On the Ability of Establishing Real-Time Channels in Point-to-Point Packet-Switched Networks

Qin Zheng, *Member, IEEE*, and Kang G. Shin, *Fellow, IEEE*

Abstract — **There are numerous applications which require packets to be delivered within pre-specified delay bounds in point-to-point packet-switched networks. To meet this requirement, we define a real-time channel as a unidirectional connection between two nodes in such a network that guarantees every packet to be delivered before a user-defined, end-to-end deadline.**

The goal of this paper is to lay a mathematical basis for the problem of establishing real-time channels by (i) deriving a necessary and sufficient condition for the schedulability of a set of channels over a link, and (ii) developing an efficient method for calculating the minimum delay bound over a link for each channel. Given the traffic characteristics of a channel, our results can be used to check whether or not every packet will be delivered within a pre-specified delay bound. The results are also applicable to a wide variety of real-time task scheduling problems.

I. Introduction

CURRENT packet-switched networks provide users with two basic types of communication service: datagrams and virtual circuits. Typical examples of these two types are Internet User Datagram Protocol (UDP) and Internet Transmission Control Protocol (TCP). The datagram service provides connectionless, unreliable communications between two hosts where each data unit (datagram) is sent independently and there is no guarantee that the datagrams will ever get delivered or delivered correctly. The virtual circuit service, on the other hand, provides connection-oriented, reliable communications, and guarantees all packets to be delivered correctly and in sequence. Each type of service has its own application domains. Datagrams are suitable for short and/or urgent communications because no connection establishment procedure is needed, while virtual circuits are more suitable for those applications requiring reliable and sequenced delivery of packets.

An important feature that both datagrams and virtual circuits do not support is the guaranteed timely delivery of packets. There are, however, numerous applications — such as interactive voice/video communications, time-constrained remote operations, and real-time control/monitoring — that require all packets to be delivered within pre-specified delay bounds. For these applications, there is a need for a third type of service — called a *real-time channel* — which guarantees the timely, sequenced delivery of packets from a source host/node to a destination host/node.

The need for real-time channels was first noted in [1], stating that in addition to TCP, at least two other types of transport layer protocols are desirable: (i) a speech protocol guaranteeing sequenced, timely delivery of messages without considering the reliability in message delivery, and (ii) a real-time protocol guaranteeing both timeliness and high reliability. The concept of real-time channel belongs to the first type of protocol, in which the reliability issue is not considered. Note that the high reliability of TCP is achieved by retransmission of packets, which is usually too time-consuming to be useful for real-time applications.

An easy way to implement a real-time channel would be to use the circuit-switching technique. Given the user's maximum traffic generation behavior, one can use a dedicated circuit with an adequate bandwidth between two hosts to guarantee the timely delivery of all packets. If a link's bandwidth is greater than that a single channel requires, several channels can be established over the link using either *time-division multiplexing* (TDM) or *frequency-division multiplexing* (FDM) techniques, as is usually done for telecommunication networks. These multiplexing techniques, however, do not exploit the bursty nature of data traffic (thus resulting in severe under-utilization of link capacity) and the different requirements of traffic types (thus resulting in an inflexible allocation of link bandwidth).

The packet-switching technique, on the other hand, uses the link capacity more efficiently and flexibly since it allocates the link bandwidth dynamically according to the traffic demands. However, most current packet-switched networks use First-Come-First-Serve (FCFS) or Round-Robin (RR) scheduling policies for packet transmissions at each transmission link. These are not suitable for real-time applications since urgent packets should be given priority over non-urgent ones.

The increase of the switching node's processing power with advanced processor technology has enabled more sophisticated switching techniques to be used. Ferrari and Verma [2] proposed to use a deadline scheduling policy instead of the FCFS or RR policy. Each arriving packet is assigned a deadline according to its requested delivery delay bound. The packet with the earliest deadline is transmitted first. The deadline scheduling policy was proved to be optimal [3] in the sense that if packets can be transmitted

Paper recommended by Jonathan M. Smith, the Editor for Experimental Systems & Applications of the IEEE Communication Society. Manuscript received July 16, 1991; revised January 22, 1992 and March 31, 1992. This work was supported in part by the Office of Naval Research under Grant N00014-92-J-1080, and the National Science Foundation under Grant MIP-9203895.

Qin Zheng is with Mitsubishi Electric Research Laboratories, Inc., Cambridge Research Center, Cambridge, MA 02139. Kang G. Shin is with the Real-Time Computing Laboratory, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2122.

IEEE Log Number 9401007.

before their deadlines using any scheduling policy, so can they using the deadline scheduling policy. Thus, using the deadline scheduling policy can enhance a link's ability to accommodate real–time channels.

Several ways of implementing the deadline scheduling policy at a switching node were discussed in [4,11,12]. However, as pointed out in [5], one problem in using deadline scheduling is the difficulty in computing guarantees. There are no known efficient solutions to the *schedulability problem*: given a set of real–time channels, can all packets in these channels be delivered before their requested delay bounds? It is essential to solve the schedulability problem when real–time channels are to be established.

Ferrari and Verma [2] obtained a solution to the schedulability problem under the assumption that the summation of the maximum packet transmission times over all real–time channels passing through a link is not larger than the minimum packet inter-arrival times of these channels. This assumption is quite restrictive in practice since it limits the traffic types to be serviced. Without using this assumption, Kandlur *et al.* [5] established a sufficient condition to check the schedulability of channels. They first derived the schedulability conditions from the fixed-priority scheduling policy. Since any set of channels which are schedulable under fixed-priority scheduling are also schedulable under deadline scheduling, this condition is a sufficient schedulability condition for the deadline policy.

It can be proved that under the assumption of [2], the sufficient condition in [5] is equivalent to the sufficient condition in [2]. So, the result in [5] subsumes that in [2]; that is, [5] can deal with situations where the assumption of [2] fails to hold. However, using sufficient schedulability conditions for establishing real–time channels may still under–utilize the network's transmission capacity since a violation of the sufficient conditions does not necessarily mean that the channels cannot be established.

The goal of this paper is thus to obtain a "true" schedulability condition that is both necessary and sufficient without any assumption about the traffic types to be serviced. Using this condition, the network's transmission capacity can be best utilized to accommodate real–time channels. We will also derive an efficient means of computing the minimum delay over a link for each real–time channel running through it. All results are obtained under both preemptive and non-preemptive scheduling policies.

This paper is organized as follows. Section II states the problem formulation and discusses how the solutions to this problem could be used to establish real–time channels. Solutions under the preemptive and non-preemptive deadline scheduling policies are given and discussed in Section III and Section IV, respectively. The paper concludes with Section V.

II. PROBLEM FORMULATION

A real–time channel is defined to be uni-directional. A bi-directional real–time channel can be created by setting up two separate uni-directional channels. The reason for this is that the traffic patterns of two directions could be significantly different. So it is more efficient to consider one direction at a time.

Since packet delays are easier to control with fixed–route packet switching than with dynamic routing, the former is used to establish real–time channels. Thus, one needs to establish a channel before using it. The channel establishment procedure includes the following steps:

1. Routing: select a source–destination route for the channel. All packets of the real–time channel will be sent over this route.
2. Performance verification: check if the selected route satisfies the delivery delay requirements of the channel. The establishment of a new channel should not affect the performance guarantees of the existing channels.
3. Connection confirm or denial: if the performance verification test passes, then the real–time channel can be established. Otherwise, the connection request is denied. The user requesting a channel to be established should either change the delay requirements or request the channel to be established later.

