# Computer Networks
# Fairness

Saad Mneimneh
Computer Science
Hunter College of CUNY
New York

*Life is not fair, but we can at least theorize*

## 1 Introduction

So far, we analyzed a number of systems in terms of average occupancy, throughput, and delay. However, we did not make any distinction among the customers arriving to the system. For instance, consider the M/M/1 system with a finite queue of packets. If multiple flows are sharing the link, it is possible to achieve high throughput by making one flow generate enough packets, and preventing other flows from sending (their packets will be dropped). Therefore, we need another property to ensure that all flows receive an equal share of the resources. Although not very well defined at this point, we will call this property *fairness*. To illustrate the basic idea, consider the following figure:

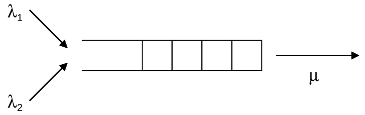

Figure 1: Fairness with two flows

In the above figure, efficiency means that $\lambda_1 + \lambda_2$ is close to $\mu$. However, fairness means that $\lambda_1$ is equal to $\lambda_2$. We previously showed that a good operating point is when $\lambda_1 + \lambda_2 \approx \mu/2$ (at 50% utlization, throughput/delay is high). We also argued that this behavior of M/M/1 is simiar to that of a network in general (because each link can be modeled as an M/M/1 system). But a network in general is not just one link; therefore, for fairness, the notion of "equal share" in not necessarily what we think of equality, in particular, how do we handle flows that use different paths?

## 2 A motivating example

Assume there is no explicit demand or reservation of bandwidth, and that all the links have a capacity of 1 (unit of bandwidth in bps). How should we assign rates to the flows in the following figure?

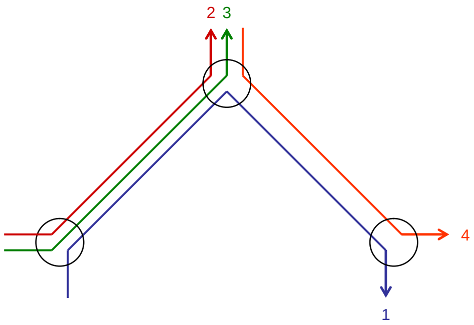

Figure 2: An example of fairness

If all flows receive an equal share of the resources, we would assign each flow a rate of 1/3. However, while it makes sense to limit the rates of flows 1, 2, and 3 to 1/3, it is pointless to do the same for flow 4. Flow 4 may receive a rate of 2/3 because no other flow will benefit by reducing the rate of flow 4 below 2/3. On the other hand, increasing the rate of flow 4 beyond 2/3 will reduce the rate of at least one other flow among those which receive a rate of 1/3. So the assignment of rates $(1/3, 1/3, 1/3, 1/4)$ is fair in some sense. This shows that fair does not necessarily mean "equal". So how should we define fairness?

## 3 Max-Min fairness

The observation made above about flow 4 is a key in defining the notion of fairness. In particular, for the fair assignment made above, increasing the rate of flow 4 results in a decrease for some other flow with smaller rate. As illustrated in the figure below, this means moving away from being fair.

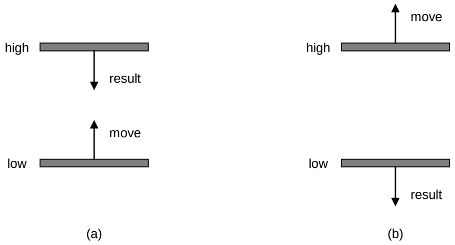

Figure 3: Moving towards (a) or away from (b) fairness

A fair assignment is such that any increase in rate results in a decrease of a smaller rate. In other words, any increase in rate will move us away from fairness (Figure 3(b)). One possible way of achieving such an assignment is by maximizing the minimum rate. This strategy can be justified by the following argument: If the minimum rate is not maximized, then one can increase that rate while decreasing only larger rates, thus the assignment is not fair. But this maximization by itself is not enough to ensure the fairness property. For instance, after maximizing the minimum rate, two flows with larger rates should obtain the same rate if the rate of one can be trated for the rate of the other. Therefore, we need to solve a hierarchy of nested problems. First we must obtain an assignment that maximizes the minimum rate. Among all such assignments, we must then find the one that maximizes the second smallest, and so on. This is called the Max-Min fairness and we will show later that is satisfies our fairness property:

