Learning to Crawl

Utkarsh Upadhyay
Resonal, Berlin, Germany
utkarsh@reason.al

Róbert Busa-Fekete
Google Research, NY, USA
busarobi@google.com

Wojciech Kotłowski
Poznan University of Technology, Poland
wkotlowski@cs.put.poznan.pl

Dávid Pál
Yahoo! Research, NY, USA
davidko.pal@gmail.com

Balázs Szörényi
Yahoo! Research, NY, USA
szorenyi.balazs@gmail.com

Abstract

Web crawling is the problem of keeping a cache of webpages *fresh*, *i.e.*, having the most recent copy available when a page is requested. This problem is usually coupled with the natural restriction that the bandwidth available to the web crawler is limited. The corresponding optimization problem was solved optimally by Azar et al. (2018) under the assumption that, for each webpage, both the elapsed time between two changes and the elapsed time between two requests follows a Poisson distribution with *known* parameters. In this paper, we study the same control problem but under the assumption that the change rates are *unknown* a priori, and thus we need to estimate them in an online fashion using only partial observations (*i.e.*, single-bit signals indicating whether the page has changed since the last refresh). As a point of departure, we characterise the conditions under which one can solve the problem with such partial observability. Next, we propose a practical estimator and compute confidence intervals for it in terms of the elapsed time between the observations. Finally, we show that the explore-and-commit algorithm achieves an $\mathcal{O}(\sqrt{T})$ regret with a carefully chosen exploration horizon. Our simulation study shows that our online policy scales well and achieves close to optimal performance for a wide range of parameters.

1 Introduction

As information dissemination in the world becomes near real-time, it becomes more and more important for search engines, like Bing and Google, and other knowledge repositories to keep their caches of information and knowledge fresh. In this paper, we consider the web-crawling problem of designing policies for refreshing webpages in a local cache with the objective of maximizing the number of incoming requests which are served with the latest version of the page. Webpages are the simplest and most ubiquitous source of information on the internet. As items to be kept in a cache, they have two key properties: (i) they need to be polled, which uses bandwidth, and (ii) polling them only provides partial information about their change process, *i.e.*, a single bit indicating whether the webpage has changed since it was last refreshed or not. Cho and Garcia-Molina (2003a) in their seminal work presented a formulation of the problem which was recently studied by Azar et al. (2018). Under the assumption that the changes to the webpages and the requests are Poisson processes with known rates, they describe an efficient algorithm to find the optimal refresh rates for the webpages.

However, the change rates of the webpages are often not known in advance and need to be estimated. Since the web crawler cannot continuously monitor every page, there is only partial information available on the change process. Cho and Garcia-Molina (2003b), and more recently Li, Cline, and Loguinov (2017), have proposed estimators of the rate of change given partial observations. However, the problem of learning the refresh rates of items while also trying to optimise the objective of keeping the cache as up-to-date for incoming requests as possible seems very challenging. On the one hand, the optimal policy found by the algorithm by Azar et al. (2018) does not allocate bandwidth for pages that are changing very frequently, and, on the other hand, rate estimates with low precision, especially for those that are changing frequently, may result in a policy that has non-vanishing regret. We formulate this web-crawling problem with unknown change rates as an online optimization problem for which we define a natural notion of regret, describe the conditions under which the refresh rates of the webpages can be learned, and show that using a simple explore-and-commit algorithm, one can obtain regret of order $\sqrt{T}$.

Though in this paper we primarily investigate the problem of web-crawling, our notion of regret and the observations we make about the learning algorithms can also be applied to other control problems which model the actions of agents as Poisson processes and the policies as intensities, including alternate objectives and observation regimes for the web-crawling problem itself. Such an approach is seen in recent works which model or predict social activities (Farajtabar 2018; Du et al. 2016), control online social actions (Zarezade et al. 2017; Karimi et al. 2016; Wang et al. 2017; Upadhyay, De, and Gomez-Rodriguez 2018), or even controlling spacing of items for optimal learning (Tabibian et al. 2019). All such problems admit online versions where the parameters for the models (*e.g.*, difficulty of items from recall events (Tabibian et al. 2019), or rates of posting of messages by other broadcasters (Karimi et al. 2016)) need to be learned while also optimising the policy the agent follows.

In Section 2, we will formally describe the problem setup and formulate the objective with bandwidth constraints. Section 3 takes a closer look at the objective function and the optimal policy to describe the properties any learning algorithm should have. We propose an estimator for learning the parameters of Poisson process with partial observability and provide guarantees on its performance in Section 4. Leveraging the bound on the estimator's performance, we propose a simple explore-and-commit algorithm in Section 5 and show that it achieves $\mathcal{O}(\sqrt{T})$ regret. In Section 6, we test our algorithm using real data to justify our theoretical findings and we conclude with future research directions in Section 7.

2 Problem Formulation

In this section, we consider the problem of keeping a cache of m webpages up-to-date by modelling the changes to those webpages, the requests for the pages, and the bandwidth constraints placed on a standard web-crawler. We assume that the cache is empty when all processes start at time 0.

We model the changes to each webpage as Poisson processes with constant rates. The parameters of these *change processes* are denoted by $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_m]$, where $\xi_i > 0$ denotes the rate of changes made to webpage i. We will assume that $\boldsymbol{\xi}$ are not known to us but we know only an upper bound $\xi_{\max}$ and lower bound $\xi_{\min}$ on the change rates. The crawler will learn $\boldsymbol{\xi}$ by refreshing the pages and observing the single-bit feedback described below. We denote the time webpage i changes for the nth time as $x_{i,n}$. We model the incoming requests for each webpage also as Poisson processes with constant rates and denote these rates as $\boldsymbol{\zeta} = [\zeta_1, \zeta_2, \ldots, \zeta_m]$. We will assume that these rates, which can also be interpreted as the *importance* of each webpage in our cache, are known to the crawler. We will denote the time webpage i is requested for the nth time as $z_{i,n}$. The change process and the request process, given their parameters, are assumed to be independent of each other.

