Scheduling Subset Tests: One-time, Continuous, and How They Relate

Edith Cohen[1,2], Haim Kaplan[2], and Yishay Mansour[1,2]
edith@cohenwang.com {haimk,mansour}@cs.tau.ac.il

[1] Microsoft Research
[2] Tel Aviv University, Israel

Abstract. A test scheduling instance is specified by a set of elements, a set of tests, which are subsets of elements, and numeric priorities assigned to elements. The schedule is a sequence of test invokations with the goal of covering all elements. This formulation had been used to model problems in multiple application domains from network failure detection to broadcast scheduling. The modeling considered both SUM_e and MAX_e objectives, which correspond to average or worst-case cover times over elements (weighted by priority), and both *one-time* testing, where the goal is to detect if a fault is currently present, and *continuous* testing, performed in the background in order to detect presence of failures soon after they occur. Since all variants are NP hard, the focus is on approximations.
We present combinatorial approximations algorithms for both SUM_e and MAX_e objectives on continuous and MAX_e on one-time schedules. The approximation ratios we obtain depend logarithmically on the number of elements and significantly improve over previous results. Moreover, our unified treatment of SUM_e and MAX_e objectives facilitates simultaneous approximation with respect to both.
Since one-time and continuous testing can be viable alternatives, we study the overhead of continuous testing, captured by the ratio of optimal one-time to continuous cover times. We establish that the worst-case ratio is $O(\log n)$, but also provide evidence, by considering Zipf distributions, that the typical ratio is lower.

1 Introduction

An instance $(E, \mathcal{S}, \boldsymbol{p})$ of a test scheduling problem is specified by a set E of elements, a set $\mathcal{S}$ of tests, where each test is a subset of elements E, and priorities p_e over elements $e \in E$. An invokation of $s \in \mathcal{S}$ tests all elements included in the set s. We seek schedules, which are sequences of tests, which cover the elements as efficiently as possible.

We distinguish between SUM_e objectives, which minimize the prioritized sum of cover times of individual elements and MAX_e objectives, which minimize the (weighted by priority) worst-off cover time of an element. Operationally, we distinguish between *continuous testing*, performed as a background process and appropriate when failures of elements may occur any time and we would like to detect the failure soon after it occurs, and *one-time testing*, where the goal is to detect if an existing fault is present by initiating a sequence of tests.

This formulation naturally extends the classic set cover problem and had been used to model problems arising in different application domains. Since all variants are NP-complete, the focus is on designing approximation algorithms.

A recently studied application is silent failure detection in networks [14, 12, 13, 6]: Elements correspond to physical or logical network elements (links, nodes, or forwarding rules in the software defined network) and tests corresponding to routing paths. Invoking a test translated to sending a probe packet. Once a failure is detected, heavyweight tools are applied to bypass or localize and correct it.

The special case of *singletons*, where each test contains a single element, was extensively studied in the context of scheduling Teletext [2], broadcast disks [1, 10, 3, 3, 3, 4], and search in unstructured p2p networks [7].

One-time schedules of subset tests with respect to SUM_e objectives were studied by Feige et al [9], who gave a 4-approximation algorithm (and matching inapproximability result) (see also [5]). We recently studied continuous schedules [6], and related stochastic and deterministic schedules and presented deterministic approximation algorithms based on derandomizing optimal memoryless schedules (memoryless schedules are a subclass of stochastic schedules which can be optimized by an LP or convex programs). Our work here builds on the results in [9, 6] which are discussed in more detail in Section 2.

Contributions: We present novel combinatorial approximation algorithms for deterministic schedules with approximation factors $O(\log^2 n)$ for MAX_e and $O(\log n)$ for SUM_e on continuous schedules and $O(\log n)$ for MAX_e on one-time schedules (Section 3 and Section 5), where $n = |E|$. These ratios significantly improve over previous results [6] with approximation ratios that depend logarithmically on the number of *tests containing an element*, which can be exponential in the number of elements. Indeed, experimentally in [6] we observed that we needed to artificially restrict the set of tests to obtain good schedules using the previous approaches.

In some contexts, including network testing, both one-time or continuous testing are applicable, and to support informed choice, we aim to understand their relation (Section 4). Clearly, the one-time optimum of an objective is never larger than the continuous optimum, and we therefore study the ratio of continuous to one-time optima. This ratio capture the overhead of continuous testing. We show that this ratio is at most logarithmic in the number of elements, for both SUM_e and MAX_e objectives. While we also show that our upper bound on the ratio are tight, in the sense that some families of instances have logarithmic ratios, we also give indications, by analyzing the ratio for Pareto distributed priorities, that in practice the ratio is typically lower.

Lastly, in Section 5 we expand on implications of our unified study, showing how to obtain continuous schedules from one-time schedules and explain how to concurrently approximate both SUM_e and MAX_e objectives.

2 Preliminaries

A *testing schedule* is a sequence $\boldsymbol{\sigma}$ of tests. The sequence is infinite for continuous testing and finite for one-time testing.

