February 1993 UILU-ENG-93-2208

SCHEDULING NON-UNIFORM TRAFFIC
A Preliminary Report

T. Weller and B. Hajek

Coordinated Science Laboratory
College of Engineering
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-93-2208	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION National Science Foundation
6c. ADDRESS (City, State, and ZIP Code) 1308 W. Main St. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code) 1800 G. Street, N.W. Washington, DC 20552
8a. NAME OF FUNDING/SPONSORING ORGANIZATION National Science Foundation 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER NSF NCR 90-04355
8c. ADDRESS (City, State, and ZIP Code) 1800 G. Street, NW Washington, DC 20552	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

Scheduling Non-Uniform Traffic---A Preliminary Report

12. PERSONAL AUTHOR(S)
Weller, Timothy and Hajek, Bruce

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical	FROM ______ TO ______	February 1993	34

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	scheduling, multi-access communication, computer networks, non-uniform traffic

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Flexible non-uniform ("bursty") traffic models are important for the analysis of integrated data networks carrying diverse classes of data. In this report, a new model of non-uniform traffic is introduced for a single-hop multi-access communication system. Transmission algorithms for this non-uniform traffic are designed and analyzed with respect to parameters of interest such as throughput, packet delay, arrival traffic characteristics, and propagation delay. The two main sections of the paper consider the problems in scheduling this non-uniform traffic for systems with small propagation delay (e.g. a packet switch) and large propagation delay (e.g. a high-speed Asynchronous Transfer Mode network based on a passive optical star), respectively.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code) / 22c. OFFICE SYMBOL

DD Form 1473, JUN 86 Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

Coordinated Science Laboratory Technical Report, February 1993

Scheduling Non-Uniform Traffic — A Preliminary Report

Timothy Weller   Bruce Hajek
Coordinated Science Laboratory
University of Illinois, Urbana IL 61801

Abstract

Flexible non-uniform ("bursty") traffic models are important for the analysis of integrated data networks carrying diverse classes of data. In this report, a new model of non-uniform traffic is introduced for a single-hop multi-access communication system. Transmission algorithms for this non-uniform traffic are designed and analyzed with respect to parameters of interest such as throughput, packet delay, arrival traffic characteristics, and propagation delay. The two main sections of the paper consider the problems in scheduling this non-uniform traffic for systems with small propagation delay (e.g. a packet switch) and large propagation delay (e.g. a high-speed Asynchronous Transfer Mode network based on a passive optical star), respectively.

1 Introduction

Several trends in communication networks are currently being shaped by the need to carry integrated traffic. Traditionally, different networks carried distinct classes of data, but recently there has been demand for the multiplexing of many classes of data onto the same network. Also, data is increasingly transmitted over longer distances than in traditional local area networks (LANs). Finally, the quantity of data to be communicated is also rapidly increasing, creating a demand for more capacity and more flexibility within high-speed networks.

The recent and widely accepted Asynchronous Transfer Mode (ATM) standard [4] for integrated networks is illustrative of the solutions proposed to meet some of the demands described above. Four key features of ATM deserve a closer look since they differ from those features of most classical protocols. First, data is transmitted in small packets, primarily in a connection-oriented network. Second, because of high data rates and small packet sizes, the relative propagation delay expressed in terms of number of packets may be quite large, especially in some wide area networks (WANs). Third, since diverse classes of data can be transported via ATM, non-uniform ("bursty") arrival traffic must be accommodated. Finally, guaranteed quality of service (QOS) is required for some classes of data using ATM, such as real-time video or voice. These applications may have requirements for minimum throughput or maximum delay.

In this paper, a model of multi-access communication which can model these four key features (among others) is used. It is suitable for analysis of high-speed computer networks from WANs to LANs to packet switches, including ATM networks. Other applications of the model are possible. Packet size and propagation delay are parameters in the model.

Most previous related work (see [14] for an overview) has focused on uniform traffic patterns. The goal of this paper is to study methods for transmitting non-uniform traffic in the type of communication networks described above. Traditional approaches such as time-division multiplexing and pure random contention schemes are not well suited for non-uniform traffic. At least some adaptive reservation of bandwidth is generally required. In this paper, a new non-uniform traffic model is introduced—as a practical representative of the entire non-uniform traffic class—which facilitates determination of QOS. The challenge is to design and analyze transmission algorithms for this type of traffic and to demonstrate the applicability of the model.

Section 2 describes the basic model of a multi-access communication system which can be applied to several physical systems. Section 3 introduces (α, S) non-uniform traffic. Section 4 presents several transmission algorithms for (α, S) traffic when the propagation delay is negligible. Section 5 presents a transmission algorithm to be used when the propagation delay is very large. Finally, Section 6 discusses relaxing some restrictions of the basic model and some open problems.

2 The Basic Model of a Multi-Access Communication System

2.1 Overview of the Model

The *basic model* of a multi-access communication system considered in this paper has many stations with a fully connected topology (see Figure 1). Previous work [1, 6] used this model to analyze optical networks, but its application can be broader. Each station is both a source and destination[1] and has a data transmitter and receiver. Time is slotted. Data is transmitted synchronously in fixed-size packets. Each source can transmit at most one packet during a given slot, and each destination can receive at most one packet during a given slot.

The receivers have *capture* capability which is the ability to receive one packet even if collisions occur due to multiple sources sending to the same destination. Capture allows higher throughput than what is possible in multi-access systems without this feature. All transmissions are assumed to be completely reliable.

The propagation delay d_{prop} is the same between any source and destination (this assumption is relaxed in [18]). Each source has a buffer to hold the queue of packets which have not yet been sent. More than one packet can arrive at a source queue during a single slot. Each source queue can be viewed as many separate *virtual queues* by partitioning traffic by destination. The relationship among the arrival traffic model, the propagation delay, and other system parameters are considered in the analysis.

There exists a low-bandwidth broadcast control channel available to all stations for the exchange of source queue state information and future transmission information. This channel might, for example, share the physical medium with the data channel. The control channel bandwidth should be a small fraction of the total bandwidth. Each station has a separate

[1] A station may send a packet to itself in this model. Removing this assumption requires only minor modifications of transmission algorithms.

transmitter and receiver for the control channel.

A *transmission algorithm* assigns transmission slots on the data channel to packets in the source queues. A transmission algorithm which assigns slots dynamically based on the arrival traffic is called an *adaptive algorithm.* In a typical adaptive algorithm for non-uniform traffic, a source i with a packet for destination j sends an announcement during slot k on the control channel of the form "will transmit a packet from i to j in slot $k+1$". Each destination monitors these announcements during slot $k + d_{prop}$ and decides which source packet to capture during slot $k + 1 + d_{prop}$. The sources also monitor the control channel to determine the outcome of packet transmissions. An example of this timing is shown in Figure 2. This transmission protocol is often referred to as "tell and go". The extra slot delay due to the offset between the control channel and the data channel is ignored in this paper since it is small and dependent on the timing of the physical system. In fact, any small constant delay due to physical system limitations is ignored since relative performance of transmission algorithms is the focus of this paper.

An example of a communication system which fits the basic model is a broadcast network using a passive optical star as in the Rainbow project at IBM [6]. The data transmission occurs using wavelength division multiplexing. Each transmitter sends data on a fixed unique wavelength and each receiver has a tunable filter to receive from any one particular transmitter per slot. The control channel is a single unique wavelength shared among the stations using time division multiplexing. Each station needs a fixed transmitter and a fixed receiver on this wavelength. Rainbow is designed primarily for use in metropolitan area networks (MANs), but can be used in LANs or even as a high-speed packet switch.

In fact, any internally non-blocking packet switch with input queues and capture capability can be subsumed under the basic model. Such a switch commonly has a central controller and $d_{prop} \approx 0$. Another system which can be accommodated by the basic model is the satellite-switched system known as an SS/TDMA network [10]. The well-known time slot assignment problem for this system offers insight into the problems studied in this paper. Most passive optical star architectures can be accommodated by the basic model, possibly with some minor modifications.

2.2 Notation and Assumptions

Let $\mathcal{N} = \{1, \ldots n\}$ denote the set of stations. The virtual queues of source i are labeled (i, j) for $j \in \mathcal{N}$. *Slot* k refers to the time period $[k, k+1)$. The sequence of consecutive integers (slots) $k, \ldots, l-1$ is written $[k : l)$. Let Z be the set of integers and Z^+ be the set of non-negative integers.

During a given slot k, new packets arrive at a source and are placed in the buffer at the beginning of the slot. Any packet in the buffer is eligible for transmission. Next the source makes a decision about which (if any) packet to send. Finally, a packet may be *transmitted* onto the data channel. Those transmitted packets which are captured d_{prop} slots later are called *departures*. The transmission of packets which are guaranteed to be captured is called *scheduling*. In each slot, a given receiver can capture at most one packet per slot.

The *access delay* of a packet is defined to be the number of whole slots that the packet is present in the system before its departure. Note that the access delay does not include the propagation delay, d_{prop}, suffered by all packets. Because a packet can arrive and depart in the same slot, the minimum possible access delay is zero.

Fix an arrival sequence. For any packet p in the sequence, let d_p denote the access delay of p. Let $d_{max} = \max_p(d_p)$ denote the maximum delay for the sequence, and let d_{ave} denote the average delay for the sequence, defined by $d_{ave} = \limsup_{T\to\infty} d_{ave}(T)$ where $d_{ave}(T) = \frac{1}{a_T}\sum_{p\in\mathcal{A}_T} d_p$, $\mathcal{A}_T$ denotes the set of packets that arrive by time T, and $a_T = |\mathcal{A}_T|$, the cardinality of $\mathcal{A}_T$, for $T \in Z^+$.

The notation for arrivals, backlogs, and departures during slot k is as follows. The number of arrivals to source i destined for j is $A_{ij}(k)$. *After* the arrivals but *before* the departures, the number of packets in virtual queue (i, j) is $N_{ij}(k)$. The number of departures from source i to destination j is $D_{ij}(k)$, which is either 0 or 1. Since there are n sources and n destinations, these elements can be represented collectively as three $n \times n$ matrices: the arrival matrix $\mathbf{A}(k)$, the backlog matrix $\mathbf{N}(k)$, and the departure matrix $\mathbf{D}(k)$ (see Figure 3). The sum of all the elements in each of these matrices is represented by the scalars $A(k)$, $N(k)$, and $D(k)$ respectively. With this matrix representation, packets that arrive to (depart from) source i for destination j are said to arrive to (depart from) cell (i, j), row i, and column j. Let $R_i(k) = \sum_{j=1}^{n} N_{ij}(k)$ be the number of packets in row i of the backlog at time k. Let $C_j(k) = \sum_{i=1}^{n} N_{ij}(k)$ be the number of packets in column j of the backlog at time k.

