SIAM J. COMPUT.
Vol. 10, No. 4, November 1981

0097-5397/81/1004-0005 $1.00/0

SOME COMPLEXITY RESULTS IN THE DESIGN OF DEADLOCK-FREE PACKET SWITCHING NETWORKS*

SAM TOUEG† AND KENNETH STEIGLITZ‡

Abstract. Deadlocks are very serious system failures and have been observed in existing packet switching networks (PSN's). Several problems related to the design of deadlock-free PSN's are investigated here. Polynomial-time algorithms are given for some of these problems, but most of them are shown to be NP-complete or NP-hard, and therefore polynomial-time algorithms are not likely to be found.

Key words. packet switching network, deadlock, complexity, NP-complete, flow control, routing, computer network, communication network

1. Basic definitions. A *packet switching network* (or PSN) is a directed graph $G = (V, E)$; the vertices V represent processors, and the edges E represent communication links. We assume messages, called *packets*, are to be passed between processors. Each vertex v_i has an associated constant b_i, the number of *buffers* at this vertex; a buffer can hold exactly one packet. Associated with each packet is an acyclic *route* $v_1, v_2, \cdots, v_q$, which is a path in G. Vertex v_1 is the *source*, and v_q is the *destination* vertex for the packet. We assume a *fixed routing procedure* [KL], where a packet's route is determined at the source node. We may also assume that the route of a packet is included as part of the message in the packet, although in practice the packet could hold only the source and destination, with each processor in the network responsible for deducing the next vertex to which the packet is to be passed.

The *moves* made by the network are of three types:

1. *Generation.* A vertex v creates a packet which is placed in an empty buffer of v.
2. *Passing.* A vertex v transfers a packet in one of its buffers to an empty buffer of vertex w, where $v \to w$ is an edge, and the route for the packet has w following v. The buffer of v holding the packet becomes empty.
3. *Consumption.* A packet in a buffer of v, such that the destination for the packet is v, is removed from that buffer and the buffer is made empty.

2. Flow control procedures. A *flow control procedure* (or *controller*) for a network is an algorithm that permits or forbids various moves in the network. One of the key problems in packet switching is preventing *deadlock states*, which are situations in which one or more packets can never make a move. Deadlock states have been observed in existing packet switching networks [KL]; they tend to occur under near-saturation input load [GHKP]. For example, in the network of Fig. 1, if all physically possible moves are permitted by the controller, v_1 generates b_1 packets with destination v_2, v_2 generates b_2 packets with destination v_3, and v_3 generates b_3 packets with destination v_1, then all buffers of all vertices will be full, no consumption moves can take place without a pass move, and no generation can take place. It is not hard to see that the network is deadlocked.

* Received by the editors July 11, 1979, and in final form October 29, 1980. This work was supported in part by the National Science Foundation under grant GK-42048, and in part by the U.S. Army Research Office, under grant DAAG29-75-0192.

† Department of Computer Science, Cornell University, Ithaca, New York 14853.

‡ Department of Electrical Engineering and Computer Science, Princeton University, Princeton, New Jersey 08544.

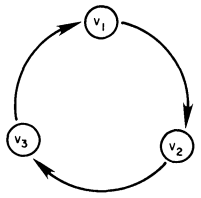

FIG. 1. *A network exhibiting deadlock with a trivial controller.*

However, if we use a controller that simply prohibits the generation (but not passing) of a packet into the last empty buffer at a vertex, then we can show that at least one empty buffer must exist somewhere in the network. Hence it is always possible to pass or consume some packet if there are any packets in the network, and this controller is deadlock-free.

In what follows, deadlock is assumed to occur with respect to some controller; that is, a controller is *deadlock-free* (or DF) for a given network if it does not permit this network to enter a state in which one or more packets can never make a move permitted by that controller.

3. Fundamental questions. We have assumed that each packet is generated with a fixed route to travel. There are still several options left to us:

1. Are we looking for a DF *uniform controller*, one that is deadlock-free for all networks? The design of optimal DF uniform controllers (optimal in the sense that they put the least restriction on moves) was investigated in [TU], [TO].
2. Are we designing a particular network with its flow control procedure such that this network is DF?

We investigate here the complexity of two deadlock-exposure problems related to the latter approach.

4. Deadlock-exposure problems. We consider the following two problems.

Problem 1. Given a network G and a set of source-destination routes in G, is the network *exposed to deadlock* (i.e., is there a deadlock state in G)?

Problem 2. Given a network G and a set of source-destination pairs of nodes in G, is there a corresponding set of routes in G such that the network is *not* exposed to deadlock?

