ソフトウェア基礎論19－3
プログラミング言語 9－3
（1986. 12. 12）

通信プロトコルにおける オーバーフローの多項式時間検証法

A Polynomial Time Validation Algorithm for Overflows in Communication Protocols

菅沼 知久　　黒沢 馨
Tomohisa SUGANUMA　　Kaoru KUROSAWA

東京工業大学 物理情報工学専攻
Dpt. of Infomation Processing Tokyo Institute of Technology

あらまし　通信プロトコルは、多くの場合、２つの有限状態機械間の通信という形でモデル化される。この様にモデル化されたプロトコルにおいて、メッセージが一種類の場合、デッドロックの検証は状態数に関する多項式時間でできることが既に証明されている。本稿はメッセージが一種類の場合、無限オーバーフローの検証も多項式時間でできることを示すものである。この結果は、有限オーバーフローの検証時間の上限をあたえるとともに、全てのメッセージを一種類に変換したプロトコルにオーバーフローが存在しなければ、元のプロトコルにもオーバーフローが存在しないという意味において重要である。

Abstract　Many communication protocols are modeled by two communicating finite state machines. A polynomial time algorithm for deadlock detection has been known for the protocols with one type message. This paper shows a polynomial time algorithm which detects infinite overflows in such protocols. The result is useful because any protocol is free from overfolows if its all different messages are converted to one type and no overflow exists in the modified protocol.

1. まえがき

オンラインやデータベース等のように、いくつかの装置が各々連絡をとりあって動作し全体として所定の目的を達成するようなシステムが今日ではますます重要になってなってきているが、このようなシステムを実現するためには、各装置の間でのデータのやり取り、即ち通信が不可欠である。この通信を行うためには計算機、端末などの複数の各通信装置の間に１つの通信に関する約束が必要である。この通信の約束のことを通信プロトコルと言う。

一方、このような複数の装置から成り立つシステムが正しく動作するためには、プロトコルが要求される仕様を満足するとともに、論理的な矛盾を含んでいない必要がある。プロトコルが複雑になるにともない、これらのことを検証する効率のよい方法の開発が望まれている。

プロトコルは多くの場合、２つの有限状態機械の通信という形でモデル化される。このモデル化されたプロトコルに於て、メッセージが一種類の場合、デッドロックの検証は多項式時間でできることが既に証明されている[1]。しかし、無限オーバーフローの検証時間に関してはこれまでほとんど報告されていない（表１参照）。ただし、無限オーバーフローとはチャンネル上の未処理メッセージの数が無限大となる状態で、通信プロトコルのエラーの一つである。

本稿はオーバーフローの検証法の開発を目的とし、その結果としてメッセージが一種類の場合、無限オーバーフローの検証が多項式時間でできることを証明する。この結果は、有限オーバーフローの検証時間の上限をあたえるとともに、全てのメッセージを一種類に変換したプロトコルにオーバーフローが存在しなければ、元のプロトコルにもオーバーフローが存在しないという意味において重要である。

表１　２プロセス間のプロトコルの検証

プロトコルの種類	一　般	メッセージ一種類
デッドロック	決定不能[2]	多項式時間[1]
ライブロック	決定不能[3]	決定可能[3]
オーバーフロー	?	本　稿

2. 準備

2.1 状態遷移図

通信を行う各プロセス（計算機等）は、ラベル付き有向グラフである状態遷移図によって記述される。状態遷移図の各枝は $-a_i$ または $+a_i$ とラベル付けされ、

