状況制約下における並行モジュールのシーケンス構成論

岩田健一[†] 笹倉万里子[†] 山崎進[†]

本論文では状況制約下における並行モジュールのシーケンスを求めるための数理モデルとしてモジュール計算系を提案する．この数理モデルでは，状況とモジュールという 2 つの構成要素を用い，状況と状況の間の制約，モジュールとモジュールの間の規則，状況とモジュールの間の関係を記述することができる．モジュール計算系は複数あって，並行動作が可能である．本論文ではこのモデルにおいてある状況からある状況への遷移と，そのときのモジュールのシーケンス（列）を得る手続きを示す．このモデルでは現実社会の事象を容易に記述することができ，本論文ではその例として，製造業における部品の収集組み立て問題を論ずる．

A Concurrent Construction of Module Sequences under Situation Conditions

Kenichi Iwata,[†] Mariko Sasakura[†] and Susumu Yamasaki[†]

In this paper, we propose a concurrent calculus model that gives a sequence of modules. The model uses *situations* and *modules*, containing the constraint between situations, the rule regarding modules, and the relation between situations and modules. A module calculus works concurrently with other calculi. We show a procedure that gives a transition from a situation to another with a sequence of modules. The model has ability to describe complex phenomena in the real world. We describe a problem of parts collection and composition in product making as its application.

1. はじめに

本論文では，状況による制約を与えた場合に，その制約のもとで解を与える並行動作可能なモジュールのシーケンス（列）を求めることができる数理モデルを提案する．

この数理モデルでは，状況の集合と，ある状況において実行可能なモジュールの集合を定義する．また，状況と状況の間の制約，状況とモジュールの間の関係，モジュールとモジュールの間の規則を与える．このとき，状況の初期値と終了値，実行すべきモジュールを与えると，以下の条件を満たす状況の列およびモジュール列を得ることができる．

- 得られる状況列は与えられた初期値と終了値を持つ．
- 得られるモジュール列は与えられた実行すべきモジュールを含む．
- 与えられた状況・モジュール間の関係や制約に違反しない．

† 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

このときに，モジュール列は 1 つだけ得ることもできるが，複数の並行動作可能なものを得ることも可能である．

この数理モデルを製造業における部品の収集組み立て問題に適用する．部品の収集組み立て問題とは，1 つの工場内で多品種少量生産を行う際に，注文ごとに微妙に異なる組み立て仕様への対応を行う必要から生じるものである．本論文で扱うのは，

- ある注文に対しどの部品をどんな順番で組み立てるかをどのように管理するか，
- 実際に工場のある区画に置いてある部品をどのような順番で取り出してくるか，

という問題である．この問題では，通常複数の人間が部品収集組み立てに携わり，同時に複数の部品を組み立てる．すなわち，本質的に並行性を有している問題である．現在はこの問題の解決はほとんど人間の判断に依存し，自動化に限界がある．

提案する数理モデルは，この並行性をモジュールと状況を用いて自然に記述できることが特徴である．部品をモジュール，工場における区画を状況にモデル化し，部品の組み立て方をモジュール間の規則，部品が工場のどの区画に置いてあるかをモジュールと状況の

各 $1 \leq i, j \leq n$ に対して次の規則を用いる．

(0) モジュール計算系 $\Im_i$ に対して次の宣言をする．

(i) $(A : \bot) \in Process_i$ のとき，

$$(A : \bot) \in Process_i \leftarrow$$

(ii) $(A : \Gamma) \in Process_i$ のとき，

$$(A : \Gamma) \in Process_i \leftarrow$$

(iii) $(\sigma, \sigma') \in I$ のとき，

$$(\sigma, \sigma') \in I \leftarrow$$

(iv) $\sigma \in Situation_i(A)$ のとき，

$$\sigma \in Situation_i(A) \leftarrow$$

(1) $gather_i(A; \sigma_1; \sigma_3) \leftarrow (A : \bot) \in Process_i,\ \sigma_2 \in Situation_i(A),\ (\sigma_1, \sigma_2) \in I,\ (\sigma_2, \sigma_3) \in I$

(2) $gather_i(A; \sigma_1; \sigma_3) \leftarrow (A : \Gamma) \in Process_i, gather_i(\Gamma; \sigma_1; \sigma_2),\ \sigma_2 \in Situation_i(A),\ (\sigma_2, \sigma_3) \in I$

(3) $gather_i(A\Gamma; \sigma_1; \sigma_3) \leftarrow gather_j(A; \sigma_1; \sigma_2), gather_i(\Gamma; \sigma_2; \sigma_3)$

(4) $gather_i(A\Gamma; \sigma_1; \sigma_3) \leftarrow gather_i(A; \sigma_1; \sigma_2), gather_j(\Gamma; \sigma_2; \sigma_3)$

図 1 述語 *gather* に関する規則
Fig. 1 The rules of the predicate *gather*.

