コルモゴロフ複雑性に基づくプロダクト派生木復元の試み

Product Evolution Estimation Based on Kolmogorov Complexity

早瀬 康裕* 神田 哲也† 石尾 隆‡

あらまし 本論文では，コルモゴロフ複雑性の考えに基づいて，ソフトウェアプロダクトの派生関係グラフを推定する手法を提案する．プロダクトのソースコードを可逆圧縮したときの，データ量の増分が少ないプロダクトの間に派生関係があるものと推定する．提案手法を既存手法 PRET と比較した結果に，出力される派生関係グラフの誤りの傾向が異なることが分かった．

Summary. This paper proposes a method to estimate a product evolution graph based on Kolmogorov Complexity. The method applies lossless compression to source code of products, then, presumes derivation relations between two products when information increment between the two products is minimum. An evaluation experiment confirms that proposed method and existing method *PRET* have different tendency of errors in estimation results.

1 はじめに

既存のソフトウェアプロダクトに類似したソフトウェアプロダクトを開発する際に，既存プロダクトのソースコードを複製し，それを必要に応じて修正することで新しいプロダクト (派生プロダクト) を作ることがある．派生プロダクトは元となったプロダクトと多くのコードを共有しているため，保守を効率的に行ったり，頻繁に使用される共通部分を集約するためには，派生関係を一元的に記録しておくことが望ましい．しかし，現実のソフトウェア開発では，派生関係の記録が失われてしまうことがある．

失なわれたプロダクトの派生関係を復元するために，神田ら [4] は，2 つのプロダクトにおいて，双方に含まれる内容の類似したファイル (共通ファイル) を発見することで，プロダクトの派生関係のグラフを推定する手法 PRET を提案した．PRET はまず，共通ファイルの多さを基準として，プロダクトを頂点とする無向グラフを構築する．その後に，無向辺の両端にあるプロダクトに含まれる，共通ファイル以外のファイルの数によって，グラフの向きを決定する．

本論文では，ソフトウェアプロダクトの派生関係グラフを推定する新しい手法として，コルモゴロフ複雑性 [5] の考えに基づいた情報量の増分を利用する手法を提案する．コルモゴロフ複雑性の考えは，ある文字列を出力する最短のプログラムの長さを，その文字列の複雑さだとするものである．コルモゴロフ複雑性の値は一般には計算によって得ることはできないため，近似値として可逆圧縮アルゴリズムによる圧縮後のデータ量が使用されている． [2] [6] [9] [10] 提案手法では，あるプロダクトのソースコードを可逆圧縮したときのデータ量と，そのプロダクトに別のプロダクトを結合して同じアルゴリズムで可逆圧縮したときのデータ量を比較し，増分が小さいほどプロダクト間の距離が近いものと判定する．

*Yasuhiro Hayase, 筑波大学

†Tetsuya Kanda, 大阪大学

‡Takashi Ishio, 大阪大学

2 提案手法

提案手法は，単一のプロダクトを可逆圧縮したデータ量と，2 つのプロダクトを結合して可逆圧縮したデータ量を比較することで，派生関係のグラフを推定する．基本的なアイデアは 2 点ある．1 点目は先行研究 [4] と同じであり，直接の派生関係にあるプロダクトの組は，そうでないプロダクトの組に比べて，ソースコードの多くの部分を共有する傾向にあるという事実であり，このことは先行研究により確認されている．2 点目は，派生プロダクトにはより多くの情報が含まれる傾向にあるという仮定である．

以下に，提案手法の手順を記す．

入力 n 個のプロダクトのソースコード集合 $P = \{p_1, p_2, ...p_n\}$ (p_i は i 番目のプロダクトのソースコード)

出力 プロダクト派生関係の推定結果である有向グラフ $G = (P, E)$ (頂点集合 P は入力 P と同一)

手順 **1** 各 $p_i(i = 1..n)$ から，画像などのバイナリファイルを取り除く．

手順 **2** 各 $p_i(i = 1..n)$ について，含まれるファイルをパス名の順に tar でアーカイブ化し，可逆圧縮したファイルサイズ $C(p_i)$ を求める．

