Online Set Packing and Competitive Scheduling of Multi-Part Tasks*

Extended Abstract

Yuval Emek[†‡] yuvale@eng.tau.ac.il

Magnús M. Halldórsson[§] mmh@ru.is

Yishay Mansour[¶] mansour@cs.tau.ac.il

Boaz Patt-Shamir[†‖] boaz@eng.tau.ac.il

Jaikumar Radhakrishnan[**] jaikumar@tifr.res.in

Dror Rawitz[†] rawitz@eng.tau.ac.il

February 17, 2010

Abstract

We consider a scenario where large data frames are broken into a few packets and transmitted over the network. Our focus is on a bottleneck router: the model assumes that in each time step, a set of packets (a burst) arrives, from which only one packet can be served, and all other packets are lost. A data frame is considered useful only if none of its constituent packets is lost, and otherwise it is worthless. We abstract the problem as a new type of *online set packing*, present a randomized distributed algorithm and a matching lower bound on the competitive ratio for any randomized online algorithm. Our bounds are expressed in terms of the maximal burst size and the maximal number of packets per frame. We also present refined bounds that depend on the uniformity of these parameters.

Keywords: online set packing, competitive analysis, packet fragmentation, multi-packet frames

---

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

†School of Electrical Engineering, Tel Aviv University, Tel Aviv 69978, Israel.

‡Microsoft Israel R&D Center, Herzeliya, Israel.

§School of Computer Science, Reykjavik University, 103 Reykjavik, Iceland. Supported in part by the Icelandic Research Fund (grant 90032021).

¶School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. Supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886, 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). This publication reflects the authors' views only.

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

**School of Technology and Computer Science, Tata Institute of Fundamental Research, Homi Bhabha Road, Mumbai 400005, India.

1 Introduction

In video transmission over the Internet, the situation is typically as follows. Data is produced in large units at the source (a video frame may be hundreds of kilobytes long), while networks use small transfer units (e.g., packets sent over Ethernet are limited to 1.5 kilobytes), and therefore in many cases, video frames are broken into a number of small packets. However, a video frame is useful at the receiving end only if *all* its constituent packets arrive. Now consider an outgoing link in a network switch (router): the capacity of the link is bounded, so when a large burst of packets headed to that link arrives, some packets will be dropped (let us ignore buffering for simplicity). The router, in case of an overflow, needs to decide online which packets to drop, so as to maximize the number of complete frames delivered.

We abstract the above scenario as an instance of the following optimization problem, which we call *on-line set packing* (OSP). There is a collection of sets whose elements are initially unknown. The elements arrive in an online fashion: When an element arrives, it announces which sets it belongs to, and the algorithm needs to assign the element to one of the sets containing it (before the next element arrives). Each set has a value, and the algorithm gets paid only for sets that are assigned all their elements: there is no reward for unfinished sets. The goal is to maximize the value of completed sets.

The video transmission scenario described above reduces to OSP a follows: elements represent time steps (not packets!), and sets represent data frames. Time step j is included in data frame i if a packet of frame i arrives at time j.

Another common scenario in packet-switching networks is the case of packets that need to traverse multiple hops: a packet is delivered only if it is not dropped by any of the switches along its route. Simplifying, this too can be viewed as a special case of as OSP using the following mapping. Let each pair (t, h) of time t and location h be modeled by an element of the OSP formulation, and let each packet be modeled by a set, whose elements are all time-location pairs which the packet is supposed to visit. In the simplified set packing model, we ignore buffers and assume that only one packet can be delivered at each (t, h) pair.

In general, sets may represent compound tasks that have mutiple parts that need to be served in possibly different locations in the system, in bounded-capacity servers. In this general case, the algorithm must be distributed: the algorithm designer needs also to worry about the availability of remote information.

Our contributions. In this paper, we introduce the problem of online set packing, present randomized distributed algorithms for it, and prove lower bounds on the competitive ratio of any (randomized, centralized) online algorithm for it. The problem appears to be a new fundamental online problem that, to the best of our knowledge, was not studied explicitly in the past. Regarding our bounds on the competitive ratio, we note that they are expressed as a function of the size of the sets, and the *load* of elements, where the load of an element is defined as the number of sets containing it. Specifically, let $k_{\max}$ denote the maximal set size (in video transmission, maximal number of packets per video frame) and $\sigma_{\max}$ denote the maximal element load (in video transmission, the maximal burst size, i.e., the maximal number of packets that may arrive simultaneously). Our first main result is a randomized distributed algorithm that guarantees to complete sets of total expected value at least

$\text{OPT}/(k_{\max}\sqrt{\sigma_{\max}})$, where OPT is the maximal value of any feasible solution. The result extends to the case of general capacity case: assume that each element j arrives with mutiplicity b_j, which allows it to be assigned to b_j sets. (In the video transmission scenario, b_j is the number of packets that can be served at time j.) In the case, our upper bound generalizes using adjusted load, defined as the load of element j divided by b_j.

We also derive more refined bounds on the competitive ratio which are sensitive to the variability of set sizes and element loads: the more uniform they are, the better bounds we get. For example, if all elements have the same load (in every time step σ packets arrive), then the competitive ratio drops to $k_{\max}$.

Our second main result is a lower bound that shows that no randomized online algorithm (including centralized algorithms) can have competitive ratio much better than $k_{\max}\sqrt{\sigma_{\max}}$, even in the unweighted case, and also show a simple lower bound of $(\sigma_{\max})^{k_{\max}-1}$ for deterministic algorithms. Our construction for the randomized case is a bit involved, and our technique uses combinatorial designs similar to projective planes.

Related work. The offline version of Set Packing is as hard as Maximum Independent Set (MIS) even when all elements have load 2, and therefore cannot be approximated to within $O(m^{1-\epsilon})$-factor, for any $\epsilon > 0$ [6] (m denotes the number of sets). Set Packing is $O(\sqrt{n})$-approximable, and hard to approximate within $n^{1/2-\epsilon}$ [8] (n denotes the number of elements). When set size is at most k, it is approximable within $k/2 + \epsilon$, for any $\epsilon > 0$ [10] and within $(k + 1)/2$ in the weighted case [4], but known to be hard to approximate to within $\Omega(k/\log k)$-factor [9].

Buchbinder and Naor have studied online primal-dual algorithms for covering and packing in the centralized setting (see, e.g., [5]). In their model, constraints of the primal (covering) program arrive one by one, and the variables can only be increased by the algorithm. This approach was applied to online set cover [1], and to the following variant of packing: In each step, a new set is introduced by listing all its elements; the algorithm may accept the set only if it is disjoint to previously accepted sets, but it may also reject it. If a set is accepted, the algorithm collects the set value immediately. In our setting, elements arrive one by one, and the benefit is earned only after a set is complete. Intuitively, both OSP and the packing framework of [5] share the same linear programming matrix of (given in Eq. (1) in Section 2 below), but in [5] columns arrive online, while in our formulation rows arrive online.

In fact, all previous online algorithms for packing (partially) disjoint structures, whether it be sets, vertices, or paths, assume that decisions are made on already completed structures. A factor k-approximation is trivial by a greedy algorithm for unweighted set packing in this case. When $k = 2$, we obtain an online matching problem, for which numerous results are known, starting with an $e/(e-1)$-competitive randomized algorithm of [11], that works for a weighted bipartite version with a certain restriction on arrival order. For online independent set problems, that relate to packing paths in graphs, very strong lower bounds generally hold [2, 7].

Throughput maximization of multi-packet frames was introduced in [12], with the additional complication of having a finite buffer. However, no bound on element load is assumed in [12] (i.e., arbitrarily many packets may arrive simultaneously at the server), and competitive algorithms were presented only under the assumption that the arrival process is "well ordered."[1] An upper bound of $O(k^2)$ and a lower bound of $\Omega(k)$ on the competitive factor for deterministic online algorithms in this model was given in [12]. These results are not directly comparable to ours, because of the difference in model: [12] considers only well-ordered instances and allows for bounded buffering.

