Online IT Ticket Automation Recommendation Using Hierarchical Multi-armed Bandit Algorithms

Qing Wang* Tao Li† S. S. Iyengar‡ Larisa Shwartz§ Genady Ya. Grabarnik¶

Abstract

The increasing complexity of IT environments urgently requires the use of analytical approaches and automated problem resolution for more efficient delivery of IT services. In this paper, we model the automation recommendation procedure of IT automation services as a contextual bandit problem with dependent arms, where the arms are in the form of hierarchies. Intuitively, different automations in IT automation services, designed to automatically solve the corresponding ticket problems, can be organized into a hierarchy by domain experts according to the types of ticket problems. We introduce a novel hierarchical multi-armed bandit algorithms leveraging the hierarchies, which can match the coarse-to-fine feature space of arms. Empirical experiments on a real large-scale ticket dataset have demonstrated substantial improvements over the conventional bandit algorithms. In addition, a case study of dealing with the cold-start problem is conducted to clearly show the merits of our proposed algorithms.

1 Introduction

1.1 Background Driven by the rapid changes in the economic environment, business enterprises constantly evaluate their competitive position in the market and attempt to come up with innovative ways to gain competitive advantage. Value-creating activities cannot be accomplished without solid and continuous delivery of IT services. The growing complexity of IT environments dictates an extensive use of analytic combined with automation. Incident management is one of the most critical processes in IT service management (ITSM) as it resolves incidents and restores provisioned services.

A typical workflow of ITSM is illustrated in Figure 1. It usually includes six steps: (1) As an anomaly detected, an event is generated and the monitoring emits the event if it persists beyond a predefined duration. (2) Events from an entire IT environment are consolidated in an enterprise event management system, which upon results of quick analysis, determines whether to create an alert and subsequently an incident ticket. (3) Tickets are collected by an IPC (Incident, Problem, and Change) system [22]. (4) A monitoring ticket, identified by IT automation services for potential automation (i.e., scripted resolution) based on the ticket description. In case the issue could not be completely resolved, this ticket is then escalated to human engineers. (5) In order to improve the performance of IT automation services and reduce human efforts for escalated tickets, the workflow incorporates an enrichment engine that uses data mining techniques (e.g., classification and clustering) for continuous enhancement of IT automation services. Additionally, the information is added to a knowledge base, which is used by the IT automation services as well as in resolution recommendation for tickets escalated to a human. (6) Manually created and escalated tickets are forwarded to human engineers for problem determination, diagnosis, and resolution, which is a very labor-intensive process.

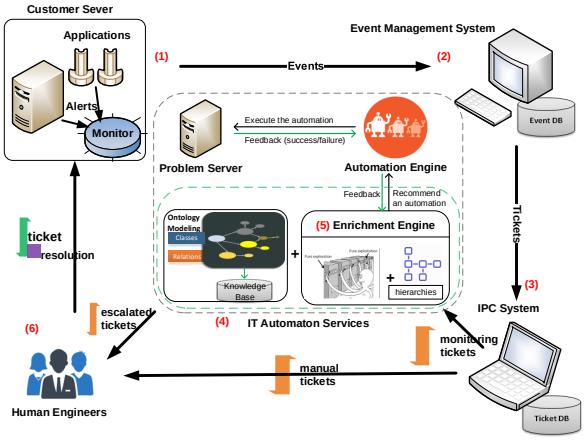

Figure 1: The overview of ITSM workflow.

1.2 Motivation In today's economic climate, IT service provider is expected to focus on innovation and

*School of Computing and Information Sciences, Florida International University. qwang028@cs.fiu.edu

†School of Computing and Information Sciences, Florida International University. taoli@cs.fiu.edu, deceased December 2017

‡School of Computing and Information Sciences, Florida International University. iyengar@cis.fiu.edu

§IBM T.J. Watson Research Center. lshwart@us.ibm.com

¶Dept. Math & Computer Science, St. John's University. grabarng@stjohns.edu

assisting customers in their core business areas. Time the experts spend on fixing operational issues has to be minimized. Enterprise IT Automation Services [2] have been introduced into ITSM as an engine for automated corrective actions (i.e., scripted resolutions) and closure of incident records.

Figure 2 shows an example of an IT service management ticket that was automatically generated by a monitoring system, and successfully fixed by the automation engine. The summary and monitoring class (i.e., an alert key) of the ticket provides an initial symptom description, which is used for automation service to identify existing automation or lack thereof. If the problem is resolved by the recommended automation, the value of "AUTORESOVLED" will be marked non-zero. To improve the efficiency of the recommending strategies of the automation engine, it is essential to understand how the symptoms could be mapped to the corresponding scripted resolutions. This is the initial motivation for our study. Based on preliminary studies [22, 19, 28, 18], we have identified three key challenges in virtual engineering technology.

ALERT_KEY	cpc_cpuutil_gntw_win_v3		AUTOMATON_NAME		CPC:WIN:GEN:R:W:System Load Handler		
OPEN_DTTM	CLIENT_ID	HOSTNAME	ORIGINAL SEVERITY	OSTYPE	COMPONET	SUBCOMP OMET	AUTO RESOVLED
2016-04-30 12:43:07	136	LEXSBWS01 VH	2	WIN	WINDOWS	CPU	1
TICKET SUMMARY	CPU Workload High. CPU 1, busy 99% time.		TICKET RESOLUTION		The CPU Utilization was quite reduced, hence closing the ticket.		

Figure 2: A sample ticket in ITSM and its corresponding automaton.

Challenge 1. *How do we appropriately solve the well-known cold-start problem in IT automation services?* Most recommender systems suffer from a cold-start problem. This problem is critical since every system could encounter a significant number of users/items that are completely new to the system with no historical records at all. The cold-start problem makes recommender systems ineffective unless additional information about users/items is collected [13, 6], which is a crucial problem for automation engine as well, since it cannot make any effective recommendation that translates into significant human efforts. Multi-armed bandit algorithm can address the cold-start problem, which balances the tradeoff between *exploration* and *exploitation*, hence, maximizing the opportunity for fixing the tickets, while gathering new information for improving the goodness of the ticket and automation matching.

