マルチメディア通信と分散処理 53−10
（1992. 1. 23）

拡散モデルに基づく協調分散制御システム

神余浩夫　　竹垣盛一
三菱電機(株)中央研究所
661 兵庫県尼崎市塚口本町 8-1-1

協調分散制御システムは、プラントの異常や制御要素の故障などのシステム状況変化に対して、システムを構成する制御要素の相互作用により、システムの機能と性能を維持するようにシステム再構成を行う。具体的には、システムが実行している定常的タスクの実行とそのリアルタイム制約を保証するように、タスクを分散スケジューリングする。本論文では、分散タスクスケジューリング手法として、ノードの近傍系に関するスケジューリングを繰り返して、全体のタスク分散状態の平衡化を図る拡散モデルを提案する。拡散モデルに基づく協調分散動作を行う場合、ノード間のタスク内部状態の同期を行う check point と、分散タスクスケジューリングを行う remapping point を設定する必要がある。両タイミングは通信性能およびタスク制約を考慮して、タスクの実行効率とリアルタイム保証率が高くなるように設定する。このタイミングに関する解析評価結果についても報告する。

Coordinative Distributed Control System Based On the Diffusion Model

Hiroo Kanamaru　　Morikazu Takegaki
Central Research Lab. Mitsubishi Electric Corp.
8-1-1,Tsukaguchi-Honmachi, Amagasaki, Hyogo, Japan

The coordinative decentralized control system makes reconfigurate itself by the coordinatinally interactions between its elemental controllers in order to keep its functions and performances under the various system conditions; an anormally on the plant and an fault of controllers. In the concrete, the controllers make scheduling the regular tasks to execute all tasks and to guarantee real-time constraints of tasks. This paper proposed the diffusion model which the task scheduling in each node-neighborhood make coordinate the global system. For appling the diffusion model to the distributed task scheduling, two inter-node communication actions are needed; one is check point to synchronize parameters of tasks, and the other is remapping point to schedule tasks between nodes. These two communication timings and intervals are designed, considering the communication traffic, load ratio of processors and real-time constraints of tasks. We analyzed appropriate these communication intervals and descrived them in this paper.

1 諸言

発電プラント、化学プラント制御システムや生産ライン、ロボット制御装置などの産業分野における情報制御システムは、対象の挙動が刻々と変化するので、その変化に追従して対象を監視制御するためにリアルタイム性が要求される。このリアルタイム性は、単に高速処理ということではなく、サンプリング周期や制御周期、タイムアウト時限などのデッドラインを保証する強リアルタイムシステムである。また、このような産業プラントが停止あるいは制御不能になると多大の損失を被るので、フォールトトレラント性も強く要求される。従来、このような要求に対して、個々の制御装置に対して処理を固定的に割り付け専用装置として、両者で多重冗長構成とすることが多かった [1]。しかし、全ての処理が故障後瞬時に待機系に切り替わる必要があるわけではないので、全ての制御要素を多重化にすることはコストと効率の点で効果的ではない。そこで、制御要素に対する処理タスク割当を状況に適応するように動的に変更する、リアルタイム分散システムの動的スケジューリング手法が重要となってくる。

筆者らは産業プラントの情報制御システムにおいて、プラント異常やシステム故障などの状況変化に対して、処理していたタスクを制御要素が協調的にバックアップあるいは負荷分散するようにタスクスケジューリングを行なう協調分散制御アーキテクチャを提案している [2]。この協調動作は、部分的範囲の制御要素間の相互作用を繰り返して、影響を全体へと波及させる拡散モデルに基づく協調動作を行ない、次第に全体システムが状況に適応することを特徴とする [3, 4]。ここで、制御要素相互でタスクの実行割当を切り替えるためには、該要素間でタスクの状態とデータの一貫性を維持する必要があり、そのための同期機構が必要となる。また、タスク実行やデータ更新についてリアルタイム制約が存在しており、スケジューリングや制御要素間通信はこれらの制約に反することなく動作しなければならない。

