# Communication Networks (0368-3030) / Fall 2013

## The Blavatnik School of Computer Science, Tel-Aviv University

Allon Wagner

# Scheduling

Recommended references:

- An Engineering Approach to Computer Networking \ S.Keshav
- Computer Networks - Performance and Quality of Service \ Ivan Marsic (available online)

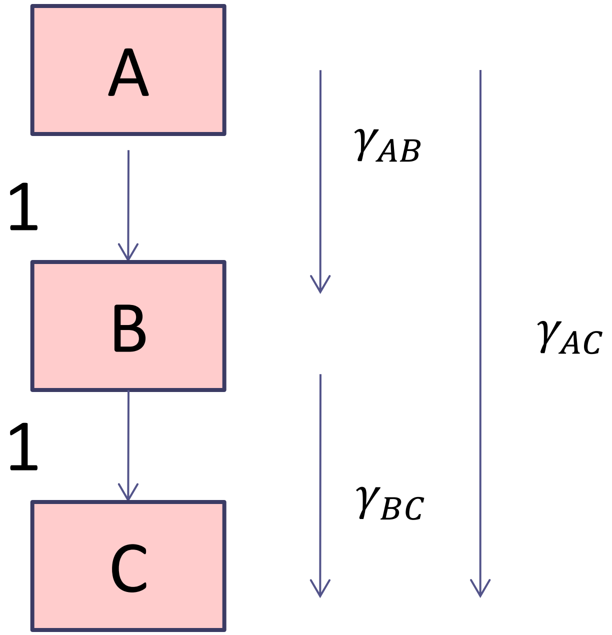
# Fair allocation

- Unfair allocation is the result of uncontrolled competition between users for the same resource.
  - Sometimes we want to augment fairness with user prioritization (i.e., maintain a fair allocation while supporting premium users)
  - There's often a tradeoff between fairness and optimality

# Fairness vs. optimality

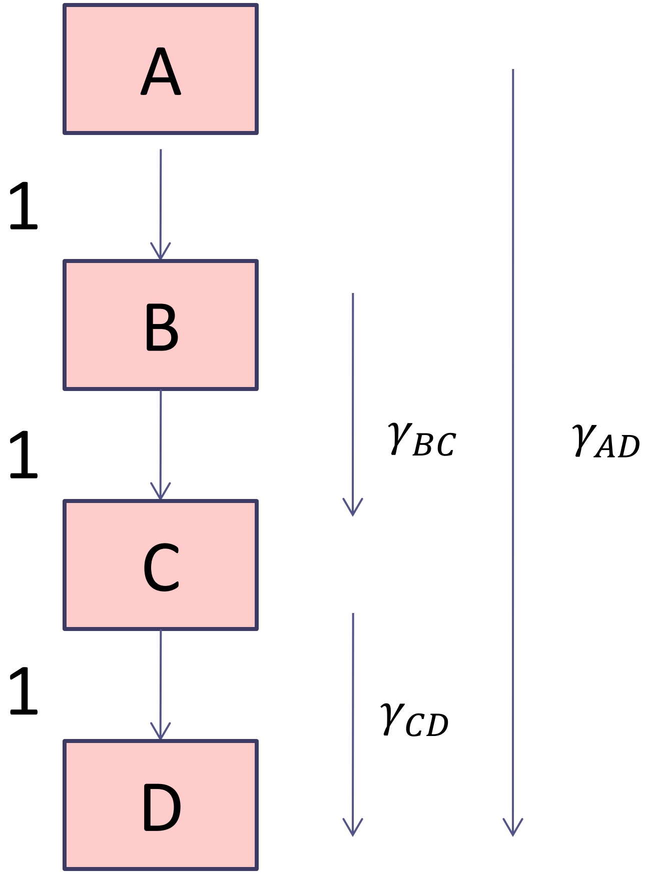
- A simple network with 3 nodes and 2 links with capacity 1 each.
- 3 flows with demand 1: A→B, A→C, B→C
- Optimal throughput:

$$\gamma_{AB} = \gamma_{BC} = 1$$
$$\gamma_{AC} = 0$$



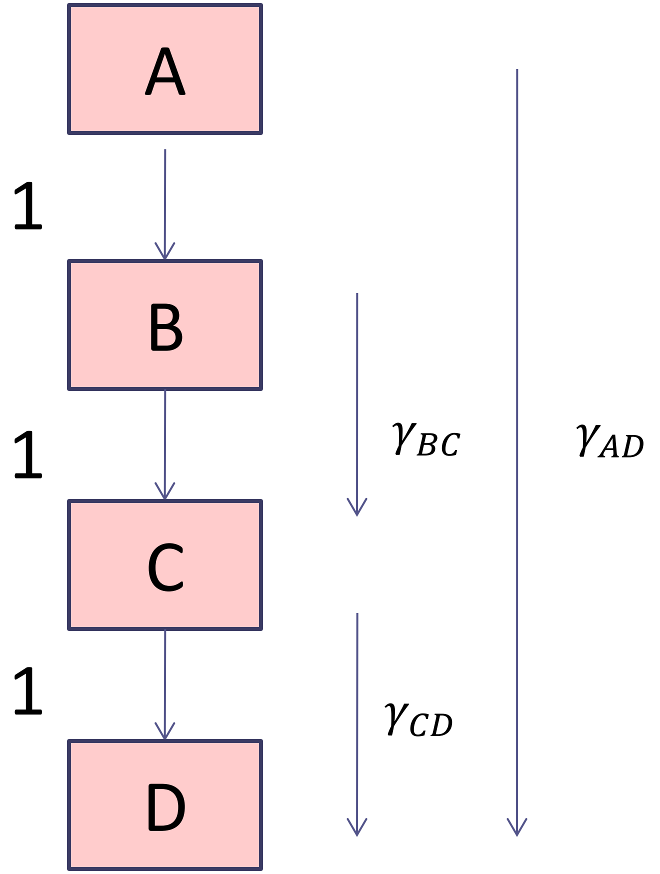
# But what is "fair allocation"?

- All users get the same throughput?
  - $\gamma_{AD} = \gamma_{BC} = \gamma_{CD} = 0.5$
  - But then $\gamma_{CD}$ actually gets more resources
- All users get the same amount of network resources?
  - $\gamma_{BC} = \gamma_{CD} = 0.75$
  - $\gamma_{AD} = 0.25$
    (total on 3 links is 0.75)



# But what is “fair allocation”?

- Each user gets a throughput proportional to how much it damages other users?
  - ▫ $\gamma_{BC} = \gamma_{CD} = \frac{2}{3}$
  - ▫ $\gamma_{AD} = \frac{1}{3}$ (total interference on the two links is $\frac{2}{3}$)



# An example

- Assume we have a simple resource with capacity 30
  - No network considerations for the time being
  - For instance: a single link, a queue in a router etc.
- User demands are $A = 20, B = 20, C = 20$
  - Fair allocation is trivial
- But what if the demands are:

$$A = 4, B = 20, C = 20$$

# Max-Min Fairness

- A sharing technique commonly used in practice
- The idea: Users who need less than what they are entitled to get their full demand. The excess is evenly distributed among the "heavy" users.
- Formally:
  - Resources are allocated in order of increasing demand
  - No source obtains a resource share larger than its demand
  - Sources with unsatisfied demands obtain an equal share of the resource

# An example

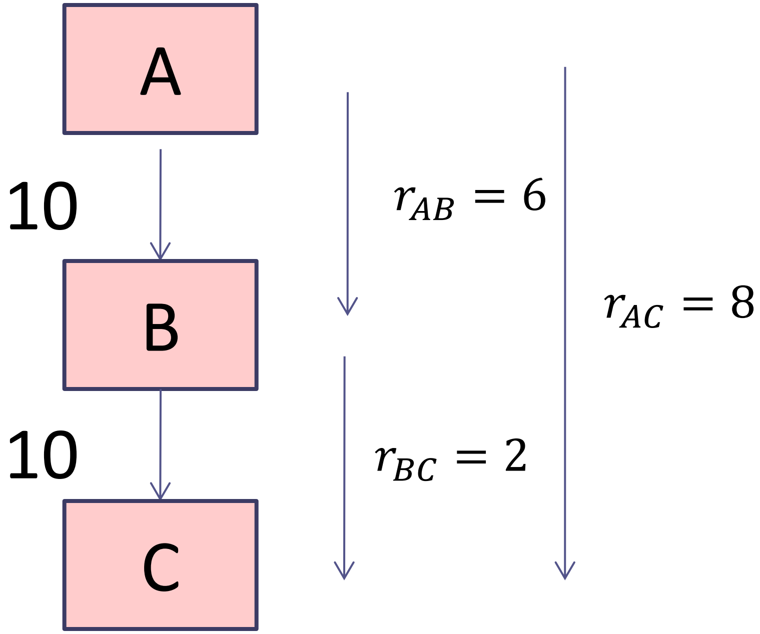
- Compute the max-min fair allocation for a set of four sources with demands 2, 2.6, 4, 5 when the resource has capacity 10.