関係，工場の区画内のどこからどこへ移動できるかを状況間の制約として与えれば，ある製品を組み立てたいときに工場の中をどのように移動してどのような順番で部品を組み立てていけばよいかが，状況およびモジュールのシーケンス（列）として得られる．与える知識は並行動作を意識しないで記述できるのに，並行動作可能なものを得ることができるのが特徴である．この問題は生産計画が単純な場合には人間の判断によって何ら問題なく遂行されうる種類の問題であるが，問題の複雑度が増すと，自動化が難しい課題であるといえる．本研究で提案する数理モデルを用いることにより今まで人間に頼っていた判断の少なくとも一部を自動化することができるため，複雑な作業が要求される多品種少量生産の現場における生産活動の助けとなるようなシステムを構築することができる．

本論文では，まず数理モデルを定義し，その健全性を示す．次に，例題として工場における並行部品収集組み立て問題への応用を示す．

2. 数理モデル

2.1 モジュール計算系

数理モデルの定義を次に示す．モジュール計算系は

$$\Im = (P, \Sigma, Process, Situatoin, I)$$

である．ここで，

(i) P はモジュールの集合，

(ii) Σ は状況の集合，

(iii) $Process$ はモジュールの集合 P 上のルールの集合，

(iv) $Situation : P \to 2^{\Sigma}$ は関数，

(v) $I \subseteq \Sigma \times \Sigma$ は制約関係．I が反射的かつ推移的であると仮定する．

モジュール列の全体を P^* で表す．空モジュール列は ϵ で表す．$A\epsilon = \epsilon A = A$ である．

$Process$ は空集合ではなく，その要素であるルールは以下の形で記述される．

$$A : B_1 \ldots B_n \quad (n \geq 1), \text{ または}$$
$$A : \bot$$

ここで $A, B_1, \ldots, B_n \in P$ である．$A : B_1 \ldots B_n$ はモジュール A が $B_1, \ldots, B_n$ から構成されることを示している．$A : \bot$ は A が基本要素であることを示す．

$Situation$ はあるモジュールに対する状況の集合を規定する．

I は状況の対の集合で，直観的には $(\sigma_i, \sigma_j) \in I$ は，状況 σ_i から σ_j に状況遷移できることを示す．

並行モジュール計算系は，モジュール計算系の組として定義される．

$$\Pi = (\Im_1, \ldots, \Im_n)$$

ここで $\Im_i = (P, \Sigma, Process_i, Situation_i, I)$（$1 \leq i \leq n$）はモジュール計算系である．

モジュール列 Γ，状況 σ_1，σ_2 に対する述語 $gather_i(\Gamma; \sigma_1; \sigma_2)$（$1 \leq i \leq n$）は**図 1** を満足する．ここで，$\sigma_1$，$\sigma_2 \in \Sigma$ であり，$\Gamma \in P^* - \{\epsilon\}$ である．

図 1 は論理プログラムの形式で書かれている．$\leftarrow$ の右辺の述語すべてが成立すれば，$\leftarrow$ の左辺の述語が成立する．図 1 の (0) はモジュール計算系が与えられたときに成り立つ述語を記述している．(1) はモジュール A が基本要素であるときの規則である．(2) はモジュール A を構成するモジュールに分解できる

という規則である．(3)，(4) はモジュール列のうちの 1 つのモジュールについて自己計算系もしくは他の計算系に問合せを行う規則である．このモデルでは，モジュール列のうち少なくとも 1 つのモジュールは自己計算系で処理しなければならないとしているので，モジュール列の先頭を自己計算系で処理する規則 (4) とそれ以外の部分を自己計算系で処理する規則 (3) の 2 つを記述している．

定義 1 のように関係 SEM を定義する．関係 $SEM(\sigma_1;\beta;\sigma_2)$ は，σ_1 から σ_2 への状況遷移をひき起こすモジュール列 β を定義したものである．

定義 1 意味関係 $SEM \subseteq \Sigma \times P^* \times \Sigma$ を次のように帰納的に定義する．

(1) $SEM(\sigma_1;\epsilon;\sigma_2) \Leftrightarrow (\sigma_1,\sigma_2) \in I$

(2) $SEM(\sigma_1;\gamma x;\sigma_3) \Leftrightarrow \sigma_2 \in Situation(x)$，
$(\sigma_2,\sigma_3) \in I$，$SEM(\sigma_1;\gamma;\sigma_2)$ $(\gamma \in P^*,\ x \in P)$

以下の補題は，モジュール列の連接に関する，関係 SEM の性質を示す．

補題 1 $SEM(\sigma_1;\beta;\sigma_2)$，$(\sigma_2,\sigma_3) \in I$ で，
$SEM(\sigma_3;\gamma;\sigma_4)$
であると仮定する．このとき，$SEM(\sigma_1;\beta\gamma;\sigma_4)$ である．

