If each node has a single transceiver, then each node may only participate in a single transaction in each time slot
  - If there is a single frequency band, then the transmission is received without conflict only if other transmitting nodes are sufficiently distant
- The authors posit that, in a network with spread spectrum, the second constraint does not hold - this is what they mean by “secondary interference”

## Secondary interference not tolerated

---

- Constraints can be represented by conflicting pairs, thus allowing the creation of a conflict graph, with servers represented by nodes and conflicts by edges
- Finding the optimal activation set is equivalent to finding the maximum weighted independent set in this graph

## Secondary interference tolerated

---

- If secondary interference is tolerated, then we must only avoid using a queue in two separate transmissions
- This is equivalent to finding a maximum weighted matching, which is in P

Thank you

---

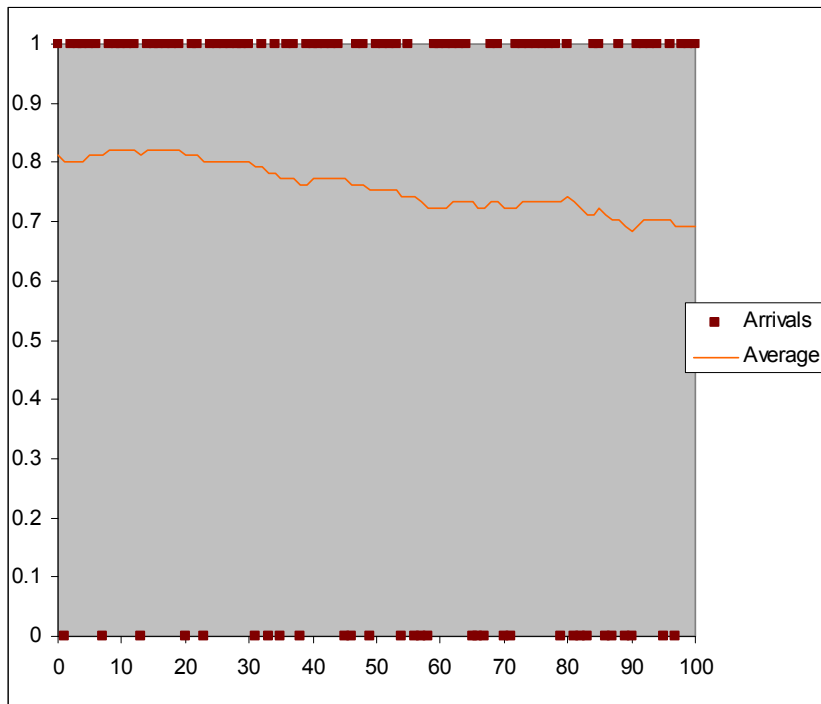
Questions?

# BACKUP

---

# Independent and identically distributed

## YES



## NO