Any matrix with non-negative integer entries is called a *traffic matrix* ($\mathbf{A}(k)$, $\mathbf{N}(k)$, and $\mathbf{D}(k)$, are traffic matrices for every k). When the time argument is absent from any scalar or matrix term then the sequence of such terms over integer time is indicated. The arrival matrix sequence $\mathbf{A}=(\mathbf{A}(k))_{k\in Z}$ describes the entire arrival process of all packets for all sources and destinations. The sequence $A_{12} = (A_{12}(k))_{k\in Z}$ represents the packets that arrive to source 1 bound for destination 2. Such a non-negative integer sequence over integer time is more commonly referred to as a *traffic stream.*

The system has no arrivals or departures until slot 0. The initial backlog is $\mathbf{N}(-1)$ and is carried over at time $k = 0$ since the backlog is measured in the middle of a slot. Unless otherwise noted, $\mathbf{N}(-1)$=0. The backlog evolution is very simple and is given by $\mathbf{N}(k) = \mathbf{N}(k-1) - \mathbf{D}(k-1) + \mathbf{A}(k)$ for $k \geq 0$. The $\mathbf{D}(k)$ matrices are examples of *switching matrices* since each represents one slot setting of the "crossbar switch" between sources and destinations. These are partial (perhaps full) permutation matrices. It is assumed that the switching time is zero.

3 A New Model of Non-Uniform Traffic

Many models of non-uniform traffic streams have been proposed (see for example [2, 5, 11]). These can generally be classified into two categories. *Random* models have random arrivals with a specified distribution that allows for the possibility of non-uniform traffic matrix sequences. Examples include Markov-modulated Poisson arrivals and geometric burst-length on/off arrivals. *Deterministic constraint* models do not require an underlying arrival distribution but instead require that each traffic matrix sequence meets some specified constraints. It is the second type of model which most easily simultaneously allows significant non-uniformity of traffic and worst-case analysis for QOS. The class of deterministic constraint models introduced in this paper are called *line-constrained* non-uniform traffic models.

The term *line* is used to refer to either a row or column of a matrix. A *line sum* is the sum of all elements in a line of a matrix. The maximum line sum of a matrix is indicated by the operator $\| \cdot \|$. *Line backlogs* are the line sums of the matrices in $\mathbf{N}$, and the *maximum line backlog* is $\max_{k\in Z} \| \mathbf{N}(k) \|$, denoted $\| \mathbf{N} \|$. Each line sum of the arrival matrix sequence $\mathbf{A}$ can be viewed as a traffic stream—the arrivals bound for a particular destination form a traffic stream as do the arrivals at a particular source. Note that a given packet is an element of two streams—one corresponding to its row (source) and one corresponding to its column (destination). There are $2n$ traffic streams in a traffic matrix sequence.

Line-constrained traffic models constrain each line (a traffic stream) of the arrival matrix sequence. By constraining each of the $2n$ traffic streams, the arrival matrix sequence can be constrained while allowing for some bursts of data. Many examples of constraints on traffic streams exist, such as the (σ, ρ) constraint of [2] and the stochastic domination constraint of [11]. In this paper an (α, S) constraint is used, defined as follows. A traffic stream is said to be (α, S) if in any S consecutive slots the stream contains at most αS packets, where S is any positive integer, and $0 < \alpha \leq 1$. An S-dimensional state is required for an αS stream regulator. An example of a similar type of traffic is found in [19]. Periodic, multiplexed traffic is one type of traffic easily accommodated by this model.

Maximum flexibility is achieved by constraining each line of $\mathbf{A}$ differently. The additional complexity of analysis though discourages this approach. Significant overall non-uniformity can be allowed by requiring each line to be (α, S). The arrival matrix sequence is then called (α, S) traffic. The parameter α represents the maximum long-run throughput allowed for any source or destination. The parameter S is the smallest time period over which the throughput constraint is enforced. An (α, S) traffic sequence can thus be simply described: "Over any time period of length S no more than αS packets arrive at a given source and no more than αS packets bound for a given destination arrive at all sources." Subject to this constraint, all traffic patterns (possibly random) are allowed. For convenience, αS is usually taken to be an integer. Arrival sequences with sources exhibiting on/off bursts or "hot spots"—commonly studied as representative non-uniform traffic—are easily accommodated with the (α, S) model.

With two degrees of freedom provided by the choice of α and S, the (α, S) traffic model is a flexible and widely applicable model of non-uniform traffic. *Design and analysis of transmission algorithms for (α, S) traffic is the focus of the remainder of this paper.* All delay analysis is for (α, S) traffic sample paths. Simulation is of limited value since there are usually too many sample paths to run individually.

4 Zero Propagation Delay

When the propagation delay between stations is negligible, each station can be made immediately aware of arrivals at all other stations. Collision-free scheduling is possible since the state of each virtual queue is globally available when each station executes a known transmission algorithm. This section develops transmission algorithms for use when d_{prop} is zero. These transmission algorithms are called *scheduling algorithms* since all collisions are avoided. The scheduling algorithms of this section are also useful for very low d_{prop} systems such as a packet switch with central controller.

For any non-zero d_{prop}, the transmission algorithms of this section can be used with additional delay of d_{prop} for each packet. Each packet is simply ignored by the transmission algorithm for d_{prop} slots after its arrival while notification of its arrival propagates to all other stations. Scheduling may be necessary for some non-uniform traffic since collisions, which cause additional delay, are difficult if not impossible to avoid without shared information.

Several properties of transmission algorithms are typical and desirable for QOS guarantees in networks:

(P1) Maximum delay is bounded

(P2) Average delay is bounded

(P3) Each line backlog is bounded

(P4) Each line backlog hits zero persistently

A class of transmission algorithms is said to have property P if every algorithm in that class has property P. These properties serve as design goals for the scheduling algorithms developed in this section.

4.1 The Time Slot Assignment Problem

The most direct approach to global information scheduling for (α, S) traffic borrows from work on a classical problem in satellite switching. It is worth a brief digression to examine this problem, known as the time slot assignment problem (TSA) for an SS/TDMA system. An *assignment* for a traffic matrix is a finite sequence of switching matrices whose sum equals the traffic matrix. The *duration* of the assignment is the length of the sequence. The number of *modes* of the assignment is the number of unique switching matrices in the sequence.

The TSA problem [10] can now be stated simply: Given a traffic matrix, find an assignment of minimum duration which also minimizes the number of switching modes. This problem has been shown to be NP-complete in [8]. For small (or zero) switching times, the goal is to simply find a minimum duration assignment. In the packet switching context, it is also desirable to find assignments which minimize the delay of packets in the traffic matrix. Several scheduling algorithms for (α, S) traffic are derived from the principles behind the solution of the TSA problem for SS/TDMA.

A matrix is *balanced* if all its line sums are equal. Any traffic matrix can be made to be balanced by adding dummy traffic (see for example [7]). A fundamental result of combinatorial mathematics is that any balanced traffic matrix can be represented as the sum of permutation matrices [9]. The minimum duration required is equal to the line sums of the balanced traffic matrix. Any algorithm which finds such an assignment is said to be 100% *efficient.* Thus, any traffic matrix with maximum line sum L has an assignment of duration L.

Algorithms which find 100% efficient assignments one step at a time by subtracting switching matrices from the remaining traffic matrix are known (see for example [10, 15]). Any matrix has an associated bipartite graph (called by the same name as the matrix) which is constructed as follows. Let one set of nodes correspond to rows of the matrix and the other set of nodes correspond to columns. Let there be an edge between any two nodes where the corresponding matrix position has a non-zero element. A node is *critical* if the corresponding line in the matrix has maximum line sum. At each step the switching matrix to subtract can be found by finding a maximum matching[2] on the traffic graph (associated with the remaining traffic matrix) which covers the critical nodes of this graph. This is called a *critical maximum matching.* See Figure 4 for a TSA example.

4.2 Batch Scheduling

Define the *batch interval* b to be slots $[b\alpha S : (b+1)\alpha S)$. A batch scheduling algorithm queues the *batch* of packets that arrive during batch interval b and schedules this batch of packets during slots $[(b+1)\alpha S - 1 : (b+2)\alpha S - 1)$ using an arbitrary 100% efficient TSA algorithm. This departure interval is batch interval $b+1$ offset by one slot (some batch b packets can arrive and depart in slot $(b+1)\alpha S - 1$). The *batch backlog matrices* $\mathbf{N}^B(k)$, $k = 0, 1, \ldots$, are

[2] A matching in a graph is any set of edges such that no two edges share a node. A *maximum* matching is a matching of maximum cardinality. A *maximal* matching is a matching which cannot be extended without removing edges. For a $2n$-node bipartite graph, maximum matching can be done in time $O(n^{2.5})$ and maximal matching can be done in time $O(n^2)$. Note that a maximum matching is a maximal matching. See [13] for details.

defined as

$$\mathbf{N}^B(k) = \begin{cases} \mathbf{N}(k) & k = b\alpha S - 1, b \in Z^+ \\ \mathbf{N}^B(k-1) - \mathbf{D}(k-1) & \text{otherwise} \end{cases}$$

Note that $\mathbf{N}^B(-1)$=$\mathbf{N}(-1)$=0 so that $\mathbf{N}^B(\alpha S - 1)$ is the first non-empty batch backlog matrix. Since the packets in each batch backlog matrix can be scheduled in αS slots, the (α, S) arrival constraint guarantees $\| \mathbf{N}^B \| \leq \alpha S$.

The batch backlog matrices are identical to the backlog matrices except that arrivals are added only at times $\alpha S - 1, 2\alpha S - 1, \ldots$ so that $\mathbf{N}^B(b\alpha S - 1) = \sum_{k=(b-1)\alpha S}^{b\alpha S - 1} \mathbf{A}(k)$. Let BATCH_MAXIMUM be the class of batch scheduling algorithms which in each slot subtract a switching matrix from the most recent batch backlog matrix found by a critical maximum matching on the associated batch backlog graph.

Theorem 1 *For $\alpha \leq 1$, BATCH_MAXIMUM has properties P1–P3. The maximum delay is $2(\alpha S - 1)$. The average delay is no more than $\frac{3}{2}(\alpha S - 1)$. The maximum line backlog is no larger than $2\alpha S$.*

Proof. Fix any scheduling algorithm in BATCH_MAXIMUM. Without loss of generality, consider batch 0. Any packet arriving after time 0 departs before time $2\alpha S - 1$ so $d_{max} = 2(\alpha S - 1)$. The number of batch 0 packet departures per slot is non-increasing in the consecutive slots $[\alpha S - 1 : 2\alpha S - 1)$ so $d_{ave} \leq \frac{3}{2}(\alpha S - 1)$. At any time, packets from at most two batches may be in the source queues so $\| \mathbf{N} \| \leq 2\alpha S$. □

In addition to using the batch backlog graphs, BATCH_MAXIMUM algorithms also use information about the line sums. A class of algorithms which do not use such side information is now defined. A *static* batch scheduling algorithm is a scheduling algorithm obtained by applying one fixed deterministic matching algorithm to the batch backlog graph in each slot. Recall that maximal matchings generally require less time to compute than maximum matchings. Let BATCH_MAXIMAL be the class of batch scheduling algorithms which in each slot subtract a switching matrix from the most recent batch backlog matrix found by a maximal matching on the associated batch backlog graph. Any scheduling algorithm in BATCH_MAXIMUM is in BATCH_MAXIMAL. If one fixed deterministic matching algorithm is used, then BATCH_MAXIMAL is a static batch scheduling algorithm. Twice as many slots may be needed to evacuate a batch as compared to maximum matching since the cardinality of any maximal matching on a bipartite graph can be as small as half the cardinality of a maximum matching on the same graph. The batch backlog matrices are now taken to be

$$\mathbf{N}^B(k) = \begin{cases} \mathbf{N}(k) & k = 2b\alpha S - 1, b \in Z^+ \\ \mathbf{N}^B(k-1) - \mathbf{D}(k-1) & \text{otherwise} \end{cases}$$

since each batch may require $2\alpha S$ slots for evacuation. Here, $\mathbf{N}^B(2\alpha S - 1)$ is the first non-empty batch backlog matrix. The following theorem is a positive result for a simple class of scheduling algorithms, namely batch scheduling algorithms (including static algorithms) using maximal matchings.

