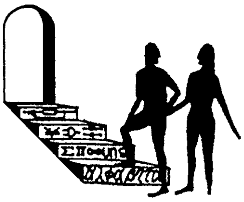
 システム/制御/情報，Vol. 56, No. 3, pp. 144–151, 2012

講　座



離散事象システムのスーパバイザ制御理論—I
—スーパバイザと可制御性

高井　重昌*

1.　はじめに

離散事象システムは，事象の生起により状態が離散的に遷移する動的システムの総称である．離散事象システムとして取り扱えるシステムの例とし，生産システム，交通システム，シーケンス制御システム，通信システム，データベースシステムなどがあげられる．スーパバイザ制御理論は，離散事象システムの論理的な仕様に対するフィードバック制御のための理論として，1980年代前半にRamadgeとWonhamによって提案された[1,2]．それからほぼ30年が経過した現在，理論的にはほぼ完成の段階にあるように思われる．海外では，1995年にスーパバイザ制御に関する本[3]が出版され，CassandrasとLafortuneによる離散事象システム全般に関する入門書[4]においても，スーパバイザ制御理論が詳しく紹介されている．一方国内では，本学会誌，および関連の他学会誌において，スーパバイザ制御に関する解説記事がいくつか掲載されているが[5–11]，筆者の知る限り，スーパバイザ制御全般について体系的にまとまった形での出版物はない．そこで，本講座では，基礎となる数学的事項に始まり，スーパバイザ制御理論の基本的結果から最近の成果までを，全6回の講座を通して体系立てて，しかも離散事象システムに関する特段の予備知識を必要とすることなく，解説することを目的としている．

第1回目である本稿では，まず，スーパバイザ制御理論の数学的基礎となるオートマトンと（形式）言語に関する基礎的事項を説明する．つぎに，スーパバイザ制御問題の定式化を行い，スーパバイザ制御のもとで与えられた制御仕様が満足されるために，仕様を表す言語が満たすべき可制御性の概念を定義する．そして，その可制御性の概念を用いて，制御仕様を満足させるスーパバイザが存在するための必要十分条件を示す．さらに，その条件を判定する方法についても述べる．以後，第2回目では，最大許容スーパバイザを構成するために必要となる最大可制御部分言語の存在性とその計算方法，第3回目では，事象の部分的な観測のもとでのスーパバイザ制御，第4回目では，複数のローカルスーパバイザによる分散スーパバイザ制御について解説する．第5回目では，制御下での事象の生起や禁止に伴うコストなど，定量的評価指標に関して最適なスーパバイザを構成する最適スーパバイザ制御について述べる．最後に第6回目では，発展的な話題として，近年コンピュータサイエンスの分野で注目されている余代数を用いたシステム表現に基づく，スーパバイザ制御の枠組みを紹介する．

* 大阪大学 大学院 工学研究科

Key Words: discrete event system, formal language, automaton, supervisory control, controllability.

2.　離散事象システムのモデル

2.1　言語

事象の生起順序に関する離散事象システムの論理的な振舞いは，その初期状態から生起する事象列で記述することができる．例として，1台のATMを利用する人がなす列において，つぎの二つの事象aとbを考える．

- a：ATMを利用しようとする人が列に到着
- b：ATMを利用した人が列から離れる

このATMにおいて，最初に到着した人がATMから離れる前に，別の人が到着し，その後最初に到着した人がATMを離れて，さらに二人到着した，といった状況は事象列$aabaa$で表すことができる．また，初期状態において，まだ事象が生起していない状況は空列εによって表現される．

一般に，システムで生起しうる事象列は一意ではない．そこで，システムの可能な振舞いを事象列の集合で表現する．いま，システムで生起する事象の集合をΣで表し，Σ^*は空列εを含む，Σの要素で構成されるすべての有限列の集合とする．そして，Σ^*の部分集合を言語という．つまり，言語とはΣの要素で構成される有限列の集合である．任意の言語$L_1,L_2\subseteq\Sigma^*$に対して，$L_1\subseteq L_2$が成り立つとき，L_1はL_2の部分言語であるという．

任意の事象列$s\in\Sigma^*$に対して，その長さを$|s|$で表す[1]．任意の事象列$s,t\in\Sigma^*$に対して，$s=tu$となる事象列$u\in\Sigma^*$が存在するならば，tはsの接頭語といい，sのすべての接頭語の集合を$\overline{s}$と書く．空列εに対して，$\varepsilon s=s\varepsilon=s$となるため，$\varepsilon,s\in\overline{s}$であることに注意されたい．たとえば，$\Sigma=\{a,b,c\}$上の事象列$s=baac\in\Sigma^*$に対して，$\overline{s}=\{\varepsilon,b,ba,baa,baac\}$である．任意の言語$L\subseteq\Sigma^*$に対して，その要素のすべての接頭語からなる集合を$\overline{L}$で表す．つまり，

$$\overline{L}=\bigcup_{s\in L}\overline{s}=\{t\in\Sigma^*\mid\exists u\in\Sigma^*:tu\in L\}$$

[1] 任意の有限集合Aに対しては，$|A|$はその要素数を表すとする．

である．$L=\overline{L}$ が成り立つとき，L は（接頭語に関して）閉じているという．すなわち，L の任意の要素 $s\in L$ に対して，そのすべての接頭語もまた L の要素であるとき，L は閉じている．$\overline{L}=\overline{\overline{L}}$ であるから，$\overline{L}$ は閉じた言語であることに注意されたい．なお，空でない任意の閉じた言語 $L\subseteq\Sigma^*$ に対しては，任意の要素 $s\in L$ に対して，$\varepsilon\in\overline{s}$ であるから，$\varepsilon\in L$ となる．上述の ATM の例では，$\Sigma=\{a,b\}$ であり，システムで生起しうる事象列の集合はつぎの言語 $L\subseteq\Sigma^*$ で表すことができる．

$$L=\{s\in\Sigma^*\,|\,\forall s'\in\overline{s}:\sharp(s',a)\geq\sharp(s',b)\}\tag{1}$$

