# 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 uso de algoritmos de agrupamento (*clustering*) com a finalidade de detectar módulos arquiteturais — grupos coesos de entidades de código-fonte (classes ou funções).

Uma das formas de avaliar um algoritmo de agrupamento no contexto da recuperação de arquitetura consiste em aplicar o algoritmo a um sistema cuja estrutura de módulos seja conhecida. Infelizmente não existem muitos sistemas cuja estrutura de módulos é bem documentada. OS ESTUDOS FEITOS ASSIM NÃO SÃO CONTUNDENTES, OBTÊM RESULTADOS DIFERENTES PARA SISTEMAS DIFERENTES E NÃO TÊM NENHUMA PISTA SOBRE O QUE FAZ UM ALGORITMO SER BOM EM UM SISTEMA E RUIM EM OUTRO.

Propomos sistemas de software gerados por computador, COM ESTRUTURA DE MÓDULOS EMBUTIDA. VANTAGENS: É POSSÍVEL AJUSTAR PARÂMETROS DOS SISTEMAS GERADOS E ASSIM GANHAR INSIGHT SOBRE OS PARÂMETROS QUE INFLUENCIAM A ACURÁCIA DE UM ALGORITMO. EM PARTICULAR, É POSSÍVEL CONTROLAR O TAMANHO DO SOFTWARE GERADO E, ASSIM, GERAR SISTEMAS GRANDES, MAIS PARECIDOS COM OS QUE SERIAM ANALISADOS PELAS TÉCNICAS EM UM CENÁRIO REAL.

O modelo é comparado com outros modelos disponíveis na literatura sobre redes complexas.

## 2. TEORIA

Neste estudo nos concentraremos em sistemas orientados a objeto, para simplificar, mas os conceitos provavelmente podem ser aplicados a outros paradigmas como procedimental ou funcional. Muitas técnicas de recuperação de arquitetura analisam uma representação abstrata de sistemas de software, as redes de dependências entre classes. Nessas redes, os vértices representam classes e existe uma aresta do vértice A para o vértice B se a classe A depende da classe B para funcionar corretamente. Essa dependência pode ser resultado diversos tipos de interação entre as classes: A estende B, um método de A chama um método de B etc. A extração da rede de um sistema de software se dá através da análise estática de seu código-fonte ou do código objeto.

Neste estudo consideramos dois sistemas de software reais e três modelos de geração de redes complexas.

Neste estudo foram avaliados três modelos de geração de redes complexas: o modelo de configuração, o modelo de Bollobás [2], o modelo de Lancichinetti, Fortunato e Radicchi [4] e um novo modelo, baseado no modelo de Bollobás. Os dois últimos modelos geram redes com uma estrutura de módulos embutida.

DESCRIÇÃO DE CADA MODELO

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.

## 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

O estudo experimental foi dividido em cinco etapas: extração das redes de software reais, análise de sistemas reais, sintonia dos parâmetros dos modelos, geração de sistemas sintéticos e comparação entre redes sintéticas e redes reais.

### 4.1 Extração

Os sistemas analisados são escritos em Java e distribuídos em diversos arquivos JAR. Alguns arquivos JAR representam código do próprio sistema e outros são bibliotecas us-

adas por eles. A ferramenta DepFind[1] foi usada para extrair (estaticamente) todas as interações entre classes, métodos e atributos dos sistemas analisados. Foi considerado o sistema + as bibliotecas. A seguir um programa feito em casa abstraiu para dependências classe para classe (lifting). Consideramos que cada JAR é um módulo arquitetural.

## 4.2 Análise de sistemas reais

A rede resultante foi submetida a diversas análises em que foram coletadas métricas da teoria dos grafos e da teoria das redes complexas:

número de classes expoente da distribuição de graus número de módulos expoente da distribuição dos tamanhos dos módulos ...

Para o ajuste do expoente ..., usamos maximum likelihood estimation etc. [3]. Usamos a implementação disponível em X[2].

## 4.3 Sintonia dos parâmetros dos modelos

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.

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 (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. incluir pesos das arestas nas análises e nos modelos.

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

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

# 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] 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.