計算機アーキテクチャ　103－6
設　計　自　動　化　69－6
(1993. 12. 16)

並列論理シミュレーションにおけるロールバックについて

世古　忠† 菊野　亨‡
† 奈良工業高等専門学校情報工学科
〒639-11 大和郡山市矢田町 22
‡大阪大学基礎工学部情報工学科
〒560 豊中市待兼山町 1-1

あらまし

本稿では，並列論理シミュレーションの高速化を Jefferson の楽観的方法に基づいて実現したときに発生するロールバックの問題について基礎的考察を行う．ロールバックは並列論理シミュレーションの性能を低下させる大きな要因であると考えられるが，これまでそのメカニズムについての解明はほとんどなされていない。ここでは，並列論理シミュレーションの詳細モデルについて説明した後，モデルに基づいてロールバックの発生の幾つかの事例について分析する．具体的には，ロールバックの発生を考慮したシミュレーション時間の定式化を与え，ロールバック発生要因の分析を行なった．更に，簡単な木状回路と規則的な入力信号の例を用いて，シミュレーションの条件を変化させたときのロールバック発生回数への影響を観測することによって，ロールバック発生要因についての実験的評価を行なった．

和文キーワード：並列論理シミュレーション，ロールバック

On Rollbacks in Parallel Logic Simulation for Tree Connected Circuit

Tadashi SEKO† and Tohru KIKUNO‡
† Nara National College of Technology
22 Yata, Yamatokoriyama, Nara 639-11, Japan
‡Faculty of Engineering Science, Osaka University
1-1 Machikaneyama, Toyonaka, Osaka 560, Japan

Abstract

We analyze rollback that commonly takes place in the optimistic implementation of parallel logic simulations. After presenting a detailed simulation model, we propose a formula which calculates simulation time based on three kinds of times π_{eval}(for event evaluation), τ_{oh}(for rollback processing) and τ_{idle}(for waiting event idle). Then, we take a simple tree-connected circuit, a regular input signal and simulation system including two or three computing processors, and discuss based on the simulation model if the given simulation with the circuit and the input brings about rollbacks or not. Although the results include only special case studies, we believe that these will become a first step toward rollback analysis in parallel logic simulations.

key word: parallel logic simulation, rollback

1 はじめに

最近のハードウェア技術並びにOS技術の進展により，汎用の超高速計算機及び専用の並列プロセッサを用いた並列論理シミュレーションの実行が注目を集めている[1][2]．並列論理シミュレーションにおけるイベントの時刻管理方式として，集中時刻管理法[3]と分散時刻管理法がそれぞれ提案されてきている．処理の並列化の導入によって効率的な論理シミュレーションを行なおうとする場合，分散時刻管理法が有利であると考えられる．

分散時刻管理法では，各ゲート毎に局所的な時刻をもち，時刻印をもつイベントを受信すると各ゲートではそのイベントを評価し、ゲートの局所時刻を進める形で，並列に論理シミュレーションが行われる．分散時刻管理法による並列論理シミュレーションはその局所時刻の進め方（つまりゲートの評価方法）により保守的方法[6][10]と楽観的方法[7][8][9]の2つに分れる．

Chandy と Misra[4]により提案された保守的方法に基づく並列論理シミュレーションにおいては，ゲートのすべての入力端子にイベントが到着したときにのみ，そのゲートが評価され，ゲートの局所時刻が進められる．つまり，1本の入力端子にイベントが到着しても，他の入力端子にイベントが到着するまで待つ必要がある．このため回路中でいわゆるデッドロックが発生し，それを解消するためのNullメーセージが多数発生するために処理性能が低下する[1][7][8]．

一方，Jefferson[5]により提案された楽観的方法に基づく論理シミュレーションは，イベントが時刻順に到着するという楽観的な予測に基づいている．したがってゲートの入力端子の少なくとも一つにイベントが到着するとき，(必要ならば他の入力端子については，それ以前の値を使って）ゲートが評価され，局所時刻が進められる．後になって，もしゲート評価に使われた前の値が誤っていることが分れば，その時点でゲートの局所時刻を前の時刻に戻し，その時刻からゲートを評価し直す（これをロールバックという）．このため，楽観的方法においてはいかにしてロールバックの回数を減らすかが重要な問題となっている．

本稿ではこのロールバックの問題の解決への第一歩として，ロールバックを引き起こす要因について基礎的考察を行なう．これまでに，並列論理シミュレーションにおけるロールバックの実験的評価については松本と瀧による研究がよく知られている．文献[7][8]ではICOTで開発された新しいアーキテクチャをもつ並列計算機上で論理シミュレータを実現し，ベンチマーク回路を使用した多くの実験を行っている．