Distributed models for solving linear programs (fractionally) are considered in [16, 14], where the complete matrix is input to the system at the start of execution, but it is distributed among the different agents. In [3] new variables and constraints can be introduced over time, and the system will stabilize to an approximately optimal solution, but the variables may be both increased and decreased by the algorithm, so it is not online in our sense.

Paper organization. The remainder of this paper is organized as follows. In Section 2 we formalize the problem and state our main results. In Section 3 we describe and analyze our algorithm. In Section 4 we prove our lower bounds. We conclude in Section 5 with some open problems.

2 Problem Statement and Results

Problem statement. A weighted set system consists of a set U of n elements, a family $\mathcal{C} = \{S_1, S_2, \ldots, S_m\}$ of m subsets of U, and a *weight function* assigning a non-negative weight $w(S)$ to each set $S \in \mathcal{C}$. We also assume that some *capacity* $b(u) \in \mathbb{N}$ is associated with each element $u \in U$. Formally, the offline version of the problem we consider is expressed by the following integer program:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{m} w_i x_i \\ \text{s.t.} \quad & \sum_{i:S_i \ni u_j} x_i \le b_j \qquad \text{for } j = 1, \ldots, n \\ & x_i \in \{0, 1\} \ . \end{aligned} \tag{1}$$

The value of x_i says whether set S_i is taken or not, w_i is the benefit obtained by completing set S_i, and b_j is the number of sets element u_j can be assigned to. We concentrate on the *online set packing* problem, defined as follows. Initially, for each set we know only its weight and size (but not its members). In each step i, a new element u_i arrives along with its capacity $b(u_i)$ and with $\mathcal{C}(u_i) \stackrel{\text{def}}{=} \{S \in \mathcal{C} : u_i \in S\}$, i.e., the names of all sets containing u_i. The algorithm must output in step i a collection of set names $\mathcal{A}(i) \subseteq \mathcal{C}(u_i)$ such that $|\mathcal{A}(i)| \le b(u_i)$. The algorithm is said to *complete* a set S if $S \in \mathcal{A}(i)$ for each of its elements $u_i \in S$. The output of an algorithm for an instance $\mathcal{I}$, denoted $\textsc{alg}(\mathcal{I})$ (or simply $\textsc{alg}$), is the collection of sets completed by the algorithm, and the benefit of the algorithm is in that case $w(\textsc{alg}(\mathcal{I}))$, where the weight of a collection of sets is the sum of the set weights. If the algorithm is randomized, the benefit for a given instance is a random variable, and we shall use 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 $w(\textsc{opt}(\mathcal{I}))/w(\textsc{alg}(\mathcal{I}))$, where $\textsc{opt}(\mathcal{I})$ denotes the collection of subsets of $\mathcal{C}$ that maximizes the target function in (1).

[1] An arrival process is said to be *well ordered* in this context if for any $1 \le i, j \le k$, the ith element of set A arrives before the ith element of set B if and only if the jth element of set A arrives before the jth element of set B.

Special interesting classes of instances are the *unweighted* instances where $w(S) = 1$ for all $S \in \mathcal{C}$, and the *unit capacity* instances where $b(u) = 1$ for all $u \in U$.

Main results. To state our results, we need to define some additional notation. Define, for each element $u \in U$, the *load* of u to be the number of sets containing it, and denote it by $\sigma(u)$, i.e., $\sigma(u) \stackrel{\text{def}}{=} |\mathcal{C}(u)|$. The *weighted load* of u is defined by $\sigma_\$(u) \stackrel{\text{def}}{=} \sum_{S \ni u} w(S) = w(\mathcal{C}(u))$. We denote $k_{\max} \stackrel{\text{def}}{=} \max\{|S| \, : \, S \in \mathcal{C}\}$ and $\overline{k} \stackrel{\text{def}}{=} \sum_{S \in \mathcal{C}} |S|/|\mathcal{C}|$, i.e., the maximal and average set size in $\mathcal{C}$, respectively. Throughout this paper we adopt the notational convention that for a multiset of numbers X, $\overline{X} \stackrel{\text{def}}{=} \sum_{x \in X} x/|X|$ and $X_{\max} \stackrel{\text{def}}{=} \max(X)$. The average applies also to the product of two values, in particular $\overline{\sigma \cdot \sigma_\$} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{u \in U} \sigma(u)\sigma_\$(u)$.

We can now state our upper bound for the unit capacity case.

Theorem 1 *There exists a randomized algorithm for* OSP *whose competitive ratio is at most* $k_{\max}\sqrt{\frac{\overline{\sigma \cdot \sigma_\$}}{\overline{\sigma_\$}}}$ *for unit-capacity instances.*

For the variable capacity case, the same bound holds, up to a constant factor, after replacing the load with *adjusted load*: the adjusted load of element u_j is $\nu(u_j) \stackrel{\text{def}}{=} \sigma(u_j)/b(u_j)$.

Note that the competitive ratio promised by Theorem 1 is never larger than $k_{\max}\sqrt{\sigma_{\max}}$, but can be much better: for example, if all sets have the same size k and all elements have the same load, then the competitive ratio improves to k in the unweighted case. We discuss some of these special cases in Section 3.3.

No algorithm can make a baseline improvement over our algorithm, or, more precisely:

Theorem 2 *For any online randomized algorithm there exists an infinite family of unweighted, unit-capacity instances of* OSP *for which the competitive ratio is* $\Omega\left(k_{\max}\left(\frac{\log\log k_{\max}}{\log k_{\max}}\right)^2 \sqrt{\sigma_{\max}}\right)$.

We note that our lower bound is based on Yao's principle: we build a distribution of the inputs for which the expected value for all deterministic algorithms is small. Also note that $k_{\max}$ and $\sigma_{\max}$ are linearly related in our construction.

The situation is much worse for deterministic algorithms:

Theorem 3 *The competitive ratio of any deterministic* OSP *algorithm is at least* ${\sigma_{\max}}^{k_{\max}-1}$*, even for unweighted unit-capacity instances.*

3 Randomized Upper Bounds

In this section we describe our algorithm for OSP and analyze it. In Section 3.2 we analyze the algorithm in the unit-capacity model. In Section 3.3 we state the extension to the general capacity model, and a few sharper results for some special cases. Due to lack of space, the proofs of results from Section 3.3 are presented only in the appendix.

3.1 The Algorithm

The algorithm we use is inspired by the Maximal Independent Set algorithm of Luby [15]. First, for any $w > 0$, we define a probability distribution R_w such that if a random variable X is distributed according to R_w, then

$$\Pr_{R_w}[X < x] = x^w \tag{2}$$

for $0 \le x \le 1$. Note that R_1 is the uniform distribution over the unit interval, and in general, R_n for a natural n, is the distribution of the maximum of n i.i.d. random variables distributed uniformly over the unit interval. Using the definition of R_w, we specify the algorithm as follows.

<u>Algorithm randPr</u>
For each set $S \in \mathcal{C}$, pick a random priority $r(S)$ according to the distribution $R_{w(S)}$.
Upon arrival of element u listing parent sets $\mathcal{C}(u)$ and capacity $b(u)$:
  Assign an arriving element u to the $b(u)$ sets with the highest priority in $\mathcal{C}(u)$.

It is important to observe that Algorithm randPr can be implemented distributively. The idea is that when an element u arrives, the sets $\mathcal{C}(u)$ that contain it can be listed explicitly (in the video transmission scenario, this means that each packet identifies the frame that contains it; in the multiple-hop route, this would just be the packet identifier). Therefore, all we need is a system-wide hash function h: applying h to the identifier of each set $S \in \mathcal{C}(u)$, we can use $h(S)$ as the random priority of S. Practically, any off-the-shelf hash function would do. And even theoretically, it suffices for the hash function to have $k_{\max} \cdot \sigma_{\max}$-wise independence, say using universal hashing.

3.2 Analysis for Unit Capacity Instances

We now prove Theorem 1. We shall use the following additional notation.

