

Calculating the flow level performance of balanced fairness in tree networks

Thomas Bonald, Jorma Virtamo

▶ **To cite this version:**

Thomas Bonald, Jorma Virtamo. Calculating the flow level performance of balanced fairness in tree networks. Performance Evaluation, 2004, 10.1016/j.peva.2004.03.001 . hal-01272523

HAL Id: hal-01272523
https://hal.science/hal-01272523

Submitted on 11 Feb 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Calculating the flow level performance of balanced fairness in tree networks

T. Bonald[1] and J. Virtamo[2]

[1]France Telecom R&D
38-40, rue du Général Leclerc
92794 Issy-les-Moulineaux Cedex 9, France

[2]Networking Laboratory
Helsinki University of Technology
P.O. Box 3000, FIN-02015 HUT, Finland

January 6, 2004

Abstract

We consider a tree network whose resources are shared by a dynamically varying number of elastic flows. We present an efficient method for calculating performance metrics such as flow throughputs when the resource allocation is balanced fair. The method is based on a recursive algorithm for computing the normalization constant of the stationary distribution. Several examples are worked out. A proof is given for the Pareto efficiency of balanced fairness in tree networks.

Keywords: balanced fairness, recursion for normalization constant, flow throughput, tree network

1 Introduction

Most traffic in today's Internet is generated by the transfer of documents such as files or web pages. This traffic is elastic in that the duration of each transfer depends on network congestion. Each document is split into a sequence of packets, referred to as a flow, whose sending rate is adapted in response to congestion indications such as packet losses, typically under the control of TCP. The quality of the transfer then depends on the time required to successively transfer all the packets of the flow. In this sense, network performance for

elastic traffic is mainly manifested at flow level and can be gauged by measures such as the average per flow throughput.

Performance studies for elastic traffic have traditionally focussed on the packet level behaviour of various congestion control algorithms, including TCP. While the design of an efficient congestion control is a critical issue, it may be argued that the average per flow throughput on a given network route depends more significantly on the number of flows sharing the corresponding links, which varies as new flows are generated and others cease. To analyze flow level behaviour in such a dynamic setting, idealized models are needed. An appropriate abstraction in this context entails entirely disregarding packet level phenomena and considering the flow content as a fluid which is transmitted as a continuous stream through the network. It is assumed in this fluid model that rate changes occur instantaneously and simultaneously on all links on every flow arrival or departure. We assume flows occur in sessions consisting of a succession of flows and separating "think times" and that sessions occur as a Poisson process.

As there are many flows with different routes being transmitted simultaneously, the evolution of the number of flows depends on how network resources are allocated. Most work has focussed on so-called utility based allocations, where bandwidth is shared so as to maximise some utility function of the instantaneous flow rates [8]. Examples of such allocations are classical max-min fairness [1] and Kelly's proportional fairness [7]. Crucially, the optimality of utility based allocations is defined for a static composition of flows. If the true random nature of traffic were taken into account, it would be necessary to define utility in terms of the performance of individual finite duration flows. In this case, it is not obvious that max-min or proportional fairness are optimal in any real sense. In random traffic, performance and therefore utility depend in general on the precise statistics of offered traffic and are virtually impossible to evaluate analytically.

An alternative notion of fairness, called "balanced fairness", has been introduced by Bonald and Proutière [2]. When flows share bandwidth with balanced fairness, performance is insensitive to detailed traffic characteristics such as the distribution of flow sizes and think-time durations. The name derives from the set of detailed balance relations satisfied by the instantaneous rates allocated to individual flows, which constitute necessary and sufficient conditions for insensitivity in the underlying stochastic networks [11, 3]. The insensitivity is such that the stationary distribution of the number of flows in progress, and consequently the average per flow throughput, depend only on the expected traffic offered on each route. Balanced fairness thus generalizes to a network context the insensitivity of bandwidth sharing of an isolated bottleneck link introduced in [4].

Insensitivity is the key to simple and robust performance results. Network dimensioning rules can be developed based on traffic intensity forecasts only, independently of the complex traffic structure which is continually evolving as new applications gain popularity. Balanced fairness can thus be viewed as a bandwidth sharing objective to be realized by appropriate packet level mechanisms. Alternatively, it may be considered as a good candidate to approximate the behaviour of elastic traffic in the "best effort" Internet. Simulations show indeed that the performance of balanced fairness is generally close to that of other "fair"

allocations like max-min fairness and proportional fairness [2].

While the performance of balanced fairness is insensitive to detailed traffic characteristics, it is still a complex function of the capacity of all links and the traffic intensity on all routes. An efficient recursive algorithm to compute performance metrics for a certain class of topologies was presented in [6]. The algorithm has been implemented in Mathematica and readily provides numerical and symbolic evaluations. We review and extend this algorithm in this paper. We focus on the practically interesting case of tree networks that may represent the successive downlink multiplexing stages of an access network where bandwidth is a scarce resource. The additional per-flow rate constraints due to the user's access line for instance are also taken into account. As illustrated below, these results can be used to analyse the accuracy of crude, simple approximations that are likely to be used in practice.

The rest of this paper is organized as follows. Section 2 gives a brief review of the notion of balanced fairness and its basic properties. The next section is devoted to the notion of Pareto efficiency which is a crucial property on which the method relies. The recursive algorithm is presented in Section 4. In Section 5 several examples of the recursion are provided and numerical evaluations illustrate the accuracy of simple approximations. The results are summarized in Section 6.