$-a_i$ はメッセージ a_i の送信
$+a_i$ はメッセージ a_i の受信

を表す（図１参照）。

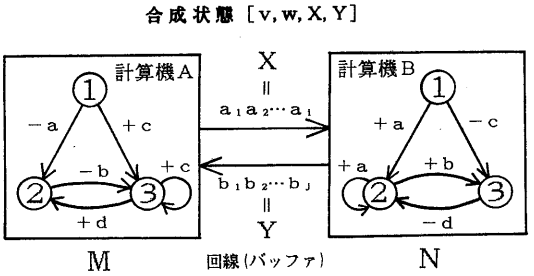

図１　プロトコルのモデル

また、ラベルの種類によって各枝はつぎの２つのいずれかに分類される（図２参照）。

sending edge : $-a_i$ とラベル付けされている枝
receiving edge : $+a_i$ とラベル付けされている枝

また、各状態遷移図の特定の１つの節点は初期状態と呼ばれる。

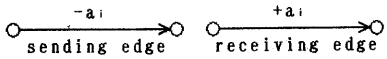

図２　sending edge と receiving edge

2.2 合成状態

通信を行う２つの有限状態機械ＭとＮの合成状態は、[v,w,X,Y] で定義される。ただし、

v : Ｍの状態
w : Ｎの状態
X : ＭからＮへのチャンネル上のメッセージの系列
Y : ＮからＭへのチャンネル上のメッセージの系列

また、$[v_0, w_0, \varepsilon, \varepsilon]$ は初期合成状態と呼ばれる。ただし、

v_0 : Ｍの初期状態
w_0 : Ｎの初期状態
ε : 空系列

2.3 無限オーバーフロー

チャンネル上のメッセージ数 |X| または |Y| が無限大となる到達可能な合成状態 [v,w,X,Y] が存在するとき、この状態を無限オーバーフローと呼ぶ。実際のシステムにおいてはこれらのメッセージはある容量を持ったバッファに蓄えられるので、無限オーバーフローが存在するプロトコルは実現不可能である。

2.4 メッセージが一種類のプロトコル

有限状態機械ＭとＮのすべてのメッセージを一種類に変換した有限状態機械を $\widetilde{M}$, $\widetilde{N}$ とする（図１は図３となる）。$\widetilde{M}$, $\widetilde{N}$ における合成状態 [v,w,x,y] を以下のように定義する。

v,w : $\widetilde{M}$, $\widetilde{N}$ の状態

x ： $\widetilde{M}$から$\widetilde{N}$へのチャンネル上のメッセージ数

y ： $\widetilde{N}$から$\widetilde{M}$へのチャンネル上のメッセージ数

以後はメッセージが一種類のプロトコルのみについて考える。

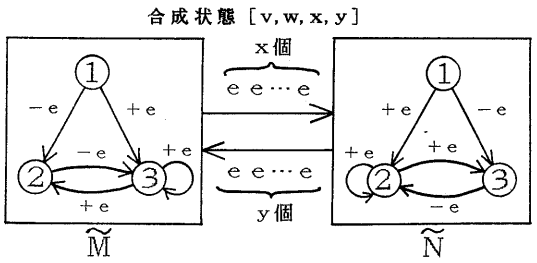

図３　メッセージが一種類のときのモデル

2.5　合成状態遷移図

合成状態遷移図は、初期状態から到達可能な合成状態を節点とし、それらの間の可能な遷移を有向枝で示した有向グラフである。

図３から導かれる合成状態遷移図の一部を図４に示す。

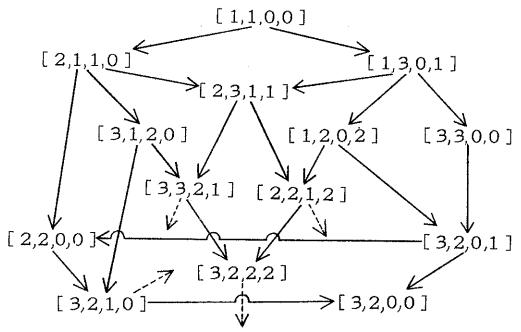

図４　合成状態遷移図

2.6 canonical form[4]

有限状態機械ＭとＮの遷移が対になって起こる（Ｍ，Ｎともに送信、Ｍ，Ｎともに受信、Ｍ受信，Ｎ送信、Ｍ送信，Ｎ受信の４つのパターンのいずれか）と制限した場合の合成状態遷移図を canonical form という（図５参照）。

