

UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Gabriel Lage Calegari

**Caracterização de redes de desenvolvimento colaborativo de software
inspirada em modelos biológicos**

Belo Horizonte, MG
2019

Gabriel Lage Calegari

**Caracterização de redes de desenvolvimento colaborativo de software
inspirada em modelos biológicos**

Versão final

Dissertação apresentada ao Programa de Pós-Graduação em
Ciência da Computação da Universidade Federal de Minas
Gerais, como requisito parcial à obtenção do título de Mestre
em Ciência da Computação.

Orientadora: Ana Paula Couto da Silva

Belo Horizonte, MG
2019

© 2019, Gabriel Lage Calegari.
. Todos os direitos reservados

Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa
CRB 6ª Região nº 1510

Calegari, Gabriel Lage.

C148c Caracterização de redes de desenvolvimento colaborativo de software inspirada em modelos biológicos / Gabriel Lage Calegari — Belo Horizonte, 2019.
xii, 82 f. il.; 29 cm.

Dissertação (mestrado) - Universidade Federal de Minas Gerais – Departamento de Ciência da Computação
Orientadora: Ana Paula Couto da Silva.

1. Computação – Teses. 2. Sistemas colaborativos – Teses. 3. Modelos biológicos – Simulação (computadores) – Teses. 4. GitHub – Teses. I. Orientadora. II. Título.

CDU 519.6*75 (043)



UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

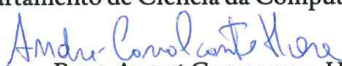
Caracterização de redes de desenvolvimento colaborativo de software
inspirada em modelos biológicos

GABRIEL LAGE CALEGARI

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:


PROFA. ANA PAULA COUTO DA SILVA - Orientadora
Departamento de Ciência da Computação - UFMG


PROF. FABRICIO MURAI FERREIRA
Departamento de Ciência da Computação - UFMG


PROF. ANDRÉ CAVALCANTE HORA
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 18 de Dezembro de 2019.

Agradecimentos

Esta não foi uma jornada fácil. Não poderia terminá-la sem agradecer aqueles que tiveram participação fundamental.

A orientadora Ana Paula, pela imensa paciência durante esses anos. Suas sugestões e ideias foram valiosas para o propósito que este trabalho buscou alcançar.

Sou grato a minha família por terem sempre incentivado a educação em nosso lar. Leitura, criatividade, e a busca de propósito eram atividades muito valorizadas por meus pais.

Ao meu amado Rafael, pela compreensão, cumplicidade e carinho que muito me ajudaram a persistir. Nos momentos de maior dificuldade, saber com quem contar é um privilégio que eu pude ter.

A minha amiga Roberta, com quem tive prazer de estudar por muitos anos, inclusive dividindo esta etapa, e de quem a vida me fez irmão; sua amizade tornou mais suave esta jornada.

A todos os professores, pelo ensino e pesquisa de qualidade.

A todos meus amigos, especialmente Mágda, Thaís, Kleisson e Fabiana, que trouxeram alegria para meus dias de desânimo.

Sou grato a todos que colaboraram, ainda que indiretamente, para o sucesso deste trabalho.

*“(...) A formiga é só trabalho.
A cigarra é só cantiga.*

*Mas sem a cantiga
Da cigarra
Que distrai da fadiga,
Seria uma barra
O trabalho da formiga!”
(José Paulo Paes)*

Resumo

Desde o surgimento da teoria da seleção natural, sistemas de colaboração ganharam evidência por terem criado um dilema de difícil solução: comportamentos colaborativos poderiam reduzir a aptidão relativa do colaborador, e agir contra ele na seleção natural. No entanto, a colaboração está por toda parte, em praticamente todos os níveis biológicos - de genes cooperando para genomas até seres já constituídos colaborando entre si. Sistemas colaborativos humanos foram ampliados com o advento da globalização e da Internet. Um desses sistemas são as redes de desenvolvimento colaborativo de software, surgindo na Web vários portais dedicados ao tema. O mais popular deles é o GitHub. Lançado em 2008, o GitHub possui mais de 36 milhões de usuários, e está organizado em torno de repositórios, onde vários usuários se reúnem para desenvolver software de forma colaborativa. O objetivo principal desta dissertação é analisar sob uma perspectiva eminentemente inspirada na biologia as redes de desenvolvimento colaborativo de software construídas no GitHub. Esta dissertação construiu um modelo para reproduzir a colaboração no GitHub como um ecossistema, caracterizando-o sob diferentes dimensões. Esses ecossistemas foram modelados como redes complexas e também caracterizados topologicamente ao longo do tempo. Por fim, alguns dos principais modelos biológicos para explicar a colaboração foram adaptados para essas redes, de modo a entender se eles também podem ser utilizados para explicar o desenvolvimento de software colaborativo.

Palavras-chave: Redes de Colaboração. Modelos Biológicos. GitHub.

Abstract

Since the emergence of the theory of natural selection, collaboration systems have come to light because they have created a difficult dilemma: collaborative behaviors could reduce a collaborator's relative fitness and act against him in natural selection. Besides this, collaboration is everywhere, at all biological level - from genes cooperating to genomes to constituted beings collaborating with each other. Human collaborative systems have been expanded with the advent of globalization and the Internet. One such system is the collaborative software development networks, with web portals devoted to the theme. The most popular of these is GitHub. Launched in 2008, GitHub has over 36 million users, and it is organized around repositories where multiple users come together to collaboratively develop software. The main objective of this dissertation is to analyze from an eminently biologically inspired perspective the collaborative software development networks built on GitHub. This dissertation built a model to reproduce the collaboration in GitHub as an ecosystem, characterizing it under different dimensions. These ecosystems have been modeled as complex networks and also characterized topologically over time. Finally, some of the key biological models for explaining collaboration have been adapted for these networks to understand whether they can also be used to explain collaborative software development.

Keywords: Collaboration Networks. Biological Models. GitHub.

Lista de Figuras

1.1	Influência crescente do GitHub	14
4.1	Página inicial de um repositório no GitHub exibindo suas funcionalidades . . .	33
4.2	Esquema relacional de GHTorrent	34
4.3	Evolução dos commits ao longo do tempo	38
4.4	Repositórios ativos ao longo do tempo	39
4.5	Visão geral da colaboração nos repositórios	40
4.6	Evolução do número de desenvolvedores nos ecossistemas	42
4.7	Entrada de novos desenvolvedores nos ecossistemas em comparação com os desenvolvedores já existentes	43
4.8	Proporção cumulativa de repositórios com um determinado número de <i>forks</i> . Este gráfico somente inclui repositórios com pelo menos um <i>fork</i>	45
5.1	Exemplo de rede de colaboração no tempo t e tempo $t+1$	50
5.2	Função de Distribuição Cumulativa (CDF) dos graus nas redes de colaboração	53
6.1	Visão parcial da árvore filogenética das linguagens de programação	64
6.2	Matriz da persistência de colaboradores ao longo dos anos	68
6.3	Função de Distribuição Cumulativa (CDF) das reputações dos usuários nas redes de colaboração	71
6.4	Matriz da estabilidade de colaboradores no ranking de reputação ao longo dos anos	73
6.5	Distribuição da conformidade dos colaboradores com relação à linguagem nas redes de colaboração	78
6.5	Distribuição da conformidade dos colaboradores com relação à linguagem nas redes de colaboração (cont.)	79
6.5	Distribuição da conformidade dos colaboradores com relação à linguagem nas redes de colaboração (cont.)	80

Lista de Tabelas

4.1	Sumário dos ecossistemas de colaboração analisados	37
4.2	Ranking das 10 linguagens de programação com o maior número de repositórios	44
4.3	Ranking das 10 linguagens de programação com o maior número de commits .	44
5.1	Sumário das redes de colaboração	52
5.2	Grau dos vértices nas redes de colaboração	54
5.3	Densidade e Assortatividade das redes de colaboração	54
5.4	Distribuição dos componentes nas redes de colaboração	55
5.5	Diâmetro do LCC das redes de colaboração	56
5.6	Distribuição do <i>betweenness</i> nas redes de colaboração	56
6.1	Sugestão de interpretação dos valores obtidos com a métrica \mathcal{C}_L^D	66
6.2	Valores máximos, mínimos, 50° (mediana) e 90° percentil da distribuição da reputação	69
6.3	<i>Skewness</i> e <i>Curtose</i> da distribuição de reputação	70
6.4	Colaboradores do grupo R1 e porcentagem de repositórios em que colaboraram	72
6.5	Quantidade de repositórios e novos colaboradores em cada ano no grupo R1 e R2	74
6.6	Número de novos colaboradores por repositório (\mathcal{A}_{R_1} e \mathcal{A}_{R_2}) e força de atração do grupo R1 (Φ)	75
6.7	Distribuição do número de colaboradores por repositório nos ecossistemas ana- lisados	75
6.8	Mediana da distribuição de commits por grupo de repositórios	76

Sumário

Agradecimentos	4
Resumo	6
Abstract	7
Lista de Figuras	8
Lista de Tabelas	9
1 Introdução	12
1.1 Objetivos	15
1.2 Contribuições	15
1.3 Estrutura da dissertação	16
2 Conceitos Fundamentais	17
2.1 Sistemas de colaboração biológicos	17
2.2 Modelos biológicos para colaboração	18
2.2.1 Modelos de Reciprocidade Direcionada	18
2.2.2 Modelos de Benefícios por Produtos	20
2.2.3 Modelos de Genes Compartilhados	21
2.3 Colaboração em redes de desenvolvimento de software e modelos biológicos	22
3 Trabalhos relacionados	24
3.1 Colaboração entre humanos à luz de modelos biológicos	24
3.2 Redes de colaboração entre humanos	25
3.3 Redes de desenvolvimento colaborativo de software	27
3.4 Diferencial desta dissertação	29
4 Ecossistemas de Colaboração	31
4.1 Definição de ecossistema de colaboração	31
4.2 Ecossistemas no GitHub	32
4.3 Caracterização dos ecossistemas de colaboração	37
4.3.1 Commits	38
4.3.2 Desenvolvedores	41

4.3.3	Linguagens e <i>forks</i>	42
4.4	Discussão	45
5	Redes de desenvolvimento colaborativo de software	48
5.1	Modelo e Métricas	48
5.2	Caracterização das redes de desenvolvimento colaborativo de software . . .	52
5.3	Discussão	57
6	Caracterização das redes de colaboração inspirada em modelos biológicos	58
6.1	Metodologia	58
6.1.1	Modelos de Reciprocidade Direcionada	59
6.1.2	Modelos de Benefícios por Produtos	62
6.1.3	Modelos de Genes Compartilhados	63
6.2	Análise Experimental	66
6.2.1	Modelos de Reciprocidade Direcionada	67
6.2.2	Modelos de Benefícios por Produtos	74
6.2.3	Modelos de Genes Compartilhados	76
6.3	Discussão	77
7	Conclusão	82
	Referências Bibliográficas	85

Capítulo 1

Introdução

A colaboração é uma atividade presente em diferentes organismos, organizações e níveis biológicos. Colaborar significa se juntar a outro agente para a realização de uma atividade com objetivos comuns. Sistemas de colaboração podem ser observados nos seres vivos em muitos níveis: de genes cooperando para genomas, passando por células colaborando para criar organismos, até organismos já constituídos colaborando entre si. Nas sociedades humanas, a colaboração faz parte do circuito social, econômico e científico [Lopes et al., 2009; Fehr & Schurtenberger, 2018].

O advento da globalização e da Internet tem afetado e ampliado os sistemas colaborativos humanos. A era da informação, como é chamado o período após 1950, tornou o mundo extremamente dependente do conhecimento tecnológico e aproximou países em torno da colaboração através da informação. De acordo com Adams [2012], o avanço dos sistemas colaborativos tem posto em xeque o equilíbrio global da ciência. Isso significa que superpotências podem deixar de sê-las caso não entendam a dinâmica dos sistemas colaborativos humanos. Nessa perspectiva, a colaboração é positiva porque permite que o conhecimento seja melhor transferido e combinado, e acelera o crescimento de economias em desenvolvimento. A colaboração, no entanto, tem custos de tempo e de agenda compartilhada, que podem ser altos. Existe ainda o risco de que a colaboração possa gerar alta convergência e que outros aspectos sejam desconsiderados. Em todas as perspectivas, compreender a dinâmica dos sistemas de colaboração parece ser a chave para elucidar essas questões.

Há vários trabalhos buscando entender a colaboração a partir da teoria da evolução biológica [Hamilton, 1963; Wynne-Edwards, 1964; Alexander, 1987; Nowak & Sigmund, 1998; Bull & Rice, 1991], com alguns em especial focando na colaboração humana [Rand & Nowak, 2013; Barclay & Raihani, 2016; Boyd & Richerson, 2009]. Outros trabalhos [Gracia-Lázaro et al., 2012; Rand et al., 2011; Ramasco et al., 2004] têm se dedicado ao entendimento dos sistemas de colaboração humanos sob a perspectiva de redes, uma vez que esses sistemas apresentam características de sistemas complexos¹.

¹Os sistemas complexos são descritos como sistemas constituídos de partes em que os comportamentos isolados somados não conferem a complexidade total do sistema [Mitchell & Newman, 2002]. Por serem constituídos de partes que se interagem, muitos sistemas complexos têm sido estudados a partir da modelagem matemática em redes. Uma rede, também chamada de grafo, é definida como sendo um

Nos últimos anos, com o surgimento do movimento do software livre, um sistema de colaboração humano específico ganhou notoriedade: as redes de desenvolvimento colaborativo de software. O software livre é essencialmente colaborativo, uma vez que qualquer indivíduo pode ler, modificar e melhorar o código existente. Essas redes tornaram-se uma fonte quase inesgotável de troca de conhecimento em torno do desenvolvimento de software. Em uma época dominada pela tecnologia, onde a palavra algoritmo ganhou um peso a ponto de ameaçar democracias², a colaboração em torno do software tem um impacto mundial.³ O desenvolvimento colaborativo de software permite que a tecnologia seja estendida a muitos, de modo que países em desenvolvimento também possam se aproveitar da transferência tecnológica.⁴

De acordo com De Luca Pretto & da Silveira [2008], o movimento em direção à colaboração é uma revolução irrefreável. Existem uma série de portais na web dedicados ao desenvolvimento de software colaborativo, como o GitHub, Assembla, BitBucket e Sourceforge, sendo o mais popular o GitHub.

O GitHub é um sistema web para repositórios de software, baseado no sistema de controle de versões **git**. Lançado em 2008, o GitHub possui mais de 36 milhões de usuários (novembro de 2019)⁵ e é um dos portais do gênero mais relevantes, ocupando a 83^a posição (novembro de 2019) entre os sites mais populares do mundo.⁶ Como parte de seu sistema de desenvolvimento de software colaborativo, o GitHub também oferece funcionalidades de gestão de projetos, *wiki* e rede social.

Recentemente, o GitHub tem sido também utilizado em projetos de colaboração de diferentes naturezas. Segundo Perkel [2016], o GitHub tem funcionado com sucesso como um repositório de dados científicos, ajudando a cientistas de diferentes áreas a, por exemplo, controlar o surto de *ebola*. A rede de colaboração científica no Github cresceu e inúmeros artigos científicos citam projetos do portal. A Figura 1.1 mostra que em comparação com 2010, o número de citações ao GitHub dobrou em áreas como ciência da computação, mas também tem crescido em áreas como biologia, neurociência e matemática.

Devido à sua influência crescente, o GitHub tem sido foco de muitos trabalhos [Dabish et al., 2012; Lima et al., 2014; Batista et al., 2017; El Asri et al., 2017], sendo que a maior parte deles está dedicada a entender a estrutura e dinâmica da colaboração, a

conjunto de itens, conhecidos como vértices ou nós, e possuem conexões entre si, as arestas [Newman, 2003]

²Os impactos dos algoritmos do Facebook e Google na democracia. <https://apublica.org/2018/05/o-impacto-dos-algoritmos-do-facebook-e-google-na-democracia/>

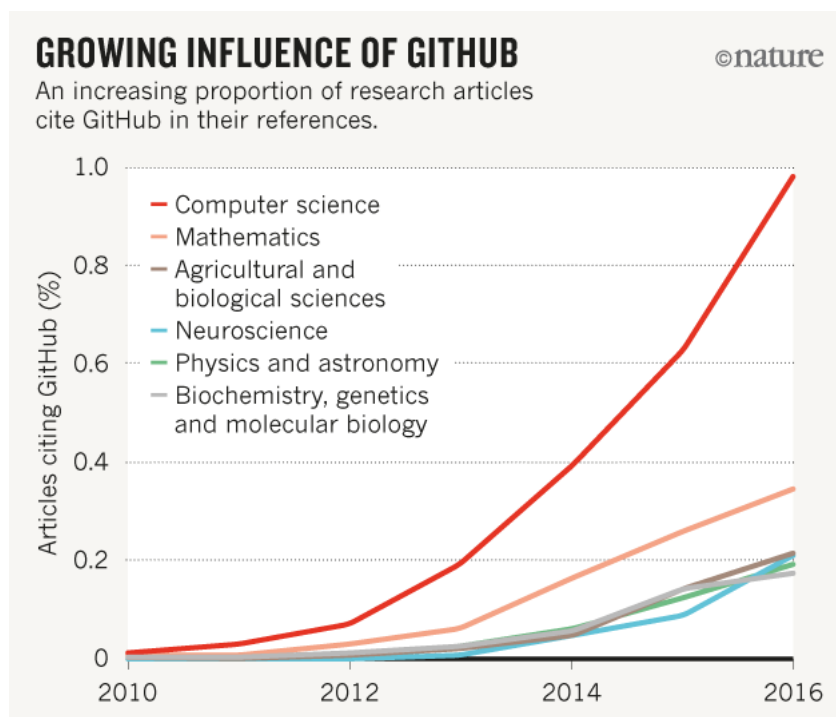
³O mundo mediado por algoritmos. <https://revistapesquisa.fapesp.br/2018/04/19/o-mundo-mediado-por-algoritmos/>

⁴Manifesto GNU. <https://www.gnu.org/gnu/manifesto.pt-br.html>

⁵Uma busca por usuários em 12/11/2019 retornou 36.431.900 usuários ativos. A busca pode ser realizada em <https://github.com/search?q=type:user&type=Users>

⁶O *ranking* pode ser consultado em <http://www.alexa.com/siteinfo/github.com>

⁷Imagem extraída do trabalho de Perkel [2016]

Figura 1.1: Influência crescente do GitHub ⁷

formação de lideranças, e a influência de certos usuários na rede de colaboração.

O GitHub também é o foco desta dissertação. Ao analisar o desenvolvimento colaborativo de software, propomos um modelo para a construção de ecossistemas de colaboração, de modo a caracterizar o GitHub sob a ótica de modelos biológicos. Esses ecossistemas diferenciam-se das abordagens frequentemente adotadas na literatura, uma vez que são focados no colaborador e conferem diversidade, incorporando uma série de linguagens e repositórios. As análises com modelos biológicos requerem ainda uma estrutura em rede temporal, que com nosso modelo, permite reconstruir o processo colaborativo indivíduo a indivíduo. É importante ressaltar que a metodologia apresentada nesta dissertação não tem como objetivo substituir ou competir com as demais metodologias propostas na literatura, que visam entender o surgimento e a evolução do processo de colaboração no GitHub. O nosso objetivo é introduzir uma nova visão, que complementa a complexa tarefa de entender e explicar o fenômeno de colaboração nesta rede de desenvolvimento de software.

Por fim, os trabalhos que estudam a colaboração humana sob o ponto de vista de modelos biológicos são recentes. Esta dissertação é uma oportunidade de contribuir para um entendimento deste fenômeno, oferecendo uma perspectiva da colaboração humana ainda não explorada na literatura, como é o caso da colaboração em redes online de desenvolvimento de software.

1.1 Objetivos

Os objetivos principais desta dissertação são os seguintes: (1) caracterizar as redes de desenvolvimento colaborativo de software ao longo do tempo e (2) analisar se alguns modelos biológicos para a colaboração disponíveis na literatura são aplicáveis às redes de desenvolvimento colaborativo de software, e como elas se comportam ao longo do tempo seguindo esses modelos. Considerando esses objetivos, esta dissertação está organizada de modo a buscar respostas para as seguintes questões de pesquisa (QP):

- **QP1 - Quais são as principais características dos ecossistemas de colaboração no GitHub?** Como buscamos entender a rede de colaboração do GitHub sob uma ótica de modelos biológicos, introduzimos o conceito de ecossistema. Diferentemente dos trabalhos encontrados na literatura, a definição de ecossistema enriquece a compreensão do processo de colaboração tanto do ponto de vista de diversidade quanto da consideração da história de cada colaborador na rede.
- **QP2 - Existem semelhanças entre as redes modeladas a partir de diferentes ecossistemas?** A noção temporal dos ecossistemas foi bem expressa a partir da modelagem em rede. Dividimos o ecossistema em *snapshots*, em que os desenvolvedores são representados pelos vértices, e uma aresta existe sempre que dois desenvolvedores realizam commit em um mesmo repositório. Analisamos a topologia dessas redes a partir de métricas clássicas de redes complexas.
- **QP3 - Os mecanismos e princípios inerentes aos modelos biológicos podem ser utilizados como ferramentas que complementam o entendimento do processo de colaboração em redes de desenvolvimento de software?** A partir de informações extraídas dos ecossistemas e das redes modeladas, apresentamos métricas que capturam diferentes princípios dos modelos biológicos, como reciprocidade, sinergismo e preferência de colaboração com semelhantes.

1.2 Contribuições

As contribuições desta dissertação perpassam pela metodologia que auxilia nas análises de redes de desenvolvimento colaborativo de software sob a ótica de modelos biológicos. A seguir descrevemos as principais contribuições desta dissertação.

Definição, Modelagem e Caracterização de Ecossistemas de colaboração.

Definimos um modelo para o estudo da colaboração no GitHub, que busca mimetizar o conceito biológico de ecossistema, caracterizado por um fluxo fechado e auto-sustentável de relacionamento entre partes bióticas e abióticas. Essa abordagem permite centralizar o foco na colaboração e garante diversidade, ao contar com a participação de inúmeros repositórios e linguagens de programação. A modelagem matemática desses ecossistemas em rede permite reproduzir a história da colaboração do usuário, e não apenas de um repositório ou de repositórios de uma linguagem, abordagens que são mais frequentes na literatura.

Caracterização de redes de desenvolvimento colaborativo de software sob a inspiração de modelos biológicos. Utilizamos modelos biológicos para estudar as redes de desenvolvimento colaborativo de software buscando entender que tipos de mecanismos já explorados na biologia são usados nas colaborações no GitHub. Um melhor entendimento dos mecanismos que regem a colaboração no GitHub, podem trazer *insights* que melhorem e fomentem o processo de colaboração nestes tipos de sistemas. A caracterização aqui apresentada complementa as análises propostas na literatura, propiciando uma nova faceta para o entendimento deste complexo processo que é a colaboração entre humanos.

1.3 Estrutura da dissertação

Esta dissertação está organizada da seguinte maneira: o Capítulo 2 apresenta os fundamentos teóricos que embasam esta pesquisa, os modelos biológicos para colaboração. O Capítulo 3 apresenta e discute os trabalhos relacionados a esta pesquisa. O Capítulo 4 conceitua ecossistemas de colaboração, apresenta nossa metodologia para contruí-los utilizando o GitHub e os caracteriza. O Capítulo 5 apresenta o modelo matemático utilizado para a análise e compreensão das colaborações ao longo do tempo e faz uma caracterização dessas redes. No Capítulo 6, as redes de desenvolvimento colaborativo de software são analisadas sob a ótica de modelos biológicos. Por fim, o Capítulo 7 apresenta as conclusões e contribuições desta dissertação, além de trabalhos futuros.

Capítulo 2

Conceitos Fundamentais

Este capítulo descreve um conjunto de modelos biológicos utilizados para entender colaboração entre seres vivos, e que serão aplicados para análise das redes de desenvolvimento colaborativo de software. A Seção 2.1 apresenta uma visão geral da literatura com relação a sistemas de colaboração sob o ponto de vista biológico. A Seção 2.2 descreve os modelos específicos selecionados para as análises nesta dissertação.

2.1 Sistemas de colaboração biológicos

Os sistemas de colaboração introduzem inúmeras questões no entendimento da evolução das espécies. Enquanto a seleção natural – mecanismo aceito pela comunidade científica como chave para a evolução das espécies – tem como pressuposto que as espécies mais aptas são aquelas que desenvolvem características que as beneficia individualmente, o pressuposto de um sistema de colaboração é simplesmente que há um indivíduo colaborando com o outro, não necessariamente garantindo benefício ao colaborador, apesar de sempre beneficiar o destinatário da ação.

Charles Darwin deu os primeiros passos para investigar de que maneira a colaboração pode influenciar modificações específicas em espécies. Em sua obra “A Origem das Espécies”, ele introduziu este dilema evolucionário: “*Natural selection cannot possibly produce any modification in any one species exclusively for the good of another species.*”¹ Desse modo, a colaboração passou a ser tida como um contrassenso para a explicação sugerida por Darwin para a evolução das espécies.

Muitas lacunas deixadas por Darwin vem sendo explicadas ao longo da história pelos neodarwinistas, incorporando os conhecimentos da genética e da biologia molecular. O dilema darwinista poderia hoje ser reescrito de outra forma: os genes relacionados a uma característica de colaboração precisam beneficiar a eles próprios desproporcionalmente se

¹Tradução: “A seleção natural não pode possivelmente produzir qualquer modificação em uma espécie qualquer exclusivamente para o proveito de outra espécie.”

quiserem aumentar em frequência.

A evolução da cooperação foi um tema de interesse de muitos biólogos nas últimas décadas, iniciando-se a partir da publicação de Hamilton [1963], que explica a seleção natural como sendo consequência de atos egoístas, e sendo assim, a evolução da colaboração propensa a ocorrer sob condições restritivas. Desde então muitos modelos surgiram na tentativa de explicar comportamentos colaborativos: seleção por parentesco [Hamilton, 1963; Wynne-Edwards, 1964], reciprocidade direcionada [Trivers, 1971; Axelrod & Hamilton, 1981], sinergismo [Queller, 1985], reciprocidade indireta [Alexander, 1987; Nowak & Sigmund, 1998], escolha do parceiro [Eshel & Cavalli-Sforza, 1982; Noë, 1990; Bull & Rice, 1991], entre outros.

Esta dissertação utilizará alguns dos modelos citados como orientação para caracterizar e analisar redes de desenvolvimento colaborativo de software. As principais características desses modelos serão descritas na Seção 2.2.

2.2 Modelos biológicos para colaboração

Sachs et al. [2004] apresentam um framework hierárquico em que vários dos modelos citados anteriormente foram organizados e podem ser melhor comparados. Essa classificação divide tipos de colaboração em três classes de modelos: reciprocidade direcionada - colaboração em que há benefícios de volta ao colaborador; benefícios por produtos - colaboração como consequência de atos egoístas; genes compartilhados - colaboração por parentesco. Cada um desses modelos serão apresentados a seguir.

