C-011

負荷と重要度に基づく動的経路制御方式と検証
Value and Load Oriented Dynamic Data Flow Control and Its Verification

長家 正和† 小山 恭平† 島川 博光†
Masakazu Nagaya Kyohei Koyama Hiromitsu Shimakawa

1. はじめに

近年の大規模制御システムでは, 膨大な数のセンサの周期データを転送するサービスに負荷が集中する問題が発生している. この問題に対処するには, データの転送経路を動的に変更し特定のサービスへの負荷集中を防ぐのが有効な解決策である. 現在, ストリームデータの実時間転送のための経路制御技術の研究 [1] や, Ad Hoc Network でのキャッシュによる負荷分散, アクセス向上の研究 [2] や, ミドルウェアを用いてデータを共有していく研究 [3] がなされている. いずれの研究も, 特定のサービスへの負荷分散という点では本研究と同一であるが, 制御システムとして重要な問題である「確実に膨大な周期データを転送する.」という配慮に欠けている. そこで本稿では, これらを配慮した, 動的データ経路制御技術について述べる.

2. データ転送の現状と問題点

図 1 に示すような近年大規模制御システムは, コストが高くなるという問題が発生している. 以下に具体的に規模を示す.

・各ノードは同一ネットワークに接続される.
・データの取得周期は数秒程度.
・データの点数は数万程度.
・クライアント数は十数個程度.

具体例として, トンネル換気制御システムや下水処理システムがあげられる.

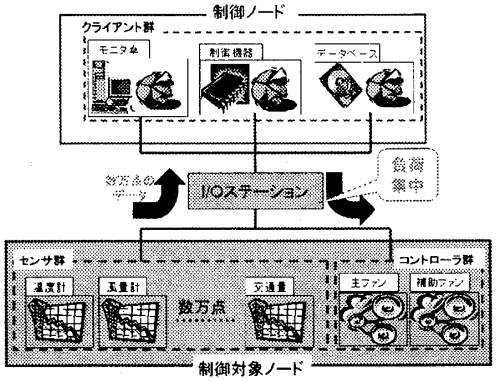

図 1: 大規模制御システム

コストが高くなる問題には次に原因が考えられる. センサより取得されたデータは I/O ステーションと呼ばれるコンピュータに転送され集積される. このため, 数万点もの膨大なサンプルデータが周期的に, I/O ステーションに集中し, 非常に高い負荷が発生している. このため I/O ステーションには通常の汎用コンピュータでは対応できず, 高性能な計算機が必要となっている.

この問題を回避するには, ひとつのデータをノードからノードへ転送し, 負荷の集中するノードを排除するのが有効な解決手法である. しかし, 現在この方式は, 次の問題に対処するのが困難なため採用されていない. この方式では, 各ノードの負荷状況に関わらずデータの転送処理を円滑に行えるような, 冗長性をもつシステム設計が必要となる. またデータが順に転送されるため, 転送途中で障害が発生する可能性が高くなる. このためデータ転送の信頼性が低下する. そこでこの問題への対処手法が必要となる. 本稿では, これらの問題を解決する新しい動的転送経路技術についてのべる.

3. 動的データ転送経路制御方式の概要

3.1 動的データ転送経路制御方式 DDFC

DDFC(Dynamic Data Flow Control) は, 2 章で示した問題を解決するための動的経路制御方式である. 本方式は, 各ノードの負荷状況と, サンプルデータに対してデータを必要とする各ノードの重要度を考慮して動的にデータの転送経路を制御する方式である. DDFC は, 図 2 で示すように, 動的に転送経路を決定すると, その経路を示したデータ経路表を作成する. そして, 各ノードは, 最初にデータ経路表をそれに記載された経路にしたがい, 上流から下流へ順次転送していき, 経路を構成する. 以下データはこの経路にそって転送される.