**Fairness property**: Given an assignment of rates, for every flow $f$ with rate $r_f$, increasing $r_f$ results in a decrease of $r_{f'}$ for some flow $f'$ such that $r_{f'} \leq r_f$.

Let $l$ denote a link, and definte:

$$F_l = \sum_{f \text{ crosses } l} r_f$$

then we need to satisfy the following constraints:

$$r_f \geq 0 \quad \forall \ f$$
$$F_l \leq c_l \quad \forall \ l$$

where $c_l$ is the capacity of link $l$.

Our algorithm Max-Min fairness algorithm will work as follows:

- Increase the rates of all flows simultaneously by the same amount until one or more link saturate ($F_l = c_l$)
- Freeze all flow passing through the saturated link(s)
- Repeat with the process with remaining (active) set of flows

Of course we need a practical way of increasing all rates simultaneoulsy until a link saturates. To do this, we find the smallest $\epsilon$ such that when $r_f$ is increased by $\epsilon$ for all active flows $f$, a link will saturate. This can be easily computed as follows:

$$\min_l \frac{c_l - F_l}{n_l}$$

where $n_l$ is the number of flows crossing link $l$.

Here's the algorithm:

Initially, $r_f = 0$ for all flows, $F_l = 0$ for all links, $F = \{\text{all flows}\}$, and $L = \{\text{all links}\}$.

```
**repeat**
    $n_l \leftarrow$ number of flows $f \in F$ crossing link $l$
    $\epsilon \leftarrow \min_{l \in L}(c_l - F_l)/n_l$
    **if** $f \in F$
        **then** $r_f \leftarrow r_f + \epsilon$
    $F_l \leftarrow \sum_{f \text{ crossing } l} r_f$
    $L \leftarrow \{l | F_l < c_l\}$
    $F \leftarrow F - \{f | f \text{ crosses a link } \notin L\}$
**until** $F = \emptyset$
```

## 4 Bottlnecks and the fairness property

Define a link $l$ to be a bottleneck for flow $f$ iff:

- $f$ crosses $l$
- $c_l = F_l$
- all flows $f'$ crossing $l$ satisfy $r'_f \leq r_f$

Note that a direct result of this definition is that if $f$ and $f'$ have a common bottleneck, then $r_f = r_{f'}$. Moreover, this definition captures our intuitive notion of a bottleneck. To see this, consider the following example (assume all links have unit capacity):

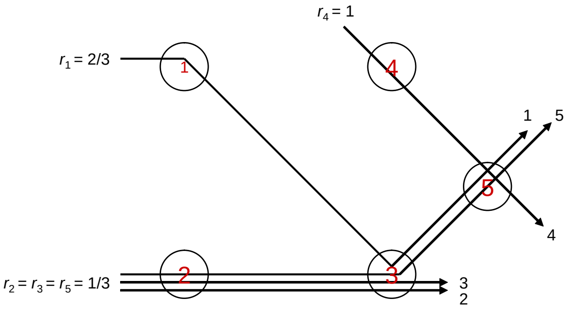

Figure 4: Bottleneck example

| flow | bottleneck |
|---|---|
| 1 | (3,5) |
| 2 | (2,3) |
| 3 | (2,3) |
| 4 | (4,5) |
| 5 | (2,3) |

From the above example, we can see that, for a given flow, increasing the capacity of links other than its bottleneck will not help increasing the rate of that flow. It is easy to see that the Max-Min fairness algorithm of Section 3, when it terminates, guarantees that every flow has a bottleneck. This is because every flow crosses a link that eventually saturates, and when a link saturates, all flows crossing that link have the largest rate. Furthermore, we can make the following equivalence, which proves that the Max-Min algorithm satisfies the fairness property.

$$\text{(1) every flow has a bottlenech} \Leftrightarrow \text{(2) fairness property}$$

(1) ⇒ (2): Assume we increase $r_f$ for some flow $f$. Since $f$ has a bottleneck $l$, $F_l = c_l$ ($l$ is saturated) and all flows $f'$ going through $l$ satisfy $r_{f'} \leq r_f$. Therefore, some some rate $r_{f'} \leq r_f$ must decrease.