2.2.1 Modelos de Reciprocidade Direcionada

Os modelos de reciprocidade direcionada tem como premissa que um indivíduo aceita um custo potencialmente alto para beneficiar um parceiro específico, e o parceiro compensa o esforço retribuindo-lhe o benefício mais tarde. A classe reciprocidade direcionada se aplica tanto às colaborações inter-específicas (aquelas realizadas entre espécies diferentes) quanto intra-específicas (aquelas realizadas com seres da mesma espécie).

Uma característica desses modelos é a não-garantia de que a colaboração se manterá, isto é, pode haver traição e o indivíduo beneficiado não ser recíproco. A ideia de traição está bem sintetizada no **Dilema do Prisioneiro Iterativo**, que também é a apre-

sentação mais comum dos modelos de reciprocidade direcionada. O dilema do prisioneiro na sua versão simples foi formalizado por Tucker [1983] da seguinte maneira:

Dois membros de uma gangue foram presos. Cada prisioneiro está confinado em uma solitária sem nenhum meio de se comunicar com o outro. Os promotores não tem provas suficientes para condená-los na acusação principal. Os prisioneiros esperam ser sentenciados com um ano de prisão. Simultaneamente, os prisioneiros oferecem a cada um uma barganha. A cada prisioneiro é dada a oportunidade de: trair o outro testemunhando que o outro cometeu o crime ou colaborar com o outro ficando em silêncio. A oferta é:

- Se A e B traem um ao outro, cada um deles ficará 2 anos na prisão
- Se A trai B, mas B fica em silêncio, A ficará em liberdade e B ficará 3 anos na prisão (ou vice-versa)
- Se A e B permanecem em silêncio, ambos ficarão 1 ano na prisão.

Nessa versão simples, o resultado do dilema tem uma alta probabilidade de ser de traição mútua, uma vez que é muito difícil que se estabeleça confiança. A versão iterativa do dilema, consiste na repetição sistemática das escolhas entre os mesmos prisioneiros, que permitem o desenvolvimento de estratégias, baseado no resultado da iteração anterior. Nessa versão, o comportamento cooperativo pode surgir depois de uma série de acordos, vinganças e perdões. Uma das estratégias que emergem da versão iterativa desse dilema é conhecida como *tit-for-tat*: “colaborar quando seu parceiro cooperou na iteração anterior mas recusar a cooperar se seu parceiro não cooperou na iteração anterior”.

O dilema do prisioneiro iterativo tem, portanto, duas características principais: iterações repetidas entre os parceiros e resposta diferencial aos parceiros (estratégia). Essas características dão origem a dois modelos de colaboração: *partner choice*, caracterizado pela existência de um mecanismo de decisão para a colaboração e *partner fidelity feedback*, com foco na continuidade da colaboração.

No modelo *Partner Choice*, um indivíduo A recebe benefícios de um parceiro colaborador B e retribui-lhe, mas evita retribuir benefícios recebidos de parceiros pouco colaborativos. Ao preferir manter a colaboração com os parceiros que mais colaboram, o indivíduo A melhora a sua própria aptidão e promove a evolução da cooperação na espécie de B, uma vez que ele beneficia os indivíduos mais colaborativos. Esse mecanismo é, porém, prejudicado quando há uma quantidade limitada de parceiros disponíveis, uma vez que nesse caso, o custo de rejeitar um parceiro potencial é mais alto (há o risco de ficar sem parceiro de colaboração). Para escolher um parceiro, o indivíduo precisa fazer avaliações sobre o quão cooperativo o parceiro é, e decidir de que forma quer manter contato com ele.

Há três formas mais comuns para avaliar a cooperatividade do parceiro: *parceling*, *distributing* e *image scoring*. No sistema *parceling*, o indivíduo inicia a colaboração com um único recurso (uma tarefa, por exemplo) e a colaboração é aumentada ao longo do tempo. Isso ocorre como uma forma de prevenção, pois o indivíduo evita perder tempo com um parceiro egoísta ou pouco colaborador. O sistema *parceling* equivale às iterações do Dilema do Prisioneiro Iterativo. O sistema *distributing* divide a colaboração em lotes espaciais (a colaboração é executada simultaneamente em vários espaços) diferentemente do *parceling*, que são lotes temporais (colaboração é incrementada com o tempo). Assim, a regra de decisão é baseada no lote espacial. Isso significa que as decisões são separadas para cada lote e podem ser feitas para vários parceiros de forma diferente. Por sua vez, o sistema *image scoring* consiste em avaliar a reputação de um parceiro quando colabora com um terceiro. Esse sistema também é chamado de “reciprocidade indireta”, pois quando A colabora com B, a reputação de A aumenta, o que o torna mais suscetível a receber benefícios de outros.

Quando a colaboração não é baseada em um processo de decisão, mas a reciprocidade ainda está envolvida, o modelo *Partner Fidelity Feedback* pode ajudar a explicar o mecanismo desse processo. Esse modelo tem como premissa que após contínuas ou até mesmo discretas iterações entre dois indivíduos que estão colaborando, surge um mecanismo de retroalimentação. Se um indivíduo A deixa de cooperar com um indivíduo B, sua própria aptidão será afetada, porque também deixará de receber benefícios de B, uma vez que a relação de fidelidade que existia entre eles foi quebrada. A estabilidade desse modelo é fortemente dependente da força de aptidão entre os parceiros. Alguns fatores podem facilitar esse acoplamento, tais como a limitação da habilidade de dispersão entre os parceiros (alta viscosidade populacional, por exemplo); e pequenas traições por parte de um parceiro em um intervalo curto de tempo, levando a uma perda muito grande de aptidão, e conseqüentemente desincentivando a traição.

2.2.2 Modelos de Benefícios por Produtos

A classe de modelos de benefícios por produtos reúne os modelos mais recentes da literatura [Connor, 2008; Cockburn, 1998; Richardson et al., 2002]. Nesses modelos, um indivíduo A recebe benefício de outro indivíduo B como um produto de algum ato egoísta de B. É possível modelar colaborações inter-específicas e intra-específicas com esses modelos.

Nesta classe, a colaboração é dita “por produtos de uma via”, quando um indivíduo A recebe um benefício como consequência de um ato egoísta de B, que não tinha a intenção

de beneficiá-lo.

A colaboração é dita “por produtos de duas vias”, quando os benefícios incidentais ocorrem em ambas as direções. Este modelo, também chamado de mutualismo por produto, é caracterizado pelo fato de um ato egoísta executado pelo indivíduo A beneficiar B, mas também um ato egoísta de B beneficiar A. Uma boa representação desse modelo é o que denominou-se *sinergismo*, quando um grupo adota um comportamento porque recebem mais benefícios do que se feitos individualmente. Isto é, o comportamento do grupo evolui pela seleção individual sempre que há uma relação de ganho à medida que o tamanho do grupo aumenta. Em algumas espécies, o pertencimento a um grupo reduz sua chance de predação, assim o ato egoísta de um (“querer estar em um grupo para se salvar”) acaba sendo um comportamento generalizado que os beneficia.

Por fim, a colaboração é dita com “reciprocidade por produtos” quando a seleção natural incorpora a reciprocidade, porque tal ato maximizaria os benefícios recebidos. Um exemplo é o caso de um pássaro africano que guia humanos para a localização de colmeias. Na destruição de uma colmeia, é comum que haja restos de mel, alimento desse pássaro. Presume-se que o pássaro evoluiu para esse comportamento, pois ele maximiza seus ganhos [Richardson et al., 2002].

2.2.3 Modelos de Genes Compartilhados

Os modelos de genes compartilhados baseiam-se na premissa de que um indivíduo colabora com outro indivíduo quando ambos compartilham alelos de um ascendente comum. Devido a isso, esse modelo é adequado para colaborações intra-específicas. O modelo de Hamilton [1963], um dos primeiros modelos biológicos para colaboração, parte dessa premissa, em que ocorre seleção por parentesco. A seleção por parentesco (*kin selection*) pode ocorrer com dois mecanismos: algumas espécies são capazes de reconhecer parentesco a partir de certas características, o que foi nomeado como *kin recognition*; outras espécies que não possuem essa habilidade costumam ser aquelas que vivem próximas do local em que nasceram, portanto há uma maior chance de encontrarem parentes. Sachs et al. [2004] subdivide essa classe em dois modelos: *kin fidelity* e *kin choice*.

No mecanismo de fidelidade por parentesco (*kin fidelity*), a colaboração ocorre devido à proximidade espacial entre os agentes. Isso ocorre, por exemplo, quando filhotes resolvem compartilhar um ninho. Aves que costumam alimentar seus filhotes no ninho, alimentarão inclusive aqueles que não são seus descendentes, mas que chegaram ali de alguma forma. Apesar da colaboração entre próximos, nesse mecanismo os indivíduos próximos podem também competir por recursos, o que pode afetar moderadamente a

seleção por cooperação.

No mecanismo de escolha por parentesco (*kin choice*), indivíduos escolhem colaborar com aqueles indivíduos com os quais reconhecem parentesco (*kin recognition*). O reconhecimento pode se dar a partir de características fenotípicas: visual, auditiva, olfativa, entre outros. Desse modo, mesmo que dois indivíduos não sejam de fato parentes, basta que haja um reconhecimento fenotípico que indique a presença de um gene comum para a colaboração para que ela ocorra.

2.3 Colaboração em redes de desenvolvimento de software e modelos biológicos

Nesta dissertação, pretendemos estudar um sistema humano específico de colaboração, formado por indivíduos que desenvolvem software em conjunto através da Internet. Portanto, é de se considerar que toda a literatura biológica acerca da colaboração possa ser utilizada para fundamentar nossas análises. O sistema de colaboração que vamos analisar neste trabalho, o GitHub, possui muitas características que nos levam a hipotetizar acerca dos mecanismos biológicos envolvidos nos modelos aqui apresentados.

Considerando os modelos de reciprocidade direcionada, podemos analisar se no GitHub há um processo de decisão que influencia na intensidade de colaboração em um repositório, e se existe alguma estratégia envolvida nesta decisão. Ao analisarmos o surgimento e persistência da colaboração nessas redes ao longo do tempo, podemos aplicar conceitos como *image scoring* como ferramenta para entendermos o fenômeno da colaboração do ponto de vista de modelos biológicos.

Os modelos de benefícios por produtos, por sua vez, também parecem se adequar bem às redes de desenvolvimento colaborativo de software, uma vez que usuários podem decidir colaborar apenas por um interesse egoísta: adicionar uma funcionalidade em um software apenas porque precisam dela, e querem economizar tempo e empenho, evitando começar um novo repositório. Portanto, queremos entender se há a presença de sinergismo nessas redes, analisando a produção de benefícios em repositórios com muitos e poucos desenvolvedores.

Finalmente, considerando os modelos de genes compartilhados, como o *kin choice*, em que características fenotípicas afetam a colaboração, podemos entender de forma mais clara como os desenvolvedores tendem a focar a colaboração com indivíduos mais similares, do ponto de vista do tipo de comunicação utilizada, ou seja, da linguagem de programação utilizada em um repositório.

Essa nova abordagem pode trazer *insights* que melhorem e fomentem o processo de colaboração nesse tipo de sistema. Por exemplo, se verificarmos que *image scoring* é um dos mecanismos utilizados na colaboração, pode-se pensar em estratégias que incentivem ainda mais a colaboração, visando os colaboradores de alta reputação. Isso promoveria a persistência da colaboração, disseminaria conhecimento, e poderia inclusive promover softwares de melhor qualidade. No caso de verificar a ocorrência de *kin choice*, podemos pensar no quanto isso afetaria a colaboração, devido ao problema de convergência de conhecimento em torno de uma única linguagem, e propor mecanismos para atenuá-lo. Por fim, entender a ocorrência de benefícios por produtos pode auxiliar na predição da evolução da colaboração, uma vez que, segundo esse modelo, a colaboração traz mais benefícios desproporcionalmente em grupos grandes.

Capítulo 3

Trabalhos relacionados

Esta dissertação trata da caracterização das redes de desenvolvimento colaborativo de software estabelecidas no GitHub, à luz de modelos biológicos. Portanto, seu contexto abrange estudos que analisam a colaboração em seres vivos, especialmente em humanos, e também estudos sobre a organização social e colaborativa no GitHub. A Seção 3.1 apresenta trabalhos sobre a colaboração entre humanos à luz de modelos biológicos; a Seção 3.2 discorre sobre estudos com redes de colaboração entre humanos; a Seção 3.3 apresenta os trabalhos sobre as redes de desenvolvimento colaborativo de software. Finalmente, a Seção 3.4 expõe o diferencial deste trabalho com relação aos apresentados.

3.1 Colaboração entre humanos à luz de modelos biológicos

A literatura científica muito se debruçou no estudo da colaboração entre diferentes espécies. No entanto, ainda há poucos trabalhos estudando a colaboração estabelecida em humanos. Em um desses trabalhos, Melis & Semmann [2010] compararam o que difere a colaboração entre humanos daquela que ocorre entre outras espécies. Nesse artigo, os autores ressaltam que a colaboração entre humanos está mais marcada por mecanismos de reforço e fiscalização, que concedem recompensa, punição e construção de reputação, do que entre animais. Além disso, seres humanos possuem mecanismos cognitivos que lhe permitem um uso mais extensivo da experiência pregressa, o que remete a colaboração entre humanos a modelos de reciprocidade direcionada. Sem os mecanismos cognitivos, em outras espécies predomina a colaboração com benefícios por produtos e seleção por parentesco.

Um trabalho similar ao de Melis & Semmann [2010] foi elaborado por Rand & Nowak [2013]. Nesse artigo eles revisaram os modelos biológicos teóricos para colaboração com experimentos em humanos. Os resultados apresentam evidências de cinco

mecanismos na colaboração humana: reciprocidade direcionada, reciprocidade indireta, seleção espacial, seleção multinível e seleção por parentesco.

O modelo *partner choice* foi investigado por meio da análise de comportamentos generosos em humanos por Barclay & Willer [2006]. Esse trabalho tem como base a teoria da sinalização dispendiosa, que assevera que um ato altruísta traz como benefícios ao executor melhor acesso a relacionamentos colaborativos e conseqüentemente maior cooperação dentro desses relacionamentos, uma vez que tornam público o custo de executá-lo. Os autores realizaram um teste para entender o comportamento altruísta através de um jogo colaborativo em duas versões: em uma havia o mecanismo de escolha do parceiro, e em outra esse mecanismo não era permitido. Os resultados mostraram que as pessoas competem para serem mais generosas que outras, quando o mecanismo de escolha do parceiro é permitido. Além disso, quando os benefícios da colaboração altruísta eram extremamente altos, tornou-se evidente que os supostos atos altruístas eram baseados em egoísmo.

Em um outro trabalho, Barclay & Raihani [2016] estudaram o mecanismo de *partner choice* em comparação com mecanismos de punição em uma versão modificada do Dilema do Prisioneiro. Nesse jogo, quando havia a possibilidade de escolha de parceiro, a colaboração mostrou-se mais alta. Quando não havia essa possibilidade, as pessoas preferiam punir os maus colaboradores em vez de deixarem de colaborar com eles. Numa versão em que os dois mecanismos eram possíveis, ambos os mecanismos eram evocados, e nesse caso a punição mostrou-se mais severa. As conclusões apontam que a punição não é um mecanismo indicado para promover a colaboração.

Wedekind & Milinski [2000] estudaram especificamente um sistema de avaliação previsto no mecanismo *partner choice*: o *image scoring*, que consiste em avaliar a maneira como um indivíduo é visto por um grupo. Nesse artigo, eles testaram a generosidade através de um jogo de doação de dinheiro. Havia uma única regra nesse jogo: os jogadores não podiam doar dinheiro para a mesma pessoa de quem recebeu uma doação na iteração anterior. O jogo era anônimo, de modo que um jogador não conhecia um ao outro senão por apelidos. Os resultados mostraram que os jogadores que haviam sido mais generosos em iterações anteriores também foram os mais beneficiados, de modo geral, o que evidencia que a cooperação foi controlada pelo mecanismo de *image scoring*.

3.2 Redes de colaboração entre humanos

Muitos trabalhos utilizaram a abordagem de redes para entender algum aspecto da colaboração entre humanos. Nesta seção apresentaremos alguns desses trabalhos. Alguns

desses trabalhos também utilizam conceitos da colaboração biológica, como por exemplo o trabalho de Gracia-Lázaro et al. [2012], que tinham como objetivo entender se as estruturas topológicas afetam a colaboração. Para isso, realizaram amplos estudos com humanos utilizando o Dilema do Prisioneiro. Nesse trabalho, eles promoveram estruturas de rede *lattice* e *scale-free*. Os resultados indicam que a estrutura não afetou o nível de colaboração, e que a colaboração se deu a partir de mecanismos de reciprocidade direcionada.

Em Rand et al. [2014], as estruturas em rede também foram levadas em consideração para entender sua influência sobre a colaboração. Nesse trabalho, os autores estudaram se as redes fixas são mais propensas à colaboração que as dinâmicas, uma vez que os colaboradores teriam mais chance de interagir. Com seus experimentos, foi possível demonstrar que redes estáticas podem levar a altos níveis de estabilidade de colaboração.

Anteriormente, um outro trabalho [Rand et al., 2011] havia estudado a influência das redes dinâmicas sobre a colaboração entre humanos. Nesse estudo, os autores mostraram que uma estrutura aleatória, totalmente fixa ou pouco dinâmica, afeta a colaboração. A manutenção da colaboração é melhor observada quando existe a possibilidade frequente de religação da rede. O principal modelo utilizado para explicar as religações é a reciprocidade direcionada, a partir do mecanismo de escolha do parceiro.

Grande parte dos trabalhos em redes de colaboração com humanos estão focados em redes científicas e de co-autoria e não utilizam conceitos biológicos. Newman [2001], por exemplo, estudou redes estáticas de colaboração científica. Nessas redes, dois cientistas estão conectados se eles colaboraram em um ou mais artigos. O trabalho consistiu de caracterização dessas redes: número de autores (vértices), número de artigos por autores, número de autores por artigo, número de colaboradores por autor (grau), tamanho do componente gigante e coeficiente de clusterização. Foram estudadas redes de diferentes áreas do conhecimento. Algumas conclusões são que essas redes são altamente clusterizadas; seguem distribuições de potência (*scale-free*), existem diferenças nas distribuições das diferentes áreas; e a presença do fenômeno *small-world* foi detectada.

A dinâmica de redes de colaboração entre humanos foi estudada em [Abbasi et al., 2011]. Esse trabalho consistiu na avaliação de métricas de redes de colaboração científica construídas para um intervalo de 40 anos. Essas propriedades foram avaliadas em um nível macro (entre países), meso (entre instituições) e micro (entre co-autores). Um dos achados desse artigo é que enquanto no nível macro a densidade, o coeficiente de clusterização, a conectividade e o *closeness* aumentaram à medida em que a rede crescia, o inverso ocorreu no nível micro.

Uzzi & Spiro [2005] estudaram a evolução da rede de colaboração entre artistas criativos da Broadway entre 1945 e 1989. Nesse trabalho, os autores observaram que propriedades que faziam variar o fenômeno *small-world* afetaram a criatividade dos artistas, que foi medida com relação ao desempenho artístico e financeiro dos musicais produzidos.

Com a ideia de construir um *framework* da dinâmica da evolução de redes de colaboração, Ramasco et al. [2004] estudaram três tipos de redes de colaboração - científica, de atores em filmes, e de diretores de empresas. Nesse estudo, os autores propuseram um modelo, constituído de uma rede bipartida com a característica de *preferential attachment*, mas sem incluir características como perda de nós e conexões. O modelo é genérico e, teoricamente, pode ser aplicado para qualquer rede de colaboração.

3.3 Redes de desenvolvimento colaborativo de software

O desenvolvimento colaborativo de software tem sido objeto de estudo de muitos pesquisadores. A maior parte desses trabalhos focam no GitHub, buscando entender como as estruturas sociais afetam a colaboração. Por exemplo, o trabalho de Lima et al. [2014] realiza uma caracterização do GitHub, enquanto rede social e rede de colaboração. Nessa análise, foram construídas várias redes: *followers graph*, uma rede direcionada contendo os usuários que seguem outros usuários; *collaborators graph*, uma rede bipartida, onde os nós repositórios estão conectados aos usuários colaboradores dos repositórios; *stargazers graph*, uma rede bipartida contendo repositórios e usuários que favoritaram um repositório; *contributors graph*, uma rede contendo a relação entre autor e commit. Esse trabalho comparou a distribuição de graus entre as redes construídas. Apesar de todas seguirem uma distribuição *scale-free*, os vértices das redes de colaboração foram os que apresentaram, em média, grau menor, e os vértices da rede social os que apresentaram, em média, grau maior. Verificou-se também que a colaboração entre usuários acontece em pequenos projetos. Nesse trabalho também foi analisado o impacto da proximidade geográfica na colaboração, que concluiu que há uma tendência maior na colaboração intra-países.

Dabbish et al. [2012] também buscaram explorar aspectos sociais no GitHub. Nesse trabalho, os autores entrevistaram usuários centrais e periféricos do GitHub para entender o valor de transparência dessas comunidades, isto é, de que forma os usuários veem as ações de outros usuários. Os resultados mostraram que há algumas características que levam os usuários a inferir comprometimento, qualidade do trabalho, significado da comunidade e relevância pessoal. Além disso, concluiu-se que essas inferências apoiam a colaboração e o gerenciamento da reputação da comunidade.

O trabalho de McDonald et al. [2014] analisa o conceito de *liderança distribuída* em cinco projetos do GitHub. A liderança distribuída é um pressuposto de que a liderança é compartilhada entre os membros de uma equipe. A liderança foi medida a partir de aná-

lises de *pull-requests*¹. Os resultados mostraram que projetos com altas taxas de sucesso entre colaboradores e alta retenção de colaboradores tendem a ter práticas distribuídas para revisão de *pull-requests*.

Casalnuovo et al. [2015] estudaram a importância que fatores sociais tem na manutenção da colaboração no GitHub. Os autores encontraram evidências de que desenvolvedores tendem a escolher colaborar em repositórios onde eles encontram colaboradores de relacionamentos anteriores, e inclusive suas contribuições são quantitativamente maiores na presença de relacionamentos mais fortes. No entanto, a presença desses laços não é suficiente para garantir contribuições contínuas por um longo período de tempo.

Em Allaho & Lee [2013], os autores realizaram análises estatísticas nas redes de colaboração emergentes sob o GitHub e o Ohloh. Para isso, eles utilizaram um modelo de rede, em que os vértices eram os desenvolvedores e uma aresta direcionada existia sempre que um desenvolvedor seguia (através da funcionalidade *follow*) outro desenvolvedor. O objetivo era investigar o efeito das estruturas de laços sociais sobre a produtividade dos repositórios. Métricas clássicas de redes complexas foram calculadas e as análises mostram que essas redes seguem distribuições de grau *scale-free* e características *small-world*. A assortatividade dessas redes foi negativa, indicando que colaboradores com grau alto tendem a se conectar com colaboradores de grau baixo. Além disso, foi calculada a correlação entre os graus e a produtividade em número de commits. Os resultados mostraram uma correlação positiva.

Também estudando o GitHub através da linha social, Hu et al. [2018] utilizaram um modelo que além da funcionalidade *follow* leva em consideração os repositórios marcados como favoritos ou que tiveram *fork*, e as organizações, às quais o usuário pertence. No entanto, esse modelo não trata da colaboração em si. Os estudos concentraram em análises de métricas de redes complexas para entender a influência dos desenvolvedores na rede.

Para entender a força da colaboração no GitHub, Batista et al. [2017] modelaram a colaboração em uma rede, em que os vértices são os colaboradores e há uma aresta sempre que dois colaboradores fizeram commit em um mesmo repositório. O trabalho selecionou todos os repositórios *non-forked* da linguagem JavaScript. Os autores coletaram métricas clássicas de redes complexas e compararam com outras três novas métricas propostas para medir a força da colaboração. Em um outro trabalho, Batista et al. [2018] com o mesmo modelo do anterior, analisaram as redes sob aspectos temporais de repositórios da linguagem Java, JavaScript e Ruby. Nesse trabalho, os autores demonstram a relevância da análise temporal, para evitar a formação de cliques e compreensões equivocadas do processo colaborativo.

O trabalho de El Asri et al. [2017] analisa redes de colaboração de alguns repositórios do GitHub selecionados, de diferentes linguagens e tempos de vida, e com pelo

¹Um *pull request* é uma notificação para o autor original do projeto, de que o trabalho recém publicado é uma modificação do outro e está pronto para ser unificado, tornando-se efetivamente uma colaboração.

menos mil colaboradores. Para cada um dos repositórios foram construídas três tipos de redes não direcionadas: (1) os vértices são os colaboradores e as arestas indicam a força da colaboração entre cada dois desenvolvedores baseada no número de arquivos que ambos editaram; (2) os vértices são os desenvolvedores que comentaram o commit de outro desenvolvedor, e as arestas foram ponderadas com a quantidade de commits que ambos interagiram juntos. (3) os vértices são os desenvolvedores que comentaram a revisão de código de outro colaborador e as arestas foram ponderadas pelo número de vezes que eles apareceram no mesmo comentário. Essas redes foram analisadas em janelas de tempo mensais, através de métricas de redes complexas.

Com o surgimento de plataformas de desenvolvimento colaborativo de software, projetos tem surgido e abandonados com uma velocidade sem precedentes. O trabalho de Coelho & Valente [2017] buscou entender as razões pelas quais os projetos são abandonados. Algumas razões encontradas foram: falta de tempo, falta de interesse, superado por um competidor, tecnologias ultrapassadas e baixa manutenibilidade. Além disso, apresentou algumas práticas que os mantenedores dos projetos seguem para evitar a morte dos projetos, como mover o repositório para uma conta de organização, novos mantenedores e aceitar novos colaboradores diretos (com direito a commit sem pull request).

Em Baudry & Monperrus [2012], os autores discorrem acerca de conceitos biológicos como ecossistema, biodiversidade e redes ecológicas, e traçam possíveis aplicações desses conceitos na Engenharia de Software com o objetivo de fornecer soluções para os desafios de construir software em larga escala abertos, mas estáveis. Assim, os autores argumentam, por exemplo, que a noção de biodiversidade pode ser aplicada ao software (em nível de código) para construir sistemas mais adaptáveis e estáveis, uma vez que biodiversidade traz estabilidade para sistemas biológicos.

Um outro trabalho muito parecido foi publicado por Mens & Grosjean [2015]. Os autores também buscam conceitos biológicos para aplicação na Engenharia de Software. Neste trabalho, eles utilizam ecossistemas de software como “uma coleção de projetos de software que são desenvolvidos e evoluem junto em um mesmo ambiente”. Nesse sentido, os autores argumentam a importância de considerar os diversos fatores que podem impactar na vida do software, incluindo os humanos, sempre fazendo analogias com conceitos biológicos.