手順 **3** P に含まれるプロダクトの全ての 2 つ組に対して，プロダクトを結合したアーカイブ $p_i \cdot p_j$ を求める．結合では，プロダクト内でのソースファイルのパス名の順に基づいて，2 つのプロダクトのファイルを混ぜこんで配置する．配置の具体例を，表 1 に示す．パス名の類似したファイルは内容も類似していることが期待されるため，このような配置により可逆圧縮プログラムが類似部分を発見しやすくなると期待される．

手順 **4** 結合によって得られたアーカイブを可逆圧縮したファイルサイズ $C(p_i \cdot p_j)$ を求める．

手順 **5** プロダクト p_i に，プロダクト p_j を追加したときの増加情報量 $I(p_i, p_j) = C(p_i \cdot p_j) - C(p_i))$ と，下に示す仮定に基づいて，プロダクトの派生関係グラフを推定する．

仮定 **1** 派生プロダクトの情報量は，派生元プロダクトの情報量より多い

仮定 **2** 派生元プロダクトに派生プロダクトを結合したときの増加情報量は少ない

仮定 **3** 1 つのプロダクトの派生元プロダクトは，たかだか 1 つである

この仮定に基づき，派生関係グラフ G = (P, E) の有向辺集合 E を，下のように定義する．

$(p, q) \in E \iff C(p) < C(q) \wedge (\forall r \in P,\quad C(r) \geq C(q) \vee I(r, q) \geq I(p, q))$

これは，全てのプロダクト q について， q よりも圧縮後のデータ量が小さいプロダクトのうち， q を結合したときの増加情報量が最も小さいプロダクトから q に辺を引くことを意味する．

3 予備実験: 圧縮手法の評価

派生関係の推定に適した可逆圧縮アルゴリズムを評価するために，プロダクトのバージョン変化と，増加情報量についての予備的な調査を行なった．具体的な手順としては，2 つのバージョンのプロダクトを結合したときに，圧縮後のデータがどの

表 **1** プロダクト結合の具体例

プロダクト a	`a/1.c a/2.c a/4.c`
プロダクト b	`b/1.c b/2.c b/3.c`
$a \cdot b$	`a/1.c b/1.c a/2.c b/2.c b/3.c a/4.c`

ように変化するかを調査した．比較した圧縮ツールとアルゴリズムは，gzip (deflate (LZ77 [8] + Huffman coding [3]))，bzip2 (ブロックソート法 [1] + Huffman coding)，xz (LZMA (LZ77 の変種 + Range Encoder [7])) の 3 つであり，全てのツールに圧縮率を最大化する-9 オプションを指定した．対象としたプロダクトは PostgreSQL 8.0.0 から 8.0.26 までの 27 のリリースである．

まず，各リリースのソースコードを個別に tar アーカイブとし，各アルゴリズムで圧縮したときの圧縮率を，表 2 に示す．圧縮率の平均値は，$gzip > bzip2 > xz$ となり，新しい圧縮ツールほど良い圧縮率を示していた．一方で，平均に対する標準偏差の割合は bzip2 が最も大きく，圧縮率が不安定であることがわかる．

次に，2 つのリリース p, q に対して， p に q を加えたときの圧縮後データ量の相対増加率 $\frac{C(p \cdot q) - C(p)}{C(p)}$ を調査した．図 1 に，圧縮後データ量の相対増加率を表したヒートマップを示す．(a) から (c) はそれぞれ， gzip, bzip2, xz を使用した場合を表しており，配色の Gamma カーブ に 0.5 採用している．図の横軸は，基準となるプロダクトのリリース番号 (8.0.X の X 部分) であり，縦軸は追加したプロダクトのリリース番号である．セルの色が黒いほど相対増加率が小さく，白いほど相対増加率が高いことを表す．

バージョンの差が大きいほど含まれる情報の差が大きいと考えられるため，もし，対角線から上下に離れるほど単調に薄くなれば，情報量の増分をデータ量の差として正確に表現できたことになる．

gzip は，変化が少なく，この図からは相対増加率を読み取ることが難しい．bzip2 は，まだら模様になっていることから，バージョンの差がより大きいプロダクトを連結したときに，相対増加率が減っている場合があることがわかる．xz は，最も連続的に情報量が減少している．

図 では， gzip の詳細が観察しづらいため，配色の Gamma カーブを 0.1 変更し，変化を強調したヒートマップを図 1(d) に示す．この図から，バージョンの差が大きくなるとともに，ファイルサイズが連続的に増加する傾向があることが読みとれる．