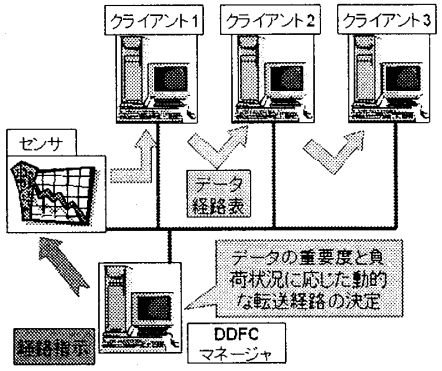

図 2: DDFC に基づく動作例

経路は各ノードの負荷とデータの重要度を考慮しているので, システムに無駄な負荷が生じることを防げる. さらに負荷が集中するノードのために高性能な計算機を用意する必要がなくなりコストダウンが可能となる.

3.2 各ノードにおける重要度

DDFC では, 各サンプルデータに対して各ノードへの転送がどのくらい重要かを示す重要度を設定することにより, きめ細やかな制御を実行することができる. 重要

†立命館大学大学院理工学研究科

度は，データの転送抜けがあると，制御に影響を及ぼす可能性が大きいノードにより高い重要度を設定するものとする．

3.3 負荷と重要度に基づくデータ転送

各ノードは，転送処理以外にもさまざまな制御処理をおこなっているため，データの受信は可能でも送信は不可能になるという場合もある．このような場合をふくめた，各クライアントの負荷と重要度に基づいた転送経路決定のアルゴリズムを考える上での仮定を示す．

1. センサは１つのノードにしかデータを送信できない．
2. 重要度が高い順に転送する．
3. アルゴリズムの複雑化を避けるため，転送経路の分岐は，負荷分散と転送障害解決以外には使用しない．また２分岐以上の分岐はしないものとする．

以上の仮定のもとに経路探索のアルゴリズムを示す．データがまだ転送されていないノードを重要度の高い順に並べたリストをデータ未転送ノード・リストと呼ぶことにする．

1. 与えられた p 個のノードを重要度の高い順に n_1,n_2,...,n_p と並べる．この順に調べて，他の２つのノードにデータを分岐転送できるノードをさがす．このノードを n_m とする．
2. n_m が見つかれば以下の処理を行う．
   (a) n_m より，重要度が高い $(m-1)$ 個のノードは n_1,n_2, ...,n_{m-1} である．これらのノードをデータ未転送ノード・リストに加える．
   (b) データ未転送ノード・リストが空なら，n_m は n_{m+1} にデータを転送する．
   データ未転送ノード・リストが空でないなら，
   i. その中でもっとも重要度が高いノードを n_h として，n_m は n_h と n_{m+1} にデータを２分岐転送する．
   ii. n_h にはデータが転送されたので，データ未転送ノード・リストから削除する．
   (c) n_1 から n_m を除いた $(p-m)$ 個のノードについて
   i. これらのノードが無いなら，つまり $p-m=0$ なら，以下の判定を行う．
      A. データ未転送ノード・リストが空なら，正しいデータ転送経路が見つかったことになる．
      B. そうでないなら，正しいデータ転送経路は見つからなかったことになる．
   ii. ノードがまだ存在するなら，つまり $p-m>0$ なら，ノード数を $(p-m)$ 個として上記の１から処理を再帰的に繰り返す．

