DISTRIBUTED ONLINE LEARNING OF THE SHORTEST PATH UNDER UNKNOWN RANDOM EDGE WEIGHTS

Pouya Tehrani, Qing Zhao

Department of Electrical and Computer Engineering
University of California, Davis, CA 95616,
{potehrani,qzhao}@ucdavis.edu

ABSTRACT

We consider the distributed shortest path problem in an undirected graph where the edge weights are random variables with unknown distributions. The objective is to design a distributed online learning algorithm to find the shortest path from a source node to a destination node where each node only knows its neighbors but not the entire network topology. The performance of the learning algorithms is measured by regret defined as the additional cost incurred over a time horizon of length T when compared to a centralized shortest path algorithm carried out under known edge weight distributions. We propose a distributed learning algorithm that achieves a regret logarithmic with the number of packets and polynomial with the network size. The same order with time and network size holds for the message complexity of the proposed algorithm. This result finds applications in cognitive radio and ad hoc networks under unknown and dynamic communication environments.

***Index Terms*—Shortest path problem, multi-armed bandit, distributed Bellman-Ford algorithm, cognitive radio.**

I. INTRODUCTION

Consider a network modeled by an undirected graph. The weight of each edge is given by i.i.d. realizations (over time) of a random variable with an unknown distribution. The objective is to find the shortest path from a source to a destination node with no prior knowledge on the network topology or the distributions of the edge weights. Through local information exchange with its neighbors, each node (including the source node) decides which neighbor to route the current packet to, aiming at minimizing the expected total cost of the resulting path by learning from past observations of its outgoing edges.

I-A. Distributed Online Learning of the Shortest Path

A centralized version of the above problem can be directly cast as a classic multi-armed bandit (MAB) problem. In the classic MAB [1]–[5], there are N independent arms. At each time, a player chooses one arm to play. An arm, when played, incurs i.i.d. random cost drawn from an unknown distribution. The performance of a sequential arm selection policy is measured by regret, defined as the total additional cost over a time horizon of length T when compared to an omniscient player who knows the cost model and always plays the best arm. Any policy with regret growing sublinearly with T achieves the same average performance as the omniscient player over an infinite horizon, and the slower the growth rate of the regret, the faster the convergence of the average performance to the optimal. It has been shown by Lai and Robbins that the minimum regret growth rate is logarithmic with time [1]. Since arms are assumed independent under the classic model, observations from one arm do not provide information about other arms. The optimal regret thus grows linearly with the number of arms.

It is not difficult to see that learning the shortest path can be considered as a classic MAB problem by treating each path from the source to the destination as an arm. Consequently, any MAB policy can be applies to achieve centralized learning of the shortest path. There are, however, two main challenges in such an approach. First, this approach yields poor performance with a regret growing linearly with the number of paths, thus, in the worst case, exponentially with the network size (in terms of the number of edges). Second, a distributed implementation under such an approach is difficult, if not impossible. In particular, all classic MAB policies rely on the number of times that each arm has been played to balance the tradeoff between exploration and exploitation. In a distributed setting where each node only interacts and observes its neighbors, an individual node does not have the global information on how many times a specific path from the source to the destination has been used for routing.

The key to the first challenge lies in the dependencies among paths that share common edges. These dependencies can be exploited in learning to achieve a regret order that is polynomial, rather than exponential, with the network size. To achieve a distributed balance between exploration and exploitation, our approach is to identify the least traversed

[0]This work was supported by the Army Research Office under Grant W911NF-12-1-0271 and by the Army Research Lab under Grant W911NF-09- D-0006.

edge in the network through local information exchange and ensure sufficient exploration of the currently least traversed edge at any given time. Based on these two key ideas, we propose a distributed learning algorithm referred to as Distributed Bellman-Ford with Learning (DBFL). The proposed algorithm achieves a regret logarithmic with time and polynomial with the network size. The same order with time and network size holds for the message complexity of the proposed algorithm.

This work applies to general distributed shortest path problems under random and unknown edge weights. In the context of communication networks, it applies to distributed shortest path routing in an unknown and dynamic environment. A specific application example is cognitive radio [6] where secondary users communicate by exploiting temporally and locally unused channels in the presence of a coexisting primary system. The availability of a link between two secondary users depends on the communication activities of nearby primary users which is in general random with a distribution unknown to the secondary users. As a consequence, the delay on each communication link can be thought of as a random variable with an unknown distribution. The shortest path here is thus the path with the smallest expected delay (intuitively, the path consisting of links that experience less primary traffic).

