Staying up to Date with Online Content Changes Using Reinforcement Learning for Scheduling

Andrey Kolobov [1] **Yuval Peres** [2] **Cheng Lu** [3] **Eric Horvitz** [4]

Abstract

From traditional Web search to virtual assistants and Web accelerators, services that rely on online information need to keep track of continuing content changes. Many of these services need to explicitly request content updates from remote sources (e.g., web pages), which gives rise to a formidable challenge. A search engine, for example, may track billions of sources, with content change timing or even frequency being initially unknown for most. How should it schedule requests for updates under constraints of network bandwidth? We introduce a novel optimization objective for this setting that has several desirable properties, and efficient algorithms for it with optimality guarantees even in the face of mixed content change observability and initially unknown model parameters. An evaluation on a set of 20M web pages crawled daily for 14 weeks shows the scalability and other properties of our approach.

1. Introduction

As the Web becomes more and more dynamic, services that rely on web data face the increasingly challenging problem of keeping up with online content changes. Whether it be a continuous-query system (Pandey et al., 2003), a virtual assistant like Cortana or Google Now, or an Internet search engine, such a service tracks many remote sources of information – web pages or data streams (Pandey et al., 2004). Users expect these services, which we call *trackers*, to be surface the latest information that appears at the sources. This is easy to do when the remote sources *push* content updates to the tracker. Unfortunately, the push model is impractical for maintaining databases as large as major search engines' indexes (Baeza-Yates & Ribeiro-Neto, 1999) and conversation agents' knowledge graphs (Hixon et al., 2015). Instead, for all sources they track, these services must continually decide when to re-request (*crawl*) their data in order pick up the changes. A policy that makes these decisions well solves *freshness crawl scheduling problem*.

Freshness crawl scheduling has several challenging aspects. For most sources, the tracker finds out whether they have changed only when it crawls them. To guess when the changes happen, and hence should be downloaded, the tracker needs a predictive model whose parameters are initially unknown. Thus, the tracker needs to learn these models *and* optimize a freshness-related objective when scheduling crawls. For some web pages, however, sitemap polling and other means can provide trustworthy near-instantaneous signals that the page has changed in *a* meaningful way, though not what the change is exactly. But even with these remote change observations and known change model parameters, freshness crawl scheduling remains highly nontrivial because the tracker cannot react to every individual predicted or actual change. The tracker's infrastructure imposes a *bandwidth constraint* on the average daily number of crawls, usually limiting it to a fraction of the change event volume. Last but not least, Google and Bing track many billions of pages (van den Bosch et al., 2015) with vastly different importance and change frequency characteristics. The sheer size of this constrained learning and optimization problem makes low-polynomial algorithms strongly preferable, despite the availability of big-data platforms.

This paper presents a holistic approach to freshness crawl scheduling that handles all of the above aspects in a computationally efficient manner with optimality guarantees using a type of reinforcement learning (Sutton & Barto, 1998). Different aspects of crawl scheduling have been studied extensively before, as described in detail in the Related Work section. Crawl schedule optimization alone, under various objectives and assuming known model parameters, has been the focus of many papers since Coffman et al. (1998). However, even without remote change sensing and under the known-model assumption, efficient optimization has so far been possible only for freshness objectives that induce policies with practically undesirable behaviors such as crawl-starving many of the sources forever (Azar et al., 2018). Learning change-predictive models has received much attention as well (e.g., (Cho & Garcia-Molina, 2003b)) but purely as a preprocessing step, without regard to the learning versus schedule optimization tradeoff. No prior work has studied joint optimization of all facets of freshness crawl scheduling that we consider here. Specifically, our

[1]akolobov@microsoft.com, Microsoft Research, Redmond, WA, USA [2]At Microsoft Research at the time of work on this publication [3]Cheng.Lu@microsoft.com, Microsoft Bing, Bellevue, WA, USA [4]horvitz@microsoft.com, Microsoft Research, Redmond, WA, USA. Correspondence to: Andrey Kolobov <akolobov@microsoft.com>.

contributions are as follows:

- We propose a natural freshness optimization objective based on harmonic numbers, and show how its mathematical properties enable efficient optimal scheduling.
- We derive efficient optimization procedures for this bandwidth-constrained objective under complete, mixed, and lacking remote change observability.
- We present a reinforcement learning algorithm that integrates these approaches with model estimation of Cho & Garcia-Molina (2003b) and converges to the optimal policy, lifting the known-parameter assumption.
- We introduce an approximate crawl scheduling algorithm that requires learning far fewer parameters, and derive a condition under which its solution is optimal.

2. Problem formalization

In settings we consider, a service we call *tracker* monitors a set W of information sources. A source $w \in W$ can be a web page, an RSS feed, a data stream, a file, etc, whose content changes from time to time. To pick up changes from a source, the tracker needs to *crawl* it, i.e., download its content. At any time when source w has changes that the tracker hasn't picked up, the tracker is *stale* w.r.t. w; otherwise, it is *fresh* w.r.t. w. We assume crawl operations to be near-instantaneous, and the set W of sources to be fixed. Growing W to improve *information completeness* (Pandey et al., 2004) is also an important but largely distinct problem; we do not consider it here.

Discrete page changes. We define a *content change* at a source to be an alteration at least minimally important to the tracker. In practice, trackers compute a digest of a source's content using data extractors, *shingles* (Broder et al., 1997), or *similarity hashes* (Charikar, 2002), and consider content changed when its digest changes. We assume changes to be instantaneous and associate change time t_c with each.

Models of change process and importance. We model each source $w \in W$'s changes as a Poisson process with *change rate* Δ_w. We also associate an *importance* score μ_w with each source, and denote these parameters jointly as $\vec{\mu}$. Importance score μ_w can be thought of as characterizing the time-homogeneous Poisson rate at which the page is served in response to the query stream, although in general it can be any positive weight measuring the significance of a page (Azar et al., 2018). While scores μ_w are defined, and hence known, by the tracker itself, change rates Δ_w generally need to be learned.

Change observability. For most sources, the tracker can find out whether the source has changed only by crawling it. In this case, even crawling doesn't tell the tracker how many times the source has changed since the last crawl. We denote the set of these sources as $W = W^-$ and say that the tracker receives *incomplete change observations* about them. However, for other sources, which we denote as W^o, the tracker may receive near-instant notification whenever they change, i.e., get *complete remote change observations*. E.g., for web pages these signals may be available from browser telemetry or sitemaps. Thus the tracker's set of sources can be represented as $W = W^o \cup W^-$ and $W^o \cap W^- = 0$.

Bandwidth constraints. Even if the tracker receives complete change observations, it generally cannot afford to do a crawl upon each of them. The tracker's network infrastructure and considerations of respect to other Internet users limit its *crawl rate* – the average number of requests per day – and the total change rate of tracked sources may be much higher. We call this limit *bandwidth constraint* R.

Optimizing freshness. We assume that the tracker operates in continuous time and starts fresh w.r.t. all tracked sources. A solution to the freshness crawl scheduling problem is a policy π – a rule that at every instant t chooses (potentially stochastically) a source to crawl or decides that none should be crawled. Executing π produces a *crawl sequence* of time-source pairs $CrSeq = (t_1, w_1), (t_2, w_2), \ldots$, where we omit the time points for which π does not choose a source to crawl. For a specific source w, we denote this sequence as $CrSeq_w = (t_1, w), (t_2, w), \ldots$ Similarly, the (Poisson) change process at the sources generates a change sequence $ChSeq = (t'_1, w'_1), (t'_2, w'_2), \ldots$, where each t'_i is a change time of source w'_i, and its restriction to source w is $ChSeq_w$. We denote the joint process governing changes at all sources as $P(\vec{\Delta})$.

3. Minimizing harmonic staleness penalty

We view maximizing freshness as minimizing costs the tracker incurs for the lack thereof, and associate the following *policy cost* J^π with every scheduling policy π, the *time-averaged expected staleness penalty*:

$$J^\pi = \lim_{T\to\infty} \mathop{\mathbb{E}}_{\substack{CrSeq\sim\pi,\\ ChSeq\sim P(\vec{\Delta})}} \left[\frac{1}{T}\int_0^T \sum_{w\in W} \mu_w C(N_w(t))dt\right] \tag{1}$$

Here, T is a planning horizon, $N_w(t)$ is the number of uncrawled changes source w has accumulated by time t, and $C : \mathbb{Z}^+ \to \mathbb{R}^+$ is a *penalty function*, to be chosen later, that assigns a cost to every possible number of uncrawled changes. Note that $N_w(t)$ implicitly depends on the most recent time w was crawled as well as on change sequence $ChSeq$, so the expectation is taken both over possible change sequences *and* possible crawl sequences $CrSeq$ generatable by π. Minimizing staleness means finding

$$\pi^* = \operatorname*{argmin}_{\pi\in\Pi} J^\pi, \tag{2}$$

subject to bandwidth constraints, where Π is a suitably chosen policy class.

Choosing $C(n)$ that is efficient to optimize *and* induces

"well-behaving" policies is of utmost importance. Cho & Garcia-Molina (2003a) and Azar et al. (2018) have considered Equation 1 with $C(n) = \mathbb{1}_{n>0}$, i.e., imposing a penalty if a source had any changes independently of their number, with $\Pi = \{CrSeq_w \sim Poisson(\rho_w)$ for all $w \in W|\vec{\rho}_w\}$, i.e., policies that crawl each source according to a Poisson process with a source-specific rate ρ_w. Under a bandwidth constraint only, they find the optimal policy in this class very efficiently, in $O(|W|\log(|W|))$ time, but this solution assigns $\rho_w = 0$ for many sources (Azar et al., 2018; Cho & Garcia-Molina, 2003a). In practice, this may not be acceptable, as it leaves the tracker stale w.r.t. these sources forever and therefore raises a question: why does the tracker monitor these sources at all?

In this paper, we propose and analyze the following penalty:

$$C(n) = H(n) = \sum_{i=1}^{n} \frac{1}{n} \text{ if } n > 0, \text{ and } 0 \text{ if } n = 0 \quad (3)$$

$H(n)$ for $n > 0$ is known as the n-th harmonic number and has several desirable properties as staleness penalty:

It is strictly monotonically increasing. Thus, it penalizes the tracker for every change that happened at a source since the previous crawl, not just the first one as in (Cho & Garcia-Molina, 2003a).

It is discrete-concave, providing diminishing penalties. This reflects the intuition that while all undownloaded changes at a source matter, the first one matters most, as it marks the transition from freshness to staleness.

"Good" policies w.r.t. this objective don't starve any source:

Proposition 1. *Under $C(n) = H(n)$, any policy that crawls each source at a fixed Poisson rate ρ_w and assigns $\rho_w > 0$ to each w with $\mu_w, \Delta_w > 0$ is strictly preferable to any such policy that assigns $\rho_w = 0$ to any such source.*

Proof. See the supplement. This is a direct consequence of Proposition 2: any policy π with $\rho_w = 0$ has $J^\pi = \infty$. ■

$C(n) = H(n)$ allows for efficiently finding optimal policies under practical policy classes. Indeed, $C(n) = H(n)$ isn't the only penalty function satisfying the above properties. For instance, $C(n) = n^d$ for $0 < d < 1$ and $C(n) = \log_d(1 + n)$ for $d > 1$ behave similarly. However, optimizing these alternatives turns out much less efficient.

4. Optimization under known change process

We now derive procedures for optimizing Equation 1 with $C(n) = H(n)$ (Equation 3) under the bandwidth constraint for sources with incomplete and complete change observations, assuming that we know the change process parameters $\vec{\Delta}$ exactly. In Section 5 we will lift the known-parameters assumption. We assume $\vec{\mu}, \vec{\Delta} > 0$, because sources that are unimportant or never change don't need to be crawled.

