Modelos Estocásticos para Síntese de Sistemas de Software Organizados em Módulos

Rodrigo Rocha Gomes e Souza

ABSTRACT

Existem muitos algoritmos para identificar os módulos de um sistema a partir de sua implementação, mas poucos estudos empíricos sobre esses algoritmos. Isso ocorre porque é difícil encontrar sistemas cuja arquitetura modular é bem documentada. Propomos suprir essa escassez através da geração automática de sistemas de *software* sintéticos, viabilizando análises empíricas em larga escala. Neste artigo apresentamos e avaliamos três modelos estocásticos que cumprem esse papel.

1. INTRODUÇÃO

A divisão conceitual de um sistema de *software* em módulos, denominada arquitetura modular, é muito importante no gerenciamento da evolução do sistema [6]. À medida que os desenvolvedores de um sistema são substituídos, no entanto, essa informação pode se perder. Alguns pesquisadores propõem o uso de algoritmos de aglomeração (*clustering*) para recuperar a arquitetura modular de um sistema a partir de sua implementação.

Espera-se que os algoritmos encontrem organizações modulares semelhantes àquelas que seriam encontradas por especialistas nos sistemas analisados. Uma forma de validar os algoritmos consiste, pois, em aplicá-lo ao código-fonte de um sistema cuja organização em módulos seja conhecida e então comparar os módulos do sistema com os módulos encontrados pelo algoritmo através de uma medida de similaridade entre particionamentos (cite mojo, edgemojo etc.). Infelizmente é difícil encontrar sistemas documentados nesse nível de detalhe, e por essa razão a avaliação empírica dos algoritmos de recuperação de arquitetura ainda é incipiente.

Propomos uma abordagem de avaliação complementar, baseada na produção 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 submeter os algoritmos a amostras arbitrariamente grandes de sistemas arbitrariamente complexos e, sobretudo, estudar como o desempenho dos algoritmos é afetado por parâmetros que definem os sistemas sintéticos.

A implementação de um sistema possui muitos detalhes, e nem todos os detalhes são relevantes para se compreender sua organização modular. Para recuperar os módulos de um sistema orientado a objetos é comum considerar apenas 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. (Neste estudo consideramos apenas sistemas orientados a objetos, mas os conceitos podem ser igualmente aplicados a outros paradigmas de programação.)

Na próxima seção apresentamos três modelos estocásticos que podem ser usados para gerar sistemas sintéticos. Na seção 3 descrevemos um experimento para avaliar os modelos quanto à capacidade de gerar sistemas semelhantes a sistemas reais. Os resultados do experimento são apresentados na seção 4. Na seção 5 discutimos os resultados obtidos e os trabalhos futuros.

2. MODELOS DE GERAÇÃO DE REDES COMPLEXAS

Pesquisas recentes mostram que redes de dependências entre classes possuem características comuns a redes complexas estudadas em diversos domínios, tais como sociologia, biologia e linguística [5, 7]. Foram desenvolvidos diversos modelos que simulam a construção de redes com tais características [1, 4]; esses modelos podem ser usados, portanto, para gerar redes de dependências entre classes.

2.1 Modelo ER com Módulos

O modelo de Erdős-Rényi [3], ou modelo de rede aleatória, gera redes não-orientadas cujo número de arestas por vértice segue uma distribuição de Poisson. Ele não modela adequadamente as redes de dependências entre classes, e por isso é usada apenas como base de comparação. Propomos aqui uma simples extensão desse modelo que considera a organização dos vértices em módulos. O modelo possui os seguintes parâmetros:

- número de vértices, $|V|$;
- probabilidade de ligação, p;

- número de módulos, C.

O algoritmo de geração de redes inicia-se com a criação de $|V|$ vértices. A seguir cada vértice é incluído em um módulo escolhido com probabilidade uniforme. Por fim, para cada par de vértices, é adicionada uma aresta, com probabilidade p.