The *cover time* $T(e,t|\boldsymbol{\sigma}) = \min\{\Delta \geq 0 \mid e \in \sigma_{\Delta+t}\}$ of element e at time t by the schedule $\boldsymbol{\sigma}$ is the elapsed time (number of positions in the sequence) after position t until a test that includes e is invoked.

We follow notation from [6]. For an element e, $\mathsf{M}_t[e|\boldsymbol{\sigma}]$ is the maximum over time t of the cover time of e at time t, and $\mathsf{E}_t[e|\boldsymbol{\sigma}]$ is the (limit of) the average over time t of the cover time of e at time t. For a time t, $\mathsf{M}_e[t|\boldsymbol{\sigma}] = \max_e p_e T(e,t|\boldsymbol{\sigma})$ is the (weighted) maximum over elements and $\mathsf{E}_e[t|\boldsymbol{\sigma}] = \sum_e p_e T(e,t|\boldsymbol{\sigma})$ is the weighted sum over the elements of the cover time of e at t. The weighting, in both cases, is according to the priorities $\boldsymbol{p}$. When clear from context, we omit the reference to the schedule $\boldsymbol{\sigma}$ in the notation.

We study two natural objectives: *MAX*$_e$, which aim to minimize the (weighted) maximum over elements and *SUM*$_e$, which aim to minimize a weighted sum over elements. For convenience, with MAX$_e$ objectives we assume priorities are scaled so that the maximum entry is 1 and with SUM$_e$, they are normalized so that the sum of entries is 1. With this normalization, when $\boldsymbol{p}$ is a probability distribution over elements, SUM$_e$ is the expected time to cover an element that is selected according to the distribution. For concreteness, we use the fault detection application for describing objectives in the sequel. We append the prefix opt to an objective to denote the optimum of the objective on the instance.

A schedule is *stochastic*, when the sequence is a random variable. With stochastic schedules, we redefine $T(e,t|\boldsymbol{\sigma})$ to be the *expected* number of steps until e is covered [6].

2.1 One-time testing

One-time testing checks for presence of a failed element. The schedule is executed until either a test detects the presence of a faulty element or to termination, if no fault is present. For element e, $T(e,0|\boldsymbol{\sigma}) = \min\{j \mid e \in \sigma_j\}$ is the cover time of e. The one-time SUM$_e$ and MAX$_e$ are

$$\begin{aligned} \text{SUM}_e\text{:} \quad &= \sum_e p_e T(e,0|\boldsymbol{\sigma}) \\ \text{MAX}_e\text{:} \quad &= \max_e p_e T(e,0|\boldsymbol{\sigma}) \; . \end{aligned}$$

An optimal deterministic one-time schedule never performs a particular test more than once, since only the first occurrence is significant and other occurrences, if any, can only extend the coverage time of yet uncovered elements. Moreover, each test should contain at least one previously uncovered element, and therefore, an optimal schedule has length at most $n = |E|$. Moreover, there is no advantage in using a stochastic schedule, because the expected times $\mathsf{E}_e[0|\boldsymbol{\sigma}]$ or $\mathsf{M}_e[0|\boldsymbol{\sigma}]$ of a stochastic schedule are the expectation of the objective over the corresponding distribution of deterministic schedules, so there is always a deterministic schedule with objective that is at most that expectation.

Singleton instances are fully specified by the assignment of priorities $\boldsymbol{p}$ to elements (tests). Both objectives $\mathsf{E}_e[0]$ and $\mathsf{M}_e[0]$ are minimized by the schedule that tests elements in order of decreasing priority p_i. Assuming elements are indexed by decreasing priority $p_1 \geq \cdots \geq p_n$, the optimal cover times are

$$\text{opt-}\mathsf{E}_e[0](\boldsymbol{p}) = \sum_{i=1}^{n} i p_i \quad (1)$$

$$\text{opt-}\mathsf{M}_e[0](\boldsymbol{p}) = \max_{i\in[n]} i p_i \ . \quad (2)$$

Subset tests: We summarize previous results for $\mathsf{E}_e[0]$ and $\mathsf{M}_e[0]$, which establish NP hardness and approximability.

SUM$_e$: For $\mathsf{E}_e[0]$, a simple greedy algorithm which sequentially selects the test that covers a set of uncovered elements with maximum sum of priorities has opt-$\mathsf{E}_e[0]$ that is at most 4 times the optimal [9] (see also [5]). The problem of minimizing $\mathsf{E}_e[0]$ (or approximating within factor of $4-\epsilon$ for any positive $\epsilon > 0$) is NP hard [9].

MAX$_e$: When priorities are uniform, optimizing $\mathsf{M}_e[0]$ is equivalent to computing a minimum set cover: The optimal $\mathsf{M}_e[0]$ is the size of the minimum cover. From hardness of approximation of set cover, $\mathsf{M}_e[0]$ is hard to approximate within anything better than a $\ln n$ ratio [8]. When priorities are uniform, the greedy set cover algorithm guarantees a schedule with approximation ratio of $\ln n$ for $\mathsf{M}_e[0]$.

