FaRS: A High-Performance Automorphism-Aware Algorithm for Graph Similarity Matching

Fan Wang[1], Weiren Yu[2], Hai H. Wang[1] and Victor Chang[1]
[1]*Aston University, Birmingham B4 7ET, U.K.*
[2]*University of Warwick CV4 7AL, U.K.*

Keywords: Web Search, Similarity Search, Link Analysis.

Abstract: Role-based similarity search, predicated on the topological structure of graphs, is a highly effective and widely applicable technique for various real-world information extraction applications. Although the prominent role-based similarity algorithm, RoleSim, successfully provides the automorphic (role) equivalence of similarity between pairs of nodes, it does not effectively differentiate nodes that exhibit exact automorphic equivalence but differ in terms of structural equivalence within a given graph. This limitation arises from disregarding most adjacency similarity information between pairs of nodes during the RoleSim computation. To address this research gap, we propose a novel single-source role similarity search algorithm, named FaRS, which employs the top Γ maximum similarity matching technique to capture more information from the classes of neighboring nodes, ensuring both automorphic equivalence and structural equivalence of role similarity. Furthermore, we establish the convergence of FaRS and demonstrate its adherence to various axioms, including uniqueness, symmetry, boundedness, and triangular inequality. Additionally, we introduce the Opt_FaRS algorithm, which optimizes the computation of FaRS through two acceleration components: path extraction tracking and pre-computation (P-speedup and Out-speedup approach). Experimental results on real datasets demonstrate that FaRS and Opt_FaRS outperform baseline algorithms in terms of both accuracy and efficiency.

1 INTRODUCTION

In the era of information technology, the research concerning the extraction of valuable information from topological structures has witnessed a substantial increase (Rao et al., 2009; Shahabi et al., 2001; Yang, 2022; Wang et al., 2018; Li et al., 2015). Role similarity analysis (Everett, 1985) is one of the significant techniques for analyzing complex graph structures, particularly social networks. It enables the accurate identification of role equivalence, also known as automorphic equivalence, of pairwise nodes within a network. The basic principle of role similarity search is that two nodes have a similar role only if they interact with similar objects. To demonstrate the practical application and importance of role and role similarity detection based on graph topology in our daily life, we present an example below. Figure 1 illustrates a social network representing a project team, where the nodes correspond to the employees involved in the project, and the edges denote their interactions and connections. The team is organized into three groups based on their assigned tasks, and each group further

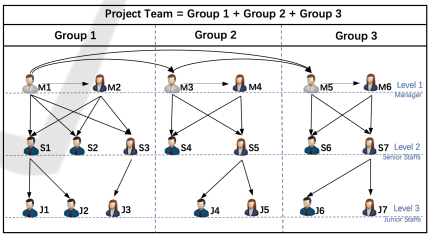


Figure 1: A Social Network of A Project Team.

comprises three job levels: project managers (level 1), senior employees (level 2), and junior staffs (level 3). An employee's role within the project is determined by his/her position.

The primary objective of a role similarity detection algorithm is to efficiently and accurately identify nodes in a social network that fulfill similar roles as a given query node. This algorithm takes a specific node as the query and computes role similarity scores between that node and all other nodes in the graph, which is commonly referred to as a *single-source* role similarity search algorithm. For instance, if node $J3$ is selected as the query node,

Wang, F., Yu, W., Wang, H. and Chang, V.
FaRS: A High-Performance Automorphism-Aware Algorithm for Graph Similarity Matching.
DOI: 10.5220/0012724000003708

In *Proceedings of the 9th International Conference on Complexity, Future Information Systems and Risk (COMPLEXIS 2024)*, pages 17-29
ISBN: 978-989-758-698-9; ISSN: 2184-5034

nodes $[J4, J5, J6, J7, J1, J2]$ are expected to exhibit higher role similarity scores with $J3$ compared to other nodes like $S1$ or $M1$, as they share similar roles within the project, specifically as junior staff. Role similarity detection finds practical applications in diverse domains, including co-authorship networks (Lee, 2012) and e-commerce website recommendation systems (Diao et al., 2019).

The RoleSim algorithm, developed by Jin (2014?), computes role similarity scores by taking the mean of the maximum matchings between a node's in-neighbors (Rothe and Schütze, 2014). However, despite successfully addressing pairwise automorphic equivalence, Jin's RoleSim algorithm has several limitations, including limited accuracy and poor performance on large networks (Section 4).

To overcome the limitations of RoleSim, we present a novel model, namely FaRS, for conducting single-source role similarity searches based on graph topology. FaRS exhibits the capability to efficiently generate more accurate role similarity scores, even for large graphs. Additionally, we introduce Opt_FaRS, an optimization algorithm for FaRS that significantly reduces computational costs further by minimizing the number of calls to the maximum matching algorithm. In summary, FaRS and Opt_FaRS have the following distinctive characteristics:

- **Accuracy:** They can provide more precise role classification results compared to the best-known existing competitors.
- **Efficiency:** FaRS is capable of efficiently computing single-source role similarity search results over extensive graphs. The performance is further enhanced by Opt_FaRS.
- **Index-free:** No additional disk space is required to store indexing results, thereby minimizing the cost of RAM usage.