We denote time points when page i is refreshed by the crawler using $(y_{i,n})_{n=1}^{\infty}$. The feedback which the crawler gets after refreshing a webpage i at time $y_{i,n}$ consists of a single bit which indicates whether the webpage has changed or not since the last observation, if any, that was made at time $y_{i,n-1}$. Let $E_i^{\emptyset}[t_0, t]$ indicate the event that neither a change nor a refresh of the page has happened between time t_0 and t for webpage i. Define $\textsc{Fresh}(i,t)$ as the event that the webpage is *fresh* in the cache at time t. Defining the maximum of an empty set to be $-\infty$, we have:

$$\textsc{Fresh}(i,t) = \begin{cases} 0 & \text{if } E_i^{\emptyset}[0,t] \\ 1_{(\max\{x_{i,j}:x_{i,j}<t\}<\max\{y_{i,j}:y_{i,j}<t\})} & \text{if } \neg E_i^{\emptyset}[0,t] \end{cases}$$

where the indicator function $1_{(\bullet)}$ takes value 1 on the event in its argument and value 0 on its complement. Hence, we can describe the feedback we receive upon refreshing a page i at time $y_{i,n}$ as:

$$o_{i,n} = \textsc{Fresh}(i, y_{i,n}). \tag{1}$$

We call this a *partial* observation of the change process to contrast it with *full* observability of the process, *i.e.*, when a refresh at $y_{i,n}$ provides the number of changes to the webpage in the period $(y_{i,n}, y_{i,n-1})$. For example, the crawler will have full observability of the incoming request processes.

The policy space Π consists of all measurable functions which, at any time t, decide when the crawler should refresh which page in its cache based on the observations up to time t that includes $\{(o_{i,n})_{n=1}^{N} \mid N = \operatorname{argmax}_n y_{i,n} < t; i \in [m]\}$.

The objective of the web-crawling problem is to refresh webpages such that it maximizes the number of requests which are served a fresh version. So the utility of a policy $\pi \in \Pi$ followed from time t_1 to t_2 can be written as:

$$U([t_1,t_2],\pi;\boldsymbol{\xi}) = \frac{1}{m}\sum_{i=1}^{m}\sum_{t_1 \le z_{i,n} \le t_2} \textsc{Fresh}(i, z_{i,n}). \tag{2}$$

Our goal is to find a policy that maximizes this utility (2).[1] However, if the class of policies is unconstrained, the utility can be maximized by a trivial policy which continuously refreshes all webpages in the cache. This is not a practical policy since it will overburden the web servers and the crawler. Therefore, we would like to impose a bandwidth constraint on the crawler. Such a constraint can take various forms and a natural way of framing it is that the expected number of webpages that are refreshed in *any* time interval with width w cannot exceed $w \times R$. This constraint defines a class of stochastic policies $\Delta_R = \{(\rho_1, \ldots, \rho_m) \in (\mathbb{R}^+)^m : \sum_{i=1}^{m} \rho_i = R\} \subset \Pi$, where each webpage's refresh time is drawn by the crawler from a Poisson process with rate ρ_i.

This problem setup was studied by Azar et al. (2018) and shown to be tractable. Recently, Kolobov et al. (2019a) have studied a different objective function which penalises the *harmonic staleness* of pages and is similarly tractable. Another example of such an objective (albeit with full observability) is proposed by Sia, Cho, and Cho (2007). We show later that our analysis extends to their objective as well.

We define the regret of policy $\pi \in \Delta_R$ as follows

$$R(T,\pi;\boldsymbol{\xi}) = \max_{\pi' \in \Delta_R} \mathbb{E}\left[U([0,T],\pi';\boldsymbol{\xi})\right] - \mathbb{E}\left[U([0,T],\pi;\boldsymbol{\xi})\right].$$

It is worth reiterating that the parameters $\boldsymbol{\xi}$ will not be known to the crawler. The crawler will need to determine when and which page to refresh given only the single bits of information $o_{i,n}$ corresponding to each refresh the policy makes.

3 Learning with Poisson Processes and Partial Observability

In this section, we will derive an analytical form of the utility function which is amenable to analysis, describe how to un-

[1]The freshness of the webpages does depend on the policy π which is hidden by function $\textsc{Fresh}$.

cover the optimal policy in Δ_R if all parameters (*i.e.*, $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$) are known, and consider the problem of learning the parameters $\boldsymbol{\xi}$ with partial observability. We will use the insight gained to determine some properties a learning algorithm should have so that it can be analysed tractably.

3.1 Utility and the Optimal policy

Consider the expected value of the utility of a policy $\boldsymbol{\rho} \in \Delta_R$ which the crawler follows from time t_0 till time T. Assume that the cache at time t_0 is given by $\boldsymbol{S}(t_0) = [s_1, s_2, \dots, s_m] \in \{0,1\}^m$, where $s_i = \textsc{Fresh}(i, t_0)$. Then, using (2), we have:

$$\begin{aligned}
&\mathbb{E}[U([t_0,T],\boldsymbol{\rho};\boldsymbol{\xi}) \mid \boldsymbol{S}(t_0)] \\
&= \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}\Bigg[\sum_{t_0<z_{i,n}<T}\textsc{Fresh}(i,z_{i,n})\,\Bigg|\,\textsc{Fresh}(i,t_0)=s_i\Bigg] \\
&= \frac{1}{m}\sum_{i=1}^{m}\int_{t_0}^{T}\underbrace{\zeta_i \mathbf{P}_{\boldsymbol{\rho}}\left(\textsc{Fresh}(i,t)=1 \mid \textsc{Fresh}(i,t_0)=s_i\right)}_{F^{(i)}_{[t_0,t]}(\boldsymbol{\rho};\boldsymbol{\xi})}\,dt
\end{aligned} \tag{3}$$

where (3) follows from Campbell's formula for Poisson Process (Kingman 1993) (expectation of a sum over the point process equals the integral over time with process' intensity measure) as well as the fact that the request process and change/refresh processes are independent.[2] In the next lemma, we show that the differential utility function $F^{(i)}_{[t_0,t]}(\boldsymbol{\rho};\boldsymbol{\xi})$, defined implicitly in (3), can be made time-independent if the policy is allowed to run for *long-enough*.