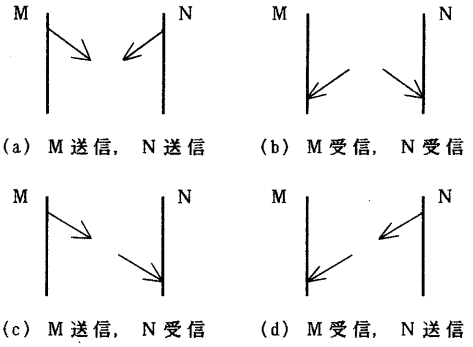

図５　canonical form の状態遷移

図３から導かれる canonical form の一部を図６に示す。

［補題２］

canonical form においては、

（ＭからＮへのチャンネル上のメッセージ数）＝（ＮからＭへのチャンネル上のメッセージ数）である。

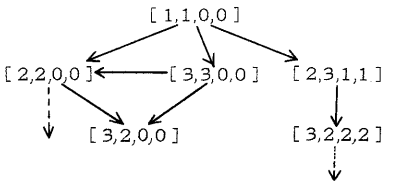

図６　canonical form

3．２プロセスのプロトコルにおける無限オーバーフローの多項式時間検証法

２プロセスの通信プロトコルにおいて次の補題が成り立つ。

［補題１］

メッセージを一種類に変換したプロトコル$\widetilde{M}$，$\widetilde{N}$に無限オーバーフローが存在しなければなければ、もとのプロトコルＭ，Ｎにも無限オーバーフローは存在しない。

（証明）

もとのプロトコルにおいて到達可能な任意の

合成状態が、メッセージを一種類に変換したプロトコルにおいても到達可能であることを示せばよい。

もとのプロトコルの合成状態遷移において、

$$s_i=[v_i, w_i, X_i, Y_i] \rightarrow s_{i+1}=[v_{i+1}, w_{i+1}, X_{i+1}, Y_{i+1}]$$

が遷移可能ならば、明らかにメッセージを一種類に変換したプロトコルにおいても、

$$\tilde{s}_i=[v_i, w_i, |X_i|, |Y_i|] \rightarrow \tilde{s}_{i+1}=[v_{i+1}, w_{i+1}, |X_{i+1}|, |Y_{i+1}|]$$

が遷移可能である。

よって、もとのプロトコルにおいて到達可能な合成状態は、メッセージを一種類に変換したプロトコルにおいても到達可能である。

（証明終わり）

3.1 同期形プロトコル

2つの有限状態機械における遷移が対となって起こるとき、これを同期形プロトコルと呼ぶ。また、その合成状態遷移図は canonical form となる。よって、次の定理が成り立つ。

［定理１］

メッセージが一種類の同期形プロトコルにおいて、無限オーバーフローが存在する必要十分条件は、チャンネル上のメッセージ数が mn である合成状態が存在することである（図７）。

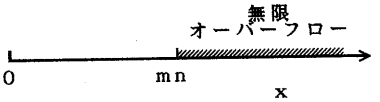

図７　定理１

（証明）

必要性：チャンネル上のメッセージ数が mn 以上になる合成状態が存在するので、メッセージ数が mn となる合成状態も必ず存在する。

十分性：初期合成状態 $s_0=[v_0, w_0, 0, 0]$ から $s=[v, w, mn, mn]$ への遷移が存在したとする。これを $s_0 \rightarrow s_1 \rightarrow \cdots \rightarrow s_r (s=s_r)$ とする。各遷移では高々１しかチャンネル上のメッセージ数は増加しないので、$r \geq mn$ である。ここで、v と w の組合せは mn 個である。一方、s_0 から s_r までの合成状態の数は mn+1 以上あるので、この中に少なくとも１組の $[v_i, w_i]$ が等しい合成状態が存在する。これを、$s_i=[v_i, w_i, x_i, x_i]$, $s_j=[v_i, w_i, x_j, x_j]$ $(s_i \rightarrow \cdots \rightarrow s_j)$とする。ここで、$x_i-x_j=k \geq 0$ の場合には、合成状態遷移図に s_i から s_j への遷移を取り除いた遷移系列も存在する。この系列上では、$s_r=[v, w, mn+k, mn+k]$ となるので、$q \leq r$ である $s_q=[v_q, w_q, mn, mn]$ が存在する。この q を新たな r として上記の操作を繰り返すと、ついには、$x_j-x_i=k>0$ となる s_i, s_j が存在する。ゆえに、この s_i から s_j への遷移を繰り返した、

$$\begin{aligned} s_0 \rightarrow \cdots \rightarrow s_i=[v_i, w_i, x_i, x_i] &\rightarrow \cdots \rightarrow s_j=[v_i, w_i, x_i+k, x_i+k] \\ &\rightarrow \cdots \rightarrow [v_i, w_i, x_i+2k, x_i+2k] \\ &\vdots \\ &\rightarrow \cdots \rightarrow [v_i, w_i, x_i+hk, x_i+hk] \quad (h=1, 2, \cdots) \end{aligned}$$

