システムのモデル化

離散事象システム

高井重昌*・鈴木達也**

*京都工芸繊維大学大学院 工芸科学研究科　京都府京都市左京区松ヶ崎
**名古屋大学大学院 工学研究科　愛知県名古屋市千種区不老町
*Graduate School of Science and Technology, Kyoto Institute of Technology, Matsugasaki, Sakyo-ku, Kyoto, Japan
**Graduate School of Engineering, Nagoya University, Furo-cho, Chikusa-ku, Nagoya, Aichi, Japan
*E-mail: takai@kit.ac.jp
**E-mail: t_suzuki@nuem.nagoya-u.ac.jp

キーワード：離散事象システム（discrete event system），オートマトン（automaton），形式言語（formal language），スーパバイザ制御（supervisory control），故障診断（fault diagnosis）．
JL 0004/07/4604-0248 ©2007 SICE

1. はじめに

事象の生起により，状態が離散的に遷移するようなシステムは離散事象システム (discrete event system) と呼ばれる．オペレーションシステム，データベースシステム，通信システム，生産システム，シーケンス制御システムなどは離散事象システムとみなすことができる好例である．もちろんこのようなシステムの挙動のすべてが離散的かつ事象駆動的な側面のみで記述できるわけではない．しかしながら，システムの検証や制御，診断，といった問題に目を向けた場合，離散的かつ事象駆動的なシステムモデルは，時として詳細な連続系のモデルよりも取り扱いやすく，結果として有用な解法を与えてくれる場合がよくある．

離散事象システムに対するモデル化手法の中でも特に理論体系が整備されている枠組みとして，形式言語理論をベースとしたアプローチが挙げられる[1),2)]．このアプローチの目的は，個別の対象にとらわれず，離散事象システム固有の特性を明らかにすることであるが，このアプローチでは連続系のシステム理論との対比も強く意識されており，連続系と同様なシステムの解析，制御器設計手法なども提案されている点で興味深い．

そこで本稿では，まず，この形式言語理論をベースとした離散事象システムのモデリング，制御問題[3),4)]について紹介する．つぎに離散事象システムモデルの適用先として有力と思われる故障診断問題[5),6)]について代表的なアプローチを紹介する．形式言語理論になじみのない読者にとっては少々とっつきにくいと思われる部分があるかもしれないが，形式言語理論の入門的な教科書を参照しながらお読みいただくとより理解が深まると思う．

2. 形式言語に基づく離散事象システムのモデリング

本章では，形式言語 (formal language) およびオートマトン (automaton) による離散事象システムのモデリングについて述べる．

2.1 言語

離散事象システムの振舞いは，システムにおいて生起した事象列によって記述できる．たとえば，加工機械の動作について，つぎの 4 つの事象 σ_i $(i = 1, 2, 3, 4)$ を考える．

- σ_1：作業開始，σ_2：作業終了，σ_3：作業中に故障発生，σ_4：故障状態から復旧

このとき，2 つのパーツを加工した後に故障，そして復旧後に 1 つのパーツを加工したという振舞いは事象列 $\sigma_1\sigma_2\sigma_1\sigma_2\sigma_1\sigma_3\sigma_4\sigma_1\sigma_2$ で表現できる．また，初期状態において，システムがまだ動作を開始していない状況は空列と呼ばれる記号 ε で表現される．

一般には，システムの可能な動作は一意ではなく，可能な動作をすべて記述するためには，事象列の集合を考える必要がある．事象の有限集合を Σ としたとき，Σ の要素の有限列からなる集合を（Σ 上の）言語 (language) という．たとえば，$\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$ としたとき，$L_1 = \{\varepsilon, \sigma_1, \sigma_1\sigma_2\sigma_3\}$，$L_2 = \{\sigma_2$ ではじまる有限事象列 $\}$ は言語の例である．Σ^* は空列 ε を含んだ，Σ の要素からなるすべての有限列の集合とする．つまり，言語は Σ^* の部分集合となる．本稿では以下，Σ の要素からなる事象列として，有限なもののみを考える．

事象列 $s, t \in \Sigma^*$ に対して，$tu = s$ なる $u \in \Sigma^*$ が存在するとき，t は s の接頭語 (prefix) であるという．空列 ε は，任意の $s \in \Sigma^*$ に対して $\varepsilon s = s\varepsilon = s$ を満足するとする．よって，ε と s はともに s の接頭語である．s のすべての接頭語の集合を $\overline{s}$ と書く．たとえば，$s = \sigma_1\sigma_2\sigma_1$ に対して，$\overline{s} = \{\varepsilon, \sigma_1, \sigma_1\sigma_2, \sigma_1\sigma_2\sigma_1\}$ である．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 $\overline{L} \subseteq \Sigma^*$ と書く．すなわち，

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

とする．たとえば，$L = \{\sigma_1\sigma_2\sigma_1, \sigma_2\sigma_1\sigma_2\}$ に対して，$\overline{L} = \{\varepsilon, \sigma_1, \sigma_2, \sigma_1\sigma_2, \sigma_2\sigma_1, \sigma_1\sigma_2\sigma_1, \sigma_2\sigma_1\sigma_2\}$ 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 (prefix-cloed) という．言語 $L = \{\sigma_1\sigma_2\sigma_1, \sigma_2\sigma_1\sigma_2\}$ に対しては，

