数理解析研究所講究録
第 470 巻 1982 年 104-113

「同期基本命令の公理的定義」の紹介

瀬川　清（早大）

Martin, Alain J., "An Axiomatic Definition of Synchronization Primitive", Acta Informatica 16, P. 219-235 (1982) の概要を述べる。

1. はじめに

セマフォ演算 P/V のような，組として用いられる二つの同期基本命令の意味を "boundness", "progress" と "fairness" と呼ばれる三つの公理による定める。また，slack と呼ばれる「自由度」により，同期基本命令は次の三つに分類できる：

slack
- 無限 —— 無限のバッファを持つもの（例 P/V）
- 有限 —— 有限のバッファを持つもの（例 send/receive）
- 0 —— バッファを持たないもの（例 CSP）

2. 二つの動作の同期 (The Synchronization of two Actions)

定義

Xaction：命令 X は実行されると，Xaction をおこす。

sequential computation：actionの有限／無限列。

concurrent computation：幾つかのsequential computationから成る。

process：concurrent computationを構成するsequential computationを呼ぶ。

cX：computationが始まってからの、X actionの実行個数（正しくは、終了個数）。

gX：現在、中断されているX actionの個数。

tX：cX＋gXと定義される。

同期：命令の組（X，Y）に対して、cXとcYの間に自明でない関係があるとき、X actionとY actionは同期しているという。

同期基本命令：X actionとY actionが同期していて、cX－cYが上限または下限を持つとき、XとYは同期基本命令であるという（このことを、厳密に後で定義する）。

3．同期基本命令の意味合い（The Semantic Characterization of Synchronization Primitives）

同期基本命令の持つべき性質として，次のようなものが考えられる。そして，これらを同期基本命令の公理とすればよいと思われる。

XとYが同期しているのならば，XとYの実行回数に大きな差はないと考えられるので，

boundness

R1：cX－cYは上限または下限を持つ

という性質があげられる。R1を満たさなくするような，命令XまたはYの実行は中断（suspend）される。次に，中断されっぱなしでは困るので，

progress

R2：中断中の命令の集合はできる限り小さくする

という性質が欲しくなる。これから，系として次のことがあげられる：

命令の組（X，Y）に対し，Xが中断されるならYは中断されないし，Yが中断されるならXは中断されない。

中断中のactionを含むprocessは，そのactionの所で，delayしているという。Xで中断しているprocessがあるのに，Yが実行されることがなくなると，deadlockが生じる。

一方，Yが実行されても，中断中のままであるときは，starvationが生じる。starvationは，もちろん困るので，

fairness

R3：processがdelayしているとき，中断している命令を含む，命令の組の実行回数は有限である

という性質があげられる。

R1とR2を満足するようなものを，弱い同期基本命令（weak synchronization primitive）と呼び，さらにR3も満足するものを，強い同期基本命令（strong synchronization primitive）と呼ぶ。

4．代数的形式化（An Algebraic Formulation of the Synchronization Requirements）

3．で述べた三つの性質（R1～R3）を形式化する。

Boundness Requirement

R1：kX，kYを整定数として

①kX，kYの少なくとも一つは有限

② $-kX \leqq cX - cY \leqq kY$

系．$kX, kY \geqq 0$

Progress Requirement

$$
\begin{aligned}
R2:\ & (gX=0 \vee cX = cY + hX) \wedge \\
& (gY=0 \vee cY = cX + hY) \wedge \\
& (gX=0 \vee gY=0)
\end{aligned}
$$

R2 の注釈

$gX=0 \vee cX=cY+hX$，すなわち，$gX\neq 0$，つまり，Xで中断するならば，$cX=cY+hX$。これは，Xで中断するのは，R1を満足しなくなるときだけである，ということを現わす。

$gX=0 \vee gY=0$，すなわち，XとY両方が中断することはないということを現わす。

Fairness Requirement

procesiに対して次のようなdiを考える

$$
d_i = \begin{cases} 0 & \text{process } i \text{ は delay 中でない。} \\ m & \text{process } i \text{ が X action で delay しているとき，その delay 中に（他の process により）実行（終了）された X の数。} \end{cases}
$$

R2：各process iに対して，d_iは有限

次の「弱い同期基本命令に関する定理」が容易に導かれる：

$$(X, Y) \text{ が弱い同期基本命令}$$

$$\Updownarrow$$

$$cX = \min(tX, tY + kX) \wedge$$

$$cY = \min(tY, tX + kY)$$

5. 同期基本命令の三つの型 (The Three Types of Synchronization Primitives)

