

Gitに対応した欠陥混入 クローン特定手法の構築

野口 耕二郎

背景

- 品質に悪影響を与えるコードクローンを優先して除去，管理する研究が盛んになってきている
- 品質に悪影響を与えるコードクローンの1つとして欠陥混入クローンが挙げられる
- 分析や検出手法の構築などのために欠陥混入クローン特定手法が必要

欠陥混入クローンに関する研究

- コードクローン作成時に10%の割合で欠陥が複製されることを明らかにした研究[1]
- より最近変更されたコードクローンに欠陥が混入しやすいことを明らかにした研究[2]
- 欠陥混入クローンの33%から欠陥が拡散することを明らかにした研究[3]

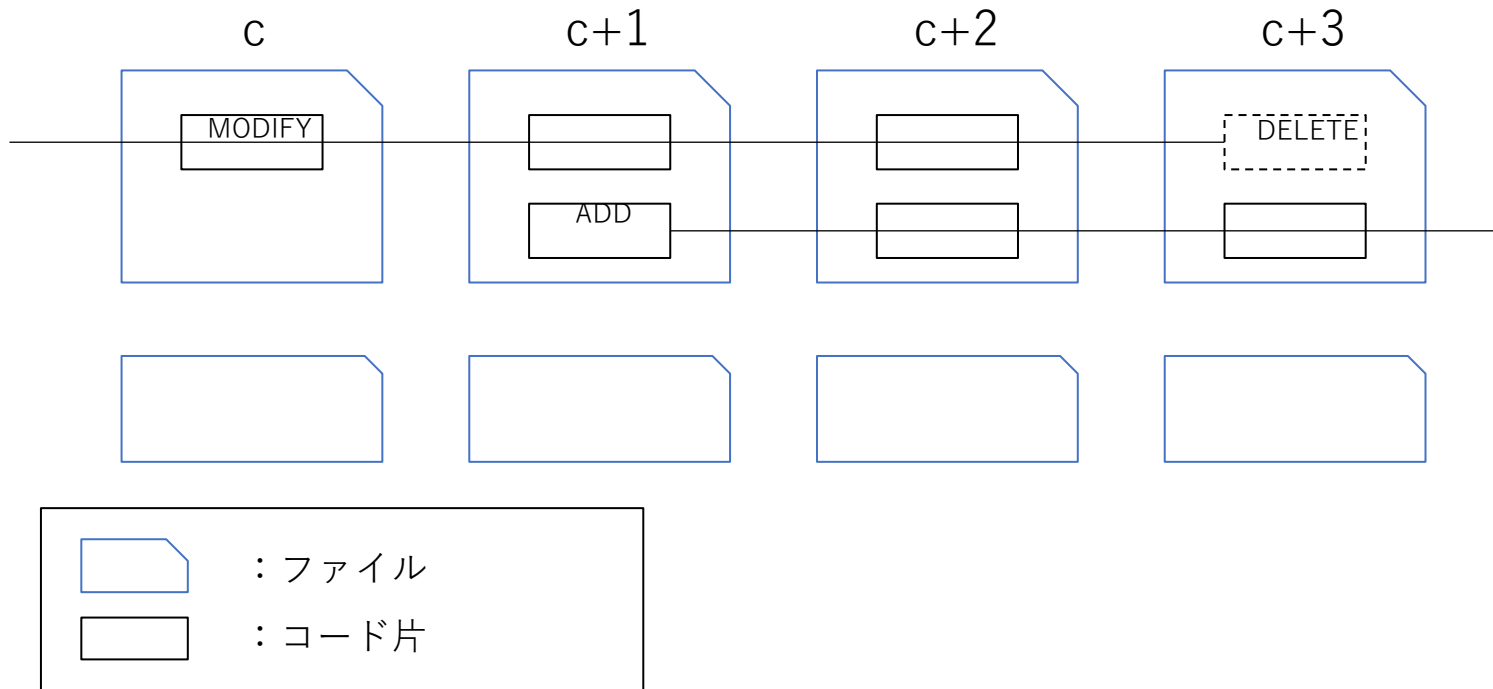
[1] J.F. Islam, M. Mondal, and C.K. Roy, "Bug replication in code clones: An empirical study," 2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER), vol.1, pp.68–78, 2016.