$L \neq \overline{L}$ であるから L は接頭語に関して閉じていない．また，$L' = \{\varepsilon, \sigma_1, \sigma_2, \sigma_1\sigma_2, \sigma_2\sigma_1\}$ に対しては，$L' = \overline{L'}$ となるので，L' は接頭語に関して閉じている．

対象である離散事象システムを G と書き，G で生起しうるすべての事象列の集合を $L(G) \subseteq \Sigma^*$ と書く．$L(G)$ は G の生成言語 (generated language) と呼ばれ，必ず接頭語に関して閉じている．また，システムの初期状態において事象が何も生起していないという状況が存在するため，$\varepsilon \in L(G)$ とする．よって，$L(G)$ は空ではない．$L(G)$ の要素のうち，タスクの終了などを表わす事象列の集合を $L_m(G) \subseteq L(G)$ で表わす．$L_m(G)$ は G の受理言語 (accepted language) と呼ばれる．システム G の振舞いはこれら 2 つの言語 $(L(G), L_m(G))$ で記述される．

2.2 オートマトン

システムの振舞いは 2 つの言語 $(L(G), L_m(G))$ で記述できるが，それらは一般に無限集合となる．言語を表現し，言語上の演算を可能とするモデルとしてオートマトンがある．オートマトンは 5 項組

$$G = (Q, \Sigma, \delta, q_0, Q_m)$$

で定義される．ここで，Q は状態の集合，$\delta : Q \times \Sigma \to Q$ は状態遷移関数，$q_0 \in Q$ は初期状態，$Q_m \subseteq Q$ は受理状態の集合である．各受理状態 $q \in Q_m$ はタスクの終了などに対応する状態を表わすのに用いられる．状態遷移関数において，$\delta(q, \sigma) = q'$ は状態 $q \in Q$ から事象 $\sigma \in \Sigma$ の生起により，状態が $q' \in Q$ に遷移することを示している．もし，q において σ が生起できない，つまり σ による状態遷移がない場合，$\delta(q, \sigma)$ は定義されないとする．

たとえば，**図 1** は加工機械の簡単なオートマトンモデルを示している．ここで，状態の集合は $Q = \{q_0, q_1, q_2\}$，事象の集合は $\Sigma = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}$，初期状態はそれへ向かった短い矢印で表現，受理状態は二重丸で表現しており，この例では q_0 だけが受理状態である．状態遷移関数は

$$\delta(q_0, \sigma_1) = q_1 \ , \ \ \delta(q_1, \sigma_2) = q_0$$
$$\delta(q_1, \sigma_3) = q_2 \ , \ \ \delta(q_2, \sigma_4) = q_0$$

であり，これら以外の状態遷移 $\delta(q, \sigma)$ は定義されない．このオートマトンモデルでは，状態 q_0 は休止中，q_1 は動作中，q_2 は故障中を意味し，事象 σ_1 は作業開始，σ_2 は作業終了，σ_3 は作業中に故障発生，σ_4 は故障状態から復旧を意味している．

状態遷移関数 δ はつぎのように $\delta : Q \times \Sigma^* \to Q$ と定義域を拡張できる．

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

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

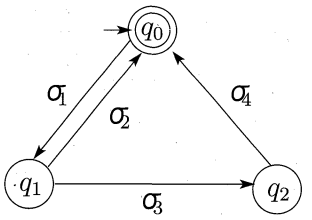

図 1　加工機械のオートマトン
状態 q_0 は休止中，q_1 は動作中，q_2 は故障中を意味し，事象 σ_1 は作業開始，σ_2 は作業終了，σ_3 は作業中に故障発生，σ_4 は故障状態から復旧を意味する．

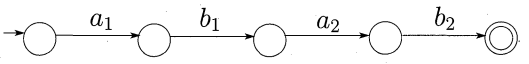

図 2　オートマトン
生成言語は $L(G) = \{\varepsilon, a_1, a_1b_1, a_1b_1a_2, a_1b_1a_2b_2\}$，受理言語は $L_m(G) = \{a_1b_1a_2b_2\}$ となる．

状態 $q \in Q$ と事象列 $s \in \Sigma^*$ に対して，$\delta(q, s)$ は q から s による状態遷移を表わし，$\delta(q, s)!$ は $\delta(q, s)$ が定義されている，すなわち q から s による状態遷移が可能なことを表わすとする．すると G の生成言語 $L(G)$ は

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

と定義される．また受理言語 $L_m(G)$ は

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

と定義される．たとえば，**図 2** で示されるオートマトン G に対して，その生成言語 $L(G)$ と受理言語 $L_m(G)$ はそれぞれ，$L(G) = \{\varepsilon, a_1, a_1b_1, a_1b_1a_2, a_1b_1a_2b_2\}$，$L_m(G) = \{a_1b_1a_2b_2\}$ となる．

