PAPER *Special Section on Discrete Mathematics and Its Applications*

On the Complexity of Protocol Validation Problems for Protocols with Bounded Capacity Channels

Yoshiaki KAKUDA†, Yoshihiro TAKADA† *and* Tohru KIKUNO†, *Members*

SUMMARY In this paper, it is proven that the following three decision problems on validation of protocols with bounded capacity channels are NP-complete. (1) Given a protocol with the channel capacity being 1, determine whether or not there exist deadlocks in the protocol. (2) Given a protocol with the channel capacity being 1, determine whether or not there exist unspecified receptions in the protocol. (3) Given a protocol with the channel capacity being 2, determine whether or not there exist overflows such that the channel capacity is not bounded by 1 in the protocol. These results suggest that, even when all channels in a protocol are bounded by 1 or 2, protocol validation should be in general interactable. It also clarifies the boundary of computational complexity of protocol validation problems because the channel capacity 0 does not allow protocols to transmit and recieve messages.

key words: *computational complexity, deadlock, NP-complete, protocol, protocol validation*

1. Introduction

A protocol consists of processes and channels where each pair of two distinct processes are connected by a channel. Messages are transmitted and received through channels. Protocol validation is to check whether or not the specification of a protocol includes protocol errors such as unspecified receptions, redundant transmissions/receptions, deadlocks, livelocks, and overflows [2]-[6]. Among them, deadlocks, unspecified receptions and overflows are typical protocol errors that should be detected by protocol validation. Intuitively, deadlocks, unspecified receptions and overflows are as follows. A deadlock is a global state of a protocol such that the protocol cannot execute any transmission and reception of messages. An unspecified reception is a process state such that a message has been transmitted but cannot be received at the state. An overflow is a channel state such that the number of messages which have been transmitted but not yet received through the channel is larger than a given positive number.

This paper discusses the computational complexity of the following three decision problems on the protocol validation; the deadlock detection problem, the unspecified reception detection problem and the overflow detection problem. Brand and Zafiropulo showed that these problems for general protocols are undecidable when the channel capacity of the protocol is unbounded [2]. In practice, however, the channel capacity is bounded by a finite constant. West proposed an algorithm to detect deadlocks, unspecified receptions and overflows when the channel capacity is bounded by a constant and the number of processes is constant [6]. Räuchle and Toueg showed that the deadlock detection problem is PSPACE-complete in case that the channel capacity is bounded by a finite constant and the number of processes is 2 [5]. It is thus interesting to find the computational complexity of the decision problems on the protocol validation when the channel capacity is bounded by a constant and there is no restriction on the number of processes.

This paper shows that both the deadlock detection and the unspecified reception detection problems are NP-complete, even if the channel capacity is restricted to 1, and that the overflow detection problem of deciding whether or not there exist overflows such that the number of messages left in a channel is larger than 1, even if the channel capacity is restricted to 2, is also NP-complete. These results mean that, even if the channel capacity is bounded by 1 or 2, the problems are difficult to solve efficiently in the worst case.

The rest of this paper is organized as follows. In Sect. 2, three decision problems on protocol validation, i.e., the deadlock detection problem, the unspecified reception detection problem and the overflow detection problem, are formulated. In Sect. 3, the NP-completeness of the deadlock detection problem for protocols with bounded capacity channels is proven, and in Sect. 4, that of the other problems is also proven. In Sect. 5, conclusions are described with future research direction.

2. Protocol Validation Problems

A protocol consists of processes and channels. Each pair of two distinct processes are connected by a channel. Messages are transmitted and received through channels. The model in this paper uses explicit finite state machines to represent processes and implicit FIFO queues to represent channels.

Definition 1 (Protocol): A *protocol* among N processes is a quadruple $P=(Q, o, M, succ)$, where $Q=(Q_1, \cdots, Q_N)$, $o=(o_1, \cdots, o_N)$, $M=(M_{1,1}, \cdots, M_{1,N}, M_{2,1},$

Manuscript received September 20, 1993.

† The authors are with the Faculty of Engineering Science, Osaka University, Toyonaka-shi, 560 Japan.

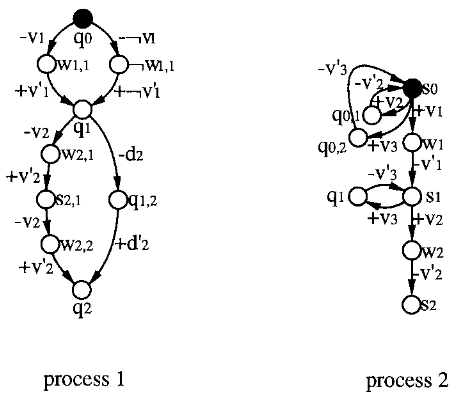


Fig. 1 Protocol example.

$\cdots$, $M_{N,N}$) and $succ=(succ_1, \cdots, succ_N)$.

Each Q_i represents a set of states of the i-th process. For any i and j $(i \neq j)$, $Q_i \cap Q_j = \phi$. A symbol ϕ denotes an empty set.

Each o_i $(o_i \in Q_i)$ represents the initial state of the i-th process.

Each $M_{i,j}$ represents a set of message types that can be transmitted through the channel from the i-th process to the j-th process. For any i, j, i' and j' $(i \neq i'$ or $j \neq j')$, $M_{i,j} \cap M_{i',j'} = \phi$ and $M_{i,i} = \phi$.

Each $succ_i$ is a partial mapping $Q_i \times M_i \rightarrow Q_i$ which represents how the i-th process transmits or receives a message and changes its state. The set M_i is a set of message types that the i-th process can transmit or receive $(M_i = M_{i,1} \cup \cdots \cup M_{i,N} \cup M_{1,i} \cup \cdots \cup M_{N,i})$. The value of $succ_i(s, x)$ represents either the state of the i-th process after it transmits a message x in state s $(s \in Q_i)$, if $x \in M_{i,j}$, or the state after it receives a message x in state s, if $x \in M_{j,i}$.

The i-th *process* is formally defined as the quadruple $(Q_i, o_i, M_{i,j} \cup M_{j,i}, succ_i)$.

A pair (s, x) for a state s $(s \in Q_i)$ and a message x $(x \in M_{i,j})$ is said to be a transmission of the i-th process, and a pair (s, x) for a state s $(s \in Q_i)$ and a message x $(x \in M_{j,i})$ is said to be a reception. A *state transition* is either a transmission or a reception.

(End of Definition)

Example 1: Figure 1 shows an example of a protocol with 2 processes. In the figure, '○' depicts a state and '●' depicts an initial state. Arrows '→' depict state transitions. Symbols '$-x$' and '$+x$' represent transmissions and receptions of a message x, respectively. The protocol represents that, for example, process 1 transmits message v_1 at state q_0 and enters state $w_{1,1}$ and process 2 receives it at state s_0. (End of Example)

Definition 2 (Global state and transitions): In a protocol P, a *global state* is a pair $G=(S, C)$, where $S=(s_1, \cdots, s_N)$ and $C=(c_{1,1}, \cdots, c_{1,N}, c_{2,1}, \cdots, c_{N,N})$. Each s_i represents a state of the i-th process $(s_i \in Q_i)$. Each $c_{i,j}$ represents a state of the channel from the i-th process to the j-th process, i.e., $c_{i,j}$ is a sequence of messages that the i-th process transmitted to the j-th process but the j-th process does not receive yet. Each message in $c_{i,j}$ belongs to $M_{i,j}$.

