

Approaching Throughput-Optimality in a Distributed CSMA Algorithm: Collisions and Stability

Libin Jiang and Jean Walrand
EECS Department, University of California at Berkeley
Berkeley, CA 94720
{ljiang,wlr}@eecs.berkeley.edu *

ABSTRACT

It was shown recently that CSMA (Carrier Sense Multiple Access)-like distributed algorithms can achieve the maximal throughput in wireless networks (and task processing networks) under certain assumptions [1]. One idealized assumption is that the sensing time is negligible, so that there is no collision. In this paper, we study more practical CSMA-based scheduling algorithms with collisions. First, we provide a model and give an explicit throughput formula, which has a simple product-form due to the quasi-reversibility structure of the model. Second, we show that the algorithm in [1] can be extended to approach throughput optimality in this case. Finally, sufficient conditions are given to ensure the convergence and stability of the proposed algorithm. Such conditions are also provided for the algorithm in [1] to achieve throughput-optimality. (Detailed proofs of all theorems here can be found in [2, 3].)

Categories and Subject Descriptors: C.2.1 [Computer-Communication Networks]: Network Architecture and Design-Distributed networks, Wireless communication

General Terms: Algorithms, Performance

Keywords: CSMA, Maximum Throughput, Distributed Scheduling, Markov Chain

1. THROUGHPUT-OPTIMALITY

In a wireless network, define a “link” as an (ordered) transmitter-receiver pair. Assume that there are K links. We say that two links *conflict* if they cannot transmit at the same time due to interference. (The conflict relationship is assumed to be symmetric.) Accordingly, define G as the conflict graph. Each vertex in G represents a link, and there is an edge between two vertices if the corresponding links conflict. (See Fig. 2 for an example.) Assume that G has N different independent sets (“IS”, not confined to “maximal independent sets”), where each IS is a set of links that can transmit simultaneously without conflict. Denote the i 'th IS by $v^i \in \{0, 1\}^K$, a 0-1 vector that indicates which links are transmitting in this IS. The k 'th element of v^i , $v_k^i = 1$ if link k is transmitting, and $v_k^i = 0$ otherwise.

We now describe the *scheduling problem* which is the focus of the paper. (But the results in this paper can be readily

*This work was supported by MURI grant BAA 07-036.18.

extended to a joint scheduling and congestion control problem as in [1].) WLOG, assume that the capacity of each link is 1. Traffic arrives at link k with an arrival rate $\lambda_k \in (0, 1)$. For simplicity, assume the following i.i.d. Bernoulli arrivals (although it can be easily generalized [3]): at time instances $0, 1, 2, \dots$, a unit-length packet arrives at link k with probability λ_k . Denote the vector of arrival rates as $\lambda \in \mathcal{R}_+^K$. We say that λ is *feasible* iff it can be written as $\lambda = \sum_{i=1}^N [\bar{p}_i \cdot v^i]$ where $\bar{p}_i \geq 0$ and $\sum_{i=1}^N \bar{p}_i = 1$. That is, there is a schedule of the independent sets (including the non-maximal ones) that can serve the arrivals. Denote the set of feasible λ by $\bar{\mathcal{C}}$. We say that λ is *strictly feasible* iff $\lambda \in \mathcal{C}$, where \mathcal{C} is the interior of $\bar{\mathcal{C}}$. A scheduling algorithm is said to be “throughput-optimal” if it can “support” any $\lambda \in \mathcal{C}$.

2. CSMA/CA-BASED SCHEDULING WITH COLLISIONS

2.1 Basic protocol

We describe the basic CSMA/CA protocol with fixed transmission probabilities (which suffices for our later development.) Let σ be the duration of each idle slot (or “minislot”). (In IEEE 802.11a, for example, $\sigma = 9\mu s$.) In the following we will simply use “slot” to refer to the minislot.

Assume that all links are saturated (i.e., always have packets to transmit). In each slot, if (the transmitter of) link i is not already transmitting and if the medium is idle, the transmitter of link i 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 start transmitting at the same slot, then a collision happens. (In this paper, we focus on networks without hidden nodes.)