以上の観察から，派生開発による情報量の増加を正確に表現できる順序は， xz, gzip，bzip2 であると期待される．

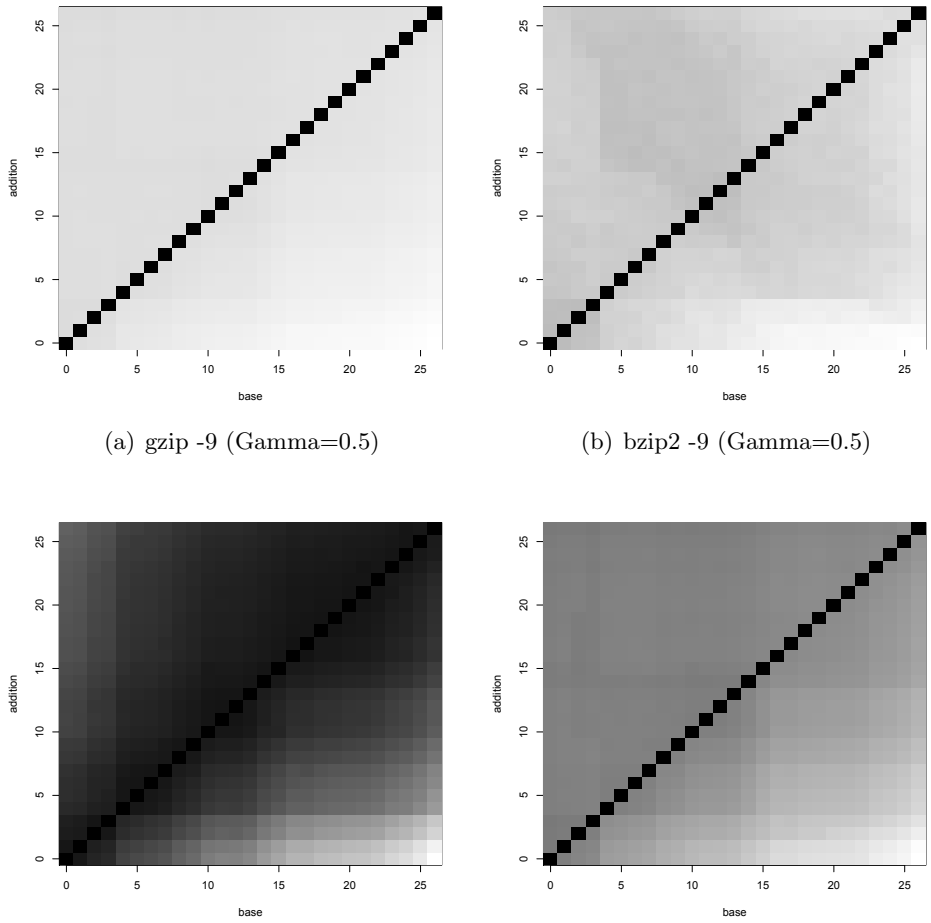
4 評価実験: 既存手法 PRET との比較

提案手法を用いて派生関係グラフを構築し，その精度を評価する．データ集合は，既存手法 PRET [4] の評価実験と同じであり，精度を PRET と比較する．データセットの概要を，下に記す．

dataset1 開発が分岐していないソフトウェアプロダクト集合 (PostgreSQL-major)
dataset2 組織内で開発が分岐したソフトウェアプロダクト集合 (PostgreSQL-8-ALL)
dataset3 ソフトウェア製品ファミリのうち新しいいくつかの製品しか残っていない場合 (PostgreSQL-latest)
dataset4 ソフトウェア製品ファミリのうち過去の製品の一部が欠落している場合 (PostgreSQL-annually)
dataset5 プロジェクトが 2 つに分岐した場合 (FFmpeg)
dataset6 プロジェクトが 3 つ以上に分岐し，複数回の併合が起こった場合 (*-BSD)

表 **2** 個々のリリースのソースコードに対する圧縮率

圧縮手法	圧縮率 (平均 ± 標準偏差)	標準偏差/平均
gzip -9	0.231 ± 0.000384	0.00166
bzip2 -9	0.182 ± 0.000368	0.00202
xz -9	0.163 ± 0.000233	0.00143



