論文内容の要旨

論文題目

Analysis and Applications of the T-complexity

(T-complexity の解析と応用)

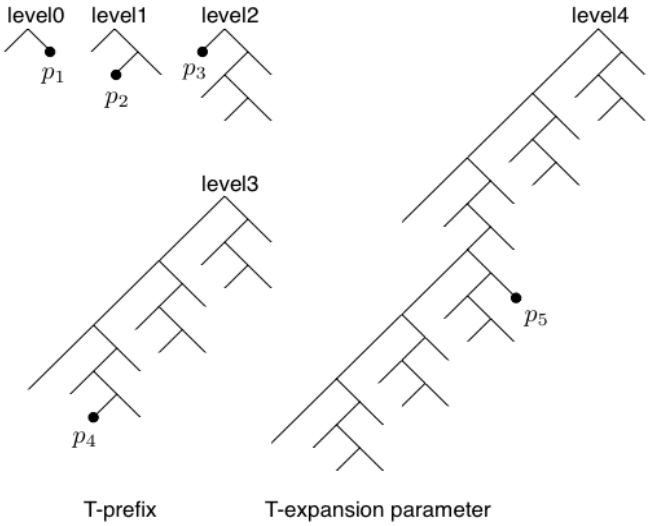
氏名　濱野　健二

本論文では，NIST 乱数検定の LZ 検定の問題点に対処するにあたって直面した T-code と T-complexity の解析と応用に関する問題に取り組み，次の項目を明らかにした．(1) 本論文で提案するアルゴリズム（forward T-decomposition）によって逐次的な T-decomposition が実現できること．(2) 本論文で提案する微分方程式の手法を使って理論的に T-complexity のプロファイルが導出できること．(3) 乱数系列に対する T-complexity の分布が特徴づけられること及び T-complexity 最大系列の特性が明らかにできること．(4) T-code に基づいて効率的なユニバーサルデータ圧縮が構成できること及びその性能が UNIX の‘compress’よりも良いこと．以上の特性により，T-complexity が系列のランダム性の良い指標になる．これらの特性を踏まえて，最後に T-complexity に基づく乱数検定（T-complexity 検定）を提案し，それによって LZ 検定の問題点の解決と NIST 乱数検定の強化ができることを示した．

本論文の第 1 章で述べた研究背景の概要は次のとおりである．暗号の安全性評価には，擬似乱数生成器からの出力系列のランダム性の計測が含まれる．その方法には，(1) 統計的手法による計測 (2) 情報理論の立場に基づく系列の圧縮性による計測がある．後者の方法の究極は，コルモゴロフ複雑度である．系列 s のコルモゴロフ複雑度 K(s)は，s を生成する最も短いプログラムサイズである．K(s)が s のサイズより小さければ，s はランダムではないと考える．このように，系列の複雑度と圧縮は密接に関連するが，一般に K(s)は計算不可能である．LZ-complexity はユニバーサルデータ圧縮 LZ78 に基づく複雑度であり，コルモゴロフ複雑度の推定に広く用いられる．NIST 乱数検定は，米国商務省標準技術局 NIST の文書 SP 800-22 が規定する乱数検定セットであり，暗号の統計的評価のために標準的に使われてきた．NIST 乱数検定には LZ-complexity に

基づく乱数検定（LZ 検定）が含まれていたが，LZ 検定には問題があり，2008 年に公式に除外された．情報セキュリティの分野では複雑度に基づく評価が極めて重要であるため，LZ 検定の問題の解決が望まれるが，LZ-complexity を使う限り解決は困難である．一方，T-code に基づく複雑度（T-complexity）は，LZ-complexity と類似性がある上に，LZ-complexity よりも系列の再帰的構造を良く検出できることが期待できる．そこで，T-complexity 検定を構成して LZ 検定の問題を解決することを考えた．しかし，その前に，T-code と T-complexity には次の問題がある．(1) LZ-complexity を求めるための LZ78 増分分解法は逐次処理できるが，T-complexity を求めるための T-decomposition は，アルゴリズム開始時に系列全体が必要なので逐次処理できない．(2) LZ78 系の効率的なユニバーサルデータ圧縮が存在するが，T-code に基づく効率的なデータ圧縮は存在しない．(3) T-complexity の最大値は乱数系列で達成されず，乱数系列より大きな T-complexity を持つ系列が存在する．

