プロセスの順序性に着目した事象駆動型システムの確率的故障診断

山口 拓真*・稲垣 伸吉*・鈴木 達也*

Probabilistic Fault Diagnosis of Event-driven System Focused on Sequential Order of Processes

Takuma YAMAGUCHI*, Shinkichi INAGAKI* and Tatsuya SUZUKI*

This paper presents a fault diagnose method for event-driven systems based on probabilistic inference, Timed Markov Model (TMM). The TMM is one of semi-Markov process models wherein state transition probabilities depend on stay time at a state. The TMM requires numerous amount of learning data to establish a diagnosis system, and it is difficult to collect enough learning data in a large and complex system. This paper focuses on a diagnosis strategy based on sequential order of processes in the event-driven system, and the method can diagnose the fault even if the amount of learning data is not rich. Finally, the usefulness of the proposed strategy is verified through some experimental results of an automatic transfer line.

Key Words: event-driven system, fault diagnosis, Timed Markov Model, sequential order of processes

1. はじめに

プログラマブルロジックコントローラ (Programmable Logic Controller: PLC) に代表される事象駆動型システムは化学プラントや工場自動化など各種生産設備の自動制御に幅広く用いられている。近年の生産システムは大規模・複雑化する傾向にあり、また点検の際にはシステムを停止する必要があるため、システムのすべての故障を発見することは困難になっている。一方、システムの故障を見逃すことは生産性の低下の要因となり、さらには重大な事故を引き起こす恐れがある。そのため、システムの故障診断法を確立し、故障の検知・復帰を行なうシステム^{1), 2)}が求められている。

イベントとはセンサの発火情報であり、事象駆動型システムとはイベント信号をもとに制御されるシステムである。事象駆動型システムにおける故障診断手法の多くはイベントに着目しており、正常時と故障時に観測されるイベントの系列をモデル化し、そのモデルの違いにより故障を診断する。そのイベントのモデルには大きく分けて確定的なモデルと確率的なモデルが存在する。確定的なモデルはイベント系列をオートマトン³⁾やベトリネット⁴⁾などの状態遷移モデルで表現し、イベント系列の生起順の違いから故障を診断する。一方、確率的なモデルはイベント系列の生起順を確率で記述したモデル^{5)~9)}であり、確定的なモデルと比較すると表現能力が高く、確定的なモデルでは診断できないような故障を診断できる。

観測されるイベントとその生起時刻の系列を確率的なモデルである時間付きマルコフモデル (Timed Markov Model: TMM) により表現し、その表現に基づいて故障診断を行なう手法^{7)~9)}が提案されている。このモデルはイベント系列の生起順だけではなく、イベントが生起する時間間隔にも着目したモデルである。そのため、確率オートマトンなどのイベント系列の生起順のみを対象とした確率モデルに比べ、イベントの生起順に変化を及ぼさない故障、たとえばアクチュエータの機能低下などの故障を診断できる。そのため、文献5)の手法を用いた分散型故障診断手法への適用⁸⁾や、未検討事項であったイベント間の生起時間間隔の確率密度関数の推定手法⁹⁾などが研究されている。しかしながら、TMMを用いた手法はイベントの生起時間間隔に関してもモデル化する必要があるため、多くの学習データを必要とする。また、動作仕様の変更などによりイベントの生起時間間隔の変更があった場合は学習をやり直さなければならず、再学習に伴うデータの収集と計画の負荷が大きいという問題がある。

本論文では、TMMの学習量を削減するため、事象駆動型システムにおいてイベントによって生起する動作に着目し、その動作の組み合わせとしてシステムの挙動を表現することを考える。事象駆動型システムは生起したイベントに対応してシステムの制御が切り替えられる。たとえばシーケンス制御では、システムがセンサの活性に連動してアクチュエータを動作させ、システムの一連の動作 (工程) が実現される。このとき、システムの挙動はアクチュエータによる複数の工程の組み合わせとして表現できる。このようにシステムの挙動をいくつかの小さな工程の組み合わせとして表現することにより、学習量の低減が期待できる。

システムの挙動をいくつかの工程に分割して考えることに

* 名古屋大学大学院工学研究科 名古屋市中種区不老町

* Graduate School of Engineering, Nagoya University, Furocho, Chikusa-ku, Nagoya

(Received July 16, 2013)

(Revised October 11, 2013)

Fig. 1 Simulated transfer line

より、学習の方法にも違いが生じる。従来手法^{7)~9)}においてシステム全体の挙動をモデル化する場合には、診断時と学習時の状況が同じでなければならない。一方、提案手法ではいくつかの工程に分割して考えているので、一つ一つの動作に分解して学習すればよい。そのため、イベントの生起時間間隔などの変更が生じたとしても、システム全体の挙動を再学習する必要はなく、変更部分のみを学習することにより対応できる。

本論文は以降、つぎのように構成される。2章では本論文の問題設定について述べる。3章ではTMMによるモデル化手法、4章ではTMMを用いた故障診断手法、5章ではTMMの学習方法について述べる。そして、6章において本提案手法を自動搬送ラインのシミュレータに適用した例を示し、その有用性を実証する。

