Probe Scheduling for Efficient Detection of Silent Failures

Edith Cohen[1,2], Avinatan Hassidim[3,4], Haim Kaplan[2], Yishay Mansour[2], Danny Raz[5,6], Yoav Tzur[3]

Abstract

Most discovery systems for silent failures work in two phases: a continuous monitoring phase that detects presence of failures through probe packets and a localization phase that pinpoints the faulty element(s). We focus on the monitoring phase, where the goal is to balance the probing overhead with the cost associated with longer failure detection times.

We formulate a general model for the underlying fundamental subset-test scheduling problem. We unify the treatment of schedulers and cost objectives and make several contributions: We propose *Memoryless schedules* – a natural subclass of stochastic schedules which is simple and suitable for distributed deployment. We show that the optimal memoryless schedulers can be efficiently computed by convex programs (for SUM objectives, which minimize average detection time) or linear programs (for MAX objectives, which minimize worst-case detection time), and surprisingly perhaps, are guaranteed to have expected detection times that are not too far off the (NP hard) stochastic optima. We study *Deterministic schedules*, which provide a guaranteed bound on the maximum (rather than expected) cost of undetected faults, but like general stochastic schedules, are NP hard to optimize. We develop novel efficient deterministic schedulers with provable approximation ratios.

Finally, we conduct an experimental study, simulating our schedulers on real networks topologies, demonstrates a significant performance gains of the new memoryless and deterministic schedulers over previous approaches.

1. Introduction

Prompt detection of failures of network elements is a critical component of maintaining a reliable network. Silent failures, which are not announced by the failed elements, are particularly challenging and can only be discovered by active monitoring.

Failure identification systems [1, 2, 3, 4] typically work in two phases: First detecting presence of a failure and then localizing it. The rational behind this design is that detection is an easier problem than localization and requires light weight mechanisms that have little impact on network performance. Once the presence of a failure is confirmed, more extensive tools which may consume more resources are deployed for localizing the failure. Moreover, in some cases, it is possible to bypass the problem, by rerouting through a different path, quicker than the time it takes to pinpoint or correct the troubled component.

A lightweight failure detection mechanism, which relies on the existing infrastructure, uses probe packets or *probes* that are sent from certain hosts or between origin destination (OD) pairs along the existing routing infrastructure (see Figure 1). The elements we monitor can be physical links [1], combination of components and paths [2], or logical components of network elements like the forwarding rules in the switches of a software-defined network [4]. If one of the elements on the probe path fails, the probing packet will not reach the destination, and in this case the probe has detected a failure. Therefore, each probe (test) type can detect if at least one element in a subset of elements had failed. Moreover, since network paths can overlap, the subsets of elements associated with different tests may overlap.

Email addresses: editco@microsoft.com (Edith Cohen), avinatan@google.com (Avinatan Hassidim), haimk@cs.tau.ac.il (Haim Kaplan), mansour@cs.tau.ac.il (Yishay Mansour), danny@cs.technion.ac.il (Danny Raz), yoavz@google.com (Yoav Tzur)

[1]Microsoft Research, CA, USA

[2]Blavatnik School of Computer Science, Tel Aviv University, Israel

[3]Google, Inc. Israel R&D Center

[4]Bar-Ilan University, Israel

[5]Technion, Israel

[6]work done while at Google, Inc. Israel R&D Center

The goal is to design schedules which optimize the tradeoff between the probing overhead and the failure detection time or more generally, the cost (or expected cost) associated with failures. We are interested in continuous monitoring, where failures may occur at any time during the (ongoing) process, and we would like to detect the failure soon after it occurs. Continuous testing comes in many flavors: deployment can be centralized or distributed across the network and may require following a fixed sequence of probes (*deterministic* schedules) or allow for randomization (*stochastic* schedules). There are also several natural objectives which we classify into two groups. Intuitively MAX_e objectives aim at minimizing the maximum expected detection time over all elements e, whereas the SUM_e objectives aim at minimizing the average (or weighted sum) of the detection time.

We illustrate differences between these objectives through the following simple example. We have n elements and 2 tests, one that covers a single element and another that covers all other $n-1$ elements. Now, if we want to minimize the maximal expected detection time we should send issue the tests in an alternating way (and get an expected value of 0.5). Any other way of scheduling the tests will increase the expected maximal detection time. On the other hand if we want to minimize the average detection time and we assume equal failure probabilities, then it makes sense to invoke the second test much more often (in fact as we show later in the paper $\sqrt{n}$ times) than the test that covers a single element. The two schedules described above are deterministic since they are determined by a fixed sequence of probes. One can also use a stochastic schedule in which we send each of the tests with probability 0.5 for the MAX_e objectives, and a stochastic schedule that sends the singleton test with probability $1/(\sqrt{n}+1)$ and the other test with probability $\sqrt{n}/(\sqrt{n}+1)$.

We present a common framework which unifies the treatment of stochastic and deterministic schedulers and of different objectives. Our unified study facilitates informed design of schedulers that are tailored to application needs. Whilst a stronger objective, such as obtaining deterministic rather than expected guarantees and controlling the worst-case rather than the average is clearly desirable, it is important to quantify the associated costs.

We first present a simple and appealing sub-class of general stochastic schedules, which we call *memoryless schedules*. Memoryless schedules perform continuous testing by invoking tests selected independently at random according to some fixed distribution. The stateless nature of memoryless scheduling translates to minimum deployment overhead and also makes them very suitable in distributed settings, where each type of test is initiated by a different controller. Going back to the example from the previous paragraph, the stochastic schedules there are memoryless since the distribution of the probes is fixed for each one of them. A general stochastic schedule for this example may be: select each of the probes with probability 0.5, but if the long probe was *not* selected in the last 2 rounds sent the long probe. This schedule uses the results of the previous steps to calculate the new probe and thus is not memoryless.

We show that the optimization problem of computing the probing frequencies under which a memoryless schedule optimizes a SUM_e objective can be formulated as a convex program and when optimizing MAX_e objectives, as a linear program. In both cases, the optimal memoryless schedule can be computed efficiently. This is in contrast to general stochastic schedules, over which we show that the optima are NP-hard to compute. Surprisingly perhaps, we also show that the natural and efficiently optimizable memoryless schedules have expected detection times that are guaranteed to be within a factor of two from the respective optimal stochastic schedule of the same objective. Moreover, detection times are geometrically distributed, and therefore variance in detection time is well-understood, which is not necessarily so for general stochastic schedules. We note that our convex program formulation can be viewed as a generalization of Kleinrock's classic "square-root law." Kleinrock's law [5] applies only to the special case of *singletons* where there is no overlap between the elements covered by each of the probes whereas our extension applies to *subset* tests.

Another important class of schedules are *deterministic schedules*. Such schedules are needed by applications requiring hard guarantees on detection times. Deterministic schedulers, however, are less suitable for distributed deployment and also come with an additional cost: the optimum of an objective on a deterministic schedule can exceed the expectation of the same objective over stochastic schedules. We study the inherent gap (which we call the D2M gap) between these optima. We show that for deterministic scheduling, performance of SUM_e or MAX_e objectives further depends on the exact order of the quantifiers in the exact definition of the particular objective in the family (average or maximum). While all variants are NP hard, there is significant variation between attainable approximation ratios for the different objectives.

Building on this, we efficiently construct deterministic schedules with approximation ratios that meet the analytic bounds. Our *random tree* (R-Tree) schedulers derive a deterministic schedule from the probing frequencies of a memoryless schedule, effectively "derandomizing" the schedule while attempting to loose as little as possible on the objective in the process. We show that when seeded, respectively, with a SUM_e or MAX_e optimal memoryless schedule, we obtain deterministic schedules with approximation ratio of $O(\log \ell)$ for the strongest SUM_e objective and ratio $O(\log \ell + \log n)$ for the strongest MAX_e objective, where n is the number of elements and ℓ is the maximum number of tests that can detect the failure of a particular element. We also present the Kuhn-Tucker

(KT) scheduler which is geared to SUM_e objectives and adapts gracefully to changing priorities which can be the result of changes in the network traffic patterns.

Finally, we evaluate the different schedulers on realistic networks of two different scales: We use both a globe-spanning backbone network and a folded-Clos network, which models a common data center architecture. In both cases, the elements we are testing are the network links. For the backbone, our tests are the set of MPLS paths and for the Clos network we use all routing paths. We demonstrate how our suite of schedulers offers both strong analytic guarantees, good performance, and provides a unified view on attainable performance with respect to different objectives. By relating performance of our deterministic schedulers to the respective memoryless optima, we can see that on many instances, our deterministic schedules are nearly optimal. We also demonstrate how our theoretical analysis explains observed performance and supports educated further tuning of schedulers.

This empirical study complements the theoretical analysis in the paper and provides a unified general treatment of silent failures detection phase. Our work, by unifying the treatment of different objectives, understanding how they relate, and developing efficient algorithms, facilitates an informed selection of objective and algorithm that are suitable for a particular application.

The paper is structured as follows. In Section 2 we present our model, general stochastic and deterministic schedules, and explain the different objectives. Memoryless schedules are introduced in Section 3. Deterministic scheduling is discussed in Section 4, followed by the R-Tree scheduler in Section 5 and Kuhn-Tucker schedulers in Section 6. Experimental results are presented in Section 7, extension of the model to probabilistic tests is discussed in Section 8, and related work is discussed in Section 9.

2. Model

An instance of a *test scheduling* problem is specified by a set V of elements (which can be thought of as network elements or links) of size n with a weight function $\boldsymbol{p}$ (which can be thought of as priority or importance of the elements) and a set $\mathcal{S}$ of tests (probe paths) of size m. For $i \in [m]$, test i is specified by a subset $s_i \subset V$ of elements. The failure of an element e can be detected by probing i if and only if $e \in s_i$, that is, if and only if test i contains the failed element. (This can be extended to the case where failures are detected with some positive probability.) We use ℓ_e to indicate the number of tests which include element e and $\ell \equiv \max_e \ell_e$.

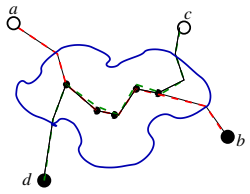


Figure 1: Network and elements covered by ab and cd origin-destination tests.

Continuous testing is specified by a *schedule* which generates an infinite sequence $\sigma = \sigma_1, \sigma_2, \ldots$ of tests. The schedule can be *deterministic* or *stochastic*, in which case, the probability distribution of the tests at time t depends on the actual tests preformed prior to time t. We also introduce *memoryless* schedules, which are a special subclass of stochastic schedules, in which the probability distribution of the tests is fixed over time. When the schedule is stochastic we use $\boldsymbol{\sigma}$ to denote the schedule itself and σ to denote a particular sequence that the schedule can generate.

2.1. Objectives

Objectives for a testing schedule aim to minimize a certain function of the number of tests invoked until a failure is detected. (We essentially measure time passed until the failure is detected by the "number of probes" required to discover it. If the probing rate is fixed this is indeed the time.) Several different natural objectives had been considered in the literature. Here we consider all these objectives through a unified treatment which allows us to understand how they relate to each other and how they can be computed or approximated.

The *detection time* $\text{T}_{\boldsymbol{\sigma}}(e,t)$ for element e at time t by a schedule $\boldsymbol{\sigma}$ is the expected time to detect a failure of element e that occurs at time t. If the schedule is deterministic, then $\text{T}_{\boldsymbol{\sigma}}(e,t) = \min_{h \geq 0} e \in s_{\sigma_{h+t}}$. If the schedule is stochastic, we take the expectation over sequences

$$\text{T}_{\boldsymbol{\sigma}}(e,t) = \mathsf{E}_{\boldsymbol{\sigma}}[\min_{h \geq 0} e \in s_{\sigma_{h+t}}] \; .$$

Note that the probability of any prefix is well defined for general stochastic schedules. Therefore $\text{T}_{\boldsymbol{\sigma}}(e,t)$, if finite, is well defined.

We classify natural objectives as MAX_e, when aiming to minimize the maximum detection time over elements, where the detection time of each element is multiplied by its weight, or as SUM_e when aiming to minimize a weighted sum over elements of their detection times. Both types of objectives are defined with respect to a weight function $\boldsymbol{p}$ over elements. Objectives in each family differ by the way they quantify over time: For example

one *MAX*$_e$ objective is to minimize the maximum detection time of an edge over all times, and a different *MAX*$_e$ objective would be to minimize the average over times of the maximum detection time of an edge in each time. Formal definitions follow below.

The weighting, or priorities of different elements, can capture the relative criticality of the element which in turn, can be set according to the volume or quality of service level of the traffic they handle. With the SUM$_e$ objectives, the weights can also correspond to estimated probability that elements fail, in which case the weighted objective capture the expected detection time after a failure, or to the product of failure probability of the element and cost of failure of this element, in which case the weighted objective is the expected cost of a failure. With the MAX$_e$ objectives we can use $p_e \equiv 1/\tau_e$, where τ_e is the minimum desired detection time for a failure of element e, or the cost of a unit of downtime of element e. We then aim to minimize the maximum cost of a failing element. In the sequel, unless otherwise mentioned, we assume that weights are scaled so that with SUM$_e$, $\sum_e p_e = 1$, and with MAX$_e$, $\max_e p_e = 1$.

To streamline the definitions and treatment of the different MAX$_e$ and SUM$_e$ objectives we define the operators M_e and E_e, which perform weighted maximum or average over elements, and M_t and E_t, which perform maximum or average over time. More precisely, for a function g of time or a function f over elements:

$$\mathsf{M}_t[g] = \sup_{\tau \geq 1} g(\tau) \qquad \mathsf{E}_t[g] = \lim_{h\to\infty} \frac{\sum_{t=1}^{h} g(\tau)}{h}$$

$$\mathsf{M}_e[f] = \max_e p_e f(e) \qquad \mathsf{E}_e[f] = \sum_e p_e f(e)$$

An application of the operator E_t requires that the limit exists and an application of the operator M_t requires that $g(\tau)$ is bounded.