2.2 Continuous testing

We summarize our model and relevant results from[6]. The distinction between stochastic and deterministic schedules is important with continuous testing, as stochastic cover times can be lower. Optimizing MAX$_e$ and SUM$_e$ over either stochastic or deterministic schedules is NP-hard. We defined, however, a subclass of stochastic schedules, which we named *memoryless schedules.* Memoryless schedules are specified by a distribution $\boldsymbol{q}$ on tests, so that at each time, the invoked test is selected according to $\boldsymbol{q}$ (independently of history). For a schedule/distribution $\boldsymbol{q}$, $\text{SUM}_e[\boldsymbol{q}] = \sum_e p_e/Q_e$ and $\text{MAX}_e[\boldsymbol{q}] = \max_e p_e/Q_e$, where $Q_e = \sum_{i|e\in s_i} q_i$. Optimal memoryless cover times, with respect to either SUM$_e$ and MAX$_e$, are at most twice the optimal stochastic ones. Moreover, optimal memoryless schedules can be computed efficiently, via Linear Programs (MAX$_e$) or convex programs (SUM$_e$). With continuous schedules, we use opt without subscript for the optimum of the objective over stochastic schedules, whereas opt$_M$ or opt$_D$, respectively, denotes a restriction to deterministic or memoryless (special case of schedules.

For singleton instances, the optimal memoryless schedule has frequencies $q_e \propto p_e$ ($q_i = p_i/\sum_e p_e$) to optimize MAX$_e$ and $q_e \propto \sqrt{p_e}$ ($q_i = \sqrt{p_i}/\sum_e \sqrt{p_e}$) to optimize SUM$_e$ [11]. The respective optima are

$$\text{opt}_M\text{-MAX}_e[\boldsymbol{p}] = \max_e \frac{p_e}{q_e} = \sum_e p_e \quad (3)$$

$$\text{opt}_M\text{-SUM}_e[\boldsymbol{p}] = \sum_e \frac{p_e}{q_e} = (\sum_e \sqrt{p_e})^2 \ . \quad (4)$$

With deterministic schedules, we further distinguish objectives within each of SUM$_e$ and MAX$_e$, according to the dependence on time. There are three SUM$_e$ objective, which from strongest to weakest are $\mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}]$, the weighted sum over elements e of the maximum over time t of detection time $T(e,t)$, $\mathsf{M}_t\mathsf{E}_e[\boldsymbol{\sigma}]$, the maximum over time of the weighted sum over e, and $\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]$, the weighted sum over elements of the average over time. There are also three MAX_e objectives, which in order of strongest to weakest are $\mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}]$, the weighted maximum over elements of the maximum over time of the detection time. $\mathsf{M}_e\mathsf{E}_t[\boldsymbol{\sigma}]$, the weighted maximum over elements of the average over time, and $\mathsf{E}_t\mathsf{M}_e[\boldsymbol{\sigma}]$, the average over time of the maximum element at that time.

The optimum of all deterministic objectives is always at least $1/2$ of the respective memoryless optimum but the two stronger objectives in each category are at least the memoryless optimum. For the stronger objectives, there are asymptotically large ratios of the deterministic to memoryless optima.

Theorem 1. *[6] Given a memoryless schedule specified by frequencies* $\boldsymbol{q}$*, we can efficiently construct a deterministic schedule* $\boldsymbol{\sigma}$ *with either*

$$\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] = SUM_e[\boldsymbol{q}] \qquad \text{and} \qquad \mathsf{E}_t\mathsf{M}_e[\boldsymbol{\sigma}] = MAX_e[\boldsymbol{q}]$$
$$\mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}] = O(\log \ell) SUM_e[\boldsymbol{q}] \qquad \text{and} \qquad \mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] = O(\log n + \log \ell) MAX_e[\boldsymbol{q}]$$

where ℓ *is the maximum over elements of the number of tests which include the element.*

As we noted in the introduction, ℓ can be exponential in the number of elements n. Our focus in this paper is on the stronger deterministic objectives, the $\mathsf{E}_e\mathsf{M}_t$ and the $\mathsf{M}_e\mathsf{M}_t$, for which we present approximation algorithms with logarithmic dependence on n rather than ℓ. We use the following in our constructions of continuous schedules:

Lemma 1. *[3, 6] For given frequencies* $\boldsymbol{q}$*, we can always construct a deterministic schedule so that the interval between invokations of test* i *is at most* $2/q_i$*.* □

The deterministic schedule is obtained by rounding frequencies down to integral powers of 2: $q'_i \leftarrow 2^{-\lceil \log_2 q_i \rceil}$. A set of frequencies that are integral powers of 2 that sum to at most 1 can be optimally scheduled so that each test is invoked with a period of at most $1/q_i$ [3, 6].

3 MAX_e schedulers

We present a $O(\log n)$ approximation for one-time MAX_e scheduling and $O(\log^2 n)$ approximation for continuous deterministic $\mathsf{M}_e\mathsf{M}_t$ scheduling. Both algorithms use the same partition over the elements: Assuming priorities are scaled so that the largest priority is equal to 1, elements are partitioned according to the value of $\lfloor -\log_2 p_e \rfloor$, so that the set E_i for $i \geq 0$ contains all elements for which $\lfloor -\log_2 p_e \rfloor = i$. We then compute a (greedy) set cover C_i for each set E_i. Pseudo code for computing the partition and covers is in Algorithm 1.