Notation 1 *Let $S \in \mathcal{C}$. Then $N[S] \stackrel{\text{def}}{=} \{S' : S' \in \mathcal{C}$ and $S \cap S' \neq \emptyset\}$, and $N(S) \stackrel{\text{def}}{=} N[S] \setminus \{S\}$.*

The key property of randPr is that the probability that a set S has the highest priority among any collection of sets is always proportional to $w(S)$, as stated in the following lemma.

Lemma 1 $\Pr[S \in \text{ALG}] = \dfrac{w(S)}{w(N[S])}$ *for every set $S \in \mathcal{C}$.*

Proof: Let $S \in \mathcal{C}$. Clearly, $\Pr[S \in \text{ALG}] = \Pr[r(S) > \max_{S' \in N(S)} r(S')]$. Denote by $r_{\max}$ the maximum priority among the priorities of sets in $N(S)$, i.e., $r_{\max} = \max_{S' \in N(S)} r(S')$. By independence of $r(S')$ for different sets in $N(S)$ we have, for $x \in [0,1]$, that

$$\Pr[r_{\max} < x] = \prod_{S' \in N(S)} \Pr[r(S') < x] = \prod_{S' \in N(S)} x^{w(S')} = x^{\sum_{S' \in N(S)} w(S')} = x^{w(N(S))} ,$$

i.e., $r_{\max}$ is distributed according to $R_{w(N(S))}$. Letting f_X denote the probability density function of random variable X, we conclude that

$$\Pr[S \in \text{ALG}] = \Pr[r(S) > r_{\max}] = \int_0^1\!\!\int_0^x f_{r(S)}(x) f_{r_{\max}}(y) dy\, dx = \int_0^1 w(S) x^{w(S)-1} \cdot x^{w(N(S))} dx = \frac{w(S)}{w(N[S])} ,$$

and we are done. $\square$

We note that Lemma 1 immediately implies that the competitive ratio of randPr is at most $k_{\max}\sigma_{\max}$ in the unweighted case, because in that case we have $w(N[S]) = |N[S]| \le k_{\max}\sigma_{\max}$ for any $S \in \mathcal{C}$, and therefore

$$\mathbb{E}[w(\text{ALG})] = \sum_{S\in\mathcal{C}} \Pr[S \in \text{ALG}] = \sum_{S\in\mathcal{C}} \frac{1}{w(N[S])} \ge \frac{n}{k_{\max}\sigma_{\max}} \ge \frac{\text{OPT}}{k_{\max}\sigma_{\max}} .$$

However, we shall prove a stronger bound. We first state the following technical lemma.

Lemma 2 *For any positive reals $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$, it holds that* $\sum_i \frac{a_i^2}{b_i} \ge \frac{(\sum_i a_i)^2}{\sum_i b_i}$.

Proof: Let $c_i = \sqrt{b_i}$ and $d_i = a_i/\sqrt{b_i}$, for $i = 1, 2, \ldots, n$. Using the Cauchy-Schwarz inequality, we obtain $(\sum_i a_i)^2 = (\sum_i c_i d_i)^2 \le \sum_i c_i^2 \sum_i d_i^2 = \sum_i \frac{a_i^2}{b_i} \sum_i b_i$, and the lemma follows. $\square$

Lemma 1 and Lemma 2 imply the following.

Lemma 3 *For any collection of sets $\mathcal{C}' \subseteq \mathcal{C}$,* $\mathbb{E}[w(\text{ALG})] \ge \frac{w(\mathcal{C}')^2}{\sum_{S_i\in\mathcal{C}'} w(N[S_i])}$.

Proof: $\mathbb{E}[w(\text{ALG})] \ge \sum_{S\in\mathcal{C}'} w(S) \cdot \frac{w(S)}{w(N[S])} \ge \frac{(\sum_{S\in\mathcal{C}'} w(S))^2}{\sum_{S\in\mathcal{C}'} w(N[S])} = \frac{w(\mathcal{C}')^2}{\sum_{S\in\mathcal{C}'} w(N[S])}$. The first inequality is by Lemma 1 and the second is by Lemma 2 with $a_i = w(S)$ and $b_i = w(N[S])$. $\square$

We now apply Lemma 3 to two collections of sets. First, to the sets in an optimal solution.

Lemma 4 $\mathbb{E}[w(\text{ALG})] \ge \frac{w(\text{OPT})^2}{k_{\max} \cdot w(\mathcal{C})}$.

Proof: By Lemma 3 with $\mathcal{C}' := \text{OPT}$, we have that

$$\mathbb{E}[w(\text{ALG})] \ge \frac{w(\text{OPT})^2}{\sum_{S\in\text{OPT}} w(N[S])} . \tag{3}$$

Now, observe that since the sets in OPT are disjoint, each set $S \in \mathcal{C}$ intersects at most $|S| \le k_{\max}$ sets in OPT, and hence $\sum_{S\in\text{OPT}} w(N[S]) \le k_{\max} \cdot w(\mathcal{C})$. Plugging this observation into Eq. (3) we obtain the lemma. $\square$

Next, we apply Lemma 3 with the collection of all sets in the instance.

Lemma 5 $\mathbb{E}[w(\text{ALG})] \ge \frac{w(\mathcal{C})^2}{n \cdot \overline{\sigma \cdot \sigma_\$}}$.

Proof: By Lemma 3 we have that $\mathbb{E}[w(\text{ALG})] \ge \frac{w(\mathcal{C})^2}{\sum_{S\in\mathcal{C}} w(N[S])}$. By summing over elements we obtain

$$\sum_{S\in\mathcal{C}} w(N[S]) \le \sum_{S\in\mathcal{C}} \sum_{u\in S} w(\mathcal{C}(u)) = \sum_{u\in U} \sigma(u) \cdot w(\mathcal{C}(u)) = n \cdot \overline{\sigma \cdot \sigma_\$} ,$$

and therefore $\mathbb{E}(w(\text{ALG})] \ge \frac{w(\mathcal{C})^2}{n \cdot \overline{\sigma \cdot \sigma_\$}}$ as required. $\square$

Proof of Theorem 1: Lemma 4 and Lemma 5 give us two lower bounds on $w(\textsc{alg})$. The maximum of these bounds is minimized when $w(\textsc{opt}) = \sqrt{\frac{w(\mathcal{C})^3 \cdot k_{\max}}{n \cdot \overline{\sigma \cdot \sigma_\$}}}$, and therefore, for any instance

$$\mathbb{E}[w(\textsc{alg})] \geq w(\textsc{opt}) \cdot \sqrt{\frac{w(\mathcal{C})}{k_{\max} n \cdot \overline{\sigma \cdot \sigma_\$}}} \ .$$

Finally, since

$$n \cdot \overline{\sigma_\$} = \sum_{u \in U} \sigma_\$(u) = \sum_{S \in \mathcal{C}} |S| \cdot w(S) \leq k_{\max} \cdot w(\mathcal{C}) \ , \tag{4}$$

it follows that

$$\mathbb{E}[w(\textsc{alg})] \geq w(\textsc{opt}) \cdot \sqrt{\frac{\overline{\sigma_\$}}{k_{\max}^2 \overline{\sigma \cdot \sigma_\$}}} = w(\textsc{opt}) \cdot \frac{1}{k_{\max}} \sqrt{\frac{\overline{\sigma_\$}}{\overline{\sigma \cdot \sigma_\$}}} \ ,$$

and we are done. □

Corollary 6 *For any instance,* $\mathbb{E}[w(\textsc{alg})] \geq w(\textsc{opt})/(k_{\max}\sqrt{\sigma_{\max}})$*.*

Proof: Follows from the fact that $\sqrt{\frac{\overline{\sigma_\$}}{\overline{\sigma \cdot \sigma_\$}}} = \sqrt{\frac{\sum_u \sigma_\$(u)/n}{\sum_u \sigma(u)\sigma_\$(u)/n}} \geq \sqrt{\frac{\sum_u \sigma_\$(u)}{\sigma_{\max} \sum_u \sigma_\$(u)}} = \frac{1}{\sqrt{\sigma_{\max}}}$. □

3.3 A Generalization and A Few Specializations

In this section we state results for one generalized case and a few special cases. Due to lack of space, no proofs are given here (they are presented in the appendix). We emphasize, however, that some new non-trivial ideas are needed to prove the results.