The above procedure can be performed in a decentralized [2] or centralized [6,12] manner. The reader is referred to [6,12] for implementation details of the channel establishment procedure. We will in this paper focus on the performance verification, i.e., checking whether the requested end–to–end delivery delay bound along a given route can be guaranteed or not.

The end–to–end packet delivery delay is the summation of delays over links and nodes along the selected route, which are composed of:

- Switching delays: the time needed to move a packet from an input link (or the application process at the source node) to an appropriate output link buffer (or the application process at the destination node).
- Queueing delays: the waiting time due to contention at the transmitter of an output link.
- Transmission and propagation delays: the time needed to transmit the packet and the time for the packet to reach the next node.

The switching delay depends on the switching architecture and technique used by the node. According to the discussion in [7], the data transfer speed inside a switching node is usually much faster than that of an output link, and thus, the switching delays are negligible as compared to the other two elements. In other words, we assume a non-blocking output–queueing model of the switching node.

The transmission and propagation delays of a packet are easy to determine. The former depends on the size of the packet and the transmission rate, and the latter on the length of each link in the route. Since the propagation delay is constant for a given route, it can be subtracted from the requested end–to–end delivery delay bound. So, we will consider only the transmission delays for the third element listed above.

In order to calculate the queueing delay, however, one has to consider (i) the characteristics of packet generation at the source node, and (ii) the scheduling policy used for each link. Because of the limited transmission capacity of each link, no bounded queueing delay can be guaranteed unless the source node specifies its pattern of traffic generation. In other words, as was assumed in [2, 5] a user requesting a real-time channel to be established must specify two parameters, T and C, describing the traffic characteristics, where T is the minimum packet inter-arrival time and C is the maximum packet-transmission time over a link.[1] C/T determines the maximum traffic load generated by the channel. Note that without any knowledge of these parameters, it is impossible to guarantee bounded queueing delays. Moreover, it is reasonable to assume this knowledge in many applications, such as interactive voice/video transmission and real-time control/monitoring. A user may exceed his pre-specified maximum packet generation rate at the risk that these packets may be delivered with delays longer than the pre-specified bound or may even be discarded. (The system is, of course, not responsible for honoring the client's requests which do not conform to the *a priori* agreed terms.)

As discussed earlier, the deadline scheduling policy is used for real-time channels because of its optimality. In order to employ this deadline scheduling policy, one must assign a deadline to each packet queued at a transmission link. Here we assume that each packet arrived at time t is assigned a deadline $t+d$, where d is called the *requested delay bound* (RDB) over the link which is either given by the client requesting the channel or determined by a channel establishment algorithm.

Based on the discussions thus far, we can now formally define schedulability problems. A real-time channel τ running through a link is described by a 3-tuple (T, C, d), where T is the minimum packet inter-arrival time at the link, C is the maximum packet-transmission time over the link, and $t+d$ is the deadline assigned to a packet arrived at time t. Without loss of generality, T, C, and d are assumed to be all positive.

A set of channels $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n$, is said to be *schedulable* over a link if for all $1 \leq i \leq n$, the maximum delay (queueing delay plus transmission delay) experienced by channel i's packets over the link is not greater than the requested delay bound d_i.

We define the following two problems related to channel schedulability:

Problem A: Given a set of n channels $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n$, are they schedulable over a link?

Problem B: Suppose $n-1$ channels, $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n-1$, are schedulable over a link. Given a new channel τ_n with the minimum packet inter-arrival time T_n and the maximum packet transmission time C_n, what is the minimum value of d_n such that all $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n$, are still schedulable over the link?

The following remarks need to be made about the above problem definitions.

[1]C equals the maximum packet length divided by the transmission rate of the link. So, it varies over links with different transmission rates.

1. Problems A and B deal with performance verification over a single link only. Recall that a real-time channel needs to guarantee an end-to-end deadline. There are two ways to use the solutions of Problems A and B for checking end-to-end deadline guarantees.
   - The client requesting the channel divides the end-to-end delay into smaller values, one for each link in the route and performs the schedulability check for each link (Problem A). The delay allocation algorithm may need information about link capacities, current link loads, link costs, and so on. If all the checks are positive, the real-time channel can be established successfully. Otherwise, the client must reallocate the link delays or select another route, and repeat the procedure.
   - The solution of Problem B is used to calculate the minimum packet delay bound for each of the links on the route. If the summation of these minimum delay bounds is not greater than the requested end-to-end delay, the real-time channel can be established successfully. Otherwise, it is impossible to establish the channel at this time unless the client chooses an alternative route or increases the requested end-to-end delay bound.
2. The description of a channel over a link with a 3-tuple (T, C, d) means that the packet inter-arrival times at the link are not smaller than a constant T. This is true at the first link of the channel if the source node abides by the traffic generation constraints. The packet inter-arrival times at intermediate links, however, may be smaller than T because of the different delays experienced by the packets at the previous links. In this case, a *logical* packet arrival time t' — which is defined as the time the packet would have arrived at the link if it had experienced the largest delays (i.e., the requested delay bounds) at all previous links — should be used to compute the packet's deadline, $t' + d$. Using its logical packet arrival time will not exceed the packet's end-to-end delay bound. Also, under heavy traffic conditions, using the logical arrival time will lower the priority of a packet which has gained time over the previous links (thus arrived earlier than normal) and let other tighter-deadline packets be transmitted first. Under light traffic conditions, using the logical arrival time will not delay the transmission of any packet, because the packet can be transmitted before its logical arrival time. (See [5] for a detailed account of this.) At the source node, the similar logical generation time can be used to deal with the situation when a client violates his pre-specified traffic constraints. The packets that are generated earlier than agreed upon are assigned the same deadlines as if they were generated according to the agreement. These early packets may suffer larger delays or may even discarded, but they will not affect the guarantees of other packets. Thus, establishing real-time channels is also an efficient means of flow control [2].

3. Application of the solutions to Problems A and B is not limited to real–time channel establishment. A link can represent any time–critical resource like a processor, and channels can represent any semi-periodic[2] time–critical tasks. So, our solutions can also be used for (semi–)periodic task scheduling problems as those discussed in [3]. Actually, the problems addressed in this paper are more general than the ones in [3] since we do not require the deadline d to be equal to the task period T.

Readers are referred to [2,5,12] for more discussions on the above issues.

III. PREEMPTIVE SCHEDULING

We derive solutions to Problems A and B for the preemptive deadline scheduling policy in this section. Under this policy, a packet with the earliest deadline is always transmitted first. When a packet with an earlier deadline arrives, the transmission of the current packet is preempted and it will be resumed after all packets with earlier deadlines are transmitted. The time needed for the preemption and resumption of transmission is assumed to be negligible. All results obtained in this section will be extended to non-preemptive deadline scheduling policy in the next section.

Define a function $\lceil x\rceil^+ = n$ if $n-1 \le x < n,\ n = 1,2,...,$ and $\lceil x\rceil^+ = 0$ for $x < 0$. Then we have the following solution to Problem A.

