International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.6, No.9, September 2017 DOI:10.15693/ijaist/2017.v6i11.456-460

Improving the Enhanced Performance for Network Utility Maximization Using Delay Based Scenario

Jalagam Priyanka[1], Dr. M. Sreedhar Reddy[2]

PG Student[1], Professor[2]
Department of Computer Science and Engineering
Samskruti College of Engineering and Technology
Kondapur, Ghatkesar, Hyderabad

ABSTRACT: Network utilities are software utilities designed to analyze and configure various aspects of computer networks. The majority of them originated on UNIX systems, but several later ports to other operating systems exist. It is well known that max-weight policies based on a queue backlog index can be used to stabilize stochastic networks, and that similar stability results hold if a delay index is used. Using optimization, we extend this analysis to design a utility Maximizing algorithm that uses explicit delay information from the head-of-line packet at each user. The resulting policy is shown to ensure deterministic worst-case delay guarantees and to yield a throughput utility that differs from the optimally fair value by an amount that is inversely proportional to the delay guarantee. Our results hold for a general class of 1-hop networks, including packet switches and multiuser wireless systems with time-varying reliability.

Key Words: Optimization, queueing, stochastic control.

I.INTRODUCTION

In the past decade, network utility maximization has attracted significant attention ever since the seminal framework was introduced. In the framework, network Protocols are understood as distributed algorithms that maximize aggregate user utility under wired or wireless network resource constraints. For the single-path unicast scenarios considered, user's utility function is typically assumed to be strictly concave function of user rate, and the resource constraints set is Linear. Various types of fairness across users can be warranted by choosing different utility functions. This Framework not only provides a powerful tool to reverse engineering existing protocols such as TC, but also Allows systematic design of new protocols for a Comprehensive review. This paper considers the problem of scheduling for maximum throughput utility in a network with random packet arrivals and time-varying channel reliability. We focus on 1-hop networks where each packet requires transmission over only one link. At every slot, the network controller assesses the condition of its channels and selects a set of links for transmission. The success of each transmission depends on the collection of links selected and their corresponding reliabilities. The goal is to maximize a concave and non decreasing function of the time-average throughput on each link. Such a function represents a utility function that acts as a measure of fairness for the achieved throughput vector. In the case when traffic is inside the network capacity region, the utility-optimal throughput vector is simply the vector of arrival rates, and the problem reduces to a network stability problem. In this case, it is well known that the network can be stabilized by max-weight policies that schedule links every slot to maximize a weighted sum of transmission rates, where the weights are queue backlogs. This is typically shown via a Lyapunov drift argument. This technique for stable control of a queueing network was first used for link and server scheduling and has since become a powerful method to treat stability in different contexts, including switches and computer networks, wireless systems and ad hoc mobile networks with rate and power allocation, and systems with probabilistic channel errors. In the case when traffic is either inside or outside of the capacity

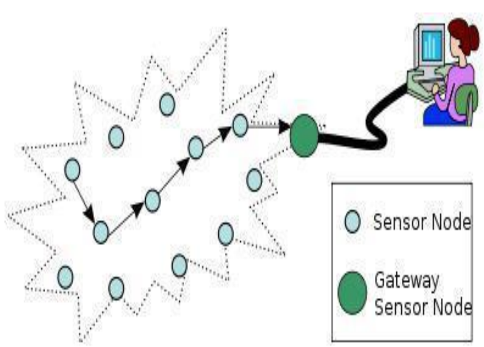
Region, it is known that the max-weight policy can be combined with a flow control policy to jointly stabilize the network while maximizing throughput utility. Utility optimization for the special case of "infinitely backlogged" sources and was perhaps first addressed for time-varying wireless downlinks without explicit queueing. The stability works all use backlog-based transmission rules, as do the works, which treat joint stability and utility optimization. However, introduces an interesting delay-based Lyapunov function for proving stability, where the delay of the head-of-line packet is used as a weight in the max-weight decision. This approach intuitively provides tighter control of the actual queueing delays. Indeed, a single head-of-line packet is scheduled based on the delay it has experienced, rather than on the amount of additional packets that arrived after it. This delay-based approach to queue stability, where the Modified Largest Weighted Delay First algorithm is developed, which uses a delay-based exponential rule. However, use delay-based rules only in the context of queue stability. To our knowledge, there are no prior works that use delay-based scheduling to address the important issue of joint stability and utility optimization. This paper fills that gap. We use a delay-based Lyapunov function and extend the analysis to treat joint stability and performance optimization via the Lyapunov optimization technique from our prior work. The extension is not obvious. Indeed, the flow control decisions in the prior work are made immediately when a new packet arrives, which directly affects the drift of backlog-based Lyapunov functions. However, such decisions do not directly affect the delay value of the head-of-line packets, and hence do not directly affect the drift of delay-based Lyapunov functions. We overcome this challenge with a novel flow control policy that queues all Arriving data, but makes packet dropping decisions just before advancing