証明は付録 A.1 に示す．

以下の定理は，並行計算系のある計算系 $Process_i$ に関わる述語 $gather_i(\Gamma;\sigma_1;\sigma_2)$ が導き出されるときには，あるモジュール列 β があって，$SEM(\sigma_1;\beta;\sigma_2)$ が成立することを示している．この意味で（関係 SEM に関して），述語 $gather_i(-;-;-)$ の健全性が示される．

定理 1 $gather_i(A;\sigma_1;\sigma_3)$ $(A \in P) \Rightarrow$
$\exists\beta \in P^*.\ SEM(\sigma_1;\beta;\sigma_3)$

証明は付録 A.2 に示す．

これらの性質をふまえた実装システムの基本構造は**図 2** のようになる．すなわち，システムの入力は並行モジュール計算系 Π と実際の問合せ述語 $gather_i(\Gamma;\sigma_1;\sigma_2)$ で，出力は $SEM(\sigma_1;\beta;\sigma_2)$ を満足する β となる．システムの内部では論理プログラムゴール形式 $\leftarrow gather_i(A;\sigma_1;\sigma_2)$ の成功か否かを計算する．

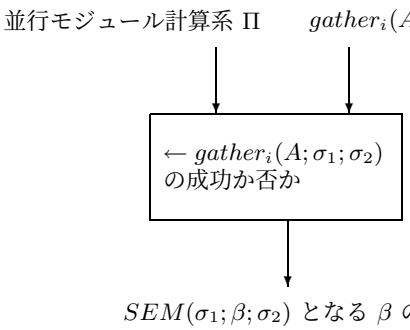

図 2 実装システムの基本構造
Fig. 2 Fundamental structure for implementation.

2.2 例　　題

$\Pi = (\Im_1, \Im_2)$ において，
$P = \{A, B_1, B_2, B_3\}$
$\Sigma = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}$
$Process_1 = \{$
$A : B_1B_2B_3$
$B_1 : \bot\ \}$
$Process_2 = \{$
$B_2 : \bot$
$B_3 : \bot\ \}$
$Situation_1(A) = \{\sigma_4\}$
$Situation_1(B_1) = \{\sigma_1\}$
$Situation_1(B_2) = \{\sigma_2\}$
$Situation_1(B_3) = \{\sigma_3\}$
制約関係：
$(\sigma_1,\sigma_2) \in I$, $(\sigma_2,\sigma_3) \in I$
$(\sigma_3,\sigma_4) \in I$, $(\sigma_2,\sigma_1) \in I$
$(\sigma_3,\sigma_2) \in I$, $(\sigma_4,\sigma_3) \in I$

以上のような条件のとき，
$gather_1(A;\sigma_1;\sigma_4)$
が与えられたとする．このときの実行を**図 3** に示す．図 3 中の □ は与えられたモジュール計算系において成り立つことを示す．$gather_1(A;\sigma_1;\sigma_4)$ は $A : B_1B_2B_3 \in Process_1$，$I$ が反射的なので $(\sigma_4,\sigma_4) \in I$ であることから図 1 の規則 (2) より $gather_1(B_1B_2B_3;\sigma_1;\sigma_4)$ に変換される．以下図 3 のように実行され，$gather_1(A;\sigma_1;\sigma_4)$ が成り立つことが示せる．規則 (3) と (4) の選び方，また，規則 (3) および (4) における j の選び方は非決定的である．

3. 応　　用

この章では，部品の収集組み立て問題に関する応用について述べる．

3.1 部品の収集組み立て問題

経営や事業活動の効率化の観点から，製造業ではトヨタカンバン方式[6),17)] に代表される少品種大量生産が追求されてきている．カンバン方式においては，資本効率の向上の観点から，余剰在庫の圧縮が主要な課題であった．