4.1. Case of incomplete change observations

When the tracker's only way to find out about changes at a source is crawling it, we consider the class of randomized memoryless policies that sample crawl times for each source according to a Poisson process with parameter ρ_w:

$$\Pi^- = \{CrSeq_w \sim Poisson(\rho_w)\ \forall w \in W^-|\vec{\rho} \geq 0\} \quad (4)$$

This policy class reflects the intuition that, since each source changes according to a Poisson process, i.e., roughly periodically, it should also be crawled roughly periodically. In fact, as Azar et al. (2018) show, any $\pi \in \Pi^-$ can be derandomized into a deterministic policy that is approximately periodic for each w. Since every $\pi \in \Pi^-$ is fully determined by the corresponding vector $\vec{\rho}$, we can easily express a bandwidth constraint on $\pi \in \Pi^-$ as $\sum_{w\in W^-} \rho_w = R$.

To optimize over Π^-, we first express the cost function from Equation 1 in terms of Π^-'s policy parameters $\vec{\rho} \geq 0$:

Proposition 2. *For $\pi \in \Pi^-$, J^π from Eq. 1 is equivalent to*

$$J^\pi = -\sum_{w\in W^-} \mu_w \ln\left(\frac{\rho_w}{\Delta_w + \rho_w}\right) \quad (5)$$

Proof. See the Supplement. Note that $J^\pi = \infty$ if $\rho = 0$ for any $w \in W^-$. The proof proceeds via a series of algebraic manipulations and relies on properties of Poisson processes, particularly their memorylessness. ■

Thus, finding $\pi^* \in \Pi^-$ can be formalized as follows:

Problem 1. *[Finding $\pi^* \in \Pi^-$]*

INPUT: *bandwidth $R > 0$; positive importance and change rate vectors $\vec{\mu}, \vec{\Delta} > 0$.*

OUTPUT: *Crawl rates $\vec{\rho} = (\rho_w)_{w\in W^-}$ maximizing*

$$\overline{J}^\pi = -J^\pi = \sum_{w\in W^-} \mu_w \ln\left(\frac{\rho_w}{\Delta_w + \rho_w}\right) \quad (6)$$

subject to

$$\sum_{w\in W^-} \rho_w = R, \rho_w \geq 0 \text{ for all } w \in W^-.$$

The next result readily identifies the optimal solution to this problem:

Proposition 3. *For $\vec{\mu}, \vec{\Delta} > 0$, policy $\pi^* \in \Pi^-$ parameterized by $\vec{\rho}^* > 0$ that satisfies*

$$\begin{cases} \rho_w = \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda}}}{2}, & \text{for all } w \in W^- \\ \sum_{w\in W^-} \rho_w = R \end{cases} \quad (7)$$

is unique, minimizes the harmonic penalty J^π in Equation 3, and is therefore optimal in Π^-.

Proof. See the Supplement. The main insight is that for any $\vec{\mu}, \vec{\Delta} > 0$ the Lagrange multiplier method, which gives

rise to Equation system 7, identifies the only maximizer of $\overline{J}^{\pi} = -J^{\pi}$ (Equation 6) in the region $\vec{\rho} > 0$, which must therefore correspond to $\pi^* \in \Pi^-$. Crucially, that solution always has $\lambda > 0$. ■

Equation system 7 is non-linear, but the r.h.s. of equations involving λ monotonically decreases in $\lambda > 0$, so, e.g., bisection search (Burden & Faires, 1985) on $\lambda > 0$ can find $\vec{\rho}^*$ as in Algorithm 1.

Algorithm 1: LAMBDACRAWL-INCOMLOBS: finding the optimal crawl scheduling policy $\pi^* \in \Pi^-$ under incomplete change observations (Problem 1)

Input: $R \geq 0$ – bandwidth;
$\vec{\mu} > 0, \vec{\Delta} > 0$ – importance and change rates;
$\epsilon > 0$ – desired precision on λ
; **Output:** $\vec{\rho}$ – vector of crawl rates for each source.
lower_bound_$\lambda \leftarrow \frac{|W^-|^2 \min_{w \in W^-}\{\Delta_w\} \min_{w \in W^-}\{\mu_w\}}{|W^-| \max_{w \in W^-}\{\Delta_w\} R + R^2}$
upper_bound_$\lambda \leftarrow \frac{|W^-|^2 \max_{w \in W^-}\{\Delta_w\} \max_{w \in W^-}\{\mu_w\}}{|W^-| \min_{w \in W^-}\{\Delta_w\} R + R^2}$
$\lambda \leftarrow$ BisectionSearch(lower_bound_λ, upper_bound_λ, ϵ)
// see, e.g., Burden & Faires (1985)
foreach $w \in W^-$ **do** $\rho_w \leftarrow \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w \Delta_w}{\lambda}}}{2}$
Return $\vec{\rho}$

Proposition 4. *LAMBDACRAWL-INCOMLOBS (Algorithm 1) finds an arbitrarily close approximation to Problem 1's optimal solution in time* $O(\log_2(\frac{\textit{upper\_bound\_}\lambda - \textit{lower\_bound\_}\lambda}{\epsilon})|W^-|)$.

Proof. See the Supplement. The key step is showing that the solution λ is in $[\text{lower\_bound\_}\lambda, \text{upper\_bound\_}\lambda]$. ■

4.2. Case of complete change observations

When the tracker receives an observation every time a source changes, the policy class Π^- in Equation 4 is clearly suboptimal, because it completely ignores change signals when deciding when to crawl a source. At the same time, crawling every source on every change signal is unviable, because the total change rate of all sources $\sum_{w \in W} \Delta_w$ can easily exceed bandwidth R. These extremes suggest a policy class whose members trigger crawls for only a fraction of the observations, dictated by a source-specific probability p_w:

$$\Pi^o = \{\text{for all } w \in W^o, \text{whenever change observation } o_w \text{ arrives, crawl } w \text{ with probability } p_w \mid 0 \leq \vec{p} \leq 1\}$$

As with Π^-, to find $\pi^* \in \Pi^o$ we first express J^{π} from Equation 1 in terms of Π^o's policy parameters $\vec{p} = (p_w)_{w \in W^o}$:

Proposition 5. *For $\pi \in \Pi^o$, J^{π} from Eq. 1 is equivalent to*

$$J^{\pi} = -\sum_{w \in W^o} \mu_w \ln(p_w) \tag{8}$$

if $\vec{p} > 0$ and $J^{\pi} = \infty$ if $p_w = 0$ for any $w \in W^o$.

Proof. See the Supplement. The key insight is that under any $\pi \in \Pi^o$, the number of w's uncrawled changes at time t is geometrically distributed with parameter p_w. ■

Under any $\pi \in \Pi^o$, the crawl rate ρ_w of any source w is related to its change rate Δ_w: every time w changes we get an observation and crawl w with probability p_w. Thus, $\rho_w = p_w \Delta_w$. However, since p_w is a probability, we must have $0 \leq p_w \leq 1$. Also, bandwidth $R > \sum_{w \in W^o} \Delta_w$ isn't sensible, because with complete change observations the tracker doesn't benefit from more crawls than there are changes. Thus, we frame finding $\pi^* \in \Pi^o$ as follows:

Problem 2. *[Finding $\pi^* \in \Pi^o$]*

INPUT: *bandwidth R s.t. $0 < R \leq \sum_{w \in W^o}$; importance and change rate vectors $\vec{\mu}, \vec{\Delta} > 0$.*

OUTPUT: *Crawl probabilities $\vec{p} = (p_w)_{w \in W^o}$ maximizing*

$$\overline{J}^{\pi} = -J^{\pi} = \sum_{w \in W^o} \mu_w \ln(p_w) \tag{9}$$

subject to

$$\sum_{w \in W^o} p_w \Delta_w = R, \;\; 0 \leq p_w \leq 1 \textit{ for all } w \in W^o$$

Solving Problem 2 requires non-linear optimization under inequality constraints, which, in general, could take time exponential in the constraint number. Our main result in this subsection is that one can find the optimal solution to Problem 2 in polynomial time in the number of constraints.

We begin by inspecting solutions to the relaxation of Problem 2 that ignores the inequality constraints:

Proposition 6. *The optimal solution $\hat{\vec{p}}^*$ to the relaxation of Problem 2 that ignores inequality constraints is unique and assigns*

$$\hat{p}_w^* = \frac{R\mu_w}{\Delta_w \sum_{w' \in W^o} \mu_{w'}} \textit{for all } w \in W^o \tag{10}$$

Proof. See the Supplement. The proof applies the method of Lagrange multipliers to this relaxation. ■

Equation 10 indicates that the relaxation's solution never has $p_w \leq 0$, but it may indeed violate the constraints $p_w \leq 1$. The main difficulty of inequality-constrained optimization is exactly in determining the subset of inequality constraints that are *active* under the optimal solution, i.e., in our case, finding all sources w for which the optimal $\vec{p}^*$ has $p_w^* = 1$.

Our algorithm LAMBDACRAWL-COMPLOBS (Algorithm 2) finds them iteratively. In each iteration (lines 4-14), we consider only the sources that haven't yet been proved to activate their $p_w \leq 1$ constraint under $\vec{p}^*$ (lines 3,12). For them, we solve a relaxation of Problem 2 that *ignores* these

constraints, using Proposition 6 (lines 5-6), and check if this solution $\vec{p}^*$ happens to violate any constraints (line 9). Our key insight is that any source w that activates or violates its constraint under Proposition 6's solution, i.e., has $\hat{p}^*_w \geq 1$, must necessarily have $p^*_w = 1$. As we prove in the Supplement, this follows from the strict concavity of $\overline{J}^\pi$ and the convexity of the optimization region given by the constraints. Whenever we can prove source w to activate its constraint in this way, we set $p^*_w = 1$, adjust the overall bandwidth constraint for the remaining sources to $R_{rem} = R - p^*_w \Delta_w = R - \Delta_w$, and remove w from further consideration (lines 10-12). Eventually, our problem gets reduced to a (possibly empty) set of sources for which Proposition 6's solution doesn't violate any constraints under the remaining bandwidth (lines 15-16). Since Proposition 6's solution is optimal in this case, the overall algorithm is optimal as well.

Algorithm 2: LAMBDACRAWL-COMPLOBS: finding the optimal crawl scheduling policy $\pi^* \in \Pi^o$ under complete change observations (Problem 2)

1 LAMBDACRAWL-COMPLOBS:
Input: $\vec{\mu}, \vec{\Delta}$ – importance and change rate vectors
2 R s.t. $0 \leq R \leq \sum_{w \in W} \Delta_w$ – bandwidth;
Output: $\vec{p}^*$ – vector specifying optimal per-page crawl probabilities upon receiving a change observation.
3 $W_{rem} \leftarrow W^o$ // remaining sources to consider
4 **while** $W^o_{rem} \neq \emptyset$ **do**
5 **foreach** $w \in W^o_{rem}$ **do**
6 $\hat{p}^*_w \leftarrow \frac{R\mu_w}{\Delta_w \sum_{w' \in W^o_{rem}} \mu_{w'}}$ for all $w \in W^o_{rem}$
7 $ViolationDetected \leftarrow False$
8 **foreach** $w \in W^o_{rem}$ **do**
9 **if** $\hat{p}^*_w \geq 1$ **then**
10 $p^*_w \leftarrow 1$
11 $R \leftarrow R - \Delta_w$ // reduce remaining bandwidth
12 $W^o_{rem} \leftarrow W^o_{rem} \setminus \{w\}$ // ignore w onwards
13 $ViolationDetected = True$
14 **if** $ViolationDetected == False$ **then** break
15 **foreach** $w \in W^o_{rem}$ **do**
16 $p^*_w \leftarrow \hat{p}^*_w$
17 Return $\vec{p}^* = (p^*_w)_{w \in W^o}$

Proposition 7. LAMBDACRAWL-COMPLOBS *returns the optimal solution to Problem 2 and runs in time* $O(|W^o|^2)$.

