# Dynamic resource sharing

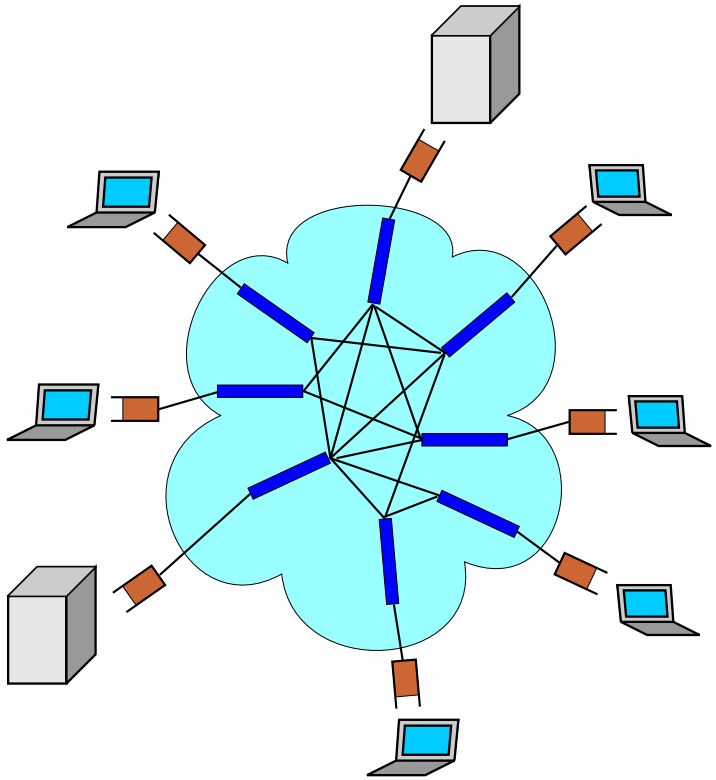
- In previous lectures we have studied different notions of fair resource sharing.
- Our focus in all these considerations has been on *static* resource sharing
  - there are given (fixed) numbers of flows on different routes
  - these flows compete for the resource (bandwidth) of the network and the resource sharing scheme determines how the resources actually are allocated to different flows
  - the sharing scheme appears crucial for the "service" experienced by different users
- Here we turn our attention on *dynamic* resource sharing
  - what is here dynamic is the bahaviour of users: flows come and go
  - at any given time there are some numbers of flows in progress on different routes; the resources at that time are shared between these flows as in the static case
  - but the new aspect to be accounted is that these numbers evolve in time stochastically as new flows arrive and when flow transfers are completed
- Our objective is to evaluate the user experienced performance in such a situation
  - the precise sharing mechanism no longer plays as crucial a role as before; the performance is to a large extent determined by the dynamic behaviour itself

## Flow level modelling of data networks

- Our interest here lies in modelling a *pure data network* (neglecting any real time traffic).
- A central concept is a *flow* which means a data object, such as a file or an item on the web page, that is transferred over the network.
- A flow is characterized by its
  - route across the network; flows having the same route constitute a flow class $k$
  - the number of classes is denoted $K$
  - size (in bytes); the mean size is denoted $\sigma$.
- As opposed to real time traffic, the data traffic
  - does not have strict packet level delay requirements
  - does not have an intrinsic rate; each flow tries to get as much capacity as possible
  - the actual bandwidth received by a flow any moment depends on the number of other concurrent flows and on the bandwidth sharing scheme
- What most interests the users is how long it takes to get a flow transferred.

## Flow level modelling paradigm

- This leads to a new flow-level modelling paradigm.
- The data of the flow is stored at the end devices (in their applications)
  - there is no flow-level buffering in the network
  - the data just flows through the network
- The network provides a shared server (bandwidth) for externally queued flows
  - the actual bandwidth each flow receives fluctuates stochastically as new flows arrive and other are completed
  - it is assumed that always when some flow arrives or departs the adjustment of the rates to the new situation is instantaneous



## Flow level modelling, cont.

- This modelling paradigm is an idealization of the reality
  - in the Internet bandwidth sharing is determined by the TCP congestion control algorithm; is neither ideally fair nor reacts immediately (initially governed by slow start)
  - the model is reasonable for *large* flows; fortunately, even though small flows are much more numerous, the large flows account most of the traffic carried in the network

## Basic quantities and the throughput

- Flows are divided into classes according to their routes.
- For class-$k$ flows we denote

$$\begin{cases} \lambda_k &= \text{arrival rate of flows [1/s]} \\ \sigma_k &= \text{mean size of the flow [bits]} \\ \rho_k &= \lambda_k\sigma_k \;=\; \text{the load of flow class } k \text{ [bits/s]} \\ T_k &= \text{mean transfer time of the flow [s]} \end{cases}$$

  Note that the $\rho_k$ here are not dimensionless but represent the mean bit rates of different traffic classes.

- The main performance parameter of interest to the user is the *throughput* $\gamma_k$ [bits/s]

$$\gamma_k = \frac{\sigma_k}{T_k}$$

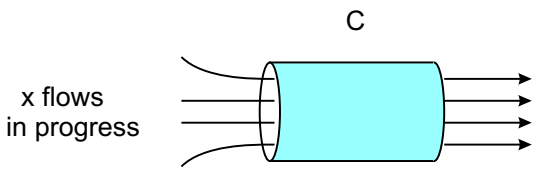
- This measures the overall bit rate experienced by the flow.

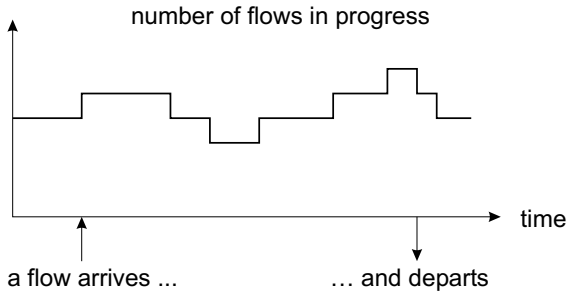
## Simple example: a processor sharing (PS) system

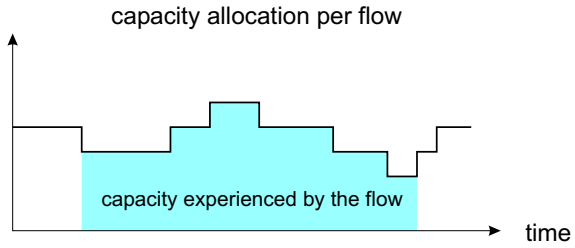
- Consider a simple system where there is only one link of capacity $C$ in the network.
- All flows go through this link
  - they belong to the same class (class index $k$ suppressed)
  - according to our assumptions, the capacity $C$ is always shared equally among the concurrent flows
- Our flow level model reduces to that of a PS queue
  - the queueing happens outside the network in the hosts
  - the link provides the server equally shared by all the flows
- According to the PS queue results

$$T = \frac{\sigma}{C-\rho} \quad \Rightarrow \quad \gamma = \frac{\sigma}{T} = C - \rho$$

- The throughput, i.e., the effective bandwidth experienced by the flows is equal to the free capacity left after the average load is subtracted from the link capacity.



