High-Throughput Random Access Using Successive Interference Cancellation in a Tree Algorithm

Yingqun Yu, *Member, IEEE*, and Georgios B. Giannakis, *Fellow, IEEE*

***Abstract*—Random access is well motivated and has been widely applied when the network traffic is bursty and the expected throughput is not high. The main reason behind relatively low-throughput expectations is that collided packets are typically discarded. In this paper, we develop a novel protocol exploiting successive interference cancellation (SIC) in a tree algorithm (TA), where collided packets are reserved for reuse. Our SICTA protocol can achieve markedly higher maximum stable throughput relative to existing alternatives. Throughput performance is analyzed for general d-ary SICTA with both gated and window access. It is shown that the throughput for d-ary SICTA with gated access is about $(\ln d)/(d-1)$, and can reach 0.693 for $d = 2$. This represents a 40% increase over the renowned first-come-first-serve (FCFS) 0.487 tree algorithm. Delay performance is also analyzed for SICTA with gated access, and numerical results are provided.**

***Index Terms*—Collision resolution, Markov chain, maximum stable throughput, random access, tree protocol.**

I. Introduction

A multiple-access system is encountered frequently in both wireline and wireless communications, and entails multiple transmitters (nodes or terminals) sending data packets to a common receiver over a shared channel [28], [2]. Since multiple simultaneous transmissions can lead to erroneous reception, a multiple-access (or medium-access control (MAC)) protocol defining rules for orderly access to the physical shared medium is of paramount importance when it comes to efficient utilization and fair sharing of resources.

Depending on whether the channel resource allocation among users is static or dynamic, multiple-access protocols can be classified in two major categories: fixed and dynamic ones [3]. While fixed allocation schemes like standard time-division multiple access (TDMA), frequency-division multiple access (FDMA), and code-division multiple access (CDMA), are suitable for steady and relatively heavy traffic, dynamic allocation alternatives assign channel resources on demand and are thus more appropriate when the traffic is discontinuous and bursty. Under such conditions, dynamic allocations based on random-access schemes such as Aloha [1], carrier-sense multiple access (CSMA) [14], and tree (also know as splitting) algorithms e.g., [4], [31], exhibit excellent delay–throughput characteristics provided that the traffic is low.

Random-access schemes rely on retransmissions to resolve current collisions as well as minimize future collisions. Collided packets are typically discarded, which reduces the throughput considerably [2]. To recover packets from multiple collided packets and thus boost throughput, interesting recent so-called network-diversity multiple access (NDMA) protocols jointly exploit multiple slots for detection [29], [36], [6]. NDMA schemes do not discard collided packets but rely on proper retransmissions and signal separation principles to resolve collisions. However, they may suffer from channel-induced ill-conditioning, difficulties with determining the number of collided packets and relatively high computational complexity. There is clearly a need to develop high-throughput random-access protocols, which not only take advantage of multiple collided packets, through what we could call retransmission diversity, but also facilitate implementation.

In this paper, we develop such a protocol that we call SICTA, because it relies on successive interference cancellation (SIC) to take advantage of collided packets in a conventional tree algorithm (TA) [35]. Tree algorithms have well-documented advantages over Aloha and CSMA. While slotted Aloha suffers from instability problems and low throughput, contention TAs are provably stable and enjoy higher throughputs [4], [31]. CSMA can improve throughput at the cost of hardware and software overhead for sensing, but applies only when the ratio of propagation delay to packet transmission time is relatively small and the stations can monitor the transmission channel, which is not the case for satellite communications, broadband hybrid fiber coaxial (HFC) networks, and fixed wireless systems. Among several improvements to the original TA, the so-called first-come-first-serve (FCFS) protocol [32], [26], [9], [21] exhibits a maximum stable throughput 0.487, which represents a 32% increase over that of stabilized slotted Aloha; see also [19] for a historical review of TAs. In recent years, with the interest for HFC networks and wireless broadband access networks growing, TAs have received revived attention by several standards, including DAVIC/DVB [7], DOCSIS [18], and have been adopted by IEEE 802.14 [13], [12]; see also [33] for throughput improvements under heavy traffic and [27] for energy-efficient collision resolution schemes.

Interference cancellation, on the other hand, has been used widely at the physical layer to cancel intersymbol interference in a decision feedback equalizer (DFE) [24], to separate spatially multiplexed streams of data in multiple-antenna systems

Manuscript received February 9, 2005; revised November 4, 2006. This work was prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The material in this paper was presented in part at IEEE INFOCOM'05, Miami, FL, March 2005.

The authors are with the Department of Electrical and Computer Engineering, 4-174 EE/CSci Building, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: yingqun@umn.edu; georgios@ece.umn.edu).

Communicated by G. Sasaki, Associate Editor for Communication Networks.
Color versions of Figures 6, 7, and 9 in this paper are available online at http://ieeexplore.ieee.org.
Digital Object Identifier 10.1109/TIT.2007.909080

[8], [11] and even single-antenna multiuser detection systems [34]. Further, from an information-theoretic viewpoint, successive decoding with interference cancellation is known to achieve all the points inside the Shannon capacity region of the multiple access channel [5].

With a cross-layer design approach, SICTA combines SIC with TAs, and thus permeates SIC benefits to the MAC layer [35]. We will analyze SICTA with both gated access and window access, and evaluate its performance using tools similar to those developed for a standard tree algorithm (STA) and Massey's modified tree algorithm (MTA) [17], [25], [20], [19]. A simple and interesting relation between the conditional mean collision resolution interval (CRI) lengths for STA and SICTA, will reveal that the maximum stable throughput for gated SICTA is around $(\ln d)/(d-1)$, while the maximum stable throughput for gated STA is around $(\ln d)/d$. Although a ternary tree $(d=3)$ maximizes throughput in STA, binary splitting $(d=2)$ for SICTA achieves the highest throughput, 0.693. Comparisons between gated access and window access will favor gated access for SICTA. Delay performance will also be analyzed for binary SICTA with gated access (generalizations to d-ary SICTA with $d>2$ follow along known lines).

The rest of the paper is organized as follows. Section II lays out modeling assumptions and TA basics. Section III describes SICTA in detail, while Section IV analyzes CRI statistics and derives the throughput. Section V deals with delay analysis and methods to compute the average packet delay. Numerical results are given in Section V-C, and conclusions are summarized in Section VI.

II. Modeling Preliminaries

After detailing the channel model and channel access algorithm (CAA), in this section we will outline the two basic conflict resolution algorithms (CRAs), namely STA and MTA, to lay out the context of SICTA—our novel protocol we develop in the ensuing section.

We assume the following modeling conditions typically adopted by random-access protocols [2].

- a1) *Infinite Population*: Infinite number of independent users are transmitting to a common receiver packets of length equal to one time unit (slot), over a slotted-time channel.
- a2) *Poisson Arrivals*: The packet arrival process is Poisson distributed with overall rate λ, and each packet arrives to a "new" user that has never been assigned a packet before.
- a3) *Collision or Perfect Reception*: If only one user sends a packet during a certain slot, it is received with no errors. However, if more users send packets over the same slot, then collision occurs and no packet information can be extracted from this single collision. In STA and MTA, collided packets are discarded, while in SICTA they are saved for future reuse.
- a4) *Immediate $0/k/e$ Feedback*: By the end of each slot, users are informed of the feedback from the receiver immediately and errorlessly. The feedback is one of
  - a) idle (0): when no packet transmission is taking place;
  - b) number of slots (k): representing the number of decoded packets plus the number of slots identified as being left idle—a notion that will become clear later; or
  - c) erroneous reception due to conflict (e): when no packet reception is successful.

For MTA, the feedback is ternary $0/1/e$, since only one $(k=1)$ successful packet reception is possible per time slot. STA, on the other hand, entails only binary feedback: collision/non-collision; that is, STA does not differentiate between 1 and 0. Ternary $0/k/e$ feedback is required by SICTA as will be explained later.

A. Channel Access Algorithms

In general, a random-access protocol consists of two parts, CRA and CAA. The former refers to the algorithm that resolves a conflict after it arises. We will discuss CRAs including STA, MTA, and SICTA in Sections III –V. Here we concentrate on CAA, which specifies when new packets may join a CRA. There are three basic CAAs [19].

1) *Gated Access:* This is the first CAA proposed by Capetanakis [4] and Tsybakov–Mikhailov [31]. According to this algorithm, new packets are transmitted in the first available slot after previous conflicts have been resolved. The time interval from the slot where an initial collision occurs up to and including the slot in which all senders recognize that all packets involved in this collision have been successfully received, is called a collision resolution interval (CRI). New packets that arrive during the current CRI are buffered and transmitted in the next CRI.

