《开源软件与社区治理》

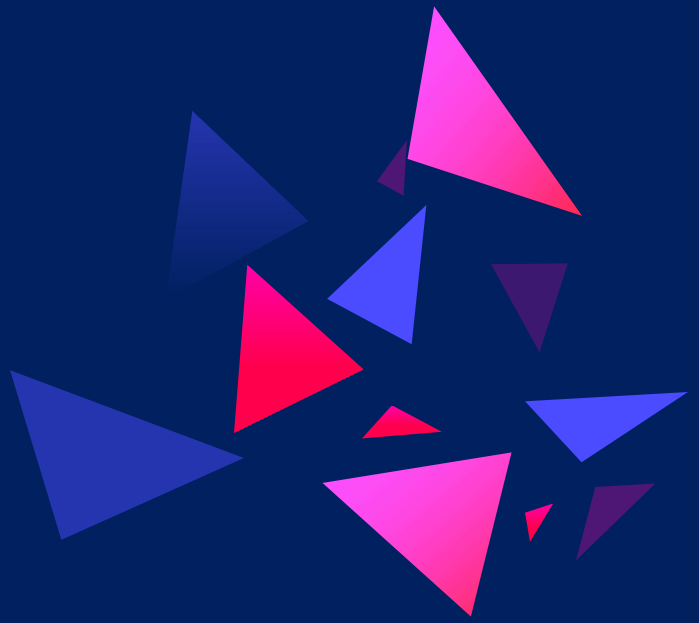
第十三讲 数据驱动的开源研究

X-lab @TianyiChow

开放实验室

CONTENTS

目 录



01 背景

Background

01 数字世界的路与桥

数字基建

开源技术和开源软件深刻影响着人类社会生产的方方面面，构成了现代数字世界的路与桥。

持续演化

开源软件具有进化特性，开源软件在开发者的持续贡献下，以动态演化的方式不断演化进步，软件的功能、形态会随着时间推移而不断进化。

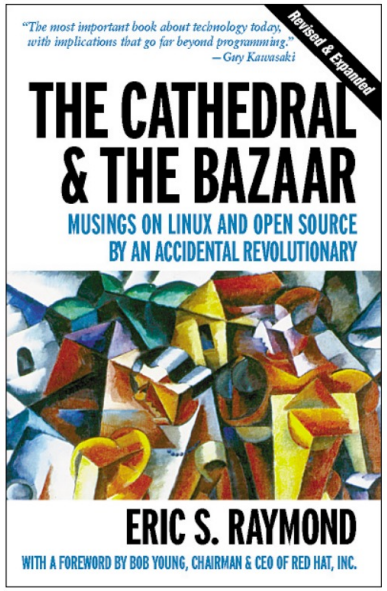
开放协作

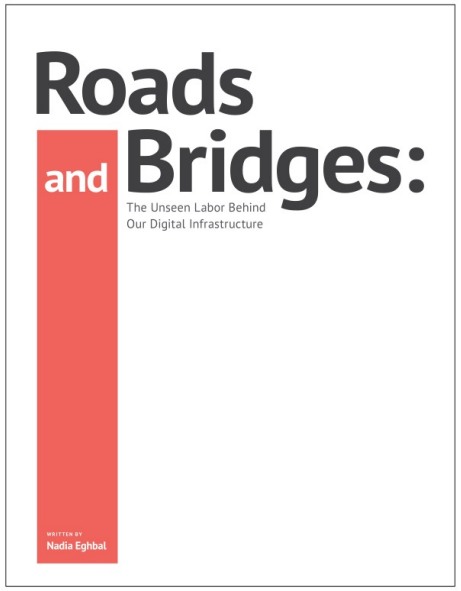
开源协作以分布式的方式开展，在工具的支持下将群体协作规模扩展到空前，数据中记录了其开发者们复杂协作关系和社交语义。

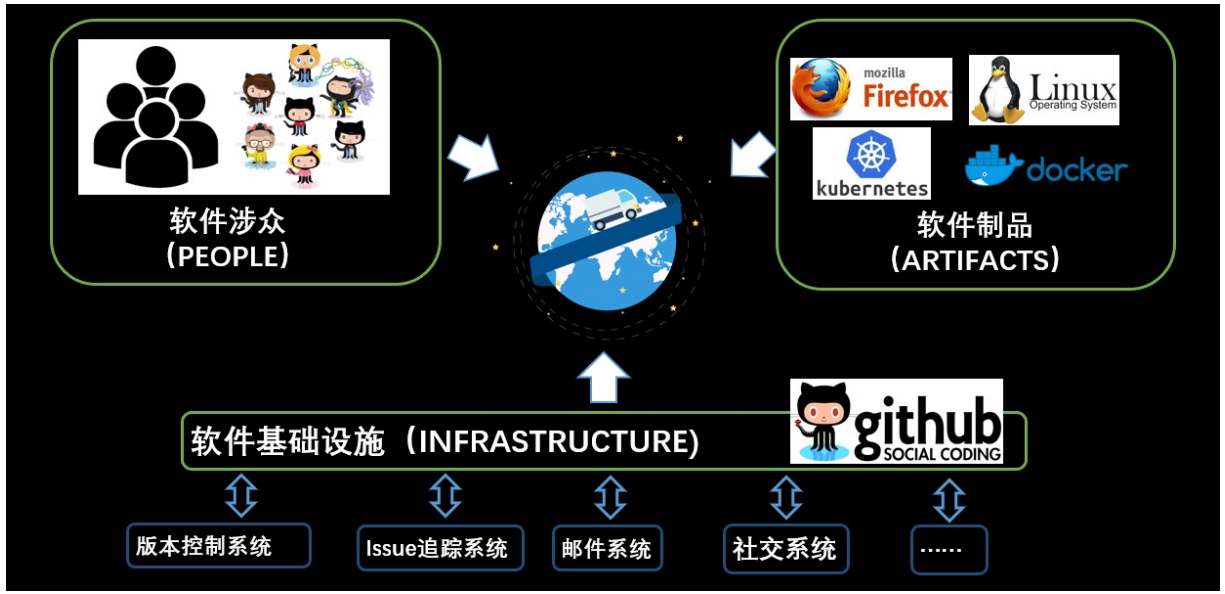
开源生态

开源生态圈日益庞大，洞察理解其宏观和微观结构对指导社区治理、开源软件的可持续发展有重要意义。

1999 → 2016



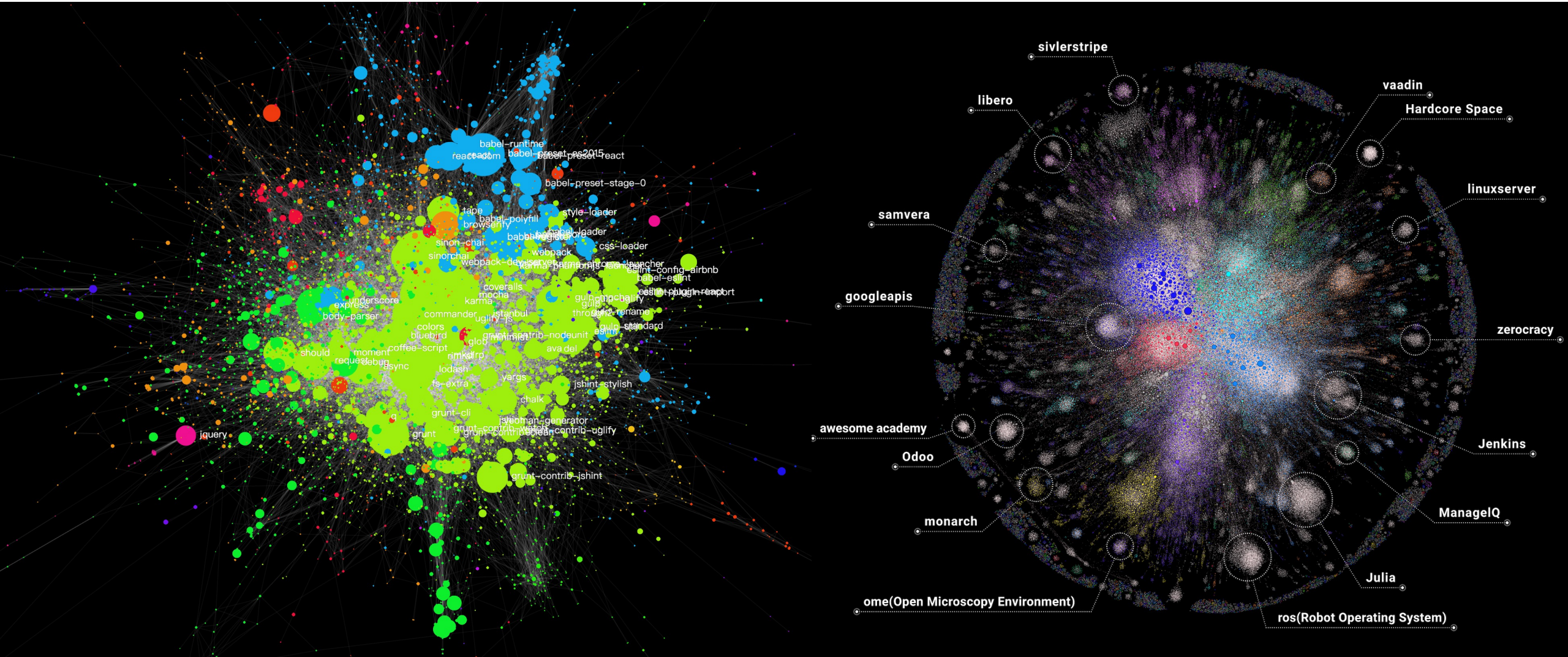




02 规模庞大且复杂的开源世界

诸多问题

开源技术发展趋势会如何？顶级社区是如何在众多同类社区中脱颖而出的？如此大规模群体协作现象背后的支配规则是什么？...

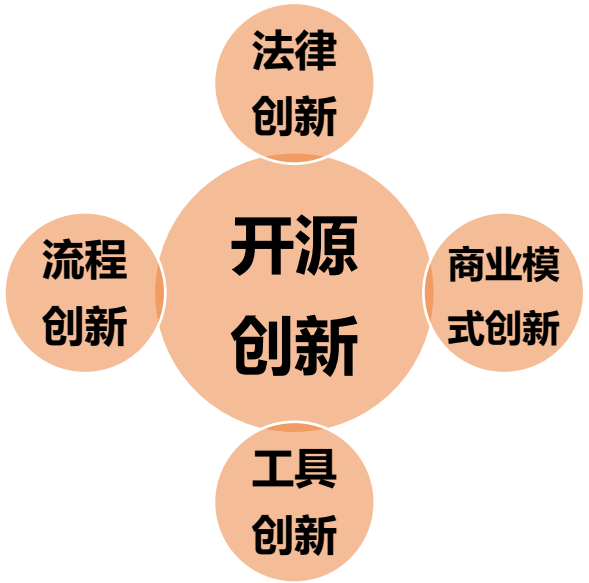


对组织进行“观测-理解-调控”的机会

- **组织**就是在一定的环境中，为实现某种共同的目标，按照一定的结构形式、活动规律结合起来的，具有特定功能的开放系统。
- 简单来说：组织是两个以上的人、目标和特定的人际关系构成的群体；组织是两个以上的人在一起为实现某个共同目标而协同行动的**共同体**。
- 组织类型：
  - 企业、政府、高校
  - 公益组织、社会企业
  - **开源社区**