並列論理シミュレーションにおいてロールバックを引き起こす要因としては，与えられた回路を構成するゲートの種類，ゲートの接続パターン，回路の分割，及び，分割された回路のプロセッサへの割当て，など多くのものが考えられる．そうしたことがロールバックの解析を困難にしている．本稿では，考察の第一歩として，極めて単純な回路に対象を限定し，規則的な入力信号を利用し，プロセッサの台数も2台に制限した上で，これらの種々の組合せについてロールバック発生の有無を調べる．

2 並列論理シミュレーション

2.1 イベントの表現

信号線 n_i 上に図1(a)に示す信号波形が与えられたとする．ここではこの信号を次に示すイベントの系列で表す．
$[n_i,S,0][n_i,t_1,1][n_i,t_2,0]\cdots[n_i,t_{n-1},1][n_i,t_n,0][n_i,E,-]$
ここで，各イベント $[n_i,t_j,x]$ は時刻 t_j で信号線 n_i の信号値が x に変化することを表す．信号線 n_i が明らかな場合には $[n_i,t_j,x]$ の代わりに $[t_j,x]$ を使う．例えば，図1(b)の入力信号は，[S,0][5,1][10,0][15,1][40,0][50,1][60,0][E,-]で表す．すべてのイベントは図2に示すイベント管理表を用いて，時刻順に格納される．同一時刻のイベントはポインタで到着順に接続される．

2.2 ゲート評価

図3の例を使ってゲート評価の説明を行う．各ゲートは局所時刻（あるいは論理時刻）を保持しているものとする．今，図3のANDゲートの局所時刻を t_L，ゲート遅延を Δ とする．信号線 n_i 上の信号を $\mathrm{I}(n_i)$ で表す．ANDゲートの入力端子 n_1,n_2 にそれぞれ図3(b)の信号 $\mathrm{I}(n_1)$,$\mathrm{I}(n_2)$ が入力されたとき，出力として図3(b)の $\mathrm{I}(n_3)$ が得られる．ここで，$t_5=t_3+\Delta$,$t_6=t_2+\Delta$ である．この $\mathrm{I}(n_3)$ の計算の過程を図3(c)-(f)に示す．

まず，図3(c)ではイベント $[n_1,t_1,1]$ の評価を行う（$\mathrm{I}(n_2)$ の値は以前のものを用いる）．このときANDゲートの出力は0で，変化しないので，イベントの生成は行わない．ゲートの局所時刻 t_L を $t_L=t_1$ に更新する．次に図3(d)ではイベント $[n_2,t_3,1]$ の評価を図3(c)の $\mathrm{I}(n_1)$ を用いて行なう．このとき出力が1となり変化するため，イベント $[n_3,t_3+\Delta,1]$ を生成する．ゲートの局所時刻を $t_L=t_3$ に更新する．図3(e)ではイベント $[n_1,t_2,0]$ の評価を図3(d)での $\mathrm{I}(n_2)$ を用いて行なう．このとき $\mathrm{I}(n_3)$ の値が $t_5=t_3+\Delta$ で変化し，イベント $[n_3,t_2+\Delta,0]$ が生成される．ゲートの局所時刻は $t_L=t_2$ に更新される．最後に図3(f)では，イベント $[n_2,t_4,0]$ の評価を図3(e)の $\mathrm{I}(n_1)$ を用いて行なう．このとき出力は不変なので，イベントは生成しない．ゲートの局所時刻は $t_L=t_4$ に更新される．

以上の4個のイベント以外に途中に割り込むイベントがない場合，最初のイベント $[n_1,t_1,1]$ が時刻 t_p で処理されるなら，$[n_2,t_3,1]$,$[n_1,t_2,0]$,$[n_2,t_4,0]$ はそれぞれ時刻 t_p+1,t_p+2,t_p+3 で処理される．このような時刻を物理時刻 (physical time) と呼ぶ．

2.3 並列論理シミュレーション

並列論理シミュレーションは，通常，大規模なMIMD型のマルチプロセッサから成るシステム上で行われる．シミュレーションシステムは1台の制御用のプロセッサとn(n≥2)台の計算用のプロセッサから構成される．制御プロセッサは並列論理シミュレーションのために(1)負荷分散を目的とする(プロセッサの)スケジューリングと(2)同期を目的とする(メッセージの)スケジューリングの機能

