The Impact of Network State AoI on Throughput in a Wireless SDN

Clement Kam*, Sastry Kompella*, and Anthony Ephremides†

Email: clement.kam@nrl.navy.mil, sastry.kompella@nrl.navy.mil, etony@ece.umd.edu
*Information Technology Division, Naval Research Laboratory, Washington, DC
†Electrical and Computer Engineering Department, University of Maryland, College Park, MD

***Abstract*—This work studies the role of Age of Information (AoI) in the network state updating process for wireless software defined networks (SDN). The SDN routers must routinely update their knowledge of the network state, which is used as a basis for making routing and scheduling decisions. However, the network updates require communication resources, so there is a tradeoff between the frequency of updates and maximum network throughput. We assume the network state is Markovian and no new observations are received in between updates, so the AoI of the network state information impacts the ability of the network to optimize its performance. We formulate the problem as a finite-horizon Partially Observable Markov Decision Process (POMDP) for each period. For a symmetric fading model of the network, we derive the limiting performance and an upper bound. To generate policies for a range of fixed time horizons, we use Monte Carlo planning-based POMDP solvers. Simulation of these policies show that there is a finite optimal update period that maximizes network throughput. In addition, we study non-uniform update intervals, which can yield even higher throughput if the interval is chosen based on the state observed. We conclude that AoI itself is not sufficient to characterize performance, but what matters is the AoI for the specific network state information.**

I. Introduction

Age of Information (AoI) [1] has garnered intense interest in the network and information theory research communities over the past decade. As a metric for the freshness of information updates, AoI enables a more purposeful approach to the networking and communication problem than being agnostic to the ultimate purpose of the communication (e.g., focusing on capacity or delay). While many works aim to minimize the AoI, doing so may not necessarily best serve the ultimate objective, especially when resources for updating AoI are shared with other data flows. In this work, we look at the relationship between AoI and the higher objective it is serving. Specifically, we focus on a wireless software-defined network (SDN), in which the SDN routers routinely participate in an updating process to acquire the state of the network. Since the updating process occurs over a bandwidth-limited wireless channel, the updates are delayed, and the network state information ages in between updates.

Software-defined networking (SDN) [2] is a paradigm that decouples the control and data plane, simplifying network management and promoting centralized control. While more prevalent in wired networks, recent studies have shown promise in applying the SDN paradigm to wireless ad hoc networks, yielding improved throughput and network lifetime compared to distributed routing protocols [3]. There have been some works related to the network update problem. The problem of where to place the SDN controller(s) to minimize delay or other cost has been studied in both wired [4] and wireless [5] settings. The network updating problem itself has mostly been studied in wired networks to reduce delay caused by forwarding loops, black holes, and other problems [6]. In [7], a linear programming framework was proposed for solving network updating problems with custom requirements like update speed or congestion control. Batch updates have been proposed to reduce resource consumption in the Internet of Things [8]. None of these works present a model of the impact of the frequency of network updates on the network performance.

There have been some works that study Age of Information in relation to other network objectives. These include age penalty functions [9], which characterizes the level of dissatisfaction with aged information, and other AoI-related metrics, such as effective AoI [10], or Age of Incorrect Information (AoII) [11], [12]. Some works consider competing objectives [13], [14] or mixtures of objectives [15], while others look at the impact of age on caching performance [16] or the performance of supervised learning tasks [17].

We recently studied a related problem in the context of cognitive radio networks [18], [19], in which the primary user's transmit dynamics are Markovian, so the age of the information sensed by the secondary impacts its probability of successful transmission. By formulating the throughput and collision probabilities in terms of the distribution of AoI and each type of information (primary being idle or transmitting), we were able to formulate a linear program that maximizes the throughput subject to a collision constraint. The problem here is related, but has much more complex dynamics and a larger state and action space. However, we similarly conclude that AoI must be associated with the information it is tracking to optimize for the ultimate objective.

In this work, we study the problem of total throughput maximization in a wireless SDN modeled as a Markov Decision Process. This includes determining the best timing of network updates, which relate to the AoI of the network state information available to the SDN controller. Our contributions are as follows:

ISBN 978-3-903176-49-2 © 2022 IFIP

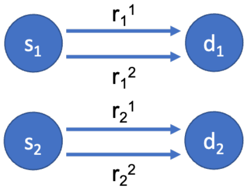

Fig. 1. Two user multilink model. Each link can be ON or OFF, which depends on the previous state and the transmission state of all the users.

- We formulate the finite horizon partially observable Markov decision process (POMDP) to generate policies for each interval between network updates, which is done by applying Monte Carlo-planning solutions for POMDPs.
- We analyze the throughput performance as the transmission interval approaches infinity, and we also solve the performance of the full observability case to serve as an upper bound.
- We propose a number of heuristic policies, some based on inspection of the POMDP-generated policies.
- To further understand the role of AoI, we study non-uniform updating policies, including a mixed period policy based on greedily choosing the period for each observed state that improves the throughput.

Simulations of the various policies demonstrate that the POMDP-based policies perform much better than the heuristics. In addition, the mixed period policy is shown to outperform the periodic policies, and its throughput vs. AoI performance is shown to be superior, illustrating the additional gains possible by pairing AoI with its associated information.

