分散的並列計算による詰将棋の解法

野下 浩平, 中山 泰一, 松本 真一, 赤澤 忠文
電気通信大学 情報工学科

Solving Tsume-shogi in Parallel on Distributed Computers (Extended Abstract)

KOHEI NOSHITA, YASUICHI NAKAYAMA,
SHIN'ICHI MATSUMOTO AND TADAFUMI AKAZAWA
Department of Computer Science
The University of Electro-Communications
Chofu, Tokyo 182, Japan
noshita@cs.uec.ac.jp

Abstract

Two parallel algorithms for solving Tsume-shogi on distributed computers without shared main memory are presented. Eight to sixty-four network-connected workstations have been used for solving one hundred hard problems with nineteen to twenty-five solution-steps. The experimental results show that our parallel algorithms can solve most of the problems much more quickly than the best sequential algorithms made so far. In particular, for ten to twenty percent of those problems, the speed-up factor is more than the number of slave computers. Some other results also demonstrate the power of our algorithms.

1　はじめに

本論文では分散的に配置された 8～64 台のワークステーション（WS）を用いて，詰将棋を並列的に解く方法とその実験結果を報告する．詰手数が 17 手以下の短篇問題は，これまでの逐次プログラムで十分速く解ける．詰手数が 19～25 手の中篇問題は，T2 のような深さ優先探索を基本にした逐次プログラムによれば，ほぼすべてが解けるが，多くの問題に対して過度に時間がかかる [2, 5]．雑誌「詰将棋パラダイス」の 92，93 年度の短大部門（19～25 手詰）全 100 題に対して，T2 は 99 題解けるが，そのうち "短時間" に解けるのは 50～60 題程度である．ここで短時間とは普通の WS で大体 1 時間以内で解ける程度の時間を意味するものとする．一方，Ito のように最良優先探索を基本にした逐次プログラムによれば，解けるものは短時間，それも数分から十数分というようにごく短い時間に解けるが，全部の解を求めるのは難しい [2]．Ito によると上記 100 題のうち解けるのは約 80 %である．

本稿では P12 と P32 という 2 つの並列プログラムを説明し，おもに短大の 100 題に対する実験結果を示す．ここでは，共有メモリをもたない並列計算モデルで，1 個の主プログラム（マスター）と複数個の従プログラム（スレーブ）からなるものを用いる．P12 は，マスターにわりあい単純な総当たり的並列探索法を用い，スレーブに深さ優先探索の T2 を用いる．一方，P32 は，マスターに最良優先探索の T3 を用い，スレーブに T2 を用いる．T2 と T3 は逐次プログラムとして伊藤や河野などのプログラムとともに現在もっとも進んだものとされている [2, 3, 5]．

実験結果によると，並列計算により 19～25 手の問題に対して "短時間" に解ける範囲が大きく広がることがわかる．上記短大 100 題のうち約 90 %の問題は短時間に解くことができる．100 題の中で短時間に解けなかった問題約 10 題は，例外なく T2 で解くのに非常に長い時間かかるものである．

1 台の WS を使う逐次プログラム T2 と比較して，WS の台数以上の速度向上比が 10 ～ 20 %の問題に見られる．92 年度短大 50 題の中で，WS が 16 台の場合 16 倍以上の速度の向上をえた問題数は 12 題（24 %），64 台の場合 64 倍以上の速度の向上をえた問題数は 5 題（10 %）である．

2　並列アルゴリズム

アルゴリズムを作成するための並列モデルとして，共有メモリをもたない数台から数十台のコンピュータ（以下 CPU という）が互いにネットワークを経由して結合された分散

的システムの上で，マスター1台とスレーブ複数台があるという簡単なモデルを用いる．CPU 間の通信速度は1回あたり，数ミリ秒から数十ミリ秒である．つまり個々の CPU 内部の計算速度に較べて，通信速度が非常に遅い．このことより，CPU 間の通信をなるべく少なくすることがアルゴリズム設計に重要である．実際，そのために個々のスレーブの粒度（1回の仕事当りの平均的な所要時間）をかなり大きくする．本稿を通して，スレーブに深さ優先探索の逐次プログラム T2 を用いる．一般に並列計算モデルとしてもっと複雑なものがいろいろ考案されているが，実験結果をみれば，本稿の目的のためにはこのモデルで十分であることがわかる．本稿の分散的並列計算システムについては，文献 [1] などで報告する．

