A Novel Combinatorial Multi-Armed Bandit Game to Identify Online the Changing top-K Flows in Software-Defined Networks

Zhaogang Shu, Haoxian Feng, Tarik Taleb, and Zhifang Zhang

***Abstract*—Identifying the top-K flows that require much more bandwidth resources in a large-scale Software-Defined Network (SDN) is essential for many network management tasks, such as load balancing, anomaly detection, and traffic engineering. However, identifying such top-K flows is not trivial, not only because of the fluctuations in flow bandwidth requirements but also because of the combinatorial explosion of problem instance sizes. In this paper, we weaken the tradeoff between exploration and exploitation and innovatively define the online top-K flows identification problem as identifying the top-K arms in a Combinatorial Multi-Armed Bandit (CMAB) model. Then, we propose a general greedy selection mechanism with some identification strategies that focus on temporal variations in the rewards. Extensive simulation experiments based on real traffic data are conducted to evaluate the performance of different strategies. In addition, the results of numerical simulations demonstrate that our proposed greedy selection mechanism significantly outperforms existing counterparts on top-K arms identification.**

***Index Terms*—Software-Defined Network, Combinatorial Multi-Armed Bandit, top-K arms identification, and temporal variations in rewards.**

I. INTRODUCTION

Software-Defined Network (SDN) [1] is a novel network architecture that decouples the network control plane and data forwarding plane, providing operators with a flexible and low-cost new way to manage the network. In SDN, operators can deploy various management policies (e.g., data flows forwarding policy) to the controllers based on a global view, and then the data forwarding plane will carry out these policies. In addition, operators can adjust these policies dynamically by collecting and analyzing the statistics on data flows.

In a real network, the distribution of traffic flows is extremely skewed [2], that is, more than 80% of the traffic flows are less than 10KB in size, and most of the packets in the network are generated by the top 10% of large flows, which means that a small number of flows in a network have large bandwidth requirements. Identifying the top-K flows is critical in a wide variety of application scenarios, such as traffic rerouting [3], anomaly detection [4], network slicing [5], [6], [7], time-sensitive network [8], [9], [10], and caching of forwarding table entries [11].

Because in SDN, managers can effectively and timely collect statistical data of traffic flows [12], [13], some researchers put forward the concept of Knowledge-Defined Networking (KDN) [14] based on SDN and Network Analytics (NA) [15]. As shown in Fig. 1 (a), KDN sets up an artificial brain that gathers knowledge (e.g., how to identify the top-K flows online) about the network and then exploits that knowledge to design various management policies. However, online identification of top-K flows is still a challenging task due to many factors. On one hand, the solution space can be exponential to the network size due to combinatorial explosion. For example, top-K flows are identified among M flows, the solution space of this problem has C_M^K combinations. We simply set $K = 10$, $M = 100$ [16], and there are as many as 17 trillion combinations possible in the solution space. On the other hand, the time fluctuations of different types of traffic demand are not consistent [17], [18], which brings greater challenges to dynamically identifying top-K flows. Even if we reduce K to 1, the task of online identification is still very challenging, as shown in Fig. 1 (b), from t_0 to t_1, the bandwidth requirement of flow 1 is the highest, thus flow 1 is the target flow. However, the bandwidth requirements of the flows change over time, so the target flow also changes over time. From t_1 to t_3, flow 2 is the target flow, while after t_3, the target flow should be flow 3.

Although now some researchers have carried out related research on finding the top-K elephant flows [19], or the identification of top-K critical flows [16] in specific scenarios, their researches focused on identifying elephant flows within a time window, or ignore temporal changes in traffic flow bandwidth requirements. As shown in the above example, to tackle the vast solution space and various kinds of flows' bandwidth requirements uncertainties, a reasonable online identification mechanism for the top-K flows is desired.

In this paper, we investigate the problem of how to identify the changing top-K flows online [20] in SDN effectively, that is to continuously identify the top-K flows. We innovatively define the problem as a variant of the best arms identification task in stochastic Combinatorial Multi-Armed Bandit (CMAB) [21], namely, identify the top-K arms in CMAB. Unlike some existing works, our work considers the temporal changes of the reward distributions of all arms and ignores the tradeoff between exploration and exploitation, a detailed definition of our study and how it differs from existing work can be found in Section III, and Section II. B, respectively. Then, we propose a general greedy arms selection mechanism based on different identification strategies.

The main contributions of this paper are summaries as follows.

- We formulate the online top-K flows identification problem in SDN as a variant of the best arms identification

Zhaogang Shu, Haoxian Feng, and Zhifang Zhang are with the computer and information college, Fujian Agriculture and Forestry University, Fuzhou, China. Email:zgshu@fafu.edu.cn, fenghx@fafu.edu.cn, and zfzhang@fafu.edu.cn.

Tarik Taleb is with the Center of Wireless Communications, The University of Oulu, Finland. Email: tarik.taleb@oulu.fi.

