A Proportionate Fair Scheduling Rule with Good Worst-case Performance*

Micah Adler† Petra Berenbrink‡ Tom Friedetzky§ Leslie Ann Goldberg¶
Paul Goldberg¶ Mike Paterson¶

November 16, 2004

Abstract

In this paper we consider the following scenario. A set of n jobs with different threads is being run concurrently. Each job has an associated weight, which gives the proportion of processor time that it should be allocated. In a single time quantum, p threads of (not necessarily distinct) jobs receive one unit of service, and we require a rule that selects those p threads, at each quantum. Proportionate fairness means that over time, each job will have received an amount of service that is proportional to its weight. That aim cannot be achieved exactly due to the discretisation of service provision, but we can still hope to bound the extent to which service allocation deviates from its target. It is important that any scheduling rule be simple since the rule will be used frequently.

We consider a variant of the Surplus Fair Scheduling (SFS) algorithm of Chandra, Adler, Goyal, and Shenoy. Our variant, which is appropriate for scenarios where jobs consist of multiple threads, retains the properties that make SFS empirically attractive but allows us to prove that it achieves proportionate fairness, a result not known previously for any simple scheduler in a multiprocessor context. We show that when the variant is run, no job lags more than $H(n)-p+1$ steps below its target number of services, where $H(n)$ is the Harmonic function. Also, no job is over-supplied by more than $O(1)$ extra services. This analysis is tight and it also extends to an adversarial setting, which models some situations in which the relative weights of jobs change over time.

Keywords: Scheduling, Algorithms, Complexity

*Supported by NSF Research Infrastructure Award EIA-0080119, NSF Faculty Early Career Development Award CCR-0133664, and the Future and Emerging Technologies Programme of the EU under contract number IST-1999-14186 (ALCOM-FT). A preliminary version of this paper appeared in Proceedings of the 2003 SPAA conference.

†University of Massachusetts, Department of Computer Science, Amherst, MA 01003-4610, USA

‡Simon Fraser University, School of Computing Science, Burnaby, B.C., V5A 1S6, Canada

§University of Durham, Department of Computer Science, Durham, DH1 3LE, UK

¶University of Warwick, Department of Computer Science, Coventry, CV4 7AL, UK

1 Introduction

In this paper, we consider the problem of scheduling a set of n jobs on p processors, where the objective is to schedule the jobs so that, at every time step in the schedule, each job has received as close to a proportionate share of scheduling slots as possible. Assuming that time is discretised into time steps (or quanta), a scheduler may, in each step, allocate processors to p out of the n jobs. If job i has a weight w_i associated with it, *proportionate fairness* requires that after t steps, the number of times that job i has been assigned a processor should be close to $t \cdot p \cdot w_i / \sum_j w_j$.

A very strict sense of this kind of fairness is called *P-Fairness* (5), which requires that after t steps, job i has been assigned to a processor for either $\lfloor (t \cdot p \cdot w_i / \sum_j w_j) \rfloor$ or $\lceil (t \cdot p \cdot w_i / \sum_j w_j) \rceil$ time steps. In other words, every job receives as close to its proportion of service as is possible given integral service constraints. P-Fairness was first introduced in (5), and a number of papers have addressed the problem of designing P-Fair scheduling algorithms that are efficient and practical (4; 1; 2; 11; 7).

An important application of proportionate scheduling algorithms is an operating system assigning jobs to quanta on a multi-processor system. There are a number of practical reasons why existing P-Fair algorithms are not ideally suited for this task. For example, these algorithms assume fixed-length quanta (i.e., no job ever blocks in the middle of a quantum), and they assume that there are no arrivals or departures of jobs. Furthermore, in (7) it is shown that when one of the existing P-Fair algorithms is applied to scenarios with both variable-length quanta and arrivals and departures, the schedule becomes non-work-conserving: at some time steps, a processor is left idle even when there are more jobs in the system than processors. Also, existing P-Fair algorithms are still somewhat complicated and, in fact, (1) provides evidence that designing simpler P-Fair algorithms may be quite difficult. However, for a task such as the assignment of jobs to quanta on a multi-processor system, it is crucial that the scheduling algorithm be extremely fast, since the scheduler will be called by the operating system on the expiration of every quantum.

A simple proportionate fair scheduling algorithm is introduced in (6). This algorithm is called SFS (Surplus Fair Scheduling), and can be viewed as a generalisation to multiprocessors of scheduling techniques based on generalised processor sharing, which have been well studied for use in uniprocessor systems (10; 8; 12). In SFS, each job i maintains a quantity S_i, the "start time". For every round where job i is run, S_i is incremented by $\frac{1}{w_i}$. At the start of each round, the p jobs are run that have the minimum values of $\alpha_i = w_i(S_i - v)$, where $v = \min_j S_j$. (6) provides empirical evidence that SFS has good fairness properties. Furthermore, the algorithm is simple, and generalises quite easily to scenarios with both variable-length quanta, as well as arrivals and departures of jobs. Also, the algorithm is always work-conserving. However, despite the experimental evidence that SFS performs well, the task of proving that SFS does in fact always schedule jobs so that each receives close to a proportionate share of the available processing power has remained an open problem. We note that a number of simple schedulers have been introduced for the (easier) uniprocessor case that have provable guarantees on fairness, including (10; 14; 15; 13).

Note that SFS is not P-fair. Initially, all values of S_i are zero. Hence, any job i has $\alpha_i = 0$ before it is run for the first time by a processor. Hence, any job that has not yet been run will have a higher priority than any job that has been run, so in the first approximately n/p steps, all jobs will be run for the first time. However, P-fairness could require high-weight jobs to be run more

than once before the low-weight jobs are run. For a high-weight job i at time $t = \lfloor n/p \rfloor$, its target service allocation $t \cdot p \cdot w_i / \sum_j w_j$ could be more than 2 (if $p|n$ this just requires $w_i > \frac{2}{n} \sum_j w_j$).

1.1 The "Leaky Bucket" Representation

A version of the SFS algorithm has an intuitively appealing interpretation, which we call the *leaky-bucket* problem; the focus of this paper will be on studying this problem. We think of each job as a bucket of water that leaks. During each time step, k_i units leak from bucket i, and the total volume of leaked water is p. This is replaced by adding p refills of unit size into each of the p emptiest buckets. Initially, all the buckets have A units of water, for some value of A.

For the correspondence between the leaky-bucket problem and SFS, we first redefine the value v of SFS as the weighted average of the start times ($v = \frac{\sum_j w_j S_j}{\sum_j w_j}$) instead of the minimum start time. Note that this new value of v grows at the same rate as with the original definition from (6), and the algorithm maintains all of the properties that make it attractive from a practical point of view. With the new version of v, the value α_i represents the water level in bucket i. During a step of SFS, the value of v increases by $p/\sum_j w_j$, which corresponds to a decrease in each α_i of $k_i = pw_i/\sum_j w_j$ and a total decrease of $\sum_i k_i = p$. On the other hand, each job i that gets run has α_i increased by 1, giving a total increase of p.

The key observation that relates the leaky-bucket problem to fairness guarantees is the following. After t steps, the number of times that job i is assigned a processor is $(tpw_i/\sum_j w_j) + X_i(t) - A$, where $X_i(t)$ denotes the current load of bucket i and A denotes the initial (and average) load. (This is formalised as Observation 1.)