## A general network

- In a general network, the situation is more complicated
  - the throughputs depend on how exactly the resources are shared between classes
- Let the state of the system be defined by the vector

$$\mathbf{x} = (x_1, x_2, \ldots, x_K)$$

  where $x_k$ is the number of class-$k$ flows in the system.
- Let $\phi_k(\mathbf{x})$ be the total capacity allocated to class $k$ in state $\mathbf{x}$
  - allocation per flow in class $k$ is $\phi_k(\mathbf{x})/x_k$
- Assuming that the flow sizes are exponentially distributed with means $\sigma_k$ the system is Markovian
  - the stationary state probabilities $\boldsymbol{\pi}(\mathbf{x}) = (\pi_1(x), \ldots, \pi_K(x))$ are determined by

$$\boldsymbol{\pi}\mathbf{Q} = \mathbf{0}, \quad \boldsymbol{\pi} \cdot \mathbf{e}^T = 1$$

    where $\mathbf{Q}$ is the transition rate matrix and $\mathbf{e} = (1, 1, \ldots, 1)$

## A general network, cont.

- In particular, the system constitutes a birth-death process with birth rates $\lambda_k$ and *state-dependent* death rates

$$\mu_k(\mathbf{x}) = \frac{1}{\sigma_k}\,\phi_k(\mathbf{x}) \qquad \text{for } x_k > 0, \text{ otherwise } 0$$

- The transition rate matrix is then completely determined

$$\begin{cases} q(\mathbf{x}, \mathbf{x}+\mathbf{e}_k) &= \lambda_k \\ q(\mathbf{x}, \mathbf{x}-\mathbf{e}_k) &= \mu_k(\mathbf{x}) \\ q(\mathbf{x}, \mathbf{x}) &= -\sum_k (\lambda_k + \mu_k(\mathbf{x})) \end{cases}$$

  where $\mathbf{e}_k$ is a vector with one at the component $k$ and 0 elsewhere; all other elements of $\mathbf{Q}$ are zero.

- Even though solving the balance equations (a set of linear equations) in principle is simple, in practice it is difficult when there are several traffic classes (due to the huge size of the state space, so-called state space explosion)

  – exact dynamic analysis of, e.g., maxmin fairness or proportional fairness is difficult

- In the following we'll consider specific allocation policies, *balanced allocations*, for which the solution is simpler. First, however, let us see how the throughput can be computed, if the state probabilities – whatever way – have been solved.

## Calculating the throughput

- Recalling the definition of the flow throughput as (mean flow size)/(mean transmission time), we can write

$$\gamma_k = \frac{\sigma_k}{T_k} = \frac{\lambda_k \sigma_k}{\lambda_k T_k} = \frac{\rho_k}{\mathrm{E}[X_k]}$$

  where the last form follows by applying Little's theorem, $\lambda_k T_k = \mathrm{E}[X_k]$.

- If the state probabilities $\pi(\mathbf{x})$ have been solved, then we can calculate

$$\mathrm{E}[X_k] = \sum_{\mathbf{x}} x_k \pi(\mathbf{x})$$

- Thus we have

$$\gamma_k = \frac{\rho_k}{\sum\limits_{\mathbf{x}} x_k \pi(\mathbf{x})}$$

## Balanced allocations

- Assume that the arrival rates $\lambda_k$ are given.
- One may ask whether it is possible to find such rate allocations $\phi_k(\mathbf{x})$ for different classes that the Markov process obeys *detailed balance* and has *given* state probabilities $\pi(\mathbf{x})$.
- Detailed balance means that the probability flows between any pair of states are equal
  - in a birth-death system there are transitions only between neighbouring states
  - detailed balance means that the probability flows between any pair of neigbouring states are balanced
- Writing the balance condition for states $\mathbf{x} - \mathbf{e}_k$ and $\mathbf{x}$

$$\lambda_k \, \pi(\mathbf{x} - \mathbf{e}_k) = \frac{1}{\sigma_k} \, \phi_k(\mathbf{x}) \, \pi(\mathbf{x})$$

  we see that the balance condition is satisfied if allocations $\phi_k(\mathbf{x})$ are chosen as follows

$$\phi_k(\mathbf{x}) = \frac{\rho_k \pi(\mathbf{x} - \mathbf{e}_k)}{\pi(\mathbf{x})}$$

## Balance function

- We define so-called *balance function* $\Phi(\mathbf{x})$

$$\Phi(\mathbf{x}) = \pi(\mathbf{x})/\rho_1^{x_1} \cdots \rho_K^{x_K} = \pi(\mathbf{x})/ \prod_{k=1}^{K} \rho_k^{x_k}$$

- In terms of the balance function the previous allocation can be written

$$\phi_k(\mathbf{x}) = \frac{\Phi(\mathbf{x} - \mathbf{e}_k)}{\Phi(\mathbf{x})}$$

- To summarize our results so far, one can choose arbitrary state probabilities $\pi(\mathbf{x})$, or which is the same thing, an arbitrary balance function $\Phi(\mathbf{x})$, and obtain a system which obeys detailed balance if one chooses the rate allocations according to the above equation.

- From the definition of the balance function, the stationary state probabilities are then

$$\pi(\mathbf{x}) = \Phi(\mathbf{x})\, \rho_1^{x_1} \cdots \rho_K^{x_K} = \Phi(\mathbf{x}) \prod_{k=1}^{K} \rho_k^{x_k}$$

- From the above allocation formula it follows that in a balanced system, the allocations at state $\mathbf{x}$ for different classes $k$ are proportional to $\Phi(\mathbf{x} - \mathbf{e}_k)$, i.e. the direction of the allocation vector is fixed,

$$\boldsymbol{\phi}(\mathbf{x}) \propto (\Phi(\mathbf{x} - \mathbf{e}_1), \ldots, \Phi(\mathbf{x} - \mathbf{e}_K))$$

.

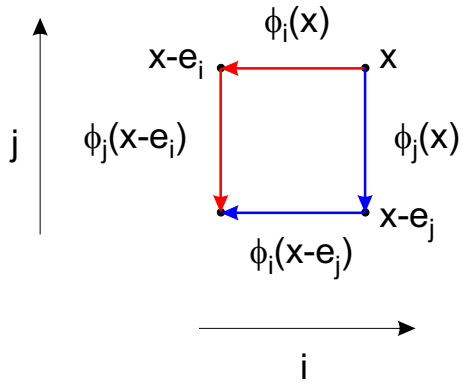
## Further properties of balanced allocations

- For a balanced network the following relation is true

$$\phi_i(\mathbf{x})\phi_j(\mathbf{x}-\mathbf{e}_i) = \phi_j(\mathbf{x})\phi_i(\mathbf{x}-\mathbf{e}_j)$$

since both sides are equal to $\Phi(\mathbf{x}-\mathbf{e}_i-\mathbf{e}_j)/\Phi(\mathbf{x})$.

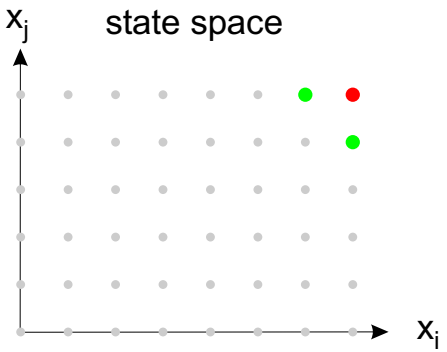
- This relation can be expressed as "left path"="right path".

