可逆アルゴリズムの調査

田島 嘉人

1　可逆コンピューティングについて

可逆コンピューティングは，大別すると 2 つの性質を持つ．1 つは論理的な可逆性．もう 1 つは，物理的な可逆性である．論理的な可逆性は，初期状態と最終状態を除いた全ての状態が直前と直後の状態を一意に定めることができる性質をもつ．つまり，非可逆な計算と異なり，計算中に出現した全ての情報を保持する．この性質を持つことで得られる利点は，ハードウェアエラー検出や，失われたデータの復元 (悪意のある攻撃によってデータが改竄されてしまったときなど)、マルチプロセッサでの投機的実行など、様々な実際の問題に応用できることであるが、もっとも重要な利点として、物理的可逆性を実現するために必要であるという事実がある．物理的可逆性をもつコンピュータは熱の発生を防ぐことができる．つまり，コンピュータのエネルギー使用を抑え，より効率的に動作させることができる．このようなコンピュータを作成することが，可逆コンピューティング研究の目的である．

可逆コンピュータをプログラムするためには，次の 3 つの要素が必要である．

- 可逆プロセッサ
- 可逆言語
- 可逆アルゴリズム

2　可逆プロセッサ

3　可逆言語

Janus や R がある．

4　可逆アルゴリズム

非可逆のアルゴリズムは可逆アルゴリズムに変換することができる．その一般的な方法は，埋め込み法と Bennett 法を用いることである．埋め込み法は，計算過程で失われる情報を出力の一部として保存することにより，可逆性を実現する．そして，Bennett 法は，埋め込み法を用いることで発生した情報を整理する．また，これらの方法を用いず，経験や勘からアルゴリズムを作成することで，後述する時間計算量・空間計算量・ゴミ出力量の観点から，より効率の良いアルゴリズムを作成できる [1][2][3].

4.1　アルゴリズムの解析手法

アルゴリズムには，その性能を客観的に測るためのいくつかの指標がある．これらの指標を用いてプログラムの性能を測ることを，アルゴリズムの解析と呼ぶ．この節では，これらの解析手法について解説する．

4.1.1　時間計算量

時間計算量は，アルゴリズムを実行した際にかかる時間を表す数値であり，一般的には，O 記法を用いて表される．

4.1.2　空間計算量

空間計算量は，実行したプログラムが使用する記憶領域の量を表す．一般的には，変数や配列の宣言で増加する．

4.1.3　ゴミ出力量

ゴミ出力量は，可逆アルゴリズムを考えるときに用いられる指標である．アルゴリズムを実行した際に出力されるデータ中で，問題の解決に必要な情報を保持していないものはゴミ情報と呼ばれる．なぜ，不必要な情報が出力されるのかというと，可逆アルゴリズムでは，逆実行するために，これらが必要となるからである．ゴミ情報の内，実行途中で出るゴミ情報を中間ゴミ，実行終了後に出るゴミ情報を最終ゴミと呼ぶ．

5　文献調査

5.1　Reversible Computation and Reversible Programming Languages[4]

この論文は，可逆計算及び可逆プログラミング言語に対して，その特徴や特性を分かりやすくまとめたものである．可逆プログラミング言語の例として，Janus を取り上げており，Janus の構文や制御構造，操作的意味論について解説している．また，可逆計算や可逆プログラミングのサーベイ論文でもある．

5.2　Reversibility for Efficient Computing[5]

現在調査中。【9.5 Reversible algorithms を読んでまとめましょう】

【CIS 6930.3753X Spr.'02 Readings for Part V: Reversible Computing, Lecture 34 を参照する】

5.3　効率的な可逆線形探索と木構造の可逆深さ優先探索の設計と解析 [1]

この研究では，効率的な可逆線形探索と可逆深さ優先探索の設計・実装を行うため，次の 3 つについて確かめた．

- 効率的な可逆線形探索において，入力ファイルのデータ構造や出力によって空間計算量は変化するのか [検証 1].
- ゴミ出力量が 0 という条件下で，効率的な可逆線形探索の時間・空間計算量にどのような関係が現れるのか [検証 2].
- 効率的な可逆深さ優先探索のアルゴリズムを設計・実装する [検証 3].

ここで，効率化されたアルゴリズムとは，可逆に適するよう設計されたアルゴリズムのことを指す．

[検証 1] については，可逆線形探索アルゴリズムでは，効率化された可逆アルゴリズム下においては，出力の種類や入力ファイルのデータ構造によって計算複雑度が変化することが確かめられた．

[検証 2] については，実行時間と入力ファイルの操作回数にトレードオフ関係が見つかり，また，探索の成否によって入力ファイルの操作回数に変化が生じることが確かめられた．

[検証 3] については，Bennett 法を用いたものに比べ優れた点を持つ可逆アルゴリズムを提案した (実装が今後の課題)．

5.4 可逆線形探索の解析と可逆深さ優先探索の設計 [2]

この研究では，現在提案されている可逆深さ優先探索に対してより効率化されたアルゴリズムを提案した．

5.5 二分木のランク計算のクリーン可逆シミュレーション [3]

この論文では，二分木のランク・アンランク計算を行う可逆アルゴリズムを提案している．

参考文献

[1] 増田大輝：効率的な可逆線形探索と木構造の可逆深さ優先探索の設計と解析，南山大学 2019 年度修士論文 (2020).

[2] 戸川淳平，鳥居大樹，吉田翔亮：可逆線形探索の解析と可逆深さ優先探索の設計，南山大学 2019 年度卒業論文 (2020).

[3] 大久保雄飛，横山哲郎，金山知俊：二分木のランク計算のクリーン可逆シミュレーション，日本ソフトウェア科学会第 33 回大会講演論文集，pp.1–12 (2016).

[4] Yokoyama, T.: Reversible computation and reversible programming languages, *Electronic Notes in Theoretical Computer Science*, Vol.253, No.6, pp.71–81 (2010).

[5] Frank, M.P. and Knight Jr, T.F.: Reversibility for efficient computing, PhD Thesis, MIT (1999).

[6] Tyagi, N., Lynch, J. and Demaine, E.D.: Toward an Energy Efficient Language and Compiler for (Partially) Reversible Algorithms, *Reversible Computation* (Devitt, S. and Lanese, I., Eds.), Cham, Springer International Publishing, pp.121–136 (2016).

[7] Demaine, E.D., Lynch, J., Mirano, G.J. and Tyagi, N.: Energy-Efficient Algorithms, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science*, New York, NY, USA, Association for Computing Machinery, p.321–332 (2016).

付録 A 要チェック（横山より）

2016 年の MIT の研究も要チェック→ [6, 7]