Internal Use Only(非公開)

TR-I-0163

素性構造の単一化に基づくパーサの並列化方式
A Parallel Method For Unification-based Parsers

加藤 進

Susumu KATO

1990.5

概要

素性構造の単一化に基づくパーサを高速化する手段の一つとして並列処理の方法が考えられる。マルチプロセッサシステムは、密結合型や疎結合型などの構成方式によって処理特性に違いがあるため、使用するマルチプロセッサシステムの処理特性とアプリケーションプログラムの持つ並列性との適合度を考慮する必要がある。本報告では、素性構造の単一化に基づくパーサの並列性を検討し、マルチプロセッサシステム上で、並列実行する方式について述べる。また、その方式の効果について、実際に、マルチプロセッサシステム上で評価した結果について報告する。

ATR Interpreting Telephony Research Laboratories
ATR 自動翻訳電話研究所

©ATR Interpreting Telephony Research Laboratories

©ATR 自動翻訳電話研究所

1. はじめに

近年、言語理論として、素性構造の単一化に基づく文法理論(単一化文法–Unification-based grammar)が発展し、自然言語処理の分野でも、これが取り入れられ、いくつかのシステムが作られている。単一化文法の枠組みを用いる利点として、以下の2点があげられる。

① 異なる部門に由来する統語論、意味論、語用論などの情報を素性構造に展開し、その素性構造と単一化という形式的体系の中で、複雑な言語現象を統一的に扱うができる。

② 語彙-統語的な文法の記述は、モジューラー性、拡張性が高く、新しい語彙項目、および、新たな語彙項目への情報の追加が容易となる。

その一方で、単一化文法の枠組みは、システム化する上で、計算量の問題を考える必要がある。効率の良い統語解析アルゴリズムと効率の良い単一化演算アルゴリズムを採用し、単一化パーサを実現しても、文法が大規模になってくると計算量が増大し、必要とされる解析時間が得られなくなる。このような単一化パーサで行われる大量の情報計算を高速に処理する手段の一つとして並列処理の方法が考えられる。

最近のマイクロプロセッサの技術革新には、目をみはるものがあり、すぐれた処理能力を持つマイクロプロセッサが次々と開発されている。そして、そのマイクロプロセッサを複数個使用したマルチプロセッサシステムが、商用の並列計算機として利用可能になってきた。マルチプロセッサシステムを利用する場合には、解くべき問題に内在する並列性を抽出し、要素プロセッサへ効率的に負荷分散することを考慮しなければならない。この点を考慮しないと、使用するマルチプロセッサシステムの不得意とする処理のオーバーヘッドのために、理論上の並列度が打ち消されることになる。

本報告では、素性構造の単一化に基づくパーサの並列性を検討し、その並列性を効率的に実行する方式を述べる。また、その方式をマルチプロセッサシステム上で、実際に、評価した結果について報告する。

2.単一化パーサの並列性

単一化パーサの処理には、図01に示すような3つのタイプの並列性が存在する。これらの並列性は、独立したものではなく、階層的な性質を持つ。その詳細を以下に述べる。

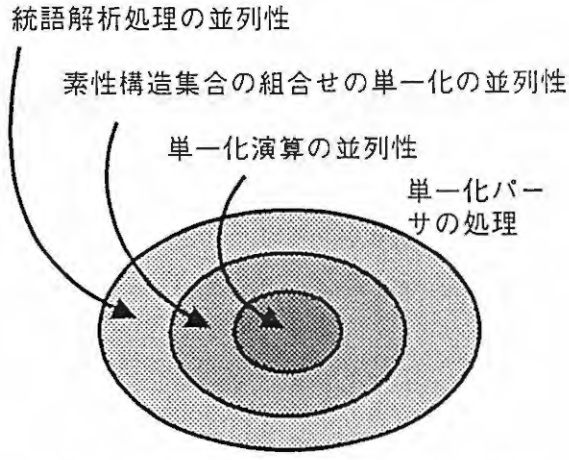


図01 単一化パーサの並列性の階層

2.1.統語解析処理の並列性

統語解析処理の中に並列性を見いだすことができる。これまでに、報告されている文脈自由文法の統語解析処理の並列性に着目したものには、一般化LRの並列化法、米澤と大澤の提案するオブジェクト指向的な方法、CKYの並列化法、松本のPAXなどがあげられる。これらの手法は、

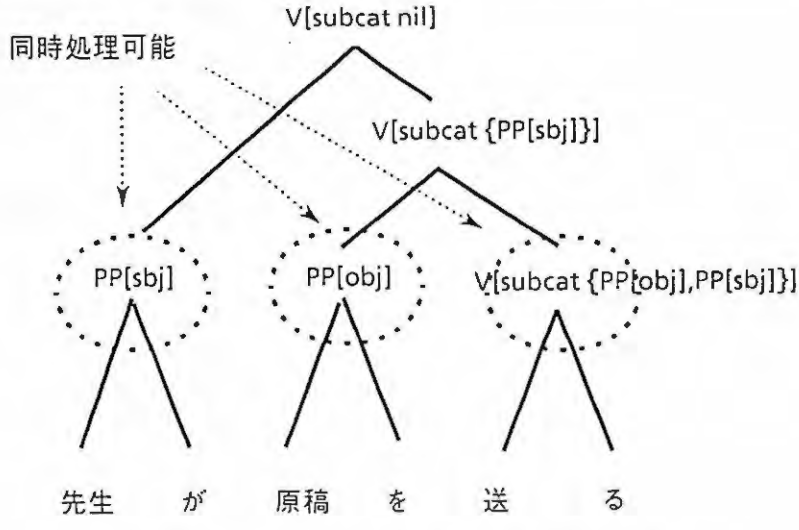


図02 (1) 統語解析処理の並列性の例

① 解析する文に曖昧性がある場合
② 解析木の部分構造がバランスしている場合

などに並列効果が現れる。図02に示す破線の部分は、解析木の部分木としてバランスしているので、同時に計算が可能である。

2.2.素性構造集合の組合せの単一化の並列性

単一化文法では、同音異義語の語彙定義、動詞の格パターンの違いによる下位範疇化の記述、意味的な違いの記述、形態素解析結果の違いなどのさまざまな要因によって、単一化パーサの解析過程では、素性構造集合と素性構造集合による組合せの単一化が適用されることになる。図03の例で、動詞の定義中に、格パターンの違いを表現するfs1とfs2の2つの素性構造が存在したとする。その場合、組合せ的に2種類の単一化が適用されるが、それらの処理は、同時に、計算が可能である。単一化パーサの分析で、解析過程に、このような素性構造集合の組合せの単一化が数多く適用されることが認められている。

