論　文

Javaソフトウェアの部品グラフにおけるべき乗則の調査

市井　誠[†a)]　松下　誠[†]　井上　克郎[†]

An Exploration of Power Law in Java Software Component Graph

Makoto ICHII[†a)], Makoto MATSUSHITA[†], and Katsuro INOUE[†]

あらまし　ソフトウェア部品とはモジュールや関数，クラス等のソフトウェアの構成単位であり，参照や呼出しなどの形で互いに利用関係をもつ．ソフトウェア部品を頂点，利用関係を有向辺としたグラフはソフトウェア部品グラフ（部品グラフ）と呼ばれ，ソフトウェアの解析手法に広く用いられている．グラフを特徴づける要素として頂点の次数分布がある．近年，WWW 上のページのリンク関係やソーシャルネットワーク等，様々な分野のグラフで次数分布がべき乗則に従うことが明らかになり，活発に研究されている．本論文では，Java ソフトウェアに含まれるクラス間の静的な利用関係に基づく部品グラフの次数分布にべき乗則が成り立つかどうかを調査した．その結果，一つのソフトウェア及び大規模なソフトウェアの集合に関し，入次数の分布がべき乗則に従い，出次数の分布は次数の大きな範囲でのみべき乗則に従うことが明らかになった．また，一部の部分集合においても同様の性質が成り立ち，特に，単語に基づく部分集合は非常に少ない部品数でも全体集合と同様の性質をもつことが明らかになった．

キーワード　ソフトウェア部品，利用関係，部品グラフ，べき乗則

1. まえがき

近年，短期間で大規模かつ高品質なソフトウェアを開発するための技術に対する需要が高まっている．その中でも，ソフトウェアの解析に基づく手法はソフトウェアの評価や理解支援，再利用支援など様々な分野で見られる．ソフトウェアの解析における重要な概念としてソフトウェア部品グラフがある．モジュールや関数，クラスなどのソフトウェアの構成単位はソフトウェア部品（部品）と呼ばれ，互いに利用関係をもつ．部品を頂点，部品間の利用関係を有向辺としたグラフはソフトウェア部品グラフ（部品グラフ）と呼ばれる．

様々な分野において，グラフは研究対象の表現形式として用いられ，その性質が調査・研究されている．次数がべき乗則に従うグラフは，一般のグラフには見られない特徴的な性質をもつことから，物理学や社会学，生物学，情報科学など様々な分野で注目されている．例えば，新たな辺が追加される際には既に大きな次数をもつ頂点ほど接続されやすい性質や，無作為に頂点を取り除いてもある程度は構造が維持されるが，ある点を境に急激に崩壊する性質などをもつ[1]〜[3]．これまで，部品グラフの性質に対する調査，特に次数分布の観点からの調査は少なかった．部品グラフの次数にべき乗則が成り立つことが明らかになれば，これらのグラフ構造の成立や安定性にかかわる性質に基づき，ソフトウェア設計などの部品グラフに反映されるソフトウェアの要素に対して新しい観点からの分析が可能であると考えられる．

本論文では部品グラフの次数に関し，以下の三つの問題を調査する．

問題 1　一つのソフトウェアシステムに関して，様々な静的な利用関係を用いた部品グラフはべき乗則に従うかどうか．

C++，Java 等のオブジェクト指向ソフトウェアの部品グラフをべき乗則の観点から調査した研究は既に存在するが，いずれも部品間の利用関係を完全には部品グラフに反映できていない．部品グラフの構築がソースコードの字句解析の結果に基づいている，若しくは，一般に利用関係の多くを占めている変数宣言やメソッド呼出しを無視しているためである．本論文では，ソースコードの簡単な意味解析に基づいて構築し

† 大阪大学大学院情報科学研究科，豊中市
Graduate School of Information Science and Technology, Osaka University, 1–3 Machikaneyama-cho, Toyonaka-shi, 560–8531 Japan
a) E-mail: m-itii@ist.osaka-u.ac.jp

た，詳細な利用関係を反映する部品グラフを調査する．また，部品グラフは有向グラフであることを考慮し，頂点の入次数と出次数を個別に調査する．

問題 2　部品グラフを構成する部品集合を一つのソフトウェア以外から得たときもべき乗則に従うかどうか．

本論文では，一つのソフトウェアに加え，複数のソフトウェア集合や，その部分集合から構成した部品グラフについて，問題 1 に対する調査と同様に次数がべき乗則に従うかどうかを調査する．

問題 3　部品そのものの性質と次数との関連．

次数は部品を解析して得られる結果であり，何らかの部品の性質を反映した値であると考えられる．その関連を知ることは，問題 1 及び 2 について考察するときの有効な手掛りとなる．

以降，**2.** で関連研究について述べ，**3.** でソフトウェア部品グラフなどの諸定義及び解析手法などについて述べる．**4.** で実験内容及びその結果について述べ，**5.** で得られた結果に関して考察する．最後に **6.** で本論文のまとめと今後の課題について述べる．

2. 関 連 研 究

筆者らは，以前に部品グラフに対して次数の調査を行い，ソフトウェア同士を比較することにより，次数分布とソフトウェア設計との関連について考察した [4]．本論文では，部品グラフの次数分布について，より一般的な性質を得ることに焦点を当てる．

Valverde らは，Java の基本ライブラリである JDK [5] や Java プログラムのクラス図を，クラス及びインタフェースを部品，汎化や依存といった関連辺を利用関係とする部品グラフとみなしたとき，次数に関してべき乗則が成り立つことを示している [6]．また，グラフ上の辺が頂点数に対して少数であるにもかかわらず頂点間の最短距離が小さいスモールワールド性 [7] をもつことも示している．更に，これらの性質は再利用性や理解容易性などを考慮し，最適なソフトウェア設計を行った結果であると考察している．

