A Novel Approach to Model and Control the Throughput of CSMA/CA Wireless Networks

Libin Jiang
Jean Walrand

Acknowledgement

This research was supported in part by MURI grant BAA 07-036.18.

A Novel Approach to Model and Control the Throughput of CSMA/CA Wireless Networks

Libin Jiang and Jean Walrand
EECS Department, University of California at Berkeley
{ljiang,wlr}@eecs.berkeley.edu

***Abstract*—In this paper, a novel model is proposed to analyze the throughput of multi-hop CSMA/CA network. Under certain assumptions, a product-form expression is derived for the stationary distribution of the underlying discrete-time Markov chain, and the throughput formula is given for each link. Interestingly, it is shown that the model includes slotted ALOHA, wireless LAN [6] and idealized CSMA [2] as special cases. Therefore, it not only provides a unified view of the different models but also covers many new cases. Based on the model, we design fully distributed algorithms to improve the throughput and fairness of CSMA/CA networks. In particular, a transmission-length control algorithm is shown to be throughput-optimal (even with collisions), although it has higher overhead than a transmission-probability control algorithm. And a cross-layer algorithm is given to achieve fair bandwidth sharing among different flows. Our analytical results are corroborated by comprehensive simulations.**

***Index Terms*—Modeling, Algorithm, CSMA/CA, wireless networks, Markov chain, optimization theory**

I. Introduction

The popularity of CSMA/CA (Carrier Sensing Multiple Access/Collision Avoidance) [1] protocol is due to its simplicity and its distributed nature. Nodes at different locations decide individually when to transmit based on a distributed carrier-sensing and random backoff algorithm. Modeling and optimizing the throughput performance of CSMA/CA network has been an active research area. Here we review a number of works that are most related to this paper.

In single-cell networks where the nodes can hear each other, all nodes have the same view of the activities in the network, which greatly simplifies the analysis. As in the well known model [6], time can be divided into idle slots, slots with successful transmissions, and slots with collisions. The probabilities of each type of slots can be computed, from which the individual throughput can be derived. Indeed, the models in [6] [7] are very accurate in predicting the network throughput.

In a more general topology where each node only hears a few neighbors, however, the activities of different nodes are not only related by *time*, but also by *space*. In particular, the nodes generally do not have the same view of channel activity and therefore they interact in a complicated way. For example, when a node k senses that the medium is idle and transmits a packet, it may collide with the simultaneously transmission(s) by one or more of its neighbors. Which neighbors could collide node k's packet, however, depend on whether the neighbors are "blocked" by other transmissions (i.e., waiting for the ending of those transmissions) which are not heard by node k. These "blockings" change and correlate in time and space, making the random process much harder to analyze.

Acknowledgement: This research was supported in part by MURI grant BAA 07-036.18.

Due to the complexity, related works in this area usually make varying degrees of assumptions and approximations. For example, [13] proposed an analytical model to study flow starvation and the effect of hidden nodes. It was assumed that every time the channel state changes, the next state (idle, busy, etc) has certain probabilities which does not depend on the current state. In [14], a comprehensive model was proposed to study the behaviors of 802.11 protocol, the impact of network topology and channel errors. It was assumed there that the collisions from different neighbors are independent. However, it is not obvious to what degree these assumption are accurate.

A methodology of modeling of CSMA/CA in general topology was proposed in [2], which was later furthered by other studies, e.g., [4], [5]. There, the transmitting and backoff process of different nodes was modeled as a continuous-time Markov chain with the "state" as the set of active nodes. This is similar to the call arriving and leaving process in circuit-switch networks [3]. The Markov chain has a simple product-form stationary distribution, determined only by the expected values (means) of the backoff time and transmission time, but is insensitive to the actual distributions of those times. However, to simplify the analysis, the model did not consider packet collisions by assuming that carrier-sensing is instantaneous (without propagation delay). So we refer to it as the *idealized CSMA* model.

In this paper, we propose a model to analyze general-topology CSMA/CA network with collisions as an integral part. The analysis is not intended to include all possible scenarios and details of the 802.11 standard, but to focus on how different links interact in time and space. By utilizing the time-reversibility in a class of interesting scenarios, we derive a product-form expression of the stationary distribution of the underlying discrete-time Markov chain, from which the throughput can be derived. Interestingly, with different assumptions on the transmission length, transmission probability and network topology, this model naturally specializes to slotted ALOHA [8], single-cell wireless LAN [6] and the idealized CSMA in [2], [4], [5]. Therefore it provides a unified view of the different models, and in the meantime covers new cases which are not in these models. Also, in most parts of the model we try not to make any approximation (except in section IV with binary exponential backoff).

Our model of CSMA/CA networks has immediate engineering applications. With the derived throughput formula, we design fully distributed algorithms to control the utilization of the network and fairness among different flows. Specifically we consider two problems: (a) given a set of feasible arrival rates at different links, how to tune the MAC-layer parameters (such as the transmission probabilities and transmission lengths) of different links to achieve the required rates? (b) With different data flows with flexible source rates in the network, how to adjust the source rates and MAC-layer parameters to achieve certain fairness objectives? For problem (a), we investigate and compare the pros and cons of two options: adjusting the transmission probability (or "Tx probability") and adjusting the transmission length (or "Tx length"). It is shown that proper choices of the Tx lengths can support any feasible arrival rates, and we propose a fully distributed algorithm to to find such Tx lengths.[1] On the other hand, if a similar algorithm is used to adjust the Tx probabilities, the system may become unstable and a set of feasible arrival rates may not be supported. However, the option of adjusting the Tx length requires that probe packets such as RTS/CTS [1] in 802.11 are used, thus may incur more overhead. For problem (b), we combine the Tx-length control algorithm and a simple source rate control algorithm to approximately achieve the fairness objective.

The rest of the paper is organized as follows. In section II we describe a simple CSMA/CA protocol with non-adaptive transmission probabilities and derive its throughput formula. In section III we show its relationship with the models of slotted ALOHA, wireless LAN and idealized CSMA. Section IV combines the result in section II with Bianchi's model of BEB (binary exponential backoff) to compute the throughput with IEEE 802.11 protocol. Section V verifies the above models by simulations. In section VI, we propose and compare a few distributed algorithms to control the throughput and fairness in CSMA/CA networks, and their performance is evaluated in section VII. We conclude the paper in section VIII.

II. Basic model and the throughput formula

We first describe the basic CSMA/CA protocols with non-adaptive transmission probabilities to be considered in this section. (Later we will extend it to the case of adaptive transmission probabilities in 802.11.) Let σ be the length of each idle slot (or "minislot"). (In 802.11a, for example, $\sigma = 9\mu s$.) In the following we will simply use "slot" to refer to the minislot. Each slot has an equal length of σ. Define a "link" as an (ordered) transmitter-receiver pair.

Basic protocol

In each slot, if link i is not already transmitting and if the medium is idle[2], the transmitter of link i tosses a coin and starts transmission with probability p_i (also denote $q_i := 1 - p_i$). If at a certain slot, link i did not choose to transmit but a conflicting link starts transmitting, then link i keeps silent until that transmission ends. If they transmit at the same time, then a collision happens.

In the main text, we focus on networks without hidden nodes, i.e., a link can sense the transmission from any other link that can collide with its transmission. So if a collision occurs, it must be that multiple conflicting links starting transmitting at the same slot. (In the Appendix, we discuss the extension of the model to networks with hidden nodes.)

We are interested in the following two setups:

Baseline setup: Assume that all the links transmit packets with an equal length of T slots, where T is an integer. (Later we will simply say that the length is T. Throughout the paper, all transmission times are assumed to be integer multiples of minislots.) When a collision occur, the collision time is also T. Fig. 1 (a) illustrates the baseline case.

Probe-packet setup: In this case, each link transmits a short probe packet with length γ before the data is transmitted. This is similar to the RTS/CTS mode in 802.11. Sending a short probe packet before the actual data can avoid collisions of long data packets[3] . On the other hand, the probe packet increases the overhead of successful transmissions. When a collision happens, only the probe packets collide, so each collision lasts a length of γ (we also call γ the "collision length"). Assume that a successful transmission of link i lasts T_i (which includes overhead τ' and the data payload). For the ease of exposition, assume that T_i's are fixed.[4] Fig. 1 (b) illustrates the Probe-packet setup.

We focus on the two setups because they are very revealing, and they possess a time-reversibility property that leads to a particularly simple throughput formula. A process is "time-reversible" if the process and its time-reversed process is statistically indistinguishable [10]. Indeed, in Fig 1 (a) and (b), if we look back in time, the reverse process perfectly follow the same protocol described above. A key reason is that the collisions start and finish at the same time.

In the rest of the section we use the notations of the probe-packet setup. Although the two setups are defined differently, the following results can be applied to the baseline setup by letting the "collision length" $\gamma = T$, the "successful transmission length" $T_i = T, \forall i$ and the overhead $\tau' = 0$. (In the baseline setup, we have assumed that there is no ACK or the length of ACK is negligible. To consider the length of ACK, one can define the collision length $\gamma = T$, the "successful transmission length" $T_i = T + T_{ACK}$ and the overhead $\tau' = 0$.)

A. Definitions and notations

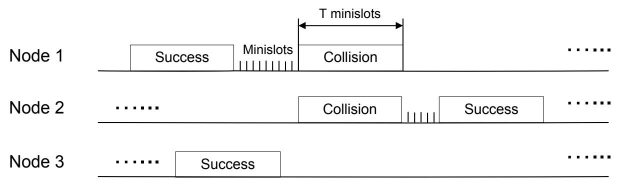
Denote the set of links by $\mathcal{N}$ and assume that there are K links (i.e., $K = |\mathcal{N}|$). We say that two links conflict if they can sense the transmission of each other (assume that if link i can sense link j, link j can also sense link i). Accordingly, define

[1]Therefore the algorithm is "throughput-optimal" in the sense that for any arrival rates that can be served by a centralized scheme, the algorithm can serve them in a distributed way.