3.4 Diferencial desta dissertação

Neste capítulo apresentamos trabalhos que buscam estudar a colaboração entre humanos à luz de modelos biológicos. Ainda não existe um consenso sobre quais meca-

nismos biológicos guiam a colaboração entre humanos, sobretudo porque esses estudos começaram recentemente. Também não foram encontrados trabalhos que analisassem o desenvolvimento colaborativo de software sob a inspiração desses modelos. Desse modo, esta dissertação tem a intenção de contribuir para essa discussão, oferecendo uma nova perspectiva da colaboração entre humanos, através do entendimento de como a colaboração entre desenvolvedores de software emerge e se perpetua no GitHub.

Nem todos os trabalhos que estudam as redes de desenvolvimento colaborativo de software utilizam o conceito de commit como unidade básica da colaboração. Conforme apresentado, muitos focam nos aspectos sociais, que estão relacionados a funcionalidades como *following*, *star* e comentários. Neste trabalho, nós modelamos redes de colaboração, em que a construção das arestas leva em consideração a existência de commits em um mesmo repositório. Nós acreditamos que como o commit é o resultado da colaboração, essa abordagem permite trazer informações diferentes daquelas encontradas analisando estrelas e comentários.

Os poucos trabalhos que utilizaram redes modeladas a partir do conceito de commits não levaram em consideração a forma temporal e linear da colaboração, ou quando o fizeram, focaram em repositórios de apenas uma única linguagem. Neste trabalho propomos um modelo que busca reproduzir o conceito biológico de ecossistema, e portanto, além de estudar as redes de forma temporal, considera toda a diversidade de repositórios e linguagens de programação. Dessa forma, acreditamos que é possível entender a evolução dos ecossistemas de forma mais fiel à realidade, pois ao focar a colaboração no colaborador, como requer os modelos biológicos, é possível estudar conjuntamente todas suas parcerias, o que pode ocorrer em diferentes repositórios simultaneamente, mas que poderiam afetar o ecossistema como um todo. No entanto, é importante ressaltar que a metodologia descrita nesta dissertação visa complementar as demais análises do processo de colaboração apresentadas na literatura, uma vez que introduz uma nova ótica em como modelar e compreender o processo de colaboração em sistemas de desenvolvimento de software.

Capítulo 4

Ecosistemas de Colaboração

Neste capítulo apresentamos a definição de ecossistemas de colaboração no escopo desta dissertação (Seção 4.1). A seguir, descrevemos a coleta e tratamento de dados do GitHub para a construção de ecossistemas de colaboração (Seção 4.2). A Seção 4.3 apresenta a caracterização desses ecossistemas. Por fim, a Seção 4.4 discute os resultados encontrados.

4.1 Definição de ecossistema de colaboração

O termo ecossistema foi mencionado pela primeira vez por Tansley [1935]. Com esse termo, Tansley [1935] defendeu que não apenas os seres vivos, mas também todos os fatores que possam influenciar o bioma fossem levados em consideração no estudo das “unidades básicas da natureza”. Com a inclusão dos fatores abióticos poderia-se observar os efeitos que eles causam sobre o bioma e vice-versa. Nessa definição, Tansley [1935] também adicionou que ecossistemas são “uma reconhecida entidade auto-contida”.

Lindeman [1942] mostrou a importância de se estudar o ecossistema dinamicamente, devido a existência de fluxos de energia e de *loops* de retroalimentação que fluem tanto entre os fatores bióticos, quanto entre os fatores bióticos e abióticos. Nesse sentido, ecossistemas foram caracterizados pela capacidade de sofrer perturbações e se recuperar delas. Isso deixa claro que a ideia de ecossistema armazena a história dos fatores bióticos e abióticos.

Como resultado da reunião do bioma com o abioma, ecossistemas são caracterizados por uma ampla diversidade. Tanto os fatores bióticos quanto abióticos apresentam características variantes entre si. Na biologia, ecossistemas variam entre desertos, florestas, oceanos, entre outros. Os fatores bióticos são adaptados para viver em cada um desses tipos de ecossistema.

Nesta dissertação, nos referimos a ecossistema de colaboração como um ecossistema modelado com foco na colaboração, isto é, um ecossistema que ao incluir os fatores bióticos e abióticos deixe evidente os relacionamentos colaborativos entre eles. Assim, um

ecossistema de colaboração se caracteriza por uma visão relativamente fechada no tempo de indivíduos colaborando entre si e o conjunto de meios e fatores envolvidos na colaboração. No GitHub, os fatores bióticos são representados pelos desenvolvedores, enquanto que os fatores abióticos são características do processo, como o repositório e a linguagem de programação utilizada.

Essa abordagem traz muitos benefícios: (1) como é capaz de reproduzir os relacionamentos colaborativos do indivíduo, pode-se enxergar de forma mais sistemática o processo colaborativo, isto é, como as partes que compõem esses sistemas influenciam umas às outras; (2) a ideia de ecossistema introduz a diversidade de “habitats” de colaboração que existe; (3) é possível examinar a influência que fatores abióticos possuem no processo colaborativo; (4) modelagem de estratégias de colaboração, baseadas, por exemplo, em reputação dos colaboradores participantes dos ecossistemas.

Como esta dissertação tem como foco o desenvolvimento colaborativo de software, vamos utilizar o conceito de ecossistema de colaboração apresentado para realizar nossas análises. A seguir, apresentamos de que forma utilizamos o GitHub para construir ecossistemas de colaboração, juntamente com uma caracterização sistemática deles.

4.2 Ecossistemas no GitHub

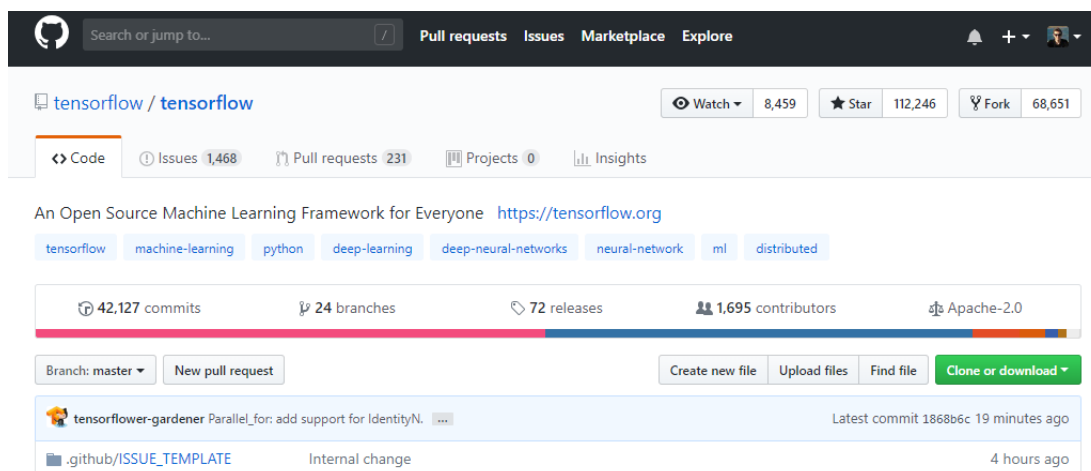
Nesta seção descrevemos o GitHub, o sistema web que utilizamos para analisar o desenvolvimento colaborativo de software. Também apresentamos que dados coletamos do GitHub, e como construímos ecossistemas de colaboração com esses dados.

Descrição do GitHub

O GitHub é um sistema web focado no desenvolvimento colaborativo de software. Ele está organizado em torno de repositórios, utilizando o sistema de controle de versões **git**. Um repositório pode pertencer a uma organização ou a um usuário. Contribuições aos repositórios podem ocorrer de duas maneiras: (1) a partir de *commits*¹ diretos ao repositório; (2) a partir de *pull requests*. Os desenvolvedores que foram designados com a permissão de fazer commits diretamente ao repositório são chamados membros-efetivos do repositório. Caso um desenvolvedor não tenha essa permissão, toda contribuição precisa

¹Um commit no sistema de versões **git** é uma publicação no repositório.

Figura 4.1: Página inicial de um repositório no GitHub exibindo suas funcionalidades



ser feita via *pull request*. Para fazer um *pull request*, o desenvolvedor precisa realizar um *fork* no repositório que deseja colaborar, isto é, precisa criar uma cópia deste. A contribuição é feita em cima dessa cópia. Um *pull request* é uma requisição para efetuar a fusão (*merge*) das alterações com o projeto original.

Qualquer discussão acerca da colaboração é feita a partir da funcionalidade *Issues*. Essa seção geralmente inclui solicitação de novas funcionalidades, rastreamento de itens a serem feitos e reporte de *bugs*. A gestão do desenvolvimento pode ser feita a partir da funcionalidade *Projects*, a partir da qual podem ser criados quadros de tarefas e alguns fluxos de trabalho podem ser automatizados.

GitHub permite manter a proximidade de um repositório com seus usuários a partir de duas funcionalidades: *watch* e *star*. Ao marcar *watch* em um repositório, o usuário passa a receber notificações de novos *pull requests* e *issues* que foram criadas. Ao marcar *star* para um repositório, o usuário indica que tem interesse naquele repositório e o GitHub passa a indicar projetos similares. A Figura 4.1 mostra essas funcionalidades na página inicial de um repositório no GitHub.

Desenvolvedores podem seguir uns aos outros a partir da funcionalidade *follow*. Assim, qualquer atividade feita pelo desenvolvedor seguido aparecerá em sua página inicial.

Conjunto de dados

Apesar de o GitHub fornecer uma REST API de acesso completo a seus dados, alguns desafios permeiam a mineração de dados. GitHub impõe um limite de 5000 requisições por hora para requisições autenticadas, enquanto o número de eventos estimados

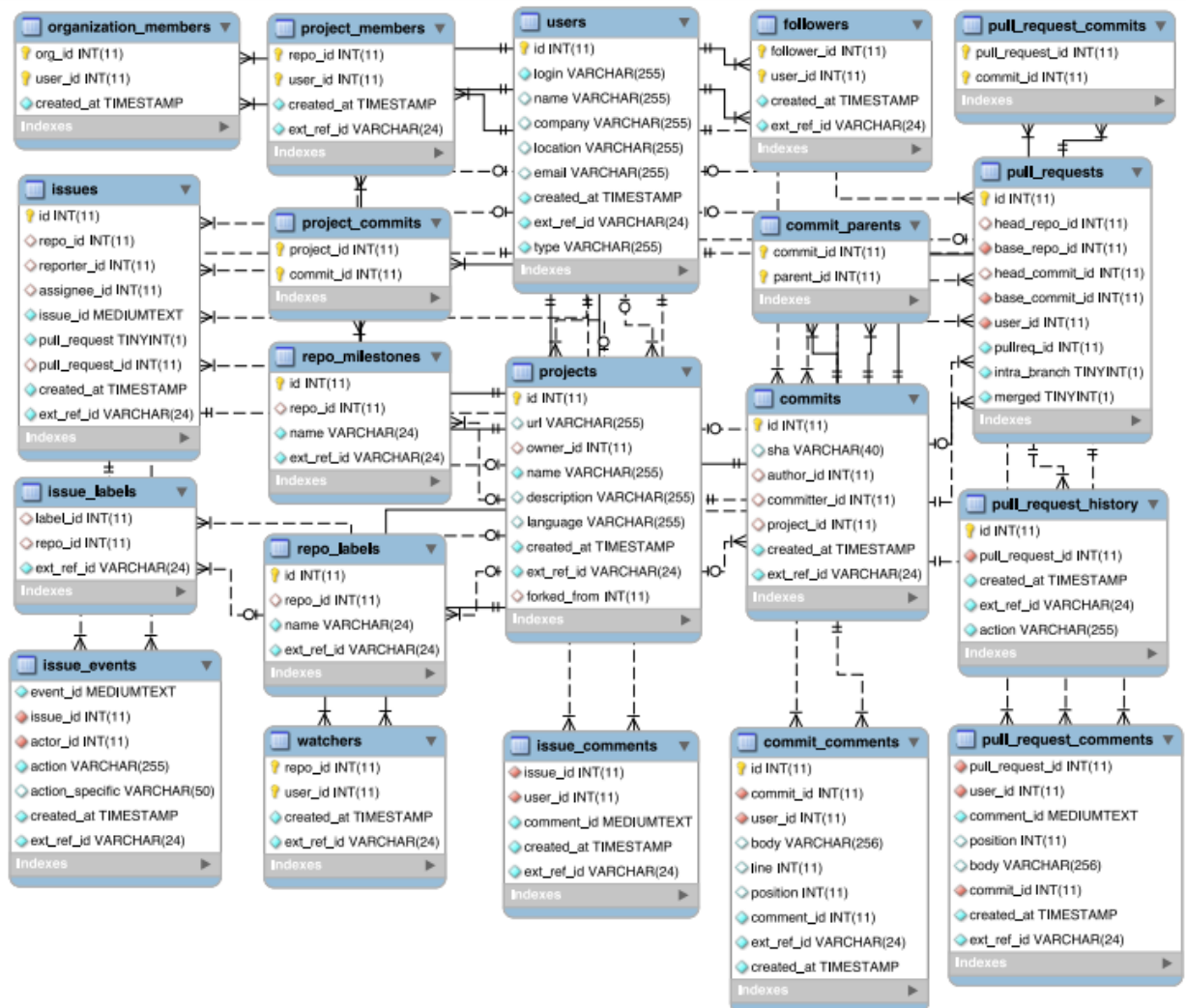


Figura 4.2: Esquema relacional de GHTorrent

ocorrendo no mesmo intervalo de tempo é muito maior. Desse modo, é impraticável que uma única conta consiga minerar quantidade suficiente de dados para uma reprodução fiel da sua dinâmica.

O projeto de código aberto GHTorrent [Gousios, 2013] surgiu com o propósito de tratar essa limitação, fornecendo um espelho do GitHub, possibilitando recuperar o histórico completo de um repositório e a lista completa de ações de um desenvolvedor. Este trabalho utilizou-se de um *dump* de GHTorrent disponibilizado em 1 de abril de 2018.

O esquema relacional dos dados disponibilizados pelo GHTorrent é apresentado na Figura 4.2. As tabelas podem ser agrupadas nas seguintes categorias:

Dados de usuário As tabelas *users* e *followers* armazenam dados como identificador, nome, login e data de criação da conta no GitHub, relacionamentos entre usuários e seguidores.

Repositórios As tabelas *projects*, *project_members*, *organization_members*, *repo_milestones*, *repo_labels*, *watchers* armazenam informações dos repositórios como nome, descrição, linguagem principal, data de criação e repositório de origem, caso seja um *fork*; além da relação de membros da organização dona do repositório (caso seja uma organização), rótulos e observadores.

Commits As tabelas *commits*, *commits_parents*, *commit_comments*, *project_commits* armazenam os dados relacionados a commits, como a data de criação, usuário e repositório envolvidos nos commits.

Pull requests As tabelas *pull_requests*, *pull_request_commits*, *pull_request_history* e *pull_request_comments* contém todas os dados relacionados a pull request, como por exemplo o ponto exato da árvore de versões que se propõe fundir, e informação se ele foi completado ou não.

Issues Dados relacionados às questões colocadas em discussão pela comunidade estão armazenados nas tabelas *issues*, *issue_labels*, *issue_events* e *issue_comments*.

Ecosistemas

Os dados coletados (50GB compactados) não foram utilizados integralmente. GitHub mantém uma página² onde exibe repositórios mais favoritados a partir do número de marcações *star* que cada repositório recebe. Esse número pode ser interpretado como o grau de disponibilidade (ou vontade) que um usuário tem em colaborar em repositórios similares. Para as análises apresentadas nesta dissertação, selecionamos três repositórios com diferentes tendências de crescimento no número total de colaboradores, bem como diferentes linguagens de programação. Escolhemos trabalhar com repositórios de linguagens diferentes, com o principal objetivo de entender como a linguagem interfere na dinâmica da colaboração. Além disso, os repositórios escolhidos têm números de commits diferentes, o que nos permite classificá-los em repositórios com alta, média e baixa intensidade de colaboração.

Para criar ecossistemas de colaboração no GitHub, utilizamos esses três repositórios selecionados como sendo as raízes de cada um dos ecossistemas. Esses repositórios são denominados repositórios-raiz dos ecossistemas, e serão utilizados para nomeá-los ao longo do texto. A criação de cada ecossistema segue o Algoritmo 1. Esse algoritmo foi elaborado com o intuito de reproduzir o conceito biológico de ecossistema, definido pelo conjunto

²A lista de repositórios pode ser consultada em <https://github.com/trending>

de comunidades (colaboradores de um repositório) e outros fatores não vivos (como por exemplo, repositórios e linguagens de programação). Um ecossistema é caracterizado pela ampla diversidade que apresenta. A partir do algoritmo proposto, é possível atingir essa diversidade, o que diferencia este trabalho de muitos na literatura, que geralmente focam em um conjunto de linguagens de programação ou em repositórios específicos. Aos dados coletados, foi aplicado um filtro para considerar a colaboração durante 10 anos, compreendendo o início de 2008 até o fim de 2017.

Algoritmo 1: Cria ecossistema de colaboração do GitHub

Entrada: *RepositorioRaiz*

Saída: *Commits* (conjunto de commits)

Commits $\leftarrow \emptyset$

Repositorios \leftarrow *RepositorioRaiz*

DesenvolvedoresComputados $\leftarrow \emptyset$

repita

Desenvolvedores \leftarrow *RecuperaDesenvolvedores*(*Repositorios*)

Desenvolvedores \leftarrow *Desenvolvedores* – *DesenvolvedoresComputados*

se *Desenvolvedores* = \emptyset **então**

 | **retorna**

fim

c \leftarrow *RecuperaCommits*(*Desenvolvedores*)

Repositorios \leftarrow *RecuperaRepositoriosDosCommits*(*c*)

Commits \leftarrow *Commits* \cup *c*

DesenvolvedoresComputados \leftarrow

DesenvolvedoresComputados \cup *Desenvolvedores*

até *Repositorios* $\neq \emptyset$

Para entender a execução do algoritmo, suponha que há um desenvolvedor no repositório raiz. Na primeira iteração do algoritmo, são recuperados todos os commits realizados pelo desenvolvedor do repositório raiz durante toda sua vida. Isto é, se o desenvolvedor colaborou em três repositórios com 1 commit em cada, vamos armazenar os commits em um conjunto e os repositórios em outro. Na segunda iteração, recuperamos os desenvolvedores que colaboraram em cada um dos três repositórios da iteração anterior. Sabendo quem são os desenvolvedores, recuperamos novamente todos os commits que eles realizaram ao longo da vida. Esses commits são unidos ao conjunto de commits, e os repositórios são adicionados ao conjunto de repositórios para serem analisados na próxima iteração. As iterações prosseguem até que não haja mais repositórios a serem recuperados, o que significa que o ecossistema está fechado. É possível alterar o algoritmo para executar um número desejado de iterações. Neste trabalho, utilizamos 3 iterações devido ao grande volume de dados. Utilizando 3 iterações, garantimos a recuperação da história completa dos desenvolvedores recuperados nas 2 iterações anteriores.

Os ecossistemas analisados nesta dissertação foram criados a partir dos repositórios-raiz TensorFlow, Spring e SignalR, de linguagens de programação C++, Java

e C#, respectivamente.

TensorFlow é uma plataforma de código aberto para aprendizado de máquina. Oferece um conjunto de ferramentas para auxiliar desenvolvedores na construção de softwares que aplicam conceitos de aprendizagem de máquina. O repositório TensorFlow contava com 73.328 commits, 2.293 colaboradores e 138.000 *stars* no momento da escrita desta dissertação.

Spring é um framework que estende a linguagem Java para a criação de aplicações empresariais em uma variedade de cenários e arquiteturas. O repositório Spring contava com 20.042 commits, 418 colaboradores e 33.900 *stars* no momento da escrita desta dissertação.

SignalR é uma biblioteca para ASP.NET que permite adicionar funcionalidades de tempo real nas aplicações Web, isto é, funcionalidades que requerem que o servidor envie conteúdos para o cliente em tempo real. No momento da escrita desta dissertação, o repositório SignalR contava com 4.681 commits, 80 colaboradores e 7.900 *stars*.

4.3 Caracterização dos ecossistemas de colaboração

Os ecossistemas TensorFlow, Spring e SignalR foram caracterizados quanto a distribuição de commits (Seção 4.3.1), desenvolvedores (Seção 4.3.2), e linguagens e *forks* (Seção 4.3.3). A Tabela 4.1 apresenta as principais características dos ecossistemas analisados.

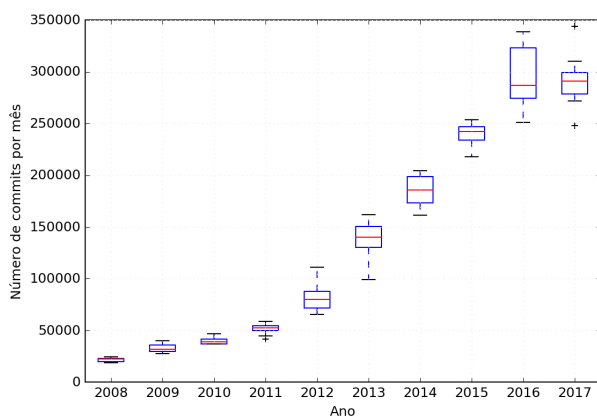
	TensorFlow	Spring	SignalR
# Commits	16.500.170	4.171.663	1.550.549
# Repositórios	52.585	14.559	5.895
# Desenvolvedores	1.203.348	397.163	178.770
# Linguagens	153	65	65

Tabela 4.1: Sumário dos ecossistemas de colaboração analisados

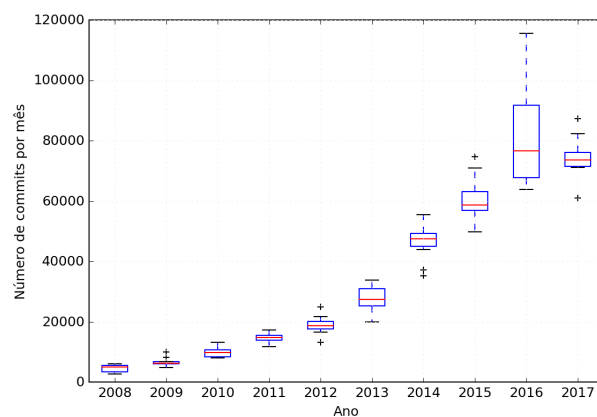
4.3.1 Commits

Entre os ecossistemas analisados, o maior número de commits foi encontrado no ecossistema TensorFlow, com cerca de 16,5 milhões de commits, seguido de Spring com pouco menos de 4,2 milhões de commits e SignalR com aproximadamente 1,5 milhão de commits. O número de commits é um bom indicador do grau de colaboração de cada ecossistema, dado que o commit é o resultado da colaboração no GitHub.

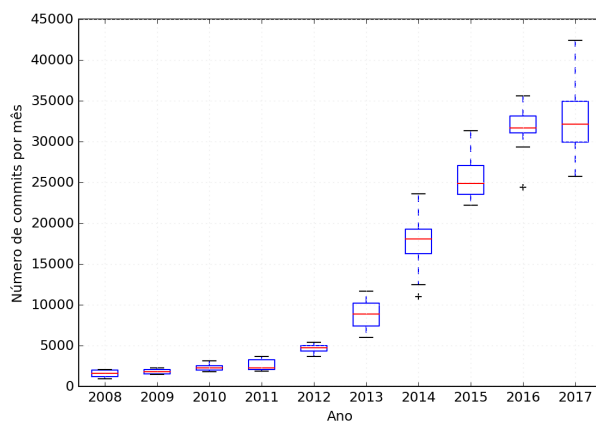
A Figura 4.3 mostra a distribuição de commits por mês em cada um dos ecossistemas em um período de 10 anos completos. A colaboração em cada ano foi sumarizada por meio de boxplots. Em todos os ecossistemas, é possível observar o crescimento gradual da colaboração, acentuando-se a partir do ano de 2012 no ecossistema TensorFlow e em 2013 nos demais. Nos primeiros anos, o número de commits variou pouco durante os meses do ano, passando a variar mais nos anos em que o número de commits foi maior. No último ano analisado, a produção caiu nos ecossistemas TensorFlow e Spring – única vez em toda a série.



(a) Ecossistema TensorFlow



(b) Ecossistema Spring



(c) Ecossistema SignalR

Figura 4.3: Evolução dos commits ao longo do tempo

O número absoluto de repositórios recebendo commits também teve um comportamento crescente ao longo dos anos, conforme mostra a Figura 4.4. O ecossistema TensorFlow tinha no último ano observado 24410 repositórios, um crescimento de 58,5 vezes com relação a 2008. O ecossistema Spring, por sua vez, atingiu 5550 repositórios em 2017, o que representa 68,8 vezes o número de repositórios em 2008. Por fim, SignalR contava com 2805 repositórios em 2017, um aumento de 187 vezes o número observado em 2008.