2 Preliminaries

The following is a short summary of the notion of balanced fairness and the network model to which it pertains; for a full account readers are referred to [2, 5].

The network consists of a set of links $\mathcal{L} = \{1, \ldots, L\}$ where link l has a capacity C_l. A random number of flows compete for the bandwidth of these links. There are N classes of flows, $\mathcal{F} = \{1, \ldots, N\}$, where each class i is characterized by a route $\mathcal{R}_i$ consisting of a set of links. When link l is on route $\mathcal{R}_i$ we use the natural notation $l \in \mathcal{R}_i$. Conversely, defining $\mathcal{F}_l \subset \mathcal{F}$ to be the set of flow classes going through link l we can write equivalently $i \in \mathcal{F}_l$. The mean volume of information offered by flows in class i per unit time, i.e., the load of class i, is denoted ρ_i. The network state is defined by the vector $x = (x_1, \ldots, x_N)$, where x_i is the number of class-i flows in progress.

Resource allocation. The total capacity $\phi_i(x)$ allocated to class-i flows is assumed to be shared equally between these flows and to depend on the network state x only. The capacity allocation must satisfy the capacity constraints,

$$\sum_{i \in \mathcal{F}_l} \phi_i(x) \leq C_l, \qquad \forall\, l \in \mathcal{L}. \tag{1}$$

The allocation is said to be balanced if

$$\frac{\phi_i(x - e_j)}{\phi_i(x)} = \frac{\phi_j(x - e_i)}{\phi_j(x)}, \qquad \forall\, i, j,\ x_i > 0,\ x_j > 0,$$

where e_i is a N-vector with 1 in component i and 0 elsewhere. The balance property implies that there is a balance function $\Phi(x)$ such that

$$\phi_i(x) = \frac{\Phi(x - e_i)}{\Phi(x)}, \qquad \forall\, i,\ x_i > 0.$$

Basically any positive function $\Phi(x)$ defines a balanced allocation. Note that not all balanced allocations are feasible. However, as shown in [2] there is a unique balanced allocation such that for any network state x all the capacity constraints (1) are satisfied and at least one of them is satisfied as an equality, i.e., at least one network link is saturated. For this allocation, the balance function is obtained recursively from

$$\Phi(x) = \max_l \left\{ \frac{1}{C_l} \sum_{i \in \mathcal{F}_l} \Phi(x - e_i) \right\}, \tag{2}$$

$\Phi(0) = 1$ and $\Phi(x) = 0$ for all x such that $x_i < 0$ for some i. This allocation is referred to as balanced fairness. Any link l that realizes the maximum in (2) is saturated in state x.

This model can be extended to account for per-flow rate limits due to access lines for instance. Let a_i be the rate limit of a class-i flow. We have the additional constraints:

$$\phi_i(x) \le x_i a_i, \qquad \forall\, i,\ x_i > 0.$$

The balance function is then obtained recursively from

$$\Phi(x) = \max \left\{ \max_l \left\{ \frac{1}{C_l} \sum_{i \in \mathcal{F}_l} \Phi(x - e_i) \right\}, \max_{i\,:\,x_i > 0} \left\{ \frac{\Phi(x - e_i)}{x_i a_i} \right\} \right\}. \tag{3}$$

Unless otherwise specified, we assume that there is no per-flow rate limit.

Stationary distribution. Assuming Poisson flow arrivals and exponential flow size distributions, it may readily be verified from the balance property that an invariant measure is given by

$$\pi(x_1, \ldots, x_N) = \Phi(x_1, \ldots, x_N) \rho_1^{x_1} \cdots \rho_N^{x_N}. \tag{4}$$

This result, however, has a much wider validity. As shown in [2], the bandwidth sharing network can be identified with a so-called Whittle network of processor sharing servers (cf. [11]). The insensitivity properties of Whittle networks allow us to conclude that the invariant measure (4) is valid for much more general traffic characteristics. Flow sizes and think time durations can have quite general distributions and need not be independent. The number of flows per session can be generally distributed. The only requirement is that sessions arrive as a Poisson process [2, 5].

An important role is played by the normalization constant,

$$G = \sum_{x_1=0}^{\infty} \cdots \sum_{x_N=0}^{\infty} \Phi(x_1, \ldots, x_N) \rho_1^{x_1} \cdots \rho_N^{x_N},$$

which may be identified as the generating function of the balance function $\Phi(x)$ and thus contains the same information. In particular, $\Phi(x)$ itself and the performance measures can be derived from $G(\rho_1, \ldots, \rho_N)$. A key performance measure for class-i flows is the flow throughput γ_i, equal to the ratio of the mean flow size to the mean sojourn time. Applying Little's result we obtain:

$$\gamma_i = \frac{\rho_i}{\mathrm{E}\left[x_i\right]} = \frac{G}{\frac{\partial G}{\partial \rho_i}} = \frac{1}{\frac{\partial \log G}{\partial \rho_i}}. \tag{5}$$

In the rest of the paper we concentrate on an efficient method for calculating the normalization constant in tree networks.

3 Pareto efficiency in tree networks

We say that a network described in Section 2 is a tree network if there exists a graph without loop where each node represents a link and a set of paths in this graph where each path represents a route, all paths having a common extremity. We refer to the corresponding link as the root of the tree. A particular tree network and the corresponding graph are given in Figure 1. We first characterize Pareto efficient allocations in tree networks then prove that balanced fairness is Pareto efficient for this network topology.