II. System Model

Although our approach can be generalized to larger, multi-hop networks, we focus on a two-user single hop setting in this work shown in Fig. 1. We consider a slotted two-user system, in which each user has multiple links with different link capacity. This single hop model can approximate a longer multi-hop route and the impact of the other user transmitting at the same time, but the following approach can also be generalized to a network with more users, destinations, paths, and states. In this work, the link availability is either ON or OFF, where a user can only transmit over it if it is ON, and no data can be transmitted when it is OFF. We consider 2 links per user, and the maximum quantity of data that can be transmitted in a slot for user u over link l is denoted as $r_{u,l}$. The evolution of the state of all links in the network can depend on the current state as well as the transmission links utilized by all of the users, and thus we model the network dynamics and throughput as a Markov Decision Process.

The true network state (status of all links) is unobservable during transmissions, and the SDN controller makes transmission decisions based on information received during routine network state updates. We assume these updates provide the SDN controller with complete knowledge, but the information immediately starts to age when the network begins making transmissions. We define the AoI of the network state as the the number of slots that have passed since the last network update (during which the network has been transmitting). Based on the network state AoI and the Markov model, the network state probability function can be calculated in each slot. We also assume that an update uses up a single slot, but this can be generalized for arbitrary update time durations. Under this model, there is a tradeoff between having fresh network state information (low AoI) and leaving enough resources (transmission slots) for data flows, which is controlled by the timing and frequency of network updates.

We divide the timeline into the intervals occurring between consecutive network state updates, such that we have perfect knowledge of the network state in the slot just before the start of the interval and no observations for the remainder of the interval. For each time interval of length $T-1$ between network state updates, we model the problem of throughput maximization for the network in Fig. 1 as a finite-horizon Partially Observable Markov Decision Process (POMDP) as follows:

- The state space $\mathcal{S}$ is the space of all possible combinations of link states, which are given as $s = (s_1^1, s_1^2, s_1^1, s_2^2)$, where $s_u^l \in \{0, 1\}$, 0 denotes the link is OFF and 1 denotes the link is ON.
- The action space $\mathcal{A}$ is given by the space of all possible decisions made by the users $a = (a_1, a_2)$, where $a_u \in \{0, 1, 2\}$, 0 denotes no transmission, 1 and 2 denotes which link the user transmits over. In every Tth slot, there is no transmission.
- The state transition probability is given by $P(s'|s, a)$.
- The reward is the total data successfully transmitted by both users, $r(s, a) = \sum_{u=1}^{2} \sum_{l=1}^{2} s_{u,l} \mathbb{1}(a_u = l) p_r(u, l|a) r_{u,l}$, where $0 \leq p_r(u, l|a) \leq 1$ is the percent reduction in rate for user u over link l given the action a.
- During the interval after the network update, there are no observations. To model the problem for a software-based solver, we can model the observations as random and independent of the actual state (we choose a uniformly randomly chosen state).

The time horizon for these potential transmissions in the interval is $T-1$ slots, and the AoI of the network state is simply the time index t. The average throughput after N intervals is given by

$$\overline{R} = \frac{1}{T_N} \sum_{t=1}^{T_N} r(t)$$

where T_N is the last slot of the nth interval, and $r(t)$ is the reward at time t, where $r(t) = 0$ when a network update is taking place before the interval (assuming the first update occurs at $t = 0$, this happens $N-1$ times). We first consider a periodic updating pattern, such that the all the T_n are equal to T ($\overline{R} = \frac{1}{NT} \sum_{t=1}^{NT-1} r(t)$). Because observations are lacking in this problem, we can track the probability of being in a particular state, called the belief state, to derive the optimal policy for a given POMDP. The belief state is expressed as $\Lambda(n) = [\lambda_{1,1}(n), \lambda_{1,2}(n), \lambda_{2,1}(n), \lambda_{2,2}(n)]$, where $\lambda_s(n)$ is the probability of being in state s at time n.

To solve the finite-horizon POMDP exactly, we would formulate the problem as a Markov Decision Process (MDP) on the belief states, in which a belief update is performed at each stage of the POMDP. Instead of the states in the original problem, the states of this so-called belief MDP are the belief states, which are fully observable. The only difference is now the state space is continuous, since the belief state is a distribution over the original states. It has been shown that for finite-horizon POMDPs, the value function is piecewise-linear and convex, and can be represented as a finite set of vectors. So while it is possible to apply value iteration, the complexity can be exponential in the number of actions and observations. As an alternative, we apply Monte Carlo planning-based methods [20], which use simulation to learn the best actions and also enable us to find solutions for large state spaces, as is the case for larger networks.

III. Network State Transition Model

While the POMDP approach can be applied to more general network state transition models, we focus on a model that we describe as a slow/fast fading interference network. We consider the ON-OFF state of each link to be Markovian and independent of the other link states, but the rate at which the state changes depends on the other user's interference. Specifically, the rate of switching between OFF and ON is higher when the other user is transmitting: the probability of user u's link l switching from ON to OFF or OFF to ON is $p_{lo;u,l}$ when the other user is not transmitting, and $p_{hi;u,l}$ when the other user is transmitting, where $p_{lo;u,l} \leq p_{hi;u,l}$. Also, the rate of switching is higher if the transmission rate is higher: if $r_{u,l_1} \geq r_{u,l_2}$ then, $p_{lo;u,l_1} \geq p_{lo;u,l_2}$ and $p_{hi;u,l_1} \geq p_{hi;u,l_2}$. The transition probabilities are given by the following expression:

$$\begin{aligned} &P((s'_{1,1}, s'_{1,2}, s'_{2,1}, s'_{2,2})|(s_{1,1}, s_{1,2}, s_{2,1}, s_{2,2}), (a_1, a_2)) \\ &= \prod_{u=1}^{2} \prod_{l=1}^{2} \Big[\mathbb{1}(s'_{u,l} = s_{u,l})(\mathbb{1}(a_{\overline{u}} = 0)\overline{p}_{lo;u,l} \\ &\quad + \mathbb{1}(a_{\overline{u}} \neq 0)\overline{p}_{hi;u,l}) + \mathbb{1}(s'_{u,l} \neq s_{u,l})(\mathbb{1}(a_{\overline{u}} = 0)p_{lo;u,l} \\ &\quad + \mathbb{1}(a_{\overline{u}} \neq 0)p_{hi;u,l}) \Big] \end{aligned}$$

where $\overline{p}_{lo;u,l} \triangleq 1 - p_{lo;u,l}$ and $\overline{p}_{hi;u,l} \triangleq 1 - p_{hi;u,l}$. To overcome the curse of dimensionality, we apply a Monte Carlo-planning based POMDP solver to generate the policy for each time horizon $T-1$ and each observed state. For each $T-1$ and initial state s_0, we save the sequence of simulated actions and use it as the policy when the observation is s_0. We simulate two approaches provided by the pomdp-py software, POUCT and POMCP. POUCT conducts a Monte Carlo tree search using an Upper Confidence Bound approach to determine the action at each step that on average yields the highest reward over the horizon, while POMCP modifies the POUCT approach by applying a particle filter to perform the belief state updates, making it more suitable for larger state spaces where an exact belief state update becomes infeasible. As an example, we simulate the policies for Model 1 in Table I[1], and the results are shown in Fig. 2. We observe that POUCT does slightly better than POMCP since it computes the belief state exactly instead of estimating it using a particle filter. We also consider a policy which only uses the policy for the longest time horizon T (for $T = 20$ called "POUCT20" and "POMCP20") and applies it to the lower periods T. Compared to the original POUCT/POMCP, we see that the performance suffers, which suggests that the strategy at each time step is dependent on the final time horizon. Since the transmission causes interference that affects the transition probabilities, the idle/transmit strategy changes depending on how long of a time horizon is available. It is possible that there are multiple policies that perform equally well for $T = 20$, and some of those would also perform well for smaller T, but the simulation-based solver is unlikely to choose those consistently. An example of the policy for different time horizons is shown in Fig. 3.

TABLE I
Network Model Parameters

	(u, l)	(1,1)	(1,2)	(2,1)	(2,2)
Model 1 (No Interference)	$p_{lo;u,l}$	0.05	0.05	0.05	0.05
	$p_{hi;u,l}$	0.1	0.1	0.1	0.1
	$p_r(u, l\|a = 0)$	0.95	0.95	0.95	0.95
	$p_r(u, l\|a \neq 0)$	0.95	0.95	0.95	0.95
	$r_{u,l}$	0.95	0.95	0.95	0.95
Model 2 (Collision)	$p_{lo;u,l}$	0.05	0.05	0.05	0.05
	$p_{hi;u,l}$	0.1	0.1	0.1	0.1
	$p_r(u, l\|a = 0)$	0.95	0.95	0.95	0.95
	$p_r(u, l\|a \neq 0)$	0	0	0	0
	$r_{u,l}$	0.95	0.95	0.95	0.95
Model 3 (User 1 has better links)	$p_{lo;u,l}$	0.025	0.05	0.05	0.075
	$p_{hi;u,l}$	0.05	0.1	0.1	0.15
	$p_r(u, l\|a = 0)$	0.91	0.95	0.95	0.99
	$p_r(u, l\|a \neq 0)$	0.41	0.45	0.45	0.49
	$r_{u,l}$	0.91	0.95	0.95	0.99
Model 4 (User 1 is indep. of User 2; no benefit from User 2 Tx)	$p_{lo;u,l}$	0.01	0.01	0.01	0.01
	$p_{hi;u,l}$	0.01	0.01	0.01	0.01
	$p_r(u, l\|a = 0)$	0.95	0.95	0	0
	$p_r(u, l\|a \neq 0)$	0	0	0	0
	$r_{u,l}$	0.95	0.95	0	0

In addition to the periodic policy, we also consider a mixed period policy based on the average reward for the period T for each state. This is formulated as choosing the set of periods $T^*(s)$ that maximizes the total expected reward (denoted $r_{T^*}(s)$) divided by the expected period. We use a greedy-like heuristic for choosing this set of periods that works as follows:

"Mixed Period" Policy Generation:

1) Start with $T^*(s) = 2$ for each s. The average reward is given by $\overline{r}(T^*) = \sum_s E[r_{T^*}(s)] / \sum_s T^*(s)$, where $r_T(s)$ is the total reward over the time horizon $T - 1$ after observing s.
2) Choose the minimum $T(s)$ that is greater than $T^*(s)$ that increases the average reward.

[1] The models in Table I will be used in later simulations.

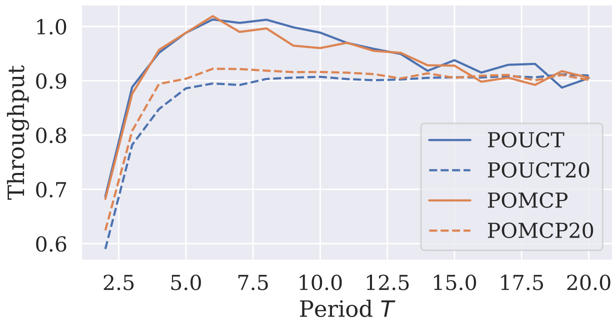

Fig. 2. POUCT vs. POMCP: "POUCT/POMCP" is derived for all T, "POUCT20/POMCP20" applies the policy derived for $T = 20$ to all periods.