2. 問題設定

2.1 診断対象のシステム

イベントを生起させる物理的主体をトークンと呼び、トークンに対する工程により生起するイベントの系列をプロセスと呼ぶことにする。本論文は、トークンが入力されるとプロセスが生起するシステムを診断の対象とする。システムにおけるイベントのラベルの総数が n_v であるとき、イベントのラベルの集合を $\mathbf{V} = \{v_i | i \in \{0, 1, \dots, n_v\}\}$ とする。ただし、イベント v_0 はトークンにより生じるイベントではなく、トークンが投入されていないことを示すイベントとする。そして、トークンのラベルの集合を $\mathbf{O} = \{o_i | i \in \{1, \dots, L\}\}$ とする。ただし、 L はシステムに入力できるトークンの上限値である。本論文ではシステムの動作をイベントの系列として表現するため、プロセスをイベントの系列として表現する。このとき、プロセスの集合 $\mathbf{P}$ とその要素 p_i を

$$p_i = v'_1 v'_2 \dots v'_{n_{p_i}} \quad \forall j \in \{1, \dots, n_{p_i}\} \quad (v'_j \in \mathbf{V}), \quad (1)$$

$$\mathbf{P} = \{p_i | i \in \{1, \dots, n_p\}\} \quad (2)$$

と定義する。トークンの入力により、(1)式において定められたプロセスが実行される。トークンからプロセスへの写像を $f: \mathbf{O} \rightarrow \mathbf{P}$ とする。システムの状態はトークンがプロセス中に生起させた直近イベント $s_{o_i} \in \mathbf{V}$ の組 $\mathbf{S} = (s_{o_1}, \dots, s_{o_L})$ とする。そして、システムの初期状態を $\mathbf{S}_0$ とする。このとき、本論文が対象とするシステムは $\Sigma = (\mathbf{V}, \mathbf{O}, \mathbf{P}, \mathbf{S}, \mathbf{S}_0, f)$

Fig. 2 Observation of timed events

Fig. 3 Diagnoser and event-driven system

と記述できる。

例として Fig. 1 のコンベアとクレーンからなるワークの搬送システムを考える。ここで、 v_i はセンサのラベル、トークンは運搬されるワークとなる。このとき、トークンにおけるプロセスはワークの搬送動作であり、ワークが投入されてから搬送終了までに生起するイベントの系列として表現される。また、システムの状態は各ワークの搬送動作中に生起させた直近のイベントの組となり、ワークの搬送状況を表わしている。

2.2 故障診断問題の定式化

イベントの生起時間間隔を考慮するため、イベントは時間付きイベントとして観測されるものとする。時間付きイベントとは発火したセンサのラベルとその生起時刻の組である。システムにおいて n 回目に観測された時間付きイベント e_n をつぎのように定義する：

$$e_n = (v_n, t_n). \quad (3)$$

$v_n \in \mathbf{V}$ は生起したイベントのラベルであり、 t_n は v_n が発火した時刻である。Fig. 2 はイベント系列とその生起時刻を表わしており、観測イベント系列 $\mathbf{e}$ を $e_{1:n} = (e_1, e_2, \dots, e_n)$ と現在時刻 t の組としてつぎのように定義する：

$$\mathbf{e} = (e_{1:n}, t). \quad (4)$$

このとき、観測イベント系列 $\mathbf{e}$ は t_n から t までの間にイベントが生起しなかったことを表わしていることに注意されたい。

事象駆動型システムを Fig. 3 のようにプラントとコントローラに分けて考え、コントローラからの出力を観測できるものとする。 k 回目におけるアクチュエータ i における制御指令値を c_k^i 、コントローラからの出力値（アクチュエータへの指令値）を $c_k = (c_k^1, c_k^2, \dots, c_k^{n_a})$ 、そのときの時刻を t_k^c と定義する。ただし、 n_a はシステムのアクチュエータ数である。この時間付き出力の系列を制御指令系列 $\mathbf{c}$ として定義する：

$$\mathbf{c} = ((c_1, t_1^c), (c_2, t_2^c), \dots, (c_{n_c}, t_{n_c}^c)). \quad (5)$$

Fig. 4 The event observation of parallel processes

また、本論文では故障はシステムの動作に影響を与えるシステムの状況であるとし、故障の集合を $\{r_0, r_1, \dots, r_{n_r}\}$ とする。つまり、故障は観測イベント系列 $\mathbf{e}$ を変化させるシステムの状況である。ただし、 n_r は想定された故障の総数である。また、故障は正常な状況を含み、 r_0 とする。 $\mathbf{e}, \mathbf{c}, r$ の確率変数を $\mathbf{E}, \mathbf{C}, R$ とするとき、システムの故障 r^* を推定する故障診断問題はつぎのような最尤推定問題として書ける。

Given : Observed event and control state sequences $\mathbf{e}, \mathbf{c}$,
find : $r^* = \arg \max_r P(R = r | \mathbf{E} = \mathbf{e}, \mathbf{C} = \mathbf{c})$. (6)

(6) 式の確率を計算するためには、 $\mathbf{E}$ と $\mathbf{C}$ 、つまり制御指令系列と観測イベントの系列の関係を確率で表現する必要がある。しかし、制御指令系列と観測イベントの系列の関係はシステムや工程に大きく依存し、かつそれぞれが系列で与えられるため、その関係を直接表現することは難しい。そこで、トークンごとのプロセスを明示的に考慮することにより、制御指令系列と観測イベント系列の関係を表現する。

2.3 各トークンにおける時間付きイベント

時間付きイベントはセンサのラベルと生起時刻の組として観測されるだけであり、時間付きイベントがどのトークンによって生起したか区別することはできない。しかし、ここでは仮に時間付きイベントがどのトークンによって生起したかを区別できる場合を考える。このとき、トークン o_i が生起させた n_i 回目のイベント $v_{n_i}^i$ とその時刻 $t_{n_i}^i$ を時間付きイベント $e_{n_i}^i$ として定義する：

$$e_{n_i}^i = \begin{cases} (v_{n_i}^i, t_{n_i}^i), & \text{if } n_i \geq 1, \\ \phi, & \text{otherwise.} \end{cases} \quad (7)$$