Theorem 2 *For $\alpha \leq \frac{1}{2}$, BATCH_MAXIMAL has properties P1–P3. The maximum delay is $4(\alpha S - \frac{1}{2})$. The average delay is no more than $3(\alpha S - \frac{1}{2})$. The maximum line backlog is no larger than $2\alpha S$.*

Proof. The bounds for d_{max} and d_{ave} are established as in the proof of Theorem 1 replacing αS by $2\alpha S$. The line backlog bound is the same as in Theorem 1 by the same reasoning. □

Remark 1 *For $\alpha > \frac{1}{2}$, Appendix A gives an example demonstrating that at least some algorithms in BATCH_MAXIMAL do not have any of the properties P1–P3.*

BATCH_MAXIMUM and BATCH_MAXIMAL algorithms provide maximum delay of $2(\alpha S - 1)$ and $4(\alpha S - \frac{1}{2})$ respectively. For non-zero d_{prop}, an additional d_{prop} for global information must be added. If αS is small then batch scheduling algorithms do very well for zero (or small) propagation delay in the sense that delay is small and throughputs as high as 1 can be obtained. As αS increases it is natural to ask if better scheduling is possible. The next section addresses this question by allowing packets to be considered for departure immediately upon arrival without waiting for transmission of the previous batch.

4.3 Continuous Scheduling

The continuous scheduling algorithms of this section find the departure matrices in each slot by using a matching on the backlog graph associated with the backlog matrix in the same slot. No batching occurs. Because of the complexity, the proofs of all lemmas and theorems in Section 4.3 are relegated to Appendix B.

Let CONTINUOUS_MAXIMAL be the class of continuous scheduling algorithms which use an arbitrary maximal matching in each slot. This is perhaps the simplest class of scheduling algorithms presented in this paper. If one fixed deterministic matching algorithm is used then a CONTINUOUS_MAXIMAL algorithm is a static continuous scheduling algorithm.

Theorem 3 *For $\alpha \leq \frac{1}{3}$, CONTINUOUS_MAXIMAL has properties P1–P3. The maximum delay is no more than $6(\alpha S - \frac{1}{3})$. The average delay is no more than $2(\alpha S - \frac{1}{2})$. The maximum backlog is no larger than αS.*

Let OLDEST_MAXIMAL be the class of continuous scheduling algorithms which use any oldest-first maximal matching in each slot. An oldest-first maximal matching is a matching with the following property—any packet not in the matching is blocked by a packet in the matching which arrived to the system not later than the blocked packet.

Theorem 4 *For $\alpha \leq \frac{1}{2}$, OLDEST_MAXIMAL has properties P1–P3. The maximum delay is no more than $2(\alpha S - 1)$. The average delay is no more than $2(\alpha S - 1)$. The maximum line backlog is no larger than αS.*

Finally, note that continuous scheduling for $\frac{1}{2} < \alpha \leq 1$ may be possible. The matchings used for such scheduling must depend on time, random events, or the system history due to the following negative result. Let STATIC_MAXIMUM be the class of static continuous scheduling algorithms using maximum matchings.

Theorem 5 *For $\alpha > \frac{1}{2}$, $S \geq 6$, and $n \geq 3$, no scheduling algorithm in STATIC_MAXIMUM has any property P1–P4. CONTINUOUS_MAXIMAL does not have any property P1–P4 since any scheduling algorithm in STATIC_MAXIMUM is also in CONTINUOUS_MAXIMAL.*

Table 1 summarizes the throughput and delay characteristics of the scheduling algorithms presented in this section. The last three columns list the upper bounds for $\| \mathbf{N} \|$, d_{ave}, and d_{max} respectively. The symbol $\star\star\star$ means that the corresponding property does not hold for the particular class of scheduling algorithm and range of α.

Table 1–Performance Summary for Zero Propagation Delay Algorithms.

Class of Scheduling Algorithms	Maximum Throughput	Property P3 $\parallel \mathbf{N} \parallel$	Property P2 d_{ave}	Property P1 d_{max}
BATCH_MAXIMUM	$\alpha \leq 1$	αS	$\frac{3}{2}(\alpha S - 1)$	$2(\alpha S - 1)$
BATCH_MAXIMAL	$\alpha \leq \frac{1}{2}$	αS	$3(\alpha S - \frac{1}{2})$	$4(\alpha S - \frac{1}{2})$
	$\alpha > \frac{1}{2}$	$\star\star\star$	$\star\star\star$	$\star\star\star$
CONTINUOUS_MAXIMAL	$\alpha \leq \frac{1}{3}$	αS	$2(\alpha S - \frac{1}{2})$	$6(\alpha S - \frac{1}{3})$
	$\alpha > \frac{1}{2}$	$\star\star\star$	$\star\star\star$	$\star\star\star$
OLDEST_MAXIMAL	$\alpha \leq \frac{1}{2}$	αS	$2(\alpha S - 1)$	$2(\alpha S - 1)$
STATIC_MAXIMUM	$\alpha > \frac{1}{2}$	$\star\star\star$	$\star\star\star$	$\star\star\star$

The strongest property is P4, persistent line backlog emptying. For static scheduling algorithms P4 holds for only relatively small throughput values as shown in the following theorem and the associated remark.

Theorem 6 *For* $\alpha \leq \frac{\sqrt{4S+9}-3}{2S}$ *(*$\alpha < \frac{1}{\sqrt{S}}$ *asymptotically in* S*), CONTINUOUS_MAXIMAL has property P4 and each line backlog hits zero at least once every* S *slots.*

Remark 2 *For* $\alpha \geq \frac{1}{\sqrt{S}}$*, an example in Appendix A demonstrates that at least some algorithms in CONTINUOUS_MAXIMAL do not have property P4.*

Recall that all the transmission algorithms of this section require global information. In particular, the state of all the system virtual queues is known at every station. What, if anything, can be done to transmit packets when global information is not available? The scheduling algorithms of this section perform well for small, non-zero d_{prop} (by delaying packets d_{prop} upon arrival). The transmission algorithm of the next section is designed for large d_{prop}. Consideration of transmission algorithms for medium values of d_{prop} can be found in [18]. Note that all of the transmission algorithms can be used for the given values of α over the entire range $d_{prop} \in [0, \infty)$. However, hybrid combinations of these transmission algorithms and new algorithms may provide better throughput and delay characteristics.

5 Large Propagation Delay

This section suggests how to transmit (α, S) traffic when the propagation delay is large, specifically as $d_{prop} \rightarrow \infty$. Roughly speaking, it is shown that the average access delay can be made small compared to d_{prop} when the arrival rate is not too high.

The transmission algorithm is first described and then the delay is analyzed. The delay of any fixed packet is random because the algorithm is random. *Frame* f refers to slots $[fM : (f+1)M)$ where M, the *frame length*, is a multiple of αS ($M = l\alpha S$). For brevity, the transmission algorithm is described for packets arriving during frame 0. These are referred to as frame 0 packets. The description applies to packets arriving during frame f by adding fM to all the times.

The transmission algorithm attempts to minimize collisions by randomly spreading out transmissions over a long period of time. The destinations are notified in advance of transmission times. Packet transmission occurs in two phases. During phase 1, each packet is transmitted three times in randomly chosen slots. During phase 2, those packets which are not received in phase 1 are scheduled after delay of d_{prop} using an algorithm described earlier. Phase 1 thus uses 75% of the total transmission slots and phase 2 uses the other 25%. See Figure 5 for the timing of this two-phase algorithm.

Phase 1 Transmission Frame 0 packets arriving at a particular source are transmitted in an interval of length $\frac{3}{4}M$. Before any transmission, the whole frame must be accumulated and all destinations must be notified of the packet transmission slots. This requires an accumulation delay, d_{acc}, from the start of frame 0 until the first packet of the frame can be transmitted. A frame 0 packet which arrives at a particular source during batch interval b is assigned three distinct slots chosen uniformly at random from among all slots in $[d_{acc} : d_{acc} + \frac{3}{4}M)$ which have not already been assigned by the source to other frame 0 packets. These are the packet's phase 1 *transmission slots*, s_1, s_2, and s_3. If the packet arrives during batch interval b (within frame 0), then during batch interval $b+1$, the vector (source, destination, s_1, s_2, s_3) is sent on the control channel to announce the future transmissions. Since a batch contains no more than αS packets, these control information vectors can be accommodated one per slot. Each vector is of length $2\log n + 3\log(\frac{3}{4}M)$. Finally, the packet is transmitted during slots s_1, s_2, and s_3. Set d_{acc}=$M + \alpha S$. This assignment of d_{acc} leaves adequate time for the sources to accumulate and announce transmission times of packets before actual transmission occurs, as described. Note that $d_{acc} \leq 2M$.

Phase 2 Transmission At time $d_{acc} + d_{prop}$, a source knows whether a particular frame 0 packet will be captured during phase 1 reception. The packets which will not be captured in phase 1 are scheduled in slots $[d_{acc} + d_{prop} + d_{sync} : d_{acc} + d_{prop} + d_{sync} + \frac{1}{4}M)$, where d_{sync} is a synchronization delay of at most M slots, chosen to avoid overlap of phase 1 and phase 2 transmissions. This scheduling is possible using a BATCH_MAXIMUM algorithm since there are no more than αM frame 0 packets for any destination and from any source.

Phase 1 Reception Focus on a particular receiver, which during slots $[d_{acc} + d_{prop} : d_{acc} + d_{prop} + \frac{3}{4}M)$ receives frame 0 packets sent in phase 1 from all sources. Let R denote the number of such packets. Note that $R \leq \alpha M$ by the assumption that the traffic sequence is (α, S). The locations of each frame 0 packet's three transmission slots are known to the receiver by time $d_{acc} + d_{prop}$. This allows the receiver to construct a directed bipartite graph $G = (U, V, E)$ where

$$\begin{aligned} U &= \{\text{nodes representing frame 0 packets destined for the receiver}\},\ |U| = R \\ V &= \{\text{nodes representing the slots to receive frame 0 packets}\},\ |V| = \frac{3}{4}M \\ E &= \{(u, v) : u \in U, v \in V, \text{ packet } u \text{ is sent during slot } v\} \end{aligned}$$

Consider any matching on G. Each slot in V is matched to at most one packet in U and each packet in U is matched to at most one slot in V. Unmatched slots in V are unused for reception of frame 0 packets and unmatched packets in U are not successful in phase 1. To maximize throughput in phase 1, the choice of which packet to receive in each slot is made by finding a maximum matching on G. An example is shown in Figure 6.

Phase 2 Reception Even though most frame 0 packets succeed in phase 1, some fail. During slots $[d_{acc} + 2d_{prop} + d_{sync} : d_{acc} + 2d_{prop} + d_{sync} + \frac{1}{4}M)$, these leftover packets are received without conflict since they are scheduled using a BATCH_MAXIMUM algorithm.

This completes the description of the transmission algorithm. This algorithm provides average delay close to the minimum possible, as shown by the next theorem which is proved in Appendix C.