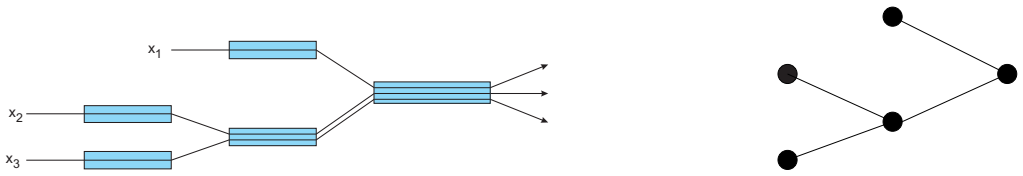


Figure 1: A tree network and the corresponding link graph.

Characterization of Pareto efficient allocations. An allocation is said Pareto efficient if for any class i and any state x such that $x_i > 0$, there is a saturated link on route $\mathcal{R}_i$. In the following, we say that a class i is active in state x if $x_i > 0$ and denote by $\mathcal{I}(x)$ the set of active classes in state x. For any $\mathcal{I} \subset \mathcal{F}$, we define the saturation set $\sigma(\mathcal{I}) \subset \mathcal{L}$ as the empty set if $\mathcal{I} = \emptyset$ and as follows otherwise:

First allocate to class-i flows, $i \in \mathcal{I}$, the capacity of their leaf link. Proceeding from the leaves towards the root, always apply the capacity constraint of the links to the aggregate flows, if any. The uppermost links that are constraining, and only those, belong to $\sigma(\mathcal{I})$.

The saturation set satisfies the following two properties:

(i) It defines a partition of $\mathcal{I}$ in the sense that for each class $i \in \mathcal{I}$ there is one and only one link $l \in \sigma(\mathcal{I})$ such that $i \in \mathcal{F}_l$.

(ii) No link ancestor of any link $l \in \sigma(\mathcal{I})$ can be saturated.

These two properties actually characterize the saturation set $\sigma(\mathcal{I})$.

Proposition 1 An allocation is Pareto efficient if and only if for any state $x \neq 0$ all links in the set $\sigma(\mathcal{I}(x))$ are saturated.

Proof. The condition is sufficient in view of property (i) above. Now consider a Pareto efficient allocation. For any given state x, there is a saturated link on the route $\mathcal{R}_i$ of each active class $i \in \mathcal{I}(x)$. Let l_i be the uppermost saturated link on route $\mathcal{R}_i$. The set $\{l_i,\ i \in \mathcal{I}(x)\}$ satisfies properties (i) and (ii) above and thus coincides with $\sigma(\mathcal{I}(x))$.

Pareto efficiency of balanced fairness. In [6], it was conjectured that balanced fairness is Pareto efficient in tree networks. This result is now established by Theorem 1 below.

Theorem 1 *Balanced fairness is Pareto efficient in tree networks.*

The proof of the theorem is given in the appendix. We have the following direct corollary.

Corollary 1 *Balanced fairness is Pareto efficient in tree networks with per-flow rate limits.*

Proof. For any given state x, a tree with class-i per-flow rate limit a_i is equivalent to a tree where each class i is divided into x_i subclasses, each connected to the tree by a link with capacity a_i.

4 Recursive algorithm in tree networks

In [6] we described a recursive algorithm for calculating the normalization constant for systems where for any state corresponding to a given set of active flow classes, $\mathcal{I}$, a given set of links, $\mathcal{S}$, is saturated. In view of the above results, trees satisfy this condition and the recursion is applicable. In fact, for trees the recursion can be simplified due to the fact that the same set of links can be saturated for many different sets of active flow classes. In the following we give the algorithm which entails successively evaluating the partial sums on those state subspaces where links $\mathcal{S}$ are saturated. The algorithm will be given separately for the cases without and with classwise rate limits.

Absence of flow rate limits. For any tree $\mathcal{T}$, with the set of links $\mathcal{L}$ and flows $\mathcal{F}$, we denote by $\Sigma = \{\mathcal{S} \subseteq \mathcal{L} : \exists \mathcal{I} \subseteq \mathcal{F} \text{ s.t. } \mathcal{S} = \sigma(\mathcal{I})\}$ the set of all feasible saturation sets, including the empty set. The algorithm consists of successively evaluating the partial sums on those state subspaces $\Omega_{\mathcal{S}}$ where links $\mathcal{S}$ are saturated:

$$G_{\mathcal{S}} = \sum_{x \in \Omega_{\mathcal{S}}} \pi(x), \quad \mathcal{S} \in \Sigma.$$

In view of Proposition 1 and Theorem 1,

$$\Omega_{\mathcal{S}} = \{x : \sigma(\mathcal{I}(x)) = \mathcal{S}\}.$$

The normalization constant is then simply given by:

$$G = \sum_{\mathcal{S}\in\Sigma} G_{\mathcal{S}}. \tag{6}$$

We define $\mathcal{T}_l$ as the subtree with root link l carrying the subset of flows $\mathcal{F}_l$. The set of all feasible saturation sets of subtree $\mathcal{T}_l$ is given by:

$$\Sigma_l = \{\mathcal{S} \subseteq \mathcal{L} : \exists \mathcal{I} \subseteq \mathcal{F}_l \text{ s.t. } \mathcal{S} = \sigma(\mathcal{I})\}.$$