(a) gzip -9 (Gamma=0.5)　(b) bzip2 -9 (Gamma=0.5)

(c) xz -9 (Gamma=0.5)　(d) gzip -9 (Gamma=0.1)

図 1　圧縮後データ量の相対増加率

表 3 に，評価結果を示す.「正解」の列は，本来の派生グラフに含まれる辺の数である.「出力」の列はツールが出力した辺の数を表し，そのうち正解と一致した辺の数が，右の「適合」の列に記されている.「適合率」と「再現率」はそれぞれ,「出力」と「正解」の数を分母とし,「適合」の数を分子とした値であり，比較したツールの中で最も良かった値が太字で強調されている.「誤り」の列は，ツールが出力した辺が間違っていた理由の内訳を表している.「逆」は，辺の向きが逆であった場合,「skip」は，直前の派生元ではなく，グラフのより上流から辺が引かれてしまった場合である．skip の 2 は,「本来の派生元の派生元の派生元」から辺が引かれていたことを意味する.

提案手法が既存手法 PRET よりも良い精度であったのは，dataset1 全体と，dataset5 で gzip を用いた場合のみであった． dataset4 では，gzip と xz を用いた場合に既存手法と同等の精度となった．一方，dataset2,3,6 では，提案の精度は，既存手法より劣っていた.

4.1 考察

dataset1 で提案手法が既存手法の精度を上回り，また dataset 4 で同等の精度であったったことから，提案手法はプロダクトの大きな変化に強い可能性があると考えられる．プロダクトが大きく変更される場合，小さな変更に比べて，ソースファイルが分割あるいは結合されたり，内容が大幅に書換えられる頻度が高まるが，そのような場合でも提案手法は変化を連続的に評価できる可能性がある．

提案手法で bzip2 を用いた場合は，総じてグラフの精度が低く，特に 2 以上の skip による誤りが多かった．予備実験で相対増加率が不安定であったことと合わせて考えると，複数のデータに対して，コルモゴロフ複雑性を比較する場合に，bzip2 の出力を用いるのは不適切である可能性が高い．

dataset6 において提案手法の結果が悪かったのは，提案手法がプロダクトが併合されることを考慮していないためであると考えられる．プロダクトの併合に対処するためには，増加情報量の最も少ないプロダクト 1 つだけではなく，2 つ以上の類似したプロダクトから，派生関係の辺を引くように変更をする必要がある．

提案手法が既存手法より優れている点として，実行時のパラメータが少なく，プロダクトと圧縮アルゴリズムだけを指定すれば良い点がある．既存手法である PRET は，ファイルの類似性の閾値を設定する必要があり，この閾値によって結果が大きく変化する．提案手法にはそのような閾値はなく，そのため，今回の実験でも正解に対して結果が良くなるようなチューニングはしていない．

表 **3**　評価結果

データ	正解	手法	出力	適合	適合率	再現率	誤り			
							逆	skip		
								1	2	>3
dataset1	12	gz	12	12	**1.00**	**1.00**	0	0	0	0
		bz2	12	12	**1.00**	**1.00**	0	0	0	0
		xz	12	12	**1.00**	**1.00**	0	0	0	0
		PRET	12	11	0.917	0.917	1	0	NA	NA
dataset2	143	gz	143	105	0.734	0.734	3	22	5	3
		bz2	143	58	0.406	0.406	11	22	8	33
		xz	143	122	0.853	0.853	4	9	2	0
		PRET	143	128	**0.895**	**0.895**	8	1	NA	NA
dataset3	37	gz	37	24	0.649	0.649	0	6	1	0
		bz2	37	21	0.568	0.568	0	4	4	1
		xz	37	29	0.784	0.784	0	1	1	0
		PRET	37	30	**0.811**	**0.811**	0	0	NA	NA
dataset4	24	gz	24	20	**0.833**	**0.833**	0	0	0	0
		bz2	24	14	0.583	0.583	0	2	3	1
		xz	24	20	**0.833**	**0.833**	0	0	0	0
		PRET	24	20	**0.833**	**0.833**	0	0	NA	NA
dataset5	15	gz	15	14	**0.933**	**0.933**	0	0	0	0
		bz2	15	1	0.0667	0.0667	2	4	1	1
		xz	15	7	0.467	0.467	3	4	0	0
		PRET	15	11	0.733	0.733	2	1	NA	NA
dataset6	17	gz	15	10	0.667	0.588	0	0	0	0
		bz2	15	10	0.667	0.588	0	0	0	0
		xz	15	8	0.533	0.471	1	1	0	0
		PRET	15	12	**0.800**	**0.706**	3	0	NA	NA