Theorem 7 *Choose any $\alpha \leq \frac{1}{4}$, integer $S \geq 1$, and $\epsilon > 0$. If d_{prop} is sufficiently large and the above randomized transmission algorithm is used, then the following is true: Given any (α, S) arrival sequence and any fixed packet p in that sequence, the expected access delay of the packet, d_p, is at most ϵd_{prop}.*

This transmission algorithm is conservative in the sense that each packet is guaranteed success on its first transmission in phase 2. Less conservative transmission algorithms may provide smaller average delay but larger maximum delay. Some comments on implementation details are in order. First, if d_{prop} has the form $(k + \frac{3}{4})M$ for some integer k, then d_{sync} can be taken to be 0. It is interesting to note that there are two ways to make $M = l\alpha S$ large. First, l can be made large. Second, each batch interval can be extended by making S large, although delay also increases for finite d_{prop}. In either case, M must be $o(d_{prop})$ as d_{prop} goes to ∞. This transmission algorithm is stable for any M, but the average access delay may be prohibitively large if too small a value for M is chosen.

The control information required is relatively small, but computation can be prohibitively large without some modification. The algorithm as given requires each source to execute the maximum matching algorithm for each destination to which it sends packets in order to determine phase 1 successes. In practice it is probably desirable to wait an additional d_{prop} before phase 2 to allow feedback information from phase 1 to propagate back to the sources, and send explicit phase 1 acknowledgements over the control channel. The maximum matching at each receiver can proceed incrementally as each new control information vector is received. Since a packet seldom gets to phase 2, the additional delay does not affect the result in Theorem 7. Of course, for $\alpha \leq \frac{1}{5}$, maximal matching will suffice (with a 60%/40% allocation of slots between phases 1 and 2).

6 Conclusion

The transmission algorithms presented in this paper perform well for (α, S) traffic using the basic model. When certain assumptions in the basic model are removed, the transmission algorithms may need to be modified. Three restrictions of the basic model are particularly important to allow it to more closely model physical systems:

- **Non-zero switching time**— Most systems can emulate non-zero switching time (and hence use the existing algorithms) by using an extra transmitter and receiver at each station, alternately using one transmitter/receiver pair for data while the other pair switches. Alternatively, the slot length can be made equal to the packet transmission time plus the switching time. If these solutions are too costly, improved transmission algorithms are needed.

- **Non-uniform propagation delay**— In this paper, all stations are separated in time by d_{prop} slots. In general, the propagation delay could be different for each pair of stations. The scheduling algorithms of Section 4 can be easily modified to accommodate such non-uniform propagation delays when a star topology is used. Improved transmission algorithms for general topologies are currently being studied.

- **Limited control channel**— The scheduling algorithms of Section 4 require no more than an average of $n \log n$ bits per slot of control channel capacity. This is achieved by announcing each cell arrival to all stations. An additional access delay of αS may be incurred to achieve this rate for non-zero d_{prop}. The transmission algorithm of Section 5 requires $2n \log n + 3n \log(\frac{3}{4}M)$ bits per slot of control channel capacity with no additional delay. When the available control channel capacity is less than $O(n \log n)$ bits per slot, additional delays must be suffered or new transmission algorithms must be designed.

These and other restrictions, such as limited transmission/reception concurrency and limited source queue access, are considered in [18]. Limited computation is also considered there, although the transmission algorithms of this paper using maximal matchings are relatively time-efficient.

Finally, a scheme for enforcing the (α, S) traffic constraint needs to be developed. Currently, it is envisioned that a virtual circuit passing through source i to destination j reserves some fraction of the "αS packets per S slots" bandwidth from both the source and the destination during circuit setup. Such connection-oriented traffic is most easily supported, but datagram traffic which is (α, S) is also supported. A mechanism for (α, S) regulation of arbitrary arrival matrix sequences would be very useful.

7 Acknowledgments

This paper is based on work supported by an AT&T Bell Laboratories Scholarship and the National Science Foundation under contract NSF NCR 90 04355.

8 Appendix A—Examples of Pathological Algorithms and Arrival Sequences

The following examples assume that $n > \alpha S$.

8.1 Example for Remark 1

An example of an (α, S) arrival sequence and an algorithm in BATCH_MAXIMAL are presented such that for $\alpha > \frac{1}{2}$ the algorithm does not have any of the properties P1–P3.

Consider the first non-empty batch backlog matrix, $\mathbf{N}^B(2\alpha S - 1)$. Define it as

$$N^B{}_{ij}(2\alpha S - 1) = \begin{cases} \alpha S - 1 & \text{if } i = j \text{ and } i \leq \alpha S \\ 1 & \text{if } j = \alpha S + 1 \text{ and } i \leq \alpha S \\ 0 & \text{otherwise} \end{cases}$$

Suppose that a static BATCH_MAXIMAL algorithm is used which determines the following departure matrices for $k \in [2\alpha S - 1 : 3\alpha S - 2)$:

$$D_{ij}(k) = \begin{cases} 1 & \text{if } i = j \text{ and } i \leq \alpha S \\ 0 & \text{otherwise} \end{cases}$$

For $k \in [3\alpha S - 2 : 4\alpha S - 2)$:

$$D_{ij}(k) = \begin{cases} 1 & \text{if } k = i + 3\alpha S - 3 \text{ and } j = \alpha S + 1 \\ 0 & \text{otherwise} \end{cases}$$

It is easily verified that each of these corresponds to a maximal matching (indeed, a maximum matching). For example, the scheduling decomposition $\mathbf{N}^B(2\alpha S - 1) = [\mathbf{D}(2\alpha S - 1) + \cdots + \mathbf{D}(3\alpha S - 3)] + [\mathbf{D}(3\alpha S - 2) + \cdots + \mathbf{D}(4\alpha S - 3)]$ for $n = 3$ and $\alpha S = 3$ is

$$\begin{pmatrix} 2 & 0 & 0 & 1 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

The BATCH_MAXIMAL algorithm considered schedules the sub-diagonal matrices first during slots $[2\alpha S - 1 : 3\alpha S - 2)$ and then column $\alpha S + 1$ during slots $[3\alpha S - 2 : 4\alpha S - 2)$ with the packet in cell $(\alpha S, \alpha S + 1)$ scheduled last (in slot $4\alpha S - 3$). Now suppose $(4\alpha S - 3) - (2\alpha S - 1) = 2\alpha S - 2$ is greater than S (as is true for S large enough with $\alpha > \frac{1}{2}$), and let $\mathbf{N}^B(2b\alpha S - 1) = \mathbf{N}^B(2\alpha S - 1)$ for $b \in Z^+$. An infinite backlog accumulates in cell $(\alpha S, \alpha S + 1)$ since in slots $S, 2S, \ldots$ the scheduling pattern begins again with the sub-diagonal matrices and no departure ever occurs from cell $(\alpha S, \alpha S + 1)$. Of course, an infinite cell backlog implies that none of the properties P1–P3 holds.

8.2 Example for Remark 2

An arrival sequence and scheduling algorithm in CONTINUOUS_MAXIMAL are presented such that $R_1(k) > 0$, $\forall k \in Z$, that is, the row 1 line sum is always non-zero after time 0. There are arrivals only in slots $b\alpha S$ for $b = 0, 1, 2, \ldots$, described as

$$A_{ij}(b\alpha S) = \begin{cases} 1 & \text{if } j = (b \bmod \alpha S) + 1 \text{ and } i \leq \alpha S \\ 0 & \text{otherwise,} \end{cases}$$

where mod is the integer modulo operator. This arrival sequence repeats every $(\alpha S)^2$ slots. In order for this sequence to be (α, S) the condition $(\alpha S)^2 \geq S$ is required. This is equivalent to $\alpha \geq \frac{1}{\sqrt{S}}$. In slot k, the matching algorithm schedules a single departure from cell (i, j) where $i = \alpha S - (k \bmod \alpha S)$ and $j = (\lfloor \frac{k}{\alpha S} \rfloor \bmod \alpha S) + 1$. In any slot, only one column is non-empty, but row 1 is never empty at a backlog measurement time so P4 does not hold.

9 Appendix B—Proofs of Zero Propagation Delay Lemmas and Theorems

Before proving the zero propagation delay lemmas and theorems, a useful lemma which relates line backlogs and average delay is presented. The lemma is not immediate from Little's Law since no arrival rate is specified.

Lemma 1 *Suppose d_{max} is finite. For a continuous scheduling algorithm using maximum matchings, $d_{ave} \leq \| \mathbf{N} \| - 1$. For a continuous scheduling algorithm using maximal matchings, $d_{ave} \leq 2\| \mathbf{N} \| - 1$.*

Proof. In the notation of this paper, the König-Egerváry Theorem [12] states that the cardinality of a maximum matching on a backlog graph is equal to the minimum number of lines in the associated backlog matrix which together contain all the non-zero entries. Since no line sum of $\mathbf{N}(k)$ is greater than $\| \mathbf{N}(k) \|$, the following relationship holds:

$$D(k) \geq \frac{N(k)}{\| \mathbf{N}(k) \|} \geq \frac{N(k)}{\max_{k \in Z} \| \mathbf{N}(k) \|} = \frac{N(k)}{\| \mathbf{N} \|}. \quad (1)$$

Recall that $\mathcal{A}_T$ is the set of packets that arrived duringn slots $[0 : T)$ and $a_T = |\mathcal{A}_T|$. Let d_T be the number of departures during slots $[0 : T)$. Clearly, $d_T \leq a_T$ since the system starts empty. Let η_T be the total delay suffered in the interval $[0, T)$ by all packets in $\mathcal{A}_T$. Let $\tilde{\eta}_T$ be the total delay suffered in the interval $[T, \infty)$ by all packets in $\mathcal{A}_T$. The following development is similar to the development of Little's Law. The second inequality uses (1).

$$\begin{aligned} d_{ave}(T) &= \frac{\eta_T + \tilde{\eta}_T}{a_T} \\ &\leq \frac{\eta_T}{d_T} + \frac{\tilde{\eta}_T}{a_T} \\ &= \frac{\sum_{k=0}^{T-1} [N(k) - D(k)]}{\sum_{k=0}^{T-1} D(k)} + \frac{\tilde{\eta}_T}{a_T} \\ &\leq \| \mathbf{N} \| - 1 + \frac{\tilde{\eta}_T}{a_T} \end{aligned}$$

Now suppose $T \to \infty$. If $a_T \to \infty$, then $\frac{\tilde{\eta}_T}{a_T} \to 0$ since $\tilde{\eta}_T \leq d_{max} \| \mathbf{N} \|$ remains finite. If a_T is bounded, then $\tilde{\eta}_T \to 0$ as the system empties and again $\frac{\tilde{\eta}_T}{a_T} \to 0$. Therefore,

$$d_{ave} = \limsup_{T \to \infty} d_{ave}(T) \leq \| \mathbf{N} \| - 1.$$

Since for a given bipartite graph the cardinality of any maximal matching is at least half the cardinality of any maximum matching, the second result follows in the same manner. □

Proof of Theorem 3. The proof of Theorem 3 relies on the following lemma. Recall that the row i and column j line sums of $\mathbf{N}(k)$ are $R_i(k)$ and $C_j(k)$ respectively.

Lemma 2 *Using any algorithm in CONTINUOUS_MAXIMAL with* $\alpha \leq \frac{1}{3}$, *if* k *is any integer such that* $\| \mathbf{N}(k) \| \leq \alpha S$ *then* $\forall i, j \in \mathcal{N}$, *at least* $R_i(k)$ *packets depart from row* i *and* $C_j(k)$ *packets depart from column* j *during slots* $[k : k + 3\alpha S)$.

