

Lower Space Bounds for Approximate Fairness

Abhimanyu Das, Debojyoti Dutta, Ashish Goel, John Heidemann, Ahmed Helmy

USC/Mahi Networks, USC/ISI, Stanford, USC/ISI, USC

abhimand@usc.edu, ddutta@usc.edu, ashishg@stanford.edu, johnh@isi.edu, helmy@usc.edu

Approximate max-min fair bandwidth allocation has been a very well studied problem. A low state scheme for the above problem will have interesting consequences for router design. In this paper we show that in the general case, if there are n flows, routers need to maintain $\Omega(n)$ state in order to give bounded fairness guarantees with low error. Our proof is independent of any assumptions about the internal algorithmic details. We are unaware of previous work in this direction.

An algorithm for bandwidth allocation among n flows is defined to be ϵ -fair if the bandwidth allocated to flow i , μ_i , is related to its demand λ_i , the error fraction ϵ , and its max-min fair rate f by: $(1 - \epsilon) \min(\lambda_i, f) \leq \mu_i \leq (1 + \epsilon) \min(\lambda_i, f)$. Unless stated otherwise, we use ϵ -fairness and approximate max-min fairness interchangeably. Thus, we have:

THEOREM 1. *Any algorithm that imposes ϵ -fairness among packets of n flows in a given window size W (ie provides fairness within every set of W consecutive packets) to within any constant relative error $\epsilon < \frac{1}{8}$ takes $\Omega(n)$ space.*

PROOF. Consider a sliding window of size W , and an algorithm, A , that provides ϵ -fairness within this sliding window. We provide a proof by contradiction. Assume A requires $g(n)$ state, where $g(n)$ is not $\Omega(n)$. Then we construct an encoder/decoder combination to transmit an n -bit string with an equal number of zeros and one's, using $g(n)$ bits. This violates information theory bounds for lossless coding.

Encoder Construction: If the i^{th} bit of the n -bit string is 0, we create a flow with rate r pkts/s (we call it a good flow), and if the i^{th} bit is 1, we create a flow with rate kr pkts/s (such a flow will be called a bad flow). Our construction uses constant sized packets. We choose k to be much greater than 2. The total sending rate of all the flows is $R = \frac{nr(k+1)}{2}$ pkts/s. The window W therefore corresponds to $\Delta = \frac{W}{R}$ seconds of the aggregate flow rate R . Let $x = r\Delta$. Thus, every window of W packets contains exactly $x = r\Delta$ packets of each of the $n/2$ good flows and $kx = kr\Delta$ packets of each of the $n/2$ bad flows. Set the capacity of the black box to be $C = 2nr$ pkts/sec. Thus, among the window W of packets, only $C\Delta = 2nx$ packets are accepted, and the rest are dropped. The max-min fair rate f of the router therefore corresponds to $3x$ accepted packets per flow within the window W . Now let us insert a total of W packets from these flows into A in a weighted round robin fashion, with weights 1 and k corresponding to good and bad flows respectively. The state of the algorithm, is sent as the encoded state. Thus the size of the code sent by the encoder is $g(n)$, which is not $\Omega(n)$ by assumption.

Decoder Construction: We make n copies of the algorithm A 's state (received from the encoder) for decoding the n bits. For the i^{th} copy, we decode bit i as follows. We send x new packets labeled i through A . Then we observe how many of these packets are accepted, and decide the value of the bit accordingly.

Consider the last W packets seen at the encoder (before the x packets are inserted at the decoder, as above). Let us partition these W packets into two sets, M and P , such that $|M| = x$ packets and $|P| = W - x$ packets. Also, let the set of

x new packets sent at the decoder be denoted by Q . By our assumption, the algorithm A provides ϵ -fairness within every window of W consecutive packets seen. In particular, this is true for each of the two windows formed by the consecutive sets MP and PQ , where $MP = M \cup P$ and $PQ = P \cup Q$. Now, we consider two cases.

Flow i is bad: We will find out the maximum number of accepted packets of flow i in Q denoted by $\max Q_i$. Consider the window MP . Now, the ideal fair share of this flow in the window MP is given by $FS_{MP}^{\text{bad}} = \frac{2nx - \frac{n}{2}x}{\frac{n}{2}} = 3x$. Also, the maximum possible number of packets of i accepted in M can be obtained by assuming that all packets of flow i in M are accepted; this is given by $\max M_i^{\text{bad}} = \binom{k}{k+1} \left(\frac{x}{2}\right) = \frac{2kx}{(k+1)n}$. Therefore the minimum possible number of accepted packets of i in P , $\min P_i^{\text{bad}}$ is given by $\min P_i^{\text{bad}} = FS_{MP}^{\text{bad}}(1 - \epsilon) - \max M_i^{\text{bad}} = 3x(1 - \epsilon) - \frac{2kx}{(k+1)n}$. Now consider the window PQ . The ideal fair share of this flow in the window PQ is given by $FS_{PQ}^{\text{bad}} = \frac{2nx - \frac{W-x}{k+1}}{\frac{n}{2}} = 3x + \frac{2x}{(k+1)n}$. Therefore, $\max Q_i$ is given by the following equation: $\max Q_i^{\text{bad}} = FS_{PQ}^{\text{bad}}(1 + \epsilon) - \min P_i^{\text{bad}} = 6x\epsilon + \frac{2x\epsilon}{(k+1)n} + \frac{2x}{n}$.

Flow i is good: In this case, we will find out the minimum possible number of accepted packets of flow i in Q denoted by $\min Q_i^{\text{good}}$. Consider the window PQ . The maximum possible number of accepted packets of i in P can be obtained by assuming that all packets of flow i in P are accepted; this is given by $\max P_i^{\text{good}} = \frac{W-x}{(k+1)\frac{n}{2}}$. The ideal fair share of this flow in the window PQ is given by $FS_{PQ}^{\text{good}} = x + \frac{W-x}{(k+1)\frac{n}{2}}$. Therefore, $\min Q_i^{\text{good}}$ is given by the following equation: $\min Q_i^{\text{good}} = FS_{PQ}^{\text{good}}(1 - \epsilon) - \max P_i^{\text{good}} = x(1 - \epsilon) - \frac{(W-x)\epsilon}{(k+1)\frac{n}{2}} = x - 2x\epsilon + \frac{2x\epsilon}{(k+1)n}$. Now, if $\min Q_i^{\text{good}} > \max Q_i^{\text{bad}}$, then we can clearly decode flow i as good or bad, based on the number packets that are accepted out of the x packets in Q . This simplifies to the following condition $x - 2x\epsilon + \frac{2x\epsilon}{n(k+1)} > 6x\epsilon + \frac{2x}{n} + \frac{2x\epsilon}{(k+1)n}$, or $1 - 2/n > 8\epsilon$. Thus, for large n , if $\epsilon < 1/8$, we can decode all the n bits without error. This violates the lower bounds in coding in the following way. The size of the minimum code is given by the entropy of the system. Now the entropy of the n -bit string is given by $\log C_{n/2}^n = \Omega(n)$. But by assumption, the code size is $g(n)$, which is not $\Omega(n)$. This is a contradiction. \square

By manipulating the parameters, it is possible to improve the bounds on ϵ .

One interesting direction would be to study the lower bounds under different models. Our on-going work also includes the study of the communication complexity of distributed schemes to impose approximate global max-min fairness. For more details, see

<http://netweb.usc.edu/ddutta/research/fairness>.