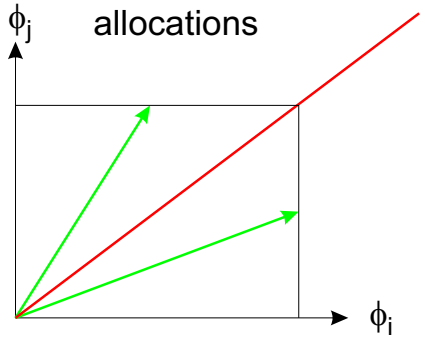



- In general, for any pair of paths having the same end points, the product of allocations along the paths (the class being defined by the direction of each segment) are equal.
- The above relation can also be written as

$$\frac{\phi_i(\mathbf{x})}{\phi_j(\mathbf{x})} = \frac{\phi_i(\mathbf{x}-\mathbf{e}_j)}{\phi_j(\mathbf{x}-\mathbf{e}_i)}$$

- This means that if we consider the projections of allocation vectors $\boldsymbol{\phi}$ on the $ij$-plane, then the projection of the allocation at $\mathbf{x}$ is related to those at points $\mathbf{x}-\mathbf{e}_i$ and $\mathbf{x}-\mathbf{e}_j$ as shown in the figure, i.e., the direction of the first one is determined by the latter ones.



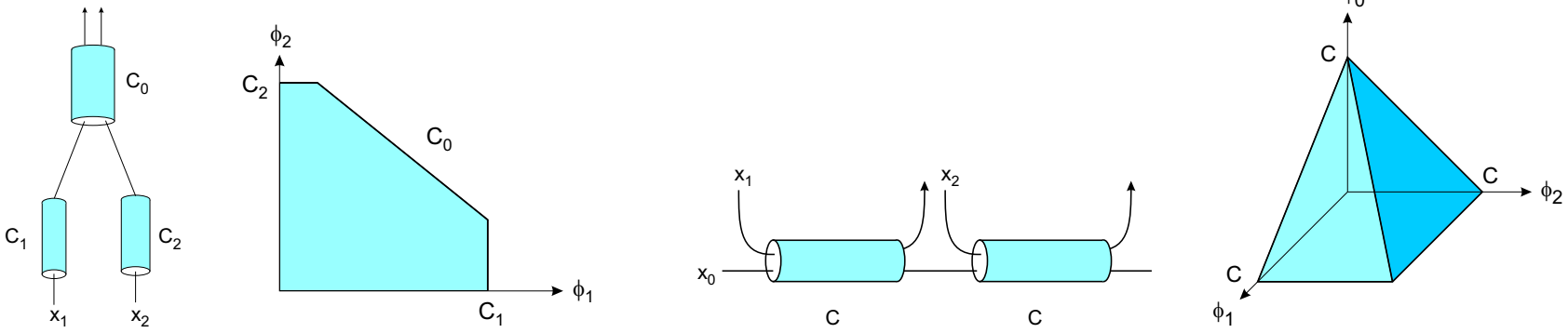



## Capacity set

- Up till now we have paid no attention to the physical constraints of the system.
- Not all allocations are feasible. In particular, the sum of allocations at any link $l$ cannot exceed its capacity $C_l$.
- An allocation vector $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_K)$ is feasible if for all links it holds that

$$\sum_{k \in \mathcal{F}_l} \phi_k \le C_l$$

  where $\mathcal{F}_l$ denotes the set of flows that go through link $l$.
- For any given problem, the set of all feasible capacity allocations, i.e., those satisfying the above constraints, is called the *capacity set* and denoted $\mathcal{C}$. Below are two examples of systems and their capacity sets: a 2-branch tree and a homogeneous line.



## Balanced fairness

- A new allocation scheme, so-called balanced fairness (BF), was introduced in 2002 by Thomas Bonald and Alexandre Proutière.
- Balanced fairness is an allocation scheme which is balanced, i.e. where the allocations can be derived from a balance function $\Phi(x)$ by the formula

$$\phi_k(\mathbf{x}) = \frac{\Phi_k(\mathbf{x} - \mathbf{e}_k)}{\Phi(\mathbf{x})}$$

  and where, in each state $x$ the allocation vector $\boldsymbol{\phi}(\mathbf{x})$ is *maximal* in the sense that it is at the boundary of $\mathcal{C}$ (making BF the *most efficient balanced allocation*).
- The balance function $\Phi(x)$ (and the allocation) is uniquely defined at each state $\mathbf{x}$:

$$\left(\frac{\Phi(\mathbf{x}-\mathbf{e}_1)}{\Phi(\mathbf{x})}, \ldots, \frac{\Phi(\mathbf{x}-\mathbf{e}_K)}{\Phi(\mathbf{x})}\right) \in \mathcal{C} \quad \Rightarrow \quad \Phi(\mathbf{x}) = \min\left\{\alpha : \left(\frac{\Phi(\mathbf{x}-\mathbf{e}_1)}{\alpha}, \ldots, \frac{\Phi(\mathbf{x}-\mathbf{e}_K)}{\alpha}\right) \in \mathcal{C}\right\}$$

## Recursion for the balance function

- If we want to determine $\Phi(\mathbf{x})$ and we are already given the values $\Phi(\mathbf{x}-\mathbf{e}_k)$ at states with lower occupancy, then the balance property $\phi_k(\mathbf{x}) = \Phi_k(\mathbf{x}-\mathbf{e}_k)/\Phi(\mathbf{x})$ fixes the *direction* of the allocation vector $\boldsymbol{\phi}(\mathbf{x})$.
- In order to make the allocation maximal, the multiplier $1/\Phi(\mathbf{x})$ must be made as large as possible, i.e., $\Phi(\mathbf{x})$ as small as possible.
- Therefore we have the *recursion*

$$\Phi(\mathbf{x}) = \min\left\{\alpha:\ \left(\frac{\Phi(\mathbf{x}-\mathbf{e}_1)}{\alpha},\ldots,\frac{\Phi(\mathbf{x}-\mathbf{e}_K)}{\alpha}\right)\in\mathcal{C}\right\}$$

- The initial value $\Phi(\mathbf{0})$ is immaterial (does not affect the allocations), and can be set arbitrarily $\Phi(\mathbf{0}) = 1$.
- With this choice, the above recursion defines the balance function uniquely for every state, and hence also the allocations are uniquely defined.
- Note, however, that with this choice the expression for $\pi(\mathbf{x})$ has to be normalized

$$\pi(\mathbf{x}) = \frac{1}{G(\boldsymbol{\rho})}\,\Phi(\mathbf{x})\rho_1^{x_1}\cdots\rho_K^{x_K}, \qquad \text{with} \quad G(\boldsymbol{\rho}) = \sum_{\mathbf{x}}\Phi(\mathbf{x})\rho_1^{x_1}\cdots\rho_K^{x_K}$$

## BF recursion more explicitly

- The allocation vector must be such that for each link $l$ we have

$$\sum_{k\in\mathcal{F}_l} \phi_k(\mathbf{x}) \equiv \sum_{k\in\mathcal{F}_l} \frac{\Phi(\mathbf{x}-\mathbf{e}_k)}{\Phi(\mathbf{x})} \le C_l$$

- For each link $l$, this sets a lower bound for $\Phi(\mathbf{x})$

$$\Phi(\mathbf{x}) \ge \frac{1}{C_l} \sum_{k\in\mathcal{F}_l} \Phi(\mathbf{x}-\mathbf{e}_k) \quad \text{for all } l$$

- Thus the smallest possible value for $\Phi(\mathbf{x})$ is the maximum of these lower bounds