Myers は C++プログラムのクラス図を部品グラフとみなしたとき，次数がべき乗則に従うこと，グラフがスモールワールド性をもつこと，及び階層的な構造をもつことを示している [8]．Myers の実験では，次数は入力と出力に分けて調査され，入次数と出次数はいずれもべき乗則に従う点では共通しているが，それぞれ異なる性質をもつことが示されている．また，部品グラフと同様の性質をもつグラフの生成モデルを提案している．

木下らは Java プログラムのクラス間の利用関係に対して，そのグラフ構造を測定している [9]．いくつかの Java プログラムに対し，入次数，出次数を個別に調査し，出次数は指数分布に近いことを示している．また，開発履歴情報を用いて入次数及び出次数の最大値の変遷を調査し，出次数はある程度開発が進んだ時点で頭打ちになることを報告している．

以上の研究では，部品のソースコードを静的解析することにより得た部品間の利用関係から部品グラフを構築し，次数などを調査しており，本論文でのアプローチと一致している．しかし，それぞれ本論文とは異なる点に主眼を置いている．Valverde らや Myers は，それぞれ部品グラフがべき乗則などの特有の性質をもつに至った要因についての考察に主眼を置いている．木下らは，グラフ手法を用いて部品グラフを解析することで，べき乗則に限らず部品グラフに広く見られる特徴を得ることに主眼を置くほか，ソフトウェアの変遷についても分析している．

また，関連研究にて用いられる部品グラフは，それぞれ一つのソフトウェアを解析して得たもののみである．本論文では，より一般的な部品グラフの性質を得るため，個別のソフトウェアに加え，複数のソフトウェアの集合や，その部分集合も調査対象としている．

更に，本論文では対象ソフトウェアに対して既存研究よりも深く解析し，より正確で詳細な部品グラフに基づき調査する．Valverde らや Myers は，部品間の利用関係のうち，継承関係やフィールド宣言など部分的な利用関係のみを用いて部品グラフを構築している．しかし，一般に部品間の利用関係は変数宣言やメソッド呼出しで占められており，不完全な部品グラフといえる．本論文では，それらを含め，静的に解析可能なすべての利用関係を用いる．また，Myers や木下らは，利用関係の解析を，字句解析により得られたソースコード中に記述された部品名にのみ基づいて行っている．このため，クラス名が完全限定名で記述されていない場合に正確な利用関係を求めることができない問題や，継承関係にあるクラスへの利用関係は解析できない問題をもつ．例えば，あるクラスへのメソッド呼出しが存在し，そのメソッドがその親クラスで定義されているとき，実際には親クラスも利用されているが，これらの手法では解析できない．本論文で用いる手法では，簡単な意味解析を行うことでソースコード

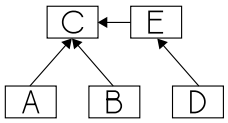

図 1　部品グラフの例
Fig. 1　Example of component graph.

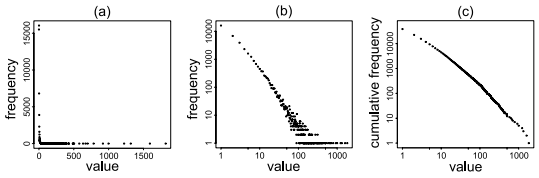

図 2　べき乗則のプロット
Fig. 2　Plot of power-law.

中の変数などの型を求めるため，上記の利用関係も解析できる．

以上で述べた研究は静的解析により得た部品グラフに基づいている．これに対し，Potanin らは動作する Java プログラムを解析し，オブジェクト間のメッセージのやり取りの関係がべき乗則に従うことを示している [10]．

3. 準　　備

3.1 ソフトウェア部品グラフ

一般に，ソフトウェア部品は広義にはモジュールや関数，クラスなどのソフトウェアの構成要素のことを，狭義には再利用を目的として設計されたソフトウェアの実体のことを指す．本論文では広義のソフトウェア部品を用いる．以降，ソフトウェア部品を単に部品と呼ぶ．

ソフトウェアは構成要素の部品間で相互に属性や振舞いを利用し合うことで一つの機能を提供する．今，ある部品がある部品を利用するとき，この部品間に利用関係が存在するという．

部品を頂点，利用関係を有向辺としたグラフを部品グラフと呼ぶ．部品グラフの例を図 1 に示す．図 1 では，部品 A, B, E は C を，D は E をそれぞれ利用していることを表している．

3.1.1 本論文における定義

本論文では，以下に定義する部品を頂点，部品間の利用関係を有向辺として部品グラフを構成する．

部品　Java クラス及びインタフェース

利用関係　以下の 6 通りの，Java ソースコードを静的に解析して得られる利用関係

- 部品がクラス若しくはインタフェースを継承
- 部品がインタフェースを実装
- 部品がクラスまたはインタフェースを変数として宣言．フィールドの型，メソッドの返値及び引数の型として宣言した場合を含む．
- 部品がクラスのインスタンスを生成
- 部品がクラス若しくはインタフェースのメソッドを呼び出す．呼び出されたメソッドが継承されたものである場合は，メソッド宣言の存在するクラス若しくはインタフェースに対しての利用関係となる．
- 部品がクラス若しくはインタフェースのフィールドを参照．参照されたフィールドが継承されたものである場合は，フィールド宣言の存在するクラス若しくはインタフェースに対しての利用関係となる．

