Static & Dynamic Appointment Scheduling with Stochastic Gradient Descent

Gary Cheng[1] and Kabir Chandrasekher[2] and Jean Walrand[3]

Abstract— **This paper considers the optimization of static and dynamic appointments for a sequence of customers with random processing times served by a single agent. In the static case, the problem is to find the appointment times for the customers that minimize the expected weighted sum of idle time of the agent, waiting time of the customers, and overtime of the agent. The dynamic formulation is original. It limits cascading delays by using warning times, which are defined relative to the start of a previous job, to tell customers when to arrive for their job. The problem in this formulation is to find the warning times which minimize the aforementioned objective. We outline an algorithmic framework based on infinitesimal perturbation analysis (IPA) and stochastic gradient descent (SGD) that can be used to solve both problems. For the static case, we show that the SGD method converges to a global minimizer with no assumptions on the joint distribution of the processing durations. For the dynamic case, we construct a warning time schedule which provably outperforms the static scheme under certain conditions. We give empirical evidence of the efficacy of both approaches using both synthetic data as well as data collected from a year of elective surgeries at a local hospital. Our results suggest the validity of the dynamic formulation as a more cost-efficient solution than the static formulation to the appointment scheduling problem.**

I. Introduction

In a vast variety of situations, customers make appointments with an agent performing a specific service. The inherent tension in scheduling appointments arises from the following incentives: the agent would like to pack the schedule in order to minimize the time it spends idle, whereas the customer would like to spend as little time as possible waiting for the agent. In the ideal solution for the agent, customers may face significant delays if previous jobs run longer than expected. By contrast, in the ideal solution for the customer, the appointments will be too spaced out and the agent will face large amounts of idle time. Here, we consider the scenario with one agent and n jobs in a pre-specified order and aim to find a reasonable trade-off between these two incentives.

A. Our Contribution

We develop a sample-based stochastic gradient descent (SGD) algorithm that adjusts the appointment times based on sampled task durations. We distinguish two situations:

1) **Static Appointments:** All customer-arrival times are fixed prior to the start of the first jobs and remain fixed throughout the day.
2) **Dynamic Appointments:** Customer-arrival times are *not* fixed before-hand and can vary as the jobs progress.

Our main contribution is the development of an algorithmic framework that can be used as a solution to both of these situations. In the case of static appointments, we show that our algorithm converges to the optimal solution. Our dynamic formulation is original and is an attempt to mitigate the issue of cascading delays that occur when the long processing of a single job delays a series of subsequent jobs. We construct a dynamic appointment schedule which, under certain conditions, provably performs at least as well as the optimal static appointment schedule. We note that in our formulation the constraints of a single agent and jobs of pre-specified order is essential and relaxing these requirements requires non-trivial extension. We believe, however, based on the simplicity of our approach and the empirical evidence we provide that our framework could prove useful in the most general form of this problem.

B. Related Work

The static appointment scheduling problem has been extensively studied for decades. In its most general form, the problem is difficult to analyze. To deal with this, many previous works add constraints in order to add more exploitable structure to the problem. One common way this is done is by restricting the distribution of service times. For example, Pegden and Rosenshine [7] considered the problem of minimizing customer waiting time and overall usage time in a continuous time, exponential service time setting. Wang [10] worked on the same problem with the assumption that service times were from a phase-type distribution e.g., exponential, coxian; with these assumptions, he was able to show that an optimal appointment can be found by solving a set of nonlinear equations. Bosch, Dietz, and Simeoni [9] consider customer waiting time and overtime in the problem formulation and introduced an optimal appointment schedule algorithm with iid Erlang service times. Constraining service times to be discrete was explored by Begen and Queyranne [2]. They proved that if the durations only take on integer values and the objective is L-convex, then the optimal solution can be found in polynomial time. However, constraining the service time distribution to be of a certain type is quite limiting and unrealistic in certain scenarios.

[1] Gary Cheng is at the Department of EECS, UC Berkeley, Berkeley, CA 94709, USA garycheng@berkeley.edu
[2] Kabir Chandrasekher is at the Department of Electrical Engineering, Stanford University, Palo Alto, CA, USA kabirc@stanford.edu
[3] Jean Walrand is at the Department of EECS, UC Berkeley, Berkeley, CA 94709, USA walrand@berkeley.edu

Many other papers outline approaches that use general purpose stochastic optimization methods. Denton and Gupta [4] approached this problem using a stochastic linear program and developed distribution-independent upper bounds. They then use a sequential bounding algorithm to find ε-optimal solutions for a given static appointment problem. Zhang and Xie [13] used a stochastic gradient descent method to look at the multi-agent, multi-customer problem. They proved many nice features of their objective function, but were unable to show optimal convergence of their algorithm. Bai, Storer, and Tonkay [1] also used an SGD approach to empirically analyze cost savings in a more specific surgery operating room setting where added constraints were placed on hospital room capacity.

The work mentioned thus far focuses on determining a fixed schedule at the beginning of the day. There has been some work on dynamic scheduling of customers which is related to our dynamic appointment formulation. Wang [11] studies the problem of scheduling the $(n+1)$st customer after the schedule of the first n has been set and cannot be changed, where all service times are exponential. The scheduler tries to find which jobs the $(n+1)$st customer should be scheduled in between. One of the problems Zacharias and Pinedo [12] considers is the online problem of scheduling appointments while a set number of jobs scheduled statically beforehand are being executed. The scheduler optimizes to find the best appointment time of the new customer given that some part of the schedule has already been completed. It should be noted that these dynamic scheduling methods still have a similar structure to the static appointment counterpart: the appointment times are defined with respect to the beginning of the day. Our dynamic appointment formulation attempts to decouple this dependency by instead defining the appointment of one job to be relative to the start time of a previous job.