The *initial global state* is the global state $G_0=(S_0, C_0)$ with all channels empty and with each process in its initial state, where $S_0=(o_1, \cdots, o_N)$ and $C_0=(\varepsilon, \cdots, \varepsilon)$. A symbol ε denotes an empty sequence.

A *global state transition* is defined as a binary relation '$\vdash$' on global states. We denote $(S, C) \vdash (S', C')$ if and only if the global state (S, C) can transit to (S', C') by exactly one state transition of one process. In other words, $(S, C) \vdash (S', C')$ if and only if there exist some i, j and x satisfying one of the following two conditions. (1) $s'_i = succ_i(s_i, x)$ $(x \in M_{i,j})$, $c'_{i,j} = c_{i,j} \cdot x$, and for any j $(j \neq i)$, $s'_j = s_j$ and $c'_{i,j} = c_{i,j}$. Here, $\sigma_1 \cdot \sigma_2$ denotes a concatenation of sequences σ_1 and σ_2. (2) $s'_j = succ_j(s_j, x)$ $(x \in M_{i,j})$, $c_{i,j} = x \cdot c'_{i,j}$, and for any j $(j \neq i)$, $s'_j = s_j$ and $c'_{i,j} = c_{i,j}$.

Let '$\vdash^*$' be the reflexive and transitive closure of '$\vdash$'. A global state $G=(S, C)$ is said to be *reachable* if and only if $G_0 \vdash^* G$. (End of Definition)

Definition 3 (Specified and executable): In a protocol P, a state transition (s_i, x) is *specified* if and only if $succ_i(s_i, x)$ is defined. A transmission (s_i, x) is *executable* if and only if there exists a reachable global state $G=(S, C)$ such that $S=(s_1, \cdots, s_i, \cdots, s_N)$. The reception (s_i, x) is *executable* if and only if there exists a reachable global state $G=(S, C)$ such that $S=(s_1, \cdots, s_i, \cdots, s_N)$ and $C=(c_{1,1}, \cdots, c_{j,i}, \cdots, c_{N,N})$ $(c_{j,i} = x \cdot Y$ where Y is a sequence of messages in $M_{j,i})$.

(End of Definition)

Definition 4 (Bounded and fixed channel): In a protocol P, let N be the number of processes, let $|Q|$ be the number of states of all processes, let $|M|$ be the number of distinct message types and let h be a positive integer that is polynomial with respect to N, $|Q|$ and $|M|$. A channel from the i-th process to the j-th process is said to be *bounded* by h if and only if, for every reachable global state $G=(S, C)$, the length of each $c_{i,j}$ is less than or equal to h.

Let h be a positive integer constant independent of N, $|Q|$ and $|M|$. A channel from the i-th process to the j-th process is said to be *fixed* to h if and only if, for every reachable global state $G=(S, C)$, the length of each $c_{i,j}$ is less than or equal to h. This h is said to be *channel capacity*. If a channel is fixed, the channel is bounded.

A protocol P is said to be fixed to h if and only if all channels in P are fixed to h.

(End of Definition)

Definition 5 (Protocol errors): (1) Deadlock: A global state $G=(S, C)$ is said to be *stable* if and only if G is reachable and $C=(\varepsilon, \cdots, \varepsilon)$. A stable global state $G=(S, C)$ is said to be a *deadlock* if and only if $S=(s_1, \cdots, s_N)$ and there exist no specified transmissions (s_i, x) $(1 \leqq i \leqq N$, $1 \leqq j \leqq N$, $s_i \in Q_i$, $x \in M_{i,j})$.
(2) Unspecified Reception: An *unspecified reception* is a reception that is executable but not specified.

(3) Overflow: An *overflow* is a channel state such that the channel is not fixed to a predefined positive integer h. (End of Definition)

Definition 6 (Protocol validation problems): (1) Given a protocol P fixed to h, determine whether or not there exist any deadlocks. This decision problem is called the *deadlock detection problem*, which is abbreviated as *DDP* hereinafter.

(2) Given a protocol P fixed to h, determine whether or not there exist any unspecified receptions. This decision problem is called the *unspecified reception detection problem*, which is abbreviated as *URDP* hereinafter.

(3) Given a protocol P fixed to h' and a positive integer h $(h<h')$, determine whether or not there exist any overflows such that a channel is not fixed to h. This decision problem is called the *overflow detection problem*, which is abbreviated as *ODP* hereinafter.

(End of Definition)

In the following sections, these three decision problems, DDP, URDP and ODP, are discussed in case that $h=1$ and $h'=2$.

3. Deadlock Detection Problem

It is obvious that DDP is in NP. In order to prove that DDP is NP-complete, it suffices to show that any NP-complete problem is polynomially transformable to DDP. In this section, the 3-satisfiability problem is proven to be polynomially transformable to DDP, and the following theorem is proven.

Theorem 1: DDP is NP-complete even when protocols are fixed to h $(h=1)$. (End of Theorem)

The 3-satisfiability problem is a problem to determine whether a boolean expression is satisfiable or not.

The rest of this section is organized as follows. In Sect. 3. 1, the 3-satisfiability problem is introduced. In Sect. 3. 2, the construction of a protocol P based on a given boolean expression B is described. There exist deadlocks in the constructed P if and only if B is satisfiable. In Sect. 3. 3, it is proven that the 3-satisfiability problem is transformable to DDP by using the construction described in Sect. 3. 2.

3. 1 3-Satisfiability Problem

The 3-Satisfiability Problem is as follows.

Definition 7 (3-Satisfiability Problem): Let a set of boolean variables be V. A boolean expression can be generally denoted by a conjunctive normal form $B=(u_{1,1}+u_{1,2}+\cdots+u_{1,L_1})\times(u_{2,1}+u_{2,2}+\cdots+u_{2,L_2})\times\cdots\times(u_{I,1}+u_{I,2}+\cdots+u_{I,L_I})$. Each $u_{i,l}$ $(1\leqq i\leqq I,\ 1\leqq l\leqq L_i)$ is either v_k or $\neg v_k$ $(v_k\in V)$ and is said to be a *literal*. Operators $+$, $\times$ and $\neg$ denote boolean sum, product and negation, respectively. For each i $(1\leqq i\leqq I)$, $(u_{i,1}+u_{i,2}+\cdots+u_{i,L_i})$ is said to be a *clause*. We assume, without loss of generality, that the same variable does not appear more than once in one clause, although two literals $u_{i,k}$ and $u_{j,l}(i\neq j)$ in distinct clauses can possibly have the same variable.

An *assignment* is a mapping $V\rightarrow\{true,\ false\}$ which assigns true or false to each variable in B. A *truth assignment* is an assignment when B holds true. A conjunctive normal form B is said to be *satisfiable* if and only if there exists at least one truth assignment.