ここで，任意の事象列 $s\in\Sigma^*$ に対して，$\sharp(s,a)$ と $\sharp(s,b)$ はそれぞれ，s に含まれる事象 a の数，b の数を示すとする．また，ATM の列に並ぶ人の数が 0 になるような事象列の集合を考えれば，それはつぎの部分言語 $L'\subseteq L$ で表すことができる．

$$L'=\{s\in L\,|\,\sharp(s,a)=\sharp(s,b)\}\tag{2}$$

(1) 式の言語 L は閉じているが，(2) 式の L' は閉じていない．

任意の言語 $L_1,L_2\subseteq\Sigma^*$ に対して，その連接 $L_1L_2\subseteq\Sigma^*$ を

$$L_1L_2=\{st\in\Sigma^*\,|\,s\in L_1,\ t\in L_2\}$$

と定義する．また，差集合 L_1-L_2 を

$$L_1-L_2=\{s\in\Sigma^*\,|\,s\in L_1,\ s\notin L_2\}$$

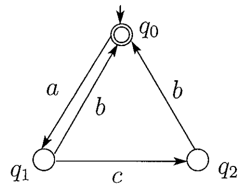
と定義する．

2.2　オートマトン

システムの振舞いを表現する言語を生成するモデルとして，本講座ではオートマトンを考える．オートマトンは 5 項組

$$G=(Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)\tag{3}$$

により定義される．ここで，Q は状態集合，Σ は前出の事象集合，$\delta:Q\times\Sigma\to Q$ は状態遷移関数，$q_0\in Q$ は初期状態，$Q_m\subseteq Q$ はマーク状態の集合である．通常，離散事象システムでは，与えられたタスクの終了などを表す状態をマーク状態として選ぶ．状態遷移関数は，各状態から事象の生起によりつぎに遷移する状態を定める．つまり，任意の状態 $q\in Q$ と任意の事象 $\sigma\in\Sigma$ に対して，$\delta(q,\sigma)=q'$ は q から σ の生起により q' に遷移することを表す．一般には，各状態においてすべての事象が生起可能であるとは限らない．たとえば，**2.1** の ATM の例において，列に並ぶ人の数を状態と考えれば，状態 0 では ATM を利用した人が列を離れるという事象 b は生起できない．q において σ が生起可能でない場合，$\delta(q,\sigma)$ は定義されないとする[1]．



第 1 図　オートマトン

【例題 1】　第 1 図に示されるオートマトンを考える．このオートマトンにおいて，$Q=\{q_0,q_1,q_2\}$，$\Sigma=\{a,b,c\}$，$Q_m=\{q_0\}$ であり，状態遷移関数 $\delta:Q\times\Sigma\to Q$ は

$$\delta(q,\sigma)=\begin{cases}q_0,\ \text{if}\ (q=q_1,\ \sigma=b)\ \text{or}\ (q=q_2,\ \sigma=b)\\ q_1,\ \text{if}\ q=q_0,\ \sigma=a\\ q_2,\ \text{if}\ q=q_1,\ \sigma=c\end{cases}$$

である．なお，初期状態は始点のない矢印で，マーク状態は二重丸で示している．

$\delta(q,\sigma)$ は，q から一つの事象 σ による状態遷移を定めているが，事象列による遷移を記述できれば便利である．そこで，状態遷移関数 δ の定義域を $Q\times\Sigma^*$ へ以下のように帰納的に拡張する．

- $\forall q\in Q:\ \delta(q,\varepsilon)=q$
- $\forall q\in Q,\ \forall s\in\Sigma^*,\ \forall\sigma\in\Sigma$:

$$\delta(q,s\sigma)=\begin{cases}\delta(\delta(q,s),\sigma),\ \text{if}\ \delta(q,s)!\\ \text{undefined},\quad\text{otherwise}\end{cases}$$

ここで，$\delta(q,s)!$ は状態 q から事象列 s による状態遷移が定義されることを表す．また，$\neg\delta(q,s)!$ は $\delta(q,s)!$ の否定を表す．

オートマトン G において，初期状態 q_0 から生起可能な事象列の集合は

$$L(G)=\{s\in\Sigma^*\,|\,\delta(q_0,s)!\}$$

と定義される．$L(G)\subseteq\Sigma^*$ は G の生成言語とよばれる．また，q_0 から Q_m の要素であるマーク状態へ遷移する事象列の集合は

$$L_m(G)=\{s\in L(G)\,|\,\delta(q_0,s)\in Q_m\}$$

と定義され，G のマーク言語とよばれる．通常，マーク状態は与えられたタスクの終了などを表すため，$L(G)$ は初期状態からの可能な振舞いを記述し，$L_m(G)$ はタスクの終了に対応する振舞いを記述する言語とみなすことができる．初期状態から生起可能な任意の事象列 $s\in L(G)$ に対して，そのすべての接頭語もまた初期状態から生起可能である．したがって，$L(G)$ は閉じた言語である．し

[1] 状態遷移関数 $\delta:Q\times\Sigma\to Q$ は，定義域 $Q\times\Sigma$ の要素 (q,σ) に対して値域 Q の要素が対応づけられない，つまり，$\delta(q,\sigma)$ は定義されない場合を許す部分関数である．

かし，$L_m(G)$ は一般には閉じていない．

（注意 1） コンピュータサイエンスの分野では，Q_m の要素は受理状態や最終状態とよばれ，$L_m(G)$ は受理言語とよばれる．本講座では，スーパバイザ制御理論における慣習に従い，Q_m の要素をマーク状態，$L_m(G)$ をマーク言語とよぶ．