3. 形式言語理論に基づく離散事象システムの制御

本章では，Ramadge と Wonham によって提案された離散事象システムの制御手法の 1 つであるスーパバイザ制御 (supervisory control)[3] を紹介する．スーパバイザ制御とは，対象とするシステムの振舞いを制限するための制御手法である．たとえば，3 種類のタスク a, b, c が実行可能なシステムを考える．**図 3** はそのオートマトンモデルである．そして，システムにおいて，まずタスク a を実行し，つぎにタスク b もしくは c を実行し，以後これを繰り返すという制御仕様を考える．この制御仕様は**図 4** のオートマトンで表現できる．この制御仕様のため，まずタスク b と c の実行を禁止し，つぎにタスク a の実行を禁止する必要がある(注1)．このように，一部の事象の生起を禁止し，システムの可能な動作が制御仕様と一致するようにシステムの振舞いを制限することがスーパバイザ制御の役割である．

(注1) スーパバイザ制御では，制御仕様で許されたタスク b, c 両方の実行を許可するが，そのうちどちらを実行すべきかは決定しない．

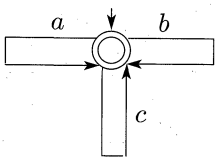

図3　対象システム
3 種類のタスク a, b, c が実行可能なシステムを表わす.

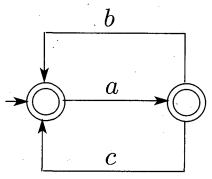

図4　制御仕様
まずタスク a を実行し，つぎにタスク b もしくは c を実行し，以後これを繰り返すという制御仕様を表わす.

スーパバイザ制御では，事象の集合 Σ を，その生起を禁止できる可制御事象 (controllable event) の集合 Σ_c と，その生起が禁止できない不可制御事象 (uncontrollable event) の集合 Σ_{uc} とに分割できるとする．ここで，$\Sigma = \Sigma_c \cup \Sigma_{uc}$, $\Sigma_c \cap \Sigma_{uc} = \emptyset$ である．不可制御事象として，制御器からのコマンドによらず，対象システムで自発的に生起する事象や，高いプライオリティをもつ事象などが考えられる．たとえば，図1の加工機械のオートマトンモデルにおいては，作業終了を表わす σ_2，故障発生を表わす σ_3 は対象システムで自発的に生起する事象であり，不可制御事象とみなすことができる．

図5 の概念図のように，スーパバイザ (supervisor) は，対象システム G で生起した事象を観測し，これまでに観測した事象列に対して，どの事象を禁止すべきかを決定する．ここでは簡単のため，スーパバイザはすべての事象の生起が観測できると仮定する．このとき，スーパバイザは関数 $f : L(G) \to 2^{\Sigma_c}$ で記述できる．ここで，2^{Σ_c} は Σ_c のべき集合，すなわち Σ_c のすべての部分集合を要素としてもつ集合である．これはつぎのように解釈する．いまシステムにおいて事象列 $s \in L(G)$ が生起したとき，スーパバイザは $f(s)$ に属する事象をすべて禁止する．そして，$s \in L(G)$ に続いて生起できる事象は，スーパバイザによって禁止されていない事象（つまり $\Sigma - f(s)$ の要素）に限られる．

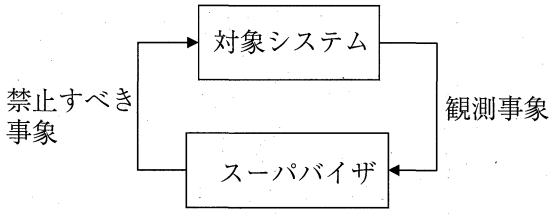

図5　スーパバイザ制御の概念図
スーパバイザは，対象システムで生起した事象を観測し，これまでに観測した事象列に対して，どの事象を禁止すべきかを決定する.

スーパバイザ $f : L(G) \to 2^{\Sigma_c}$ の制御動作のもとでシステムによって生成される言語を $L(G, f)$ と書く．$L(G, f)$ はつぎのように帰納的に定義される．

- $\varepsilon \in L(G, f)$
- $(\forall s \in L(G, f), \forall \sigma \in \Sigma)$

$$s\sigma \in L(G, f) \Leftrightarrow [s\sigma \in L(G)] \wedge [\sigma \notin f(s)]$$

条件 $\sigma \notin f(s)$ は事象 σ が事象列 s の生起後に f によって禁止されていないことを意味する．この条件が加わるため，$L(G, f) \subseteq L(G)$ となり，システムで生成される言語が $L(G)$ から $L(G, f)$ へ制限されることになる．

また，f の制御動作のもとでシステムによって受理される言語を $L_m(G, f)$ と書く．$L_m(G, f)$ は

$$L_m(G, f) = L(G, f) \cap L_m(G)$$

と定義する．一般に $L_m(G, f) \subseteq \overline{L_m(G, f)} \subseteq L(G, f)$ となる．次式が成り立つとき，f はノンブロッキング (nonblocking) であるという．

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

f がノンブロッキングでないならば，$s \notin \overline{L_m(G, f)}$ なる $s \in L(G, f)$ が存在する．そのような s が生起した場合は，システムは受理状態に到達できず，これはタスクを終了することができないことを意味する．よって，通常スーパバイザ f はノンブロッキングであることが要求される．