2.2 Modelo LFR

Lancichinetti, Fortunato e Radicchi [4] criaram um modelo de rede com estrutura de módulos embutida. O modelo LFR não foi baseado em nenhum domínio em particular, mas incorpora distribuições estatísticas 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. Mais precisamente, o número de vértices cujo grau é k é proporcional a $k^{-\gamma}$, onde o expoente γ tipicamente varia entre 2 e 3, e o número de módulos com n vértices é proporcional a n^{β}, com β variando entre 1 e 2.

O modelo possui os seguintes parâmetros:

- número de vértices, $|V|$;
- expoente da distribuição de graus, γ;
- grau médio, $\langle k \rangle$;
- grau máximo, k_{max};
- expoente da distribuição de tamanhos dos módulos, β;
- número mínimo de vértices por módulo, T_{min};
- número máximo de vértices por módulo, T_{max};
- parâmetro de mistura, μ.

O algoritmo gera redes que seguem aproximadamente os parâmetros fornecidos. O parâmetro de mistura indica a fração das arestas de cada vértice que são compartilhadas com vértices de outro módulo. Por exemplo, se $\mu = 0,4$ e o vértice v pertence ao módulo M, então 40% das suas arestas estão ligadas a vértices que não pertencem ao módulo M. Arestas que ligam vértices de módulos distintos são chamadas de *arestas externas*.

Discussão

O modelo gera grafos não-orientados, uma representação muito simplificada das redes de dependências entre classes. 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 O modelo XXX

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 [1], que foi inspirado na rede de *links* entre páginas da *web*. O modelo possui os seguintes parâmetros:

- número de vértices, $|V|$;
- arquitetura modular, A;
- probabilidades p, q e r, tal que $p + q + r = 1$;
- probabilidade μ;
- δ_{in} e δ_{out}.

A arquitetura modular é uma rede que representa as dependências permitidas entre módulos. Dois vértices da rede gerada só podem ser ligados se os módulos correspondentes na arquitetura forem ligados por uma aresta.

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 do algoritmo apresentada 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 p 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 q é criado um vértice w juntamente com uma aresta de um vértice pré-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 r é criada uma aresta de um vértice pré-existente v para um vértice pré-existente w, v escolhido de acordo com $k_{out} + \delta_{out}$ e w escolhido de acordo com $k_{in} + \delta_{in}$. A escolha de w não é livre. 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 γ_{in} e γ_{out}, respectivamente. Pode-se demonstrar analiticamente que, quando $|V|$ tende a infinito, os expoentes podem ser calculados pelas seguintes expressões [1]:

$$\gamma_{in} = 1 + \frac{1 + \delta_{in}(p+q)}{p+r} \quad (1)$$

$$\gamma_{out} = 1 + \frac{r+q}{1 + \delta_{out}(p+q)} \quad (2)$$

Discussão

Este modelo supera as limitações encontradas no modelo LFR: as redes são orientadas, nem todos os vértices são 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

Na seção anterior discutimos características gerais dos modelos de geração de redes. Com o objeto de avaliar a capacidade dos modelos de gerar redes semelhantes a redes de dependências entre classes, realizamos um experimento com os três modelos e dois sistemas, o jogo VilloNanny 2.2.4 e o programa IRPF 2009. O experimento foi dividido em cinco etapas: extração das redes dos sistemas (redes reais), análise das redes reais, escolha dos parâmetros dos modelos, geração de redes sintéticas e comparação entre redes sintéticas e as redes reais correspondentes.

3.1 Extração

Os sistemas analisados foram implementados na linguagem Java e distribuídos em diversos arquivos JAR, cada arquivo contendo várias classes. Para construir a rede de um sistema, consideramos que cada arquivo JAR é um módulo arquitetural. Essa é uma aproximação razoável, uma vez que arquivos JAR distintos são, em geral, desenvolvidos por equipes diferentes. (Naturalmente, o desafio de um algoritmo de clustering é descobrir os módulos *dentro* de cada arquivo JAR.) A extração das dependências entre as classes foi realizada pela ferramenta DepFind[1].