This paper is organized as follows. In Section 2, we provide a comprehensive review of the current RoleSim algorithm (Rothe and Schütze, 2014). Section 3 presents an in-depth analysis of the limitations associated with RoleSim-like algorithms when applied to graphs. In Section 4, we introduce a novel role similarity search algorithm, FaRS, and provide proofs of its convergence, uniqueness, symmetry, boundedness, and triangular inequality. Furthermore, Section 5 presents the Opt_FaRS algorithm, which optimizes the performance of FaRS. Finally, in Section 6, we conduct experiments using real datasets to demonstrate the superior performance of our FaRS and Opt_FaRS algorithms in terms of both accuracy and efficiency when compared to state-of-the-art similarity search algorithms.

2 RELATED WORK

Several related researches have been derived based on the RoleSim (Shao et al., 2019; Chen et al., 2021; Chen et al., 2020). (Shao et al., 2019) proposed a seedless de-anonymization method called RoleMatch, which consists of two parts: the novel role similarity detection algorithm RoleSim++, and the NeighborMatch matching algorithm based on the calculated role similarity score. RoleSim++ calculates the role similarity value of a node pair based on the maximum matching value extracted from the in-neighbors' and out-neighbors' role similarity value matrix of the node pair, which is different from the RoleSim algorithm. To improve the computational efficiency of RoleSim++, they proposed the α-RoleSim++ algorithm, which extracts information only from node pairs' role similarity scores greater than the threshold, and ignores other pairs.

Currently, StructSim (Chen et al., 2020) is the most state-of-the-art algorithm for role similarity search. StructSim calculates the role similarity scores through the maximum matching value of the horizontal similarity between each k-neighborhood subgraph. To improve the computational efficiency of the StructSim algorithm, the maximum match in the algorithm is replaced with the BinCount match. In the BinCount matching algorithm, the index of the nodes of each layer needs to be recorded. To create the index of the nodes of each layer more efficiently, Flajolet-Martin Sketch was proposed.

3 PRELIMINARY

Here, we provide an overview of the formulae and properties of the RoleSim algorithm. In the context of network similarity analysis, there are four types of equivalence that are considered: structural equivalence, automorphic equivalence, exact coloration, and regular equivalence (Rothe and Schütze, 2014). Among these, automorphic equivalence is the most fundamental type of equivalence for role similarity. The RoleSim algorithm (Rothe and Schütze, 2014) focuses on exploring the role similarity between pairs of nodes in networks (graphs) and provides a real-valued measure of role similarity that verifies automorphic equivalence. The algorithm is based on the recursive principle that "two nodes share the same role if they interact with equivalent sets of neighbors". By evaluating the role similarity scores and performing role classification based on how node pairs interact with other nodes, the algorithm effectively captures the role similarities. Before delving into the details

of RoleSim, we present the fundamental intuition and various notations used throughout this paper.

In a directed graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, $\mathbf{V}$ and $\mathbf{E}$ denote the vertices and edges in $\mathbf{G}$, respectively. A node u is an *in-neighbor* of node v if $(u,v) \in E$. Similarly, an *out-neighbor* can be defined as a node that has an outgoing edge to another node. The sets of in-neighbors and out-neighbors of a node v in the graph are denoted by $I(v)$ and $O(v)$, respectively. The *in-degree* and *out-degree* of a node v in the graph represent the number of in-neighbors and out-neighbors of v and are denoted by $\deg_v^-$ and $\deg_v^+$, respectively. Furthermore, $\text{mindeg}^-(u,v)$ is the smaller in-degree between node u and node v, which can be expressed mathematically as $\text{mindeg}^-(u,v) = \min(\deg_u^-, \deg_v^-)$. Similarly, $\text{maxdeg}^-(u,v)$ is the larger in-degree between node u and node v, i.e., $\text{maxdeg}^-(u,v) = \max(\deg_u^-, \deg_v^-)$. For example, consider the node pair $(S1, J1)$ in Figure 1. The in-degree of node $S1$ is 2, and $\text{mindeg}^-(S1,J1) = \min(2,1) = 1$ and $\text{maxdeg}^-(S1,J1) = \max(2,1) = 2$.

The RoleSim algorithm is founded upon the concept of maximal matching of neighbors' similarity, which recursively establishes the similarity between nodes as the mean similarity of the maximum weight matching among their neighbors. *Maximum Weighted Matching* (MWM) is a well-known problem in graph theory where the objective is to find, in a weighted graph, a matching that has the highest possible sum of weights. The RoleSim algorithm calculates the role similarity $rs(u,v)$ between nodes u and v using the following formula. The complete matrix of pairwise similarity values between all nodes is referred to as $\mathbf{R}$:

$$rs(u,v) = (1-C) \max_{\mathcal{MA}(u,v)} \frac{\sum_{(x,y)\in\mathcal{MA}(u,v)} rs(x,y)}{deg_u^- + deg_v^- - mindeg^-(u,v)} + C \tag{1}$$

Here, $x \in I(u)$, $y \in I(v)$, $\mathcal{MA}(u,v)$ denotes a matching between $I(u)$ and $I(v)$, C signifies the decay factor $(0 < C < 1)$, and $deg_u^- + deg_v^- - mindeg^-(u,v)$ is equivalent to $maxdeg^-(u,v)$.

A weighted bipartite matching of $\mathcal{MA}(u,v)$ can be defined using $rs(x,y)$ scores as the weights. The weight of the matching is given by the sum of the $rs(x,y)$ scores for all (x,y) pairs in $\mathcal{MA}(u,v)$, denoted by $w(\mathcal{MA}(u,v))$. Mathematically, it means $w(\mathcal{MA}(u,v)) = \sum_{(x,y)\in\mathcal{MA}(u,v)} rs(x,y)$. A matching $\mathcal{MA}(u,v)$ is said to be maximal if its weight is the maximum among all possible matchings, denoted as $\widetilde{\mathcal{M}}(u,v)$, and the weight of $\widetilde{\mathcal{M}}(u,v)$ is denoted by $M(u,v)$, i.e., $M(u,v) = w(\widetilde{\mathcal{M}}(u,v))$.