The complexity of each of these two problems depends on the given buffer configuration of G and on the given flow control procedure applied in G. There are two possible buffer configurations for a network $G = (V, E)$. With a *general buffer configuration*, each node $v_i \in V$ has an individual buffer capacity b_i. With the *regular buffer configuration*, the buffer capacity b is the same for all the nodes $v \in V$. We consider four types of flow control procedures. The last three are commonly used in existing packet switching networks to avoid performance degradation under near-saturation input load.

(1) *Unrestricted flow control.* A node v accepts a packet p provided it has at least one empty buffer available for storing p. There are no other restrictions on packet moves.

(2) *Isarithmic flow control* [DA], [PR], [PRH]. With this form of flow control we have an additional restriction on the total number of packets that might be contained in the network at any given time. A packet can be generated in a node if this node has at least one empty buffer and the total number of packets in the network is less than a certain constant K.

(3) *Individual end-to-end window flow control* [GHKP], [KL], [PRH]. As in (1) with the following additional constraint. Each (v_i, v_j) source-destination pair has a constant k_{ij} associated with it; k_{ij} is an upper limit on the number of packets with source v_i and destination v_j. Such a packet can be generated in the node v_i provided v_i has at least one empty buffer and there are fewer than k_{ij} such packets in the network.

(4) *Regular end-to-end window flow control* [KL], [GHKP], [[RH]. As in (3) but k_{ij} is the same constant k for all the (v_i, v_j) source-destination pairs.

The complexity of Problem 1 and Problem 2 under the different buffer configurations and different flow control procedures[1] is summarized in Table 1 and Table 2.

TABLE 1
Time complexity of Problem 1.

Flow control applied	unrestricted	isarithmic: K	individual end-to-end: k_{ij}	regular end-to-end: k
General buffer conf.: b_i	Polynomial: $O(\|E\|)$	Polynomial: $O(\|V^3\|)$	NP-complete	NP-complete
Regular buffer conf.: b	Polynomial: $O(\|E\|)$	Polynomial: $O(\|V^3\|)$	NP-complete	NP-complete

TABLE 2
Time complexity of Problem 2.

Flow control applied	unrestricted	isarithmic: K	individual end-to-end: k_{ij}	regular end-to-end: k
General buffer conf.: b_i	NP-complete	NP-complete	NP-hard (in NP?)	NP-hard (in NP?)
Regular buffer conf.: b	NP-complete	NP-complete	NP-hard (in NP?)	NP-hard (in NP?)

THEOREM 1. *If the network $G = (V, E)$ has a general buffer configuration and the unrestricted flow control is applied, then Problem* 1 *is solvable in $O(|E|)$ time.*

Proof. It is easy to verify that such a network has a deadlock state if and only if the routes in this network form at least one cycle. A simple breadth-first-search along the routes of G may be used to detect such a cycle. □

COROLLARY 1. *If the network $G = (V, E)$ has a regular buffer configuration and the unrestricted flow control is applied, then Problem* 1 *is solvable in $O(|E|)$ time.*

Proof. Immediate consequence of Theorem 1. □

THEOREM 2. *If the network $G = (V, E)$ has a general buffer configuration and the isarithmic flow control (with constant K) is applied, then Problem* 1 *is solvable in $O(|V|^3)$ time.*

Proof. In such a network G there is a deadlock state if and only if the routes of G form at least one cycle such that the sum of the buffer capacities of the nodes along this cycle is at most K. A slight modification of Warshall's algorithm can be used to determine the *minimal cycle* formed by the routes in G (minimal in the sense that the

[1] In the formulation of these problems both the buffer configuration and the flow control procedure, with the corresponding constants, are given as part of the input.

sum of the buffer capacities of the nodes along this cycle is minimal with respect to all the other cycles). □

COROLLARY 2. *If the network $G=(V, E)$ has a regular buffer configuration and an isarithmic flow control is applied, then Problem* 1 *is solvable in $O(|V|^3)$ time.*

Proof. This is a particular case of Theorem 2. □

In [VA] an exponential-time algorithm is given to solve Problem 1 when G has a general buffer configuration and an individual end-to-end window flow control is applied. We now prove that, in this case, Problem 1 is NP-complete and, consequently, a polynomial-time algorithm is not likely to be found.