2) *Window Access:* After the current CRI finishes, only packets that arrive in a specific time period—in this case, a window on the arrival-time axis—are allowed to join the new CRI. If we let Δ denote the maximum window size, and τ the time elapsed from the end of the current window until the end of the CRI it generates, then the next window size is $\min\{\Delta,\tau\}$. This CAA is known as window algorithm (WA). One variant called simplified window algorithm (SWA) maintains a constant window size Δ even when $\tau<\Delta$; transmitters simply delay the start of the next CRI until a full window of size Δ is formed. WA and SWA have identical throughputs, while SWA exhibits higher delay than WA [19].

3) *Free Access:* New packets are transmitted immediately at the beginning of the next slot following their arrival.

Gated access and window access are commonly referred to as blocked access and will be the focus of this paper. Free access has advantages over blocked access because it entails limited

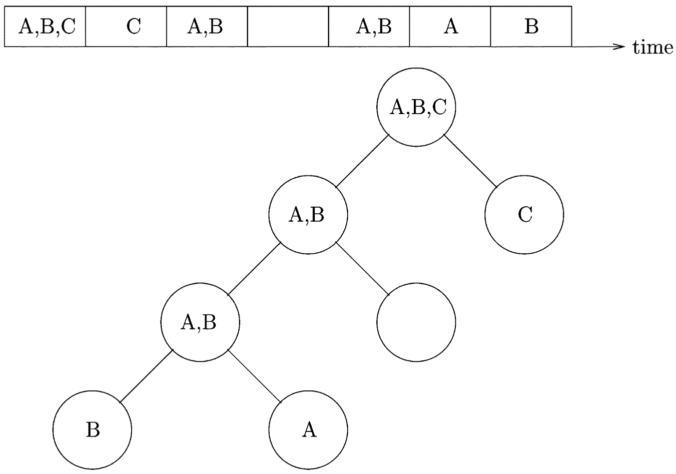

Fig. 1. An STA example.

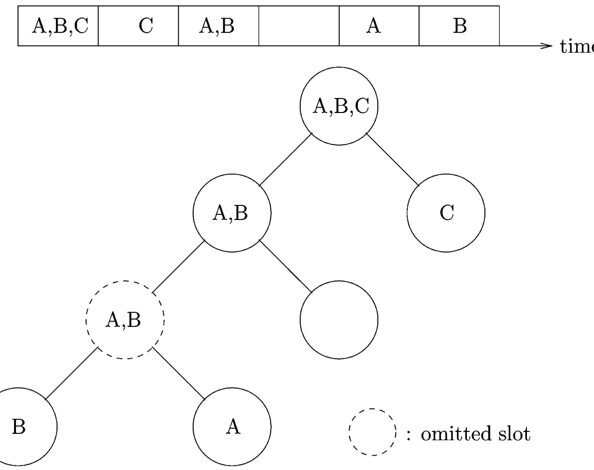

Fig. 2. An MTA example: a collision followed by an idle slot indicates that the dashed subset contains two or more packets.

sensing; that is, transmitters do not need to monitor the entire channel history and can drop in and out at any time. However, free access incurs performance loss and renders our SIC approach more complex to implement. For this reason, it will not be considered in this paper.

B. STA and MTA

To appreciate differences and similarities with our approach, let us review briefly the two conventional CRAs, namely, STA and MTA. For simplicity, we use a binary tree as an example. Consider the end time of the first transmission in a CRI. If the receiver feeds back idle or success, the CRI ends at this point. However, if the receiver feeds back collision, each user tosses a two-sided coin and joins either the first (right) subset with probability p, or, the second (left) subset with probability $1-p$. The initial collision is resolved when both of these subsets are resolved with the right subset being always deliberated first. This procedure continues recursively until all packets are received successfully. The best way to visualize STA is through a binary tree with the root being the first slot of the CRI. The rooted binary tree structure in Fig. 1 represents a particular pattern of idles, successes, and collisions resulting from such a sequence of binary splitting steps. In this Example, 7 slots are required to resolve three initially collided packets A, B, and C. Note that in our tree examples of STA and MTA (including SICTA in Section III), nodes are processed using the depth-first approach, which is the most widely used scan order for contention tree algorithms.

Let l'_n be the CRI length given that n packets initially collide. For STA, we have

$$l'_n = 1 + l'_i + l'_{n-i}, \qquad n \geq 2 \tag{1}$$

where i denotes the number of users joining the right subset. It is known that fair splitting with $p = 1/2$ is optimal and results in 0.347 throughput [16], [17], [25]. Notice that in Fig. 1, the collision in slot 3 is followed by an idle in slot 4, generating a deterministic collision in slot 5, since the two collided packets involved in slot 3 are assigned to the second subset. An improvement results if we do not transmit over this second subset and proceed directly to the next level of the binary tree. This algorithm is known as MTA [16], [25]. As one can verify from Fig. 2, only six slots are required to resolve the collision for the same tree structure of Fig. 1. Using the same definition of l'_n as in (1), we have

$$l'_n = \begin{cases} 1 + l'_i + l'_{n-i}, & \text{if } 1 \leq i \leq n \\ l'_i + l'_{n-i} (= 1 + l'_n), & \text{if } i = 0 \end{cases}, \quad n \geq 2. \tag{2}$$

Relative to (1), the slot for transmitting the left subset is omitted when the right subset is empty $(i = 0)$.

This simple modification increases throughput from 0.347 to 0.375 [16], [17], [25]. However, for binary MTA a biased splitting with $p = 0.582$ is optimal, and results in a throughput of 0.381 [16], [17], [25].

III. The Novel SICTA Protocol

Collided packets are discarded in STA and MTA with no attempt to extract pertinent packet information. In contrast, SICTA retains them for future reuse. It exploits the structure of a conventional TA and employs SIC to resolve collisions. In STA, all packets are resolved in an orderly one-by-one fashion according to the underlying tree structure. This nice property lends itself naturally to SIC, provided that collided packets are not discarded. To illustrate the basic idea, consider the motivating example of a binary tree depicted in Fig. 3. Let $\boldsymbol{y}_t$ denote the received signal vector at the end of slot t. From the received signal vector at the second slot, $\boldsymbol{y}_2$, the receiver decodes the packet A, and recovers the transmitted signal vector $\boldsymbol{x}_{\mathrm{A}}$ corresponding to the packet A. Subsequently, the interference from packet A to packet B in the first slot is canceled to obtain

$$\widetilde{\boldsymbol{y}}_1 = \boldsymbol{y}_1 - \boldsymbol{y}_{\mathrm{A}}. \tag{3}$$

Based on $\widetilde{\boldsymbol{y}}_1$, the packet B can be recovered as well. Notice that unlike STA which requires three slots to resolve this collision, only two slots suffice here. As also can be seen from this

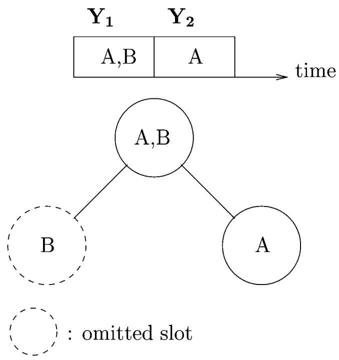

Fig. 3. A motivating SICTA example.

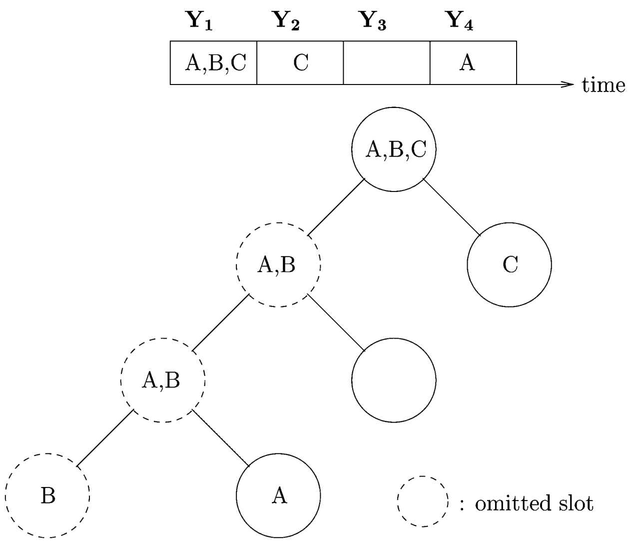

Fig. 4. SICTA example 2: after decoding A, the receiver recovers B, and feeds back $k = 2$.

example, SICTA requires neither estimation of the number of users involved in a collision nor complex signal separation techniques, which are critically needed for NDMA [29], [36]. Only simple interference cancellation suffices.

