Artigo 1

Rodrigo Rocha Gomes e Souza

ABSTRACT

1. INTRODUÇÃO

Recuperação de arquitetura de *software* é o ato de extrair aspectos da arquitetura de um sistema através de artefatos como código-fonte. Muitos esforços têm se concentrado no particionamento de sistemas em módulos arquiteturais — grupos coesos de entidades de código-fonte (classes ou funções) — através de algoritmos de agrupamento (*clustering*). Não há consenso, no entanto, sobre quais algoritmos de agrupamento são mais adequados para a tarefa de recuperação de módulos arquiteturais.

Uma formas de avaliar um algoritmo de agrupamento consiste em aplicá-lo a um sistema cuja organização do código-fonte em módulos seja conhecida e então comparar os módulos do sistema com os módulos encontrados pelo algoritmo através de uma medida de distância entre particionamentos. Infelizmente é difícil encontrar sistemas documentados nesse nível de detalhe, e por isso os estudos experimentais realizados são superficiais. Alguns estudos revelam que a avaliação de um algoritmo varia bastante de sistema para sistema. Devido ao pequeno tamanho das amostras analisadas, no entanto, não existe uma explicação que permita que se determine a priori se um algoritmo fornece bons resultados para um sistema.

Propomos uma abordagem de avaliação complementar, baseada na análise de sistemas de software sintéticos. Esses sistemas são gerados por computador a partir de modelos parametrizáveis e obedecem a estruturas modulares conhecidas. Com essa abordagem é possível avaliar algoritmos de agrupamento com amostras arbitrariamente grandes de sistemas arbitrariamente complexos e, sobretudo, estudar como o desempenho dos algoritmos é afetado por parâmetros que definem os sistemas sintéticos.

Neste artigo apresentamos um modelo para a geração de sistemas sintéticos e avaliamos o realismo desse modelo, isto é, a similaridade entre os sistemas sintéticos e sistemas reais.

Para fins de comparação, essa mesma análise é realizada sobre modelos genéricos disponíveis na literatura sobre redes complexas.

Na próxima seção.... na seção x.... por fim, ...

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Redes de Dependências entre Classes

Para recuperar os módulos de um sistema orientado a objetos é suficiente considerar as suas classes e os relacionamentos de dependência entre elas — uma rede de dependências entre classes. Essas redes podem ser extraídas automaticamente a partir da análise estática do código-fonte ou do código objeto do sistema que se deseja estudar.

Pesquisas recentes na teoria das redes complexas encontraram características comuns a redes que representam objetos de diversos domínios (sociologia, biologia, linguística, tecnologia etc.), incluindo as redes de dependências entre classes. A partir daí foram criados diversos modelos que procuram explicar como essas redes são formadas.

2.2 Conceitos

Arestas externas

Lei de potência

2.3 Modelos de Geração de Redes Complexas

Bollobas: pensando na Web. Não tem estrutura de módulos embutida. Parâmetros são probabilidades.

O modelo X...

TODOS ASSUMEM LEI DE POTÊNCIA

2.3.1 Modelo LFR

Lancichinetti, Fortunato e Radicchi [4] criaram um modelo de rede com estrutura de módulos embutida que é usado para avaliar algoritmos de clustering. O modelo não foi baseado em nenhum domínio em particular, mas incorpora características encontradas em redes de vários domínios. Ele gera grafos não-orientados e não-ponderados cuja distribuição dos graus dos vértices e cuja distribuição dos tamanhos dos módulos são ambas leis de potência e, portanto, bastante heterogêneas. Mais precisamente, o número de vértices cujo grau é k é proporcional a $k^{-\gamma_1}$, onde o expoente γ_1 tipicamente varia entre 2 e 3, e o número de módulos com x vértices é proporcional a x^{γ_2}, com γ_2 entre 1 e 2.

O modelo possui os seguintes parâmetros:

- quantidade de vértices (N);
- expoente da distribuição de graus (γ_2);
- grau médio ($\langle k \rangle$);
- grau máximo (k_{max});
- expoente da distribuição de tamanhos dos módulos (γ_1);
- número mínimo de vértices por módulo (c_{min});
- número máximo de vértices por módulo (c_{max});
- parâmetro de mistura (μ).

O modelo gera redes que seguem aproximadamente os parâmetros fornecidos. O parâmetro de mistura é um número entre 0 e 1 que indica a fração das arestas de cada vértice que são compartilhadas com vértices de outro módulo. Por exemplo, se $\mu = 0,4$, então 40% das arestas de cada vértice são ligadas a vértices que não pertencem ao mesmo módulo do vértice de origem.