I-B. Related Work

Several policies exist for the classic MAB with independent arms [1]–[4]. These policies achieve the optimal logarithmic regret order for certain specific light-tailed cost/reward distributions. In particular, Auer et al. proposed in [3] the so-called UCB policy that achieves the logarithmic regret order for all distributions with bounded support. In [7], [8], the UCB policy was extended to all light-tailed distributions. In [5], a learning algorithm that achieves the optimal logarithmic regret order for all light-tailed distributions and sublinear regret order for heavy-tailed distributions was proposed.

For the weight minimization problem, most of the existing work focuses on deterministic and known edge weights [9]–[16]. Centralized learning of the shortest path has been considered in the literature [17]–[19]. In particular, in [17], it is assumed that the link state random variables have bounded support and each individual link is observable to a central controller. An online learning policy based on the UCB policy in [3] was developed that achieves $O(m^4 \log T)$ regret where m is the number of edges. [18], considers a more general observation model where only the total cost of a path, rather than the cost of each edge, is observed. An online learning algorithm was proposed that achieves $O(md^3 \log T)$ regret for any light-tailed distributions, where d is the dimension of the path set that is upper bounded by m. This algorithm also applies to heavy-tailed distributed with a regret that is linear in m and sublinear in T (with the sublinear rate depending on the highest order of the finite moment of the weight distributions). In [19], the problem was addressed under an adversarial bandit model in which the link costs are chosen by an adversary and are treated as arbitrary bounded deterministic quantities. An algorithm was proposed to achieve regret sublinear with time and polynomial with the network size. These centralized learning algorithms do not apply to the distributed shortest path problem considered in this paper. Opportunistic routing under unknown local broadcast models was considered in [20], [21]. While the shortest path problem aims to find a fixed path with the least expected cost and has applications beyond wireless networks, opportunistic routing exploits the local broadcast nature of the wireless links and aims to determine the next relay node based on specific realizations of the wireless links. It thus has a different scope from the problem addressed in this paper.

II. PROBLEM STATEMENT

Consider an undirected graph $G(V, E)$ where V is the set of nodes and E the set of edges. The source node $S \in V$ aims to communicate with the destination node $F \in V$ through the network (graph) G. For every edge $e \in E$ there is a corresponding random weight $W(e)$ with an unknown probability distribution. It is assumed that edge weights $W(e)$ have a bounded support in $[d_{\min}, d_{\max}]$ with $d_{\min} > 0$.

Each node $i \in V$ knows only its neighbors $\mathcal{N}(i)$, but not the entire network topology. When a node i routes a packet to one of its neighbors through an edge e, it will observe a realization of the random weight $w(e)$. For each packet t generated by the source S, it is routed to the destination F through distributed decisions made at each intermediate node. Let C^* be the expected total cost of the shortest path and $C_\pi(t)$ is the realized cost of the path selected for packet t by policy π. The regret of a policy π is given by

$$R_\pi(T) = TC^* - \mathbb{E}_\pi[\sum_{t=1}^{T} C_\pi(t)], \tag{1}$$

where $\mathbb{E}_\pi$ denotes expectation with respect to the random process induced by policy π.

III. DISTRIBUTED BELLMAN-FORD UNDER KNOWN WEIGHTS

In this section, we review the distributed Bellman-Ford algorithm for finding the shortest path under known deterministic edge weights. The basic idea of the distributed Bellman-Ford algorithm constitutes one component of the proposed DBFL. One new result we developed in this section is an upper bound on the converge time of the distributed Bellman-Ford (see Lemma 1). This result is needed in analyzing the regret performance of DBFL in Sec. IV-B.

When the edge weights are known deterministic values with $d_{\min} < \{d_{ij}\}_{ij \in E} < d_{\max}$, the distributed version of

the Bellman-Ford algorithm can be used [22] to find the shortest path. In the distributed asynchronous Bellman-Ford algorithm, the initial distance D_i from each node i to the destination, can be any arbitrary value. This would eliminated the need for initial synchronization at the beginning of the algorithm. The algorithm requires that nodes transmit new values of their estimated distance D_i to their neighbors from time to time. Then based on any new values of D_j received from its neighbors, node i updates D_i using

$$D_i \stackrel{\Delta}{=} \min_{j \in \mathcal{N}(i)} [d_{ij} + D_j]. \tag{2}$$

Lemma 1: Let $\mathcal{T}_{DBF}$ be the convergence time (the number of messages that needs to be exchanged for convergence) of the distributed Bellman-Ford. Then we have

$$\mathcal{T}_{DBF} \leq \frac{d_{\max}}{d_{\min}} |V|. \tag{3}$$

Proof: omitted due to space limit.