Using the notation of $\widetilde{\mathcal{M}}(u,v)$ and $M(u,v)$, the definition of $rs(u,v)$ from Equation 1 can also be expressed as follows (Rothe and Schütze, 2014):

$$rs(u,v) = (1-C)\frac{\mathcal{M}(u,v)}{maxdeg^-(u,v)} + C \tag{2}$$

The matching selection process used by RoleSim is explained using the following example.

Example 3.1. *Consider a directed graph $G = (\mathbf{V}, \mathbf{E})$, where $(u,v) \in V$ are two nodes. The set of in-neighbors of node u is denoted as $I(u) = \{a,b,c\}$, while the set of in-neighbors of node v is denoted as $I(v) = \{d,e,f,g,h\}$ in G. A subset of the RoleSim matrix of values ($\mathbf{R}$) is presented in Figure 2, where each value represents the similarity of the pairings of neighbors between these two vertices. Assume that these values have the following ordering: $rs(a,d) = \max(rs(a,:))$, $rs(b,f) = \max(rs(b,:))$, and $rs(c,e) = \max(rs(c,:))$.*

In RoleSim, a matching involves selecting a single cell from each row and column. When the number of rows is different from the number of columns, the size of the matching is limited to $mindeg^-(u,v)$. In this example, the matching size is restricted to 3. A maximal matching is a matching where the sum of the selected cells is maximized. As depicted in Figure 2, following the principle of maximum weighted matching, the maximal matching results of the in-neighbor similarity matrix are enclosed by a solid square and can be expressed as $\mathcal{M}(u,v) = rs(a,d) + rs(b,f) + rs(c,e)$. In the subsequent sections of this paper, $\mathcal{M}^1(u,v)$ will be used to refer to this maximal weighted matching result of the in-neighbor similarity matrix for the node pair (u,v) generated by the RoleSim algorithm, and it is referred to as the first-order maximal weighted matching result. This distinction is made to differentiate it from the higher Γ^{th} order maximal weighted matching used in the proposed FaRS algorithm (Section 5). For instance, $\mathcal{M}^2(u,v)$ denotes the second-largest weighted matching result.

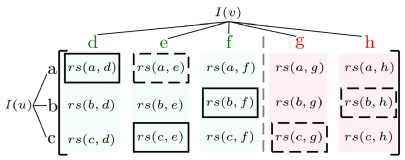


Figure 2: In-Neighbour Similarity Matrix of Node-Pair (u,v).

The RoleSim algorithm, which follows an iterative process to calculate the role similarity score between node pairs $(u,v) \in \mathbf{V}$, consists of two phases. First, the role similarity search scores matrix $\mathbf{R}$ is initialized. In the second phase, during the k^{th} iteration, the role similarity score between the node pair (u,v) is computed based on the role similarity scores from the previous $(k-1)^{th}$ iteration. This computation is performed using the following equation:

$$rs_k(u,v) = (1-C)\frac{\mathcal{M}^1_{k-1}(u,v)}{\max\deg^-(u,v)} + C \tag{3}$$

The task of retrieving similarity relationships between a given node q and all other nodes of graph is known as *single-source similarity search* and is denoted as $\mathbf{rs}(:,q)$, which is a vector of similarity scores between q and rest nodes. It can be calculated as:

$$\mathbf{rs}_k(:,q) = (1-C)\mathbf{M}^1_{k-1}(:,q) \oslash \mathbf{MAXDEG}^-(:,q) + \mathbf{C}_{n\times 1} \tag{4}$$

$\oslash$ is element-wise division of the respected positions of two vectors, and n is the number of nodes in the graph. The vector $\mathbf{M}^1_{k-1}(:,q)$ contains the first order maximum weighted matching of node-pair $((i,q)|i \in \mathbf{V})$, while the vector $\mathbf{MAXDEG}^-(:,q)$ contains the maximum in-degree of node-pair $((i,q)|i \in \mathbf{V})$.

The RoleSim algorithm satisfies the following properties (Rothe and Schütze, 2014). In Section 5.2, we will demonstrate that these properties also held in our algorithm FaRS and Opt_FaRS.

1. **Boundedness:** The similarity score $rs(*,*)$ always exists and is unique, and $C \leq rs(*,*) \leq 1$.
2. **Monotone Convergence:** The value of $rs_0(*,*)$ is the upper bound of $rs_k(*,*)$, that is, $rs_k(*,*) \geq rs_{k+1}(*,*)$.
3. **Convergence:** The result of $rs_k(*,*)$ converges to $rs(*,*)$, that is, as k approaches infinity, $lim_{k\to\infty} rs_k(*,*) = rs(*,*)$.
4. **Triangle inequality:** The RoleSim algorithm satisfies the triangle inequality.

4 LIMITATIONS OF RoleSim

Despite its usefulness, the RoleSim algorithm suffers from two main limitations, namely inaccuracy and computational inefficiency. We provide a detailed analysis of the causes of these limitations here.