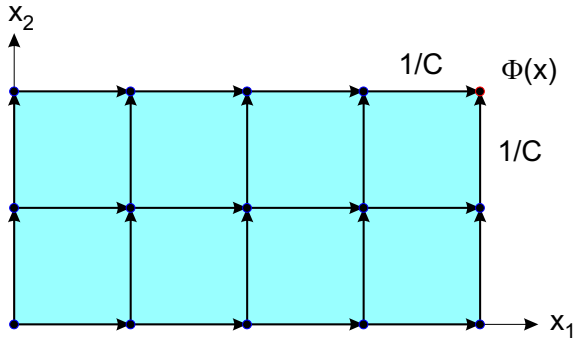
$$\Phi(\mathbf{x}) = \max_l \frac{1}{C_l} \sum_{k\in\mathcal{F}_l} \Phi(\mathbf{x}-\mathbf{e}_k)$$

- This is the BF recursion (with the initial value $\Phi(\mathbf{0}) = 1$).
- The link that realizes this maximum is saturated (capacity fully utilized).
- To emphasize the importance of this result and the idea behind balanced fairness, we reiterate that the values of $\Phi(\mathbf{x})$ (and, therfore, of $\pi(\mathbf{x})$) are obtained *recursively*, with one pass through the state space. One need not solve linear equations $\boldsymbol{\pi}\mathbf{Q} = 0$, $\boldsymbol{\pi}\cdot\mathbf{e}^T = 1$.

## Example 1. Two classes sharing a single link

- A single link of capacity $C$ shared by two classes.
- The recursion equation

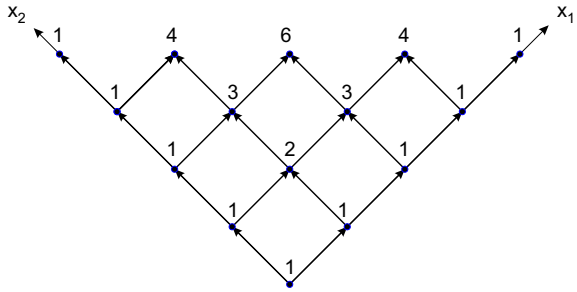
$$\Phi(\mathbf{x}) = \frac{1}{C}\left(\Phi(\mathbf{x}-\mathbf{e}_1) + \Phi(\mathbf{x}-\mathbf{e}_2)\right)$$




- The balance function

$$\Phi(\mathbf{x}) \;=\; (\text{number of paths}) \times \frac{1}{C^{(\text{path length})}}$$

$$\;=\; \binom{x_1+x_2}{x_1} \frac{1}{C^{x_1+x_2}}$$




- To understand why the binomial coefficient appears, compare with the Pascal triangle (see lower figure).
- Capacity allocation:

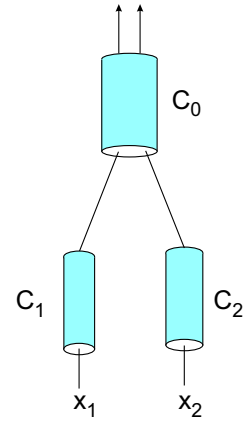
$$\phi_i(\mathbf{x}) = \frac{\Phi(\mathbf{x}-\mathbf{e}_i)}{\Phi(\mathbf{x})} = \frac{x_i}{x_1+x_2}C, \;\; i = 1,2$$

- Per flow the allocated capacity is the same $\frac{C}{x_1+x_2}$ for all flows: a single PS system.

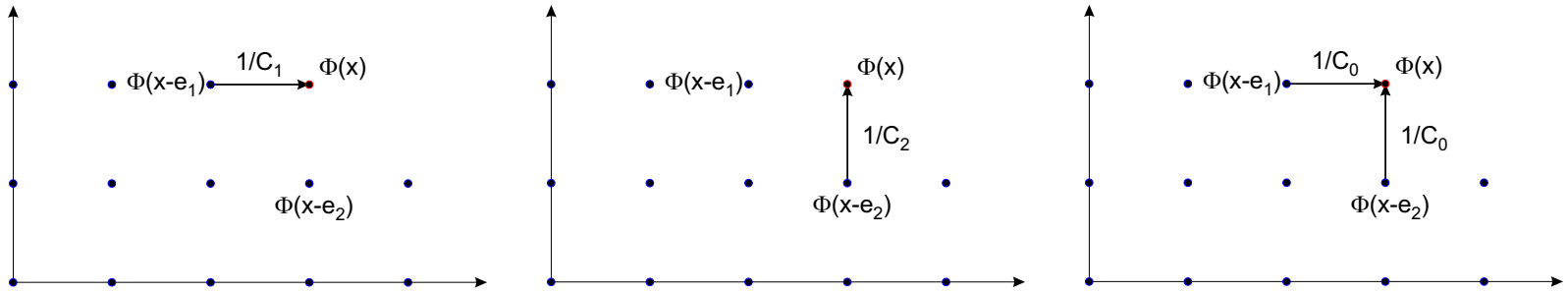
## Example 2. Tree network

- The recursion

$$\Phi(\mathbf{x}) = \max_l \left\{ \frac{1}{C_l} \sum_{k \in \mathcal{F}_l} \Phi(\mathbf{x} - \mathbf{e}_k) \right\}$$