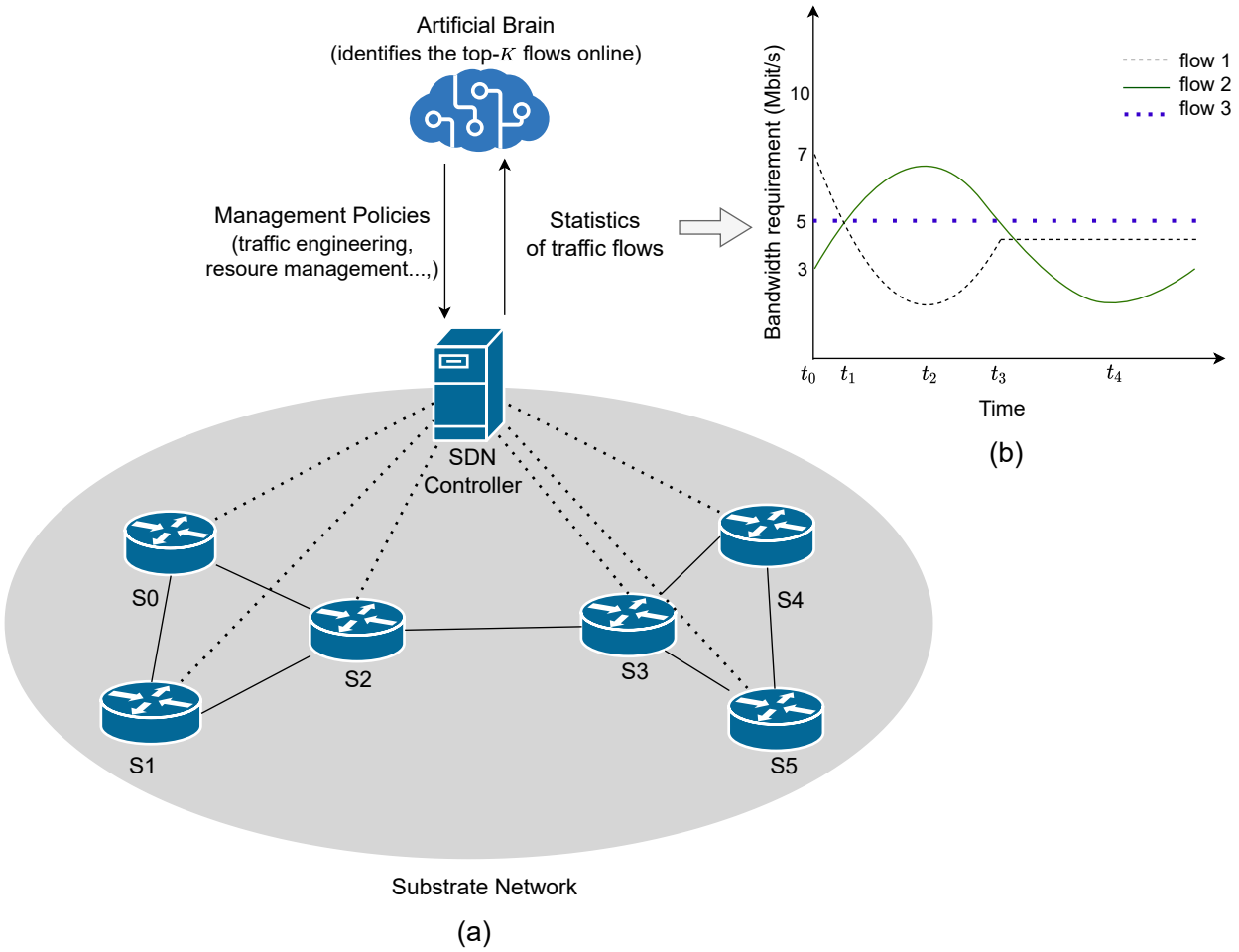

Fig. 1. The problem of identifying the top-K flows in SDN based on the KDN concept.

task in CMAB with the aim of maximizing the cumulative reward. Then, we propose a general greedy arms selection mechanism based on different identification strategies.

- We verify the performance of our proposed arms selection mechanism using traffic data from two real network topologies. Extensive simulation results demonstrate that our proposed mechanism far outperforms the benchmark algorithms in performance, and the identification result of our proposed mechanism can be close to 99% of the theoretically optimal solution.

The remainder of the paper is organized as follows: Section II reviews the related work. In Section III, we state the research problem in our work. Section IV introduces the details of our proposed general greedy arms selection mechanism. In Section V, we present the numerical results, and finally, we conclude the paper in Section VI.

II. RELATED WORK

In this section, we present the study of related work on finding top-K flows. In addition, because the Multi-Armed Bandit (MAB) and CMAB model can bring a lot of benefits to the research in the networking field [22], we also introduce the related work of best arms identification in MAB in this part, which is closely related to our research.

A. *Finding top-K flows*

Metwally A *et al.* proposed the first algorithm *Space-Saving* [23] that can guarantee both the correctness and the order of the top-K flows in the case of the data skew. Besides, their proposed algorithm only uses minimal space. To further reduce memory usage, unlike *Space-Saving*, Ben-Basat R *et al.* [24] used statically allocated memory rather than pointers. However, such strategies greatly overestimated the size of the flows, thereby degrading the accuracy performance. To solve this defect, Yang T *et al.* [19] proposed a probabilistic method to keep top-K flows in the bucket, moreover, they used multiple hash tables with different hash functions to address the problems of wrong elections and wrong estimation.

In [19], [23], [24], a flow is defined as a combination of certain packet header fields (e.g.,5-tuple), but in some other scenarios, like traffic engineering, a flow can be defined as a source-destination pair. Zhang J *et al.* [16] investigated the traffic flow rerouting problem in SDN. Interestingly, they found that compared to simply rerouting top-K flows, their

TABLE I
COMPARISON OF RELATED WORKS

Literature	Number of K	Reward distribution of arms in experiments	Consider tradeoff between exploration and exploitation
Audibert *et al.* [25]	1	Allows Bernoulli distributions	Y
Bubeck *et al.* [26]	Varies with the total number of arms	Allows Bernoulli distributions	Y
Zhang *et al.* [27]	No more than 5	Allows Gaussian distribution or exponential distribution	Y
Zhuang *et al.* [28]	No more than 3	Allows Bernoulli distributions	Y
Ours	Varies with the size of the network	Generated from real traffic data, and varies over time	N

reinforcement learning-based strategy can better identify the key flows in the network, thereby taking into account the overhead of rerouting and maximizing link utilization.

However, the mentioned related works only considered identifying the top-K flows within a specific time window and ignored the flows' bandwidth requirements fluctuations.