Challenge 2. *How do we utilize the interactive feedback to adaptively optimize the recommending strategies of the enterprise automation engine to enable a quick problem determination by IT automation services?* The automation engine (see Figure 1) automatically takes action based on the contextual information of the ticket and observes the execution feedback (e.g., success or failure) from the problem server. The current strategies of the automation engine do not take advantage of these interactive information for continuous improvement. Based on the aforementioned discussion, we present an online learning problem of recommending an appropriate automation and constantly adapting the up-to-date feedback given the context of the incoming ticket. This can be naturally modeled as a contextual multi-armed bandit problem, which has been widely applied into various interactive recommender systems [10, 27, 25]. To the best of authors' knowledge, it is the first study to formulate the strategies of the automation recommendation in IT automation services as a contextual bandit problem.

Challenge 3. *How do we efficiently improve the performance of recommendation using the automation hierarchies of IT automation services?* Domain experts usually define the taxonomy (i.e., hierarchy) of the IT problems explicitly (see Figure 3). Correspondingly, the scripted resolutions (i.e., automations) also contain the underlying hierarchical problems' structure.

For example, a ticket is generated due to a failure of the DB2 database. The root cause may be database deadlock, high usage or other issues. Intuitively, if the problem was initially categorized as a database problem, the automated ticket resolutions have a much higher probability to fix this problem, than if it hasn't been categorized as such, and all other categories (e.g., file system and networking) are now taken into consideration. We formulate this as a contextual bandit problem with dependent arms organized hierarchically, which can match the feature spaces from a coarse level first, and then be refined to the next lower level of taxonomy. The existing bandit algorithms can only explore the flat feature spaces by assuming the arms are independent.

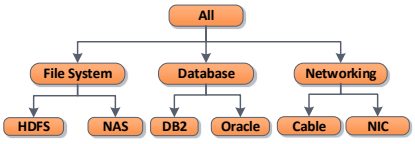

Figure 3: An example of taxonomy in IT tickets.

1.3 Contribution To the best of our knowledge, this paper is the first work to formulate the automation recommendation in IT automation services as a multi-armed bandit problem by considering the dependencies among arms in the form of hierarchies. We demonstrate this approach on the automation recommendation for IT service management. The contribution mainly focuses on proposing hierarchical multi-armed bandit algorithms to overcome the aforementioned three key challenges. The key features of our contribution include:

- A new online learning approach, designed to (1) solve the cold-start problem, and (2) continuously recommend an appropriate automation for the incoming ticket and adapt based on the feedback to improve the goodness of match between the problem and automation in IT automation services.
- Utilization of the hierarchies, integrated into bandit algorithms to model the dependencies among arms.

The effectiveness and efficiency of our proposed methods are verified on a large dataset of tickets from IBM Global Services.

The remainder of this paper is organized as follows. The related work is presented in Section 2. In Section 3, we give the mathematical formalization of the problem. The solution to the problem is provided in Section 4. Section 5 describes comparative experiments and an empirical case study conducted over the real ticket data, which demonstrate the efficacy of the proposed algorithms. Finally, Section 6 summarizes and concludes the paper.

2 Related Work

In this section, we provide a short survey of the literature related to the automated IT service management and multi-armed bandit algorithms with dependent arms.

The automation of IT service management is largely achieved through service-providing facilities in combination with automation of subroutine procedures such as *problem detection*, *problem determination*, and *ticket resolution recommendation* for the service infrastructure. Automatic problem detection is typically realized by system monitoring software, such as IBM Tivoli Monitoring [3] and HP OpenView [1]. Tang et al. [17] proposed an integrated framework to minimize the false positive and maximize the coverage for system fault detection to optimize this procedure. For problem determination, a hierarchical multi-label classification method [22, 26] was proposed to classify the problem types in the monitoring IT tickets. In order to determine the root cause, authors [23, 21, 24, 25] analyzed the historical events to reveal the underlying temporal causal relationship between these sequential data. Automated ticket resolution recommendation [16] is a big challenge in IT service management since it requires vast domain knowledge about the target infrastructure.

Traditional recommendation technologies in ITSM focus on recommending the proper resolutions to a ticket reported by the system's user. Recently, Wang et al. [19] proposed a cognitive framework that enables an automation improvement through resolution recommendation utilizing the ontology modeling technique. A deep neural network ranking model [28] was utilized for recommendation of the best top-n matching resolutions by quantifying the qualify of each historical resolution. However, previous literature tried to resolve a ticket reported by the system's user. To the best of our knowledge, none of the existing studies has attempted to address the aforementioned challenges for those tickets automatically generated by a monitoring system, and it is the first time to leverage the multi-armed bandit model to optimize the online automation procedure via feedback in IT automation services.

Most recently, an interactive recommender system has become increasingly significant due to the abundance of online services. For an individual user, they continuously refine the recommendation results using up-to-date feedback [27]. The tradeoff between *exploration* and *exploitation* inherent in learning from interactive feedback has been well dealt with using contextual multi-armed bandit algorithms [14, 7, 10, 15, 25]. However, most prior work (e.g., Upper Confidence Bound (UCB) [5] and Thompson sampling [7]) assumes independent arms, which rarely holds true in reality. Since the real-world items tend to be correlated with each other, a delicate framework [12] is developed to study the bandit problem with dependent arms. In light of the topic modeling techiques, Wang et al. [18] come up with a new generative model to explicitly formulate the item dependencies as the clusters on arms. Pandey et al. [11] used the taxonomy structure to exploit dependencies among the arm in the context-free bandit setting. CoFineUCB approach [20] utilized a coarse-to-fine feature hierarchy to reduce the cost of exploration, where the hierarchy was estimated by a small number of existing user profiles. In contrast, we study the contextual bandit problem with a given hierarchical structure of arms, where the hierarchy is constructed by domain experts based on the features of items.

