Principled Data-Driven Decision Support for Cyber-Forensic Investigations

Soodeh Atefi [1], **Sakshyam Panda** [2], **Manos Panaousis** [2], **Aron Laszka** [3]

[1] University of Houston, [2] University of Greenwich, [3] Pennsylvania State University

Abstract

In the wake of a cybersecurity incident, it is crucial to promptly discover how the threat actors breached security in order to assess the impact of the incident and to develop and deploy countermeasures that can protect against further attacks. To this end, defenders can launch a cyber-forensic investigation, which discovers the techniques that the threat actors used in the incident. A fundamental challenge in such an investigation is prioritizing the investigation of particular techniques since the investigation of each technique requires time and effort, but forensic analysts cannot know which ones were actually used before investigating them. To ensure prompt discovery, it is imperative to provide decision support that can help forensic analysts with this prioritization. A recent study demonstrated that data-driven decision support, based on a dataset of prior incidents, can provide state-of-the-art prioritization. However, this data-driven approach, called DISCLOSE, is based on a heuristic that utilizes only a subset of the available information and does not approximate optimal decisions. To improve upon this heuristic, we introduce a principled approach for data-driven decision support for cyber-forensic investigations. We formulate the decision-support problem using a Markov decision process, whose states represent the states of a forensic investigation. To solve the decision problem, we propose a Monte Carlo tree search based method, which relies on a k-NN regression over prior incidents to estimate state-transition probabilities. We evaluate our proposed approach on multiple versions of the MITRE ATT&CK dataset, which is a knowledge base of adversarial techniques and tactics based on real-world cyber incidents, and demonstrate that our approach outperforms DISCLOSE in terms of techniques discovered per effort spent.

1 Introduction

Cybersecurity is an ever growing concern for businesses and individuals alike. While it is not always possible to prevent cybersecurity incidents, one should always strive to mitigate them promptly and to learn from them as much as possible. Thus, it is imperative in the aftermath of an incident to promptly discover the techniques that the threat actors used to breach security. By discovering how threat actors operate, we can learn to develop more effective cyber defences and detect—sufficiently early—cyber-attacks that bypass these defences. To this end, victims of cyber incidents can launch *cyber-forensic investigations*.

Since we do not know in advance which adversarial techniques the threat actors used, we have to prioritize the investigation of these techniques under *uncertainty*, not knowing if the investigation of a technique will waste precious time or reveal crucial information. In prior work, Nisioti et al. (2021) demonstrated that one can effectively prioritize the investigation of techniques based on datasets of prior incidents, exploiting the fact that most threat actors follow common tactics. However, this existing data-driven approach is based on a heuristic that does not approximate optimal decisions, regardless of the size of the dataset or the available computational power.

To address this limitation, we propose a principled data-driven approach for prioritization. We introduce a novel model of cyber-forensic investigations based on Markov decision processes, which enables us to formally define the prioritization problem. Note that this problem is inherently challenging for multiple reasons. First, datasets of prior incidents are limited in size as many businesses are reluctant to share detailed information about their security. Second, although threat actors follow common tactics, it is inherently difficult to predict which particular techniques were used in a particular incident.

In light of these challenges, we propose a computational approach for decision support that combines a non-parametric machine-learning model, based on k-nearest neighbor, with a Monte Carlo tree search algorithm. By working directly with the dataset instead of training a parametric model, we get as much "mileage" out of the limited dataset as possible.

The remainder of this paper is organized as follows. In Section 2, we model cyber-forensic investigations as Markov decision processes and formulate the problem of cyber-forensic decision support. In Section 3, we describe our computational approach, which is based on k-NN regression and Monte Carlo tree search. In Section 4, we evaluate our approach on datasets of real-world incidents, demonstrating that it outperforms the state-of-the-art approach. In Section 5, we give a brief overview of related work. Finally, in Section 6, we provide concluding remarks.

2 Model and Problem Formulation

To formally define the problem of providing optimal decision support for cyber forensics, we first introduce funda-

Symbol	Description
$\mathcal{A}$	set of adversarial techniques (actions in MDP)
$\mathcal{I}$	set of prior cyber incidents (dataset)
$I_Y \subseteq \mathcal{A}$	set of techniques used in incident I
$I_N \subseteq \mathcal{A}$	set of techniques not used in incident I
C_a	effort cost of investigating technique $a \in \mathcal{A}$
B_a	benefit from discovering technique $a \in \mathcal{A}$
$Y_t \subseteq \mathcal{A}$	set of used techniques discovered by step t
$N_t \subseteq \mathcal{A}$	set of not-used techniques discovered by step t
$\Pr[a\|Y_t, N_t]$	prob. that tech. a was used given state $\langle Y_t, N_t \rangle$
$\gamma \in (0, 1)$	temporal discount factor

Table 1: List of Model Symbols

mental assumptions and notations for the key elements of cyber-forensic investigations (Section 2.1). Then, building on these assumptions and notations, we model investigations as Markov decision processes (Section 2.2), formulating our problem as optimizing decisions within a process (Section 2.3). For a list of symbols used in our model, we refer the reader to Table 1.

2.1 Adversarial Techniques and Cyber Forensics

Adversarial Techniques We consider a forensic investigation whose objective is to discover—as soon as possible—what *techniques* the threat actors used in the incident that is under investigation. The motivation for discovering how the threat actors breached security is to learn from the incident and to prevent or mitigate future breaches. We let $\mathcal{A}$ denote the *set of all techniques* that threat actors may have used in an incident (e.g., DLL search order hijacking, spearphishing attachment, drive-by compromise). In practice, the set $\mathcal{A}$ can be taken from a well-known knowledge base, such as MITRE ATT&CK (Barnum 2012), which provides a comprehensive list of common adversarial techniques.

Incident Next, we let $I_Y \subseteq \mathcal{A}$ denote the *set of techniques that were used* by threat actors in incident I; and for ease of presentation, we let $I_N = \mathcal{A} \setminus I_Y$ denote the *set of techniques that were not used* in incident I. A key aspect of cyber-forensic investigations is that I_Y is not known by the forensic experts at the beginning of the investigation. Since an investigation is typically launched when a breach is detected, forensic experts might know some elements of I_Y (e.g., techniques that triggered an intrusion detection system, leading to the detection of the breach); however, discovering set I_Y is the very objective of the investigation. To capture this uncertainty that forensic experts face, we model the set of techniques I_Y as a *random variable*, whose realization is unknown at the beginning of the investigation. The distribution of this random variable represents the threat actors' tactics, that is, how likely they are to use certain subsets of techniques in a cyber attack.

Prior Incidents In practice, we can estimate the distribution of I_Y from prior incidents, which have already been investigated. Note that for this estimation, we can use a public repository of prior incidents, which were perpetrated by other threat actors against other targets (e.g., MITRE Cyber Threat Intelligence Repository from MITRE Corporation). While there are differences between how different threat actors operate, most threat actors do follow common tactics. Therefore, subsets of techniques that were frequently used in prior incidents are likely to have been used in the incident that we are currently investigating. We let $\mathcal{I}$ denote the set of prior incidents; and we assume that for each prior incident $\hat{I} \in \mathcal{I}$, we know the set $\hat{I}_Y \subseteq \mathcal{A}$ and that $\hat{I}_Y$ was drawn from the same distribution as I_Y.

Cyber-forensic Investigation During the investigation of incident I, forensic experts discover the realization of I_Y step-by-step. In each step, the experts choose and investigate a technique $a \in \mathcal{A}$ that they have not investigated yet, and they discover whether or not the threat actors used technique a. Note that we assume this discovery to be perfect (i.e., if a technique is investigated, experts correctly learn whether it was used or not); our model and computational approach could be generalized in a straightforward manner to account for false negatives and positives in discovery, but this is not the focus of our work.

Costs and Benefits of Forensic Discovery To investigate whether a particular technique was used by the threat actors, forensic experts have to spend time, effort, resources, etc. We let constant C_a denote the *cost of investigating technique* $a \in \mathcal{A}$, which represents the time, effort, resources, etc. spent. In practice, we can estimate these costs based on domain experts' knowledge (Nisioti et al. 2021). On the other hand, discovering that a technique was used by the threat actors yields benefit since it provides information that we can use to prevent or mitigate future breaches. We let constant B_a denote the *benefit obtained from discovering* that technique $a \in \mathcal{A}$ was used in the incident. In practice, we can estimate these benefit values based on the impacts of using various techniques, which we can take from well-known knowledge bases (e.g., MITRE ATT&CK (Barnum 2012)). Some experts may inherently prefer investigating certain actions over others, e.g., because certain actions provide crucial information on how the adversary breached the system. All such preferences can be captured by the benefit values.

2.2 Markov Decision Process

Based on the assumptions and notations introduced in the preceding subsection, we now model the cyber-forensic investigation of an incident as a *Markov decision process* (MDP) (Puterman 2014), whose steps correspond to the step-by-step investigation of the adversarial techniques. Modeling the cyber-forensic investigation as an MDP provides the foundation for formulating our decision-support problem in the next subsection. To define an MDP, we have to specify the *state space* of the process, the set of *actions* that can be taken in each state, the *transition probabilities* between subsequent states, and the *immediate reward* for taking a particular action in a particular state.

State Space At any step during the investigation of an incident, the forensic experts have already investigated some

techniques, while other techniques remain yet to be investigated. Further, for each technique that the experts have already investigated, they have discovered whether or not the threat actors used the technique in the incident. To formalize the state of the investigation process, we let $Y_t \subseteq \mathcal{A}$ denote the *set of techniques that have been investigated by step t and were used by the threat actors*, and we let $N_t \subseteq \mathcal{A}$ denote the *set of techniques that have been investigated by step t but were not used by threat actors*. Note that by definition, $Y_t \subseteq I_Y$ and $Y_t \cap N_t = \emptyset$ for every t. At the beginning of the investigation, before the first step, the process is in an *initial state* $\langle Y_0, N_0 \rangle$. In practice, Y_0 can be the set of techniques that triggered an intrusion detection system, leading to the detection of the incident and the launch of the investigation.

Action Space At each step t, the forensic experts have to choose a technique that they have not investigated yet, and investigate it. To formalize this decision, we let the set of actions that can be taken in a state correspond to the set of techniques that have not yet been investigated: in state $\langle Y_t, N_t \rangle$, the *set of actions* is the set of techniques $\mathcal{A} \setminus (Y_t \cup N_t)$.

