異機種分散処理環境におけるインタフェース自動生成処理方式
(3)　エンコード、デコード方式

7Q−4

田中伸明　関根徹　末広亮太　榎本典行　本田邦夫
松下電器産業(株)　情報通信東京研究所

1. はじめに

[1]で報告した基本処理方式において、転送出来るデータ構造の種類とデータ転送フォーマット、および、エンコード、デコードのアルゴリズムについて述べる。特にポインタの転送方法に特徴があるので、それを中心に述べる。

2. 他システムにおけるポインターの転送

従来、NCS[2],MiG[3],rpcgen[4]等が開発され、分散プログラム開発用ツールとして広く利用されている。現在では、これらのツールはそれぞれの分散プログラミング環境中で、不可欠のものとなっている。

RPC用I/Fジェネレータを使う場合、そのジェネレータが、ポインタを使って表現されたデータ構造を、どのように扱うかによって、実際に使用する場合の開発者の負担が大きく変わる。

現存するRPC用I/Fジェネレータには、ポインタでつながれたリスト構造、ツリー構造などを転送することはできないという制約を持つものがある。また、リスト、ツリーを扱うことができるものでも、2つ以上のポインタが、1つのデータ領域を指している場合に、転送時にそれぞれのポインタごとにコピーが行われ、転送先ではそれぞれのポインタは、別の領域を指す場合がある。(図1)つまり、送信側と受信側では、データ構造に違いが生ずる。また、このことより、このようなツールは、双方向リスト、環状リストなど、参照関係にループがある場合は無限にエンコードを続けようとする。

参考文献[5]のシステムでは、このようなデータ構造を保持したままの転送が可能である。しかし、[5]がサポートしている言語は、CLU言語のみである。CLU言語では、すべてのデータは、そのデータの存在する領域を示すポインタで表され、今回、我々がターゲットとしたC,Pascalなどのように、ポインタとデータ本体の区別が明確な言語とは大きく違う。

3. 本処理方式におけるポインターの転送

本処理方式では、2つ以上のポインタが1つのデータ領域を指している場合にも、データのコピーは1回だけ行い、転送先でも、各ポインタは、同じデータ領域を指すようにした。(図2)このことにより、双方向リスト、環状リスト等のデータ構造も転送できる。

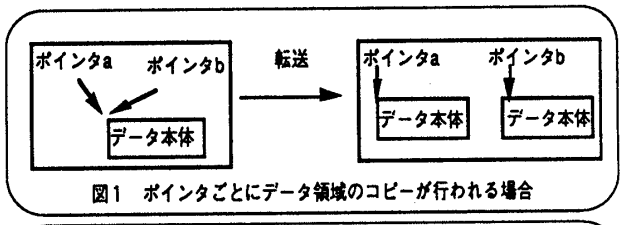

図1　ポインタごとにデータ領域のコピーが行われる場合

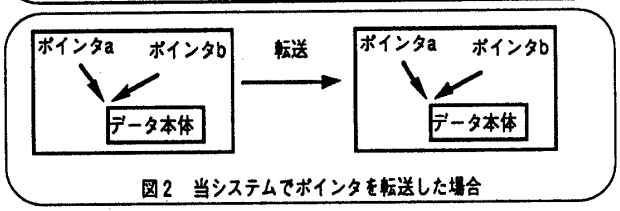

図2　当システムでポインタを転送した場合

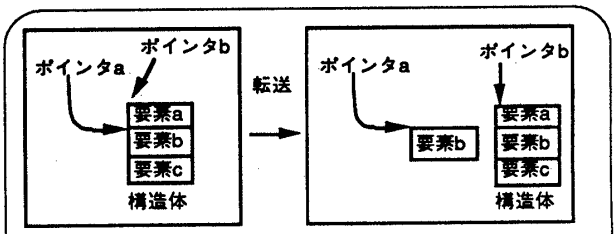

図3　転送できない場合
(構造体の要素をポインタが指している。)

ただし、構造体、配列のようなデータ構造中を指しているポインタがある場合、そのデータ構造は、転送時に保持されない。(図3)これは、ポインタが先にデコードされる場合に、後からデコードされる構造体、配列の格納される領域を知ることができないため、ポインタのデコード時に、そのポインタの値を決められないからである。

4. エンコード、デコードアルゴリズムの目標

転送フォーマットと、エンコード、デコードのアルゴリズムは、次のことを目標として設計した。

1) エンコード時に、ポインタでつながれたデータ構造のトラバースをする回数が1回ですむようにする。

2) 通信バッファへデータを入れていく際に、先にバッファに入れたデータや、あとから入れるデータを参照しない。

An Automatic Interface Generator for Distributed Processing Systems in Heterogeneous Environment
(3)　Encoding and Decoding
Nobuaki TANAKA, Toru SEKINE, Ryota SUEHIRO,Noriyuki ENOMOTO, Kunio HONDA
Matsushita Electric Industrial Co., Ltd.