[2] M.Mondal, C.K.Roy,andK.A.Schneider,"Identifyingcodecloneshavinghighpossibilities of containing bugs," Proceedings of the 25th International Conference on Program Comprehension, p.99–109, ICPC '17

[3] M. Mondal, C.K. Roy, and K.A. Schneider, "Bug propagation through code cloning: An empirical study," 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp.227–237, Sep. 2017.

既存の欠陥混入クローン特定技術

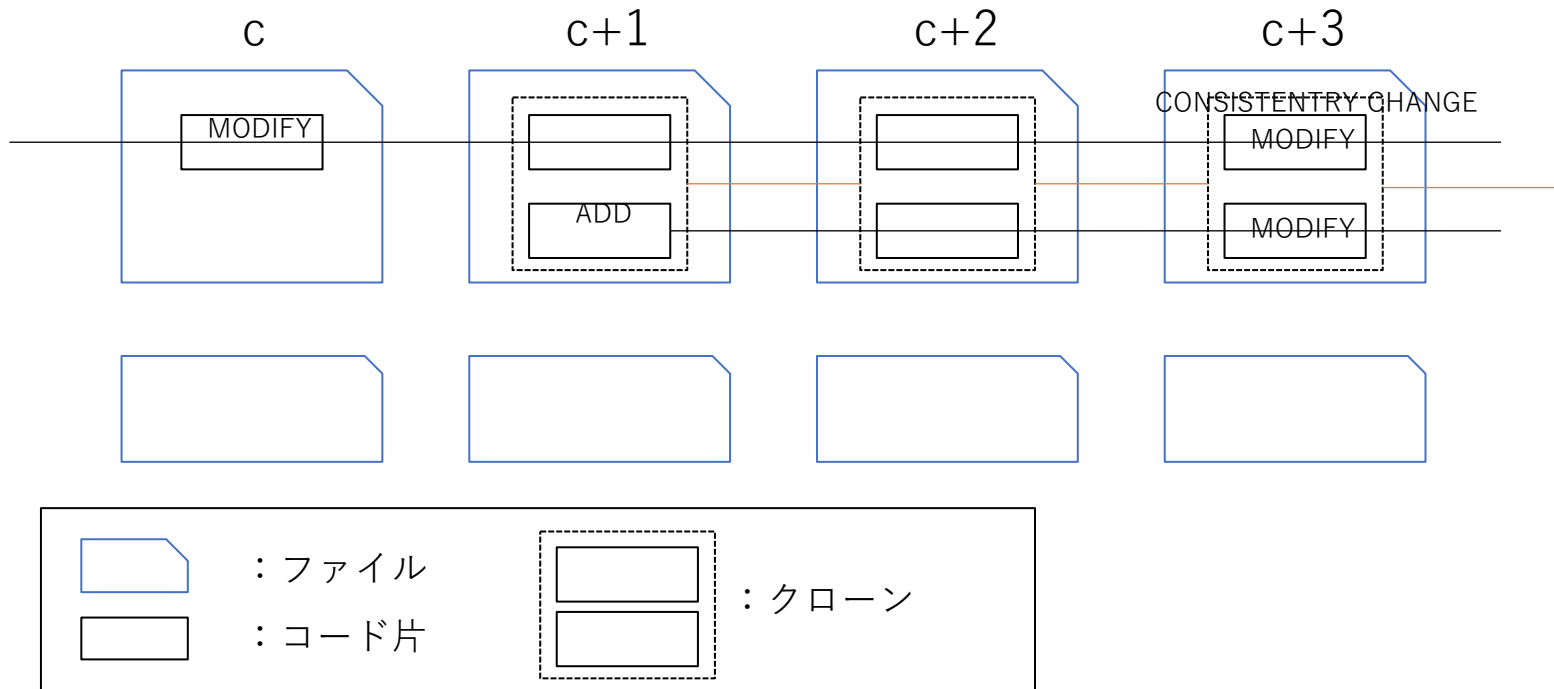
- コード片系譜抽出
 - リビジョン前後で同一のコード片の特定を未来に向かって繰り返し，コード片の変更の歴史を抽出
 - クローン系譜の抽出に必要