Since $\Omega_\emptyset = \{0\}$ and $\pi(0) = \Phi(0) = 1$ we have $G_\emptyset[\mathcal{T}] = 1$. For any $\mathcal{S} \neq \emptyset$, the partial state sum can be reduced into a product of separate partial sums, each corresponding to a subtree with saturated root,

$$G_{\mathcal{S}} = \prod_{l\in\mathcal{S}} G_l.$$

For the G_l we deduce a recursion in the same way as in [6],

$$G_l = \frac{\sum_{\mathcal{S}\in\Sigma_l\setminus\{l\}} G_{\mathcal{S}} \sum_{i\in\mathcal{F}_l\setminus\mathcal{F}_{\mathcal{S}}} \rho_i B_i(l,\mathcal{S})}{C_l - \sum_{i\in\mathcal{F}_l} \rho_i}, \quad \forall l \in \mathcal{L}[\mathcal{T}],$$

where $\mathcal{F}_{\mathcal{S}} \equiv \cup_{l\in\mathcal{S}}\mathcal{F}_l$ denotes the set of all classes going through the saturation set $\mathcal{S}$ and $B_i(l,\mathcal{S})$ is equal to 1 if $\sigma(\mathcal{F}_{\mathcal{S}} \cup \{i\}) = \{l\}$ and 0 otherwise.

Presence of flow rate limits. For any tree $\mathcal{T}$ the set of all feasible saturation sets is the same as without rate limits and, as before, the normalization constant is decomposed into partial sums by (6). The main difference stems from the fact that now the set $\Omega_\emptyset$ may contain other states in addition to the state 0. We also have to be careful to state to which tree each state space or state sum relates to; we do this by indicating the tree in question in brackets. Sometimes we need to emphasize the link capacities of a tree $\mathcal{T}$ by writing $\mathcal{T}(C)$, where C is the capacity vector $C = \sum_{l\in\mathcal{L}} C_l e_l$ with e_l denoting a vector with one in the lth component and zero elsewhere.

The state space $\Omega_\emptyset[\mathcal{T}(C)]$ is equivalent to the state space of the corresponding loss system with inelastic bandwidth requirements a_i:

$$\Omega_\emptyset[\mathcal{T}(C)] = \{x : \forall l, \sum_{i\in\mathcal{F}_l} x_i a_i \leq C_l\}.$$

We denote by $G_\emptyset[\mathcal{T}(C)]$ the state sum over $\Omega_\emptyset[\mathcal{T}(C)]$:

$$G_\emptyset[\mathcal{T}(C)] = \sum_{x\in\Omega_\emptyset[\mathcal{T}(C)]} \pi(x) = \sum_{x\in\Omega_\emptyset[\mathcal{T}(C)]} \prod_{i\in\mathcal{F}} \frac{1}{x_i!}\left(\frac{\rho_i}{a_i}\right)^{x_i}, \tag{7}$$

where the invariant measure in $\Omega_\emptyset[\mathcal{T}(C)]$ has been made explicit, again using the convention $\pi(0) = 1$. The normalization constant (7) can be easily evaluated through the so-called convolution-truncation algorithm [10].

The reduction of the partial sum becomes, for any $S \neq \emptyset$,

$$G_{\mathcal{S}}[\mathcal{T}] = G_\emptyset[\mathcal{T} \setminus\setminus \underset{l\in\mathcal{S}}{} \mathcal{T}_l] \prod_{l\in\mathcal{S}} G_l[\mathcal{T}_l],$$

where $\mathcal{T}\setminus\setminus\mathcal{T}_l$ denotes the subtraction of subtree $\mathcal{T}_l$ from tree $\mathcal{T}$ and reduction by C_l of the link capacities on the route from l to the root of the remaining tree. $\mathcal{T} \setminus\setminus_{l\in\mathcal{S}} \mathcal{T}_l$ denotes the result of repeated application of the subtraction operation over all subtrees $\mathcal{T}_l$ with $l \in \mathcal{S}$ and reduction of the remaining tree by removal of all redundant links, i.e. links that do not belong to any saturation set.

The recursion for the $G_l[\mathcal{T}_l]$ now reads

$$G_l[\mathcal{T}_l] = \frac{\sum_{\mathcal{S}\in\Sigma_l\setminus\{l\}} G_{\mathcal{S}}[\mathcal{T}_l] \sum_{i\in\mathcal{F}_l\setminus\mathcal{F}_{\mathcal{S}}} \rho_i B_i[\mathcal{T}_l \setminus\setminus_{j\in\mathcal{S}} \mathcal{T}_j]}{C_l - \sum_{i\in\mathcal{F}_l} \rho_i}, \quad \forall l \in \mathcal{L},$$

where, generically for any tree $\mathcal{T}(C)$ with capacity vector C and root link denoted by 0, $B_i[\mathcal{T}(C)]$ stands for the blocking probability of class-i flows due to the root link 0 in the corresponding loss system, given by

$$B_i[\mathcal{T}(C)] = 1 - \frac{G_\emptyset[\mathcal{T}(C - a_i \sum_{l\in\mathcal{R}_i[\mathcal{T}]} e_l)]}{G_\emptyset[\mathcal{T}(C)]} - \sum_{l\in\mathcal{R}_i[\mathcal{T}], l\neq 0} \frac{G_\emptyset[\mathcal{T}(C) \setminus\setminus \mathcal{T}_l(C)]\, G_\emptyset[\mathcal{T}_l(C)]}{G_\emptyset[\mathcal{T}(C)]}\, B_i[\mathcal{T}_l(C)].$$

5 Application to specific tree networks

We now provide several examples of the recursion as well as numerical evaluations that illustrate the accuracy of simple approximations.

