Heurísticas para Identificação de Ambiguidade de Autores em Projetos *Open Source*

Talita S. Orfanó, Kecia A. M. Ferreira, Mariza A. S. Bigonha

Programa de Pós-Graduação
Ciência da Computação
UFMG

Contexto

- Popularização de Projetos *Open Source*
- Mineração de dados em repositórios públicos
  - GitHub, BitBucket, GitLab
- Diversos estudos analisam o comportamento dos responsáveis pelas contribuições nos projetos
  - **Autores** ou **contribuidores**

Problema

- Contribuidores ambíguos

Talita Orfanó
torfano@dominio2.com.br

Talita Santana Orfanó
talita.orfano@dominio1.com

Problema

- Identificar a ambiguidade é uma tarefa difícil
- Não existe padrão universal para criação de prefixos de e-mail
- Ameaça importante à validade de estudos que se baseiam nessa informação

Wiese et al. (2016)

Problema

- Wiese et al. (2016) compararam heurísticas para identificar múltiplos endereços de e-mail pertencentes a um mesmo membro
  - Listas de discussão de e-mail
- Pouco estudo relacionado a assertividade dessas heurísticas no contexto exclusivo de repositórios GitHub

Heurísticas

- Bird et al. (2006)
- Canfora et al. (2011)
- Robles & Gonzales-Barahona (2005)
- Goeminne & Mens (2011)
- Kounters et al. (Simples) (2011)

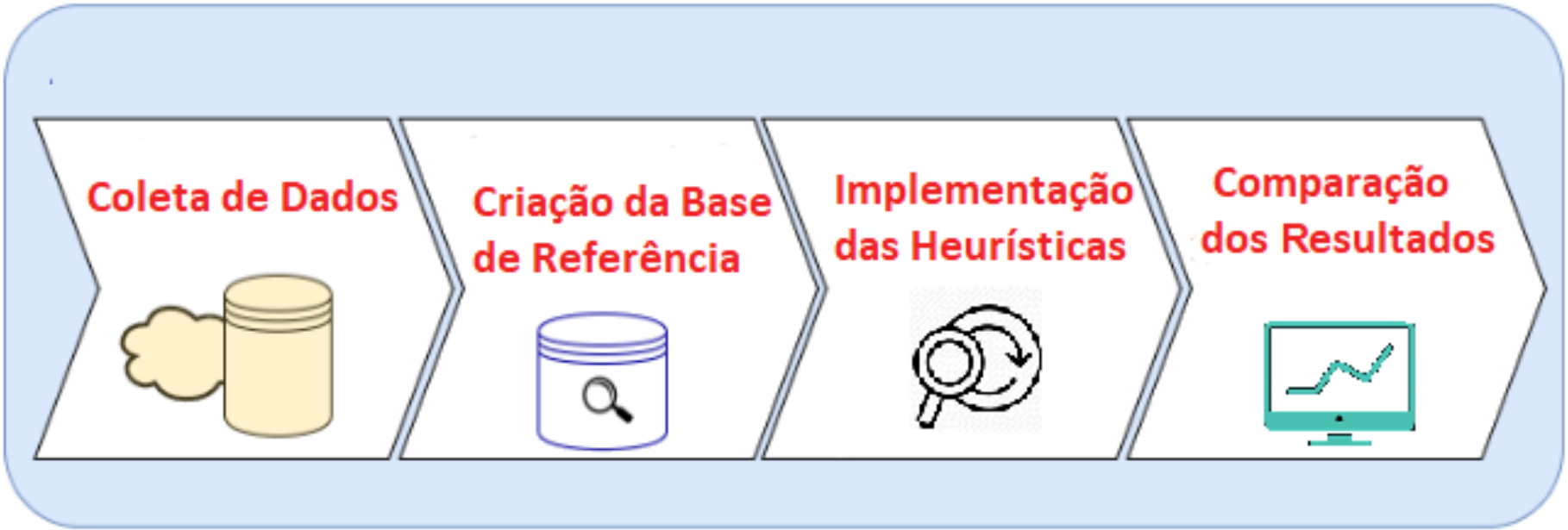
Objetivo

Analisar de forma comparativa cinco heurísticas de detecção de ambiguidades, a partir de contribuidores de projetos do GitHub