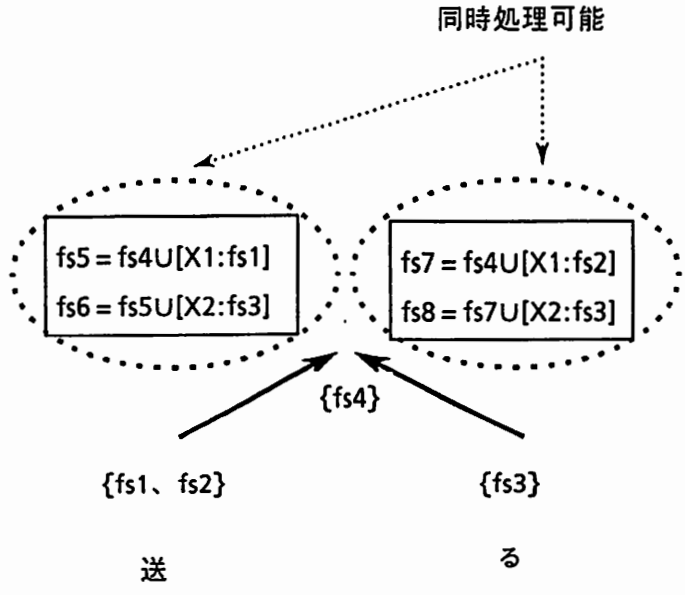
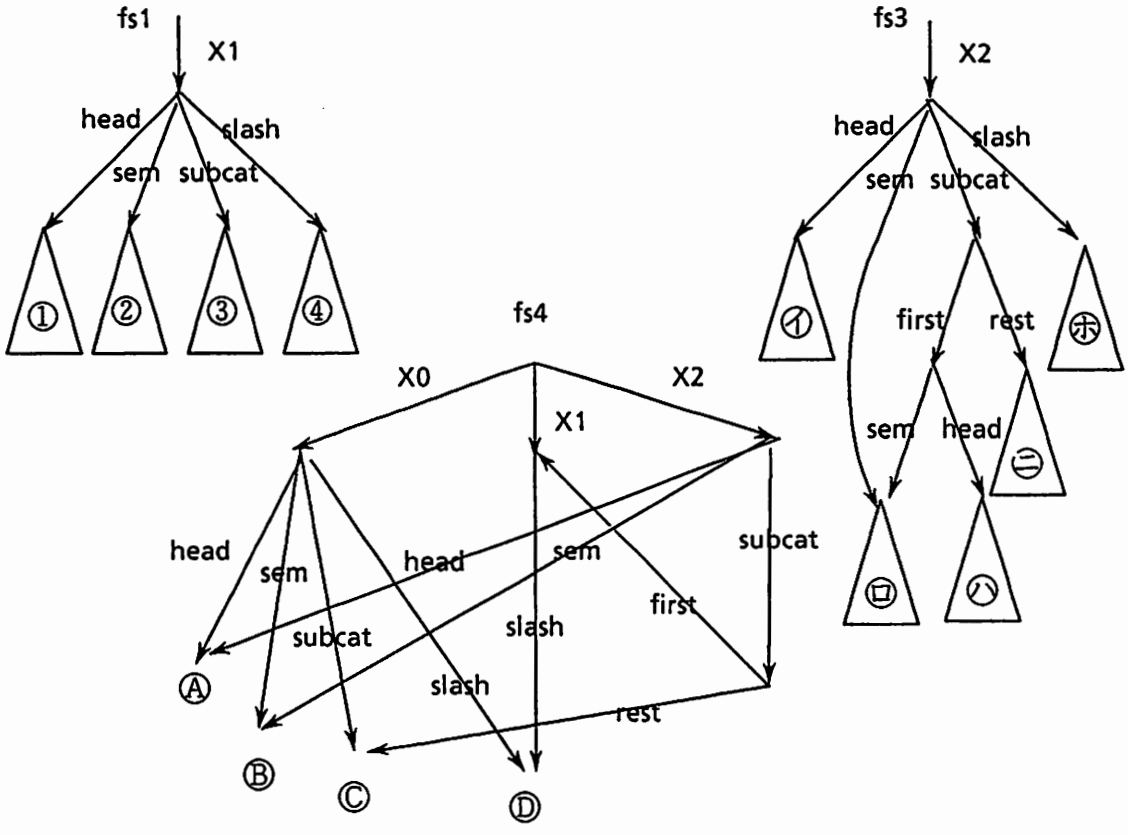


図03 素性構造集合の組合せの単一化の並列性の例

2.3.単一化演算の並列性

解析の基本演算である単一化の中に並列性が存在する。図04に示す単一化演算の例で、fs1は、統語規則の右辺の第一項へ単一化

される素性構造、fs3は、統語規則の右辺の第二項へ単一化される素性構造、fs4は、統語規則の素性構造をそれぞれ表現している。この例では、それぞれの素性構造の表現する制約条件を分解することによって、head、sem、subcat、slashの素性の部分的な単一化演算を同時に処理することが可能となる。単一化パーサの解析処理全体の中で、単一化演算の占める比率が非常に高いため、単一化演算を並列化することができれば、高速化が期待できる。



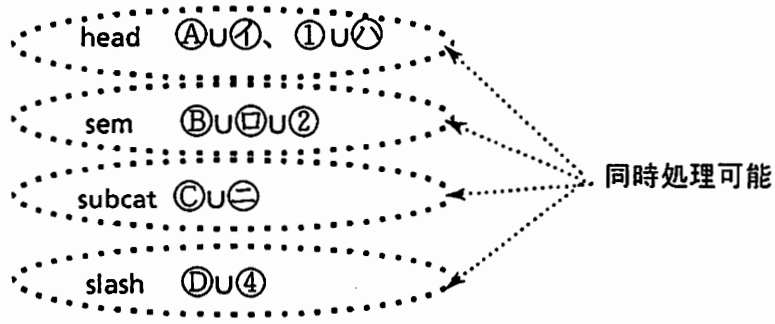
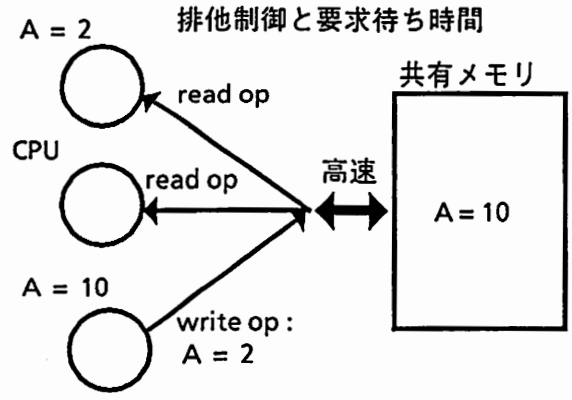