をもつ．一方，各計算プロセッサはあらかじめ静的に割り当てられたゲートを評価する．

図4に並列論理シミュレーション・システムの概念図を示す．この図は，図5に示す回路分割に従ってゲートを2台のプロセッサ P_1, P_2 に割当てた例を描いている．入力端子には図6に示す2種類の入力信号を加えると仮定する．更に，1イベント当りの評価には1物理時間がかかり，各ゲート遅延を1論理時間と仮定する．また，計算プロセッサではイベントの評価とイベントの送受信処理が同時に行えると仮定する．

図7にシミュレーションの様子を示す．同図の横軸 t_p は物理時刻を表わす．また，P_i に関する横軸上で $t_p = k$ と $t_p = k+1$ の間にある記号 (G, n, t, y) は“プロセッサ P_i が物理時刻 $t_p = k$ に，イベント $[n, t, x]$ を評価し，その後イベント $[n', t+1, y]$ を生成し，ゲート G の局所時刻を t に更新すること”を表す．但し，イベント $[n', t+1, y]$ を生成するのは，ゲート G の出力端子 n' の値が変化した場合に限るものとする．このとき，n' が $P_j (j \neq i)$ に割り当てられたゲート G' の入力端子になっているなら，イベント $[n', t+1, y]$ は P_j に送信される．このイベントの転送に要する時間も1物理時間と仮定する．なお，イベントの転送の様子を図7では $t+1/y$ のラベルを付けた点線で示す．今，ゲート G' の局所時刻を t' とするとき関係式 $t+1 < t'$ が成立すると，ゲート G' でロールバックが発生するという．

2.4 ロールバック

ロールバック発生に至るまでの様子を図8に示す．ここではANDゲートの局所時刻を8，遅延を1論理時刻と仮定し，3つのイベント $[n_2, 10, 1], [n_1, 20, 0], [n_2, 15, 0]$ をこの順に評価するものとする．最初のイベント $[n_2, 10, 1]$ の評価時の入力信号 $I(n_1)$,$I(n_2)$ が図8(a)の通りであったとする．イベント $[n_2, 10, 1]$ の評価によってイベント $[n_3, 11, 1]$ が生成される．次に，イベント $[n_1, 20, 0]$ の評価によってイベント $[n_3, 21, 0]$ が生成される．この時点で図8(b)に示すように，t=11 から t=21 までの間は $I(n_2)$ と $I(n_3)$ の値が1となり，ゲートの局所時刻は20になっている．

引き続いて，イベント $[n_2, 15, 0]$ の評価をする．このときイベントの時刻印は15となっていて，ゲートの局所時刻より小さいのでロールバックが発生する．ロールバックに対する処置として，ゲートの局所時刻を15に戻して，イベント $[n_2, 15, 0]$ の評価を行う必要がある．この場合，t=15 から 20 までの間の $I(n_2)$ の値は図8(b)の評価で仮定していた値と異なっている．この $I(n_2)$ の値の変更に伴い，$I(n_3)$ も t=16 から 21 までの間で値を変更する必要がある．この $I(n_3)$ の値の再評価がロールバックの重要な処理となる．この例の場合には，イベント $[n_2, 15, 0]$ の再評価によってイベント $[n_3, 16, 0]$ が生成され，さらに引き続いて，イベント $[n_1, 20, 0]$ の再評価によって正しい波形が計算される．

3 シミュレーションモデル

3.1 イベントの評価とロールバック処理

次の Step1-Step8 に各計算プロセッサ P_i でのイベント評価とロールバック処理の作業手順を述べる．

Step1:(シミュレーション時刻の更新) … プロセッサ P_i のイベント管理表 E_i が空でなければ P_i の物理時刻を1単位時刻だけ進める．

Step2:(イベントの選択) … E_i から最小時刻のイベントで，リストの先頭に位置しているイベント $[n, t_s, x]$ を取り出す．

Step3:(ロールバックの判定) … ネット n を入力端子とするゲートを G とし，その局所時刻を t_L とする．もし $t_s < t_L$ ならば，ゲート G でロールバックが発生したと判断して，次は Step7 へ．

Step4:(イベントの評価) … イベント $[n, t_s, x]$ の評価を行う．その結果，ゲート G の出力の値が変化した場合に限り，新しいイベント $[n', t_s + \Delta, y]$ を生成する．ここでネット n' は G の出力端子であり，Δ はゲート G の遅延である．