Observation 1 *Consider any fixed system and scheduler. After t steps, the number of times that job i is assigned a processor is $(tpw_i/\sum_j w_j) + X_i(t) - A$.*

Proof. $X_i(t)$ is equal to A plus the number of times that bucket i is refilled minus tk_i. The number of refills is equal to the number of services for job i. Also, recall that $k_i = pw_i/\sum_j w_j$. □

In this paper we study the following scheduler, that corresponds to a variant of SFS. Instead of refilling the p emptiest buckets, at each step we refill the buckets sequentially, where, for each of the p refills, we choose the current emptiest bucket and add one unit of water. Thus, the same bucket can be refilled multiple times. We refer to this process of refilling buckets as scheduler $\mathcal{S}_0$. We study $\mathcal{S}_0$ for two reasons: first, understanding this variant will lead to insights concerning the behaviour of the p-emptiest-buckets scheduler that corresponds to SFS. Second, $\mathcal{S}_0$ also represents an important scenario from a practical perspective. In particular, $\mathcal{S}_0$ corresponds to the case where several processors can service (different threads of) the same job simultaneously. Thus, it can be used when jobs have multiple threads which must all be executed and may be executed simultaneously. We here assume that each job always has at least p available threads.

We derive bounds on how much the loads of buckets deviate from the average load A, using scheduler $\mathcal{S}_0$. These bounds translate directly to additive bounds on the deviation between the number of times that a job is serviced in t steps, and the number of times that it should be serviced.

We also prove analogous results for *adversarial systems*, where the sequence k_i is no longer fixed, but instead can vary with time. This corresponds to the situation where the weights of jobs can vary with time, which might occur as a result of changes in the relative importance of the individual jobs, or due to jobs arriving and departing. While this is an important practical consideration, to the best of our knowledge there have not been any previous results proven for proportionate fair scheduling of jobs with varying weights.

We define the system more formally in Section 1.2 and state our results in Section 1.3. Note that in some respects our leaky-bucket model resembles the scheduling problems studied in (3; 9) in which a scheduler must, in an on-line manner, repeatedly select one of a number of *buffers* to be served, and the service has the effect of reducing the queue at that buffer. Meanwhile, packets are arriving at buffers, and the general aim is to minimise the maximum length attained by any queue. These papers study the *competitive ratio* of scheduling algorithms, which is the ratio between the performance of the scheduler (largest queue length arising), and the largest queue length for an optimal schedule (chosen in an off-line manner). The rule *Longest queue first* studied in (9) corresponds with the scheduler $\mathcal{S}_0$ we consider here (where buckets correspond to buffers, and loss of water level corresponds to queue length). By contrast, in this paper we do not compare the bucket loads obtained via $\mathcal{S}_0$ with sequences of bucket loads obtained via some alternative schedule; instead we are just concerned with the extent to which they may deviate, in absolute terms, from their initial values (which are the target loads at all times). The problem we face here is the fractional leakage rates in conjunction with discretised service provision. $\mathcal{S}_0$ is shown to be – in the adversarial setting – at least as good as any other scheduler in addressing that problem.

1.2 Models, Terminology and Notation

Let $B_1, \ldots, B_n$ denote a sequence of n buckets, having associated variables $X_1, \ldots, X_n$, where $X_i \in \mathbb{R}$ denotes the amount of material being held in B_i. X_i will be called the *load* of B_i.

The system evolves over discrete time steps, so that $X_i = X_i(t), t = 0, 1, 2, \ldots$. In a single time step, each X_i is first reduced by some amount $k_i \geq 0$. Assume that the total depletion is p, i.e., $p = \sum_j k_j$. We consider algorithms which restore the total load $\sum_j X_j$ by adding the p units back, but are constrained to do so by adding p refills of size 1 to some of the X_i's. Given a rule for selecting which buckets are replenished, we consider how much the X_i's may fluctuate from their original levels.

A system is said to be *stable* whenever there are upper and lower bounds on the values that any of the X_i can take, over time. We are interested in proving stability, and furthermore in identifying bounds on the values of the X_i. The main technical challenge is in finding lower bounds. If initially we have $X_i(0) = A$ for $i = 1, \ldots, n$, we analyse how large A must be to ensure that no bucket ever becomes empty. Let the *intermediate state* $X'_i(t)$ denote the level of the i-th bucket after the depletions in step t but before the refills. That is, the level changes from $X_i(t-1)$ to $X'_i(t)$ by doing depletions, then to $X_i(t)$ by refills. (In Section 3 we extend the notation to denote the loads resulting when some but not all of the refills have been completed.) Define the *outcome* ψ of the system to be $\inf_{t=1,2,\ldots;\ i=1,\ldots,n} X'_i(t)$, i.e., the greatest lower bound (if it exists) on any bucket load. Thus a stable system is one that has a finite outcome, and we are looking for bounds on the outcome.

FIXED SYSTEMS In a fixed system, the values k_i are constants, and are the parameters of problem instances. Each k_i is the rate of depletion for B_i. In step t of this basic system, we first *deplete* each $X_i(t-1)$ by k_i, giving a sequence of *intermediate* values $X_i'(t)$ with $X_i'(t) = X_i(t-1) - k_i$ for $1 \leq i \leq n$. Then scheduler $\mathcal{S}_0$ selects some buckets in order to refill them using the following rules. Iteratively for p rounds, $\mathcal{S}_0$ finds an emptiest bucket and adds 1 to its load. This means that one bucket can be refilled more than once in a time step.

ADVERSARIAL SYSTEMS In *adversarial systems*, we no longer have a fixed sequence of k_i but assume the presence of an adversary. At each step the adversary is free to choose a sequence $k_1(t), \ldots, k_n(t)$ (where t again denotes the time parameter), subject to $k_i(t) \geq 0$, and $\sum_j k_j(t) = p$. Hence the depletion rate of a bucket can differ from round to round. These systems turn out to be useful to establish bounds on worst-case behaviour.

1.3 Summary of Results

Let $H(n)$ denote the harmonic function, $H(n) = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$.

Theorem 1 of Section 2 is a strong positive result about the outcome achieved by scheduler $\mathcal{S}_0$ in the adversarial setting. The theorem states that (i) There is an adversary $\mathcal{A}_0$ which achieves an outcome of at most $A - (H(n) + p - 1)$ against *any* scheduler $\mathcal{S}$ that refills buckets in p units of size 1 at each step (regardless of how $\mathcal{S}$ chooses the buckets). Also, (ii) Scheduler $\mathcal{S}_0$ achieves an outcome of at least $A - (H(n) + p - 1)$ against any adversary $\mathcal{A}$. Thus, scheduler $\mathcal{S}_0$ is optimal in the adversarial setting. Theorem 1 implies that in any fixed system, the outcome ψ is at least $A - (H(n) + p - 1)$. This implies that no job lags more than $H(n) + p - 1$ steps below the target number of services.

In Section 3 we show that this bound is tight, for fixed systems. In particular, Theorem 2 states that for all $\tau > 0$ there are constant depletion rates $k_1, k_2, \ldots, k_n$ such that the minimum load of any bucket is less than $A - (1 - \tau)(H(n) + p - 1)$ when scheduler $\mathcal{S}_0$ is run. The construction uses a sequence of k_i's that converges to 0 very quickly, suggesting that scheduler $\mathcal{S}_0$ may do better in cases in which the relative values of the k_i's are constrained. Theorem 3 identifies some cases in which this is true. In particular, if each k_j is a multiple of the smallest depletion rate $k_{\min}$ then scheduler $\mathcal{S}_0$ achieves an outcome of at least $A - pk_{\max}/k_{\min}$.

It is also shown that for scheduler $\mathcal{S}_0$ there is an easy upper bound of $A + 1$ on the maximum load of any bucket. This means that no job gets more than one extra service at any point.