Theorem 1: A set of n channels $\tau_i = (T_i, C_i, d_i),\ i = 1,2,...,n$, are schedulable over a link under the preemptive deadline policy if and only if

$$\forall t \ge 0,\ \sum_{i=1}^{n} \lceil (t-d_i)/T_i \rceil^+ C_i \le t,$$

Proof of the necessary condition: Let 0 be the starting time for the system. In other words, the system is empty (of packets) at time $t = 0$. Then, $\forall t > 0$, a necessary condition for no packets to miss their deadlines in $[0,t]$ is that the amount of time, Γ, needed to transmit all those packets arrived during $[0,t]$ with deadlines $\le t$ is not greater than t. Since the minimal packet inter-arrival time of channel i is T_i, there are at most $\lceil (t-d_i)/T_i\rceil^+$ packets arrived over channel i during $[0,t]$ with deadlines $\le t$, which need at most $\lceil (t-d_i)/T_i\rceil^+ C_i$ units of time to transmit. Thus, the maximum value of Γ is $\sum_{i=1}^{n} \lceil (t-d_i)/T_i\rceil^+ C_i$. This proves the necessary condition.

Proof of the sufficient condition: We prove this by contradiction. Suppose a packet misses its deadline at time t_1, meaning that at least one packet with deadline $\le t_1$ has not been transmitted over the link by t_1 (a packet is said to have been transmitted when the last bit of the packet leaves the transmitting node). Then, from the property of a preemptive deadline scheduling policy, there must exist $t' < t_1$ such that during the time period $[t', t_1]$, the node is busy transmitting only those packets with deadlines $\le t_1$. Let t_0 be the smallest such t', then there are no packets with deadlines $\le t_1$ queued for the link at time t_0^-. Thus, it is concluded that in the time period $[t_0, t_1]$, the link is busy transmitting only those packets which arrive at the link during the time period $[t_0, t_1]$ and having deadlines $\le t_1$. Based on the same reasoning as the proof of the necessary condition, the maximum amount of time needed to transmit these packets is $\Gamma = \sum_{i=1}^{n} \lceil (t_1 - t_0 - d_i)/T_i\rceil^+ C_i$. Since one packet misses its deadline at t_1, this Γ must be larger than $t_1 - t_0$, that is,

$$\sum_{i=1}^{n} \lceil (t_1 - t_0 - d_i)/T_i \rceil^+ C_i > t_1 - t_0.$$

By letting $t = t_1 - t_0$, the above inequality contradicts the condition that $\forall t \ge 0,\ \sum_{i=1}^{n} \lceil (t-d_i)/T_i\rceil^+ C_i \le t$. □

An interesting special case of Theorem 1 is when $d_i = T_i$ for all $1 \le i \le n$. Since $\lceil (t-T_i)/T_i\rceil^+ \le t/T_i$, the inequality of Theorem 1 is satisfied if the maximum utilization of the link, $\sum_{i=1}^{n} C_i/T_i$, is not greater than 1. Also, it is easy to see that the maximum utilization of the link does not exceed 1 is a necessary condition for the schedulability of channels. Thus, we conclude that when the requested delay bounds are equal to the minimum packet inter-arrival times, the channels are schedulable over the link if and only if the maximum link utilization does not exceed 1. This is the well-known result reported in [3] for the periodic task scheduling problem.

Two additional properties about the schedulability of channels over a link follow immediately from Theorem 1.

1. Increasing the requested delay bounds d_i's will not affect the schedulability of channels.
2. A new channel can always be added if its requested delay bound is large enough and the total utilization of the link does not exceed 1.

These two properties are stated formally in the following corollary.

Corollary 1:

(1) Suppose a set of channels $\tau_i = (T_i, C_i, d_i),\ i = 1,...,n$, are schedulable over a link, then for any $d_i' \ge d_i$, the set of channels $\tau_i' = (T_i, C_i, d_i'),\ i = 1,...,n$, are also schedulable over the link.

(2) Suppose $n-1$ of channels $\tau_i = (T_i, C_i, d_i),\ i = 1,...,n-1$, are schedulable on a link, then by adding one more channel $\tau_n = (T_n, C_n, d_n)$, the set of n channels $\tau_i = (T_i, C_i, d_i),\ i = 1,...,n$, are also schedulable on the link if the maximum link utilization $\sum_{i=1}^{n} C_i/T_i < 1$ and $d_n \ge T_n + t_n$, where $t_n = \max\{d_1, ..., d_{n-1}, (\sum_{i=1}^{n-1}(1 - d_i/T_i)C_i + C_n)/(1 - \sum_{i=1}^{n-1} C_i/T_i)\}$.

Proof: (1) is a direct result of Theorem 1.

To prove (2), using Theorem 1 we need to show that $\forall t \ge 0,\ \sum_{i=1}^{n} \lceil (t-d_i)/T_i\rceil^+ C_i \le t$. For $t < d_n$, $\sum_{i=1}^{n} \lceil (t - d_i)/T_i\rceil^+ C_i = \sum_{i=1}^{n-1} \lceil (t-d_i)/T_i\rceil^+ C_i$. Since the first $n-1$

[2]In the sense that there exists a minimum task inter-arrival time.

channels are schedulable over the link, from Theorem 1, the right–hand side of the above equation is less than, or equal, to t. Thus, $\sum_{i=1}^{n}\lceil(t-d_i)/T_i\rceil^+C_i \le t$ for $t < d_n$.

For $t \ge d_n$,

$$\begin{aligned}\sum_{i=1}^{n}\lceil(t-d_i)/T_i\rceil^+C_i &\le \sum_{i=1}^{n}(1+(t-d_i)/T_i)C_i\\ &\le \sum_{i=1}^{n-1}(1-d_i/T_i)C_i+(C_n/T_n)t_n\\ &\quad +(\sum_{i=1}^{n}C_i/T_i)t\end{aligned}$$

From the definition of t_n, we have

$$\sum_{i=1}^{n-1}(1-d_i/T_i)C_i \le (1-\sum_{i=1}^{n-1}C_i/T_i)t_n.$$

Thus,

$$\sum_{i=1}^{n}\lceil(t-d_i)/T_i\rceil^+C_i \le t_n + (\sum_{i=1}^{n}C_i/T_i)(t-t_n).$$

Since $\sum_{i=1}^{n}C_i/T_i < 1$, the inequality $\sum_{i=1}^{n}\lceil(t-d_i)/T_i\rceil^+C_i \le t$ follows. □

Theorem 1 has a neat mathematical form and is useful for deriving properties about the schedulability of channels. However, one may find it difficult to use for solving Problem A in practice, because the inequality of Theorem 1 is supposed to be checked over an infinite length interval $[0,\infty)$. Two observations can resolve this difficulty. First, the left–hand side of the inequality is a piece-wise constant function. Thus, we only need to check the inequality at some discrete points in $[0,\infty)$. Second, there exists a point t_{max} such that under the condition that the total utilization of the link $\sum_{i=1}^{n}C_n/T_n < 1$, the inequality of Theorem 1 always holds for $\forall\, t \ge t_{max}$. So, we only need to consider a *finite* set of points to validate the inequality of Theorem 1. To this end, we have the following theorem which is a practically–realizable version of Theorem 1.

Theorem 2: A set of channels $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n$, are schedulable over a link with the preemptive deadline scheduling policy if and only if both of the following hold:

1. $\sum_{i=1}^{n}C_i/T_i \le 1$.
2. $\forall t \in S$, $\sum_{i=1}^{n}\lceil(t-d_i)/T_i\rceil^+C_i \le t$ where $S = \cup_{i=1}^{n}S_i$, $S_i = \{d_i + nT_i : n = 0, 1, ..., \lfloor(t_{max}-d_i)/T_i\rfloor\}$, and $t_{max} = \max\{d_1, ..., d_n, (\sum_{i=1}^{n}(1-d_i/T_i)C_i)/(1-\sum_{i=1}^{n}C_i/T_i)\}$.