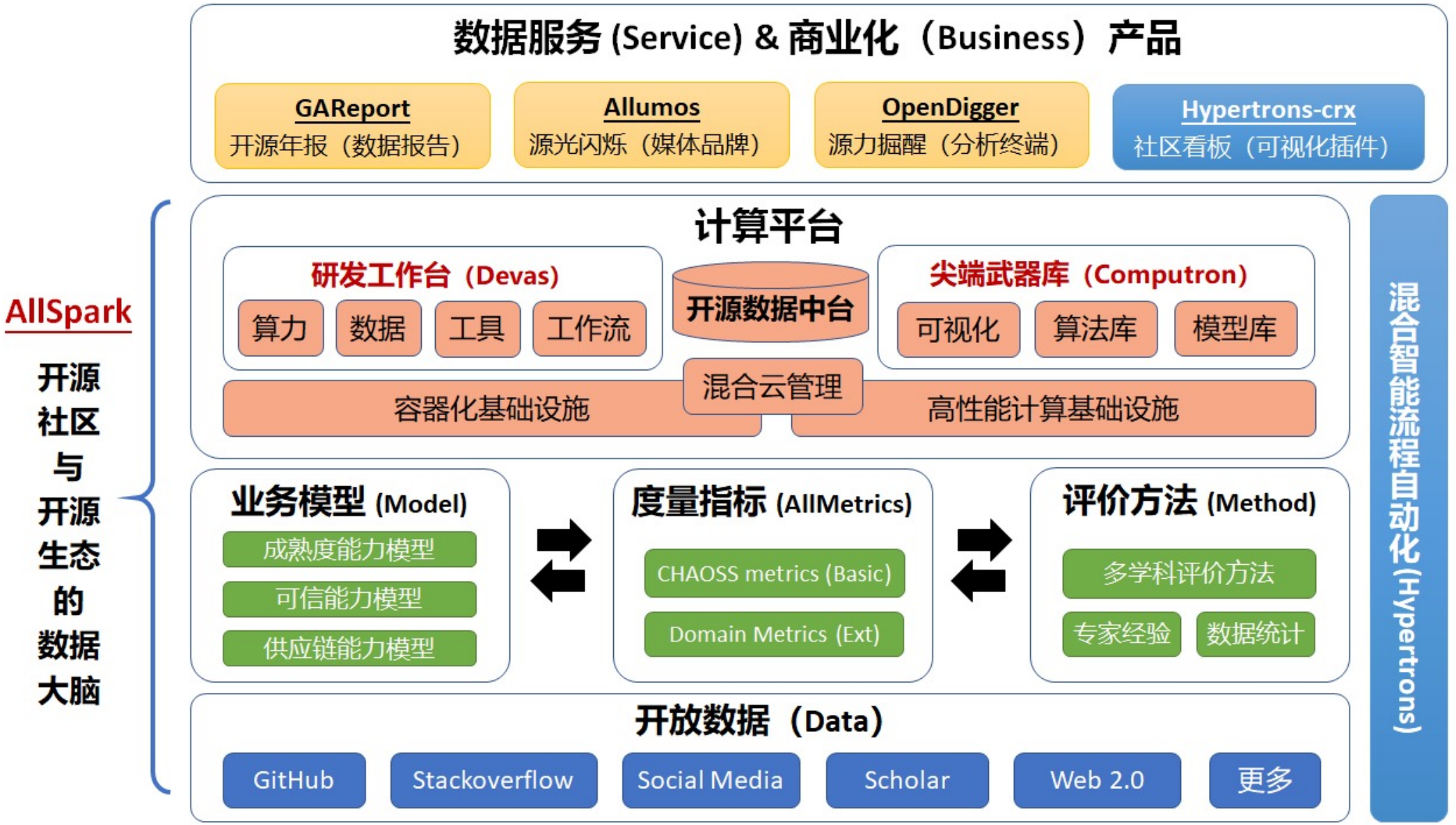
（开放式组织）

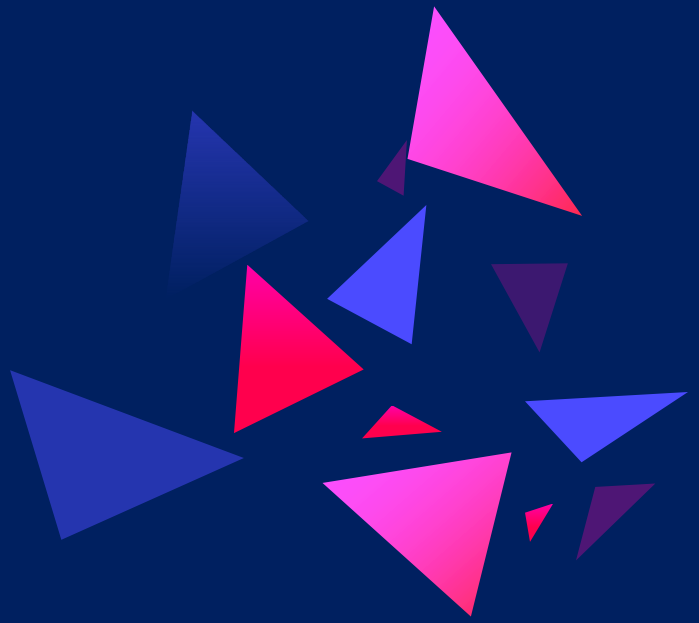
- **数字化**
- **开放**
- **分享**
- **协作**
- **相互依赖**



- **生产力**
- **创新力**
- **竞争力**
- **影响力**

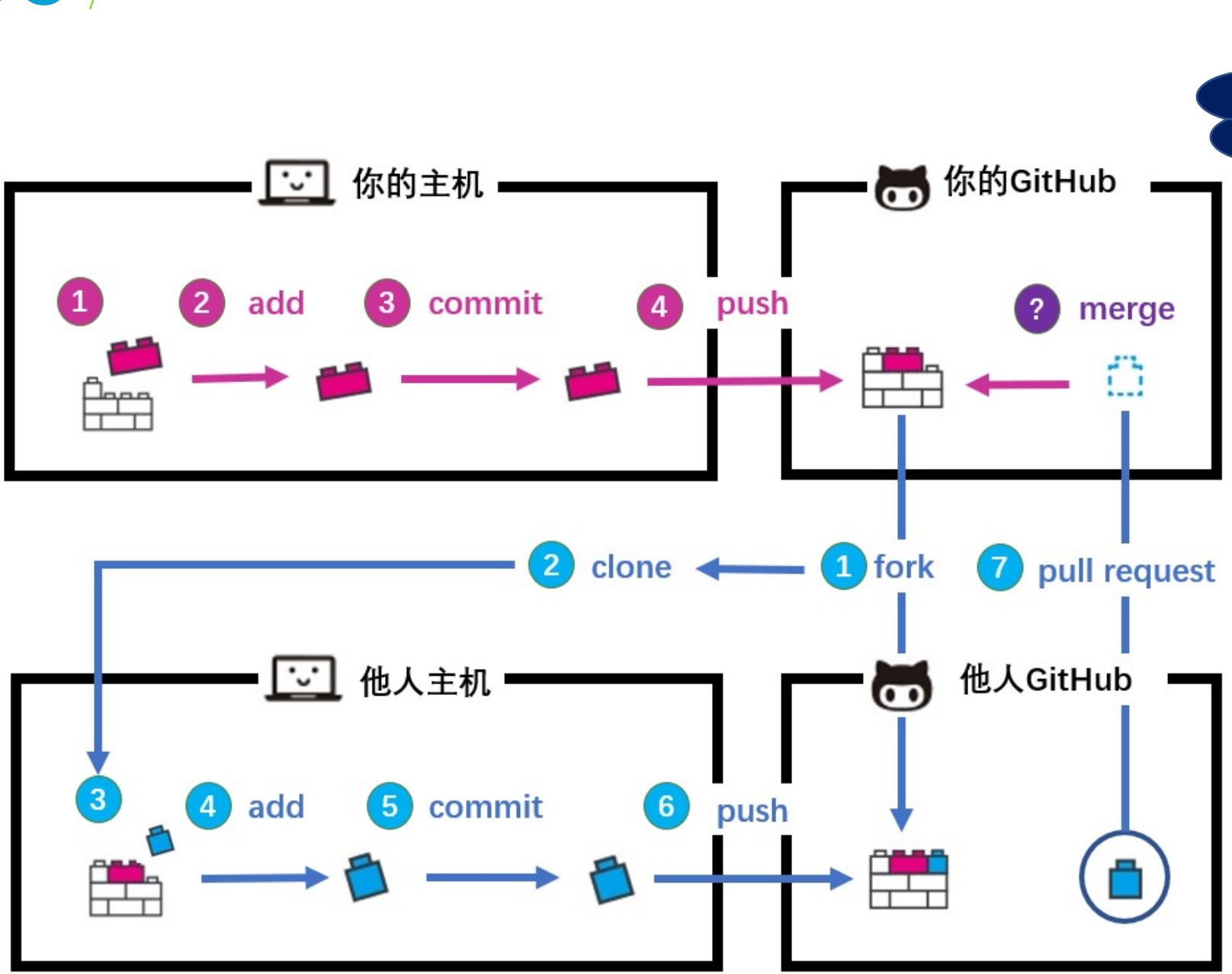
04 全景图


数据服务 (Service) & 商业化（Business）产品
GAReport
开源年报（数据报告）
Allumos
源光闪烁（媒体品牌）
OpenDigger
源力掘醒（分析终端）
Hypertrons-crx
社区看板（可视化插件）
AllSpark
开源社区与开源生态的数据大脑
计算平台
研发工作台（Devas）
算力
数据
工具
工作流
开源数据中台
尖端武器库（Computron）
可视化
算法库
模型库
混合云管理
容器化基础设施
高性能计算基础设施
业务模型 (Model)
成熟度能力模型
可信能力模型
供应链能力模型
度量指标 (AllMetrics)
CHAOSS metrics (Basic)
Domain Metrics (Ext)
评价方法 (Method)
多学科评价方法
专家经验
数据统计
开放数据（Data）
GitHub
Stackoverflow
Social Media
Scholar
Web 2.0
更多
混合智能流程自动化(Hypertrons)



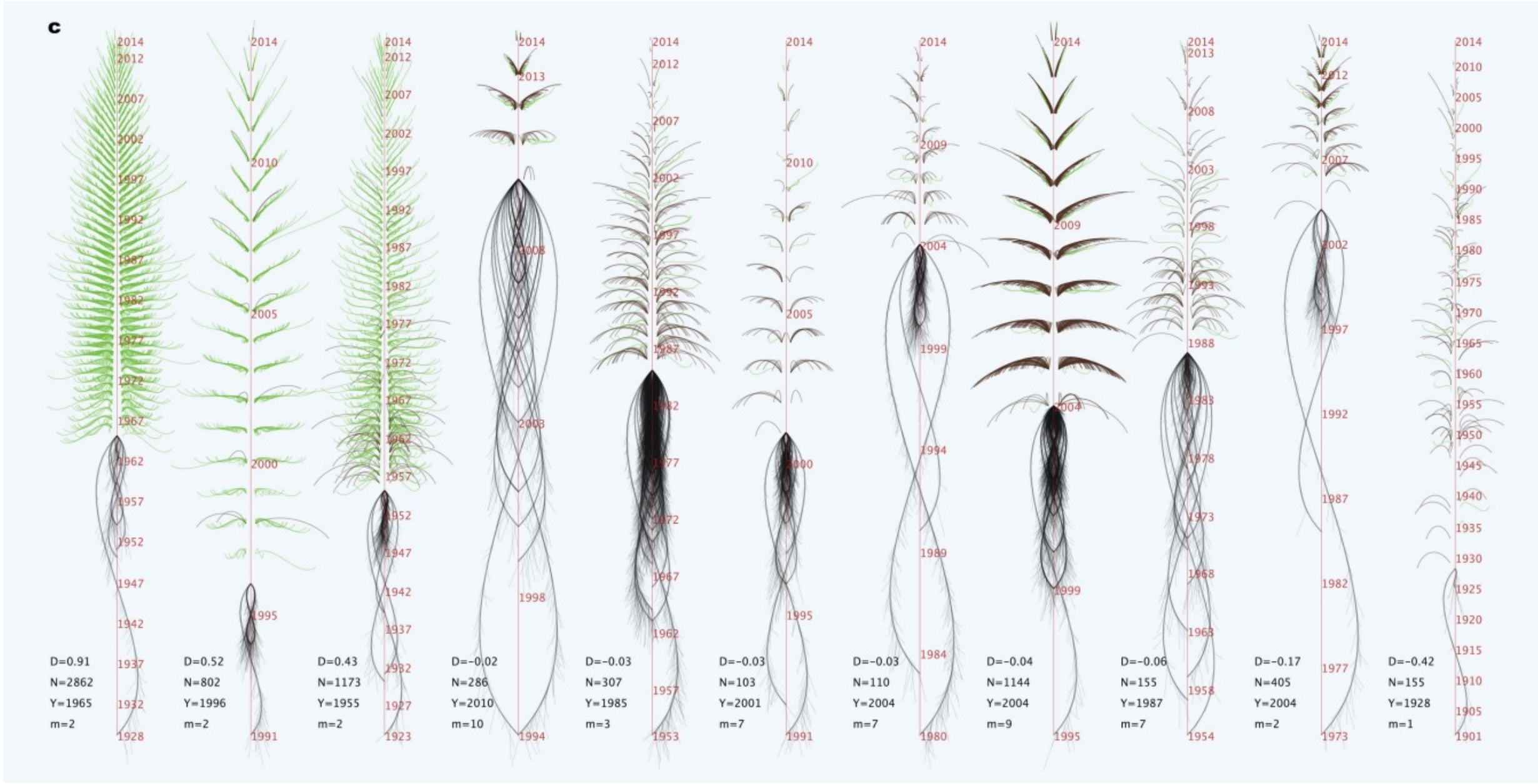
02 开源场景下的数据

Data in Open Source


Social Coding
你的主机
你的GitHub
1
2
add
3
commit
4
push
?
merge
2
clone
1
fork
7
pull request
他人主机
他人GitHub
3
4
add
5
commit
6
push

