リスポンシブシステムにおける
動的な性能の解析方法について

今井博英

新潟大学大学院自然科学研究科博士後期課程

情報理工学専攻

要旨

本論文では，リスポンシブシステム中にプロセッサ故障が発生した場合の，過渡状態におけるシステムの動的な性能の解析方法を示す．

ここでは，複数の処理要素のグループを持ち，各処理要素のグループは複数のプロセッサと一つの待ち行列からなるリスポンシブシステムを解析の対象とする．また，システムで処理されるタスクは，任意の発生間隔と処理時間及び制限時間を持つものとし，それらが従う確率分布に基づいて複数のタスククラスに分類されている．

まず，このようなシステム中で処理されるタスク数の変化を連続時間型マルコフ連鎖でモデル化した系内数モデルを定義し，続いて処理される代表的なタスクの処理の過程を，同じく連続時間型マルコフ連鎖でモデル化したタグ付きタスクモデルを定義する．次に，これらの２つのモデルを用いた過渡状態における性能の解析方法を示す．最初に系内数モデルを用いてタグ付きタスクモデルの初期分布を求める．次にこの分布を用いてタグ付きタスクモデルの定常分布を求めることによって，故障前後の過渡状態におけるタスクの棄却率を求めて性能を解析する．またタグ付きタスクモデルの定常分布を短時間で求めるために，極限状態への到達確率を表す吸収確率を用いた解析方法，並びに状態を集約しながら吸収確率を短時間で求める方法を示す．

最後に，一つの処理要素のグループを持ちタスククラスに優先度が定義されているシステム，及び複数の処理要素のグループを持つ分散リスポンシブシステムの解析例を示し，本方法が過渡状態の解析に有効であることを示す．

本論文で示す解析方法を用いることによって，リスポンシブシステムの設計段階において，システムの動的な性能を容易に評価することができ，信頼度の高いシステムを実現することが可能となる．

目 次

1. 序論 **1**

1.1 研究の目的と背景 1

1.2 従来の解析方法 2

1.3 論文の構成 3

2. システムのモデル **4**

2.1 概要 4

2.2 リスポンシブシステム 4

2.3 連続時間型マルコフ連鎖 6

2.4 系内数モデル 7

2.5 タグ付きタスクモデル 7

2.6 モデルの変換 8

2.6.1 系内数モデルからタグ付きタスクモデルへの変換 8

2.6.2 故障前のモデルから故障後のモデルへの変換 9

2.7 まとめ 10

3. 解析方法 **11**

3.1 概要 11

3.2 吸収確率 11

3.3 状態の分類 12

3.4 集約に基づく吸収確率行列の導出 13

3.5 過渡状態の解析方法 16

3.5.1 故障前に発生したタスク 16

3.5.2 故障後に発生したタスク 18

3.6 吸収確率を求めるための計算量 19
3.7 まとめ 21

4. 解析例 22
4.1 概要 22
4.2 一個の処理要素のグループからなるシステムの解析 22
4.2.1 システムの仮定 22
4.2.2 系内数モデル 24
4.2.3 タグ付きタスクモデル 25
4.2.4 変換行列 28
4.2.5 解析結果 29
4.3 複数の処理要素のグループからなるシステムの解析 35
4.3.1 システムの仮定 35
4.3.2 系内数モデル 36
4.3.3 タグ付きタスクモデル 37
4.3.4 変換行列 40
4.3.5 解析結果 43
4.4 まとめ 49

5. 結論 50

6. 謝辞 51

7. 参考文献 52

8. 著者の論文 55

9. 研究会報告等 56

10. 口頭発表等 57

付 録 A QN のタグ付きタスクモデルの同値類分割方法 58

1. 序論

1.1 研究の目的と背景

近年，コンピュータ技術の発達によりさまざまな用途にコンピュータが使用されている．このようなコンピュータシステムの中で，処理されるタスクの処理時間に制限が課せられているものはリアルタイムシステムと呼ばれている [1][2]．リアルタイムシステムのうちで，制限時間以内にタスクが終了しない場合に破壊的な影響が及ぶ可能性のある用途に用いられるものをハードリアルタイムシステムという．一方，制限時間以内に処理を終了しない場合であっても，システムの処理性能や操作性が低下するものの破壊的な事態にはいたらない用途に用いられるものをソフトリアルタイムシステムと呼んでいる．このようなリアルタイムシステムの例として，プラント制御システムや航空管制システムなどが挙げられる [3]．

ハードリアルタイムシステムにおいては，システム中の故障による影響の大きさを考慮して，システム中に故障が発生した場合であっても処理を継続することが出来るように，フォールトトレラントシステムとして構成されることが多い．このようなリアルタイムシステムとフォールトトレラントシステムの両方の性質を併せ持つシステムはリスポンシブシステムと呼ばれている [4][5]．リスポンシブシステムを実現する場合には，システム中に故障が発生した場合にも制限時間以内にタスクの処理が終了可能であることを確認するために，あらかじめシステムの性能を解析し評価することが必要である．

また，近年のネットワーク技術の発達により，コンピュータネットワークを利用した種々のマルチメディア通信サービスが実現されつつある [6][7]．このようなシステムは，大量の情報を効率よく処理するために分散型リアルタイムシステムとして実現されることが多い．また，故障が発生した場合にも処理を中断する事がないように，フォールトトレラント性も考慮されている．このようなシステムを実現する場合にも，先に述べたシステムと同様に，システム中に故障が発生した場合あるいは短時間に大量の要求が発生した場合等の過渡状態においても，期

待された性能が得られることを確認するために，十分にシステムの性能を評価し解析する必要がある．

本研究は，リスポンシブシステム中に故障が発生した場合のシステムの動的な処理性能の変化を評価することを目的とし，過渡状態におけるタスクの棄却率を求める方法を示す．このために，システム中のタスクの数の変化をモデル化した系内数モデルと，システムで処理される代表的なタスクの処理の過程をモデル化したタグ付きタスクモデルを定義し，これらを用いて解析を行う方法を提案する．また，解析に要する時間を短縮するために，タグ付きタスクモデルの定常状態確率の計算に吸収確率を用い，さらに吸収確率を状態の集約を行いながら求めて行く方法を示す．

1.2 従来の解析方法

リアルタイムシステムの解析方法として，今までに，シミュレーションによる方法の他に，ペトリネット [8][9][10] や待ち行列モデル [11] を用いた方法が提案されている．また，フォールトトレラントシステムの解析方法として，マルコフリウォードモデル [13] [14] や拡張マルコフリウォードモデル [15]，待ち行列理論 [12] を用いた方法等が提案されている．しかし，これらの方法はいずれも定常状態におけるシステムの性能，もしくは故障を含む定められた期間内の処理能力を解析するものであり，システムに故障が発生した場合の過渡的な性能の解析はシミュレーションに頼っているのが現状である．しかし，この方法では，精度の高い解析を行うために多くの回数のシミュレーションを必要とするため，経済的及び時間的に大きな負担を強いられることは避けられない．

過渡状態における解析を効率よく行うためのシステムのモデル化の一つの方法として，タグ付きタスクモデルに基づくシステムの解析方法が提案されている [10][18]．これは，システム中のある一つのタスクに注目し，そのタスクの処理が終了するまでの過程をモデル化することによって，システムの性能を解析する方法である．しかし，このようなモデル化を行った場合には，システムが大規模化し，あるいは処理内容が複雑化するのに伴って，状態数が急激に増大するために解析が困難になるという問題がある．このような大規模なマルコフ連鎖を解析する方法とし

て，文献 [18] では，初期状態において状態確率が小さい状態を切り捨て，残りの状態のみを用いて計算を行う方法が提案されている．しかし，モデルの構造的な特徴が考慮されていない等の問題点がある．また，文献 [19][20] では，既約なマルコフ連鎖の定常状態の確率を集約と分割を繰り返すことによって求める方法が示されており，文献 [21] ではマルコフ連鎖の状態を推移率の値が異なる大小 2 種類の集合に分割し，推移率の大きい状態同士を集約することによって解析する方法が提案されている．しかし，著者等が提案したタグ付きタスクモデルは既約ではなく，また推移率の値にも大きな差がないためこれらの方法を適用することができないという問題がある．

1.3 論文の構成

本論文は以下の章からなる．

第 2 章では，本論文で対象とするリスポンシブシステムを示し，その解析に用いる系内数モデルとタグ付きタスクモデルを定義する．また，系内数モデルからタグ付きタスクモデルへの変換方法を示す．

第 3 章では，系内数モデルとタグ付きタスクモデルを用いた，システムに故障が発生した場合の過渡状態の解析方法を示す．ここでは，タグ付きタスクモデルの状態を到達可能性に基づく同値類に分割し，同値類を集約しながら計算することによって，タスクの棄却率を短時間に求める方法を示す．また，集約を行うことによる演算量の変化について考察し，本方法を適用するための基準を示す．

第 4 章では，システムの解析例を示す．はじめに，4.2 節では，単一の処理要素のグループを持つシステム，即ちシステムが 1 個の待ち行列と複数のプロセッサから構成され，処理されるタスク間に優先順位が課されているシステムの解析例を示す．続いて，4.3 節では，複数の処理要素のグループを持つシステム，即ちネットワークで接続された複数の処理要素と，グループ毎に 1 個の待ち行列を持つ分散型リスポンシブシステムの解析例を示す．

最後に，第 5 章では結論を述べる．

2. システムのモデル

2.1 概要

本章では，リスポンシブシステム及びそのモデルについて述べる．過渡状態の解析を行うために，システムの系内数の変化を連続時間型マルコフ連鎖でモデル化した系内数モデルと，システムで処理される代表的なタスクの処理を，同じく連続時間型マルコフ連鎖でモデル化したタグ付きタスクモデルを用いる．

2.2 節では，本論文で解析するリスポンシブシステムの諸定義及び仮定を示し，2.3 節では，連続時間型マルコフ連鎖について説明する．2.4 節で系内数モデルを定義し，2.5 節ではタグ付きタスクモデルを定義する．さらに，2.6 節では，モデルの変換方法をを示す．

2.2 リスポンシブシステム

リスポンシブシステムは，システム中のプロセッサに故障が発生した場合にも正しく処理を続行できるようにするために，複数の処理要素のグループを持ち，各処理要素のグループは一個の待ち行列と複数のプロセッサを有している．本論文で扱うリスポンシブシステムについて，次のように仮定する．

仮定 2.1 (リスポンシブシステム)

- システムは $\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_n, \ldots, \mathcal{R}_{N_r}$ の N_r 個の処理要素のグループからなる．
- 各処理要素のグループ $\mathcal{R}_n$ は，K_n 個のプロセッサと，長さ $L_n - K_n$ の一個の待ち行列を持つ． ∎

以後，L_n を処理要素のグループ $\mathcal{R}_n$ の容量と呼び，各処理要素のグループ中に存在している，各クラス毎のタスク数を系内数と呼ぶ．

リスポンシブシステムは，故障が発生した場合でも速やかにシステムの再構成を行い処理を継続しなければならない．本論文では，システム中に発生し得る故障及び，故障が検出された後のシステムの再構成について次の仮定をおく．

仮定 2.2 (故障とシステムの再構成)

(1) 故障はシステムが定常状態にあるときに発生する．
(2) システム中の故障はプロセッサにのみ発生する． ∎

本論文では処理されるタスクが非周期的に発生し，処理時間が任意の値を取り得るシステムについて検討を行う．またタスクの制限時間として，タスクが発生してから処理がすべて終了するまでの制限時間であるデッドラインを用いる．このようなタスクについて次のような仮定をおく．

仮定 2.3 (タスク)

(1) タスクは $1, 2, \ldots, j, \ldots, N_c$ までの N_c 個のクラスに分けられる．
(2) クラス j に属するタスクの，
- 発生間隔はパラメータ λ_j の指数分布に従う．
- 処理要素のグループ $\mathcal{R}_n$ での処理時間はパラメータ $\mu_{n,j}$ の指数分布に従う．
- デッドラインはパラメータ ν_j の指数分布に従う．

(3) デッドラインを超えたタスクは直ちに捨てられる． ∎

システムの状態を，各処理要素のグループ中に存在するクラス j のタスクの数を用いて以下のように定義する．

定義 2.1 (システムの状態)
システムの状態を次式で定義する．

$$(i_{1,1}, i_{1,2}, \ldots, i_{1,N_c}, i_{2,1}, \ldots, i_{n,j}, \ldots, i_{N_r,N_c})$$

ここで，$i_{n,j}$ は処理要素のグループ $\mathcal{R}_n$ で処理されているクラス j のタスクの系内数を表す．

システムの状態集合を $\mathcal{S}^{(N)}$ で表す． ∎

2.3 連続時間型マルコフ連鎖

本論文では，系内数の変化の過程，及びタスクが処理される過程を連続時間型マルコフ連鎖 [24][25] を用いてモデル化する．

連続時間型マルコフ連鎖の確率過程 $\{Z(t)\}_{t\geq 0}$ を考える．この確率過程の状態集合を，$\mathcal{S}=\{1,2,\cdots,N\}$ とし，$Z(t)(\in \mathcal{S})$ は時刻 t の状態を表す．微小時間 Δ 以内に状態 i から $j(\neq i)$ へ推移する確率が $g_{i,j}\Delta$ で表せる場合，$g_{i,j}$ を状態 i から j への推移率と呼び，対角要素が

$$g_{i,i}=-\sum_{j=1,j\neq i}^{n} g_{i,j} \tag{2.1}$$

である $N\times N$ の行列 $\boldsymbol{G}=[g_{i,j}]$ を推移率行列と呼ぶ．また，時刻 t における確率過程の状態が i である確率を $p_i(t)$ で表し，これを状態確率と呼び，行ベクトル $\boldsymbol{p}(t)=(p_1(t),p_2(t),\cdots,p_N(t))$ を状態確率ベクトルと呼ぶ．ある初期状態確率ベクトル $\boldsymbol{p}(0)$ を与えることにより，$\boldsymbol{p}(t)$ は次式によって得られる．

$$\frac{d}{dt}\boldsymbol{p}(t)=\boldsymbol{p}(t)\boldsymbol{G} \tag{2.2}$$

また，定常状態 $(t\to\infty)$ での状態確率ベクトルを $\boldsymbol{p}$ で表す．

連続時間型マルコフ連鎖の状態の推移にのみ着目した確率過程を，隠れマルコフ連鎖と呼ぶ．この確率過程は離散時間型であり，状態集合は基となる連続時間型マルコフ連鎖に等しい．この隠れマルコフ連鎖について，状態が i であったときに，次の状態推移で j に推移する確率を $h_{i,j}$ で表し，これを推移確率と呼ぶ．推移確率は基となる連続時間型マルコフ連鎖の推移率を用いて次式により得られる [22]．

(1) $g_{i,i}\neq 0$ の場合

$$\begin{aligned} h_{i,j} &= -g_{i,j}/g_{i,i}\,(i\neq j)\\ h_{i,i} &= 0 \end{aligned}$$

(2) $g_{i,i}=0$ の場合

$$\begin{aligned} h_{i,j} &= 0\,(i\neq j)\\ h_{i,i} &= 1 \end{aligned}$$

ここで行列 $\boldsymbol{H} = [h_{i,j}]$ を推移確率行列と呼ぶ．初期分布が等しい場合，隠れマルコフ連鎖の定常分布と，対応する連続時間型マルコフ連鎖の定常分布は等しくなる [22]．

2.4 系内数モデル

システムの定常状態を解析し，次節で定義するタグ付きタスクモデルの初期状態確率を求めるために，システムの系内数の変化を連続時間型マルコフ連鎖でモデル化し，これを系内数モデルという．系内数モデルの状態はシステムの状態に等しい．

時刻 t における状態 $\boldsymbol{n}(\in \mathcal{S}^{(N)})$ の状態確率，状態確率ベクトルを，それぞれ $p^{(N)}_{\boldsymbol{n}}(t)$，$\boldsymbol{p}^{(N)}(t)$ で表し，定常分布を $\boldsymbol{p}^{(N)}$ で表す．さらに，状態 $\boldsymbol{n}$ から $\boldsymbol{n}'$ への推移率を $g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'}$ で表し，推移率行列を $\boldsymbol{G}^{(N)}$ で表す．推移率は，各クラスのタスクの発生間隔，処理時間，及び制限時間の分布のパラメータ λ_j，$\mu_{n,j}$，ν_j を用いて表される．

2.5 タグ付きタスクモデル

システムで処理される代表的なタスクに注目し，そのタスクの処理が終了するまで，もしくは捨てられるまでの処理の過程を連続時間マルコフ連鎖で表したモデルをタグ付きタスクモデルという．また，注目しているタスクをタグ付きタスクと呼ぶ．

タグ付きタスクモデルを用いることにより，故障前後の過渡状態において発生したタスクが，システムに故障が発生する直前にどのような状態にあり，再構成後にどように処理が継続されるか，もしくは捨てられるかを解析することが可能となる．

タグ付きタスクモデルの状態は，タグ付きタスクが存在するシステム内の位置をシステムの状態に追加することによって表す．タグ付きタスクモデルの状態を以下のように定義する．

定義 2.2 (タグ付きタスクモデルの状態)

タグ付きタスクモデルの状態は，タグ付きタスクの処理が終了し，システムから出力したことを表す状態 C，タグ付きタスクが制限時間を満たせないかまたは待ち行列に入り切らずに捨てられたことを表す状態 F，及び次のようなタグ付きタスクがシステム中に存在することを表す状態の集合からなる．

$$(i_{1,1}, i_{1,2}, \ldots, i_{1,N_c}, i_{2,1}, \ldots, i_{n,j}, \ldots, i_{N_r,N_c}, l, r)$$