Proof. See the Supplement. The proof formalizes the above intuitions and critically relies on the concavity of $\overline{J}^\pi$. ■

In practice, the running time bound of $O(|W^o|^2)$ is very loose. Each iteration usually discovers several active constraints, and many sources don't activate their constraint under $\vec{p}^*$, so the computation time is close to linear in $|W^o|$.

4.3. Crawl scheduling under mixed observability

In practice, trackers have to simultaneously handle sources W^- for which complete change observations are available and sources W^o for which they aren't at the same time, under a common bandwidth constraint R. How should we optimize scheduling mixed-observability scenarios?

Consider the policy class that combines Π^- and Π^o:

$$\Pi^{\ominus} = \begin{cases} \{CrSeq_w \sim Poisson(\rho_w) \text{ for all } w \in W^- | \vec{\rho}\}, \\ \{\text{for all } w \in W^o, \text{ whenever change observation} \\ o_w \text{ arrives, crawl } w \text{ with probability } p_w \ | \vec{p}\} \end{cases} \tag{11}$$

For $\pi \in \Pi^{\ominus}$, Propositions 2 and 5 imply that J^π from Equation 1 is equivalent to

$$J^\pi = -\sum_{w \in W^-} \mu_w \ln\left(\frac{\rho_w}{\Delta_w + \rho_w}\right) - \sum_{w \in W^o} \mu_w \ln(p_w) \tag{12}$$

Optimization over $\pi \in \Pi^{\ominus}$ can be stated as follows:

Problem 3. *[Finding* $\pi^* \in \Pi^{\ominus}$*]*

INPUT: *bandwidth* $R > 0$*; importance and change rate vectors* $\vec{\mu}, \vec{\Delta} > 0$.

OUTPUT: *Crawl rates* $\vec{\rho} = (\rho_w)_{w \in W^-}$ *and crawl probabilities* $\vec{p} = (p_w)_{w \in W^o}$ *maximizing*

$$\overline{J}^\pi = -J^\pi = \sum_{w \in W^-} \mu_w \ln\left(\frac{\rho_w}{\Delta_w + \rho_w}\right) + \sum_{w \in W^o} \mu_w \ln(p_w) \tag{13}$$

subject to

$$\sum_{w \in W^-} \rho_w + \sum_{w \in W^o} p_w \Delta_w = R,$$

$$\rho_w > 0 \text{ for all } w \in W^-, \ \ 0 < p_w \leq 1 \text{ for all } w \in W^o$$

The optimization objective (Equation 13) is strictly concave as a sum of concave functions over the region described by the constraints, and therefore has a unique maximizer. To find it efficiently, we observe that solving Problem 3 amounts to deciding how to split the global bandwidth constraint R into an allotment R^o for sources with complete change observations and $R^- = R - R^o$ for the rest. For any candidate split, LAMBDACRAWL-COMPLOBS and LAMBDACRAWL-INCOMLOBS give us the reward-maximizing policy parameters $\vec{p}^*(R^o)$ and $\vec{\rho}^*(R^-)$, respectively, and Equation 13 then tells us the overall value $\overline{J}^*(R^o, R^-)$ of that split. We also know that for the optimal split, $R^{o^*} \in [0, \min\{R, \sum_{w \in W_o} \Delta_w\}]$, as described in Problem 2 and discussion immediately before it. Thus, we can find Problem 3's maximizer to any desired precision using a method such as Golden-section search (Kiefer, 1953) on R^o. LAMBDACRAWL (Algorithm 3) implements this idea, where SPLIT-EVAL-$\overline{J}^*$ (line 7) evaluates $\overline{J}^*(R^o, R^-)$ and OptMaxSearch denotes an optimal search method.

Proposition 8. LAMBDACRAWL *(Algorithm 3) finds an*

arbtirarily close approximation to Problem 3's optimal solution using $O(\log(\frac{R}{\epsilon}))$ *calls to* LAMBDACRAWL-INCOMLOBS *and* LAMBDACRAWL-COMPLOBS.

Proof. This follows directly from the optimality of LAMBDACRAWL-INCOMLOBS and LAMBDACRAWL-COMPLOBS (Propositions 3 and 7), as well as of OptMaxSearch such as Golden section, which makes $O(\log(\frac{R}{\epsilon}))$ iterations. ■

Algorithm 3: LAMBDACRAWL: finding optimal mixed-observability policy $\pi^* \in \Pi^{\ominus}$ (Problem 3)

Input: $R > 0$ – bandwidth;
$\vec{\mu} > 0, \vec{\Delta} > 0$ – importance and change rates;
$\epsilon^{\text{no-obs}}, \epsilon > 0$ – desired precisions
Output: $\vec{\rho}^*, \vec{p}^*$ – crawl rates and probabilities for sources without and with complete change observations.

1 $R^o_{min} \leftarrow 0$
2 $R^o_{max} \leftarrow \min\{R, \sum_{w \in W_o} \Delta_w\}$
3 $\vec{\rho}^*, \vec{p}^* \leftarrow$ *OptMaxSearch*(Split-Eval-$\overline{J}^*, R^o_{min}, R^o_{max}, \epsilon$)
4 // E.g., Golden section search (Kiefer, 1953)
5 Return $\vec{\rho}^*, \vec{p}^*$
6
7 SPLIT-EVAL-$\overline{J}^*$:
Input: R^o – bandwidth for sources with complete change observations, $R, \vec{\mu}, \vec{\Delta}, \epsilon^{\text{no-obs}}$
Output: $\overline{J}^*$ (Equation 13) for the given split
8 $\vec{\rho} \leftarrow$ LAMBDACRAWL-INCOMLOBS$(R - R^o, \vec{\mu}_{W^-}, \vec{\Delta}_{W^-}, \epsilon^{\text{no-obs}})$
9 $\vec{p} \leftarrow$ LAMBDACRAWL-COMPLOBS$(R^o, \vec{\mu}_{W^o}, \vec{\Delta}_{W^o})$
10 Return $\sum_{w \in W^-} \mu_w \ln\left(\frac{\rho_w}{\Delta_w + \rho_w}\right) + \sum_{w \in W^o} \mu_w \ln(p_w)$

5. Reinforcement learning for scheduling

Although staleness minimization is efficient under the known-model assumption, in reality change rates are usually unavailable and vary with time, requiring constant relearning. As it turns out, their estimation can be done with only minor changes to LAMBDACRAWL that turn it into a type of model-based reinforcement learning (RL) algorithm.

Suppose for a given source w the tracker has observed a sequence of binary change indicator variables $z_1, z_2, \ldots, z_U$, where $t_0, \ldots, t_U$ are times when the observations arrived, and, for $1 \le j \le U$, $z_j = 1$ iff the source changed compared to time t_{j-1} *at least* once. Consider two cases:

Incomplete change observations for w**.** Crawling the source is the only way for the tracker to find out about any changes: after each crawl at times $t_1, \ldots, t_U$, the tracker checks for changes compared to w's previous crawl and records $z_{t_j} = 1$ or $z_{t_j} = 0$ accordingly. There may be more than one change since the previous crawl, but the tracker cannot determine this. Denoting $a_{t_j} = t_j - t_{j-1}, j \ge 1$, Cho & Garcia-Molina (2003b) show that $\hat{\Delta}$ that solves

$$\sum_{j:z_{t_j}=1} \frac{a_{t_j}}{e^{a_{t_j}\Delta} - 1} - \sum_{j:z_{t_j}=0} a_{t_j} = 0, \tag{14}$$

is a maximum-likelihood estimator of Δ for the given source. The l.h.s. of the equation is monotonically decreasing in Δ, so $\hat{\Delta}$ can be efficiently found numerically. This estimator is consistent under mild conditions (Cho & Garcia-Molina, 2003b), e.g., if the sequence $\{a_{t_j}\}_{j=1}^{\infty}$ doesn't converge to 0, i.e., if the observations are spaced apart.

Complete change observations for w**.** In this case, for all j, $z_{t_j} = 1$; a change observation arrives only when there is a change and indicates *exactly one* change. This is a standard setting for estimating Poisson process intensity, and a consistent MLE estimator $\hat{\Delta}$ is simply the time-averaged number of observed changes (Taylor & Karlin, 1998):

$$\Delta = \frac{U+1}{t_U}, \tag{15}$$

LAMBDALEARNANDCRAWL, a model-based RL version of LAMBDACRAWL that uses these estimators to learn model parameters simultaneously with scheduling is presented in Algorithm 4. It operates in epochs of length T_{epoch} time units each (lines 3-13). At the start of each epoch n, it calls LAMBDACRAWL (Algorithm 3) on the available $\vec{\hat{\Delta}}_{n-1}$ change rate estimates to produce a policy $(\vec{\rho}^*_n, \vec{p}^*_n)$ optimal with respect to them (line 4). Executing this policy during the current epoch, for the time period of T_{epoch}, and recording the observations extends the observation history (lines 7-8). (Note though that for sources $w \in W^o$, the observations don't depend on the policy.) It then re-estimates change rates using a suffix of the augmented observation history (lines 10-13). Under mild assumptions, LAMBDALEARNANDCRAWL converges to the optimal policy:

Proposition 9. LAMBDALEARNANDCRAWL *(Algorithm 4) converges in probability to the optimal policy under the true change rates* $\vec{\Delta}$*, i.e.,* $\lim_{N_{epochs}\to\infty}(\vec{\rho}_{N_{epochs}}, \vec{p}_{N_{epochs}}) = (\vec{\rho}^*, \vec{p}^*)$*, if* $\vec{\Delta}$ *is stationary and* $S(n)$*, the length of the history's training suffix, satisfies* $S(N_{epoch}) = length(obs\_hist)$.

Proof. See the Supplement. It follows from the consistency and positivity of the change rate estimates, as well as LAMBDACRAWL's optimality ■

Using LAMBDALEARNANDCRAWL in practice requires attention to several aspects:

Stationarity of $\vec{\Delta}$. Source change rates may vary with time, so the length of history suffix for estimating $\vec{\Delta}$ should typically be shorter than the entire available history.

Singularities of $\vec{\hat{\Delta}}$ *estimators*. Given a limited observation history, the MLE in Equation 14 can produce $\hat{\Delta} = \infty$ if all crawls detected a change (the r.h.s. is 0). Similarly, Equation 15 can produce $\hat{\Delta}_w = 0$ if no observations about

Algorithm 4: LAMBDALEARNANDCRAWL: finding optimal crawl scheduling policy $\pi^* \in \Pi^{\ominus}$ (Problem 3) under initially unknown change model

Input: $R > 0$ – bandwidth;
$\vec{\mu} > 0, \vec{\hat{\Delta}}_0 > 0$ – importance and initial change rate guesses
$\epsilon^{\text{no-obs}}, \epsilon > 0$ – desired precisions
$T_{epoch} > 0$ – duration of an epoch
$N_{epochs} > 0$ – number of epochs
$S(n)$ – for each epoch n, observation history suffix length for learning $\vec{\Delta}$ in that epoch

1 // $obs\_hist[S(n)]$ is $S(n)$-length observation history suffix
2 $obs\_hist \leftarrow ()$
3 **foreach** $1 \leq n \leq N_{epochs}$ **do**
4 $\quad \vec{\rho}^*_n, \vec{p}^*_n \leftarrow \text{LAMBDACRAWL}(R, \vec{\mu}, \vec{\hat{\Delta}}_{n-1}, \epsilon^{\text{no-obs}}, \epsilon)$
5 $\quad$ // $\vec{Z}_{new}$ holds observations for all sources from start to
6 $\quad$ // end of epoch n. Execute policy $(\vec{\rho}^*_n, \vec{p}^*_n)$ to get it
7 $\quad \vec{Z}_{new} \leftarrow ExecuteAndObserve(\vec{\rho}^*_n, \vec{p}^*_n, T_{epoch})$
8 $\quad Append(obs\_hist, \vec{Z}_{new})$
9 $\quad$ // Learn new $\vec{\Delta}$ estimates using Eqs. 14 and 15
10 $\quad$ **foreach** $w \in W^-$ **do**
11 $\quad\quad \hat{\Delta}_{n_w} \leftarrow Solve(\sum_{j:z_{j_w}=1} \frac{a_j}{e^{a_j \hat{\Delta}} - 1} + \frac{0.5}{e^{0.5\hat{\Delta}} - 1} - \sum_{j:z_{j_w}=0} a_j - 0.5 = 0, obs\_hist[S(n)])$
12 $\quad$ **foreach** $w \in W^o$ **do**
13 $\quad\quad \hat{\Delta}_{n_w} \leftarrow Solve(\frac{U_{S(n)}+0.5}{S(n)+0.5}, obs\_hist[S(n)])$

w arrived during a given period. To avoid them without affecting estimation consistency, we use smoothing to add imaginary observation intervals of length 0.5 to Equation 14 and imaginary 0.5 observation to Equation 15 (lines 11,13).