現在，BTO（Build To Order：受注生産）の普及が進み，注文ごとに異なる組み立てスペックへの対応

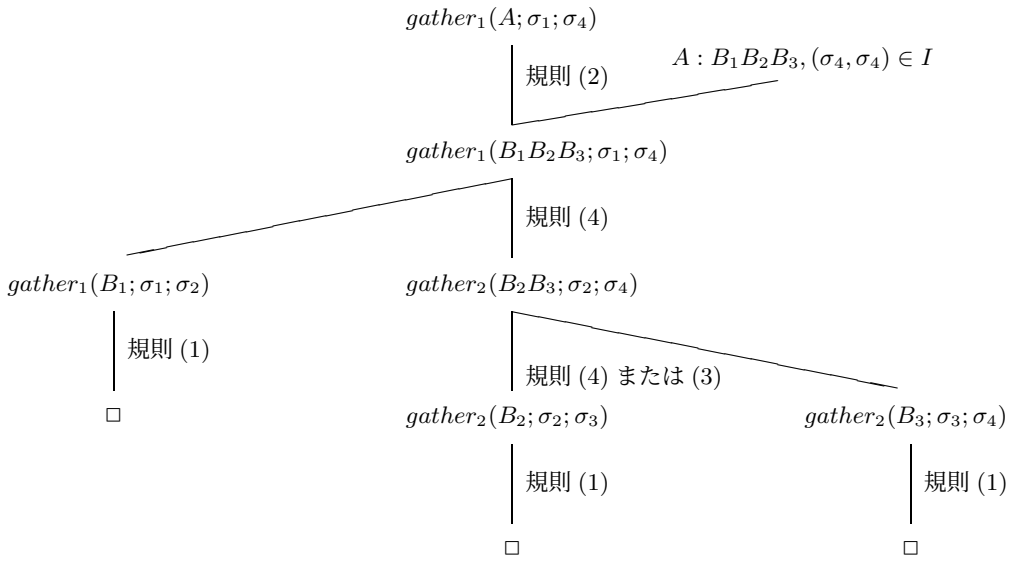

図 3 例題の実行
Fig. 3 The implementation of the example.

が必要となっている．このため，組み立て現場への部品の供給やそのために倉庫から部品を集める作業など，以前は大きな問題とは考えられなかった作業がボトルネックになってきており，効率化が求められている．しかし，これらは比較的複雑な作業であり，多くを人間の判断に依存せざるをえないのが現状である．

たとえば，大型計算機の組み立ての場合，BTO によって顧客から指示された組み立て仕様がもりこまれた組み立て指示が与えられる．組み立てごとに組み立てる手順や製品が異なるため，組み立て指示書を見て，必要な部品を倉庫から集めなければならない．すべての部品を一度に組み立て現場に集めてくることは，部品の収集に時間がかかることや一時的な置き場所が必要になるなどの不利益があるため，必要とされる部品から先に集めなくてはならない．組み立てにおいてどの部品が先に必要とされるかは製品ごとに異なるため，通常は組み立て指示書では指示されず，現場の人間の判断で行っている．

本論文で提案する数理モデルを利用すれば，今まで人間に頼っていた判断を自動化することができるため，多品種少量生産の現場における生産活動の助けとなるようなシステムを構築することができる．

3.2 部品の収集組み立て問題におけるモジュール計算系の直観的な意味

2.1 節で導入した数理モデルの部品組み立て問題における直観的な意味を説明する．部品組み立て問題においてはモジュールは部品，状況は倉庫の中での位置を意味する．したがって，モジュール計算系を定義する各要素は以下のように解釈される．

P： 部品の集合．

Σ： 倉庫の中での位置の集合．

$Process$： 部品の組み立て規則．すなわち，ある部品を組み立てるのにどの部品が必要かを示す．部品が基本要素であるということは，その部品がそのまま倉庫に入っていると解釈される．

$Situation$： 部品が基本要素である場合にはその部品が倉庫の中のどの位置に置いてあるかを意味する．基本要素でない場合には倉庫の中でその部品を組み立てる場所を意味する．同じ部品が複数の場所に置いてある場合もある．

I： 倉庫の中でどこからどこに移動できるかを示す．

モジュール計算系はこの問題においては部品を集めてくる人もしくは部品を組み立てる人に相当する．この問題では作業者と呼ぶことにする．並行モジュール計算系は複数の作業者で部品を組み立てる状況を示すことになる．

述語 $gather_i(A;\sigma_1;\sigma_2)$（$A \in P$，$\sigma_i$，$\sigma_2 \in \Sigma$）はこの問題において，作業者 i が倉庫内の場所 σ_1 から σ_2 に移動する間に部品 A を組み立てられるかどうかと解釈される．$gather_i(A;\sigma_1;\sigma_2)$ が成り立てば組み立てられるということである．そのとき，$\exists\beta \in P^*.\ SEM(\sigma_1;\beta;\sigma_3)$ で示される β は部品 A の組み立てに必要な部品の列を示している．

4. 実　　装

4.1 概　　要

前記のシステムについて，試験的な実装を行った．実装は Java で行い，規模は 3,600 行程度である．モジュール計算系 $gather_i$ をオブジェクトとして実装し，内部で再帰的に呼ぶようになっている．また，$gather$ の実行結果から，状況の遷移と実行モジュールの列を得るような仕組みとなっている．

図 4 にモジュール計算系が 3 つの場合のシステム図を示す．モジュール計算系はそれぞれに $Process_i$，$Situation_i$ を持ち，共通の P，Σ，I を参照する．モジュール計算系のうちのどれかに外部から問合せがあると，モジュール計算系どうしで適宜通信をして，最終的なモジュール列，状態遷移列を問合せが行われたモジュールから出力する．

図 1 の (2)，(3)，(4) の適用に関しては非決定的であるため，実装側で工夫する必要がある．今回の実装では，まず (2) によって，1 つのモジュール系のみで解決を試み，成り立たなかった場合に (3)，(4) を試行するようにした．このようにすると，モジュール系が含むルールを分割することにより，モジュール系ごとに役割を分担させることができる．