2 Adversarial Systems

In this section we restrict our attention to adversarial systems, as introduced in Section 1.2. In contrast to fixed systems, we may assume without loss of generality that the states (sequences of X_i values) are sorted in non-decreasing order, i.e., $X(t) = (X_1(t), \cdots, X_n(t))$ with $X_1(t) \leq X_2(t) \leq \cdots \leq X_n(t)$. This means we may "re-sort" our buckets at the end of every step. We introduce the following notation to represent quantities that are used throughout the proof. For $1 \leq i \leq n$ let

1. $\mu_i(t) = \frac{1}{i} \sum_{j=1}^{i} X_j(t)$

2. $V_i(t) = \mu_i(t) - H(i)$

3. $V(t) = \min_i V_i(t)$

4. $\ell(t) = \min\{\ell : \ V_\ell(t) = V(t)\}$.

$\mu_i(t)$ is the average load of buckets $B_1, \ldots, B_i$; by our "sortedness" assumption, it is the average of the i lowest loads. $V_i(t)$ is a measure of the outcome achievable by an adversary restricted to $B_1, \ldots, B_i$; $V(t)$ is the minimum value of $V_i(t)$ for all choices of i; finally $\ell(t)$ is smallest value of i that minimises $V_i(t)$.

The following technical lemma is used in the proof of Theorem 1.

Lemma 1 *For any state $X(t)$, $j \in \{1, \ldots, n-1\}$,*

$$V_{j+1}(t) > V_j(t) \quad \Longleftrightarrow \quad X_{j+1}(t) > \mu_j(t) + 1.$$

Proof. Consider the following sequence of equivalent inequalities.

$$\begin{aligned}
V_{j+1}(t) &> V_j(t) \\
\mu_{j+1}(t) - H(j+1) &> \mu_j(t) - H(j) \\
\frac{1}{j+1}\sum_{k=1}^{j+1} X_k(t) - H(j+1) &> \frac{1}{j}\sum_{k=1}^{j} X_k(t) - H(j) \\
\frac{X_{j+1}(t)}{j+1} - \frac{1}{j+1} &> \frac{X_1(t) + \cdots + X_j(t)}{j(j+1)} \\
X_{j+1}(t) - 1 &> \frac{X_1(t) + \cdots + X_j(t)}{j} = \mu_j(t).
\end{aligned}$$

□

Informally, Theorem 1 says that regardless of what scheduler is used, adversary $\mathcal{A}_0$ (as defined in the proof) manages to get the minimum $X'_i(t)$ at least as far as $(H(n) + p - 1)$ below the starting line. But, irrespective of any strategy used by an adversary, scheduler $\mathcal{S}_0$ will keep the all-time minimum no lower than that level. Hence, in the presence of a "worst-case" mechanism for depleting the buckets, scheduler $\mathcal{S}_0$ is as good as any other scheduler using p unit refills.

Theorem 1 *Let $\mathcal{S}$ be an arbitrary scheduler that refills buckets in p units of size 1 at each step. Let $\mathcal{S}_0$ be the scheduler defined in Section 1.2.*

1. *There is an adversary $\mathcal{A}_0$ which achieves an outcome of $\leq A - (H(n) + p - 1)$ against any scheduler $\mathcal{S}$.*

2. *The scheduler $\mathcal{S}_0$ achieves an outcome of $\geq A - (H(n) + p - 1)$ against any adversary $\mathcal{A}$.*

Proof. We put $A = 0$, allow bucket loads to become negative, and establish $-(H(n) + p - 1)$ as the bound on how negatively large any X_i need become. Hence, $X(0) = (0, \ldots, 0)$.

PROOF OF (1) To define $\mathcal{A}_0$, we use the following rule that for any state finds a set of depletions that add up to p. Then we show that after finitely many steps the lowest bucket load becomes at most $-(H(n)+p-1)$.

$\mathcal{A}_0$'s behaviour for state $X(t)$:

1. If $X_1(t) \leq -(H(n)-1)$ then let $X'_1(t) = X_1(t) - p$.
2. Else { *since* $X_1(t) > -(H(n)-1)$ *we have* $\ell(t) > 1$ }
3. Let $j = \ell(t)$
4. If $\exists i \in 1,\ldots,j$ with $X_i(t) - (\mu_j(t) - \frac{1}{j}) \neq \lfloor X_i(t) - (\mu_j(t) - \frac{1}{j}) \rfloor$
5. $X'_i(t) = \lfloor X_i(t) - (\mu_j(t) - \frac{1}{j}) \rfloor$
6. Let $i' \in 1,\ldots,j; i' \neq i$.
7. $X'_{i'}(t) = X_{i'}(t) - p + (X_i(t) - X'_i(t))$
8. Else fail. We show that this case never arises.

Note first that if $X_1(t) \leq -(H(n)-1)$, then the adversary's choice of depletions is simply to deplete the minimum bucket (with load $X_1(t)$) by p, applying line 1 above, and the claimed outcome is attained. (Recall that bucket loads are sorted so that $X_i(t) \leq X_{i+1}(t)$ for $1 \leq i \leq n-1$.)

Suppose for a contradiction that line (8) is reached. In that case, the quantities $X_i(t)$ for $1 \leq i \leq j$ all satisfy $X_i(t) - (\mu_j(t) - \frac{1}{j}) \in \mathbf{Z}$. Since $\mu_j(t)$ is the mean of these quantities, there must exist i with $X_i(t) - (\mu_j(t) - \frac{1}{j}) \geq 1$. This implies that $\mu_{j-1}(t) \leq \mu_j(t) - \frac{1}{j}$, hence $V_{j-1}(t) \leq V_j(t)$, hence $\ell(t) \leq j-1$, a contradiction.

We claim that if lines (2-7) are applied, then one or both of the following takes place in going from $X(t)$ to $X(t+1)$.

1. The number of values of $i \in \{1,\ldots,\ell(t)\}$ with $X_i(t) - (\mu_j(t) - \frac{1}{j}) \neq \lfloor X_i(t) - (\mu_j(t) - \frac{1}{j}) \rfloor$ goes down by 1, or
2. $V(t+1) \leq V(t) - \frac{1}{\ell(t)}$.

Suppose that the scheduler $\mathcal{S}$ refills any bucket with load $X'_k(t)$ for $k > j$. Then the total load of the j least-loaded buckets goes down by ≥ 1. Hence their average load goes down by at least $1/j$, and we have $\mu_j(t+1) \leq \mu_j(t) - \frac{1}{j}$. so we obtain the second of the above outcomes. It follows that $V_j(t+1) \leq V_j(t) - \frac{1}{j}$, and hence $V(t+1) \leq V(t) - \frac{1}{j}$.

If $\mathcal{S}$ gives all p refills to buckets with loads $X'_i(t)$, for $1 \leq i \leq j$, then we have $\mu_j(t+1) = \mu_j(t)$, and since all refills are integer-valued, $\mathcal{S}$ cannot change the number of buckets whose loads differ from $\mu_j(t) - \frac{1}{j}$ by an integer. $\mathcal{A}_0$ has thus reduced by 1 the number of such buckets, in line 5.

Putting the above two observations together, either the quantity $\mu_j(t)$ must decrease by at least some constant (while $\ell(t)$ does not increase), or alternatively (after $\leq \ell(t)$ iterations) the value of $\ell(t)$ must decrease. Hence the process terminates, since $\ell(t) = 1$ means that line 1 applies.

