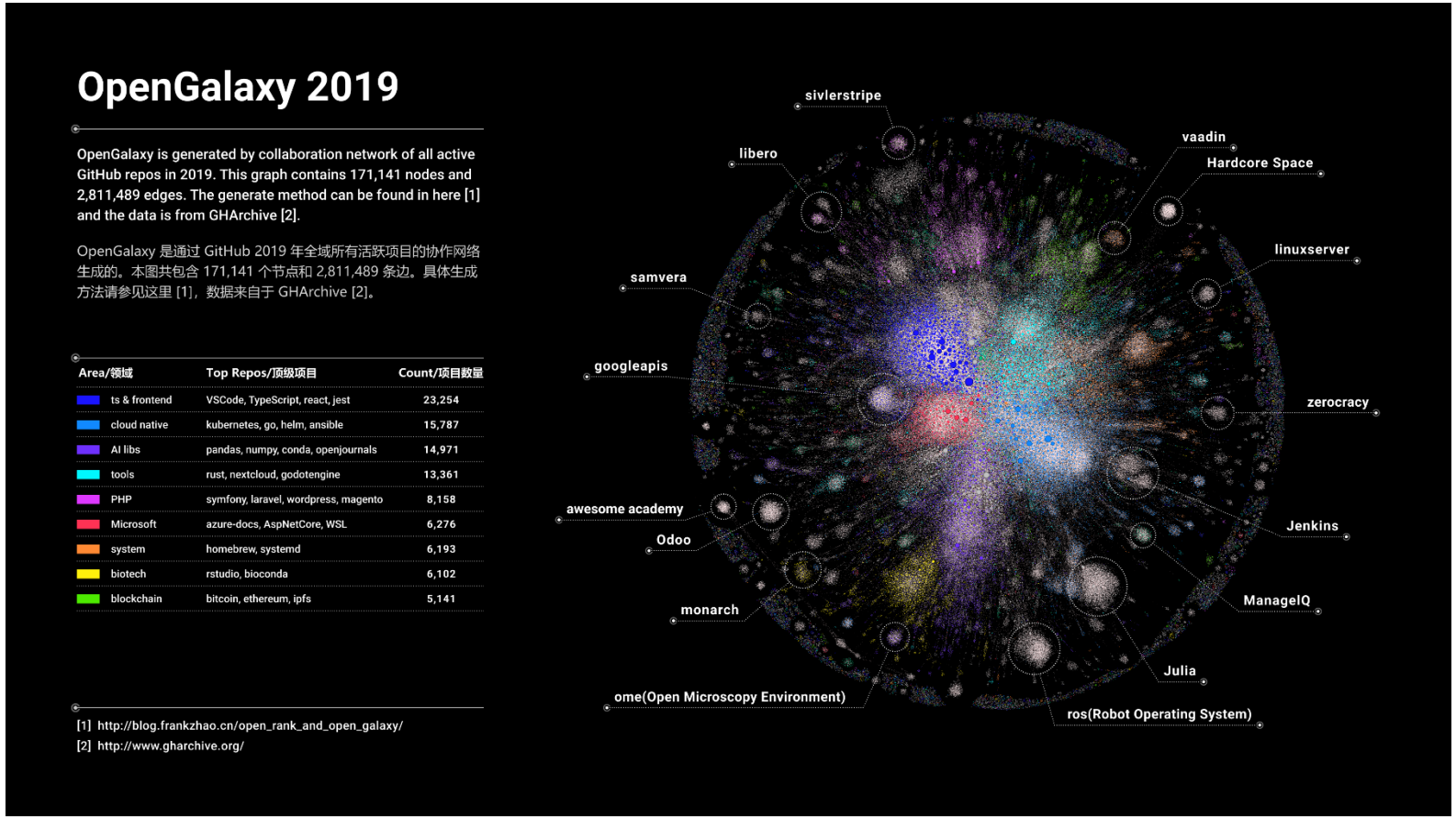

# OpenRank and it’s application

Shengyu Zhao, Xiaoya Xia  Fan Huang

# 3W+1H+2A(application)

# What is OpenRank?

- An open metrics in heterogeneous value evaluation network
- It can approximate  the overall relative importance of a collaboration network in GitHub




https://conf.researchr.org/details/icpc-2024/icpc-2024-tool-demonstration/5/OpenGalaxy-An-interactive-exploration-platform-for-a-visualized-GitHub-Full-Domain-c