The one-time final schedule $\boldsymbol{\sigma}$ is a concatenation of the set covers C_i by increasing $i \geq 0$. See ONETIMEMAXSCHEDULE in Algorithm 2 for pseudocode.

Theorem 2. *Consider the one-time schedule* $\boldsymbol{\sigma}$ *computed by* ONETIMEMAXSCHEDULE *when the covers in* PARTITIONP2 *are computed using the greedy set cover algorithm. Then*

$$\mathsf{M}_e[0|\boldsymbol{\sigma}] \leq O(\ln |n|) opt\text{-}\mathsf{M}_e[0] \ .$$

Algorithm 1 Partition elements by powers-of-2

function PARTITIONP2($\boldsymbol{p}, E$)
  $\boldsymbol{p} \leftarrow \boldsymbol{p}/\max(\boldsymbol{p})$ ▷ Scale $\boldsymbol{p}$ so that the maximum priority is 1.
  for $i \geq 0$ **do**
    $E_i \leftarrow \{e \mid p_e \in (2^{-(i+1)}, 2^{-i}]\}$ ▷ Partition E according to priorities
  $U \leftarrow \emptyset$
  for $i \geq 0$ **do**
    $E_i \leftarrow E_i \setminus U$ ▷ remove elements already covered by higher-priority tests
    $C_i \leftarrow$SET-COVER$(E_i, \mathcal{S})$
    $U \leftarrow U \cup \{$elements covered by $C_i\}$
  return $\boldsymbol{p}, \boldsymbol{E}, \boldsymbol{C}$

Proof. We first upper bound the optimum:

$$\text{opt-}\mathsf{M}_e[0] \leq \quad \mathsf{M}_e[0|\boldsymbol{\sigma}] \leq \max_e p_e \sum_{j \leq \lfloor -\log_2 p_e \rfloor} |C_i| \leq \quad \max_{i \geq 0} 2^{-i} \sum_{j \leq i} |C_i| \tag{5}$$

We now lower bound the optimum:

$$\begin{aligned} \text{opt-}\mathsf{M}_e[0] &\geq \max_e p_e |\text{OPT-COVER}\{h \in E \mid p_h \geq p_e\}| \\ &\geq \max_i 2^{-(i+1)} \max_{j \leq i} |\text{OPT-COVER}\{E_j\}| \geq \max_{i \geq 0} 2^{-(i+1)} |\text{OPT-COVER}\{E_i\}| \quad (6) \\ &\geq \max_{i \geq 0} 2^{-(i+1)} \frac{|C_i|}{\ln |E_i|} \geq \frac{1}{2 \ln n} \max_{i \geq 0} 2^{-i} |C_i| \quad (7) \end{aligned}$$

To verify (7), note that a lower bound on opt-$\mathsf{M}_e[0]$ is the maximum over elements e of the product of p_e by the size of the minimal set cover of all elements with priority at most p_e. For $e \in E_i$, this is lower bounded by $2^{-(i+1)}$ (the lowest possible priority of a member of E_i) times the size of the minimum set cover of n_i, which is lower bounded in turn by the size of the greedy cover $|C_i|$ divided by the worst-case approximation ratio $\ln |E_i|$.

Combining (5) and (7), to conclude the proof it suffices to establish

$$\max_i 2^{-i} \sum_{j \leq i} |C_j| \leq 2 \max_i 2^{-i} |C_i| \; . \tag{8}$$

Let k be i which maximizes $2^{-i} \sum_{j \leq i} |C_j|$. From our choice of k,

$$2^{-k} \sum_{j \leq k} |C_j| \geq 2^{-k+1} \sum_{j \leq k-1} |C_j| \; . \tag{9}$$

We expand and substitute (9) to obtain

$$\begin{aligned} 2^{-k} \sum_{j \leq k} |C_j| &= (1/2) \left(2^{-k+1} \sum_{j \leq k-1} |C_j| \right) + 2^{-k} |C_k| \\ &\leq (1/2) \left(2^{-k} \sum_{j \leq k} |C_j| \right) + 2^{-k} |C_k| \quad (10) \end{aligned}$$

Therefore,

$$2^{-k}\sum_{j\le k}|C_j| \le 2^{-k+1}|C_k| \implies |C_k| \ge \sum_{j\le k-1}|C_j| \ . \tag{11}$$

We are now ready to establish (8), using (11):

$$\max_i 2^{-i}\sum_{j\le i}|C_j| = 2^{-k}\sum_{j\le k}|C_j| \le 2\cdot 2^{-k}|C_k| \le 2\max_i 2^{-i}|C_i| \ .$$

□

Algorithm 2 One-Time and Continuous schedules for MAX_e

```
1: function OneTimeMaxSchedule(p, E)
2:     (p, E, C) ←PartitionP2(p, E)
3:     σ ← C_1, C_2, ...
4:     return σ
5: function ContMaxSchedule(p, E)
6:     (p, E, C) ←PartitionP2(p, E)
7:     for i ≥ 0 do
8:         for s ∈ C_i do
9:             q[s] ← 2^{-i}
10:    return ContSingletonSchedule (q / Σ_i q[i])    ▷ Return a schedule according to
       specified frequencies as in Lemma 1
```

To obtain a continuous schedule $\boldsymbol{\sigma}$ (Pseudocode ContMaxSchedule in Algorithm 2), we first compute the partition and covers (PartitionP2 in Algorithm 1, using the greedy set cover algorithm). For all i, we assign frequencies 2^{-i} to the tests participating in the cover C_i and normalize so that the sum of frequencies is 1. The schedule is obtained by applying Lemma 1.

