論　文

リスポンシブマルチプロセッサシステムのモデル化と性能評価

学生員　土屋　達弘[†]　　正　員　角田　良明[†]　　正　員　菊野　亨[†]

Modeling and Evaluation of Responsive Multiprocessor Systems

Tatsuhiro TSUCHIYA[†], *Student Member*, Yoshiaki KAKUDA[†] *and* Tohru KIKUNO[†], *Members*

あらまし　リスポンシブシステムはフォールトトレラントシステムとリアルタイムシステムを統合したシステムであり，リアルタイム性を考慮した柔軟な障害回復機能を備えることが要求される．稼動中のプロセッサの数がプロセッサの故障・修復によって増減するマルチプロセッサシステムで，タスクのリアルタイム処理機能の他に，故障プロセッサのシステムからの隔離，修復されたプロセッサのシステムへの結合という再構成機能を備えたものは，リスポンシブシステムの一例とみることができる．一般に，システム再構成の作業のために通常のタスク処理が中断されることがあり，リアルタイム処理を要求されるタスクがデッドライン違反を引き起こす可能性がある．本論文では，この可能性を陽に考慮したシステムの故障・修復の振舞い，および，タスクの到着率の変化といったシステムの外部環境の変動を表現するためのモデル化手法を確率リウォードネット上で提案する．更に，提案したモデルを用いた性能評価を通して，システム再構成の実行のタイミングを制御することで再構成に伴う影響を最小に抑えて，システム性能の向上が達成できることを示す．

キーワード　リスポンシブシステム，フォールトトレラントシステム，リアルタイムシステム，再構成，確率リウォードネット

1. まえがき

生産工場のオートメーションシステムや銀行のオンラインシステムといったリアルタイムな処理が必要とされるシステムの多くは，処理の時間的な制約を満たす能力と同時に，高い信頼性をも要求される．このようなシステムの発展，用途の拡大に伴い，従来は独立に行われていた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設計を統合することへの要求が高まってきている．Malek により提唱されたリスポンシブシステムは，このようなリアルタイム処理技術と障害回復技術とを統合した機能をもつシステムとして定義される[(4),(6),(7)]．

より具体的には，リスポンシブシステムの特徴は，障害に対するリアルタイム性を考慮した回復処理にある．我々は既に，メッセージの転送エラーや通信の遅延によってプロセス間の整合が失われた際に，例外処理ルーチンを用いずに正常な状態にリアルタイムに移行するリスポンシブプロトコルを提案し，その設計法についての議論を行っている[(3),(5)]．本論文ではリアルタイム処理が要求されるマルチプロセッサシステムについて，リスポンシブシステムの見地からそのモデル化および性能評価について議論する．

本論文では，故障したプロセッサをシステムから分離する再構成と，修理されたプロセッサをシステムへ復帰させる再構成の2種類の再構成によって，信頼性を確保するマルチプロセッサシステムを考える．このような再構成可能なフォールトトレラントシステムの評価には，処理性能と信頼性を統合した概念であるパフォーマビリティが広く用いられている[(11)]．通常，パフォーマビリティに関する尺度の計算はいわゆるマルコフリウォードモデルを用いて行われる[(14)]．リアルタイムシステムの評価にパフォーマビリティを導入した研究としては，例えば文献(9),(12)がある．

一方，リアルタイムシステムで実行されるタスクはデッドライン以内に完了することが要求され，その違反がシステムの信頼性に与える影響に応じてシステムは次のように分類される．まず，タスクのデッドライン違反がシステム全体の障害に直結する場合は

† 大阪大学基礎工学部情報工学科，豊中市
Faculty of Engineering Science, Osaka University, Toyonaka-shi, 560 Japan

ハードリアルタイムシステムと呼ばれ，航空機の制御等に用いられている．ハードリアルタイムシステムはNMR(N-Modular Redundancy) に代表される非常に冗長な空間的な資源の利用により信頼性を確保しており，構成要素の故障の検出，隔離といった一連の再構成の処理を，通常の処理への影響なしに行う．一方，デッドライン違反が直ちにはシステム全体の障害となることはない場合をソフトリアルタイムシステムと呼ぶ．ここでの議論では再構成の作業のために通常のタスク処理の中断を伴う，より一般的なソフトリアルタイムシステムを仮定する．

再構成の作業を詳細に見ると，プロセッサへのタスクの再割当て，プロセッサの同期，データの整合性の確保などの処理が含まれている．再構成の作業によって通常のタスク処理を中断すると，タスクのデッドライン違反を引き起こす可能性がある．文献(12)では，再構成の実行が引き起こすタスクのデッドライン違反は考えられていない．従って修復されたプロセッサは，修復後は直ちにシステムへの復帰が行われることになる．これに対し文献(9)では，再構成の作業がもたらすタスクのデッドライン違反を考慮したモデルを提案している．更にそのモデルを用いて，修復されたプロセッサを復帰させるか否かをミッション終了までの時間の長さに応じて動的に決定することによって，システム性能を向上させることができることを示している．しかし，ミッションの長さがあらかじめ正確に決められている場合にしか適用できない，有効に作用するミッションの長さが限られてくる，等の問題がある．