IV. DISTRIBUTED BELLMAN-FORD WITH LEARNING

IV-A. The DBFL Algorithm

In this section, we propose a distributed learning algorithm for the shortest path problem with unknown random weights. Referred to as DBFL (Distributed Bellman-Ford with Learning), this algorithm partitions the sequence of packets generated at the source into two types: the exploration packets and the exploitation packets. The exploration packets are routed through the currently least traversed edge in the network to ensure sufficient learning of all edges in the network. The exploitation packets are routed through the shortest path determined by the distributed Bellman-Ford using the current empirical mean of each edge weight. Specifically, let $\mathcal{E}(t)$ denote the index set of the exploration packets up to (and possibly including) the tth packet. Let $n_{ij}(t)$ denote the number of times that edge $i \leftrightarrow j$ has served as a relaying link for the packets in $\mathcal{E}(t)$. Define

$$l(t-1) \stackrel{\Delta}{=} \min_{\{i \leftrightarrow j\} \in E} n_{ij}(t-1) \tag{4}$$

as the least traversed link before the transmission of packet t. Consider packet t. If $t \notin \mathcal{E}(t)$, then packet t is routed opportunistically based on the shortest path determined by the distributed Bellman-Ford computed from the current empirical mean of each link weight. If $t \in \mathcal{E}(t)$, the source node adds an exploration header to packet t and routes the packet through the least traversed link $l(t-1)$(it will become clear later how the least traversed link is identified in a distributed manner). After reaching the least traversed link $l(t-1)$, the exploration header is removed by the end node of $l(t-1)$ and the packet is treated as a regular exploitation packet and is relayed to the destination. After the transmission of each exploration packet, a distributed algorithm, referred to as LTE as detailed in Fig. 2, is carried out to identify the new least traversed link and the path to reach this link from the source, which will be used to route the next exploration packet. At the same time, the distributed Bellman-Ford algorithm is carried out to find the shortest path using the sample mean as the current estimate of each edge weight. The resulting path will be used for all the upcoming exploitation packets until it is updated after the transmission of the next exploration packet. An illustration of the distributed learning algorithm is given in Fig. 1.

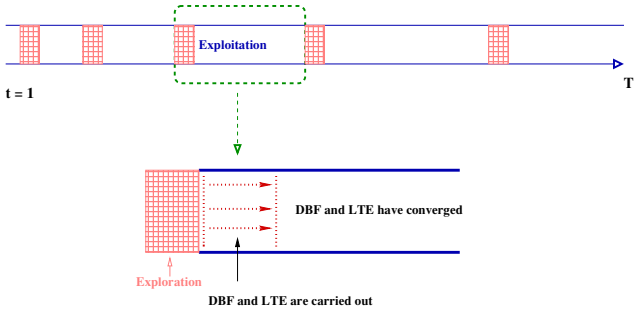

Fig. 1. The DBFL Policy

Note that we do not require a node knows whether an exploration packet has been delivered in order to start the distributed LTE and the distributed Bellman-Ford algorithms. The local information exchange for these algorithms will be initiated when a node sees a change in its current local information (e.g., an increase in n_{ij} after relaying an exploration packet). Furthermore, the exploitation packets are routed based on the current Bellman-Ford at each node without assuming its convergence. As a consequence, the algorithm is fully distributed; each individual node does not need to maintain a global count on the number of exploration packets that have been delivered (except the source) or to know whether its current local information reflects convergence.

We point out that in DBFL, learning is carried out only based on observations obtained during the transmissions of the exploration packets. As shown in Theorem 1, this is sufficient to achieve the optimal logarithmic regret order with a computational complexity that also grows only logarithmically, since the edge weight estimation and the distributed Bellman-Ford algorithm only need to be carried out after the transmission of each exploration packet. The algorithm can, of course, also learn from the observations obtained during the transmissions of the exploitation packets. This results in a better leading constant in the logarithmic regret at the price of higher computational complexity.

IV-B. Analysis of Regret and Message Complexity

An important design parameter in DBFL is the number of exploration packets in a sequence of T packets generated by the source. The cardinality of $\mathcal{E}(T)$ (denoted by $|\mathcal{E}(T)|$)

Finding the Least Traversed Edge (LTE)

- *Initialization:*

  Each node i ($i \in \Omega - \{N\}$) sets $n_{ij} = m_j = 0$ for each $j \in \mathcal{N}(i)$.