Theorem 3. *The schedule $\boldsymbol{\sigma}$ computed by* ContMaxSchedule *satisfies*

$$\mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] \le O(\ln^2 n)opt_D\text{-}\mathsf{M}_e\mathsf{M}_t \ .$$

The proof of the Theorem uses the following Lemma:

Lemma 2.

$$\mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] \le 2\sum_{j\ge 0}2^{-j}|C_j| \ , \tag{12}$$

where $\boldsymbol{C}$ is the set of covers returned by PartitionP2 *(Algorithm 1)*

Proof. Consider the normalization of $\boldsymbol{q}$ in line 10. The sum of q_i before normalization is $\sum_i 2^{-i}|C_i|$, and thus the final frequency of tests in C_i are $\frac{2^{-i}}{\sum_i 2^{-i}|C_i|}$. Consider an element $e\in E_i$. It is covered by a test s in C_j for some $j\le i$ with frequency at least $q[s]\equiv 2^{-i}/\sum_i 2^{-i}|C_i|$. The schedule $\boldsymbol{\sigma}$ invokes s at least every $2q[s]$ steps (Lemma 1). Therefore, $\mathsf{M}_t[e]\le 2p_e/q[s]\le 2\cdot 2^{-i}\sum_{j\ge 0}2^{-j}|C_j|/2^{-i} = 2\sum_{j\ge 0}2^{-j}|C_j|$. □

We can now proceed to the proof of Theorem 3. The optimum $\text{opt}_D\text{-}\mathsf{M}_e\mathsf{M}_t$ is lower bounded by the smallest priority in the set E_i times the size of the optimal set cover of E_i. We obtain the same lower bound we used for one-time schedules (7) in the proof of Theorem 2:

$$\begin{aligned}\text{opt}_D\text{-}\mathsf{M}_e\mathsf{M}_t &\geq \max_{i\geq 0} 2^{-(i+1)}|\text{OPT-COVER}\{E_i\}| \\ &\geq \max_{i\geq 0} 2^{-(i+1)}|C_i|/\ln(|E_i|) \geq \frac{1}{2\ln(n)} \max_{i\geq 0} 2^{-i}|C_i| \,. \qquad (13)\end{aligned}$$

By combining the upper bound in Lemma 2 and the lower bound (13), we obtain that to establish the approximation ratio of $O(\log^2 n)$, it suffices to show that for some fixed constant k,

$$\sum_{j\geq 0} 2^{-j}|C_j| \leq k\log(n) \max_{j\geq 0} 2^{-j}|C_j| \,. \qquad (14)$$

To establish (14), we consider the sequence $|C_i|$, marking selected positions. We mark C_0 and then mark C_i if $|C_i| > 1.5|C_j|$, where C_j is the previously marked item. The number of marked positions is $\leq \log_{1.5} n$. This is because for all i, $|C_i| \leq |E_i| \leq n$. Consider now two consecutive marked items C_h and $C_{h'}$ where $h' > h$. We have that $|C_j| \leq 1.5|C_h|$ for every $j \in [h, h')$. Therefore, $\sum_{j=h}^{h'-1} 2^{-j}|C_j| \leq 3 \cdot 2^{-h}|C_h|$. Summing over the entire sequence we get that

$$\sum_{j\geq 0} 2^{-j}|C_j| \leq 3\log_{1.5} n \max_j 2^{-j}|C_j| \leq 6\log_2 n \max_j 2^{-j}|C_j| \,. \qquad (15)$$

4 Relating One-time and Continuous Testing

We study the ratio of optimal continuous to optimal one-time cover times and provide both upper and lower bounds, for both the SUM_e and MAX_e objectives. We show that the stochastic optimum is within $O(\ln m)$ of the one-time optimum, where m is the number of tests in the optimal one-time schedule. Therefore, the upper bounds holds also for the weaker memoryless opt_M and deterministic optima opt_D.

Our lower bounds use a family of instances where instance I_m has m tests, where the ratio of opt_M to the one-time optimum for instance I_m is H_m, where $H_i = \sum_{j=1}^{i} 1/j$ is the ith Harmonic number. This implies a logarithmic lower bound on the ratio also for the stochastic and deterministic objectives.