という遷移が合成状態遷移図に存在するので無限オーバーフローが起こる（図８参照）。

（証明終わり）

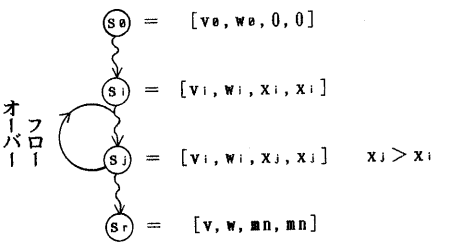

図８　同期形プロトコルのオーバーフロー

［系１．１］

同期型プロトコルにおいて、無限オーバーフローが存在するかどうかは $o(m^3n^3)$ で判定できる（証明略）。

3.2 非同期形プロトコル（特殊な場合）

2つの有限状態機械における遷移のスピードが必ずしも等しくないとき、これを非同期形プロトコルと呼ぶ。一般のプロトコルは非同期形

である。

［定理２］

メッセージが一種類の非同期形プロトコルにおいて、以下の条件を満たすある正の整数 k が存在するならば、無限オーバーフローが存在する。

初期状態 s_0 から $s=[v,w,x,y]$（ただし、$Max(x,y)=kmn$）への遷移が存在し、その遷移系列上の各合成状態 $s_i=[v_i,w_i,x_i,y_i]$ において、$Min(x_i,y_i)<k$ （図９参照）。

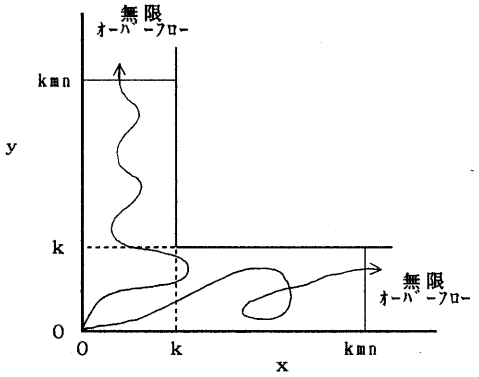

図９　定理２の説明

（証明）

$x=kmn$ となる場合について考える（$y=kmn$ の場合も同様に証明できる）。

s_0 から s への遷移を $s_0 \rightarrow s_1 \rightarrow \cdots \rightarrow s_r$（$s=s_r$, $s_i=[v_i,w_i,x_i,y_i]$）とする。ここで、$Max(x_i,y_i)=k$ となる最初の合成状態を s_h、$x_i=k$ となる最後の i を h' とする。すると、各遷移では高々１しか x_i, y_i は増加しないので、

$$h \geqq k$$
$$r - h' \geqq kmn - k$$

である。よって、s_0 から s_{h-1} まで遷移の中に k 個、$s_{h'}$ から s_r までの遷移の中に kmn-k+1 個、合計 kmn+1個以上の $y_i<k$ である合成状態が存在する（図１０参照）。一方、$y_i<k$ である v と w と y の組合せは kmn であるから、s_0 から s_{h-1} までと $s_{h'}$ から s_r までの合成状態の中に最低１組の $[v_i,w_i,y_i]$ が等しい合成状態 $s_i=[v_i,w_i,x_i,y_i]$ と $s_j=[v_i,w_i,x_j,y_i]$ ($s_i \rightarrow \cdots \rightarrow s_j$) が存在する。ここで、$x_i \geqq x_j$ ならば、定理１の証明と同様に s_i から s_j への遷移を取り除いた s_0 から $s_q=[v_q,w_q,x_q,kmn]$（$q \leqq r$）への遷移が存在する。この q を新たな r として上記の操作を繰り返すと、必ず $x_i<x_j$ となる s_i, s_j が存在する。ゆえに、s_i から s_j への遷移を繰り返すことにより、無限オーバーフローが起こる。