また，これらの研究ではシステム外部の環境が不変であることを暗に仮定している．しかし，例えば工場のオートメーションシステムで新しい工程が立ち上がるときには，非周期的なタスクが大量に発生するといったことが考えられる．従ってシステムの正確な評価には，このような外部の環境の変化を包含したモデル化が不可欠である．

本論文では，システム外部の環境の変化を許す場合について，再構成の実行によるタスク処理への影響を考慮したモデル化と性能評価法を示す．システム外部の環境が変化する場合には，修復の済んだプロセッサのシステムへの復帰のための再構成が仮に可能であっても，より影響の低い環境になるまであえてその実行を延期するという方法が，修復後に直ちに再構成を行う方法に比べより有効となることが期待できる(文献(10),(13)は，既に，フォールトトレラントシステムにおけるこのような可能性について言及している)．提案するモデル化ではそのような再構成の実行法が記述できるようにする．更に，具体的な評価尺度を計算して再構成の実行のタイミングを適切に定めることでシステム性能の向上が可能であることを示す．

従来の代表的モデルであるマルコフリウォードモデルでは状態での滞在時間に応じてリウォードを累積するため，システムの故障・修復に要する時間間隔に比べて瞬間的な短い時間の作業である再構成の影響を適切に評価することができなかった．更に，状態数が非常に大きくなるため記述が困難であるという問題点があった．我々は既に，(状態での滞在時間ではなく)状態間での遷移の発生に対応してリウォードを累積するように拡張したマルコフリウォードモデルを提案している[15]．この拡張によって再構成の影響を評価することが可能になったが，状態数が大きくなるという問題点は避けられない．今回のモデル化にあたってはペトリネットの一種である確率リウォードネット(stochastic reward net)[1]を用いる．確率リウォードネットはペトリネット特有の簡潔な表現が可能であるので，拡張マルコフリウォードモデルに比べかなり記述しやすくなっている．なお，拡張マルコフリウォードモデルと同様，状態(マーキング)間の遷移に応じてリウォードを累積することによって再構成の影響を評価することができる．

本論文の構成は次のとおりである．**2.** では本論文で対象とするマルチプロセッサシステムとその外部環境について述べる．**3.** では確率リウォードネットを用いた新しいモデル化を提案する．**4.** ではモデルに基づいて性能評価を行い，典型的な事例について再構成を実行する最適なタイミングが求まることを示す．

2. システムと外部環境

本論文で対象とするマルチプロセッサシステムについて説明する．初期状態においてシステムは，疎結合された n 台のプロセッサから構成される．稼動中のプロセッサに故障が発生した場合，再構成によってそのプロセッサはシステムから隔離される．またシステムは，プロセッサの故障の際にデータの整合性が失われないようにする等価化処理機構をもつものとする．故障したプロセッサは修復を受けて，修復の完了後に再構成によってシステムに復帰させることができる．再構成の実行時には，その作業のため通常の処理は中断されるものとする．従って，その中断中にシステムに

到着したタスクは失われる．また，プロセッサは1台ずつ修復されるものとし，そのために必要な資源はシステムに復帰するまでプロセッサによって占有されるとする．バス等のプロセッサ以外のシステムの構成要素は故障しないものと仮定する†．タスクの到着は非周期的であって，ポアソン分布に従うものとする．タスク相互には順序に関する依存関係はない．各タスクは到着してから時間 d 以内に処理を完了することが求められており，この制約がいわゆるデッドラインと呼ばれているものである．タスクが到着したとき，空きプロセッサが存在すれば必ずそのプロセッサにタスクは割り当てられ，処理される．一方，空きプロセッサがなければ少なくとも1台のプロセッサが解放されるまで待たされる．タスクの処理時間はパラメータ μ の指数分布に従うものとする．

以上の仮定から，システムが m 個の正常なプロセッサで稼動している場合，システムは $M/M/m$ 待ち行列システムとみなせる．但し，m が小さくなってシステム全体の処理能力 $m\mu$ がタスクの到着率を下回るような過負荷の状況ではリアルタイム処理を継続することができない．このような状況のシステムに到着したタスクは，再構成の実行時と同様，失われる．従って，システムがこのような過負荷な状況にあるとき，システムは障害状態にあると考える．

次に，システムの外部環境の変化について述べる．正確な性能評価を行うにはタスク量の変化などに代表されるシステムの外部環境の変化をモデルにとり入れる必要がある．本論文では，マルコフ連鎖で近似できる外部環境の変化について考える．ここでは例を用いて外部環境の変化とそのモデル化について直観的な説明をする．今，センサから送られる値によってプラントを監視，管理する次のようなシステムを考える．まずプラントが安定して動作している場合，特別の処理は必要ないのでシステムの負荷は低く保たれる．次にセンサからの値がときどきあるしきい値を超える場合，システムはプラントの動作が不安定になっていると認識し，より頻繁にセンサの値を調べ，監視を強める．センサからの値の変動が長期化した場合，システムはプラントが異常な状態にあるものとみなし，正常な状態に戻すような制御を行う．このようなプラントの変化はランダムに起こると考えられるので，図1 のマルコフ連鎖で表される (最終的には，図1のマルコフ連鎖の表現も後述の図3の (右側) のネット表現に変換する)．図1において円は状態を，円内の数字 $i(i=1,2,3)$ は

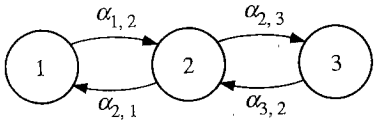

図1　外部環境の変化
Fig. 1　Changes of the environment.