a new packet to the head-of-line. This policy is structurally different from the utility optimization works. This new structure leads to deterministic guarantees on the worst-case delay of any non dropped packet and provides throughput utility that can be pushed arbitrarily close to optimal. Specifically, for any integer, we can construct an algorithm that ensures all non dropped packets have delay less than or equal to slots, with total throughput utility that differs from optimal by . The deterministic delay guarantee is particularly challenging to establish, and for this we introduce a new technique of concavely extending a utility function. We further show via simulation that our algorithms maintain good performance when the i.i.d. arrivals are replaced by ergodic but temporally correlated "bursty" arrivals with the same rates. However, the worst-case delay required to achieve the same utility performance is increased in this case. This is not surprising if we compare to known results for backlog-based Lyapunov algorithms. Backlog-based algorithms were first developed under i.i.d. assumptions, but with increased delay—for non-i.i.d. case. Thus, while we limit our analytical proofs to the i.i.d. setting, we expect the algorithm to approach optimal utility in more general cases, as supported by our simulations. While our algorithm can be used to enforce any desired delay guarantee, it is important to emphasize that it does not maximize throughput utility subject to this guarantee. Such a problem can be addressed with Markov decision theory, which brings with it the curse of dimensionality (see structural results and approximations and weighted stochastic shortest-path approaches). In this paper, we claim only that the achieved utility is within of the largest possible utility of any stabilizing algorithm. However,because (for large) our utility is close to this ideal utility value, it is even closer to the maximum utility that can be achieved subject to the worst-case delay constraint. That is because a basic stability constraint is less stringent than a worst case delay constraint, and so the optimal utility under a stability constraint is greater than or equal to the optimal utility under a worst-case delay constraint. Furthermore, our approach offers the low-complexity advantages associated with Lyapunov drift and Lyapunov Specifically, the policy makes real-time transmission decisions based only on the The flow control decisions here can also be implemented in a distributed fashion at each

II. NETWORK MODEL

A) COMPUTER NETWORKING

A hop is one portion of the path between source and destination. Data packets pass through routers and gateways on the way. Each time packets are passed to the next device, a hop occurs. To see how many hops it takes to get from one host to another ping or traceroute/tracepath commands can be used. Consider a 1-hop network that operates in discrete time with normalized time-slots. There are links, and packets arrive randomly every slot and are queued separately for transmission over each link. Current system state and does not require a priori knowledge of the channel-state probabilities. Transmission decisions based only on the flow control decisions here can also be implemented in a distributed fashion at each link, as is the case with most other Lyapunov based utility optimization algorithms. link,asis the case with most other Lyapunov-based utility optimization algorithms. It is known that average queue congestion and delay is convex in the arrival rate if traffic from an arbitrary arrival process is probabilistically split, this is not necessarily true (or relevant) for dynamically controlled networks, particularly when the control depends on the queue backlogs and delays themselves. Actual network delay problems involve not only optimization of rate based utility functions, but engineering of the Lagrange multipliers (which are related to queue backlogs) associated with those utility functions.



