【別添2】

CCFinderNexGen の開発

1. 背景

本テーマでは、拙作コードクローン（重複コード、類似コード）検出ツールである CCFinder（以下旧 CCFinder）を機能拡張し、より広範な目的に利用できるようにします。

コードクローンとは、ソースコード中の全く同一のコードの断片、または、類似したコード断片のことであり、開発者がソースコードをコピー＆ペーストするなどの原因により、ソースコード中に作り込まれます。コードクローンが存在すると、あるコード断片にバグが見つかったとき、そのコードクローンが 100 個あれば、その 100 個のコード断片をすべて探し出し修正する必要があります。特に、大規模なソースコードにおいては、手作業でコード断片を見つけ出すことは困難であり、ソフトウェアの保守を困難にする原因だと言われています。

旧 CCFinder は、当初から大規模なソースコードに適用することが想定されて開発され、数百万行規模のソースコードに対する安定した運用実績を持っています。産業界においては、数々のメーカーでリファクタリングのため、あるいは、ソースコードの品質の調査のために使われています。学界においては、旧 CCFinder 自身が研究として発表されており、さらに、配布した旧 CCFinder によって実験を行った研究がいくつも発表されています。

旧 CCFinder は 2000 年に配布が開始され、現在までに数百の組織に配布されています。利用が広がるにつれ、当初の開発においては想定していなかった用途にも適用され始めました。たとえば、旧 CCFinder が対応していないプログラミング言語で記述されたソースコードからコードクローンを検出したい、あるいは、CVS などのバージョン管理ツールと組み合わせて使いたい、といった要求が寄せられました。しかし、旧 CCFinder はその設計上の制約から、それらの要求に応えることはできませんでした。

2. 目的

本テーマでは、旧 CCFinder の制約を解消してこれらの要求を実現するとともに、これらの用途における実行性能を改善するため、新たな機能を実装し、同時に、必要とされる設計の変更を行います。

旧 CCFinder の用途が限定されている原因として、ハードコードされたソースコード変形処理と内部データの扱いがあげられます。旧 CCFinder は、プログラミング言語の文法を考慮し、ソースコードに特化した処理を行うことで、コードクローンの検出精度を向上させています。ソースコードの断片をコピー＆ペーストした後に変数名を変更したり、ソースコードの意味を変えないような些細な変更がなされた場合でもコードクローンを検出できるようにするため、同一文字列を検出するアルゴリズムを実行する前に、前処理としてソースコードの変形処理を行います。変形処理は、ソースコード中のパターンと、そのパターンが見つかった場合にどのようにトークンを足したり削ったりするかを定めたルールに基づいて、ソースコードを変形します。変形処理のためのルーチンは特殊なものであり、通常の正規表現やパーサ（構文解析器）を用いずにハードコードされているため（理由は後述）、利用者がコードクローン検出処理のカスタマイズを行うことができません。具体的には、以下のよ

うな問題があります。

（問題点１）変形ルールをカスタマイズできない：利用者が、新しいプログラミング言語や方言をサポートするようにカスタマイズすることは不可能です。

（問題点２）異なる文法が混じっているようなソースコードをサポートできない：たとえば、ソースコードに埋め込まれたSQL文から重複を検出したり、JavaDocなどの構造を持ったコメント部分から類似文字列を検出したりといった作業ができません。

これら以外にも、主に内部データの処理方法の実装に起因する問題として、

（問題点３）微妙な検出条件を設定できない：たとえば、特定のソースファイルと類似する部分は除外してコードクローンを検出するとか、指定した識別子は他の識別子とマッチしないようする、といったことができません。

（問題点４）インクリメンタルな処理ができない：ソースファイルを一つ修正すると、すべてのソースファイルについて解析し直す必要があり、頻繁なソースコードの修正を前提とする応用で問題になっています。

３. 開発の内容

本テーマでは、前述の問題点 1 から 4 を解消すべく、旧 CCFinder を完全に再設計・際実装しました。開発されたツール CCFinderX の構成を図 1 に示します。