The simple example in Fig. 3 has illustrated how SIC can reduce the number of transmission slots, which clearly offers the potential to improve throughput. A key observation which extends to more complicated scenarios is that the first slot in the left subtree can be omitted. The reason is that after the receiver decodes all packets in the right subtree, it cancels them from the signal received during the root slot. Therefore, we have the following equation for the conditional CRI length:

$$l_n = 1 + l_i + l_{n-i} - 1 = l_i + l_{n-i}, \qquad n \geq 2. \tag{4}$$

Compared with (2), (4) holds for an arbitrary i, not just for $i = 0$.

A more complicated example is shown in Fig. 4, which uses the same tree structure as Fig. 1, but three slots of transmission are now omitted. Since only packet C is transmitted in the second slot, it is successfully decoded from $\boldsymbol{y}_2$ and is subtracted from $\boldsymbol{y}_1$ to obtain: $\widetilde{\boldsymbol{y}}_1 = \boldsymbol{y}_1 - \boldsymbol{y}_{\mathrm{C}}$. The signal vector $\widetilde{\boldsymbol{y}}_1$ contains information about packets A and B, but the receiver can recover none of the two from their superposition; hence, it feeds back only one success $(k = 1)$ to users. It is clearly unnecessary for

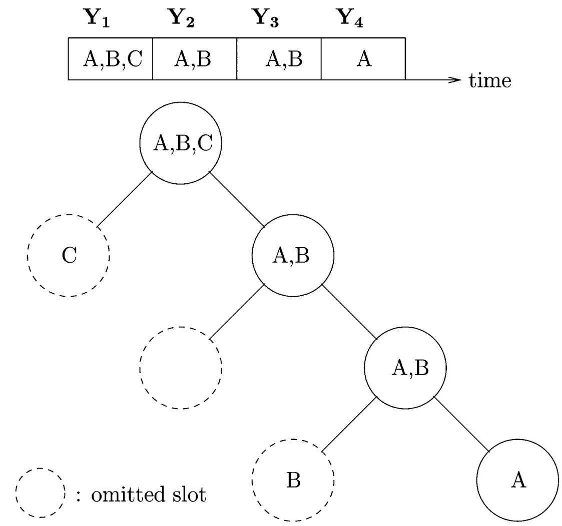

Fig. 5. SICTA example 3: after decoding A, the receiver recovers B and C, identifies a left-idle slot, and feeds back $k = 4$.

A and B to transmit in the third slot, because the receiver already knows $\widetilde{\boldsymbol{y}}_1$. Therefore, SICTA proceeds directly to the next level of the tree and nothing is transmitted in slot 3 according to the given binary tree. Similar to MTA, a collision slot followed by an idle slot means that the next slot can be skipped. Finally, after the receiver recovers $\boldsymbol{y}_{\mathrm{A}}$ at the end of slot 4, packet B can be recovered from $\widetilde{\widetilde{\boldsymbol{y}}}_1 = \widetilde{\boldsymbol{y}}_1 - \boldsymbol{x}_{\mathrm{A}} = \boldsymbol{y}_1 - \boldsymbol{x}_{\mathrm{C}} - \boldsymbol{x}_{\mathrm{A}}$, and the receiver feeds back $k = 2$. We notice that when the feedback is idle, SICTA operates similar to MTA.

The third example is depicted in Fig. 5. After decoding packet A at the end of the fourth slot, B and C can be recovered. The receiver identifies an idle left leaf as shown in the tree by subtracting $\boldsymbol{x}_{\mathrm{A}}$ and $\boldsymbol{x}_{\mathrm{B}}$ from $\boldsymbol{x}_2$ and checking the energy of $\boldsymbol{y}_2 - \boldsymbol{x}_{\mathrm{A}} - \boldsymbol{x}_{\mathrm{B}}$. Here, we suppose that the received power of every packet is high enough compared with the noise power so that the receiver can correctly identify an idle slot. Since three packets plus one left idle slot are identified, the receiver feeds back $k = 4$. Now the meaning of the number of identified slots should be clear: it indicates the number of successfully decoded packets plus the number of left-slots identified as being idle.

Next, we present an algorithmic description of SICTA. Each user maintains two counters: a local counter D_t and a system counter S_t, both reset to 0 at the beginning of a new CRI. The local counter determines packet transmission times for users involved in the current CRI. The system counter records the boundary of each CRI, which dictates when new arrivals during the current CRI can access the channel. In gated access, this happens exactly after the current CRI ends.

At the beginning of each slot t, a user transmits a packet if and only if $D_t = 0$. The local counter value is updated depending on the receiver's feedback as follows.

i) If feedback $= e$ and $D_t > 0$, then

$$D_{t+1} = D_t + 1. \tag{5}$$

ii) If feedback $= e$ and $D_t = 0$, then

$$D_{t+1} = \begin{cases} 0, & \text{with probability } p \\ 1, & \text{with probability } 1 - p. \end{cases} \tag{6}$$

iii) If feedback $= 0$ and $D_t > 1$, then

$$D_{t+1} = D_t. \tag{7}$$

iv) If feedback $= 0$ and $D_t = 1$, then

$$D_{t+1} = \begin{cases} 0, & \text{with probability } p \\ 1, & \text{with probability } 1-p. \end{cases} \tag{8}$$

v) If feedback $= k (k \geq 1)$, then

$$D_{t+1} = D_t - (k-1). \tag{9}$$

If $D_{t+1} \leq 0$, the user infers that his/her packet has been successfully decoded by the receiver and quits the remaining collision resolution procedure. If $D_{t+1} = 1$, the user realizes that his/her packet has not been resolved by SIC and other competitors are present. In this case, D_{t+1} should be further randomized to enable splitting (otherwise, a doomed collision will happen in slot $t+1$):

$$D_{t+1} = \begin{cases} 0, & \text{with probability } p \\ 1, & \text{with probability } 1-p. \end{cases} \tag{10}$$

After the value of D_{t+1} has been updated as mentioned earlier, a user will defer transmission in slot $t+1$ if the counter value $D_{t+1} > 0$; otherwise, a retransmission will be initiated.

The system counter is employed by each user to determine the CRI boundaries, and is updated as follows: $S_{t+1} = S_t + 1$ if feedback $= e$, $S_{t+1} = S_t$ if feedback $= 0$, and $S_{t+1} = S_t - (k-1)$ if feedback $= k$. The current CRA ends when the system counter reaches zero.

So far, we have tacitly assumed that decoded packets involved in the SIC are error-free. In practice, cyclic redundancy check (CRC) codes can be readily used to detect decoding errors [28]. Once an error in a packet is identified by CRC, this packet will not be employed in the SIC step. Instead, the tree will be temporarily frozen and retransmission of the corresponding packet will be requested. The remaining tree resumes until a correct version of this packet is received. If an undetectable error happens, it may propagate through the SIC process and cause an incorrect splitting of the tree. After observing enough times of failure, the receiver will just discard the current tree and start a new one. Since undetectable errors are very rare events, their effect on throughput will be practically negligible. For this reason, we will henceforth assume that no error propagation occurs.

IV. CRI Length Statistics and Throughput

In this section, we will use standard techniques [17], [25], [20], [19] to compute CRI length statistics for d-ary SICTA. We first derive the conditional probability generating function (PGF) $Q_n(z)$ of the random variable l_n, the conditional CRI length given collision multiplicity n. This PGF can be used to recursively compute the first moment $L_n := \mathbb{E}\{l_n\}$, as well as higher order moments. To obtain a direct nonrecursive expression for L_n, we will compute the unconditional probability generating function (UPGF) $Q(x,z)$ of the CRI length and the corresponding unconditional mean CRI length $L(x)$. Once the direct form of L_n is found, a simple relationship between L_n and L'_n, the conditional mean CRI length of STA, can be immediately identified, which will be used further to derive the throughput of d-ary SICTA for both gated and window access.

A. PGF, UPGF, and Mean CRI Length

Generalizing (4) to the d-ary SICTA yields the following expression for the conditional CRI length:

$$l_n = \begin{cases} 1, & \text{if } n = 0,1 \\ \sum_{j=1}^{d} l_{I_j}, & \text{if } n \geq 2 \end{cases} \tag{11}$$

where I_j is the number of nodes that flipped the value j, $j \in \{1,\ldots,d\}$, and joined the jth subset. Upon defining the conditional PGF as

$$Q_n(z) := \sum_{k=0}^{\infty} \Pr\{l_n = k\} z^k = \mathbb{E}\{z^{l_n}\} \tag{12}$$

it follows readily that

$$Q_0(z) = Q_1(z) = z. \tag{13}$$

Taking the conditional expectation on the right-hand side (RHS) of (12) leads to

$$\begin{aligned} Q_n(z) &= \mathbb{E}\left\{\mathbb{E}\left\{z^{\sum_{j=1}^{d} l_{I_j}} | I_1 \ldots I_d\right\} | n\right\} \\ &= \sum_{i_1 \ldots i_d}^{n} \binom{n}{i_1 \ldots i_d} \prod_{j=1}^{d} p_j^{i_j} Q_{i_j}(z), \qquad n \geq 2 \end{aligned} \tag{14}$$

where $\sum_{i_1 \ldots i_d}^{n}$ denotes the sum over all possible combinations of $i_1 \ldots i_d$ such that

$$\sum_{j=1}^{d} i_j = n,$$
$$i_j \geq 0, \quad \binom{n}{i_1 \ldots i_d} := \text{multinomial coefficient} \tag{15}$$

and $p_j > 0$ is the probability of joining the jth subset with $\sum_{j=1}^{d} p_j = 1$.

