Aspectos Temporais para Medir a Força da Colaboração no GitHub

Natércia A. Batista, Michele A. Brandão, Ana Paula C. da Silva, Mirella M. Moro

Universidade Federal de Minas Gerais (UFMG) – Belo Horizonte, MG – Brasil

{natercia,micheleabrandao,ana.coutosilva,mirella}@dcc.ufmg.br

Abstract. *Time is fundamental to understand interactions between individuals in a social network. Here, we consider the temporal aspect to measure the collaboration strength between GitHub developers. The results show the temporal aspect enables to build a more realistic collaboration graph.*

Resumo. *Tempo é fundamental para entender a interação entre indivíduos em uma rede social. Aqui, consideramos o aspecto temporal para medir a força da colaboração entre desenvolvedores no GitHub. Os resultados revelam que tal aspecto permite construir um grafo de colaboração mais realista.*

1. Introdução

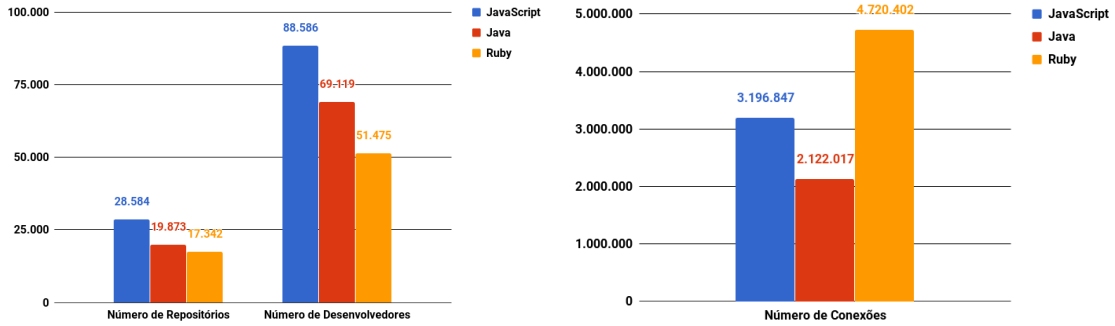
Social Coding é uma abordagem de desenvolvimento de software que propõe um ambiente colaborativo entre os desenvolvedores, encorajando a discussão e compartilhamento de ideias e conhecimento [Dabbish et al. 2012]. Essa metodologia tem alterado a forma de desenvolvimento de software e criado um novo tipo de rede de colaboração. Note que uma colaboração ocorre quando desenvolvedores contribuem em um mesmo repositório.

Trabalhos Relacionados. Estudos abordam diferentes propriedades e investigam comportamentos no GitHub [Tsay et al. 2014] (principal website de *social coding*), incluindo métricas que podem ser calculadas utilizando dados temporais e relacionamentos posteriores [Casalnuovo et al. 2015]. Porém, tais métricas são calculadas como propriedades de um projeto (repositório). Aqui, o foco está na colaboração entre pares de desenvolvedores e na melhoria de propriedades que utilizam dados temporais.

Contribuições. A medida da força dos relacionamentos pode ser utilizada para várias aplicações, como melhorar recomendação de desenvolvedores, avaliar a formação de times ou melhorar algoritmos de análise. Assim, nosso objetivo é encontrar propriedades semânticas da relação entre desenvolvedores e analisar como é possível utilizar essas propriedades para determinar a força do relacionamento entre eles. Para tal, analisamos a correlação entre métricas para a força dos relacionamentos. Essas métricas são propriedades topológicas e semânticas previamente propostas e propriedades semânticas que consideram aspectos temporais. Também discutimos como tais aspectos podem ser utilizados para modelar um grafo que representa a rede de colaboração.

2. Metodologia

Esta seção apresenta a base de dados e o modelo de rede utilizado para as análises, e a definição das métricas topológicas e semânticas para força dos relacionamentos.