3.2 べ き 乗 則

本論文では，部品グラフの特徴として，入次数及び出次数の分布がそれぞれべき乗則に従うかどうかに注目する．べき乗則は，Pareto の法則，Zipf の法則とも呼ばれ [11]，論文の共著関係 [3]，WWW 上のページのリンク関係 [1] など様々な分野において見られる．

ある分布がべき乗則に従うとき，要素 X が値 x をもつ確率 P は，x の $-a$ 乗に比例する．これを確率分布関数で表すと式 (1) となる．

$$P[X = x] \sim x^{-a} \tag{1}$$

この分布をプロットするときには一般に両対数軸が用いられる．通常の軸を用いてプロットすると図 2 (a) のようになり，特徴をとらえづらいためである．

式 (1) の両辺の対数をとると式 (2) のようになり，ある分布がべき乗則に従うときは，両対数軸でプロットしたときに点が傾き $-a$ の直線上に並ぶことが分かる（図 2 (b)）

$$\log P[X = x] \sim -a \log x \tag{2}$$

しかし，実際のデータの度数をプロットすると，値の大きな部分にばらつきが生じるため，大きな方から度数を累積させた値（累積度数）をプロットする．式 (1) を累積分布関数で表すと式 (3) のようになり，要素 X が x 以上の値をもつ確率は，x の $-(a-1)$ 乗に比例することが分かる(注1)．つまり，この場合も両対数軸を用いてプロットすると値が直線上に並ぶ（図 2

(注1)：累積分布関数の導出に関しては文献 [11] 参照．

(c)).

$$P[X > x] \sim x^{-(a-1)} \quad (3)$$

式 (3) の（すなわち，式 (1)，(2) の）a の値は，累積度数を両対数軸上にプロットした結果を直線へ線形回帰したときの直線の傾きから求められる．また，分布がべき乗則に従う度合として，回帰時の自由度調整済み寄与率 R^{*2} を用いる [12]．R^{*2} は，回帰モデルがデータを説明できている割合を示す値であり，0 から 1 の値をとる．

3.3 ソフトウェアメトリックス

本論文では，個々の部品に定義される特性値のことをソフトウェアメトリックス（メトリックス）と呼ぶ(注2)．

入次数及び出次数が部品の表面的な特徴と関連があるかどうかを調査するため，対象の部品のソースコードのみを用いて計測可能なメトリックス，特にソフトウェア部品の品質と関連をもつメトリックスとの相関を求める．入次数は利用される頻度であることから複雑度などの品質と関連をもち，出次数は利用する部品数であることから規模と関連をもつことが考えられるためである．本論文では，メトリックスとして一般に規模の計測に用いられる LOC（Lines of Code），及びオブジェクト指向ソフトウェアの品質測定に用いられる CK メトリックス [13] のうち，対象の部品のソースコードのみを用いて計測可能な WMC（Weighted Methods per Class）及び LCOM（Lack of Cohesion of Methods）を用いる．以下にそれぞれの詳細を示す．

LOC　コメント行を除くソースコードの行数．部品の規模を表す．

WMC　クラスに定義されたメソッドの重み付き和．この値が大きいほど，複雑なクラスである．メソッドの重みとしては以下の 2 種類を用い，それぞれ WMC1, WMC2 とする：

- **WMC1**：すべて 1．この値はクラスに定義されるメソッド数そのものであり，実装された機能の大きさを表すと考えられる．
- **WMC2**：サイクロマチック数 [14]．分岐や繰返しに基づき計測され，実装の複雑さを表す．

LCOM　クラスの凝集度．一般に LCOM は複数の定義が存在するが，本論文ではクラス中のメソッドが属性を利用する割合を表す LCOM5 を用いる [15]．LCOM5 は，あるクラス中のすべてのメソッドがそれぞれすべての属性を利用するとき 0，すべてのメソッドがそれぞれただ 1 個の属性のみを利用するとき 1 となる値であるこの値が小さいほど凝集度が高い，すなわち実装された機能のまとまりが強い．なお，LCOM5 は実装をもつメソッドが 1 個以上であるクラスにのみ定義されるため，LCOM5 と次数との相関は LCOM5 が定義可能なクラスのみを用いて求める．

表 1　実験で用いる部品集合
Table 1　Component set for experiment.

	頂点数	辺数	LOC
JDK	11,556	107,198	1.1M
APACHE	59,486	303,755	4.2M
ALL	180,637	1,808,982	14M
RND10K	10,000	6,184	780K
RND1K	1,000	52	80K
REL10K	9,286	17,201	2.1M
REL1K	972	1,218	250K
REL1H	163	1,086	76M
KWD10K	8,938	24,317	1.6M
KWD1K	1,002	1,564	290K
KWD1H	129	124	28K

3.4 部品集合

実験では，一つのソフトウェアの部品集合として 1 種類，複数のソフトウェアの集合の部品集合として 2 種類，部分集合の部品集合として 8 種類を用いる．それぞれの部品数（頂点数），利用関係数（辺数），総行数を表 1 に，概要を以下に示す．

3.4.1 一つのソフトウェア

JDK　Java の基本ライブラリである JDK1.4 に含まれる部品集合．基礎的な部品を中心とした利用関係が形成されていると考えられる．関連研究においても調査されている集合であり，結果を比較できる．

3.4.2 複数のソフトウェア集合

APACHE　Apache Project [16] で開発された複数のソフトウェアの集合．それぞれのソフトウェアは比較的独立して開発されているが，共通ライブラリとしてのソフトウェアも開発されており，それらを中心とした利用関係が形成されていると考えられる．