既存の欠陥混入クローン特定技術

- クローン系譜抽出

- クローンペアを構成するコード片の系譜を紐付けることで抽出
- クローンに加えられた操作を知るために必要

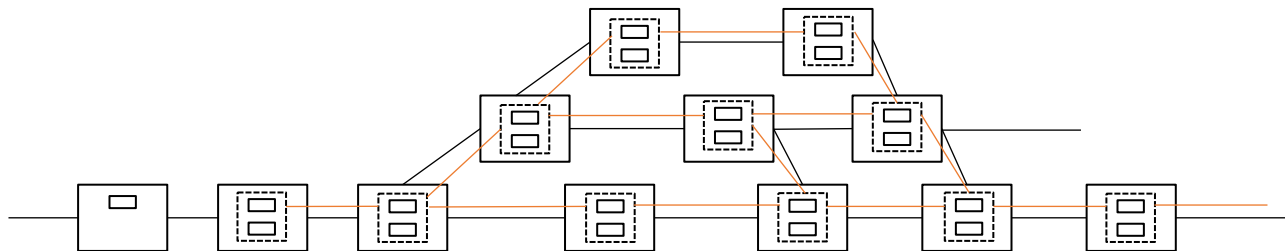


既存の欠陥混入クローン特定技術

- 欠陥混入クローン特定
 - 欠陥修正を含むクローンを特定する
 - 系譜をたどり，欠陥混入リビジョンでクローンが作成されたことを確認
 - 欠陥混入から修正までの間に変更が加えられていないことを確認

既存の系譜抽出手法の問題

- SVNなどの集中型バージョン管理システムからの系譜抽出を想定したもの
- 現在主流のGitなどの分散バージョン管理システムには対応できない
 - ブランチやマージにより変更履歴が複雑になる
 - 系譜抽出のための計算コストが膨大になる
(SVN: $O(n^3)$ → Git: $O(n^4)$)
 - 正確に系譜抽出できない



DVCSから欠陥混入クローンを収集するためには、変更履歴の複雑さを解消する手法の構築が必要

目的とアプローチ

- 目的
 - Gitに対応した欠陥混入クローン特定手法の構築する
 - 計算コストと欠陥混入クローンの特定精度を実証実験で明らかにする
- アプローチ：Gitにおける変更履歴の複雑さの解消
 - コード片抽出
 - 変更が加えられたコード片のみ抽出する
 - クローン特定
 - 欠陥混入，修正リビジョンのクローンのみを特定する
 - 欠陥混入リビジョンで作成されたクローンを特定
 - 欠陥修正リビジョンで修正されたクローンを特定

既存手法と提案手法の違い

	既存手法	提案手法
コード片抽出	CTAGS	GumTree
クローン検出	NiCad	NiCad
系譜抽出	する	しない
欠陥混入クローン 特定に用いる情報	コード片, クローン系譜	変更コードクローン 欠陥混入, 修正コード片ペア

変更コード片登録

- クローンペアの各クローン片と一致する変更コード片を探し，登録する
- クローンに加えられた変更がわかるようになる

欠陥混入，修正コード片ペアの特定

- 欠陥混入，修正時に変更が加えられたコード片を特定する
- 欠陥混入時のコード片が欠陥修正時にどの箇所に存在するかを確かめるためにGumTree[4]を用いる

欠陥混入クローンの特定

- 変更コードクローンと欠陥混入，修正コード片ペアの情報を用いる
- 欠陥混入時にクローン片が作成され，欠陥修正時に変更が加えられたクローンを特定
- クローンに登録されている変更コード片が欠陥混入，修正コード片ペアに含まれるかを確認

計算量の比較

- 既存手法
 - コード片系譜抽出をGitに対応させた場合： $O(n^4)$
- 提案手法
 - コード片登録： $O(n^3)$
 - 欠陥混入，修正コード片ペア特定： $O(n^3)$
- 計算量は既存手法 > 提案手法であり，削減することができた

特定結果

プロジェクト	言語	期間	リビジョン数	欠陥混入rev. (all)	欠陥混入rev. (クローン)	buggy clone rev / buggy rev
kafka	java	2015/1/1- 2020/1/1	5844	974	73	7.5%
ExoPlayer	java	2015/1/1- 2020/1/1	7663	2655	84	3.2%
druid	java	2015/1/1- 2020/1/1	5208	2241	314	14.0%
skywalking	java	2015/1/1- 2020/1/1	5390	1737	160	9.2%
guava	java	2015/1/1- 2020/1/1	2578	537	38	7.1%
ghidra	java	2015/1/1- 2020/1/1	2054	363	26	7.2%
dbeaver	java	2015/1/1- 2020/1/1	12774	3967	106	2.7%