2.1 プログラム P12

P12 は，マスタープログラムに S1，スレーブプログラムに T2 を用いる．S1 は，わりあい素朴な総当り的並列探索に基づくものである．スレーブ T2 のアルゴリズムは深さ優先探索にもとづくものである．T2 の詰将棋特有の工夫などに関する詳細は文献 [4, 5] を参照されたい．S1 の基本的な考え方を次にまとめる．

（1）動的展開　先手局面に与える“展開値”に基づいてゲーム木を展開する．1回の展開は，先手局面から先手と後手の相続く2手指すことに対応する．各時点において，木の葉（先手局面）全体の中で最大の展開値をもつものを選択して次の時点で展開する．詰局面あるいは不詰局面が確定すると，通常通りミニマックスの規則によって木の簡約を行なう．

（2）展開値の計算　局面の評価値を求め，基礎点を決め，最後に選択のための展開値を計算する．

○ 評価関数　手番の局面でアルファベータ法による先読みを行い，後手玉の自由度により評価値を決める．これは T2 の評価関数とほぼ同じである．

○ 基礎点　先手番では，すべての先手の着手でできる局面の評価値をキーとして整列（ソート）する．整列結果の順にしたがって，各局面に

$$h, h-1, h-1, h-2, h-2, h-2, h-3, ..., 1, 1, ...1, 0, 0, ...0$$

という点を改めて与える．実験では $h=7$ に選ぶ．

各々の先手番の着手でできる後手局面では，同様にすべての後手の着手でできる先手局面の評価値を整列する．後手局面の上記の点を k とするとき，整列結果の順にしたがって，各先手局面に

$$k, 0, 0, 0, \ldots, 0$$

という点（基礎点）を与える．先頭の局面以外は0点である．

○ 展開値　展開のために選択された局面の展開値を v とする．展開の後にできる局面（2手指した局面）の新しい展開値は，その局面の基礎点を k とすると $C * v + k$ とする．根の展開値は0とする．実験では $C = 3$ に選ぶ．

○ 調整　後手局面が詰みであることが判明すると，その兄弟局面のうち，整列順で次に位置する B 個の局面（とその子孫）の展開値を通常より M 大きくする．ここで M は十分大きい値である．実験では $B = 2$ に選ぶ．

(3) スレーブの呼出し　スレーブはT2であるので，深さ t をパラメータに与えれば，反復深化によって，深さ t 以下の詰みがあればその深さと詰め手順，詰みがなければ深さ t 以下では詰まないことを報告する．全体の探索の深さを m とすると，マスターの探索の深さは $m - t$ になる．深さ t はスレーブの粒度を決める最大の要因であるが，通信に要する時間を考慮して $t \geqq 9$ とする．9より小さいとスレーブの純計算に要する時間が小さすぎ，通信の手間が相対的に大きくなる．T2の手数と時間の関係からみて，実験では $t = 17, 19, 21$ などを用いる．

このアルゴリズムは，先読みを利用した総当たり的な並列探索である．それは先読みによる順序付けにより，先手と後手の両方に強いと予想した手が左下から右方向に並んでいるからである．しかし，次に説明するように，純粋な左下優先探索の並列版とは異なる．実際，純粋な左下優先探索の並列版は実験によるとあまりよくない．

ここでは，基礎点の計算法により（よさそうな順に整列しているので）概ね左下優先になっているが，同じ深さの2つの節点の間で必ずしも左側が優先されるわけではない．これは展開値の計算法よりわかる．また，先手局面とは異なり，後手の基礎点の計算法により，2つ以上の後手局面が兄弟であっても普通は同時にスレーブに割り当てられることはない．これは1つの後手局面の詰まないことが判明すると，（ミニ局面であるので）兄弟の後手局面をすべて削除するが，スレーブに割り当てられ実行中である場合中断する必要があるからである．一方，ある局面の詰むことが判明すると，基礎点の調整によってその弟の

局面の優先度が非常に大きくなり，これらの局面をなるべく早く探索するようになる．これは評価関数で後手の最強と予想した着手によっても詰んだので，その弟局面の詰む可能性が高い（正解手順ないしはそれから派生する変化手順の上にある可能性が高い）と予想して，優先的に調べるという考えによる．

以上の考えは，良さが証明された方針というわけではないが，一種のヒューリスティックスであり，その善悪は実験によって評価することになる．

2.2 プログラム P32

P32 は，マスターに T3，スレーブに T2 を用いる．いずれも逐次プログラムとして作ったものであり，すでに多くの実験によってその解答能力はよくわかっている．T3 の特徴は次の通りである [5].