与えられた言語がスーパバイザの制御動作のもとで生成，受理されるにはどのような性質が必要であるかについて述べる．まずスーパバイザ制御において最も基本的な性質である言語の可制御性 (controllability) を定義する．

定義 1 言語 $K \subseteq L(G)$ が

$$\overline{K}\Sigma_{uc} \cap L(G) \subseteq \overline{K} \qquad (1)$$

を満足するとき，($L(G)$ と Σ_{uc} に関して) 可制御であるという．ここで，$\overline{K}\Sigma_{uc} = \{s\sigma |\ s \in \overline{K},\ \sigma \in \Sigma_{uc}\}$ である．

可制御性は言語 K の性質であり，$L(G)$ と Σ_{uc} に関して定義される．$K \subseteq L(G)$ 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が続いて生起したとしても，$\overline{K}$ の要素になることを意味する．

例 1 事象の集合が $\Sigma = \{a, b, c\}$ であるシステム G を考える．G で生成される言語は $L(G) = \{\varepsilon, a, ab, ac\}$ とする．$K = \{ab\}$ なる言語 $K \subseteq L(G)$ に対して，$\Sigma_c = \{b, c\}$, $\Sigma_{uc} = \{a\}$ とすると，$\overline{K}\Sigma_{uc} \cap L(G) = \{a\} \subseteq \{\varepsilon, a, ab\} = \overline{K}$ より，K は可制御である．一方，$\Sigma_c = \{a\}, \Sigma_{uc} = \{b, c\}$ とすると，$\overline{K}\Sigma_{uc} \cap L(G) = \{ab, ac\}$ となり，(1) 式が満足されない．よって K は可制御ではない．

まず，スーパバイザ f によって生成される言語 $L(G,f)$ に関して，つぎの定理が成り立つ．

定理 1 空でない言語 $K \subseteq L(G)$ に対して，$L(G,f) = \overline{K}$ となるスーパバイザ $f : L(G) \to 2^{\Sigma_c}$ が存在するための必要十分条件は K が可制御となることである．

空でない可制御言語 $K \subseteq L(G)$ に対して，$L(G,f) = \overline{K}$ なるスーパバイザ $f : L(G) \to 2^{\Sigma_c}$ は

$$f(s) = \Sigma_c - \{\sigma \in \Sigma_c |\ s\sigma \in \overline{K}\} \qquad (2)$$

と構成できる．言語 $K \subseteq L(G)$ を与えられた制御仕様と考えると，(2) 式は，各事象列 $s \in L(G)$ に対して，$s\sigma \in \overline{K}$ となり制御仕様に含まれる事象 σ 以外はすべて禁止することを意味する．

例 2 例 1 のシステムにおいて，$\Sigma_c = \{b,c\}$, $\Sigma_{uc} = \{a\}$ である場合を考える．このとき，言語 $K = \{ab\} \subseteq L(G)$ は可制御であるから，定理 1 より，$L(G,f) = \overline{K}$ なるスーパバイザ $f : L(G) \to 2^{\Sigma_c}$ が存在し，このような f は (2) 式より，

$$f(s) = \begin{cases} \{c\}, & \text{if } s = a \\ \{b,c\}, & \text{otherwise} \end{cases}$$

と得られる．

つぎに，スーパバイザ f の制御動作のもとでの受理言語 $L_m(G,f)$ に関する結果を述べる．言語 $K \subseteq L(G)$ が

$$\overline{K} \cap L_m(G) = K \qquad (3)$$

を満足するとき，$L_m(G)$ に関して閉じている ($L_m(G)$-closed) という．そして，つぎの定理が成り立つ．

定理 2 空でない言語 $K \subseteq L_m(G)$ に対して，$L_m(G,f) = K$ なるノンブロッキングなスーパバイザ $f : L(G) \to 2^{\Sigma_c}$ 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．

例 3 例 1 のシステムにおいて，$\Sigma_c = \{b,c\}$, $\Sigma_{uc} = \{a\}$ である場合を考える．$L_m(G) = \{ab, ac\}$ とすると，言語 $K = \{ab\} \subseteq L_m(G)$ は，

$$\overline{K} \cap L_m(G) = \{\varepsilon, a, ab\} \cap \{ab, ac\} = \{ab\} = K$$

より，$L_m(G)$ に関して閉じている．K は可制御であるから，$L(G,f) = \overline{K}$ なるスーパバイザ $f : L(G) \to 2^{\Sigma_c}$ が存在し，このような f に対して，

$$L_m(G,f) = L(G,f) \cap L_m(G) = \overline{K} \cap L_m(G) = K$$

$$\overline{L_m(G,f)} = \overline{K} = L(G,f)$$

が成り立つ．

関数 $f : L(G) \to 2^{\Sigma_c}$ で定義されたスーパバイザはオートマトンで実現することができ，対象システムと同期して状態遷移することで制御を行う．詳しくは文献 1), 2) を参照されたい．

4. 離散事象システムの故障診断

離散事象システムの故障診断 (fault diagnosis) に関しては様々なアプローチがあるが，大きく分けて確定的なモデルに基づく手法と確率的なモデルに基づく手法とに大別できる．故障診断の目的は，観測された事象列からその背後に潜む故障（正常な場合も含めて）を検出，特定することである．確定的なモデルの代表例としては，2,3 章で紹介した形式言語理論に基づいたアプローチが挙げられる[5),7)]が，本章では以下，まず形式言語理論に基づいた確定的なアプローチの基本的な概念を紹介し，つぎに事象間の生起時間間隔の確率モデルに基づいた手法を例を交えて紹介する．