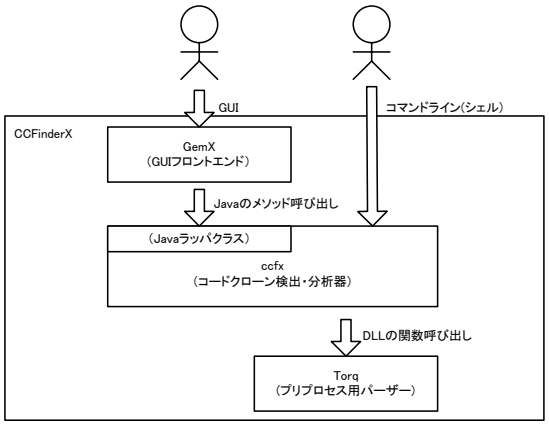

図 1 CCFinderX を構成する 3 つのコンポーネント

図 1 下部にある Torq は、前述の問題点１、２への対策として開発されたコンポーネントです。「拡張された正規表現」の表記法およびマッチングルーチンであり、旧 CCFinder のプリプロセス処理に相当する部分を、スクリプトによって行うことを可能にします（機能の詳細については紙面の都合上省略いたします）。これにより、CCFinderX は、C/C++, Java, COBOL, Visual Basic, C#といったプログラミング言語に対応することが可能になっています。図 1 中央の ccfx は（torq と併せて）、旧 CCFinder に相当するコードクローン検出ツールです。図 1 上部の GemX については後述します。

また、前述の問題点３,４への対策として、コードクローン検出処理のデータの流れを再設計しました。具体的には、内部処理を大きく３つに分割し、一部の内部データを一時データではなく、永続データとします(図 2 参照)。これにより、(1)入力となるソースコードの一部が修正された場合に、その変更を検出して影響範囲を特定することで、インクリメンタルなコードクローン検出処理を行う、(2)ソースコードをあらかじめ読み込んでおき、いろいろな

変形ルールを与えてコードクローンを検出する、(3)検出しておいたコードクローンに対して、さまざまなクエリに基づいてフィルタリングを行う(たとえば、特定のプログラムパターンに従うコードクローンだけを出力せよ、など)ことができます。

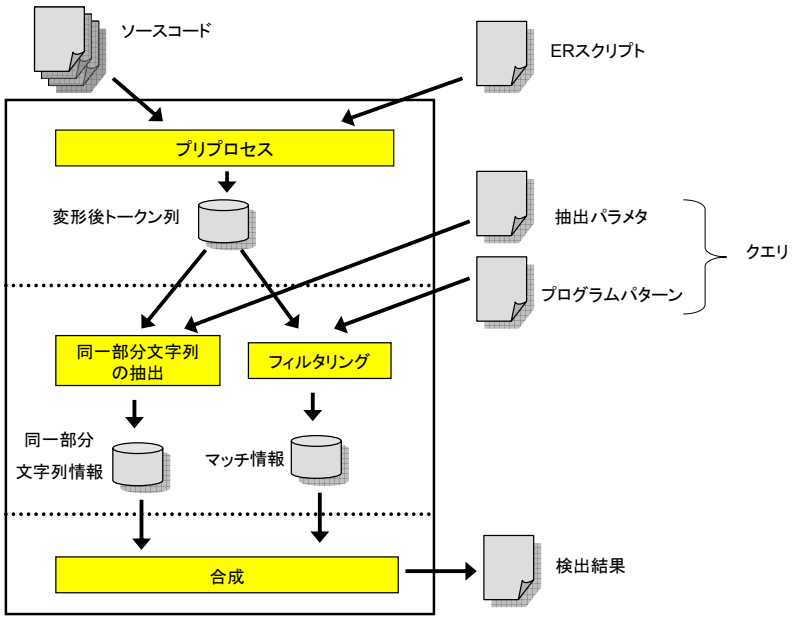

図 2 CCFinderX の(うち ccfx, torq の部分のについて)の処理の流れ

図 1 の上部にある GemX は、CCFinderX の機能を GUI を通じて対話的に利用するためのフロントエンドです。応募時点では、旧 CCFinder の GUI フロントエンドを流用する予定でしたが、開発が進むにつれて、旧 CCFinder との機能的な相違が大きいことがわかり、新規に開発することとしました。GemX は、ファイルリスト、クローンセットリスト、散布図、ソーステキストといったビューを備え、コードクローンを検出するだけではなく、散布図やメトリクスによるフィルタリングなど、さまざまな分析することを可能にしています(図 3 参照)。