Tree with three branches. We first consider the simple case of a tree which has a common link, the root, with capacity C_0 and three branches with capacities C_1, C_2 and C_3 attached to it, see Figure 2. We assume that $C_i \leq C_0$ for all i and $C_1 + C_2 + C_3 > C_0$; otherwise some of the links would be redundant. Without loss of generality we can further assume that $C_1 \geq C_2 \geq C_3$.

To identify the feasible saturation sets, it is necessary to distinguish between different cases according to the ordering of C_0 with respect to the pairwise sums of branch capacities. Below we display the recursion equations for one of the four cases, viz. $C_1 + C_3 \geq C_0 \geq C_2 + C_3$, the analysis in the other cases being similar. The feasible saturation sets are then $\emptyset, \{1\}, \{2\}, \{3\}, \{2,3\}$ and $\{0\}$. We deduce:

$$G = G_\emptyset + G_1 + G_2 + G_3 + G_2G_3 + G_0$$

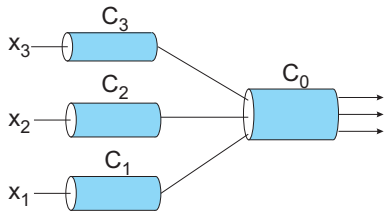


Figure 2: A tree with three branches.

with

$$G_\emptyset = 1; \quad G_i = \frac{\rho_i}{C_i - \rho_i},\ i = 1,2,3; \quad G_0 = \frac{G_2G_3\rho_1 + G_1(\rho_2+\rho_3) + (G_2+G_3)\rho_1}{C_0 - \rho_1 - \rho_2 - \rho_3}.$$

We get:

$$G(\rho) \;=\; \frac{1}{1 - \frac{\rho_1+\rho_2+\rho_3}{C_0}} \left(\frac{1 - \frac{\rho_1}{C_0}}{1 - \frac{\rho_1}{C_1}} + \frac{1 - \frac{\rho_2+\rho_3}{C_0}}{(1 - \frac{\rho_2}{C_2})(1 - \frac{\rho_3}{C_3})} - 1 \right) .$$

Note that the two-branch tree can be obtained as a special case by letting $\rho_3 = 0$.

Tree with n branches of same capacity. When the number of branches increases and the link capacities are general, the expressions become more complex. However, if we assume all the branches to have equal capacity, $C_i = C$ for all i, we can handle any number, n, of branches. Denote by m the largest integer such that $m \times C \le C_0$. A direct application of the recursive algorithm leads to

$$G = \sum_{k=0}^{m} \sum_{i_1,\ldots,i_k} \prod_{p=1}^{k} \frac{\rho_{i_p}}{C - \rho_{i_p}} + \sum_{i_1,\ldots,i_m} \frac{\sum_{i \neq i_1,\ldots i_m} \rho_i}{C_0 - \sum_i \rho_i} \prod_{p=1}^{m} \frac{\rho_{i_p}}{C - \rho_{i_p}}.$$

In case of homogeneous load, i.e., $\rho_i = \rho$ for all i, we get

$$G = \sum_{k=0}^{m} \binom{n}{k} \left(\frac{\rho}{C-\rho} \right)^k + \frac{(n-m)\rho}{C_0 - n\rho} \binom{n}{m} \left(\frac{\rho}{C-\rho} \right)^m .$$

Figure 3 depicts the throughput, measured in C, obtained with (5) for a system with $n = 20$ branches and C_0 equalling $m = 1, \ldots, n$ times C. The horizontal axis represents the scaled total load of the root $n\rho/C_0$. Note that the cases $m = 1$ and $m = n$ are identical, both representing effectively a single link system. In the former case, the branch links are redundant. In the latter case, the root link is redundant and the system is equivalent to n independent single link systems.

Tree with four multiplexing levels. As an example of a larger network consider the 4-level tree of Figure 4 consisting of 10 links with the shown capacities and 9 flow classes numbered by their access links. In particular, we study the throughput of class 10 going

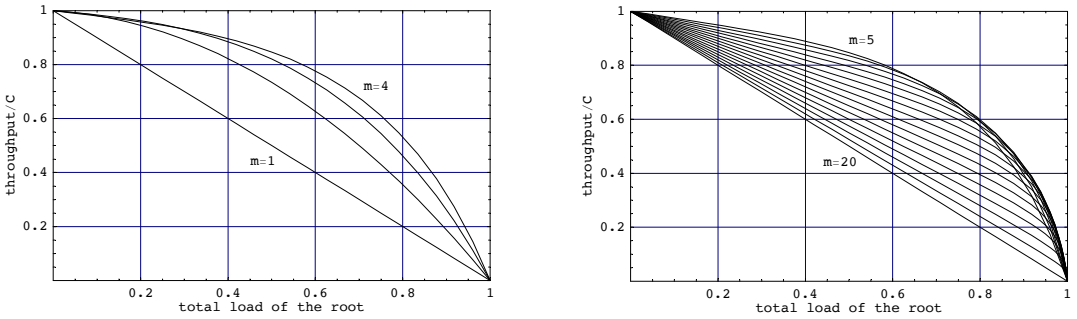


Figure 3: Throughput in a tree with a root and 20 identical branches. Capacity of the root is $m = 1, \ldots, 4$ (left) or $m = 5, \ldots, 20$ (right) times that of a branch.

through links 10, 7, 3, and 1 as a function of its own load ρ_{10}. The other classes are assumed to have fixed loads as follows: $\rho_1 = \rho_2 = \rho_4 = \rho_6 = \rho_7 = 2$ and $\rho_5 = \rho_8 = \rho_9 = 1$. With these loads, all four links on route 10 have the average residual capacity of 3 units.