Step5:(新しいイベントの処理) … ネット n' を入力端子とするゲートを G' とする．もし G' がプロセッサ P_i に割当てられていれば，新しいイベント $[n', t_s + \Delta, y]$ を E_i に登録する．一方，G' が他のプロセッサ P_j(j≠i) に割当てられていれば，イベントを P_j に転送する．

Step6:(イベントの受信) … Step1 から Step5 の実行中に，他の複数のプロセッサ P_k(k≠i) から新しいイベントを受信しておれば，これらのイベントをイベント管理表 E_i に登録する．しかし，同じイベントがすでに E_i にあれば新たな登録はしない．次は再び Step 1 へ行く．

step7:(ゲートの局所時刻の巻き戻し) …ゲートの局所時刻を t_s 以前の値に戻すと共に，評価済みのイベントの履歴も巻き戻す．更に，時刻印 t $(t_s + \Delta \leq t \leq t_L + \Delta)$ をもつイベントを取り消すためのアンチメッセージを発送する．

step8:(イベントの再評価) … 時刻印 t$(t_s \leq t \leq t_L)$ をもつすべてのイベントを再評価する．次は Step4 へ．

3.2 適用例

再び図5の回路と図6の入力について考える．3.1で説明したモデルに基づくシミュレーションの様子を図7と図9に示す．$0 \leq t_p \leq 17$ の部分が図7に，$17 \leq t_p \leq 37$ の部分が図9に描いてある．図9より物理時刻 $t_p = 18$ でプロセッサ P_1 からイベント $[n_9, 32, 1]$ がプロセッサ P_2 に転送され，それが物理時刻 t_p=19 で評価されている．このときゲート G_5 の局所時刻は既に 60 となっている．このためロールバックがゲート G_5 で発生し，図9上で網目で描い

ている2つのイベントの評価がロールバック処理として行なわれる.

4 基本的考察

ロールバックの発生の要因は多く考えられるが, ここでは最も基本的なものに限って説明する.

4.1 シミュレーション時間

今, n台の計算プロセッサを用いた並列論理シミュレーションを行うとする. このときプロセッサ P_j がシミュレーションに要する時間 T_j は次式で表すことができる.

$$T_j = \tau_{eval} \times N_1 + (\tau_{eval} \times N_2 + \tau_{oh}) \times P + \tau_{idle} \times Q$$

ここで
- τ_{eval}... イベント1個当たりの評価時間
- τ_{oh}... ロールバック1回当たりのオーバヘッド時間 (具体的には, Step7の作業に要する時間を指す).
- τ_{idle}... プロセッサの遊休時間の平均
- N_1... (ロールバックに伴なう再評価を除いて) P_j で処理するイベントの総数.
- N_2... ロールバック一回当たりに再評価するイベント数の平均
- P... ロールバックの発生回数
- Q... プロセッサの遊休の発生回数

である.
従って, 並列論理シミュレーションに要する時間Tは

$$T = max\{T_j\}$$

と表される.

並列論理シミュレーションを詳しく観察すると, 上述の3種類の時間 $\tau_{eval}, \tau_{oh}, \tau_{idle}$ 以外に

- τ_{com}... プロセッサ間でのイベントの送受信に要する時間の平均

があって, ロールバックの発生に大きくかかわっていることが分かる.

4.2 ロールバックの主な要因

ロールバックの発生の要因としては 多くのものが考えられ, しかも実際にはそれらが (単独ではなく) 相互に関連し合ってロールバックを引き起こしていると予想される. ここでは, ロールバック発生の主な要因として次の5つを指摘しておく.

(1) 与えられる回路の特性... 回路を構成するゲートの種類, ゲートの接続パターン

(2) 与えられる入力信号の特性... 入力信号の値の変化パターン (変化の回数, 時間間隔), 複数種類の入力信号がある場合、それらの関係

(3) 回路の分割 π... 与えられた回路をn個の部分回路に分割する方法 (例えば, 2つの部分回路間を結ぶネットの属性は分割 π に強く依存する).

(4) 入力信号の回路への割当て σ... 与えられた回路の各入力端子に対する入力信号の割当

(5) 処理速度 τ_{eval} と通信速度 τ_{com}... 計算プロセッサが1個のイベントを処理するのに要する時間 τ_{eval} と計算プロセッサ間でイベントを転送するのに要する時間 τ_{com} の値 (あるいは、その値の比)

これらの要因についての実験的検討を 5. で述べる.

