# Towards Team Formation in Software Development: A Case Study of Moodle

The 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2020)
25 June 2020 Virtual Conference Hosted by College of Computing, Prince of Songkla University

Mr. Noppadol Assavakamhaenghan
Mr. Ponlakit Suwanworaboon
Miss Waralee Tanaphantaruk
Assistant Professor Dr.Suppawong Tuarob
Dr. Morakot Choetkiertikul

# Outline

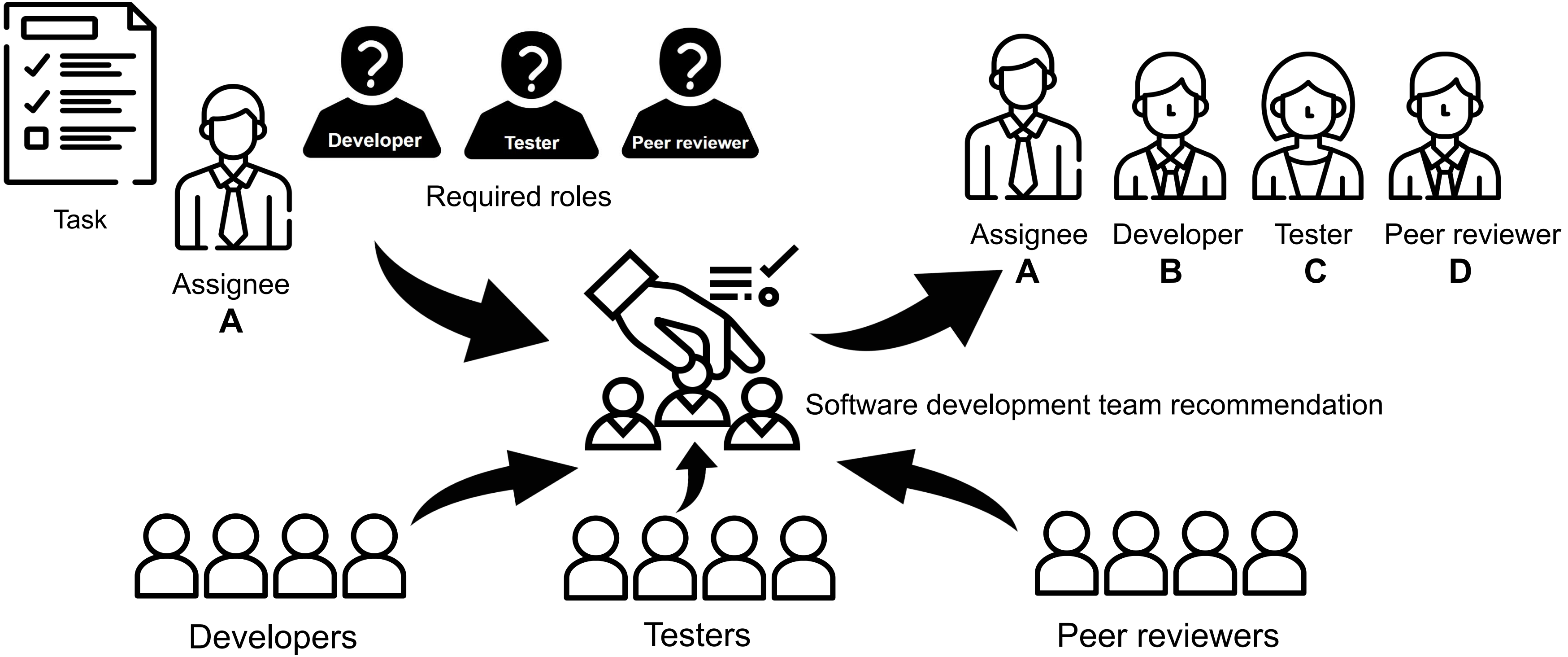
- Introduction
- Methodology
- Experiment & Result
- Conclusion