Lemma 1 (Adapted from (Azar et al. 2018))**.** *For any given $\varepsilon > 0$, let $\boldsymbol{\rho} \in \Delta_R$ be a policy which the crawler adopts at time t_0 and let the initial state of the cache be $\boldsymbol{S}(t_0) = [s_1, s_2, \dots, s_m] \in \{0,1\}^m$, where $s_i = \textsc{Fresh}(i, t_0)$. Then if $t - t_0 \geq \frac{1}{\xi_{\min}}\log\left(\frac{2\sum_{j=1}^{m}\zeta_i}{\varepsilon}\right)$, then $\sum_{i=1}^{m}\frac{\zeta_i\xi_i}{\xi_i+\rho_i} < \sum_{i=1}^{m}F^{(i)}_{[t_0,t]}(\boldsymbol{\rho};\boldsymbol{\xi}) < \sum_{i=1}^{m}\frac{\zeta_i\xi_i}{\xi_i+\rho_i} + \varepsilon$.*

Hence, as long as condition described by Lemma 1 holds, the differential utility function for a policy $\boldsymbol{\rho} \in \Delta_R$ is time independent and can be written as just $F(\boldsymbol{\rho};\boldsymbol{\xi}) = \sum_{i=1}^{m}\frac{\zeta_i\xi_i}{\xi_i+\rho_i}$. Substituting this into (3), we get:

$$\begin{aligned}
\mathbb{E}[U([t_0,T],\boldsymbol{\rho};\boldsymbol{\xi})] &\approx \frac{1}{m}\sum_{i=1}^{m}\int_{t_0}^{T}\frac{\rho_i}{\rho_i+\xi_i}\zeta_i\,dt \\
&= \frac{T-t_0}{m}\sum_{i=1}^{m}\frac{\rho_i\zeta_i}{\rho_i+\xi_i}
\end{aligned} \tag{4}$$

This leads to the following time-horizon independent optimisation problem for the optimal policy:

$$\underset{\boldsymbol{\rho}\in\Delta_R}{\text{maximize}}\ F(\boldsymbol{\rho};\boldsymbol{\xi}) = \sum_{i=1}^{m}\frac{\rho_i\zeta_i}{\rho_i+\xi_i} \tag{5}$$

[2]Note that the *rates* of the processes can be correlated; only the events need to be drawn independently.

Azar et al. (2018) have considered the approximate utility function given by (5) to derive the optimal refresh rates $\boldsymbol{\rho}^\star$ for known $\boldsymbol{\xi}$ in $\mathcal{O}(m\log m)$ time (See Algorithm 2 in (Azar et al. 2018)).[3]

This approximation has bearing upon the kind of learning algorithms we could use while keeping the analysis of the algorithm and computation of the optimal policy tractable. The learning algorithm we employ must follow a policy $\boldsymbol{\rho} \in \Delta_R$ for a certain amount of *burn-in* time before we can use (4) to approximate the performance of the policy. If the learning algorithm changes the policy *too quickly*, then we may see large deviations between the actual utility and the approximation. However, if the learning algorithm changes the policy *slowly*, where Lemma 1 can serve as a guide to the appropriate period, then we can use (4) to easily calculate its performance between t_0 and T. Similar conditions can be formulated for the approximate objectives proposed by Kolobov et al. (2019a) and Sia, Cho, and Cho (2007) as well.

Now that we know how to uncover the optimal policy when $\boldsymbol{\xi}$ are known, we turn our attention to the task of learning it with partial observations.

3.2 Learnability with Partial Observations

In this section, we address the problem of partial information of Poisson process and investigate under what condition the rate of the Poisson process can be estimated. In our setting, for an arbitrary webpage, we only observe binary outcomes $(o_n)_{n=1}^{\infty}$, defined by (1). The refresh times $(y_n)_{n=1}^{\infty}$ and the Poisson process of changes with rate ξ induce a distribution over $\{0,1\}^{\mathbb{N}}$ which is denoted by μ_ξ. If the observations happen at regular time intervals, i.e. $y_n - y_{n-1} = c$ for some constant c, then the support $\mathcal{S}_\xi$ of μ_ξ is:

$$\mathcal{S}_\xi = \left\{(o_n)_{n=1}^{\infty} \in \{0,1\}^{\mathbb{N}} : \lim_{n\to\infty}\frac{\sum_{j=1}^{n} o_j}{n} = 1 - e^{-c\,\xi}\right\}.$$

This means that we can have a consistent estimator, based on the strong law of large numbers, if the crawler refreshes the cache at fixed intervals.

However, we can characterise the necessary property of the set of partial observations which allows parameter estimation of Poisson processes under the general class of policies Π. This result may be of independent interest.

Lemma 2. *Let $\{y_0 := 0\} \cup (y_n)_{n=1}^{\infty}$ be a sequence of times, such that $\forall n.\, y_n > 0$, at which observations $(o_n)_{n=1}^{\infty} \in \{0,1\}^{\mathbb{N}}$ are made of a Poisson process with rate ξ, such that $o_n := 1$ iff there was an event of the process in $(y_{n-1}, y_n]$, define $w_n = y_n - y_{n-1}$, $I = \{n : w_n < 1\}$ and $J = \{n : w_n \geq 1\}$. Then:*

1. *If $\sum_{n\in I} w_n < \infty$ and $\sum_{n\in J} e^{-\xi w_n} < \infty$, then any statistic for estimating ξ has non-vanishing bias.*

[3]The optimal policy can be obtained in $\mathcal{O}(m)$ time by using the method proposed by Duchi et al. (2008).

2. *If $\sum_{n\in I} w_n = \infty$, then there exist disjoint subsets $I_1, I_2, \dots$ of I such that $\left(\sum_{n\in I_k} w_n\right)_{k=1}^{\infty}$ is monotone and $\sum_{n\in I_k} w_n \in (1,2)$ for $k = 1, 2, \dots$ For any such sequence $\mathcal{I} = (I_k)_{k=1}^{\infty}$, the mapping $c_{\mathcal{I}}(\xi) = \lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} \exp\left(-\xi \sum_{n\in I_k} w_n\right)$ is strictly monotone and*
$$\left[\frac{1}{K}\sum_{k=1}^{K} \mathbb{I}\left(\sum_{n\in I_k} o_n \geq 1\right)\right] \xrightarrow{a.s.} 1 - c_{\mathcal{I}}(\xi).$$