Our generalization concerns servers with variable capacities: the case where each element u may be assigned to $b(u)$ sets. In this case, we generalize the notion of load of an element as follows.

Definition 1 *The* adjusted load *of an element u is* $\nu(u) \stackrel{\text{def}}{=} \sigma(u)/b(u)$*.*

Intuitively, the adjusted load of an element u is the ratio of demand to supply: u is needed by $\sigma(u)$ sets, but only $b(u)$ sets can be provided for.

Using additional arguments on top of the arguments used in the unit-capacity case, we show the following.

Theorem 4 *The competitive ratio of* $\textsc{randPr}$ *is at most* $16e \cdot k_{\max}\sqrt{\frac{\overline{\nu \cdot \sigma_\$}}{\overline{\sigma_\$}}}$*.*

Next, we state a few sharper bounds for the unweighted case. Specifically, we have the following results.

Theorem 5 *If all sets have the same size k, then* $\mathbb{E}[|\textsc{alg}|] \geq |\textsc{opt}| \cdot \overline{\sigma}^2/(k \cdot \overline{\sigma^2})$*.*

The following corollary of Theorem 5 is our only upper bound that is independent of σ.

Corollary 7 *If all sets have the same size k and all elements have the same load, then* $\mathbb{E}[|\textsc{alg}|] \geq \frac{|\textsc{opt}|}{k}$*.*

Theorem 6 *If all elements have the same load σ then,* $\mathbb{E}[|\textsc{alg}|] \geq \frac{|\textsc{opt}|}{k \cdot \sqrt{\sigma}}$*.*

4 Lower Bounds

In this section we prove lower bounds on the competitive ratio of any online algorithm for OSP. The bad examples in our online lower bounds are all unweighted, and further, all sets have a common size k and all elements have unit capacity. In view of Corollary 7, however, element loads necessarily vary. The deterministic lower bound is rather simple; the randomized lower bound is more involved, and uses a construction based on combinatorial designs.

4.1 Deterministic Online Algorithms

In this section we establish a lower bound on the competitive ratio of any deterministic OSP algorithm.

Proof of Theorem 3: Fix a deterministic OSP algorithm. We construct an unweighted OSP instance $\mathcal{C}$ containing σ^k sets, each of size exactly k. The construction ensures that $|\text{ALG}| \leq 1$ while $|\text{OPT}| \geq \sigma^{k-1}$.

We describe the construction by building the sets incrementally, as a function of the algorithm at hand. Call a set *active* at a given time if the algorithm assigned to it all its elements up to that point. Initially, all σ^k sets are active. After each phase $i = 1, \ldots, k$ there will be at most σ^{k-i} active sets. This is ensured by partitioning the sets that are active before phase i into σ^{k-i} collections of σ sets each; for each such collection of σ sets we introduce a new element, which is a member of these σ sets. Clearly, at most one set from each collection remains active when the phase ends, and therefore $|\text{ALG}| \leq 1$ after k phases.

Note that at this point most sets have less than k elements defined. We now introduce new elements to complete all sets to size k. All these elements have load one (i.e., each belongs to a single set).

Observe that in an optimal solution, it is possible to complete σ^{k-1} sets by assigning the first phase elements to sets that were not chosen by the algorithm. These sets survive, since they do not participate in the following $k-1$ phases. The theorem follows. $\square$

4.2 Randomized Algorithms

We now turn to the main technical contribution of this section: developing lower bounds for the competitive ratio of randomized OSP algorithms and establishing Theorem:2. At the outset, we use Yao's principle: we build a distribution of the inputs for which the expected value for all deterministic algorithms is small.

We first give an intuitive explanation of a weaker lower bound. The input consists of a collection of t^2 sets denoted S_{ij}, $i, j \in \{1, \ldots, t\}$. All elements will be contained in exactly t sets. The adversary first presents to the algorithm t elements $u_1, \ldots, u_t$ such that $u_i \in S_{ij}$, for every j. Then, it presents t^2 random elements $v_1, \ldots, v_{t^2}$ that satisfy the following condition: if $v_\ell \in S_{ij}, S_{i'j'}$, then $i \neq i'$ and $j \neq j'$. This completes the construction. Let X be the solution computed by a given online algorithm after presented with $u_1, \ldots, u_t$. Any two sets from X contain some common element v_ℓ with constant probability. By standard arguments then, only $O(\log t)$ sets from X survive after the presentation of $v_1, \ldots, v_{t^2}$. On the other hand, an optimal solution may complete $S_{1j}, \ldots, S_{tj}$ for some j. Hence, the performance ratio of the algorithm is $\Omega(t/\log t)$. Observe that $\sigma = t$, $\overline{k} = t$ and $k_{\max} = \tilde{O}(t)$ (with high probability). Thus, we obtain $\Omega(\sigma/\log\sigma)$ and $\Omega(\overline{k}/\log\overline{k})$ lower bounds for OSP.

The stronger lower bound that we claim is achieved using similar ideas, but in several steps. The presentation of our construction is divided into two parts. In the first we describe a gadget that is based on combinatorial designs. Afterwards we use this gadget to obtain the required distribution.

4.2.1 The Gadget

Our basic building block in the construction is a combinatorial object we call (M,N)-*gadget*, reminiscent of affine planes. An (M,N)-gadget defines a way in which $M\cdot N$ sets intersect: in a full gadget, any two sets intersect; we use slightly defective gadgets, in which relatively many sets (that will be chosen by an optimal solution) do not intersect, but any deterministic algorithm will have only few complete sets in expectation.

Specifically, let N be a prime power and $M\le N$ be a positive integer. Let $\mathcal{F}$ be a finite field of cardinality $|\mathcal{F}|=N$ and let $\mathcal{F}_M\subseteq\mathcal{F}$ be some subset of $\mathcal{F}$ of cardinality $|\mathcal{F}_M|=M$. An (M,N)-*gadget* is a combinatorial structure that consists of $M\cdot N$ *items* identified with pairs in $\mathcal{F}_M\times\mathcal{F}$. A *line* in this gadget is a subset of the items. Specifically, for every $a,b\in\mathcal{F}$ we define the line $L_{a,b}=\{(i,j)\in\mathcal{F}_M\times\mathcal{F}: j=ai+b\}$, where the arithmetic is performed over $\mathcal{F}$, and for every $c\in\mathcal{F}_M$, we define the line $L_{\infty,c}=\{c\}\times\mathcal{F}$. It is easy to verify that the (M,N)-gadget satisfies the following properties.

Proposition 1 *Let $(i,j),(i',j')\in\mathcal{F}_M\times\mathcal{F}$. If $i\ne i'$, then there exists exactly one line $L\in\{L_{a,b}: a,b\in\mathcal{F}\}$ such that $(i,j),(i',j')\in L$. If $i=i'$ and $j\ne j'$, then there exists exactly one line $L\in\{L_{\infty,c}: c\in\mathcal{F}_M\}$ such that $(i,j),(i',j')\in L$.*

Proposition 2 *Let $(i,j)\in\mathcal{F}_M\times\mathcal{F}$. For every $a\in\mathcal{F}$, there exists exactly one $b\in\mathcal{F}$ such that $(i,j)\in L_{a,b}$. Also, there exists exactly one $c\in\mathcal{F}_M$ such that $(i,j)\in L_{\infty,c}$.*

Subsequently, it will be convenient to identify the elements of $\mathcal{F}$ with the integers in $[N]$ and the elements of $\mathcal{F}_M$ with the integers in $[M]$. Resolving ambiguities will be clear from the context, especially when arithmetic operations are involved. In the context of OSP, the items of an (M,N)-gadget will represent sets, and its lines will represent elements. We will usually describe how the sets are mapped to the items of the (M,N)-gadget and identify each set with the corresponding pair $(i,j)\in[M]\times[N]$, and each element with some line in $\{L_{a,b}: a,b\in[N]\}\cup\{L_{\infty,c}: c\in[M]\}$. A set is said to belong to *row* $i\in[M]$ under the (M,N)-gadget if it is mapped to item (i,j) for some $j\in[N]$.

