PC クラスタ環境における並列分枝限定法

大西克実[†] 榎原博之[‡] 中野秀男[†]

平成 14 年 4 月 27 日

Abstract

最近の PC 及びネットワーク技術の発達により従来の並列コンピュータとは異なったメタコンピューティング環境上での分散処理がとりわけコストパフォーマンスの観点から注目されている．本報告では，学内のネットワーク環境を利用し，21 台の PC を利用したクラスタ処理環境を構築したのでその構成・特徴などについて報告する．また，構築した計算環境上で，並列分枝限定法を実行するシステムを用意しその効果について検証を行ったのでその結果も紹介する．

キーワード 分散処理環境，分枝限定法，組合せ最適化問題

1 はじめに

従来計算機センターなどで提供されているサービスでは，高速かつ大容量の記憶装置などが必要となる利用者のために大型の計算機を用意し提供してきた．最近では PC(パーソナルコンピュータ) 関連技術の発達により各研究室内での計算機の利用が進んでいる．論文作成・情報収集のためのクライアント計算機以外にも PC 上で動作する Linux に代表される PC-UNIX を利用したコンピューティング環境も比較的容易に構築できるようになっている．また，インターネット技術の発達によりネットワークを通じて複数の計算機を接続し，一つの計算機資源として利用する試み (メタコンピューティング) がよく利用されており，これを実現するための PVM[10] や MPI のようなライブラリも提供されている．これら分散環境におけるコンピューティング技法は今後も ITS での応用などますます必要になると思われる．

大阪市立大学学術情報総合センター内のネットワークを用いて接続された PC21 台を利用してこのような分散処理環境を構築したので，本報告ではその概要を報告する．さらに，システムの構成・構築の他に具体的な利用例として組合せ最適化問題 [5] を並列分枝限定法 [11] を用いて解く例を取り上げ並列化の効果についても紹介する．

組合せ最適化問題とは，交通機関における人員配置を考える問題や，生産工場でのスケジューリング・配送などの問題を一般化した問題である．組合せ最適化問題の中でも特に難しい問題を解くために適用される解法として分枝限定法が有効な手法として知られている．分枝限定法を実装する場合には，分枝変数・探索・再割当てなどの組合せにより様々な方法が考えられるが，解こうとする問題ごとに大きさが同じでもその難易度や分枝の様子が異なるため，ある特定の問題例に対して有効な方法が他の問題例では必ずしも有効ではない．並列分子限定法の場合，さらに並列化の方法も考慮する必要がある．本報告では，特に分枝木の構成に影響が大きい分枝変数の選択方法の違いが並列化の効果に与える影響について述べる．

本報告では，並列分枝限定法で解く具体的な組合せ最適化問題として巡回セールスマン問題を取り上げる．巡回セールスマン問題 (TSP) は，組合せ最適化問題の中で，問題の記述の容易さ，応用の広さなどから特に広く研究されている問題であり [7]，最適解を求めるアルゴリズムや，近似解を求めるアルゴリズムなどさまざまな解法が考えられている．

[†]大阪市立大学学術情報総合センター,onisi@media.osaka-cu.ac.jp
[‡]関西大学工学部情報処理教室

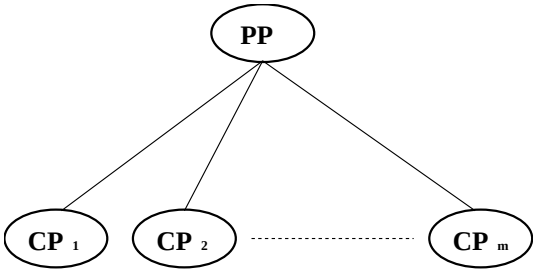

図 1: 星状型マルチプロセッサシステム

2 分散処理環境

近年パーソナルコンピュータの性能は非常に向上しており，情報収集や文書作成のためのクライアント計算機としての利用だけに留まらなくなってきている．従来は大型計算機を共用していたような科学技術計算においても Linux に代表される PC-UNIX で構成された計算機が研究室のような環境でも利用可能であり，そのような環境と違和感なく利用できる計算機環境が望まれている．そのため，インターネットに代表されるネットワーク関連技術を利用しこれらのクライアント計算機が相互に接続された計算機環境を構築することが進められている．このような計算機環境を分散処理環境と呼び，メタコンピューティング環境もしくはグリッドコンピューティング環境などとも呼ばれる．