3. *If $\sum_{n\in J} e^{-w_n\xi} = \infty$ then, there exists a sequence $\mathcal{J} = (J_k)_{k=1}^{\infty}$ of disjoint subsets of J such that $\left(\sum_{n\in J_k} e^{-w_n\xi}\right)_{k=1}^{\infty}$ is monotone and $\sum_{n\in J_k} e^{-w_n\xi} \in [1/e, 2/e)$ for $k = 1, 2, \dots$ For any such $\mathcal{J}$, the mapping $c_{\mathcal{J}}(\xi) = \lim_{K\to\infty}\left[\frac{1}{K}\sum_{k=1}^{K}\prod_{n\in J_k}\left(1 - e^{-\xi w_n}\right)\right]$ is strictly monotone and*
$$\lim_{K\to\infty}\left[\frac{1}{K}\sum_{k=1}^{K} \mathbb{I}\left(o_n \geq 1,\ \forall n \in J_k\right)\right] \xrightarrow{a.s.} c_{\mathcal{J}}(\xi),$$

Note that it is possible that, for some ξ, the statistics almost surely converge to a value that is unique to ξ, but for some other one they do not. Indeed, when $w_n = \ln n$, then $\sum_{n\in I} w_n < \infty$ and $\sum_{n\in J} e^{-\xi w_n} < \infty$ for $\xi = 2$, but $\sum_{n\in J} e^{-\xi w_n} = \infty$ for $\xi = 1$. More concretely, assuming that the respective limits exist, we have:
$$\liminf_{n\in J} \frac{w_n}{\ln n} > \frac{1}{\xi} \implies \sum_n e^{-\xi w_n} < \infty \quad \text{and}$$
$$\limsup_{n\in J} \frac{w_n}{\ln n} \leq \frac{1}{\xi} \implies \sum_n e^{-\xi w_n} = \infty.$$

In particular, if $\limsup_{n\in J} \frac{w_n}{\ln n} = 0$, it implies that $\sum_{n\in J} e^{-\xi w_n} = \infty$ for all $\xi > 0$, which, in turn, implies that it will be possible to learn the true value for any parameter $\xi > 0$.

Lemma 2 has important implications on the learning algorithms we can use to learn $\boldsymbol{\xi}$. It suggests that if the learning algorithm decreases the refresh rate ρ_i for a webpage too quickly, such that $\mathbf{P}\left(\liminf_{n\to\infty} \frac{w_{i,n}}{\ln n} > \frac{1}{\xi_i}\right) > 0$ (assuming the limit exists), then the estimate of each parameter ξ_i has non-vanishing error.

In summary, in this section, we have made two important observations about the learning algorithm we can employ to solve the web-crawling problem. Firstly, given an error tolerance of $\varepsilon > 0$, the learning algorithm should change the policy only after $\sim \frac{1}{\xi_{\min}} \log\left(2\sum_i \zeta_i/\varepsilon\right)$ steps to allow for time invariant differential utility approximations to be valid. Secondly, in order to obtain consistent estimates for $\boldsymbol{\xi}$ from partial observations, the learning algorithm should not change the policy so drastically that it violates the conditions in Lemma 2. These observations strongly suggest that to obtain theoretical guarantees on the regret, one should use *phased* learning algorithms where each phase of the algorithm is of duration $\sim \frac{1}{\xi_{\min}} \log\left(2\sum_i \zeta_i/\varepsilon\right)$, the policy is only changed when moving from one phase to the other, and the changes made to the policy are such that constraints presented in Lemma 2 are not violated. Parallels can be drawn between such learning algorithms and the algorithms used for online learning of Markov Decision Processes which rely on bounds on mixing times (Neu et al. 2010). In Section 5, we present the simplest of such algorithms, *i.e.*, the explore-and-commit algorithm, for the problem and provide theoretical guarantees on the regret. Additionally, in Section 6.2, we also empirically compare the performance of ETC to the phased ε-greedy learning algorithm.

In the next section, we investigate practical estimators for the parameters $\widehat{\boldsymbol{\xi}}$ and the formal guarantees they provide for the web-crawling problem.

4 Parameter Estimation and Sensitivity Analysis with Partial Observations

In this section, we address the problem of parameter estimation of Poisson processes under partial observability and investigate the relationship between the utility of the optimal policy $\boldsymbol{\rho}^\star$ (obtained using true parameters) and policy $\widehat{\boldsymbol{\rho}}$ (obtained using the estimates).

Assume the same setup as for Lemma 2, *i.e.*, we are given a finite sequence of observation times $\{y_0 := 0\} \cup (y_n)_{n=1}^{N}$ in advance, and we observe $(o_n)_{n=1}^{N}$, defined as in (1), based on a Poisson process with rate ξ. Define $w_n = y_n - y_{n-1}$. Then log-likelihood of $(o_n)_{n=1}^{N}$ is:
$$\mathcal{L}(\xi) = \sum_{n:o_n=1} \ln(1 - e^{-\xi w_n}) - \sum_{n:o_n=0} \xi w_n \qquad (6)$$
which is a concave function. Taking the derivative and solving for ξ yields the maximum likelihood estimator (Cho and Garcia-Molina 2003b). However, as the MLE estimator lacks a closed form, coming up with a non-asymptotic confidence interval is a very challenging task. Hence, we consider a simpler estimator.

Let us define an intermediate statistic $\widehat{p}$ as the fraction of times we observed that the underlying Poisson process produced no events, $\widehat{p} = \frac{1}{N}\sum_{n=1}^{N}(1 - o_n)$. Since $\mathbf{P}(o_n = 0) = e^{-\xi w_n}$ we get $\mathbb{E}[\widehat{p}] = \frac{1}{N}\sum_{n=1}^{N} e^{-\xi w_n}$. Motivated by this, we can estimate ξ by the following moment matching method: choose $\widetilde{\xi}$ to be the unique solution of the equation
$$\widehat{p} = \frac{1}{N}\sum_{n=1}^{N} e^{-\widetilde{\xi} w_n}, \qquad (7)$$
and then obtain estimator $\widehat{\xi}$ of ξ by clipping $\widetilde{\xi}$ to range $[\xi_{\min}, \xi_{\max}]$, $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$. The RHS in (7) is monotonically decreasing in $\widetilde{\xi}$, therefore finding the solution of (7) with error γ can be done in $O(\log(1/\gamma))$ time based on binary search. Additionally, if the intervals are

of fixed size, *i.e.*, $\forall n.\, w_n = c$, then $\widetilde{\xi}$ reduces to the maximum likelihood estimator. Such an estimator was proposed by Cho and Garcia-Molina (2003b) and was shown to have good empirical performance. Here, instead of smoothing the estimator, a subsequent clipping of $\widetilde{\xi}$ resolves the issue of its instability for the extreme values of $\widehat{p} = 0$ and $\widehat{p} = 1$ (when the solution to (7) becomes $\widetilde{\xi} = \infty$ and $\widetilde{\xi} = 0$, respectively). In the following lemma, we will show that this estimator $\widehat{\xi}$ is also amenable to non-asymptotic analysis by providing a high probability confidence interval for it.

