階層ベイズクラスタリングアルゴリズムの並列化

福島 健一郎, 本田 岳夫, 奥村 学
北陸先端科学技術大学院大学 情報科学研究科
email:{fukusima,honda,oku}@jaist.ac.jp

1 はじめに

クラスタリングはデータ集合の内部構造（ベクトルデータ）のみによって分類（クラスタ分析）を行なう手法である。クラスタリングには分析の目的やその用途に応じて、いろいろな方法が提唱され、その分類もいろいろあるが、そのうちの一つが階層クラスタリングである。階層クラスタリングアルゴリズムは類似度をもとに樹系図（デンドログラム）と呼ばれる階層木を構成するものである。そのアルゴリズムは、一般に、(1) クラスタ間の類似度を計算する、(2) もっとも類似度の近いものを融合する、(3) すべてのアイテムが一つになるまでこの処理を繰り返す、という手順を踏む。ステップ1では、クラスタどうしのすべての組合せを計算する必要がある。すると、アイテム数が増えれば増えるほど、組合せの数は増大し、この時間計算量は深刻な問題になってくる。アイテム数を n とすると、一般に計算量は $O(n^2)$ である。

我々は現実の並列環境に実装可能な並列アルゴリズムを提案し、階層クラスタリングが持つ時間計算量の問題を解消する。そして、現実の並列環境に応じて自然言語処理のシソーラス自動構築に必要な何千、何万ものアイテムの処理が可能なシステムを目指す。我々は階層クラスタリングの一例として、Iwayama[1, 2]らの提案した階層ベイズクラスタリング (HBC) を用いて実装する。我々の提案する並列アルゴリズムはHBCを対象に実装されるが、本アルゴリズムはHBCに特化したものではない。重心距離を類似尺度に用いた階層クラスタリングアルゴリズムならば変更なしに適用できる。また、他の尺度を用いた階層クラスタリングアルゴリズムでも、少ない変更で適用できるはずである。

2 並列アルゴリズムの提案

2.1 並列化のポイント

階層クラスタリングは類似尺度の違いで様々な手法が取れるが、本研究の対象とするHBCは類似尺度に確率的な手法をとりいれている。ここではその詳細を省くが、HBCは二つのクラスタがマージしたと仮定した計算 ($SC(c_x \cup c_y)$ の計算) を行なう。このSC計算はマージされる可能性のあるクラスタの総組合せを計算する。

今、シーケンシャルアルゴリズムにおいて、まずクラスタの組み合わせを計算するSCはそれぞれが独立に計算可能である。さらに、そのSCそのもののベクトル計算も各ベクトル次元において独立に計算可能である。我々はこの二つの部分を並列化のターゲットにした。

この二つの部分を並列化したモデルを軸に、どちらか一方だけを並列化したモデルも考えることができる。今、ベクトル計算だけを並列化したモデルをモデル1、SC計算だけを並列化したモデルをモデル2、両者を同時に並列化したモデルをモデル3、と呼ぶことにする。

2.2 並列アルゴリズム

並列アルゴリズムは、SC計算、ベクトル計算をどのくらいの割合で並列化するかを指定して使うことになる。これがモデル3(1+2)であり、ベクトル計算、SC計算の並列度のどちらかを極端にしたモデルがモデル1、あるいはモデル2となる。

アルゴリズムの構造化のために、ここで抽象的なノードとして三つのノードを導入する。まず、クラス

タリングそのものを統括する働きのあるホストノードがある。これはデータの読み込み、ファイル出力、下位抽象ノードへのデータ転送、下位抽象ノードの同期などを担当する。このホストノードは、モデルに唯一つである。

次に、アイテムの組み合わせを担当するコントロールノード（以下、CN）がある。これは、ホストノードの下位に位置し、ホストと計算結果などの受け渡しを行う。また、各CNは自分が担当するアイテム組み合わせの計算（SC 計算）を下位の抽象ノードへ命令する。このCNの数はSC 計算をどれだけの割合で並列化するかによって決まる。

三つ目に、ベクトル計算を担当するベクトルノード（以下、VN）がある。これは、CNの下位に位置し、CNから命じられたアイテムの組み合わせ計算のうち自分が担当するベクトル次元だけを計算する。その計算結果をCNに転送する。このVNの数はベクトル計算をどれだけの割合で並列化するかによって決まる。

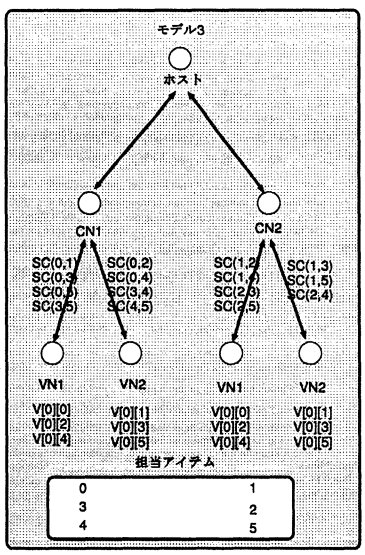

図 2.1:一般化並列モデル-モデル 3

モデル3 を例にアルゴリズムを簡単に説明する（図2.1）。まず、初期化作業として、それぞれのCNは自らが持つ担当アイテムに従ってSC の計算を下位のVNに指示する。例えば、CN 1 ならクラスタ 0、3、4からの組合せの計算を担当することになる。この担当アイテムによってSC 計算は分散される。また、その下に位置するVNはCNより指示されたSC のベクトル計算を行なう。この際、自分が持つベクトルデータで計算を行ない、その結果を上位のCNに送る。ベクトルデータは各VNに分散されているから、ベクトル計算もVNの数だけ分散されている。