When the operators are applied to the function $\mathrm{T}_{\boldsymbol{\sigma}}(e,t)$, we use the shorthand $\mathsf{M}_t[e|\boldsymbol{\sigma}] \equiv \mathsf{M}_t[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$, $\mathsf{E}_t[e|\boldsymbol{\sigma}] \equiv \mathsf{E}_t[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$, $\mathsf{M}_e[t|\boldsymbol{\sigma}] \equiv \mathsf{M}_e[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$, $\mathsf{E}_e[t|\boldsymbol{\sigma}] \equiv \mathsf{E}_e[\mathrm{T}_{\boldsymbol{\sigma}}(e,t)]$. For a particular element e, $\mathsf{M}_t[e|\boldsymbol{\sigma}]$ is the maximum over time t of the expected (over sequences) number of probes needed to detect a failure of e that occurred in time t, and $\mathsf{E}_t[e|\boldsymbol{\sigma}]$ is the limit of the average over time t of the expected number of probes needed to detect a failure of e that occurred in time t. For a particular time t, $\mathsf{M}_e[t|\boldsymbol{\sigma}]$ is the weighted maximum over the elements of the expected detection time of a failure at t, and $\mathsf{E}_e[t|\boldsymbol{\sigma}]$ is the weighted sum over the elements of their expected detection times at t. We consider all objectives that we can obtain from combinations of these operators. The operator pairs M_e and M_t (maximum over time or over elements) and E_e and E_t (average of expectation) commute, but other pairs do not, and we obtain six natural objectives, three MAX$_e$ and three SUM$_e$.

MAX$_e$ objectives: The three MAX$_e$ objectives are

- $\mathsf{M}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]]$, the weighted maximum over elements of the maximum over time of the detection time.
- $\mathsf{M}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]]$, the weighted maximum over elements of the average over time of the detection time.
- $\mathsf{E}_t[\mathsf{M}_e[t|\boldsymbol{\sigma}]]$, the average over time of the maximum detection time of an element at that time.

We shorten notation as follows.

$$\begin{aligned}
\mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] &= \mathsf{M}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]] \equiv \sup_{e,t} p_e \mathrm{T}_{\boldsymbol{\sigma}}(e,t) \\
\mathsf{M}_e\mathsf{E}_t[\boldsymbol{\sigma}] &= \mathsf{M}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]] \equiv \max_e p_e \mathsf{E}_t[e|\boldsymbol{\sigma}] \\
\mathsf{E}_t\mathsf{M}_e[\boldsymbol{\sigma}] &= \mathsf{E}_t[\mathsf{M}_e[t|\boldsymbol{\sigma}]] \equiv \lim_{h\to\infty} \frac{1}{h} \sum_{t=1}^{h} \max_e p_e \mathrm{T}_{\boldsymbol{\sigma}}(e,t)\ . \qquad (1)
\end{aligned}$$

SUM$_e$ objectives: The three SUM$_e$ objectives are

- $\mathsf{E}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]]$, the weighted sum over elements e of the maximum over time t of the detection time.
- $\mathsf{M}_t[\mathsf{E}_e[t|\boldsymbol{\sigma}]]$, the maximum over time of the weighted sum over e of the detection time.
- $\mathsf{E}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]]$, the weighted sum over elements of the average over time of the detection time.

We shorten notation as follows.

$$\mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}] = \mathsf{E}_e[\mathsf{M}_t[e|\boldsymbol{\sigma}]] = \sum_e p_e \mathsf{M}_t[e|\boldsymbol{\sigma}]$$

$$\mathsf{M}_t\mathsf{E}_e[\boldsymbol{\sigma}] = \mathsf{M}_t[\mathsf{E}_e[t|\boldsymbol{\sigma}]] = \sup_t \sum_e p_e \mathrm{T}_{\boldsymbol{\sigma}}(e,t) = \sup_t \mathsf{E}_e[t|\boldsymbol{\sigma}]$$

$$\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] = \mathsf{E}_e[\mathsf{E}_t[e|\boldsymbol{\sigma}]] = \sum_e p_e \mathsf{E}_t[e|\boldsymbol{\sigma}]$$

When the schedule $\boldsymbol{\sigma}$ is clear from context, we omit the reference to it in the notation. There are clearly schedules, deterministic or stochastic, over which our objectives are not defined. The $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{E}_e\mathsf{M}_t$, and $\mathsf{M}_t\mathsf{E}_e$ are defined when $\mathsf{M}_t[e]$ is defined for all elements e and the $\mathsf{M}_e\mathsf{E}_t$ and $\mathsf{E}_e\mathsf{E}_t$ are defined when $\mathsf{E}_t[e]$ is defined for all elements e. The $\mathsf{E}_t\mathsf{M}_e$ requires that the limit in Equation (1) exists. Formally, we define a schedule to be *valid* if for all elements e, $\mathsf{M}_t[e]$ and $\mathsf{E}_t[e]$ are well defined, and for all tests i the relative frequency of probing i converges, that is, the limit $\lim_{h\to\infty} \frac{\sum_{t=1}^{h} \Pr[\sigma_t=i]}{h}$ exists.[7] Henceforth we limit our attention only to valid schedules, which for brevity we will keep calling schedules.

2.2. Relating and optimizing objectives

The following lemma specifies the basic relation between the objectives. Its proof is straightforward.

Lemma 2.1. *For any schedule $\boldsymbol{\sigma}$,*

$$\mathit{SUM}_e\text{:} \qquad \mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}] \geq \mathsf{M}_t\mathsf{E}_e[\boldsymbol{\sigma}] \geq \mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \tag{2}$$

$$\mathit{MAX}_e\text{:} \qquad \mathsf{M}_e\mathsf{M}_t[\boldsymbol{\sigma}] \geq \mathsf{E}_t\mathsf{M}_e[\boldsymbol{\sigma}] \geq \mathsf{M}_e\mathsf{E}_t[\boldsymbol{\sigma}] \tag{3}$$

For any objective we want to find schedules that minimize it. We denote the infimum of the objective over deterministic schedules by the prefix opt_D, over memoryless schedules by opt_M, and over stochastic schedules by opt. For example for the objective $\mathsf{M}_e\mathsf{E}_t$, opt_D-$\mathsf{M}_e\mathsf{E}_t$ is the infimum $\mathsf{M}_e\mathsf{E}_t$ over deterministic schedules. Since memoryless and deterministic schedules are a subset of stochastic schedules, the deterministic or the memoryless optima are always at least the stochastic optimum: For any objective $\mathrm{opt}_D \geq \mathrm{opt}$ and $\mathrm{opt}_M \geq \mathrm{opt}$.

Relations (2) and (3) clearly hold with respect to the deterministic, memoryless, or stochastic optima of each objective. Lemma 2.2 shows that for stochastic schedules, the three optima of the objectives within each category (SUM_e or MAX_e) are in fact equal.

Lemma 2.2.

$$\mathit{opt}\text{-}\mathsf{E}_e\mathsf{M}_t = \mathit{opt}\text{-}\mathsf{M}_t\mathsf{E}_e = \mathit{opt}\text{-}\mathsf{E}_e\mathsf{E}_t \tag{4}$$

$$\mathit{opt}\text{-}\mathsf{M}_e\mathsf{M}_t = \mathit{opt}\text{-}\mathsf{E}_t\mathsf{M}_e = \mathit{opt}\text{-}\mathsf{M}_e\mathsf{E}_t\ . \tag{5}$$

Proof. The complete proof is provided in Appendix B.1. Proof sketch: For a stochastic schedule $\boldsymbol{\sigma}$ and a number N, we define a "cyclic" schedule $\boldsymbol{\sigma}_N$ which repeats a prefix of $\boldsymbol{\sigma}$ of length N. We show that for a sufficiently large N, for any item e, $\mathsf{E}_t[e|\boldsymbol{\sigma}_N] \leq (1+\epsilon)\mathsf{E}_t[e|\boldsymbol{\sigma}]$. We randomize the start time of $\boldsymbol{\sigma}_N$ to obtain a schedule for which $\mathrm{T}(e,t)$ is the same for all times t and equals $\mathsf{E}_t[e|\boldsymbol{\sigma}_N]$. Then (4) follows by applying this construction to opt-$\mathsf{E}_e\mathsf{E}_t$ and (5) follows by applying it to opt-$\mathsf{M}_e\mathsf{E}_t$. □

We denote by opt-SUM_e and opt-MAX_e the stochastic optima of all three SUM_e or MAX_e objectives:

$$\text{opt-}\mathrm{SUM}_e \equiv \text{opt-}\mathsf{E}_e\mathsf{M}_t = \text{opt-}\mathsf{M}_t\mathsf{E}_e = \text{opt-}\mathsf{E}_e\mathsf{E}_t \qquad \text{and} \quad \text{opt-}\mathrm{MAX}_e \equiv \text{opt-}\mathsf{M}_e\mathsf{M}_t = \text{opt-}\mathsf{E}_t\mathsf{M}_e = \text{opt-}\mathsf{M}_e\mathsf{E}_t\ .$$

We show that optimizing any of our SUM_e or MAX_e objectives is NP hard, the proof is based on a reduction to exact cover by sets of size 3 (X3C) and is provided in Appendix B.2.

Lemma 2.3. *Computing the optimal schedules for opt-$\mathrm{SUM}_e$and opt-$\mathrm{MAX}_e$is NP hard.*

[7] Deterministic schedules that are cyclic or stochastic schedules with finite memory are always valid, but general sequences may not be.

3. Memoryless schedules

Memoryless schedules are particularly simple stochastic schedules specified by a probability distribution $\boldsymbol{q}$ on the tests. At each time, independently of history, we draw a test $i \in [m]$ at random according to $\boldsymbol{q}$ ($i \in [m]$ is selected with probability q_i) and probe i, where the notation $[m] = \{1, \ldots, m\}$ is the set of integers from 1 to m. It is easy to see that in memoryless schedules, detection times are distributed geometrically. We show that memoryless schedules perform nearly as well, in terms of expected detection time, as general stochastic schedules. For notational convenience, we use the distribution $\boldsymbol{q}$ to denote also the memoryless schedule itself.

We first show that all SUM_e objectives and all MAX_e objectives are equivalent on any memoryless schedule.

Lemma 3.1. *For any memoryless schedule $\boldsymbol{q}$,*

$$\mathsf{E}_e\mathsf{M}_t[\boldsymbol{q}] = \mathsf{M}_t\mathsf{E}_e[\boldsymbol{q}] = \mathsf{E}_e\mathsf{E}_t[\boldsymbol{q}] = \sum_e \frac{p_e}{Q_e} \equiv \mathit{SUM}_e[\boldsymbol{q}]$$
$$\mathsf{M}_e\mathsf{M}_t[\boldsymbol{q}] = \mathsf{E}_t\mathsf{M}_e[\boldsymbol{q}] = \mathsf{M}_e\mathsf{E}_t[\boldsymbol{q}] = \max_e \frac{p_e}{Q_e} \equiv \mathit{MAX}_e[\boldsymbol{q}] \ ,$$

where $Q_e = \sum_{i|e\in s_i} q_i$.

Proof. The detection time of a failure of e via a memoryless schedule is a geometric random variable with parameter Q_e. In particular, for each element e, the distribution $T(e,t)$ are identical for all t and its expectation, $1/Q_e$, is equal to $\mathsf{M}_t[e]$ and $\mathsf{E}_t[e]$. From linearity of expectation, the $\mathsf{E}_e\mathsf{E}_t$, $\mathsf{M}_t\mathsf{E}_e$, and $\mathsf{E}_e\mathsf{M}_t$ are all equal to $\sum_e p_e/Q_e$. Similarly, $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{M}_e\mathsf{E}_t$, and $\mathsf{E}_t\mathsf{M}_e$ are all equal to $\max_e \frac{p_e}{Q_e}$. □

We use the notation $\text{opt}_M\text{-SUM}_e$ and $\text{opt}_M\text{-MAX}_e$ for the memoryless optima. That is

$$\text{opt}_M\text{-SUM}_e = \min_{\boldsymbol{q}} \text{SUM}_e[\boldsymbol{q}]$$
$$\text{opt}_M\text{-MAX}_e = \min_{\boldsymbol{q}} \text{MAX}_e[\boldsymbol{q}] \ .$$

3.1. Memoryless Optima

We show that the memoryless optima with respect to both the SUM_e and MAX_e objectives can be efficiently computed. This is in contrast to deterministic and stochastic optima, which are NP hard.

$$\text{minimize} \sum_e \frac{p_e}{\sum_{i|e\in s_i} q_i} \qquad (6)$$
$$\forall i,\ q_i \geq 0$$
$$\sum_i q_i = 1$$

(a) Convex program for SUM_e

$$\text{maximize}\ z \qquad (7)$$
$$\forall e,\ \frac{1}{p_e} \sum_{i|e\in s_i} q_i \geq z$$
$$\forall i,\ q_i \geq 0$$
$$\sum_i q_i = 1$$

(b) LP for MAX_e.

Figure 2: Computing SUM_e and MAX_e optimal memoryless schedules.

Theorem 3.1. *The optimal memoryless schedule for SUM_e objectives, that is, the distribution $\boldsymbol{q}$ such that $\mathit{SUM}_e[\boldsymbol{q}] = \mathit{opt}_M\text{-}\mathit{SUM}_e$ is the solution of the convex program* (6) *(Figure 2).*

The optimal memoryless schedules with respect to the MAX_e objectives can be computed using an LP.