Coleta de Dados

- Joda-Time: 64 autores, 3.252 estrelas
- PowerShell: 162 autores, 8.447 estrelas
- ElasticSearch: 908 autores, 26.884 estrelas

Escopo


Coleta de Dados
Criação da Base de Referência
Implementação das Heurísticas
Comparação dos Resultados

Bird et al. (2006)

- As duas identidades <nome, e-mail> são consideradas como pertencentes a um mesmo autor se a similaridade de *Levenshtein* for maior ou igual a *t*

- Foram escolhidos arbitrariamente seis valores para *t*: 0,84; 0,87; 0,90; 0,93; 0,96 e 0,99.

Bird et al. (2006)

- Similaridade entre o nome completo >= *t*
- Similaridade entre o primeiro nome >= t e similaridade entre último nome >= *t*
- Similaridade entre prefixo de e-mail >= *t*
- Similaridade entre o prefixo de e-mail com derivações do primeiro e último nome >= *t*

Canfora et al. (2011)

Comparação de nomes

- Ignora o nome do meio
  - Ex: John Smith é igual a John Kennedy Smith
- Compara nome com as iniciais do outro
  - Ex: j k s é igual a John Kennedy Smith
- Compara apenas o último nome
  - Ex: John Smith é igual a Smith
- E outras derivações originárias do nome dos contribuidores

Canfora et al. (2011)

Comparação de e-mails

- Compara prefixos
- Compara nome extraído a partir do prefixo
  - john.smith é igual a John Smith
- E outras informações oriundas do prefixo e comparadas com o nome dos autores

Simple - Kouters et al. (2011)

- Verifica se o nome ou prefixo do e-mail contém parte (de tamanho l) do segundo nome e o inverso também
- Nessa heurística também foram escolhidos arbitrariamente cinco valores para l, variando de 4 a 8
- Exemplos:

<John Smith, johnsmith@domainA> e <Jonathan Smith,jsmith@domainB>
<Maria Silva, maria.s@domainA> e <Mariana Santos,mariana.s@domainB>

Robles et al. (2005)

- A partir do nome do desenvolvedor, cria-se um conjunto de possíveis prefixos de e-mail e compara com o prefixo de outro desenvolvedor
- Alguns exemplos de prefixos:
  - nome@dominio.com
  - nome_sobrenome@dominio.com
  - nome.sobrenome@dominio.com
  - nsobrenome@dominio.com
  - snome@dominio.com

Goeminne et al. (2011)