Each link transmits a short probe packet with length γ (all “lengths” here are measured in the number of slots) before the data is transmitted (similar to the RTS/CTS mode in IEEE 802.11). This can avoid collisions of long data packets. When a collision happens, only the probe packets collide, so each collision lasts a length of γ . Assume that a successful transmission of link i lasts τ_i (which includes a constant overhead τ' and the data payload τ_i^p which is a random variable). Clearly $\tau_i \geq \tau'$. Let the mean of τ_i be T_i . (Overhead such as DIFS, etc, is included in γ and τ' .) Fig. 1 illustrates the timeline of a 3-link network where link 1, 2 conflict, and link 2, 3 conflict.

The above model possesses a quasi-reversibility property that will lead to a simple throughput formula. A process is “time-reversible” if the process and its time-reversed pro-

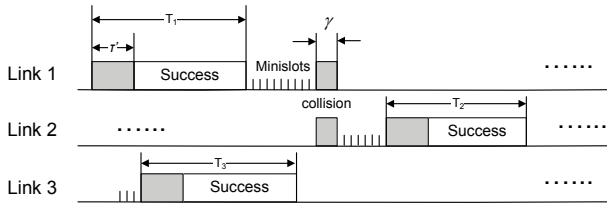


Figure 1: Timeline in the basic model (In this figure, $\tau_i = T_i, i = 1, 2, 3$ are constants.)

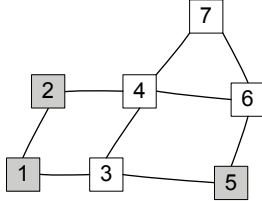


Figure 2: An example conflict graph (each square represents a link). In this on-off state x , links 1, 2, 5 active. So $S(x) = \{5\}$, $\phi(x) = \{1, 2\}$, $h(x) = 1$.

cess is statistically indistinguishable. Our model, in Fig. 1, reversed in time follows the same protocol as described above, except for the order of the overhead and the payload which are reversed. A key reason for this property is that the collisions start and finish at the same time.

2.2 Notation

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 link k is active (transmitting) in state x , and $x_k = 0$ otherwise. Then x is a vector indicating which links are active at a given slot. 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”) $C_m(x), m = 1, 2, \dots$. 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 $S(x) := \{k | k \in C_m(x) \text{ with } |C_m(x)| = 1\}$, and the set of links which are experiencing collisions as $\phi(x)$. Also, define the “collision number” $h(x)$ as the number of components in $G(x)$ with size larger than 1. Fig. 2 shows an example.

2.3 Throughput computation

By properly define the “state”, the above CSMA/CA protocol defines a discrete time Markov chain, whose stationary distribution is found in [2] utilizing quasi-reversibility. Then, the probability of any on-off state x can be computed:

THEOREM 1. *In the stationary distribution, the probability of $x \in \{0, 1\}^K$ is*

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

where T_i is the mean transmission length of link k , and E is a normalizing term such that $\sum_x p(x) = 1$.

Now we re-parametrize T_k by a variable r_k . Let $T_k := \tau' + T_0 \cdot \exp(r_k)$, where τ' is the overhead of a successful transmission (including RTS, CTS, ACK packets, DIFS, etc), and $T_k^p := T_0 \cdot \exp(r_k)$ is the mean length of the payload. $T_0 > 0$ is a constant “reference payload length”. Let \mathbf{r} be the vector of r_k ’s. By Theorem 1, the probability of x (with a given \mathbf{r}) is

$$p(x; \mathbf{r}) = \frac{1}{E(\mathbf{r})} g(x) \cdot \prod_{k \in S(x)} (\tau' + T_0 \cdot \exp(r_k)) \quad (2)$$

where $g(x) = \gamma^{h(x)} \prod_{i=1}^K p_i^{x_i} q_i^{1-x_i}$ is not related to \mathbf{r} , and the normalizing term

$$E(\mathbf{r}) = \sum_{x'} [g(x') \cdot \prod_{k \in S(x')} (\tau' + T_0 \cdot \exp(r_k))]. \quad (3)$$

Then, the probability that link k is transmitting payload in a given slot is

$$s_k(\mathbf{r}) = \frac{T_0 \cdot \exp(r_k)}{\tau' + T_0 \cdot \exp(r_k)} \sum_{x: k \in S(x)} p(x; \mathbf{r}) \quad (4)$$

Recall that the capacity of each link is 1. So $s_k \in [0, 1]$ is also the *service rate* of link k .