(a) Repositórios e desenvolvedores (b) Conexões (links entre desenvolvedores)

Figura 1: Estatísticas para cada uma das linguagens no dataset

Base de Dados. A base provém do GHTorrent [Gousios 2013] e foi modificada e disponibilizada por Batista et al. [2017], que incluiu informações da rede do GitHub para as linguagens JavaScript e Ruby. Com a mesma metodologia de coleta, aqui também incluímos a linguagem Java. Batista et al. [2017] considera a data de início da colaboração entre dois desenvolvedores como a mais recente entre as datas dos primeiros commits de ambos em um repositório, e data de fim de colaboração como o último commit realizado no repositório por *qualquer* colaborador. Aqui, o fim da colaboração foi *alterado* para a menor data entre os últimos commits do *par* de desenvolvedores no repositório, o que permite limitar o relacionamento dentro da janela temporal de contribuição em comum.

Modelagem da Rede. A rede de desenvolvedores é representada por um grafo ponderado $\mathcal{G} = (\mathcal{V}, \mathcal{E})$: $\mathcal{V}$ é o conjunto de nós para os desenvolvedores, e $\mathcal{E}$ é o conjunto de arestas não direcionadas que conectam dois desenvolvedores quando ambos contribuem em um mesmo repositório. O peso das arestas corresponde aos valores que podem ser calculados a partir das métricas propostas por Alves et al. [2016] e Batista et al. [2017] (apresentadas a seguir). A Figura 1 (a) mostra a quantidade de repositórios e desenvolvedores presentes na rede para cada uma das linguagens, e a Figura 1 (b) apresenta a quantidade de conexões existentes em cada uma das redes. Apesar da linguagem Java possuir o segundo maior número de repositórios e desenvolvedores, tal rede não é tão densa quanto as demais. Por outro lado, a rede da linguagem Ruby possui as menores quantidades de repositórios e desenvolvedores, mas possui a maior quantidade de conexões.

Propriedades topológicas e semânticas. A Tabela 1 apresenta as características topológicas e semânticas para avaliar como desenolvedores se relacionam.

Propriedades semânticas com aspecto temporal. A Tabela 1 apresenta três propriedades semânticas que consideram o aspecto temporal dos relacionamentos propostas por Batista et al. [2017]. LPC e GPC calculam o tempo de contribuição entre um par de desenvolvedores em um dado repositório com variações local e global, respectivamente. Após as alterações na definição da data de fim de colaboração entre desenvolvedores, os resultados para tais métricas são alterados. O cálculo de LPC pode ser enviesado, principalmente em repositórios de curta duração como exemplificado na Tabela 2. O par de desenvolvedores AB contribuiu por menos tempo que o par CD, mas possui um valor superior para a métrica. Para entender a força de colaboração baseada no tempo para todos os pares de desenvolvedores, é proposta uma visão global das colaborações de toda a rede. A métrica GPC considera o tempo de contribuição máximo entre todos os pares

Tabela 1: Dados dois nós A e B, $\mathcal{N}(A)$ e $\mathcal{N}(B)$ são os vizinhos de A e B, $w(A)$ e $w(B)$ são o grau ponderado (soma do peso de cada aresta conectada ao nó), $w(A,B)$ o peso da aresta entre A e B, e $\mathcal{R}$ o conjunto de repositórios compartilhados por A e B.