Proof: The first condition is easy to prove as follows. $\sum_{i=1}^{n}C_i/T_i$ is the maximum total utilization by all channels, and if it is greater than 1, there is no way that these channels are schedulable over the link.

To prove the second condition, we use Theorem 1. Since the value of $\lceil(t-d_i)/T_i\rceil^+C_i$ changes only on the set $S_i' = \{d_i + nT_i : n = 0, 1, ...\}$, we only need to check the inequality of Theorem 1 on the set $S' = \cup_{i=1}^{n}S_i'$.

Furthermore, $\lceil(t-d_i)/T_i\rceil^+ \le 1 + (t-d_i)/T_i$ for all $t \ge \max\{d_i : i = 1, ..., n\}$. So, it is easy to verify that for $t \ge t_{max}$, the inequality of Theorem 1 always holds. So, we only need to check the inequality on the set $S' \cap \{t : t \le t_{max}\}$, which is the set S. □

Example 1: Given three channels $\tau_1 = (T_1, C_1, d_1) = (10, 2, 5)$, $\tau_2 = (T_2, C_2, d_2) = (8, 4, 8)$, and $\tau_3 = (T_3, C_3, d_3) = (12, 3, d_3)$. Use Theorem 2 to check their schedulability for $d_3 = 9$ and $d_3 = 8$.

<u>*Solution*</u>: First we check that the total utilization

$$\sum_{i=1}^{3}C_i/T_i = 0.95 < 1.$$

For $d_3 = 9$, $t_{max} = 35$ from Theorem 1. Then, $S_1 = \{5, 15, 25, 35\}$, $S_2 = \{8, 16, 24, 32\}$, $S_3 = \{9, 21, 33\}$, and $S = S_1 \cup S_2 \cup S_3$. It is easy to verify that $\forall t \in S$, $\sum_{i=1}^{3}\lceil(t-d_i)/T_i\rceil^+C_i \le t$. Thus, the channels are schedulable with $d_3 = 9$.

Similarly, for $d_3 = 8$, $t_{max} = 40$. Then $S_1 = \{5, 15, 25, 40\}$, $S_2 = \{8, 16, 24, 32, 40\}$, $S_3 = \{8, 20, 32\}$, and $S = S_1 \cup S_2 \cup S_3$. At $t = 8$, $\sum_{i=1}^{3}\lceil(t-d_i)/T_i = 9$. Thus, the inequality of Theorem 2 is not satisfied. We conclude that the channels cannot be scheduled with $d_3 = 8$. □

Using the first result of Corollary 1, we can conclude that in the above example, 9 is the minimum integer value of d_3 such that the three channels are schedulable. So, if one wants to establish channel τ_3 over a link on which τ_1 and τ_2 already exist, the minimum requested delay bound he/she can ask is 9. We then want to know if there is an efficient way to find this minimum requested delay bound. This is Problem B stated in Section 2.

Given $n-1$ real–time channels which are schedulable over a link, we need to determine the minimum value of d_n for an n–th channel such that all the n channels are still schedulable over the link. To do this, we may first set $d_n = C_n$ and use Theorem 2 to check whether the channels are schedulable or not. If they are, C_n is the minimum delay bound for channel n. Otherwise, d_n must be increased according to the extent that the inequality of Theorem 2 is violated. This idea leads to the following theorem which solves Problem B under a preemptive deadline scheduling policy.

Theorem 3: Let $f(t, d_n) = \sum_{i=1}^{n}\lceil(t-d_i)/T_i\rceil^+C_i$ and S be the set defined in Theorem 2 with $d_n = C_n$. Then, $d_n = C_n$ is the solution to Problem B if $\forall t \in S$, $f(t, C_n) \le t$. Otherwise, the solution to Problem B is $d_n = \max\{d^t : t \in G\}$, where $G = S \cap \{t : f(t, C_n) > t\}$ and d^t is computed as $d^t = C_n + k_f^tT_n + \epsilon_f^t + \epsilon_t^t$, with $k_f^t = \lceil(f(t, C_n) - t)/C_n\rceil - 1$, $\epsilon_f^t = f(t, C_n) - t - k_f^tC_n$, $\epsilon_t^t = t - C_n - k_t^tT_n$, $k_t^t = \lfloor(t - C_n)/T_n\rfloor$.

Proof: From Theorem 2, if $\forall t \in S$, $f(t, C_n) \le t$, then all n channels are schedulable over the link with the n–th channel choosing $d_n = C_n$. Since d_n can not be smaller than C_n, $d_n = C_n$ is the solution to Problem B.

Otherwise, we need to find a minimum d_n such that $\forall t \geq 0$, $f(t, d_n) \leq t$. Let $d_n = C_n + \delta d$ and $\delta f(t, \delta d) = \lceil (t - C_n - \delta d_n)/T_n \rceil^+ C_n - \lceil (t - C_n)/T_n \rceil^+ C_n$. Then, $f(t, d_n) = f(t, C_n) + \delta f(t, \delta d)$. For any $t_0 \in G$, define

$$f_{t_0}(t) = \begin{cases} 0 & \text{if } t < t_0 \\ f(t_0, C_n) & \text{if } t \geq t_0. \end{cases}$$

Then, $\forall t \geq 0$, $f(t, d_n) \leq t$ if and only if $\forall t \geq 0$, $\forall t_0 \in G$, $f_{t_0}(t) + \delta f(t, \delta d) \leq t$. So, we only need to find the minimum value of δd such that the latter inequality holds.

Let t_0 be a fixed point in G. We first find a minimum δd^{t_0} such that the inequality $f_{t_0}(t) + \delta f(t, \delta d^{t_0}) \leq t$ holds for all $t \geq 0$. Write $\delta d^{t_0} = k_d^{t_0} T_n + \epsilon_d^{t_0}$, $t_0 - C_n = k_t^{t_0} T_n + \epsilon_t^{t_0}$, with $k_d^{t_0} = \lfloor \delta d^{t_0}/T_n \rfloor$, $k_t^{t_0} = \lfloor (t_0 - C_n)/T_n \rfloor$. Since the first $n-1$ channels are schedulable over the link, we can restrict $\delta d^{t_0} \leq t_0 - C_n$. Under this condition, functions $f_{t_0}(t), \delta f(t, \delta d^{t_0})$ and t are plotted in Fig. 1. From Fig. 1, it is easy to see that $f_{t_0}(t) + \delta f(t, \delta d^{t_0}) \leq t$ holds for all $t \geq 0$ if and only if this inequality holds at $t = t_0$ and $t = t_1$, where $t_1 = t_0 + \epsilon_d^{t_0} - \epsilon_t^{t_0}$. Notice that $\delta f(t_0, \delta d^{t_0}) = -(k_d^{t_0} + 1) C_n$ and $\delta f(t_1, \delta d^{t_0}) = -k_d^{t_0} C_n$. Thus, $k_d^{t_0}$ and δd^{t_0} must satisfy the following two inequalities:

$$f(t_0, C_n) - (k_d^{t_0} + 1) C_n \leq t_0$$

$$f(t_0, C_n) - k_d^{t_0} C_n \leq t_1 = t_0 + \epsilon_d^{t_0} - \epsilon_t^{t_0}.$$

