B-035

GPU を考慮した並列分散 GA の高速処理
Applying Distributed Genetic Algorithm to GPU

井沼　安広 †　　小柳　滋 †
Yasuhiro INUMA　　Shigeru OYANAGI

1. はじめに

画像処理に使われる GPU(Graphics Processing Unit) を，汎用計算用途に使用する GPGPU(General Purpose computation on GPU) が近年注目されている．この GPGPU を利用し，メタヒューリスティックアルゴリズムである遺伝的アルゴリズム (Genetic Algorithm: GA) を高速に処理しようとする研究が行われている．[1]-[3]

遺伝的アルゴリズムは，生物の進化を模倣したアルゴリズムで，個体群に対し遺伝的オペレータである「評価」「選択」「交叉」「突然変異」の操作を繰り返し適用し，最適解に近似していく．各遺伝的オペレータは，並列処理が可能であり，評価のみを GPU に処理させる手法 [3] と，評価以外も GPU に処理させる手法 [1][2] が提案されている．後者の手法は，GPU の弱点である CPU と GPU 間のアクセスオーバーヘッドを最小限に抑えることができ，高速化が期待できる．

並列分散 GA(Parallel Distributed GA: PDGA) は，母集団を複数のサブ母集団に分割し一定間隔で移住と呼ばれる操作を行うことで，GA を並列分散環境で実行できるように拡張したものである．

本稿では GPGPU(CUDA) 環境上での並列分散 GA について，個体群の中から高速に高適応度個体を選び出しエリートとして保存する手法を提案する．

2. 既存研究

GPU を利用した GA の高速処理に関する既存研究を 2 つ紹介する．

筒井らの研究 [1] では，サブ母集団を GPU の Block に割り当て Block 間で移住操作を行う形で並列分散 GA を GPU に適用させている．二次割り当て問題を対象とし，CPU と比較して最大 23.9 倍もの高速化ができたと報告している．

大礒らの研究 [2] では，バイトニックソートを用いたエリート保存戦略を実装し，GPU 上で GA を実行する手法を提案している．バイトニックソートは，対象データの数が 2^n 個でなければならないという制約があるが，並列環境上でも実装が可能なソーティングアルゴリズムである．進化ロボティクス問題を対象とし，CPU と比較して最大 23.0 倍もの高速化ができたと報告している．

†立命館大学 情報理工学部, Department of Information Science and Technology, Ritsumeikan University.

3. 提案手法

大礒らの研究 [2] でも報告されているが，個体群に対し厳密にランク付けを行うソートでは，個体の入れ替え操作が発生するため，処理時間が多くなってしまいがちである．そこで我々は，個体群に対してソートは行わずに高適応度な個体を選び出すことで，処理量を削減しつつ解探索性能の維持または向上が期待できるのではないかと考え，次に述べるエリート選択手法を提案する．

以下，2^n個 のスレッドにより並列処理される，個体 $[x](x = 0, 1, ..., 2^n - 1)$ を考える．

3.1 (Step0) ポインタ初期化

エリート個体候補の番号を保持するポインタ (Pointer) を，SharedMemory で 2^n個 確保し，$Pointer[x] = x$ で初期化する．

3.2 (Step1) 隣接個体間で適応度を比較

個体 $[x]$ の評価済み適応度が，$Fitness[x]$ に格納されているものとする．スレッド $y(y = 0, 1, ..., (2^n - 1)/2)$ では，$Fitness[2y]$ と $Fitness[2y + 1]$ を比較する．(図 1)

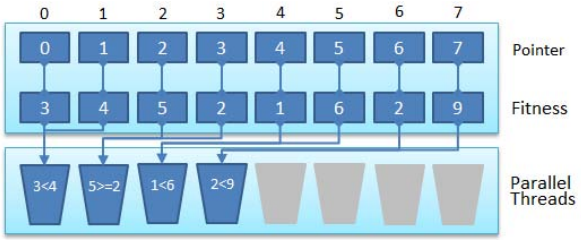

図 1: (Step1) 隣接個体間で適応度を比較

3.3 (Step2) 比較結果をもとに，ポインタを更新

(Step1) で比較した結果，$Fitenss[2y] \geq Fitness[2y+1]$ ならば $Pointer[y] = 2y$，$Fitenss[2y] < Fitness[2y+1]$ ならば $Pointer[y] = 2y + 1$ とする．ここで，個体 $[Pointer[y]]$ をエリート個体候補として選択する．(図 2)