Theorem 3.2. *The optimal memoryless schedule for MAX_e, that is, the distribution $\boldsymbol{q}$ which satisfies $\mathit{MAX}_e[\boldsymbol{q}] = \mathit{opt}_M\text{-}\mathit{MAX}_e$ is the solution of the LP* (7) *(Figure 2).*

Singletons instances: When each test is for a single element, the optimal solution of the convex program (6) has the frequencies of each element proportional to the square root of p_e [5], that is, $q_e = \sqrt{p_e}/\sum_e \sqrt{p_e}$. The SUM$_e$ optimum for an instance with weighting $\boldsymbol{p}$ is

$$\text{opt}_M\text{-SUM}_e(\boldsymbol{p}) = \sum_e \frac{p_e}{q_e} = \sum_e \sqrt{p_e} \sum_i \sqrt{p_i} = (\sum_i \sqrt{p_i})^2 \ . \tag{8}$$

In contrast, the solution of the LP (7) has optimal probing frequencies q_e proportional to p_e, that is, $q_e = p_e/\sum_e p_e$ and the MAX$_e$ optimum is $\text{opt}_M\text{-MAX}_e(\boldsymbol{p}) = \max_e \frac{p_e}{q_e} = \sum_e p_e$.

3.2. Memoryless versus Stochastic

For both SUM$_e$ and MAX$_e$ objectives, the optimum on memoryless schedules is within a factor of 2 of the optimum over general stochastic schedules.

Theorem 3.3.

$$\textit{opt-SUM}_e \leq \textit{opt}_M\textit{-SUM}_e \leq 2\textit{opt-SUM}_e \tag{9}$$
$$\textit{opt-MAX}_e \leq \textit{opt}_M\textit{-MAX}_e \leq 2\textit{opt-MAX}_e \tag{10}$$

Proof. The left hand side inequalities follow from memoryless schedules being a special case of stochastic schedules. To establish the right hand side inequalities, consider a stochastic schedule and let q_i be (the limit of) the relative frequency of test i (Recall that we only consider valid schedules, where the limit exists). We have

$$\mathsf{M}_t[e] \geq \mathsf{E}_t[e] \geq \frac{p_e}{2\sum_{i|e\in s_i} q_i}$$

Therefore, the average over elements $\sum_e \frac{p_e}{2\sum_{i|e\in s_i} q_i}$ must be at least half the optimum of (6) and the maximum over elements $\max_e \frac{p_e}{2\sum_{i|e\in s_i} q_i}$ must be at least half the optimum of (7). □

The following example shows that Theorem 3.3 is tight in that the "2" factors are realizable. That is, there are instances where the memoryless optimum is close to being a factor of 2 larger than the respective stochastic optimum.

Lemma 3.2. *For any $\epsilon > 0$, there is an instance on which*

$$\textit{opt}_M\textit{-MAX}_e = \textit{opt}_M\textit{-SUM}_e \geq (2-\epsilon)\textit{opt-MAX}_e = \textit{opt-SUM}_e$$

Proof. The instance has n elements, corresponding n singleton tests, and uniform priorities p_e. The optimal memoryless schedule, the solution of both (7) and (6), has $q_e = 1/n$ and $\mathsf{M}_t[e] = \mathsf{E}_t[e] = n$ for each element. The optimal deterministic schedule repeats a permutation on the n elements and has $\mathsf{M}_t[e] = \mathsf{M}_e[t] = n$ and $\mathsf{E}_t[e] = \mathsf{E}_e[t] = (n+1)/2$ for all e, t. The optimal stochastic selects a permutation uniformly at random every n steps and follows it. It has $\mathsf{M}_t[e] = \mathsf{E}_t[e] = (n+1)/2$ for all elements. □

4. Deterministic scheduling

The distinction between objectives within each of the MAX$_e$ and SUM$_e$ groups does matter with deterministic scheduling. For an instance and objective, we attempt to understand the relation between the deterministic and stochastic optima. For deterministic MAX$_e$ objectives, the comparison is to opt-MAX$_e$ and for SUM$_e$ objectives, it is to opt-SUM$_e$.

We show that on all instances, the deterministic $\mathsf{E}_e\mathsf{E}_t$ is equal to opt-SUM$_e$. Deterministic $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_e\mathsf{M}_t$, however, are always strictly larger (proof is provided in Appendix B.3.

Lemma 4.1.

$$\textit{opt}_D\textit{-}\mathsf{E}_e\mathsf{E}_t = \textit{opt-SUM}_e \tag{11}$$
$$\textit{opt}_D\textit{-}\mathsf{E}_e\mathsf{M}_t \geq 2\textit{opt-SUM}_e - 1 \tag{12}$$
$$\textit{opt}_D\textit{-}\mathsf{M}_e\mathsf{M}_t \geq 2\textit{opt-MAX}_e - 1 \tag{13}$$

We can show that finding the optimal schedules for all these objectives is NP hard.

Lemma 4.2. *Computing any one of the following optima is NP hard: opt$_D$-$\mathsf{E}_e\mathsf{E}_t$, opt$_D$-$\mathsf{M}_t\mathsf{E}_e$, opt$_D$-$\mathsf{E}_e\mathsf{M}_t$, opt$_D$-$\mathsf{M}_e\mathsf{M}_t$, opt$_D$-$\mathsf{M}_e\mathsf{E}_t$, and opt$_D$-$\mathsf{E}_t\mathsf{M}_e$.*

This proof, similarly to the proof of Lemma 2.3, is also based on a reduction to the to exact cover by sets of size 3 (X3C) and details are omitted. Additional relations, upper bounding the deterministic objective by the stochastic objective follow from relations with memoryless optima which are presented next.

For a deterministic schedule and an objective, the *approximation ratio* is the ratio of the objective on the schedule to that of the (deterministic) optimum of the same objective. We are ultimately interested in efficient constructions of deterministic schedules with good approximation ratio and in quantifying the cost of determinism, that is, asking how much worse a deterministic objective can be over the respective stochastic objective.

We define the *D2D*, *D2M*, and *D2S* of a deterministic schedule as the ratio of the objective on the schedule to that of the deterministic, memoryless, or stochastic optimum of the same objective. Since both deterministic and stochastic optima are NP hard to compute, so is the D2D (the approximation ratio) and the D2S. The D2M of any given schedule, however, can be computed efficiently by computing the memoryless optimum. The D2M can then be used to bound the D2D and D2S, giving an upper bound on how far our schedule is from the optimal deterministic or stochastic schedule. In particular, the relation D2S $\leq$ D2M ≤ 2 D2S follows from Theorem 3.3. We study the relation between the memoryless and deterministic optima below.

4.1. Memoryless versus deterministic

Since a deterministic schedule is a special case of a stochastic schedule, from Theorem 3.3, the memoryless optimum is at most twice the deterministic optimum. The proof of Lemma 3.2 shows:

Lemma 4.3. *For any ϵ, there is an instance on which*

$$\begin{aligned} A =& \textit{opt}_M\textit{-MAX}_e = \textit{opt}_M\textit{-SUM}_e = \textit{opt}_D\textit{-}\mathsf{M}_e\mathsf{M}_t = \textit{opt}_D\textit{-}\mathsf{E}_e\mathsf{M}_t = \textit{opt}_D\textit{-}\mathsf{E}_t\mathsf{M}_e \\ B =& \textit{opt}_D\textit{-}\mathsf{M}_e\mathsf{E}_t = \textit{opt}_D\textit{-}\mathsf{E}_e\mathsf{E}_t = \textit{opt}_D\textit{-}\mathsf{M}_t\mathsf{E}_e = \textit{opt-MAX}_e = \textit{opt-SUM}_e \\ & A \geq (2-\epsilon)B \end{aligned}$$

That is, for the weaker SUM$_e$ and MAX$_e$ deterministic objectives, a gap of 2 is indeed realizable, meaning that it is possible for the deterministic optimum to be smaller than the respective memoryless optimum. For the strongest objectives, $\mathsf{E}_e\mathsf{M}_t$ for SUM$_e$ and $\mathsf{M}_e\mathsf{M}_t$ for MAX$_e$, we show that the deterministic optimum is at least the memoryless optimum:

Lemma 4.4.

$$\begin{aligned} \textit{opt}_M\textit{-SUM}_e &\leq \textit{opt}_D\textit{-}\mathsf{E}_e\mathsf{M}_t \\ \textit{opt}_M\textit{-MAX}_e &\leq \textit{opt}_D\textit{-}\mathsf{M}_e\mathsf{M}_t \end{aligned}$$

Proof. Similar to the proof of Theorem 3.3: Consider a deterministic schedule and let q_i be (the limit of) the relative frequency of test i. We have $\mathsf{M}_t[e] \geq \frac{p_e}{\sum_{i|e\in s_i} q_i}$. □

We next consider the other direction, upper bounding the deterministic optimum by the memoryless optimum. For the objectives $\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{M}_e\mathsf{E}_t$, which are respectively the weakest SUM$_e$ and MAX$_e$ objectives, we show that the deterministic optimum is at most the memoryless optimum. Moreover, we can efficiently construct deterministic schedules with D2M arbitrarily close to 1 (and thus approximation ratio of at most 2).

Lemma 4.5.

$$\begin{aligned} \textit{opt}_D\textit{-}\mathsf{M}_e\mathsf{E}_t &\leq \textit{opt}_M\textit{-MAX}_e \\ \textit{opt}_D\textit{-}\mathsf{E}_e\mathsf{E}_t &\leq \textit{opt}_M\textit{-SUM}_e \end{aligned}$$

and for any $\epsilon > 0$ we can efficiently construct deterministic schedules with $\mathsf{M}_e\mathsf{E}_t$ or $\mathsf{E}_e\mathsf{E}_t$ D2M $\leq (1+\epsilon)$.

Proof. For any ϵ, for a long enough run of the memoryless schedule $\boldsymbol{q}$, there is a positive probability that for all elements, the average over time of $T(e,t)$ (in the part of the sequence where it is finite) is at most $(1+\epsilon)\mathsf{E}_t[e|\boldsymbol{q}]$. We obtain the deterministic schedule by cycling through such a run. If the run is sufficiently long then the suffix in which $T(e,t)$ is infinite is a small fraction of the run and the resulting schedule σ has $\mathsf{M}_e\mathsf{E}_t[\sigma] \leq (1+\epsilon)\text{opt}_M\text{-MAX}_e$ and $\mathsf{E}_e\mathsf{E}_t[\sigma] \leq (1+\epsilon)\text{opt}_M\text{-SUM}_e$. □

We are now ready to relate the D2D and D2M. We obtain D2D $\leq$ 2 D2M, and for $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_e\mathsf{M}_t$ (see Lemma 4.4), we have D2D $\leq$ D2M. Accordingly, D2M $\geq 1/2$, and for $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{M}_e\mathsf{M}_t$ we have D2M ≥ 1. The optimum D2M is the minimum possible over all schedules. We refer to the supremum of optimum D2M over instances as the *D2M gap* of the scheduling problem.

In contrast, for the strongest objectives, $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{M}_e\mathsf{E}_t$, and $\mathsf{E}_e\mathsf{M}_t$, we construct a family of instances with asymptotically large optimal D2M, obtaining a lower bound on the D2M gap. We also show that opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ and opt$_D$-$\mathsf{E}_t\mathsf{M}_e$ are hard to approximate better than $\ln(n)$: (Proof details are provided in Appendix B.4)

Lemma 4.6. *There is a family of instances with m tests and n elements such that each element participates in ℓ tests with the following lower bounds on D2M: The $\mathsf{E}_t\mathsf{M}_e$-D2M (and thus $\mathsf{M}_e\mathsf{M}_t$-D2M) $\Omega(\ln n)$ and $\Omega(m)$. The $\mathsf{E}_e\mathsf{M}_t$ D2M is $\Omega(\log \ell)$. Moreover, these instances can be realized on a network, where elements are links and tests are paths.*

Lemma 4.7. *The problems opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ and opt$_D$-$\mathsf{E}_t\mathsf{M}_e$ are hard to approximate to anything better than $\ln(n)$.*

Proof. When $\boldsymbol{p}$ is uniform, opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ is equivalent to set cover – an approximation ratio for opt$_D$-$\mathsf{M}_e\mathsf{M}_t$ implies the same approximation ratio for set cover [2], which is hard to approximate [6].

This also extends to opt$_D$-$\mathsf{E}_t\mathsf{M}_e$, again using uniform $\boldsymbol{p}$. A minimum set cover of size k implies a schedule (cycling through the cover) with $\mathsf{E}_t\mathsf{M}_e$ of k. Also, a schedule with $\mathsf{E}_t\mathsf{M}_e$ at most k means that $\mathsf{M}_e[t] \leq k$ for at least one t, means there is a cover of size k. □

Summary of relations

A summary of these relations, which also includes results from our R-Tree schedulers (Section 5) is provided in Table 1. The lower bounds on the D2M gap are established in Lemma 4.6 through example instance on which the optimum D2M is large. The lower bound on approximability is established in Lemma 4.7. Both lower and upper bounds for $\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{M}_e\mathsf{E}_t$ are established in Lemma 4.5, $\mathsf{E}_e\mathsf{M}_t$ D2M upper bound in Theorem 5.1, and $\mathsf{M}_e\mathsf{M}_t$ D2M in Theorem 5.2.

objective	scheduling D2M	D2M gap	approximability
$\mathsf{E}_e\mathsf{E}_t$	1	1	
$\mathsf{M}_e\mathsf{E}_t$	1	1	
$\mathsf{E}_e\mathsf{M}_t$	$O(\ln m)$	$\Omega(\ln m)$	
$\mathsf{M}_e\mathsf{M}_t$, $\mathsf{E}_t\mathsf{M}_e$	$O(\log n + \log \ell)$	$\Omega(\log n)$, $\Omega(m)$	$\Omega(\log n)$

Table 1: D2M upper bounds of our schedulers and lower bounds on the D2M gap and on efficient approximability.