（証明終わり）

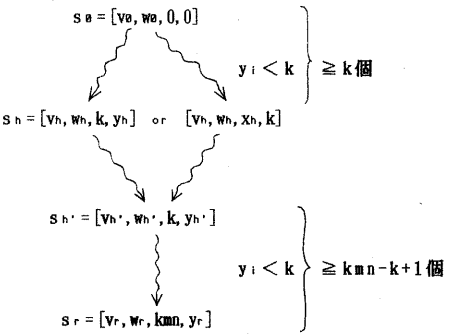

図１０　s_0 から s_r への遷移

この非同期形のプロトコルにおいて片方のチャンネル上のメッセージ数がある正の整数 k で制限されているとき、定理２より次の系２．１、２．２が成り立つ。

［系２．１］

メッセージが一種類で常に $x<k$ である非同期プロトコルにおいては、$y=kmn$ となることが（$y<k$ ならば、$x=kmn$ となることが）、無限オーバーフローが存在する必要十分条件である。

［系２．２］

任意の合成状態 $[v_i,w_i,x_i,y_i]$ において $Min(x_i,y_i)<k$ であるとき、無限オーバーフローが存在するかどうかは $o(k^2m^3n^3)$ で判定できる（証明略）。

3.3 非同期形プロトコル（一般の場合）

［定理３］

メッセージが一種類のプロトコルにおいて、$s=[v,w,x,y]$ が到達可能ならば、任意の整数 z（$0 \leqq z \leqq Min(x,y)$）に対して canonical form

において到達可能な s'=[v', w', z, z] が少なくとも一つ存在する。

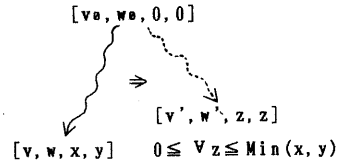

図１１　定理３

（証明）

x≧y の場合について考える（x≦yの時も同様にして証明できる）。

s が到達可能であるから、$s_0 \to s_1 \to \cdots \to s_r$（$s_0$は初期状態、$s=s_r$）という状態遷移が存在する。また、$s_i$ から s_{i+1} へは枝 f_{i+1} を通って遷移するとする。ここで、f_i (i=0, 1, …, r-1) はMまたはNの枝である。この枝の列をFとし、$F=f_1, f_2, \cdots, f_r$を、Mの枝の列 $D=d_1, d_2, \cdots, d_h$ とNの枝の列 $E=e_1, e_2, \cdots, e_{h'}$ とに分ける。ここで、DおよびEに含まれる sending edge と receiving edge の数をそれぞれ $n_s(D)$, $n_r(D)$, $n_s(E)$, $n_r(E)$ とすると、２つのチャンネルに残っているメッセージ数は x と y であるから、

$$n_s(D) - n_r(E) = x$$
$$n_s(E) - n_r(D) = y$$

である。よって、k=x-y（≧0）とすると、

$$n_s(D) + n_r(D) = n_s(E) + n_r(E) + k$$

より、

$$h = h' + k \geqq h'$$

となる。

つぎに、Mにおいて枝 d_i (i=0, 1, …, h) は v_i から v_{i+1} へ、Nにおいて枝 e_i (i=0, 1, …, h') は状態 w_i から w_{i+1} へのびているとする。

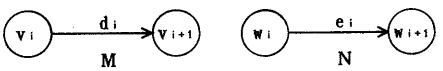

図１２　各枝と節点との関係

canonical form における状態遷移 $s'_0 \to s'_1 \to \cdots \to s'_{h'}$ を次のような規則によって作る。

1) s'_0 を初期合成状態とする。

2) $s'_i=[v_i, w_i, x_i, x_i]$ (i=1, 2, …, h') とする。ただし、x_i は以下のようにして決める。

a)もし、d_i が sending edge で e_i が receiving ege ならば、$x_i=x_{i-1}$。

b)もし、d_i が receiving edge で e_i が sending ege ならば、$x_i=x_{i-1}$。

c)もし、d_i と e_i が sending ege ならば、$x_i=x_{i-1}+1$。

d)もし、d_i と e_i が receiving ege ならば、$x_i=x_{i-1}-1$。

s'_0 から $s'_{h'}$ への遷移が可能であることを示すには、$x_i \geqq 0$ (i=1, 2, …, h') を示せばよい。

ここで、次のようなD, Eの部分列を考える。

$$D' = d_1, d_2, \cdots, d_i$$
$$E' = e_1, e_2, \cdots, e_i \quad (i=1, 2, \cdots, h')$$