3.4 (Step3) 繰り返し適用

選択されるエリート個体候補が 1 個になるまで，(Step1)-(Step2) を繰り返す．最終的に，個体 $[Pointer[0]]$ が最高適応度を持つ個体となるので，これをエリートとして保存する．(図 3)

これ以外の個体には，「交叉」「突然変異」オペレータを適用する．

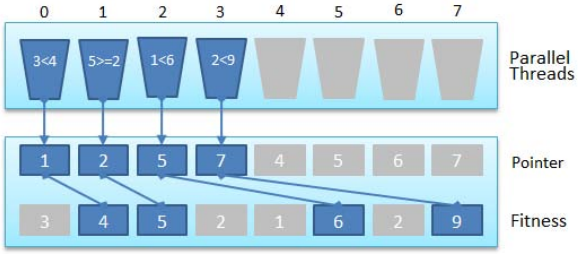

図 2: (Step2) 比較結果をもとに，ポインタを更新

図 3: (Step3) 繰り返し適用

4. 実装とベンチマーク

筒井らの研究 [1] と同様にサブ母集団を GPU の Block に割り当て Block 間で移住操作を行う形で並列分散 GA を GPU に適用した．また，並列分散 GA の移住トポロジーは，8 つのサブ母集団に対して，Ring 型トポロジーを設定した．

ベンチマークに使用する問題は，巡回セールスマン問題 (TSP) とし，都市データは TSPLIB の eil51.tsp(51 都市) を用いる．

また，CPU 側で通常の GA(既存手法とする) を実行し，処理時間と適応度の推移を比較する．設定した GA のパラメータを表 1 に，実行環境を表 2 に示す．

表 1: GA のパラメータ

-	Proposal	Conventional
Total Population	128*8=1024	1024
Sub population	128	-
Generation	20000	20000
Migration interval	5	-
Migration rate	0.6	-
Selection	Tournament	Tournament
Elite Selection	Proposal	Bubble sort
Elite	1	1
Crossover	2 Point	2 Point
Mutation	Bit	Bit
Mutation rate	0.05	0.05

表 2: 実行環境

CPU	Intel Core i5 2500K 3.60GHz
メモリ	4.00GB
GPU	NVIDIA GeForce 320M 0.95GHz

4.1 評価

図 4 は，適応度の推移を表したグラフである．同じ世代数でも，既存手法より提案手法のほうが，良い解に到達していることがわかる．

また処理時間は，提案手法が 32378.45(msec)，既存手法が 330626.2(msec) となり，提案手法は既存手法に比べ，約 10.2 倍の高速化を達成した．

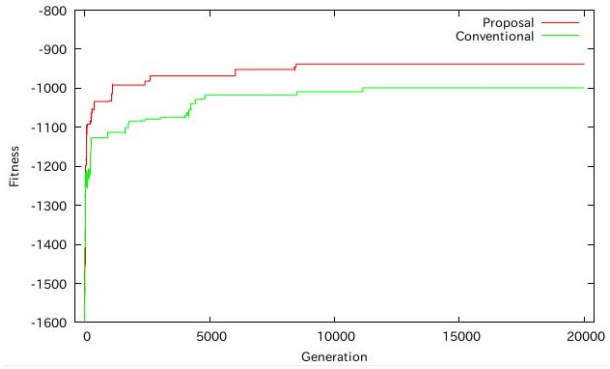

図 4: 適応度の推移

5. おわりに

本稿では，GPU を考慮した並列分散 GA による GA の処理時間削減と解探索性能向上を目指した．巡回セールスマン問題 (51 都市) を対象とした結果，既存手法と比較して，解探索性能が向上し約 10.2 倍の高速化に成功した．

参考文献

[1] S.Tsutsui and N.Fujimoto.“ Solving Quadratic Assignment Problems by Genetic Algorithms with GPU Computation: a case study ”, Proceedings of GECCO 2009 Workshop on Computational Intelligence on Consumer Games and Graphic Hardware(CIGPU-2009), pp2523-2530 (2009)

[2] 大礒 正嗣, 松村 嘉之, 保田 俊行, 大倉 和博,“ CUDA 環境におけるデータ並列化を用いた遺伝的アルゴリズムの実装手法 ”, 知能と情報, Vol.23, No.1.pp.18-28, (2011)

[3] 雀堂 浩司,“ GPU 計算を用いた進化計算の高速化 ”, 融合情報学輪講資料, (2010)