5 実験的考察

実験的考察の第1歩として規則的な入力信号を簡単な回路に入力したときのロールバックの発生の有無について述べる. 以下ではゲートの評価時間 $\tau_{eval} = 1$(物理時刻), ロールバック処理のオーバヘッドの時間 $\tau_{oh} = 1$(物理時刻), 1イベント当たりの通信時間 $\tau_{com} = 1$(物理時刻) とし, ゲートの遅延時間を 1(論理時刻) とした. 各ネット上の信号値の初期値としては, x(unknown) を与えた.

5.1 回路の特性

図10に示す回路を考え, ゲート G_1-G_5 に AND, OR, NAND, NOR, EXOR を割り当てた種々の組合せ回路について検討する. 入力信号としては図11(a) に示す規則的な信号 I(1) を仮定する. 更に, 2台のプロセッサ P_1,P_2 を用いた並列論理シミュレーションを想定し, P_1,P_2 のそれぞれにゲートの集合 $\{G_1,G_2,G_4\}$, $\{G_3,G_5\}$ を割り当てるものとする.

[命題1]
図10中の各ゲート G_i を AND ゲートとした組合せ回路を C_1 と表す. 信号線 $n_1, n_2, n_3, n_4, n_5, n_6$ のそれぞれに入力信号 I(1) を与える. このとき組合せ回路 C_1 においてはゲート G_5 でロールバックが5回発生する. 次に, 図10中の各ゲート G_i を OR, NAND, NOR ゲートで置き換えて得られる組合せ回路をそれぞれ, C_2, C_3, C_4 と表す. このとき, 組合せ回路 C_2, C_3, C_4 ではいずれもゲート G_5 でロールバックが5回発生する.

[命題2]
図10中の各ゲート G_i を EXOR ゲートとした組合せ回路を C_5 と表す. 同一の入力信号 I(1) を各信号線 $n_i(1 \leq i \leq 6)$ に与える. このとき, 組合せ回路 C_5 では, ゲート G_5 でロールバックが1回だけ発生する.

[命題3]
図10中のゲート G_1, G_2 をそれぞれ NAND,OR ゲートで置き換え, ゲート G_3, G_4, G_5 をすべて AND ゲートで置き換えるとする. 入力信号については I(1) を各信号線 $n_i(1 \leq i \leq 6)$ に与える. このとき, ゲート G_5 でロールバックが3回発生する.

5.2 その他の特性

[命題 4]

信号線 $n_1, n_2, n_3, n_4, n_5, n_6$ のそれぞれに, 図 11(b) の信号 I(2) を与えるように変更した場合でも, [命題 1],[命題 2],[命題 3] は成り立つ.

[命題 5]

信号線 n_1, n_2, n_3, n_4, n_5 に図 11(a) の信号 $I(1)$ を与え, 信号線 n_6 に図 11(c) の信号 I(3) を与える. このとき組合せ回路 C_1, C_2, C_3, C_4, C_5 のいずれにおいてもロールバックは全く発生しない.

[命題 6]

信号線 n_1, n_2, n_3, n_4, n_5 に図 11(a) の信号 $I(2)$ を与え, 信号線 n_6 に図 11(c) の $I(4)$ を与える. このとき組合せ回路 C_1, C_2, C_3, C_5 ではいずれも 1 回のロールバックが発生するが, 組合せ回路 C_4 ではロールバックが発生しない.

[命題 7]

図 12 に示す組合せ回路 (C_7) を考える. 信号線 $n_1, n_2, n_3, n_4, n_5, n_6$ には図 11(a) の入力 $I(1)$ を共通に与えるものとする. 3 台のプロセッサ P_1, P_2, P_3 を用いた並列論理シミュレーションを行なうとし, 次の 3 種類の分割 π_1, π_2, π_3 について議論する.

$$\pi_1 = (\{G_1, G_2, G_9, G_{13}, G_{15}\}, \{G_3, G_4, G_5, G_6, G_{10}, G_{11}\}, \{G_7, G_8, G_{12}, G_{14}\})$$

$$\pi_2 = (\{G_1, G_2, G_9, G_{13}, G_{15}\}, \{G_3, G_4, G_7, G_8, G_{10}, G_{12}\}, \{G_5, G_6, G_{11}, G_{14}\})$$

$$\pi_3 = (\{G_1, G_2, G_7, G_8, G_9, G_{12}\}, \{G_3, G_4, G_{10}, G_{13}, G_{15}\}, \{G_5, G_6, G_{11}, G_{14}\})$$

表 1 に評価結果をまとめる. 同表では, ロールバックの発生回数の他に, 再評価されたイベント数, シミュレーション時刻なども示している.