Differentiating (14) with respect to z and setting $z = 1$ yields the recursion

$$L_n = \sum_{j=1}^{d} \sum_{i_j=0}^{n} \binom{n}{i_j} p_j^{i_j} (1-p_j)^{n-i_j} L_{i_j}, \qquad n \geq 2 \tag{16}$$

with initial conditions

$$L_0 = L_1 = 1. \tag{17}$$

Therefore, L_n can be computed using the iteration

$$L_n = \frac{\sum_{j=1}^{d} \sum_{i_j=0}^{n-1} \binom{n}{i_j} p_j^{i_j} (1-p_j)^{n-i_j} L_{i_j}}{1 - \sum_{j=1}^{d} p_j^n}, \qquad n \geq 2. \tag{18}$$

However, we are also interested in a closed-form expression for L_n, which will facilitate our throughput analysis. To this end, we assume that the initial collision size n is Poisson distributed with mean x. Upon defining the UPGF as

$$Q(x,z) := \sum_{n=0}^{\infty} Q_n(z) e^{-x} \frac{x^n}{n!} \tag{19}$$

and substituting (13) and (14) into (19), we have

$$Q(x,z)=\sum_{n=0}^{\infty}\sum_{i_1\dots i_d}^{n}\binom{n}{i_1\dots i_d}\prod_{j=1}^{d}p_j^{i_j}Q_{i_j}(z)e^{-x}\frac{x^n}{n!}+(z-z^d)(1+x)e^{-x}. \quad (20)$$

The first term on the RHS of (20) can be rewritten as

$$\sum_{i_1=0}^{\infty}Q_{i_1}(z)e^{-p_1x}\frac{(p_1x)^{i_1}}{i_1!}\times\cdots\times\sum_{i_{d-1}=0}^{\infty}Q_{i_{d-1}}(z)e^{-p_{d-1}x}\frac{(p_{d-1}x)^{i_{d-1}}}{i_{d-1}!}\times\sum_{n=\sum_{j=1}^{d-1}i_j}^{\infty}Q_{n-\sum_{j=1}^{d-1}i_j}(z)e^{-p_dx}\frac{(p_dx)^{n-\sum_{j=1}^{d-1}i_j}}{\left(n-\sum_{j=1}^{d-1}i_j\right)!}. \quad (21)$$

Substituting back to (20), we obtain a functional equation for $Q(x,z)$

$$Q(x,z)=\prod_{j=1}^{d}Q(p_jx,z)+(z-z^d)(1+x)e^{-x}. \quad (22)$$

Differentiating (22) on both sides and setting $z=1$ yields the unconditional mean CRI length $L(x)$ as the solution of

$$L(x)=\sum_{j=1}^{d}L(p_jx)-(d-1)(1+x)e^{-x}. \quad (23)$$

Both (22) and (23) can be solved using the power series technique. Let us first solve (23) and postpone (22) to Section V-A. Suppose that the power series representation for $L(x)$ is

$$L(x)=\sum_{n=0}^{\infty}\alpha_nx^n. \quad (24)$$

Differentiating (19) with respect to z and setting $z=1$ yields

$$L(x)=\frac{\partial Q(x,z)}{\partial z}|_{z=1}=e^{-x}\sum_{n=0}^{\infty}L_n\frac{x^n}{n!}. \quad (25)$$

It follows directly from (24) and (25) that

$$\sum_{n=0}^{\infty}L_n\frac{x^n}{n!}=e^x\sum_{n=0}^{\infty}\alpha_nx^n=\left(\sum_{n=0}^{\infty}\frac{x^n}{n!}\right)\left(\sum_{n=0}^{\infty}\alpha_nx^n\right). \quad (26)$$

Equating coefficients of x^n on both sides of (26), we arrive at

$$L_n=\sum_{i=0}^{n}\frac{n!}{(n-i)!}\alpha_i. \quad (27)$$

Equation (27) expresses L_n as a weighted sum of α_i, $i=0,\dots,n$. To find L_n, it suffices to find α_n, which can be done by solving the functional (23).

Substituting (24) into (23) and expanding e^{-x} in a power series, we have

$$\sum_{n=0}^{\infty}\alpha_nx^n=\sum_{j=1}^{d}\sum_{n=0}^{\infty}\alpha_np_j^nx^n-(d-1)\times\sum_{n=0}^{\infty}\frac{(-1)^nx^n(1+x)}{n!}. \quad (28)$$

Solving (28) for α_n, we obtain [1]

$$\alpha_0=1,\quad \alpha_1=0,$$
$$\alpha_i=\frac{(d-1)(i-1)}{1-\sum_{j=1}^{d}p_j^i}\cdot\frac{(-1)^i}{i!},\qquad i\ge 2. \quad (29)$$

Now we are ready to obtain a nonrecursive expression for L_n. Substituting (29) into (27) yields

$$L_n=1+\sum_{i=2}^{n}\binom{n}{i}\frac{(d-1)(i-1)(-1)^i}{1-\sum_{j=1}^{d}p_j^i},\qquad n\ge 2. \quad (30)$$

There is a simple connection between STA and SICTA. With $L'_n:=\mathbb{E}\{l'_n\}$, it is known from [17] that

$$L'_0=L'_1=1,$$
$$L'_n=1+\sum_{j=1}^{d}\sum_{i_j=0}^{n}\binom{n}{i_j}p_j^{i_j}(1-p_j)^{n-i_j}L'_{i_j},\qquad n\ge 2 \quad (31)$$

and a closed form of L'_n is

$$L'_n=1+\sum_{i=2}^{n}\binom{n}{i}\frac{d(i-1)(-1)^i}{1-\sum_{j=1}^{d}p_j^i},\qquad n\ge 2. \quad (32)$$

Comparing (30) with (32), we easily obtain a relationship between L_n and L'_n

$$(d-1)(L'_n-1)=d(L_n-1). \quad (33)$$

In fact, an alternative approach to computing L_n is to prove that first (33) holds true, and then compute L_n since L'_n is known. We can start from (16), (17), (31), and use induction to prove (33). As this is rather straightforward, the proof is omitted for brevity.

It is easy to show that each term in the summation on the RHS of (30) is minimized by setting $p_j=1/d$, $j=1,\dots,d$. This proves that similar to d-ary STA, fair splitting is also optimal for d-ary SICTA.

B. Maximum Stable Throughput for Gated Access

Before studying the stability of d-ary SICTA with gated access, let us first summarize the result for d-ary STA. For d-ary STA with gated access, L'_n/n satisfies [17]

$$\frac{L'_n}{n}=\frac{d}{-\sum_{j=1}^{d}p_j\ln(p_j)}+f_1(n)+O(n^{-1}) \quad (34)$$

where $f_1(n)$ is a fluctuating function of small amplitude, between 10^{-6} and 10^{-3}. Under the infinite population Poisson model, the d-ary STA is stable for $\lambda<-\sum_{j=1}^{d}p_j\ln(p_j)/d-\varepsilon$ and unstable for $\lambda>-\sum_{j=1}^{d}p_j\ln(p_j)/d+\varepsilon$, where ε is a very small positive number. The sum $-\sum_{j=1}^{d}p_j\ln(p_j)$ reaches its maximum value $\ln d$ when $p_j=1/d$, $j=1,\dots,d$. Therefore, the maximum stable throughput for d-ary STA is $\lambda_{max}\approx(\ln d)/d$, which is achieved by fair splitting. Since $(\ln d)/d$ is maximized for $d=3$, ternary STA achieves a

[1]Note that α_1 cannot be determined from (28). However, it is easy to see that $\alpha_1=0$ from (27), since $L_1=\alpha_0+\alpha_1$ and $L_1=1$.

