プログラミング—言語・基礎・実践— 3－8
（1991. 7. 16）

協調処理言語 Cellula/C における仮想共有空間機構

吉田紀彦　　下川俊彦
九州大学工学部情報工学科

我々が提案している協調処理モデル Cellula は共有空間上のデータ－パターン照合を通信の基礎としている．したがって，このモデルの分散処理系を構築する上では，分散アーキテクチャ上に仮想的に共有空間を実現する仮想共有空間機構を組み込むのが有用である．パターン照合通信はデータ構造が更新されないという特徴を持ち，仮想共有空間の実現を一般の仮想共有メモリに比べて容易なものにしている．現在開発中の処理系第2版 Cellula /C では，第1版よりも対象範囲の広い仮想共有空間機構を実現することを目指している．

Virtual Shared Space Mechanism in Cellula/C Cooperative System

Norihiko Yoshida　　*Toshihiko Shimokawa*
Department of Computer Science and Communication Engineering
Kyushu University
Hakozaki, Fukuoka 812, JAPAN

Pattern-directed communication of our cooperation model *Cellula* is based on data-pattern matching in shared space, so its distributed implementation requires a virtual shared space mechanism which realizes shared space virtually on a distributed environment. This mechanism is easier to implement than ordinary virtual shared memory mechanisms, since pattern-directed communication never rewrite data structures. We are now making the second distributed implementation, and it will have more applicable virtual shared space mechanism.

1. はじめに

一般に仮想(分散)共有メモリ機構[Li89, Stum90]とは,物理的な共有メモリを持たないアーキテクチャの上で論理的な共有メモリを仮想的に構築する機構をいう.すなわち,この機構はバス結合型疎結合アーキテクチャ,ネットワーク・アーキテクチャの上でシステム全体に渡る一様なメモリ・アドレスを提供し,なおかつ,密結合アーキテクチャにおけるキャッシュ技法を応用することによって通信量の低減を実現するものである.

我々が提案している協調処理モデルCellula[Yosh90]は,主体間相互作用としてパターン照合に基づく通信を採用している.すなわち実行主体間の通信は,ある環境領域内に書かれたデータとパターンの照合によってなされる(なお,本モデルではこの主体と環境とがセルとして不可分に一体化している).この環境は主体間の共有空間として存在する.そこで,Cellulaをワークステーション・ネットワーク上の処理系として具体化するにあたって,仮想共有メモリ機構を応用してこの共有空間を仮想的に実現することを図った(なお,ここではアドレスによる低レベルのアクセスではなくパターン照合による高レベルのアクセスを対象とするため,仮想共有メモリではなく仮想共有空間と呼ぶ).

本稿ではこの処理系の第2版であるCellula/Cについて,その仮想共有空間の機構を述べる.以下,第2章では一般の仮想共有メモリ機構について説明し,第3章では協調処理モデルCellulaについて本稿に関連する部分を概観する.そして,第4章でCellula/Cの仮想共有空間機構について述べる.第5章はまとめである.

2. 仮想共有メモリ・アルゴリズム

一般に仮想共有メモリの重点は通信量の軽減にあり,通信データを一時的に自ノードで局所的に保持する,またはあらかじめ放送しておくなどの手法でこれを達成する.これを実現するアルゴリズムは幾つか存在するが,次の3つが代表的である(括弧内は仮の呼称)[Stum90].なお,ここではバス/ネットワーク上の各プロセッサをノード,互いに通信可能なノードからなるシステム全体をドメインと呼ぶ.

① [サーバ方式]システム内に唯一の共有メモリ・サーバを置く.共有メモリに対するすべてのアクセスはこのサーバに集中する.

② [複製方式]すべてのノードに共有メモリの同一コピーを持たせる.読込みは各ノード内で局所的に行う.一方,書込みはドメイン全体に放送される.

③ [分散方式]共有メモリをデータ・ブロック(ページなど)に分割して各ノードに割り当てておく.ノードは自分の持つブロック以外にアクセスする場合,その持主からコピーを受け取って自ノードのメモリに納める.以後の読込みはノード内で局所的に行う.書込みが生じるとそのコピーが持主に返され

るとともに,そのブロックの無効化(invalidation)メッセージが他のノードに放送される.