状態名をそれぞれ表す．ここでは状態1,2,3は，それぞれプラントが異常な状態，監視の強化が必要な状態，正常な状態に対応しているものとする．また，矢印は遷移を表しており，$\alpha_{i,j}$ は状態 i から j への推移確率 (transition rate) を表している．状態 i でのタスクの到着率を λ_i とおく．タスクの到着率が外部環境に応じて異なるものとし，一般には $\lambda_i \neq \lambda_j (i \neq j)$ が成立する．パラメータ $\alpha_{i,j}, \lambda_i$ の具体的な値はアプリケーションに依存して決まる．

3. 確率リウォードネットによるシステムのモデル化

3.1 確率リウォードネット

本論文ではシステムのモデル化にペトリネットの一種である確率リウォードネット (stochastic reward net)[1] を用いる．(図2，図3参照)．確率リウォードネットは一般化確率ペトリネット (generalized stochastic Petri net) の各マーキングと各トランジションに，リウォードレートとリウォードインパルスをそれぞれ割り当てたものである．マーキングの変化に応じてリウォードレートとリウォードインパルスを累積していった値を用いてシステムの性能評価を行う．

確率リウォードネットのトランジションは，発火に要する時間が0の即時発火トランジションと，割り当てられた発火率 (firing rate) をパラメータとする指数分布に従う時間トランジションの2種類に分けられる．これら2種類のトランジションを，図2と図3では，それぞれ細い線と長方形で表現している．各トランジション t に対してはマーキング M を引数として真，偽の値を返す2値関数である発火可能関数 (enabling function) $e_t(M)$ が定義されており，$e_t(M) =$ 真 の場合のみ t が発火可能となる．本論文では特に断らない限り，任意のトランジション t の発火可能関数 e_t

† これらの仮定は本質的なものではない．本論文の仮定と異なる修復方針，およびプロセッサ以外の構成要素の故障・修復などは，本論文のシステムの故障・修復モデル (図2) を拡張・修正することで表現可能である．

は，任意のマーキング M について $e_t(M)=$真 とする．発火に関しては即時発火トランジションが時間トランジションよりも優先権があるものとし，即時発火トランジションは発火可能になれば必ず瞬時に発火する．一方，時間トランジションの発火率はマーキングの関数として与えられる．

確率リウォードネットの各マーキング M に割り当てられるリウォードレートを $\rho(M)$ で表す．リウォードの計算にあたってはマーキングにとどまった時間に応じてリウォードレートを累積する．また各トランジション t のリウォードインパルスは t の発火ごとに累積する．t のリウォードインパルスを，t が発火した際のマーキング M を引数として $r_t(M)$ で表す．

トランジションの発火に伴うマーキングの変化は確率過程 $\{(M_n,\tau_n,\theta_n), n=0,1,2,...\}$ とみなすことができる．ここで，M_n は初期マーキング M_0 から開始して n 番目に到達したマーキング，τ_n は n 番目に発火したトランジション(つまり，M_n において τ_n が発火して M_{n+1} になる)，θ_n は τ_n の発火時間である．時刻 θ におけるマーキングを $M(\theta)$ とすると，時刻0から時刻 θ までの間に累積されたリウォード $Y(\theta)$ は次のように定義される．

$$Y(\theta)=\int_0^{\theta}\rho(M(u))du+\sum_{\{n|\theta_n\leq\theta\}}r_{\tau_n}(M_n)$$

$Y(\theta)$ の期待値 $E[Y(\theta)]$ を利用すると，システムの性能評価を定量的に行うことができる．$E[Y(\theta)]$ の具体的な計算法は文献(1)で示されており，本論文でもこの計算法を用いる．

3.2 故障・修理モデル

文献(12)では再構成に伴うタスクのデッドライン違反を考慮していないので，故障したプロセッサの修復が完了すると直ちにそのプロセッサはシステムに復帰させられている．すなわち，プロセッサの状態は動作しているか，故障しているかの2通りに分けられている．しかし再構成によって通常のタスク処理が中断される場合，修復されたプロセッサをシステムに復帰させることによってシステムの性能と信頼性が低下することがある[8]．そこで本論文では，各プロセッサの状態を1)正常に稼動している状態，2)故障していて修復が未完了の状態，3)修復が完了しており復帰を待っている状態，の三つに分けて考える．