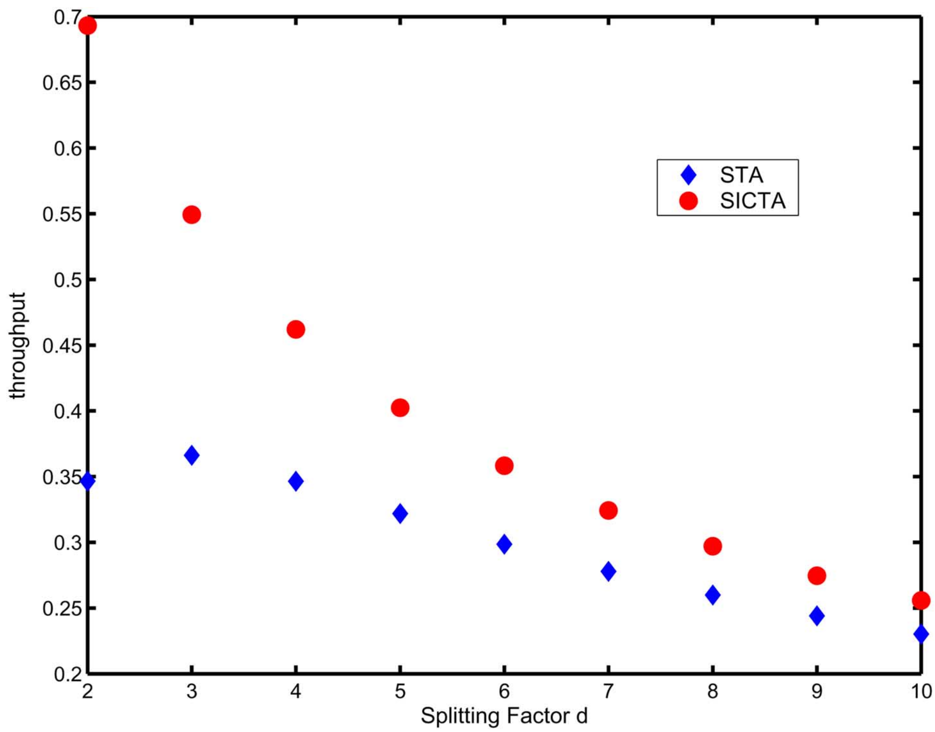

Fig. 6. Maximum throughput as a function of d for gated access SICTA.

maximum throughput of 0.3662, while the binary STA achieves throughput 0.3466.

Now let us return to SICTA. From (33) and (34), we have that [c.f. (34)]

$$\frac{L_n}{n} = \frac{d-1}{-\sum_{j=1}^{d} p_j \ln(p_j)} + \frac{d-1}{d} f_1(n) + O(n^{-1}). \quad (35)$$

Similarly, d-ary SICTA is stable for

$$\lambda < -\sum_{j=1}^{d} p_j \ln(p_j)/(d-1) - \varepsilon'$$

and unstable for

$$\lambda > -\sum_{j=1}^{d} p_j \ln(p_j)/(d-1) + \varepsilon'$$

where ε' is a very small positive number. Therefore, the maximum stable throughput for d-ary SICTA is $\lambda_{max} \approx (\ln d)/(d-1)$, which is also achieved by fair splitting. Unlike STA, binary SICTA achieves the highest throughput 0.6931, while ternary SICTA results in 0.5493. Also, notice that as $d \rightarrow \infty$, SICTA degenerates quickly to STA. Table I and Fig. 6 summarize the throughput values of the general d-ary STA and SICTA with gated access. Notice that although the maximum stable throughput is derived for the Poisson model, it holds under more general arrival processes, namely, stationary ergodic arrivals, which can be shown using Loynes' theorem [15].

The main advantages of SICTA are its high throughput and low complexity. The high 0.693 throughput renders SICTA suitable for medium to high traffic load, and widens the applicability of random access. Along with its relatively low implementation complexity, SICTA provides a promising alternative for random access in satellite communications, broadband HFCs, and fixed wireless systems. Indeed, the best conventional TA is the FCFS algorithm, which offers throughput 0.487. On the other hand, Tsybakov and Likhanov proved that all conventional random-access schemes, including tree algorithms, are unstable for throughput values higher than 0.568 [30]. SICTA not only outperforms FCFS over the slotted collision channel, but also exceeds the 0.568 Tsybakov–Likhanov bound, simply because SICTA takes advantage of information extracted from collided packets via SIC. Notice that the 0.568 upper bound does not apply here because SICTA capitalizes on extra information available from the physical layer which conventional TAs ignore.

TABLE I
MAXIMUM STABLE THROUGHPUT FOR d-ARY STA AND SICTA WITH GATED ACCESS

d	STA	SICTA
2	0.3466	<u>0.6931</u>
3	<u>0.3662</u>	0.5493
4	0.3466	0.4621
5	0.3219	0.4024
6	0.2986	0.3584
7	0.2780	0.3243
8	0.2599	0.2971
9	0.2441	0.2747
10	0.2303	0.2558

C. Maximum Stable Throughput for Window Access

Now let us turn our attention to window access. With SWA, the system can be viewed as a discrete-time queueing system in which packets that arrive during the window $(i\Delta, (i+1)\Delta)$ are served in the ith CRI. Intuitively speaking, the system is stable so long as the mean time to resolve a conflict arising in

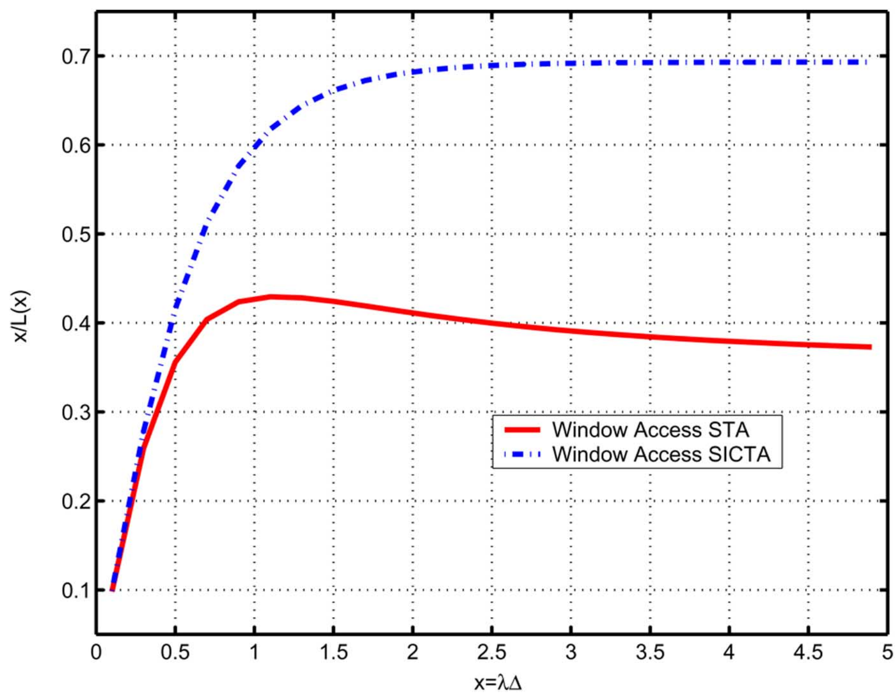

Fig. 7. Possible throughput for window access binary STA and SICTA.

the window interval Δ, namely, the unconditional mean CRI length $L(\lambda\Delta)$, is less than Δ; i.e.,

$$L(\lambda\Delta) < \Delta \tag{36}$$

which can be proved rigorously using Pakes' lemma [22]. Condition (36) can be rewritten as

$$\lambda < \frac{\lambda\Delta}{L(\lambda\Delta)} = \frac{x}{L(x)} \tag{37}$$

where x is the average number of packets that arrive during a time period Δ.

Fig. 7 plots the function $x/L(x)$ for both binary STA and SICTA with fair splitting. For binary STA, $x/L(x)$ is maximized at $x^* = 1.15$ and the maximum value is 0.429. As a result, the system is stable for $\lambda < 0.429$ if the maximum window size is chosen as $\Delta = x^*/\lambda$. The fact that the optimal $x^* = 1.15$ is consistent with the observation that binary STA resolves collisions among a small number of packets more efficiently than among a large number of packets, since n/L'_n (which can be regarded as the *CRA efficiency*) tends to drop as n increases; see also column 1 of Table II for binary STA. Therefore, window access performs better than gated access because it enables most CRIs to start with a small number of packets (close to 1). The famous 0.487 FCFS algorithm belongs to the window access family.

However, column 2 of Table II suggests that the contrary holds true with SICTA: it resolves collisions among a large number of packets more efficiently than among a small number of packets. This is also confirmed by Fig. 7, in which $x/L(x)$ increases to approximately 0.693 as x grows arbitrarily large. Therefore, $\Delta = \infty$ is optimal for window access SICTA, a case where window access becomes equivalent to gated access.