ただし、 ϕ はイベントが生起していないことを示す値である。

トークン o_i が生起させる時間付きイベントの系列 $e_{1:n_i}^i = (e_1^i, \dots, e_{n_i}^i)$ と現在時刻 t の組をトークンイベント系列 $\mathbf{e}^i$ として定義する：

$$\mathbf{e}^i = (e_{1:n_i}^i, t). \quad (8)$$

また、 $\mathbf{e}^i$ の確率変数を $\mathbf{E}^i$ とする。ただし、繰り返しになるが、本論文で取り扱う故障診断においては、時間付きイベントがどのトークンによって生起したかを区別できない。すなわち、 $\mathbf{e}^i$ を直接観測できないことに注意されたい。

2.4 各変数間の関係

本論文は対象とするシステムを以下の条件を満たすものとして、故障診断手法を適用する：

条件 1 $\mathbf{e}^i$ はプロセスに従う順序性を持ち、 $\mathbf{e}$ は各 $\mathbf{e}^i$ の重ね合わせとなる。

条件 2 プロセスはほかのプロセスから直接影響を受けず、システムの制御およびシステムの故障状況のみから直接影響を受ける。

条件 3 プロセスは有限時間内に終了する。

条件 4 時間付きイベントがどのトークンによって生起したか区別できない。

条件 1 では観測されるイベントはトークンによって生起され、プロセスで生起したイベントの重ね合わせとして表わすことができることを示している。例として Fig. 4 では、プロセスは $p_1 = v_1 v_2 v_3$ であり、すべてのトークンはプロセスの順番にイベントを生起させる。イベントはトークンによって生起するため、 $\mathbf{e}$ は $\mathbf{e}^i$ の重ね合わせとして表現される。このとき、各 $\mathbf{e}^i$ を重ね合わせのイベント系列として表現する関数を

$$g : (\mathbf{e}^1, \mathbf{e}^2, \dots, \mathbf{e}^L) \mapsto \mathbf{e} \quad (9)$$

と定義する。条件 2 を Fig. 1 の搬送システムの例で考えると、ワークのプロセスは投入されているほかのワークの個数や状況に依存せず、システムの制御と故障状況のみに依存して実行されることを意味する。しかし、プロセスは制御により、ほかのプロセスから間接的に影響を受けることがある。本論文ではこのほかのプロセスからの間接的な影響は既知とする（詳細は 4.1 節において述べる）。また条件 2 により、提案手法ではプロセスの生起や状況によってほかのプロセスへ影響を与えるような故障（ワークの投入に関する故障など）は取り扱うことができない。プロセスが有限時間内に終了しないとすると、システムの稼働時間の増加に従いプロセス数が無限に増加し続けてしまうため、条件 3 においてプロセスは有限時間内に終了するものとする。条件 4 は 2.3 節において述べたとおりである。

上記の条件を満たすとき、各変数の関係は Fig. 5 のベイジアンネットワークにより表現できる。図中の灰色のノードは観測できる変数を、白色はできない変数を表わしている。ここで、条件 2 より $\mathbf{E}^i$ は R と $\mathbf{C}$ に関して条件付き独立であるとしている。また条件 1 より故障 R の影響は $\mathbf{E}^i$ に直接影響を与え、間接的に観測イベント系列 $\mathbf{E}$ に影響を与える。このとき、 $\mathbf{E}^i$ は $\mathbf{E}$ を表現するための、観測できない中間変数であると捉えることができる。

以降では、式の表記の簡略化のために、たとえば $E_{1:n_i}^i = e_{1:n_i}^i$ を $E_{1:n_i}^i$ のように、実現値を省略して確率変数を表現する。

3. システムの故障診断

本章では (6) 式の故障確率 $P(R | \mathbf{E}, \mathbf{C})$ の計算法について

Fig. 5 Relation between observations and faults: Grey nodes mean observable ones, white nodes non-observable ones

述べる．故障確率 $P(R|\mathbf{E}, \mathbf{C})$ は Fig. 5 のベイジアンネットワークから，

$$P(R|\mathbf{E}, \mathbf{C}) = \sum_{\mathbf{E}^1, \dots, \mathbf{E}^L} P(R|\mathbf{E}^1, \dots, \mathbf{E}^L, \mathbf{C}) P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{E}, \mathbf{C}) \quad (10)$$

と表現できる．ここで， $P(R|\mathbf{E}^1, \dots, \mathbf{E}^L, \mathbf{C})$ は各トークンイベント系列 $\mathbf{E}^i$ と制御指令系列 $\mathbf{C}$ が与えられたときの故障確率であり， $P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{E}, \mathbf{C})$ は観測イベント系列 $\mathbf{E}$ と制御指令系列 $\mathbf{C}$ が与えられたときの， $\mathbf{E}^i$ の生起確率である．つまり，観測イベント系列 $\mathbf{E}$ と制御指令系列 $\mathbf{C}$ に基づき各トークンイベント系列 $\mathbf{E}^i$ の生起確率を計算し，各 $\mathbf{E}^i$ が得られたときの故障確率を計算している．

3.1 トークンに着目した故障確率の計算

(10) 式の確率 $P(R|\mathbf{E}^1, \dots, \mathbf{E}^L, \mathbf{C})$ は Fig. 5 のベイジアンネットワークとベイズの定理より

$$P(R|\mathbf{E}^1, \dots, \mathbf{E}^L, \mathbf{C}) = \frac{P(\mathbf{E}^1, \dots, \mathbf{E}^L|R, \mathbf{C})P(R)}{\sum_R P(\mathbf{E}^1, \dots, \mathbf{E}^L|R, \mathbf{C})P(R)} = \frac{P(R) \prod_{i=1}^L P(\mathbf{E}^i|R, \mathbf{C})}{\sum_R P(R) \prod_{i=1}^L P(\mathbf{E}^i|R, \mathbf{C})} \quad (11)$$

