数理解析研究所講究録
第 381 巻 1980 年 223-241

同期付生成システム(SPS)について

名古屋大　工　山下雅史
三重大　工　稲垣康善
名古屋大　工　本多波雄

1. まえがき

一般に，並行処理システム(あるいは，その同期機構)は，並行に実行される各プロセス(事象の系列)の構成やその出現法則を定める機構と，各事象がその実行時に満足すべき条件を定める機構の対として表現できる．本稿では，上に述べた第1の機構を文脈自由文法によって，第2の機構をオートマトンによって記述する新しい並行処理システムモデルとして，同期付生成システム(Synchronized Production System-SPS)を提案する．さらに，ペトリネット言語と同じ意味でSPS言語を定義し，これを用いてSPSの各部分クラスの間の，又，SPSと他のモデルとの間の能力比較を行う．尚，本報告で議論の対象とするのは主に正規文法と有限オートマトンによって定義されるSPSである．

2. 同期付生成システム(SPS)

本章では，SPSの定義を与える．まず，以下で使用するいくつかの記法を約束しておく．

［記法］ i)記号の集合Σ上の正規集合の族を$R(\Sigma)$で示す．ii)正規集合αの系列xによる微分を$\partial_x \alpha$とする．即ち，$\partial_x \alpha = \{u \mid xu \in \alpha\}$である．iii)非負整数の$n$項組$v = (v_{\sigma_1}, \cdots, v_{\sigma_n})$の要素$v_{\sigma_i}$を$(v)_{\sigma_i}$で表わす．又，$v \pm \sigma$ $(\sigma \in \{\sigma_i\})$によって，$v_\sigma$に$(v)_\sigma \pm 1$を代入する事によって$v$から作られる$n$項組（複号同順）を示す．iv)集合A，Bに対して，$A - B = A \cap \neg B$ とする．v) λは空系列，0は零ベクトルを示す．▨

［定義1］ SPSは2項組$(\mathcal{P}, \mathcal{A})$である．ここに，$\mathcal{P}$は4項組$(\Sigma_N, \Sigma_T, S, P)$で定められるシステムであり，$\Sigma_N$は非終端記号の集合，$\Sigma_T$は終端記号の集合（$\Sigma = \Sigma_N \cup \Sigma_T$とする），$S$は$\Sigma_N$の特別な要素で開始記号と呼ばれる．又，$P$は生成規則の集合であり，各生成規則は，$A \to \alpha$ $(A \in \Sigma_N, \alpha \in R(\Sigma), \lambda \notin \alpha)$の形を持つ．$\mathcal{A}$は図1に示すようなオートマトンであり，その計算状況は，オートマトンの内部状態と補助テープの状況の2項組で表現される．$\mathcal{A}$の入力記号集合は

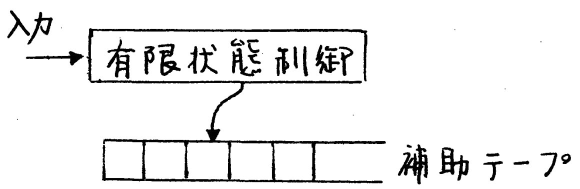


図1. オートマトン

Σである．$\mathcal{A}$が計算状況cにあって，sを入力され，計算状況c'に推移したとき，この推移を$c \vdash^{s}_{\mathcal{A}} c'$と表わす．文脈から明らかなとき，$s$を省略することがある．又，$\vdash$の推移的閉包を$\overset{*}{\vdash}_{\mathcal{A}}$と書く．▨

〔定義2〕SPSの時点表示(id)は2項組(v, c)である．ここに，$\Omega = \{\sigma \mid \exists(A \to \alpha) \in P, \exists x \in \Sigma^*, \sigma = \partial_x \alpha - \{\lambda\}\} \cup \{\{S\}\}$とすると，$v$は非負整数の$|\Omega|$項組，すなわち，$v = \otimes_{\sigma \in \Omega} v_\sigma$である．又，$c$はオートマトン$\mathcal{A}$の計算状況である．▨

〔定義3〕時点tのid $I_t = (v_t, c_t)$から時点$t+1$のid $I_{t+1} = (v_{t+1}, c_{t+1})$への推移は以下のように定義される．

$I_t = (v_t, c_t)$に対して，$N_t = \{(s, \sigma, c) \mid s \in \Sigma, \sigma \in \Omega, \partial_s \sigma \neq \phi, (v_t)_\sigma \geq 1, c_t \vdash^{s}_{\mathcal{A}} c\}$とする．

このとき，$I_t = (v_t, c_t)$から$I_{t+1} = (v_{t+1}, c_{t+1})$への推移が存在する必要十分条件は，$N_t$の要素$(s, \sigma, c)$が存在して，$c_{t+1} = c$かつ，$v_{t+1}$が以下に定められるようなものであることである．$N_t$の要素$(s, \sigma, c)$に対して，$\Delta_1 = v_t - \sigma$，$\Delta_2 = \Delta_1 + (\partial_s \sigma - \{\lambda\})$とする．

(1) $s \in \Sigma_T$のとき，

$$v_{t+1} = \begin{cases} \Delta_1 : \partial_s \sigma = \{\lambda\} \text{ のとき} \\ \Delta_2 : \partial_s \sigma \neq \{\lambda\} \text{ のとき} \\ \Delta_1 \text{ 又は } \Delta_2 : \text{それ以外のとき} \end{cases}$$

である．

(2) $s \in \Sigma_N$ のとき，$(s \to \alpha) \in P$ に対して，$\Delta_1' = \Delta_1 + \alpha$，$\Delta_2' = \Delta_2 + \alpha$ とする．

$$\mho_{t+1} = \begin{cases} \Delta_1' : D_s\sigma = \{\lambda\} \text{ のとき} \\ \Delta_2' : D_s\sigma \neq \{\lambda\} \text{ のとき} \\ \Delta_1' \text{ 又は } \Delta_2' : \text{それ以外のとき，である．} \end{cases}$$

