Online Scheduling of Traffic Diversion and Cloud Scrubbing with Uncertainty in Current Inputs

Lei Jiao
Department of Computer and Information Science,
University of Oregon, Eugene, OR, USA

Ruiting Zhou
School of Cyber Science and Engineering,
Wuhan University, Wuhan, China

Xiaojun Lin
School of Electrical and Computer Engineering,
Purdue University, West Lafayette, IN, USA

Xu Chen
School of Data and Computer Science,
Sun Yat-sen University, Guangzhou, China

ABSTRACT

Operating distributed Scrubbing Centers (SCs) to mitigate massive Distributed Denial of Service (DDoS) traffic in large-scale networks faces critical challenges. The operator needs to determine the diversion rule installation and elimination in the networks, as well as the scrubbing resource activation and revocation in the SCs, while minimizing the long-term cost and the cumulative decision-switching penalty without knowing the exact amount of the malicious traffic. We model and formulate this problem as an online nonlinear integer program. In contrast to many other online problems where future inputs are unknown but at least current inputs are known, a key new challenge here is that even part of the current inputs are unknown when decisions are made. To "learn" the best decisions online, we transform our problem via a gap-preserving approximation into an online optimization problem with only the known inputs, which is further relaxed and decoupled into a series of one-shot convex programs solvable in individual time slots. To overcome the intractability, we design a progressive rounding algorithm to convert fractional decisions into integral ones without violating the constraints. We characterize the competitive ratio of our approach as a function of the key parameters of our problem. We conduct evaluations using real-world data and confirm our algorithms' superiority over de facto practices and state-of-the-art methods.

CCS CONCEPTS

• **Networks** → **Cloud computing**; • **Theory of computation** → *Online algorithms.*

KEYWORDS

DDoS defense, Cloud scrubbing, Online algorithm, Competitive analysis

ACM Reference Format:
Lei Jiao, Ruiting Zhou, Xiaojun Lin, and Xu Chen. 2019. Online Scheduling of Traffic Diversion and Cloud Scrubbing with Uncertainty in Current Inputs.

Mobihoc '19, July 2–5, 2019, Catania, Italy

ACM ISBN 978-1-4503-6764-6/19/07…$15.00
https://doi.org/10.1145/3323679.3326525

In *Mobihoc '19: The Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing, July 2–5, 2019, Catania, Italy.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3323679.3326525

1 INTRODUCTION

Recently, there has been increasing interest in moving the defense of Distributed Denial of Service (DDoS) attacks to the cloud. Internet Service Providers (ISPs) (e.g., AT&T) and DDoS protection service providers (e.g., Cloudflare) can set up the so-called Scrubbing Centers (SCs) to "scrub" the suspicious traffic and route only the legitimate traffic to their customers [21]. This service architecture can be visualized in Fig. 1, where the blue and the green solid lines represent the suspicious traffic (i.e., the traffic that potentially contains both the malicious traffic and the legitimate traffic), and the dashed lines represent the corresponding legitimate traffic. In the SCs, the suspicious traffic is inspected so that the malicious traffic, if any, is filtered out, and only the legitimate traffic is re-injected into the network and travels to the destinations.

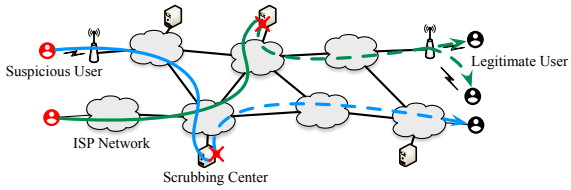

Figure 1: Filtering malicious traffic via scrubbing centers

In this paper, we consider the operation of a large-scale system of geographically distributed SCs that scrub a massive amount of traffic for many customers simultaneously. To divert suspicious traffic to appropriate SCs, the provider usually installs Border Gateway Protocol (BGP) rules in the routers [17]. Although one may also use Domain Name Systems [23] for traffic diversion, this approach can only be applied to application-level DDoS. Hence, we focus on using BGP rules in order to target a wider range of DDoS traffic.

Unfortunately, optimally operating such a distributed system is a highly challenging problem. There exists significant uncertainty in the inputs, as suspicious traffic often appears and disappears in the system unpredictably with time-varying volumes. As a result, this problem is intrinsically an "online" problem, where multiple intertwined decisions (e.g., routing, resource allocation) need to be made jointly, dynamically, and irrevocably on the fly, without knowing future inputs. While it is desirable to adapt decisions over time (e.g.,

install/remove BGP rules, allocate/revoke scrub resources) to reduce the operational cost (e.g., energy cost, BGP routing space cost, network footprint [14, 30]), one must also consider the "switching" cost for changing decisions. Specifically, installing new BGP rules causes propagation traffic and convergence delay before the system enters a consistent state [18]; allocating additional scrub resources incurs start-up time, system oscillation, reliability risk, and/or hardware wear and tear [22]. Dynamically balancing the operational cost and the switching cost is non-trivial, because the decision made in any time slot affects the switching cost between itself and the decision for the next time slot, while the latter has not been made until the next time slot actually arrives.

One critical new challenge that makes existing online algorithms that address switching costs [10, 20, 22, 26] not directly applicable to our problem is as follows. In most existing online problems, although future information is unavailable, the inputs to the current time slot are at least known. In contrast, for our problem, when making the decision for the current time slot, even the current inputs are not completely known. In particular, when choosing an SC for a minimal network footprint to scrub a suspicious traffic flow, one does not know how much malicious traffic is actually carried in that suspicious flow. Instead, this information is only learnt after an SC is chosen and the traffic is scrubbed. Further, one may not even know the amount of the suspicious traffic when allocating scrubbing resources. This difference creates a fundamental challenge as a suspicious flow may turn out to contain all, some, or no malicious traffic, and thus a decision that has been made may turn out inappropriate when the malicious traffic is revealed. Similarly, scrubbing resources allocated beforehand may turn out to be insufficient when the amount of traffic to be scrubbed is disclosed.

Intuitively, after the decision is made and executed online, one can learn from the history of the past outcomes the best decision that should have been made. This has some similarity to *regret minimization* in the online learning settings [9, 11, 13]. However, regret minimization problems often have no switching cost in the objective. Further, they compare the online decisions to either the *static* offline optimal decisions that stay unchanged over time [9], or the best *shifting*/*drifting* offline optimums where the decision changes are not free but constrained [11, 13]. Besides the switching cost, our problem inherently has dynamic decisions, and it makes more sense to compare to the *dynamic* offline optimal decisions, as in *competitive analysis*. However, in most competitive analysis, all inputs to each current time slot are known [10, 26], which is not true in our case. Thus, we would refer to the problem that we study as a *competitive online learning* problem, which can be regarded as a combination of regret minimization and competitive analysis.

To the best of our knowledge, this paper is the first formal study of scheduling traffic diversion and scrubbing resource management to mitigate DDoS in a dynamic online setting. Existing efforts on DDoS defense using clouds/SCs either do not consider redirection costs, switching costs, and algorithms with performance guarantees [14, 28, 30], or focus on empirical systems without theoretical/algorithmic insights [23, 27]. Meanwhile, existing research on online cloud resource allocation with switching costs [15, 20, 22, 29] often cannot capture all the factors in our scenario, such as discrete-decision control and partially unknown inputs. From the algorithmic perspective, this is also the first treatment using competitive online algorithms for problems with unknown, nonlinear costs and integer decision variables in each current time slot. See Section 7 for more discussions. We make three contributions:

First, we model and formulate this problem as a nonlinear integer program that minimizes the long-term total cost, consisting of the operational cost of the diversion rules and the scrub resources, the switching cost of changing diversion and resource allocation decisions, and the network footprint of suspicious and legitimate traffic. Our models grasp the essential elements of the scenario based on rather mild and general assumptions and can capture arbitrary network topologies, flow patterns, volume variations, attack heterogeneities, malicious/legitimate traffic combinations, and resource price dynamics. To explore possible further cost reduction, in this paper, we additionally allow dynamically switching on/off entire SCs to save SC-level costs such as those of cooling, lighting, and power provisioning, which is intriguing for small SCs that often have inferior Power Usage Effectiveness (PUE) [16].

Second, we propose and design a key set of novel algorithms for solving our problem online with provable competitive guarantees. To overcome the difficulty that the current inputs of the amount of the malicious traffic are not known, we approximate the online learning component in our problem, and transform the problem into another different but related online fractional problem that only involves the known inputs. Afterwards, we develop an online algorithm to decouple this new problem into a series of one-shot convex programs with carefully-designed logarithmic terms replacing the nonlinear switching cost in the objective [10], which can be solved in each individual time slot only by observing the inputs that are available in each time slot and taking the solution obtained from the previous time slot. Furthermore, we devise a progressive rounding algorithm to convert the factional decisions into integers in batches via invoking our third algorithm which iteratively selects and rounds two fractions together, rather than separately, to compensate each other without violating any constraints of the problem [8]. Having all the three component algorithms, we prove the overall competitive ratio of our online algorithm framework, i.e., the maximum ratio of the total cost incurred by the online approach with partially unknown current inputs over the total cost incurred by the offline optimal approach that knows all the inputs in advance. We also provide guidelines for extending our algorithms and analysis to the situation where both the suspicious and the malicious traffic contained are unknown in each time slot.

Third, we conduct evaluations based on about 250-GB real-world dynamic DDoS traces from Booter in 2013 [25]. We let the DDoS traffic from the 5822 US sources travel over real-world Autonomous System (AS) topologies [5] to attack up to 48 US targets [1], with different amounts of legitimate traffic. Having 11 SCs to scrub all such traffic at real-world locations [3, 4] with real-world operational costs [6, 7] and switching costs [22], we obtain several promising results: (i) our approach saves up to 50%, 58%, and 25% total cost compared to today's industrial practice, a greedy algorithm, and a state-of-the-art algorithm, respectively; (ii) it achieves even better performance, i.e., 62% lower total cost, as the legitimate traffic volume grows; (iii) for small-scale SCs, switching on/off entire SCs via our approach can reduce 20%∼26% total cost by reducing non-IT energy consumption; (iv) our approach scales well and its execution time grows only mildly as the problem size increases.

2 MODELING AND FORMULATION

Networks and Scrubbing Centers. We use $\mathcal{I}$ to refer to the set of all SCs, possibly connected to different ISP networks or ASes, and the integer U_i, $\forall i \in \mathcal{I}$ to refer to the SC i's capacity in terms of the total amount of resources, such as the total number of Virtual Machines (VMs). We consider a horizon of time slots $\mathcal{T} = \{1, 2, 3, ..., T\}$, corresponding to the frequency of making decisions.

Traffic Flows and Routing. We consider "flow" as the basic traffic management unit. We denote the set of all flows in the system by $\mathcal{J}$, where each flow corresponds to a pair of a source IP address range and a destination IP address range. We define a binary indicator $\lambda_{jt} \in \{0, 1\}$, $\forall j \in \mathcal{J}$, $\forall t \in \mathcal{T}$, and set it to 1 if the flow j appears and is suspicious in the time slot t and set it to 0 if it does not appear or is not suspicious in the time slot t. We make no assumption on how λ_{jt} varies with j and t, in order to capture the arbitrary dynamics of all the flows and their routing over time. We only consider "single-path routing", i.e., a flow always takes one path and cannot travel through multiple paths simultaneously.