The sheer number of parameters LAMBDALEARNANDCRAWL needs to learn raises a question: can we employ an approximation with fewer parameters to learn? The following result suggests one such approximation:

Proposition 10. *Suppose the tracker's set of sources W^- is such that for some constant $c > 0$, $\frac{\mu_w}{\Delta_w} = c$ for all $w \in W^-$. Then minimizing harmonic penalty under incomplete change observations (Problem 1) has $\rho^*_w = \frac{\mu_w R^-}{\sum_{w' \in W^-} \mu_{w'}}$.*

Proof. See the Supplement. The proof proceeds by plugging in $\Delta_w = \frac{1}{c}\mu_w$ into Equation system 7. ■

In essence, the proposition states that if the importance-to-change-rate ratio is constant across all sources, then each source's crawl rate is independent of its change rate, and even the ratio constant itself. This greatly simplifies crawl scheduling, because for sources $w \in W^-$, we don't need to learn any change rates, provided that the constant-ratio assumption holds at least approximately. Moreover, if LAMBDACRAWL-INCOMLOBS (Algorithm 1) is replaced by assigning $\rho^*_w, w \in W^-$ as in Proposition 10, an $O(|W^-|)$ operation, which improves the computational efficiency of LAMBDACRAWL and LAMBDALEARNANDCRAWL in its own right. Our evaluation (Section 7) explores the quality of this approximation empirically.

6. Related work

Most literature related to our work comes from research on polling-based cache/database maintenance and web crawling. The term "Web caching," in spite of its seeming relevance, refers to *HTTP caches*, a different cache type found in browsers, proxy servers, and content distribution networks (CDNs). They refresh their content when prompted by a client request or when a source pushes fresh content (as in the case of CDNs). In contrast, this paper focuses on settings where neither of these is feasible. A major example of such trackers are search engines and knowledge graphs. Olston & Najork (2010) provides an extensive survey of crawling research; we review its most relevant aspects below.

Maintaining fresh cache via polling. This problem has been studied under a variety of settings and assumptions. Besides monitoring online information sources, mathematically similar settings have been researched in the context of scheduling maintenance service to machines (Anily et al., 1998; Bar-Noy et al., 1998). In the context of a search engine index, the closest works are Cho & Garcia-Molina (2003a), Wolf et al. (2002), Pandey & Olston (2005), and Azar et al. (2018). Like Cho & Garcia-Molina (2003a) and Azar et al. (2018), we use the method of Lagrange multipliers as part of optimization , and adopt the Poisson change model of Cho & Garcia-Molina (2003a) and many works since. Our contributions differ from prior art in several ways: (1) optimization objectives and properties of resulting policies (see the next subsection); (2) crawl scheduling for sources with complete change observations, both separately from and jointly with the commonly studied setting of sources whose changes are detectable only at crawl time; (3) learning model parameters in a principled way during crawling.

Optimization objectives. Staleness penalty minimization we proposed can be viewed as a generalization of the binary freshness objective studied by Cho & Garcia-Molina (2003a) and Azar et al. (2018) whereby the cache incurs a fixed penalty if and only if the source has changed in any way at least once since it was last crawled. As Azar et al. (2018) show, the optimal policy for this objective has the practically undesirable property of starving many sources, leaving the tracker's copies to diverge from the original. In contrast, our objective guarantees that this doesn't happen. Other related technical objectives include minimizing age (Cho & Garcia-Molina, 2000), a.k.a. *obsolescence* (Gal & Eckstein, 2001). On the other hand, Pandey & Olston (2005) focus on a cache "quality" measure that depends on user behavior, and Wolf et al. (2002) study another user-centric objective, embarrassment minimization, whereby pages likely to be most clicked get crawled more frequently.

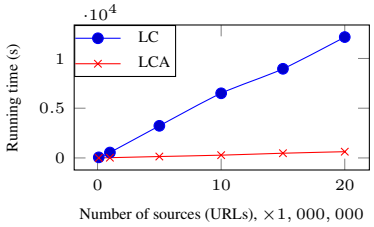

Figure 1. Solution time of LAMBDACRAWL and LAMBDACRAWLAPPROX as a function of source set size $|W|$. Both scale *linearly* in $|W|$ in practice, but LAMBDACRAWLAPPROX shows significant time savings. See Figures 2 and 3 for policy cost comparisons.

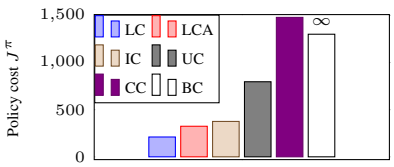

Figure 2. Average policy cost per page of LAMBDACRAWL (*LC*) and other algorithms on the 20M-URL dataset. **Lower bars indicate better policies.** The optimal LAMBDACRAWL incurs 31% lower cost than the next-best prior algorithm *ImportanceCrawl* (*IC*). BinaryCrawl (Azar et al., 2018) has $J^\pi = \infty$ due to crawl-starving many URLs.

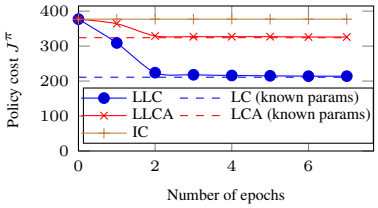

Figure 3. Convergence of LAMBDALEARNANDCRAWL (*LLC*) and LAMBDALEARNANDCRAWLAPPROX (*LLCA*) for a subsample of the dataset with $|W| = 100000$. 1 epoch = 14 days. Dashed lines indicate their asymptotic policy cost (under known change rates). *ImportanceCrawl* (IC) is shown for comparison as the best-of-the-rest approach.

User-centric measures are desirable in principle, but involve many hard-to-verify assumptions about the user population and become ambiguous for trackers, such as caches, supporting several distinct services, e.g., a knowledge graph, image search, and document search at the same time. Coupled with computational tractability, this makes technical objectives like freshness preferable in many practical systems.

Acquiring model parameters. Like ours, many previous works rely on page importance and change models. Importance can be defined and quickly determined from information readily available to search engines, e.g., page relevance to queries (Wolf et al., 2002), query-independent popularity such as PageRank (Page et al., 1998), and other factors (Pandey & Olston, 2005)). Learning page change rates is more delicate. Change rate estimators we use are due to (Cho & Garcia-Molina, 2003b); our contribution in this regard is integrating model estimation with crawl scheduling while providing theoretical guarantees, as well as identifying conditions when estimation can be side-stepped using an approximation. While many prior works adopt the homogeneous Poisson change process, its non-homogeneous variant (Gal & Eckstein, 2001) as well as a quasi-deterministic change model (Wolf et al., 2002) have also been considered. As an alternative to estimating source change models individually, Cho & Ntoulas (2002) propose inferring them indirectly using page similarities. Predictions of page change kind and significance, as in (Radinsky & Bennett, 2013), are also helpful for a crawl scheduler in prioritizing crawls.

7. Empirical evaluation

In the experiments, we explore LAMBDACRAWL's (labelled *LC* in the figures) scalability and solution quality relative to LAMBDACRAWLAPPROX (*LCA*), a LAMBDACRAWL version that uses the approximation from Proposition 10 instead of LAMBDACRAWL-INCOMLOBS for $w \in W^-$, *ImportanceCrawl* (*IC*) and *ChangeRateCrawl* (*CC*) (Cho & Garcia-Molina, 2003a), *BinaryCrawl* (*BC*) (Azar et al., 2018), and *UniformCrawl* (*UC*) — please see the Supplement, Section 10 for the description of these algorithms from prior literature. We also assess the convergence speed of LAMBDALEARNANDCRAWL.

For the dataset, we crawled $20,151,962$ URLs daily over 14 weeks to estimate their crawl rates reliably using Equations 14 and 15. These URLs are data sources for the knowledge graph of a major virtual assistant. Approximately 13% of them had complete change observations from reliable sitemaps. We set their importance scores μ_w to values defined by the production crawler based on PageRank and popularity. The bandwidth constraint was set to 20% of the number of pages in the corresponding experiment.

The experiments used a Python implementation of all algorithms from this paper, All algorithms were implemented in Python (*the code will be open-sourced*) and run on a Windows 10 laptop with 16GB RAM and an Intel quad-core 2.11GHz i7-8650U CPU.

The results are shown in Figures 1, 2, and 3, with additional details in Supplement, Section 10. Note that, as described there and per Proposition 10, *ImportanceCrawl* is a special case of LAMBDACRAWL and LAMBDACRAWLAPPROX without the LAMBDACRAWL-COMPLOBS component and assuming uniform $\frac{\mu_w}{\Delta_w}$. Thus, as the $\frac{\mu_w}{\Delta_w}$ distribution across most of the dataset happens to be nearly uniform (Figure 4 in Supplement, Section 10), our theory explains why *ImportanceCrawl* does relatively well on this data.

8. Conclusion

We have introduced a new objective and a suite of highly efficient algorithms for it to address the freshness crawl scheduling problem faced by many services from search engines to databases. In particular, we have presented LAMBDALEARNANDCRAWL, which integrates model parameter learning with scheduling optimization. To give convergence speed guarantees for this kind of approaches in the future, we intend to look at multi-armed bandit (MAB) models similar to Immorlica & Kleinberg (2018). Their current fundamental distinction from our setting is the reward mechanism: a MAB yields a reward only when an arm is pulled, not continuously as in freshness optimization. Nonetheless, we believe that a new bandit model will provide practically and theoretically important insights into freshness crawl scheduling.

References

Anily, S., Glass, C., and Hassin, R. The scheduling of maintenance service. *Discrete Applied Mathematics*, pp. 27–42, 1998.

Azar, Y., Horvitz, E., Lubetzky, E., Peres, Y., and Shahaf, D. Tractable near-optimal policies for crawling. *Proceedings of the National Academy of Sciences (PNAS)*, 2018.

Baeza-Yates, R. A. and Ribeiro-Neto, B. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

Bar-Noy, A., Bhatia, R., Naor, J., and Schieber, B. Minimizing service and operation costs of periodic scheduling. In *SODA*, pp. 11–20, 1998.

Broder, A., Glassman, S., Manasse, M., and Zweig, G. Syntactic clustering of the web. In *WWW*, pp. 1157–1166, 1997.

Burden, R. L. and Faires, J. D. *Numerical Analysis*. PWS Publishers, 3rd edition, 1985.