PROOF OF (2) Suppose for a contradiction that there exists an adversary $\mathcal{A}$ which achieves an outcome of less than $-(H(n)+p-1)$ against $\mathcal{S}_0$. This means that after t time steps for some finite t, state $X(t)$ satisfies $X_1(t) < -(H(n)-1)$ (this is because $\mathcal{A}$ would deplete the smallest bucket by p in order to minimise the outcome), i.e. $\mu_1(t) < -(H(n)-1)$, hence $V_1(t) < -H(n)$.

Observe that, by definition, for all states $X(t)$, $V_n(t) = -H(n)$ since the overall average is always 0. Initially, $V_j(1) > V_{j+1}(1)$ for all $j < n$ since we assume all buckets to start with load 0, i.e., the harmonic numbers are all that count. Finally, $V(t) < -H(n) = V_n(t)$.

Let $t' = \min\{t \ : \ V_j(t) < -H(n) \text{ for some } j\}$. From our observation above we have $V_n(t') = -H(n)$, so we can choose the smallest j so that we also have $V_j(t') < V_{j+1}(t')$. Consider the previous state $X(t'-1)$, and we will show that $V_j(t'-1) < -H(n)$, contradicting our assumption that $X(t')$ was the first state with the given property (which is not a property of $X(0)$). Since $V_j(t') < V_{j+1}(t')$ we have by Lemma 1

$$X_{j+1}(t') > \mu_j(t') + 1 = \frac{X_1(t') + \cdots + X_j(t')}{j} + 1.$$

Now observe that $\mathcal{S}_0$ cannot have just refilled any bucket with load $X_k(t')$ for $k > j$, just prior to reaching state $X(t')$. Since those buckets have a load more than 1 above the average of the lowest j buckets, it follows that the lowest bucket has a load less than $X_k(t') - 1$ for all $k > j$. $\mathcal{S}_0$ would certainly have preferred one of the buckets with loads $X_1(t'), \ldots, X_j(t')$, which would be lower prior to a refill of size 1. If $Y_1, \ldots, Y_j$ denote the loads of the buckets corresponding to $X_1(t'), \ldots, X_j(t')$ in state $X(t'-1)$ (these are not necessarily $X_1(t'-1), \ldots, X_j(t'-1)$), then $Y_1 + \cdots + Y_j \leq X_1(t') + \cdots + X_j(t')$, since $\mathcal{S}_0$ added p to these buckets and $\mathcal{A}$ previously subtracted at most p. Hence

$$\mu_j(t'-1) \leq \frac{Y_1 + \cdots + Y_j}{j} \leq \mu_j(t').$$

So $V_j(t'-1) \leq V_j(t') < -H(n)$, contradicting the minimality in the choice of t'. □

3 Fixed Systems

In Section 2 we showed that scheduler $\mathcal{S}_0$ keeps the all-time minimum to $A - (H(n) + p - 1)$ against an adversary that may select the depletion rates in every step. Of course, this upper bound on the initial bucket load holds also in the case of our fixed system. This gives the following corollary.

Corollary 1 *If $A \geq H(n) + p - 1$, then loads $X_i(t)$ will always be non-negative, for any leakage rates $k_1, \ldots, k_n$ with scheduler S_0.*

Theorem 2 uses a construction of leakage rates that show that the above condition $A \geq H(n) + p - 1$ is necessary as well as sufficient. However, the construction leads to a very large ratio between highest and lowest depletion rates. This motivates the restriction to rational depletion rates, where bounds are obtained in terms of the above ratio.

In Theorem 3 we will show an alternative upper bound on the initial bucket load A needed to maintain positive loads. This gives better results in the case that the k_i's are all small multiples of a common value.

3.1 Arbitrary Real-valued Depletion Rates

In the following we show that in the worst case (worst case over all choices of k_i's summing to p) with scheduler $\mathcal{S}_0$ a bucket may be depleted by up to $H(n) + p - 1$. An alternative statement is that, assuming all buckets are initially set to the same level, they must start with a level of $A = H(n) + p - 1$ in order to avoid becoming empty.

The worst-case behaviour we obtain for constant depletion rates essentially matches the upper bound on worst-case behaviour for the adversarial case. More precisely (see Theorem 2 below), we show how to construct sets of depletion rates that lead to a minimum bucket value that is arbitrarily close to the $A - (H(n) + p - 1)$ obtained in the adversarial case.

Here is a brief overview of the strategy of the proof. Note that the adversary in part (1) of Theorem 1 works by forcing the scheduler to raise the level of some bucket B at least one unit above the others' average. This trick is then repeated recursively on the un-raised buckets, until the last one is depleted by the required amount. This process is approximately mimicked by fixed leakage rates as follows. Give some bucket a very slow leakage rate. Verify that it is raised almost to the level of one above the others. Verify that its leakage rate is so slow that it remains almost at that level for long enough that the others can simulate the strategy recursively. This results in a choice of leakage rates with $k_1 \gg k_2 \gg \cdots \gg k_n$ where bucket B_1 with leakage rate k_1 is destined to reach a level close to $A - (H(n) + p - 1)$.

It is convenient to assume $A = 0$ and ask how negatively large can the value of any bucket become. Thus we show that bucket B_1 may be depleted to a level arbitrarily close to $-(H(n)+p-1)$, for suitable choice of the k_i.

We use the following notation to represent the sequence of values generated by an n-bucket system S with leakage rates $k_1, \ldots, k_n$, for $n \geq 2$. We need to use notation that is a bit more "detailed" than that of Section 2, in order to refer to loads reached when r refills have been completed, for $0 \leq r \leq p$.

1. For $i \in \{1, \ldots, n\}$ let $X_i^p(0) = 0$.
2. For $t \in \{0, 1, 2 \ldots\}$ and $i \in \{1, \ldots, n\}$ let $X_i^0(t+1) = X_i^p(t) - k_i$.
3. For $r \in \{0, \ldots, p-1\}$, $(X_1^{r+1}(t), \ldots, X_n^{r+1}(t))$ is obtained from $(X_1^r(t), \ldots, X_n^r(t))$ by adding 1 to a minimal component of $(X_1^r(t), \ldots, X_n^r(t))$.
4. For $r \in \{0, \ldots, p\}$, $1 \leq i \leq n$, let $\mu_i^r(t) = \frac{1}{i} \sum_{j=1}^{i} X_j^r(t)$.

Informally, $X_i^r(t)$ denotes the load of bucket i at time t, after the scheduler has completed r out of the p refills. So item (3) above is stating the rule that a bucket with minimal load is chosen each time. $\mu_i^r(t)$ denotes that average load of buckets 1 through i, and recall that these are not necessarily the buckets with the lowest loads (in contrast with $\mu_i(t)$ in the previous section); they are just the buckets with leakage rates $k_1, \ldots, k_i$.

Observation 2 *It follows from the definition of $\mu_i^r(t)$ that*

$$X_i^r(t) - \mu_{i-1}^r(t) = \frac{i}{i-1}(X_i^r(t) - \mu_i^r(t)).$$

Equivalently, by rearranging terms:

$$\mu_i^r(t) - \mu_{i-1}^r(t) = \frac{X_i^r(t) - \mu_i^r(t)}{i-1}.$$

Observation 3 *For $i > 1$, bucket i will not be refilled if its level is above the average of the first i buckets; formally, for $0 \leq r < p$,*

$$\text{if } X_i^r(t) - \mu_i^r(t) > 0 \text{ then } X_i^{r+1}(t) = X_i^r(t).$$

Observation 4 *For all r,i,t, $X_i^r(t) \leq 1 - \frac{1}{n}$.*

Observation 4 is proved by contradiction. Let $t' = \min\{t : \exists r, i,\ X_i^r(t) > 1 - \frac{1}{n}\}$.