5. R-Tree schedules

We present an efficient construction of deterministic schedules from a distribution $\boldsymbol{q}$ and relate detection times of the deterministic schedule to (expected) detection times of the memoryless schedule defined by $\boldsymbol{q}$.

We can tune the schedule to either MAX$_e$ or SUM$_e$ objectives, by selecting accordingly the input frequencies $\boldsymbol{q}$ as a solution of (7) or (6). We then derive analytic bounds on the D2M of the schedules we obtain.

The building block of *random tree* (R-Tree) schedules is *tree schedules*, which are deterministic schedules specified by a mapping of tests to nodes of a binary tree. A tree schedule is specified with respect to probing frequencies $\boldsymbol{q}$ and has the property that for any test, the maximum probing interval in the deterministic schedule is guaranteed to be close to $1/q_i$. However, if we do not place the tests in the tree carefully then for an element covered by multiple tests the probing interval can be close to that of its most frequent test, but yet far from the desired (inverse of) $Q_e = \sum_{i|e\in s_i} q_i$. Therefore, even when computed with respect to $\boldsymbol{q}$ which solves (6), the tree schedule can have $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{E}_e\mathsf{E}_t$ D2M ratios $\Omega(\ell)$.

We define a distribution over tree schedules obtained by randomizing the mapping of tests to nodes. We then bound the expectation of the $\mathsf{E}_e\mathsf{M}_t$ and $\mathsf{E}_e\mathsf{E}_t$ (when applied to $\boldsymbol{q}$ which solves (6)) and $\mathsf{M}_e\mathsf{M}_t$ (when applied to $\boldsymbol{q}$ which solves (7)) over the resulting deterministic schedules. Given a bound on the expectation of an objective, there is a constant probability that a tree schedule randomly drawn from the distribution will satisfy the same bound (up to a small constant factor). An R-Tree schedule is obtained by constructing multiple tree schedules drawn from the distribution, computing the objectives on these schedules, and finally, returning the best performing tree schedule. Note that even though the construction is randomized, the end result, the R-Tree schedule, is deterministic, since it is simply a tree schedule.

Specifically, lets take SUM_eas an example, we apply the R-Tree schedule construction several times with $\boldsymbol{q}$'s solving (6). The tree with the best $\mathsf{E}_e\mathsf{M}_t$ has $O(\log(\ell))$ $\mathsf{E}_e\mathsf{M}_t$ D2M and the tree with the best $\mathsf{E}_e\mathsf{E}_t$ has a constant $\mathsf{E}_e\mathsf{E}_t$ D2M. Furthermore we can also find a tree which satisfies both guarantees.

Theorem 5.1. *A deterministic schedule with $\mathsf{E}_e\mathsf{M}_t$ D2M ratio of $O(\log \ell)$ and a constant $\mathsf{E}_e\mathsf{E}_t$ D2M ratio can be constructed efficiently.*

The theorem is tight since from Lemma 4.6, the $\mathsf{E}_e\mathsf{M}_t$ D2M gap on some instances is $\Omega(\log \ell)$, and therefore, we can not hope for a better dependence on ℓ.[8]

For MAX_e, we show that when we apply the R-Tree schedule construction to $\boldsymbol{q}$ which is the optimum of (7), we obtain a deterministic schedule with $O(\log \ell + \log n)$ $\mathsf{M}_e\mathsf{M}_t$ D2M.

Theorem 5.2. *A deterministic schedule with $\mathsf{M}_e\mathsf{M}_t$ D2M ratio of $O(\log \ell + \log n)$ can be constructed efficiently.*

From Theorems 5.1 and 5.2, we obtain the following upper bounds on the D2M gap and efficiently construct deterministic schedules satisfying these bounds (summarized in Table 1).

$$\text{opt}_D\text{-}\mathsf{E}_e\mathsf{M}_t = O(\log \ell)\text{opt-SUM}_e$$
$$\text{opt}_D\text{-}\mathsf{M}_e\mathsf{M}_t = O(\log \ell + \log n)\text{opt-}\mathsf{M}_e\mathsf{M}_t$$

We provide construction details of our R-Tree schedulers. The analysis, which includes the proofs of Theorems 5.1 and 5.2, is deferred to Appendix A.

5.1. Tree schedules

A tree schedule is a deterministic schedule guided with frequencies $\boldsymbol{q}$ where probes to test i are spaced $[1/q_i, 2/q_i)$ probes apart. When q_i has the form $q_i = 2^{-j}$, test i is performed regularly with period 2^j.

Assume for now that $q_i = 2^{-L_i}$ for positive integer L_i for all i. We map each i to nodes of a binary tree where i is mapped to a node at level L_i and no test can be a child of another. This can be achieved by greedily mapping tests by decreasing level – we greedily map tests with level $L_i = 1$, then tests with $L_i = 2$ and so on. Once a test is mapped to a node, its subtree is truncated and it becomes a leaf.

From this mapping, we can generate a deterministic schedule as follows: The sequence is built on alternations between left and right child at each node. Each node "remembers" the last direction to a child. To select a test, we do as follows. First visit the root and select the child that was not visited previous time. If a leaf, we are done, otherwise, we recursively select the child that was not previously visited and continue. This until we get to a leaf. We then output test i. This process changed "last visit" states on all nodes in the path from the root to the leaf. It is easy to see that if a leaf at level L is visited once every 2^L probes. An example of a set of frequencies, a corresponding mapping, and the resulting schedule is provided in Figure 3.

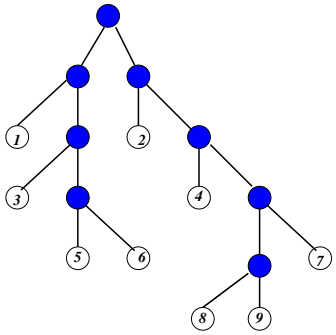
If probabilities are of general form, we can map each test according to the highest order significant bit (and arbitrarily fill up the tree). When doing this we get per-test ratio between the actual and desired probing frequencies of at most 2. Alternatively, we can look at the bit representation of q_i– separately map all "1" positions in the first few significant bits to tree nodes. In this case the average probing frequency of each test is very close to q_i but the maximum time between probes depends on the relation between the tree nodes to which the bits of test i are mapped to. The only guarantee we have on the maximum is according to the most significant bit $2^{-\lceil \log_2(1/q_i) \rceil}$. Under "random" mappings the expectation of the maximum gets closer to the average.

5.2. Random tree schedules

Consider an instance and a memoryless schedule with frequencies $\boldsymbol{q}$. We assume that q_i have the form 2^{-L_i} for positive integers L_i (this is without loss of generality as we can only look at the highest order bit and loose a factor of at most 2). We construct a tree schedule for $\boldsymbol{q}$ by mapping the tests to nodes randomly as follows. We process tests by increasing level. In each step (level), all tests of the current level are randomly mapped to the available tree nodes at that level. After a test is mapped to a node, its subtree is truncated.

For each level N (which can be at most the maximum L_i), we can consider the *level-N schedule*, which is a cyclic schedule of length 2^N. The schedule specifies the probes for all tests with level $L_i \leq N$, and leaves some spots "unspecified".

[8] As a side note, recall that according to (11) there exist schedules with $\mathsf{E}_e\mathsf{E}_t$ D2M close to 1, so with respect to $\mathsf{E}_e\mathsf{E}_t$ this only shows that we can simultaneously obtain a $\mathsf{E}_e\mathsf{M}_t$ D2M that is logarithmic in ℓ and at the same time a constant $\mathsf{E}_e\mathsf{E}_t$ D2M.



$q_1 = q_2 = 1/4, q_3 = q_4 = 1/8, q_5 = q_6 = q_7 = 1/16, q_8 = q_9 = 1/32$

L	schedule															
2	1	2	x	x												
3	1	2	3	4	1	2	x	x								
4	1	2	3	4	1	2	5	x	1	2	3	4	1	2	6	7
5	1	2	3	4	1	2	5	8	1	2	3	4	1	2	6	7
-	1	2	3	4	1	2	5	9	1	2	3	4	1	2	6	7

Figure 3: Mapping tests to nodes of a binary tree to produce a deterministic schedule. The table shows the level-L schedule for $L = 2, 3, 4, 5$. The full deterministic schedule cycles through the level-5 schedule.

We now specify the level-N schedule of the tree. Consider a completion of the tree to a full binary one with 2^N leaves (truncate everything below level N). Associate with each leaf a a binary number $\overline{a}$ which contains a 0 at digit i (from right to left, i.e. the least significant digit correspond to the child of the root and the most significant digit corresponds to the leaf itself) if the ith child on its path from the root is a left child. We refer to $\overline{a}$ as the *position* of leaf a.

We construct the sequence by associating test i with all leaf descendants of the node containing it, and with all the positions of the sequence corresponding to these leaves. Putting it in another words the level-N schedule of the tree cycles through the leaves a at level-N (of the completion of the tree) according to the order defined by $\overline{a}$ and probes the test associated with each leaf. A test with $q_i = 2^{-L_i}$ is probed in regular intervals of 2^{L_i}. The first probe is distributed uniformly at random from $[0, 2^{L_i} - 1]$.

Level-N schedules constructed from the same mapping for different depths N are consistent in the following sense: The level $N' > N$ schedule is $2^{N'-N}$ repetitions of the level-N schedule in terms of the tests specified by a level-N schedule (those with level $L_i \leq N$) and also specifies tests with $N < L_i \leq N'$.

6. The Kuhn-Tucker scheduler

The Kuhn-Tucker conditions on the optimal solution of our convex program (6) imply that the values

$$r_i = \frac{\partial \sum_e \frac{p_e}{\sum_{i|e \in s_i} q_i}}{\partial q_i} = -\sum_{e|e \in s_i} \frac{p_e}{(\sum_{j|e \in s_j} q_j)^2} .$$

are balanced for different tests. Based on that, we suggest a deterministic greedy heuristic for SUM_e, illustrated in Algorithm 1. For each element e, we track $x[e] \geq 1$ which is the elapsed number of probes since e was last probed. We then choose the test i with maximum $\sum_{e \in s_i} p_e x[e]^2$.

We conjecture that the KT schedule has $\mathsf{E}_e\mathsf{E}_t$ which is at most twice the optimal. Viewing the quantity $\sum_e p_e x[e]^2$ as "potential" the average reduction in potential is the $\mathsf{E}_e\mathsf{E}_t$ of the sequence. We do not provide bounds on the approximation ratio, but test this heuristic in our experiments.

Algorithm 1 Kuhn-Tucker (KT) schedule

function BEST-TEST
  $v \leftarrow 0$
  for $s \in \mathcal{S}$ **do**
    $y \leftarrow 0$
    for $e \in s$ **do**
      $y \leftarrow y + p_e x[e]^2$
    if $(y > v)$ **then**
      $b \leftarrow s$; $v \leftarrow y$
  return b ▷ test with maximum $\sum_{e \in s_i} p_e x[e]^2$

function KT-SCHEDULE($V, p, \mathcal{S}$)
  for $e \in V$ **do**
    $x[e] \leftarrow 1$
  while True **do**
    $s \leftarrow$BEST-TEST()
    output s
    For e let $x[e] \leftarrow x[e] + 1$
    for $e \in s$ **do**
      $x[e] \leftarrow 1$

The KT scheduler can be deployed when priorities are modified on the go. This is in contrast to other schedulers which pre-compute the schedule .

SUM_e in memoryless schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
Convex	95.66	59.72	25.92	32.02
LP	105.29	68.77	118.38	32.02
Uniform	229.16	72.27	260.46	33.00
SAMP SC	111.56	82.70	86.17	32.00
SAMP KT	108.54	61.45	86.17	32.00

$\mathsf{E}_e\mathsf{E}_t$ in deterministic schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
SC	60.27	49.42	51.52	16.50
KT	58.04	33.93	14.63	16.50
RT CON	66.43	49.21	17.92	30.76
RT LP	85.47	63.81	88.07	31.00
RT-S CON	57.87	46.91	18.69	
RT-S LP	59.70	50.24	87.47	

$\mathsf{M}_t\mathsf{E}_e$ in deterministic schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
SC	70.43	62.08	93.80	16.50
KT	70.04	62.08	20.36	16.50
RT CON	72.23	56.53	24.65	36.05
RT LP	95.81	73.34	113.47	36.20
RT-S CON	60.08	50.02	23.38	
RT-S LP	63.09	53.82	96.67	

$\mathsf{E}_e\mathsf{M}_t$ in deterministic schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
SC	124.93	109.46	114.29	32.00
KT	130.11	93.02	35.34	32.00
RT CON	180.00	179.91	53.40	144.14
RT LP	319.12	261.65	269.01	146.70
RT-S CON	121.24	103.91	42.61	
RT-S LP	123.35	107.42	183.89	

Table 2: SUM_e objectives. Table shows expected time with memoryless schedules (same for all SUM_e objectives) and $\mathsf{E}_e\mathsf{E}_t \leq \mathsf{M}_t\mathsf{E}_e \leq \mathsf{E}_e\mathsf{M}_t$ on different deterministic schedulers.

7. Experimental Evaluation

We evaluated the performance of our schedulers for testing for silent link failures in two networks. The first is a backbone network (denoted GN in the sequel) of a large enterprise. We tested 500 of the network links with 3000 MPLS paths going through them.

The second network we considered is a (very regular) folded Clos network (denoted Clos) of 3 levels and 2048 links. On this network we considered all paths between endpoints. The Clos network is a typical interconnection network in data centers.