n_m が見つからなければ，正しいデータ転送経路は見つからなかったことになる．

このようにしてDDFCは，正しい転送経路を決定する．

3.4 負荷の統一的表現

各ノードは異なる性能を持ち，かつ転送以外の処理をおこなっている．このため全てのノードの負荷を統一的に扱う手段がないと，システム全体として均一な負荷をもつような経路制御をすることができない．そこで各ノードが自身の負荷の度合より単位時間あたり転送できる量を算出する．この算出した値を余裕転送能力と呼び負荷表現を定量的に扱う．負荷には，ノードの負荷とネットワークの負荷の２種類がある．各ノードの転送能力は，これらの負荷に制約される．そこで，負荷により制約された転送能力のことをそれぞれをノード制約の転送能力，ネットワーク制約の転送能力と呼ぶこととする．余裕能力はこれら２つの能力の小さい方に制約される．
具体的に余裕転送能力の算出方法を以下に示す．ノード制約の転送能力はCPUとネットワークにデータを流しだすチップセットの性能に左右される．CPUとチップセットの間はデータがブロック転送されるのでCPUの空き時間と転送できるデータ量は比例しない．あるデータ量の転送に必要な，実行時間をあらかじめ計測しておき，これを表形式で保持しておく．この表内の，現在のCPU空き時間以下で最大の実行時間に相当するデータ量からノード制約の転送能力を算出する．ネットワーク制約の転送能力はネットワークの性能に左右される．DDFCでは，各ノードがおこなう通信を全て管理しているため，ネットワークが通信できるデータの総量から各ノードが通信している総量をさしひいて，ネットワーク制約の転送能力を算出する．これら２つのうち，小さい方を余裕転送能力とし負荷表現を統一する．

3.5 転送障害への解決

各ノードに対して，周期データや要求したデータがあらかじめ定めた期間過ぎても到着しない場合をデータの未到着とし，転送障害と呼ぶこととする．DDFCに基づくシステムでは，複数のノードを経由してデータを転送がおこなわれる．このため転送過程で転送障害が発生する可能性が高くなる．そのため転送障害にすみやかに対処する必要がある．具体例を図３に示し，その対処方を以下に示す．ノードBからノードCの間でパケット損失などによる転送障害が発生したとする．ノードCより下流のノードには，データが到着しないため下流のノードはデータの未到着を検出する．データ未到着を検出したノードは再送要求をDDFCマネージャに送出するものとする．もしこれらがすべてのノードが再送要求を出すと，DDFCマネージャの負荷が高くなる．そこで，これをさけるために，データ未到着を検出したノードは，自らより下流のクライアントに再送要求停止命令を出してか

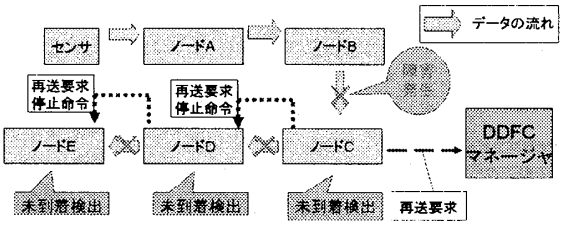

図 3: 転送障害対応図

ら,DDFC マネージャに再送要求を送るものとする．一部再送要求停止命令の到着がまにあわず再送要求を出してしまう下流のノードがあるかもしれないが, この仕組により大幅に DDFC マネージャの負荷を抑えることができる．DDFC マネージャは, 再送要求を受けた後, 障害が発生したノードより上流でかつ負荷の一番軽いノードにノード C への再送命令を出す．再送命令を受けたノードは, 従来からのノードと, ノード C の 2 つの経路にデータを転送する．

4. 実験

4.1 目的と手順

DDFC では, データを複数のノードを経由して転送するためデータの到着時間がばらつく可能性がある．そこで, データの到着時間を, 従来の方式と DDFC とで比較する．従来からの集中方式に基づいたシステムと DDFC に基づくシステムを用意した．

1. 集中方式に基づくシステムは, センサの代わりをする計算機よりデータを転送し,I/O ステーション, ノードの役割をはたす計算機を経由したあと再びセンサにデータを返す経路をとり, それに消費した時間を計測した．

