

形式的なソフトウェア部品検索のための仕様からの特徴抽出

足立 智隆[†] 織田 健[‡]

電気通信大学大学院 電気通信学研究科 情報通信工学専攻[†]

電気通信大学 電気通信学部 情報通信工学科[‡]

1 はじめに

ソフトウェア開発の大規模化に伴い、コスト増大・品質低下が問題になっている。問題を解決する手段として、ソフトウェア部品再利用がある。ソフトウェアの検索分野ではソースコードのクラス名等とキーワードに論理演算子を用いて記述した検索キーとで字面の一致をとる手法が広く用いられている [1]。しかしプログラムの機能や振る舞いを検索対象としたものは少ない。そこで本稿ではソフトウェア部品検索のためのソフトウェアの機能の特徴抽出手法の提案を行う。

2 背景

2.1 ソフトウェア部品検索

ソフトウェア部品検索手法は主にキーワード手法と形式的仕様記述言語を用いたものがある。キーワード検索手法は検索キーが非形式的で取得する部品が要求を満たす確率が低くまた、形式的な手法は記述量が多く技術力も必要になる。理想の検索手法は形式的で抽象度の高いものが良い。

2.2 過去の研究

理想の検索を目指すために我々は形式的仕様言語 CafeOBJ を用いて要求段階の初期で決まる、データ型の定義、型の入出力、データ型間の関係を検索キーに用いた部品を抽象的に表現した検索手法を提案した [2]。プログラムコードを検索対象とせず、仕様を検索対象とするのはコード以外にも再利用できるであろう開発行程、仕様、設計などのプロダクトは存在し、これらを再利用可能にすることも提案している。

2.3 CafeOBJ

CafeOBJ は代数理論を用いており、関数のインタフェイスを示したシグニチャ定義と、関数の公理系の記述による代数の限定により機能を表現する。

1つの機能を表す単位をモジュールといい、他のモジュールの組み込み、ソート (集合) の宣言、ソートによる関数の定義、関数の公理の定義が含まれる。また、公理の定義で条件付きの演算を扱うことができ関数の振る舞いの制限を行なえる。

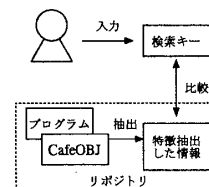


図 1: 検索手法の全体像

3 研究目標

我々の行ってきたソフトウェア部品検索のベースは型の入出力の表現を検索キーとしていた。これらは検索ユーザが求めるプログラムの機能を対象としたものでなく、ユーザ側は機能重視の検索が求められている。そこでソフトウェア部品検索に利用できる仕様からの特徴抽出の提案を目標とする。検索手法は図 1 を考えており、特徴抽出はリポジトリ内の抽出の部分の新しい手法にあたる。

4 特徴とは

本稿ではソフトウェアの機能の特徴をその部品の代表的な振る舞いと定義する。機能はアルゴリズムや構造を考慮しないソフトウェアの入出力の関係を示したものである。例えば降順ソートではリストの要素の中の順序を示している式である $list(n) \geq list(n+1)$ がソフトウェアの機能を利用して実行した出力の式で出力時の特徴を表現している関係式と考えられる。一方入力の場合はこの場合は無いが一般的には入力条件などがある。入出力それぞれやその関係を捉えることで何が行なわれたかが判明しそれがソフトウェアの機能の特徴となる。そして特徴を抽出するとはこの関係式を仕様より余計な機能を排除して導き出すことを指す。過去の研究でこの特徴である $list(n) \geq list(n+1)$ の式はデータ間の関係を表していてこの部分が型定義の情報などより機能を端的に表してあるものであった。

5 仕様からの特徴抽出

5.1 特徴抽出での着目点

機能を抽出するという観点から以下の点に着目して特徴抽出の手順の構築を行なった。

- 細分化および関数ごとの役割決め
- 再帰について

1 点目では CafeOBJ の機能解析を行ないやすくなるためと、条件式等の機能表現に関わるものの判別、そ

Feature extraction from specification for formal software retrieval

Tomotaka Adachi, Takeshi Oda

[†]Graduate School of Electro-Communications Department of Information and Communication Engineering

[‡]The University of Electro-Communications, Department of Information and Communication Engineering

して大きく演算の振る舞いが異なるリストとそれ以外のデータ型の区別を行なった。2点目はリストと組み合わせること新たな機能の表現式が導出できることから特に注目して特徴抽出に利用した。