オートマトン G の生成言語 $L(G)$ とマーク言語 $L_m(G)$ に対して，

$$L_m(G) \subseteq \overline{L_m(G)} \subseteq L(G)$$

が成立し，一般に等号は成立しない．

$$\overline{L_m(G)} = L(G)$$

が成り立つ場合，G はノンブロッキングであるという．これは，生起可能な任意の事象列 $s \in L(G)$ に対して，$st \in L_m(G)$ となる事象列 $t \in \Sigma^*$ が存在する，すなわち，タスクの終了を意味するマーク状態に到達可能であることを意味する．一方，G がノンブロッキングでない場合，$s' \in L(G) - \overline{L_m(G)}$ なる事象列 s' が生起すると，システムはマーク状態には到達できず，タスクが終了できないことになる．

任意の言語 $L \subseteq \Sigma^*$ に対して，$L_m(G) = L$ となるオートマトン G が存在する．状態集合 Q が有限集合の場合には，G は有限オートマトンとよばれる．そして，有限オートマトンによってマークされる言語を正規言語という．正規言語に対しては，それをマーク言語とする有限オートマトンを用いることで，様々な言語上の演算が可能となる．

（注意 2） 本講座では，離散事象システムのモデルとしてオートマトンを用いる．言語を生成するモデルとして，たとえばペトリネット [12] を用いることも可能である．

3. オートマトン上の演算

本章では，スーパバイザ制御理論において有用な，いくつかのオートマトン上の演算を紹介する．

3.1 Trim 演算

(3) 式のオートマトン G より，

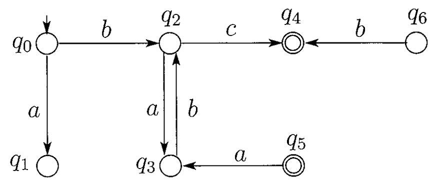
- 初期状態 q_0 から到達できない状態
- どのマーク状態にも到達できない状態

を取り除くことで得られるオートマトンを出力する Trim 演算を紹介する．

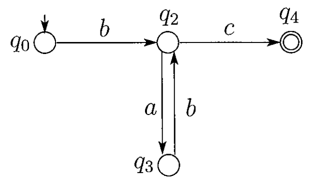
$$Q_{tr} = \{q \in Q \mid \exists s,t \in \Sigma^* : \delta(q_0,s) = q,\ \delta(q,t) \in Q_m\}$$

とおくと，G に対する Trim 演算は形式的に以下のように定義される．

$$Trim(G) = \begin{cases} (Q_{tr},\ \Sigma,\ \delta_{tr},\ q_0,\ Q_m \cap Q_{tr}), & \text{if } q_0 \in Q_{tr} \\ \Phi, & \text{otherwise} \end{cases} \quad (4)$$



第 2 図 オートマトン G



第 3 図 Trim 演算の結果 $Trim(G)$

ここで，状態遷移関数 $\delta_{tr} : Q_{tr} \times \Sigma \to Q_{tr}$ は，任意の $q \in Q_{tr}$ と任意の $\sigma \in \Sigma$ に対して，

$$\delta_{tr}(q,\sigma) = \begin{cases} \delta(q,\sigma), & \text{if } \delta(q,\sigma) \in Q_{tr} \\ \text{undefined}, & \text{otherwise} \end{cases}$$

と定義される．また，$q_0 \notin Q_{tr}$ は，q_0 から到達可能なマーク状態が存在しない，つまり $L_m(G) = \emptyset$ と等価である．この場合の Trim 演算の結果 Φ は，状態集合が空集合であり，$L(\Phi) = L_m(\Phi) = \emptyset$ を満足する空オートマトンである．

Trim 演算の定義より，$Trim(G)$ はノンブロッキングなオートマトンである．よって，ノンブロッキングでないオートマトン G に対して，$Trim(G) = G'$ とおけば，G' は $L_m(G) = L_m(G')$ を満足するノンブロッキングなオートマトンである．また，$Trim(G) = G$ を満足するオートマトン G は trim であるという．

【例題 2】 第 2 図に示されるオートマトン G に対して，Trim 演算を施した結果の $Trim(G)$ を第 3 図に示す．

3.2 補集合演算

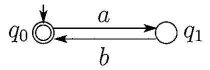
二つの言語 $L_1, L_2 \subseteq \Sigma^*$ に対して，包含関係 $L_1 \subseteq L_2$ は $L_1 \cap (\Sigma^* - L_2) = \emptyset$ と等価である．したがって，言語の包含関係を調べる際に，言語 $L \subseteq \Sigma^*$ に対してその補集合 $\Sigma^* - L$ を計算する補集合演算は有用である．

任意の言語 $L \subseteq \Sigma^*$ に対して，$L_m(G) = L$ なる trim なオートマトンを $G = (Q,\ \Sigma,\ \delta,\ q_0,\ Q_m)$ とする．このとき，$L_m(G^c) = \Sigma^* - L$ なるオートマトン G^c は以下のように構成される．

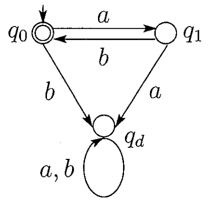
ステップ 1　G に対して，新しい状態 $q_d \notin Q$ を導入し，$\tilde{G} = (Q \cup \{q_d\},\ \Sigma,\ \tilde{\delta},\ q_0,\ Q_m)$ を構成する．ここで，$\tilde{\delta} : Q \cup \{q_d\} \times \Sigma \to Q \cup \{q_d\}$ は，任意の $q \in Q \cup \{q_d\}$ と任意の $\sigma \in \Sigma$ に対して，

$$\tilde{\delta}(q,\sigma) = \begin{cases} \delta(q,\sigma), & \text{if } q \in Q \text{ and } \delta(q,\sigma)! \\ q_d, & \text{otherwise} \end{cases}$$

