

Gitリポジトリに対応した コードクローン系譜抽出 ツールの評価

オープンソースソフトウェア工学研究室 B4

60226192 NUR ATHIRAH RAMLI

コードクローン

- コードクローンには4つのタイプがある [1]
 - タイプ1(T1):
 - 空白とコメントの違いを除き, 構文的に同じコードフラグメント
 - タイプ2(T2):
 - タイプ1のクローンの違いに加え, 識別子名とリテラル値の違いを除き, 構文的に同一のコードフラグメント
 - タイプ3(T3):
 - タイプ1とタイプ2のクローンの違いに加え, ステートメントレベルで異なるが, 構文的に類似したコードフラグメント
 - ステートメントが追加/変更/削除されている
 - タイプ4(T4):
 - 構文的に異なるが, 同じ機能を持っているコードフラグメント
- 本研究はタイプ1〜3集中していく

[1] Wang, W., Li, G., Ma, B., Xia, X., & Jin, Z. (2020). Detecting Code Clones with Graph Neural Network and Flow-Augmented Abstract Syntax Tree. 2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER), 261-271.

クローン系譜とは

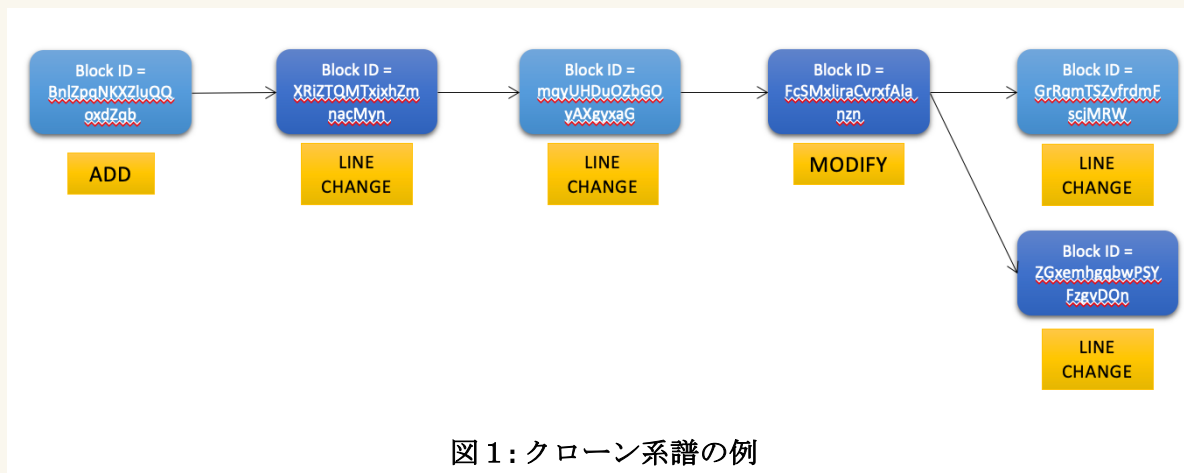


図 1: クローン系譜の例

- クローン系譜は, コードクローンの変更の履歴のことである [2]
- クローン系譜から, クローンの色々な情報 (いつ発生したか, コードクローンがどのように変更されたかなど) が得られる [3]