Transition Probabilities Recall that forensic experts do not know in advance (i.e., before investigating) which techniques were used by the threat actors, and we capture this uncertainty by assuming that I_Y is a random variable whose realization is unknown at the beginning of the investigation. When experts investigate a technique a, they discover whether technique a was used or not (i.e., whether technique a is in the realization of I_Y). Hence, if action $a \in \mathcal{A} \setminus (Y_t \cup N_t)$ is taken in state $\langle Y_t, N_t \rangle$, then the process transitions either to state $\langle Y_{t+1}, N_{t+1} \rangle = \langle Y_t \cup \{a\}, N_t \rangle$ (if technique a was used) or to state $\langle Y_{t+1}, N_{t+1} \rangle = \langle Y_t, N_t \cup \{a\} \rangle$ (if technique a was not used). The *probabilities of these transitions* are

$$\begin{aligned} &\Pr\left[\langle Y_{t+1}, N_{t+1} \rangle = \langle Y_t \cup \{a\}, N_t \rangle\right] \\ &\quad = \Pr\left[a \in I_Y \,|\, Y_t \subseteq I_Y \wedge N_t \cap I_Y = \emptyset\right] \end{aligned} \tag{1}$$

and

$$\begin{aligned} &\Pr\left[\langle Y_{t+1}, N_{t+1} \rangle = \langle Y_t, N_t \cup \{a\} \rangle\right] \\ &\quad = \Pr\left[a \notin I_Y \,|\, Y_t \subseteq I_Y \wedge N_t \cap I_Y = \emptyset\right] \\ &\quad = 1 - \Pr\left[a \in I_Y \,|\, Y_t \subseteq I_Y \wedge N_t \cap I_Y = \emptyset\right]. \end{aligned} \tag{2}$$

For ease of notation, we will let $\Pr[a \,|\, Y_t, N_t]$ denote the first probability (i.e., probability that the threat actors used technique a given that the state is $\langle Y_t, N_t \rangle$). In practice, we have to estimate these conditional probabilities based on the set of prior incidents $\mathcal{I}$.

Rewards We formulate rewards to capture the benefits of the cyber-forensic investigation. The experts obtain a benefit B_a from investigating technique a only if technique a was used in the incident. Hence, if the process transitions from state $\langle Y_t, N_t \rangle$ to state $\langle Y_{t+1}, N_{t+1} \rangle = \langle Y_t \cup \{a\}, N_t \rangle$, then we let the *reward for step t be* B_a; if the process transitions to state $\langle Y_{t+1}, N_{t+1} \rangle = \langle Y_t, N_t \cup \{a\} \rangle$, then we let the *reward for step t be* 0.

2.3 Cyber-forensic Decision-Support Problem

We represent the decision-support system as a policy π, which maps a state $\langle Y_t, N_t \rangle$ to a recommended action $a_t \in \mathcal{A} \setminus (Y_t \cup N_t)$ in each time step t. Our goal is to provide a *policy that maximizes the expected rewards obtained during the forensic investigation*. Following Nisioti et al. (2021), we formulate this objective with a *budget limit* G on the total cost $\sum C_{a_t}$:

$$\max_{\pi} \mathbb{E}_{I_Y} \left[\sum_{t=0}^{T_{limit}} 1_{\{a_t \in I_Y\}} \cdot B_{a_t} \,\middle|\, a_t = \pi\left(Y_t, N_t\right) \right] \tag{3}$$

where $T_{limit} = \max_T \sum_{t=0}^{T} C_{a_t} \leq G$ (i.e., T_{limit} is the last step before the investigation budget G is exhausted), and C_{a_t} is the cost of investigating the action that is chosen in time step t. Note that we focus on this objective because it is practical and enables a fair comparison with DISCLOSE (Nisioti et al. 2021); in Equation (8), we will provide a more conventional formulation with temporal discounting.

In practice, the budget may be flexible, in which case our goal is to provide a policy that attains a good cost-benefit tradeoff. We will quantify this tradeoff in our experiments using the area under the cost-benefit curve (i.e., AUC for the expected benefit as a function of the budget limit).

3 Computational Approach

To solve the decision-support problem based on the objective in Equation (3), we propose a computational approach based on *Monte Carlo tree search* (MCTS) and *k-nearest neighbour* (k-NN) algorithms. Specifically, we implement the policy π as an MCTS algorithm (Section 3.2), relying on k-NN for estimating transition probabilities (Section 3.1).

Note that in recent years, deep reinforcement learning (DRL) algorithms have garnered significant attention from researchers for their applications in cybersecurity decision support (e.g., (Ganesan et al. 2016; Kurt et al. 2018)). While we could apply DRL algorithms to our problem in principle, they are ill-suited to our problem for multiple reasons. First, DRL algorithms tend to be sample inefficient (i.e., require a large number of training experiences to learn a performant policy), which poses a significant challenge since it is difficult to collect large datasets of prior incidents with sufficient details. Second, the limited size of the dataset enables us to make decisions based directly on the dataset, instead of having to train a parametric machine-learning model. By working directly with the dataset, decisions can take advantage of all the information in the dataset.

3.1 Probability Estimation

In a nutshell, Monte Carlo tree search finds which action to take in a given state by randomly sampling sequences of actions, simulating how they play out starting from the given state, and choosing the action that leads to the highest rewards on average. We can simulate the cyber-forensic investigation process from any given state using the state-transition probabilities (Equations (1) and (2)).

Empirical Probabilities However, since we do not know the underlying probability distribution in practice, we have to estimate the probabilities based on the set of prior incidents $\mathcal{I}$. We can estimate the state-transition probability $\Pr[a \mid Y_t, N_t]$ as the *conditional empirical probability*:

$$\Pr[a|Y_t, N_t] \equiv \Pr\left[a \in I_Y | Y_t \subseteq I_Y \wedge N_t \cap I_Y = \emptyset\right] \quad (4)$$

$$\approx \frac{\left|\left\{\hat{I} \in \mathcal{I}_{\langle Y_t, N_t\rangle} \,\middle|\, a \in \hat{I}_Y\right\}\right|}{\left|\mathcal{I}_{\langle Y_t, N_t\rangle}\right|} \quad (5)$$

where

$$\mathcal{I}_{\langle Y_t, N_t\rangle} = \left\{\hat{I} \in \mathcal{I} \,\middle|\, Y_t \subseteq \hat{I}_Y \wedge N_t \cap \hat{I}_Y = \emptyset\right\}. \quad (6)$$

In other words, $\mathcal{I}_{\langle Y_t, N_t\rangle}$ is the set of prior incidents which "match" the current state of the investigation, that is, the set of prior incidents in which threat actors did use all of the techniques from Y_t but did not use any of the techniques from N_t. Equation (5) estimates the probability $\Pr[a \mid Y_t, N_t]$ as the ratio of incidents from $\mathcal{I}_{\langle Y_t, N_t\rangle}$ in which threat actors used technique a.

k-nearest Neighbors The weakness of this estimation is that its accuracy depends on the cardinality of the set $\mathcal{I}_{\langle Y_t, N_t\rangle}$, and as the sets Y_t and N_t grow with each step of the cyber-forensic investigation, the set $\mathcal{I}_{\langle Y_t, N_t\rangle}$ shrinks. In fact, due to the limited number of prior incidents, the number of prior incidents that "match" the current state of the investigation may reach zero, which precludes us from calculating the estimates at all. To address this issue, we extend the set $\mathcal{I}_{\langle Y_t, N_t\rangle}$ to include prior incidents that do not exactly "match" the current state of the investigation but are sufficiently similar. We measure similarity between the current state $\langle Y_t, N_t\rangle$ and a prior incident $\hat{I} \in \mathcal{I}$ using a *Hamming distance over the techniques* $Y_t \cup N_t$ that have already been investigated:

$$d(\langle Y_t, N_t\rangle, \hat{I}) = \left|Y_t \cap \hat{I}_N\right| + \left|N_t \cap \hat{I}_Y\right|. \quad (7)$$

The first term of the right-hand side is the number of techniques that were used in incident I but not in prior incident $\hat{I}$, while the second term is the number of techniques that were used in prior incident $\hat{I}$ but not in incident I. Equivalently, we could formulate the following metric: $s(\langle Y_t, N_t\rangle, \hat{I}) = \left|Y_t \cap \hat{I}_Y\right| + \left|N_t \cap \hat{I}_N\right|$, which measures similarity by counting techniques where the current state and the prior incident are the same; in contrast, our formulation measures dissimilarity (specifically, Hamming distance) by counting techniques where the current state and the prior incident differ. These measures are practically equivalent since $d(\langle Y_t, N_t\rangle, \hat{I}) = |Y_t \cup N_t| - s(\langle Y_t, N_t\rangle, \hat{I})$; hence, selecting k incidents with highest similarity is equivalent to selecting k incidents with lowest distance (i.e., k-NN).

Finally, we replace the set of "matching" prior incidents $\mathcal{I}_{\langle Y_t, N_t\rangle}$ in Equation (5) with the set of k prior incidents that are closest with respect to metric d (breaking ties arbitrarily). Notice that this estimation of the probability $\Pr[a \mid Y_t, N_t]$ is actually a *k-nearest neighbor regression* with prior incidents $\mathcal{I}$ as the dataset, d as the distance metric, and a binary value representing if technique a was used in an incident as the output feature. While the value of k could be a constant hyper-parameter, we found that our approach performs better if k varies throughout the investigation. Hence, we let $k = \beta_1 + \beta_2 \cdot t$, where β_1 and β_2 are hyper-parameters, which can find experimentally.

Algorithm 1 Exploration Decision

Function *ExplorationDecision*(Y, N, M, F)**:**

$MP \leftarrow \operatorname{argmax}_{\mathcal{A}' \subseteq \mathcal{A} \setminus Y \cup N,\ |\mathcal{A}'| = F} \sum_{a \in \mathcal{A}'} \Pr[a \mid Y, N] \cdot B_a / C_a$

$n[Y, N] \leftarrow \sum_{a \in \mathcal{A} \setminus (Y \cup N)} n[Y, N, a]$

return $\operatorname{argmax}_{a \in MP} R[Y, N, a] + M\sqrt{\frac{\ln n[Y,N]}{n[Y,N,a]+1}}$

3.2 Monte Carlo Tree Search

Since Monte Carlo tree search randomly samples sequences of actions, its coverage of the state space becomes sparser and sparser as it looks further into the future. Therefore, when choosing between actions, it should assign more importance to the near future than the uncertain far future. We can express this consideration by reformulating the objective as *maximizing the expected discounted sum of rewards*:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{|\mathcal{A}|-1} \gamma^t \cdot 1_{\{a_t \in I_Y\}} \cdot B_{a_t}/C_{a_t} \,\middle|\, a_t = \pi\left(Y_t, N_t\right)\right] \quad (8)$$

where $\gamma \in (0, 1)$ is a *temporal discount factor*: rewards obtained at step t are discounted by a factor γ^t.