1. Suppose that $X_i^0(t') > 1 - \frac{1}{n}$, but $X_i^p(t'-1) \leq 1 - \frac{1}{n}$. This is impossible since $X_i^0(t') = X_i^p(t'-1) - k_i < X_i^p(t'-1)$.

2. Alternatively suppose that $X_i^r(t') > 1 - \frac{1}{n}$ but $X_i^{r-1}(t') \leq 1 - \frac{1}{n}$ for $1 \leq r \leq p$. Then $X_i^{r-1}(t') > -\frac{1}{n}$. But for bucket B_i to be refilled, we need $X_i^{r-1}(t') \leq \mu_n^{r-1}(t')$ (by Observation 3). This is a contradiction since for all t, $\mu_n^{r-1}(t) \leq \mu_n^{p-1}(t) = -\frac{1}{n}$.

Lemma 2 *For all $t \geq 0$, $i \in \{1, \ldots, n\}$, $r \in \{0, \ldots, p\}$:*

$$|X_i^r(t) - \mu_i^r(t)| \leq (p + H(n)).$$

Proof. It follows from Theorem 1 (taking the initial loads A to be zero) that

$$\forall i, r, t \quad X_i^r(t) \geq -(H(n) + p - 1). \tag{1}$$

It follows from Observation 4 that

$$\forall i, r, t \quad X_i^r(t) \leq 1. \tag{2}$$

(We have just replaced the upper bound of $1 - \frac{1}{n}$ by 1 in order to simplify the expressions used later on.) From (1) and (2) we have

$$\forall i, r, t \quad X_i^r(t) \in [-(H(n) + p - 1), 1]. \tag{3}$$

Since the quantity $\mu_i^r(t)$ is the mean of a subset of the $X_i^r(t)$ values we have

$$\forall i, r, t \quad \mu_i^r(t) \in [-(H(n) + p - 1), 1]. \tag{4}$$

The result follows from (3) and (4) since the the difference between any pair of values in $[-(H(n) + p - 1), 1]$ is at most the length of the interval. □

We choose leakage rates $k_1, \ldots, k_n$ as follows. Let τ be a small positive number. Then

$$\begin{aligned} k_i &= \frac{\tau^i}{20^i (p+H(n))^i n^i} \quad \text{for } i = 2, 3, \ldots, n, \\ k_1 &= p - \textstyle\sum_{j>1} k_j. \end{aligned}$$

Let $J_i = \left\lceil \frac{4}{k_i}(p + H(n)) \right\rceil$. Hence for $i > 1$,

$$J_i = \left\lceil \frac{4(20^i)(p + H(n))^{i+1} n^i}{\tau^i} \right\rceil. \tag{5}$$

Note that if τ is sufficiently small then k_1 is positive, and in addition, $k_i > \sum_{j>i} k_j$ and $J_i > \sum_{j<i} J_j$.

Informally, in Lemma 3, we want to find a step in which bucket i gets the last refill of the step, and the level of bucket i was (before refilling) at most τ below the average of the first i buckets. We show that we can find such a step in any sequence of length J_i in which no bucket with index higher than i is refilled.

Lemma 3 *Let leakage rates k_i be constructed as above with $0 < \tau < \frac{1}{2n}$. Let $i \in \{1, 2, \ldots, n\}$ and $T \geq 1$. Suppose that for all $t \in \{T, \ldots, T + J_i - 1\}$ and for all $j > i$, we have $X_j^0(t) = X_j^p(t)$. Then there exists $t' \in \{T, \ldots, T + J_i - 1\}$, with*

(1) $X_i^{p-1}(t') - \mu_i^{p-1}(t') > -\tau$ and

(2) $X_i^p(t') = X_i^{p-1}(t') + 1$.

Proof. The lemma is proved by induction on i. The base case, $i = 1$, is straightforward.

For the inductive step, suppose $i \geq 2$ and suppose $T \geq 1$ has the property that $\forall t \in \{T, \ldots, T + J_i - 1\}$, $\forall j > i$, we have $X_j^0(t) = X_j^p(t)$. We want to show that there is a $t' \in \{T, \ldots, T + J_i - 1\}$ such that (1) and (2) hold. By the inductive hypothesis, we know that for all T' in the range $T \leq T' \leq T + J_i - J_{i-1}$, the following is true. If, for all $t \in \{T', \ldots, T' + J_{i-1} - 1\}$, we have $X_i^0(t) = X_i^p(t)$, then there is a $t'' \in \{T', \ldots, T' + J_{i-1} - 1\}$ such that

(1′) $X_{i-1}^{p-1}(t'') - \mu_{i-1}^{p-1}(t'') > -\tau$ and

(2′) $X_{i-1}^p(t'') = X_{i-1}^{p-1}(t'') + 1$.

For $t \in \{T, \ldots, T + J_i - 2\}$ and $r \in \{0, \ldots, p\}$, note that between $X^r(t)$ and $X^r(t+1)$, the system deletes $\sum_{j \leq i} k_j$ units from the first i buckets, and adds back p units. Thus, $\mu_i^r(t+1) - \mu_i^r(t) = \frac{1}{i} \sum_{j>i} k_j$ Since $k_i > \sum_{j>i} k_j$ we have

$$\mu_i^r(t+1) - \mu_i^r(t) < \frac{1}{i} k_i < k_i. \tag{6}$$

Using (6) and the fact that $X_i^r(t+1) - X_i^r(t) \geq -k_i$, we also have the following.

$$X_i^r(t+1) - \mu_i^r(t+1) > X_i^r(t) - \mu_i^r(t) - 2k_i. \tag{7}$$

If in addition we assume that $X_i^p(t) = X_i^0(t)$ and $X_i^p(t+1) = X_i^0(t+1)$, then we also have for $r \in \{0, \ldots, p\}$ that

$$X_i^r(t+1) - \mu_i^r(t+1) \leq X_i^r(t) - \mu_i^r(t) - k_i. \tag{8}$$

(7) and (8) give useful upper and lower bounds on the rate of decrease of $X_i^r(t) - \mu_i^r(t)$ as a function of t. To prove the inductive step we consider two cases.

Case 1: $X_i^p(T-1) - \mu_i^p(T-1) \geq 0$.
Let $t' = \min\{t : t \geq T \text{ and } X_i^p(t) > X_i^0(t)\}$. (Thus, t' is the first step, after and including step T, during which B_i is refilled.) We show that t' satisfies conditions (1) and (2) in the statement of the lemma.

First, note that $t' \leq T + 2(p + H(n))/k_i$ using (8) and Lemma 2. This implies $t' \leq T + \frac{1}{2}J_i \leq T + J_i - 1$.

Next, we show $t' > T$. To establish this fact, we will show that, for all $r < p$, we have $X_i^r(T) - \mu_i^r(T) > 0$ (and note Observation 3). To see this, note that

$$\begin{aligned}
\mu_i^r(T) = \mu_i^0(T) + \frac{r}{i} &= \mu_i^p(T-1) - \frac{1}{i}\sum_{j \leq i} k_j + \frac{r}{i} \\
&= \mu_i^p(T-1) - \frac{1}{i}(p - \sum_{j > i} k_j) + \frac{r}{i} \\
&\leq \mu_i^p(T-1) - \frac{1}{i}(p - k_i - r) \\
&\leq X_i^p(T-1) - k_i - \frac{1}{i}(p - k_i - r) + k_i \\
&\leq X_i^r(T) - \frac{1}{i}(p - k_i - r) + k_i < X_i^r(T).
\end{aligned}$$

where the last inequality uses the condition that r is strictly less than p.