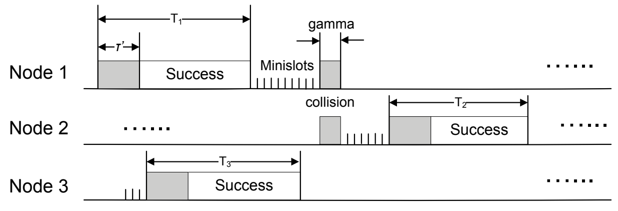
[2]By "link i not transmitting", we mean that "the transmitter of link i not transmitting".

[3]although RTS/CTS in 802.11 is originally designed to reduce the hidden-node and exposed-node problems [9].

[4]Using renewal theory, our result can be easily extended to the case where T_i's are random variables.

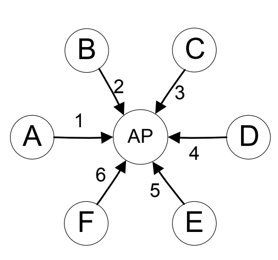


(a) Baseline setup

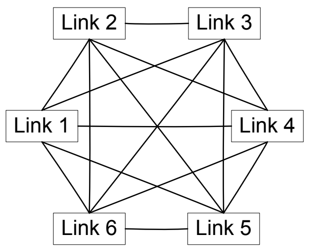


(b) Probe-packet setup

Fig. 1: The timeline in the Baseline setup and the Probe-packet setup



(a) Network topology



(b) Conflict graph

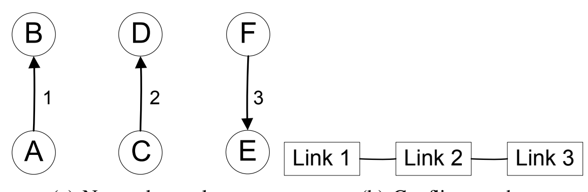
Fig. 2: Infrastructure network

G as the conflict graph. Each vertex in G represents a link, and there is an edge between two vertexes if the corresponding links conflict. Denote $e(i, j) = 1$ if there is an edge between link i and j.

For example, Fig. 2 (a) shows a wireless LAN with 6 links. Assume that all links can sense the transmissions of each other, then the network's conflict graph is a full graph (Fig. 2 (b)). (We use circles to represent nodes and squares to represent links.) Fig. 3 (a) shows an ad-hoc network with 3 links. Assume that link 1, 2 conflict, and link 2, 3 conflict. Then the network's conflict graph is Fig. 3 (b).

Let the "on-off state" be $x \in \{0, 1\}^K$, and x_k be the k'th element of x. Define $x_k = 1$ if the k'th link is active (transmitting) in state x, and $x_k = 0$ otherwise. Then x is a vector indicating which links are active at a given time.

Let $G(x)$ be the subgraph of G after removing all vertexes (each representing a link) with state 0 (i.e., any link j with $x_j = 0$) and their associated edges. In general, $G(x)$ is composed of a number of connected components (or simply called "components") $C_1(x), C_2(x), \dots, C_{M(x)}(x)$ where $M(x)$ is the total number of components in $G(x)$. If a component $C_m(x)$ has only one active link (i.e., $|C_m(x)| = 1$), then this link is having a successful transmission; if $|C_m(x)| > 1$, then all the links in the component are experiencing a collision. Let the set of "success" links in state x be $S(x) := \{k|k \in C_m(x)$ with $|C_m(x)| = 1\}$. And define the "collision number" $h(x)$ as the number of components in $G(x)$ with size larger than 1. That is, $h(x) := |\mathcal{N}(x)|$ where the index set $\mathcal{N}(x) := \{m \mid |C_m(x)| > 1\}$. Denote the set of links which are experiencing collisions as $\phi(x) := \cup_{m\in\mathcal{N}(x)} C_m(x)$.



(a) Network topology (b) Conflict graph

Fig. 3: Ad-hoc network

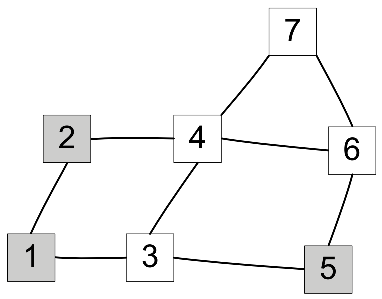


Fig. 4: Example: conflict graph and on-off states. Links 1, 2, 5 are on.

For example, Fig. 4 shows a conflict graph. The set of links $\mathcal{N} = \{1, 2, 3, 4, 5, 6, 7\}$. At state x, assume that link 1, 2 and 5 are active. Then there are two connected component in $G(x)$: $C_1(x) = \{1, 2\}, C_2(x) = \{5\}$, $M(x) = 2$. Since $|C_1(x)| = 2$, link 1 and 2 are experiencing a collision. Since $|C_2(x)| = 1$, link 5 is having a successful transmission. Consequently, $S(x) = \{5\}$, $\phi(x) = \{1, 2\}$ and the collision number $h(x) = 1$.

Further, define the "extended state" as

$$y := \{x, (a_k, \forall k : x_k = 1)\} \tag{1}$$

where a_k is the remaining time (including the current slot) before the transmission of link k ends. If $k \in S(x)$, then $a_k \in \{1, 2, \dots, T_k\}$; if $k \in \phi(x)$, then $r_k \in \{1, 2, \dots, \gamma\}$. An important observation here is that the transmissions in a collided component $C_m(x)$ is "synchronized", i.e., the links in $C_m(x)$ must have started transmitting at the same time, and will end transmitting at the same time, so all links in the component $C_m(x)$ have the same remaining time. That is, $a_k = a^{(m)}$ for any $k \in C_m(x)$ where $|C_m(x)| > 1$, and $a^{(m)}$ denotes the remaining time of the component $C_m(x)$. (To see this, any two links i and j in this component with an edge between them must have started transmitting at the same time. Otherwise, if i starts earlier, j would not transmit since it already hears i's transmission; and vice versa. By induction, all links in the component must have started transmitting at the same time.)

One can see that the transition among the set of extended states forms a discrete-time Markov chain. For example, in Fig. 4, assume that in the current slot, the extended state $y = \{x = (1100100), a_1 = a_2 = 5, a_5 = 2\}$. Then in the next slot, the remaining times will surely become $a_1 = a_2 = 4$, $a_5 = 1$.

Since link 1, 2, 5 are still active, link 3, 4, 6 cannot start transmitting. But link 7 will start transmitting with probability p_7, and stay inactive with probability $q_7 = 1 - p_7$. Therefore, in the next slot the state y will transit to state $y' = \{x = (1100101), a_1 = a_2 = 4, a_5 = 1, a_7 = T_7\}$ with probability p_7, and transit to state $y'' = \{x = (1100100), a_1 = a_2 = 4, a_5 = 1\}$ with probability q_7. The transition probabilities to other states are all 0.

B. Stationary distribution and throughput computation

The stationary distribution of the above Markov chain is expressed in the following theorem. (Readers who are more interested in the application of the expression could skip the proof.)

Theorem 1: In the stationary distribution, the probability of the extended state y as defined by (1) is

$$p(y) = \frac{1}{E} \prod_{i:x_i=0} (1 - p_i) \prod_{j:x_j=1} p_j = \frac{1}{E} \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i} \quad (2)$$

where E is a normalizing term such that $\sum_y p(y) = 1$. Interestingly, $p(y)$ only depend on x, but not the remaining time a_k's.

Proof: We verify that (2) satisfies the balance equation

$$p(y) = \sum_{y'} Q(y', y) p(y') \quad (3)$$

where $Q(y', y)$ is the transition probability from y' to y.

For a given state x and a component $C_m(x)$, define the "neighboring set" of $C_m(x)$, $\partial C_m(x)$, as the set of inactive links prevented from transmission by the links in $C_m(x)$. That is,

$$\partial C_m(x) := \{j | x_j = 0 \text{ and } e(j,k) = 1 \text{ for some } k \in C_m(x)\}$$

For example, in Fig. 4, $\partial C_1(x) = \{3, 4\}$.

Consider a state y as defined by (1) at time t. (In the following, y and $y(t)$ are interchangeable.) In the following we will consider which states at time $t-1$ can possibly transit to y (i.e., state y' with $Q(y', y) > 0$) and then verify the balance equation (3). Denote the index set of components in y whose transmissions started before time t as

$$\begin{aligned} \mathcal{B}(y) \quad = \quad & \{m \in \{1, 2, \ldots, M(x)\} \mid |C_m(x)| > 1, a^{(m)} < \gamma; \\ & \text{or } |C_m(x)| = 1, a_{k(m)} < T_{k(m)}\} \end{aligned}$$

where $k(m)$ denotes the only link in $C_m(x)$ when $|C_m(x)| = 1$.

Define

$$\begin{aligned} N_{\mathcal{B}}(y) &:= \cup_{m \in \mathcal{B}(y)} C_m(x) \\ \partial N_{\mathcal{B}}(y) &:= \cup_{m \in \mathcal{B}(y)} \partial C_m(x) \\ \mathcal{A}(y) &:= N_{\mathcal{B}}(y) \cup \partial N_{\mathcal{B}}(y). \end{aligned}$$

For example, in Fig. 4, assume that the links in $C_1(x)$ started their (collided) transmissions before t, and link 5 (the only link in $C_2(x)$) starts its transmission at time t. Then, $N_{\mathcal{B}}(y) = \{1, 2\}, \partial N_{\mathcal{B}}(y) = \{3, 4\}$ and $\mathcal{A}(y) = \{1, 2, 3, 4\}$.

By definition, all links in $N_{\mathcal{B}}(y)$ are on at time $t-1$. Also, the links in $\partial N_{\mathcal{B}}(y)$ must be off at time $t-1$. (If link $j \in \partial N_{\mathcal{B}}(y)$ was on at time $t-1$, it would have collided with some link in $N_{\mathcal{B}}(y)$. Thus it should also be on at time t, since all collided links start and end transmission at the same time. This leads to a contradiction.) Hence, the on-off state (x_j) of any link $j \in \mathcal{A}(y)$ do not change from time $t-1$ to t, and no coin toss is needed for link j at time t. In other words, the set $\mathcal{A}(y)$ (including link 1, 2, 3, 4 in Fig. 4) is the "rigid" part of the network from time $t-1$ to t.