と計算できる． $P(R)$ は故障の事前確率であり，過去のシステム運用に基づく統計的確率や専門家の先験的知識などから決定される．また， $P(\mathbf{E}^i|R, \mathbf{C})$ はトークンイベント系列の生起確率である．

3.2 観測に基づくトークンイベント系列の生起確率計算

(10) 式の確率 $P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{E}, \mathbf{C})$ はベイズの定理によりつぎのように表現できる．

$$P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{E}, \mathbf{C}) = \frac{P(\mathbf{E}|\mathbf{E}^1, \dots, \mathbf{E}^L)P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{C})}{\sum_{\mathbf{E}^1, \dots, \mathbf{E}^L} P(\mathbf{E}|\mathbf{E}^1, \dots, \mathbf{E}^L)P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{C})} \quad (12)$$

このとき， $P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{C})$ は Fig. 5 のベイジアンネットワークからつぎのように分解できる．

$$P(\mathbf{E}^1, \dots, \mathbf{E}^L|\mathbf{C}) = \sum_R P(R) \prod_{i=1}^L P(\mathbf{E}^i|R, \mathbf{C}). \quad (13)$$

3.3 故障確率の計算

故障確率 $P(R|\mathbf{E}, \mathbf{C})$ は (10), (11), (12), (13) 式から

$$P(R|\mathbf{E}, \mathbf{C}) = \frac{1}{Z} \sum_{\mathbf{E}^1, \dots, \mathbf{E}^L} \left\{ P(R) \prod_{i=1}^L \{P(\mathbf{E}^i|R, \mathbf{C})\} \times P(\mathbf{E}|\mathbf{E}^1, \dots, \mathbf{E}^L) \sum_R \{P(R) \prod_{j=1}^L P(\mathbf{E}^j|R, \mathbf{C})\} \right\} \quad (14)$$

と計算できる．ただし， Z は正規化項である．(14) 式では各トークンイベント系列 $\mathbf{E}^j$ の生起確率 $P(\mathbf{E}^j|R, \mathbf{C})$ を求め，故障確率を計算している．この計算では各 $\mathbf{E}^j$ に対して周辺化計算しなければならないため，生じ得る $\mathbf{E}^j$ のすべての候補に対して確率を計算する必要がある．このとき， $\mathbf{E}^j$ 本来は $\mathbf{E}^j$ の候補は生じ得るイベントの組み合わせを考慮しなければならない．しかし， $\mathbf{E}^j$ はプロセスのイベントの順序どおりに生起するため，イベントの組み合わせを考慮する必要はなく，プロセスの順序に従って計算すればよい．確率 $P(\mathbf{E}|\mathbf{E}^1, \dots, \mathbf{E}^L)$ は各トークンイベント系列 $\mathbf{E}^i$ が与えられたときの観測イベント系列 $\mathbf{E}$ の生起確率である． $\mathbf{E}$ は $\mathbf{E}^i$ の重ね合わせとして表現されるため，各 $\mathbf{E}^i$ が与えられる場合は $\mathbf{E}$ を一意的に定めることができる．つまり，

$$P(\mathbf{E}|\mathbf{E}^1, \dots, \mathbf{E}^L) = \begin{cases} 1 & \text{if } \mathbf{e} = g(\mathbf{e}^1, \dots, \mathbf{e}^L) \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

となる．したがって，(14) 式の故障確率を計算するためには故障確率 $P(R)$ ，トークンイベント系列の生起確率 $P(\mathbf{E}^j|R, \mathbf{C})$ の値が必要である．次章では TMM を用いて $P(\mathbf{E}^j|R, \mathbf{C})$ を求める方法について説明する．

4. 時間付きマルコフモデル

Timed Markov Model (TMM) は準マルコフ過程モデルの一つであり，状態遷移確率が現在の状態における滞在時間 τ に依存するモデルである．時間付きイベント系列を TMM で表現するために，TMM の状態を生起したイベントのラベルとする⁷⁾．このとき，TMM の状態遷移確率は生起したイベントとつぎのイベントが生起するまでの時間 τ の同時確率として表現できる．なお，イベントの生起時間間隔 τ は離散時間であるとし，状態遷移確率は同時確率分布により表わされる．

以下に故障 r においてトークン o_i があるプロセス中に生成させるトークンイベント系列を表現する TMM を考える．

定義 1. 時間付き遷移確率 (Timed Transition Probability: TTP) とは，故障が r であり，イベント $v_{n_i}^i$ が時刻 $t_{n_i}^i$ に生起し，そして， $\tau = t_{n_i+1}^i - t_{n_i}^i$ 時間後に $v_{n_i+1}^i$ が発火するときの同時確率分布 $P(E_{n_i+1}^i = e_{n_i+1}^i | E_{n_i}^i = e_{n_i}^i, R = r)$ である．ここで，時間付き遷移確率はイベントが生起する絶対時刻に依らないと仮定すると，

$$\begin{aligned} P(E_{n_i+1}^i = e_{n_i+1}^i | E_{n_i}^i = e_{n_i}^i, R = r) &= P(V_{n_i+1}^i = v_{n_i+1}^i, T_{n_i+1}^i = t_{n_i}^i + \tau | V_{n_i}^i = v_{n_i}^i, T_{n_i}^i = t_{n_i}^i, R = r) \\ &= P(V_{n_i+1}^i = v_{n_i+1}^i, T = \tau | V_{n_i}^i = v_{n_i}^i, R = r) \\ &\stackrel{\text{def}}{=} a_{v_{n_i}^i v_{n_i+1}^i}^r(\tau) \end{aligned} \quad (16)$$