- *Local Updates and Exchanges at Node i:*

  When any of the following events occurs:
  - n_{ij} increases by one;
  - node i receives a new value of m_j from a neighbor j that is bigger than the current stored value of m_j.

  then
  - update m_i and the exploration neighbor o_i:

$$m_i \triangleq \min_{j \in \mathcal{N}(i)} n_{ij}$$
$$o_i \triangleq \arg\min_{j \in \mathcal{N}(i)} n_{ij} \quad (5)$$

  - send the new values (if changed) of n_{ij} or m_i or both to neighbors.

  When the following event occurs:
  - node i receives a new value of m_j from a neighbor j that is smaller than the current stored values of both m_j and m_i.

  then
  - update m_i with the new value of m_j; set $o_i = j$.
  - send the new value of m_i to neighbors.

Fig. 2. The distributed LTE algorithm for finding the least traversed edge.

balances the tradeoff between exploration and exploitation. It is not difficult to see that the regret order is lower bounded by $|\mathcal{E}(T)|$. Nevertheless, the sequence of exploration packets needs to be chosen sufficiently dense to ensure effective learning of the expected weights of links. The key issue here is to find the minimum cardinality of the exploration packets that ensures the additional cost caused by incorrectly identified node routing priorities during the transmissions of the exploitation packets having an order no larger than $|\mathcal{E}(T)|$. As shown in the theorem below, $|\mathcal{E}(T)|$ can be set to a logarithmic order with T, leading to the optimal logarithmic regret order of the learning algorithm DBFL.

Theorem 1: Let $L \leq |E|$ be an upper bound on the maximum number of edges that contributes to a path and c be an arbitrary nontrivial lower bound on the difference between the expected cost of the shortest path and the second shortest path. Set $G = (\frac{2L(d_{max}-d_{min})^2 d^2}{c^2} + \frac{1}{\log 2})|E|$. For each packet $t > 1$, if $|\mathcal{E}(t-1)| < G \log t$, then include t in $\mathcal{E}(t)$. Under this sequence of exploration packets, policy DBFL achieves regret $O(G|V| \log T)$ which is logarithmic with the number of packets and polynomial with the number of unknowns (worst case $O(|E|^4|V| \log T)$). Also the message complexity of the proposed policy is $O(|V|^2 G \log T)$.

Proof: Here we only sketch the proof due to space limit. We represent each path k as a vector $\mathbf{p}_k$ with $|E|$ entries consisting of 0's and 1's representing whether or not an edge is on the path. The vector space of all paths is embedded in a d-dimensional ($d \leq |E|$) subspace of $\mathcal{R}^{|E|}$. The cost of path k for packet t is thus given by the linear function

$$C_k(t) = < [W_1(t), W_2(t), \ldots, W_{|E|}(t)],\ \mathbf{p}_k > .$$

.

Regret of the policy DBFL is incurred during both exploration and exploitation. In the horizon of T packets, the policy spends $G \log T$ packets on exploration. Therefore regret incurred by the exploration packets is $O(G \log T)$.

In the exploitation, the regret consists of two parts: regret when the Bellman-Ford recursion has not converged, and regret when a path different than the actual shortest path is chosen after convergence of the Bellman-Ford. From Lemma 1, we know that the distributed Bellman-Ford converges in $O(|V|)$ and since this recursion should be renewed after each exploration packet is sent through the network, the regret incurred in this case is $O(G|V| \log T)$.

It only remains to consider the regret incurred by the exploitation packets when the Bellman-Ford recursion has converged. Based on the structure of the exploration, after N exploration packets for each edge $e \in E$ we have $n_e \geq \lfloor \frac{N}{|E|} \rfloor$. Therefore $n_e(t) \geq \lfloor \frac{G \log t}{|E|} \rfloor$. Based on Chernoff bound and taking advantage of the network structure (using barycentric spanner basis of the network which is represented by a vector space in $\mathcal{R}^{|E|}$ as explained above) we can show that the number of times that the non-optimal path is chosen in the exploitation phase when the recursion has converged up to time t is bounded above by $2d \log T$. Consequently the regret for the exploitation after Bellman-Ford convergence is $O(d \log T)$.

Regarding the message complexity, there are total of $G \log T$ exploration phases and at the end of each one the distributed Bellman-Ford and the LTE algorithms are run. Those algorithms both have $O(|V|^2)$ message complexities and there are total $G \log T$ of them. Therefore the message complexity of the proposed policy is $O(|V|^2 G \log T)$. ■

V. CONCLUSION

In this paper a shortest path problem is considered where the edge weights are random variables with unknown distributions. A dynamic distributed learning algorithm (DBFL) is developed that achieves regret order logarithmic with the time horizon length and polynomial with the network size.