3 Problem Formulation

In this section, we provide a mathematical formulation of the problem and describe the new contextual multi-armed bandit algorithms, which can utilize a taxonomy defined by domain experts explicitly depicting the dependencies among arms. A glossary of notations mentioned in this paper is summarized in Table 1.

3.1 Basic Concepts and Terminologies Let $\mathcal{A} = \{a^{(1)}, ..., a^{(K)}\}$ denote a set of automations (i.e., scripted resolutions) feasible in IT automation system, where K is the number of the automations. Every time a ticket is reported, the online IT automation recommendation process selects an automation $a^{(i)} \in \mathcal{A}$ using contextual information (i.e., the symptom description in the ticket)

Table 1: Important Notations

Notation	Description
$a^{(i)}$	the i-th arm.
$\mathcal{A}$	the set of arms, $\mathcal{A} = \{a^{(1)}, ..., a^{(K)}\}$.
$\mathcal{H}$	the hierarchy (taxonomy) defined by domain experts.
$\mathcal{X}$	d -dimensional context feature space.
$\mathbf{x}_t$	the context at time t.
$r_{k,t}$	the reward (payoff) of pulling the arm $a^{(k)}$ at time t.
$\hat{r}_{k,t}$	the predicted reward (payoff) for the arm $a^{(k)}$ at time t.
π	the policy for pulling arm sequentially.
R_π	the cumulative reward of the policy π.
$\mathbf{S}_{\pi,t}$	the sequence of $(\mathbf{x}_i, \pi(\mathbf{x}_i), r_{\pi(\mathbf{x}_i)})$ observed until time $t = 1, ..., T$.
θ_k	the coefficients predicting reward of the arm $a^{(k)}$.
σ_k^2	the reward prediction variance for arm $a^{(k)}$.
α, β	the parameters of the distribution of σ_k^2.
$\mu_\theta, \mathbf{\Sigma}_\theta$	the parameters of the distribution of θ.

and recommends it as a possible resolution for the ticket. Specifically, the contextual information for the reported ticket at time t is represented as a feature vector $\mathbf{x}_t \in \mathcal{X}$, where $\mathcal{X}$ denotes the d-dimensional feature space. After recommending an IT automation $a^{(k)}$ at time t, its corresponding feedback is received, indicating whether the ticket has been successfully resolved or not.

We formalize the online IT automation recommendation process as a contextual multi-armed bandit problem where automations are constantly recommended and the underlying recommendation model is instantly updated based on the feedback collected over time. In general, a contextual multi-armed problem involves a series of decisions over a finite but possibly unknown time horizon T. In our formalization, each automation corresponds to an arm. Pulling an arm indicates its corresponding automation is being recommended, and the feedback (e.g., success or failure) received after pulling the corresponding arm is used to compute the reward.

In the contextual multi-armed bandit setting, at each time $t = [1, T]$, a policy π makes a decision for selecting an automation $\pi(\mathbf{x}_t) \in \mathcal{A}$ to perform an action according to the contextual vector $\mathbf{x}_t$ of the current ticket. Let $r_{k,t}$ denote the reward for recommending an automation $a^{(k)}$ at time t, whose value is drawn from an unknown distribution determined by the context $\mathbf{x}_t$ presented to automation $a^{(k)}$. The total reward received by the policy π after T iterations is

$$R_\pi = \sum_{t=1}^{T} r_{\pi(\mathbf{x}_t)}. \tag{3.1}$$

The optimal policy π^* is defined as the one with maximum accumulated expected reward after T iterations,

$$\pi^* = \arg\max_\pi E(R_\pi) = \arg\max_\pi \sum_{t=1}^{T} E(r_{\pi(\mathbf{x}_t)}|t). \tag{3.2}$$

Our goal is to identify a good policy for maximizing the total reward. Herein we use reward instead of regret to express the objective function, since maximization of the cumulative reward is equivalent to minimization of regret during the T iterations [27].

Before selecting the optimal automation at time t, a policy π is updated to refine a model for reward prediction of each automation according to the historical observations $S_{\pi,t-1} = \{(\mathbf{x}_i, \pi(\mathbf{x}_i), r_{\pi(\mathbf{x}_i)}) | i = [1, t-1]\}$. The reward prediction helps to ensure that the policy π includes decisions to increase the total reward. The reward $r_{k,t}$ is typically modeled as a linear combination of the feature vector $\mathbf{x}_t$ given as follows:

$$r_{k,t} = \mathbf{x}_t^T \theta_k + \xi_k, \tag{3.3}$$

where θ_k is a d-dimensional coefficient vector, and ξ_k denotes an observation noise, a zero-mean Gaussian noise with variance σ_k^2, i.e., $\xi_k \sim \mathcal{N}(0, \sigma_k^2)$. Then,

$$r_{k,t} \sim \mathcal{N}(\mathbf{x}_t^T \theta_k, \sigma_k^2), \tag{3.4}$$

and our objective function in Equation 3.2 can be reformulated as:

$$\pi^* = \arg\max_\pi \sum_{t=1}^{T} E_{\theta_{\pi(\mathbf{x}_t)}}(\mathbf{x}_t^T \theta_{\pi(\mathbf{x}_t)} | t). \tag{3.5}$$

To address the aforementioned problem, contextual multi-armed bandit algorithms have been proposed to balance the tradeoff between exploration and exploitation for arm selection, including ϵ-greedy, Thompson sampling, LinUCB, etc.