# How OpenRank works?

$$PageRank: PR(i) = \frac{1-d}{N} + d \times \sum_{j=1}^{N} \frac{pr(j)}{C_{out}(j)}$$

$$OpenRank: v^{(k)} = ASv^{(k-1)} + (E - A)v^{(0)}$$

- Data field : open source software / web network
- Paradigm: network iteration +initial value
- Network : heterogeneous (developer and repository)

# Where is the data from? OpenDigger

- Information service interface:
- GitHub collaboration behavior logs data : 5.8 billion
- NPM package data : 2.47 million
- Go language module data: 1.02 million
- PyPI package data: 0.449 million
- Indicator: openrank value , activity
- data storage： clickhouse,Neo4j

https://open-digger.x-lab.info/docs/user_docs/intro

# Why we need openrank?

- Precise operation and behavior guidance
- efficiency and equity

- Ecological sustainability
- Temporal : growth potential

# Motivating Open Source Collaborations Through Social Network Evaluation: A Gamification Practice from Alibaba

- OpenRank leaderboard(ORL)：
- A collaboration network
- Motivate collaboration in social network perspective
- Evaluation : case study which span 12-month period

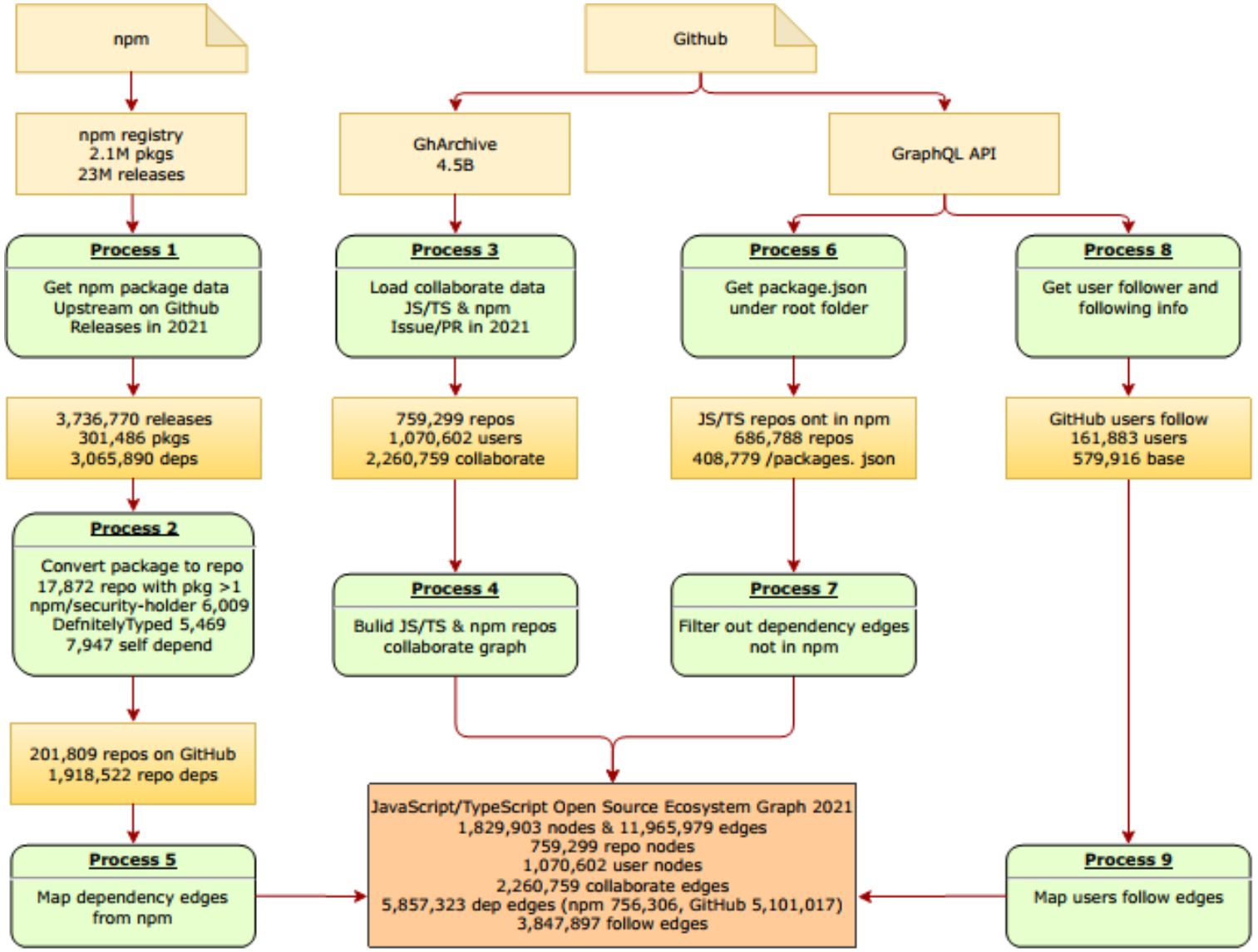
https://conf.researchr.org/details/icse-2024/icse-2024-software-engineering-in-practice/23/Motivating-Open-Source-Collaborations-Through-Social-Network-Evaluation-A-Gamificati