提案手法の有用性評価

- 既存手法では2種の欠陥混入クローンパターンが定義されており，特定することができる
 - 欠陥複製パターン
 - 複製後混入パターン
- 既存手法では特定できないGit特有の欠陥混入クローンパターンを特定できることを示す

Git特有の欠陥混入クローンパターン

1. masterブランチに存在するコード片からクローン作成
2. 修正用に作ったブランチで自分で作成したコード片からクローン作成
3. 修正中に更新されたmasterブランチで作成されたコード片を取り込んでからクローン作成

各パターン取得方法

- パターン1

- クローン片の片方がADDの欠陥混入クローンを取得するようにする

- パターン2

- masterブランチのある欠陥混入コミットでクローン片の両方がADD
- 開発用ブランチで同じコミットで作成されたか、違うコミットで作成されたか

- パターン3

- 考え中

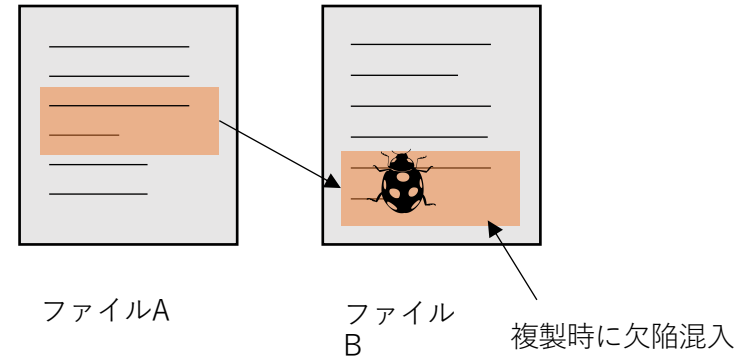
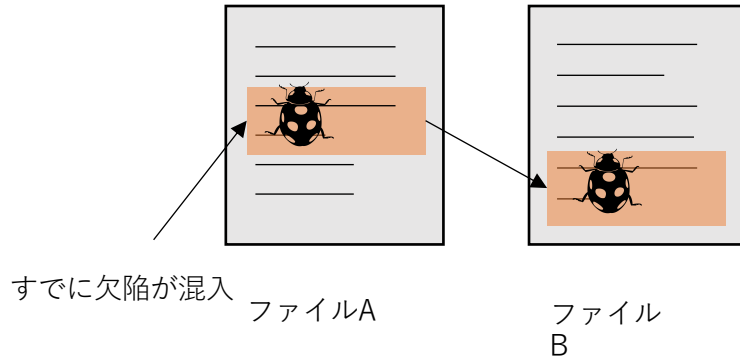
今後の予定