Propriedades topológicas		
Métrica	**Definição**	**Interpretação**
Neighborhood Overlap (NO)	Mede a similaridade dos vizinhos entre dois nós por: $NO_{(A,B)} = \frac{\lvert\mathcal{N}(A)\rvert\cap\lvert\mathcal{N}(B)\rvert}{\lvert\mathcal{N}(A)\rvert\cup\lvert\mathcal{N}(B)\rvert}$	De acordo com [Easley and Kleinberg 2010], NO pode ser utilizada para calcular a força de links. Quanto maior o valor de NO, mais forte é o relacionamento.
Adamic-Adar Coefficient (AA)	Customizada para este contexto e calculada por: $AA_{(A,B)} = \frac{\sum \forall z \in \lvert\mathcal{N}(A)\rvert\cap\lvert\mathcal{N}(B)\rvert}{log\lvert\mathcal{N}(z)\rvert}$.	Vizinhos que não se relacionam com muitos outros recebem maior peso.
Preferential Attachment (PA)	Quanto maior o número de vizinhos de um nó, maior seu valor de PA calculado por: $PA_{(A,B)} = \lvert\mathcal{N}(A)\rvert\lvert\mathcal{N}(B)\rvert$.	De acordo com Barabási and Albert 1999, há uma relação linear entre o número de vizinhos de um nó e a probabilidade de conexão (i.e., "o rico fica mais rico").
Propriedades semânticas (Alves et al. [2015])		
Métrica	**Definição e Interpretação**	
Number of shared repositories (SR)	Refere-se ao número de repositórios compartilhados e é definida pela cardinalidade do conjunto $\mathcal{R}$ (i.e., $SR_{(A,B)} = \lvert\mathcal{R}\rvert$)	
Jointly developers contribution to shared repositories (JCSR)	Definida por: $JCSR_{(A,B)} = \frac{\sum_{\forall r_i \in \mathcal{R}} JCSR_{(A,B,r_i)}}{\lvert\mathcal{R}\rvert}$. Exemplo: considerando dois repositórios r_1 e r_2 em que r_1 é compartilhado apenas pelos desenvolvedores A e B, a contribuição conjunta do par AB em r_1 ($JCSR_{(A,B,r_1)}$) é igual a 1. Para r_2, considerando que é compartilhado pelos desenvolvedores A, B e C, a contribuição conjunta de A e B em r_2 ($JCSR_{(A,B,r_2)}$) é 0.66. Supondo que A e B compartilham apenas r_1 e r_2, a contribuição conjunta para eles é a média dos valores para cada repositório: ($JCSR_{(A,B)}$) = 0.83.	
Jointly developers commits to shared repositories (JCOSR)	Dado que $NC_{(A,r_j)}$ é o número total de commits realizados pelo desenvolvedor A no repositório r_j, $NC_{(B,r_j)}$ é o número total de commits realizados pelo desenvolvedor B no repositório r_j, e $NC_{(r_j)}$ é o número total de commits realizados no repositório r_j independente do desenvolvedor, então: $JCOSR_{(A,B)} = \sum_{\forall r_i \in \mathcal{R}} \frac{(NC_{(A,r_i)}+NC_{(B,r_i)})}{NC_{(r_i)}}$.	
Propriedades semânticas com aspecto temporal (Batista et al. [2017])		
Métrica	**Definição e Interpretação**	
Local Potential Contribution (LPC)	Considerando que $T_{(A,B,r_i)}$ é o intervalo de tempo em que ambos os desenvolvedores contribuíram em um repositório r_i definido pelas datas de início e fim de colaboração (representadas pelo primeiro e último commit em um repositório), esta métrica pode ser calculada por $LPC_{(A,B)} = \frac{\sum_{\forall r_i \in \mathcal{R}} \frac{T_{(A,B,r_i)}}{T_{(r_i)}}}{\lvert\mathcal{R}\rvert}$.	
Global Potential Contribution (GPC)	Para compensar o viés no cálculo da métrica LPC, Batista et al. [2017] estenderam para uma versão global definida como: $GPC_{(A,B)} = \frac{\sum_{\forall r_i \in \mathcal{R}} T_{(D_1,D_2,r_i)}}{\max_{\forall (D_1,D_2)\in \mathcal{D},\ r_i \in \mathcal{R}} T(D_1,D_2,r_i)}$. onde D é o conjunto de desenvolvedores total na rede, e $T_{(A,B,r_i)}$ é o intervalo de tempo em dois desenvolvedores contribuíram num repositório comum r_i.	
Previous Collaboration (PC)	Contabiliza as colaborações prévias entre dado par de desenvolvedores. Assim, no tempo t é definida por: $PC_{(A,B,t)} = \frac{\sum_{\forall r_i \in \mathcal{R}} \frac{1}{ND_{(r_i,t)}}}{\lvert\mathcal{R}\rvert}$, onde $ND_{(r_i,t)}$ é o número total de desenvolvedores que contribuem no repositório r_i no tempo t, antes do desenvolvedor B iniciar a colaboração em r_i. Altos valores da fração $1/ND_{(r_i,t)}$ indicam que A tem mais chances de trabalhar com B e vice-versa. Por exemplo, se há apenas duas pessoas em dado repositório, há apenas uma possibilidade de colaboração; em contrapartida, quanto mais colaboradores no mesmo repositório, mais opções de escolha e assim, menor é a possibilidade de colaboração com um desenvolvedor em particular. Ou seja, existe maior possibilidade de estabelecer uma conexão com um desenvolvedor se a atenção do mesmo não for dividida em muitas opções.	