On the other hand, any link $j \in \mathcal{A}^c(y) = \mathcal{N} \backslash \mathcal{A}(y)$ must toss a coin at time t to decide whether it transmits or not. This is because the following cases (where no coin toss is needed) are impossible: (1) link j has been on the same transmission at time $t-1$ and t, (2) link j is blocked by the same transmission by another link at time $t-1$ and t. Case (1) means that $j \in N_{\mathcal{B}}(y) \subseteq \mathcal{A}(y)$. Case (2) means that there is a link k with $e(k, j) = 1$ and $k \in N_{\mathcal{B}}(y)$, thus $j \in \mathcal{A}(y)$. Therefore both cases are impossible.

Denote $x_{\mathcal{A}} = \{x_k : k \in \mathcal{A}(y)\}$, $x_{\mathcal{A}^c} = \{x_k : k \in \mathcal{A}^c(y)\}$. Notice that at time $t-1$, any $y(t-1) \in \mathcal{F}$ can transit to state y at time t with probability $\prod_{k \in \mathcal{A}^c(y)} p_k^{x_k(t)} q_k^{1-x_k(t)}$, where $x_k(t)$ is part of $y(t) = y$, and

$$\begin{aligned} \mathcal{F} : \quad = \quad & \{y(t-1) | x_{\mathcal{A}}(t-1) = x_{\mathcal{A}}(t) \text{ and all active nodes in} \\ & \mathcal{A}^c(y), \text{ if any, have remaining time } 1\}. \end{aligned}$$

In particular, $x_{\mathcal{A}^c}(t-1)$ can be any 0-1 combination, and $x_{\mathcal{A}}$ remains unchanged from $t-1$ to t.

To see this, consider Fig. 4, where we have assumed that $\mathcal{A}^c(y) = \{5, 6, 7\}$. It is easy to check that at time $t-1$, the on-off states of link 5, 6, 7 can be any vector in $\{0, 1\}^3$. For example, say that $x_5(t-1) = 0, x_6(t-1) = x_7(t-1) = 1$, and link 6 and 7's transmissions end before time t. Then, the state transit to y with probability $p_5 q_6 q_7$.

In general, we have

$$\begin{aligned} & \sum_{y(t-1) \in \mathcal{F}} p(y(t-1)) \prod_{k \in \mathcal{A}^c(y)} p_k^{x_k(t)} q_k^{1-x_k(t)} \\ = \; & \Big(\frac{1}{E} \sum_{y(t-1) \in \mathcal{F}} \prod_{j \in \mathcal{N}} p_j^{x_j(t-1)} q_j^{1-x_j(t-1)}\Big) \prod_{k \in \mathcal{A}^c(y)} p_k^{x_k(t)} q_k^{1-x_k(t)} \\ = \; & \Big(\frac{1}{E} \sum_{y(t-1) \in \mathcal{F}} \prod_{j \in \mathcal{A}^c(y)} p_j^{x_j(t-1)} q_j^{1-x_j(t-1)}\Big) \cdot \\ & \prod_{j \in \mathcal{A}(y)} p_j^{x_j(t)} q_j^{1-x_j(t)} \prod_{k \in \mathcal{A}^c(y)} p_k^{x_k(t)} q_k^{1-x_k(t)} \\ = \; & \frac{1}{E} \prod_{j \in \mathcal{A}(y)} p_j^{x_j(t)} q_j^{1-x_j(t)} \prod_{k \in \mathcal{A}^c(y)} p_k^{x_k(t)} q_k^{1-x_k(t)} \\ = \; & p(y(t)) \end{aligned}$$

i.e., the distribution (2) is invariant (or "stationary"). ■

From this result, it is not difficult to have the following.

Theorem 2: With the stationary distribution, the probability of $x \in \{0, 1\}^K$ is

$$\begin{aligned} p(x) &= \frac{1}{E} (\gamma^{h(x)} \prod_{k \in S(x)} T_k) \prod_{i:x_i=0} (1 - p_i) \prod_{j:x_j=1} p_j \\ &= \frac{1}{E} (\gamma^{h(x)} \prod_{k \in S(x)} T_k) \prod_{i \in \mathcal{N}} p_i^{x_i} q_i^{1-x_i} \quad (4) \end{aligned}$$

where E is a normalizing term such that $\sum_x p(x) = 1$, i.e., all the probabilities sum up to 1.

Proof: By Theorem 1, we have

$$\begin{aligned} p(x) &= \sum_{y \text{ with on-off state } x} p(y) \\ &= \sum_{y \text{ with on-off state } x} \frac{1}{E} \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i}. \end{aligned}$$

In state x, there are $h(x)$ components with size larger than 1, and for each of them, the remaining time $a_m \in \{1, 2, \dots, \gamma\}$. For each link $k \in S(x)$, the remaining time $a_k \in \{1, 2, \dots, T_k\}$. Therefore, there are $\gamma^{h(x)} \prod_{k\in S(x)} T_k$ extended states with the same on-off state x, and each of them has equal probability as determined by (2). So

$$p(x) = \frac{1}{E}(\gamma^{h(x)} \prod_{k\in S(x)} T_k) \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i}.$$

■

After $p(x)$ is computed, the throughput of link k is

$$s_k = (1 - \frac{\tau'}{T_k}) \sum_{x:k\in S(x)} p(x). \quad (5)$$

where τ' is the overhead of a successful transmission (e.g., RTS, CTS, ACK packets). Note that the throughput is "normalized", i.e., it is the probability that link k is transmitting its payload in a given slot. (So if link k's PHY data rate used for payload transmission is R Mbps, then link k's actual throughput is $s_k R$ Mbps.)

III. Several interesting special cases

Theorem 2 is a quite general result which can specialize to the following interesting cases.

A. Slotted Aloha

By setting $\gamma = 1$, $T_k = 1, \forall k$ in our model, both the collision length and transmission length are one slot (of course, the slot here can be longer than $\sigma = 9\mu s$ in 802.11a), which is slotted Aloha. By Theorem 2, we have

$$p(x) = \frac{1}{E} \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i}.$$

It is easy to see that the normalizing constant $E = 1$. Thus, $p(x) = \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i}$, as expected.

B. CSMA with full conflict graph (e.g., single-cell wireless LAN)

If the conflict graph is a full graph, any pair of links conflict with each other, i.e., if more than one link transmit at the same time, there is a collision. In this scenario, the collision number $h(x) \le 1$. If $h(x) = 1$. then call x a "collision state" and let $p_C(x)$ be its probability. Note that $|S(x)| = 0$ in a collision state x. By Theorem 2,

$$\begin{aligned} p_c(x) &= \frac{1}{E}(\gamma^{h(x)} \prod_{k\in S(x)} T_k) \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i} \\ &= \frac{1}{E}\gamma \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i}. \end{aligned} \quad (6)$$

If $h(x) = 0$ and $|S(x)| = 0$, then no link is active ($x_k = 0, \forall k$) in the state. In this case x is a "idle state" and let $p_I(x)$ be its probability. Then

$$p_I(x) = \frac{1}{E} \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i} = \frac{1}{E} \prod_{i\in\mathcal{N}} q_i. \quad (7)$$

Finally, if $h(x) = 0$ and $|S(x)| = 1$, then only one link, denoted by $s(x)$, is active and its transmission is successful. Define x is a "success state" and let its probability be $p_S(x)$. Then

$$p_S(x) = \frac{1}{E} T_{s(x)} \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i} = \frac{1}{E} T_{s(x)} p_{s(x)} \prod_{i\ne s(x)} q_i. \quad (8)$$

Equation (6), (7) and (8) agree to the results in [6] for 802.11 networks when all links conflict with each other.

C. Idealized CSMA

In idealized CSMA [2], it is assumed that the carrier-sensing is instant (without delay). Therefore, a minislot is arbitrarily small such that the transmission probability p_i in each minislot can be made arbitrarily small to avoid collisions.

Consider a sequence of system indexed by n where the length of a minislot decreases like $1/n$. Since all transmission time is measured by the times of minislot, suppose that $\gamma^{(n)} = n \cdot \gamma$, $T_k^{(n)} = n \cdot T_k$ and $p_k^{(n)} = p_k/n$. By Theorem 2, we have

$$\begin{aligned} & p^{(n)}(x) \\ = \; & \frac{1}{E^{(n)}}((\gamma^{(n)})^{h(x)} \prod_{k\in S(x)} T_k^{(n)}) \prod_{i\in\mathcal{N}} (p_i^{(n)})^{x_i} (1 - p_i^{(n)})^{1-x_i} \\ = \; & \frac{1}{E^{(n)}} \prod_{k:x_k=1, k\in S(x)} [T_k^{(n)} p_k^{(n)}] \cdot \\ & \prod_{m\in\mathcal{G}(x)} [\gamma^{(n)}(\prod_{j\in C_m(x)} p_j^{(n)})] \cdot \prod_{i:x_i=0} (1 - p_i^{(n)}) \\ = \; & \frac{1}{E^{(n)}} \prod_{k:x_k=1, k\in S(x)} (T_k p_k) \cdot \\ & \prod_{m\in\mathcal{G}(x)} [\frac{\gamma}{n^{|C_m(x)|-1}} \prod_{j\in C_m(x)} p_j^{(n)}] \cdot \prod_{i:x_i=0} (1 - p_i^{(n)}). \end{aligned}$$

As $n \to \infty$, $\frac{\gamma}{n^{|C_m(x)|-1}} \to 0$ for $m \in \mathcal{G}(x)$ and $\prod_{i:x_i=0}(1 - p_i^{(n)}) \to 1$. Therefore

$$\begin{aligned} p^{(n)}(x) &\to \frac{1}{E_\infty} \prod_{k:x_k=1, k\in S(x)} (T_k p_k) \text{ if there is no collision component} \\ p^{(n)}(x) &\to 0 \text{ if there exists any collision component} \end{aligned} \quad (9)$$

where E_∞ is a proper normalizing constant.