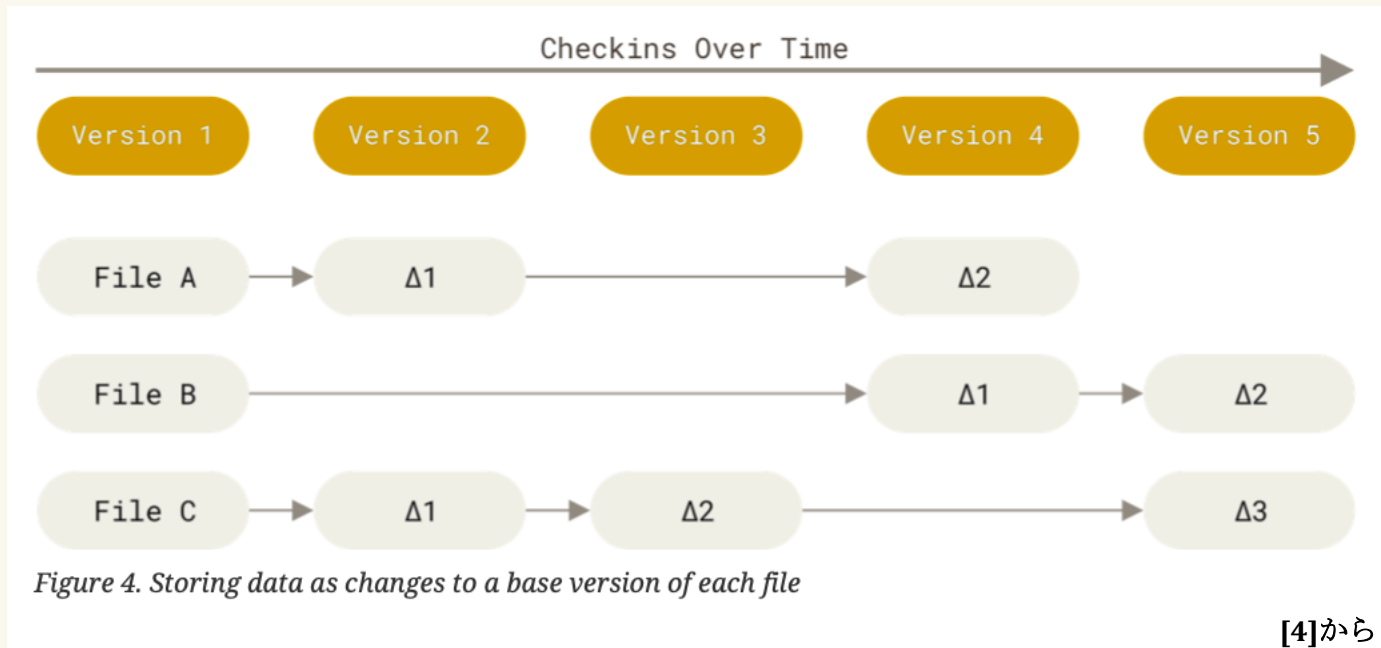
[2] M. Zibran. Management Aspects of Software Clone Detection and Analysis. 2014.

[3] T. Moriwaki, K. Hotta, H. Igaki, K. Inoue, S. Kusumoto. Suggestion of the source code reuse analysis technique in multiple projects based on the clone genealogy. Technical report of IEICE, vol. 114 number 510, pages 61-66, 2015.