また，(1) においては，$\sigma_2 \in Situation_i(A)$ を満たすような σ_2 の探索が必要となる．このとき σ_2 が一意に決まらない場合，どれを選択するかは実装にまかされている．あまりかけ離れた位置の σ_2 を選ぶと非現実的な解となるので，制約関係 I で，ある意味で近いものを選ぶよう，広さ優先探索をして σ_2 を求めるようにしている．すなわち推移的に $(\sigma_1, \sigma_2) \in I$ を満たす σ_2 のうち，σ_1 に近いものから順に探し，はじめに見つかった $\sigma_2 \in Situation(A)$ について返す．

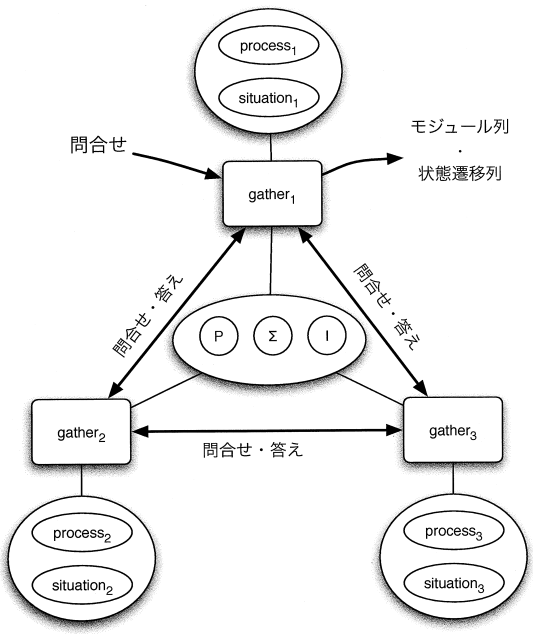

図 4 モジュール計算系が 3 つの場合のシステム図
Fig. 4 The system with three modules.

4.2 実 行 例 1

$\Pi = (\Im_1)$ において，

$P = \{X, Y, Z\}$
$\Sigma = \{s1, s2, s3\}$
$Process_1 = \{$
$X : \perp$
$Y : X$
$Z : X, Y$ $\}$
$Situation_1(X) = \{s1\}$
$Situation_1(Y) = \{s2\}$
$Situation_1(Z) = \{s3\}$
制約関係：
$(s1, s2) \in I$, $(s2, s1) \in I$
$(s2, s3) \in I$, $(s3, s2) \in I$

を与えたときの実行例を示す．直観的な意味としては，**図 5** に示したように，$s1$，$s2$，$s3$ という区画があり，隣り合った区画は互いに移動可能であって，区画 $s1$ には部品 X が在庫されており，区画 $s2$ では，部品 X を利用して部品 Y を組み立てることができる．また区画 $s3$ では部品 X と部品 Y を利用して製品 Z を組み立てることができる．

このとき，実行モジュールとして Z，初期状況として $s1$，終了状況として $s3$ を与えると，**図 6** のような実行結果が得られる．意味は，区画 $s1$ で部品 X を 2 つ収集し，そのあと区画 $s2$ へ移動して，部品 X を 1 つ使用して部品 Y を組み立て，そのあと区画 $s3$ へ移動して，部品 X と部品 Y から製品 Z を組み立てるというものである．

situations	modules
s3	Z : X, Y
s2	Y : X
s1	X :

図 5 実行例の状況とモジュール
Fig. 5 The situations and modules.

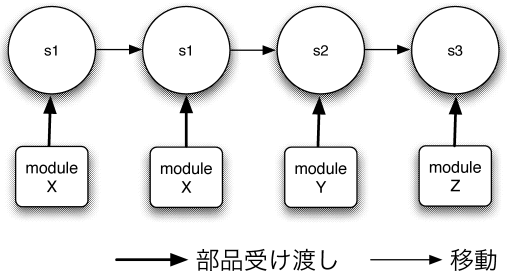

図 6 実行結果
Fig. 6 The result of implementation.

4.3 実 行 例 2

$\Pi = (\Im_1, \Im_2, \Im_3)$ において，

$P = \{W, X, Y, Z\}$

$\Sigma = \{s1, s2, s3\}$

$Process_1 = \{Z : W, X, Y$
$W : \perp\}$

$Process_2 = \{Y : \perp\}$

$Process_3 = \{X : \perp\}$

$Situation_1(Z) = \{s3\}$

$Situation_1(W) = \{s3\}$

$Situation_2(Y) = \{s2\}$

$Situation_3(X) = \{s1\}$

制約関係：

$(s1, s2) \in I$, $(s2, s1) \in I$

$(s2, s3) \in I$, $(s3, s2) \in I$

を与えたときの実行例を示す．

直観的な意味としては，4.2 節と同様，図 5 に示したように，$s1$，$s2$，$s3$ という区画があり，隣り合った区画は互いに移動可能である．区画 $s1$ には部品 X が在庫されており，区画 $s2$ には部品 Y が在庫されている．また区画 $s3$ では部品 W が在庫されており，さらに部品 X，Y，W を利用して製品 Z を組み立てることができる．ただし，作業者が 3 人おり，1 人は Z の組み立ておよび部品 W の収集を受け持ち，1 人は Y の収集のみを受け持つ．残りの 1 人は部品 X の収集を受け持つと考えられる．