B. Best arms identification

The Best arms identification problem is a different viewpoint [25] in the MAB model, it allows the players to play the bandit within a given number of rounds, also called a budget. Then, players should determine an or a set of arms that with a higher expected reward. J.-Y. Audibert *et al.* [25] proposed a *Successive Rejects* policy that gradually rejects arms that seem suboptimal, but this policy is only used to identify the best arm. Similar to the *Successive Rejects* idea, Bubeck S *et al.* [26] proposed the *Successive Accepts and Rejects* (SAR) policy which can identify the top-K arms. Further, Zhang *et al* [27] proposed a quantile version of SAR (Q-SAR) which determines the optimal arms set through the quantile of the reward distributions rather than the mean. Zhuang *et al.* [28] considered a different scenario of top-K arms identification, they proposed two sampling strategies to identify the arms with extremely high or low expected rewards that are very different from others.

However, while in the above studies, the reward distributions of the arms were invisible to the players when designing the experiment, the rewards were generated by specific distributions. But in reality, the distribution of rewards may change over time. About the difference between our work and related works is summarized in Table I, and a detailed definition of the research problem in our work can be found in Section III.

III. PROBLEM STATEMENT

In this section, we describe the problem of identifying the changing top-K flows online in SDN. Note that, for ease of reference, the notations used in this paper are summarized in Table II.

A. Problem Definition

In SDN, at every sampling granularity T (e.g., 5 minutes, 15 minutes...), we need to identify the top 10% of flows that require more bandwidth resources based on all statistics

TABLE II
SUMMARY OF NOTATIONS

Notation	Definition
N	number of nodes in a network
n	number of total rounds
M	number of total arms
K	number of top-K arms
y_i	reward distribution of the i-th arm
$X_{i,t}$	random reward of the i-th arm in the t-th round
$\mathcal{X}_{i,t}$	the random reward information of the i-th arm up to the t-th round
$X_{i,t}^E$	expected reward of the i-th arm in the t-th round
S_t	the set of arms that the agent selects in the t-th round
S_t^*	the set of optimum arms in the t-th round
r	the cumulative regret
T	sampling granularity
α	weight
d	sliding window

collected for all network flows, in our current work, flows are defined as source-destination pairs [16]. This process is shown in Fig 2 that is similar in spirit to [29], [30], where $data_{T_n}$ represents the statistical data collected from t_{n-1} to t_n.

Obviously, the problem of identifying top-10% flows online in SDN can be transformed into a CMAB model. Assuming a total of N nodes in the network topology, in the CMAB model, there are a total of M arms, and after each sampling granularity T, the player needs to select K arms together rather than one by one, where $M = N * (N-1)$, $K = 10\%M$ [2], [16]. If we regard a flow as an arm, and consider the bandwidth requirement $X_{i,t}$ of the flow i from $t-1$ to t as the random reward generated by selecting arm i in round t. Then, our problem can be defined as Eq. (1), where n represents the total number of rounds.

$$maximize \sum_{t}^{n} \sum_{i}^{K} X_{i,t} \tag{1}$$

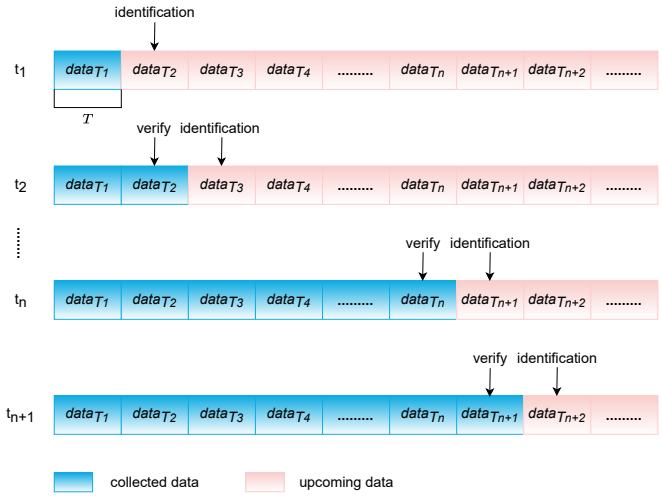

Fig. 2. Issues about identifications and validations over time.

Further, in the CMAB model, in the round t, the set S_t of the arms selected by the player is defined as a super-arm [21], and clearly, the number of super-arm combinations is C_M^K. If we define in the round t, the set of arms that can make the player's total reward [1] maximum be the optimal super-arm S_t^*, then our goal can be equivalent to Eq. (2), because the goal of maximizing the cumulative rewards through the game is equivalent to minimize the difference between the optimum super-arm and the super-arm that selected by the player [31].

$$minimize \quad r = \sum_{t}^{n} (\sum_{i \in S_t^*} X_{i,t} - \sum_{j \in S_t} X_{j,t}) \tag{2}$$

B. Discussion

To make the CMAB model more suitable for practical application scenarios, we add some new constraints and assumptions.

- **Time-varying rewards.** In a general CMAB or some other multi-armed bandit models, each arm is associated with a reward distribution, which can be stationary or non-stationary. Typically, the i-th arm's reward distribution can be represented as y_i and y_i is usually unknown to the players. However, in the network, the reward distribution for an arm is not only unknown, but also changes over time due to the sudden and time-varying nature of user needs [17], [18]. In this case, y_i can be redefined as $y_i(t)$, which means that the reward distribution of the i-th arm is a function of time.
- **Reward information sharing.** How to balance exploration and exploitation is a key topic in CMAB models. But in SDN, we do not need to consider this constraint. Due to the global view and flexibility of SDN, the network managers can easily collect global network flow statistics, such as traffic matrix. Thus, after each round, players can obtain the random reward for the arms that are not selected. In other words, after each round t, $\forall i \in M, X_{i,t}$ are available.