As expected, there is no collision in the network as $n \to \infty$. Also, the non-collision states have a product-form distribution which matches the results in [2], [5], [4].

IV. Throughput in 802.11 network with adaptive contention windows

Using the method in [6], we compute the throughput with BEB (binary exponential backoff) via a fixed point equation. First, given a vector of transmission probabilities of all links $\mathbf{p} \in [0,1]^K$, the vector of collision probabilities $\mathbf{c} \in [0,1]^K$ can be computed by our model. In the following, we write $p(x)$ as $p(x;\mathbf{p})$ to reflect the dependency of the stationary distribution on $\mathbf{p}$.

In a long period of time W, the time that link k spends in collision is $W\sum_{x:k\in C_m(x),|C_m(x)|>1} p(x;\mathbf{p})$. Thus the number of collisions in W is $W\sum_{x:k\in C_m(x),|C_m(x)|>1} p(x;\mathbf{p})/\tau$. The time that link k spends in successful transmissions is $W\sum_{x:k\in S(x)} p(x;\mathbf{p})$, so the number of successful transmissions in W is $W\sum_{x:k\in S(x)} p(x;\mathbf{p})/T_k$. Therefore, its collision probability for a given transmission is

$$c_k = \frac{\sum_{x:k\in C_m(x),|C_m(x)|>1} p(x;\mathbf{p})/\tau}{\sum_{x:k\in C_m(x),|C_m(x)|>1} p(x;\mathbf{p})/\tau + \sum_{x:k\in S(x)} p(x;\mathbf{p})/T_k}. \tag{10}$$

On the other hand, the vector $\mathbf{p}$ is also a function of $\mathbf{c}$ by the well-known equation in [6].

$$p_k = \frac{2(1-2c_k)}{(1-2c_k)(W_k+1)+c_kW_k(1-(2c_k)^m)} \tag{11}$$

where W_k is the minimum contention window of link k, and m is the maximal backoff stage.

Using equations (10) and (11) we can solve for $\mathbf{p}=\mathbf{p}^*$ and $\mathbf{c}=\mathbf{c}^*$. Then, the normalized throughput of link k is

$$s_k(\mathbf{p}^*) = (1-\frac{\tau'}{T_k})\sum_{x:k\in S(x)} p(x;\mathbf{p}^*).$$

V. Comparison of Simulation and model

In this section we simulate two networks ("Network 1" and "Network 2") whose conflict graphs are shown in Fig. 3 (b) and Fig. 6 (a). First, we perform simulations with fixed transmission probabilities p_i. Assume the baseline setup, where the transmission length of each link is $T = 100$ minislots. Thus, $\gamma = 100$, $T_i = 100, \forall i$ and the overhead $\tau' = 0$. For convenience, in each simulation we set a common $p_i = p$ for each link. Fig. 5 (b) and Fig. 6 (b) compare each link's normalized throughput in the simulation with the value predicted by the model in the two networks. For each network, there are three sets of data corresponding to $p = 1/32, 1/16, 1/8$. The simulation results and the model match very well. Also, we observe that depending on the topology and the p value, different links can receive very uneven throughput.

Next, we present simulation results with BEB with maximal backoff stage $m = 7$. In each set of data in Fig. 7 and 8, we fix the minimum contention window $W_k = W, \forall k$. Again, there is a close match between our model and the simulation results.

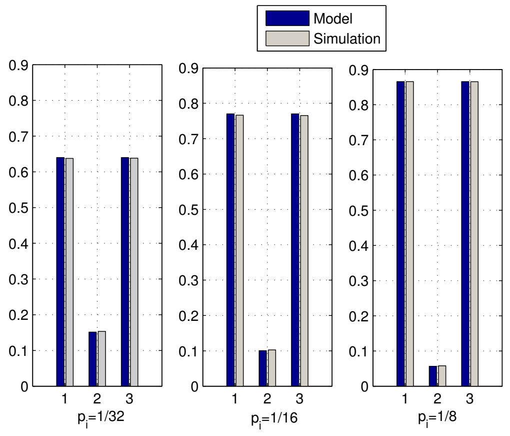
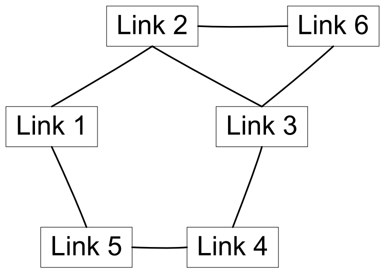
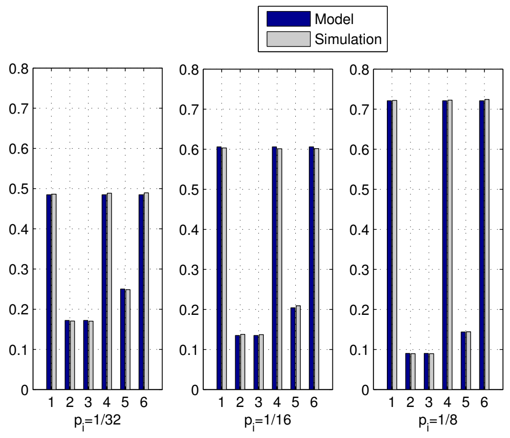


Fig. 5: Network 1



(a) Conflict graph of Network 2



(b) Comparison of simulation and model

Fig. 6: Network 2

VI. Distributed algorithms for high throughput and fairness

In this section we consider two problems: (a) In a CSMA/CA network with general topology, given a set of feasible arrival rates at different links, how to tune the transmission probability p_k's and transmission length T_k's to achieve the required rates? (b) With different data flows with flexible source rates in the network, how to adjust the source rates, p_k's and T_k's to achieve a fair allocation of bandwidth? Since the packet sizes are fixed in the baseline setup, in this section we assume the *Probe-packet setup*.

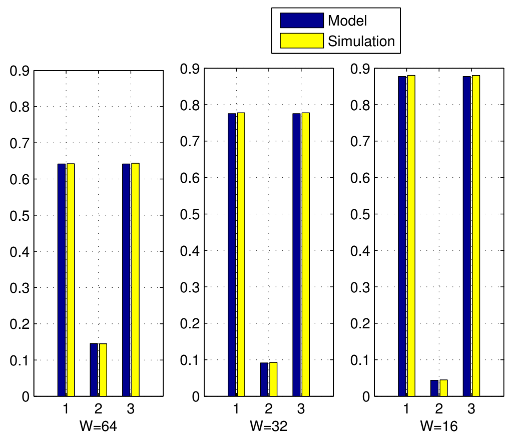


Fig. 7: Network 1 with BEB

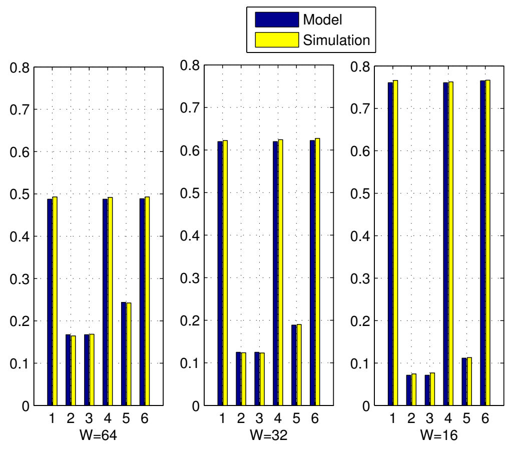


Fig. 8: Network 2 with BEB

To simplify the problem we consider two options: (a) adjust the Tx probabilities only, with fixed Tx lengths; (b) adjust the Tx lengths only, with fixed Tx probabilities. Option (a) turns out to be more difficult since increasing the Tx probability may increase collisions which is counter-productive. On the other hand, by using option (b) with probe packets (e.g., RTS/CTS), the negative effect of collisions will not worsen with larger Tx lengths, since only the small probe packets collide. In this section we mainly investigate option (b) but with comparison with option (a).

Use the throughput formula derived in section II, we design simple and distributed algorithms for the above purposes. The algorithms are very easy to implement. *Each link k adjusts its p_k or T_k (and the input rate) only based on its own queue length (or the observed arrival rate and service rate on the link).* Note that the nodes themselves do not need to know the throughput formula.

A. Algorithm to support any feasible arrival rate

First, consider the problem of supporting any feasible arrival rate (say, generated by UDP flows). Fix $p_k \in (0,1), \forall k$. (We choose $p_k = 1/16, \forall k$ such that the collision level is reasonably low, although the following discussion holds as long as $p_k \in (0,1), \forall k$.) By Theorem 2,

$$p(x) \propto (\prod_{k\in S(x)} T_k)\gamma^{h(x)} \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i} = g(x)\cdot(\prod_{k\in S(x)} T_k)$$

where $g(x) = \gamma^{h(x)} \prod_{i\in\mathcal{N}} p_i^{x_i} q_i^{1-x_i}$ is a constant which does not depend on the Tx length T_k's.

Denote $T_k := \tau' + T_0 \cdot \exp(r_k)$, where τ' is the overhead of a successful transmission (e.g., RTS, CTS, ACK packets), and $T_0 \cdot \exp(r_k)$ is the time used by the payload. $T_0 > 0$ is a "reference payload length". Let $\mathbf{r}$ be the vector of r_k's. Then

$$p(x;\mathbf{r}) = \frac{1}{E(\mathbf{r})} g(x) \cdot \prod_{k\in S(x)} (\tau' + T_0 \cdot \exp(r_k)) \tag{12}$$

where

$$E(\mathbf{r}) = \sum_{x'} [g(x') \cdot \prod_{k\in S(x')} (\tau' + T_0 \cdot \exp(r_k))]. \tag{13}$$

.