Note that we also replace the reward B_{a_t} with the benefit to cost ratio B_{a_t}/C_{a_t}. The rationale behind this is to incentivize the tree search to consider cost C_a regardless of how far the investigation is from reaching a budget limit G; otherwise, the tree search would focus only on immediate benefit B_a. Note that we found the ratio B_{a_t}/C_{a_t} formulation to work best in practice (e.g., better than the more intuitive semi-MDP formulation with $\gamma^{\sum_{\tau=0}^{t} C_{a_\tau}}$ temporal discount and reward B_{a_t}).

In each step of the investigation, we run a Monte Carlo tree search (Algorithm 2), starting from the current state $\langle Y_t, N_t\rangle$, which outputs an action a_t that is estimated to result in the maximum expected discounted sum of rewards. To estimate the expected rewards for each action, the algorithm performs a number of iterations. In each iteration, it first generates a sequence of actions and states, starting from the current state, which is a random sample of how the investigation might play out if certain actions are selected. Then, in the back-propagation phase of the iteration, it updates its estimates of the expected rewards based on the experience of the sampled sequence. While our algorithm follows the common principles of MCTS, it is tailored to our problem and takes advantage of the specific rules of our MDP.

Variables and Initialization Here, we describe the variables that are maintained by the algorithm throughout the iterations and how they are initialized; we will describe later how they are updated.

Algorithm 2 MCTS for Forensic Decision Support

Input: state $\langle Y_t, N_t \rangle$, constants $\mathcal{A}, \boldsymbol{B}, \boldsymbol{C}, \gamma, K, D, M, F$
Output: action a_t
Initialization:
$\forall \langle Y, N, a \rangle : n[Y, N, a] \leftarrow 0$
$\forall \langle Y, N, a \rangle : R[Y, N, a] \leftarrow 0$
$\forall \langle Y, N \rangle : R[Y, N] \leftarrow$
$\sum_{a \in \mathcal{A} \setminus (Y \cup N)} \sum_{j=0}^{|\mathcal{A} \setminus (Y \cup N)|-1} \gamma^j \frac{\Pr[a|Y_t, N_t] \cdot B_a / C_a}{|\mathcal{A} \setminus (Y \cup N)|}$

for K *times* **do**
  $i \leftarrow t$
  while $Y_i \cup N_i \neq \mathcal{A}$ **and** $i < t + D$ **do**
    $a_i \leftarrow$ *ExplorationDecision*(Y_i, N_i, M, F)
    $n[Y_i, N_i, a_i] \leftarrow n[Y_i, N_i, a_i] + 1$
    if $random(0, 1) < 0.5$ **then**
      $Y_{i+1} \leftarrow Y_i \cup \{a_i\}$
      $N_{i+1} \leftarrow N_i$
    else
      $Y_{i+1} \leftarrow Y_i$
      $N_{i+1} \leftarrow N_i \cup \{a_i\}$
    end
    $i \leftarrow i + 1$
  end
  $i \leftarrow i - 1$
  while $i \geq t$ **do**
    $R[Y_i, N_i, a_i] \leftarrow$
      $\Pr[a_i|Y_i, N_i] \cdot (B_{a_i}/C_{a_i} + \gamma \cdot R[Y_i \cup \{a_i\}, N_i])$
      $+ (1 - \Pr[a_i|Y_i, N_i]) \cdot \gamma \cdot R[Y_i, N_i \cup \{a_i\}]$
    $R[Y_i, N_i] \leftarrow \max_{a \in \mathcal{A} \setminus (Y_i \cup N_i)} R[Y_i, N_i, a]$
    $i \leftarrow i - 1$
  end
end
$a_t \leftarrow \operatorname{argmax}_{a \in \mathcal{A} \setminus (Y_t \cup N_t)} R[Y_t, N_t, a]$

Variable $n[Y, N, a]$ is the number of times that the algorithm has tried action a in state $\langle Y, N \rangle$ (initialized to 0 at the beginning of the search). Note that this and all other initializations can be lazy, i.e., we can store these variables in a dictionary that is initially empty, and we can assign an initial value to a variable when we use it for the first time. For ease of presentation, the pseudocode initializes all variables explicitly at the beginning.

Variable $R[Y, N, a]$ is an estimate of the expected discounted sum of rewards if we started from state $\langle Y, N \rangle$, took action a first, and then followed an optimal policy.

Finally, variable $R[Y, N]$ is an estimate of the expected discounted sum of rewards if we started from state $\langle Y, N \rangle$ and then followed an optimal policy. While these variables could be initialized to 0, we can improve the performance of the search by initializing them with an estimate:

$$R[Y, N] \leftarrow \sum_{a \in \mathcal{A} \setminus (Y \cup N)} \sum_{j=0}^{|\mathcal{A} \setminus (Y \cup N)|-1} \gamma^j \frac{\Pr[a|Y_t, N_t] \cdot B_a / C_a}{|\mathcal{A} \setminus (Y \cup N)|}$$

Starting from state $\langle Y, N \rangle$, we can investigate techniques $\mathcal{A} \setminus (Y \cup N)$; however, we do not know the optimal order in which to investigate them. Our estimate calculates expected rewards assuming a random order: each technique $a \in \mathcal{A} \setminus (Y \cup N)$ is equally likely to be investigated first (discount factor 1), second (factor γ), third (factor γ^2), ..., and last (factor $\gamma^{|\mathcal{A} \setminus (Y \cup N)|-1}$). Our estimate can be calculated very quickly, especially since the probability estimates $\Pr[a|Y_t, N_t]$ need to be calculated anyway for the back-propagation phase. Note that for terminal states (i.e., when $\mathcal{A} \setminus (Y \cup N) = \emptyset$), this formula correctly calculates the expected rewards to be 0, following the notational convention that summation over an empty set yields 0.

Selection and Expansion In each iteration of the MCTS, we first generate a sequence of states and actions $\langle Y_t, N_t \rangle, a_t, \langle Y_{t+1}, N_{t+1} \rangle, a_{t+1}, \langle Y_{t+2}, N_{t+2} \rangle, a_{t+2}, \ldots$, which we then use in the back-propagation phase to improve our estimates $R[Y, N]$ and $R[Y, N, a]$ for the states and actions in the sequence. We generate the sequence starting from state $\langle Y_t, N_t \rangle$ by alternating between selecting an action a_i to take and simulating the resulting transition to state $\langle Y_{i+1}, N_{i+1} \rangle$. To select each action a_i, we use the exploration rule described in Algorithm 1, which takes a state $\langle Y, N \rangle$ and outputs a selected action a, balancing exploration and exploitation. This rule is based on the commonly used UCT (Upper Confidence Bound 1 applied to trees) formula (Kocsis and Szepesvári 2006) with a myopic pruning (see below). The first term $R[Y, N, a]$ gives preference to actions that should be selected because—according to our current estimates—they lead to high expected rewards; while the second term gives preference to actions that have been explored less (i.e., tried fewer times) than other actions in the given state. The exploration factor M is a constant hyper-parameter that balances exploration and exploitation, which we can find experimentally. Note that we add 1 to the divisor to avoid division by zero since $n[Y, N, a]$ is initialized to 0; however, this is just for ease of exposition, the addition of 1 is negligible as we run a large number of iterations.

After selecting an action a_i, we update the number of times $n[Y_i, N_i, a_i]$ that action a_i has been tried, and we simulate the random transition to state $\langle Y_{i+1}, N_{i+1} \rangle$. However, instead of basing the transition on the actual probability $\Pr[a_i|Y_i, N_i]$, we select one of the two possible transitions with the same probability (i.e., 0.5 probability that a_i was used, and 0.5 probability that it was not). We use uniform probabilities for two reasons. First, we factor in the actual probability $\Pr[a_i|Y_i, N_i]$ during back-propagation; hence, we obtain unbiased estimates regardless of the selection probabilities. Second, uniform probabilities lead to a more balanced and thorough exploration of the transitions; we found that with the actual probabilities, transitions that have high impact but low probability are not explored enough.

We stop generating the sequence of states and actions when we reach either a terminal state ($Y_i \cup N_i = \mathcal{A}$) or the depth limit ($i \geq t + D$, where D is a hyper-parameter).

Myopic Pruning To improve search performance, we prune the search tree during selection by exploring only those actions that are most attractive in terms of the expected immediate reward. Specifically, in state $\langle Y, N \rangle$, Algorithm 1 explores only those F actions that have the highest

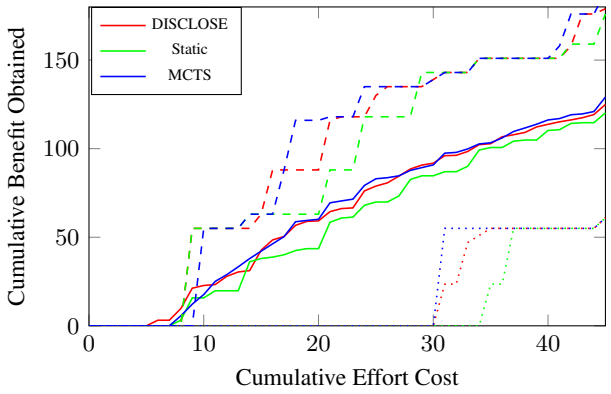

Figure 1: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 45**) on dataset **v6.3**.