ここで，l はタグ付きタスクが滞在している処理グループを表し，r はその処理グループ内での位置を表す．∎

状態 C 並びに F は，一度この状態に推移すると以後は他の状態に推移することはない．このような状態を吸収状態と呼び，これ以外の状態を過渡的状態と呼ぶ [22]．タグ付きタスクモデルの状態集合を $\mathcal{S}^{(T)}$ で表し，吸収状態と過渡的状態の集合をそれぞれ $\mathcal{S}_A^{(T)}$，$\mathcal{S}_T^{(T)}$ で表す．

状態 $\boldsymbol{u}(\in \mathcal{S}^{(T)})$ の状態確率と状態確率ベクトルを，それぞれ $p_{\boldsymbol{u}}^{(T)}(t)$，$\boldsymbol{p}^{(T)}(t)$ で表し，定常分布を $\boldsymbol{p}^{(T)}$ で表す．さらに，状態 $\boldsymbol{u}$ から $\boldsymbol{u}'$ への推移率を $g_{\boldsymbol{u},\boldsymbol{u}'}^{(T)}$ で表し，推移率行列を $\boldsymbol{G}^{(T)}$ で表す．推移率は，各クラスのタスクの発生間隔，処理時間，制限時間分布のパラメータ λ_j，$\mu_{n,j}$，ν_j を用いて表される．また，タグ付きタスクモデルの隠れマルコフ連鎖の推移確率を $h_{\boldsymbol{u},\boldsymbol{u}'}^{(T)}$ で表し，推移確率行列を $\boldsymbol{H}^{(T)} = [h_{\boldsymbol{u},\boldsymbol{u}'}^{(T)}]$ で表す．また，推移確率行列を吸収状態と過渡的状態に関してブロック分けしたものを，それぞれ $\boldsymbol{H}_{AA}^{(T)}$，$\boldsymbol{H}_{TA}^{(T)}$，$\boldsymbol{H}_{AT}^{(T)}$，$\boldsymbol{H}_{TT}^{(T)}$ と表す．

2.6 モデルの変換

2.6.1 系内数モデルからタグ付きタスクモデルへの変換

タグ付きタスクモデルの初期分布を求めるために，系内数モデルからタグ付きタスクモデルへの変換を行う必要がある．この変換を以下のように定義する．

定義 2.3 (系内数モデルからタグ付きタスクモデルへの変換行列)
システムの状態が $\boldsymbol{n}(\in \mathcal{S}^{(N)})$ であったときにタグ付きタスクが発生した場合，タグ付きタスクモデルの状態が $\boldsymbol{u}(\in \mathcal{S}^{(T)})$ となる確率を $\gamma_{\boldsymbol{n},\boldsymbol{u}}$ で表す．また，行列

$\mathbf{\Gamma} = [\gamma_{\boldsymbol{n},\boldsymbol{u}}]_{\boldsymbol{n}\in\mathcal{S}^{(N)},\boldsymbol{u}\in\mathcal{S}^{(T)}}$ を系内数モデルからタグ付きタスクモデルへの変換行列と呼ぶ. ∎

$\mathbf{\Gamma}$ は，タスクのスケジューリング方法によって決定される.

2.6.2 故障前のモデルから故障後のモデルへの変換

本方法では，故障が発生する前のシステムのモデルと，故障によりシステムの構成が変更された後のシステムのモデルを用いて解析を行う．系内数モデルとタグ付きタスクモデルについて，それぞれ故障前のシステムのモデルの状態から，故障後のモデルへの変換行列を次のように定義する.

定義 2.4 (故障前の系内数モデルから故障後のモデルへの変換行列)
故障前の系内数モデルの状態集合を $\mathcal{S}^{(N,-)}$ で表し，故障後の系内数モデルの状態集合を $\mathcal{S}^{(N,+)}$ で表す．故障直前にシステムが状態 $\boldsymbol{n}(\in \mathcal{S}^{(N,-)})$ であったとき，システムの再構成後の状態が $\boldsymbol{n}'(\in \mathcal{S}^{(N,+)})$ である確率を $\phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'}$ で表し，これを系内数モデルの故障による状態推移確率と呼ぶ．また，行列 $\mathbf{\Phi}^{(N)} = [\phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'}]_{\boldsymbol{n}\in\mathcal{S}^{(N,-)},\boldsymbol{n}'\in\mathcal{S}^{(N,+)}}$ を故障前の系内数モデルから故障後のモデルへの変換行列と呼ぶ. ∎

定義 2.5 (故障前のタグ付きタスクモデルから故障後のモデルへの変換行列)
故障前のタグ付きタスクモデルの状態集合を $\mathcal{S}^{(T,-)}$ で表し，故障後のタグ付きタスクモデルの状態集合を $\mathcal{S}^{(T,+)}$ で表す．故障直前にシステムが状態 $\boldsymbol{u}(\in \mathcal{S}^{(T,-)})$ であったときに，故障によるシステムの再構成後に状態が $\boldsymbol{u}'(\in \mathcal{S}^{(T,+)})$ となる確率を $\phi^{(T)}_{\boldsymbol{u},\boldsymbol{u}'}$ で表し，タグ付きタスクモデルの故障による状態推移確率と呼ぶ．また，行列 $\mathbf{\Phi}^{(T)} = [\phi^{(T)}_{\boldsymbol{u},\boldsymbol{u}'}]_{\boldsymbol{u}\in\mathcal{S}^{(T,-)},\boldsymbol{u}'\in\mathcal{S}^{(T,+)}}$ を故障前のタグ付きタスクモデルから故障後のモデルへの変換行列と呼ぶ. ∎

$\mathbf{\Phi}^{(N)}$ 並びに $\mathbf{\Phi}^{(T)}$ は，システムの故障発生時の再構成方法とタスクのスケジューリングよって決定される.

2.7 まとめ

本章では，まず解析の対象とするリスポンシブシステム、及びシステム中に発生する故障に関する仮定を示した．本論文で解析するシステムは複数の処理要素のグループを持ち，各処理要素のグループは複数のプロセッサと一つの待ち行列からなっている．また，システムで処理されるタスクは，任意の発生間隔と処理時間並びに制限時間を有しており，それらの確率分布に従って複数のタスククラスに分類されている．

つぎに，このようなリスポンシブシステム中で処理されているタスク数の変化を，連続時間型マルコフ連鎖でモデル化した系内数モデルと，システムで処理される代表的なタスクの処理の過程を連続時間型マルコフ連鎖でモデル化したタグ付きタスクモデルの2つのモデルを定義した．

最後に，系内数モデルからタグ付きタスクモデルへの変換行列，及び系内数モデルとタグ付きタスクモデルの各々について，故障前のシステムを表すモデルから故障後のシステムを表すモデルへの変換行列を定義した．

3. 解析方法

3.1 概要

リスポンシブシステムにおいては，タスクに課された制限時間を満たせない場合，あるいは待ち行列が溢れた場合には，タスクがシステムに受け入れられずに捨てられてしまう．本章では系内数モデルとタグ付きタスクモデルを用いることによって，システム中に故障が発生した場合の過渡状態におけるタスク棄却率を求める方法を示す．タスク棄却率とは，タスクが最後まで処理されずに捨てられる割合を表し，タグ付きタスクモデルの定常分布から得られる．しかし，このモデルは大規模なマルコフ連鎖となるため，本方法では定常分布を短時間に計算するために吸収確率を導入する．さらに，タグ付きタスクモデルの状態を到達可能性に基づく同値類に分割し，同値類を集約しながら計算することにより，吸収確率を短時間で計算する方法を示す．

3.2 節では吸収確率を定義し，3.3 節では到達可能性とそれに基づく同値類を定義する．3.4 節で同値類を集約しながら吸収確率を計算する方法を示し，3.5 節で過渡状態の解析方法を示す．最後に 3.6 節で集約しながら吸収確率を求める場合の計算量について検討を行う．

3.2 吸収確率

過渡状態の解析においては，タグ付きタスクモデルの初期値を変化させながら，定常分布を繰り返し求めることが必要となる．本方法では，定常分布を短時間に求めるため，タグ付きタスクモデルの吸収確率を導入する．

吸収確率ならびに吸収確率行列を次のように定義する．

定義 3.1 (吸収確率，吸収確率行列)
タグ付きタスクモデルにおいて，初期状態が $\boldsymbol{u}$ であったときに，極限状態 $(t \to \infty)$

において状態が $\boldsymbol{u}'$ である確率を吸収確率と呼び，${\alpha^{(T)}}_{\boldsymbol{u}\boldsymbol{u}'}$ で表す．さらに，行列 $\boldsymbol{\alpha}^{(T)} = [{\alpha^{(T)}}_{\boldsymbol{u}\boldsymbol{u}'}]_{\boldsymbol{u},\boldsymbol{u}'\in\mathcal{S}^{(T)}}$ を吸収確率行列と呼ぶ．∎

$\boldsymbol{\alpha}^{(T)}$ を吸収状態 $\mathcal{S}_{\mathcal{A}}^{(T)}$ と過渡的状態 $\mathcal{S}_{\mathcal{T}}^{(T)}$ に関してブロック分けし，それぞれ，$\boldsymbol{\alpha}_{\mathcal{A}\mathcal{A}}^{(T)}$，$\boldsymbol{\alpha}_{\mathcal{A}\mathcal{T}}^{(T)}$，$\boldsymbol{\alpha}_{\mathcal{T}\mathcal{T}}^{(T)}$，$\boldsymbol{\alpha}_{\mathcal{T}\mathcal{A}}^{(T)}$ で表す．これらのブロックはそれぞれ次式により得られる [22]．

$$\begin{aligned}
\boldsymbol{\alpha}_{\mathcal{A}\mathcal{A}}^{(T)} &= \boldsymbol{I} && (3.1)\\
\boldsymbol{\alpha}_{\mathcal{A}\mathcal{T}}^{(T)} &= \boldsymbol{\alpha}_{\mathcal{T}\mathcal{T}}^{(T)} = 0 && (3.2)\\
\boldsymbol{\alpha}_{\mathcal{T}\mathcal{A}}^{(T)} &= \boldsymbol{H}_{\mathcal{T}\mathcal{A}}^{(T)} + \sum_{n=1}^{\infty}(\boldsymbol{H}_{\mathcal{T}\mathcal{T}}^{(T)})^n \boldsymbol{H}_{\mathcal{T}\mathcal{A}}^{(T)} \\
&= (\boldsymbol{I} - \boldsymbol{H}_{\mathcal{T}\mathcal{T}}^{(T)})^{-1}\boldsymbol{H}_{\mathcal{T}\mathcal{A}}^{(T)} && (3.3)
\end{aligned}$$

上式より，実際に値を計算する必要があるのは，過渡状態から吸収状態への吸収確率を表す $\boldsymbol{\alpha}_{\mathcal{T}\mathcal{A}}^{(T)}$ のみであることがわかる．

$\boldsymbol{\alpha}^{(T)}$ を用いて，定常分布は次式より求めることができる．

$$\boldsymbol{p}^{(T)} = \boldsymbol{p}^{(T)}(0)\boldsymbol{\alpha}^{(T)}$$

3.3 状態の分類

実際のシステムにおけるタグ付きタスクモデルは大規模な連続時間型マルコフ連鎖となるため，式 (3.3) より $\boldsymbol{\alpha}_{\mathcal{T}\mathcal{A}}^{(T)}$ を求めるのは多くの時間を要する．そのため，本論文では，タグ付きタスクモデルの状態を互いに到達可能な状態に分割し，これらを集約しながら計算することによって吸収確率を短時間で求める方法を示す．

はじめに，状態の到達可能性を以下のように定義する [24]．

定義 3.2 (状態の到達可能性)
タグ付きタスクモデルの時刻 t における状態を $Z^{(T)}(t)$ で表す．このとき，状態 $\boldsymbol{u}$ と $\boldsymbol{u}'$ について，

$$\mathrm{Prob}[Z^{(T)}(t+s) = \boldsymbol{u}' | Z^{(T)}(t) = \boldsymbol{u}] > 0$$

ならば，$\boldsymbol{u} \to \boldsymbol{u}'$ と表し，$\boldsymbol{u}$ は $\boldsymbol{u}'$ へ到達可能であるという．

つづいて，相互到達可能性について以下のように定義する．

定義 3.3 (相互到達可能性)
状態 $\boldsymbol{u}$ と $\boldsymbol{u}'$ について，$\boldsymbol{u} \to \boldsymbol{u}'$ かつ $\boldsymbol{u}' \to \boldsymbol{u}$ ならば，$\boldsymbol{u} \leftrightarrow \boldsymbol{u}'$ と表し，相互到達可能であるという．∎

$\boldsymbol{u} \leftrightarrow \boldsymbol{u}'$ ならば $\boldsymbol{u}' \leftrightarrow \boldsymbol{u}$ であり，$\boldsymbol{u} \leftrightarrow \boldsymbol{u}'$ かつ $\boldsymbol{u}' \leftrightarrow \boldsymbol{u}''$ ならば，$\boldsymbol{u} \leftrightarrow \boldsymbol{u}''$ であるから，$\leftrightarrow$ は対称的かつ推移的関係である．従って，状態をこの関係により排反な集合に分割することができる．タグ付きタスクモデルの過渡的状態の集合 $\mathcal{S}_T^{(T)}$ を関係 $\leftrightarrow$ によって分類した部分集合を $\mathcal{S}_1^{(T)}, \mathcal{S}_2^{(T)}, \ldots, \mathcal{S}_b^{(T)}, \ldots, \mathcal{S}_a^{(T)}, \ldots, \mathcal{S}_M^{(T)}$ で表し，同値類と呼ぶ．このとき同値類 $\mathcal{S}_a^{(T)}$ に含まれる状態 $\boldsymbol{u}(\in \mathcal{S}_a^{(T)})$ から，$\mathcal{S}_b^{(T)}(\neq \mathcal{S}_a^{(T)})$ に含まれる状態 $\boldsymbol{u}'(\in \mathcal{S}_b^{(T)})$ について，$\boldsymbol{u} \to \boldsymbol{u}'$ となるような状態の組が存在する場合は $a > b$ となるように番号をつける．

このように相互到達可能性に基づいて過渡的状態を分割することにより，一度異なる同値類に推移した場合は，再び元の同値類に戻ることはないため，同値類に含まれる状態確率の和は徐々に減少する．この性質に基づき，同値類中の状態確率の和が定められた閾値以下となった同値類を一つの状態に集約することによって，$\boldsymbol{\alpha}_{TA}^{(T)}$ を短時間で求めることができる．

状態確率ベクトル $\boldsymbol{p}^{(T)}(t)$ を同値類によってブロック分けし，$\boldsymbol{p}_a^{(T)}(t) = [p_{\boldsymbol{u}}^{(T)}(t)]_{\boldsymbol{u} \in \mathcal{S}_a^{(T)}}$ と表し，推移率行列 $\boldsymbol{G}^{(T)}$ と隠れマルコフ連鎖の推移確率行列 $\boldsymbol{H}^{(T)}$ についても同様に，$\boldsymbol{G}_{ab}^{(T)} = [g_{\boldsymbol{u},\boldsymbol{u}'}^{(T)}]_{\boldsymbol{u} \in \mathcal{S}_a^{(T)}, \boldsymbol{u}' \in \mathcal{S}_b^{(T)}}$，$\boldsymbol{H}_{ab}^{(T)} = [h_{\boldsymbol{u},\boldsymbol{u}'}^{(T)}]_{\boldsymbol{u} \in \mathcal{S}_a^{(T)}, \boldsymbol{u}' \in \mathcal{S}_b^{(T)}}$ と表す．

3.4 集約に基づく吸収確率行列の導出

前節で求めた同値類 $\mathcal{S}_a^{(T)}(a = 1, 2, \ldots, M)$ を集約しながら計算することによって $\boldsymbol{\alpha}_{TA}^{(T)}$ を短時間で求める方法を示す．

定義より ${\alpha^{(T)}}_{\boldsymbol{u},C}$ 及び ${\alpha^{(T)}}_{\boldsymbol{u},F}$ は，初期状態が $\boldsymbol{u}$ であったときに，極限状態において状態 C もしくは F に推移する確率を表す．このため，初期分布を状態 $\boldsymbol{u}$ のみを 1，それ以外の全ての状態確率を 0 としてタグ付きタスクモデルの定常分布を計算した場合の状態 C と F の状態確率がそれぞれ，${\alpha^{(T)}}_{\boldsymbol{u},C}$，${\alpha^{(T)}}_{\boldsymbol{u},F}$ となる．このとき，状態 $\boldsymbol{u}$ が属する同値類を $\mathcal{S}_m^{(T)}$ とすると，$\mathcal{S}_a^{(T)}(a = m+1, \cdots, M)$ に含まれる状態には推移することはないため，これらは無視することができる．また，$\mathcal{S}_a^{(T)}(a = 1, \cdots, m)$ についても，状態確率の和がある値以下になったものを，

一つの状態に集約することによって計算量を減らすことができる．

以下に，$\boldsymbol{\alpha}_{\mathcal{TA}}^{(T)}$ の導出手順を示す．なお，集約条件の閾値 ε と，計算終了条件の閾値 δ は，精度を考慮してあらかじめ決めておくものとする．

手順 3.1 (吸収確率の導出)
全ての過渡状態 $\boldsymbol{u} \in \mathcal{S}^{(T)}$ について以下の計算を繰り返す．ここで，状態 $\boldsymbol{u}$ が属する同値類を $\mathcal{S}_m^{(T)}$ とする．