4.1 Limitation 1 (Inaccuracy)

The role similarity score in RoleSim is determined by computing the maximum matching from the neighbor similarity score matrix (Gabow et al., 2001). As mentioned earlier, the maximum matching refers to selecting the highest sum value of one cell from each row and column in the in-neighbor similarity matrix. If the number of rows differs from the number of columns, the matching size is constrained to be the minimum of the two. It means that RoleSim excludes the in-neighbor nodes that represent the surplus rows/columns in the similarity matrix from the matching selection process. Moreover, even when an in-neighbor row/column is included in the matching selection, most similarity scores associated with it are disregarded. This is because RoleSim only considers one value from each row and column of the in-neighbor score matrix (the first-order maximal weighted matching). Consequently, these two factors can compromise the accuracy of RoleSim and result in ambiguous role classification. To demonstrate the limitations of the RoleSim algorithm, we employ Example 3.1 once again and refer to Figure 3.

Figure 2 demonstrates a limitation of the RoleSim algorithm, where it only captures information from a specific column (green area) while disregarding another column (red area). This is due to the algorithm's focus on maximizing the matching based on the smaller in-degree between the compared nodes. As a result, the algorithm neglects the in-neighbors' information that corresponds to the difference in in-degrees between the nodes. Additionally, beyond the first-order maximal matching (the three solid square cells), all information from the column (d,e,f) is ignored. These contribute to the inaccuracy of RoleSim.

In Figure 1, we compare the role similarity search results of RoleSim with our proposed FaRS algorithm ($J3$ as a query). The right table in Figure 3 shows the results obtained from RoleSim, where all nodes $[M6,J3,J5,J7,J1,J2,J4,J6]$ have the same role similarity scores with node $J3$. It suggests that these nodes belong to the same role classification. However, in reality, $[J3,J5,J7,J1,J2,J4,J6]$ are part of the junior staff classification (level 3), whereas $M6$ belongs to the manager classification (level 1). Thus, the results of RoleSim may contain errors that require correction. The left table in Figure 3 displays the results produced by our proposed algorithm, FaRS, which can correctly reflect this fact.

Moreover, the RoleSim algorithm suffers from an accuracy problem in recognizing the structural equivalence (group classification in Figure 1). In contrast, the scores produced by FaRS in the left table are different from those of RoleSim, and thus, the similarity levels can be ordered and ranked. For instance, the role similarity scores between node $J3$ and nodes $[J4,J5]$ are 0.4, and both $[J4,J5]$ belong to group 2. Similarly, we can identify $[J6,J7]$ and $[J3,J1,J2]$ as belonging to group 1 and group 3, respectively. Note that although $J3$ and $[J1,J2]$ have different in-neighbors (the in-neighbor of node $J3$ is node $S3$, and the in-neighbor of $[J1,J2]$ is node $S1$), both nodes $S3$ and $S1$ belong to group 1. Thus, $J3$ and $[J1,J2]$ are more structurally related. FaRS recog-

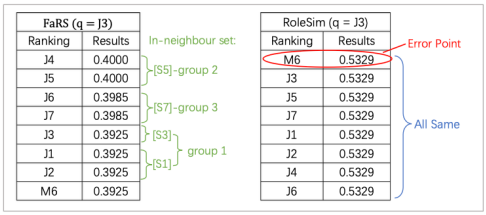


Figure 3: Role similarity search results of RoleSim & FaRS.

nizes this fact, which is evident in the results of FaRS ($FaRS(J1,J3) = FaRS(J2,J3) = FaRS(J3,J3)$).

This example illustrates that FaRS not only corrects the errors in RoleSim, improves the accuracy of automorphic equivalence, but also yields structural equivalence, which RoleSim entirely overlooked.

Limitation 2 (Computational Inefficiency). The single-source similarity search in RoleSim is computationally expensive due to redundant computations and repeated use of the maximum matching algorithm. In contrast, FaRS improves computational efficiency by minimizing the number of maximum matching calls and focusing on shared information, resulting in faster processing times.

5 PROPOSED SCHEMA

This section introduces FaRS, a new role-based similarity search algorithm that surpasses existing algorithms such as RoleSim in identifying and categorizing nodes in graphs.

5.1 FaRS

Most existing role-based similarity algorithms, including RoleSim, have limitations (as discussed in Section 4) when applied to real-life applications. To overcome these limitations and improve role-based similarity search, we propose the FaRS algorithm. FaRS selects the top Γ best matching pairs in the maximum matching process, providing more accurate role similarity scores. We denote the role similarity score vector between a single query q and node set set_A as $\mathbf{RS}^{\Gamma}(\text{set}_A, q)$, the role similarity score matrix between node sets set_A and set_B as $\mathbf{RS}^{\Gamma}(\text{set}_A, \text{set}_B)$, and the role similarity score between node pair (u,v) as $RS^{\Gamma}(u,v)$. Γ^{th} maximum weighted matching is defined as the maximum weighted matching of the matrix excluding the top $(\Gamma-1)$ largest weighted matching (Murty, 1968). In Section 7, we show that setting Γ equal to 3 can achieve the best balance between accuracy and efficiency of FaRS.