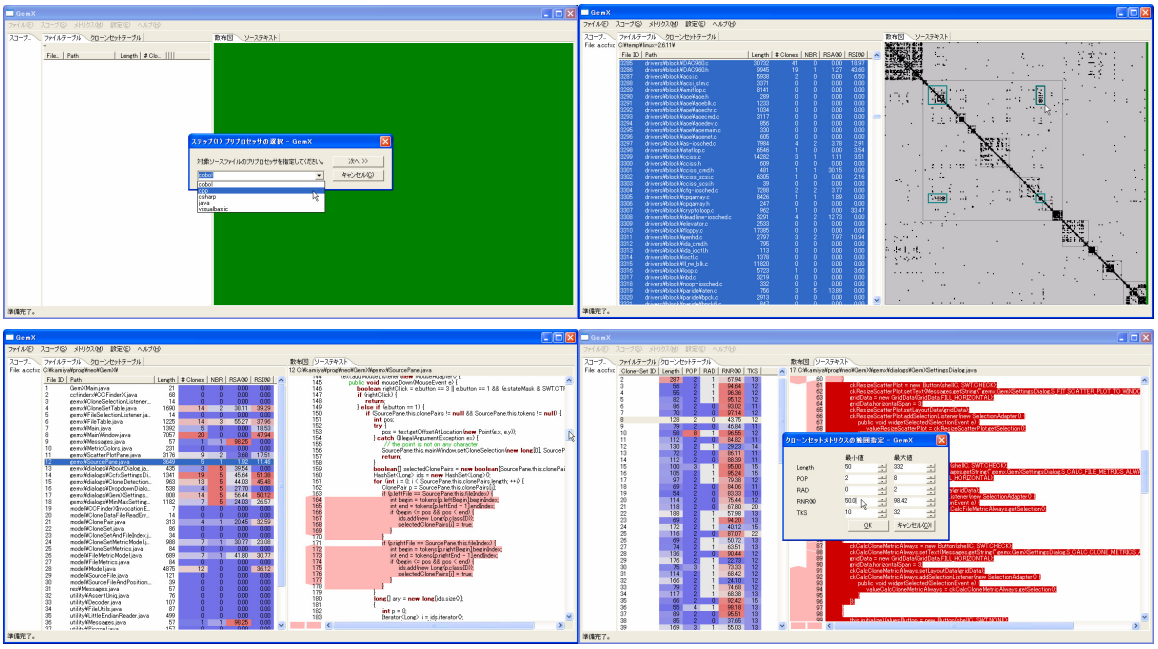

図 3 GemX のスナップショット

4. 従来の技術(または機能)との相違

ソースコード中から類似したコードを探すには、従来は、grep などのツールを利用して、該当するコード断片からパターンを作って検索するという作業が行われてきました。これに対して、旧 CCFinder(および CCFinderX)は、ソースコード全体を一度に読み込み、互いに同一または類似したコード断片(すなわち、コードクローン)となっている場所をすべて探し出し、その場所を出力するツールです。従来のツールで必要とされる手作業での検索パターン作成という手順を含むことなく、コードクローンを探すことが可能です。

また、これまでに発表されているコードクローンを検出するためのツールとしては、Dup, Duploc, CloneDR などが発表されていますが、旧 CCFinder(および CCFinderX)はスケーラビリティ(処理可能な入力ソースコードの規模)において群を抜いており、単にコードクローンを検出するだけではなく、検出されたコードクローンの分析機能を備えているという点で、ユニークなツールとなっています。

5. 期待される効果

コードクローン検出ツールは、ソースコードの修正、リファクタリング、品質評価、コード剽窃の検出といった場面で利用される技術です。ソフトウェアの保守で悩んだ経験があるすべてのプログラマ、開発組織に利用してほしいと考えています。

具体的には、プログラマや保守者は、CCFinderX を用いてソースコードを修正する際にコードクローンを検出することで、修正しようとしているコードのコピーを見つけ出すことができます。これにより、修正し忘れを防ぐことができます。リファクタリングにおいては、リファクタリングによって除去すべき重複コードを検出することができます。

コードクローン検出は、ソースコードの品質評価にも用いられます。コードクローンが多く含まれるソースコードは修正が困難であるため、コードクローンの含有量を、ソースコードの劣化(code decay)のメトリクスとして用いることができます。

2 つ以上のソフトウェアのソースコードを入力として与え、それらの間のコードクローンを検出することで、コードの剽窃を検出することができます。近年、商用ソフトウェアに GPL のコードが混入することが問題となっており、そのような検出に CCFinderX を利用することが可能です。また、一つのソフトウェアの 2 つのバージョンを比較することで、ソースコードにどのような変更が行われたかを追跡することができます。

6. 普及(または活用)の見通し

CCFinderX は、2005 年 10 月 20 日に正式にリリースされ、25 日現在、評価用パッケージを 9 セット配布したところです(旧 CCFinder の実績からの類推では、週に 1 セット程度のペースで配布が続く見込みです)。

7. 開発者名(所属)

神谷年洋(産業技術総合研究所 情報技術研究部門)

(参考)開発者URL

http://www.ccfinder.net/