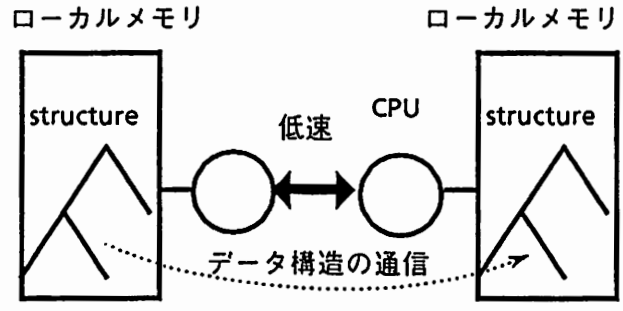
図04 単一化演算の並列性の例

3.マルチプロセッサシステムの処理特性

マルチプロセッサシステムは、ハードウエアの構成方式に着目すると、おおまかに密結合型と疎結合型の2つに分類することができる。



(1)密結合型のマルチプロセッサシステム



(2)疎結合型のマルチプロセッサシステム

図05 マルチプロセッサシステムのオーバヘッド

密結合型のマルチプロセッサシステムは、共有メモリ方式を採用し、情報交換には、プロセスで共通の変数を用いる。したがって、複数の要素プロセッサから同一のデータに対してアクセスの競合が発生するため、排他制御を行う必要がある(図05(1))。このような競合の排他制御を実現する機構としてlockやsemaphoreなどが用意されている。これらの機構は、適切な使い方をしないと、プロセスがデッドロックにおちいってしまったり、長時間待ち状態になるプロセスが発生することになる。このタイプのシステムは、データの共有が容易であるという長所を持つが、その反面、メモリアクセスは、すべて、バス、クロスバなどのハードウエアを通して行うことになるので、要素プロセッサ数は数十程度が限界になる。

疎結合のマルチプロセッサシステムは、分散メモリ方式を採用し、情報交換は、要素プロセッサ間でのメッセージパッシングによ

り行われる。要素プロセッサ間の通信速度が低速であるため、頻繁に素性構造のようなデータ構造を他の要素プロセッサへ転送する処理は、通信量が多くなり、全体の処理効率の低下につながる(図05(2))。商用の計算機の中には、ハイパーキューブ、メッシュなどの結合方式で要素プロセッサが、数百から数千台のオーダーで構成されるものが存在する。そのため、実行するアプリケーションの処理単位が、要素プロセッサごとに分割することができれば高い台数効果を得ることができる。

4.並列パーサの構想

前述した単一化パーサの並列性を効果的に実行するマルチプロセッサシステムのハードウエアの構成を図06に示す。このシステムは、密結合された共有メモリのCPUのクラスタをさらに疎結合した構成になっている。したがって、疎結合型のマルチプロセッサシステムの高並列性と密結合型のマルチプロセッサシステムのデータの共有の容易さの両方の長所を利用することができる。このハードウエアの構成は、CPUクラスタを一つの要素プロセッサと考えると、疎結合型のマルチプロセッサシステムと見ることができるので、疎結合型のマルチプロセッサシステムの特性から、CPUクラスタへ割り付ける処理は、できるだけデータ構造の転送を含まないものが望まれる。

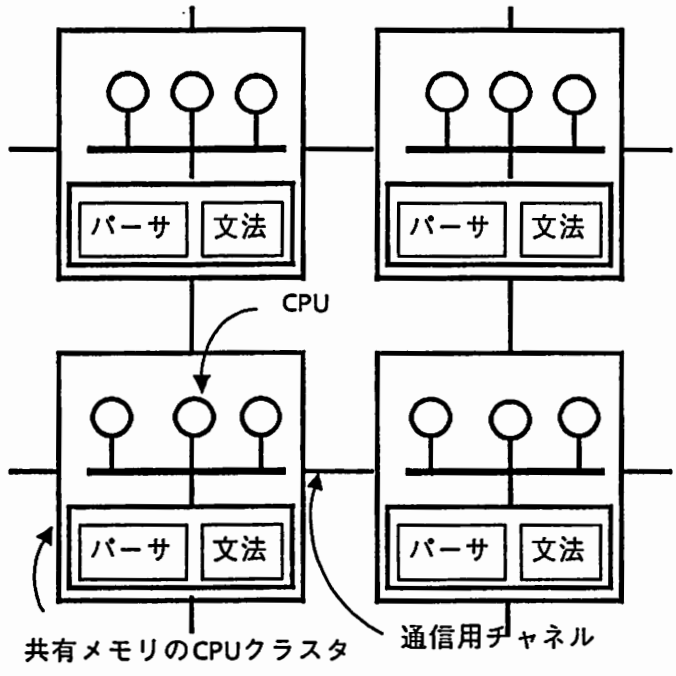


図06 マルチプロセッサシステムの構成

パーサは、構文解析過程で、CPUクラスタ間での通信を最小限におさえ、さらに、CPUクラスタ内の処理の負荷を軽減することを目的とした

① 素性構造集合の組合せの単一化をCPUクラスタへ分散する方式

② 統語解析処理と単一化演算をCPUクラスタ内で並列実行する方式

の2段階の戦略をとる。①の方式は、マルチプロセッサシステムの各CPUクラスタ上に、文法をコンパイルすることによって作成した素性構造とパーサプログラムを配置し、それぞれのCPUクラスタで構文解析処理を行わせる。そして、その時に適用される素性構造集合の組合せの単一化の処理で、CPUクラスタ番号と素性構造集合の要素数の関係から、単一化すべき素性構造集合を分割することによってCPUクラスタへ単一化の組合せを分散させる方式である。②の方式は、①の方式によって軽減された処理をさらに、統語解析処理と単一化演算の持つ並列性によって高速化する方式である。以下に、それらの方式の詳細を述べる。

4.1.組合せの単一化を分散する方式

この方式の基本的な動作原理は、素性構造集合に、その素性構造集合がどのCPUクラスタに存在しているかを示すCPUクラスタ属性を設定し、単一化すべき素性構造集合をCPUクラスタ属性の示すCPUクラスタへ均等に分割しながら単一化を適用することにある。CPUクラスタ属性を含む素性構造集合の構造体は、

```
type FS-LIST  = record
        DECOMPOSED-FS-LIST
        CLUSTER-ATTRIBUTE
end;
```

である。DECOMPOSED-FS-LISTは、分割された素性構造集合である。CLUSTER-ATTRIBUTEは、この素性構造集合が存在するクラスタを管理するスロットでありSTART-CLUSTERとEND-CLUSTERの対の値をとる。

