並列制約充足アルゴリズムとそのKL1による実現

杉本 勉
NTTデータ通信(株)

生駒憲治
(財)新世代コンピュータ技術開発機構

1 はじめに

最近知識処理の分野では、制約に基づいた問題解決や制約プログラミングといった、知識を制約として扱う方法が注目されている。ここでいう制約とは広い意味で知識そのものを指す。例えば物理法則や方程式も制約であり、ものとものの間の「関係」もまた制約である。

本研究は制約に基づく問題解決の中でいわゆる「組合せ問題」である制約充足問題について、その並列アルゴリズムと並列言語KL1で実現する方法を提案するものである。充足の方法として「併合法(merge method)」という方式を用いている。この方法は逐次的な処理よりも分割統治法に似た並列処理のほうがより自然である。すなわち局所性が強く、併合する順序というものは関係ないため部分的にできるところから進めることができるのである。

なお、本研究はICOT Symmetry上のPDSSおよびマルチPSI上で実現されており、プログラムサイズは約1,300行である。

2 併合法による制約充足

制約とは変数の組とその変数間で考えられる関係をすべて列挙したもののペアである（以下では後者を制約というときもある）。制約に基づいて問題解決を行う場合、制約ネットワークを考えることが多い。これにはいくつかの表現方法があるが、本研究では制約をネットワークのノード、制約間の変数の共有関係をネットワークのアークと考える。そこで、「併合法」とは制約ネットワークのノード間（制約間）の併合操作によって充足させる方法である、ということができる(図1)。

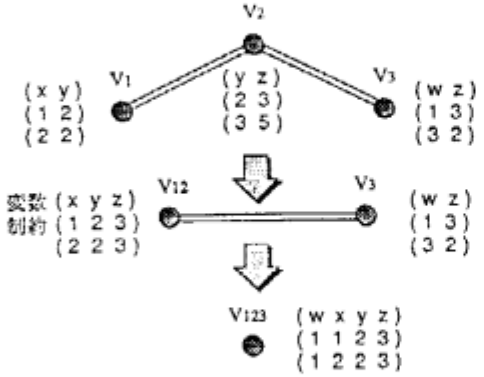


図 1: 併合法の例

逐次処理の場合、ノード間の併合順序（併合系列）によって計算量に大きな差が生じる。したがって、最適（または準最適）併合系列を求めることが考えられるが、本研究ではそれを求めることはしない。これは併合は逐次的にしなければならないのものではないからである。積極的に部分的な併合を行えば、それは自然と並列処理になる。すなわち、併合処理全体を分割統治法としてとらえることができる。

3 並列制約充足アルゴリズム

本稿で提案する並列制約充足アルゴリズムでは制約ネットワークよりも変数の共有関係に着目した制約フロー(図2)が重要な役割を果たす。ここで制約はインクリメンタルに入力されるものとする。

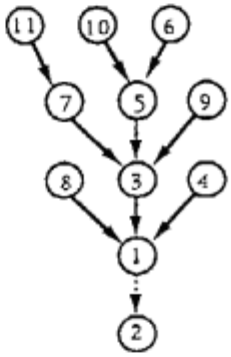


図 2: 図5のクロスワード問題の制約フロー

制約フローの制約ノードVは7つの要素の組から成る。それらは、制約の識別子ID、状態フラグFlg、変数の組T、変数間の関係（制約）W、中間解A、入力チャネルIn、出力チャネルOut、である。図3は制約フローの生成アルゴリズムである。

制約充足は制約フローの生成と並行して行われるが、ここでは別々に説明する(図4)。制約フローに入力される制約はほかの情報と一緒に一つのメッセージとして流れていく。このメッセージは4つまたは5つの要素の組である。4つの場合は、ID,T,Wと制約フロー内での位置を決めるための評価得点S（初期値は0）を、5つの場合はメッセージ通信用変数Msgを加えたものである。ここで、この4つまたは5つの組のメッセージをaddメッセージと呼び、Msgをreplyメッセージと呼ぶことにする。

[1]dummy
[2]dummy

1. 制約ノードV上にaddメッセージがある。
   1.1 制約ノード内の変数の組T1とaddメッセージ内の変数の組T2の関係を調べる。
      1.1.1 共有関係はない：何もせず受け取った制約を別の制約に転送する。転送する制約がないならば2.へ。
      1.1.2 共有関係がある：共有部分の長さをS1とし、addメッセージ内のS2と比較する。
         1.1.2.1 S1=<S2：何もせずaddメッセージを別のノードへ転送する。転送する制約がないならば2.へ。
         1.1.2.2 S1>S2：addメッセージの制約を制約ノードとして配置し、制約ノードVとフロー関係を結ぶ。さらに、addメッセージの得点をS1にし、replyメッセージを加えて別のノードに送る。転送する制約がないならば2.へ。
2. addメッセージの行き先はもうない。
   2.1 addメッセージ=(ID,T,W,S)：その位置に配置する。
   2.2 addメッセージ=(ID,T,W,S,Msg)：replyメッセージ(Msg)にdecideをバインドする。

図3: 制約フロー生成アルゴリズム

1. 制約ノードのフラグが
   1.1 Flg=remove：直ちにそのノードは削除される。
   1.2 Flg=decide：そのノードのTとWを結合先の制約ノードに送る（このメッセージをmergeメッセージと呼ぶ）。2.へ。フラグFlgをmergedにする。
2. mergeメッセージの中の制約と中間解Aを併合させる。（このとき、制約ノードの制約とは併合しない。）
3. 制約がすべてネットワークに入るとconquerメッセージが流される。conquerメッセージを受け取ると
   3.1 Flg=merged：ノードの中間解A1とメッセージの中間解A2を併合し、新しい中間解A3を生成して、それをconquerメッセージにのせ別のノードに転送する。転送する制約がないならば4.へ。
   3.2 Flg=state：ノードの中間解A1とメッセージの中間解A2それに自分の制約を併合して、新しい中間解A3を生成し、それをconquerメッセージにのせ別のノードに転送する。転送する制約がないならば4.へ。
4. すべての制約ノードを回ったconquerメッセージが持っている内容が答である。

図4: 制約充足アルゴリズム

4 KL1による実現

並列言語KL1では制約フローの制約ノードはプロセスとして実現されている。さらに制約フロー自身はリング型ネットワークとして実現されている。したがって共有関係によって得られた併合先は制約ノードで記憶し、メッセージ通信によって併合先に送られる。

これは、目的の制約ノードに届くには直接届けるよりステップ数はかかるがプログラム的にはリング状にして1つのノード（プロセス）には入出力チャネルがそれぞれ1つづつしか存在しないようにした方が分かり易いからである。