- 开放
- 分享
- 协作

06 知识的积累和演化

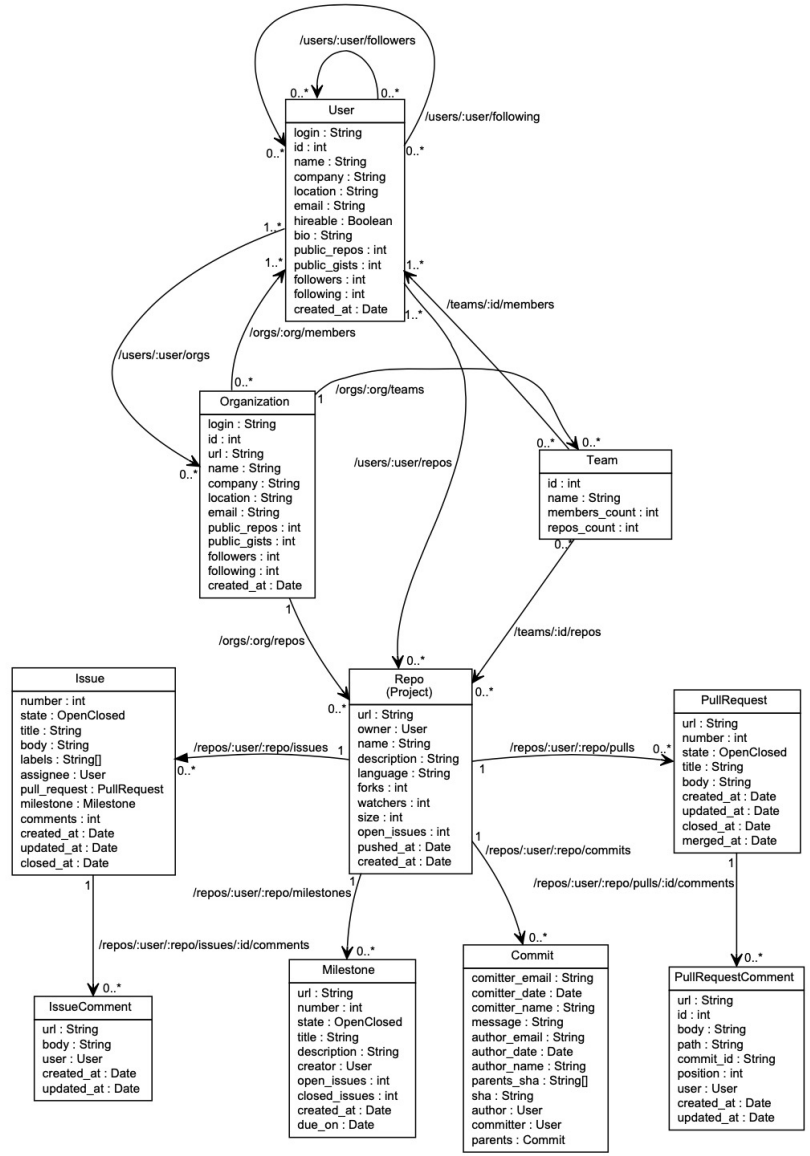

c
D=0.91
N=2862
Y=1965
m=2
D=0.52
N=802
Y=1996
m=2
D=0.43
N=1173
Y=1955
m=2
D=-0.02
N=286
Y=2010
m=10
D=-0.03
N=307
Y=1985
m=3
D=-0.03
N=103
Y=2001
m=7
D=-0.03
N=110
Y=2004
m=7
D=-0.04
N=1144
Y=2004
m=9
D=-0.06
N=155
Y=1987
m=7
D=-0.17
N=405
Y=2004
m=2
D=-0.42
N=155
Y=1928
m=1

07 开放协作平台下的事件流数据

事件数据记录


```
{
  "id": "2489651077",
  "type": "PushEvent",
  "actor": {
    "id": 4070158,
    "login": "caleb-eades",
    "gravata Follow link (ctrl + click)
    "url": "https://api.github.com/users/caleb-eades",
    "avatar_url": "https://avatars.githubusercontent.com/u/4070158?"
  },
  "repo": {
    "id": 20469468,
    "name": "caleb-eades/MinecraftServers",
    "url": "https://api.github.com/repos/caleb-eades/MinecraftServers"
  },
  "payload": {
    "push_id": 536863983,
    "size": 1,
    "distinct_size": 1,
    "ref": "refs/heads/master",
    "head": "6ea9a1f5b0b3c4204272a5fe2587a5ee146c3a49",
    "before": "8e94c95939b8f7db4c085da258698f07ae2b9cf3",
    "commits": [
      {
        "sha": "6ea9a1f5b0b3c4204272a5fe2587a5ee146c3a49",
        "author": {
          "email": "5bbfe2c07a3ef0b22b72711a2edf1c023f6433c5@gmail.com",
          "name": "caleb-eades"
        },
        "message": "Auto Snapshot Server State",
        "distinct": true,
        "url": "https://api.github.com/repos/caleb-eades/MinecraftServers/commits/6ea9a1f5b0b3c4204272a5fe2587a5ee1
      }
    ]
  },
  "public": true,
  "created_at": "2015-01-01T15:00:05Z"
}
```


GitHub元数据



08 数据结构

	name	type	description	payload_type
2	id	String	uique identity of this event instance	All
3	type	String	event instance type,event instances of different event type have different payload from GitHub API	All
4	action	String	action in the payload of event instance	All
5	actor_id	UInt64	GitHub user id of someone who trigger this envent instance	All
6	actor_login	String	GitHub user login of someone who trigger this envent instance	All
7	repo_id	UInt64	the unique identity of a repository where this event was tragered	All
8	repo_name	String	repository name	All
9	org_id	UInt64	organization unique identity of this repository	All
10	org_login	String	login name of this organization	All
11	created_at	DateTime	when this event instance was generated on GitHub	All
12	created_date	Date	same as created_at but in Date format	All
13	issue_id	UInt64	unique identity of this issue on GitHub	Issues, IssueComment, PullRequest, PullRequestReviev
14	issue_number	UInt32	the id of this issue in this repository	Issues, IssueComment, PullRequest, PullRequestReviev
15	issue_title	String	titile of this issue	Issues, IssueComment, PullRequest, PullRequestReviev
16	issue_body	String	body of this issue event	Issues, IssueComment, PullRequest, PullRequestReviev
17	issue_labels	Nested	issue labels of the repository	Issues, IssueComment, PullRequest, PullRequestReviev
18	issue_labels.name	String	issue label name	Issues, IssueComment, PullRequest, PullRequestReviev
19	issue_labels.color	String	issue label color	Issues, IssueComment, PullRequest, PullRequestReviev
20	issue_labels.default	UInt8	whether the label is a default label	Issues, IssueComment, PullRequest, PullRequestReviev
21	issue_labels.description	String	issue label description	Issues, IssueComment, PullRequest, PullRequestReviev
22	issue_author_id	UInt64	GitHub user id of this issue's author	Issues, IssueComment, PullRequest, PullRequestReviev
23	issue_author_login	String	GitHub user login of this issue's author	Issues, IssueComment, PullRequest, PullRequestReviev
24	issue_author_type	String	GitHub user type of this issue's author	Issues, IssueComment, PullRequest, PullRequestReviev

https://github.com/X-lab2017/open-digger/blob/master/docs/assets/data_description.csv

09 例子

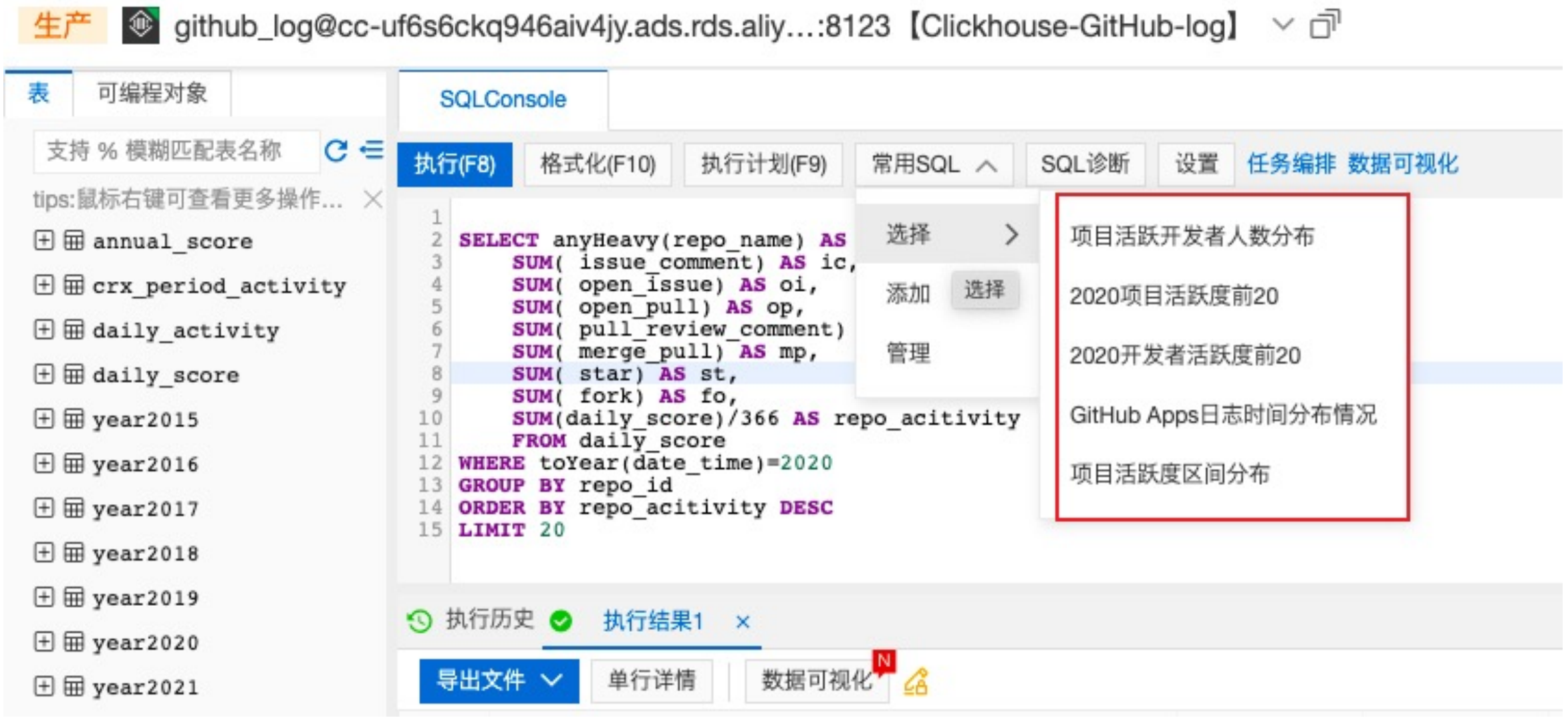
```
repo_activity = '''
SELECT contribute_list.repo_id AS repo_id,contribute_list.repo_name as repo_name, round(sum(sqrt(contribute_list.score)),2) AS repo_activity
FROM
(SELECT
icc.repo_id AS repo_id, icc.repo_name as repo_name, icc.actor_id AS actor_id, {issueCommentWeight}*icc.count+{openIssueWeight}*oic.count+{openPullWeight}*opc.count+{pullReviewWeight}*rcc.count+
{mergePullWeight}*mpc.count AS score
FROM
(SELECT repo_id, repo_name, actor_id, COUNT(*) count FROM {db}.{table} WHERE type='IssueCommentEvent' AND action='created' GROUP BY repo_id,repo_name, actor_id) AS icc
LEFT JOIN
(SELECT repo_id,repo_name, actor_id, COUNT(*) count FROM {db}.{table} WHERE type='IssuesEvent' AND action='opened' GROUP BY repo_id,repo_name, actor_id) AS oic
ON icc.repo_id=oic.repo_id AND icc.actor_id=oic.actor_id
LEFT JOIN
(SELECT repo_id,repo_name, actor_id, COUNT(*) count FROM {db}.{table} WHERE type='PullRequestEvent' AND action='opened' GROUP BY repo_id,repo_name, actor_id) AS opc
ON icc.repo_id=opc.repo_id AND icc.actor_id=opc.actor_id
LEFT JOIN
(SELECT repo_id,repo_name, actor_id, COUNT(*) count FROM {db}.{table} WHERE type='PullRequestReviewCommentEvent' AND action='created' GROUP BY repo_id,repo_name, actor_id) AS rcc
ON icc.repo_id=rcc.repo_id AND icc.actor_id=rcc.actor_id
LEFT JOIN
(SELECT repo_id,repo_name, issue_author_id AS actor_id, COUNT(*) as count FROM {db}.{table} WHERE type='PullRequestEvent' AND action='closed' AND pull_merged=1 GROUP BY repo_id,repo_name,
actor_id) AS mpc
ON icc.repo_id=mpc.repo_id AND icc.actor_id=mpc.actor_id) AS contribute_list
GROUP BY repo_id,repo_name
ORDER BY repo_activity DESC
LIMIT {topN}
'''
```