図07に示すような解析過程の部分的な処理の例を用いて、素性構造集合の単一化を分散化する方法を説明する。この例では、組合せによって、②の単一化処理で4つ、③の単一化処理で8つの結果を逐次処理で生成する。ここでは、8つのCPUクラスタを用いる。各CPUクラスタは、自分のCPUクラスタ番号と全体のCPUクラスタ数を知ることができる。

まず最初に、①の処理では、各CPUクラスタで、素性構造集合{fs1、fs2}を{fs1}と{fs2}に分割する。{fs1、fs2}の要素数2で全体のCPUクラスタ数8を割った商4をCPUクラスタ属性の幅になるように、CPUクラスタ属性を設定する。したがって、{fs1}には、CPUクラスタ0から3、{fs2}には、CPUクラスタ4から7のCPUクラスタ

属性が設定される。各CPUクラスタは、自分自身のCPUクラスタ番号をCPUクラスタ属性値に保持する素性構造集合を分割された素性構造集合の中から選択する。したがって、その状態は、

	クラスタ番号	0	1	2	3	4	5	6	7
①	素性構造番号	1	1	1	1	2	2	2	2

のようになる。また、その時の分割された素性構造集合は、

クラスタ0から3
　DECOMPOSED-FS-LIST　：　{fs1}
　CLUSTER-ATTRIBUTE　：　0,3

クラスタ4から8
　DECOMPOSED-FS-LIST　：　{fs2}
　CLUSTER-ATTRIBUTE　：　4,8

の内容になる。

次に、①の処理で分割された{fs1}、{fs2}と{fs3、fs4}の単一化処理の②では、{fs3、fs4}を{fs1}と{fs2}のCPUクラスタ属性の示す各CPUクラスタに均等に分散するようにする。例えば、CPUクラスタ0では、CPUクラスタ属性によって、{fs1}がCPUクラスタ0から3

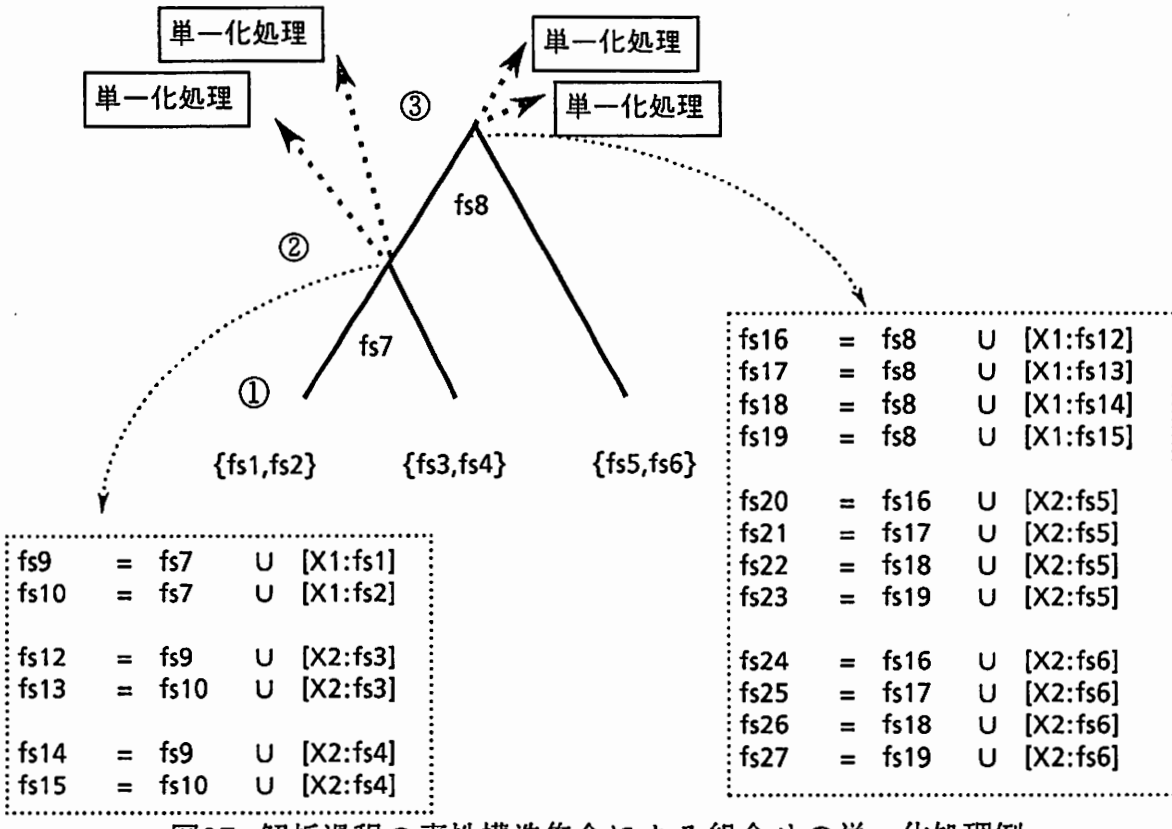


図07　解析過程の素性構造集合による組合せの単一化処理例

に存在していることがわかるので、{fs3、fs4}の要素数2でCPUクラスタ数4を割った商2をCPUクラスタ属性の幅になるようにする。したがって、{fs3}には、CPUクラスタ0から1、{fs4}には、CPUクラスタ2から4のCPUクラスタ属性が設定される。各CPUクラスタは、自分自身のCPUクラスタ番号をCPUクラスタ属性値に保持する素性構造集合を分割された素性構造集合の中から選択するので

	クラスタ番号	0	1	2	3	4	5	6	7
①	素性構造番号	1	1	1	1	2	2	2	2
②	素性構造番号	3	3	4	4	3	3	4	4

の状態になる。その時の、②で作成される素性構造集合は、分割された素性構造集合のCLUSTER-ATTRIBUTEの値を継承して、

クラスタ0から1
　　DECOMPOSED-FS-LIST　：{fs12}
　　CLUSTER-ATTRIBUTE　：0,1

クラスタ2から3
　　DECOMPOSED-FS-LIST　：{fs14}
　　CLUSTER-ATTRIBUTE　：2,3
　　.....

が、それぞれのCPUクラスタに作成される。

同様な処理を{fs12}、{fs14}などの分割で作成された素性構造集合を使って進めていくと、③の処理では、

	クラスタ番号	0	1	2	3	4	5	6	7
①	素性構造番号	1	1	1	1	2	2	2	2
②	素性構造番号	3	3	4	4	3	3	4	4
③	素性構造番号	5	6	5	6	5	6	5	6

クラスタ0
　　DECOMPOSED-FS-LIST　：{fs20}
　　CLUSTER-ATTRIBUTE　：0,0

クラスタ1
　　DECOMPOSED-FS-LIST　：{fs24}
　　CLUSTER-ATTRIBUTE　：1,1
　　.....