The values of $k_d^{t_0}$ and $\epsilon_d^{t_0}$ which satisfy the above inequalities and minimize $\delta d^{t_0} = k_d^{t_0} T_n + \epsilon_d^{t_0}$ are:

$$k_d^{t_0} = \lceil (f(t_0) - t_0)/C_n \rceil - 1 = k_f^{t_0}$$

$$\epsilon_d^{t_0} = f(t_0, C_n) - t_0 + \epsilon_t^{t_0} - k_d^{t_0} C_n = \epsilon_f^{t_0} + \epsilon_t^{t_0}.$$

The minimum δd^{t_0} thus obtained such that

$$\forall t \geq 0, \; f_{t_0}(t) + \delta f(t, \delta d^{t_0}) \leq t$$

is

$$\delta d^{t_0} = k_d^{t_0} T_n + \epsilon_d^{t_0} = k_f^{t_0} T_n + \epsilon_f^{t_0} + \epsilon_t^{t_0}.$$

Since $\delta f(t, \delta d)$ is a decreasing function of δd, the minimum value of δd that satisfies

$$f_{t_0}(t) + \delta f(t, \delta d) \leq t, \; \forall t \geq 0, \forall t_0 \in G$$

is thus

$$\delta d = \max\{\delta d^{t_0} : \; t_0 \in G\}.$$

This proves that $d = \max\{d^t : \; t \in G\}$ is the solution to Problem B. □

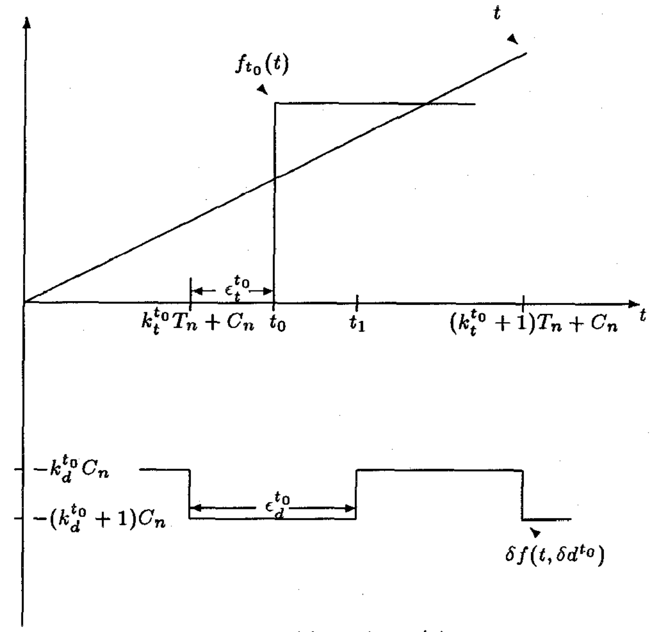

Fig. 1. Plots of $f_{t_0}(t), \delta f(t, \delta d^{t_0})$ and t .

Example 2: We want to establish a real-time channel $\tau_3 = (T_3, C_3, d_3) = (12, 3, d_3)$ over a link over which two other channels $\tau_1 = (T_1, C_1, d_1) = (10, 2, 5)$, $\tau_2 = (T_2, C_2, d_2) = (8, 4, 8)$ have already been established. What is the minimum value of d_3 such that all three channels are schedulable over the link?

Solution: The maximum utilization of the link

$$\sum_{i=1}^{3} C_i/T_i = 2/10 + 4/8 + 3/12 = 0.95 < 1$$

so such a d_3 exists. Using Theorem 3, first let $d_3 = C_3 = 3$. Then, $f(t, 3) = \lceil (t-5)/10 \rceil^+ 2 + \lceil (t-8)/8 \rceil^+ 4 + \lceil (t - 3)/12 \rceil^+ 3$. From the definitions in Theorem 2, $t_{max} = 65$ and

$$\begin{aligned} S_1 &= \{5, 15, 25, 35, 45, 55, 65\} \\ S_2 &= \{8, 16, 24, 32, 40, 48, 56, 64\} \\ S_3 &= \{3, 15, 27, 39, 51, 63\} \\ S &= S_1 \cup S_2 \cup S_3. \end{aligned}$$

It is easy to verify that the inequality $f(t, 3) \leq t$ holds over S except in $G = \{8, 16\}$ with $f(8, 3) = 9$ and $f(16, 3) = 18$. Using the formulae in Theorem 3, $d^8 = 9$ and $d^{16} = 6$. Thus, we get the solution $d_3 = \max\{d^t : t \in G\} = 9$. The correctness of the solution was verified in Example 1.

Notice that the result of [2] can not be used for this example since $C_1 + C_2 + C_3 = 9 > \min\{T_1, T_2, T_3\} = 8$. To compare the proposed scheme with that of [5], let us compute the worst-case delay for τ_3's packets from the priority scheduling policy. Since τ_3 is not supposed to affect the delay guarantees for τ_1 and τ_2's packets, its packets must be assigned the lowest priority.[3] Then, when packets of all

[3]Assigning τ_3's priority higher than τ_1 or τ_2 will cause one of τ_1 or τ_2's packet to miss its deadline when two packets, one for each channel, arrive simultaneously.

three channels (one per channel) arrive at the link simultaneously, the delay of τ_3's packet will be 15. Thus, the solution in [5] to Example 2 would be at least 15.[4] One can see that the gap between the sufficient condition of [5] and our sufficient and necessary condition for the schedulability of real-time channels is quite large, indicating the superiority of the latter. □

It is worth mentioning that our results remain true even if the requested delay bound d_i is larger than the minimum packet inter-arrival time T_i.

As mentioned in Section 2, if the summation of the minimum delay bounds over the links of a real-time channel — i.e., $D^* = \sum_{i=1}^{m} d_i^*$, where m is the number of links of the channel, d_i^* is the minimum delay bound (solution of Problem B) over link i — is not greater than the requested end-to-end delay D, the real-time channel can be successfully established. However, unless $D^* = D$, the minimum delay bound d_i^* should not be used directly to calculate the deadline of a packet for the run-time scheduling since doing this would guarantee the end-to-end delay bound to be D^* instead of D, which is too tight and may hinder the establishment of other channels in future. So, we need to use the minimum delay bound d_i^* to calculate a requested delay bound d_i such that $\sum_{i=1}^{m} d_i = D$. This is called the *Requested Delay Assignment Procedure* (RDAP). The main objective of RDAP is to free over-reserved link resources (i.e., increase d_i^* to d_i) in order to accommodate more channels in future. RDAP should depend on the topology of the network and traffic patterns. Intuitively, more resources should be freed over links in the "downtown" area where more requests for establishing channels over them are likely to occur. If no particular network topology or traffic patterns are known, a reasonable RDAP is to increase d_i's equally over links.

The complexity of the solutions to Problems A and B presented in Theorems 2 and 3 is of the order of the size of set S which, from its definition in Theorem 2, is reasonably small when the maximum link utilization $\sum_{i=1}^{n} C_i/T_i$ is not too close to one. However, as the link utilization $\sum_{i=1}^{n} C_i/T_i$ approaches one, S could become very large. One way to deal with this problem is to avoid the heavily-loaded links (e.g., with utilizations > 0.9) during the virtual circuit routing phase. This is based on the reasoning that links with high utilizations induce large delays, and thus, new real-time channels should not be added to them.