2. DDFC に基づくシステムは, センサの代わりをする計算機よりノードの役割をはたす計算機を 3 つ経由したあと, 再びセンサにデータを返す経路をとり, それに消費した時間を計測した．

なお各計算機は 100Mbit/s のイーサネットでスイッチングハブを介して同一のネットワークに接続する．

4.2 結果

実験はデータの送信を 100 回試行した．それぞれの結果を次に示す．集中方式の結果を図 4 に示す．DDFC の結果を図 5 に示す．図 4 と図 5 にみられる, 極端に送信に時間がかかった例は, イーサネットでパケットの衝突が発生し CSMA/CD 方式により遅延が発生したためだと考えられる．

図 4 と図 5 を比較してわかるように, 双方のシステムでデータ転送時間のばらつきは小さい．集中方式では, 平均転送時間が 0.958mscc で, 転送時間のばらつきは上下に 0.06mscc 程度である．一方 DDFC に基づくシステムでは, 平均転送時間が 2.004msec でばらつきは上下に 0.05mscc 程度である．

4.3 評価

実時間制御システムにおいては, 処理にかかる時間の最悪値がみつもれ, かつ最悪の場合と最良の場合とのばらつきが小さいことが望ましい．この意味では, 本論文で提案する DDFC 方式は従来から集中方式に基づくシステムと同等もしくはそれ以上の性能をもつといえる．集中方式では当然ながらひとつのノードに負荷が集中するが, DDFC 方式ではこの問題が解消されている．したがって大規模プラントにおけるようにセンサ数が数万点をこえるような場合には, 負荷が集中するノードが性能上のボトルネックとなる従来方式よりも DDFC 方式を採用する方がよい．負荷が集中するノードがないので, すべてのノードは安価な計算機で構築できるので, コスト的にも DDFC 方式の方が有利である．すべてのノードが, あらゆるデータを蓄えたひとつのノードにアクセスするため取り扱えるノード数に限りがある．しかし DDFC 方式では, この問題はおこりえない．DDFC は集中方式に比べ, 約 2 倍の転送時間を必要とした．しかしデータの周期が数秒程度のシステムにおいては無視できる程度の差である．この結果,DDFC 方式が集中方式より劣る点の影響は非常に小さく DDFC の有効性を確認できた．

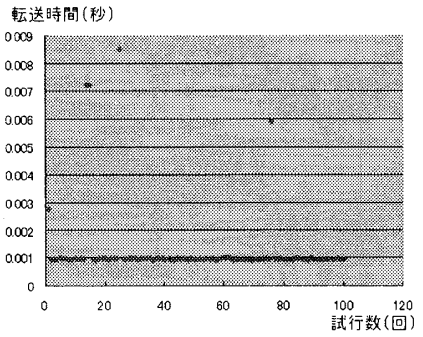

図 4: 集中方式

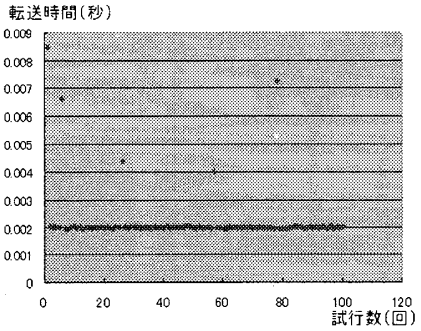

図 5: DDFC

5. おわりに

本稿では, 大規模制御システムにおける, 動的経路制御方式を実現するために必要不可欠な手法とその有効性を示した．今後,DDFC の実装をより進め検証を行い, 実用レベルに到達することを目指す．

参考文献

[1] 岡部寿男，藤川賢治，“ユビキタスネットワーク実現のための QoS ルーティング技術” 43 巻，6 号，情報処理，2002 年，pp.645-652.

[2] Guohong Cao, Liangzhong Yin, Chita R. Das “Cooperative Cache-Based Data Access in Ad Hoc Networks” 2004 IEEE Published by the IEEE Computer Vol. 37, No. 2; Feb 2004, pp. 32-39

[3] Ian Foster, Robert L.Grossman “Data integration in a bandwidth-rich world” Communications of the ACM Vol. 46, No. 11; 2003, pp. 50-57