Competitive Router Scheduling with Structured Data*

Yishay Mansour[1] **, Boaz Patt-Shamir[2] ***, and Dror Rawitz[2]

[1] School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel.
mansour@cs.tau.ac.il
[2] School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel.
{boaz,rawitz}@eng.tau.ac.il

Abstract. We consider the task of transmitting structured information over bounded-capacity links. Our information model is a stream of basic units called *superpackets* that are broken into k packets each. To model the possible structure and redundancy of the superpackets, we assume that for each superpacket there is a collection of minimal subsets of packets whose delivery makes the superpacket *useful*. This very general model encompasses, for example, MPEG streams, where one can think of a group of pictures (GoP) as a superpacket. The fundamental difficulty is that networks can forward only the primitive packets, but applications can use only superpackets, and thus if no minimal subset is delivered, the whole superpacket becomes useless. Our aim is to maximize goodput (number of useful superpackets) in the face of overloaded communication links, where we are forced to drop some packets.

Specifically, we assume that an arbitrary stream of packets arrives at a router with multiple bounded-capacity outgoing links. An on-line algorithm needs to decide, for each superpacket, which outgoing link to use (all packets of the same superpacket must use the same link) and, in case of an overload at a link, which packets to drop and which to transmit so as to maximize goodput. We analyze a simple randomized competitive algorithm to the general case and provide a nearly matching lower bound on the competitive ratio of any randomized on-line algorithm.

1 Introduction

Consider a video stream encoded in MPEG-2 [1]. Grossly oversimplifying, the structure of the stream is as follows. The stream is broken into *Groups of Pictures* (GoP), which may last a few minutes each. A GoP consists of a single *I-frame*, a few *P-frames*, and many *B-frames*. An I-frame is a stand-alone picture that

* Research supported in part by the Next Generation Video Consortium, Israel.

** Supported in part by a grant from the Israel Science Foundation (grant No. 709/09) and grant No. 2008-321 from the United States-Israel Binational Science Foundation (BSF), and by Israel Ministry of Science and Technology.

*** Supported in part by the Israel Science Foundation (grant 1372/09) and by Israel Ministry of Science and Technology.

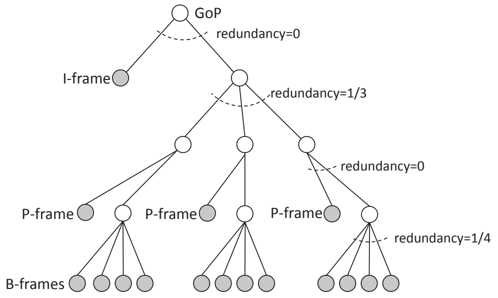

Fig. 1. *A tree representation of a GoP. Gray nodes represent data. A node with redundancy β is deemed useful if no more than a fraction β of its children is non-useful.*

requires no other information for decoding; decoding a P-frame requires its preceding I-frame; and decoding a B-frame requires its preceding "reference frame" (be it I- or P-frame).[3] The implication of this structure is that if an I-frame is lost, then the whole GoP is lost, and if a B-frame is lost, then only a fraction of a second is lost. But then again, if too many B-frames are lost (where "too many" is defined subjectively), the GoP should be considered again worthless. This structure can be modeled by a tree. Figure 1 illustrates a simple example.

The root represents the GoP; if either the I-frame (left child) or the other data (right subtree) are lost, then the GoP is lost; however, the right subtree may be considered useful even if one of its children is lost; and similarly, each of these (depth 2) nodes is useful only if both its P-frame child and at least $3/4$ of its B-frames are not lost. While this is not an accurate description of MPEG, we note that the hierarchical tree structure is very natural and appears in many other formats (e.g., XML documents [2]), with or without redundancy. Conceivably, more complex forms of redundancy are also used.

So suppose now that we need to manage a router that delivers multiple video streams, such that each stream may use any of a number of outgoing links (see Figure 2). At every step, some packets arrive at the router, and the router needs to decide which outgoing link is used for each packet, and, in case of an overflow in that link, which packets to discard. Note that in our example, if we drop an I-frame from each GoP, then all GoP's are useless at the receiving ends, even if the link has delivered all P- and B-frames (this is an instance of a high throughput, low goodput situation). In this paper we study, from the theoretical viewpoint, algorithms that decide which packets to drop so as to maximize the goodput of bounded-capacity links.

To this end, we consider the following abstract model. Senders generate basic information units, called *superpackets*, that are broken into packets by the network protocol at the senders. The router needs to decide which link is used by each new superpacket: all subsequent packets of that superpacket must use

[3] In fact, P-frames depend on the previous reference frame; and B-frames depend on both their immediate surrounding frames. In addition, MPEG partitions frames into "slices," which are transmitted in network packets.

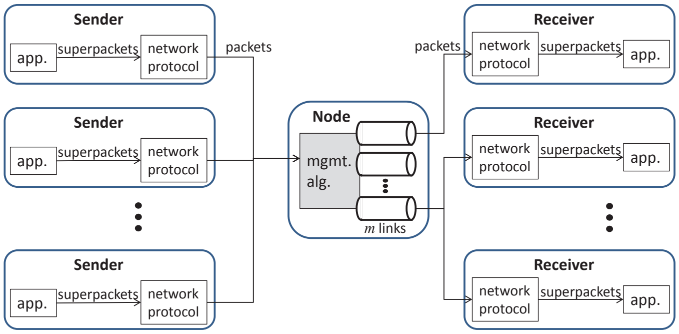

Fig. 2. *Basic system setup. Our focus is on the link management algorithm (shaded). All packets belonging to the same superpacket must use the same link.*

the same link.[4] If the number of packets assigned to a link exceeds its capacity, the management algorithm needs to decide which packet to drop and which to forward. To allow for arbitrary structure and redundancy, we assume that each superpacket is associated with a collection of *feasible subsets* that is closed under set inclusion (i.e., if $S \supset S'$ and S' is feasible, then so is S). A superpacket is considered useful only if the set of its delivered packets is one of its feasible subsets. The goal of the algorithm is to maximize the number useful superpackets at the receivers.