Actually, the fact that for SICTA gated access is better than window access is a plus since gated access is easier to implement than window access. In window access, each user has to record the arrival epoch of each packet to enable window-based splitting. Specifically for the FCFS algorithm, the window size has to be optimized for each arrival rate and the maximum throughput 0.487 is only guaranteed for Poisson arrivals. In comparison, gated access does not have these limitations, and can thus afford simple implementation in practice.

TABLE II
CRA EFFICIENCY AS A FUNCTION OF n FOR BINARY STA AND SICTA

n	n/L'_n	n/L_n
1	1.0000	1.0000
2	0.4000	0.6667
3	0.3913	0.6923
4	0.3801	0.6942
5	0.3726	0.6935
6	0.3678	0.6931
7	0.3646	0.6930
8	0.3622	0.6931
9	0.3604	0.6931

V. AVERAGE DELAY ANALYSIS

In this section, we study delay characteristics of binary SICTA with gated access, focusing on the average packet delay. The technique can also be used to obtain higher moments of delay as well, but is omitted for brevity. It is easy to see that the method also applies to d-ary SICTA for $d > 2$, but the computation becomes increasingly complex. Delay analysis for window access can be performed using the method described in [10], which is not discussed in this paper.

Our approach mimics the one used in [23], [20] to compute the packet delay for binary STA and MTA with gated access. Suppose that a randomly chosen "tagged" packet arrives during the kth CRI and leaves the system during the $(k+1)$th CRI. The total delay t experienced by this "tagged" packet is partitioned into the following two components as depicted in Fig. 8:

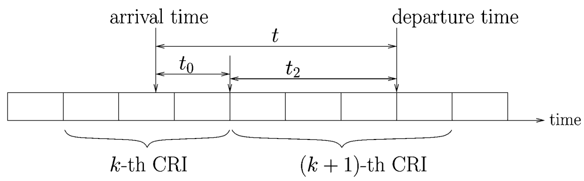

Fig. 8. Two Delay Components: t_0 and t_2.

t_0: the *initial delay* required before joining the $(k+1)$th CRI, which measures the residual life of the kth CRI at the moment of its arrival; and

t_2: the collision resolution time for the "tagged" packet to leave the system, which covers the rest of the time from the first transmission of the "tagged" packet until its resolution.

We begin by deriving the distribution of the CRI length in the steady state

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3 \ldots), \tag{38}$$

where π_j is the probability that a random chosen CRI is j slots long. Let c_k be the length of the kth CRI, $k = 0, \ldots, \infty$. For gated access SICTA, c_{k+1} depends only on the number of packets at the beginning of the $(k+1)$th CRI, denoted as s_{k+1}, which is exactly the number of new packets arriving during the kth CRI. Conditioned on the event that $c_k = i$, the variable s_{k+1} is Poisson distributed with mean λi due to Poisson arrivals

$$\Pr\{s_{k+1} = n | c_k = i\} = e^{-\lambda i} \frac{(\lambda i)^n}{n!}, \qquad n \geq 0. \tag{39}$$

It is clear that the CRI lengths $\{c_{k+1}, k = 0, 1, \ldots, \infty\}$ form a homogenous Markov chain. For this reason, the steady-state distribution can be found by solving the usual global balance equation

$$\boldsymbol{\pi} = \boldsymbol{\pi} \boldsymbol{P} \tag{40}$$

under the constraint

$$\sum_{j=1}^{\infty} \pi_j = 1 \tag{41}$$

where the infinite-size matrix $\boldsymbol{P}$ represents the transition probability matrix of the Markov chain and its (i,j)th entry $P_{i,j}$ is the probability that a CRI of length i is followed by a CRI of length j. We will derive $P_{i,j}$ by solving (22) later.

For now, we use the fact that we have Poisson packet arrivals and apply standard renewal-theoretic results to obtain $\widetilde{\pi}_n$, the probability that the "tagged" packet joins the system during a CRI of length n [20]

$$\widetilde{\pi}_n = \frac{n\pi_n}{\sum_{n=1}^{\infty} n\pi_n}. \tag{42}$$

If we condition on the event that the "tagged" packet arrives during a CRI of length n, then t_0 has a uniform distribution over the interval $[0, n)$, and the distribution of t_2 should be conditioned on the fact that the "tagged" packet will be competing with a group of other Poisson distributed packets with mean λn. In addition, the two conditional distributions are independent and the distribution of the sum $t = t_0 + t_2$ factors into the product of their marginal distributions. Therefore, the Laplace–Stieltjes transform of the packet delay can be written immediately in the form

$$D^*(s) = \sum_{n=0}^{\infty} \widetilde{\pi}_n \cdot U^*(n,s) \cdot G(\lambda n, e^{-s}) \tag{43}$$

where $U^*(n,s) = (1 - e^{-sn})/(sn)$ is the Laplace–Stieltjes transform of the uniform distribution over $[0, n)$ with mean $n/2$ and variance $n^2/12$; and $G(x,z)$ is the UPGF of t_2, given that the "tagged" packet is competing with Poisson traffic having mean x, as described later. Once we obtain $D^*(s)$, we can compute all the moments of t using standard techniques. In this paper, we are only interested in the mean of the packet delay

$$\overline{T} = \mathbb{E}\{t_0\} + \mathbb{E}\{t_2\} = \sum_{n=0}^{\infty} \widetilde{\pi}_n \cdot \left[\frac{n}{2} + T_{2|n}\right] \tag{44}$$

where $T_{2|n}$ is the conditional mean of t_2, the time spent in the next CRI given that the "tagged" packet joins a CRI with length n.

In the following, we delve further on $\boldsymbol{\pi}$ and t_2.

A. Steady-State Distribution of the CRI Length

To obtain the steady-state distribution of the CRI length, we need to solve (40) and (41). First we shall find $\boldsymbol{P}$, whose (i,j)th entry $P_{i,j}$ is the probability that a CRI of length i is followed by another CRI of length j. Since the length of the next CRI depends only on the number of initial contenders that arrive during the current CRI, it follows from (39) that

$$\begin{aligned} P_{i,j} &= \sum_{n=0}^{\infty} \Pr\{s_{k+1} = n | c_k = i\} \Pr\{l_n = j\} \\ &= \sum_{n=0}^{\infty} \Pr\{l_n = j\} e^{-\lambda i} \frac{(\lambda i)^n}{n!} = q_j(\lambda i) \end{aligned} \tag{45}$$

where the function $q_j(x)$ is defined as

$$q_j(x) := \sum_{n=0}^{\infty} \Pr\{l_n = j\} e^{-x} \frac{x^n}{n!}, \qquad j = 0, \ldots, \infty. \tag{46}$$

It turns out that the set $\{q_j(x)\}_{j=0}^{\infty}$ is closely related with the UPGF $Q(x,z)$ defined in (19). Substituting (12) into (19) and switching the summation order leads to

$$\begin{aligned} Q(x,z) &= \sum_{n=0}^{\infty} \left[\sum_{j=0}^{\infty} \Pr\{l_n = j\} z^j\right] e^{-x} \frac{x^n}{n!} \\ &= \sum_{j=0}^{\infty} \left[\sum_{n=0}^{\infty} \Pr\{l_n = j\} e^{-x} \frac{x^n}{n!}\right] z^j = \sum_{j=0}^{\infty} q_j(x) z^j. \end{aligned} \tag{47}$$

Therefore, $\{q_j(x)\}_{j=0}^{\infty}$ are the exact coefficients of the power series expansion of $Q(x,z)$ over z, which can be obtained by solving (22).

Substituting the power series representation (47) into (22) for the binary case with fair splitting ($d = 2$, $p_0 = p_1 = 1/2$), we have

$$\sum_{j=0}^{\infty} q_j(x) z^j = \left[\sum_{j=0}^{\infty} q_j\left(\frac{x}{2}\right) z^j\right]^2 + (z - z^2)(1+x)e^{-x} \quad (48)$$

which yields the solution after equating the coefficients of z^j on both sides

$$q_0(x) = 0, \quad q_1(x) = (1+x)e^{-x}, \quad q_2(x) = \frac{x^2}{4}e^{-x}$$
$$q_j(x) = \sum_{i=1}^{j-1} q_i\left(\frac{x}{2}\right) q_{j-i}\left(\frac{x}{2}\right), \qquad j \geq 3. \quad (49)$$

Equation (49) offers a recursive means to compute $q_j(x)$.

We are now ready to obtain $\boldsymbol{P}$ from (45). Subsequently, we can find the steady-state distribution of the CRI length from the balance equations (40) and (41). To this end, we need to truncate the infinite-dimensional $\boldsymbol{\pi}$ to a finite size before numerically solving the balance equations.