Definition 1 (FaRS). *Given a directed graph $G = (\mathbf{V}, \mathbf{E})$, a query $q \in V$, and $|V| = n$, the single-source FaRS role similarity scores w.r.t. q, denoted by $\mathbf{RS}^{\Gamma}(:,q)$, are defined as follows:*

$$\begin{aligned}\mathbf{RS}^{\Gamma}(:,q) = (1-C)\big(&\max_{\mathbf{MA}^1(:,q)} \textstyle\sum_{(x,y)\in\mathbf{MA}^1(:,q)} RS^{\Gamma}(x,y) + \\ &\lambda \cdot \max_{\mathbf{MA}^2(:,q)} \textstyle\sum_{(x,y)\in\mathbf{MA}^2(:,q)} RS^{\Gamma}(x,y) + \\ &\cdots + \lambda^{(\Gamma-1)} \cdot \max_{\mathbf{MA}^{\Gamma}(:,q)} \textstyle\sum_{(x,y)\in\mathbf{MA}^{\Gamma}(:,q)} RS^{\Gamma}(x,y)\big) \\ \oslash(1+\lambda+\cdots+\lambda^{(\Gamma-1)})&(\mathbf{DEG}^{-}_{i=1:n} + [\mathbf{deg}^{-}_{q}]_{n\times 1} - \mathbf{MINDEG}^{-}(:,q)) + \mathbf{C}_{n\times 1}\end{aligned} \tag{5}$$

where $\mathbf{MA}^{\Gamma}(:,q)$ is the top Γ^{th} order matching of the in-neighbour similarity matrix of nodes ($i = 1:n$) and query q, and the corresponding maximum weighted matching values can be represented $\boldsymbol{M}^{\Gamma}(:,q)$. The normalization coefficient for the match of each order is denoted by λ^{γ} ($0 \le \lambda^{\gamma} \le 1$ and $1 \le \gamma \le \Gamma$). $\oslash$ denotes the element-wise division of the corresponding positions of two vectors. The vector $\mathbf{DEG}^{-}_{i=1:n}$ is the in-degree of all the nodes $i (i \in \mathbf{V})$. $[\mathbf{deg}^{-}_{q}]_{n\times 1}$ is a vector whose values are the in-degree of node q. $\mathbf{MINDEG}^{-}(:,q)$ is a vector, and the values of this vector are the minimum value of node-pair $[(deg^{-}_{i}, deg^{-}_{q}) | i \in \mathbf{V}]$. C is the decay factor ($0 < C < 1$).

Definition 4.1 presents the FaRS algorithm for role similarity which captures the top Γ maximum weighted matching values from a node pair's in-neighbour similarity matrix. In order to prevent division by zero in the numerator of Eq. 5, the following special cases are incorporated into the equation.

$$\begin{cases} RS^{\Gamma}(u,q) = C & deg^{-}_{u} = 0 \quad or \quad deg^{-}_{q} = 0 \\ \mathbf{RS}^{\Gamma}(:,q) = \mathbf{C}_{n\times 1} & deg^{-}_{q} = 0 \end{cases}$$

Lemma 5.1. *Given the in-neighbor similarity matrix of a node pair (u,v) in G, the value of $mindeg^{-}(u,v)$ is the upper bound for the various orders of maximal weighted matching $\mathcal{M}^{\gamma}(u,v)$ (where $1 \le \gamma \le \Gamma$). These matching values are arranged in descending order, i.e., $mindeg^{-}(u,v) \ge \mathcal{M}^{1}(u,v) \ge \mathcal{M}^{2}(u,v) \ge \cdots \ge \mathcal{M}^{\Gamma}(u,v) \ge 1$.*

The proof of this lemma is omitted from this paper due to space limitations and it can be found from the technical report [removed].

Computation Of FaRS. Given a graph $G = (V,E)$, the computation of the single-source FaRS role similarity scores with respect to the query q, denoted as $\mathbf{RS}^{\Gamma}(:,q)$, follows an iterative process until convergence. The total number of iterations in the algorithm is denoted as K, with each iteration represented by k ($K = \max(k)$). Initially, the matrix $\mathbf{RS}^{\Gamma}_{0}(V,V)$ is initialized as $\mathbf{ones}_{n\times n}$. Then, the role similarity scores $\mathbf{RS}^{\Gamma}(:,q)$ at iteration k are computed using the following equation. The second phase is repeated until convergence is reached.

$$\mathbf{RS}_k^{\Gamma}(:,q) = (1-C)(\mathbf{M}_{k-1}^{1}(:,q) + \lambda \cdot \mathbf{M}_{k-1}^{2}(:,q) + \cdots + \lambda^{(\Gamma-1)} \cdot \mathbf{M}_{k-1}^{\Gamma}(:,q)) \oslash (1+\lambda+\cdots+\lambda^{(\Gamma-1)})\mathbf{MAXDEG}^{-}(:,q) + \mathbf{C}_{n\times 1} \tag{6}$$

Theorem 5.2. ***Convergence:*** *For a directed graph $G = (\mathbf{V}, \mathbf{E})$, and any query $q \in V$, the FaRS role similarity search algorithm is converged with the initialisation of $\mathbf{RS}_{k=0}^{\Gamma} = \mathbf{Ones}_{n\times n}$, and the iterative computation of the FaRS algorithm w.r.t. query q at iteration k satisfies $\lim_{k\to\infty} \mathbf{RS}_k^{\Gamma}(:,q) = \mathbf{RS}^{\Gamma}(:,q)$.*

Proof. To save space, we set $\Gamma = 2$ without loss of generality. We aim to demonstrate the convergence of the role similarity scores $\mathbf{RS}_k^2(:,q)$ to $\mathbf{RS}^2(:,q)$ as k approaches infinity. This requires proving the convergence of node-pair role similarity search scores. Specifically, we need to show that $\lim_{k\to\infty} RS_k^2(i,q) = RS^2(i,q)$ holds for a randomly selected node i from $\mathbf{V}$.

