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

Para recuperar os módulos de um sistema 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.

2.1 Modelos de Geração de Redes Complexas

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

O modelo de Lancichinetti não foi feito baseado em nenhum domínio em particular. Parâmetros são métricas da rede que se quer obter. Não-orientado.

O modelo X...

3. O MODELO NOVO

Valorizar o modelo novo. Em relação a Bollobas, ele tem módulos embutidos. Em relação a Lancichinetti, pode especificar a arquitetura, mixing é variável, o grafo da arquitetura não é necessariamente completo.

4. 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.

4.1 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.

4.2 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

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

Extraímos ainda a rede de dependências entre MÓDULOS...

[1] `http://depfind.sourceforge.net/`
[2] `http://www.santafe.edu/~aaronc/powerlaws/`

4.3 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).

4.4 Geração de sistemas sintéticos

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

4.5 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.

5. RESULTADOS

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

TABELA 2: métricas das redes sintéticas.

TABELA 3 (ou GRÁFICO): distâncias

6. 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.

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

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