We first observe that on a given instance, the optimal one-time cover time can be at most the respective optimum by a continuous schedule:

Lemma 3. *On any instance $I = (E, \mathcal{S}, \boldsymbol{p})$,*

$$\textit{opt-}\mathsf{E}_e[0](I) \leq \textit{opt-SUM}_e(I)$$
$$\textit{opt-}\mathsf{M}_e[0](I) \leq \textit{opt-MAX}_e(I)$$

Proof. Draw an optimal continuous stochastic schedule and generate a deterministic one-time schedule by running it starting at t. With at least a fixed probability, on most drawings/times t, the values $\mathsf{E}_e[t]$ (respectively $\mathsf{M}_e[t]$) are at most their expectation ($\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{E}_t\mathsf{M}_e$, respectively). □

4.1 Ratio for SUM$_e$

Theorem 4. *On any instance $I = (E, \mathcal{S}, \boldsymbol{p})$,*

$$opt_M\text{-}SUM_e(I) \leq \ln(m) opt\text{-}\mathsf{E}_e[0](I) \quad (16)$$
$$opt_D\text{-}\mathsf{E}_e\mathsf{M}_t(I) \leq 2\ln(m) opt\text{-}\mathsf{E}_e[0](I) \quad (17)$$

where m is the number of tests in the optimal one-time schedule. Moreover, there is a family of instances I_i ($i \geq 1$), where instance I_i has i tests, for which

$$\frac{opt_M\text{-}SUM_e(I_m)}{opt\text{-}\mathsf{E}_e[0](I_m)} = \ln(m) + O(1) .$$

Proof. Given a one-time schedule $\boldsymbol{\sigma} = s_1, \ldots, s_m$ with m tests, we construct a memoryless schedule $\boldsymbol{q}$ so that SUM$_e[\boldsymbol{q}]$ is at most $\ln m$ times $\mathsf{E}_e[0|\boldsymbol{\sigma}]$. The memoryless schedule $\boldsymbol{q}$ invokes test s_i with frequency $q_i = \frac{1}{iH_m}$. For any element e,

$$\mathsf{M}_t[e|\boldsymbol{q}] = \frac{1}{\sum_{i|e\in s_i} q_i} \leq \frac{1}{q_{\min\{i|e\in s_i\}}} = H_m min\{i|e \in s_i\} .$$

To establish (16), we can see that SUM$_e[\boldsymbol{q}]$, which must be at least opt$_M$-SUM$_e$, is

$$\text{SUM}_e[\boldsymbol{q}] = \sum_e p_e \mathsf{M}_t[e|\boldsymbol{q}] \leq H_m \sum_e p_e \min\{i|e \in s_i\} = H_m \mathsf{E}_e[0|\boldsymbol{\sigma}] \leq \ln(m)\mathsf{E}_e[0|\boldsymbol{\sigma}] .$$

To establish (17), we construct a deterministic continuous schedule $\boldsymbol{\sigma}'$ by applying Lemma 1 with respect to frequencies q_i for s_i. The resulting schedule invokes s_i at least every $2^{-\lceil \log_2(iH_m)\rceil}$ steps. We obtain that for any e,

$$\mathsf{M}_t[e|\boldsymbol{\sigma}'] \leq 2/\max_{i|e\in s_i} q_i \leq 2/q_{\min\{i|e\in s_i\}} = 2H_m \min\{i|e \in s_i\} .$$

Therefore,

$$\mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}'] = \sum_e p_e \mathsf{M}_t[e|\boldsymbol{\sigma}'] \leq 2\sum_e p_e H_m \min\{i|e \in s_i\} = 2H_m \mathsf{E}_e[0|\boldsymbol{\sigma}] .$$

We now establish the second claim. For each $m > 1$, we construct a singletons instance I_m with m tests/elements with priorities $p_i = \frac{1}{i^2}\frac{1}{S_m}$, where $S_m = \sum_{j=1}^m 1/i^2 \leq \pi^2/6$. The optimum $\mathsf{E}_e[0]$ for this instance is attained by invoking tests by decreasing p_i and according to (1), has:

$$\text{opt-}\mathsf{E}_e[0](I_m) = \sum_i ip_i = H_m/S_m . \quad (18)$$

The optimal memoryless SUM$_e$ for I_m has square-root frequencies (4) $q_i = 1/(iH_m)$:

$$\text{opt}_M\text{-SUM}_e(\boldsymbol{p}) = (\sum_e \sqrt{p_e})^2 = H_m^2/S_m . \quad (19)$$

Combining (18) and (19), we get the relation $\frac{\text{opt}_M\text{-SUM}_e(I_m)}{\text{opt-}\mathsf{E}_e[0](I_m)} = H_m$. $\square$

4.2 Ratio for MAX$_e$

Theorem 5. *On any instance $I = (E, \mathcal{S}, \boldsymbol{p})$,*

$$\textit{opt}_D\text{-}\mathsf{M}_e\mathsf{M}_t(I) \leq O(\ln(m))\textit{opt-}\mathsf{M}_e[0](I) \tag{20}$$

where m is the number of tests in the optimal one-time sequence. Moreover, there is a family of instances I_i ($i \geq 1$), where instance I_i has i tests, for which

$$\frac{\textit{opt}_M\textit{-MAX}_e(I_m)}{\textit{opt-}\mathsf{M}_e[0](I_m)} = \ln(m) + O(1) .$$