6 むすび

本稿では, 並列論理シミュレーションにおいて楽観的方法で入力イベントを処理した場合に発生するロールバックについて基本的考察を行なった. その結果, ロールバック発生のメカニズムを解明するための幾つかの糸口を得ることができた.

今後の課題としては, 次の (1)〜(3) がある.

(1) 5. で行なってきた考察をさらに組織的なものへと発展させること, 及び, $\tau_{eval}, \tau_{oh}, \tau_{com}$ の値に関する制限を弱めてより一般的な解析を行うこと.

(2) 4. で述べた T_j の式に基づいてロールバックの発生メカニズムをより詳細に分析すること.

(3) ロールバックをその発生原因あるいは発生要因ごとに分類して, ロールバック発生のメカニズムの解明もその分類に応じて進めること.

謝辞 本研究を進める上で貴重な御意見をいただいた前 ICOT(現, 三洋電機 (株)) 松本幸則氏, 神戸大学 瀧和男助教授に深謝致します. また, 原稿作成に当たり御協力いただいた大阪大学基礎工学部 楠本真二博士, 奈良工業高等専門学校 西野繁之君に感謝致します.

参考文献

[1] 瀧　和男編：”第五世代コンピュータの並列処理”, bit 別冊, 共立出版,1993 年 7 月号.

[2] 広瀬　文保：”イベント法マシンの性能評価”, 信学論 (A), J75-A,11,pp.1691-1698(1992).

[3] Agrawal, P. : "Concurrency and communication on hardware simulators", IEEE Trams. Computer-Aided Design, Vol. CAD-5, No.4, pp. 617-623(1986).

[4] Chandy, K. M. and Misra, J. : "Distributed simulation: A case study in design and verification of distributed programs," IEEE Trans. on Software Eng., Vol. 5, No. 5, pp.440-452(1979).

[5] Jefferson, D. R. : "Virtual time", ACM Trans. on Programming Language and Systems, Vol.7, No.3, pp.404-425(1985).

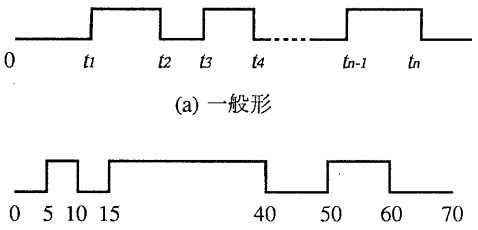
[6] 工藤, 木村, 天野, 寺沢：”問合わせに基づく並列論理シミュレーションアルゴリズム”, 信学論 (D-I), J75-D-I,4, pp.221-231(1992).

[7] 松本, 瀧：”バーチャルタイムによる並列論理シミュレーション”, 情処学論, Vol. 33, No.3, pp.387-395(1992).

[8] Matsumoto, Y. and Taki, K. : "Parallel logic simulation on a distributed memory machine", Proceedings of 1992 European Conference on Design Automation, pp.76-80(1992).

[9] Chung, M. J. and Chung, Y. : "Data parallel simulation using time-warp on the connection machine", Proceedings of 26th ACM/IEEE Design Automation Conference, pp.98-103(1989).

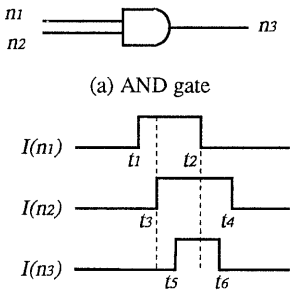
[10] Soule, L. and Gupta, A. : "Parallel distributed-time logic simulation", IEEE Design & Test of Computers, pp.32-48(1989).



(a) 一般形

(b) 例

図 1 入力信号



(a) AND gate

(b) $I(n_3)$ の計算

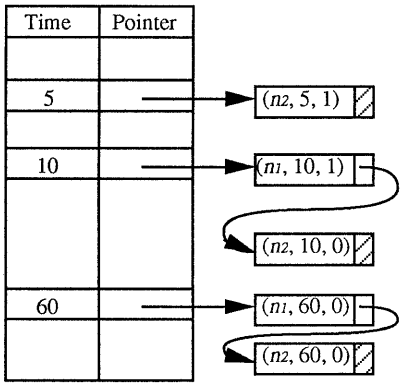
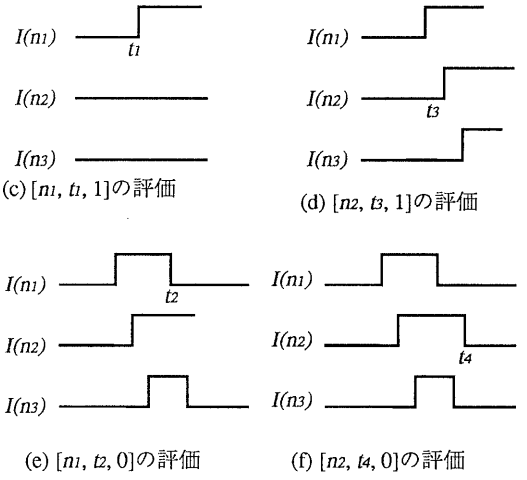