Thompson sampling is one of the earliest heuristic methods to address the contextual bandit problems, belonging to the probability matching family [4]. The main idea is to allocate the pulling chance according to the probability that an arm produces the maximum expected reward given the context $\mathbf{x}_t$ at time t. Particularly, Thompson sampling method learns and maintains the posterior distribution of the parameters in the reward prediction model for each arm. At every time t, Thompson sampling first samples the model parameters from its posterior distribution learnt at time $t-1$. The sampled parameters together with the contextual information $\mathbf{x}_t$ are used for reward prediction. The arm with maximum predicted reward is then selected to pull. Based on the feedback after pulling at time t, the posterior distribution of the model parameters for the selected arm at time t is updated and ready for arm selection at time $t+1$.

LinUCB [10], an extension of the UCB algorithm [5], is another contextual bandit algorithm. It pulls the arm with the largest score computed by combining both reward expectation and deviation, which are computed in light of the reward prediction model.

Although different multi-armed bandit algorithms have been proposed and extensively adopted in diverse real applications, most of them do not take the dependencies between arms into account. In the IT environment, the automations (i.e., arms) are organized with its taxonomy, i.e., a hierarchical structure. The following section will introduce our approach to make use of the arm dependencies in the bandit settings for IT automation recommendation optimization.

3.2 Automation Hierarchy In IT automation services, the automations can be classified with a predefined taxonomy. It allows us to reformulate the problem as a bandit model with the arm dependencies described by a tree-structured hierarchy.

Let $\mathcal{H}$ denote the taxonomy, which contains a set of nodes (i.e., arms) organized in a tree-structured hierarchy. Given a node $a^{(i)} \in \mathcal{H}$, $pa(a^{(i)})$ and $ch(a^{(i)})$ are used to represent the parent and children sets, respectively. Accordingly, we have Property 3.1.

PROPERTY 3.1. *If $pa(a^{(i)}) = \emptyset$, node $a^{(i)}$ is assumed to be the root node. If $ch(a^{(i)}) = \emptyset$, then $a^{(i)}$ is a leaf node, which represents an automation. Otherwise, $a^{(i)}$ is a category node when $ch(a^{(i)}) \neq \emptyset$.*

Since the goal is to recommend an automation for ticket resolving and only a leaf node of $\mathcal{H}$ represents an automation, the recommendation process cannot be completed until a leaf node is selected at each time t. Therefore, the multi-armed bandit problem for IT automation recommendation is reduced to select a path of $\mathcal{H}$ from root to a leaf node, where multiple arms along the path are sequentially selected with respect to the contextual vector $\mathbf{x}_t$ at time t.

Let $pth(a^{(i)})$ be a set of nodes, consisting of all the nodes along the path from root node to $a^{(i)}$ in $\mathcal{H}$. Further, assume $\pi_{\mathcal{H}}(\mathbf{x}_t|t)$ to be the path selected by policy π in light of the contextual information $\mathbf{x}_t$ at time t. Hence, we can have Property 3.2 for every arm selection policy π.

PROPERTY 3.2. *Given the contextual information $\mathbf{x}_t$ at time t, if a policy π selects a node $a^{(i)}$ in the hierarchy $\mathcal{H}$ and receives positive feedback (i.e., success), the policy π receives positive feedback as well by selecting the nodes in $pth(a^{(i)})$.*

Let $r_{\pi_{\mathcal{H}}(\mathbf{x}_t|t)}$ denote the reward obtained by the policy π after selecting the multiple arms along the path $\pi_{\mathcal{H}}(\mathbf{x}_t|t)$ at time t. The reward is computed as follows,

$$r_{\pi_{\mathcal{H}}(\mathbf{x}_t|t)} = \sum_{a^{(i)} \in \pi_{\mathcal{H}}(\mathbf{x}_t|t), ch(a^{(i)}) \neq \emptyset} r_{\pi(\mathbf{x}_t|ch(a^{(i)}))}, \tag{3.6}$$

where $\pi(\mathbf{x}_t|ch(a^{(i)}))$ represents the arm selected from the children of $a^{(i)}$, given the contextual information $\mathbf{x}_t$.

Therefore, after T iterations, the total reward received by the policy π is computed as below,

$$R_{\pi_{\mathcal{H}}} = \sum_{t=1}^{T} r_{\pi_{\mathcal{H}}(\mathbf{x}_t|t)}. \tag{3.7}$$

The optimal policy π^* with respect to $\mathcal{H}$ is determined by

$$\pi^* = \arg\max_{\pi} E(R_{\pi_{\mathcal{H}}}) = \arg\max_{\pi} \sum_{t=1}^{T} E(r_{\pi_{\mathcal{H}}(\mathbf{x}_t)}|t). \tag{3.8}$$

The reward prediction for each arm is conducted by Equation 3.4, and then the optimal policy can be equivalently determined by

$$\pi^* = \arg\max_{\pi} \sum_{t=1}^{T} \Big(\sum_{\substack{a^{(i)} \in \pi_{\mathcal{H}}(\mathbf{x}_t|t), \\ ch(a^{(i)}) \neq \emptyset}} E_{\theta_{\pi(\mathbf{x}_t|ch(a^{(i)}))}}(\mathbf{x}_t^T \theta_{\pi(\mathbf{x}_t|ch(a^{(i)}))}|t) \Big) \tag{3.9}$$

Both Thompson sampling and LinUCB mentioned above will be incorporated into our new learning models that leverage the hierarchies defined by domain experts. In such settings, bandit algorithms can achieve faster convergence by exploring feature space hierarchically.

4 Solution & Algorithm

In this section, we propose the HMAB (Hierarchical Multi-Armed Bandit) algorithms for exploiting the dependencies among arms organized hierarchically.