C. Paper Organization

The rest of the paper is organized in the following manner. In section II, we formally define the static appointment time problem, describe our SGD algorithm, and prove that the aforementioned algorithm converges to the optimal solution. In section III, we carefully describe the dynamic extension of the appointment scheduling problem, prove some performance guarantees of the formulation, and outline our SGD algorithm for this case. Finally, we provide extensive empirical evidence corroborating our theoretical results as well as demonstrating the utility of these algorithms in section IV.

II. Static Appointment Scheduling

We begin by formally defining the static appointment problem. An agent has an ordered set of n jobs that need to be completed in the order given; one job must be finished before the next job can begin. For $m = 1, \dots, n$, the duration of job m is a continuous random variable X_m. We make no assumption on the dependence of these random variables. Customers are told their appointment time $\mathbf{a} = (a_1, \dots, a_n)$ before any of the jobs begin; appointment times cannot be changed once they have been given out. Customers will arrive exactly at their appointment time, and the first customer will always start service at time 0. Thus, we can assume that $a_1 = 0$. Every customer must wait until all previous jobs are completed before being served. If a job completes and the next customer has not arrived yet, the agent must remain idle until the next customer arrives. The agent begins to incur overtime past a given time d. Customer waiting times, agent idle time, and agent overtime are penalized at a cost rate of α, β, and γ, respectively, to give us the wait cost, idle cost, and overtime cost; these parameters are all required to be nonnegative. The problem then is to find the appointment times $\mathbf{a} = (a_1, \dots, a_m)$ that minimize the expected sum of idle cost, wait cost, and overtime cost.

For our problem, we assume there is an oracle that provides iid samples from the joint distribution of job durations $\mathbf{X} = (X_1, \dots, X_n)$. The start time S_m and the end time Y_m of job m for $m = 1, \dots, n$ are formally defined below. Note that we do not write explicitly the dependency of S_m and Y_m on $\mathbf{a}$ and $\mathbf{X}$ for ease of notation.

$$S_1 := 0 \qquad\qquad S_m := \max(Y_{m-1}, a_m)$$
$$Y_m := S_m + X_m$$

The wait cost C_W, idle cost C_I, overtime cost C_O, and total cost C are formally defined as follows. We omit the dependency of C_W, C_I, C_O, and C on $\mathbf{a}$ and $\mathbf{X}$ for ease of notation. We define $(z)^+$ to mean $\max(z, 0)$.

$$C_W := \alpha \sum_{m=2}^{n} (Y_{m-1} - a_m)^+ \qquad C_I := \beta \sum_{m=2}^{n} (a_m - Y_{m-1})^+$$
$$C_O := \gamma (Y_n - d)^+ \qquad\qquad C := C_W + C_I + C_O$$

Our objective function is the expected total cost: $\mathbb{E}_{\mathbf{X}}[C]$.

A. SGD for Static Appointment Scheduling

The static appointment SGD algorithm (**SA-SGD**) is as follows. Assume that the vector of appointment times is $\mathbf{a}^t$ at step t of the algorithm. One draws a new sample $\mathbf{X}^t$ of the random vector $\mathbf{X}$. One then calculates the sample gradient $\nabla_{\mathbf{a}} C(\mathbf{a}^t, \mathbf{X}^t)$ with respect to $\mathbf{a}$ and takes a step in the opposite direction of the gradient. That is,

$$\mathbf{a}^{t+1} = \mathbf{a}^t - \eta_t \nabla_{\mathbf{a}} C(\mathbf{a}^t, \mathbf{X}^t)$$

$\eta_t > 0$ is the step size, and we set it to be $\eta_t = \mathcal{O}(1/\sqrt{t})$. This choice is justified in the next subsection. The sample gradient $\nabla_{\mathbf{a}} C(\mathbf{a}^t, \mathbf{X}^t)$ is calculated using Infinitesimal Perturbation Analysis (IPA) [5]. This is an analysis of how much cost would be added if a_m were slightly increased.

The IPA procedure is illustrated in Figure 1. In particular, this figure shows the processing of six jobs with their appointment times $\{a_1 = 0, a_2, \dots, a_6\}$. We detail some

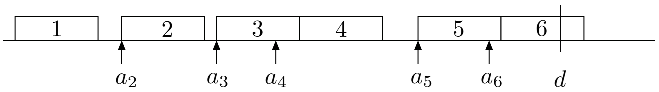

Fig. 1. A realization of the static appointment service process

cases that outline how the sample gradient is calculated under different circumstances. **Case 1**: increasing a_2 by $\delta \ll 1$ increases the idle time before job 2 but decreases the idle time after job 2 and does not affect waiting times nor the overtime. Thus, $\partial C/\partial a_2 = 0$. **Case 2**: increasing a_3 by $\delta \ll 1$ increases the idle time before job 3 and decreases the idle time after job 4. It also increases the waiting time of job 4. Thus, $\partial C/\partial a_3 = \alpha$. **Case 3**: increasing a_4 and a_6 by $\delta \ll 1$ only decrease the waiting time for job 4 and 6 respectively, so $\partial C/\partial a_4 = \partial C/\partial a_6 = -\alpha$. **Case 4:** Using similar observations one finds that $\partial C/\partial a_5 = \alpha + \beta + \gamma$.