Cost of Traffic Diversion. We install and remove BGP rules (or routes) dynamically in the networks. We use e_{ijt}, $\forall i \in \mathcal{I}$, $\forall j \in \mathcal{J}$, $\forall t \in \mathcal{T}$ to denote the cost of hosting the BGP rule in the networks to divert the flow j to the SC i in the time slot t, and use f_{ij}, $\forall i \in \mathcal{I}$, $\forall j \in \mathcal{J}$ to denote the cost of installing this BGP rule in the networks. e_{ijt} can capture the expense, such as that incurred by the rule occupying the router space; f_{ij} can capture the performance impact, such as BGP rules' propagation traffic and convergence delay.

Cost of Scrubbing. We allocate and revoke resources such as VMs dynamically, with their installed automation software such as intrusion detection systems, to inspect and scrub the traffic. As different suspicious flows may contain different types of malicious traffic, each suspicious flow often requires a different type of VMs, such as a VM with a specified set of scrub software installed, which cannot be shared across different flows. For simplicity, we further assume every VM consumes the same amount of physical resources. We use c_{ijt} to denote the operational cost to run a VM for the flow j at the SC i in the time slot t, and use d_{ij} to denote the switching cost of turning on one VM. c_{ijt} can capture the electricity price, the VM monetary cost, the software maintenance expense, for example; d_{ij} can capture the resource preparation delay, reliability risk, and hardware wear and tear. We also use V_j to denote the amount of suspicious traffic that is inspected by a single VM for the flow j.

Moreover, in this paper, we allow switching on/off the entire SCs for further cost saving, due to the following reasons. First, there may be no suspicious traffic for an SC to scrub at certain times. Second, shutting down SCs can save the non-IT (e.g., cooling, lighting, power provisioning) energy cost, which is particularly useful for small and micro data centers that consume considerable non-IT energy compared to their IT energy. We use a_{it} to represent the unit operational cost to run the SC i in the time slot t, and use b_i to represent the switching cost to switch on the SC i. a_{it} and b_i capture the SC-level cost. For example, a_{it} can represent the non-IT energy cost of an SC, or the SC facility management cost; b_i can represent the hardware wear and tear of all the shared, SC-level non-IT resources and infrastructural facilities.

Network Footprint. "Network footprint" is important in traffic scrubbing [14, 30], and captures the network resource consumption outside the SCs. The network footprint of a flow can be defined as the product of the volume of the flow times the length of the path the flow travels through. The path length can be in terms of the number of hops, the network latency, etc. We use g_{ij} to denote the length of the path via which the flow j travels from its source to the SC i, and use h_{ij} to denote the length of the path via which the flow j travels from the SC i to its destination. We use σ_{jt} to denote the volume of the suspicious flow j before it reaches any SC, and use δ_{jt+1}, where $\delta_{jt+1} \leq \sigma_{jt}$, to denote the volume of the legitimate flow j after dropping the malicious traffic at any SC. We highlight that, when making the scheduling decisions at t, we know σ_{jt}, but we only know δ_{jt+1} after making the decisions. To capture this, without loss of generality, we write the time subscript in δ_{jt+1} as $t+1$ instead of t [11].

Scheduling Decisions. We have three categories of decision variables. $x_{it} \in \{0, 1\}$ denotes whether we activate the SC i in the time slot t. $y_{ijt} \in \{0, 1, 2, 3, ...\}$ denotes the number of VMs we allocate for the flow j at the SC i in the time slot t. $z_{ijt} \in \{0, 1\}$ denotes whether we divert the flow j to the SC i in the time slot t. We assume all parameters and variables equal zero for all $t \notin \mathcal{T}$.

Problem Formulation. We formulate the problem $\mathbb{P}$:

$$
\begin{aligned}
\min \quad & \mathsf{P} = \textstyle\sum_t \sum_i a_{it} x_{it} + \sum_t \sum_i b_i \left(x_{it} - x_{it-1}\right)^+ \\
& + \textstyle\sum_t \sum_i \sum_j c_{ijt} y_{ijt} + \sum_t \sum_i \sum_j d_{ij} \left(y_{ijt} - y_{ijt-1}\right)^+ \\
& + \textstyle\sum_t \sum_i \sum_j e_{ijt} z_{ijt} + \sum_t \sum_i \sum_j f_{ij} \left(z_{ijt} - z_{ijt-1}\right)^+ \\
& + \textstyle\sum_t \sum_i \sum_j \left(\sigma_{jt} g_{ij} + \delta_{jt+1} h_{ij}\right) z_{ijt} \\
\text{s.t.} \quad & U_i x_{it} \geq \textstyle\sum_j y_{ijt}, \quad \forall i, \forall t, && \text{(1a)} \\
& V_j y_{ijt} \geq \sigma_{jt} z_{ijt}, \quad \forall i, \forall j, \forall t, && \text{(1b)} \\
& \textstyle\sum_i z_{ijt} \geq \lambda_{jt}, \quad \forall j, \forall t, && \text{(1c)} \\
& x_{it} \in \{0, 1\}, \quad \forall i, \forall t, && \text{(1d)} \\
& y_{ijt} \in \{0, 1, 2, 3, ...\}, z_{ijt} \in \{0, 1\}, \forall i, \forall j, \forall t. && \text{(1e)}
\end{aligned}
$$

The objective minimizes the total cost of operating and switching on/off SCs and the resources inside SCs, the total cost of hosting and installing/removing BGP rules in the networks, plus the total network footprint over time. Different types of costs can have weights associated to them; we omit all the weights for the ease of presentation. Constraint (1a) ensures that the resources are allocated within the capacities of the SCs. Constraint (1b) ensures that sufficient resources are allocated for scrubbing the suspicious traffic. Constraint (1c), with Constraint (1e), ensures that every suspicious flow is diverted to only one SC. Constraints (1d) and (1e) ensure that all decisions are integers. In the objective, the special function $(\cdot)^+ \triangleq \max\{\cdot, 0\}$ captures the switching cost. If turning on/off SCs are out of the interest and all SCs are always on, one can remove the terms involving x_{it} from the objective, remove x_{it} from (1a), and remove the entire (1d). The problem will not become easier without x_{it}; our algorithms and analysis can be adjusted accordingly.

Problem Challenges. We need to overcome two challenges to design an online algorithm with any performance guarantee. The first challenge is online learning. We need to make decisions on the fly in each time slot without knowing all the future inputs, while expecting such decisions to have a guaranteed performance gap towards the (even unknown) offline optimum. For most other studies, when making decisions online in one time slot, the entire inputs to that time slot are often known; in contrast, in our problem we can only observe the partial inputs for a time slot, and learn the

rest of the inputs for that time slot after making the decisions. The second challenge is the intractability. We make discrete decisions. Even in the offline setting and even without the switching cost, our problem is a more complex version of the covering problems, known to be NP-hard [10]. Although there are existing approximation algorithms with performance guarantees for covering problems, the online nature, plus the covering constraints that chain multiple variables, makes existing algorithms hard to be applied here while maintaining or adapting their performance guarantees.

3 ALGORITHM DESIGN

Our algorithm framework consists of the online fractional algorithm, Algorithm 1, and the progressive rounding algorithm, Algorithm 2 which iteratively invokes Algorithm 3. In every time slot in an online manner, we relax the integral constraints and obtain the fractional solutions, and then round such fractions into integers.

Algorithm 1: Online Fractional Algorithm, $\forall t$

Solve the problem $\widetilde{\mathbb{P}}_t$ below and get the solution $\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}\}$:

$$\begin{aligned}\min\quad & \widetilde{\mathrm{P}}_\mathrm{t} = \textstyle\sum_i a_{it}x_{it} + \sum_i \sum_j c_{ijt}y_{ijt} \\ & + \textstyle\sum_i \sum_j (e_{ijt} + \sigma_{jt}g_{ij} + \delta_{jt}h_{ij})z_{ijt} \\ & + \textstyle\sum_i \frac{b_i}{\eta}\left((x_{it}+\varepsilon)\ln\frac{x_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} - x_{it}\right) \\ & + \textstyle\sum_i \sum_j \frac{d_{ij}}{\eta_i}\left((y_{ijt}+\varepsilon)\ln\frac{y_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon} - y_{ijt}\right) \\ & + \textstyle\sum_i \sum_j \frac{f_{ij}+2\delta_{jt}\max_i h_{ij}}{\eta}\left((z_{ijt}+\varepsilon)\ln\frac{z_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon} - z_{ijt}\right) \\ \text{s. t.}\quad & \text{(1a), (1b), (1c), without ``}\forall t\text{''}, \\ & x_{it} \le 1,\ z_{ijt} \ge 0,\ \forall i,\ \forall j,\end{aligned}$$

where $\varepsilon > 0$, $\eta = \ln\left(1+\frac{1}{\varepsilon}\right)$, $\eta_i = \ln\left(1+\frac{U_i}{\varepsilon}\right)$, $\forall i$.

Algorithm 1, inspired by the regularization technique [10], invokes a standard convex solver to solve the convex program $\widetilde{\mathbb{P}}_t$ in each time slot t, and uses the solution solved from the problem $\widetilde{\mathbb{P}}_t$ as the solution to our original problem $\mathbb{P}$ in t. We construct $\widetilde{\mathbb{P}}_t$ by (i) extracting the one-shot slice of $\mathbb{P}$, (ii) relaxing integral constraints to continuous domains, (iii) replacing the switching cost terms in the objective by our carefully-designed logarithmic terms, i.e., "regularization", and (iv) changing δ_{jt+1} to δ_{jt}, $\forall j$ and adding the new terms $2\delta_{jt}\max_i h_{ij}$ to f_{ij}, $\forall i$, $\forall j$.

Our core ideas here lie in (iii) and (iv) mentioned as above. The regularization process substitutes the function $(\Delta_t - \Delta_{t-1})^+$ by the regularizer $(\Delta_t+\varepsilon)\ln\frac{\Delta_t+\varepsilon}{\Delta_{t-1}+\varepsilon} - \Delta_t$, with $\varepsilon > 0$. This convex, differentiable, logarithmic-based term enables us to decouple the problem and solve $\widetilde{\mathbb{P}}_t$ by taking the solution of the previous time slot, i.e., the solution to $\widetilde{\mathbb{P}}_{t-1}$, as input, while guaranteeing the gap towards the offline optimal decisions, as shown in Section 4.1. To overcome the difficulty of online learning, we additionally modify δ_{jt+1} and add $2\delta_{jt}\max_i h_{ij}$, which is a new step on top of regularization and enables us to approximate the online learning term by using the inputs at t without worrying about not seeing the inputs that only come at $t+1$, as shown in Section 4.3. Essentially, we bound the unknown inputs at $t+1$ by the known inputs at t, which is our key to address online learning.