THEOREM 3. *If G has a general buffer configuration and an individual end-to-end window flow control is applied, then Problem* 1 *is* NP*-complete.*

Proof. Let $G=(V, E)$ be a network with a general buffer configuration and an individual end-to-end window flow control. In Problem 1 we ask if such a network has a deadlock state. This problem is in NP; we can guess a deadlock state and check its consistency with the given buffer configuration and end-to-end window flow control in polynomial time. We now show how to reduce (in polynomial time) the CNF-satisfiability problem [AHU] to Problem 1. Let $\mathbf{F}=\mathbf{F}_1\mathbf{F}_2\cdots\mathbf{F}_q$ be an expression in CNF, where the $\mathbf{F}_j$ are the factors. Let $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$ be the literals in $\mathbf{F}$. We construct the following network $G=(V, E)$. The nodes V of G are given by the set

$$V=\{v\}\cup\{F_j|\text{ for } 1\leqq j\leqq q\}\cup\{x_i, \bar{x}_i, w_i, \bar{w}_i|\text{ for } 1\leqq i\leqq n\}.$$

The edges of G are given by the following set

$$\begin{aligned}E=&\{(v, F_j)|\text{ for } 1\leqq j\leqq q\}\\&\cup\{(w_i, v), (\bar{w}_i, v)|\text{ for } 1\leqq i\leqq n\}\\&\cup\{(x_i, w_i), (\bar{x}_i, \bar{w}_i), (w_i, \bar{x}_i)|\text{ for } 1\leqq i\leqq n\}\\&\cup\{(F_j, x_i)|\ x_i \text{ is a variable in } F_j\}\\&\cup\{(F_j, \bar{x}_i)|\ \bar{x}_i \text{ is a variable in } F_j\}.\end{aligned}$$

The buffer capacity is $b=1$ for all the nodes except for the node v which has q buffers. The routes in the network are the following:

1. $\langle v, F_j\rangle$ for $1\leqq j\leqq q$.
2. $\langle x_i, w_i, \bar{x}_i, \bar{w}_i\rangle$, $\langle w_i, v\rangle$ and $\langle \bar{w}_i, v\rangle$ for $1\leqq i\leqq n$.
3. $\langle F_j, x_i\rangle$ if x_i is a variable in F_j.
4. $\langle F_j, \bar{x}_i\rangle$ if $\bar{x}_i$ is a variable in F_j.

The upper limit on the number of packets in each of these routes is set to one (this is a case of regular end-to-end window flow control with $k_{ij}=k=1$). We claim that this network has a deadlock state if and only if $\mathbf{F}$ is satisfiable. Suppose $\mathbf{F}$ is satisfiable. In each node F_j we generate a packet whose destination is the node x_i or $\bar{x}_i$ where $x_i=1$ or $\bar{x}_i=1$ is a variable assignment that makes the factor $\mathbf{F}_j$ true. In each such x_i or $\bar{x}_i$ node we put one packet whose route is $\langle x_i, w_i, \bar{x}_i, \bar{w}_i\rangle$. The next node in the route of this packet is either w_i or $\bar{w}_i$, and in such a node we generate a packet whose destination is the node v. Finally, in v we generate q packets, one in each of the $\langle v, F_j\rangle$ routes. A variable assignment that satisfies $\mathbf{F}$ cannot include both $x_i=1$ and $\bar{x}_i=1$; therefore no $\langle x_i, w_i, \bar{x}_i, \bar{w}_i\rangle$ route has a packet in both the x_i and $\bar{x}_i$ nodes; this is consistent with the end-to-end window flow control upper limit of one packet per route. It is easy to check that the network is in a deadlock state. Suppose now that the network has a deadlock state. Then there is at least one cycle of deadlocked packets in this state. Therefore, the node v must be in the deadlocked set of nodes, and there are q packets

filling all the buffers of v. Since we may have at most one packet in each of the $\langle v, F_j\rangle$ routes it must be that each $\langle v, F_j\rangle$ route has exactly one packet stored at the source node v. Therefore, each node F_j must also be deadlocked and must contain one packet whose destination is a node of the form x_i or $\bar{x}_i$. Then, this x_i or $\bar{x}_i$ node must also be deadlocked and its buffer must contain a packet in the $\langle x_i, w_i, \bar{x}_i, \bar{w}_i\rangle$ route. Since the upper limit on the number of packets in this route is one, it is not possible that both x_i and $\bar{x}_i$ are in the deadlocked set of nodes (therefore there can be no two F_j and F_k nodes with packets whose respective destination is x_i and $\bar{x}_i$). Let X be the set of x_i and $\bar{x}_i$ destination nodes for the packets in the F_j nodes. It is now clear that assigning the value 1 to all the variables in X is a consistent variable assignment which satisfies all the $\mathbf{F}_j$ factors. □

An example of a CNF Boolean expression, $\mathbf{F} = (\mathbf{x}_1 + \mathbf{x}_2)(\bar{\mathbf{x}}_1 + \mathbf{x}_3)$, of the corresponding network, of a variable assignment X satisfying $\mathbf{F}$, and of the corresponding deadlock state are shown in Fig. 2.