本論文では、リアルタイム制約下で状況に応じて動的に制御要素へのタスク割当を変更する協調分散型制御システムについて述べる。はじめに、産業分野における情報制御システムについて触れ、提案するシステムの適用範囲を示す。次に、拡散モデルの原理と協調動作の定式化を行なう。次に、本手法のリアルタイム分散タスクスケジューリング方式と、そのためのデータ同期通信について解説し、それらの時間的制約条件に関する考察を報告する。

2 情報制御システムにおける協調

はじめに、情報制御システムにおいて分散協調する目的と、協調動作が分散スケジューリングと同等であることについて述べる。

2.1 協調の目的

化学プラントの運転監視システムや生産ラインの制御システムなど、産業システムにおける情報制御システムの例を図 1 に示す。プラントのセンサと接続されるコントローラは、周期的なサンプリングと応答、入力イベントに対する処理などのタスクを定常的に処理している。その周期は数十 msec から数 sec であることが多い。コントローラは直接あるいはセルコントローラを仲介として基幹バスに接続される。基幹バスには監視操作端末 (OPS) や記録集表用の計算機が接続され、コントローラと百 msec から秒単位で通信が行なわれる。また、プラント立ち上げ時やメンテナンス時には、上位の計算機からコントローラに対するプログラムやパラメータなどの転送や、OPS からの特権的操作が行なわれる。これらのタスクは対象の動特性に追従する

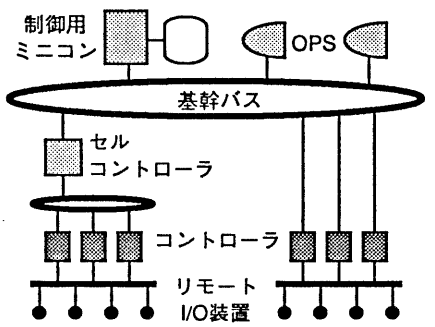

図 1: 情報制御システムの例

ため、処理や応答について遅延限界(deadline)が明示的である強リアルタイム制約を有する。コントローラや制御用計算機などの制御要素は各タスクのリアルタイム制約を満足しなければならない。

情報制御システムを構成する分散制御要素はその時のシステム状態や環境においてシステム機能と性能を維持するように協調動作する。ここで、機能とは継続的あるいは周期的に実行されているタスクの処理であり、性能とはタスク処理および応答速度である。すなわち、協調動作とは、定常的なタスクのリアルタイム制約を満足するようにタスクに対してプロセッサ、メモリ、通信路などの資源を動的にスケジューリングして、システム再構成を行うことである。しかし、図1のような実際的なシステム構成では、上位計算機から下位コントローラまで全ての制御要素が同じタスクの実行を相互に切替えることは、現実的でない。タスク実行切替を行なうには、少なくともタスクに必要なI/O、データ、プログラムおよびH/Wをそれぞれの制御要素が備える必要があり、比較的よく似た役割を果たしている制御要素、セルコントローラの下のコントローラ群、あるいは基幹バスに接続されたコントローラ、計算機群のそれぞれに中で協調動作が可能となる。

2.2 分散リアルタイムシステムのスケジューリング手法

一般的な分散リアルタイムシステムの動的スケジューリング問題はNP困難であることが知られている。しかし、スケジューリングに時間を要しては本来のタスクのリアルタイム性が保証できなくなる。また、全体を集中的にスケジューリングする要素を用意することは、耐故障性と可用性の点で不利である。これらの複雑性と問題点から、現在提案されている分散リアルタイムスケジューリング方式は、ノードに発生したタスクのスケジューリングを行なうローカルスケジューリングと、ノード間の主に負荷分散を目的とした分散スケジューリングのふたつの部分から構成されることが多い[5]。

Ramamrithanらは、タスクのリアルタイム制約保証率をできるだけ高くするため、焦点配置法と入札法を組み合わせた手法を提案している[6, 7]。焦点配置法はノードで保証できないタスクがあれば、処理時間余裕があると予想されるノードを選んでタスクを送る。入札法はタスクの送り先を入札によって決定する。これらの手法をヒューリスティックに組み合わせて、効果的な分散タスクスケジューリングを行なう。Chengらは、いくつかのタスクが同じデッドラインを持つ時、そのタスクグループをネットワーク上に分散し、焦点配置法と入札法を組み合わせてグループの実行制御を行なう手法を提案している[5]。