# Introduction

# Software Team

- Software development is a team based activities and it has effect on software project directly.
- Currently, team selection is done manually by experience team leader.
- Many software project is successfully resolve, but there stills exists the problem during process. (e.g. software development task reopening)

# Team Recommendation

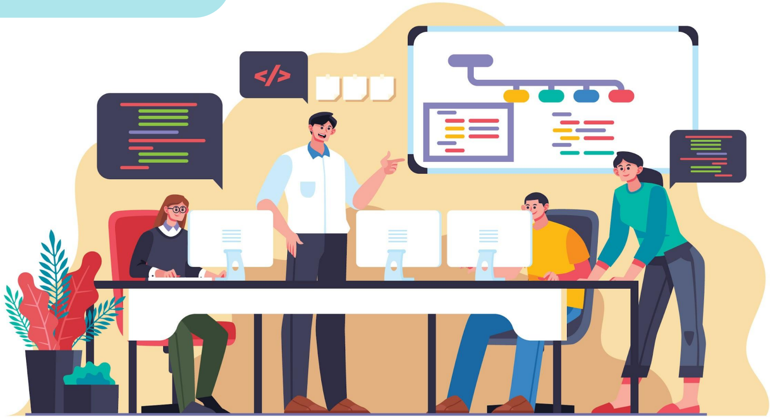


# Contributions

- We establish software team recommendations as a computational problem.
- We propose to adopt Liu et. al.’s approach to address the software team recommendation problem.
- We evaluate the result of the recommendation on the real-world Moodle dataset.

[1] H. Liu, M. Qiao, D. Greenia, R. Akkiraju, S. Dill, T. Nakamura, Y. Song, and H. M. Nezhad, “A machine learning approach to combining individual strength and team features for team recommendation,” in Preceedings of The 13th International Conference on Machine Learning and Applications, 12 2014, pp. 213–218.

# Methodology



# Liu et. al.'s approach

- Feature Weight Learning
  - Individual features
  - Team features
  - TeamStrength Score
  - Feature Weight Optimization
- Searching for the best team
  - We modified the algorithm to fit the software teams recommendation problem

# Individual Features

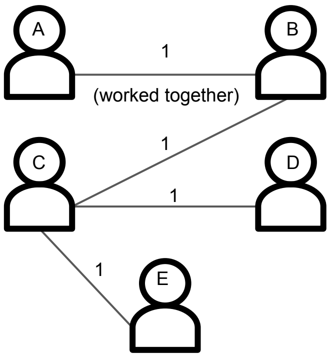
- **Experience**: Number of issues (i.e. tasks) that a person participated in
- **Win Experience:** Number of successfully resolved issues that a person participated in
- **Win rate:** Ratio of Win Experience to Experience
- **Role Experience:** Number of issues in which a person participated in a particular role

# Team Features

- **Closeness**

$$Closeness = \frac{2}{|T| \times (|T| - 1)} \sum_{p_i, p_j \in T} \frac{1}{ShortestPath(p_i, p_j)}$$

|T| is the cardinality of the team




- **Connections**

$$Connection = \frac{2}{|T| \times (|T| - 1)} \sum_{p_i, p_j \in T} e_{ij}$$

$e_{ij}$ is the number of connections (tags in comments) between $v_i$ and $v_j$

# TeamStrength Score

$$TeamStrength(T) = \frac{1}{|T|} \sum_{p_i \in T} \vec{W}_1 F(p_i) + \vec{W}_2 G(T)$$

- $F(p_i)$ is the function to calculate features of the person $p_i$.
- $G(T)$ is the function to calculate features of the Team T.
- $W_1$ and $W_2$ are the features weight vectors for a person and a team, respectively.

# Feature Weight Optimization

- The $W_1$ and $W_2$ are optimized by Logistic Regression with Non Negativity constraint.
- Binary classification of classes ***Win*** and ***Not Win***
- A ***Win*** issue is an issue whose status is *closed* with *fixed* or *done* resolution.

  Furthermore, it must not be *reopened*. The Rest of issue are ***Not Win*** issue

# Searching for the best team

- MaxLogit algorithm derived from Liu et al. ‘s is modified to recommend top K best teams.