の状態になる。③の処理で作成された{fs20}や{fs24}のCLUSTER-ATTRIBUTEのSTART-CLUSTERとEND-CLUSTERの値は、等しいので、それ以上の素性構造集合の分割は行われない。結局、素性構造集合の組合せは、図08のように各CPUクラスタで分散されて実行されることになる。

このように各CPUクラスタで素性構造集合の分割を続けながら単一化を適用していくと、組合せの単一化を複数のCPUクラスタに分散することになる。分割された素性構造集合は、解析過程で、他の統語規則の適用によって単一化が行われることになるので、さらに並列効果が向上することになる。これらの手続は、他のCPUクラスタへ素性構造の転送を行わなずに、ローカルのCPUクラスタメモリ内で素性構造集合から素性構造を選択する処理により解析が進められるので高速化が可能となる。

4.2.CPUクラスタ内の並列化方式

CPUクラスタ内では、統語解析処理と単一化演算の並列化が行われる。以下に、それらの方式について述べる。

4.2.1.統語解析処理の並列化方式

統語解析アルゴリズムは、JPSGの統語規則がチョムスキー標準形に従っているので、CKY法を並列実行する方式を採用している。配列の要素を一つのプロセスとして動作させ、素性構造へのポ

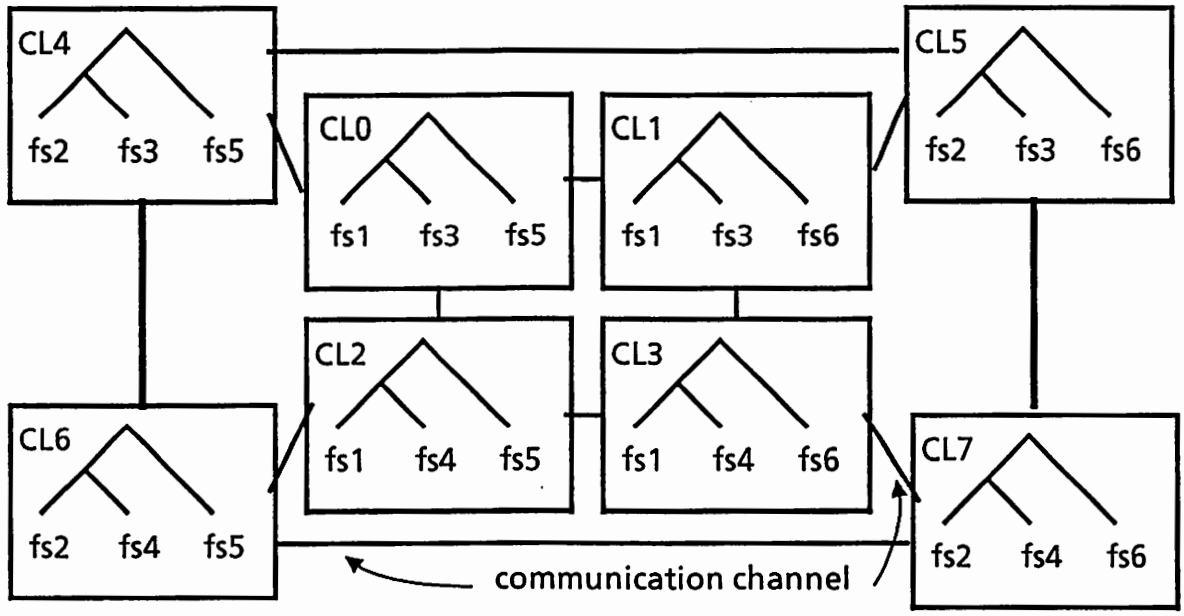


図08 各CPUクラスタの並列実行の様子

インタをメッセージとして送信することによって解析を行う。メッセージの構造は、

```
type MESSAGE  = record
        TO-CATEGRY
        TO-FS-LIST
        FROM-PROCESS
end;
```

である。TO-CATEGRYは、どういう統語カテゴリにまとめあげられているかを配列要素プロセスに知らせるために用いる。TO-FS-LISTは、形態素解析された語彙から導出される素性構造集合や配列要素プロセスが生成する素性構造集合にクラスタ属性を付与した構造体へのポインタである。FROM-PROCESSは、メッセージが、どこの配列要素プロセスから送られてきたかを示す。その情報によって統語規則の右辺の第一項か第二項への素性構造であるかを判断する。

図09に、文が解析される様子を示す。配列要素プロセスの処理手順は、最初に、送られてきた統語規則の右辺の第一項へのメッセージのカテゴリと第二項へのメッセージのカテゴリから新しいカテゴリを導出する統語規則が存在するかを調べる。もし、その統語規則が存在すれば、CPUクラスタへ単一化の組合せを分散する方式と単一化演算を並列化する方式を適用しながら、単一化処理を行う。そして、単一化が成功する場合にのみ、結果の素性構造をメッセージ化して、送るべき配列要素プロセスに送信する。

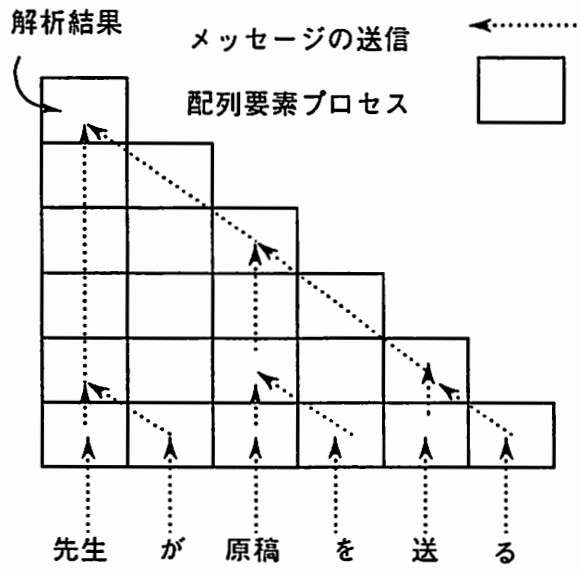


図09 CKY法による解析

統語規則を一つのプロセスとする方式(図10)も考えられる。しかし、統語規則をプロセス化する方式では、日本語を解析する場合、名詞と助詞を扱うプロセスへメッセージが集中し、統語解析処理の持つ並列性を十分に発揮できなくなる可能性がある。