B. Collision Resolution Delay t_2

After obtaining the steady-state distribution, the next step is to compute the conditional mean of t_2, namely, $T_{2|n}$, given that the "tagged" packet joins a CRI of length n.

The collision resolution time t_2 is the duration that a "tagged" packet spends in the CRI from the first transmission of the "tagged" packet until it is successfully resolved. Let $t_{2,m}$ denote the time the "tagged" packet spends in the CRI, when there are m other packets competing with the "tagged" packet for the channel at the beginning. Due to Poisson arrivals, m should be Poisson distributed with mean λn when the "tagged" packet joins a CRI of length n; therefore, we have

$$\begin{aligned} T_{2|n} &= \sum_{m=0}^{\infty} \mathbb{E}\{t_{2,m}\} e^{-\lambda n} \frac{(\lambda n)^m}{m!} \\ &= \sum_{m=0}^{\infty}\sum_{k=1}^{\infty} k \Pr\{t_{2,m} = k\} e^{-\lambda n} \frac{(\lambda n)^m}{m!} = T_2(\lambda n) \end{aligned} \quad (50)$$

where the function $T_2(x)$ is defined as

$$T_2(x) := \sum_{m=0}^{\infty}\sum_{k=1}^{\infty} k \Pr\{t_{2,m} = k\} e^{-x} \frac{x^m}{m!}. \quad (51)$$

With the goal of computing $T_2(x)$, we notice that if the "tagged" packet joins the first (right) subset (sub-CRI) after the first collision, we have

$$t_{2,m} = 1 + t_{2,i}, \qquad m \geq 1 \quad (52)$$

where i refers to the number of other packets joining the right sub-CRI; if the "tagged packet" packet joins the second (left) subset (sub-CRI), we have

$$\begin{aligned} t_{2,m} &= 1 + l_i + t_{2,m-i} - 1 \\ &= l_i + t_{2,m-i}, \qquad m \geq 1. \end{aligned} \quad (53)$$

Let us define the PGF of $t_{2,m}$ as

$$G_{m+1}(z) := \sum_{k=0}^{\infty} \Pr\{t_{2,m} = k\} z^k \quad (54)$$

and

$$G_{m+1}^{(s)}(z) := \sum_{k=0}^{\infty} \Pr\{t_{2,m} = k | \text{it joins sub- CRI } s\} z^k, \qquad m \geq 1 \quad (55)$$

where s is 0 or 1, for the first and second sub-CRI, respectively. It is clear that $t_{2,0} = 1$, which implies that

$$G_1(z) = z. \quad (56)$$

It follows from (52) and (53) that

$$G_{m+1}^{(0)}(z) = z \sum_{i=0}^{m} B_{m,i} G_{i+1}(z), \qquad m \geq 1 \quad (57)$$

and

$$G_{m+1}^{(1)}(z) = \sum_{i=0}^{m} B_{m,i} Q_i(z) G_{m-i+1}(z), \qquad m \geq 1 \quad (58)$$

where

$$B_{n,i} = \binom{n}{i} p^i (1-p)^{n-i} \quad (59)$$

is the binomial probability of an $(i, n-i)$ split.

In fair splitting, the "tagged" packet joins both sub-CRIs with equal probability; thus, we have

$$\begin{aligned} G_{m+1}(z) &= \frac{1}{2} G_{m+1}^{(0)}(z) + \frac{1}{2} G_{m+1}^{(1)}(z) \\ &= \frac{1}{2} z \sum_{i=0}^{m} B_{m,i} G_{i+1}(z) \\ &\quad + \frac{1}{2} \sum_{i=0}^{m} B_{m,i} Q_i(z) G_{m-i+1}(z), \quad m \geq 1. \end{aligned} \quad (60)$$

Following the technique used in Section III, let us define

$$G(x, z) := \sum_{m=0}^{\infty} G_{m+1}(z) e^{-x} \frac{x^m}{m!}. \quad (61)$$

Substituting (60) into (61) yields

$$\begin{aligned} G(x,z) &= \sum_{m=0}^{\infty} \frac{1}{2} z \sum_{i=0}^{m} B_{m,i} G_{i+1}(z) e^{-x} \frac{x^m}{m!} \\ &\quad + \sum_{m=0}^{\infty} \frac{1}{2} \sum_{i=0}^{m} B_{m,i} Q_i(z) G_{m-i+1}(z) e^{-x} \frac{x^m}{m!} \\ &\quad + z e^{-x} - \frac{1}{2} z^2 e^{-x} - \frac{1}{2} z^2 e^{-x}. \end{aligned} \quad (62)$$

The first term in (62) can be rewritten as

$$\begin{aligned} &\frac{1}{2} z \sum_{i=0}^{\infty} e^{-\frac{x}{2}} \frac{(x/2)^i}{i!} \sum_{m=i}^{\infty} G_{i+1}(z) e^{-\frac{x}{2}} \frac{(x/2)^{m-i}}{(m-i)!} \\ &\qquad = \frac{1}{2} z G\left(\frac{x}{2}, z\right). \end{aligned} \quad (63)$$

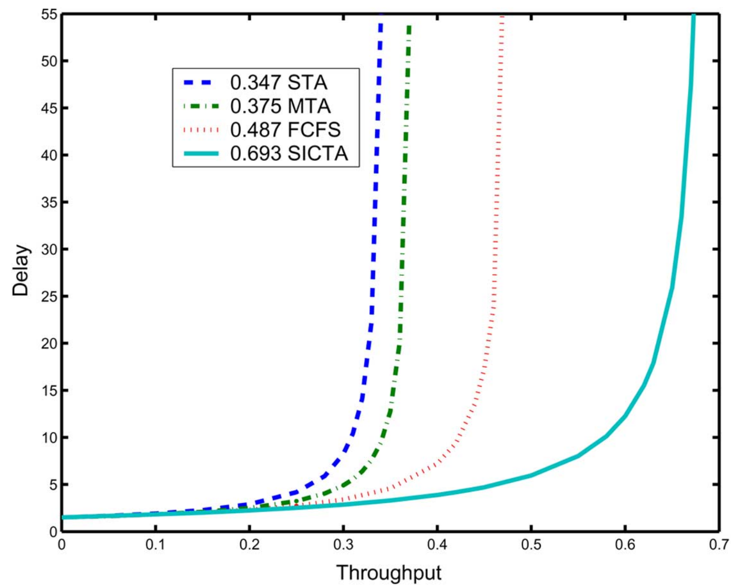

Fig. 9. Delay versus throughput.

Similarly, we rewrite the second term in (62) as

$$\frac{1}{2}\sum_{i=0}^{\infty} Q_i(z)e^{-\frac{x}{2}}\frac{(x/2)^i}{i!}\sum_{m=i}^{\infty} G_{m-i+1}(z)e^{-\frac{x}{2}}\frac{(x/2)^{m-i}}{(m-i)!} = \frac{1}{2}Q\left(\frac{x}{2},z\right)G\left(\frac{x}{2},z\right). \quad (64)$$

Combining (63) and (64), we arrive at

$$G(x,z) = \frac{1}{2}zG\left(\frac{x}{2},z\right) + \frac{1}{2}Q\left(\frac{x}{2},z\right)G\left(\frac{x}{2},z\right) + (z-z^2)e^{-x}. \quad (65)$$

Since $T_2(x) = \frac{\partial G(x,z)}{\partial z}|_{z=1}$, differentiating (65) and setting $z=1$ yields

$$T_2(x) = \frac{1}{2} + T_2\left(\frac{x}{2}\right) + \frac{1}{2}L\left(\frac{x}{2}\right) - e^{-x}. \quad (66)$$

Once again, we use the power series method to solve this functional equation. Letting

$$T_2(x) = \sum_{k=0}^{\infty}\theta_k x^k \quad (67)$$

and substituting it back into (66) yields the solution

$$\theta_0 = 1,$$
$$\theta_k = \frac{2^{-k}}{2(1-2^{-k})}\alpha_k - \frac{1}{1-2^{-k}}\cdot\frac{(-1)^k}{k!}, \quad k \geq 1 \quad (68)$$

where $\{\alpha_k\}_{k=0}^{\infty}$ are obtained from (29).

C. Numerical Results

We used the technique of [20] to numerically calculate the steady-state distribution of CRI lengths and the average delay. We omit the details but present the final results. Fig. 9 illustrates throughput versus average packet delay for binary STA, MTA, and SICTA all with gated access, and FCFS. Table III summarizes the calculated results, where the data for STA and MTA are from [20] and the data for FCFS are from [19]. It is clear that SICTA's delay curve stays always below others.