We introduce some notation to calculate the sample gradient in the general case. Jobs m that incur no wait cost are called *heads of chains*. From the examples outlined above, it is clear that only heads of chains have the potential to affect the wait times of jobs after them. Heads of chains will have an indicator $H_m = 1$. The jobs (not including job m) that are affected by perturbing a_m are said to be *part of job m's chain*. In particular, a_m affects the start time of job i if $i > m$, job m incurs no wait cost, and jobs $m+1, \ldots, i$ all incur some waiting cost. P_m will count the number of jobs (not including job m) affected by a perturbation of a_m i.e., the number of jobs in job m's chain. Only perturbing the last chain affects the idle cost, so the indicator L_m will equal 1 for jobs in the last chain. The indicator O will equal 1 if the end time of the last job is greater than d. Using these indicators, we can observe that the gradient in general is:

$$\frac{\partial C}{\partial a_m} = H_m(\alpha P_m + \beta L_m + \gamma L_m O) - (1 - H_m)\alpha$$

As this calculation shows, the sample cost is piecewise linear in the parameters $\mathbf{a}$ with coefficients that belong to a finite set. It is therefore differentiable almost everywhere. At the points where it is not, we define $\nabla_{\mathbf{a}} C(\mathbf{a}, \mathbf{X})$ to be the right derivative. By the bounded convergence theorem, $\nabla_{\mathbf{a}} \mathbb{E}[C(\mathbf{a}, \mathbf{X})] = \mathbb{E}[\nabla_{\mathbf{a}} C(\mathbf{a}, \mathbf{X})]$. This shows that the objective is differentiable, and it also justifies our use of the sample gradient in the SGD algorithm.

B. Convergence to the Optimum for Static Scheduling

To prove the convergence of SGD, we begin by proving the cost function is convex.

Lemma II.1. *C_O, C_W, C_I, C, and $\mathbb{E}[C]$ are all convex in* $\mathbf{a}$

Proof. We first show Y_m is convex in $\mathbf{a}$ for all m by induction; the others follow directly. $Y_1 = X_1$ is a constant function, which is trivially convex. Assume that Y_{m-1} is convex, then $Y_m = \max(Y_{m-1}, a_m) + X_m$ is also convex.

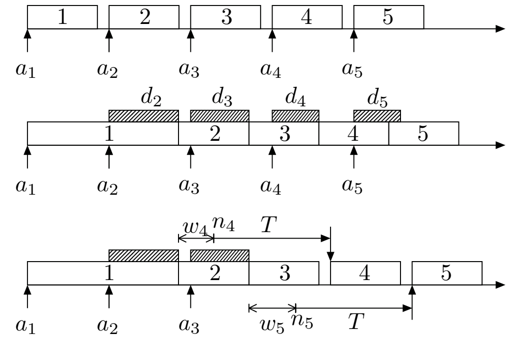

Fig. 2. Expected realization (top); Cascade of delays with static appointment times (middle); Limitation of cascade with notification times (bottom). a_2 and a_3 in the bottom figure indicate that w_2 and w_3 respectively were used to simulate static appointments (see III)

$C_O = \gamma \max(Y_n - d, 0)$ is convex because Y_n is convex in $\mathbf{a}$ and the function $\max(y - d, 0)$ is convex in y. $C_W = \alpha \sum_{m=2}^{n} \max(Y_{m-1} - a_m, 0)$ is a sum of convex functions in $\mathbf{a}$ and is therefore convex. $C_I = \beta\,(Y_n - \sum_{i=1}^{n} X_m)$ is an equivalent way of writing C_I because the amount of idle time is equal to the last completion time minus the total time spent processing jobs. From this expression, it is clear that C_I is convex. C is convex because it is the sum of convex functions. We also know that $\mathbb{E}[C]$, which is a weighted average over convex sample cost functions, is convex. □

Noting that D can be taken as 24 hours (the amount of time in a day) and the partial derivative with respect to a_m is bounded by the value $n\alpha + \beta + \gamma$, the following statement follows immediately from Shamir and Zhang[8] Theorem 2 and Lemma II.1. The next theorem proves that static SGD algorithm converges to appointment times that minimize the expected cost.

Theorem 1. *For $F := \mathbb{E}_X[C]$ convex and for constants D, $G^2 = (n\alpha + \beta + \gamma)\sqrt{n}$, it holds that $\mathbb{E}[\|\nabla_{\mathbf{a}} C(\mathbf{a}^t, \mathbf{X}^t)\|_2] \leq G^2$ for all $\mathbf{a}$, and $\sup_{a,a'} \|a - a'\|_2 \leq D$. Consider SGD with step sizes $\eta_t = c/\sqrt{t}$ where $c > 0$ is a constant. Then for any $t > 1$, it holds that*

$$\mathbb{E}[F(a_t) - F(a^*)] \leq \left(\frac{D^2}{c} + cG^2\right) \frac{2 + \log(t)}{\sqrt{t}}$$

Note that while our analysis focuses on the case where there is one α wait cost penalty for all customers, there is no reason why we could not have different wait time penalties α_m for each individual customer. The optimality and convergence properties of the SGD also hold in this case.

III. Dynamic Appointment Scheduling

A glaring issue that the static appointment scheme cannot address is cascading delays. Figure 2 illustrates this problem. The expected schedule of the customers comprise the top part of the figure. The middle part of the figure shows what happens if the first job takes longer than expected: it pushes back the execution of the subsequent jobs and causes the delays $d_2, \ldots, d_5$ for those customers. The bottom part of the figure illustrates the notification times scheme. Instead of giving a fixed appointment time a_4 to customer 4, one notifies them at time n_4 that he should come at time n_4+T. Here, *lead time* T is the time customers have to arrive for their job. We assume that it is unrealistic to ask customers to arrive in a variable amount of time, so we fix T as a constant. One expects that if T is not very large compared to the processing times, dynamic appointments will reduce the cost. The *notification time* n_4 is anchored at some time during the execution of job 2. Specifically, we set it to be $n_4 = \min(S_2+w_4, Y_2)$ where S_2 is the start time of job 2 and Y_2 is its completion time. w_4 is a parameter to be optimized, which we will call customer 4's *warning time*. As the figure indicates, if the execution of job 2 is postponed because of prior delays the appointment time of customer 4 slides automatically so that his waiting time does not increase. The situation is similar for customer 5. The job during which the notification of job j is anchored to called its *K-mapping*, denoted by $K(j)$. For instance, in Figure 2, $K(4) = 2$.