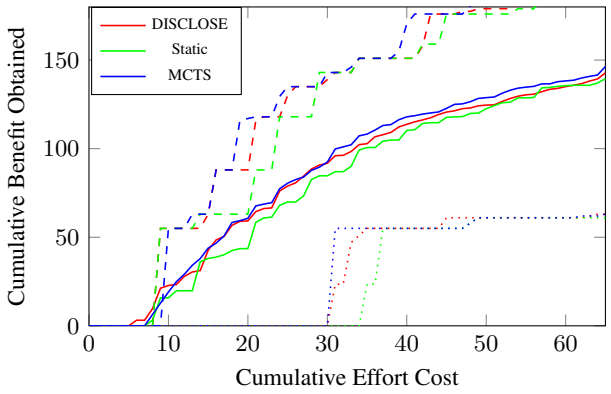

Figure 2: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 65**) on dataset **v6.3**.

$\Pr[a \mid Y, N] \cdot B_a/C_a$, where F is a hyper-parameter. While myopically focusing on actions with the highest expected immediate reward may occasionally disregard optimal actions, it typically provides better results by letting the search thoroughly explore the most promising branches of the tree.

Backpropagation During the back-propagation phase of each iteration, we loop over the sequence of states and actions backwards, and we update our estimates based on the experience of this sequence. For each action a_i, we update the estimate $R[Y_i, N_i, a_i]$ using the following formula:

$$\begin{aligned} &R[Y_i, N_i, a_i] \leftarrow \\ &\quad \Pr[a_i|Y_i, N_i] \cdot (B_{a_i}/C_{a_i} + \gamma \cdot R[Y_i \cup \{a_i\}, N_i]) \\ &\quad + (1 - \Pr[a_i|Y_i, N_i]) \cdot (\gamma \cdot R[Y_i, N_i \cup \{a_i\}]). \end{aligned}$$

This formula calculates a best estimate of the expected discounted sum of rewards (relying on the k-NN based probability estimate $\Pr[a_i|Y_i, N_i]$): if technique a_i was used in the incident, we obtain immediate reward B_{a_i}/C_{a_i} and then continue from state $\langle Y_i \cup \{a_i\}, N_i\rangle$, factoring in discount γ as this state is one step into the future; if technique a_i was not used, a similar argument applies. Note that each $R[Y, N]$ is either the initial estimate, which is 0 for terminal states, or an estimate based on prior iterations (see below).

For each state $\langle Y_i, N_i\rangle$, we update the estimate $R[Y_i, N_i]$ using the following formula:

$$R[Y_i, N_i] \leftarrow \max_{a \in \mathcal{A} \setminus (Y_i \cup N_i)} R[Y_i, N_i, a] \tag{9}$$

This formula calculates a best estimate since in each state $\langle Y_i, N_i\rangle$, we should take the optimal action a that maximizes the expected discounted sum of rewards (based on our best estimates $R[Y_i, N_i, a]$).

4 Numerical Evaluation

We evaluate our proposed approach numerically on public datasets of real-world cyber incidents. For each cyber incident, we simulate how our approach would have prioritized the investigation, and plot the benefit obtained through discovery as a function of the effort spent. We compare our approach to two baselines, DISCLOSE and a naïve policy. Our implementation and datasets are publicly available.[1]

4.1 Experimental Setup

Dataset We use the MITRE ATT&CK Enterprise repository (Barnum 2012), which is a public repository of adversarial tactics, techniques, & procedures, referencing real-world cyber incidents in which some of these techniques were used. Since its original publication, both the ATT&CK framework and its CTI dataset have been regularly updated. We use three versions of the repository in our evaluation: v6.3 (2019), which is the version that Nisioti et al. (2021) used to evaluate DISCLOSE; v10.1 (2021); and v11.3 (2022), which is the latest version at the time of writing. Evaluating our approach on multiple versions demonstrates that it can be applied to newer versions without any changes (other than standard hyper-parameter optimization).

For a fair comparison, we use the same 31 techniques $\mathcal{A}$ and the same benefit B and cost C values as Nisioti et al. (2021). Since the categorization of techniques changed slightly between versions, we had to map some techniques to equivalent ones for later versions. This leaves us with 29 techniques for versions v10.1 and v11.3. The benefit and cost values are based on the Common Vulnerability Scoring System and interviews with cyber-forensic experts.

For each technique, we collected cyber incidents in which the technique was used via the `external_references` field (replicating Nisioti et al. (2021)). Our v6.3, v10.1, and v11.3 datasets contain 331, 670, and 716 cyber incidents, respectively. Version v11.3 includes 425 incidents that are new compared to v6.3, and 73 incidents that are new compared to v10.1. In these datasets, every incident used at least 2 techniques. In dataset v11.3, incidents used 4.1 techniques on average (min. 2; max. 17). One example of an incident that used many techniques is the Frankenstein campaign (Biasini 2019), which employed Phishing, OS Credential Dumping, Exfiltration Over C2 Channel, and other techniques.

Baselines We compare our approach to two baselines, DISCLOSE (implemented as specified in Nisioti et

[1] https://github.com/SoodehAtefi/DecisionSupport-AAAI-23

al. (2021)) and a naïve baseline, which we call the *static policy*. At every step, the static policy selects the technique (from the set of techniques that have not been investigated yet) that is most frequent across all prior incidents, instead of considering conditional probabilities. This naïve baseline represents investigation without decision support (i.e., decisions that ignore the state).

Simulation Setup and Metrics Since the datasets are relatively small, we use a leave-one-out cross-validation: when evaluating a policy on an incident, we treat all other incidents in our dataset as prior incidents $\mathcal{I}$. We always simulate the investigation starting with a single randomly chosen technique from I_Y as the singleton set Y_0 (and letting $N_0 = \emptyset$). Same as Nisioti et al. (2021), we terminate an investigation when the *cumulative effort cost* (i.e., $\sum_{a \in Y_t \cup N_t} C_a$) reaches 45 or 65 to assess which techniques could have been promptly discovered by the approach. We also conducted experiments with two more budget limits (70 and 100) and without a budget limit to provide a more comprehensive comparison (we include results in Appendix A.1). Further, to quantify promptness, we measure the *area under the benefit-effort curve* (AUCBE): we plot the discovery of benefits obtained as a function of the cumulative effort cost for three different approaches (e.g., see Figure 1). Higher AUCBE values are better. Solid lines are averages over all incidents, dotted lines are 25% quantiles, and dashed lines are 75% quantiles.

Hyperparameter Optimization While the baseline does not have any hyper-parameters, our proposed approach has a number of them. First, we optimized the hyper-parameters for the k-NN probability estimation (β_1, β_2) using a grid search, maximizing the average AUCBE over all incidents. During this search, we restricted the MCTS to one-action depth ($D = 1$, in which case the other parameters of the MCTS are irrelevant), providing us with good hyper-parameters for probability estimation. Then, we optimized the hyper-parameters for MCTS using Hyperopt Python library, again maximizing AUCBE. Note that we optimized the hyper-parameters for datasets separately. We provide results from the hyper-parameter search in Appendix A.2.

4.2 Numerical Results

Running Time For a single decision, the MCTS algorithm takes less than 7 seconds on average using a single core of a 2.4GHz Intel Core i9 CPU, and less than a second using multiple cores. Compared to the amount of time that forensic analysts need to investigate the selected technique (at the very least minutes), these running times are negligible.

Benefits of Prioritization Figures 1 and 2 show cumulative benefits obtained during the cyber-forensic investigations (i.e., $\sum_{a \in Y_t} B_a$) as functions of the cumulative effort costs (i.e., $\sum_{a \in Y_t \cup N_t} C_a$) using MCTS, DISCLOSE, and static approach on the v6.3 dataset. We conducted the experiments with budgets 45 and 65. We provide results for other versions of the dataset and different budgets in Appendix A.1. Each curve is based on all incidents in the dataset. For the v6.3 dataset, our approach outperforms the baselines in both of the scenarios (45, and 65). The average AUCBE of MCTS, DISCLOSE, and static policy for budget limit 45 are 3,503, 3,411, and 3,175 respectively. The average AUCBE of our approach, DISCLOSE, and static policy for budget limit 65 are 6288, 6108, and 5826 respectively.

For the v11.3 dataset, our approach outperforms the baselines in both of the scenarios as well. The average AUCBE of MCTS, DISCLOSE, and static policy for budget limit 45 are 4,061, 3,982, and 3,865, and for budget limit 65 are 7,072, 6,975, and 6,816 respectively. This demonstrates that while our principled approach provides superior prioritization, the decision-support problem is very challenging due to the inherent uncertainty of cyber incidents.

5 Related Work

Several prior research efforts focused on enhancing the efficacy of forensic investigations by attempting to decrease the time or resources required for an investigation without impacting its quality and objectivity. Our approach is primarily motivated by DISCLOSE (Nisioti et al. 2021), a data-driven decision-support framework, which creates TTP space using the MITRE ATT&CK dataset, computes conditional probabilistic relations and proximity values between TTPs, and proposes actions based on these relations. DISCLOSE can assist an expert in each step of the investigation by considering benefits, costs, and available budget. Our approach extends this work by introducing a formal model of the cyber-forensic investigation process, modeling it as a Markov decision process. Further, we propose a computational approach which, at each step of the investigation, considers all the techniques that have been investigated, instead of considering only the very last technique—as DISCLOSE does.

Horsman et al. (2014) present CBR-FT, a case driven reasoning based technique for device triage. Similar to our approach, CBR-FT uses a knowledge base of past cases to calculate probable subsequent actions based on system file paths, which is then used to offer prediction of triage process of the current case under investigation. Unlike the system file paths, we use TTPs which allows for more flexible analysis and reasoning of adversarial behavior. Attack graphs were included in forensic investigation by Lui et al. (2012) to guide the decision process. The notion of anti-forensic steps, parallel to other adversarial actions, were included in the attack graphs to represent that the adversary may hide relevant forensic evidence to dissuade the defender. Likewise, Nassif and Hruschka (2012) propose the use of clustering techniques to support forensic investigation. Using a similar approach, Barrere et al. (2017) proposed an algorithm using condensed attack graphs that transformed the structure of the original attack graph allowing for more effective exploration. Algorithms are also proposed to support forensic investigation of online social networks (Arshad et al. 2020) and for unsupervised prediction for proactive forensic investigation of insider threats (Wei, Chow, and Yiu 2021). Similar to our work, these algorithms aim to increase the efficacy of forensic investigation by decreasing the investigation time; however, they approach the problem differently by decreasing the time required for crucial tasks.