Charikar, M. Similarity estimation techniques from rounding algorithms. In *STOC*, pp. 380–388, 2002.

Cho, J. and Garcia-Molina, H. Synchronizing a database to improve freshness. In *ACM SIGMOD International Conference on Management of Data*, 2000.

Cho, J. and Garcia-Molina, H. Effective page refresh policies for web crawlers. *ACM Transactions on Database Systems*, 28(4):390–426, 2003a.

Cho, J. and Garcia-Molina, H. Estimating frequency of change. *ACM Transactions on Internet Technology*, 3(3): 256–290, 2003b.

Cho, J. and Ntoulas, A. Effective change detection using sampling. In *VLDB*, 2002.

Coffman, E. G., Liu, Z., and Weber, R. R. Optimal robot scheduling for web search engines. *Journal of Scheduling*, 1(1), 1998.

Gal, A. and Eckstein, J. Managing periodically updated data in relational databases. *Journal of ACM*, 48:1141–1183, 2001.

Hixon, B., Clark, P., and Hajishirzi, H. Learning knowledge graphs for question answering through conversational dialog. 2015.

Immorlica, N. and Kleinberg, R. Recharging bandits. In *FOCS*, 2018.

Kiefer, J. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4(3): 502–506, 1953.

Kolobov, A., Lubetzky, E., Peres, Y., and Horvitz, E. Optimal freshness crawl under politeness constraints. In *SIGIR*, 2019.

Olston, C. and Najork, M. Web crawling. *Foundations and Trends in Information Retrieval*, 3(1):175–246, 2010.

Page, L., Brin, S., Motwani, R., and Winograd, T. The pagerank citation ranking: Bringing order to the web. Technical report, MA, USA, 1998.

Pandey, S. and Olston, C. User-centric web crawling. In *WWW*, 2005.

Pandey, S., Ramamritham, K., and Chakrabarti, S. Monitoring the dynamic web to respond to continuous queries. 2003.

Pandey, S., Dhamdhere, K., and Olston, C. WIC: A general-purpose algorithm for monitoring web information sources. 2004.

Radinsky, K. and Bennett, P. Predicting content change on the web. In *WSDM*, 2013.

Sutton, R. and Barto, A. G. *Introduction to Reinforcement Learning*. MIT Press, 1st edition, 1998.

Taylor, H. and Karlin, S. *An Introduction To Stochastic Modeling*. Academic Press, 3rd edition, 1998.

van den Bosch, A., Bogers, T., and de Kunder, M. A longitudinal analysis of search engine index size. 2015.

Wolf, J. L., Squillante, M. S., Yu, P. S., Sethuraman, J., and Ozsen, L. Optimal crawling strategies for web search engines. In *WWW*, 2002.

SUPPLEMENT

9. Proofs

PROOF OF PROPOSITION 1. This is a direct consequence of Proposition 2, which implies that any policy π with $\rho_w = 0$ for any source w with $\mu_w, \Delta_w > 0$ has $J^\pi = \infty$, whereas any π with $rho_w > 0$ for every $\mu_w, \Delta_w > 0$ has $J^\pi = \infty$ ■.

PROOF OF PROPOSITION 2.

First we rearrange Equation 1 as follows, dropping the distributions under the expectation and using W instead of W^- throughout the proof to make the notation less cumbersome:

$$\begin{aligned} J^\pi &= \lim_{T\to\infty} \mathop{\mathbb{E}}_{\substack{CrSeq\sim\pi,\\ ChSeq\sim P(\vec{\Delta})}} \left[\frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w C(N_w(t))\right) dt\right] \\ &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \mathbb{E}\left[C(N_w(t))\right]\right) dt \end{aligned}$$

Then we use the definition of expectation, chain rule of probabilities, and variable t_{prev} to denote the time when source w was last crawled before time t (although t_{prev} is specific to each source w, for clarity of notation we make this implicit):

$$\begin{aligned} J^\pi &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \mathbb{E}[C(N_w(t))]\right) dt \\ &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{m=0}^{\infty} \Big(C(m)\cdot \mathbb{P}[N_w(t)=m]\Big)\right) dt \\ &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{m=0}^{\infty} \left(C(m)\int_0^t \mathbb{P}[N_w(t)=m \mid t-t_{prev}=T']\,\mathbb{P}[t-t_{prev}=T']dT'\right)\right) dt \end{aligned}$$

Now using the fact that our policy is a set of Poisson processes with parameters ρ_w and page changes are governed by another set of Poisson processes with parameters Δ_w, we can plug in appropriate expressions for the probabilities:

$$\begin{aligned} J^\pi &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{m=0}^{\infty} \left(C(m)\int_0^t \left(\frac{(T'\Delta_w)^m e^{-T'\Delta_w}}{m!}\right)\left(\rho e^{-\rho_w T'}\right)dT'\right)\right) dt \\ &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{m=0}^{\infty} \left(\frac{C(m)\rho_w\Delta_w^m}{m!}\int_0^t T'^m e^{-(\rho_w+\Delta_w)T'}dT'\right)\right) dt \end{aligned}$$

We do a variable substitution $u = (\rho_w + \Delta_w)T'$, so $T' = \frac{u}{\Delta_w+\rho_w}$ and $dT' = \frac{du}{\Delta_w+\rho_w}$:

$$J^\pi = \lim_{T\to\infty}\frac{1}{T}\int_0^T\left(\sum_{w\in W}\mu_w\sum_{m=0}^{\infty}\left(\frac{C(m)\rho_w\Delta_w^m}{m!}\int_0^t\left(\frac{u}{\Delta_w+\rho_w}\right)^m e^{-u}\frac{du}{\Delta_w+\rho_w}\right)\right)dt$$
$$= \lim_{T\to\infty}\frac{1}{T}\int_0^T\left(\sum_{w\in W}\mu_w\sum_{m=0}^{\infty}\left(\frac{C(m)}{m!}\left(\frac{\rho_w}{\Delta_w+\rho_w}\right)\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\int_0^t u^m e^{-u}du\right)\right)dt$$

Consider $F(m,t)=\int_0^t u^m e^{-u}du$. By definition of gamma functions, $F(m,t)=\Gamma(m+1)-\Gamma(m+1,t)=m!-\Gamma(m+1,t)$. Recalling that $C(m)=H(m)$ for $m>0$ and $C(0)=0$ (Equation 3), we get

$$J^\pi = \lim_{T\to\infty}\frac{1}{T}\int_0^T\left(\sum_{w\in W}\mu_w\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\rho_w}{\Delta_w+\rho_w}\right)\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m(m!-\Gamma(m+1,t))\right)\right)dt$$
$$= \lim_{T\to\infty}\frac{1}{T}\left(\sum_{w\in W}\mu_w\left(\frac{\rho_w}{\Delta_w+\rho_w}\right)\int_0^T\left[\sum_{m=1}^{\infty}\left(H(m)\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\right)-\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\Gamma(m+1,t)\right)\right]dt\right)$$

Now consider for each $w\in W$ functions $G(T)=\int_0^T\sum_{m=1}^{\infty}\left(H(m)\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\right)dt$, $R(t)=\int_0^T\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\Gamma(m+1,t)\right)dt$, and $\lim_{T\to\infty}\frac{1}{T}(G(T)-R(T))$ so that

$$J^\pi=\sum_{w\in W}\left(\mu_w\left(\frac{\rho_w}{\Delta_w+\rho_w}\right)\left(\lim_{T\to\infty}\frac{1}{T}(G(T)-R(T))\right)\right) \tag{16}$$

Thus, if $\lim_{T\to\infty}\frac{1}{T}(G(T)-R(T))$ exists, is finite, and we can compute it, we can compute J^π as well. Focusing on $G(T)$ and recalling that $\sum_{m=1}^{\infty}H(m)x^m=-\frac{\ln(1-x)}{1-x}$ for $|x|<1$, we see that $G(T)=-\frac{\ln(\frac{\rho_w}{\Delta_w+\rho_w})}{\frac{\rho_w}{\Delta_w+\rho_w}}T$, so $\lim_{T\to\infty}\frac{G(T)}{T}=-\frac{\ln(\frac{\rho_w}{\Delta_w+\rho_w})}{\frac{\rho_w}{\Delta_w+\rho_w}}$ exists. Focusing on $R(T)$, we see that since $m!<\Gamma(m+1,t)$ for any $t>0$ and since $R(T)$ is an integral of a non-negative function, we have $0\le R(T)\le G(T)<\infty$, so $\lim_{T\to\infty}\frac{R(T)}{T}$ exists as well. Therefore, we can write $\lim_{T\to\infty}\frac{1}{T}(G(T)-R(T))=\lim_{T\to\infty}\frac{G(T)}{T}-\lim_{T\to\infty}\frac{R(T)}{T}$.

We already know $\lim_{T\to\infty}\frac{G(T)}{T}$. To evaluate $\lim_{T\to\infty}\frac{R(T)}{T}$, we upper-bound $R(T)$. Again using the fact that it is an integral of a non-negative function, we have

$$R(T)=\int_0^T\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\Gamma(m+1,t)\right)dt$$
$$<\int_0^{\infty}\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\Gamma(m+1,t)\right)dt$$
$$=\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\int_0^{\infty}\Gamma(m+1,t)dt\right)$$
$$=\sum_{m=1}^{\infty}\left(\frac{H(m)}{m!}\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\Gamma(m+2)\right)$$
$$=\sum_{m=1}^{\infty}\left((m+1)H(m)\left(\frac{\Delta_w}{\Delta_w+\rho_w}\right)^m\right)$$

We have used the fact that $\int_0^{\infty}\Gamma(m+1,t)dt=\Gamma(m+2)=(m+1)!$. The series $\sum_{m=1}^{\infty}(m+1)H(m)x^m$ converges for

$|x| < 1$ to some limit $L > 0$, so we have $\lim_{T\to\infty} \frac{R(T)}{T} \leq \lim_{T\to\infty} \frac{L}{T} = 0$ and $\lim_{T\to\infty} \frac{1}{T}(G(T) - R(T)) = -\frac{\ln(\frac{\rho_w}{\Delta_w+\rho_w})}{\frac{\rho_w}{\Delta_w+\rho_w}}$. Plugging this back into Equation 9, we get

$$J^\pi = -\sum_{w\in W} \mu_w \ln\left(\frac{\rho_w}{\Delta_w + \rho_w}\right)$$

■

PROOF OF PROPOSITION 3. The high-level idea is to apply the method of Lagrange multipliers to Problem 1's relaxation *without* inequality constraints $\vec{\rho} \geq 0$ and show that (a) only one local maximum of this relaxation is within the region given by $\vec{\rho} \geq 0$ – the one satisfying Equation system 7 – and (b) solutions that touch the boundary of this region, i.e., have $\rho_w = 0$ for any $w \in W^-$, are suboptimal. In fact, part (b) follows immediately from Proposition 1, so we only need to solve the relaxation and show part (a).