Fig. 6 The example of Left-to-Right TMM representing a process

となる。ただし $V_{n_i+1}^i, T_{n_i+1}^i, T$ は $v_{n_i+1}^i, t_{n_i+1}^i, \tau$ の確率変数である。

各トークンのイベントの生起時間を時刻ではなく、時間間隔として表現することにより、トークンの投入時刻に依らずトークンイベント系列を表現することができる。

TMM はトークンがあるプロセス中に生起させる時間付きイベントの生成過程を表現している。システムにおいて各トークンは区別できないため、同一プロセスでは各トークンの間に差はないものとする。つまり、TMM はプロセスのみに依存する。また、各プロセスにおいて生起するイベントのラベルは順序性をもつ。例として Fig. 4 のシステムを考える。このとき、すべてのトークンはプロセスに従い v_1, v_2, v_3 と順番にイベント生起するため、TMM は Fig. 6 の Left-to-Right 型のモデルとして表現できる。また、Fig. 6 の TMM の状態遷移は限られているため、少ない TTP で表現することができる。このように、システムで観測されるイベント系列 $\mathbf{E}$ ではなく、各トークンのイベント系列 $\mathbf{E}^i$ に着目することにより、TTP はすべてのイベントの組み合わせを考慮する必要はなく、限られた状態遷移で表現できる。よって、TTP の数は $O(n_v^2)$ から $O(n_v)$ へ減少し、学習量の削減を期待できるのである。

TMM の初期状態を v_0 とし、この状態はトークンが投入されていない待機状態を表す。トークンにおいて初回に観測されるイベント v_1 が観測されたとき、TMM は初期状態から状態 v_1 に確率 1 で遷移するものとする。

4.1 制御指令系列による影響

トークンイベント系列 $\mathbf{E}^i$ は制御指令系列 $\mathbf{C}$ の影響を受ける。 $\mathbf{E}^i$ が $\mathbf{C}$ から受ける影響は制御指令値の変更によって生じるため、制御手法に着目するとその影響がわかる場合がある。その例として一時停止制御と速度変化制御について考え、 $\mathbf{C}$ の影響を考慮した TTP をつぎのように定義する。

定義 2. 故障が r であり、制御指令系列 $\mathbf{C}$ と時間付きイベント $E_{n_i}^i$ が観測され、時間付きイベントが生起する絶対時刻に依らないと仮定したとき、つぎの時間付きイベント $E_{n_i+1}^i$ が生起する TPP を

$$f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c}) \stackrel{\text{def}}{=} P(E_{n_i+1}^i | E_{n_i}^i, R, \mathbf{C}) \quad (17)$$

とする。

4.1.1 一時停止制御

一時停止制御はリソースの競合や例外処理などによりアクチュエータを一時的に停止させる制御であり、TMM の一時停止処理に相当する。一時停止制御が t_a から t_b まで動作し

Fig. 7 A example of $a_{v_{n_i}^i v_{n_i+1}^i}^r(\tau)$

Fig. 8 $f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c})$ after a stop action control

Fig. 9 $f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c})$ after a speed change control

ているときの TPP はつぎのようになる。

$$f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c}) = \begin{cases} a_{v_{n_i}^i v_{n_i+1}^i}^r(\tau) & (0 \leq \tau \leq t_a) \\ 0 & (t_a < \tau \leq t_b) \\ a_{v_{n_i}^i v_{n_i+1}^i}^r(\tau - (t_b - t_a)) & (t_b < \tau). \end{cases} \quad (18)$$

Fig. 7 は制御指令系列 $\mathbf{C}$ の影響がないときの TTP を示している。Fig. 8 は一時停止制御の影響がある場合の TTP であり、停止期間に 0 の確率を挿入することにより表現できる。

4.1.2 速度変化動作

つぎにアクチュエータの動作速度が変化する場合の影響について考える。アクチュエータの動作速度が α 倍に変化するとき、トークンのイベントの生起確率は α 倍に、アクチュエータの動作時間は $1/\alpha$ 倍に変化すると考えられる。時刻 t_a から動作速度が変化するときの TTP は

$$f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c}) = \begin{cases} a_{v_{n_i}^i v_{n_i+1}^i}^r(\tau) & (0 \leq \tau \leq t_a) \\ \alpha a_{v_{n_i}^i v_{n_i+1}^i}^r\left(\frac{1}{\alpha}(\tau - t_a) + t_a\right) & (t_a < \tau) \end{cases} \quad (19)$$

となる。Fig. 9 は時刻 t_1 において実行速度が変化したときの TTP であり、TTP の確率は時間的に伸縮したように表現できる。

4.2 トークンイベント系列の生起確率の計算

故障状況が r であるときの、トークンイベント系列の生起確率は TMM より

$$P(\mathbf{E}^i = \mathbf{e}^i | R, \mathbf{C})$$

$$= P(E_{1:n_i}^i = e_{1:n_i}^i, [T_{n_i+1}^i > t] = \text{True} | R, \mathbf{C}) \quad (20)$$

と表現できる．ここで， $[X]$ は論理式 X が成立するか否かを表わす確率変数であり，実現値として $\{\text{True}, \text{False}\}$ を取るものとする．(20) 式は，故障状況が r であり，制御指令系列が $\mathbf{C}$ の場合のとき， $e_{1:n_i}^i$ が得られ，かつ現在時刻 t までセンサの発火がないときの確率である．