	repo_id	repo_name	score
0	12888993	home-assistant/core	43123.34
1	31792824	flutter/flutter	35046.32
2	41881900	microsoft/vscode	27198.43
3	12888993	home-assistant/home-assistant	26493.57
4	245491942	superbytes/kllvm	24960.04
...	...	...	...
999995	205732133	Parkhomyuk/angular8-empty	5.10
999996	192707142	dannyongtey/past-year-frontend	5.10
999997	211448897	liuy1994/react-study	5.10
999998	192077709	westkite1201/SeoCalender	5.10
999999	179761294	DovydasTamasauskas/vieta	5.10

```
for year in ["2016","2017","2018",'2019','2020']:
    print (year)
    sql = """
select type, count(*) as count from github_log.year{}
group by type
order by count DESC
""".format(year)
    #print (sql)
    r = sql_excution(sql)
    df_r = DataFrame(r)
    df_r.rename(columns={0:'type',1:year},inplace=True)
    #df_r_20XX[year]=df_r[year]
    df_r_20XX=pd.merge(df_r_20XX,df_r,how='outer',on='type')
```

	type	2015	2016	2017	2018	2019	2020
0	PushEvent	104635229	164860497	219006306	258785448	313361663	434644537
1	CreateEvent	28409767	43826527	57257967	66771401	86949230	135026390
2	IssueCommentEvent	19152405	25752681	29298824	31103806	38258931	54387512
3	WatchEvent	18989330	26165851	32640023	37066146	46106502	49417317
4	PullRequestEvent	10842233	17862510	22931206	27595853	45495842	84960199
5	IssuesEvent	9902675	12806175	14947779	16375510	18835498	25767740
6	ForkEvent	7038647	9538111	11772889	12819951	16454299	18690655
7	DeleteEvent	4477508	7664313	9889711	12193106	17820958	28591620
8	PullRequestReviewCommentEvent	3484645	5590531	7689486	8576786	12011032	16658810
9	GollumEvent	1901527	2290215	2594598	2530760	2520153	2799344
10	CommitCommentEvent	1300700	1045209	918029	820195	1079464	2585913
11	MemberEvent	1175125	1927450	2068373	2174242	2560447	3104383
12	ReleaseEvent	708668	1114942	1518071	1878545	2372869	4199861
13	PublicEvent	203169	281224	409501	493776	1642038	2581325

10 如何访问数据？



https://shimo.im/docs/TvKwRCK3xqhwhJXk/

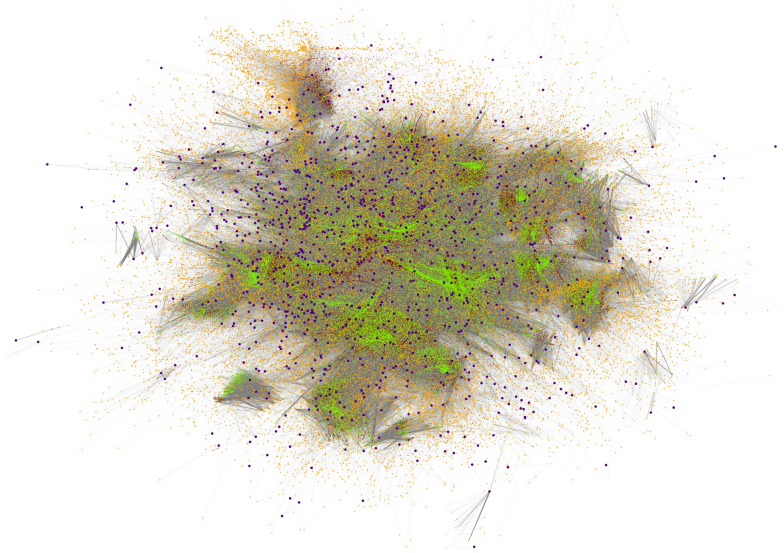
数据基础设施使用演示
Demonstration

X-lab @TianyiChow

开 放 实 验 室

11 数据统计

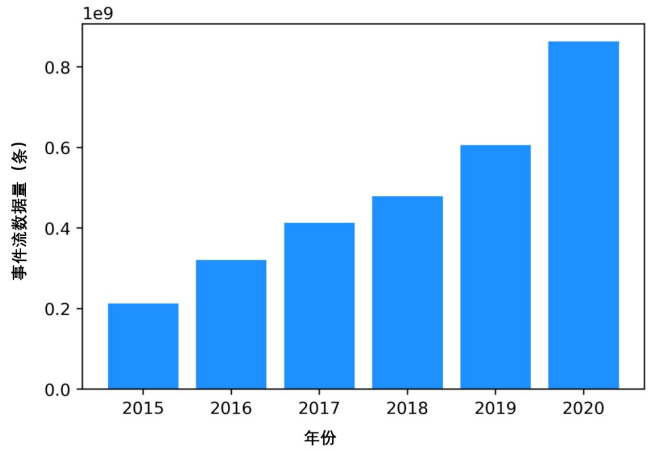
开源社区的异质网络结构



数据聚合计算

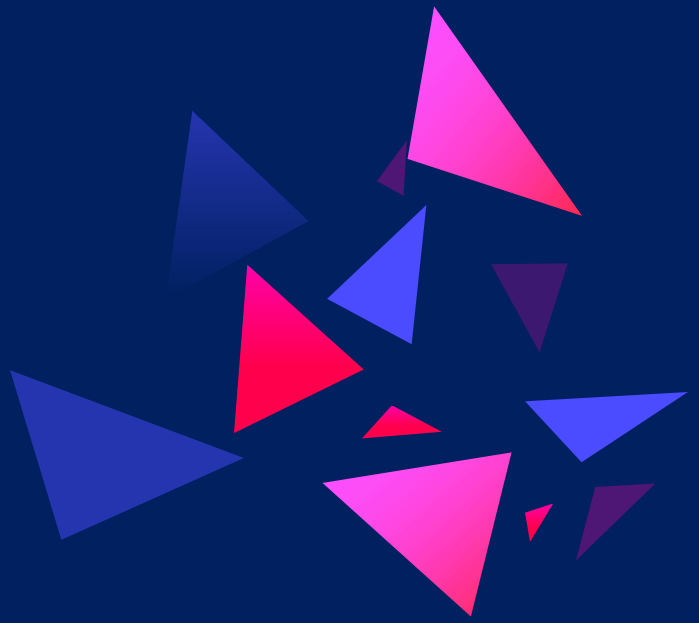
	repo_id	repo_name	score
0	12888993	home-assistant/core	43123.34
1	31792824	flutter/flutter	35046.32
2	41881900	microsoft/vscode	27198.43
3	12888993	home-assistant/home-assistant	26493.57
4	245491942	superbytes/kllvm	24960.04
...	...	...	...
999995	205732133	Parkhomyuk/angular8-empty	5.10
999996	192707142	dannyongtey/past-year-frontend	5.10
999997	211448897	liuy1994/react-study	5.10
999998	192077709	westkite1201/SeoCalender	5.10
999999	179761294	DovydasTamasauskas/vieta	5.10

GitHub事件流数据统计



GitHub事件流数据

事件类型	2016	2017	2018	2019	2020
IssueCommentEvent	25752681	29298824	31103806	38258931	54387512
WatchEvent	26165851	32640023	37066146	46106502	49417317
PullRequestEvent	17862510	22931206	27595853	45495842	84960199
IssuesEvent	12806175	14947779	16375510	18835498	25767740
ForkEvent	9538111	11772889	12819951	16454299	18690655
PullRequestReviewComment	5590531	7689486	8576786	12011032	16658810
CommitCommentEvent	1045209	918029	820195	1079464	2585913



03 异质图网络建模

Modeling Data with HIN

12 开源研究

如何开展研究？

目前数据基础设施已经有了比较好的实现，实验室的主要研究任务强依赖于数据和数据基础设施。需要确定围绕开源数据开展的研究问题发现和定义、可用理论框架介绍、研究方法和解决方案等。

研究课题需要与开源场景的实际应用契合，要有实用价值，这要求对开源业务场景具有较深入的理解。

研究问题

- 问题描述
- 应用场景
- 核心定义

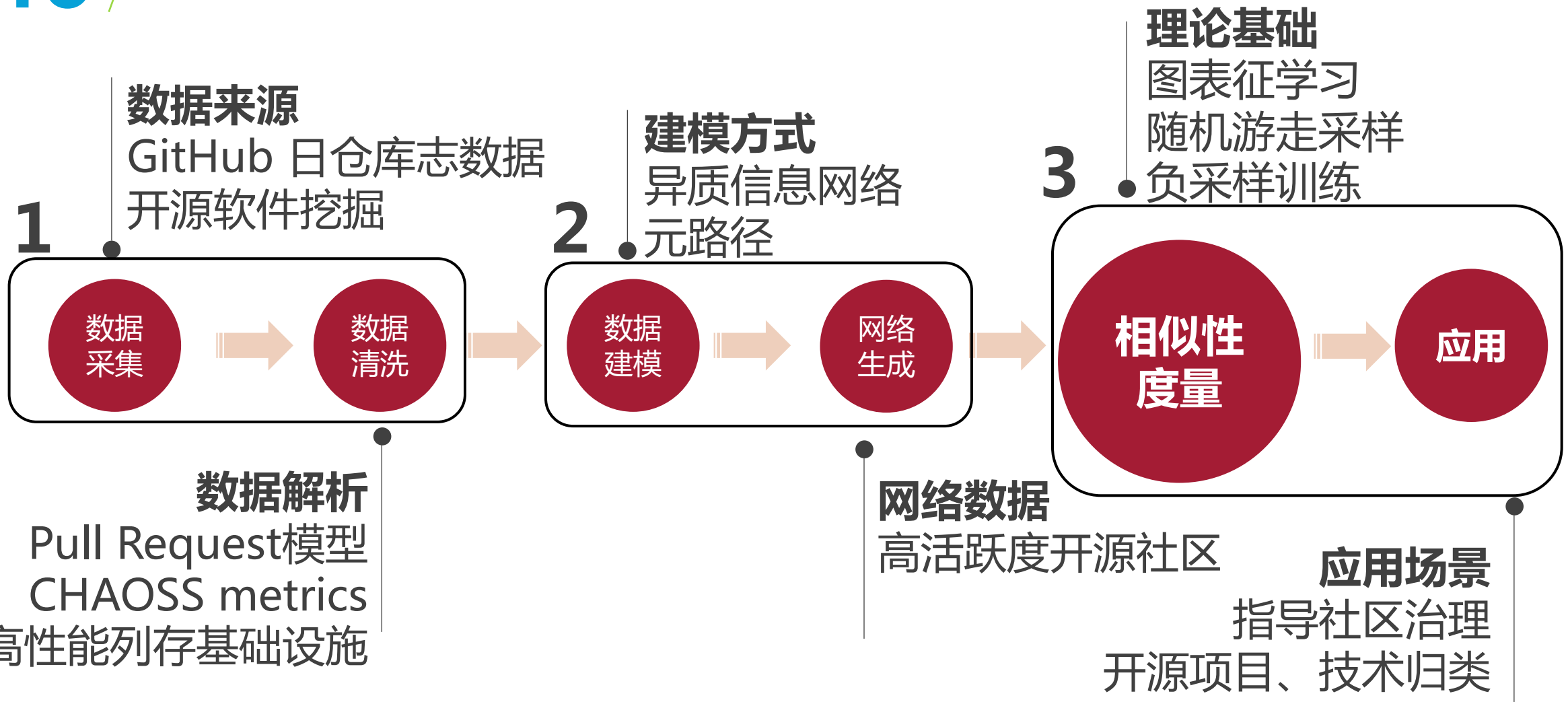
理论框架

- 核心问题拆解
- 寻找涵盖子问题的可用理论

研究方法

- 研究现状
- 问题分析、匹配
- 方法设计
- *评价体系*

如何表示数据中的复杂交互关系？
How to model the “social coding” semantic?