As presented in Equation 3.4, the reward $r_{k,t}$ depends on random variable $\mathbf{x}_t$, θ_k and σ_k^2. We assume θ_k and σ_k^2 follow a conjugate prior distribution, Normal Inverse Gamma (abbr., $\mathcal{NIG}$) distribution. σ_k^2 is drawn from the Inverse Gamma (abbr., $\mathcal{IG}$) distribution shown in Equation 4.10.

$$p(\sigma_k^2|\alpha_k, \beta_k) \sim \mathcal{IG}(\alpha_k, \beta_k), \tag{4.10}$$

where α_k and β_k are the predefined hyper parameters for the $\mathcal{IG}$ distribution. Given σ_k^2, the coefficient vector θ_k is generated by a Gaussian prior distribution with the hyper parameter μ_{θ_k} and $\mathbf{\Sigma}_{\theta_k}$:

$$p(\theta_k|\mu_{\theta_k}, \mathbf{\Sigma}_{\theta_k}, \sigma_k^2) \sim \mathcal{N}(\mu_{\theta_k}, \sigma_k^2 \mathbf{\Sigma}_{\theta_k}), \tag{4.11}$$

At each time t, a policy π will select a path $\pi_{\mathcal{H}}(\mathbf{x}_t|t)$ from $\mathcal{H}$ according to the context $\mathbf{x}_t$. Assuming $a^{(p)} \in \pi_{\mathcal{H}}(\mathbf{x}_t|t)$ is the leaf node (i.e., an automaton), then we have $pth(a^{(p)}) = \pi_{\mathcal{H}}(\mathbf{x}_t|t)$. After recommending the automation $a^{(p)}$, a reward $r_{p,t}$

is obtained. Since the reward $r_{p,t}$ is shared by all the arms along the path $pth(a^{(p)})$, a set of triples $F = \{(\mathbf{x}_t, a^{(k)}, r_{k,t}) | a^{(k)} \in pth(a^{(k)}), r_{k,t} = r_{p,t}\}$ are acquired. A new sequence $S_{\pi,t}$ is generated by incorporating the triple set F into $S_{\pi,t-1}$. The posterior distribution for every $a^{(k)} \in pth(a^{(k)})$ needs to be updated with the new feedback sequence $S_{\pi,t}$. The posterior distribution of θ_k and σ_k^2 given $S_{\pi,t}$ is a $\mathcal{NIG}$ distribution with the hyper parameter μ_{θ_k}, $\mathbf{\Sigma}_{\theta_k}$, α_k and β_k. These hyper parameters at time t are updated based on their values at time $t-1$:

(4.12)

$$\begin{aligned}
\mathbf{\Sigma}_{\theta_{k,t}} &= (\mathbf{\Sigma}_{\theta_{k,t-1}}^{-1} + \mathbf{x}_t \mathbf{x}_t^T)^{-1} \\
\mu_{\theta_{k,t}} &= \mathbf{\Sigma}_{\theta_{k,t}} (\mathbf{\Sigma}_{\theta_{k,t-1}}^{-1} \mu_{\theta_{k,t-1}} + \mathbf{x}_t r_{k,t}) \\
\alpha_{k,t} &= \alpha_{k,t-1} + \frac{1}{2} \\
\beta_{k,t} &= \beta_{k,t-1} + \\
&\quad \frac{1}{2}[r_{k,t}^2 + \mu_{\theta_{k,t-1}}^T \mathbf{\Sigma}_{\theta_{k,t-1}}^{-1} \mu_{\theta_{k,t-1}} - \mu_{\theta_{k,t}}^T \mathbf{\Sigma}_{\theta_{k,t}}^{-1} \mu_{\theta_{k,t}}]
\end{aligned}$$

Algorithm 1 The algorithms for HMAB

procedure MAIN($\mathcal{H}, \pi, \lambda$) ▷ main entry, π is the policy
  for $t \leftarrow 1, T$ **do**
    Initialize parameters of $a^{(m)} \in \mathcal{H}$ to α_m, β_m, $\mathbf{\Sigma}_{\theta_m} = \mathbf{I}_d$, $\mu_{\theta_m} = \mathbf{0}_{d\times 1}$
    Get contextual vector $\mathbf{x}_t \in \mathcal{X}$
    for each path P of $\mathcal{H}$ **do**
      Compute the reward of P using Equation 3.6, by calling EVAL($\mathbf{x}_t, a^{(k)}, \pi$) for each arm $a^{(k)} \in P$
    end for
    Choose the path P^* with maximum reward
    Recommend the automation $a^{(*)}$ (leaf node of P^*)
    Receive reward $r_{*,t}$ by pulling arm $a^{(*)}$
    UPDATE($\mathbf{x}_t, P^*, r_{*,t}, \pi$)
  end for
end procedure

procedure EVAL($\mathbf{x}_t$, $a^{(k)}$, π) ▷ get a score for $a^{(k)}$, given $\mathbf{x}_t$
  if π is TS **then**
    Sample $\sigma_{k,t}^2$ according to Equation 4.10
    Sample $\theta_{k,t}$ according to Equation 4.11
    return $\hat{r}_{k,t} = \mathbf{x}_t^T \theta_{k,t}$
  end if
  if π is LinUCB **then**
    return $\hat{r}_{k,t} = \mathbf{x}_t^T \mu_{\theta_{k,t-1}} + \frac{\lambda}{\sigma_{k,t-1}} \sqrt{\mathbf{x}_t^T \mathbf{\Sigma}_{\theta_{k,t-1}}^{-1} \mathbf{x}_t}$
  end if
end procedure

procedure UPDATE($\mathbf{x}_t, P, r_t, \pi$) ▷ update the inference. ▷ P is the path in $\mathcal{H}$, r_t is the reward.
  for each arm $a^{(k)} \in P$ **do**
    Update $\alpha_{k,t}$, $\beta_{k,t}$, $\mathbf{\Sigma}_{\theta_{k,t}}$, $\mu_{\theta_{k,t}}$ using 4.12
  end for
end procedure

Note that the posterior distribution of θ_k and σ_k^2 at time t is considered as the prior distribution of time $t+1$. On the basis of the aforementioned inference of the leaf node $a^{(k)}$, we propose HMAB algorithms presented in Algorithm 1 developing different strategies including HMAB-TS($\mathcal{H}$, α, β) and HMAB-LinUCB($\mathcal{H}$, λ).