Algorithm 2 progressively and respectively rounds the solutions $\widetilde{\mathbf{x}}_\mathbf{t}$, $\widetilde{\mathbf{y}}^*_\mathbf{t}$, and $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$ into integers by invoking Algorithm 3 as a subroutine

Algorithm 2: Progressive Rounding Algorithm, $\forall t$

Take $\widetilde{\mathbf{x}}_\mathbf{t}$ from Algorithm 1;
Invoke Algorithm 3 on $\widetilde{\mathbf{x}}_\mathbf{t}$ to get the integral $\bar{\mathbf{x}}_\mathbf{t}$;
Fix $\bar{\mathbf{x}}_\mathbf{t}$ in $\widetilde{\mathbb{P}}_t$, solve $\widetilde{\mathbb{P}}_t$ to get its partial fractional solution $\widetilde{\mathbf{y}}^*_\mathbf{t}$ and $\widetilde{\mathbf{z}}^*_\mathbf{t}$;
Invoke Algorithm 3 on $\widetilde{\mathbf{y}}^*_\mathbf{t}$ to get the integral $\bar{\mathbf{y}}_\mathbf{t}$;
Fix $\bar{\mathbf{x}}_\mathbf{t}$ and $\bar{\mathbf{y}}_\mathbf{t}$ in $\widetilde{\mathbb{P}}_t$, solve $\widetilde{\mathbb{P}}_t$ to get its partial fractional solution $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$;
Invoke Algorithm 3 on $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$ to get the integral $\bar{\mathbf{z}}_\mathbf{t}$.

Algorithm 3: Weighted Pairwise Rounding Algorithm

1 For $\widetilde{\mathbf{x}}_\mathbf{t}$, set $\mathcal{M} = \{\emptyset\}$, $\mathcal{N} = \mathcal{I}$, $C_n = U_n\frac{\max_j V_j}{\min_j \sigma_{jt}}$, $\forall n \in \mathcal{N}$;
2 For $\widetilde{\mathbf{y}}^*_\mathbf{t}$, set $\mathcal{M} = \mathcal{I}$, $\mathcal{N} = \mathcal{J}$, $C_n = \frac{V_n}{\sigma_{nt}}$, $\forall n \in \mathcal{N}$;
3 For $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$, set $\mathcal{M} = \mathcal{J}$, $\mathcal{N} = \mathcal{I}$, $C_n = 1$, $\forall n \in \mathcal{N}$;
4 Denote $\widetilde{\mathbf{x}}_\mathbf{t}$, or $\widetilde{\mathbf{y}}^*_\mathbf{t}$, or $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$, by A^m_n, $\forall m \in \mathcal{M}$, $\forall n \in \mathcal{N}$;
5 Denote $\bar{\mathbf{x}}_\mathbf{t}$, or $\bar{\mathbf{y}}_\mathbf{t}$, or $\bar{\mathbf{z}}_\mathbf{t}$, by $\bar{A}^m_n$, $\forall m \in \mathcal{M}$, $\forall n \in \mathcal{N}$;
6 **for** $m \in \mathcal{M}$ **do**
7 $B_n \overset{\text{def}}{=} A^m_n - \lfloor A^m_n \rfloor$, $\forall n \in \mathcal{N}$;
8 $\mathcal{N}' \overset{\text{def}}{=} \mathcal{N} \setminus \{n \mid B_n \in \{0,1\}\}$;
9 **while** $|\mathcal{N}'| > 1$ **do**
10 Select $n_1, n_2 \in \mathcal{N}'$, where $n_1 \ne n_2$;
11 Set $D_1 = \min\{C_{n_1}(1-B_{n_1}), C_{n_2}B_{n_2}\}$;
12 Set $D_2 = \min\{C_{n_1}B_{n_1}, C_{n_2}(1-B_{n_2})\}$;
13 Set $E_{n_1} = B_{n_1} + \frac{D_1}{C_{n_1}}$, $E_{n_2} = B_{n_2} - \frac{D_1}{C_{n_2}}$,
14 $F = \widetilde{\mathrm{P}}_\mathrm{t}(\lfloor A^m_{n_1} \rfloor + E_{n_1}, \lfloor A^m_{n_2} \rfloor + E_{n_2})$;
15 Set $E'_{n_1} = B_{n_1} - \frac{D_2}{C_{n_1}}$, $E'_{n_2} = B_{n_2} + \frac{D_2}{C_{n_2}}$,
16 $F' = \widetilde{\mathrm{P}}_\mathrm{t}(\lfloor A^m_{n_1} \rfloor + E'_{n_1}, \lfloor A^m_{n_2} \rfloor + E'_{n_2})$;
17 **if** $F \le F'$ **then** Set $B'_{n_1} = E_{n_1}$, $B'_{n_2} = E_{n_2}$;
18 **else** Set $B'_{n_1} = E'_{n_1}$, $B'_{n_2} = E'_{n_2}$;
19 **if** $B'_{n_1} \in \{0,1\}$ **then** Set $\bar{A}^m_{n_1} = \lfloor A^m_{n_1} \rfloor + B'_{n_1}$, $\mathcal{N}' = \mathcal{N}' \setminus \{n_1\}$;
20 **else** Set $B_{n_1} = B'_{n_1}$;
21 **if** $B'_{n_2} \in \{0,1\}$ **then** Set $\bar{A}^m_{n_2} = \lfloor A^m_{n_2} \rfloor + B'_{n_2}$, $\mathcal{N}' = \mathcal{N}' \setminus \{n_2\}$;
22 **else** Set $B_{n_2} = B'_{n_2}$;
23 **end**
24 **if** $|\mathcal{N}'| = 1$ **then** Set $\bar{A}^m_n = \lceil A^m_n \rceil$ for the only $n \in \mathcal{N}'$;
25 **end**

at every t. After rounding $\widetilde{\mathbf{x}}_\mathbf{t}$, it re-solves the problem to get $\widetilde{\mathbf{y}}^*_\mathbf{t}$ and $\widetilde{\mathbf{z}}^*_\mathbf{t}$; then, after rounding $\widetilde{\mathbf{y}}^*_\mathbf{t}$, it re-solves the problem to get $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$; finally, it rounds $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$. Following Constraints (1a) and (1b), we round from the "outer" variables to the "inner" variables along the "covering" chains, so that every time when fixing the rounded variables, $\widetilde{\mathbb{P}}_t$ is still feasible and can be re-solved. More discussion is in Section 4.2.

Algorithm 3, inspired by the dependent rounding technique [8], chooses a pair of fractions for rounding in each iteration until a single fraction may be left. The algorithm has two properties: (i) at least one of two fractions is rounded into an integer in each iteration; (ii) the weighted sum of the two fractions remains unchanged before and after rounding. To verify (i), for Line 13, we have $E_{n_1} = 1$ and $E_{n_2} \ge 0$, if $D_1 = C_{n_1}(1-B_{n_1})$; we have $E_{n_2} = 0$ and $E_{n_1} \le 1$, if $D_1 = C_{n_2}B_{n_2}$. We also naturally have $E_{n_2} \le 1$ and $E_{n_1} \ge 0$. We have analogous observations for Line 15. If a single fraction is left, as in Line 24, we round it up without violating the constraints of our problem. To verify (ii), no matter we execute Line 17 or 18, we always have $C_{n_1}B'_{n_1} + C_{n_2}B'_{n_2} = C_{n_1}B_{n_1} + C_{n_2}B_{n_2}$.

We choose between Lines 17 and 18 according to the objective function value of the rounded variables (i.e., all other variables not rounded yet stay unchanged) in Lines 14 and 16. The algorithm lets two fractions compensate each other so that all constraints are obeyed, while intuitively rounding each variable separately can violate the constraints. We adopt some new notations, such as $\mathcal{M}$ and $\mathcal{N}$, to provide a uniform formal treatment for $\widetilde{\mathbf{x}}_\mathbf{t}$, $\widetilde{\mathbf{y}}^*_\mathbf{t}$, and $\widetilde{\mathbf{z}}^{**}_\mathbf{t}$.

4 PERFORMANCE ANALYSIS

This section focuses on the "competitive analysis" for our algorithm framework. That is, we prove that the total cost over time incurred by the integer decisions produced by our online algorithms, i.e., $\mathsf{P}(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\})$, is upper-bounded by a constant times the offline optimum, i.e., $r\mathsf{P}_{opt}$:

$$\begin{aligned}
&\mathsf{P}(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\}) && \text{(2a)}\\
&\le \mathsf{P}'(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\}) && \text{(2b)}\\
&\le r_2\mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\}) && \text{(2c)}\\
&\le r_1 r_2 \mathsf{P}'_{opt} && \text{(2d)}\\
&\le r_1 r_2 r_3 \mathsf{P}_{opt}, && \text{(2e)}
\end{aligned}$$

where the constant $r = r_1 r_2 r_3$ is the "competitive ratio". To connect (2a) to (2e), we construct an auxiliary problem $\mathbb{P}'$ as the bridge, and in $\mathbb{P}'$ the online learning part is "handled", as described previously.

- Section 4.1: From (2c) to (2d), we connect P' evaluated with online fractional solutions to its offline optimum, and so we show the performance of our online fractional solutions.
- Section 4.2: From (2b) to (2c), we connect P' evaluated with online integer solutions to P' evaluated with online fractional solutions, and so we show the performance of rounding.
- Section 4.3: From (2a) to (2b), we connect our original problem $\mathbb{P}$ to the problem $\mathbb{P}'$, evaluated with the same online integer solutions; from (2d) to (2e), we connect the offline optimum of $\mathbb{P}'$ to that of $\mathbb{P}$. They jointly show the performance of our approximation of online learning.

4.1 Performance of Online Fractions

We prove $\mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\}) \le r_1 \mathsf{P}'_{opt}$ and show r_1 in this section.

Step 1: Formulating Problem $\mathbb{P}'$ **.** We construct and formulate the problem $\mathbb{P}'$ below, which is derived from our original problem $\mathbb{P}$. Note the differences between the formulation of $\mathbb{P}$ and that of $\mathbb{P}'$ in both the objective and the constraints:

$$\begin{aligned}
\min\quad & \mathsf{P}' = \textstyle\sum_t \sum_i a_{it}x_{it} + \sum_t \sum_i b_i u_{it}\\
& + \textstyle\sum_t \sum_i \sum_j c_{ijt}y_{ijt} + \sum_t \sum_i \sum_j d_{ij}v_{ijt}\\
& + \textstyle\sum_t \sum_i \sum_j (e_{ijt} + \sigma_{jt}g_{ij} + \delta_{jt}h_{ij})z_{ijt}\\
& + \textstyle\sum_t \sum_i \sum_j (f_{ij} + 2\delta_{jt}\max_i h_{ij})w_{ijt}\\
\text{s.t.}\quad & \text{(1a), (1b), (1c),}\\
& u_{it} \ge x_{it} - x_{it-1}, \quad \forall i, \forall t, && \text{(3a)}\\
& v_{ijt} \ge y_{ijt} - y_{ijt-1}, \quad \forall i, \forall j, \forall t, && \text{(3b)}\\
& w_{ijt} \ge z_{ijt} - z_{ijt-1}, \quad \forall i, \forall j, \forall t, && \text{(3c)}\\
& x_{it} \le 1, \quad \forall i, \forall t, && \text{(3d)}\\
& u_{it} \ge 0, v_{ijt} \ge 0, w_{ijt} \ge 0, z_{ijt} \ge 0, \forall i, \forall j, \forall t. && \text{(3e)}
\end{aligned}$$

Step 2: Formulating Problem $\mathbb{D}$ **.** We derive the Lagrange dual problem, i.e., $\mathbb{D}$, of $\mathbb{P}'$, where we use α_{it}, β_{ijt}, γ_{jt}, μ_{it}, ρ_{ijt}, ϕ_{ijt}, and τ_{it} to denote the dual variables for (1a) through (1c), and (3a) through (3d), respectively.