(1) 最良優先探索であり，原則として静的評価である．

(2) 展開の進行によって評価値の調整を行なうことがある．

(3) 後手（ミニ側）の展開は逐次的である．つまり後手の着手が詰みと判定された後にのみ後手は次の着手を展開する．

T3 では，先読みによる評価値の計算のほかに，さまざまなヒューリスティックスによる評価値の調整を行なう．また，ハッシュ法を使って，近似局面 [3] を拡張した“かなり似た局面”での着手の優先度の増加，評価値の計算，ほぼ同一局面の詰・不詰（現在は不詰のみ）の判定を行なう．さらに不要な節点の削除によりかなり徹底的に記憶領域を再利用する．後手の展開が逐次的であるが，記憶領域の節約のほか，無駄合の判定に利用する．

次に，T3 から T2 を呼ぶための基準を説明する．P32 には，スレーブの数 n のほかに，全体の深さ m とスレーブの深さ t をパラメータとして与える．T3 から T2 を呼ぶのは，評価値最大として選択された節点の深さが $m-t+1$ になった時である．ただし，この接続点での無駄合の判定はマスター T3 側で行なうので，接続点（とその先祖）に合と飛角香による取りの対が生じると，1 対につき 2 手（T2 を呼ばずに）T3 の探索をさらに深くする．スレーブ側 T2 の深さは一定 t であるが，T2 の中では無駄合に関連してそれ以上深く探索することがある．さらに，T2 を呼ぶための深さの節点に到達しても，その節点の評価値が十分大きい場合には，T3 の探索をさらに深くする．なお，T3 では，後手の 1 手と先手の全着手を対に展開するようになっているので，T2 に与える問題局面は，評価値最大として選択された節点の局面の親局面である．

P32 は，19〜25 手詰の問題をほぼすべて解くものと予想される．例えば $m = 25, t = 15$ と選べば，原則としてマスター側の深さは 11 ，スレーブ側の深さは 15 になる．マスターの T3 は，最良優先探索であるが，11 手詰までの問題であれば（記憶領域が不足しないで）ほぼすべて解ける．T3 の探索の葉に対応するスレーブは T2 であるので，15 手詰までの問題をほぼ完全に解答できる．

3 計算実験

スレーブは最大 64 台使うが，もっとも速い CPU（7 台）の速度に対して，もっとも遅い CPU の速度は約 1/4 である．主要な実験は，これらの CPU を用いたが，その中間的な速度をもつ CPU を約 10 台使ったものも多い．さらに，CPU はどれも共同利用であるので，他のユーザの使用状況に応じて速度が変わることが多い．それで計算実験で示す時間は，実際に要した時間であり，純 CPU 時間ではない．逐次プログラム T2 との比較のために T2 の解答時間を参照するが，この時間はもっとも速い CPU の約 1/2 の速さの CPU で測定したものである．こちらは CPU 時間であるので，一般に実際にかかった時間より小さい．次の記号を用いる．

n: スレーブの台数，m: 探索の全体の深さ，t: スレーブ T2 の探索の深さ．

計算実験では主として，92 と 93 年度の短大それぞれ 50 題の合計 100 題をとりあげる．不完全な問題が少数あるが，おおむね 19〜25 手詰である（93 年度には同じ問題が 1 対ある）．

詰手数	19 手	21 手	23 手	25 手	17 手以下 (余詰)	問題数の合計
92 年度	11	14	10	12	3	50
93 年度	10	13	15	10	2	50

3.1 P12

全体の深さ m を与える必要があるので，$m = 25$ とする．反復深化により m を変化させると，19 手詰の問題などが一層速く解けることがある．たとえば，19 手詰めの問題に $m = 25$ とすると，詰まない部分問題に対してスレーブの T2 は必要以上に深く探索するからである．しかし今回の実験では，(P12 に不利ではあるが) m を固定した．

○ **92 年度短大 全 50 題**　$m = 25, t = 21$ とする．$n = 64$ に対して

50 問題中　44 題が 3 時間以内で解けた．
　　　　　43 題が 1 時間以内で解けた．
　　　　　31 題が 10 分以内で解けた．

3 時間以内で解けた 44 題に対して，T2（$n = 1$），$n = 16$ の場合も含めて計算時間を比較したのが図 1 である．ここで，T2（$n = 1$）は $m = t$ であり，またこの T2 は原則として完全な探索を行なうもので，スレーブに用いたものとほぼ同じある．文献 [2] の T2 では一部に先手の着手の選択にヒューリスティックスを用いている．ただし，今回も T2（$n = 1$）で時間のかかりすぎる問題については，ヒューリスティックスを用いた場合の時間も用いている（このような特別扱いした 6 問題はどれも難しいものばかりであり，最短のものでも 8 時間かかる）．