- 欠陥混入クローンをパターン別に分けて，提案手法でgit特有パターンを特定できることを示す
- 修論を仕上げる

補足資料

欠陥混入クローンの種類

- 欠陥複製（左）
 - 欠陥を含むコード片を複製
- Context Bug（右）
 - クローン片の作成時に欠陥が混入



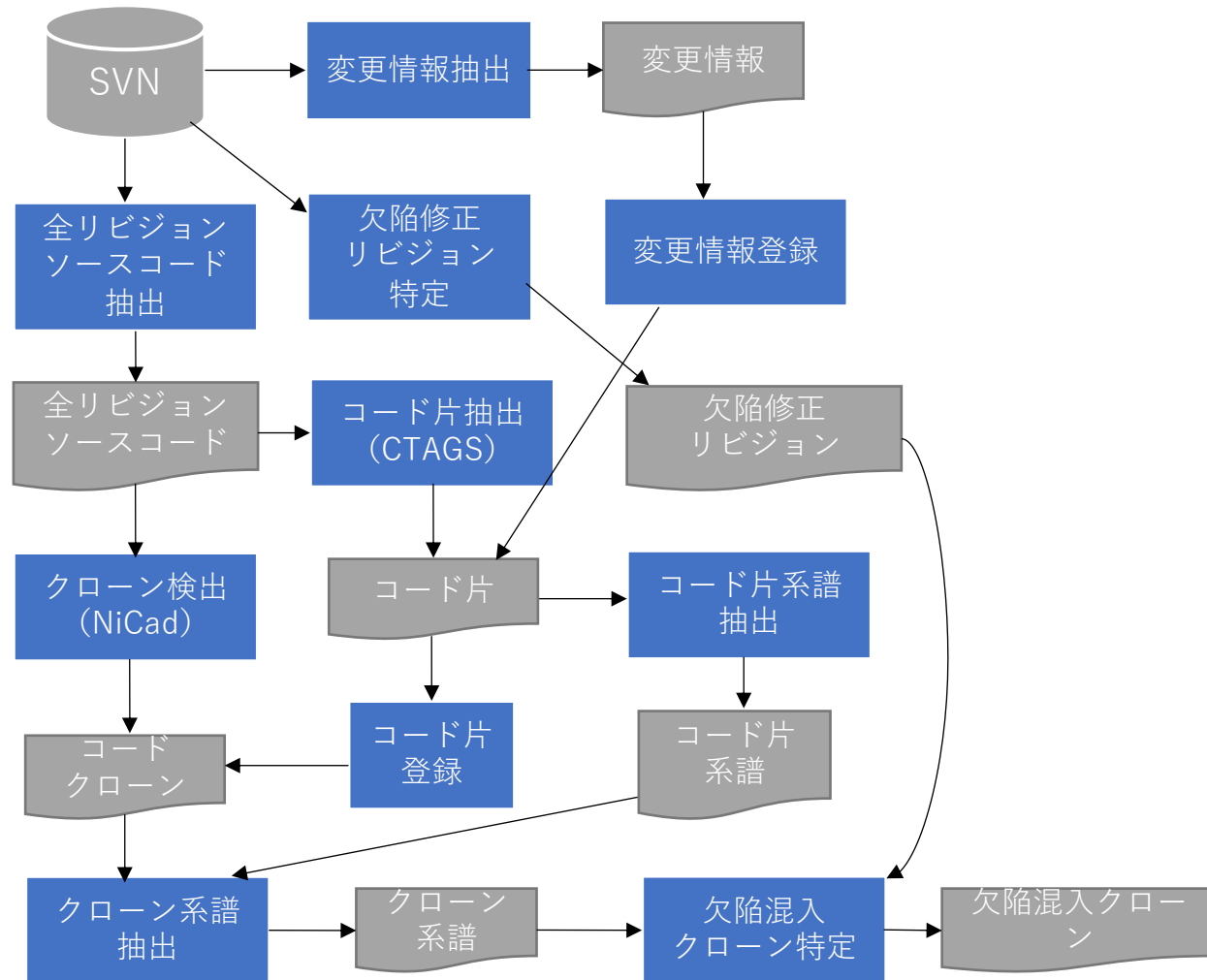
Jiangらの手法

1. クローン検出 (Deckard[])
2. ASTに基づいて一貫性がないクローンを取得
3. 一貫性がないクローンとそれ以外を分ける
4. 開発者が検査して欠陥混入クローンを報告

Xieらの手法

1. J-RexとC-Rexを使ってSVNリポジトリからソースコードとバグ修正リビジョンを抜き出す
2. テストコードの除去
3. クローン検出 (NiCad[])
4. クローン系譜抽出
5. バグ修正を含むクローン系譜とそうでない系譜を分ける

Mondalらの手法の処理概要図



全リビジョンの抽出

- コード片, クローンの抽出のために全リビジョンのソースコードを抜き出す
- `svn export` コマンドを対象期間のリビジョン全てに対して実行する

欠陥修正リビジョンの抽出

- コミットメッセージに欠陥修正を示すキーワードを含むリビジョンを欠陥修正リビジョンとする
 - キーワード：bug, fix, errorなど
- 欠陥修正リビジョンで加えられた変更は欠陥修正変更とする