Quick and Choo (2014) highlight the effects of large amounts of digital-forensic data on modern forensic investigations. Saeed et al. (2020) present game-theoretic approaches used to model interactions between an investigator and an adversary using anti-forensic method. By finding the Nash equilibrium of the game, the authors find the optimal trade-off strategy for each player. Even though our work does not explicitly model anti-forensic techniques and their analysis, our approach is developed considering the potential existence of anti-forensic techniques for an incident under investigation.

6 Discussion and Conclusions

To address the limitations of DISCLOSE, we introduced a principled approach by modeling cyber-forensic investigation as Markov decision processes and proposing an MCTS and k-NN regression based computational approach. A key advantage of our proposed approach is that it works directly with the data (i.e., at every step and every iteration, probabilities are estimated based on the dataset of all prior incidents), instead of training a parametric machine-learning model. By relying on non-parametric machine-learning models, we aim to get as much "mileage" out of the data as possible, which is both enabled and necessitated by the limited amount of public data about cyber incidents. Another key advantage of our proposed approach is that the tree search approximates best estimates—and hence optimal decisions—based on the dataset (as the number of iterations increases).

While these advantages enabled our approach to outperform baselines, including DISCLOSE, we found the prioritization problem to be very challenging. The primary reason for this is the inherent difficulty of predicting how threat actors work. In fact, DISCLOSE itself is not too far from the naïve baseline policy.

Acknowledgements

This material is based upon work sponsored by the National Science Foundation under Grant No. CNS-1850510. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We thank the anonymous reviewers for their valuable feedback and suggestions.

References

Arshad, H.; Omlara, E.; Abiodun, I. O.; and Aminu, A. 2020. A semi-automated forensic investigation model for online social networks. *Computers & Security*, 97: 101946.

Barnum, S. 2012. Standardizing cyber threat intelligence information with the structured threat information expression (STIX). *MITRE Corporation*, 11: 1–22.

Barrère, M.; Steiner, R. V.; Mohsen, R.; and Lupu, E. C. 2017. Tracking the bad guys: An efficient forensic methodology to trace multi-step attacks using core attack graphs. In *2017 13th International Conference on Network and Service Management (CNSM)*, 1–7. IEEE.

Biasini, N. 2019. It's alive: Threat actors cobble together open-source pieces into monstrous Frankenstein campaign. Cisco Talos Intelligence Group, https://blog.talosintelligence.com/frankenstein-campaign/.

da Cruz Nassif, L. F.; and Hruschka, E. R. 2012. Document clustering for forensic analysis: An approach for improving computer inspection. *IEEE transactions on information forensics and security*, 8(1): 46–54.

Ganesan, R.; Jajodia, S.; Shah, A.; and Cam, H. 2016. Dynamic scheduling of cybersecurity analysts for minimizing risk using reinforcement learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1): 1–21.

Horsman, G.; Laing, C.; and Vickers, P. 2014. A case-based reasoning method for locating evidence during digital forensic device triage. *Decision Support Systems*, 61: 69–78.

Kocsis, L.; and Szepesvári, C. 2006. Bandit based Monte-Carlo planning. In *Proceedings of the 17th European Conference on Machine Learning (ECML)*, 282–293. Springer.

Kurt, M. N.; Ogundijo, O.; Li, C.; and Wang, X. 2018. Online cyber-attack detection in smart grid: A reinforcement learning approach. *IEEE Transactions on Smart Grid*, 10(5): 5174–5185.

Liu, C.; Singhal, A.; and Wijesekera, D. 2012. Using attack graphs in forensic examinations. In *2012 Seventh International Conference on Availability, Reliability and Security*, 596–603. IEEE.

MITRE Corporation. 2022. MITRE Cyber Threat Intelligence Repository. https://github.com/mitre/cti, accessed on August 25th, 2022.

Nisioti, A.; Loukas, G.; Laszka, A.; and Panaousis, E. 2021. Data-Driven Decision Support for Optimizing Cyber Forensic Investigations. *IEEE Transactions on Information Forensics & Security*, 16: 2397–2412.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Quick, D.; and Choo, K.-K. R. 2014. Impacts of increasing volume of digital forensic data: A survey and future research challenges. *Digital Investigation*, 11(4): 273–294.

Saeed, S. H.; Arash, H. L.; and Ghorbani, A. A. 2020. A survey and research challenges of anti-forensics: Evaluation of game-theoretic models in simulation of forensic agents' behaviour. *Forensic Science International: Digital Investigation*, 35: 301024.

Wei, Y.; Chow, K.-P.; and Yiu, S.-M. 2021. Insider threat prediction based on unsupervised anomaly detection scheme for proactive forensic investigation. *Forensic Science International: Digital Investigation*, 38: 301126.

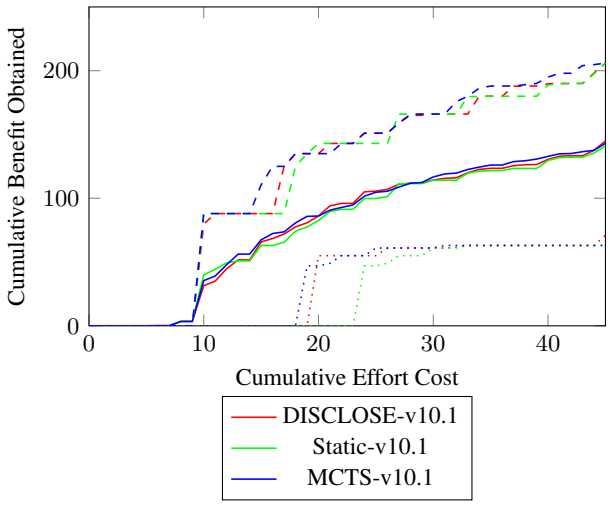

Figure 3: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 45**) on **v10.1**.

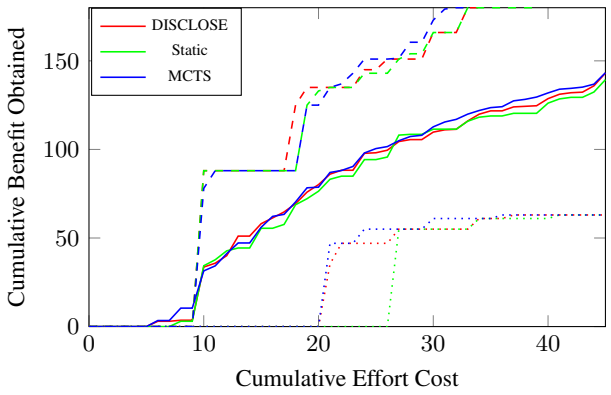

Figure 4: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 45**) on **v11.3**.

A Appendix

A.1 Evaluation with Different Budgets

Figures 3 to 15 show cumulative benefits obtained during the cyber-forensic investigations (i.e., $\sum_{a \in Y_t} B_a$) as functions of the cumulative effort costs (i.e., $\sum_{a \in Y_t \cup N_t} C_a$) using MCTS, DISCLOSE, and static approach on the v6.3, v10.1, and v11.3 datasets with budget limits of 45 and 65 (except v6.3), 70, 100, and without budget limit. We provided figures for the v6.3 dataset with budget limits 45 and 65 in the main text (Section 4). Solid lines are averages over all incidents, dotted lines are 25% quantiles, and dashed lines are 75% quantiles.

A.2 Hyper-Parameter Search

For each budget limit and each dataset we performed hyper-parameter optimization (β_1, and β_2) using Myopic approach. For each dataset and different budget limit, there is a heatmap (see Figures 16 to 30). Each cell of heatmap shows AUCBE of average percentage of benefit attained up to the budget limit. β_1 variable ranges from 1 to 130, and β_2 ranges from 0 to 6 with 0.1 intervals. For the sake of better visualization, heatmaps are cropped at y-axis (β_1) ranges 1 to 61, 1 to 101, and 1 to 121 for datasets v6.3, v10.1, and v11.3 respectively. For all heatmaps the y-axis shows β_1 and x-axis shows β_2.

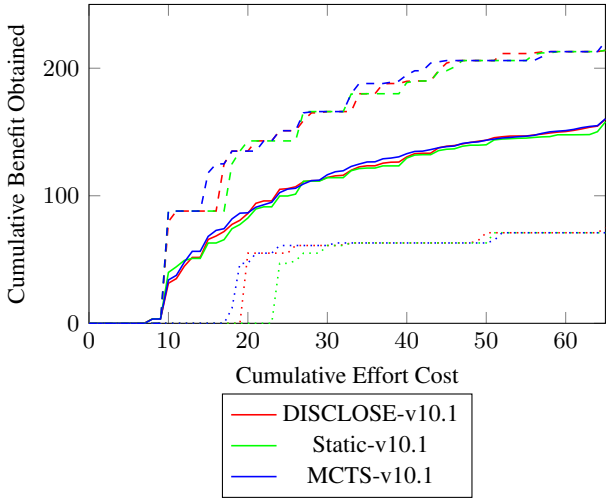

Figure 5: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 65**) on **v10.1**.

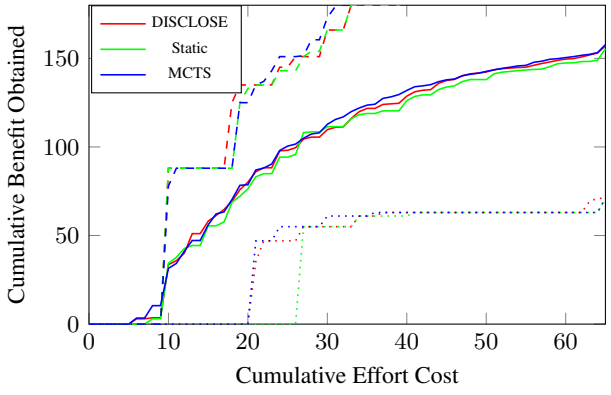

Figure 6: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 65**) on **v11.3**.

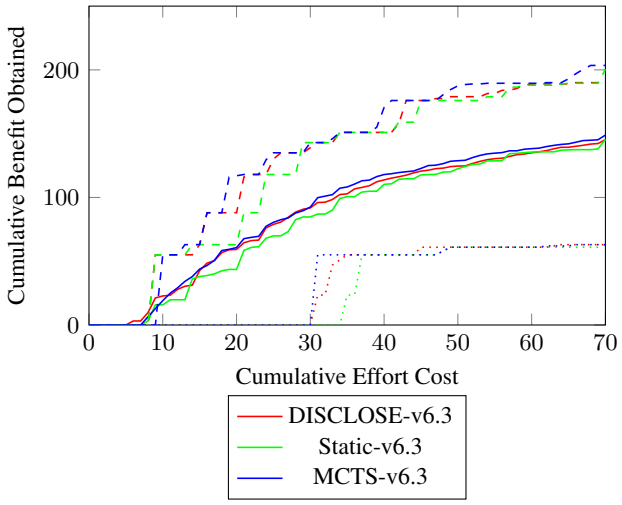

Figure 7: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 70**) on **v6.3**.