このとき，実行モジュールとして Z，初期状況として $s3$，終了状況として $s3$ を与えると，**図 7** のような実行結果が得られる．意味は，作業員 1 は区画 $s3$ で部品 W を収集した後，製品 Z の組み立てを担当する．作業員 2 は区画 $s2$ で部品 Y の収集を担当する．また作業員 3 は部品 X を倉庫から出す係である．区画 $s3$ で製品 Z を組み立てようとすると部品 X，Y が必要になるので，作業員 1 に部品 X の収集，作業員 2 に部品 Y の収集の作業指示を出す．作業員 3 が $s1$ へ移動し，X を 1 つ収集し，作業員 1 へ渡す．作業員 2 は $s2$ に移動し部品 Y を収集して作業員 1 へ渡す．作業員 1 は $s3$ で部品 W を収集する．作業員 1 は必要な部品がすべてそろったので，製品 Z を生産する．

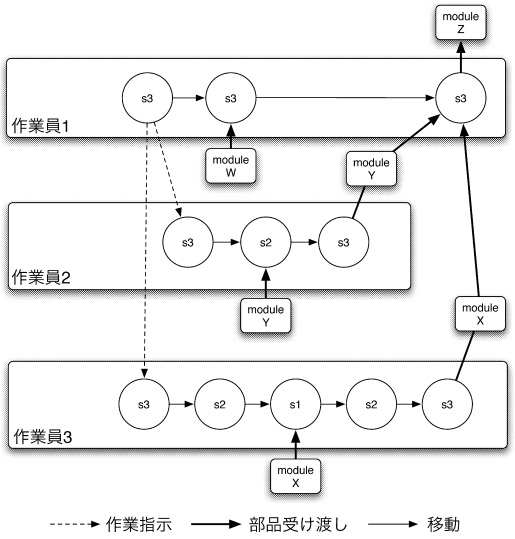

図 7 実行結果 2
Fig. 7 The result of the example 2.

5. 関 連 研 究

並行システム記述系としてよく知られているものに CCS や CSP に代表されるプロセス代数[1),5),9)] がある．これと比べると，本論文で提案している並行モジュール計算系は 1 つ 1 つのモジュール計算系が通信処理プロセスであると見なすことができる．

並行モジュール計算系では，それぞれのモジュールが知識に関する並行システムであると考えられる．この計算系ではそれぞれのモジュール計算系が扱える知識モジュールは必ずしも共有されていない形での分散システムになっている．またモジュールはアクションセマンティックス[11)] のアクションの一種と考えることもできる．

これまでの知識表現および推論の研究は，(1) 論理的解析，(2) エージェント技術，(3) イベント計算の観点から行われている．

(1) 論理的解析の観点では，人工知能の分野で研究されてきている様々な形式的システムと関連がある[4),8),10),13)]．
(2) エージェント技術に関しては文献 14) に詳しい．
(3) イベント計算では，文献 7) でイベントと状況の関係が述べられている．これは本論文のモジュールと状況の関係に似ている．この枠組みは文献 2) において様相論理を用いて定式化されている．イ

ベントを因果関係と見た場合に関しては文献 3) で詳しく論じられている．

我々はこれまでにも状況とモジュールを用いたシステムに関する研究を行い[16]，その一般的なユーザインタフェースの設計と構築も行っている[15]．本論文はそれをさらに並行系として拡張し，また，具体的な応用分野への適用を論じている．

6. おわりに

本論文では状況制約下における並行モジュールのシーケンスを求めるための健全な数理モデルとしてモジュール計算系を提案した．

また，その数理モデルの応用として，並行部品収集組み立て問題への適用を論じ，3 種類の制約を記述することによって，現実社会の複雑な事象を自然に記述することができることを示した．

実世界の問題への応用に際しては，問題の規模が大きくなることなどから，自動化が求められる．しかし本論文であつかった数理モデルでは，解を求めるための列が複数存在しうる．適当な解を求めるためには実装で工夫すべき点であり，今後の課題である．また，提案した数理モデルの関係 SEM に関する完全性については今後の課題である．