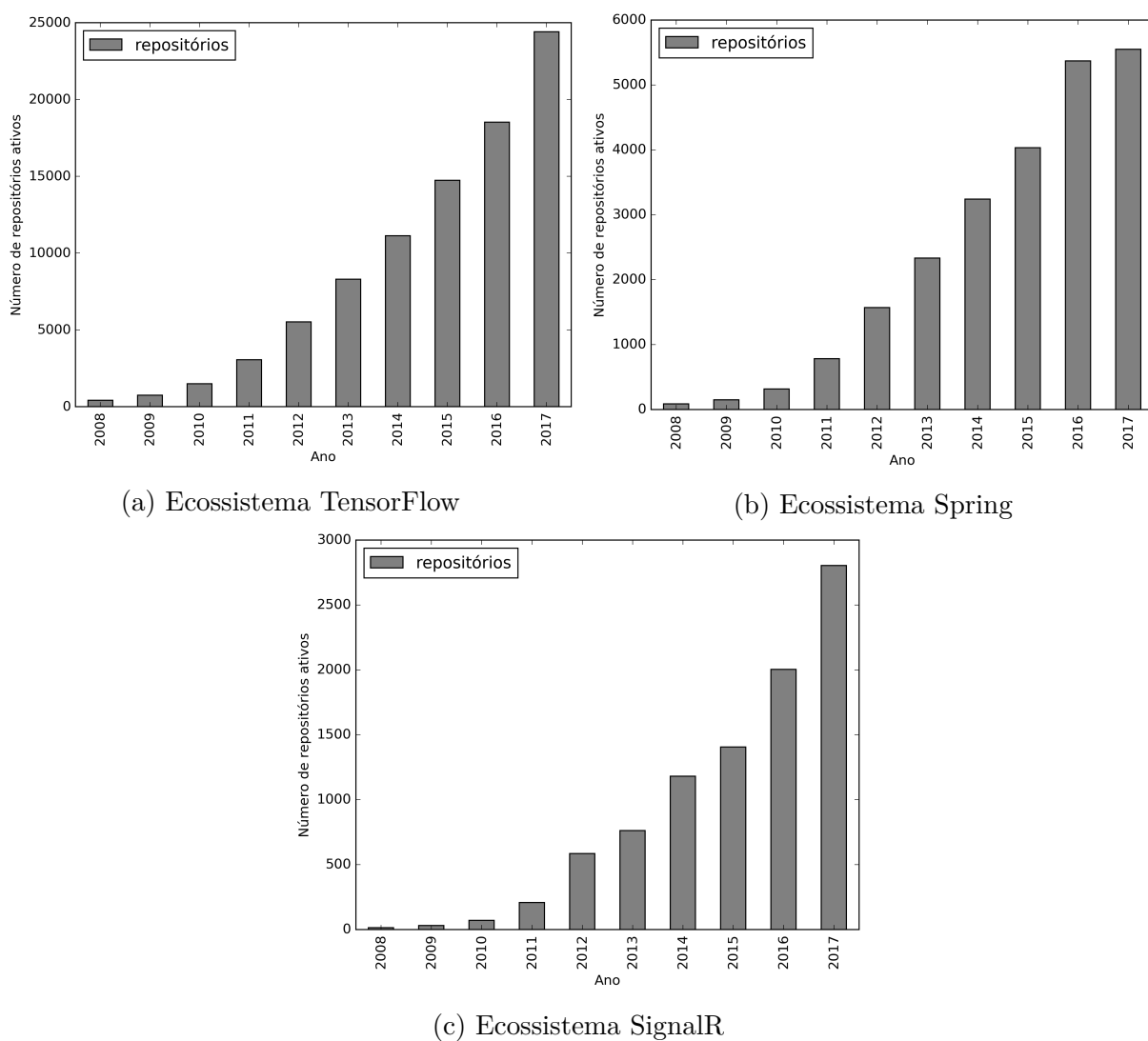


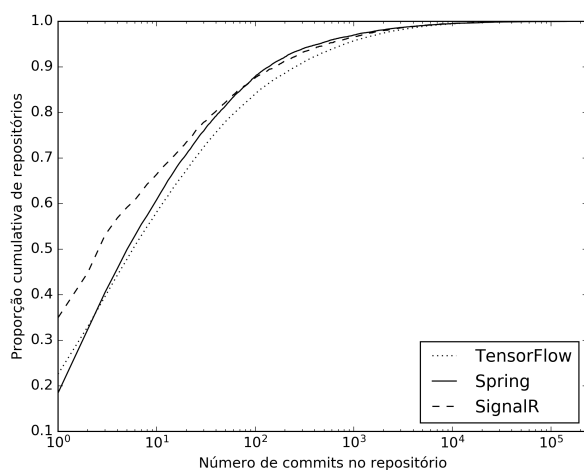
Figura 4.4: Repositórios ativos ao longo do tempo

A maior parte dos repositórios que compõe os ecossistemas têm poucos commits, como pode ser visto na Figura 4.5a. A distribuição cumulativa é muito concentrada, com metade dos repositórios tendo até 6 commits nos ecossistemas TensorFlow e Spring, e 3 commits no ecossistema SignalR. Somente cerca de 10% dos repositórios tem mais de 100 commits. O número máximo de commits que um repositório teve nos ecossistemas TensorFlow, Spring e SignalR foi, respectivamente, 234.590, 184.133, e 175.242.

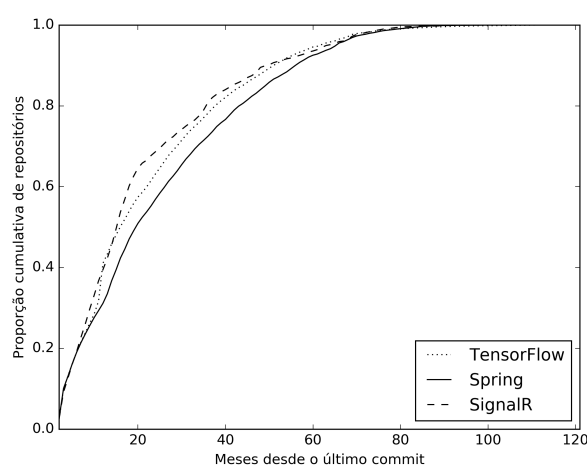
O baixo número de commits na maior parte dos repositórios conduz à investigação acerca do tempo em que os repositórios receberam seu último commit. A Figura 4.5b

mostra a proporção de repositórios que receberam commits nos últimos x meses. Cerca de 63% dos repositórios estiveram ativos (recebendo commits) nos últimos 24 meses analisados (jan/2016 a dez/2017) para o ecossistema TensorFlow. No ecossistema Spring esse número cai para aproximadamente 57%. A maior taxa de repositórios ativos no período foi observada no ecossistema SignalR, de 69%.

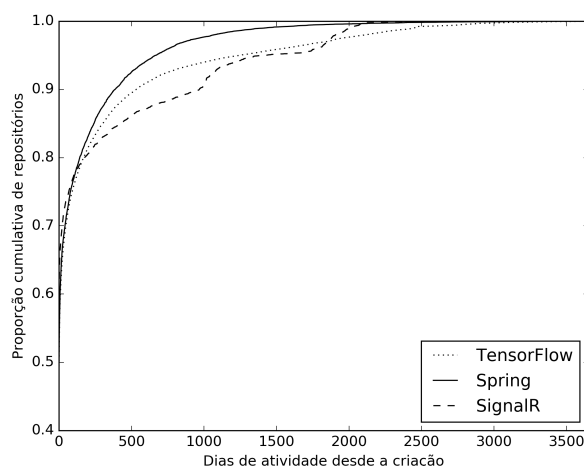
O tempo de vida dos repositórios, isto é, a diferença entre a data de criação e o último commit, também foi medido para todos os ecossistemas. A Figura 4.5c apresenta esses resultados. A mediana do tempo de vida dos repositórios é de 9 dias nos ecossistemas TensorFlow e Spring e de 2 dias no ecossistema SignalR. Apenas 13% dos repositórios tem duração maior que 1 ano no ecossistema TensorFlow. Essa taxa é de 11% no ecossistema Spring e de 16% em SignalR.



(a) Proporção cumulativa de repositórios com um dado número de commits. A maioria dos repositórios tem poucos commits.



(b) Proporção cumulativa de repositórios ativos durante os últimos x meses desde 1º de janeiro de 2008.



(c) Proporção cumulativa de repositórios que tiveram atividade nos últimos x dias desde a sua criação.

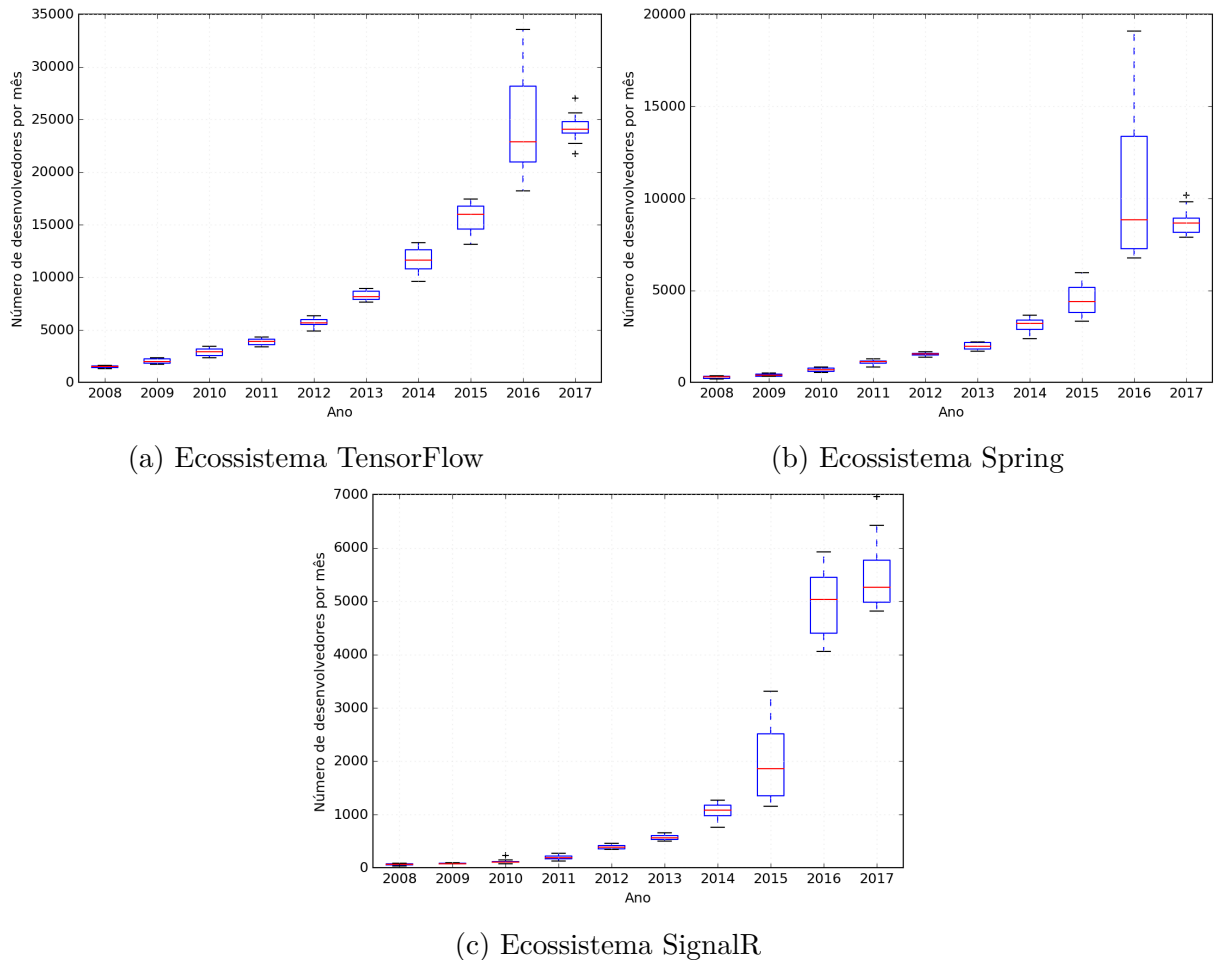
Figura 4.5: Visão geral da colaboração nos repositórios

4.3.2 Desenvolvedores

Durante o período de 10 anos analisados, 1.203.348 desenvolvedores únicos colaboraram no ecossistema TensorFlow, 397.163 no ecossistema Spring, e 178.770 em SignalR. A Figura 4.6 exibe a variação do número de desenvolvedores realizando commits por mês nos ecossistemas. Houve um crescimento gradual do número de desenvolvedores colaborando, para todos os anos analisados, exceto no ano de 2017. A variação de desenvolvedores durante os meses do ano foi baixa até o ano de 2014, tornando-se maior a partir de então.

A evolução do número de desenvolvedores colaborando nos ecossistemas possui um comportamento muito similar à evolução do número de commits e podem estar correlacionadas. Para comprovar essa hipótese, realizamos um teste de correlação de Pearson entre o número de desenvolvedores e o número de commits por mês, no período de dez anos (tamanho da amostra = 120 e $p < 0.01$). Para todos os ecossistemas, encontramos uma correlação forte entre as variáveis com p-value < 0.00001 , sendo $r = 0,8803$ no ecossistema TensorFlow; $r = 0,9310$ no ecossistema Spring; e $r = 0,9141$ em SignalR. Essa alta correlação é esperada: o aumento do número de desenvolvedores pode gerar o aumento da força de trabalho. No entanto, não podemos afirmar a existência de causalidade, ou seja, podemos ter um número elevado de novos desenvolvedores colaborando no ecossistema, mas somente poucos desenvolvedores efetivamente realizam commits. No Capítulo 6, analisamos mais detalhadamente os mecanismos utilizados pelos desenvolvedores para que a colaboração entre os pares evolua e persista nestes ecossistemas.

Essa análise não diferencia novos desenvolvedores daqueles que já estavam colaborando nos ecossistemas. A Figura 4.7 compara ano a ano o número de novos desenvolvedores com o número de desenvolvedores que já colaboravam nos ecossistemas. No ecossistema TensorFlow, o número de novos desenvolvedores nunca superou o número de desenvolvedores antigos em nenhum ano. O maior valor registrado para a relação entre novos e antigos desenvolvedores foi 0,55 no ano de 2016. Diferentemente, os ecossistemas Spring e SignalR tiveram mais participação de novos desenvolvedores do que de desenvolvedores antigos na maior parte dos anos observados. Novamente, foi no ano de 2016 que ocorreu o maior valor para essa relação, sendo de 1,19 no ecossistema Spring e 1,46 em SignalR.



(a) Ecosistema TensorFlow

(b) Ecosistema Spring

(c) Ecosistema SignalR

Figura 4.6: Evolução do número de desenvolvedores nos ecossistemas

4.3.3 Linguagens e forks

Além do estudo da intensidade de atividades nos ecossistemas, analisamos também as linguagens de programação nos diferentes repositórios e como os desenvolvedores utilizam a ferramenta *fork*.

Os ecossistemas possuem repositórios com linguagens principais muito diversas. O ecossistema TensorFlow, por exemplo, possui repositórios de 153 linguagens diferentes. Esse número é quase 3 vezes menor nos ecossistemas Spring e SignalR (65), mas ainda preserva a diversidade de linguagens.

Realizamos duas análises com relação às linguagens de programação: (1) quantos repositórios utilizavam cada linguagem de programação e, (2) quantos commits são feitos em cada linguagem. A Tabela 4.2 mostra as dez linguagens com o maior número de repositórios nesses ecossistemas. Por sua vez, a Tabela 4.3 apresenta as dez linguagens com o maior número de commits para cada um dos ecossistemas analisados. O termo “Indefinido” sumariza o total de repositórios ou commits em que não há informação de

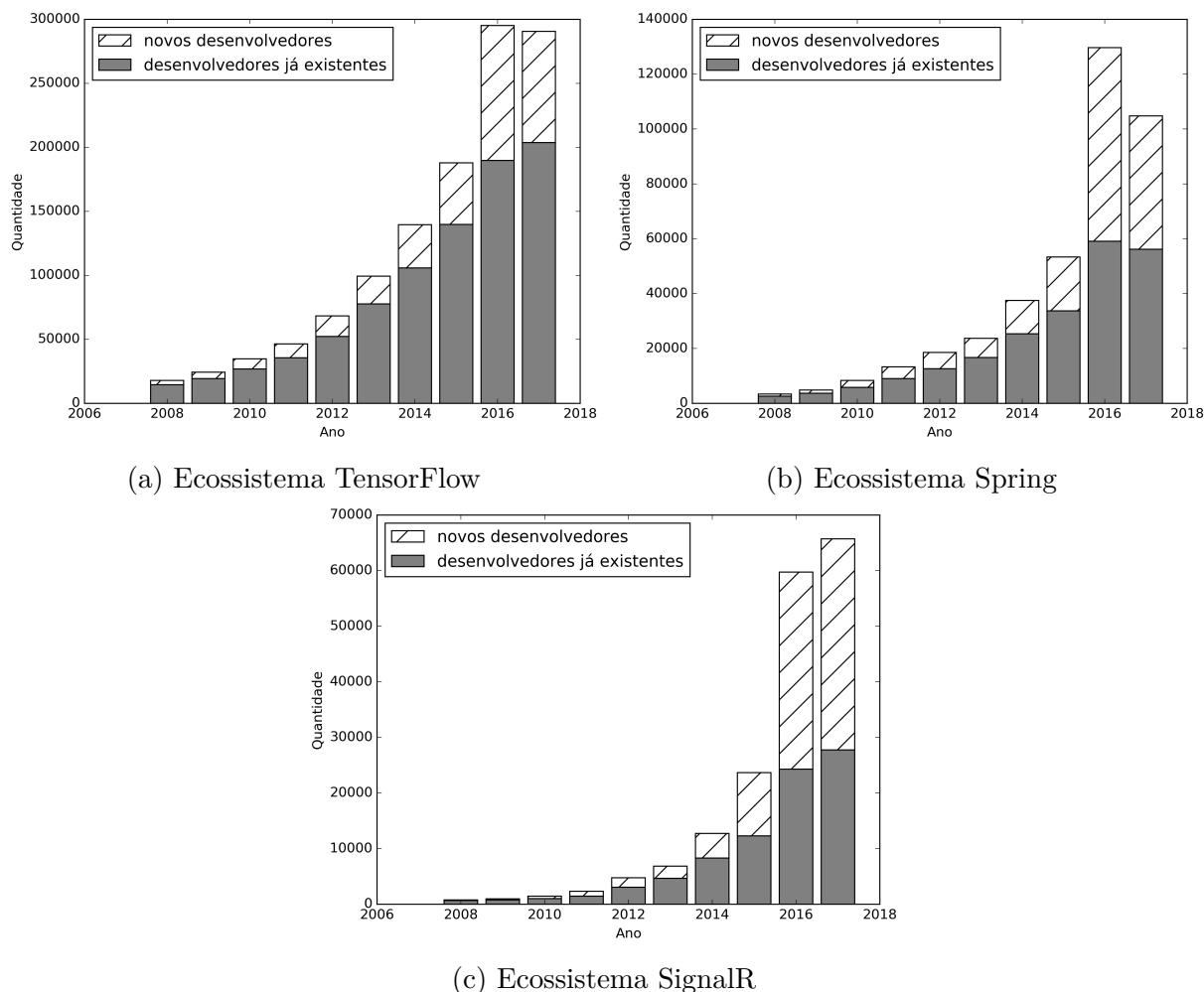


Figura 4.7: Entrada de novos desenvolvedores nos ecossistemas em comparação com os desenvolvedores já existentes

linguagem nos dados disponibilizados por GHTorrent.

Em todos os ecossistemas, muitas linguagens que aparecem entre as dez primeiras em um dos *rankings* também aparece em outro. No entanto, o posicionamento não se mantém. Calculamos a correlação de Spearman (r_s) entre o *ranking* de linguagens por repositório e o *ranking* de linguagens por commit. Para o ecossistema TensorFlow, $r_s = 0,742$, o que demonstra uma correlação alta; no entanto, para o ecossistema Spring ($r_s = 0,558$) e SignalR ($r_s = 0,153$) as correlações são insignificantes (para $\alpha = 0,05$; tamanho da amostra = 10; $r_c = 0,648$).

Outra característica observada é que a linguagem do repositório-raiz dos ecossistemas nem sempre é a primeira linguagem nos posicionamentos, mas está pelo menos entre as três primeiras.

Conforme mencionado na Seção 4.2, desenvolvedores sem direito de commit no repositório devem criar um *fork*, realizar as alterações necessárias e solicitar um *pull request* para que a colaboração seja efetivada no repositório de origem. No ecossistema TensorFlow, 39% dos repositórios têm pelo menos um *fork*. Essa porcentagem é similar

Posição	TensorFlow		Spring		SignalR	
	Linguagem	# repositórios	Linguagem	# repositórios	Linguagem	# repositórios
1	Python	8.643	Java	6.251	JavaScript	1.970
2	JavaScript	4.810	JavaScript	1.289	C#	1.097
3	C++	3.915	Shell	819	CoffeScript	208
4	Java	3.389	Ruby	427	Python	171
5	C	2.741	Groovy	303	HTML	156
6	Go	1.635	CSS	292	Java	127
7	Ruby	1.630	HTML	222	CSS	123
8	HTML	1.210	Python	218	Ruby	118
9	Julia	1.067	C	179	C++	100
10	Shell	1.034	Scala	167	Shell	72
#	Outras	8.953	Outras	862	Outras	535
#	Indefinido	13.558	Indefinido	3.530	Indefinido	1.218

Tabela 4.2: Ranking das 10 linguagens de programação com o maior número de repositórios

Posição	TensorFlow		Spring		SignalR	
	Linguagem	# commits	Linguagem	# commits	Linguagem	# commits
1	C	2.371.050	Java	1.343.290	C#	469.123
2	C++	2.326.058	Ruby	372.591	Shell	187.890
3	Python	2.284.208	JavaScript	280.977	Java	161.039
4	Java	1.544.964	Python	267.873	Python	125.606
5	Ruby	1.026.953	Shell	211.124	JavaScript	91.241
6	JavaScript	799.907	C	195.007	PHP	73.534
7	Go	602.837	Go	184.524	Go	59.461
8	Shell	417.760	C++	149.138	Ruby	56.820
9	HTML	274.198	Scala	108.017	C++	35.621
10	Scala	250.390	Nix	94.337	Scala	33.596
#	Outras	1.844.603	Outras	246.128	Outras	180.824
#	Indefinido	3.033.735	Indefinido	787.208	Indefinido	107.787

Tabela 4.3: Ranking das 10 linguagens de programação com o maior número de commits

no ecossistema Spring, com *forks* ocorrendo em 32% dos repositórios. Diferentemente, o ecossistema SignalR apresenta mais da metade (55%) dos seus repositórios com pelo menos um *fork*.

A Figura 4.8 apresenta a distribuição cumulativa do número de *forks* por repositório, para cada um dos ecossistemas. Em todos os ecossistemas, a maior parte dos repositórios possui apenas um *fork*: apenas 13,7% dos repositórios do ecossistema TensorFlow possuem dois ou mais *forks*; essa porcentagem é de cerca de 20,6% no ecossistema Spring e de 4,3% em SignalR.

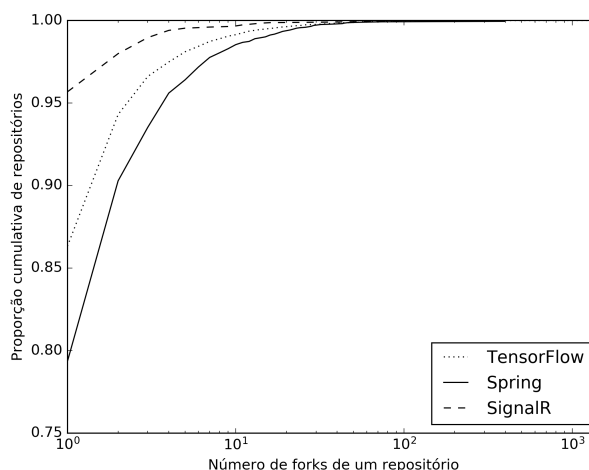


Figura 4.8: Proporção cumulativa de repositórios com um determinado número de *forks*. Este gráfico somente inclui repositórios com pelo menos um *fork*.

4.4 Discussão

Ao introduzirmos o conceito de ecossistema nesta dissertação, buscamos compreender, sob uma nova ótica, a dinâmica do processo de colaboração no GitHub. Como principal diferença entre os demais trabalhos na literatura, definimos um “sistema” fechado, que armazena a história dos fatores bióticos (desenvolvedores) e abióticos (repositórios e linguagens de programação). Uma das características principais é a presença da diversidade em diferentes aspectos, como por exemplo, linguagens de programação e repositórios de diferentes propósitos. A partir da caracterização dos três diferentes ecossistemas, concluímos que:

A intensidade da colaboração aumenta em todos os ecossistemas, porém com taxas diferentes. Houve crescimento gradual do número de commits ao longo dos anos, porém a variação ocorreu de forma diferente em cada ecossistema. O ecossistema TensorFlow sofreu uma evolução mais rápida da colaboração, se comparado com os demais. Algumas explicações possíveis estão relacionadas ao número de desenvolvedores, que é o maior entre os ecossistemas; ou até mesmo o empenho de alguns desenvolvedores ser maior que o empenho de outros.

A maior parte dos repositórios possuem poucos commits. A ocorrência de poucos repositórios com mais de 100 commits foi uma característica comum a todos os ecossistemas, o que pode ter relação com a estrutura oferecida pelo GitHub, uma vez que colaboradores que não são membros-efetivos de repositórios necessitam criar um *fork* do mesmo antes de colaborar. No entanto, a nossa análise irá considerar todos os repositórios pertencentes a cada ecossistema, como forma de garantir uma caracterização mais detalhada da história da colaboração em diferentes perfis de desenvolvedores e tipos

de repositórios.

Os ecossistemas estão “vivos”. Nossas análises mostraram que grande parte dos repositórios que compõem os ecossistemas estão ativos e recebendo commits. A porcentagem de repositórios ativos nos últimos 24 meses analisados é similar entre os ecossistemas, fortalecendo a ideia de que o comportamento evolutivo deles é semelhante. A porcentagem de repositórios com tempo de vida maior que 1 ano também é similar entre os ecossistemas. Em TensorFlow, porém, o tempo de vida dos repositórios é maior, o que poderia novamente sugerir mais empenho dos desenvolvedores desse ecossistema. Porém, novas análises são necessárias para entender o que provocou essa diferença.

A intensidade da colaboração, considerando a métrica número de commits, aumenta à medida que novos desenvolvedores participam do ecossistema. Apesar de o número de commits aumentar com a entrada de novos colaboradores, dando indícios de que novos colaboradores aumentam a “força de trabalho”, no caso de TensorFlow, o total de desenvolvedores novos é sempre menor do que os desenvolvedores que já fazem parte dos ecossistemas. Isso pode indicar duas coisas: (1) novos colaboradores são mais ativos, ou (2) existem colaboradores fieis, que sustentam a colaboração. No Capítulo 6, esse processo será analisado através do mecanismo de *sinergismo*.

Os ecossistemas são diversos em linguagens de programação. Os três ecossistemas possuem um número grande de repositórios com linguagens de programação diferentes. TensorFlow é 3 vezes mais diverso que os demais, o que pode ser uma característica específica dos repositórios que o compõe. Dependendo do propósito do software, é comum combinar várias linguagens, o que torna seus desenvolvedores mais abertos quanto ao uso de linguagens de programação. É relevante destacar que existe diversidade inclusive entre os ecossistemas: existem linguagens nos rankings que construímos que não são comuns a todos os ecossistemas.

Forks são muito utilizados. *Forks* correspondem a mais da metade (55%) dos repositórios que compõe o ecossistema SignalR, o que indica que o processo de colaboração nesse ecossistema é realizado mais frequentemente por membros não-efetivos do que por membros efetivos. Como a evolução desse ecossistema é mais lenta do que os demais, pode-se ponderar se o mecanismo de colaboração por *forks* gera resultados positivos ou negativos para o processo de colaboração. Repositórios que utilizam *forks* estão interessados em contribuições de qualidade, pois cada uma das contribuições será avaliada antes de ser efetivada. No entanto, precisamos de mais análises para verificar se esse controle de qualidade pode inibir a colaboração entre desenvolvedores.

A caracterização dos ecossistemas definidos nesta dissertação fornece indícios relevantes para o estudo mais detalhado do processo de colaboração entre os seus desenvolvedores, como por exemplo a diversidade de linguagens, repositórios e a intensidade das colaborações estabelecidas. No Capítulo 5, apresentamos esses ecossistemas modelados como redes complexas para entender se a estrutura topológica influencia diferentes aspectos

tos da colaboração. No Capítulo 6, analisamos essas redes segundo modelos biológicos para entender os mecanismos utilizados na evolução da colaboração.

Capítulo 5

Redes de desenvolvimento colaborativo de software

Neste capítulo definimos o modelo matemático utilizado para a caracterização e análise das colaborações estabelecidas nos ecossistemas TensorFlow, Spring e SignalR (Seção 5.1). A partir do modelo definido, aplicamos métricas encontradas na literatura para caracterizarmos como os desenvolvedores pertencentes aos ecossistemas se relacionam entre si e com os repositórios (Seção 5.2). Finalmente, a Seção 5.3 discute os resultados à luz do modelo proposto.