過去直近のイベント生起時刻 $t_{n_i}^i$ から現在時刻 t までの間センサが発火しないため，(20) 式はつぎのように計算される．

$$P(\mathbf{E}^i = \mathbf{e}^i | R, \mathbf{C})$$

$$= P(E_{1:n_i}^i, [T_{n_i+1}^i > t] | R, \mathbf{C})$$

$$= P([T_{n_i+1}^i > t] | E_{1:n_i}^i, R, \mathbf{C}) P(E_{1:n_i}^i | R, \mathbf{C}). \quad (21)$$

ここで，(21) 式右辺の確率 $P([T_{n_i+1}^i > t] | E_{n_i}^i, R, \mathbf{C})$ は直近のイベントから現在時刻までにイベントが生起しない確率であり，TMM のマルコフ性と TTP を用いて

$$P([T_{n_i+1}^i > t] | E_{n_i}^i, R, \mathbf{C})$$

$$= P([T_{n_i+1}^i > t] | E_{n_i}^i, R, \mathbf{C})$$

$$= 1 - \sum_{v_{n_i+1}^i \in \mathbf{V}} \int_{t_{n_i}^i}^t P(V_{n_i+1}^i = v_{n_i+1}^i, T_{n_i+1}^i = \tau | E_{n_i}^i, R, \mathbf{C}) d\tau$$

$$= 1 - \sum_{v_{n_i+1}^i \in \mathbf{V}} \int_{t_{n_i}^i}^t f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c}) d\tau \quad (22)$$

と計算することができる．

また， $T_{n_i+1}^i = t$ となったとき，つまり新たに時間付きイベントが生起したとき，(22) 式の n_i は n_{i+1} に更新される．このとき，確率 $P(E_{1:n_i+1}^i | R, \mathbf{C})$ は以下のように計算される．

$$P(E_{1:n_i+1}^i | R, \mathbf{C}) = P(E_{n_i+1}^i | E_{1:n_i}^i, R, \mathbf{C}) P(E_{1:n_i}^i | R, \mathbf{C})$$

$$= P(E_{n_i+1}^i | E_{n_i}^i, R, \mathbf{C}) P(E_{1:n_i}^i | R, \mathbf{C})$$

$$= f_{v_{n_i}^i v_{n_i+1}^i}^r(\tau, \mathbf{c}) P(E_{1:n_i}^i | R, \mathbf{C}). \quad (23)$$

各時刻において，イベントが観測されないときは (22) 式を用いて，観測されたときは (23) 式を用いて更新することにより，各時刻の $P(\mathbf{E}^i = \mathbf{e}^i | R, \mathbf{C})$ が得られる．

5. TTP の学習方法

本章では TMM のパラメータである TTP の学習について述べる．TTP は，一つのトークンがある一つのプロセス中に生起させる，時間付きイベントの生起確率を表現している．一つのプロセス中の TTP を学習するため，学習データを集める場合においては，投入されているトークン数が常に一つである状況とする．この状況下において，各故障状況ごとに集められた学習データからプロセス中に生起する時間付きイベント $e_{n_i}^i$ の生起確率，つまり $a_{v_{n_i}^i v_{n_i+1}^i}^r(\tau)$ を求める．そして，TMM における各 TTP を学習することにより，(10) 式

Fig. 10 Diagram of simulated transfer line

の確率を計算する．

6. 実験環境

本章では提案した手法を Fig. 10 の自動搬送ラインに適用し，その性能を調べる．なお，システムは Fig. 1 のような物理シミュレータを用いてシミュレーションした．

6.1 自動搬送ライン

このシステムは 4 機のコンベア 1, 2, 3, 4 (長さ 5 m) と 1 機のクレーンから構成され，ワークをアンロードステーション 1, 2, 3 に搬送する．各コンベアには三つのセンサが取り付けられており，Fig. 10 の位置 $S_1 \sim S_{12}$ に設置されている．

コンベア 1 の制御はセンサ S_3 にワークが侵入するとアクチュエータが一時停止し，離れると始動する．クレーンはセンサ S_3 にワークが侵入すると始動し，ワークから搬送先 (コンベア 2, 3, 4) を読み取り，その搬送先へワークを運搬する．コンベア 2, 3, 4 は安全に搬送するために S_4, S_7, S_{10} にワークが侵入すると一時停止し，タイマによって再始動する (始動時にイベント v_6, v_{10}, v_{14} を発火する)．各イベントは Table 1 に記されたとおりに発火する．Table 2 はシステムのプロセスを示しており，各ワークのプロセスはあらかじめ搬送先に応じて決められている．また，Table 3 は各アクチュエータへの指令値 c_k の集合を表わしており，制御指令系 $\mathbf{c}_k$ はこれらの組み合わせとなり，制御指令系列 $\mathbf{c}$ は c_k の時間付きの系列として与えられる．コンベアの停止はほかのワークに影響を与える制御であるため，この影響は 4.1.1 項の手法を用いて計算した．

ワークの投入間隔の確率は平均 1.8 m，分散 0.4 m^2 の正規分布とし，この分布に従い投入する．また，1 回の試行はワークを 30 個搬送することとする．