If at a state x, $k \in S(x)$, it is possible that link k is transmitting the overhead or the payload. We define a more detailed state (x,z), where $z \in \{0,1\}^K$. Let $z_k = 1$ if $k \in S(x)$ (i.e., k is in the "success" state) and link k is transmitting its payload (instead of overhead). Let $z_k = 0$ otherwise. Then similar to the proof of Theorem 2 (and equation (12)), we have

$$p((x,z);\mathbf{r}) = \frac{1}{E(\mathbf{r})} g(x,z) \cdot \exp(\sum_k z_k r_k) \tag{14}$$

where

$$g(x,z) = g(x) \cdot (\tau')^{|S(x)| - \mathbf{1}'\mathbf{z}} T_0^{\mathbf{1}'\mathbf{z}}. \tag{15}$$

where $\mathbf{1}'\mathbf{z}$ is the number of links that are transmitting the payload in state (x,z).

Definition 1: 1) A vector of arrival rate $\lambda \in \mathcal{R}^K$ (where K is the number of links) is *feasible* if it can be written as

$$\lambda_k = \sum_{(x,z)} \bar{p}((x,z)) \cdot z_k \tag{16}$$

where $\sum_x \bar{p}((x,z)) = 1$ and $\bar{p}((x,z)) \geq 0$ (i.e., $\bar{p}((x,z))$ is a probability distribution over the detailed state (x,z)). This is because if λ can be scheduled by the network, the fraction of time that the network spent in the detailed states must be non-negative and sum up to 1. (Note that (16) is the probability that link k is sending its payload given the distribution of the detailed states.)

For example, in the Ad-hoc network in Fig. 3, $\lambda = (0.5, 0.5, 0.5)$ is feasible, because (16) holds if we let the probability of the detailed state $(x = (1,0,1), z = (1,0,1))$ be 0.5, the probability of the detailed state $(x = (0,1,0), z = (0,1,0))$ be 0.5, and all other detailed states have probability 0.

2) A vector of arrival rate $\lambda \in \mathcal{R}^K$ is *strictly feasible* if it can be written as (16) where $\sum_x \bar{p}((x,z)) = 1$ and $\bar{p}((x,z)) > 0$ (which is a slightly stronger condition). In other words, $\lambda \in \mathcal{R}^K$ is strictly feasible if it is in the *interior* of the region of feasible arrival rates. In the previous example, $\lambda = (0.5, 0.5, 0.5)$ is not strictly feasible since

it cannot be written as (16) where all $\bar{p}((x,z)) > 0$. But $\lambda' = (0.49, 0.49, 0.49)$ is strictly feasible.

Theorem 3: Given any strictly feasible arrival rate vector $\lambda \in \mathcal{R}^K$, there exists $\mathbf{r}^* \geq 0$ such that the throughput (or "service rate") of link k is larger than the arrival rate for all k:

$$s_k(\mathbf{r}^*) = \frac{T_0 \cdot \exp(r_k^*)}{\tau' + T_0 \cdot \exp(r_k^*)} \sum_{x:k\in S(x)} p(x;\mathbf{r}^*) \geq \lambda_k, \forall k. \quad (17)$$

Remark 1: The result seems surprising since it says that even with collisions, and overhead τ' in each successful packet, the achievable throughput can still support any strictly feasible arrival rates. An intuitive explanation is as follows: if the payload size is large enough in each packet, and the duration of each collision does not increase with the transmission length, then the overhead and collisions are negligible compared to the time the system spends on successful payload transmissions. However, since long packets are not always preferable in practice, the arrival rates should not be set "too close" to the theoretical limit. In the UDP simulation later, we normally choose the arrival rates to be about 80% of the maximal possible rates.

Remark 2: The fact that proper choice of Tx lengths can approach the maximal throughput in a single-cell network is easy to recognize, but is non-trivial for general topology.

Proof: We start with the following function (the "log likelihood function" if we estimate the parameter $\mathbf{r}$ with the observation $\bar{\mathbf{p}}$). We will show that $\mathbf{r}^*$ that maximizes $F(\mathbf{r})$ satisfies (17).

$$\begin{aligned} & F(\mathbf{r}) \\ = & \textstyle\sum_{(x,z)} \bar{p}((x,z)) \log(p((x,z);\mathbf{r})) \\ = & \textstyle\sum_{(x,z)} \bar{p}((x,z))[\sum_k z_k r_k + \log(g(x,z)) - \log(E(\mathbf{r}))] \\ = & \textstyle\sum_k \lambda_k r_k + \sum_{(x,z)}[\bar{p}((x,z)) \log(g(x,z))] - \log(E(\mathbf{r})) \end{aligned}$$

where $\lambda_k = \sum_{(x,z)} \bar{p}((x,z)) \cdot z_k$ is the arrival rate at link k. Note that $F(\mathbf{r})$ is concave in $\mathbf{r}$. This is because (a) the first term is linear in $\mathbf{r}$; (b) the second term does not involve $\mathbf{r}$ and (c) $E(r)$ as defined in (13) can be expanded to a sum of exponential terms of $\mathbf{r}$. So $\log(E(\mathbf{r}))$ is a log-sum-exp function which is convex [12]. Therefore $F(\mathbf{r})$ is concave in $\mathbf{r}$.

Consider the following optimization problem

$$\sup_{\mathbf{r}\geq 0} F(\mathbf{r}) \ . \quad (18)$$

Since $\log(p((x,z);\mathbf{r})) \leq 0$, we have $F(\mathbf{r}) \leq 0$. Therefore $\sup_{\mathbf{r}\geq 0} F(\mathbf{r})$ exists. Since λ is strictly feasible, $\sup_{\mathbf{r}} F(\mathbf{r})$ can be attained (The proof of this subtle point.is similar to that in [11] and is omitted here. The model in [11], however, did not consider any collision.) So the problem is the same as $\max_{\mathbf{r}\geq 0} F(\mathbf{r})$. With $\eta \geq 0$ as the dual variables ($\eta \in \mathcal{R}^K$), the Lagrangian of the problem is $\mathcal{L}(\mathbf{r};\eta) = F(\mathbf{r}) + \eta^T\mathbf{r}$. Let η^* be the optimal vector of dual variables. Thus, the solution of (18), $\mathbf{r}^*$, satisfies

$$\begin{aligned} & \frac{\partial \mathcal{L}(\mathbf{r}^*;\eta^*)}{\partial r_k} = \frac{\partial F(\mathbf{r}^*)}{\partial r_k} + \eta_k^* \\ = & \lambda_k - \frac{1}{E(\mathbf{r}^*)} \sum_{x:k\in S(x)} [g(x) \cdot T_0 \cdot \exp(r_k^*) \cdot \\ & \prod_{j\in S(x), j\neq k} (\tau' + T_0 \cdot \exp(r_j^*))] + \eta_k^* \\ = & \lambda_k - \frac{1}{E(\mathbf{r}^*)} \frac{T_0 \cdot \exp(r_k^*)}{\tau' + T_0 \cdot \exp(r_k^*)} \sum_{x:k\in S(x)} [g(x) \cdot \\ & \prod_{j\in S(x)} (\tau' + T_0 \cdot \exp(r_j^*))] + \eta_k^* \\ = & \lambda_k - \frac{T_0 \cdot \exp(r_k^*)}{\tau' + T_0 \cdot \exp(r_k^*)} \sum_{x:k\in S(x)} p(x;\mathbf{r}^*) + \eta_k^* \\ = & \lambda_k - s_k(\mathbf{r}^*) + \eta_k^* = 0. \end{aligned}$$

Since $\eta_k^* \geq 0$, we have $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$. ■

Since $\mathbf{r}^*$ that maximizes $F(\mathbf{r})$ makes $s_k(\mathbf{r}^*) \geq \lambda_k, \forall k$, then we design the algorithm to solve problem (18) in a distributed way. For this purpose, r_k is changed in the direction of the gradient $\partial F(\mathbf{r})/\partial r_k = \lambda_k - s_k(\mathbf{r})$. That is, if $\lambda_k > s_k(\mathbf{r})$, then increase r_k; if $\lambda_k < s_k(\mathbf{r})$, then decrease r_k.

ALGORITHM 1. TX-LENGTH CONTROL ALGORITHM to support any strictly feasible arrival rate:

The vectors $\mathbf{r}$ is updated at time $t_i, i = 1, 2, \ldots$. Let $t_0 = 0$ and $t_i - t_{i-1} = M$ (millisecond), $i = 1, 2, \ldots$. Let "period i" be the time between t_{i-1} and t_i, and $\mathbf{r}(i)$ be the value of $\mathbf{r}$ at the end of period i, i.e., at time t_i. Initially, link k set $r_k(0) = 0$. Then at time $t_i, i = 1, 2, \ldots$, update

$$r_k(i) = \min\{[r_k(i-1) + \alpha(i)(\lambda'_k(i) - s'_k(i))]_+, r_{max}\}$$

where $\lambda'_k(i), s'_k(i)$ are the empirical average arrival rate and service rate in period i. [5] Note that $\lambda'_k(i), s'_k(i)$ are random variables which are generally not equal to λ_k and $s_k(\mathbf{r}(i))$ in period i. But as will be discussed below, if the step size $\alpha(i)$ is small, then $\mathbf{r}(i)$ is quasi-static and the randomness is averaged over multiple periods.

The intuition of the algorithm is very simple: if the empirical arrival rate of link k is larger than the service rate, then link k should transmit more aggressively by using a larger Tx length, and vice versa.

Remark: Note that the algorithm requires that probe packets like RTS are used. This is to ensure that the length of collision does not increase when the transmission length increases. This, however, introduces overhead for successful transmissions.