誤りの内訳を観察すると，提案手法を PRET と比較した場合， skip 誤りが多く，辺が逆向きである誤りは同等かやや少ない傾向にあることがわかる．また，圧縮アルゴリズムごとに，提案手法の skip 誤り数にばらつきがあるが，その大小関係はデータセットによって異なることが見てとれる．このことから，複数の圧縮アルゴリズムによる比較結果を併用することで， skip 誤りを減らすことができる可能性がある．また，PRET に本手法を組み合わせることで，辺が逆向きである誤りを減らし，精度を高めることが出来るかもしれない．

5 関連研究

藤原ら [10] は，製品・サービスの価値評価法として，コルモゴロフ複雑性に基づいてインタビューやアンケート調査結果を分析する手法を提案した．この研究では，コルモゴロフ複雑性に基づく距離指標を定義し，それに基いて階層クラスタリングを行っている．また，コルモゴロフ複雑性の近似として gzip を使用している．

二村ら [9] は，プログラムの難読化が適切に行われているかを評価する指標に，コルモゴロフ複雑性を用いた．コルモゴロフ複雑性の近似には， bzip2 を使用している．

6 まとめ

本論文では，コルモゴロフ複雑性の近似として可逆圧縮アルゴリズムを用いることで，プロダクトの派生関係グラフを推定する手法を提案した．OSS を対象とした評価実験の結果，既存手法である PRET とは誤りの傾向が異なることと，可逆圧縮アルゴリズムによって推定結果に違いがあることが判明した．

今後の課題として，可逆圧縮アルゴリズムの組み合わせや，既存手法との組み合わせによって，より高い精度で派生関係を推定する手法を考案することが考えられる．

謝辞

本研究は科研費 25730036 および 科研費 25220003 の助成を受けたものです．

参考文献

[1] Burrows, M., Wheeler, D. J., Burrows, M., and Wheeler, D. J.: A block-sorting lossless data compression algorithm, Technical report, 1994.

[2] Cilibrasi, R. and Vitanyi, P.: Clustering by compression, *Information Theory, IEEE Transactions on*, Vol. 51,No. 4(2005), pp. 1523–1545.

[3] Huffman, D. A.: A Method for the Construction of Minimum-Redundancy Codes, *Proceedings of the Institute of Radio Engineers*, Vol. 40,No. 9(1952), pp. 1098–1101.

[4] Kanda, T., Ishio, T., and Inoue, K.: Extraction of Product Evolution Tree from Source Code of Product Variants, *Proceedings of the 17th International Software Product Line Conference*, SPLC '13, 2013, pp. 141–150.

[5] Kolmogorov, A. N.: On Tables of Random Numbers, *Theor. Comput. Sci.*, Vol. 207,No. 2(1998), pp. 387–395.

[6] Li, M., Chen, X., Li, X., Ma, B., and Vitanyi, P.: The similarity metric, *Information Theory, IEEE Transactions on*, Vol. 50,No. 12(2004), pp. 3250–3264.

[7] Martin, G. N. N.: Range encoding: an algorithm for removing redundancy from a digitized message, *Proceedings of the Video & Data Recording Conference*, Southampton, Jul. 1979.

[8] Ziv, J. and Lempel, A.: A Universal Algorithm for Sequential Data Compression, *IEEE Trans. Inf. Theor.*, Vol. 23,No. 3(2006), pp. 337–343.

[9] 二村阿美, 門田暁人, 玉田春昭, 神崎雄一郎, 中村匡秀, 松本健一: 命令の乱雑さに基づくプログラム理解性の評価, ソフトウェア工学の基礎 *XIX,* 日本ソフトウェア科学会 *FOSE2012*, December 2012, pp. 151–160.

[10] 藤原由希子, 五藤智久, 井口浩人: コルモゴロフ複雑性に基づく製品・サービスの価値評価, 情報科学技術フォーラム講演論文集, Vol. 8, No. 2, FIT(電子情報通信学会・情報処理学会) 運営委員会, August 2009, pp. 55–62.