Although we do not need to consider the balance between exploration and exploitation, due to the time-varying distribution of rewards for each arm, how to use the historically collected reward sequence to estimate the expected reward $X_{i,t+1}^E$ for the i-th arm in the $t+1$ round is difficult [32]. In short, the fewer past observations an arm retains, the greater the stochastic error associated with an arm's estimate of the mean reward, while using more past observations at the same time increases the risk of these being biased.

IV. General Greedy Selection Mechanism

In this section, we first introduce the general greedy selection algorithm we designed, then introduce different identification strategies, and finally analyze the complexity of our designed algorithm.

A. General Greedy Selection Algorithm

- **Motivation.** Greed [20] is an important idea. Not only is it very simple and easy to implement, but it always excels at solving many practical problems, although it does not provide theoretical guarantees. The core of the greedy idea is that in each round t, the player will always perform the action A_t with the largest expected reward, where $A_t = argmax_{i \in M} X_{i,t}^E$. Note that, $X_{i,t}^E$ is calculated based on the i-th arm's historical reward data $\mathcal{X}_{i,t} = \{X_{i,1}, X_{i,2}, \cdots, X_{i,t-1}\}$, and its specific calculation method is given in Section. IV.B. In each round, we can directly select the top K arms with the largest expected reward to maximize the expected reward.
- **Implementation.** Generally, greed can be implemented in the following three steps. (1) For each arm $i \in M$, calculate its expected reward in the next round according to all the historical random reward data $\mathcal{X}_{i,t}$. (2) Rank all the arms according to their expected reward. (3) The top K arms are selected as the super-arm in the next round. Note that, as shown in Fig.2, our CMAB starts in the second round to avoid the problem of setting the initial reward of each arm. Algorithm 1 shows the pseudo-code of the general greedy selection algorithm.

In Algorithm 1, lines 8 to 9 indicate that after each round, we need to maintain some necessary information, such as the collected historical reward information and the number of rounds.

B. Identification strategies

As mentioned before, how the expected reward of arm i is calculated plays a decisive role in our online identification task. In this part, we propose several different strategies for calculating expected rewards, which are also defined as identification strategies.

[1] In our work, the reward of selecting a super-arm in round t is the sum of the rewards of the arms in S_t, and a more complex definition of the reward of S_t is beyond the scope of this article.

Algorithm 1 General Greedy Selection Algorithm

Input: number of total arms M, number of top-K arms K, all collected random reward $\mathcal{X}_{i,1}$, for all arm $i \in M$, total number of rounds n.

Output: cumulative regret r.

1: Initialization: let $r = 0$, $t = 1$;
2: **while** $n > 0$ **do**
3: **for** each arm $i \in M$ **do**
4: Using $\mathcal{X}_{i,t}$, calculate $X^E_{i,t+1}$ according to various equations in Section. IV.B;
5: **end for**
6: Sort all arms $i \in M$ by its $X^E_{i,t+1}$ in ascending order;
7: Let $S_{t+1} = \{i_0, i_1, \cdots, i_k\}$; // get the super-arm
8: Let $t = t + 1, n = n - 1$; // end of current round
9: For $i \in M$, let $\mathcal{X}_{i,t+1} = \mathcal{X}_{i,t} + X_{i,t+1}$;
10: $r = r + (\sum_{i \in S^*_{t+1}} X_{i,t+1} - \sum_{j \in S_{t+1}} X_{j,t+1})$;
11: **end while**
12: **return** r.

1) mean-greedy: Taking the average of all historical random rewards of arm i as its expected reward for the next round is a naive calculation method [33], [34], which we call mean-greedy. $X^E_{i,t+1}$ can be calculated through Eq. (3).

$$X^E_{i,t+1} = \frac{X_{i,1} + X_{i,2} + ... + X_{i,t}}{t} \tag{3}$$

2) weighted-greedy: The mean-greedy strategy is only suitable for stationary bandits model, and for some non-stationary bandits models, it makes sense to give more weight to the recent reward than the reward of a long time in the past [35]. We call this strategy weighted-greedy, and $X^E_{i,t+1}$ is calculated through Eq. (4), where $\alpha \in (0, 1]$ is the weight parameter.

$$\begin{aligned} X^E_{i,t+1} &= \alpha X_{i,t} + (1-\alpha)^2 \alpha X_{i,t-2} + \cdots \\ &\quad + (1-\alpha)^{t-1} \alpha X_{i,1} \\ &= \sum_{j=1}^{t} (1-\alpha)^{t-j} \alpha X_{i,j} \end{aligned} \tag{4}$$

3) sliding window-greedy: Similar to the weighted-greedy strategy, sliding window-greedy also focuses on recent random rewards information. However, unlike the weighted-greedy, sliding window-greedy only focuses on the recent d random reward values, and completely ignores random rewards information collected a long time ago. In this strategy, $X^E_{i,t+1}$ can be calculated by Eq. (5), where $d \geq 1$ is the sliding window.

$$X^E_{i,t+1} = \frac{\sum_{j=t-d}^{t} X_{i,j}}{d} \tag{5}$$

C. Complexity analysis

Next, we analyze the complexity of our proposed general greedy selection algorithm in a single round (lines 3 to 8 in Algorithm 1). First, for the first **for** loop, we have to calculate $X^E_{i,t+1}$. Note that, we do not have to use all the collected historical random rewards information. For $mean - greedy$, $X^E_{i,t+1} = \frac{(t-1) \cdot X^E_{i,t-1} + X_{i,t}}{t}$, and for $weighted - greedy$, $X^E_{i,t+1} = (1-\alpha) \cdot X^E_{i,t-1} + \alpha \cdot X_{i,t}$. Thus, for these two strategies, maintaining the expected reward for the last round is sufficient for each arm. However, for the $sliding\ window - greedy$, calculating $X^E_{i,t+1}$ relies on the recent d historical random reward data, for such cases, we need to maintain d data for each arm. Then, the temporal complexity for the sorting process is $\mathbf{O}(Mlog_2M)$. The above results are summarized in Table III.