提案するシステムの故障・修復モデルの確率リウォードネットを用いた記述例を図2に示す．図2のネット表現で，各トークンは個々のプロセッサを表現しており，プレース $pup, pdn, prep$ がプロセッサの上述の三つの状態にそれぞれ対応している．従って，図2のマーキングは6台のプロセッサのうち3台が正常に稼動中であり，2台が故障していて現在修復中もしくは修復待ちであり，残りの1台が復帰待ちであることを表している．以下では，マーキング M においてプレース p にあるトークンの総数を $\#(M,p)$ で表す．

トークンはネット上を次のように移動する．プロセッサの故障は時間トランジション $tfail$ によって表現されており，$tfail$ の発火によって pup 中の一つのトークンが pdn へと移動する．各プロセッサの故障率を γ とおけば，マーキング M における $tfail$ の発火率は，稼動中のプロセッサ数($\#(M,pup)$)と故障率(γ)の積，すなわち $\#(M,pup)\cdot\gamma$ となる．故障したプロセッサはシステムから切り離され，修復を受ける．修復の完了は時間トランジション $trep$ に対応しており，これが発火すると pdn 中の1個のトークンが $prep$ に移動する．$prep$ 中のトークンは修復が完了して，システムへの復帰を待っているプロセッサに対応している．図2の場合には $prep$ から $trep$ へ抑止アークがあり，これによって $prep$ にトークンがない場合にのみ $trep$ が発火可能となる．このことはプロセッサの修復が1台ずつ行われるということを表現している†．またこの修復の方針により，$trep$ の発火率はマーキングに関係なく修理率δ として定められ，一定である．修復が完了したプロセッサをシステムへ復帰させるための再構成は，図2の場合には即時発火トランジション $tret$ で表されている．

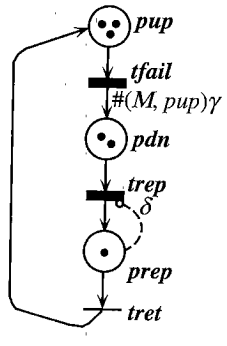

図2 故障・修復モデル
Fig. 2 Failure-repair behavior of the system.

† $trep$ の発火可能関数 e_{trep} を $\#(M,prep)=0$ の場合のみ真となるよう定めることで，抑止アークを用いない表現も可能である．

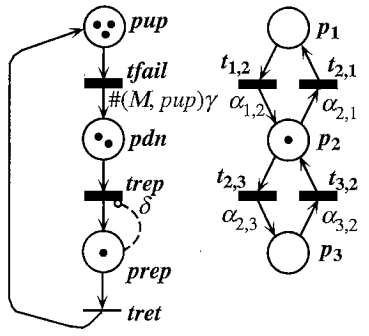

図3　故障・修復モデルへの外部環境の変化の導入
Fig. 3　Unified model.

1. で述べたように，リスポンシブルマルチプロセッサシステムの性能は，プロセッサ復帰のための再構成を行うタイミングの決め方に大きく左右される．そこで再構成のタイミングを $tret$ の発火可能関数 e_{tret} で記述する．図2のモデルには外部環境の変化が表現されておらず，それに応じた再構成の実行は記述できない．その具体的な記述については，次の **3.3** で説明する．

3.3　再構成実行の条件の記述

3.2 で示したシステムの故障・修理モデルにより，可用性(アベイラビリティ)や信頼性(リライアビリティ)と言った古典的な信頼性の評価尺度を得ることができる．しかし，本論文で対象とするシステムでは外部環境の変化を仮定しており，そうした仮定のもとで信頼性だけでなく，実行されたタスクの量に関する尺度でもあるパフォーマビリティを得るためには，外部環境の変化を(図2のモデルに)導入することが必要になる．この拡張を行ったモデル上で外部環境の変化に応じた再構成の実行のための条件を記述することで，最適な実行条件の導出が可能になる．

図3の確率リウォードネットはこの拡張されたモデルの記述例を表している．図3の記述は二つの部分から構成されており，左側の部分がシステムの故障・修復のモデル(図2と同じ)であり，右側の部分が外部環境の変化を表現している．この右側のネット表現は先に示した図1のマルコフ連鎖に対応している．図3の3個のプレース p_1, p_2, p_3 はそれぞれ図1の状態1, 2, 3に対応し，トークンの存在するプレースが現在の外部環境を表す．トランジション $t_{1,2}, t_{2,1}, t_{2,3}, t_{3,2}$ の発火によって外部環境の変化を表現する．これらのトランジションの発火率はそれぞれ $\alpha_{1,2}, \alpha_{2,1}, \alpha_{2,3}, \alpha_{3,2}$ である．

再構成の実行はタスク処理の中断を伴うため，その影響を考慮して再構成のタイミングをうまく制御することによってシステムの性能を最適化することが必要になる．例えば，処理すべきタスクがより少なくなるまで再構成を遅らせることで，再構成によるタスク処理の中断の影響をより小さく抑えることができる．更に，稼動中のプロセッサの数を増やすことはシステム全体での故障率を増加させることにもなるので，同時に稼動するプロセッサ数の上限についても考察する必要がある．