Our contribution. Following [3, 4], we study the fundamental Priority algorithm for link management, augmented with a simple randomized strategy that allocates superpackets to links based on the links capacities. Algorithm Priority assigns to each superpacket a random priority (based only on its weight, cf. Section 2.2), and in case of overflows, low-priority packets are dropped. This algorithm enjoys many nice features: in particular let us mention that it is highly distributed in the sense that it can be employed consistently in multiple locations without any communication overhead (see [3]). For our context, we note that the algorithm works without any knowledge of the feasible sets.

Our main result is competitive analysis of this simple algorithm. Specifically, suppose that at most σ packets arrive in a step and that each link can serve at most one packet per step; suppose further that each superpacket contains k packets. We prove that the algorithm guarantees expected goodput of $\Omega(\text{OPT}/(k\sqrt{\sigma/m}))$, where OPT denotes the maximal number of superpackets that can be delivered in the given input sequence and m is the number of links. In fact, we prove our result in a more general setting: first, we consider *weighted* goodput (i.e., when each superpacket has a different value, and the goal is to maximize the total value of useful superpackets), where the competitive ratio is not affected; and second, we consider *capacitated* links, where each link i has capacity c_i. In this case the expected weight of superpackets delivered by our

[4] This requirement, referred to as "stickiness" or "persistence" is typical in communication protocols, e.g. TCP.

algorithm is $\Omega(\text{OPT}/(k\sqrt{\sigma/c}))$, where $c = \sum_{i=1}^{m} c_i$. Notice that the competitive ratio depends only on the total available bandwidth c, regardless of the way it is broken into links. Also note that the competitive ratio improves linearly with $\sqrt{c}$. Finally, we provide a refined analysis that takes into account a parameter we define, called the *effective redundancy* of the input sequence.

We present a lower bound on the competitive ratio for the case of m unit capacity links without redundancy. Based on [3], we show that in this case, no randomized on-line algorithm can improve on our results by more than a polylogarithmic factor.

We show that our results extend to more general models. In some cases, there may be more than just two values for a superpacket (no value or full value). Superpackets may be structured so that there are a few "service levels," with different values, so that the value of a delivered superpacket is the value of the highest satisfied service level. We show that our algorithm is competitive in this model as well. We also show that our upper bound applies to the *instantaneous network* model, where we are given a network, a source s and a destination t, and the algorithm needs to choose a path from s to t for each superpacket: all packets of a superpacket must follow the same path. A conflict between two superpackets occurs, if the routes of the superpackets intersect, and there exists a time step in which packets from both arrive. The motivation for this model is the case where a superpacket is a set of short virtual circuits between s and t over a network of unit-capacity links.

Related work. Buffer overflow management has been studied extensively in the last decade from the competitive analysis viewpoint (starting with [5, 6]: see [7] for a recent survey). The simplest superpacket model, in which each superpacket consists of k packets that need all be delivered, was introduced in [8]. Emek at al. [3] consider the basic problem (k-packet superpackets, single link, no redundancy) under the name Online Set Packing, and introduce the PRIORITY algorithm (based on Turan's Theorem [9]). They prove an upper bound on the competitive ratio of PRIORITY and a lower bound on the competitive ratio of any on-line algorithm for that problem. In [4], basic redundancy is introduced: in our terms, there is a constant $0 \leq \beta < 1$ such that any subset of at least $(1-\beta)k$ packets is feasible (in other words, a super packet is useful if at most a β-fraction of its packets are lost). A general technique for dealing with buffers is also introduced in [4].

The offline version of single link management, without redundancy and superpacket structure, is equivalent to the Set Packing problem (SP), where each superpacket corresponds to a set and each time step corresponds to an element. SP is as hard as Maximum Independent Set even when all elements are contained in at most two sets (i.e., $\sigma \leq 2$), and therefore cannot be approximated to within an $O(n^{1-\epsilon})$-factor, for any $\epsilon > 0$, where n is the number of sets [10]. Letting T denote the number of time steps (elements), SP is $O(\sqrt{T})$-approximable, and hard to approximate within $T^{1/2-\epsilon}$, for any $\epsilon > 0$ [11]. When set size is at most k, SP is approximable within $\frac{k}{2} + \epsilon$, for any $\epsilon > 0$ [12] and within $\frac{k+1}{2}$ in the weighted case [13], but known to be hard to approximate to within $O(k/\log k)$-factor [14].

Paper organization. The remainder of the paper is organized as follows. In Section 2 we formalize the model and present our algorithm. The analysis of our algorithm and the lower bound are given in Section 3. Extensions are given in Section 4.