(2) ⇒ (1): Assume some flow $f$ does not have a bottleneck. Therefore, for every link $l$ that $f$ crosses, either $F_l < c_l$ or there exists a flow $f'$ crossing $l$ with $r_{f'} > r_f$ and $F_l = c_l$. As a result, we can increase $r_f$ by only decreasing rates for flows $f'$ such that $r_{f'} > r_f$, a contradiction.

# 5 Generalizations

It is possible to generalize the Max-Min fairness algorithm to handle priorities. For instance, assume that each flow $f$ is associated with a priority index $p_f$, which is an integer. Then, conceptually, each flow $f$ can be considered as a collection of $p_f$ flows of equal priority. We are back to the previous setting, and the Max-Min fairness algorithm will then change as follows (the requirement that $p_f$ is an integer can be relaxed):

$$\epsilon \leftarrow \min_{l \in L} \frac{c_l - F_l}{\sum_{f \text{ crosses } l} p_f}$$

$$r_f \leftarrow r_f + p_f \epsilon$$

As a result, the fairness property will be the following:

**Fairness property**: Given an assignment of rates, for every flow $f$ with rate $r_f$, increasing $r_f$ results in a decrease of $r_{f'}$ for some flow $f'$ such that $r_{f'}/p_{f'} \leq r_f/p_f$.

Another variation for the Max-Min fairness algorithm is to require that for every flow $f$ and every link $l$, $r_f \leq (F_l - c_l)/q_l$, for some given $q_l$. This essentially means that we do not saturate the link. Therefore, we leave some unutilized capacity to account for possible fluctuation in the rates. This can be achieved by adding (a ficticious) flow $f_l$ on link $l$ with priority $q_l$ (while all flows have priority 1) to account for the unused capacity. Note that $f_l$ will be the last flow to remain active until link $l$ saturates (since link $l$ is the only link that $f_l$ crosses); therefore, every flow crossing $l$ will satisfy $r_f \leq (F_l - c_l)/q_l$. In this case, the total rate crossing link $l$ will be

$$\frac{n_l c_l}{n_l + q_l}$$

It is possible also to combine both variations, i.e. priorities and unused capacity. In this case, we add a ficticious flow on link $l$ with priority equal to $p_l q_l$, where $p_l = \max_{f \text{ crosses } l} p_f$. The total rate crossing link $l$ will be

$$\frac{\sum_{f \text{ crosses } l} p_f c_l}{\sum_{f \text{ crosses } l} p_f + p_l q_l}$$

## 6 Fairness index

From the Max-Min fairness algorithn, we expect $n$ flows sharing a common bottleneck to receive the same rates. But what if they don't in reality? How do we measure fairness? A famous fairness index is the following:

$$F(r) = \frac{(\sum_i r_i)^2}{n \sum_i r_i^2}$$

This index has nice properties:

- $0 < F(r) \leq 1$:
  - totally fair: all $r_i$'s are equal: $F(r) = 1$
  - totally unfair: only one user is given the resource: $F(r) = 1/n$ (which goes to zero when $n \to \infty$)
- independent of scale: the unit of measurment is irrelevant, i.e. multiplying all rates by the same constant keeps the index unchanged
- sensitive: any slight change in allocation shows up in the index
- fractional: if only $k$ users share the resource equally, $F(r) = k/n$

## References

Dimitri Bertsekas and Robert Gallager, Data Networks