If the size of S is still thought to be too large even for moderately loaded links, one can use the following sufficient condition for the schedulability of n channels over a link that only needs to validate an inequality on at most n points.

A set of channels $\tau_i = (T_i, C_i, d_i), i = 1, 2, ..., n$, is said to be *ordered* if $d_1 \le d_2 \le \cdots \le d_n$.

Theorem 4: A set of ordered channels $\tau_i = (T_i, C_i, d_i)$, $i = 1, ..., n$, are schedulable over a link with the preemptive deadline scheduling algorithm if $\sum_{i=1}^{n} C_i/T_i < 1$, and

$$\forall k \in K, \ \sum_{i=1}^{k}(1 + (d_k - d_i)/T_i)C_i \le d_k,$$

where $K = \{1, 2, ..., n\} - \{k : d_k = d_{k+1}, 1 \le k \le n-1\}$.

Proof: We want to prove that satisfaction of the above inequality implies that of the condition of Theorem 1. Denote $\Delta(t) = \sum_{i=1}^{n} \lceil (t-d_i)/T_i \rceil^+ C_i - t$. Let $K = \{k_1, ..., k_m\}$ such that $k_1 < k_2 < \cdots < k_m$, and let $d_{k_0} = 0$, $d_{k_{m+1}} = \infty$. Then, $\forall t \in [d_{k_j}, d_{k_{j+1}})$,

$$\begin{aligned}
\Delta(t) &= \sum_{i=1}^{k_j} \lceil (t-d_i)/T_i \rceil^+ C_i - t \\
&\le \sum_{i=1}^{k_j} (1 + (t-d_i)/T_i)C_i - t \\
&= (\sum_{i=1}^{k_j} C_i/T_i - 1)t + \sum_{i=1}^{k_j}(1 - d_i/T_i)C_i \\
&\le (\sum_{i=1}^{k_j} C_i/T_i - 1)d_{k_j} + \sum_{i=1}^{k_j}(1 - d_i/T_i)C_i \\
&= \sum_{i=1}^{k_j}(1 + (d_{k_j} - d_i)/T_i)C_i - d_{k_j} \\
&\le 0.
\end{aligned}$$

Since the above inequality holds for $j = 0, 1, ..., m$, we have $\forall t \ge 0, \Delta(t) \le 0$. Thus the condition of Theorem 1 is satisfied. □

An interesting by-product of Theorem 4 is a sufficient condition for the schedulability of channels when d_i/T_i equals a constant θ, called the the *system hazard* [8], for all $1 \le i \le n$. Peng and Shin [8] proved that the schedulability condition is $\theta \ge \sum_{i=1}^{n} C_i/T_i$. Using Theorem 4, we have another sufficient condition using the system hazard:

$$\theta \ge \max_{k \in K}\{(1 + (1 - \sum_{i=1}^{k} C_i/T_i)/\sum_{i=1}^{k} C_i/T_k)^{-1}\}.$$

Since $C_i/T_k \le C_i/T_i$, Peng's condition in [8] implies ours, so it is not as tight as ours. The following example shows how much of improvement over Peng's condition can be achieved by using our sufficient condition:

- Task τ_1 : $T_1 = 4$, $C_1 = 2$,
- Task τ_2 : $T_2 = 16$, $C_2 = 4$.

Using Peng's condition, the minimum system hazard θ is $\sum_{i=1}^{2} C_i/T_i = 0.75$. From our condition, θ can be as small as 0.6. This shows the tightness (superiority) of the sufficient condition of Theorem 4.

It is worth pointing out that the simplicity of the conditions of Theorem 4 is achieved at the cost that they are only sufficient, but not necessary, conditions. Violation of sufficient conditions does not necessarily mean that the

[4]Actually, since 15 is larger than T_3, [5] cannot give a solution for Example 2.

channels are not schedulable over the link. However, in case the simplicity of the algorithm is more desirable than the tightness of the results, the sufficient conditions of Theorem 4 can be used for the establishment of real-time channels as described below.

A set of channels is said to be *strongly schedulable* over a link if the conditions of Theorem 4 are satisfied. Then, we have the following problem which is a modified version of Problem B.

Problem B': Suppose $n-1$ channels, $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n-1$, are strongly schedulable over a link. Given a new channel τ_n with minimum packet inter-arrival time T_n and maximum packet transmission time C_n, what is the minimum value of d_n such that all $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n$ are still strongly schedulable?

The following theorem solves Problem B'.

Theorem 5: Let $d_n = C_n$. If for $k = 1, \cdots, n$, $\delta_k = \sum_{d_i \le d_k}(1 + (d_k - d_i)/T_i)C_i - d_k \le 0$, then $d_n = C_n$ is the solution to Problem B'. Otherwise, let $K_G = \{k : \delta_k > 0\}$. The solution to Problem B' is $d_n = \max\{d_n^k : k \in K_G\}$, where $d_n^k = C_n + (T_n/C_n)\delta_k$ if $C_n + (T_n/C_n)\delta_k < d_k$, otherwise, $d_n^k = d_k + (C_n - d_k + (T_n/C_n)\delta_k)/(1 + (1 - \sum_{d_i \le d_k} C_i/T_i)(T_n/C_n))$.

We omit the proof of the Theorem 5 here since it follows the same idea as that of Theorem 3.

For the purpose of comparison, we re-do Example 2 with Theorem 5. First, set $d_3 = C_3 = 3$. Here only two points need to be checked: $\delta_1 = (1 + (d_1 - d_3)/T_3)C_3 + C_1 - d_1 = 0.5 > 0$, and $\delta_2 = (1 + (d_2 - d_1)/T_1)C_1 + (1 + (d_2 - d_3)/T_3)C_3 + C_2 - d_2 = 57/20 > 0$. Thus, $K_G = \{1, 2\}$. From Theorem 5, $d_3^1 = 5$ and $d_3^2 = 13.3$. So $d_3 = 14$ is the minimum (integer-valued) requested delay bound which can be assigned to channel τ_3 such that the three channels are strongly schedulable over the link.

This example shows that the result obtained from Theorem 5 is usually not as good as that obtained from Theorem 3. However, since the computational complexity of Theorem 5 is in the order of the number of channels to be scheduled, Theorem 5 is useful when a smaller channel establishment time is required.

In this example, the solution obtained from Theorem 5 is shown to be better than Kandlur's [5] (see Example 2). Although this cannot be said in general, our solution always requires less computation than Kandlur's.

IV. NON-PREEMPTIVE SCHEDULING

All the results obtained in the last section are based on the assumption that a preemptive deadline scheduling policy is used. Using a preemptive scheduling policy implies that the transmission of a packet may be interrupted and resumed later when another packet with a tighter deadline arrives. Unlike task scheduling, this may be difficult to implement in practice, because it requires the in-progress transmission of packet to be aborted. So, it is important to study the use of a non-preemptive scheduling policy. The following theorem offers a simple means of checking the schedulability of channels over a link under a non-preemptive deadline scheduling policy.

Theorem 6: In the presence of non real-time packets, a set of real-time channels $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, \cdots, n$, are schedulable over a link under the non-preemptive deadline scheduling policy if and only if

$$\forall\, t \ge d_{min},\ \sum_{j=1}^{n} \lceil (t - d_j)/T_j \rceil^+ C_j + C_p \le t,$$

where $d_{min} = \min\{d_i : 1 \le i \le n\}$, and C_p is the time needed to transmit a maximum-size packet.