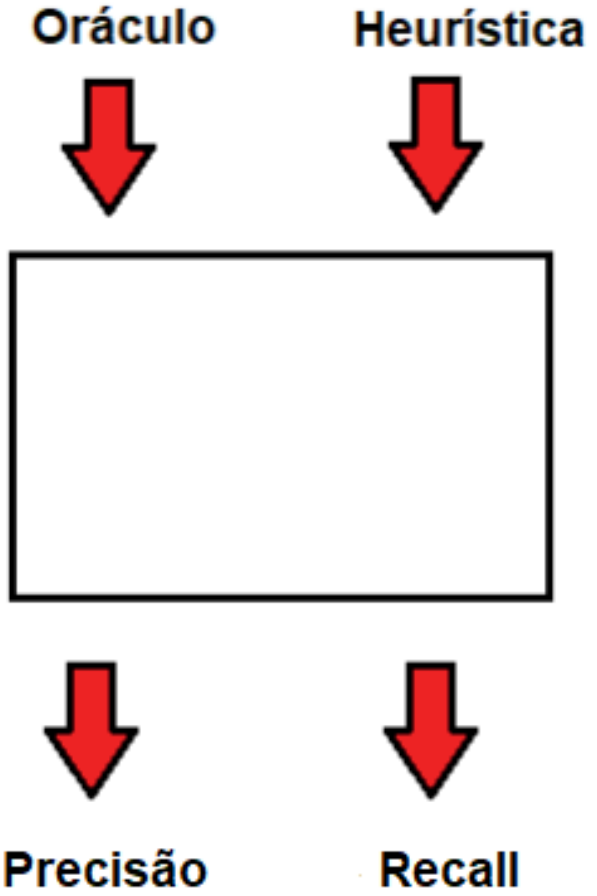
- Verifica a similaridade de Levenshtein entre os nomes ou prefixos de e-mail, se é maior ou igual a *t*
- Verifica se o prefixo de e-mail contém parte do nome de tamanho l
- Cria um conjunto de potenciais prefixos de e-mail, combinando todas as partes do nome e comparando com o prefixo de e-mail de outro desenvolvedor

Goeminne et. al.

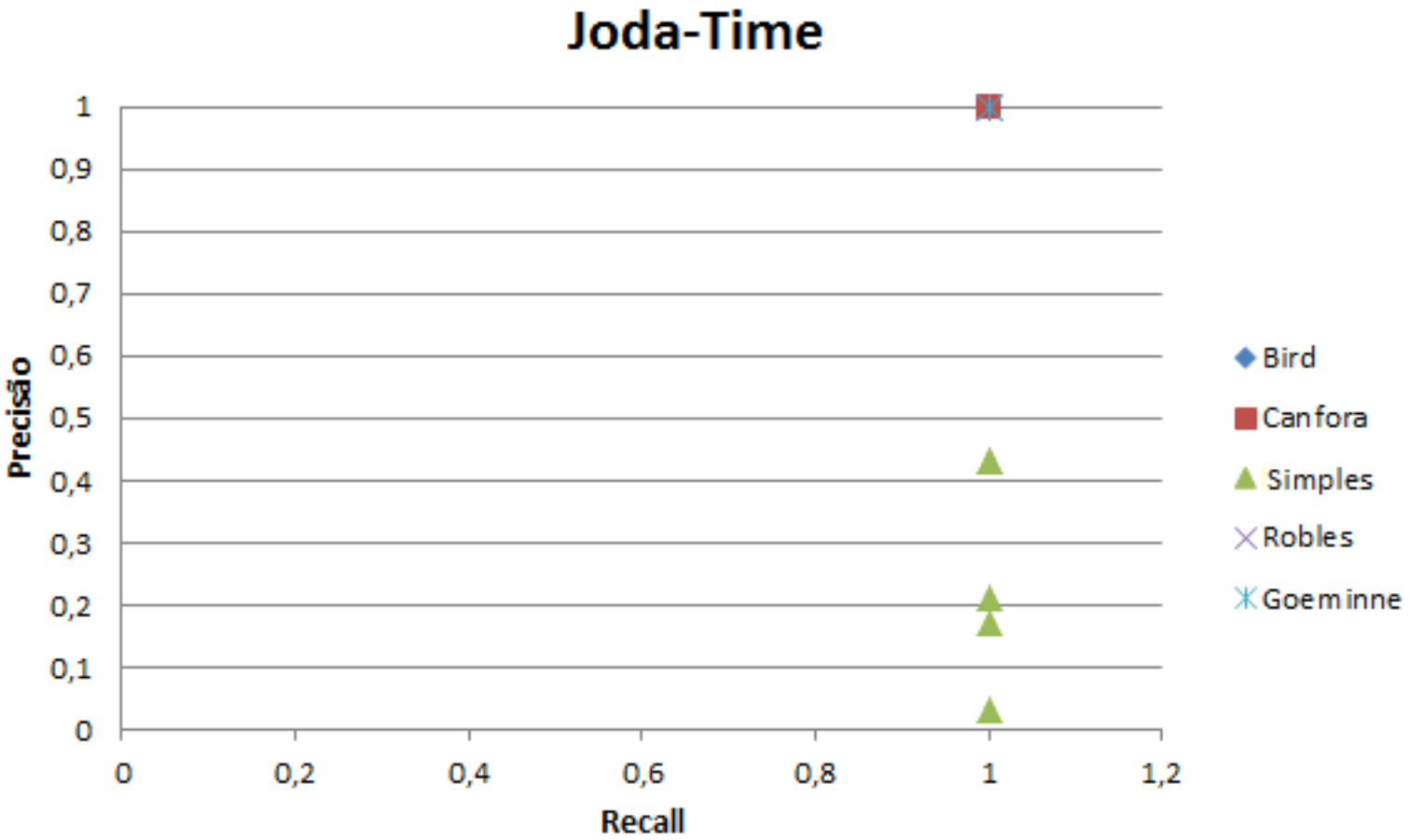
- Exemplos de prefixos:
  - nomesobrenome, nome.sobrenome, nome_sobrenome
  - nsobrenome, nome-sobrenome, nome+sobrenome
  - sobrenomenome, sobrenome_nome, sobrenome.nome
  - e assim por diante ...