For the Clos network, we only considered uniform weights (priorities), meaning that all links are equally important. For the GN network, we considered uniform weights (denoted GN-U), weights that are proportional to the number of MPLS paths traversing the link (GN-P, where P designates popularity), and Zipf distributed weights with parameter 1.5 (GN-Z).

On these four networks (links and paths with associated weights), Clos, GN-U, GN-P, and GN-Z, we simulated our schedulers and evaluated their performance with respect to the different objectives.

Memoryless schedulers: We solved the convex program (6) for SUM_e objectives and the LP (7) for MAX_e objectives to obtain optimal memoryless probing frequencies $\boldsymbol{q}$. These optimization problems were solved using Matlab (for the LP) and CVX (for the convex program, see `http://cvxr.com/cvx/`).

We compared these optimal memoryless schedules to other memoryless schedules obtained using three naive selections of probing frequencies: the first is uniform probing of all paths (Uniform), the second is uniform probing of a smaller set of paths that cover all the links (SAMP SC), and the third is probing according to frequencies generated by the Kuhn-Tucker schedule (SAMP KT).

The performance of these schedules, in terms of the expected detection times $\mathrm{T}(e,t)$ is shown in Table 2 (SUM_e objective) and Table 3 (MAX_e objective). The schedulers optimized for one of the objectives, SUM_e or MAX_e, clearly dominate all others with respect to the objective it optimizes. We can see that while on some instances the alternative schedulers perform close to optimal, performance gaps can sometimes be substantial. In particular, a schedule optimized for one objective can perform poorly with respect to the other objective. We note, however, that our unified treatment facilitates designing schedules which trade off performance with respect to two objectives.

We illustrate the qualitative difference between the SUM_e and MAX_e objectives through Figure 4 (A). The figure shows a reverse CDF of $\mathrm{T}(e,t)$, the expected time to detect a failure of a link of the backbone network with uniform weights (GN-U). (Recall that $\mathrm{T}(e,t)$ is fixed for all t for memoryless schedules.) Given a reverse CDF of a schedule, the maximum point on the curve is the MAX_e of the schedule whereas the average value (area under

the curve) is the SUM_e of the schedule. We can see that the schedule computed by the LP (7), which optimizes MAX_e has a smaller maximum whereas the schedule computed by the convex program (6) has a smaller area.

Deterministic schedulers: We now evaluate our deterministic schedulers. Here, $\mathrm{T}(e,t)$, the elapsed time from time t till the next path containing e is scheduled, is deterministic. We used two different implementation of the R-Tree algorithm (Section 5). In the first, the algorithm was seeded with the frequencies computed by the LP (RT LP) or by the convex program (RT CON) when applied to the full set of paths. We discuss the second implementation in the sequel. We also implemented the Kuhn-Tucker (KT) scheduler (Section 6), and the classic greedy Set Cover algorithm (SC) which was previously used for the $\mathsf{M}_e\mathsf{M}_t$ metric [3, 2, 4] (minimum set cover is the optimal deterministic scheduler for $\mathsf{M}_e\mathsf{M}_t$ when priorities are uniform). This scheduler cycles through a sequence consisting of this set cover.

Table 2 shows the values of all SUM_e objectives for the different memoryless and deterministic schedulers and Table 3 shows the same for the MAX_e objectives. It is easy to verify the relations between the three different SUM_e objectives and three different MAX_e objectives (see Lemma 2.1). The gaps between the objectives show again that an informed selection of the objective is important. We can also see that with uniform priorities (GN-U and Clos) the SC scheduler performs well. Indeed, in this case minimum set cover produces the optimal deterministic schedule for $\mathsf{M}_e\mathsf{M}_t$ and $\mathsf{E}_t\mathsf{M}_e$. When priorities are highly skewed, however, as is the case for GN-Z, its performance deteriorates.

The KT scheduler performed well on the SUM_e objectives, which it is designed for. Because of its adaptive design, which does not involve precomputation of a fixed schedule, the KT scheduler is highly suitable for applications where priorities are changing on the go. One such scenario is when priorities of different elements correspond to the current traffic levels traversing the element. The KT scheduler gracefully adapts to changing traffic levels.

Our R-Tree schedulers (RT CON and RT LP) did not perform well on some of the instances, and in some cases, performed worse than SC and KT. The reason, as the analysis shows (see Section 5), is the logarithmic dependence on ℓ, which in our case, is the maximum number of paths used to cover an element in the solution of the LP and convex programs. The collection of paths computed by the LP and Convex solvers turned out to have high redundancy, where subpaths have many alternatives and the fractional solvers tend to equally use all applicable paths. We can see evidence for this fragmentation in Figure 4.

To address this issue, we seeded the R-Tree algorithm with respective solutions of the LP and Convex programs applied to a modified instance with a pre-selected small subset of the original paths. The subset was picked so that it contains a cover of the links and also tested to ensure that the objective of the optimization problem does not significantly increase when implementing this restriction. On those instances, tests which constitute a set cover of the links and produced by the greedy approximation algorithm, performed well. We denote the respective schedulers obtained this way using the LP and convex solutions, by RT-S LP and RT-S CON.

The results of this experiment are included in Tables 2 and 3. We can observe that this heuristic substantially improves the performance of the R-Tree algorithm for all objectives. Moreover, RT-S was never worse than SC, and when SC was not optimal, substantially improved over SC. We leave the question of how to choose the subset to best balance the loss in the objective of the memoryless schedule with the gain in better derandomization for further research.

Memoryless vs. Deterministic: Memoryless schedulers are stateless and highly suitable for distributed deployment whereas deployment of deterministic schedulers requires some coordination between probes initiated from different start points. However, due to their stochastic nature, with memoryless scheduling we can only obtain guarantees on the expectation whereas with deterministic schedulers we can obtain worst case guarantees on the time (or weighted cost) until a failure is detected. We demonstrate this issue by illustrating, in Figure 4 (B) the distribution over the links of the backbone graph of the maximum detection time in the deterministic R-Tree scheduler, $\mathsf{M}_t[e]$, and the 99th percentile line for the memoryless schedulers (elapsed time to detection in 99% of the time). Figure 4 (C) shows the same data for the schedulers RT-S LP and RT-S CON which were derived after restricting the set of paths over which optimization was performed. One can see that when there are strict requirements on worst-case detection times, deterministic schedules dominate.

Moreover, even when comparing expected (memoryless) versus worst-case (deterministic) detection times, we can see that our best deterministic schedulers often have $\mathsf{E}_e\mathsf{E}_t$, $\mathsf{M}_t\mathsf{E}_e$, and $\mathsf{M}_e\mathsf{E}_t$ detection times that are $20\% - 50\%$ smaller than the respective memoryless optimum. Our analysis shows (Section 4.1) that on these objectives it is possible for the optimal deterministic detection times to be up to a factor of 2 smaller than the respective memoryless optimum. On the remaining objectives, the deterministic optimum can not be better than the memoryless one and can be much worse (asymptotically so). Recall that while the memoryless optimum can

MAX_e in memoryless schedulers:

algorithm	GN-U	GN-P	GN-Z	Clos
Convex	221.53	21.81	6.85	32.02
LP	132.05	12.65	2.67	32.02
Uniform	2787	12.73	249.28	34.00
SAMP SC	143.00	53.65	72	32.00
SAMP KT	243.00	22.74	72	32.00

$\mathsf{M}_e\mathsf{E}_t$ in deterministic schedulers

algorithm	GN-U	GN-P	GN-Z	Clos
SC	72.00	43.41	48.17	16.50
KT	122.00	11.97	4.28	16.50
RT CON	162.00	20.15	5.31	40.61
RT LP	173.90	18.92	2.91	40.53
RT-S CON	92.50	22.15	4.90	
RT-S LP	71.50	22.16	1.90	

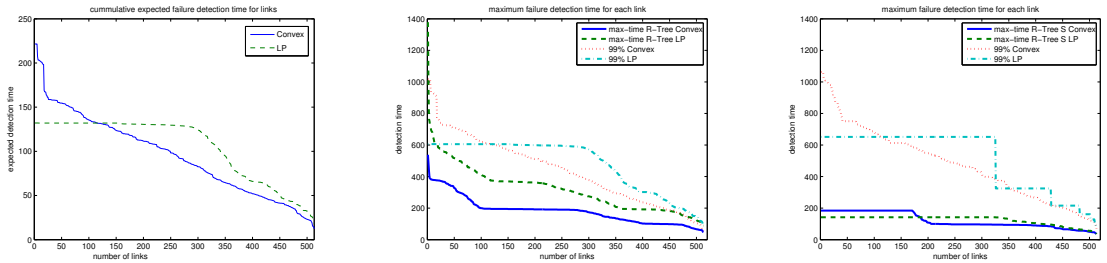
$\mathsf{E}_t\mathsf{M}_e$ in deterministic schedulers

algorithm	GN-U	GN-P	GN-Z	Clos
SC	142.99	54.02	50.80	32.00
KT	234.30	34.02	4.54	32.00
RT CON	345.85	55.26	6.12	147.71
RT LP	531.12	65.31	7.05	156.80
RT-S CON	182.78	42.69	6.01	
RT-S LP	142.00	43.22	3.21	

$\mathsf{M}_e\mathsf{M}_t$ in deterministic schedulers

algorithm	GN-U	GN-P	GN-Z	Clos
SC	143.00	95.02	113.00	32.00
KT	243.00	35.65	9.00	32.00
RT CON	468.00	85.72	24.00	257.00
RT LP	833.00	97.79	13.95	225.00
RT-S CON	184.00	50.00	14.00	
RT-S LP	142.00	54.00	5.00	

Table 3: MAX_e objectives. Table shows expected time with memoryless schedules (same for all MAX_e objectives) and $\mathsf{M}_e\mathsf{E}_t \leq \mathsf{E}_t\mathsf{M}_e \leq \mathsf{M}_e\mathsf{M}_t$ on different deterministic schedulers.



(A) (B) (C)

Figure 4: (A): Distribution of time to detect a fault of a link in GN-U (the backbone network with uniform priorities). (B)-(C):Distribution of time to detect a fault: RT LP and RT CON (deterministic) vs. memoryless over GN-U. (B): RT LP and RT CON (C): RT-S LP and RT-S CON

be precisely computed, the deterministic optimum is NP hard to compute (Lemma 2.3). Therefore, these relations tell us that in many cases our best deterministic schedules obtained nearly optimal schedules.

8. Extension to probabilistic tests

A useful extension of our model allows for a probability π_{ei} that depends on i and e that a failure to e is found with test i. We assume that different probes invoking the same or different tests are independent. Probabilistic tests can model ECMP (equal cost multi-paths) and transient (inconsistent) failures: Transient failures are modeled by a fixed probability $\pi_{ei} \in (0, 1]$ of packet loss. Tests under ECMP are modeled by s_i being a unit flow between the origin and destination that defines a probability distribution over tests, where the "flow" traversing e is π_{ei}.

With probabilistic tests, we may as well use stochastic schedules, in particular, memoryless schedules, which also offer strong guarantees on the variance of detection times. Our models and results for memoryless schedules have straightforward extensions to probabilistic tests. The convex program for opt_M-SUM_e can be modified to incorporate probabilistic tests if we replace in (6) $\sum_{i|e\in s_i} q_i$ by $\sum_i \pi_{ei} q_i$. The LP for opt_M-MAX_e can be modified by replacing in (7) for each element e $\sum_{i|e\in s_i} q_i$ by $\sum_i \pi_{ei} q_i$.

9. Related work

This basic formulation of failure detection via probes applies in multiple network scales, from backbone networks to data centers [3, 2]. A recent application is testing of all forwarding rules in a software-defined network [4]. Beyond the detection of network failures, the fundamental optimization problems we study model classic and emerging resource replication and capacity allocation problems.

Previous considerations of the detection problem for network failures focused on MAX_e objective when all elements have equal importance (uniform priorities) [3, 2, 4]. In this particular case, deterministic scheduling is equivalent to finding a minimum size set of tests which covers all elements, which is the classic set covering problem. The optimal memoryless schedule is a solution of a simplified LP, which computes an optimal fractional cover. In practice, however, some elements are much more critical than others, and the uniform modeling does not capture that. Ideally, we would like to specify different detection-time targets for failures which depend on the criticality of the element. A set cover based deterministic schedule, however, may perform poorly when elements have different priorities and there was no efficient algorithm for constructing good deterministic schedules. Moreover, the SUM_e objectives, which were not previously considered for network failure detection application, constitute a natural global objective for overall performance, for example, when elements have associated fail probability, SUM_e minimization corresponds to minimizing expected failure detection time.

The special case of *singletons* (each test contains a single element) received considerable attention and models several important problems. The SUM_e objective on memoryless schedules is the subject of Kleinrock's well known "square root law" [5]. Scheduling for Teletext [7] and broadcast disks [8], can be formulated as deterministic scheduling of singletons. Both $\mathsf{E}_e\mathsf{E}_t$ and $\mathsf{M}_e\mathsf{M}_t$ objectives were considered. Our Kuhn-Tucker scheduler for SUM_e generalizes a classic algorithm for singletons [9, 10] which has a factor 2 approximation for the $\mathsf{E}_e\mathsf{E}_t$ [10]. Bar-Noy et al. [10, 11] established a gap ≤ 2 between the optimal deterministic and memoryless schedules, this is in contrast to the difficulty of general subset tests, where we show that gaps can be asymptotic. Interestingly, however, even for singletons, $\mathsf{M}_e\mathsf{M}_t$ optimal deterministic scheduling is NP hard [10]. Several approximation algorithms were proposed for deterministic scheduling [12, 10, 11]. In particular, Bar-Noy et al. [10, 11] proposed tree-schedules, which are an ingredient in our R-Tree schedule constructions, as a representation of deterministic schedules. Memoryless schedules with respect to the SUM_e objective modeled replication or distribution of copies of resources geared to optimize the success probabilities or search times in unstructured p2p networks [13]. Our convex program formulation extends the solution to a natural situation where each test (resource) is applicable to multiple elements (requests).