Proof of the necessary condition: We prove this by contradiction. Suppose the channels are schedulable over the link and there exists a $t_0 \ge d_{min}$ such that $\sum_{j=1}^{n} \lceil (t_0 - d_j)/T_j \rceil^+ C_j + C_p > t_0$. Consider the following scenario. The link has been idle for all $t < 0$. At $t = 0^-$, a maximum-size non real-time packet arrived at the link and then began transmitting at $t = 0$ (would last for C_p seconds). Starting from $t = 0$, all the real-time channels' messages are generated at their maximum rates. Then, the time needed to transmit all these real-time packets with deadlines $\le t_0$ is $\sum_{j=1}^{n} \lceil (t_0 - d_j)/T_j \rceil^+ C_j$. Since all n channels are schedulable over the link, these packets must be transmitted before time t_0, i.e.,

$$\sum_{j=1}^{n} \lceil (t_0 - d_j)/T_j \rceil^+ C_j + C_p \le t_0.$$

This contradicts the assumption.

Proof of the sufficient condition: We use contradiction again. Suppose the inequality of the theorem holds, but the channels are not schedulable over the link. Then, there exists a t_1 such that a real-time packet is being transmitted at time $t = t_1$ and misses its deadline. Let $t_0 \le t_1$ be the *earliest* time such that in the time interval $[t_0, t_1]$, the link is busy transmitting only those packets with deadlines $\le t_1$. Thus, at $t = t_0^-$, the link is either idle, transmitting a non real-time packet, or transmitting a real-time packet with deadline $\ge t_1$.

If the link is idle at $t = t_0^-$, then $t_1 - t_0 \ge d_{min}$ since a message generated after t_0 misses its deadline at t_1. Also, during the time interval $[t_0, t_1]$, the link is busy transmitting only those messages which are generated at the link during the time interval $[t_0, t_1]$ and with deadlines $\le t_1$. The maximum time needed to transmit these messages is $T = \sum_{j=1}^{n} \lceil (t_1 - t_0 - d_j)/T_j \rceil^+ C_j$. Since one message misses its deadline at t_1, this T must be larger than $t_1 - t_0$, that is,

$$\sum_{j=1}^{n} \lceil (t_1 - t_0 - d_j)/T_j \rceil^+ C_j > t_1 - t_0.$$

By letting $t = t_1 - t_0$, one can see that the above inequality contradicts the inequality of the theorem.

If the link is transmitting a non real-time message at $t = t_0^-$, then $t_1 - t_0 + C_p \geq d_{min}$ since a message generated after $t_0 - C_p$ misses its deadline at t_1. Also, during the time interval $[t_0, t_1]$, the link is busy transmitting only those messages generated at the link during the time period $[t_0 - C_p, t_1]$ (this corresponds to the worst-case that the non real-time packet has the maximum length) and having deadlines $\leq t_1$. The maximum time needed to transmit these messages is $T = \sum_{j=1}^{n} \lceil (t_1 - t_0 + C_p - d_j)/T_j \rceil^+ C_j$. Since there is a message missing its deadline at t_1, this T must be larger than $t_1 - t_0$, i.e.,

$$\sum_{j=1}^{n} \lceil (t_1 - t_0 + C_p - d_j)/T_j \rceil^+ C_j > t_1 - t_0.$$

By letting $t = t_1 - t_0 + C_p$, one can see that the above inequality contradicts the inequality of the theorem with $i = 0$.

Now suppose the link is transmitting a message belonging to a real-time channel i_0 at $t = t_0^-$. From the definition of t_0, this message must have been generated at the link at time $t_0^- - C_p$ (again, this corresponds to the worst-case that the real-time packet is of the maximum length) and has a deadline $> t_1$. Then, $d_{min} \leq t_1 - t_0 + C_p < d_{i_0}$. Also, during the time interval $[t_0, t_1]$, the link is busy transmitting only those messages generated at the link during the time period $[t_0 - C_p, t_1]$ and have deadlines $\leq t_1$. The maximum time needed to transmit these messages is $T = \sum_{j=1}^{n} \lceil (t_1 - t_0 + C_p - d_j)/T_j \rceil^+$. Since there is a message missing its deadline at t_1, this T must be larger than $t_1 - t_0$, i.e.,

$$\sum_{j=1}^{n} \lceil (t_1 - t_0 + C_p - d_j)/T_j \rceil^+ > t_1 - t_0.$$

By letting $t = t_1 - t_0 + C_p$, one can see the above inequality contradicting the inequality of the theorem. □

Results similar to Theorems 2 – 5 for the non-preemptive scheduling policy can be obtained from Theorem 6 as follows.

Theorem 7: In the presence of non real-time packets, a set of real-time channels $\tau_i = (T_i, C_i, d_i)$, $i = 1, 2, ..., n$, are schedulable over a link under the non-preemptive deadline-driven scheduling policy if and only if both of the following hold:

1. $\sum_{j=1}^{n} C_j/T_j \leq 1$.
2. $\forall t \in S$, $\sum_{i=1}^{n} \lceil (t - d_i)/T_i \rceil^+ C_i + C_p \leq t$, where $S = \cup_{i=1}^{n} S_i$, $S_i = \{d_i + nT_i : n = 0, 1, \cdots, \lfloor (t_{max} - d_i)/T_i \rfloor\}$, and $t_{max} = \max\{d_1, \cdots, d_n, (C_p + \sum_{i=1}^{n} (1 - d_i/T_i) C_i)/(1 - \sum_{i=1}^{n} C_i/T_i)\}$.

Theorem 7 allows us to check only a finite number of points to verify the schedulability of channels. The following theorem gives the solution to Problem B under the non-preemptive scheduling policy.

Theorem 8: Let $f(t, d_n) = \sum_{j=1}^{n} \lceil (t - d_j)/T_j \rceil^+ C_j + C_p$ and S be the set defined above with $d_n = C_n + C_p$. Then, $d_n = C_n + C_p$ is the worst-case delay if $f(t, C_n) \leq t$, $\forall t \in S$. Otherwise, the worst-case delay is $d_n = \max\{d^t : t \in G\}$, where $G = S \cap \{t : f(t, C_n) > t\}$ and d^t is computed as $d^t = C_n + k_f^t T_n + \epsilon_f^t + \epsilon_t^t$, with $k_f^t = \lfloor (f(t, C_n) - t)/C_n \rfloor$, $\epsilon_f^t = f(t, C_n) - t - k_f^t C_n$, $\epsilon_t^t = t - C_n - k_t^t T_n$, $k_t^t = \lfloor (t - C_n)/T_n \rfloor$.

Also, we have the sufficient conditions for the schedulability of channels under a non-preemptive scheduling policy.

Theorem 9: A set of ordered channels $\tau_i = (T_i, C_i, d_i)$, i 1, ..., n, are schedulable over a link with the non-preemptive deadline–driven algorithm if $\sum_{i=1}^{n} C_i/T_i < 1$ and

$$\sum_{i=1}^{k} (1 + (d_k - d_j)/T_j) C_j + C_p \leq d_k, \quad \forall k \in K,$$

where $K = \{1, 2, \cdots, n\} - \{k : d_k = d_{k+1}, 1 \leq k \leq n - 1\}$.

Similarly, a set of channels is said to be *strongly schedulable* under a non-preemptive deadline scheduling policy if the conditions of Theorem 9 are satisfied. The following theorem solves Problem B'.