5.1 Modelo e Métricas

Nesta dissertação, um ecossistema de colaboração do GitHub é composto pelos metadados de colaboração de inúmeros desenvolvedores, que possuem atividade em repositórios de diferentes tamanhos e linguagens de programação, e que, em algum momento tiveram relação, ainda que indireta, com um repositório denominado raiz, por meio do qual foram recuperados. As caracterizações realizadas nesses ecossistemas e apresentadas no Capítulo 4, no entanto, não levaram em conta a estrutura topológica da colaboração ao longo do tempo, isto é, como as relações entre os colaboradores e os repositórios surgiram e evoluíram durante o período observado. A história do processo colaborativo é essencial para entender a colaboração sob a inspiração de modelos biológicos, por exemplo, para entender se um usuário com boa reputação atrai outros desenvolvedores para colaborar com ele, precisamos observar sua reputação em um tempo t e seus colaboradores em um tempo $t+1$. Assim, modelamos os ecossistemas como redes dinâmicas.

Modelo

Um ecossistema de colaboração, observado durante T unidades de tempo, é modelado como um conjunto de grafos (redes) $\mathcal{G}_T = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_n\}$. Cada grafo $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$, $1 \leq t \leq n$, representa um snapshot do ecossistema durante δ unidades de tempo, com $1 \leq \delta \leq T$. Em um grafo \mathcal{G}_t , \mathcal{V}_t é o conjunto de vértices e \mathcal{E}_t é o conjunto de arestas. A duração do snapshot δ está associada ao grau de dinamismo da rede: para redes altamente dinâmicas, um valor alto de δ pode não capturar a dinâmica topológica, enquanto que um valor baixo pode não capturar a colaboração entre os usuários.

Neste modelo, os vértices representam os colaboradores. Um vértice $v_t \in \mathcal{V}_t$ é uma terna $v_t = (i, C_t, L_t)$, onde i é o identificador do colaborador; C_t é um conjunto de commits realizados no tempo t , onde um $c_j \in C_t$ é uma dupla $c_j = (r, q_j)$, em que r é o identificador do repositório e q_j , a quantidade de commits realizada no tempo t ; e L_t é um conjunto de linguagens utilizadas na colaboração no tempo t , onde um $l \in L_t$ é uma dupla $l = (r, n)$, em que r é o identificador do repositório e n é o nome da linguagem de programação principal do repositório.

Uma aresta $e_t \in \mathcal{E}_t$ é uma terna $e_t = (u_t, v_t, R_t)$, onde u_t e v_t são os identificadores dos colaboradores u e v , e R_t é o conjunto de repositórios em que ambos colaboraram. Cada $r \in R_t$ é representado pelo identificador do repositório.

Considere dois vértices u e v . Se cada um deles realizou pelo menos um commit no mesmo repositório então há uma aresta não-direcionada (u, v) . Independente do número de repositórios em que ambos colaboraram, apenas uma aresta é representada para eles. Além disso, dado o ecossistema a ser modelado, uma aresta pode existir em um snapshot \mathcal{G}_t , mas não em um snapshot \mathcal{G}_{t+1} . O mesmo ocorre com os colaboradores: se um colaborador deixa de colaborar no tempo $t+1$, ele deixa de aparecer na rede. A Figura 5.1 mostra um exemplo da dinâmica dessas redes. Observe que no tempo $t + 1$, o desenvolvedor E deixou de colaborar e os desenvolvedores A, B e D começaram a colaborar em um mesmo repositório, deixando a rede com apenas um componente.

Métricas

Para caracterizar os ecossistemas de colaboração, as seguintes métricas clássicas de redes complexas [Newman, 2003] foram utilizadas:

Grau O grau k de um vértice v é definido como o número de arestas que se conectam a ele.

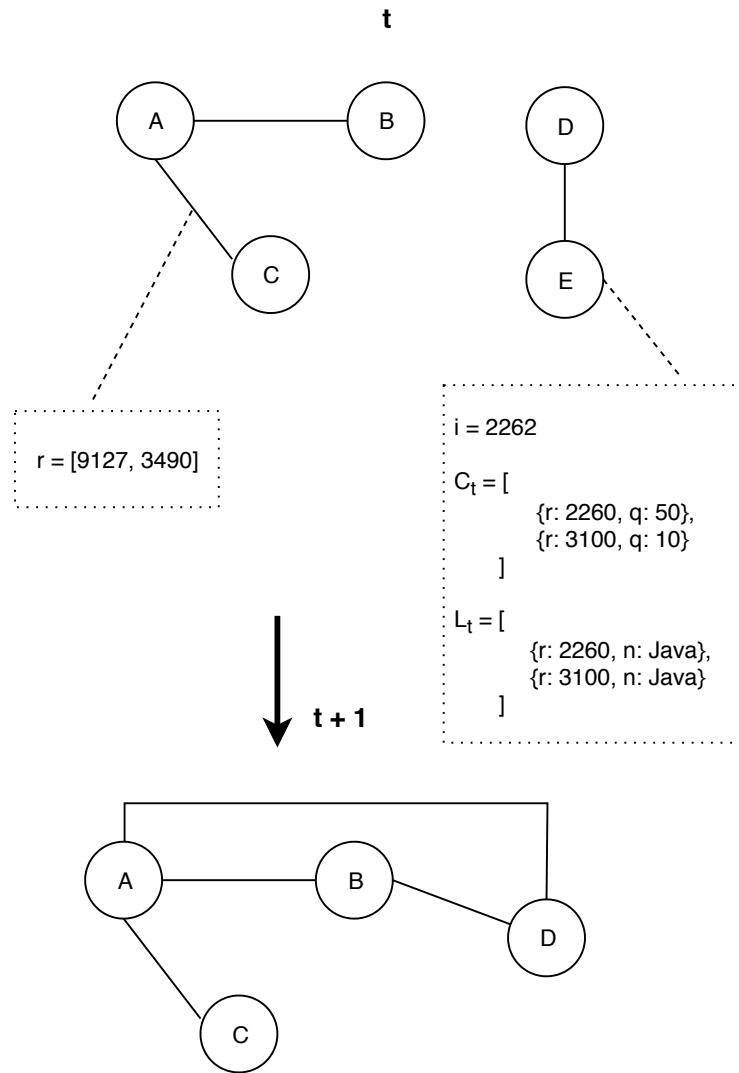


Figura 5.1: Exemplo de rede de colaboração no tempo t e tempo $t+1$.

Em redes aleatórias, a distribuição de grau segue uma distribuição de Poisson [Erdős & Rényi, 1960]. No caso de redes livres de escala, essa distribuição segue uma lei de potência [Barabasi, 2003].

Densidade A densidade é uma medida que indica o grau de conexão de um grafo com relação a um grafo completo. A densidade (d) em grafos não direcionados, como é o caso das redes modeladas neste trabalho, é definida pela equação:

$$d = \frac{2m}{n(n-1)}, \quad (5.1)$$

onde n é o número de vértices e m é o número de arestas.

Assortatividade A assortatividade é uma propriedade que descreve os padrões de conectividade dos vértices na rede. Se considerarmos como métrica de referência o grau de um vértice, uma rede assortativa é aquela em que se observa a tendência de

vértices de graus semelhantes se conectarem. Esse é o caso, quando vértices de grau alto se conectam com vértices de grau alto, e vértices de grau baixo se conectam com vértices de grau baixo. Quando se observa o fenômeno oposto, diz-se que a rede é disassortativa. A assortatividade é calculada a partir da equação

$$r = \frac{\sum_i e_{ii} - \sum_i a_i b_i}{1 - \sum_i a_i b_i} \quad (5.2)$$

onde e é uma matriz cujos elementos e_{ij} são as frações de arestas que conectam um vértice de grau i a um vértice de grau j . Se $r = 0$, diz-se que não há padrão de assortatividade, quando $r = 1$, ela é perfeitamente assortativa, e quando $r = -1$, ela é perfeitamente disassortativa.

Centralidade de *betweenness* Em teoria de grafos, uma métrica de centralidade indica a importância de um vértice na rede, focada em algum aspecto. Na centralidade de *betweenness*, o foco é na intermediação, isto é, na quantidade de caminhos geodésicos entre dois vértices que passam pelo vértice para qual se quer calcular a centralidade. Vértices com alto *betweenness* estão associados a formação de pontes na rede, e a remoção de arestas desses vértices podem levar a desconexão da rede. A centralidade de *betweenness* pode ser calculada através da relação

$$B_u = \sum_{ij} \frac{\theta(i, u, j)}{\theta(i, j)}, \quad (5.3)$$

onde $\theta(i, u, j)$ é o número de caminhos geodésicos entre os vértices i e j que passam por u , e $\theta(i, j)$ é o número total de caminhos geodésicos entre i e j .

Componentes Um componente de um grafo é um subgrafo no qual dois vértices quaisquer estão conectados por caminhos. O maior componente de um grafo é aquele que possui o maior número de vértices e é conhecido como LCC - *Largest Connected Component*.

Diâmetro O diâmetro L de uma rede é definido como sendo o comprimento, em número de arestas, do maior caminho geodésico entre dois vértices. Um caminho geodésico é o menor caminho de um vértice a outro na rede. É expresso pela relação

$$L = \max\{d_v^u\}, \quad (5.4)$$

onde $\{d_v^u\}$ é o conjunto de comprimentos de caminhos geodésicos entre quaisquer dois vértices da rede. Se um grafo possui mais de um componente, então o diâmetro do grafo é expresso pelo diâmetro do maior componente.

5.2 Caracterização das redes de desenvolvimento colaborativo de software

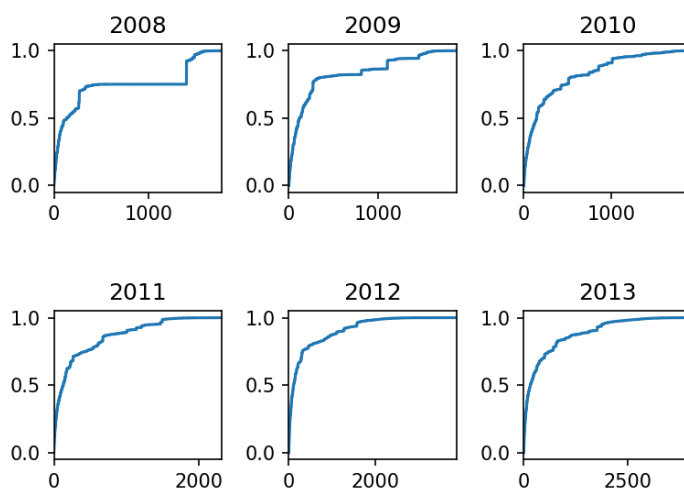
Os ecossistemas TensorFlow, Spring e SignalR foram modelados como redes, segundo o modelo apresentado na Seção 5.1. O objetivo inicial era parametrizar o modelo com $T = 10$ anos, e $\delta = 1$ ano, da mesma maneira que foi realizada a caracterização dos ecossistemas no Capítulo 4. No entanto, devido ao grande número de arestas, não foi possível gerar 10 redes para cada ecossistema. Para os ecossistemas SignalR e Spring foi possível gerar 7 redes ($T = 7$, $\delta = 1$; 2008 a 2014). Para o ecossistema TensorFlow, no entanto, apenas 6 redes foram geradas ($T = 6$, $\delta = 1$; 2008 a 2013).

A Tabela 5.1 apresenta um sumário das redes geradas, com o total de colaboradores e arestas em cada ano, para cada ecossistema. Todas as redes apresentam um número crescente de colaboradores e de arestas.

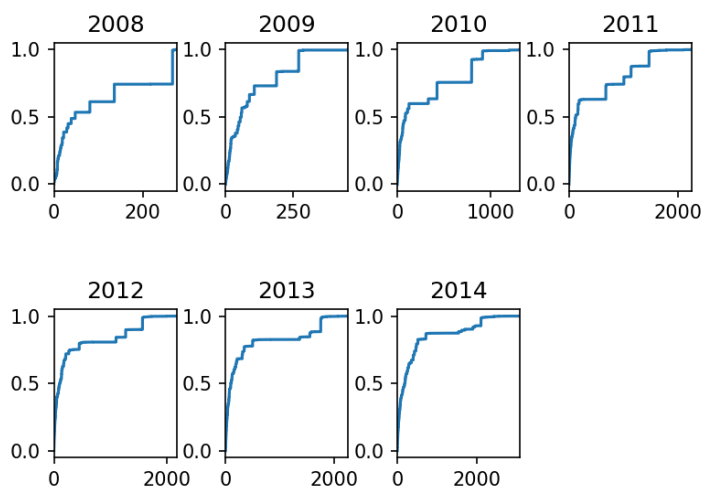
Ano	TensorFlow		Spring		SignalR	
	Colaboradores	Arestas	Colaboradores	Arestas	Colaboradores	Arestas
2008	5.619	1.236.362	1.046	53.710	170	3.670
2009	8.228	1.262.029	1.687	79.801	314	7.751
2010	12.189	1.920.582	3.275	482.034	551	9.815
2011	17.137	2.648.091	5.714	1.275.019	1.060	28.600
2012	24.947	4.249.785	8.105	1.441.129	2.087	71.783
2013	34.424	7.901.750	10.110	1.990.699	2.830	116.995
2014	-	-	16.713	3.580.652	5.553	291.107

Tabela 5.1: Sumário das redes de colaboração

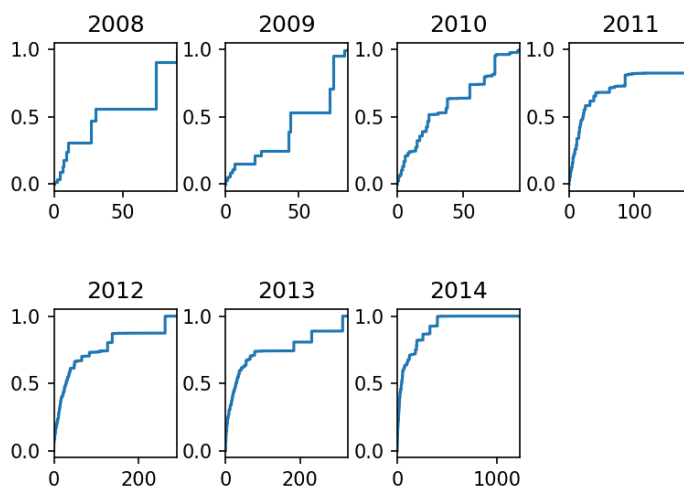
Grau Nas redes de colaboração modeladas, o grau de um vértice indica a quantidade de indivíduos únicos com os quais um determinado indivíduo colaborou durante o ano. A Tabela 5.2 exibe a média, desvio-padrão e mediana dos graus das redes para cada ano. Em todos os ecossistemas o desvio-padrão é alto, e sempre maior que a mediana. A Figura 5.2 mostra a distribuição de grau ao longo dos anos para cada ecossistema. É possível observar que ao longo dos anos a curva aproxima-se de uma exponencial, indicando a existência de uma cauda pesada, em que graus mais altos vão se tornando cada vez mais raros. Esse fenômeno ocorre mais rapidamente nas redes maiores (TensorFlow e Spring) e mais lentamente em SignalR.



(a) TensorFlow



(b) Spring



(c) SignalR

Figura 5.2: Função de Distribuição Cumulativa (CDF) dos graus nas redes de colaboração

Ano	TensorFlow			Spring			SignalR		
	Média	Desv. Padrão	Mediana	Média	Desv. Padrão	Mediana	Média	Desv. Padrão	Mediana
2008	440,06	574,47	136,00	102,69	104,85	48,00	43,18	31,67	30,00
2009	306,76	429,41	121,00	94,61	95,87	58,00	49,36	26,29	44,00
2010	315,13	392,35	146,00	294,37	339,15	87,00	35,63	27,54	24,00
2011	309,05	417,79	130,00	446,28	539,23	123,00	53,96	64,70	21,00
2012	340,70	511,33	110,00	355,61	529,72	101,00	68,79	85,01	29,00
2013	459,08	653,47	155,00	393,81	607,31	99,00	82,68	107,12	28,00
2014	-	-	-	428,49	628,14	185,00	104,84	127,02	44,00

Tabela 5.2: Grau dos vértices nas redes de colaboração

Densidade O grau é uma importante medida de diversidade da colaboração, mas a densidade ajuda a dimensionar a sua intensidade em comparação a um cenário em que todos os desenvolvedores colaborassem entre si. As densidades das diferentes redes são apresentadas na Tabela 5.3. Todas as redes apresentam uma densidade baixa, indicando que apenas uma fração dos colaboradores estão conectados entre si. As redes do ecossistema SignalR, nos dois primeiros anos, apresenta uma densidade um pouco mais elevada, mas posteriormente apresenta densidade semelhante às redes dos demais ecossistemas.

Ano	TensorFlow		Spring		SignalR	
	Densidade	Assortatividade	Densidade	Assortatividade	Densidade	Assortatividade
2008	0,0783	0,8746	0,0982	0,9948	0,2555	0,7332
2009	0,0373	0,7218	0,0561	0,8989	0,1577	0,8539
2010	0,0258	0,6079	0,0899	0,7070	0,0648	0,7886
2011	0,0180	0,6365	0,0781	0,7012	0,0509	0,9838
2012	0,0136	0,5237	0,0439	0,8403	0,0330	0,9821
2013	0,0133	0,6227	0,0389	0,9227	0,0292	0,9934
2014	-	-	0,0256	0,8926	0,0188	0,8036

Tabela 5.3: Densidade e Assortatividade das redes de colaboração

Assortatividade Uma vez que se conhece a quantidade de colaboração em média para cada indivíduo e a fração de colaboração na rede com relação a uma rede em que todos se colaboram, é interessante entender se existe um padrão para que a colaboração aconteça. Nas redes analisadas, a assortatividade é alta, como mostra a Tabela 5.3. Isso demonstra que em geral os colaboradores estão conectados a outros colaboradores de mesmo grau, ou seja, que possuem intensidade de colaboração semelhante. Isso pode ser explicado em parte pela construção da rede, uma vez que todas as pessoas de um mesmo repositório estão conectadas na rede. No entanto, essas pessoas poderiam colaborar em repositórios com diferentes números de colaboradores, e o número alto da métrica sugere que as pessoas tendem a colaborar em repositórios com número de colaboradores mais ou menos igual ao que já vinham colaborando. Nos ecossistemas Spring e SignalR, o menor valor observado foi 0,7012 (2011) e 0,7332 (2008), respectivamente. TensorFlow foi o ecossistema cujas redes

apresentaram menores valores de assortatividade: somente duas redes com valores acima de 0,7; o menor valor foi 0,5237 (2012).

Componentes A Tabela 5.4 exibe o número de componentes de cada rede. Em geral, as redes possuem muitos componentes, cujo número cresce à medida que a rede aumenta. Nas redes TensorFlow, a maior parte dos componentes é pequeno, uma vez que o maior componente reúne quase a totalidade dos colaboradores, em todos os anos. No ecossistema Spring, a porcentagem de colaboradores no maior componente é baixa nos dois primeiros anos, e aumenta em direção à totalidade a partir de 2010. Por sua vez, o ecossistema SignalR apresenta uma dispersão maior dos colaboradores em torno dos componentes. A porcentagem dos colaboradores no maior componente só ultrapassa 50% no primeiro e nos dois últimos anos observados. O resultado dessa métrica mostrou diferentes padrões topológicos dos ecossistemas. Apesar de alguns trabalhos da literatura que investigam a colaboração no GitHub também apresentarem redes com múltiplos componentes, o nosso modelo leva em conta a dinâmica do ecossistema, isto é, colaboradores que estavam desconectados/conectados em um tempo t podem aparecer conectados/desconectados em um tempo $t+1$, alterando toda a estrutura da rede.

Ano	TensorFlow		Spring		SignalR	
	# Componentes	% de nós em LCC	# Componentes	% de nós em LCC	# Componentes	% de nós em LCC
2008	118	85,17	58	26,86	12	52,94
2009	184	83,98	89	36,16	23	27,07
2010	285	87,76	113	73,04	40	35,93
2011	420	87,33	148	89,22	72	32,64
2012	656	87,22	204	86,39	193	36,61
2013	872	90,67	270	88,93	321	53,43
2014	-	-	211	93,84	336	82,39

Tabela 5.4: Distribuição dos componentes nas redes de colaboração

Diâmetro O diâmetro do LCC das redes foi bastante variável entre os ecossistemas, conforme pode-se observar na Tabela 5.5. Um diâmetro de 10, por exemplo, indica que o maior caminho entre um colaborador e outro é através de 10 arestas. Como os colaboradores estão conectados entre si devido aos repositórios em que colaboraram, nessas redes, o diâmetro é um indicador da quantidade mínima de repositórios que o maior componente possui. Essa variação observada está intimamente ligada a dinâmica da colaboração. Quando o diâmetro da rede aumenta de um ano para outro, é um sinal de que a rede conseguiu incluir colaboradores de outros repositórios, trazendo diversidade ao sistema. Quando ele diminui, há duas possibilidades: (1) os colaboradores que lá estavam começaram a colaborar em mais repositórios em conjunto; (2) colaboradores abandonaram alguns repositórios. Portanto, do ponto de vista da colaboração, essa variação é interessante para o ecossistema, porque por um lado traz diversidade, e por outro pode significar o estreitamento de laços de colaboração.

Ano	TensorFlow	Spring	SignalR
2008	10	3	2
2009	10	4	2
2010	10	14	9
2011	11	9	10
2012	10	10	7
2013	11	10	10
2014	-	9	9

Tabela 5.5: Diâmetro do *LCC* das redes de colaboração

Centralidade de *betweenness* Os valores calculados para a centralidade de *betweenness* são apresentados na Tabela 5.6. A média é extremamente baixa e com desvio padrão alto em todas as redes de todos os ecossistemas. Até o 90° percentil a métrica apresentou quase sempre valores nulos, indicando que apenas muito poucos colaboradores intermediam a colaboração. Esses colaboradores são fundamentais para a troca de experiência no desenvolvimento de software, pois são pontes entre diferentes experiências de colaboração em repositórios e, por isso, podem auxiliar na manutenção da colaboração ao longo do tempo, além de contribuírem para disseminação de conhecimento.

Ecossistema	Ano	Média	Desv. Padrão	90%	max
TensorFlow	2008	0,000246	0,002274	0,000091	0,077027
	2009	0,000174	0,001580	0,000077	0,089386
	2010	0,000138	0,001209	0,000069	0,072883
	2011	0,000099	0,000807	0,000059	0,041531
	2012	0,000068	0,000573	0,000040	0,051726
	2013	0,000057	0,000556	0,000029	0,041775
	2014	0,000106	0,000837	0,000003	0,035415
Spring	2008	0,000030	0,000647	0,000000	0,020195
	2009	0,000079	0,000858	0,000000	0,014711
	2010	0,000378	0,003831	0,000000	0,109275
	2011	0,000276	0,002219	0,000029	0,051458
	2012	0,000196	0,001555	0,000014	0,061118
	2013	0,000167	0,001237	0,000013	0,034183
	2014	0,000106	0,000837	0,000003	0,035415
SignalR	2008	0,000367	0,001141	0,000000	0,003896
	2009	0,000053	0,000242	0,000000	0,002130
	2010	0,000585	0,004555	0,000000	0,064192
	2011	0,000318	0,002574	0,000000	0,051714
	2012	0,000145	0,001465	0,000000	0,042693
	2013	0,000370	0,005278	0,000000	0,162274
	2014	0,000304	0,005003	0,000000	0,318904

Tabela 5.6: Distribuição do *betweenness* nas redes de colaboração

5.3 Discussão

A modelagem em rede proposta trouxe para os ecossistemas de colaboração uma visão histórica do comportamento colaborativo dos seus desenvolvedores. A partir dessa modelagem foi possível acompanhar os relacionamentos colaborativos que um desenvolvedor manteve durante um tempo t , e também os recursos do ambiente que ele utilizou para isso (línguas de programação e repositórios). Com esse modelo é possível observar conexões feitas e desfeitas, componentes que antes eram desconectados se conectando, e vice-versa.

Os resultados para a métrica de grau mostram que apesar de redes de tamanhos distintos, o comportamento dinâmico delas é similar. Todas elas evidenciam a existência e fortalecimento de uma cauda pesada, que denota que poucos colaboradores possuem um número de pares muito grande. Essa visão emergente do fenômeno só é possível dado a modelagem temporal, uma vez que podemos acompanhar seu surgimento e evolução. Além disso, porque essas redes focam na linearidade, e buscam entender o histórico do colaborador (sem nenhum filtro por tipo de repositório ou linguagem), esses resultados capturam mais fielmente como as conexões são estruturadas no processo colaborativo.

A existência de padrões de conexão também foi verificada nos ecossistemas estudados em rede. Os colaboradores demonstraram e mantiveram a característica de colaborar em repositórios de tamanho similar ao que já vinham colaborando. Esse resultado é intrigante e requer maiores investigações para melhor compreensão do fenômeno.

As redes pouco densas podem indicar que a colaboração é um processo custoso, pois envolve estabelecer relações com os pares e mantê-las ao longo do tempo. Os custos são pessoais, e muito provavelmente variáveis entre os colaboradores, o que justificaria a existência de colaboradores mais e menos colaborativos. O diâmetro do maior componente é revelador do alcance, da diversidade e do estreitamento de laços provocado pela colaboração. Nesse sentido, a métrica *betweenness* foi essencial para prover uma visão da importância que poucos colaboradores exercem sobre a rede de colaboração, uma vez que estabelecem pontes entre colaboradores, e provocam a troca de conhecimento entre repositórios.

Capítulo 6

Caracterização das redes de colaboração inspirada em modelos biológicos

Este capítulo apresenta a principal contribuição desta dissertação, que é a caracterização da colaboração que ocorre em diferentes ecossistemas extraídos do GitHub a partir de medidas de colaboração inspiradas nos conceitos utilizados pelos modelos biológicos apresentados no Capítulo 2. A Seção 6.1 apresenta a metodologia seguida para a caracterização das redes TensorFlow, Spring e SignalR, descrevendo métricas propostas nesta dissertação. Os resultados das análises são apresentados na Seção 6.2 e discutidos na Seção 6.3.

6.1 Metodologia

Nesta seção são apresentados os métodos utilizados para caracterizar as redes de colaboração sob a inspiração das seguintes classes de modelos biológicos: modelos de reciprocidade direcionada, modelos de benefícios por produtos e modelos de genes compartilhados. Em nossas análises, a evolução temporal das redes de colaboração é fundamental para a definição de novas métricas que visam compreender o processo de colaboração que ocorre nos ecossistemas TensorFlow, Spring e SignalR tendo como base os conceitos como *partner choice* e *kin choice*.

6.1.1 Modelos de Reciprocidade Direcionada

Para avaliar as redes de colaboração de acordo com os modelos de reciprocidade direcionada, propomos dois tipos diferentes de análises. Na primeira, o principal foco é analisar a persistência da colaboração. Essa análise tem como objetivo verificar a existência (ou não) da reciprocidade entre os pares de desenvolvedores de cada uma das redes de colaboração. Na segunda análise, avaliamos se existe algum tipo de estratégia adotada pelos colaboradores para a promoção da colaboração nessas redes.