サーバ方式は最も単純な方法であるが,サーバがドメインのボトルネックとなる.

キャッシュの場合と同様に,複製方式では書込みメッセージ,分散方式では無効化メッセージの衝突が問題となり,通常はこれを解決するためにデータ・ブロックごとにアービタをおく.したがって,サーバ方式でのサーバと同様にアービタがドメインのボトルネックとなる.この回避手段として,例えば和田ら[Yama90]は論理的二分木構造を導入した階層的アービトレーションを提案している.

また,複製方式では書込みメッセージを実現するために"確実な"放送(reliable broadcast)の機構が提供されている必要がある.分散方式での無効化メッセージはマルチキャストによっても実現することができる.

一方,データ・ブロックへのアクセスには,書き手が唯一か複数か,読み手が唯一か複数かなど様々なパターンがある.そして,そのすべてに唯一のアルゴリズムで対処するのは有効ではないことが指摘されている[Stum90, Benn90].そして,実行に先だって前処理でデータ・ブロックのアクセス・パターンを解析して,それぞれに適当な仮想共有メモリ・アルゴリズムを割り当てるというシステムも,構想として提案されている[Benn90].

3. Cellulaにおけるパターン照合通信

パターン照合通信は,実行主体がある環境領域内にそれぞれデータないしパターンを書き,データ-パターン間の照合が成功するとパターンを書いた主体にそのデータが渡される,という通信形態である.これは人工知能やエキスパート・システムでいうプロダクション・システムに端を発するものと考えることもでき,例えば黒板システム[Enge88],並列処理言語Linda[Carr89]などの基盤としても広まりつつある.

環境領域は,黒板,グローバル・データベース,タプル空間などとも呼ばれ,それを共有する主体から参照可能な掲示板に例えることもできる.主体は環境内に何らかのデータを書くこと,または何らかのパターンを書いてそれに合致するデータを読み込むことができる.この掲示と検索・収納によって情報の授受がなされる.なお,データおよびパターンは適当な形で構造化されている.対応するデータのないパターンを書いた主体はそのようなデータが現れるまで待つのが普通であり,また逆に,対応するパターンのないデータを書いた主体はそのようなパターンが現れるのを待つことがある.

このように通信相手がデータ-パターン照合によって決定されるため,パターン照合通信は共有メモリ通信(アドレス指定)やメッセージ通信(宛先指定)よりも

自由度の高い情報授受が実現できる.すなわち,全体的な組織形態や制御構造が各主体の自律的判断に委ねられるような応用に対して特に有効である.

Cellulaでは具体的に通信のための命令として,次の2つを用意している.

out (environment, data, {attributes})	データを書く.
in (environment, pattern, {attributes})	パターンを書いて対応するデータを読み込む.

attributes : exclusive / non-exclusive, blocking / non-blocking など

これは基本的にはLindaの命令を踏襲している.しかし,環境が局所化されているのでこれを指定しなければならないところと,Lindaのinとreadに見られるような通信の種別を,命令ではなくデータ/パターンの属性で区別するところが異なり,属性の組合せによってより多様な通信方式を実現している.

特に,データ/パターンのいずれかが排他(exclusive)属性ならば照合成立後にデータは環境内から消去される.したがって,データが排他属性ならば1対1通信が実現する.一方,いずれもが非排他(non-exclusive)属性ならばデータは環境内に存在し続けるため,1対多通信が実現する.

なお,本稿ではデータ/パターンを領域に置くことを"書く",領域から取り込むことを"読む"と記し,より抽象度の高い操作,すなわちデータを領域に書くことは"提示する",パターンを書いて対応するデータを読むことは"収納する"と記す.

4. Cellula処理系の仮想共有空間機構

ワークステーション・ネットワークなどの分散環境上でパターン照合通信を具体化する処理系は一般に,仮想共有メモリ機構を応用して,通信媒体である環境領域を仮想共有空間として実現している.例えば分散黒板システムの実現[Jaga89]には,第2章でまとめた仮想共有メモリ・アルゴリズムにそれぞれ対応する次の3つの方式がある.