# 2application: NPM and Global

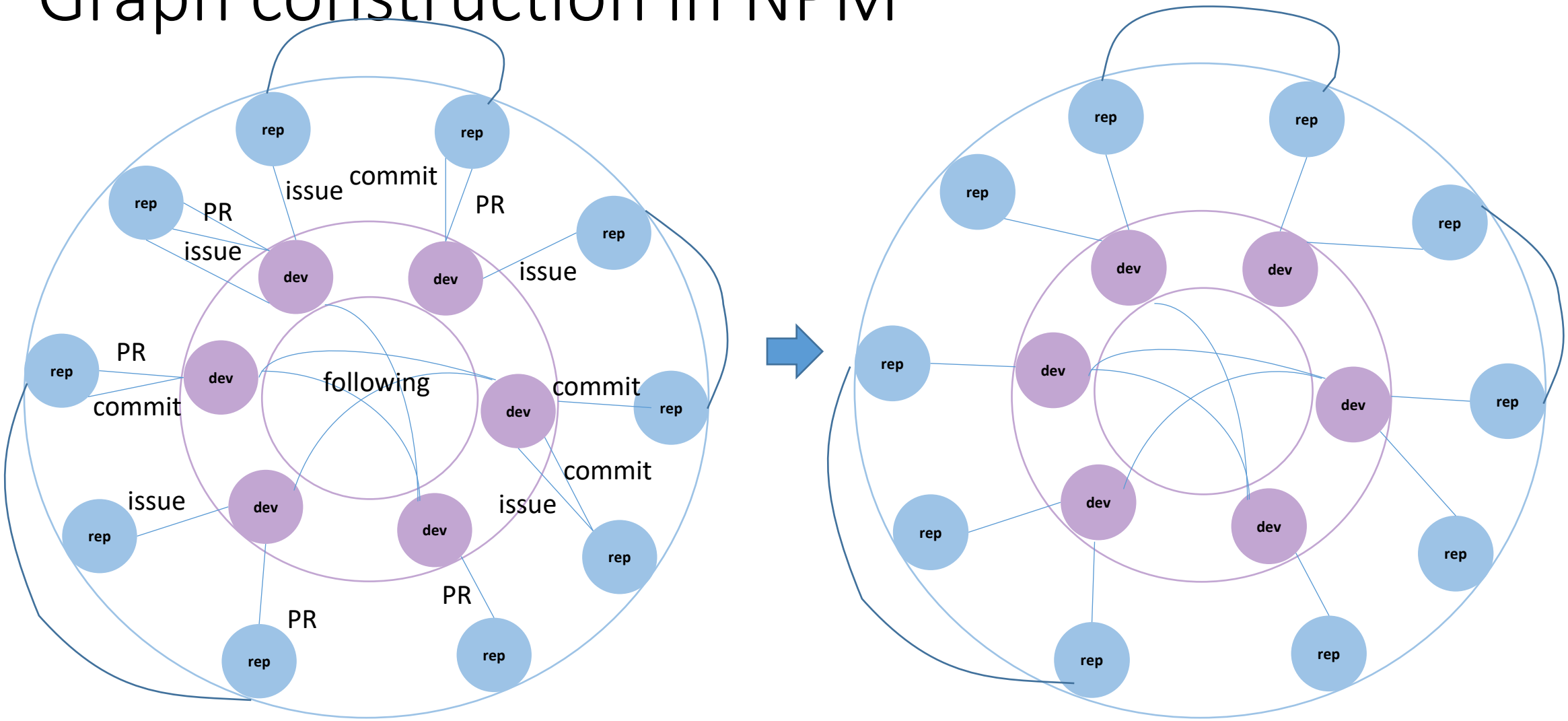
# Evaluating Open Source Project Influence based on NPM Ecology Dataset