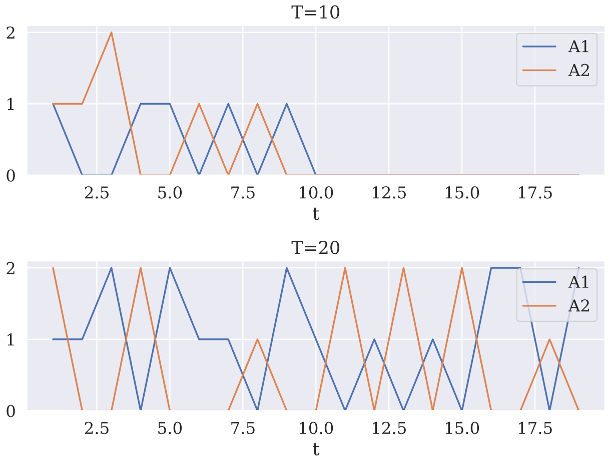

Fig. 3. Policies for $T = 8, 20$; State=$(0, 0, 0, 0)$; Model 2.

3) Repeat 2) until it is no longer possible to increase the average reward.

The average reward model assumes the probability of observing each state during an update is equal, but this is not the case, and incorporating the true probabilities would improve the performance. More importantly, the main insight we get from our results is that the performance of a fixed period (horizon) can be improved upon by dynamically choosing a different horizon length depending on which network state is observed. This leads us to conclude that AoI alone is not enough, but the AoI has a different impact depending on the corresponding information (i.e., the network state), and that should be accounted for.

A. No Observations Case (State-Independent, $T \to \infty$)

To obtain insight into the expected performance as $T \to \infty$, we study the case where the network never makes any observations and the network state is unknown. Under this condition, we propose a state-independent, stationary randomized transmission policy, which is denoted as $\pi(a_1, a_2)$. For this policy, we would like to find the expected throughput, given by the following:

$$E_\pi[r(s,a)] = \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}} r(s,a)\mu^\pi(s;a)$$

where $\mu^\pi(s;a)$ is the occupation measure, or the long-run average amount of time spent on the state-action pair (s,a), under the policy $\pi(a_1, a_2)$. Substituting for $r(s,a)$ as defined in Sec. II, we obtain the following:

$$\begin{aligned}
E_\pi[r(s,a)] &= \sum_{s\in\mathcal{S}}\sum_{a\in\mathcal{A}}\Big(\sum_{u=1}^{2}\sum_{l=1}^{2}\mathbb{1}(s_{u,l}=1)\mathbb{1}(a_u=l) \\
&\qquad \times p_r(u,l|a)r_{u,l}\Big)\mu^\pi(s;a) \\
&= \sum_{a\in\mathcal{A}}\pi(a)\sum_{s\in\mathcal{S}}\Big(\sum_{u=1}^{2}\sum_{l=1}^{2}\mathbb{1}(s_{u,l}=1)\mathbb{1}(a_u=l) \\
&\qquad \times p_r(u,l|a)r_{u,l}\Big)\mu^\pi(s) \qquad (1) \\
&= \sum_{a\in\mathcal{A}}\pi(a)\Big(\sum_{u=1}^{2}\sum_{l=1}^{2}\mathbb{1}(a_u=l)p_r(u,l|a)r_{u,l}\Big) \\
&\qquad \times \sum_{s\in\mathcal{S},s_{u,l}=1}\mu^\pi(s). \qquad (2)
\end{aligned}$$

where $\mu^\pi(s)$ is the long-run average amount of time spent in state s, and (1) is due to the independence of states and actions. For the term in (2), we have aggregated the states for which $s_{u,l}$ is 1. The probability of $s_{u.l}$ transitioning from 0 to 1 is the same as the probability of it transitioning from 1 to 0 under this model. Therefore, $\sum_{s\in\mathcal{S},s_{u,l}=1}\mu^\pi(s) = \sum_{s\in\mathcal{S},s_{u,l}=0}\mu^\pi(s) = 1/2$. Finally, we have

$$E_\pi[r(s,a)] = \frac{1}{2}\sum_{a\in\mathcal{A}}\pi(a)\Big(\sum_{u=1}^{2}\sum_{l=1}^{2}\mathbb{1}(a_u=l)p_r(u,l|a)r_{u,l}\Big)$$

which is linear in the policy $\pi(a)$. The only other constraint we have is $\sum_a \pi(a) = 1$, so the optimal policy is to choose the single action a that yields the maximum $\sum_{u=1}^{2}\sum_{l=1}^{2}\mathbb{1}(a'_u = l)p_r(u,l|a')r_{u,l}$. Equivalently, we compute $p_r(u,l|a')r_{u,l}$ for each (u,l), choose the link l that maximizes that quantity for each user u, and take the sum of the quantities for both users.