Online inference of our hierarchical bandit problem starts with MAIN procedure. As a ticket $\mathbf{x}_t$ arrives at time t, the EVAL procedure computes a score for each arm of different levels. In each level, the arm with the maximum score is selected to be pulled. After receiving reward by pulling an arm, the new feedback is used to update the HMAB algorithms by the UPDATE procedure.

5 Experiment Setup

To demonstrate the efficiency of our proposed algorithms, we conduct a large scale experimental study over a real ticket dataset from IBM Global Services. First, we outline the general implementation of the baseline algorithms for comparison. Second, we describe the dataset and evaluation method. Finally, we discuss the comparative experimental results of the proposed and baseline algorithms, and present a case study to demonstrate the effectiveness of HMAB algorithms.

5.1 Baseline Algorithms In the experiments, we demonstrate the performance of our methods by comparing the following baseline algorithms:

1. `Random`: a random item recommended to the targeted user without considering the contextual information.
2. `EpsGreedy`(ϵ): a random arm with probability ϵ selected, as well as the arm of the largest predicted reward $\hat{r}_{k,t}$ with probability $1-\epsilon$, where ϵ is a predefined parameter.
3. `TS`(α, β): Thompson sampling described in Section 3.1, randomly draws the coefficients from the posterior distribution, and selects the item with the largest estimated payoff according to Equation 3.4. Both α and β are hyper parameters. We initial α and β with the same value.
4. `LinUCB`(λ): the parameter λ is used to calculate the score, a linear combination of the expectation and deviation of the reward. The arm with the largest score is selected. When $\lambda = 0$, it is equivalent to the `Exploit` policy.

Our methods proposed in this paper include:

1. `HAMB-EpsGreedy`($\mathcal{H}$, ϵ): a random arm with probability ϵ is selected, and the arm of the highest estimated reward $\hat{r}_{k,t}$ with probability $1-\epsilon$ with respect to the hierarchy $\mathcal{H}$, which is a predefined parameter as well as ϵ.

2. `HMAB-TS`$(\mathcal{H}, \alpha, \beta)$: it denotes our proposed hierarchical multi-armed bandit with `Thompson sampling` outlined in Algorithm 1. $\mathcal{H}$ is the taxonomy defined by domain experts. α and β are hyper parameters.

3. `HMAB-LinUCB`$(\mathcal{H}, \lambda)$: it represents our proposed algorithm based on `LinUCB` presented in Algorithm 1. Similarly, $\mathcal{H}$ is the hierarchy depicting the dependencies among arms. And the parameter λ is given with the same use in `LinUCB`.

5.2 Dataset Description Experimental tickets are collected by IBM Tivoli Monitoring system [3]. This dataset covers from July 2016 to March 2017 with the size of $|\mathcal{D}| = 116,429$. Statistically, it contains 62 automations (e.g., NFS Automation, Process CPU Spike Automation, and Database Inactive Automation) recommended by the automation engine to fix the corresponding problems. The execution feedback including success, failure and escalation, indicates whether the problem has been resolved or needs to be escalated to human engineers. These collected feedback can be utilized to improve the accuracy of recommended results. Thereby, the problem of automation recommendation can be regarded as an instance of the contextual bandit problem. As we mentioned above, an arm is an automation, a pull is to recommend an automation for an incoming ticket, the context is the information vector of ticket's description, and the reward is the feedback on the result of the execution of recommended automation on the problem server. An automation hierarchy $\mathcal{H}$ shown in Figure 4 with three layers constructed by domain experts is introduced to present the dependencies among automations. Moreover, each record is stamped with the open time of the ticket.

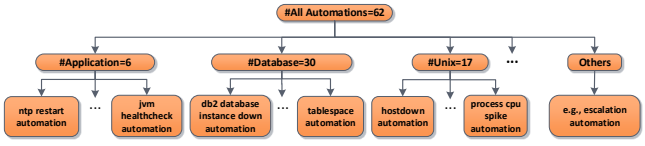

Figure 4: An automation hierarchy defined by domain experts.

We now discuss how to construct ticket features for the experiments. To reduce the noise of the data, the domain experts only selected the categorical attributes (e.g., ALERT_ KEY, CLIENT_ ID, SEVERITY and OSTYPE) with high representative information of tickets. These categorical information of a ticket is encoded as a binary vector [9]. In addition, we augmented a constant feature with value 1 for all vectors. Therefore, each ticket is represented as a binary feature vector $\mathbf{x}$ of dimension 1,182.

5.2.1 Evaluation Method We apply the *replayer* method to evaluate our proposed algorithms on the aforementioned dataset. The *replayer* method is first introduced in [9], which provides an unbiased offline evaluation via the historical logs. The main idea of *replayer* is to replay each user visit to the algorithm under evaluation. If the recommended item by the testing algorithm is identical to the one in the historical log, this visit is considered as an impression of this item to the user. The ratio between the number of user clicks and the number of impressions is referred to as Click-through rate (CTR). The work in [9] shows that the CTR estimated by the *replayer* method approaches the real CTR of the deployed online system. In this problem, a user is a ticket, and an item is an automation. A user visit means a ticket comes into IT automation services, and a user click indicates the ticket has been successfully solved by the recommended automation.

5.2.2 Relative Success Rate Optimization for Online Automation Recommendation In this section, we discuss the performance evaluation for each proposed algorithm on the dataset. The averaged reward (i.e., the overall success rate of the corresponding automations) is considered as the metric in the experiments. We define it as the total reward divided by the total number of times a given automation has been recommended (i.e., $\frac{1}{n}\sum_{t=1}^{n} r_t$). It is obvious that the higher the success rate, the better the performance of the algorithm. To avoid the leakage of business-sensitive information, the relative success rate is reported, which is the overall success rate of an algorithm divided by the overall success rate of random selection.