台数以上の速度向上については次の通りである．

T2（$n = 1$）と比較して，$n = 64$ で P12 の速さが 64 倍以上になっているものは 5 問題である（10 %）．$n = 16$ で 16 倍以上になっているものは 12 問題である（24 %）．

50 題の中で 3 時間以内に解けなかった 6 題は次の通りである．

前期 9 番，前期 17 番，前期 18 番，後期 8 番，後期 13 番，後期 22 番

この中で後期 22 番だけは T2 でも解けていない．残りの 5 題は T2 が十分に時間をかけて解いた．Ito によると，50 題の中で 38 題解いているが，上記の 6 題はいずれも解けていない．

なお，$m = 25$ として，$t = 21$ のかわりに $t = 17, 19$ を用いることを調査したが，ほぼ同様の結果をえることがわかっている．

まとめると，P12 によると“短時間”で 50 題中の 90 %程度解けることがわかる．

3.2　P32

○ **92，93 年度短大全 100 題**　次の 2 段階で解く．ここで $n = 16$ であるが，一部 $n = 8$ とする．

（1）$m = 25, t = 15$ で制限時間 30 分，　（2）$m = 21, t = 15$ で制限時間 30 分

どちらかで解ければ 1 時間以内で解けることになる．

これで解けた問題数は，

92 年度 50 題中 42 題であり，
93 年度 50 題中 45 題である．

第 1 段階では，70 %弱の問題が 10 分以内で解ける．

92 年度の残りの 8 題の中で，3 題はそれぞれ 45 分，45 分，約 1 時間半で解ける．P32 は前期 9 番が解ける．残りの 5 題は P12 の残りの 5 題に一致する．すなわち，次の 5 題は 2 時間以内（$n = 8$）では解けない．

前期 17 番，前期 18 番，後期 8 番，後期 13 番，後期 22 番

93 年度の残りの 5 題の中で，2 題はそれぞれ 1 時間以内で解ける．次に示す残った 3 題はどれも 2 時間以内 ($n = 8$) では解けない．

前期 14 番，前期 23 番，後期 25 番

T2 は 93 年度の 50 題すべて解いたが，この 3 題には非常に時間がかかる．Ito は 40 題解いた．

以上より，P32 でも“短時間”で 100 題中の 90 %程度解けることがわかる．

○ 裸玉　裸玉問題の代表的な 20 題（詰将棋パラダイス，1975 年 2 月号）は河野により 10 題解いたとの報告がある（1994）．P32（と T2）では現在まで 14 題が解けている．その中に将棋図巧第 98 番 31 手詰も含む．ただし，P32 の開発中のテストの際に解けたものであり，(パラメータを固定した）一様なやり方で解いたものではないので，参考結果である．詰むという意味で正しい問題であることのチェックができたと考えてよい．

解けた 14 題の番号（括弧内は P32 の解の詰手数）：
1(31), 3(21), 5(19), 6(21), 7(15), 8(19), 9(17), 10(15), 12(21), 13(33), 16(17), 18(19), 19(15), 20(21)

なお 8 番は T2 で解いた．河野が解いたものはすべて P32 が解いた．

4　おわりに

実験結果をまとめると，P12 と P32 は短大の約 90 % を 1 時間以内に解くが，逐次プログラム T2 が 1 時間以内に解いたのは 50〜60 % であることをみれば，高速化に関して並列アルゴリズムの有効なことがわかる．また，深さ優先探索の逐次プログラムと比較して台数効果以上の速度向上を実現できる実例を示した．理論的には台数効果以上の速度向上は実現できそうにないが，ここでは特定の逐次プログラムとの比較であることに注意されたい．

謝辞

伊藤琢巳，河野泰人の両氏には実験データをはじめとして様々な示唆をうけた．両氏の実験データのうち参考文献に出ていないものは，大部分 CSA 例会での配布資料である．

参考文献

[1] 赤澤 忠文, 中山 泰一, 野下 浩平, ゲーム木の並列探索のための分散的実行管理機構の設計と実現, 第 49 回情報処理学会全国大会 (1994), 4U-7.

[2] 伊藤 琢巳, 野下 浩平, 詰将棋を速く解く 2 つのプログラムとその評価, 情報処理学会論文誌, **35**, 8 (1994), 1531-1539.