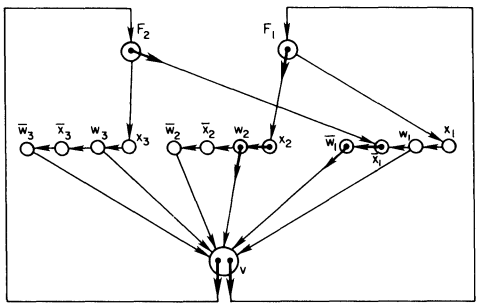

FIG. 2. *Network deadlock corresponding to a satisfiable* CNF *Boolean expression.*

In this case $\mathbf{F}_1 = (\mathbf{x}_1 + \mathbf{x}_2)$ and $\mathbf{F}_2 = (\bar{\mathbf{x}}_1 + \mathbf{x}_3)$. The corresponding network has the following routes:

1. $\langle F_1, x_1\rangle$, $\langle F_1, x_2\rangle$, $\langle F_2, \bar{x}_1\rangle$ and $\langle F_2, x_3\rangle$.
2. $\langle x_i, w_i, \bar{x}_i, \bar{w}_i\rangle$, $\langle w_i, v\rangle$ and $\langle \bar{w}_i, v\rangle$ for $i = 1, 2$ and 3.
3. $\langle v, F_j\rangle$ for $i = 1$ and 2.

All the nodes have a buffer capacity of one packet except the node v which has a buffer capacity of two packets. All the routes have an end-to-end window flow control of one packet per route. A variable assignment satisfying $\mathbf{F}$ is given by $X = \{\mathbf{x}_2, \bar{\mathbf{x}}_1\}$, (i.e., $\mathbf{x}_2 = \bar{\mathbf{x}}_1 = 1$). A corresponding deadlock state is the following. We have two packets in v with destination F_1 and F_2, a packet in F_1 with destination x_2, a packet in F_2 with destination $\bar{x}_1$, a packet in x_2 with destination w_2, a packet in $\bar{x}_1$ with destination $\bar{w}_1$, a packet in w_2 with destination v and a packet in $\bar{w}_1$ whose destination is also v.

COROLLARY 3. *If the network G has a general buffer configuration and a regular end-to-end window flow control is applied, then Problem* 1 *is* NP*-complete.*

Proof. In the proof of Theorem 3, the window flow control constant was globally set to $k_{ij} = k = 1$. □

THEOREM 4. *If the network G has a regular buffer configuration and a regular end-to-end window flow control is applied, then Problem* 1 *is* NP*-complete.*

Proof. The reduction of the CNF-satisfiability problem to Problem 1 given in Theorem 3 can be modified in the following way. Each node in G is now provided with q buffers, there is an additional set of $q-1$ nodes, $\{y_j |$ for $1 \leqq j \leqq q-1\}$, and except for the node v, all the nodes $w \in V$ have an additional set of $q-1$ edges directly connecting

them to the y_j nodes forming $q-1$ $\langle w, y_j\rangle$ new routes. Also, for each node y_j $(1 \leqq j \leqq q-1)$ there is an additional set of q edges directly connecting y_j to the nodes F_k, for $1 \leqq k \leqq q$, thus forming q $\langle y_j, F_k\rangle$ new routes. The end-to-end window flow control upper limit for all the routes is still one packet per route. We first prove that if $\mathbf{F}$ is satisfiable then the network has a deadlock state. We begin by constructing the network state described in the first part of Theorem 3. Then, in each node w $(w \neq v)$ that contains exactly one packet in this state, we generate $q-1$ additional packets, one packet for each of the new $\langle w, y_j\rangle$ $(1 \leqq j \leqq q-1)$ routes. Also, in each node y_j $(1 \leqq j \leqq q-1)$ we generate q packets, one for each one of the new $\langle y_j, F_k\rangle$ $(1 \leqq k \leqq q)$ routes. This is a deadlock state. We now show that if the network has a deadlock state then $\mathbf{F}$ is satisfiable. If the network has a deadlock state then there is at least one cycle of deadlocked packets. Therefore, a node