(1) 計算に必要となる初期状態 $\boldsymbol{u}$ から到達可能な状態のみの集合 $\mathcal{S}$ を求め，その初期分布 $\boldsymbol{p}^{(T)}(0) = [p_{\boldsymbol{u}'}^{(T)}(0)]_{\boldsymbol{u}' \in \mathcal{S}}$，隠れマルコフ連鎖の推移確率行列 $\boldsymbol{H}^{(T)}$ を次式より求める．

$$
\begin{aligned}
\mathcal{S} &= \mathcal{S}_{\mathcal{A}}^{(T)} \bigcup_{a=1}^{m} \mathcal{S}_a^{(T)} \\
p_{\boldsymbol{u}'}^{(T)}(0) &= \begin{cases} 1 & (\boldsymbol{u}' = \boldsymbol{u}) \\ 0 & (\boldsymbol{u}' \neq \boldsymbol{u}) \end{cases} \\
\boldsymbol{H}^{(T)} &= [\boldsymbol{H}_{ab}^{(T)}]_{a,b \in \{\mathcal{A},1,2,\ldots,m\}}
\end{aligned}
$$

(2) $a \leftarrow m$，$t \leftarrow 0$

(3) $\sum_{\boldsymbol{u}' \in \mathcal{S}_a^{(T)}} p_{\boldsymbol{u}'}^{(T)}(n) < \varepsilon$ ならば，同値類 $\mathcal{S}_a^{(T)}$ を集約する．

(a) 推移確率行列を集約を行う (図 3.1)．

はじめに，同値類 $\mathcal{S}_a^{(T)}$ から，まだ集約されていない同値類 $\mathcal{S}_b^{(T)}(b = 1, 2, \ldots, a)$，ならびに吸収状態 $\mathcal{S}_{\mathcal{A}}^{(T)}$ への推移確率を再度計算する．

これは，同値類 $\mathcal{S}_a^{(T)}$ の状態確率の割合を重みとして各状態からの推移率に掛け，それらを足し合わせることにより求める．

$$
\boldsymbol{H}_{ab}^{(T)} \leftarrow \frac{\boldsymbol{p}_a^{(T)}(t)}{\sum_{\boldsymbol{u}' \in \mathcal{S}_a^{(T)}} p_{\boldsymbol{u}'}^{(T)}(t)} \boldsymbol{H}_{ab}^{(T)}, \ (b = 1, 2, \cdots, a, \mathcal{A}) \tag{3.4}
$$

つぎに，既に集約されている同値類 $\mathcal{S}_c^{(T)}(c = a+1, a+2, \ldots, m)$ から $\mathcal{S}_a^{(T)}$ への推移確率行列と，同値類 $\mathcal{S}_a^{(T)}$ 内での推移確率を表すブロック $\boldsymbol{H}_{aa}^{(T)}$ を次式を用いて再度計算する．

$$
\boldsymbol{H}_{ca}^{(T)} \leftarrow ||\boldsymbol{H}_{ca}^{(T)}||, \ (c = a, a+1, \cdots, m) \tag{3.5}
$$

ここで，$||\boldsymbol{H}_{ab}^{(T)}||$ は行列の要素を全て足し合わせることを表す．

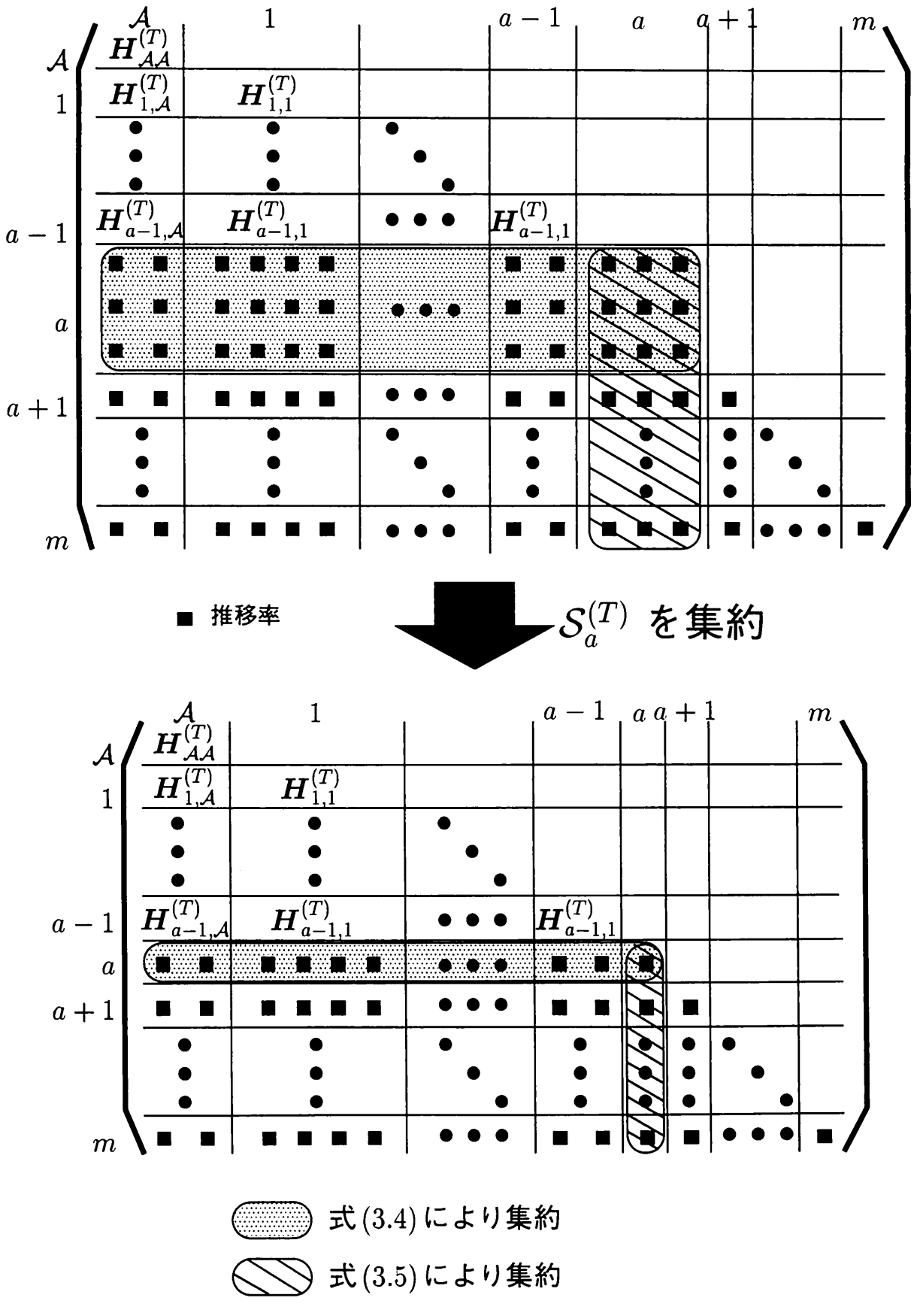


図 3.1: 推移確率行列の集約

(b) 集約後の状態確率を同値類 $S_a^{(T)}$ の状態確率の和とする．

$$\boldsymbol{p}_a^{(T)}(t) \leftarrow \sum_{\boldsymbol{a}\in S_a^{(T)}} p_{\boldsymbol{a}}^{(T)}(t) \tag{3.6}$$

$a \leftarrow a-1$ として，このステップの最初に戻る．

(4) 1 単位時間後の状態確率分布を次式で求める．

$$\boldsymbol{p}^{(T)}(t+1) \leftarrow \boldsymbol{p}^{(T)}(t)\boldsymbol{H}^{(T)}$$

(5) $t \leftarrow t+1$ とし，$1-(p_C^{(T)}(t)+p_F^{(T)}(t)) > \delta$ ならば (3) に戻る．そうでなければ，次式より ${\alpha^{(T)}}_{\boldsymbol{u},C}$，${\alpha^{(T)}}_{\boldsymbol{u},F}$ を求める．

$${\alpha^{(T)}}_{\boldsymbol{u},C} = \frac{p_C^{(T)}(t)}{p_C^{(T)}(t)+p_F^{(T)}(t)} \tag{3.7}$$

$${\alpha^{(T)}}_{\boldsymbol{u},F} = \frac{p_F^{(T)}(t)}{p_C^{(T)}(t)+p_F^{(T)}(t)} \tag{3.8}$$

3.5 過渡状態の解析方法

故障が発生した場合の過渡状態で処理されるタスクの棄却率を求める手順を示す．ここで故障発生時刻を 0 とし，時刻 t で発生したタスクの棄却率を $q_F(t)$ で表す．

本方法では，故障前と故障後のシステムについて，それぞれ系内数モデルとタグ付きタスクモデルの，合計 4 つのモデルを用いて解析を行う．これらの 4 つのモデルの時刻 t での状態確率，推移率行列，定常分布は，それぞれ $\boldsymbol{p}(t)$，$\boldsymbol{G}$，$\boldsymbol{p}$ に表 3.1 に示す記号をつけて表す．また，故障前の系内数モデルから故障前のタグ付きタスクモデルへの変換行列を $\boldsymbol{\Gamma}^{(-)}$ で表し，故障後の系内数モデルから故障後のタグ付きタスクモデルへの変換行列を $\boldsymbol{\Gamma}^{(+)}$ で表す．

3.5.1 故障前に発生したタスク

はじめに，故障前に発生したタスクの棄却率 $q_F(t)\,(t<0)$ の解析方法の概要を示す (図 3.2)．仮定 2.2 より，故障はシステムが定常状態である時に発生するため，タグ付きタスクモデルの初期分布 $\boldsymbol{p}^{(T,-)}(0)$ は故障前の系内数モデルの定常分布を

表 3.1: モデルを表す記号

モデル	記号
故障前の系内数モデル	$(N,-)$
故障前のタグ付きタスクモデル	$(T,-)$
故障前の系内数モデル	$(N,+)$
故障前のタグ付きタスクモデル	$(T,+)$

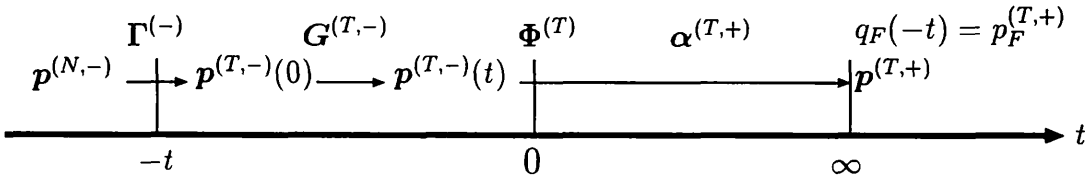


図 3.2: 故障前に発生したタスクの解析方法

用いて $\boldsymbol{p}^{(N,-)}\boldsymbol{\Gamma}^{(-)}$ と表せる．この初期分布を用いて故障 t 時間前に発生したタスクの故障発生直前の状態 $\boldsymbol{p}^{(T,-)}(t)$ を求め，これに故障によるタグ付きタスクモデルの変換行列 $\boldsymbol{\Phi}^{(T)}$ を掛けて故障直後の分布に変換し，さらに吸収確率行列 $\boldsymbol{\alpha}^{(T,+)}$ を掛けることによりタグ付きタスクモデルの定常分布 $\boldsymbol{p}^{(T,+)}$ が得られる．この分布の状態 F の確率 $p_F^{(T)}$ が，故障 t 時間前に発生したタスクの棄却率 $q_F(-t)$ となる．

以下に計算手順を示す．なお，刻み幅 Δ は精度を考慮してあらかじめ決めておくものとする．また，計算の終了時刻を t_e とする．

手順 3.2 (故障前に発生したタスクの棄却率の導出)

(1) 故障前の系内数モデルの定常分布 $\boldsymbol{p}^{(N,-)}$ を次式を解いて求める．

$$\boldsymbol{p}^{(N,-)}\boldsymbol{G}^{(N)} = \boldsymbol{0}$$

(2) 故障前の系内数モデルの定常分布 $\boldsymbol{p}^{(N,-)}$ から，次式を用いて，故障前のタグ付きタスクモデルの初期分布 $\boldsymbol{p}^{(T,-)}(0)$ を求める．

$$\boldsymbol{p}^{(T,-)}(0) \leftarrow \boldsymbol{p}^{(N,-)}\boldsymbol{\Gamma}^{(-)}$$

(3) $t \leftarrow 0$

(4) 故障後のタグ付きタスクモデルの定常分布 $\boldsymbol{p}^{(T,+)}$ を次式で求める．

$$\boldsymbol{p}^{(T,+)} = \boldsymbol{p}^{(T,-)}(t)\boldsymbol{\Phi}^{(T)}\boldsymbol{\alpha}^{(T,+)}$$

棄却率 $q_F(t)$ は次式より得られる．

$$q_F(-t) = p_F^{(T,+)}$$

(5) Δ 時間後の故障前のタグ付きタスクモデルの状態確率分布を次式で求める

$$\boldsymbol{p}^{(T,-)}(t+\Delta) = \boldsymbol{p}^{(T,-)}(t)(\boldsymbol{I} + \boldsymbol{G}^{(T,-)}\Delta)$$

(6) $t \leftarrow t + \Delta$ とし，$t \geq t_e$ を満たさない場合は (4) に戻る．

3.5.2 故障後に発生したタスク

故障後に発生したタスクの棄却率 $q_F(t)\,(t > 0)$ を導出する方法の概要を示す (図 3.3)．はじめに，故障前の系内数モデルの定常分布 $\boldsymbol{p}^{(N,-)}$ と故障前の系内数モデルから故障後の系内数モデルへの変換行列 $\boldsymbol{\Phi}^{(N)}$ を用いて故障後の系内数モデルの初期分布 $\boldsymbol{p}^{(N,+)}(0)$ を求める．これを用いて，t 時間後の系内数モデルの状態確率 $\boldsymbol{p}^{(N,+)}(t)$ を求め，それに系内数モデルからタグ付きタスクモデルへの変換行列 $\boldsymbol{\Gamma}^{(+)}$ をかけることにより，故障 t 時間後に発生したタグ付きタスクモデルの初期分布 $\boldsymbol{p}^{(T,+)}(0)$ が得られる．これに，吸収確率行列 $\boldsymbol{\alpha}^{(T,+)}$ を掛けて得られるタグ付きタスクモデルの定常分布 $\boldsymbol{p}^{(T,+)}$ の状態 F の確率 $p_F^{(T,+)}$ が，故障 t 時間後に到着したタスクの棄却率 $q_F(t)$ となる．

以下に，導出手順を示す．なお，刻み幅 Δ は精度を考慮してあらかじめ決めておくものとする．また，計算の終了時刻を t_e とする．

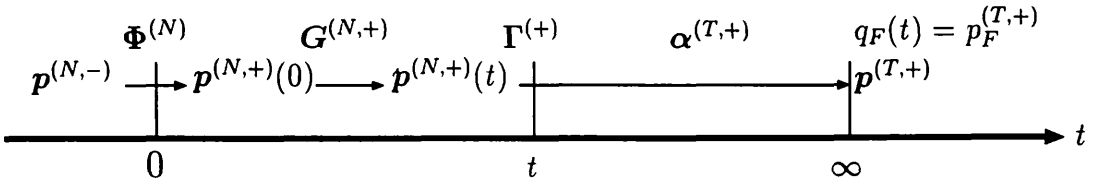


図 3.3: 故障後に発生したタスクの解析方法

手順 3.3 (故障後に発生したタスクの棄却率の導出)

(1) 系内数モデルの定常分布 $\boldsymbol{p}^{(N,-)}$ から，次式を用いて故障後の系内数モデルの初期分布 $\boldsymbol{p}^{(N,+)}(0)$ を求める．

$$\boldsymbol{p}^{(N,+)}(0) = \boldsymbol{p}^{(N,-)}\boldsymbol{\Phi}^{(N)}$$

(2) $t \leftarrow 0$

(3) 故障後のタグ付きタスクモデルの定常分布を次式で求める．

$$\boldsymbol{p}^{(T,+)} = \boldsymbol{p}^{(N,+)}(t)\boldsymbol{\Gamma}^{(+)}\boldsymbol{\alpha}^{(T,+)}$$

棄却率 $q_F(t)$ は，次式より得られる．

$$q_F(t) = p_F^{(T,+)}$$

(4) Δ 時間後の故障後の系内数モデルの状態確率分布を次式で求める．

$$\boldsymbol{p}^{(N,+)}(t+\Delta) = \boldsymbol{p}^{(N,+)}(t)(\boldsymbol{I} + \boldsymbol{G}^{(N,+)}\Delta)$$

(5) $t \leftarrow t+\Delta$ とし，$t \geq t_e$ を満たさない場合は (3) に戻る．

3.6 吸収確率を求めるための計算量

本方法を用いて $\boldsymbol{\alpha}_{TA}^{(T)}$ を導出する場合には，集約を行わない場合に比べて，同値類が集約可能か否かを調べるための演算と同値類を集約するための演算が必要となる．

ここでは，タグ付きタスクモデルの定常分布を集約を行いながら求めた場合と集約を行わずに求めた場合の乗算回数を比較することにより，両者を選択するための基準を明らかにする．

隠れマルコフ連鎖の推移確率行列のブロック $\boldsymbol{H}_{ab}^{(T)}$ を，式 (3.4) により縦方向だけ集約した行列を $\boldsymbol{H}^{(T)\prime}{}_{ab}$ で表し，式 (3.5) により，縦横両方向とも集約した行列を $\boldsymbol{H}^{(T)\prime\prime}{}_{ab}$ で表す．また，集合 $\mathcal{S}^{(T)}$ の状態数を $N(\mathcal{S}^{(T)})$，行列 $\boldsymbol{H}^{(T)}$ の 0 でない要素数を $N(\boldsymbol{H}^{(T)})$ で表す．

