4K-05

GA を用いた消込処理問題への GPU の適用とハミング距離を考慮した局所探索の併用による探索能力の向上

奥由輔† 佐藤裕二‡

†‡法政大学情報科学部コンピュータ科学科

1. まえがき

消込処理とは，売掛金などが入金された際に明細書と照合し，入金の遅延や売掛金が請求通りであるかを確認する作業を示す．企業間の取引では，請求の合計額と入金額が一致しない場合があるため，入金額を基に対応する請求の内訳を求める必要がある．

上記問題は，部分和問題[1]の会計額が大きい特別なケースとして定義することができ，部分和問題は NP 完全問題[2]であることが分かっている．また，一般的な部分和問題は動的計画法（Dynamic Programming, DP）[3]で効率的に解けることが示されている．しかし，DP は，入金額の大きさに比例した大量のメモリを要するため，企業間の取引で扱うような高額な入金額の場合，適用できない．従って，DP と比べてメモリサイズを抑えることができる遺伝的アルゴリズム（Genetic Algorithm, GA）を用いた消込処理問題の解法と，ハミング距離を考慮した局所探索の併用による探索能力の向上を提案し，その有用性を示した[4,5]．本稿では，Graphics Processing Unit （GPU）を用いた並列高速化を提案し，その評価を行う．

2. 消込処理のための GA プログラムの設計

消込処理問題における染色体の定義は，各請求項目にそれぞれ遺伝子座を対応させ，値が“1”である部分に対応した請求が入金されたと考える．従って，染色体の定義長は全請求件数と一致する．

遺伝的操作における評価値は，請求金額の合計額と入金額の差額の絶対値の逆数を設定するのが妥当であるが，消込処理問題は複雑な形状を持つ多峰性の探索問題であることが分かっているため[4,5]，探索空間の起伏が極端に大きくなってしまう場合がある．そこで、0 から 1 の間の値に正規化した式(1)に示す評価関数を定義する．

$$f_e = \frac{f_p}{f_p + f_s} \tag{1}$$

ここで，f_p は染色体の合計額であり，f_s は f_p と入金額との差の絶対値を示している．f_e=1 となった個体が入金額に対応する個体であり，最適解となる．

選択と交叉方法は，予備実験の結果，ルーレット選択と一様交叉の組合せの際に全請求件数 N に依存せず同一の交叉率，突然変異率で安定した収束性を得られたため，ルーレット選択と一様交叉の組合せで評価実験を行う．

3. ハミング距離を考慮した局所探索

組合せ件数のほぼ等しい他の組合せを探索する局所探索として，ハミング距離を考慮した局所探索を提案した[4,5]．本稿では，ハミング距離が“0”または“1”（HM0or1）を用いて評価する．

ハミング距離を考慮した局所探索（HM0）と単純突然変異または一般的な局所探索（MU）の違いを示す概念図を図 1 に示す．MU は，現時点でのエリート個体の周辺を探索するのに対し，HM0 は，原点を基準値としてエリート個体と原点からのハミング距離がほぼ等しくなる同心円上を探索する．従って，だまし問題や複雑な形状を持つ多峰性の探索問題である消込処理問題には，有効であると考えられる．