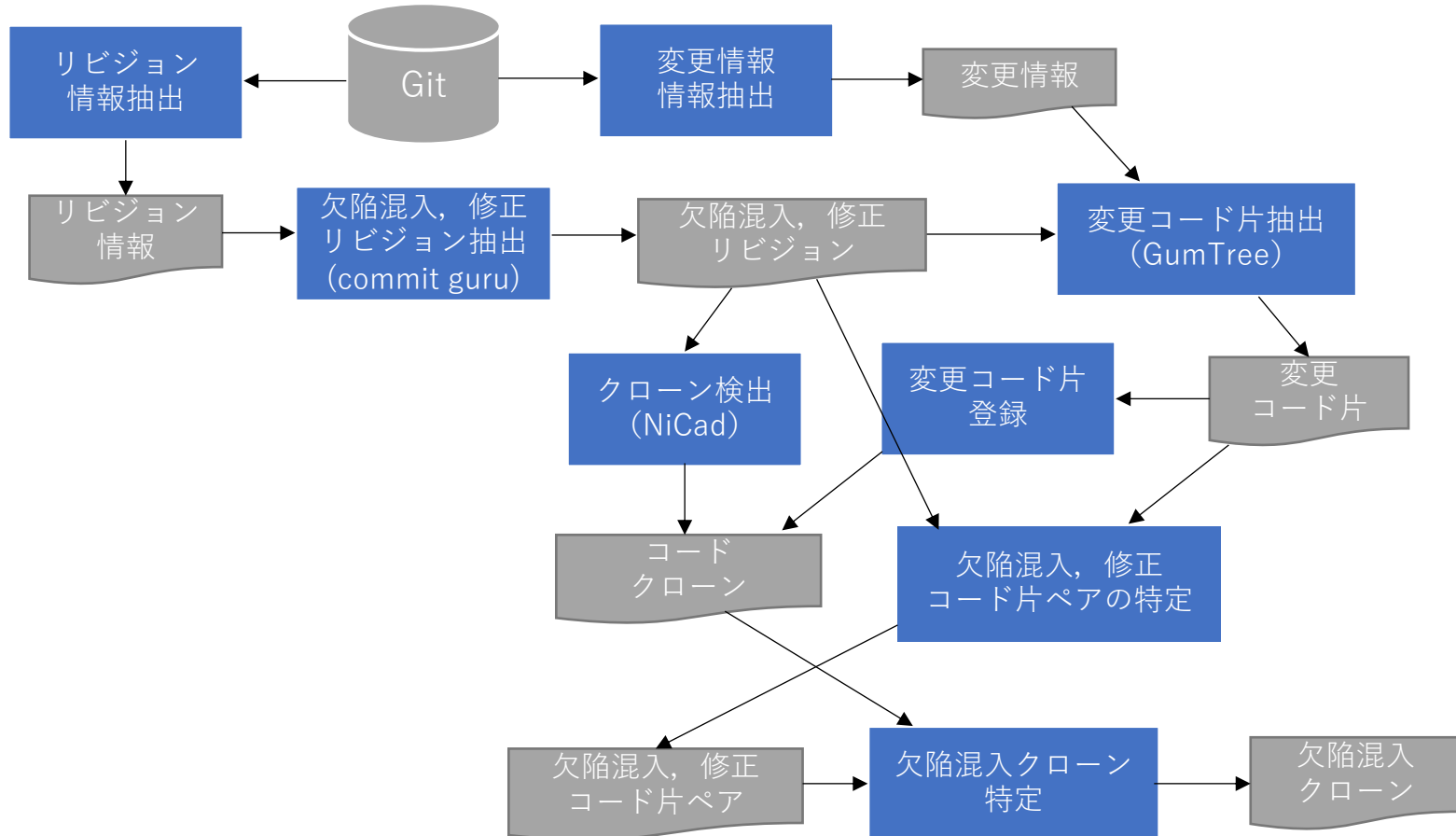
コード片系譜抽出

- Lozanoらの研究[]で用いられている手法を参考にしている
- コミット前後でコード片の内容の類似度がもっとも大きいものを同一のコード片とする
- リビジョンを進めて同様に同一のコード片をつなげていくことで系譜を抽出している

クローン系譜抽出

- クローンペアを構成するコード片はコード片系譜に接続されている
- コード片系譜とクローンペアを紐づけることでクローン系譜を抽出できる

Git対応特定処理の概要図



Git対応時に実装が困難となる処理

- コード片系譜抽出
 - Gitでは変更履歴が途中で枝分かれしたり，連結したりする
 - 複雑な変更履歴の中で正確にコード片系譜を抽出するのは困難
 - 処理が複雑になるため計算時間が膨大になる
- クローン系譜抽出
 - 変更コード片以外のコード片の登録が困難
 - 系譜の中で親子関係を正確に構築するのが困難