When $k = 0$, according to the initialisation, $\mathbf{RS}_0^2 = \mathbf{Ones}_{n\times n}$, and the definition of FaRS (Eq. 5), $\mathbf{RS}_1^2(:,q)$ can be calculated as follows:

$$RS_1^2(i,q) = (1-c)\frac{\sum_{(x,y)\in M_0^1(i,q)} RS_0^2(x,y) + \lambda\sum_{(x',y')\in M_0^2(i,q)} RS_0^2(x',y')}{(1+\lambda)\,\mathrm{maxdeg}^{-}(i,q)} + C$$
$$= (1-c)\frac{(1+\lambda)\,\mathrm{mindeg}^{-}(i,q)}{\underbrace{(1+\lambda)\,\mathrm{maxdeg}^{-}(i,q)}_{\leq 1}} + C \quad = 1 = RS_0^2(i,q)$$

Next we assume that $\mathrm{RS}_k^2(*,*) \leqslant RS_{k-1}^2(*,*)$ holds for any node pair in the graph , we will proof $\mathrm{RS}_{k+1}^2(*,*) \leqslant RS_k^2(*,*)$ holds next. Refer to the computation of FaRS (Eq. 6), $\mathbf{RS}_{k+1}^2(:,q)$ can be generated as follows:

$$RS_{k+1}^2(i,q) = (1-C)\frac{M_k^1(i,q) + \lambda M_k^2(i,q)}{(1+\lambda)\,\mathrm{maxdeg}^{-}(i,q)} + C$$
$$= (1-c)\frac{\sum_{(x,y)\in M_k^1(i,q)} RS_k^2(x,y) + \lambda\sum_{(x',y')\in M_k^2(i,q)} \mathrm{RS}_k^2(x',y')}{(1+\lambda)\,\mathrm{maxdeg}^{-}(i,q) + C}$$
$$\Updownarrow \text{hypothesis}\, \mathrm{RS}_k^2(*,*) \leqslant RS_{k-1}^2(*,*)$$
$$\leq (1-C)\frac{\sum_{(x,y)\in M_{k-1}^1(i,q)} RS_{k-1}^2(x,y) + \lambda\sum_{(x',y')\in M_{k-1}^2(i,q)} RS_{k-1}^2(x',y')}{(1+\lambda)\,\mathrm{maxdeg}^{-}(i,q)} + C$$
$$= \underbrace{(1-C)\frac{M_{k-1}^1(i,q) + \lambda M_{k-1}^2(i,q)}{(1+\lambda)\,\mathrm{maxdeg}^{-}(i,q)} + C}_{\text{refer to Eq. 6}} = \mathrm{RS}_k^2(i,q)$$

Thus we conclude $\mathrm{RS}_{k+1}^2(*,*) \leqslant RS_k^2(*,*)$ holds for any node-pair in the graph, when the assumption $\mathrm{RS}_k^2(*,*) \leqslant RS_{k-1}^2(*,*)$ holds.

We have demonstrated that $\lim_{k\to\infty} RS_k^2(i,q) = RS^2(i,q)$ for any randomly chosen node i in the network $\mathbf{V}$. This result holds true for all nodes in $\mathbf{V}$. Consequently, as the number of iterations k increases towards infinity, the role similarity scores of FaRS eventually converge. Specifically, we have $\lim_{k\to\infty} \mathbf{RS}_k^{\Gamma}(:,q) = \mathbf{RS}^{\Gamma}(:,q)$.

□

5.2 The Axiomatic Properties of FaRS

In this subsection, we establish that FaRS preserves the crucial axiomatic properties of RoleSim. Note that, for brevity, without loss of generality, we assume that Γ in the FaRS algorithm is fixed at 2 in all proofs. In the following, let $G = (\mathbf{V}, \mathbf{E})$ be a graph, q be a randomly selected query, and k be an iteration number.

Theorem 5.3. ***Symmetry:*** *The role similarity scores generated by Eq. 6 satisfy $\mathbf{RS}_k^{\Gamma}(:,q) = \mathbf{RS}_k^{\Gamma}(q,:)$, where $\mathbf{RS}_k^{\Gamma}(q,:)$ denotes the q^{th} row of the role similarity matrix.*

Theorem 5.4. ***Monotone Convergence:*** *The role similarity scores generated by Eq. 6 satisfy $\mathbf{RS}_k^{\Gamma}(:,q) \leq \mathbf{RS}_{k-1}^{\Gamma}(:,q)$.*

For the detailed proof of Theorem 5.3 and Theorem 5.4, please refer to the technical report [removed] due to limited space.

Theorem 5.5. ***Boundedness:*** *The role similarity scores generated by Eq. 6 satisfy $\mathbf{C}_{n\times 1} \leq \mathbf{RS}_k^{\Gamma}(:,q) \leq \mathbf{1}_{n\times 1}$. Here $\mathbf{C}_{n\times 1}$ is a vector whose values are all C, and $\mathbf{1}_{n\times 1} = ones(n,1)$.*

Proof. We prove Theorem 5.5 by showing that for any node $u \in \mathbf{V}$, $C \leq \mathbf{RS}_k^2(u,q) \leq 1$. We begin by initializing $\mathbf{RS}_k^2 = ones(n,n)$, where n is the number of nodes of the graph. Eq. 6 is as follows:

$$RS_k^2(u,q) = (1-C)\frac{\mathcal{M}_{k-1}^1(u,q) + \lambda\mathcal{M}_{k-1}^2(u,q)}{(1+\lambda)maxdeg^{-}(u,q)} + C$$
$$\leq (1-C)\underbrace{\frac{(1+\lambda)mindeg^{-}(u,q)}{(1+\lambda)maxdeg^{-}(u,q)}}_{0\leq *\leq 1} + C \leq 1$$