---

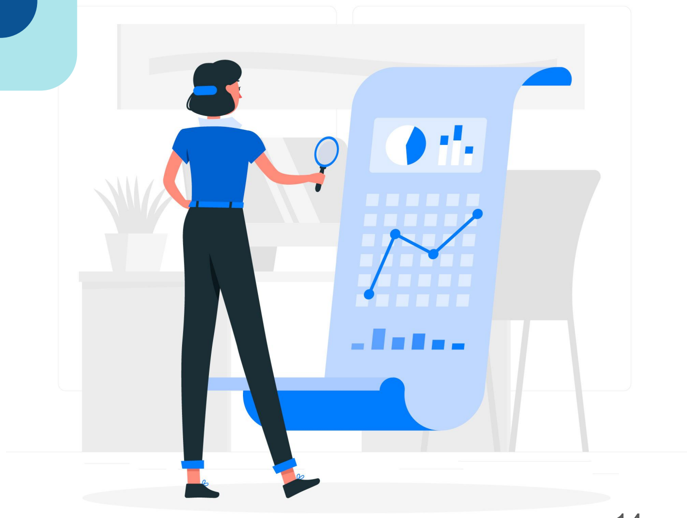
**Algorithm 1:** Maxlogit for the topK Best Team

**input :** An issue with a set of required roles, set of candidates for each role, function $Cost(T)$ is the inverse of TeamStrength score using features from $T$, number of iterations $N$, smoothing factor $\tau$, number of recommended team $K$

**output:** top K best teams and their cost

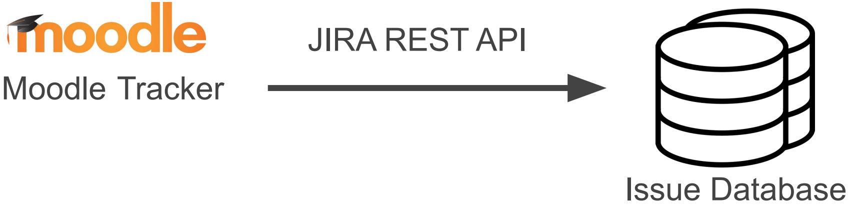
$RecTeam$ = array()<br>
Randomly select candidate for each role and generate a team $T$<br>
Append $(T, Cost(T))$ to $RecTeam$<br>
**for** $i = 1$ **to** $N$ **do**<br>
    Calculate $Cost(T)$<br>
    Randomly select a role, and replace it with a randomly selected an alternative candidate, get a new team $T'$<br>
    Calculate $Cost(T')$<br>
    Append $(T', Cost(T'))$ to $RecTeam$<br>
    $prob \leftarrow PROBABILITY(Cost(T), Cost(T'))$<br>
    $r \leftarrow random(0, 1)$<br>
    **if** $r \leq prob$ **then**<br>
        $T \leftarrow T'$<br>
    **end**<br>
**end**<br>
sort $RecTeam$ on $cost$ ascending<br>
$RecTeam \leftarrow RecTeam[: K]$<br>
**return** $RecTeam$<br>
<br>
**Function** $PROBABILITY(Cost(T), Cost(T'))$<br>
    $v_t = exp^{-Cost(T)/\tau}$<br>
    $v_{t'} = exp^{-Cost(T')/\tau}$<br>
    $prob = \frac{v_{T'}}{max(v_t, v_{T'})}$<br>
    **return** $prob$<br>
**end**

# Experiment & Results



# Dataset

- 88,655 issues was collected through JIRA REST API from Moodle Issue Tracker
- We filter out in-progress issues
- Only issue that explicitly show the role (i.e. developer, tester, reviewer, and integrator) of members are used
- In total, we perform our study on 26,744 issues



# Experimental Setting

- The issues were splitted chronologically into 80% (21,827) training set and 20% (4,917) test set.
- For training issues, it contains 18,094 ***Win*** issues and 3,733 ***Not Win*** issues.
- The test set contain only ***Win*** issues.
- Random Approach is used as baseline.