Análise da persistência da colaboração No GitHub, podemos considerar a persistência na qual um desenvolvedor segue colaborando na rede como reciprocidade no processo de colaboração. Assumir esta equivalência entre reciprocidade e persistência se fundamenta no fato de que o processo de colaboração é iterativo, isto é, o desenvolvedor ao perceber que possivelmente a sua participação trouxe benefícios ao repositório e aos demais colaboradores (por exemplo, com a difusão de conhecimento), a reciprocidade existirá enquanto a colaboração persistir. Uma outra razão para examinar a reciprocidade como sendo persistência deve-se ao fato de a colaboração ocorrer em rede, e não simplesmente par-a-par como descreve simplificada os modelos biológicos. Portanto, realizou-se uma análise da persistência da colaboração dentro das redes ao longo do tempo.

Baseando-se no conceito de reciprocidade, podemos considerar três visões distintas de persistência da colaboração: (1) percentual de colaboradores que continuam colaborando na rede, independente do repositório ou parceiro; (2) percentual de colaboradores que continuam colaborando nos mesmos repositórios do ano anterior; (3) percentual de colaboradores que continuam colaborando com os mesmos parceiros. Dentre as três visões, a primeira é a que mais se adequa ao objetivo desta dissertação, uma vez que considera a persistência em rede. A segunda visão pode ser útil para entender se há persistência de colaboração também no nível de repositório, e os motivos de um desenvolvedor deixar de colaborar em um repositório para colaborar em outro, quando essa persistência falha. A terceira visão é relevante quando se estuda a colaboração par-a-par. Essa visão, no entanto, restringe muito os efeitos emergentes da colaboração, que é de ordem global. Além disso, como o modelo em rede proposto considera que uma aresta existe caso os colaboradores trabalhem em um mesmo repositório, para analisar a descontinuação da persistência entre dois pares é necessário investigar qual dos pares deixou de colaborar. Por exemplo, se um desenvolvedor A colabora com um desenvolvedor B no repositório 1 em um tempo t , e essa aresta desaparece em um tempo $t+1$, é preciso investigar quem foi o “traidor”, ou seja, quem é que deixou de colaborar no repositório.

Por fim, a persistência dos colaboradores ao longo dos anos pode ser analisada sob duas visões, uma de anterioridade e outra de posterioridade. Na visão de anterioridade, queremos saber a porcentagem de colaboradores que participam da rede de colaboração no ano i que também colaboraram previamente no ano j ($i > j$). Na visão de posterioridade, o objetivo é conhecer o percentual de colaboradores no ano i que permanece colaborando no ano j , isto é, ($i < j$). Ambas as visões podem ser calculadas a partir da relação

$$\Psi = \frac{D_i \cap D_j}{|D_i|}, \quad (6.1)$$

onde D_i é o conjunto de desenvolvedores no ano i , e D_j é o conjunto de desenvolvedores no ano j , e $|D_i|$ é a cardinalidade do conjunto D_i .

A visão de anterioridade é relevante porque permite saber o percentual de colaboradores que é novo no ecossistema no ano observado ($1 - \Psi$). Por sua vez, a visão de posterioridade traz a ideia de reciprocidade e de fidelidade.

Análise de estratégia da colaboração Ao analisar a colaboração entre desenvolvedores do ponto de vista do modelo de reciprocidade direcionada, é necessário definir o modelo que descreve a dinâmica de retribuição de colaboração entre os pares de desenvolvedores. Consideramos que a dinâmica de colaboração nas redes TensorFlow, Spring e SignalR são bem descritas pelo modelo *Partner Choice*. De todas as estratégias apresentadas na literatura, a que melhor parece se adequar ao GitHub é a *image scoring*, que consiste em avaliar a reputação de um parceiro antes de colaborar com ele (no GitHub, desenvolvedores estão sendo constantemente avaliados pelos *pull-requests*¹ que realizam). Assim, definimos uma métrica que utiliza as informações extraídas dos ecossistemas para reproduzir o mecanismo de estratégia *image scoring*. O objetivo é avaliar se repositórios com colaboradores de alta reputação tem capacidade de atrair mais colaboradores do que aqueles que não os tem. Caso isso de fato ocorra nestas redes, temos indícios que uma das estratégias utilizadas para a colaboração realmente segue os princípios do sistema de decisão *image scoring*.

Para calcular a reputação de um colaborador no GitHub, propomos uma métrica que é uma função do número de commits realizado pelo desenvolvedor. O número de commits foi considerado antes de outros possíveis métodos (como o número de seguidores), porque nele resume a colaboração, e como destacado anteriormente, cada commit é avaliado por outros desenvolvedores através do mecanismo de *pull-request*. A avaliação prévia de um commit pode significar, de certa forma, uma chancela da qualidade da colaboração

¹Um *pull-request* é um pedido de fusão das suas colaborações através de um ou mais commits, para o repositório original.

por parte dos demais desenvolvedores. Desse modo, a reputação R de um colaborador c durante o ano t é dada pela equação

$$R_t^c = \frac{\sum_r \gamma_t^{(c,r)}}{\sum_r \gamma_t^r}, \quad (6.2)$$

onde r é o repositório, $\gamma_t^{(c,r)}$ é a quantidade de commits realizada pelo colaborador c no repositório r durante o ano t , e γ_t^r é a quantidade de commits feita por todos os colaboradores no repositório r durante o ano t . Se o repositório r possui apenas um colaborador, tanto $\gamma_t^{(c,r)}$ quanto γ_t^r são iguais a 0. Essa consideração foi feita para evitar que desenvolvedores ganhassem reputação apenas por serem o único colaborador do repositório, dado que neste modelo a reputação de um colaborador é devido à sua colaboração direta com terceiros.

Para avaliar a estratégia *image scoring*, calculamos a reputação de cada colaborador da rede. Em seguida, para definirmos um limiar com o objetivo de dividirmos os colaboradores entre grupos diferentes segundo o nível (ou grandeza) da reputação, calculamos a distribuição de probabilidade cumulativa das reputações individuais. Assim, definimos o grupo **R1**, com os colaboradores cujos valores de reputação estão acima do limiar, e o grupo **R2**, com os demais colaboradores.

Segundo nossa hipótese, os colaboradores de **R1** são mais promotores da colaboração do que os colaboradores de **R2**. Para validá-la, poderíamos calcular a fração do total de novos colaboradores em **R1** com relação aos novos colaboradores em **R2**. Um valor maior do que 1 mostra que o grupo que tem maior reputação (**R1**) conseguiu atrair mais desenvolvedores. No entanto, como os colaboradores em **R1** participam na maior parte dos repositórios, a força de atração dos novos colaboradores poderia ser influenciada não somente pelo *image scoring*, mas pelo fato de estarem presentes em uma diversidade maior de repositórios (aumentando a chance de atração). Para isolar essa possível causa, vamos considerar a atração de cada grupo ponderada pelo número de repositórios dos quais os desenvolvedores participam, isto é

$$\mathcal{A}_{Ri} = \frac{\delta_i}{\rho_i} \quad (6.3)$$

onde \mathcal{A}_{Ri} é a quantidade de novos colaboradores por repositório no grupo Ri , isto é, sua atratividade, δ_i é o número de novos colaboradores no grupo Ri , e ρ_i é a quantidade de repositórios nos quais os desenvolvedores pertencentes ao grupo Ri participam.

Verificamos se a estratégia de *image scoring* se aplica nessas redes através do que vamos chamar de *força de atração do grupo R1* (Φ), que é a razão entre \mathcal{A}_{R1} e \mathcal{A}_{R2} . Assim, temos que

$$\Phi = \frac{\mathcal{A}_{R1}}{\mathcal{A}_{R2}}, \quad (6.4)$$

um valor Φ maior do que 1 indica que a estratégia *image scoring* é utilizada no ecossistema.

Dado que a estratégia *image scoring* depende do histórico de reputação dos colaboradores, é importante verificarmos a persistência de colaboradores pertencentes ao grupo **R1** ao longo dos anos. Na análise de persistência dos colaboradores no grupo **R1**, aplicamos mesma metodologia que utilizamos para analisar a persistência da colaboração dos desenvolvedores na rede, independentemente das suas reputações, com os conceitos de anterioridade e posterioridade.

6.1.2 Modelos de Benefícios por Produtos

A classe de modelos de benefícios por produtos reúne as colaborações em que um ato egoísta resulta em um benefício. Esse tipo de comportamento poderia ocorrer no GitHub, se uma pessoa decidisse colaborar em um repositório para adicionar uma funcionalidade que ela necessita, por exemplo. A colaboração atenderia a um desejo de beneficiar a si próprio (adicionar a funcionalidade porque ela necessita), porém a colaboração afetaria a todos os potenciais colaboradores desse repositório. Uma das maneiras de observar a emergência de benefícios por produtos nas redes de colaboração é investigando a existência de sinergismo. Quando há sinergismo, os benefícios de se colaborar em numerosos grupos são desproporcionalmente maiores com relação aos benefícios de quando se colabora em grupos pequenos.

Podemos citar inúmeros tipos de benefícios que um desenvolvedor recebe ao colaborar em grupos numerosos no GitHub: o desenvolvedor pode ficar conhecido e influente; o desenvolvedor programa o software mais rapidamente; a chance de seu projeto avançar é maior ao disponibilizá-lo em repositórios mais populares². Muitos desses benefícios são refletidos pelo número de commits que o repositório tem.

Desse modo, a fim de validar se há sinergismo nas redes de colaboração do GitHub, definimos como benefício o número de commits, pois ele resume a intensidade da colaboração, e é a maneira pela qual ocorre a troca de conhecimento na rede. Portanto, queremos encontrar indícios de sinergismo no GitHub verificando se o número de commits realizados é desproporcionalmente maior à medida que o repositório recebe mais colaboradores.

Para isso, calculamos a distribuição do número de colaboradores por repositório, com o objetivo de encontrar valores-limite que podem ser usados para classificar os repositórios. A classificação em diferentes grupos considerou toda a rede do ecossistema (grafo

²Na biologia, peixes preferem se manter em cardumes do que separados, porque sua taxa de predação média é desproporcionalmente menor. Assim, para os peixes, o sinergismo confere como benefício uma taxa de predação menor.

estático), desprezando as colaborações estabelecidas através de *forks*. Ao considerarmos o grafo estático, buscamos um parâmetro para definir o que é um repositório grande diante de toda a história do ecossistema. Por fim, ao filtrarmos os *forks*, focamos a colaboração efetivada nos repositórios originais, essencial para a nossa análise.

Os repositórios foram classificados em 5 grupos:

- G1: do valor mínimo de colaboradores por repositório até o primeiro quartil (25%);
- G2: acima do primeiro quartil até o segundo quartil (50%);
- G3: acima do segundo quartil até o terceiro quartil (75%);
- G4: acima do terceiro quartil até o 90° percentil;
- G5: acima do 90° percentil até o valor máximo de colaboradores por repositório.

Uma vez tendo classificado esses repositórios, é preciso comparar os benefícios promovidos pelos grupos. Para isso, selecionamos como parâmetro de comparação a mediana das distribuições de commits nos repositórios de cada grupo. Por exemplo, um valor de mediana muito maior em um grupo G5, se comparado a um grupo G1, indica que a maioria dos repositórios em G5 (que tem mais colaboradores) possui uma quantidade maior de commits que em G1 (que tem menos colaboradores). Assim, comparando os benefícios, conjecturamos que se um desenvolvedor decidir colaborar em um repositório de grupos que possuem um maior número de desenvolvedores é porque ele está sendo influenciado pelo princípio de sinergismo.

6.1.3 Modelos de Genes Compartilhados

Os modelos de genes compartilhados constituem uma classe bem específica de colaboração, pois o reconhecimento do genoma está envolvido no processo da tomada de decisão em estabelecer ou não uma colaboração. No GitHub, como toda colaboração é um processo à distância, é improvável considerar o reconhecimento genético de parceiros (em seu sentido estritamente biológico) como promotor da colaboração.

No entanto, um dos modelos dessa classe (*kin choice*) diz respeito à colaboração quando se reconhece características fenotípicas em outros colaboradores. Nesse caso, dado que a colaboração no GitHub é um processo sem contato físico, uma maneira de aplicar esse modelo é assumir que o conhecimento de uma linguagem de programação reflete o processo de identificar uma característica fenotípica semelhante.

A nossa metodologia para aplicação dos modelos de genes compartilhados para entender a colaboração no GitHub se baseia nas conclusões apresentadas em Valverde & Solé [2015]. Os autores estudaram a evolução das linguagens de programação e mostraram que essa evolução ocorreu de forma “quase biológica, marcada pelos avanços sociais e tecnológicos dos últimos 60 anos”. Nesse trabalho, eles reconstruíram as redes filogenéticas das linguagens de programação. Isso foi possível graças aos rastros deixados por cada uma das linguagens que influenciaram umas às outras. A Figura 6.1 mostra uma parte da árvore filogenética construída.

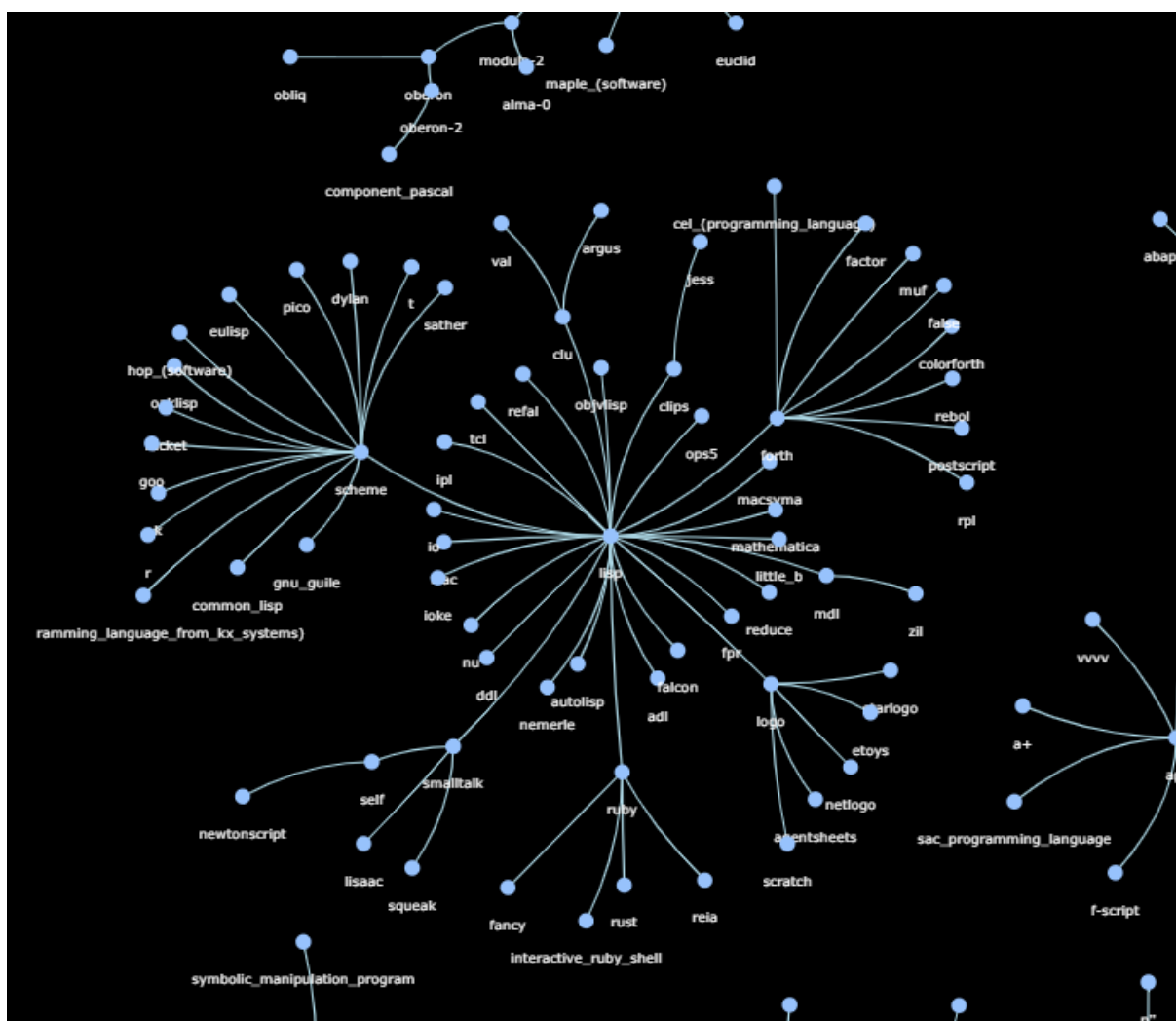


Figura 6.1: Visão parcial da árvore filogenética das linguagens de programação

Essa árvore está modelada como um grafo com 347 vértices (linguagens de programação) e 327 arestas (influência direta de uma linguagem sobre a outra). O grafo possui 20 componentes, indicando 20 “troncos linguísticos” que não possuem relação entre si. O maior componente dessa árvore possui um diâmetro de 13, o que significa que há no máximo 13 arestas (saltos) de distância entre uma linguagem e outra, quando uma recebe influência de outra.

Para validar se o processo de colaboração envolve também aspectos descritos pelo modelo de genes compartilhados, investigamos o quão similares são as linguagens utilizadas por um determinado desenvolvedor considerando as colaborações estabelecidas pelo mesmo. Assim, se em média, os colaboradores tendem a se engajar em colaborações usando linguagens similares, podemos dizer que existem indícios de que a colaboração no GitHub siga conceitos dos modelos de genes compartilhados.

Definimos que a similaridade entre duas linguagens ($\mathcal{S}_{l_2}^{l_1}$) é dada pela equação

$$\mathcal{S}_{l_2}^{l_1} = 1 - \frac{d_{l_2}^{l_1}}{14} \quad (6.5)$$

onde $d_{l_2}^{l_1}$ é o tamanho do caminho geodésico entre as duas linguagens.

Quando as linguagens são iguais ($l_1 = l_2$), considera-se que a distância $d_{l_2}^{l_1} = 0$, e portanto, $\mathcal{S}_{l_2}^{l_1} = 1$. Quando não fazem parte do mesmo tronco linguístico, e não recebem influência uma da outra, convencionou-se que a distância $d_{l_2}^{l_1} = 14$, uma vez que o diâmetro do maior componente conectado é 13. De fato, essa distância é infinita. Assim, nessa escala, duas linguagens são mais similares quanto mais próximas de 1 e mais dissimilares quanto mais próximas de 0.

Por exemplo, nessa rede filogenética de linguagens, a distância entre a linguagem C e C# é de 3 saltos ($d_{C\#}^C = 3$), pois o menor caminho de influência é dado pelo conjunto $[C \rightarrow C++ \rightarrow Java \rightarrow C\#]$, com três arestas. Assim, a similaridade entre C e C# é de 0,78 na escala estabelecida.

Uma vez definido o grau de similaridade entre duas linguagens, o próximo passo é definirmos o quanto um desenvolvedor é fiel a estabelecer colaborações em repositórios com linguagens similares. Definimos por \mathcal{C}_L^D o grau de conformidade das linguagens que um desenvolvedor D utiliza nos commits realizados nas redes consideradas. Assim, temos:

$$\mathcal{C}_L^D = \frac{\sum_{S_{l_2}^{l_1}} \mathcal{S}_{l_2}^{l_1} (\gamma_{l_1} + \gamma_{l_2})}{\sum_{S_{l_2}^{l_1}} \gamma_{l_1} + \gamma_{l_2}} \quad (6.6)$$

onde L é o conjunto de linguagens em que o desenvolvedor D colaborou, γ_{l_1} é o número de commits efetuados na linguagem 1, e γ_{l_2} é o número de commits efetuados na linguagem 2, com $\mathcal{C}_L^D \in [0, 1]$.

Se um desenvolvedor realizou commits apenas em uma linguagem, seu $\mathcal{C}_L^D = 1$. Se essa colaboração ocorreu apenas entre duas linguagens e que são de “troncos linguísticos” diferentes, seu $\mathcal{C}_L^D = 0$. A Tabela 6.1 apresenta uma sugestão de interpretação para os valores da métrica \mathcal{C}_L^D .

Para ilustrar um caso de uso da métrica proposta, suponha que um desenvolvedor tenha colaborado em repositórios das linguagens: Java com 100 commits, C# com 200 commits e C com 50 commits. Através da árvore filogenética, sabe-se que $d_{C\#}^{Java} = 1$, $d_C^{Java} = 1$ e $d_{C\#}^C = 2$. Portanto, $\mathcal{S}_{C\#}^{Java} = 0,9286$, $\mathcal{S}_C^{Java} = 0,9286$ e $\mathcal{S}_{C\#}^C = 0,8571$. Desse

\mathcal{C}_L^D	Interpretação
0,00	perfeitamente inconforme
$0 < \mathcal{C}_L^D \leq 0,28$	conformidade muito fraca
$0,28 < \mathcal{C}_L^D \leq 0,42$	conformidade fraca
$0,42 < \mathcal{C}_L^D \leq 0,71$	conformidade moderada
$0,71 < \mathcal{C}_L^D \leq 0,85$	conforme
$0,85 < \mathcal{C}_L^D < 1,00$	muito conforme
1,00	totalmente conforme

Tabela 6.1: Sugestão de interpretação dos valores obtidos com a métrica \mathcal{C}_L^D

modo, substituindo na equação 6.1.3 tem-se que:

$$\mathcal{C}_L^D = \frac{[S_{C\#}^{Java}(\gamma_{Java} + \gamma_{C\#})] + [S_C^{Java}(\gamma_{Java} + \gamma_C)] + [S_{C\#}^C(\gamma_C + \gamma_{C\#})]}{[\gamma_{Java} + \gamma_{C\#}] + [\gamma_{Java} + \gamma_C] + [\gamma_C + \gamma_{C\#}]} \quad (6.7)$$

$$\mathcal{C}_L^D = \frac{[0,9286(100 + 200)] + [0,9286(100 + 50)] + [0,8571(50 + 200)]}{(100 + 200) + (100 + 50) + (50 + 200)} \quad (6.8)$$

$$\mathcal{C}_L^D = \frac{278,58 + 139,29 + 214,27}{300 + 150 + 250} \quad (6.9)$$

Por fim, obtém-se:

$$\mathcal{C}_U^L = \frac{632,14}{700} = 0,9030 \quad (6.10)$$

Um resultado entre 0,85 e 1,00 indica que a conformidade do desenvolvedor com relação às linguagens de programação é “muito conforme”, ou seja, que o desenvolvedor buscou colaborar em repositórios de linguagens muito similares.

6.2 Análise Experimental

Nesta seção apresentamos os resultados das análises das redes dos ecossistemas TensorFlow, Spring e SignalR através das métricas definidas à luz dos conceitos dos principais modelos biológicos utilizados nesta dissertação. A Seção 6.2.1 apresenta os resultados obtidos nos experimentos com modelos de reciprocidade direcionada. Os resultados das análises com modelos de benefícios por produtos são apresentados na Seção 6.2.2. A Seção 6.2.3 apresenta os resultados das análises com os modelos de genes compartilhados.

6.2.1 Modelos de Reciprocidade Direcionada

A seguir são apresentados os resultados obtidos nas análises da persistência e de estratégia da colaboração, que são os conceitos chaves considerados nesta dissertação como ferramenta para explicar a colaboração no GitHub sob a ótica do modelo de reciprocidade direcionada.

Análise da persistência da colaboração

Realizamos a análise de persistência dos colaboradores ao longo dos anos no mesmo ecossistema, segundo as visões de anterioridade e posterioridade, conforme apresentado na Seção 6.1. A Figura 6.2 apresenta a matriz da persistência de colaboradores com essas duas visões. A linha indica o ano de referência, e a coluna o ano de comparação. A leitura da matriz é feita desta forma: **(1) posterioridade (elementos acima da diagonal principal)**: o valor (i, j) indica a porcentagem de colaboradores no ano i que permanece colaborando no ano j . **(2) anterioridade (elementos abaixo da diagonal principal)**: o valor (i, j) equivale ao percentual de colaboradores do ano i que também colaborou no ano j .

Todas as redes dos três ecossistemas analisados apresentaram o mesmo comportamento descrito a seguir. O percentual de colaboradores que se mantiveram colaborando no ano consecutivo ao ano de referência foi sempre o maior, e decaiu quando consideramos intervalos maiores que 1 ano. Nas redes TensorFlow, entre 44% a 50% persistiram colaborando no ano consecutivo; nas redes Spring, essa taxa variou entre 33% e 47%; por fim nas redes SignalR, persistiram colaborando entre 29% a 54% dos desenvolvedores. Esses resultados mostram que esses sistemas são muito dinâmicos, mas que ainda há uma porcentagem considerável que se mantém colaborando.

A variação nos demais anos analisados é menor, sugerindo a existência de usuários fiéis que garantem a estabilidade da colaboração. Nos últimos anos analisados para esses ecossistemas, a taxa de abandono dos usuários variou apenas 2% no ecossistema TensorFlow (2012-2013), enquanto que ela variou 1% no ecossistema Spring (2013-2014) e se manteve estável no ecossistema SignalR (2013-2014).

Em geral, essas redes apresentam taxas de persistência relativamente significativas: no ecossistema TensorFlow, 24% dos colaboradores de 2008 aparecem colaborando em 2013. Nos ecossistemas Spring e SignalR, a porcentagem de colaboradores de 2008 que aparece colaborando em 2014 é de 17% e de 21%, respectivamente. Esses resultados



Figura 6.2: Matriz da persistência de colaboradores ao longo dos anos

mostram que esses ecossistemas são muito dinâmicos, com altas taxas de abandono, principalmente quando consideramos intervalos de anos distantes (6 anos em TensorFlow, e 7 anos em Spring e SignalR).

Os elementos inferiores à diagonal principal mostram a porcentagem de desenvolvedores do ano de referência que vieram de anos anteriores. No ecossistema Tensorflow, de 31% a 34% dos colaboradores de um ano vêm do seu ano anterior. Essa taxa variou de 22% a 29% nas redes Spring; e de 17% a 29% nas SignalR. Isso reforça o fato de que existem desenvolvedores fieis, ainda que o comportamento das redes seja muito dinâmico. Além disso, esse resultado mostra que esses ecossistemas têm alta capacidade para atrair novos colaboradores, o que traz muitos benefícios para esses sistemas: além de trazer novas ideias para os repositórios em que eles colaboram, promovem novos conhecimentos em torno do desenvolvimento de software.

Análise de estratégia da colaboração

Com o objetivo de validarmos a hipótese de que os colaboradores com maior reputação promovem (e atraem) mais colaboradores para o desenvolvimento em seus repositórios, avaliamos a reputação de cada um dos colaboradores das redes, conforme a métrica definida na Seção 6.1, e que reflete a estratégia de colaboração *image scoring*. A Tabela 6.2 mostra os valores mínimo, máximo, mediana e o 90° percentil da distribuição da reputação. O valor mínimo da reputação em todos os anos é zero, e reflete os casos em que repositórios são sustentados por um único desenvolvedor. No entanto, uma vez que estamos analisando a dinâmica da colaboração ao longo dos anos, este mesmo desenvolvedor pode ter reputação diferente de zero, em um ano posterior ou anterior, ou seja, se em algum momento da sua história o mesmo participou em repositórios com mais colaboradores. O valor máximo da reputação se aproximou de 1, que é quando as contribuições dos desenvolvedores são as mais relevantes para seus repositórios. Os valores de reputação não estão igualmente distribuídas entre os colaboradores, ou seja, poucos desenvolvedores possuem reputações elevadas.

