

分散システムの記述とプロセス動作

4Q-5

河田 慶三, 平川 豊, 竹中 豊文
ATR通信システム研究所

1. はじめに

分散プロセスで構成される通信システムのサービス記述手法(プロセス間通信は記述せず、状態の書換え規則で記述)が提案されている[1]。本稿ではこのサービス記述からプロセス動作を構成する手法について考察する。

2. 分散システムの記述

2.1 分散システム

文献[1]で用いられている通信システムの状態記述例を図1(1)に示す。各プロセスの状態は、状態記述要素の集合で表現されている。例えば、プロセスp2の状態(呼出中(p2,p3), 通話保留中(p2,p1))は、プロセスp2がp1との通話を保留して、第三者p3を呼出中である状態を表現している。

本稿では、これを図1(2)のようなグラフでモデル化する。点v1から点v2へのラベルsの有向枝により、プロセスv1がプロセスv2との間にsなる関係があることを表現している。

このように分散システムの状態は、枝にラベルを持つ有向グラフ $g=(V,A)$ で表現することができる。またその動作は、グラフ上の点が入力(イベント)を受けたときの新たなグラフへの変換と捉えることができる。

イベントはプロセス引数を持たないと仮定する。

- V: 点の集合(プロセスを表す)
- A: 枝の集合(プロセス間の関係を表す)
- S: 属性の集合(関係を識別するラベル)
- E: イベントの集合

2.2 分散システムの記述

分散システムのふるまいを複数個のグラフ書き換え規則で表現する。まず、書き換え規則を定式化する。

[規則の定義]

規則rは次の3項の組 $r=(e, g1, g2)$ である。ここでeはイベントであり $g1=(V1, A1)$ と $g2=(V2, A2)$ はイベントが生じた点を入力点として指定したグラフである。g1は書き換え前のグラフ(現状態)であり、g2は書き換え後のグラフ(次状態)である。グラフの枝だけ書き換えるため $V1=V2$ である。またg1では入力点からV1のすべての点へのパスが存在する。図2に規則の例を示す。

次に書き換え規則適用法をグラフを用いて定式化する。

[規則適用の優先順位]

同一イベントeに対する任意の2つの規則 $r1=(e, g11, g12)$, $r2=(e, g21, g22)$ に対してg11がg21を部分グラフ

として含むならばr1がr2よりも適用の優先度が高い。
[規則の適用]

分散システムの状態を表現するグラフをgとする。g上の点vでイベントeが生じた場合、eに対するすべての規則 $r=(e, g1, g2)$ に対してgのvを含む部分グラフg1と同形なものをすべて選び出す。次に優先度を考慮して実行する規則を1個選択する。その規則を $r1=(e, g11, g12)$ とする。選択した部分グラフをg12に変更する。

3. プロセス動作の構成

3.1 概要

ここでは、グラフ書き換え規則が与えられたとき、この規則に従ったプロセス動作の構成手法を示す。構成されるプロセス動作は、図3に示すように、グラフ書き換え規則に関するデータを持ち、周囲のプロセスとの通信を行いながら、前記の[規則の適用]手順をシステム全体として実現する。

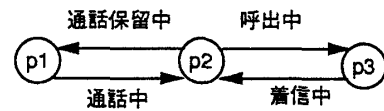
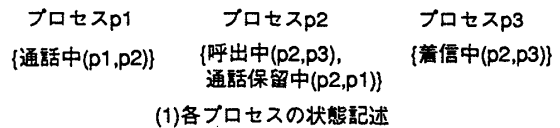


図1 分散システムの状態記述

$r1 = (e1, g1, g2)$

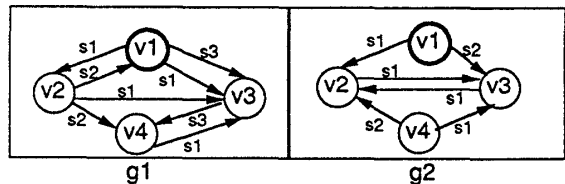


図2 規則の例

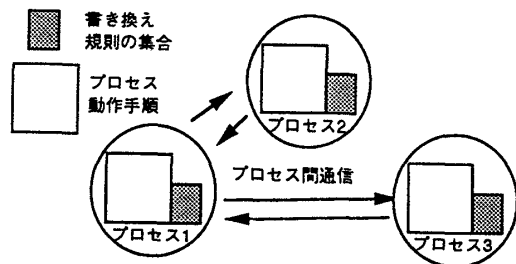


図3 分散システムの動作

以下では、上記のg1またはその部分グラフと同形なグラフが見つかったとき、これをg1の各点から各プロセスへの対応関係（以後、対応と呼ぶ）で表現している。一般的には、複数の対応が見つかる場合がある。

次に、必要な用語を定義する。

[近傍] グラフg上の点vに対し、vとvに隣接した点および枝で構成されるgの部分グラフをvの近傍と呼ぶ。
 [D(v_i)] 極大木t上の点v_iの子孫（図4(5)参照）と各子孫にg1上で隣接する点からなる集合をD(v_i)と書く。v_iの子孫にはv_i自身も含まれる。

[プロセスの保持するデータ]

(1) プロセスの近傍（図4(3)参照）

プロセスpはpの近傍に関する情報（プロセスの状態に相当）を保持する。

(2) 規則と規則中のグラフの極大木

規則r=(e, g1, g2)に対し、g1の入力点を根とする極大木tを選ぶ。プロセスは規則の集合と各規則に対する極大木を保持する。極大木の枝が通信の経路に相当する。

[プロセス間で通信されるデータ]

(1) 判定要求メッセージ m-match(p, r, v).