Fの中で d_i が e_i の後ろに現れるとして、次のようなFの部分列を考えると、

$$F' = f_1, f_2, \cdots, e_i, \cdots, d_i$$

F'の中のMの sending edge の数は $n_s(D')$ である。F'の中のNの receiving edge の数を $n_r(E'')$ とすると、次の式が成り立つ。

$$n_r(E'') \geqq n_r(E') \qquad (1)$$

F'はFの部分列であるから、F'による遷移は可能である。よって、

$$n_s(D') - n_r(E'') \geqq 0 \qquad (2)$$

である。(1), (2)式より、

$$n_s(D') - n_r(E') \geqq n_s(D') - n_r(E'') \geqq 0 \qquad (3)$$

となる。ここで、

$$n_s(D') + n_r(D') = n_s(E') + n_r(E') = i$$

であるから、(3)式より、

$$x_i = n_s(D') - n_r(E') = n_s(E') - n_r(D') \geqq 0 \qquad (4)$$

となる。e_i が d_i の後ろに現れる場合にも同様にして(4)式が成立する。

以上より、canonical form においても $s'_{h'}$ は到達可能である。ここで、

$$D_1 = d_1, d_2, \cdots, d_{h'}$$
$$D_2 = d_{h'+1}, \cdots, d_h$$

とすると、

$$n_s(E) - n_r(D_1) - n_r(D_2) = y$$

であるから、

$$x_{h'} = n_s(E) - n_r(D_1)$$
$$= y + n_r(D_2)$$
$$\geqq y$$

となる。各遷移において x_i は高々１しか増えないので、s'_0 から $s'_{h'}$ への遷移の途中において $x'_k=z(0 \leqq \forall z \leqq y)$ となる s'_k が必ず存在する。よって、$s'=s_k$ は到達可能である。

（証明終わり）

定理３から次の系３．１が得られる。

［系３．１］

メッセージが一種類のプロトコルにおいて２つのチャンネル上のメッセージ数が等しい合成状態 $s=[v,w,x,x]$ が合成状態遷移で到達可能である必要十分条件は、canonical form においても s が到達可能であることである。

（証明）

必要性：caninical form は合成状態遷移図の部分グラフである。よって、canonical form において s が到達可能であれば合成状態遷移においても到達可能である。

十分性：定理３において、y=x, z=x とすることにより、合成状態遷移で $s=[v,w,x,x]$ が到達可能ならば、canonical form において到達可能なある $s'=[v',w',x,x]$ が存在する。この場合、定理３の証明における s' の作り方から v'=v, w'=w、すなわち、s'=s となる。よって、s は canonical form においても到達可能である。

（証明終わり）

メッセージが一種類でチャンネル上のメッセージ数の上限が保証されていない一般のプロトコルにおいて、無限オーバーフローの検証に役立つ以下の定理が成り立つ。

［定理４］

メッセージが一種類の非同期形のプロトコルにおいて、$s=[v,w,mn,mn]$ が到達可能ならば無限オーバーフローが存在する。

（証明）

s が到達可能であるならば系３．１より、s は canonical form においても到達可能である。よって、定理１より無限オーバーフローが存在する。

（証明終わり）

［定理５］

メッセージが一種類のプロトコルにおいて、無限オーバーフローが存在するための必要十分条件は、$s=[v,w,x,y]$（$Max(x,y)=m^2n^2$ または $Min(x,y)=mn$）が到達可能であることである。