1. Give 2 to every user. 1's demand is fulfilled, we have 2 excess capacity to distribute.
2. Give 2.6 to users 2,3,4. 2's demand is fulfilled, we have excess 0.2 to distribute.
3. Final allocation is: 2, 2.6, 2.7, 2.7

# Generalization to a graph

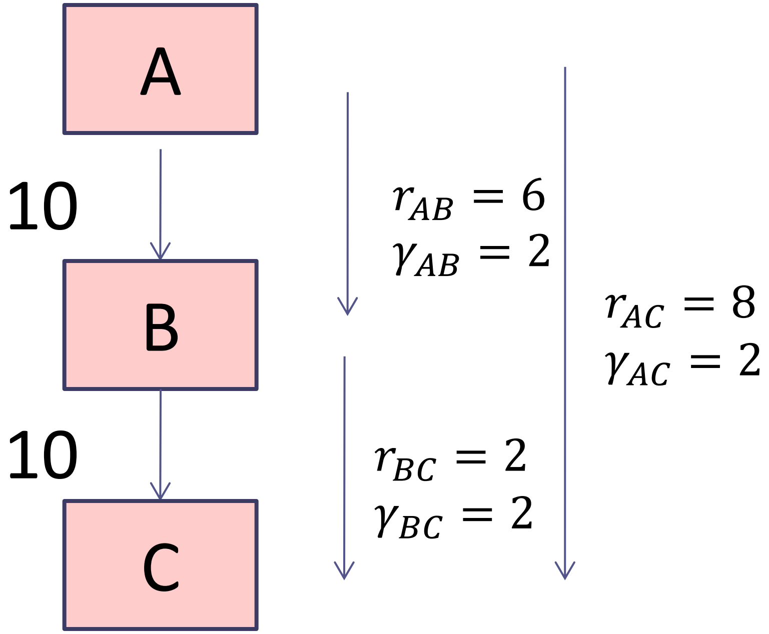
- We have a directed graph $G = (V, E)$, with capacity $c_e$ for each edge $e$.
- We have a set of ongoing calls (flows). Each call $i$ has demand $r_i$ and a (fixed) path $p_i$.
- Algorithm:
  - ▫ Increase all flows equally until one link fills or until a flow gets its full demand.
  - ▫ Fix the rate of the bottleneck flows.
  - ▫ Continue with the unfixed flows.

# An example



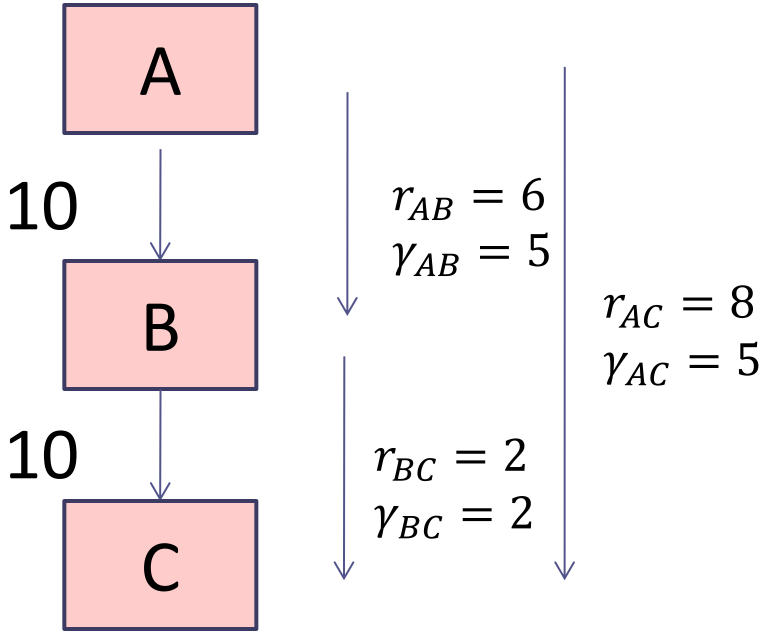
# An example

- Give 2 to all flows
- Demand BC is fulfilled.



# An example

- Continue increasing flows equally. When AB and AC get each 5, edge (A,B) is saturated.



# Generalized Processor Sharing (GPS)

- A *theoretical* scheduler:
  - ▫ Do a round robin
  - ▫ When visiting a non-empty queue, serve an infinitesimal amount of bits
- GPS is max-min fair!
  - ▫ If there are $N$ connections, each one gets $^{1}/_{N}$ of the resource.
  - ▫ If one connection does not utilize its entire share, the scheduler can use this time to do another round robin and serve equally all other connections.
  - ▫ This is max-min fairness by definition

# Weighted Fair Queuing (WFQ)

- WFQ emulates GPS
- It computes the time a packet would complete service had we been serving packets with a GPS scheduler, and serves packets  in order of these finish times.
- We call a packet’s finishing time under GPS a *finish number* to emphasize that it is only a service tag that indicates the relative order in which the packet is to be served, and has nothing to do with the actual time at which the packet is served.