2 Preliminaries

2.1 Models

Data model. Our basic concept is a *superpacket*, typically denoted S, which is comprised of k *packets*. The complete set of superpackets is denoted $\mathcal{C}$. Each superpacket $S \in \mathcal{C}$ is associated with a *feasibility collection* $\mathcal{F}_S \subseteq 2^S$. $\bar{S} \in \mathcal{F}_S$ is called a *feasible subset* of S. We assume that $\mathcal{F}_S$ is closed under set inclusion, or *monotone*, for any superpacket S, namely that if $\bar{S} \in \mathcal{F}_S$, then $S' \in \mathcal{F}_S$ for any S' such that $\bar{S} \subseteq S'$. The case where $\mathcal{F}_S = \{S\}$, for every $S \in \mathcal{S}$, is referred to as *all-or-nothing*. In this case a superpacket is lost if even one of its packets is dropped.

Each superpacket $S \in \mathcal{C}$ has a *weight* $w(S) > 0$. In the *unweighted* model, $w(S) = 1$ for all $S \in \mathcal{C}$. Given a set of superpackets $\mathcal{C}' \subseteq \mathcal{C}$ we define $w(\mathcal{C}') \overset{\text{def}}{=} \sum_{S \in \mathcal{C}'} w(S)$. The input is a sequence of packets that arrive online. We assume that the online algorithm can associate packets with superpackets (e.g., packets contain their parent superpacket ID in their headers). We stress, however, that the algorithm has no knowledge on feasibility collections of superpackets. The system progresses in discrete time steps, where the time horizon is denoted by T. In each step t, a set of $\sigma(t)$ packets arrive. (We assume that no superpacket has two packets arriving at the same step.) The arrival time of a packet p is denoted by $\text{arr}(p)$. The set of superpackets whose packet arrive at time t is denoted $\mathcal{C}(t)$, i.e., $\mathcal{C}(t) = \{S \in \mathcal{C} : \exists p \in S \text{ s.t. } \text{arr}(p) = t\}$. The *burst size* at time t is denoted $\sigma(t) = |\{p : \text{arr}(p) = t\}|$; the *weighted* burst size is denoted $\sigma_\$(t) = \sum_{S \in \mathcal{C}(t)} w(S)$.

System model. In the *single-link model*, we have an integer *capacity* $c \geq 1$, and the algorithm selects, at each time step t, c packets to forward. All other packets that arrived at time t are lost (possibly causing the loss of their superpackets). In the *multiple links* model, there are m links with capacities $c_1, \ldots, c_m$, where $c_i \geq 1$ for $i = 1, \ldots, m$. We denote $c = \sum_{i=1}^{m} c_i$. The algorithm selects a single link for each superpacket (for all its packets), and then, in each time step, the algorithm does, for each link, the single link task: select which packets will be forwarded, subject to that link capacity constraint.

Given an algorithm ALG and an instance $\mathcal{I}$, we denote the set of completed superpackets by ALG($\mathcal{I}$) (or simply by ALG), and call it the *goodput* of the algorithm. If the algorithm is randomized, its goodput for a given instance is a random variable, and we shall refer to its expected value. We measure the performance of algorithms using competitive analysis: The *competitive ratio* of an algorithm is the supremum, over all instances $\mathcal{I}$, of $w(\text{OPT}(\mathcal{I}))/w(\text{ALG}(\mathcal{I}))$, where $w(\text{OPT}(\mathcal{I}))$ is the maximum possible goodput for $\mathcal{I}$.

Additional notation. We define for every set of packets S,

$$N[S] \overset{\text{def}}{=} \{S' \in \mathcal{C} : \exists p \in S, p' \in S' \text{ s.t. } \operatorname{arr}(p) = \operatorname{arr}(p')\}$$

and $N(S) \overset{\text{def}}{=} N[S] \backslash \{S\}$. Notice that $N(S) = N[S]$ if $S \notin \mathcal{C}$. For a finite sequence of values $x_1, \ldots, x_n$, we denote $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $x_{\max} = \max\{x_1, \ldots, x_n\}$. We use the notation $\overline{xy} = \frac{1}{n}\sum_{i=1}^{n} x_i y_i$.

2.2 Algorithm

The following algorithm was proposed in [3] for the Online Set Packing problem.

Algorithm 1 : Priority

```
For each superpacket $S$ with weight $w(S)$: select a priority $r(S) \in [0,1]$ independently at random by the cumulative distribution function $\Pr[r(S) < x] = x^{w(S)}$.
for all time step $t$ do
    Receive $\sigma(t)$ packets
    Deliver the $c$ packets whose superpackets have the largest priority.
end for
```

Note that, if $w(S)$ is integral, the priority $r(S) \in [0,1]$ of a superpacket S is distributed like the maximum of $w(S)$ independent $U[0,1]$ random variables.

For our case, where we have m links, we use the following simple link allocation algorithm. Each link is then managed by its own replica of Priority.

Algorithm 2 : PLink

```
Whenever a packet $p$ from superpacket $S$ arrives:
if $p$ is the first packet from $S$ then
    set $\ell(S) \in \{1, \ldots, m\}$ randomly by $\Pr[\ell(S) = i] = \frac{c_i}{c}$
end if                                        ▷ else $\ell(S)$ was set previously
send $p$ to link $\ell(S)$ (managed by Algorithm Priority$_{\ell(S)}$)
```

3 Multiple Capacitated Links

In this section we study the case where weighted superpackets arrive at a server with multiple links. We show that the competitive ratio of PLink is $O(k\sqrt{\overline{\sigma\sigma_\$}/c \cdot \overline{\sigma_\$}})$. (Recall that c is the total capacity of all links.) We also present an almost matching lower bound for the all-or-nothing case with unit capacity links that is based on the lower bound for single unit capacity link from [3]. Our lower bound applies even to unweighted input sequences. We conclude the section with a discussion on the difference between the effects of many links and of large capacity.