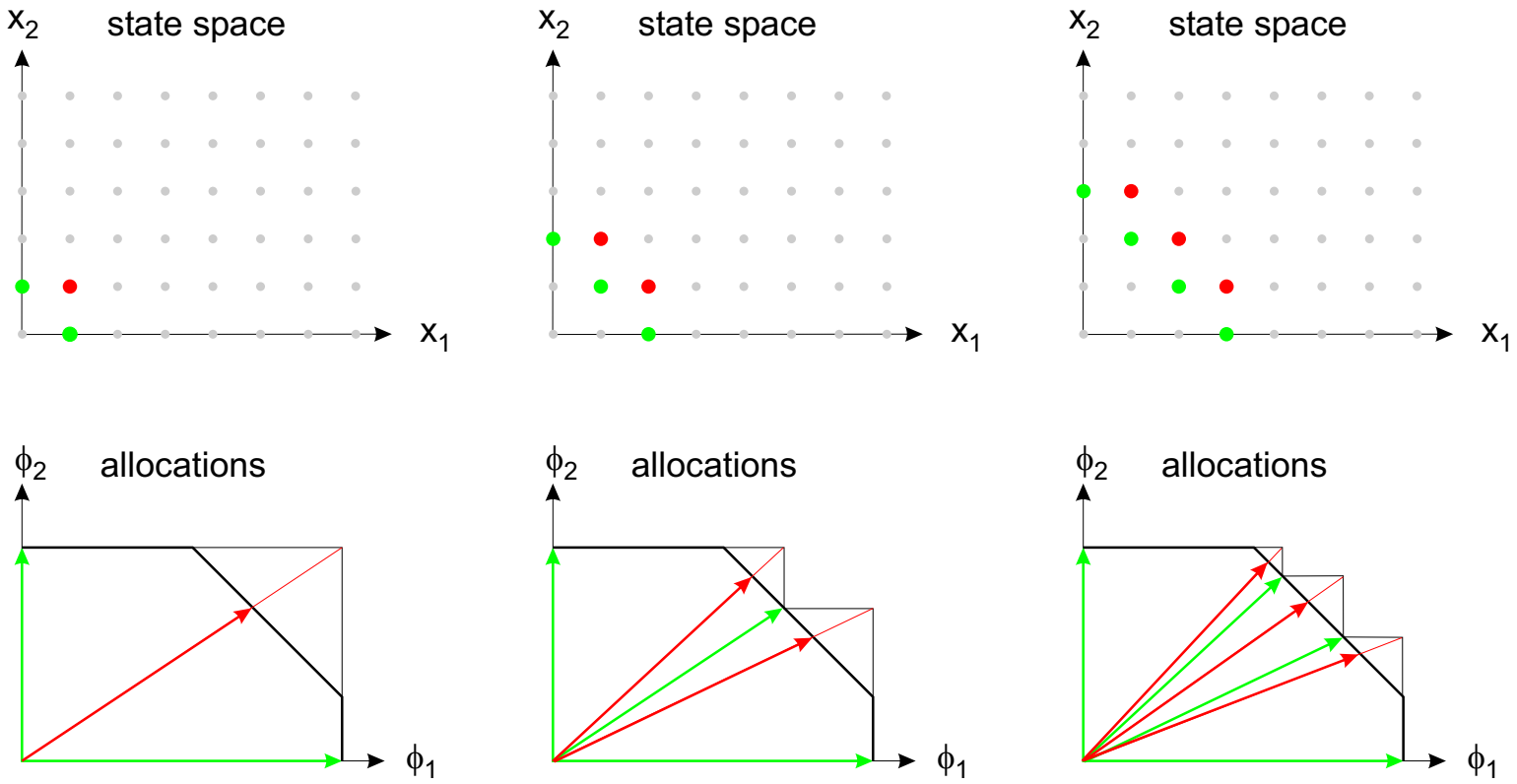

- In the case of a 2-branch tree the maximum of these:




- Next we'll demonstrate that in all interior points ($x_1 > 0,\ x_2 > 0$) the third case applies, i.e., the allocation is limited by the capacity constraint of the common trunk $C_0$.

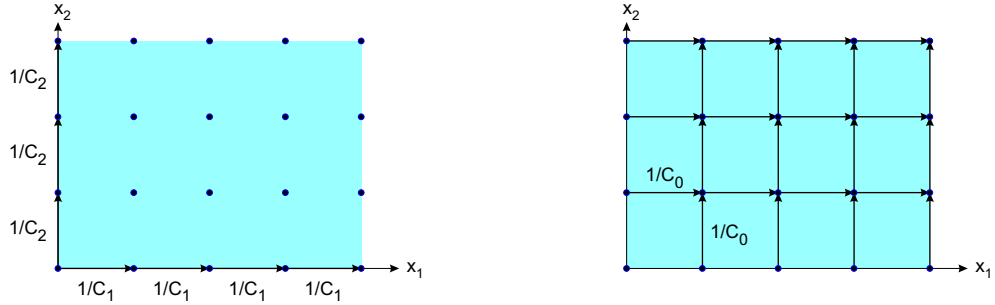
## Tree network, cont.

- We give a "graphical proof" for the saturation of the common trunk in all interior points.
- The proof is based on earlier remark how the direction of allocation vector at $\mathbf{x}$ is determined by those at points $\mathbf{x} - \mathbf{e}_1$ and $\mathbf{x} - \mathbf{e}_2$ (in the present 2-dimensional case the projections of the allocation vectors onto the $(\phi_1, \phi_2)$-plane are the vectors themselves).



## Tree network, cont.

- We have now established that the recursion for the balance function goes as follows.

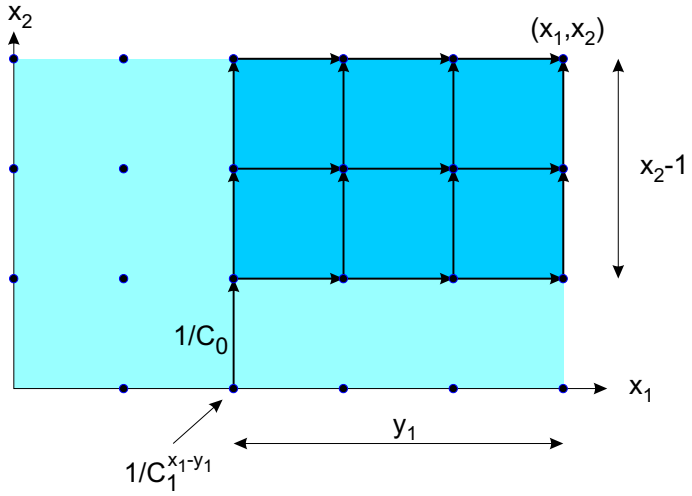



- Thus for the states on the $x_1$-axis and $x_2$-axis, respectively, we have

$$\Phi(x_1, 0) = 1/C_1^{x_1}, \qquad \Phi(0, x_2) = 1/C_2^{x_2}$$

- For the interior states $x_1 > 0,\ x_2 > 0$, the balance function can be calculated starting from the points on one of the axes. The figure illustrates a "source point" on $x_1$-axis. It contributes to the balance function at $(x_1, x_2)$ by the amount

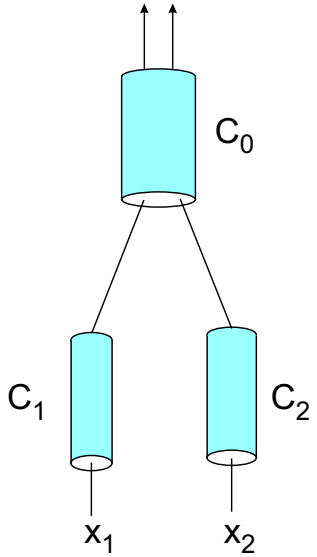
$$\binom{y_1 + x_2 - 1}{y_1} \frac{1}{C_1^{x_1 - y_1}} \frac{1}{C_0^{x_2 + y_1}}$$

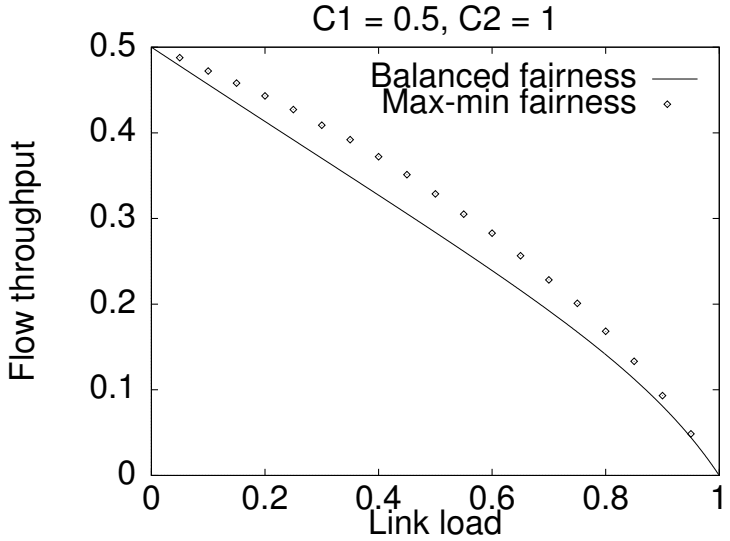


## Tree network, cont.

- Summing over all source points on the $x_1$-axis as well as on the $x_2$-axis, we get (to keep the expression simpler we set $C_0 = 1$, i.e. $C_0$ is used as the unit of capacity).