We formalize the notification time problem as follows. An agent is given a set of n jobs that need to be completed in the order given. Each customer j will be notified during the execution of a previous job $1 \leq K(j) < j$. The first job will start at time 0 ($a_1 = 0$) and will have the K-mapping: $K(1) = -1$. All other jobs $j > 1$ after must have a mapping of $K(j) \geq 1$, and the notification time of those jobs are $n_j = \min(S_{K(j)}+w_j, Y_{K(j)})$. Thus, the policy is parameterized by $K(j)$ for $j = 2, \ldots, n$ and $\mathbf{w} = (w_2, \ldots, w_n)$. Jobs j where $K(j) = 1$ can have a $w_j \geq -T$; a negative warning time in this context simulates a static appointment at time $w_j + T$ (e.g., in Figure 2, $K(2) = K(3) = 1$ is used to simulate the "static" appointment times a_2 and a_3). Jobs j where $K(j) > 1$ must have $w_j \geq 0$; this is because a negative warning time when $K(j) > 1$ requires knowing $S_{K(j)}$ before it is even realized. The arrival time of customer j is then $R_j := n_j+T$. Observe that if $j < i$, then it must be that $K(j) \leq K(i)$. Formally:

$$
\begin{aligned}
S_1 &:= R_1 := 0 & R_j &:= S_{K(j)} + \min(X_{K(j)}, w_j) + T \\
S_j &:= \max(R_j, Y_{j-1}) & Y_j &:= S_j + X_j
\end{aligned}
$$

The wait cost, idle cost, overtime cost, and total cost are the following. For the sake of brevity, we omit the dependency of the costs on $\mathbf{w}$ and $\mathbf{X}$.

$$
\begin{aligned}
C_W &:= \alpha \sum_{j=2}^{n} (S_j - R_j)^+ & C_I &:= \beta \sum_{j=2}^{n} (S_j - Y_{j-1})^+ \\
C_O &:= \gamma (Y_n - d)^+ & C &:= C_W + C_I + C_O
\end{aligned}
$$

Unfortunately, in contrast with the static case, the cost is generally not convex in $\mathbf{w}$ even when the K vector is fixed.[1]

[1] Consider two jobs, where $\Pr(X_1 = 2) = 1$, $T = 1$, $\alpha = \beta > \varepsilon$, and $\gamma = 0$. The slope of the cost function with respect to w_2 increases from α to β and then subsequently decreases to ε.

A. Properties

We can observe that for very long lead times (T greater than the duration of the entire schedule) the dynamic scheme trivially performs just as well as the static scheme by setting $K(j) = 1$ for all $j > 1$ and then using w_j to simulate the static appointment scheme. To explore how a "short" lead time performs, we introduce some notation:

Definition III.1 (Simple K-mapping)**.** $K(1) = -1$ *and for all* $j > 1$, $K(j) = j - 1$.

We now construct the following lemma (justified by the newsvendor result in [6]) which will be useful in proving a performance guarantee:

Lemma III.2 (Optimal 2-job static appt.)**.** *Consider a schedule which only contains the* $j-1$*th and* j*th jobs and* $\gamma = 0$*.* $\hat{a}_j$*, which satisfies the condition* $\Pr(X_{j-1} \leq \hat{a}_j) = \alpha/(\alpha + \beta)$*, is the optimal static appointment time.*

Note that $\hat{a}_j$ can be quickly constructed either by using the static SGD algorithm from this paper or the classic newsvendor results mentioned above.

Using Lemma III.2, we can quickly form an estimate of the optimal warning time: $\hat{w}_2 := \hat{a}_2 - T$ and $\hat{w}_j := (\hat{a}_j - T)^+$ for $j > 2$. We now give a performance guarantee for the dynamic appointment formulation.

Theorem 2. *Let* OPT_{static}*,* $\hat{OPT}_{simple}$*,* OPT_{simple}*, and* $OPT_{dynamic}$ *be the average cost incurred by the optimal static schedule, the estimate of the optimal simple K-mapping schedule using* $\hat{w}$*, the optimal simple K-mapping schedule, and the optimal dynamic appointment schedule respectively. Assume* $\gamma = 0$*, independent job durations, and* $\forall j > 2,\ T \leq \hat{a}_j$*, then:*

$$
OPT_{static} \geq \hat{OPT}_{simple} \geq OPT_{simple} \geq OPT_{dynamic}
$$

Proof. Recall that the static appointment cost function:

$$
\begin{aligned}
&OPT_{static} = \min_{\mathbf{a}} \sum_{m=2}^{n} \mathbb{E}_{\mathbf{X}}[\alpha(S_{m-1} + X_{m-1} - a_m)^+ \\
&\quad + \beta(a_m - S_{m-1} - X_{m-1})^+] \\
&\geq \min_{\mathbf{a}} \sum_{m=2}^{n} \mathbb{E}_{X_{m-1}}[\alpha(\mathbb{E}_{S_{m-1}}[S_{m-1}] + X_{m-1} - a_m)^+ \\
&\quad + \beta(a_m - \mathbb{E}_{S_{m-1}}[S_{m-1}] - X_{m-1})^+] \\
&= \min_{\mathbf{b}} \sum_{m=2}^{n} \mathbb{E}_{X_{m-1}}[\alpha(X_{m-1} - b_m)^+ + \beta(b_m - X_{m-1})^+] \\
&\quad \text{s.t. } b_m = a_m - \mathbb{E}_{S_{m-1}}[S_{m-1}] \\
&\geq \min_{\mathbf{b}} \sum_{m=2}^{n} \mathbb{E}[\alpha(X_{m-1} - b_m)^+ + \beta(b_m - X_{m-1})^+] \\
&= \min_{\mathbf{w}} \sum_{m=2}^{n} \mathbb{E}[\alpha(X_{m-1} - w_m - T)^+ \\
&\quad + \beta(w_m + T - X_{m-1})^+] \\
&= OPT_{temp}
\end{aligned}
$$

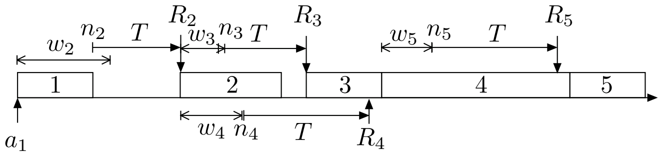

Fig. 3. A realization of the notification time service process. Customer 2 is notified after the end of job 1; therefore, moving back w_2 does not do anything. Moving back w_3 would increase the wait time of customer 4, but customer 5's waiting time is unaffected.

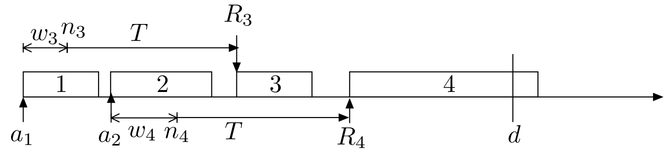

Fig. 4. Another realization of the notification time service process. Moving w_3 does not affect the total idle time because job 3 is not apart of the last chain, but moving back w_4 would increase total idle time. Overtime works similarly provided that the end time of the last job ends after d. Perturbing the last chain (i.e., job 4) is the only way that over time can increase. a_2 indicates that w_2 was used to simulate a static appointment (see III)

The first inequality is an application of Jensen's inequality; it follows since the X_{m-1} is independent of S_{m-1}. The second inequality follows as the optimal unconstrained cost must be less than or equal to its constrained counterpart. Notice that in the line following, we know that $b_m^* = \hat{a}_m$ due to Lemma III.2; thus, we are able to re-parameterize $\mathbf{b}$ as $w_m = b_m - T$ since $w_m \geq 0$ for $m > 2$ at optimum (recall w_2 is allowed to be negative). Denote $OPT_{temp} := \min_{\mathbf{w}} \sum_{m=2}^n \mathbb{E}[\alpha(X_{m-1} - w_m - T)^+ + \beta(w_m + T - X_{m-1})^+]$. In particular, we can observe that using $\hat{w}_m = b_m^* - T$ achieves OPT_{temp}. We use this fact to prove that $OPT_{static} \geq O\hat{P}T_{simple}$ using some algebraic manipulation:

$$\begin{aligned}
OPT_{temp} \geq \sum_{m=2}^{n} &\mathbb{E}[\alpha(X_{m-1} - (\min(\hat{w}_m, X_{m-1}) + T))^+ \\
&+ \beta(\min(\hat{w}_m, X_{m-1}) + T - X_{m-1})^+] \\
= O\hat{P}T_{simple}&
\end{aligned}$$

Note that this inequality is true for any valid w_m, not just the optimal $\hat{w}_m$. Note that $O\hat{P}T_{simple} \geq OPT_{simple}$ and $OPT_{simple} \geq OPT_{dynamic}$ are true by construction.

□

B. SGD for Dynamic Appointment Scheduling

Like the static variant, the dynamic algorithm uses SGD for optimizing the warning time parameters $\mathbf{w}$ that determine the notification times. The IPA calculation of the gradient is slightly different from the static case. Although the non-convexity of the problem means that the SGD scheme with any fixed K vector is no longer guaranteed to converge to a global minimum, SGD does converge to a critical point [3].

Let us first look at Figure 3 and examine some cases that outline how the sample gradient is calculated. **Case 1**: observe that w_2 takes place after the end of job 1. Consequently, increasing w_2 by $\delta \ll 1$ does not affect any outcome. Therefore, $\partial C/\partial w_2 = 0$. **Case 2**: if we increase w_3 by $\delta \ll 1$, then the idle time between jobs 2 and 3 increases. Customer 4 also has to wait longer than before, but customer 5 is unaffected because his notification time occurs during job 4, which has not happened yet. Therefore, $\partial C/\partial w_3 = \alpha + \beta$. Note that because customer 3 does not incur any wait cost, perturbing w_3 does change the start time of job 3 and thereby increases the waiting costs of the customers after it. **Case 3**: Increasing w_4 and w_5 decreases the wait time of the 4th and 5th customer respectively, so that $\partial C/\partial w_4 = \partial C/\partial w_5 = -\alpha$. Note that because customers 4 and 5 incur some waiting cost, increasing w_4 and w_5 can only decrease the waiting cost of those customers. Now, looking at Figure 4, **Case 4**: we see that perturbing w_3 does not affect the idle costs or overtime costs: $\partial C/\partial w_3 = 0$. **Case 5:** perturbing w_4 affects both idle and overtime cost: $\partial C/\partial w_4 = \beta + \gamma$. We can observe that the idle cost changes only if perturbing w_j changes the end time of the last job of the day.