VI. REFERENCES

[1] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules", in *Advances in Applied Mathematics*, vol. 6, no. 1, pp. 422, 1985.

[2] R. Agrawal, "Sample mean based index policies with $O(\log(n))$ regret for the multi-armed bandit problem, in *Adv. Appl. Probab.*, vol. 27, no. 4, pp. 10541078, Dec. 1995.

[3] P. Auer, N. Cesa-Bianchi, and P. Fisher, "Finite-time Analysis of the Multiarmed Bandit Problem," in *Machine Learning*, vol. 47, no. 2,3, pp. 235-256, May, 2002.

[4] T. Lai, "Adaptive Treatment Allocation and The Multi-Armed Bandit Problem", Ann. Statist., vol 15, pp. 1091-1114, 1987.

[5] K. Liu and Q. Zhao, "Multi-Armed Bandit Problems with Heavy Tail Reward Distributions", *in Proc. of Allerton Conference on Communications, Control, and Computing*, September, 2011.

[6] Q. Zhao and B. Sadler, "A Survey of Dynamic Spectrum Access", *IEEE Signal Processing Magazine*, vol. 24, pp. 7989, May 2007.

[7] R. Kleinberg, "Online Decision Problems with Large Strategy Sets", Ph.D. Thesis, MIT, 2005.

[8] K. Liu and Q. Zhao, "Extended UCB1 for Light-Tailed Reward Distributions", available at http://arxiv.org/abs/1112.1768.

[9] C. Oliveira, P. Pardalos, "A Survey of Combinatorial Optimization Problems in Multicast Routing", *Computers and Operations Research*, vol. 32, no. 8, pp. 1953-1981, August 2005.

[10] B. Das and V. Bharghavan, "Routing in Ad Hoc Networks Using Minimum Connected Dominating Sets", *in Proc. of IEEE International Conference on Communications (ICC)*, June, 1997.

[11] B. Awerbuch, "Optimal Distributed Algorithms for Minimum Weight Spanning Tree, Counting, Leader Election, and Related Problems", *in Proc. of the 19th Annual ACM Symposium on Theory of Computing*, 1987.

[12] J. Yu, J. Chong, "A Survey of Clustering Schemes for Mobile Ad Hoc Networks", *IEEE Communications Surveys and Tutorials*, vol. 7, no. 1, pp. 32-48, 2005.

[13] X. Cheng, B. Narahari, R. Simha, M. Cheng, D. Liu, "Strong Minimum Energy Topology in Wireless Sensor Networks: NP-Completeness and Heuristics", *IEEE Transactions on Mobile Computing*, vol. 2, no. 3, pp. 248-256, 2003.

[14] D. Li, X. Jia, H. Liu, "Energy Efficient Broadcast Routing in Static Ad Hoc Wireless Networks", *IEEE Transactions on Mobile Computing*, vol. 3, no. 2, pp. 144-151, 2004.

[15] P. Wan, K. M. Alzoubi, and O. Frieder, "Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks", *Mobile Networks and Applications*, vol. 9, No. 2, pp. 141-149, 2004.

[16] F. Dai, J. Wu, "An Extended Localized Algorithm for Connected Dominating Set Formation in Ad Hoc Wireless Networks", *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 10, pp. 908-920, 2004.

[17] Y. Gai, B. Krishnamachari, and R. Jain, "Combinatorial Network Optimization with Unknown Variables: Multi-Armed Bandits with Linear Rewards and Individual Observations", *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, 2012.

[18] K. Liu and Q. Zhao, "Adaptive Shortest-Path Routing under Unknown and Stochastically Varying Link States", *in Proc. of the 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, May, 2012.

[19] B. Awerbuch, R. Kleinberg, "Online Linear Optimization and Adaptive Routing", *Journal of Computer and System Sciences*, pp. 97-114, 2008.

[20] A. Bhorkar, M. Naghshvar, T. Javidi and B. Rao, "An Adaptive Opportunistic Routing Scheme for Wireless Ad-hoc Networks", *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, January 2012.

[21] A. Bhorkar and T. Javidi, " No Regret Routing for ad-hoc wireless networks", *in Proceedings of Asilomar Conference on Signals, Systems, and Computers*, November 2010.

[22] Dimitri P. Bertsekas and Robert G. Gallager, "Data Networks", *(2nd edition) Prentice Hall*, 1992, ISBN 0132009161.