3.1 Analysis of Algorithm PLink

We start by stating lower bounds on the survival probability of a superpacket S under Priority, in the single link case. (We abuse notation by using Priority to denote the set of surviving superpackets.) Note that the bounds hold for arbitrary feasible subsets of S. (Similar results appear in [3], but assuming that the only feasible set is S itself). Due to lack of space, the proofs are omitted.

Lemma 1. *Let $c = 1$. For any superpacket $S \in \mathcal{C}$ and for any feasible subset $\bar{S}$ of S we have* $\Pr[S \in \textsc{Priority}] \geq \frac{w(S)}{w(N(\bar{S}))+w(S)}$.

Lemma 2. *Let $c > 1$. For any superpacket $S \in \mathcal{C}$ and for any feasible subset $\bar{S}$ of S we have* $\Pr[S \in \textsc{Priority}] \geq \frac{1}{2} \cdot \min\left\{\frac{c \cdot w(S)}{w(N(\bar{S}))+w(S)}, 1\right\}$.

We now extend the lemmas to the multiple links case. We abuse notation once again by using PLink to denote the set of surviving superpackets.

Lemma 3. *For any $S \in \mathcal{C}$ and for any feasible subset $\bar{S}$ of S we have*

$$\Pr[S \in \textsc{PLink}] \geq \frac{c}{4} \cdot \frac{w(S)}{2w(N(\bar{S})) + c \cdot w(S)} \ .$$

Proof. Let $\ell(S)$ denote the link that was selected for S by PLink. Also let $N_i(S) = \{S' \in N(S) : \ell(S') = i\}$. By the independence of the random choices we get that $\mathbb{E}[w(N_i(\bar{S}))] = \frac{c_i}{c} \cdot w(N(\bar{S}))$, and by Markov's Inequality we have that $\Pr\left[w(N_i(\bar{S})) > \frac{2c_i}{c} \cdot w(N(\bar{S}))\right] < \frac{1}{2}$.

According to Lemmas 1 and 2 it follows that

$$\begin{aligned}
\Pr\left[S \in \textsc{PLink} \mid \substack{\ell(S)=i \text{ and} \\ w(N_i(\bar{S})) \leq \frac{2c_i w(N(\bar{S}))}{c}}\right] &\geq \frac{1}{2} \min\left\{\frac{c_i w(S)}{\frac{2c_i w(N(\bar{S}))}{c} + w(S)}, 1\right\} \\
&\geq \frac{1}{2} \min\left\{\frac{c_i w(S)}{\frac{2c_i w(N(\bar{S}))}{c} + c_i w(S)}, 1\right\} \\
&= \frac{1}{2} \min\left\{\frac{c w(S)}{2w(N(\bar{S})) + c w(S)}, 1\right\} \\
&= \frac{1}{2} \cdot \frac{c w(S)}{2w(N(\bar{S})) + c w(S)}
\end{aligned}$$

Hence

$$\begin{aligned}
\Pr[S \in \textsc{PLink} \mid \ell(S) = i] \geq & \Pr\left[w(N_i(\bar{S})) \leq \frac{2c_i w(N(\bar{S}))}{c} \mid \ell(S) = i\right] \cdot \\
& \Pr\left[S \in \textsc{PLink} \mid \substack{\ell(S)=i \text{ and} \\ w(N_i(\bar{S})) \leq \frac{2c_i w(N(\bar{S}))}{c}}\right] \\
\geq & \frac{c}{4} \cdot \frac{w(S)}{2w(N(\bar{S})) + c w(S)} \ .
\end{aligned}$$

Therefore,

$$\Pr[S \in \text{PL}\textsc{ink}] \;=\; \sum_i \frac{c_i}{c} \cdot \Pr[S \in \text{PL}\textsc{ink} \mid \ell(S) = i] \;\geq\; \frac{c}{4} \cdot \frac{w(S)}{2w(N(\bar{S})) + cw(S)} \;.$$

□

The following lemma states the property that allows us to bound the competitive ratio of PLink.

Lemma 4. *Let $\mathcal{C}' \subseteq \mathcal{C}$ be a collection of superpackets, and for every $S \in \mathcal{C}'$ let $\bar{S} \subseteq S$ be a feasible subset of S. Then, either (i) $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{w(\mathcal{C}')}{8}$, or (ii) $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{c}{16} \frac{w(\mathcal{C}')^2}{\sum_{S \in \mathcal{C}'} w(N(\bar{S}))}$.*

Proof. By linearity of expectation we have

$$\begin{aligned}
\mathbb{E}[w(\text{PL}\textsc{ink})] \;&=\; \sum_{S \in \mathcal{C}} w(S) \cdot \Pr[S \in \text{PL}\textsc{ink}] \\
&\geq\; \sum_{S \in \mathcal{C}'} \frac{w(S)}{4} \frac{c \cdot w(S)}{2w(N(\bar{S})) + cw(S)} \\
&=\; \frac{c}{4} \sum_{S \in \mathcal{C}} \frac{w(S)^2}{2w(N(\bar{S})) + cw(S)} \\
&\geq\; \frac{c}{4} \cdot \frac{w(\mathcal{C}')^2}{2\sum_{S \in \mathcal{C}'} w(N(\bar{S})) + cw(\mathcal{C}')} \;,
\end{aligned}$$

where the first inequality follows from Lemma 3 and the second is due to the following implication of the Cauchy-Schwarz Inequality: for any positive reals $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$, it holds that $\sum_i \frac{a_i^2}{b_i} \geq \frac{\left(\sum_i a_i\right)^2}{\sum_i b_i}$.