de desenvolvedores como denominador a fim de normalizar o valor da métrica calculada. Para o exemplo anterior apresentado na Tabela 2, considerando que o maior tempo de contribuição da rede foi de 12 meses, os resultados são divididos por este denominador. Assim, a métrica GPC permite classificar a força potencial de contribuição em relação a todas as contribuições possíveis dentro da rede e por este motivo ela deve ser preferencialmente escolhida para o desenvolvimento das análises.

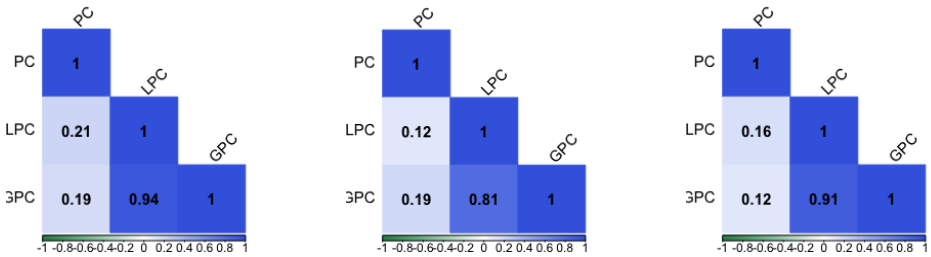
3. Resultados

Nesta seção, apresentamos a análise de correlação entre as propriedades com aspectos temporais e as demais existentes a fim de entender o relacionamento entre elas. Apresen-

Tabela 2: Exemplos de cálculo da LPC e GPC para dois pares de desenvolvedores.

Par de Desenvolvedores	Repositório	Duração Repositório	Tempo de Contribuição	LPC
Dev A - Dev B	R1	3 meses	3 meses	3/3 = 1,00
Dev C - Dev D	R2	12 meses	6 meses	6/12 = 0,50

Par de Desenvolvedores	Tempo de Contribuição	Maior Tempo de Contribuição da Rede	GPC
Dev A - Dev B	3 meses	12 meses	3/12 = 0,25
Dev C - Dev D	6 meses		6/12 = 0,50



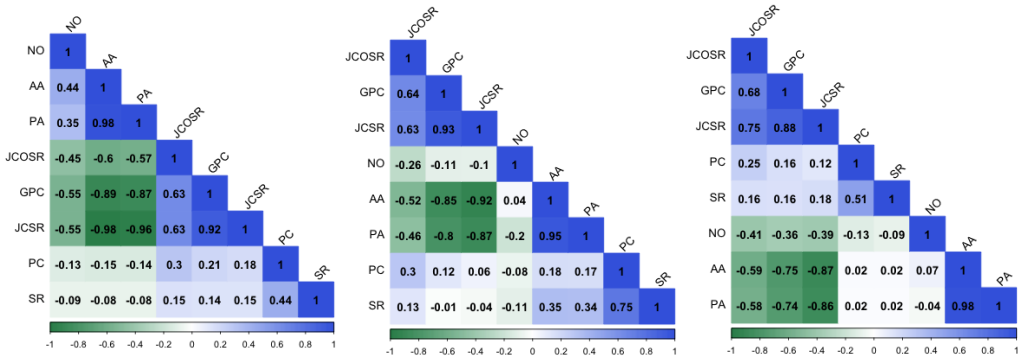
(a) JavaScript (b) Java (c) Ruby

Figura 2: Coeficiente de correlação de Spearman entre as novas propriedades com aspectos temporais: GPC, LPC e PC.

tamos resultados apenas para o coeficiente de correlação de Spearman, pois obtivemos resultados similares para o coeficiente de correlação de Pearson.