集約しない場合の1単位時間後の確率分布の導出に必要な乗算回数と，同値類 $\mathcal{S}_{m+1}^{(T)},\ldots,\mathcal{S}_{M}^{(T)}$ が集約されている場合の1単位時間後の確率分布の導出に必要な乗算回数との差 $D_{\mathrm{mul}}(m)$ は次式で表される．

$$\begin{aligned} D_{\mathrm{mul}}(m) &= \sum_{a=m+1}^{M}\Big\{(N(\boldsymbol{H}_{a\mathcal{A}}^{(T)})-N(\boldsymbol{H}^{(T)\prime}{}_{a\mathcal{A}})) \\ &\quad +\sum_{b=1}^{m}(N(\boldsymbol{H}_{ab}^{(T)})-N(\boldsymbol{H}^{(T)\prime}{}_{ab})) \\ &\quad +\sum_{b=m+1}^{a}(N(\boldsymbol{H}_{ab}^{(T)})-N(\boldsymbol{H}^{(T)\prime\prime}{}_{ab}))\Big\} \end{aligned} \tag{3.9}$$

さらに，同値類 $\mathcal{S}_{m}^{(T)}$ が集約される時刻を t_m で表し，計算終了時刻を t_0 とし，$t_{M+1}=0$ とすると，タグ付きタスクモデルの定常分布を求める際の乗算の減少回数 D_{mul} は次式で表される．

$$\begin{aligned} D_{\mathrm{mul}} &= \sum_{m=0}^{M}(t_m-t_{m+1})D_{\mathrm{mul}}(m) \\ &= t_0\Big\{N(\boldsymbol{H})-\sum_{a=1}^{M}\Big(N(\boldsymbol{H}^{(T)\prime}{}_{a\mathcal{A}})+\sum_{b=1}^{a}N(\boldsymbol{H}^{(T)\prime\prime}{}_{ab})\Big)\Big\} \\ &\quad +\sum_{m=1}^{M}t_m\Big\{N(\boldsymbol{H}^{(T)\prime}{}_{m\mathcal{A}})-N(\boldsymbol{H}_{m\mathcal{A}}^{(T)}) \\ &\qquad +N(\boldsymbol{H}^{(T)\prime\prime}{}_{mm})-N(\boldsymbol{H}_{mm}^{(T)}) \\ &\qquad +\sum_{a=m+1}^{M}(N(\boldsymbol{H}^{(T)\prime\prime}{}_{am})-N(\boldsymbol{H}^{(T)\prime}{}_{am}))\Big\} \end{aligned} \tag{3.10}$$

また，全ての同値類を集約するために増加する乗算回数は次式で表せる．

$$N(\boldsymbol{H}_{\mathcal{TT}}^{(T)})+N(\boldsymbol{H}_{\mathcal{TA}}^{(T)}) \tag{3.11}$$

よって，乗算回数が減少するための条件は次式で与えられる．

$$\begin{aligned} N(\boldsymbol{H}_{\mathcal{TT}}^{(T)})+N(\boldsymbol{H}_{\mathcal{TA}}^{(T)}) &\leq t_0\Big\{N(\boldsymbol{H})-\sum_{a=1}^{M}\Big(N(\boldsymbol{H}^{(T)\prime}{}_{a\mathcal{A}})+\sum_{b=1}^{a}N(\boldsymbol{H}^{(T)\prime\prime}{}_{ab})\Big)\Big\} \\ &\quad +\sum_{m=1}^{M}t_m\Big\{N(\boldsymbol{H}^{(T)\prime}{}_{m\mathcal{A}})-N(\boldsymbol{H}_{m\mathcal{A}}^{(T)}) \\ &\qquad +N(\boldsymbol{H}^{(T)\prime\prime}{}_{mm})-N(\boldsymbol{H}_{mm}^{(T)}) \\ &\qquad +\sum_{a=m+1}^{M}(N(\boldsymbol{H}^{(T)\prime\prime}{}_{am})-N(\boldsymbol{H}^{(T)\prime}{}_{am}))\Big\} \end{aligned} \tag{3.12}$$

この式の右辺第2項目は，

$$\begin{aligned} N({\boldsymbol{H}^{(T)\prime}}_{m\mathcal{A}}) &\leq N(\boldsymbol{H}^{(T)}_{m\mathcal{A}}) \\ N({\boldsymbol{H}^{(T)\prime\prime}}_{mm}) &\leq N(\boldsymbol{H}^{(T)}_{mm}) \qquad (m=1,2,\ldots,M;\ a=m+1,m+2,\ldots,M) \\ N({\boldsymbol{H}^{(T)\prime\prime}}_{am}) &\leq N({\boldsymbol{H}^{(T)\prime}}_{am}) \end{aligned} \tag{3.13}$$

であることから，正の値になることはないため，乗算回数が減少することはない．このことから計算終了時刻 t_0 が大きく，同値類 $\mathcal{S}_m^{(T)}$ が集約される時刻 $t_m(m=1,2,\ldots,M)$ が小さい程，さらに，同値類の数 M が少ない程，本方法により乗算回数が減少することがわかる．

3.7 まとめ

本章では，系内数モデルとタグ付きタスクモデルを用いて，リスポンシブシステム中にプロセッサ故障が発生した場合の過渡状態におけるタスクの棄却率を求める方法を示した．

タスクの棄却率はタグ付きタスクモデルの定常状態確率で表され，過渡状態の解析においてはタグ付きタスクモデルの初期分布を変えながら，極限状態確率を繰り返し求める必要がある．本方法では，定常状態確率を求めるために吸収確率を定義し，これを用いて短時間にタスクの棄却率を求めている．さらに，タグ付きタスクモデルの状態を相互到達可能性に基づく同値類に分割し，状態確率の和が定められた閾値以下となった同値類を集約することによって短時間に吸収確率を求めることが可能になっている．

最後に，集約を行いながら吸収確率を求める場合の演算回数と，集約を行わずに求める場合の演算回数を比較し，本方法を適用するための基準を示した．

4. 解析例

4.1 概要

本章では，処理要素のグループが一つでありタスクの処理に優先度があるリスポンシブシステムと，分散型リスポンシブシステムのようにネットワークで接続された複数の処理要素のグループからなるリスポンシブシステムの解析例を示す．

はじめに，4.2 節では，処理要素のグループが一つであり，タスクの処理に優先度があるリスポンシブシステムの解析例を示し，4.3 節では，複数の処理要素のグループからなるシステムの解析例を示す．最後に 4.4 節で解析結果をまとめる．

4.2 一個の処理要素のグループからなるシステムの解析

4.2.1 システムの仮定

本節では，処理要素のグループが一つであり，処理されるタスクに優先度があるリスポンシブシステムの解析を行う．

はじめに，本節で扱うシステムで処理されるタスクの優先度について以下のような仮定をおく．

仮定 4.1 (タスクの優先度)

(1) 同じタスククラスに属するタスクは等しい優先度を持つ．
(2) クラスの番号が小さいタスクほど優先度は高い． ∎

つぎに，システムとタスクのスケジューリング方法について次のように仮定する．

仮定 4.2 (システムとタスクのスケジューリング)

(1) システムは K 個のプロセッサと，長さ $L-K$ の一つの待ち行列を持つ．
(2) すべてのタスクはプリエンプティブに処理される．

(3) システムに到着したタスクは，空いているプロセッサがあれば直ちにそこに割り当てられる．空きがない場合は待ち行列に入れられる．

(4) 待ち行列中のタスクは優先度の高い順に並び，同一優先度のタスクは先着順に並ぶ．

(5) 待ち行列に空きがない時に到着したタスクについて，このタスクより低い優先度のタスクが待ち行列中にない場合は捨てられ，低い優先度のタスクがある場合には，待ち行列の最後のタスクが捨てられて，到着したタスクはシステムに受理される．∎

プロセッサに故障が発生したときには，故障したプロセッサで処理されていたタスクはその時のシステムの状況に応じて的確に処理されねばならない．ここでは，このようなタスクは次のように処理されるものとする．ここで，故障後のプロセッサ数とシステム容量をそれぞれ $K^{(+)}$，$L^{(+)}$ で表す．

仮定 4.3 (故障時の処理)

(1) 故障が発生したプロセッサで処理されていたタスクは，次のように処理される．

- 他に空いているプロセッサがあれば，そこに割り当てて処理を継続する．
- 空いているプロセッサがない場合は，一旦待ち行列の先頭に戻される．しかし，そのタスクよりも優先度が低いか，もしくは後から発生した同じクラスのタスクがプロセッサで処理されている場合は，そのうちの最も優先度の低いタスクと入れ換える．

(2) 故障発生以前の最も近いチェックポイントの状態が保存され，タスクの処理が再開される際には，その情報を元にしてロールバック処理を行なう．ただし，ロールバックに要する時間はタスクの処理時間に比べて十分に短いものとする．

(3) システム内に存在するタスク数が $L^{(+)}$ を超える場合は，待ち行列の最後にあるタスクが捨てられる．∎

4.2.2 系内数モデル

前節で示したリスポンシブシステムの解析を行う場合は，各クラスのタスクの系内数の組合せが系内数モデルの状態となる．系内数モデルの状態と状態集合を以下のように定義する．

定義 4.1 (系内数モデルの状態)
系内数モデルの状態を次式で定義する．

$$(i_1, i_2, \ldots, i_k, \ldots, i_{N_c}) \tag{4.1}$$

ここで，$i_k (k = 1, 2, \ldots, N_c)$ はクラス k に属するタスクの系内数を表す． ∎

系内数の最大値が L であることから，系内数モデルの状態集合 $\mathcal{S}^{(N)}$ は次式のようになる．

$$\mathcal{S}^{(N)} = \{(i_1, i_2, \ldots, i_k, \ldots, i_{N_c}) | \sum_{k=1}^{N_c} i_k \leq L\} \tag{4.2}$$

次に，推移率 $g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'}$ を求める．推移率は，全ての状態 $\boldsymbol{n}$ について，次のような場合分けを行うことにより求めることができる．ここで，$\boldsymbol{n} = (i_1, i_2, \ldots, i_k, \ldots, i_{N_c})$ と表す．

(1) タスクが発生することによる推移

待ち行列に空きがある場合とない場合によって，推移する状態が異なるため，さらに以下のように場合分けすることにより，推移率を求める．

- $\sum_{k=1}^{N_c} i_k < L$

  待ち行列に空きがある場合を表しており，この条件を満たす状態にあるときに発生したタスクは受理され，系内数が増加する．発生したタスクのクラスを j で表すとすると，推移率は次式で与えられる．

$$\begin{aligned} \boldsymbol{n}' &= (i_1, \ldots, i_j + 1, \ldots, i_{N_c}) \\ g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= \lambda_j \quad (j = 1, 2, \ldots, N_c) \end{aligned}$$

- $\sum_{k=1}^{N_c} i_k = L$

  待ち行列に空きがない場合を表しており，このときに待ち行列の最後のタスクよりも優先度の高いタスクが発生すると，待ち行列の最後のタス

クが捨てられ，発生したタスクは受理される．よって，待ち行列中のタスクのうち最も優先度の低いタスクのクラスを j とし，発生したタスクのクラスを j' とすると，推移率は次式で与えられる．

$$\begin{aligned} \boldsymbol{n}' &= (i_1,\ldots,i_{j'}+1,\ldots,i_j-1,\ldots,i_{N_c}) \\ g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= \lambda_{j'} \quad (j'=1,2,\ldots,j-1) \end{aligned}$$

(2) タスクの処理が終了するか，もしくは捨てられることによる推移

$\sum_{k=1}^{N_c} i_k \neq 0$ である状態は，プロセッサにタスクが割り当てられていることを表しており，タスクの処理が終了するか制限時間を超えることにより，系内数が減少する．プロセッサに割り当てられているタスクのうち最も優先度が低いタスクのクラスを j とすると，プロセッサには優先度の高いタスクから割り当てられるので，クラス $j'(=1,2,\ldots,j)$ のタスクのうち，プロセッサに割り当てられている個数は $\min(K-\sum_{k=1}^{j'-1} i_k, i_{j'})$ となる．また，全てのタスクが制限時間を超える可能性があるので，推移率は次式で与えられる．

$$\begin{aligned} \boldsymbol{n}' &= (i_1,\ldots,i_{j'}-1,\ldots,i_{N_c}) \\ g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= \begin{cases} \min(K-\displaystyle\sum_{k=1}^{j'-1} i_k, i_{j'})\mu_{j'} + i_{j'}\nu_{j'} & (j'=1,2,\ldots,j) \\ i_{j'}\nu_{j'} & (j'=j+1,j+2,\ldots,N_c) \end{cases} \end{aligned}$$

4.2.3 タグ付きタスクモデル

次に，クラス J のタスクを解析する場合のタグ付きタスクモデルを示す．

ここで検討を行うシステムの場合，処理要素のグループが一つであることから，定義2.2におけるタグ付きタスクモデルの状態を表すパラメータのうち，タグ付きタスクが滞在する処理要素のグループを表す l は無視することができる．また，仮定4.2に示した優先度順のスケジューリングを行うため，タスククラス J よりも優先度の低いクラスに属するタスクはタグ付きタスクの処理に影響を与えない．従って，このようなタスククラスに属するタスクの系内数 $i_k(k=J+1,J+2,\ldots,N_c)$ とタグ付きタスクの処理要素内での位置 r は考慮する必要がない．以上を考慮し，タグ付きタスクモデルの状態を次のように定義する．

定義 4.2 (タグ付きタスクモデルの状態)
タグ付きタスクモデルの状態は，状態 C，F と次式のようなタグ付きタスクが処理中である状態からなる．

$$(i_1, i_2, \ldots, i_k, \ldots, i_J)$$

ここで，i_k $(k = 1, 2, \ldots, J-1)$ はクラス k のタスクの系内数を表し，i_J は，タグ付きタスクよりも早く発生して，まだ処理が終了していないクラス J のタスクの個数を表す． ∎

クラス J のタスクに関するタグ付きタスクモデルの状態集合 $\mathcal{S}^{(T)}$ は，次式で表される．

$$\mathcal{S}^{(T)} = \{C, F, (i_1, i_2, \ldots, i_k, \ldots, i_J) | \sum_{k=1}^{J} i_k < L\}$$

タグ付きタスクモデルの推移率 $g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'}$ は，すべての状態 $\boldsymbol{u}$ について，次のような場合分けを行うことにより求めることができる．ここで，$\boldsymbol{u} = (i_1, i_2, \ldots, i_k, \ldots, i_J)$ と表す．

(1) タスクが発生することによる推移
タグ付きタスクの位置により，推移する状態が異なるため，さらに以下のように場合分けすることにより，推移率を求める．

(a) $\sum_{k=1}^{J} i_k < L - 1$
タグ付きタスクより優先度が高いタスクが発生することによって状態が推移する．発生したタスクのクラスを $k(= 1, 2, \ldots, J-1)$ とすると，推移率は次式で与えられる．

$$\begin{aligned} \boldsymbol{u}' &= (i_1, \ldots, i_k + 1, \ldots, i_J) \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \lambda_k \quad (k = 1, 2, \ldots, J-1) \end{aligned}$$

(b) $\sum_{k=1}^{J} i_k = L - 1$
待ち行列に空きがなく，かつタグ付きタスクが待ち行列の最後尾にある．従って，タグ付きタスクよりも優先度の高いタスクが発生することによってタグ付きタスクが捨てられる．また，タグ付きタスク自身が制限時間を満たせなかった場合にもタグ付きタスクは捨てられる．よって，この場合

の推移率は次式で与えられる.

$$\begin{aligned} \boldsymbol{u}' &= F \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \sum_{k=1}^{J-1} \lambda_k + \nu_J \end{aligned}$$

(2) タグ付きタスク以外のタスクの処理が終了するか，もしくは捨てられることによる推移

プロセッサに割り当てられているタスクのうち，最も優先度の低いクラスを j とすると，プロセッサには優先度の高いタスクから割り当てられるので，クラス $j'(=1,2,\ldots,j)$ のタスクのうちプロセッサに割り当てられている個数は $\min(L-\sum_{k=1}^{j'-1} i_k, i_{j'})$ となる．また，すべてのタスクは制限時間が満たせなかったことにより捨てられることがあるので，推移率は次式で表される.

$$\begin{aligned} \boldsymbol{u}' &= (i_1,\ldots,i_{j'}-1,\ldots,i_J) \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \begin{cases} \min(K-\sum_{k=1}^{j-1} i_k, i_{j'})\mu_k + i_{j'}\nu_{j'} & (j'=1,2,\ldots,j) \\ i_{j'}\nu_{j'} & (j'=j+1,j+2,\ldots,J) \end{cases} \end{aligned}$$

(3) タグ付きタスクの処理が終了することによる推移

この推移は，$\sum_{k=1}^{J} i_k < K$ である場合にのみ発生し，推移率は次式で表される.

$$\boldsymbol{u}' = C$$

$$g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} = \mu_J$$

(4) タグ付きタスクが捨てられることによる推移

$\sum_{k=1}^{J} i_k = L-1$ である場合は，既に (1)-(b) で示した.