The *3-Satisfiability problem* is defined as follows. Given a boolean expression $B=(u_{1,1}+u_{1,2}+u_{1,3})\times(u_{2,1}+u_{2,2}+u_{2,3})\times\cdots\times(u_{I,1}+u_{I,2}+u_{I,3})$ in a conjunctive normal form where each clause consists of three literals, determine whether B is satisfiable or not. This decision problem is abbreviated as *3SAT* hereinafter.

(End of Definition)

As for the computational complexity of the above 3-Satisfiability problem, the following theorem is known.

Theorem 2: 3SAT is NP-complete [1].

(End of Theorem)

3. 2 Construction of Protocol Based on Boolean Expression

This section describes how to construct a protocol P based on a boolean expression B in a conjunctive normal form where each clause consists of three literals. In the constructed P, there exist deadlocks if and only if B is satisfiable.

The construction is stated informally as follows. When the boolean expression B consists of I clauses, the protocol P consists of $(I+3)$ processes. The i-th process $(1\leqq i\leqq I)$ corresponds to the i-th clause in B. At first, the $(I+1)$-th process determines an assignment for B and transmits messages to each i-th process $(1\leqq i\leqq I)$ according to the assignment. Each the i-th process $(1\leqq i\leqq I)$ receives l messages $(0\leqq l\leqq 3)$ from the $(I+1)$-th process when l literals in the i-th clause in B are assigned false. If the i-th clause in B is assigned false, i.e., all three literals in the clause are assigned false, the i-th process receives three messages from the $(I+1)$-th process, and then transmits a message x_i to the $(I+2)$-th process. When m clauses are assigned false, the $(I+2)$-th process receives m messages from m processes. Then the $(I+2)$-th process transmits a message y_i back to one of these m processes. If the j-th process $(1\leqq j\leqq I)$ receives the message y_i, the j-th process transmits a message z_i $(1\leqq i\leqq I+2,\ i\neq j)$ to every process in order that all processes return to their initial states. If no clauses are assigned false, no processes transmit x_i to the $(I+2)$-th process, and then a deadlock occurs.

For convenience of explanation, the construction of the protocol P based on the boolean expression B is divided into the *fundamental* part and the *additional* part. We describe the protocol P as $(Q,\ o,\ M,\ succ)$ where $Q=(Q_1^F\cup Q_1^A,\ \cdots,\ Q_{I+3}^F\cup Q_{I+3}^A)$, $o=(o_1,\ \cdots,$

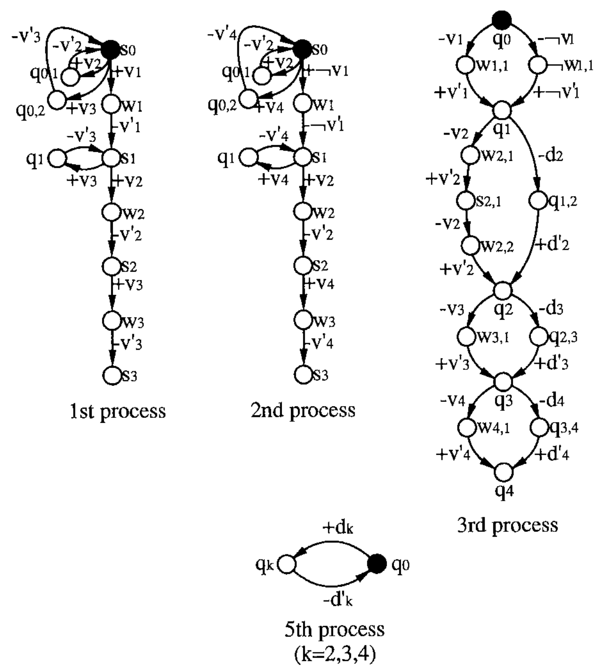


Fig. 2 Example of fundamental transformation for deadlock detection problem (The 4th process does not appear in this figure).

o_{I+3}), $M=(M^F_{1,1}\cup M^A_{1,1}, \cdots, M^F_{I+3,I+3}\cup M^A_{I+3,I+3})$ and $succ=(succ_1, \cdots, succ_{I+3})$. Each Q^F_i represents a set of states in the fundamental part of the i-th process, and each Q^A_i represents a set of states in the additional part. Each $M^F_{i,j}$ represents a set of message types in the fundamental part, and each $M^A_{i,j}$ represents a set of message types in the additional part. Each $succ_i$ is also devided into two parts.

Example 2: The fundamental part of a protocol P based on a boolean expression $B=(v_1+v_2+v_3)\times(\neg v_1+v_2+v_4)$ is shown in Fig. 2. The protocol P is shown in Fig. 3. In the figures, '○' depicts a state and '●' depicts an initial state. Arrows '→' depict state transitions. Symbols '$-x$' and '$+x$' represent transmissions and receptions of a message x, respectively. In this example, $I=2$. (End of Example)

The fundamental part of the $(I+1)$-th process is constructed in the following manner. Suppose that the sequence of distinct variables in B in the order of their appearance is $W=\langle v_1, \cdots, v_K\rangle$, where $K=|V|$ is the number of variables and v_k $(1\leqq k\leqq K)$ is the k-th variable in B. Let the number of appearances of a literal u_k, which is either v_k or $\neg v_k$, in B be $cnt(u_k)$. Let the clauses which include a literal u_k be the $cls(u_k, 1)$-th clause, the $cls(u_k, 2)$-th clause, $\cdots$ and the $cls(u_k, cnt(u_k))$-th clause. (In Example 2, $K=4$, $W=\langle v_1, v_2, v_3, v_4\rangle$, $cnt(v_2)=2$, $cls(v_2, 1)=1$, $cls(v_2, 2)=2$, $cnt(\neg v_1)=1$ and $cls(\neg v_1, 1)=2$.) The $(n+1)$-th process selects either v_k or $\neg v_k$ as u_k for each k $(1\leqq k\leqq K)$, and then transmits messages u_k to all the $cls(u_k, 1)$-th clause, the $cls(u_k, 2)$-th clause, $\cdots$, and