図 2 イベント管理表



(c) [n_1, t_1, 1]の評価

(d) [n_2, t_3, 1]の評価

(e) [n_1, t_2, 0]の評価

(f) [n_2, t_4, 0]の評価

図 3 イベントの評価

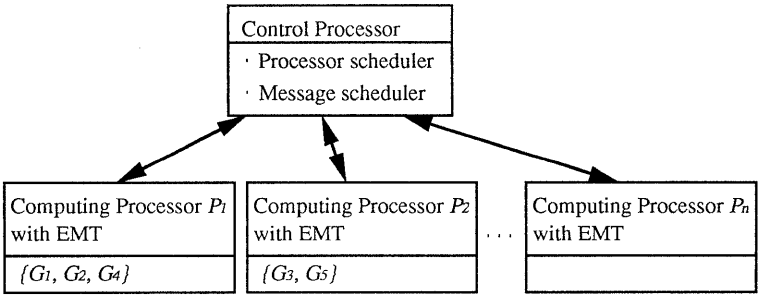


図 4 シミュレーションシステム

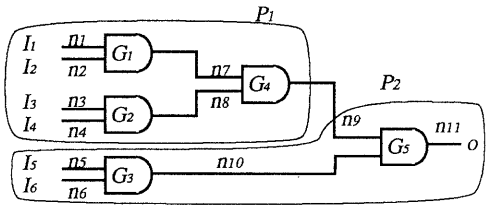


図5 回路例

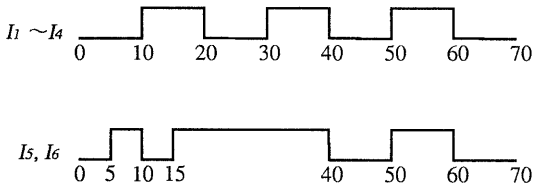


図 6 入力信号の例

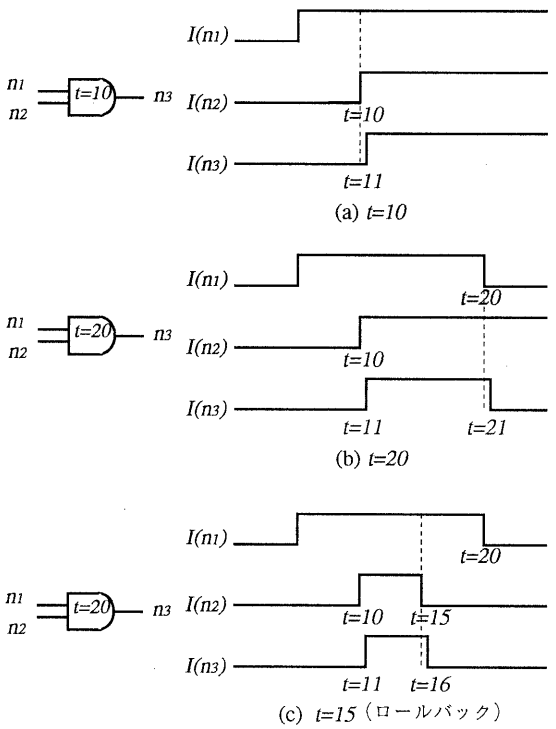


図 8 ロールバックの説明図

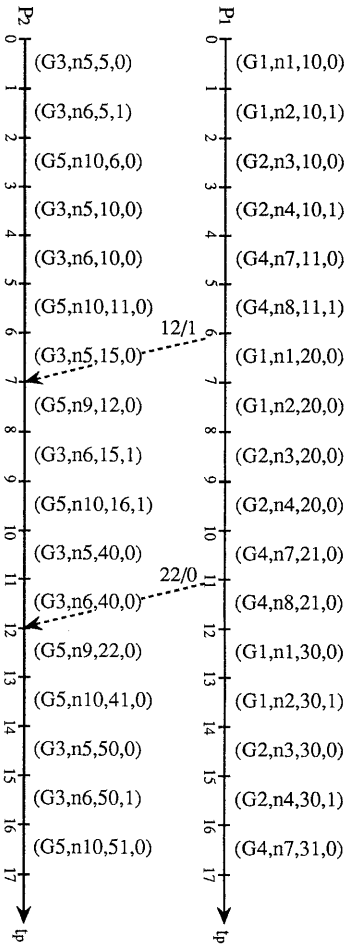


図 7 シミュレーションの進行($0 \leqq tp \leqq 17$)