Data collection:

- GitHub event log
- Metadata of NPM central registry
- GitHub GraphQL



# Graph construction in NPM




social collaboration and repository dependency network

# OpenRank in NPM

- Initial : activity
- Iteration: $OpenRank: v^{(k)} = ASv^{(k-1)} + (E - A)v^{(0)}$

$$openrank(rep) = \left( \sum_{dep \epsilon N_{rep}} \frac{OpenRank(dep) \cdot W_{dep2repo}}{C(dep)} \cdot repoActivity + \sum_{r \epsilon N_{rep}} \frac{OpenRank(r) \cdot W_{rep2rep}}{C(r)} \cdot repoDependency \right)$$

$$\cdot (1 - repoAttribute) + repoAttribute \cdot initial$$

$$repoActicity + repoDependency = 1$$

# Parameters

- The weight between edges

**IssueComment**

**OpenIssue**

**OpenPullRequest**

**ReviewComment**

**repoActivity**

**repoDenpendency**

**repoAttibute**

**userActivity**

**userFollow**

**userAttribute**

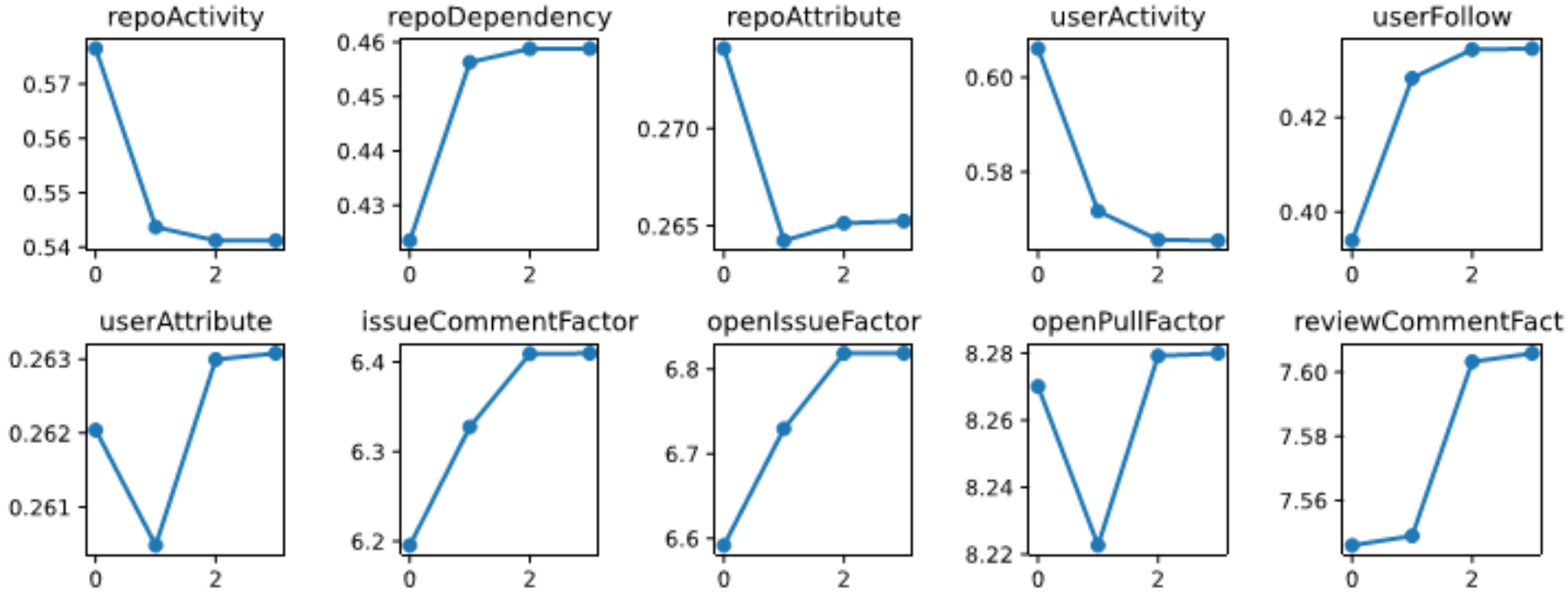
# Parameter iteration

- Take openissue as an example:
- Step 1: Questionnaire : Active developers(number : n)
- give a score on how important of openissue ( $s_1, s_2, \cdots s_n$ )
- Step 2 : $$w_{issue} = \frac{\sum_{i=1}^{n} s_i}{n}$$
- Step 3 : OpenRank iteration
- Setp 4 : $$w_{issue} = \frac{\sum_{i=1}^{n} OpenRank(i) \cdot s_i}{\sum_{i=1}^{n} OpenRank(i)}$$

$$W_{dev2repo} = \sum_{i \epsilon type} num_i \cdot w_i$$

# Parameter iteration

- OpenPullRequest> ReviewComment>OpenIssue>IssueComment
- Developer contribution is approximately equal to repository contribution