$$\Phi(\mathbf{x}) = \sum_{y_1=0}^{x_1-1} \binom{y_1 + x_2 - 1}{y_1} \frac{1}{C_1^{x_1-y_1}} + \sum_{y_2=0}^{x_2-1} \binom{y_2 + x_1 - 1}{y_2} \frac{1}{C_1^{x_2-y_2}}$$



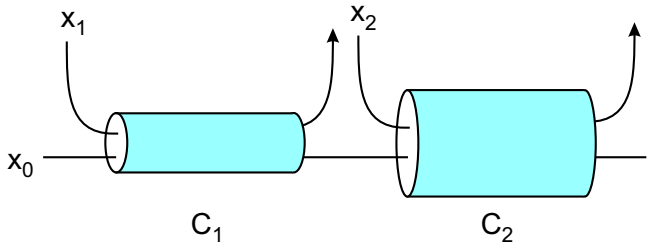



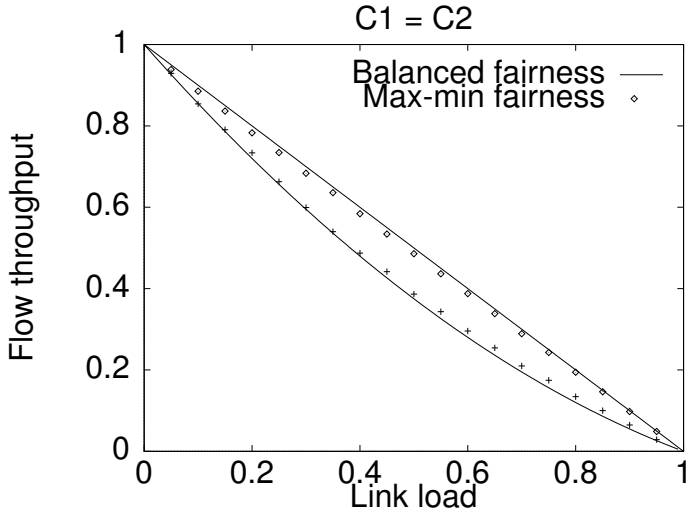

- The balance function can be similarly derived for a general tree.
- We will later see how a throughput curve like the one shown here can be calculated even without solving explicitly $\Phi(\mathbf{x})$.

## Example 3. Heterogeneous line

- By similar considerations one can derive the balance function for heterogeneous lines in closed form (minimum of the link capacities is used as the unit of bandwidth):

$$\Phi(\mathbf{x}) = \sum_{y_1+y_2 \le x_0} \binom{x_1+y_1-1}{y_1} \binom{x_2+y_2-1}{y_2} \frac{1}{C_1^{x_1+y_1} C_2^{x_2+y_2}}$$



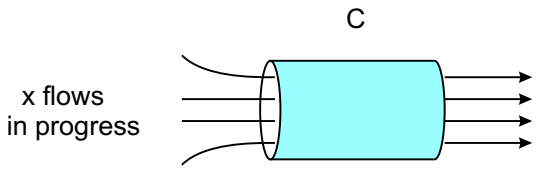

- It should be emphasized, that though in these examples we have concentrated on finding the balance function analytically, it is always possible to find the values numerically by the basic recursion.

## Throughput in terms of the state sum (normalization constant)

- Previously we found that
$$\gamma_k = \frac{\sigma_k}{T_k} = \frac{\rho_k}{\mathrm{E}[X_k]}$$
- Recall the definition of the normalization constant, also called the state sum
$$G(\boldsymbol{\rho}) = \sum_{x_1=0}^{\infty} \cdots \sum_{x_K=0}^{\infty} \Phi(\mathbf{x})\, \rho_1^{x_1} \cdots \rho_K^{x_K}$$
- The mean occupancy $\mathrm{E}[X_k] = \Sigma_{\mathbf{x}}\, x_k \pi(\mathbf{x})$ can be expressed in terms of $G(\boldsymbol{\rho})$,
$$\mathrm{E}[X_k] = \frac{1}{G(\boldsymbol{\rho})} \sum_{x_1=0}^{\infty} \cdots \sum_{x_K=0}^{\infty} x_k \Phi(\mathbf{x})\, \rho_1^{x_1} \cdots \rho_K^{x_K} = \frac{1}{G(\boldsymbol{\rho})}\, \rho_k \frac{\partial}{\partial \rho_k} G(\boldsymbol{\rho}) = \rho_k \frac{\partial}{\partial \rho_k} \log G(\boldsymbol{\rho})$$
- Thus we have the result
$$\gamma_k = \frac{G(\boldsymbol{\rho})}{\frac{\partial}{\partial \rho_k} G(\boldsymbol{\rho})} = \frac{1}{\frac{\partial}{\partial \rho_k} \log G(\boldsymbol{\rho})}$$
- So, if we know how to find $G(\boldsymbol{\rho})$ then we can calculate the throughputs $\gamma_k$.

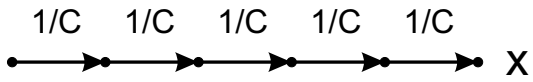
## Calculating the throughput via the state sum

- Consider first the trivial example of a single link shared equally between the flows
- In this case we have




$$\Phi(x) = \frac{1}{C^x}$$

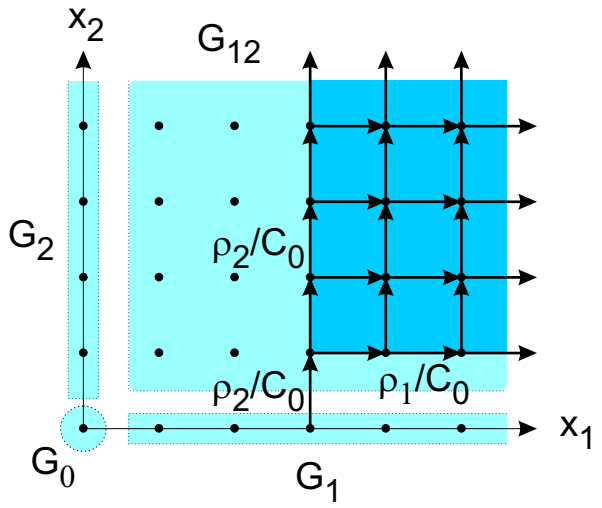
$$G(\rho) = \sum_{x=0}^{\infty} \Phi(x)\rho^x = \sum_{x=0}^{\infty} (\rho/C)^x = \frac{C}{C-\rho}$$




from which we regain the familiar result

$$\gamma = \frac{1}{\frac{d}{d\rho}\log G(\rho)} = C - \rho$$

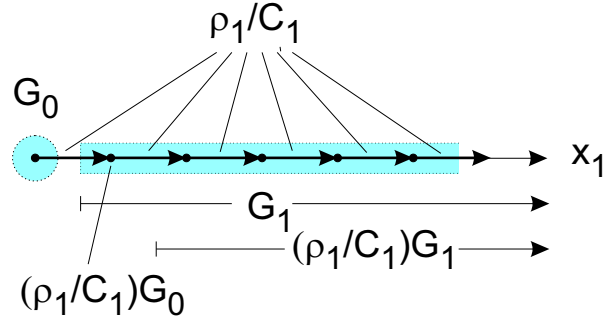
## Calculating the troughputs for a 2-brach tree