また、小さな部品から大きな部品へと解析するスタイルを採用する。これは、どの段階の部品でも検索対象として取り出せることを目標にしている。特徴抽出の手順は1関数からモジュールへの解析へとなる。

5.2 特徴抽出手順

以下は特徴抽出の流れとなっている。なお、各段階の細かいルールについては割愛する。

1. signature 部分の op より関数ごとに分割
2. 関数ごとに公理を分割し計算順序を確認
3. 各関数ごとに種類を確認し解析: リストの有無と条件式の有無の4通りで解析方法が異なるため種類を分ける。条件式はこの手法における特徴抽出のキーとなるものである。解析は関数での入力の条件式からどのように出力に変化するかを式変形などでデータの条件式の移り変わりを見ていく。そして出力時の条件式を判断しこれを出力の条件式とし各関数の特徴とする。
4. 階層構造の関係を考慮して特徴を合成: 関数ごとの特徴を順序通りに合わせてモジュール単位の特徴として抽出する。複数モジュールがある時には下位階層から上位階層へと組み合わせ、階層構造に影響されない特徴を出力する。

5. 結果を DB 仕様に出力

5.3 実例検証

図2はリストを入力するとリスト中の最小値を返す仕様である。特徴抽出を実行すると、以下になる。

1. 分割する関数は今回は min のみ。
2. 階層に関しては1階層なので考慮しない。
3. 解析は min のみでリストである。9, 10行目に再帰を行っている事と条件文を含んでいることを考慮して以下のように導き出す。ここで (E.E.L) は先頭の要素2つとリストという表現である。

$$\begin{aligned} L(n+1) &\leq L(n) \text{ のとき} \\ \min(L(n) + L(n+1) + L(n+2 \sim)) \\ &= \min(L(n+1) + L(n+2 \sim)) \end{aligned}$$

この式の $L(n+1)$ は $L(n+1) \leq L(n)$ であることとリストの最後まで実行される再帰の式なので抽出結果は同様に行った10行目の式の解析結果と合わせて $L \rightarrow x (x \leq L(n) \quad x \in L)$ となる。

4. 階層構造がないので以下のような結果となる。

$$L : list \rightarrow x (x \leq L(n) \quad x \in L)$$

```
1. module MINOUTPUT1 [ X <= RAT ] {
2.   protecting (RAT-LIST [ X <= RAT ])
3.   signature {
4.     op min : List -> Rat
5.   }
6.   axioms {
7.     vars L L' : List
8.     vars E E' : Rat
9.     ceq min(E . E' . L)
      = min(E' . L) :if E>E'or E==E' .
10.    ceq min(E . E' . L)
      = min(E . L) :if E'>E .
11.    eq min(E) = E .
12.    eq min(nil) = nil .
13.  }
14. }
以下省略
```

図 2: 最小値を求める仕様

5.4 結果検証

5.3節で導きだされた結果は List の中で一番小さい要素を出力するということを意味している。よって図2の仕様の特徴を抽出できたと考えられる。

6 考察

提案手法を検索に用いる時には5.3節の結果を検索ユーザに入力を求める検索キーにすることで形式的な仕様に対して簡易的な検索キーを用いて検索を行うことができると考えられる。また、定数(10などの値)等の取り扱いについては解析時に2つのデータ型を分けて考えたことから数値のデータ型としてA等の文字式に置き換えて解析を行なう。そして複数モジュール時(階層を持つ仕様)のときにも抽出可能であることは他の実例で確認できている。本手法を検索に用いた時、他のキーワード手法などと比べると扱いやすさは劣るが機能を中心とした正確な検索が行われると考えられる。また我々の提案した旧手法より、ユーザの求める機能中心の検索や入力の単純化が行われたと考えられる。しかし、単純な<, >の使用していない仕様に関しては旧手法の方が有利であると考えられる。

今後の課題としては大きなシステムの仕様に組み込むようにすること、実装化が課題として残っている。

7 おわりに

今回は形式的仕様言語から機能の特徴を抽出する手法を提案した。今後の課題としては検索への本格的な適用、適用範囲の拡大、実装などが挙げられる。

参考文献

- [1] R.C. Seacord, S.A. Hissam, and K.C. Wallnau. Agora a search engine for software components. *IEEE Internet Computing*, pp. 62-70, Nov/Dec 1998.
- [2] 新井宏文, 織田健. 形式的記述に基づくモジュールシグニチャを用いたソフトウェア部品検索. 第2回情報科学技術フォーラム CD-ROM B-007, 2003.