5 応用

応用としてクロスワード問題とNクイーン問題を解いた。ここでは図5のクロスワード問題の定式化の方法を述べる。

クロスワード問題は、はめ込むべき単語はあらかじめ列挙されているものとする。マス目一つ一つを変数と考え、はめ込む場所に対して考えられる単語をすべて列挙することで一つの制約とする。すなわち3文字をあてはめる場所があればそれに対する制約とは3文字からなる単語すべてである（図5）。

**	1	2	3	**
4	5	6	7	8
9	10	11	**	12
13	14	15	16	17
**	18	19	20	21

**	b	o	l	**
l	l	i	e	e
e	o	n	**	e
e	n	t	e	r
**	d	e	m	i

No.1
Variables(5) = [4,5,6,7,8]
Constraints(4) = [[b,l,o,n,d],[e,n,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

No.2
Variables(5) = [13,14,15,16,17]
Constraints(4) = [[b,l,o,n,d],[e,n,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

No.3
Variables(5) = [1,5,10,14,18]
Constraints(4) = [[b,l,o,n,d],[e,n,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

⋮

図5: クロスワード問題

6 まとめ

併合法に基づく並列制約充足アルゴリズムとそのKL1による方法について述べた。

KL1は節の実行それ自身がネットワークノードに、共有変数がアークに対応するため、ネットワークの記述が自然であり優れていることを確かめることができた。しかし、現在のアルゴリズムでは、並列性を十分に生かしていないという欠点がある。今後、並列実行部分の解析機能を強化したい。

また、64PE マルチPSI上での実行に関する詳しいことはここでは述べないが、負荷分散を工夫することによって1PE時に比べ最大17倍の速度向上を得ることができた。

最後に、本研究を進めるにあたり貴重な意見を頂いた元ICOT研究員（現 東芝）永井氏、NTTデータ通信（株）久保田担当部長、佐塚氏に感謝致します。

参考文献

[1] Hentenryck,P.Van."Constraint Satisfaction in Logic Programming", The MIT Press,(1989)

[2] 西原,松尾,池田,"擬整合ラベリング問題における併合法の最適化", 人工知能学会誌, Vol.3, No.2, (1988)