分散処理環境では，ネットワーク上で既存のワークステーションや PC-UNIX 機などの計算機が複数接続されている環境を対象として分散処理を行うプログラムを作成する．このような処理環境を一つの仮想的な並列計算機とみなして並列処理プログラムの開発・実行を支援するためのツールとして PVM(Parallel Virtual Machine) や MPI と呼ばれるソフトウェア・ツールも提供されている．これらのツールを利用すると複数の異なる UNIX ワークステーションが接続された環境を仮想的に大きな一つの並列計算機と見なし，計算機間の通信処理などを容易に記述することが出来る．PVM などのツールでは分散処理環境を構成する関数，各計算機でプログラムを起動する関数，分散したプログラム間でメッセージを交換する関数など分散処理環境の利用に必要なさまざまな関数群を用意している．これらのツールでは，通常の UNIX ワークステーションだけではなく，スーパーコンピューターや超並列計算機をも仮想並列計算機の中に組み込んで使用することができる．

本報告では，並列処理システムとして図 1に示すような星状型マルチプロセッサシステムを，PVM ライブラリを利用し分散処理環境上に構築する．PP は親プロセッサを示し，$\mathrm{CP}_i(i = 1, 2, \ldots, m)$ は子プロセッサを示す．共有メモリは用意せず，各プロセッサは固有のメモリだけを利用する． また，プロセッサ間のデータの伝送は PVM ライブラリに用意されている関数を呼び出しメッセージとしてネットワークを通じて伝送する．

3 クラスタ構築例

分散処理環境を構成する計算機の能力については特に均一である必要はないが，台数の増加による性能向上を確認するためには機種が均一である方が比較を行いやすい．そのため今回構築した環境は以下のような計算機 21 台で構成している．

- CPU

  Intel Celeron Processor 433MHz

- MEMORY

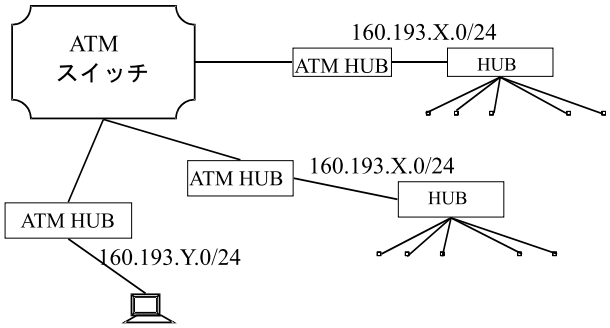

図 2: 構成

128MB SDRAM(66MHz)

- Network

  Intel EtherExpress Pro 10/100B

- マザーボード

  440BX 使用 MicroATX 規格

- VGA

  ATI model 4742 graphics accelerator

- HD

  ST36421A(6GB),WDC WD64AA(6GB),Maxtor 2B20H(20GB)

- OS

  FreeBSD 3.3-RELEASE

これらの計算機を写真図 7,8,9,10, 11,12のように 3 つのクラスタとして構成し，図 2のようなネットワーク構成で接続した．図 7,8のクラスタの方が時期としては始めに構築されているが，裏面の配線を見比べるとわかるように適切な長さのケーブルを用いてクラスタを構成しないと機器間の接続状況が非常に把握しづらくなり以後のメンテナンスが面倒になる．ネットワーク構成を示した図 2 の中で通常の 10MB Ethernet ハブとしてかかれている 2 カ所が写真図 7,9と 11のクラスタである．ネットワークとの接続は Ethetnet-ATM エッジハブ，ATM 交換機を通して同じネットワーク (160.193.XX.0/24) に属するように VLAN 機能を使って相互に接続されている．ネットワーク構成を表す図 2でも示したように本報告の計算機実験では 160.193.YY.0/24 と言う今回のクラスタとは別の計算機も分散処理環境を構築する計算機として参加している．これは各研究室の計算機も分散処理環境に参加できることを示しており，学内のように透過的なインターネット接続環境が用意されていれば，１台に限らず複数台の計算機を分散処理環境に参加させることが可能である．

上で述べたように，図 7のクラスタ (第一期) と図 9 のクラスタ (第二期)，第三期図 11のクラスタは機材の調達時期等が異なっている．けれども，均一な環境を実現するために可能であれば全く同じ計算機を導入したい．ここでパーソナルコンピュータの性能向上が著しく，発売される機種の変更が頻繁に行われることが問題になる．上で列挙した中でも特にハードディスクについては同じ部品を入手できず，各クラスタ間で別構成になった．ただ，ハードディスクは実メモリで不足するような場合のアクセスになると思われるので CPU の変更よりも影響は少ないと考えられる．CPU については半年毎に世代が入れ替わり以前の製品