Proof. Consider the output of PARTITIONP2 (Algorithm 1) when used with an optimal set cover subroutine. From (6), we obtain the lower bound:

$$\text{opt-}\mathsf{M}_e[0] \geq \max_e p_e|\text{OPT-COVER}\{h \in E \mid p_h \geq p_e\}| \geq \max_i 2^{-(i+1)}|C_i| . \tag{21}$$

Consider a continuous schedule $\boldsymbol{\sigma}$ computed by CONTMAXSCHEDULE (Algorithm 2) when PARTITIONP2 (Algorithm 1) is used with an optimal set cover subroutine. From Lemma 2, we have

$$\mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] \leq 2\sum_{j\geq 0} 2^{-j}|C_j|$$

Using (15) we have

$$\frac{\text{opt}_D\text{-}\mathsf{M}_e\mathsf{M}_t}{\text{opt-}\mathsf{M}_e[0]} = \frac{2\sum_{j\geq 0} 2^{-j}|C_j|}{\max_i 2^{-(i+1)}|C_i|} \leq 6\log_2 n .$$

which establishes claim (20).

We construct a family of singletons instances, where instance I_n has n elements/tests, where element i has priority $p_i = 1/i$. The optimal one-time schedule includes tests by decreasing priority p_i and according to (2) has opt-$\mathsf{M}_e[0](I_n) = \max_i ip_i = 1$. The optimal memoryless schedule uses $q_i \propto p_i$ and from (3) has opt$_M$-MAX$_e(I_n) = \sum_i p_i = H_n$. □

4.3 Singletons with Pareto Priorities

We study the ratio for singleton instances with Pareto priorities. The instance $I_{m,\alpha}$ is specified by the number of elements/tests m and the parameter α, where the priority of element i is $p_i \propto i^{-\alpha}$.

We established that a ratio of $\ln(m) + O(1)$ for SUM$_e$ is attained with $\alpha = 2$ and for MAX$_e$ with $\alpha = 1$.

We first consider SUM$_e$ for $\alpha \neq 2$, using the expressions (1) and (4) for the one-time and memoryless optima:

$$\frac{\text{opt}_M\text{-SUM}_e(I_{m,\alpha})}{\text{opt-}\mathsf{E}_e[0](I_{m,\alpha})} = \frac{(\sum_i \sqrt{p_i})^2}{\sum_i ip_i} \approx \frac{(\int_1^m x^{-\alpha/2}dx)^2}{\int_1^m x^{1-\alpha}dx}$$
$$= \frac{(\frac{2}{2-\alpha}(m^{1-\alpha/2}-1))^2}{\frac{1}{2-\alpha}(m^{2-\alpha}-1)} = \frac{4}{2-\alpha}\frac{m^{2-\alpha}+1-2m^{1-\alpha/2}}{m^{2-\alpha}-1} .$$

The ratio is asymptotically $4/(2-\alpha)$ when $\alpha < 2$ and $4/(\alpha-2)$ for $\alpha > 2$.

We similarly consider MAX_e for $\alpha = 1$, using the expressions (2) and (3) for the one-time and memoryless optima:

$$\frac{\text{opt}_M\text{-MAX}_e(I_{m,\alpha})}{\text{opt-}\mathsf{M}_e[0](I_{m,\alpha})} = \frac{\max_i ip_i}{\sum_e p_e} \approx \frac{\max_i i^{1-\alpha}}{\int_1^m x^{-\alpha}dx} = (1-\alpha)\frac{\max_i i^{1-\alpha}}{m^{1-\alpha}-1}$$

The one-time optimum is $\max_i i^{1-\alpha} = 1$ (realized for $i = 1$) when $\alpha > 1$, and is $\max_i ip_i = m^{1-\alpha}$ (realized for $i = m$) when $\alpha > 1$. The memoryless (continuous) optimum is $(m^{1-\alpha}-1)/(1-\alpha)$ for $\alpha < 1$ and $\approx 1/(\alpha-1)$ for $\alpha > 1$. Combining the ratio, asymptotically, is $\approx \alpha - 1$ for $\alpha > 1$ and $\approx 1-\alpha$ for $\alpha < 1$.

Interestingly, the SUM_e ratio is constant for all $\alpha \neq 2$ and the MAX_e ratio is constant for $\alpha \neq 1$.

5 More on continuous scheduling

5.1 SUM_e continuous schedulers

The proof of (17) in Theorem 4 provides a construction of a continuous schedule from a one-time schedule, so that the $\mathsf{E}_e\mathsf{M}_t$ of the resulting schedule is at most $O(\log n)$ times $\mathsf{E}_e[0]$ of the one-time schedule. Combining this with one-time optimal cover times being at most the respective continuous ones, and with existence of4-approximate one-time schedulers [9, 5], we obtain the following:

Lemma 4. *There is an $O(\log n)$-approximation algorithm for $\mathsf{E}_e\mathsf{M}_t$ deterministic scheduling.*

To obtain the continuous schedule, we first construct a 4-approximate one-time schedule [9, 5] with m tests. We then assign frequency $1/(iH_m)$ to the ith test. Lastly, we construct a deterministic schedule according to these frequencies using Lemma 1.