# Feature Exploration

TABLE I: The mean of features comparing between Liu et al. and Moodle dataset.

| Features | Liu et al. dataset | | Moodle dataset | |
|---|---|---|---|---|
| | Win | Not Win | Win | Not Win |
| Experience | 0.4921 | 0.3805 | 0.4748 | 0.4293 |
| Win Experience | 0.4351 | 0.3243 | 0.4738 | 0.4261 |
| Win Rate | 0.8352 | 0.7176 | 0.8241 | 0.7937 |
| Role Experience | 0.7467 | 0.5839 | 0.4463 | 0.4072 |
| Team Closeness | 0.4111 | 0.3717 | 0.7418 | 0.8682 |
| Connection | 0.1896 | 0.1924 | 0.0378 | 0.0339 |

# Feature Weight

TABLE II: The feature weights from Logistic Regression model using in Liu et al. approach.

| Feature | Liu et al. dataset | Moodle dataset |
|---|---|---|
| Experience | 0.1545 | 0.0341 |
| Win Experience | - | - |
| Win Rate | 2.9949 | 12.2525 |
| Role Experience | 1.9881 | - |
| Team Closeness | 0.4993 | - |
| Connection | 1.8699 | - |
| Intercept | -3.6404 | -8.3843 |

# Evaluation

- Validated using standard evaluation metrics for recommendation systems
  - Mean Reciprocal Rank (MRR)
  - Mean Rank of Hits
  - Mean Rank (MR)
  - Hit@10
  - Mean Average Precision (MAP).
- Two evaluation protocols
  - Exact Match
  - Partial Match

Mahidol University

Mahidol University

# Result

TABLE III: Evaluation results from recommendation outputs using exact match and partial match protocol.

| | Evaluation Metric | Random Approach | Liu et. al. Approach |
|---|---|---|---|
| | MRR | 0.0011 | 0.0024 |
| | MR of Hits | 7.3461 | 5.3889 |
| Exact Match | MR | 10.9586 | 10.9545 |
| | Hit @ 10 | 0.0058 | 0.0081 |
| | MAP | 0.0011 | 0.0024 |
| Partial Match | MAP | 0.0227 | 0.0290 |

# P@k Exact Match

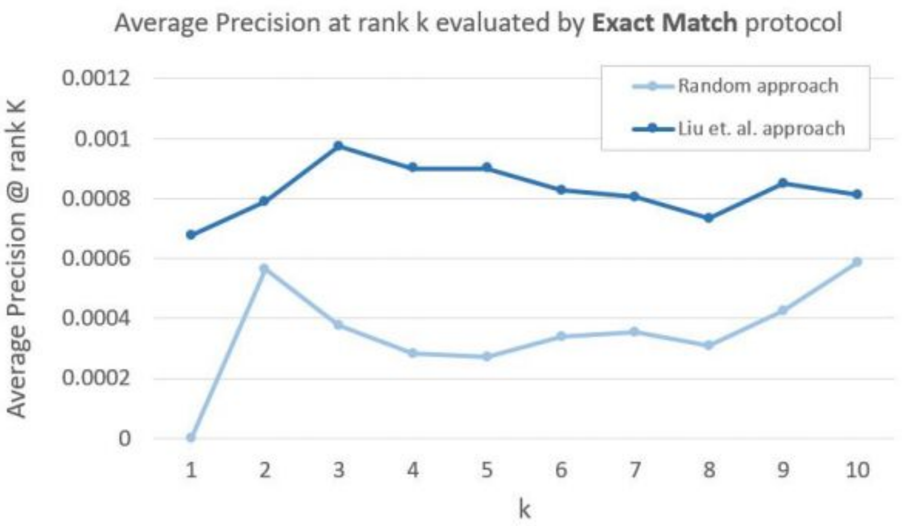



Fig. 2: The changes of average precision at rank $K$ comparing the random approach (baseline) and Liu et. al. approach