については入手困難になると思われたのではじめのクラスタを購入する際に CPU のみを別途購入してそちらを次のクラスタを用意する際に利用するようにした．第三期の購入時期になると CPU だけでなく入出力などで重要な役割を果たすチップセットの入手が困難となり同じチップセットではあるが，当初利用していたマザーボードとは異なる製品を利用せざるを得なくなった．

4　クラスタの利用

構築したクラスタの利用例として，巡回セールスマン問題に対する並列分枝限定法を実行するプログラムを PVM を利用して構築しいくつかの標準的な例題を解いた結果を報告する．

4.1　巡回セールスマン問題

本報告では，組合せ最適化問題の中でも，よく取り扱われている巡回セールスマン問題を対象として計算機実験を行う．巡回セールスマン問題としては，各節点間の枝の重みの扱いによって非対称巡回セールスマン問題と対称巡回セールスマン問題とに分けられるが，ここではより困難な対称巡回セールスマン問題を取り上げる．

対称巡回セールスマン問題とは無向グラフの最小コスト巡回路を求める問題である． n 節点 $1, 2, \ldots, n$ をもつグラフにおいて，各枝 (i, j) の重みを c_{ij} とする．ただし，$c_{ij} = c_{ji}$ である．各枝 (i, j) に変数 x_{ij} を付す（ただし一般性を失うことなく $i < j$ とする）と，以下のような式で示される．

$$\text{目的関数} \quad \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} c_{ij} x_{ij} \rightarrow \text{最小化}$$

$$\text{制約条件} \quad \sum_{j=1}^{i-1} x_{ji} + \sum_{j=i+1}^{n} x_{ij} = 2\,;\, i = 1, 2, \ldots, n$$

$$\begin{aligned} x_{ij} \quad &= \quad 0\, or\, 1; \\ &\qquad i = 1, 2, \ldots, n-1\,; \\ &\qquad j = 2, 3, \ldots, n\,;\, i < j \end{aligned}$$

$$\begin{aligned} x_{ij} = 1 \qquad &\text{となる枝の集合は} \\ &\text{長さ } n \text{ の巡回路を作る．} \end{aligned}$$

巡回セールスマン問題の例としては， TSPLIB と呼ばれる一般に公開されている問題例の中からいくつかを選んで実行する．TSPLIB には，数都市程度の問題から数千都市程度の問題まで幅広い問題例が含まれているが，本研究では 48 都市 (GR48)，51 都市 (EIL51)，70 都市 (ST70)，99 都市 (RAT99) の 4 題を取り上げ計算機実験を行った．

4.2　分枝限定法

次に本報告で利用した組合せ最適化解法である分枝限定法について述べる．分枝限定法は，ほとんどすべての組合せ最適化問題に適用可能な計算手法である．分枝限定法では，直接解きがたい問題を一部の変数の値を固定することでより解きやすい部分問題に分解し，それらのすべてを解くことにより間接的にもとの問題を解く．この部分問題に分解する操作のことを分枝操作とよぶ．この操作は生成された部分問題に

対しても順次適用され，もとの問題はより規模の小さい，すなわち簡単な問題に分解される．こうして簡単になった部分問題をすべて解くと，その結果としてもとの問題が解かれることになる．

可能な分解操作をすべて行うだけでは単純な列挙法と同じであり，大規模な問題に対して適用することはできない．これを避けるため不必要な部分問題の生成を可能な限り抑える限定操作とよばれる操作が必要になる．分枝操作も限定操作もなされていない問題は活性部分問題とよばれ，次に処理すべき活性部分問題を探索木の中から選び出すことを探索とよぶ．探索の方法としては，分枝木上の位置に着目した探索（深さ優先探索，幅優先探索），部分問題の下解値に着目した探索，それらを組合せた探索などが考えられる．探索の過程における分枝変数の固定状況を辿ると木構造のデータが得られ分枝木とよぶ．

4.3 分枝変数の選択

前節で述べたように巡回セールスマン問題は，グラフの最小コスト巡回路を求める問題として定義され，前節の定義からある枝を表す変数 x_{ij} を “0” 若しくは “1” にすることで部分問題に分割することができる．枝を選択する方法としては，最適解に対する下界値をより改善する枝を選ぶ方法などが考えられているが，本研究では，まず距離の短い枝は巡回路に含まれやすいと思われるので枝をコスト順に選択する方法を利用する．また，巡回セールスマン問題は，すべての都市を訪問する順序を並び換えその中からコスト最小の順列を求める問題とも考えられる．この場合，今までの訪問都市の中に含まれていない都市を次に選ぶことで部分問題を作ることになる．この次に訪れる都市への枝を選択する方法を利用する．前者の場合と比較して，枝刈りの影響を考える必要があるが，分枝の初期段階でより多くの部分問題が作られることが予想されるので，前者とは分枝木の形が異なったものになると思われる．本研究では，これら二つの分枝方法を取り上げ，分枝方法の違いが並列分枝限定法に与える影響についても述べる．