TABLE III
COMPLEXITY ANALYSIS IN A SINGLE ROUND

Method	Temporal Complexity	Spatial Complexity
mean-greedy	$\mathbf{O}(M + Mlog_2M)$	$\mathbf{O}(M)$
weighted-greedy	$\mathbf{O}(M + Mlog_2M)$	$\mathbf{O}(M)$
sliding window-greedy	$\mathbf{O}(M + Mlog_2M)$	$\mathbf{O}(dM)$

V. NUMERICAL RESULTS

In this section, we evaluate the performance of our proposed general greedy selection algorithm with different identification strategies. We first introduce the dataset we used, then we describe the state-of-the-art top-K arms identification algorithms. Finally, we conduct an extensive simulation to compare our proposed algorithm with some other identification algorithms.

A. Dataset

In our work, we evaluate the performance of different algorithms using two real-world network topologies (Abilene and Geant, respectively)[2] that are widely used in the field of computing network research [16], [36], [37]. Table IV summarizes the properties of these topologies, such as the number of nodes, sampling granularity, and the number of top-K flows. For the Abilene network, we collected bandwidth requirements for 6 days (starting April 2, 2004) for a total of 1728 traffic matrices. For the Geant network, we choose a total of 672 traffic matrices over a week (starting May 5, 2005) as our dataset.

TABLE IV
INFORMATION OF TOPOLOGY

Topology	N	M	K	T	n
Abilene	12	132	13	5 minutes	1728
Geant	22	462	46	15 minutes	672

Note that, for some other identification algorithms (SAR, Q-SAR), We need to use the first 70% bandwidth demand data in the dataset to randomly generated reward data for arms determination. Please refer to Section.V.B for specific instructions.

[2]information available at: http://sndlib.zib.de/home.action

B. Introduction of Compared Algorithms

In our simulation, we compare our proposed greedy selection algorithm with the random selection algorithm, SAR [26] and Q-SAR [27]. The random selection algorithm is the weakest baseline. If the performance of an algorithm is weaker than random selection, we can consider this algorithm to be meaningless. SAR and Q-SAR are excellent related works that are used to identify the top-K best arms in MAB. Before presenting our simulation results, we first give a brief introduction to these compared algorithms.

- Random: The random selection algorithm completely ignores the historical reward information of the arms, and in each round t, it always randomly selects K different arms.
- SAR [26]: SAR focuses on identifying the top-K arms in a multi-armed bandit game with a fixed budget. It first divides the rounds into $M-1$ phases, then in each phase, SAR pulls each active arm at the same frequency. After each phase, SAR either accepts the arm with the highest empirical average or removes the arm with the lowest empirical average.
- Q-SAR [27]: Q-SAR is a revised version of SAR. Similar to SAR, Q-SAR also first divides the given budget into $M-1$ phase, and then pulls all the arms equally. Unlike SAR, which uses the empirical mean as the summary statistic, Q-SAR uses quantiles as the summary statistic, and then it decides whether to accept or reject an arm based on the best and worst empirical gaps instead of all empirical gaps.

The tasks of both SAR and Q-SAR are to output a set $\{i_1, i_2, \cdots, i_K\}$ corresponding to the set of arms with the K highest mean rewards after exploring the multi-arm gambling machine to a specified round (fixed budget). In the original work of related researchers, the reward for each arm obeyed a fixed reward distribution, and we can call this type of problem an offline problem, which means that the reward distribution does not change over time. In our evaluation, we use the nearly top 70% bandwidth requirement data of the dataset as training data (for Abilene, training data begins at 00:00 on April 4, 2004, and ends at 06:10 on April 6, 2004, and for Geant, training data begins at 00:00 on May 5, 2005, and ends at 21:30 on May 9, 2005) to randomly generate random rewards for the arms for SAR and Q-SAR algorithms. Then, we use the remaining 30% of the data as the testing data (for Abilene, there are a total of 201 bandwidth demand matrix data, and for Geant, the number is 501) to verify the performance of all algorithms.

C. Evaluation indicators

To more intuitively reflect the performance difference of the algorithms, in addition to the cumulative regret defined in Eq. (2), we also define an additional indicator $R_{S^*,S}(t)$ to measure the cumulative similarity between the super-arms selected by algorithms and the optimal super-arms, where $R_{S^*,S}(t)$ can be calculated by Eq. (6).

$$R_{S^*,S}(t) = \frac{\sum_t \sum_{i \in S_t} X_{i,t}}{\sum_t \sum_{i \in S_t^*} X_{i,t}} \tag{6}$$

D. Evaluation results

First, we need to evaluate the effectiveness of greed. We compare the performance of the naive mean-greedy strategy and random selection algorithm. As shown in Fig. 3, 4, in a period of time, the mean-greedy strategy can effectively approach the theoretical optimum super-arm at each round and outperform the random selection algorithm. However, its performance starts to degrade when the reward distributions of arms change.

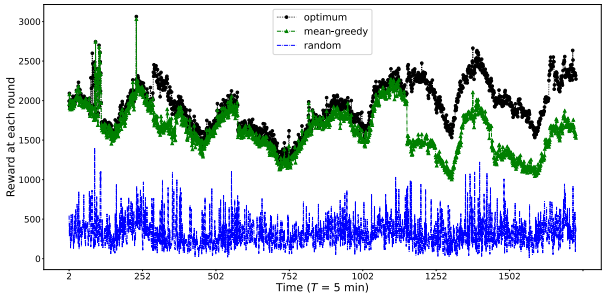

Fig. 3. Experiment 1 in Abilene network.

