利用関係とコードクローン関係についてのメトリクスに基づくソフトウェア部品分類手法の評価

2013SE099 国枝達也　2014SE073 成田涼
指導教員：横森励士

1　はじめに

近年のソフトウェアは大規模化しており，ソフトウェアを構成する部品の数も増大してきている．このような環境下では，ソフトウェアを構成する部品を類似性などを利用して効率的に把握することが求められている．濱島らは，ソフトウェアを構成するそれぞれの部品からメトリクスを抽出して，非階層クラスター分析による分類を行う手法を提案した [1]．メトリクスの値がある程度大きいクラスターの中に，各機能の中心となるような部品や何らかの共通性を持つ部品群が存在することが示されたが，得られた部品群にどのような部品が含まれているべきかという再現率の観点からの網羅性について評価が不十分であった．

本研究では，[1] で提案された手法に対して評価実験を行い，分類された部品集合がどのような特徴を持つかを，適合率，再現率の観点から評価を行う．これにより，[1] の手法がどのような部品群を抽出することができているかを考察する．さらに分類において用いるメトリクスの数や分割するクラスター数などを変化させることで，分類結果がどのように変化するかを調査し，それぞれの場合の特徴を調査する．これらの分析により，ソフトウェアの効率的な把握の支援において，本手法をどのように活用できるかを考える．

2　関連研究

2.1　ソフトウェア部品グラフ

ソフトウェア部品とは，その内容をカプセル化したうえで，ソフトウェアを実現する環境において交換可能な形で配置できるようにしたシステムモジュールの一部をさす．本研究では，Java ソフトウェアを対象とし，それぞれのソースコードが記述されているファイルを部品の単位として部品グラフをモデル化する．部品グラフ上の頂点は各ファイルを表し，辺はコードクローン関係や利用関係などの部品間の関係を表現する．本研究では，部品グラフ上の各頂点に出入りする辺の数を数え，利用関係とコードクローン関係に関するメトリクスとして分類に利用する．

2.2　メトリクスを用いた類似したソフトウェア部品の抽出手法について

ソフトウェア間において盗用された部品を検出する仕組みとして，小堀らはトークン数の出現回数などを比較し，それらの多くが一致した部品同士を類似部品とみなす方法を提案し，大量のソフトウェアから得られた部品群の中から，コピーなどによって生成された部品を効果的に検出した [2]．これは複数のソフトウェア間に存在する類似または同一の部品を検出する仕組みである．

濱島らの研究では，1 つのソフトウェアを構成するそれぞれの部品からメトリクスを抽出して，非階層クラスター分析による分類を行うことで，同じような役割を持っていたり，1 つの役割を実現する部品群を一纏めにして抽出することができると考えた [1]．利用関係はほかの部品からの利用と被利用に関するメトリクスであり，部品の外部とのやりとりで実現する機能の関連性を利用している．ソフトウェア部品の構成要素が同じような部品は，ほかの部品も同じように利用していることが推測される．結果として利用関係の出力辺の数も同じような値になると考えられる．ソフトウェアの部品として同じような立場にある部品は，同じような部品から利用されることが推測される．結果として利用関係の入力辺の数も同じような値になると考えられる．コードクローン関係はソースコードの複製などによって作られたコード片に関するメトリクスであり，部品内部の関連性を利用している．共通したコード断片を共有するクラス群中のクラス同士はそれぞれコードクローン関係を持っており，コードクローンを共有する部品の数も同じような値になりやすい．それぞれのメトリクスは異なる基準を用いて分類を行っていると考えられる．実際に分類した結果を評価し，メトリクスの値がある程度大きいクラスターの中に，各機能の中心となるような部品や何らかの共通性を持つ部品群が存在することを確認した．

3　利用関係とコードクローン関係についてのメトリクスに基づくソフトウェア部品分類手法の評価

3.1　研究目的

[1] における評価実験では，分類した部品群に関連性の高い部品が存在していることが示されたが，その観点で分類した部品をどれだけ検出できたかという再現率の観点からの評価が不十分であった．本研究では，[1] で提案された手法に対して評価実験を行い，分類された部品集合がどのような特徴を持つかを，適合率，再現率の観点から評価を行う．これにより，[1] の手法がどのような部品群を抽出することができているかを考察する．さらに分類において用いるメトリクスの数や分割するクラスター数などを変化させることで，分類結果がどのように変化するかを調査し，特徴を調査する．これらの分析により，得られた結果からどのように本手法を活用するべきか考察する．