[3] 市吉,瀧,"大規模並列マシン上の並列プログラムのマッピングの実験と解析",(1989)

```
% 4-Queen
No.1
Variables(2) = [1,2]
Constraints(6) =
     [[1,3],[1,4],[2,4],[3,1],[4,1],[4,2]]

No.2
Variables(2) = [1,3]
Constraints(8) =
     [[1,2],[1,4],[2,1],[2,3],[3,2],[3,4],[4,1],[4,3]]

No.3
Variables(2) = [1,4]
Constraints(10) =
     [[1,2],[1,3],[2,1],[2,3],[2,4],[3,1],[3,2],[3,4],[4,2],[4,3]]

No.4
Variables(2) = [2,3]
Constraints(6) =
     [[1,3],[1,4],[2,4],[3,1],[4,1],[4,2]]

No.5
Variables(2) = [2,4]
Constraints(8) =
     [[1,2],[1,4],[2,1],[2,3],[3,2],[3,4],[4,1],[4,3]]

No.6
Variables(2) = [3,4]
Constraints(6) =
     [[1,3],[1,4],[2,4],[3,1],[4,1],[4,2]]

End >>
```



```
% 5-Queen

No.1
Variables(2) = [1,2]
Constraints(12) =
     [[1,3],[1,4],[1,5],[2,4],[2,5],[3,1],[3,5],[4,1],[4,2],[5,1], ... ]

No.2
Variables(2) = [1,3]
Constraints(14) =
     [[1,2],[1,4],[1,5],[2,1],[2,3],[2,5],[3,2],[3,4],[4,1],[4,3], ... ]

No.3
Variables(2) = [1,4]
Constraints(16) =
     [[1,2],[1,3],[1,5],[2,1],[2,3],[2,4],[3,1],[3,2],[3,4],[3,5], ... ]

No.4
Variables(2) = [1,5]
Constraints(18) =
     [[1,2],[1,3],[1,4],[2,1],[2,3],[2,4],[2,5],[3,1],[3,2],[3,4], ... ]

No.5
Variables(2) = [2,3]
Constraints(12) =
     [[1,3],[1,4],[1,5],[2,4],[2,5],[3,1],[3,5],[4,1],[4,2],[5,1], ... ]

No.6
Variables(2) = [2,4]
Constraints(14) =
     [[1,2],[1,4],[1,5],[2,1],[2,3],[2,5],[3,2],[3,4],[4,1],[4,3], ... ]

No.7
Variables(2) = [2,5]
Constraints(15) =
     [[1,2],[1,3],[1,5],[2,1],[2,3],[2,4],[3,1],[3,2],[3,4],[3,5], ... ]

No.8
Variables(2) = [3,4]
Constraints(12) =
     [[1,3],[1,4],[1,5],[2,4],[2,5],[3,1],[3,5],[4,1],[4,2],[5,1], ... ]

No.9
Variables(2) = [3,5]
Constraints(14) =
     [[1,2],[1,4],[1,5],[2,1],[2,3],[2,5],[3,2],[3,4],[4,1],[4,3], ... ]

No.10
Variables(2) = [4,5]
Constraints(12) =
     [[1,3],[1,4],[1,5],[2,4],[2,5],[3,1],[3,5],[4,1],[4,2],[5,1], ... ]

End >>
```

```
% 6-Queen

No.1
Variables(2) = [1,2]
Constraints(20) =
     [[1,3],[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,1],[3,5],[3,6], ... ]

No.2
Variables(2) = [1,3]
Constraints(22) =
     [[1,2],[1,4],[1,5],[1,6],[2,1],[2,3],[2,5],[2,6],[3,2],[3,4], ... ]

No.3
Variables(2) = [1,4]
Constraints(24) =
     [[1,2],[1,3],[1,5],[1,6],[2,1],[2,3],[2,4],[2,6],[3,1],[3,2], ... ]

No.4
Variables(2) = [1,5]
Constraints(26) =
     [[1,2],[1,3],[1,4],[1,6],[2,1],[2,3],[2,4],[2,5],[3,1],[3,2], ... ]

No.5
Variables(2) = [1,6]
Constraints(28) =
     [[1,2],[1,3],[1,4],[1,5],[2,1],[2,3],[2,4],[2,5],[2,6],[3,1], ... ]

No.6
Variables(2) = [2,3]
Constraints(20) =
     [[1,3],[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,1],[3,5],[3,6], ... ]

No.7
Variables(2) = [2,4]
Constraints(22) =
     [[1,2],[1,4],[1,5],[1,6],[2,1],[2,3],[2,5],[2,6],[3,2],[3,4], ... ]

No.8
Variables(2) = [2,5]
Constraints(24) =
     [[1,2],[1,3],[1,5],[1,6],[2,1],[2,3],[2,4],[2,6],[3,1],[3,2], ... ]

No.9
Variables(2) = [2,6]
Constraints(26) =
     [[1,2],[1,3],[1,4],[1,6],[2,1],[2,3],[2,4],[2,5],[3,1],[3,2], ... ]

No.10
Variables(2) = [3,4]
Constraints(20) =
     [[1,3],[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,1],[3,5],[3,6], ... ]

No.11
Variables(2) = [3,5]
Constraints(22) =
     [[1,2],[1,4],[1,5],[1,6],[2,1],[2,3],[2,5],[2,6],[3,2],[3,4], ... ]

No.12
Variables(2) = [3,6]
Constraints(24) =
     [[1,2],[1,3],[1,5],[1,6],[2,1],[2,3],[2,4],[2,6],[3,1],[3,2], ... ]

No.13
Variables(2) = [4,5]
Constraints(20) =
     [[1,3],[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,1],[3,5],[3,6], ... ]

No.14
Variables(2) = [4,6]
Constraints(22) =
     [[1,2],[1,4],[1,5],[1,6],[2,1],[2,3],[2,5],[2,6],[3,2],[3,4], ... ]

No.15
Variables(2) = [5,6]
Constraints(20) =