# OpenRank in NPM

- Social collaboration
- Repository dependency

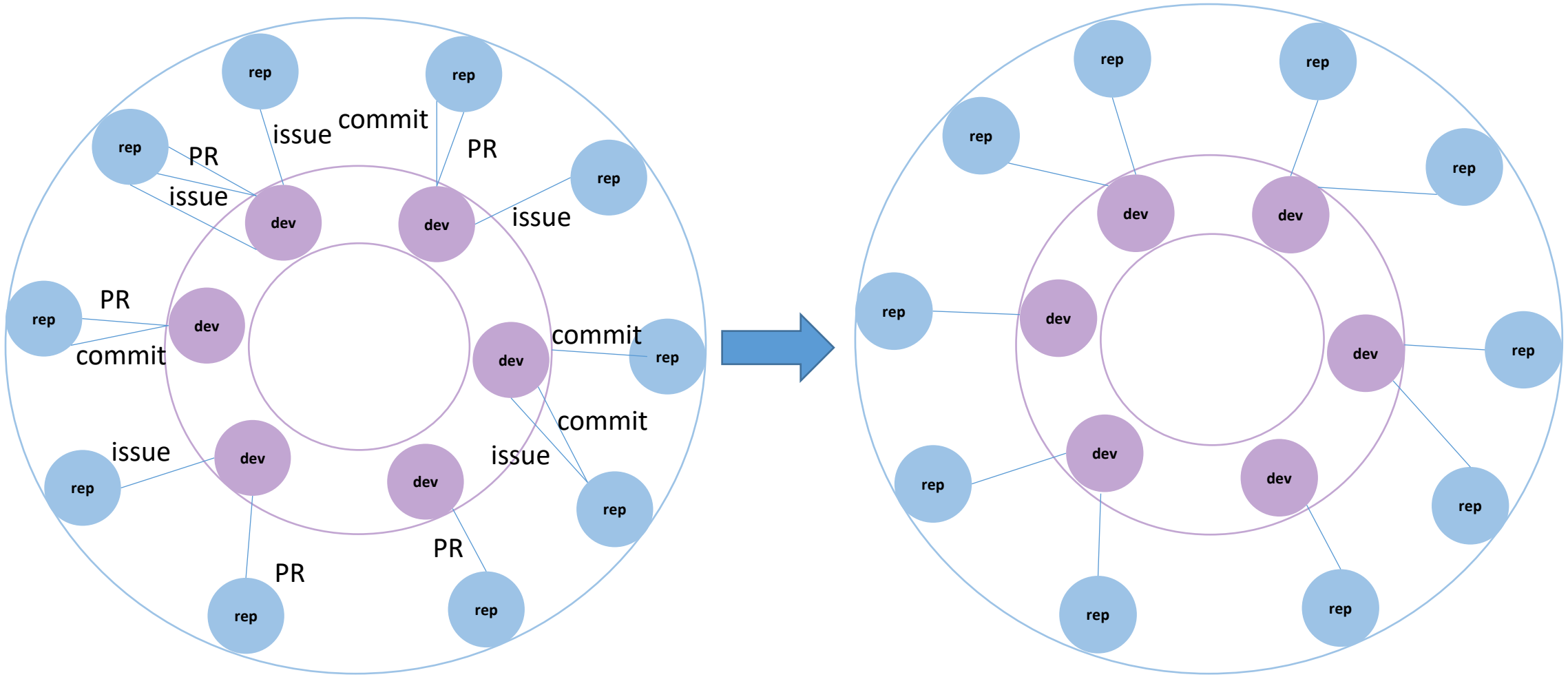
**Table 2: tradition method ranking value**

| index | name | newissues | merge PR | commits |
|---|---|---|---|---|
| 1 | microsoft/vscode | 23152 | 173 | 39152 |
| 2 | facebook/react | 1101 | 761 | 969 |
| 3 | facebook/ create-react-app | 785 | 63 | 228 |
| 4 | facebook/jest | 661 | 295 | 458 |
| 5 | prettier/prettier | 613 | 1095 | 5064 |
| 6 | eslint/eslint | 578 | 327 | 1171 |
| 7 | babel/babel | 506 | 639 | 1375 |
| 8 | DefinitelyTyped/ DefinitelyTyped | 454 | 4908 | 4937 |
| 9 | expressjs/express | 192 | 4 | 85 |
| 10 | mochajs/mocha | 153 | 66 | 331 |

**Table 1: OpenRank value**

| index | name | OpenRank |
|---|---|---|
| 1 | DefinitelyTyped/DefinitelyTyped | 2970.180693 |
| 2 | eslint/eslint | 2429.046561 |
| 3 | facebook/react | 1992.727977 |
| 4 | microsoft/vscode | 1792.456317 |
| 5 | mochajs/mocha | 1509.175633 |
| 6 | prettier/prettier | 1480.113878 |
| 7 | babel/babel | 1459.010543 |
| 8 | facebook/jest | 1423.999665 |
| 9 | facebook/create-react-app | 893.5299907 |
| 10 | expressjs/express | 866.6468937 |
| 11 | webpack/webpack | 854.8518659 |
| 12 | typescript-eslint/typescript-eslint | 813.422885 |
| 13 | benmosher/eslint-plugin-import | 808.8084172 |
| 14 | typicode/husky | 764.4078974 |
| 15 | education/GitHubGraduation-2021 | 714.5837127 |
| 16 | axios/axios | 706.3648035 |
| 17 | prettier/eslint-config-prettier | 664.3103209 |
| 18 | chaijs/chai | 649.6974667 |
| 19 | vercel/next.js | 645.9703493 |
| 20 | lodash/lodash | 631.051654 |