ALL　JDK に APACHE を含む様々なオープンソースソフトウェアを加えたもの．JDK の様なライブラリ部品からアプリケーション，小規模なサンプルコードまで様々な種類の部品が含まれる．APACHE と比較して互いに関連性の低い多数のソフトウェアの集合である．含まれるソフトウェアの共通点はほとんど JDK を利用していることのみであり，JDK を中心とした

(注2)：一般には，メトリックスはソフトウェアの開発活動中に計測された値から得られる特性値を意味する．

利用関係が形成されると考えられる．

3.4.3 部 分 集 合

部分集合 ALL より，以下の 3 通りの方針に基づき抽出した部品集合を用いる．それぞれ明示的にはソフトウェア単位やモジュール単位とならない．

無作為 無作為に取り出した部品集合

利用関係 ある部品を利用する部品集合．基準となる部品は，部品数が約 10,000，約 1,000，約 100 となるようなものからそれぞれ無作為に選ぶ．

単語 ある単語をソースコード中に含む部品集合．基準となる単語は，部品数が約 10,000，約 1,000，約 100 となるようなものからそれぞれ無作為に選ぶ．

以上の方針に基づき，以下の 8 通りの部品集合を調査に用いる．

［RND10K，RND1K］ 方針：無作為による部品集合．

［REL10K］ 方針：利用関係による部品集合．基準となる部品は `java.util.HashMap`．

［REL1K］ 方針：利用関係による部品集合．基準となる部品は `java.io.OutputStreamWriter`．

［REL1H］ 方針：利用関係による部品集合．基準となる部品は `java.awt.GraphicsEnvironment`．

［KWD10K］ 方針：単語による部品集合．単語は `getstring`．

［KWD1K］ 方針：単語による部品集合．単語は `labels`．

［KWD1H］ 方針：単語による部品集合．単語は `getsummary`．

3.5 解 析 手 法

3.5.1 SPARS-J

Java ソースコードの解析，及び利用関係の解析にはソフトウェア部品検索システムである SPARS-J の解析部を用いる [17]．SPARS-J は Java ソースコードを解析し，クラス及びインタフェースを部品として登録する．また，部品間の静的な利用関係を解析する．SPARS-J で解析される利用関係は **3.1** で述べた本論文での定義と一致している．

3.5.2 解析の流れ

（1） Java ソースコードの解析：SPARS-J 登録部が Java ソースコードを解析してデータベースに登録する．同時に利用関係の解析及びメトリックスの計算を行う．

（2） 解析内容の出力：SPARS-J のデータベースから解析結果である部品情報，すなわちメトリックス及び部品間の利用関係を出力する．

（3） 出力の分析：出力された情報に対して度数分布のプロットや相関の計算などを行う．

4. 実　　験

4.1 実 験 内 容

実験 1 問題 1 に対する実験として，一つのソフトウェアの部品集合である JDK を対象とし，本論文における定義の部品グラフに関して入次数及び出次数がべき乗則に従うかどうかを確認する．入次数及び出次数がべき乗則に従うかどうかを確認するために，両対数軸上にそれぞれの分布をプロットし，直線回帰を行う．

実験 2 問題 2 に対する実験として，様々な部品集合の部品グラフを対象として，実験 1 と同様の実験を行う．複数ソフトウェアの集合の部品集合である APACHE 及び ALL，部分集合である RND1H〜KWD10K を対象として，実験 1 と同様の結果が得られるかどうかを調査する．

実験 3 問題 3 に対する実験として，入次数及び出次数と，個々の部品との関連を調査する．部品集合 ALL に関して，入次数及び出次数とメトリックスとの相関係数を求める．相関係数としては，分布の異なる値の集合同士の相関を求めることができるスピアマンの順位相関係数を用いる．入次数及び出次数の関連する組合せについては，得られた相関係数が一般に弱い相関と解釈される値 (0.2〜0.4) の場合，散布図によりその関連を調査する．

また，本論文で用いたメトリックスで表現されない性質，例えば部品の役割などとの関連を調査するため，入次数及び出次数の上位 5 部品の内容を確認する．

4.2 実 験 結 果

4.2.1 実　験　1

入次数の累積度数分布を図 3 に，出次数の累積度数分布を図 4 に示す．それぞれを直線に回帰して求めた，式 (1) での a の値及び R^{*2} の値を表 2 に示す．

入次数 プロットがほぼ直線であり，回帰した結果の R^{*2} も非常に高い値であることから入次数はべき乗則に従うことが分かる．

出次数 プロットは，次数の大きな部分については直線であるが，次数が小さな範囲で傾きに変化が見られ，完全にはべき乗則に従っていない．直線に回帰したときの R^{*2} もやや低い値である．Myers の調査 [8] でも入力と出力は個別に調査されているが，この傾向は観測されていない．

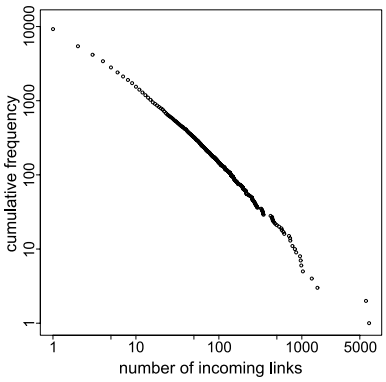

図 3　入次数の累積度数分布（JDK）
Fig. 3　Cumlative distribution of incoming links (JDK).

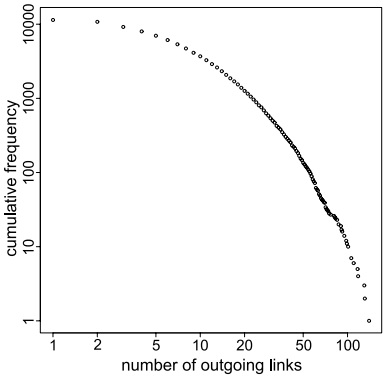

図 4　出次数の累積度数分布（JDK）
Fig. 4　Cumlative distribution of outgoing links (JDK).