Theorem 4: Assume that the vector of arrival rates λ can be supported by some $\mathbf{r} \in [0, r_{max}]^K$.[6]

(i) If $\alpha(i)$ is non-increasing and satisfies $\sum_i \alpha(i) = \infty$ and $\sum_i \alpha(i)^2 < \infty$, then the algorithm converges, i.e., $\mathbf{r}(i) \rightarrow \mathbf{r}^*$ as $i \rightarrow \infty$ with probability 1, and $\liminf_{N\rightarrow\infty} \sum_{i=1}^N s'_k(i)/N \rightarrow \lambda_k$.

[5]Let link k send dummy packets when the queue is empty. This is to ensure that the CSMA Markov chain has the right stationary distribution.

[6]Clearly, as $r_{max} \rightarrow \infty$, the capacity region goes to the maximum.

(ii) If $\alpha(i) = \alpha$ (i.e., constant step size), then $\liminf_{N\to\infty} \sum_{i=1}^{N} s'_k(i)/N \to \lambda_k$ as $\alpha \to 0$.

Remark: If we replace $\lambda'_k(i)$ in Algorithm 1 with $\lambda'_k(i)+\epsilon$ where $\epsilon > 0$ is small enough such that $\lambda+\epsilon\mathbf{1}$ can be supported by some $\mathbf{r} \in [0, r_{max}]^K$. Then all queues are positive recurrent in (i). The same is true in (ii) if α is small enough.

Proof: Here we give an intuitive explanation. The complete proof is omitted due to the limit of space. Since r_k is always bounded in the algorithm, the mixing time of the CSMA Markov chain is bounded. Then as the step size becomes smaller and smaller, r_k is "quasi-static" such that the service rate is averaged (over multiple periods) to $s_k(\mathbf{r})$, and the arrival rate is averaged to λ_k. Thus the algorithm solves the optimization problem (18) by a stochastic approximation argument in [18]. (In [17], the convergence of the algorithms in [11] is proved similarly using the result in [18].) ■

B. Algorithm to achieve fairness among flows

The second problem we consider is how to achieve fair bandwidth allocation in general topology. For ease of exposition we assume single-hop flows (Algorithms for multi-hop flow can be obtained similarly). Assume that each flow k has a utility function $v_k(f_k)$ where $f_k \in [0,1]$ is the flow rate, and $v_k(\cdot)$ is increasing and strictly concave. Write $\mathbf{f} \in [0,1]^K$ as the vector of f_k's. The fairness objective used here is the total utility of all flows in the network.

ALGORITHM 2. A JOINT ALGORITHM TO ACHIEVE FAIR BANDWIDTH ALLOCATION:

The following distributed algorithm approximately maximizes the total utility of the network with one-hop flows.[7]

The vectors $\mathbf{r}$ and $\mathbf{f}$ are updated at time t_i, $i = 1,2,\ldots$. Let $t_0 = 0$ and $t_i - t_{i-1} = M$ (millisecond), $i = 1,2,\ldots$. Let "period i" be the time between t_{i-1} and t_i, and $\mathbf{r}(i), \mathbf{f}(i)$ be the value of $\mathbf{r}$ and $\mathbf{f}$ at the end of period i, i.e., at time t_i. Initially, link k set $r_k(0) = 0, f_k(0) = 1, \forall k$. Then at time t_i, $i = 1,2,\ldots$, do the following:

- Congestion control: Link k sets $f_k(i) = \rho \cdot \hat{f}_k(i)$ where ρ is slightly smaller than 1 and $\hat{f}_k(i) = \arg\max_{f'\in[0,1]}[\beta \cdot v_k(f') - r_k(i)f']$, where $\beta > 0$ is a "weighting factor".
- r_k is updated as follows: $r_k(i) = [r_k(i-1) + \alpha(i)(\hat{f}_k(i-1) - s'_k(i))]_+$, where $s'_k(i)$ is the empirical average service rate in period i, and $\alpha(i) > 0$ are step sizes.

Part (2) of the algorithm means that if the arrival rate of flow k is larger than the service rate, then r_k should be increased, i.e., link k transmits more aggressively. Meanwhile, part (1) means that the source rate should be reduced.

Larger value of β leads to higher total utility but longer packets. The reason is similar to that mentioned in Theorem 3. Since long packets are not always preferable in practice, the weighting factor β can be used to control the tradeoff.

Although the algorithm is very intuitive, the explanation is a little involved. For brevity, we give it in Appendix IX-B.

[7]For multi-hop flows, similar algorithms can be obtained by using the "back-pressure" concept [16]. The "back-pressure" of a link is the difference between the queue lengths (or certain dual variables) of the transmitter and the receiver of the link. Details are omitted here for simplicity.

The convergence properties of the algorithm are given in the following.

Theorem 5: Assume that $v'_k(0) < G < \infty$ for all k.

(i) If $\alpha(i)$ is non-increasing and satisfies $\sum_i \alpha(i) = \infty$ and $\sum_i \alpha(i)^2 < \infty$, then the algorithm converges, i.e., $\hat{f}_k(i) \to \hat{f}^*_k = s_k(\mathbf{r}^*)$ as $i \to \infty$ with probability 1, where $\hat{f}^*_k, \mathbf{r}^*$ are the optimal solution or optimal dual variables of problem (20), which approximately maximize the total utility. And all queues are stable (positive recurrent).

(ii) If $\alpha(i) = \alpha$ (i.e., constant step size), then $\liminf_{N\to\infty} \sum_{i=1}^{N} \hat{f}_k(i)/N \to \hat{f}^*_k = s_k(\mathbf{r}^*)$ as $\alpha \to 0$. All queues are stable (positive recurrent) if α is small enough.

Proof: We give a sketch of the proofs. First, r_k is always bounded in the algorithm. In particular, $r_k(i) \le \beta \cdot G + \max_j \alpha(j), \forall k, i$. (This can be proved by induction: (a) $r_k(0) = 0 \le \beta \cdot G + \max_j \alpha(j)$; (b) If $r_k(i) \ge \beta \cdot G$, then $\hat{f}_k(i) = 0$, which makes $r_k(i+1) \le r_k(i)$; if $r_k(i) < \beta \cdot G$, then $r_k(i+1) \le \beta \cdot G + \alpha(i+1) \le \beta \cdot G + \max_j \alpha(j)$.) Then similar to the proof of Theorem 4, the service rate is averaged as the step size goes smaller and smaller. Thus the algorithm solves the optimization problem in Appendix IX-B.

In (i), since $\hat{f}_k(i) \to \hat{f}^*_k = s_k(\mathbf{r}^*)$, and the actual input rate $f_k(i) < \hat{f}_k(i)$, thus all queues are positive recurrent. (This idea has been used in [19] to not only reduce the queueing delay, but also enable per-neighbor queues instead of per-flow queues in multi-hop networks.) A similar conclusion applies to (ii) if α is small enough. ■

VII. SIMULATION STUDY OF THE ALGORITHMS

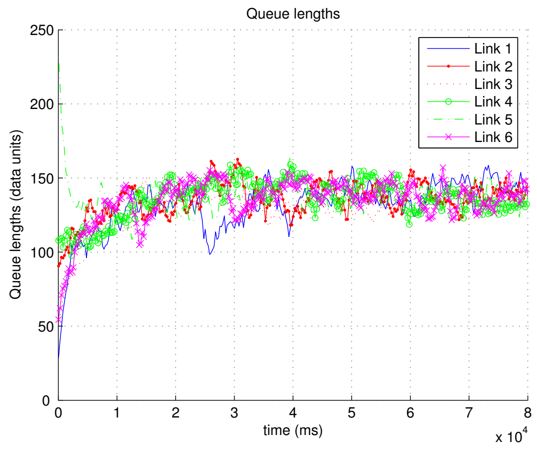
A. Fixed arrival rates

We evaluate the performance of the Tx-length control algorithm, with given arrival rate vector λ (for example, generated by UDP flows). In the following simulations, the update in Algorithm 1 is performed every $M = 5ms$. Let the constant step size $\alpha = 0.023$, and the "reference payload length" $T_0 = 50$. The upper bound $r_{max} = 3$. Since the transmission length $T_k = \tau' + \exp(r_k)$ is assumed to be an integer, we round up $\exp(r_k)$ to the nearest integer. For comparison purpose, we also simulate a similar algorithm based on adjusting the Tx probability, where we fix T_k and let

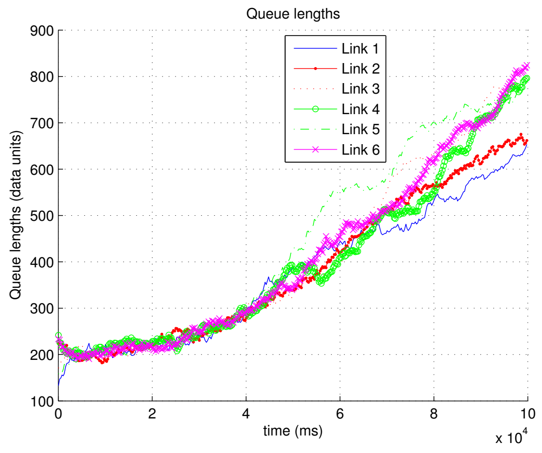
$$p_k = \min\{\frac{1}{16}\exp(r_k), 0.25\}$$

so that p_k increases with $\mathbf{r}_k$ but is upper bounded by 0.25 (which roughly corresponds to a contention window of 8 in the language of 802.11.)

First consider the network in Fig. 2 (a wireless LAN). Assume arbitrary initial values of r_k's between 0 and 1.15 at time 0. And in Algorithm 1, fix the Tx probabilities at $p_k = 1/16, \forall k$ and in the Tx-probability control algorithm, fix the Tx length at $T_k = 100, \forall k$. In both cases, the collision length (i.e., RTS length) is $\gamma = 10$, and the overhead of successful transmission is $\tau' = 20$. For convenience, assume that the PHY data rates of all links used for payload transmission is 1 "data unit" per millisecond. (For example, if this PHY data rate is 12Mbps, then 1 data unit here is equal to 12 kilobits.) Therefore if the normalized throughput of link k is s_k, then the actual throughput is s_k data unit per millisecond.