Proof. Fix k and $i \in \mathcal{N}$. The result is proved only for row i because of symmetry between rows and columns. Consider two cases. First, suppose that for each $j \in \mathcal{N}$, at least $N_{ij}(k)$

packets depart from cell (i, j) during slots $[k : k + 3\alpha S)$. Summing over j yields that at least $R_i(k)$ packets depart from row i during slots $[k : k + 3\alpha S)$, and the first case is covered. In the second case, there is some $j \in \mathcal{N}$ such that less than $N_{ij}(k)$ packets depart from cell (i, j) during slots $[k : k + 3\alpha S)$, so that $N_{ij}(l) \geq 1$ for $k \leq l \leq k + 3\alpha S$. Therefore, at least one packet must depart from either row i or column j during each of the $3\alpha S$ slots in $[k : k + 3\alpha S)$. At most $C_j(k) - N_{ij}(k) + \alpha S \leq 2\alpha S - 1$ of these $3\alpha S$ packets can be from column j but not row i, since at most αS new packets arrive to column j during slots $[k : k + 3\alpha S)$. Thus, at least $3\alpha S - (2\alpha S - 1) = \alpha S + 1$ packets depart row i during slots $[k : k + 3\alpha S)$. Since $\alpha S + 1 \geq R_i(k)$, both cases are covered and the lemma is proved.

□

Proof of Theorem 3 continued. Fix any algorithm in CONTINUOUS_MAXIMAL. Argue by induction that $\| \mathbf{N}(k) \| \leq \alpha S$, $\forall k \in Z$. By assumption $\| \mathbf{N}(k) \| = 0$ for $k < 0$. For any $k \in Z$, if $\| \mathbf{N}(k) \| \leq \alpha S$ then $\| \mathbf{N}(k + S) \| \leq \alpha S$ by Lemma 2 and the (α, S) arrival constraint. Therefore $\| \mathbf{N}(k) \| \leq \alpha S$ for all k by induction, so property P3 holds. Now consider the delay of any fixed packet arriving to cell (i, j) in slot k (counted in $N_{ij}(k)$). At most $2\alpha S$ packets arrive to row i and at most $2\alpha S$ packets arrive to column j during slots $[k : k + 6\alpha S)$, so there are at most $C_j(k) - 1 + R_i(k) - 1 + 4\alpha S \leq 6\alpha S - 2$ packets which can block the fixed packet during slots $[k : k + 6\alpha S)$. Therefore, $d_{max} \leq 6\alpha S - 2$. By Lemma 1, $d_{ave} \leq 2\alpha S - 1$. This completes the proof of Theorem 3. □

Proof of Theorem 4. The proof is exactly the same as that of Theorem 3 using Lemma 2 except that $3\alpha S$ is replaced by $2\alpha S$ everywhere. In the second case, at most $C_j(k) - N_{ij}(k) \leq \alpha S - 1$ of the $2\alpha S$ packets can be from column j but not row i, since no new packets which arrive to column j during slots $[k : k + 2\alpha S)$ can block packets in $N_{ij}(k)$. The maximum delay is no more than $2(\alpha S - 1)$ since a packet in cell (i, j) can be blocked by at most $\alpha S - 1$ row i packets and at most $\alpha S - 1$ column j packets. Clearly, $d_{ave} \leq d_{max}$. □

Proof of Theorem 5. Fix $S \geq 6$. Before considering an arbitrary algorithm in the class STATIC_MAXIMUM, a particular algorithm for an $n = 2$ station system will be examined. Consider the following arrival matrix sequence during slots $[0 : 2l(m + 1))$:

$$\overbrace{\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}, \underbrace{\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}}_{\text{Repeat } m \text{ times}}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}}^{\text{Repeat } l \text{ times}},$$

The inner two matrices are repeated m times and then the whole sequence of $2m+2$ matrices is repeated l times. Consider a STATIC_MAXIMUM algorithm with the following property—in each slot, the algorithm schedules the packet in the upper right corner cell or the lower left corner cell, ignoring the packet in the upper left corner cell. This upper left corner cell backlog can be made arbitrarily large by making l large. The sequence satisfies the (α, S) constraint for sufficiently large m. The backlog matrices in this previous two station example are $\begin{pmatrix} \beta & 0 \\ 1 & 0 \end{pmatrix}$ and $\begin{pmatrix} \beta & 1 \\ 0 & 0 \end{pmatrix}$, where β represents any positive integer. For $n \geq 2$, any two station pair is susceptible to the previous pattern when one "corner" is ignored by the static scheduling algorithm.

Fix any algorithm X in STATIC_MAXIMUM. If X schedules the lower left corner of the backlog matrix $\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix}$ then it must schedule the upper left corner of the backlog matrix $\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$ or an infinite cell backlog can occur as in the two station example. The following two lists of backlog matrices have the element inside the parentheses representing the maximum matching that algorithm X uses every time that particular backlog matrix occurs. Once the first matching is fixed, the others are implied in order to avoid the infinite cell backlog pattern in the two station example. Recall that the matching algorithm used by X considers only whether each element is zero or non-zero.

$$\begin{pmatrix} (1) & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & 0 \\ (1) & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 1 & 0 \\ 0 & (1) & 0 \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & (1) & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} (1) & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & 0 \\ (1) & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & (1) \\ 0 & 0 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 1 & (1) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

There is a contradiction between the last matching in each set so the theorem is proved, since some two station pair will admit the infinite backlog pattern of the previous example. □

Proof of Theorem 6. The proof is by induction on time k with the following proposition.

Proposition P_k: Any line backlog is zero in at least one slot during any S consecutive slots ending by time $k + S$.

Property P4 means that P_k is true for all $k \in Z$. Clearly P_k is true for $k < 0$ since the system starts empty. Fix $k \geq 0$. Suppose P_{k-1} is true. Fix a line, row i, without loss of generality. It is shown that $R_i(l) = 0$ for some $l \in [k : k + S)$. By the induction hypothesis, $R_i(l) = 0$ for some $l \in [k-1 : k+S-1)$. If $R_i(k-1) > 0$, then $R_i(l) = 0$ for some $l \in [k : k+S-1)$, and therefore for some $l \in [k : k+S)$. On the other hand, suppose $R_i(k-1) = 0$. Let n_c be the number of different row i cells to which packets arrive during slots $[k : k+S)$. The columns containing these cells are called *covered columns*. Note that $n_c \leq \alpha S$ and $\alpha \leq \frac{1}{3}$ for $S \geq 1$ by the theorem assumption. Also, $\| \mathbf{N}(k-1) \| \leq \alpha S$ by Proposition P_{k-1} and the (α, S) constraint. Apply Lemma 2 to see that at least $C_j(k-1)$ packets depart from any covered column j during slots $[k-1 : k+3\alpha S-1)$. At most $n_c \alpha S$ packets arrive to covered columns (including all cells in row i) during slots $[k : k+S)$. Since at least one packet departs from some covered column in each slot until row i is empty, $R_i(l) = 0$ for some $l \in [k : k + 3\alpha S + n_c \alpha S)$. Since $n_c \leq \alpha S$ and $3\alpha S + (\alpha S)^2 \leq S$ (because $\alpha \leq \frac{\sqrt{4S+9}-3}{2S}$), $R_i(l) = 0$ for some $l \in [k : k+S)$. Thus, P_k is true and Theorem 6 is proved by induction. □

10 Appendix C—Proof of Large Propagation Delay Theorem 7

Fix any receiver. The key to the proof of Theorem 7 is to show that phase 1 is successful with high probability. That is the purpose of the next lemma. Let G be any possible bipartite graph constructed for phase 1 by the receiver as described above. Let $\mathcal{U} = (U_1, \ldots, U_n)$ be a partition of U, where U_i is the set of nodes corresponding to packets transmitted by source i. The phase 1 transmission slot selection algorithm constructs G from $\mathcal{U}$ and V. Let $P_{\mathcal{U}}$ be the probability that a maximum matching on G does not cover U, given a partition $\mathcal{U}$.

Lemma 3 *For* $\alpha \leq \frac{1}{4}$,

$$\max_{\mathcal{U}:|U| \leq \alpha M} P_{\mathcal{U}} \to 0 \text{ as } M \to \infty$$

Lemma 3 will be proved after a technical lemma is presented. Fix any $A \subset U$. The edge set $E^A = E \cap (A \times V)$ contains only edges from E which have a vertex in A. Analysis of the matching used in phase 1 reception is facilitated by the introduction of sets E^0 and E^1, where E^1 has the same distribution as E^A, and the sets are constructed as follows. Define A_i to be the subset of A corresponding to source i packets. Each packet $u \in A$ chooses any three elements $v_1(u), v_2(u), v_3(u) \in V$, uniformly at random, independently of all other packets in

A. Let all such selections determine E^0, so that $E^0 = \{(u, v_l(u)) : u \in A, l = 1,2,3\}$. Note that the vectors $(v_1(u), v_2(u), v_3(u))$ and set E^0 are similar to the transmission slot vectors of (s_1, s_2, s_3) and set E^A, but that *source conflicts* can occur in E^0. A source conflict occurs when for some i, there is more than one edge from A_i to some v. This violates the basic model constraint of one packet/source/slot. The set E^0 is now modified for each $v \in V$ to remove all *source conflicts.* The heads of all but one of the conflicting edges are moved to new nodes in V. Each new node is chosen uniformly at random from all nodes which does not create a new source conflict. E^1 is the modified set with all source conflicts removed. The construction of E^0 and E^1 is complete, and it is clear that E^A and E^1 have the same distribution. Figure 7 illustrates an example of the construction of E^1 from E^0.

For any edge set E, define the *shadow* of A as $S_E(A) = \{v \in V : (u,v) \in E, u \in A\}$. Clearly, $S_{E^A}(A) = S_E(A)$. Fix any $B \subset V$. Since E^A and E^1 have the same distribution, $P\{S_E(A) \subset B\} = P\{S_{E^1}(A) \subset B\}$. Also, since $S_{E^0}(A) \subset S_{E^1}(A)$ by construction, $P\{S_{E^1}(A) \subset B\} \leq P\{S_{E^0}(A) \subset B\}$. Therefore the following lemma holds.