Suppose, for a contradiction of (1′), that for some $r < p$, $X_i^r(t') - \mu_i^r(t') \leq -\tau$. Then, by Equation (7), $X_i^r(t'-1) - \mu_i^r(t'-1) < 2k_i - \tau$. By definition of t' we have $X_i^p(t'-1) = X_i^0(t'-1)$, so

$$X_i^{p-1}(t'-1) - \mu_i^{p-1}(t'-1) < 2k_i - \tau. \tag{9}$$

Define t_1 and t_2 as follows:

$$\begin{aligned}
t_1 &= \min\{t : T \leq t < t' \text{ and } X_i^{p-1}(t) - \mu_i^{p-1}(t) \in (-\tau + 2k_i, -\tau + \tfrac{\tau}{i})\}, \\
t_2 &= \max\{t : T \leq t < t' \text{ and } X_i^{p-1}(t) - \mu_i^{p-1}(t) \in (-\tau + 2k_i, -\tau + \tfrac{\tau}{i})\}.
\end{aligned}$$

To see that t_1 and t_2 are well-defined given (9), note that the interval $(-\tau + 2k_i, -\tau + \frac{\tau}{i})$ has length $\frac{\tau}{i} - 2k_i$, which is greater than $2k_i$. We have just shown that $X_i^{p-1}(T) - \mu_i^{p-1}(T) > 0$, and by Equation (9), $X_i^{p-1}(t'-1) - \mu_i^{p-1}(t'-1) < 2k_i - \tau < 0$. Equation (7) guarantees that for $t \in \{T, \ldots, t'-1\}$, $X_i^{p-1}(t+1) - \mu_i^{p-1}(t+1) - (X_i^{p-1}(t) - \mu_i^{p-1}(t)) > -2k_i$.

Furthermore, using (7), for $i > 1$,

$$t_2 - t_1 \geq \frac{(\tau/i) - 2k_i}{2k_i} > \frac{\tau/2i}{2k_i} = \frac{\tau}{2i}\frac{(20^i)(p + H(n))^i n^i}{2\tau^i}$$

$$= \frac{5n}{4i} \frac{4(20^{i-1})(p + H(n))^i n^{i-1}}{\tau^{i-1}} \geq J_{i-1}.$$

If $X_i^{p-1}(t) - \mu_i^{p-1}(t) \in (-\tau + 2k_i, -\tau + \frac{\tau}{i})$ then $X_i^{p-1}(t) - \mu_i^{p-1}(t) < \frac{i-1}{i}(-\tau)$, so by Observation 2, $X_i^{p-1}(t) - \mu_{i-1}^{p-1}(t) < -\tau$, which implies

$$X_i^{p-1}(t) - \mu_{i-1}^{p-1}(t) < -\tau \text{ for } t \in \{t_1, \ldots, t_2\}. \tag{10}$$

To see that Equation (10) holds, we use the fact that $X_i^{p-1}(t+1) - \mu_i^{p-1}(t+1) \leq X_i^{p-1}(t) - \mu_i^{p-1}(t)$. By Equation (8), this holds for $t \in \{T, \ldots, t' - 2\}$.

Applying the inductive hypothesis by plugging in t_1 for the T' in the statement of the hypothesis, there must exist $t'' \in \{t_1, \ldots, t_1 + J_{i-1} - 1\}$ satisfying $(1')$ and $(2')$. We have established that $t_2 \geq t_1 + J_{i-1}$, hence $t'' \leq t_2$. From (10) we have

$$X_i^{p-1}(t'') - \mu_{i-1}^{p-1}(t'') < -\tau.$$

From $(1')$, we have

$$X_{i-1}^{p-1}(t'') - \mu_{i-1}^{p-1}(t'') > -\tau.$$

Also, $(2')$ implies that, for $j < i - 1$, $X_j^{p-1}(t'') \geq X_{i-1}^{p-1}(t'')$ so $X_j^{p-1}(t'') - \mu_{i-1}^{p-1}(t'') > -\tau$. Hence we should have $X_i^p(t'') = X_i^{p-1}(t'') + 1$ (B_i refilled at step t''), where by construction, $t'' \leq t_2$. But the definition of t_2 implies that $t_2 < t'$, contradicting the definition of t'. We conclude that

$$X_i^r(t') - \mu_i^r(t') \geq -\tau \text{ for all } r < p. \tag{11}$$

We continue by showing that $X_i^{r+1}(t') - X_i^r(t') = 1$ is satisfied by $r = p-1$ and not by $r < p-1$. Suppose for a contradiction that

$$X_i^{r+1}(t') - X_i^r(t') = 1 \text{ for } r < p - 1. \tag{12}$$

From Observation 3 we have $X_i^r(t') - \mu_i^r(t') \leq 0$. This implies:

$$\begin{aligned} X_i^r(t' - 1) - \mu_i^r(t' - 1) &\leq 2k_i \\ \Rightarrow X_i^{p-1}(t' - 1) - \mu_i^{p-1}(t' - 1) &\leq 2k_i - \tfrac{p-1-r}{i}. \end{aligned}$$

But for $r < p - 1$, this contradicts (11), since the right-hand side of the above expression is less than $-\tau$:

$$2k_i - \frac{p - 1 - r}{i} \leq 2k_i - \frac{1}{i} < -\frac{1}{2i} < -\frac{1}{2n} < -\tau.$$

Case 2: $X_i^p(T - 1) - \mu_i^p(T - 1) < 0$.
We show that for some $t \in \{T, \ldots, T + \frac{J_i}{4}\}$, we have $X_i^p(t) - \mu_i^p(t) \geq 0$, so that Case 1 applies, and Case 1 then promises a t' satisfying (1) and (2) with $t' \leq T + \frac{J_i}{4} + \frac{J_i}{2}$, so we still have $t' \in \{T, \ldots, T + J_i - 1\}$.

Claim 1: Suppose that $T \leq T'' \leq J_i - \frac{2\tau}{k_i}$ and $X_i^p(T'') - \mu_i^p(T'') < 0$. Then there exists $t_2 \in \{T'', \ldots, T'' + \frac{2\tau}{k_i}\}$ such that $X_i^p(t_2) > X_i^0(t_2)$.

Proof of Claim 1: Suppose for a contradiction that no such t_2 exists.

From (8) there exists t_3 with $T'' \leq t_3 \leq T'' + \frac{\tau}{k_i}$ and

$$X_i^p(t_3) - \mu_i^p(t_3) < -\tau.$$

Applying the inductive hypothesis by plugging in t_3 for the T' in the statement of the hypothesis, there exists t'' with $t_3 \leq t'' \leq t_3 + J_{i-1}$ such that $(1')$ and $(2')$ hold so $X_{i-1}^{p-1}(t'') - \mu_{i-1}^{p-1}(t'') > -\tau$ and for $j < i$, $X_j^{p-1}(t'') \geq X_{i-1}^{p-1}(t'')$. (That last inequality follows from the rule that a bucket that is chosen to be refilled, must have minimal load.) By Observation 2 we have

$$X_i^p(t_3) - \mu_{i-1}^p(t_3) < \frac{i}{i-1}(-\tau).$$

Now by what we assumed for contradiction, that no t_2 exists that satisfies the claim, $X_i^p(t'') \leq X_i^p(t_3)$ and $\mu_{i-1}^{p-1}(t'') \geq \mu_{i-1}^{p-1}(t_3)$ so $X_i^p(t'') - \mu_{i-1}^{p-1}(t'') < \frac{i}{i-1}(-\tau)$. So $X_i^{p-1}(t'')$ is now minimal amongst $X_j^{p-1}(t'')$ (for $1 \leq j \leq i$), hence $X_i^p(t'') = X_i^{p-1}(t'') + 1$. But $t'' < T'' + \frac{\tau}{k_i} + J_{i-1}$ hence $t'' < T'' + \frac{2\tau}{k_i}$, a contradiction. This finishes the proof of Claim 1.