3.2 研究の手順

研究の手順を以下に示す．

1. 調査対象となるソフトウェアを CCFinder[3] で解析し，コードクローン関係を取得する．
2. 調査対象となるソフトウェアを Classycle[4] で解析し，利用関係を取得する．
3. 入手したコードクローン関係，利用関係を元に，部品グラフを構築する．
4. 部品グラフ上の頂点ごとに，利用関係の出力辺数，入力辺数，コードクローン辺数などの情報を求める．さらに，分析に利用するメトリクスを選択し，各部品のメトリクス値を表に出力する．
5. 出力された表を入力とし，R 上で非階層クラスター分析を行い，各部品が所属するクラスターを得る．
6. それぞれのクラスターに含まれる部品を分類し，特徴を調査する．多くの場合，各クラスターには複数の部品群が含まれており，それぞれの部品群に分割した上でそれぞれを調査対象とする．

分析では，以下に示す利用関係とコードクローン関係に関するメトリクスを分類に利用する．それぞれのメトリクスが異なった基準で部品を分類しており，複数を組み合わせることでより詳細な分類を行うことができると考える．

C Sharing Files　コードクローンを共有するファイルの数を表す．同じクローン集合に属していると同じ値になりやすく，同じ記述を含む集合内の部品が，強く連結することで部品集合になると考えられる．

Uses Internal　利用しているソフトウェア内部のファイル数を表す．同じような機能を利用することで利用先の数も似た値になりやすく，似た機能を利用した部品集合になると考えられる．

Used By　利用されているファイル数を表す．同じような場所から利用されていると似た値になりやすく，同時に使われやすい機能群となると考えられる．

4 評価実験

4.1 評価実験での調査項目

本研究では，研究の目的を踏まえて，以下の調査項目を設定し，それらを明らかにするために評価実験を行う．

Q1　分類された部品群内の部品がどのような着眼点で似ているといえるかについて，条件を複数用意し，分類された部品の集合がどの条件を満たすかを調べる．分類結果として得られた部品群の部品のどれくらいが関係を持っているかを調査する．

Q2　Q1 で類似していると判断した部品群を対象として，部品群の外の部品で，その観点で類似している部品がどれだけ存在するかを調べる．調査対象の手法により分類された部品群に含まれているべき部品をどれだけ含むことができたかを再現率の観点から調査する．

Q3　調査対象の手法による分類に利用するメトリクスの数を増やすことで，分類の精度が上がるかを調べる．メトリクスの数を増やしたときに，分割するクラスター数をどのような値にするべきかについてもあわせて調査する．

Q4　分類に利用するメトリクスの組み合わせを変えることで，得られる部品群の傾向が変化すると考えられる．それぞれのメトリクスで抽出できる部品群の傾向の違いを調査する．

4.2 部品が類似しているかについての判断基準

分類を行った結果，複数の部品からなる部品の集合が得られるが，それらが類似しているかどうかについては様々な観点から判断することができる．以下では，どのような観点で似ていると考えられるかについて複数の基準を設定し，類似性を判断するための基準とする．その条件を満たした場合，どう似ているといえるかについて考察する．

条件 1　部品の派生元や実装しているインターフェースが同じである．派生元が同じであることや，共通の機能を持っているという観点から類似している．

条件 2　利用先の 50% 以上が同じである．または，同じ利用先を一定数以上持つ．同じ対象を扱っている場合，同じ対象に対する機能群を構成すると考えられる．

条件 3　クラス内に同一のシグネチャを持つメソッドが複数存在する．部品の役割が同じであるといえ，同様の処理が行われている部品であるという点で関連していると考えられる．

条件 4　部品群を 1 つのグループとして見たときに機能群を構成している．1 つの役割に対して，その一部を実現する部品群であると考えられる．

条件 5　同じパッケージに所属している．開発者によって役割や性質，目的などに応じて分類されたものであり，同じ目的の部品群であると考えられる．

条件 6　ファイル名の一部が一致している．ファイル名は役割を示す名前をつけることが原則であるので，類似した役割を持つと考えられる．

条件 7　共通のコードクローンを有する．同じ処理を行っていると考えられる．

また，単体部品でグループをなす部品は，

条件 A　部品を統括する部品や，多くの部品に扱われるような，プログラムにおいて重要なファイルである．この条件を満たす部品は，プログラム全体を把握する際の起点とすることができる部品である．