分散スケジューリングの問題点は、複雑性を回避するためにタスクとノードをどのようなサブシステムに分割し、その中でタスク配置をどのようにして決めるかということにある。筆者らが提案する拡散協調モデルは、サブシステムをノードとその近傍系に限定して、数的複雑性を回避している。以下、拡散協調モデルについて詳細に述べる。

3 拡散協調モデル

3.1 拡散モデルの概要

協調分散型システムでは、全体を統括管理する要素や特定の待機系を持たず、全ての要素がアプリケーションタスクの実行を担当し、それらが協調的に動作することによって耐故障性や可用性を高めている[3, 2]。従って、故障のバックアップや負荷分散などのタスクスケジューリングおよび同期通信を各制御要素が自律的にかつ協調的に行なわなくてはならない。このとき、ある要素の状況変化に対して全ての要素が一斉に対応する必要はなく、関係の深い要素から順次データ転送やスケジューリングを行なっても、全体として矛盾は生じない。そして、各要素が全体を意識することなく動作しながら、全体が状況に適したシステム構成へと遷移していく。この自己組織的な協調モデルとして、筆者らは拡散協調モデルを提案している[2]。

一般的なネットワーク構造を用いた拡散協調モデルの概念図を図2に示す。分散制御要素であるノードiは、直接的な関連を持つ相互接続されたノード群とサブシステムS_iを構成する。こ

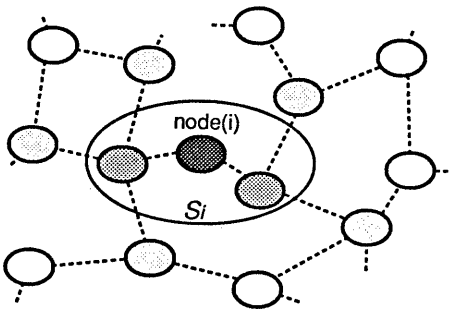

図 2: 拡散協調モデルの概念図

のサブシステムをiの近傍系という。ノードiはその近傍系内部において局所的なシステム再構成や相互作用によるパラメータ変更などを行なう。各ノードはそれぞれの近傍系について並列的に再構成を行なうので、局所的な変化は拡散的に全体へと波及して、新しい平衡状態が形成される。例えば、分散システムの負荷分散に関して言えば、ノードiがデッドラインを守れないタスクを多く抱えているとすると、まず近傍系S_iの局所的な分散タスクスケジューリングによって、S_iの他ノードにタスクが大きく分散される。次に、iの近傍系ノードが自分の近傍系にタスクを分散し、この繰り返しによって遠距離のノードへと拡散的に負荷分散が進んでいく。各近傍系が負荷均等状態となった時全体は平衡状態となって安定する。このとき、システムの各制御要素は協調的に動作しているとみなすことができる。

3.2 拡散モデルの協調性

拡散協調モデルは各近傍系の相互作用を繰り返すことによって、全体が平衡状態になるようにシステム状態が維持される。ここで問題となるのは、全体を関知しない近傍系のみの相互作用だけで、システム全体として協調的な状態に達するかどうかである。もし、分散システムの制御要素群の状態(ネットワーク様相)の挙動を、同じ漸近挙動を持つ一般化正準-散逸システムに置換できるならば、拡散協調モデルに基づく協調性の議論はネットワーク様相の漸近挙動の安定性問題として扱うことができる[8]。

ネットワーク様相が$x=(x_1,x_2,\cdots,x_n)$である分散システムにおいて、局所的な状態遷移則を次式で定義する。

$$x_i(t+1)=\Phi(x_i(t),x_{\partial i}(t))$$

ここで、$x_{\partial i}$はx_iの近傍の状態を表す。このサブシステムにおける局所的および大域的なエネルギー関数を次式で定義する。

$$f_i(t)=f_i(x_i(t),x_{\partial i}(t))$$

$$f(t)=\sum_i f_i(t)$$

全体システムの協調状態とはエネルギー最小状態であるとすると、

$$\lim_{t\to\infty} f(t)=\min f$$

となるネットワーク様相xがシステムの平衡状態となる。

このような条件を満足する局所状態遷移則およびエネルギー関数を設計できれば、拡散協調モデルに基づく分散システムの協調安定性を保証できる。エネルギー関数は各ノードの負荷としてのタスク処理時間や担当タスクの責任度の合計などが定義できる。局所状態遷移則は近傍系における制御要素の相互作用として定義できる。ただし、一般的な遷移則が常に上式を満足するとは限らない。