And $\frac{\mathcal{M}_{k-1}^1(u,q) + \lambda\mathcal{M}_{k-1}^2(u,q)}{(1+\lambda)maxdeg^{-}(u,q)}$ is a non-negative number. If the in-degree of node pair (u,q) equals zero, then $RS_k^2(u,q) = C$. Therefore, $C \leq RS_k^2(u,q) \leq 1$ is satisfied. Since $u \in \mathbf{V}$ is a random node, we can infer that $\mathbf{C}_{n\times 1} \leq \mathbf{RS}_k^2(:,q) \leq \mathbf{1}_{n\times 1}$. □

Theorem 5.6. ***Triangle inequality:*** *For any nodes $(a,b) \in \mathbf{V}$, the role similarity scores satisfy the following inequality: $d_k(a,b) \leq d_k(a,q) + d_k(b,q)$ where $d_k(a,q) = 1 - RS_k^{\Gamma}(a,q)$.*

Proof. The proof of Theorem 5.6 employs the mathematical induction method. Since $d_k(a,q) = 1 - RS_k^2(a,q)$, Eq. (6) can be rewritten as follows:

$$d_k(a,q) + d_k(b,q) \leq d_k(a,b)$$
$$\Updownarrow \quad d_k(a,q) = 1 - RS_k^2(a,q)$$
$$1 - RS_k^2(a,q) + 1 - RS_k^2(b,q) - 1 + RS_k^2(a,b) \leq 0$$
$$\Updownarrow$$
$$RS_k^2(a,q) + RS_k^2(b,q) - RS_k^2(a,b) \leq 1 \tag{7}$$

To ensure that Eq. 7 is satisfied, we first initialise the role similarity scores at the iteration $k = 0$, as $RS_0^2 = \mathbf{Ones}_{n\times n}$, where n denotes the number of nodes in the graph. At iteration $k = 0$, Eq. 7 can be written as: $RS_0^2(a,q) + RS_0^2(b,q) - RS_0^2(a,b) = 1 + 1 - 1 \leq 1$. Thus, at iteration $k = 0$, Eq.7 holds. Next, assuming that Eq.7 is satisfied at iteration k, we need to prove that it also holds at the $k+1$ iteration.

$$RS_{k+1}^2(a,q) + RS_{k+1}^2(b,q) - RS_{k+1}^2(a,b)$$
$$= \frac{(1-C)}{(1+\lambda)} \left(\underbrace{\left(\frac{\sum_{(x,y)\in M_k^1(a,q)} RS_k^2(x,y)}{\max\deg^-(a,q)} + \frac{\sum_{(y,z)\in M_k^1(b,q)} RS_k^2(y,z)}{\max\deg^-(b,q)} - \frac{\sum_{(x,z)\in M_k^1(a,b)} RS_k^2(x,z)}{\max\deg^-(a,b)} \right)}_{\beta 1} + \lambda \underbrace{\left(\frac{\sum_{(x',y')\in M_k^2(a,q)} RS_k^2(x',y')}{\max\deg^-(a,q)} + \frac{\sum_{(y',z')\in M_k^2(b,q)} RS_k^2(y',z')}{\max\deg^-(b,q)} - \frac{\sum_{(x',z')\in M_k^2(a,b)} RS_k^2(x',z')}{\max\deg^-(a,b)} \right)}_{\beta 2} \right) + C$$

For general purposes, we assume $deg_a^- \leq deg_q^- \leq deg_b^-$. The equation β1 can be generated as follows:

$$\beta 1 = \left(\frac{1}{deg_q^-} - \frac{1}{deg_b^-}\right) \sum_{(x,y)\in M_k^1(a,q)} RS_k^2(x,y)$$
$$+ \frac{1}{deg_b^-} \left(\sum_{(x,y)\in M_k^1(a,q)} RS_k^2(x,y) + \sum_{(y,z)\in M_k^1(b,q)} RS_k^2(y,z) - \sum_{(x,z)\in M_k^1(a,b)} RS_k^2(x,z) \right)$$

Then we define a matching $\mathcal{MA}_k^1(b,q) = \{(y,z)|(x,y) \in M_k^1(a,q) \bigwedge (x,z) \in M_k^1(a,b)\}$. $I(q)$ can be devided into two parts $I(q1)$ and $I(q2)$, where $I(q1)^- = \{y|y \in M_k^1(a,q)\}$ and $I(q2) = I(q) - I(q1)$. So $\mathcal{MA}_k^1(b,q) = \mathcal{MAI}_k^1(b,q) + \mathcal{MAII}_k^1(b,q)$, where $\mathcal{MAI}_k^1(b,q) = \{(y,z)|y \in I(q1), z \in I(b)\}$, and $\mathcal{MAII}_k^1(b,q) = \{(y,z)|y \in I(q2), z \in I(b)\}$. According to the Lemma 5.1, we have:

$$\beta 1 \leq \left(\frac{1}{deg_q^-} - \frac{1}{deg_b^-}\right) deg_a^- + \frac{1}{deg_b^-} \left(\sum_{(x,y)\in M_k^1(a,q)} RS_k^2(x,y) + \sum_{(y,z)\in \mathcal{MAI}_k^1(b,q)} RS_k^2(y,z) \right.$$
$$\left. + \sum_{(y,z)\in \mathcal{MAII}_k^1(b,q)} RS_k^2(y,z) - \sum_{(x,z)\in M_k^1(a,b)} RS_k^2(x,z) \right)$$
$$\leq \left(\frac{1}{deg_q^-} - \frac{1}{deg_b^-}\right) deg_a^- + \frac{1}{deg_b^-} (deg_a^- + deg_a^- + deg_q^- - deg_a^- - deg_a^-) \leq 1$$