$$\begin{aligned}
\max\quad & \mathsf{D} = \textstyle\sum_t \sum_j \lambda_{jt}\gamma_{jt} + \sum_t \sum_i \left(\sum_j \lambda_{jt} - \sigma'_{it}\right)\tau_{it}\\
\text{s.t.}\quad & a_{it} - U_i\alpha_{it} + \mu_{it} - \mu_{it+1} + \sigma'_{it}\tau_{it} - \sigma'_{it}\textstyle\sum_i \tau_{it} = 0, \quad \forall i, \forall t,\\
& c_{ijt} + \alpha_{it} - V_j\beta_{ijt} + \rho_{ijt} - \rho_{ijt+1} = 0, \quad \forall i, \forall j, \forall t,\\
& e'_{ijt} + \sigma_{jt}\beta_{ijt} - \gamma_{jt} + \phi_{ijt} - \phi_{ijt+1} \ge 0, \forall i, \forall j, \forall t,\\
& b_i - \mu_{it} \ge 0, \quad \forall i, \forall t,\\
& d_{ij} - \rho_{ijt} \ge 0, \quad \forall i, \forall j, \forall t,\\
& f'_{ijt} - \phi_{ijt} \ge 0, \quad \forall i, \forall j, \forall t,\\
& \text{all dual variables} \ge 0,
\end{aligned}$$

where $\sigma'_{it} = U_i \max_j V_j / \min_j \sigma_{jt}$, $e'_{ijt} = e_{ijt} + \sigma_{jt}g_{ij} + \delta_{jt}h_{ij}$, $f'_{ijt} = f_{ij} + 2\delta_{jt}\max_i h_{ij}$.[1]

Step 3: Composing Solution of $\mathbb{D}$ **.** We construct the function M which converts $\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\}$, with $\widetilde{\mathbb{P}}_t$'s dual variables $\widetilde{\alpha}_{it}$, $\widetilde{\beta}_{ijt}$, $\widetilde{\gamma}_{jt}$, $\widetilde{\tau}_{it}$ and $\widetilde{\omega}_{ijt}$, into the solution of $\mathbb{D}$. In the rest of this paper, we write "D" to refer to the specific value of the function D evaluated with the specifically constructed solution as in Lemma 1.

Lemma 1. *The following constructed solution is feasible for* $\mathbb{D}$*:*

$$\alpha_{it} = \widetilde{\alpha}_{it}, \beta_{ijt} = \widetilde{\beta}_{ijt}, \gamma_{jt} = \widetilde{\gamma}_{jt}, \tau_{it} = \widetilde{\tau}_{it}, \mu_{it} = \frac{b_i}{\eta}\ln\frac{1+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon},$$
$$\rho_{ijt} = \frac{d_{ij}}{\eta_i}\ln\frac{U_i+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon}, \phi_{ijt} = \frac{f'_{ijt}}{\eta}\ln\frac{1+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon}, \forall i, \forall j, \forall t.$$

Proof. See Appendix A.1. The proof is by exhibiting that the constructed solution satisfies $\mathbb{D}$'s constraints, via the Karush-Kuhn-Tucker (KKT) conditions of $\widetilde{\mathbb{P}}_t$ which is defined in Algorithm 1. □

Step 4: Bounding. Having the above, now we are ready to prove $\mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\}) \le r_1 \mathsf{D}(\{\mathsf{M}(\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}), \forall t\})$. Note that, due to weak duality, $\mathsf{D}(\{\mathsf{M}(\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}), \forall t\}) \le \mathsf{P}'_{opt}$ holds automatically. Thus, showing the following theorem is sufficient for deriving r_1.

Theorem 1. *The non-switching cost in* $\mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\})$ *can be upper-bounded as follows:*

$$\textstyle\sum_t \sum_i a_{it}\widetilde{x}_{it} + \sum_t \sum_i \sum_j c_{ijt}\widetilde{y}_{ijt} + \sum_t \sum_i \sum_j e'_{ijt}\widetilde{z}_{ijt} \le \mathsf{D}.$$

The switching cost in $\mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\})$ *can be upper-bounded as follows, respectively:*

$$\begin{aligned}
&\textstyle\sum_t \sum_i b_i(\widetilde{x}_{it} - \widetilde{x}_{it-1})^+ \le \left((1+\varepsilon)\ln\left(1+\frac{1}{\varepsilon}\right)\max_j V_j \sum_i U_i\right)\mathsf{D},\\
&\textstyle\sum_t \sum_i \sum_j d_{ij}(\widetilde{y}_{ijt} - \widetilde{y}_{ijt-1})^+ \le\\
&\qquad \left(\max_i\left\{(U_i+\varepsilon)\ln\left(1+\frac{U_i}{\varepsilon}\right)\right\}\max_j V_j|\mathcal{I}|\right)\mathsf{D},\\
&\textstyle\sum_t \sum_i \sum_j f'_{ijt}(\widetilde{z}_{ijt} - \widetilde{z}_{ijt-1})^+ \le \left((1+\varepsilon)\ln\left(1+\frac{1}{\varepsilon}\right)|\mathcal{I}|\right)\mathsf{D}.
\end{aligned}$$

Joining all the above, we have $r_1 = 1 + (1+\varepsilon)\ln\left(1+\frac{1}{\varepsilon}\right)\max_j V_j \sum_i U_i + \max_i\left\{(U_i+\varepsilon)\ln\left(1+\frac{U_i}{\varepsilon}\right)\right\}\max_j V_j|\mathcal{I}| + (1+\varepsilon)\ln\left(1+\frac{1}{\varepsilon}\right)|\mathcal{I}|$.

Proof. See Appendix A.2. The proof is by using the KKT conditions of $\widetilde{\mathbb{P}}_t$ which is defined in Algorithm 1. □

[1] When deriving the dual problem, we make equivalent transformations for (3d) to facilitate our analysis. First, we change it to $\sigma'_{it}x_{it} \le \sigma'_{it}$, $\forall i, \forall t$. Afterwards, we replace it by a number of constraints [12], such as $\sum_i \sigma'_{it}x_{it} - \sigma'_{it}x_{it} \ge \sum_j \lambda_{jt} - \sigma'_{it}$, $\forall i, \forall t$, derived by (1a), (1b), and (1c). We do not write all such constraints; writing all of them is rather a mechanical process and we omit those details to save space.

4.2 Performance of Online Rounding

We prove $\mathsf{P}'(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\}) \leq r_2 \mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\})$ and show r_2 in this section. We consider each term in $\mathsf{P}'(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\})$ individually, and bound it by a corresponding constant times $\sum_t \sum_i a_{it}\widetilde{x}_{it}$ which is part of $\mathsf{P}'(\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\})$. We have the following theorem.

Theorem 2. *The non-switching cost in* $\mathsf{P}'(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\})$ *can be upper-bounded as follows, respectively:*

$$\sum_t \sum_i a_{it}\bar{x}_{it} \leq \max_{i,t} \frac{a_{it}}{\sigma'_{it}} r'_2 \sum_t \sum_i a_{it}\widetilde{x}_{it},$$
$$\sum_t \sum_i \sum_j c_{ijt}\bar{y}_{ijt} \leq \max_{i,j,t} \frac{c_{ijt}\sigma_{jt}}{V_j} r'_2 \sum_t \sum_i a_{it}\widetilde{x}_{it},$$
$$\sum_t \sum_i \sum_j e'_{ijt}\bar{z}_{ijt} \leq \max_{i,j,t} e'_{ijt} r'_2 \sum_t \sum_i a_{it}\widetilde{x}_{it}.$$

The switching cost in $\mathsf{P}'(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\})$ *can be upper-bounded as follows, respectively:*

$$\sum_t \sum_i b_i(\bar{x}_{it} - \bar{x}_{it-1})^+ \leq \max_{i,t} \frac{b_i}{\sigma'_{it}} r'_2 \sum_t \sum_i a_{it}\widetilde{x}_{it},$$
$$\sum_t \sum_i \sum_j d_{ij}(\bar{y}_{ijt} - \bar{y}_{ijt-1})^+ \leq \max_{i,j,t} \frac{d_{ij}\sigma_{jt}}{V_j} r'_2 \sum_t \sum_i a_{it}\widetilde{x}_{it},$$
$$\sum_t \sum_i \sum_j f'_{ijt}(\bar{z}_{ijt} - \bar{z}_{ijt-1})^+ \leq \max_{i,j,t} f'_{ijt} r'_2 \sum_t \sum_i a_{it}\widetilde{x}_{it}.$$

Joining all the above, we have $r_2 = r'_2\Big(\max_{i,t} a_{it} + \max_{i,j,t} c_{ijt}U_i + \max_{i,j,t} e'_{ijt} + \max_i b_i + \max_{i,j} d_{ij}U_i + \max_{i,j,t} f'_{ijt}\Big)$, *where* $r'_2 = \left(1 + \max_{i,t} \frac{\sigma'_{it}}{\sum_j \lambda_{jt}}\right) \max_{i,t} \frac{\sigma'_{it}}{a_{it}}$, *and* $\sigma'_{it} = U_i \frac{\max_j V_j}{\min_j \sigma_{jt}}$.

Proof. See Appendix A.3. The proof is by following the design of Algorithm 3 and the definition of $\mathbb{P}$. □

4.3 Performance of Learning Approximation

We prove $\mathsf{P}(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\}) \leq \mathsf{P}'(\{\bar{\mathbf{x}}_\mathbf{t}, \bar{\mathbf{y}}_\mathbf{t}, \bar{\mathbf{z}}_\mathbf{t}, \forall t\})$ and $\mathsf{P}'_{opt} \leq r_3 \mathsf{P}_{opt}$, and show r_3 in this section. We have the following two theorems.

Theorem 3. P *is upper-bounded by* P'*, due to the following fact:*

$$\sum_t \sum_i \sum_j \delta_{jt+1} h_{ij} \bar{z}_{ijt} \leq \sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}_{ijt} + \sum_t \sum_j \left(2\left(\delta_{jt} \max_i h_{ij}\right) \sum_i \left(\bar{z}_{ijt} - \bar{z}_{ijt-1}\right)^+\right).$$

Proof. See Appendix A.4. The proof is by following the design of Algorithm 3, and the definitions of $\mathbb{P}$ and $\mathbb{P}'$. □

Theorem 4. *We have* $r_3 = \left(1 + 3\left(\min_{i,j} V_j U_i\right)\left(\max_{i,j} \frac{\max_i h_{ij}}{f_{ij}}\right)\right)$.

Proof. See Appendix A.5. The proof is by following the definitions of $\mathbb{P}$ and $\mathbb{P}'$. □

5 EXTENSION

We can extend our algorithms and analysis to address the situation where, besides the clean traffic, the suspicious traffic for each time slot is also unknown before making the decision in each time slot.