$$w \in \{v\} \cup \{y_j | 1 \leqq j \leqq q-1\}$$

must be in the deadlocked set of nodes, and there are q packets filling all the buffers of this node. Since we may have at most one packet in each of the $\langle w, F_j\rangle$ routes $(1 \leqq j \leqq q)$ it must be that each $\langle w, F_j\rangle$ route has exactly one packet stored at the source node w. Therefore, each node F_j must also be deadlocked and must contain q packets. Each one of the $\langle F_j, y_k\rangle$ routes $(1 \leqq k \leqq q-1)$ can have at most one packet, so F_j must contain at least one packet whose destination is a node of the form x_i or $\bar{x}_i$. Then this x_i or $\bar{x}_i$ node must also be deadlocked and its buffers contain q packets; at least one of these packets must be in the $\langle x_i, w_i, \bar{x}_i, \bar{w}_i\rangle$ route. From here the proof is identical to the last part of the proof of Theorem 3. □

COROLLARY 4. *If the network G has a regular buffer configuration and an individual end-to-end window flow control is applied, then Problem* 1 *is* NP*-complete.*

Proof. Theorem 3 shows that the problem is in NP and Theorem 4 shows that the problem is NP-hard. □

We now consider the complexity of finding deadlock-free routes in a network under several buffer and flow configurations.

THEOREM 5. *If a network G has a regular or general buffer configuration and the unrestricted flow control is applied, then Problem* 2 *is* NP*-complete.*

Proof. We are given a network $G = (V, E)$ and a set of source-destination pairs of nodes with an unrestricted flow control, and we ask if there is a corresponding set of routes such that G does not have a deadlock state. This is equivalent to the following question. Is there a corresponding set of routes that do not form a cycle in G? This latter problem is NP-complete. In fact, we can guess a cycle-free set of routes and check the correctness of our guess in polynomial time, and therefore the problem is in NP. We now show that the CNF-satisfiability problem can be reduced to it in polynomial time. Let $\mathbf{F} = \mathbf{F}_1\mathbf{F}_2 \cdots \mathbf{F}_q$ be a CNF expression, and $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$ be the literals in $\mathbf{F}$. We construct the following network $G = (V, E)$. The nodes of G are given by the set

$$V = \{s\} \cup \{F_j | \text{ for } 1 \leqq j \leqq q\} \cup \{x_i, \bar{x}_i, w_i, \bar{w}_i | \text{ for } 1 \leqq i \leqq n\}.$$

The edges of G are given by the set

$$\begin{aligned} E = &\{(s, x_i), (s, \bar{x}_i) | \text{ for } 1 \leqq i \leqq n\} \\ &\cup \{(x_i, w_i), (\bar{x}_i, \bar{w}_i) | \text{ for } 1 \leqq i \leqq n\} \\ &\cup \{(w_i, \bar{x}_i), (\bar{w}_i, x_i) | \text{ for } 1 \leqq i \leqq n\} \\ &\cup \{(w_i, F_j) | x_i \text{ is a variable in } F_j\} \\ &\cup \{(\bar{w}_i, F_j) | \bar{x}_i \text{ is a variable in } F_j\}. \end{aligned}$$

The source-destination pairs are the following:

1. $s - F_j$ for $1 \leqq j \leqq q$.
2. $w_i - \bar{x}_i$ and $\bar{w}_i - x_i$ for $1 \leqq i \leqq n$.

We claim that **F** is satisfiable if and only if these source-destination pairs have a corresponding cycle-free set of routes in G. We first note that the only routes connecting w_i with $\bar{x}_i$ and $\bar{w}_i$ with x_i are the direct ones using the $(w_i, \bar{x}_i)$ and $(\bar{w}_i, x_i)$ edges. Suppose **F** is satisfiable, and let X be a consistent set of variables that, when all are set to 1, satisfy **F**. We can give the following cycle-free set of routes. The routes for the $s - F_j$ pairs are $\langle s, x_i, w_i, F_j \rangle$ if $x_i \in X$ and x_i is in F_j, or $\langle s, \bar{x}_i, \bar{w}_i, F_j \rangle$ if $\bar{x}_i \in X$ and $\bar{x}_i$ is in F_j. The routes for $w_i - \bar{x}_i$ and $\bar{w}_i - x_i$ are respectively $\langle w_i, \bar{x}_i \rangle$ and $\langle \bar{w}_i, x_i \rangle$. Suppose this set of routes forms a cycle; the only cycles in G are of the form $\langle x_i, w_i, \bar{x}_i, \bar{w}_i, x_i \rangle$, which includes both the (x_i, w_i) and $(\bar{x}_i, \bar{w}_i)$ edges. Then, both x_i and $\bar{x}_i$ must be in the set of variables X contradicting the consistency of X. Conversely, let R be a cycle-free set of routes corresponding to the given set of source-destination pairs. We showed that R must contain the $(w_i, \bar{x}_i)$ and $(\bar{w}_i, x_i)$ edges for $1 \leqq i \leqq n$. Since R is cycle-free, it cannot contain both the (x_i, w_i) and $(\bar{x}_i, \bar{w}_i)$ edges for any $1 \leqq i \leqq n$. For each destination F_j $(1 \leqq j \leqq q)$ there must be a variable x_i or $\bar{x}_i$ (and a corresponding (x_i, w_i) or $(\bar{x}_i, \bar{w}_i)$ edge) such that the route from the source s to F_j passes through this variable. Let X be the set of these variables when j ranges from 1 to q. It is now clear that X is a consistent set of variables that satisfies all the $\mathbf{F}_j$ factors. □