Análise das propriedades semânticas com aspecto temporal. As métricas GPC e LPC tendem a ser correlacionadas devido à sua forma de cálculo semelhante e às características das redes onde as mesmas foram aplicadas. A Figura 2 apresenta a correlação de Sperman para as três novas métricas. Conforme esperado, GPC e LPC são altamente correlacionadas monotonicamente. Em um modelo computacional para medir a força dos relacionamentos, a métrica escolhida seria a GPC por representar melhor a conexão do par de desenvolvedores no tempo em relação a todos os demais pares na rede. Além disso, a métrica PC não se relaciona com nenhuma das demais, independente do coeficiente de correlação. Este comportamento pode ser justificado por conta da janela temporal de colaboração. Para as métricas LPC e GPC, dois desenvolvedores só têm um valor de colaboração atribuído caso tenham trabalhado em um mesmo repositório na mesma janela temporal. Caso contrário, o valor tanto de LPC quanto de GPC para este par é igual a zero. No entanto, há uma aresta entre os dois desenvolvedores na rede, ainda que não tenham trabalhado na mesma janela temporal. Por isso, há um valor para PC calculado para tal par. Nesse cenário, a maioria dos pares de desenvolvedores tem valor zero para LPC e GPC e valor diferente de zero para PC, o que explica a não correlação de ambos. Assim, a não correlação entre tais métricas pode ser justificada pelos valores das arestas que não possuem interseção de tempo de contribuição, ou pode acontecer ainda que apenas os mesmos pares sejam considerados.

A fim de avaliar a correlação entre as métricas, realizamos a análise descartando as arestas em que não houve contribuição em uma mesma janela temporal. Após a remoção das arestas, a quantidade de pares de desenvolvedores é 16% da base de dados original para JavaScript, 27% para Java e 14% para Ruby. A análise das correlações para o coeficiente de Spearman e Pearson com os pares de desenvolvedores filtrados revela resultados



(a) JavaScript (b) Java (c) Ruby

Figura 3: Coeficiente de correlação de Spearman entre todas as métricas apresentadas

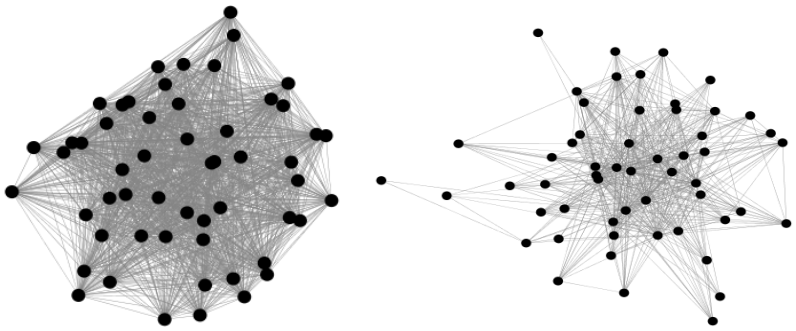
muito próximos aos da Figura 2. Assim, podemos considerar que a métrica PC gera uma informação de colaboração diferente das demais, fazendo com que seja importante calculá-la e considerá-la para medir a força de colaboração.