ここでは再構成の実行の条件を $\boldsymbol{\varepsilon}=(\varepsilon_1, \varepsilon_2, \varepsilon_3)$ で表現する．ここで各 $\varepsilon_i (i=1,2,3)$ は再構成の実行のための外部環境の状態 i に関するしきい値を表している．すなわち，外部環境が状態 i にあるとき，稼動中のプロセッサの総数が ε_i 以上であれば，仮に修復されたプロセッサが存在していても復帰のための再構成は行われない．逆に，稼動中のプロセッサの総数が ε_i 未満であれば直ちに再構成を実行する(もちろん，再構成を待っているプロセッサがシステム上に存在しなければ実行しない)．また，外部環境が変化するまで再構成を遅らせることは，異なる i, j に対して $\varepsilon_i \neq \varepsilon_j$ とすることで表現する．例えば，$\boldsymbol{\varepsilon}=(9,10,10)$ で稼動中のプロセッサの総数が9台であった場合，外部環境が状態1にあれば修復済みのプロセッサが存在しても復帰は行われない．しかし，外部環境の状態が1から2へ移行したなら再構成が直ちに実行される．

このプロセッサ復帰のための再構成の実行条件 $\boldsymbol{\varepsilon}$ は次のようにして図3のモデルに導入する．図3のネット上で復帰のための再構成はトランジション $tret$ に対応している．そこで $tret$ の発火可能関数 e_{tret} を次のように定めることによって条件 $\boldsymbol{\varepsilon}=(\varepsilon_1, \varepsilon_2, \varepsilon_3)$ をこのモデルに導入することができる．

$$\begin{aligned} e_{tret}(M) = &((\#(M,pup) < \varepsilon_1) \wedge (\#(M,p_1) = 1)) \\ &\vee((\#(M,pup) < \varepsilon_2) \wedge (\#(M,p_2) = 1)) \\ &\vee((\#(M,pup) < \varepsilon_3) \wedge (\#(M,p_3) = 1)) \end{aligned}$$

最後に，リウォードレートとリウォードインパルスをどのように与えるかという問題が残っている．一般に，性能評価の評価尺度の計算にあたっては，その尺度ごとにリウォードレートとリウォードインパルスを適切に定めることが必要になる．具体的な評価尺度の計算については **4.** で述べる．

表1 パラメータの値
Table 1 Values of parameters.

n	10	λ_1	30 per second
γ	0.001 per hour	λ_2	20 per second
δ	0.1 per hour	λ_3	10 per second
μ	7.0 per second	d	1.0 second
$\alpha_{1,2}$	0.1 per hour	$\alpha_{2,1}$	0.1 per hour
$\alpha_{2,3}$	0.1 per hour	$\alpha_{3,2}$	0.1 per hour

4. 性能評価

本節では，提案したモデルの各種パラメータに具体的に値を代入して，いくつかの代表的な評価尺度についての計算を試みる．用いたパラメータの値を表1に示す．l を再構成の作業の平均時間とし，$l = 0, 10, 30$ 秒の場合について計算を行った．その際，システムの動作開始(時刻 $\theta = 0$)のとき，プロセッサはすべて正常であり，外部環境は状態1にあると仮定した．評価尺度の値は累積されたリウォードの期待値 $E[Y(\theta)]$ を計算することによって時間 θ の関数として得られる．また，十分に大きい θ の値に対して，評価尺度の値を最適化する再構成の実行条件 ε を導く．

4.1 デッドライン違反のタスク数

ここでは，単位時間当りのデッドライン違反のタスク数(すなわちデッドライン以内に処理を終えることができなかったタスクの数)を評価してみる．次に述べるようにリウォードレートとリウォードインパルスを定めれば，$E[Y(\theta)/\theta]$ の値が求めるべきデッドライン違反のタスクの平均数を与えることになる．ここで，$E[Y(\theta)]$ は時刻 θ までにデッドライン違反をしたタスクの総数の期待値に等しい．

まず，再構成が実行されることなくシステムが安定している場合のデッドライン違反のタスク数をリウォードレートで表すことを考える．リウォードレートはマーキングにとどまった時間に応じて累積されるから，各マーキングが表すシステム構成と外部環境における単位時間当りのデッドライン違反のタスク数の期待値を，そのマーキングのリウォードレートとすればよい．従って，(リウォードレート) = (タスクの到着率λ) × (タスクがデッドライン違反をする確率) とおく．タスクがデッドライン違反をする確率を求めるにはレスポンスタイムの分布が必要となるが，$M/M/m$ 待ち行列システムについては解法が既に知られている[(2)]．