# P@k Partial Match

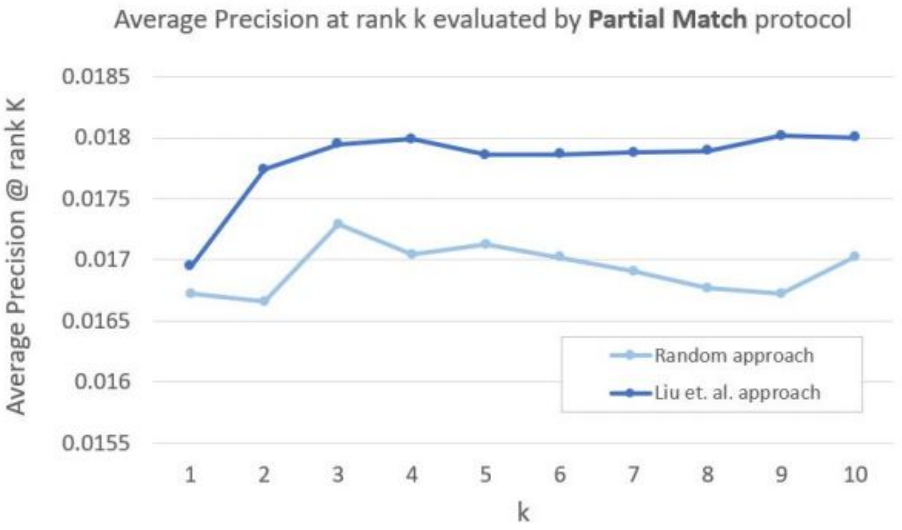
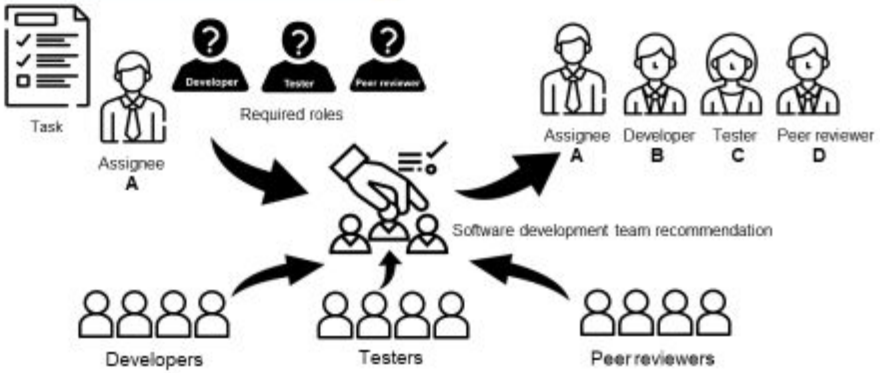



Fig. 3: The changes of average precision at rank $K$ comparing the random approach (baseline) and Liu et. al. approach

# Conclusion

## Team Recommendation








## Liu et. al.'s approach

- Feature Weight Learning
  - Individual features
  - Team features
  - TeamStrength Score
  - Feature Weight Optimization
- Searching for the best team
  - We modified the algorithm to fit the software teams recommendation problem





## Dataset

- 88,655 issues was collected through JIRA REST API from Moodle Issue Tracker
- We filter out in-progress issues
- Only issue that explicitly show the role (i.e. developer, tester, reviewer, and integrator) of members are used
- In total, we perform our study on 26,744 issues






## Result

TABLE III: Evaluation results from recommendation outputs using exact match and partial match protocol.

| | Evaluation Metric | Random Approach | Liu et. al. Approach |
|---|---|---|---|
| Exact Match | MRR | 0.0011 | 0.0024 |
| | MR of Hits | 7.3461 | 5.3889 |
| | MR | 10.9586 | 10.9545 |
| | Hit @ 10 | 0.0058 | 0.0081 |
| | MAP | 0.0011 | 0.0024 |
| Partial Match | MAP | 0.0227 | 0.0290 |

Mahidol University

Mahidol University

# Thank you