Correlação com propriedades existentes. Considerando as propriedades topológicas e semânticas existentes e as propriedades semânticas com aspectos temporais, a Figura 3 apresenta as correlações para o coeficiente de Spearman. Nas três linguagens de programação, a métrica GPC é altamente correlacionada à JCOSR. Desta forma, podemos afirmar que o tempo de contribuição entre desenvolvedores está bem relacionado com a quantidade de commits realizados pelo par de desenvolvedores em relação aos demais contribuidores em um mesmo repositório. Esta correlação é pertinente, uma vez que os commits de ambos os desenvolvedores são mais frequentes na janela temporal de trabalho entre eles. Além disso, a GPC também é bem correlacionada com a métrica JCSR, que avalia a contribuição de um par de desenvolvedores normalizada pela quantidade de desenvolvedores total nos repositórios. Em todas as correlações, as métricas SR e PC não estão fortemente correlacionadas a nenhuma outra. Em relação às métricas topológicas, AA e PA são fortemente correlacionadas em todos os cenários. Dessa forma, podemos considerar apenas uma entre elas em conjunto com a NO.

4. Aplicação

A baixa correlação entre LPC e GPC com as demais é devido à atribuição do valor zero a arestas entre desenvolvedores sem colaborações na mesma janela temporal de um repositório. Com isso, podemos utilizar o critério de janela temporal para a formação das arestas entre desenvolvedores na modelagem da rede. Para exemplificar a nova modelagem, selecionamos um repositório de JavaScript com um total de 55 colaboradores e 1.485 arestas, conforme Figura 4a. O repositório selecionado teve início em março de 2012 e término em setembro de 2015. Note que a janela temporal considera esse período de duração do repositório. Esta primeira rede apresenta densidade igual a 1 devido à formação de uma grande clique com conexões entre todos os desenvolvedores.

Aplicando os novos critérios temporais, a Figura 4b apresenta a rede do mesmo repositório, mas com as arestas geradas apenas entre desenvolvedores que colaboraram em uma mesma janela temporal. Desta forma, a quantidade de arestas na rede é reduzida para 571, ou seja, 38% da quantidade original. A densidade da nova rede é igual a 0,385,



(a) (b)

Figura 4: Grafo de um único repositório (a) com links entre todos os desenvolvedores e (b) com links entre desenvolvedores com contribuição numa mesma janela de tempo.

inferior à anterior com densidade igual a 1. Portanto, essa nova modelagem expressa um cenário mais realista do relacionamento entre desenvolvedores em um grande repositório. Neste caso, o grau médio dos nós presentes na rede fica em torno de 20,8.

5. Conclusão e Trabalhos Futuros

Analisamos como diferentes propriedades topológicas e semânticas se relacionam com as que consideram o aspecto temporal. Os resultados mostram que GPC e PC podem ser utilizadas em conjunto em um modelo computacional para medir a força dos relacionamentos. A principal contribuição é melhorar a modelagem do grafo para representar a rede social de colaboração. A modelagem sem o aspecto temporal conecta todos os desenvolvedores que contribuíram em um mesmo repositório. Tal abordagem cria várias cliques para cada repositórios. No entanto, as propriedades com aspecto temporal revelaram que cerca de 80% dos pares de desenvolvedores não contribuíam em uma mesma janela temporal. Assim, em trabalhos futuros planejamos modelar as redes atribuindo arestas apenas entre desenvolvedores que colaboraram num mesmo período de tempo.

Agradecimentos. Trabalho parcialmente financiado por CAPES, CNPq e FAPEMIG.

Referências

Alves et al., G. B. (2016). The strength of social coding collaboration on github. In *SBBD Short Papers*, pages 247–252, Salvador, Brasil.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

Batista, N. A. et al. (2017). Collaboration strength metrics and analyses on github (aceito para publicação). In *IEEE/WIC/ACM WI*, Leipzig, Germany.

Casalnuovo, C. et al. (2015). Developer onboarding in github: The role of prior social links and language experience. In *ESEC/FSE*, pages 817–828, Bergamo, Italy.

Dabbish, L. et al. (2012). Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository. In *CSCW*, pages 1277–1286, Seattle, USA.

Easley, D. and Kleinberg, J. (2010). *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press.

Gousios, G. (2013). The GHTorrent Dataset and Tool Suite. In *MSR*, pages 233–236, S.Francisco, USA.

Tsay, J. et al. (2014). Influence of Social and Technical Factors for Evaluating Contribution in GitHub. In *ICSE*, pages 356–366, Hyderabad, India.