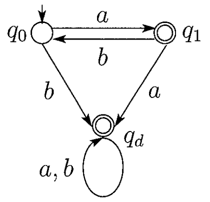
と定義される．



第 4 図　trim なオートマトン G



第 5 図　オートマトン $\tilde{G}$



第 6 図　オートマトン G^c

ステップ 2　$\tilde{G}$ において，マーク状態の集合 Q_m の補集合 $(Q \cup \{q_d\}) - Q_m$ をマーク状態の集合とし，$G^c = (Q \cup \{q_d\}, \Sigma, \tilde{\delta}, q_0, (Q \cup \{q_d\}) - Q_m)$ を構成する．

ステップ 1 で構成される $\tilde{G}$ は，G の状態遷移構造を保存したうえで，$Q \cup \{q_d\}$ に属するすべての状態において，Σ に属するすべての事象が生起可能になるように G に対して状態遷移を付加したものである．したがって，$L(\tilde{G}) = \Sigma^*$ と $L_m(\tilde{G}) = L_m(G) = L$ を満足する．さらに，ステップ 2 で，$\tilde{G}$ のマーク状態の集合 Q_m の補集合を新たにマーク状態の集合としたものが G^c であるから，$L(G^c) = \Sigma^*$ と $L_m(G^c) = \Sigma^* - L$ が成り立つ．

【例題 3】　$\Sigma = \{a,b\}$ とし，言語 $L = \{(ab)^k \in \Sigma^* \mid k \geq 0\}$ を考える．ここで，$(ab)^k$ は ab を k 回繰り返した事象列とする．たとえば，$(ab)^3 = ababab$ である．また $(ab)^0 = \varepsilon$ とする．このとき，$L_m(G) = L$ なる trim なオートマトン G を第 4 図に示す．そして，ステップ 1 で構成されるオートマトン $\tilde{G}$ と，ステップ 2 で構成されるオートマトン G^c をそれぞれ，第 5 図と第 6 図に示す．

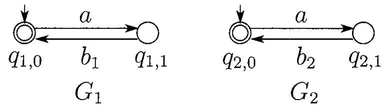
3.3　合成演算

制御対象のシステムのモデルを作成する際，システムを構成する各要素ごとのモデルを作成し，それらを合成することでシステム全体のモデルを得ることがよくある．ここでは，複数のオートマトンモデルを合成する演算として，同期合成を紹介する．n 個のオートマトン $G_i = (Q_i, \Sigma_i, \delta_i, q_{i,0}, Q_{i,m})$ $(i = 1,2,\ldots,n)$ の同期合成は

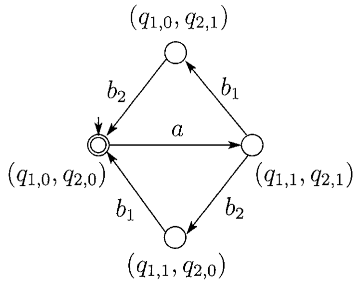
$$\|_{i=1}^n G_i = (Q, \Sigma, \delta, q_0, Q_m) \tag{5}$$

で定義される．ここで，

$$Q = Q_1 \times Q_2 \times \cdots \times Q_n$$



第 7 図　オートマトン G_1 と G_2



第 8 図　同期合成 $G_1 \| G_2$

$$\Sigma = \bigcup_{i=1}^n \Sigma_i$$
$$q_0 = (q_{1,0},\ q_{2,0}, \ldots, q_{n,0})$$
$$Q_m = Q_{1,m} \times Q_{2,m} \times \cdots \times Q_{n,m}$$

であり，状態遷移関数 $\delta : Q \times \Sigma \to Q$ はつぎのように定義される．任意の $q = (q_1, q_2, \ldots, q_n) \in Q$ と任意の $\sigma \in \Sigma$ に対して，$\delta(q,\sigma)!$ であるための条件は

$$\forall i \in \{1,2,\ldots,n\} : \sigma \in \Sigma_i \Rightarrow \delta_i(q_i,\sigma)!$$

である．$\delta(q,\sigma)!$ のとき，次状態 $\delta(q,\sigma) = (q'_1, q'_2, \ldots, q'_n)$ を

$$\forall i \in \{1,2,\ldots,n\} : q'_i = \begin{cases} \delta_i(q_i,\sigma), & \text{if } \sigma \in \Sigma_i \\ q_i, & \text{otherwise} \end{cases}$$

と定める．つまり，σ を事象集合 Σ_i に含むオートマトン G_i の各状態 q_i において σ が生起可能であることが，q において σ が生起可能であるための条件であり，σ が生起すると，各 q_i において同時に状態遷移を行う．このため，この合成手法は同期合成とよばれる．ここで，σ を事象集合に含まないオートマトン G_j は，σ の遷移に関与せず，その状態を変えないことに注意されたい．

【例題 4】　第 7 図に示す二つのオートマトン G_1 と G_2 を考える．G_1 の事象集合は $\Sigma_1 = \{a, b_1\}$，G_2 の事象集合は $\Sigma_2 = \{a, b_2\}$ である．G_1 と G_2 の同期合成 $G_1 \| G_2$ を第 8 図に示す．

すべての G_i の事象集合が等しい，つまり

$$\Sigma = \Sigma_1 = \Sigma_2 = \cdots = \Sigma_n \tag{6}$$

のとき，任意の $q = (q_1, q_2, \ldots, q_n) \in Q$ と任意の $\sigma \in \Sigma$ に対して，$\delta(q,\sigma)!$ であるための条件は

$$\forall i \in \{1,2,\ldots,n\} : \delta_i(q_i,\sigma)!$$

となり，すべての G_i の状態 q_i で σ が生起可能であることが要求され，σ が生起するとき，すべての G_i で状態

が遷移する．よって，(6) 式が成り立つ特別な場合には，

$$L(\|_{i=1}^{n} G_i) = \bigcap_{i=1}^{n} L(G_i)$$
$$L_m(\|_{i=1}^{n} G_i) = \bigcap_{i=1}^{n} L_m(G_i)$$