4.4 下界値と上界値

分枝限定法では，各部分問題の最適値に対する下界値と上界値の値が厳密に求められると，限定操作が効率良く行われ，分枝操作を必要とする活性部分問題の数が少なくなり，最適解を求めるために必要な時間を短くできる．このため，問題の性質を利用した精度のよい下界値・上界値を求めることが望まれる．

対称巡回セールスマン問題の下界値を求める方法として，最小 1-tree の性質を利用した方法を用いる [6, 8]．1-tree とは，節点の数と同じ本数の枝からなり，無向グラフとして見た場合に連結しており，ある指定された一つの節点の次数が 2 であるようなグラフのことである．最小 1-tree は，次数 2 の節点 v_a を除く他の節点からなる最小木（minimum spanning tree）に，v_a に交わる枝のうちで重みの軽いものから順に選んだ 2 本の枝を加えることにより求められる．

対称巡回セールスマン問題では，各部分問題に対する上界値を求める良いアルゴリズムは提案されていない．一般に，多くの組合せ最適化問題では最適解を得る必要がないならば，近似解法を用いて大規模な問題においても実用的な計算時間で近似解を求めることができる．この近似解法を用いて得られる初期問題の局所最適解を繰り返し複数個求め，それらの中で最も良い局所解を上界値として採用する方法を使用する．本研究では，近似解法として，近傍探索法の一つである B.W.Kernighan と S.Lin の提案した解法 [9] を取り上げる．この解法は α-optimal heuristic method を改良したものであり，この解法 “KL-method” を用いて求めた近似解を上界値として用いる．

4.5 並列分枝限定法

本報告では，並列処理システムとして星状型マルチプロセッサシステムを PVM ライブラリを利用し，分散処理環境上に構築する．このようなマルチプロセッサシステム上で並列分枝限定法を実現する方法として次のような方法を採用している．

表 1: Virtual machine の構成ホスト

	A	B
機種	PC(クラスタ)	PC
OS	FreeBSD3.3	FreeBSD3.3
メモリ	128MB	256MB
CPU	Celeron 433MHz	AMD 200MHz
台数	最大 19 台	1 台

親プロセッサは与えられた問題のデータと解の管理，および子プロセッサへの問題の割当を行う．子プロセッサは親プロセッサから与えられた活性部分問題を初期問題とみなし，探索と分枝操作，限定操作を繰り返し行う．この方法では，親プロセッサに関しては記憶領域および通信時間が節約でき負荷は軽くなる．しかし，子プロセッサに割り当てる部分問題は，すべてが同じ時間計算量（手間）で終了するわけではなく，その違いをあらかじめ知ることができないため，各子プロセッサで探索終了時間に差が生じる．活性部分問題がなくなった空き状態の子プロセッサをそのままにしておくことは，計算機の利用効率の点から考えても明らかに良くないので，このような子プロセッサに対して，親プロセッサが再び部分問題を再割当する必要がある．

子プロセッサから活性部分問題がなくなったという報告が来た場合，親プロセッサに用意するリストに活性部分問題があれば，ただちにその中から問題を一つ選び再割当を行う．割り当てるべき活性部分問題がリストにない場合，その他の子プロセッサから，部分問題を一問づつ集め親プロセッサのリストに登録し，その中から問題を一つ選び再割当を行う．

5 実験結果

PVM の仮想並列計算機を構成する UNIX ワークステーションとして，表 1に示すような計算機群を用いる．A の計算機の中から 1 台を選び，親プロセッサ用のプログラムを実行する．親プロセッサ用を除く他の A の計算機上では子プロセッサ用のプログラムを実行する．上界値を得るための近似解法を実行するプログラムは別の計算機 B 用として用意し，他の親プロセッサ・子プロセッサとは独立に実行する．このプロセッサで求めた近似解は，限定操作の効率をよくするための暫定解として，他のプロセッサに放送される．

子プロセッサが次に分解するべき活性部分問題を選ぶ方法には，**4.2**で述べたように複数の方法が考えられ，最良下界値優先探索，深さ優先探索，幅優先探索及びこれらの複合型などが提案されている．本研究では，子プロセッサでの探索方法は，一般的に良い暫定解が早く見つかり，枝刈りの効率が上がるとされている最良下界値優先探索を各実験において共通の探索方法として採用する．