To capture such uncertainty, we formulate a new problem $\mathbb{P}^\#$:

$$\begin{aligned} \min \quad & \mathsf{P}^\# = \mathsf{P} + \sum_t \sum_i \sum_j h_{ij}\left(\sigma_{jt+1} z_{ijt} - V_j y_{ijt}\right)^+ \\ & + \sum_t \sum_j p_j \left(\lambda_{jt+1} - \sum_i z_{ijt}\right)^+ \\ \text{s.t.} \quad & \text{(1a), (1d), (1e),} \end{aligned}$$

where in P we redefine $\sigma'_{ijt+1} = \sigma_{jt+1} g_{ij} + \delta_{jt+1} h_{ij}$, $\forall i, \forall j, \forall t$, without changing everything else; we also have the new input p_j, $\forall j$ denote the penalty of not diverting the flow j. Here, note that both σ_{jt+1} and δ_{jt+1} are with the time subscript $t+1$, rather than t, to reflect the fact of firstly making the decision and afterwards seeing the traffic to be filtered. Making decisions beforehand may miss some flows and/or provide insufficient resources for scrubbing, and therefore we introduce $\sum_t \sum_i \sum_j h_{ij}\left(\sigma_{jt+1} z_{ijt} - V_j y_{ijt}\right)^+$ and $\sum_t \sum_j p_j \left(\lambda_{jt+1} - \sum_i z_{ijt}\right)^+$ into the objective while removing the corresponding constraints. The latter means that the missed flows, if any, incur penalty, such as the SC provider's revenue loss. The former means that the unfiltered traffic, if any, travels to the destinations as well and incurs network footprint (note that, for each suspicious flow j, δ_{jt+1} only refers to the clean traffic identified and produced by the SC; it does not necessarily refer to all the actual clean traffic contained in σ_{jt+1}, because the SC may not be able to identify all the clean traffic in σ_{jt+1}).

To make our online algorithms and analysis applicable to $\mathbb{P}^\#$, we make a few key reformulations as follows. Having

$$\left(\sigma_{jt+1} z_{ijt} - V_j y_{ijt}\right)^+ \leq \sigma_{jt+1} z_{ijt},$$
$$\left(\lambda_{jt+1} - \sum_i z_{ijt}\right)^+ \leq \left(\lambda_{jt+1} - \sum_i z_{ijt+1}\right)^+ + \left(\sum_i z_{ijt+1} - \sum_i z_{ijt}\right)^+,$$

we define the problem $\mathbb{P}^\diamond$, where $\mathsf{P}^\# \leq \mathsf{P}^\diamond$:

$$\begin{aligned} \min \quad & \mathsf{P}^\diamond = \mathsf{P} + \sum_t \sum_i \sum_j h_{ij}\sigma_{jt+1} z_{ijt} + \sum_t \sum_j p_j w_{jt} \\ & + \sum_t \sum_j p_j \left(\sum_i z_{ijt} - \sum_i z_{ijt-1}\right)^+ \\ \text{s.t.} \quad & \text{(1a), (1d), (1e),} \\ & w_{jt} + \sum_i z_{ijt} \geq \lambda_{jt}, \forall j, \forall t, \qquad (4a) \\ & w_{jt} \geq 0, \qquad \forall j, \forall t. \qquad (4b) \end{aligned}$$

In $\mathbb{P}^\diamond$, we introduce the auxiliary variable w_{jt} and the auxiliary constraints (4a) and (4b) so that, except P and $\sum_t \sum_i \sum_j h_{ij}\sigma_{jt+1} z_{ijt}$, the decisions are in the same time slot as the corresponding inputs. We observe that if we treat $\mathbb{P}^\diamond$ as $\mathbb{P}$, then all our proposed algorithms and analysis apply, with straightforward adaptions. Just note that, in the analysis now, we also need to connect $\mathsf{P}^\diamond_{opt}$ and $\mathsf{P}^\#_{opt}$. In fact, analogous to the proof of Theorem 4, we can achieve it using $\left(\lambda_{jt+1} - \sum_i z_{ijt+1}\right)^+ \leq \left(\lambda_{jt+1} - \sum_i z_{ijt}\right)^+ + \left(\sum_i z_{ijt} - \sum_i z_{ijt+1}\right)^+$.

6 EXPERIMENTAL STUDY

6.1 Data and Settings

DDoS Traffic. We use the Booter traces that record more than 250 GB of real-world, UDP-based DDoS packets during 2 consecutive days in August 2013 [25]. The traffic rate varies dynamically, up to 5.48 Gbps. We consider 5 minutes as a single time slot, and aggregate the traffic rates accordingly. While in this dataset the attack sources from all over the world attack one target in the Netherlands, to limit our scope, we only consider the 5822 US attack sources (around 33% of all attack sources), and envisage 48 targets in the capital cites of the lower 48 states in the continental US, respectively.

Legitimate Traffic. For each DDoS flow in each time slot, we synthesize a legitimate flow so that they together constitute a suspicious flow. The legitimate flow originates from an IP address with the same prefix and thus from the same AS as the attack source. We control and vary a percentage, and set the volume of the legitimate flow as this percentage times the volume of the DDoS flow.

Traffic Paths. We map the IP addresses of the attack/legitimate sources and the targets to ASes [1]. Afterwards, leveraging the 2016 CAIDA dataset of the AS relationships with geographic annotations [5], we find the shortest AS path between each attack/legitimate

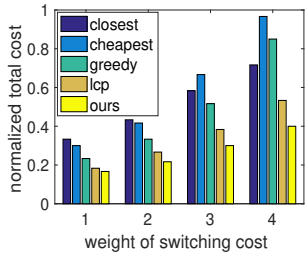

Figure 2: Impact of switching cost

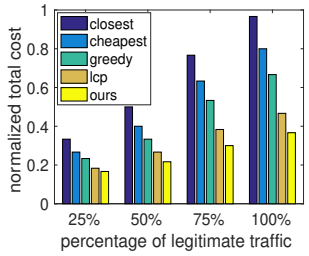

Figure 3:Impact of flow composition

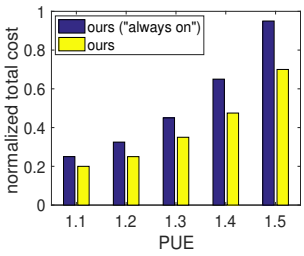

Figure 4: Benefit of turning off SCs

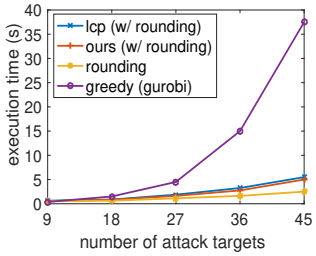

Figure 5: Comparison of execution time

source and each target, which we assume is the path of the traffic, and get the geo-locations of the involved AS links between the directly-connected ASes (as the dataset does not contain the geo-locations of the ASes themselves).

Scrubbing Centers. We use Level 3 [4] and Incapsula [3] which have 5 and 9 scrubbing centers, respectively, in the US at 11 unique locations all covered by the AS links [5]. For simplicity and without loss of generality, we assume 11 SCs with equal capacities at such locations; we assume one VM is sufficient to scrub one flow, and the total capacity of all SCs equals 1.2 times the number of flows.

Scrub Resource Price and Diversion Rule Price. Regarding the scrub resource price, we adopt the dynamic, hourly wholesale electricity prices from different Regional Transmission Organizations (RTOs) in the US [6]. We know the geo-coordinates of each SC via its co-located AS link [5], and thus we can map each SC to its geographically closest RTO and use the corresponding prices. Regarding the diversion rule price, we find that it is comparable to the wholesale electricity price per time slot [7].

Unit Switching Costs. Regarding the unit switching costs of the scrub resources and the diversion rules, we consider the VM start-up time and the BGP convergence time, respectively, both of which can range from seconds to minutes today [18, 22]. This captures the SCs' performance degradation or even service unavailability. Rather than using different metrics, in addition to the start-up/convergence time, as the switching costs, we vary the weights associated to the switching costs to mimic a variety of metrics.

Algorithms. We compare 5 algorithms: "closest" refers to today's industrial practice [4] that diverts suspicious traffic to the SC that is the closest along the AS path to each customer; "cheapest" refers to the method that diverts suspicious traffic to the SC that has the cheapest resources; "greedy" refers to the method that solves the one-shot slice of our original problem at every time slot while ignoring the switching cost; "lcp" refers to the state-of-the-art "lazy capacity provisioning" algorithm for solving online convex optimization problems with the switching cost [22]; "ours" refers to our own approach. All algorithms are online; we have no offline optimum to show, as it takes an unacceptably long time to solve even a relatively small-scale instance of our original integer program by today's most advanced solvers (e.g.,gurobi [2]).

6.2 Evaluation Results

Fig. 2 compares the long-term total cost of the algorithms when the weight associated to the switching cost varies. As the weight grows, the total (weighted) cost increases. ours achieves the best result, up to 50%, 58%, 53%, and 26% less total cost than closest, cheapest, greedy, and lcp, respectively. Except ours and lcp, the other algorithms neglect the switching cost and thus behave worse. greedy takes care of all types of operational costs while closest and cheapest only consider part of them. Thus, greedy is often better. Note that closest gradually becomes the best of the three, as the weight goes greater; this is because closest does not change SCs for the flows and incurs no switching cost of diversion rules.

Fig. 3 demonstrates the total cost over time as the amount of the legitimate traffic increases, until it equals the volume of the malicious traffic. ours remains the best, up to 62% better than all the rest, since it is the only one that considers the network footprint of the legitimate traffic when making online decisions. greedy is better than closest and cheapest, as explained above; closest is always the worst, because as the legitimate traffic grows, the total suspicious traffic also increases, and always using the SC closest to customers can incur large network footprint before the traffic is scrubbed.

Fig. 4 visualizes the benefit of allowing switching off entire SCs to save the non-IT energy. The total cost in this figure also includes the operational cost and the switching cost of SCs, besides the cost of the scrub resources inside SCs. We choose PUEs in the range of $1.1 \sim 1.5$ to capture the worse power efficiency of small data centers; the larger the PUE is, the more non-IT energy it can save if we shut the SC down. Compared to always having entire SCs on, ours saves $20\% \sim 26\%$ total cost due to the non-IT energy reduction.

Fig. 5 illustrates the execution time of the algorithms in a single time slot, as we increase the number of attack targets. We exclude closest and cheapest as their execution time is trivial. greedy invokes the gurobi solver to solve our integer program at each time slot, and the execution time grows very fast as the problem becomes larger. Solving the fractional problem and then rounding the solutions, like lcp and ours, is more scalable, which takes up to 5 seconds. The execution time of rounding grows mildly itself.

7 RELATED WORK

We summarize existing research in three categories, and highlight their insufficiency compared to our work in this paper.

Clouds and Data Centers for DDoS Mitigation. Yu et al. [28] dynamically allocated resources in a single cloud to filter DDoS traffic. Fayaz et al. [14] used network functions to scrub DDoS traffic by solving the data center, server, and VM allocation problems. Zilberman et al. [30] compared different strategies for placing SCs across the Internet in terms of network footprint, link load, and network latency. Liu et al. [23] enabled victims to redirect the DDoS traffic to clouds with victim-chosen bandwidth allocation policies. Somani et al. [27] found cloud's ability of absorbing DDoS could be

compromised by the heavy resource usage of clients under attack, and recruited clients' resources during attack for attack mitigation.

Online Cloud Resource Allocation with Switching Cost. Lin et al. [22] designed the "lazy capacity provisioning" online algorithm for allocating servers with switching costs in data centers. Jiao et al. [19, 20] developed regularization-based online algorithms for the joint control of edge clouds and internal servers, and for the en route resource allocation in hierarchical clouds, respectively, both with switching costs. Pu et al. [24] used regularization as part of the study of resource allocation, content caching, and request distribution in an online manner in cloud radio access networks. Zhang et al. [29] and Fei et al. [15] predicted the demand via online gradient decent and "following the regularized leader", respectively, and with such predictions, used "ski rental" based algorithms, bin packing, and primal-dual algorithms for VM/traffic management.