If $c \cdot w(\mathcal{C}') \geq 2\sum_{S \in \mathcal{C}'} w(N(\bar{S}))$, then $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{c}{4} \cdot \frac{w(\mathcal{C}')^2}{2cw(\mathcal{C}')} = \frac{w(\mathcal{C}')}{8}$, and otherwise, $\mathbb{E}[w(\text{PL}\textsc{ink})] \;>\; \frac{c}{16} \cdot \frac{w(\mathcal{C}')^2}{\sum_{S \in \mathcal{C}'} w(N(\bar{S}))}$. □

Lemmas 5 and 6 below apply Lemma 4 with two different collections $\mathcal{C}'$.

Lemma 5. *Either $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{w(\textsc{opt})}{8}$, or $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{w(\textsc{opt})^2}{16kw(\mathcal{C})}$.*

Proof. For each superpacket $S \in \textsc{opt}$ fix $\bar{S}$ to be the subset of S which contains the packets delivered by $\textsc{opt}$. Clearly $\bar{S}$ is a feasible subset of S. Hence, by Lemma 4 with $\mathcal{C}' = \textsc{opt}$ we have that either $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{w(\textsc{opt})}{8}$ or $\mathbb{E}[w(\text{PL}\textsc{ink})] \;\geq\; \frac{c}{16} \frac{w(\textsc{opt})^2}{\sum_{S \in \textsc{opt}} w(N(\bar{S}))}$. In the latter case, observe that each superpacket in $\mathcal{C}$ intersects at most ck superpackets in $\textsc{opt}$. Hence, $\sum_{S \in \textsc{opt}} w(N(\bar{S})) \;\leq\; ckw(\mathcal{C})$. It follows that $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{w(\textsc{opt})^2}{16kw(\mathcal{C})}$. □

Lemma 6. *Either $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{w(\mathcal{C})}{8}$, or $\mathbb{E}[w(\text{PL}\textsc{ink})] \geq \frac{cw(\mathcal{C})^2}{16T \cdot \overline{\sigma\sigma_\$}}$.*

Proof. Fix a superpacket S and let $\bar{S} = S$. By Lemma 4 with $\mathcal{C}' = \mathcal{C}$ we have that either $\mathbb{E}[w(\textsc{PLink})] \ge \frac{w(C)}{8}$, or $\mathbb{E}[w(\textsc{PLink})] \ge \frac{c}{16} \cdot \frac{w(\mathcal{C})^2}{\sum_{S\in\mathcal{C}} w(N(S))}$. Summing over the superpackets we get that

$$\sum_{S\in\mathcal{C}} w(N(S)) \;<\; \sum_{t} \sigma(t)\sigma_{\$}(t) \;=\; T \cdot \overline{\sigma\sigma_{\$}} \;, \tag{1}$$

and the lemma follows. □

Combining Lemmas 5 and 6 we obtain our main result.

Theorem 1. *The competitive ratio of* PLink *is at most* $16k\sqrt{\frac{\overline{\sigma\sigma_{\$}}}{c\overline{\sigma_{\$}}}}$.

Proof. If either $w(\textsc{PLink}) \ge w(\textsc{opt})/8$ or $w(\textsc{PLink}) \ge w(\mathcal{C})/8$, then we are done. Otherwise, we have that

$$\mathbb{E}[w(\textsc{PLink})] \ge \frac{w(\textsc{opt})^2}{16kw(\mathcal{C})} \qquad \text{and} \qquad \mathbb{E}[w(\textsc{PLink})] \ge \frac{cw(\mathcal{C})^2}{16T \cdot \overline{\sigma\sigma_{\$}}} \;.$$

The maximum of these bounds is minimized when $w(\textsc{opt}) = \sqrt{\frac{ck \cdot w(\mathcal{C})^3}{T \cdot \overline{\sigma\sigma_{\$}}}}$, and therefore, for any instance

$$\mathbb{E}[w(\textsc{Priority})] \;\ge\; w(\textsc{opt}) \cdot \frac{1}{16}\sqrt{\frac{cw(\mathcal{C})}{k \cdot T \cdot \overline{\sigma\sigma_{\$}}}}.$$

Finally, since $T \cdot \overline{\sigma_{\$}} \;=\; \sum_t \sigma_{\$}(t) \;\le\; \sum_{S\in\mathcal{C}} k \cdot w(S) \;=\; k \cdot w(\mathcal{C})$, it follows that

$$\mathbb{E}[(\textsc{Priority})] \;\ge\; w(\textsc{opt}) \cdot \frac{1}{16}\sqrt{\frac{c \cdot \overline{\sigma_{\$}}}{k^2 \overline{\sigma\sigma_{\$}}}} \;=\; w(\textsc{opt}) \cdot \frac{1}{16k}\sqrt{\frac{c \cdot \overline{\sigma_{\$}}}{\overline{\sigma\sigma_{\$}}}} \;.$$

□

Note that the upper bound we provide in Theorem 1 does not depend on the number of links, but rather on the input sequence and on the total capacity of the links.

Corollary 1. *The competitive ratio of* PLink *is at most* $16k\sqrt{\sigma_{\max}/c}$.

Proof. Follows from the fact that $\overline{\sigma\sigma_{\$}} \le \overline{\sigma_{\$}} \cdot \sigma_{\max}$. □

3.2 A Lower Bound

We now present a lower bound for the multiple links case. It uses the simple scenario of unweighted, unit-capacity per link (i.e., $m = c$) instances, and thus it applies to more general setting a fortiori. However, we assume that the only feasible subset of a superpacket is all packets, i.e., no redundancy is considered.

