バッチプロセスにおける異常の診断とその回避

2017 年

橋爪　悟

内容

第 1 章. 緒言 1
1.1. 研究の背景 1
1.1.1. 近年の国内化学産業の状況 1
1.1.2. バッチプロセスの概要 2
1.1.3. 化学プロセスにおける異常対応 3
1.1.4. 異常対応モデルを用いた異常対応 5
1.2. 既往の研究 6
1.3. 本研究の目的 8
1.4. 本論文の構成 10
第 2 章. 異常対応を目的としたバッチプロセスのモデル化 11
2.1. はじめに 11
2.2. 用語の定義 12
2.3. 離散事象システムにおける異常 15
2.4. ペトリネットを用いた離散事象システムの表現方法 16
2.5. 異常対応モデルの構築手法 18
2.5.1. 異常対応モデル構築の基本的な考え方 18
2.5.2. 異常対応モデルの構築手順 19
2.6. まとめ 22
第 3 章. バッチプロセスにおける異常の診断 24
3.1. はじめに 24
3.2. ペトリネットに基づく動作制約の導出 26
3.3. ペトリネットに基づく異常診断 27
3.3.1. 異常の検出 27
3.3.2. 異常の特定 29
3.3.3. 異常診断および異常対応モデルの更新 30
3.4. まとめ 33
第 4 章. バッチプロセスにおける異常状態の回避 35

4.1. はじめに 35
4.2. 異常状態回避問題 37
4.2.1. ペトリネットに基づく異常状態回避問題の定式化 37
4.2.2. ペトリネットに基づく異常状態回避問題の解法 38
4.3. 異常対応に係る提案手法の適用例 41
4.3.1. 対象のプロセス 41
4.3.2. 異常の検出 44
4.3.3. 異常の特定 45
4.3.4. 異常の回避 47
4.4. 異常対応モデルに係る考察 48
4.4.1. 異常対応モデルを用いた異常対応 48
4.4.2. 異常対応モデル構築に係る考察と今後の課題 52
4.5. まとめ 53
第 5 章. バッチプロセスにおける異常動作の回避 54
5.1. はじめに 54
5.2. 異常動作回避問題 55
5.2.1. ペトリネットに基づく異常動作回避問題の定式化 55
5.2.2. C/E ネット制御問題の定式化とその解法 56
5.2.3. 拡張 C/E ネット制御問題の定式化とその解法 62
5.3. C/E ネット最小実現問題 67
5.3.1. C/E ネット最小実現問題の定式化 67
5.3.2. C/E ネット最小実現問題の解法 68
5.3.3. 提案解法の効果の確認 73
5.4. 異常の回避のための制御器設計に係る考察 75
5.5. まとめ 76
第 6 章. 不可観測な状態・不可制御な動作を含むバッチプロセスの異常の回避 78
6.1. はじめに 78
6.2. 不可観測な状態，不可制御な動作のモデル化 79
6.3. 異常の検出 81
6.3.1. 動作制約の生成 81
6.3.2. 不可観測な状態を含む動作制約 84

6.4. 異常状態回避問題 85
6.4.1. 不可観測な状態を含む異常状態回避問題 85
6.4.2. 不可制御な動作を含む異常状態回避問題 87
6.4.3. 適用例 92
6.5. 異常動作回避問題 97
6.6. まとめ 100
第 7 章. 不確実性を含むバッチプロセスの異常の回避 102
7.1. はじめに 102
7.2. 不確実性を伴うバッチプロセスのモデル化 103
7.3. 異常回避問題の定式化 105
7.4. 異常回避の方法 106
7.5. 適用例 107
7.6. 考察 112
7.7. まとめ 113
第 8 章. 結言 114
8.1. 本研究の成果 114
8.2. 今後の展望 117
付録 1. ペトリネットに係る用語 119
付録 2. 半言語 120
参考文献 121
本論文に関する著者の研究業績 126
謝辞 127

第1章.緒言

1.1. 研究の背景

1.1.1. 近年の国内化学産業の状況

化学工業における石油化学製品は出荷額割合で全体の 55%を占める（2010 年データ[1]）．特に総合化学メーカーにおいては，長年において売り上げの主流を石油化学製品が占めていた．石油化学製品の製造プロセスはナフサからエチレン・プロピレン等のオレフィン系やベンゼン・トルエン等の芳香族系を分解・生成するモノマー製造プロセスと，これら生成物を重合するポリマー製造プロセスに大別される．一般に，これら石油化学製品は原料を連続的に投入し，幾つかの処理を加えて最終製品を連続的に生産する連続プロセスによる製造が行われる．連続プロセスは，製造設備のスケールに応じ，多くの生産量を実現できることから，製品数が比較的少なく，多数の川下工程への材料供給が主な役割であり，大量生産が必要となる石油化学製品と親和性が高いと言える．

一方，石油化学関連のプラントは，近年海外での建設が相次いでおり，海外における安価資源（原料・ユーティリティ・労務費）の利用により，オレフィン系や芳香族系等の汎用化学製品の販売価格は低下の一途をたどっている．このことは，今まで国内化学産業を支えてきた石油化学事業の縮小化を招く一要因となっている．例えば，三菱化学株式会社では 2014 年に鹿島第一エチレンプラントの停止[2]，住友化学株式会社では 2013 年に国内自社エチレンプラントの全停止[3]の施策を行っており，今後も国内における石油化学事業の縮小化の波は継続していくことが予測される．以上のことから，国内の総合化学メーカーは戦後 60 年近く続いた石油化学事業を中心とした事業スタイルから脱却し，新たな事業を開拓する必要があると考えられる．

また，ここ十数年の間，消費者嗜好の多様化，製品ライフサイクルの短縮化が進み，従来化学工業で主流であった少品種大量生産から，これらの多様性に早期に対応するための多品種少量生産へと移行することが必要となっている．更に，近年注目されているドイツにおける「Industrie4.0」や日本における「society5.0[4][5]」で提案されている消費者と生産者を直接つなぐオーダーメイドシステムは多品種少量生産システムの最終目標とも言える．多品種少量生産を実現するためには，理論的には二通りの方法がある．一つは固有の製品の生産を目的とした小規模プラントを多数建設し製造する方法で

あり，もう一つは同一のプラントでできるだけ多くの製品を製造する方法である．前者はコストが極めて高くなること，およびプラントを一から設計し稼働させ所望の製品を得るまでの時間が短縮化された製品ライフサイクルに適合しないことから現実的でない．よって，後者で多品種少量生産を実現することが現実的な方法と言える．

1.1.2. バッチプロセスの概要

連続プロセスと異なる生産形態として，バッチプロセスがある．本研究では，連続プロセス，バッチプロセスを次のように定義する．

連続プロセス：

原料を連続的に設備に供給し，反応，蒸留，抽出，乾燥等の処理を連続して行い，目的の製品を連続的に生産するシステムを連続プロセスと呼ぶ． ■

バッチプロセス[20]：

あらかじめ定められた量の原料を一つ以上の装置を使い，有限時間の間に所定の一連の処理を行い，目的の製品を間欠的に生産するシステムをバッチプロセスと呼ぶ．また，バッチプロセスによって生産された製品をバッチと呼ぶ． ■

連続プロセスにおいては，反応条件や原料・触媒等を変更することで，一つの設備で種々の製品を生産する．一方，バッチプロセスにおいては，これらの条件変更に加え，使用する装置の種類や使用順序，または装置で実行される処理の順序を変更することが容易であり，一つの設備でさらに多くの製品を生産することが可能である．また，バッチプロセスは生産量の変更も連続プロセスに比べて容易である．このようなことから，バッチプロセスは多品種少量生産に適した生産形態と言える．

しかし，バッチプロセスの計画・設計・運用等に関する原理，技術の蓄積は，連続プロセスのそれに比べて格段に少ないのが現状である．この理由として，バッチプロセスが次に示す特徴を有していることが挙げられる[22]．

バッチプロセスの特徴：

- バッチプロセスは主に非定常運転である．
- 同一の設備でさまざまな種類の製品を生産する．
- 同一の装置で複数の処理を実行する．
- 複数の製品を同一の設備内で同時に生産する． ■

したがって，このような特徴を有するバッチプロセスの計画・設計・運用等は，未だに経験者の知識やノウハウ等に頼らざるをえない．しかし，今後，求められる製品は高機能化・多様化が進むことが予想され，それらを生産するプロセスも大規模化・複雑化が必要となる．よって，この種の生産形態の合理的な計画・設計・運用手法等の開発が期待されるところである．

1.1.3. 化学プロセスにおける異常対応

化学プロセスでは，安全性が高く，高い製品品質を維持した運転を実現するため，異常への対応は重要である[7]．化学プロセスにおいては，機器の老朽化や故障，ミスオペレーションや不純物のコンタミネーション等，さまざまな原因で異常が発生する．異常が発生した際には，次に定義する異常対応が重要となる．

異常対応：

次の(1)から(3)に至る，異常に係る一連の対応を本研究では異常対応と呼ぶ．また，(1)および(2)を合わせて**異常診断**と呼ぶ．

(1)プラント内で発生した異常を検出する（異常の検出）

(2)検出された異常の原因を特定する（異常の特定）

(3)検出・特定された異常の再発を確実に回避する（異常の回避） ■

異常対応においては，まず，プラント内で発生した異常に気付くことが必要となる．本研究ではこれを異常の検出と呼ぶ．ここでは，温度や圧力等の状態や弁の開度，ポンプの起動停止等の操作端の状態等を観測し，それらが正常な状態から逸脱しているか否かを判断することで，プロセスに異常が発生したことを検出する（気付く）ことが可能となる．一方，異常が初期の段階においては，プロセスに現れる状態変化は小さく，それに気付くことは困難であることが多い．このように，現実のプロセスにおいては異常が発生しているが，それが潜在的であるため検出できず，異常が伝搬し異常が拡大した後，それが顕在化することで検出可能となる場合がある．潜在的な異常が大きな異常，すなわち災害へと発展することを防ぐには，異常の早期な検出方法の確立が必要となる．例えば，化学プロセスにおける配管の詰まり（異常）は，潜在的に進行することが多く，配管が完全に詰まる，もしくは運転が困難となるレベルで流体が流れなくなるという異常へと発展する．この配管がブローダウンタンク送りの配管であれば，この異常は暴走

反応による爆発等の大きな異常（災害）へと発展する危険性がある．このため，異常を早期に検出し，その対策を行うことが重要となる．

次に，発生した異常の再発を回避するため，検出された異常の発生原因を特定する必要がある．原因の特定は，異常を検出した際のプロセスの状態に係る情報を用い，それを解析することが必要となる．本研究ではこれは異常の特定と呼ぶ．

そして，特定した異常の原因に応じた対策を行うことで異常の再発を回避することが可能となる．本研究ではこれを異常の回避と呼ぶ．異常の回避の方法は，機器の交換による“機器保全”，プロセス改良による“プロセス再設計”，そして制御により異常の回避を実現する“制御器の設計”の 3 つに大別できるものと思われる．機器保全においては，TBM（Time Based Management）や CBM（Condition Based Management）等の保全管理手法があり，弁，センサー，ポンプ，配管，フランジ等の機器の故障が発生する前にその機器を交換することを目的としている．特に CBM では定期検査や定期観測によって機器の異常を検出しその交換を行う手法であり，本研究で定義した異常対応と同等の取り組みと言える．プロセスの再設計においては，新たな機器の導入，プロセスフローの見直しや機器の材質の見直し等を検討することで，発生した異常の再発を回避する．制御器の設計においては，プロセス条件や操作手順の適正化を実現する制御を組み込むことで，異常の再発を回避する．

上で述べた配管の例において，異常の原因は様々なものが考えられる．重合や二量化反応による固体の析出が原因であれば，配管を流す流体の温度を適正にするという制御を加えることで，また高粘性流体の滞留が原因であれば，流体輸送時の流速（流量）を高めに設定するという制御を加えることで，それぞれ異常の回避が可能となる．また，腐食や配管に設置された弁の故障が原因であれば機器交換により異常の回避が可能となり，不溶物の詰まりであればフィルターを新たに設置することや使用する触媒の交換というプロセス再設計により異常の回避が可能となる．

本研究では，このうち制御器により異常の回避を実現する方法に着目する．以後，「異常の回避」は制御器による異常の回避を指すものとする．

上で述べたように，異常対応における異常の検出・特定・回避に係る情報は密接に関連している．よって，各々で得られた情報を互いにやり取りすることで，各々の問題を合理的に解決することができると考えられる．このことから，異常対応における一連の取り組みは統一された枠組みで解くことが重要と考えられる．一方，プロセスシステム

工学の分野においては，各意思決定問題を解決するにあたり，対象となるプロセスの動作を表す動的モデルを用い，そのモデルの上で方法論を検討することが行われる．

よって，異常対応における一連の問題を合理的に解決するためには，これらを統一された枠組みでとらえ，共通した動的モデルを用いた方法論を開発することが望ましいと考えられる．

1.1.4. 異常対応モデルを用いた異常対応

化学プロセスにおける異常対応を，動的モデルを用いて行うことを考えた場合，その動的モデルは次の要件を具備すべきである．

動的モデルが具備すべき要件：

(1) 異常を表す動作や状態が記述可能

(2) 発生した異常に対して，その原因が特定可能

(3) そのモデルの上で異常を回避するための制御が検討可能 ■

本研究では，次に定義する異常対応モデルのもとで，バッチプロセスにおける異常対応に係る一連の問題を合理的に解決する手法を開発することを目指す．異常対応モデルの概念を表したものが図 1-1 である．

異常対応モデル：

バッチプロセスにおける異常診断とその回避に係る一連の問題を解決するため，これらの問題に共通して用いることのできるバッチプロセスの動作を表す動的モデルを異常対応モデルと呼ぶ． ■

一方，バッチプロセスの動作は時々刻々と連続的に遷移するシステムの状態変化を概括的にとらえることで，「事象が非同期的，離散的に生起することによって，状態が不連続に遷移する」とする離散事象システムとしてとらえることができる[29]．そこで，本研究ではバッチプロセスの動作を離散事象システムとしてとらえることとした．これは，バッチプロセスは，定常運転および非定常運転を問わず離散的な状態遷移を伴うことが多いため，離散事象システムとしてとらえることで，その動作やそこで発生する異常を正確に表現できると考えたからである．また，バッチプロセスの制御は，温度や圧力等の状態変数を連続的に制御するレギュレトリ制御の他に，あらかじめ定められた手

順や条件のもとで望ましくない状態に陥らないように目標状態に遷移させるための手順制御や並行的に複数のバッチを製造するためのリソース管理に係る協調制御が重要となる．そして，これら手順制御や協調制御の設計のためには，バッチプロセスの動作を離散事象システムとして表すことが望ましいと考えたからである．以後，特別な記載がない限り，「制御器」は手順制御および協調制御のための制御器とする．

バッチプロセスの動作を離散事象システムとしてモデル化することで，先述した動的モデルが具備すべき要件のうち，1 は異常を表す事象や離散状態で，2 は状態間の因果関係で，3 は離散事象システムの制御問題で表現することができる．

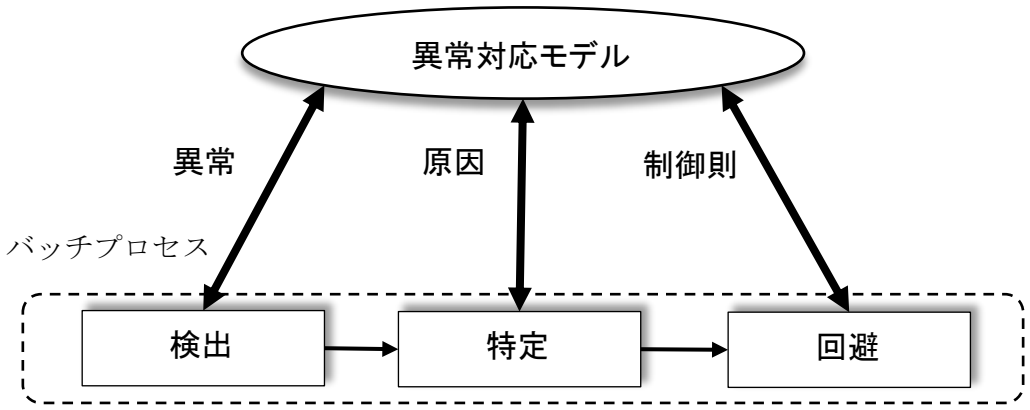


図 1-1　異常対応モデルの概念

1.2.　既往の研究

既往の研究においては，異常診断と異常の回避を一連の問題としてとらえているものはほとんどない．本項では，異常診断と異常の回避を分けてとらえ，既往の研究を述べる．

１．異常診断

化学プロセスにおける異常診断はモデルを用いる手法が一般に知られており，統計モデルと動的モデルの 2 つに大別される．統計モデルの例として，複数のプロセス変数を主成分分析等の統計解析手法にて組み合わせる（次元圧縮する）ことで，新たな統計量を導出し，それに基づき異常診断を行う手法がある．Kresta らは，過去の正常なプロセス変数群のデータを用いて，主成分分析法に基づく統計量（T^2 統計量や Q 統計量）

を算出し，異常が発生した際，これらの統計量がある正常の閾値から逸脱する仕組みを利用して異常を検出する手法を提案した[8]．しかし，統計モデルによる手法は，バッチプロセスのようにプロセスの進行とともにプロセス変数の正常範囲が変化する対象においては，その適用が困難であった．その後，バッチプロセスを対象とした手法として，Nomikos らはマルチウェイ主成分分析を用いた異常診断法を提案した[9]．しかし，統計モデルに基づく異常の特定は，関連性の高いプロセス変数の抽出は可能であるが，その原因を特定することはできない．また，プロセス変数間の組合せが変わると，新たなモデルの構築が必要となり，そのたびに正常なデータが必要となるため，現実のバッチプロセスへの適用が困難と言える．

一方，動的モデルの例として，Iri ら，および Shiozaki らが提案した符号付き有向グラフに基づく異常診断手法がある[10][11][12]．符号付き有向グラフは，プロセス変数を表すノードと，それらの間の因果関係を表すアークから構成される．異常の発生に伴い，異常を表すノードから，アークで因果関係が紐付けられた他のノードへとそれが伝搬してく様子がモデル化される．実プラントにおける異常伝搬の兆候とグラフから得られる異常伝搬モデルを比較することで，プロセスに生じた異常を検出することができる．また，その異常伝搬モデルの開始となる異常を表すノードを，プロセスで発生した異常の原因とすることで異常の特定ができる．離散事象モデルを用いた異常診断手法として，Sampath らは観測ができない事象を含む離散事象システムをオートマトンで記述し，そのモデルを用いて異常診断を行う手法を提案した[33][34]．彼らの手法は，異常を含まない（正常な動作のみで構成される）オートマトンをオブザーバーとして用い異常を検出する，また，現実のプロセスで発生する異常の伝搬による事象の発生形態と異常を陽に組み込んだモデルから得られる事象の発生形態を比較することで異常の原因を特定する枠組みであった．以後，離散事象モデルを用いた異常診断における研究は，彼らの手法を基礎としているものが多い．例えば，Mosterman[35]は temporal causal graph を用いたハイブリットシステムにおける異常診断のアルゴリズムを，Ashley と Holloway[36]はペトリネットに基づく状態システムモデルを用いた異常診断手法を，Bhattacharyya ら[37]はネットワークシステムにおける異常を有限オートマトンを用いて診断する手法を，Daigle ら[38]は離散事象システムに基づく異常診断システムを用いて連続システムにおける異常の特定する手法を，Roth ら[39]は同定モデルを用いた異常診断手法をそれぞれ提案した．また，Dotoli らは観測できない動作をモデル化する手法を提案し，これを異常解析に適用することを考案した[41]．

しかし，これら異常診断に係る多くの研究は，異常の早期発見を実現し，異常の伝搬を止めることで，異常が災害へ発展することを防止することを目的とするものが多く，本研究で考える異常の再発の回避という枠組みでの検討はほとんどない．

２．異常の回避

化学プロセスにおける異常の回避の方法論は，制御器の設計問題の一つと考えることができる．連続プロセスにおいては，その動作を連続時間システムとしてとらえ，制御目的にあったレギュレトリ制御器を設計する手法が数多く提案されている．一方，本研究のように対象を離散事象システムとしてとらえ，手順制御や協調制御の設計を考える場合，その研究は連続時間システムにおける制御器設計手法と比較すると少ないのが現状である．

離散事象システムでの制御器設計問題として，まず初めに，Ramadge と Wonham によって提案された離散事象システムの制御の枠組みとしての制御理論がある[13]．以前は離散事象システムの制御に係る問題は競合解消問題，排他制御問題，インターロック設計問題等のようにそれぞれ個別に検討されていたのが通常であった．そのなかで，この制御器設計に係る理論は，離散事象システムに対するこれらの問題を統一的な枠組みの中でとらえようとした画期的なものであり，以後，離散事象システムの制御問題を扱う研究の基礎となった．しかし，彼らの手法では，並行動作を陽に扱うことができないという欠点があった．Sanchez は制御器設計理論をバッチプロセスへ適用したが，やはり，並行動作は扱えないという欠点があった[14]．Moody らは，対象の動作をペトリネットにて表現し，状態に係る制御目的を満たすような制御器を設計する手法と提案した[15][45]．一方，橋爪，小野木らは，対象の動作をペトリネットの一つのサブクラスである条件／事象ネットにて表現し，動作に係る制御目的を満たすような制御器を設計する問題を定式化し，この問題の性質とその解法を示し，バッチプロセスへの適用を試みた[16][17][18]．

1.3. 本研究の目的

1.2 で示したように，バッチプロセスの異常対応に係る既往の研究においては，次の 3 つの点において十分な研究がなされていないように思われる．

既往の研究の問題点：

(1) 異常の診断とその回避は異なるモデルを用いて別々に解かれており，それらを統一的に扱う枠組みが確立されていない.

(2) 現実のバッチプロセスは観測できない状態や制御できない動作を含んでおり，それらを含むプロセスにおける異常の回避の方法論が確立されていない.

(3) 現実のバッチプロセスは不確実性を含んでおり，異常の回避も不確実性を考慮することが望まれているが，その方法論が確立されていない. ■

(1)に関して，1.2 で示したように，従来，異常の診断とその回避は異なる研究として取り扱われ，別々のモデルを用いて解かれていた．しかし，1.1 で述べたように，異常の診断と異常の回避は密接に関わっており，これらを同一のモデルを用い，互いに関連付けて解くことで，より合理的な解が得られることが期待できる.

(2)に関して，現実のバッチプロセスにおいては，組成や粘度のような“観測ができない状態”や，反応や分離のような物質そのものが自発的に作用してプロセスを異なる状態へと遷移させる“制御できない動作”が頻出し，これらの要素の存在が異常の回避を困難とさせている. Moody ら[45]は制御できない動作を含む制御器の設計手法を検討しているが，観測できない状態を考慮していない．よって，現実で頻出する観測できない状態と制御できない動作を同時に含むプロセスにおける異常の回避の方法論を確立することは重要と言えよう.

(3)に関して，バッチプロセスは柔軟性に富んだ生産形態であるため，生産量，生産品目の変更等の生産計画の変更が頻繁に行われる．また，プロセスのもつ非線形性や非定常運転に伴う動作・状態間の因果関係の変化等があり，これらをプロセスのもつ不確実性とするならば，異常の回避においても不確実を含むプロセスを対象とした，将来予測に基づく異常の回避の方法論が必要と考えられる．しかし，従来，これに係る研究は十分に行われていなかった.

そこで本研究では，プロセスシステム工学的な視点から，バッチプロセスにおける異常の診断とその回避のための合理的な方法論を確立することを目的とし，次の 3 つの目標の達成を目指す.

本研究の具体的な目標：

[1] 異常の診断とその回避を統一的に扱う方法論を確立する．

[2] 不可観測な状態，不可制御な動作の要素を含むプロセスを対象に，異常の回避の方法論を確立する．

[3] 不確実性の要素を含むプロセスを対象に，将来の予測に基づき異常を回避する方法論を確立する． ■

具体的な目標[1]から[3]は，既往の研究の問題点(1)から(3)にそれぞれ対応している．すなわち，本研究では，既往の研究における問題を解決することで，バッチプロセスにおける異常の診断とその回避のための合理的な方法論を開発する．

1.4. 本論文の構成

本論文の構成は以下の通りである．第 2 章では，バッチプロセスにおける異常を明確にし，異常対応モデルの上での異常の表現方法を示す．そして，異常対応モデルの現実的な構築手法を提案する．第 3 章では，異常対応モデルのもと，異常診断を合理的に行う手法を開発する．第 4 章および第 5 章では，原因の特定された異常の再発を回避することを目的とする制御器の設計問題を定式化し，その解法に係る検討を行う．第 6 章では，観測できない状態や制御できない動作を含むバッチプロセスを対象とした異常の回避の方法を検討する．第 7 章では，不確実性を伴うプロセスを対象とした，将来の予測に基づき異常を回避する方法を検討する．最後に第 8 章で本論文について総括する．

第2章.異常対応を目的としたバッチプロセスのモデル化

本章においては，バッチプロセスで発生する異常を定義し，本研究における異常のとらえ方を明確にする．次に，バッチプロセスの動作を離散事象システムとしてとらえた際の，異常のモデル化について検討する．また，異常対応モデルを構築する方法についての検討を行う．

2.1. はじめに

第 1 章で述べたように，本研究においては，バッチプロセスにおける異常の診断とその回避のための合理的な方法論の開発を目的としている．しかし，化学プロセスにおける異常の概念は，従来は曖昧であり，それを一般化された形で取り扱うことが困難であった．本章においては，現実の化学プロセスにおける異常を明確に定義すること，そして，異常対応モデルの上で，定義した異常の表現方法に関する検討を行う．

現実のバッチプロセスにおいては，“温度，圧力が反応暴走を誘発する値まで到達する”，“温度，液面が急に変動する”，“弁の開度が全閉であるにも係わらず弁の取り付けられた配管の流量が流れる”，“熱交換器における加熱弁と冷却弁が同時に全開状態となっている”，“反応が進行していると考えられる状況で温度が変化しない”等の異常が発生する．バッチプロセスで発生する異常はその特徴に応じて分類することで，それぞれの特徴に応じた異常対応の検討が可能となると考えられる．そして，これらの異常対応に係る問題は，異常を動的モデルの上で表現し，そのモデルを用いることで合理的な方法が導かれることがある．

本章の構成は次の通りである．2.2 では異常の定義を行う．2.3 および 2.4 では異常のモデル化に係る検討を行い，異常対応モデルにおける異常の表現方法を検討する．2.5 では，異常対応モデルの現実的な構築方法として，未経験の異常の発生に伴い，それを表す異常を異常対応モデルに都度組み込んでいく手法について検討する．

2.2. 用語の定義

本論に入る前に，本研究で用いる共通的な用語を以下に定義する．

プラント：

反応器，蒸留塔等の装置および弁，センサー等の機器から構成され，生産プロセスを実現する現実の設備をプラントと呼ぶ．バッチプロセスに用いるプラントを特にバッチプラントと呼ぶ． ■

プロセス変数：

流量，液面，圧力，温度，組成，密度，粘度等の物理量，弁，ポンプ等の機器からの信号値，さらにはそれらを組合せた変数をプロセス変数と呼ぶ． ■

プロセス[21]：

原料を物理的あるいは化学的に変換し，目的とする物質またはエネルギーを生産するという全体的目的をもつシステムをプロセスと呼ぶ． ■

プロセスの動作：

目的の生産物を生産するために実行される一連の処理をプロセスの動作と呼ぶ．処理は，供給，加熱，撹拌，冷却，弁の開閉，ポンプ起動停止，機器のスイッチ入切等，外部からの操作でプラントに加えられるものと，反応，分離等，物質そのものが自発的に作用して生じるものがある．本研究においては，処理を動作の一部とし，両者を区別せずに用いる． ■

プロセスの状態：

プロセス変数の状態の組合せをプロセスの状態と呼ぶ． ■

化学プロセス：

化学製品を生産するシステムを化学プロセスと呼ぶ．本研究では，その生産形態として，連続プロセス，バッチプロセス，連続・バッチの混合プロセスの 3 つタイプがあるとする．また，化学プロセスにおいて成立することはバッチプロセスにおいても成立するものとする． ■

以上の用語のもと，本研究では，化学プロセスの異常を次のように定義する．

異常：

(1)異常状態

プロセスが正常に動作する限り，決して現れることがない状態

(2)異常動作

プロセスの状態を正常状態から異常状態に遷移させる動作 ■

この定義により，バッチプロセスで発生する異常は，異常動作によって引き起こされ，その状態は正常状態から異常状態へと遷移すると捉えることができる．以後，異常状態と異常動作を単に異常と呼ぶこともある．

次に，本研究では化学プロセスに現れる異常（状態）を次の二つのタイプに分類する．

異常の分類：

(1)異常 F1

1 つのプロセス変数が，それ単独で示す異常

(2)異常 F2

個々のプロセス変数は正常であっても，それらを組として見たとき現れる異常 ■

異常 F1 の例として，“ある圧力が閾値よりも高い圧力となる異常”，“ある温度の上昇速度が閾値よりも高い温度上昇速度となる異常”等が挙げられる．異常 F1 は，その定義から分かるように，単独のプロセス変数の状態から正常・異常の判定が可能である．一方．異常 F2 は，複数のプロセス変数を組みとして見たときに初めて現れる異常である．異常 F2 の例として，“反応器へ原料の仕込みを行う際，原料を供給する弁と排出する弁の開度がともに全開である異常”，“急激な発熱が予想される反応プロセスの間，冷却用ユーティリティ弁の開度が全閉である異常”，“流量の調整弁の開度が全開であるが，その弁が取り付けられた配管の流量計の指示値が 0 である異常”等が挙げられる．このような異常の検出においては複数のプロセス変数間で満たすべき制約に着目し，制約の成否で異常を検出する必要がある．

バッチプロセスにおける異常動作と異常状態の一例を図 2-1 に示す．バッチプロセスにおいては，ポンプのキャビテーションや配管の漏れ，詰まり等の異常動作に伴いプロセスの状態が異常 F1 や異常 F2 で示す異常状態へと遷移する．

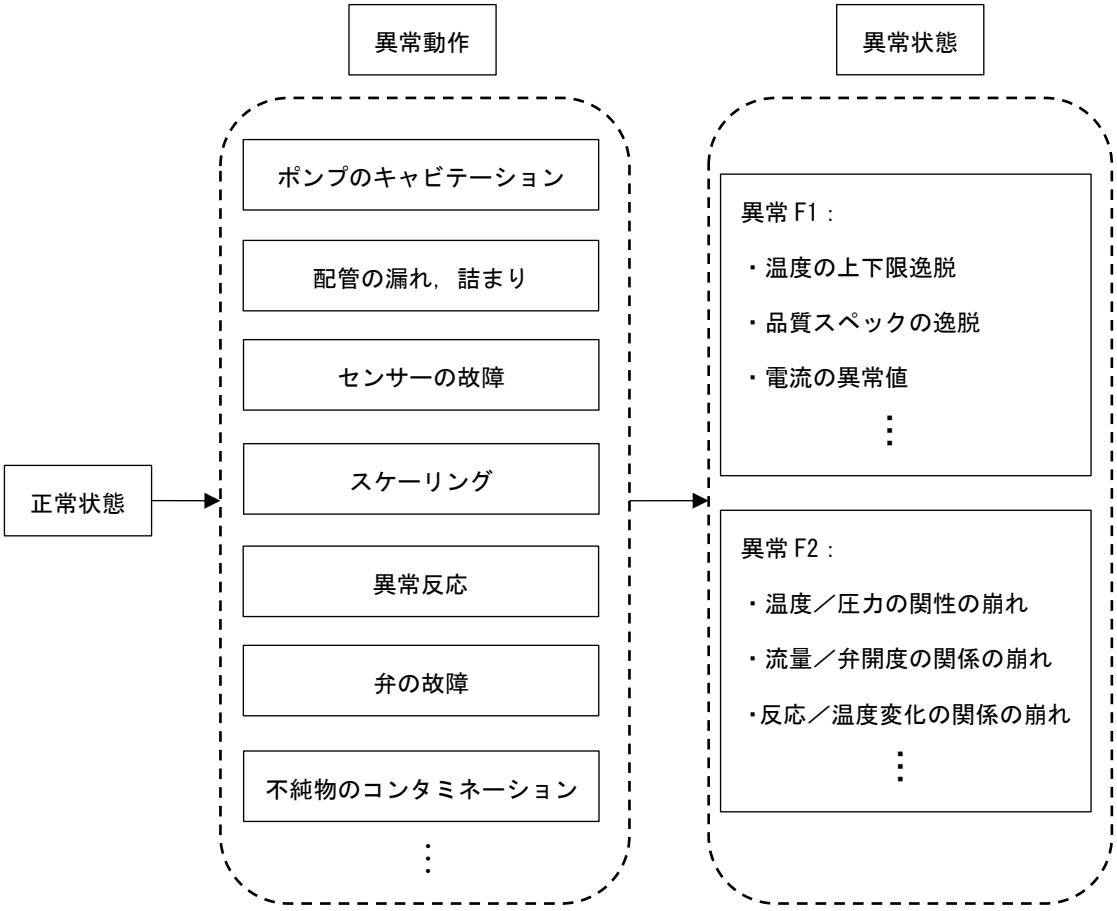


図 2-1　バッチプロセスにおける異常動作，異常状態の一例

一般に，異常 F1 の検出は比較的容易であり，閾値の設定方法においてもプロセスの特徴や条件，使用する機器の耐熱や耐圧等の仕様によって定めることが可能である．一方，異常 F2 に関しては異常 F1 と比較して，十分な検討がなされていないのが現状である．したがって，異常 F2 を検出する方法を確立することができれば，今まで異常と気づかなかった潜在的な異常に対しても，それを検出できる可能性が高まると考えられる．

本研究においては，異常 F2 をとらえるため，次の動作制約を定義する．

動作制約：

プロセスが正常な動作をする限り複数のプロセス変数が満たすべき条件　■

これより異常 F2 の検出は，バッチプロセスの動作に合わせて状態が遷移するたび，プロセスの状態が動作制約を満たしているか否かを判定することで可能となる．

ここで，“流量の調整弁の開度が全開であるが，その弁が取り付けられた配管の流量計の指示値が通常に比べで少ない現象”がみられたとしよう．この例においては弁開度と流量，それぞれを単独で見ると正常な範囲であり異常とはならないが，これらを組として見ると異常であることが分かる．よって，弁開度と流量の間には動作制約が存在し，それが不成立となっていることが考えられる．そして，この例ように複数のプロセス変数を組として見ることで，単独のプロセス変数のみを観測することでは検出できなかった潜在的な異常を検出することが可能となる．

一方，動作制約はバッチプロセスの規模が複雑化，大規模化すると，それ自体を求めることが難しくなる．よって，動作制約による異常の検出を行うためには，動作制約の効率的な求め方が必要となる．

2.3. 離散事象システムにおける異常

バッチプロセスの計画・設計・運用にあたっては，“反応器内の温度がある温度に到達したならば触媒を添加し，次にさらにある温度に到達したならば撹拌機の回転数を上げ，反応を進行させる”，”液面がある値に到達したならば出口弁を開けて抜き出しを開始し，液面がさらに別の値に到達したならば出口弁を閉じて抜き出しを完了する”等のように，プロセスの状態が所定の値に到達することによって動作が切り替わることが多い．このような場合には，途中の状態変化を無視し，プロセスの動作を概括的にとらえることができる．このように，連続的に変化する状態量を幾つかの代表値で離散化し，離散化された状態量が変化する状態変化のみに着目すると，そこで発生する状態変化もまた時間に関して離散的に発生する状態変化とみなすことができる．このような状態変化は事象と呼ばれ，システムの状態がこのように変化するシステムは離散事象システムと呼ばれる[29][44].

事象：

瞬間的に発生するとみなすことができるシステムの動作，状態変化は事象と呼ばれる．また，システムを離散事象システムとしてとらえた時，瞬時的に発生し，ある状態から別の状態へと遷移させることがらを事象と呼ぶこともある． ■

離散事象システム：

事象が離散的に生起することにより状態が不連続に遷移するシステムは離散事象システムと呼ばれる．■

現実のバッチプロセスにおける動作，状態は，離散事象システムの上でそれぞれ事象の発生形態および離散的な状態（の組合せ）としてとらえることができる．本研究では，現実のバッチプロセスの動作を離散事象システムとしてモデル化し，その上で，異常対応に係る問題の解決を図る．2.2 で定義した異常は離散事象システムの上で，次のようにとらえることができる．

離散事象システムでの異常のとらえ方：

⑴異常状態

システムが正常に動作する限り，決して現れることがない離散的な状態

⑵異常事象

システムの状態を正常状態から異常状態へと遷移させる事象 ■

バッチプロセスの動作を離散事象システムとしてとらえ，異常状態および異常事象を組み込んだ動的モデルを構築する．このモデルを活用することで，ある異常を検出し，その異常状態へと遷移させた異常事象を特定するという一連の異常診断が可能となる．また，異常事象の発生の回避や異常状態への遷移の回避を実現する制御器は，それを離散事象システムの制御問題に帰着させることで設計可能となる．

2.4. ペトリネットを用いた離散事象システムの表現方法

バッチプロセスの動作を離散事象システムとしてとらえ，異常対応に係る問題をそのモデルの上で考える場合，離散事象システムを何かしらの表現形式で記述する必要がある．バッチプロセスを取り扱うためには，その表現形式においては，次に示すバッチプロセスの動作の特徴を正確に表現できることが重要である．

バッチプロセスの動作の特徴：

⑴逐次的動作

ある動作（処理）を行うことにより，別の動作（処理）が順に行われていく動作であり，動作（処理）の実行の間に直接の因果関係が存在する．

⑵並行的動作

複数の動作（処理）が，互いに独立して行われていく動作であり，動作（処理）の実行の間に因果関係は存在しない.

⑶非決定的動作

複数の動作（処理）が実行可能であるが，そのうちのどれを実行するか，あらかじめ定まっていない. ■

また，異常対応に係る問題の合理的な解法を開発する際，離散事象システムの表現形式に由来する解析手法を利用することが重要となると考えられる．そして，対象とするプロセスが大規模化，複雑化した場合でもモデル化可能な表現形式が望ましい.

そこで本研究においては，逐次的・並行的・非決定的動作を陽に表現でき，表現形式に由来する解析手法を多く備え，優れた階層表現を有するペトリネットが離散事象システムの動作を表す表現形式として最も適していると考え，このペトリネットを用いることとした．ペトリネットがもつこれらの特徴は，複数のプロセス変数が組となって示す異常の表現を容易とし，ペトリネットの解析手法に基づく制御器設計を可能とし，階層構造を利用することで大規模かつ複雑なバッチプロセスのモデル化を容易とする.

ペトリネット[23][24]は，プレース，トランジションの 2 種類のノードをもつ 2 部有向グラフである（ペトリネットの用語，表記に関する詳細は付録 1 に記載する）．ペトリネットにおいては，離散事象システムにおける事象はトランジションで，状態はプレースの集合上のトークンの配置であるマーキングで，動作はトランジションの発火形態で表現される.