# Weighted Fair Queuing (WFQ)

- Active connections:
  - Have something to send
  - The largest finish number of a packet scheduled to be sent from their queues is larger than the current round number.
- Round number
  - An abstraction of the GPS time
  - A real-valued variable that increases at a rate inversely proportional to the number of active connections

# Weighted Fair Queuing (WFQ)

- $R(t)$ - the round time at time $t$
- $F(i,k)$ - the finish number for the $k$-th packet on connection $i$.
- $P(i,k)$ - length of the $k$-th packet on connection $i$.
- Suppose that the current round is $R(t)$ and that packet $k$ arrives to queue $i$:
  - If $i$ is empty: $F(i,k) = R(t) + P(i,k)$
  - If $i$ is active: $F(i,k) = F(i,k-1) + P(i,k)$
- Therefore:
  - $F(i,k) = \max\{F(i,k-1), R(t)\} + P(i,k)$

# Weighted Fair Queuing (WFQ)

- $R(t)$ - the round time at time $t$
- $F(i,k)$ - the finish number for the $k$-th packet on connection $i$.
- $P(i,k)$ - length of the $k$-th packet on connection $i$.
- Suppose that the current round is $R(t)$ and that packet $k$ arrives to queue $i$:
  - ▫ If $i$ is empty: $F(i,k) = R(t) + P(i,k)$
  - ▫ If $i$ is active: $F(i,k) = F(i,k-1) + P(i,k)$
- Therefore:
  - ▫ $F(i,k) = \max\{F(i,k-1), R(t)\} + P(i,k)$

# Weighted Fair Queuing (WFQ)

- $n(t)$ - number of active flows at time $t$
- If $x$ is a timespan in which the number of active connections did not change then:
  - $R(t+x) = R(t) + \frac{x}{n(t)}$

# Adding flow weights $w_i$

- The GPS scheduler serves each flow in every round proportionally to its weight.
  - $\frac{w_j}{W(t)}$
  - $W(t)$ is the sum of the weights of ***active*** flows in time $t$.
- Generalized equations:
- $F(i,k) = \max\{F(i,k-1), R(t)\} + \frac{P(i,k)}{w_i}$
- $R(t+x) = R(t) + \frac{x}{W(t)}$

# Example (Keshav’s book, chapter 9)

- We consider a WFQ scheduler with 3 equally weighted connections, A, B, and C. The link service rate is 1 unit/s.
  - At time 0: packets of size 1,2, 2, arrive to A, B, C, respectively.
  - At time 4: a packet of size 2 arrive to A.
- Let’s start with a GPS simulation
- At time 0: $R(0) = 0$.
- All queues are empty and so the finish numbers for the packets are 1, 2, 2 respectivelty.

$$F(A, 1) = 1, F(B, 1) = 2, F(C, 1) = 2$$

# Example (Keshav’s book, chapter 9)

- We have 3 active connections → the round number increases at rate $\frac{1}{3}$ unit/s.
- Time 3 (sec):
  - $R(3) = 1.$
  - Packet (A, 1) completes service and A becomes inactive.
- Between times 3 and 4 we have two active connections, and so the round number increases at rate $\frac{1}{2}$ unit/s.

# Example (Keshav's book, chapter 9)

- Time 4 (sec):
  - $R(4) = 1.5$
  - B, C are still active and a new packet arrives to A.
  - A is empty: $F(A,2) = R(4) + P(A,2) = 3.5$.
  - Round number increases again at rate $\frac{1}{3}$ unit/s → It would take another 1.5 seconds until we complete round 2.

# Example (Keshav's book, chapter 9)

- Time 5.5 (sec):
  - R(5.5) = 2.
  - Packets (B, 1) and (C, 1) simultaneously finish service.
  - Round number starts to increase at rate 1 unit/s.
- Time 7 (sec):
  - R(7) = 3.5.
  - Packet (A, 2) finished service.

# Example (Keshav's book, chapter 9)

- Now, let's consider the WFQ scheduler.
- Time 0:
  - Waiting packets:

    $$F(A,1) = 1, F(B,1) = 2, F(C,1) = 2$$
  - A is served first
- Time 1 (sec):
  - Waiting packets:

    $$F(B,1) = 2, F(C,1) = 2$$
  - Either B or C is served, break tie somehow.
  - Let's assume we served B.

