リストの可逆分割アルゴリズムを利用した ゴミ情報が最適な可逆クイック整列法の生成

山下　健太　　横山　哲郎
南山大学情報理工学部ソフトウェア工学科

1　はじめに

本論文では文献 [1] におけるゴミ情報の生成・マージの手法を応用し，より一般性の高い手法を提案する．文献 [1] では，ゴミ情報量が最適なリストの可逆分割アルゴリズムを提案している．ここではさらに提案したアルゴリズムを利用してクイックソートを生成し，分割の際のゴミ情報量が最適であること，再帰呼び出しの際にゴミ情報の最適な伝播が実現できていることの 2 点を示している．しかし，この手法はクイックソートに特有のサブプロシージャに対して最適なゴミ情報の生成を保証することにより実現されており，この手法をそのまま他のアルゴリズムに応用することが難しい．そこで，本研究ではアルゴリズムのより小さな構成要素に最適なゴミ情報を生成させ，それらの最適性を保ったままより大きな構成要素へとマージを行う手法を提案し，既存の手法の応用性を高める．なお，ここではゴミ情報とは非可逆アルゴリズムを可逆化する際に記録する必要があり，可逆化する前の非可逆アルゴリズムにおいては記録する必要のない情報を表す．

2　関連研究

2.1　可逆アルゴリズム

可逆アルゴリズムとは，出力から入力を復号できるアルゴリズムである．なお，この性質は可逆性と呼ばれる．本来は可逆性をもたないアルゴリズムについても，ゴミ情報と呼ばれる入力を特定するための情報を出力に追加することで可逆化を行うことができる．

2.2　In-Place アルゴリズム

In-Place アルゴリズムとは，入力されたデータを直接改変することによって処理を行うアルゴリズムであり，可逆化を行うためにはゴミ情報を出力する必要がある．例えば n 個の要素をもつリストをソートする In-Place アルゴリズムを可逆化する場合，$\log_2 n!$ ビットのゴミ情報を出力する必要がある [2] [3].

Generating Reversible Quicksort with Minimal Garbage Size through Use of Split Algorithm of Lists
Kenta YAMASHITA　Tetsuo YOKOYAMA
Department of Software Engineering, Faculty of Information Sciences and Engineering, Nanzan University

3　可逆クイックソート

3.1　既存の手法

文献 [1] では，従来のリストの分割操作に代わり，次のアルゴリズム $RSplit_F(L)$ を提案している．また，このアルゴリズムを利用したクイックソートのアルゴリズム $Q'(L)$ を示している． $RSplit_F(L)$ はリスト L を入力とし，L の先頭の要素をピボットとして L をサブリスト Y_1，Y_2，Y_3 に分割する．分割後，Y_1 の要素の L における位置 $\{x_i\}_{i=1}^{|Y_1|}$ を関数

$$f(\{x_i\}_{i=1}^{k}) = \sum_{i=1}^{k} x_i - 1 C_i \tag{1}$$

によって $RIndex$ に変換し，これを記録する．文献 [1] では $RSplit_F(L)$ を利用してリストの分割を行うことで，可逆クイックソート $Q'(L)$ を実現している．可逆性を実

Algorithm 1 Reversible Split algorithm: Forward computing （[1] より改変）

Input: a discrete list L.
Output: a three-layered series list (Y_1, Y_2, Y_3) **and reversal index :** $RIndex$
$RSplit\_F(L)$:
$(Y_1, Y_2, Y_3) \leftarrow Split(L), RIndex \leftarrow f(\{pos(y_i) - 1\}_{y_i \in Y_1})$
return $(Y_1, Y_2, Y_3), RIndex$

Algorithm 2 Reversible Quicksort algorithm: Forward computing （[1] より改変）

Input: a discrete list L
Output: a linear list L^* **and reversal index**
$Q'(L)$:
if $|L| \leq 1$ **then**
　return $(L, 1)$
else
　$(Y_1, Y_2, Y_3), C_0 \leftarrow RSplit\_F(L)$
　$(Y_1, C_1) \leftarrow Q'(Y_1), (Y_3, C_2) \leftarrow Q'(Y_3)$
　return $([Y_1 : Y_2 : Y_3], |Y_3|(|L| - 1)! + C_0|Y_1|!|Y_3|! + (C_1 - 1)|Y_3|! + C_2)$
end if

現するために，$Q'(L)$ は 4 つの整数 $|Y_3|$，C_0，C_1，C_2 を記録する必要がある．これらはそれぞれ Y_3 の要素数，*RIndex*，Y_1 を復号するための情報，Y_3 を復号するための情報を表す．これらを 4 つの値として記録する場合，$Q'(L)$ が再帰呼び出しを行うたびに返り値の数が大きくなる．そのため，$Q'(L)$ では式

$$\begin{aligned} \textit{reversal index} = \quad & |Y_3|(|L|-1)! + C_0|Y_1|!|Y_3|! \\ & +(C_1-1)|Y_3|! + C_2 \end{aligned} \tag{2}$$