For simplicity, assume that each link can transmit at most one packet per slot, so that for all links and all n slots. It is useful to assume a link can transmit even if it does not have a packet, in which case a null packet is transmitted. a link condition vector for slot , which determines the probability of successful transmission on each slot. Specifically, given particular and vectors, the probability of successful transmission on link is given by a reliability function The reliability function for each is general and is assumed only to take real values between 0 and 1 (representing probabilities), and to have the property that whenever . The channel condition vector is assumed to be i.i.d. over slots and independent of the process. Assume it takes values in a set of arbitrary cardinality. The vector is known to the network controller at the beginning of each slot. In practice, is the result of a

channel measurement or estimation that is done every slot? The estimate might be in exact, in which case the reliability function represents the probability that the actual network channels on slot are sufficient to support the attempted transmission over link (given and the estimate for slot). We assume the reliability function is known. Recent online techniques for estimation of packet error rates are considered in. In the context, a number of other decision parameters to be chosen on each slot also affect reliability, such as modulation, power levels, sub band selection, coding type, etc. These choices can be represented as a parameter spac . In this case, the reliability function can be extended to include the parameter choice made every slot: . This does not change our mathematical analysis although for simplicity we focus on the reliability function structure of (2). We assume that ACK/NACK information is given at the end of the slot to inform each link if its transmission was successful or not. Packets that are not successful do not leave the queue (unless they are dropped in a packet drop decision). With this model of link success, the transmission variable in (1) is given by where is an indicator variable that is 1 if the transmission over link is successful, and 0 otherwise. That is with probability with probability The successes/failures over each link on slot are assumed to be independent of past events given the current and values. The successes/failures might be correlated over each link. This is not captured in the functions alone and can only be fully described by a joint success distribution function for all possible success/failure outcomes for a given and However, it turns out that the network capacity region, and hence the associated maximum utility point, is independent of such interlink success correlations [12]. Hence, it suffices to use only the marginal distribution functions for each.

B. EXAMPLES OF PACKET SWITCHES AND WIRELESS NETWORKS

The above model applies to a wide class of 1-hop networks. For example, it applies to the packet switch models of [5] and [7] by defining to be a null vector (so that there is no notion of channel variation) and by defining as the set of all link transmission vectors that satisfy permutation constraints (see Section VI-A). For wireless networks with interference but without time-varying channels, the set can be defined as all link activations that do not interfere with each other (i.e., that do not produce collisions), as in [3]. The reliability function can be used to extend the model to treat cases where interfering links result in probabilistic reception. Furthermore, the opportunistic scheduling systems of with time-varying ON/OFF channels can be modeled with being the vector of ON/OFF channel states on each slot, and with the function taking the value 1 whenever and , and 0 otherwise. Finally, the model supports probabilistic reception in the case when the link reliability can vary from slot to slot. A simple example is when represents the current probability that a link transmission would be successful, so that if this example has the success probability over link a pure function of and, and hence implicitly assumes that the set limits all simultaneous link transmissions to orthogonal channels. More complex inters channel interference models can be described by more complex functions.

III. DELAY-BASED FLOW CONTROL

Let be the vector of arrival rates, so that is the arrival rate to link (in units of packets/slot). The network capacity region is defined as the closure of the set of all long-term throughput vectors that the system can support. The set is known to be the same as the closure of the set of all arrival rate vectors for which there exists a stabilizing scheduling algorithm, subject to the constraint that the flow controllers are turned off. Specifically, it is shown that the set is given by the set of all time-average transmission rates that can be achieved by stationary and randomized algorithms, called -only algorithms, that observe every slot and choose a (possibly random) transmission vector according to a probability distribution that depends only on the observed channel state . Thus, for every vector, there is an -only algorithm, with a corresponding random service vector that yields for each where the expectation in is with respect to the distribution of and the distribution of given. Optimization Objective Let be a continuous and concave utility function of the - dimensional vector, where is used to represent the time-average throughput on each link (in units of packets/slot). The function can take positive or negative values and is assumed to be defined over the hypercube, where inequality is taken Entry wise, and are vectors with all entries equal to 0 and 1, respectively.

C. PROBLEM TRANSFORMATION WITH VIRTUAL QUEUES

It is not difficult to show that the stochastic network optimization problem can be transformed using a vector of auxiliary variables that are chosen every slot according to the constraints. The transformed problem is Maximize Subject to for all and are achievable on the network we say that a nonnegative discrete-time stochastic process is strongly stable if .This.