Let **F** be the satisfiable CNF Boolean expression and X be the set of variables defined in the example given for Theorem 3. The corresponding network G and the cycle-free set of routes are illustrated in Fig. 3. The routes are $\langle s, x_2, w_2, F_1 \rangle$, $\langle s, \bar{x}_1, \bar{w}_1, F_2 \rangle$, $\langle w_i, \bar{x}_i \rangle$ and $\langle \bar{w}_i, x_i \rangle$ for $i = 1, 2, 3$.

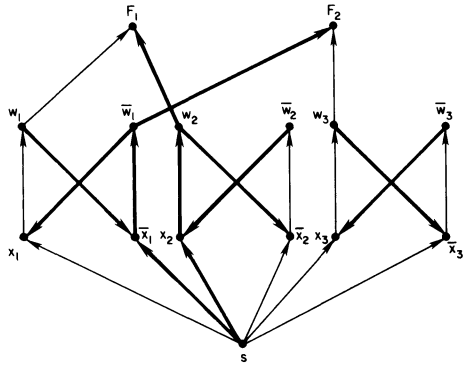

FIG. 3. *The network G and a cycle-free set of routes.*

THEOREM 6. *If the network G has a regular or a general buffer configuration and the isarithmic flow control (with constant K) is applied, then Problem* 2 *is* NP*-complete.*

Proof. The unrestricted flow control is equivalent to an isarithmic flow control where the constant K is larger than the total buffer capacity of the nodes of the network. Then, Problem 2 with an unrestricted flow control can be reduced (in polynomial time) to Problem 2 with an isarithmic flow control, and therefore this latter problem must be NP-hard. Also, if we are given a network G with the isarithmic constant K and a set of source-destination pairs, we can guess a corresponding deadlock-free set of routes in G and then check in polynomial time (using Warshall's algorithm) that the sum of the buffer capacities of the nodes along the minimal cycle is larger than K. Therefore, Problem 2 with an isarithmic flow control is in NP. □

THEOREM 7. *If the network G has a regular or general buffer configuration and a regular or individual end-to-end flow control is applied, then Problem* 2 *is* NP*-hard.*

Proof. The unrestricted flow control is equivalent to a regular or individual end-to-end window flow control where the corresponding upper limits k or k_{ij} are large enough (for example if they exceed the total buffer capacity of the network). Then, Problem 2 with the unrestricted flow control (shown to be NP-complete in Theorem 5) reduces in polynomial time to Problem 2 with end-to-end window flow control. □

We do not know whether the problems shown to be NP-hard in Theorem 7 are in NP or not.

5. State reachability, reachable deadlock states. Let G be a network and F be the flow control procedure applied in G. We say that a network state S of G is *reachable with respect to* F if, starting with an empty network G, there is a sequence of network moves allowed by F that results in the state S in G. We may be interested only in deadlock states that are reachable from an initially empty network G and redefine the notion of "exposure to deadlock" as follows. A network G is *exposed to deadlock* if G has a reachable deadlock state. An example of a non-reachable deadlock state S is the following.

Consider the network $G = (V, E)$, where

$$V = \{v_1, v_2, v_3\}, \qquad E = \{(v_1, v_2), (v_2, v_3), (v_3, v_1)\},$$

and each node has only one buffer. The routes in G are r_1: $\langle v_1, v_2, v_3\rangle$, r_2: $\langle v_2, v_3, v_1\rangle$ and r_3: $\langle v_3, v_1, v_2\rangle$, and the unrestricted flow control is used in G. Let S be the following network state. A packet p_1 is in the node v_2 along the route r_1, a packet p_2 is in the node v_3 along the route r_2 and a packet p_3 is in the node v_1 along the route r_3. It is easy to check that S is a deadlock state which is not reachable from an initially empty network G.