を用いて 4 つの値から *reversal index* を算出し，これを記録する．文献 [1] において，$\{pos(y_i)-1\}_{y_i \in Y_1}$ と *RIndex* の値が全単射の関係にあることが証明されているため，*reversal index* によって可逆性が実現できていること，情報を最適にマージできていることが確認できる．しかし，文献 [1] ではゴミ情報が最適にマージされている原理が明らかにされていない．また，文献 [1] における手法は 4 つの整数をマージするものであり，一般性が低い．そのため，現時点では既存の手法を他のアルゴリズムに応用することが難しい．

3.2 ゴミ情報の定式化と演算子 ⋈

本研究ではゴミ情報を 2 つの整数の組として定式化し，2 つのゴミ情報をマージして *reversal index* を求める演算子 ⋈ を提案する．演算子 ⋈ による演算は，式

$$(g_1, h_1) \bowtie (g_2, h_2) = (g_1 \times h_2 + g_2, h_1 \times h_2) \tag{3}$$

として定義できる．ただし，n は自然数，g_n はゴミ情報の値，h_n は g_n の取り得る最大値に 1 を加算した値とする．また，ここで演算子 ⋈ は結合性をもつ．演算子 ⋈ を用いて 4 つの整数をマージし，式 (2) と同一の *reversal index* を生成可能である．式 (2) の *reversal index* の最適性が証明済みであるため，⋈ によってゴミ情報の最適なマージができていることが確認できる．

3.3 プロセスを細分化した可逆分割アルゴリズム

Algorithm1 の処理を細分化した結果が次の Algorithm3 である．Algorithm3 は入力としてリスト L，サブリスト (Y_1, Y_2, Y_3)，ゴミ情報 *RIndex* を受け取り，リスト L の先頭の要素を適切なサブリストに格納する．要素を Y_1 に格納した際はゴミ情報 ${}_{|Y_1|+|Y_3|-1}C_{|Y_1|}$ を生成し，*RIndex* に加算することでマージを行う．以上の処理を再帰的に繰り返し，Algorithm3 はリストの分割を行う．

Algorithm 3 Reversible Split algorithm: Forward computing （Algorithm1 より改変）

Input: a discrete list L, **a three-layered series list** (Y_1, Y_2, Y_3), **and reversal index :** *RIndex*.
Output: a three-layered series list (Y_1, Y_2, Y_3), **and reversal index :** *RIndex*.
$RSplit\_F'(L, (Y_1, Y_2, Y_3), RIndex)$:
if $Y_2 = \emptyset$ **then**
  $Y_2 \cup L[0]$, $L \leftarrow tail(L)$
else if $L[0] < Y_2[0]$ **then**
  $Y_1 \cup L[0]$, $L \leftarrow tail(L)$
  $RIndex \leftarrow RIndex + {}_{|Y_1|+|Y_3|-1}C_{|Y_1|}$
else
  $Y_3 \cup L[0]$, $L \leftarrow tail(L)$
end if
if $L \neq \emptyset$ **then**
  $((Y_1, Y_2, Y_3), RIndex) \leftarrow RSplit\_F'(L, (Y_1, Y_2, Y_3), RIndex)$
end if
return $((Y_1, Y_2, Y_3), RIndex)$

4 おわりに

本研究では，ゴミ情報の定式化・ゴミ情報のマージを行う演算子 ⋈ の定義を行った．これらを利用し，既存の可逆クイックソートのプロセスの細分化，ゴミ情報量が最適な可逆マージソートを実現を達成した．今後の課題として，サブリストの要素の位置の組み合わせ以外を記録するためのゴミ情報の生成プロセスの確立が挙げられる．これを実現できれば，リストの分割・マージを利用したアルゴリズム以外のアルゴリズムに対しても，本研究における提案の適用が可能になると考えられる．

謝辞

本研究は JSPS 科研費 25730049 の助成を受けたものです．

参考文献

[1] Early, D., Gao, A. and Schellekens, M.: Frugal Encoding in Reversible $\mathcal{MOQA}$: A Case Study for Quicksort, *Proc. Reversible Computation*, Glück, R. and Yokoyama, T. (Eds.), Lecture Notes in Computer Science, Vol. 7581, Springer-Verlag, pp. 85–96 (2012).

[2] Perumalla, K. S.: *Introduction to Reversible Computing*, CRC Press (2013).

[3] Hall, J. S.: A Reversible Instruction Set Architecture and Algorithms, *Proc. Physics and Computation*, IEEE Press, pp. 128–134 (1994).