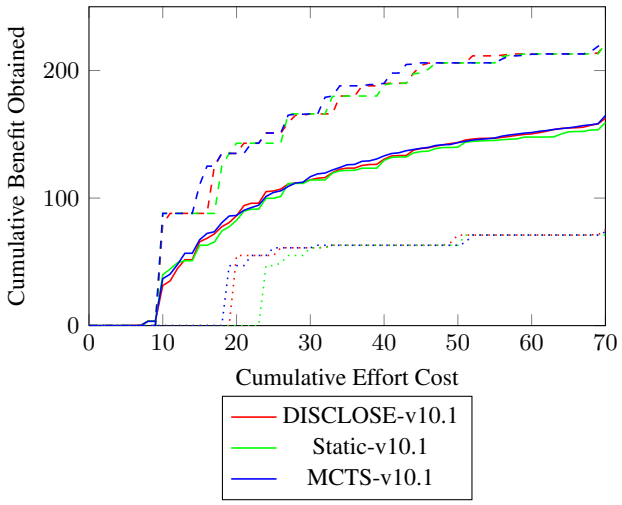

Figure 8: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 70**) on **v10.1**.

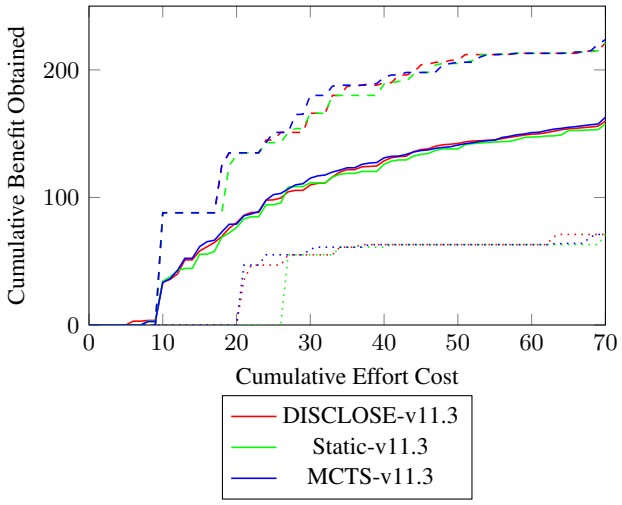

Figure 9: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 70**) on **v11.3**.

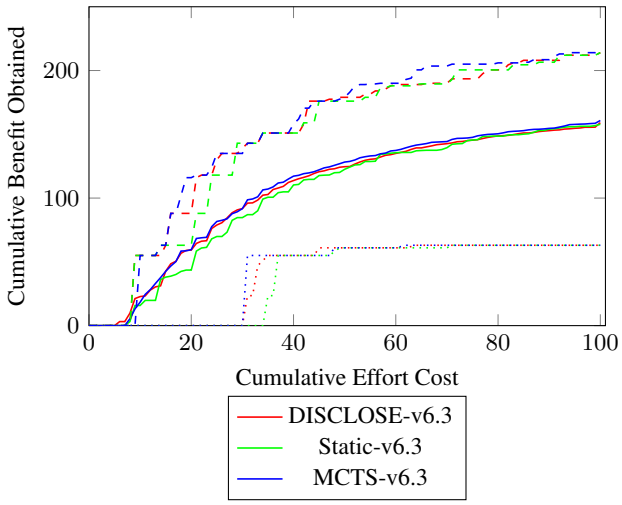

Figure 10: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 100**) on **v6.3**.

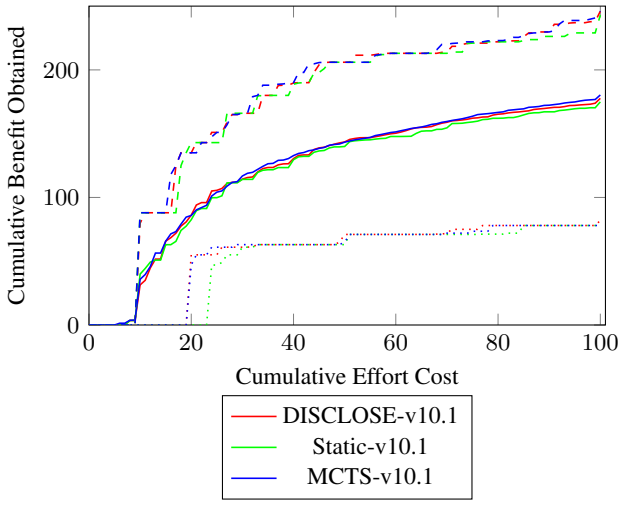

Figure 11: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 100**) on **v10.1**.

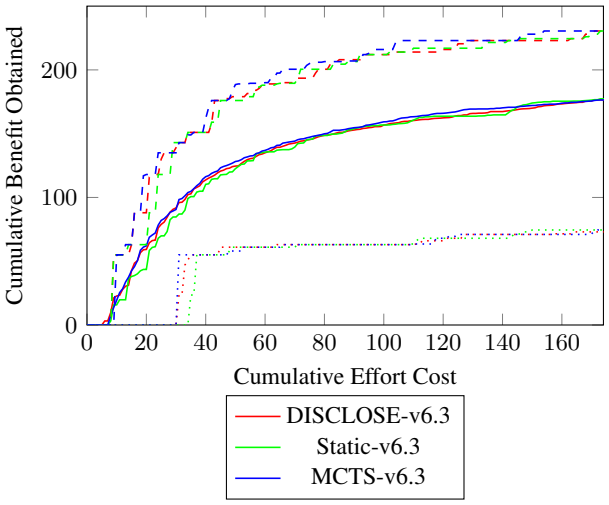

Figure 13: Cumulative benefit obtained as a function of cumulative effort cost (**without budget limitation**) on **v6.3**.

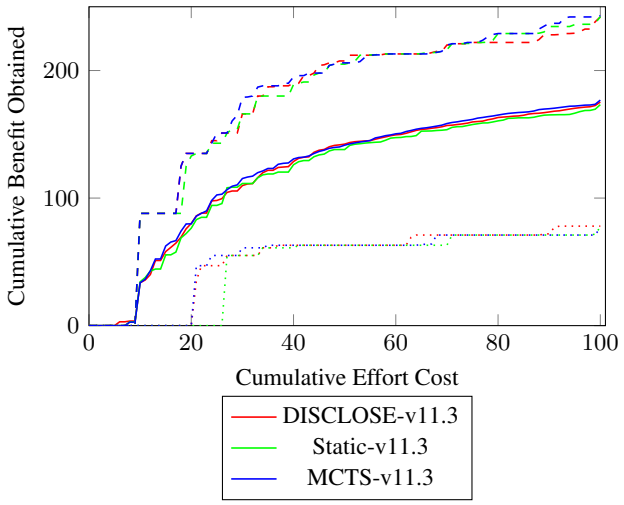

Figure 12: Cumulative benefit obtained as a function of cumulative effort cost (**up to budget 100**) on **v11.3**.

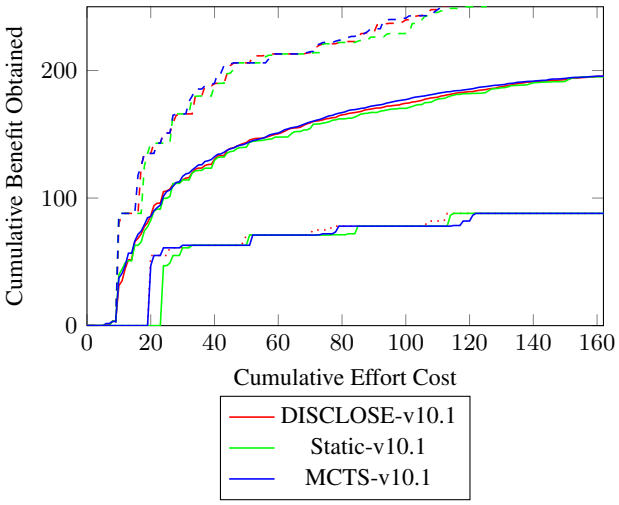

Figure 14: Cumulative benefit obtained as a function of cumulative effort cost (**without budget limitation**) on **v10.1**.

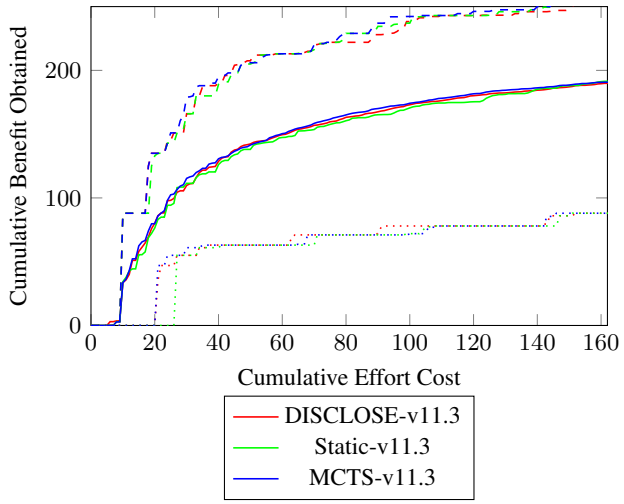

Figure 15: Cumulative benefit obtained as a function of cumulative effort cost (**without budget limitation**) on **v11.3**.