# Example (Keshav's book, chapter 9)

- Time 3 (sec):
  - Waiting packets:

$$F(C,1) = 2$$

- Time 5 (sec):
  - Waiting packets:

$$F(A,2) = 3.5$$

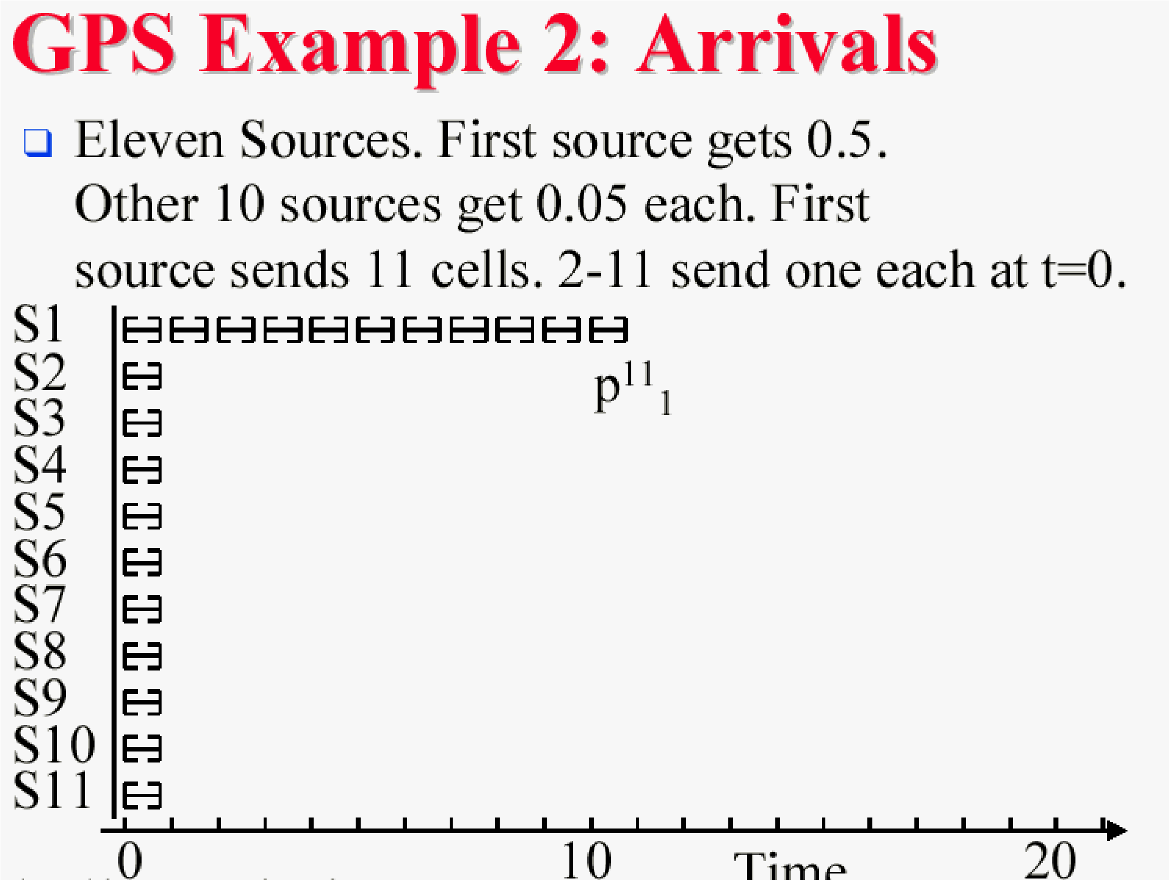
  - Serve A
- Time 7 (sec): the system becomes idle

# Worst Case Fair WFQ (WF$^2$Q)

- Under WFQ, a connection can receive substantially more service than with GPS
  - Unfair in terms of absolute fairness
- WF$^2$Q has better fairness
  - It can be shown that no packet-by-packet scheduler can be fairer
- **WFQ:** From all the packets awaiting service, serve the one with the smallest finish number.
- **WF$^2$Q:** From all the packets awaiting service, serve the one that has the smallest finish number, but consider only packets that have already started service (and possibly finished) in the corresponding GPS system.