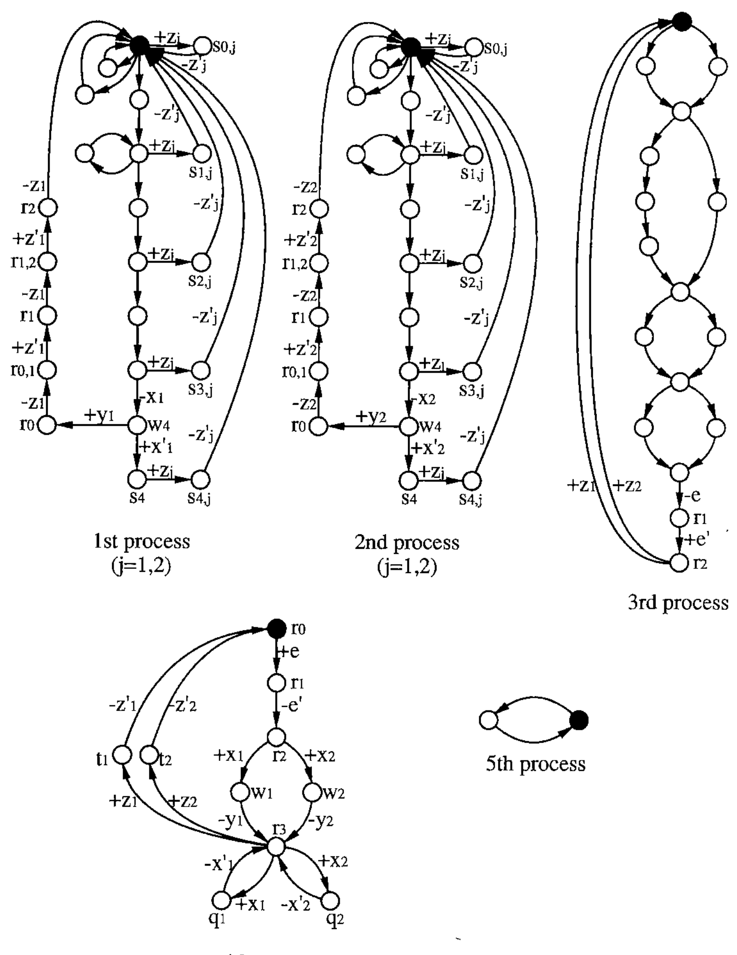


Fig. 3 Example of transformation for deadlock detection problem.

the $cls(u, cnt(u_k))$-th clause.

$$Q^F_{I+1}=\{q_0, \cdots, q_K\}\cup\{w_{k,1}, s_{k,1}, \cdots, w_{k,cnt(v_k)-1}, s_{k,cnt(v_k)-1}, w_{k,cnt(v_k)}| cnt(v_k)>0\}\cup\{\neg w_{k,1}, \neg s_{k,1}, \cdots, \neg s_{k,cnt(\neg v_k)-1}, \neg w_{k,cnt(\neg v_k)}|cnt(\neg v_k)>0\}\cup \{q_{k-1,k}|1\leqq k\leqq K,\ cnt(v_k)=0 \text{ or } cnt(\neg v_k)=0\},$$

$$o_{I+1}=q_0,$$

$$M^F_{I+1,I+3}=\{d_k|1\leqq k\leqq K,\ cnt(v_k)=0 \text{ or } cnt(\neg v_k)=0\}.$$

For each $i\,(1\leqq i\leqq I)$,

$$M^F_{I+1,i}=\{v_k|1\leqq k\leqq K,\ v_k \text{ is in the } i\text{-th clause}\}\cup\{\neg v_k|1\leqq k\leqq K, \neg v_k \text{ is in the } i\text{-th clause}\}.$$

If $cnt(v_k)=0$, for each $k(1 \leqq k \leqq K)$,

$$succ_{I+1}(q_{k-1}, d_k)=q_{k-1,k} \quad (d_k \in M_{I+1,I+3}),$$
$$succ_{I+1}(q_{k-1,k}, d'_k)=q_k \quad (d'_k \in M_{I+3,I+1}).$$

If $cnt(v_k) \neq 0$, for each $k(1 \leqq k \leqq K)$,

$$succ_{I+1}(q_{k-1}, v_k)=w_{k,1} \quad (v_k \in M_{I+1,cls(v_k,1)}),$$
$$succ_{I+1}(w_{k,1}, v'_k)=s_{k,1} \quad (v'_k \in M_{cls(v_k,1),I+1}),$$
$$succ_{I+1}(s_{k,1}, v_k)=w_{k,2} \quad (v_k \in M_{I+1,cls(v_k,2)}),$$
$$\vdots$$
$$succ_{I+1}(w_{k,cnt(v_k)-1}, v'_k)$$
$$=s_{k,cnt(v_k)-1} \quad (v'_k \in M_{cls(v_k,cnt(v_k)-1),I+1}),$$
$$succ_{I+1}(s_{k,cnt(v_k)-1}, v_k)$$
$$=w_{k,cnt(v_k)} \quad (v_k \in M_{I+1,cls(v_k,cnt(v_k))}),$$
$$succ_{I+1}(w_{k,cnt(v_k)}, v'_k)=q_k \quad (v'_k \in M_{cls(v_k,cnt(v_k)),I+1}).$$

In case $cnt(\neg v_k)=0$ and case $cnt(\neg v_k) \neq 0 (1 \leqq k \leqq K)$, $succ_{I+1}$ is defined in a similar way and their descriptions are omitted. Note that $cnt(v_k)=0$ and $cnt(\neg v_k)=0$ do not hold simultaneously because there exists at least one variable v_k in B. Each $M^F_{i,I+1}$ $(1 \leqq i \leqq I)$ is described later as a component of the i-th process.

The fundamental part of the i-th process $(1 \leqq i \leqq I)$ is constructed in the following manner. Suppose for each $i(1 \leqq i \leqq I)$ that the i-th clause is composed of three literals $u_{var(i,1)}$, $u_{var(i,2)}$ and $u_{var(i,3)}$, where each $u_{var(i,l)}(1 \leqq l \leqq 3)$ is either $v_{var(i,l)}$ or $\neg v_{var(i,l)}$ $(v_{var(i,l)} \in V)$ and $1 \leqq var(i, 1) < var(i, 2) < var(i, 3) \leqq K$. (In Example 2, $var(2, 1)=1$, $var(2, 2)=2$, $var(2, 3)=4$ and $u_{var(2,3)}=v_4$.) The i-th process receives a message $u_{var(i,l)}$ from the $(I+1)$-th process if a literal $u_{var(i,l)}$ is assigned false. If the i-th clause is assigned false, i,e., all the three literals in the i-th clause are assigned false, then the i-th process receives three messages and reachs state s_3.

$$Q^F_i=\{s_0, w_1, s_1, w_2, s_2, w_3, s_3, q_{0,1}, q_{0,2}, q_1\},$$
$$o_i=s_0,$$
$$M^F_{i,I+1}=\{v'_k | \text{literal } v_k \text{ is in the } i\text{-th clause}\} \cup$$
$$\{\neg v'_k | \text{literal } \neg v_k \text{ is in the } i\text{-th clause}\}.$$

In the following $succ_i$, suppose that $u_{var(i,l)}=v_{var(i,l)}$ and $u'_{var(i,l)}=v'_{var(i,l)}$ if the i-th clause includes the literal $v_{var(i,l)}$, and $u_{var(i,l)}=\neg v_{var(i,l)}$ and $u'_{var(i,l)}=\neg v'_{var(i,l)}$ if the i-th clause includes the literal $\neg v_{var(i,l)}$.