4.5で述べたように，子プロセッサは，親プロセッサからの要求に応じて他の子プロセッサに再割当てされる部分問題を選ぶために子プロセッサ自身がもっている活性部分問題のリストの中を探索し，親プロセッサに部分問題を送る．ここで親プロセッサに送る部分問題を探索する方法も負荷の分散を考えるときには影響がある．本研究では，この場合の探索方法については，幅優先探索を共通の探索方法とする．幅優先探索を行う場合，親プロセッサに送られる部分問題は，他の探索法の場合と比べて，分枝木上でより根に近い位置にある問題が送られるようになる．根に近い部分問題の方が，変数の固定が進んでいないため，その部分問題の評価に要する時間が長くなることを期待できる．これにより，再割当ての回数が不必要に多くならないようにする．

実験の結果を図 3,4,5,6 に示す．それぞれ実行時間と利用した計算機の台数に応じた加速度をグラフにしている．図中 “△” は 4.2で述べたように活性部分問題を分解する際にコスト順で選んだ枝を選択する場合 (中塗りが経過時間，中抜きが加速度) を表し，次に訪問する都市を選ぶ場合が “◇” で表される．

並列計算を行う場合，理想的な状態としては n 台の計算機を利用する場合 n 倍の線形な加速度を得ることが望まれる．その点から結果のグラフを見ると実行時間が長いすなわち難しいと思われる問題例 (ST70) の方が簡単な問題例 (EIL51) よりも線形に近い加速度が得られている．さらに，分枝方法を変更した影響 (コスト順 “△”，都市順 “◇”) の違いを見ると必ずしもどちらかの分枝方法がすべての問題に対して共通して有効であるわけではなく，特に台数が増えてきた場合には分枝方法の違いにより加速度が飽和する程度に違いが見られることがわかる．線形に近い加速度を得るためには問題に応じて適切な条件で実行する必要があることがわかる．

6 むすび

本報告では，当センター内に構築した PC-UNIX 機クラスタによる分散処理環境の構築例について述べた．さらに，応用として並列分枝限定法を実装しその効果を確認している．今後本報告で扱ったような分散処理環境はますます重要になると思われるので引き続きクラスタの増設等に取り組んでいきたいと考えている．ここでは性能比較のために単一構成の計算機によるクラスタを考えたが実際に大規模なアプリケーションを実行する場合には異機種をも含めた環境でのクラスタ構成も有効であると思われるので異機種も含めたクラスタ構成やそのような環境での性能テストなども今後の課題である．

参考文献

[1] 大西克実，榎原博之，中野秀男：“並列分枝限定法における分枝変数の選択に関する考察”, 信学論,**J84D-1-9**,p1318-1326，2001.

[2] 今井正治，吉田雄二，福村晃夫： “分枝限定アルゴリズムの並列化とその評価”，信学論，**J62-D(6)**, pp.403-410, 1979.

[3] 池上，青柳，飯塚： “分散記憶型マシンにおける並列整数計画法”，信学論，**J75-D-I-9**, pp.801-808, 1992.

[4] 品野勇治，檜垣正浩，平林隆一： “並列分枝限定法における解の探索規則”，計測自動制御学会論文，**32-9**, pp.1379-1387(1996)

[5] 茨木俊英： “組合せ最適化　分枝限定法を中心として”，講座・数理計画法第８巻　産業図書，1983.

[6] 今野浩，鈴木久敏： “整数計画法と組合せ最適化”，ＯＲライブラリー第７巻　日科技連出版社，1982.

[7] E.L.Lawler and J.K.Lenstra and A.H.G.Rinnooy Kan etc.: “The travelling salesman problem”,John Wiley and Sons, 1985.

[8] M.Held and R.M.Karp: “The Traveling Salesman Problem and Minimum Spanning Trees: Part II”, Math.Progm, **1**, pp. 6-25, 1971.

[9] S.Lin and B.W. Kernighan: “An efficient heuristic algorithm for the traveling salesman problem”, Operations Research, **21**, pp. 498–516, 1973.

[10] V.S.Sunderam: “PVM: A Framework for Parallel Distributed Computing”, Journal of Concurrency: Practice and Experiecnce, **2**(4), pp. 315-339, December 1990.

[11] Catherine Roucariro: “Parallel branch and bound algorithms - an overview”, In M.Cosnard et al., Parallel and Distributed Alogorithm, pp. 153–16, North-Holland, 1989.

