リファクタリングを目的としたコードクローン分析ツール Aries

肥後芳樹[†] 神谷年洋[††]
楠本真二[†] 井上克郎[†]

Aries: Refactoring Supporting Tool based on Code Clone Analysis

Yoshiki Higo,[†] Toshihiro Kamiya,[††] Shinji Kusumoto[†]
and Katsuro Inoue[†]

1. はじめに

近年，コードクローンがソフトウェア保守を困難にしている 1 つの要因といわれている．コードクローンとはソースコード中に存在する同一，または類似したコード片のことである．コードクローンが生成される原因としてはさまざまな理由が考えれれるが，その最も大きな原因の 1 つとしてコピーアンドペーストによる修正，拡張作業が挙げられる．あるコード片にバグが含まれていた場合，そのコード片のコードクローン全てに対して修正の是非を考慮する必要がある．このような作業は，特に大規模ソフトウェアでは非常に手間のかかる作業である．

組込み開発のソフトウェアのソースコードは近年大規模になっており，また，実行時のパフォーマンスを稼ぐためにループを意図的に展開したコードクローンが作り込まれる場合があるなど，コードクローンの発生は避けられない問題となっている．本稿では，我々が開発してきたコードクローン検出ツール CCFinder とリファクタリングを目的としたコードクローン分析ツール Aries を紹介する．

2. コードクローン検出ツール: CCFinder

CCFinder[2] は与えられたソフトウェアのソースコード中に存在するコードクローンを検出し，その位置を出力する．CCFinder はコードクローンをトークンの列として検出する．そのため，検出したコードクローンは必ずしも集約に適した単位とはなっていない．

† 大阪大学大学院情報科学研究科
Graduate Scholl of Information and Science Technology, Osaka University
†† 産業技術総合研究所情報技術研究部門
Infomation Technology Research Institute, Advanced Industrial Science and Technology

3. コードクローン分析ツール: Aries

Aries[3] は GUI ベースのコードクローン分析ツールであり，リファクタリング支援を目的として開発されている．Aries は内部で CCFinder を用いている．

3.1 集約に適したコードクローンの検出

Aries は CCFinder の検出したコードクローンから， 構造的なまとまりを持った部分を**集約に適したコードクローン**として抽出する．図 1 はその例を示している．図 1 では，A と B の 2 つのコード片が示されている．A と B それぞれの灰色の部分は，その部分が A と B の間の最大長のコードクローンであることを示している．コード片 A ではいくつかのデータがリスト構造の先頭から順に連続して格納されている．一方コード片 B では，リスト構造の後方から順に連続してデータが格納されている．これら 2 つのコード片には，リスト構造を扱う共通のロジック（for 文）が含まれているが，コード片の最初と最後には，偶然クローンとなった部分（代入文）も含まれてしまっている．集約を目的とした場合，灰色の部分全体よりも for 文のみをコードクローンとして抽出する方が望ましい．Aries ではこのような場合，灰色で示されたコードクローンから構造的なまとまりを持った部分，つまり for 文の部分のみを抽出する．

3.2 メトリクスを用いた特徴づけ

Aries は集約に適したコードクローンがどのように

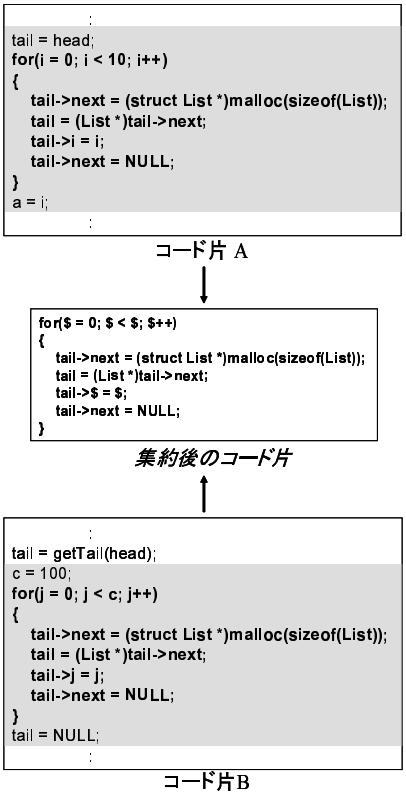

図 1 コードクローン集約の例
Fig. 1 Example of merging two code fragments

除去できるのかを予測するために，メトリクスを用いてそれらを特徴づける．そして，それらが既存のどのリファクタリングパターン[1)] に適用可能であるかを予測する．この節では Aries が用いているメトリクスのうちのいくつかを簡単に紹介する．

NRV(S):クローンセット S ☆に含まれるコード片が，どの程度その外部で定義された変数に対して**参照**を行っているかを表す．このメトリクスはコード片とその周囲との結合度を表しており，値が低いほど集約が容易である．

NSV(S):$NRV(S)$ と同様，コード片とその周囲との結合度を表す．$NRV(S)$ との違いは，参照ではなく**代入**をカウントしている点である．

DCH(S):クローンセット S に含まれるコード片がどの程度クラス階層中に広がっているかを表す．例えば，全てのコード片がある一つのクラス内に含まれる場合は 0，あるクラスとその直接の子クラス内に含まれる場合は 1 と，コード片が広い範囲に存在するほど，この値も大きくなる．

☆ "クローンセット"とは，互いに類似したコード片の集合を現す．

3.3 絞り込み例: Extract Method

これまでに述べたことから，リファクタリングパターン "Extract Method" を行なう際の条件として，は例えば以下のものが上げられる．

(条件 1) 対象となる単位は文単位,

(条件 2) $DCH(S)$ の値が 0,

(条件 3) $NSV(S)$ の値が 1 以下,

"Extract Method" とはメソッド内のコード片に対して適用されるので，(条件 1) が必要である．また，全てのコードクローンが同一のクラス内に存在する場合は容易に集約が可能であるので，(条件 2) を設定している．コードクローンの内部において，外部定義変数に対して代入を行なっている場合は，その変数を引数として与え，返り値として返し，メソッドの呼び出し元に反映させなければならない．このような変数が複数あった場合は新たなデータクラスを定義し，そのオブジェクトを介して値を受け渡す必要がある．しかし，もしこのような変数が 1 つの場合は単に return 文を用いて返すだけで良く，容易に集約を行なうことができるので，(条件 3) を考慮している．

この条件を用いることによって，ユーザは "Extract Method" が容易に適用可能であるクローンセットを得ることができる．しかし Aries はその集約の有効性までは考慮していない．Aries が提示したクローンセットを集約するか否かはユーザが判断する必要がある．

4. ま と め

本稿では，コードクローン検出ツール CCFinder およびコードクローン分析ツール Aries について述べた．これらを用いてソースコード中に含まれるコードクローンを検出し，対話的に分析をすることで，効率的にコードクローンを集約することが可能である．

謝辞 本研究は，文部科学省 科学研究費補助金 特別研究員奨励費 (課題番号：16・8351) の助成を得た．

参 考 文 献

1) M. Fowler, *Refactoring: improving the design of existing code*, Addison Wesley, 1999.
2) T. Kamiya, S. Kusumoto, and K. Inoue, *CCFinder: A multi-linguistic token-based code clone detection system for large scale source code* IEEE Transactions on Software Engineering, vol.28, no.7, pp.654-670, Jul. 2002.
3) 肥後芳樹, 神谷年洋, 楠本真二, 井上克郎 "コードクローンを対象としたリファクタリング支援環境", 電子情報通信学会論文誌, Vol.88-D-I, No.2, pp.1868-195, Feb. 2005.