よって，2.2 で述べた異常の分類はペトリネットの上で次のように記述することが可能である.

ペトリネット上での異常の分類の記述の仕方：

⑴異常 F1

一つプレースに置かれたトークン数の正常値からの逸脱

⑵異常 F2

複数のプレースに置かれたトークン数の正しい組合せからの逸脱 ■

また，異常状態を表すマーキングを異常マーキング，異常事象を表すトランジションを異常トランジションとする.

バッチプロセス，離散事象システム，ペトリネットの関係を示したものが図 2-2 である．バッチプロセスの動作を離散事象システムとしてモデル化する．次に，離散事象システムをペトリネットで記述することでペトリネットモデルを得ることができる．本研究では，このペトリネットモデルを異常対応モデルとする．

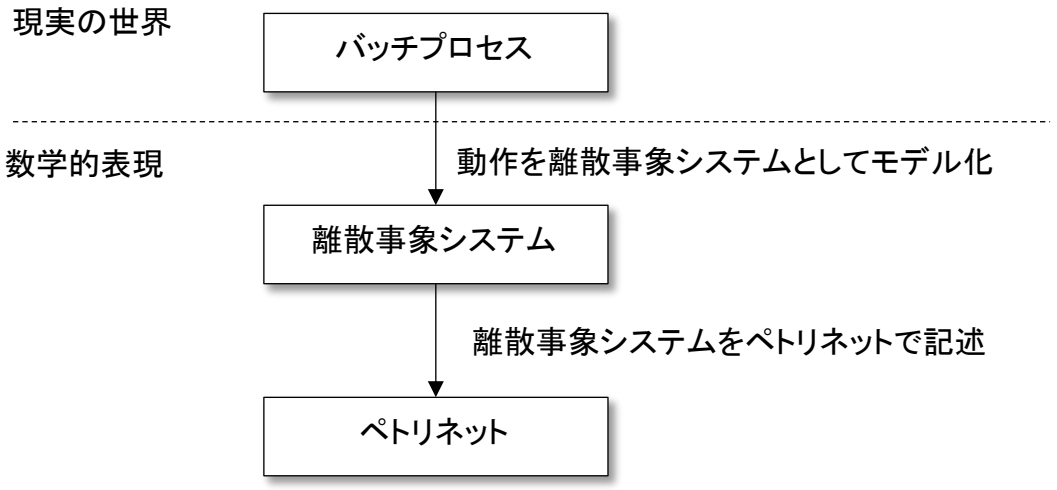


図 2-2　バッチプロセス，離散事象システム，ペトリネットの関係

2.5.　異常対応モデルの構築手法

2.5.1.　異常対応モデル構築の基本的な考え方

第 1 章述べたように，異常対応に係る問題を動的モデルを用いて解く場合，そのモデルの上で異常を表す動作や状態が記述されていることが必要となる．そして，それを矛盾なく行うためには，そのバッチプロセスで将来起こりうる異常をすべて予測し，それをあらかじめモデルに組み込むことが必要となる．しかし，将来起こりうるすべての異常を予測することは非常に困難と考えられる．そこで本研究では，次に示す動的モデルの逐次更新に係る基本的な考えを異常対応モデルにおいても適用することとする．

動的モデルの逐次更新に係る基本的な考え方：

Step 1:

正常でない状態が観測されるたびに，観測された状態が経験済みの状態か未経験の状態か確認する．

Step 2:

もし，その状態が未経験の状態であれば，現在のモデルに正常でない動作をモデル化し組み込む. ■

この考えを異常対応モデルに用いる場合，次の二段階の手順となる：

(1) 異常を検出する

(2) 未経験の異常を組み込むことで，モデルを更新する

この手順は異常対応モデルを構築する現実的な手段となると考えられる．よって，本研究における異常対応モデル構築手法は，まずバッチプロセスの正常な動作を表すモデルを構築する．そして未経験の異常が検出されるたびに，現在のモデルに検出された未経験の異常を組み込み，異常対応モデルを更新していく．このようにモデルを逐次更新することで，取り扱うことのできる異常を増やしていくこととする．

2.2 で述べたように，本研究では動作制約の成否によって，異常の検出を行う．よって，上記の手順は次のように具体化される．

(1)-1 バッチプロセスの状態が動作制約を満たしているか確認する．

(1)-2 動作制約を満たしていない場合，その状態が未経験の異常か否かを確認する．

(2) 未経験の異常であれば，未経験の異常をモデルに組み込み，モデルを更新する．

2.5.2. 異常対応モデルの構築手順

2.5.1 では，異常対応モデル構築の基本的な考えを示した．本項では，モデルの更新に係る検討を行う．2.4 で示したように，本研究では，異常対応モデルとしてペトリネットモデルを用いることとした．よって，本項では異常対応に用いるペトリネットモデルの更新方法を検討する．

本研究におけるモデルの更新手法を説明する前に，ペトリネットにおける状態の遷移，すなわちマーキングの遷移を表す状態遷移方程式について簡単に述べる．ペトリネットモデルNにおいて，初期マーキング$\boldsymbol{M}_0$から到達可能なマーキング$\boldsymbol{M}$は，(2.1)式の状態遷移方程式によって記述することができる．

$$\boldsymbol{M} = \boldsymbol{M}_0 + \boldsymbol{D}\boldsymbol{x} \tag{2.1}$$

ここで，mをNのプレース数，nをNのトランジション数とすると，$\boldsymbol{M}$および$\boldsymbol{M}_0$は$m \times 1$ベクトルで，$\boldsymbol{D}$はNの$m \times n$接続行列である．また，接続行列の要素$D(i,j)$はトランジションt_jが一回発火することによって生じるプレースp_iのトークン数の変化である．また，

$\boldsymbol{x}$はNの発火回数ベクトルであり$n \times 1$のベクトルで表され，各要素$x(j)$はトランジションt_jが$\boldsymbol{M_0}$から$\boldsymbol{M}$に至るまでに発火した回数を表す非負の整数である．

ペトリネットのマーキングはバッチプロセスの状態を表したものであるから，状態遷移方程式は，バッチプロセスの実行可能な動作によって初期状態から到達可能な状態を表している．

次に，未経験の異常が発生した際の異常対応モデルの更新手法を述べる．時刻τ_{i-1}から時刻τ_iにおけるバッチプロセスの動作を表すペトリネットモデルN_iの状態遷移方程式は，(2.2)式で表される．

$$\boldsymbol{M}_i = \boldsymbol{M}_{i-1} + \boldsymbol{D}_i \boldsymbol{x}_i \tag{2.2}$$

ここで，$\boldsymbol{M}_i(\boldsymbol{M}_{i-1})$は$N_i$の時刻$\tau_i(\tau_{i-1})$におけるマーキングで，$\boldsymbol{D_i}$は$N_i$の接続行列で，$\boldsymbol{x}_i$は$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$へ遷移させる発火回数ベクトルである．また，$m_i$を$N_i$のプレース数，$n_i$を$N_i$のトランジション数とすると，$\boldsymbol{M}_i$は$m_i \times 1$ベクトルで，$\boldsymbol{D_i}$は$m_i \times n_i$行列で，$\boldsymbol{x}_i$は$n_i \times 1$ベクトルである．

このとき，異常対応モデルの更新手法は次のような手順で行うことができる．

異常対応モデルの更新：

Step 1: （異常の検出）

新たな状態が観測されるたびに，$N_i \coloneqq N_{i-1}$とし，時刻τ_iにおける状態$\boldsymbol{M}_i$が動作制約を満たすか否かを確認する．満たしている場合，$i := i + 1$とし，Step1 を繰り返す．満たさない場合は，Step2 へと移動する．

Step 2: （未経験の異常の確認）

$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$への遷移がN_iから矛盾なく説明できる，すなわち経験済みの異常を表す異常トランジションおよびその発火によって遷移した異常マーキングならば，$i := i + 1$とし，Step1 へと戻る．説明できない場合，未経験の異常が発生したとし，Step3 へ移動する．

Step 3: （未経験の異常のモデル化）

$\boldsymbol{M}_{i-1}$から$\boldsymbol{M_i}$へと遷移させる未経験の異常を表す異常トランジションをN_iに追加し，モデルを更新する．$i := i + 1$とし，Step1 へと戻る． ■

上記の異常対応モデルの更新は，Step 2 で検出された未経験の異常をどのようにして現在モデルに組み込むかに大きく依存している．本研究ではベクトル演算の考えに基づき，異常のモデル化を行う．

未経験の異常のモデル化：

Step2 において，「$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$への遷移がN_iから矛盾なく説明できる」とは，(2.2)式において，$\boldsymbol{x}_i$が非負整数解をもつことである．よって，$\boldsymbol{x}_i$が非負整数解をもたない場合，現在のネットN_iでは，$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$への遷移が説明できない．すなわち(2.3)式となる．

$$\boldsymbol{M}_i \neq \boldsymbol{M}_{i-1} + \boldsymbol{D}_i\boldsymbol{x}_i \tag{2.3}$$

ここで，$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$への遷移は未経験の事象の発生によるものとする．(2.3)式の代わりに，(2.4)式を考える．

$$\boldsymbol{M}_i = \boldsymbol{M}_{i-1} + \boldsymbol{D}_i\boldsymbol{x}_i + \boldsymbol{b}_i \tag{2.4}$$

$\boldsymbol{b}_i$は(2.3)式の右辺を左辺と等しくする役割で導入される$m_i \times 1$ベクトルである．言い換えると$\boldsymbol{b}_i$は，N_iのプレースに置かれたトークンの数を(2.4)式が成り立つよう調整するトランジションの役割を担う．よって，このような手順でモデルを更新すると，未経験の事象を表すトランジションが追加されることになり，モデルに含まれるトランジションの数は徐々に増える．一方で，プレースの数はモデルの更新によって変わらない．(2.4)式で導入した$\boldsymbol{b}_i$は(2.5)式のように書き直すことができる．

$$\boldsymbol{b}_i = \boldsymbol{D}_i'\boldsymbol{x}_i' \tag{2.5}$$

ここで，l_iを新たに発生した未経験の事象の数とすると，接続行列$\boldsymbol{D}_i'$は$m_i \times l_i$ベクトルで，発火回数ベクトル$\boldsymbol{x}_i'$は$l_i \times 1$ベクトルである．(2.5)式を(2.4)式に代入することで，新たな状態遷移方程式(2.6)を得ることができる．

$$\boldsymbol{M}_i = \boldsymbol{M}_{i-1} + [\boldsymbol{D}_i \quad \boldsymbol{D}_i'] \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{x}_i' \end{bmatrix} \tag{2.6}$$

更新されたN_iはプレース数m_i，トランジション数$(n_i + l_i)$のペトリネットとなる．ここで，$[\boldsymbol{D}_i \quad \boldsymbol{D}_i']$は更新された$N_i$の$m_i \times (n_i + l_i)$接続行列で，$\begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{x}_i' \end{bmatrix}$は新たな発火回数ベクトルであり，$(n_i + l_i) \times 1$ベクトルで表される．モデル更新式は，次の(2.7)式，(2.8)式となる．

$$\boldsymbol{D}_i := [\boldsymbol{D}_i \quad \boldsymbol{D}_i'] \tag{2.7}$$

$$\boldsymbol{x}_i := \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{x}_i' \end{bmatrix} \tag{2.8}$$

このような手順で未経験の事象をモデルに組み込む．この“未経験の事象”を“未経験の異常”とみなせば，本手法は未経験の異常トランジションの組み込み，すなわち異常対応モデルの更新方法となる． ■

バッチプロセスの効率的な運用にあたっては，複数のバッチを並列に実行することで製品リードタイムを短くすることが必要となり，これはバッチプロセスの動作の並行性を十分に活かした運転を実現することで達成される．一方で，製品の多様化に伴い，同一プラントで製造するバッチの組合せも膨大となり，当初想定していない未経験の異常が発生する可能性は高くなると考えられる．例えば，“あるバッチ A とバッチ B はバッチ切り替え時に洗浄を行うことなく同じ反応器で生産すること可能であったが，品質改善のためバッチ A に微量添加剤を加えたバッチ A’とバッチ B は，微量添加物と B の分解反応による異常が発生した．”この例のように，製品の多様化は，未経験の異常の発生を誘発する要因となりうるため，提案した異常対応モデルの更新法は，多様化する消費者嗜好に対応するための手法であるとも言える．

また，バッチプロセスでは同じ種類のバッチを同じ手順で製造することを繰り返すため，モデル更新によって未経験の異常を組み込み，そのモデルのもとで，その異常の回避を実現する制御器が設計できれば，次のバッチを生産する際の異常の再発回避という観点からも，本手法は効果的に働くことが期待される．異常の回避に係る検討は第 4 章で行う．

一方，モデル更新に関する提案手法においては，(2.5)式で示す未経験の異常を表すトランジションの発火回数ベクトルと接続行列の分解の解は一意に決まらない．よって，どうのように分解し，どうような異常をモデル化するかは，その異常の特性やプロセスの情報等，現実のプロセスに関する知識が必要となると考えられる．この課題に対するシステム工学的からの検討は，今後の課題である．

2.6. まとめ

本章では，従来その概念が曖昧であった異常を一般化された形で定義し，定義した異常の動的モデルの上での表現方法を明らかにした．また，異常対応を行うための動的モデルの構築手法に係る検討を行った．

2.2 では，化学プロセスにおける異常を明確にし，異常の分類をおこなった．本研究では，プロセスが正常な動作をする限り，決して現れることない状態を異常状態とし，異常動作の発生によりバッチプロセスの状態が正常状態から異常状態へと遷移するとした．そして，異常の分類を，プロセス変数が単独で示す異常 F1 と，個々のプロセス変数が正常であっても，それらを組として見たとき現れる異常 F2 に分類した．化学プ

ロセスにおいては，異常 F1 に係る検討は進んでいるが，異常 F2 に係る検討は十分になされていないのが現状である．異常 F2 の検出のため，本研究においては，プロセスが正常な動作をする限り複数のプロセス変数が満たすべき条件として動作制約を定義し，動作制約の成否によって異常を検出する方法を提案した．2.3 では，バッチプロセスの動作を離散事象システムとしてとらえ，その上で，2.2 で定義した異常のとらえ方を明らかにした．2.4 では，離散事象システムの表現形式として，バッチプロセスの動作の特徴である逐次的動作・並行的動作・非決定的動作を陽に表現でき，固有の解析手法を備え，階層表現に優れたペトリネットを用いることを提案した．そして，ペトリネットを異常対応モデルとすることとし，この上で，定義した異常の表現方法を明らかにした．

異常対応モデルを用いて，異常対応に係る問題の解決を図る場合，第 1 章で示したように異常の動作や状態を表すモデルを異常対応モデルに組み込む必要がある．しかし，あらかじめすべての異常を予測して組み込んでおくことは現実的ではない．2.5 では異常対応モデルの現実的な構築手法として，バッチプロセスの状態を観測し，未経験の異常が検出されるたび，その異常を随時モデルに追加することで，モデルを更新していく手法を提案した．本手法においては，(2.5)式で示す未経験の異常を表すトランジションの発火回数ベクトルと接続行列の分解の解は一意に決まらない．よって，どうのように分解し，どうような異常をモデル化するかは，その異常の特性やプロセスの情報等，現実のプロセスに関する知識が必要となると考えられる．この課題に対するシステム工学的からの検討は，今後の課題である．

一方，2.5 で示した異常対応モデルの更新の Step1 では，バッチプロセスの状態が動作制約を満たすか否かを確認することで，バッチプロセスの異常を検出している．すなわち，異常の検出のため，何らかの方法で動作制約を求めることが必須である．

第 3 章においては，バッチプロセスの異常の検出のために必要な動作制約を，ペトリネットの解析手法から導く方法および異常診断の方法を検討する．

第3章.バッチプロセスにおける異常の診断

第 2 章においては，バッチプロセスにおける異常を明確にした．また，バッチプロセスの動作を離散事象システムとしてとらえ，ペトリネットを用いて表現すること，およびペトリネットの上で異常の表現方法および異常対応モデルの構築方法について検討した．本章においては，ペトリネットで表現される異常対応モデルを用いた異常診断，すなわち，異常の検出および異常の特定に関する検討を行う．

3.1. はじめに

第 1 章で示したように，バッチプロセスは主に非定常運転であり，複数のバッチを同時並行して生産し，一つの機器で複数の動作を行うという複雑な生産形態である．そのような複雑な生産プロセスを安全に運用するため，異常診断は重要な役割を果たすが，プロセスが大規模化・複雑化するに従い，そこに現れるプロセス変数の組合せが膨大となり，また，それが頻繁に変化するため，その対応が困難となっていた．本章においては，バッチプロセスにおける異常診断を効率的に行う方法を開発する．

本研究においては第 2 章で示したように，異常を，

異常 F1：1 つのプロセス変数が，それ単独で示す異常

異常 F2：複数のプロセス変数が組になって初めて示す異常

の 2 つのタイプに分類した．

異常 F1 に関しては，バッチの種類や使用するプラントに応じ，プロセス変数単独でその状態に係る閾値が定まる．そして，この閾値を逸脱した状態を異常とし，それを検出することが可能である．

一方，異常 F2 に関しては，バッチプロセスにおいては複数の動作を同時並行して行うため，プロセス変数の組合せが膨大であり，その検出は容易ではない．このため，現実の生産現場においてはバッチプロセスの特徴である複数の動作を同時並行して行うことはせず，あらかじめ決められた時期に，決められたバッチのみを，単純化された手順に従い生産し，目標バッチ数の生産完了後，次の種類のバッチへ移行するということが多くみられる．このように生産形態を単純化することは，この種の異常を防ぐ方法の

一つではあるが，バッチプロセスの特徴を十分に活かせておらず，プラントを効率的に運用できているとは言い難い．消費者嗜好の多様化に対応した競争力ある生産を実現するため，安全性を確保した上で，バッチプロセスの特徴を活かした効率的な生産を行うことが今後不可欠になると考えられる．安全性を確保するためには，異常 F2 の早期な検出と，その異常の発生の原因を特定する，異常診断が特に重要となる．

本研究においては第 2 章に示したように，異常 F2 の検出のため，プロセスが正常な動作をする限り複数のプロセス変数が満たすべき条件である動作制約に着目し，動作制約の成否によって異常の検出を行う方法を提案した．また，検出された異常の原因の特定では，バッチプロセスを正常状態から異常状態へと遷移させた異常動作をその原因とし，動的モデルを用いてそれを特定する方法が考えられる．以上のことから，本研究における異常診断の考え方は次の通りである．

異常診断の考え方：

(1)異常の検出

観測データが得られるたびに，動作制約の成否を確認する．動作制約が成立しない場合，対象となるバッチプロセスに異常が発生したと見なす．

(2)異常の特定

動的モデルを用い，正常状態から異常状態へと遷移させた異常動作を求め，それを異常の発生の原因とする． ■

このことから，バッチプロセスにおける異常診断を実現するために必要なことがらは次のとおりである．

バッチプロセスにおける異常診断を実現するために必要なことがら：

(1)異常動作と異常状態を異常対応モデルに組み込む．

(2)動作制約を効率的に導出する． ■

(1)に関しては，2.5 で示したように，未経験の異常が発生するたびにモデルを更新する手法を開発した．(2)に関しては，現実のプロセスの規模，複雑さが増すと，動作制約を経験やノウハウから発見的に求めることが困難となる．そこで，本章では動作制約の効率的な導出および異常の特定に係る検討を行う．本章の構成は次の通りである．3.2 では，バッチプロセスの正常な動作をペトリネットモデルで表現することで，動作制約

が効率的に導出できることを示す．また 3.3 では，本研究で開発したペトリネットモデルに基づく異常診断の方法を示す．

3.2. ペトリネットに基づく動作制約の導出

本研究においては第 2 章で述べたように，動作制約の成否に基づき異常の検出を行う手法を提案した．本手法を用いるためには，動作制約を何らかの方法にて導出する必要がある．

動作制約を導出する方法として，本研究では，ペトリネットにおけるプレースインバリアント解析に着目した方法を開発した．ペトリネットモデル$\boldsymbol{N}$において，次の同次方程式を満たす$1 \times m$のベクトル$\boldsymbol{y}$をプレースインバリアントと呼ぶ．ここで，$\boldsymbol{D}$はNの$m \times n$接続行列である（m, nをNのプレース数，トランジション数とする）．

$$\boldsymbol{yD} = 0 \tag{3.1}$$

状態遷移方程式(3.2)（(2.1)式と同じ）に$\boldsymbol{y}$を両辺に左から掛けると，次の(3.3)式が得られる．

$$\boldsymbol{M} = \boldsymbol{M}_0 + \boldsymbol{Dx} \tag{3.2}$$

$$\boldsymbol{yM} = \boldsymbol{yM}_0 = \mathrm{c} \tag{3.3}$$

ここで，$\boldsymbol{M}$はNのマーキングで，$\boldsymbol{M_0}$はNの初期マーキングであり，どちらも$m \times 1$ベクトルである．(3.3)式の右辺の中の$\boldsymbol{M}_0$が既知ならば，(3.3)式の右辺は定数$(\boldsymbol{yM}_0 = \mathrm{c})$を表す．これより(3.3)式は，$\boldsymbol{M}_0$から可達なすべてのマーキング$\boldsymbol{M}$において，そのトークンの重みつき総和が不変であることを示している．

バッチプロセスの正常な動作を表すペトリネットモデルのプレースがバッチプロセスのプロセス変数と関連付けられているならば，そのモデルから得られるプレースインバリアントは，バッチプロセスが正常な動作をする限り，複数のプロセス変数が満たすべき動作制約となる．よって，バッチプロセスの正常な動作を表すペトリネットモデルからプレースインバリアントを求めることで，(3.4)式で表される動作制約を求めることが可能となる．

$$C_k : \quad \boldsymbol{y_k M} = \mathrm{c}_k (k = 1,2,\cdots,K) \tag{3.4}$$

ここで，$\boldsymbol{y}_k$はNのインバリアントで，Kは独立なインバリアントの個数である．また，c_kは$\mathrm{c}_k = \boldsymbol{y_k M_0}$となる．

プレースインバリアントはペトリネットの構造が決まれば，容易に導出が可能である．したがって，バッチプロセスが大規模かつ複雑化した場合においても，本手法を用いることで，動作制約を効率的に導出することが可能である．

なお，プレースインバリアントの 0 でない成分に対応するプレースの集合は台と呼ばれ，プレースインバリアント$\boldsymbol{y}$の台は$\|\boldsymbol{y}\|$と表す．台は，その台の空でない真部分集合がいずれも台でないとき，極小台と呼ばれ，対応するインバリアントを極小台インバリアントと呼ぶ．ペトリネットNにおける任意のインバリアントは，極小台インバリアントの線形結合で求めることができる．また，極小台インバリアントを求める解法としては，Fourier-Motzkin 法が知られている[26]．しかし，バッチプロセスにおける異常の検出においては，極小台インバリアントから導出される動作制約のみを用いることが必ずしも良いわけではない．これに関しては，第 6 章で検討を行う．

3.3. ペトリネットに基づく異常診断

3.3.1. 異常の検出

動作制約に基づく異常の検出について考え方を述べる．まず，3.2 で述べたバッチプロセスの正常な動作を表すペトリネットモデルから動作制約を導出する．そして，新たな観測データが得られるたびに，動作制約の成否を確認し，動作制約が成立していないならば，異常が発生したと判断する．つまり，プロセス変数の観測のみによって異常の発生を検出することが可能となる．以上のことから，異常の検出に係る方法は次の通りである．

異常の検出の基本的な考え方：

Step 1: 動作制約の導出

バッチプロセスの正常な動作を表すペトリネットモデル（異常対応モデル）からプレースインバリアントを求め，それらが表す動作制約を求める．

Step 2: 運用時の対応

新たな観測データが得られるたびに Step 1 で求めた動作制約の成否を調べる．もし，成立していない動作制約があれば，バッチプロセスが望ましくない状態，すなわち異常が発生したと判断する． ■

注意:

すべての動作制約が成立していても，必ずしもバッチプロセスが望ましい状態であるとは限らない．しかし，成立していない動作制約があれば，異常が発生していると判断することができる． ■

例題 3-1：

あるバッチプロセスの正常な動作をペトリネットで表現したところ，図 3-1 で示すペトリネットモデルNを得た．Nから得られるトランジションの発火形態の集合Sは以下の通りである．

$$S = \{t_1t_3t_2\cdots, t_1t_2t_3\cdots, t_1t_3t_1t_2t_2t_3\cdots, t_1t_3t_1t_2t_3t_2\cdots, \cdots, \cdots\}$$

Sの要素はすべて正常な動作となる．

Nの接続行列$\boldsymbol{D}$は次の通りである．

$$\boldsymbol{D} = \begin{bmatrix} -2 & 1 & 1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$

(3.1)式で表される同次方程式から，次のプレースインバリアント$\boldsymbol{y}_1$，$\boldsymbol{y}_2$，$\boldsymbol{y}_3$，$\boldsymbol{y}_4$が得られる．

$$\boldsymbol{y}_1 = [0 \quad 1 \quad 1 \quad 1]$$
$$\boldsymbol{y}_2 = [1 \quad -1 \quad 1 \quad 0]$$
$$\boldsymbol{y}_3 = [1 \quad 0 \quad 2 \quad 1]$$
$$\boldsymbol{y}_4 = [1 \quad -2 \quad 0 \quad -1]$$

初期マーキングは$\boldsymbol{M}_0 = [3 \quad 1 \quad 0 \quad 1]^{\mathrm{T}}$であるから，次の動作制約を得ることができる．ここで，$M(i)$はプレースp_iにおけるトークンの数を表している．

$$C_1: \boldsymbol{y}_1\boldsymbol{M} = M(2) + M(3) + M(4) = 2 \ (= \boldsymbol{y}_1\boldsymbol{M}_0)$$
$$C_2: \boldsymbol{y}_2\boldsymbol{M} = M(1) - M(2) - M(3) = 2 \ (= \boldsymbol{y}_2\boldsymbol{M}_0)$$
$$C_3: \boldsymbol{y}_3\boldsymbol{M} = M(1) + 2M(3) + M(4) = 4 \ (= \boldsymbol{y}_3\boldsymbol{M}_0)$$
$$C_4: \boldsymbol{y}_4\boldsymbol{M} = M(1) - 2M(2) - M(4) = 0 \ (= \boldsymbol{y}_4\boldsymbol{M}_0)$$

動作制約が$\boldsymbol{y}_1\boldsymbol{M} \neq 2$，$\boldsymbol{y}_2\boldsymbol{M} \neq 2$，$\boldsymbol{y}_3\boldsymbol{M} \neq 4$，$\boldsymbol{y}_4\boldsymbol{M} \neq 0$の何れかの条件となった時点で，動作制約が成立していないこととなり，バッチプロセスに異常が発生したことを検出できる． ■

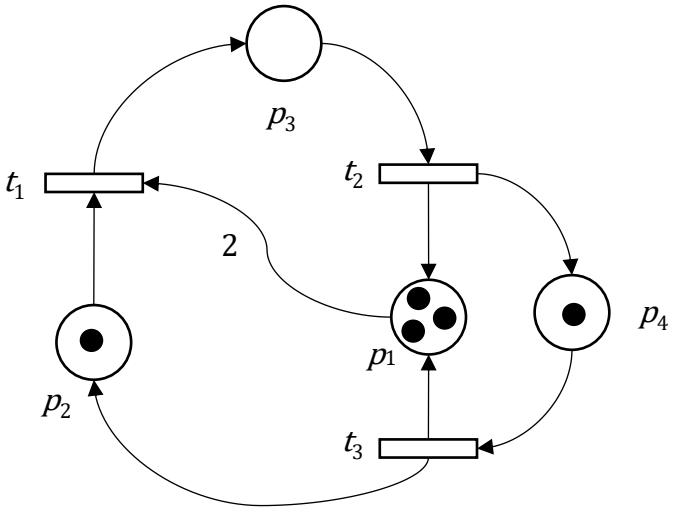


図 3-1　バッチプロセスの正常な動作を表すペトリネットモデルN

3.3.2.　異常の特定

本項では異常の特定について述べる．第 2 章で説明したように，バッチプロセスの異常の原因の特定を，動的モデルを用いて行うためには，異常状態と正常状態から異常状態へと遷移させる異常動作がモデルに組み込まれていることが必要となる．2.5 では，異常対応モデルの現実的な構築方法として，未経験の異常が発生するたび，その異常をモデルに組み込む手法を示した．よって，異常を検出した際，それが経験済みの異常であれば，その異常が表す異常動作を異常の原因として特定することができる．

以上のことから，異常の特定の基本的な考え方は次のとおりである．ここで，時刻τ_{i-1}から時刻τ_iにおけるバッチプロセスの動作を表すペトリネットモデルN_iの状態遷移方程式は，(3.5)式（(2.2)式と同じ）で表される．

$$\boldsymbol{M}_i = \boldsymbol{M}_{i-1} + \boldsymbol{D}_i \boldsymbol{x}_i \tag{3.5}$$

ここで，$\boldsymbol{M}_i(\boldsymbol{M}_{i-1})$は$N_i$の時刻$\tau_i(\tau_{i-1})$におけるマーキングで，$\boldsymbol{D_i}$は$N_i$の接続行列，$\boldsymbol{x}_i$は$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$へ遷移させる発火回数ベクトルである．

異常特定の基本的な考え方：

Step 1:　（異常の検出）

バッチプロセスの状態を観測し，時刻τ_iにおいて新たな状態が観測されるたびに，$N_i := N_{i-1}$とし，その状態$\boldsymbol{M}_i$が動作制約を満たすか否かを確認する．満たしている場合，$i := i + 1$とし，Step1 を繰り返す．満たさない場合は，Step2 へと移動する．

Step 2: （異常の特定）

$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$への遷移がN_iから矛盾なく説明できる，すなわち，$\boldsymbol{M}_i$が経験済みの異常を表す異常トランジションの発火によって$\boldsymbol{M}_{i-1}$から遷移した異常マーキングならば，その異常トランジションが表す動作を異常の原因と特定する．そして，$i := i+1$とし，Step1 へと戻る．説明できない場合，未経験の異常が発生したとし，モデルの更新を行う． ■

3.3.3. 異常診断および異常対応モデルの更新

本項では 2.5，3.3.1，3.3.2 で示した異常の検出，検出された異常の原因の特定，未経験の異常のモデル化に係る一連の流れを次に示す（図 3-2）．

異常診断および異常対応モデルの更新の進め方：

Step 1: （動作制約の導出（3.2））

時刻τ_0におけるバッチプロセスの正常な動作を表すペトリネットモデルN_0からプレースインバリアント解析をもとに，動作制約$C_k(k = 1,2,\cdots,K)$を導出する．

Step 2: （異常の検出（3.3.1））

新たな状態が観測されるたびに，$N_i := N_{i-1}$とし，その状態$\boldsymbol{M}_i$が，すべての動作制約を満たしているか確認する．もし，すべての動作制約を満たしているならば，時刻τ_iにおける状態$\boldsymbol{M}_i$は正常状態となり，$i \coloneqq i+1$とし，Step 2 を繰り返す．そうでなければ，異常が発生したとし，Step 3 へ進む．

Step 3: （異常の特定（3.3.2））

$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$への遷移がN_iから矛盾なく説明できる異常トランジションがあるか確認する．もし，説明が可能な異常トランジションがあれば，その異常トランジションを時刻τ_{i-1}から時刻τ_iの間に発生した異常の原因とし，$i \coloneqq i+1$として，Step 2 へ戻る．そうでなければ，未経験の異常が発生したとし，Step 4 へ進む．

Step 4: （モデルの更新（2.5.2））

N_iにおいて，$\boldsymbol{M}_{i-1}$から$\boldsymbol{M}_i$の遷移が矛盾なく説明できるよう，新たな異常トランジションをモデルN_iに組み込みモデルを更新する．モデル更新後，$i \coloneqq i+1$として，Step 2 へ戻る． ■

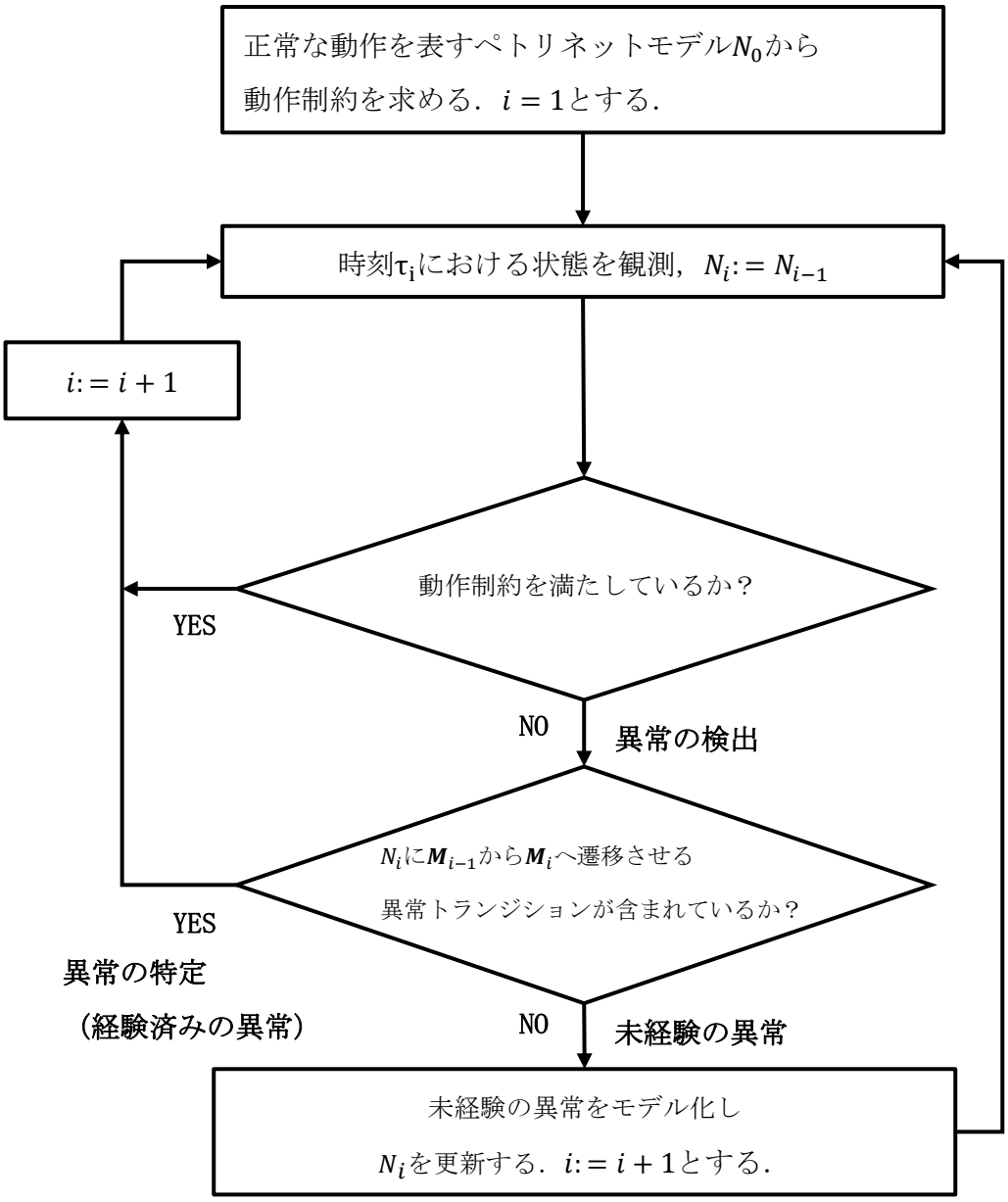


図 3-2　異常の検出，異常の特定と未経験の異常のモデル化に係るフローチャート

例題 3-2：

例題 3-1 で示したバッチプロセスにおいて，異常診断を実施した結果，表 3-1 に示すように，時刻τ_4において，動作制約C_2，C_3，C_4が満たされなくなり，異常が検出された．よって，時刻τ_3から時刻τ_4の間に異常（動作）が発生したこととなる．しかし，現行のモデルN_4においては，次の状態遷移方程式を満たす$\boldsymbol{x}_4$の非負整数解が得られないため，対象となるバッチプロセスに未経験の異常動作が発生したと推定した．

$$\boldsymbol{M}_4 = \boldsymbol{M}_3 + \boldsymbol{D}_4\boldsymbol{x}_4$$
$$= \boldsymbol{M}_3 + \boldsymbol{D}_0\boldsymbol{x}_4$$
$$\begin{bmatrix}0\\0\\0\\2\end{bmatrix} = \begin{bmatrix}0\\0\\2\\0\end{bmatrix} + \begin{bmatrix}-2 & 1 & 1\\ -1 & 0 & 1\\ 1 & -1 & 0\\ 0 & 1 & -1\end{bmatrix}\boldsymbol{x}_4$$

そこで，未経験の異常動作をモデル化するため，時刻τ_3から時刻τ_4の間にトランジション t_1，t_2，t_3 は発火していない，すなわち$\boldsymbol{x}_4 = [0 \quad 0 \quad 0]$とし，$\boldsymbol{b}_4$を求める，

$$\boldsymbol{b}_4 = \begin{bmatrix}0\\0\\0\\2\end{bmatrix} - \begin{bmatrix}0\\0\\2\\0\end{bmatrix} + \begin{bmatrix}-2 & 1 & 1\\ -1 & 0 & 1\\ 1 & -1 & 0\\ 0 & 1 & -1\end{bmatrix}[0 \quad 0 \quad 0]^T = \begin{bmatrix}0\\0\\-2\\2\end{bmatrix}$$

ここで，一組の異常トランジションにより異常が発生したとし，$\boldsymbol{b}_4$を次のように分解する．

$$\boldsymbol{D'}_4 = \begin{bmatrix}0\\0\\-2\\2\end{bmatrix},\ \boldsymbol{x'}_4 = [1]$$

異常トランジションfを新たに組み込んだペトリネットは

$\boldsymbol{D}_4 := [\boldsymbol{D}_4 \quad \boldsymbol{D'}_4]$および$\boldsymbol{x}_4 := [\boldsymbol{x}_4 \quad \boldsymbol{x'}_4]^{\mathrm{T}}$とし，次のような接続行列$\boldsymbol{D}_4$で表すことができる（図 3-3）．

$$\boldsymbol{D}_4 = \begin{bmatrix}-2 & 1 & 1 & 0\\ -1 & 0 & 1 & 0\\ 1 & -1 & 0 & -2\\ 0 & 1 & -1 & 2\end{bmatrix}$$

N_4から得られるトランジションの発火形態の集合S_fは次の通りである．

$$S_f = \{t_1t_3t_2\cdots, t_1t_2t_3\cdots, t_1t_3t_1t_2t_2t_3\cdots, t_1t_3t_1ft_3t_3t_1\cdots, t_3t_1t_1ft_3t_3t_1\cdots, \cdots\}$$

例題 3-1 で得られたSとS_fの関係は$S \subseteq S_f$となることは明らかである．よって，$S_f - S$は異常トランジションfを含む異常動作の集合となる． ■