これ以外の場合は，タグ付きタスクが制限時間を満たせずに捨てられる場合のみであるので，推移率は次式で表される.

$$\boldsymbol{u}' = F$$

$$g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} = \nu_J$$

4.2.4 変換行列

はじめに，系内数モデルからタグ付きタスクモデルへの変換行列 $\mathbf{\Gamma}$ を求める．系内数モデルのすべての状態 $\boldsymbol{n}$ について，次のような条件を調べることによって，$\gamma_{\boldsymbol{n},\boldsymbol{u}}$ を求める．

- $\sum_{k=1}^{J} i_k < L$ の場合

  待ち行列に空きがあるか，もしくは，タグ付きタスクよりも優先度の低いタスクが待ち行列に存在するので，タグ付きタスクは受理される．

$$\boldsymbol{u} = (i_1, i_2, \ldots, i_J)$$

$$\gamma_{\boldsymbol{n},\boldsymbol{u}} = 1$$

- $\sum_{k=1}^{J} i_k = L$ の場合

  待ち行列に空きがないので，タグ付きタスクは直ちに捨てられる．

$$\boldsymbol{u} = F$$

$$\gamma_{\boldsymbol{n},\boldsymbol{u}} = 1$$

次に，故障前の系内数モデルから故障後の系内数モデルへの変換行列 $\mathbf{\Phi}^{(N)}$ を求める手順を示す．故障前の系内数モデルのすべての状態 $\boldsymbol{n}$ について以下の場合分けを行い，$\phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'}$ を求める．

- $\sum_{k=1}^{N_c} i_k \leq L^{(+)}$ の場合

  故障により捨てられるタスクはないので，状態は変化しない．

$$\boldsymbol{n}' = \boldsymbol{n}$$

$$\phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} = 1$$

- $\sum_{k=1}^{N_c} i_k > L^{(+)}$ の場合

  待ち行列内に存在する最も優先度の低いタスクが属するクラスを j とすると，クラス j のタスクが一つ捨てられる．

$$\boldsymbol{n}' = (i_1, i_2, \ldots, i_j - 1, \ldots, i_{N_c})$$

$$\phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} = 1$$

最後に，故障前のタグ付きタスクモデルから故障後のタグ付きタスクモデルへの変換行列 $\mathbf{\Phi}^{(T)}$ を求める手順を示す．故障前のタグ付きタスクモデルのすべての状態 $\boldsymbol{u}$ について以下の場合分けを行うことにより，$\phi^{(T)}_{\boldsymbol{u},\boldsymbol{u}'}$ を求める．

- $\sum_{k=1}^{J} i_k \leq L^{(+)}$ 場合

  システム容量に空きがあるか，タグ付きタスクよりも優先度の低いタスクがあるので，タグ付きタスクの状態は変化しない．

$$\boldsymbol{u}' = \boldsymbol{u}$$

$$\phi^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} = 1$$

- $\sum_{k=1}^{J} i_k > L^{(+)}$ の場合

  システム容量に空きがなく，システム内で処理されているタスクのうち，タグ付きタスクが最も優先度の低いタスクであるので，タグ付きタスクは捨てられる．

$$\boldsymbol{u}' = F$$

$$\phi^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} = 1$$

4.2.5 解析結果

表4.1 に示すパラメータを用いて，タスククラス 3 のタスクの棄却率を解析した結果を示す．このパラメータの場合，故障後のタグ付きタスクモデルの状態数は 562 個である．

タグ付きタスクモデルの状態集合を相互到達可能性に基づいて同値類に分割した結果を図 4.1 に示す．なお，ここでは状態 C 以外の全ての状態から生じるタグ付きタスクが捨てられたことを表す状態 F への推移は省略してある．図をみると分かるように，タグ付きタスクと同じタスククラスに属し，タグ付きタスクよりも早く発生してまだ処理が終了していないタスクの個数を表す i_3 が等しい状態が同値類となる．これは，タグ付きタスクより優先度が高いクラスのタスクは，タグ付きタスクの位置にかかわらず優先的に処理されるため，これらの系内数 i_1，i_2

は増加する可能性がある．しかし，i_3 は減少するのみであり増加することはないので，i_3 が異なる状態は互いに到達可能とはならないからである．

集約条件の閾値 ε，計算終了条件の閾値 δ と，$\boldsymbol{\alpha}_{TA}^{(T)}$ の計算時間の関係を図 4.2 に示す．$\varepsilon = 0$ は集約を行わない場合の計算時間を表す．全ての閾値の組合せにおいて，集約を行った方が計算時間が減少しており，本方法が計算時間の削減に有効であることがわかる．また，計算時間は集約条件の閾値 ε に依存し，計算終了の閾値 δ にはほとんど依存していない．これは式 (3.12) から予想される結果と一致する．$\delta = 1 \times 10^{-5}$ の場合，計算時間は，$\varepsilon = 0.001$ で約 60%，$\varepsilon = 0.01$ で約 50% となっている．

次に，閾値と吸収確率 $\boldsymbol{\alpha}^{(T)}{}_{\boldsymbol{u},\boldsymbol{u}'}$ の相対誤差の関係を図 4.3 に示す．図は各閾値の組合せで得られた吸収確率の相対誤差のうち最大のものを表しており，集約を行わずに $\delta = 1 \times 10^{-9}$ として得られた値を真値とみなしている．$\delta \leq 1 \times 10^{-4}$ の範囲では，計算時間と同様に ε のみに依存しており，$\delta > 1 \times 10^{-4}$ の範囲では，$\varepsilon = 0.001$ 以外では，ε に依存せず，δ に依存している．これは，$\delta > 1 \times 10^{-4}$ の範囲ではすべての状態が集約されないうちに終了条件が満たされてしまうため，このような結果となるものと考えられる．

最後に．棄却率 $q_F(t)$ の計算結果を図 4.4 に示す．どの閾値の場合も誤差は 1%以内となっており十分な精度で解析可能であることがわかる．

表 4.1: パラメータ

(a) リスポンシブシステムのパラメータ

L	システム容量	15
K	プロセッサ数	3
N_c	タスククラス数	3

(b) タスククラスのパラメータ

n	λ_n	μ_n	ν_n
1	0.05	0.1	0.001
2	0.025	0.05	0.0005
3	0.0125	0.025	0.00025

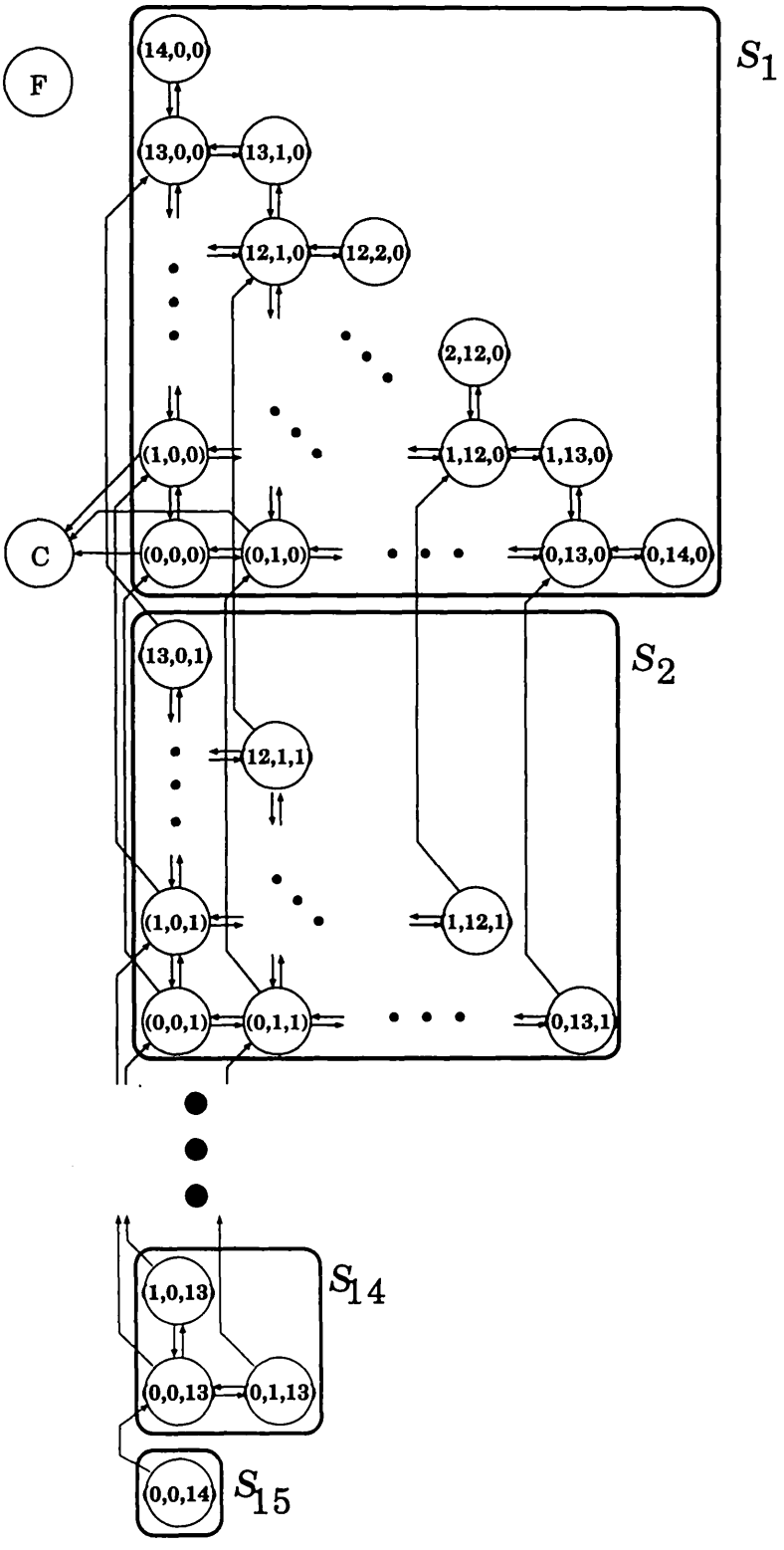


図 4.1: タグ付きタスクモデルの同値類

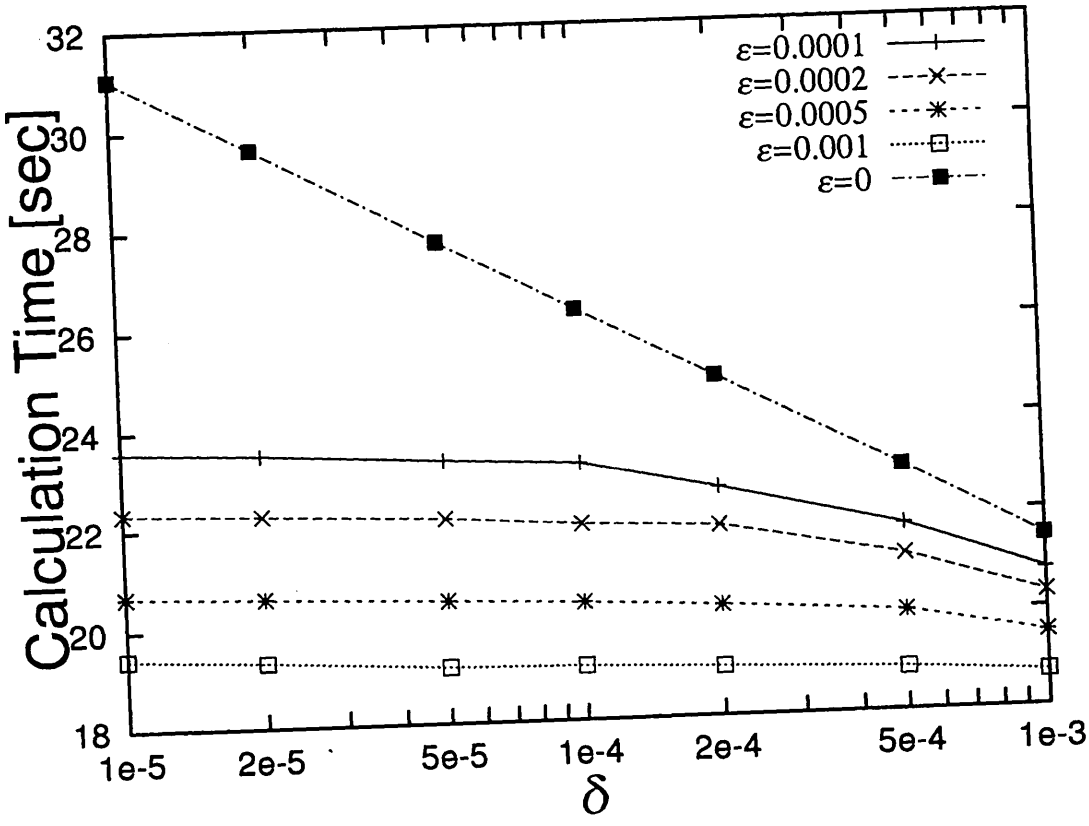


図 4.2: 閾値と $\alpha_{TA}^{(T)}$ の計算時間 (CPU:PentiumII/300MHz)

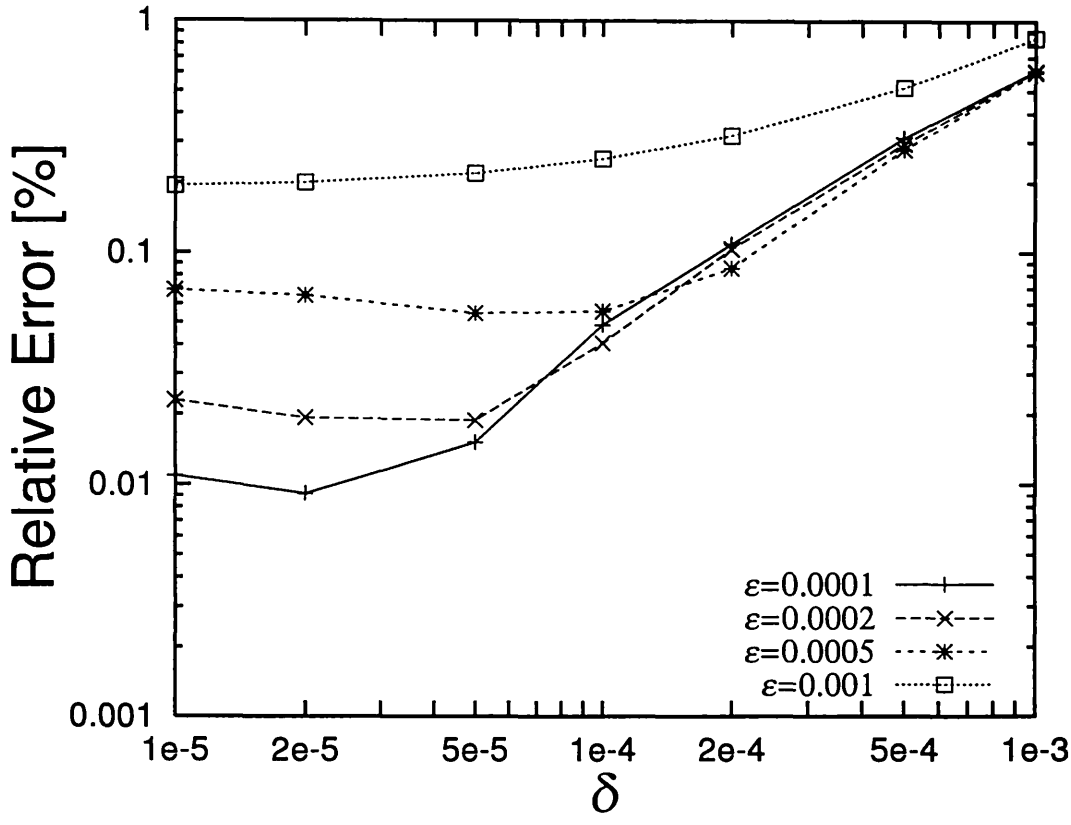


図 4.3: 閾値と $\alpha^{(T)}{}_{\boldsymbol{u},\boldsymbol{u}'}$ の相対誤差の最大値

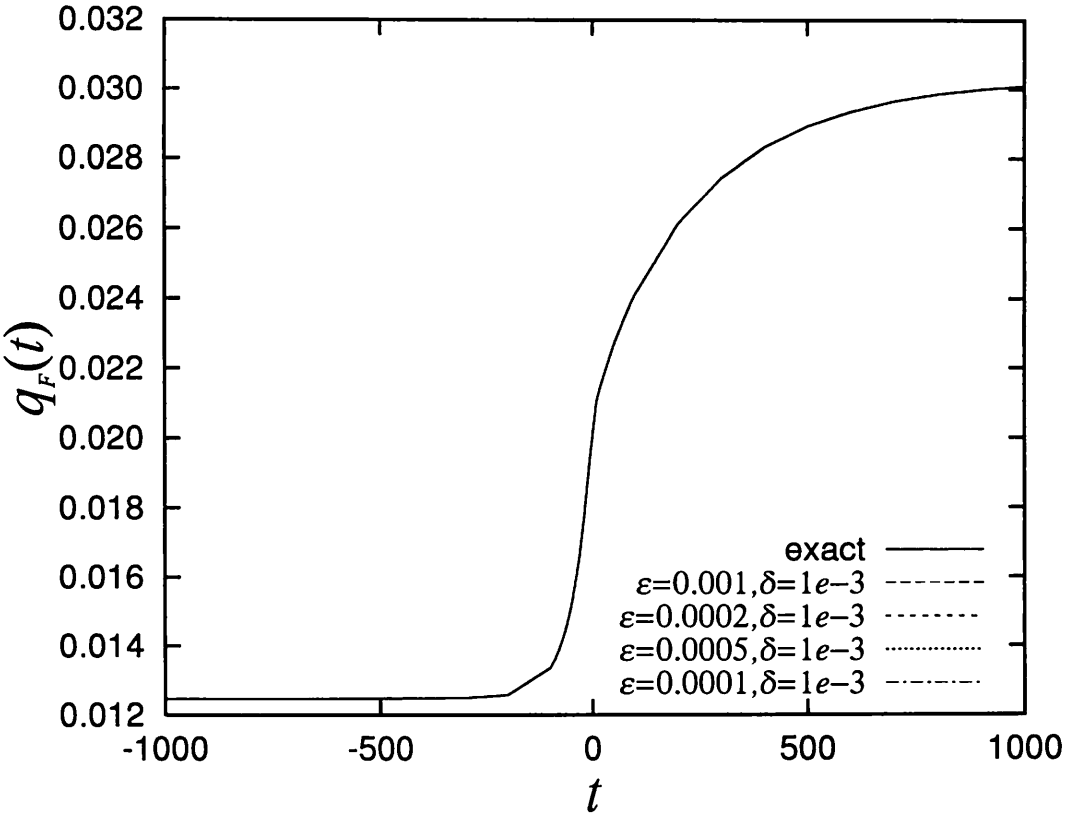

0.032
0.030
0.028
0.026
0.024
0.022
0.020
0.018
0.016
0.014
0.012
$q_F(t)$
-1000
-500
0
500
1000
t
exact
ε=0.001,δ=1e−3
ε=0.0002,δ=1e−3
ε=0.0005,δ=1e−3
ε=0.0001,δ=1e−3