Lastly, our focus here is continuous testing, which is performed as a background process, but it is also natural to consider *one-time* testing, where a schedule is designed to be executed once [14, 15]. In [16] we study the relation of one-time and continuous testing.

Conclusion

We study the fundamental problem of continuous testing using subset tests. Our study is comprehensive and unifies models and algorithms. We reveal the relations between different objectives and between stochastic and deterministic schedules and propose efficient scheduling algorithms with provable performance guarantees. For the important application of probe scheduling for silent failure detection, we conduct simulations of our algorithms on realistic networks and demonstrate their effectiveness in varied scenarios. Beyond silent failure detection, we believe the optimization problems we address and our scheduling algorithms will find applications in other resource allocation domains.

References

[1] R. R. Kompella, J. Yates, A. G. Greenberg, A. C. Snoeren, Detection and localization of network black holes, in: INFOCOM, 2007.

[2] H. X. Nguyen, R. Teixeira, P. Thiran, C. Diot, Minimizing probing cost for detecting interface failures: Algorithms and scalability analysis, in: INFOCOM, 2009.

[3] Q. Zheng, G. Cao, Minimizing probing cost and achieving identifiability in probe based network link monitoring, IEEE Tran. Computers.

[4] H. Zeng, P. Kazemian, G. Varghese, N. McKeon, Automatic test packet generation, in: CONEXT, 2012.

[5] L. Kleinrock, Queueing Systems, Volume II: Computer Applications, Wiley-Interscience, New York, 1976.

[6] U. Feige, A threshold of $\ln n$ for approximating set cover, J. ACM 45 (1998) 634–652.

[7] M. Ammar, J. Wong, On the optimality of cyclic transmission in teletext systems, IEEE Tran. Communication 35 (1) (1987) 68–73.

[8] S. Acharya, R. Alonso, M. Franklin, S. Zdonik, Broadcast disks: data management for asymmetric communication environments, in: ACM SIGMOD, 1995.

[9] S. Hameed, N. H. Vaidya, Log-time algorithms for scheduling single and multiple channel data broadcast, in: Proc. of ACM/IEEE MobiCom, 1997.

[10] A. Bar-Noy, R. Bhatia, J. Naor, B. Schieber, Minimizing service and operation costs of periodic scheduling, Math. Oper. Res. 27 (3) (2002) 518–544.
[11] A. Bar-Noy, V. Dreizin, B. Patt-Shamir, Efficient algorithms for periodic scheduling, Computer Networks 45 (2) (2004) 155–173.
[12] C. Kenyon, N. Schabanel, N. E. Young, Polynomial-time approximation scheme for data broadcast, in: ACM STOC, 2000.
[13] E. Cohen, S. Shenker, Replication strategies in unstructured peer-to-peer networks, in: Proceedings of the ACM SIGCOMM Conference, 2002.
[14] U. Feige, L. Lovasz, P. Tetali, Approximating min-sum set cover, in: Proceedings of 5th International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX), Vol. 2462 of LLNCS, Springer, 2002, pp. 94–107.
[15] E. Cohen, A. Fiat, H. Kaplan, Efficient sequences of trials, in: Proc. 14th ACM-SIAM Symposium on Discrete Algorithms, 2003.
[16] E. Cohen, H. Kaplan, Y. Mansour, Scheduling subset tests: One-time, continuous, and how they relate, in: The 16th. International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX), 2013.

Appendix A. R-Tree schedules analysis

Appendix A.1. Tree schedules for Singleton tests

For a given instance, the best D2M we can hope for is when the deterministic scheduler is able to perform each test in precise intervals of $1/q_i$, which results, for singletons instances, in maximum probing interval of $1/q_i$. Tree schedules achieve this when $q_i = 2^{-L_i}$ for all i. A deterministic tree schedule for singletons has D2M that is at most 2, and therefore, for all our objectives, the D2M gap is at most 2.

The $\mathsf{M}_e\mathsf{M}_t$ D2M gap and the $\mathsf{E}_e\mathsf{M}_t$ D2M gap, however, are *exactly* 2. Consider an instance with two elements one with priority $p_1 = 1 - \epsilon$ and the other with priority $p_2 = \epsilon$.

Consider $\mathsf{M}_e\mathsf{M}_t$. The optimal memoryless schedule (7) has $q_1 = 1 - \epsilon$ and $q_2 = \epsilon$ and $\max_e p_e\mathsf{M}_t[e] = 1$. Whenever there are at least two elements with positive priorities, any deterministic scheduler has $\mathsf{M}_t[e] \geq 2$ for all elements. Therefore, the $\mathsf{M}_e\mathsf{M}_t$ of any deterministic schedule is at least 2 and the D2M is at least 2.

Consider $\mathsf{E}_e\mathsf{M}_t$. The optimal memoryless schedule (6) has $q_1 = \frac{\sqrt{1-\epsilon}}{\sqrt{1-\epsilon}+\sqrt{\epsilon}}$ and $q_2 = \frac{\sqrt{\epsilon}}{\sqrt{1-\epsilon}+\sqrt{\epsilon}}$ and the $\mathsf{E}_e\mathsf{M}_t = p_1/q_1 + p_2/q_2 = (\sqrt{1-\epsilon} + \sqrt{\epsilon})^2 \approx 1$. A deterministic schedule has $\mathsf{M}_t[e] \geq 2$ for both elements and thus $\mathsf{E}_e\mathsf{M}_t = p_1\mathsf{M}_t[1] + p_2\mathsf{M}_t[2] = 2$. It follows that the $\mathsf{E}_e\mathsf{M}_t$ D2M ration is $\geq 2 - \epsilon$ for any small $\epsilon > 0$.

Several deterministic schedules for singletons with ratio at most 2 (and better than 2 when possible for the particular instance, in particular when priorities are small) were previously proposed [10, 11]. Tree schedules are of interest to us here because they can be "properly" randomized to yield good performance in our treatment of general instances.

Appendix A.2. R-Tree schedules for subset tests

For a single element e, we analyze the expected (over our randomized construction of a deterministic tree schedule) maximum probe interval in the deterministic schedule. We show

Lemma Appendix A.1. *The expected maximum is $\Theta(\log \ell_e)/Q_e$, where $\ell_e = \{i \mid e \in s_i\}$. I.e., for any element e,*

$$E_{alg}[\max_t \boldsymbol{T}(e,t)] \leq c\log(\ell_e)/Q_e \ ,$$

where $\boldsymbol{T}(e,t)$ is the elapsed time from time t until e is probed.

Proof. Given a level N schedule, we say that a subinterval of $[0, 2^N - 1]$ is *hit by a test* if contains a leaf of the test. We say it is hit by an element e if it is hit by at least one test containing the element.

Consider a particular element e. We now look only at the tests which include the element. To simplify notation, let q_i, $i \in [\ell_e]$ be the frequencies of these tests, let $Q = \sum q_i$, and $q_{\max} = \max_i q_i$.

We consider the schedule for some level

$$N \in [\log_2(\frac{1}{q_{max}}), \max_i L_i] \ .$$

We will make a precise choice of N later on.

Note that any interval of size $\geq 1/q_{\max}$ must be hit by the test with maximum frequency. We are now looking to bound the distribution of the size of the largest interval that is not hit.

Consider now a subinterval $\subset [0, 2^N - 1]$ of size $D < 1/q_{\max}$. We can assume that $D = 2^j$ for some j and the interval left endpoint is an integral multiple of D.

We upper bound the probability that the interval it is not hit by e. The probability that it is not "hit" by a test with frequency q_i is $q_i D$. These probabilities of not hitting the interval by different tests are negatively correlated: conditioned on some of the tests not hitting the interval, it only makes it more likely that other tests do hit the

interval – hence, the probability that the interval is not hit by any test is at most the product $\prod_i (1 - q_i D)$, which in turn is bounded from above by $\prod_i (1 - q_i D) \leq \exp(-\sum q_i D) = \exp(-QD)$.

We now upper bound the probability that there exists at least one subinterval of size $D = 2^j$ and left endpoint that is an integral multiple of D, that is not hit by any test. We do a union bound on $2^N/D$ intervals of this property and this probability is at most

$$\frac{2^N}{D} \exp(-QD) \; . \tag{A.1}$$

Note that if using $D = \frac{x}{2}$, this upper bounds the probability that there exists an interval of size x that is not hit (without restrictions on endpoints). This probability, in terms of x, is

$$\frac{2^{N+1}}{x} \exp(-Qx/2) \tag{A.2}$$

We now restrict our attention to a subset S of the tests which satisfy $q_i \geq \frac{Q}{2\ell_e}$. We have $Q_S \equiv \sum_{i \in S} q_i \geq Q/2$. We now look only at the tests in S. since this is a subset of the tests that include e, it is sufficient to bound the expectation of the largest open interval with respect to these tests. Since the highest level in S is $N = \lceil \log_2(2\ell_e/Q) \rceil \leq 1 + \log_2(\ell_e/Q)$, we can look at the level N schedule. We substitute this N and $Q_S \geq Q/2$ in (A.2) we obtain that the probability of an empty interval of size x is

$$\frac{8\ell_e}{xQ} \exp(-xQ/4) \; . \tag{A.3}$$

For $x = 8 \ln \ell_e / Q$ in (A.3), we obtain a bound of $1/(\ell_e \ln \ell_e) \leq 1/2$ (for $\ell_e \geq 2$, $\ell_e = 1$ is already covered as q_{max}).

We can now obtain an upper bound on the expectation of the maximum empty interval by summing over positive integers i, the product of interval size $(i+1)x$ and an upper bound on the probability of an empty interval of at least size ix, for positive integer i, we obtain that the expectation is $O(x) = (1/Q)O(\ln \ell_e)$. □

Proof of Theorem 5.1

Proof. We start with frequencies $\boldsymbol{q}$ and build a deterministic tree schedule using our randomized construction. We show that the expected $\mathsf{E}_e\mathsf{M}_t$ of the deterministic schedule that we obtain is at most $\Theta(\ln \ell)$ times the $\mathsf{E}_e\mathsf{M}_t$ of the memoryless schedule for $\boldsymbol{q}$. To obtain our claim, we take $\boldsymbol{q}$ to be the optimum of (6).

We apply Lemma Appendix A.1. The lemma shows that for each element e we have $E_{alg}[\max_t \mathrm{T}(e,t)] \leq c \log(\ell_e)/Q_e$. Now we take a weighted sum over elements using $\boldsymbol{p}$. We get that,

$$E_{e \sim p_e} E_{alg}[\max_t T(e,t)] \leq \sum_e p_e \frac{c \log(\ell_e)}{Q_e}$$

This is equivalent to,

$$\begin{aligned} E_{alg} E_{e \sim p_e}[\max_t \mathrm{T}(e,t)] &\leq \sum_e c \log(\ell_e) \frac{p_e}{Q_e} \\ &\leq c \log(\ell_{\max}) \sum_e \frac{p_e}{Q_e} \; . \end{aligned}$$

This implies that with probability at least $1/2$ (over the coin flips of the algorithm) we get a deterministic schedule whose $\mathsf{E}_e\mathsf{M}_t$ is $2c \log(\ell_{\max}) \sum_e \frac{p_e}{Q_e}$. It follows that the $\mathsf{E}_e\mathsf{M}_t$ D2M ratio is at most $2c \log(\ell_{\max})$.

We now show that the $\mathsf{E}_e\mathsf{E}_t$ D2M ratio of a random tree schedule is constant with constant probability. Using the same reasoning as in the proof above for $\mathsf{E}_e\mathsf{M}_t$ it suffices to show that for each e, $E_{alg} E_t[T(e,t)] \leq c/Q_e$.

Fixing e and an arbitrary time t, as in the proof of Lemma Appendix A.1, we can easily derive that $\Pr[T(e,t) \geq D] \leq \exp -Q_e D$. In particular we get that $\Pr[T(e,t) \geq i/Q_e] \leq \exp(-i)$. So the fraction of times t in which $T(e,t) \geq i/Q_e$ is at most $\exp(-i)$. It follows that

$$E_{alg} E_t[T(e,t)] \leq (2/Q_e) \sum_i \exp(-i) \leq c/Q_e$$

for some constant c. □

Proof of Theorem 5.2

Proof. We use (A.3) in the proof of Lemma Appendix A.1. For an element e, the probability of an empty interval of size at least x is at most $\frac{8\ell_e}{xQ}\exp(-xQ/4)$. Using $x \equiv D_e = 8(\ln n + \ln \ell_e)/Q$ we obtain that there is an interval empty of tests for e of length at least D_e with probability at most $1/n^2$.

By the probability union bound over the elements we get that the probability that for all e there is no empty interval of length more than D_e is at least $1 - 1/n$. □

Appendix B. Deferred Proofs

Appendix B.1. Proof of Lemma 2.2

The proof of Lemma 2.2 will follow from two claims. The first claim shows that given a stochastic schedule we can find a distribution over test sequences of length N, such that the performance of the schedule that repeatedly samples its next N tests from this distribution approaches the performance of the stochastic schedule we started out with as N approaches infinity.

The second claim shows that given a schedule which is defined, as above, via a distribution over test sequences of length N, we can define a schedule with the same performance, such that for any fixed item e, the detection time $T(e,t)$ is the same for all times t.

We use the following definition. A stochastic *N-test schedule* $\boldsymbol{\sigma}_N$ is defined via a distribution D over test sequences of length N, and it repeatedly samples D to generate its next N tests.