[3] Y. Kawano, Using similar positions to search game trees, *Combinatorial Game Workshop*, Berkeley (1994).

[4] K. Noshita, A program for solving Tsume-shogi quickly and accurately, *Proc. of the Game Playing System Workshop*, Tokyo (1991), 56-59.

[5] 野下 浩平, 詰将棋, 数学セミナー, **33**, 10 (1994), 26-27.

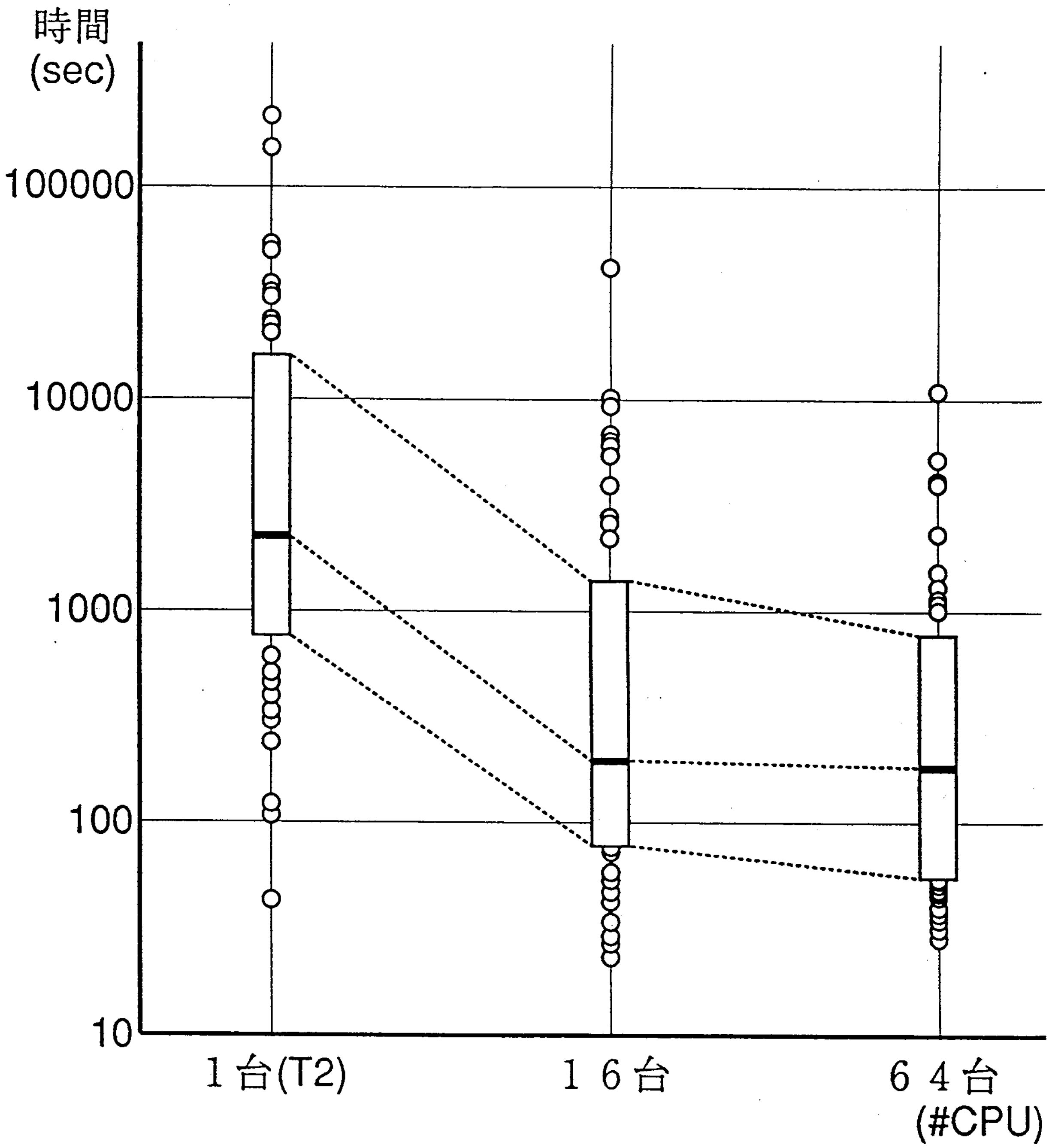

図 1: 計算時間の台数効果, ▬: 中央値 (median),
□: ±25 %, ○ : 実際の値 (actual value).

Fig. 1: Computation time for the number of CPUs.