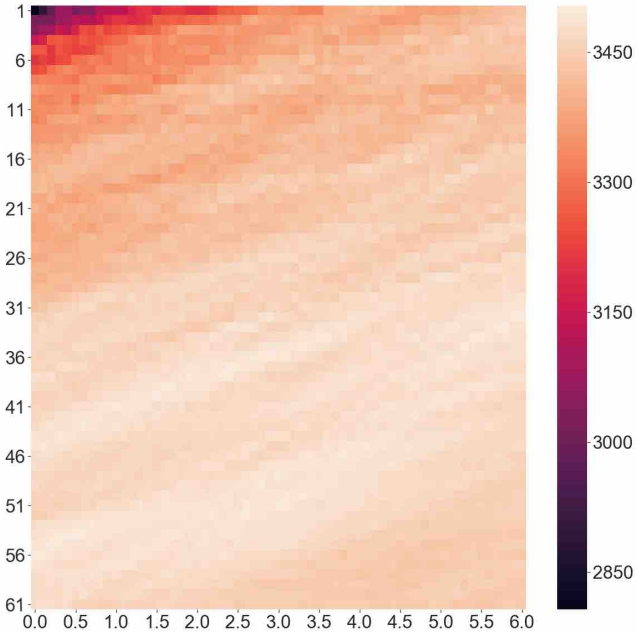

Figure 16: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 40 and 1.5 respectively) **up to budget 45** on **v6.3**

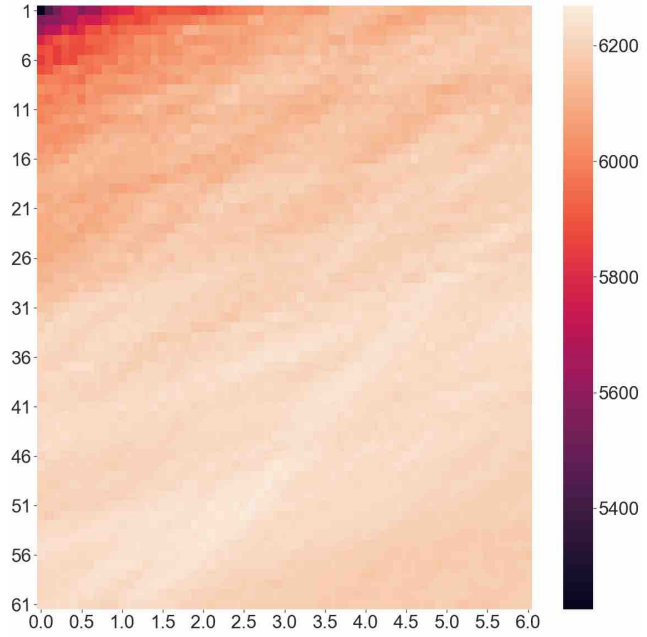

Figure 17: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 51 and 2.6 respectively) **up to budget 65** on **v6.3**

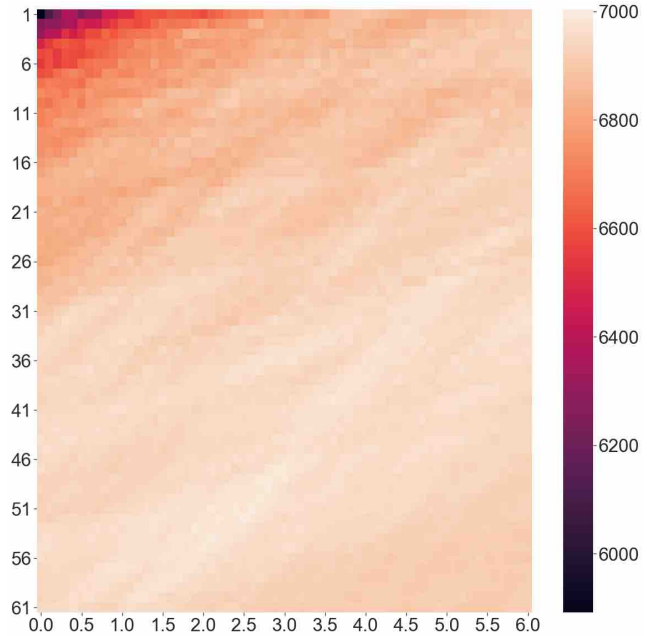

Figure 18: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 51 and 2.6 respectively) **up to budget 70** on **v6.3**

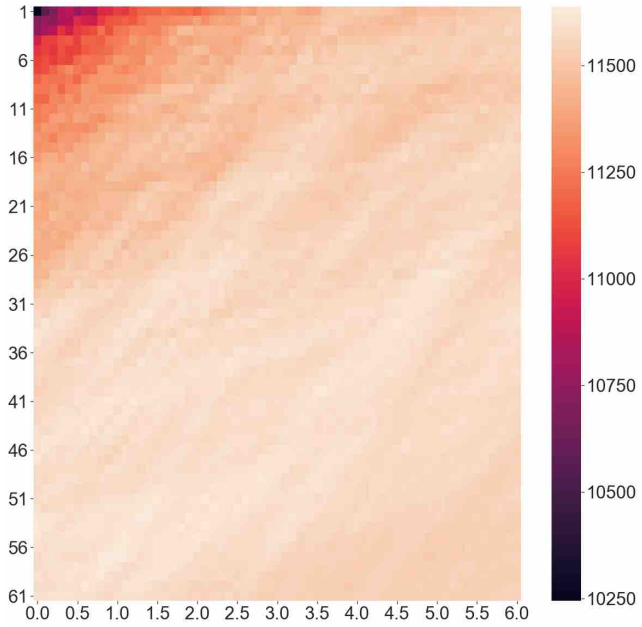

Figure 19: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 57 and 0.9 respectively) **up to budget 100** on **v6.3**

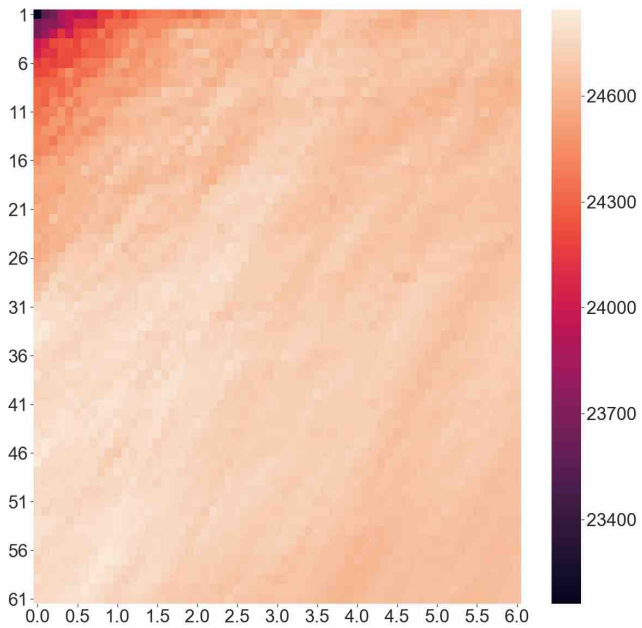

Figure 20: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 47 and 0 respectively) **without budget limitation** on **v6.3**

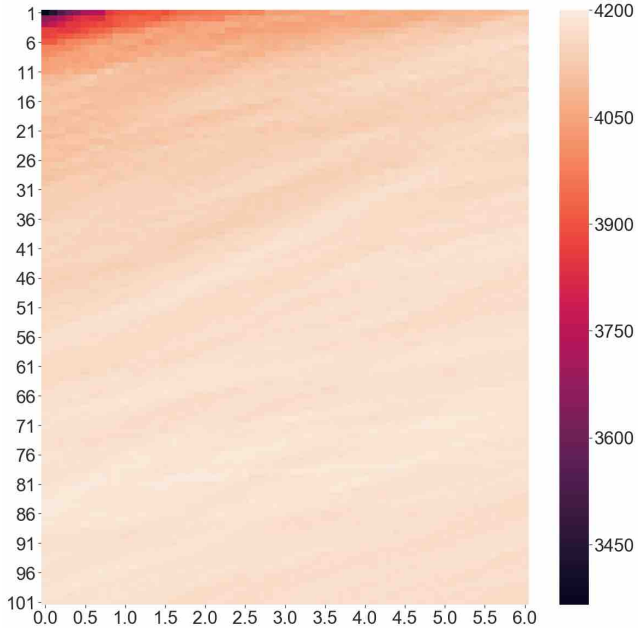

Figure 21: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 81 and 1.4 respectively) **up to budget 45** on **v10.1**

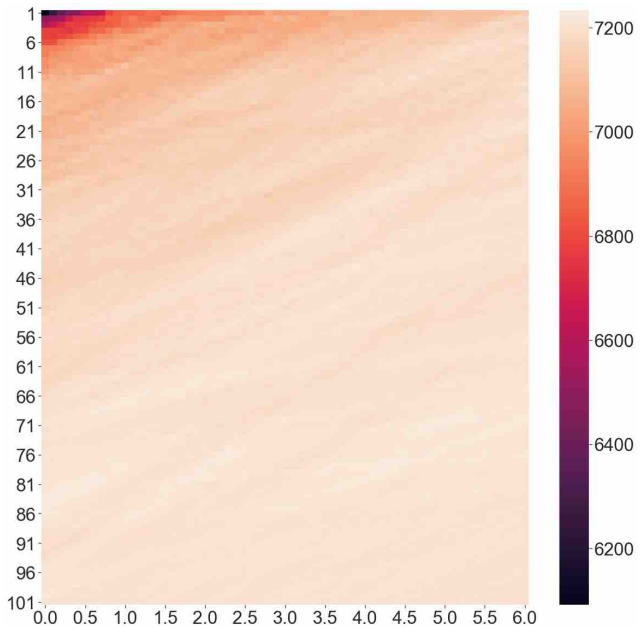

Figure 22: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 80 and 1.8 respectively) **up to budget 65** on **v10.1**

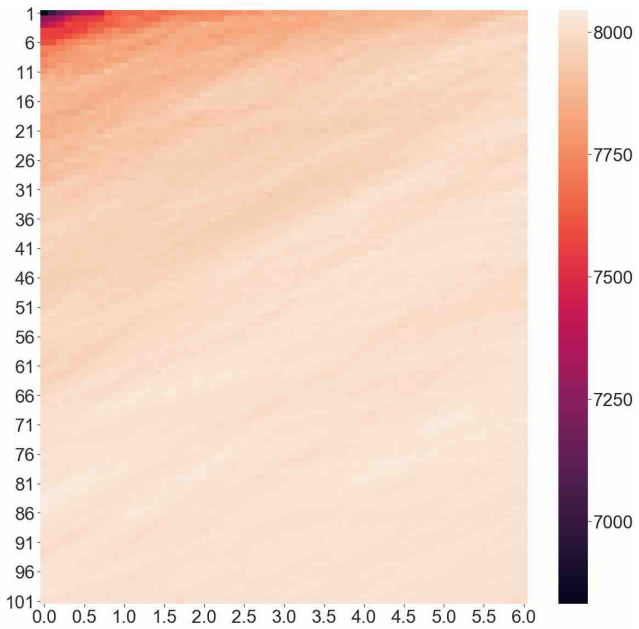

Figure 23: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 84 and 0 respectively) **up to budget 70** on **v10.1**