$$sccc_i(s_0, u_{var(i,1)})=w_1 \quad (u_{var(i,1)} \in M_{I+1,i}),$$
$$succ_i(w_1, u'_{var(i,1)})=s_1 \quad (u'_{var(i,1)} \in M_{i,I+1}),$$
$$succ_i(s_1, u_{var(i,2)})=w_2 \quad (u_{var(i,2)} \in M_{I+1,i}),$$
$$succ_i(w_2, u'_{var(i,2)})=s_2 \quad (u'_{var(i,2)} \in M_{i,I+1}),$$
$$succ_i(s_2, u_{var(i,3)})=w_3 \quad (u_{var(i,3)} \in M_{I+1,i}),$$
$$succ_i(w_3, u'_{var(i,3)})=s_3 \quad (u'_{var(i,3)} \in M_{i,I+1}),$$
$$succ_i(s_0, u_{var(i,2)})=q_{0,1} \quad (u_{var(i,2)} \in M_{I+1,i}),$$
$$succ_i(q_{0,1}, u'_{var(i,2)})=s_0 \quad (u'_{var(i,2)} \in M_{i,I+1}),$$
$$succ_i(s_0, u_{var(i,3)})=q_{0,2} \quad (u_{var(i,3)} \in M_{I+1,i}),$$
$$succ_i(q_{0,2}, u'_{var(i,3)})=s_0 \quad (u'_{var(i,3)} \in M_{i,I+1}),$$
$$succ_i(s_1, u_{var(i,3)})=q_1 \quad (u_{var(i,3)} \in M_{I+1,i}),$$
$$succ_i(q_1, u'_{var(i,3)})=s_1 \quad (u'_{var(i,3)} \in M_{i,I+1}).$$

The $(I+2)$-th process has no fundamental part and its additional part is described later.

The fundamental part of the $(I+3)$-th process is constructed in the following manner, although it does not have any additional part.

$$Q^F_{I+3}=\{q_0\} \cup$$
$$\{q_k | 1 \leqq k \leqq K, cnt(v_k)=0 \text{ or } cnt(\neg v_k)=0\},$$
$$o_{I+3}=q_0,$$
$$M^F_{I+3,I+1}=\{d'_k | 1 \leqq k \leqq K, cnt(v_k)=0 \text{ or } cnt(\neg v_k)=0\},$$
$$succ_{I+3}(q_0, d_k)=q_k \quad (d_k \in M_{I+1,I+3}),$$
$$succ_{I+3}(q_k, d'_k)=q_0 \quad (d'_k \in M_{I+3,I+1}).$$

The additional part of the $(I+1)$-th process is constructed in the following manner.

$$Q^A_{I+1}=\{r_1, r_2\},$$
$$M^A_{I+1,I+2}=\{e\},$$
$$succ_{I+1}(q_K, e)=r_1 \quad (e \in M_{I+1,I+2}),$$
$$succ_{I+1}(r_1, e')=r_2 \quad (e' \in M_{I+2,I+1}).$$

For each $i(1 \leqq i \leqq I)$,

$$succ_{I+1}(r_2, z_i)=q_0 \quad (z_i \in M_{i,I+1}).$$

The additional part of the i-th process $(1 \leqq i \leqq I)$ is constructed in the following manner. The i-th process transmits a message x_i to the $(I+2)$ process when the i-th clause in B is assigned false.

$$Q^A_i=\{w_4, s_4, r_0, r_{0,1}, r_1, r_{1,2}, \cdots, r_{I-1}, r_{I-1,I}, r_I\} \cup$$
$$\{s_{l,j} | 0 \leqq l \leqq 4, 1 \leqq j \leqq I, j \neq i\},$$
$$M^A_{i,I+2}=\{x_i\},$$
$$succ_i(s_3, x_i)=w_4 \quad (x_i \in M_{i,I+2}),$$
$$succ_i(w_4, x'_i)=s_4 \quad (x'_i \in M_{I+2,i}),$$
$$succ_i(s_3, y_i)=r_0 \quad (y_i \in M_{I+2,i}),$$
$$succ_i(r_0, z_i)=r_{0,1} \quad (z_i \in M_{i,1}),$$
$$succ_i(r_{0,1}, z'_i)=r_1 \quad (z'_i \in M_{1,i}),$$

$$\vdots$$

$succ_i(r_{i-2}, z_i) = r_{i-2,i-1} \quad (z_i \in M_{i,i-1}),$

$succ_i(r_{i-1,i-2}, z_i') = r_{i-1} \quad (z_i' \in M_{i-1,i}),$

$succ_i(r_{i-1}, z_i) = r_{i-1,i} \quad (z_i \in M_{i,i+1}),$

$succ_i(r_{i-1,i}, z_i') = r_i \quad (z_i' \in M_{i+1,i}),$

$$\vdots$$

$succ_i(r_{I-1}, z_i) = r_{I-1,I} \quad (z_i \in M_{i,I+2}),$

$succ_i(r_{I-1,I}, z_i') = r_I \quad (z_i' \in M_{I+2,i}),$

$succ_i(r_I, z_i) = s_0 \quad (z_i \in M_{i,I+1}).$

For each l and $j\,(0 \leqq l \leqq 4,\ 1 \leqq j \leqq I,\ j \neq i)$,

$M^A_{i,j} = \{z_j, z_j'\},$

$succ_i(s_l, z_j) = s_{l,j} \quad (z_j \in M_{j,i}),$

$succ_i(s_{l,j}, z_j') = s_0 \quad (z_j' \in M_{i,j}).$

The $(I+2)$-th process $(1 \leqq i \leqq I)$ is constructed in the following manner. Each the i-th process $(1 \leqq i \leqq I)$ transmits a message x_i if the i-th clause is assigned false. The $(I+2)$-th process selects one x_j of these messages and transmits a message y_j to the j-th process.

$Q^A_{I+2} = \{r_0, r_1, r_2, r_3\} \cup$
$\quad \{w_i, q_i, p_i \mid 1 \leqq i \leqq I\},$

$o_{I+2} = \{r_0\},$

$M^A_{I+2,I+1} = \{e'\},$

$succ_{I+2}(r_0, e) = r_1 \quad (e \in M_{I+1,I+2}),$

$succ_{I+2}(r_1, e') = r_2 \quad (e' \in M_{I+2,I+1}).$

For each $i\,(1 \leqq i \leqq I)$,

$M^A_{I+2,i} = \{y_i, x_i', z_i'\},$

$succ_{I+2}(r_2, x_i) = w_i \quad (x_i \in M_{i,I+2}),$

$succ_{I+2}(w_i, y_i) = r_3 \quad (y_i \in M_{I+2,i}),$

$succ_{I+2}(r_3, x_i) = q_i \quad (x_i \in M_{i,I+2}),$

$succ_{I+2}(q_i, x_i') = r_3 \quad (x_i' \in M_{I+2,i}),$

$succ_{I+2}(r_3, z) = p_i \quad (z_i \in M_{i,I+2}),$

$succ_{I+2}(p_i, z_i') = r_0 \quad (z_i' \in M_{I+2,i}).$

In this paper, we may use the same message name in distinct channels and the same state name in distinct processes, although $M_{i,j} \cap M_{i',j'} = \phi$ and $Q_i \cap Q_j = \phi$ as described in Definition 1. In case that there exist the same name in distinct channels or processes in P, suppose that those names are translated into distinct names by adding the distinct channel names or process names.