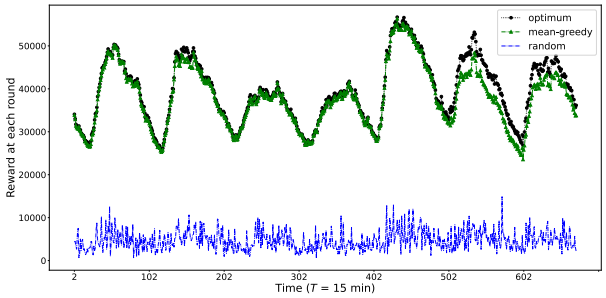

Fig. 4. Experiment 1 in Geant network.

Next, we compare the performance of different identification strategies in our general greedy selection algorithm. Initially, we set $w = 0.01, d = 20$. As shown in Fig. 5, 6, because both weighted-greedy and sliding window-greedy strategies focus much more on the recent historical random reward information, it can adapt to the temporal changes of the reward distributions better (after 1150 rounds in the Abilene, and after 500 rounds in the Geant). However, we also notice that sometimes their performance may drop drastically (e.g., in the 1445 round in Abilene). Thus, we should investigate how the weight and sliding window affect the performance.

We try to gradually increase w and reduce d to reduce the focus on historical reward data from a long time ago, and observe the change in performance. Interestingly, at least in our validation experiments, performance improves effectively as we pay more and more attention to recently received random

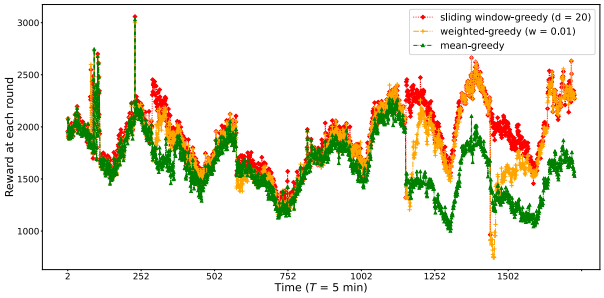

Fig. 5. Experiment 2 in Abilene network.

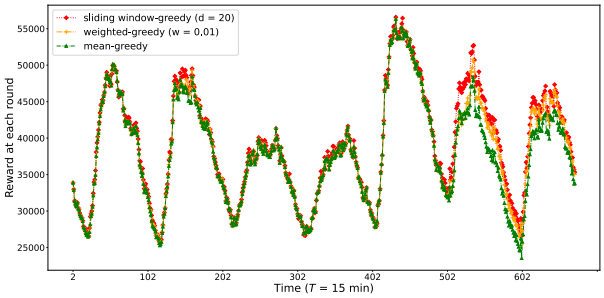

Fig. 6. Experiment 2 in Geant network.

rewards information, as shown in Fig 7, 8, 9, and 10. For the weight-greedy strategy, the performance seems to reach a maximum when w increases to a certain extent. Finally, we set $w = 0.9, d = 1$ for the following comparative experiments.

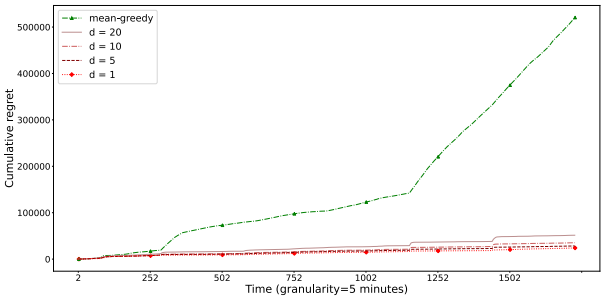

Fig. 7. Experiment to determine d in Abilene network.

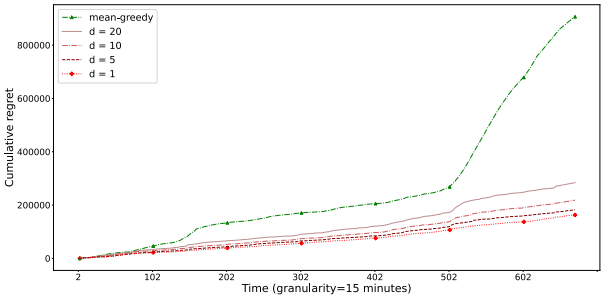

Fig. 8. Experiment to determine d in Geant network.

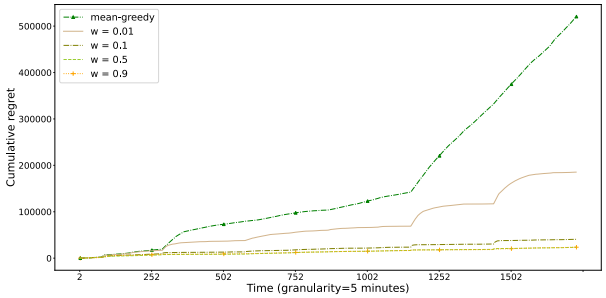

Fig. 9. Experiment to determine w in Abilene network.

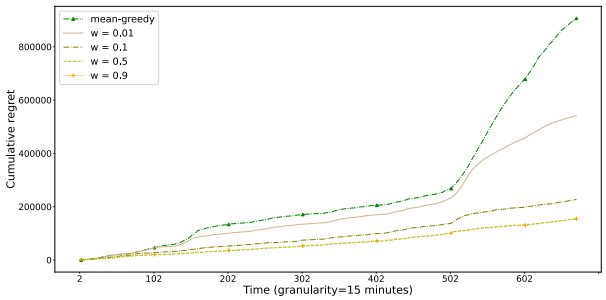

Fig. 10. Experiment to determine w in Geant network.