As before, one can calculate the normalization constant,

$$G(\rho_{10}) = \frac{6\,(5-\rho_{10})\,(951-411\,\rho_{10}+46\,{\rho_{10}}^2)}{(3-\rho_{10})^4},$$

from which one obtains the exact throughput γ_{10} by derivation (5).

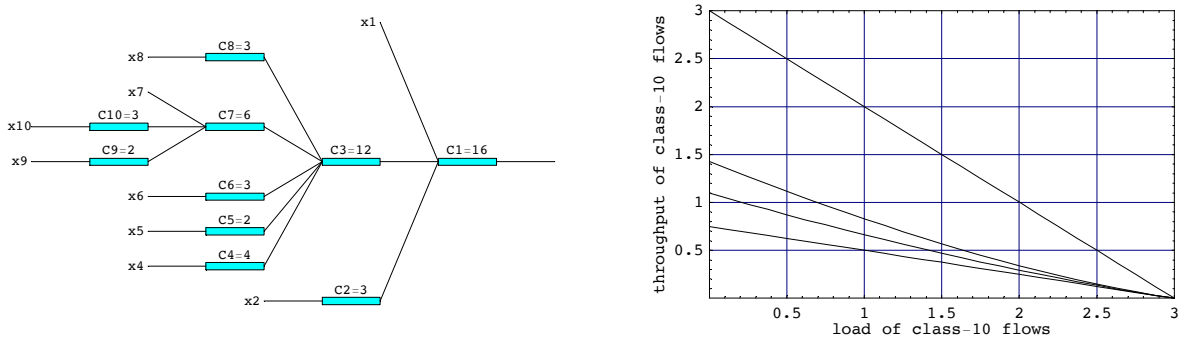


Figure 4: An example of a 4-level tree and comparison of class-10 throughput lower and upper bounds with the exact result. From bottom up: store-and-forward (proven lb), parking lot (conjectured lb), exact, deterministic (proven ub).

The throughput is compared with various approximations in Figure 4. The so-called store and forward network[1] was shown in [5] to constitute a lower bound for the throughput for any balanced fair network. Another approximation is obtained by neglecting all the capacity constraints outside the considered route. This results in a so-called parking lot network for which there is an explicit expression for the flow throughput [6] and which is likely to constitute another (tighter) lower bound. The upper bound comprises of the so-called deterministic approximation derived on constraining all the cross traffic classes by

[1]Flows are routed step by step with equal bandwidth sharing on each link.

links whose capacity equals their offered load [5]. In this stability limit, the cross traffic classes become deterministic and their load can be subtracted from the link capacities on the main route. Throughput is then determined by the bottleneck link; in this example, all four links are bottlenecks with the residual capacity of 3 units. Clearly, the bounds are tighter for any route where residual capacities are less homogeneous than in the present example.

Tree with per-flow rate limits. The example comprises a 2-branch tree with each branch carrying flows of two classes with different access rate limitations, shown by the diagram on the left of Figure 5. The link capacities of the branches are $C_1 = 7$ and $C_2 = 10$, and the common link has the capacity $C_0 = 15$. The access rates of the flows in branch 1 are $a_{1,1} = 1$ and $a_{1,2} = 3$, while for branch 2 the access rates are $a_{2,1} = 2$ and $a_{2,2} = 4$. In the graph of Figure 5 the throughput of each class is given as a function of its own load, with all the other loads being equal to 1. For low loads, the throughput equals the access rate, whereas the throughput goes to zero when the total load of a branch approaches its capacity.

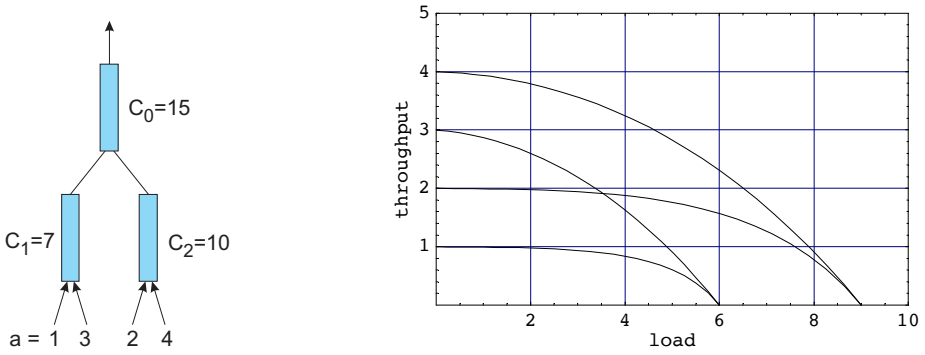


Figure 5: Throughputs of flows in a 2-branch tree with two access rate classes loading each branch.