今，診断する対象が 2.2 節で示された 5 項組のオートマトン G で与えられるとし，また G によって生成される言語を単に L と表記する．ここで，事象の集合 Σ を観測可能な事象の集合 Σ_o と観測できない事象の集合 Σ_{uo} とに分割，すなわち，$\Sigma = \Sigma_o \cup \Sigma_{uo}$ とする．また，問題となる故障事象の集合 Σ_f は通常観測できないことから（観測できれば自明な問題となるため），$\Sigma_f \subset \Sigma_{uo}$ となる．つぎに故障の識別可能性を考えるために，故障事象の集合 Σ_f をつぎのように分割する．

$$\Sigma_f = \Sigma_{f1} \cup \Sigma_{f2} \cup \cdots \cup \Sigma_{fm} \qquad (4)$$

Σ_{fi} は故障のタイプを表わしており，故障診断と言った場合，通常どのタイプの故障が生起したかを推定することを考える．またこのときの分割を，故障タイプのインデックスからなる集合 Π_f で定義する．以下では，可診断性を定式化するためにいくつかの表記を導入する．まず，ある事象列から可観測な事象のみからなる系列を抽出する写像 $P : \Sigma^* \to \Sigma_o^*$ を定義する．言い換えるとこの写像 P は，ある事象列から不可観測な事象を取り去ることに相当する．つぎにある事象列 y に対して，P の L に関する逆写像を次式で定義する．

$$P_L^{-1}(y) = \{s \in L | P(s) = y\} \qquad (5)$$

これは，不可観測な事象を取り除いたあとで y という事象列になるすべてのもとの事象列 $s \in L$ を与える写像となる．また，最後に生起する事象がタイプ i に属する故障事象であるようなすべての事象列の集合 $\Psi(\Sigma_{fi})$ を次式のように定義する．

$$\Psi(\Sigma_{fi}) = \{s\sigma_f \in L | \sigma_f \in \Sigma_{fi}\} \qquad (6)$$

最後に，L の中で事象列 s に引き続いて生起可能な事象列 t の集合 L/s を次式で定義する．

$$L/s = \{t \in \Sigma^* | st \in L\} \qquad (7)$$

これらの表記を用いて診断可能となるための条件を次式で定義する．

$$\omega \in P_L^{-1}[P(st)] \Rightarrow \Sigma_{fi} \in \omega$$
$$(\forall i \in \Pi_f), (\forall s \in \Psi(\Sigma_{fi})), (\forall t \in L/s) \qquad (8)$$

ただし，$\Sigma_{fi} \in \omega$ は Σ_{fi} の要素が ω に含まれていることを表わす．この条件は，st に対して不可観測な事象を取り去ってできた事象列 $P(st)$ と同じ事象列を与えるすべてのもとの事象列が Σ_{fi} に含まれる故障事象を含んでいることを意味する．この条件を D で記すと，可診断性 (diagnosability) は以下のように定義される．

$$\| t \| \geq n_i \Rightarrow D$$
$$(\forall i \in \Pi_f), (\exists n_i \in \mathcal{N}), (\forall s \in \Psi(\Sigma_{fi})), (\forall t \in L/s) \qquad (9)$$

ただし，$\| t \|$ は事象列 t の長さを表わす．したがってこの可診断性は，タイプ i の故障発生後，n_i 回の事象の生起の後に故障のタイプを特定できることを意味する．この可診断性は同じシステムであっても Π_f の決め方によって異なった結果をもたらす．

例 4 **図 6** において，$\alpha, \beta, \gamma, \delta$ は可観測な事象であり，σ_{uo} は不可観測な事象を表わす．また，$\sigma_{f1}, \sigma_{f2}, \sigma_{f3}$ は故障事象を表わす．ここで故障タイプとして，$\Sigma_{f1} = \{\sigma_{f1}, \sigma_{f2}\}$, $\Sigma_{f2} = \{\sigma_{f3}\}$ と定義すると図 6 に示すシステムは可診断となる．これは故障事象 σ_{f1} と σ_{f2} を区別する必要がないためである．この場合，たとえばタイプ 1 の故障 $(i = 1)$ に対しては，(8) 式中の $P(st)$ の 1 つとして $\alpha\beta\gamma$ が考えられるが，$P_L^{-1}(\alpha\beta\gamma) = \{\alpha\beta\sigma_{f1}\sigma_{f2}\gamma, \alpha\beta\sigma_{f1}\sigma_{uo}\gamma\}$ となり，どちらの事象列も Σ_{f1} に属する故障 σ_{f1} を含むため可診断となる．一方，区別したい故障タイプを $\Sigma_{f1} = \{\sigma_{f1}\}$, $\Sigma_{f2} = \{\sigma_{f2}\}$, $\Sigma_{f2} = \{\sigma_{f3}\}$ として定義すると可診断とはならない．これは，たとえ γ が観測されたとしても故障 σ_{f2} の生起を特定することができないためである（σ_{uo} と区別ができない）．