Lemma 3. *Under the condition of Lemma 2, for any $\delta \in (0, 1)$, and N observations it holds that*

$$\mathbf{P}\left(|\widehat{\xi} - \xi| \geq \left(\frac{1}{N}\sum_{n=1}^{N} w_n e^{-\xi_{\max} w_n}\right)^{-1} \sqrt{\frac{\log \frac{2}{\delta}}{2N}}\right) \leq \delta$$

where $\widehat{\xi} = \max\{\xi_{\min}, \min\{\xi_{\max}, \widetilde{\xi}\}\}$ and $\widetilde{\xi}$ is obtained by solving (7).

Similar analysis can be done for the setting when one has full observability (Upadhyay et al. 2019, See Appendix K). With the following lemma we bound the sensitivity of the expected utility to the accuracy of our parameter estimates $\widehat{\boldsymbol{\xi}}$.

Lemma 4. *For the expected utility $F(\boldsymbol{\rho}; \boldsymbol{\xi})$ defined in* (5)*, let $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho}; \boldsymbol{\xi})$, $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$ and define the suboptimality of $\widehat{\boldsymbol{\rho}}$ as $\operatorname{err}(\widehat{\boldsymbol{\rho}}) := F(\boldsymbol{\rho}^\star; \boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}}; \boldsymbol{\xi})$. Then $\operatorname{err}(\widehat{\boldsymbol{\rho}})$ can be bounded by:*

$$\operatorname{err}(\widehat{\boldsymbol{\rho}}) \leq \sum_i \frac{1}{\widehat{\xi}_i \min\{\widehat{\xi}_i, \xi_i\}} \zeta_i (\widehat{\xi}_i - \xi_i)^2.$$

This lemma gives us hope that if we can learn $\widehat{\boldsymbol{\xi}}$ well enough such that $|\widehat{\xi}_i - \xi_i| \sim \mathcal{O}\left(1/\sqrt{T}\right)$ for all i, then we can obtain sub-linear regret by following the policy $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho}} F(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$. This indeed is possible and, in the next section, we show that an explore-and-commit algorithm can yield $\mathcal{O}(\sqrt{T})$ regret. We would like to also bring to the notice of the reader that a similar result up to constants can be shown for the Harmonic staleness objective proposed by Kolobov et al. (2019a) (see (Upadhyay et al. 2019, Appendix E)) and the accumulating delay objective by Sia, Cho, and Cho (2007) (see (Upadhyay et al. 2019, Appendix F)).

5 Explore-Then-Commit Algorithm

In this section, we will analyse a version of the explore-and-commit (ETC) algorithm for solving the web-crawling problem. The algorithm will first learn $\boldsymbol{\xi}$ by sampling all pages till time τ and then commit to the policy of observing the pages from time τ till T at the rates $\widehat{\boldsymbol{\rho}}$ as given by the Algorithm 2 in (Azar et al. 2018), obtained by passing it the estimated rates $\widehat{\boldsymbol{\xi}}$ instead of the true rates $\boldsymbol{\xi}$.

Revisiting the policy space. The constraint we had used to define the policy space Δ_R was that given any interval of width w, the expected number of refreshes in that interval should not exceed wR, which limited us to the class of Poisson policies. However, an alternative way to impose the constraint is to bound the time-averaged number of requests made per unit of time asymptotically. It can be shown that given our modelling assumptions that request and change processes are memory-less, the policy which maximizes the utility in (2) given a fixed number of observations per page will space them equally. This motivates a policy class $\mathcal{K}_R = \{\boldsymbol{\kappa} = (\kappa_1, \dots, \kappa_m) : \sum_{i=1}^{m} 1/\kappa_i = R\} \subset \Pi$ as the set of deterministic policies which refresh webpage i at regular time intervals of length κ_i. Policies from $\mathcal{K}_R$ allow us to obtain tight confidence intervals for $\widehat{\boldsymbol{\xi}}$ by a straightforward application of Lemma 3. However, the sensitivity of the utility function for this policy space to the quality of the estimated parameters is difficult to bound tightly. In particular, the differential utility function for this class of policies (defined in (3)) is not strongly concave, which is a basic building block of Lemma 4. This precludes performance bounds which are quadratic in the error of estimates $\widehat{\boldsymbol{\xi}}$, which lead to worse bounds on the regret of the ETC algorithm. These reasons are further expounded in the extended version of the paper (Upadhyay et al. 2019, see Appendix I). Nevertheless, we can show that using the uniform-intervals policy $\boldsymbol{\kappa}^{\text{UI}}$ incurs *lower* regret than the uniform-rates policy $\boldsymbol{\rho}^{\text{UR}}$, while still making on average R requests per unit time (Upadhyay et al. 2019, see Appendix H).

Hence, to arrive at regret bounds, we will perform the exploration using *Uniform-interval exploration* policy $\boldsymbol{\kappa}^{\text{UI}} \in \mathcal{K}_R$ which refreshes webpages at regular intervals $\forall i.\, \kappa_i = \frac{m}{R}$, which will allow us to use Lemma 3 to bound the error of the estimated $\widehat{\boldsymbol{\xi}}$ with high probability.

Lemma 5. *For a given $\delta \in (0, 1)$, after following the uniform-interval policy $\boldsymbol{\kappa}^{\text{UI}}$ for time τ, which is assumed to be a multiplicity of m/R, we can claim the following for the error in the estimates $\widehat{\boldsymbol{\xi}}$ produced using the estimator proposed in Lemma 3:*

$$\mathbf{P}\left(\forall i \in [m] \colon |\widehat{\xi}_i - \xi_i| \leq e^{\frac{\xi_{\max} m}{R}} \sqrt{\frac{R \log \frac{2m}{\delta}}{2\tau m}}\right) \geq 1 - \delta.$$

With these lemmas, we can bound the regret suffered by the ETC algorithm using the following Theorem.