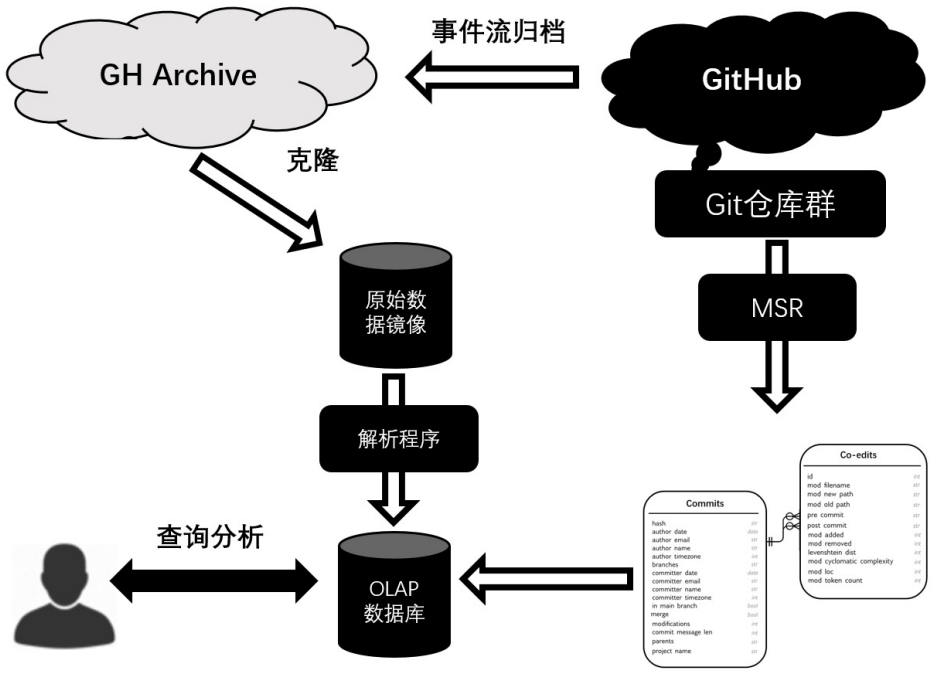

1
数据来源
GitHub 日仓库志数据
开源软件挖掘
2
建模方式
异质信息网络
元路径
3
理论基础
图表征学习
随机游走采样
负采样训练
数据采集
数据清洗
数据建模
网络生成
相似性度量
应用
数据解析
Pull Request模型
CHAOSS metrics
高性能列存基础设施
网络数据
高活跃度开源社区
应用场景
指导社区治理
开源项目、技术归类

定义 1. 同质/异质信息网络 [37][38] 信息网络被定义为一个有向图 $G = (\mathcal{V}, \mathcal{E}, \varphi, \psi)$，其中 $\varphi : \mathcal{V} \to \mathcal{A}$ 为实体类型映射函数，$\psi : \mathcal{E} \to \mathcal{R}$ 为关系类型映射函数。每一个实体 $v \in \mathcal{V}$ 属于实体类型集合 $\mathcal{A} : \varphi(v) \in \mathcal{A}$ 中的某一实体类型，每一条链接 $e \in \mathcal{E}$ 属于关系类型集合 $\mathcal{R} : \psi(e) \in \mathcal{R}$ 中的某一关系类型。如果一个信息网络中的实体类型集合 $|\mathcal{A}| > 1$ 或者关系类型集合 $|\mathcal{R}| > 1$，该信息网络是一个异质信息网络；否则，该信息网络是一个同质信息网络。

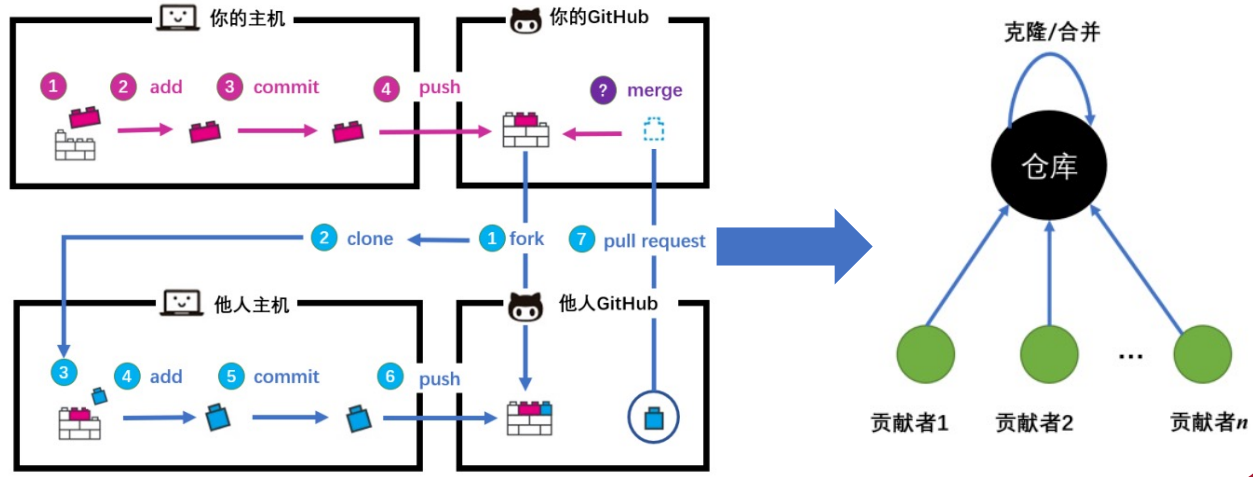
定义 2. 网络模式 网络模式 $\mathcal{S} = (\mathcal{A}, \mathcal{R})$ 是定义在信息网络 $G = (\mathcal{V}, \mathcal{E}, \varphi, \psi)$ 上的元模式，它以关系类型集合 $\mathcal{R}$ 内的链接关系为边，是定义在实体类型集合 $\mathcal{A}$ 上的有向图。网络模式定义了实体对象和关系集合的类型约束，遵循某一特定网络模式 $\mathcal{S}$ 的信息网络 G 被称为该网络模式的**网络实例**。