We investigated the complexity of Problem 1 and Problem 2 under the new definition of deadlock exposure; the results, quite similar to previous ones, are summarized in Table 3 and Table 4.

TABLE 3
Time complexity of Problem 1 *with the reachable deadlock definition.*

Flow control applied	unrestricted	isarithmic: K	individual end-to-end: k_{ij}	regular end-to-end: k
General buffer conf.: b_i	Polynomial: $O(\|E\|)$	?	NP-hard (in NP?)	NP-hard (in NP?)
Regular buffer conf.: b	Polynomial: $O(\|E\|)$	?	NP-hard (in NP?)	NP-hard (in NP?)

TABLE 4
Time complexity of Problem 2 *with the reachable deadlock definition.*

Flow control applied	unrestricted	isarithmic: K	individual end-to-end: k_{ij}	regular end-to-end: k
General buffer conf.: b_i	NP-complete	NP-hard (in NP?)	NP-hard (in NP?)	NP-hard (in NP?)
Regular buffer conf.: b	NP-complete	NP-hard (in NP?)	NP-hard (in NP?)	NP-hard (in NP?)

Let S be a state of network $G = (V, E)$. We define the *precedence graph G' of G relative to S* as follows. $G' = (V, E')$ is a directed graph where (v, w) is an edge in E' if and only if when the network is in state S, there is a packet in w whose route passes through the node v before reaching the node w.

LEMMA 1. *With the unrestricted flow control procedure, if the precedence graph G' of a network G relative to a state S is acyclic then the state S is reachable.*

Proof. The algorithm given in Fig. 4 shows how the state S can be reached in G.

```
begin
  topologically sort the directed acyclic graph G';
  comment After this node sorting if (v_i, v_j) is an edge of G' then i < j;
  for j ← |V| step −1 until 1 do
    begin
      successively generate and move along their respective routes, from their
      source node to the node v_j, all the packets that are in the buffers of v_j when
      the state S is reached;
    end
end
```

FIG. 4. *Algorithm for reaching the state S.*

Consider the inner loop of the algorithm. Suppose a packet with destination v_j cannot be generated or passed to an intermediary node v_i along its route, it must be that

1. the buffers of v_i are full of packets, so $i > j$;

2. (v_i, v_j) must be an edge of the precedence graph G', therefore $i < j$; the contradiction is obvious. □

A *cyclic deadlock state* of a network G is a state in which there exists a cycle of nodes in G such that the buffers of each node in the cycle are full of packets waiting to be passed to the next node in the cycle, and the nodes which are not in this cycle are empty.

LEMMA 2. *With the unrestricted flow control procedure, if a network G has a deadlock state S then it has a reachable cyclic deadlock state.*

Proof. If G has a deadlock state S then the routes of G contain a cycle, and therefore G has a cyclic deadlock state S_0. According to Lemma 1, if S_0 is not a reachable state then the precedence graph G' of G relative to S_0 must have at least one cycle $v_1 \to v_2 \to \cdots \to v_1$. We can construct a cyclic deadlock state in the following way (without regard for reachability). We begin with the empty network G and we consider the edge (v_1, v_2) of the cycle. This edge is in the precedence graph and, by definition, there must be a packet in v_2 whose route r passes through the node v_1 before v_2. We successively fill the buffers of v_1 and of all the intermediary nodes along the route r up to (but not including) v_2 with packets with route r. We do the same with the edge (v_2, v_3) and the successive edges of the cycle until we reach a node whose buffers are already full of packets[2] and a cyclic deadlock S_1 is thus formed. Note that S_1 involves a subset of the routes in S_0, and the deadlocked packets in each such route are one step nearer to the source of the route than they were in the state S_0. If S_1 is also not a reachable deadlock state then the corresponding precedence graph contains a cycle and the same process can be repeated to yield a new cyclic deadlock state S_2 (and so on). This process eventually results in a reachable deadlock state. In fact, let l be the

[2] This eventually occurs since $v_1 \to v_2 \to \cdots \to v_1$ is a cycle; note that it may occur before all the edges in the cycle are considered.

length of the longest route in G, then after at most l new nonreachable cyclic deadlock states generated by this process we have a cyclic deadlock state where every deadlocked packet is at the source of its route. Such a deadlock is reachable (the corresponding precedence graph is acyclic). □