     [[1,3],[1,4],[1,5],[1,6],[2,4],[2,5],[2,6],[3,1],[3,5],[3,6], ... ]

End >>
```

```
%% Cross-word Problem 1
%%     **  1  2  3 **     |    **  b  o  l **
%%      4  5  6  7  8     |     l  l  i  e  e
%%      9 10 11 ** 12     |     i  o  n **  e
%%     13 14 15 16 17     |     e  n  t  e  r
%%     ** 18 18 20 21     |    **  d  e  m  j

No.1
Variables(5) = [4,5,6,7,8]
Constraints(4) =
     [[b,l,o,n,d],[e,r,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

No.2
Variables(5) = [13,14,15,16,17]
Constraints(4) =
     [[b,l,o,n,d],[e,n,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

No.3
Variables(5) = [1,5,10,14,18]
Constraints(4) =
     [[b,l,o,n,d],[e,n,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

No.4
Variables(5) = [2,6,11,15,19]
Constraints(4) =
     [[b,l,o,n,d],[e,n,t,e,r],[l,l,i,e,e],[o,i,n,t,e]]

No.5
Variables(4) = [18,19,20,21]
Constraints(2) =
     [[d,e,m,i],[e,c,r,i]]

No.6
Variables(4) = [8,12,17,21]
Constraints(2) =
     [[d,e,m,i],[e,c,r,i]]

No.7
Variables(3) = [1,2,3]
Constraints(3) =
     [[b,o,l],[i,o,n],[l,i,e]]

No.8
Variables(3) = [4,9,13]
Constraints(3) =
     [[b,o,l],[i,o,n],[l,i,e]]

No.9
Variables(3) = [9,10,11]
Constraints(3) =
     [[b,o,l],[i,o,n],[l,i,e]]

No.10
Variables(2) = [16,20]
Constraints(2) =
     [[e,m],[l,e]]

No.11
Variables(2) = [3,7]
Constraints(2) =
     [[e,n],[l,e]]

End >>
```

```
%% Cross-word Problem 2
%%   1  2 ** **  3 ** **  4  |   s  e ** **  s ** **  i
%%  **  5  6  7  8  9 10 11  |  **  c  o  u  s  s  i  n
%%  12 13 ** 14 15 16 17 18  |   l  e ** s  e  i  n  c
%%  19 20 ** 21 ** 22 23 **  |   u  r ** e ** c  i ** 
%%  24 25 26 27 ** ** 28 **  |   e  v  e  r ** ** t ** 
%%  ** 29 ** ** 30 31 32 33  |  **  e ** ** o  u  i  r
%%  34 35 36 37 38 39 40 **  |   n  l  i  a  s  s  e ** 
%%  ** 41 ** ** 42 43 44 45  |  **  e ** **  t  e  r  a

No.1
Variables(8) = [2,5,13,20,25,29,35,41]
Constraints(1) =
     [[e,c,e,r,v,e,l,e]]

No.2
Variables(7) = [10,17,23,28,32,40,44]
Constraints(3) =
     [[c,o,u,s,s,i,n],[n,l,i,a,s,s,e],[i,n,i,t,i,e,r]]

No.3
Variables(7) = [5,6,7,8,9,10,11]
Constraints(3) =
     [[c,o,u,s,s,i,n],[n,l,i,a,s,s,e],[i,n,i,t,i,e,r]]

No.4
Variables(7) = [34,35,36,37,38,39,40]
Constraints(3) =
     [[c,o,u,s,s,i,n],[n,l,i,a,s,s,e],[i,n,i,t,i,e,r]]

No.5
Variables(5) = [14,15,16,17,18]
Constraints(1) =
     [[s,e,i,n,e]]

No.6
Variables(4) = [7,14,21,27]
Constraints(4) =
     [[e,v,e,r],[o,u,i,r],[t,e,r,a],[u,s,e,r]]

No.7
Variables(4) = [24,25,26,27]
Constraints(4) =
     [[e,v,e,r],[o,u,i,r],[t,e,r,a],[u,s,e,r]]

No.8
Variables(4) = [30,31,32,33]
Constraints(4) =
     [[e,v,e,r],[o,u,i,r],[t,e,r,a],[u,s,e,r]]

No.9
Variables(4) = [42,43,44,45]
Constraints(4) =
     [[e,v,e,r],[o,u,i,r],[t,e,r,a],[u,s,e,r]]

No.10
Variables(3) = [12,19,24]
Constraints(6) =
     [[l,u,e],[s,s,e],[o,s,t],[s,i,c],[u,s,e],[i,n,e]]

No.11
Variables(3) = [3,8,15]
Constraints(6) =
     [[l,u,e],[s,s,e],[o,s,t],[s,i,c],[u,s,e],[i,n,e]]

No.12
Variables(3) = [30,38,42]
Constraints(6) =
     [[l,u,e],[s,s,e],[o,s,t],[s,i,c],[u,s,e],[i,n,e]]

No.13
Variables(3) = [31,39,43]
Constraints(6) =
     [[l,u,e],[s,s,e],[o,s,t],[s,i,c],[u,s,e],[i,n,e]]

No.14
Variables(3) = [9,16,22]
Constraints(6) =
     [[l,u,e],[s,s,e],[o,s,t],[s,i,c],[u,s,e],[i,n,e]]

No.15
Variables(3) = [4,11,18]
Constraints(6) =
     [[l,u,e],[s,s,e],[o,s,t],[s,i,c],[u,s,e],[i,n,e]]

No.16
Variables(2) = [1,2]
Constraints(4) =
     [[l,e],[s,e],[u,r],[c,i]]

No.17
Variables(2) = [22,23]
Constraints(4) =
     [[l,e],[s,e],[u,r],[c,i]]

No.18
Variables(2) = [12,13]
Constraints(4) =
     [[l,e],[s,e],[u,r],[c,i]]

No.19
Variables(2) = [19,20]
Constraints(4) =
     [[l,e],[s,e],[u,r],[c,i]]

End >>
```