To apply the method of Lagrange multipliers to the relaxation, we set $f(\vec{\rho}) = \overline{J}^\pi = \sum_{w\in W^-} \mu_w \ln\left(\frac{\rho_w}{\Delta_w+\rho_w}\right)$ and $g(\vec{\rho}) = \sum_{w\in W^-} \rho_w - R$. We need to solve

$$\begin{cases} \nabla f(\vec{\rho}) = \lambda \nabla g(\vec{\rho}) \\ g(\vec{\rho}) = 0. \end{cases}$$

For any $w \in W^-$, we have $\frac{\partial g}{\partial \rho_w} = 1$ and $\frac{\partial f}{\partial \rho_w} = \mu_w \frac{\Delta_w+\rho_w}{\rho_w} \frac{\Delta_w}{(\Delta_w+\rho_w)^2} = \frac{\mu_w \Delta_w}{\Delta_w \rho_w + \rho_w^2}$, so the above system of equations turns into

$$\begin{cases} \frac{\mu_w \Delta_w}{\Delta_w \rho_w + \rho_w^2} = \lambda, & \text{for all } w \in W^- \\ \sum_{w\in W^-} \rho_w = R \end{cases}$$

and therefore

$$\begin{cases} \lambda \rho_w^2 + \lambda \Delta_w \rho_w - \mu_w \Delta_w = 0, & \text{for all } w \in W^- \\ \sum_{w\in W^-} \rho_w = R \end{cases}$$

Solving each quadratic equation separately, we get

$$\begin{cases} \rho_w = \frac{-\Delta_w \pm \sqrt{\Delta_w^2 + \frac{4\mu_w \Delta_w}{\lambda}}}{2}, & \text{for all } w \in W^- \\ \sum_{w\in W^-} \rho_w = R \end{cases}$$

This gives all potential solutions to the relaxation of Problem 1. Now consider the inequality constraints $\vec{\rho} \geq 0$ omitted so far. Observe that any real solution to the above system that has $\rho_w = \frac{-\Delta_w - \sqrt{\Delta_w^2 + \frac{4\mu_w \Delta_w}{\lambda}}}{2}$ for any $w \in W^-$ implies $\rho_w < 0$ for $\mu_w, \Delta_w > 0$, which violates these constraints. Therefore, any solution to Problem 1 itself must satisfy

$$\begin{cases} \rho_w = \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w \Delta_w}{\lambda}}}{2}, & \text{for all } w \in W^- \\ \sum_{w\in W^-} \rho_w = R \end{cases}$$

and have $\lambda \geq 0$ (otherwise $\vec{\rho} < 0$, again violating the inequality constraints). Although the first group of equations are non-linear, note that each $\rho_w(\lambda)$ is strictly monotone decreasing in λ for $\lambda \geq 0$, so $\sum_{w\in W^-} \rho_w$ is strictly monotone decreasing too implying that $\sum_{w\in W^-} \rho_w(\lambda) = R$ has a unique solution in λ, and therefore there is a unique $\vec{\rho}$ satisfying the above system of equations. Thus, this $\vec{\rho}$ is the solution to Problem 1 and therefore corresponds to $\pi^* \in \Pi^-$. ■

PROOF OF PROPOSITION 4

We start by combining Equation system 7,

$$\begin{cases} \rho_w = \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda}}}{2}, & \text{for all } w \in W^- \\ \sum_{w\in W^-} \rho_w = R, \end{cases} \tag{17}$$

into one equation:

$$\sum_{w\in W^-} \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda}}}{2} = R$$

Bisection search is guaranteed to converge to *some* solution λ of this equation as long as we initialize the search with lower_bound_λ, upper_bound_λ s.t. $\lambda \in [\text{lower\_bound\_}\lambda, \text{upper\_bound\_}\lambda]$. However, all $\lambda_- \leq 0$ that solve Equation system 7 correspond to solutions that have $\vec{\rho} < 0$. At the same time, from the proof of Proposition 3 we know that there is exactly one $\lambda_+ > 0$ that solves Equation system 7, and it corresponds to the (unique) the optimal solution $\vec{\rho}^*$ of Problem 1. We want bisection search to find only this $\lambda_+ > 0$, so we want lower_bound_λ, upper_bound_λ s.t. $\lambda_+ \in [\text{lower\_bound\_}\lambda, \text{upper\_bound\_}\lambda]$ *and* lower_bound_λ, upper_bound_$\lambda > 0$.

To find these bounds, we observe that the l.h.s. of the above equation is monotonically decreasing in λ for $\lambda > 0$, so if we find any $\lambda_l > 0$ that guarantees

$$\sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda_l}} \geq \Delta_w + \frac{2R}{|W|} \text{ for all } w \in W^- ,$$

then we have

$$\sum_{w\in W^-} \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda_l}}}{2} \geq R$$

and hence $\lambda_l \leq \lambda_+$. To find such λ_l, we perform a series of algebraic manipulations:

$$\begin{aligned}
&\sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda_l}} \geq \Delta_w + \frac{2R}{|W|} \text{ for all } w \in W^-, \lambda_l > 0 \\
\iff &\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda_l} \geq \left(\Delta_w + \frac{2R}{|W^-|}\right)^2 \text{ for all } w \in W^-, \lambda_l > 0 \\
\iff &\frac{4\mu_w\Delta_w}{\lambda_l} \geq \frac{4\Delta_w R}{|W^-|} + 4\left(\frac{R}{|W^-|}\right)^2 \text{ for all } w \in W^-, \lambda_l > 0 \\
\iff &\lambda_l \leq \frac{|W^-|^2\mu_w\Delta_w}{|W^-|\Delta_w R + R^2} \text{ for all } w \in W^-, \lambda_l > 0 \\
\impliedby &\lambda_l \leq \frac{|W^-|^2 \min_{w\in W^-}\{\mu_w\} \min_{w\in W^-}\{\Delta_w\}}{|W^-| \max_{w\in W^-}\{\Delta_w\} R + R^2}, \lambda_l > 0
\end{aligned}$$

Since the r.h.s. of this inequality is always positive, we set lower_bound_$\lambda = \frac{|W^-|^2 \min_{w\in W^-}\{\mu_w\} \min_{w\in W^-}\{\Delta_w\}}{|W^-| \max_{w\in W^-}\{\Delta_w\} R + R^2}$. An

analogous chain of reasoning shows that we can choose upper_bound_$\lambda = \frac{|W^-|^2 \max_{w\in W^-}\{\Delta_w\} \max_{w\in W^-}\{\mu_w\}}{|W^-| \min_{w\in W^-}\{\Delta_w\} R + R^2}$.

To establish a bound on the running time, we observe that each iteration of LAMBDACRAWL-INCOMLOBS involves evaluating $\sum_{w\in W^-} \rho_w$, which takes $O(|W^-|)$ time, so by the properties of bisection search the total running time is $O(\log_2(\frac{\text{upper\_bound\_}\lambda - \text{lower\_bound\_}\lambda}{\epsilon})|W^-|)$. ■

PROOF OF PROPOSITION 5 Let $Ch_w(t)$ denote the total number of changes that have happened at source w in time interval $[0, t]$. As in the proof of Proposition 2, we start with rearranging the cost function in Equation 1, dropping the distributions under the expectation and using W instead of W^o throughout the proof to make the notation less cumbersome. Using the definition of expectation and chain rule of probabilities to get:

$$
\begin{aligned}
J^\pi &= \lim_{T\to\infty} \mathop{\mathbb{E}}_{\substack{CrSeq\sim\pi,\\ ChSeq\sim P(\vec{\Delta})}} \left[\frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w C(N_w(t))\right) dt\right] \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \mathbb{E}\left[C(N_w(t))\right]\right) dt \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{m=0}^{\infty} \Big(C(m)\cdot \mathbb{P}[N_w(t)=m]\Big)\right) dt \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{c=0}^{\infty}\sum_{m=0}^{\infty} \Big(C(m)\cdot \mathbb{P}[N_w(t)=m|Ch_w(t)=c]\mathbb{P}[Ch_w(t)=c]\Big)\right) dt
\end{aligned}
$$

Since changes at every source w are governed by a Poisson process with rate Δ_w, $\mathbb{P}[Ch_w(t)=c] = \frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!}$. Now, consider $\mathbb{P}[N_w(t)=m|Ch_w(t)=c]$. Recall that whenever source $w\in W^o$ changes, we find out about the change immediately; with probability p_w the policy $\pi\in\Pi^o$ then crawls source w straight away, and with probability $(1-p_w)$ it waits to make this decision until we find out about w's next change. Therefore, the only way we can have $N_w(t)=m$ is if our policy crawled source w $m+1$ changes ago *and* has *not* chosen to crawl w after any of the m changes that happened since, or it hasn't chosen to crawl w since "the beginning of time". Thus, $N_w(t)$ is geometrically distributed, assuming that at least $m+1$ changes actually happened at source w in the time interval $[0,1]$. Thus, we have

$$
\mathbb{P}[N_w(t)=m|Ch_w(t)=c] = \begin{cases} p_w(1-p_w)^m, & \text{if } c\geq m+1 \\ (1-p_w)^m, & \text{if } c=m \\ 0 & \text{otherwise} \end{cases}
$$

Recalling that $C(m)=H(m)$ for $m>0$ and $C(0)=0$ (Equation 3) and putting everything together, we have

$$
\begin{aligned}
J^\pi &= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{c=0}^{\infty}\sum_{m=0}^{\infty} \Big(C(m)\cdot \mathbb{P}[N_w(t)=m|Ch_w(t)=c]\mathbb{P}[Ch_w(t)=c]\Big)\right) dt \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{c=1}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!}\right)\left(p_w \sum_{m=1}^{c-1} H(m)(1-p_w)^m + H(c)(1-p_w)^c\right)\right) dt \\
&= \lim_{T\to\infty} \frac{1}{T}\int_0^T \left(\sum_{w\in W} \mu_w \sum_{c=1}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!}\right)\left(p_w \sum_{m=1}^{\infty} H(m)(1-p_w)^m - p_w \sum_{m=c}^{\infty} H(m)(1-p_w)^m + H(c)(1-p_w)^c\right)\right) dt
\end{aligned}
$$

Now consider

$$G(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=0}^{\infty} C(m)(1-p_w)^m \right) \right) dt$$

$$R(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \right) \right) dt$$

$$F(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(C(c)(1-p_w)^c \right) \right) dt$$

Since

$$J^\pi = \lim_{T \to \infty} \frac{1}{T}(G(T) - R(T) + F(T)),$$

if we show that each of $\lim_{T\to\infty} \frac{G(T)}{T}$, $\lim_{T\to\infty} \frac{R(T)}{T}$, and $\lim_{T\to\infty} \frac{F(T)}{T}$ exists and manage to compute them, then we will know J^π. The rest of the proof focuses on computing these limits.

Consider $G(T)$. Note that $p_w \sum_{m=0}^{\infty} C(m)(1-p_w)^m = p_w \sum_{m=1}^{\infty} H(m)(1-p_w)^m$ doesn't depend on c. We can use the identity $\sum_{m=1}^{\infty} H(m)x^m = -\frac{\ln(1-x)}{1-x}$ for $|x| < 1$ to get $p_w \sum_{m=0}^{\infty} C(m)(1-p_w)^m = -\ln(p_w)$, so $G(T) = -\int_0^T \left(\sum_{w\in W} \mu_w \sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \ln(p_w) \right) dt$. To simplify $G(T)$ further, note that $\sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right)$ is just the probability of *any* number of changes occurring in time interval $[0, t]$ under a Poisson process, and therefore equals 1. Thus, $G(T) = -\int_0^T \left(\sum_{w\in W} \mu_w \ln(p_w) \right) dt = -T \sum_{w\in W} \mu_w \ln(p_w)$, so

$$\lim_{T\to\infty} \frac{G(T)}{T} = -\sum_{w\in W} \mu_w \ln(p_w).$$

Consider $R(T)$. Observe that $C(m) = H(m) < m$ for $m > 1$, so $p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m < p_w \sum_{m=c}^{\infty} m(1-p_w)^m = \frac{(1-p_w)^c(cp_w - p_w + 1)}{p_w}$. Because $(1-p_w)^c$ decreases in c much faster than $\frac{1}{c(cp_w-p_w+1)}$ for any fixed $0 < p_w \leq 1$, for any $w \in W$ there is a c_w^* s.t. $p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m < \frac{(1-p_w)^c(cp_w-p_w+1)}{p_w} < \frac{1}{c}$ for any $c > c_w^*$. Let $c^* = \max_{w\in W} c_w^*$. We can then upper-bound $R(T)$ as follows:

$$R(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \right) \right) dt = R_1(T) + R_2(T),$$

where

$$R_1(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{c^*-1} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \right) \right) dt$$

$$R_2(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=c^*}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \right) \right) dt$$

Considering $R_1(T)$, recalling that $C(m) = H(m)$ for $m > 0$, we have $p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \leq p_w \sum_{m=1}^{\infty} C(m)(1-p_w)^m = -\ln(p_w)$, as shown previously. Also, for any c we have $\int_0^T \frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} dt = \frac{\Gamma(c+1) - \Gamma(c+1, \Delta_w T)}{\Delta_w c!}$. Therefore,

$$R_1(T) = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{c^*-1} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \right) \right) dt$$
$$< - \sum_{w \in W} \mu_w \ln(p_w) \sum_{c=0}^{c^*-1} \frac{\Gamma(c+1) - \Gamma(c+1, \Delta_w T)}{\Delta_w c!}$$

Since $\lim_{T\to\infty} \frac{\Gamma(c+1,\Delta_w T)}{T} = 0$, $R_1(T)$ is therefore upper-bounded by a finite sum of terms that all go to 0 when divided by T as $T \to \infty$. Since $R_1(T) \geq 0$ also holds, we have $\lim_{T\to\infty} \frac{R_1(T)}{T} = 0$.