In Figure 6, we compare the exact throughput of flows in a class with the same conjectured upper and lower bounds as in Figure 4. The system is the same as in the previous example, and the considered class is that with access rate limit $a_{1,2} = 3$ in branch 1. The throughput of flows in this class, as well as the corresponding throughput bounds, are plotted as a function of its own load, when the loads of the other classes equal 1 (the exact throughput curve is identical to the second uppermost curve in Figure 5).

6 Summary

Balanced fairness is a new notion of bandwidth allocation with the very gratifying property that flow level performance metrics are insensitive to detailed traffic characteristics. This is particularly important for data network engineering since performance can be predicted

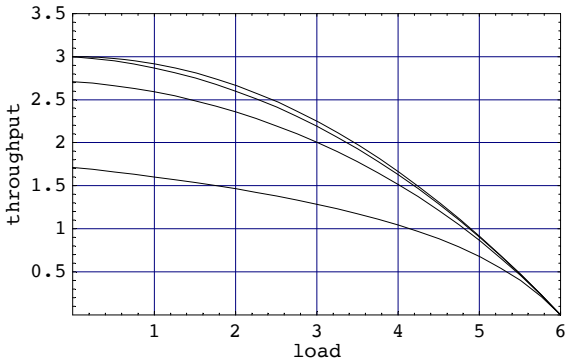


Figure 6: Comparison of conjectured upper and lower bounds with the exact throughput. From bottom up: store-and-forward, parking lot, exact, deterministic.

from an estimate of overall traffic volume alone and is independent of changes in the mix of user applications.

The balanced fair allocation for any network is uniquely determined by the basic recursion (2). For larger networks, however, straightforward application of the recursion is hampered by the usual state space explosion problem. Our main contribution is the derivation of a recursive algorithm for directly calculating the normalization constant and flow throughputs in the practically interesting case of tree networks, with and without flow rate limits. The comparison of exact results with a number of bounds is the prelude to a more thorough future evaluation of approximations useful for practical engineering purposes.

Acknowledgments

Part of this work was done during J. Virtamo's visit to France Telecom R&D. He wishes to thank J. Roberts and France Telecom R&D for their kind hospitality. The work was also financially supported by the Academy of Finland.

Appendix. Proof of Theorem 1

In view of Proposition 1, it is sufficient to prove that in any state $x \neq 0$, all links in the set $\sigma(\mathcal{I}(x))$ are saturated under balanced fairness. We prove the theorem by induction on $|x| \equiv \sum_{i=1}^{N} x_i$. The assertion is certainly true for $|x| = 1$. Now let x be such that $|x| \geq 2$ and assume the assertion is true for any state $x' \neq 0$ such that $|x'| < |x|$. We denote by $\mathcal{I} \equiv \mathcal{I}(x)$ the set of active classes in state x.

By properties (i) and (ii) of Section 3, it is easy to see that if there are several links $l \in \sigma(\mathcal{I})$, the assertion is decomposed into separate claims for each subtree $\mathcal{T}_l$ and is true by the induction assumption. Thus, we can assume that $\sigma(\mathcal{I}) = \{0\}$, where link 0 denotes

the root of the tree, and our task is to show that the root is indeed saturated.

The set of flow classes $\mathcal{I}$ can be divided into two disjoint sets $\mathcal{I}'$ and $\mathcal{I}''$, non-critical and critical, respectively, with

$$\mathcal{I}' = \{i \in \mathcal{I} : x_i > 1 \text{ or } \sigma(\mathcal{I} \setminus \{i\}) = \{0\}\}, \quad \mathcal{I}'' = \{i \in \mathcal{I} : x_i = 1 \text{ and } \sigma(\mathcal{I} \setminus \{i\}) \neq \{0\}\}.$$

We have for any link l,

$$\begin{aligned}
\frac{1}{C_0}\sum_i \Phi(x-e_i) &= \frac{1}{C_0}\Big(\sum_{i\in\mathcal{F}_l}\Phi(x-e_i) + \sum_{j\notin\mathcal{F}_l}\Phi(x-e_j)\Big) \\
&\geq \frac{1}{C_0 C_l}\Big(C_l\sum_{i\in\mathcal{F}_l}\Phi(x-e_i) + \sum_{i\in\mathcal{F}_l, j\notin\mathcal{F}_l}\Phi(x-e_i-e_j)\Big) \qquad (8)\\
= \frac{1}{C_0 C_l}\Big(C_l\sum_{i\in\mathcal{I}'\cap\mathcal{F}_l}\Phi(x-e_i) &+ \sum_{i\in\mathcal{I}'\cap\mathcal{F}_l, j\notin\mathcal{F}_l}\Phi(x-e_i-e_j) + C_l\sum_{i\in\mathcal{I}''\cap\mathcal{F}_l}\Phi(x-e_i) + \sum_{i\in\mathcal{I}''\cap\mathcal{F}_l, j\notin\mathcal{F}_l}\Phi(x-e_i-e_j)\Big).
\end{aligned}$$