参考文献

1) Bruns, G.: *Distributed Systems Analysis with CCS*, Prentice-Hall (1996).
2) Cervesato, I., Chittaro, L. and Montanari, A.: A general modal framework for the event calculus and its skeptical and credulous variants, *Proc. 12th European Conference on Artificial Intelligence*, pp.12–16 (1996).
3) Dean, T. and Boddy, M.: Reasoning about partially ordered events, *Artificial Intelligence*, Vol.36, pp.375–399 (1988).
4) Genesereth, M.R. and Nilsson, N.J.: *Logical Foundations of Artificial Intelligence*, Morgan Kaufmann (1988).
5) Hoare, C.A.R.: *Communicating Sequential Processes*, Prentice-Hall (1985).
6) Kanban Just-In-Time at Toyota, *Management Begins at the Workplace*, Japan Management Association (Ed.), Productivity Press Inc. (1989).
7) Kowalski, R.A.: Database updates in the event calculus, *J. Logic Programming*, Vol.12, pp.121–146 (1992).
8) Lloyd, J.W.: *Foundations of Logic Programming*, 2nd Extended Edition, Springer-Verlag (1993).
9) Milner, R.: *Communication and Concurrency*, Prentice-Hall (1989).
10) Minker, J. (Ed.): *Foundations of Deductive Databases and Logic Programming*, Morgan Kaufmann (1987).
11) Mosses, P.M.: *Action Semantics*, Cambridge University (1992).
12) Nielson, F. (Ed.): *ML with Concurrency*, *Monograph in Computer Science*, Springer (1996).
13) Reiter, R.: *Knowledge in Action*, The MIT Press (2001).
14) Russell, S. and Norvig, P.: *Artificial Intelligence—A Modern Approach*, Prentice-Hall (1995).
15) Sasakura, M., Iwata, K. and Yamasaki, S.: An interactive environment for generating sequential information, *Information Visualization* (*IV06*), pp.441–416 (2006).
16) Yamasaki, S. and Sasakura, M.: A calculus effectively performing event formation with visualization, *Proc. ISHPC-VI* (CD-ROM) (2005).
17) 大野耐一：トヨタ生産方式—脱規模の経営をめざして，ダイヤモンド社 (1978).

付　　録

A.1 補題 1 の証明

証明 列 γ の長さに関する帰納法で示す．

(1) $\gamma = \epsilon$ の場合，定義 1 より $(\sigma_3, \sigma_4) \in I$．列 β の長さに関する帰納法を用いる．

(a) $\beta = \epsilon$ とすると，仮定より $SEM(\sigma_1; \beta; \sigma_2)$ なので，定義 1 から，$(\sigma_1, \sigma_2) \in I$．仮定から $(\sigma_2, \sigma_3) \in I$．また，$\gamma = \epsilon$ から，$(\sigma_3, \sigma_4) \in I$．関係 I は推移的であるので，$(\sigma_1, \sigma_4) \in I$．よって定義 1 から，$SEM(\sigma_1; \epsilon; \sigma_4)$．今の場合，$\beta\gamma = \epsilon$ なので，$SEM(\sigma_1; \beta\gamma; \sigma_4)$．

(b) $\beta = \beta' A$ ($\beta' \in P^*$, $A \in P$) とすると，仮定 $SEM(\sigma_1; \beta; \sigma_2)$ より $SEM(\sigma_1; \beta' A; \sigma_2)$．定義 1 から，

$$\exists \sigma_2' \in Situation(A),$$
$$(\sigma_2', \sigma_2) \in I, SEM(\sigma_1; \beta'; \sigma_2'). \qquad \text{(A.1.1)}$$

仮定から，$(\sigma_2, \sigma_3) \in I$．また，$\gamma = \epsilon$ により $(\sigma_3, \sigma_4) \in I$．関係 I は推移的なので，$(\sigma_2', \sigma_4) \in I$．式 (A.1.1) の $SEM(\sigma_1; \beta'; \sigma_2')$ と定義 1 により，$SEM(\sigma_1; \beta' A; \sigma_4)$．すなわち，$SEM(\sigma_1; \beta\gamma; \sigma_4)$．

(2) 列 γ の長さが k であるとき補題が成立すると仮定する．今列 γ の長さが $k+1$ で，$\gamma = \gamma' B$ ($\gamma' \in P^*$, $B \in P$) の場合，仮定から $SEM(\sigma_3; \gamma' B; \sigma_4)$．定義

1 から，

$\exists\sigma'_4 \in Situation(B)$,

$(\sigma'_4, \sigma_4) \in I, SEM(\sigma_3; \gamma'; \sigma'_4)$. (A.1.2)

関係 $SEM(\sigma_1; \beta; \sigma_2)$，$(\sigma_2, \sigma_3) \in I$ を仮定しているので，帰納法の仮定により，

$SEM(\sigma_1; \beta\gamma'; \sigma'_4)$. (A.1.3)

式 (A.1.2)，(A.1.3) より $SEM(\sigma_1; \beta\gamma' B; \sigma_4)$．すなわち，$SEM(\sigma_1; \beta\gamma; \sigma_4)$．これで帰納法が完了する．