① 黒板サーバ方式(blackboard server approach)

② 複製黒板方式(replicated blackboard approach)

③ 分散黒板方式(distributed blackboard approach)

また例えばLindaの疎結合アーキテクチャ上での処理系では,代表的なものとして,確実な放送の機構を持つアーキテクチャの上に複製方式で実現したS/Net's Linda Kernel[Carr86]がある.

Cellulaの分散処理系は,Ethernetで結合されたUnixワークステーション・ネットワーク上にすでに第1版[Yosh91]を作成済みであり,今は第2版を作成中である.第1版では排他通信(排他属性のデータ/パターンによる通信)のみを対象とする仮想共有空間機構を実装し,通信量低減についての効果を確認した.そこで,第2版

では次の方針に従って,より対象範囲の広い仮想共有空間機構を実現することを目指している.

① 第1版ではTCP/IPを基盤としたが,第2版では通信効率改善のためにUDPを基盤とする.TCP/IPは1対1の確実な通信機構である.一方,UDPは1対多の通信が可能であるが着信保証がないため,不確実な放送機構とみなすことができる.

② 排他通信と非排他通信に対して別個のアルゴリズムで対処する.Cellulaではアクセス・パターンが属性としてプログラム中に明示されるため,アルゴリズムの使い分けは容易である.

③ 後に触れるように第1版では高度なアルゴリズムを用いて効果を絞り出したが,第2版では処理系の読解性・保守性を重視する立場からアルゴリズムの単純化を目指す.

Cellula処理系の仮想共有空間機構では,データは物理的なページなどのブロックではなく,通信単位であるデータ構造(具体的にはタプルまたはキー付値と呼ぶ)ごとに管理する.重要なこととして,データ構造は共有空間内に存在するかしないかのいずれかであって更新されることがない.したがって,管理単位をデータ構造とすることはアルゴリズムの設計にあたって利点となる.

以下,非排他通信と排他通信のそれぞれについて,設計中の仮想共有空間アルゴリズムを述べる.なお,ここでは領域の本体(環境セルのプロセス部)の置かれているノードをホーム・ノードと呼ぶ.

(1) 非排他通信のための仮想共有空間アルゴリズム

書かれたデータをすべてのノードが読む可能性があるため,複製方式を基本に分散方式を加味したアルゴリズムを用いる.

すなわち,非排他属性のデータは書かれた時点でドメイン全体に放送される.ただし,UDPという不確実な放送機構を用いるため,通信量は1対1通信と変わらないという大きな利得があるが,反面すべてのノードにコピーが渡る保証はない.そこで,ホーム・ノードからのみ“着信確認(ACK)”を要求するものとし,言うなれば“原本”がホーム・ノード上に存在するようにしておく.より詳しく説明すると:

① [データが書かれた場合]対応するパターンをまず自ノード内で探し,あれば局所的に収納させる.なければドメイン全体に放送し,かつホーム・ノード内で対応するパターンを探す.

② [パターンが書かれた場合]対応するデータをまず自ノード内で探し,次いでホーム・ノード内で探す.対応データが自ノード内にないとは,本当に存在しない/放送が不確実でコピーが届いていない,のいずれかを意味する.そこ

で,いずれ対応データがホーム・ノードから自ノードに渡された際に,それを局所的に留め置いてコピーとし,以降の照合に備える.

②の後半が分散方式に相当するが,データが更新されることがないため,無効化メッセージの必要はないという大きな利得がある.

なお,非排他データであっても対応するパターンが排他ならば,そのデータは消去される.この処理はホーム・ノードがドメイン全体にそのデータの消去メッセージを放送することで行う.ここでの放送は確実でなければならず,他のすべてのノードから着信確認を要求することで,不確実な放送機構(UDP)の上でシミュレートする.このように非排他データの消去処理は多大な通信量を要するが,これはプログラムの後処理部で生じるのがほとんどである.

(2) 非排他通信のための仮想共有空間アルゴリズム

非排他通信は1対1であるが,書かれたデータをどのノードが読むかは事前には判らない.そこで,同一ノード内で提示と収納がなされる場合にホーム・ノードとの間で生じる通信を吸収することを図る.

処理系第1版に実装した仮想共有空間機構では,この目的のために次の2つのアルゴリズムを組み合わせて用いた.