From (6) we know that, for $T \leq t \leq T + J_i - 2$,

$$\mu_i^p(t+1) \leq \mu_i^p(t) + k_i,$$

and from Claim 1 we know that, for $T \leq t \leq J_i - \frac{2\tau}{k_i}$, if $X_i^p(t) - \mu_i^p(t) < 0$ then there exists $t' \leq t + \frac{2\tau}{k_i}$ with $X_i^p(t') \geq X_i^0(t') + 1$. Also, $X_i^0(t+1) = X_i^p(t) - k_i$. Putting these together, if $X_i^p(t) - \mu_i^p(t) < 0$,

$$\begin{array}{rcl} X_i^p(t + \frac{2\tau}{k_i}) - \mu_i^p(t + \frac{2\tau}{k_i}) & \geq & X_i^p(t) - \mu_i^p(t) + 1 - (2k_i)(\frac{2\tau}{k_i}) \\ \Rightarrow X_i^p(t + \frac{2\tau}{k_i}) - \mu_i^p(t + \frac{2\tau}{k_i}) & \geq & X_i^p(t) - \mu_i^p(t) + \frac{1}{2}. \end{array}$$

We also know from Lemma 2 that

$$X_i^p(T) - \mu_i^p(T) \geq -(p + H(n)).$$

The number of steps needed for $X_i^p(t) - \mu_i^p(t)$ to become non-negative is upper bounded by

$$(p + H(n)) \cdot 2 \cdot \frac{2\tau}{k_i} \leq 2\tau J_i < \frac{1}{4} J_i.$$

Hence, Case 1 is recovered within $\leq \frac{1}{4} J_i$ steps. This is the end of the proof of Lemma 3. □

Lemma 4 *Consider any $i > 1$. Using the expressions for τ, k_i and J_i identified previously, suppose $X_i^p(t) - \mu_i^p(t) > 1 - \tau - 1/i$. Suppose that buckets $i+1, \ldots, n$ are not refilled during steps $t+1, \ldots, t + \sum_{j<i} J_j$. Then B_i is not refilled during these steps.*

Proof. Suppose for a contradiction that B_i is refilled at step $t+t'$ for some t' satisfying $1 \leq t' \leq \sum_{j<i} J_j$ but that B_i is not refilled during steps $t+1, \ldots, t+t'-1$. First note that $X_i^r(t+1) \geq X_i^p(t) - k_i$. Also, by Equation (6), $\mu_i^r(t+1) < \mu_i^r(t) + k_i = \mu_i^p(t) - (p-r)/i + k_i$. Putting these together, we see that

$$\begin{aligned} X_i^r(t+1) - \mu_i^r(t+1) &> X_i^p(t) - \mu_i^p(t) + \frac{p-r}{i} - 2k_i \\ &> 1 - \tau - \frac{1}{i} - 2k_i \geq \frac{3}{8} - 2k_i > 0, \end{aligned}$$

so (using Observation 3) $t' > 1$. Let $t'' = t' - 1 \geq 1$. By Equation (7), $X_i^p(t+t'') - \mu_i^p(t+t'') > 1 - \tau - 1/i - 2k_i t''$. Since $t'' < 2J_{i-1}$, we have

$$\begin{aligned} X_i^p(t+t'') - \mu_i^p(t+t'') &> 1 - \tau - \frac{1}{i} - 4k_i J_{i-1} \\ &> 1 - \tau - \frac{1}{i} - \frac{1}{4} > \frac{1}{8}. \end{aligned}$$

Then

$$\begin{aligned} X_i^r(t+t') - \mu_i^r(t+t') &> X_i^r(t+t'-1) - \mu_i^r(t+t'-1) - 2k_i + (p-r)/i \\ &= X_i^r(t+t'') - \mu_i^r(t+t'') - 2k_i + (p-r)/i \\ &> \frac{1}{8} - 2k_i + (p-r)/i > 0, \end{aligned}$$

so B_i cannot be refilled at step $t+t'$, giving a contradiction. □

Theorem 2 *For all $\lambda > 0$ there exist constant depletion rates $k_1, k_2, \ldots, k_n$ such that using scheduler $\mathcal{S}_0$, the minimum load of any bucket is less than $A - (1-\lambda)(H(n) + p - 1)$.*

Proof. To prove the theorem, start the system at time $T_{n+1} = 0$. Now, for all $i \in \{2, \ldots, n\}$, let T_i denote the smallest $t > T_{i+1}$ such that $X_i^p(T_i) - \mu_i^p(T_i) > 1 - \tau - 1/i$.

Using Lemma 3, we show by induction on i (starting from the base case, $i = n$, and working down to $i = 2$) that $T_i \leq T_{i+1} + J_i$. Consider the interval $T_{i+1}+1, \ldots, T_{i+1}+J_i$. By Lemma 4, buckets $B_{i+1}, \ldots, B_n$ are not refilled during this interval. By Lemma 3, there is a T_i in the interval with

(1) $X_i^{p-1}(T_i) - \mu_i^{p-1}(T_i) > -\tau$, and

(2) $X_i^p(T_i) = X_i^{p-1}(T_i) + 1$.

Then, since $\mu_i^p(T_i) = \mu_i^{p-1}(T_i) + 1/i$, we have $X_i^p(T_i) - \mu_i^p(T_i) > 1 - \tau - 1/i$. Since we have verified that $T_i \leq T_{i+1} + J_i$, we may use the inequality $T_2 - T_i \geq J_2 + \ldots + J_{i-1}$ in what follows.

Now using Equation (7) for any $i \in \{2, \ldots, n\}$ we have

$$\begin{aligned} X_i^p(T_2) - \mu_i^p(T_2) &> X_i^p(T_i) - \mu_i^p(T_i) - 2k_i(T_2 - T_i) \\ &> 1 - \tau - \frac{1}{i} - 2k_i(J_2 + \cdots + J_{i-1}) \\ &\geq 1 - \tau - \frac{1}{i} - 2k_i 2J_{i-1} \\ &> 1 - 2\tau - \frac{1}{i} \geq \frac{i-1}{i}(1 - 4\tau). \end{aligned}$$

Therefore, by Observation 2 (second equation),

$$\mu_i^p(T_2) - \mu_{i-1}^p(T_2) = \frac{X_i^p(T_2) - \mu_i^p(T_2)}{i-1} > \frac{1-4\tau}{i}.$$

So

$$\begin{aligned} X_n^p(T_2) - X_1^p(T_2) &= X_n^p(T_2) - \mu_1^p(T_2) \\ &= X_n^p(T_2) - \mu_{n-1}^p(T_2) + \\ &\quad (\mu_{n-1}^p(T_2) - \mu_{n-2}^p(T_2)) + \cdots + (\mu_2^p(T_2) - \mu_1^p(T_2)) \\ &\geq (1 - 4\tau)\left(1 + \sum_{j=2}^{n-1} \frac{1}{j}\right) \\ &\geq (1 - 4\tau)H(n-1). \end{aligned}$$