Discussão

O modelo gera grafos não-orientados, uma representação muito simplificada das redes de dependências entre classes. Por essa razão ele não é adequado para testar algoritmos de agrupamento que consideram a informação de sentido das arestas. Além disso, ele considera que todos os vértices possuem arestas para vértices de outros módulos, o que certamente não é verdade no domínio de software. Por fim, os vértices de um módulo podem se ligar a vértices de qualquer outro módulo, sem restrição. Essa característica difere do que se encontra em programas de computador, onde as dependências entre módulos são controladas (por exemplo, ao se adotar uma arquitetura em camadas).

2.3.2 Modelo Aleatório com Módulos

Baseado em Erdos-Renyi...

2.4 O modelo S

Considerando as limitações do modelo LFR, propomos um novo modelo de rede com estrutura de módulos embutida, baseado no modelo de grafo orientado de Bollobás [2], que foi inspirado na rede de links entre páginas da Web. O modelo possui os seguintes parâmetros:

- número de vértices (N);
- arquitetura (a) — uma rede representando os módulos e as ligações permitidas entre módulos;
- probabilidades α, β e γ, tal que $\alpha + \beta + \gamma = 1$;
- probabilidade μ;
- δ_{in} e δ_{out}.

Inicialmente é criada uma rede contendo um vértice com autolaço (aresta ligando o vértice a ele próprio) para cada módulo representado na arquitetura e cada vértice é incluído no módulo correspondente. O algoritmo consiste em alterações sucessivas à rede até se alcançar o número de vértices desejado. Na descrição a seguir, "escolher um vértice v de acordo com $k_{out} + \delta_{out}$" significa escolher um vértice v de modo que a probabilidade de se escolher um vértice v_i é proporcional a $k_{out}(v_i) + \delta_{out}$, onde $k_{out}(v_i)$ é o grau de saída do vértice v_i. Analogamente, k_{in} significa grau de entrada.

Cada iteração do algoritmo realiza uma das seguintes alterações na rede:

- **Criação de vértice com grau de saída = 1**. Com probabilidade α é criado um vértice v juntamente com uma aresta de v para um vértice pré-existente w, onde w é escolhido de acordo com $k_{in} + \delta_{in}$. O vértice v é incluído no módulo de w.
- **Criação de vértice com grau de entrada = 1**. Com probabilidade γ é criado um vértice w juntamente com uma aresta de um vértice existente v para w, onde v é escolhido de acordo com $k_{out} + \delta_{out}$. O vértice w é incluído no módulo de v.
- **Criação de uma aresta**. Com probabilidade β é criada uma aresta de um vértice existente v para um vértice existente w, v escolhido de acordo com $k_{out} + \delta_{out}$ e w escolhido de acordo com $k_{in} + \delta_{in}$. Com probabilidade $1 - \mu$, w é escolhido dentre os vértices do mesmo módulo que v; com probabilidade μ, w é escolhido dentre os vértices de módulos adjacentes ao módulo de v segundo a arquitetura.

Como neste modelo a rede é orientada, pode-se considerar separadamente uma distribuição dos graus de entrada e uma distribuição dos graus de saída. Da mesma forma que no modelo LFR, essas distribuições seguem leis de potência, com expoentes X_in e X_out, respectivamente. Pode-se demonstrar analiticamente que, quando N tende a infinito, os expoentes podem ser calculados pelas seguintes expressões [2]:

$$X_{in} = 1 + \frac{1+\delta_{in}(\alpha+\gamma)}{\alpha+\beta}$$

$$X_{out} = 1 + \frac{\beta+\gamma}{1+\delta_{out}(\alpha+\gamma)}$$

Discussão

Este modelo supera as limitações encontradas no modelo LFR: as redes são orientadas, nem todos os vértices são necessariamente ligados a vértices de outros módulos e é possível restringir as dependências entre módulos através da arquitetura fornecida como parâmetro. Além disso ele é um modelo evolutivo: o algoritmo pode ter como ponto de partida uma rede existente e então expandi-la criando mais vértices e arestas. A desvantagem em relação ao modelo LFR é o controle reduzido sobre a rede: não há como impor restrições sobre o grau máximo, sobre a distribuição dos tamanhos dos módulos e nem estabelecer limites de tamanho para os módulos.

3. EXPERIMENTO

Realizamos um experimento a fim de avaliar os modelos de redes de acordo com a sua capacidade de gerar redes semelhantes a redes de sistemas reais. Para isso consideramos dois sistemas, o jogo VilloNanny 2.2.4 e o programa IRPF 2009, e ajustamos os parâmetros dos modelos de acordo com métricas extraídas desses sistemas. O experimento foi dividido em cinco etapas: extração das redes dos sistemas (redes reais), análise das redes reais, sintonia dos parâmetros dos modelos, geração de redes sintéticas e comparação entre redes sintéticas e as redes reais correspondentes.