To define the gradient in general, we define a new indicator random variable $Z_j = 1\{w_j \geq X_{K(j)}\}$; this indicator is useful for handling situations outlined in Case 1. As before, we use $H_j = 1$ if job j incurs no wait cost (the head of a chain) and P_j as the number of jobs (not including job j) affected by perturbing w_j (number of other jobs in the job j's chain). In particular, w_j affects the start time of job i if $Z_j = 0$, $i > j$, job j incurs no wait cost, jobs $j+1, \dots, i$ all incur some waiting cost and the ith customer has been warned before the start of the jth job, $K(i) < j$. Let L_j be the indicator that changing w_j affects the start time of the last job. Notice that perturbing w_j affects the overtime cost only if $L_j = 1$ and the last job ends after d. We will again use O as the indicator that the last job goes over d. In all, the sample gradient is:

$$\frac{\partial C}{\partial w_j} = (1 - Z_j)[H_j(\alpha P_j + \beta L_j + \gamma L_j O) - (1 - H_j)\alpha]$$

As a heuristic, we generally set $\partial C/\partial w_j = \varepsilon$ when $Z_j = 1$ for some $\varepsilon > 0$ to prevent the SGD from getting stuck in flat spots. A positive ε can also be thought of as a penalty for overestimating the duration of the $K(j)$th job.

There are three variants of the SGD algorithm: brute force K-mapping optimization (**BK-NT-SGD**), fixed K-mapping optimization (**FK-NT-SGD**), or an adjustable K-mapping optimization (**AK-NT-SGD**). As the name suggests, the brute force K-mapping optimization performs SGD on all valid K-mappings. It calculates the empirical cost of the converged warning times for each K-mapping and returns the lowest cost K-mapping and warning time pair. While this takes care of the problem of K-mapping, this search is over an exponential number of K-mappings, so we also present two

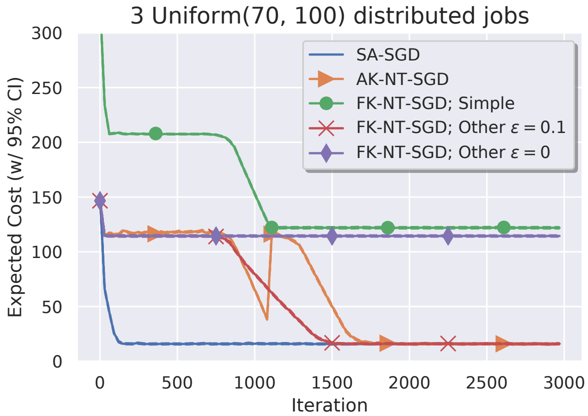

Fig. 5. The simple K-mapping does poorly; the "Other" K-mapping ($[-1, 1, 1]$) matches the static appointment scheme. Choosing $\varepsilon = 0.1$ leads to convergence. AK-NT-SGD is able to find the best K-mapping $T = 200$

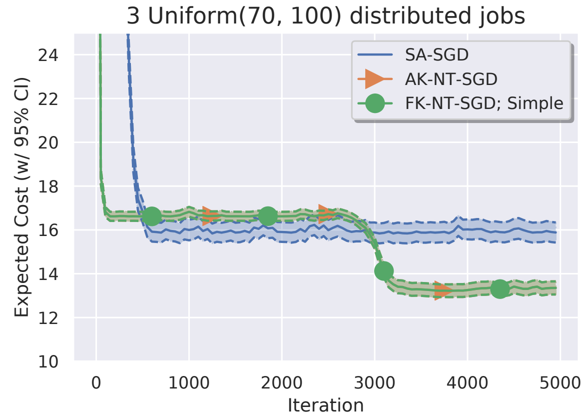

Fig. 6. The simple K-mapping outperforms the static appointment scheme. $T = 10$

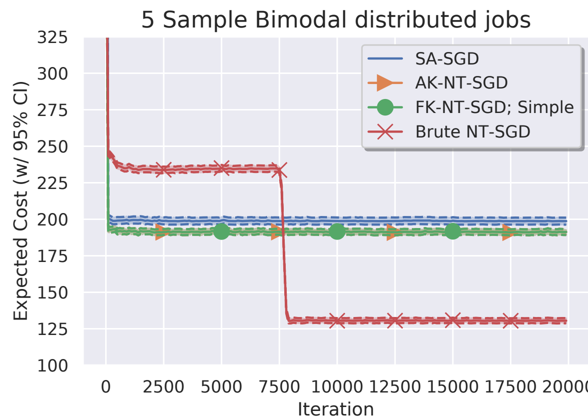

Fig. 7. The simple K-mapping outperforms the static appointment scheme; however, the simple K-mapping is not optimal. $T = 10$

approximate versions of the brute force algorithm. The fixed K-mapping version optimizes $\mathbf{w}$ for a fixed K-mapping; the default choice of K-mapping that we use is the simple K-mapping. The adjustable K-mapping algorithm decrements $K(j)$ if w_j becomes smaller than $\lambda > 0$ and increments it if w_j becomes larger than the $100 - \zeta$th percentile of the random variable $X_{K(j)}$ for $\zeta \geq 0$; λ and ζ are hyperparameters. We use the simple K-mapping to initialize the AK-NT-SGD algorithm.