To be fair, for SAR and Q-SAR, we set the exploration budget to be large enough (in our work, we set the fixed budget to be 100 times the total number of arms) to ensure that they can adequately perceive each arm's historical random reward information. We then compare the performance differences of various algorithms on testing dataset. As shown in Fig 11, 12, the performances of both SAR and Q-SAR are not ideal, especially SAR, which has a very high cumulative regret. Although the performance of Q-SAR in Geant network is significantly better than that of SAR, its performance is still much weaker than the greedy selection algorithm we designed. We also note that the performance of sliding window-greedy far outperforms all other strategies, and its cumulative regret is almost 0 over time. We then use $R_{S^*,S}$ ($t = 501, 201$ in Abilene, Geant, respectively) to evaluate the comparison results. We find that the identification result of sliding window-greedy can reach 99% of the optimal solution, which is almost equivalent to the optimal solution. The above results are summarized in Table. V.

Note that, for SAR and Q-SAR, after exploration, they do not update the evaluation of each arm, which means that they do not bring extra cost. Thus, both SAR and Q-SAR assume that the reward distribution of each arm does not change over time. However, in many scenarios, the above assumption does not work. For our proposed scheme, to adapt to the temporal variations in the rewards, we update the evaluation of each arm after each round. It definitely brings extra cost, but as concluded in Table. III, generally, the extra cost is acceptable.

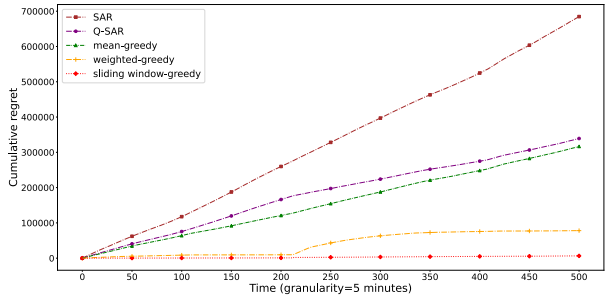

Fig. 11. A comparison of various algorithms in Abilene network

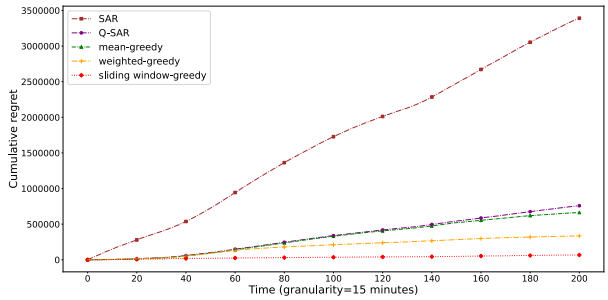

Fig. 12. A comparison of various algorithms in Geant network

VI. Conclusion and Future Work

In this paper, we investigated the online top-K flows identification problem in SDN and modeled it as a variant of the CMAB. With the help of SDN's global view, we appended the assumption of reward information sharing. We were pleasantly surprised to find that a simple greedy selection strategy works well for this task. At the same time, with the constraint of temporally changing reward distributions of arms, some traditional best arms identification algorithms lost their efficiency. In addition, there are some interesting researches that we can carry out in the future. For example, currently, we did not combine the top-K flow identification with some other network management tasks (e.g., traffic engineering, anomaly detection, SFC migration [38]...). In such cases, the reward of a super-arm should not be simply set up as a superposition of its sub-arms' rewards, in other words, complicated forms of the super-arms need to be considered based on specific scenarios. Second, we also need to adjust our CMAB model to some other multi-armed bandit models, like sleeping bandits, and contextual bandits, to adapt to the realistic scenarios.

TABLE V
Summary of Comparison Result

Topology	Method	$R_{S^*,S}$
Abilene	SAR	33.96%
	Q-SAR	67.33%
	mean-greedy	69.44%
	weighted-greedy	92.47%
	sliding window-greedy	***99.38%***
Geant	SAR	58.70%
	Q-SAR	90.75%
	mean-greedy	91.92%
	weighted-greedy	95.92%
	sliding window-greedy	***99.18%***

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research work was supported in part by the Fujian Province Natural Science Foundation of China under Agreement 2020J01574 and Industry-university-research Innovation Fund of China under Agreement 2021FNA05003. This research work was also conducted in ICTFICIAL OY and is partially supported by the European Union's Horizon Europe program for Research and Innovation through the aerOS project under Grant No. 101069732. It was also partially supported by the Academy of Finland 6Genesis project under Grant No. 318927 and the Academy of Finland IDEA-MILL project under Grant No. 352428

References

[1] McKeown N, Anderson T, Balakrishnan H, et al. OpenFlow: enabling innovation in campus networks[J]. ACM SIGCOMM computer communication review, 2008, 38(2): 69-74.
[2] Benson T, Akella A, Maltz D A. Network traffic characteristics of data centers in the wild[C]//Proceedings of the 10th ACM SIGCOMM conference on Internet measurement. 2010: 267-280.
[3] A. Sivaraman et al., "Programmable packet scheduling at line rate," in Proc. ACM SIGCOMM, 2016, pp. 44–57.
[4] Farris I, Taleb T, Khettab Y, et al. A survey on emerging SDN and NFV security mechanisms for IoT systems[J]. IEEE Communications Surveys & Tutorials, 2018, 21(1): 812-837.
[5] Afolabi I, Taleb T, Samdanis K, et al. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions[J]. IEEE Communications Surveys & Tutorials, 2018, 20(3): 2429-2453.
[6] Shu Z, Taleb T. A novel QoS framework for network slicing in 5G and beyond networks based on SDN and NFV[J]. IEEE Network, 2020, 34(3): 256-263.
[7] SHU Z, TALEB T, SONG J S. Resource Allocation Modeling for Fine-Granular Network Slicing in Beyond 5G Systems[J]. IEICE Transactions on Communications, 2021.
[8] Abbou A N, Taleb T, Song J S. A software-defined queuing framework for QoS provisioning in 5G and beyond mobile systems[J]. IEEE Network, 2021, 35(2): 168-173.
[9] Prados-Garzon J, Taleb T. Asynchronous time-sensitive networking for 5G backhauling[J]. IEEE Network, 2021, 35(2): 144-151.
[10] Abbou A N, Taleb T, Song J S. Towards SDN-based Deterministic Networking: Deterministic E2E Delay Case[C]//2021 IEEE Global Communications Conference (GLOBECOM). IEEE, 2021: 1-6.
[11] O. Rottenstreich and J. Tapolcai, "Optimal rule caching and lossy compression for longest prefix matching," IEEE/ACM Trans. Netw., vol. 25, no. 2, pp. 864–878, Apr. 2017.
[12] Chowdhury S R, Bari M F, Ahmed R, et al. Payless: A low cost network monitoring framework for software defined networks[C]//2014 IEEE Network Operations and Management Symposium (NOMS). IEEE, 2014: 1-9.
[13] Van Adrichem N L M, Doerr C, Kuipers F A. Opennetmon: Network monitoring in openflow software-defined networks[C]//2014 IEEE Network Operations and Management Symposium (NOMS). IEEE, 2014: 1-8.
[14] Mestres A, Rodriguez-Natal A, Carner J, et al. Knowledge-defined networking[J]. ACM SIGCOMM Computer Communication Review, 2017, 47(3): 2-10.