15 关系表示

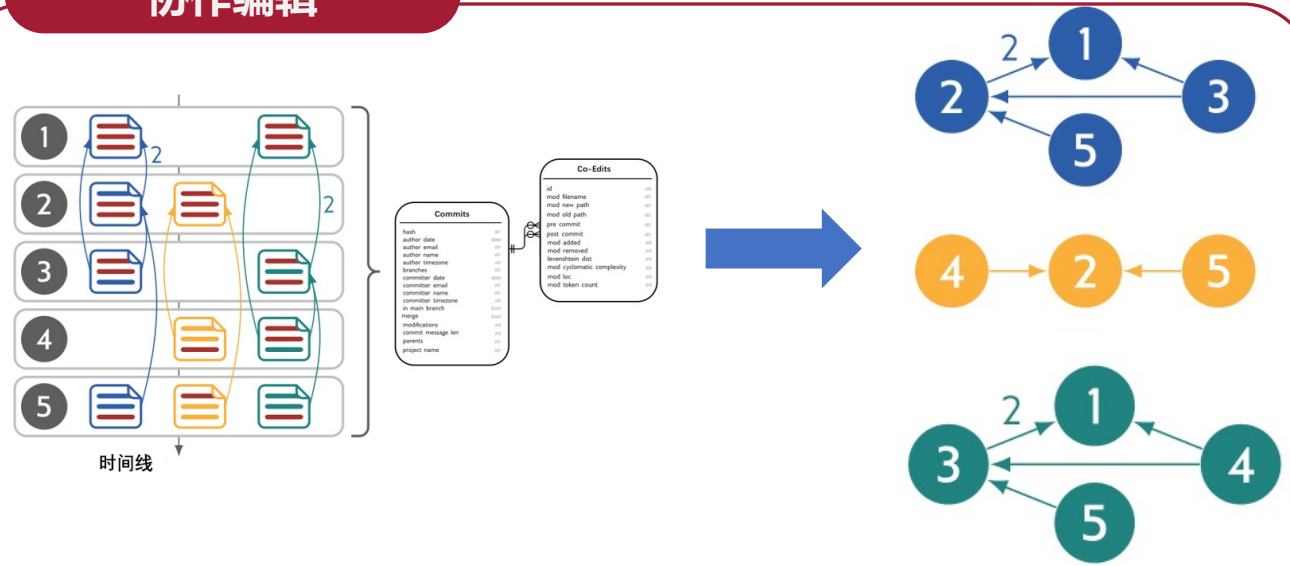
系统结构



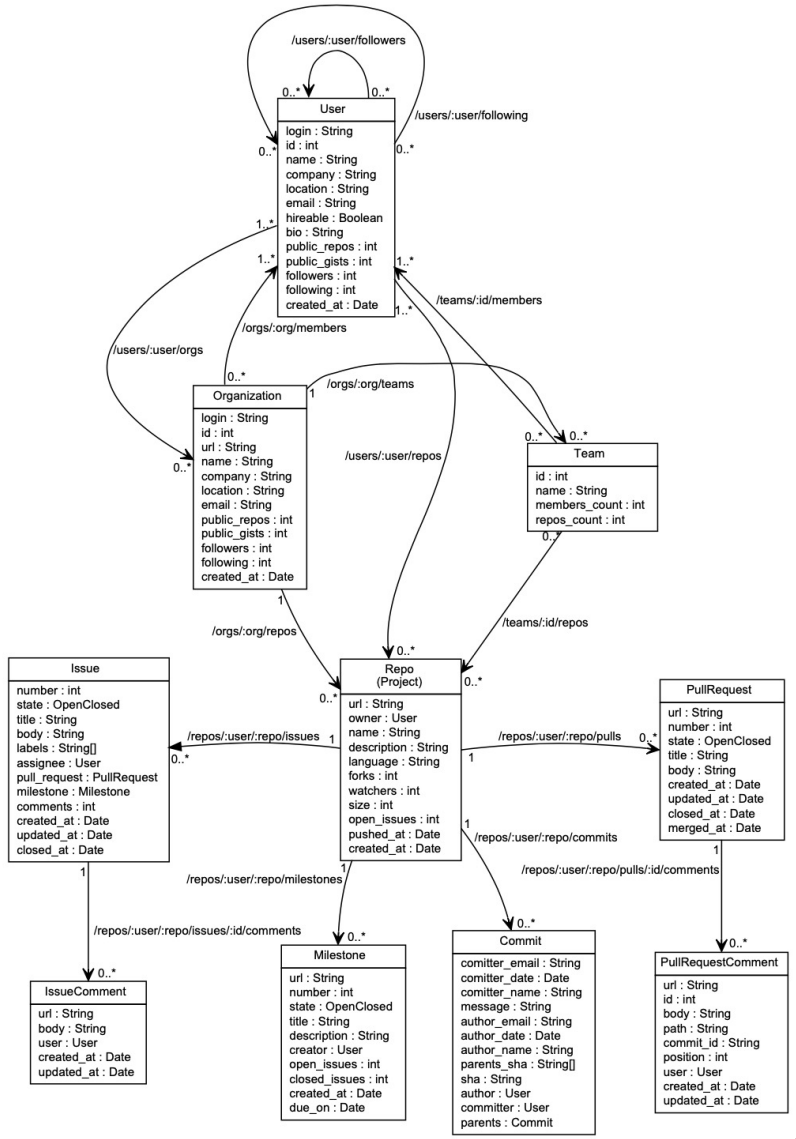
社交语义



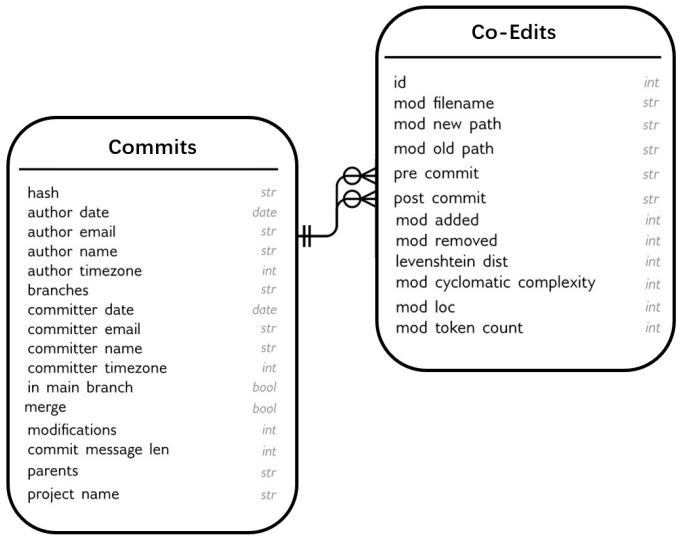
协作编辑



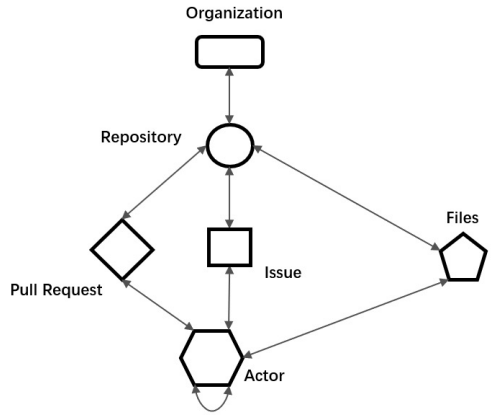
GitHub事件流数据



仓库挖掘



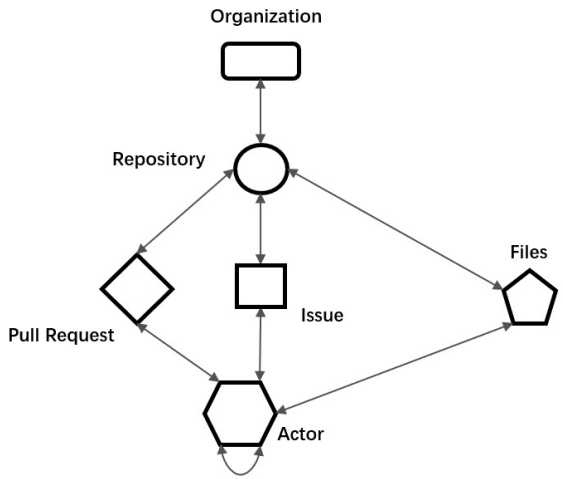
异质网络模式



不同类型节点间的链接关系：$\mathcal{R}_*$

Organization	$\mathcal{R}_{org-repo}$	Repository
Issue	$\mathcal{R}_{issue-repo}$	Repository
Pull Request	$\mathcal{R}_{pr-repo}$	Repository
Files	$\mathcal{R}_{file-repo}$	Repository
Actor	$\mathcal{R}_{actor-issue}$	Issue
Actor	$\mathcal{R}_{actor-pr}$	Pull Request
Actor	$\mathcal{R}_{actor-file}$	Files
Actor	$\mathcal{R}_{actor-actor}$	Actor

异质网络模式



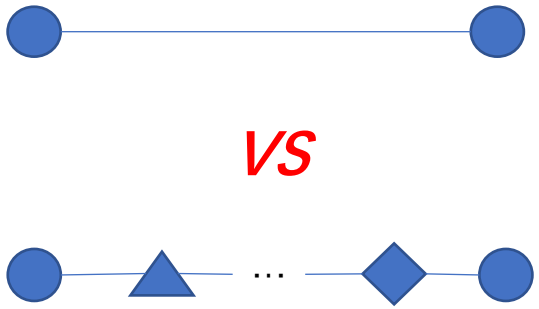
不同类型节点间的链接关系： $\mathcal{R}_*$

Organization	$\mathcal{R}_{org-repo}$	Repository
Issue	$\mathcal{R}_{issue-repo}$	Repository
Pull Request	$\mathcal{R}_{pr-repo}$	Repository
Files	$\mathcal{R}_{file-repo}$	Repository
Actor	$\mathcal{R}_{actor-issue}$	Issue
Actor	$\mathcal{R}_{actor-pr}$	Pull Request
Actor	$\mathcal{R}_{actor-file}$	Files
Actor	$\mathcal{R}_{actor-\text{actor}}$	Actor

开源协作网络模式中的链接关系

链接关系	关系集合
$R_{org-repo}$	(belong_to, own_by)
$R_{issue-repo}$	(belong_to, own_by)
$R_{pr-repo}$	(belong_to, own_by)
$R_{file-repo}$	(belong_to, own_by)
$R_{actor-issue}$	(open, reopened, created, closed, comment)
$R_{actor-pr}$	(open, reopened, created, closed, comment)
$R_{actor-file}$	(added, deleted, created)
$R_{actor-actor}$	(follow, unfollow)

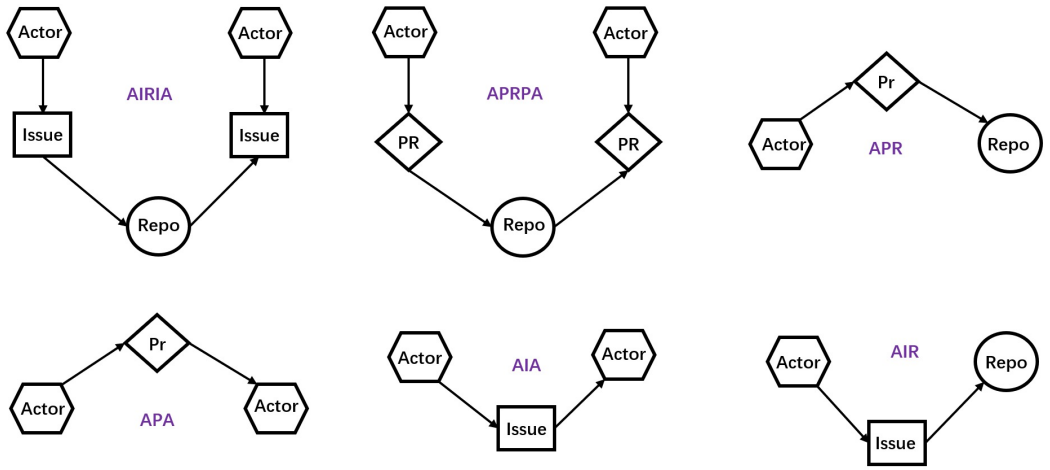
18 基于链接的特征



链接关系：表示实体节点之间的交互关系，构成网络局部结构特征

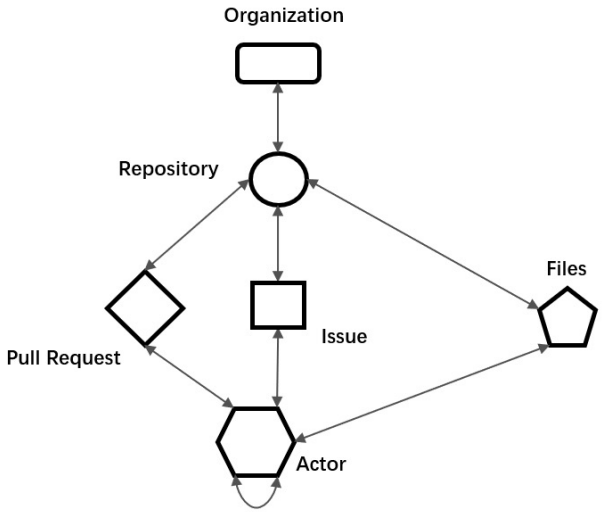
元路径：在异质网络中，复杂的交互关系可以通过不同的元路径表示。

定义 3. 元路径 [33] 元路径 ρ 是定义在网络模式 $\mathcal{S} = (\mathcal{A}, \mathcal{R})$ 上的路径，表示为 $A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \ldots \xrightarrow{R_l} A_{l+1}$，这表示了 $A_1, A_2, \ldots, A_{l+1}$ 之间的复合关系 $R = R_1 \circ R_2 \circ \ldots \circ R_l$，其中 $\circ$ 表示这些关系上的复合运算符。



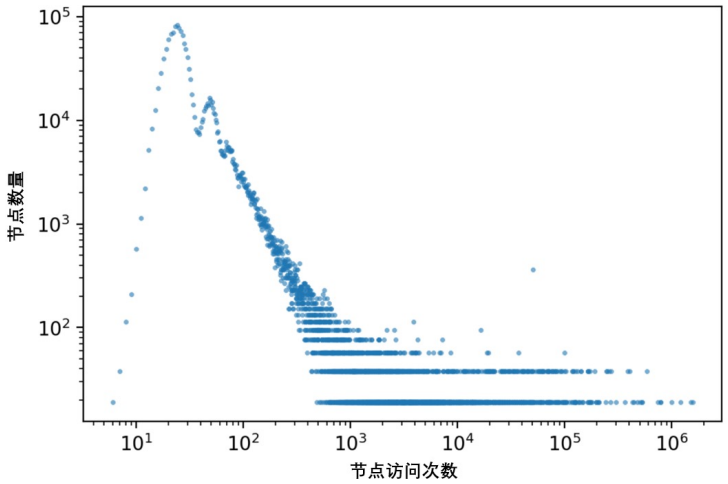
路径实例	元路径	语义解释
Aris-Issue-Spark-Issue-Ted	AIRIA	贡献者在同一个项目下参与讨论
Aris-PR-Spark-Pr-Ted	APRPA	贡献者们对同一个项目提交 Pr
Ted-PR:(New features)-Augur	APR	贡献者向开源项目提交 Pr
Ted-PR:(New features)-Aris	APA	贡献者在同一个 PR 上协作
Ted-Issue:(Bug report)-Aris	AIA	贡献者对同一个问题的讨论
Ted-Issue:(Bug report)-Spark	AIR	贡献者参与 Spark 下的 Issue 讨论