が成り立つ．つまり，同期合成により，各 G_i の生成言語 $L(G_i)$ およびマーク言語 $L_m(G_i)$ の交わりを得ることができる．

また，**5.** で後述するように，同期合成は対象システムとスーパバイザを合成した閉ループシステムのモデルとして用いることができる．

4. スーパバイザ制御問題

離散事象システムのスーパバイザ制御においては，事象の生起順序に関する論理的な制御仕様を取り扱い，そのような仕様を $L(G)$ もしくは $L_m(G)$ の部分言語 K で表現する．そして，スーパバイザの目的は，これまでに生起した事象列の観測情報に基づき，制御仕様を満足しなくなる事象の生起を禁止することで，システムで生成もしくはマークされる言語を K に制限することである．しかし，すべての事象がスーパバイザによって禁止できるとは限らない．そこで，システムで生起する事象の集合 Σ を二つの部分集合 Σ_c と Σ_{uc} に分割する．つまり，$\Sigma = \Sigma_c \cup \Sigma_{uc}$ かつ $\Sigma_c \cap \Sigma_{uc} = \emptyset$ である．ここで，Σ_c はスーパバイザによりその生起が禁止できる可制御事象の集合，Σ_{uc} はその生起が禁止できない不可制御事象の集合である．

形式的に，スーパバイザは関数 $S : L(G) \to \Gamma$ で表現できる．ここで，$\Gamma \subseteq 2^{\Sigma}$ は

$$\Gamma = \{\gamma \subseteq \Sigma \mid \Sigma_{uc} \subseteq \gamma\}$$

と定義され，各要素 $\gamma \in \Gamma$ は制御パターンとよばれる．各 $s \in L(G)$ に対して，$S(s) \in \Gamma$ は，スーパバイザが s の生起を観測した際に，つぎに生起を許容する事象の集合である．Γ の定義より，$\Sigma_{uc} \subseteq S(s)$ が成り立ち，これは不可制御事象の生起は常に許可され，決して禁止されることがないことを保証する．

スーパバイザ $S : L(G) \to \Gamma$ によって制御されるシステム G を S/G で表す．S/G によって生成される言語 $L(S/G)$ をつぎのように帰納的に定義する．

- $\varepsilon \in L(S/G)$
- $\forall s \in L(S/G), \forall \sigma \in \Sigma$:

$$s\sigma \in L(S/G) \Leftrightarrow [s\sigma \in L(G) \wedge \sigma \in S(s)]$$

ここで，$\sigma \in S(s)$ は，事象列 $s \in L(S/G)$ の生起後に，スーパバイザ S によって事象 σ の生起が許可されていることを意味する．S/G では，G で生起可能であり，かつ S で許可された事象のみが生起するため，

$$L(S/G) \subseteq L(G)$$

が成り立つ．さらに，$L(S/G)$ の定義より，$L(S/G)$ は空でない閉じた言語である．S/G によってマークされる言語 $L_m(S/G)$ を

$$L_m(S/G) = L(S/G) \cap L_m(G)$$

と定義する．一般に

$$L_m(S/G) \subseteq \overline{L_m(S/G)} \subseteq L(S/G)$$

が成り立つ．

$$\overline{L_m(S/G)} = L(S/G) \tag{7}$$

が成り立つとき，スーパバイザ S はノンブロッキングであるという．たとえ G がノンブロッキング，つまり $\overline{L_m(G)} = L(G)$ を満足したとしても，(7) 式が成り立つとは限らない．一般にはタスクの終了を意味するマーク状態へ到達することが望まれるため，通常，S はノンブロッキングであることが要求される．

スーパバイザ制御問題では，制御仕様を表す言語が G の生成言語 $L(G)$ の部分言語で与えられる場合と，マーク言語 $L_m(G)$ の部分言語で与えられる場合がある．前者はたとえば，デッドロック状態に陥らないといった安全性仕様のみを取り扱う場合に対応し，仕様として与えられた空でない言語 $K \subseteq L(G)$ に対して，

$$L(S/G) = K$$

となるスーパバイザ S を構成するのがスーパバイザ制御問題となる．後者はマーク状態への到達可能性を要求するライブ性も仕様に含める場合に対応し，仕様として与えられた空でない言語 $K \subseteq L_m(G)$ に対して，

$$L_m(S/G) = K$$

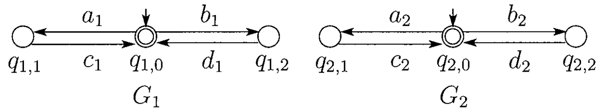
となるノンブロッキングなスーパバイザ S を構成することがスーパバイザ制御問題となる．

【例題 5】 二つのプロセス P_1 と P_2 に対する読出し・書込み問題を考える．プロセス P_1 と P_2 はそれぞれ，第 9 図に示すオートマトン G_1 と G_2 でモデル化されるとする．ここで，各事象の意味は以下の通りである．

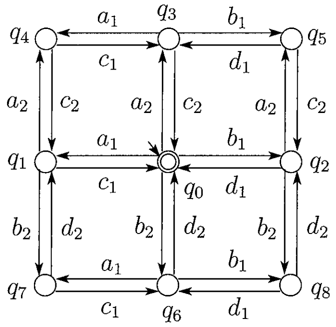
- a_i：プロセス P_i が読出し開始
- b_i：プロセス P_i が書込み開始
- c_i：プロセス P_i が読出し終了
- d_i：プロセス P_i が書込み終了

そして，全体システムのオートマトンモデル G を第 10 図に示す同期合成 $G_1 \| G_2$ で表現する．ここで，$G_1 \| G_2$ における各状態は以下の通りである．