Our lower bound uses, as a black box, the following lower bound from [3] for Online Set Packing (osp).

Theorem 2 ([3]). *For any randomized online algorithm, there exists an infinite family of unweighted, unit-capacity instances of* OSP *for which the competitive ratio is* $\tilde{\Omega}(k\sqrt{\sigma_{\max}})$.

Next building on Theorem 2 we obtain a lower bound for the multiple uncapacitated links case.

Theorem 3. *For any online randomized algorithm there exists an infinite family of unweighted, instances for which, under the* m *unit capacity link model, the competitive ratio is* $\tilde{\Omega}(k\sqrt{\sigma_{\max}/c})$.

Proof. Let $\mathcal{I}'$ be the instance whose existence is promised by Theorem 2. Define an instance $\mathcal{I}$ where each superpacket in $\mathcal{I}'$ is replicated c times. (Note that $c = m$ in this case.) Clearly, $|\text{OPT}(\mathcal{I})| \geq c \cdot |\text{OPT}(\mathcal{I}')|$, since it is possible to route the ith copy of each set to link i. We show that given any randomized online algorithm ALG for the multiple links case, one can obtain an algorithm ALG$'$ for the single link case such that $\mathbb{E}[\,|\text{ALG}'|\,] \geq \mathbb{E}[\,|\text{ALG}|\,]/c$. Hence, $|\text{OPT}|/\mathbb{E}[\,|\text{ALG}|\,] \geq |\text{OPT}'|/\mathbb{E}[\,|\text{ALG}'|\,]$, and the theorem follows.

Given an algorithm ALG, define ALG$_i$ to be the set of completed superpackets that were routed to link i. Let ℓ be the link that maximizes performance, i.e., $\ell = \text{argmax}_i\, \mathbb{E}[|\text{ALG}_i|]$. Clearly, $\mathbb{E}[|\text{ALG}_i|] \geq \mathbb{E}[|\text{ALG}|]/c$. We construct an algorithm ALG$'$ for OSP that simulates link ℓ. More specifically, given an input sequence, ALG$'$ makes m copies of each superpacket, and executes ALG on the new instance. Let ALG$_\ell(t)$ be the set of superpackets whose packets were transmitted by ALG on link ℓ at time t. If $S_j \in \text{ALG}_\ell(t)$, where S^j is a copy of S, then ALG$'$ transmits a corresponding packet from S, namely $\text{ALG}'(t) = \left\{S \;:\; \exists j,\; S^j \in \text{ALG}_\ell(t)\right\}$. Since no two copies of S can be completed by ALG, it follows that $|\text{ALG}'| \geq |\text{ALG}_\ell|$. Hence, $\mathbb{E}[|\text{ALG}'|] \geq \mathbb{E}[|\text{ALG}|]/c$, as required. □

We note that our lower bound shows that our upper bound is essentially tight—for the case of unit capacity links.

3.3 The Effect of Many Links and Large Capacity

As we mentioned above, it is interesting to note that the competitive ratio of Algorithm PLink depends only on the total available bandwidth, regardless of how it is partitioned among the links. However, the lower bound of Theorem 3 is proved specifically for the case of unit capacity links. It is natural to ask whether link capacity plays an important role in algorithm performance. The answer is positive, as demonstrated by the following scenario. Consider two models, one with m unit-capacity links and another with a single link with capacity m. Suppose that there are $n \geq m$ superpackets without any redundancy, and let the arrival sequence consists of all possible $\binom{n}{m}$ bursts of size m in arbitrary order. In the m unit-capacity links model, only m superpackets can be completely delivered, because each channel can deliver only one complete superpacket. However, in the single-link capacity m model, all superpackets are delivered. This means that the optimum may change dramatically when links are consolidated.

We conclude this section with two observations about the effect of sufficiently many unit capacity links. The proofs are omitted.

First, we consider the effect on the optimal solution.

Observation 7. *If* $m > k(\sigma_{\max} - 1)$*, then* $\textsc{opt} = \mathcal{C}$*.*

Next, we consider the effect on the competitive ratio,

Theorem 4. *Suppose that* ALG *is an algorithm that assigns superpackets to links uniformly at random, and consider unweighted instances without redundancy. Then if* $m \geq \frac{k\overline{\sigma^2}}{\epsilon\bar{\sigma}}$ *then* $\mathbb{E}[\,|\textsc{alg}|\,] \geq (1-\epsilon)|\textsc{opt}|$*.*

We note that since $\frac{\overline{\sigma^2}}{\bar{\sigma}} \leq \sigma_{\max}$, the same result holds for $m \geq \frac{k\sigma_{\max}}{\epsilon}$.

4 Extensions

In this section we present a refinement of the analysis of PLink, and then we extend the analysis to more general settings. More specifically, we provide a refined analysis of PLink that takes into account the *effective redundancy* of the input sequence (Section 4.1). We show that PLink can be used in the case where there are several feasibility collections for each superpacket, and each collection is associated with a different service level (Section 4.2). We also extend our results to the *instantaneous network* model (Section 4.3).

4.1 Effective Redundancy

To refine the analysis of Algorithm PLink, we defined the following concepts. The *burstiness* of a superpacket S is defined as $\mathrm{B}(S) = \sum_{p \in S} \sigma_\$(\mathrm{arr}(p))$. The minimal burstiness of S is $\mathrm{B}_{\min}(S) = \min_{\bar{S} \in \mathcal{F}_S} \mathrm{B}(\bar{S})$. Let $\rho_S = \mathrm{B}_{\min}(S)/\mathrm{B}(S)$ and $\rho = \max_{S \in \mathcal{C}} \rho_S$. ρ is called the *effective redundancy* of the input sequence.