- Consider again the same 2-branch tree as before.
- We divide the 2-dimensional state space into four parts as illustrated in the picture: the origin, positive $x_1$-axis ($x_1 > 0$), positive $x_2$-axis ($x_2 > 0$), and positive $(x_1, x_2)$-quadrant ($x_1 > 0,\ x_2 > 0$).
- Correspondingly, the state sum is divided into four parts

$$G(\boldsymbol{\rho}) = G_0(\boldsymbol{\rho}) + G_1(\boldsymbol{\rho}) + G_2(\boldsymbol{\rho}) + G_{12}(\boldsymbol{\rho})$$

- In each of the four parts a simple recursion with constant coefficients applies. This allows us to find simple recursion between the four numbers $G_0,\ G_1,\ G_2,\ G_{12}$.
  - note that now write the recursion for the function $\Phi(\mathbf{x})\rho_1^{x_1}\rho_2^{x_2}$ directly, therefore we have the $\rho_1$- and $\rho_2$-factors in the recursion coefficients




- From the figure: $G_i(\boldsymbol{\rho}) = \frac{\rho_i}{C_i} G_0(\boldsymbol{\rho}) + \frac{\rho_i}{C_i} G_i(\boldsymbol{\rho})$. Thus,

$$G_i(\boldsymbol{\rho}) = \frac{\frac{\rho_i}{C_i} G_0(\boldsymbol{\rho})}{1 - \frac{\rho_i}{C_i}}, \quad i = 1, 2$$

## Recursion (continued . . . )

- Similarly $G_{12}(\boldsymbol{\rho})$ can be expressed in terms of $G_1(\boldsymbol{\rho})$ and $G_2(\boldsymbol{\rho})$.
- Each state on positive $i$-axis contributes to $G_{12}(\boldsymbol{\rho})$ as a "source of recursion" an amount that is its own weight times

$$\frac{\rho_j}{C_0} \cdot \frac{1}{1 - (\frac{\rho_1}{C_0} + \frac{\rho_2}{C_0})}, \qquad j = 2 - i + 1 \text{ (the other class)}$$

- The first factor comes from the "bridge".
- The second factor from the infinite sum $S$ over the area in dark blue (see next slide).
- Since the contribution is the same for each state on $i$-axis, in total the whole positive $i$ gives rise to a contribution to $G_{12}(\boldsymbol{\rho})$ of magnitude

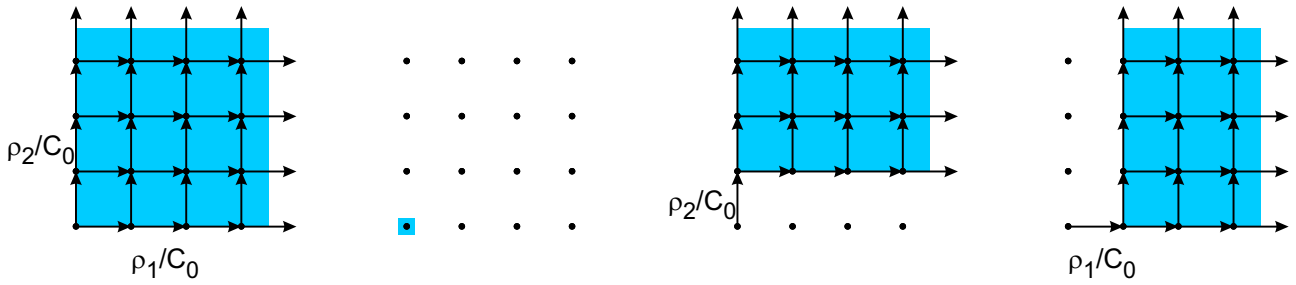
$$\frac{\rho_j}{C_0} \cdot \frac{1}{1 - (\frac{\rho_1}{C_0} + \frac{\rho_2}{C_0})} \cdot G_i(\boldsymbol{\rho})$$

- Adding the contributions from both axes we get

$$G_{12}(\boldsymbol{\rho}) = \frac{\rho_2 G_1(\boldsymbol{\rho}) + \rho_1 G_2(\boldsymbol{\rho})}{C_0 - (\rho_1 + \rho_2)}$$

## Recursion (continued . . . )

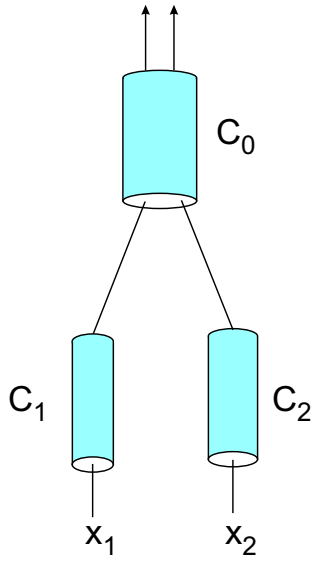
- The infinite sum is evaluated in the same way as the sum of geometric series:
  $S = 1 + q + q^2 + \ldots = 1 + q\,(1 + q + q^2 + \ldots) = 1 + q\,S \;\Rightarrow\; S = 1/(1-q)$.
- But now we have recursion in two dimensions:




$$S \quad = \quad 1 \quad + \quad \frac{\rho_2}{C_0}\,S \quad + \quad \frac{\rho_1}{C_0}\,S$$

$$S \;=\; \frac{1}{1 - (\frac{\rho_1}{C_0} + \frac{\rho_2}{C_0})}$$

## Example 1: 2-branch tree



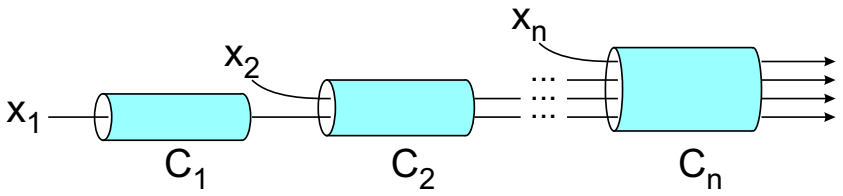

$$\begin{cases} G_0(\boldsymbol{\rho}) &= 1 \\ G_1(\boldsymbol{\rho}) &= \dfrac{\frac{\rho_1}{C_1}\cdot G_0(\boldsymbol{\rho})}{1-\frac{\rho_1}{C_1}} \\ G_2(\boldsymbol{\rho}) &= \dfrac{\frac{\rho_2}{C_2}\cdot G_0(\boldsymbol{\rho})}{1-\frac{\rho_2}{C_2}} \\ G_{12}(\boldsymbol{\rho}) &= \dfrac{\frac{\rho_1}{C_0}\cdot G_2(\boldsymbol{\rho})+\frac{\rho_2}{C_0}\cdot G_1(\boldsymbol{\rho})}{1-\frac{\rho_1+\rho_2}{C_0}} \end{cases}$$

$$G(\boldsymbol{\rho}) = G_0(\boldsymbol{\rho}) + G_1(\boldsymbol{\rho}) + G_2(\boldsymbol{\rho}) + G_{12}(\boldsymbol{\rho}) = \frac{1}{1-\frac{\rho_1+\rho_2}{C_0}} \cdot \left( \frac{1-\frac{\rho_1}{C_0}}{1-\frac{\rho_1}{C_1}} + \frac{1-\frac{\rho_2}{C_0}}{1-\frac{\rho_2}{C_2}} - 1 \right)$$

$$\gamma_k = \frac{G(\boldsymbol{\rho})}{\frac{\partial}{\partial \rho_k} G(\boldsymbol{\rho})}$$

## Example 2: Parking lot topology

- In a similar way one can derive the state sum for the so-called parking lot topology.