Let $\mathcal{C}'$ denote the sets corresponding to a given gadget. Proposition 1 guarantees that every two sets in $\mathcal{C}'$ that do not belong to the same row intersect in exactly one element $L_{a,b}$, $a,b\in[N]$. It also guarantees that every two sets in $\mathcal{C}'$ that do belong to the same row intersect in exactly one element $L_{\infty,c}$, $c\in[M]$. Moreover, by Proposition 2, for every $a\in[N]$, each set in $\mathcal{C}'$ contains exactly one element in $\{L_{a,b}: b\in[N]\}$ and exactly one element in $\{L_{\infty,c}: c\in[M]\}$. Therefore, throughout the arrival of all elements in $\{L_{a,b}: a,b\in[N]\}\cup\{L_{\infty,c}: c\in[M]\}$, each set in $\mathcal{C}'$ appears exactly $N+1$ times and any two sets intersect exactly once.

Consider a collection $\mathcal{C}'$ of $M\cdot N$ sets and a bijection $\mu:\mathcal{C}'\to[M]\times[N]$. The bijection μ can be viewed as placing the $M\cdot N$ sets in an $M\times N$ matrix. By saying that line L is *applied* to $\mathcal{C}'$ under μ,

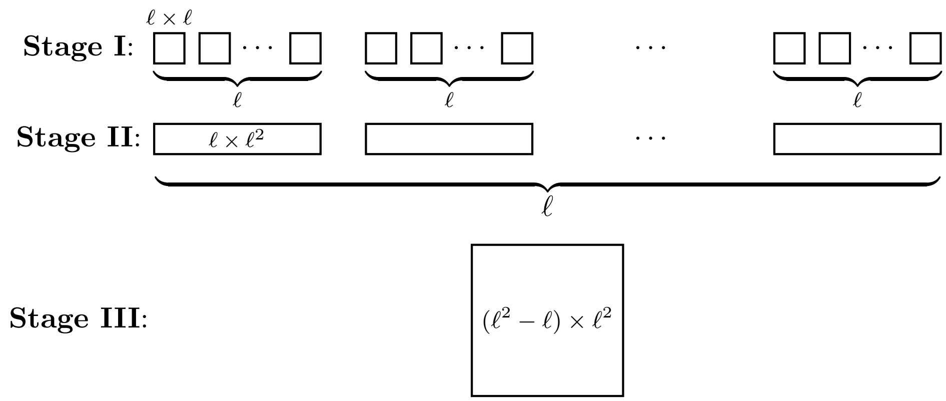

Figure 1: Stages I, II and III.

we refer to the arrival of an element which is contained in every set $S \in \mathcal{C}'$ such that $\mu(S) \in L$. By saying that an (M, N)-gadget is *applied* to $\mathcal{C}'$ under μ, we refer to the following procedure:

(i) For $a = 1, \dots, N$, for $b = 1, \dots, N$, apply the line $L_{a,b}$ to $\mathcal{C}'$ under μ.

(ii) For $c = 1, \dots, M$, apply the line $L_{\infty,c}$ to $\mathcal{C}'$ under μ.

If step (ii) is omitted, then we say that an (M, N)-gadget is *applied* to $\mathcal{C}'$ under μ *without the rows.*

We can now summarize the properties of an (M, N)-gadget we use later.

Lemma 8 *An application of an (M, N)-gadget to the set collection $\mathcal{C}'$ under the bijection μ consists of N^2 elements of load M and M elements of load N; each set in $\mathcal{C}'$ contains exactly $N+1$ elements. Upon completion of the application, $|\mathcal{A} \cap \mathcal{C}'| \leq 1$ in any feasible solution $\mathcal{A}$. If the application is without the rows, then it consists of N^2 elements of load M; each set in $\mathcal{C}'$ contains exactly N elements. Upon completion of the application, all sets in $|\mathcal{A} \cap \mathcal{C}'|$ belong to the same row (with respect to μ) for any feasible solution $\mathcal{A}$.*

4.2.2 The Distribution

We are now ready to establish the main lemma of this section.

Lemma 9 *Let ℓ be a positive power of some prime. There exists a collection $\mathcal{J}$ of unweighted* OSP *instances with ℓ^4 sets and a probability distribution $\mathcal{D}$ over $\mathcal{J}$ such that (1) all sets are of size $k = \Theta(\ell^2)$; (2) $\sigma_{\max} = \Theta(\ell^2)$, $\overline{\sigma} = \Theta(\ell)$, and $\overline{\sigma^2} = \Theta(\ell^3)$; (3) $\text{OPT}(J) \geq \ell^3$ for every instance $J \in \mathcal{J}$; and (4) $\mathbb{E}_{\mathcal{D}}[\text{ALG}(J)] = O((\frac{\log \ell}{\log \log \ell})^2)$ for every deterministic* OSP *algorithm* ALG.

Proof: Let $\mathcal{C}$ be a collection of ℓ^4 sets. We construct a random instance $J \in_{\mathcal{D}} \mathcal{J}$ in four *stages* as follows (see depiction in Figure 1).

Stage I. The collection $\mathcal{C}$ is partitioned (arbitrarily) into ℓ^2 subcollections $\mathcal{C}_1^I, \dots, \mathcal{C}_{\ell^2}^I$, each containing ℓ^2 sets. For $t = 1, \dots, \ell^2$, we choose a bijection $\mu_t^I : \mathcal{C}_t^I \to [\ell] \times [\ell]$ uniformly at random. We then apply an (ℓ, ℓ)-gadget to $\mathcal{C}_t^I$ under μ_t^I without the rows.

Stage II. The collection $\mathcal{C}$ is partitioned into ℓ subcollections $\mathcal{C}_1^{II}, \ldots, \mathcal{C}_\ell^{II}$, where

$$\mathcal{C}_t^{II} = \left\{S : S \in \mathcal{C}_z^I \text{ for } \ell+1 \leq z \leq t\ell\right\} ,$$

for every $1 \leq t \leq \ell$. Let $\pi_z : [\ell] \to [\ell]$ be a permutation chosen uniformly at random for every $1 \leq z \leq \ell^2$. For $t = 1, \ldots, \ell$, we construct the bijection $\mu_t^{II} : \mathcal{D}_t^{II} \to [\ell] \times [\ell^2]$ by mapping each set $S \in \mathcal{C}_z^I$, $(t-1)\ell + 1 \leq z \leq t\ell$, to the pair $(\pi_z(i), j + (\ell-1)(z-(t-1)\ell))$ such that $\mu_z^I(S) = (i,j)$. In other words, the $\ell \times \ell^2$ matrix induced by μ_t^{II} is obtained from concatenating the $\ell \times \ell$ matrices induced by $\mu_{(t-1)\ell+1}^I, \ldots, \mu_{t\ell}^I$ after their rows were randomly permuted. We then apply an (ℓ, ℓ^2)-gadget to $\mathcal{C}_t^{II}$ under μ_t^{II} without the rows.

Stage III. Choose $u_t \in [\ell]$ uniformly at random for every $t \in [\ell]$ and let $\mathcal{S}$ be the set subcollection consisting of the sets in row u_t for all $t \in [\ell]$, namely,

$$\mathcal{S} = \bigcup_{1 \leq t \leq \ell} \left\{S \in \mathcal{C}_t^{II} : \mu_t^{II}(S) \in \{u_t\} \times [\ell^2]\right\} .$$

Let $\mu^{III} : \mathcal{C} \setminus \mathcal{S} \to [\ell^2 - \ell] \times [\ell^2]$ be an arbitrary bijection. We apply an $(\ell^2 - \ell, \ell^2)$-gadget to $\mathcal{C} \setminus \mathcal{S}$ under μ^{III}.