3.2 Análise de sistemas reais

De cada sistema foram extraídas as seguinte métricas:

- número de vértices, $|V|$;
- número de arestas, $|E|$;
- número de arestas externas $|E_{ext}|$;
- grau médio, $\langle k \rangle$;
- grau máximo, k_{max};
- expoente da distribuição de graus, γ;
- número de módulos, $|M|$;
- tamanho do menor módulo, t_{min};
- tamanho do maior módulo, t_{max};
- expoente da distribuição dos tamanhos dos módulos, β

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

Além disso, extraímos a arquitetura modular do sistema. Na arquitetura, existe uma aresta de um módulo A para um módulo B somente se pelo menos um vértice de A se liga a um vértice de B.

3.3 Escolha dos parâmetros dos modelos

Na discussão a seguir, X^s e X^m representam, respectivamente, o valor de uma métrica X extraída de um sistema e o valor de um parâmetro X de um modelo.

Para o modelo de rede aleatória com módulos, consideramos $|V|^m = |V|^s$ e $|M|^m = |M|^s$, isto é, mantivemos o número de vértices e o número de módulos do sistema. O valor da probabilidade de ligação, p, foi fixado em $\frac{2|E|^s}{|V|^s(|V|^s-1)}$; desta forma espera-se que o número de arestas da rede gire em torno de $|E|^s$.

Para o modelo LFR a maioria dos parâmetros foi fixada de acordo com as métricas do sistema. O parâmetro de mistura, μ foi definido de acordo com a expressão $|E_{ext}|^s/|E|^s$.

Para o modelo XXX a arquitetura modular fornecida como parâmetro é igual à arquitetura extraída do sistema, número de vértices idem e calculamos $p, r, q, \delta_{in}, \delta_{out}$ de forma a satisfazer às seguintes restrições:

- $q = p$. Como no estudo consideramos redes não-orientadas, não faz sentido distinguir os dois casos em que há a criação de um vértice e uma aresta, já que apenas o sentido da aresta é alterado.
- O número de arestas do modelo deveria ser igual ao número de arestas do sistema
- O grau máximo ...
- O tamanho da maior comunidade ...

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

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

3.4 Geração de sistemas sintéticos

Como os modelos são estocásticos, é insuficiente gerar apenas uma rede de cada modelo. Para realizar uma análise mais representativa, consideramos uma amostra de X redes para cada modelo. (Por que X? Ver calculadora de tamanho de amostra [3])

3.5 Comparação com os sistemas reais

Estatística de Kolmogorov-Smirnov (+qqplot pra ilustrar alguns casos) para comparar... * distribuição de graus * distribuição de coeficiente de clustering * distribuição do número de arestas externas por vértice * distribuição do número de módulos vizinhos por vértice * distribuição dos tamanhos dos módulos

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

Este artigo apresentou uma nova abordagem, baseada na geração de sistemas sintéticos, de avaliação de algoritmos de aglomeração aplicados à recuperação de módulos de sistemas de *software*. Três modelos de sistemas sintéticos foram apresentados e avaliados experimentalmente quanto à capacidade de produzir sistemas realistas.

ESPECULAÇÃO SOBRE O PAPEL DA MODELAGEM ESTATÍSTICA NA ENGENHARIA DE SOFTWARE...

Acreditamos que outras linhas de pesquisa da engenharia de *software*, a exemplo da análise de impacto e da localização de funcionalidades, podem se beneficiar do tipo de avaliação apresentado neste artigo. Para isso é preciso ampliar os modelos aqui discutidos com os conceitos de cada linha de pesquisa.

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

6. REFERENCES

[1] 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.

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

[3] P. Erdös and A. Rényi. On random graphs, i. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[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.

[3] `http://www.surveysystem.com/sscalc.htm`

[5] C. R. Myers. Software systems as complex networks: structure, function, and evolvability of software collaboration graphs. *Phys Rev E Stat Nonlin Soft Matter Phys*, 68(4 Pt 2):046116, Oct 2003.

[6] D. L. Parnas. On the criteria to be used in decomposing systems into modules. *Commun. ACM*, 15(12):1053–1058, 1972.

[7] S. Valverde and R. V. Solé. Hierarchical small worlds in software architecture. (Directed Scale-Free Graphs), 2003.