6.1.1 故障状況

本論文で想定する故障を Table 4 に示す．各ベルトコンベアの速度が 22 cm/sec から 18 cm/sec の間である場合を「正常 (r_0)」とし，18 cm/sec から 15 cm/sec の間である場合は「速度低下 (r_1, r_2, r_3, r_4)」であるとする．クレーンにおいてワークの搬送に要する時間が正常時より 0.3 sec 多くなることを「速度低下 (r_5)」であるとする．また，システムの故障に関する事前情報はなないものとして，事前確率 $P(R)$ には等確率を割り振る．

Table 1 Event labels in the transfer line

Event	Object	Action
v_1	S_1	Enter sensor
v_2	S_2	Enter sensor
v_3	S_3	Enter sensor (Stop Conveyor 1)
v_4	S_3	Leave sensor (Start Conveyor 1)
v_5	S_4	Enter sensor (Stop Conveyor 2)
v_6	Conveyor 2	Start Conveyor 2
v_7	S_5	Enter sensor
v_8	S_6	Enter sensor
v_9	S_7	Enter sensor (Stop Conveyor 3)
v_{10}	Conveyor 3	Start Conveyor 3
v_{11}	S_8	Enter sensor
v_{12}	S_9	Enter sensor
v_{13}	S_{10}	Enter sensor (Stop Conveyor 4)
v_{14}	Conveyor 4	Start Conveyor 4
v_{15}	S_{11}	Enter sensor
v_{16}	S_{12}	Enter sensor

Table 2 Fired events of processes for each destination

Process	Fired events	Destination
p_1	$v_1 v_2 v_3 v_4 v_5 v_6 v_7 v_8$	Conveyor 2
p_2	$v_1 v_2 v_3 v_4 v_9 v_{10} v_{11} v_{12}$	Conveyor 3
p_3	$v_1 v_2 v_3 v_4 v_{13} v_{14} v_{15} v_{16}$	Conveyor 4

Table 3 Command set for each actuator

Label	Object	Command set
c_k^1	Conveyor 1	{Start, Stop }
c_k^2	Conveyor 2	{Start, Stop }
c_k^3	Conveyor 3	{Start, Stop }
c_k^4	Conveyor 4	{Start, Stop }
c_k^5	Crane	{Move a work, Wait }

Table 4 Candidates of faults

Symbol	Detail of fault
r_0	The system is normal.
r_1	Speed of Conveyor 1 is reduced.
r_2	Speed of Conveyor 2 is reduced.
r_3	Speed of Conveyor 3 is reduced.
r_4	Speed of Conveyor 4 is reduced.
r_5	Speed of Crane is reduced.

Table 5 Success rate of diagnosis with proposed method

# of data	r_0	r_1	r_2	r_3	r_4	r_5	Average
10 trials	0.84	0.87	0.92	0.82	1.00	0.84	0.88
20 trials	0.91	0.88	0.96	0.87	1.00	0.96	0.93
50 trials	0.92	0.9	0.97	0.96	1.00	0.97	0.95
100 trials	0.95	0.97	0.99	0.99	1.00	0.98	0.98
200 trials	0.95	0.97	0.99	0.99	1.00	0.97	0.98
300 trials	0.96	0.99	0.93	0.98	1.00	0.98	0.97
400 trials	0.97	0.99	0.98	0.99	1.00	1.00	0.98
500 trials	0.96	0.99	1.00	0.99	1.00	1.00	0.99
1000 trials	0.96	0.99	1.00	0.96	1.00	1.00	0.99

6.2 実験結果

故障診断における検証データは各故障ごとに 100 試行、計 600 試行のデータを使用した。この検証データに提案手法を適用した故障診断結果を **Table 5** に示す。また、比較対象としてシステムの観測イベント系列 $\mathbf{E}$ をそのまま TMM によりモデル化する手法⁷⁾を従来手法として用いる。従来手法で

Table 6 Success rate of diagnosis with conventional method

# of data	r_0	r_1	r_2	r_3	r_4	r_5	Average
10 trials	0.37	0.10	0.71	0.78	0.87	0.17	0.50
20 trials	0.55	0.41	0.78	0.77	0.86	0.24	0.60
50 trials	0.46	0.69	0.74	0.84	0.98	0.38	0.68
100 trials	0.45	0.76	0.84	0.89	0.97	0.48	0.73
200 trials	0.34	0.76	0.9	0.92	0.99	0.68	0.77
300 trials	0.44	0.89	0.95	0.96	0.98	0.58	0.80
400 trials	0.51	0.82	0.93	0.97	0.98	0.55	0.79
500 trials	0.53	0.81	0.94	0.91	0.99	0.56	0.79
1000 trials	0.53	0.85	0.95	0.94	0.99	0.56	0.80

Fig. 11 The relation between average success rate and learning data**Table 7** The number of TTP in each method

Conventional method	Proposed method
1536	90

はプロセスやトークンを考慮しないため、学習データは検証データと同一の状況で収集された。提案手法では 5 章でも述べたとおり、常にワークが一つのみ投入されている状況において学習データを収集した。

従来手法による故障診断結果を **Table 6** に示す。また、平均診断成功率と学習データ量の関係を示したグラフを **Fig. 11** に示し、各手法における TTP の数を **Table 7** に示す。なお、故障の同定は次式の故障確率のエントロピーが 0.025 を下回る場合と判断した。

$$H = - \sum_R P(R|\mathbf{E}, \mathbf{C}) \log_2 P(R|\mathbf{E}, \mathbf{C}). \quad (24)$$

提案手法の診断結果である **Table 5** では、学習データ量が増加すると診断成功率も高くなる傾向があった。また、提案手法は 100 試行の学習データでも高い診断成功率を示した。一方、従来手法の診断結果である **Table 6** では、提案手法と比較すると全体的に診断成功率は低くなった。また、300 試行の学習データを用いたとき、平均の診断成功率が最も高くなった。