TABLE III
PACKET DELAY FOR STA, MTA, FCFS AND SICTA

λ	STA	MTA	λ	FCFS	λ	SICTA
0.10	1.909	1.825	0.10	1.81	0.1	1.8244
0.20	2.896	2.502	0.20	2.31	0.20	2.245
0.25	4.184	3.238	0.25	2.73	0.30	2.861
0.30	8.246	4.912	0.30	3.40	0.40	3.887
0.31	10.38	5.543	0.35	4.58	0.50	5.951
0.32	14.11	6.392	0.40	7.22	0.60	12.25
0.33	22.28	7.603	0.42	9.43	0.62	15.51
0.34	55.11	9.471	0.44	13.57	0.63	17.89
0.35	–	12.76	0.45	17.34	0.65	25.91
0.36	–	20.17	0.46	23.92	0.66	33.48
0.37	–	54.13	0.48	92.91	0.67	47.51

VI. CONCLUSION

As a physical layer tool, SIC has well appreciated merits for, e.g., multiuser detection. Here we have shown how in a cross-layer design, SIC benefits can be migrated to the MAC layer. Specifically, we derived a novel random access protocol, that we termed SICTA, since it is based on SIC to resolve collisions using a tree algorithm (TA). Its main advantage is high throughput, reaching 0.693—the highest among all contention tree algorithms to date. The reason behind this impressive improvement is the novel idea of having packet information extracted (via SIC) from collided packets and subsequently used in the tree structure. We also derived methods for calculating the CRI distribution in steady state, the mean CRI length, and the average packet delay. Although only the first moment was given, the technique can be extended to obtain higher order moments as well.

Our future work will address possible improvements of SICTA and explore its application in practice. Another interesting direction is to derive a (tight) upper bound on the throughput for the slotted channel where collided packets are not discarded, which will tell us how far SICTA is from a fundamental benchmark.

REFERENCES

[1] N. Abramson, "The Aloha system—Another alternative for computer communications," in *Proc. Fall Joint Comput. Conf.*, Apr. 1970, vol. 37, pp. 281–285.

[2] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[3] B. Bing, *Broadband Wireless Access*. Amsterdam, The Netherlands: Kluwer Academic, 2000.

[4] J. I. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inf. Theory*, vol. IT-25, no. 5, pp. 505–515, Sep. 1979.

[5] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[6] G. Dimic, N. D. Sidiropoulos, and L. Tassiulas, "Wireless networks with retransmission diversity access mechanisms: Stable throughput and delay properties," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2019–2030, Aug. 2003.

[7] ETSI, *Digital Video Broadcasting (DVB); DVB Interaction Channel for Cable-TV Distribution Systems (CATV),* 1998, prETS 300 800.

[8] G. J. Foschini, "Layered space-time architecture for wireless communications in a fading environment when using multiple antennas," *AT&T Bell Labs Tech. J.*, vol. 1, no. 2, pp. 41–59, 1996.

[9] R. Gallager, "A perspective on multiaccess channels," *IEEE Trans. Inf. Theory*, vol. IT-27, no. 2, pp. 124–142, Mar. 1985.

[10] L. Georgiades, L. F. Merakos, and P. Papantoni-Kazakos, "A method for the delay analysis of random multiple-access algorithms whose delay process is regenerative," *IEEE J. Select. Areas Commun.*, vol. 5, no. 6, pp. 1051–1062, Jul. 1987.

[11] G. D. Golden, G. J. Foschini, R. A. Valenzuela, and P. W. Wolniansky, "Detection algorithm and initial laboratory results using the V-BLAST space-time communications architecture," *Electron. Lett.*, vol. 35, no. 7, pp. 14–19, Jan. 1999.

[12] N. Golmie, Y. Saintillan, and D. H. Su, "A review of contention resolution algorithms for IEEE 802.14 networks," *IEEE Commun. Surve.*, First Quarter, 1999.

[13] *Cable-TV Access Method and Physical Layer Specification, Draft 3, Revison 3*, IEEE 802.14 WG, IEEE, 1998.

[14] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part I—Carrier sense multiple-access modes and their throughput delay characteristics," *IEEE Trans. Commun.*, vol. COM-23, no. 9, pp. 1400–1416, Sep. 1975.

[15] R. Loynes, "The stability of a queue with nonindependent inter-arrival and service times," *Proc. Camb. Philos. Soc.*, pp. 497–520, 1962.

[16] J. L. Massey, "Collision Resolution Algorithm and Random Access Communications," in *Multiuser Communication Systems (CISM Course and Lecture Notes)*, G. Longo, Ed. New York: Spriger-Verlag, 1981, vol. 265.

[17] P. Mathys and P. Flajolet, "Q -ary collision resolution algorithms in random-access systems with free or blocked channel access," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 2, pp. 217–243, Mar. 1985.

[18] MCNS, *Data-Over-Cable Service Interface Specifications, Radio Frequency Interface Specification*, Ref. SP-RFIv1.1-PI01-990226 1999.

[19] M. L. Molle and G. C. Polyzos, Conflict Resolution Algorithms and Their Performance Analysis Dept. Computer Sci. Eng., University of California, San Diego, La Jolla, CA, 1993, Tech. Rep. CS93-300.

[20] M. L. Molle and A. C. Shih, Computation of the Packet Delay in Massey's Standard and Modified Tree Conflict Resolution Algorithms With Gated Access Computer Syst. Res. Inst., Univ. Toronto, Toronto, ON, Canada, 1992, Tech. Rep. CSRI-264.

[21] J. Mosely and P. A. Humblet, "A class of efficient contention resolution algorithms for multiple access," *IEEE Trans. Commun.*, vol. COM-33, no. 2, pp. 145–151, Feb. 1985.

[22] A. G. Pakes, "Some conditions for ergodicity and recurrence of Markov chains," *Oper. Res.*, vol. 17, pp. 1058–1061, 1969.

[23] G. C. Polyzos, A Queueing Theoretic Approach to the Delay Analysis for a Class of Conflict Resolution Algorithms Computer Syst. Res. Inst., Univ. Toronto, Toronto, ON, Canada, 1989, Tech. Rep. CSRI-224.

[24] J. G. Proakis, *Digital Communications*, 4th ed. New York: McGraw-Hill, 2000.

[25] R. Rom and M. Sidi, *Multiple Access Protocols: Performance and Analysis*. New York: Springer-Verlag, 1990.

[26] G. Ruget, "Some Tools for the Study of Channel Sharing Algorithms," in *Multiuser Communication Systems (CISM Course and Lecture Notes)*, G. Longo, Ed. New York: Spriger-Verlag, 1981, vol. 265.

[27] Y. E. Sagduyu and A. Ephremides, "Energy-efficient collision resolution in wireless ad-hoc networks," in *Proc. INFOCOM Conf.*, San Francisco, CA, Mar./Apr. 2003, vol. 1, pp. 492–502.

[28] A. S. Tanenbaum, *Computer Networks*, 4th ed. Upper Saddle River, NJ: Prentice-Hall, 2003.

[29] M. K. Tsatsanis, R. Zhang, and S. Banerjee, "Network-assisted diversity for random access wireless networks," *IEEE Trans. Signal Process.*, vol. 48, no. 3, pp. 702–711, Mar. 2000.

[30] B. S. Tsybakov and N. Likhanov, "Upper bound on the capacity of a random multiple access system," (in Russian) *Probl. Pered. Inform.*, vol. 23, no. 3, pp. 246–236, Jul./Sep. 1987.

[31] B. S. Tsybakov and V. A. Mikhailov, "Free synchronous packet access in a broadcast channel with feedback," (in Russian) *Probl. Pered. Inform.*, vol. 14, no. 4, pp. 32–59, Oct./Dec. 1978.

[32] B. S. Tsybakov and V. A. Mikhailov, "Random multiple packet access: Part-and-try algorithm," (in Russian) *Probl. Pered. Inform.*, vol. 16, no. 4, pp. 65–79, Oct./Dec. 1980.

[33] M. X. van den Broek, I. B. J. F. Adan, N. S. Shankar, and S. C. Borst, "A novel mechanism for contention resolution in HFC networks," in *Proc. INFOCOM Conf.*, San Francisco, CA, Mar./Apr. 2003, vol. 2, pp. 979–989.

[34] S. Verdú, *Multiuser Detection*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[35] Y. Yu and G. B. Giannakis, "SICTA: A 0.693 contention tree algorithm using successive interference cancellation," in *Proc. INFOCOM Conf.*, Miami, FL, Mar. 2005, vol. 3, pp. 1908–1916.

[36] R. Zhang, N. D. Sidiropoulos, and M. K. Tsatsanis, "Collision resolution in packet radio networks using rotational invariance techniques," *IEEE Trans. Commun.*, vol. 50, no. 1, pp. 146–155, Jan. 2002