表 2　次数分布の特性量（JDK）
Table 2　Characteristic value of degree distributions - JDK.

	a			R^{*2}
入次数	2.11	±	0.00866	0.987
出次数	3.10	±	0.0818	0.876

4.2.2 実　験　2

まず，複数のソフトウェア集合の部品集合である APACHE 及び ALL の結果を示す．入次数及び出次数のプロットを図 5 及び図 6 に，それぞれを直線に回帰して求めた a の値及び R^{*2} の値を表 3 及び表 4 に示す．

図から，APACHE，ALL 両方の部品集合に関して JDK と同様の傾向が見られることが分かる．すなわち，複数のソフトウェア集合に関しても以下が分かる．

入次数　べき乗則に従う．a の値は約 2 であり，JDK での値に近い．

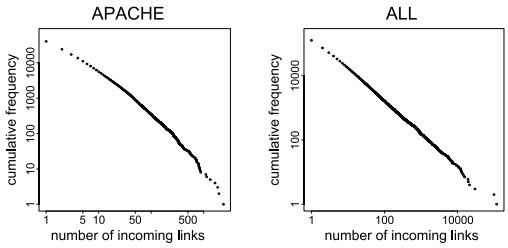

図 5　入次数の累積度数分布（APACHE と ALL）
Fig. 5　Cumlative distribution of incoming links (APACHE & ALL).

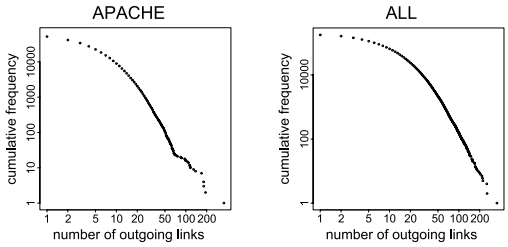

図 6　出次数の累積度数分布（APACHE と ALL）
Fig. 6　Cumlative distribution of outgoing links (APACHE & ALL).

表 3　入次数分布の特性量（APACHE と ALL）
Table 3　Charasteristic value of incoming degree distributions (APACHE & ALL).

	a			R^{*2}
APACHE	2.37	±	0.0109	0.981
ALL	2.02	±	0.00145	0.999

表 4　出次数分布の特性量（APACHE と ALL）
Table 4　Charasteristic value of outgoing degree distributions (APACHE & ALL).

	a			R^{*2}
APACHE	3.41	±	0.0637	0.942
ALL	3.66	±	0.0693	0.903

出次数　値の大きな部分でのみべき乗則に従う．直線に回帰したときの R^{*2} の値も JDK と同様にやや低く，a の値は約 3 である．

続いて，部分集合の結果を示す．入次数のプロットを図 7 に，直線に回帰して求めた a の値及び R^{*2} の値を表 5 に示す．また出次数のプロットを図 8 に示す．これらについて，部品集合の構成方針ごとに特徴を挙げる．

無作為　RND1K は入次数・出次数ともにべき乗則には従っていない．RND10K は入次数がべき分布の特徴をもつ．

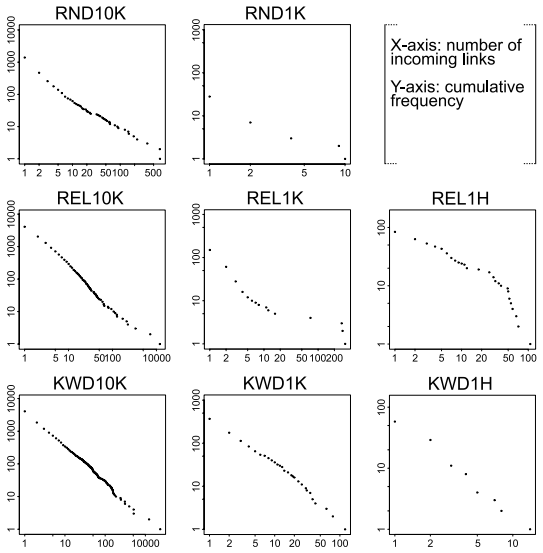

図 7　入次数の累積度数分布（部分集合）
Fig. 7　Cumlative distribution of incoming links (subset).

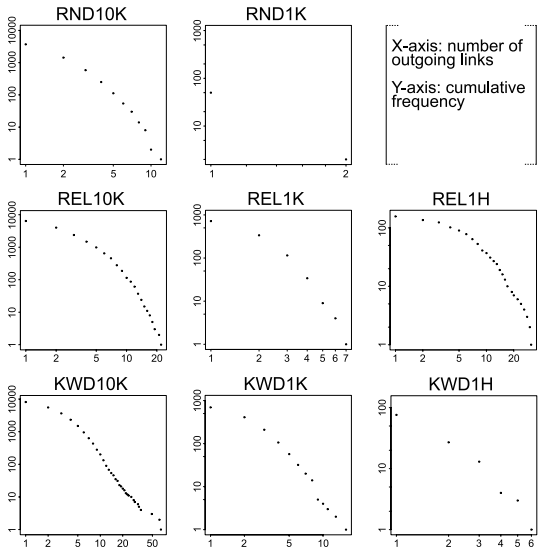

図 8　出次数の累積度数分布（部分集合）
Fig. 8　Cumlative distribution of outgoing links (subset).