- The recursion for the pieces of the state sum is (here $G_1(\boldsymbol{\rho})$ refers to the state sum over states $x_1 \geq 0,\ x_2 = 0, \ldots,\ x_n = 0$ and $G_k(\boldsymbol{\rho})$ refers to the state sum over states $x_1 \geq 0,\ \ldots,\ x_{k-1} \geq 0,\ x_k > 0,\ x_{k+1} = 0, \ldots,\ x_n = 0$)

$$\begin{cases} G_1(\boldsymbol{\rho}) &= \frac{1}{1-\frac{\rho_1}{C_1}} \\ G_k(\boldsymbol{\rho}) &= \left(1 + \frac{\frac{\rho_k}{C_k}}{1-\frac{\rho_1+\cdots+\rho_k}{C_k}}\right) \cdot G_{k-1}(\boldsymbol{\rho}) \;=\; \frac{1-\frac{\rho_1+\cdots+\rho_{k-1}}{C_k}}{1-\frac{\rho_1+\cdots+\rho_k}{C_k}} \cdot G_{k-1}(\boldsymbol{\rho}) \end{cases}$$

- By denoting the link $k$ load by $R_k = \Sigma_{j=1}^{k} \rho_j$, the solution for the state sum reads

$$G(\boldsymbol{\rho}) = \frac{1}{1-\frac{R_1}{C_1}} \cdot \frac{1-\frac{R_1}{C_2}}{1-\frac{R_2}{C_2}} \cdots \frac{1-\frac{R_{n-1}}{C_n}}{1-\frac{R_n}{C_n}}$$

- From this the throughput can be calculated by derivation

$$\gamma_k = \frac{G(\boldsymbol{\rho})}{\frac{\partial}{\partial \rho_k} G(\boldsymbol{\rho})} = \left(\frac{1}{C_k - R_k} + \sum_{l=k+1}^{n} \left(\frac{1}{C_l - R_l} - \frac{1}{C_l - R_{l-1}}\right)\right)^{-1}$$

## Some properties of balanced fairness

- Broadly speaking, BF represents in a general network setting with many shared resources the closest counterpart of simple processor sharing of a single resource, inheriting many of its nice properties.
- BF is not an utility-based allocation
  - in utility-based allocations, given a set of flows, there is some utility for each flow depending on the bandwidth allocated to the flow, and the allocation is made such that the total utility is maximized
  - BF cannot be obtained as a solution to this kind of utility maximization for any utility function (except in some very simple topologies)

*Stability*

- Under balanced fairness the system is stable iff the load vector $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_K)$ is inside the capacity set $\mathcal{C}$, $\boldsymbol{\rho} \in \mathcal{C}$.
- Among all balanced allocations, BF is the most effective in the sense that the idle system probability is largest.

## Properties of balanced fairness, cont.

*Insensitivity*

- The state distribution $\pi(\mathbf{x}) = \Phi(\mathbf{x})\,\rho_1^{x1} \cdots \rho_K^{x_K}$ depends on the file size distributions only through their means $\sigma_k$; the same $\pi(\mathbf{x})$ applies no matter what the shape of the file size distribution is
  - consequently, all performance measures, such as the throughput, that can be derived from the state distribution are insensitive
- The state distribution depends on the arrival rates $\lambda_k$ and mean file sizes $\sigma_k$ only through their products, the loads $\rho_k = \lambda_k \sigma_k$
  - one can arbitrarily change the arrival rates and file sizes in different classes, as far as their product is kept constant, and always we have the same state distribution $\pi(\mathbf{x})$
- In contrast, the utility-based sharing schemes, such as maxmin or proportional fairness, are sensitive, unless the happen to be identical with balanced fairness (in some simple topologies, proportional fairness coincides with balanced fairness).

## Properties of balanced fairness, cont.

*Conditional transfer time*

- BF shares with PS also the property that the expected transfer time of a flow, given its precise size $s$, is proportional to $s$. For class-$k$ flows,

$$\mathrm{E}\left[T_k \,|\, S_k = s\right] = s/\gamma_k$$

*Implementation*

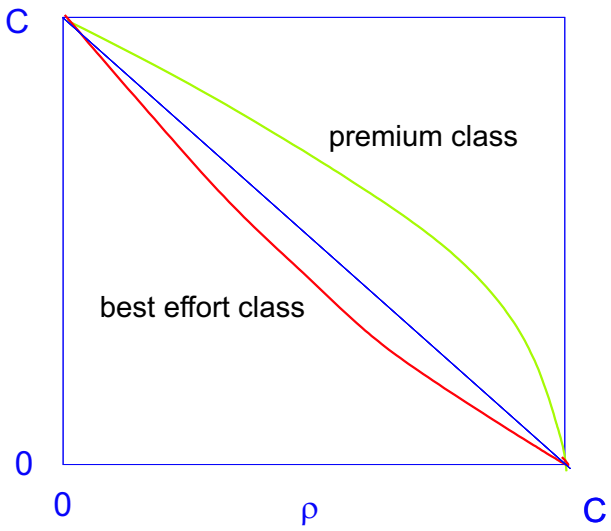
- BF is a rather theoretical concept, its realization in practice may not be easy
  - no decentralized algorithm is known for realizing BF; it needs full centralized control, which may be impractical
  - for some utility-based allocations, maxmin and proportional fairness, decentralized algorithms have been proposed

## Comparison with other allocation schemes

- Though BF is a rather theoretical concept, it is in many ways a very useful one.
- It turns out that for many systems BF approximates rather well the performance of the utility based allocations, e.g., maxmin fairness and proportional fairness
  - because of the simple recursion, BF is, however, easier to analyze
  - for some systems, as we have seen, even an exact analytical solution can be found
- For some systems one can associate slightly modified systems which under BF give either an upper or lower bound for the performance of the original system.

## Final remarks on dynamic bandwidth sharing

- In the very beginning we noted that in the dynamic setting, where flows come and go, the precise way resources are allocated is not crucial; more important is that the number of other flows with which the resources have to be shared varies stochastically.
- One indication of this is that BF has similar performance as maxmin and proportional fairness.
- Here we give another example, where a single resource, a link, is shared by flows divided into two classes, premium class and best effort class. The bandwidth is shared between all the flows with weights, so that the flows in the best effort class have a weight 1 while those in the premium class have a weight 10.
- While the throughputs of the classes are clearly different, the difference is not as dramatic as the big difference in the weights might suggest. Only when the load approaches 1, does the ratio of the throughputs tend to 10.



## Summary

- For data traffic the most important performance metric is the flow throughput.
- To a great extent it is determined by the dynamic sharing of resources between stochastically varying number of flows.
- The dynamics can analyzed by an idealized model, where the flows are buffered at the sources and the network is a shared server providing bandwidth to different flows; the resource sharing is readjusted instantaneously at every arrival or departure of a flow.
- While a Markovian analysis is possible for any sharing scheme whenever the flow sizes are exponentially distributed, such an analysis becomes excessively difficult when the number of flow classes increases.
- Balanced fairness is a novel bandwidth sharing concept, which allows calculation of the state probabilities recursively, without solving a set of equations. In some simple systems analytic evaluation is feasible.
- Under balanced fairness, the performance of the system is insensitive.
- While balanced fairness may be difficult to realize in practice, the concept can be used as a calculational approximation tool to analyze the performance of other fairness concepts (maxmin, proportional fairness etc) in any given network topology.