3.3 二相拡散モデルの挙動

拡散協調モデルに基づく協調性を保証できる例として、状態遷移則を二相拡散モデルに基づく制御要素間の負荷分配を考える[4]。二相拡散モデルとは、鎖状接続された制御要素群について隣接する制御要素を近傍系する。まず、ふたつの制御要素の間に仮想的なノード間ペアを作り、そのなかでタスク分散を行なった後、実ノードで両側の仮想ペアのタスク分散の結果を統合し、近傍系のタスク分散配置を決定する。実ノードiの負荷(処理時間)をx_i、両隣接装置との仮想ノードにおけるiの担当負荷をx_{i-},x_{i+}とすると、次の関係が成り立つ。

$$x_i(t)=x_{i+}(t)+x_{i-}(t)$$

$$x_{i+}(t+1) = x_{i+}(t) + \alpha(x_{(i+1)-}(t) - x_{i+}(t))$$
$$x_{i-}(t+1) = x_{i-}(t) + \alpha(x_{(i-1)+}(t) - x_{i-}(t))$$

ただし、α は隣接間の負荷移動量に対する係数である。

エネルギー関数は計算機の負荷のばらつき (分散) として定義する。

$$f_i(t) = (x_i(t) - x_m)^2$$
$$f(t) = \sum_i (x_i(t) - x_m)^2$$

ただし、x_m は全計算機における平均負荷である。

$$x_m \equiv \sum_i x_i/n := const$$

このとき、エネルギー関数の時間変化に対する増分は、

$$f(t+1)-f(t) = \sum_i (x_i(t)-x_m)^2-(x_i(t)-x_m)^2)$$
$$= \sum_i (x_i(t+1)^2 - x_i(t)^2)$$
$$= \sum_i (x_{i+}(t+1)^2 - x_{i+}(t)^2 + x_{i-}(t+1)^2 - x_{i-}(t)^2)$$
$$= \sum_i \alpha(\alpha-1)(x_{i-}(t) - x_{i+}(t))^2$$

すなわち、隣接計算機間の負荷分配係数を、$0 \leq \alpha \leq 1$ とすると、大域的なエネルギー関数は単調減少となり、$f(t \to \infty) = 0$ の均等負荷分散状態に漸近する。実際の負荷移動はタスク単位であるからエネルギー関数は不連続変化するが、移動量が負荷分配係数の範囲内であれば問題ない。

この結果は、同じ局所状態遷移則に基づき構成された全ノードに関するエネルギー大域的遷移則の Lyapunov 安定条件と一致する [2]。

4 スケジューリングと同期通信

4.1 タスク実行切替え動作

前述の二相拡散モデルに従ってタスク配置を決定しても、タスク実行ノードを切替えて継続的に処理するためには、これまで実行していたノードから実行引き継ぎに必要なデータと切替えタイミングの同期が必要である例えば、ホットスタンバイ型の待機冗長系の場合、システム稼働中に常用系から待機系に常時転送して、故障検出に応じてタスク実行を待機系に切替える。転送すべきデータは、タスクの内部フラグ、トラッキングデータなどであり、これらをチェックポイントデータ、転送タイミングをチェックポイント (CP) と呼ぶ。待機系を持たない協調分散型システムの場合、タスク実行切替は近傍系におけるタスクスケジューリングのタイミングであり、これをリマッピングポイント (RP) と呼ぶ。協調分散型の情報制御システムでは、このふたつの同期通信によって、システムにおけるタスクとデータの一貫性を保証している [9]。