19 网络实例

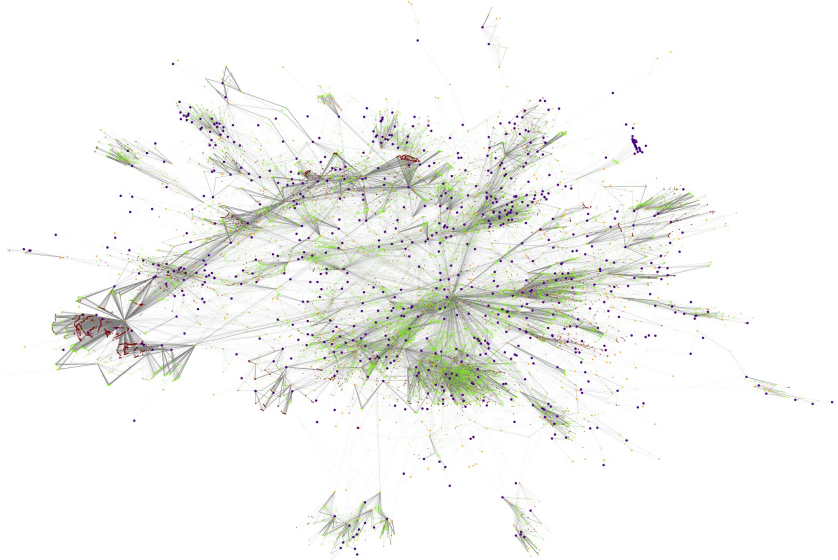


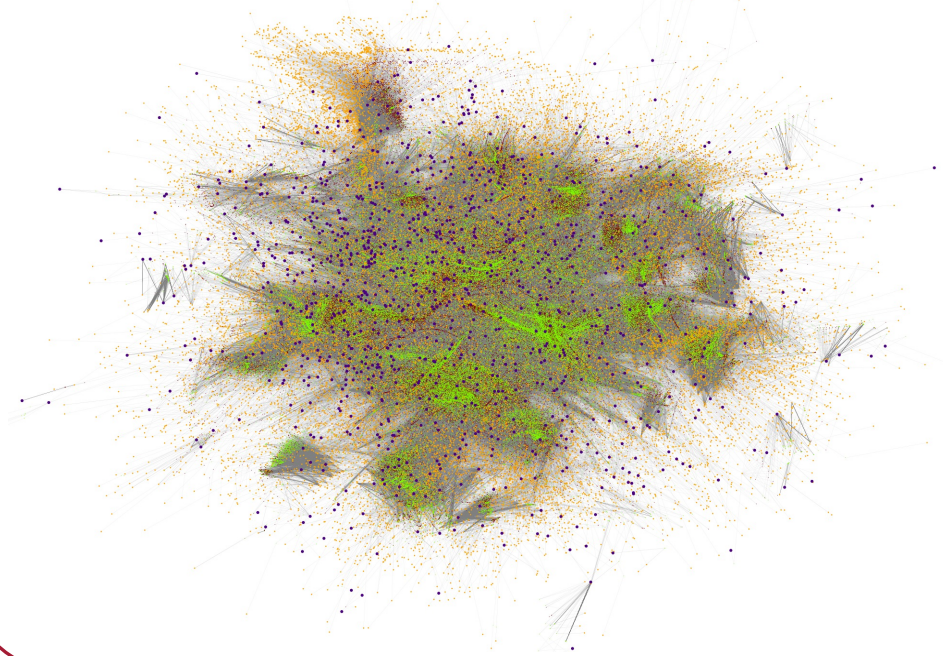
从事件流数据中，选择实体节点和交互关系，可以用网络表示历史交互过程

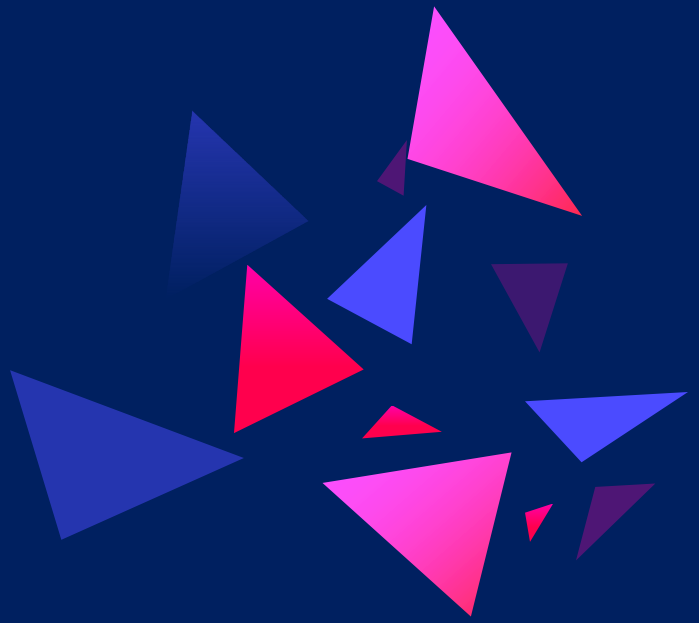
网络呈幂律分布



异质网络实例







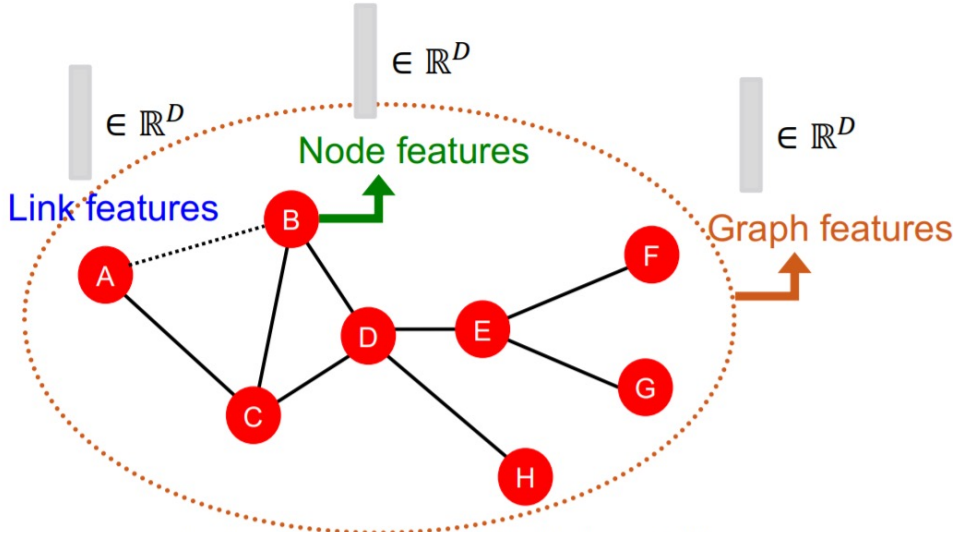
04 图数据挖掘

Graph Data Mining

如何从数据中挖掘出有价值的信息?
How to get insights from data?

图数据的机器学习应用

- 为节点、链接、图设计特征
- 从训练数据中获取这些特征



目标：为一系列目标做预测

设计时的选择：

- 特征：d维向量
- 模块：节点、链接(边)、节点集合、图
- 目标函数：我们想解决哪些问题？

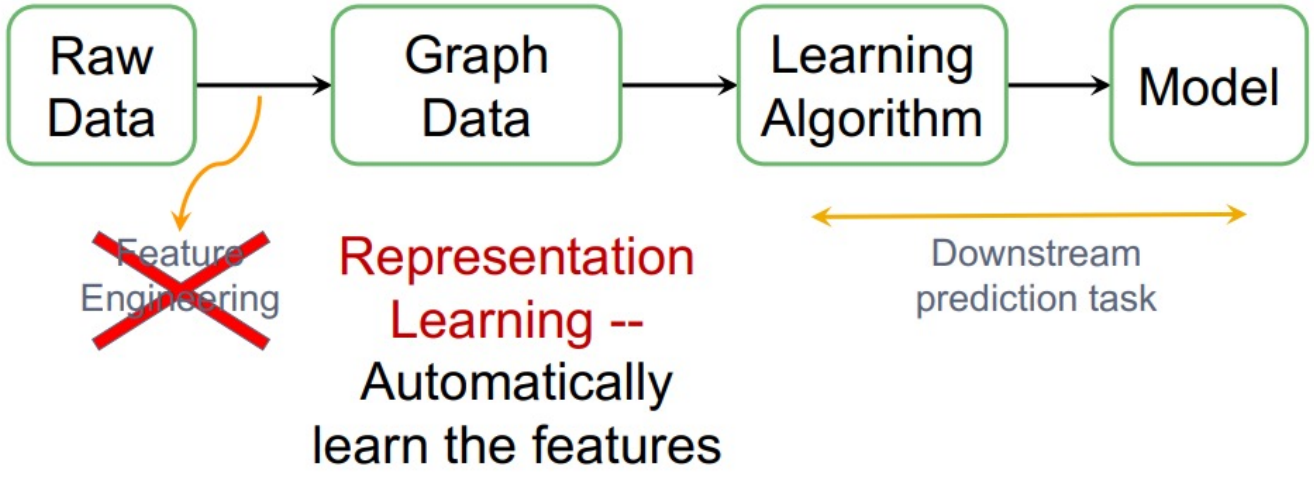
图上的机器学习：

- 输入：G=(V,E)
- 学习一个函数：F: V→$\mathbb{R}$

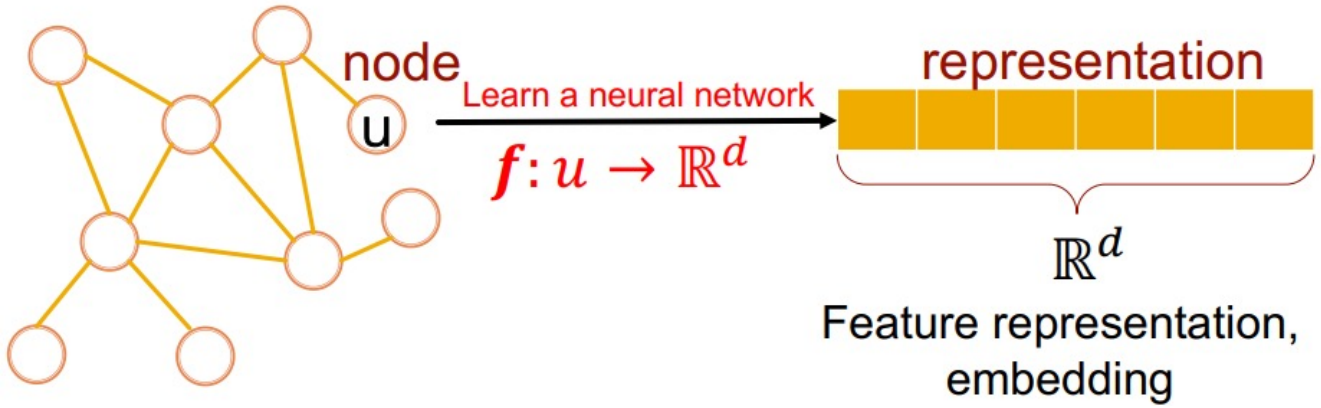
如何学习这个函数是图上的机器学习任务的关键！

图机器学习及应用的整体流程：

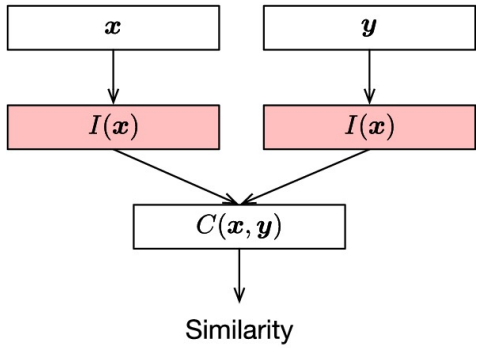
- 数据和特征表示
- 特征抽取
- 下游任务



表征学习：将节点映射到一个d维向量，使得相似节点的向量表示尽可能地接近



22 开源项目相似性



开源项目的相似性

对于开源软件生态中的任意一个开源软件，其特征集合为 $S_f = (s_1, s_2, ..., s_n)$，其中 s_i 为同质属性特征构成的特征子集。现在对于任意两个给定的开源软件项目 A 和 B，存在作用在特征集合 S_f 上的特征映射函数 $F(\cdot)$ 和非负的距离计算函数 $D(\cdot)$，使得 $Similarity(A, B) = D(F(S_f^A), F(S_f^B))$

Spark

是否相似？

Iceberg

Apache Spark

Spark is a unified analytics engine for large-scale data processing. It provides high-level APIs in Scala, Java, Python, and R, and an optimized engine that supports general computation graphs for data analysis. It also supports a rich set of higher-level tools including Spark SQL for SQL and DataFrames, MLlib for machine learning, GraphX for graph processing, and Structured Streaming for stream processing.