In contrast to the baseline algorithms outlined in Section 5.1, the corresponding HMABs configured with different parameter settings achieve much better performance on the dataset shown in Figure 5, Figure 6 and Figure 7, respectively. To be clarified, we set the parameter $\lambda > 1$ of LinUCB and HMAB-LinUCB in the experiments deliberately to reveal the merits of HMAB-LinUCB because their performance are almost equal with $\lambda < 1$. By observing the results, we find that HMAB-LinUCB has the best performance compared with other algorithms. Through these substantial experiments, we conclude that the proposed algorithms outperformed the strong baselines with the assumption that arms are independent.

5.2.3 A Comparative Case Study In order to better illustrate the merits of the proposed algorithms, we present a case study on the recommendation for an escalated ticket in IT automation services.

As mentioned above, the recommendation for an es-

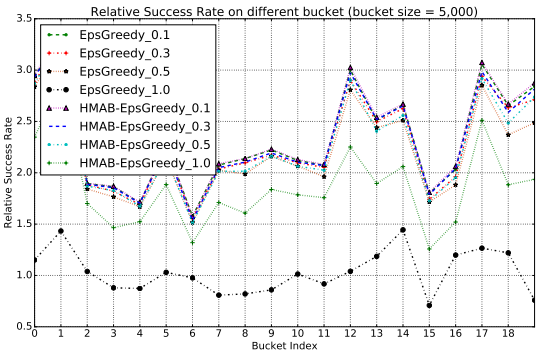

Figure 5: The Relative Success Rate of EpsGreedy and HMAB-EpsGreedy on the dataset is given along each time bucket with diverse parameter settings.

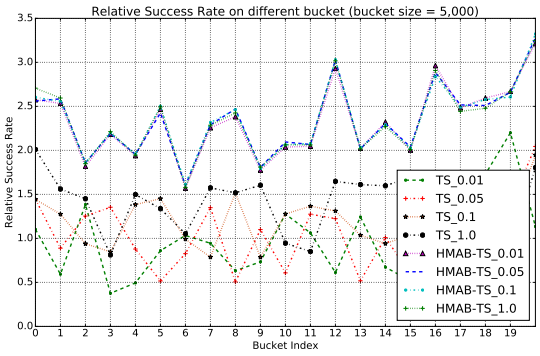

Figure 6: The Relative Success Rate of TS and HMAB-TS on the dataset is given along each time bucket with diverse parameter settings.

calated ticket can be regarded as a cold-start problem due to the lack of the corresponding automations. In other words, there is no historical records for resolving this ticket. Note that both our proposed HMABs and conventional MABs are able to deal with the cold-start problem by *exploration*. To compare their performance, we calculate the distribution of the recommended automations over different categories (e.g., database, unix, and application). Figure 9 presents an escalated ticket, which records a database problem. Such a problem has been repeatedly reported over time in the dataset. Since this ticket reports a database problem, intuitively the automations in the database category should have a high chance of being recommended. The category distributions of our proposed HMABs and conventional MABs are provided in Figure 8, as well as the baseline category distribution, which is the prior category distribution obtained from all the automations of the hierarchy. From Figure 8, we observe that 1) compared with TS, HMAB-TS explores more automations from the database category; and 2) in HMAB-TS the database category has the highest percentage among all th automation categories. This shows that our proposed HMABs can achieve better performance by making use of the predefined hierarchy.

To further illustrate the effectiveness of HMABs, we provide the detailed results of recommended automations for the escalated tickets. As shown in Figure 9, automations from the database category (e.g., database instance down automation, db2 database inactive automation) are frequently recommended according to the context of the ticket, which clearly indicate the issue is due to the inactive database. By checking the recommended results, domain experts figure out the **database instance down automation**, one of the top recommended automations, can successfully fix such a cold-start ticket problem, which clearly demonstrate the effectiveness of our proposed algorithms.

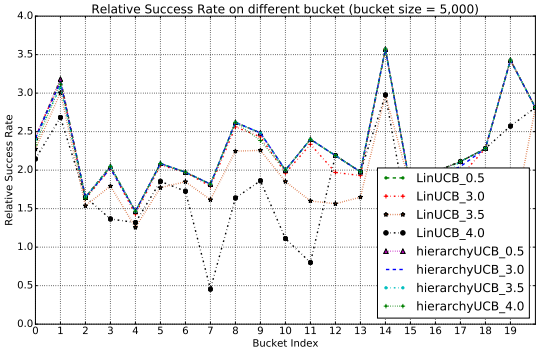

Figure 7: The Relative Success Rate of LinUCB and HMAB-LinUCB on the dataset is given along each time bucket with diverse parameter settings.

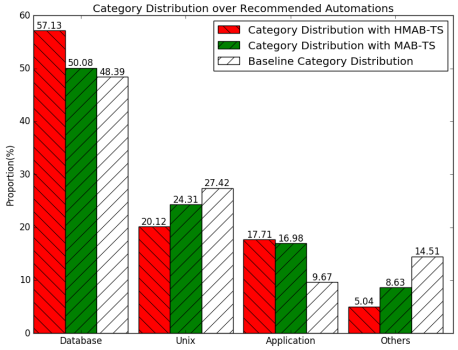

Figure 8: The comparison of category distribution on the recommended automations.

ALERT_KEY	ac2_dbinact_grzc_std	AUTOMATON NAME	Escalation Handler
TICKET SUMMARY	Database fin91dmo status is inactive.	TICKET RESOLUTION	The database is down. It has been restarted, hence closing the ticket.