各タスクのデッドラインを到着時間から $d = 1.0$ 秒以内とした場合について，この解法を用いて，稼動中のプロセッサ数とタスクのデッドライン違反の確率との相関を表す図4を得た．なお，稼動中のプロセッサ数 i が小さくて $\lambda > i\mu$ が成り立つ場合は，システムの処理能力がタスクの到着率を下回っており，システムはリアルタイムな処理を継続することができない障害状態にある．障害状態のシステムに到着したタスクは失われると考えるので，タスクがデッドライン違反をする確率は1となり，リウォードレートはタスクの到着率に等しくなる．

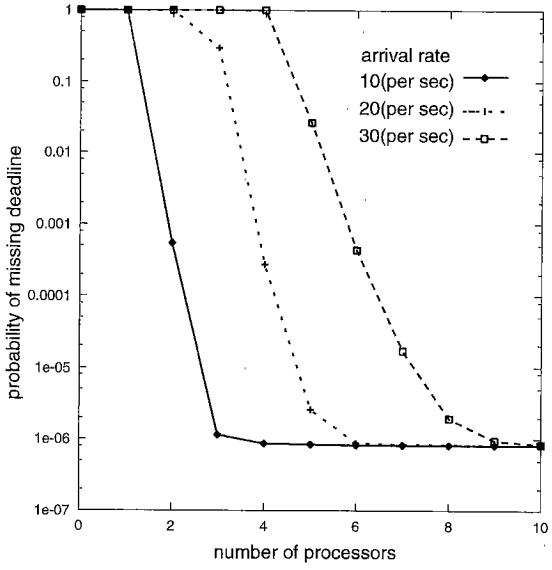

図4 デッドライン違反の確率
Fig. 4 Probability of missing deadline.

次に，再構成を実行したときの影響を受けてデッドライン違反をするタスクの数をリウォードインパルスで表すことを考える．リウォードインパルスはトランジションの発火回数に応じて累積されるため，再構成のように瞬間的な事象の評価には非常に適している．プロセッサの故障と復帰による再構成は図3ではそれぞれトランジション $tfail$ と $tret$ の発火に対応しているので，それぞれの再構成を1回実行した際にその影響でデッドライン違反をするタスクの期待数を $tfail$ と $tret$ のリウォードインパルスとすればよい．ここでは再構成による処理の遅延は平均 l 秒かかると仮定しているので，$tfail$ と $tret$ のリウォードインパルス r_{tfail}, r_{tret} は次のようになる†．

† 再構成実行の際にシステム中に存在するタスクもこの値の中に含まれると考える．

$$r_{tfail}(M) = r_{tret}(M) = \begin{cases} l\lambda_1 & (\#(M,p_1)=1) \\ l\lambda_2 & (\#(M,p_2)=1) \\ l\lambda_3 & (\#(M,p_3)=1) \end{cases}$$

再構成の作業時間 $l=10$ と $l=30$(秒)のそれぞれの場合について，上で説明したようにリウォードレート，リウォードインパルスを割り当て，可能なすべての条件εに対して $E[Y(\theta)/\theta]$ を計算した．その結果，$l=10$ のときの最適な再構成の実行条件が $\varepsilon=(7,8,10)$ であり，$l=30$ の場合の最適な再構成の実行条件が $\varepsilon=(7,7,9)$ であることがわかった．図5には，$l=10$ と $l=30$ のそれぞれの場合について，最適な場合と$\varepsilon=(10,10,10)$ の場合の2通りの結果を示してある．また，$l=0$ で再構成の影響がない場合，最適な実行条件は$\varepsilon=(10,10,10)$ であり，その結果についても図5に示している．

これらの結果から，l が大きいほど，すなわち再構成の影響が大きいほど，再構成の制御の効果が増加することがわかった．特に，$l=30$ の時に，修復したプロセッサを必ず直ちにシステムに復帰させる方法 ($\varepsilon=(10,10,10)$) に比べ，$\varepsilon=(7,7,9)$ とする方法を用いれば，デッドライン違反をするタスク数を約 7% 減少させることができることがわかった．また，図5において時刻 $\theta=0$ から $\theta=100$(時間)までの間にデッドライン違反のタスク数が急激に減っている理由としては，$\theta=0$ における外部環境をタスクの到着率が最も大きい状態である状態1としていることが考えられる．

4.2 レスポンスタイム

リスポンシブシステムに対しても，タスクの時間制約を満たすことだけでなく，短いレスポンスタイムの保証が要求される場合がある．実行されたタスクのレスポンスタイムを最小にする再構成の実行条件は，明らかに $\varepsilon=(10,10,10)$ である．これと比較すると，$l=10$ のときにデッドライン違反をするタスク数を最小にする条件 $\varepsilon=(7,8,10)$ は，稼動プロセッサの台数を平均的に少なめに保つので，レスポンスタイムがより長くなることが予想される．一方，$l=30$ の時にデッドライン違反をするタスク数を最小にする条件 $\varepsilon=(7,7,9)$ は，更に少なめにプロセッサの台数を保つので，レスポンスタイムも更に悪化することが考えられる．従って，適切な再構成の実行条件を選ぶ議論には，レスポンスタイムについての考察が不可欠である．