B. Full Observability Case (Throughput Upper Bound)

To obtain an upper bound for the throughput, we consider the case of a fully observable Markov Decision Process, in which the network state is known in each slot. We solve the average-cost infinite horizon problem by applying relative value iteration (RVI) [21], [22]. We denote the mapping $Th(s) = \max_{a\in\mathcal{A}}\left[r(s,a) + \sum_{s'\in\mathcal{S}} P(s'|s,a)h(s')\right]$, $s \in \mathcal{S}$. This is the dynamic programming mapping that will be applied in our RVI algorithm. Using state 0 as the arbitrary reference state, the kth iteration of the RVI for maximizing throughput is given by the following:

$$Th^k(0) = \max_{a\in\mathcal{A}}\left[r(0,a) + \sum_{s'\in\mathcal{S}} P(s'|0,a)h^k(s')\right]$$

$$h^{k+1}(s) = \max_{a\in\mathcal{A}}\left[r(s,a) + \sum_{s'\in\mathcal{S}} P(s'|s,a)h^k(s')\right] - Th^k(0)$$

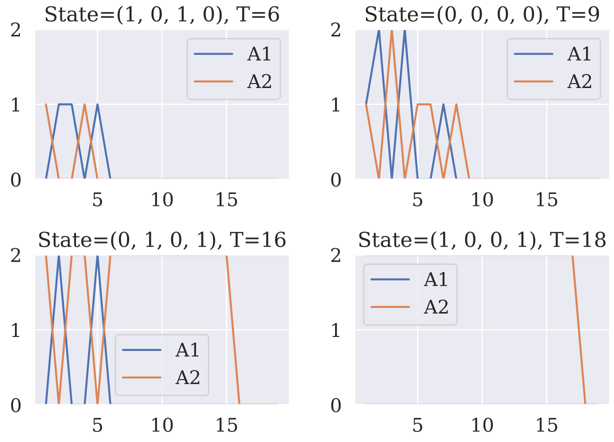

Fig. 4. Selected POUCT-Derived Policies, Model 2.

If the algorithm converges, the optimal cost is given by $Th^k(0)$. In some cases where the algorithm does not converge, a modified RVI [23] can be used.

C. Heuristics

For comparison, we propose a number of heuristics that can be more efficiently generated than the Monte Carlo-derived policies. We first propose a "Greedy" heuristic, in which the network state is observed, and each user transmits for the rest of the period over the ON link would yield the greatest reward, and is idle if neither link is ON.

To develop other heuristics, we visualize some selected strategies that came out of the POMDP solver (specifically, POUCT) in Fig. 4 for Model 2 in Table I. Although there is no obvious intuitive pattern to the policies, there are portions that have an alternating characteristic, in which the users take turns transmitting and being idle. We thus propose a "Greedy Alternating" heuristic, in which the two users take turns transmitting on the active link that yields the greatest reward. We also propose another simple heuristic that we call "Tx Alt," in which both users always transmit, but each user alternates transmitting over its two links starting with the link that yields the greatest reward. We will see that this results in no throughput when the channel success is based on a collision model.

IV. Simulations

A. Generating Policies

To generate policies for this problem, we use the pomdp-py [24] Python framework for modeling and solving POMDPs, which includes the Monte-Carlo planning approaches POUCT and POMCP [20]. Since the POUCT method was coded to handle infinite horizon problems, we modified it to handle finite horizon problems, which fits our system model with the timeline divided up into intervals. Then we derive a transmission policy for a given time horizon $T-1$ by running the solver for each observed network state s for $T-1$ time steps. The sequence of actions taken is stored for later use as the strategy when observing that particular network state s.

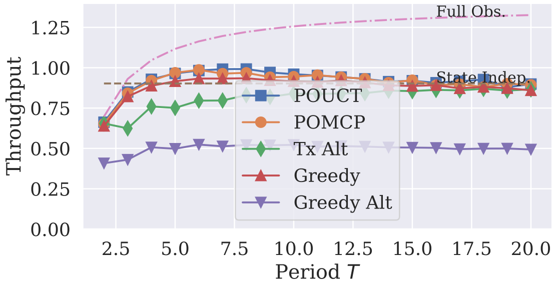

Fig. 5. Throughput vs. Period, Model 1 (no interference).

B. Periodic Network Updates

We first focus on the case where network updates occur periodically, and we simulate for different period lengths T. The policies simulated are the POUCT- and POMCP-derived policies, as well as the heuristics from Sec. III-C. We run the simulation for 1000 time slots and average results of 100 runs. The network model parameters we simulate are given in Table I.

The first set of simulations is for Model 1, which is a case of symmetric users, symmetric links, and no interference. The simulation results are plotted in Fig. 5. We also plot the theoretical performance for the full observability case (labeled "Full Obs."), derived using RVI. To compare to a network update period of T, we weighed the optimal throughput from RVI by $(T-1)/T$ to account for the update slots where there would be no transmission. We observe that the full observability performance upper bounds the performance of all policies.

POUCT, POMCP, and "Greedy" all do quite well, while the alternating ("Alt") policies are not as good. There is clearly a finite optimal period T^*, and it turns out that the throughput is maximized at $T^* = 6$ for POUCT, which corresponds to an average AoI of 2.5. This confirms our intuition that for wireless SDNs, the update period should be carefully chosen to be neither too small nor too large to optimize performance. For up to about $T = 15$, those policies perform better than the state-independent policy of Sec. III-A (dashed line). To see if the performance approaches the state-independent policy, we generated and simulated the $T = 40$ policy, using the same policy for smaller periods, and the results are shown in Fig. 6. We know from Fig. 2 that applying the policy for the larger period to smaller periods experiences performance degradation, but since it was derived for $T = 40$, it performs the same as the POUCT for that data point. We see that at $T = 40$, the performance is approaching the state-independent performance, as speculated.

Next we simulated for Model 2, which is a case of symmetric users, symmetric links, and a collision model. The results are plotted in Fig. 7. We observe that "Tx Alt" does not result in any throughput due to the collision model. Again, the POUCT and POMCP approaches perform well, and the throughput is maximized at $T^* = 7$. The pure greedy policy does not do well, but the "Greedy Alt." performs satisfactorily

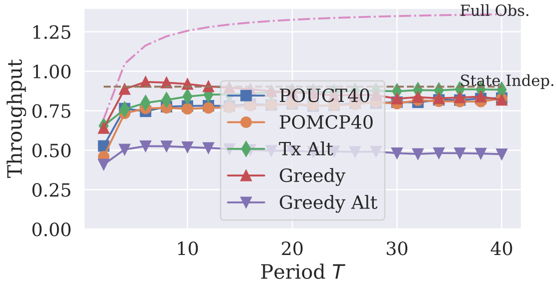

Fig. 6. Throughput vs. Period, Model 1 (no interference), using $T = 40$ policy.