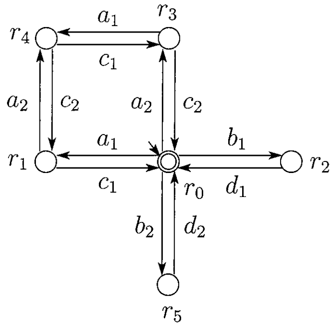
$$q_0 = (q_{1,0}, q_{2,0}) \quad q_1 = (q_{1,1}, q_{2,0}) \quad q_2 = (q_{1,2}, q_{2,0})$$
$$q_3 = (q_{1,0}, q_{2,1}) \quad q_4 = (q_{1,1}, q_{2,1}) \quad q_5 = (q_{1,2}, q_{2,1})$$
$$q_6 = (q_{1,0}, q_{2,2}) \quad q_7 = (q_{1,1}, q_{2,2}) \quad q_8 = (q_{1,2}, q_{2,2})$$



第 9 図　オートマトン G_1 と G_2



第 10 図　同期合成 $G_1 \| G_2$



第 11 図　制御仕様 G_K

このシステムにおいて，各プロセスの読出し開始と書込み開始のみが制御できる，すなわち $\Sigma_c=\{a_1,a_2,b_1,b_2\}$，$\Sigma_{uc}=\{c_1,c_2,d_1,d_2\}$ とする．そして，一つのプロセスの書込みが排他的に行われる，つまり，書込みの間，他のプロセスは読出しも書込みもできない，という制御仕様を考える．そのような制御仕様は，第 11 図に示すオートマトン G_K で表現でき，$K=L_m(G_K)$ とおく．この制御仕様に対して，マーク状態への到達可能性も要求すると，$L_m(S/G)=K$ となるノンブロッキングなスーパバイザ S を構成することがスーパバイザ制御問題となる．

5.　スーパバイザの存在条件

本章では，スーパバイザ制御理論において最も重要な概念である言語の可制御性 [2] を定義し，**4.** で定式化したスーパバイザ制御問題の解となるスーパバイザが存在するための必要十分条件を示す．そして，有限オートマトンを用いた，その条件の判定法を紹介する．

5.1　可制御性とスーパバイザの存在条件

まず，言語の可制御性 [2] を定義する．

【定義 1】　言語 $K\subseteq L(G)$ が

$$\overline{K}\Sigma_{uc}\cap L(G)\subseteq\overline{K} \tag{8}$$

を満足するとき，K は（$L(G)$ と Σ_{uc} に関して）可制御であるという．ここで，$\overline{K}\Sigma_{uc}$ は $\overline{K}$ と Σ_{uc} の連接である．

（注意 3）任意の言語 $K\subseteq L(G)$ に対して，$\overline{\overline{K}}=\overline{K}$ であるから，K の可制御性は $\overline{K}$ の可制御性と等価である．また，K が閉じた言語の場合，(8) 式は

$$K\Sigma_{uc}\cap L(G)\subseteq K$$

と等価である．

言語 $K\subseteq L(G)$ が可制御であるということは，任意の事象列 $s\in\overline{K}$ に対して，任意の不可制御事象 $\sigma\in\Sigma_{uc}$ が続いて生起した場合，$s\sigma$ が $\overline{K}$ の要素になることを意味する．このため，制御仕様が可制御であるならば，不可制御事象の生起が禁止できなくても，スーパバイザ制御により制御仕様を満足させることができることになる．

まず，制御仕様が G の生成言語 $L(G)$ の部分言語 $K\subseteq L(G)$ で与えられる場合において，スーパバイザ制御問題の解であるスーパバイザが存在するための必要十分条件 [2] を示す．なお，これはスーパバイザ制御理論の基礎となる結果であるため，その証明も示しておく．

【定理 1】　任意の空でない部分言語 $K\subseteq L(G)$ に対して，$L(S/G)=K$ なるスーパバイザ $S:L(G)\to\Gamma$ が存在するための必要十分条件は，K が閉じた可制御言語となることである．

（証明）まずは必要性を証明する．$L(S/G)=K$ なるスーパバイザ $S:L(G)\to\Gamma$ を考える．$L(S/G)$ は閉じた言語であるから，K も閉じている．K が可制御であることを示す．$s\sigma\in K\Sigma_{uc}\cap L(G)$ なる任意の $s\in K=L(S/G)$ と任意の $\sigma\in\Sigma_{uc}$ に対して，$s\sigma\in L(G)$ かつ $\sigma\in\Sigma_{uc}\subseteq S(s)$ であるため，$s\sigma\in L(S/G)=K$ が成り立つ．よって，K は可制御である．

つぎに十分性を証明する．任意の $s\in L(G)$ に対して，

$$S(s)=\Sigma_{uc}\cup\{\sigma\in\Sigma_c\mid s\sigma\in K\} \tag{9}$$

となる $S:L(G)\to\Gamma$ を考える．そして，$L(S/G)=K$ を示すため，任意の $s\in L(G)$ に対して，

$$s\in L(S/G)\Leftrightarrow s\in K \tag{10}$$

となることを，s の長さに関する帰納法により証明する．

$|s|=0$ の場合，K は空でない閉じた言語であるから，$s=\varepsilon$ に対して，$\varepsilon\in L(S/G)\cap K$ が成り立つ．

$|s|\leq n$ なる任意の $s\in L(G)$ に対して，(10) 式が成り立つと仮定する．$|s|=n+1$ なる任意の $s\in L(G)$ を考える．s は $s=s'\sigma$ と書くことができる．ここで，$|s'|=n$ かつ $\sigma\in\Sigma$ である．まず，$s'\sigma\in L(S/G)$ と仮定する．$L(S/G)$ の定義より，$s'\in L(S/G), \sigma\in S(s')$ かつ $s'\sigma\in L(G)$ である．$\sigma\in\Sigma_c$ ならば，(9) 式より，$s'\sigma\in K$ である．$\sigma\in\Sigma_{uc}$ ならば，帰納法の仮定から $s'\in K$ であるから，K の可制御性より，$s'\sigma\in K\Sigma_{uc}\cap L(G)\subseteq K$ である．逆に，$s'\sigma\in K\subseteq L(G)$ と仮定すると，$s'\in K$ であるから，帰