# An example (by Bennett & Zhang)

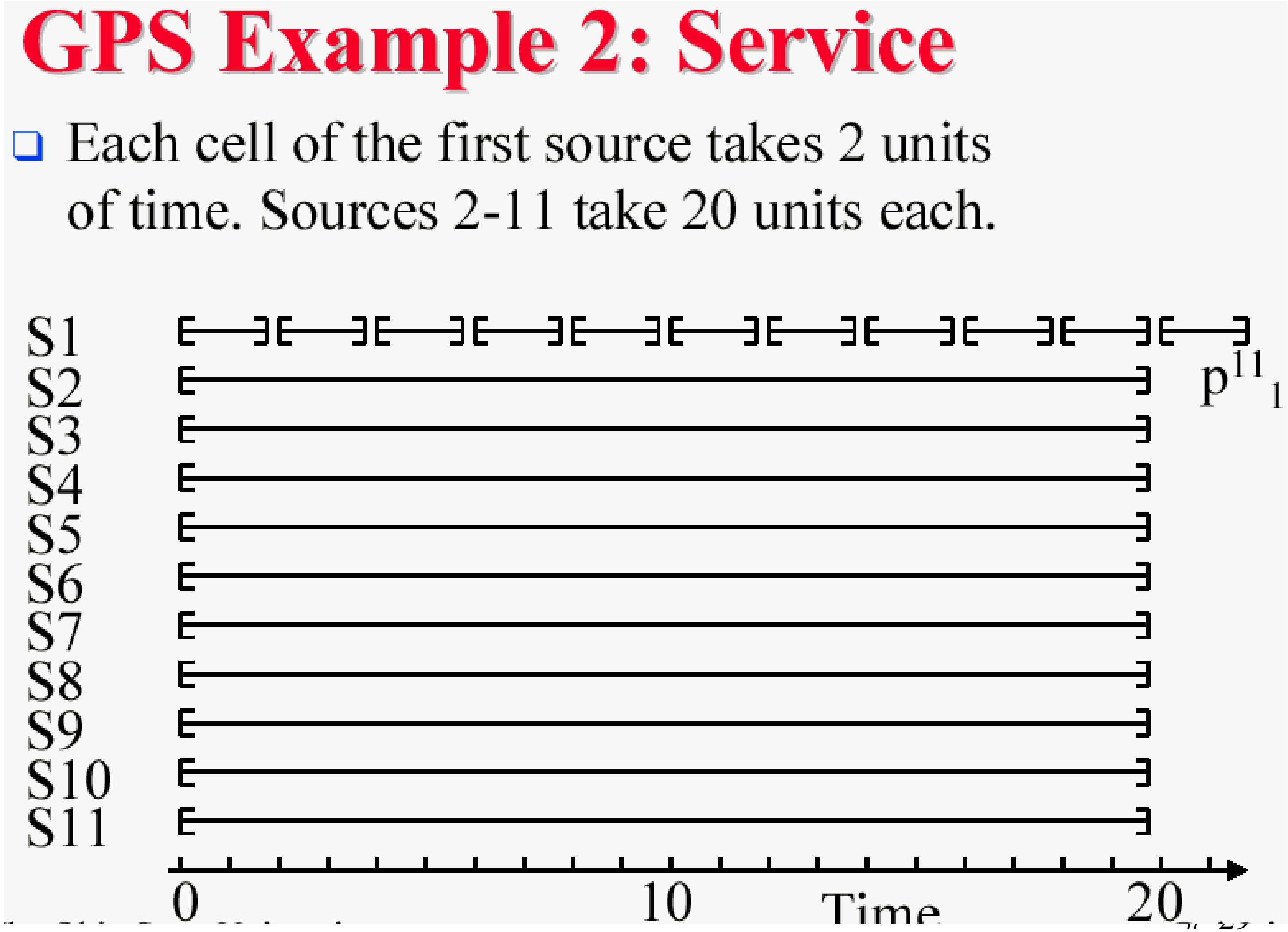
- 11 sources, with source 1 having a weight of 10 and the others a weight of 1.
- Source 1 has 11 packets to send, the others have 1.



## GPS Example 2: Arrivals

❑ Eleven Sources. First source gets 0.5. Other 10 sources get 0.05 each. First source sends 11 cells. 2-11 send one each at t=0.