# Data construction in global GitHub



# Global OpenRank

- Initial value : repository(1) and developer(log(activity)/openrank last month)
- Iteration: social collaboration

$$OpenRank:\ v^{(k)} = ASv^{(k-1)} + (E - A)v^{(0)}$$

$$OpenRank(rep) = (1 - repoDependency) \cdot inital +$$

$$repoDpendency \cdot \left( \sum_{i \epsilon N_{dev}} \frac{OpenRank(i) \cdot w_{dev2repo}}{C(i)} \right)$$

# parameters

$$W = \sum_{i \epsilon event} num_i \cdot w_i \ ,$$

$$event = \{Issue\ Comment,\ PR\ \text{Re}view, Close\ issue,\ Close\ PR,\ Open\ Issue,\ Open\ PR\}$$

$$W_{repo2dev} = \frac{W}{acitvity_{rep}}, W_{dev2repo} = \frac{W}{activity_{dev}}$$

$$repoDependency: 0.3$$

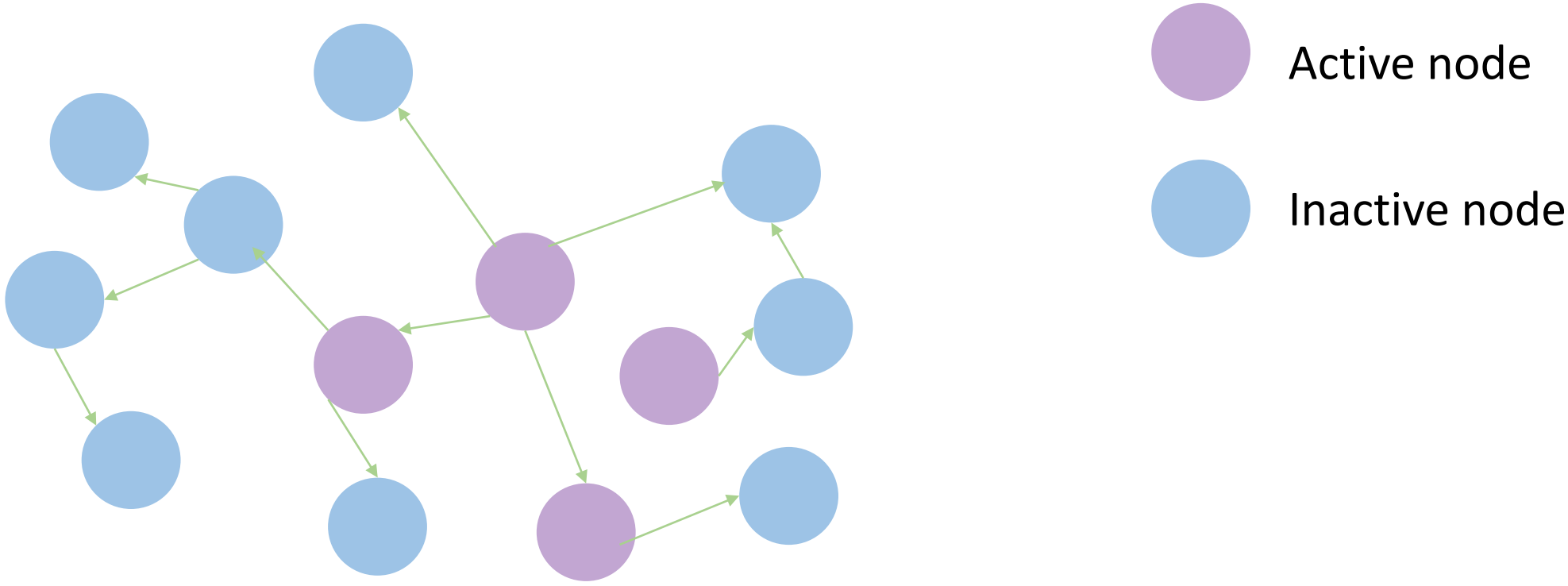
$$userDependency: 0.5$$

**Table 2: AHP Evaluation Matrix and Results.**

| Behavior | Issue/PR Comment | PR Review | Close Issue | Close PR | Open Issue | Open PR | Eigenvector | Weight(%) |
|---|---|---|---|---|---|---|---|---|
| Issue Comment | 1 | 0.5 | 0.5 | 0.333 | 0.25 | 0.2 | 0.401 | 5.252 |
| PR Review | 2 | 1 | 0.5 | 0.5 | 0.333 | 0.2 | 0.567 | 7.427 |
| Close Issue | 2 | 2 | 1 | 0.5 | 0.333 | 0.25 | 0.742 | 9.712 |
| Close PR | 3 | 2 | 2 | 1 | 0.5 | 0.333 | 1.222 | 14.695 |
| Open Issue | 4 | 3 | 3 | 2 | 1 | 0.333 | 1.698 | 22.235 |
| Open PR | 5 | 5 | 4 | 3 | 3 | 1 | 3.107 | 40.679 |

# Evaluation OpenRank: spread algorithm in maximum influence

- Influence maximization



# Evaluation OpenRank: spread algorithm in maximum influence

- Spread algorithm in influence maximization:

  1.independent cascade (IC)

**Algorithm 1** Independent Cascade Model

```
1:  Graph G = (V, E)                    ▷ Graph with set of vertices V and edges E
2:  Initial set of active nodes S ⊆ V
3:  Queue Q ← S                         ▷ Initialize queue with active nodes