$(s, \sigma, c) \in N_t$ によって I_t から I_{t+1} への推移ができるとき，これを $I_t \overset{s}{\vdash} I_{t+1}$ と表わす．s を推移 $I_t \overset{s}{\vdash} I_{t+1}$ で読まれた記号という．s を明示する必要がなければ省略する．$I_1 \overset{s_1}{\vdash} I_2 \overset{s_2}{\vdash} \cdots \overset{s_{n-1}}{\vdash} I_n$ であるとき，$I_1 \overset{\sigma}{\vdash} I_n$ $(\sigma = s_1 \cdots s_{n-1})$ と書く．又，$\exists \sigma \in \Sigma^*$ かつ，$I_1 \overset{\sigma}{\vdash} I_n$ であるとき，$I_1 \overset{*}{\vdash} I_n$ と書く．▨

[定義4] SPS $\mathcal{S} = (\mathcal{P}, \mathcal{A})$ の動作とは，id の推移列，$I_0 \vdash I_1 \vdash \cdots$ である．ここに，$I_0 = (\mho_0, c_0)$ $(\mho_0 = \emptyset + \{S\}$，$c_0$ は $\mathcal{A}$ の初期計算状況$)$ である．▨

[定義5] $N_t = \phi$ であるような id I_t に到達したとき，SPS は停止する．特に $\mathcal{S}$ が $I_h = (\mho_h, c_h)$ $(\mho_h = \emptyset)$ に到達したとき，$\mathcal{S}$ は完了したという．さらに，c_h が $\mathcal{A}$ の受理計算状況であるとき，$\mathcal{S}$ は受理状態で完了したという．▨

SPS $\mathcal{S} = (\mathcal{P}, \mathcal{A})$ の動作は，以下のように定義されるマーク付導出木の動作によって，模型的に表現できる．

[定義6] $\mathcal{P} = (\Sigma_N, \Sigma_T, S, P)$ において，$(A \to \alpha) \in P$ は，$\forall u\ (\in \alpha)\ [(A \to u) \in P]$ の略記である．従って，$\mathcal{P}$ は無限個の生成規則を持ち得る文脈自由文法である．▨

[定義7] $\mathcal{P}$の導出途中木および導出木の集合を$\mathcal{D}(\mathcal{P})$と書く．導出途中木も単に導出木と呼ぶことにする．$d \in \mathcal{D}(\mathcal{P})$の項(item)とは，根節点S，および，同じ親を持つ子節点の文字列(生成規則の右辺に現れる)である．導出木dの各項に高々1個の"•"(mark)を挿入してできる木をマーク付導出木(Marked Derivation Tree - mdt)と呼ぶ．$\mathcal{P}$のmdtの集合を$M(\mathcal{P})$と書く．(図2参照) ▨

このとき，SPSの時点表示，および動作は以下のように定義することもできる．

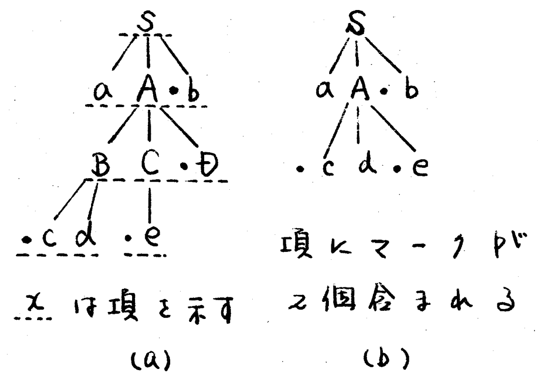


図2 (a) mdt (b) mdtではない

[定義8] SPS $\mathcal{S}=(\mathcal{P},\mathcal{A})$の時点表示(id)は2項組$I=(m, c)$である．ここに，$m \in M(\mathcal{P})$，$c$は$\mathcal{A}$の計算状況である．▨

[定義9] 時点tにおけるidが$I_t=(m_t, c_t)$であるとき，時点$t+1$のid $I_{t+1}=(m_{t+1}, c_{t+1})$は以下のように定義される．$m_t$に含まれるマーク直後の記号で，しかも$\mathcal{A}$が計算状況$c_t$のときに受付けることが可能な記号の集合を$N_t$とする．$s \in N_t$に対して，$\mathcal{A}$は$s$を入力して計算状況を$c_{t+1}$に移すとする．$s$を含む項を$s_1 \cdots s_{i-1} \bullet s\, s_{i+1} \cdots s_k$とする．

(1) $s \in \Sigma_T$ のとき，項を $s_1 \cdots s_{i-1}\, s \bullet s_{i+1} \cdots s_k$ に置換えた mdt を m_{t+1} とする．

(2) $s \in \Sigma_N$ のとき，項を $s_1 \cdots s_{i-1}\, \underset{\bullet n_1 \cdots n_\ell}{s} \bullet s_{i+1} \cdots s_k$ に置換えた mdt を m_{t+1} とする（ただし，$(s \rightarrow n_1 \cdots n_\ell) \in P$ とする.）

(3) 操作 (1)，(2) の結果，マークが項の右端にくれば，そのマークは消去される．

SPSはN_tの中の一つの元を非決定的に選択して，次のidに推移する．このとき，選択されたN_tの要素sを推移によって読まれた記号といい，この推移を $I_t \Vdash^{s} I_{t+1}$ と表わす．文脈から明らかなときsを省略する．$I_1 \Vdash^{s_1} I_2 \Vdash^{s_2} \cdots \Vdash^{s_{n-1}} I_n$ であるとき，$I_1 \Vdash^{\sigma} I_n$ ($\sigma = s_1 \cdots s_{n-1}$) と書く．また，$\exists \sigma \in \Sigma^*$ かつ，$I_1 \Vdash^{\sigma} I_n$ のとき $I_1 \Vdash^{*} I_n$ と書く．▨