以上の結果より提案手法のほうが少ない学習データ量でも正しく診断できていることがわかる。この差はモデルとする対象の違いにより生ずるものと考えられる。提案手法ではシステムの観測イベント系列 $\mathbf{E}$ をトークンイベント系列 $\mathbf{E}^i$ に分割して考えている。一方、従来手法では $\mathbf{E}$ をそのままモデル化しているため、トークン間の組み合わせを考慮しなければならない。そのため、**Table 7** のように多くの TTP を学

Table 8 Success rate of diagnosis when setting is changed

Method	r_0	r_1	r_2	r_3	r_4	r_5	Average
Proposed method	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Previous method	0.00	0.32	0.22	0.55	0.76	0.00	0.31

習しなければならない。学習データの量が同じ場合、学習すべき TTP の数が少ないほうが、一つの TPP の学習量に割り当てられる学習データの量が増えるので、提案手法では従来手法より精度が高い TPP を得ることができたと考えられる。また、Table 5, Table 6 では平均診断成功率に約 20% の差が生じている。これは、トークンイベント系列が生起するイベントのラベルに関して順序性をもつため、観測イベント系列 **E** を TMM によって表現しやすくなったためであると考えられる。

つぎにワーク投入間隔が変化させた場合において実験を行った。システムへのワークの投入間隔は平均 4.0 m, 分散 0.4 m² の正規分布に変更した。なお、学習データはワークの投入間隔を変更する前のデータを用いる。各故障ごとに 100 試行、計 600 試行のデータを検証データとし、学習データ量は 1000 試行とした。

各手法における診断結果を **Table 8** に示す。提案手法はワークの投入間隔を変更前とほぼ変わらない診断成功率であったのに対し、従来手法では診断成功率が大きく減少した。提案手法では観測イベント系列を各トークンのプロセスごとに分割して考えているため、トークンの入力時間に依存せず診断できている。一方、従来手法ではトークンイベント系列を考慮していないため、ワークの投入間隔の変更により正しく診断できなかった。トークンの入力時刻はシステム外部の変更の影響により変化することがある。その場合において、提案手法では学習データの収集をやり直す必要がないため、学習データの収集の負荷を減らすことができる。

7. おわりに

本論文では事象駆動システムにおけるプロセスに着目し、システム全体で得られる観測イベントを各プロセスごとに分割して、Timed Markov Model によりモデル化を行なった。ベイジアンネットワークを用いて故障と生起するイベント系列の関係性を記述し、Timed Markov Model を用いた確率的故障診断法を提案した。そして、自動搬送ラインに提案手法を適応し、その性能を調べた。

提案手法はシステムにおける観測イベント系列を各プロセスごとに分割し、各プロセスに着目して学習を行なっている。そのため、少ない学習量で診断でき、より高い精度で診断できた。また、トークンの投入時刻に依らず故障を診断できることを示した。今後の課題として、システムの制御の変更の影響を見積もる必要があることから、この影響の見積もり手法の確立やほかのシステムへの適用が考えられる。

参 考 文 献

- 1) F. Schiller, J. Schroder and J. Lunze: Diagnosis of tran-

sient fault in quantised systems, *Engineering Applications of Artificial Intelligence*, **14**-14, 519/536 (2001)

- 2) A. Darwiche and G. Provan: Exploiting system structure in model-based diagnosis of discrete-event systems, *Proc. 7th Intl. Workshop on Principles of Diagnosis*, 95/105 (1996)
- 3) M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen and D. Teneketxis: Diagnosability of Discrete Event Systems, *IEEE Trans. Autom. Contr.*, **40**-9, 1555/1575 (1995)
- 4) A. Ramirez-Trevino, E. Ruiz-Beltran, I. Rivera-Rangel and E. Lopez-Mellado: Online Fault Diagnosis of Discrete Event Systems. A Petri Net-Based Approach, *IEEE Trans. Autom. Contr.*, **4**-1, 31/39 (2007)
- 5) J. Lunze and J. Schroder: State observation and diagnosis of discreteevent systems described by stochastic automata, *Discrete Event Dyna. Syst.: Theory Appl.*, **11**-4, 319/369 (2001)
- 6) F. Liu, D. Qiu, H. Xing and Z. Fan: Decentralized Diagnosis of Stochastic Discrete Event Systems, *IEEE Trans. Autom. Contr.*, **53**-2, 535/546 (2008)
- 7) J. Lunze: Diagnosis of Quantized Systems Based on a Timed Discrete-Event Model, *IEEE Trans. Syst. Man. Cybern.*, **30**-3, 322/335 (2000)
- 8) 山口, 稲垣, 鈴木: 分散型故障診断におけるアークの重要度指標を用いた Bayesian Network の構造構築, 計測自動制御学論文集, **48**-11, 705/712 (2012)
- 9) 齋藤, 鈴木, 稲垣, 青木: 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, 計測自動制御学論文集, **42**-9, 1067/1075 (2006)

[著 者 紹 介]

山 口 拓 真 (学生会員)

2011 年名古屋大学大学院工学研究科博士後期課程機械理工学専攻に入学。現在に至る。離散事象駆動システムにおける故障診断の研究に従事。

稲 垣 伸 吉 (正会員)

2003 年東京大学大学院工学系研究科博士課程修了。工学博士。同年名古屋大学工学部助手。2008 年同講師。現在に至る。自律分散システム、多脚歩行ロボット、確率的情報処理の研究に従事。

鈴 木 達 也 (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。2006 年同教授。現在に至る。ハイブリッドシステム論とその実システムへの応用に関する研究に従事。