Ecossistema	Ano	min	max	50%	90%
TensorFlow	2008	0	0,986111	0,001818	0,060290
	2009	0	0,993464	0,001993	0,055556
	2010	0	0,989071	0,001845	0,065114
	2011	0	0,995074	0,002760	0,074362
	2012	0	0,996078	0,002809	0,082460
	2013	0	0,998054	0,001678	0,065270
Spring	2008	0	0,979592	0,003817	0,127868
	2009	0	0,993730	0,003378	0,117647
	2010	0	0,996212	0,001858	0,079187
	2011	0	0,996169	0,001362	0,082017
	2012	0	0,987730	0,001957	0,093391
	2013	0	0,996575	0,001436	0,080917
	2014	0	0,996835	0,000973	0,043752
SignalR	2008	0	0,802632	0,007486	0,119079
	2009	0	0,940476	0,002547	0,116414
	2010	0	0,993197	0,011236	0,225989
	2011	0	0,983871	0,008534	0,209112
	2012	0	0,991304	0,005319	0,176470
	2013	0	0,994220	0,002876	0,250490
	2014	0	0,995671	0,003210	0,200000

Tabela 6.2: Valores máximos, mínimos, 50° (mediana) e 90° percentil da distribuição da reputação

Para caracterizar e quantificar a dispersão da distribuição da reputação, calculamos o coeficiente de *skewness* e a métrica curtose, como mostra a Tabela 6.3. Todos os valores de curtose e *skewness* foram positivos em todas as redes analisadas. Um valor de curtose maior que 0 indica que a função de distribuição possui a curva de distribuição mais afunilada e com pico mais alto do que a distribuição normal. Um valor de *skewness* maior que 0 indica que a função de distribuição é assimétrica com cauda maior à direita.

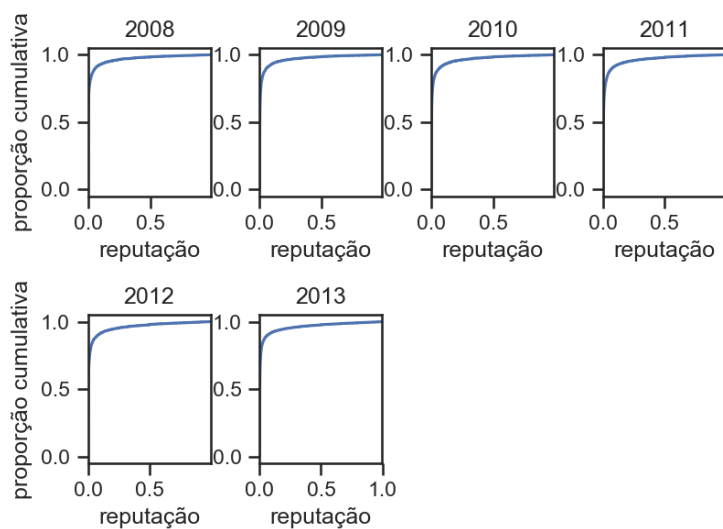
Ano	<i>TensorFlow</i>		<i>Spring</i>		<i>SignalR</i>	
	<i>Skewness</i>	Curtose	<i>Skewness</i>	Curtose	<i>Skewness</i>	Curtose
2008	5,48	33,73	4,33	21,36	3,80	14,78
2009	5,64	36,33	4,41	21,77	4,28	20,94
2010	5,34	32,20	4,96	27,33	3,39	11,90
2011	4,96	27,42	4,97	27,77	3,47	13,26
2012	4,80	25,26	4,53	22,21	3,65	13,91
2013	4,91	25,95	4,79	24,82	3,14	9,33
2014	-	-	5,74	35,92	3,37	11,08

Tabela 6.3: *Skewness* e Curtose da distribuição de reputação

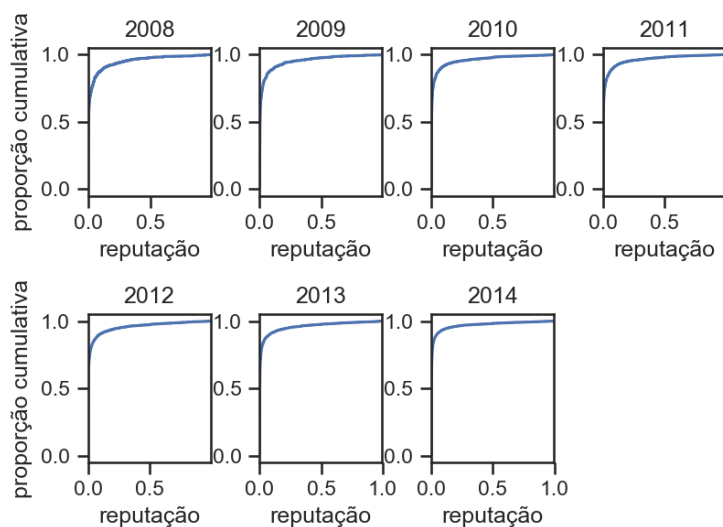
A Figura 6.3 apresenta as curvas da função de distribuição acumulada (CDF) das reputações dos desenvolvedores. Essas curvas reforçam a característica dispersa dessas distribuições, evidenciando que valores mais altos de reputação ocorrem em poucos casos, ou seja, existem colaboradores que podem ser determinantes para atração de novos colaboradores, e possivelmente esta atração pode ser explicada pela imagem positiva que esses desenvolvedores com maior reputação possuem dentro da rede. Apenas em torno dos últimos 10% da distribuição acumulada é que os valores aumentam consideravelmente. Por essa razão, adotou-se como valor-limite de reputação o 90° percentil, para a divisão dos grupos **R1** e **R2**. Esses valores encontram-se na Tabela 6.2.

A Tabela 6.4 exibe o número de colaboradores no grupo **R1** em cada rede de cada ecossistema, e a porcentagem de repositórios em que os colaboradores desse grupo fizeram commits, com relação ao total de repositórios da rede. É possível perceber que, em todas as redes, os desenvolvedores do grupo **R1** são altamente engajados e colaboram em mais da metade dos repositórios.

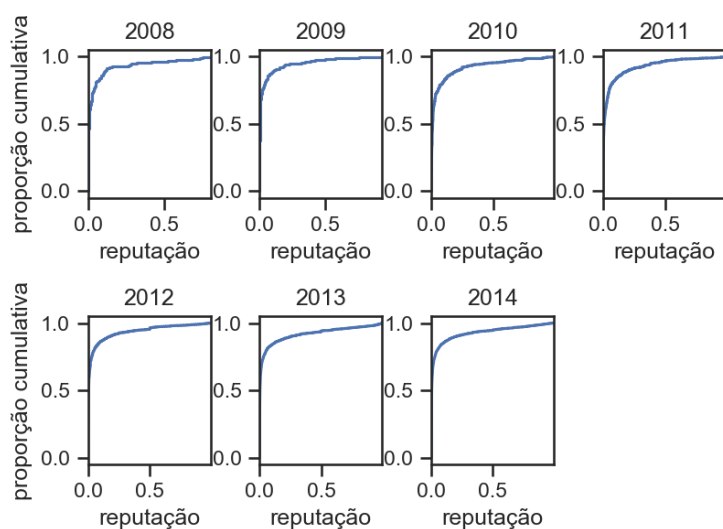
A persistência dos colaboradores no grupo **R1** também foi verificada ao longo dos anos, conforme mostra a Figura 6.4. Essa matriz traz duas visões para a persistência de colaboradores no grupo **R1**: **posterioridade (elementos acima da diagonal principal)**: o valor (i, j) indica a porcentagem de colaboradores que estavam no grupo **R1** no ano i , e que continua no grupo **R1** no ano j . **anterioridade (elementos abaixo da diagonal principal)**: o valor (i, j) equivale ao percentual de colaboradores que está no grupo **R1** no ano i , e que também estava no grupo **R1** no ano j .



(a) TensorFlow



(b) Spring



(c) SignalR

Figura 6.3: Função de Distribuição Cumulativa (CDF) das reputações dos usuários nas redes de colaboração

Ecosistema	Ano	# colaboradores	% repositórios
TensorFlow	2008	562	83,21
	2009	819	71,05
	2010	1219	62,69
	2011	1714	51,25
	2012	2495	52,08
	2013	3443	59,65
Spring	2008	105	73,56
	2009	164	68,87
	2010	328	73,12
	2011	572	76,21
	2012	811	81,28
	2013	1011	77,90
	2014	1672	77,87
SignalR	2008	17	66,67
	2009	32	51,61
	2010	55	62,50
	2011	106	65,55
	2012	207	55,71
	2013	283	50,78
	2014	548	47,67

Tabela 6.4: Colaboradores do grupo **R1** e porcentagem de repositórios em que colaboraram

Analisando-se a porção superior à diagonal principal, podemos ver que uma quantidade significativa de desenvolvedores se mantém no grupo **R1** ao longo dos anos. No ecossistema TensorFlow, 17% dos colaboradores com alta reputação em 2008 se mantém no grupo **R1** em 2013. No ecossistema Spring, isso ocorre com 23% dos colaboradores, enquanto que no ecossistema SignalR, 12% dos colaboradores que estavam em **R1** em 2008 se mantém em **R1** no final de 2014.

Spring é o ecossistema cujo grupo **R1** foi mais “estável”. A taxa de decaimento de um colaborador de **R1** é menor em Spring ao longo dos anos, do que nos demais. No entanto, para todos os ecossistemas, a taxa de decaimento diminuiu, sugerindo a busca por uma estabilização. No ecossistema TensorFlow, 51% dos colaboradores de 2008 deixaram **R1** em 2009, porém houve uma variação de 1% desses colaboradores entre 2012 e 2013. A despeito do que ocorreu nos outros anos, essa variação foi positiva, indicando a volta desses colaboradores ao **R1**. O mesmo fenômeno ocorreu no ecossistema Spring (variação de 1% positiva) e SignalR (variação de 6% positiva) entre 2013 e 2014.

A porção inferior à diagonal principal revela a diversidade e dinâmica de **R1**. No ecossistema TensorFlow, de 23% a 33% dos colaboradores que estão em **R1** também estavam nele no ano anterior. Essa taxa variou entre 25% e 39% no ecossistema Spring, e

entre 13% e 31% em SignalR. Isso mostra que **R1** recebe colaboradores diferentes ao longo dos anos, o que também é um fenômeno muito positivo para as redes de desenvolvimento colaborativo de software, uma vez que usuários muito colaborativos também são aqueles que detêm grande parte do conhecimento em torno do projeto. Desse modo, quando a diversidade em **R1** é alta, a troca de conhecimento também pode ser alta.

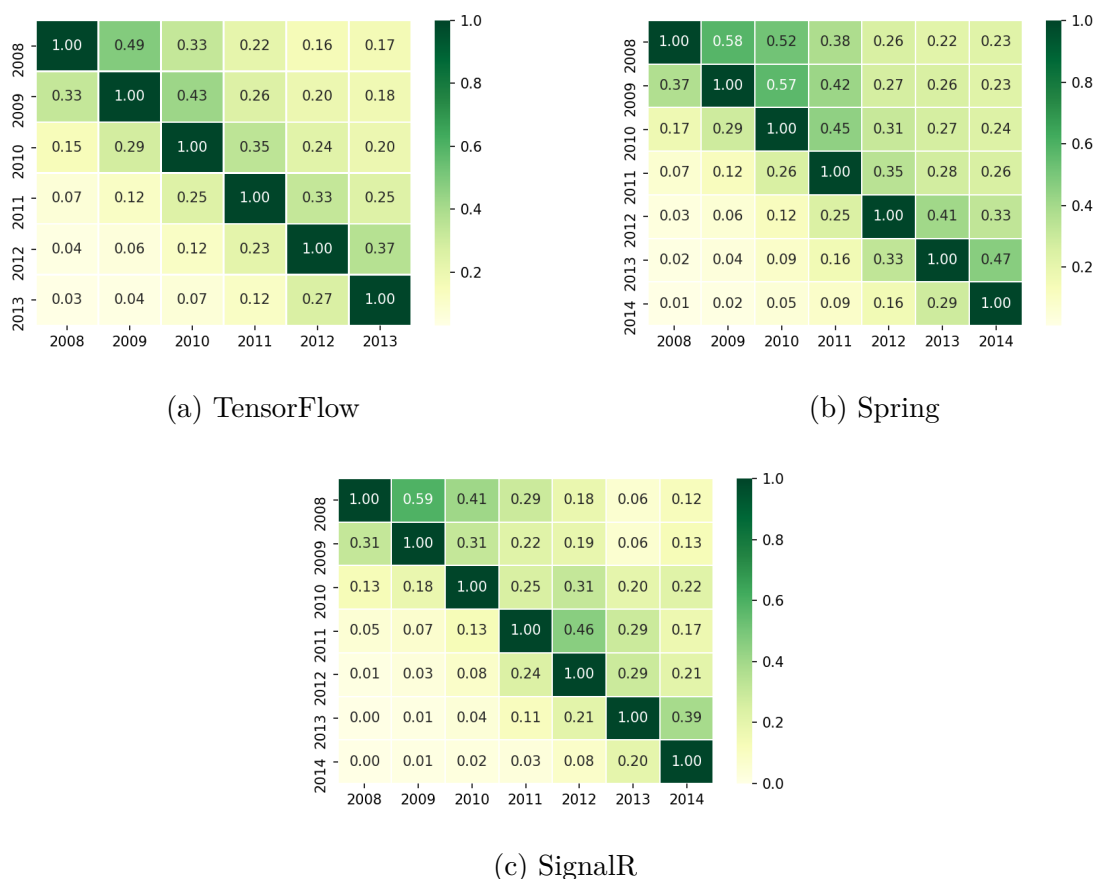


Figura 6.4: Matriz da estabilidade de colaboradores no ranking de reputação ao longo dos anos

A seguir, focamos nossa análise na possível presença do mecanismo de *image scoring* nas redes definidas. A Tabela 6.5 apresenta o número de repositórios de cada grupo. Ela também mostra a quantidade de novos colaboradores que esses repositórios receberam no ano seguinte em cada grupo. Repositórios do grupo **R1** receberam mais colaboradores que **R2**. No ecossistema TensorFlow, entre 2009 e 2011 o número de novos colaboradores em **R1** foi de 2 a 20 vezes maior do que em **R2**. Tensorflow foi o único ecossistema em que o número de novos colaboradores em **R1** não superou o **R2** durante todos os anos. No ecossistema Spring, **R1** recebeu de 2 a 75 vezes mais colaboradores que **R2** de 2008 a 2013. No mesmo período, em SignalR, o número de novos colaboradores nos repositórios de **R1** foi de 2 a 32 vezes maior do que em **R2**.

No entanto, como **R1** tem mais repositórios que **R2**, calculamos a atração de colaboradores de cada grupo ponderado pelo número de repositórios (\mathcal{A}_{R_1} e \mathcal{A}_{R_2}), conforme

descrevemos na Seção 6.1. A partir desses valores, calculamos a força de atração de **R1** (métrica Φ). Os resultados são apresentados na Tabela 6.6.

Em geral, o grupo **R1** apresentou mais colaboradores por repositório do que o grupo **R2**. No ecossistema TensorFlow, entre 2008 e 2010, o grupo **R1** atraiu de 1,37 a 2,96 vezes mais colaboradores que **R2**. No ecossistema Spring, o grupo **R1** chegou a atrair de 2,48 a 22,51 vezes mais colaboradores que **R2**. Por sua vez, o grupo **R1** no ecossistema SignalR atraiu de 1,02 a 2,33 vezes mais colaboradores que **R2**. Somente em dois anos (2011-2012), para o ecossistema TensorFlow, que **R2** superou **R1**, em cerca de duas vezes o número de desenvolvedores. Isso ocorreu uma única vez no ecossistema Spring (2011), quando **R2** atraiu 39% mais desenvolvedores que **R1**.

Levando em consideração que na maior parte do tempo **R1** atraiu mais colaboradores que **R2**, e com intensidades muito maiores do que nos casos em que **R2** superou **R1**, podemos conjecturar a existência do mecanismo *image scoring* como um dos possíveis fatores que incentivam a colaboração nesses ecossistemas.

Ecossistema	Ano base	Grupo R1		Grupo R2	
		# repositórios	# novos colaboradores no ano base + 1	# repositórios	# novos colaboradores no ano base + 1
TensorFlow	2008	310	7985	42	394
	2009	445	12118	125	1250
	2010	706	17402	440	7933
	2011	1250	25555	1199	44990
	2012	1752	32422	1894	71202
Spring	2008	61	1018	16	54
	2009	94	2793	28	37
	2010	193	5280	64	243
	2011	409	5352	142	2579
	2012	677	9060	148	800
	2013	994	16495	202	1202
SignalR	2008	10	161	4	5
	2009	16	211	11	66
	2010	35	508	20	129
	2011	98	978	50	490
	2012	147	1473	86	369
	2013	235	3030	143	835

Tabela 6.5: Quantidade de repositórios e novos colaboradores em cada ano no grupo **R1** e **R2**

6.2.2 Modelos de Benefícios por Produtos

Para verificar se há sinergismo como mecanismo colaborativo no GitHub, primeiramente foi calculada a distribuição do número de colaboradores por repositório em cada um dos ecossistemas analisados nesta dissertação. A Tabela 6.7 exhibe os resultados. Em todos os ecossistemas, metade dos repositórios tem até 2 colaboradores, enquanto 90% tem até 40 no ecossistema TensorFlow, 16 no ecossistema Spring, e 17 em SignalR.

A partir da distribuição dividimos os repositórios entre os 5 grupos definidos na

Ecosistema	Ano	\mathcal{A}_{R_1}	\mathcal{A}_{R_2}	Φ
TensorFlow	2008	25,76	9,38	2,96
	2009	27,23	10	2,72
	2010	24,65	18,03	1,37
	2011	20,44	37,52	0,54
	2012	18,50	37,59	0,49
Spring	2008	16,69	3,37	4,95
	2009	29,71	1,32	22,51
	2010	27,36	3,80	7,20
	2011	13,08	18,16	0,72
	2012	13,38	5,40	2,48
	2013	16,59	5,95	2,79
SignalR	2008	1,61	1,25	1,29
	2009	13,19	6,00	2,20
	2010	14,51	6,45	2,25
	2011	9,98	9,8	1,02
	2012	10,02	4,29	2,33
	2013	12,89	5,84	2,21

Tabela 6.6: Número de novos colaboradores por repositório (\mathcal{A}_{R_1} e \mathcal{A}_{R_2}) e força de atração do grupo R1 (Φ)

Ecosistema	min	25%	50%	75%	90%	max
TensorFlow	1	1	2	10	40	1869
Spring	1	1	2	5	16	1703
SignalR	1	1	2	6	17	402

Tabela 6.7: Distribuição do número de colaboradores por repositório nos ecossistemas analisados

Seção 6.1, com base no total de colaboradores em cada repositório. Para identificarmos a presença ou não de sinergismo, definimos como limiar a mediana do número de commits para cada grupo de cada uma das redes. A Tabela 6.8 apresenta os resultados. No ecossistema TensorFlow, em todos os anos a mediana foi progressiva entre os grupos, isto é, quanto maior o número de desenvolvedores colaborando no repositório, maior a mediana de commits encontrada. Esse resultado se repetiu no ecossistema Spring e também em SignalR; este último com duas exceções: em 2009, o grupo G4 apresentou uma mediana maior que G5, e em 2010, o grupo G2 apresentou uma mediana maior que G3.

Um outro fenômeno interessante ocorreu: ao longo dos anos, a mediana de commits em G5 diminuiu em todos os ecossistemas. Ao mesmo tempo, os ecossistemas cresceram. Isso pode estar relacionado a excesso de oferta na rede, isto é, com o surgimento de mais repositórios, os colaboradores podem dividir a contribuição em mais repositórios.

Esses resultados dão indícios de que os colaboradores reconhecem que a colaboração

Ecosistema	Ano	G1	G2	G3	G4	G5
TensorFlow	2008	3,0	14,5	92,0	521,5	1637,0
	2009	2,0	16,5	52,0	562,0	1238,0
	2010	1,0	10,0	33,0	265,0	1215,5
	2011	2,0	5,0	26,0	163,0	1083,0
	2012	3,0	10,0	27,0	120,0	688,0
	2013	5,0	14,0	36,0	114,5	631,0
Spring	2008	12,0	78,0	338,0	474,5	1211,0
	2009	3,0	32,0	140,0	390,0	1201,0
	2010	3,5	29,0	60,0	250,0	1155,5
	2011	3,5	13,0	22,5	93,0	839,0
	2012	4,0	13,0	32,0	86,5	644,0
	2013	5,0	14,0	40,0	102,0	820,0
	2014	6,0	15,0	39,0	138,0	985,0
SignalR	2008	3,5	0,0	42,0	1333,0	3342,0
	2009	2,5	19,0	31,0	2019,0	1626,0
	2010	4,0	53,5	46,0	102,0	2157,5
	2011	2,0	30,0	31,0	129,0	1256,0
	2012	3,0	8,5	36,0	167,0	458,0
	2013	7,0	22,0	67,5	278,5	711,0
	2014	5,0	17,0	37,0	213,0	629,5

Tabela 6.8: Mediana da distribuição de commits por grupo de repositórios

em grandes grupos é mais benéfica, o que evidencia o mecanismo de sinergismo.

6.2.3 Modelos de Genes Compartilhados

Para verificar se a estratégia do modelo *kin choice* é também responsável pelo fenômeno colaborativo observado nos ecossistemas TensorFlow, Spring e SignalR, calculamos a conformidade de cada um dos colaboradores com relação às linguagens de programação, conforme apresentado na Seção 6.1. A Figura 6.5 apresenta um histograma da distribuição da métrica de conformidade C_L^D .

Podemos observar que a conformidade é total ou muito alta para mais de 90% dos colaboradores dos ecossistemas analisados. No entanto, um fenômeno interessante ocorreu: nos demais casos (menos de 10%), ou a conformidade é totalmente inconforme ou muito fraca. Desse resultado díspare, pode-se pensar em duas situações: (1) desenvolvedores que preferem engajar em colaborações nas quais as linguagens utilizadas sejam muitos diferentes; (2) desenvolvedores que precisam colaborar com linguagens muito diferentes.

O primeiro caso pode se tratar de um desenvolvedor que gosta de praticar linguagens de diferentes paradigmas, buscando, por exemplo, o desenvolvimento em diferentes repositórios (aumentando a chance de encontrar linguagens totalmente diferentes). O segundo caso ocorre quando é preciso combinar em um mesmo sistema de software, duas ou mais linguagens, que podem ser de origens diferentes e ter propósitos diversos, o que é comum em aplicações Web, por exemplo, em que se combina diferentes linguagens, cada uma para uma responsabilidade do sistema.

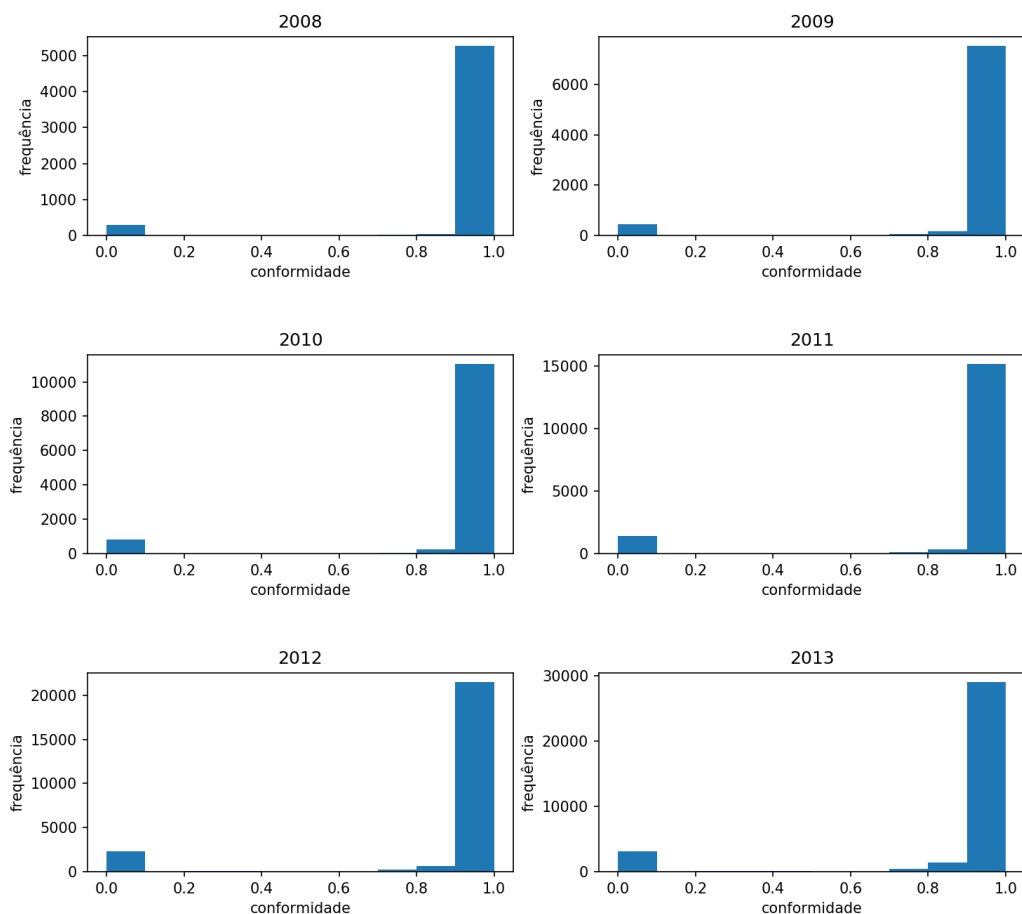
Esses resultados corroboram a principal ideia do modelo *kin choice*, que é a preferência de colaborar com semelhantes. Na biologia, esse resultado está relacionado com o fato de que colaborar com o semelhante aumenta as chances de sobrevivência de genes parecidos, o que pode ser favorecido pela seleção natural. No caso em estudo, como não há propriamente genes envolvidos no processo, essa mimetização é intrigante e merecedora de investigações mais profundas.

6.3 Discussão

As análises aqui apresentadas tinham como objetivo responder às seguintes questões: (1) é possível definir métricas simples que representam quantitativamente a semântica dos conceitos de modelos biológicos que descrevem colaboração entre humanos? (2) Quais são os comportamentos encontrados em cada um dos ecossistemas, à luz dos conceitos dos modelos biológicos? (3) Quais as limitações das nossas análises?