納法の仮定より，$s' \in L(S/G)$ である．さらに，(9) 式より，$\sigma \in S(s')$ となり，$s'\sigma \in L(S/G)$ である．

よって，帰納法により，任意の $s \in L(G)$ に対して，(10) 式が成り立つ． □

定理 1 の十分性の証明より，空でない閉じた可制御言語 $K \subseteq L(G)$ に対して，$L(S/G)=K$ なるスーパバイザ $S:L(G) \to 2^{\Sigma}$ は (9) 式で構成できる．

つぎに，制御仕様がマーク言語 $L_m(G)$ の部分言語 $K \subseteq L_m(G)$ で与えられる場合において，スーパバイザ制御問題の解であるスーパバイザが存在するための必要十分条件 [2] を示す．

【定義 2】 言語 $K \subseteq L_m(G)$ が

$$\overline{K} \cap L_m(G) = K \tag{11}$$

を満足するとき，K は $L_m(G)$ に関して閉じているという．

【定理 2】 任意の空でない部分言語 $K \subseteq L_m(G)$ に対して，$L_m(S/G)=K$ なるノンブロッキングなスーパバイザ $S:L(G) \to 2^{\Sigma}$ が存在するための必要十分条件は，K が可制御かつ $L_m(G)$ に関して閉じていることである．

空でない部分言語 $K \subseteq L_m(G)$ が可制御かつ $L_m(G)$ に関して閉じており，**定理 2** の条件が満足されるとする．K は空でない可制御言語であるから，$\overline{K}$ は空でない閉じた可制御言語となる．よって，**定理 1** より，$L(S/G)=\overline{K}$ なるスーパバイザ $S:L(G) \to 2^{\Sigma}$ が存在する．K は $L_m(G)$ に関して閉じているので，

$$L_m(S/G) = L(S/G) \cap L_m(G) = \overline{K} \cap L_m(G) = K$$

が成り立つ．しかも，

$$\overline{L_m(S/G)} = \overline{K} = L(S/G)$$

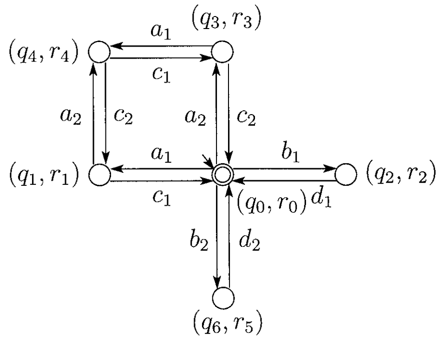
より，S はノンブロッキングである．つまり，**定理 2** において，K の可制御性は $L(S/G)=\overline{K}$ なるスーパバイザ $S:L(G) \to 2^{\Sigma}$ の存在性を保証し，K が $L_m(G)$ に関して閉じていることは，そのような S に対して，$L_m(S/G)=K$ が成り立つための条件である．

5.2 スーパバイザの存在条件の判定方法

定理 1，**定理 2** で示したスーパバイザの存在条件の判定方法について述べる．なお，G は有限オートマトンとする．まず，与えられた正規言語 $K \subseteq L(G)$ が可制御か否かを判定する方法を示す．$G_K=(R,\ \Sigma,\ \zeta,\ r_0,\ R_m)$ を $L_m(G_K)=K$ である trim な有限オートマトンとする．G_K は trim であるから，$L(G_K)=\overline{K}$ が成り立つ．G と G_K の同期合成を

$$G \parallel G_K = (Q \times R,\ \Sigma,\ \xi,\ (q_0,r_0),\ Q_m \times R_m)$$

とおくと，K が可制御でないための必要十分条件は，つぎの二条件を満足する $G \parallel G_K$ の状態 $(q,r) \in Q \times R$ が存在することである．



第 12 図 同期合成 $G \parallel G_K$

<u>条件 1</u> $\exists s \in L(G \parallel G_K) : \xi((q_0,r_0),s)=(q,r)$

<u>条件 2</u> $\exists \sigma \in \Sigma_{uc} : \delta(q,\sigma)! \wedge \neg\zeta(r,\sigma)!$

条件 1 は，状態 $(q,r) \in Q \times R$ がある事象列 $s \in L(G \parallel G_K) = L(G) \cap L(G_K) = L(G) \cap \overline{K} = \overline{K}$ により，初期状態 (q_0,r_0) から到達可能であることを意味する．条件 2 より，この事象列 $s \in \overline{K}$ に対して，$s\sigma \in L(G)$ かつ $s\sigma \notin L(G_K)=\overline{K}$ なる不可制御事象 $\sigma \in \Sigma_{uc}$ が存在する．つまり，$s\sigma \in \overline{K}\Sigma_{uc} \cap L(G)$ であるが，$s\sigma \notin \overline{K}$ なる $s \in \overline{K}$ と $\sigma \in \Sigma_{uc}$ が存在することになり，これは可制御性の条件に反する．G と G_K が有限オートマトンであるため，$Q \times R$ の要素数は有限であり，上述の方法により，K が可制御か否かを判定することができ，その計算量は $O(|Q| \cdot |R| \cdot |\Sigma_{uc}|)$ である．

【例題 6】 **例題 5** のシステムと制御仕様を考え，言語 $K \subseteq L_m(G)$ の可制御性を調べる．第 11 図の G_K は $L_m(G_K)=K$ なる trim な有限オートマトンである．第 10 図の G と G_K の同期合成 $G \parallel G_K$ を第 12 図に示す．$\Sigma_{uc}=\{c_1,\ c_2,\ d_1,\ d_2\}$ であるため，条件 1 と条件 2 を満足する $G \parallel G_K$ の状態は存在せず，K は可制御と判定できる．もし，$\Sigma_{uc}=\{a_1,\ a_2,\ c_1,\ c_2,\ d_1,\ d_2\}$ とすれば，第 12 図において，たとえば，状態 (q_6,r_5) は初期状態 (q_0,r_0) から到達可能であり，しかも不可制御事象 $a_1 \in \Sigma_{uc}$ に対して，$\delta(q_6,a_1)!$ かつ $\neg\zeta(r_5,a_1)!$ となるため，K は可制御ではない．