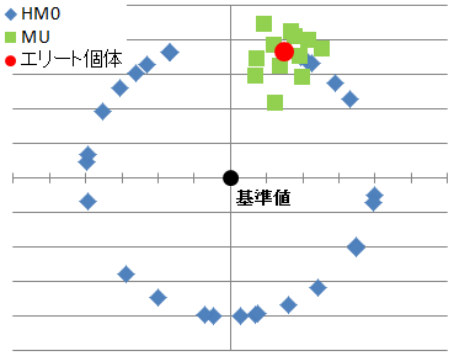

図 1　ハミング距離を考慮した局所探索と一般的な局所探索の概念図

4. GPU 環境での GA プログラムの設計

4.1. GPU の概要

ここでは，GPU は NVIDIA GeForce GTX970 を用いる．GPU は，複数の Streaming Multi-processor（SM）を持ち，SM は複数の Thread-Processer（CUDA コア）を持つ．GPU の各スレッドの処理速度は CPU と比較して低速だが，多数のスレッドを並列実行し，高速化を図る．スレッドはブロックに，ブロックはグリッドにグループ化され，階層的に管理されている．さらに，スレッドはワープと呼ばれる 32 スレッドに纏められており，ワープ内のスレッドが異なる分岐結果を実行する場合，ワープ内の全スレッドが両方の条件分岐コードを実行するため，待ち時間が生じる．また，一つの配列に対して，連続したスレッドが連続したアドレスに同時にアクセスする場合，高速なアクセスが可能となる．GPU のメモリモデルは，スレッド固有のレジスタ，ブロック内共有のシェアードメモリ，グリッド共有のグローバルメモリに階層化されており，それぞれのメモリアクセスの局所性が高いほど，高速かつ低容量である．また，各 SM に搭載されているシェアードメモリは，生成された各ブロックに対して 48KB 以内に制限されている．

4.2. GPU 環境での GA の実装手法

GPU 環境での GA 実装の提案モデルを図 2 に示す．図 2 は，母集団を一旦グローバルメモリに格納した後に 13 の SM に分割し，さらに 52 のブロックに分割して割当て，192 のスレッドに対して各個体および，ハミング距離を考慮した局所探索を割当てるモデルを示している．図 2 中の括弧内の数字は，その集団での個体数を示している．予備実験の結果，ブロック数 52，ブロック内の集団（ブロック集団）の個体数 192 の組み合わせ（1SM あたり

Applying GPU to Credit Erasure Processing Problems with GA and Improving in Searching Capability with Local Search by Taking Hamming Distances
†Yusuke Oku, Hosei University
‡Yuji Sato, Hosei University

Copyright ©2015 Information Processing Society of Japan.
All Rights Reserved.

768 個体）の際に効率的な収束性を得られたため，ここでは，上記組合せを用いる．

GPU 上で島モデル型 GA を実装する際，SM と同数のブロックを生成し，各ブロックにサブ集団を割当てると考えやすい．しかし，消込処理で扱う 100 を超える遺伝子長の場合，コア数に対して十分な個体数を設定すると，各ブロックに対するシェアードメモリ容量の制限値を超えてしまう問題が生じる．この問題を解決するため，各 SM に割当てたサブ集団をさらに複数のブロックに分割し，集団を char 型で定義してシェアードメモリの使用量を抑える．また，メモリ参照を多用する GA の特徴を考慮して，各ブロックでの遺伝的操作は，メモリ参照による遅延が少ないレジスタとシェアードメモリのみを用いる実装手法とする．

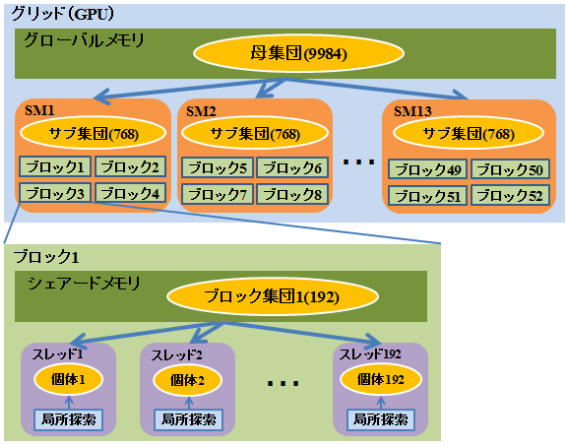

図２　GPU 環境での GA の実装モデル

4.3 GPU 環境での遺伝的操作

GPU 環境での遺伝的操作は，CPU と同様にルーレット選択と一様交叉の組合せを用いて，各スレッドに対して 2 個体選択し，交叉・突然変異および局所探索を行う．また，CUDA では，ワープ単位で処理が実行されるため，各サブ集団の個体数は，SM あたりのコア数 128 とワープ内のスレッド数 32 の公倍数から決定する．

ブロック集団の合計評価値の算出手順を図 3 に示す．図 3 は，シェアードメモリ内の配列に対して，後半部分の値を前半部分の値に足し込んでいく処理を示している．連続したスレッドは連続したアドレスにアクセスするため，コアレスアクセスによって高速にメモリアクセスを行うことが出来る．さらに，各 STEP 内の演算処理は，並列実行可能なため，実行時間のオーダーは，要素数 n として $O(\log_2 n)$となる．従って，本手法では，実行回数とダイバージェント分岐を大幅に削減できる．また，評価値の最大値を算出する場合も同様の処理を行う．