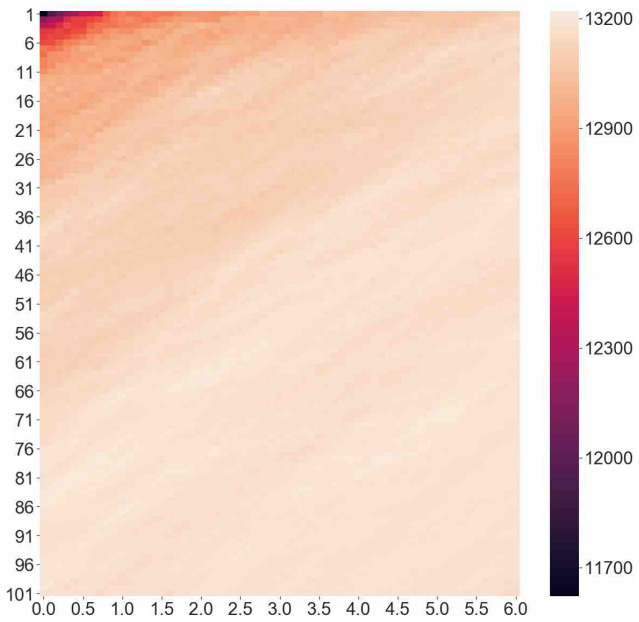

Figure 24: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 82 and 0.2 respectively) **up to budget 100** on **v10.1**

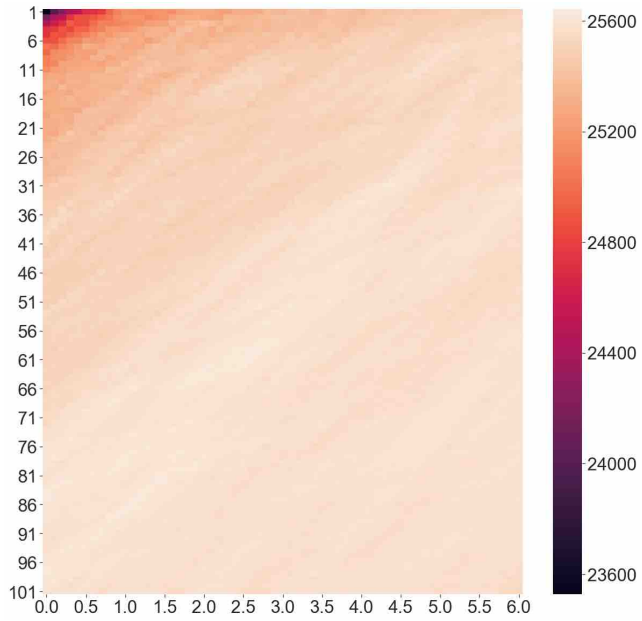

Figure 25: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 87 and 1 respectively) **without budget limitation** on **10.1**

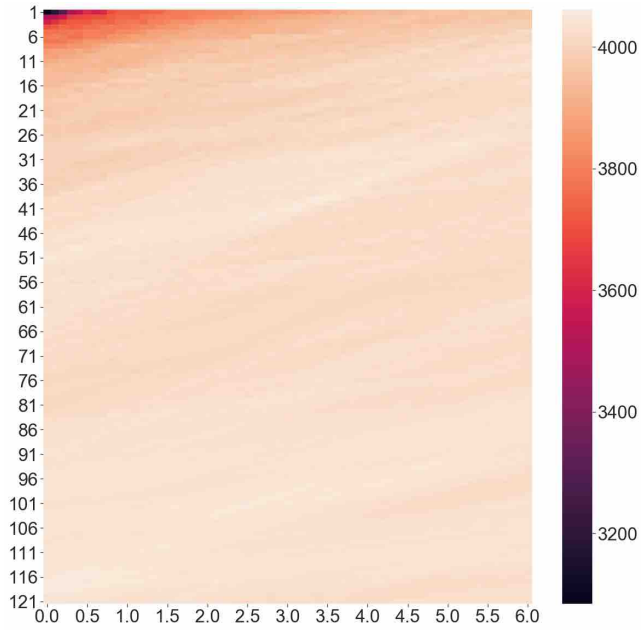

Figure 26: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 39 and 3.5 respectively) **up to budget 45** on **v11.1**

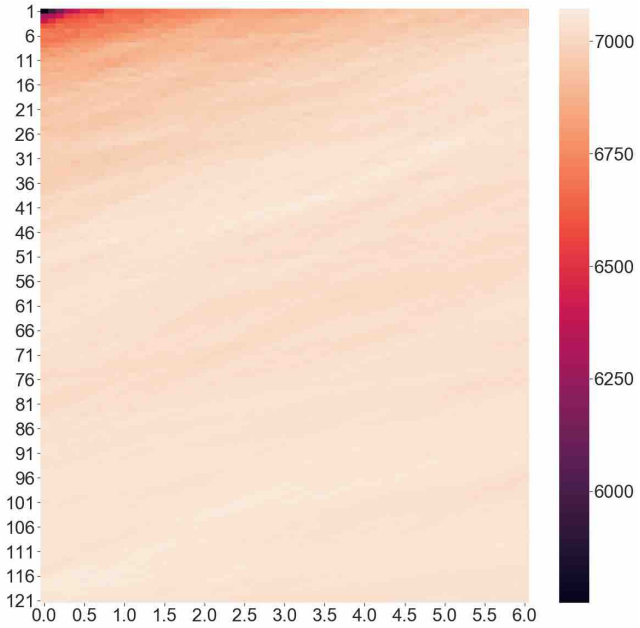

Figure 27: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 39 and 3.5 respectively) **up to budget 65** on **v11.1**

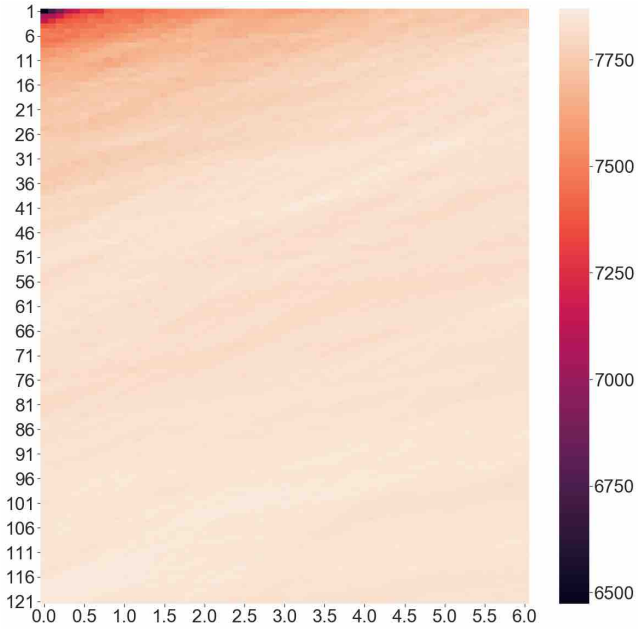

Figure 28: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 103 and 2.2 respectively) **up to budget 70** on **v11.1**

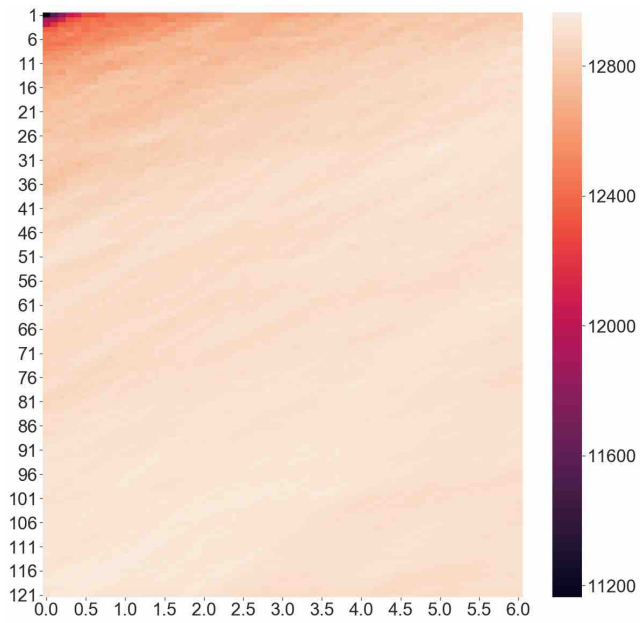

Figure 29: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 118 and 0.6 respectively) **up to budget 100** on **v11.1**

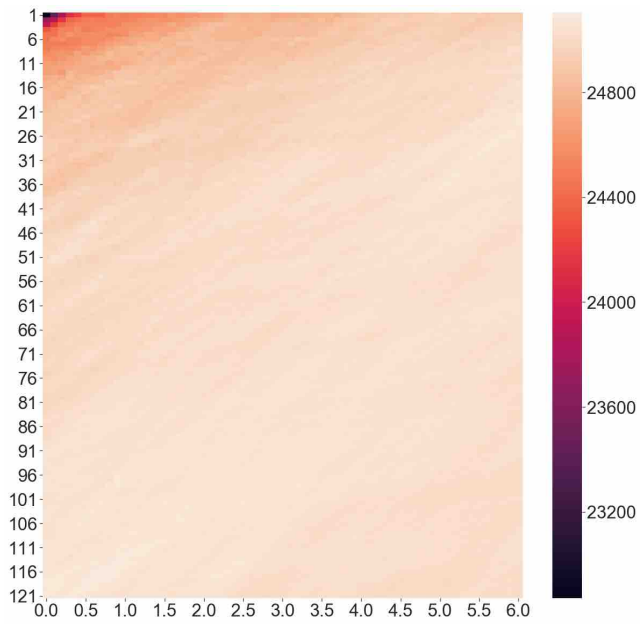

Figure 30: Benefit percentage attained AUCBE (optimal values for β_1 and β_2 are 118 and 0.6 respectively) **without budget limitation** on **11.1**