ロビーサービスを用いた分散処理によるゲーム木の探索の高速化

郡司　直廉、森　秀樹、上原　稔

東洋大学工学部情報工学科

1. はじめに

一般に、チェス、将棋、囲碁、リバーシ（オセロ）などの二人零和有限確定完全情報ゲームの人工知能は、ゲーム木という探索木を用いて先読みを行い、より自分に有利な局面になるように手を進めるという方法を取る。しかしゲーム木の探索には、先読みの手数や分岐因子が多いほど時間がかかる。普通は枝刈りアルゴリズムを用いて無駄な探索を省いたり、定石（定跡）データベースを用いたりするので単純に先読みの手数と分岐因子だけでは比較することはできないが、探索速度はまだまだ向上させる必要があるといえる。[1][2]

本研究ではこのような問題点を解決するため、ロビーサービス型のゲームサーバと、それに接続した PC を用いて分散処理を行う方法を提案する。この方法の特徴は、インターネットに接続された汎用の PC を用いるため、専用コンピュータを用いる場合に比べ安価であるという点、ユーザーがゲームで遊んでいる間に CPU パワーを提供してもらうため、ユーザーがほとんど恩恵を得られない現在の各種分散コンピューティングプロジェクトに比べユーザー数を増やしやすいという点である。

本論文では、提案システムの設計と実装について議論し、実験を行うことで提案システムの有効性を確認する。

2. 関連研究

ゲーム木の探索を高速に行うために、IBM の Deep Blue[3]のような専用の並列コンピュータを使用する方法がある。

Deep Blue は POWER2 Super Chip(P2SC)を搭載したノードを 32 持つ並列コンピュータに、合計 512 個のチェス専用 VLSI プロセッサを搭載したチェス専用のコンピュータで、1 秒間に 2 億手を読む能力を備える。このようなコンピュータは非常に強力であるが高価で、一般人が気軽に人工知能と対戦するというようなことも不可能である。

また、低価格で高い計算能力を得る方法としては分散コンピューティングがある。これは、1 つの膨大な量の計算処理を複数に分け、それをインターネットに接続された PC に分散して処理をさせるもので、現在、distributed.net[4]や SETI@home[5]といったプロジェクトが行われている。

The speeding-up of Game-Tree Search for Lobby Service by Distributed Computing
Naoyuki Gunji, Hideki Mori, Minoru Uehara
Dept. of Information and Computer Sciences, Toyo University

この方法は、低コストで高い処理能力を得られるが、CPU パワーを提供したユーザーが得られる恩恵が無い（あるいは、得られる確率が非常に低い）ため、プロジェクトにそれなりに興味のある人からしか CPU パワーを提供してもらえない、という問題点がある。

3. 人工知能の仕様

本研究は、分散処理によるゲーム木の探索の高速化を目的としたものであり、ゲームには比較的単純なリバーシを用いた。

ゲーム木の探索を高速化するためには、分散化の他に、枝刈りや評価関数のアルゴリズムの最適化も重要である。しかし、これらは各ゲームに固有のものもあるため、最適化はゲーム毎に行う必要がある。そのため本研究では、各ゲームに依存する部分に関しては、人工知能の基本的なアルゴリズムとした。具体的には以下の通りである。

- ゲーム木の探索法は、一般的なミニマックス法を用いた深さ優先探索とした。
- 枝刈りのアルゴリズムには、各ゲームに依存しないアルファ・ベータ法を使用した。
- 評価関数は、リバーシの「相手に取られない石（角に置いた石など）の数が多いほど有利になる」という特徴から、そのような石に重みをつけて計算するアルゴリズムとした。

4. ゲームサーバのアーキテクチャ

本システムでは、通常のロビーサービスとは異なり、ゲームサーバ上でユーザー同士の対戦の他にユーザー対コンピュータ（人工知能）の対戦も行えるようにする。

図 1 にゲームサーバとクライアントの接続を示す。既存のロビーサービスと異なる点は、ユーザー同士だけでなく、人工知能との対戦も可能である点、各クライアント PC が人工知能の処理を行う専用のスレッド（人工知能ノード）を持ち、ゲームサーバは分散処理の中心となる人工知能サーバのスレッドを持つという点である。

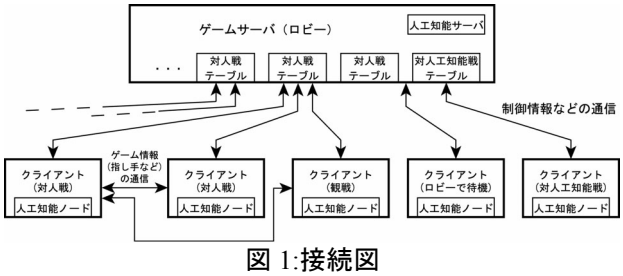

図 1:接続図

5. 探索アルゴリズム

ゲーム木の探索において、木の根に近い部分は人工知能サーバが、そこから先は複数の人工知能ノードが分散して探索を行う。図 2 にその模式図を示す。

この方法は、通信するデータのサイズが小さく、また通信の回数が少ないということが特徴である。このため、低速なネットワーク環境でも影響を受けにくい。

また、サーバがすべての探索範囲の割り当てを終えた後に、探索処理を終えたノード（A とする）が現れた場合、その時点でまだ探索を行っている他のノードの探索範囲を、A に対しても探索させるようにしている。（つまり、1 つの探索範囲を 2 つ以上のノードが同時に探索する状態になる。） このため、最後に大きな探索範囲があり、そこに低速な PC が割り当てられ、なかなか探索を終えずにいた場合でも、いずれ高速な PC がその探索範囲を探索することでより早く探索を終わらせられる可能性が出てくる。これにより低速な PC がボトルネックとなりにくくなっている。