表 5　入次数分布の特性量（部分集合）
Table 5　Charasteristic value of incoming degree distributions (subset).

	a			R^{*2}
RND10K	1.94	±	0.0210	0.979
RND1K	2.27	±	0.177	0.926
REL10K	2.30	±	0.0203	0.986
REL1K	1.63	±	0.0815	0.806
REL1H	1.83	±	0.0646	0.867
KWD10K	2.12	±	0.00927	0.993
KWD1K	2.21	±	0.0332	0.980
KWD1H	2.62	±	0.0805	0.983

利用関係　部品数が 1000 部品未満である REL1H でもべき乗則に従う．ただし，全体的に a の値がもとの部品集合である ALL と比較して小さく，利用関係が一部の部品に集中していることが分かる．

単語　利用関係の部品集合と同様，部品数が 1000 未満の部品集合においても入次数にべき乗則が成り立つことが分かる．また，a の値も ALL に近い値となっている．

4.2.3 実　験　3

部品集合 ALL について，入力及び出次数上位 5 部品をそれぞれ表 6 及び表 7 に，次数とメトリックスとの相関を表 8 に示す．また，表 8 にて弱い相関が見られる入次数と WMC1 及び出次数と LCOM の散布図を図 9 に示す．なお，入次数と WMC1 の散布図は見やすさのために片対数軸を用いている．

入次数はどのメトリックスとも相関が低い．WMC1 とやや相関が見られるが，散布図から関連性は見られない．また，入次数が上位の部品は，JDK の基礎的な役割をもつ部品で占められている．例えば String は

表 6　入次数上位 5 部品
Table 6　top 5 components on number of incoming links.

部品名	LOC	入力	出力
java.lang.String	675	116,239	21
java.lang.Object	35	98,261	4
java.lang.Class	605	29,682	41
java.lang.Exception	15	21,046	2
java.lang.Throwable	136	19,519	12

表 7　出次数上位 5 部品
Table 7　top 5 components on number of outgoing links.

部品名	LOC	入力	出力
org.apache...FunctionEval	364	1	354
org.jgraph.GPGraphpad	2,196	129	255
com.jgraph.GPGraphpad	2,200	130	253
org.jgraph.GPGraphpad	542	210	253
org.eclipse...ASTConverter	4,520	3	222

表 8　次数とメトリックスの相関
Table 8　Correlation among degree and metrics.

	出次数	LOC	WMC1	WMC2	LCOM
入次数	0.002	0.070	0.239	0.075	0.118
出次数	-	0.824	0.641	0.751	0.399
LOC	-	-	0.793	0.816	0.462
WMC1	-	-	-	0.567	0.494
WMC2	-	-	-	-	0.332

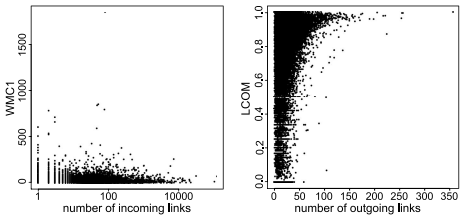

図 9　次数とメトリックスの散布図
Fig. 9　Scatter plots of degree and metrics.

文字列を表現し，Object はすべての親クラスである．しかし，他の共通点は見られない．これより，入次数は，部品のソースコードのみの解析で得られた部品の規模や複雑度，凝集度との直接的な関連は低く，設計時に与えられた役割と関連が高いと考えられる．ただし，メトリックスとの相関については，本論文で用いたメトリックスは互いにある程度の相関があり，一般のメトリックスと相関がないとはいえない．例えば，今回用いなかった，部品の役割の基礎性や汎用性などの再利用にかかわる値を計測できるメトリックスを用いることで，相関が得られる可能性がある．

出次数は LOC や WMC と相関が高い．部品グラフの出力辺は，ソースコード内で他の部品を利用する記述に基づくため，出次数と LOC の相関が見られることは自然である．出次数と WMC との相関は，LOC がそれぞれの WMC と相関が高いことが反映されていると考えられる．また，出次数は LCOM ともやや相関が見られ，散布図より，出次数の大きな部品は LCOM が高い傾向が分かる．LCOM は WMC や LOC とやや相関が見られることが反映されたものだと考えられる．LCOM が WMC 等と相関が見られる理由としては，一般にクラスの規模が大きくなるとクラスのもつ属性のうち一部のみを利用するメソッドが増え，凝集度が下がる傾向にあるためであると考えられる．これは，メソッドの抽出といったリファクタリングなどにより生じる．

4.3　実験結果のまとめ

1. で述べた問題 1〜3 に関し，実験により明らかになったことを以下にまとめる．

問題 1　実験 1 にて，変数宣言やメソッド呼出しなどを含む様々な静的な利用関係に基づく部品グラフについて，次数がべき乗則に従うかどうか調査した．結果として，入次数は非常に高い度合いでべき乗則に従うことが明らかになった．また，出次数は両対数軸プロットにおいて，値の大きな部分ではほぼ直線であるが値の小さな部分で曲線となり，完全にはべき乗則に従わないことが明らかになった．

問題 2　実験 2 にて，一つのソフトウェア以外から構成される部品グラフについて同様の調査を行った．結果として，複数のソフトウェア集合の部品グラフは，一つのソフトウェアの部品グラフと同様の結果が得られた．また，部分集合による部品グラフは，部分集合の構成の仕方によって異なる結果が得られた．

問題 3　実験 3 にて，部品そのものの性質と入力及び出次数との関連について，相関を求め，次数上位の部品の内容を分析して調査した．結果として，入次数は本論文で用いたメトリックスとは相関がなく，設計時に部品に与えられた役割や責任と関連し，出次数は部品そのものの規模や複雑さと関連することが明らかになった．