Stage IV. The remaining ℓ^2 elements of each set in $\mathcal{S}$ have load 1 and they arrive in arbitrary order.

We now turn to analyzing the construction of the random instance J. Lemma 8 guarantees that each set appears (i.e., have elements arriving) ℓ times in Stage I; each set appears ℓ^2 times in Stage II; the sets in $\mathcal{C} \setminus \mathcal{S}$ appear ℓ^2 times in step III; and the sets in $\mathcal{S}$ appear ℓ^2 times in step IV. Therefore $k = \Theta(\ell^2)$. From the perspective of the elements, we have the following situation. In Stage I there are ℓ^4 elements, each with load ℓ (i.e., each element is contained in ℓ sets); there are ℓ^5 elements in Stage II, with load ℓ each; Stage III consists of $\Theta(\ell^4)$ elements with load $\Theta(\ell^2)$ each; and Stage IV consists of ℓ^5 elements of load 1 each. Therefore $\sigma_{\max} = \Theta(\ell^2)$, $\overline{\sigma} = \Theta(\ell)$, and $\overline{\sigma^2} = \Theta(\ell^3)$.

Next, we show that $\text{OPT}(J) \geq \ell^3$ for every instance $J \in \mathcal{J}$. This is done by proving that an optimal algorithm can complete all sets in $\mathcal{S}$. Indeed, every two sets $S, S' \in \mathcal{S}$ either belong to disjoint subcollections in Stages I and II or they belong to the same row in some subcollection. In either cases, S and S' do not intersect. Therefore all sets in $\mathcal{S}$ can be kept OPT-active throughout the execution.

It remains to prove that $\mathbb{E}_{\mathcal{D}}[\text{ALG}(J)] = O((\frac{\log \ell}{\log \log \ell})^2)$ for every deterministic OSP algorithm ALG. Consider some bijection μ_t^I in Stage I and recall that this bijection partitions the sets in $\mathcal{C}_t^I$ to ℓ rows with ℓ sets in each row. By definition, the first ℓ elements in the corresponding (ℓ, ℓ)-gadget are the lines $L_{1,1}, \ldots, L_{1,\ell}$. After the arrival of these ℓ elements, there is at most one ALG-active set in each such line. The random choice of the bijection μ_t^I implies that this ALG-active set "falls" into some designated row with probability $1/\ell$, hence with high probability, every row in the gadget contains $O(\frac{\log \ell}{\log \log \ell})$ ALG-active sets. This is implied by the classical *balls into bins* result [13, 17]. When the application of the gadget is terminated, there exists at most one row with ALG-active sets (this holds with probability 1). Let A denote the event that upon termination of Stage I, each one of the subcollections $\mathcal{C}_1^I, \ldots, \mathcal{C}_{\ell^2}^I$ admits $O(\frac{\log \ell}{\log \log \ell})$ ALG-active sets, all of which belong to the same row in the corresponding bijection. By union bound, we conclude that event A occurs with high probability.

Consider some bijection μ_t^{II} in Stage II and recall that this bijection partitions the sets in $\mathcal{C}_t^{II}$ to ℓ rows with ℓ^2 sets in each row. The random choice of the permutations π_z can be interpreted as assigning each row of some Stage I bijection to one of the rows of μ_t^{II} with equal probability. In particular, a row with ALG-active sets in some Stage I bijection "falls" into some designated row in μ_t^{II} with probability $1/\ell$, hence conditioned on event A, with high probability, every row in μ_t^{II} contains $O((\frac{\log \ell}{\log \log \ell})^2)$ ALG-active sets. When the application of the corresponding (ℓ, ℓ^2)-gadget terminates, at most one of these rows still contain ALG-active sets (this holds with probability 1). Let B denote the event that upon termination of Stage II, each one of the subcollections $\mathcal{C}_1^{II}, \ldots, \mathcal{C}_\ell^{II}$ admits $O((\frac{\log \ell}{\log \log \ell})^2)$ ALG-active sets, all of which belong to the same row in the corresponding bijection. By union bound, we conclude that event B occurs with high probability.

Let X denote the random variable that counts the number of sets in $\mathcal{S}$ that were ALG-active upon termination of Stage II. The random choices of $u_1, \ldots, u_\ell$ in Stage III (which determine the subcollection $\mathcal{S}$) imply that $\mathbb{E}[X \mid B] = O((\frac{\log \ell}{\log \log \ell})^2)$. On the other hand, the application of the $(\ell^2 - \ell, \ell^2)$-gadget in Stage III guarantees that at most one set in $\mathcal{C} \setminus \mathcal{S}$ can be completed by ALG. It follows that $\mathbb{E}_{\mathcal{D}}[\text{ALG}(J)] \leq O((\frac{\log \ell}{\log \log \ell})^2) + 1$. The assertion follows. $\square$

Proof of Theorem 2: By Lemma 9, for any given number m there exists a collection of instances of OSP, each with $\Theta(m)$ sets of size $k = \Theta(\sqrt{m})$ each, with average element load $\overline{\sigma} = \Theta(m^{1/4})$ and $\overline{\sigma^2} = \Theta(m^{3/4})$, such that the optimal value obtained from each of these instances is $\Omega(m^{3/4})$, and the expected benefit of any deterministic algorithm from a random instance in the collection is $O((\frac{\log m}{\log \log m})^2)$. It follows from Yao's Lemma [19] that the competitive ratio of any randomized algorithm is $\Omega(m^{3/4}(\frac{\log \log m}{\log m})^2)$ which is both $\Omega\left(k\sqrt{\sigma_{\max}} \cdot (\frac{\log \log k}{\log k})^2\right)$ and $\Omega\left(k\sqrt{\frac{\overline{\sigma^2}}{\overline{\sigma}}}(\frac{\log \log k}{\log k})^2\right)$. $\square$

5 Conclusions and Open Problems

We have introduced the online set packing problem and presented a competitive algorithm that solves it. Many questions remain open in this area. We mention a few major ones.

- It seems interesting to generalize the problem to arbitrary packing problems, where the entries in the matrix are arbitrary non-negative integers.
- Recalling the networking motivation, it is interesting to understand the effect of buffers on the problem.
- A set is gained in OSP only if all its elements were assigned to it. What about the case where the set can be gained even if a few elements are missing?

Acknowledgments. We thank Geir Agnarsson and Bjarni V. Halldórsson for independently proving Lemma 2.

References

[1] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor. The online set cover problem. In *Proc. 35th Ann. ACM Symp. on Theory of Computing*, pages 100–105. ACM, 2003.

[2] N. Alon, U. Arad, and Y. Azar. Independent sets in hypergraphs with applications to routing via fixed paths. In *Proc. 2nd APPROX*, pp. 16–27, 1999.

[3] B. Awerbuch and R. Khandekar. Stateless distributed gradient descent for positive linear programs. *SIAM J. Comput.*, 38(6):2468–2486, 2009.

[4] P. Berman. A $d/2$-approximation for maximum weight independent set in d-claw free graphs. *Nordic J. Comput.*, 7(3):178–184, 2000.

[5] N. Buchbinder and J. Naor. Online primal-dual algorithms for covering and packing. *Math. Oper. Res.*, 34(2):270–286, 2009.

[6] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182(1):105–142, 1999.

[7] M. M. Halldórsson, K. Iwama, S. Miyazaki, and S. Taketomi. Online independent sets. *Theoretical Computer Science*, 289(2):953–962, 2002.

[8] M. M. Halldórsson, J. Kratochvil, J. A. Telle. Independent sets with domination constraints. *Disc. Applied Math.*, 99(1-3):39–54, 2000.

[9] E. Hazan, S. Safra, and O. Schwartz. On the complexity of approximating k-dimensional matching. In *Proc. 6th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 83–97, 2003.

[10] C. A. J. Hurkens and A. Schrijver. 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. Disc. Math.*, 2(1):68–72, 1989.

[11] R. M. Karp, U. V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proc. 22nd Ann. ACM Symp. on Theory of Computing*, pages pages 352–358. ACM, 1990.