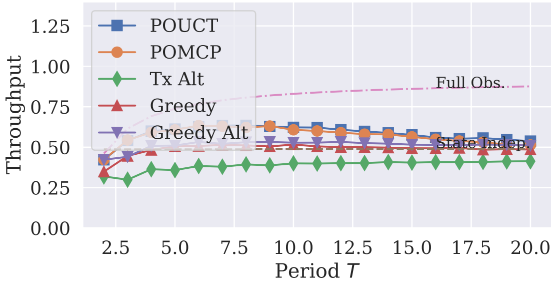

Fig. 8. Throughput vs. Period, Model 3 (asymmetric).

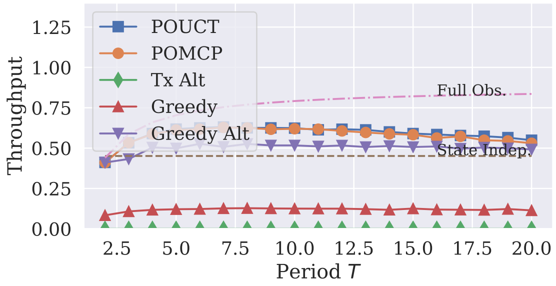

Fig. 7. Throughput vs. Period, Model 2 (collision).

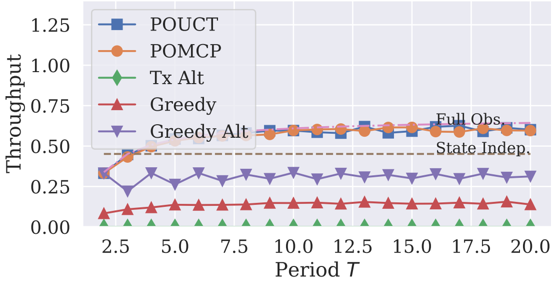

Fig. 9. Throughput vs. Period, Model 4 (no User 2).

since the users are taking turns.

Model 3 is a more general case, for which the users and links are asymmetric, and the results are plotted in Fig. 8. The transition probabilities, interference-free rate reduction, and reward are chosen to have an average that is equal to that of Models 1 and 2. The performance of POUCT and POMCP compares very well with that of Model 2 even with significantly less reduction in rate when there is interference.

As a sanity check, we simulate Model 4, in which case the states do not change often, there is a collision model, User 1's state transition is not affected by User 2's transmission, and User 2 gets no reward and thus has no reason to transmit. Because of these characteristics of the model, there is also limited benefit from being able to observe the state at all times. The results are plotted in Fig. 9, and indeed, we notice that POUCT and POMCP perform very similar to the upper bound, "Full Obs." The heuristics do not perform well because they typically involve User 2 transmitting, which reduces the throughput due to the collisions.

In all models, the POMCP compares well with POUCT, which provides support for its use in both small networks, like the one studied here, and networks with larger state spaces, where it is more needed.

C. Non-Uniform Updates

To further understand the role of AoI when the main objective is throughput maximization, we consider non-uniform updating policies, namely the observation-dependent "Mixed Period" approach of Sec. III, and a random updating case, in which the probability of updating in a slot is $1/T$. To get $E[r_{T^*}(s)]$ in the greedy-like "Mixed Period" policy generation, we take the average reward for each time horizon T and state s for the POUCT-derived policy, using the results from simulations in Sec. IV-B. Once we generate the "Mixed Period" policy, we simulate it and the other policies for Models 1–4 and plot the results in Figs. 10–13. We observe that for Models 2 through 4 the "Mixed Period" policy achieves a throughput that is on average higher than under any other periodic policy, since it optimizes the period length depending on the network state that is observed. Thus, if we were focused on AoI to optimize throughput, the necessary insight is that AoI is dependent on the information content itself. For the "Mixed Period" policy, the average AoI, though not a sufficient statistic, turns out to be 3.14, 3.91, 3.62, and 7.32 for Models 1, 2, 3, and 4, respectively. Note also that for Model 4, the "Mixed Period" performs similar to the full observation when $T = 20$, which makes sense because we know User 2 should not transmit and the state changes very infrequently.

For these simulations, we also plot the throughput performance as a function of AoI in Figs. 14–17. We first notice the throughput is low for AoI $= 1$. This is because the "Mixed Period" policy sets the period to 2 when the state is $(0, 0, 0, 0)$, so when AoI $= 1$, the links are typically OFF. We observe that for other T for Models 1 and 4, the "Mixed Period" approach has noticeably better throughput for AoI performance than the other policies since in Model 1 both can transmit interference-free, and in Model 2 user 1 can transmit all the time since there is no reward for user 2 transmitting. The throughput is only slightly better for Models 2 and 3 since they must

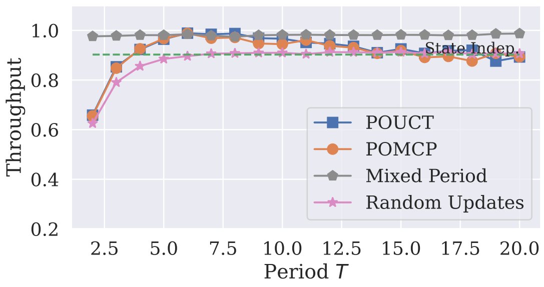

Fig. 10. Throughput vs. Period, Model 1, periodic (POUCT/POMCP) vs non-periodic.