以上の問題を踏まえて，冒頭に述べた項目を第 2 章以降で順番に示した．以下，T-code の基本事項を説明した後，本論文を構成する章の内容を述べる．



T-prefix			T-expansion parameter		
p_1	=	1	k_1	=	1
p_2	=	10	k_2	=	1
p_3	=	0	k_3	=	2
p_4	=	001010	k_4	=	1
p_5	=	00101011			

図 1　レベル 0～4 の T-code の符号木．左方向の枝に 0，右方向の枝に 1 が対応する．

T-code の符号語は，符号木を再帰的に成長させることにより，レベル 1，レベル 2，レベル 3…の順に生成する．レベル i の符号語の集合を S_i とおく．S_0 はアルファベット A とする．S_i から S_{i+1} を生成する方法は次のようになる．S_i から 1 つの符号語を選び，それを T-prefix p_{i+1} とし，正整数を 1 つ選び，それを T-expansion parameter k_{i+1} とする．S_i を表す符号木を考えると，S_{i+1} の符号木は，S_i の符号木の p_{i+1} に対応する葉に S_i の符号木全体を k_{i+1} 回コピーして得られる．A={0,1}のときの生成例を図 1 に示す．与えられた系列 s を分解して，系列 s が最長符号語に対応する T-code のパラメタを決定することを T-decomposition という．T-complexity は，T-decomposition が決定した T-expansion parameter の値をもとに計算できる．T-decomposition は系列を分解して，T-prefix を順番に連結した形にする．各 T-prefix は，それまでの T-prefix を再帰的に連結し，最後に 1 シンボルを付けた形になる．LZ78 増分分解法では，各ワードが，それまでのワードの中で自身に最も長く一致するワードの後ろに 1 シンボルを付けた形になる．このように，LZ78 増分分解法と T-decomposition には類似性がある．

第 2 章では，系列を先頭から逐次処理する T-decomposition アルゴリズムを提案した．2.2 節では，T-expansion parameter を 1 に固定する場合，2.3 節では，一般の場合のアルゴリズムを提案した．本アルゴリズムは，図 2 に例示したようにトライ木を成長させながら実行する．トライ木 t_i は i 番目までの T-prefix を記憶し，i 番目の T-prefix は根から番号 i を持つ節点までの経路として表現される．系列を読み込みながら，トライ木を根から葉に向かって辿って T-prefix の構成要素を 1 つずつ決定することを繰り返すことにより逐次的な T-decomposition が実現できる．本アルゴリズムの計算量のオーダーと計算時間の計測結果も記した．計算時間は実用的な系列長で十分に短い．また，従来の T-decomposition と異なり逐次処理するため，オンラインアプリケーションにも向いている．