[定義10] SPS $\mathcal{S} = (P, A)$ の動作は，idの推移列 $I_0 \Vdash I_1 \Vdash \cdots$ である．ここに，$I_0 = (\bullet S, C_0)$ （C_0はAの初期計算状況）である．

SPS $\mathcal{S}$は，$N_t = \phi$ となる id $I_h = (m_h, C_h)$ に到達したとき停止する．特に，$m_h \in D(P)$ のとき（マークが全てなくなったとき）完了したという．さらに，C_hがAの受理計算状況であれば，$\mathcal{S}$は受理状態で完了したという．▨

ここで，定義2~5で定めたSPSの動作と定義8~10で定め

たSPSの動作が本質的に同等であることを示す.

≪補題1≫任意のSPS $\mathcal{S}$ に対して, $[I_0 = (\mho_0, c_0) \vdash^{\sigma} I_* = (\mho_*, c_*)] \Leftrightarrow [I_0 = (\bullet S, c_0) \Vdash^{\sigma} I_* = (m_*, c_*)]$ $(\mho_0 = 0 + \{S\})$ かつ, $\mho_*$ と m_* は以下の関係をみたす.

m_* のマークの付いた項を $x \bullet y$ とする. xy の導出に用いられた生成規則を $A \to \alpha$ (即ち $xy \in \alpha$) とする. このとき, $\forall_x \alpha - \{x\} = \sigma$ となるようなマーク付きの項 $x \bullet y$ の個数は $\mho_\sigma$ に一致する.

(略証) 推移列の長さに関する帰納法により証明できる. ▨

補題1から, 二つの定義のいずれをSPSの動作の定義としてもよいことが分った. 以降では, $\vdash$ と $\Vdash$ を区別せず $\vdash$ と書く. SPSは以下のように解釈することによって並行処理システムを表現する.

(1) 各項は, 並行に処理されるプロセス

(2) マークは, 各プロセスに割付られたプロセッサの実行位置

(3) Σ_N の元は, 新しいプロセスを作りプロセッサをそのプロセスに割付る命令

(4) Σ_T の元は, プロセスを構成する事象, を示す.

(例1) 2種類のタスク T_1, T_2 を処理するシステムを考える. 無作為に, 任意の個数の T_1, T_2 がシステムに到来するものと

する．各タスクは，到来と同時にプロセッサを割付られ，並行に処理されるが，各タスクの実行は互いに相互排除されねばならないものとする．このシステムをSPSによって表わす．

$\mathcal{S}=(\mathcal{P},\mathcal{A})$，$\mathcal{P}=(\{S\},\{T_1,T_2,\sigma,\omega\},S,P)$，$P=\{S\to S\omega T_1\sigma \mid S\omega T_2\sigma \mid \omega T_1\sigma \mid \omega T_2\sigma\}$，$\mathcal{A}=(\{q_0,q_1\},\{S,T_1,T_2,\omega,\sigma\},\delta,q_0,\{q_0\})$ 図3の有限オートマトンである．