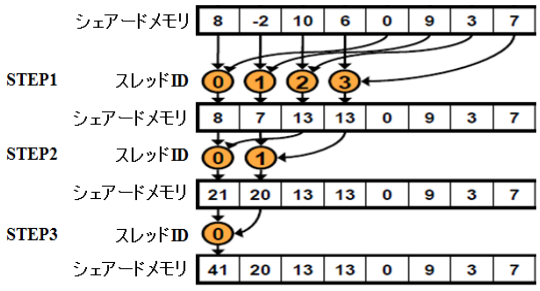

図３　シェアードメモリ内の配列の合計値算出例

ハミング距離を考慮した局所探索では，各スレッドの個体に対して，原点からのハミング距離がほぼ等しい個体を 3 個体生成し，その中で最も評価値の高い個体をそのスレッドの次世代個体とする処理を全世代で行う．従って，局所探索では，スレッド間で情報共有が行われないため，レジスタのみを用いた処理が可能となる．

5. 評価実験

各請求金額 3～9 桁，全請求件数 N=74，入金件数 k=34, or 46，入金額 S=424,744,482 のテストデータを用いて，GA-CPU と GA-GPU および，GA-CPU に HM0or1 を追加した場合（GA-CPU+HM0or1）と GA-GPU に HM0or1 を追加した場合（GA-GPU+HM0ro1）の実行時間の比較結果を表 1 に示す．GA は 50 回の平均実行時間を記録している．表 1 より，GA-GPU は GA-CPU と比較して約 58 倍高速に最適解に収束している．また，GA-CPU+HM0or1 は GA-CPU と比較して約 4.2 倍高速に最適解に収束しているのに対し，GA-GPU+HM0or1 は GA-GPU と比較して約 6.5 倍高速に最適解に収束している．この要因として，CPU では，探索点の個体を比較的低速な静的メモリ（グローバル変数）に格納しているのに対し，GPU では，探索点の個体をレジスタに格納していることが考えられるが，より詳細な考察が今後の課題である．また，GA-GPU+HM0or1 は，GA-CPU と比較して約 350 倍高速に最適解に収束しており，今回提案した GPU を用いた GA の並列高速化が有効に働いていると考えられる．

表１　GA-CPU と GA-GPU の比較

	Execution time (Average)	Execution time (Worst)	Generations (Average)	Generations (Worst)
GA-CPU	16m25s	55m57s	216499	788910
GA-GPU	17.0s	1m15s	19576	86984
GA-CPU +HM0or1	3m56s	19m48s	38331	193574
GA-GPU +HM0or1	2.8s	11.1s	2444	9909

6. むすび

本稿では，消込処理問題の GA を用いた解法に関して，ハミング距離を考慮した局所探索を併用した GA の GPU 上での実装方法と，配列に対してコアレスアクセスを行うことで，ダイバージェント分岐の削減を実現し，CPU での標準的 GA と比較して約 350 倍の高速化が図れることを示した．今後，全請求件数 N を実際の企業間の取引で扱われる値に設定した評価実験，局所探索点の数の考察，および分枝限定法[6]などとの比較が課題である．

文　献

[1] http://ja.wikipedia.org/wiki/%E9%83%A8%E5%88%86%E5%92%8C%E5%95%8F%E9%A1%8C (cited Mar.24, 2013)

[2] 福田和真，萩原斉，中森眞理雄．ナップザック問題の確率アルゴリズムの解析．数理解析研究所講究録 947 巻, pp. 162-171 (1996)

[3] 長尾智晴著,”最適化アルゴリズム”, 昭晃堂 (2006)

[4] 武田由美，奥由輔，岸田隆視，佐藤裕二．消込処理問題への GA の適用とハミング距離を考慮した局所探索能力の向上について．第 7 回進化計算シンポジウム講演論文集 2013.

[5] Yuji Sato, Yusuke Oku, Masanori Fukuda. Applying GA with Local Search by Taking Hamming Distances into Consideration to Credit Erasure Processing Problems. pp. 1183-1190 GECCO-2014.

[6] http://ja.wikipedia.org/wiki/%E5%88%86%E6%9E%9D%E9%99%90%E5%AE%9A%E6%B3%95 (cited May.29, 2014)

Copyright ©2015 Information Processing Society of Japan.
All Rights Reserved.