3. 3 Proof of NP-Completeness

It is obvious that DDP is in NP. In order to prove that DDP is NP-complete, it suffices to show that the 3SAT is polynomially transformable to DDP. As described in Sect. 3. 2, a protocol P can be constructed for any boolean expression B in a conjunctive normal form where each clause consists of three literals. There exists the following relation between B and P.

Relation 1: A sequence of executable state transitions through states $q_0, q_1, \cdots, q_K$ in the $(I+1)$-th process in the constructed protocol P is called a *cycle*. There exists a one-to-one correspondence between cycles and assignments for B. If the variable $v_k\,(1 \leqq k \leqq K)$ in B is assigned false, then the $(I+1)$-th process of P transmits the message v_k to all processes which correspond to clauses including the literal v_k, otherwise, the $(I+1)$-th process of P transmits the message $\neg v_k$.

(End of Relation)

According to Relation 1, the following lemma holds true for the constructed protocol P based on B.

Lemma 1: When a protocol P based on a boolean expression B is constructed as described in Sect. 3. 2, there exist deadlocks in P if and only if B is satisfiable.

(End of Lemma)

Proof 1(Lemma 1): First, the sufficiency will be shown below. Suppose that there exist no truth assignments for B, i.e., B is always false for any assignment a. This implies that at least one clause of B is false for any assignment a. Suppose that the i-th clause $(u_{i,1} + u_{i,2} + u_{i,3})$ is assigned false, i.e., $u_{i,1}$, $u_{i,2}$ and $u_{i,3}$ are all false. The $(I+1)$-th process in P transmits three messages $u_{i,1}$, $u_{i,2}$ and $u_{i,3}$ to the i-th process, as described in Relation 1. The i-th process receives these three messages and reaches state s_3 accordingly. Then, the i-th process transmits a message x_i on state s_3 to the $(I+2)$-th process. Since more than one clause may possibly be assigned false, more than one process may transmit messages x_i to the $(I+2)$-th process. Then, the $(I+2)$-th process receives one x_j of these messages on state r_2, and transmits a message y_j to the j-th process. The other messages $x_i\,(i \neq j)$ are received by the $(I+2)$-th process on state r_3. When the j-th process receives a message y_j, and transmits messages z_i to all processes except itself and the $(I+3)$-th process. Then, every process returns to the initial states again by receiving z_i. Therefore, there exist no deadlocks in P.

Second, the necessity will be shown below. Suppose that there exist no deadlocks in protocol P in the first cycle. According to the construction of P, every process returns to the initial state at the end of the first cycle by receiving a message z_i $(1 \leqq i \leqq I)$. Suppose that the j-th process transmits these messages $z_j\,(1 \leqq j \leqq I)$. Since the j-th process has to reach state r_0 in order to transmit these messages z_j $(1 \leqq j \leqq I)$, the j-th process reaches state r_0 through states s_1, s_2 and s_3. This implies that the $(I+1)$-th process transmits three messages $u_{j,1}$, $u_{j,2}$ and $u_{j,3}$ and the j-th process receives these three messages in the first cycle. As described in Relation 1, the first cycle corresponds to an assignment a for B. Since the $(I+1)$-th process transmits the three messages $u_{j,1}$, $u_{j,2}$ and $u_{j,3}$, the corresponding three literals $u_{j,1}$, $u_{j,2}$ and $u_{j,3}$ in B is assigned false in the assignment a. Hence, at least the j-th clause is assigned false and B is not satisfied by the assignment a. According to the construction of P, the above discussion holds true for any cycle and any assignment. Therefore, B is not satisfied by any assignment a.

Since the sufficiency and the necessity were shown, it is proven that there exist deadlocks in P if and only if B is satisfiable. (End of Proof)

Theorem 1 is proven as follows.

Proof 2 (Theorem 1): As described in Sect. 3. 2, a protocol P can be constructed for any boolean expression B in a conjunctive normal form where each clause consists of three literals. According to Lemma 1, there exist deadlocks in the constructed P if and only if B is satisfiable. This construction can be done in polynomial time obviously.

The channel in the constructed protocol P is fixed to h $(h=1)$ because a handshake communication mechanism is introduced for all channels where more than one message may possibly be stored. The handshake communication mechanism means a mechanism such that, when the i-th process transmits a message x to the j-th process $(j\neq i)$, the j-th process always returns an acknowledgement message x' to the i-th process.

Therefore, 3SAT is polynomially transformable to DDP for fixed channel capacity h $(h=1)$. Since DDP is in NP obviously, DDP is proven to be NP-complete. (End of Proof)

4. Unspecified Reception Detection Problem and Overflow Detection Problem

The proof of NP-completeness of URDP and ODP for fixed protocols can be proven in a similar way of the proof of Theorem 1.

Theorem 3: URDP is NP-complete even when protocols are fixed to h $(h=1)$. (End of Theorem)

Proof 3 (Theorem 3): To prove that Theorem 3 holds, a protocol P based on a boolean expression B in a conjunctive normal form is constructed. Since the fundamental part of the construction of a protocol P is the same as that described in Sect. 3. 2, only the additional part is described below.

The additional part of the $(I+1)$-th process is constructed in the following manner.

$$Q^A_{I+1}=\{r_0,\ r_1\},$$
$$M^A_{I+1,1}=\{x\},$$
$$succ_{I+1}(q_K,\ x)=r_0 \quad (x\in M_{I+1,1}),$$
$$succ_{I+1}(r_0,\ x')=r_1 \quad (x'\in M_{1,I+1}).$$

The additional part of the 1-st process is constructed in the following manner.

$$Q^A_1=\{p_l \mid 0\leqq l\leqq 3\}\cup\{r^t_1,\ r^t_2,\ r^t_3,\ r^f_1,\ r^f_2,\ r^f_3\},$$
$$M^A_{1,I+1}=\{x'\},$$
$$M^A_{1,2}=\{t,\ f\}.$$

For each l $(0\leqq l\leqq 2)$,

$$succ_1(s_l,\ x)=p_l \quad (x\in M_{I+1,1}),$$
$$succ_1(p_l,\ x')=r^t_1 \quad (x'\in M_{1,I+1}).$$
$$succ_1(s_3,\ x)=p_3 \quad (x\in M_{I+1,1}),$$
$$succ_1(p_3,\ x')=r^f_1 \quad (x'\in M_{1,I+1}),$$
$$succ_1(r^t_1,\ t)=r^t_2 \quad (t\in M_{1,2}),$$
$$succ_1(r^t_2,\ t')=r^t_3 \quad (t'\in M_{2,1}),$$
$$succ_1(r^f_1,\ f)=r^f_2 \quad (f\in M_{1,2}),$$
$$succ_1(r^f_2,\ f')=r^f_3 \quad (f'\in M_{2,1}).$$

The additional part of the i-th process $(2\leqq i\leqq I)$ is constructed in the following manner. In the following expressions, if $i\neq n$ then $j=i+1$ and if $i=I$ then $j=I+2$.

$$Q^A_i=\{p^t_l,\ p^f_l \mid 0\leqq l\leqq 3\}\cup\{r^t_1,\ r^t_2,\ r^t_3,\ r^f_1,\ r^f_2,\ r^f_3\},$$
$$M^A_{i,i-1}=\{t',\ f'\},$$
$$M^A_{i,j}=\{t,\ f\}.$$