Avaliação

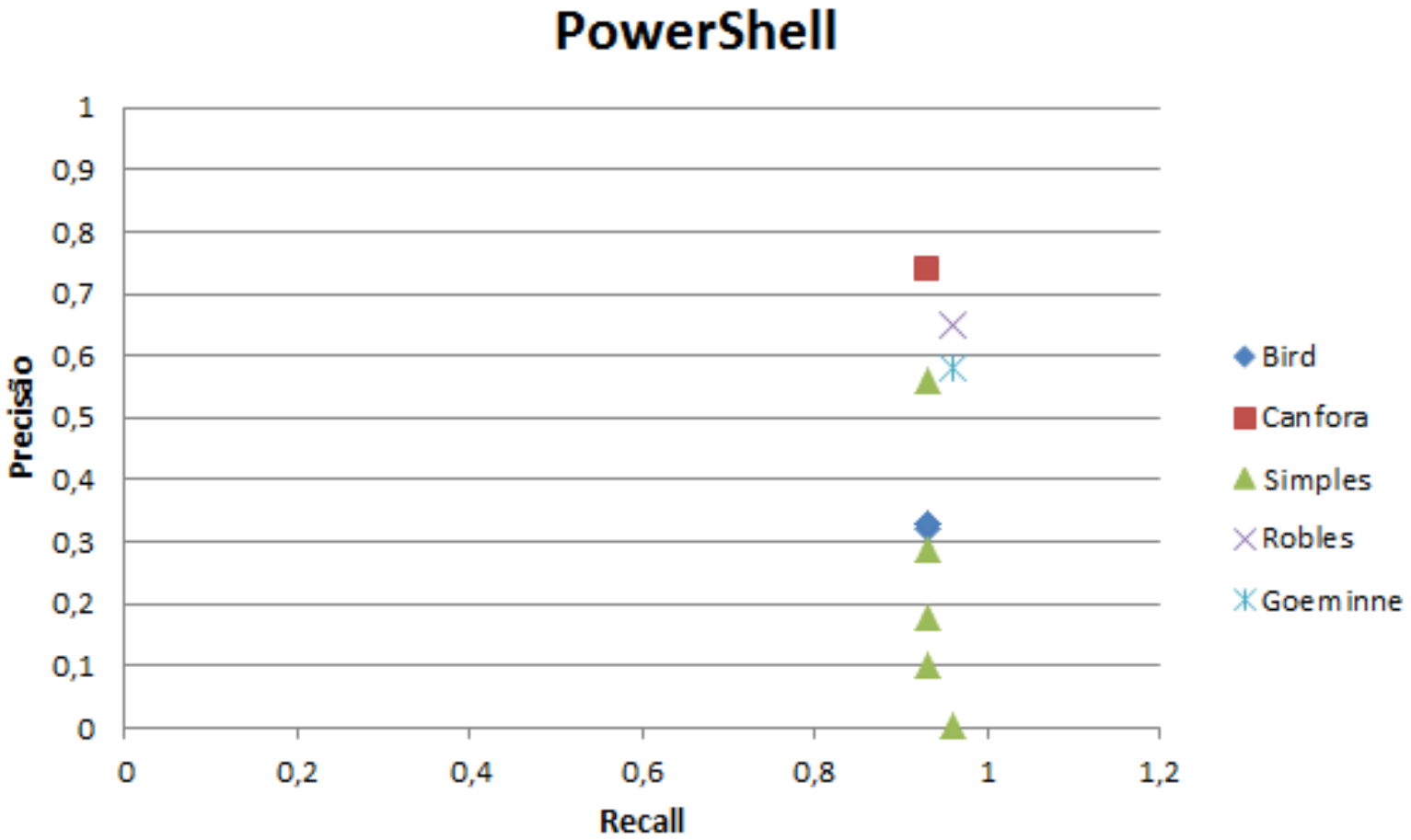
Comparação da base de referência construída com os autores identificados por cada heurística