Theorem 1. *Let π^{EC} denote the explore-and-commit algorithm which explores using the uniform-interval exploration policy for time τ (assumed to be a multiplicity of $\frac{m}{R}$), estimates $\widehat{\boldsymbol{\xi}}$ using the estimator proposed in* (7)*, and then uses the policy $\widehat{\boldsymbol{\rho}} = \operatorname{argmax}_{\boldsymbol{\rho} \in \Delta_R} F(\boldsymbol{\rho}; \widehat{\boldsymbol{\xi}})$ till time T. Then for a given $\delta \in (0, 1)$, with probability $1 - \delta$, the expected regret of the explore and commit policy π^{EC} is bounded by:*

$$R(T, \pi^{\text{EC}}; \boldsymbol{\xi}) \leq \left(\frac{\tau}{m} + \frac{(T - \tau)}{\tau} \frac{e^{2\frac{\xi_{\max} m}{R}} R \log\left(2m/\delta\right)}{2m^2 \xi_{\min}^2}\right) \sum_{i=1}^{m} \zeta_i.$$

Further, we can choose an exploration horizon $\tau^\star \sim \mathcal{O}(\sqrt{T})$ such that, with probability $1 - \delta$, the expected regret is $\mathcal{O}(\sqrt{T})$.

Proof. Since the utility of any policy is non-negative, we can upper-bound the regret of the algorithm in the exploration phase by the expected utility of the best stationary policy $\boldsymbol{\rho}^\star = \operatorname{argmax}_{\boldsymbol{\rho}\in\Delta_R} F(\boldsymbol{\rho};\boldsymbol{\xi})$, which is $\frac{\tau}{m}\sum_{i=1}^m \zeta_i \frac{\rho_i^\star}{\rho_i^\star+\xi_i} < \frac{\tau}{m}\sum_{i=1}^m \zeta_i$. In the exploitation phase, the regret is given by $\frac{T-\tau}{m}(F(\boldsymbol{\rho}^*,\boldsymbol{\xi}) - F(\widehat{\boldsymbol{\rho}},\boldsymbol{\xi}))$ (see (4)), which we bound using Lemma 4. Hence, we see that (with a slight abuse of notation to allow us to write $R(T,\boldsymbol{\kappa}^{\text{UI}};\boldsymbol{\xi})$ for $\boldsymbol{\kappa}^{\text{UI}} \in \mathcal{K}_R$):

$$\begin{aligned} R(T,\pi^{\text{EC}};\boldsymbol{\xi}) &= R(\tau,\boldsymbol{\kappa}^{\text{UI}};\boldsymbol{\xi}) + R(T-\tau,\widehat{\boldsymbol{\rho}};\boldsymbol{\xi}) \\ &\le \frac{\tau}{m}\sum_{i=1}^m \zeta_i + \frac{(T-\tau)}{m}\sum_{i=1}^m \frac{\zeta_i(\widehat{\xi}_i - \xi_i)^2}{\widehat{\xi}_i \min\{\widehat{\xi}_i,\xi_i\}} \end{aligned} \tag{8}$$

As we are using the estimator from Lemma 3, we have $\widehat{\xi}_i \min\{\widehat{\xi}_i,\xi_i\} \ge \xi_{\text{min}}^2$. Using this and Lemma 5 with (8), we get with probability $1-\delta$:

$$\begin{aligned} &R(T,\pi^{\text{EC}};\boldsymbol{\xi}) \\ &\le \tau \overbrace{\frac{1}{m}\sum_{i=1}^m \zeta_i}^{A} + (T-\tau)\frac{\sum_{i=1}^m \zeta_i}{m\xi_{\text{min}}^2}(\widehat{\xi}_i - \xi_i)^2 \\ &= A\tau + (T-\tau)\frac{\sum_{i=1}^m \zeta_i}{m\xi_{\text{min}}^2}\left(e^{\frac{\xi_{\max} m}{R}}\sqrt{\frac{R\log\left({}^{2m}/_{\delta}\right)}{2m\tau}}\right)^2 \\ &= A\tau + \frac{(T-\tau)}{\tau}\underbrace{\frac{\sum_{i=1}^m \zeta_i}{2m^2\xi_{\text{min}}^2} e^{2\frac{\xi_{\max} m}{R}} R\log\left({}^{2m}/_{\delta}\right)}_{B} \\ &= A\tau + \frac{BT}{\tau} - B \end{aligned} \tag{9}$$

This proves the first claim.

The bound in (9) takes the minimum value when $\tau^\star = \sqrt{\frac{B}{A}}\sqrt{T} = \sqrt{\frac{e^{2\frac{\xi_{\max} m}{R}} R\log\left({}^{2m}/_{\delta}\right)}{2m\xi_{\text{min}}^2}}\sqrt{T}$, giving with probability $1-\delta$, the worst-case regret bound of:

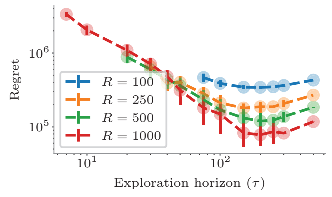
$$R(T,\boldsymbol{\rho}^{\text{EC}};\boldsymbol{\xi}) < 2\sqrt{ABT}$$

This proves the second part of the theorem. $\square$

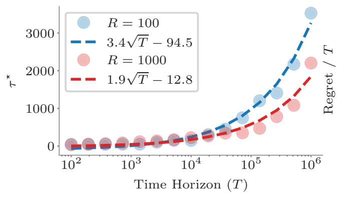
This theorem bounds the expected regret conditioned on the event that the crawler learns $\widehat{\boldsymbol{\xi}}$ such that $\forall i.\, |\widehat{\xi}_i - \xi_i| < \sqrt{\frac{\log {}^{2m}/_{\delta}}{{}^{2\tau R}/_{m}}}$. These kinds of guarantees have been seen in recent works (Rosenski, Shamir, and Szlak 2016; Avner and Mannor 2014).