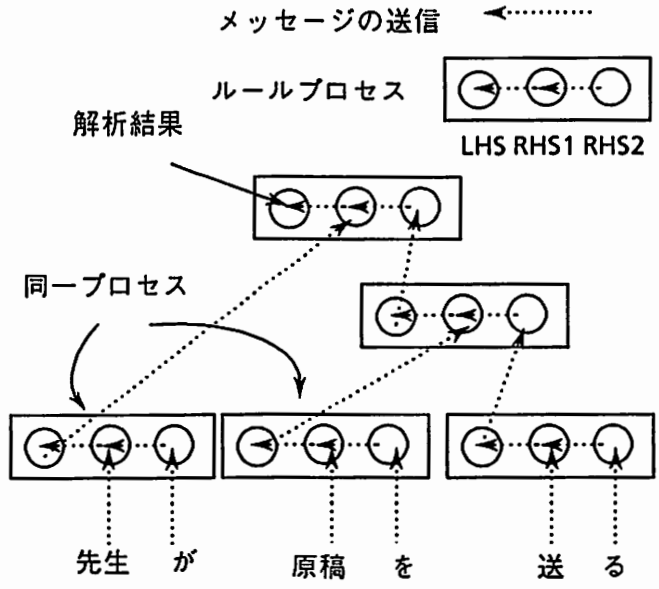


図10 統語規則をプロセス化した場合の解析

4.2.2.単一化演算の並列化方式

一回の単一化演算を部分的な単一化演算に分割するアルゴリズムを図11に示す。ノード構造は、基本的に、

```
type NODE   = record
        ARC-LIST
        PARENT
        FORWARD
        COPY
        VALUE
        CHECK-LIST
        BUFFERING-LIST
end;
```

である。ここで、ARC-LISTは、素性と素性の示すノードの対の集合である。PARENTは、このノードを参照する素性と親のノードの対の集合である。FORWARDは、forwarding操作に使用される。forwarding操作とは、ノードを参照する時の優先順位を変える操作

```
function    PARALLEL-UNIFY(INPUT-NODES);
    INPUT-NODESの全てのノードに対してforwarding
    操作を適用する;
    if  (INPUT-NODESにアトミックノードが含まれている)
        then
            if  (INPUT-NODESの全てのノードが
                アトミックノードであり、かつ、等しい値を持つ)
                then
                    return(INPUT-NODESのどれか1つのノード):
                else
                    単一化は失敗;
    COPIED-NODE←GET-COPY-NODE(INPUT-NODES);
    INPUT-NODES←APPEND(COPIED-NODEのBUFFERING-LIST,INPUT-NODES);
    if  (INPUT-NODESの各ノードが全ての親ノードから参照された)
        then
            SORTED-ARCS←SORT-ARCS(INPUT-NODES);
            for (FEATURE,NODES)  in  SORTED-ARCS  paralleldo
            RESULT←PARALLEL-UNIFY(NODES);
            COPIED-NODEのARC-LISTにARC(FEATUREとRESULT)を追加する;
        else
            COPIED-NODEのBUFFERING-LISTにNODESを追加する;
    return COPIED-NODE;
end;


function    GET-COPY-NODE(INPUT-NODES);
    COPIED-NODE ←
        if  (INPUT-NODESの全てのノードがコピーノードを持っていない)
            then   ノードを作成する;
            elseコピーノードのどれか一つを選択する;
    for NODE  in  INPUT-NODES  do
        if  (NODEがコピーノードを持っている)
            then
                そのコピーノードをCOPIED-NODEへforwardingする;
                そのコピーノードのBUFFERING-LISTのノードをCOPIED-NODEの
                BUFFERING-LISTに追加する;
                NODEのCOPYスロットにCOPIED-NODEをセットする;
    return COPIED-NODE;
end;

function    SORT-ARCS(INPUT-NODES);
    RESULT    ←  INPUT-NODESの各ノードのARC-LISTをFEATUREに
                  よって分類する;
    return RESULT;
end;
```

図11 単一化演算の並列アルゴリズム

である。そのノードの保持している情報を無視し、FORWARDが示しているノードを参照することである。COPYは、このノードの複製されたノードを保持する。VALUEは、ノードがアトミックノードの場合に、素性値を保持する。CHECK-LISTは、このノードが全ての親ノードから参照されたかどうかを調べるために用いられる。BUFFERING-LISTは、全ての親ノードから参照されるまで、単一化処理を待機させるために、単一化されるべきノードの集合を保持する。

このアルゴリズムの特徴は、単一化すべきノードが、全ての親ノードから参照されるまで単一化処理を遅らせ、プロセス間での構造の競合を少なくしている点にある。例えば、図12の単一化演算の例では、Xの素性による②と⑤のノードの単一化処理のプロセスは、②がYの素性によっても参照されているので、停止する。②と⑤のノードは、Yの素性の単一化処理によって、⑦のノードと併合されて単一化処理が初めて適用されることになる。

このアルゴリズムは、かなり処理の粒度が小さい。そのため、一回の単一化の計算が、プロセスを生成する負荷よりも十分に大きい場合や分散処理用のOSであるMACHのスレッドのようなlight weight processを用いてアルゴリズムを実現する場合などに利用可能となる。