また，正規言語 $K \subseteq L_m(G)$ が $L_m(G)$ に関して閉じているか否かを判定するには，$K \subseteq \overline{K} \cap L_m(G)$ は常に成り立つため，$\overline{K} \cap L_m(G) \subseteq K$ か否かを調べれば十分である．これは補集合演算と同期合成を用いて判定可能である．

5.3 スーパバイザのオートマトン実現

4. では，スーパバイザ S を関数 $S:L(G) \to \Gamma$ として定義した．ここでは，スーパバイザ S のオートマトン実現について述べる．

制御仕様として与えられた言語 $K \subseteq L_m(G)$ は可制御かつ $L_m(G)$ に関して閉じているとする．このとき，任意の $s \in L(G)$ に対して，

$$S(s) = \Sigma_{uc} \cup \{\sigma \in \Sigma_c \mid s\sigma \in \overline{K}\}$$

であるスーパバイザ $S: L(G) \to \Gamma$ に対して，$L(S/G)=\overline{K}$ かつ $L_m(S/G)=K$ が成り立つ．つまり，任意の事象列 $s \in L(S/G)=\overline{K}$ に対して，つぎに生起可能な事象の集合が $\{\sigma \in \Sigma \mid s\sigma \in \overline{K}\}$ に制限される．このような制御動作は，同期合成を用いて表現することができる．$G'_K=(R, \Sigma, \zeta, r_0, R)$ を $L(G'_K)=L_m(G'_K)=\overline{K}$ であるオートマトンとすると，G と G'_K の同期合成 $G \parallel G'_K$ において，

$$\begin{aligned} L(G \parallel G'_K) &= L(G) \cap L(G'_K) \\ &= L(G) \cap \overline{K} \\ &= \overline{K} \\ &= L(S/G) \end{aligned}$$

かつ

$$\begin{aligned} L_m(G \parallel G'_K) &= L_m(G) \cap L_m(G'_K) \\ &= L_m(G) \cap \overline{K} \\ &= K \\ &= L_m(S/G) \end{aligned}$$

が成り立ち，G'_K との同期合成により，スーパバイザ S による制御動作と同様に，G の事象の生起が制限される．よって，G'_K はスーパバイザ S のオートマトン実現とみなすことができ，S によって制御されたシステム S/G は，同期合成 $G \parallel G'_K$ で表現できる．

6. おわりに

本稿ではまず，離散事象システムの論理的な振舞いを表現するモデルとしてオートマトンを紹介し，スーパバイザ制御問題を定式化した．そして，スーパバイザ制御における基本的概念である言語の可制御性の定義，可制御性に基づくスーパバイザの存在条件を示した．制御仕様として与えられた言語が可制御でない場合，その最大可制御部分言語を用いて，最大許容スーパバイザを構成することになる．最大可制御部分言語の計算については，次回の第2回目で詳しく解説する．

(2011年11月1日受付)

参考文献

[1] P. J. Ramadge and W. M. Wonham: Supervisory control of discrete-event processes; *Feedback Control of Linear and Nonlinear Systems, Lecture Notes in Control and Information Sciences*, No. 39 (A. V. Balakrishnan and M. Thoma Eds.), Springer-Verlag, pp. 202–214 (1982)

[2] P. J. Ramadge and W. M. Wonham: Supervisory control of a class of discrete event processes; *SIAM J. Contr. Optim.*, Vol. 25, No. 1, pp. 206–230 (1987)

[3] R. Kumar and V. K. Garg: *Modeling and Control of Logical Discrete Event Systems*, Kluwer Academic Publishers (1995)

[4] C. G. Cassandras and S. Lafortune: *Introduction to Discrete Event Systems, Second Edition*, Kluwer Academic Publishers (2008)

[5] 潮：離散事象システムにおける制御問題とスーパバイザ；システム／制御／情報，Vol. 34, No. 9, pp. 531–538 (1990)

[6] 増田：Lookahead 戦略によるスーパバイザ制御；システム／制御／情報，Vol. 42, No. 8, pp. 428–433 (1998)

[7] 高井：離散事象システムの分散スーパバイザ制御；計測と制御，Vol. 40, No. 12, pp. 927–931 (2001)

[8] 高井：時間付き離散事象システムのスーパバイザ制御；システム／制御／情報，Vol. 46, No. 3, pp. 138–143 (2002)

[9] 高井，潮： コンカレント離散事象システムのスーパバイザ制御；システム／制御／情報，Vol. 51, No. 2, pp. 96–101 (2007)

[10] 高井，鈴木：離散事象システム；計測と制御，Vol. 46, No. 4, pp. 248–254 (2007)

[11] 山崎：言語測度を用いた離散事象システムのスーパバイザ制御；システム／制御／情報，Vol. 51, No. 11, pp. 506–511 (2007)

[12] J. L. Peterson（市川，小林訳）：ペトリネット入門，共立出版 (1984)

著者略歴

たかい　しげまさ

高井　重昌 (正会員)

1966年6月26日生．1991年3月神戸大学大学院工学研究科システム工学専攻修士課程修了．1992年6月大阪大学工学部助手，1998年4月和歌山大学システム工学部講師，1999年10月同助教授，2004年10月京都工芸繊維大学工芸学部助教授，2007年4月同大学大学院工芸科学研究科准教授，2009年4月大阪大学大学院工学研究科教授となり現在に至る．離散事象システムの制御，診断の研究に従事．博士（工学）．電子情報通信学会，計測自動制御学会，IEEEの会員．