IV. Empirical Results

In Subsection IV-A, we will begin with a toy schedule of 3 uniformly distributed jobs to demonstrate the importance of K-mapping and choice of T. Next, we evaluate the performance of our scheduling algorithms on a single queue of sample-based bimodal distribution that mimics what a true job duration may look like for various α, β, γ, and T. In Subsection IV-B, we repeat the sample-based bimodal experiment on a real hospital operating room schedule. We then demonstrate cost savings of the warning time scheme against the static scheme on over a years worth of real operating room schedules.

A. Results on synthetic data

We begin with a schedule of three independent uniformly distributed jobs. We fix $\alpha = 1$, $\beta = 3$ and $\gamma = 1.5$ because we prioritize the agent's time more than customer waiting time. We compare the static appointment cost against the notification time cost in Figure 5 with lead time $T = 200$ and in Figure 6 with a shorter lead time $T = 10$. We plot the average cost incurred by the schedules on 1000 sampled job duration vectors. We also form a 95% confidence interval on the sample average cost. We can see that these plots corroborate the findings described in Theorem 2. Only when T is short enough is the simple K-mapping able to outperform the static scheme.

Next we look at a schedule of 5 sample-based bimodally distributed jobs. Each job distribution is created out of 100 samples from a "true" bimodal distribution: 70% of the data points are generated from a $\mathcal{N}(60, 25)$ distribution and the other 30% of data is generated from a $\mathcal{N}(120, 25)$; negative samples are thrown out. The cost function is then evaluated on 1000 held out samples from the "true" bimodal distribution and a 95% confidence interval is also formed. This setup is more realistic because the bimodal distribution is similar to how jobs, such as surgeries, are distributed. Additionally, in practice, we would most likely only have access to a limited history of realizations. We again choose the cost parameters $\alpha = 1$, $\beta = 3$ and $\gamma = 1.5$. We plot the cost as a function of iteration of the SGD algorithm in Figure 7. Note that even though Theorem 2 does not hold (because $\gamma > 0$), the dynamic appointment scheme still outperforms the static appointment scheme.

It is clear that the choices of α, β, γ, T have a large impact on how well the notification time scheme works. So for a variety of choices of α, β, γ, T, we calculate the optimal static appointments and run all three variants of notification time SGD on the 5 sample-based bimodally distributed jobs.

α	β	γ	T	SA	FK-NT	BK-NT	AK-NT
1.0	3.0	1.5	40	196.7	**113.8**	**113.8**	**113.8**
1.0	3.0	1.5	80	196.7	225.8	**165.7**	**165.7**
1.0	3.0	1.5	120	196.7	555.7	**165.7**	196.0
3.0	1.0	1.5	40	269.0	**230.9**	**230.9**	**230.9**
3.0	1.0	1.5	80	269.0	246.9	**245.8**	248.2
3.0	1.0	1.5	120	269.0	339.7	**260.6**	**260.6**
1.0	1.5	3.0	40	187.5	**128.9**	**128.9**	**128.9**
1.0	1.5	3.0	80	187.5	202.4	**177.6**	**177.6**
1.0	1.5	3.0	120	187.5	511.1	**177.6**	211.0
1.0	1.0	1.0	40	140.5	**97.0**	**97.0**	**97.0**
1.0	1.0	1.0	80	140.5	**126.2**	**126.2**	139.9
1.0	1.0	1.0	120	140.5	249.5	**139.2**	**139.2**

Fig. 8. Best cost on 12 different α, β, γ, T combinations on 5 sample bimodal random variables after choosing the lowest cost solution for each of the 4 algorithms from 5 trials ($d = 420$).

These results can be found in Table 8. In each of these examples, even AK-NT-SGD may not always find the best K-mapping and warning time pair, it still generally performs better than the static appointment scheme.

B. Results on hospital operating room data

We look at one year of operating room (OR) data from a real hospital to see how our algorithm leads to cost savings. We use our data set to form an oracle. When we query the oracle for the duration of a surgery of a certain type, the oracle randomly selects a surgery of the same type and returns the duration of that particular surgery. We again test our algorithms for various choices of realistic α, β, γ, and T with 5 randomly initialized trials on one real operating room schedule from our data set; the best performing result from each algorithm can be found in Table 9. In this experiment, the optimal K-mapping outperforms the static cost. Again, even though many times AK-NT-SGD is unable to arrive at optimal K-mapping, it still often outperforms the static appointment scheme and the simple K-mapping scheme.

Next, we obtain 345 real operating room schedules from the data set. These schedules are selected because each 1) have at least 3 surgeries scheduled and 2) only include surgery types which have at least 20 previous observations. We use the realistic parameters: $\alpha = 1$, $\beta = 3$, $\gamma = 1.5$, $T = 60$ min, and $d = 7$ hours $\times$ 60 min. For every schedule, we run the SA-SGD, AK-NT-SGD, and FK-NT-SGD algorithms each 3 times. For each of these OR schedules, we calculate the percent cost saved by using the best performing notification schedule (i.e., the better of the returned AK-NT-SGD and FK-NT-SGD schedules) in place of the optimal static schedule. These results are shown in Figure 10. The average cost savings is 9.8%; the median cost savings is 9.2%. In only 16 of the 345 schedules did the notification scheme perform worse than the static scheme by more than 1%.