分散スケジューリングを行なうサブシステムにおいて、少なくとも RP の時点でスケジューリングされる各タスクの CP データは等価でなくてはならない。しかし、CP を余りに細かく設定する、例えば CP データ更新毎に転送すると、通信量の増加とノードにおける通信処理の増大によって、本来のタスク実行に利用できる時間が圧迫される。逆に、CP を間隔を大きくすると、ノード間の時刻遅延がタスクのリアルタイム制約に影響を与えたり、システム再構成が迅速に行なわれなくなる。従って、タスクのリアルタイム保証とプロセッサ能力と通信バンドの効率的利用のためには、タスクの要求や性質に応じて適切な CP および RP タイミングを与えなければならない。以下では、これらの適切なタイミングについて解析的評価を行なう。

4.2 協調分散制御システムの CP

CP による常用・待機系の内部データの同期手法は待機冗長によるフォールトトレラント方式では広く使われている方法である [1]。待機冗長系における CP タイミングの代表的なものに以下が挙げられる。

- 両系メモリ書き込み法
  常用系で CP データをメモリ上に書き込む毎に待機系のメモリも更新し、両系のメモリ内容を常に一定に保つ。ハードウェアの工夫が必要。

- 周期転送法
  CP データを一定周期で待機系に転送する。転送周期間に I/O アクセスが行われるとき、待機系でその動作の再現が可能なように履歴 (ジャーナル) を作成する必要がある。

- I/O 同期方式 [10]
  ファイルの書き込みなどジャーナルが必要な I/O アクセスをタイミングとして、待機系に CP データを転送する。待機系は直前の I/O アクセス終了時点から処理を再開できる。

- CP 転送命令方式
  アプリケーションタスクにおいて CP 転送命令により CP データを転送する。タスク終了時には命令が明示されてなくとも CP 転送を行う。

協調分散制御システムでは特定の待機系が存在せず、全ての装置がそれぞれのタスク処理を行っている。また、各制御要素は異なる周期の複数のタスクを処理している。さらに、タスクの入出力はプラントとの I/O であり、ジャーナルを参照して再現できないものが多い。以上の理由から、協調分散制御システムにおける CP 転送方式は、I/O 同期方式を包含した CP 転送命令方式を用いる。すなわち、アプリケーションタスク中に、出力が行われるときなど区切りのいい箇所に CP 転送命令を埋め込み、CP データ転送を明示的に制御する。

4.3 チェックポイントのタイミング

CP の設定間隔と、それに伴う通信量および通信遅延について評価する。ここで、近傍系サブシステム内部の通信にブロードキャストを使用せず、サブシステム内の m 個の制御要素に対してそれぞれメッセージ転送を行なうものとする。CP 動作タイミングを図 3 に示す。

あるタスクに関するチェックポイント i において転送する CP データ量を $q_i$byte とし、このサブシステム内部の通信容量を一律 Bbyte/sec、両ノードにおける通信プロトコルのオーバーヘッドを Psec とする。いま、近傍系の全ての制御要素に対して CP 転送した時の通信時間は次式で与えられる。

$$L_i = m(q_i/B + P)$$

図 3 において、CP $(i-1)$ と i の間隔を T_{Ci} とすると、i における CP 通信の時間占有率は次式で定義できる。

$$R_i = L_i/T_{Ci}$$

次に、このタスク中に設けた CP 数を減らして CP 間隔を大きくすることを考える。設定した CP $(i = 1 \cdots n)$ における CP 転送を $i = n$ でまとめて行なうものとする。同じタスクに関していえば、いくつかの CP で同一の CP データを転送していることがあるので、CP をまとめた時に転送すべき CP データ量は減少する。簡単のため、各 CP における CP データ転送量を等しいとし、任意のふたつの CP$i, j$ の間で CP データが重なる割合を係数 α の一定とする。

$$\alpha := const, (0 \leq \alpha \leq 1)$$

$\alpha = 0$ のとき、CP データに重なりはなく、1 のとき CP データは一致していることを意味する。このとき、CP$i = n$ における転送データ量は次式となる。