Lemma 4 $P\{S_E(A) \subset B\} \leq P\{S_{E^0}(A) \subset B\}, \forall A \subset U, \forall B \subset V.$

Proof of Lemma 3. Since $P_{\mathcal{U}}$ is increasing in $\mathcal{U}$, it is enough to show that

$$\max_{\mathcal{U}:|U|=\alpha M} P_{\mathcal{U}} \to 0 \text{ as } M \to \infty$$

Hall's Marriage Theorem allows $P_{\mathcal{U}}$ to be written in terms of the shadows of subsets of U. For arbitrary $\mathcal{U}$ with $|U| = \alpha M$,

$$\begin{aligned}
P_{\mathcal{U}} &= P\{\exists A \subset U, \exists B \subset V \text{ with } |B| = |A| - 1 \text{ and } S_E(A) \subset B\} \\
&\leq \sum_{A,B:|B|=|A|-1} P\{S_E(A) \subset B\} \text{ by union bound} \\
&\leq \sum_{A,B:|B|=|A|-1} P\{S_{E^0}(A) \subset B\} \text{ by Lemma 4} \\
&= \sum_{i=1}^{\alpha M} \binom{\alpha M}{i} \binom{\frac{3}{4}M}{i-1} \left(\frac{i}{\frac{3}{4}M}\right)^{3i} \\
&\leq \sum_{i=1}^{\alpha M} \left(\frac{\alpha M e}{i}\right)^{i} \left(\frac{\frac{3}{4}Me}{i}\right)^{i} \left(\frac{i}{\frac{3}{4}M}\right)^{3i} \text{ by Stirling's bound} \\
&= \sum_{i=1}^{\alpha M} \left(\frac{16\alpha e^2 i}{9M}\right)^{i}
\end{aligned}$$

$$= \frac{C}{M} + \sum_{i=2}^{\alpha M} \left(\frac{Ci}{M}\right)^i \text{ where } C = \frac{16}{9}\alpha e^2$$

The ratio of consecutive terms in the last sum above is $\frac{C}{M}\left(\frac{i+1}{i}\right)^i (i+1)$. This ratio is positive and increasing in i, so the maximum term in the last sum is either the $i = 2$ term or the $i = \alpha M$ term. Therefore,

$$P_{\mathcal{U}} \leq \frac{C}{M} + (\alpha M - 1) \max\left\{\left(\frac{2C}{M}\right)^2, (C\alpha)^{\alpha M}\right\}$$

Since $C\alpha < 1$ and $\mathcal{U}$ is arbitrary, the lemma follows. □

Proof of Theorem 7. Lemma 3 shows that the probability of an arbitrary packet being successful in phase 1 can be made arbitrarily close to 1 for large enough M. If successful, a packet's access delay is no more than $d_{acc} + \frac{3}{4}M$. If not, then a packet is scheduled in phase 2 and suffers additional delay no more than $d_{prop} + d_{sync} - \frac{1}{2}M$.

Fix any $\epsilon > 0$. Choose M large enough so that

$$\max_{\mathcal{U}:|U|=\alpha M} \leq \frac{\epsilon}{3},$$

which is possible by Lemma 3 and suppose that $d_{prop} \geq \frac{9M}{\epsilon}$. Consider any (α, S) arrival sequence and let p be an arbitrary packet in the sequence. Equation (10) implies that all packets that arrive during the same frame as p (including p) are successful in phase 1 with probability at least $1 - \frac{\epsilon}{3}$. The expected access delay of packet p is then

$$\begin{aligned} d_p &\leq E[\text{phase 1 delay}] + \frac{\epsilon}{3}E[\text{additional phase 2 delay}] \\ &\leq d_{acc} + \frac{3}{4}M + \frac{\epsilon}{3}\left(d_{prop} + d_{sync} - \frac{1}{2}M\right) \end{aligned}$$

Recall that $d_{sync} \leq \frac{3}{4}M$ and $d_{acc} \leq 2M$, so that

$$d_p \leq d_{prop}\left(\frac{3M}{d_{prop}} + \frac{\epsilon}{3} + \frac{\epsilon M}{12 d_{prop}}\right)$$

The last three terms are smaller than or equal to $\frac{\epsilon}{3}$ and therefore $d_p \leq \epsilon d_{prop}$. This completes the proof of Theorem 7. □

If source conflicts are ignored, then Lemma 3 holds for $\alpha \leq \frac{1}{2}$ according to [3]. The proof technique used there does not allow the application of Lemma 4. It is conjectured here that Lemma 3 is true for $\alpha \leq \frac{1}{2}$ with source conflicts removed, since removal only increases the size

References

[1] M.-S. Chen, N. R. Dono, and R. Ramaswami. A media-access protocol for packet-switched wavelength division multiaccess metropolitan area networks. *IEEE Journal on Selected Areas in Communications*, 8:1048–1057, August 1991.

[2] R. L. Cruz. A calculus for network delay, part I:network elements in isolation. *IEEE Transactions on Information Theory*, 37:114–131, January 1991.

[3] R. L. Cruz and B. Hajek. Global load balancing by local adjustments. In *Proceedings of the 19th Annual Conference on Information Sciences and Systems*, pages 448–454. Johns Hopkins University, 1986.

[4] M. de Prycker. *Asynchronous Transfer Mode.* Ellis Horwood, New York, NY, 1991.

[5] A. Descloux. Stochastic models for ATM switching networks. *IEEE Journal on Selected Areas in Communications*, 9:450–457, April 1991.

[6] N. R. Dono, P. E. Green, K. Liu, R. Ramaswami, and F. F.-K. Tong. A wavelength division multiple access network for computer communication. *IEEE Journal on Selected Areas in Communications*, 8:983–994, August 1991.

[7] I. S. Gopal, G. Bongiovani, M. A. Bonuccelli, D. T. Tang, and C. K. Wong. An optimal switching algorithm for multibeam satellite systems with variable bandwidth beams. *IEEE Transactions on Communications*, 30:2475–2481, November 1982.

[8] I. S. Gopal and C. K. Wong. Minimizing the number of switchings in an SS/TDMA system. *IEEE Transactions on Communications*, 33:497–501, June 1985.

[9] M. Hall, Jr. *Combinatorial Theory.* Blaisdell, Waltham, MA, 1969.

[10] T. Inukai. An efficient SS/TDMA time slot assignment algorithm. *IEEE Transactions on Communications*, 27:1449–1455, October 1979.

[11] J. Kurose. On computing per-session performance bounds in high-speed multi-hop computer networks. In *Sigmetrics 1992*, pages 128–139, New Port, Rhode Island, June 1992. ACM.

[12] C. L. Liu. *Topics in Combinatorial Mathematics.* Mathematic Association of America, 1972.

[13] C. Papadimitriou and K. Steiglitz. *Computational Complexity.* Prentice Hall, Englewood Cliffs, NJ, 1989.

[14] R. Ramaswami. Multiwavelength lightwave networks for computer communication. *IEEE Communications Magazine*, 31:78–88, February 1993.

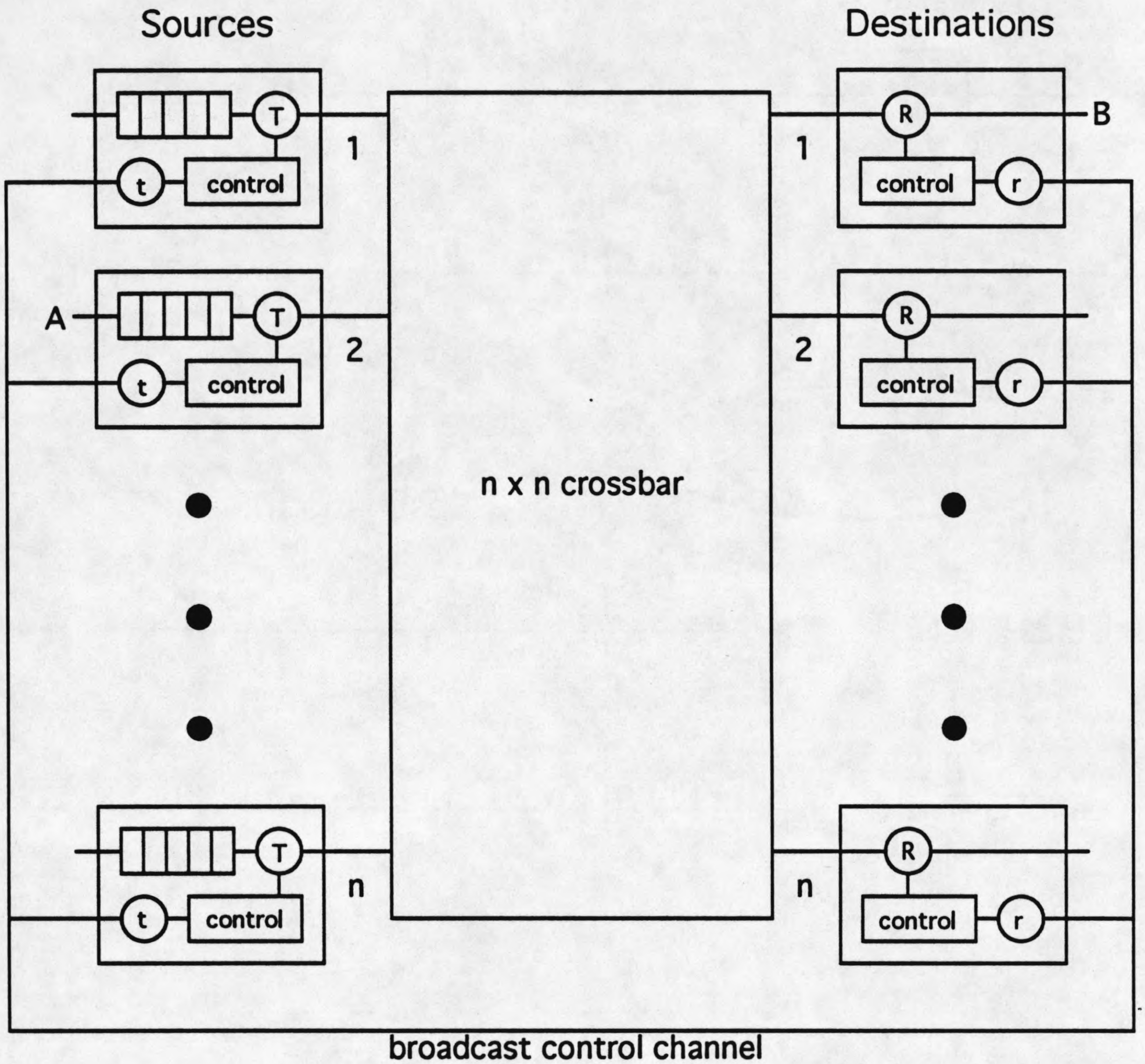
[15] R. Ramaswamy and P. Dhar. Comments on "An efficient SS/TDMA time slot assignment algorithm". *IEEE Transactions on Communications*, 32:1061–1065, September 1984.

[16] E. Shamir and E. Upfal. One factor in random graphs based on vertex choice. *Discrete Mathematics*, 41:281–286, 1982.

[17] D. W. Walkup. Matchings in random regular bipartite digraphs. *Discrete Mathematics*, 31:59–64, 1980.

[18] T. Weller. *Scheduling Non-Uniform Traffic in a Multi-Access Communication System.* PhD thesis, University of Illinois at Urbana–Champaign, 1993. In preparation.

[19] L. Zhang. *A New Architecture for Packet Switching Network Protocols.* PhD thesis, Massachussetts Institute of Technology, 1989.