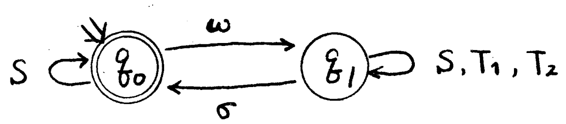


図3. 有限オートマトン

$\mathcal{S}$の動作の一例を図4に示す．

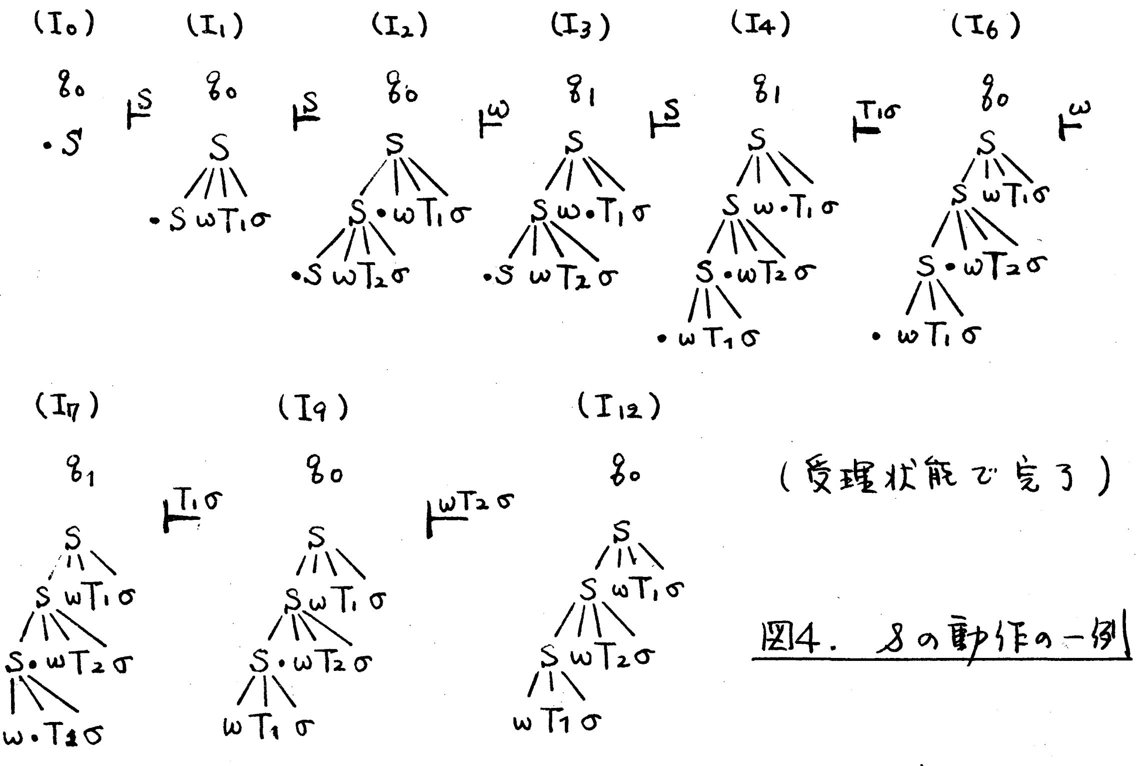


図4. $\mathcal{S}$の動作の一例

$\mathcal{S}$はSで2種類のタスクを受付，プロセッサを割付る一種のオペレーティング・システムの抽象化と考えられる．σとω

はタスク間の相互排除を実現するためのセマフォである．I4では，システムは3つのプロセス$wT_1\sigma$，$wT_2\sigma$，および途中まで実行の進んだ$wT_3\sigma$を持っている．又，I6でマークの直後の2つのwのいずれを実行するかの選択は非決定になされる．即ち，一度プロセッサを割付けられたプロセスは，以前からあるプロセスと同一の取扱いを受ける．▨

3. SPS言語

種々の並行処理システムのモデルの能力比較には，例えばペトリネット言語(1)のように，正しい最終状態に至る可能な計算履歴の集合によって定義される言語がしばしば利用される．そこで，SPSに対しても，SPS $\mathcal{S}$によって定義される言語$L(\mathcal{S})$を定義する．

[定義11] 準同形写像$h: \Sigma \to \Sigma_* \cup \{\lambda\}$を$h(s) = u$ (u = if ($s \in \Sigma_*$) then s else λ)で定義する．

$\sigma = \sigma_1 \cdots \sigma_n$ ($\sigma_i \in \Sigma$) に対して，hを$h(\sigma) = h(\sigma_1) \cdots h(\sigma_n)$によって拡張する．

推移列$I = I_1 \cdots I_n$，$I_1 \vdash I_n$に対して，$\pi(I) = h(\sigma)$と定義する▨

[定義12] SPSにおいて，$\mathcal{S}$を受理状態で完了させる推移列の集合を$\mathcal{I}(\mathcal{S})$とかく．このとき，$L(\mathcal{S}) = \{\pi(I) \mid I \in \mathcal{I}(\mathcal{S})\}$を$\mathcal{S}$によって定義されるSPS言語と呼ぶ．▨

本稿の以下の部分では，オートマトンが有限オートマトン

で表現されるSPSおよびSPS言語について考察する．オートマトンが有限オートマトンであるSPSのクラスを$\mathcal{M}$，$\mathcal{L}=\{L(\mathcal{S})\mid \mathcal{S}\in\mathcal{M}\}$によって$\mathcal{M}$に対応するSPS言語の族を示す．

3.1 $\mathcal{M}_B$，$\mathcal{L}_B$ の性質　本節では，有限台数のプロセッサによって実行できる（あるいは，同時に並行に実行されるプロセス数に上限がある）という意味で，実際的なシステムのモデルに対応するSPSおよびSPS言語の族を考察する．

〔定義13〕$\mathcal{M}_B=\{\mathcal{S}\mid {}^{\exists}k, {}^{\forall}I_t,\ I_0 \overset{*}{\vdash} I_t=(v_t, g_t)\Rightarrow \sum_{\sigma}(v_t)_\sigma \leq k\}$，$\mathcal{L}_B=\{L(\mathcal{S})\mid \mathcal{S}\in\mathcal{M}_B\}$とする．▨

≪定理1≫正規集合の族を$\mathcal{R}$として，　$\mathcal{L}_B=\mathcal{R}$

（略証）$\mathcal{R}\subseteq\mathcal{L}_B$ は明らか．$\mathcal{L}_B\subseteq\mathcal{R}$を示す．どの$\mathcal{S}\in\mathcal{M}_B$ についても，その id の種類は有限個しかない．従って，$L(\mathcal{S})\in\mathcal{R}$である．▨

このように，システムに現れるプロセッサ数を有限個数に制限すれば，対応する言語は単純な族$\mathcal{R}$に含まれる．従って，以下では，プロセッサ数の上限がないシステムを考察の対象とする．しかし，$\mathcal{S}\in\mathcal{M}$が$\mathcal{M}_B$に入るか否かを決定することができることを示しておくことはシステム解析の立場からも意味のあることであろう．

≪定理2≫${}^{\forall}\mathcal{S}\in\mathcal{M}$に対して，$\mathcal{S}\in\mathcal{M}_B$ か否かを決定する問題は可解である．

（略証）次の二つの命題 1)，2) は同値である．

1) $\mathcal{S} \notin \mathcal{M}_B$

2) $(v_0, q_0) \vdash^* (v, q) \vdash^* (v', q)$ $(v < v')$ となる $\mathcal{S}$ の動作が存在する．

2) ⇒ 1) は容易．1) ⇒ 2) を示す．1) が成立し，2) が成立しないと仮定すると，次のような id の無限列が存在する．

$I_0 \vdash^* I_1 \vdash^* I_2 \vdash^* \cdots$ $(I_i = (v_i, q)\ (1 \leq i))$，かつ $\forall i, j \geq 1$，$v_i \nleq v_{i+j}$，かつ，$\Xi(v_i)_\sigma < \Xi(v_{i+j})_\sigma$．