という条件を用いて単体として捉える意味がある部品であるかを評価する．

表 1 OpenRocket に対して適用した結果

クラスター	グループ	部品数	条件 1	条件 2	条件 3	条件 4	条件 5	条件 6	条件 7
1	1	10	53%(19)	50%(20)	43%(23)		45%(22)	48%(21)	71%(14)
3	1	12	48%(25)		67%(18)		50%(24)	48%(25)	67%(18)
5	1	6	48%(25)		33%(18)		25%(24)	24%(25)	33%(18)
6	2	2	14%(14)	5%(42)				15%(13)	40%(5)
8	4	2						50%(2)	
8	7	2		6%(32)			25%(8)		
9	1	3		14%(21)					
12	1	4	29%(7)	9%(43)	12%(25)		18%(22)	19%(21)	29%(14)
14	2	4	21%(19)	100%(4)	16%(25)		18%(22)	19%(21)	100%(4)
14	3	2	14%(14)				20%(10)	15%(13)	
14	6	2	100%(2)	100%(2)	40%(5)		50%(4)	67%(3)	100%(2)
15	1	2	8%(25)		29%(7)		8%(24)	8%(25)	29%(7)
15	2	4	16%(25)		57%(7)		17%(24)	16%(25)	80%(5)
16	3	2					22%(9)		
16	4	2		6%(34)			22%(9)	40%(5)	
17	3	3	100%(3)	43%(7)	38%(8)		100%(3)	100%(3)	100%(3)
17	4	3	50%(6)	60%(5)	38%(8)		38%(8)		100%(3)
18	1	2		8%(25)				50%(4)	
18	2	2	67%(3)	6%(31)	67%(3)		67%(3)	67%(3)	100%(2)
19	1	2	14%(14)	12%(17)				8%(13)	
20	2	2	14%(14)	9%(23)			20%(10)	15%(13)	
20	3	2			67%(3)	17%(12)	25%(8)		100%(2)

4.3 評価実験での調査結果

OpenRocket[5] を対象として分類を行った．OpenRocket は，モデルロケットの設計およびシミュレーションを行うソフトウェアで，318 の部品 (ソースファイル) から構成されている．

A1 C Sharing Files と Uses Internal をメトリクスとして用いて分類した．調査対象となったクラスターは 15 であり，単体部品に分類された部品は 33，複数の部品からなるグループは 22 となった．単体部品に分類された部品のうち条件 A を満たすものは 9 であった．意味づけできた部品数は 84 であり，全体の約 26.4% となった．表 1 は，OpenRocket を調査対象の手法で分類した結果から，複数の部品からなるグループが各条件を満たしているかを示している．条件を満たした場合，各条件の列に A2 で用いる情報が示されている．空欄の場合，その行のグループで条件を満たさなかったことを示す．

- 条件 4 以外の条件では，多くのグループが条件を満たしていたが，条件 4 を満たすグループは少ない．分類に利用した C Sharing Files は，コードクローンを共有するファイルの数を表し，似た機能を持つグループに分類されやすく，一つの役割を実現する機能群にはなりにくいと考えられる．
- 条件 1 を満たすグループは条件 5 も満たすことが多い．共通のコードクローンを有するような，類似した機能を持つグループであるという観点から，共通の機能を持っている部品は，同じパッケージに所属しやすいと考えられる．
- 条件 3 を満たすグループは，条件 7 も満たすことが多い．クラス内に類似したメソッドを持ち，それらがコードクローンとなっていると考えられる．
- 単体部品に分類された部品について，メトリクスの値が大きいとき各機能の中心となる部品であったり，他の部品を統括する部品として抽出された．

A2 各グループにおいて類似していると判断した条件毎に，グループ内の部品を対象として，その条件を満たしているグループ外の部品がほかにどれだけ存在するかを調査した．これらの情報からは，本来含まれるべき部品のうちどれくらいを結果に含むことができたかについての再現率に関する情報を得ることができる．表 1 はその結果で，再現率を百分率で示し，括弧の中の数は再現率の分母となる，グループに含まれているべき部品の数を示す．

- 再現率が低いグループが多い条件が多い．提案手法で得られる部品群は，強く関連した部品のみを集めていることが分かる．
- 条件 7 の再現率が高いグループが多い．分類に使用した C Sharing Files はソースコード内のコードクローンを共有しているファイルの数に関するメトリクスであるので，本来含まれるべき部品の間で同じ値になりやすいと考えられる．

A3 表 2 は Q1 で得られたクラスターを対象として，そのクラスターに属していた部品が，3 つのメトリクスを用いた場合にクラスター数 20, 30, 40 でどう分類されたかを示す．

- 表 2 の部品群のうちメトリクスを 3 つ用いた分類で a'〜a''' や b'〜b''' にそれぞれ共通して分類された部品は，強く関連した部品群を構成している．3 つのメトリクスを用いた分類の結果，より

細分化された複数のクラスターに分類された．

- 4 つの分類で全て同じクラスターに分類された部品群は，強く関連した部品群であり，Q1 で行った分類でも同じ部品群中のグループに所属しており，強い関連性を有していた．
- 大きなまとまりにならなかった部品は，本手法で抽出することが難しい部品群に所属する部品や，単体部品で構成される部品群であると考えられる．