図 4.4: 棄却率の変化

4.3 複数の処理要素のグループからなるシステムの解析

4.3.1 システムの仮定

本節では，分散型リスポンシブシステムのように，ネットワークで接続された複数の処理要素のグループからなるシステムの解析例を示す．

分散型リスポンシブシステムは，待ち行列ネットワークによってモデル化される．待ち行列ネットワーク (QN) は有向グラフを用いて表される．グラフの各々のノードは複数のプロセッサとひとつの待ち行列からなる処理要素のグループを表し，有向エッジはタスクの移動を表す．QN を次のように定義する．

定義 4.3 (待ち行列ネットワーク)
待ち行列ネットワーク QN を次式で定義する．

$$\mathrm{QN} = (V, E, \Pi) \tag{4.3}$$

$$V = \{v_0, v_1, \ldots, v_{N_r}\}: \text{ ノードの集合}$$
$$E = \{e_{n,n'}\}: \text{ エッジの集合}$$
$$\Pi = \{\pi_{n,n'}\}: \text{ エッジのラベルの集合}$$

ここで，ノード v_0 は QN の外部を表し，エッジ $e_{n,n'}$ とそのラベル $\pi_{n,n'}$ はノード v_n で処理が終了したタスクが，ノード $v_{n'}$ へ確率 $\pi_{n,n'}$ で移動することを表す．∎

QN のノード及びタスクについて以下のような仮定をおく．

仮定 4.4 (ノードとタスク)

(1) QN はフィードフォワード網であり，すべてのノードには $\pi_{n,n'} > 0$ ならば，$n > n'(1 \le n \le N_r; 0 \le n' \le N_r)$ となるように番号を付ける．
(2) ノード v_n は K_n 個のプロセッサと長さ $L_n - K_n$ の待ち行列を持つ．
(3) タスクの到着はパラメータ λ のポアソン分布に従う．
(4) ノード v_n でのタスクの処理に要する時間はパラメータ μ_n の指数分布に従う．
(5) すべてのノードにおいて，タスクは先着順優先方式で処理される．
(6) タスクの制限時間はパラメータ ν の指数分布に従う．また，制限時間を超えたタスクは直ちに捨てられる．∎

また，故障が発生した場合の処理について以下のように仮定する．

仮定 4.5 (故障と故障時の処理)

(1) 故障は唯ひとつのノードのプロセッサにのみ起こる．
(2) 故障が発生したプロセッサで処理されていたタスクは，他に空いているプロセッサがあれば，そこに割り当てられ処理を継続する．空いているプロセッサがなければ，待ち行列の先頭に戻される．
(3) システム中のタスクの数が故障後のノードの容量を超える場合は，待ち行列の最後のタスクが捨てられる． ∎

故障が発生したノードの番号を n_F で表し，故障後のノード v_{n_F} のプロセッサ数を $K_{n_F}^{(+)}$，容量を $L_{n_F}^{(+)}$ で表す．

4.3.2 系内数モデル

分散型リスポンシブシステムにおける系内数モデルの状態を次のように定義する．

定義 4.4 (系内数モデルの状態)
QN の系内数モデルの状態を次式により定義する．

$$(i_1, i_2, \ldots, i_n, \ldots, i_{N_r}) \tag{4.4}$$

ここで $i_n (0 \leq i_n \leq L_n)$ はノード v_n の系内数を表す． ∎

系内数モデルの状態集合 $\mathcal{S}^{(N)}$ は，次式で表される．

$$\mathcal{S}^{(N)} = \{(i_1, i_2, \ldots, i_n, \ldots, i_{N_r}) \mid 0 \leq i_n \leq L_n; n = 1, 2, \ldots, N_r\} \tag{4.5}$$

系内数モデルの推移率 $g_{\boldsymbol{n},\boldsymbol{n}'}^{(N)}$ は，次のように場合分けをすることにより求めることができる．ここで，$\boldsymbol{n} = (i_1, i_2, \ldots, i_{N_r})$，$\boldsymbol{n}' = (i_1', i_2', \ldots, i_{N_r}')$ とする．

[系内数モデルの推移率]

(1) ノード $v_n (\in \{v_n | \pi_{0,n} > 0 \wedge i_n < L_n\})$ に外部からタスクが到着し受理される場合

$$\begin{aligned} i_n' &= i_n + 1 \\ g_{\boldsymbol{n},\boldsymbol{n}'}^{(N)} &= \lambda \pi_{0,n} \quad (n = 1, 2, \ldots, N_c) \end{aligned}$$

(2) ノード $v_n(\in \{v_n|i_n \neq 0\})$ での処理を終えたタスクがノード $v_{n'}$ に送られる場合

$$\begin{aligned} i'_n &= i_n - 1 \\ i'_{n'} &= \min(i_{n'} + 1, L_{n'}) \\ g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= \min(K_n, i_n)\pi_{n,n'}\mu_n \quad (n' = n+1, n+2, \ldots, N_c) \end{aligned}$$

(3) ノード $v_n(\in \{v_n|i_n \neq 0\})$ での処理を終えたタスクが QN の外部に出力されるか，ノード v_n にあるタスクが捨てられる場合

$$\begin{aligned} i'_n &= i_n - 1 \\ g^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= \min(K_n, i_n)\pi_{n,0}\mu_n + i_n\nu_n \quad (n' = n+1, n+2, \ldots, N_c) \end{aligned}$$

4.3.3 タグ付きタスクモデル

分散型リスポンシブシステムのタグ付きタスクモデルを以下のように定義する．

定義 4.5 (タグ付きタスクモデルの状態)
タグ付きタスクモデルは，状態 C と F，さらにタグ付きタスクが処理中であることを表す次のような状態からなる．

$$(i_1, i_2, \ldots, i_n, \ldots, i_{N_r}, l, r)$$

ここで，i_n はノード v_n の系内数，$l(1 \leq l \leq N_r)$ はタグ付きタスクが滞在しているノード，$r(0 \leq L_n - K_n)$ はタグ付きタスクのノード内での位置を表す．$r = 0$ の場合はタグ付きタスクがプロセッサで処理されていることを表し，$r > 0$ の場合は待ち行列の r 番目にあることを表す． ∎

タグ付きタスクモデルの状態集合 $\mathcal{S}^{(T)}$ は次式のようになる．

$$\begin{aligned} \mathcal{S}^{(T)} = \{&C, F, (i_1, i_2, \ldots, i_n, \ldots, i_{N_r}, l, r) \\ &| \ 1 \leq l \leq N_r;\ 0 \leq r \leq L_n - K_n; \\ &\quad 0 \leq i_n \leq L_n \ (n = 1, 2, \ldots, l-1, l+1, \ldots, N_r); \\ &\quad 1 \leq i_l \leq L_l \ (r = 0 \text{である場合}); \\ &\quad K_l + r \leq i_l \leq L_l \ (r > 0 \text{である場合})\} \end{aligned} \tag{4.6}$$

タグ付きタスクモデルの推移率 $g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'}$ は次のように場合分けすることによって求めることができる．ここで，$\boldsymbol{u} = (i_1, i_2, \ldots, i_{N_r}, l, r)$，$\boldsymbol{u}' = (i'_1, i'_2, \ldots, i'_{N_r}, l', r')$ とする．

[タグ付きタスクモデルの推移率]

タグ付きタスクが滞在しているノード以外の系内数が変化する場合は QN モデルの推移率に等しい．タグ付きタスクが滞在しているノード v_l の系内数が変化する場合の推移率は次のように場合分けをすることにより求めることができる．

(1) タグ付きタスクの処理が終了し QN を退去することによる推移

$r = 0$ かつ $\pi_{l,0} \neq 0$ である場合にこのような推移が起こる可能性があり，推移率は次式で与えられる．

$$\begin{aligned} \boldsymbol{u}' &= C \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \mu_l \pi_{l,0} \end{aligned}$$

(2) タグ付きタスクの処理が終了し，ノード $v_n (\in \{v_n | \pi_{l,n} > 0\})$ へ送られることによる推移

$r = 0$ かつ $\pi_{l,n} > 0$ かつ $i_n < L_n$ である場合にこのような推移が起こる可能性があり，推移率は次式で与えられる．

$$\begin{aligned} i'_l &= i_l - 1 \\ i'_n &= i_n + 1 \\ l' &= n \\ r' &= \max(i_n - K_n, 0) \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \mu_l \pi_{l,i} \end{aligned}$$

(3) タグ付きタスクが捨てられることによる推移

制限時間を満たせないか，もしくはノード l での処理が終了しノード $v_n (\in \{v_n | \pi_{l,n} > 0\})$ へ送られたときに，ノード v_n の待ち行列に空きがない場合にこのような推移が起こる．

よって，$i_n = L_n$ である状態 $\boldsymbol{u}$ からの推移率は次式で与えられる．

$$\boldsymbol{u}' = F$$

$$g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} = \mu_l \pi_{l,n} + \nu$$

それ以外の状態からの推移率は，次式で与えられる．

$$\begin{aligned} \boldsymbol{u}' &= F \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \nu \end{aligned}$$

(4) タスクが QN 外部からノード v_l に到着することによる推移

この推移は $i_l < L_l$ である場合にのみ起こり，推移率は次式で与えられる．

$$\begin{aligned} i'_l &= i_l + 1 \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \lambda \pi_{0,l} \end{aligned}$$

(5) ノード v_l で処理されていたタグ付きタスク以外のタスクが制限時間を満たせずに捨てられることによる推移

この場合は，タグ付きタスクがプロセッサで処理されている場合 $(r = 0)$ と，待ち行列にある場合 $(r > 0)$ によって推移する状態が異なる．

(a) $r = 0$ の場合

タグ付きタスクはプロセッサに割り当てられているので，タグ付きタスクの位置は変化しない．この場合の推移率は次式で与えられる．

$$\begin{aligned} i'_l &= i_l - 1 \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= (i_l - 1)\nu \end{aligned}$$

(b) $r > 0$ の場合

待ち行列中で，タグ付きタスクより前にあるタスクが捨てられる場合と，後ろにあるタスクが捨てられる場合がある．

タグ付きタスクよりも前にあるタスクが捨てられる場合の推移率は，次式で与えられる．

$$\begin{aligned} i'_l &= i_l - 1 \\ r' &= r - 1 \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= (K_l + r - 1)\nu \end{aligned}$$

タグ付きタスクよりも後ろにあるタスクが捨てられる場合の推移率は，次式で与えられる．

$$\begin{aligned} i_l' &= i_l - 1 \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= (i_l - K_l + r)\nu \end{aligned}$$

(6) ノード v_l で処理されていたタグ付きタスク以外のタスクの処理が終了し，ノード v_n へ送られることによる推移

タグ付きタスクがプロセッサに割当てられている場合 $(r = 0)$ と，タグ付きタスクが待ち行列中にある場合 $(r > 0)$ で，推移する状態が異なる．

(a) $r = 0$ の場合

タグ付きタスクはプロセッサに割り当てられているので，タグ付きタスクの位置は変化しない．この場合の推移率は次式で与えられる．

$$\begin{aligned} i_l' &= i_l - 1 \\ i_n' &= \min(i_n + 1, L_n) \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= \min(i_l - 1, K_l - 1)\pi_{l,n}\mu_l \end{aligned}$$

(b) $r > 0$ の場合

タグ付きタスクは待ち行列の一つ前に移動する．この場合の推移率は次式で与えられる．

$$\begin{aligned} i_l' &= i_l - 1 \\ i_n' &= \min(i_n + 1, L_n) \\ r' &= r - 1 \\ g^{(T)}_{\boldsymbol{u},\boldsymbol{u}'} &= K_l\pi_{l,n}\mu_l \end{aligned}$$

4.3.4 変換行列

はじめに，系内数モデルからタグ付きタスクモデルへの変換行列 Γ を求める．系内数モデルのすべての状態 $\boldsymbol{n}$ について，次のような条件を調べることによって，$\gamma_{\boldsymbol{n},\boldsymbol{u}}$ を求める．

- タスクが外部から到着する可能性のある全てのノード $v_n(\in \{v_n|\pi_{0,n} > 0\})$ について，以下の条件を調べる．

  (1) $i_n = L_n$ の場合

  ノード n に到着したタスクは捨てられる．

$$\begin{aligned} \boldsymbol{u} &= F \\ \gamma_{\boldsymbol{n},\boldsymbol{u}} &= \pi_{0,n} \end{aligned}$$

  (2) 上記以外の場合

  到着したタスクは受理される．

$$\begin{aligned} \boldsymbol{u} &= (i_1, \ldots, i_n + 1, \ldots, i_{N_r}, n, \max(i_n - K_n, 0)) \\ \gamma_{\boldsymbol{n},\boldsymbol{u}} &= \pi_{0,n} \end{aligned}$$

次に，故障前の系内数モデルから故障後の系内数モデルへの変換行列 $\boldsymbol{\Phi}^{(N)}$ を求める手順を示す．故障前の系内数モデルのすべての状態 $\boldsymbol{n}$ について以下の場合分けを行い，$\phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'}$ を求める．

(1) $n_{n_F} \leq L^{(+)}_{n_F}$ の場合

故障が発生したノードの待ち行列に空きがあるので状態は変化しない．

$$\begin{aligned} \boldsymbol{n}' &= \boldsymbol{n} \\ \phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= 1 \end{aligned}$$

(2) $n_{n_F} > L^{(+)}_{n_F}$ の場合

故障したプロセッサで処理されていたタスクが捨てられる．

$$\begin{aligned} \boldsymbol{n}' &= (i_1, i_2, \ldots, i_{n_F-1}, L^{(+)}_{n_F}, i_{n_F+1}, \ldots, i_{N_r}) \\ \phi^{(N)}_{\boldsymbol{n},\boldsymbol{n}'} &= 1 \end{aligned}$$

最後に，故障前のタグ付きタスクモデルから故障後のタグ付きタスクモデルへの変換行列 $\boldsymbol{\Phi}^{(T)}$ を求める手順を示す．故障前のタグ付きタスクモデルのすべての状態 $\boldsymbol{u}$ について以下の場合分けを行うことにより，$\phi^{(T)}_{\boldsymbol{u},\boldsymbol{u}'}$ を求める．

(1) $l \neq n_F$ の場合

タグ付きタスクモデルが滞在しているノード以外で故障が発生したことを表す．この場合，タグ付きタスクの位置には影響はなく，ノード n_F のタスクが故障後のノード容量を満たすまで捨てられる．

$$\begin{aligned} \boldsymbol{u}' &= (i_1, i_2, \ldots, \min(i_{n_F}, L_{n_F}^{(+)}), \ldots, i_{N_r}, l, r) \\ \phi_{\boldsymbol{u},\boldsymbol{u}'}^{(T)} &= 1 \end{aligned}$$

(2) $l = n_F$ の場合

タグ付きタスクが滞在しているノードで故障が発生したことを表す．この場合は，さらに次のように場合分けを行う．

(a) $r = L_{n_F} - K_{n_F}$ の場合

タグ付きタスクが待ち行列の最後にあることを表す．この場合，タグ付きタスクは捨てられる．

$$\begin{aligned} \boldsymbol{u}' &= F \\ \phi_{\boldsymbol{u},\boldsymbol{u}'}^{(T)} &= 1 \end{aligned}$$

(b) $r = 0$ かつ $i_{n_F} > K_{n_F}^{(+)}$ の場合

タグ付きタスクが待ち行列に戻されるか，もしくは，タグ付きタスク以外のタスクが待ち行列に戻される場合の 2 種類の可能性があり，それぞれ次式で与えられる．

$$\begin{aligned} \boldsymbol{u}' &= (i_1, \ldots, \min(i_{n_F}, L_{n_F}^{(+)}), \ldots, i_{N_r}, l, 1) \\ \phi_{\boldsymbol{u},\boldsymbol{u}'}^{(T)} &= 1/K \\ \boldsymbol{u}'' &= (i_1, \ldots, \min(i_{n_F}, L_{n_F}^{(+)}), \ldots, i_{N_r}, l, 0) \\ \phi_{\boldsymbol{u},\boldsymbol{u}''}^{(T)} &= 1 - 1/K \end{aligned}$$