IV. SCHEDULING FOR A 3 BY 3 PACKET SWITCH

Here, we consider a crossbar constrained 3 by 3 packet switches, having three input ports and three output ports. There are nine queues, representing packets that arrived to input port that must be delivered to output port, for, Scheduling matrices are chosen every slot within the set of six permutation matrices, so that at most one packet is served per input and per output on a given slot. Arrival processes to each queue are independent Bernoulli processes, i.i.d. over slots with rates .We simulate the modified delay-based utility maximization algorithm of Section V, which does not require knowledge of the arrival rates . All simulations are over 1 million slots. The utility function of achieved throughput is where denotes the natural logarithm. We choose as a positive integer, so that the algorithm guarantees a worst-case delay of slots. We first consider a switch with feasible input rates, The rates are chosen so that all input ports and output ports have a loading of 0.95. For example, the loading of input port 1 is, being the sum of the rates in the first row of the matrix. Because input rates are inside the capacity region of the switch. Simulation of a 3 switch with overloaded traffic. The arrival rates are given. Each process is i.i.d. over slots. (a) Performance for overloaded switch. (b) Performance for overloaded switch. (c) Performance for overloaded switch. The utility optimal throughput matrix is. Thus, the algorithm should learn to drop as few packets as possible.We use, which guarantees a worst-case delay of slots.

V. OPPORTUNISTIC SCHEDULING FOR A TWO-USER WIRELESS DOWNLINK

Here, we consider a two-user wireless downlink with ON/OFF channels. The channel state processes are independent and i.i.d. over slots with and Arrivals are independent Bernoulli processes, i.i.d. over slots with rates and Every slot, the network controller observes the channel states and chooses a single queue to serve, transmitting exactly one packet over a served channel that is ON and no packets over a channel that is not served or that is OFF. The capacity region is shown. We simulate the delay-based algorithm of Section III-F, which uses knowledge of the arrival rates. We use a utility function and use, which yields near-optimal utility. All simulations run for 4 million slots. We create 50 different simulation runs, for arrival rates that scale linearly toward the point (0.5, 1.0). The resulting achieved throughput vectors are shown in the right panel. The example arrival rate points in the left panel of Fig. 6 are all inside the capacity region, and hence the achieved throughputs should be the same. This is indeed the case, as shown by the corresponding example points on the right panel. panel is outside of the capacity region, and its optimal achieved throughput is shown on the boundary point in the right panel. Note that once the arrival rates exceed the point in the left panel, the achieved throughput is the same and is very close to (0.4, 0.4) (shown as in the right panel).While these achieved throughputs are for the delay-based algorithm with known arrival rates, we note that we also simulated the modified delay-based algorithm with unknown arrival rates, as well as the queue-based algorithm of (version CLC2 in). The achieved throughputs for all of these algorithms are nearly identical, and the picture in the right panel.

VII. CONCLUSION

We have established a delay-based policy for joint stability and utility optimization. The policy provides deterministic worst-case delay bounds, with total throughput utility that is inversely proportional to the delay guarantee. The Lyapunov optimization approach for this delay-based problem is significantly different from that of back log based policies. We believe these results add significantly to our understanding of network delay and delay efficient control laws.

REFERENCES

1) M. J. Neely, “Delay-based network utility maximization,” in Proc. IEEEINFOCOM, Mar. 2010, pp. 1–9.
2) L. Georgiadis, M. J. Neely, and L.Tassiulas, “Resource allocation and cross-layer control in wireless networks,” Found.Trends Netw., vol. 1, no. 1, pp. 1–149, 2006.
3) L. Tassiulas and A. Ephremides,“Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks,” IEEE Trans. Autom. Control, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.
4) L. Tassiulas and A. phremides,“Dynamic server allocation to parallelqueues with randomly varying connectivity,” IEEE Trans. Inf. Theory, vol.39, no. 2, pp. 466–478, Mar. 1993.
5) N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, “Achieving 100% throughput in an input-queued switch,” IEEE Trans. Commun., vol. 47, no.8, pp. 1260–1267, Aug. 1999.
6) P. R. Kumar and S. P. Meyn, “Stability of queueing networks and scheduling policies,” IEEE Trans. Autom. Control, vol.40, no. 2, pp. 251–260, Feb. 1995
7) N.Kahale and P. E.Wright, “Dynamic global packet routing inwireless networks,” in Proc. IEEE INFOCOM, 1997, vol. 3, pp. 1414–1421.