5.　考　　察

本論文の実験により，二つの興味深い結果が得られた．一つは，様々な利用関係を辺とした部品グラフでは，入次数はべき乗則に従うが，出次数は厳密には従わないという非対称性がみられた点である．もう一つは，部品集合の部分集合から構成した部品グラフにも全体の部品グラフと同様の性質が見られた点である．以下，入次数，出次数に関して得られた結果について個別に考察した上でそれらの非対称性について考察する．また，部分集合に見られたべき乗則について考察する．

5.1　入　次　数

部品グラフの入次数がべき乗則に従うことの意味は，単にごく一部の部品が頻繁に利用されることのみではない．次数分布がべき乗則に従うグラフの生成モデルは複数提案されており，いずれも新しい辺は既に多くの次数をもつ頂点に接続されやすい点で共通している [18]．これは，表 6 に示されるような部品グラフ中で大きな入次数をもつ部品は，新たな部品が追加されたときに，より大きな入次数をもつことを意味する．これより，部品グラフに新たな部品が追加された場合も，次数の大きな部品群の入れ換わりは生じにくいと考えられる．

この性質により，ソフトウェア開発の進行に伴うソフトウェアアーキテクチャの変化を観測できると考えられる．部品グラフは，ソフトウェア開発の進行に伴う部品や利用関係の追加や削除により変化するが，上述した性質より，入次数が上位の部品群には変化が生

じにくい．しかし，ソフトウェアアーキテクチャなどの大きな枠組みでの変更が生じた場合には，それらの部品に入れ換わりが生じると考えられる．例えば，新たな共通部品を追加し，既存の多くの部品を，追加した部品を利用するように変更することが挙げられる．したがって，入次数が上位となっている部品群の構成の移り変わりを見ることで，アーキテクチャの変化の検出や，その安定度の測定ができると考えられる．

以上は次数がべき乗則に従うグラフに一般に見られる性質に基づいた考察であるが，部品グラフは入次数と出次数の非対称性など，一般のグラフとは異なる性質も観察されている．そのため，本考察の内容に対し実際に調査・検証を行うことが必要である．

5.2 出 次 数

出次数は，CK メトリックスでは CBO（Coupling Between Objects）として定義され，この値が大きいと複雑であり保守や理解が困難であるとされる [13]．実験では，出次数は値の大きな範囲のみに注目するとべき乗則に従うことが明らかになった．これは，高い出次数をもち保守性などに問題のある部品（クラス）は一般的に存在することを意味し，以下の 2 通りの解釈が可能である．まず，保守性に問題がある部品も，多くはリファクタリングされず放置されることが考えられる．これは多くの開発者若しくは管理者が，設計品質について意識不足であることを意味する．これに対し，複雑になることを避けられない部品が一般に生じやすいことも考えられる．これはオブジェクト指向など，現在のソフトウェア開発パラダイムによる品質の限界を意味する．本研究で得られた結果からは一方に絞ることはできず，様々なソフトウェアに対して調査し，傾向を分析することが必要である．

出次数が従う分布について考える．実験により得られた出次数の分布は，値の大きな範囲ではべき乗則のような長い裾野（long-tail）をもつが，値の小さな範囲では対数正規分布のような頂点をもつ形状となっている．この特徴はファイルサイズの分布として知られる Double-Pareto 分布に一致している [19]．出次数は LOC と相関が高いことを考慮すると，ファイルサイズと同様の分布に従うことは直感的に妥当であるが，この観点からの調査及び検証を改めて行う必要がある．

5.3 入次数と出次数の非対称性

Myers らは入力と出力の非対称性，すなわち入次数と出次数はいずれもべき乗則に従うが，プロットの傾きが異なる点に注目していた [8]．本論文では，この非対称性は分布そのものが異なるという形で現れた．これは本論文では辺としてより詳細な利用関係を用いたためであると考えられる．出力に注目したとき，継承関係や属性はほとんどもたない部品が多いのに対し，メソッド呼出しや変数宣言を含めた場合には，小数の利用関係をもつ部品が最も多くなるためにこのような結果が得られたと考えられる．

また，入次数は部品集合により値域が大きく異なったのに対して，出次数は ALL でも 400 未満であり，事実上の最大値が存在することを示唆している．これは，入力と出力が異なる部品の性質に関連していたことが要因であると考えられる．入次数が大きな部品は基礎的な役割をもつ部品であり，部品集合が大きくなればよりいっそう利用され次数が大きくなるのに対して，出次数が大きな部品は規模が大きく複雑な部品であり，部品集合の大きさとは独立であることを反映していると考えられる．

5.4 部分集合におけるべき乗則

部分集合のグラフについても全体と同様の性質がみられるものがあった．次数がべき乗則に従うグラフは，スケールフリーネットワークとも呼ばれ，無作為にノードを取り除いていっても，ある程度は構造が維持されるが，ある時点から急速に構造が崩壊する性質をもつ [2]．

利用関係に基づいて取り出した場合は，最低限，基準となる部品に対する利用関係が存在するため，無作為に取り出した場合と比較して少ない部品数でもべき乗則がみられた．しかし，逆にその部品に利用関係が集中する傾向が見られた．単語に基づく取出し方は，明示的には利用関係とは無関係な取出し方であるが，結果として最も全体に近い性質が得られた．これは，同じモジュールに属する部品など，関連性のある部品群は共通する語を含むことが多いことが一つの要因であると考えられる．このような部品群は密な利用関係をもつことが多い．また，もう一つの要因として，ある部品が対象の語を名前として含む場合にはその部品を利用する部品も変数宣言などの形で語を含むため，利用関係が維持されることが挙げられる．