Considering $R_2(T)$, we use our definition of c^* to write

$$\begin{aligned}
R_2(T) &= \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=c^*}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(p_w \sum_{m=c}^{\infty} C(m)(1-p_w)^m \right) \right) dt \\
&< \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=c^*}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{c!} \right) \left(\frac{1}{c} \right) \right) dt = \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=c^*}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{(c+1)!} \right) \right) dt \\
&\leq \int_0^T \left(\sum_{w \in W} \mu_w \sum_{c=0}^{\infty} \left(\frac{e^{-\Delta_w t}(\Delta_w t)^c}{(c+1)!} \right) \right) dt \\
&= \int_0^T \left(\sum_{w \in W} \mu_w \left(\frac{1 - e^{-\Delta_w t}}{\Delta_w t} \right) \right) dt \\
&= \sum_{w \in W} \mu_w \frac{\ln(\Delta_w T) + \Gamma(0, \Delta_w T) + \gamma}{\Delta_w},
\end{aligned}$$

where γ is the Euler-Mascheroni constant. Since $\lim_{T\to\infty} \frac{\Gamma(0,\Delta_w T)}{T} = 0$, $R_2(T)$ is therefore upper-bounded by a finite sum of terms that all go to 0 when divided by T as $T \to \infty$. Since $R_2(T) \geq 0$ also holds, we have $\lim_{T\to\infty} \frac{R_2(T)}{T} = 0$. Thus, we have

$$\lim_{T\to\infty} \frac{R(T)}{T} = \lim_{T\to\infty} \frac{R_1(T) + R_2(T)}{T} = \lim_{T\to\infty} \frac{R_1(T)}{T} + \lim_{T\to\infty} \frac{R_2(T)}{T} = 0$$

Consider $F(T)$. Observe that for a suitably chosen constant $s > 0$, $sR(T) > F(T)$. Therefore, since $\lim_{T\to\infty} \frac{R(T)}{T} = 0$, $\lim_{T\to\infty} \frac{F(T)}{T} = 0$ too.

We have thus shown that

$$J^\pi = \lim_{T\to\infty} \frac{G(T) - R(t) + F(T)}{T} = \lim_{T\to\infty} \frac{G(T)}{T} - \lim_{T\to\infty} \frac{R(T)}{T} + \lim_{T\to\infty} \frac{F(T)}{T} = - \sum_{w \in W} \mu_w \ln(p_w)$$

■

PROOF OF PROPOSITION 6. Since under any $\vec{p} \geq 0$ crawl rates $\vec{\rho}$ are related to crawl probabilities via $\rho_w = p_w \Delta_w$, to apply the method of Lagrange multipliers to the relaxation of Problem 2 that takes into account only the bandwidth constraint we set $f(\vec{p}) = \overline{J}^\pi = \sum_{w \in W^o} \mu_w \ln(p_w)$ and $g(\vec{\rho}) = \sum_{w \in W^o} p_w \, \Delta_w - R$. We need to solve

$$\begin{cases} \nabla f(\vec{p}) = \lambda \nabla g(\vec{p}) \\ g(\vec{p}) = 0. \end{cases}$$

For any $w \in W^o$, we have $\frac{\partial g}{\partial p_w} = \Delta_w$ and $\frac{\partial f}{\partial p_w} = \frac{\mu_w}{p_w}$, so the above system of equations turns into

$$\begin{cases} \frac{\mu_w}{p_w} = \lambda \Delta_w, & \text{for all } w \in W^o \\ \sum_{w \in W^o} p_w \Delta_w = R \end{cases}$$

and therefore

$$\begin{cases} p_w = \frac{R\mu_w}{\Delta_w \sum_{w \in W^o} \mu_w} \text{for all } w \in W^o \\ \lambda = \frac{\sum_{w \in W^o} \mu_w}{R} \end{cases}$$

This is the only solution yielded by the method of Lagrange multipliers, so it is the unique maximizer of the relaxation. ■

PROOF OF PROPOSITION 7.

To establish LAMBDACRAWL-COMPLOBS's correctness, we first prove the following lemma, which establishes that any source w that violates its $p_w \leq 1$ constraint in any iteration of LAMBDACRAWL-COMPLOBS must have $p^*_w = 1$:

Lemma 1. *Let $\vec{p}^*$ be the maximizer of Problem 2, and let $\vec{p}'^*$ be the maximizer of the relaxation Problem 2 with the same inputs but with inequality constraints ignored. Then any source w that has $\vec{p}'^*$ has $p'^*_w > 1$, violating its inequality constraint, necessarily has $p^*_w = 1$.*

Proof. For convenience, we rewrite Problem 2 as an equivalent problem of maximizing $\overline{J}_{mod} = \sum_{w \in W^o} \mu_w \ln(\rho_w)$ under the constraints $\sum_{w \in W^o} \rho_w = R$ and $\rho_w \leq \Delta_w$ for every $w \in W^o$. Consider its relaxation with $\rho_w \leq \Delta_w$ for every $w \in W^o$ ignored. By the equivalent of Proposition 6 for this reformulation, $\vec{\rho}'^*$ is unique and must have $p'^*_w = \rho'^*_w / \Delta_w$ for each source w. The Lagrangian of $\mathcal{L}(\vec{\rho}, \lambda_0) := \overline{J}_{mod}(\vec{\rho}) - \lambda_0(\sum_{w \in W^o} \rho_w - R)$ must have $\nabla \mathcal{L}(\vec{\rho}'^*, \lambda'^*_0) = 0$ for the optimal solution $(\vec{\rho}'^*, \lambda'^*_0)$ (which, again, encodes the optimal p'^*_w for the relaxation of the original formulation of Problem 2 via $p'^*_w = \rho'^*_w / \Delta_w$). From this we see that $\frac{\partial}{\partial \rho_u} \overline{J}_{mod} \restriction_{\rho'^*_u} = \lambda'^*_0 = \frac{\partial}{\partial \rho_w} \overline{J}_{mod} \restriction_{\rho'^*_w}$, for any sources $u, w \in W^o$. Since $\frac{\partial}{\partial \rho_w} \overline{J}_{mod} = \frac{\mu_w}{\rho_w}$, this implies

$$\frac{\mu_u}{\rho'^*_u} = \frac{\mu_w}{\rho'^*_w} \quad \text{for all } u, w \in W^o \,. \tag{18}$$

Consider the slack-variable formulation of Problem 2 with slack variables $\{q_w\}_{w \in W^o}$. Inequality constraints in this formulation turn into $\rho_w = \Delta_w - q_w$ for $q_w \geq 0$. In this formulation, we now have

$$\mathcal{L}(\vec{\rho}, \lambda_0, \lambda_1, \ldots, \lambda_{|W^o|}) := \mathcal{L}(\vec{\rho}, \lambda_0) - \sum_{w \in W^o} \lambda_w (\rho_w - \Delta_w) \,.$$

where, by the Karush-Kuhn-Tucker conditions, $\lambda_w \geq 0$ for all w. By complementary slackness, $\lambda^*_w = 0$ for every $w \in W^o$ such that $q_w > 0$, i.e., for every w that does *not* activate its inequality constraint, under the optimal solution $(\vec{\rho}^*, \lambda^*_0, \lambda^*_w, \ldots, \lambda^*_{|W^o|})$. This implies that $\frac{\partial}{\partial \rho_u} \overline{J}_{mod} \restriction_{\rho^*_u} - \lambda^*_u = \lambda^*_0 = \frac{\partial}{\partial \rho_w} \overline{J}_{mod} \restriction_{\rho^*_w} - \lambda^*_w$, for any sources $u, w \in W^o$, i.e.,

$$\frac{\mu_u}{\rho'^*_u} - \lambda^*_u = \lambda^*_0 = \frac{\mu_w}{\rho'^*_w} - \lambda^*_w \quad \text{for all } u, w \in W^o \,. \tag{19}$$

Now, suppose for contradiction that there is a source $u \in W^o$ that has $p'^*_u > 1$ but $p^*_u < 1$, implying that $\rho'^*_u > \Delta_u$ but $\rho^*_u < \Delta_u$. This, in turn, implies that (a) $\rho'^*_u > \rho^*_u$ and (b) $\lambda^*_u = 0$, since u doesn't activate its inequality constraint under $\vec{\rho}^*$ (and hence under $\vec{p}^*$). Then, since $\sum_{v \in W^o} \rho^*_v = \sum_{v \in W^o} \rho'^*_v = R$, there must also exist some other source $w \neq u$ such that $\rho'^*_w < \rho^*_w$. For this source, $\lambda^*_w \geq 0$, so $\lambda^*_w \geq \lambda^*_u$.

Recall that $\overline{J}_{mod}$ is strictly concave, and its partial derivatives $\frac{\mu_v}{\rho_v}$ are monotone decreasing in every non-negative ρ_v. Together with $\rho'^*_u > \rho^*_u$, $\rho'^*_w < \rho^*_w$, $\lambda^*_w \geq \lambda^*_u$, and Equation 19, this implies

$$\frac{\mu_u}{\rho'^*_u} < \frac{\mu_u}{\rho^*_u} \leq \frac{\mu_w}{\rho^*_w} < \frac{\mu_w}{\rho'^*_w}$$

But this contradicts Equation 18, completing the proof of the lemma. ■

The optimality of LAMBDACRAWL-COMPLOBS now follows by induction. Its every iteration except the last one identifies at least one constraint that is active under $\vec{p}^*$, by the above lemma, and thereby assigns an optimal p^*_w to some sources, leaving optimal crawl probabilities for others to be found in subsequent iterations. The solution for the sources remaining in the final iteration, which does not violate any inequality constraints, is optimal by Proposition 6. Therefore, LAMBDACRAWL-COMPLOBS arrives at the optimal solution, and that solution is unique because Problem 2's maximization objective is concave as a sum of concave functions, and the optimization region is convex.

We note that the proof so far is similar to the proof of Lemma 3.3 from (Kolobov et al., 2019) for a different concave function F under constraints of the form $x_1 + \ldots + x_k \leq c_k$, where $x_1, \ldots, x_k$ is a subset of F's variables and c_k is a constraint.