表 3-1　バッチプロセスの状態遷移と動作制約

時刻	観測される状態	不成立な動作制約	プロセスの状態	ペトリネットモデル
$\boldsymbol{\tau_0}$	$\boldsymbol{M}_0$: $[3 \quad 1 \quad 0 \quad 1]^{\mathsf{T}}$	-	正常	N_0
$\boldsymbol{\tau_1}$	$\boldsymbol{M}_1$: $[1 \quad 0 \quad 1 \quad 1]^{\mathsf{T}}$	-	正常	$N_1(=N_0)$
$\boldsymbol{\tau_2}$	$\boldsymbol{M}_2$: $[2 \quad 1 \quad 1 \quad 0]^{\mathsf{T}}$	-	正常	$N_2(=N_1)$
$\boldsymbol{\tau_3}$	$\boldsymbol{M}_3$: $[0 \quad 0 \quad 2 \quad 0]^{\mathsf{T}}$	-	正常	$N_3(=N_2)$
$\boldsymbol{\tau_4}$	$\boldsymbol{M}_4$: $[0 \quad 0 \quad 0 \quad 2]^{\mathsf{T}}$	C_2, C_3, C_4	異常	-

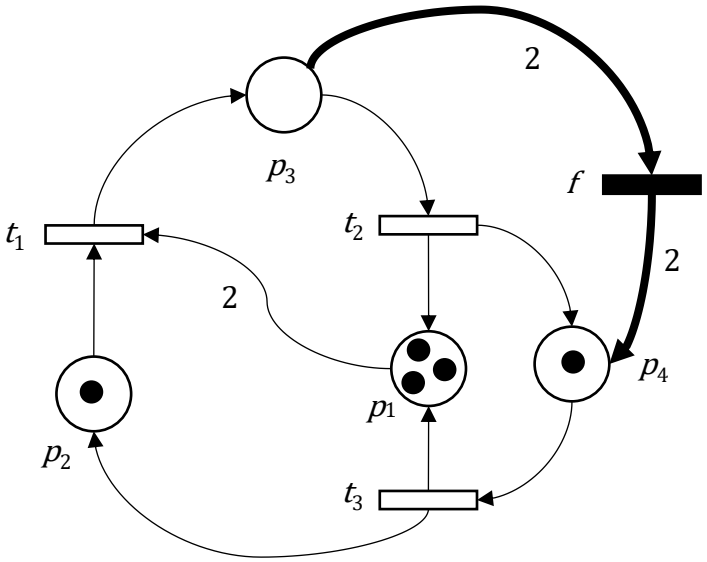


図 3-3　未経験の異常$\boldsymbol{f}$を組み込んだペトリネットモデル$\boldsymbol{N_4}$

3.4.　まとめ

バッチプロセスは複数の動作を同時並行的に行うため，プロセス変数の組合せが膨大となり，それが頻繁に変化するという特徴を有する．そのため，異常 F2 に係る異常診断が重要となるが，プロセスが大規模化・複雑化するに従い，その対応が困難となっていた．そこで本章においては，バッチプロセスにおける異常診断を効率的に行う方法の検討を行った．

3.2 では，動作制約を導出する手法としてペトリネットの解析手法であるプレースインバリアント解析に着目し検討を行った．ペトリネットモデルのプレースがバッチプロ

セスのプロセス変数と関連付けられているならば，そのペトリネットから得られるプレースインバリアントは，バッチプロセスが正常な動作をする限り複数のプロセス変数が満たすべき動作制約となる．このことから，バッチプロセスの正常な動作を表すペトリネットモデルを用い，プレースインバリアント解析から動作制約を効率的に導出する方法を開発した．3.3 では，バッチプロセスにおける異常診断に係る検討を行い，異常の検出においては，動作制約の成否を判定することで，バッチプロセスに異常が発生したことを検出する手法を開発した．また，異常の検出後，その異常状態への遷移が現在のペトリネットモデル内にある異常動作（トランジション）から説明できるかを確認し，そのような異常動作があれば，それを異常の原因として特定する手法を開発した．また，異常診断と未経験の異常の組み込みに係るモデルの更新手法を同一の枠組みで行う方法を示した．

一方，異常を随時組み込むことで更新されるペトリネットモデルは，異常を表す動作や状態がモデル化されている．よって，そのモデルの上で異常状態への遷移や異常動作が行われることを回避する制御器の設計ができれば，その異常の再発を回避することが可能となる．第 4 章においては，異常回避のための制御器の設計に関する検討を行う．

第4章.バッチプロセスにおける異常状態の回避

第 3 章ではペトリネットを用いた異常診断の手法を開発した．一方，異常診断で用いた異常対応モデルには，異常がモデル化されており，その上で異常の回避を目的とした制御器を設計することが可能となる．本章においては，異常の回避を目的とした制御器の設計問題を定式化し，その解法について検討する．

4.1. はじめに

連続プロセスで広く用いられるレギュレトリ制御器の設計は，伝達関数や状態方程式等の優れた動的モデルが存在しており，そのモデルを用いた設計手法が多く存在している．よって，同制御器を用いた異常の回避のための設計手法も豊富に存在していると言える．一方，バッチプロセスにおける手順制御器や協調制御器の設計手法はプロセス制御器と比較して十分な研究がなされておらず，経験者の知識やノウハウに依存する部分が多かった．そのため，プロセスが大規模化・複雑化するに従い，その種の制御器の設計が困難となっていた．本章においては，異常対応モデルの上で異常の再発の回避を目的とする手順制御器や協調制御器の合理的な設計手法について検討する．

第 2 章で示したように，本研究ではバッチプロセスに現れる異常を，次のようにとらえることとした．

異常状態：プロセスが正常に動作する限り，決して現れることがない状態

異常動作：プロセスの状態を正常状態から異常状態に遷移させる動作

したがって，異常を回避するにあたっても，次の 2 通りが考えられるものと思われる．

異常の回避の考え方：

⑴異常状態の回避

プロセスの状態に着目し，状態が異常状態に陥ることを回避する．

⑵異常動作の回避

プロセスの動作に着目し，異常動作が行われることを回避する． ■

以下では，これらに対して，それぞれ次のような制御器の設計問題を考える．

異常状態回避問題 PS：

”プロセスの状態が常に動作制約を満たすように動作すること”を制御目的にもつ制御器を設計する問題 ■

異常動作回避問題 PD：

”プロセスの動作が異常動作を行うことなく常に正常に動作すること”を制御目的にもつ制御器を設計する問題 ■

これらの問題の相違点をまとめ表 4-1 に示す.

表 4-1　異常の回避を目的とした制御器の設計問題の考え方

問題	制御目的	実現方法	着眼点
異常状態回避問題 PS	異常状態の回避	動作制約の順守	プロセスの状態遷移に着目
異常動作回避問題 PB	異常動作の回避	望みの動作の実行	プロセスの動作履歴に着目

なお，上の異常動作の定義に従えば，異常状態の回避と異常動作の回避は本質的には全く同等の問題であることは明らかである．しかし，現実のバッチプロセスを考えた場合，そこでの状態と動作は多種多様である．そこには，反応系プロセスのようにその状態の多くがセンサーを使って観測できないようなプロセスもあれば，流動系プロセスのようにその動作の多くが弁を使って操作できるプロセスもある．したがって，異常の回避を考えるにあたっては，対象となるプロセスの特徴や適用できる操作方法を考慮しながら，それぞれに適した形でこれにあたることが必要となる．このことからも，上の 2 つの異常の回避の問題を考えることは大きな意義がある.

本章においては，これら 2 つの問題のうち問題 PS に係る検討を行う．本章の構成は次の通りである．4.2 では，問題 PS をペトリネットの上で定式化を行い，定式化された問題の解法を示す．4.3 では，第 2 章，第 3 章で示した異常対応に係る一連の問題の解法を，バッチプロセスにおける冷却プロセスに適用した例を示す．また，4.4 では第 3 章から本章における一連の異常対応に係る考察を述べる．また，問題 PD に係る検討は第 5 章で行う.

4.2. 異常状態回避問題

4.2.1. ペトリネットに基づく異常状態回避問題の定式化

本章ではまず初めに幾つかの用語を定義する．なお，バッチプロセスの動作を表すプラントネットは, 第 3 章の異常の検出で用いた異常の動作や状態が組み込まれた異常対応モデルを用いることができる.

プラントネットN_P：

バッチプロセスの動作を表すペトリネットモデル$N_P = (P_P, T_P, F_P, W_P, \boldsymbol{M}_{P0})$をプラントネットと呼ぶ. ここで, P_Pはバッチプロセスのプロセス変数を表すプレースの集合を，T_Pはバッチプロセスの動作を表すトランジションの集合を，$F_P \subseteq P_P \times T_P \cup T_P \times P_P$はアークの集合を, $W_P: F_P \to \mathbb{N}$(ただし, $\mathbb{N}$は自然数の集合)はアークの重みを定める関数を，$\boldsymbol{M}_{P0}$はN_Pの初期マーキングを表す. ■

制御器ネットN_C：

バッチプロセスの動作を操作する制御器を表すペトリネットモデル$N_C = (P_C, T_C, F_C, W_C, \boldsymbol{M}_{C0})$を制御器ネットと呼ぶ. ここで, P_Cは制御器に含まれる状態に相当するプレースの集合を，T_Cは制御器に含まれる事象に相当するトランジションの集合を，$F_C \subseteq P_C \times T_C \cup T_C \times P_C$はアークの集合を，$W_C: F_C \to \mathbb{N}$はアークの重みを定める関数を，$\boldsymbol{M}_{C0}$は$N_C$の初期マーキングを表す. ■

閉ループシステムネット：

プラントネットN_Pに制御器ネットN_Cを合成して得られるペトリネットモデルを閉ループシステムネット$N_P \oplus N_C$と呼ぶ．ここで+は直和を，∪は和集合を，⊕はペトリネットの合成を表す演算子である.

N_PとN_Cの合成は$N_P \oplus N_C = (P_P + P_C, T_P \cup T_C, F_P \cup F_C, W_P \cup W_C, \boldsymbol{M}_{P0} \cup \boldsymbol{M}_{C0})$となる．■

第 2 章で示したように，本研究では，バッチプロセスに現れる異常状態を

異常 F1：1 つのプロセス変数が単独で示す異常

異常 F2：複数のプロセス変数が組になって初めて示す異常

の 2 つに分類した．本研究では異常 F2 の異常を検出するため，バッチプロセスが正常な動作をする限り複数のプロセス変数が満たすべき条件である動作制約を定義し, 動作制約の成否によって，この異常を検出する方法を開発した．第 3 章では，バッチプロセスの正常な動作を表すペトリネットモデルから, ペトリネットの解析手法の一つである

プレースインバリアント解析を用いることで，(4.1)式で表す動作制約（(3.4)式と同じ）が効率的に導出できることを示した．

$$C_k: \quad \boldsymbol{y_k M_P} = c_k (k = 1,2,\cdots,K) \tag{4.1}$$

ここで，バッチプロセスの正常な動作を表すペトリネットモデルをNとすると，$\boldsymbol{y_k}$はNのプレースインバリアントで，KはNの独立なプレースインバリアントの個数である．また，$\boldsymbol{M_0}$をNの初期マーキングとし既知とすると，$c_k = \boldsymbol{y_k M_0}$は定数となる．

異常状態を回避するためには，動作制約が常に成立するような制御器が必要となる．4.1 で示した異常状態回避問題 PS は，ペトリネットの上で次のように定式化することができる．

異常状態回避問題 PS：

バッチプロセスの動作を表すプラントネットN_Pとその動作制約$C_k (k = 1,2,\cdots,K)$が与えられたとき，(4.2)式のもとで閉ループシステムネット$N_P \oplus N_C$の状態が動作制約を常に満たすような制御器ネットN_Cを求めよ．

$$T_C \subseteq T_P \tag{4.2}$$ ■

(4.2)式は，問題 PS の解となるN_Cの制御器トランジションの集合T_Cが，プラントトランジションの集合T_Pの部分集合であることを示している．このことから，この制御器がバッチプロセスの動作をトリガーとして動作することを示している．例えば，“熱交換器の冷却弁と加熱弁に対して，冷却弁が開くことで制御器が作動し，冷却弁が閉まるまでは加熱弁を開くことはできないという制御を加える”ことを考えよう．この制御器は，バッチプロセスの“冷却弁開”の動作をトリガーに“加熱弁開にしない”という動作を行う．

4.2.2. ペトリネットに基づく異常状態回避問題の解法

本項では，ペトリネットに基づく異常状態回避問題 PS の解法について述べる．4.1 で述べたように問題 PS は状態遷移に着目し，異常状態への遷移を回避する．第 3 章で示したように，時刻τ_iで動作制約が不成立となると，時刻τ_{i-1}から時刻τ_iで異常が発生したと考えることができる．ここで，時刻τ_i以前の時刻$\tau_{i-1}, \tau_{i-2}, \tau_{i-3}, \cdots$における正常状態を表すマーキング$\boldsymbol{M}_{Pi-1}, \boldsymbol{M}_{Pi-2}, \boldsymbol{M}_{Pi-3}, \cdots$に着目する．直感的な考え方は，異常状態へ遷移する 1 ステップ前の$\boldsymbol{M}_{Pi-1}$に着目し，この正常状態への遷移を回避することで異常

状態への回避を防ぐものである．$\boldsymbol{M}_{Pi-1} - \boldsymbol{M}_{Pi}$の 0 より大きな値となる成分（プレース）は，異常状態への遷移により，そのプレースに置かれたトークン数が減ったことを意味する．よって，このようなプレースに置かれるトークン数に制約を課すことで，異常状態への遷移を回避することができる．

この考えを一般化したものが線形不等式(4.3)である．ここで，バッチプロセスの動作を表すプラントネット$\boldsymbol{N_P}$とし，mをN_Pのプレース数，nをN_Pのトランジション数とする．

$$\boldsymbol{L}\boldsymbol{M_P} \leq c \tag{4.3}$$

ここで，$\boldsymbol{L}$は$1 \times m$の整数ベクトルで，$\boldsymbol{M}_P$はN_Pのマーキングで$m \times 1$ベクトル，cは定数である．(4.3)式は，バッチプロセスの状態に係る新たな制約である．

このようなマーキングに係る不等式で表される制約から制御器を設計する手法に Moody らの手法がある[15][45]．以下では，Moody らが提案した手順に従い，制御器を設計する．

(4.3)式にスラック変数μを導入すると，次の(4.4)式となる．

$$\boldsymbol{L}\boldsymbol{M_P} + \mu = c \tag{4.4}$$

この式を変形することで得られる(4.5)式は，プラントネットN_Pに制御器ネットN_Cを合成した閉ループシステムネット$N_P \oplus N_C$の動作が満たすべきプレースインバリアントであることを示す．

$$[\boldsymbol{L} \quad \boldsymbol{I}]\begin{bmatrix}\boldsymbol{M_P}\\ \mu\end{bmatrix} = c \tag{4.5}$$

ここで，$\boldsymbol{I}$は単位行列を表す．$[\boldsymbol{L} \quad \boldsymbol{I}]$は$N_P \oplus N_C$のインバリアントであるから，(4.6)式が得られる．

$$[\boldsymbol{L} \quad \boldsymbol{I}]\begin{bmatrix}\boldsymbol{D_P}\\ \boldsymbol{D_c}\end{bmatrix} = 0 \tag{4.6}$$

ここで，$\boldsymbol{D_P}$はN_Pの$m \times n$接続行列で，$\boldsymbol{D_c}$はN_Cの$1 \times n$接続行列である．(4.6)式を変換することで$\boldsymbol{D_c}$が一意に求められる（(4.7)式）．

$$\boldsymbol{D_c} = -\boldsymbol{L}\boldsymbol{D_P} \tag{4.7}$$

すなわち，(4.3)式を満たすため，制御器プレースの集合P_Cの要素が一つ($P_C = \{p_c\}$)からなるN_Cが求められる．また，p_Cの初期マーキングμ_0は，(4.4)式より，次の(4.8)式となる．

$$\mu_0 = c - \boldsymbol{L}\boldsymbol{M_{P0}} \tag{4.8}$$

ここで，$\boldsymbol{M}_{P0}$はN_Pの初期マーキングで$m \times 1$ベクトルである．

また，2.4 において異常 F1 をペトリネットの上で“一つプレースに置かれたトークン数の正常値からの逸脱”と表現できることを示した．よって，異常 F1 の回避は，(4.3)式の$\boldsymbol{L}$の 0 でない要素が 1 つの問題に帰着される．

以上のことから，ペトリネットに基づく問題 PS の解法は次のとおりである．

問題 PS の解法：

Step 1:

異常状態への遷移を回避するよう，状態に係る新たな制約$\boldsymbol{LM_P} \leq \boldsymbol{c}$を設定する．

Step 2:

$\boldsymbol{D_c} = -\boldsymbol{LD}_P, \mu_0 = c - \boldsymbol{LM}_{P0}$とし，制御器ネット$N_C$を求める． ■

問題 PS においては，状態に係る新たな制約$\boldsymbol{LM_P} \leq \boldsymbol{c}$が定まれば，制御器は一意に求められる．

異常の回避のための制御器においては，目的とする異常の回避を実現し，その上で，できるだけ他の正常な動作の制限しないように設計することが望ましい．直感的な考えとしては，上述したように，異常状態の 1 ステップ前の正常状態への遷移を防ぐことが，他の動作を最も制限しない異常状態の回避方法と思われる．しかし，そのような考えが常に最適であるとは限らない．よって，原理的には，異常状態への回避を実現する複数ある制御器の候補の中から，その制御器を追加することで得られるバッチプロセスの動作（閉ループシステムの動作）が，もとのバッチプロセスの動作に最も近いものを選ぶとことで，最適な制御器が得られる．しかし，これを厳密に行うためには，すべての可達なマーキングを求める必要があり非常に計算負荷の高い手法である．より効率的に最適な制御器を求める手法の開発は，今後の課題である．

例題 4-1：

例題 3-2 でモデル化した異常を回避するための制御器を設計する．時刻τ_4の異常マーキングM_{P4}の 1 ステップ前の時刻τ_3の正常マーキング$M_{P3} = [0 \quad 0 \quad 2 \quad 0]^{\mathrm{T}}$を考える．$M_{P3} - M_{P4}$の 0 より大きい要素は$M_P(3) = 2 > 0$である．このことから，異常状態への遷移を回避する目的で決定される新たな制約(4.3)式を，

$$M_P(3) \leq 1$$

とする．次に，スラック変数μを用いて，(4.3)式を等式に変換する．

$$[0 \quad 0 \quad 1 \quad 0]M_P + \mu = 1$$

N_Cの接続行列$\boldsymbol{D}_C$およびp_cの初期マーキングμ_cは，次のとおりである．

$$\boldsymbol{D}_C = -[0 \quad 0 \quad 1 \quad 0]\begin{bmatrix} -2 & 1 & 1 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & 0 & -2 \\ 0 & 1 & -1 & 2 \end{bmatrix} = [-1 \quad 1 \quad 0 \quad 2]$$

$$\mu_c = 1 - [0 \quad 0 \quad 1 \quad 0]\begin{bmatrix} 3 \\ 1 \\ 0 \\ 1 \end{bmatrix} = 1$$

制御器ネットを加えた閉ループシステムネットを図 4-1 に示す．このペトリネットでは，異常トランジションfは発火せず，動作制約が不成立となる異常マーキングへの遷移は回避される． ■

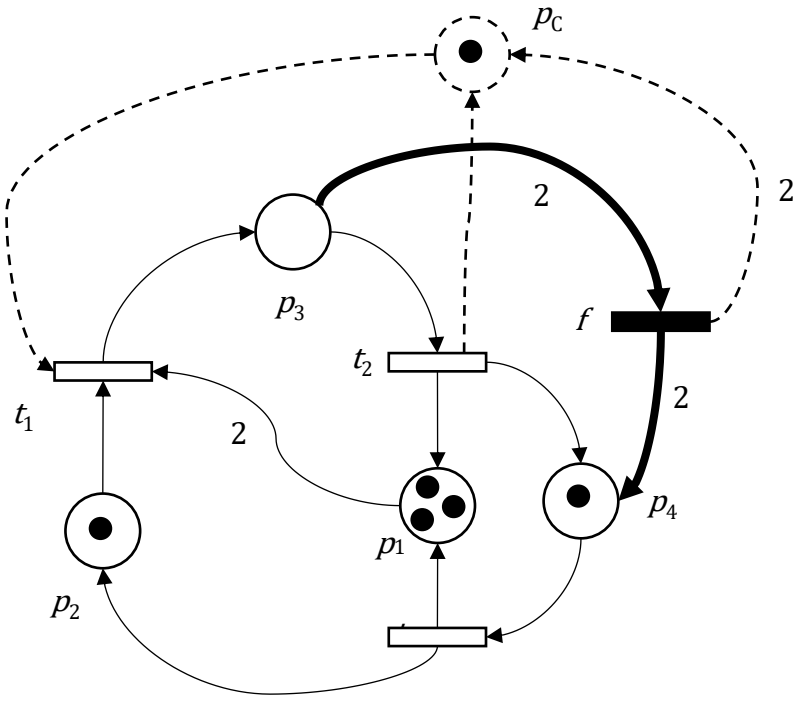


図 4-1　異常トランジションを回避するように設計される閉ループシステムネット

4.3.　異常対応に係る提案手法の適用例

4.3.1.　対象のプロセス

第3章および本章で開発したバッチプロセスにおける異常診断とその回避の手法を，図 4-2 に示すプロセスに適用した事例を示す．図 4-2 はバッチ反応器を冷却するためのジャケット循環型冷却プロセスを表している．このプロセスでは，冷媒の弁を操作することで反応器の中の反応物の温度を制御しており，ポンプ吐出側の流量，ポンプの回

転数（ポンプの起動停止）および弁の開度を観測している．また，次の前提条件が成立しているものとする．

前提条件：

- ✓ 流量計・弁・ポンプの状態の示すセンサーからの出力値は常に正しい．
- ✓ 冷媒タンクの液面は常に一定とする．
- ✓ 初期状態で配管および反応器ジャケットに冷媒は微小量存在する．
- ✓ 冷媒タンクの液ヘッドで，ある程度の流量は確保できるが，ポンプを起動することで最大値へと到達する．

このプロセスの起動時における目標とする操作は次の通りである：

弁を開け，しばらくした後，ポンプを動かすことで，冷媒を反応器ジャケットへ供給する．

この操作を行うことで，ポンプ吐出側にある冷媒用の流量は，起動直後は小さな値であるが，徐々に上昇していき，最終的にある値F_Sに到達する．

図 4-3 および表 4-2 は時刻τ_0における冷却プロセスの動作を表すペトリネットモデルN_0を表す．$p_1,\cdots,p_7$は冷却プロセスにおけるプロセス変数の状態を，$p_8,\cdots,p_{13}$は事象の順序関係を表す．N_0においては，操作の柔軟性を考慮し，弁とポンプの間には順序関係に係る制約は課されていない．N_0の状態遷移方程式は(4.9)式となる．ここで，$\boldsymbol{x}$はN_0の発火回数ベクトルであり8×1の整数ベクトルである．

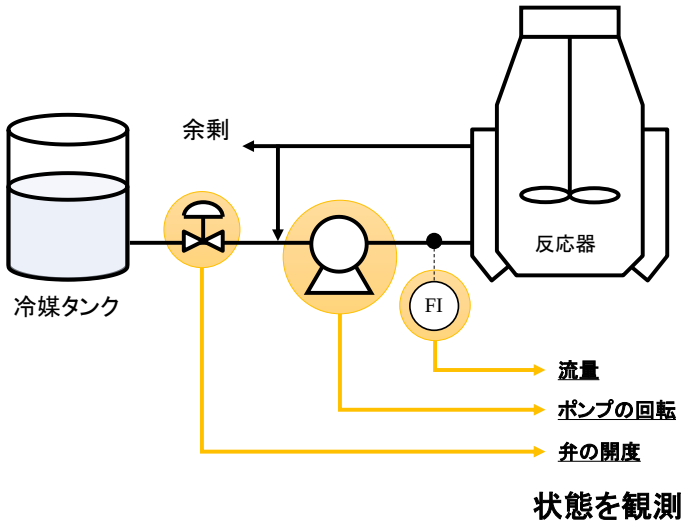


図 4-2　ジャケット付バッチ反応器における冷却プロセス

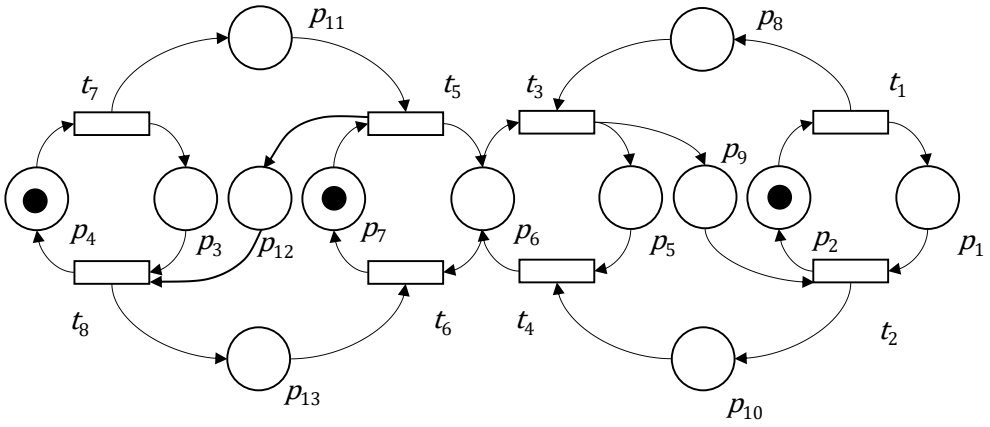


図 4-3　冷却プロセスの初期モデルN_0

表 4-2　プレースの説明

プレース	意味
p_1	ポンプ起動中
p_2	ポンプ停止中
p_3	弁が開
p_4	弁が閉
p_5	流量Fが最終的な値F_Sと一致
p_6	流量Fが 0 より大きくF_Sより小さい
p_7	流量Fが 0
p_8	ポンプ起動後，流量増加
p_9	流量FがF_S到達後，ポンプ停止
p_{10}	ポンプ停止後，流量減少
p_{11}	弁の開け，流量増加
p_{12}	流量増加後，弁の閉止
p_{13}	弁を閉め，流量減少
p_c	制御器プレース

$$\boldsymbol{M}_i = \begin{bmatrix}0\\1\\0\\1\\0\\0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} + \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1\\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1\\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0\\ 0 & 0 & -1 & 1 & 1 & -1 & 0 & 0\\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0\\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0\\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1\\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix} \boldsymbol{x} \tag{4.9}$$

4.3.2. 異常の検出

動作制約の導出

N_0のプレースインバリアント解析より，以下の動作制約が得られる．C_1からC_3はポンプの動作とそれに伴う流量の状態変化に係る動作制約であり，例えばC_2はポンプ起動開始により流量Fが上昇し，流量FがF_Sに到達後，ポンプ停止までは流量Fは流量F_Sに維持されることを表している．したがって，C_2の動作制約が不成立となった場合，ポンプ起動時に何らかの異常が発生したと推測できる．C_3はポンプ停止により流量FがF_Sから減少することを表している．次にC_4，C_5は弁の操作とそれに伴う流量変化に係る動作制約であり，例えばC_5は弁が開いている状態では流量Fは 0 ではない値を示すことを表している．したがって，C_5の動作制約が不成立となった場合，弁に何らかの異常が発生したと推測できる．C_6は流量Fが 0 である状態，0 より大きくF_Sより小さい状態，F_Sである状態のいずれかの状態であることを表している．

C_1： $M(1) + M(2) = 1$

C_2： $-M(1) + M(8) + M(9) = 0$

C_3： $-M(5) + M(9) + M(10) = 0$

C_4： $M(3) + M(4) = 1$

C_5： $M(3) - M(5) - M(6) - M(11) + M(13) = 0$

C_6： $M(5) + M(6) + M(7) = 1$

異常の検出

プロセスの状態の遷移を図 4-4 および表 4-3 に示す．時刻τ_1における状態 $\boldsymbol{M}_1$は正常であるが，時刻τ_2における状態$\boldsymbol{M}_2$は動作制約C_2を満たさない．これにより，時刻τ_2で

異常が検出された．動作制約C_2が不成立となる異常はポンプ起動時に流量が上昇しないことを表す異常である．プロセスが正常な動作であれば，流量は最大値まで到達するはずであるが，実際には最大値まで到達せず，再び流量が低下し，元の値へと戻った（図4-4）．これは，目標とする操作が実行されなかったために異常が発生したと推測された．

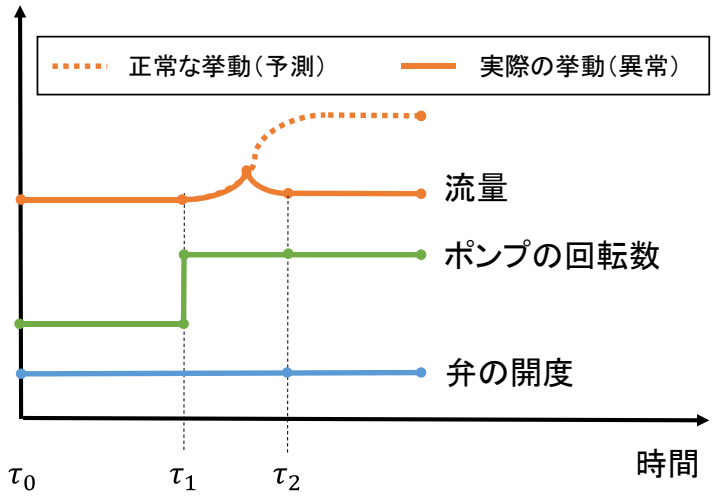


図 4-4　観測されたプロセスの状態

表 4-3　状態の遷移とプロセスの状態

時刻	観測される状態	不成立な動作制約	プロセスの状態	ペトリネットモデル
$\boldsymbol{\tau_0}$	$\boldsymbol{M}_0 = [01010\ 01000000]^{\mathsf{T}}$	-	正常	N_0
$\boldsymbol{\tau_1}$	$\boldsymbol{M}_1 = [10010\ 01100000]^{\mathsf{T}}$	-	正常	$N_1(= N_0)$
$\boldsymbol{\tau_2}$	$\boldsymbol{M}_2 = [10010\ 01000000]^{\mathsf{T}}$	C_2	異常	$N_2(\neq N_0)$

4.3.3.　異常の特定

未経験の異常の発生：

$\boldsymbol{M}_1$から$\boldsymbol{M}_2$の状態遷移は現在のモデル$N_2 = N_0$では説明ができない．すなわち，$\boldsymbol{M}_2$が表す状態は未経験の異常の発生により遷移した異常状態であることが分かった．

モデルの更新：

モデルの更新に係る(4.10)式，(4.11)式より（2.5 で示した(2.4)式，(2.5)式と同じ）モデルの更新を行う．

$$\boldsymbol{M}_i = \boldsymbol{M}_{i-1} + \boldsymbol{D_i x}_i + \boldsymbol{b}_i \tag{4.10}$$

$$\boldsymbol{b}_i = \boldsymbol{D}'_i \boldsymbol{x}'_i \tag{4.11}$$

ここで時刻τ_1からτ_2で，トランジションt_1からt_8が発火しない，すなわち$\boldsymbol{x} = \boldsymbol{0}$とする．また，$\boldsymbol{M_1}$から$\boldsymbol{M_2}$の状態遷移から$\boldsymbol{b_2}$は(4.12)式となる．ここで$\boldsymbol{x}'_2 = [1]$とした．

$$\boldsymbol{b}_2 = \boldsymbol{M}_2 - \boldsymbol{M}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} [1] \tag{4.12}$$

(4.12)式により検出した未経験の異常を組み込んだ新たなペトリネットモデル$\boldsymbol{N_2}$を得ることができる．モデル$\boldsymbol{N_2}$の状態遷移方程式は(4.13)式となる．$\boldsymbol{x}$はN_2の発火回数ベクトルであり9×1の整数ベクトルで表す．

$$\boldsymbol{M}_2 = \boldsymbol{M}_1 + \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x} \tag{4.13}$$

図 4-5 は，更新したペトリネットモデル$\boldsymbol{N_2}$を表す．新たに追加されたトランジションfは異常動作を表す．これは，バルブの開く前にポンプを起動したことで発生したポンプのキャビテーションを表す．fとp_7にある自己ループは，そのキャビテーションが連続して発生したことを表す．

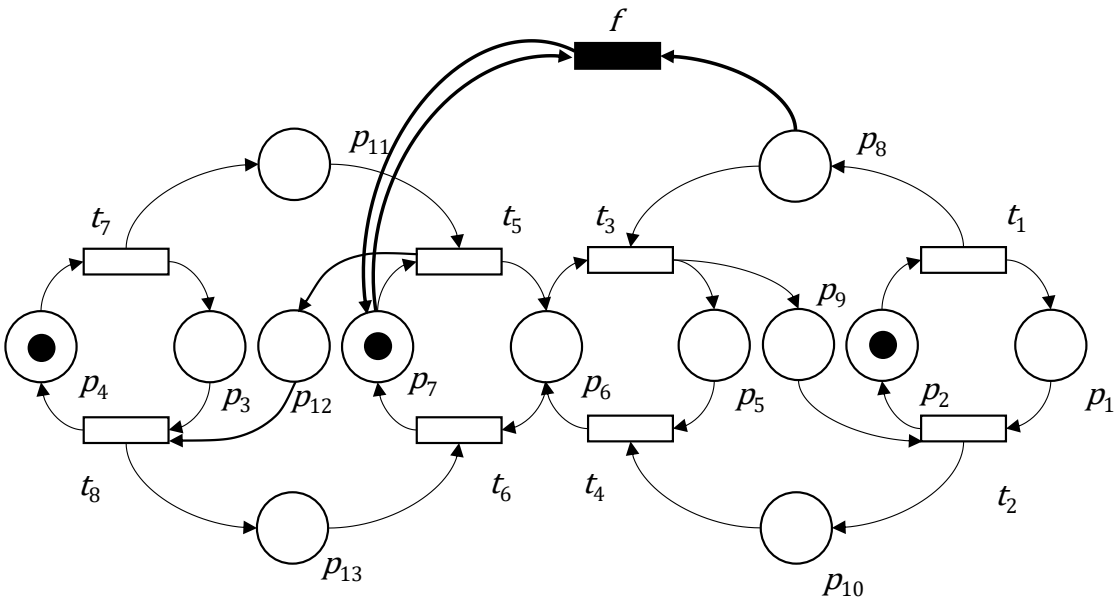


図 4-5　更新されたモデルN_2

4.3.4.　異常の回避

発生した異常に対し，その再発を回避するための制御器を設計する．ここでは，4.2で開発した異常状態の回避を考える．正常状態を表すM_1から異常状態を表すM_2へと遷移している．ここではM_1に着目し，動作制約が常に満たされるような制御器の設計を考える．ここでは，線形不等式(4.14)を考える．

$$M(7) + M(8) \leq 1 \tag{4.14}$$

よって，制御器ネットの接続行列D_cは，

$D_c = [-1 \quad 0 \quad 1 \quad 0 \quad 1 \quad -1 \quad 0 \quad 0 \quad 1]$であり，$M_C(C) = 0$となる．（図 4-6）

p_Cは制御器プレースを表し，ポンプ起動をバルブの開いた後に行うことを保証するものである．これにより，異常動作キャビテーションを回避することができる．

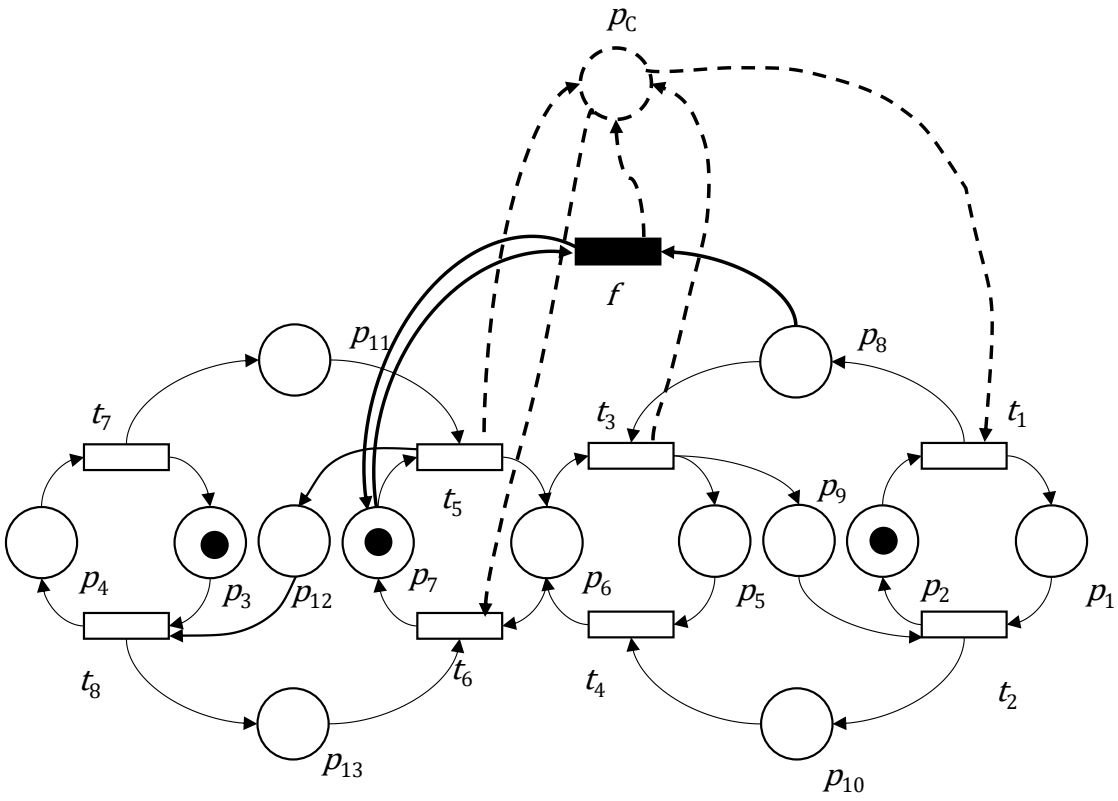


図 4-6　異常回避のための制御器ネット

4.4.　異常対応モデルに係る考察

4.4.1.　異常対応モデルを用いた異常対応

第 2 章，第 3 章，第 4 章で示した異常対応に係る問題とそれぞれの問題で用いた異常対応モデルを表 4-4 にまとめる．

第 3 章で示したように，異常の検出においては，バッチプロセスの正常な動作を表すペトリネットモデルから，プレースインバリアント解析手法により動作制約を求め，動作制約の成否で異常を検出する方法を開発した．また，異常の検出後，その異常状態への遷移が現在のペトリネットモデル内にある異常動作（異常トランジション）から説明できるかを確認し，そのような異常動作があれば，それを異常の原因として特定する手法を開発した．また第 2 章で示したように未経験の異常に対しては，検出するたびにモデルに組み込む手法を開発した．第 4 章では，異常を組み込んだペトリネットモデルをもとに，特定された異常の再発の回避を目的とした制御器の設計手法の一つを提案し