Note that the proof of the regret bounds can be easily generalised to the full-observation setting (Upadhyay et al. 2019, Appendix K) and for other objective functions (Upadhyay et al. 2019, see Appendix E and F).

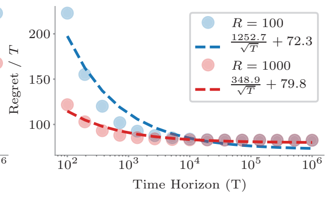
Finally, note that using the doubling trick the regret bound can be made horizon independent at no extra cost. The policy can be de-randomized to either yield a fixed interval policy in $\mathcal{K}_R$ or, to a carousel like policy with similar performance guarantees (Azar et al. 2018, See Algorithm 3). With this upper-bound on the regret of the ETC algorithm, in the next section we explore the empirical performance of the strategy.



(a) Regret / τ (with $T = 10^4$)



(b) $\tau^\star$ / T



(c) Normalized Regret / T

Figure 1: Performance of the ETC algorithm. Panel (a) shows the regret suffered with different exploration horizons (keeping $T = 10^4$ fixed) showing that a minima exists. Panel (b) shows that the optimal value of the horizon scales as $\mathcal{O}(\sqrt{T})$ while panel (c) shows that the time-horizon normalized regret of the ETC algorithm decreases as $\mathcal{O}({}^{1}/_{\sqrt{T}})$.

6 Experimental Evaluation

We start with the analysis of the ETC algorithm, which shows empirically that the bounds that we have proven in Theorem 1 are tight up to constants. Next, we compare the ETC algorithm with phased ε-greedy algorithm and show that phased strategies can out-perform a well-tuned ETC algorithm, if given sufficient number of phases to learn. We leave the detailed analysis of this class of algorithms for later work. An empirical evaluation of the MLE estimator and the moment matching estimator for partial observations, and the associated confidence intervals proposed in Lemma 3 is done in the extended version of the paper (Upadhyay et al. 2019, Appendix J). These show that, for a variety of different parameters, the performance of the MLE estimator and the moment matching estimator is close to each other.

6.1 Evaluation of ETC Algorithm

For the experimental setup, we make user of the MSMACRO dataset (Kolobov et al. 2019b). The dataset was collected over a period of 14 weeks by the production crawler for Bing. The crawler visited webpages approximately once per day. The crawl time and a binary indicator of whether the webpage had changed since the last crawl or not are included in the dataset along with an importance score for the various URLs. We sample 5000 webpages from the dataset while taking care to exclude pages which did not change at all or which changed upon every crawl so as to not constraint the estimates of the change rates artificially to $\xi_{\min}$ or $\xi_{\max}$. We calculate their rate of change ($\boldsymbol{\xi}$) using the MLE estimator (6). The corresponding importance values $\boldsymbol{\zeta}$ are also sampled from the importance value employed by the production web-crawler. We set $\xi_{\min} = 10^{-9}$ and $\xi_{\max} = 25$. The experiments simulate the change times $((x_{i,n})_{n=1}^{\infty})_{i\in[m]}$ for webpages 50 times with different random seeds and report quantities with standard deviation error bars, unless otherwise stated.

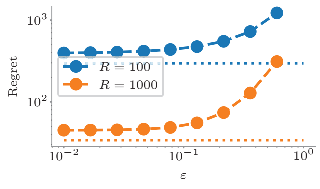
We first empirically determine the regret for different exploration horizon τ and bandwidth parameter R. To this end, we run a grid search for different values of τ (starting from the minimum time required to sample each webpage at least once), simulate the exploration phase using uniform-interval policy $\boldsymbol{\kappa}^{\text{UI}}$ and simulated change times to determine the parameters $\widehat{\boldsymbol{\xi}}$ and the regret suffered during the exploration phase. We calculate $\widehat{\boldsymbol{\rho}}$ using Algorithm 2 of Azar et al. (2018), similarly calculate $\boldsymbol{\rho}^{\star}$ using the true parameters $\boldsymbol{\xi}$, calculate their respective utility after the commit phase from τ till time horizon $T = 10^4$ using (4), and use it to determine the total regret suffered. We report the mean regret with the error bars indicating the standard deviations in Figure 1a. We see that there indeed is an optimal exploration horizon, as expected, and the value of both the horizon and the regret accumulated depends on R. We explore the relationship between the optimal exploration horizon $\tau^{\star}$ and the time horizon T next by varying T from 10^2 to 10^6 and calculating the optimal horizon $\tau^{\star}$ (using ternary search to minimize the empirical mean of the utility) for $R \in \{10^2, 10^3\}$; plots for other values of R are qualitatively similar. Figure 1b shows that the optimal exploration horizon $\tau^{\star}$ scales as $\mathcal{O}(\sqrt{T})$.

Finally, we plot the time-horizon normalized regret suffered by π^{EC} when using the optimal exploration horizon $\tau^{\star}$ in Figure 1c. We see that the normalized regret decreases as $\frac{1}{\sqrt{T}}$, as postulated by Theorem 1. Plots for different values of $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ are qualitatively similar. It can also be seen in all plots that if the allocated bandwidth R is high, then the regret suffered is lower but the dependence of the optimal exploration threshold $\tau^{\star}$ on the available bandwidth is non-trivial: in Figure 1b, we see that the $\tau^{\star}_{R=100} < \tau^{\star}_{R=1000}$ if $T < 10^3$ and $\tau^{\star}_{R=100} > \tau^{\star}_{R=1000}$ if $T > 10^4$.

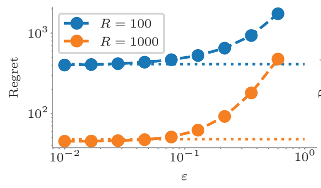
It is noteworthy that Figures 1b and 1c suggest that the bounds we have proven in Theorem 1 are tight up to constants.