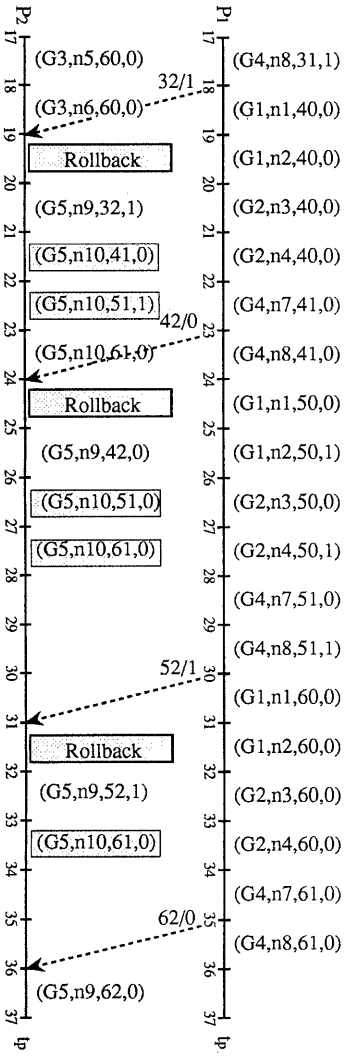


図 9　シミュレーションの進行($17 \leqq tp \leqq 37$)

表 1　回路分割の影響

	ロールバックの発生回数	評価したイベントの総数	再評価したイベントの総数	シミュレーション時刻
π_1	7	284	8	73
π_2	6	285	7	75
π_3	6	288	8	75

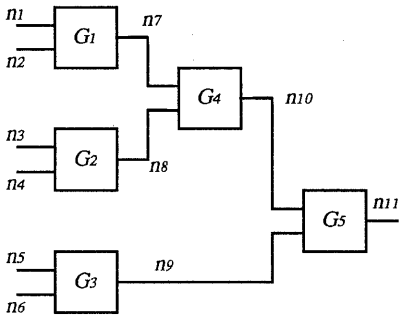


図 10　回路構成の説明図

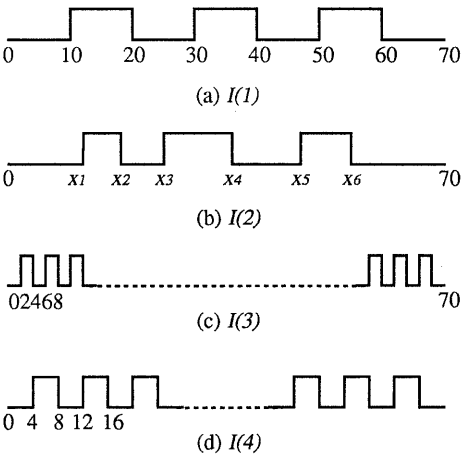


図 11　入力 *I(1), I(2), I(3), I(4)*

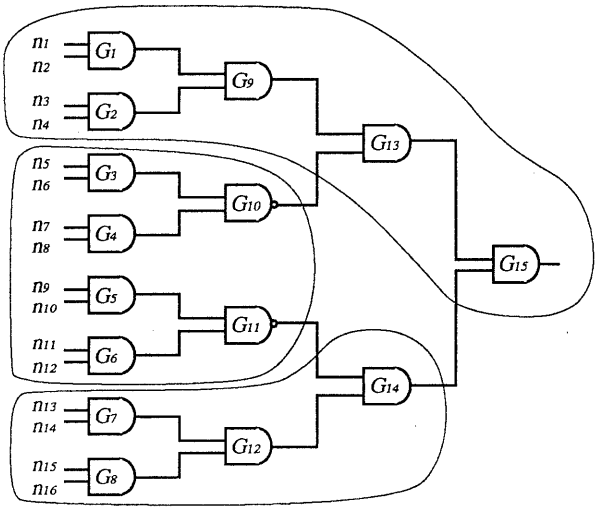


図 12　回路 C_7 と分割 π_1