表 2 クラスター数の違いによる分類の比較 1

メトリクスの数	2	3		
クラスターの数	20	20	30	40
BulkheadConfig	a	a'	a''	a'''
CenteringRingConfig	a	a'	a''	a'''
SleeveConfig	a	a'	a''	a'''
ThicknessRingComponentConfig	a	a'	a''	a'''
IntegerModel	a	b'	b''	b'''
ComponentTreeModel	a	b'	b''	b'''
SwingWorkerDialog	a	c'	c''	c'''
FixedPrecisionUnit	a	c'	c''	c'''
LicenseDialog	a	a'	d''	d'''
BareComponentTreeModel	a	a'	d''	d'''
GeneralUnit	a	b'	e''	e'''
MutableCoordinate	a	c'	a''	f'''
Coordinate	a	d'	f''	g'''

A4 3 つのメトリクスの中から 2 つのメトリクスを選ぶ 3 通りの組み合わせと，3 つのメトリクスを全て選ぶ組み合わせの，計 4 通りの組み合わせにおける結果を比較した．C Sharing Files を含む 3 つの組み合わせでは，A3 で示すような強く結びついた部品の集合からそのまま得られたが，C Sharing Files を含まない組み合わせでは，それらを完全な形で得ることはできなかった．

5 考察

4 節で得られた結果から，調査対象の手法について次のような特徴があると考えられる．

- 非階層クラスター分析での分類だけではクラスターの中に複数の部品群が存在し十分に分類できておらず，分類された部品を手作業で分類する必要がある．
- OpenRocket の部品のうち，意味付けをし分類することができた部品は，メトリクスの値が大きい部品に限られ，全体の 26.4% であり，[1] の傾向と類似していた．
- 単体部品で構成されるグループに分類された部品は，値が大きいメトリクスを 1 つ以上持つとき，各機能の中心となる部品や，他の部品を統括する部品となった．
- 意味付けを行うことができたグループの多くでは，強く関連するコピーされた部品が抽出されたが，コピーされた部品以外はグループ外になっていた．よって，調査対象の手法は関連のある部品の集合を抽出することには適さないが，強く関連するコピーされた部品の集合の抽出には役立つと考えられる．
- 分類に用いるメトリクスの数やクラスター数を変えて分類した場合でも，強く関連するコピーされた部品群は多くの場合で同一のクラスターに分類された．強く関連するコピーされた部品は同じクラスターに分類されやすいと考えられる．
- 分類に利用するメトリクスの組み合わせを変えて分類を行った場合でも，抽出された部品はメトリクスの組み合わせによらず変化しない部分があった．これらの部品群は分類に用いたメトリクスに関わらず強い共通性を持っていると考えられる．これらの部品群は内部が同じような構成になっているとも考えられ，[2] で言及していたコピーからの変更で生成された部品群であるとも考えられる．[2] と用いるメトリクスが異なっても，値の類似性からコピーなどによって生成された部品の検出に用いることができると考えられる．それ以外の部品について，メトリクスの組み合わせによって異なる部分があり，また C Sharing Files の有無によって結果の傾向が変わることから，コードクローンのメトリクスを中心として，それぞれのメトリクスの意味を追求することが今後の課題である．

6 まとめと今後の課題

本研究では，利用関係やコードクローン関係に関するメトリクスを用いて，ソフトウェア部品を分類する手法について，評価を行った．結果からは，その基準で分類した結果となる部品集合のうち，強く関連する部品のみを分類できていることが確認でき，利用するメトリクスを追加したり，クラスター数を変更しても，それらの部品は部品群としてまとまった状態であった．

今後の課題として，利用するメトリクスを変更した場合の結果の違いを調査することで，それぞれのメトリクスの役割などを調査したいと考えている．

参考文献

[1] 濱島直輝，日下光 :“利用関係とコードクローン関係に関するメトリクスに基づく類似部品抽出手法の提案”，南山大学情報理工学部 2016 年度卒業論文，2017.

[2] K. Kobori, T. Yamamoto, M. Matsushita, and K. Inoue: ”Classification of Java Programs in SPARS-J”, International Workshop on Community-Driven Evolution of Knowledge Artifacts, 2003.

[3] T. Kamiya, S. Kusumoto, K. Inoue: ”CCFinder: A multilinguistic token-based code clone detectionsystem for large scale source code”, IEEE Transactions on SoftwareEngineering, vol. 28, no. 7, pp. 654-670, 2002.

[4] Classycle: http://classycle.sourceforge.net/.

[5] OpenRocket: http://openrocket.info/.