Since in each iteration LAMBDACRAWL-COMPLOBS removes at least one source from further consideration, it makes at most $|W^o|$ iterations. In each iteration it applies Proposition 6, which takes $O(|W^o|)$ time, yielding the overall time complexity of $O(|W^o|^2)$. ■

PROOF OF PROPOSITION 8. See the paper. ■

PROOF OF PROPOSITION 9. LAMBDALEARNANDCRAWL starts with strictly positive finite estimates $\vec{\hat{\Delta}}_0$ of change rates. Since LAMBDACRAWL, which LAMBDALEARNANDCRAWL uses for determining crawl rates for the next epoch, is optimal to any desired precision (Proposition 8), it follows from Proposition 1 that it returns positive $\vec{\hat{\rho}}^*_1, \vec{\hat{p}}^*_1 > 0$, and $0 < \vec{\mu}, R < \infty$ guarantees that these crawl rates are also finite. In subsequent iterations, $\vec{\hat{\Delta}}_n$ are estimated using Equations 14 and 15 with smoothing terms (lines 11 and 13 of Algorithm 4), and the smoothing terms ensure that the change rate estimates are finite and bounded away from 0: $0 < \delta_{min} \leq \vec{\hat{\Delta}}_n < \infty$, where δ_{min} is implied by the aforementioned estimators and specific smoothing term values. This, along with finite positive $\vec{\mu}$ and R, ensures that $0 < \vec{\hat{\rho}}^*_{n+1}, \vec{\hat{p}}^*_{n+1} < \delta_{max}$. Hence, by induction, no source is ever starved, and no source is crawled infinitely frequently. This ensures, together with consistency of estimators from Equations 14 and 15, that if at least the last iteration N_{epoch} uses the entire observation history, i.e. $S(N_{epoch}) = length(obs\_hist)$, then change rate estimates converge to the true change rates in probability: $\text{plim}_{N_{epochs} \to \infty} \vec{\hat{\Delta}}_{N_{epochs}} = \vec{\Delta}$, as long as $\vec{\Delta}$ doesn't change with time. Optimality of LAMBDACRAWL then implies probabilistic convergence of $(\vec{\hat{\rho}}^*_{N_{epochs}}, \vec{\hat{p}}^*_{N_{epochs}})$ to $(\vec{\hat{\rho}}^*, \vec{\hat{p}}^*)$ as well. ■

PROOF OF PROPOSITION 10. According to Equation system 7, the parameter vector $\vec{\rho}$ of the optimal $\pi^* \in \Pi^-$ that minimizes the expected harmonic penalty in the absence of remote change observations (Problem 1) satisfies

$$\begin{cases} \rho_w = \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w \Delta_w}{\lambda}}}{2}, & \text{for all } w \in W^- \\ \sum_{w \in W^-} \rho_w = R \end{cases}$$

If $\frac{\mu_w}{\Delta_w} = c$ for all $w \in W^-$, $c > 0$, then we can express $\Delta_w = c' \mu_w$ where $c' = 1/c$ and plug it into the above equations to get

$$\begin{aligned}
\rho_w &= \frac{-\Delta_w + \sqrt{\Delta_w^2 + \frac{4\mu_w\Delta_w}{\lambda}}}{2} \\
&= \frac{-c'\mu_w + \sqrt{(c'\mu_w)^2 + \frac{4c'\mu_w^2}{\lambda}}}{2} \\
&= \frac{-c'\mu_w + \mu_w\sqrt{\frac{c'^2\lambda+4c'}{\lambda}}}{2} \\
&= \mu_w \left(\frac{-c' + \sqrt{\frac{c'^2\lambda+4c'}{\lambda}}}{2} \right) \text{ for all } w \in W^- \qquad (20)
\end{aligned}$$

Plugging this into the remaining equation from the above system, $\sum_{w\in W^-} \rho_w = R$, we get

$$\begin{aligned}
\sum_{w\in W^-} \mu_w \left(\frac{-c' + \sqrt{\frac{c'^2\lambda+4c'}{\lambda}}}{2} \right) = R \implies \\
\frac{-c' + \sqrt{\frac{c'^2\lambda+4c'}{\lambda}}}{2} = \frac{R}{\sum_{w\in W^-}\mu_w} \implies \\
\frac{c'^2\lambda + 4c'}{\lambda} = \left(\frac{2R}{\sum_{w\in W^-}\mu_w} + c' \right)^2 \implies \\
\lambda \left(\frac{2R}{\sum_{w\in W^-}\mu_w} + c' \right)^2 - c'^2\lambda = 4c' \implies \\
\lambda = \frac{4c'}{\left(\frac{2R}{\sum_{w\in W^-}\mu_w} + c' \right)^2 - c'^2}
\end{aligned}$$

Plugging this and $\Delta_w = c'\mu_w$ back into Equations 20, we get for all $w \in W^-$

$$
\begin{aligned}
\rho_w &= \mu_w \left(\frac{-c' + \sqrt{\frac{c'^2 \left(\frac{4c'}{\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 - c'^2} \right) + 4c'}{\frac{4c'}{\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 - c'^2}}}}{2} \right) \\
&= \mu_w \left(\frac{-c' + \sqrt{\frac{4c' \left(\left(\frac{c'^2}{\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 - c'^2} \right) + 1 \right)}{\frac{4c'}{\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 - c'^2}}}}{2} \right) \\
&= \mu_w \left(\frac{-c' + \sqrt{\frac{\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 \left(\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 - c'^2 \right)}{\left(\frac{2R}{\sum_{w \in W^-} \mu_w} + c' \right)^2 - c'^2}}}{2} \right) \\
&= \mu_w \left(-c' + \frac{R}{\sum_{w \in W^-} \mu_w} + c' \right) \\
&= \frac{\mu_w R}{\sum_{w \in W^-} \mu_w}
\end{aligned}
$$

■

10. Details of the Experiments and Additional Plots

In the experiments we used the following algorithms for comparison:

***ImportanceCrawl* (*IC*)** is the name we give to the heuristic proposed by Cho & Garcia-Molina (2003a) that sets

$$\rho_w = \frac{\mu_w R}{\sum_{w' \in W} \mu_{w'}},$$

i.e., crawls sources at a rate proportional to their importance. This heuristic doesn't differentiate between pages with and without remote change observations. Note that our Proposition 10 implies that *ImportanceCrawl* is actually a special case of LAMBDACRAWL— without the LAMBDACRAWL-COMPLOBS component and assuming that $\frac{mu_w}{\Delta_w}$ is the same for all sources in W. Thus, our theory identifies conditions under which *ImportanceCrawl*, previously considered merely a heuristic, is optimal.

***ChangeRateCrawl* (*CC*)** is the name we give to another heuristic proposed by Cho & Garcia-Molina (2003a) that sets

$$\rho_w = \frac{\Delta_w R}{\sum_{w' \in W} \Delta_{w'}},$$

thereby crawling sources at a rate proportional to their change rate. Cho & Garcia-Molina (2003a) pointed out that *ChangeRateCrawl* can be very suboptimal if the set W includes sources that change frequently. This causes *ChangeRateCrawl* to over-commit crawl bandwidth to sources whose changes are near-impossible to keep up with, at the expense of almost ignoring the rest. Our experimental results agree with this observation — *ChangeRateCrawl* turned out to be the second weakest-performing baseline.

***BinaryCrawl* (*BC*)** is the algorithm proposed by Azar et al. (2018) that produces crawl rates that minimizes binary staleness or, as this objective is equivalently stated in Azar et al. (2018), maximizes binary freshness

$$\lim_{T \to \infty} \mathbb{E}\left[\frac{1}{T}\int_0^T \left(\sum_{i=1}^n \mu_i \cdot \mathbb{1}_{\text{FRESH}(i,t)}\right) dt\right]$$

.

It is related to our objective, but has a major difference — according to it, the marginal penalty for the 2nd, 3rd, etc change at a source is 0.

This difference is crucial in practice, because minimizing binary staleness generally yields $\vec{\rho}^*$ with $\rho^*_w = 0$ for many sources even if they $\Delta_w > 0$. This effectively tells the tracker to ignore changes to these sources — an unacceptable strategy in real applications. Indeed, harmonic penalty (Equation 1) assigns $J^\pi = \infty$ to such strategies, causing *BinaryCrawl* to be the weakest-performing baseline in our experiment, in spite of its formal optimality guarantee w.r.t. binary staleness.

***UniformCrawl* (*UC*)** is a heuristic that assigns an equal crawl rate to all sources:

$$\rho_w = R/|W|.$$

In spite of its simplicity, in our experiments it outperforms *ChangeRateCrawl*, as predicted by Cho & Garcia-Molina (2003a), and *BinaryCrawl* (Azar et al., 2018).

Dataset details. Proposition 10 suggests that the performance gap between LAMBDACRAWL, LAMBDACRAWLAPPROX, and *ImportanceCrawl* depends on the distribution of the ratios $\frac{\mu_w}{\Delta_w}$ across the set of sources W: if they are all equal, the policy cost of LAMBDACRAWL and LAMBDACRAWLAPPROX should be the same, with *ImportanceCrawl*'s being similar (dependent on the fraction of sources with complete remote observations, which *ImportanceCrawl* ignores). As Figure 4 shows, these ratios are indeed similar across much of our dataset.

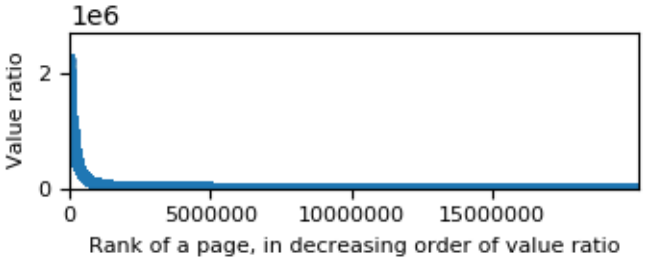

Figure 4. Ranking of 20,151,303 URLs in our dataset by their $\frac{\mu_w}{\Delta_w}$ ratio. This ratio distribution is nearly constant across most of the dataset, except the URLs at the head of the ranking.

Details of the reinforcement learning experiment (Figure 3). The goal of the last experiment was assessing the convergence properties of LAMBDALEARNANDCRAWL and LAMBDALEARNANDCRAWLAPPROX. We sampled 100,000 URLs from the aforementioned 20M-URL dataset uniformly at random. Using the dataset, we simulated runs of LAMBDALEARNANDCRAWL and its Proposition 10-based approximation LAMBDALEARNANDCRAWLAPPROX. We set the length of 1

epoch (Algorithm 4) to 14 days, and simulated a 98-day crawling and learning process as follows. We generated 98 days' worth of page changes using the ground-truth change rates (unknown to LAMBDALEARNANDCRAWL and LAMBDALEARNANDCRAWLAPPROX at the beginning). As in the real dataset, for 13% of the URLs we (simulated) instantaneous change notifications to our algorithms. Then we ran LAMBDALEARNANDCRAWL and LAMBDALEARNANDCRAWLAPPROX on this simulated data, which at each epoch n reestimated page change rates using data from executing their policies in the previos epochs, recomputed the policy, and executed it during the n-th epoch. For the 1st epoch, both algorithms ran the "warm-start" policy produced by *ImportanceCrawl*, which doesn't need change rate estimates. At the start of each epoch as well as the end of the last, 6th epoch, we evaluated the algorithms' newly computed policies using the ground truth change rates from the dataset Equation 13. We repeated the above procedure 20 times and plotted the results in Figure 3. Along with LAMBDALEARNANDCRAWL's and LAMBDALEARNANDCRAWLAPPROX's performance we plotted the (hypothetical) policy quality of LAMBDACRAWL and LAMBDACRAWLAPPROX assuming the ground-truth change rates were known to them — this is the asymptotic performance level of the learning algorithms. Figure 3 also shows the performance of *ImportanceCrawl* for comparison: the RL algorithms start with its performance due to using it as a warm start, but improve theirs thanks to learning to nearly asymptotic level after just 2 epochs.