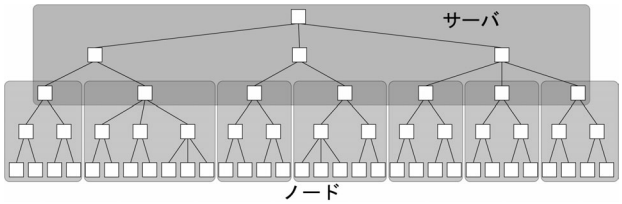

図 2:探索アルゴリズム 1

また、上記のアルゴリズムを一部変更した、分散用ノードを用いるアルゴリズムを示す。図 3 にその模式図を示す。この探索アルゴリズムは、サーバに分散用ノードを接続し、そこに人工知能ノードを接続する。つまり、サーバが探索した先を分散用ノードが探索し、その先を人工知能ノードが探索するというものである。

この方法は、ノードの数を増やしてもサーバにかかる負荷が増えにくいという利点があるが、分散用ノードが停止したときの損失が大きい、ノードを効率よく管理できない、通信の遅延が大きいといった欠点がある。

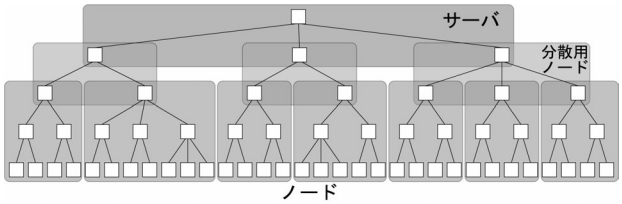

図 3:探索アルゴリズム 2

6. 実験と評価

分散処理を行う PC の台数を増やすことによる、探索速度及び効率の変化を調べるための実験を行った。低速の PC が混在した場合も想定し、表 1 に示す 5 台の PC を用意した。

実験内容は、PC1 でゲームサーバを、PC1～5 でクライアントを動作させ、探索範囲の大きくなる中盤の局面において 8 手先までの探索（平均探索局面数:2289023.5）を行い、探索に要する時間を測定するものとした。この実験を、クライアントの PC 数を 1 台～5 台まで変化させて行った。なお、ゲームサーバの動作する PC1 ではクライアントも同時に動作させている。その理由は、クライアント数が 5 程度ではゲームサーバは CPU パワーをほとんど消費しないためである。また、実験には探索アルゴリズム 1 のプログラムを用いた。

実験の結果を表 2 に示す。PC の台数や低速の PC の有無に関わらず、ほとんどの状況においてほぼ 100%の効率で分散処理が行われた。5 台の時に効率が 100%を上回る結果となったのは、PC1 台で探索を行った場合とは異なる順序で探索が行われたことで枝刈りの効率が良くなり、探索局面数が減少したためだと思われる。

	CPU	探索速度（局面/秒）
PC1	Athlon XP 2500+ (1.83GHz)	57523.96
PC2	Athlon XP 1700+ (1.47GHz)	49518.59
PC3	Athlon 1.4GHz	40410.91
PC4	Pentium II 233MHz	9120.17
PC5	MMX Pentium 166MHz	4692.11

表 1:実験に用いた PC の性能

クライアントを動作させる PC	探索時間(秒)		分散処理の効率(%)
	実効値	理論値	
PC1	39.204	39.281	100.20
PC1,2	23.195	21.168	91.26
PC1,2,3	15.571	15.368	98.70
PC1,2,3,4	14.719	14.480	98.38
PC1,2,3,4,5	13.258	14.060	106.05

表 2:実験の結果

7. むすび

本論文では、ゲーム木の探索を高速化する手法として、ロビーサービス型のゲームサーバとそれに接続したクライアント PC を用いる方法を述べた。また、実験を通じ、提案システムが有効であることが確認できた。今後は、バナー広告のような形でサイト訪問者の CPU パワーを利用する Java アプレットの実装などについて研究を進める。

参考文献

[1] 馬場口　登, 山田　誠二: 人工知能の基礎, 昭晃堂 (1999)
[2] 松原　仁: 将棋とコンピュータ, 共立出版株式会社 (1994)
[3] IBM: Kasparov vs. Deep Blue, http://www.research.ibm.com/deepblue/
[4] distributed.net: http://www.distributed.net/
[5] SETI@Home: http://setiathome.ssl.berkeley.edu/