この初期化作業が終了したら、もっとも良いペアをホストは決定し、マージとデータの更新に移る。仮にここでクラスタ 0 と 1 がマージされたとする。担当アイテムのクラスタ 1 が消去され、クラスタ 0 と 1 に関するSC も消去される。そして、新しくクラスタ 0 が更新され、クラスタ 0 との組合せを再計算する必要がある。この再計算は、初期化の時とは少し異なって、更新されたクラスタ 0 と担当アイテムの組合せを計算することになる。この処理を繰り返し、クラスタリングは進む。

モデル 1 は図 2.1 におけるCNの数が一個になり、VNが複数個（使用するプロセッサ数に相当する）、モデル 2 はCNが複数個（同じくプロセッサ数）、その下に位置するVNは一個になるだけで、モデル 3 と処理は同じである。

3 nCUBE/2 への実装

3.1 実験

我々の提案した並列アルゴリズムを検証するために、超並列計算機 nCUBE/2[5] へモデル 1 とモデル 2 を実装、評価した。モデル 3 はその両者を並列化したモデルであるから、その二つから推測することができる。評価に使用したデータは EDR 共起辞書 [4] からアイテムを名詞、その名詞に共起する動詞をベクトルとして抽出した。そして、上位何語かを頻度順で選び出し、以下のようなデータセットを用意した。また、nCUBE/2 はプロセッサ数を 2^n 個でしか確保することができないので、評価は log スケールでとってある。

set - (item,vector)

Set1 - (100,2500)	Set2 - (200,2500)
Set3 - (300,2500)	Set4 - (400,2500)

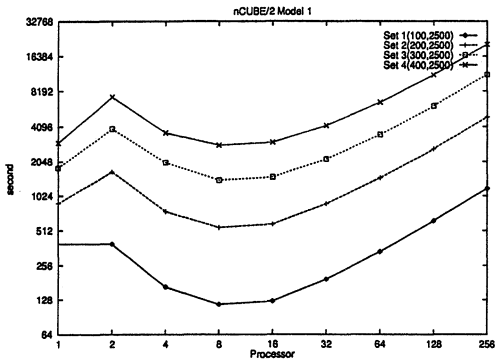

図 3.1:モデル 1 処理時間グラフ (Log scale)

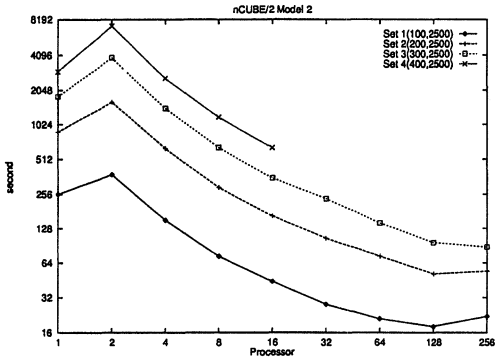

図 3.2:モデル 2 処理時間グラフ (Log scale)

3.2 考察

モデル 1 のグラフ（図 3.1）がプロセッサ数 8 個から上向きになっているのは、各VNが計算した結果をCNが集める操作が原因になっている。このCNの合計操作は通信コストがかかるため、プロセッサ数が増えると処理時間を増大させる原因になる。

モデル 2 はモデル 1 のような合計操作を必要としない。そのため、プロセッサ数を増やせば増やすほど処理時間は減少する（図 3.2）。

どちらのグラフもプロセッサ 2 台の時点で処理時間が増加している。これは nCUBE/2 の搭載メモリが少ないことから、ホストノードだけでプロセッサ 1 台を占有してしまい、2 台使用しているとは言え、結果的に処理ノードであるVNは 1 台になるため処理はシーケンシャルになる。シーケンシャルであるにもかかわらず、ノード間の通信コストがかかることから処理はシーケンシャル以上の時間を必要とする。

また、モデル 2 の Set4 が 16 台以降、プロットされていないのはメモリ不足から生じる問題である。本実験で使用した nCUBE/2 の 16 台まではメモリを 16MB 持つものの、残りのプロセッサは 4MB しか持たない。そのため、16 台以降は処理に必要なメモリを確保できなかったためである。逆にモデル 1 は Set4 においても 4MB で足りている。つまり、モデル 1 はベクトルデータを分散させて持つことから、モデル 2 よりメモリの消費量が少なくて済む。このことから、VNの数を増やすことでメモリを節約できることが実験結果からも分かる。