THEOREM 8. *If the network $G = (V, E)$ has a general or regular buffer configuration and the unrestricted flow control is applied, then Problem* 1 *(with the new definition of deadlock exposure) is solvable in $O(|E|)$ time.*

Proof. If such a network G has a reachable deadlock state then the routes in G form at least one cycle. If the routes of G form a cycle then G has a cyclic deadlock state and, by Lemma 2, it has a reachable deadlock state. So, G has a reachable deadlock state if and only if the routes in G form at least one cycle. An $O(|E|)$ time breadth-first-search along the routes of G can be used to detect such a cycle. □

THEOREM 9. *If the network $G = (V, E)$ has a regular buffer configuration and a regular end-to-end window flow control is applied, then Problem* 1 *(with the new definition of deadlock exposure) is* NP*-hard.*[3]

Proof. It is easy to check that the polynomial time reductions of the CNF-satisfiability problem to Problem 1 given in the proofs of Theorem 3 and Theorem 4 involve reachable deadlock states. □

COROLLARY 5. *If the network $G = (V, E)$ has a general or regular buffer configuration and an individual or regular end-to-end window flow control is applied then Problem* 1 *(with the new definition of deadlock exposure) is* NP*-hard.*

Proof. Immediate consequence of Theorem 9. □

Note that, at the present time, nothing is known about the complexity of Problem 1 (with the new definition of deadlock exposure) when an isarithmic flow control procedure is applied.

THEOREM 10. *If a network has a regular or general buffer configuration and the unrestricted flow control is applied then Problem* 2 *(with the new definition of deadlock exposure) is* NP*-complete.*

Proof. We are given a network $G = (V, E)$ and a set of source-destination pairs of nodes with an unrestricted flow control, and we want to determine if there is a corresponding set of routes such that G does not have any reachable deadlock state. In the proof of Theorem 8 we showed that, with any set of routes, G does not have a reachable deadlock state if and only if these routes do not form a cycle. Then our problem is to determine if there is a set of routes that do not form a cycle. This problem was shown to be NP-complete in the proof of Theorem 5. □

COROLLARY 6. *If the network $G = (V, E)$ has a general or regular buffer configuration and an isarithmic or an end-to-end window flow control is applied then Problem* 2 *(with the new definition of deadlock exposure) is* NP*-hard.*

Proof. The NP-complete problem of Theorem 10 can be easily reduced to the problems stated in the corollary: isarithmic and end-to-end window flow control with large constants are equivalent to unrestricted flow control. □

REFERENCES

[AHU] A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[DA] D. W. DAVIES, *The control of congestion in packet switching networks*, IEEE Trans., COM-20 (1973), pp. 546–550.

[3] Note that it is not known whether these problems are in NP.

[G] K. D. GUNTHER, *Prevention of buffer deadlocks in packet switching networks*, IFIP-IIASA Workshop on Data Communications, Ladenberg, Austria, pp. g.15–g.19.

[GHKP] A. GIESSLER, J. HAENLE, A. KOENIG AND E. PADÉ, *Free buffer allocation—an investigation by simulation*, Computer Networks, 2 (1978), pp. 191–208.

[KL] L. KLEINROCK, *Queueing Systems vol.* II: *Computer Applications*, John Wiley, New York, 1976.

[PR] W. L. PRICE, *Simulation studies of an isarithmically controlled store-and-forward data communications network*, Proc. IFIP Congress 74, Stockholm, August 1974, pp. 151–154.

[PRH] W. L. PRICE AND J. D. HAENLE, *Some comments on a simulated datagram store-and-forward network*, Computer Networks, 2, (1978), pp. 70–73.

[RH] E. RAUBOLD AND J. HAENLE, *A method of deadlock-free resource allocation and flow control in packet networks*, Proc. ICCC 76, Toronto, Ont., Canada, August, 1976, pp. 483–487.

[TO] S. TOUEG, *Design of deadlock- and livelock-free packet switching networks*, Ph.D. Thesis, Dept. of EECS, Princeton University, Princeton, NJ, 1979.

[TU] S. TOUEG AND J. D. ULLMAN, *Deadlock-free packet switching networks*, Proc. 11th ACM Symposium on the Theory of Computing, Atlanta, GA, May, 1979, pp. 89–98.

[VA] V. AHUJA, *Determining deadlock exposure for a class of store and forward communication networks*, IBM J. Res. Dev., 24 (1980), pp. 49–55.