We now refine Lemma 6 to include ρ as follows.

Lemma 8. *Either* $\mathbb{E}[w(\text{PLink})] \geq \frac{w(\mathcal{C})}{8}$*, or* $\mathbb{E}[w(\text{PLink})] \geq \frac{cw(\mathcal{C})^2}{16\rho T \cdot \overline{\sigma\sigma_\$}}$*.*

Proof. We follow the proof of Lemma 6, but we take $\bar{S}$ to be a feasible subset of S with minimal burstiness, namely such that $\mathrm{B}(\bar{S}) = \mathrm{B}_{\min}(S)$. Equation (1) is replaced with $\sum_{S \in \mathcal{C}} w(N(\bar{S})) < \sum_{S \in \mathcal{C}} \mathrm{B}(\bar{S}) \leq \sum_{S \in \mathcal{C}} \rho \mathrm{B}(S) = \rho \cdot T \cdot \overline{\sigma\sigma_\$}$. □

We can therefore conclude that in this case we have an improvement of $\sqrt{\rho}$ factor over Theorem 1:

Theorem 5. *The competitive ratio of* PLink *is at most* $16k\sqrt{\frac{\rho\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}}$*.*

We note that ρ decreases if bursts are roughly the same weight or if there are no packets whose delivery is essential to the survival of superpackets.

To motivate the parameter ρ, consider the case where superpackets are hierarchically structured. Specifically, we assume that the feasible collection of a superpacket S is defined by a *structure tree* T_S, whose leaves are the packets, and whose root is identified with the superpacket. A structure tree is a rooted tree with a *redundancy parameter* assigned to each node, subject to the following restriction for internal nodes: Let $d(v)$ denote the number of children of a node v. The redundancy parameter of a node v is $\beta_v^S \in \{0, \frac{1}{d_v}, \ldots, \frac{d_v - 1}{d_v}\}$. (Assume w.l.o.g. that $\beta_v^S = 0$ if $d(v) \leq 1$.) The interpretation of redundancy is defined recursively as follows. A subset S' of the leaves of a structure tree T is said to be *feasible* if either of the following conditions hold: (i) S' contains one packet and T contains one leaf; or (ii) Let $v_1, \ldots, v_d$ be the children of the root of T, with structure trees $T_1, \ldots, T_d$, respectively. Let $S'_1, \ldots, S'_d$ be the subsets of S' corresponding to $T_1, \ldots, T_d$, resp. Then S' is feasible if at least $(1 - \beta_r^S)d$ of the subsets $S'_1, \ldots, S'_d$ are feasible.

Consider a superpacket S with its structure tree T_S. We define the *redundancy* of a leaf v in T_S as $\hat{\rho}_v \overset{\text{def}}{=} \prod_{i=0}^{\ell}(1 - \beta_{v_i}^S)$, where $v = v_0, v_1, \ldots, v_\ell = r$ is the path from v to the root r. The redundancy of the superpacket S is defined as $\hat{\rho}_S \overset{\text{def}}{=} \max_{v \text{ is a leaf}} \rho_v$. Notice that $\hat{\rho}_S$ depends on T_S and on β^S, but not on the input sequence.

Observation 9. *$\rho_S \leq \hat{\rho}_S$ for every superpacket $S \in \mathcal{C}$.*

It follows that we can replace ρ with $\hat{\rho} = \max_S \hat{\rho}_S$ in Theorem 5. We note that $\hat{\rho}_S = 1$ in the GoP example (see Figure 1), since the I-frame is contained in any feasible set. However, the competitive ratio will improve, if we send the I-frame twice.

4.2 Multiple Service Levels

In the model considered in Section 3, each superpacket S has a single weight (value) $w(S)$ that is collected if a feasible subset of S is delivered. In some cases, there may be more than just two values for the superpacket (no value or full value). We consider this case here. Intuitively, we consider superpackets structured so that there are a few "service levels," with different values, so that the value of a delivered superpacket is the value of the highest satisfied service level. We show that Algorithm PLink is competitive in this case as well.

Formally, we assume that with each superpacket i there are ℓ feasibility collections $\mathcal{F}_S^1 \supset \mathcal{F}_S^2 \cdots \supseteq \mathcal{F}_S^\ell$ and a weight $w(S)$. There are also ℓ payment levels $0 < \alpha_1 < \cdots < \alpha_\ell \leq 1$, such that the profit obtained from a superpacket S with delivered packets S' is $\alpha_i \cdot w(S)$, where i is the maximal service level i such that $S' \subseteq \mathcal{F}_S^i$.

Let $w_i(S) \overset{\text{def}}{=} w(S)(\alpha_i - \alpha_{i-1})$, for $i = 1, \ldots, \ell$, namely $w_i(S)$ stands for the marginal profit obtained by going from service level $i - 1$ to service level i.

Let $\textsc{opt}_i$ denote the optimal value with respect to the instance with the weight function w_i.