5.2 Choose-ℓ continuous testing

To better understand our continuous schedulers, the one in Lemma 4 for SUM_e and ContMaxSchedule (in Algorithm 2) for MAX_e, we define a natural restriction of continuous scheduling, where each element has to commit to at most ℓ of the tests which include it. Only the selected tests may cover e at run time. Clearly, choose-ℓ optimum is at most the choose-h optimum when $\ell > h$. Continuous scheduling as we defined it is choose-∞ (or choose-$|\mathcal{S}|$) and the most restricted is choose-1. We now note that our results on the approximation ratio of the schedulers and the ratio between optimal continuous and one-time schedules actually hold for choose-1 continuous testing. This implies at most an $O(\log n)$ ratio between the optima of choose-1 and choose-∞ continuous testing. We present a family of instances where the ratio is $\Omega(\log n)$, showing that this is tight. Our instances correspond to complete binary trees, with elements corresponding to nodes and each tests to a root to leaf paths. Each path is labeled by the bit string of the position of the leaf. The priority of element at level i is $\propto 2^{-i}$. The optimal choose-∞ schedule chooses paths in reverse bit order of the leaf labels. This schedule covers a level i node every 2^i steps and optimizes both SUM_e and MAX_e. The choose-1 optimum, however, must be logarithmically larger.

5.3 Improved $\mathsf{E}_e\mathsf{M}_t$ scheduler

We improve on the SUM$_e$ scheduler in Lemma 4 by optimally assigning test frequencies for the underlying choose-1 assignment $S(e)$ of elements to tests, which assigns each element to the first test in the one-time schedule which covers it. We argue that for a given mapping S, the frequency distribution on tests which optimizes choose-1 $\mathsf{E}_e\mathsf{M}_t$ is $q_i \propto \sum_{e|S(e)=i} p_e$. To see that, recall that we want to minimize $\sum_e p_e/q_{S(e)}$. If we define $p_i = \sum_{e|S(e)=i} p_e$, this is the same a singleton scheduling for p_i which is optimized by square-root frequencies (4).

5.4 Combinations of objectives

Our schedulers can be adjusted to concurrently approximate MAX$_e$ and SUM$_e$ objectives. With one-time testing, we can simply interleave the two schedules which results in at most a factor of 2 loss in the approximation quality. With continuous schedules, we need to slightly adjust our algorithms to achieve that: Recall that our continuous schedulers for MAX$_e$ and SUM$_e$ associate frequencies with tests and construct a schedule from these frequencies. With two objectives, we take the test-wise maximum frequency (and renormalize). This again results in loosing at most a factor of two in the approximation of each objective.

References

1. S. Acharya, R. Alonso, M. Franklin, and S. Zdonik. Broadcast disks: data management for asymmetric communication environments. In *ACM SIGMOD*, 1995.
2. M. Ammar and J. Wong. On the optimality of cyclic transmission in teletext systems. *IEEE Tran. Communication*, 35(1):68–73, 1987.
3. A. Bar-Noy, R. Bhatia, J. Naor, and B. Schieber. Minimizing service and operation costs of periodic scheduling. *Math. Oper. Res.*, 27(3):518–544, 2002.
4. A. Bar-Noy, V. Dreizin, and B. Patt-Shamir. Efficient algorithms for periodic scheduling. *Computer Networks*, 45(2):155–173, 2004.
5. E. Cohen, A. Fiat, and H. Kaplan. Efficient sequences of trials. In *Proc. 14th ACM-SIAM Symposium on Discrete Algorithms*, 2003.
6. E. Cohen, A. Hassidim, H. Kaplan, Y. Mansour, D. Raz, and Y. Tzur. Probe scheduling for efficient detection of silent failures. Technical Report cs.NI/1302.0792, arXiv, 2013.
7. E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *Proceedings of the ACM SIGCOMM'02 Conference*, 2002.
8. U. Feige. A threshold of $\ln n$ for approximating set cover. *J. Assoc. Comput. Mach.*, 45:634–652, 1998.
9. U. Feige, L. Lovasz, and P. Tetali. Approximating min-sum set cover. In *Proceedings of 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX)*, volume 2462 of *LLNCS*, pages 94–107. Springer, 2002.
10. S. Hameed and N. H. Vaidya. Log-time algorithms for scheduling single and multiple channel data broadcast. In *Proc. of ACM/IEEE MobiCom*, 1997.
11. L. Kleinrock. *Queueing Systems, Volume II: Computer Applications*. Wiley-Interscience, New York, 1976.
12. H. X. Nguyen, R. Teixeira, P. Thiran, and C. Diot. Minimizing probing cost for detecting interface failures: Algorithms and scalability analysis. In *INFOCOM*, 2009.
13. H. Zeng, P. Kazemian, G. Varghese, and N. McKeon. Automatic test packet generation. In *CONEXT*, 2012.
14. Q. Zheng and G. Cao. Minimizing probing cost and achieving identifiability in probe based network link monitoring. *IEEE Tran. Computers*, 2012. to appear.