$v_i \nleq v_{i+j}$ かつ $\Xi(v_i)_\sigma < \Xi(v_{i+j})_\sigma$ となる v_i の列は常に有限列であるから，仮定と矛盾する．従って，1) と 2) が同値であることが示された．

故に，可能な id の推移列の全てを探索すると，$\mathcal{S} \in \mathcal{M}_B$ ならば，探索は有限ステップで終了し，又，$\mathcal{S} \notin \mathcal{M}_B$ ならば，2) をみたす id の推移列を見出すことができ，$\mathcal{S} \notin \mathcal{M}_B$ が認識できることから，この問題は可解である．▨

3.2 $\mathcal{M}_1$，$\mathcal{L}_1$ の性質　SPS の文法が $A \to \alpha B \mid \beta$ $(\alpha, \beta \in R(\Sigma))$ という形を持つクラスは $\mathcal{M}_B$ に含まれるので（実際，対応する SPS 言語の族は $\mathcal{L}_B$ に等しい）単純なクラスである．しかし，文法が $A \to B\alpha \mid \beta$ $(\alpha, \beta \in R(\Sigma))$ という形を持つ場合には，上記のクラスとは異なり，その小さい部分クラスにおいても，かなり大きい能力を持つことが示される．

［定義14］SPSを定義する文法が，$A \to Bu \mid v$ $(u, v \in \Sigma_T^*,$ $|u|, |v| \leq k)$ という形のSPSのクラスを$\mathcal{M}_k$で示す．$\mathcal{L}_k = \{L(\mathcal{S}) \mid \mathcal{S} \in \mathcal{M}_k\}$とする．▨

本稿の以下の部分では$\mathcal{M}_1$と$\mathcal{L}_1$について考察する．

≪定理3≫ベクトル加算システム（VAS）の到達可能問題と$\mathcal{L}_1$の空問題は互いに帰着可能である．

（証明）A）VASの到達可能問題が$\mathcal{L}_1$の空問題に帰着できることを示す．任意に与えられたVASを$\mathcal{V} = (v_0, V)$とする．v_0：非負整数のn項組，$V = \{v_1, \ldots, v_m\}$：整数のn項組の有限集合，である．$\mathcal{V}$の到達可能集合を$R(\mathcal{V})$とする．非負整数のn項組v_fが与えられたとき，v_fが$R(\mathcal{V})$に属するか否かを決定する問題を，$\mathcal{L}_1$の空問題に帰着する．$v_i = (c_1^i, \ldots, c_n^i)$ $(0 \leq i \leq m, f)$とする．v_i中の中の要素を集めたベクトルを$A_i = (c^i_{a(i,1)}, \cdots, c^i_{a(i,l_i)})$，負の要素を集めたベクトルを$B_i = (c^i_{b(i,1)}, \cdots, c^i_{b(i,t_i)})$とする．このとき，以下のようにSPS $\mathcal{S} \in \mathcal{M}_1$を構成する．$\mathcal{S} = (\mathcal{P}, \mathcal{A})$，$\mathcal{P} = (\Sigma_N, \Sigma_T, S, P)$，$\Sigma_N = \{S, \bar{S}\} \cup \{S^i_{a(i,j),k} \mid 0 \leq i \leq m,\ 1 \leq j \leq l_i,\ 1 \leq k \leq c^i_{a(i,j)}\}$，$\Sigma_T = \{\sigma_1, \ldots, \sigma_n\}$，$P = \{S \to S^0_{a(0,1),1}\sigma_{a(0,1)}\} \cup \{\bar{S} \to S^i_{a(i,1),1}\sigma_{a(i,1)} \mid 1 \leq i \leq m\} \cup \{S^i_{a(i,j),k} \to S^i_{a(i,j),k+1}\sigma_{a(i,j)},\ S^i_{a(i,l_i),c^i_{a(i,l_i)}-1}{}^{\dagger} \to \sigma_{a(i,l_i)} \mid$

---

†) $c^i_{a(i,l_i)} = 1$のとき$S^i_{a(i,l_i),0} = S^i_{a(i,l_i-1),c^i_{a(i,l_i-1)}}$とする．

$0 \le i \le m,\ 1 \le j \le l_i,\ 1 \le k \le c^i_{a(i,j)}-1 \} \cup \{ s^i_{a(i,j),c^i_{a(i,j)}} \to s^i_{a(i,j+1),1}\ \sigma_{a(i,j+1)} \mid 0 \le i \le m,\ 1 \le j \le l_i - 1 \} \cup \{ s^i_{a(i,l_i),c^i_{a(i,l_i)}} \to \bar{s} \mid 0 \le i \le m \}$，である．$\mathcal{A} = (K, \Sigma, \delta, q_s, F)$ は，$K = \{q_A \mid A \in \Sigma_N\} \cup \{q_B \mid B = T^i_{b(i,j),k},\ 1 \le i \le m,\ 1 \le j \le t_i,\ 1 \le k \le c^i_{b(i,j)}\} \cup \{q_C \mid C = R^f_{a(f,j),k},\ 1 \le j \le l_f,\ 1 \le k \le c^i_{b(i,j)}\}$，$\Sigma = \Sigma_N \cup \Sigma_T$，$F = \{ q_{R^f_{a(f,l_f),c^f_{a(f,l_f)}}} \}$，δは図5の推移図で与えられる．ただし，Ⓐは q_A を示す．さらに，$A = X^i_{y(i,j),k}$ $(X \in \{S, T, R\},\ y \in \{a, b\})$ を $X^i_{j,k}$ と，特に，$k = c^i_{y(i,z_i)}$ $(y \in \{a, b\},\ z \in \{l, t\})$ のとき，$X^i_{j,c}$ と略記する．