Number of Reduction
Execution Time (second)
Reduction/Second

Application = Cross-Word Problem

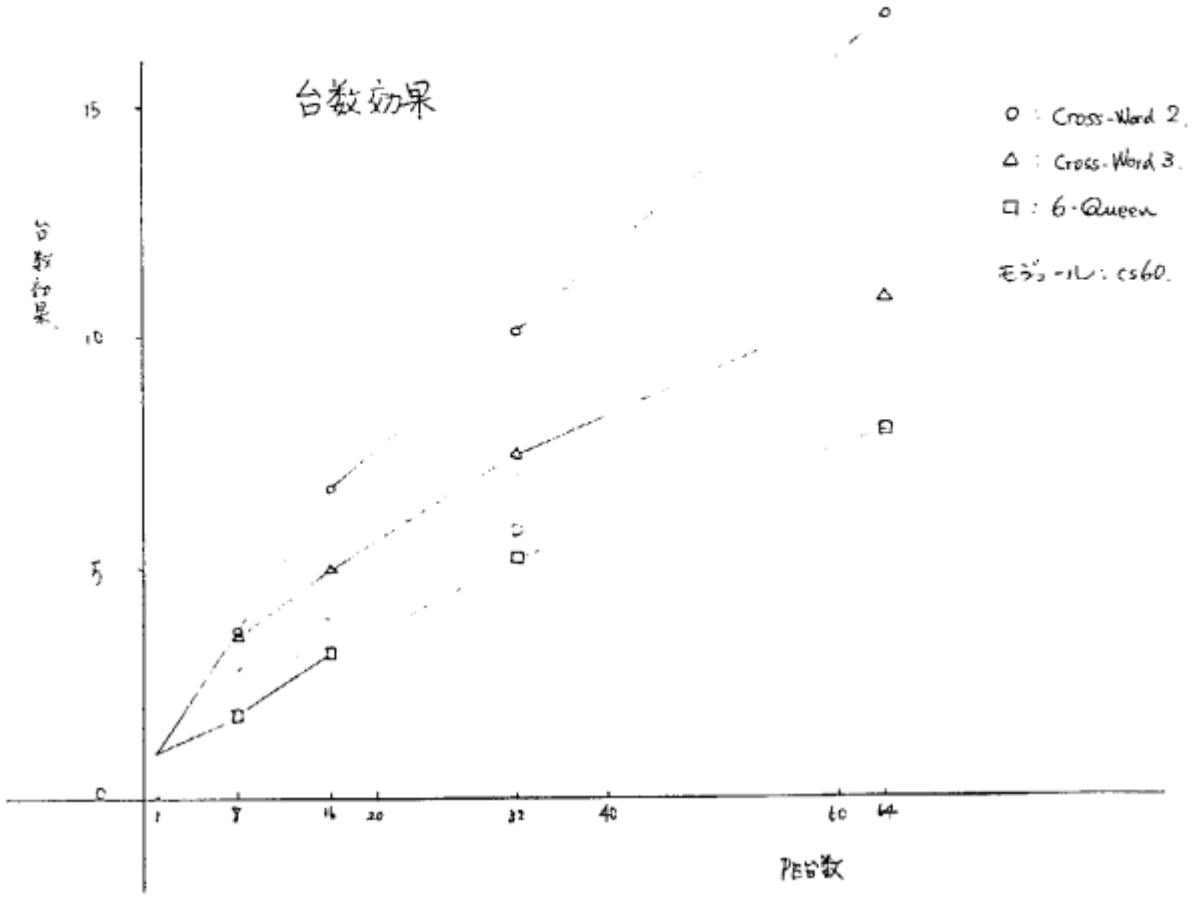
	8-PEs	16-PEs	32-PEs	64-PEs
No.1	1-PE = 41,654red. 1.517sec. 27,458red/sec			
cs40	42,209 1.609 26,233	42,200 2.017	42,208 2.172 19,433	42,200 2.119
cs50	43,963 1.528	43,961 1.534	43,950 1.670 26,317	43,961 1.493 29,444
cs60	43,838 1.261 34,764	43,815 1.009	43,813 0.959 45,686	43,901 1.189
cs70	56,566 1.802 31,390	54,496 1.327 41,067	54,463 1.337	54,467 1.513
cs80	65,068 2.517 25,851	64,926 1.793 36,210	64,893 1.968	64,898 2.303
No.2	1-PE = 59,893,506red. 3,138.344sec. 19,084red/sec			
cs50	59,911,581 2,841.053 21,087	59,907,615 2,766.404 21,655	59,906,328 2,836.369	59,906,297 2,029.417
cs60	59,907,823 867.788 69,035	59,907,217 475.911	59,907,193 313.106	59,907,135 184.715 324,321
No.3	1-PE = 8,074,026red. 441.010sec. 18,308red/sec			
cs40	8,074,804 476.063 16,962	8,074,889 479.681	8,074,883 484.026	8,074,802 485.623 16,628
cs50	8,205,473 429.574 19,101	8,205,490 374.315	8,205,479 364.474 22,513	8,205,333 422.566
cs60	8,180,524 124.333 65,795	8,188,091 88.794	8,180,504 59.150	8,181,898 40.799 200,542
cs70	8,765,030 148.906 58,863	8,762,059 78.964	8,760,715 55.997	8,759,399 55.301 158,395
cs80	9,344,877 208.235	9,338,791 107.859	9,335,797 86.862	9,334,398 84.738