(c) $r = 0$ かつ $i_{n_F} \leq K_{n_F}^{(+)}$ の場合

タグ付きタスクは引続きプロセッサで処理される．

$$\begin{aligned} \boldsymbol{u}' &= (i_1, \ldots, \min(i_{n_F}, L_{n_F}^{(+)}), \ldots, i_{N_r}, l, 0) \\ \phi_{\boldsymbol{u},\boldsymbol{u}'}^{(T)} &= 1 \end{aligned}$$

(d) 上記以外の場合

タグ付きタスクは待ち行列の一つ後ろに戻される．

$$\begin{aligned} \boldsymbol{u}' &= (i_1, \ldots, \min(i_{n_F}, L_{n_F}^{(+)}), \ldots, i_{N_r}, l, r+1) \\ \phi_{\boldsymbol{u},\boldsymbol{u}'}^{(T)} &= 1 \end{aligned}$$

4.3.5 解析結果

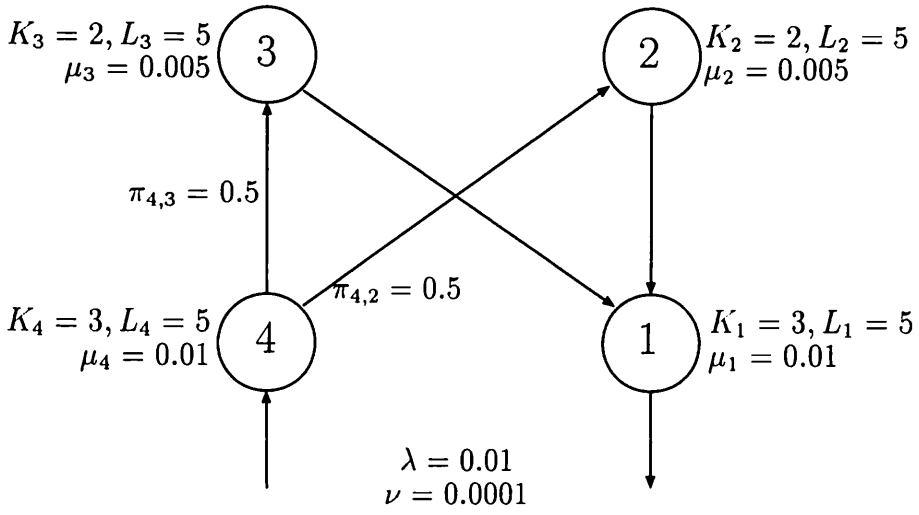
図 4.5 に示す 2 つの QN を，本方法を用いて解析を行った結果を示す．Type1 と Type2 では故障後のタグ付きタスクモデルの状態数は等しいが同値類の数が大きく異なる．これは，Type1 の場合，ノード 1,2,3 にタスクが到着するためには，必ずノード 4 を通過しなければならないため，タグ付きタスクがノード 4 で処理されている場合は，タグ付きタスクの位置が変化することなしに，ノード 1,2,3 の系内数が増加することはない．このため，タグ付きタスクがノード 4 で処理されていることを表す状態は同値類にまとめることができず，同値類の数が多くなる．一方，Type2 の場合はこのような制限がないため，タグ付きタスクの位置が等しい状態が同値類となり，Type1 に比べて同値類の数が少なくなる．

計算時間の結果を図 4.6 に示す．集約を行いながら計算する場合は，Type1, Type2 ともに，計算時間は ε に依存しており，δ にはほとんど影響されないことがわかる．$\delta = 1 \times 10^{-4}$ の場合，Type1 では $\varepsilon = 0.01$ で集約を行わない場合の約 70%，$\varepsilon = 0.1$ では約 53%の計算時間となっており，Type2 では，$\varepsilon = 0.01$ で約 66%，$\varepsilon = 0.1$ で約 48%の計算時間となっている．また，閾値が同じ場合には，同値類の数が少ない方が，集約を行いながら計算することにより計算時間が減少することがわかる．また，Type1 の場合 $\delta = 1 \times 10^{-2}$ のとき，集約を行わずに計算した方が，$\varepsilon = 0.01$ として集約を行いながら計算した場合よりも，計算時間が短くなっている．これは，集約条件を調べるための演算や集約を行うための演算が，状態を集約することによる演算の減少量よりも多くなるためである．

次に，閾値と吸収確率 $\boldsymbol{\alpha}^{(T)}{}_{\boldsymbol{u},\boldsymbol{u}'}$ の相対誤差の関係を図 4.7 に示す．図は各閾値の組合せで得られた吸収確率の相対誤差のうち最大のものを表しており，集約を行わずに $\delta = 1 \times 10^{-9}$ として計算した値を真値とみなしている．Type1 の場合，誤差は閾値 ε に依存しており，Type2 の場合は，$\delta \geq 5 \times 10^{-4}$ の範囲では δ に依

存しており，$\delta < 5 \times 10^{-4}$ の範囲では，ε に依存している．これは，δ が大きい場合は，すべての同値類が集約されないうちに 計算の終了条件が満たされてしまうためであると考えられる．

最後に，棄却率 $q_F(t)$ の計算結果を図 4.8 に示す．Type1，Type2 ともに誤差は 0.5%以内に収まっており，十分な精度で解析が行えることがわかる．

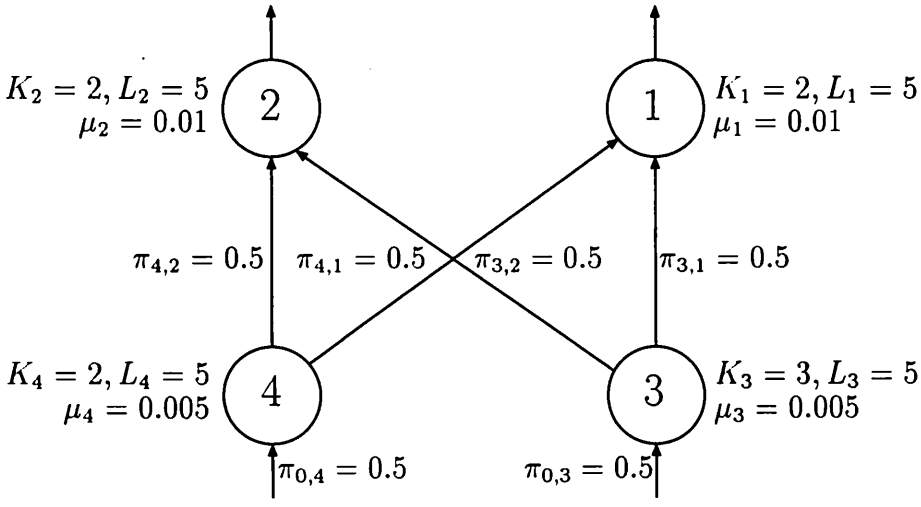


故障が発生するノード 4

故障後タグ付きタスクモデルの状態数 6914

故障後タグ付きタスクモデルの同値類数 445

(a) Type 1



$\lambda = 0.01$
$\nu = 0.001$

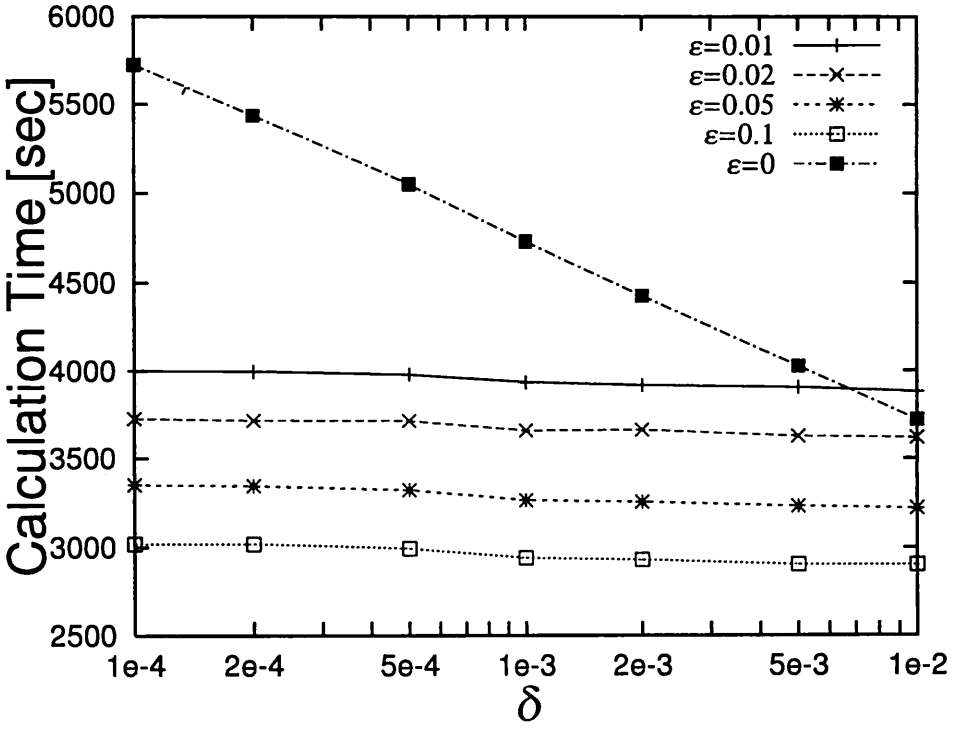
故障が発生するノード 3

故障後タグ付きタスクモデルの状態数 6914

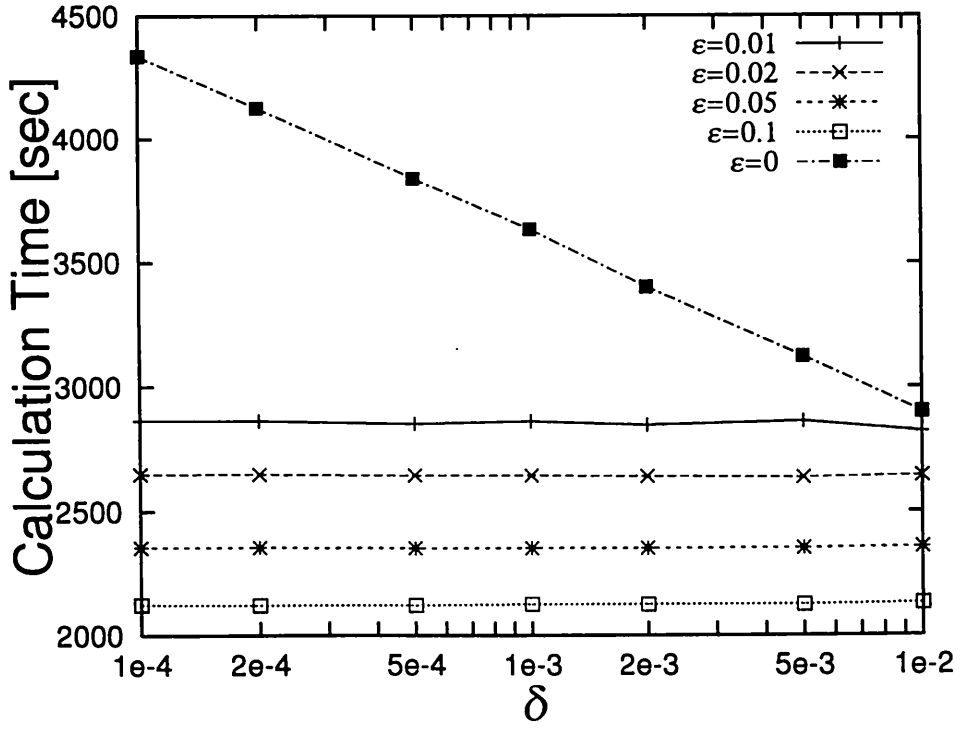
故障後タグ付きタスクモデルの同値類数 17

(b) Type 2

図 4.5: QN

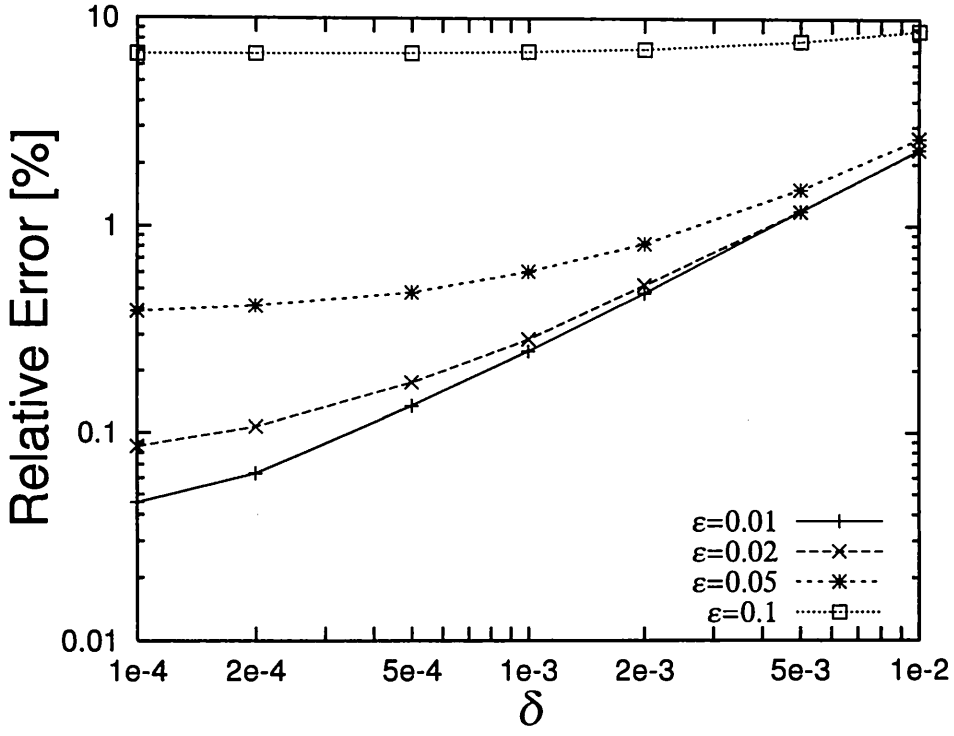


(a) Type 1

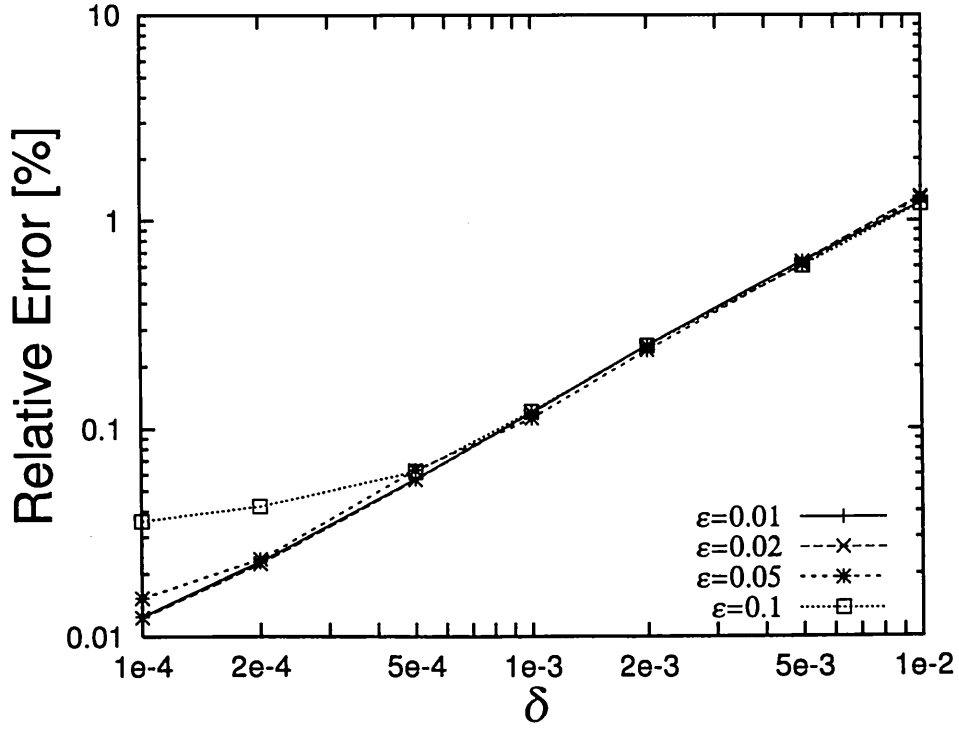


(b) Type 2

図 4.6: 閾値と $\alpha_{\mathcal{TA}}^{(T)}$ の計算時間 (CPU:R10000/195MHz)