(a) Queue lengths with Tx-length control algorithm



(b) Queue lengths with Tx-probability control algorithm

Fig. 9: Adjusting Tx lengths or Tx probabilities

Let the normalized arrival rate $\lambda_k = 0.1325$ for all link $k = 1, 2, \ldots, 6$. Since $\sum_k \lambda_k < 1$, the arrival rates are strictly feasible for the Tx-length algorithm. Also, it can be achieved by the Tx-probability algorithm with $p_k = 0.034, \forall k$ or $p_k = 0.132, \forall k$. Then, each link adjusts its Tx length (or Tx probability) individually based on its local information. From Fig. 9, we see that the queues are stable with the Tx-length algorithm, but may be unstable with the Tx-probability algorithm, even if λ is feasible. The reason for the instability is that multiple links may find that their service rates are not high enough and thus increase the Tx probabilities simultaneously. This, however, may lead to more collisions and further reduce the service rates. This vicious cycle goes on which destabilizes the queues.

Next we simulate the ad-hoc network in Fig. 3. The arrival rate vector $\lambda = (0.4, 0.4, 0.4)$, which is strictly feasible since $\lambda_1 + \lambda_2 < 1$ and $\lambda_2 + \lambda_3 < 1$. As before, fix the Tx probabilities at $p_k = 1/16, \forall k$. Fig. 10 shows the queue lengths with the Tx-length algorithm. As expected, the queues are stable which means that the arrival rates are supported. Also, since link 2 faces more contention, its Tx length is longer than other links.

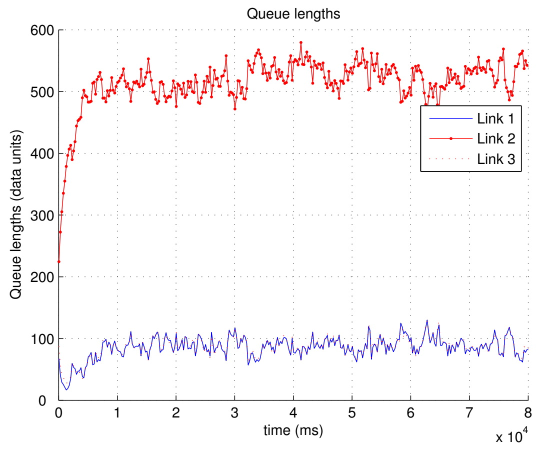


Fig. 10: Tx-length control algorithm in the Ad-hoc network in Fig. 3

B. Adaptive input rates to achieve fairness among flows

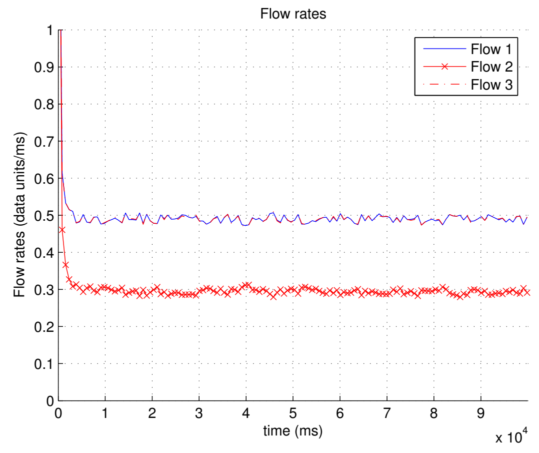
This section presents simulation results with Algorithm 2. The parameters $M = 5ms$, $\alpha = 0.023$, and $T_0 = 5$. For simplicity, assume that all flows are one-hop and each of them uses one link. Fig. 11 (a) shows the flow rates in the ad-hoc network in Fig. 3. The utility function of each flow is $v_k(f_k) = \log(0.05 + f_k)$, and $\beta = 1.5$. It is justified that flow 2 receives a lower throughput because in some sense, its transmission uses more resources than the flow 1 and 3 (since flow 1, 3 have to be quiet when flow 2 is active). Fig. 11 (b) shows the flow rates in the wireless LAN, where the utility of flow k is assumed to be $v_k(f_k) = (k/5) \cdot \log(0.05 + f_k), k = 1, 2, \ldots, 6$, and $\beta = 1$. The throughput allocation is reasonable in view of their different utility functions.

VIII. Conclusion

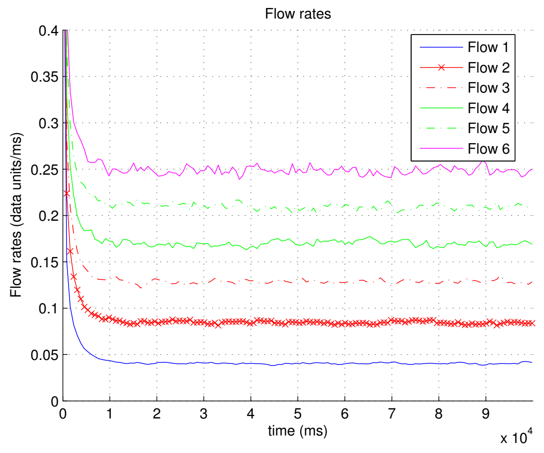
We have proposed a model to analyze the throughput of CSMA/CA network in general topology. By utilizing the time-reversibility in a class of scenarios, we are able to to compute the throughput and collision probabilities of each link. Via a fixed-point equation, the throughput with adaptive contention windows (i.e., BEB) can also be computed. Moreover, the model gives a unified view of the models of slotted ALOHA, single-cell wireless LAN and the idealized CSMA and reveals their connections.

With this model, we have further developed distributed algorithms to control the throughput and fairness of different data flows. We have studied and compared the pros and cons of adjusting the Tx length or adjusting the Tx probability. It was shown that a Tx-length control algorithm can support any strictly feasible arrival rate, and has good convergence property. Tx-probability algorithm, on the other hand, has lower overhead but may become unstable even if the arrival rates are feasible. A joint rate control and Tx-length control algorithm has been proposed to achieve fairness in distributed CSMA/CA networks.

With the expressions for the stationary distribution of the Markov chain, we could verify whether certain assumptions



(a) Bandwidth sharing in the Ad-hoc network in Fig. 3



(b) Bandwidth sharing in the wireless LAN in Fig. 2

Fig. 11: Achieving fairness by Algorithm 2

made in [13], [14] hold exactly or approximately. This is also a subject for future study.

References

[1] IEEE 802.11 specification, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1999.

[2] R. Boorstyn, A. Kershenbaum, B. Maglaris and V. Sahin, "Throughput Analysis in Multihop CSMA Packet Radio Networks," IEEE Trans. on Communications, 35(3):267-274, March 1987.

[3] F. P. Kelly, "Loss networks," Ann. Appl. Prob., vol. 1, no. 3, 1991.

[4] X. Wang and K. Kar, "Throughput Modelling and Fairness Issues in CSMA/CA Based Ad-Hoc Networks," Proceedings of IEEE Infocom 2005, Miami, March 2005.

[5] S.C. Liew, C. Kai, J. Leung, B. Wong, "Back-of-the-Envelope Computation of Throughput Distributions in CSMA Wireless Networks," submitted for publication, http://arxiv.org//pdf/0712.1854

[6] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," IEEE Journal on Selected Areas in Communications, 18(3):535–547, 2000.

[7] A. Kumar, E. Altman, D. Miorandi, and M. Goyal, "New insights from a fixed point analysis of single cell IEEE 802.11 WLANs," In Proc. IEEE INFOCOM, USA, March 2005.

[8] Tobagi, F.A., "Modeling and performance analysis of multihop packet radio networks," Proceedings of the IEEE, vol. 75, no. 1, pp. 135- 155, Jan. 1987.

[9] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A Media Access Protocol for Wireless LANs. In Proc. ACM SIGCOMM, London, UK, 1994.

[10] F. P. Kelly, "Reversibility and Stochastic Networks," Wiley, 1979.

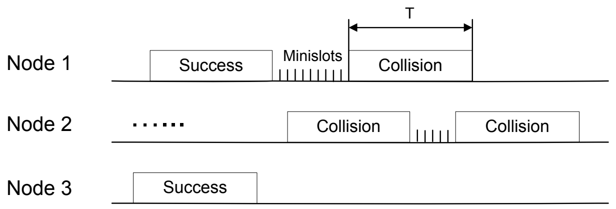


Fig. 12: Timeline with hidden nodes (Here, link 1 and 2 are hidden from each other)

[11] L. Jiang and J. Walrand, "A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks," the Forty-Sixth Annual Allerton Conference on Communication, Control, and Computing, Sep. 23-26, 2008.

[12] S. Boyd and L. Vandenberghe, "Convex Optimization", Cambridge University Press, 2004.

[13] M. Garetto, T. Salonidis and E. Knightly, "Modeling Per-flow Throughput and Capturing Starvation in CSMA Multi-hop Wireless Networks," In Proc. IEEE INFOCOM, Spain, 2006.

[14] M. Carvalho and J. J. Garcia-Luna-Aceves, "A Scalable Model for Channel Access Protocols in Multihop Ad Hoc Networks," In Proc. ACM MobiCom, Philadelphia, PA, USA, September 2004.

[15] K. Jamieson, H. Balakrishnan, "PPR: partial packet recovery for wireless networks," ACM SIGCOMM Computer Communication Review, vol. 37, no. 4, Oct. 2007.

[16] L. Tassiulas, "Adaptive back-pressure congestion control based on local information," IEEE Transactions on Automatic Control, Vol. 40, No. 2, pp. 236-250, February 1995.

[17] J. Liu, Y. Yi, A. Proutiere, M. Chiang, and H. V. Poor, "Maximizing utility via random access without message passing," Microsoft Research Technical Report, September 2008.

[18] V. Borkar, "Stochastic Approximation: A Dynamical Systems Viewpoint," 2008, draft copy.

[19] Loc Bui, R. Srikant, and Alexander Stolyar, "Novel Architectures and Algorithms for Delay Reduction in Back-Pressure Scheduling and Routing," to appear in the IEEE INFOCOM 2009 Mini-Conference.