[12] A. Kesselman, B. Patt-Shamir, and G. Scalosub. Competitive buffer management with packet dependencies. In *23rd IEEE Int. Symp. on Parallel and Distributed Processing (IPDPS) 2009, Rome, Italy, May 23-29, 2009*, pages 1–12. IEEE, 2009.

[13] V. F. Kolchin, B. A. Sevastyanov, and V. P. Chistyakov. *Random Allocations*. Winston & Sons, Washington, 1978.

[14] F. Kuhn, T. Moscibroda, and R. Wattenhofer. The price of being near-sighted. In *Proc. 17th Ann. ACM-SIAM Symposium on Discrete Algorithms*, pages 980–989, New York, NY, USA, 2006. ACM.

[15] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM J. Comput.*, 15(4):1036–1053, Nov. 1986.

[16] C. H. Papadimitriou and M. Yannakakis. Linear programming without the matrix. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 121–129, New York, NY, USA, 1993. ACM.

[17] M. Raab and A. Steger. "Balls into Bins" - A Simple and Tight Analysis. In *Proc. 2nd Intl. Workshop on Randomization and Approximation Techniques in Computer Science*, pages 159–170, 1998.

[18] T. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135(1-2):1–54, 2002.

[19] A. C.-C. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 222–227, 1977.

APPENDIX: Additional Proofs

Proofs from Section 3.3: The Variable capacity case

Given a set S and an element $u \in S$, we say that S *survives* u if the priority chosen for S in randPr is one of the $b(u)$ highest among the sets competing for u. S is said to survive if S survives u for all $u \in S$.

We start the proof by showing that the worst scenario, from the viewpoint of a set S_0, is when S_0 never meets another set more than once, namely $\mathcal{C}(u) \cap \mathcal{C}(v) = \{S_0\}$ for every $u, v \in S_0$ such that $u \neq v$.

Let $S_0, S_1 \in \mathcal{C}$ be two sets such that $u, v \in S_0 \cap S_1$. Define the following new collection of sets $\mathcal{C}' = \mathcal{C} \setminus \{S_1\} \cup \{S'_1, S'_2\}$ where $S'_1 = S_1 \setminus \{v\}$ and $S'_2 = \{v\}$. We also define $w'(S) = w(S)$ for every $S \in \mathcal{C}' \setminus \{S'_1, S'_2\}$ and $w'(S'_1) = w'(S'_2) = w(S_1)$. Notice that we simply divided S_1 into two sets both having the same weight as S_1. Also notice that $w'(\mathcal{C}'(x)) = w(\mathcal{C}(x))$ for every element x.

Lemma 10 $\Pr[S_0 \in \text{ALG}(\mathcal{C})] \geq \Pr[S_0 \in \text{ALG}(\mathcal{C}')]$.

Proof: Suppose that we first choose the priorities of the sets in $\mathcal{C} \setminus \{S_1\}$. Given these random choices we define the following events: $A_<$ occurs if there are less than $b(u) - 1$ sets in $\mathcal{C}(u) \setminus \{S_1\}$ whose priority is larger than $r(S_0)$; $A_=$ occurs if there are exactly $b(u) - 1$ sets in $\mathcal{C}(u) \setminus \{S_1\}$ whose priority is larger than $r(S_0)$; and $A_>$ occurs if there are at least $b(u)$ sets in $\mathcal{C}(u) \setminus \{S_1\}$ whose priority is larger than $r(S_0)$. We define $B_<$, $B_=$ and $B_>$ similarly with respect to v.

To prove the lemma we show that

$$\Pr[S_0 \in \text{ALG}(\mathcal{C}) \mid A_i \cap B_j] \geq \Pr[S_0 \in \text{ALG}(\mathcal{C}') \mid A_i \cap B_j] \ ,$$

for every $i, j \in \{<, =, >\}$.

There are several cases. If $A_>$ or $B_>$ occur, then S_0 survives with respect to neither $\mathcal{C}$ nor $\mathcal{C}'$ regardless of the random choices involving S_1, S'_1 and S'_2. Namely

$$\Pr[S_0 \in \text{ALG}(\mathcal{C}) \mid A_> \cap B_j] = \Pr[S_0 \in \text{ALG}(\mathcal{C}') \mid A_> \cap B_j] = 0 \ ,$$

for every $j \in \{<, =, >\}$, and

$$\Pr[S_0 \in \text{ALG}(\mathcal{C}) \mid A_i \cap B_>] = \Pr[S_0 \in \text{ALG}(\mathcal{C}') \mid A_i \cap B_>] = 0 \ ,$$

for every $i \in \{<, =, >\}$. If both $A_<$ and $B_<$ occur, then S_0 survives u and v with respect to both $\mathcal{C}$ and $\mathcal{C}'$ regardless of the random choices of S_1, S'_1 and S'_2, i.e.

$$\Pr[S_0 \in \text{ALG}(\mathcal{C}) \mid A_< \cap B_<] = \Pr[S_0 \in \text{ALG}(\mathcal{C}') \mid A_< \cap B_<] = 1 \ .$$

If $A_<$ and $B_=$ occur, then

$$\Pr[S_0 \in \text{ALG}(\mathcal{C}) \mid A_< \cap B_=] = \Pr[r(S_1) < r(S_0)] = \Pr[r(S'_1) < r(S_0)] = \Pr[S_0 \in \text{ALG}(\mathcal{C}') \mid A_< \cap B_=] \ .$$

Similarly, If $A_=$ and $B_<$ occur, then $\Pr[S_0 \in \textsc{alg}(\mathcal{C}) \mid A_= \cap B_<] = \Pr[S_0 \in \textsc{alg}(\mathcal{C}') \mid A_= \cap B_<]$. Last, if both $A_=$ and $B_=$ occur, then we have

$$\begin{aligned}\Pr[S_0 \in \textsc{alg}(\mathcal{C}) \mid A_= \cap B_=] &= \Pr[r(S_1) < r(S_0)] \\ &> \Pr[r(S_1'), r(S_2') < r(S_0)] \\ &= \Pr[S_0 \in \textsc{alg}(\mathcal{C}') \mid A_= \cap B_=] ,\end{aligned}$$

as required. $\square$

Below, we bound a probability of survival of a set S under the condition that all weights are integral. Define $Q_S = \sum_{u \in S} w(\mathcal{C}(u))/b(u)$ and $q_S = 1 - 1/(2Q_S)$. Notice that since all weights are integral, we have $Q_S \geq \sum_{u \in S} |\mathcal{C}(u)|/b(u) \geq |S|$ and $q_S \geq \frac{1}{2}$.

Lemma 11 $\Pr[r(S) \geq q_S] \geq \min\left\{w(S)/(4Q_S), 1 - e^{-1}\right\}$.

Proof: Recall that $\Pr[r(S) \geq q_S] = 1 - q_S^{w(S)}$ (see Eq. (2)). There are two possibilities. First, if $q_S \leq 1 - 1/w(S)$, then

$$\Pr[r(S) \geq q_S] \geq 1 - (1 - 1/w(S))^{w(S)} \geq 1 - e^{-1} .$$

On the other hand, if $q_S > 1 - 1/w(S)$, then since $(1-x)^y \leq e^{-xy} \leq 1 - xy + \frac{1}{2}x^2y^2$ if $0 < x < 1 < y$ and $xy < 1$, we have

$$\Pr[r(S) \geq q_S] = 1 - (1 - (1 - q_S))^{w(S)} \geq 1 - e^{-(1-q_S)w_S} \geq \frac{(1 - q_S)w(S)}{2} = \frac{w(S)}{4Q_S} ,$$

and the lemma follows. $\square$

The following analog of Lemma 1 contains the key idea in the extension to the general capacity case.

Lemma 12 *For every set $S \in \mathcal{C}$ we have* $\Pr[S \in \textsc{alg}] \geq \min\left\{w(S)/(4Q_s), 1 - e^{-1}\right\} \cdot e^{-1}$.