The first two terms are developed as follows:

$$\begin{aligned}
C_l\sum_{i\in\mathcal{I}'\cap\mathcal{F}_l}\Phi(x-e_i) + \sum_{i\in\mathcal{I}'\cap\mathcal{F}_l, j\notin\mathcal{F}_l}\Phi(x-e_i-e_j) &\geq \sum_{i\in\mathcal{I}'\cap\mathcal{F}_l, j\in\mathcal{F}_l}\Phi(x-e_i-e_j) + \sum_{i\in\mathcal{I}'\cap\mathcal{F}_l, j\notin\mathcal{F}_l}\Phi(x-e_i-e_j) \\
&= \sum_{i\in\mathcal{I}'\cap\mathcal{F}_l, j\in\mathcal{I}}\Phi(x-e_i-e_j) = C_0\sum_{i\in\mathcal{I}'\cap\mathcal{F}_l}\Phi(x-e_i),
\end{aligned}$$

where the last equality is due to the fact that link 0 is saturated in state $x - e_i$ if $i \in \mathcal{I}'$. For the last term of (8) we have,

$$\sum_{i\in\mathcal{I}''\cap\mathcal{F}_l, j\notin\mathcal{F}_l}\Phi(x-e_i-e_j) = \sum_{i\in\mathcal{I}''\cap\mathcal{F}_l}\Big(\sum_{l'\in\sigma(\mathcal{I}\setminus\mathcal{F}_l)}C_{l'}\Big)\Phi(x-e_i) \geq (C_0 - C_l)\sum_{i\in\mathcal{I}''\cap\mathcal{F}_l}\Phi(x-e_i).$$

The first step can be reasoned as follows: For any $i \in \mathcal{I}''$, the saturation set $\sigma(\mathcal{I} \setminus \{i\})$ is different from $\{0\}$ by definition. None of the links on the route $\mathcal{R}_i$ can belong to $\sigma(\mathcal{I} \setminus \{i\})$; otherwise, we would have $\sigma(\mathcal{I} \setminus \{i\}) = \sigma(\mathcal{I})$ so that $\sigma(\mathcal{I} \setminus \{i\}) = \{0\}$. If in addition $i \in \mathcal{F}_l$, we deduce that neither link l nor any ancestor of l can belong to $\sigma(\mathcal{I} \setminus \{i\})$ so that $\sigma(\mathcal{I}\setminus\{i\}) = \sigma(\mathcal{F}_l \cap \mathcal{I}\setminus\{i\}) \cup \sigma(\mathcal{I}\setminus\mathcal{F}_l)$. The first step is obtained by applying the partition property (i) of the saturation set $\sigma(\mathcal{I}\setminus\mathcal{F}_l)$:

$$\sum_{j\notin\mathcal{F}_l}\Phi(x-e_i-e_j) = \sum_{l'\in\sigma(\mathcal{I}\setminus\mathcal{F}_l)}\sum_{j\in\mathcal{F}_{l'}}\Phi(x-e_i-e_j) = \sum_{l'\in\sigma(\mathcal{I}\setminus\mathcal{F}_l)}C_{l'}\Phi(x-e_i).$$

The second step, the inequality, follows from the fact that $\sigma(\mathcal{I}) = \{0\}$ which implies:

$$C_0 \leq \sum_{l'\in\sigma(\mathcal{F}_l)}C_{l'} + \sum_{l'\in\sigma(\mathcal{I}\setminus\mathcal{F}_l)}C_{l'} = C_l + \sum_{l'\in\sigma(\mathcal{I}\setminus\mathcal{F}_l)}C_{l'}.$$

Substituting these inequalities in (8) we finally obtain,

$$\frac{1}{C_0}\sum_i \Phi(x-e_i) \;\;\geq\;\; \frac{1}{C_l}\sum_{i\in\mathcal{F}_l}\Phi(x-e_i).$$

Thus, link 0 realizes the maximum of

$$\Phi(x) = \max_l \left(\frac{1}{C_l}\sum_{i\in\mathcal{F}_l}\Phi(x-e_i)\right)$$

and is saturated in state x completing the proof.

References

[1] D. Bertsekas and R. Gallager, Data Networks (2nd ed.), Prentice Hall, Englewood Cliffs, 1992.

[2] T. Bonald, A. Proutière, Insensitive bandwidth sharing in data networks, Queueing Systems 44 (2003) 69-100.

[3] T. Bonald, A. Proutière, Insensitivity in processor-sharing networks, Performance Evaluation 49 (2002) 193-209.

[4] T. Bonald, A. Proutière, G. Régnié, J. Roberts, Insensitivity results for statistical bandwidth sharing, in: Proceedings of ITC 17, Elsevier, 2001, 125-136.

[5] T. Bonald, A. Proutière, On performance bounds for balanced fairness, Performance Evaluation 55 (2004) 25-50.

[6] T. Bonald, A. Proutière, J. Roberts and J. Virtamo, Computational aspects of balanced fairness, in: Proceedings of the 18th International Teletraffic Congress ITC-18, Berlin, Germany, 31 August - 5 September, 2003, 801-810.

[7] F.P. Kelly, A. Maulloo and D. Tan, Rate control in communication networks: shadow prices, proportional fairness and stability, Journal of the Operational Research Society 49 (1998) 237-252.

[8] J. Mo and J. Walrand, Fair end-to-end window-based congestion control, IEEE/ACM Transactions on Networking 8 (2000) 556-567.

[9] J. Roberts, U. Mocci, and J. Virtamo (eds.), Broadband Network Teletraffic, Springer-Verlag, Berlin, 1996.

[10] K.W. Ross, Multiservice Loss Models for Broadband Telecommunication Networks, Springer-Verlag, Berlin, 1995.

[11] R. Serfozo, Introduction to Stochastic Networks, Springer-Verlag, Berlin, 1999.