た．このように，異常対応に係る問題を，共通したペトリネットモデル（異常対応モデル）を用いて行うことで，各々の問題で得られる異常に係る情報を各問題で最大限に利用することが可能となる．

また，未経験の異常が発生するとペトリネットモデルのトランジションの数は増えるものの，プレースの数は変わらない．一方，異常状態回避のための制御器をモデルに組み込むと，制御プレースの数だけプレース数が増加する．このように，未経験の異常の発生とその再発を回避する制御器を追加するごとにモデルが大きくなっていく．そして，このモデルには制御器の働きにより異常は発生しないことから，新たな正常なバッチプロセスの動作を表すペトリネットモデルとすることができる（図 4-7）．

本研究で提案するペトリネットを用いた異常対応方法を現実のプロセスへ適用する場合，次のようなステップが考えられる．

ペトリネットを用いた異常対応方法の現実のプロセスへの適用手順：

(1) 現実のバッチプロセスの正常な動作をペトリネットにてモデル化し，そのモデルから動作制約を導出する．

(2) 現実のプロセスが動作する間，新たな状態が観測されるたびに，動作制約の成否を確認する．

(3) 観測されたプロセスの状態が動作制約を満たさない場合，異常が発生したとし，運転の停止や条件変更等の必要な処置を行う．

(4) 検出された異常が未経験のものである場合，その異常を現在のペトリネットモデルに組み込む．

(5) 制御器を用いることで，検出された異常の再発の回避が可能であれば，それを実現する制御器をペトリネットモデルの上で設計する．

(6) 設計された制御器の妥当性を専門家の視点から確認し，必要であればシミュレーションにて検証を行う．

(7) 検証された制御器を現実の自動制御システム（DCS（Distributed Control System）や PLC（Programmable Logic Controller））に実装し，次回生産から，この異常の確実な回避を実現する． ■

上記(1)から(5)の内容は第 2 章から第 4 章に記載の内容である．(1)および(2)では対象とするバッチプロセスの正常な動作を表すペトリネットモデルから動作制約を導出し，それを用いて異常の検出を行う．(3)において異常が検出されたならば，現在進行中のプロ

セスはその運転の停止もしくは条件変更等の処置を行い，暫定的に異常を回避する．(4)では検出された異常の原因の特定のステップであり，これが未経験の異常によるものであれば，制御器の導入により異常の再発を回避できる可能性がある．(5)では検出された未経験の異常の回避が制御器によって可能であれば，それを実現する制御器をペトリネットモデルの上で設計するステップである．(6)は(5)で設計した制御器の検証のステップである．簡単な制御器であれば，専門家の判断で，その妥当性が評価可能であるが，複雑なプロセスの場合，シミュレーションによる検証が必要となることもある．(7)は制御器の実装のステップである．シーケンス制御用のプログラム言語（国際規格 IEC 61131-3）の一つである SFC（Sequential Functional Chart）はペトリネットに基づきに規格化されており，またこの言語は DCS および PLC で標準的に扱うことができることから，ペトリネット上で設計した制御器は SFC を用いて制御システムへ実装することが可能である．また，橋爪ら[16]の開発した CESeC (Condition/Event net based Sequence Controller synthesis support tool)を用いることも可能である．CESeC は PC 上に実現されたアプリケーションであり，アプリケーション上のペトリネットのトランジションの発火に伴い，制御バスを経由して弁やポンプ等の操作端の操作が可能となる．CESeC を用いれば，ペトリネットで設計した制御器を他のプログラム言語で再記述することなく，そのまま利用可能となる．

表 4-4　異常対応モデルとそれを用いた異常対応

	異常対応モデル	手法
異常の検出	バッチプロセスの正常な動作を表すペトリネットモデル	プレースインバリアント解析から動作制約を求め，動作制約を用い異常を検出
異常の特定	上記のモデルに異常を表すモデルを追加したペトリネットモデル	検出された異常を表す異常トランジションから異常を特定
異常の回避	上記のモデルに制御器を表すモデルを追加したペトリネットモデル	異常状態・異常動作回避問題を解くことで得られる制御器を用いて異常を回避

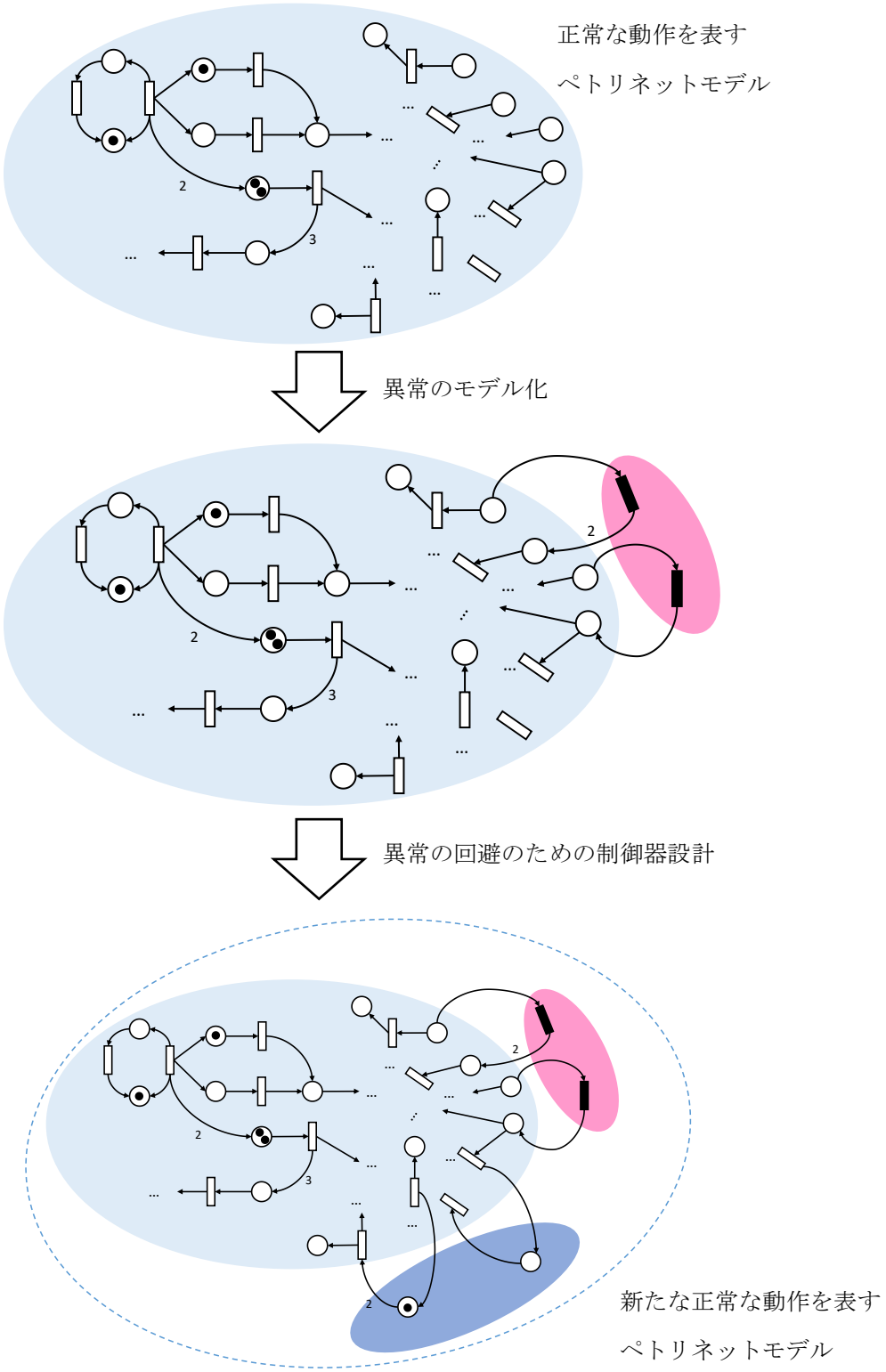


図 4-7　異常対応モデルの更新

4.4.2. 異常対応モデル構築に係る考察と今後の課題

本研究における異常対応モデル更新の考えは，バッチプロセスにおける運転を繰り返すたびに，その運転の安全性が逐次向上できることを意味する．また，本手法を現実のバッチプロセスにおいて，より効果的に運用するには，次の 3 つの方法が考えられる．

(1) 一般に，化学プロセスにおいては，商業運転前に試製造と呼ばれるサンプル品の生産と運転管理の確認のためのフェーズが存在する．バッチプロセスにおいては，複数のバッチサンプルを生産するための試製造にて商業運転と同等の操作を行い，異常が発生するたびに，都度修正を行うことで，商業運転における運転手法を明確にすることが行われる．この段階で本研究の手法を適用することで，効率的に安全性の向上を実現する制御器が設計できる．このことにより，商業運転時には，異常を回避した安全な運転が確保できるものと思われる．

(2) 近年発達の目覚ましいダイナミックプロセスシミュレータを用い，事前に異常の回避のための制御器を設計する枠組みが考えられる．ダイナミックプロセスシミュレータは，プロセスの動作を模擬できるツールである．シミュレータ上で構築したバッチプロセスを現実のものとみなし，その上で様々な異常を発生させながら異常対応モデルを更新していき，異常の回避のための制御器を設計する．そして，シミュレータ上で設計した制御器を現実のプロセスへ適用することで，現実のプロセスで異常を発生させることなく，異常を回避した運転が実現できると考えられる．

(3) 既存のプロセスに対しては，過去の運転データから異常を同定し，それをもとに異常の回避のための制御器を設計する枠組みが考えられる．2.5 で示したように，未経験の異常の組み込みは，正常状態と異常状態の情報（データ）から行っている．よって，過去の運転実績から，異常の発生時の運転データを抽出することで，その時に発生した異常をモデル化し，それを回避するための制御器の設計が可能となる．

一方，異常のモデル化にあたっては，事前にどのレベル（粒度）の異常を対象とするかを明確にすることが重要となる．細かい動作を表現する異常対応モデルからは，その細かな動作によって生じる異常とその回避のための制御器が導出される．よって，どのような異常を取り扱うか，そしてどのレベルの制御器が必要かを検討した上で，異常対応モデルの構築を進めることが必要となる．2.4 で述べたように，ペトリネットモデル

の特徴の一つとして階層表現に優れていることが挙げられる．現実のバッチプロセスを対象とする際は，この階層性を利用し，どのレベル（階層）において，どのような異常を検出し，どのような制御器が必要かを考え，異常対応モデルを構築することが望ましい．階層性の利用については，今後の課題である．

4.5. まとめ

バッチプロセスにおける異常の再発を確実に回避するための手順制御器や協調制御器の設計は経験者の知識やノウハウに依存する部分が多く，プロセスが大規模化，複雑化するに従い，この種の制御器の設計が困難となっていた．そこで本章においては，異常の再発を回避するための手順制御器や協調制御器の設計に係る検討を行った．

4.1 では，異常の回避を，プロセスの状態に着目し，状態が異常状態に陥ることを回避するとした異常状態の回避と，プロセスの動作に着目し，異常動作が行われることを回避するとした異常動作の回避の 2 つに分類し，それぞれ異常状態回避問題，異常動作回避問題を定式化した．4.2 では，異常状態回避問題をバッチプロセスの状態が常に動作制約をみたすように制御器を設計する問題としてペトリネット上で定式化し，バッチプロセスの状態遷移に着目した解法を示した．4.3 では，冷却プロセスを例に，異常の検出，検出された異常の原因の特定，特定された異常の再発の回避に至る一連の異常対応を，同一のペトリネットモデルのもとで解決可能であることを示した．

第 2 章で述べたように，バッチプロセスでは，同じ種類のバッチを同じ手順で製造することを必要とされるバッチ数繰り返す．よって，モデル化された異常の回避を実現する制御器の設計ができれば，異常の再発の回避という観点で有効な手段となると考えられる．また，4.3 で示した適用例のように，種類の異なるバッチでも共通して使用する装置に係る異常の回避であれば，本手法の有効性は更に高まるものと考えられる．

第 5 章では異常の回避のもう一つの方法である，異常動作の回避に着目し，異常動作回避問題に係る検討を行う．

第5章.バッチプロセスにおける異常動作の回避

第 4 章ではバッチプセスにおける異常の回避の二通りの考え方を示し，そのうちプロセスの状態遷移に着目した異常状態の回避に係る検討を行った．本章においては，バッチプロセスの動作に着目した異常の回避に係る検討を行う．

5.1. はじめに

第4章で示したように,本研究ではバッチプロセスにおける異常の回避の考え方を,

異常状態の回避：状態が異常状態に陥ることを回避する．

異常動作の回避：異常動作が行われることを回避する．

の 2 つに分類した．そして，異常の回避のための制御器の設計問題を，これらの異なる着眼点から，それぞれ異常状態回避問題 PS，異常動作回避問題 PD として，これをとらえた．

異常状態回避問題 PS：”プロセスの状態が常に動作制約を満たすように動作すること”を制御目的にもつ制御器を設計する問題

異常動作回避問題 PD:”プロセスの動作が異常動作を行うことなく常に正常に動作すること”を制御目的にもつ制御器を設計する問題

第 4 章でも示したように，これら 2 つの問題は本質的には同等であることは明らかである．しかし，現実のバッチプロセスは多種多様な特性をもつため，その特性合わせた制御器の設計手法を選択し，これを進める必要がある．第 4 章においては，このうち問題 PS をペトリネットの上で定式化し，バッチプロセスの状態の遷移に着目した解法を示した．本章では，問題 PD に係る検討を行う．問題 PD においては，異常動作が発生する前に正常な動作を制御することで，“間接的”に異常動作の発生を回避する．

本章の構成は次の通りである．5.2 では問題 PD をペトリネットのサブクラスである条件／事象ネット（C/E ネット（Condition/Event Net）を用いて定式化する．そして，定式化した問題に対し，著者らの研究の一つである C/E ネット制御問題に係る考えを用いることでその解法を導出する．一方，問題 PS と異なり問題 PD では開発した解法を用いて得られる制御器は一意に決まらないため, 複数ある制御器の中から何かしらの

基準を従い，最適な制御器を選ぶ必要がある．本研究では，異常の未然の回避という観点から，できるだけ構成が簡易な制御器を選択することとした．5.3 では，著者らの研究の一つである C/E ネット設計問題に係る考えを用い，構成が最も簡易な制御器を求める問題を定式化し，その解法を提案する．5.4 においては，異常の回避における異常状態の回避，異常動作の回避の 2 つの方法についてまとめる．

5.2. 異常動作回避問題

5.2.1. ペトリネットに基づく異常動作回避問題の定式化

4.1 に示した異常動作回避問題 PB は，正常な動作を制御目的として与え，そのもとでその動作のみを行う制御器の設計を試みるものである．したがって，この問題を定式化するにあたっては，バッチプロセスの動作を正しく記述することが重要となる．第 2 章で示したように，本研究ではバッチプロセスの動作を離散事象システムとしてとらえ，次にこれをペトリネットで表現することとした．したがって，問題 PB をペトリネット上で定式化するにあたっては，離散事象システムとしてとらえたバッチプロセスの動作を，ペトリネットが理解できる形で表現することが必要となる．Grabowski[25]が提案した半言語はこのための 1 つの表現形式である．半言語は半順序多重集合である半語の集まりであり，一つ一つの半語は先行関係をもつ複数の動作から構成される一連の動作群を表す．そこでは，2.4 で示したようにバッチプロセスに頻出する逐次的な動作，並行的な動作，非決定的な動作を正しく表現することができる．このため，本研究においては，バッチプロセスの望みの動作である動作仕様を，この半言語を使って表すこととした．なお，半言語の詳細については付録 2 に示す．

以上のもとで，本研究では，異常動作回避問題を次のように定式化することとした．

異常動作回避問題 PB：

バッチプロセスの動作を表すプラントネットN_Pとその望みの動作（異常事象の発生を含まない動作）を表す半言語Xが与えられたとき，閉ループシステムネット$N_P \oplus N_C$の動作がこの望みの動作のみ行うような制御器ネットN_Cを求めよ． ■

この問題を具体化するために，次のネット半言語およびアクティビティを定義する．

ネット半言語：

ペトリネット$N=(P,T,F,W,M_0)$から得られるトランジションの発火形態（半言語）を$PL(N)$とし，Nのネット半言語とする． ■

アクティビティ：

ペトリネットNのネット半言語$PL(N)$の部分集合を考え，部分集合内のすべての半語が他の半語の接頭半語でなく，且つ，他の半語を順序付けすることで得られる半語ではない場合，このような部分集合$FP(N)$をペトリネットNのアクティビティと呼ぶ．この$FP(N)$を用いることで，ペトリネットNの動作をすべて表現できる． ■

問題 PB を半言語，アクティビティを用いて具体化すると次のように定式化される．

異常動作回避問題 PB：

バッチプロセスの動作を表すプラントネットN_Pとその動作仕様（異常事象の発生を含まない動作）としてプラントトランジションの集合T_P上の半言語Xが与えられたとき，$X=FP(N_P \oplus N_C)$を満たすような制御器ネットN_Cを求めよ． ■

5.2.2. C/E ネット制御問題の定式化とその解法

本項では，5.2.1 で定式化した異常動作回避問題 PB の解法に係る検討を行う．現実のバッチプロセスを離散事象システムとしてとらえると，そこに現れる離散的な状態量は，“ある状態量が指定された値に到達した”，“ある条件が成立した”等のように 2 値変数として表現されることが多い．また，たとえ，それが 3 値以上の離散変数であってもそれを複数個の 2 値変数で表現することもできる．したがって，煩雑にはなるが，任意のペトリネットは，すべてのプレースの容量が 1 であるペトリネットの 1 つのサブクラスである C/E ネットで表現することができる．

橋爪，小野木らは，対象プラントの動作を C/E ネットにて表現し，制御の対象であるプラントが望みの動作のみを行うような制御器を設計する問題を定式化し，この問題の性質とその解法を示した[16][17][18]．本研究においても，問題 PB の解法を考えるにあたり，橋爪，小野木らの提案した手法を用いる．なお，橋爪らは同様の問題に対し，ペトリネットを用いた検討も行っている[19]．

5.2.1 で定式化した問題 PB を C/E ネットを用いて表現すると，次の C/E ネット制御問題P_C^{ce}となる．ただし，$N_P=(P_P,T_P,F_P,\boldsymbol{M}_{P0})$は C/E ネットで表現されたバッチプロ

セスの動作を表すプラントネットで，$N_C = (P_C, T_C, F_C, \boldsymbol{M}_{C0})$は C/E ネットで表現された制御器の動作を表す制御器ネットである．なお，C/E ネットのすべてのアークの重みは 1，すなわち$W_P: F_P \to \{1\}$ $(W_C: F_C \to \{1\})$より，C/E ネットにおけるN_P，N_Cはそれぞれ上に示した 4 項組で表される．

C/E ネット制御問題P_C^{ce}：

バッチプロセスの動作を表すプラントネットN_Pとその動作仕様（異常事象の発生を含まない動作）として，プラントトランジションの集合T_P上の半言語Xが与えられたとき，(5.2)式のもとで(5.1)式を満たすような御器ネットN_Cを求めよ．

$$X = FP(N_P \oplus N_C) \tag{5.1}$$

$$T_C \subseteq T_P \tag{5.2}$$ ■

(5.2)式は，問題P_C^{ce}の解となるN_Cの制御器トランジションの集合T_Cが，プラントトランジションの集合T_Pの部分集合であることを示している．このことから，制御器はバッチプロセスの動作をトリガーとして動作することを示している．

問題P_C^{ce}の解法を述べる前に，いくつかの用語を定義する．

アトム：

C/E ネットの中でプレース数が 1 の C/E ネットをアトム$n = (P_n, T_n, F_n, M_{n0})$（ただし$|P_n| = 1$）と呼ぶ．また，$S$をアトムの集合とした際，$S$中のアトムをすべて合成して得られる C/E ネットを$\bigoplus S$で表す． ■

冗長なアトム：

アトムの集合Sとし，あるアトム$n \in S$に対し，$FP\big(\bigoplus(S - \{n\})\big) = FP(\bigoplus S)$であるとき，$n$は C/E ネット$\bigoplus S$に対し冗長であるという．すなわち，冗長なアトムは，それを削除してもネットの動作は変わらない． ■

相補的なアトム：

アトムnの入力トランジションと出力トランジションを互いに入れ替え，トークンの有無を逆にしたアトムn'をnの相補的なアトムという．nとn'の間には，$FP(n) = FP(n')$の関係が成り立つ． ■

半言語の制限：

半語の制限とは，半語$pw = (U, R, \pi)$に対し，次のように定義される．

$$pw' = pw|\Sigma' = \left(U', R \cap U'^2, \pi'\right)$$

ただし，$\Sigma' \subseteq \Sigma$，$U' = \{x \in U \mid \pi(x) \in \Sigma'\}$，$\pi' : U' \longrightarrow \Sigma'$である．半語の制限を簡単に述べると，もとの半語$pw$から$\Sigma' \subseteq \Sigma$に関わる部分のみ抽出して得られる半語$pw'$のことである． ■

適合するアトム：

トランジションの集合T上の半言語Xに適合するアトム$n = (P_n, T_n, F_n, M_{n0})$（ただし，$T_n \subseteq T$）とは，次の式を満たすアトムである．

$$X \mid T_n \subseteq PL(n)$$

すなわち，半言語Xに対し，アトムを構成するトランジションT_nで制限をかけた半言語が，アトムnの部分集合であるとき，アトムnは半言語Xに適合するという．また，半言語Xに適合するすべてのアトムの集合を$CA(X)$とする． ■

これらの準備のもと，橋爪らは次の 2 つの命題を明らかにした[46][47]．

命題 5-1：

$N = N_1 \oplus \cdots \oplus N_m$とし，$X$をトランジションの集合$T$上の半言語とする．このとき，$X \subseteq PL(N) \Leftrightarrow X|T_i \subseteq PL(N_i) \quad (i = 1,2,\cdots,m)$が成り立つ．ただし，$T_i$をネット$N_i$のトランジションとし，$T = T_1 \cup \cdots \cup T_m$とする． ■

命題 5-2：

S_1，S_2をアトムの集合とし，$S_1 \subseteq S_2 \subseteq CA(X)$とする．もし，$\mathrm{X} = FP(\oplus S_1)$ならば，$\mathrm{X} = FP(\oplus S_2)$である． ■

そして，これらの命題より，問題P_C^{ce}の解の存在性に関し，次の(1)，(2)が成り立つことを明らかにした．ここで$CA(X)$はT_P上の半言語で表される動作仕様Xに適合するすべてのアトムの集合で，$\oplus\, CA(X)$は$CA(X)$の要素をすべて合成することで得られるネットで，S_Pは$N_P = \oplus\, S_P$を満たすバッチプロセスの動作を表すアトムの集合である．

問題P_C^{ce}の解の存在性：

(1) $X = FP\big(\oplus CA(X)\big)$となる場合，問題$\mathrm{P}_C^{ce}$の解が存在し，$N_C = \oplus\, CA(X) - N_P$が解の一つとなる．

(2) $\mathrm{X} \neq FP\big(\oplus CA(X)\big)$となる場合，問題$\mathrm{P}_C^{ce}$の解は存在しない． ■

問題P_C^{ce}の解の存在性から，次に示す解法が導かれる[46][47]．

問題P_C^{ce}の解法：

Step 1:

$CA(X)$を求める．

Step 2:

$S_P \subseteq CA(X)$であれば，Step3 へ進む．$S_P \nsubseteq CA(X)$であれば，問題P_C^{ce}に解は存在しない．

Step 3:

$X = FP\left(\oplus CA(X)\right)$であれば， $N_C = \oplus CA(X) - N_P$が問題P_C^{ce}の解の一つとなる．そうでなければ，問題P_C^{ce}の解は存在しない．

Step 4:

$X = FP(\oplus S)$を満たすアトムの集合$S_P \subseteq S \subseteq CA(X)$を見つけ，$N_C = \oplus S - N_P$とする． ■

Step 2 において，命題 5-1，命題 5-2 から$X = FP(N)$となるネット Nは，$\oplus CA(X)$から冗長なアトムを削除したネットであり，$S_P \nsubseteq CA(X)$となる場合，問題P_C^{ce}の解は存在しないこととなる．Step 3 において， $N_C = \oplus CA(X) - N_P$が解の一つとなる．また，Step3 で得られたN_Cから冗長なアトムを取り除くことで得られるすべてのネットも問題の解となる．Sの導出の仕方は一意には決まらないため，提案した解法においては，制御器は一意に決まらない．これに関しては，5.3 で検討を行う．

また，問題P_C^{ce}は必ずしも解をもつとは限らない．5.2.3 では，より多くの異常の回避を実現するため，問題P_C^{ce}の拡張を検討する．

例題 5-1：

図 5-1 に示す C/E ネットで表現されたプラントネットN_Pを考える．N_Pには異常トランジションfがモデル化されている．原理的にはN_Pの動作$FP(N_P)$を求め，その半言語の中から， fを含む半言語をすべて取り除くことで，異常fのみを回避する制御器が設計される．ここは，簡単な例として，図 5-2 のプラントトランジションの集合T_P上の半語で表される異常を含まない動作仕様X_1を考える．

問題P_C^{ce}の解法で示したように，まず，動作仕様X_1に適合するすべてのアトムの集合$CA(X_1)$を求めると図 5-3 が得られる（Step1）．次に$CA(X_1)$がS_Pを含んでいるかを確認すると，図 5-3 から$S_P \subseteq CA(X_1)$であることが分かる（Step2）．ここで，トランジションa，eを含むアトム（図 5-3 左上のアトム）は，プラントネット上では，相補的なアト

ムとなっていることに注意する．そして，動作仕様X_1に適合するアトムの集合$CA(X_1)$の要素をすべて合成した C/E ネット$\oplus CA(X_1)$を求めると，$X_1 = FP\left(\oplus CA(X_1)\right)$となる．このことから，問題の解の一つは$N_{C1} = \oplus CA(X_1) - N_P$である（Step3）．一方，得られた$\oplus CA(X_1)$から冗長なアトムを取り除いた解の一つが図 5-4 となる．図 5-4 の破線で示した部分が制御器ネットN_{C1}を表している．この閉ループシステムネット$N_P \oplus N_{C1}$の動作$FP(N_P \oplus N_{C1})$は図 5-5 であり，異常を回避していることが確認できる．この制御器ネットN_{C1}はトランジションb，dの発火に順序を付けることで，異常の発生を間接的に回避している． ■

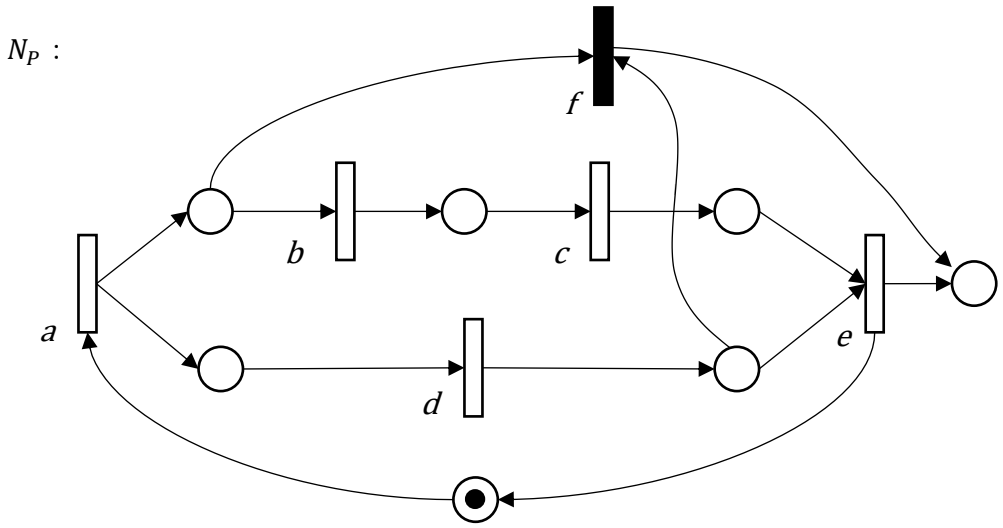


図 5-1　プラントネットN_P

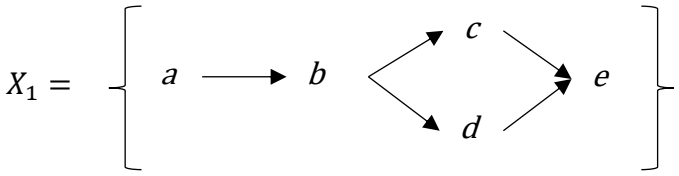


図 5-2　異常を含まない動作仕様X_1

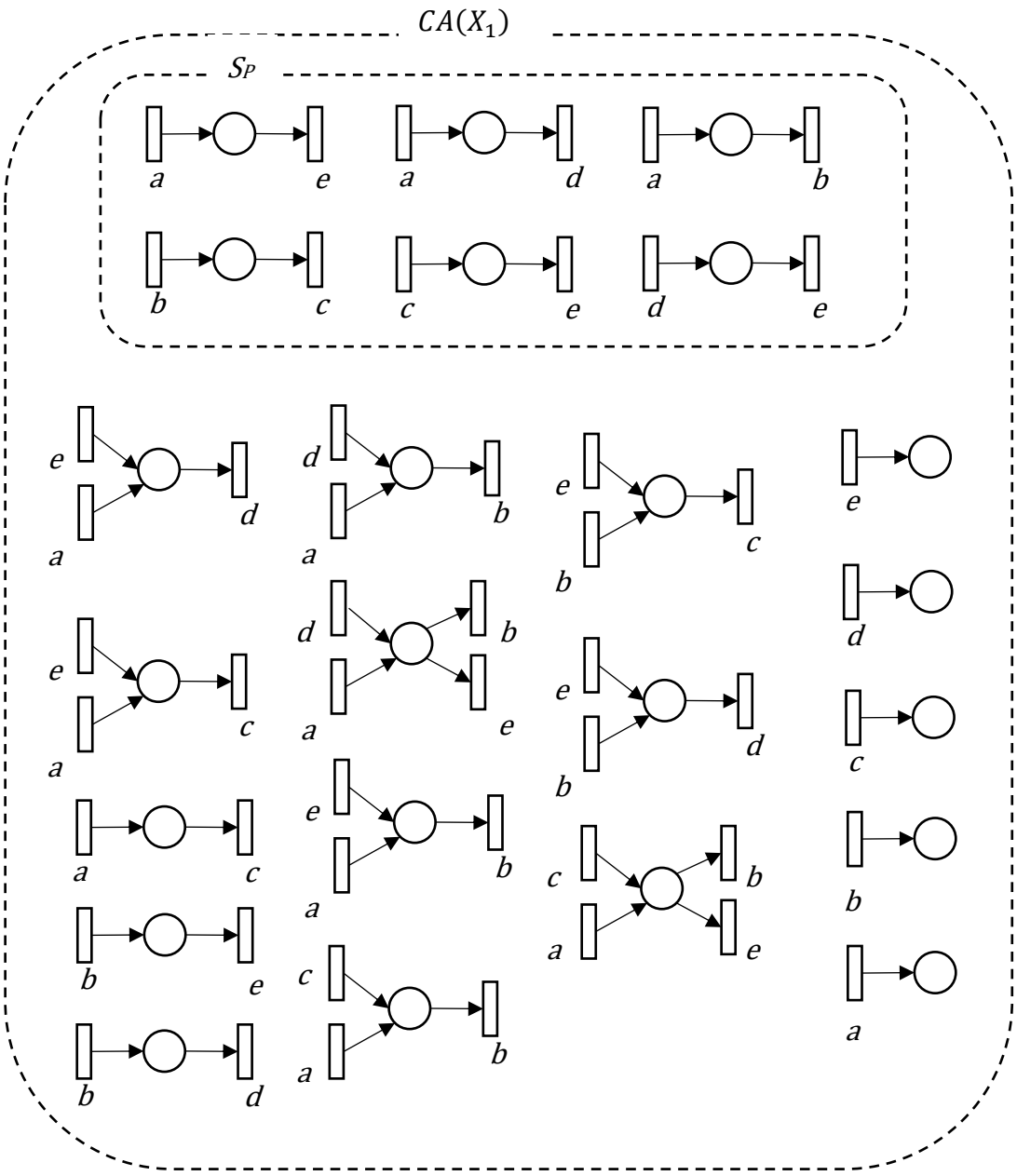


図 5-3　動作仕様X_1に適合するアトムの集合$CA(X_1)$

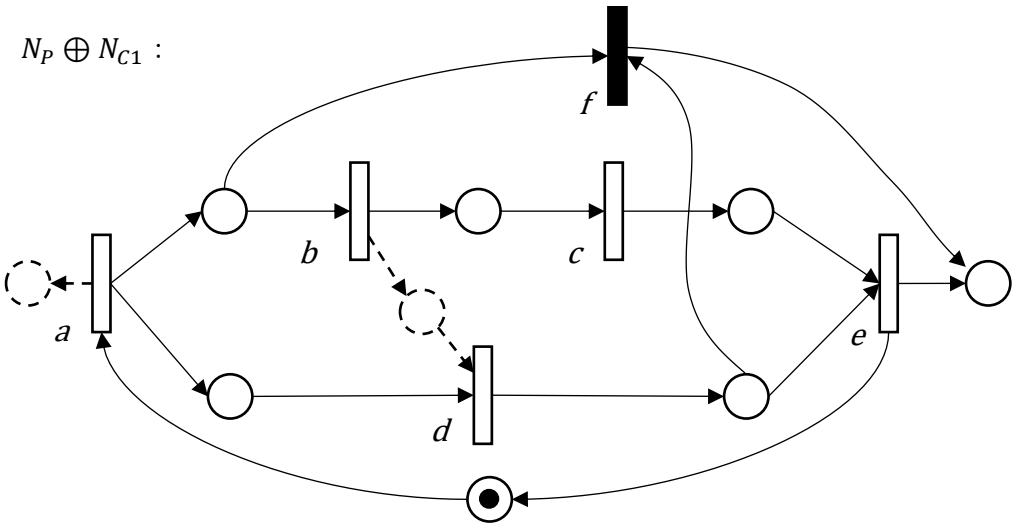


図 5-4　閉ループシステムネット$N_P \oplus N_{C1}$

$$FP(N_P \oplus N_{C1}) = \left\{ a \longrightarrow b \begin{array}{l} \nearrow c \\ \searrow d \end{array} \begin{array}{l} \searrow \\ \nearrow \end{array} e \right\} = X$$

図 5-5　閉ループシステムネットの動作$FP(N_P \oplus N_{C1})$

5.2.3.　拡張 C/E ネット制御問題の定式化とその解法

C/E ネット制御問題P_C^{ce}は必ずしも解をもつとは限らない．そこで本項では問題P_C^{ce}を拡張することを考える．5.2.2 で定式化した問題P_C^{ce}は，バッチプロセスが異常動作を含まない望みの動作となるよう，制御器プレースを新たに追加し，もとのバッチプロセスの動作を抑制した．よって，問題P_C^{ce}で得られる制御器はバッチプロセスの動作をトリガーとして動作するものであった．ここで，制御器が現実のバッチプロセスの動作のみではなく，制御器内で作動する動作に対しても同様の振舞いを行うことを考える．すると，この制御器はバッチプロセスの動作をトリガーとし制御器内での動作を操作する，制御器内での動作をトリガーとしバッチプロセスの動作を操作する，制御器内での動作をトリガーとし制御器内の別の動作を操作する，というようにその操作範囲が拡張される．これに伴い回避できる異常は問題P_C^{ce}に比べ増えると考えられる．

制御器内の動作を表すトランジションを扱えるよう，制御器トランジションの集合T_Cを，(5.3)式のように拡張する．

$$T_C = T_{CP} + T_{CC} \quad \text{ただし } T_{CP} \subseteq T_P \ , \ T_{CC} \cap T_P = \phi \tag{5.3}$$

ここで，T_{CP}はプラントトランジションの集合T_Pの部分集合となるトランジションの集合で，T_{CC}は制御器内の動作を表すトランジションの集合であり，+は直和を表す．(5.3)式のもと，問題P_C^{ce}を次のように拡張する．

拡張 C/E ネット制御問題$\mathbf{P}_{EC}^{ce}$：

バッチプロセスの動作を表すプラントネットN_Pとその動作仕様（異常事象の発生を含まない動作）としてプラントトランジションの集合T_P上の半言語Xが与えられたとき，(5.5)式のもとで(5.4)式を満たすような制御器ネットN_Cを求めよ．

$$X = FP(N_P \oplus N_C)|T_P \tag{5.4}$$

$$T_C = T_{CP} + T_{CC},\ T_{CP} \subseteq T_P,\ T_{CC} \cap T_P = \emptyset \tag{5.5}$$■

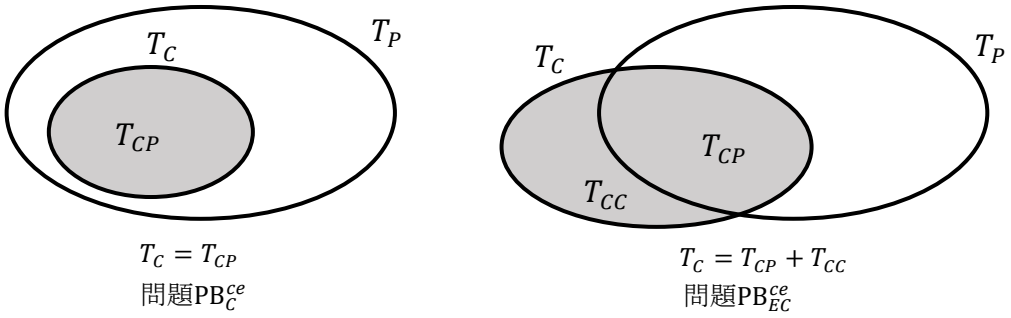
問題P_{EC}^{ce}は制御プレースだけでなく，制御器内の動作を表すトランジション$s \in T_{CC}$もN_Pに追加する．そして，閉ループシステムネットの動作からT_{CC}の要素を取り除くこと得られる動作が目的の動作となるように，制御器を設計する．

図 5-6, 表 5-1 に問題P_C^{ce}と問題P_{EC}^{ce}の違いをまとめる．(5.3)式から明らかなように，問題P_C^{ce}ではT_CとT_{CP}が一致する．また，$T_{CC} = \emptyset$とすると，問題P_{EC}^{ce}は問題P_C^{ce}と一致することは明らかであり，問題P_C^{ce}は問題P_{EC}^{ce}の特殊な場合とみなすことができる．

表 5-1　問題$\mathbf{P}_C^{ce}$と問題$\mathbf{P}_{EC}^{ce}$における制御器ネット

	制御器トランジション T_C	望みの動作の生成
問題PB_C^{ce}	T_{CP}	$X = FP(N_P \oplus N_C)$
問題PB_{EC}^{ce}	$T_{CP} + T_{CC}$	$X = FP(N_P \oplus N_C)\|T_P$

制御器ネット$N_C = (P_C, T_C, F_C, \boldsymbol{M}_{C0})$　$T_{CP} \subseteq T_P, T_{CC} \cap T_P = \phi$



$T_C = T_{CP}$
問題PB_C^{ce}

$T_C = T_{CP} + T_{CC}$
問題PB_{EC}^{ce}

図 5-6　問題$\mathbf{P}_C^{ce}$と問題$\mathbf{P}_{EC}^{ce}$におけるトランジションの集合

次に問題P_{EC}^{ce}の解法を示すため，拡張適合アトムを定義する．

拡張適合アトム：

トランジションの集合T上の半言語Xに拡張適合するアトム$n = (P_n, T_n, F_n, M_{n0})$（ただし, $T_n \subseteq T$）とは, $X \mid T_n \subseteq PL(n)|T$を満たすアトムである. ■

定義から明らかなように, Xに適合するアトムは拡張適合する. この拡張適合アトムに係る 2 つの命題を以下に示す.