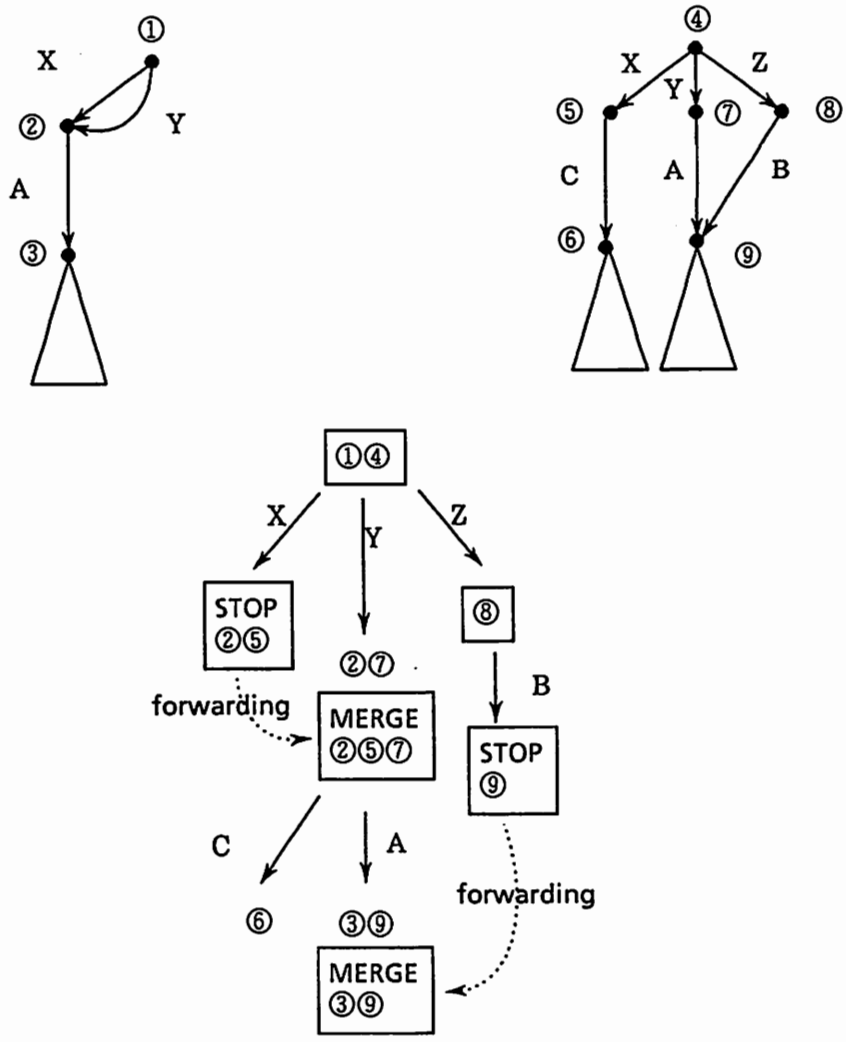


図08　単一化演算の処理例

5.並列パーサの効果

組合せの単一化を分散する方式は、疎結合型の並列計算機で、CPUクラスタ内の並列化方式は、密結合型の並列計算機で、別々に評価する実験を行った。本報告で示した単一化パーサの並列性は、階層的な性質を持つので、総合的な並列度は、それぞれの並列度を掛け合わせた数値で考えることができる。

5.1.組合せの単一化を分散する方式の効果

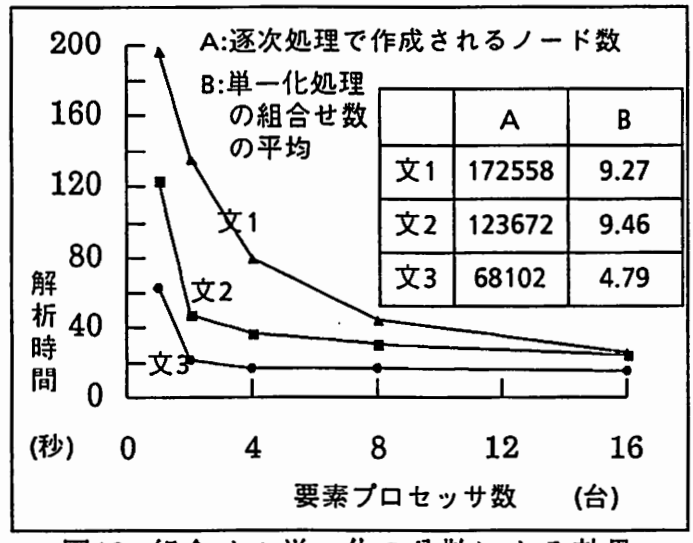
組合せのの単一化を分散化するパーサをインテル社製の疎結合型の並列処理計算機であるiPSC2(CPUチップi80386とローカルメモリから構成される要素プロセッサが、ハイパーキューブ結合で最大128台まで拡張可能)上に実現し、文の解析速度を測定する実験を行った。パーサは、Common Lispに通信機能の拡張を施した"iPSC2 LISP"で記述しており、統語解析アルゴリズムは、CKY法、単一化アルゴリズムは、Wroblewski法を用いている。文法は、日本語を解析するために作成したもので、統語規則数23、語彙定義数53(総語彙数74)で構成されている。一個の素性構造には、平均31.7個のノードが含まれている。

解析文

文1　それではこちらからそちらに登録用紙を至急にお送り致します。

文2　こちらからそちらに登録用紙をお送り致します。

文3　お名前とご住所をお願いします。



	A	B
文1	172558	9.27
文2	123672	9.46
文3	68102	4.79

図13　組合せの単一化の分散による効果

測定時間を図13に示す。文1では、16台の要素プロセッサを用いて7.5倍程度高速化することができた。この方式は、単一化の組合せ数が多くなれば、それだけ台数効果も向上するものと考えられる。

5.2.CPUクラスタ内の並列処理方式の効果

現在、2つのCPUクラスタ内の並列処理方式のうち、統語解析処理の並列化方式を実現したプロトタイプパーサが完成している。簡単な文法を用いて、文の解析速度を測定する実験を試みた。並列計算機は、シークエント社製の密結合型の並列計算機であるSYMMETRY(CPUチップi80386と共有メモリをバス結合し、要素プロセッサは、最大30台まで拡張が可能)を使用した。パーサは、C言語で記述しており、統語解析アルゴリズムは、CKY法、単一化アルゴリズムは、Wroblewski法を用いている。文法は、文を解析する時の並列度が高くなるように意図的に作成したもので、統語規則数35、語彙定義数18(総語彙数18)で構成されている。解析文4は、終端記号が6つから構成され、それぞれの終端記号は、3つの素性構造を導出する。文4の並列度は、シミュレータによって、約10倍程度の並列度を持つことがあらかじめ分かっている。

測定時間を図14に示す。文4では、8台の要素プロセッサを用いて5倍程度高速化することができた。台数効果が頭打ちになっている原因として、パーサ側の問題としては、分析の結果、プロセス間におけるメッセージの通信に時間がかかっていることがわかった。システム側の問題としては、バス要求の待ち時間やキャッシュの

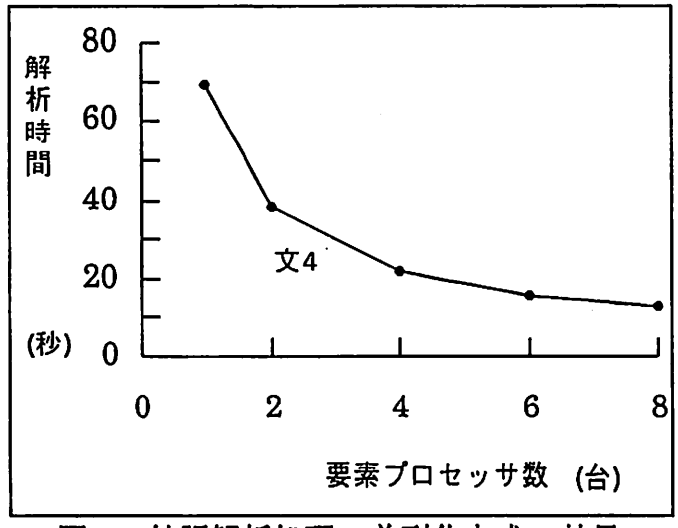


図14 統語解析処理の並列化方式の効果

ヒット率などのハードウエアの物理的な特性が影響しているものと考えられる。また、プロセスのスケジューリングの時間も考えられる。

6. おわりに

本報告で、素性構造の単一化に基づくパーサの3つの階層的な並列性を考察し、疎結合型と密結合型のマルチプロセッサの長所を併せ持つハードウエア上で、素性構造の単一化に基づくパーサを並列実行する方式を述べた。また、その方式の並列効果を調べるために、並列計算機を用いた実験を行い、その結果について報告した。

計算機アーキテクチャの発展予想として、1990年代は、並列処理の時代といわれており、今後、マルチプロセッサシステムが、単一CPUの計算機に取って変わっていくものと思われる。このような時代の流れにそって、並列計算機を有効利用した翻訳システムの構築に、本報告が役立つことを期待する。

謝辞

本研究の機会を与えてくださるとともに適切な助言を述べられたATR自動翻訳電話研究所 樽松 明 社長、ならびに、同言語処理研究室 飯田 仁 主幹研究員に感謝します。

参考文献

[1]Wroblewski,D.:Nondestructive graph unification. in sixth national conference on artificial intelligence .