Theorem 10: Let $d_n = C_n + C_p$. If for $i = 0, \cdots, n$, $k = 1, \cdots, n$,

$$\delta_k(i) = \sum_{d_j \leq \min\{d_i, d_k\}} (1 + (d_k - d_j)/T_j) C_j + C_i - d_k \leq 0$$

then $d_n = C_n + C_p$ is the solution to Problem B. Otherwise, let $K_G^i = \{k : \delta_k(i) > 0\}$. The solution to Problem B is

$$d_n = \max_{0 \leq i \leq n} \{\max\{d_n^k(i) : k \in K_G^i\}$$

where $d_n^k(i) = C_n + (T_n/C_n)\delta_k(i)$ if $C_n + (T_n/C_n)\delta_k(i) < d_k$, otherwise, $d_n^k = d_k + (C_n - d_k + (T_n/C_n)\delta_k(i))/(1 + (1 - \sum_{d_i \leq d_k} C_i/T_i)(T_n/C_n))$.

We omitted proofs of Theorems 7 – 10 since using Theorem 6, they are basically the same as that of Theorems 2 – 5.

V. CONCLUSION

We presented solutions to two fundamental problems associated with the establishment of real-time channels: (1) checking the schedulability of channels, and (2) computing the minimum delay bound over a link. Both preemptive and non-preemptive deadline scheduling policies are studied. The solutions give direct answers to the performance verification problem for the establishment of real-time channels.

We have not addressed the routing problem, which is the first step in establishing a real-time channel. One possible solution to this problem is to use the minimum delay bound obtained from Problem B as the *length* of a link and choose the shortest length route between the source and the destination. However, since such a link length depends on the number of real–time channels running through the link, we must coordinate the simultaneous establishment of several real–time channels if a distributed channel establishment procedure is used. Another remaining problem is error control. Transmission errors are inevitable in computer networks. If an error control protocol like ARQ is used at either the link–to–link or the end–to–end level, it would be very difficult to calculate the packet delivery delay since the acknowledgment and retransmission delays must also be considered. An even more difficult situation is when a link/node failure occurs, which means the source–to–destination route must be altered. Solutions to these problems are currently under development and will be reported in forthcoming papers. We are also implementing our algorithms on HARTS [9], an experimental distributed real–time system currently being developed at the Real–time Computing Laboratory, The University of Michigan.

ACKNOWLEDGMENT

The authors thank Parameswaran Ramanathan and Moncef Hamdaoui of University of Wisconsin–Madison for pointing out an error in a previous version of Theorem 6.

REFERENCES

[1] R. McFarland, “Protocols in a computer internetworking environment,” *Proceedings of EASCON 79*, 1979.

[2] D. Ferrari and D. C. Verma, “A scheme for real-time channel establishment in wide-area networks,” *IEEE J. Selected Areas Communication*, pp. 368–379, 1990.

[3] C. L. Liu and J. W. Layland, “Scheduling algorithms for multiprogramming in a hard-real-time environment,” *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[4] H. Saito, “Optimal queueing discipline for real–time traffic at ATM switching nodes,” *IEEE Transactions on Communications*, vol. 38, no. 12, pp. 2131–2136, 1990.

[5] D. D. Kandlur, K. G. Shin, and D. Ferrari, “Real–time communicaiton in multi-hop networks,” in *Proc. 11-th Int'l Conf. on Dist. Comput. Syst.*, pp. 300–307, May 1991. (An improved version is also to appear in *IEEE Transactions on Parallel and Distributed Systems.*)

[6] D. D. Kandlur and K. G. Shin, “Design of a communication subsystem for HARTS,” Technical Report CSE–TR–109–91, CSE Division, EECS Department, The University of Michigan, 1991.

[7] I. Cidon, I. Gopal, G. Grover, and M. Sidi, “Real-time packet switching: A performance analysis,” *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1576–1586, 1988.

[8] D. Peng and K. G. Shin, “A new performance measure for scheduling independent real-time tasks,” *Journal of Parallel and Distributed Computing*, vol. 19, no. 1, pp. 11–26, September 1993.

[9] M.-S. Chen, K. G. Shin, and D. D. Kandlur, “Addressing, routing and broadcasting in hexagonal mesh multiprocessors,” *IEEE Trans. Comput.*, vol. C–39, no. 1, pp. 10–18, January 1991.

[10] Q. Zheng and K. G. Shin, “Real–time communication in local area ring networks,” in *Conference on Local Computer Networks*, pp. 416–425, September 1992.

[11] Q. Zheng, “Real-time Fault-tolerant Communication in Computer Networks”, *PhD thesis*, University of Michigan, 1993. Available via anonymous ftp from ftp.eecs.umich.edu in directory outgoing/zheng.

Qin Zheng (S'89-M'94) received the B.S. and M.S. degrees in Electrical Engineering from the University of Science and Technology of China in 1982 and 1985, respectively, and the Ph.D. degree in Electrical Engineering from the University of Michigan in 1993.

He joined Mitsubishi Electric Research Labs, Inc., Cambridge Research Center in 1993 and is currently working on high-speed computer networks and distributed real-time industrial systems.

Kang G Shin (S'74-M'78-SM'84-F'92) is Professor of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, Michigan.

He has authored/coauthored over 250 technical papers (more than 110 of these in archival journals) and several book chapters in the areas of distributed real-time computing and control, fault-tolerant computing, computer architecture, robotics and automation, and intelligent manufacturing. In 1987, he received the Outstanding IEEE Transactions on Automatic Control Paper Award for a paper on robot trajectory planning. In 1989, he also received the Research Excellence Award from The University of Michigan. In 1985, he founded Real-Time Computing Laboratory, where he and his colleagues are currently building a 19-node hexagonal mesh multicomputer, called **HARTS**, to validate various architectures and analytic results in the area of distributed real-time computing.

He has also been applying the basic research results of real-time computing to manufacturing-related applications ranging from the control of robots and machine tools to the development of open architectures for manufacturing equipment and processes. Recently, he has initiated research on the open-architecture Information Base for machine tool controllers.

He received the B.S. degree in electronics Engineering from Seoul National University, Seoul, Korea in 1970, and both the M.S. and Ph.D. degrees in Electrical Engineering from Cornell University, Ithaca, New York in 1976 and 1978, respectively. From 1978 to 1982 he was on the faculty of Rensselaer Polytechnic Institute, Troy, New York. He has held visiting positions at the U.S. Airforce Dynamics Laboratory, AT&T Bell Laboratories, Computer Science Division within the Department of Electrical Engineering and Computer Science at UC Berkeley, and International Computer Science Institute, Berkeley, CA. He also chaired the Computer Science and Engineering Division at The University of Michigan for three years beginning 1991.

He is an IEEE fellow, was the Program Chairman of the 1986 IEEE Real-Time Systems Symposium (RTSS), the General Chairman of the 1987 RTSS, the Guest Editor of the 1987 August special issue of *IEEE Transactions on Computers* on Real-Time Systems, a Program Co-Chair for the 1992 *International Conference on Parallel Processing*, and served numerous technical program committees. He also chaired the IEEE Technical Committee on Real-time Systems during 1991-93, is a Distinguished Visitor of the Computer Society of the IEEE, an Editor of *IEEE Trans.on Parallel and Distributed Computing*, and an Area Editor of *International Journal of Time-Critical Computing Systems.*