$K = kX + kY \ (\geqq 0)$ を slack と呼び，その値により，同期基本命令を類別する。

$K = 0$ を R1, R2 に代入したものを考える。逆に，それらにより，同期基本命令の一つを定義する。

定義

命令 X と Y は次の二つの条件を満たすとき，バッファ無し通信命令 (unbuffered communication primitive) という：

$$R1: \ cX = cY$$

$$R2: \ gX = 0 \vee gY$$

ここでは R3 (fairness) を考えていない。すなわち，弱い同期基本命令の類別を行なっている。

$K = \infty$の場合は、$kX < \infty$、$kY = \infty$とすると、

$$R1:\ cX \leqq cY + kX$$

$$R2:\ (gX = 0 \vee cX = cY + kX) \wedge gY = 0$$

となり、ここで $s = kX - cX + cY$ とおくと

$$R1:\ s \geqq 0$$

$$R2:\ (gX = 0 \vee s = 0) \wedge gY = 0$$

となる。$kY = \infty$より、$gY = 0$となるので、次の定義が得られることになる。

定義

次の条件を満たすPとVをセマフォ命令と呼ぶ：

s：初期値s_0の整変数（セマフォと呼ぶ）

$$A_0:\ s \geqq 0$$

$$A_1:\ cP + s = cV + s_0$$

$$A_2:\ gP = 0 \vee s = 0$$

$K < \infty$の場合を考えると次の定義が導かれる。

定義

次の条件を満たすPとVを、対称P/V命令と呼ぶ；

s：初期値 s_0 のセマフォ

s_m：正整定数

B_0：$0 \leqq s \leqq s_m$

B_1：$cP + s = cV + s_0$

B_2：$qP = 0 \lor s = 0$

B_3：$qV = 0 \lor s = s_m$

6．古典的定理（Some Classical Theorem）

ここで示されているいくつかの定理の内容は、我々が常識だと思っていることである。そのようなものが、先に定義したものから導かれるということは、「定義」自身が妥当なものであるということを現わすことになろう。

相互排除に関する定理

Theorem. *Consider an arbitrary number of concurrent processes, each consisting for what concerns synchronization in a strict alternation of P and V operations, starting with a P and ending with a V, on any of the n ($n>0$) semaphores s_i: $0 \leqq i < n$. If np is the number of processes having completed a P and not yet completed the following V, then $np \leqq \sum_i s0_i$, where $s0_i$ is the initial value of s_i.*

Corollary. *If $\sum_i s0_i = 1$, there is at most one process at a time inside the program part enclosed by a P and the following V.*

生産者／消費者問題に関する定理

Theorem. *Given an arbitrary number of concurrent processes of the "producer" type defined by the program text:*

$$*[\ldots P(s);\ PUT;\ V(r);\ \ldots]$$

and an arbitrary number of concurrent processes of the "consumer" type defined by the program text:

$$*[\ldots P(r);\ GET;\ V(s);\ \ldots],$$

where PUT and GET are arbitrary atomic actions, then:

$$-r_0 \leqq cPUT - cGET \leqq s_0$$

holds, where s_0 and r_0 are the initial values of s and r, respectively.

deadlock に関する定理

Theorem. *Given n processes defined by the program text:*

$$P(a);\ P(b);\ V(a);\ V(b)$$

and m processes defined by the program text:

$$P(b);\ P(a);\ V(b);\ V(a)$$

with $m+n>0$, the computation is deadlock-free if the following relation holds on the initial values a_0 and b_0 of a and b:

$$a_0>0 \wedge b_0>0 \wedge (a_0>n \vee b_0>m).$$

8、対称 P／V 命令と生産者／消費者問題型命令

生産者／消費者問題で次のように置きかえを行なう：

$$t：cPUT - cGET + r_0$$

$$P'(t)：\ P(r);\ GET;\ V(s)$$

$$V'(t)：\ P(s);\ PUT;\ V(r)$$

Theorem. *If $P'(t)$ and $V'(t)$ are considered as indivisible actions, the following relations are invariantly true:*

$$\begin{aligned}
&\mathrm{C0}:\ 0 \leqq t \leqq s_0 + r_0\\
&\mathrm{C1}:\ cP' + t = cV' + r_0\\
&\mathrm{C2}:\ qP' = 0 \vee t = 0\\
&\mathrm{C3}:\ qV' = 0 \vee t = s_0 + r_0.
\end{aligned}$$

すなわち、生産者／消費者問題のための同期基本命令を考えると、対称P／V命令となる。

9．P／V命令の実現（An Implementation of P and V operations）

二進セマフォ系により、一般のセマフォ系を次のように実現すると、先のA_0～A_2を満たすことがわかる。

```
P(s): P(m);
    [s>0→s, p:=s−1, p+1
    ▯s=0→q:=q+1; V(m); P(z)
    ];
    V(m).
V(s): P(m);
    [q=0→s, r:=s+1, r+1
    ▯q>0→q, p, r:=q−1, p+1, r+1; V(z); P(m)
    ];
    V(m).
```

Initially: $s=s0$, $m, z = 1, 0$ $p, q, r = 0, 0, 0$.

10．結論

セマフォ命令の公理として、先のA_0～A_2が必要十分であるというのは、仮説であるが、妥当なものであると思われる。