Thus,

$$\begin{aligned} X_n^0(T_2 + 1) - X_1^0(T_2 + 1) &= X_n^p(T_2) - k_n - (X_1^p(T_2) - k_1) \\ &\geq (1 - 4\tau)H(n-1) + k_1 - k_n \\ &= (1 - 4\tau)H(n-1) + p - \sum_{j=2}^{n-1} k_j, \end{aligned}$$

and finally, since by Observation 4 we have $X_n^0(T_2 + 1) \leq A - \frac{1}{n} + 1$,

$$\begin{aligned} X_1^0(T_2 + 1) &\leq A - \frac{1}{n} + 1 - (1 - 4\tau)H(n-1) - p + \sum_{j=2}^{n-1} k_j \\ &\leq A - (H(n) + p - 1) + (4\tau H(n-1) + 2k_2) \\ &\leq A - (1 - 5\tau)(H(n) + p - 1). \end{aligned}$$

□

3.2 Rational Depletion Rates

In this section we will show that the system behaves much more evenly if the depletion rates have a "mild" constraint, specifically that ratios between them are relatively small integers. To state the results we will assume in the following that all the k_i's are positive rationals. As before we have $k_1 + \ldots + k_n = p$, but now we assume, without loss of generality, that $k_j = pm_j/M$ for $1 \leq j \leq n$, where $M = \sum_j m_j$ and the greatest common divisor of all of the m_j's is 1. All the results so far stated in this paper also hold in this setting. Note that, using the notation of Section 2:

For all $t \geq 0$ we have $\sum_j X_j(t) = nA$ and $\sum_j X'_j(t) = nA - p$.

In the following we will show that the system is periodic. This result will be used below to give a lower bound for the outcome achieved by scheduler $\mathcal{S}_0$.

Lemma 5 *For rational values, $k_j = pm_j/M$, $1 \leq j \leq n$, the system returns to its initial state after M steps.*

Proof. During this period of M steps, any bucket B_i is refilled at most pm_i times, because otherwise it would have a load of at least $A - k_i M + (pm_i + 1) = A + 1$. This would contradict Observation 4 that no bucket will ever have a load exceeding $A - \frac{1}{n} + 1 < A + 1$. Since $\sum_j (pm_j) = pM$, each bucket B_i must be filled *exactly* $pm_i = k_i M$ times, and so the system returns to its initial state. □

Theorem 3 *Let $k_1 \geq k_2 \geq \ldots \geq k_n$ be the set of depletion rates, where $k_j = pm_j/M$, for $1 \leq j \leq n$, the* gcd *of the (integer) m_j's is 1, and $\sum_j m_j = M$.*

1. *Scheduler $\mathcal{S}_0$ achieves an outcome of at least $A - pm_1$.*
2. *If each k_j is a multiple of k_n, then scheduler $\mathcal{S}_0$ achieves an outcome of at least $A - pk_1/k_n$.*

Proof. According to Lemma 5, the system returns to its initial state after M steps, and so also at any multiple of M steps. Hence, the deviation of B_i's load from the initial value is bounded above by $k_i M \leq k_1 M = pm_1$.

If each k_j is a multiple of k_n then we have $m_n = 1$ and the result follows, since $m_1/m_n = k_1/k_n$. □

4 Conclusions and Further Work

Our bounds on deviation from fairness do not depend on t (the elapsed number of time quanta), and it may be that the largest deviation possible grows very slowly as a function of t, and that as a result better bounds could be found in terms of t. Note that quanta are typically on the order of milliseconds (sometimes even smaller), and jobs can run on the order of days. Hence t might be somewhere around 10^9, for example. For this range of t we may actually be able to get better worst-case behaviour; at least it is not ruled out by our construction of Theorem 2. An alternative approach is to obtain better bounds in terms of the ratio between largest and smallest values of the depletion rates, which in Theorem 2 is more than exponential in n.

We believe that the result for the adversarial model in which the k_i's may be chosen by an adversary, should be adaptable to a situation where the number of buckets is allowed to vary. Given an upper bound N on the number of buckets that may be present, then for a step with $n < N$ buckets present, we could have $N - n$ buckets with $k_i = 0$. It seems likely that the results showing the buckets do not fall too far below average could be made to apply to this case. Note that the result is not immediate, since the buckets with $k_i = 0$ could be refilled.

References

[1] J. Anderson and A. Srinivasan. A new look at Pfair priorities. Technical report TR00-023, Dept of Computer Science, Univ. of North Carolina, 2000.

[2] J. Anderson and A. Srinivasan. Early-release fair scheduling. In *Proceedings of the 12th Euromicro Conference on Real-Time Systems, Stockholm, Sweden*, pages 35–43, 2000.

[3] A. Bar-Noy, A. Freund, S. Landa and J. Naor. Competitive On-line Switching Policies. In *Proceedings of the 13th ACM-SIAM Symposium on Discrete Algorithms*, pages 525-534, 2002.

[4] S. Baruah, J. Gehrke, and C. G. Plaxton. Fast scheduling of periodic tasks on multiple resources. In *Proceedings of the 9th International Parallel Processing Symposium*, pages 280–288, 1996.

[5] S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D. A. Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 15: pages 600–625, 1996.

[6] A. Chandra, M. Adler, P. Goyal, and P. Shenoy. Surplus fair scheduling: A proportional-share cpu scheduling algorithm for symmetric multiprocessors. In *Proceedings of the 4th Symposium on Operating System Design and Implementation (OSDI 2000), San Diego, CA*, pages 45–58, 2000.

[7] A. Chandra, M. Adler, and P. Shenoy. Deadline fair scheduling: Bridging the theory and practice of proportionate-fair scheduling in multiprocessor servers. In *Proceedings of IEEE Real-time Technology and Applications Symposium (RTAS 2001)*, pages 3–14, 2001.

[8] K. Duda and D. Cheriton. Borrowed virtual time (bvt) scheduling: Supporting latency-sensitive threads in a general-purpose scheduler. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP'99)*, pages 261–276, 1999.

[9] R. Fleischer and H. Koga. Balances Scheduling toward Loss-Free Packet Queuing and Delay Fairness. *Algorithmica* 38: pages 363-376 (2004).

[10] P. Goyal, X. Guo, and H.M. Vin. A hierarchical cpu scheduler for multimedia operating systems. In *Proceedings of USENIX 2nd Symposium on Operating System Design and Implementation (OSDI'96)*, pages 107–121, 1996.

[11] M. Moir and S Ramamurthy. Pfair scheduling of fixed and migrating periodic tasks on multiple resources. In *Proceedings of the 20th Annual IEEE Real-Time Systems Symposium*, pages 294–303, 1999.

[12] J. Nieh and M S. Lam. The design, implementation and evaluation of smart: A scheduler for multimedia applications. In *Proceedings of the 16th ACM Symposium on Operating Systems Principles (SOSP'97)*, pages 184–197, 1997.

[13] Waldspurger and Weihl Stride scheduling: Deterministic proportional-share resource management. Technical report MIT/LCS/TM-528, 1995.

[14] A.K. Parekh and R.G. Gallager. A generalized processor sharing approach to flow control in integrated services networks - the single node case. *IEEE/ACM Transactions on Networking*, pages 344–357, 1993.

[15] S.J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proceedings of 13th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 1994)*, pages 636–646, 1994.

[16] P. Dietz and D. Sleator. Two algorithms for maintaining order in a list. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 365–372, 1987.