3. A DISTRIBUTED ALGORITHM TO APPROACH THROUGHPUT-OPTIMALITY

The following theorem [2] states that any $\lambda \in \mathcal{C}$ can be supported by properly choosing the mean payload lengths $T_k^p = T_0 \exp(r_k), \forall k$.

THEOREM 2. *Assume that $\gamma, \tau' > 0$, and transmission probabilities $p_k \in (0, 1), \forall k$ are fixed. Given any $\lambda \in \mathcal{C}$, there exists a unique $\mathbf{r}^* \in \mathcal{R}^K$ such that the service rate of link k is equal to the arrival rate for all k :*

$$s_k(\mathbf{r}^*) = \lambda_k, \forall k. \quad (5)$$

And \mathbf{r}^* is the solution of the convex optimization problem

$$\max_{\mathbf{r}} L(\mathbf{r}; \lambda) \quad (6)$$

where $L(\mathbf{r}; \lambda) = \sum_k (\lambda_k r_k) - \log(E(\mathbf{r}))$. This is because $\partial L(\mathbf{r}; \lambda) / \partial r_k = \lambda_k - s_k(\mathbf{r}), \forall k$.

The following fully distributed algorithm [2] is motivated by a gradient algorithm to solve problem (6).

Algorithm 1. Transmission-length control Algorithm

The vector \mathbf{r} is updated at time $t_j, j = 1, 2, \dots$. Let $t_0 = 0$ and $t_j - t_{j-1} = T(j), j = 1, 2, \dots$. Here, we use a constant $T(j) = T, \forall j$. Let “period j ” be the time between t_{j-1} and t_j , and $\mathbf{r}(j)$ be the value of \mathbf{r} set at time t_j . Initially, link k sets $r_k(0) \in [r_{min}, r_{max}]$ where r_{min}, r_{max} are two parameters. Then at time $t_j, j = 1, 2, \dots$, each link k updates

$$r_k(j) = r_k(j-1) + \alpha(j) [\lambda'_k(j) + \epsilon - s'_k(j) + g(r_k(j-1))] \quad (7)$$

where $\alpha(j) > 0$ is the step size, $\lambda'_k(j), s'_k(j)$ are the empirical average arrival rate and service rate¹ at link k in period j

¹We let link k send dummy packets when the queue is empty (so each link is saturated). This ensures that the CSMA Markov chain has the desired stationary distribution. The transmitted dummy packets are also included when $s'_k(j)$ is computed.

(i.e., the actual amount of arrivals and service in period j divided by $T(j)$.) Note that $\lambda'_k(j), s'_k(j)$ are random variables which are generally not equal to λ_k and $s_k(\mathbf{r}(j-1))$. With the small constant $\epsilon > 0$, the algorithm “pretends” to serve the arrival rate $\lambda + \epsilon \cdot \mathbf{1}$ which is slightly higher than the actual λ . Define $g(y) = 0$ for $y \in [r_{min}, r_{max}]$, $g(y) = r_{min} - y$ for $y < r_{min}$, and $g(y) = r_{max} - y$ for $y > r_{max}$. It is a “penalty function” to keep \mathbf{r} bounded [2].

Algorithm 1 says that when $r_k \in [r_{min}, r_{max}]$, if the empirical arrival rate of link k (plus ϵ) is larger than the service rate, then link k transmits more aggressively by using a larger mean transmission length, and vice versa.