また，文献 5) では可診断となるための必要十分条件も示されている．

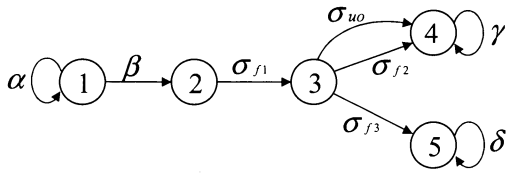

図 6　分割によって可診断性が変わる例

$\alpha, \beta, \gamma, \delta$ は可観測な事象であり，σ_{uo} は不可観測な事象を表わす．また，$\sigma_{f1}, \sigma_{f2}, \sigma_{f3}$ は故障事象を表わす．このモデルでは，σ_{f1} と σ_{f2} を異なる故障タイプとして区別することはできない．

つぎに診断器の構成方法であるが，制御器と同様に診断器も対象システムと同期して状態遷移を行いながら，故障の診断を行う．診断器の状態は対象システムの状態の中で可観測な事象の生起によって遷移される状態の集合 Q_o と故障タイプのラベルからなる集合を組み合わせた集合のべき集合で構成される．したがって，診断器の状態には対象システムがとりうる状態と故障タイプの情報が含まれることになる．ここでは診断器の状態の具体的作成法や状態遷移則の導出[5] は紙面の都合上省略するが，対象システムと診断器の一例を以下に示す．

例 5 **図 7** に示す対象システムにおいて，$\alpha, \beta, \gamma, \delta, \sigma$ は可観測な事象であり，σ_{uo} は不可観測な事象を表わす．また，$\sigma_{f1}, \sigma_{f2}, \sigma_{f3}$ は故障事象を表わし，$\Sigma_{f1} = \{\sigma_{f1}\}$, $\Sigma_{f2} = \{\sigma_{f2}, \sigma_{f3}\}$ と定義する．このシステムに対して**図 8** に示すような診断器が得られるが，この診断器の各状態中に記載されている 4N や 9F1 等の記号は，数字が対象システムの状態を，F♯ は故障のタイプを表わす．ただし，N は正常を表わし，A は生起を特定できない故障タイプが存在

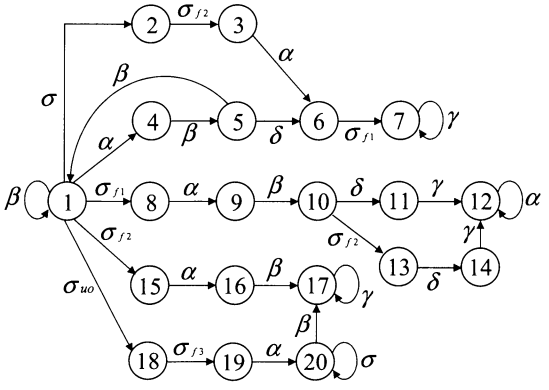

図 7　対象システム

$\alpha, \beta, \gamma, \delta, \sigma$ は可観測な事象であり，σ_{uo} は不可観測な事象を表わす．また，σ_{f1}, σ_{f2}, σ_{f3} が故障事象を表わす．

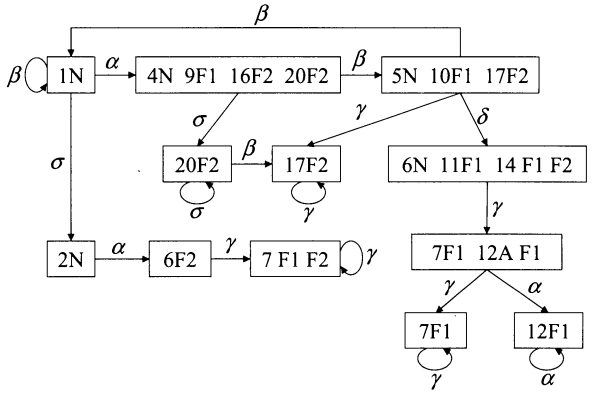

図 8　図 7 の対象システムに対する診断器

診断器は対象システムと同期して状態遷移を行う．診断器の各状態中に記載されている **4N** や **9F1** 等の記号は最初の数字が対象システムの状態を，**F♯** は故障のタイプを表わす．ただし，**N** は正常を表わし，**A** は生起を特定できない故障タイプが存在することを示す．

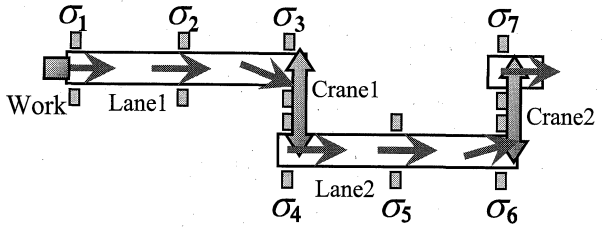

図 9　自動搬送ラインの例
ワークがセンサを通過するごとに事象としてそれを検出する．

することを示す．対象システムと診断器の動作は以下の通りである．初期状態においては対象システムの状態は 1 で診断器の状態は 1N である．このとき，診断器の状態が 1N であることは，対象システムの状態が 1 で故障タイプが N，つまり正常であることを意味している．その後，対象システムにおいて事象 α の生起が観測されたとする．このとき，対象システムにおいて到達している可能性のある状態は 4, 9, 16, 20 であり，それぞれの場合において N, F1, F2, F2 の故障が発生している可能性がある．この情報が診断器において初期状態から α で遷移した状態に記載されている．