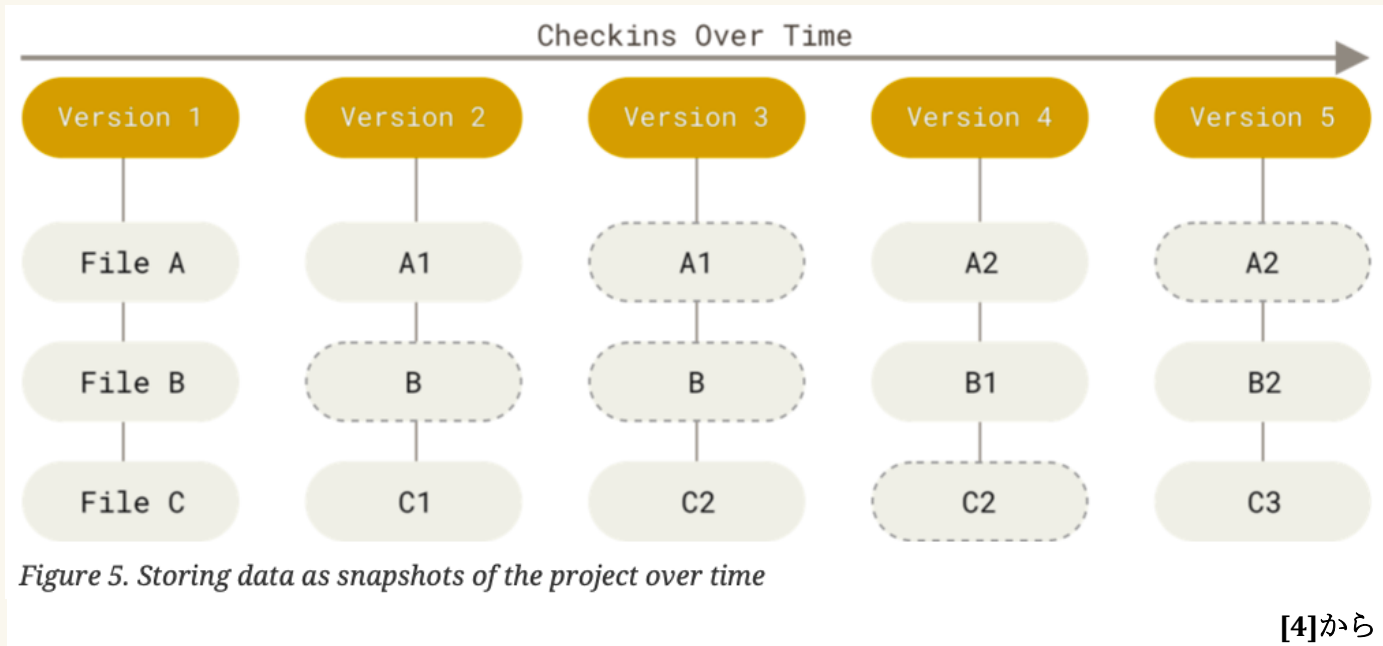
Git

- バージョン管理システムは2つの種類がある：
 - 集中型バージョン管理システム
 - Centralized Version Control System
 - CVS, Subversionなど
 - 分散型バージョン管理システム
 - Distributed Version Control System
 - Git, GNU ArchやMercurialなど
- GitはCVSとSVNのデータ格納方法が異なる[4]

[4] S. Chacon and B. Straub. Pro Git. Apress, 2 edition, 2014. ISBN 978-1484200773.



- CVSやSVNなどは図4の通りにデータを格納する
- 各ファイルの変更がリポジトリに加える



- 図5はGitのデータ格納方法を表す
- スナップショットとしてデータを格納

背景

- コードをコピーする時に, バグ伝搬を引き起こす可能性がある
- コードクローンを検出方法の一つはクローン系譜を使用する [5]
- 既存のクローン系譜抽出ツールはCVSやSVNからデータを受け取る [2]

[5] M. Kim, V. Sazawal, D. Notkin, and G. Murphy. An empirical study of code clone genealogies. SIGSOFT Softw. Eng. Notes 30, 5 (September 2005), 187–196.

背景

- 現代, 多数のソフトウェアはGitをリポジトリとして使用している [6]
- GitはCVSとSVNのデータ格納方法が異なり[4], 既存のクローン系譜抽出ツールは対応していない可能性がある
- そのため, Gitに対応した新たなクローン系譜抽出ツールが作成された

[6] I. Tepavac, K. Valjevac, S. Kliba, M. Mijač. Version Control Systems, Tools and Best Practices: Case Git. CASE 27, Zagreb, 2015.

目的

- Gitに対応したクローン系譜抽出ツールを評価する
- Gitに対応したツールを分析する

関連研究

- CGE [5]
 - コードクローンの進化を理解するために, クローン系譜の概念を提案した
- gCad [7][8]
 - 最新のクローン系譜抽出ツール

[7] R. K. Saha, C. K. Roy and K. A. Schneider. An automatic framework for extracting and classifying near-miss clone genealogies. 2011 27th IEEE International Conference on Software Maintenance (ICSM), Williamsburg, VI, 2011, pp. 293-302.

[8] R. K. Saha, C. K. Roy and K. A. Schneider. gCad: A Near-Miss Clone Genealogy Extractor to Support Clone Evolution Analysis. 2013 IEEE International Conference on Software Maintenance, Eindhoven, 2013, pp. 488-491.

アプローチ

1. 複数のプロジェクトを用いて, 正しく接続されているコード片ペアの割合を計算する
2. 計算結果を用いて, 分析を実施する
 - ランダムサンプリングも行う

Swiftプロジェクトのデータを用いて

- 接続済コード片/全体のコード片を計算
 - 接続済コード片 : 48,721
 - 全体のコード片 : 164,558

$$\frac{48721}{164558} = 0.296$$

- 100件のコード片をランダムで選択し、正しく接続されているかどうかを確認する
 - 系譜の図が作成したが、分析がまだやっている中

今後の予定

- 分析を行う
- 論文執筆の引き続き