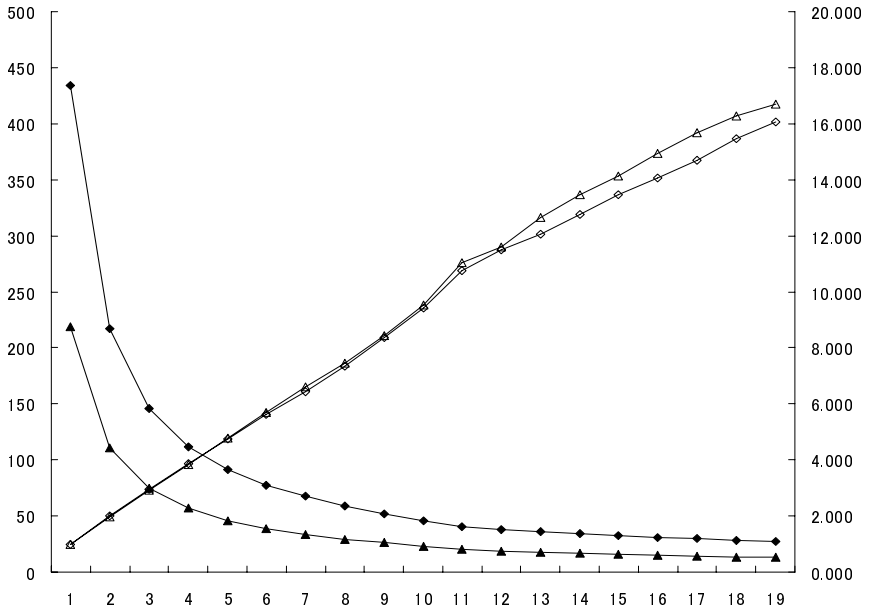

図 3: 実験結果 (GR48)[台数/時間 (秒)，加速度]

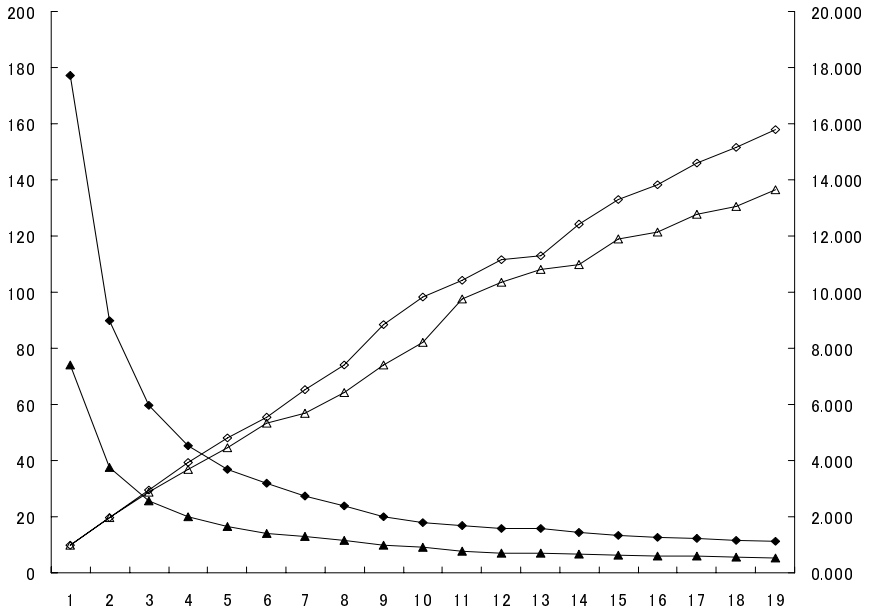

図 4: 実験結果 (EIL51)[台数/時間 (秒)，加速度]