このように診断器は観測可能な事象列を観測することでその時点で到達している可能性のある状態と生起している可能性のある故障タイプを推定することができる．文献 7) では上記のアプローチを実プラントに適用した例が示されているので興味のある読者は参照されたい．また，上述した可診断性は条件的に少し厳しいため，各故障タイプに対応した可観測なインデックス事象を定義し，インデックス事象が観測されれば故障タイプを特定できる，という性質を表わす I-可診断性 (I-diagnosability) の概念[5] も定義されている．

一方，上記のように必ずしも故障を 1 つの事象として捉えられない場合もある．たとえば，**図 9** に示すような搬送ラインでは，左端から投入されたワークがセンサを通過するごとに事象としてそれを検出するが，ラインが完全に止まってしまうのではなく，アクチュエータ等の不具合により搬送速度が 10% 程度低下するとか，センサに関しても完全に反応しなくなる，という故障ではなく，3 回に 1 回程度未検出となる，といった故障も十分にありうる．このような場合は事象の生起順序のみではなく，生起時刻の情報にも目を向けないと診断は行えない．文献 6) では，事象列に生起時刻の情報を付加したモデルに基づいてこのような故障に対する診断手法が提案されている．そこでは，まず時間付事象列を以下のように定義する．

$$E_t(0, t_h) = (e_0, t_0; e_1, t_1; e_2, t_2; \cdots, ; e_H, t_H)$$

ここで，e_k は k 番目に生起した事象を表わし，t_k はその生起時刻を表わす．t_h は観測対象とする時間区間を表わし，e_H, t_H はその時間区間内で生起した最後の事象とその生起時刻を表わす．診断は直感的には，正常な場合と故障が発

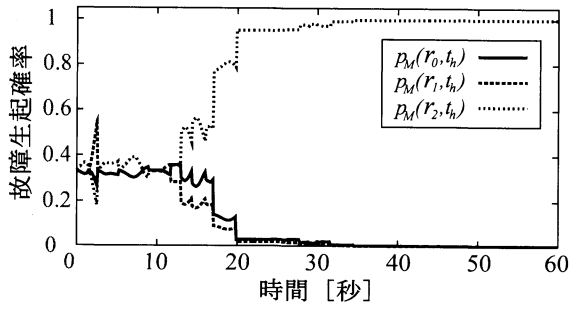

図 10　故障診断の例
「故障タイプ r_0：正常」,「故障タイプ r_1：センサ σ_5 が 20%の確率で無反応」,「故障タイプ r_2：ライン 1 の速度が 10%低下」としており，実際には故障タイプ r_2 が発生した場合を示している．$p_M(r_2, t_h)$ が最終的に最も高い値を示している．

生した場合にそれぞれ観測される $E_t(0, t_h)$ に差が生じることを利用して行われる．より具体的には故障タイプ（正常時も含めて）を r としたときに，各故障タイプに対して，それが時刻 t_h までに発生している確率

$$p_M(r, t_h) = Prob(r|E_t(0, t_h)) \qquad (10)$$

を逐次的に計算することで診断が行われる[6]．上式の右辺の計算には，次式で与えられるような，各故障タイプ r のもとで起こりうるすべての連続して生起する事象間 (σ_i と σ_j とする) の生起時間間隔に関する確率密度関数が必要となる．

$$p(\tau, \sigma_i, \sigma_j, r) = \frac{d}{d\tau} Prob(e_{k+1} = \sigma_i, t_{k+1} \leq t_k + \tau | e_k = \sigma_j, r)$$

診断のためにはこの関数を事前に推定しておく必要があるが，そのための学習データは，現実に観測され蓄積されたプラントの日常の運転データ（故障も含めた）から必要箇所を抜き出して用いることになる．また，データからの確率密度関数の推定にはさまざまな手法が考えられるが，文献 8) ではシステマティックな手法の 1 つとして最大エントロピー法が用いられており，興味のある読者は参照されたい．

例 6　図 9 のラインにある一定間隔でワークを連続投入した場合の，(10) 式で与えられる故障確率の推移を**図 10** に示す[8]．ただし図 10 では，「故障タイプ r_0：正常」,「故障タイプ r_1：センサ σ_5 が 20%の確率で無反応」,「故障タイプ r_2：ライン 1 の速度が 10%低下」としており，実際には故障タイプ r_2 が発生した場合を示している．この場合，$p_M(r_2, t_h)$ が最終的に最も高い値を示しており，診断が正しく行われている．