4:  while Q is not empty do
5:      v ← dequeue(Q)                  ▷ Get a node from the queue
6:      for each neighbor u of v do
7:          if u is not active then
8:              Activate u with probability p_vu
9:              if u is activated then
10:                 Add u to Q
11:             end if
12:         end if
13:     end for
14: end while
```

# Evaluation OpenRank: spread algorithm in maximum influence

- Spread algorithm in influence maximization:
- 2.Linear Threshold(LT)

**Algorithm 1** Linear Threshold Model

Graph $G = (V, E)$ ▷ Graph with set of vertices $V$ and edges $E$
Initial set of activated nodes $A \subseteq V$
Threshold $\theta_i$ for each node $v_i \in V$
**while** there are nodes $v \in V$ such that $v \notin A$ and $\sum_{u \in N(v)} w_{uv} \cdot indicator(u \in A) \geq \theta_v$ **do**
    Choose a node $v$ that satisfies the condition
    Add $v$ to $A$
**end while**

# Evaluation OpenRank: spread algorithm in maximum influence

- Spread algorithm in influence maximization:

- 3.SIR/SIS(non-progressive : activated nodes can be de-activated )

**Algorithm 1** SIR Model

```
Graph G = (V, E)                          ▷ Graph with set of vertices V and edges E
Initial set of infectious nodes I_0 ⊆ V
Transmission rate β
Recovery rate γ
Time step t
for t ← 1 to T do                         ▷ Iterate over time steps
    for v ∈ V do                          ▷ Loop over all nodes
        if v is infectious at time t − 1 then
            for each neighbor u of v do
                if u is susceptible at time t − 1 then
                    Infect u with probability β
                end if
            end for
            Recover v with probability γ
        end if
    end for
end for
```

Thank you!