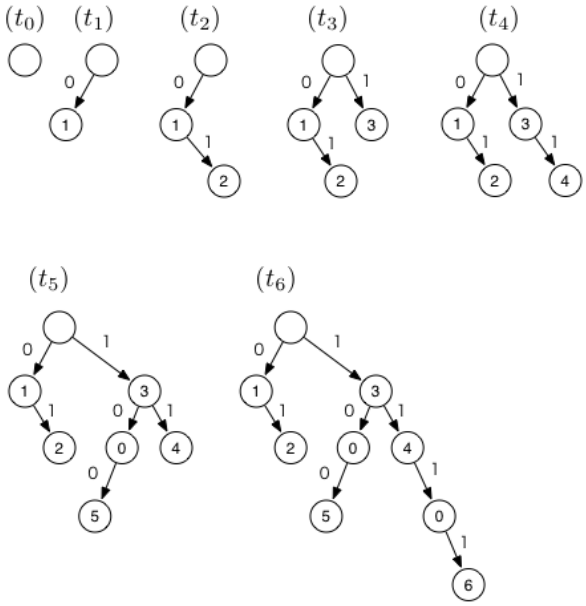


図 2　トライ木の成長過程

第 3 章は，T-complexity 及び LZ-complexity のプロファイルの理論的導出に関する研究である．Titchener は，T-complexity 最大系列のプロファイルが対数積分関数（li 関数）で表現できることを実験で予想した．その後，Titchener らは，li 関数をトップダウン的に使って，この予想を理論的に証明した（cf. 3.2 節）．3.3 節では，T-code の符号木の平均符号語長に関する微分方程式に基づいて T-complexity 最大系列のプロファイルを導出する方法を提案した．この導出法は，プロファイルが li 関数を使って必然的に表現されることをボトムアップ的に示すことができる点で，先行研究の証明法より優れている．同様の導出法を使って，3.3 節の後半では，乱数系列の T-complexity のプロファイルを，3.4 節では，LZ-complexity 最大系列のプロファイル及び乱数系列の LZ-complexity のプロファイルを求めた．これらの導出結果が，先行研究等の結果と合致することも確認した．このように，本章で提案した導出法には一般性がある．

第 4 章では，T-complexity 最大系列の特性を NIST 乱数検定，離散フーリエ変換によるスペクトラム，自己相関関数を使って解析した．4.2 節では，T-complexity 最大系列と LZ-complexity 最大系列の生成アルゴリズムを示した．4.3 節では，T-complexity 最大系列と LZ-complexity 最大系列の解析結果を示した．この解析結果により，T-complexity 最大系列は LZ-complexity 最大系列よりも非ランダムであることを確認した．また，T-complexity 最大系列の非ランダム性の原因を定性的，定量的に考察した．

第 5 章では, T-cod に基づく辞書式データ圧縮法を提案した. 5.2 節では, LZ78 系圧縮を概説し, 5.3 節では，T-code に基づくデータ圧縮の唯一の先行研究をまとめ，その問題点を整理した．5.4 節では，本提案法の詳細を記した．本提案法は，辞書に追加されるフレーズに T-code の再帰的構造がある．また， 3 方式（A, B, C）のフレーズ作成規則を検討した．5.5 節では，従来法，UNIX の‘compress’及び本提案法の性能を比較した（表 1）．本提案法は，従来法及び UNIX の‘compress’より圧縮率が良い．なお，C 方式のユニバーサル性も考察した．

表 1　Calgary コーパスに対する圧縮率の比較

Name	従来法	comp.	A	B	C
bib	0.500	0.418	0.348	0.376	0.392
book1	0.550	0.432	0.410	0.393	0.400
book2	0.510	0.411	0.381	0.385	0.375
geo	0.691	0.760	0.609	0.635	0.640
news	0.545	0.483	0.428	0.431	0.436
obj1	0.635	0.653	0.543	0.552	0.567
obj2	0.495	0.521	0.412	0.443	0.455
paper1	0.555	0.472	0.437	0.448	0.440
paper2	0.543	0.440	0.428	0.426	0.417
paper3	0.571	0.476	0.462	0.462	0.453
paper4	0.589	0.524	0.479	0.497	0.494
paper5	0.604	0.550	0.512	0.505	0.513
paper6	0.558	0.491	0.440	0.445	0.452
pic	0.120	0.121	0.107	0.107	0.113
progc	0.557	0.483	0.431	0.435	0.445
progl	0.340	0.379	0.310	0.327	0.342
progp	0.401	0.389	0.318	0.327	0.347
trans	0.376	0.408	0.292	0.313	0.361

第 6 章では，6.1 節で暗号分野における乱数検定の研究背景を述べた後，6.2 節で T-complexity 検定の構成法を示した．本検定で乱数系列を検定したときに異常がないこと（LZ 検定の問題点を解決すること）を確認した．6.3 節では，2 種類の非ランダム系列に対して，T-complexity 検定は，NIST 乱数検定に含まれる全ての検定，LZ 検定，2006 年に提案された修正 LZ 検定よりも検出力が著しく良いことを示した．

第 7 章で本論文の成果を総括し，今後の課題を次のように述べた．(1) T-complexity 検定を普及させること．(2) forward T-decomposition のアルゴリズムを改良して計算量を削減すること．(3) T-complexity の漸近分布の数学的証明を与えること．