Neste capítulo, definimos métricas que descrevem modelos de reciprocidade direcionada, modelos de benefícios por produtos e modelos de genes compartilhados. Para descrever modelos de reciprocidade direcionada, utilizamos o conceito de *image scoring*, e definimos uma métrica de reputação, baseada na atividade colaborativa (commits) do GitHub. Apresentamos também um método de avaliação do sinergismo, um mecanismo de colaboração que ocorre nos modelos de benefícios por produtos. Por fim, introduzimos uma métrica simples para conformidade do desenvolvedor com relação às linguagens de programação, de modo a reproduzir o modelo de genes compartilhados *kin choice*. Nossas abordagens estão sempre centradas nas atividades (commits) feitas pelos desenvolvedores e nas suas relações dentro desses ecossistemas.

Os resultados de nossas análises corroboram com o senso comum de que modelar e analisar processos de colaboração em diferentes sistemas é uma tarefa extremamente complexa. Avaliamos diferentes mecanismos que, combinados entre si, podem influenciar um desenvolvedor na decisão de investir esforços (tempo e conhecimento, por exemplo) para colaborar em redes de desenvolvimento de software.

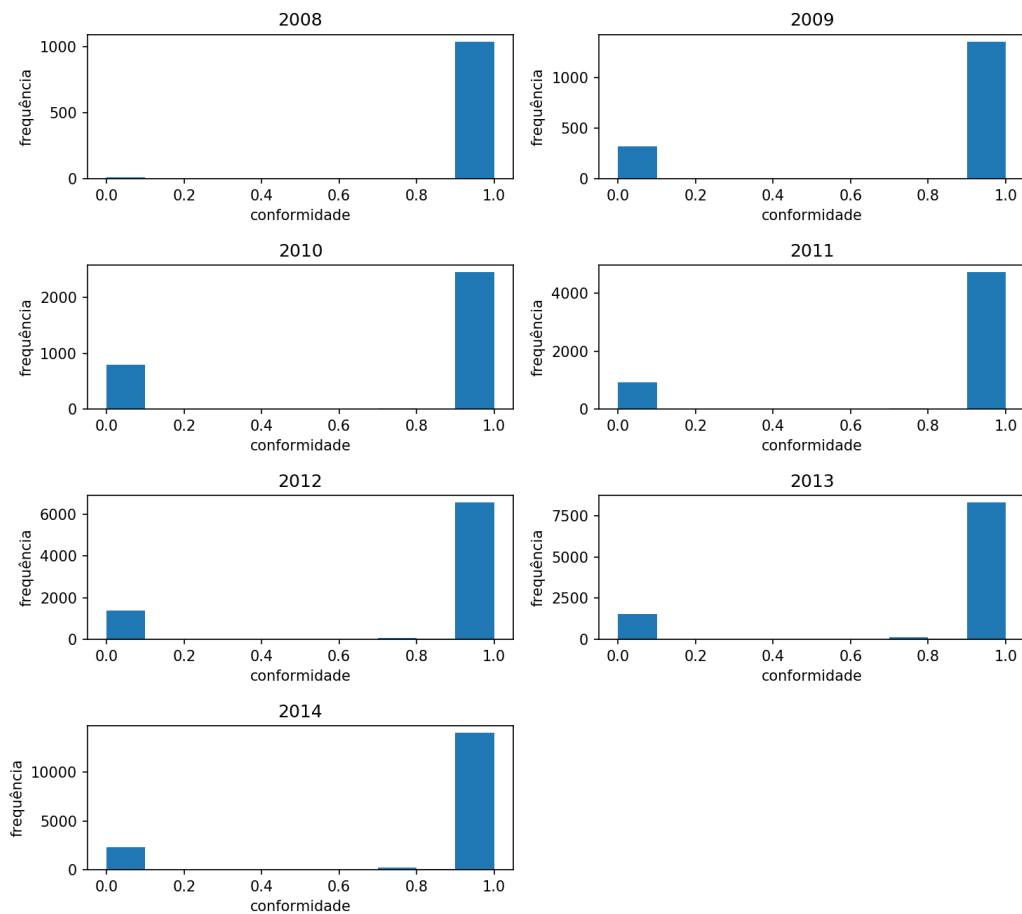


(a) TensorFlow

Figura 6.5: Distribuição da conformidade dos colaboradores com relação à linguagem nas redes de colaboração

Na análise de persistência da colaboração foi possível observar que apesar de as taxas de abandono da colaboração serem altas, há uma porcentagem significativa (em torno de 30%) dos colaboradores que se mantém fiéis colaborando na rede. Esse resultado traz duas reflexões: (1) reforça que o processo de colaboração é custoso, e que precisamos entender quais são esses custos para aproveitar a alta capacidade de atração de novos colaboradores que esses ecossistemas possuem; (2) o processo de colaboração traz altos benefícios, que fazem com que as pessoas permaneçam colaborando ao longo de muitos anos.

Encontramos fortes evidências de que *image scoring* é um dos mecanismos envolvidos nessa alta capacidade de atração dos ecossistemas. Esse resultado confirma os encontrados por Melis & Semmann [2010] que mostram que a colaboração entre huma-

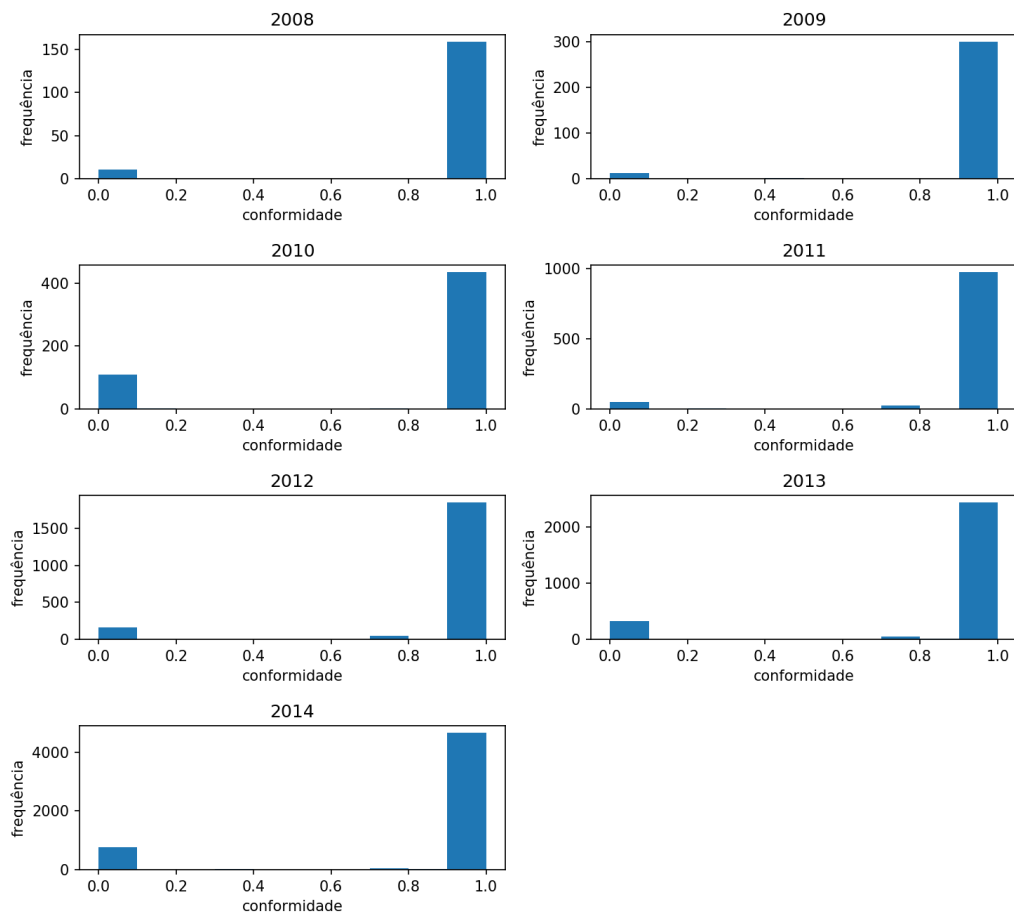


(b) Spring

Figura 6.5: Distribuição da conformidade dos colaboradores com relação à linguagem nas redes de colaboração (cont.)

nos é marcada pela construção da reputação, e por Wedekind & Milinski [2000], que demonstram que humanos tendem a colaborar com outros de alta reputação.

Mostramos também que os colaboradores com as maiores reputações têm dificuldade de mantê-las, o que pode estar relacionado aos custos já mencionados. No entanto, as distribuições das reputações ao longo dos anos apresentam comportamentos muito similares entre si. Isso indica que os colaboradores continuam tão produtivos quanto antes, e que para a evolução do sistema de colaboração, essa alternância no topo da distribuição pode ser benéfica: os colaboradores com maior reputação são também aqueles que mais injetam conhecimento no repositório, trazendo no código em que escrevem todas suas experiências progressas. Esse pode ser um dos mecanismos que provoca a evolução da colaboração nos ecossistemas.



(c) SignalR

Figura 6.5: Distribuição da conformidade dos colaboradores com relação à linguagem nas redes de colaboração (cont.)

Nossas análises mostram que é mais provável encontrar muitos commits em repositórios com muitos colaboradores. Isso pode sugerir o sinergismo, mecanismo que é marcado pelo alto número de benefícios quanto maior o tamanho do grupo. No sinergismo, os colaboradores se reúnem em grupos maiores, porque reconhecem que terão mais benefícios dessa forma. No GitHub, isso se expressa a partir da alta colaboração em projetos de maior impacto social. Esse resultado está de acordo com o trabalho de Barclay & Willer [2006], que relaciona comportamentos egoístas a altos benefícios.

Nas redes analisadas, encontramos uma conformidade muito alta com relação às linguagens de programação. A nossa análise deixa indícios de que o mecanismo *kin choice* está presente no tipo de sistema em estudo. Nessas redes, desenvolvedores tendem a focar em uma única linguagem ou linguagens muito parecidas. Se por um lado isso indica que

essas redes são propícias para a formação de especialistas, por outro lado também indica que existe o risco de a alta convergência afetar os resultados emergentes da colaboração. Cada linguagem de programação é dedicada a resolver um tipo de problema, então o risco é que ao estabelecer consenso, alguns problemas que precisam ser levados em consideração sejam deixados de lado, devido à orientação da linguagem.

Esses resultados estão intimamente relacionados à forma como modelamos a colaboração, através da ideia de ecossistema de colaboração. A partir do modelo proposto, é possível: (1) avaliar os efeitos da linguagem sobre a colaboração; (2) verificar a evolução da colaboração evoluindo ao redor de pessoas com alta reputação e; (3) a alta alternância dos desenvolvedores com maiores reputações. Poderíamos encontrar indícios de benefícios por produtos utilizando análises estatísticas básicas, correlacionando tamanho do repositório a número de commits, em amostras de repositórios aleatórios. No entanto, nosso modelo considera uma única população, o que demonstra uma preferência da mesma população por repositórios que provoquem altos benefícios.

Ao considerarmos ecossistemas distintos nas nossas análises, o objetivo era verificar se o comportamento dos desenvolvedores permanecia o mesmo, independente das características inerentes a cada um dos ecossistemas. No entanto, nossas análises com modelos biológicos mostraram que o comportamento é comum aos ecossistemas, o que torna mais forte a tese de que a colaboração é dependente do colaborador, e por isso deve ser, sempre que possível, focada nele.

Uma possível limitação das nossas análises com modelos biológicos é que foram considerados apenas alguns aspectos específicos da colaboração no GitHub, como commits, linguagens de programação e repositórios. Outras facetas que também podem influenciar o processo de colaboração, como por exemplo, aspectos sociais do próprio GitHub, e também os externos, relativos ao colaborador em sua vida fora do GitHub, não foram consideradas nas nossas análises. Além disso, analisamos apenas três ecossistemas, o que já nos proporcionou uma boa visão do comportamento colaborativo no GitHub, mas ao ampliar esse número poderemos ter conclusões ainda mais abrangentes sobre esses fenômenos.

Apesar das limitações conhecidas e descritas, nossas análises corroboram com a afirmação de que o processo de colaboração em redes de desenvolvimento colaborativo de software é complexo e possui muitas facetas. Assim, acreditamos que os resultados apresentados neste trabalho são o primeiro passo para descrever o processo de colaboração (surgimento e persistência) em redes de desenvolvimento colaborativo de software sob a ótica biológica. Desta forma, ao descrevermos a colaboração com base em modelos biológicos, podemos aplicar teorias de diferentes áreas para propor novos sistemas colaborativos mais robustos, eficientes, e que otimize os benefícios advindos do ato de colaborar.

Capítulo 7

Conclusão

Nesta dissertação, analisamos redes de desenvolvimento colaborativo de software com o objetivo de entender se mecanismos e princípios que regem os modelos biológicos podem ser utilizados para complementar o entendimento em torno do processo colaborativo inerente à estas redes.

Visando responder a esse questionamento, primeiramente apresentamos o conceito de “ecossistema de colaboração”, que foi utilizado nesta dissertação como forma de reproduzir o conceito biológico de ecossistema. Esses ecossistemas, assim como os biológicos, tem a intenção de incluir os fatores bióticos (colaboradores) e abióticos (repositórios e linguagens de programação) que estão relacionados com a colaboração em estudo. Adicionalmente, esses ecossistemas são relativamente fechados no tempo e devem incluir aspectos dinâmicos, que permitam reproduzir a história da colaboração.

Nossas análises foram realizadas com os ecossistemas de colaboração que construímos a partir de metadados do GitHub. Para construir esses ecossistemas, selecionamos três linguagens aleatórias, e três repositórios (os quais nomeamos repositórios-raiz) de tamanhos distintos, em busca de perceber as semelhanças e/ou diferenças entre o comportamento colaborativo deles. Esses ecossistemas consistem de um conjunto de commits, que armazenam informações que possibilitam reconstruir a história da colaboração no GitHub.

A partir da caracterização dos ecossistemas de colaboração TensorFlow, Spring e SignalR, encontramos algumas características comuns entre os três grupos, a despeito do diferente número de desenvolvedores, repositórios e commits. A intensidade da colaboração é crescente em todos os ecossistemas, porém em diferentes taxas. Além disso, a ocorrência de somente poucos repositórios com mais de 100 commits foi uma característica geral entre os ecossistemas. O uso de *forks* era amplo nesses ecossistemas, o que traz uma noção do grau de “interesse colaborativo” deles. Essa noção também foi expressa pelo fato de que a maior parte dos repositórios que compõem os ecossistemas estão ativos e recebendo commits. Essa característica dinâmica da colaboração também se revela pela altas taxas de entrada e saída de desenvolvedores do ecossistema ano a ano.

Dado que dinâmica é um fator importante para o entendimento dos modelos biológicos, apresentamos um modelo para a construção de redes de colaboração a partir

dos ecossistemas. Essas redes representam um *snapshot* do ecossistema em intervalos de 1 ano. Nestas redes, os desenvolvedores são representados pelos vértices, e uma aresta existe sempre que dois desenvolvedores realizam um commit em um mesmo repositório. Assim, caracterizamos essas redes utilizando métricas clássicas de redes complexas. As redes dos ecossistemas também apresentaram algumas características comuns: a distribuição de graus possui cauda pesada, o que significa que poucos colaboradores possuem um número muito grande de pares. Além disso, essas redes eram pouco densas, o que indica a dificuldade de estabelecer (e manter) relações colaborativas.

Outra característica relevante dessas redes é que elas são formadas por muitos componentes, e como são dinâmicas, esses componentes podem aparecer conectados/desconectados em um tempo t e desconectados/conectados em um tempo $t + 1$. Em geral, o maior componente dessas redes aparece com mais da metade dos colaboradores.

Consideramos o aspecto temporal das redes de colaboração para analisá-las a luz de modelos biológicos. Definimos métricas que descrevem modelos de reciprocidade direcionada, modelos de benefícios por produtos e modelos de genes compartilhados. Essas métricas são focadas em alguns aspectos do GitHub, como os commits e linguagens de programação.

Com base nos conceitos encontrados nos modelos de reciprocidade direcionada, encontramos indícios de que a colaboração segue princípios do modelo *partner choice*, em que os colaboradores desenvolvem um mecanismo para decidir o parceiro de colaboração. Nossos resultados mostraram que possivelmente *image scoring* é um dos mecanismos utilizados, segundo a definição de reputação da métrica que propomos. Nessas redes, desenvolvedores com alta reputação atraem novos desenvolvedores para seus repositórios. Observamos, também, que a colaboração no GitHub é composta por uma porcentagem significativa de usuários fieis, que mantêm a colaboração ao longo dos anos. Esse comportamento convive com outro oposto: as altas taxas de abandono, que sinalizam para os custos da colaboração.

Considerando as orientações dos modelos de benefícios por produtos, que se concentram na colaboração não-intencional, verificamos que é mais provável encontrar muitos commits em repositórios com muitos colaboradores. Os resultados encontrados sugerem a existência do mecanismo de sinergismo, explicado pelos modelos de benefícios por produtos. Adicionalmente, adaptamos os modelos de genes compartilhados a um contexto virtual, em que genes não estão interagindo diretamente. Nas redes analisadas, desenvolvedores tendem a focar em uma única linguagem ou linguagens muito parecidas. Esses resultados levantam dúvidas sobre os riscos de uma rede altamente convergente. Cada linguagem de programação é dedicada a resolver um tipo de problema, então o risco é que, ao estabelecer consenso, alguns problemas que precisam ser levados em consideração sejam deixados de lado, devido à orientação da linguagem.

Por fim, conforme citado no Capítulo 1, a metodologia proposta neste trabalho,

que tem como principal pilar os princípios de modelos biológicos, visa complementar os demais trabalhos na literatura, oferecendo uma nova faceta para uma análise mais aprofundada do processo de colaboração em sistemas colaborativos de desenvolvimento de software. Uma vez que este é o primeiro passo para estudar as redes de desenvolvimento colaborativo de software à luz de modelos biológicos, buscamos explorar os princípios e mecanismos dos modelos de forma mais simples possível. A simplicidade das nossas análises é fundamental para introduzirmos, de forma compreensível, explicações de cunho biológico para o entendimento da colaboração no GitHub. A partir da análise apresentada nesta dissertação, vislumbramos as seguintes direções para trabalhos futuros: (1) inclusão de aspectos sociais como as funcionalidades *stars* e *followers* de forma complementar a métrica de reputação aqui proposta; (2) utilização do tempo de vida de repositórios e o aumento da reputação do usuário como definições de benefícios por produto e; (3) exploração de mais informações para o aumento da diversidade do ecossistema a ser analisado, enriquecendo as análises do processo de colaboração sob a ótica de modelos biológicos.

Referências Bibliográficas

- Abbasi, A.; Hossain, L.; Uddin, M. S. & Rasmussen, K. J. R. (2011). Evolutionary dynamics of scientific collaboration networks: Multi-levels and cross-time analysis. *CoRR*, abs/1107.5870.
- Adams, J. (2012). Collaborations: The rise of research networks. *Nature*, 490(7420):335-336.
- Alexander, R. D. (1987). *The biology of moral systems*. Aldine de Gruyter.
- Allaho, M. Y. & Lee, W.-C. (2013). Analyzing the social ties and structure of contributors in open source software community. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM '13*, pp. 56--60, New York, NY, USA. ACM.
- Axelrod, R. & Hamilton, W. D. (1981). The evolution of cooperation. *Science*, 211(4489):1390--1396.
- Barabasi, A.-L. (2003). *Linked: How Everything is Connected to Everything Else and What it Means for Business, Science and Everyday Life*. Plume Books.
- Barclay, P. & Raihani, N. (2016). Partner choice versus punishment in human prisoners dilemmas. *Evolution and Human Behavior*, 37(4):263 – 271. ISSN 1090-5138.
- Barclay, P. & Willer, R. (2006). Partner choice creates competitive altruism in humans. *Proceedings of the Royal Society B: Biological Sciences*, 274(1610):749--753.
- Batista, N. A.; Brandão, M. A.; Alves, G. B.; da Silva, A. P. C. & Moro, M. M. (2017). Collaboration strength metrics and analyses on github. In *Proceedings of the International Conference on Web Intelligence*, pp. 170--178. ACM.
- Batista, N. A.; Sousa, G. A.; Brandão, M. A.; da Silva, A. P. C. & Moro, M. M. (2018). Tie strength metrics to rank pairs of developers from github. *Journal of Information and Data Management*, 9(1):69--69.
- Baudry, B. & Monperrus, M. (2012). Towards ecology inspired software engineering. *arXiv preprint arXiv:1205.1102*.
- Boyd, R. & Richerson, P. J. (2009). Culture and the evolution of human cooperation. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1533):3281–3288.

- Bull, J. & Rice, W. (1991). Distinguishing mechanisms for the evolution of co-operation. *Journal of Theoretical Biology*, 149(1):63--74.
- Casalnuovo, C.; Vasilescu, B.; Devanbu, P. & Filkov, V. (2015). Developer onboarding in github: the role of prior social links and language experience. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pp. 817--828.
- Cockburn, A. (1998). Evolution of helping behavior in cooperatively breeding birds. *Annual Review of Ecology and Systematics*, 29:141--177. ISSN 00664162.
- Coelho, J. & Valente, M. T. (2017). Why modern open source projects fail. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017*, p. 186196, New York, NY, USA. Association for Computing Machinery.
- Connor, R. (2008). The benefits of mutualism: A conceptual framework. *Biological Reviews*, 70:427 - 457.
- Dabbish, L.; Stuart, C.; Tsay, J. & Herbsleb, J. (2012). Social coding in github: Transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work, CSCW '12*, pp. 1277--1286, New York, NY, USA. ACM.
- De Luca Pretto, N. & da Silveira, S. (2008). *Além das redes de colaboração: internet, diversidade cultural e tecnologias do poder*. SciELO - EDUFBA. ISBN 9788523208899.
- El Asri, I.; Kerzazi, N.; Benhiba, L. & Janati, M. (2017). From periphery to core: a temporal analysis of github contributors collaboration network. In *Working Conference on Virtual Enterprises*, pp. 217--229. Springer.
- Erdős, P. & Rényi, A. (1960). On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.*, 5(1):17--60.
- Eshel, I. & Cavalli-Sforza, L. L. (1982). Assortment of encounters and evolution of cooperativeness. *Proceedings of the National Academy of Sciences*, 79(4):1331--1335.
- Fehr, E. & Schurtenberger, I. (2018). Normative foundations of human cooperation. *Nature Human Behaviour*, 2(7):458--468.
- Gousios, G. (2013). The GHTorrent dataset and tool suite. In *Proceedings of the 10th Working Conference on Mining Software Repositories, MSR '13*, pp. 233--236.
- Gracia-Lázaro, C.; Ferrer, A.; Ruiz, G.; Tarancón, A.; Cuesta, J. A.; Sánchez, A. & Moreno, Y. (2012). Heterogeneous networks do not promote cooperation when humans play a prisoner's dilemma. *Proceedings of the National Academy of Sciences*, 109(32):12922--12926. ISSN 0027-8424.

- Hamilton, W. D. (1963). The genetical evolution of social behavior. *Journal of Theoretical Biology*, 7:1–16.
- Hu, Y.; Wang, S.; Ren, Y. & Choo, K.-K. R. (2018). User influence analysis for github developer social networks. *Expert Systems with Applications*, pp. 108 – 118. ISSN 0957-4174.
- Lima, A.; Rossi, L. & Musolesi, M. (2014). Coding together at scale: Github as a collaborative social network. In *Proceedings of 8th AAAI International Conference on Weblogs and Social Media (ICWSM 2014)*.
- Lindeman, R. L. (1942). The trophic-dynamic aspect of ecology. *Ecology*, 23(4):399–417.
- Lopes, H.; Santos, A. C. & Teles, N. (2009). The motives for cooperation in work organizations. *Journal of Institutional Economics*, 5(3):315338.
- McDonald, N.; Blincoe, K.; Petakovic, E. & Goggins, S. (2014). Modeling distributed collaboration on github. *Advances in Complex Systems*, 17(07n08):1450024.
- Melis, A. & Semmann, D. (2010). How is human cooperation different? *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 365:2663–74.
- Mens, T. & Grosjean, P. (2015). The ecology of software ecosystems. *Computer*, 48:85–87.
- Mitchell, M. & Newman, M. (2002). Complex systems theory and evolution. *Encyclopedia of Evolution*, pp. 1--5.
- Newman, M. E. (2001). The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404--409.
- Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM REVIEW*, 45:167--256.
- Noë, R. (1990). A veto game played by baboons: a challenge to the use of the prisoner's dilemma as a paradigm for reciprocity and cooperation. *Animal Behaviour*, 39(1):78--90.
- Nowak, M. A. & Sigmund, K. (1998). Evolution of indirect reciprocity by image scoring. *Nature*, 393(6685):573.
- Perkel, J. (2016). Democratic databases: science on GitHub. *Nature*, 538(7623):127--128. ISSN 0028-0836.
- Queller, D. (1985). Kinship, reciprocity, and synergism in the evolution of social behavior. *Nature*, 318:366–367.

- Ramasco, J. J.; Dorogovtsev, S. N. & Pastor-Satorras, R. (2004). Self-organization of collaboration networks. *Phys. Rev. E*, 70:036106.
- Rand, D. G.; Arbesman, S. & Christakis, N. A. (2011). Dynamic social networks promote cooperation in experiments with humans. *Proceedings of the National Academy of Sciences*, 108(48):19193--19198. ISSN 0027-8424.
- Rand, D. G. & Nowak, M. A. (2013). Human cooperation. *Trends in Cognitive Sciences*, 17(8):413 – 425. ISSN 1364-6613.
- Rand, D. G.; Nowak, M. A.; Fowler, J. H. & Christakis, N. A. (2014). Static network structure can stabilize human cooperation. *Proceedings of the National Academy of Sciences*, 111(48):17093--17098. ISSN 0027-8424.
- Richardson, D. S.; Burke, T. & Komdeur, J. (2002). Direct benefits and the evolution of female-biased cooperative breeding in seychelles warblers. *Evolution*, 56(11):2313--2321. ISSN 00143820, 15585646.
- Sachs, J. L.; Mueller, U. G.; Wilcox, T. P. & Bull, J. J. (2004). The evolution of cooperation. *The Quarterly review of biology*, 79(2):135--160.
- Tansley, A. G. (1935). The use and abuse of vegetational concepts and terms. *Ecology*, 16(3):284--307.
- Trivers, R. L. (1971). The evolution of reciprocal altruism. *The Quarterly review of biology*, 46(1):35--57.
- Tucker, A. W. (1983). The mathematics of tucker: A sampler. *The Two-Year College Mathematics Journal*, 14(3):228--232. ISSN 00494925.
- Uzzi, B. & Spiro, J. (2005). Collaboration and creativity: The small world problem. *American journal of sociology*, 111(2):447--504.
- Valverde, S. & Solé, R. V. (2015). Punctuated equilibrium in the large-scale evolution of programming languages. *Journal of the Royal Society, Interface*, 12 107.
- Wedekind, C. & Milinski, M. (2000). Cooperation through image scoring in humans. *Science*, 288(5467):850--852. ISSN 0036-8075.
- Wynne-Edwards, V. C. (1964). Group selection and kin selection. *Nature*, 201(4924):1147.