命題 5-3：

$N = N_1 \oplus \cdots \oplus N_m$とし, Xをトランジションの集合T上の半言語とする. このとき, $X \subseteq PL(N)|T \Leftrightarrow X|T_i \subseteq PL(N_i)|T \quad (i = 1,2,\cdots,m)$が成り立つ. ただし, T_iをネットN_iのトランジションとし, $T \subseteq T_1 \cup \cdots \cup T_m$とする. ■

命題 5-4：

Xをトランジションの集合T上の半言語とし, $T' \supset T$とする. また, NをT'の部分集合をトランジションの集合とするアトムで, Xに拡張適合するアトム全体を合成して得られる C/E ネットとする. このとき, $PL(N) = \emptyset$である. ■

命題 5-3 より, C/E ネットNのネット半言語がXにである場合, そのネットはXに拡張適合するアトムから構成されることが分かる. また, 命題 5-4 より, 問題P_{EC}^{ce}の解が存在するかどうかは, 問題P_{C}^{ce}の場合と異なり, Xに拡張適合するアトムをすべて合成しても確かめられないことを意味する.

これらの命題から, 問題P_{EC}^{ce}の解の存在性に関し, 次の(1), (2)が導かれる. ここで, $CA(X)$はT_P上の半言語で表される動作仕様Xに適合するすべてのアトムの集合で, $ECA(X)$はXに拡張適合するすべてのアトムの集合で, S_Pは$N_P = \oplus S_P$を満たすバッチプロセスの動作を表すアトムの集合である.

問題P_{EC}^{ce}の解の存在性：

(1) $S_P \subseteq CA(X)$ ならば, 問題P_{EC}^{ce}の解が存在するための必要十分条件は, $X = FP\left(\oplus\,(S_P \cup S_C)\right)|T_P$が成り立つような$S_C \subseteq ECA(X)$が存在することである.

(2) $S_P \not\subseteq CA(X)$ならば, 問題P_{EC}^{ce}の解は存在しない. ■

問題P_{EC}^{ce}の解の存在性より, 問題P_{EC}^{ce}の解となるN_Cは, $ECA(X)$を求め, N_Pに$ECA(X)$の要素を幾つか合成して得られる C/E ネットの動作がXに一致するような拡張適合アトムの組合せを見つけることで得られる. このことから, 次の解法が考えられる.

問題P_{EC}^{ce}の解法：

Step 1: ：

$CA(X)$を求める．

Step 2:

$S_P \subseteq CA(X)$であれば，Step 3 へ進む．$S_P \nsubseteq CA(X)$であれば，問題P_{EC}^{ce}に解は存在しない．

Step 3:

$X = FP\left(\oplus CA(X)\right)$であれば，Step 4 に進む．そうでなければ，$l = 1$として Step 5 に進む．

Step 4:

$X = FP(\oplus S)$を満たすアトムの集合$S_P \subseteq S \subseteq CA(X)$を見つけ，$N_C = \oplus S - N_P$とし，終了する．

Step 5:

T_{CC}の要素数がl以下の$ECA(X)$を求める．

Step 6:

$X = FP\left(\oplus\ (S_P \cup S_C)\right) | T_P$が成り立つような$S_C \subseteq ECA(X)$を探索する．もし，そのような$S_C$が求められれば$N_C = \oplus S_C$として終了する．そうでなければ，$l = l + 1$として，Step 5 へ戻る．(Step 5, Step 6 をあらかじめ定められた回数内で繰り返す)． ■

Step 6 において，原理的には$ECA(X)$の要素の組合せ問題を解くこととなるため，$ECA(X)$の要素数が増えるほど，計算量は組合せ爆発的に増大する．現実的な解法として，Step 6 で記述したように，$ECA(X)$の導出およびそれを用いた解の探索は，T_{CC}の要素数を徐々に増やしながら，あらかじめ定められた回数で行うこととする．効率的な解の探索法の検討は，今後の課題である．

例題 5-2：

例題 5-1 と同じ構造のプラントネットN_Pにおいて，図 5-7 に示す異常を含まない動作仕様X_2を考える．問題 5-1 の動作仕様X_1と異なり，トランジションeの発火後，再度トランジションaを発火し，トランジションbとトランジションdを並行に発火させる動作となる．この動作仕様X_2に適合するアトムの集合$CA(X_2)$のすべての要素を合成して得られた C/E ネット$\oplus\ CA(X_2)$の動作$FP\left(\oplus\ CA(X_2)\right)$は動作仕様$X_2$と一致しない．そこで$T_{CC} = \{s\}$とし，$X_2$に拡張適合するアトムの集合$ECA(X_2)$を求める．ここでは 108 の要

素からなる$ECA(X_2)$が求められる. そして, $X_2 = FP\big(\oplus\,(S_P \cup S_C)\big)|T_P$となる$S_C \subset ECA(X_2)$を求めると，図 5-8 の破線で示す制御器ネット$N_{C2}$が得られる. ■

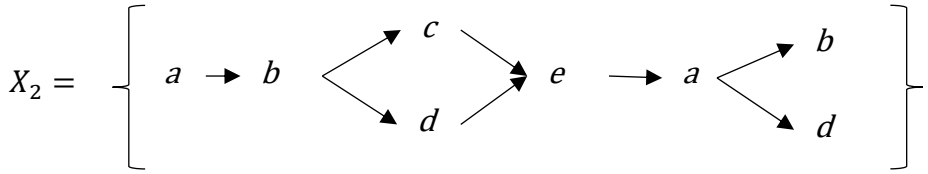
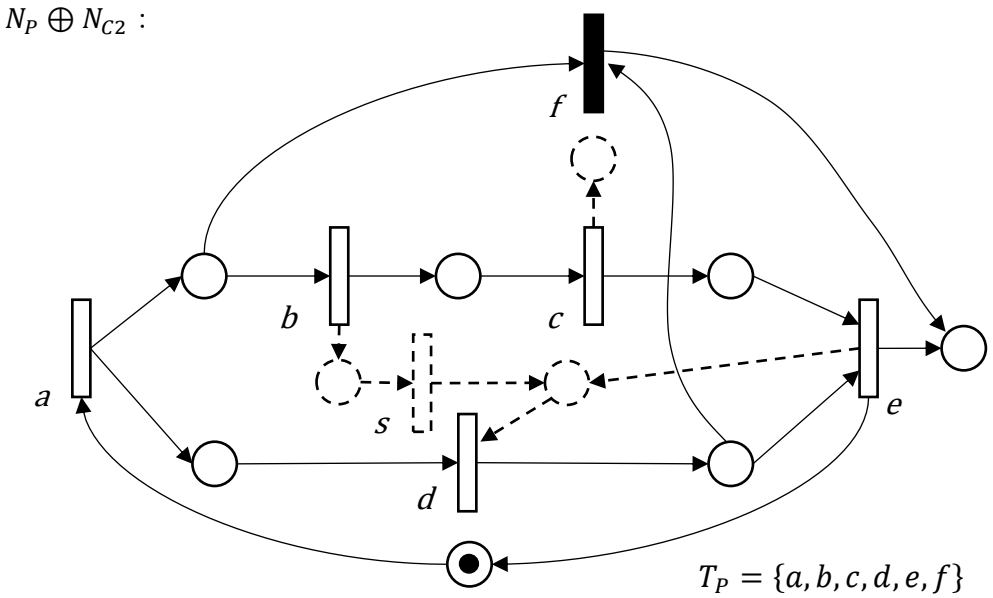


図 5-7　異常を含まない動作仕様X_2



$FP(N_P \oplus N_{C2})|T_P =$

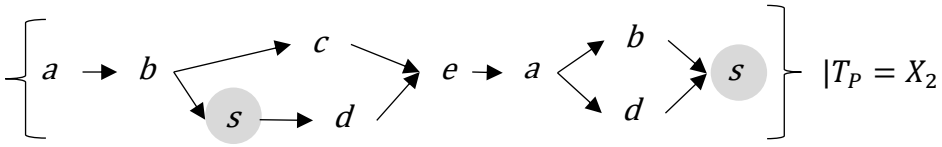


図 5-8　制御器ネットN_{C2}を合成した閉ループシステムネット$N_P \oplus N_{C2}$

5.3. C/E ネット最小実現問題

5.3.1. C/E ネット最小実現問題の定式化

バッチプロセスにおける制御器を設計するにあたり，制御器の構成を決定する，すなわち，制御目的の達成のため，どの制御量を観測し，どの操作量を用いて制御するかを決定する必要がある．制御器の構成を決定することの難しい点は，バッチプロセスにおける多く制御量，操作量の候補の中から，制御目的を達成するための最適な組合せを選ぶ必要があることが挙げられる．また，制御器自身も次に示す理由により，その構成が簡易なものが望ましいと考えられる．

構成が簡易な制御器が望ましい理由：

- 制御器を構成する情報量が少ないほど制御器の更新や日々のメンテナンスが容易になる等，管理・運用面での利点がある．
- 制御器を構成する制御量が複数ある場合，それを観測するセンサーのうち，どれか一つが故障することで制御器が正常に動作しなくなる．そのため必要最小限の制御量が望ましい．
- バッチプロセスの特性の一つにプロセスのもつ非線形性やプロセス変数間の因果関係の変動があり，制御器に必要以上の多くの情報を取り込むほど，これらの変動に対する影響を受けやすくなり，場合によっては制御器が正常に動作しなくなる．そのため必要最小限の制御量が望ましい． ■

これらのことより，構造な簡易な制御器を用いることは，異常を未然に回避することに繋がるように思われる．しかし，本来簡易な構成が望まれるものの，どの制御構成を選択するかは，専門家の判断に任されていた面も多い．

5.2.2 では，異常の回避を目的とした異常動作回避問題 PB において，それをペトリネットのサブクラスである C/E ネットで定式化し，C/E ネット制御問題P_C^{ce}とした．そして，その解法を示し，そこから求められる制御器は一意に定まらないことを述べた．現実のバッチプロセスへの適用を考えた場合，問題P_C^{ce}の解（制御器）候補のうち，異常を未然に回避するという観点から構成が最も簡単な制御器を選択するという考え方がある．これは，5.2.2 で示した C/E ネット制御問題P_C^{ce}の解の候補となる制御器ネットN_Cのうち，プロセス変数の状態を表すプレースの数が最も少ないN_Cを求める問題へと帰着される．

既往の研究として，松谷，橋爪らは“望みの動作が半言語Xで与えられたとき，$X = FP(N)$となる実行可能ネットNを構成する問題”として C/E ネット合成問題P_S^{ce}を定式化した[28]．そして，橋爪らは問題P_S^{ce}の解となる実行可能ネットのうちプレース数が最小となる C/E ネットN_{min}を求める問題として，次に示す C/E ネット最小実現問題$\mathrm{P}_{S,min}^{ce}$を定式化した[48]．

C/E ネット最小実現問題$\mathrm{P}_{S,min}^{ce}$：

システムの望みの動作を表す半言語Xが仕様として与えられたとき，$X = FP(N)$を満たす実行可能ネットNのなかでプレース数が最小のものを求めよ． ■

問題$\mathrm{P}_{S,min}^{ce}$を解くことで得られるN_{min}は，問題P_C^{ce}における閉ループシステムネット$N_P \oplus N_C$とみなすことができる．よって，N_{min}からN_Pを取り除くことで，プレース数が最も少ないN_Cを得ることができる．このことから，異常動作の回避のための制御器のうち，その構成が最も簡単な制御器を求める問題とその解法は，橋爪らの提案した問題$\mathrm{P}_{S,min}^{ce}$の考えを用いることができる．5.3.2 では，問題$\mathrm{P}_{S,min}^{ce}$の解法について検討する．

5.3.2. C/E ネット最小実現問題の解法

C/E ネット最小実現問題$\mathrm{P}_{S,min}^{ce}$の解法の原理的な考え方として，動作仕様Xに適合するアトムの集合$CA(X)$を求め，それらの要素をすべて合成した C/E ネット$\bigoplus CA(X)$を求める．次に，冗長なアトムをすべて削除して最小実現となる$S \subseteq CA(X)$を求め，$\bigoplus S$を問題の解とするものである．しかし，Xに適合するアトムの数$|CA(X)|$は，$|T_X|$（ただし，$|T_X|$は X に含まれるトランジション数とする）の増加に伴い，$2^{|T_X|}$で増加する．また，最小実現に関わる計算量は，$|CA(X)|$の増加とともに，$2^{|CA(X)|}$で増加する．すなわち，問題$\mathrm{P}_{S,min}^{ce}$は本質的に組合せ問題であり，組合せ爆発を避ける効率的なアルゴリズムが必要となる．

本研究で開発した解法を述べる前に，次の命題を述べる．ここで，動作仕様Xにおいて，二つのトランジションt_1，t_2がX上で隣接しているとき，t_1，t_2は直接の先行関係にあるという．例えば図 5-9 では，点線で示したようにX上でgとcが隣接している．このとき，gとcは直接の先行関係にあると言う．また，Xの直接の先行関係にあるすべてのトランジションの組t_1，t_2について，t_1，t_2の両者を共にもつアトムが，あるアトムの集

合Sに含まれるとき，SはXの直接の先行関係を被覆するという．このとき，橋爪らは次の三つの命題を明らかにした[48].

命題 5-5：

仕様Xにおいてt_1，t_2が直接の先行関係にあるならば，問題$\mathrm{P}_{S,min}^{ce}$の実行可能ネットを構成するアトムの中にt_1，t_2の両者を共にもつアトムが存在する． ■

命題 5-6：

Sを$CA(X)$の部分集合とし，$N = \oplus S$とする．このとき，$T_N = T_X$でSがXの直接の先行関係を被覆するならば，$X \subseteq FP(N)$である．ただし，T_NをネットNに含まれるトランジションの集合とする． ■

命題 5-7：

$S, S' \subseteq CA(X)$とし，$N = \oplus S$および$N' = \oplus S'$とする．このとき，$T_N = T_{N'}$で$S \subseteq S'$ならば，$PL(\oplus S) \supseteq PL(\oplus S')$である． ■

命題 5-5 に関し，二つのトランジションがXにおいて隣接しているならば，実行可能ネットにおいてそれらは一つのプレースを介して接続されていることを表す．命題 5-6 に関し，Xにおいて隣接している二つのトランジションが一つのプレースを介して接続されている C/E ネットでは，Xが発生可能であることを表す．命題 5-7 に関し，任意の C/E ネットにプレースを追加していくと，その動作は徐々に限定されていくことを表す.

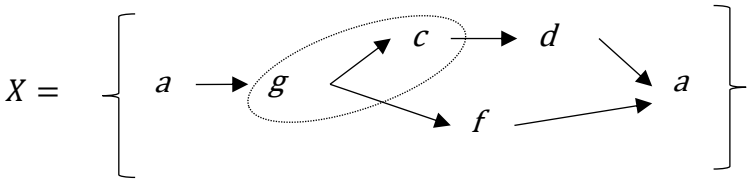


図 5-9　動作仕様Xの直接の先行関係の例

この三つの命題から，次の解法が考えられる．

解法の基本的な考え方：

Step 1:　（命題 5-5,5-6）

Xの直接の先行関係を被覆するXに適合するアトムの集合$S \subseteq CA(X)$を求める．$\oplus S$は$FP(\oplus S) \supseteq X$となる．

Step 2: （命題 5-7）

Xに適合するアトムn_1，…をSに加えて$PL(\oplus S) \supseteq \cdots \supseteq PL(\oplus S')$としていき，ちょうど$FP(\oplus S') = X$となるアトムの集合$S' = S \cup \{n_1, \cdots\} \subseteq CA(X)$を求める．

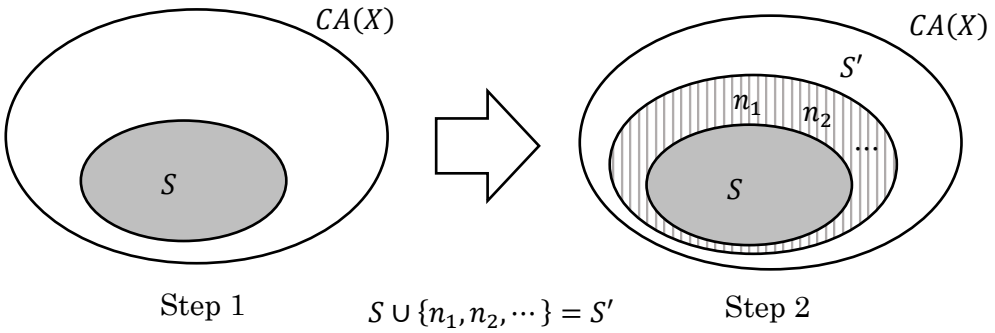
Step 3:

Step 1 で得た$\oplus S'$に含まれる冗長なアトムを，0-1 計画法を使って探索し，それらを取り除いた実行可能ネットを作成する． ■

プレース数が小さい実行可能ネットを得るために，Step 1 においてどのようなアトムから構成される S を選ぶか，また Step 2 においてどのようなアトムを S に加えるかが重要となる（図 5-10）．

本研究では，Step 1 において，“なるべく多くのトランジションをもつアトムから構成されるS”を選ぶこととした．これは，1 つのプレースが複数のトランジションの入出力プレースを兼ねることで，実行可能ネットのプレース数が減少することを期待したためである．Step 2 において，$\oplus S$の可達なマーキングにおいて，発生可能ではあるが本来発生してはいけない余分なトランジションを調べ，Xに適合するアトムのなかからこれらの発生を抑制できるアトム$n_1, n_2, \cdots$を選択しSに追加することとした．余分なトランジションの発生を抑制するために追加されたアトムを抑制アトムと呼ぶ．考案した解法では，“なるべく多くの余分なトランジションをもつアトム”を抑制アトムとして選ぶこととした．これは，1 つのプレースで複数の余分な半語の発生を抑制することによって，プレース数の減少を図ろうとしたためである．Step 3 において，橋爪らが提案した 0-1 計画法を用いた冗長なアトムの削除方法を適用した[28][48]．

Step 1 において選ばれるSは必ずしも唯一には定まらない．したがって，最適実行可能ネットの探索にあたっては，新たなSを選び，それ以降の手続きを繰り返すことが必要となる．また，本解法は必ずしも最適解が獲得できることを保証するものではないことに注意する．



$CA(X)$: 動作仕様Xに適合するアトムの集合
$n_1, n_2, \cdots$: 抑制アトム
S, S' : 直接の先行関係を被覆するアトムの集合

図 5-10 実行可能ネット探索

例題 5-3：

図 5-11 に示す半言語を動作仕様Xとする問題$\mathrm{P}_{S,min}^{ce}$に本項で述べた解法を適用したときの様子を図 5-12 から図 5-14 に示す．Step 1 より，直接の先行関係を被覆するアトムからなる集合Sを合成する．このうち，n_1はXに含まれる 6 組の直接の先行関係$\{a, b\}$, $\{a, d\}$, $\{a, g\}$, $\{b, c\}$, $\{c, d\}$, $\{c, g\}$に関わるトランジションを含むアトムである．Step 2 において，$FP(\oplus S)$に含まれる余分な半語を取り除く．例えば，3 番目の半語はa, g, fの後ろのaの発火を抑制する必要がある．図 5-13 におけるn_2はこのためのアトムである．この手続きを繰り返すと図 5-13 が定まり，$FP(\oplus S') = X$となる$\oplus S'$が得られる．Step3 では冗長なアトムn_3を取り除く．このような手順の繰り返しで探索を行う． ■

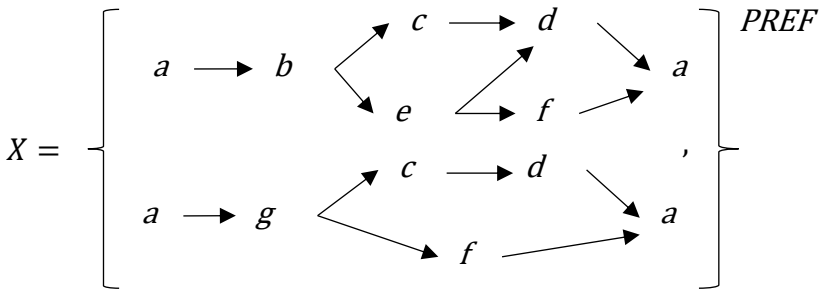


図 5-11 動作仕様

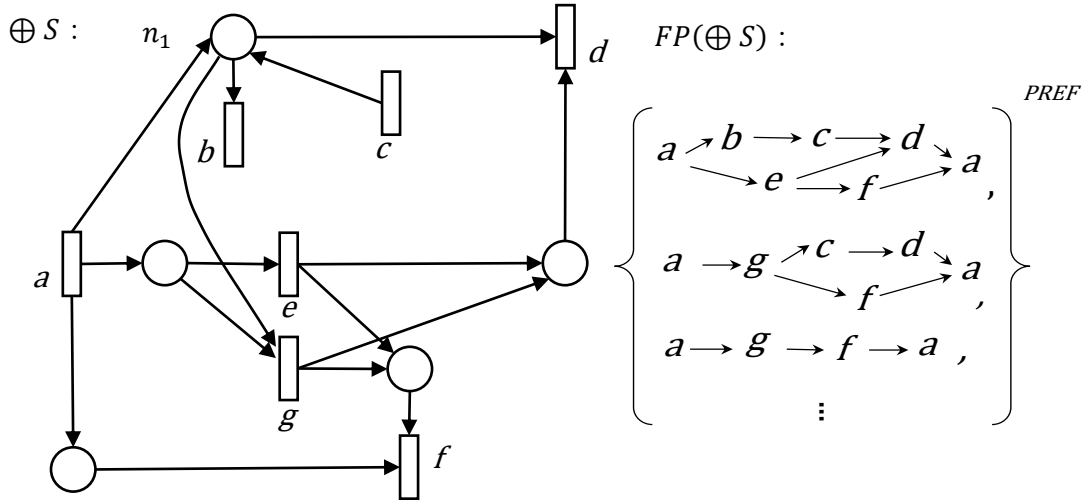


図 5-12　Step 1 で得られるネット⊕*S*

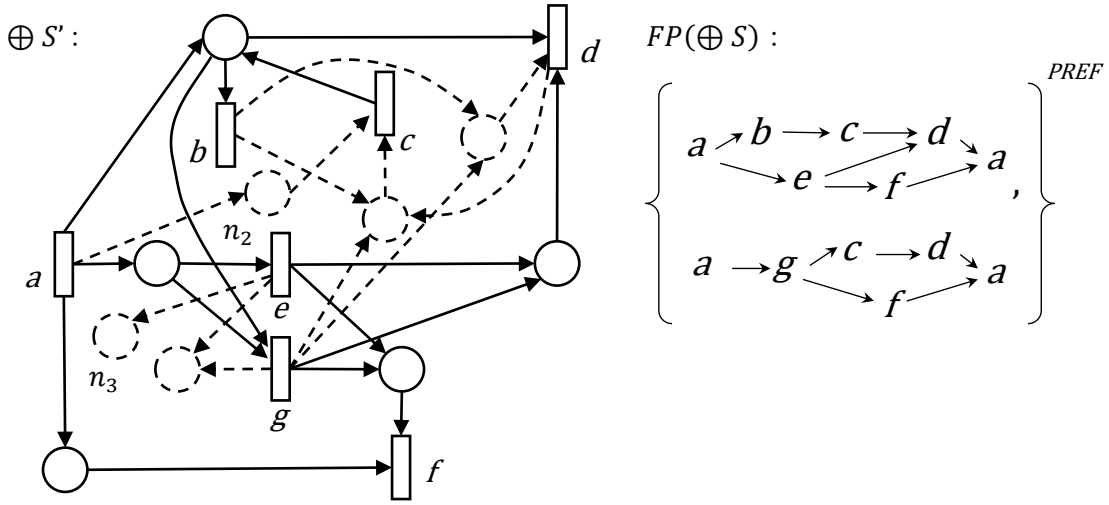


図 5-13　Step 2 で得られるネット⊕*S'*

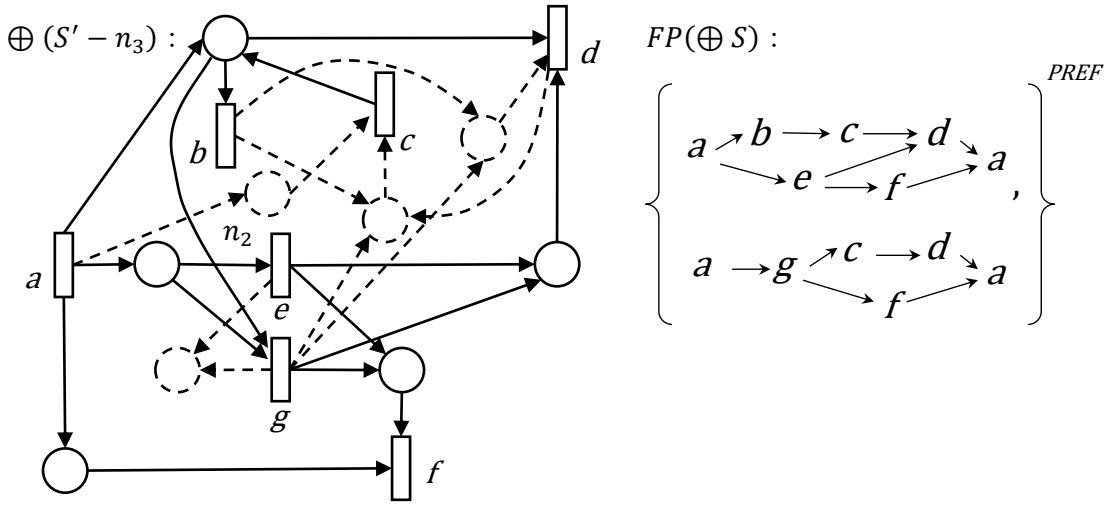


図 5-14　Step 3 で得られるネット⊕ $(S' - \{n_3\})$

5.3.3.　提案解法の効果の確認

提案解法の効果を確認するため，(1)動作仕様の構造，(2)動作仕様の規模の 2 点に着目し，その有効性を確認した．(1)に関し，動作仕様の構造を，並行性と非決定性をもつ動作（C-N），並行性のみもつ動作（C），非決定性のみもつ動作（N），いずれの特徴ももたない動作（0）の 4 種類に分け，それぞれ 100 問ずつランダムに問題を作成し，最小実現となるプレース数$\mathbf{z}^*$と提案手法で得られるプレース数$\mathbf{z}$の差から，解法の精度を確認した(表 5-2)．(2)に関し，上のランダム問題に含まれるトランジションの数$|T_X|$を変更させ，それに対する提案手法における解の探索効率を確認した(表 5-3)．なお，使用した CPU は Intel Core2 Duo@2.53GHz である．また，最小実現となるプレース数を求めるために，原始的な手法である列挙法にて解を求めた．表 5-2，表 5-3 に示す結果から提案解法の効果として，次のことが明らかとなった．

提案解法の効果：

(1) 動作仕様に並行性，非決定性を含むほど，最小実現となる実行可能ネットが得られやすい．

(2) 動作仕様の規模が大きくなるほど，探索時間の短縮が図られる．　■

2.4 で示したように，バッチプロセスの動作の特徴として，並行的動作，非決定的動作等が挙げられる．このことより，(1)の結果は，提案解法が提案解法がバッチプロセスにおいて有効な手段となることを示している．

また，望みの動作仕様の規模が大きさは，そのオペレーションの複雑さを示すものと思われる．そのような状況においては，自ずと異常を誘発するミスオペレーションを行う可能性が高くなる．そのため，複雑なオペレーションに対しては異常回避のための制御器の必要性がより高いと言える．また，バッチプロセスは一般に製品寿命が短いため，そのような制御器を早期に設計し実装することが求められる．しかし，一般に，動作仕様の規模が大きくなるとそれを実現する制御器の設計には時間が掛かり，必要とさえる期限内で制御器が設計できず，手動でのオペレーションに頼らざるをえない状況もある．(2)の結果は，提案解法がこのような異常の回避に係る課題を解決する有効な手段となることを示している．

表 5-2　提案解法の精度

問題の種類			$\Delta z = z - z^*$			
	並行性	非決定性	0	1	2	≥ 3
0	×	×	58	40	2	0
N	×	○	67	33	0	0
C	○	×	80	17	3	0
C-N	○	○	68	29	3	0

表 5-3　探索時間の比較

問題の種類	$\|T_x\|$		
	6	8	10
0	10m 秒未満 （10m 秒未満）	10m 秒未満 （1.78 秒）	0.125 秒 （14 分）
N	10m 秒未満 （10m 秒未満）	0.031 秒 （0.98 秒）	8.35 秒 （16 分）
C	10m 秒未満 （10m 秒未満）	3.08 秒 （5.3 秒）	3.68 秒 （19 分）
C-N	10m 秒未満 （10m 秒未満）	2.06 秒 （20 秒）	12.1 秒 （225 分）

上部　　：　提案解法による探索時間

下部，括弧内　　：　列挙法による探索時間

5.4. 異常の回避のための制御器設計に係る考察

第 4 章で示したように，本研究では異常の回避の考え方を，”異常状態の回避”と”異常動作の回避”の二通りの方法を考え，それぞれ異常の回避問題を定式化し，その解法を検討した．異常状態回避問題 PS では，動作制約を順守することを制御目的とし，新たに制御器プレースを追加することで，制御目的を満たすような制御器を設計した．そして，問題 PS は，本研究で開発した解法から得られる解（制御器）は一意に決まることを示した．一方，異常動作回避問題 PB は，異常動作の回避を制御目的とし，新たに制御器プレースと制御器トランジションを追加することで，制御目的を満たすような制御器を設計した．しかし，開発した解法から得られる解（制御器）は一意に決まらない．そこで問題 PB の解となる制御器の候補のうち，その構成が最も簡易な制御器を求める手法として，5.3 で検討した最小実現の考えが適用できることを明らかにした．これらをまとめて表 5-4 に示す．

表 5-4　異常の回避における解の相違点

問題	追加される制御器ネットN_Cの要素	求められる解N_C
異常状態回避問題 PS	制御器プレース	一意に決まる
異常動作回避問題 PB	制御器プレース 制御器トランジション	一意に決まらない （最小実現問題に帰着）

異常の回避のため制御器は，バッチプロセスの動作を抑制することで，対象とする異常を回避する．よって，本論でも述べたように，その制御器においては，バッチプロセスの正常な動作をできる限り影響を与えないように設計することが望ましい．一方，現実の化学プラントにおいては，異常状態や異常動作だけでなく，それと因果関係の高い正常状態や正常動作においても，これを制御するということで，より安全性を高める保守的な考えもある．このことは，バッチプロセスの多くの動作を抑制し，その柔軟さを損なうことに繋がる．結果として，バッチプロセスの柔軟性を活かした生産性の高い運転の実現が困難となると考えられる．よって，異常の回避のための制御器の設計においては，生産性という要素も考慮して設計することも必要となると考えられる．伊藤らは，ペトリネットモデルを用いバッチプロセスにおける設計と計画の統合に係る手法を開発した[49]．ここに，本研究で示した異常の回避の手法を統合することで，安全性と生産性というトレードオフの関係をもつ二目的の意思決定問題を合理的に解決できることが期待される．これに関しては，今後の課題である．

また本研究では，異常の回避を，状態と動作の異なる視点から，それぞれ制御器の設計問題を定式化し，その解法を提案した．一方，本論でも述べたように，これらの制御問題は本質的に同等である．よって，状態と動作を同時に考慮することで，制御器の設計手法の合理化や扱える問題の種類が増えることが期待される．これに関しては，今後の課題である．

5.5. まとめ

本章においては，異常の回避の考え方のうち，プロセスの動作が異常動作を行うことなく常に正常に動作するような制御器を設計することを目的とする異常動作回避問題 PD に着目し，問題の定式化およびその解法に係る検討を行った．

5.2 では，バッチプロセスの望みの動作（動作仕様）の記述方法として半言語を導入し，バッチプロセスの動作が半言語で記述された動作仕様と一致するような制御器を設計することを目的とした問題 PD を提案した．そして，問題 PD をペトリネットのサブクラスである C/E ネットを用いて定式化し，C/E ネット制御問題P_C^{ce}とした．また，より多くの種類の異常の回避を実現するため，制御器内の動作を表すトランジションを加えた拡張 C/E ネット制御問題P_{CE}^{ce}を定式化し，その解法を示した．一方，問題P_C^{ce}において，その解法から得られる制御器は一意に決まらない．5.3 では，問題P_C^{ce}の解（制御器）の候補のうち，異常の未然の回避という観点から，構成が最も簡単な制御器を求める問題を扱った．そして，その問題を C/E ネット設計問題P_S^{ce}の一つの問題である C/E ネット最小実現問題$\mathrm{P}_{S,min}^{ce}$に帰着させ，その問題の解法を示した．提案した解法は，動作仕様にバッチプロセスの動作の特徴である非決定的動作，並行的動作を含む場合に効果的であることを明らかにした．また，一般に動作仕様のサイズが大きくなるほど，それを実現する制御器の設計には時間が掛かるが，提案解法においては，そのサイズが大きくても短時間で制御器の設計が可能である，効率的なアルゴリズムであることを示した．望みの動作のサイズは，バッチプロセスにおけるオペレーションの複雑さを表しており，そのような動作を手動で行うとミスオペレーションの頻度も高くなる．そのため，異常を含まない望みの動作を実現する制御器を早期に設計することが求められる．このことから，提案解法は異常の回避に係る課題の一つを解決する手段となることを明らかとした．

第 3 章から本章までの内容は，バッチプロセスに含まれる状態および動作は，すべて観測可能な状態であり，また制御可能な動作であった．一方，現実のバッチプロセスにおいては，観測ができない状態や制御できない動作が含まれる．第 6 章においては，観測ができない状態や制御できない動作が含まれるバッチプロセスにおける異常の回避に係る検討を行う．

第6章. 不可観測な状態・不可制御な動作を含むバッチプロセスの異常の回避

前章までは，バッチプロセスのすべてのプロセス変数の状態が観測可能であり，すべての動作が制御可能であるという前提のもとで検討を行った．本章においては，観測できない状態や制御できない動作を含むバッチプロセスにおける異常対応に係る検討を行う．

6.1. はじめに

現実の化学プロセスの特徴の一つに，機械・電気系に比べ，その状態の観測が不可能なプロセス変数や制御不可能な動作が多いことがある．例えば，組成や粘度等のようにオンライン分析計が高価であるため観測が困難であるもの，触媒濃度のように微量な成分の組成であるため観測そのものが極めて困難であるもの，反応速度のように観測するという方法が存在しないものがある．また，反応プロセスにおいては，原料および触媒を投入し，温度および圧力が規定の値へと到達すると，反応は自発的に進行していき，もとの原料へと戻すことはできない．反応はその条件が満たされると自発的に実行されるため，その動作（反応）を外部から抑制することできない．同様に抽出分離や液液分離等においても，その条件が満たされると成分間の移動は自発的に行われ，その動作（分離）を外部から抑制することはできない．

また，バッチプロセスにおいては一つのプラントで多品種を生産するため，常に適したセンサーが配置されているとは限らない．そこには，少量の生産のために反応器へ仕込んだ反応物が液面センサー，温度センサー位置に到達せず，そのプロセス変数の状態が観測できないということもある．流量センサーは，その仕様によって定まる観測下限値があり，非定常運転が主なバッチプロセスにおいては，ある動作の間は流量が観測できないこともある．また，DCS や PLC 等の自動制御システムに異常の回避のための制御器を実装するとした場合，制御システムとプラント内の弁やポンプ等の操作端は計装線で接続された仕様であることが必要である．しかし，ポンプ，冷却器の起動停止スイッチや触媒投入用の弁等，バッチプロセス内にある操作端の幾つかはこのような仕様

になっておらず，プラント内で人が直接操作することを行っている．これらは制御できない動作の一つと考えられる．

このことから，化学プロセス，特にバッチプロセスにおいては，その状態の観測が不可能なプロセス変数や制御が不可能な動作が多く含まれている．そして，異常の回避のための制御器の設計に関しても，これらに対応することは必要であろう．しかし，第1章で示したように，観測ができないプロセス変数と制御が不可能な動作を同時に含むプロセスを対象とした設計手法は確立されていない．よって，本章においては，これらを“不可観測な状態”，“不可制御な動作”とし，バッチプロセスにこれらの要素が含まれる場合における異常の回避に係る検討を行う．また，これまで取り扱った観測できる状態や制御できる動作は“可観測な状態”，“可制御な動作”とする．

本章の構成は次の通りである．6.2 では，不可観測な状態，不可制御な動作のペトリネットモデル上での表現方法を検討する．6.3 では，不可観測な状態を含むバッチプロセスにおける異常の検出に係る検討を行い，可観測な状態のみから構成される新たな動作制約を導出する．6.4，6.5 においては，不可観測な状態，不可制御な動作を含むバッチプロセスの異常状態の回避，異常動作の回避に係る検討を行う．

6.2. 不可観測な状態，不可制御な動作のモデル化

本節においては，不可観測な状態，不可制御な動作のモデル化について検討する．これまでに示したように，本研究においてはバッチプロセスの動作を離散事象としてとらえ，それをペトリネットで記述することとした．よって，不可観測な状態，不可制御な動作においても，これを離散事象システムとしてとらえ，それをペトリネットで記述することを考える．

不可観測な状態，不可制御な動作は離散事象システムの上で，観測が不可能な離散的な状態である”不可観測な状態”，制御が不可能な事象である“不可制御な事象”として，それぞれとらえることができる．これをペトリネットで記述するため，不可観測なプレース，不可制御なトランジションを定義する．

不可観測なプレース：

トークンの数が観測できないプレースを不可観測なプレースと呼ぶ． ■

不可制御なトランジション：

外部からの操作によって発火が抑制できないトランジションを不可制御なトランジションと呼ぶ. ■

一方，これまでに扱ったトークンの数が観測できるプレースは”可観測なプレース”，外部からの操作によって発火が抑制できるトランジションは”可制御なトランジション”とする.

図 6-1 に示すバッチプロセスの動作を表すペトリネットモデルNを用い，不可観測なプレース，不可制御なトランジションについて説明する. Nにおいてプレースp_8, p_9, トランジションt_6はぞれぞれ不可観測なプレース，不可制御なトランジションである. 定義で示すようにp_8に置かれたトークンは観測できない. 一方，t_6は外部からの操作によって発火が抑制できないトランジションであり，図 6-1 中にあるような制御器プレースp_Cからt_6へのアークは接続できない.