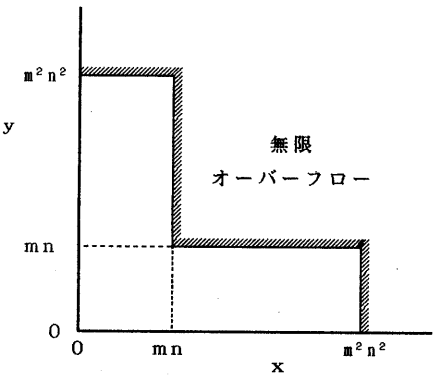

図１３　無限オーバーフローする領域

（証明）

必要性：Max(x,y)が m^2n^2 以上になるので、$Max(x,y)=m^2n^2$ となる合成状態が必ず存在する。

十分性：初期状態 s_0 から $s=s_r$ への各遷移状態 s_i（i=1,2,…,r）において $s_i=[v_i,w_i,x_i,y_i]$（$Min(x_i,y_i)=mn$）という合成状態が存在した場合、定理３により [v',w',mn,mn] が到達可能であるから、定理４により無限オーバーフローが存在する。そうでない場合には、片方のチャンネル上のメッセージ数が mn 未満であり、かつ、もう片方のチャンネル上のメッセージ数が m^2n^2 であるから、定理２（k=mn）により無限オーバーフローが存在する。

（証明終わり）

２プロセスでメッセージが一種類のプロトコルに無限オーバーフローが存在するかどうかを判定するアルゴリズムを以下に示す。

［準備］

有限状態機械Ｍの各状態に１からｍの番号を、有限状態機械Ｎの各状態に１からｎの番号をつけておく（特に初期状態は１とする）。また、各合成状態 [i, j, x, y]（Min(x, y)＜mn かつ Max(x, y)＜m^2n^2）に対応する $2m^4n^4-m^3n^3$ 個の４次元配列 TEST(i, j, x, y) を用意する。

［メッセージが一種類のプロトコルの無限オーバーフローを検証するアルゴリズム］

```
(step 1)
begin
  TEST(i,j,x,y) := 未生成
                   for all i,j,x,y
  TEST(1,1,0,0) := 生成済
  OPEN  := {[1,1,0,0]}
(step 2)
  while  OPEN ≠ φ  do
    (a) OPENから要素を１つ取り出し、
        これを[i',j',x',y']とする。
    (b) [i',j',x',y'] から生成可能な全ての
        合成状態を生成する。
    (c) if  その中に Max(x,y)=m²n² または
        Min(x,y)=mn となる状態がある。
        then  stop  {無限オーバーフローが
                     存在する。}
        else  TEST(i,j,x,y) = 未生成
        であるすべての生成された合成状態
        [i,j,x,y] に対し、TEST(i,j,x,y)
        := 生成済 とし、かつ、[i,j,x,y]
        を OPEN に加える。
end.
```

step 1 に要する時間は明らかに $o(m^4n^4)$ である。

$2m^4n^4 - m^3n^3$ 個の合成状態 [i, j, x, y] が、OPEN に加えられるのは高々１回である。また、OPEN 中の１つの合成状態 [i', j', x', y'] から生成される合成状態は、高々 4mn 個である。したがって、step 2 に要する時間は $o(m^5n^5)$ である。よって、次の系が成り立つ。

［系５．１］

メッセージが一種類のプロトコルにおいて、無限オーバーフローが存在するかどうかは $o(m^5n^5)$ で判定できる。

４．まとめ

メッセージが一種類のプロトコルの無限オーバーフローの検証が多項式時間で可能なことを示した。対象とする各プロトコルの種類と検証時間を表２に示す。

表２　無限オーバーフローの検証時間

対象とするプロトコル	オーダー	判定条件
同期形	m^3n^3	定理１
片方のチャンネル有限（＜k）	$k^2m^3n^3$	定理２
一般	m^5n^5	定理５

ただし，m はM、n はNの状態数。
k は正の整数。

［参考文献］

[1] 黒沢，桑原，茂野：「分脈自由文法を利用した通信プロトコルの一検証法」信学技報 AL85-85(1986)

[2] D. Brand and P. Zafiropulo: "On Communicating finite-state Machines", IBM Res. Rep. RZ1053, 1981

[3] M. G. Gouda, C. H. Chow, and S. S. Lam: "On the Decidability of Livelock Detection in Networks of Communicating Finite State Machines", Proc. 4th IFIP Workshop on Protocol Specification, Testing, and Verification, edited by Y. Yemini, R. Strom, and S. Yemini, North-Holland, 1985, pp. 47-56.

[4] J. Rubin and C. H. West: "An Improved Protcol Validation Technique", Computer Networks, vol. 6, pp. 65-73(1982)