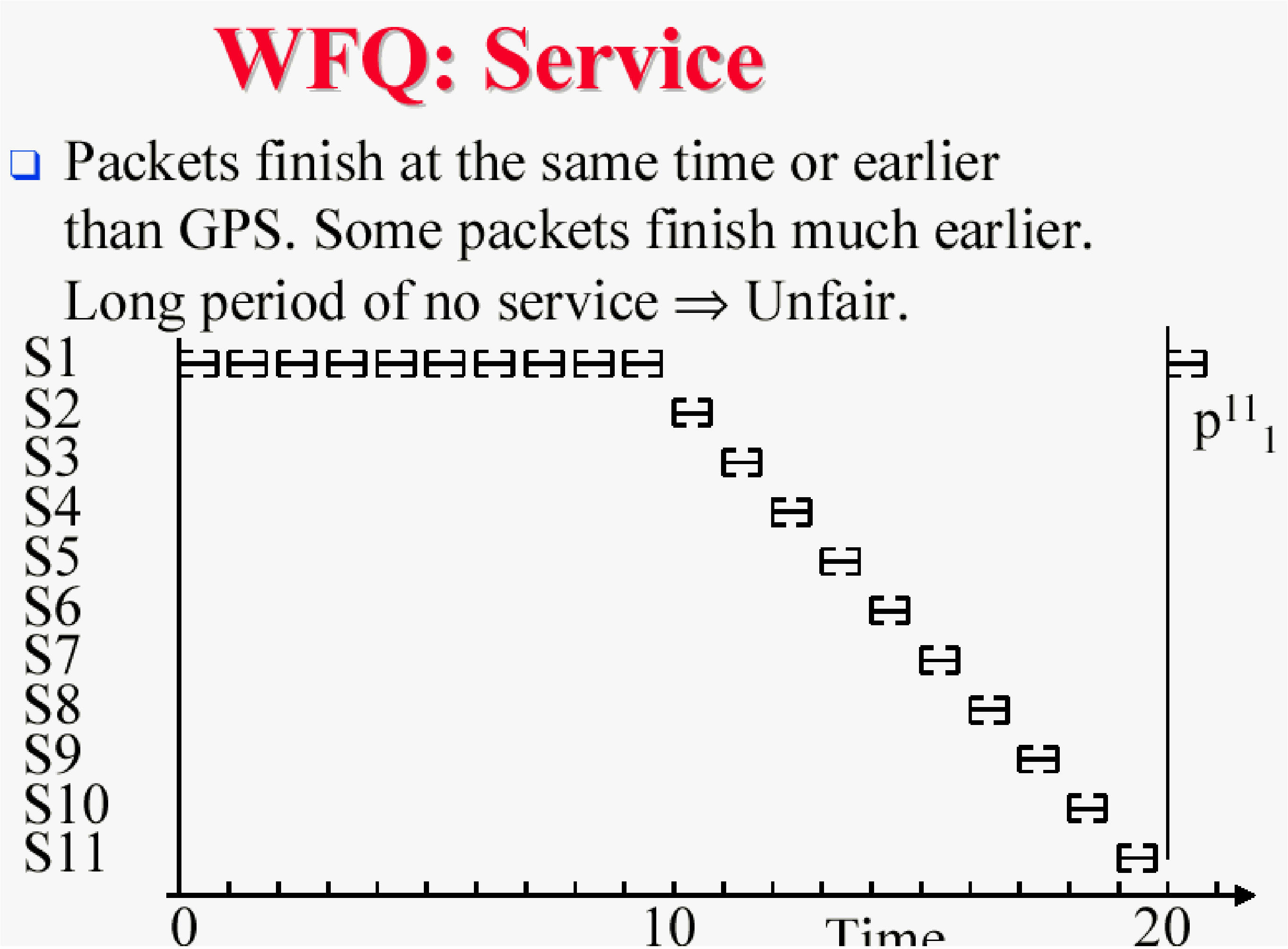
GPS scheduler: First 10 cells of source 1 finish service simultaneously with the cells of all the other sources. Then cell 11 of source 1 starts.



## GPS Example 2: Service

- Each cell of the first source takes 2 units of time. Sources 2-11 take 20 units each.

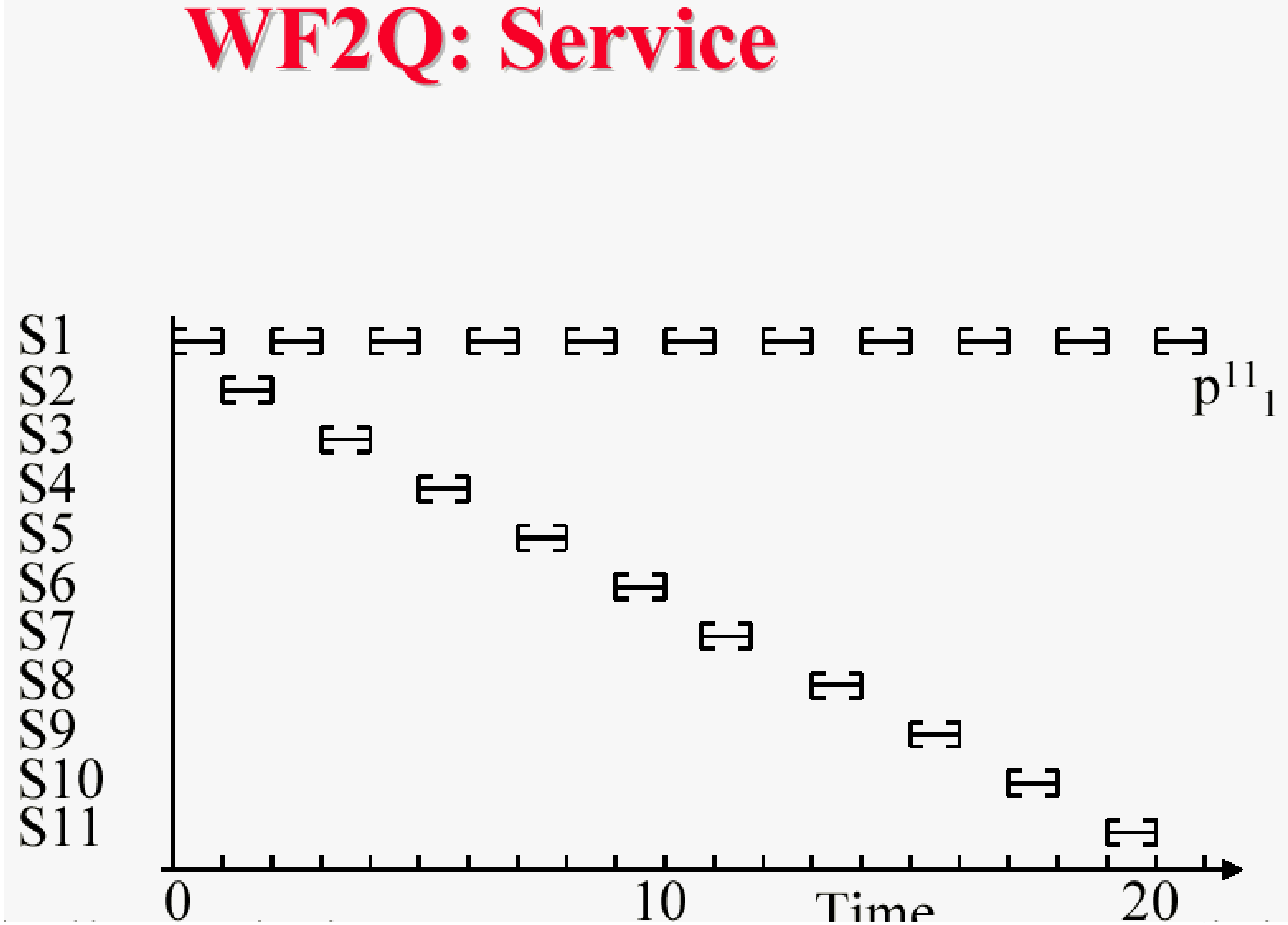
WFQ: serves the first 9 cell of source 1 before any cell of the other sources! Cell 10 of source 1 ends simultaneously (under GPS) with the other sources and can be sent before them (a tie). Cell 11 is sent after the cells of all the other sources.