5.　おわりに

本稿では，離散事象システムのシステム論的アプローチについておもに形式言語理論をベースとした手法について

紹介し，制御問題，診断問題について例を挙げて解説した．形式言語理論に基づくアプローチについての詳細，およびより発展的な話題については各参考文献を参照していただきたい．日本語で書かれたスーパバイザ制御，故障診断の解説記事としては，文献1), 9)～11)などがある．英語で書かれているが，文献2)では，形式言語理論に基づく離散事象システムの制御について，基礎的事項がわかりやすく解説されており，興味のある読者には一読をお勧めする．また，モデリング，スーパバイザ制御，故障診断のツールとしては，トロント大学で開発されたTCT[12]，ミシガン大学で開発されたUMDES-LIB[13]などがある．また，2006年に開催されたThe 8th International Workshop on Discrete Event Systemsにおいて，2つのツールデモンストレーションのセッションがあり，計15ツールが紹介されている[14]．

本稿では紹介できなかったが，離散事象システムに対するモデル化手法は多岐に渡っており，形式言語理論以外にも，並行システムが表現可能なペトリネット(Petri net)[15]，連続系と同様な状態空間モデルが導出できるMax-Plus代数(Max-Plus algebra)[16]，システムの動作検証用のモデルとして有用な時間論理(temporal logic)[17]等がある．どのモデル化手法を用いるべきかは対象とする問題に依存しているため，問題に適したモデル化手法を選択することが重要となる．また，離散事象系の離散状態に連続時間系の挙動を埋め込んだハイブリッドシステム(hybrid system)[18]に関する研究も盛んに行われている．ハイブリッドシステムに関してはまだまだ理論面での課題も多く，若手の研究者にとっても魅力的な分野であると思われる．

形式言語理論がもともとは計算機科学や情報科学の分野で発展した体系であることからもわかるように，離散事象システムに関する理論は，計算機科学や情報科学で得られた成果にシステム制御工学的なアイデアを融合して新たな理論体系として発展する場合が多い．その意味では，新たな学際領域を切り拓く上でのキーテクノロジーとなりうる理論体系と言えよう．

(2006年12月11日受付)

参考文献

1) 潮 俊光：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，**34**–9, 531/538 (1990)
2) C. G. Cassandras and S. Lafortune: Introduction to Discrete Event Systems, Kluwer Academic Publishers (1999)
3) P. J. Ramadge and W. M. Wonham: Supervisory control of a class of discrete event processes, SIAM Journal on Control and Optimization, **25**–1, 206/230 (1987)
4) W. M. Wonham: Supervisory Control of Discrete-Event Systems, http://www.control.utoronto.ca/cgi-bin/dldes.cgi
5) M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen and D. Teneketzis: Diagnosability of discrete-event systems, IEEE Transactions on Automatic Control, **40**–9, 1555/1575 (1995)
6) J. Lunze: Diagnosis of quantized systems based on a timed discrete-event model, IEEE Transactions on Systems, Man, and Cybernetics, Part A, **30**–3, 322/335 (2000)
7) M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen and D. C. Teneketzis: Failure diagnosis using discrete-event models, IEEE Transactions on Control Systems Technology, **4**–2, 105/124 (1996)
8) 斉藤光生，鈴木達也ほか：最大エントロピー原理を用いた時間付きマルコフモデルによる事象駆動系の故障診断，計測自動制御学会論文集，**42**–9, 1067/1075 (2006)
9) 高井重昌：離散事象システムの分散スーパバイザ制御，計測と制御，**40**–12, 927/931 (2001)
10) 高井重昌：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，**46**–3, 138/143 (2002)
11) 潮 俊光：モデルに基づく離散事象システムの故障診断，システム/制御/情報，**42**–2, 97/102 (1998)
12) http://www.control.utoronto.ca/people/profs/wonham/wonham.html
13) http://www.eecs.umich.edu/umdes/toolboxes.html
14) Proceedings of the 8th International Workshop on Discrete Event Systems (2006)
15) 村田忠夫：ペトリネットの解析と応用，近代科学社 (1992)
16) B. Heidergott, G. Jan Olsder and J. van der Woude: Max Plus at Work, Princeton University Press (2006)
17) E. A. Emerson: Temporal and Modal Logic, Handbook of Theoretical Computer Science, MIT Press (1991)
18) たとえば，特集：ハイブリッドシステムの最前線，計測と制御，**44**–7 (2005)

[著者紹介]

高井重昌君(正会員)

1989年神戸大学工学部システム工学科卒業．91年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．92年大阪大学工学部助手，98年和歌山大学システム工学部講師，99年同助教授，2004年京都工芸繊維大学工芸学部助教授，06年同大大学院工芸科学研究科助教授となり，現在に至る．博士(工学)．離散事象システムなどの研究に従事．95年システム制御情報学会論文賞受賞．電子情報通信学会，システム制御情報学会，IEEE各会員．

鈴木達也君(正会員)

1991年名古屋大学大学院博士課程後期課程電子機械工学専攻修了．工学博士．同年名古屋大学工学部助手，2000年同助教授，06年同教授，現在に至る．この間，1998年から1年間，U.C.Berkeley客員研究員．ハイブリッドシステム論に基づく知的システムの実現や人間行動の解析とその人間機械系への応用等に関する研究に従事．1995年電気学会論文賞，2004年SICEシステム情報部門学術講演会論文賞などを受賞．電気学会，電子情報通信学会，日本機械学会，システム制御情報学会，日本ロボット学会，日本人間工学会，IEEEの各会員．