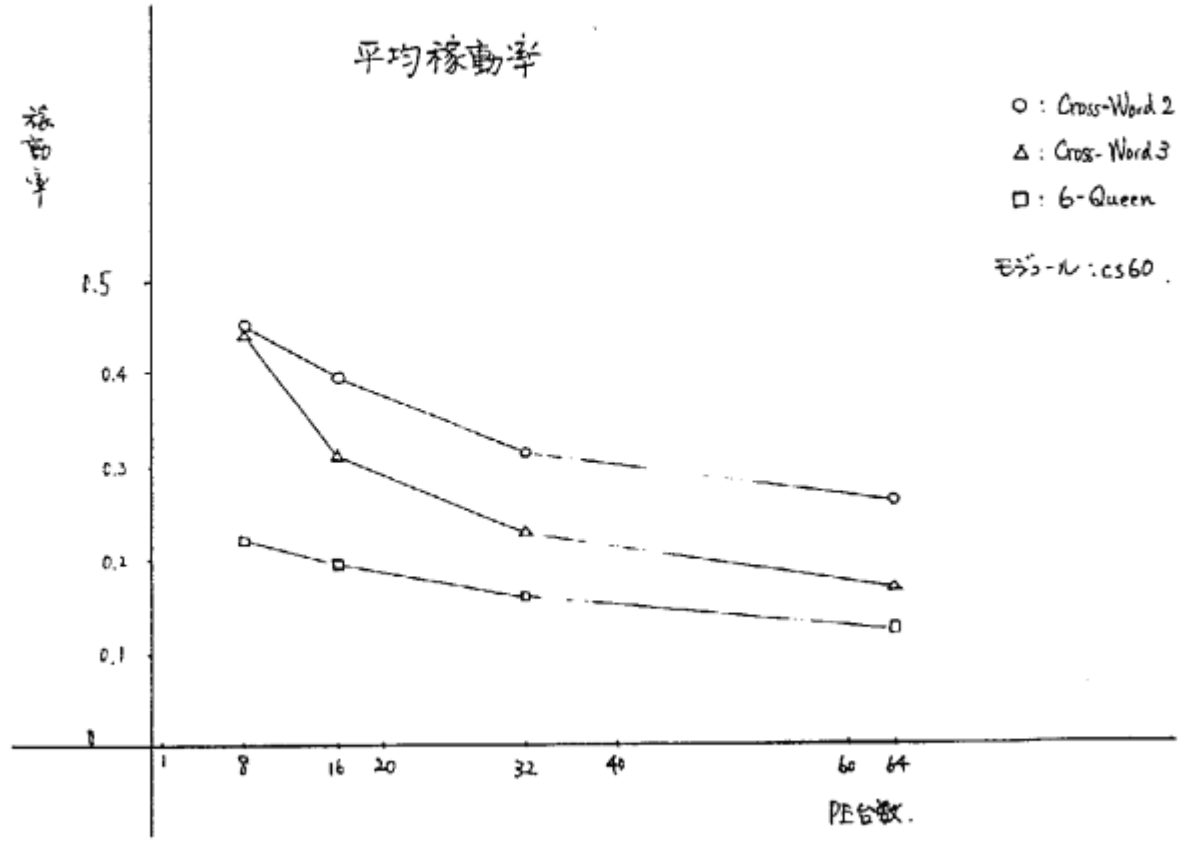
	44,877			110,156

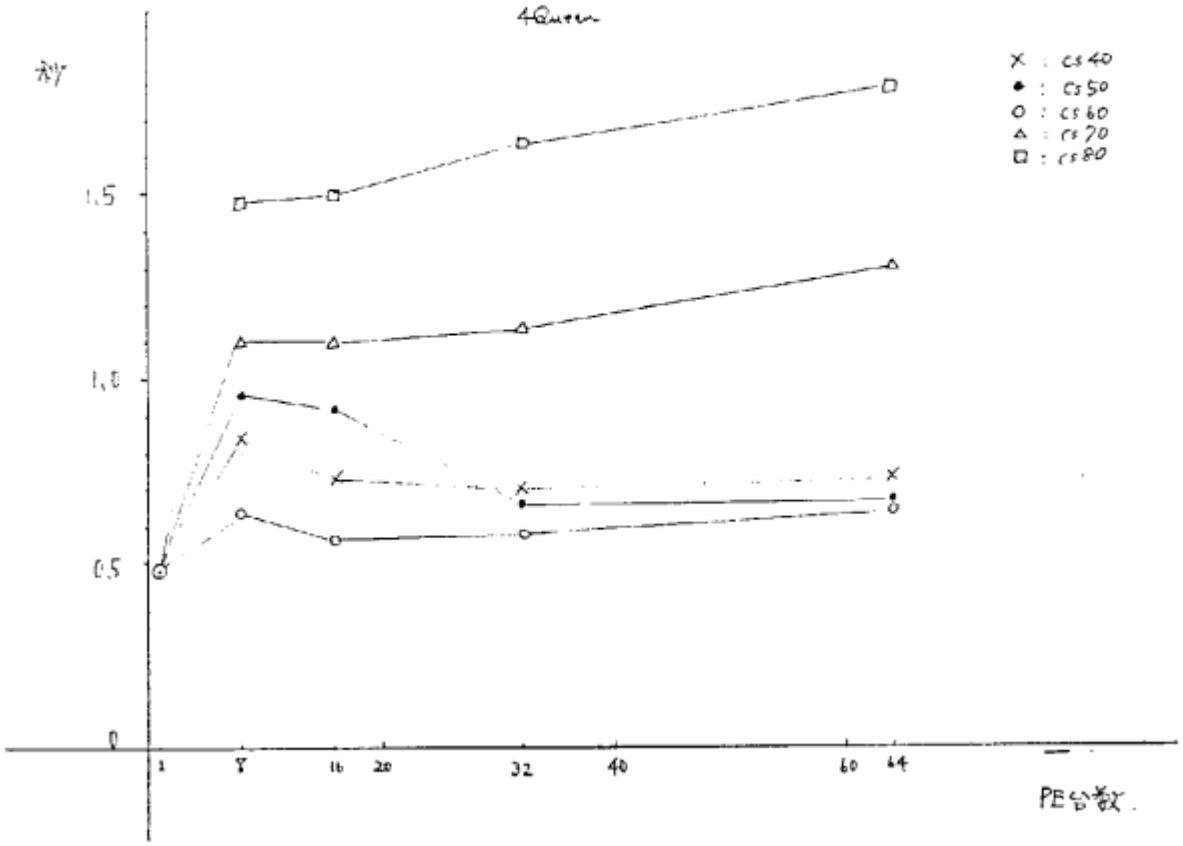
Application = N-Queen

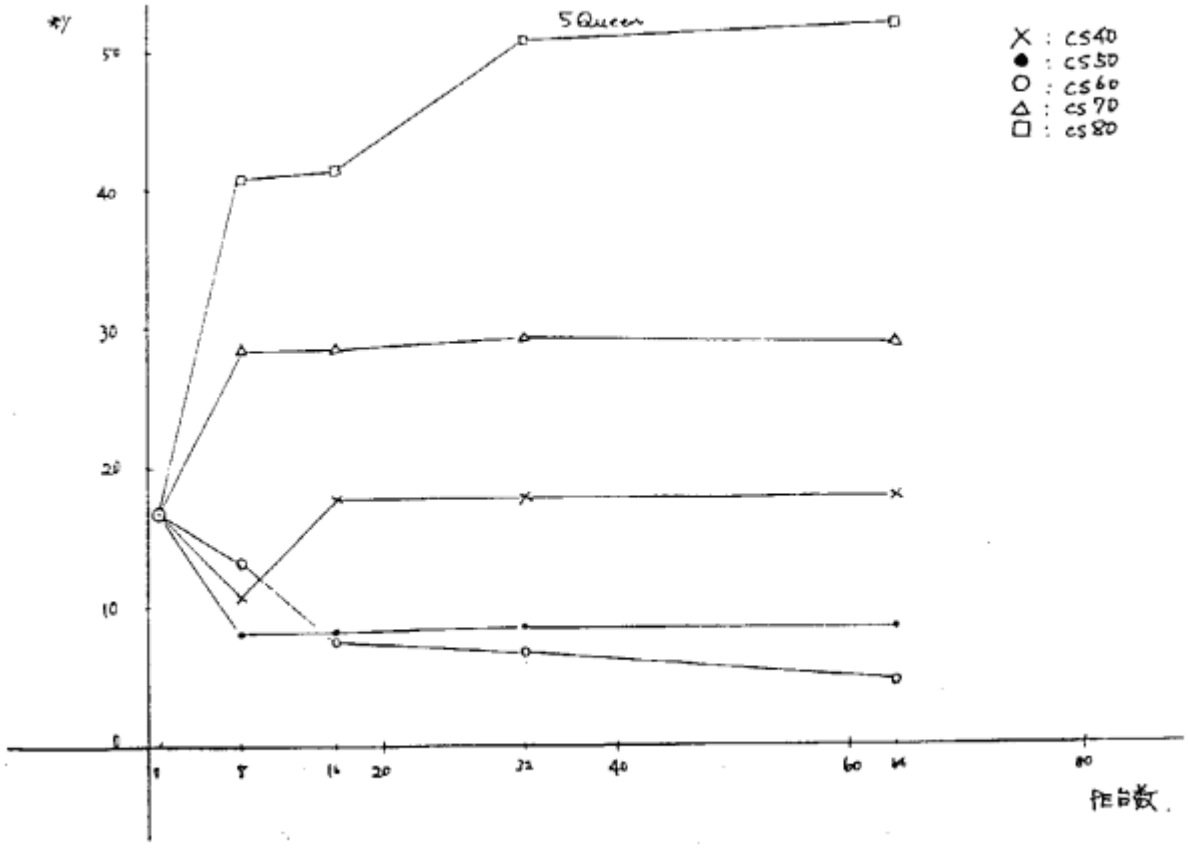
	8-PEs	16-PEs	32-PEs	64-PEs
N = 4	1-PE = 15,404red. 0.490sec. 31,437red/sec			
cs40	15,717 0.824 19,074	15,701 0.736	15,701 0.703 22,334	15,701 0.736
cs50	15,838 0.953 16,619	15,818 0.906	15,790 0.676 23,357	15,797 0.682
cs60	16,316 0.633 25,776	16,306 0.564 28,911	16,307 0.575	16,304 0.633
cs70	29,852 1.099	29,784 1.084 27,476	29,752 1.131	29,734 1.219 24,392
cs80	41,786 1.469 28,445	41,624 1.490	41,563 1.632	41,544 1.774 23,418
N = 5	1-PE = 566,447red. 16.780sec. 33,757red/sec			
cs40	566,964 10.881 52,106	566,962 17.816	566,956 17.952 31,582	566,956 17.870
cs50	567,115 8.236 68,858	567,114 8.492	567,130 8.585 66,092	567,112 8.491
cs60	575,891 13.215 43,579	574,383 7.541	574,404 6.799	573,850 4.760 120,557
cs70	1,058,527 28.767 36,796	1,056,123 28.949	1,055,029 29.458 35,815	1,054,474 28.991
cs80	1,469,341 40.876 35,946	1,464,033 41.413	1,460,777 50.736	1,459,011 51.089 28,550
N = 6	1-PE = 16,305,013red. 528.087sec. 30,876red/sec			
cs40	16,305,789 451.015 36,154	16,305,801 404.933	16,305,805 213.936 76,218	16,305,787 413.654
cs50	16,306,019 239.726	16,306,130 208.119	16,306,037 215.562	16,306,023 210.396