δは次の諸性質を持つ．

i) 初めて id が $(v, q_{\bar{s}})$ に到達したとき，v は $\forall \sigma_j$ $(1 \le j \le n)$ に対して $(v)\sigma_j = (v_0)_j$ をみたす．

ii) どの id $I = (v, q_{\bar{s}})$ ならびに I から i $(1 \le i \le m)$ のループを一周した id $I' = (v', q_{\bar{s}})$ とに対しても，$\Delta = v' - v$ とすると，$\forall \sigma_j$ $(1 \le j \le n)$ に対して，$(\Delta)\sigma_j = (v_i)_j$ をみたす．

iii) どの id $I = (v, q_{\bar{s}})$ に対しても I からループを通らないで id $I' = (v', q_{R^f_{l_f,c}})$ に到達したとすると，$\Delta = v - v'$ とすれば，$\forall \sigma_j$ $(1 \le j \le n)$ に対して，$(\Delta)\sigma_j = (v_f)_j$ をみたす．

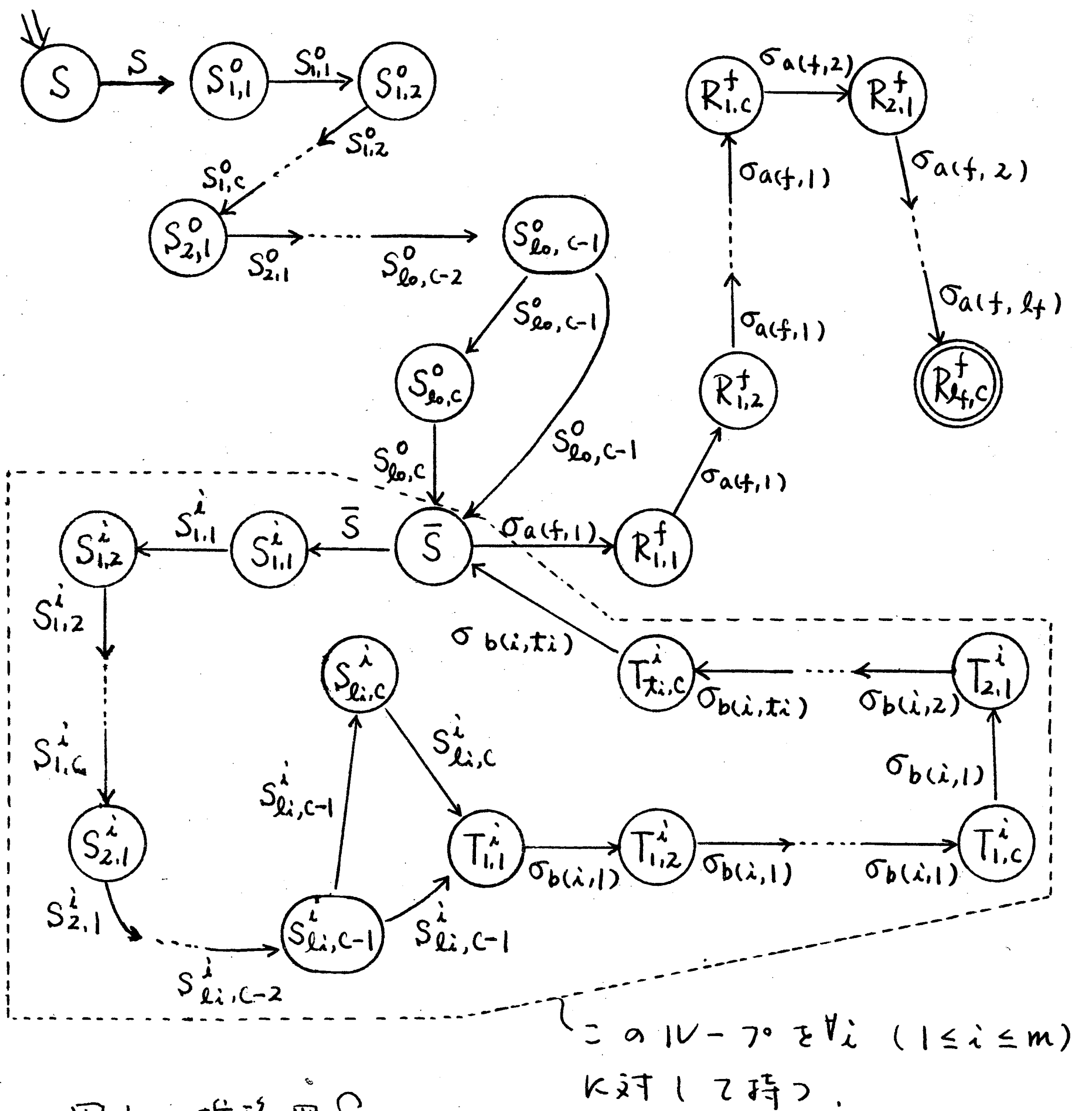


図5. 推移図δ

以上の性質から，$\mathcal{R}(\mathcal{V}) \ni v_f \Leftrightarrow L(\delta) \neq \phi$ である．

B) $\mathcal{L}_1$ の空問題をVASの到達可能問題に帰着する．id $I = (U, q)$ のベクトル表現とは，$v(I) = U \otimes V$ ($V = \bigotimes_{r \in K} V_r$, $V_q = 1$, $\forall q' \neq q$, $V_{q'} = 0$) である．このとき，VAS $\mathcal{V} = (v_0, V)$ を以下のように構成する．$v_0 = v(I_0)$ ($I_0 = (U_0, q_0)$, $U_0 = 0 + \{S\}$, q_0 はオートマトンの初期状態)，$V =$