Claim Appendix B.1. *Given a stochastic schedule $\boldsymbol{\sigma}$, for any $\epsilon > 0$ there exists N_ϵ, such that for any $N \geq N_\epsilon$ there is an N-test schedule $\boldsymbol{\sigma}_N$ such that for every e we have*

$$\mathsf{E}_t[e|\boldsymbol{\sigma}_N] \leq (1+\epsilon)\mathsf{E}_t[e|\boldsymbol{\sigma}] \; .$$

Proof. The next N tests of σ_N are obtained by drawing a prefix of N tests from σ. We will collect constraints on the minimum size of N_ϵ and eventually pick N_ϵ to be large enough to satisfy all these constraints.

Now, since the schedule $\boldsymbol{\sigma}_N$ samples sequences of length N repeatedly from the same distribution, we can consider the time modulus N, hence,

$$\mathsf{E}_t[e|\boldsymbol{\sigma}_N] \quad = \quad \lim_{h\to\infty}\frac{1}{h}\sum_{t=1}^{h}\mathrm{T}(e,h|\boldsymbol{\sigma}_N) = \frac{1}{N}\sum_{t=1}^{N}\mathrm{T}(e,t|\boldsymbol{\sigma}_N) \; .$$

So we have to show that for sufficiently large N

$$\frac{1}{N}\sum_{t=1}^{N}\mathrm{T}(e,t|\boldsymbol{\sigma}_N) \leq \mathsf{E}_t[e|\boldsymbol{\sigma}](1+\epsilon) \; .$$

Denote by $\mathrm{T}^*(e,t|\boldsymbol{\sigma})$ the random variable of the cover time of e at time t by the schedule σ, so $\mathsf{E}[\mathrm{T}^*(e,t|\boldsymbol{\sigma})] = \mathrm{T}(e,t|\boldsymbol{\sigma})$.

By the definition of $\boldsymbol{\sigma}_N$ we have that for any t, $1 \leq t \leq N$,

$$\mathrm{T}(e,t|\boldsymbol{\sigma}_N) \quad \leq \quad \mathrm{T}(e,t|\boldsymbol{\sigma}) + \Pr[\mathrm{T}^*(e,t|\boldsymbol{\sigma}) > N-t]\mathrm{T}(e,1|\boldsymbol{\sigma}_N) \; . \tag{B.1}$$

From Markov inequality, applied to the random variable $\mathrm{T}^*(e,1|\boldsymbol{\sigma})$, we get that $\Pr[\mathrm{T}^*(e,1|\boldsymbol{\sigma}) \geq N] \leq \frac{\mathrm{T}(e,1|\boldsymbol{\sigma})}{N}$. Picking $N_\epsilon \geq \max_e \mathrm{T}(e,1|\boldsymbol{\sigma})/\epsilon$ we have that $\Pr[\mathrm{T}^*(e,1|\boldsymbol{\sigma}) \geq N] \leq \epsilon$ for all items e. Substituting this and $t = 1$ in Equation (B.1) we get that

$$T(e,1|\boldsymbol{\sigma}_N) \leq \mathrm{T}(e,1|\boldsymbol{\sigma}) + \epsilon\mathrm{T}(e,1|\boldsymbol{\sigma}_N) \; .$$

which implies that

$$\mathrm{T}(e,1|\boldsymbol{\sigma}_N) \leq \frac{\mathrm{T}(e,1|\boldsymbol{\sigma})}{(1-\epsilon)} \; . \tag{B.2}$$

Substituting Equation (B.2) back into (B.1) we get

$$\mathrm{T}(e,t|\boldsymbol{\sigma}_N) \quad \leq \quad \mathrm{T}(e,t|\boldsymbol{\sigma}) + \Pr[\mathrm{T}^*(e,t|\boldsymbol{\sigma}) > N-t]\frac{\mathrm{T}(e,1|\boldsymbol{\sigma})}{(1-\epsilon)} \; . \tag{B.3}$$

Markov inequality, for any time t, gives

$$\Pr[\mathrm{T}^*(e,t) > N - t] \leq \min\{1, \frac{\mathrm{T}(e,t|\boldsymbol{\sigma})}{N-t+1}\} \,. \tag{B.4}$$

Substituting this in (B.3) we get

$$\mathrm{T}(e,t|\boldsymbol{\sigma}_N) \quad \leq \quad \mathrm{T}(e,t|\boldsymbol{\sigma}) + \min\{1, \frac{\mathrm{T}(e,t|\boldsymbol{\sigma})}{N-t+1}\}\frac{\mathrm{T}(e,1|\boldsymbol{\sigma})}{(1-\epsilon)} \,. \tag{B.5}$$

We now sum (B.6) over all $1 \leq t \leq N$

$$\sum_{t=1}^{N} \mathrm{T}(e,t|\boldsymbol{\sigma}_N) \quad \leq \quad \sum_{t=1}^{N} \mathrm{T}(e,t|\boldsymbol{\sigma}) + \frac{\mathrm{T}(e,1|\boldsymbol{\sigma})}{(1-\epsilon)} \sum_{t=1}^{N} \min\{1, \frac{\mathrm{T}(e,t|\boldsymbol{\sigma})}{N-t+1}\} \,. \tag{B.6}$$

Our goal now is to bound the second term on the right hand side of (B.6). Since we only consider valid schedules, for each e, there must be $N_{e,\epsilon}$ so that for all $h \geq N_{e,\epsilon}$,

$$\frac{1}{h}\sum_{t=1}^{h} \mathrm{T}(e,t|\boldsymbol{\sigma}) \leq \mathsf{E}_t[e|\boldsymbol{\sigma}](1+\epsilon) \,. \tag{B.7}$$

We will select $N_\epsilon \geq \max_e N_{e,\epsilon}$ so (B.7) holds for any $h = N \geq N_\epsilon$.

It follows that to upper bound $\sum_{t=1}^{N} \min\{1, \frac{\mathrm{T}(e,t|\boldsymbol{\sigma})}{N-t+1}\}$ we can consider the following optimization problem:

$$\max \sum_{t=1}^{N} \min\{1, \frac{x_t}{N-t+1}\} \qquad \text{s.t.} \sum_{t=1}^{N} x_t \leq B$$

where in our setting $x_t = T(e,t)$ and $B = (1+\epsilon)N\mathsf{E}_t[e|\sigma]$. We substitute $y_t = x_{N-t+1}$ and the optimization problem simplifies to

$$\max \sum_{t=1}^{N} \min\{1, \frac{y_t}{t}\} \qquad \text{s.t.} \sum_{t=1}^{N} y_t \leq B$$

The solution to the optimization is to set $y_t = t$ for $t \in [1,z]$ for the largest z such that $\sum_{j=1}^{z} j = (1+z)z/2 \leq B$, $y_{z+1} = B - (1+z)z/2$ and $y_t = 0$ for $t \geq z+2$. We get that $z \leq \sqrt{2B}$ and $\sum_{t=1}^{N} \min\{1, \frac{y_t}{t}\} \leq \sqrt{2B}+1$.

Substituting this bound back in (B.6) we get that

$$\begin{aligned}
\sum_{t=1}^{N} \mathrm{T}(e,t|\boldsymbol{\sigma}_N) \quad &\leq \quad \sum_{t=1}^{N} \mathrm{T}(e,t|\boldsymbol{\sigma}) + \frac{\mathrm{T}(e,1|\boldsymbol{\sigma})}{1-\epsilon}\left(1 + \sqrt{2N(1+\epsilon)\mathsf{E}_t[e|\boldsymbol{\sigma}]}\right) \\
&\leq \quad (1+\epsilon)N\mathsf{E}_t[e|\boldsymbol{\sigma}] + \mathrm{T}(e,1|\boldsymbol{\sigma})4\sqrt{N\mathsf{E}_t[e|\boldsymbol{\sigma}]} \qquad \text{(B.8)} \\
&= \quad N\mathsf{E}_t[e|\boldsymbol{\sigma}]\left(1 + \epsilon + \frac{4\mathrm{T}(e,1|\boldsymbol{\sigma})}{\sqrt{N\mathsf{E}_t[e|\boldsymbol{\sigma}]}}\right)
\end{aligned}$$

where inequality (B.8) is by substituting (B.7) and assuming that $\epsilon < 0.5$.

We will now set N_ϵ appropriately. First we need $N_\epsilon \geq \max_e N_{e,\epsilon}$. Second, we need that $N_\epsilon \geq \max_e \mathrm{T}(e,1)/\epsilon$. Third, N_ϵ should be large enough so that $\frac{4\mathrm{T}(e,1|\boldsymbol{\sigma})}{\sqrt{N_\epsilon \mathsf{E}_t[e|\boldsymbol{\sigma}]}} \leq \epsilon$. We get that

$$\mathsf{E}_t[e|\boldsymbol{\sigma}_N] = \frac{1}{N}\sum_{t=1}^{N} \mathrm{T}(e,t|\boldsymbol{\sigma}_N) \leq \mathsf{E}_t[e|\boldsymbol{\sigma}](1+2\epsilon)$$

from which the proof follows by using $\epsilon/2$ rather than ϵ in our constraint on N_ϵ specified above. □

A stochastic shifted N-test schedule $S(\boldsymbol{\sigma}_N)$ is defined with respect to a stochastic N-test schedule $\boldsymbol{\sigma}_N$ as follows. It samples uniformly a random $i \in [1,N]$ and a sequence x from σ_N (recall that x is an infinite sequence of tests composed from blocks of N tests) and starts from test i in x.

Claim Appendix B.2. *Given σ_N, for any t and e we have,*

$$T(e,t|S(\boldsymbol{\sigma}_N)) = \mathsf{E}_t[e|\boldsymbol{\sigma}_N] \; . \tag{B.9}$$

Proof. We first will show that $\mathrm{T}(e,t|S(\boldsymbol{\sigma}_N))$ is independent of t and then show that it equals $\mathsf{E}_t[e|\boldsymbol{\sigma}_N]$. We can see that it is independent of t by the definition of $S(\boldsymbol{\sigma}_N)$ from which we get

$$\mathrm{T}(e,t|S(\boldsymbol{\sigma}_N)) \quad = \quad \frac{1}{N}\sum_{i=1}^{N}\mathrm{T}(e,t+i|\boldsymbol{\sigma}_N) = \frac{1}{N}\sum_{i=1}^{N}\mathrm{T}(e,(t+i) \bmod N|\boldsymbol{\sigma}_N) = \frac{1}{N}\sum_{i=1}^{N}\mathrm{T}(e,i|\boldsymbol{\sigma}_N) \; .$$

Now, since the schedule σ_N samples sequences of length N, we can consider the time modulus N, hence,

$$\mathsf{E}_t[e|\sigma_N] \quad = \quad \lim_{h\to\infty}\frac{1}{h}\sum_{t=1}^{h}\mathrm{T}(e,h|\boldsymbol{\sigma}_N) = \frac{1}{N}\sum_{i=1}^{N}\mathrm{T}(e,i|\boldsymbol{\sigma}_N)$$

□

Appendix B.2. Proof of Lemma 2.3

Proof. We obtain a scheduling instance using the same set of elements and subsets (tests) as the X3C instance. We use a uniform $\boldsymbol{p}$ over elements with $p_e = 1/(3k)$ for SUM_e objectives and $p_e = 1$ for MAX_e objectives.

We first consider deterministic schedules. From an exact cover, we define a deterministic schedule by cycling through the same permutation of the cover. The deterministic schedule has $\mathsf{M}_t[e] = k$ and $\mathsf{E}_t[e] = (k+1)/2$ for all elements e. The maximum $\max_e T(e,t)$ at any time t is k and the average is $(k+1)/2$. Therefore, the schedule has $\mathsf{M}_e\mathsf{M}_t$, $\mathsf{E}_t\mathsf{M}_e$, and $\mathsf{E}_e\mathsf{M}_t$ equal to k and $\mathsf{M}_e\mathsf{E}_t$, $\mathsf{E}_e\mathsf{E}_t$, and $\mathsf{M}_t\mathsf{E}_e$ equal to $(k+1)/2$.

Consider an arbitrary deterministic schedule and time t. We must have $\max_e T(e,t) \geq k$, since at most $3i$ elements can be covered in i probes, so to cover all $3k$ elements we need at least k probes. We have equality if and only if the sequence of k probes following t constitutes a cover. A cover of size k must be an exact cover. Therefore $\mathsf{E}_t\mathsf{M}_e{=}\, k$ implies exact cover of size k.

Similarly, we claim that on any schedule, $(1/k)\sum_e T(e,t) \geq (k+1)/2$. This is because $\sum_e T(e,t) = \sum_e m_e$, where m_e is the smallest d such that $e \in \sigma_{t+d}$. Since there can be at most 3 elements of each value of $m_e \geq 1$, we have that $\sum_e T(e,t) \geq 3\sum_{d=1}^{k} d = 3k(k+1)/2$ and our claim follows. Moreover, equality holds only if the sequence of k probes from t on is an exact cover. Therefore $\mathsf{M}_t\mathsf{E}_e{=}\,(k+1)/2$ implies exact cover of size k.

Consider an arbitrary deterministic schedule and let q_e be the average probing frequency of element e (recall that we only consider valid schedules, where q_e is well defined). We have $\mathsf{M}_t[e] \geq 1/q_e$ and $\mathsf{E}_t[e] \geq (1+1/q_e)/2$. Moreover, equality can hold only when $1/q_e$ is integral and probes are evenly spaced every $1/q_e$ probes except for vanishingly small fraction of times. For the X3C instance we have $\sum_e q_e = 3$, and from convexity, $\sum_e 1/q_e$ or $\max_e 1/q_e$ are minimized only when all q_e are equal to $1/k$. This means that the $\mathsf{M}_e\mathsf{M}_t$ and $\mathsf{E}_e\mathsf{M}_t$ can be equal to k or the $\mathsf{M}_e\mathsf{E}_t$ and $\mathsf{E}_e\mathsf{E}_t$ are equal to $(k+1)/2$ equal to $(k+1)/2$ only if each element is probed every k probes (except vanishingly small) number of times. This means that most sequences of k consecutive probes constitute an exact cover.