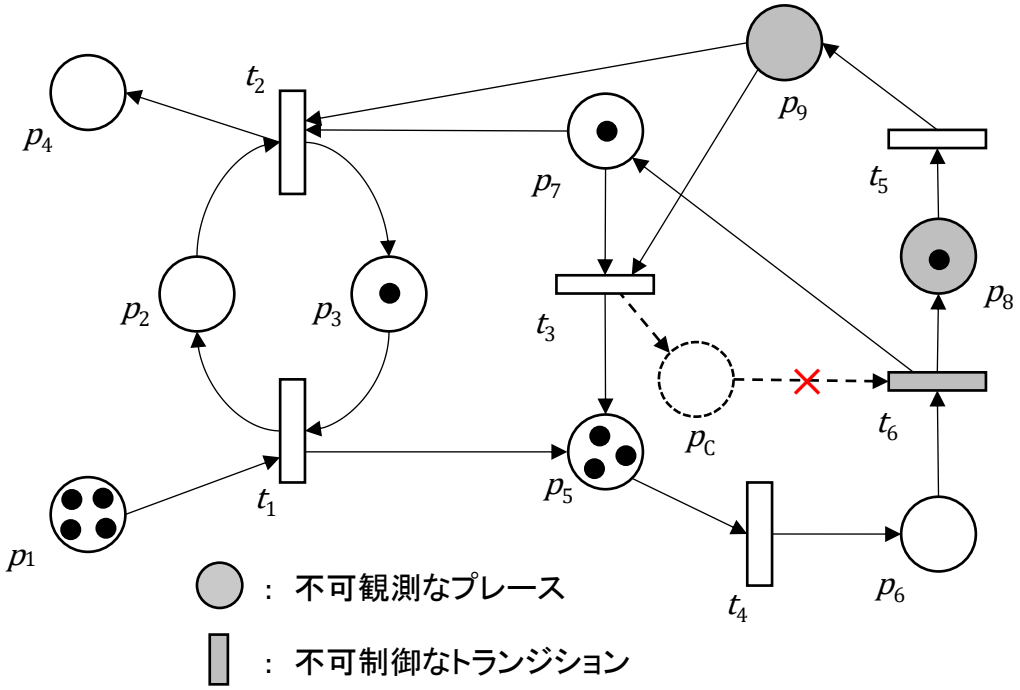


図 6-1　不可観測なプレース，不可制御なトランジションの例

本研究では，プレースが表すプロセス変数の状態を観測することで，動作制約の成否を確認し，異常の回避においては，バッチプロセスの動作を制御器で操作する（ペトリネットモデルの上では，制御プレースにてバッチプロセスの動作を表すトランジションの発火を抑制する）ことで，それを回避した．よって，不可観測な状態や不可制御な

動作が含まれるバッチプロセスにおいては，これまで検討した手法では，動作制約の成否の判定や，異常の回避のための制御器の設計が困難となることが考えられる．6.3 からは，これに対応するための方法について検討する．

6.3. 異常の検出

6.3.1. 動作制約の生成

本節ではバッチプロセスにおいて不可観測な状態が含まれる場合，そのプロセスに発生する異常の検出について検討を行う．

本研究では第 2 章で定義したように，異常を

異常 F1：1 つのプロセス変数が単独で示す異常

異常 F2：複数のプロセス変数が組になって初めて示す異常

の 2 つに分類した．異常 F1 に関しては，そのプロセス変数の状態が観測できない場合，そのプロセス変数と相関の高い別の観測可能なプロセス変数から，状態を推定することとなる．例えば，組成は圧力が一定の条件下においては，温度と高い相関性を示し，反応速度は温度変化や冷媒・熱媒の使用量の組合せと高い相関性を示すことが多い．このように可観測なプロセス変数と不可観測なプロセス変数の相関性からある検量線を求め，それを用いることで不可観測なプロセス変数の状態を推定し，そこから異常を検出する方法がある．

一方，異常 F2 に関しては，それを検出するため，“プロセスが正常な動作をする限り複数のプロセス変数が満たすべき条件”である動作制約に着目し，動作制約の成否によって，これを検出することとした．そして第 3 章では，動作制約はバッチプロセスの正常な動作を表すペトリネットモデルからプレースインバリアント解析手法を用い導出できることを示した．この動作制約に不可観測な状態が含まれる場合，プレースインバリアントの性質を活かし，異常の検出を行うことが合理的な方法であろう．

バッチプロセスの正常な動作を表すペトリネットモデルをNとし，m,nをNのプレース数とトランジション数，$1\times m$ベクトル$\boldsymbol{y}$をNのプレースインバリアントとすると，プレースインバリアントに係る(6.1)式，(6.2)式が得られる（(3.1)式，(3.3)式と同じ）．

$$\boldsymbol{yD} = 0 \tag{6.1}$$

$$\boldsymbol{yM} = \boldsymbol{yM_0} = \mathrm{c} \tag{6.2}$$

ここで，$\boldsymbol{M}$はNのマーキングで，$\boldsymbol{M_0}$はNの初期マーキングであり，それぞれ$m \times 1$ベクトルである．$\boldsymbol{D}$はNの$m \times n$接続行列であり，$\boldsymbol{M_0}$を既知とすると，$\boldsymbol{yM_0} = \mathrm{c}$は定数となる．このとき，$\boldsymbol{M^O}$を$\boldsymbol{M}$の可観測なプレースに関わる部分，$\boldsymbol{M^{UO}}$を$\boldsymbol{M}$の不可観測なプレースに関わる部分，$\boldsymbol{y^O}$を$\boldsymbol{y}$の可観測なプレースに関わる部分，$\boldsymbol{y^{UO}}$を$\boldsymbol{y}$の不可観測なプレースに関わる部分とすると，(6.2)式は次式のように変換することができる．

$$\mathbf{y}\boldsymbol{M} = [\boldsymbol{y^O} \quad \boldsymbol{y^{UO}}] \begin{bmatrix} \boldsymbol{M^O} \\ \boldsymbol{M^{UO}} \end{bmatrix} = \mathrm{c} \tag{6.3}$$

台$\|\boldsymbol{y^{UO}}\|$の要素は不可観測なプレースを表し，動作制約に$\|\boldsymbol{y^{UO}}\|$の要素が多くなれば，異常の検出の精度が悪化する．ここで，プレースインバリアント同士を線形結合したものも，プレースインバリアントとなる性質を用い，プレースインバリアントに含まれる$\|\boldsymbol{y^{UO}}\|$の要素をできるだけ少なくすることで，異常の検出の精度を向上させることを考える．簡単な例を例題 6-1 に示す．

例題 6-1：

バッチプロセスの動作をモデル化したところ，図 6-2 で示すペトリネットモデルNが得られた．Nの状態遷移方程式およびプレースインバリアントは，以下の通りである．

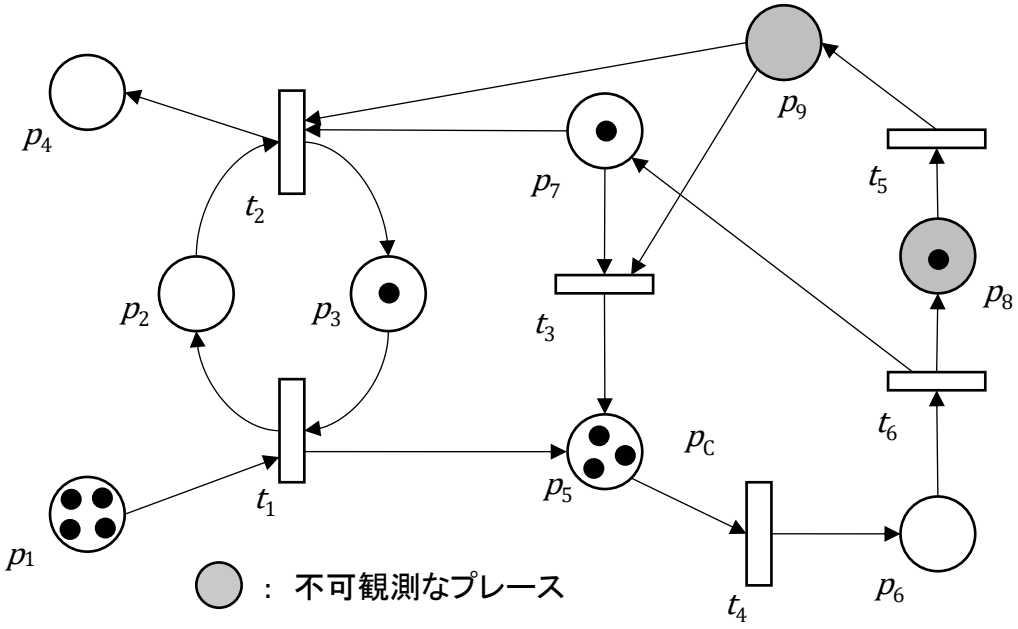


図 6-2　不可観測なプレースを含むペトリネット

$$\boldsymbol{M} = \begin{bmatrix} 4 \\ 0 \\ 1 \\ 0 \\ 3 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & -1 & -1 & 0 & 1 & 0 \end{bmatrix} x$$

$\boldsymbol{y_1}$: [0 1 1 0 0 0 0 0 0]

$\boldsymbol{y_2}$: [1 1 0 1 0 0 0 0 0]

$\boldsymbol{y_3}$: [1 0 −1 1 0 0 0 0 0]

$\boldsymbol{y_4}$: [0 0 1 0 1 1 1 0 0]

$\boldsymbol{y_5}$: [0 0 1 0 1 1 0 1 1]

$\boldsymbol{y_6}$: [0 −1 0 0 1 1 1 0 0]

$\boldsymbol{y_7}$: [0 −1 0 0 1 1 0 1 1]

$\boldsymbol{y_8}$: [1 0 0 1 1 1 1 0 0]

$\boldsymbol{y_9}$: [1 0 0 1 1 1 0 1 1]

$\boldsymbol{y_{10}}$: [0 0 0 0 0 0 1 −1 −1]

よって，次に示す 10 個の動作制約が求められた．

C_1 : $M(2) + M(3) = 1$

C_2 : $M(1) + M(2) + M(4) = 4$

C_3 : $M(1) - M(3) + M(4) = 3$

C_4 : $M(3) + M(5) + M(6) + M(7) = 5$

C_5 : $M(3) + M(5) + M(6) + M(8) + M(9) = 5$

C_6 : $-M(2) + M(5) + M(6) + M(7) = 4$

C_7 : $-M(2) + M(5) + M(6) + M(8) + M(9) = 4$

C_8 : $M(1) + M(4) + M(5) + M(6) + M(7) = 8$

C_9 : $M(1) + M(4) + M(5) + M(6) + M(8) + M(9) = 8$

C_{10} : $M(7) - M(8) - M(9) = 0$

このとき，プレースp_8，p_9を不可観測なプレースとすると，p_8，p_9が含まれる動作制約C_5，C_7，C_9はその成否を正確に判定することができない．ここで，プレースインバリアントの線形結合によって，不可観測なプレースを削除し，異常の検出の精度を高めるこ

とを考える．$\boldsymbol{y}_5 + \boldsymbol{y}_7 + 2\boldsymbol{y}_{10}$とプレースインバリアントを結合することで，新たなプレースインバリアント$\boldsymbol{y}_{11}$を作ると，

$$\boldsymbol{y}_{11} = \boldsymbol{y}_5 + \boldsymbol{y}_7 + 2\boldsymbol{y}_{10} = [\ 0 \quad -1 \quad 1 \quad 0 \quad 2 \quad 2 \quad 2 \quad 0 \quad 0]$$

が得られる．すなわち，新たな動作制約として，不可観測な状態を含まない動作制約C_{11}を得ることができる．

$$C_{11}:\ -M(2) + M(3) + 2M(5) + 2M(6) + 2M(7) = 9 \qquad \blacksquare$$

例題 6-1 は，粘度，モル組成，反応速度等の不可観測な状態を含む動作制約を，温度, 圧力, 流量等の可観測な状態のみが含まれる動作制約に変換する手法を示しており，異常 F1 で述べた検量線から推定する手法と同等と言える．しかし，不可観測な状態が示す異常は検出することができない．6.3.2 では，不可観測な状態を含む動作制約において，異常の検出のための新たな動作制約を導出する方法について検討する．

6.3.2. 不可観測な状態を含む動作制約

動作制約に不可観測なプロセス変数を含む場合，異常の検出のため，新たな動作制約の導出を考える．(6.3)式において，もし，$\boldsymbol{y}^{UO}\boldsymbol{M}^{UO}$の上下限値$(u_1,\ u_2)$が推定できるとき，次の(6.4)式を得ることができる．

$$u_1 \le \boldsymbol{y}^{UO}\boldsymbol{M}^{UO} \le u_2 \tag{6.4}$$

この式から，可観測な状態のみで構成される新たな動作制約（不等式）を導くことができる．

$$b_1 \le \boldsymbol{y}^{O}\boldsymbol{M}^{O} \le b_2 \quad \text{ただし，} b_1 = \boldsymbol{y}\boldsymbol{M}_0 - u_2\,, b_2 = \boldsymbol{y}\boldsymbol{M}_0 - u_1 \tag{6.5}$$

(6.5)式は，(6.6)式のようにまとめることができる．

$$\begin{bmatrix} -\boldsymbol{y}^{O} \\ \boldsymbol{y}^{O} \end{bmatrix} \boldsymbol{M}^{O} \le \begin{bmatrix} -b_1 \\ b_2 \end{bmatrix} \tag{6.6}$$

(6.6)式は，不可観測なプロセス変数が(6.4)式の状態となるように，可観測なプロセス変数の状態が満たすべき条件を示している．ここで，(6.6)式の不等式で表される動作制約を不等号動作制約（IOC：Inequality Operational Constraint），等式で表される従来型の動作制約を等号動作制約（EOS：Equality Operational Constraint）とする．第 3 章で示したように, 異常状態は動作制約を満たさないプロセスの状態であった. よって,

バッチプロセスの状態が IOC を満たさないとき，バッチプロセスは異常状態であると言える．

例題 6-2：

例題 6-1 と同じペトリネットモデルNを考える．EOC C_5はプレースインバリアント$\boldsymbol{y}_5$から導かれ，3 つの可観測なプレース$p_3, p_5, p_6 \in \|\boldsymbol{y}_5^O\|$，2 つの不可観測なプレース$p_8, p_9 \in \|\boldsymbol{y}_5^{UO}\|$から構成される．

C_5 ： $M(3) + M(5) + M(6) + M(8) + M(9) = 5$

ここで，次の不等式を考える．つまり，p_8, p_9の合計値の値にある推定値を与える．

$1 \leq M(8) + M(9) \leq 3$

すると，p_3, p_5, p_6に置かれるトークンの合計に関し，次の IOC C_{12}が得られる．

$C_{12}: 2 \leq M(3) + M(5) + M(6) \leq 4$ ■

6.4. 異常状態回避問題

6.4.1. 不可観測な状態を含む異常状態回避問題

本項では，不可観測な状態を含むバッチプロセスにおける異常状態回避問題を考える．6.3 では，バッチプロセスの正常な動作を表すペトリネットモデルから得られる等号動作制約 EOC（第 2 章で定義した動作制約と同じ）と，不可観測な状態の取りうる値を推定することで得られる不等号動作制約 IOC の 2 つを導いた．EOC は，プレースインバリアント解析によって導出されるので，その成立は保証される．しかし，IOC は推定値から得られる制約であるため，その成立は保証されない．例えば，例題 6-2 において，発火ベクトル$\boldsymbol{x} = [0,0,0,3,3,3,]^{\mathsf{T}}$とすると，その状態は$\boldsymbol{M} = [4,2,1,0,0,0,4,1,3]^{\mathsf{T}}$となる．よって，$M(3) + M(5) + M(6) = 1 < 2$となり，IOC C_{12}は成立しないことが分かる．

第 4 章で示したように，本研究では，動作制約を順守することを目的とする“異常状態回避問題”を定義し，異常の回避のための制御器の設計に関する検討を行った．本項で新たに定義した IOC に関しても，これを遵守するための制御器が必要となる．また本章にでは不可観測な状態，不可制御な動作を含むバッチプロセスを対象としており，異常の回避のための制御器の設計においては，不可観測な状態と同様に不可制御な動作に関しても考慮しなければならない．Moody らは，不可制御な事象を含む離散事象システムを対象に，ペトリネットのフィードバックコントローラを構築する手法を提案し

た[45]．本研究で取り扱う問題と Moody らの問題は似ているが，Moody らはすべてのプレースが可観測なプレースであるとの仮定のもと，これらの検討を行っている．よって，これら二つの大きな違いは，不可観測なプレースが存在するか否かであり，本研究の方が，より現実のプロセスに近い条件と言える．

$\boldsymbol{y}^{UO}\boldsymbol{M}^{UO}$の上下限値の不等式(6.7)（(6.4)式と同じ）に新たにスラック変数μ_1およびμ_2を導入することで，(6.7)式を(6.8)式の等式に変換することができる．

$$u_1 \le \boldsymbol{y}^{UO}\boldsymbol{M}^{UO} \le u_2 \tag{6.7}$$

$$\begin{aligned} \boldsymbol{y}^{O}\boldsymbol{M}^{O} - \mu_1 &= b_1 \\ \boldsymbol{y}^{O}\boldsymbol{M}^{O} + \mu_2 &= b_2 \end{aligned} \tag{6.8}$$

この式を行列形式で表現すると，次の(6.9)式が得られる．

$$\begin{bmatrix} \boldsymbol{y^O} & \boldsymbol{0} & -1 & 0 \\ \boldsymbol{y^O} & \boldsymbol{0} & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{M^O} \\ \boldsymbol{M^{UO}} \\ \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{6.9}$$

すなわち，$[\boldsymbol{y^O} \quad \boldsymbol{0} \quad -1 \quad 0]$および$[\boldsymbol{y^O} \quad \boldsymbol{0} \quad 0 \quad 1]$は，制御器ネット$N_C$を追加したペトリネットモデルの新たなプレースインバリアントとなる．よって，(6.10)式が導かれる．

$$\begin{bmatrix} \boldsymbol{y^O} & \boldsymbol{0} & -1 & 0 \\ \boldsymbol{y^O} & \boldsymbol{0} & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{D} \\ \boldsymbol{D_C} \end{bmatrix} = \boldsymbol{0} \tag{6.10}$$

ここで，$\boldsymbol{D_C}$はN_Cの$2 \times n$（ただしnはNのトランジションの個数）接続行列である．このことから，(6.11)式，(6.12)式が得られる．

$$\boldsymbol{D_C} = \begin{bmatrix} \boldsymbol{y^O} & \boldsymbol{0} \\ -\boldsymbol{y^O} & \boldsymbol{0} \end{bmatrix} \boldsymbol{D} \tag{6.11}$$

$$\begin{aligned} \mu_{10} &= -b_1 + \boldsymbol{y}^{O}\boldsymbol{M}_0^{O} \\ \mu_{20} &= \quad b_2 - \boldsymbol{y}^{O}\boldsymbol{M}_0^{O} \end{aligned} \tag{6.12}$$

ここで，μ_{10}およびμ_{20}は 2 つの制御器プレースp_{C1}, p_{C2}のそれぞれの初期トークン数である．この制御器は，バッチプロセスの状態が IOC（(6.6)式）を不成立にさせる，つまり異常状態になることを回避する役割を果たす．

例題 6-3：

例題 6-2 で示した IOC $C_{12} : 2 \le M(3) + M(5) + M(6) \le 4$ に対し，スラック変数μ_1およびμ_2を導入することで，次の等式を導く．

$$M(3) + M(5) + M(6) - \mu_1 = 2$$

$$M(3) + M(5) + M(6) + \mu_2 = 4$$

よって，制御器ネットN_Cの接続行列D_Cは次のように導かれる．

$$D_C = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & -1 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & -1 & -1 & 0 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & -1 & -1 & 0 & 0 & 1 \end{bmatrix}, \ \mu_{10} = 2, \ \mu_{20} = 0$$

図 6-3 の破線は制御器ネットN_Cを示している． ■

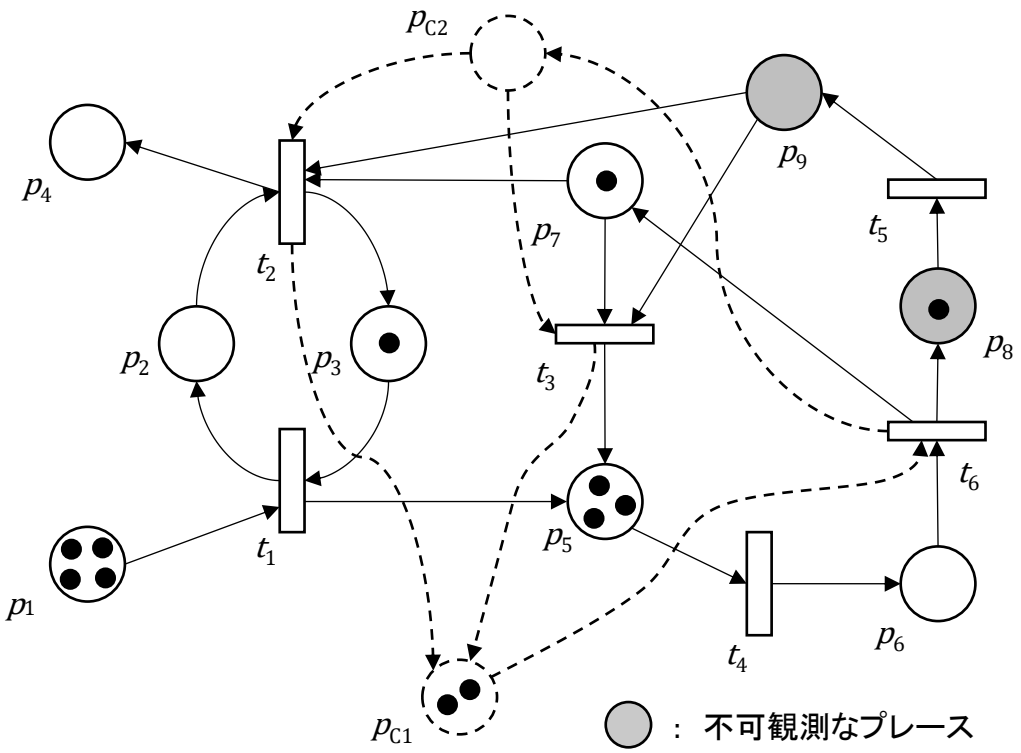


図 6-3　不可観測なプレースを含む制御器の設計

6.4.2.　不可制御な動作を含む異常状態回避問題

本項では，不可観測な状態に加え不可制御な動作を含むバッチプロセスについて考える．6.2 で示したように，本研究では不可制御な動作をペトリネットの上で”不可制御なトランジション”と定義した．不可制御な動作は制御器によってその動作を操作する

ことができない．ペトリネットの上では，制御器プレースから不可制御なトランジションへのアークが引けない，つまり(6.13)式がすべての制御器プレースp_{Ci}と不可制御なトランジションt_jの間で成立する必要がある．

$$D_C(i,j) \geq 0 \tag{6.13}$$

ここで，制御器ネットN_Cの接続行列の要素$D_C(i,j)$はトランジションt_jが一回発火することによって生じるプレースp_iのトークン数の変化である．

ここで，$\boldsymbol{D}^{UC}$を$\boldsymbol{D}$の不可制御なトランジションの列から構成される接続行列，$\boldsymbol{D}_C^{UC}$を$\boldsymbol{D}_C$の不可制御なトランジションに対応する部分とすると，(6.11)式は(6.14)式に変換できる．

$$\boldsymbol{D}_C^{UC} = \begin{bmatrix} \boldsymbol{y^O} & \boldsymbol{0} \\ \boldsymbol{-y^O} & \boldsymbol{0} \end{bmatrix} \boldsymbol{D}^{UC} \tag{6.14}$$

そして(6.14)式のすべての要素が 0 以上であれば，不可制御なトランジションの発火を抑制していないこととなる（(6.15)式）．

$$\boldsymbol{D}_C^{UC} = \begin{bmatrix} \boldsymbol{y^O} & \boldsymbol{0} \\ \boldsymbol{-y^O} & \boldsymbol{0} \end{bmatrix} \boldsymbol{D}^{UC} \geq 0 \tag{6.15}$$

しかし，$\boldsymbol{D}_C^{UC}$のある成分において$D_C^{UC}(i,j) \ngeq 0$となる場合，この制御器ネットは不可制御なトランジションの発火を抑制していることとなり，実現不可能となる．もし，(6.15)式が成立しないのであれば，(6.6)式の代わりとなる(6.16)式を考える．

$$\begin{bmatrix} \boldsymbol{h_1} \\ \boldsymbol{h_2} \end{bmatrix} \boldsymbol{M^O} \leq \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \tag{6.16}$$

(6.16)式は，次に定義する制御器合成条件を満たす必要がある．

制御器合成条件：

SC1：(6.16)式を満たす$\boldsymbol{M^O}$は常に(6.6)式を満たすこと．

SC2：$\begin{bmatrix} \boldsymbol{-h_1} & \boldsymbol{0} \\ \boldsymbol{-h_2} & \boldsymbol{0} \end{bmatrix} D^{UC} \geq 0$ ■

(6.16)式から，接続行列D_Cおよび制御器プレースの初期トークンに関し，次の(6.17)式，(6.18)式が導かれる．

$$\boldsymbol{D_C} = \begin{bmatrix} \boldsymbol{-h_1} & \boldsymbol{0} \\ \boldsymbol{-h_2} & \boldsymbol{0} \end{bmatrix} \boldsymbol{D} \tag{6.17}$$

$$\begin{aligned} \mu_{10} &= d_1 - \boldsymbol{h_1 M_0^O} \\ \mu_{20} &= d_2 - \boldsymbol{h_2 M_0^O} \end{aligned} \tag{6.18}$$

この制御器によって，バッチプロセスが不可観測な状態に加え，不可制御な動作が存在する場合においても，(6.6)式で示す IOC を満たさない状態（異常状態）となることを回避することが可能となる．

本章で開発した不可観測な状態，不可制御な動作を含むバッチプロセスにおける異常状態回避のための制御器設計手順を次に示す．

異常状態回避のための制御器設計手順：

Step 1:

バッチプロセスの正常な動作を表すペトリネットモデルから，プレースインバリアント解析により等号動作制約 EOS を導出する．もし，EOS が可観測な状態のみから構成される，すなわち$\|\boldsymbol{y}^{UO}\| = \emptyset$であれば，終了する．そうでなければ，Step 2 へ進む．

Step 2:

$\boldsymbol{y}^{\mathrm{UO}}\boldsymbol{M}^{UO}$の上下限値$u_1, u_2$を推定し，(6.6)式で表される IOC を導出する．

Step 3:

もし，バッチプロセスが可制御な動作のみから構成されている，すなわち$\boldsymbol{D}^{\mathrm{UC}} = \boldsymbol{0}$ならば，制御器ネット$N_{\boldsymbol{C}}$の接続行列$\boldsymbol{D}_{\boldsymbol{C}}$およびその初期マーキングは，(6.11)式，(6.12)式から得られる．$\boldsymbol{D}^{\mathrm{UC}} \neq \boldsymbol{0}$であれば，Step 4 へ進む．

Step 4:

もし，(6.15)式を満たすならば，$\boldsymbol{D}_{\boldsymbol{C}}$およびその初期マーキングは，(6.11)式，(6.12)式から得られる．そうでなければ，制御器合成条件を満たす$\boldsymbol{h}_1, \boldsymbol{h}_2, d_1, d_2$（(6.16)式の変数）を探索する．そして，$\boldsymbol{D}_{\boldsymbol{C}}$およびその初期マーキングは(6.17)式，(6.18)式から得られる． ■

本開発手法における制御器設計は，制御器合成条件を満たす$\boldsymbol{h}_1, \boldsymbol{h}_2, d_1, d_2$を探索する必要がある．そして，得られる制御器においては，バッチプロセスの動作を過度に制限しないことが望ましい．しかし，そのような制御器合成条件を厳密に求めることは本質的に組合せ問題であり解決が困難な問題である．よって，バッチプロセスの動作をできる限り制限しない御器を設計するには，次に示すヒューリスティックな手法で制御器合成条件 SC1，SC2 を求めることが効果的と考えられる．

SC1：

SC1 を満たす条件のうち，$\Sigma - \Sigma'$の要素数をできるだけ小さくなるような条件を選択する．ここで，Σは(6.6)式を満たすマーキングの集合，Σ'は(6.16)式を満たすマーキングの集合であり，$\Sigma' \subseteq \Sigma$である．ここで，ΣおよびΣ'は単なるマーキングであり，その導出が困難な可達なマーキングではない．よって，SC1 を満たす条件のうち，Σ'の要素数が多くなるよう，探索することが合理的な方法と思われる． ■

SC2：

不可制御な動作は，直接操作し，その動作を抑制する（止める）ことはできない．そのため，他の可制御な動作を代替動作として直接操作することで，不可制御な動作を間接的に抑制する．一つの考えとしては，不可制御な動作の直前に行われる可制御な動作を代替動作として選択するというものがある．これは直感的に，直前の動作を制御することで，バッチプロセスの動作への影響を最小限にできると考えられるためである．■

本研究では，このようなヒューリスティックな手法により，制御器合成条件を満たす$\boldsymbol{h}_1, \boldsymbol{h}_2, d_1, d_2$を探索する．厳密解の探索手法に係る検討は，今後の課題である．

一方，本研究では異常の回避を目的とした制御器を取り扱っているが，現実のバッチプロセスには，“あらかじめ定まった手順通りにプラントを動かし，目標の状態へと遷移させる”ことを目的とした制御器（レシピ制御）が別に存在している．よって，現実のプロセスにおいては，異常を回避しつつ，プロセスの状態を目標の状態へ遷移させることが求められる．このことは，異常の回避の制御器を組み込むことで制限されるバッチプロセスの動作が，目標となる「望みの状態」やあらかじめ定まった「望みの動作」を包含していれば良いことを示している．これは，厳密解を求めることと比べ，計算負荷が小さくなることは明らかであろう．よって他の制御器の目的と組み合わせた異常の回避の制御器設計手法は，厳密解を求める方法よりも現実的な手法と考えられる．

例題 6-4：

例題 6-3 で設計した制御器において，t_6が不可制御なトランジションである場合，制御器プレースp_{C1}からt_6へのアークは引くことはできない．IOC の代替案として，

$M(3) + M(5) \geq 2$

$M(3) + M(5) + M(6) \leq 4$

を考える．この代替案(6.16)式における$\boldsymbol{h}_1, \boldsymbol{h}_2, d_1, d_2$はそれぞれ次の通りである．

$\boldsymbol{h}_1 = [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0]$ ， $d_1 = 2$

$\boldsymbol{h}_2 = [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0]$ ， $d_2 = 4$

制御器合成条件 SC1 を満たすことは明らかである．SC2 は，

$$\begin{bmatrix} -\boldsymbol{h}_1 & \boldsymbol{0} \\ -\boldsymbol{h}_2 & \boldsymbol{0} \end{bmatrix} \boldsymbol{D}^{UC} = \begin{bmatrix} 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & -1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \geq 0$$

となり，これも満たす．よって，この代替案は制御器合成条件を満たす．接続行列$\boldsymbol{D_C}$，制御器プレースの初期トークンμ_{10}, μ_{20}は次のとおりである（図 6-4）．

$$\boldsymbol{D_C} = \begin{bmatrix} 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & -1 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{D}$$

$$= \begin{bmatrix} 0 & -1 & -1 & 1 & 0 & 0 \\ 0 & -1 & -1 & 0 & 0 & 1 \end{bmatrix}$$

$\mu_{10} = d_1 - [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0]\boldsymbol{M}_0^O = 2$

$\mu_{20} = d_2 - [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0]\boldsymbol{M}_0^O = 0$ ■

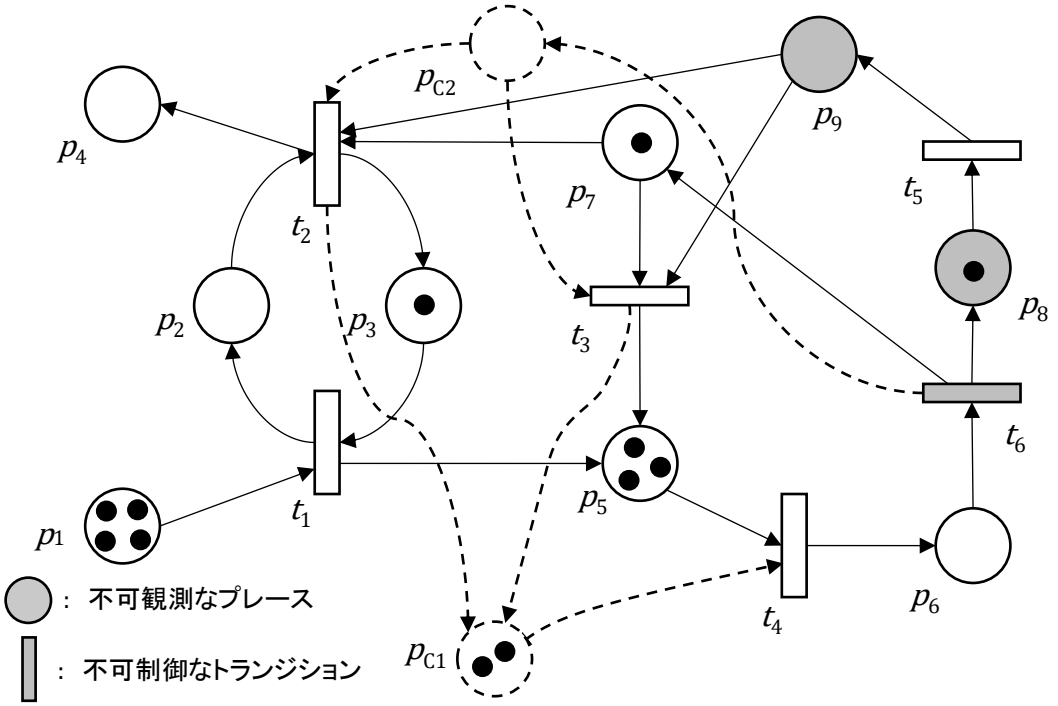


図 6-4　不可制御なトランジションを含む制御器の設計

6.4.3. 適用例

プロセス説明

本節で開発した手法を，図 6-5 のバッチ反応プロセスに適用した事例を示す．このプロセスの動作は次の通りである．

バッチ反応プロセスの動作：

1. 入口弁を開き，反応器に原料の供給を開始する．原料がある液面に到達したら，入口弁を閉める．
2. 加熱および加圧を同時に開始し，それぞれある目標温度，ある目標圧力へと到達させる．
3. 目標値到達後，触媒を添加する．しばらくして，反応が始まる．ある一定時間，その温度と圧力を維持する．
4. 冷却および減圧を開始し，反応器の圧力が大気圧に到達したら，出口弁を開け，製品を取り出す． ■

ペトリネットモデル

図 6-6 は本バッチ反応プロセスのペトリネットモデルとその接続行列を表す．表 6-1 にこのペトリネットモデルのプレースとトランジションの意味を示す．また，このペトリネットモデルに対するプレースインバリアント解析から，次の 7 つの独立な EOS が得られる．

C_1 ： $M(1)+M(2)=1$

C_2 ： $M(6)+M(7)=1$

C_3 ： $M(3)+M(4)+M(5)=1$

C_4 ： $M(1)+M(4)+M(6)=2$

C_5 ： $M(2)+M(3)+M(7)+M(8)+M(9)+M(10)+M(11)+M(12)+M(19)=1$

C_6 ： $M(13)+M(14)+M(15)=1$

C_7 ： $M(16)+M(17)+M(18)=1$

制御器の設計

動作制約C_5は不可観測なプレースp_{19}が含まれている．このプレースは多くとも 1 つのトークンが置かれるため，次の新たな不等号動作制約 IOC が導出される．

$$0 \le M(2)+M(3)+M(7)+M(8)+M(9)+M(10)+M(11)+M(12) \le 1$$

これは，次のC_8，C_9に変換できる．

C_8 ： $M(2)+M(3)+M(7)+M(8)+M(9)+M(10)+M(11)+M(12) \geq 0$

C_9 ： $M(2)+M(3)+M(7)+M(8)+M(9)+M(10)+M(11)+M(12) \leq 1$

(6.11)式，(6.12)式から図 6-7 の破線で示す制御器ネットが得られる．しかし，t_{13}は”反応プロセスが開始される”を表し，反応物に触媒を添加した後（p_{10}）は，それを停止することができない不可制御なトランジションである．よって，この制御器ネットは実現できない．C_8，C_9の代替案として，

C_{10} ： $M(2)+M(3)+M(7)+M(8)+M(9)+M(11)+M(12) \geq 0$

C_9 ： $M(2)+M(3)+M(7)+M(8)+M(9)+M(10)+M(11)+M(12) \leq 1$

を考える．C_9，C_{10}は制御器合成条件を満たす．よって，(6.17)式，(6.18)式より図 6-8 の破線で示す制御器ネットが得られる．この制御器は触媒を添加するタイミングを制御することで，プロセスの状態を許容の範囲に維持する． ■