$$\bar{q}_n = q \sum_{i=1}^{n} nCi\alpha^{i-1}(1-\alpha)^{n-i}$$

$$q = q_i, i = 1, 2, \ldots, n$$

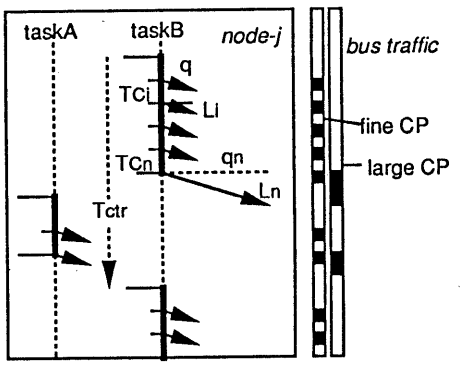

図 3: CP タイミング

この式で α が 0 および 1 のとき、$\bar{q}_n$ はそれぞれ、nq, q となる。このとき、通信時間と占有率はそれぞれ次式となる。

$$L_n = m(\bar{q}_n/B + P) \leq m(nq/B + P)$$

$$R_n = m(\bar{q}_n/B + P)/nT_{Ci} \leq R_i$$

$$T_{Cn} = nT_{Ci}$$

この結果は、CP 間隔を大きくとり、CP データをまとめて転送した方が、平均的な通信効率を表す通信時間占有率が小さくなることを示している。この効果は通信プロトコルの処理が重く、CP データのオーバーラップ率が高いほど期待できる。ただし、一回の CP における通信時間は大きくなるので、L_n が他のタスクのリアルタイム制約に干渉する場合がある。すなわち、最大の CP データ転送時間がシステム中のタスクの最小デッドライン時間を越えないように、CP 間隔を大きく設定することが通信量の点で効率的である。

4.4 スケジューリングのタイミング

リアルタイム制約を持つタスクは実行が他制御要素に切り替わった後も、そのリアルタイム制約が保証されなければならないので、サブシステム内の切替えに関わる制御要素群はタスクスケジューリングの結果に従ってタスクを実行する際に同期をとる。分散スケジューリングの通信が同期機構として働くので、特別な同期機構を用意する必要はない [11]。本節では、二相拡散アルゴリズムに従うタスク分散スケジューリングアルゴリズムについて、通信量と同期通信の遅延とスケジューリングタイミングあるいは RP タイミングを評価する。

サブシステムにおける複数ノード間の RP タイミングチャートを図 4 に示す。はじめに、二相拡散モデルに基づくスケジューリング所要時間を導く。隣接制御要素との 1 回あたりの通信量を q_R、仮想ノードにおけるタスク配置を決定する時間を S とすると、スケジューリング所要時間は隣接制御要素との通信 2 回とタスク配置決定時間であるから、次式となる。

$$L_R = 2m(q_R/B + P) + S$$

二相拡散モデルの場合、近傍系の通信相手数は、$m = 2$ である。

RP では近傍系においてスケジューリングされるタスクの内部状態の一貫性が保証されていなければならないので、サブシステムにおける RP の間隔 T_R は全てのタスクのどの CP 間隔よりも長くなくてはならない。これが破られると、タスク実行のノードを切替えた時、タスクの正当な実行が保証できなくなる。

$$T_R \geq T_{Ci}, \quad \forall i$$

同じ理由で、RP の同期通信の途中で CP データ転送が必要となるタスク処理を行なうことはできない。制御要素がマルチタスク環境であっても、分散タスクスケジューリングを含む RP 同期通信処理は優先度の高い非プリエンプティブ処理として扱わなければならない。この条件下でタスクのリアルタイム制約を満足するためには、RP 処理の遅延時間が最小の CP 間隔と CP データ転送時間の合計よりも短いことが必要である。

$$L_R \leq \min T_{Ci} + L_{Ci}$$

CP データ通信時間 L_{Ci} と RP データ通信時間 L_R を比較すると、二相拡散モデルでは S は十分小さく、それぞれの通信量 q_{Ci}, q_R の比が支配的である。一般に、CP 間隔を大きく設定すると、両者の通信量は、