□

A.2 定理 1 の証明

証明 ここではより一般的な主張 “述語 $gather_i(\Gamma_0; \sigma_1; \sigma_3)$（$\Gamma_0 \neq \epsilon$）が成立するときには，$\exists\beta \in P^*. SEM(\sigma_1; \beta; \sigma_3)$ である” を示す．主張の証明には，述語 $gather_j(-; -; -)$（$1 \leq j \leq n$）の再帰的な適用回数 k に関する帰納法を用いる．

（主張の証明）$gather_i(\Gamma_0; \sigma_1; \sigma_3)$（$\Gamma_0 \neq \epsilon$）が成立すると仮定する．

(1) $k = 0$ の場合．図 1 の規則 (1) によって，ある $(A : \bot) \in Process_i$ があって，$\Gamma_0 = A$ で述語 $gather_i(\Gamma_0; \sigma_1; \sigma_3)$ が得られることになる．$(\sigma_1, \sigma_2) \in I$ なので，定義 1 から，$SEM(\sigma_1; \epsilon; \sigma_2)$．$(\sigma_2, \sigma_3) \in I$，$\sigma_2 \in Situation(A)$ なので関係 SEM の再帰的定義を用いると，$SEM(\sigma_1; A; \sigma_3)$．

(2) $k \leq l$ の場合に，主張が正しいとする．$k = l + 1$ で，$gather_i(\Gamma_0; \sigma_1; \sigma_3)$（$\Gamma_0 \neq \epsilon$）が得られるとする．

(a) 規則 (2) により，述語 $gather_i(\Gamma; \sigma_1; \sigma_2)$ の仮定（再帰適用回数 l 以下）の下に，$(A : \Gamma) \in Process_i$，$\Gamma_0 = A$ で，述語 $gather_i(\Gamma_0; \sigma_1; \sigma_3)$ が得られているとする．帰納法の仮定により，ある β' があって $SEM(\sigma_1; \beta'; \sigma_2)$．規則 (2) にある，$(\sigma_2, \sigma_3) \in I$，$\sigma_2 \in Situation(A)$ の仮定を加えて，関係 SEM の定義により，$SEM(\sigma_1; \beta' A; \sigma_3)$．

(b) 規則 (3) により，述語 $gather_j(A; \sigma_1; \sigma_2)$ の仮定（再帰適用回数 l'），$gather_i(\Gamma; \sigma_2; \sigma_3)$ の仮定（再帰適用回数 l''）の下に，述語 $gather_i(\Gamma_0; \sigma_1; \sigma_3)$（$\Gamma_0 \neq \epsilon$）が成立すると仮定する（$l' + l'' \leq l$）．帰納法の仮定から，$SEM(\sigma_1; \beta; \sigma_2)$，$SEM(\sigma_2; \gamma; \sigma_3)$ となる，β，$\gamma \in P^*$ が存在する．関係 I が反射的で，$(\sigma_2, \sigma_2) \in I$ なので，補題 1 を適用すると，

$SEM(\sigma_1; \beta\gamma; \sigma_3)$.

(c) 規則 (4) により，述語 $gather_i(A; \sigma_1; \sigma_2)$ の仮定（再帰適用回数 l'），$gather_j(\Gamma; \sigma_2; \sigma_3)$ の仮定（再帰適用回数 l''）の下に，述語 $gather_i(\Gamma_0; \sigma_1; \sigma_3)$（$\Gamma_0 \neq \epsilon$）が成立すると仮定する（$l' + l'' \leq l$）．(b) の場合と同様に補題 1 を適用すると，

$SEM(\sigma_1; \beta'\gamma'; \sigma_3)$.

（主張の証明終り） □

(平成 18 年 4 月 28 日受付)
(平成 18 年 8 月 18 日再受付)
(平成 18 年 9 月 15 日採録)

岩田 健一

1993 年名古屋工業大学工学部電気情報工学科卒業，1995 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．同年日本電気株式会社入社．2002 年より岡山大学自然科学研究科博士後期課程在学中．

笹倉万里子（正会員）

京都大学工学部情報工学科卒業，(財) 京都産業情報センター，(財) 京都高度技術研究所勤務を経て，1995 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．1996 年同大学院大学情報科学研究科博士後期課程中退．同年岡山大学工学部情報工学科助手．2003 年同大学大学院自然科学研究科助手．現在に至る．博士（工学）．プログラムや推論等の視覚化システムに関する研究に従事．日本ソフトウェア科学会，人工知能学会，IEEE-CS 各会員．

山崎 進

1970 年京都大学工学部電気工学第二学科卒業．1974 年同大学大学院工学研究科博士後期課程退学．同年より 1987 年まで京都大学工学部情報工学科に勤務．京都大学工学博士．1987 年岡山大学工学部情報工学科教授．1985/1986 年英国ウォーリック大学計算機科学科客員研究員．言語理論，定理自動証明，プログラミング言語の意味論，非単調推論系，分散プログラミング等の研究に従事．EATCS，AAAI，日本数学会各会員．