Competitive Analysis and Online Convex Optimization. The theory/algorithm community has studied the online optimization problem with switching costs as well. Buchbinder et al. were the first to embed regularization into competitive analysis for the covering problem [10], and designed a primal-dual-based online algorithm against the drifting offline optimum (i.e., decision changes were constrained) for online learning [11]. Andrew et al. [9] identified the incompatibility between the sublinear regret in online learning and the constant competitive ratio in competitive analysis. Shi et al. [26] proposed algorithms to achieve the optimal competitive ratios for problems with affine policies. Chen et al. [13] focused on the high-dimension decision space and proposed online balanced decent to improve both the competitive ratio and the regret.

Previous research has never investigated the problem studied in this paper, and previous solution techniques are insufficient for addressing our problem. [28], [14], and [30] seem the closest to our work in terms of the problem space; however, they either consider a single cloud only, or neglect the redirection rules and the switching costs, with no performance guarantees for their algorithms. [23] and [27] focus on the system aspects of a similar scenario, and lack theoretical/algorithmic insights. [22] and [20] consider problems simpler than ours, because they (i) only allocate resources without making decisions for workload or flow distribution, (ii) only consider fractional decisions, and (iii) assume all inputs in each time slot are observable, without learning. [19] and [24] rely on randomized rounding algorithms, and do not address online learning. [29] and [15] adopt online learning to predict their inputs, and solve the online problem based on the predictions. Our approach follows a different philosophy compared to theirs: (i) we do not rely on predictions, but directly approximate the learning component in our problem; (ii) we do not use "regret" and compare to the offline optimum under the best static predictions of inputs, but instead we compare to the offline optimum under the actual, real inputs. Regarding the algorithmic literatures, all of them mentioned above do not accommodate integer variables. Beisdes, [10] and [26] do not consider learning; [11] can handle the dynamic resource price in the objective, but not the fluctuating demand in the constraints; [11] and [13] are against drifting offline optimums, and [9] is against the static regret, which makes less sense in our case. In contrast, we propose the combination of an online algorithm and two deterministic rounding algorithms to solve an integer program, against the offline optimum in terms of the competitive ratio.

8 CONCLUSION

In this paper, we investigate the online optimization of the joint scheduling of the traffic diversion rules in the networks and the scrubbing resources in the clouds to inspect unpredictable time-varying traffic which are only partially observable when making the scheduling decisions in each time slot. We design an online algorithmic framework inspired by the fusion of three techniques: online learning approximation, regularization, and dependent rounding. We formally prove the competitive ratio of our approach, and also conduct extensive evaluations to exhibit its practical advantages.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation (Grant No. CNS 1703014 and CNS 1717493), the Technological Innovation Major Projects of Hubei Province (Grant No. 2017AAA125), the Science and Technology Program of Wuhan City (Grant No. 2018010401011288), the National Natural Science Foundation of China (Grant No. U1711265), and the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (Grant No. 2017ZT07X355). Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

[1] 2018. Free IP address to ASN database. https://iptoasn.com/.
[2] 2018. Gurobi Optimization - The State-of-the-Art Mathematical Programming Solver. http://www.gurobi.com.
[3] 2018. Incapsula Global Network Map | Instantly Localizing Your Content. https://www.incapsula.com/incapsula-global-network-map.html.
[4] 2018. Level 3® DDoS Mitigation. http://www.level3.com/~/media/files/brochures/en_secur_br_ddos_mitigation.pdf.
[5] 2018. The CAIDA AS Relationships (Geo) Dataset. http://www.caida.org/data/as-relationships-geo/.
[6] 2018. U.S. Energy Information Administration (EIA). https://www.eia.gov/electricity/wholesale/.
[7] 2018. What does a BGP route cost? http://bill.herrin.us/network/bgpcost.html.
[8] Alexander A Ageev and Maxim I Sviridenko. 2004. Pipage Rounding: A New Method of Constructing Algorithms with Proven Performance Guarantee. *Journal of Combinatorial Optimization* 8, 3 (2004), 307–328.
[9] Lachlan Andrew, Siddharth Barman, Katrina Ligett, Minghong Lin, Adam Meyerson, Alan Roytman, and Adam Wierman. 2013. A tale of two metrics: Simultaneous bounds on competitiveness and regret. In *ACL COLT*.
[10] Niv Buchbinder, Shahar Chen, and Joseph Seffi Naor. 2014. Competitive Analysis via Regularization. In *ACM-SIAM SODA*.
[11] Niv Buchbinder, Shahar Chen, Joshep Seffi Naor, and Ohad Shamir. 2012. Unified Algorithms for Online Learning and Competitive Analysis. In *ACL COLT*.
[12] Robert D Carr, Lisa K Fleischer, Vitus J Leung, and Cynthia A Phillips. 2000. Strengthening Integrality Gaps for Capacitated Network Design and Covering Problems. In *ACM-SIAM SODA*.
[13] Niangjun Chen, Gautam Goel, and Adam Wierman. 2018. Smoothed Online Convex Optimization in High Dimensions via Online Balanced Descent. In *ACL COLT*.
[14] Seyed Kaveh Fayaz, Yoshiaki Tobioka, Vyas Sekar, and Michael Bailey. 2015. Bohatei: Flexible and Elastic DDoS Defense.. In *USENIX Security*.
[15] Xincai Fei, Fangming Liu, Hong Xu, and Hai Jin. 2018. Adaptive VNF Scaling and Flow Routing with Proactive Demand Prediction. In *IEEE INFOCOM*.
[16] Mohan Ganeshalingam, Arman Shehabi, and Louis-Benoit Desroches. 2017. Shining a Light on Small Data Centers in the U.S. *Lawrence Berkeley National Laboratory Report* (2017).
[17] Vasileios Giotsas, Philipp Richter, Georgios Smaragdakis, Anja Feldmann, Christoph Dietzel, and Arthur Berger. 2017. Inferring BGP Blackholing Activity in the Internet. In *ACM IMC*.
[18] Thomas Holterbach, Stefano Vissicchio, Alberto Dainotti, and Laurent Vanbever. 2017. Swift: Predictive Fast Reroute. In *ACM SIGCOMM*.
[19] Lei Jiao, Lingjun Pu, Lin Wang, Xiaojun Lin, and Jun Li. 2018. Multiple Granularity Online Control of Cloudlet Networks for Edge Computing. In *IEEE SECON*.
[20] Lei Jiao, Antonia Tulino, Jaime Llorca, Yue Jin, and Alessandra Sala. 2017. Smoothed Online Resource Allocation in Multi-tier Distributed Cloud Networks.

IEEE/ACM Transactions on Networking 25, 4 (2017), 2556–2570.
[21] Mattijs Jonker, Anna Sperotto, Roland van Rijswijk-Deij, Ramin Sadre, and Aiko Pras. 2016. Measuring the Adoption of DDoS Protection Services. In *ACM IMC*.
[22] Minghong Lin, Adam Wierman, Lachlan LH Andrew, and Eno Thereska. 2013. Dynamic Right-sizing for Power-proportional Data Centers. *IEEE/ACM Transactions on Networking* 21, 5 (2013), 1378–1391.
[23] Zhuotao Liu, Hao Jin, Yih-Chun Hu, and Michael Bailey. 2016. MiddlePolice: Toward Enforcing Destination-defined Policies in the Middle of the Internet. In *ACM CCS*.
[24] Lingjun Pu, Lei Jiao, Xu Chen, Lin Wang, Qinyi Xie, and Jingdong Xu. 2018. Online resource allocation, content placement and request routing for cost-efficient edge caching in cloud radio access networks. *IEEE Journal on Selected Areas in Communications* 36, 8 (2018), 1751–1767.
[25] J.J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Zambenedetti Granville, and A. Pras. 2015. Booters - An Analysis of DDoS-as-a-service Attacks. In *IFIP/IEEE IM*.
[26] Ming Shi, Xiaojun Lin, Sonia Fahmy, and Dong-Hoon Shin. 2018. Competitive Online Convex Optimization with Switching Costs and Ramp Constraints. In *IEEE INFOCOM*.
[27] Gaurav Somani, Manoj Singh Gaur, Dheeraj Sanghi, Mauro Conti, and Muttukrishnan Rajarajan. 2018. Scale Inside-out: Rapid Mitigation of Cloud DDoS Attacks. *IEEE Transactions on Dependable and Secure Computing* 15, 6 (2018), 959–973.
[28] Shui Yu, Yonghong Tian, Song Guo, and Dapeng Oliver Wu. 2014. Can We Beat DDoS Attacks in Clouds? *IEEE Transactions on Parallel and Distributed Systems* 25, 9 (2014), 2245–2254.
[29] Xiaoxi Zhang, Chuan Wu, Zongpeng Li, and Francis CM Lau. 2017. Proactive VNF Provisioning with Multi-timescale Cloud Resources: Fusing Online Learning and Online Optimization. In *IEEE INFOCOM*.
[30] Polina Zilberman, Rami Puzis, and Yuval Elovici. 2017. On Network Footprint of Traffic Inspection and Filtering at Global Scrubbing Centers. *IEEE Transactions on Dependable and Secure Computing* 14, 5 (2017), 521–534.

A PROOFS

A.1 Proof of Lemma 1

We write the Karush-Kuhn-Tucker (KKT) conditions of $\widetilde{\mathbb{P}}_t, \forall t$ below, which are sufficient and necessary to characterize $\{\widetilde{\mathbf{x}}_\mathbf{t}, \widetilde{\mathbf{y}}_\mathbf{t}, \widetilde{\mathbf{z}}_\mathbf{t}, \forall t\}$ and the dual variables of $\widetilde{\mathbb{P}}_t$. Note that these KKT conditions are for $\widetilde{\mathbb{P}}_t$ (which is defined in Algorithm 1), not for $\mathbb{P}$ or $\mathbb{P}'$.

$$a_{it} - U_i\widetilde{\alpha}_{it} + \frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} + \sigma'_{it}\widetilde{\tau}_{it} - \sigma'_{it}\textstyle\sum_i \widetilde{\tau}_{it} = 0, \forall i, \tag{5a}$$

$$c_{ijt} + \widetilde{\alpha}_{it} - V_j\widetilde{\beta}_{ijt} + \frac{d_{ij}}{\eta_i}\ln\frac{\widetilde{y}_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon} = 0, \forall i, \forall j, \tag{5b}$$

$$e'_{ijt} + \sigma_{jt}\widetilde{\beta}_{ijt} - \widetilde{\gamma}_{jt} + \frac{f'_{ijt}}{\eta}\ln\frac{\widetilde{z}_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon} - \widetilde{\omega}_{ijt} = 0, \forall i, \forall j, \tag{5c}$$

$$\widetilde{\alpha}_{it}(U_i\widetilde{x}_{it} - \textstyle\sum_j \widetilde{y}_{ijt}) = 0, \forall i, \tag{5d}$$

$$\widetilde{\beta}_{ijt}(V_j\widetilde{y}_{ijt} - \sigma_{jt}\widetilde{z}_{ijt}) = 0, \forall i, \forall j, \tag{5e}$$

$$\widetilde{\gamma}_{jt}(\textstyle\sum_i \widetilde{z}_{ijt} - \lambda_{jt}) = 0, \forall j, \tag{5f}$$

$$\widetilde{\tau}_{it}\left(\textstyle\sum_i \sigma'_{it}x_{it} - \sigma'_{it}x_{it} - \left(\sum_j \lambda_{jt} - \sigma'_{it}\right)\right) = 0, \forall i, \tag{5g}$$

$$\widetilde{\omega}_{ijt}\widetilde{z}_{ijt} = 0, \forall i, \forall j, \tag{5h}$$

$$\text{all primal and dual variables} \geq 0. \tag{5i}$$

By (5a), (5b), (5c), and (5i), it can be verified that the solution constructed in Lemma 1 satisfies all the constraints of the problem $\mathbb{D}$. These KKT conditions are used in other proofs as well.