Lemma 10. *Executing Algorithm* PLINK *with the original weights results in* $\mathbb{E}[w_i(\text{PLINK})] \geq 16k\sqrt{\frac{\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}} \cdot w_i(\textsc{opt}_i)$ *for a service level i, where ρ_i is defined by the feasible collections for service level i.*

Proof. Since w_i is proportional to the original weight function w, for every i, it follows that the analysis of Algorithm PLINK continues to hold with respect to w_i, even if the random priorities are chosen according to w (see Lemmas 1-3). Therefore $\mathbb{E}[w_i(\text{PLINK})] \geq 16k\sqrt{\overline{\sigma\sigma_{i\$}}/c\overline{\sigma_{i\$}}} \cdot w_i(\textsc{opt}_i) = 16k\sqrt{\overline{\sigma\sigma_\$}/c\overline{\sigma_\$}} \cdot w_i(\textsc{opt}_i)$. □

Theorem 6. *The competitive ratio of* PLINK *is at most* $16k\sqrt{\frac{\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}}$.

Proof. By linearity of expectation and Observation 10 we have that

$$\begin{aligned}
\mathbb{E}[w(\text{PLINK})] &= \sum_i \mathbb{E}[w_i(\text{PLINK})] \\
&\geq \sum_i 16k\sqrt{\frac{\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}} \cdot w_i(\textsc{opt}_i) \\
&\geq 16k\sqrt{\frac{\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}} \sum_i w_i(\textsc{opt}) \qquad = 16k\sqrt{\frac{\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}} \cdot w(\textsc{opt}) \,.
\end{aligned}$$

□

4.3 Instantaneous Network Model

We can extend our results to the following scenario we call the instantaneous network model. In this model we are given a graph with unit capacity edges and two distinguished nodes, a source s and a destination t, and the algorithm needs to choose a path from s to t for each superpacket: all packets of a superpacket must follow the same path. A conflict between superpackets S and S' occurs if the routes of S and S' intersect, and there exists a time step in which packets from both S and S' arrive.

We observe that the instantaneous network model can be reduced to the unit capacity multiple links model.

Lemma 11. *There exists a reduction from the instantaneous network model to the unit capacity multiple links model.*

Proof. Consider any feasible solution. For each superpacket S, let $p(S)$ be the path from s to t that is used for S by the solution. Let $C = \{e_1, ..., e_f\}$ be a minimum s,t-cut in the network, and for each superpacket S, let $e(S)$ be some edge in C that is contained in $p(S)$, namely $e(S) \in C \cap p(S)$. Define

$\mathcal{C}_i = \{S : e(S) = e_i\}$. Clearly, $\cup_i \mathcal{C}_i = \mathcal{C}$. Let $p_1, ..., p_f$ be f simple edge disjoint paths from s to t, where $e_i \in p_i$. We reassign superpackets to paths as follows: $p'(S) = p_i$ if $e(S) = e_i$, namely if $S \in \mathcal{C}_i$. Since the superpackets in $\mathcal{C}_i$ intersect at e_i using p, no new conflict is introduced by the new assignment p'. □

It follows that

Theorem 7. *There exists a randomized algorithm for the instantaneous network model whose competitive ratio is at most* $16k\sqrt{\frac{\overline{\sigma\sigma_\$}}{c\overline{\sigma_\$}}}$.

Theorem 8. *For any online randomized algorithm for the instantaneous network model there exists an infinite family of unweighted, instances for which the competitive ratio is* $\tilde{\Omega}(k\sqrt{\sigma_{\max}/c})$.

References

1. International Organization for Standardization: MPEG-2 standard (2000) ISO/IEC 13818-2:2000.
2. World Wide Web Consortium: Extensible markup language (XML) 1.0. W3C Recommendation (November 2008) http://www.w3.org/TR/REC-xml/ .
3. Emek, Y., Halldórsson, M.M., Mansour, Y., Patt-Shamir, B., Radhakrishnan, J., Rawitz, D.: Online set packing and competitive scheduling of multi-part tasks. In: 29th Annual ACM Symposium on Principles of Distributed Computing. (2010)
4. Mansour, Y., Patt-Shamir, B., Rawitz, D.: Overflow management with multipart packets. In: IEEE INFOCOM. (2011)
5. Mansour, Y., Patt-Shamir, B., Lapid, O.: Optimal smoothing schedules for real-time streams. In: 19th Annual ACM Symposium on Principles of Distributed Computing. (2000) 21–30
6. Kesselman, A., Lotker, Z., Mansour, Y., Patt-Shamir, B., Schieber, B., Sviridenko, M.: Buffer overflow management in QoS switches. In: 33rd Annual ACM Symposium on Theory of Computing. (2001) 520–529
7. Goldwasser, M.H.: A survey of buffer management policies for packet switches. SIGACT News **41**(1) (2010) 100–128
8. Kesselman, A., Patt-Shamir, B., Scalosub, G.: Competitive buffer management with packet dependencies. In: 23rd IPDPS. (2009) 1–12
9. Alon, N., Spencer, J.H.: The Probabilistic Method. 3rd edn. Wiley Interscience (2008)
10. Håstad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. Acta Mathematica **182**(1) (1999) 105–142
11. Halldórsson, M.M., Kratochvíl, J., Telle, J.A.: Independent sets with domination constraints. Discrete Applied Mathematics **99**(1–3) (2000) 39–54
12. Hurkens, C.A.J., Schrijver, A.: On the size of systems of sets every t of which have an SDR, with an application to the worst-case ratio of heuristics for packing problems. SIAM J. Discrete Math. **2**(1) (1989) 68–72
13. Berman, P.: A $d/2$ approximation for maximum weight independent set in d-claw free graphs. Nord. J. Comput. **7**(3) (2000) 178–184
14. Hazan, E., Safra, S., Schwartz, O.: On the complexity of approximating k-dimensional matching. In: 6th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems. Volume 2764 of LNCS. (2003) 83–97