[15] Clemm, A., et al. "DNA: An SDN framework for distributed network analytics," Integrated Network Management (IM), IFIP/IEEE International Symposium on. IEEE, 2015.

[16] Zhang J, Ye M, Guo Z, et al. CFR-RL: Traffic engineering with reinforcement learning in SDN[J]. IEEE Journal on Selected Areas in Communications, 2020, 38(10): 2249-2259.

[17] D. Bega et al., "Optimising 5G infrastructure markets: The business of network slicing," in Proc. IEEE Conf. Comput. Commun. (INFOCOM), Atlanta, GA, USA, 2017, pp. 1–9.

[18] V . Sciancalepore et al., "Mobile traffic forecasting for maximizing 5G network slicing resource utilization," in Proc. IEEE Conf. Comput.Commun. (INFOCOM), Atlanta, GA, USA, 2017, pp. 1–9.

[19] Yang T, Zhang H, Li J, et al. HeavyKeeper: An Accurate Algorithm for Finding Top-k Elephant Flows[J]. IEEE/ACM Transactions on Networking, 2019, 27(5): 1845-1858.

[20] Lin T, Li J, Chen W. Stochastic online greedy learning with semi-bandit feedbacks[J]. Advances in Neural Information Processing Systems, 2015, 28.

[21] Chen W, Wang Y, Yuan Y. Combinatorial multi-armed bandit: General framework and applications[C]//International conference on machine learning. PMLR, 2013: 151-159.

[22] Maghsudi S, Hossain E. Multi-armed bandits with application to 5G small cells[J]. IEEE Wireless Communications, 2016, 23(3): 64-73.

[23] Metwally A, Agrawal D, Abbadi A E. Efficient computation of frequent and top-k elements in data streams[C]//International conference on database theory. Springer, Berlin, Heidelberg, 2005: 398-412.

[24] Ben-Basat R, Einziger G, Friedman R, et al. Heavy hitters in streams and sliding windows[C]//IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications. IEEE, 2016: 1-9.

[25] J.-Y. Audibert, S. Bubeck, and R. Munos. Best arm identification in multi-armed bandits. In Proceedings of the Twenty-Third Annual Conference on Learning Theory, pages 41–53, 2010.

[26] Bubeck S, Wang T, Viswanathan N. Multiple identifications in multi-armed bandits[C]//International Conference on Machine Learning. PMLR, 2013: 258-265.

[27] Zhang M, Ong C S. Quantile bandits for best arms identification[C]//International Conference on Machine Learning. PMLR, 2021: 12513-12523.

[28] Zhuang H, Wang C, Wang Y. Identifying outlier arms in multi-armed bandit[J]. Advances in Neural Information Processing Systems, 2017, 30.

[29] Ksentini A., Taleb T., and Benletaief K., "QoE-based Flow Admission Control in Small Cell Networks", in IEEE Trans. on Wireless Communications., Vol. 15, No. 4, Apr. 2016, pp. 2474 – 2483.

[30] Taleb T. and Ksentini A., "VECOS: A Vehicular Connection Steering Protocol," in IEEE TRANS. on Vehicular Technology, Vol. 64, No. 3, Mar. 2015. pp. 1171 – 1187.

[31] Allesiardo R, Féraud R, Maillard O A. The non-stationary stochastic multi-armed bandit problem[J]. International Journal of Data Science and Analytics, 2017, 3(4): 267-283.

[32] Besbes O, Gur Y, Zeevi A. Stochastic multi-armed-bandit problem with non-stationary rewards[J]. Advances in neural information processing systems, 2014, 27.

[33] Lattimore T, Szepesvári C. Bandit algorithms[M]. Cambridge University Press, 2020.

[34] Sun L, Hou J, Shu T. Spatial and temporal contextual multi-armed bandit handovers in ultra-dense mmWave cellular networks[J]. IEEE Transactions on Mobile Computing, 2020, 20(12): 3423-3438.

[35] Sutton R S, Barto A G. Reinforcement learning: An introduction[M]. MIT press, 2018.

[36] Eramo V, Lavacca F G, Catena T, et al. Application of a Long Short Term Memory neural predictor with asymmetric loss function for the resource allocation in NFV network architectures[J]. Computer Networks, 2021, 193: 108104.

[37] Eramo V, Catena T. Application of an Innovative Convolutional/LSTM Neural Network for Computing Resource Allocation in NFV Network Architectures[J]. IEEE Transactions on Network and Service Management, 2022.

[38] Feng H, Shu Z, Taleb T, et al. An Aggressive Migration Strategy for Service Function Chaining in the Core Cloud[J]. IEEE Transactions on Network and Service Management, 2022.