[2]Kasper,R.:A Unification method for Disjunctive Feature Descriptions. in proceedings of 25th Annual Meeting of ACL.

[3]Yonezawa and Ohsawa:Object-Oriented Parallel Parsing for Context-Free Grammars. in proceedings of COLING'88

[4]加藤、小暮(1987):「素性構造の単一化手法の効率」情報処理学会研究会資料NL64-9

[5]沼崎、田中、田村(1989):「並列論理型言語による一般化·LR構文解析アルゴリズムの実験」情報処理学会研究会資料NL74-5

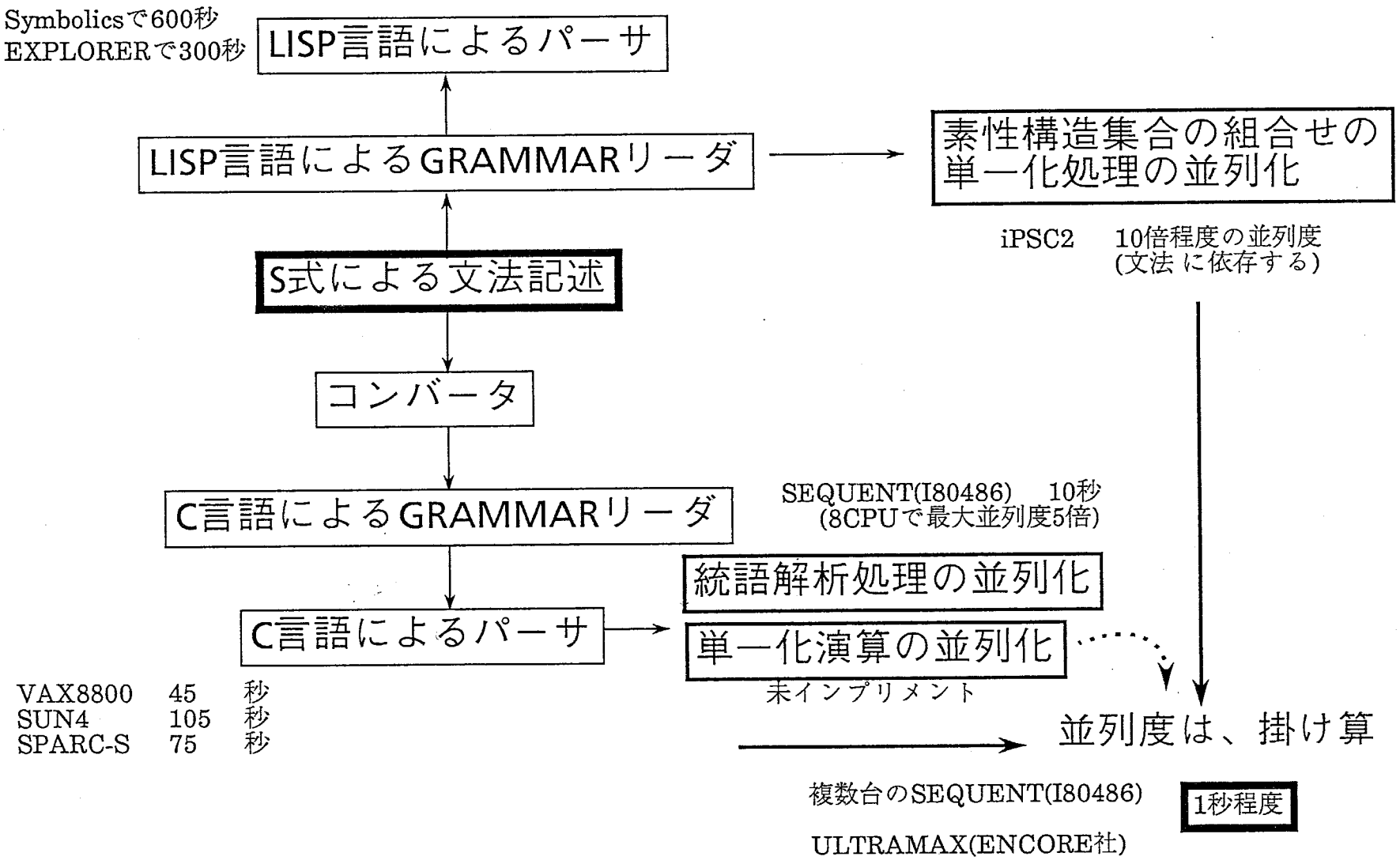
付録1　商用の並列計算機の動向

1988年度までに購入可能であった主な並列処理計算機の性能

	iNTEL社 iPSC2	NCUBE社 N-CUBE	BBN社 GP1000	Sequent社 SYMETTRY	Alliant社 FX/8	ENCORE社 MALTIMAX
CPUの種類	i80386	専用CPU	m68020	i80386	(m68020)	NS32532
1CPUの能力	4MIPS	2.5MIPS	2.5MIPS	4MIPS	(2.5MIPS)	8.5MIPS
結合方式	ハイパーキューブ	ハイパーキューブ	多段スイッチ	バス結合	クロスバ	バス結合
最大CPU数	128	1024	256	30	8	20

1989年度以降に購入可能になった(なる)主な並列処理計算機の性能

	iNTEL社 iPSC860	NCUBE社 N-CUBE2	BBN社 TC2000	(Sequent社) (予定)	Alliant社 FX/2800	Cogent Research 社 XLNT
CPUの種類	i80680	専用CPU	m88100	(i80486)	i80680	T800
1CPUの能力	35MIPS	7.5MIPS	19MIPS	(20MIPS)	35MIPS	5MIPS
結合方式	ハイパーキューブ	ハイパーキューブ	多段スイッチ	(バス結合)	クロスバ	バス＋クロスバ
最大CPU数	128	8192	63(504)	(20)	28	1900

付録2　並列処理による総合的な効果予想

文“分からない点がございましたら私どもにいつでもお聞きください。”の解析の例