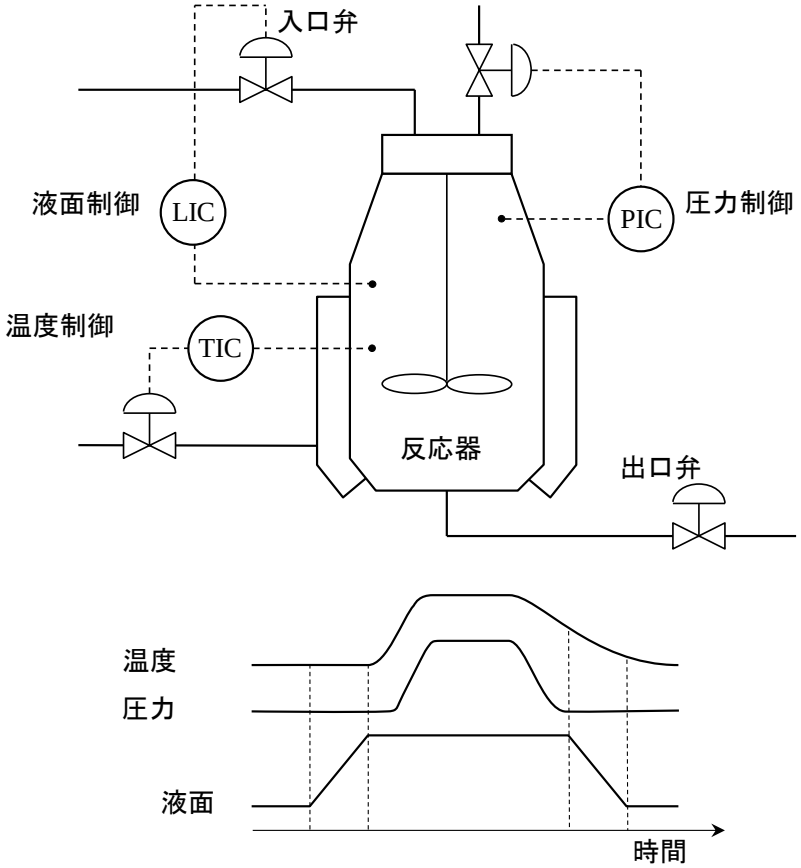
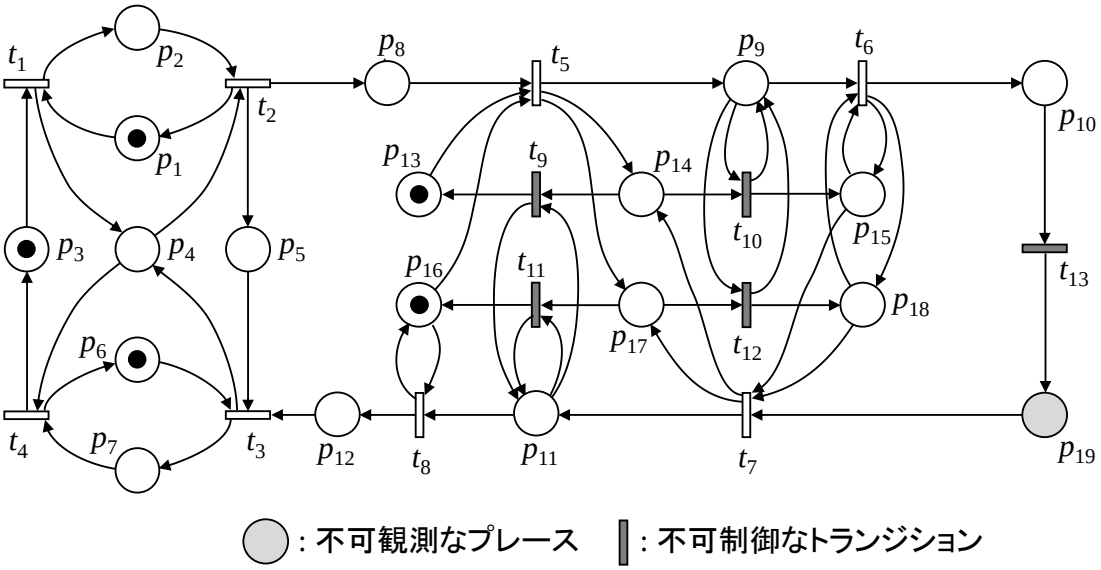


図 6-5　バッチ反応プロセス



$$
\boldsymbol{D} = \begin{bmatrix}
-1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -1 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & -1 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
$$

図 6-6　バッチ反応プロセスのペトリネットモデル

表 6-1　プレースの説明

プレース／トランジション	意味
p_1	入口弁閉
p_2	入口弁開
p_3	液面なし（L_0）
p_4	液面がL_0からL_Sの間
p_5	ある液面（L_S）
p_6	出口弁閉
p_7	出口弁開
p_8	反応操作開始準備完了
p_9	加熱および加圧操作実行中
p_{10}	触媒添加完了
p_{11}	冷却および減圧操作実行中
p_{12}	反応器内の圧力が大気圧まで減圧している
p_{13}	低温度（T_L）
p_{14}	温度がT_LからT_Hの間
p_{15}	高温度（T_H）
p_{16}	低圧（P_L）
p_{17}	圧力がP_LからP_Hの間
p_{18}	高圧（P_H）
p_{19}	反応プロセスが進行中
t_1	入口弁を開く
t_2	入口弁を閉じる
t_3	出口弁を開く
t_4	出口弁を閉じる
t_5	加熱および加圧を開始する
t_6	触媒を添加する
t_7	冷却および減圧を開始する
t_8	反応プロセス完了
t_9	低温度に到達する
t_{10}	高温度に到達する
t_{11}	低圧に到達する
t_{12}	高圧に到達する
t_{13}	反応プロセスが開始される

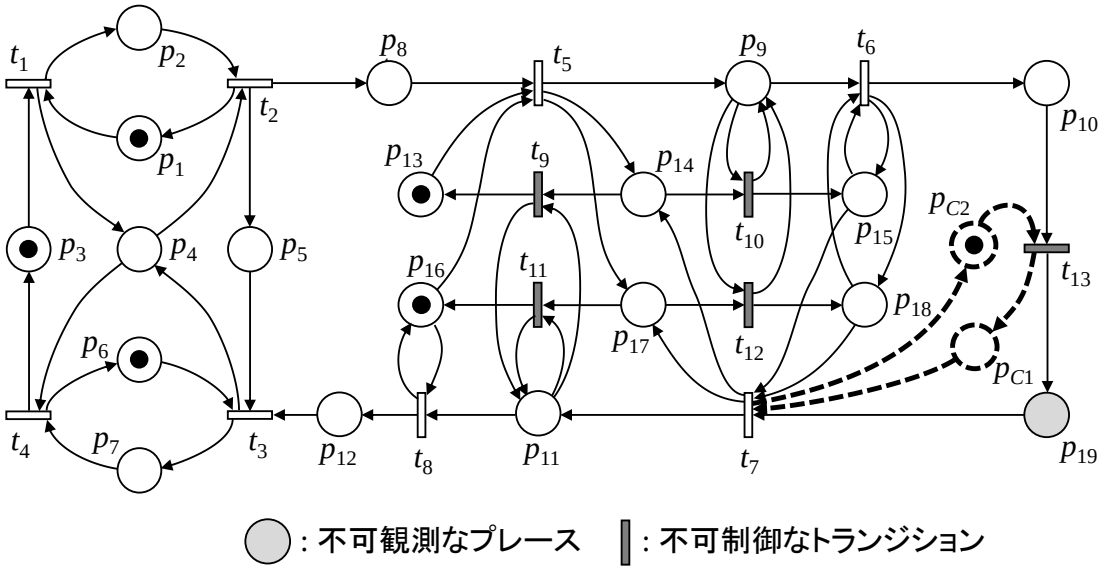


図 6-7　実現不可能な制御器

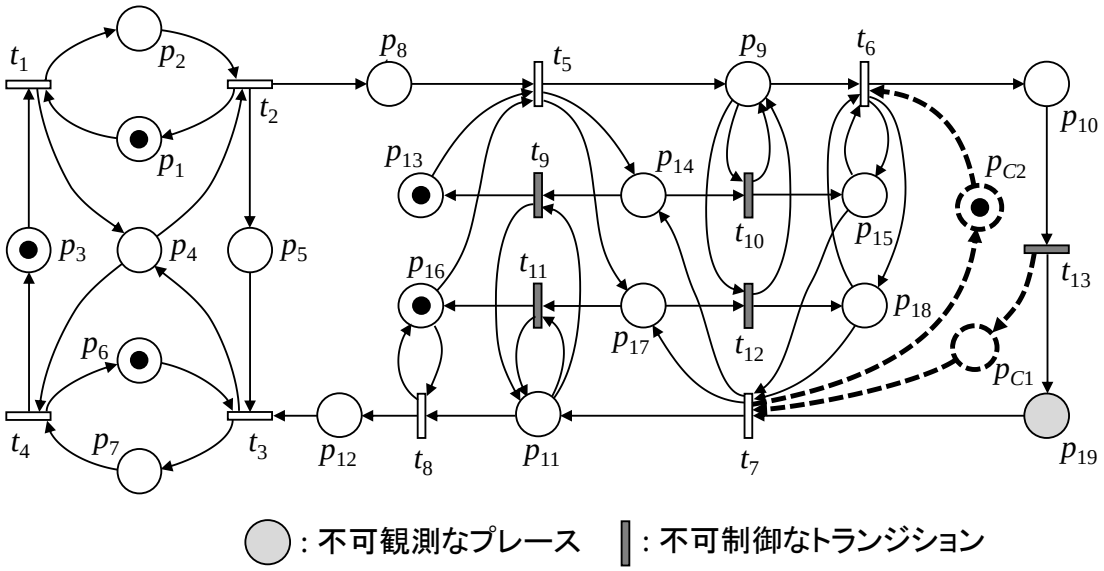


図 6-8　実現可能な制御器

6.5. 異常動作回避問題

第 4 章で示したように，本研究では異常の回避の考え方を，異常状態の回避と異常動作の回避の 2 つのタイプに分類した．6.4 においては，不可観測な状態，不可制御な動作が含まれるバッチプロセスにおいて，異常状態の回避に着目した制御器の設計手法に係る検討を行った．本節では，異常動作の回避に着目する．

第 5 章で示したように，本研究ではバッチプロセスの異常動作の回避のための制御器設計問題を，拡張 C/E ネット制御問題P_{EC}^{ce}として定式化した．よって，不可観測な状態，不可制御な動作を含むバッチプロセスの異常動作の回避に係る検討も，問題P_{EC}^{ce}をこれに対応するように拡張することが自然と考えられる．また，異常動作の回避を考える場合，状態に係る情報は必要ないので，以下では，不可制御な動作のみを取り扱う．5.2.3 では問題P_{EC}^{ce}を考えるにあたり，制御器内での動作を導入し，制御器トランジションの集合T_Cを(6.19)式（(5.3)式と同じ）のように拡張した．

$$T_C = T_{CP} + T_{CC} \text{ ただし } T_{CP} \subseteq T_P \text{ , } T_{CC} \cap T_P = \phi \tag{6.19}$$

ここで，T_{CP}はプラントトランジションの集合T_Pの部分集合となるトランジションの集合で，T_{CC}は制御器内での動作を表すトランジションの集合であり，+は直和を表す．次に，プラントトランジションの集合T_Pを可制御なプラントトランジションの集合$T_{P,C}$と不可制御なプラントトランジションの集合$T_{P,UC}$に分割する．

$$T_P = T_{P,C} + T_{P,UC} \tag{6.20}$$

T_{CP}は$T_{P,C}$の部分集合であることから，(6.19)式は次のように変換される．

$$T_C = T_{CP} + T_{CC} \text{ ただし } T_{CP} \subseteq T_{P,C} \text{ , } T_{CC} \cap T_P = \emptyset \tag{6.21}$$

よって，不可制御な動作を含む C/E ネット制御問題$\mathrm{P}_{EC,U}^{ce}$は次のように定式化できる．

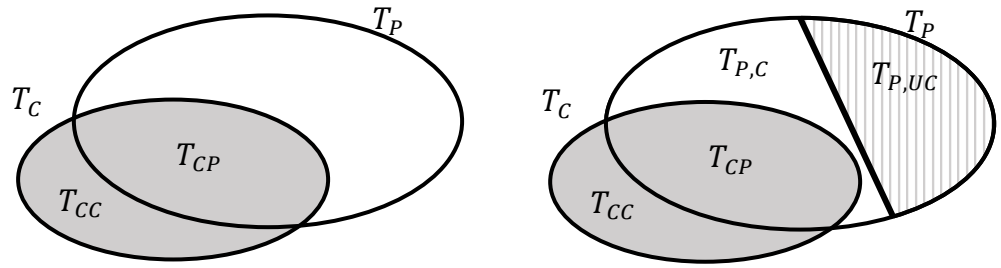
不可制御な動作を含む C/E ネット制御問題$\mathrm{P}_{EC,U}^{ce}$：

バッチプロセスの動作を表すプラントネットN_P，バッチプロセスの可制御な動作を表すトランジションの集合$T_{P,C}$，バッチプロセスの不可制御な動作を表すトランジションの集合$T_{P,UC}$とし，$T_P = T_{P,C} + T_{P,UC}$とする．動作仕様（異常事象の発生を含まない動作）としてT_P上の半言語Xが与えられたとき，(6.23)式のもとで(6.22)式を満たすような制御器ネットN_Cを求めよ．

$$X = FP(N_P \oplus N_C)|T_P \tag{6.22}$$

$$T_C = T_{CP} + T_{CC},\ T_{CP} \subseteq T_{P,C},\ T_{CC} \cap T_P = \emptyset \tag{6.23}$$ ■

問題P_{EC}^{ce}と問題$\mathrm{P}_{EC,U}^{ce}$の違いを図 6-9 に示す．



$T_C = T_{CP} + T_{CC}$

問題PB_{EC}^{ce}

$T_C = T_{CP} + T_{CC}$
$T_P = T_{P,C} + T_{P,UC}$

問題$\mathrm{PB}_{EC,U}^{ce}$

T_P ：プラントトランジションの集合

T_C ：制御器トランジションの集合

図 6-9　問題$\mathbf{P}_{EC}^{ce}$と問題$\mathbf{P}_{EC,U}^{ce}$におけるトランジションの集合の違い

問題$\mathrm{P}_{EC,U}^{ce}$においては，プラントトランジションのうち，可制御な動作を表す$T_{P,C}$の要素のみから制御器を構成する．よって，制御器プレースから不可制御なトランジションへのアークが結ばれることは無い，すなわち$F_C \subseteq (P_C \times T_{P,C}) \cup (T_{P,C} \times P_C)$である．

ここで，バッチプロセスにおいてその動作の完了が観測できない”不可観測な動作”を新たに考える．これをペトリネット上で表現すると，次のように定義できる．

不可観測なトランジション：

発火したことを直接検出することができないトランジションを不可観測なトランジションと呼ぶ． ■

問題$\mathbf{P}_{EC,U}^{ce}$を解くことで得られるN_Cのアークは$F_C \subseteq (P_C \times T_{P,C}) \cup (T_{P,C} \times P_C)$であるため$T_{P,UC}$をトリガーとした$N_C$が構成されることは無い．よって，$T_{P,UC}$に不可観測な動作も要素に加えれば，問題$\mathbf{P}_{EC,U}^{ce}$は不可制御な動作，不可観測な動作の両方に対応していることとなる．

動作仕様Xに適合するアトムの集合$CA(X)$の部分集合として，可制御なトランジションから構成されるアトムの集合$CA(X)_C$を考える．T_nをアトムnに含まれるトランジションの集合とすると，$CA(X)_C = \{n \mid T_n \subseteq T_{P,C}\}$である．また，$ECA(X)_C$を$ECA(X)$の部分集合とすると，$ECA(X)_C = \{n \mid T_n \cap T_{P,UC} = \emptyset\}$である．また，$N_P = \bigoplus S_P$とする．このことから，問題$\mathbf{P}_{EC,U}^{ce}$の解の存在性に関して次のことが明らかに成り立つ．

問題$\mathrm{P}_{EC,U}^{ce}$の解の存在性：

(1) $S_P \subseteq CA(X)$ならば，問題$\mathrm{P}_{EC,U}^{ce}$の解N_Cが存在するための必要十分条件は $X = FP\left(\oplus\,(S_P \cup S_C)\right) | T_P$が成り立つような$S_C \subseteq ECA(X)_C$が存在することである．

(2) $S_P \not\subseteq CA(X)$ならば，問題$\mathrm{P}_{EC,U}^{ce}$の解N_Cは存在しない ■

このことから，問題$\mathrm{P}_{EC,U}^{ce}$の解法は，5.2.3 で示した解法の$ECA(X)$を$ECA(X)_C$で置き換えたものとなる．よって，問題P_{EC}^{ce}と同様に，この問題も本質的には組合せ問題となる．効率的な解の探索法の検討は，今後の課題である．

例題 5-3：

例題 5-2 と同じ構造のプラントネットN_{P2}，動作仕様X_2を考える．N_{P2}のプラントトランジションeは不可制御であり不可観測なトランジションとする．例題 5-2 で得られた制御器ネットN_{C2}はトランジションeが含まれており，これは実現できない．トランジションeを含まない制御器ネットN_{C3}の一例を図 6-10 に示す．図中の破線はN_{C3}を表し，トランジションeは含まれていない． ■

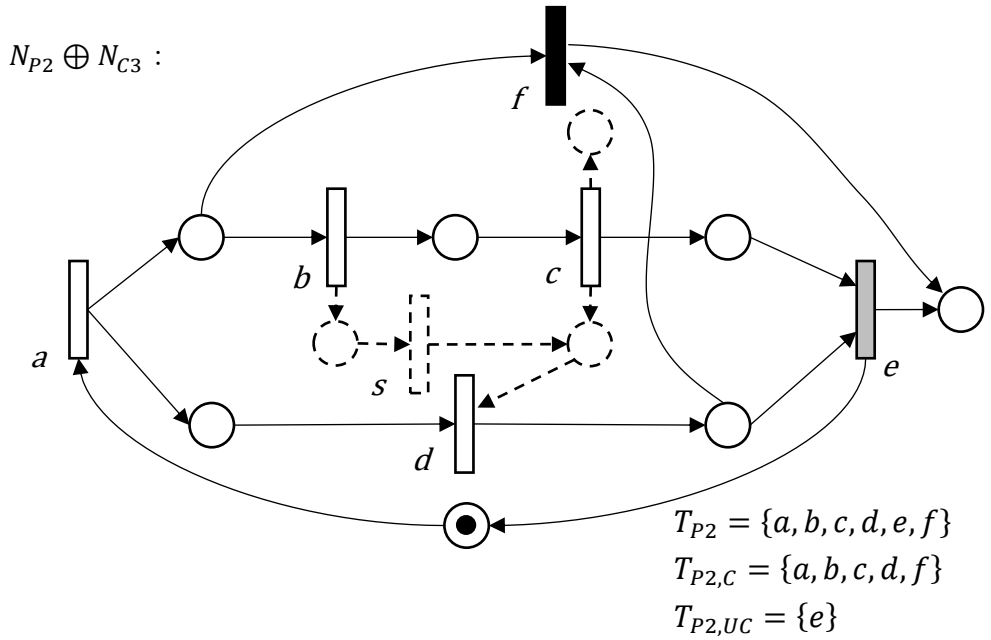


図 6-10　制御器ネットN_{C3}を合成した閉ループシステムネット$N_{P2} \oplus N_{C3}$

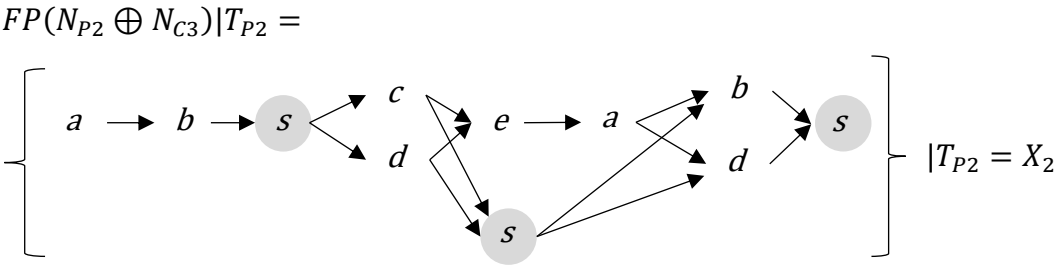


図 6-11　閉ループシステムネットN_{P2}⊕N_{C3}の動作

6.6.　まとめ

バッチプロセスの特徴の一つに，機械・電気系に比べ，その状態の観測が不可能なプロセス変数である“不可観測な状態”および制御が不可能な動作である“不可制御な動作”の要素が頻出することが挙げられる．しかし，これらの要素を同時に含む異常の回避のための制御器設計手法は確立されていなかった．そこで本章では，不可観測な状態，不可制御な動作を含むバッチプロセスにおける異常の回避のための制御器設計に係る検討を行った．

6.2 では，ペトリネットの上で，不可観測な状態，不可制御な動作の表現方法について検討した．6.3 では，異常の検出のために用いる動作制約に不可観測な状態が含まれる場合における検討を行った．第 3 章では，動作制約はバッチプロセスの正常な動作を表すペトリネットモデルから，プレースインバリアント解析によって導出されることを示した．そして，同じペトリネットから導出されるプレースインバリアント同士を線形結合することで得られるベクトルも，そのネットのプレースインバリアントとなる性質を利用し，動作制約含まれる不可観測な状態の要素をできるだけ少なくし，検出の精度を高める手法を提案した．また，不可観測な状態が取りうる値を推定することで，動作制約の不可観測な状態に係る要素を削除し，可観測な状態のみから構成される新たな動作制約（不等号動作制約）を導出する方法を提案した．6.4 では，不可観測な状態，不可制御な動作を含むバッチプロセスにおける異常の回避のため，新たに導出した不等号動作制約を順守することを目的とした制御器の設計手法を開発した．6.5 では，異常動作を含まない望みの動作を行うことを目的とした制御器を設計する異常動作回避問題 PB を，不可制御な動作や，その完了が観測できない動作である“不可観測な動作”を含むバッチプロセスに適用するための検討を行った．そこでは，第 5 章で定式化した拡

張 C/E ネット制御問題P_{EC}^{ce}をこれらの動作が扱えるように拡張した，不可制御な動作を含む C/E ネット制御問題$\mathrm{P}_{EC,U}^{ce}$を定式化した．そして，不可制御な動作や不可観測な動作が含まれないアトムの集合から制御器を設計する手法を提案した．

本章では，不可観測な状態や不可制御な動作を含むバッチプロセスにおいても，その異常の回避は，第 4 章，第 5 章で示した異常状態の回避，異常動作の回避と同じ考え，すなわち“動作制約の順守”と“異常動作を含まない望みの動作の実行”で検討することが可能であることを示した．一方，異常状態の回避においては，可達マーキングを求めることが困難であることから，厳密解ではなくヒューリスティックな解の探索方法を提案した．また，異常動作の回避においては，この問題が本質的に組み合わせ問題となることから，これにおいても，効率的な解の探索方法が必要となることを示した．これら，解の探索方法に係る検討は，今後の課題である．

本章までの内容は，バッチプロセスにおける状態・動作間の因果関係の変化や，プロセスに含まれる非線形な要素等による不確実性は考慮していなかった．しかし，化学プロセスの特徴は，非線形性の高い物理量に加え，プロセス特性・環境変化・運用変更等にあり，これらをバッチプロセスにとっての不確実性ととらえるならば，不確実性を含むプロセスへの対応も重要である．第 7 章では，不確実性を含むバッチプロセスにおける異常の回避に係る検討を行う．

第7章. 不確実性を含むバッチプロセスの異常の回避

バッチプロセスの特徴は，非線形性の高い物理量と突然のシステム特性・環境変化にある．したがって，バッチプロセスの運用にあたっては，これらの不確実性にも適切に対応できることが必要となる．このため，さまざまな不確実性のもとで，将来，発生することが予想される異常を回避することもバッチプロセスの運用にとっては重要となる．本章では，不確実性を伴うバッチプロセスを対象に，不確実性の影響を予測しながら異常状態に陥ることを回避するための方法を示し，その適用可能性について議論する．

7.1. はじめに

本論文ではこれまでに，バッチプロセスの動作を離散事象システムとしてとらえ，その記述方法としてペトリネットを用い，異常の診断・回避のための方法論を展開してきた．そこでは，状態および動作に係る因果関係をもとに，発生した異常を検出し，その原因を特定し，その再発を回避するための制御器を構成してきた．しかし，現実のバッチプロセスにおいては，プロセスの特性やそれを取り巻く環境の変化，および運用形態の変更等によって，状態・動作間の因果関係に変化が現れることもある．例えば，一般に晶析を伴う化学反応は反応の過度期においては組成の影響を強く受けるものの，一旦，定常期に到達すると組成よりもむしろ温度からの影響を受けることも多い．化学プロセスの特徴は，非線形性の高い物理量に加え，このようなプロセス特性・環境変化・運用変更等にある．したがって，これらをプロセスにとっての不確実性ととらえるならば，不確実性を伴うプロセスへの対応も重要である．

さらには，現実のバッチプロセスにおいては発生した異常に対応するばかりでなく，予想されるさまざまな不確実性のもとで，状態・動作に係る因果関係の変化を推定しながら，将来，発生することが予想される異常にも対応することが必要となる．本研究では，これに対応するため，不確実性の影響を確率過程を使って表現し，そのもとでプロセスの将来動作が表現できる動的な確率モデルを採用することとした．そして，これに基づく将来，発生が予想される異常を診断・回避するための方法を考案した．

本章では，これら不確実性を含むバッチプロセスの異常の回避に係る検討を行い，その可能性について議論する．本章の構成は次の通りである．7.2 では，不確実性を伴うバッチプロセスの異常回避のためのシステムモデルとして用いるベイジアンネットワークについて，その採用理由とそれを動的モデルに変換する手法について説明する．7.3 では，不確実性を伴うバッチプロセスにおける異常回避問題を定式化し，7.4 では，その解法として提案する 2 つの手法について説明する．7.5 では，現実の生産プロセスを簡略化した模擬プロセスに対し，提案手法を適用し，その有効性を確認する．7.6 では，これまで検討したペトリネットモデルによる異常回避と，本章で提案するベイジアンネットワークによる異常回避を比較し，それぞれの特徴をまとめる．

7.2. 不確実性を伴うバッチプロセスのモデル化

ベイジアンネットワークは確率モデルの 1 つであり，確率変数間の因果関係を非巡回有向グラフで表したものである．それはベイズ統計学を基本とし，変数間の因果関係は条件付き確率で表される．ベイジアンネットワークを使うにあたっての自由度は高く，そのため現在では医療診断，危険予知，機械学習等様々な対象に適用が試みられている[50].

本研究では，不確実性を伴うバッチプロセスの異常回避のためのシステムモデルとして，このベイジアンネットワークを採用することとした．その一例を図 7-1 に示す．ここで，各ノードはバッチプロセスのプロセス変数に相当する確率変数を表し，各アークはその親ノードに原因をその子ノードに結果をもつ因果関係を表す．また，それぞれの確率変数間には対応する因果関係の大きさを表す条件付き確率表が添えられている．

本研究において，不確実性を伴うバッチプロセスのモデルにベイジアンネットワークを採用した理由は次のようである．

ベイジアンネットワークの採用理由:

(1) バッチプロセスでは状態・動作の因果関係が複雑に絡み合っているが，ベイジアンネットワークはこの構造を分かり易くグラフ表現することができる．

(2) ベイジアンネットワークは，バッチプロセスに現れる不確実性の影響およびその大きさをプロセス変数間の条件付き確率表で表すことができる．

(3) ベイジアンネットワークは，将来，発生する可能性のある異常を確率推論を使って予測することができる．

(4) 機械学習のツールとしてのベイジアンネットワークの機能を活用することによって，不確実性がバッチプロセスに及ぼす影響を学習することができる． ■

ベイジアンネットワークは本来は静的なモデルである．このため，時間経過に伴う状態・動作の変化である動特性を表すためには，ベイジアンネットワークに何かの仕掛けを施すことが必要となる．本研究では，同じ構造をもつベイジアンネットワークを時間軸に沿って複数個作成し，関連するノードで隣接するネットワークをつなげることによってバッチプロセスの動特性を表すこととした．これはダイナミックベイジアンネットワークと呼ばれている動的モデルであり，人間の行動モデル等に応用されているものである[51][52]．ダイナミックベイジアンネットワークの一例を図 7-2 に示す．図では赤に相当するノードが隣り合うネットワーク共通のノードとなっており，これを共有することによって時間経過を考慮した一連のネットワークが構成されることとなる．

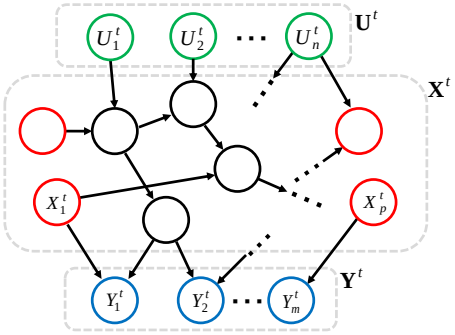


図 7-1　時刻tにおけるベイジアンネットワーク

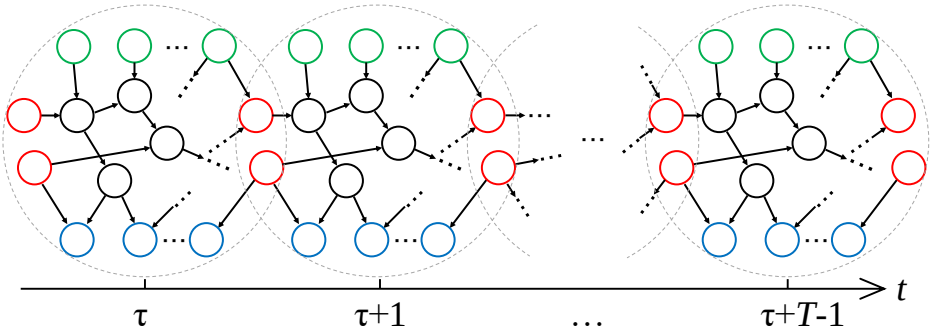


図 7-2　ダイナミックベイジアンネットワーク

7.3. 異常回避問題の定式化

本研究で開発した不確実性を伴うバッチプロセスのモデル化手法を説明するため，次のような異常状態回避問題を考える．なお，この問題は，現実の生産プロセスの一部を抜き出し，それを抽象化した問題である．

不確実性を伴うバッチプロセスの異常回避問題 PU:

不確実性を伴うバッチプロセスにおいて，現在までに得られている状態履歴に関する情報をもとに，プロセスの状態が将来も異常状態に陥ることなく目標状態（または許容状態）を維持するように操作せよ． ■

この問題の難しい点は，プロセスに不確実性が加わるため，将来に渡ってプロセスの状態が異常状態に陥ることなく目標状態を維持できるような不変の行動を現実には決定できない点にある．したがって，そこでは，不確実性の影響を予測しながら異常回避のための行動を決定しなければならない．

問題 PU の構造をダイナミックベイジアンネットワークで表現するにあたり，それぞれのプロセス変数を確率変数で表す．そして，プロセス変数のうち操作変数U_iに相当するものの集合を$\mathbf{U}$で，制御変数Y_iに相当するものの集合を$\mathbf{Y}$で，それ以外のプロセス変数X_iに相当するものの集合を$\mathbf{X}$で表し，この$\mathbf{X}$をあらためて状態変数と呼ぶ．

$$\mathbf{U} = \{U_1, U_2, \cdots, U_n\} \tag{7.1}$$

$$\mathbf{Y} = \{Y_1, Y_2, \cdots, Y_n\} \tag{7.2}$$

$$\mathbf{X} = \{X_1, X_2, \cdots, X_n\} \tag{7.3}$$

ここで，状態変数のうちの一部は観測可能であるが，一部は観測できないものとする．すなわち，

$$\mathbf{X} = \mathbf{X}_O \cup \mathbf{X}_U \tag{7.4}$$

であり，$\mathbf{X}_O$は可観測な状態変数の集合を，$\mathbf{X}_U$は不可観測な状態変数の集合を表す．以下では，確率変数で表された時刻tにおけるそれぞれのプロセス変数を添え字tを使ってU_i^t, Y_i^t, X_i^tで表し，それらの実現値を小文字を使ってu_i^t, y_i^t, x_i^tで表す．さらに，tにおけるY_iの目標値をY_i^{*t}で表す．

以上のような準備のもとで，バッチプロセスモデルにベイジアンネットワークを採用することによって，上の問題 PU は次のような問題として定式化できる．

不確実性を伴うバッチプロセスの異常回避問題 PU:

不確実性を伴うプロセスの状態が，常に時間区間$\tau \le t \le \tau + T$（τは現在時刻，Tは定数）において異常状態に陥ることなく，できる限り制御変数の実現値$\{y_i^{\tau \le t \le \tau+T}\}$がその目標値である$\{y_i^{*\tau \le t \le \tau+T}\}$を維持できるような$\tau$における操作変数の実現値$\{u_i^{\tau}\}$を求めよ．ただし，$\tau$より前の$\{u_i^{t<\tau}\}$,$\{y_i^{t<\tau}\}$および$\{x_{0i}^{t<\tau}\}$はいずれも既知とする． ■

この問題はバッチプロセスが不確実性の影響を受けるなか，常に現在時刻よりTまで先の状態・動作を予測しながら，プロセスの状態が期間Tにおいてできる限り目標状態に近い状態を維持するよう現時点での操作量を繰り返し決定しようとする問題である．この問題の定式化においては，バッチプロセスに現れる不確実性は確率変数間の因果関係の大きさを表す確率表の変化によって表現される．

7.4. 異常回避の方法

ベイジアンネットワークの特徴は，原因から結果を予測するだけでなく，結果から原因を推定することもできる点にある．したがって，この確率推論を活用することによって，将来の目標状態から時間に関して逆向きにたどりながら現在，行うべき操作を導出できるものと考えられる．

本研究では，観測状態$\mathbf{X}_O^{\tau} = \{x_{0i}^{\tau}\}$と，目標状態$\mathbf{Y}^{*\tau \le t \le \tau+T} = \{y_i^{*\tau \le t \le \tau+T}\}$をもとに，これを次のように行うこととした．

操作量の決定（方法 1）：

$$u_i^{\tau} = \arg\max_{U_i^{\tau}} P(U_i^{\tau} \mid \mathbf{Y}^{*\tau}, \mathbf{Y}^{*\tau+1}, \cdots, \mathbf{Y}^{*\tau+T-1}, \mathbf{X}_O^{\tau}) \tag{7.5}$$

(7.5)式は，時刻τで観測された状態$\boldsymbol{X}_O^{\tau}$のもとで，時間区間$\tau \le t \le \tau + T$において目標状態$\mathbf{Y}^{*\tau \le t \le \tau+T}$が達成されたと仮定したときに，その可能性が最も大きい操作変数の値u_i^{τ}（i=1,…,n）をそれぞれ個別に求めるものである．したがって，そこでは他の操作変数の影響については考慮されていない．このため，複数の操作変数が互いに影響しあうときには，(7.5)式では必ずしも望ましい結果が得られないであろう．そこで，本研究では，操作変数間の干渉が大きいプロセスのときは，(7.5)式に代わって次のように操作量U^{τ}を決めることとした．

操作量の決定（方法 2）：

$$(u_1^\tau, \cdots, u_n^\tau) = \arg \max_{(U_1^\tau, \cdots, U_n^\tau)} P(U_1^\tau, \cdots, U_n^\tau \mid \mathbf{Y}^{*\tau}, \mathbf{Y}^{*\tau+1}, \cdots, \mathbf{Y}^{*\tau+T-1}, \mathbf{X}_O^\tau) \tag{7.6}$$

(7.6)式は$\mathbf{X}_O^\tau$のもとで目標状態$\mathbf{Y}^{*\tau \le t \le \tau+T}$が達成されたと仮定したとき，$n$個の操作変数のすべての組合せのなかからその可能性が最も大きい組合せを推定し，それを操作量とするものである．なお，(7.5)式と(7.6)式は各操作変数が条件付き独立，すなわち

$$P(U_1^\tau, \cdots, U_n^\tau \mid \mathbf{Y}^{*\tau}, \mathbf{Y}^{*\tau+1}, \cdots, \mathbf{Y}^{*\tau+T-1}, \mathbf{X}_O^\tau) = \prod_{j=1}^{n} P\left(U_j^\tau \mid \mathbf{Y}^{*\tau}, \mathbf{Y}^{*\tau+1}, \cdots, \mathbf{Y}^{*\tau+T-1}, \mathbf{X}_O^\tau\right) \tag{7.7}$$

が成り立つならば同等となる．しかし，そうでなければ，両者は必ずしも同等とはならない．以上が本研究において考案した操作量の決定方法である． ■

7.5. 適用例

対象プロセス

適用例として，図 7-3 に示す 4 つの処理装置（M_1－M_4）と，材料の置場となる 4 つのバッファ（Y_1－Y_4）からなる生産プロセスを考える．なお図中の数値は運用条件を示したものである．このプロセスには，毎日初めに 2 種類の材料 A，B がまとまって到着し，Y_1 に置かれる．それぞれの材料は図 7-3 に示されたルートで処理される．ただし材料 A は，M_1 での処理時に処理速度に応じた発生率で不良が発生し，再処理材料として Y_1 へ戻される．M_1 の処理速度は 3 段階に設定でき，処理速度が短いほど不良発生率が高い．M_3,M_4 は 1 日 1 回指定された数（バッチサイズ）の材料を 1 日かけてまとめてバッチ処理する．M_3 で指定できるバッチサイズは 3 種類であり，M_4 は Y_3 の材料数が一定値を超えていれば，すべての材料を処理する．M_3,M_4 とも，各日の初めに直前のバッファに必要な数がないときに限ってその日の処理を行わない（1 日の途中で必要量に達しても処理は開始されない）ものとする．また Y_4 の製品はそれぞれ 1 個ずつ一定速度で搬出される．

この生産プロセスに対し，次の異常回避問題を考える．

異常回避問題：

日毎の Y_1，Y_2，Y_4 の材料数が，それぞれ指定された閾値を超えた状態を異常状態とする．プロセスが異常状態に陥るのを回避するよう，M_1 の処理速度および M_3 の

バッチサイズを決定せよ．ただし処理速度およびバッチサイズの変更は，材料が到着したタイミングでのみ行うものとする． ■

ダイナミックベイジアンネットワークの構築

対象プロセスを変数間の因果関係をもとに DBN によりモデル化する．なお，対象プロセスにおける変数間の因果関係はすべて既知であるものとする．作成した時刻 t におけるネットワークを図 7-4 に示す．各ノードは，緑色が操作変数（M_1 の処理速度，M_3 のバッチサイズ），青色が目標変数，赤色が時刻間に共通する状態変数を表す．またそれぞれのノードが表す内容に表 7-1 示す．異常回避問題を解くためには，目標変数として Y_1，Y_2，Y_4 の材料数を表す確率変数を考える必要がある．一方バッファの材料数を表す状態変数値は，離散化した数値として与えられている．したがって与えられた目標を満たすには，閾値以下である状態変数値の確率の合計が 1 となればよいことになる．しかし条件を満たす状態変数値が複数存在する場合には，合計が 1 となる状態は無数に存在する．そこで，目標となる各バッファの材料数を表すノードを親とする子ノードとして，バッファの材料数が閾値以下（正常値）であるか閾値より大きい（異常値）かの 2 値をとる変数 $J^t_{Y_i}$を作成した．そしてその条件付き確率表を，材料数が閾値以下であれば正常値の確率が 1 となるように与えることで，材料数の判別をする判別ノードとし，それらを目標変数とした．

図 7-4 のネットワークを合成することで現時刻 τ から T 日分の対象プロセスの動的挙動を表す DBN モデルを得る．

数値実験

対象プロセスの動作を行うシミュレータを作成し，提案手法に基づいて 30 日間運転した状態を模擬する数値実験を行い，提案した異常回避の方法の有効性を確認する．対象プロセスは不確実性を含んでおり，実験にあたっては 200 通りの初期状態と材料到着パターンをランダムに作成し，方法 1 および方法 2 に基づく 2 種類の提案手法を適用した．また，比較のため，ヒューリスティクスに基づく手法（局所探索法）による運転も行った．

局所探索法：

(1) M_3 では，Y_4 にある製品個数が閾値を超えないように，Y_3 からの流入量を考慮しながら処理バッチの大きさを決定

(2) M_1 では，Y_2 の材料 A が不足しないように，処理時間を決定

結果および考察

各方法において，閾値を超越，つまり異常状態となった平均日数を表 7-2 に示す．なお，$E_{\mathrm{av}}(\mathrm{Y}_i)$は$\mathrm{Y}_i$において異常状態となった日数の 200 回の試行の平均を表す．方法 1 において，$T = 1$の場合，すなわち 1 日分のネットワークしか用いない場合には Y_1 での超過回数が多くなっているが，$T = 3$の場合には結果が大幅に改善されている．この結果から，より先の時刻における状態を考慮した場合は，その分余裕を考慮することになり，異常状態に陥りにくいことが分かる．さらに，$\mathrm{Y}_2, \mathrm{Y}_4$ の結果も改善されており，より先の状態を予測することで，より望ましい異常回避が実現できていることを示している．一方，方法 2 においては，$T = 1$の場合でも方法 1 に比べて良い結果が得られている．したがってこのプロセスでは，操作変数の相互作用を考慮することが重要であることがわかる．また，局所探索法では，Y_2 と Y_4 で異常状態となっている期間が長く，局所的な情報のみでは，異常の回避が難しいことが分かる．