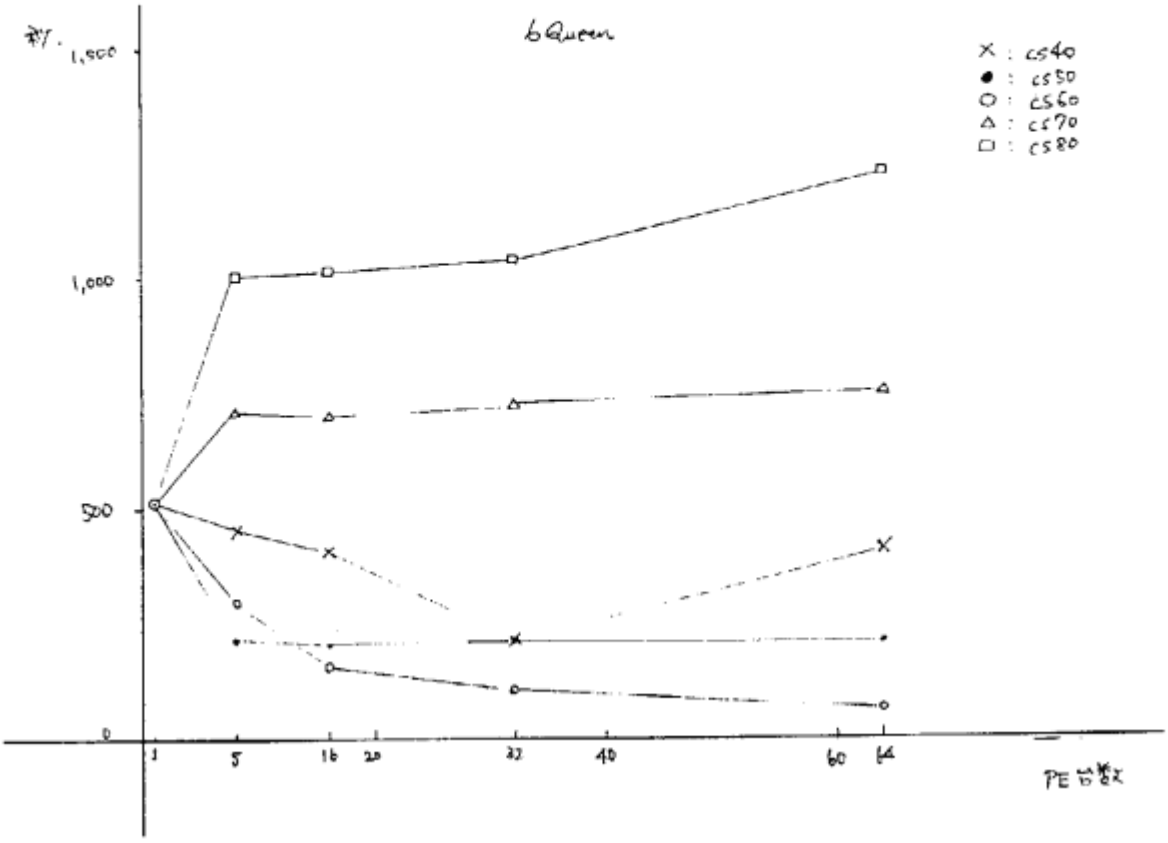
	68,019	78,350		
	16,505,367	16,522,157	16,483,739	16,446,626
cs60	299.555	166.827	102.890	66.864
	55,100	.		245,971
	28,357,362	28,298,810	28,271,904	28,258,991
cs70	714.966	698.620	727.881	757.403
		40,507		37,310
	38,423,487	38,300,801	38,235,390	38,182,477
cs80	1,014.958	1,040.207	1,084.208	1,251.983
	37,857			30,498

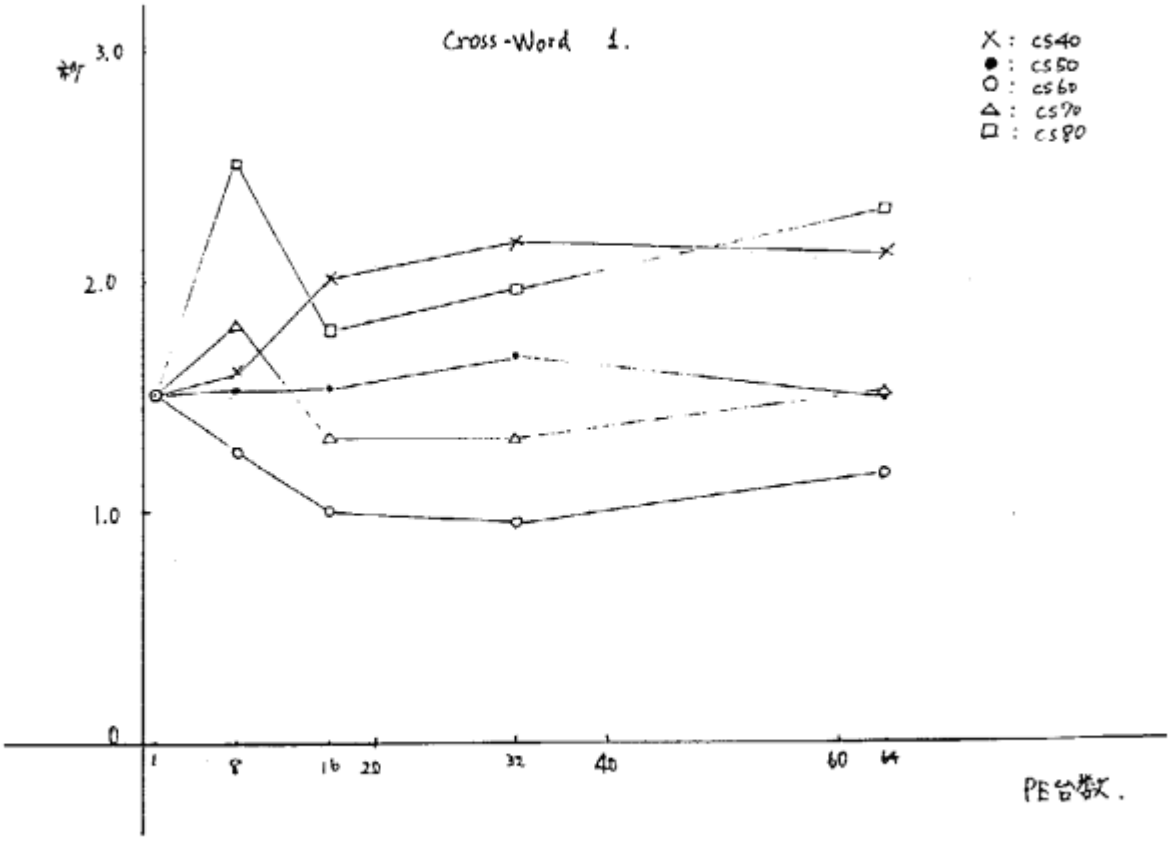

台数効果
15
10
5
0
台数効果
1
8
16
20
32
40
60
64
PE台数
○ : Cross-Word 2
△ : Cross-Word 3
□ : 6-Queen
モジュール : cs60