$\{v \mid v = v(I') - v(I)$, $I = (U, q)$, $\sum_{\sigma}(U)\sigma = 1$, $I \vdash I'\}$
とすれば，SPSの動作の定義より，

$$L(\mathcal{S}) \neq \phi \Leftrightarrow \mathcal{R}(\mathcal{V}) \ni 0 \qquad ▨$$

定理3から，$\mathcal{M}_1$ の同期能力はVASと同等であると考えられるが，SPS言語の族$\mathcal{L}_1$は空系列入のラベル付を許したペトリネット言語の族$\hat{\mathcal{L}}_{PN}$に真に含まれることが証明できる．

《定理4》 $\hat{\mathcal{L}}_{PN} \supsetneq \mathcal{L}_1$

A) $\hat{\mathcal{L}}_{PN} \supseteq \mathcal{L}_1$ を示す．$\forall \mathcal{S} \in \mathcal{M}_1$ に対して，$L(\mathcal{S})$ に等しいペトリネット言語を持つペトリネットNを構成する．$\mathcal{S} = (\mathcal{P}, \mathcal{A})$ に対し，以下の要領で $N = (P_N, T_N, E_N, M_0, \mu$ を作る．μはラベル付関数である．まず，Pに含まれる規則に出現する終端記号に添字を付け，規則の異った場所に出現する終端記号は互いに区別できるようにしておく．$\mathcal{P} = (\Sigma_N, \Sigma_T, S, P)$ ，$\mathcal{A} = (K, \Sigma, \delta, q_0, F)$ とする．

<u>構成手続</u>　作られるペトリネットNのP_Nは，以下の手続で作られる p_u の形をした記号の集合，T_Nは，t_y^x の形をした記号の集合，E_Nは (p_u, t_y^x) 又は (t_y^x, p_u) の形をした2項組の集合である．$K_\alpha = \{q \mid \delta(q, \alpha) = q'$ for some q' in $K\}$ $(\alpha \in \Sigma)$ とする．（ただし，終端記号の添字は無視して考える．）

(I) 開始記号Sに対してp_Sを作る．新しいp節点が現れな

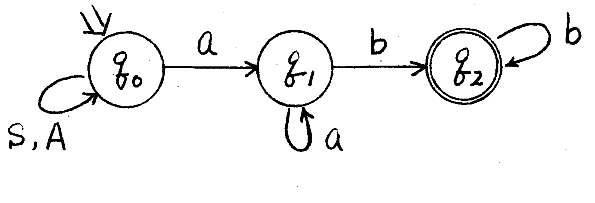
くなるまで以下の手続を続ける．p節点を $p_{\alpha_1,\cdots,\alpha_n}$ とする．α_i $(1 \le i \le n)$ に対して以下の操作を行なう．

i) $\alpha_i = a$ $(\in \Sigma_T)$ のとき，$\forall q \in K_a$ に対して，$t^q_{\alpha_i}$ と $(p_{\alpha_1,\cdots,\alpha_n}, t^q_{\alpha_i})$ とを作る． ii) $\alpha_i = A\alpha$ (又は，$A = S$, $a = \lambda$) のとき，$\forall q \in K_A$ に対して，t^q_{Aa} と $(p_{\alpha_1,\cdots,\alpha_n}, t^q_{Aa})$ とを作る．$A \to B_1b_1 \mid B_2b_2 \mid \cdots \mid B_lb_l \mid b_{l+1} \mid \cdots \mid b_{l+k}$ が P の元であるとする．$(t^q_{Aa}, p_{B_1b_1,\cdots,b_{l+k}})$ を作る．$a \neq \lambda$ のときは，p_a と (t^q_{Aa}, p_a) とを作る．さらに，$p_{B_1b_1,\cdots,b_{l+k}}$ が以前から作られていなければ，これを作る．

(II) $\forall q \in K$ に対して，p_q を作る．$\forall q$, t^q_x に対して，$\delta(q, x) = q'$ とすれば，(p_q, t^q_x)，$(t^q_x, p_{q'})$ を作る．

(III) $M_0(p_S) = M_0(p_{q_0}) = 1$，他の節点に対しては $M_0(p) = 0$ である．

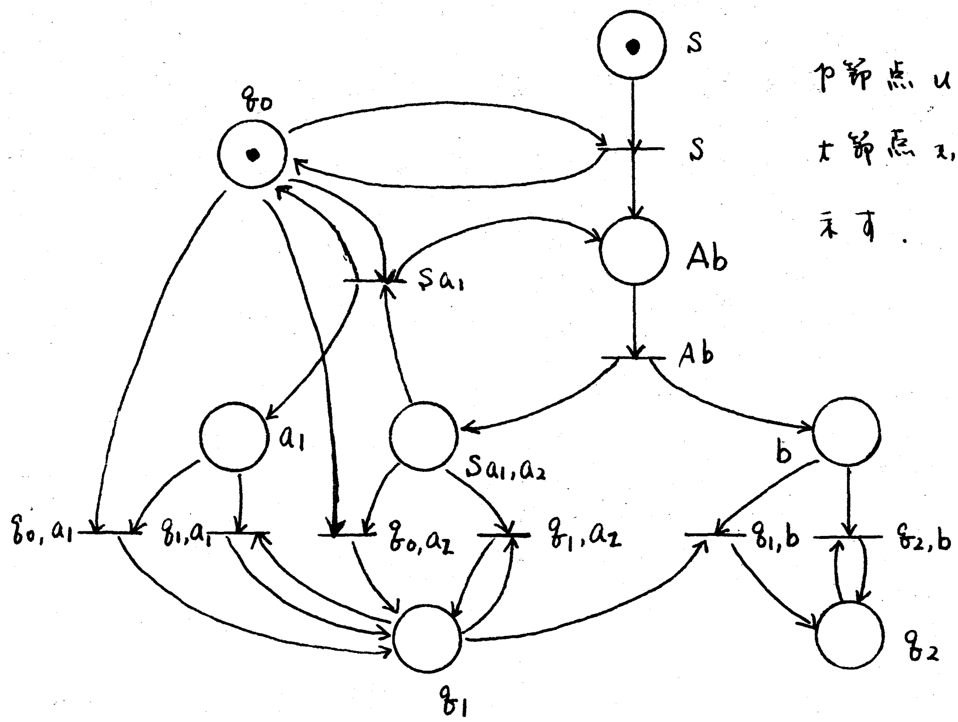
(例2) $\mathcal{S} = (\mathcal{P}, \mathcal{A})$, $\mathcal{P} = (\Sigma_N, \Sigma_T, S, P)$, $P = \{S \to Ab, A \to Sa \mid a\}$, $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ は下の推移図で与えられるものとする．