図 7-5 は$T = 3$とした場合のある試行における，各バッファの材料数の日ごとの変化を示したものである．方法 1 および方法 2 では，局所探索法に比べて Y_2 と Y_4 の負荷を，Y_1 へと移行させていることが見てとれる．これを実現するには，到着個数の不確実性や不良品の発生，M_4 での処理のばらつき等，複数の不確実な要素に対応する必要がある．提案手法はこのような不確実性を考慮しながら操作量を決定することを可能にしている． ■

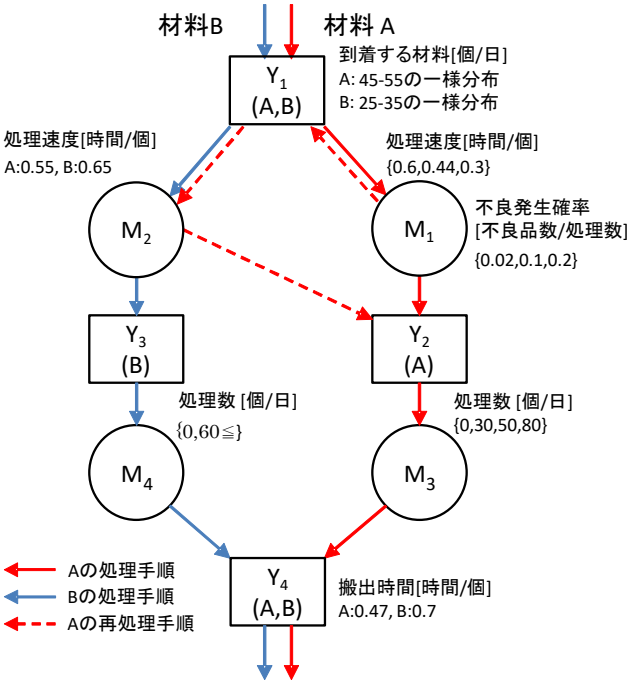


図 7-3　生産プロセス

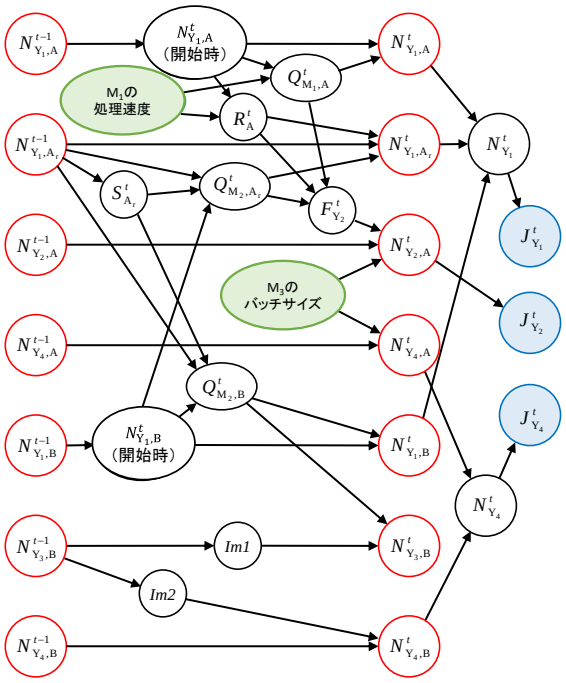


図 7-4　ベイジアンネットワーク

表 7-1　確率変数の説明

$N^t_{Y_i}$	Y_i における材料の量
$N^t_{Y_i,A}$，$N^t_{Y_i,B}$	Y_i における材料 A/B の量
$N^t_{Y_1,A_r}$	Y_1 における再処理材料 A の量
$Q^t_{M_i,A}$，$Q^t_{M_i,B}$	M_i での材料 A/B の処理数
$Q^t_{M_2,A_r}$	M_2 での再処理材料 A の処理数
R^t_A	再処理材料 A の数
$F^t_{Y_2}$	Y_2 への材料流入量
S^t_A	再処理材料 A の処理開始有無
$Im\ 1$，$Im\ 2$	中間変数
$J^t_{Y_i}$	Y_i における材料の量が閾値を超えたか否か

表 7-2　シミュレーション結果

	方法 1		方法 2		局所探索法
	T=1	T=3	T=1	T=3	
$E_{av}(Y_1)$	6.425	0.205	0.525	0.060	0.000
$E_{av}(Y_2)$	1.185	0.630	0.505	0.540	6.210
$E_{av}(Y_4)$	0.880	0.715	0.425	0.790	2.680

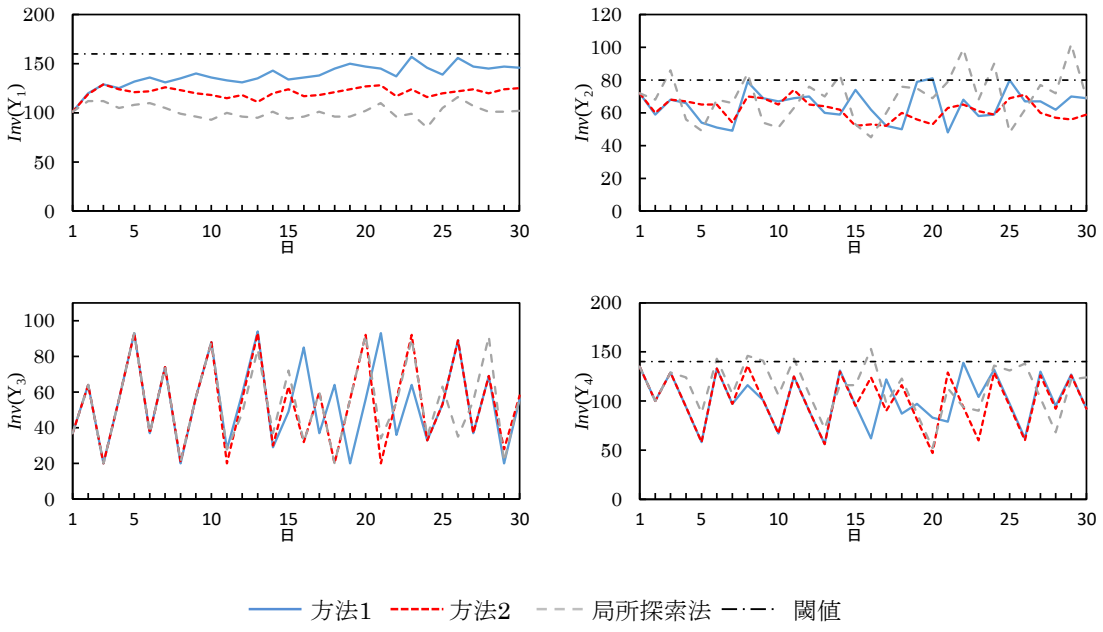


図 7-5　各バッファに置かれた材料の時間変化

7.6. 考察

7.1 にも示したように, 本論文では第 6 章までにバッチプロセスに発生した異常を検出しその再発を回避する方法を示した．そして，本章では，不確実性を伴うバッチプロセスに発生することが予想される異常を回避する方法を示した．そこでは，前者についてはペトリネットでバッチプロセスを表現し, 後者についてはベイジアンネットワークでこれを表現した.

ペトリネットモデルの作成にあたっては，バッチプロセスの状態と動作に着目し，動作によって引き起こされる状態遷移によってプロセスの振る舞いを表現した．一方，ベイジアンネットワークモデルの作成は, これとは全く異なる視点であるプロセス変数間の因果関係に着目し，これを表現した．したがって，そこには陽な形での状態・動作は表現されておらず, プロセスの振る舞いはその因果をたどることによって知ることとなる．このようなモデルを採用したのは，バッチプロセスに現れる不確実性を表現するためには，後者の方が自然であろうと考えたためである．現実のバッチプロセスに加わる不確実性は，それが突発的に状態や動作を変えるのではなく，7.1 に示した反応速度を決める要因が組成から温度に変わる例のように, さまざまな原因によってプロセス変数間の因果関係とその大きさが変わることによって現れるものと考えられる. 現実のバッチプロセスに出現する種々の不確実性を分析しそれらの考察をもとに，本章では，不確実性を自然な形で表現できるベイジアンネットワークをバッチプロセスモデルに採用することとした．これらをまとめて， 表 7-3 に示す.

本研究では，これらの異なる目的をもつ異常回避問題を，それぞれ個別に取り扱った．一方，これら 2 つの方法を組み合わせ，それぞれの手法の利点を活かすことで，より合理的な異常回避が実現できる可能性もある．例えば，不確実性を伴うバッチプロセスにおいて，将来のある時刻の最も確率の高い状態・動作の因果関係およびそれらと異常の因果関係をベイジアンネットワークを用いて予測する．そして，その因果関係を抽出し，それをペトリネットで記述する．プロセスの状態・動作の因果関係と異常が陽に記述されたペトリネットモデルが得られれば，第 3 章～第 6 章で扱った異常対応に係る手法によって，その解決を図ることができる.

これら 2 つの方法の組合せに係る検討は，今後の課題である.

表 7-3　異常回避のとらえ方

	ペトリネットモデルを使った異常回避（第 3 章～第 6 章）	ベイジアンネットワークを使った異常回避（第 7 章）
目的	“発生した”異常の検出・特定とその再発の回避	“将来発生が予想される”異常の回避
モデルの着眼点	離散化された状態・動作のもとでの状態遷移に着目	状態・動作よりもむしろプロセス変数間の因果関係に着目
モデルの表現	確定モデルとしてのペトリネットで表現	確率モデルとしてのベイジアンネットワークで表現
不確実性の表現	陽な形で表現せず	因果関係を表す確率表で表現
回避の方法	現在までの状態・動作履歴を活用することによって回避	将来挙動を予測することによって回避

7.7.　まとめ

本章では，プロセス特性・環境変化・運用変更等の不確実性を含むバッチプロセスを対象に，その不確実性によって生じる状態・動作に係る因果関係の変化を推定しながら，将来，発生することが予想される異常の回避に係る検討を行った．7.2 では，不確実性を含むバッチプロセスのモデル化手法として，ベイジアンネットワークを用いることとした．また，本モデルは本来静的なシステムモデルであるが，同じ構造をもつモデルを時間軸に沿って複数個作成し，関連するノードで隣接するネットワークをつなげることで，動的なモデルとするダイナミックベイジアンネットワークを構築し，これを不確実性を含むバッチプロセスの異常回避のためのシステムモデルとした．7.3 では，不確実性を伴うバッチプロセスにおいて，“現在までに得られている状態履歴に関する情報をもとに，プロセスの状態が将来も異常状態に陥ることなく目標状態（または許容状態）を維持するように操作せよ”とした不確実性を伴う異常回避問題を定式化し，7.4 ではその解法として，操作量を独立に解く手法と，操作量間の相関性を考慮して解く方法の 2 つを提案した．7.5 では，提案した 2 つの方法を現実の生産プロセスを簡略化した例題に適用し，不確実性をバランス良く考慮し，将来予測を行いながら最適な操作量を決定する提案手法は，予測を行わず局所的な情報のみを扱う手法より，不確実性を伴うプロセスの異常回避に効果的であることを確認した．

第8章. 結言

8.1. 本研究の成果

本研究ではプロセスシステム工学的な視点から，バッチプロセスにおける異常の診断とその回避のための合理的な方法論を確立することを目的とした．そして，その目的を達成するため，次の 3 つの検討を行った．

[1] 異常の診断とその回避を統一的に扱う方法論の確立．

[2] 不可観測な状態，不可制御な動作の要素を含むプロセスを対象に，異常の回避の方法論の確立．

[3] 不確実性の要素を含むプロセスを対象に，将来の予測に基づき異常を回避する方法論の確立．

各章における具体的な成果は次の通りである．

第 2 章では具体的な検討する前の準備として，従来その概念が曖昧であった異常を一般化された形で定義し，定義した異常の動的モデルの上での表現方法を明らかにした．ここでは，化学プロセスにおける異常を，プロセスが正常に動作する限り，決して現れることがない状態である“異常状態”とプロセスの状態を正常状態から異常状態に遷移させる動作である“異常動作”と定義した．そして，化学プロセスで発生する異常を，1 つのプロセス変数がそれ単独で示す異常である“異常 F1”と個々のプロセス変数は正常であっても，それらを組として見たとき現れる異常である“異常 F2”に分類した．異常 F2 に関しては，これまでに十分な検討がなされておらず，その検出方法も確立されていなかった．本研究では異常 F2 をとらえるため，プロセスが正常な動作をする限り複数のプロセス変数が満たすべき条件である“動作制約”を定義し，動作制約の成否に基づき，異常の検出を行うことを提案した．また，これらの異常に係る問題を取り扱うための離散事象システムモデルとして，バッチプロセスの動作の特徴である逐次的動作，並行的動作，非決定的動作を陽に記述でき，システムモデル固有の解析手法を備え，階層表現に優れたペトリネットに着目し，これを本研究で用いる異常対応モデルとした．そして，ペトリネットモデル上で，異常状態を表すマーキングや異常動作を表すトランジションを用いることで，現実のプロセスにおける異常が表現可能であることを明らかにした．

また，動的モデルの上で異常対応に係る問題を解くことを考えた場合，それを矛盾なく行うためには，将来起こりうる異常をすべて予測し，それをモデルに組み込むことが必要となる．しかし，すべての異常を予測することは非常に困難であり，現実的には不可能と考えられる．そこで，本研究では，未経験の異常が発生するたびに，その異常を表す状態や動作を既存のモデルに組み込むことで，徐々に扱うことのできる異常を増やす現実的なモデルの構築手順を提案した．

[1]については，第 3 章から第 5 章で検討を行った．ここでは，異常の検出，検出された異常の原因の特定，特定された異常の再発を回避する制御器の設計に関し，それぞれモデルに基づく手法の検討を行った．

第 3 章では，異常の診断に係る検討を行った．本研究では動作制約の成否により，異常の検出を行う方法を提案した．動作制約はプロセスの規模や複雑さが増すと，それを導出することが困難となる．そこで，本研究ではバッチプロセスの正常な動作を表すペトリネットモデルから，プレースインバリアント解析を用いて，動作制約を導出する方法を開発した．また，動作制約に基づき検出された異常が異常対応モデルに含まれている経験済みの異常であれば，それを表す異常動作を異常発生の原因として特定する方法を開発した．そして，先に述べたモデル更新手法と組み合わせることで，異常診断と異常対応モデルの構築を同時に行う方法を示した．

第 4 章および第 5 章では，異常の回避に係る検討を行った．原因の特定された異常は，それを回避する制御器を設計することで，その再発を回避できる．異常の回避の方法として状態と動作に着目し，動作制約を順守することを目的とした制御器を設計する“異常状態回避問題”と異常動作を含まない望みの動作を行うことを目的とした制御器を設計する“異常動作回避問題”の 2 つの制御器設計問題を定式化した．

第 4 章では，異常状態回避問題を取り扱い，動作制約を順守することを目的に，プロセスの状態に係る新たな不等式制約を設定し，これに基づく制御器の設計手法を示した．第 5 章では，異常動作回避問題を取り扱い，バッチプロセスの動作をペトリネットのサブクラスである C/E ネットで表し，半言語で記述されたトランジション発火形態としての動作仕様が与えられた際，バッチプロセスの動作が動作仕様のみとなるような制御器を設計する C/E ネット制御問題P_C^{ce}を定式化し，その解法を示した．一方，問題P_C^{ce}は制御器が常に求まるとは限らない．そこで，問題P_C^{ce}の枠組みでは解くことができなかった問題の解を得ることを目的に，制御器内での動作を加えた拡張 C/E ネット制御問題P_{EC}^{ce}を定式化し，その解法を開発した．また，問題P_C^{ce}を解くことで得られる制御

器は一意に決まらない．そこで，異常の未然の回避という観点に基づき，複数ある制御器の候補の中から構成が最も簡易な制御器を得ることを目的とした制御器設計問題を考案した．そして，その問題を C/E ネット合成問題の一つの問題である C/E ネット最小実現問題に帰着させ，その解法を開発した．

[2]については，第 6 章で検討を行った．ここでは，現実のバッチプロセスに頻出する，その状態の観測が不可能なプロセス変数や制御が不可能な動作を含む“不可観測な状態や不可制御な動作を含むバッチプロセス”に対する異常の回避に係る検討を行い，状態と動作それぞれに着目した制御器設計手法の検討を行った．異常状態の回避においては，不可観測な状態が取りうる値を推定し，動作制約の不可観測な状態に係る要素を削除することで，観測可能な状態のみから構成される新たな動作制約（不等号動作制約）を導出した．そして，バッチプロセスの状態が常に不等号動作制約を遵守することを目的とする制御器の設計手法を開発した．また，不可制御な動作が含まれる場合においても，この手法を拡張することで制御器の設計が可能であることを示した．一方，異常動作の回避においては，第 5 章で定式化した問題P_{EC}^{ce}を拡張し，不可制御な動作を含まない要素から制御器を合成する手法を開発した．

[3]については，第 7 章で検討を行った．化学プロセスの特徴は，非線形性の高い物理量に加え，プロセス特性・環境変化・運用変更等にあり，これらをバッチプロセスにとっての不確実性とすると，予想されるさまざまな不確実性のもとで，状態・動作に係る因果関係の変化を推定しながら，将来，発生することが予想される異常にも対応することが求められる．本研究では，不確実性を含むバッチプロセスに対し，不確実性の影響を予測しながら異常状態に陥ることを回避するための方法として，ダイナミックベイジアンネットワークを用いた手法の適用可能性について検討した．そして，異常状態を回避する確率が最も高くなるような操作量をネットワークから求める方法を開発した．この方法を不確実性を伴う生産プロセスに適用し，数値シミュレーションの結果，将来予測をしない場合と比較して異常を回避できる確率が上がることを示した．

以上，本研究においては既往の研究の問題の解決を図ることを目的とした[1]から[3]の検討を通じ，バッチプロセスにおける異常対応の合理的な方法論を確立した．[1]においては，バッチプロセスの動作を離散事象システムとしてとらえ，それをペトリネットで記述したモデルを異常対応モデルとし，このモデルのもと異常対応の各問題の解法を開発することで，これを達成した．[2]においては，異常対応モデルに不可観測なプレースおよび不可制御なトランジションを新たに導入し，それらの要素を考慮した制御器設

計手法を確立することで，これを達成した．[3]においては，プロセスのもつ不確実性の影響を確率過程を使って表現し，プロセスの将来動作が表現できる動的な確率モデルであるダイナミックベイジアンネットワークを用いた将来予測に基づく異常回避方法を開発することで，これを達成した．

本研究成果により，対象とするプロセスの規模や複雑さが増した場合においても，その動作をペトリネットでモデル化することで，開発した手法により異常対応の論理的な解決が可能となった．また，観測できない状態および制御できない動作を含むプロセスや不確実性を含むプロセスに対しても異常の回避を実現する制御器の設計が可能となり，既往の異常の回避の研究と比較し，より多くの種類の異常を扱えるようになった．この結果として，安全性および品質維持をより高いレベルで実現するバッチプロセスの運転が可能となると考えられる．

8.2. 今後の展望

本研究で開発た手法はいくつかの課題が残されており，これらの解決を図ることで，バッチプロセスにおける異常対応に対し，更に有効な手段となることが期待される．最後に，本研究における今後の展望を述べる．

(1) 本研究では未経験の異常が検出されるたび，それを表す動作や状態をモデルに組み込むことで，異常対応モデルを更新してく手法を提案した．一方で，本手法では新たに組み込まれる異常のモデル構造は一意に決まらず，扱うプロセスや発生した異常の知識が必要であることを示した．異常のモデル構造は，それを回避するための制御器の設計に大きな影響を及ぼすため，異常の正確なモデル化は重要である．未経験の異常の正確なモデル化手法が確立すれば，本研究で開発した異常の診断とその回避に係る手法は更に有効な技術となることが期待される．

(2) 動的モデルを用いた異常対応は，診断できる異常や設計される制御器はモデルの粒度に大きく依存する．様々な粒度をもつモデルを構築するため，階層構造をもつ動的モデルを用い，上位階層にて大まかな動作を，下位階層にて細かな動作を記述し，各階層で異常の診断とその回避のための制御器を設計するという方法がある．ペトリネットモデルは階層性に優れており階層的な異常対応モデルを構築することができるが，階層間での異常情報，制御則の情報や同階層における異なるネットモジュール同士の情報のやり取りの仕方等，検討すべき課題は残されて

いる．階層構造をもつペトリネットモデルに基づく異常対応の方法論が確立されれば，より実用的な異常対応が可能となることが期待される．

(3) 第 7 章において，本研究では不確実性を含むバッチプロセスの異常の回避に関し，確率モデルであるベイジアンネットワークによる手法を提案した．一方，本手法と，第 2 章から第 6 章で検討したペトリネットモデルに基づく手法との融合が図れれば，更に優れた異常対応が実現できることが期待される．

付録1. ペトリネットに係る用語

ペトリネットは，プレース，トランジションの 2 種類のノードを持つ 2 部有向グラフである．ここで，アークはプレースからトランジションに接続するもの，あるいはトランジションからプレースに接続するもののいずれかであり，同じ種類のノードを接続することはできない．図的な表現では図 8-1 に示すように，プレースは○，トランジションは□で描かれる．アークには非負整数の重みがつけられる場合がある．重みはアークの本数で表すか，アークに重みを併記して表す．本研究においては，アークに重みを併記する方法を採用し，重みがつけられないアーク，すなわち非負整数が併記されていないアークは重みが 1 であるとみなす．さらに，各プレースの上には，非負整数個のトークンが割り当てられる．トークンは●で描かれる．各プレース上のトークンの配置をマーキングという．m 個のプレースからなるペトリネット全体のマーキングは，m 次元ベクトル$\boldsymbol{M}$で表される．$\boldsymbol{M}$のp_i番目の成分は$M(i)$で表され，プレースp_iに置かれるトークンの数を表す．

形式的には，ペトリネットNは$N = (P, T, F, W, \boldsymbol{M}_0)$ で表される．ここで，

$P = \{p_1, p_2, \cdots, p_m\}$はプレースの集合．

$T = \{t_1, t_2, \cdots, t_n\}$はトランジションの集合．

$F \subseteq (P \times T) \cup (T \times P)$は，アークの集合．

$W : F \rightarrow \{1,2,\cdots\}$はアークの重みを定める関数．

$\boldsymbol{M}_0 : P \rightarrow \{0,1,2,\cdots\}$は初期マーキング

となる．

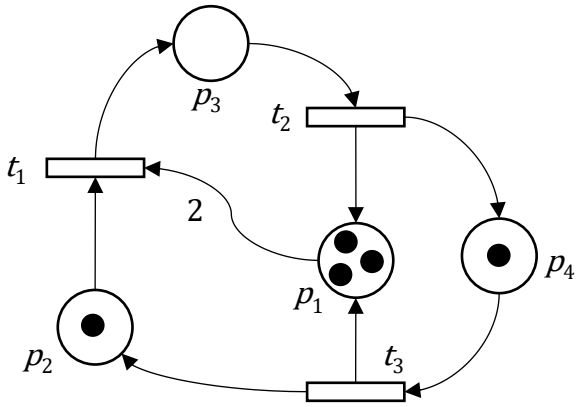


図 8-1　ペトリネットモデル

付録2. 半言語

アルファベットΣ上の同型な半順序多重集合の族を半語pwといい，$[U, R, \pi]$で表す．ただし，Uは有限集合，Rは Uにおける半順序関係，$\pi : U \to \Sigma$はラベル付け関数である．そして，Σ上の半語全体の部分集合を半言語PLという．半語は，直感的には文字の半順序付けられた多重集合である．図 8-2 に$\Sigma = (a, b, c, d, e, f)$からなる半言語の例を示す．

また，ある半語Yが表す動作の途中までの動作を表す半語を接頭半語と言い，Y^{PREF}とする．一方，半語Y_1，Y_2に関し，Y_1の適当な要素の間を順序付けることでY_2が得られる場合，Y_1はY_2により並行性に富むと言える．図 8-2 において，pw_1はpw_2の接頭半語であり，pw_4のトランジション cと fに順序付けをすることでpw_3が得られるため，pw_4はpw_3に比べて並行性に富むことが分かる．

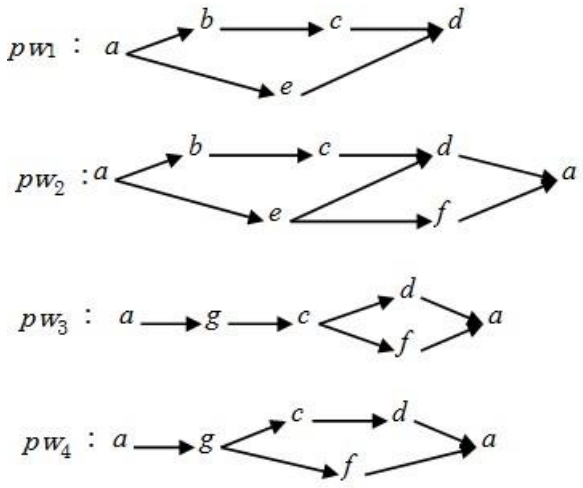


図 8-2　半言語の例

参考文献

[1]. 石油化学工業協会, “石油化学工業の現状,” *石油化学工業協会*(2013)

[2]. 三菱化学株式会社, “鹿島事業所における基礎石油化学工業の構造改革について,” *三菱化学株式会社ニュースリリース*(2012)

[3]. 住友化学株式会社, “千葉工場における石油化学事業の再構築について,” *住友化学株式会社ニュースリリース*(2013)

[4]. 内閣府, “科学技術基本計画,” http://www8.cao.go.jp/cstp/kihonkeikaku/5honbun.pdf(2016)

[5]. 内閣府, “超スマート社会の姿と超スマート社会に向けた取り組みについて,” http://www8.cao.go.jp/cstp/tyousakai/kiban/3kai/siryo1.pdf(2015)

[6]. Nimmo,I, “Adequately address abnormal situation operations,” *Chemical Engineering Progress* pp.36-45(1995)

[7]. V. Venkatasubramanian, R. Rengaswamy , K .Yin, and Surya N. Kavuri, “A review of process fault detection and diagnosis,” *Computers and Chemical Engineering*, Vol.27, pp.293-311(2003)

[8]. J.V.Kresta,J.F.Macgregor and Thomas E. Marlin, “Multivariate statistical monitoring of process operating performance,” *Can.J.Eng*,Vol.69,pp.35-47(1991)

[9]. P.Nomikos, J.F.MacGregor “Monitoring Batch Processes Using Multiway Principal Component Analysis,” *AIChE J*, Vol.40, pp,1361-1375(1994)

[10]. Iri, M., Aoki, K., O’Shima, E., and Matsuyama, H, “An algorithm for diagnosis of system failures in the chemical process,” *Computers and Chemical Engineering*,Vol.3, pp.489-493(1979)

[11]. Shiozaki, J., Matsuyama, H., O’Shima, E., and Iri, M, “An improved algorithm for diagnosis of system failures in the chemical process,” *Computers and Chemical Engineering*,Vol.9,pp.285-293(1985)

[12]. Shiozaki, J., Matsuyama, H., Tano, K., and O’Shima, E, “Fault diagnosis of chemical processes by the use of signed directed graphs Extension to five-range patterns of abnormality,” *International Chemical Engineering*, Vol.25, pp.651-659 (1985)

[13]. Ramadge, P. J. and W. M. Wonham, “The Control of Discrete Event Systems,” *Proc. IEEE*, Vol.77, 81 (1989)

[14]. Sanchez, A. “Formal Specification and Synthesis of Procedural Controllers for Process Systems,” *Lecture Notes in Control and Information Sciences*, Springer-verlag, London, U.K. (1996)

[15]. Moody, J. O., Yamalidou, K., Lemmon, M. D., and Antsaklis, P. J., “Feedback control of Petri nets based on place invariants,” *In Proceedings of the 33rd IEEE Conference on Decision and Control*, Vol.3, pp.3104-3109, Lake Buena Vista, FL. (1994)

[16]. 橋爪進, 蓬田 荘洋, 上田邦良, 矢嶌智之, 小野木克明, 西村義行, “条件/事象ネットに基づくシーケンス制御系のモジュラ設計,” *化学工学論文集*, Vol.26, pp.443-449 (2000)

[17]. K. Onogi, S. Hashizume, and Y. Nishimura, “Design of Sequential Control Systems Based on Condition/Event Nets,” *Proceedings of PSE Asia 2000*, Kyoto, Japan, p.261 (2000)

[18]. S. Hashizume, T. Suzuki, K. Onogi, and Y. Nishimura, “A construction problem of condition/event-nets and its solvability,” *Trans. SICE*, Vol.28, pp.632-639 (1992)

[19]. S. Hashizume, Y. Mitsuyama, Y. Matsutani, K. Onogi, and Y. Nishimura, “Construction of Petri nets from a given partial language,” *IEICE Trans. Fundamentals*, Vol.E79-A, pp.2192-2195 (1996)

[20]. SP88 Committee, *Batch Control Part1: Models and Terminology*, ISA-The Instrumentation, System, and Automation Society, North Carolina, U.S.A. (1995)

[21]. 黒田千秋, ”システム解析,” *朝倉書店* (2014)

[22]. 化学工学会編, “化学工学の進歩 38 バッチプロセス工学,” *槇書店* (2004)

[23]. 村田忠夫, ”ペトリネットの解析と応用,” *近代科学社* (1992)

[24]. J.L.Peterson, ”ペトリネット入門,” *共立出版* (1984)

[25]. J.Grabowski, ”On partial language,” *Fundamenta Informaticae*, Vol.4(2), pp.427-498(1981)

[26]. Martinez, J. and M. Silva, “A Simple and Fast Algorithm to Obtain All Invariants of a Generalized Petri Net,” *Informatik-Fachberichete* 52, C. Girault and W. Reisig, ed., pp.301-310, Springer-Verlag, Berlin, Germany (1982)

[27]. 橋爪進, 松谷豊, 小野木克明, 西村義行, ”与えられた半言語に適合するアトム全体を求めるアルゴリズム,” *計測自動制御学会論文集*, Vol.32,pp.1560-1565(1996)

[28]. 松谷豊, 橋爪進, 小野木克明, 西村義行, ”補助事象なし条件／事象ネット構成問題の解集合を求める一方法,” *計測自動制御学会論文集*, Vol.32,pp.1454-1460(1996)

[29]. Christos G. Cassandras and Stephane Lafortune, “Introduction to Discrete Event Systems,” *Springer*(2010)

[30]. J.Desel, W.Reisig,”The synthesis problem of Petri nets,” *Acta Informatica*, Vol.33, pp.33-297(1996)

[31]. E.Badouel, L.Bernardinello and P.Darondeau, “The synthesis problem for elementary net systems is NP-complete,” *Theoretical Computer Science* Vol.186, pp.107-134(1997)

[32]. P.Graubmann, “The construction of EN systems from a given trace behavior,” *Lect. Notes in Computer Science*, Vol.340, pp.133-153(1988)

[33]. M.Sampath,R.Sengupta,S.Lafortune,K.Sinnamohidden,D.Teneketzis, ”Diagnosability of discrete event systems,” *IEEE Transactions Automatic Control*, Vol.40, pp.1555–1575(1995)

[34]. M.Sampath,R.Sengupta,S.Lafortune,K.Sinnamohidden,D.Teneketzis, ”Failure diagnosis using discrete event systems,” *European Journal of Control*, Vol.18, pp.82–93(1995)

[35]. Mosterman, P.J, “Dyagnosis of Physical Systems with Hybrid Models Using Parametrized Causality,” *Lect Note in Computer Science2034*, pp.447-458, Springer, Berlin, Germany(2001)

[36]. J.Ashley and L.E.Hollyway, “Qualitative Diagnosis of Condition System,” *Discrete Event Dynamic Systems: Theory and Applications*, Vol.14, pp.395-412(2004)

[37]. S.Bhattacharyya, R.Kumar and Z.Hung, “A Discrete Event Systems Approach to Network Fault Management: Detection and Diagnosis of Faults,” *Asian J.Control*, Vol.13, pp.471-471(2011)

[38]. M.Daigle, X.Koutsoukos and G.Biswas, “A Qualitative Event-Based Approach to Continuous Systems Diagnosis,” *IEEE trans. Control System Technology*, Vol.17, pp.780-793(2009)

[39]. M.Roth, S.Schneider, J-J.Lesage and L.Lits, “Fault Detection and Isolation in Manufacturing Systems with an Identified Discrete Event Model,” *International System Science*, Vol.43,pp.1826-1841(2012)

[40]. M.Dotoli, M.Pia Fanti, A.M.Magniti and W.Ukovich, “Identification of the Unobservable Behaviour of Industrial Automation Systems by Petri Nets,” *Control Engineering Practice*, Vol.19, pp.958-966(2011)

[41]. M.P.Faniti, M.Doto¡o and A.M.Managini, “Fault Detection of Discrete Event Systems Using Petri Nets and Integer Linear Programing,” *Proc. 17th IFAC World Congress*, pp.6554-6559, Seoul, Korea(2008)

[42]. T.Ito, S.Hashizume, T.Yajima and K.Onogi, “Fault Detection for Batch Processes Based on Petri Net Models,” *Prc. 3rd International Symposium on Design*, *Operation and Control of Chemical Processes*, pp.483-488, Seoul, Korea(2005)

[43]. T.Ito, S.Hashizume, T.Yajima and K.Onogi, “Building of Observers to Detect Fault for Batch Processees Based on Discrete Event Systems Approaches,” *Journal of Chemical Engineering of Japan*, Vol.39,pp.1069-1077(2006)

[44]. 熊谷貞俊, 薦田憲久, “ペトリネットによる離散事象システム論,” *コロナ社*(1995)

[45]. J.O.Moody, and P.J.Antasklis, “ Supervisory Control of Discrete Event Systems Using Petri Nets,” *Kluwer Academic Publisher*(1998)

[46]. 橋爪進, 小野木克明, 西村義行, “条件/事象ネットの冗長性について,” *計測自動制御学会論文集*, Vol.28, pp.401-408 (1992)

[47]. 橋爪進, 鈴木隆, 小野木克明, 西村義行, ”条件/事象ネットの補助事象なし設計問題とその解法,” *計測自動制御学会論文集*, Vol.28, pp.1248-1256(1992)

[48]. 橋爪進, 矢嶌智之, 小野木克明, “与えられた動作を行う条件/事象ネットの最小実現,” *電子情報通信学会第 22 回 回路とシステム軽井沢ワークショップ論文集*, pp.528-533 (2009)

[49]. T.Ito, S.Hashizume, T.Yajima and K.Onogi, “Integration between Scheduling and Design of Batch Systems Based on Petri Net Models,” *IEICE Trans. Fundamentals*, Vol.E88-A, pp.2989-2998 (2005)

[50]. 木村陽一, 赤穂昭太郎, 麻生英樹, “ベイジアンネット学習の知能システムへの応用,” *計測と制御*, Vol.38, pp.468－473 (1999)

[51]. 岸本圭史, 小栗 宏次, “直前の一定期間の運転行動を考慮した AR-HMM に基づく停止行動予測,” *電子情報通信学会論文誌. A, 基礎・境界*, Vol.J92-A, pp.624－632 (2009)

[52]. 西野正彬, 中村幸博, 八木貴史, 武藤伸洋, 阿部匡伸, “ダイナミックベイジアンネットワークを用いたあいまいな表現を含むスケジューラデータと GPS データからの行動予測,” *電子情報通信学会技術研究報告ライフインテリジェンスとオフィス情報システム*, Vol.110, pp.29－34 (2010)

本論文に関する著者の研究業績

学術論文

1. 橋爪進, 上野宏晃, 橋爪悟, 矢嶌智之, 小野木克明, ”与えられた動作を行う条件/事象ネットの最小実現”, *電子情報通信学会論文誌 A*,Vol.J94-A, pp.912-922(2011)
2. S.Hashizume, S.Hashizume, T.Yajima, K.Oongi, “Control of Discrete Event Systems Using Condition/Event Nets and Partial Languages”, *SICE Journal of Control, Measurement, and System Integration*, Vol.5, pp.139-146 (2012)
3. 矢嶌智之, 添田幸宏, 橋爪悟, 橋爪進, 小野木克明, “不確実性を含む動的システムのモデリングと意思決定支援”, *化学工学論文集*, Vol.41, pp.374-380(2015)
4. S. Hashizume, S.Hashizume, T.Yajima, K.Oongi, “Construction of Batch Process System Models for Fault Analysis”, *Journal of Chemical Engineering of Japan*, Vol.49, pp.689-697 (2016)
5. S.Hashizume, S.Hashizume, T.Yajima, K.Oongi, “Fault Avoidance Control of Batch Processes with Uncontrollable Events and Unobservable Statuses”, *Journal of Chemical Engineering of Japan*, Vol.49, pp.698-706 (2016)

国際会議

1. S.Hashizume, S.Hashizume, T.Yajima, K.Onogi, “Condition/Event Net Synthesis for Discrete Event Control Using Partial Language Specifications” *Proc. of SICE Annual Conference 2013*, Nagoya, Japan, Sep.14 -17 (2013)

謝辞

本研究に取り組む機会を与えていただくとともに，さまざまなご指導，ご鞭撻を賜りました愛知工業大学情報科学部情報科学科 教授 小野木克明先生に謹んで感謝の意を表します．先生には，研究に対する姿勢だけでなく公私にわたり多くのことを学ばせていただきました．また，本研究を行うにあたり，数々のご助力をいただきました名古屋大学大学院工学研究科化学・生物工学専攻 教授 田川智彦先生に深く感謝申し上げます．

本研究をまとめる上で，貴重なご助言を賜りました名古屋大学大学院工学研究科化学・生物工学専攻 教授 堀添浩俊先生，名古屋大学大学院工学研究科機械理工学専攻 教授 鈴木達也先生，名古屋大学大学院工学研究科化学・生物工学専攻 講師 橋爪進先生に厚く御礼申し上げます．

名古屋大学大学院工学研究科化学・生物工学専攻 助教 矢嶌智之先生には，常日頃から研究に係るあらゆる事柄に対して親身にご指導を賜りましたことを心から感謝いたします．

本研究を共に遂行した当時博士課程前期課程 上野宏晃氏（現（株）ノリタケカンパニーリミテッド），当時博士課程前期課程 山川達也氏（現（株）トヨタコミュニケーションシステム），当時博士課程前期課程 今泉ゆうか氏（現トヨタ紡織（株）），当時博士課程前期課程 添田幸宏氏（現日本特殊陶業（株））に心から感謝いたします．

最後に，学生生活を支え，応援してくれた家族に感謝します．