A.2 Proof of Theorem 1

First, we bound the non-switching cost.

$$\begin{aligned}
&\textstyle\sum_t\sum_i a_{it}\widetilde{x}_{it} + \sum_t\sum_i\sum_j c_{ijt}\widetilde{y}_{ijt} + \sum_t\sum_i\sum_j e'_{ijt}\widetilde{z}_{ijt}\\
=\ &\textstyle\sum_t\sum_i\left(U_i\widetilde{\alpha}_{it} + \sigma'_{it}\sum_i\widetilde{\tau}_{it} - \sigma'_{it}\widetilde{\tau}_{it} - \frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon}\right)\widetilde{x}_{it}\\
&+\textstyle\sum_t\sum_i\sum_j(V_j\widetilde{\beta}_{ijt} - \widetilde{\alpha}_{it} - \frac{d_{ij}}{\eta_i}\ln\frac{\widetilde{y}_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon})\widetilde{y}_{ijt}\\
&+\textstyle\sum_t\sum_i\sum_j(\widetilde{\gamma}_{jt} + \widetilde{\omega}_{ijt} - \sigma_{jt}\widetilde{\beta}_{ijt} - \frac{f'_{ijt}}{\eta}\ln\frac{\widetilde{z}_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon})\widetilde{z}_{ijt} \qquad (6a)\\
=\ &\textstyle\sum_t\sum_j\lambda_{jt}\widetilde{\gamma}_{jt} + \sum_t\sum_i\left(\sum_j\lambda_{jt} - \sigma'_{it}\right)\widetilde{\tau}_{it}\\
&-\textstyle\sum_t\sum_i\widetilde{x}_{it}\frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} - \sum_t\sum_i\sum_j\widetilde{y}_{ijt}\frac{d_{ij}}{\eta_i}\ln\frac{\widetilde{y}_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon}\\
&-\textstyle\sum_t\sum_i\sum_j\widetilde{z}_{ijt}\frac{f'_{ijt}}{\eta}\ln\frac{\widetilde{z}_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon} \qquad (6b)\\
\leq\ &\mathrm{D}. \qquad (6c)
\end{aligned}$$

We reach (6a) by following (5a)∼(5c). Then, we reach (6b) by following (5d)∼(5h). Finally, we have (6c) due to $\sum_t \widetilde{x}_{it}\frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} \geq 0, \forall i$; and $\sum_t \widetilde{y}_{ijt}\frac{d_{ij}}{\eta_i}\ln\frac{\widetilde{y}_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon} \geq 0$, $\sum_t \widetilde{z}_{ijt}\frac{f'_{ijt}}{\eta}\ln\frac{\widetilde{z}_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon} \geq 0, \forall i, \forall j$. As an example, we show the following, and the others can be shown analogously:

$$\begin{aligned}
&\textstyle\sum_t \widetilde{x}_{it}\frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon}\\
=\ &\textstyle\sum_t(\widetilde{x}_{it}+\varepsilon)\frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} - \sum_t\varepsilon\frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon}\\
\geq\ &\textstyle(\sum_t(\widetilde{x}_{it}+\varepsilon))\ln\frac{\sum_t(\widetilde{x}_{it}+\varepsilon)}{\sum_t(\widetilde{x}_{it-1}+\varepsilon)} + (\widetilde{x}_{i0}+\varepsilon)\ln\frac{\widetilde{x}_{i0}+\varepsilon}{\widetilde{x}_{iT}+\varepsilon} \qquad (7a)\\
\geq\ &\textstyle\sum_t(\widetilde{x}_{it}+\varepsilon) - \sum_t(\widetilde{x}_{it-1}+\varepsilon) + \widetilde{x}_{i0} - \widetilde{x}_{iT} \qquad (7b)\\
=\ &0,
\end{aligned}$$

where (7a) follows from (8a), and (7b) follows from (8b). We also use $\widetilde{x}_{it} = 0, \forall i, \forall t \leq 0$ by definition. (8a) and (8b) are two facts:

$$(\textstyle\sum_n p_n)\ln\frac{\sum_n p_n}{\sum_n q_n} \leq \sum_n p_n \ln\frac{p_n}{q_n}, \forall p, q > 0, \tag{8a}$$

$$p - q \leq p\ln\frac{p}{q}, \forall p, q > 0. \tag{8b}$$

Second, we bound the switching cost. Particularly, we bound $\sum_t\sum_i b_i(\widetilde{x}_{it} - \widetilde{x}_{it-1})^+$ as an example, and omit the details for bounding $\sum_t\sum_i\sum_j d_{ij}(\widetilde{y}_{ijt} - \widetilde{y}_{ijt-1})^+$ and $\sum_t\sum_i\sum_j f'_{ijt}(\widetilde{z}_{ijt} - \widetilde{z}_{ijt-1})^+$. In P' we have the term $\sum_t\sum_i b_i u_{it}$; with (3a) and (3e), we know the optimal value of u_{it} is $\widetilde{u}_{it} = (\widetilde{x}_{it} - \widetilde{x}_{it-1})^+, \forall i, \forall t$.

$$\begin{aligned}
&\textstyle\sum_t\sum_i b_i(\widetilde{x}_{it} - \widetilde{x}_{it-1})^+\\
=\ &\textstyle\sum_t\sum_{i\in\mathcal{I}_t} b_i(\widetilde{x}_{it} - \widetilde{x}_{it-1}) \qquad (9a)\\
=\ &\textstyle\sum_t\sum_{i\in\mathcal{I}_t} b_i((\widetilde{x}_{it}+\varepsilon) - (\widetilde{x}_{it-1}+\varepsilon)) \qquad (9b)\\
\leq\ &\textstyle\sum_t\sum_{i\in\mathcal{I}_t} b_i(\widetilde{x}_{it}+\varepsilon)\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} \qquad (9c)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}_t}\frac{b_i}{\eta}\ln\frac{\widetilde{x}_{it}+\varepsilon}{\widetilde{x}_{it-1}+\varepsilon} \qquad (9d)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}_t}\left(U_i\widetilde{\alpha}_{it} + \sigma'_{it}\sum_i\widetilde{\tau}_{it}\right) \qquad (9e)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}'_t}\left(U_iV_j\widetilde{\beta}_{ijt} - U_i\frac{d_{ij}}{\eta_i}\ln\frac{\widetilde{y}_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon}\right.\\
&\left.+\sigma'_{it}\textstyle\sum_i\widetilde{\tau}_{it}\right) \qquad (9f)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}'_t}\left(U_iV_j\widetilde{\beta}_{ijt} + \sigma'_{it}\sum_i\widetilde{\tau}_{it}\right) \qquad (9g)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}''_t}\left(\frac{U_iV_j}{\sigma_{jt}}\left(\widetilde{\gamma}_{jt} - \frac{f'_{ijt}}{\eta}\ln\frac{\widetilde{z}_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon} + \widetilde{\omega}_{ijt}\right)\right.\\
&\left.+\sigma'_{it}\textstyle\sum_i\widetilde{\tau}_{it}\right) \qquad (9h)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}''_t}\left(\sigma'_{it}\widetilde{\gamma}_{jt} + \sigma'_{it}\sum_i\widetilde{\tau}_{it}\right) \qquad (9i)\\
\leq\ &\textstyle\eta(1+\varepsilon)\sum_t\sum_{i\in\mathcal{I}''_t}\left(\sigma'_{it}\lambda_{jt}\widetilde{\gamma}_{jt} + \sigma'_{it}\sum_i\left(\sum_j\lambda_{jt} - \sigma'_{it}\right)\widetilde{\tau}_{it}\right) \qquad (9j)\\
\leq\ &\textstyle\left((1+\varepsilon)\ln\left(1+\frac{1}{\varepsilon}\right)\max_j V_j\sum_i U_i\right)\mathrm{D}. \qquad (9k)
\end{aligned}$$

We get (9a) as we change the index set to $\mathcal{I}_t = \{i|\widetilde{x}_{it} > \widetilde{x}_{it-1}\}$. We have (9b) and (9c) by following (8b). We reach (9d) as $\widetilde{x}_{it} \leq 1, \forall i, \forall t$. (9e) follows from (5a). Considering the index set $\mathcal{I}'_t = \mathcal{I}_t \cap \{i|\widetilde{\alpha}_{it} > \widetilde{\alpha}_{it-1}\}$, we follow (5b) to get (9f). We next reach

(9g) because $\sum_t U_i \frac{d_{ij}}{\eta_i} \ln \frac{\widetilde{y}_{ijt}+\varepsilon}{\widetilde{y}_{ijt-1}+\varepsilon} = U_i \frac{d_{ij}}{\eta_i} \ln\left(1+\frac{\widetilde{y}_{ijT}}{\varepsilon}\right) \geq 0$. Considering the index set $\mathcal{I}_t'' = \mathcal{I}_t' \cap \{i | \widetilde{\beta}_{ijt} > \widetilde{\beta}_{ijt-1}\}$, we follow (5c) to get (9h). Then, note that we have $\sum_t \frac{f'_{ijt}}{\eta} \ln \frac{\widetilde{z}_{ijt}+\varepsilon}{\widetilde{z}_{ijt-1}+\varepsilon} \geq \frac{\min_t f'_{ijt}}{\eta} \ln\left(1+\frac{\widetilde{z}_{ijT}}{\varepsilon}\right) \geq 0$, and $\widetilde{w}_{ijt} = 0$. The latter is further due to the following. With (5d), $\widetilde{\alpha}_{it} > \widetilde{\alpha}_{it-1} \geq 0$, and $\widetilde{x}_{it} > \widetilde{x}_{it-1} \geq 0$, we have at least one j such that $\widetilde{y}_{ijt} > 0$, because otherwise we would have $\sum_j \widetilde{y}_{ijt} = 0$, contradicting $\sum_j \widetilde{y}_{ijt} = U_i \widetilde{x}_{it} > 0$; with (5e) and $\widetilde{\beta}_{ijt} > \widetilde{\beta}_{ijt-1} \geq 0$, we have $\widetilde{z}_{ijt} = \frac{V_j}{\sigma_{jt}} \widetilde{y}_{ijt} > 0$; with (5h), we finally have $\widetilde{\omega}_{ijt} = 0$. Based on this, we reach (9i). Finally, we reach (9j) because λ_{jt} and $\sum_j \lambda_{jt} - \sigma'_{it}$ are integers (note $\sigma'_{it} = U_i \frac{\max_j V_j}{\min_j \sigma_{jt}} = U_i \max_j V_j$, where U_i, V_j, and σ_{jt} are integers). If both of them are no less than 1, then (9j) naturally holds; if either of them is no greater than 0, then note that D will also change accordingly and as a result, (9j) still holds.