RECOMMENDED CATEGORY	(%)	RECOMMENDED AUTOMATON
DATABASE	**57.13**	(1) **database** instance **down** automation; (2) db2 **database inactive** automation; (3) mysql **database offline** automation.
UNIX	20.12	(1) asm space check diskgroup **dbautomation;** (2) **hostdown** automation; (3) certification **expiration** automation.
APPLICATION	17.71	(1) ntp **restart** automation; (2) mq manager **down** automation.
OTHERS	5.04	(1) system load automation; (2) others.

Figure 9: The *exploration* by HMAB-TS of a cold-start ticket case.

6 Conclusion

This paper is the first work to model automation recommendation in IT automation services as a contextual multi-armed bandit problem, where the arms are organized in the form of a taxonomy. We first provide the mathematical formulation of this problem. Next we suggest algorithms for the solution to the problem. To show the effectiveness of our proposed solutions, empirical experiments are conducted on a real ticket dataset compared with conventional bandit algorithms, which assume that the arms are independent. In a case study of solving a cold-start problem, our proposed algorithms show a better performance due to usage of the hierarchy.

There are several directions for future research. We plan to use multi-document summarization technologies based on the problem inference to create a scripted resolution for a "new" ticket, and automatically create the automation in IT automaton services. Second, we're going to apply deep learning technology [8] for the ticket-feature representation to effectively process both categorical and non-categorical attributes of IT tickets.

7 Acknowledgements

The work was supported in part by the National Science Foundation under Grant Nos. IIS-1213026, CNS-1126619, and CNS-1461926.

References

[1] HP OpenView : Network and Systems Management Products. `http://www8.hp.com/us/en/software/enterprise-software.html`.

[2] IBM Enterprise IT Automation Services. `www.redbooks.ibm.com/redpapers/pdfs/redp5363.pdf`.

[3] IBM Tivoli : Integrated Service Management. `http://ibm.com/software/tivoli/`.

[4] S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *ICML*, pages 127–135, 2013.

[5] P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

[6] S. Chang, J. Zhou, P. Chubak, J. Hu, and T. S. Huang. A space alignment method for cold-start tv show recommendations. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 3373–3379. 2015.

[7] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*, pages 2249–2257, 2011.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] L. Li, W. Chu, J. Langford, T. Moon, and X. Wang. An unbiased offline evaluation of contextual bandit algorithms with generalized linear models. *JMLR*, 26:19–36, 2012.

[10] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW*, pages 661–670. ACM, 2010.

[11] S. Pandey, D. Agarwal, D. Chakrabarti, and V. Josifovski. Bandits for taxonomies: A model-based approach. In *SDM*, pages 216–227. SIAM, 2007.

[12] S. Pandey, D. Chakrabarti, and D. Agarwal. Multi-armed bandit problems with dependent arms. In *ICML*, pages 721–728. ACM, 2007.

[13] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR*, pages 253–260. ACM, 2002.

[14] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

[15] L. Tang, Y. Jiang, L. Li, C. Zeng, and T. Li. Personalized recommendation via parameter-free contextual bandits. In *SIGIR*, pages 323–332. ACM, 2015.

[16] L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik. Recommending resolutions for problems identified by monitoring. In *IFIP/IEEE IM*, pages 134–142. IEEE, 2013.

[17] L. Tang, T. Li, L. Shwartz, F. Pinel, and G. Y. Grabarnik. An integrated framework for optimizing automatic monitoring systems in large it infrastructures. In *SIGKDD*, pages 1249–1257. ACM, 2013.

[18] Q. Wang, C. Zeng, W. Zhou, T. Li, L. Shwartz, and G. Y. Grabarnik. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *arXiv preprint arXiv:1708.03058*, 2017.

[19] Q. Wang, W. Zhou, C. Zeng, T. Li, L. Shwartz, and G. Y. Grabarnik. Constructing the knowledge base for cognitive it service management. In *Services Computing (SCC), IEEE International Conference on*, pages 410–417. 2017.

[20] Y. Yue, S. A. Hong, and C. Guestrin. Hierarchical exploration for accelerating contextual bandits. *arXiv preprint arXiv:1206.6454*, 2012.

[21] C. Zeng and T. Li. Event pattern mining. *Event Mining: Algorithms and Applications*, pages 71–121, 2015.

[22] C. Zeng, T. Li, L. Shwartz, and G. Y. Grabarnik. Hierarchical multi-label classification over ticket data using contextual loss. In *2014 IEEE NOMS*, pages 1–8. IEEE, 2014.

[23] C. Zeng, L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik. Mining temporal lag from fluctuating events for correlation and root cause analysis. In *10th International Conference on Network and Service Management (CNSM)*, pages 19–27. IEEE, 2014.

[24] C. Zeng, L. Tang, W. Zhou, T. Li, L. Shwartz, G. Grabarnik, et al. An integrated framework for mining temporal logs from fluctuating events. *IEEE Transactions on Services Computing*, 2017.

[25] C. Zeng, Q. Wang, S. Mokhtari, and T. Li. Online context-aware recommendation with time varying multi-armed bandit. In *SIGKDD*, pages 2025–2034. 2016.

[26] C. Zeng, W. Zhou, T. Li, L. Shwartz, and G. Y. Grabarnik. Knowledge guided hierarchical multi-label classification over ticket data. *IEEE Transactions on Network and Service Management*, 2017.

[27] X. Zhao, W. Zhang, and J. Wang. Interactive collaborative filtering. In *CIKM*, pages 1411–1420. 2013.

[28] W. Zhou, W. Xue, R. Baral, Q. Wang, C. Zeng, T. Li, J. Xu, Z. Liu, L. Shwartz, and G. Ya Grabarnik. Star: A system for ticket analysis and resolution. In *SIGKDD*, pages 2181–2190. 2017.

[29] S. Wang, J. Tang, Y. Wang, and H. Liu. Exploring Implicit Hierarchical Structures for Recommender Systems. In *IJCAI*, pages 1813–1819. 2015.