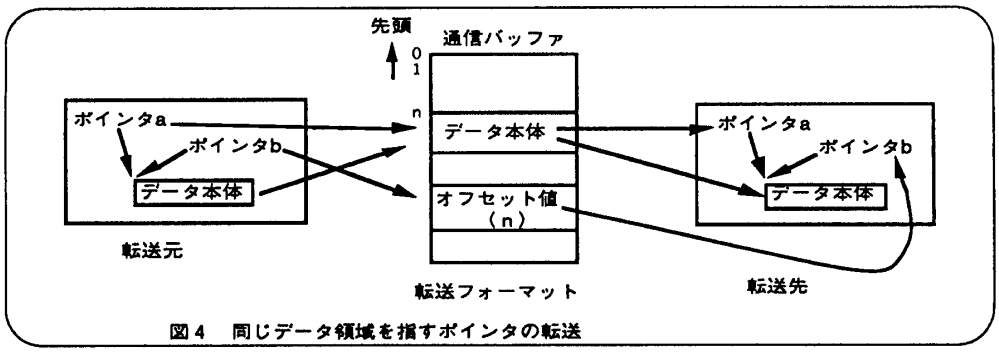

図4　同じデータ領域を指すポインタの転送

1) は、効率向上のためである。また、2) により、通信バッファ中にデータを入れる動作も、バッファに先頭からデータを追加していくだけで、前後の通信バッファ中をアクセスする必要がない。よって、大量のデータを転送する場合に、すべてのデータのエンコードが終わる前に、すでにエンコードの終わった部分から、順に転送することが可能となる。

5. 転送フォーマットと、エンコード、デコード方法

5.1 転送フォーマット

データ転送フォーマットは、基本的にはASN.1を用い、RPCのインターフェースとなる関数の引数を、そのデータ型に従ってASN.1フォーマットにマッピングしている。ただし、ASN.1ではポインタのような参照関係を表す方法がない。そこで、ポインタは次節に示すようにエンコードしている。（図4参照）

5.2 エンコード方法

エンコード時には、すでにエンコードしたポインタの情報を保持しておくテーブル（エンコードデータ情報管理テーブルと呼ぶ。）を用いる。ポインタのエンコード方法は次の通り。

a.1) ポインタをエンコードする場合はそのポインタが、すでに通信バッファ中にあるかどうかを「エンコードデータ情報管理テーブル」を検索して調べる。

a.2) まだ、エンコードされていないポインターであれば、そのデータ本体をエンコードして、通信バッファに追加し、そのポインタの値と、そのデータの、通信バッファ中での、先頭からの距離（オフセットと呼ぶ。）を「エンコードデータ情報管理テーブル」に登録する。このとき、通信バッファ中のデータには、ポインタであることを示すtagを付けておく。

a.3) すでにそのデータがテーブルに登録されていれば、データ本体のエンコードは行わず、前にエンコードしたデータの、通信バッファ中のオフセットのみをエンコードして、通信バッファに追加する。このデータにはオフセットであることを示すtagを付けておく。

5.3 デコード方法

デコード時にも、すでにデコードしたポインタの情報を管理するための「デコードデータ情報管理テーブル」を用いる。ポインタのデコード方法は次の通り。

b.1) デコードを行うときに、tagが a.2で付けたtagであれば、そのデータをデコードし、メモリ上に格納する。転送先でのポインタは、そのデータ領域を指したものにする。このとき、「デコードデータ情報管理テーブル」に、そのデータの通信バッファ中でのオフセットと、ポインタを登録しておく。

b.2) tagが a.3で付けたtagであれば「デコードデータ情報管理テーブル」を検索して、すでにデコードされているデータを指すポインターを求める。

これらの管理用テーブルには、ハッシュ技法を用いて検索の高速化を図っている。

5. まとめ

今回発表した手法により、C言語等の一般的な言語で、ポインタを使って表されたデータ構造を、そのまま転送できるRPC用I/Fの生成が可能になった。また、この手法では、転送するデータのトラバースを1回行うだけでエンコードが行える。ただし3章で述べたように、まだ転送できるデータ構造に制限があるため、効率を落とさずに、転送できるデータ構造の種類を拡大できるアルゴリズムの開発が今後の課題である。

なお、本研究開発は、通商産業省工業技術院大型プロジェクト「電子計算機相互運用データベースシステムの研究開発」の一環として、新エネルギー産業技術総合開発機構（NEDO）より委託を受けて実施したものである

参考文献

[1]末広　他："異機種分散処理環境におけるインタフェース自動生成処理方式　（1）基本処理方式"、情報処理学会第41回全国大会講演論文集

[2]"Network Computing System (NCS) Reference",Apollo Computer Inc.,1987.

[3]"NeXT System Reference Manual",NeXT,Inc.

[4]"rpcgen Programming Guide" , Sun Microsystems, Inc. , 1988 .

[5] M.Herlihy,B.Liskov,"A Value Transmission Method for Abstract Data Types",ACM Transactions on Programming Languages and Systems , Vol.4 , No.4 , pp527-551,October 1982.