[20] F. P. Kelly, A.K. Maulloo and D.K.H. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," Journal of the Operational Research Society 49, pp. 237-252, 1998.

IX. Appendix

A. Throughput with Hidden Nodes

1) Computing the rates of uncorrupted bits: In this section, we discuss how to compute the throughput of individual links with the existence of hidden nodes.

We say that link 1 *interferes* with link 2 if link 1's packet corrupts link 2's packet if the two packets overlap. And define that link 1 is *hidden* from link 2 if link 2 cannot hear link 1 but link 1 interferes with link 2. Define the hearing matrix H whose element $H_{ij} = 1$ if link i can be heard by link j. Assume that H is symmetric, i.e., $H_{ij} = H_{ji}$. Similarly define the interference matrix F whose element $F_{ij} = 1$ if link i interferes with link j. Also, assume the baseline setup, where all the links transmit packets with equal length T, and no RTS/CTS is used. Fig. 12 illustrates the timeline in a simple network where link 1 and 2 are hidden from each other.

Since the links decide when to transmit based on the hearing matrix H, the random process can be analyzed as before by defining the conflict graph based on H. Specifically, there is an edge between two links i, j if and only if $H_{ij} = H_{ji} = 1$.

Then, based on Theorem 2, the probability of all states can be obtained. In state x, the bits transmitted by i is successful iff $x_i = 1$ and $x_k = 0$ for any interfering link k (including the links hidden from i), . So, the probability that link i is transmitting uncorrupted bits is

$$\bar{s}_i = \sum_x p(x) I\{x_i = 1, x_k = 0, \forall k : F_{ki} = 1\} \tag{19}$$

where $I\{\cdot\}$ is the indicator function.

Note that since a link j which is hidden from link i can start transmission when link i's packet is ongoing (Fig. 12), it's likely that only part of link i's packet is corrupted. So $\bar{s}_i$ in (19) is equal to the throughput of link i (denoted by s_i) only if the uncorrupted bits in a packet can be recovered. This is called "partial packet recovery" which is useful in improving the throughput in this situation [15]. However, if the receiver does not have the capability, then the whole packet is discarded. In that case, $\bar{s}_i$ is an upper bound of s_i.

Although exactly computing the throughput without partial packet recovery seems not easy, one could use the approximated method in [13]. The key assumptions made there include that the transmission probabilities of the nodes are constant over time, and that the collisions caused by different hidden nodes are independent.

2) How bad is the hidden-node problem?: Since our eventual purpose is to achieve good throughput performance of CSMA/CA network, in this section we investigate whether it is more important to solve the hidden-node problem first or to maximize the throughput given the existence of hidden nodes. We consider a simple example where two links, link 1 and 2, are hidden from each other and assume that there is no partial packet recovery. Assume $p_1 = p_2 = p$, and the packet lengths are all T. Then, link 1's throughput without considering collisions is $x_1 := T/[T + (1/p - 1)]$, and link 1's throughput is $s_1 = x_1\beta$, where β is the probability that a certain packet transmitted by link 1 is not collided by link 2. β can be computed as follows:

$$\beta = [1 - x_2] \cdot (1 - p)^{T-1}$$

where $1 - x_2 = 1 - x_1$ is the probability that link 2 is not transmitting at the beginning (i.e., the first minislot's time) of the link-1 packet, and $(1 - p)^{T-1}$ is the (conditional) probability that link 2 does not start transmitting during the remaining $T - 1$ minislots' time of the link-1 packet. Fig. 13 plots s_1 as a function of $p_i = p, i = 1, 2$. The throughput is very sensitive to the choice of p, and the maximal per-link throughput is less than 0.14.

If the two links can hear each other, then the maximal per-link throughput, according to the throughput formula derived before, is about

$$\max_p \frac{p(1-p)T}{p^2T + 2p(1-p)T + (1-p)^2} \approx 0.455.$$

Also, the per-link throughput is above 0.4 for a wide range of p between 0.02 and 0.32. So, even if we choose the optimal transmission probability, the throughput with hidden nodes is only about 1/3 of the case without hidden nodes. Therefore, it is important to first solve the hidden-node problem before designing algorithms to optimize p_i or the transmission length T_i.

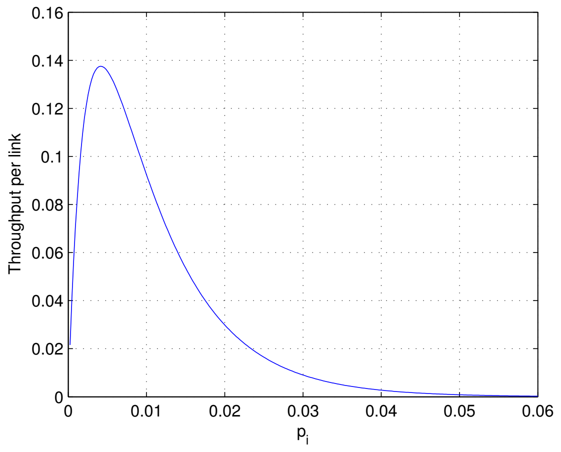


Fig. 13: Throughput of each link with hidden nodes

B. Explanation of Algorithm 2

To explain Algorithm 2, we show that it distributively solves the following convex optimization problem, where $u_{(x,z)}$ can be viewed as the probability of the detailed state (x, z):

$$\begin{aligned} \max_{\mathbf{u},\mathbf{f}} \quad & \{\sum_{(x,z)} -u_{(x,z)} \log(u_{(x,z)}) + \\ & \sum_{(x,z)} u_{(x,z)} \cdot [\log(g(x,z)) + 1] + \beta \cdot \sum_k v_k(f_k)\} \\ \text{s.t.} \quad & \sum_{(x,z): z_k = 1} u_{(x,z)} \geq f_k, \forall k \\ & u_{(x,z)} \geq 0, \sum_{(x,z)} u_{(x,z)} = 1. \end{aligned} \tag{20}$$

Since $\mathbf{u}$ is a probability distribution, the first term in the objective function,$\sum_{(x,z)} -u_{(x,z)} \log(u_{(x,z)})$ (the entropy of the distribution $\mathbf{u}$) is bounded. Also, the second term $\sum_{(x,z)} u_{(x,z)} \cdot [\log(g(x,z)) + 1] \in [\min_{(x,z)}\{\log(g(x,z)) + 1\}, \max_{(x,z)}\{\log(g(x,z)) + 1\}]$ is also bounded. So when β is large, the "importance" of the total utility dominates the objective function. This approximately achieves the maximal total utility.

Remark: Larger value of β leads to higher total utility but longer packets. Intuitively, the reason is similar to that mentioned in Theorem 3. Since long packets are not always preferable in practice, the weighting factor β can be used to control the tradeoff.

Proof: We will show that r_k can be viewed as a dual variable in problem (20); and Algorithm 2 can be viewed as a dual algorithm to solve problem (20).

A partial Lagrangian (with $\mathbf{y} \geq 0$ as the vector of dual variables) is

$$\begin{aligned}
& \mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{y}) \\
= & \sum_{(x,z)} -u_{(x,z)} \log(u_{(x,z)}) + \sum_{(x,z)} u_{(x,z)} \cdot [\log(g(x,z)) + 1] \\
& +\beta \cdot \sum_k v_k(f_k) + \sum_k y_k [\sum_{(x,z):z_k=1} u_{(x,z)} - f_k] \\
= & \sum_{(x,z)} \{u_{(x,z)}[-\log(u_{(x,z)}) + \log(g(x,z)) + 1 + \sum_{k:z_k=1} y_k]\} \\
& + \sum_k \{\beta \cdot v_k(f_k) - y_k f_k\}.
\end{aligned}$$

So

$$\frac{\partial L(\mathbf{u},\mathbf{f};\mathbf{y})}{\partial u_{(x,z)}} = -\log(u_{(x,z)}) + \log(g(x,z)) + \sum_{k:z_k=1} y_k.$$

If $u_{(x,z)} = p((x,z);\mathbf{y})$ (cf. equation (14)), then the partial derivative

$$\frac{\partial L(\mathbf{u},\mathbf{f};\mathbf{y})}{\partial u_{(x,z)}} = \log(E(\mathbf{y}))$$

which is the same for all state (x,z) (Given the dual variables $\mathbf{y}$, $\log(E(\mathbf{y}))$ is a constant). This means that $u_{(x,z)} = p((x,z);\mathbf{y}) > 0$ maximizes $\mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{y})$ over $\mathbf{u}$ subject to $u_{(x,z)} \geq 0, \sum_{(x,z)} u_{(x,z)} = 1$ (since it is impossible to increase $\mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{y})$ over $\mathbf{u}$ by slightly perturbing $\mathbf{u}$). Also, f_k as determined in part (2) maximizes $\mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{y})$ over $\mathbf{f}$. Therefore, by setting $r_k = y_k$, the resulting probabilities of all states (x,z), together with f_k as determined in part (2), maximizes $\mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{y})$. Denote $l(\mathbf{y}) = \max_{\mathbf{u},\mathbf{f}} \mathcal{L}(\mathbf{u},\mathbf{f};\mathbf{y})$, and $\mathbf{u}(y), \mathbf{f}(y)$ as the maximizers. Then the dual problem of (20) is $\min_{\mathbf{y}\geq 0} l(\mathbf{y})$

To find the optimal dual variable $r_k = y_k$, we change y_k in the reverse direction of the partial gradient $\partial l(\mathbf{y})/\partial y_k = \sum_{(x,z):z_k=1} u_{(x,z)}(\mathbf{y}) - f_k(\mathbf{y})$: if $f_k(\mathbf{y}) > \sum_{(x,z):z_k=1} u_{(x,z)}$, i.e., the arrival rate is larger than the service rate, then r_k should increase, and vice versa. This is part (2) of Algorithm 2

To sum up, r_k can be viewed as a dual variable in problem (20). Algorithm 2 can be viewed as a dual algorithm to solve problem (20) ■