4 PVM への実装

4.1 実験

PVM[3, 7] はネットワークで接続された様々なコンピュータを一つの仮想並列計算機として使うことのできるソフトウェアである。我々はより大きなデータセットを扱うためとユーザの使用環境によらない並列モデルの実現のために、PVM へ実装した。PVM にはモデル 1、モデル 2 に加え、モデル 3 も実装した。実験に使用したデータセットは nCUBE/2 と同じものを使用した。また、PVM に使用した環境はイーサネットで接続された SUN の WS であり、個々の空きメモリや CPU パワーは異なっている。この実験において、マシン、ネットワークともユーザが使用していない理想的な条件が最大のパフォーマンスを引き出すと思われるが、今回、それは敢えて行なわなかった。我々は通常の条件のもとでの PVM 環境でのパフォーマンスがもっとも重要だと考えたからである。

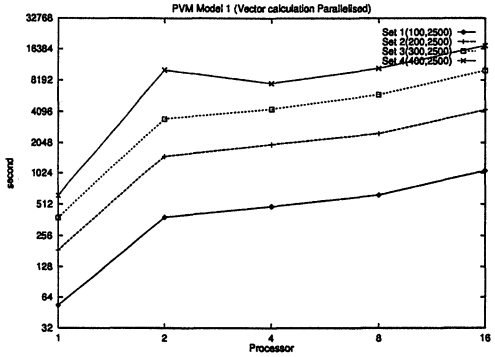

図 4.1 モデル 1 処理時間グラフ (Log scale)

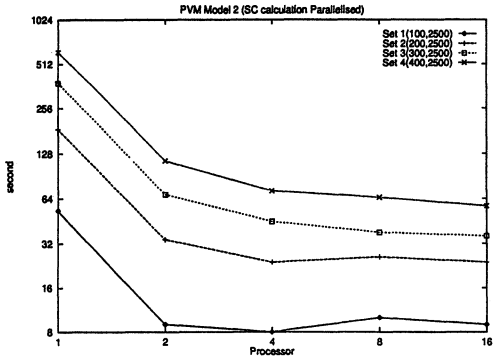

図 4.2 モデル 2 処理時間グラフ (Log scale)

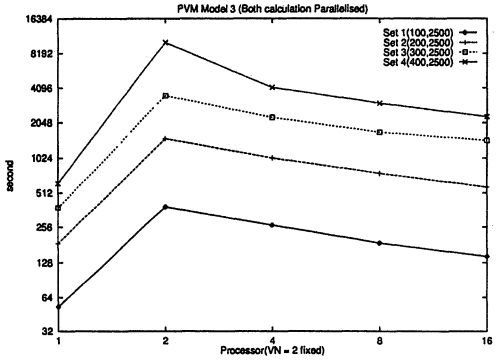

図 4.3 モデル 3 処理時間グラフ (Log scale)

4.2 考察

PVMによる実験結果もnCUBE/2と同様になった(図4.1、図4.2、図4.3)。ただし、通信速度がnCUBE/2よりも低速であることからモデル1のグラフが最初から右上がりになっている。

また、モデル3は2台の時点ではCNが1、VNが2であることからモデル1のそれに相当し、グラフは上昇する。しかし、4台からはCNが2、4、8と増えていくことでモデル2の要素が入り、グラフは下降することになる。モデル3は両者のモデルの特性を併せ持っていることがよく分かる。このことから、データ量やユーザの使用環境に応じてCNとVNのパラメータを設定することが重要だと考えられる。

5 結論

本稿では、階層クラスタリングアルゴリズムの並列化を提案した。そして、その一例として階層ベイズクラスタリングの並列アルゴリズムを実装することで、現実の並列環境で動作することも示した。

超並列計算機nCUBE/2と並列仮想環境のPVMの二つで実装した結果、ベクトル計算を並列化するよりもアイテムの組合せ計算を並列化した方が時間計算量の減少ができた。PVM環境のマシン16台でモデル2を使用した場合最大10倍の速度向上が得られた。また、ベクトル計算の並列度を上げることで、扱えるアイテム数の増加を果たすことが考えられる。今後の課題となるパラメータの自動決定などの手法の追加で、リーズナブルなモデルとして機能するように目指したい。さらに、PVM環境においては個々のマシン性能の違いを考慮した負荷分散を考えることでよりパフォーマンスを上げることができると思われる。

参考文献

[1] Makoto.Iwayama,Takenobu.Tokunaga Hierachical Bayesian Clustering for Automatic Text Classification,IJCAI95,2:1322-1327,1995

[2] T.Tokunaga ,M.Iwayama and H.Tanaka. Automatic Thesauri Construction Based on Grammatical Relations ,IJCAI95,2:1308-1313,1995

[3] Adam Beguelin,Jack Dongarra PVM and HeNCE: Tools for Heterogeneous Network Computing,Enviroments and Tools for Parallel Scientific Computing,139-153,1993

[4] 日本電子化辞書研究所、"EDR 電子化辞書使用説明書",1994

[5] Frontier利用者の会、北陸先端科学技術大学院大学情報科学センター利用の手引,1994

[6] nCUBE/2 Programmer's Guide 3.1,1992

[7] 村田英明訳,PVM3 ユーザーズガイド&リファレンスマニュアル,1994