Lemma 12 induces the following partition of $\mathcal{C}$. Denote $\mathcal{C}' = \{S \in \mathcal{C} : \Pr[S \in \textsc{alg}] \geq (1 - e^{-1})e^{-1}\}$ and $\mathcal{C}'' = \mathcal{C} \setminus \mathcal{C}'$. Also, denote $\textsc{opt}' = \textsc{opt} \cap \mathcal{C}'$ and $\textsc{opt}'' = \textsc{opt} \cap \mathcal{C}''$, where $\textsc{opt}$ denotes an optimal solution. We have the following result for the first easy cases.

Lemma 13 *If either (i) $w(\textsc{opt}') \geq w(\textsc{opt})/2$ or (ii) $w(\mathcal{C}') \geq w(\mathcal{C})/2$, then* $\mathbb{E}[w(\textsc{alg})] \geq \frac{1}{2}e^{-1}(1 - e^{-1})w(\textsc{opt})$.

Proof: If $w(\textsc{opt}') \geq w(\textsc{opt})/2$, then

$$w(\textsc{alg}) \geq \sum_{S_i \in \textsc{opt}'} e^{-1}(1 - e^{-1})w(S_i) = e^{-1}(1 - e^{-1})w(\textsc{opt}') \geq \frac{1}{2}e^{-1}(1 - e^{-1})w(\textsc{opt}) .$$

If $w(\mathcal{C}') \geq w(\mathcal{C})/2$, then

$$w(\textsc{alg}) \geq \sum_{S_i \in C'} e^{-1}(1 - e^{-1})w(S_i) = e^{-1}(1 - e^{-1})w(C') \geq \frac{1}{2}e^{-1}(1 - e^{-1})w(C) \geq \frac{1}{2}e^{-1}(1 - e^{-1})w(\textsc{opt}) .$$

$\square$

The next two lemmas analyze the case where (i) $w(\textsc{opt}') < w(\textsc{opt})/2$, and (ii) $w(\mathcal{C}') < w(\mathcal{C})/2$. They are analogs of Lemmas 4 and 5.

Lemma 14 $\mathbb{E}[w(\textsc{alg})] \geq \dfrac{w(\textsc{opt})^2}{16ek_{\max} \cdot w(\mathcal{C})}$.

Proof: Summing over the disjoint sets in the optimal solution we get

$$\mathbb{E}[w(\textsc{alg})] \geq \sum_{S_i \in \textsc{opt}''} w(S_i) \cdot e^{-1} \frac{w(S_i)}{4Q_{S_i}} = \frac{1}{4e} \sum_{S_i \in \textsc{opt}''} \frac{w(S_i)^2}{Q_{S_i}} .$$

Using Lemma 2 with $a_i = w(S_i)$ and $b_i = Q_{S_i}$, we have that

$$\mathbb{E}[w(\textsc{alg})] \geq \frac{1}{4e} \cdot \frac{(\sum_{S_i \in \textsc{opt}''} w(S_i))^2}{\sum_{S_i \in \textsc{opt}''} Q_{S_i}} \geq \frac{1}{4e} \cdot \frac{w(\textsc{opt}'')^2}{\sum_{S_i \in \textsc{opt}''} Q_{S_i}} \geq \frac{1}{16e} \cdot \frac{w(\textsc{opt})^2}{\sum_{S_i \in \textsc{opt}''} Q_{S_i}} .$$

Since each element u is assigned to at most $b(u)$ sets from $\textsc{opt}''$, we have

$$\sum_{S_i \in \textsc{opt}''} Q_{S_i} = \sum_{S_i \in \textsc{opt}''} \sum_{u \in S_i} \frac{w(\mathcal{C}(u))}{b(u)} \leq \sum_{u \in U} b(u) \frac{w(\mathcal{C}(u))}{b(u)} = \sum_{u \in U} w(\mathcal{C}(u)) \leq k_{\max} \cdot w(\mathcal{C})$$

and the lemma is proven. □

Lemma 15 $\mathbb{E}[w(\textsc{alg})] \geq \dfrac{w(\mathcal{C})^2}{16en \cdot \overline{\nu\sigma_\$}}$.

Proof: Using Lemmas 2 and 12 we get that

$$\mathbb{E}[w(\textsc{alg})] \geq \frac{1}{4e} \sum_{S_i \in \mathcal{C}''} \frac{w(S_i)^2}{Q_{S_i}} \geq \frac{1}{4e} \cdot \frac{w(\mathcal{C}'')^2}{\sum_{S_i \in \mathcal{C}''} Q_{S_i}} \geq \frac{1}{16e} \cdot \frac{w(\mathcal{C})^2}{\sum_{S_i \in \mathcal{C}''} Q_{S_i}} .$$

Since each element u belongs to $\sigma(u)$ sets we have

$$\sum_{S_i \in \mathcal{C}''} Q_{S_i} = \sum_{S_i \in \mathcal{C}''} \sum_{u \in S_i} \frac{w(\mathcal{C}(u))}{b(u)} \leq \sum_{u \in U} \sigma(u) \frac{w(\mathcal{C}(u))}{b(u)} = \sum_{u \in U} \nu(u)\sigma_\$(u) = n \cdot \overline{\nu\sigma_\$} ,$$

and the lemma follows. □

Proof of Theorem 4: Considering the lower bounds provided by Lemma 14 and by Lemma 15, we conclude that the larger of the two bounds on $w(\textsc{alg})$ is minimized when $w(\textsc{opt}) = \sqrt{\frac{w(\mathcal{C})^3 \cdot k_{\max}}{n \cdot \overline{\nu \cdot \sigma_\$}}}$. Hence, we have

$$\mathbb{E}[w(\textsc{alg})] \geq w(\textsc{opt}) \cdot \frac{1}{16e} \sqrt{\frac{w(\mathcal{C})}{k_{\max} n \cdot \overline{\nu\sigma_\$}}} \geq w(\textsc{opt}) \cdot \frac{1}{16e} \sqrt{\frac{\overline{\sigma_\$}}{k_{\max}^2 \cdot \overline{\nu\sigma_\$}}} = w(\textsc{opt}) \cdot \frac{1}{16ek_{\max}} \sqrt{\frac{\overline{\sigma_\$}}{\overline{\nu\sigma_\$}}} ,$$

since $n \cdot \overline{\sigma_\$} \leq k_{\max} \cdot w(\mathcal{C})$ (see Eq. (4)). □

Additional Proofs from Section 3.3

Proof of Theorem 5: It follows from Lemma 5 that

$$\mathbb{E}[|\textsc{alg}|] \geq \frac{m^2}{n\overline{\sigma^2}} = \frac{n^2\overline{\sigma}^2}{k^2 n\overline{\sigma^2}} \geq \frac{\overline{\sigma}^2}{k\overline{\sigma^2}} \cdot |\textsc{opt}| ,$$

where the equality is due to the fact that $m\overline{k} = n\overline{\sigma}$ always holds, and the last equality is due to the fact that $|\text{OPT}| \leq n/k$ in the unit capacity, fixed k case. ◻

Proof of Theorem 6: From Eq. (3), we have that

$$\mathbb{E}[|\text{ALG}|] \geq \frac{|\text{OPT}|^2}{\sum_{S \in \text{OPT}} |N(S)| + 1} \geq \frac{|\text{OPT}|^2}{\sum_S k(S)} = \frac{|\text{OPT}|^2}{m\overline{k}} ,$$

and from Lemma 5, we also get that

$$\mathbb{E}[|\text{ALG}|] \geq \frac{m^2}{\sigma^2 n} = \frac{m^2}{\sigma m \overline{k}} = \frac{m}{\sigma \overline{k}} ,$$

since $n\sigma = m\overline{k}$.

The larger of the two bounds on $|\text{ALG}|$ is minimized when $|\text{OPT}| = m/\sqrt{\sigma}$, and therefore

$$\mathbb{E}[|\text{ALG}|] \geq \frac{|\text{OPT}|}{\overline{k} \cdot \sqrt{\sigma}} ,$$

as required. ◻