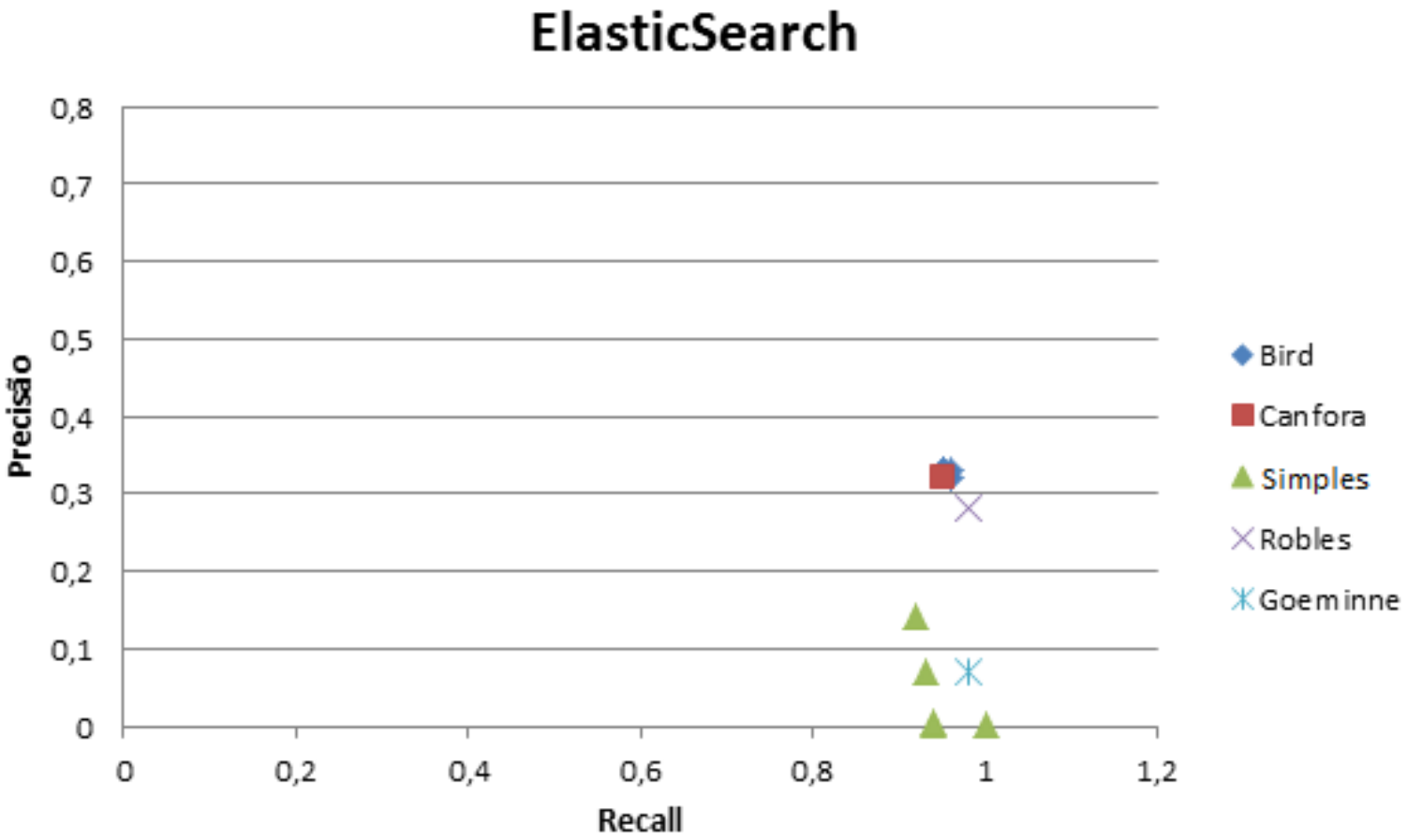
Resultados


Joda-Time
Precisão
1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
0
0,2
0,4
0,6
0,8
1
1,2
Recall
Bird
Canfora
Simples
Robles
Goeminne

Resultados


PowerShell
Precisão
1
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
0
0,2
0,4
0,6
0,8
1
1,2
Recall
Bird
Canfora
Simples
Robles
Goeminne

Resultados


ElasticSearch
Precisão
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
0
0,2
0,4
0,6
0,8
1
1,2
Recall
Bird
Canfora
Simples
Robles
Goeminne

Análise dos Resultados

- Precisão
  - Tende a reduzir proporcionalmente com o aumento do número de contribuidores no projeto
  - Joda-Time, pequeno porte: 100%
  - PowerShell, médio porte: 74%
  - ElasticSearch, grande porte: 33%

Análise dos Resultados

- Recall
  - Nos três projetos, todas as heurísticas obtiveram ótimos resultados
  - Raramente deixam de detectar alguma ambiguidade existente

Análise dos Resultados

- Examinando manualmente o resultado do ElasticSearch
- Observam-se muitas identidades unidas de forma equivocada devido ao prefixo idêntico das mesmas
  - Como os prefixos “github”, “contact” e “mail
  - “github@smithsrock.com” e “github@tobigue.de”

Análise dos Resultados

- Melhores heurísticas para se utilizar em um projeto de nível médio ou grande:
  - Bird, Canfora e Robles

- Projeto pequeno:
  - Também a heurística de Goeminne

Conclusão

- Importância da detecção de ambiguidade de autores em um projeto GitHub
  - Tratamento dos dados coletados
- As heurísticas tendem a detectar todas as ambiguidades do conjunto de dados em análise
- No entanto, quanto maior o número de contribuidores, menor a precisão das heurísticas

Trabalhos Futuros

- Analisar um conjunto maior de projetos do GitHub
  - Obter uma conclusão madura e precisa da melhor heurística a ser usada para cada caso

- Implementar uma heurística que diminua o número de falsos-positivos para softwares com um grande número de autores e ambiguidades

Heurísticas para Identificação de Ambiguidade de Autores em Projetos Open Source

Talita S. Orfanó, Kecia A. M. Ferreira, Mariza A. S. Bigonha
{talita.orfano, mariza}@dcc.ufmg.br, kecia@decom.cefetmg.br

Obrigada!

Programa de Pós-Graduação
Ciência da Computação
UFMG