T - data transmitter
R- data receiver
t - control channel transmitter
r - control channel receiver

Figure 1 - The Basic Model

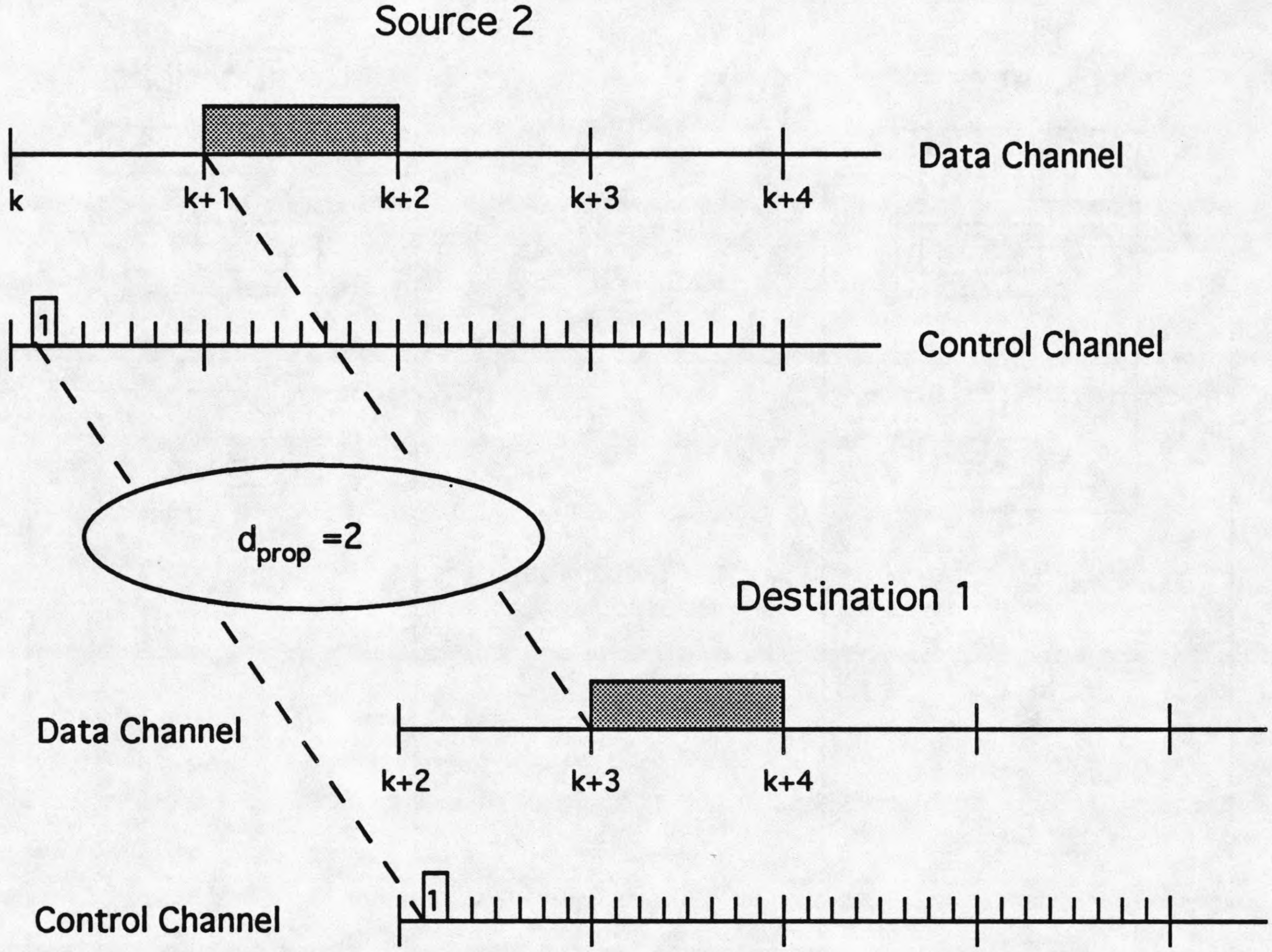


Figure 2 - Channel Timing Diagram

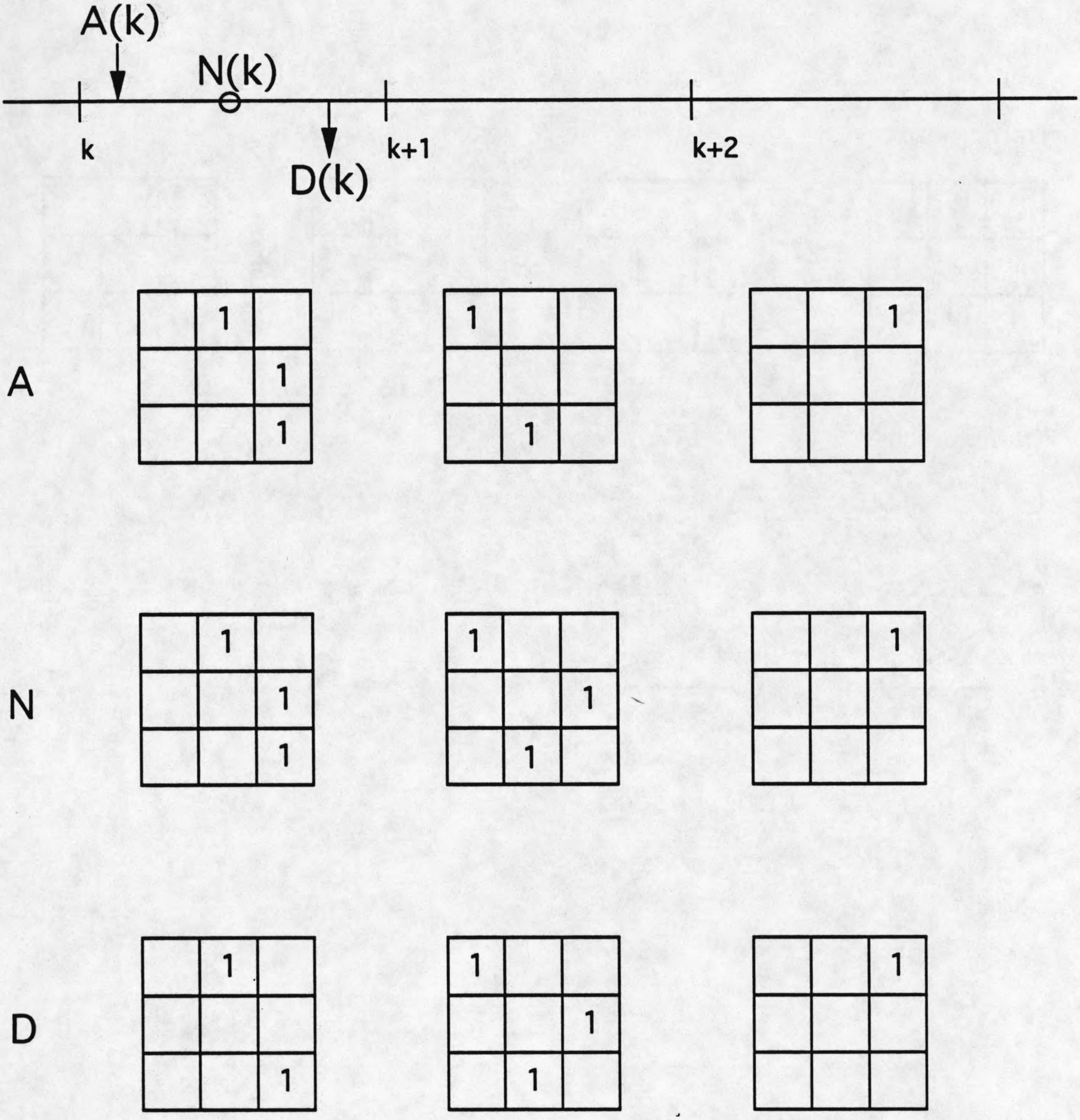


Figure 3 - Matrix Sequences

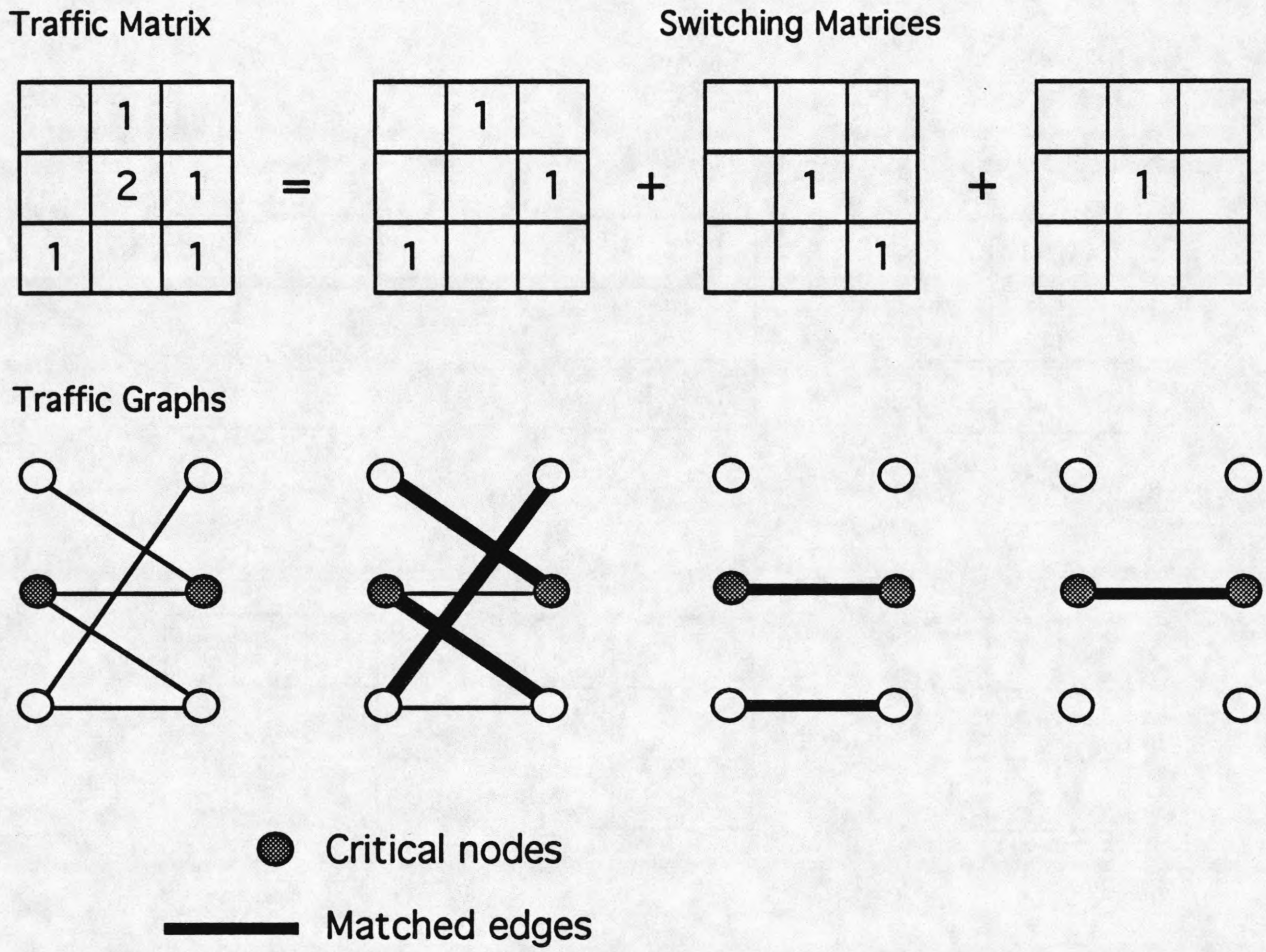