## WFQ: Service

- ❑ Packets finish at the same time or earlier than GPS. Some packets finish much earlier. Long period of no service ⇒ Unfair.

WF$^2$Q: serves the first cell of source 1. But when it’s done (time 1), cell 2 of source 1 hasn’t started yet under GPS (it starts only at time 2), and so one of the other sources is served, an so on.



WF2Q: Service

# שאלה ממבחן (מועד ב' 2007)

- נתונה טבלה המתארת את זמן הגעת פקטות לערוץ ואת גודלן.
- הערוץ הוא בקצב 1 Kbyte/s והוא משותף לשלוש תחנות C, D, E.
- האם ישנן משקולות לתחנות C, D, E כך שחבילות 1 ו – 2 יסיימו שתיהן אחרי חבילות 3 ו – 4 תחת $WF^2Q$?
- אם כן, תן את המשקולות, אם לא הסבר מדוע.

| Packet# | Size (Kbyte) | Flow | זמן הגעה (sec) |
|---|---|---|---|
| 1 | 2 | C | 0 |
| 2 | 2 | C | 0 |
| 3 | 10 | D | 0 |
| 4 | 10 | E | 1 |

# שאלה ממבחן (מועד ב' 2007)

- We'll assign weights as follows C = 1, D = E = 10.
- GPS simulation
  - Time 0:
    - $F(C,1) = \left[R(t) + \frac{P(i,k)}{w_i}\right] = 0 + \frac{2}{1} = 2$
    - $F(C,2) = \left[F(i,k-1) + \frac{P(i,k)}{w_i}\right] = 2 + \frac{2}{1} = 4$
    - $F(D,1) = \left[R(t) + \frac{P(i,k)}{w_i}\right] = 0 + \frac{10}{10} = 1$
    - $R(t)$ progresses at rate 1/11 KB/s
  - Time 1 sec:
  - ` $R(t) = \frac{1}{11}$
    - $F(E,1) = \left[R(t) + \frac{P(i,k)}{w_i}\right] = \frac{1}{11} + \frac{10}{10} = 1\frac{1}{11}$
    - E starts service immediately (because up till now it was inactive).
    - The finish numbers are constant and will not change, so no need to continue the simulation.

# שאלה ממבחן (מועד ב' 2007)

- We’ll assign weights as follows C = 1, D = E = 10.
- WF²Q:
  - Time 0:
    - $F(C,1) = 2, F(C,2) = 4, F(D,1) = 1$
    - We serve connection D.
  - Time 10 sec:
    - $(D,1)$ finished service
    - The waiting packets are:
      - $F(C,1) = 2, F(C,2) = 4, F(E,1) = 1\frac{1}{11}$
      - $(E,1)$ has already started service under GPS at $t = 1$ sec
        - Because it arrived on an inactive connection
      - We service connection E
      - Only when E will finish, we will service connection C.