$$q_{Ci} \gg q_R$$

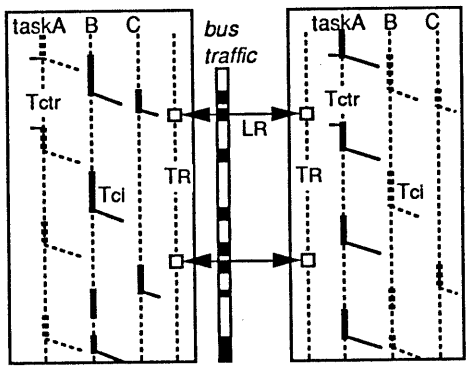

図 4: RP のタイミング

となるので、L_R の最大値条件は満足されやすい条件である。

RP 周期を大きくしても、被スケジューリングタスク数もノード数も変わらないので、RP データ通信時間は変わらない。しかし、RP 周期 T_R を大きくすると、状況に応じたシステム再構成の周期が長くなり、拡散モデルに従って全体が平衡状態に達するまでに長時間を要することになる。従来の待機冗長系における故障後のタスク切替え時間は、そのタスクの制御周期の 2 倍以内を要求していた。協調分散制御システムにもこの要求を適用すると、最小を制御周期を T_{ctr} として、

$$2T_{ctr} \geq T_R \geq T_{Ci}, \quad \forall i$$

となる。

以上をまとめると、CP 間隔 T_{Ci} は CP データ転送時間 L_{Ci} が最小の T_{Ci} を越えない限り、大きくとった方が有利であり、RP 間隔 T_R は最大の T_{Ci} より大きく、最小のタスク切替え要求時間 $2T_{ctr}$ より小さくしなければならない。後者の条件が満足できない場合、通信オーバーヘッドが増加するが、CP 間隔を小さくする。全条件を満足できる時、二相拡散タスク分散方式に基づく、協調分散制御システムを構成することができる。

5 結論

拡散協調モデルに基づき、プラント異常やシステム故障時の動的バックアップや負荷分散を行なう協調分散制御システムについて、二相拡散モデルによる分散スケジューリングの協調性とチェックポイント、リマッピングポイントのタイミング設定条件について解析的評価を行なった。

情報制御システムのリアルタイム性と無停止性に関する厳しい制約は、一般的な計算機システムに比べてフォールトトレラントを含むシステム再構成を非常に難しいものにしている。本論文で示したように、タスクスケジューリング方式、通信方式、同期のタイミングなどにタスクのリアルタイム性を満足するためのいくつもの条件が付与される。これらの制約条件を明確にした協調機構を与えることによって、協調分散型の情報制御アーキテクチャが構築できる。

今後の研究として、拡散モデルに基づく分散リアルタイムタスクスケジューリング方式の具体化と、それに伴う各種リアルタイム制約条件の解析評価およびシミュレーション等を予定している。

参考文献

[1] 南谷：フォールトトレラントコンピュータ, オーム社, 143/165 (1991)

[2] 神余, 竹垣：協調分散制御システムアーキテクチャ: CODA の概念モデル, 計測自動制御学会論文誌, 27-4, 458/465 (1991)

[3] H.Kanamaru, M.Takegaki : The Coordinative Decentralized Control System Based on Local Communications, IECON'90 16th Annual Conf. of IEEE IES, 533/538 (1990)

[4] M.Takegaki, H.Kanamaru, M.Fujita : The Diffusion Model Based Task Remapping for Distributed Real-Time Systems, 12th IEEE Real-Time Systems Symp. (1991)

[5] S.Cheng et.al. : Scheduling Algorithms fot Hard Real-Time Systems, Hard Real-Time Systems, IEEE CS Press, 150/173 (1988)

[6] K.Ramamrithan and J.Stankovic : Dynamic Task Scheduling in Distributed Hard Real-Time Systems, IEEE Software, 1-3, 65/75 (1984)

[7] J.Stankovic et.al : Evaluation of a Flexible Task Scheduling Algorithm for Distributed Hard Real-Time Systems, IEEE Trans. on Computers, C-34,12, 1130/1143 (1985)

[8] 深尾：分散システム論, 昭晃堂, 93/133 (1987)

[9] 神余, 竹垣：協調分散型制御システムのフォールトトレラント方式, 情報処理学会研究会 SWoPP '91, ARC89-8, 57/64 (1991)

[10] 真矢 他：分散システムにおける高速回復方式の提案, 電情通論文誌, Vol.J74-D-I, No.10, 729/738 (1991)

[11] 神余, 竹垣：協調分散システムにおける同期機構, 第 34 回システム制御情報学会研究発表講演会論文集, 65/66 (1990)