For each l $(0\leqq l\leqq 2)$,

$$succ_i(s_l,\ t)=p^t_l \quad (t\in M_{i-1,i}),$$
$$succ_i(p^t_l,\ t')=r^t_1 \quad (t'\in M_{i,i-1}),$$
$$succ_i(s_l,\ f)=p^f_l \quad (f\in M_{i-1,i}),$$
$$succ_i(p^f_l,\ f')=r^f_1 \quad (f'\in M_{i,i-1}).$$
$$succ_i(s_3,\ t)=p^t_3 \quad (t\in M_{i-1,i}),$$
$$succ_i(p^t_3,\ t')=r^f_1 \quad (t'\in M_{i,i-1}),$$
$$succ_i(s_3,\ f)=p^f_3 \quad (f\in M_{i-1,i}),$$
$$succ_i(p^f_3,\ f')=r^f_1 \quad (f'\in M_{i,i-1}),$$
$$succ_i(r^t_1,\ t)=r^t_2 \quad (t\in M_{i,j}),$$
$$succ_i(r^t_2,\ t')=r^t_3 \quad (t'\in M_{j,i}),$$
$$succ_i(r^f_1,\ f)=r^f_2 \quad (f\in M_{i,j}),$$
$$succ_i(r^f_2,\ f')=r^f_3 \quad (f'\in M_{j,i}).$$

The additional part of the $(I+2)$-th process is constructed in the following manner.

$$Q^A_{I+2}=\{r_0,\ p^t,\ p^f,\ q^s,\ q^u_1,\ q^u_2,\ q^u_3\},$$
$$o_{I+2}=r_0,$$

$M^A_{I+2,I}=\{t', f'\}$,

$M^A_{I+2,I+3}=\{u\}$,

$succ_{I+2}(r_0, f)=p^f \quad (f\in M_{I,I+2})$,

$succ_{I+2}(p^f, f')=q^s \quad (f'\in M_{I+2,I})$,

$succ_{I+2}(r_0, t)=p^t \quad (t\in M_{I,I+2})$,

$succ_{I+2}(p^t, t')=q^u_1 \quad (t'\in M_{I+2,I})$,

$succ_{I+2}(q^u_1, u)=q^u_2 \quad (u\in M_{I+2,I+3})$,

$succ_{I+2}(q^u_2, u')=q^u_3 \quad (u'\in M_{I+3,I+2})$.

There does not exist any additional part in the $(I+3)$-th process.

It is obvious that the above construction of a protocol P based on B can be done in polynomial time. Any unspecified receptions do not appear in the fundamental part of protocol P.

The constructed protocol P behaves as follows. If B is satisfiable, then there exists a case such that all the 1-st process, the 2-nd process, $\cdots$, and the I-th process do not reach state s_3. In this case, each of these processes transmits a message t rather than a message f to another process. Consequently the $(I+2)$-th process receives a message t, and transmits a message u to the $(I+3)$-th process. Since no reception of the message u is specified in the $(I+3)$-th process, an unspecified reception occurs here. If B is not satisfiable, then at least one of the 1-st process, the 2-nd process, $\cdots$, and the I-th process reaches state s_3. Suppose that the i-th process $(1\leqq i\leqq I)$ reaches state s_3. The i-th process transmits a message f and each the j-th process $(i\leqq j\leqq I)$ transmits messages f to another process. Consequently the $(I+2)$-th process receives a message f, and unspecified receptions do not occur. Therefore, there exist unspecified receptions in P if and only if B is satisfiable.

The constructed protocol P is fixed to h $(h=1)$ because of the similar reason in the proof of Theorem 1.

Therefore, 3SAT is polynomially transformable to URDP for fixed channel capacity. Since URDP is in NP obviously, URDP is proven to be NP-complete.

(End of Proof)

Theorem 4: ODP is NP-complete for a positive integer h $(h=1)$ even when protocols are fixed to 2.

(End of Theorem)

Proof 4 (Theorem 4): To prove that Theorem 4 holds, a protocol P based on a boolean expression B in conjunctive normal form is also constructed. Since the construction of the protocol P is the same as that in the proof of Theorem 3 except the construction of the $(I+2)$-th process, only the additional part of the $(I+2)$-th process is described below.

The $(I+2)$-th process is constructed in the following manner.

$Q^A_{I+2}=\{r_0, p^t, p^f, q^s, q^u_1, q^o_1, q^o_2, q^o_3, q^o_4\}$,

$o_{I+2}=r_0$,

$M^A_{I+2,I}=\{t', f'\}$,

$M^A_{I+2,I+3}=\{z\}$,

$succ_{I+2}(r_0, f)=p^f \quad (f\in M_{I,I+2})$,

$succ_{I+2}(p^f, f')=q^s \quad (f'\in M_{I+2,I})$,

$succ_{I+2}(r_0, t)=p^t \quad (t\in M_{I,I+2})$,

$succ_{I+2}(p^t, t')=q^u_1 \quad (t'\in M_{I+2,I})$,

$succ_{I+2}(q^u_1, z)=q^o_1 \quad (z\in M_{I+2,I+3})$,

$succ_{I+2}(q^o_1, z)=q^o_2 \quad (z\in M_{I+2,I+3})$,

$succ_{I+2}(q^o_2, z')=q^o_3 \quad (z'\in M_{I+3,I+2})$,

$succ_{I+2}(q^o_3, z')=q^o_4 \quad (z'\in M_{I+3,I+2})$.

It is obvious that the above construction of a protocol P based on B is polynomially transformable and in the protocol all channels except a channel from the $(I+2)$-th process to the $(I+3)$-th process are fixed to 1 and the channel from the $(I+2)$-th process to the $(I+3)$-th process is fixed to 2.

If B is satisfiable, then the $(I+2)$-th process transmits a message z twice. Suppose that a positive integer h $(h=1)$ is given. There exist an overflow for

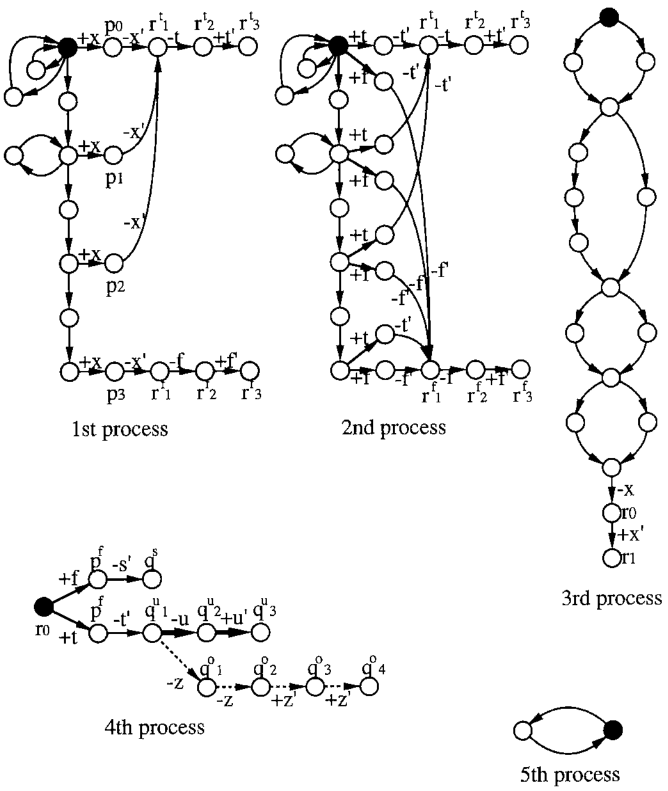


Fig. 4 Example of transformations for unspecified reception detection problem and overflow detection problem.

the positive integer h $(h=1)$ if and only if B is satisfiable.