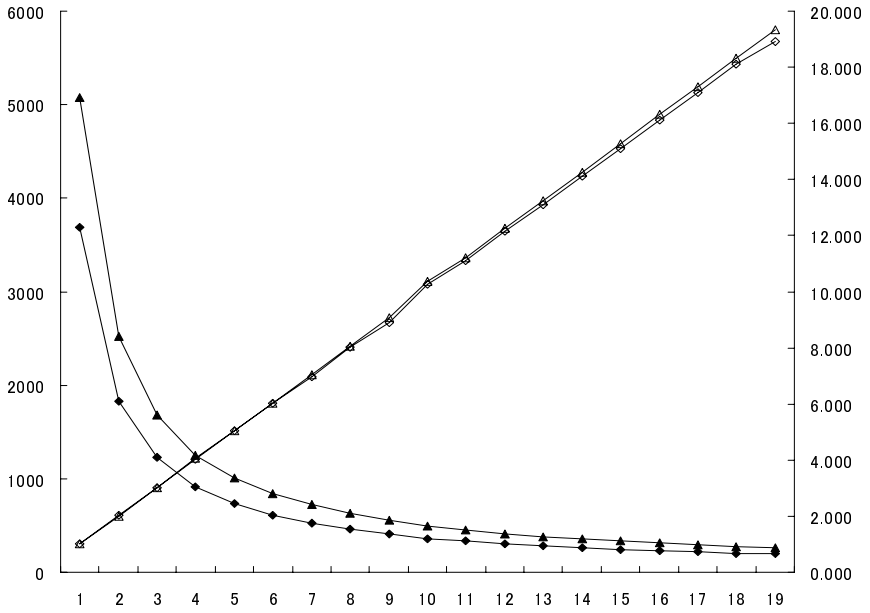

図 5: 実験結果 (ST70)[台数/時間 (秒)，加速度]

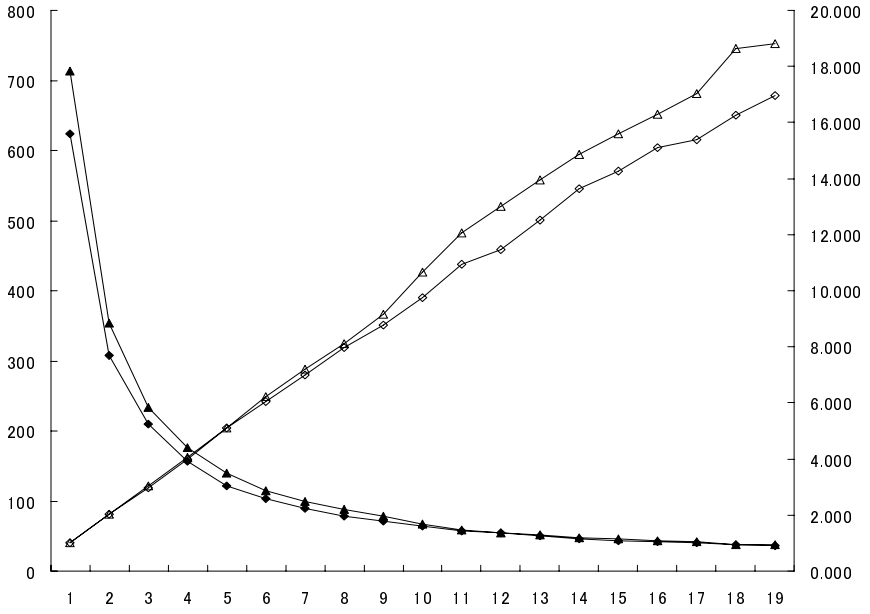

図 6: 実験結果 (RAT99)[台数/時間 (秒)，加速度]