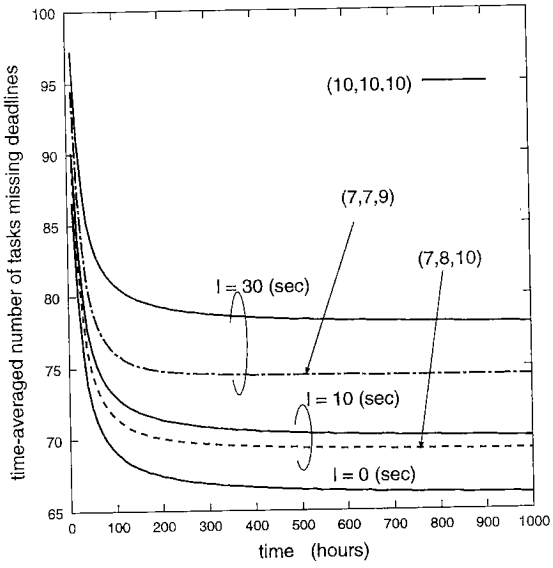

図5 単位時間当りのデッドライン違反のタスクの平均数
Fig. 5 Time-averaged number of tasks missing deadlines.

実行されたタスクの平均レスポンスタイムは，正常状態のシステムに到着して処理されたタスクがシステム内に滞在した時間の合計の期待値を，それらのタスクの総数の期待値で割って求められる．このうち，正常状態のシステムに到着したタスクの総数の期待値は，デッドライン違反をしたタスクの場合と同様の方法で計算できる．つまり，正常状態のシステムを表す各マーキングに対して，対応する外部環境での単位時間当りのタスクの到着数の平均(すなわちタスクの到着率)をリウォードレートとして割り当てる．なお，リウォードインパルスと障害状態のシステムを表しているマーキングのリウォードレートは0とする．このようにしておいて $E[Y(\theta)]$ を計算すればよい．

一方，タスクのシステム内での滞在時間の合計の期待値は，次のようにリウォードレートとリウォードインパルスを定めた上で，$E[Y(\theta)]$ を計算すればよい．正常状態のシステムを表しているマーキングに対しては，そのマーキングが表す状況での平均レスポンスタイムと単位時間当りのタスクの平均到着数との積をリウォードレートとする．そして，リウォードインパルスと障害状態のシステムを表しているマーキングに対するリウォードレートは0とする．

こうして$\varepsilon=(10,10,10)$，$\varepsilon=(7,7,9)$，$\varepsilon=(7,8,10)$ の再構成の実行条件について求めた結果を図6に示す．$\varepsilon=(7,7,9)$ と $\varepsilon=(7,8,10)$ の場合，

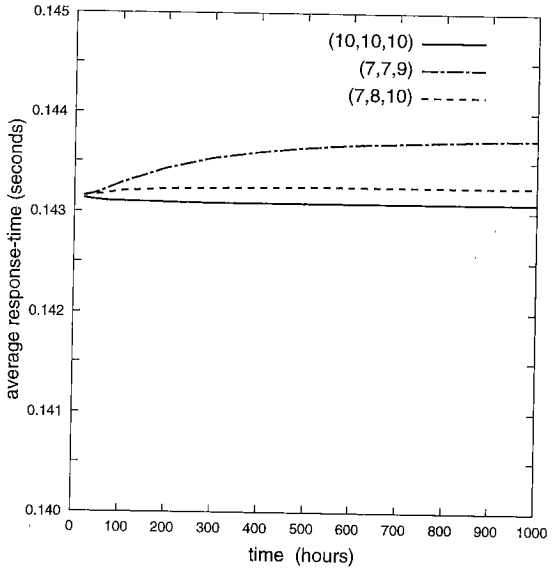

図6 平均レスポンスタイム
Fig. 6 Average response-time.

平均レスポンスタイムが時刻 $\theta = 0$ から $\theta = 500$(時間)にかけてやや上昇している理由は，$\theta = 0$ でプロセッサ10台が稼動しており，時間の経過と共に稼動中のプロセッサ数が減少するためであると考えられる．図6から，これら3通りの再構成法のレスポンスタイムの差は微小なものであることがわかる．その主な理由としては，プロセッサの台数がある数以上になるとタスクが到着したときにシステム内に空いたプロセッサの存在する確率がほぼ1になることが考えられる．その状況下では待たされることなくタスクが実行されるため，プロセッサの並列性によるレスポンスタイムの向上は，もはやそれ以上できなくなる．

5. むすび

本論文では，リアルタイム処理が要求されるマルチプロセッサシステム上でプロセッサの故障と修復に対応した再構成が行われる場合について，システムのモデル化について議論した．再構成のリアルタイム処理に与える影響は従来の研究では無視されることが多かったが，提案したモデル化ではそれを明確に考慮した．更に，従来の研究では暗黙のうちに不変であると仮定されることの多かったシステムの外的な環境についても，その変化の過程をシステムの故障・修理モデルに組み入れた．更に，このモデル上で外的な環境の変化に応じて修復のための再構成の実行のタイミングを制御するための条件ϵを記述した．最後に，具体的な数値例を用いた性能評価を通し，適切な実行条件ϵを選ぶことによって，デッドライン違反をするタスクを減少させることができることも示した．