$L(\mathcal{S}) = \{a^i b^i \mid i \geq 1\}$ である．

gに対して作られるペトリネットはPを$P' = \{S \to Ab, A \to Sa_1 | a_2\}$に変更し、手続に従うと図6のようになる。

（例終り）



p節点uはp_uを、t節点x, yはt_y^xを示す。

図6　gから作られるペトリネット

(IV) 最終マーキングの集合は$\{M \mid (\sum_p M(p) = 1) \wedge (M(p_q) = 1 \Leftrightarrow q \in F)\}$である。

(V) μ：ラベル付関数は、

$$\mu(t_\alpha^q) = \begin{cases} \alpha : \alpha \in \Sigma_T \text{（添字を無視する）} \\ \lambda : \text{その他} \end{cases}$$ ，である。　（構成終）

B) $\exists L$ s.t. $L \in \mathcal{L}_{PN}^{\lambda} \wedge L \notin \mathcal{L}_1$ を示す。$L = \{a^i b^i a^i b^i \mid i \geq 1\}$とすれば証明できる。証明は、それほど自明ではないが、ここでは省略する。(2)　▨

文脈自由言語の族を$\mathcal{F}$とする.

《定理5》(i) $\mathcal{L}_1 \supsetneq \mathcal{R}$　(ii) $\mathcal{L}_1$ と $\mathcal{F}$ は比較不可能である.

(略証) (i) $\mathcal{L}_1 \supseteq \mathcal{R}$ は容易, $\mathcal{L}_1 \neq \mathcal{R}$ は例2より明らか.

(ii) $\mathcal{L}_1 \not\supseteq \mathcal{F}$ は, 定理4と文献(1)より明らか.

$\mathcal{F} \not\supseteq \mathcal{L}_1$ を $L=\{a^ib^ia^i \mid i\geq 1\}\in\mathcal{L}_1$ を示すことで示す.

$\mathcal{G}=(\mathcal{P},\mathcal{A})$, $\mathcal{P}=(\Sigma_N,\Sigma_T,S,P)$. $P=\{S\to Aa,$ $A\to Ba$, $B\to Sb\mid b\}$. $\mathcal{A}=(K,\Sigma,\delta,q_0,F)$ は図7の推移図によって与えられる. ▨

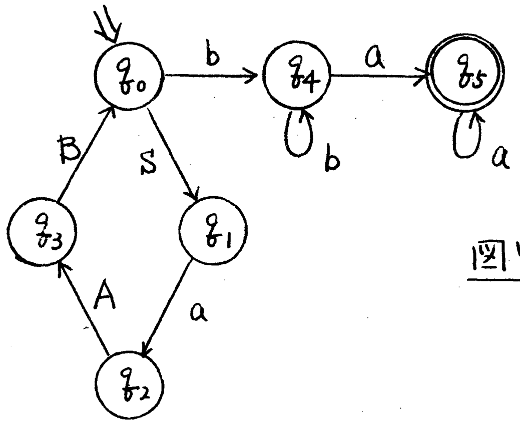


図7　推移図

以上から得られた結果を図8にまとめる.

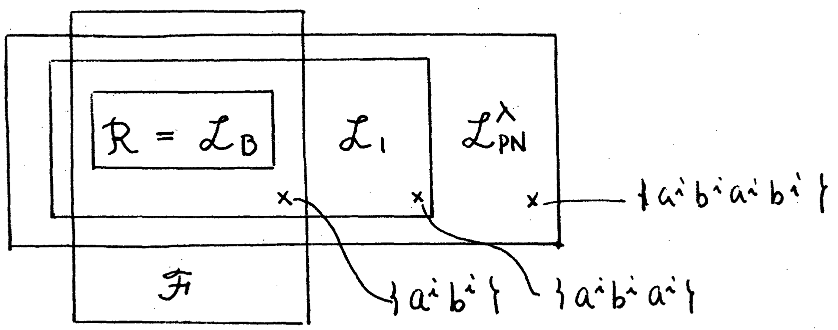


図8. $\mathcal{L}_1$, $\mathcal{L}_B$と他の言語族との関係

4. あとがき

並行処理システムのモデルSPSを定義し，SPS言語を用いて，SPSのいくつかの部分クラスの能力を考察した．ここでは，特に，左線形文法と有限オートマトンから定義されるSPSを中心に考察した．本稿に含められなかった多くの問題については機会を改めて報告する．

謝辞　御指導を賜る本学福村晃夫教授，三重大学大山口通夫助教授，日頃熱心に御討論を頂く，阿曽弘具講師をはじめとする福村・本多研究室の方々に感謝する．

（文献）

(1) M. Hack, "Petri net Languages", Computer Structure Group Memo 124 project MAC

(2) 山下，稲垣，本多，"同期付生成システム（SPS）－新しい並行処理モデル"信学会オートマトンと言語研究会，1980．3 発表予定