THEOREM 3. Define the “capacity region”

$$\begin{aligned} \mathcal{C}(r_{min}, r_{max}, \epsilon) &:= \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C}, \text{ and} \\ \mathbf{r}^* &:= \arg \max_{\mathbf{r}} L(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1}) \in (r_{min}, r_{max})^K\}. \end{aligned}$$

If $\lambda \in \mathcal{C}(r_{min}, r_{max}, \epsilon)$, then

(i) If $\alpha(j) > 0$ is non-increasing and satisfies $\sum_j \alpha(j) = \infty$, $\sum_j \alpha(j)^2 < \infty$ and $\alpha(1) \leq 1$, then the algorithm converges: $\mathbf{r}(j) \rightarrow \mathbf{r}^* = \arg \max_{\mathbf{r}} L(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1})$ as $j \rightarrow \infty$ with probability 1, where \mathbf{r}^* satisfies $s_k(\mathbf{r}^*) = \lambda_k + \epsilon > \lambda_k, \forall k$.

(ii) If $\alpha(j) = \alpha, \forall j$ (i.e., constant step size) where α is small enough, then all queues are stable.

We have $\mathcal{C}(r_{min}, r_{max}, \epsilon) \rightarrow \mathcal{C}$ as $r_{min} \rightarrow -\infty$, $r_{max} \rightarrow \infty$ and $\epsilon \rightarrow 0$, by Theorem 2. Therefore, one can choose $r_{min}, r_{max}, \epsilon$ to arbitrarily approach \mathcal{C} . Note that r_{max} also affects the maximal mean payload length T_k^p , so there is a tradeoff between the capacity region and the maximal T_k^p .

4. AN ALGORITHM IN IDEALIZED CSMA

The following algorithm was designed in [1, 3] for an idealized CSMA model without collisions [4]. (Notice its similarity to Algorithm 1.) For link k , the transmission time is exponentially distributed with mean 1, and the backoff time is exponentially distributed with mean $1/\exp(r_k)$. Like before, larger r_k means more aggressive transmissions. In the idealized CSMA [4], carrier-sensing is assumed to be instantaneous (without propagation delay), so collisions do not occur. Although this is an approximation for wireless networks, the model can be used in a general task processing problem (or a constrained queueing system) [3].

Algorithm 2: Adjusting r_k 's in the idealized CSMA

Recall that $t_0 = 0$ and $T(j) = t_j - t_{j-1}$. Initially, set $\mathbf{r}(0) = \mathbf{0}$. At time t_j where $j = 1, 2, \dots$, let

$$r_k(j) = [r_k(j-1) + \alpha(j) \cdot (\lambda'_k(j) + d(r_k(j-1)) - s'_k(j))]_D, \forall k \quad (8)$$

where $\alpha(j) > 0$ is the step size, and $[\cdot]_D$ means the projection to the set $D := [0, r_{max}]$ where $r_{max} > 0$. We allow $r_{max} = +\infty$, in which case $[\cdot]_D$ is the same as $[\cdot]_+$. If the function $d(\cdot) > 0$, then Algorithm 2 “pretends” to serve some arrival rates higher than the actual ones.

The following are some results in [3] on the convergence and stability of Algorithm 2. (Similar results apply to the joint scheduling and congestion control algorithm in [1].)

THEOREM 4. (i). Let $r_{max} = +\infty$ and $d(r_k(j-1)) = \min\{c/r_k(j-1), \bar{w}\}$ where $c, \bar{w} > 0$ are small constants. Choose decreasing $\{\alpha(j)\}$ and increasing $\{T(j)\}$ that satisfy the following conditions (with $\eta(j) := \sum_{i=1}^j \alpha(i)$ and

$f(j) := \exp\{(\frac{5}{2}K + 1) \cdot (\bar{\lambda} + \bar{w}) \cdot \eta(j)\}$ where $\bar{\lambda}$ is an upper-bound of $\lambda'_k(j), \forall k, j$)

$$\begin{aligned} \sum_j \alpha(j) &= \infty, \sum_j \alpha(j)^2 < \infty \\ \sum_{j=1}^{\infty} [\alpha(j+1) \cdot \eta(j)]^2 &< \infty \\ \sum_{j=1}^{\infty} [\alpha(j+1) \cdot \eta(j) \cdot f(j)/T(j+1)] &< \infty \end{aligned} \quad (9)$$

For example, $\alpha(j) = c_0 / [(a \cdot j + b + 1) \log(a \cdot j + b + 1)]$ and $T_i = a \cdot j + b$ (with constants $a, b, c_0 > 0$) satisfy (9).