https://spark.apache.org/

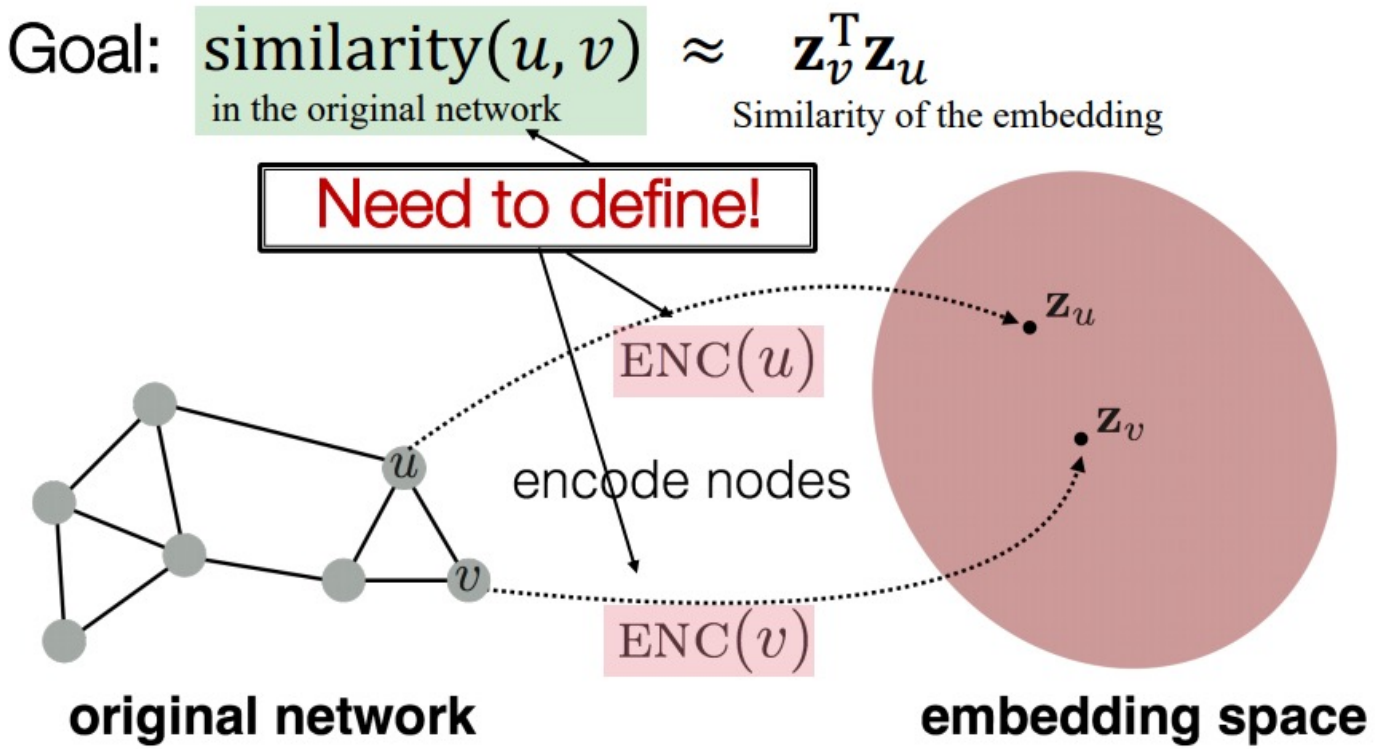
build running | build passing | pyspark coverage 82%

ICEBERG

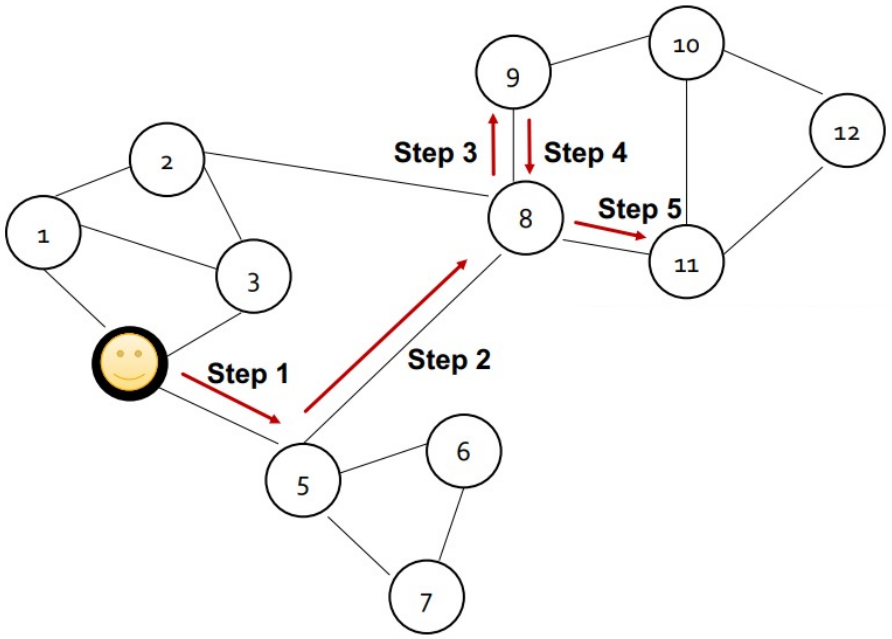
build passing | chat on Slack

Apache Iceberg is a new table format for storing large, slow-moving tabular data. It is designed to improve on the de-facto standard table layout built into Hive, Trino, and Spark.

问题 3. 基于异质信息网络的表征学习 给定一个异质信息网络 $G = (\mathcal{V}, \mathcal{E})$，基于异质信息网络的表征学习任务目标为学习到一个函数 $\Phi : \mathcal{V} \to \mathbb{R}^d$，将节点 $v \in \mathcal{V}$ 映射到一个低维的欧几里得空间，得到一个 d 维的表征向量矩阵 $X \in \mathbb{R}^{|\mathcal{V}| \times d}, d \ll |\mathcal{V}|$，且该表征向量矩阵可以表示异质信息网络的结构和语义特征 [40]。



24 随机游走采样



给定一个网络，从网络的任意节点出发，通过随机游走采样可以得到序列对。节点序列中节点的共现次数反映了节点的相似性。

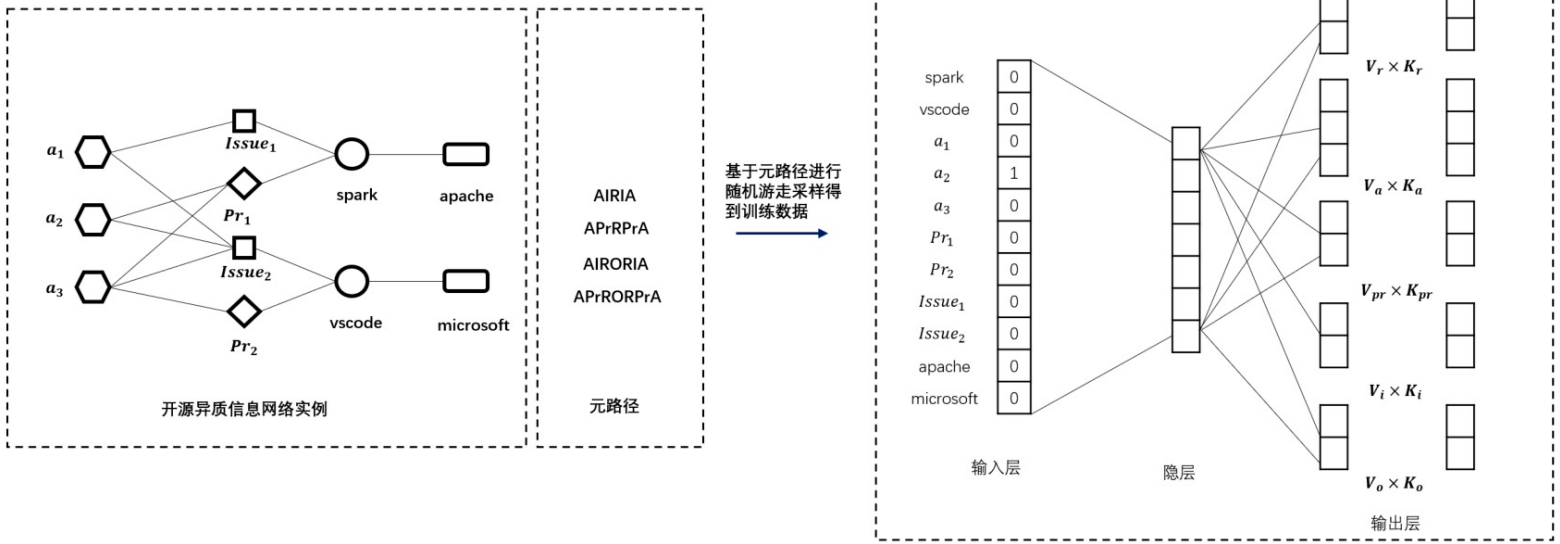
$$P(n_i = x | n_{i-1} = v) = \begin{cases} \frac{\pi_{v,x}}{Z}, & (v,x) \in \mathcal{E} \\ 0, & \text{其他} \end{cases}$$

优化目标函数为：

$$\arg\max_{\theta} \prod_{v \in V} \prod_{c \in \mathcal{N}(v)} P(c|v;\theta)$$

在网络的局部性假设下，相邻节点表示节点之间具有更相似的临近关系，所以随机游走采样结果中节点对的共同出现频率可以很好的表示出节点的相近程度，对序列数据的建模方法可以很好的迁移到该场景下。

基于异质图表征学习的相似性度量



优化目标

$$\arg\max_{\theta} \prod_{v \in V} \prod_{A_t \in \mathcal{A}} \prod_{c_{A_t} \in \mathcal{N}_{A_t}(v)} P(c_{A_t}|v;\theta)$$

$$p(c_{A_t}|v;\theta) = \frac{e^{X_{c_{A_t}} \cdot X_v}}{\sum_{u_{A_t} \in V_{A_t}} e^{X_{u_{A_t}} \cdot X_v}}$$

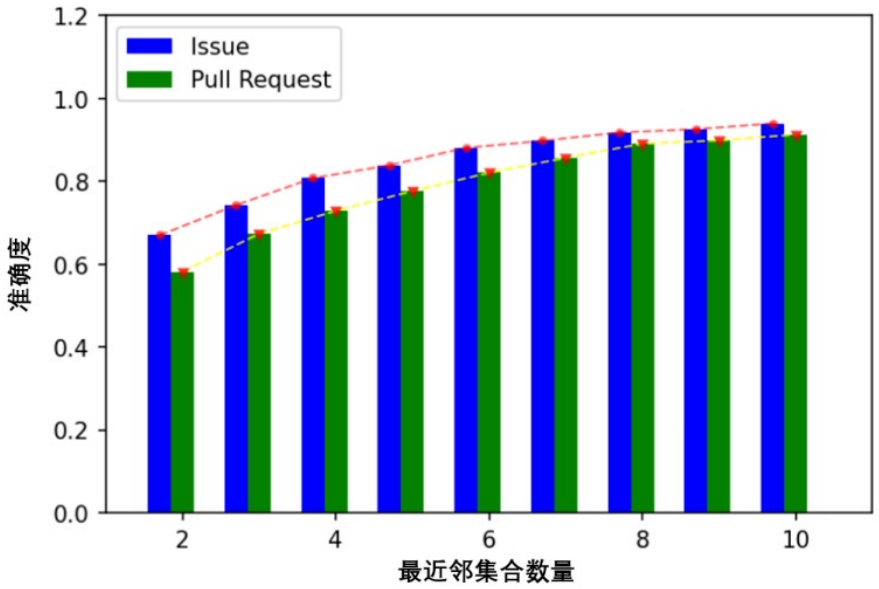
元路径指导的加权随机游走采样

$$P(n_{t+1} = x | n_t = v; \rho) = \begin{cases} \dfrac{w_{v,x}}{\sum_{u \in \mathcal{N}_{A_{t+1}}(v)} w_{uv}}, & (v,x) \in \mathcal{E} \text{ 且 } \phi(x) = A_{t+1} \\ 0, & \text{其他} \end{cases}$$

基于异质图的负采样

$$\mathcal{O} = \log \sigma(X_{c_{A_t}} \cdot X_v) + \sum_{M}^{m=1} \mathbb{E}_{u^m_{A_t} \sim P_{A_t}(u_{A_t})} [\log \sigma(-X_{u^m_{A_t}} \cdot X_v)]$$

26 结果展示



排名	spark	vscode	flink
0	spark	vscode	iceberg
1	arrow	tolerant-php-parser	shardingsphere
2	iceberg	iconv-lite-umd	zeppelin
3	orc	debug-adapter-protocol	calcite
4	arrow-site	node-native-keymap	hudi
5	arrow-testing	monaco-typescript	spark

Apache Arrow

build unknown | codecov 79% | oss-fuzz fuzzing | license Apache 2 | Follow 8.4k

Powering In-Memory Analytics

Apache Arrow is a development platform for in-memory analytics. It contains a set of technologies that enable big data systems to process and move data fast.

ICEBERG

build passing | chat on Slack

Apache Iceberg is a new table format for storing large, slow-moving tabular data. It is designed to improve on the de-facto standard table layout built into Hive, Trino, and Spark.


EAST CHINA NORMAL UNIVERSITY
華東師範大學


DaSE
Data Science
& Engineering

感谢聆听


X-lab
开放实验室