3.1 Extração

Os sistemas analisados são implementados na linguagem Java e foram distribuídos em diversos arquivos JAR, cada arquivo contendo várias classes. Alguns arquivos JAR contêm classes específicas de um sistema, mas muitos deles são bibliotecas; consideramos que as bibliotecas que um sistema usa são parte do sistema, e cada arquivo JAR corresponde a um módulo arquitetural. Essa é uma aproximação razoável, uma vez que arquivos JAR distintos são, em geral, desenvolvidos por equipes diferentes e distribuídos independentemente.

A ferramenta DepFind[1] foi usada para extrair, através da análise estática dos arquivos JAR, a rede de dependências entre as classes. A organização em módulos foi extraída a partir da enumeração das classes contidas em cada arquivo JAR.

3.2 Análise de sistemas reais

O grau de um vértice é o número de arestas que estão ligadas a ele. Baseado em pesquisas anteriores, consideramos que a distribuição estatística dos graus dos vértices segue aproximadamente uma lei de potência, $P(k)\ k^{-\gamma}$, onde $P(k)$ é a probabilidade de um vértice escolhido possuir grau k e γ é o expoente da distribuição. Usamos esse mesmo tipo de distribuição para modelar a distribuição dos tamanhos dos módulos.

Para cada rede foram coletadas diversas métricas da teoria dos grafos e da teoria das redes complexas:

- número de vértices - ...
- número de arestas - ...
- número de arestas externas - número de arestas que ligam vértices em módulos distintos.
- grau médio - ...
- grau máximo - ...
- expoente da distribuição de graus - ...
- número de módulos
- tamanho do menor módulo
- tamanho do maior módulo
- expoente da distribuição dos tamanhos dos módulos

[1] `http://depfind.sourceforge.net/`

Os expoentes das distribuições de graus e de tamanhos dos módulos foram estimados através da técnica de máxima verossimilhança [3] através de uma implementação disponível disponível na Internet[2].

Extraímos ainda a rede de dependências entre MÓDULOS...

3.3 Sintonia dos parâmetros dos modelos

Dado um sistema e um modelo, os parâmetros do modelo foram escolhidos de forma a gerar redes cujas métricas fossem próximas às metricas da rede do sistema.

No modelo de Lancichinetti os parâmetros correspondem às métricas.

No outro modelo isso foi feito na tentativa e erro (oops, experimentalmente).

Distribuição de graus foi usada para todos os modelos.

Modelo de Lancichinetti tem os seguintes parâmetros.

No modelo de Bollobás os parâmetros não correspondem a métricas da rede resultante, e por isso foi preciso sintonizar os parâmetros de forma experimental (correspondência analítica existe, mas não consegui usar).

3.4 Geração de sistemas sintéticos

Foram gerados 10 redes para cada modelo. Por que 10?

3.5 Comparação com os sistemas reais

Usamos a métrica de distância entre redes definida por [1], implementação de Charles. Essa distância leva em consideração aspectos locais das redes. Comparar apenas os parâmetros globais não é tão bom porque não diferencia entre modelos, já que existem vários modelos que, como se sabe, geram power law.

Consideramos para cada modelo a média entre as distâncias de cada rede gerada pelo modelo.

4. RESULTADOS

TABELA 1: sistemas analisados. Nome, versão, métricas (tamanho, número de módulos, ...)

TABELA 2: métricas das redes sintéticas.

TABELA 3 (ou GRÁFICO): distâncias

5. DISCUSSÃO

O quão bem o modelo funciona, vantagens e desvantagens em relação a outras abordagens (complementares).

Especulação sobre o papel da modelagem estatística na engenharia de software.

Focar na hipótese: redes sintéticas com parâmetros ajustáveis dão insights sobre as ferramentas de engenharia reversa / evolução de software.

[2] `http://www.santafe.edu/~aaronc/powerlaws/`

Trabalhos futuros: explorar métricas de arquitetura, avaliar algoritmos de clustering, considerar outros modelos.

6. REFERENCES

[1] R. F. S. Andrade, J. G. V. Miranda, S. T. R. Pinho, and T. P. Lobão. Measuring distances between complex networks. *Physics Letters A*, 372(32):5265–5269, August 2008.

[2] B. Bollobás, C. Borgs, J. Chayes, and O. Riordan. Directed scale-free graphs. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 132–139, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[3] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data, 2007.

[4] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 78(4), 2008.