We now consider stochastic schedules. From an exact cover, we define a stochastic schedule by a uniform distribution $(1/k)$ on each of the k shifts of the same permutation of the cover. On this schedule, all our objectives have value $(k+1)/2$. It remains to show that for each of the objectives, a schedule with time $(k+1)/2$ implies an exact cover.

Observe that with our choice of weighting, on any schedule opt-$\mathrm{MAX}_e{\geq}$ opt-SUM_e. Therefore if the stochastic optimum of either the SUM_e or MAX_e objectives is $(k{+}1)/2$, then opt-SUM_e is also $(k{+}1)/2$ which implies, from (11), that opt_D-$\mathsf{E}_e\mathsf{E}_t{=}\,(k+1)/2$, which implies exact cover. □

Appendix B.3. Proof of Lemma 4.1

Proof. We first establish (11). We show that given a stochastic schedule $\boldsymbol{\sigma}$ and $\delta > 0$, we can construct a deterministic schedule $\boldsymbol{\sigma}_D$, such that $\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}_D] \leq (1+\delta)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]$. The main difficulty which makes this proof more technical stems from existence of valid stochastic schedules with deterministic instantiations which are not valid (limits and frequencies are not well defined). Therefore, we can not simply assume a positive probability of a (valid) deterministic schedule with an average cost that is close to that of $\boldsymbol{\sigma}$.

Our construction consists of several steps. We first show that there is a deterministic testing sequence[9] $\boldsymbol{\sigma}'$ so that the average cost on the first N time steps (for sufficiently large N that depends on ϵ) is within $(1+\epsilon)$ of that of the stochastic schedule. We then focus on a sub-sequence of steps $[t_0, N]$ of size $\Omega(N)$ so that the average property still holds and in addition, the cost of steps t_0 is at most a constant times the average. We then argue that the maximum interval between tests of an element on the prefix of σ' is bounded by a value $X = O(\sqrt{N})$. Lastly, we obtain σ_D as a cyclic schedule which repeats steps $[t_0, N]$ of σ'. We show that the average cost is within $(1 + O(\epsilon))$ from the average cost on times $[t_0, N]$ of σ' which in turn, is within $(1+\epsilon)$ to the average cost of the original $\boldsymbol{\sigma}$.

From $\boldsymbol{\sigma}$ being valid, there must be $N_{EE} > 0$ such that for all $N \geq N_{EE}$

$$\frac{1}{N}\sum_{t=1}^{N}\sum_{e} p_e \mathrm{T}(e,t) \leq (1+\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \,.$$

Fix some $N \geq N_{EE}$. We draw a particular execution of $\boldsymbol{\sigma}$ obtaining an infinite deterministic sequence σ'. From Markov inequality, with probability at least $1 - (1+2\epsilon)/(1+\epsilon) > 0$,

$$\frac{1}{N}\sum_{t=1}^{N}\sum_{e} p_e \mathrm{T}(e,t|\sigma') \leq (1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \,. \tag{B.10}$$

We therefore assume that we have a sequence σ' which satisfies (B.10).

We now focus on a subset $[t_0, N]$ of time steps, where t_0 is the minimum t such that $\sum_e p_e \mathrm{T}(e,t|\sigma') \leq 10\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]$. From (B.10), assuming $\epsilon \leq 1/2$, it follows that $t_0 \leq 0.2N$. Let $N' = N - t_0 + 1 \geq 0.8N$ be the length of the interval $[t_0, N]$. We establish that

$$\frac{1}{N'}\sum_{t=t_0}^{N}\sum_{e} p_e \mathrm{T}(e,t|\sigma') \leq (1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \,. \tag{B.11}$$

We establish (B.11) using (B.10):

$$\begin{aligned}
\sum_{t=t_0}^{N}\sum_{e} p_e \mathrm{T}(e,t|\sigma') &= \sum_{t=1}^{N}\sum_{e} p_e \mathrm{T}(e,t|\sigma') - \sum_{t=1}^{t_0-1}\sum_{e} p_e \mathrm{T}(e,t|\sigma') \\
&\leq N(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] - (t_0-1)10\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \\
&= N'(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \,.
\end{aligned}$$

We now bound the maximum elapsed times between tests of an element e in the sequence σ' in the time interval $[t_0, N]$. Consider an interval $[i, i + x_e - 1]$ of x_e time steps, completely contained in $[t_0, N]$ (that is $i + x_e - 1 \leq N$) in which element e is not tested then

$$\sum_{t=i}^{i+x_e-1} \mathrm{T}(e,t|\sigma') = \sum_{j=1}^{x_e} j \geq x_e^2/2 \,. \tag{B.12}$$

On the other hand, since σ' satisfies (B.10), noting that $i + x_e - 1 \leq N$, we must have

$$p_e \sum_{t=i}^{i+x_e-1} \mathrm{T}(e,t|\sigma') \leq \sum_{t=1}^{N}\sum_{e} p_e \mathrm{T}(e,t|\sigma') \leq N(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] \,. \tag{B.13}$$

Combining (B.12) and (B.13), we obtain that

$$x_e \leq \sqrt{\frac{2(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]N}{p_e}} \,. \tag{B.14}$$

[9] We use the term sequence rather than schedule because the sequence may not be a valid schedule.

Let

$$X = \sqrt{\frac{2(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]N}{\min_e p_e}}\ , \tag{B.15}$$

we established that

$$\forall e \forall t \in [t_0, N-X+1],\ \mathrm{T}(e,t|\sigma') \leq X\ . \tag{B.16}$$

Lastly, we define the deterministic schedule $\boldsymbol{\sigma}_D$ which cycles through the steps $[t_0, N]$ of σ'. Since $\boldsymbol{\sigma}_D$ is cyclic, we have

$$\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}_D] = \frac{1}{N'}\sum_{t=1}^{N'}\sum_e p_e \mathrm{T}(e,t|\boldsymbol{\sigma}_D)\ . \tag{B.17}$$

We therefore upper bound the latter by relating it to σ'.

$$\begin{aligned}
\sum_{t=1}^{N'}\sum_e p_e\mathrm{T}(e,t|\boldsymbol{\sigma}_D) &\leq \sum_{t=t_0}^{N}\sum_e p_e\mathrm{T}(e,t|\sigma') + X\sum_e p_e\mathrm{T}(e,t_0|\sigma') \\
&\leq \sum_{t=t_0}^{N}\sum_e p_e\mathrm{T}(e,t|\sigma') + 10X\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] &\text{(B.18)}\\
&\leq N'(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] + \epsilon N\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}] &\text{(B.19)}\\
&\leq N'\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}](1+2\epsilon+\epsilon(N/N')) \\
&\leq N'\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}](1+4\epsilon)\ . &\text{(B.20)}
\end{aligned}$$

To verify the first inequality, we apply (B.16) obtaining that for $t \leq N-X+1$, $\mathrm{T}(e,t-t_0+1|\boldsymbol{\sigma}_D) = \mathrm{T}(e,t|\sigma')$. For the remaining X time steps that correspond to $t \in (N-X+1, N]$ of σ' ($t \in (N'-X+1, N']$ of $\boldsymbol{\sigma}_D$) we have

$$\mathrm{T}(e,t-t_0+1|\boldsymbol{\sigma}_D) \leq \begin{cases} \mathrm{T}(e,t|\sigma'),\ \text{if } \mathrm{T}(e,t|\sigma') \leq N-t \\ N-t+\mathrm{T}(e,1|\boldsymbol{\sigma}_D)\ \text{, otherwise.}\end{cases}$$

$$\leq \mathrm{T}(e,t|\sigma') + \mathrm{T}(e,t_0|\boldsymbol{\sigma}_D) = \mathrm{T}(e,t|\sigma') + \mathrm{T}(e,t_0|\sigma')\ .$$

Inequality (B.18) follows from our choice of t_0. Inequality (B.19) holds if we choose

$$N \geq \frac{200(1+2\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]}{\epsilon^2\min_e p_e}\ ,$$

to guarantee that $10X \leq \epsilon N$. Lastly, (B.20) uses $N' \geq 0.8N$. Combining (B.20) with (B.17), we obtain $\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}_D] \leq (1+4\epsilon)\mathsf{E}_e\mathsf{E}_t[\boldsymbol{\sigma}]$. We conclude the proof of (11) by choosing $\epsilon = \delta/4$.

We now establish the inequalities (12) and (13). Given a deterministic schedule $\boldsymbol{\sigma}$, we define a cyclic deterministic schedule $\boldsymbol{\sigma}_C$ which repeats a sequence $\boldsymbol{\sigma}'_C$ of some length N and which satisfies $\forall e,\ \mathsf{M}_t[e|\boldsymbol{\sigma}_C] \leq \mathsf{M}_t[e|\boldsymbol{\sigma}]$. Consider the schedule $\boldsymbol{\sigma}$ and associate a state with each time t, which is a vector that for each e, contains the elapsed number of steps since a test for e was last invoked. At $t=1$ we have the all zeros vector. When a test s is invoked, the entries for all elements in s are reset to 0 and the entries of all other elements are incremented by 1. From definition, the maximum value for entry e is $\mathsf{M}_t[e]$. Therefore, there is a finite number of states. The segment $\boldsymbol{\sigma}'_C$ is any sequence between two times with the same state. It is easy to see that the cyclic schedule $\boldsymbol{\sigma}_C$ obtained from $\boldsymbol{\sigma}'_C$ has the desired property.

We now take the deterministic cyclic schedule $\boldsymbol{\sigma}_C$ and construct a stochastic schedule $\boldsymbol{\sigma}'$ by selecting a start point $i \in N$ uniformly at random, executing steps $[i, N]$ of $\boldsymbol{\sigma}'_C$, and then using $\boldsymbol{\sigma}_C$. For each element e, we have

$$\mathsf{M}_t[e|\boldsymbol{\sigma}'] \leq \frac{\mathsf{M}_t[e|\boldsymbol{\sigma}_C]+1}{2} \leq \frac{\mathsf{M}_t[e|\boldsymbol{\sigma}]+1}{2}\ .$$

By combining,

$$\text{opt-}\mathsf{E}_e\mathsf{M}_t \leq \sum_e p_e\mathsf{M}_t[e|\boldsymbol{\sigma}'] \leq \sum_e \frac{\mathsf{M}_t[e|\boldsymbol{\sigma}]+1}{2} = (\mathsf{E}_e\mathsf{M}_t[\boldsymbol{\sigma}]+1)/2\ .$$

By taking the infimum of $\mathsf{E}_e\mathsf{M}_t$ over all deterministic schedules we conclude the claim. The argument for $\mathsf{M}_e\mathsf{M}_t$ is similar. □

Appendix B.4. Proof of Lemma 4.6

Proof. We choose n, $\ell \geq 1$, and $m \geq 2\ell$ such that $n = \binom{m}{\ell}$, and construct an instance with n elements and m tests such that each element is included in exactly ℓ test and every subset of ℓ tests has exactly one common element. We use a uniform $\boldsymbol{p}$.

The instance is symmetric and therefore in the solution of the convex program (6) or (7) all the m tests have equal rates $q = 1/m$. The memoryless schedule with this $\boldsymbol{q}$ optimizes both SUM_e and MAX_e for $\boldsymbol{p}$. For any element and any time, the expected detection time by a memoryless schedule with $\boldsymbol{q}$ is m/ℓ. But for any particular deterministic schedule and a particular time there is an element that requires $m - \ell$ probes (for any sequence of $m - \ell$ tests there must be at least ℓ tests not included and we take the element in the intersection of these tests. This means that at any time, the worst-case element detection time is factor $\frac{m-\ell}{m/\ell} \geq \ell/2 = \Theta(\ln n/\ln m)$ larger than the memoryless optimum.

When fixing the number of tests m, this is maximized (Sperner's Theorem) with $\ell = m/2$ and the MAX_e ratio is $\Theta(m)$. When fixing the number of elements n, the maximum ratio is $\arg\max_\ell n = \binom{2\ell}{\ell}$ and we obtain $\ell = \Theta(\ln n)$.

We use the same construction as in Lemma 4.6 and take a uniform $\boldsymbol{p}$ over elements. Lastly, to show $\mathsf{E}_e\mathsf{M}_t$ of $\Omega(\ln \ell)$ consider a sequence of m probes. The expectation over elements of the number of probes that test the element, is at most ℓ. So at least half the elements are probed at most ℓ times. There are at least $m/2$ distinct tests. The expected over e maximum difference between probes to an element e over a sequence of m is $\Omega(\ln \ell)m/\ell$. This is because every combination of ℓ distinct probes corresponds to an element, and thus, for the "average" element, the probes can be viewed as randomly placed, making the expectation of the maximum interval a logarithmic factor larger than the expectation.

We now show how the instances can be realized on a network. We use n pairs of links. Each pair includes a link which corresponds to an element in our instance and a "dummy" link. The pairs are connected on a path of size n. Each test is an end to end path which traverses one link from each pair. Each (real) link is covered by exactly ℓ paths and every subset of ℓ paths has one common (real) link. The network is a path of length n of pairs of parallel links, a real and a dummy link. Real links e have $p_e = 1$ and the dummy link have $p_e = +\infty$. (If we want to work with respect to some SUM_e optimum we take $p_e \equiv p$ (for some $p < 1$) for real links and $p_e = 0$ for dummy links). Each path traverses one link from each pair and includes ℓ real links. □

The Lemma is tight in the sense that it is always possible to get a schedule with D2M equal to m by cycling over a permutation of the tests.