図 7: クラスタ 1 正面

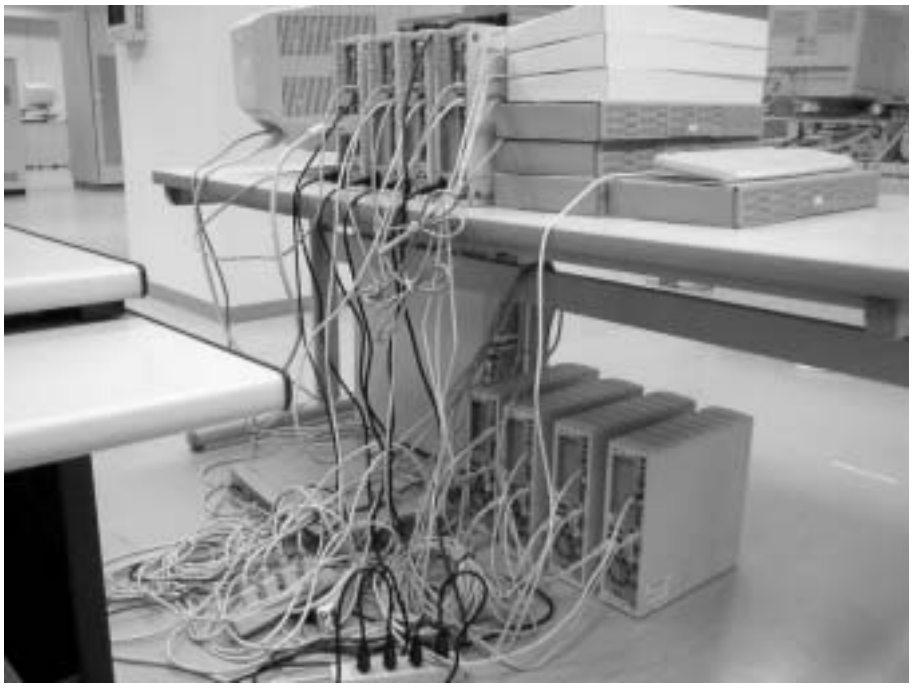

図 8: クラスタ 1 背面

図 9: クラスタ 2 正面

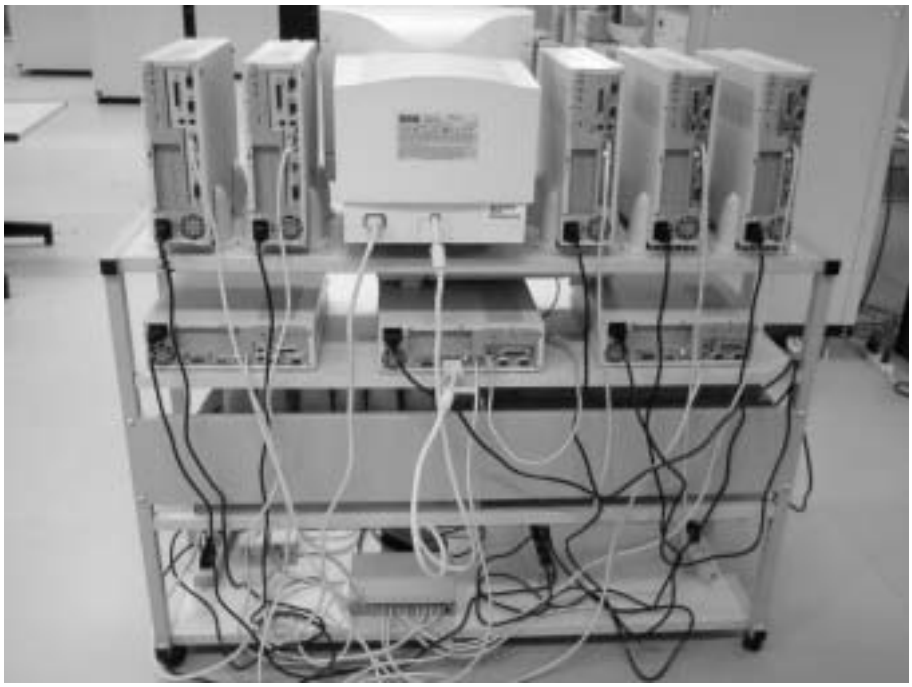

図 10: クラスタ 2 背面

図 11: クラスタ 3 正面

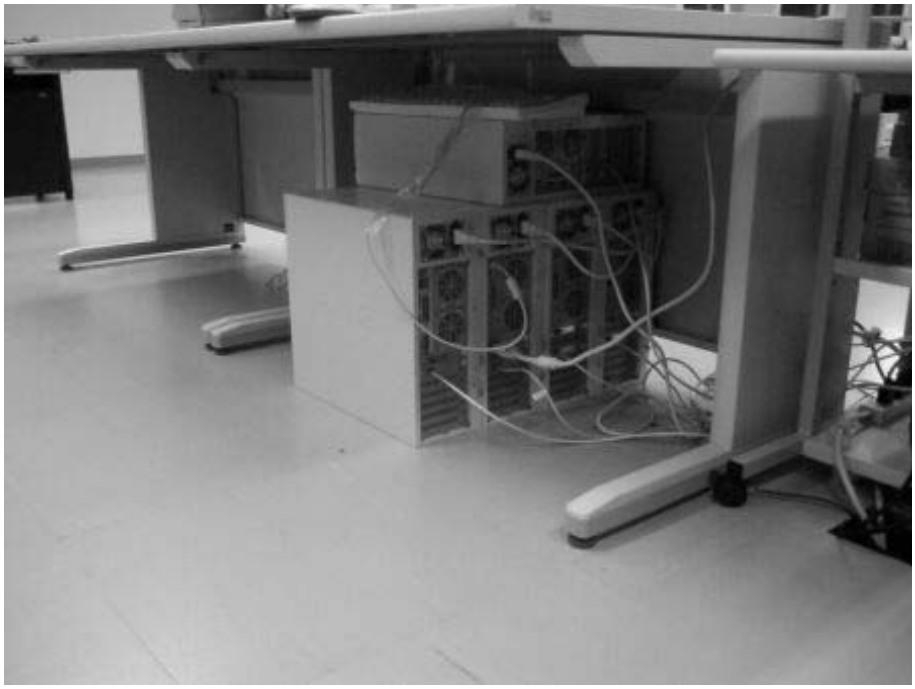

図 12: クラスタ 3 背面