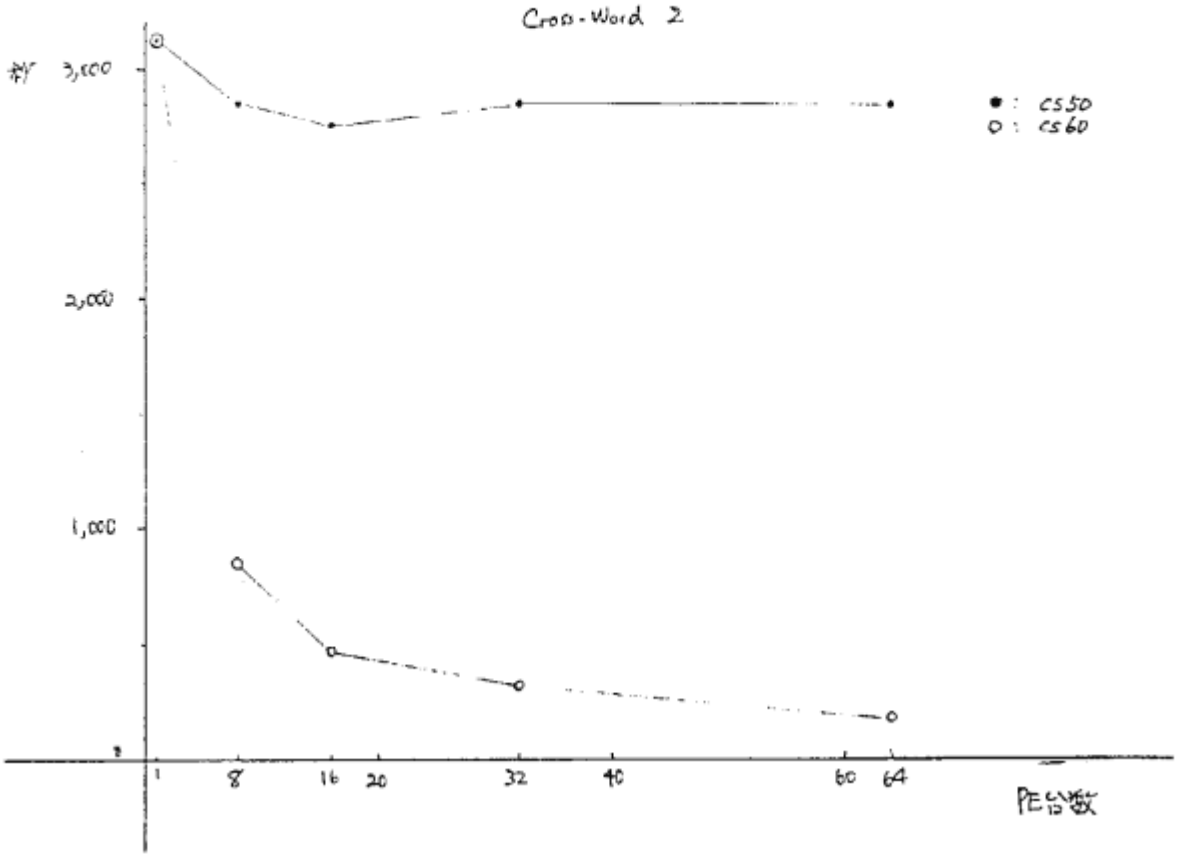

平均稼動率
稼動率
0.5
0.4
0.3
0.2
0.1
0
1
8
16
20
32
40
60
64
PE台数
○ : Cross-Word 2
△ : Cross-Word 3
□ : 6-Queen
モジュール : cs60

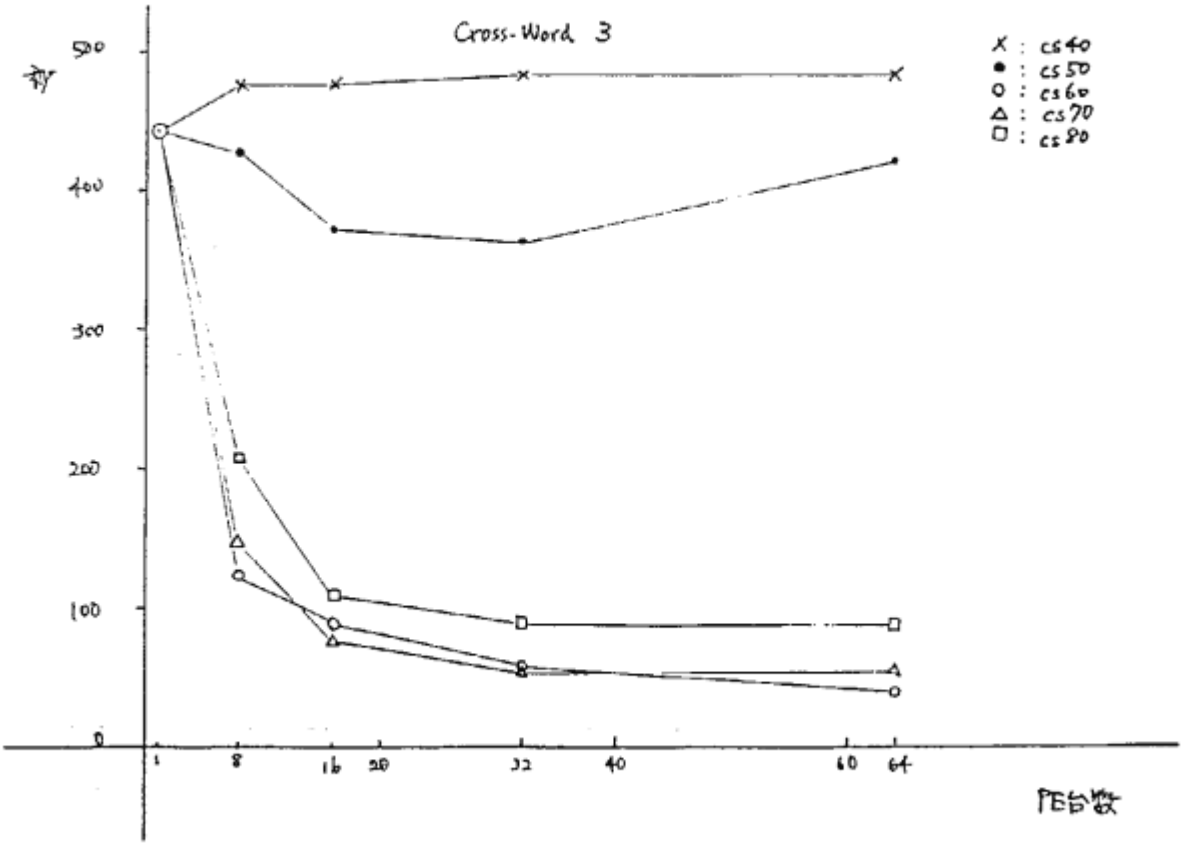

4Queen
×：cs40
●：cs50
○：cs60
△：cs70
□：cs80
1.5
1.0
0.5
0
1
7
16
20
32
40
60
64
PE台数


5Queen
×：cs40
●：cs50
○：cs60
△：cs70
□：cs80
50
40
30
20
10
0
7
14
20
32
40
60
80
PE台数


6Queen
×: cs40
●: cs50
○: cs60
△: cs70
□: cs80
1,000
500
0
1
5
16
20
32
40
60
64
PE台数


Cross-Word 1.
×: cs40
●: cs50
○: cs60
△: cs70
□: cs80
3.0
2.0
1.0
0
1
8
16
20
32
40
60
64
PE台数.


Cross-Word 2
3,000
2,000
1,000
0
1
8
16
20
32
40
60
64
● : cs50
○ : cs60


Cross-Word 3
500
400
300
200
100
0
1
8
16
20
32
40
60
64
× : cs40
● : cs50
○ : cs60
△ : cs70
□ : cs80