(2) 応答メッセージ

D(v_i)中の各点とプロセスの対応。

すなわちD(v_i)からPへの単写の集合M_iである。

(3) グラフ変更指示（本稿では記載していない）

3. 2 プロセス動作

(1) イベントeが生起したプロセスpの動作

イベントeに関する規則の集合Rを選ぶ。規則r_iに対しバッファRes(r_i)を用意する。すべてのRes(r_i)を空にする。

LOOP: Rが空の場合goto END.

Rから要素r_iを1個取り出す。r_iの入力点をvとする。

match(p, p, r_i, v)を行う。goto LOOP.

END:

空でないRes(r_i), Res(r_j)でr_iがr_jより優先度が高い場合はRes(r_j)を空にする。すべてのRes(r_k)が空の場合は停止する。

空でないRes(r_k)から要素mを1個取り出す。

これにより実行すべき規則r_kと、r_k上の点とプロセスの対応mが得られた。

(これ以降の処理概要は省略する。)

(2) m-match(q, r, v)を受信したプロセスpの動作

Res(r)を空にする。

match(q, p, r, v)を行なう。qにRes(r)を送信する。

(3) matchの定義

match(q, p, r, v)

1. [vの近傍各点からpの近傍各点への対応Fを求める]

v, pの近傍の点集合をそれぞれN(v), N(p)とする。

N(v)の要素v_iからN(p)へのつぎのような対応（単写fの集合F）を求める。

f(v_i)=p_{ij}と書く。任意のラベルs∈Sに対して、vからv_iへラベルsの枝が存在するときpからp_{ij}への枝が存在する。またf(v)=pである。

Fが空の場合goto END.

極大木tのvの子（図4(4)参照）の集合をT(v)とする。（ここでT(v)⊂N(v)である。）

T(v)が空集合の場合Res(r)にFを代入しgoto END.

2. [近傍で求めた対応Fと子から得られたすべての対応M_iから、極大木t上の点vの子孫及びそれらの近傍の各点からプロセスPへの対応を構成する]

LOOP:Fが空ならばgoto END.

集合Fから写像f_kを1個取り出す。

(1) T(v)のすべての要素v_iに対し、プロセスf_k(v_i)にメッセージm-match(p, r, v_i)を送信する。

(2) f_k(v_i)からメッセージM_iを受信する。

M_iはD(v_i)中の各点とプロセスの対応である。

すなわちD(v_i)からPへの単写m_iの集合である。

(3) M_iが空の場合、goto LOOP.

Fと受信した各M_iから、D(v)中の各点とプロセスの対応H（D(v)からPへの単写hの集合）を作成する。

D(v)はN(v)と各D(v_i)の和集合である。

H={h | ∇v_i∈T(v), ∃m_i∈M_i, h(D(v_i))=m_i(D(v_i)), h(N(v))=f_k(N(v))}

Res(r)=Res(r)∪H. goto LOOP.

END:

□

4. おわりに

本稿では、プロセス間通信の規定されていない分散システムの動作記述から、各プロセスの動作を構成する手法について述べた。提案した手法では、プロセス間通信が膨大であるという問題がある。今後は実用的なプロセス間通信を生成するために、実際的な制約を導入し効率化をはかる。

参考文献

[1]Y.HIRAKAWA, Y.HARADA, Y.TAKENAKA 'A Description Method for Advanced Telecommunication Services', 電子情報通信学会, 日韓合同交換システム研究会, SSE89-87, 1989

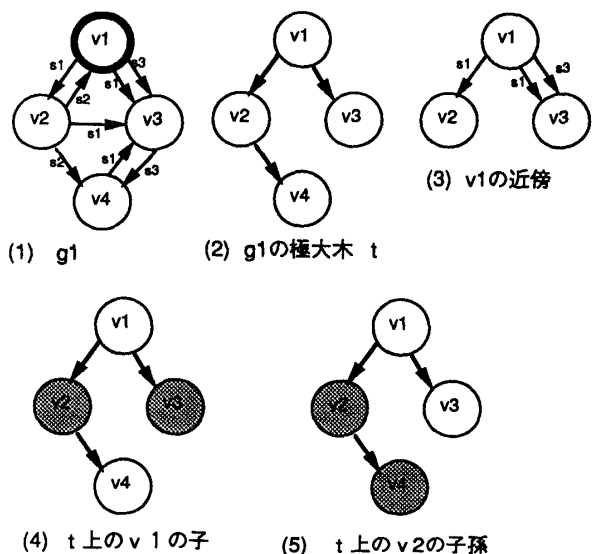


図4 参考図