Therefore, 3SAT is polynomially transformable to ODP for the protocol fixed to 2 and the positive integer h $(h=1)$. Since the restricted ODP is in NP obviously, the restricted ODP is proven to be NP-complete.

(End of Proof)

Example 3: An example of the transformation from a boolean expression $B=(v_1+v_2+v_3)\times(\neg v_1+v_2+v_4)$ to a protocol P is shown in Fig. 4. Any symbols in the figure are the same as those in Figs. 2 and 3. The part for URDP is depicted by solid lines and bold lines. The part for ODP is the same as that for URDP except addition of state transitions depicted by dotted lines and deletion of state transitions depicted by bold lines.

(End of Example)

5. Conclusions

It is known that in order to solve the deadlock detection problem, the unspecified reception detection problem and the overflow detection problem for bounded protocols we have only to enumerate reachable global states [6]. The number of such global states is at most $|Q|^N((|M|+1)^h)^{N\cdot N}$, where $|Q|$ is the number of states, N is the number of processes, $|M|$ is the number of distinct message types and h is the capacity of bounded channels. If both the capacity h and the number of processes N are fixed, then these problems can be solved in a polynomial time.

This paper has shown their NP-completeness in case of fixed h $(h=1)$ and unlimited N. Räuchle and Toueg showed PSPACE-completeness of the deadlock detection problem in case of finite h and limited $N(N=2)$ [5]. Therefore, the new result on the deadlock detection problem clarifies the boundary of computational complexity of the problem, which is shown in Fig. 5.

Since the NP-completeness of the deadlock detection problem, the unspecified reception detection problem and the overflow detection problem were shown, there does not seem to exist any algorithms efficient in the worst case, even when all channels in protocols are fixed to 1 or 2. This suggests that, for protocols without any restriction on the number of processes, algorithms efficient in average should be developed [3], [4].

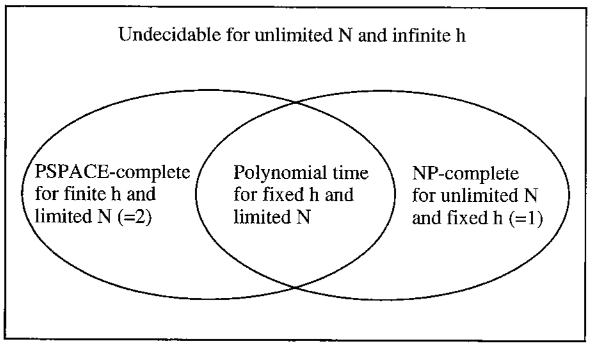

Undecidable for unlimited N and infinite h
PSPACE-complete for finite h and limited N (=2)
Polynomial time for fixed h and limited N
NP-complete for unlimited N and fixed h (=1)

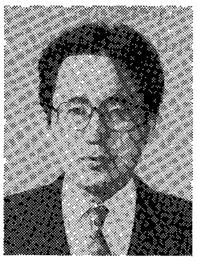
Fig. 5 Computational complexity of deadlock detection problem on protocol validation.

Acknowledgments

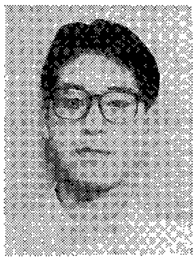
The authors would like to thank the referees for their valuable suggestions and constructive comments.

References

[1] Aho, A. V., Hopcroft, J. E. and Ullman, J. D., *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA., 1974.
[2] Brand, D. and Zafiropulo, P., "On communicating finite state machines," *J. Assoc. Comp. Mach.*, vol. 30, no. 2, pp. 323-342, Apr. 1983.
[3] Kakuda, Y., Wakahara, Y. and Norigoe, M., "An acyclic expansion algorithm for fast protocol validation," *IEEE Trans. Software Eng.*, vol. SE-14, no. 8, pp. 1059-1070, Aug. 1988.
[4] Kakuda, Y., Wakahara, Y. and Norigoe, M., "Communication protocol validation for protocols with unbounded overflow," *IEICE Technical Report*, EC84-65, Feb. 1985.
[5] Räuchle, T. and Toueg, S. "Exposure to deadlock for communicating processes is hard to detect," *Inform. Proc. Letters*, vol. 21, pp. 63-68, Aug. 1985.
[6] West, C. H. "General technique for communications protocol validation," *IBM J. Res. Devel.*, vol. 22, no. 4, pp. 394-404, Jul. 1978.



Yoshiaki Kakuda was born in Hiroshima, Japan, on June 29, 1955. He received the B.S. degree in electronic engineering from Hiroshima University, Hiroshima, Japan, in 1978. He also received the M.S. degree and Ph.D. degree in system engineering from the same university in 1980 and 1983, respectively. From 1983 to 1991, he was with Research and Development Laboratories, Kokusai Denshin Denwa Co., Ltd. (KDD), where he last held the position of Senior Research Engineer. From 1991 he has been an Associate Professor with the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. His current research interests include protocol engineering and responsive systems. He was a program co-chair man of the Second International Workshop on Responsive Computer Systems in 1992. He is a member of the IEEE Computer Society and the Information Processing Society of Japan. He received the Telecom. System Technology Award from Telecommunications Advancement Foundation in 1992.



Yoshihiro Takada was born in Kagawa, Japan, on September 30, 1964. He received the B.M. and M.E. degrees in information and computer sciences in 1989 and 1991 respectively from Osaka University, Osaka, Japan. He is currently a doctorial candidate in Osaka University. His research interests include human factors in software development. He is a member of Information Processing Society of Japan and Japan Acoustic Society.

Tohru Kikuno was born in Ehime, Japan, on September 11, 1947. He received the B.E., M.Sc., and Ph.D. degrees in electrical engineering from Osaka University, Toyonaka, Osaka, Japan, in 1970, 1972, and 1975, respectively. He joined Hiroshima University from 1975 to 1987. He is currently a Professor in the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University since 1990. His research interests include analysis and design of fault-tolerant systems, quantitative evaluation of software development process and design of testing procedure of computer protocols. He was a general co-chairman of the Second International Workshop on Responsive Computer Systems in 1992. He is a member of the IEEE (U.S.A), ACM (U.S.A.) and Inf. Proc. Soc. (Japan). He received the Paper Award from the Institute of Electronics, Information, and Communication Engineers of Japan in 1993.