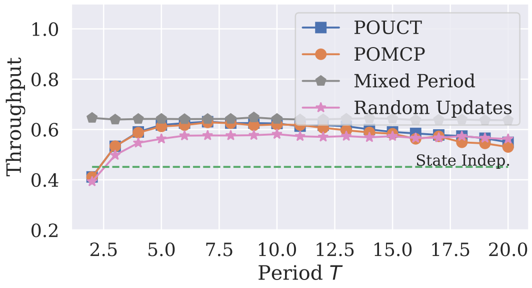

Fig. 11. Throughput vs. Period, Model 2, periodic (POUCT/POMCP) vs non-periodic.

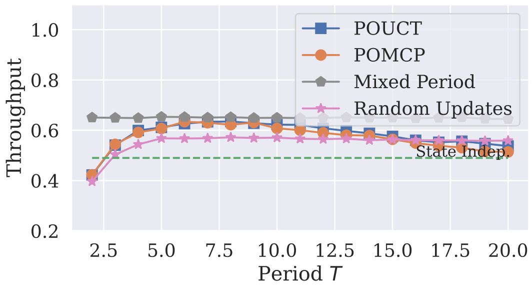

Fig. 12. Throughput vs. Period, Model 3, periodic (POUCT/POMCP) vs non-periodic.

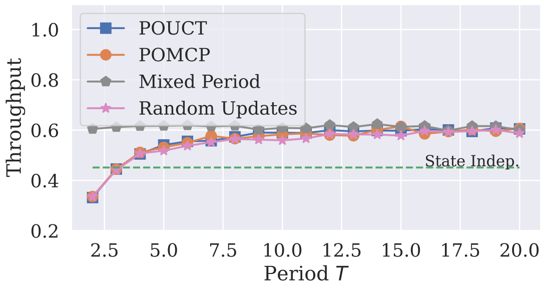

Fig. 13. Throughput vs. Period, Model 4, periodic (POUCT/POMCP) vs non-periodic.

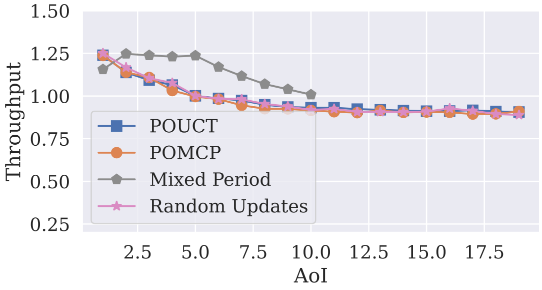

Fig. 14. Throughput vs. AoI, Model 1.

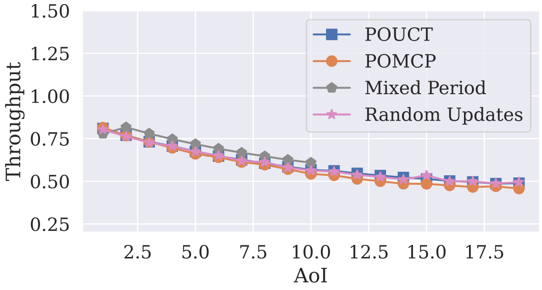

Fig. 15. Throughput vs. AoI, Model 2.

alternate transmit and idle to limit interference. In all four models, the maximum AoI for “Mixed Period” does not go up to 20 because it takes advantage of the shorter periods where the information has not aged as much.[2] Of course, there is a tradeoff in that reducing AoI uses up resources (time slots), so updating cannot be done too frequently. The “Random Update” throughput for AoI closely follows the curve for the “POUCT” approach since it does not favor any period over any another for a given state. Again, the takeaway is that to take full advantage of the AoI of the network state, the AoI must be associated with the information (network state).

V. Conclusion

In this work, we studied the impact of the AoI of network state updates in a wireless SDN on the total network throughput. For each time interval after getting a network update, we formulated the throughput maximization problem as a POMDP where there are no observations for the time horizon before another network update occurs. To solve this problem, we use Monte Carlo planning-based approaches to generate policies for a given time horizon, which can even handle larger state spaces. For the purpose of this work, we focus on a special case of a two-user multi-link network model with a slow/fast fading interference model, and we derive the state-independent performance when there are no observations made, which characterizes the performance as $T \to \infty$. We also solve the full observability case (MDP) by

[2] The maximum AoI almost goes up to 20 for Model 4, because the state changes with low probability, so there is less of a penalty for having larger AoI and using a longer T is advantageous.

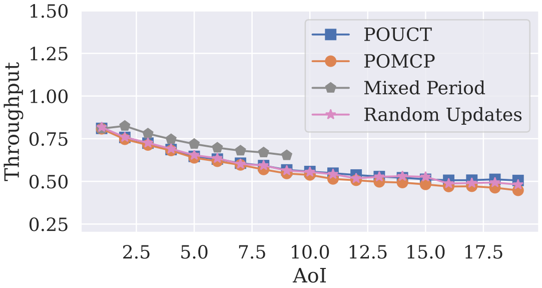

Fig. 16. Throughput vs. AoI, Model 3.