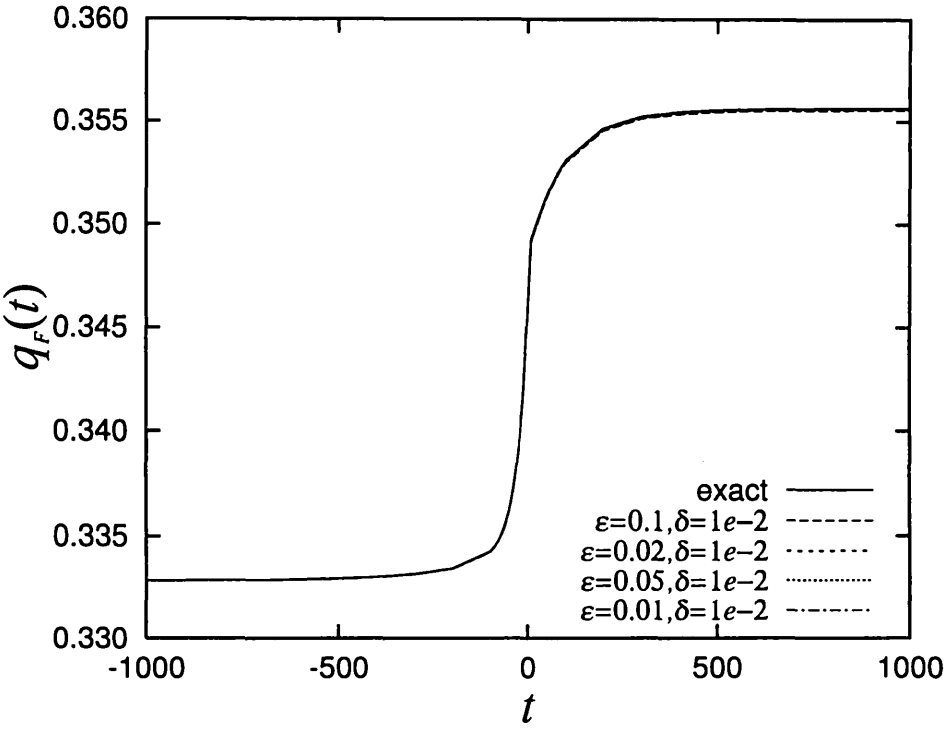


(a) Type 1

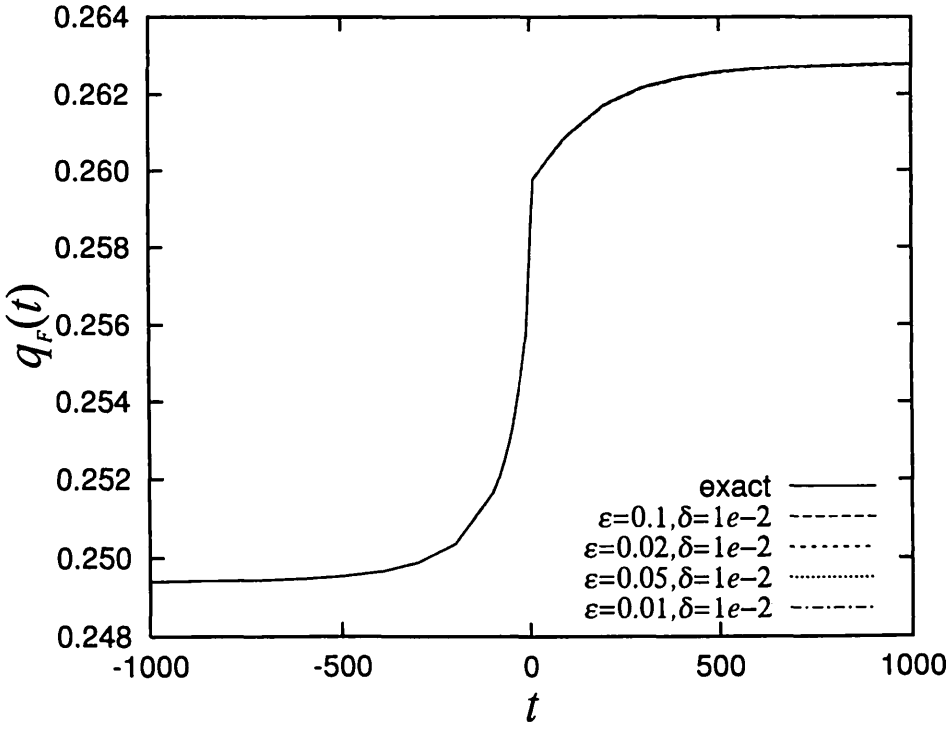


(b) Type 2

図 4.7: 閾値と吸収確率の最大相対誤差



(a) Type 1



(b) Type 2

図 4.8: 棄却率の変化

4.4 まとめ

本章では，一つの処理要素のグループを持ちタスククラスに優先度が定義されているシステム，及び複数の処理要素のグループを持つ分散リスポンシブシステムの解析例を示した．

その結果，いずれのシステムにおいても，集約を行うことによってそうでない場合に較べて約半分の時間で吸収確率を導出することができ，そのとき得られた棄却率の誤差は1%未満であった．

またこれらの例から，吸収確率の計算に要する時間と得られた棄却率の誤差は，

(1) 計算終了条件の閾値である δ が小さい場合には，集約条件の閾値 ε に大きく依存する．
(2) 閾値の組み合わせによっては，集約を行った方が計算時間が増加する場合がある．

ことを示した．

なお，ここでは解析に用いた計算機システムの制約から，タグ付きタスクモデルの状態数が562個及び6,914個と比較的状態数が少なく，計算機の実メモリのみを用いて解析を行える2種類のシステムについて解析を行った．しかし，本稿で提案した状態の同値類分割と集約に基づく処理の高速化は，解析の進行と共に必要な記憶容量を急激に減少させ得るため，仮想記憶を必要とするような大規模なシステムの解析においては，ここで示した例よりも更に高速化を図ることができるものと考えられる。

5. 結論

本論文では，リスポンシブシステム中に故障が発生した場合のシステムの動的な処理性能の変化を評価するために，過渡状態におけるタスクの棄却率を求める方法を示した．

第2章では，解析の対象とするリスポンシブシステムを示し，システム内のタスク数の変化の過程をモデル化した系内数モデルと，システムで処理される代表的なタスクの処理の過程をモデル化したタグ付きタスクモデルを定義した．第3章では，系内数モデルとタグ付きタスクモデルを用いて，過渡状態におけるタスクの棄却率を求める方法を示した．タスクの棄却率は，タグ付きタスクモデルの定常分布で表されるため，これを短時間に求めるために吸収確率を定義し，これを用いて計算する方法を示した．さらに，タグ付きタスクモデルの状態を相互到達可能性に基づく同値類に分割し，状態確率が小さくなった同値類を集約しながら計算することにより吸収確率を短時間に求める方法を示した．また，集約を行いながら，吸収確率を求める際に計算量が減少する条件を求め，本方法を適用するための基準を明らかにした．

第4章では，単一の処理要素のグループを持ち，タスククラス毎に優先度が定義されているリスポンシブシステム，及び複数の処理要素のグループを持つ分散型リスポンシブシステムの解析例を示した．その結果，いずれのシステムにおいても，集約を行うことによって，集約を行わない場合に較べて約半分の時間で吸収確率を導出することができ，その時の棄却率の誤差は1%未満に収まることを示した．

本研究で示した手法を用いることによって，リスポンシブシステムの設計段階において，システムの動的な性能を容易に評価することができるため，信頼度の高いシステムを実現することが可能になる．

今後の課題としては，システム中に故障が発生した場合のみならず，負荷が急激に変化した場合等の過渡状態における性能の解析方法の検討，閾値と誤差に関する検討，及び大規模なシステムの解析と処理の高速化等があげられる．

6. 謝辞

本研究を進めるにあたり，直接ご指導頂き，常に適切なご助言を頂いた，新潟大学大学院自然科学研究科・石井郁夫教授に感謝いたします．

直接ご指導頂き，常に適切なご助言を頂いた，新潟工科大学・角山正博教授に感謝いたします．

新潟大学工学部・仙石正和教授，牧野秀夫教授，山本正信教授，新潟大学大学院自然科学研究科・木竜徹教授ならびに，高橋俊彦助教授には貴重な時間をさいて本論文の査読をして戴き，数多くの助言を賜わりましたことを深く感謝いたします．

東京農工大・中静真助教授をはじめ，新潟大学工学部情報工学科，新潟大学大学院自然科学研究科の諸先生に感謝いたします．

また機会あるごとに，ご助言と激励を頂いた，福嶋康夫技官をはじめ，新潟大学工学部情報工学科の技官・職員の方々に感謝いたします．

また，研究を進めるにあたり，ご協力を頂いた新潟大学大学院自然科学研究科・石井研究室の学友諸氏に感謝いたします．

7. 参考文献

[1] 井原廣一, “リアルタイムシステムとは”, 情報処理, Vol. 35, No. 1, pp. 12–17, Jan. 1994.

[2] John A. Stankovic, “Misconceptions About Real-Time Computing – A Serious Problem for Nexg-Generation Systems –”, IEEE Computer, Vol. 21, No. 10, pp. 10–18 Oct. 1988.

[3] 竹垣盛一, “ハードリアルタイムシステムの応用例”, 情報処理, Vol. 35, No. 1, pp. 34–40, Jan. 1994.

[4] M.Malek, “A Consensus-Based Model for Responsive Computing”, IEICE Trans. Inf. & Syst., Vol. E76-D, No. 11, pp. 1319–1324, Nov. 1993.

[5] 角田良明, 菊野亨, “リスポンシブシステム: リアルタイムシステムとフォルトトレラントシステムの統合”, 情報処理, Vol. 35, No. 1, pp. 48–54, Jan. 1994.

[6] 中野　博隆, “ビデオオンデマンド”, 信学誌, Vol. 77, No. 8, pp. 883–885, Aug. 1994.

[7] 潮田健太郎. “分散処理構成のビデオ・サーバ, 負荷分散でストリーム単価軽減”. 日経エレクトロニクス, No. 637, pp. 115–122, Jun. 1995.

[8] B. Berthomieu and M. Diaz. “Modeling and Verification of Time Dependent Systems using Time Petri Nets”. IEEE Trans. on Software Engineer., Vol. 17, No. 3, pp. 259–273, Mar 1991.

[9] S. M. Rezaul Islam and Hany H. Ammar, “Performability Analysis of Distributed Real-Time Systems” IEEE Trans. on Comput., Vol. 40, No. 11, pp. 1239–1251, Nov. 1991.

[10] Trivedi K.S. Muppala J.K. and Woolet S.P. "Real-Time-Systems Performance in the Presence of Failsures". IEEE Computer, Vol. 24, No. 5, pp. 37–47, May 1991.

[11] Wei Zhao and John A. Stankovic. "Performance Analysis of FCFS and Improved FCFS Scheduling Algorithms for Dynamic Real-Time Computer Systems", Proc. of The IEEE 10th Real-Time Symtems Symp., pp. 156–165, 1989.

[12] John F. Meyer, "On Evaluating the Performability of Degradable Computing Systems", IEEE Trans. on Comput., Vol. C-29, No. 8, pp. 720–731, Aug. 1980.

[13] R.M.Smith, K.S.Trivedi and A.V. Ramesh, "Performability Analysis: Measures, an Algorithm, and a Case Study", IEEE Trans. on Comput., Vol. C-37, No. 4, pp.406–417, Apr. 1988.

[14] M. Akber Qureshi and William H. Sanders, "A New Methodology for Calculating Distributions of Reward Accumulated During a Funite Interval", Proc. of The 26th Annual International Symposium on Fault-Tolerant Computing, pp. 116–125, Jun. 1996.

[15] Hermann de Meer and Hans Mauser, "A Modeling Approach for Dynamically Reconfigurable Systems", Proc. of 2nd International Workshop on Responsive Computer Systems, Oct. 1992.

[16] John F. Meyer, "Closed-Form Solutions of Performability", IEEE Trans. on Compt., Vol. C-31, No. 7, Jul. 1982.

[17] Rolf B. Haugen and Even Skogan, "Queueing Systems with Stochastic Time Out", IEEE Trans. on Comm., Vol. COM-28, No. 12, Dec. 1980.

[18] Benjamin Melamed and Micha Yadin. "Randomization Procedures in the Computation of Cumulative-Time Distributions over Discrete State Markov Processes". Operations Research, Vol. 32, No. 4, pp. 926–944, July-Aug, 1984.

[19] P. J. Schweitzer, "AGGREGATION METHOD FOR LARGE MARKOV CHAIN", Mathematical Computer Performance and Reliability, pp. 275–286, 1984.

[20] P. Semal, "Refinable Bounds for Large Markov Chains". IEEE Trans. on Comput., Vol. 44, No. 10, pp. 1216–1222, Oct. 1995.

[21] K. S. Trivedi and A. Bobbio, "An Aggregation Technique for the Transient Analysis of Stiff Markov Chain", IEEE Trans. on Comput., Vol. C-35, No. 9, pp. 803–814, Sept. 1986.

[22] Barry L. Nelson, "STOCHASTIC MODELING – Analysis and Simulation –", McGraw-Hill, Inc., 1995.

[23] E. Gelenbe, I. Mitrani(秋丸春男, 橋田温監訳), "計算機システムの解析と設計", オーム社, 1988.

[24] R.A.ハワード, "ダイナミックプログラミングとマルコフ過程", 培風館, 1971.

[25] 宮沢政清, "確率と確率過程", pp.131–149, 近代科学社, 1993.

8. 著者の論文

[1] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, 内藤 祥雄, “リスポンシブシステムにおけるプロセッサ故障時の解析方法について”, 信学論(D-I), Vol.J78–D–I, No.8, pp.716–723, Aug. 1995.

[2] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “タグ付きタスクモデルの一解法について”, 信学論(D-I), Vol.J80–D–I, No.10, pp.836–844, Oct. 1997.

[3] Hiroei IMAI, Masahiro TSUNOYAMA, Ikuo ISHII, and Hideo MAKINO, “Analysis of Responsive Systems at Processor Faults”, Proceedings of the 1995 Pacific Rim International Symposium on Fault-Tolerant Systems, pp.716–723, Dec. 4-5 1995.

9. 研究会報告等

[1] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, 内藤 祥雄, “リアルタイムシステム内の故障とタスクの laxity について”, 信学技報, FTS93-62, pp.55–62, Feb. 1994.

[2] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “リスポンシブシステムにおけるプロセッサ故障時の解析方法について”, 信学技報, FTS95-16, pp.39–46, Apr. 1995.

[3] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “タグ付きタスクモデルのためのマルコフ連鎖の一解法”, 信学技報, FTS96-52, pp.33–40, Dec. 1996.

[4] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “タグ付きタスクモデルによる待ち行列ネットワークの一解法”, 信学技報, FTS97-70, pp.55–62, Dec. 1997.

10. 口頭発表等

[1] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, 内藤 祥雄, “リアルタイムシステムにおけるプロセッサ故障と応答時間”, 電子情報通信学会春季大会, pp.6-244. March 1993.

[2] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, 内藤 祥雄, “リアルタイムシステム内のタスクが laxity を越える確率”, 電子情報通信学会信越支部大会, pp.193–194, Oct. 1993.

[3] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “分散型リアルタイムシステムのモデル化について”, 電子情報通信学会信越支部大会, pp.467–468, Oct. 1995.

[4] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “状態の集約に基づく大規模マルコフ連鎖の一解法”, FTC 研究会, Jul. 1996.

[5] 今井 博英, 角山 正博, 石井 郁夫, 牧野 秀夫, “タグ付きタスクモデルによる待ち行列網の一解法”, 電子情報通信学会信越支部大会, Oct. 1997.

付 録A　QN のタグ付きタスクモデルの同値類分割方法

はじめに，待ち行列の外部からノード $v_n(n=1,2,\ldots,N_r)$ に至るすべての経路に含まれるノードの集合 U_n を求める．

手順 A.1 (U_n の導出)

(1) $U_n \leftarrow \emptyset(n=1,\ldots,N_r)$, $n \leftarrow 1$

(2) QN の外部入力からノード v_n に至るすべてのパスに含まれるノードの番号を求め，$n_1,n_2,\ldots$ と表す．このとき，$U_{n_1},U_{n_2},\ldots$ にノード v_n を追加する．

(3) $n \leftarrow n+1$ とし，$n \leq N_r$ ならば (2) に戻る．

つぎに，U_n を用いて状態集合 $\mathcal{S}^{(T)}$ を相互到達可能性に基づく同値類に分類する．QN はフィードフォワード網であることから，状態 $\boldsymbol{u}=(i_1,i_2,\ldots,i_{N_r},l,r)$ について，l もしくは r が異なる状態は互いに到達可能とはならない．l と r が等しい状態の集合を $\mathcal{S}^{(T)}_{(l,r)}(l=1,2,\ldots,N_r;\ r=0,1,\ldots,L_l-K_l)$ で表す．

$\mathcal{S}^{(T)}_{(l,r)}$ に含まれる 2 つの状態 $\boldsymbol{u}=(i_1,i_2,\ldots,i_{N_r},l,r)$，$\boldsymbol{u}'=(i'_1,i'_2,\ldots,i'_{N_r},l,r)$ について，l と r が変化することなしに $i_n(n=1,2,\cdots,N_r)$ が増加ならびに減少することが可能ならば，$\boldsymbol{u}$ と $\boldsymbol{u}'(i_n \neq i'_n)$ は互いに到達可能となる．

ここで，ノード v_n のタスク数が減少し得るのはノード v_n のタスクが制限時間を満たせずに捨てられる場合である．一方，l と r が変化せずに，ノード v_n のタスク数が増加し得るのは次の 2 つの条件を満す場合に限られる．

(1) U_l に v_n が含まれていない場合．

この場合には，QN の外部から v_l を通らずにノード v_n に至るパスが存在する．外部から新たに到着したタスクがこのパスを通って処理されることにより，l と r が変化せずに v_n の系内数が増加し得る．

(2) $r=0$ かつ $K_l>1$.

この場合には，ノード v_l が複数のプロセッサを持つため，タグ付きタスク以

外のタスクが v_l での処理を終え，次のノードへ移ることができる．v_n が U_l に含まれる場合でも l と r が変化せずに v_n の系内数が増加し得る．

$\mathcal{S}^{(T)}_{(l,r)}$ を，条件 (1) と (2) を基に分類することによって，同値類を求めることができる．