Then, for any $\lambda \in \mathcal{C}$, $\mathbf{r}(j)$ converges with probability 1 to some \mathbf{r}^* which satisfies $s_k(\mathbf{r}^*) > \lambda_k, \forall k$. And all queues are “rate-stable” [3]. So the algorithm is throughput-optimal.

(ii) Let $r_{max} < +\infty$ and $d(r_k(j-1)) = \epsilon > 0, \forall j, k$. Assume that $\lambda \in \mathcal{C}'(r_{max}, \epsilon)$, where

$$\begin{aligned} \mathcal{C}'(r_{max}, \epsilon) &:= \{\lambda | \lambda + \epsilon \cdot \mathbf{1} \in \mathcal{C}, \text{ and} \\ &\arg \max_{\mathbf{r} \geq \mathbf{0}} F(\mathbf{r}; \lambda + \epsilon \cdot \mathbf{1}) \in [0, r_{max}]^K\} \end{aligned}$$

where $F(\mathbf{r}; \lambda) := \sum_{k=1}^K (\lambda_k r_k) - \log\{\sum_{i=1}^N \exp[\sum_{k=1}^K (v_k^i r_k)]\}$.

Then, there exist constants $\alpha(j) = \alpha, \forall j$ and $T(j) = T, \forall j$ such that all queues are stable.

Alternatively, with decreasing $\{\alpha(j)\}$ and increasing $\{T(j)\}$ satisfying $\sum_j \alpha(j) = \infty, \sum_j \alpha(j)^2 < \infty, \sum_{j=1}^{\infty} [\alpha(j)/T(j)] < \infty$ (e.g., $\alpha(j) = 1/j, T(j) = j$), $\mathbf{r}(j)$ converges with probability 1 to some \mathbf{r}^* with $s_k(\mathbf{r}^*) > \lambda_k, \forall k$, and all queues are “rate-stable”.

As $r_{max} \rightarrow \infty$ and $\epsilon \rightarrow 0$, $\mathcal{C}'(r_{max}, \epsilon) \rightarrow \mathcal{C}$. So the maximal throughput can be arbitrarily approximated in this case.

5. RELATED WORKS

In [5], Rajagopalan and Shah independently proposed a scheduling algorithm similar to that in [1] in the context of optical networks. In [6], Ni and Srikant developed an algorithm to deal with collisions which is different from [2]. In [7], Liu et al. carried out a convergence analysis of a utility maximization algorithm extended from [1].

6. REFERENCES

- [1] L. Jiang, J. Walrand, “A Distributed CSMA Algorithm for Throughput and Utility Maximization in Wireless Networks,” *the 46th Annual Allerton Conference on Communication, Control, and Computing*, Sep. 2008.
- [2] L. Jiang, J. Walrand, “Approaching throughput-optimality in a Distributed CSMA Algorithm with Contention Resolution,” Technical Report, UC Berkeley, Mar. 2009. URL: <http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-37.html>
- [3] L. Jiang, J. Walrand, “Convergence and Stability of a Distributed CSMA Algorithm for Maximal Network Throughput,” Technical Report, UC Berkeley, Mar. 2009. URL: <http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-43.html>
- [4] X. Wang, K. Kar, “Throughput Modeling and Fairness Issues in CSMA/CA Based Ad-Hoc Networks,” *Proceedings of IEEE Infocom 2005*, Miami, Mar. 2005.
- [5] S. Rajagopalan and D. Shah, “Distributed Algorithm and Reversible Network,” *Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, USA, Mar. 2008.
- [6] J. Ni, R. Srikant, “Distributed CSMA/CA algorithms for achieving maximum throughput in wireless networks,” in *Proc. of Information Theory and Applications Workshop*, Feb. 2009.
- [7] J. Liu, Y. Yi, A. Proutiere, M. Chiang, and H. V. Poor, “Convergence and Tradeoff of Utility-Optimal CSMA,” <http://arxiv.org/abs/0902.1996>.