今後の課題としては，実用的なリスポンシブマルチプロセッサシステムへの提案した手法の適用が考えられる．例えば，順序関係の存在する一般的なタスクスケジューリングに対しては，通常のリアルタイム処理，および再構成のリアルタイム処理への影響を表現しているリウォードレートとリウォードインパルスの値を，シミュレーション等を用いて得る必要がある．更に，実際のシステムにはプロセッサ以外にもバスやネットワーク等の構成要素があり，これらの故障を考慮したモデルの詳細化も必要と考えられる．また，その他の課題として，外部環境の変化を表すネット表現の一般化等がある．

文献

(1) Ciardo G., Trivedi K.S. and Muppala J.K.: "SPNP : stochastic petri net package", Proc. 3rd Int'l Workshop Petri Nets and Performance Models, pp.142–151 (Dec. 1989).
(2) Gross D., Harris C.M.: Fundamentals of Queueing Theory, John Wiley, New York (1985).
(3) Kakuda Y. and Kikuno T.: "Issues in responsive protocols design", Dependable Computing and Fault-Tolerant Systems, **7**, Springer-Verlag, pp.17–26 (1993).
(4) 角田良明, 菊野 亨："リスポンシブシステム", 計測と制御, **32**, 9, pp.750–758 (Sept. 1993).
(5) Kakuda Y., Kikuno T. and Kawashima K.: "Automated verification of responsive protocols modeled by extended finite state machines", Real-Time Systems, Kluwer Academic Publishers, **7**, 3, pp.275–289 (Nov.1994).
(6) Kopetz H. and Kakuda Y. (eds.): "Responsive Computer Systems", Dependable Computing and Fault-Tolerant Systems, **7**, Springer-Verlag (1993).
(7) Malek M.: "Responsive systems (A challenge for the nineties)", Proc. 16th Symp. on Microprocessing and Microprogramming, Keynote Address, Amsterdam, The Netherlands, North-Holland, Microprocessing and Micro programming 30, pp.9–16 (Aug. 1990).
(8) de Meer H. and Mauser H.: "A modeling approach for dynamically reconfigurable systems", Proc. 2nd International Workshop on Responsive Computer Systems, pp.149–158 (Oct. 1992).
(9) de Meer H., Trivedi K.S. and Cin M.D.: "Guarded repair of dependable systems", Theoretical Computer Science, **128**, 1–2 (June 1994).
(10) Melhem R.G.: "Bi-level reconfigurations of fault tol-

erance arrays in bi-modal computational environments", Proc. FTCS19, pp.488–195 (June 1989).
(11) Meyer J.F :"On evaluating the performability of degradable computing systems", IEEE Trans. Comput., **C-29**, 8, pp.720–731 (Aug. 1980).
(12) Muppala J.K., Woolet S.P. and Trivedi K.S.: "Real-time-systems performance in the presence of failures", Computer, **24**, 5, pp.37–47 (May 1991).
(13) Shin K.G., Krishna C.M. and Lee Y.H.: "Optimal dynamic control of resources in a distributed system", IEEE Trans. Software Eng., **15**, 10, pp.1188–1197 (Oct. 1989).
(14) Smith R.M., Trivedi K.S. and Ramesh A.V.: "Performability analysis : measures, an algorithm, and a case study", IEEE Trans. Comput., **C-37**, 4, pp.406–417 (April 1988).
(15) 土屋達弘, 陳 昶, 角田良明, 菊野 亨 : "再構成可能なリスポンシブシステムのモデリングと性能評価", 信学技報, **CPSY93**-57 (1994–03).

(平成6年12月8日受付，7年4月17日再受付)

土屋　達弘

平5阪大・基礎工・情報退学．平7同大大学院修士課程了．現在，同大学院博士課程在学中．フォールトトレラントシステム，リアルタイムシステム，リスポンシブシステムに関する研究に従事．

角田　良明

昭53広島大・工・電子卒．昭58同大大学院博士課程（システム工学専攻）了．工博．同年国際電信電話（株）入社．平3同社研究所主任研究員．現在，大阪大学基礎工学部情報工学科助教授．主に，プロトコル工学，リスポンシブシステムに関する研究に従事．平4第7回電気通信普及財団テレコムシステム技術奨励賞受賞．情報処理学会，IEEE等各会員．

菊野　　亨

昭45阪大・基礎工・制御卒．昭50同大大学院博士課程了．工博．同年広島大学工学部講師．同大助教授を経て，昭62大阪大学基礎工学部情報工学科助教授．平2同大教授．主にフォールトトレラントシステム，VLSI向きアルゴリズム，組合せ最適化問題の解法に関する研究に従事．平5電子情報通信学会論文賞受賞．情報処理学会，ACM，IEEE等各会員．