Figure 4 - Time Slot Assignment Example

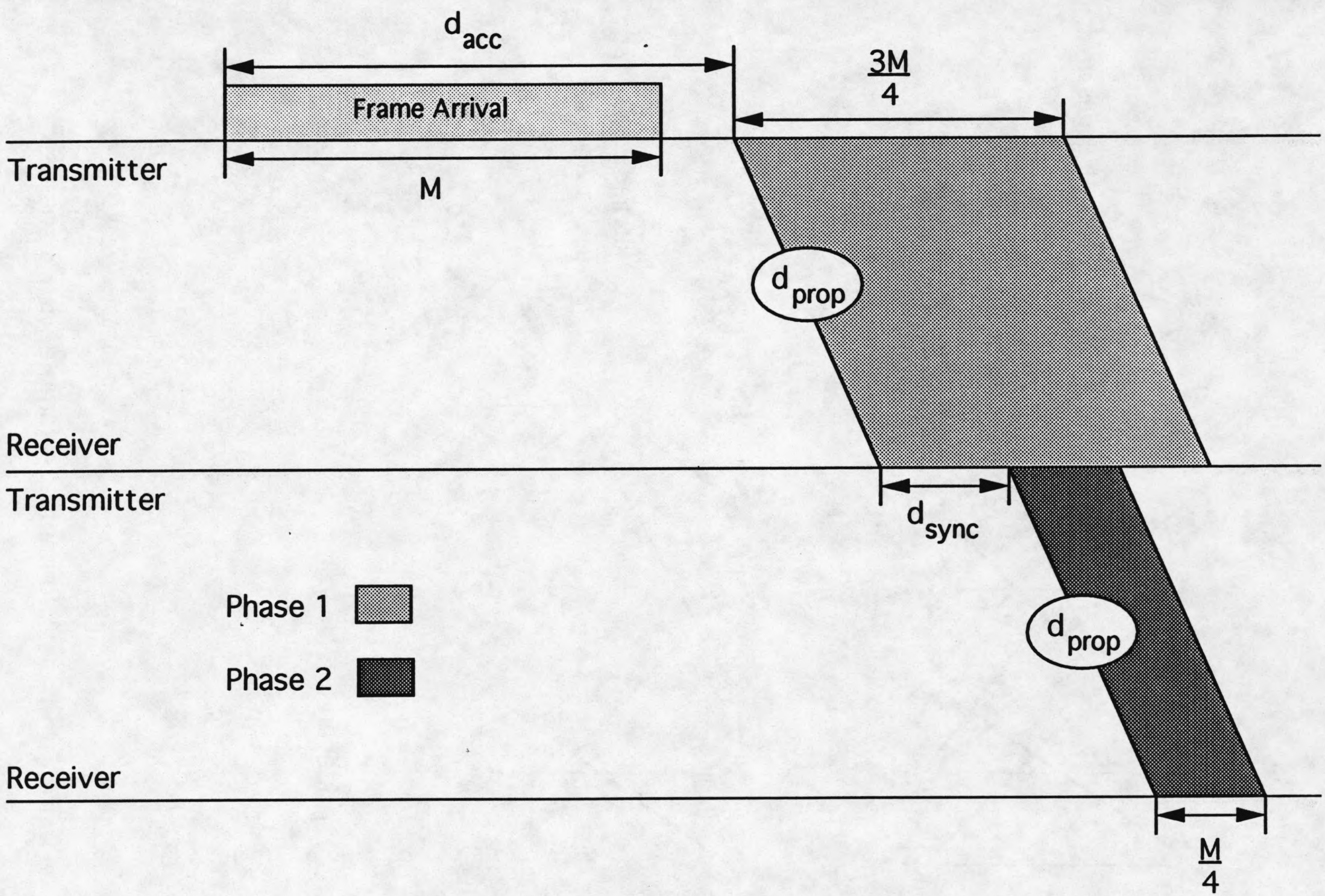


Figure 5 - Large Propagation Delay Algorithm Timing

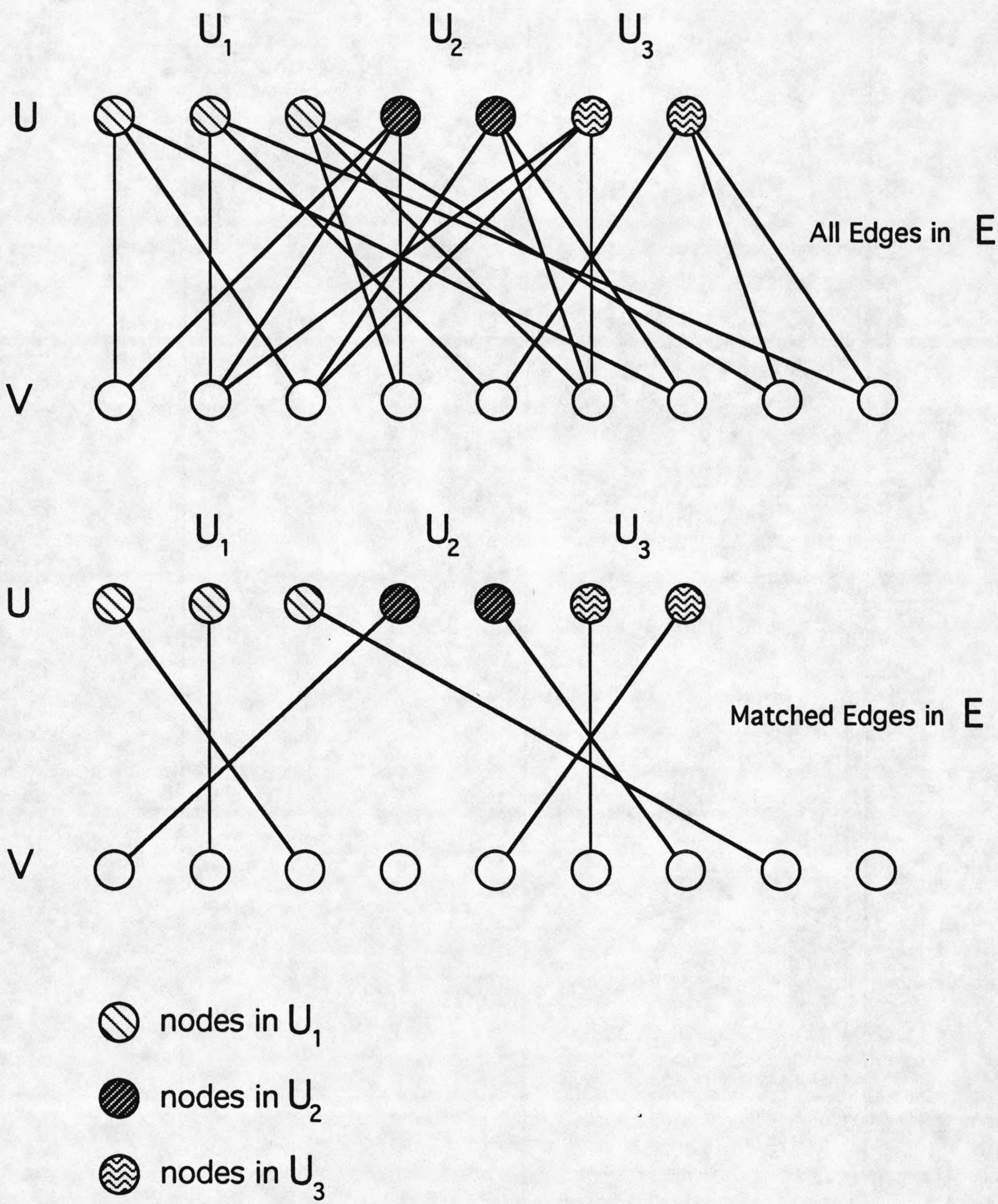


Figure 6 - Phase 1 Reception example with 3 sources

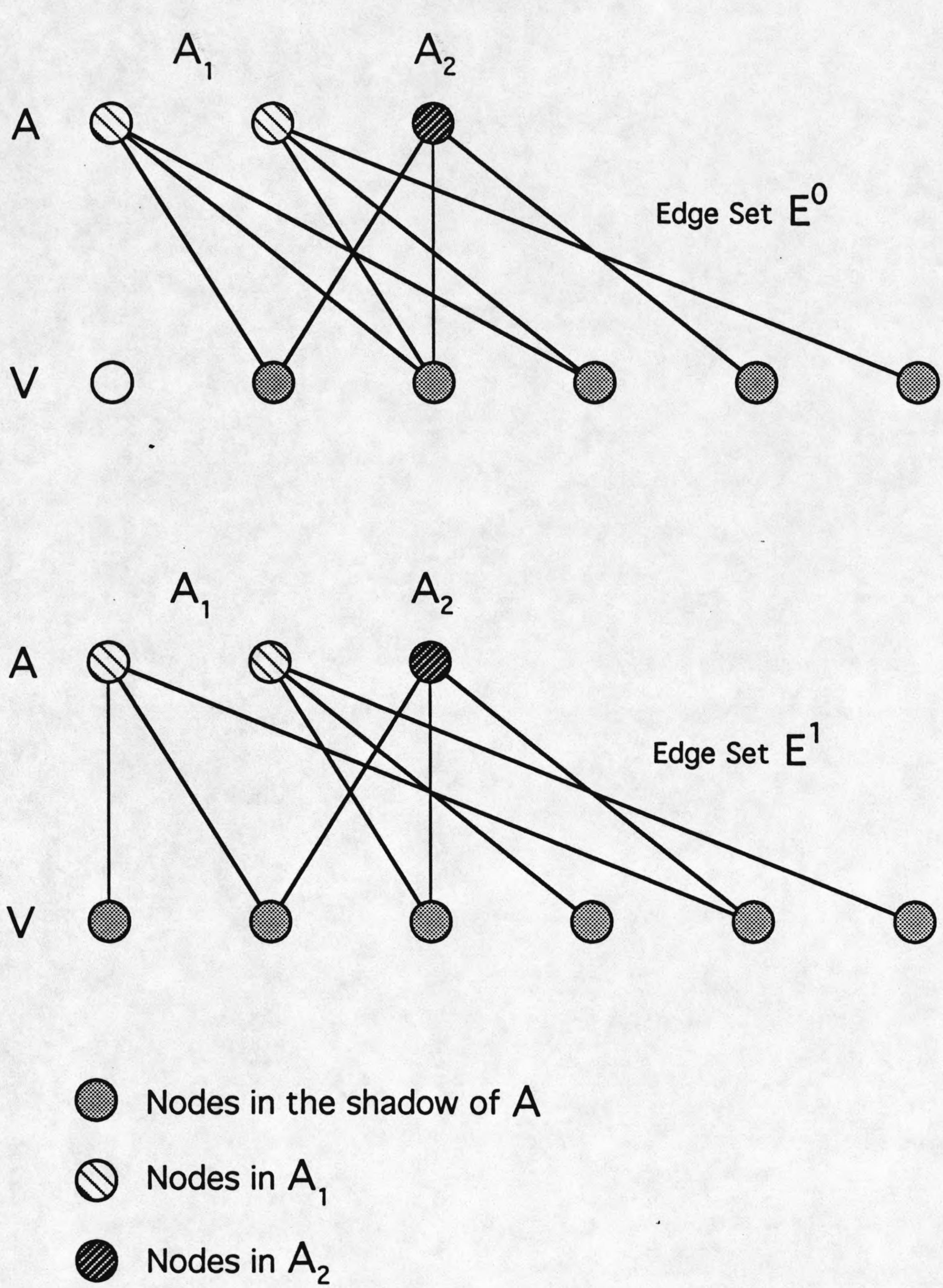


Figure 7 - Example constructing E^1 from E^0 to remove source conflicts