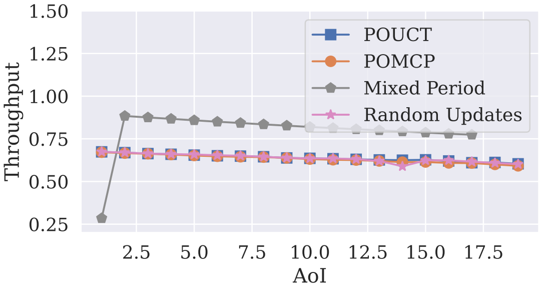

Fig. 17. Throughput vs. AoI, Model 4.

applying relative value iteration, which give us a throughput upper bound. Simulations of policies with periodic updating demonstrate that there is a finite optimal update period that maximizes the total network throughput. We also study non-uniform update intervals, including a policy that chooses the period depending on the network state observation, which is shown to outperform the periodic updating policies. The plots of throughput vs. AoI demonstrate the importance of associating AoI with the information, to more fully realize the gains when controlling the AoI. Future work includes optimizing the time horizons to account for the likely next observation, studying more general models of the network, and applying learning approaches when the network model parameters are unknown.

References

[1] S. Kaul, R. Yates, and M. Gruteser, “Real-time status: How often should one update?” in *Proc. IEEE INFOCOM*, Orlando, FL, Mar. 2012, pp. 2731–2735.

[2] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[3] J. Wang, Y. Miao, P. Zhou, M. S. Hossain, and S. M. M. Rahman, “A software defined network routing in wireless multihop network,” *Journal of Network and Computer Applications*, vol. 85, pp. 76–83, 2017, intelligent Systems for Heterogeneous Networks. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804516303058

[4] G. Wang, Y. Zhao, J. Huang, and W. Wang, “The controller placement problem in software defined networking: A survey,” *IEEE Network*, vol. 31, no. 5, pp. 21–27, 2017.

[5] Q. Qin, K. Poularakis, G. Iosifidis, S. Kompella, and L. Tassiulas, “Sdn controller placement with delay-overhead balancing in wireless edge networks,” *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1446–1459, 2018.

[6] D. Li, S. Wang, K. Zhu, and S. Xia, “A survey of network update in sdn,” *Frontiers of Computer Science*, vol. 11, no. 1, pp. 4–12, 2017. [Online]. Available: https://doi.org/10.1007/s11704-016-6125-y

[7] S. Luo, H. Yu, L. Luo, and L. Li, “Customizable network update planning in sdn,” *Journal of Network and Computer Applications*, vol. 141, pp. 104–115, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804519301675

[8] W. Ren, Y. Sun, H. Luo, and M. Guizani, “Bllc: A batch-level update mechanism with low cost for sdn-iot networks,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 1210–1222, 2019.

[9] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, and N. B. Shroff, “Update or wait: How to keep your data fresh,” *IEEE Transactions on Information Theory*, vol. 63, no. 11, pp. 7492–7508, Nov 2017.

[10] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, “Towards an effective age of information: Remote estimation of a Markov source,” to appear in 2018 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Apr. 2018.

[11] C. Kam, S. Kompella, and A. Ephremides, “Age of incorrect information for remote estimation of a binary markov source,” in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2020, pp. 1–6.

[12] A. Maatouk, S. Kriouile, M. Assaad, and A. Ephremides, “The age of incorrect information: A new performance metric for status updates,” *IEEE/ACM Transactions on Networking*, vol. 28, no. 5, pp. 2215–2228, 2020.

[13] S. Gopal and S. K. Kaul, “A game theoretic approach to dsrc and wifi coexistence,” in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2018, pp. 565–570.

[14] S. Gopal, S. K. Kaul, and R. Chaturvedi, “Coexistence of age and throughput optimizing networks: A game theoretic approach,” in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2019, pp. 1–6.

[15] N. Rajaraman, R. Vaze, and G. Reddy, “Not just age but age and quality of information,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1325–1338, 2021.

[16] C. Kam, S. Kompella, G. D. Nguyen, J. E. Wieselthier, and A. Ephremides, “Information freshness and popularity in mobile caching,” in *2017 IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 136–140.

[17] M. K. C. Shisher, H. Qin, L. Yang, F. Yan, and Y. Sun, “The age of correlated features in supervised learning based forecasting,” 2021.

[18] C. Kam, S. Kompella, and A. Ephremides, “Age of sensed information in a cognitive radio network,” in *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, 2021, pp. 1–8.

[19] ——, “The role of aoi in a cognitive radio network: Lyapunov optimization and tradeoffs,” in *MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM)*, 2021, pp. 303–308.

[20] D. Silver and J. Veness, “Monte-carlo planning in large pomdps,” in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23. Curran Associates, Inc., 2010. [Online]. Available: https://proceedings.neurips.cc/paper/2010/file/edfbe1afcf9246bb0d40eb4d8027d90f-Paper.pdf

[21] D. J. White, “Dynamic programming, markov chains, and the method of successive approximations,” *Journal of Mathematical Analysis and Applications*, vol. 6, no. 3, pp. 373–376, 1963.

[22] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 2, no. 2.

[23] P. J. Schweitzer, “Iterative solution of the functional equations of undiscounted markov renewal programming,” *Journal of Mathematical Analysis and Applications*, vol. 34, no. 3, pp. 495–501, 1971.

[24] K. Zheng and S. Tellex, “pomdp_py: A framework to build and solve pomdp problems,” in *ICAPS 2020 Workshop on Planning and Robotics (PlanRob)*, 2020, arxiv link: ”https://arxiv.org/pdf/2004.10099.pdf”. [Online]. Available: https://icaps20subpages.icaps-conference.org/wp-content/uploads/2020/10/14-PlanRob_2020_paper_3.pdf