これらの結果は，一つのソフトウェア及びソフトウェア集合の部品グラフを対象とする手法が，それらの部分集合の部品グラフに対しても適用できる可能があることを示唆している．ただし，実際の適用実験を行い有効性を評価することが必要である．

6. むすび

本論文では，次数分布がべき乗則に従うかどうかという観点から様々な部品集合の部品グラフを調査した．

まず，既存研究よりも多くの利用関係に基づき構成した部品グラフの次数分布がべき乗則に従うかどうかを調査した．その結果，入次数はべき分布に従い，出次数はべき分布に似た異なる分布に従うことがあきらかになった．また，その性質は一つのソフトウェアだけではなく，複数のソフトウェア集合や，その部分集合についても成り立つことが明らかになった．更に，個々の部品の次数と関連する部品の性質に関して，入次数は部品の役割，出次数は部品の規模や複雑さと関連することが明らかになった．

本論文では部品間の利用関係を区別せずに用いて部品グラフを構成したが，利用関係の種類ごとに個別の部品グラフを作成し，同様の調査を行うことが今後の課題である．

謝辞 本研究の一部は文部科学省「e-Society 基盤ソフトウェアの総合開発」の委託に基づいて行われた．また，本研究の一部は，文部科学省 21 世紀 COE プログラム（研究拠点形成費補助金）の研究助成によるものである．ここに記して謝意を表す．

文　献

[1] R. Albert, H. Jeong, and A.L. Barabási, “Diameter of the world-wide web,” Nature, vol.401, no.6749, pp.130–131, Sept. 1999.
[2] A.L. Barabási, Linked: The New Science of Networks, NHK 出版，2002.
[3] M.E.J. Newman, “The structure of scientific collaboration networks,” Proc. Natl. Acad. Sci. USA, vol.98, no.2, pp.409–415, 2001.
[4] 市井　誠，松下　誠，井上克郎，“ソフトウェア部品の利用関係におけるスケールフリー性の調査,” 信学技報，CS2005-69, Dec. 2005.
[5] “Java technology,” http://java.sun.com/
[6] S. Valverde, R. Ferrer-Cancho, and R.V. Solé, “Scale-free networks from optimal design,” Europhys. Lett., vol.60, no.4, pp.512–517, 2002.
[7] D.J. Watts and S.H. Strogatz, “Collective dynamics of ’small-world’ networks,” Nature, vol.393, no.6684, pp.440–442, June 1998.
[8] C.R. Myers, “Software systems as complex networks: Structure, function, and evolvability of software collaboration graphs,” Phys. Rev. E, vol.68, no.4, 046116, 2003.
[9] 木下喜幸，玉井哲雄，“グラフ手法による java プログラムの構造と構造変化の分析,” 情処学研報，2006-SE-152, pp.41–48, May 2006.
[10] A. Potanin, J. Noble, and M. Frean, “Scale-free geometry in oo programs,” Commun. ACM, vol.48, no.5, pp.99–103, May 2005.
[11] M.E.J. Newman, “Power laws, pareto distributions and zipf’s law,” Contemporary Physics, vol.46, pp.323–351, 2005.
[12] 久米　均，飯塚悦功，入門統計的方法 2 回帰分析，岩波書店，1987.
[13] S.R. Chidamber and C.F. Kemerer, “A metrics suite for object oriented design,” IEEE Trans. Softw. Eng., vol.20, no.6, pp.476–493, June 1994.
[14] 山田　茂，高橋宗雄，ソフトウェアマネジメントモデル入門，共立出版，1993.
[15] L.C. Briand, J.W. Daly, and J. Wust, “A unified framework for cohesion measurement in object-oriented systems,” Empir. Softw. Eng., vol.3, no.1, pp.65–117, 1998.
[16] “The apache software foundation,” http://www.apache.org/
[17] K. Inoue, R. Yokomori, T. Yamamoto, M. Matsushita, and S. Kusumoto, “Ranking significance of software components based on use relations,” IEEE Trans. Softw. Eng., vol.31, no.3, pp.213–225, March 2005.
[18] 増田直紀，今野紀雄，複雑ネットワークの科学，産業図書，2005.
[19] M. Mitzenmacher, “Dynamic models for file sizes and double pareto distributions,” Internet Mathematics, vol.1, no.3, pp.305–333, 2003.

（平成 18 年 7 月 24 日受付，19 年 1 月 10 日再受付）

市井　誠

平 16 阪大・基礎工・情報卒．現在，同大大学院情報科学研究科博士後期課程在学中．ソフトウェア部品検索の研究に従事．情報処理学会会員．

松下　誠

平 5 阪大・基礎工・情報卒．平 10 同大大学院博士後期課程了．同年同大・基礎工・情報・助手．平 14 阪大・情報・コンピュータサイエンス・助手．平 14 同専攻・助教授．博士（工学）．ソフトウェアプロセス，オープンソース開発の研究に従事．情報処理学会，日本ソフトウェア科学会，ACM 各会員．

井上　克郎　（正員）

昭 54 阪大・基礎工・情報卒．昭 59 同大大学院博士課程了．同年同大・基礎工・情報・助手．昭 59〜昭 61 ハワイ大マノア校・情報工学科・助教授．平元阪大・基礎工・情報・講師．平 3 同学科・助教授．平 7 同学科・教授．平 14 阪大・情報・コンピュータサイエンス・教授．博士（工学）．ソフトウェア工学の研究に従事．情報処理学会，日本ソフトウェア科学会，IEEE，ACM 各会員．