α	β	γ	T	SA	FK-NT	BK-NT	AK-NT
1.0	3.0	1.5	30	119.9	122.7	**91.4**	100.7
1.0	3.0	1.5	60	119.9	299.6	**106.7**	106.8
1.0	3.0	1.5	90	119.9	700.5	**111.3**	130.2
3.0	1.0	1.5	30	132.5	**105.6**	**105.6**	125.2
3.0	1.0	1.5	60	132.5	150.7	**114.9**	164.3
3.0	1.0	1.5	90	132.5	327.7	**130.6**	149.3
1.0	1.5	3.0	30	92.3	82.8	**74.4**	**74.4**
1.0	1.5	3.0	60	92.3	167.1	**78.8**	85.1
1.0	1.5	3.0	90	92.3	498.3	**87.3**	87.7
1.0	1.0	1.0	30	76.9	65.1	**63.7**	64.1
1.0	1.0	1.0	60	76.9	116.9	**65.3**	74.5
1.0	1.0	1.0	90	76.9	275.1	**71.6**	74.3

Fig. 9. Best cost from on 12 different α, β, γ, T combinations on 6 real operating room surgery duration random variables after choosing the lowest cost solution for each of the 4 algorithms from 5 trials ($d = 420$ minutes). T is realistic, ranging from 30 to 90 minutes.

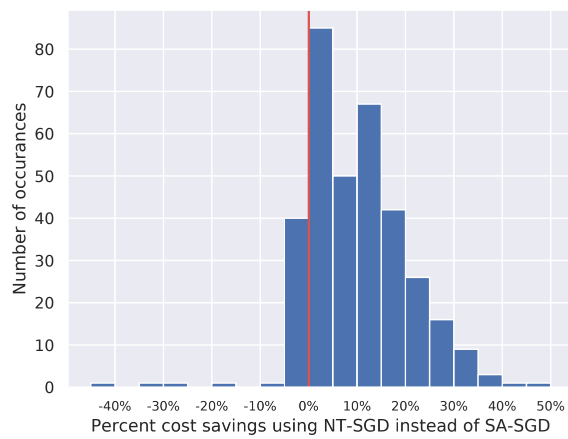

Fig. 10. For 345 realizations of operating rooms in our validation set, we calculate the cost saved by using the AK-NT-SGD scheme as opposed to the SA-SGD scheme. We then compile a histogram of the cost savings. $\alpha = 1$, $\beta = 3$, $\gamma = 1.5$, $T = 60$; $d = 7$ hours $\times$ 60 min $= 420$

V. Conclusion

In this paper, we modeled the static appointment problem as a minimization of a linear combination of expected wait cost, idle cost, and overtime cost, and we demonstrate that this function is convex with bounded gradients. We approached the appointment scheduling problem with a stochastic gradient descent algorithm and showed that the method will converge to the optimum at a rate of $\mathcal{O}(\log(t)/\sqrt{t})$. Next, we introduced the concept of notification times as a new way of dynamically approaching the appointment scheduling problem. We modified the SGD algorithm to suit this problem and constructed a warning time schedule which provably outperforms the static appointment scheme under certain conditions. Moreover, we empirically demonstrated on synthetic and on real hospital OR data that the dynamic scheme generally creates lower cost schedules than the optimal static scheme.

There is still much to explore in these two problem variants. Our work addresses the problem of scheduling jobs in a fixed order; the next step would be to integrate finding the optimal ordering of the jobs into the SGD algorithm. Another open question is whether the dynamic scheme can provably outperform the static scheme for *all* choices of lead times.

REFERENCES

[1] Miao Bai, Robert H. Storer, and Gregory L. Tonkay. “A sample gradient-based algorithm for a multiple-OR and PACU surgery scheduling problem.” In: *IISE Transactions* 49.4 (2017), pp. 367–380.

[2] Mehmet A. Begen and Maurice Queyranne. “Appointment Scheduling with Discrete Random Durations.” In: *Mathematics of Operations Research* 36.2 (2011), pp. 240–257.

[3] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. “Optimization Methods for Large-Scale Machine Learning.” In: *SIAM Review* 60.2 (2018), pp. 223–311.

[4] Brian Denton and Diwakar Gupta. “A Sequential Bounding Approach for Optimal Appointment Scheduling.” In: *IIE Transactions* 35.11 (2003), pp. 1003–1016.

[5] M. Eric Johnson and John Jackman. “Infinitesimal Perturbation Analysis: A Tool for Simulation.” In: *The Journal of the Operational Research Society* 40.3 (1989), pp. 243–254.

[6] Moutaz Khouja. “The single-period (news-vendor) problem: literature review and suggestions for future research.” In: *Omega* 27.5 (1999), pp. 537–553.

[7] Claude Dennis Pegden and Matthew Rosenshine. “Scheduling arrivals to queues.” In: *Computers and Operations Research* 17.4 (1990), pp. 343–348.

[8] Ohad Shamir and Tong Zhang. “Stochastic Gradient Descent for Non-smooth Optimization: Convergence Results and Optimal Averaging Schemes.” In: *Proceedings of the 30th International Conference on Machine Learning - Volume 28*. ICML’13. Atlanta, GA, USA: JMLR.org, 2013, pp. I-71–I-79.

[9] Peter Vanden Bosch, Dennis Dietz, and John Simeoni. “Scheduling Customer Arrivals to a Stochastic Service System.” In: 46 (Aug. 1999), pp. 549–559.

[10] Patrick Wang. “Optimally scheduling N customer arrival times for a single-server system.” In: *Computers and Operations Research* 24.8 (1997), pp. 703–716.

[11] Patrick Wang. “Static and dynamic scheduling of customer arrivals to a single-server system.” In: *Naval Research Logistics (NRL)* 40.3 (1993), pp. 345–360.

[12] Christos Zacharias and Michael Pinedo. “Appointment Scheduling with No-Shows and Overbooking.” In: *Production and Operations Management* 23.5 (2014), pp. 788–801.

[13] Zheng Zhang and Xiaolan Xie. “Simulation-based optimization for surgery appointment scheduling of multiple operating rooms.” In: *IIE Transactions* 47.9 (2015), pp. 998–1012.