A.3 Proof of Theorem 2

First, we bound the non-switching cost. We have the following:

$$\begin{aligned}
&\sum_t \sum_i a_{it} \bar{x}_{it} \\
&\leq \max_{i,t} \tfrac{a_{it}}{\sigma'_{it}} \sum_t \sum_i \sigma'_{it} \bar{x}_{it} \\
&\leq \max_{i,t} \tfrac{a_{it}}{\sigma'_{it}} \left(\sum_t \sum_{i \in \mathcal{I} \setminus \{i'\}} \sigma'_{it} \widetilde{x}_{it} + \sum_t \sigma'_{i't} \lceil \widetilde{x}_{i't} \rceil \right) && \text{(10a)} \\
&\leq \max_{i,t} \tfrac{a_{it}}{\sigma'_{it}} \left(\sum_t \sum_i \sigma'_{it} \widetilde{x}_{it} + \max_{i,t} \tfrac{\sigma'_{it}}{\sum_j \lambda_{jt}} \sum_t \sum_j \lambda_{jt} \right) && \text{(10b)} \\
&\leq \max_{i,t} \tfrac{a_{it}}{\sigma'_{it}} \left(\sum_t \sum_i \sigma'_{it} \widetilde{x}_{it} + \max_{i,t} \tfrac{\sigma'_{it}}{\sum_j \lambda_{jt}} \sum_t \sum_i \sigma'_{it} \widetilde{x}_{it} \right) && \text{(10c)} \\
&\leq \max_{i,t} \tfrac{a_{it}}{\sigma'_{it}} r'_2 \sum_t \sum_i a_{it} \widetilde{x}_{it}. && \text{(10d)}
\end{aligned}$$

$$\begin{aligned}
&\sum_t \sum_i \sum_j c_{ijt} \bar{y}_{ijt} \\
&\leq \max_{i,j,t} \tfrac{c_{ijt}\sigma_{jt}}{V_j} \sum_t \sum_i \sum_j \tfrac{V_j}{\sigma_{jt}} \bar{y}_{ijt} \\
&\leq \max_{i,j,t} \tfrac{c_{ijt}\sigma_{jt}}{V_j} \sum_t \sum_i \left(\tfrac{\max_j V_j}{\min_j \sigma_{jt}} \left(\sum_{j \in \mathcal{J} \setminus \{j'\}} \widetilde{y}^*_{ijt} + \left\lceil \widetilde{y}^*_{ij't} \right\rceil \right) \right) && \text{(11a)} \\
&\leq \max_{i,j,t} \tfrac{c_{ijt}\sigma_{jt}}{V_j} \sum_t \sum_i \sigma'_{it} \bar{x}_{it} && \text{(11b)} \\
&\leq \max_{i,j,t} \tfrac{c_{ijt}\sigma_{jt}}{V_j} r'_2 \sum_t \sum_i a_{it} \widetilde{x}_{it}. && \text{(11c)}
\end{aligned}$$

$$\begin{aligned}
&\sum_t \sum_i \sum_j e'_{ijt} \bar{z}_{ijt} \\
&\leq \max_{i,j,t} e'_{ijt} \sum_t \sum_i \sum_j \bar{z}_{ijt} \\
&\leq \max_{i,j,t} e'_{ijt} \sum_t \sum_j \left(\sum_{i \in \mathcal{I} \setminus \{i'\}} \widetilde{z}^{**}_{ijt} + \left\lceil \widetilde{z}^{**}_{i'jt} \right\rceil \right) && \text{(12a)} \\
&\leq \max_{i,j,t} e'_{ijt} \sum_t \sum_i \sum_j \tfrac{V_j}{\sigma_{jt}} \bar{y}_{ijt} && \text{(12b)} \\
&\leq \max_{i,j,t} e'_{ijt} \sum_t \sum_i \sigma'_{it} \bar{x}_{ijt} && \text{(12c)} \\
&\leq \max_{i,j,t} e'_{ijt} r'_2 \sum_t \sum_i a_{it} \widetilde{x}_{it}. && \text{(12d)}
\end{aligned}$$

(10a), (11a), and (12a) follow from Lines 13, 15, and 24 of Algorithm 3. All values are rounded into integers using either Line 13 or 15, except i' or j' that indexes the last single value rounded up in Line 24. Note if no single value is left, then (10d), (11c), and (12d) still hold. (10b) introduces λ_{jt}, so that (10c) follows from Constraints (1a), (1b), and (1c). (11b) follows from Constraint (1a), given U_i and $\bar{x}_{it}$ are integers. (12b) follows from Constraint (1b), given $\sigma_{jt} \leq V_j$, $\widetilde{z}^{**}_{ijt} \leq 1$, and the integral $\bar{y}_{ijt}$; (12c) follows from Constraint (1a). Finally, (10d) constructs our target term $\sum_t \sum_i a_{it} \widetilde{x}_{it}$; (11c) and (12d) follow from the entire (10a) through (10d).

Second, we bound the switching cost. Particularly, we bound $\sum_t \sum_i b_i (\bar{x}_{it} - \bar{x}_{it-1})^+$ in (13a), using (10a) ~ (10d):

$$\begin{aligned}
&\sum_t \sum_i b_i (\bar{x}_{it} - \bar{x}_{it-1})^+ \\
&\leq \sum_t \sum_i b_i \bar{x}_{it} \\
&\leq \max_{i,t} \tfrac{b_i}{\sigma'_{it}} \sum_t \sum_i \sigma'_{it} \bar{x}_{it} \\
&\leq \max_{i,t} \tfrac{b_i}{\sigma'_{it}} r'_2 \sum_t \sum_i a_{it} \widetilde{x}_{it}. && \text{(13a)}
\end{aligned}$$

We can bound $\sum_t \sum_i \sum_j d_{ij} (\bar{y}_{ijt} - \bar{y}_{ijt-1})^+$ and $\sum_t \sum_i \sum_j f'_{ijt} (\bar{z}_{ijt} - \bar{z}_{ijt-1})^+$ analogously, and we omit the details.

A.4 Proof of Theorem 3

$$\begin{aligned}
&\sum_t \sum_i \sum_j \delta_{jt+1} h_{ij} \bar{z}_{ijt} \\
&= \sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}_{ijt-1} && \text{(14a)} \\
&\leq \sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}_{ijt} \\
&\quad + \sum_t \sum_i \sum_j \delta_{jt} h_{ij} \left| \bar{z}_{ijt} - \bar{z}_{ijt-1} \right| && \text{(14b)} \\
&\leq \sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}_{ijt} \\
&\quad + \sum_t \sum_j \left(\left(\delta_{jt} \max_i h_{ij} \right) \sum_i \left| \bar{z}_{ijt} - \bar{z}_{ijt-1} \right| \right) && \text{(14c)} \\
&\leq \sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}_{ijt} \\
&\quad + \sum_t \sum_j \left(2 \left(\delta_{jt} \max_i h_{ij} \right) \sum_i \left(\bar{z}_{ijt} - \bar{z}_{ijt-1} \right)^+ \right). && \text{(14d)}
\end{aligned}$$

We have (14a) because by definition we have $\delta_{jT+1} = 0$ and $\bar{z}_{ij0} = 0$. We introduce the absolute value in (14b), and change it to (14c). We reach (14d) because for every j, there is one and only one i where $\bar{z}_{ijt} = 1$ at every t, according to Line 24 of Algorithm 3 (if no variable is left, one can choose an arbitrary variable to set to 1); as such i may be different for $t-1$ and t, the coefficient becomes 2.

A.5 Proof of Theorem 4

We simplify the notations a bit. Let us rewrite P and P′:

$$\begin{aligned}
\mathsf{P} &= \mathsf{Q} + \textstyle\sum_t \sum_i \sum_j \delta_{jt+1} h_{ij} z_{ijt} + \sum_t \sum_i \sum_j f_{ij} (z_{ijt} - z_{ijt-1})^+, \\
\mathsf{P}' &= \mathsf{Q} + \textstyle\sum_t \sum_i \sum_j \delta_{jt} h_{ij} z_{ijt} + \sum_t \sum_i \sum_j f'_{ijt} (z_{ijt} - z_{ijt-1})^+,
\end{aligned}$$

where $f'_{ijt} = f_{ij} + 2\delta_{jt} \max_i h_{ij}$, and Q is the rest term which involves the variables $\{\mathbf{x_t}, \mathbf{y_t}, \mathbf{z_t}, \forall t\}$ and appears in both P and P′. Assuming $\{\bar{\mathbf{x}}^*_\mathbf{t}, \bar{\mathbf{y}}^*_\mathbf{t}, \bar{\mathbf{z}}^*_\mathbf{t}, \forall t\}$ are offline optimal solutions to $\mathbb{P}$, we get

$$\begin{aligned}
\mathsf{P}'_{opt} &\leq \mathsf{Q}(\{\bar{\mathbf{x}}^*_\mathbf{t}, \bar{\mathbf{y}}^*_\mathbf{t}, \bar{\mathbf{z}}^*_\mathbf{t}, \forall t\}) + \textstyle\sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}^*_{ijt} \\
&\quad + \textstyle\sum_t \sum_i \sum_j f'_{ijt} (\bar{z}^*_{ijt} - \bar{z}^*_{ijt-1})^+ && \text{(15a)} \\
&\leq \mathsf{Q}(\{\bar{\mathbf{x}}^*_\mathbf{t}, \bar{\mathbf{y}}^*_\mathbf{t}, \bar{\mathbf{z}}^*_\mathbf{t}, \forall t\}) + \textstyle\sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}^*_{ijt-1} \\
&\quad + \textstyle\sum_t \sum_i \sum_j (f'_{ijt} + \delta_{jt} h_{ij}) (\bar{z}^*_{ijt} - \bar{z}^*_{ijt-1})^+ && \text{(15b)} \\
&= \mathsf{Q}(\{\bar{\mathbf{x}}^*_\mathbf{t}, \bar{\mathbf{y}}^*_\mathbf{t}, \bar{\mathbf{z}}^*_\mathbf{t}, \forall t\}) + \textstyle\sum_t \sum_i \sum_j \delta_{jt+1} h_{ij} \bar{z}^*_{ijt} \\
&\quad + \textstyle\sum_t \sum_i \sum_j (f'_{ijt} + \delta_{jt} h_{ij}) (\bar{z}^*_{ijt} - \bar{z}^*_{ijt-1})^+ && \text{(15c)} \\
&\leq \left(1 + 3 \max_{i,j,t} \tfrac{\delta_{jt} \max_i h_{ij}}{f_{ij}} \right) \mathsf{P}_{opt}. && \text{(15d)} \\
&\leq \left(1 + 3 \left(\min_{i,j} V_j U_i \right) \left(\max_{i,j} \tfrac{\max_i h_{ij}}{f_{ij}} \right) \right) \mathsf{P}_{opt}. && \text{(15e)}
\end{aligned}$$

We reach (15a), because $\{\bar{\mathbf{x}}^*_\mathbf{t}, \bar{\mathbf{y}}^*_\mathbf{t}, \bar{\mathbf{z}}^*_\mathbf{t}, \forall t\}$ are not necessarily the optimal solution for $\mathbb{P}'$. We further have (15b) as we introduce $\sum_t \sum_i \sum_j \delta_{jt} h_{ij} \bar{z}^*_{ijt-1}$. We reach (15c) because by definition we have $\delta_{jT+1} = 0$ and $\bar{z}^*_{ij0} = 0$. We reach (15d) when we replace $f'_{ijt} + \delta_{jt} h_{ij}$ by f_{ij} so that we can construct P_{opt}. We eventually reach (15e) by the definition of $\mathbb{P}$.