① データ/パターンのホーム・ノードへの送出を一定時間だけ遅らせ,その間に対応するパターン/データが自ノード内で書かれたならば局所的に収納させる.

② ①の後,パターンのホーム・ノードへの送出の際にそのコピーを自ノード内に残し,対応するデータが自ノード内で書かれたならばコピーとの照合で代用する.

①で導入する遅延はパターン照合通信の順序に影響を及ぼすが,プログラム実行のセマンティクスを変化させることはない.例えば汎用仮想共有メモリ・システムMunin [Benn90] では(目的は異なるが)やはりデータ・ブロックの送出を遅らせるアルゴリズムを採用し,これを“緩い一貫性”(loose coherence)管理と呼んでいる.

しかし,遅延時間を決定する際の理論的根拠がないこと,遅延という実時間性の導入が処理系のこの部分のみを特別なものにしていること,さらに第2版で新たに実装する非排他通信用アルゴリズムとの整合性などを考慮し,第2版では上の①を廃して②の改良のみで対処する方向で考えている.より詳しく説明すると:

① パターンが書かれたならば,対応するデータをまず自ノード内で探し,あれば局所的に収納させる.なければホーム・ノードに送出し,同時にそのコピーを自ノード内に残す.

② ホーム・ノード内で照合が成立するとパターンに対応するデータが自ノードに送られてくる.パターンのコピーがまだあれば(自ノード内での照合

がまだ成立していないから)そのデータを読み込み,なければ(自ノード内での照合がすでに成立しているから)そのデータを必要に応じてホーム・ノードに送り返す.

非排他用/排他用のいずれのアルゴリズムも,これらを用いない例えばサーバ方式などの処理系と比較して,非排他データの消去の場合を除いては通信量が増えることはない.

5. おわりに

パターン照合通信は環境領域内のデータ-パターン間照合に基づくため,その領域を仮想共有空間として実現するのが有用である.逆に,パターン照合通信はデータ構造が更新されないという特徴を持ち,仮想共有空間の実現を一般の仮想共有メモリに比べて容易なものにしている.また,UDPという不確実な放送機構の上でも有効なアルゴリズムを可能にしている.

今後は本稿で示した2つの仮想共有空間アルゴリズムのより詳細な検討と,処理系への実際の組み込みを行っていく予定である.

謝辞

協調処理モデルCellulaの設計支援および処理系第1版作成を行って頂いた現NTTソフトウェア研究所の楢崎修二氏に謝意を表する.

参考文献

[Benn90] Bennett,J.K., Carter,J.B. and Zwaenepoel,W., "Munin : Distributed Shared Memory Based on Type-Specific Memory Coherence", Proc. Second ACM SIGPLAN Symp. on Principles and Practice of Parallel Programing (PPoPP), Seattle (1990) 168-176.

[Carr86] Carriero,N. and Gelernter,D., "The S/Net's Linda Kernel", ACM ToCS 4:2 (1986) 110-129.

[Carr89] Carriero,N. and Gelernter,D., "How to Write Parallel Programs : a Guide to the Perplexed", ACM Comp. Surv. 21:3 (1989) 323-358.

[Enge88] Engelmore,R. and Morgan,T. eds., *Blackboard Systems*, Addison-Wesley (1988).

[Jaga89] Jagannathan,V. et al. eds., *Blackboard Architectures and Applications*, Academic Press (1989).

[Li89] Li,K. and Hudak,P., "Memory Coherence in Shared Virtual Memory Systems", ACM ToCS 7:4 (1989) 321-359.

[Stum90] Stumm,M. and Zhou,S., "Algorithms Implementing Distributed Shared Memory", IEEE Comp. 23:5 (1990) 54-64.

[Yama90] 山崎剛, 和田耕一, "疎結合並列計算機における仮想共有メモリの実現", 並列処理シンポジウム JSPP '90 論文集, 筑波 (1990) 9-16.

[Yosh90] 吉田紀彦, 楢崎修二, "場と一体化したプロセスの概念に基づく並列協調処理モデル Cellula", 情報処理学会論文誌 31:7 (1990) 1071-1079.

[Yosh91] 吉田紀彦, 楢崎修二, "協調処理モデル Cellula の分散処理系", 情報処理学会論文誌 32:7 (1991) 掲載予定.