6.2 Phased ε-greedy Algorithm

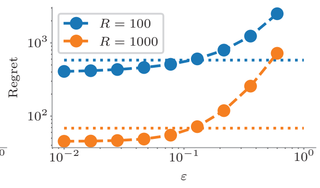
In Section 5, we have shown guarantees on the performance of explore-then-commit algorithm which is the simplest form of policies which adheres to the properties we mention in Section 3. However, it is not difficult to imagine other strategies which conform to the same recommendations. For example, consider a phased ε-greedy algorithm which runs with $\boldsymbol{\rho}^{\text{UR}}$ for duration given in Lemma 1, estimates $\widehat{\boldsymbol{\xi}}$, calculates $\widehat{\boldsymbol{\rho}}$, and then follows the policy $\boldsymbol{\rho}^{\varepsilon}$, where $\rho^{\varepsilon}_i = (1-\varepsilon)\widehat{\rho}_i + \varepsilon\frac{R}{m}$, and then starts another phase, improving its policy with improving estimates of $\widehat{\boldsymbol{\xi}}$. Since $\forall i \in [m].\, \rho^{\varepsilon}_i > \varepsilon\frac{R}{m}$, the policy



(a) $T = 10^{3.67}$ / 3 phases



(b) $T = 10^{3.83}$ / 6 phases



(c) $T = 10^4$ / 9 phases

Figure 2: Performance of the phased ε greedy algorithm. The dotted lines show the regret of the ETC algorithm, with optimal exploration horizon, same bandwidth R and time horizon T. While the ETC algorithm performs well when the number of phases is small, with increasing number of phases, the phased ε-greedy algorithm is able to obtain lower regret than ETC.

will continue exploring all the webpages, ensuring eventual consistency in the estimates of $\widehat{\boldsymbol{\xi}}$.

We performed simulations with the $\boldsymbol{\rho}^{\varepsilon}$ algorithm and found that though it performed worse than ETC for small time horizons (Figures 2a and 2b), it performed better when given sufficient number of phases (see Figure 2c). Exploring the regret bounds of such policies is part of our future work.

7 Conclusion

In this paper, we have taken the first step towards solving the problem of learning changing rates of web-pages while solving the web-crawling problem, while also providing a guiding framework for analysing online learning problems where the agent's policy can be described using a Poisson process. We have shown that the learning algorithms should be *phased* and there are restrictions on how much they can change the policy from step to step while keeping learning feasible. Further, by bounding the performance of a Poisson policy using a deterministic policy, we have proved that a simple explore-and-commit policy has $\mathcal{O}(\sqrt{T})$ regret under mild assumptions about the parameters.

This leaves several interesting avenues of future work open. Though we have proved a theoretical upper bound on regret of $\mathcal{O}(\sqrt{T})$ and have empirically seen that bound is tight up to constants for the explore-and-commit algorithm, it is not clear whether this bound is tight for the class of all *phased* strategies. We will explore the class of such strategies in a planned extension. Lastly, we believe there are rich connections worth exploring between this work and the recent work on the Recharging Bandits paradigm (Immorlica and Kleinberg 2018). We have taken some initial steps in the direction in the extended version of the paper (Upadhyay et al. 2019, See Appendix I and H).

Acknowledgements

W. Kotłowski was supported by the Polish National Science Centre under grant No. 2016/22/E/ST6/00299.

References

Avner, O., and Mannor, S. 2014. Concurrent bandits and cognitive radio networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 66–81. Springer.

Azar, Y.; Horvitz, E.; Lubetzky, E.; Peres, Y.; and Shahaf, D. 2018. Tractable near-optimal policies for crawling. *Proceedings of the National Academy of Sciences* 115(32):8099–8103.

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

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

Du, N.; Dai, H.; Trivedi, R.; Upadhyay, U.; Gomez-Rodriguez, M.; and Song, L. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*.

Duchi, J.; Shalev-Shwartz, S.; Singer, Y.; and Chandra, T. 2008. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, 272–279. ACM.

Farajtabar, M. 2018. *Point Process Modeling and Optimization of Social Networks*. Ph.D. Dissertation, Georgia Institute of Technology.

Immorlica, N., and Kleinberg, R. D. 2018. Recharging bandits. *IEEE 59th Annual Symposium on Foundations of Computer Science*.

Karimi, M. R.; Tavakoli, E.; Farajtabar, M.; Song, L.; and Gomez Rodriguez, M. 2016. Smart broadcasting: Do you want to be seen? In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 1635–1644. ACM.

Kingman, J. 1993. *Poisson Processes*. Oxford Science Publications.

Kolobov, A.; Peres, Y.; Lu, C.; and Horvitz, E. 2019a. Staying up to date with online content changes using reinforcement learning for scheduling.

Kolobov, A.; Peres, Y.; Lubetzky, E.; and Horvitz, E. 2019b. Optimal freshness crawl under politeness constraints. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 495–504. ACM.

Li, X.; Cline, D. B.; and Loguinov, D. 2017. Temporal update dynamics under blind sampling. *IEEE/ACM Transactions on Networking (TON)* 25(1):363–376.

Neu, G.; Antos, A.; György, A.; and Szepesvári, C. 2010. Online markov decision processes under bandit feedback. In *Advances in Neural Information Processing Systems*, 1804–1812.

Rosenski, J.; Shamir, O.; and Szlak, L. 2016. Multi-player bandits–a musical chairs approach. In *International Conference on Machine Learning*, 155–163.

Sia, K. C.; Cho, J.; and Cho, H.-K. 2007. Efficient monitoring algorithm for fast news alerts. *IEEE Transactions on Knowledge and Data Engineering* 19(7):950–961.

Tabibian, B.; Upadhyay, U.; De, A.; Zarezade, A.; Schölkopf, B.; and Gomez-Rodriguez, M. 2019. Enhancing human learning via spaced repetition optimization. *Proceedings of the National Academy of Sciences* 201815156.

Upadhyay, U.; Busa-Fekete, R.; Kotlowski, W.; Pal, D.; and Szorenyi, B. 2019. Learning to crawl. *arXiv preprint arXiv:1905.12781*.

Upadhyay, U.; De, A.; and Gomez-Rodriguez, M. 2018. Deep reinforcement learning of marked temporal point processes. In *Advances in Neural Information Processing Systems*.

Wang, Y.; Williams, G.; Theodorou, E.; and Song, L. 2017. Variational policy for guiding point processes. In *ICML*.

Zarezade, A.; De, A.; Upadhyay, U.; Rabiee, H. R.; and Gomez-Rodriguez, M. 2017. Steering social activity: A stochastic optimal control point of view. *Journal of Machine Learning Research* 18:205–1.