β2 has the similar trend as β1, which is $\beta 2 \leq 1$. $RS_{k+1}^2(a,q) + RS_{k+1}^2(b,q) - RS_{k+1}^2(a,b) \leq \frac{(1-C)}{(1+\lambda)}(1+\lambda) + C \leq 1$ The proof shows that $RS_{k+1}^2(a,q) + RS_{k+1}^2(b,q) - RS_{k+1}^2(a,b) \leq 1$, which implies $RS_k^2(a,q) + RS_k^2(b,q) - RS_k^2(a,b) \leq 1$ holds with random choose k. This in turn implies $d^k(a,q) + d^k(b,q) \leq d^k(a,b)$. □

6 COMPUTATION OPTIMIZATION

This section introduces two techniques aimed at accelerating the computation of FaRS. These techniques effectively reduce the number of calls made to the maximum matching algorithm and exploit "shared" information to minimize repetitive operations. The resulting accelerated algorithm is referred to as Opt_FaRS.

6.1 Pruning Approach

The methodology of Opt_FaRS comprises two stages: the pre-processing phase and the iterative computation phase. In the pre-processing phase, the algorithm involves extracting the `tracking path` and computing the `candidate pool`.

Definition 2 (Multi-Hop Backward Tracking Path). *Given a connected graph $G = (\mathbf{V}, \mathbf{E})$, a query $q \in \mathbf{V}$, and the number of total iterations K defined in FaRS, the tracking path P with respect to query q is denoted as $\boldsymbol{P(q)} = <\boldsymbol{p}^1, \boldsymbol{p}^2 \ldots, \boldsymbol{p}^L>$, where p^i represents the set of i^{th}-hop backward tracking nodes with respect to query q, and L is the actual number of iterations performed by the FaRS algorithm before convergence, which is also known as the level of the tracking path. It satisfies the condition $1 \leq L \leq K$. $\boldsymbol{P}$ is iteratively defined as follows and any repeated nodes in $\boldsymbol{p}^l$ are e removed from the set to ensure uniqueness:*

$$\begin{cases} \boldsymbol{p}^1 = \{q\} \\ \boldsymbol{p}^l = I(x^1) \bigcup I(x^2) \cdots \bigcup I(x^{|\boldsymbol{p}^{l-1}|}) where\ x^1, x^2, \ldots, x^{|\boldsymbol{p}^{l-1}|} \in \boldsymbol{p}^{l-1} \end{cases} \tag{8}$$

The tracking path is determined by the query node and the structure of the graph. This calculation can be illustrated using the following example.

Example 6.1. *Consider a graph G with five nodes, a query $q = d$, and the number of iterations $K = 6$ in FaRS, as shown in the left side of Figure 4. The tracking paths generated according to Eq.* (8) *are depicted on the right side of Figure 4.*

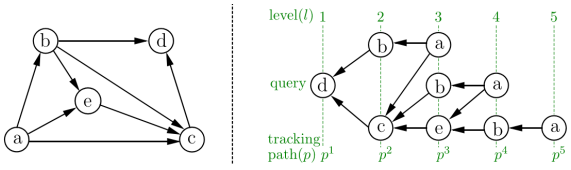


Figure 4: Left side: Example of Graph G. Right side: Multi-Hop Backward Tracking Path of the Graph G.

In Figure 4, the track path represents a traverse starting from the query node d. As per Definition 2, the second element in the track path represents the in-neighbour nodes of the query node d; thus, we obtain $\boldsymbol{p}^2 = \{b,c\}$. To generate $\boldsymbol{p}^3$, we apply Eq.(8)

and obtain the following: $\boldsymbol{p}^3 = I(\boldsymbol{p}^2) = I(b) \cup I(c) = \{a,a,b,c\}$. *After eliminating the repeated elements, the final result is* $\boldsymbol{p}^3 = \{a,b,c\}$. $\boldsymbol{p}^4$ *and* $\boldsymbol{p}^5$ *can be calculated in a similar way. It is worth mentioning that the tracking path terminated at* $\boldsymbol{p}^5 = a$ *and where a has an in-degree of zero, ensuring convergence.*

The objectives of extracting the tracking path $\mathbf{P}$ of a graph G in relation to q are twofold: firstly, it can significantly reduce the computation of redundant information; secondly, it can decrease the number of iterations required for FaRS. When the level number L of the tracking path is less than the given iteration number K, it suffices to perform only L times iterations. This is because, based on the structure of the graph, the role similarity scores converge after L iterations, implying that $Fa\mathbf{RS}_k^\Gamma = Fa\mathbf{RS}_L^\Gamma$ $(L \le k \le K)$.

Given the definition of the graph's tracking path, we can explain how to generate candidate pools (**CP**) by using the tracking path elements as indices.

Definition 3 (Candidate pool). *Given* $\boldsymbol{P} = < \boldsymbol{p}^1, \boldsymbol{p}^2 \ldots, \boldsymbol{p}^L >$ *as a track path of a connected graph G, a candidate pool is defined for each* $\boldsymbol{p}^l$ *in* $\boldsymbol{P}$. *Each candidate pool is a subset of the similarity matrix, consisting of a fixed number of rows that include all the nodes in G with out-neighbors. These nodes are denoted as* **out**, *defined as* $\mathbf{out} = \{x \in \mathbf{V} | O(x) \neq \emptyset\}$. *The columns of each candidate pool are determined by the corresponding tracking path element* $\boldsymbol{p}^l$. *During the kth iteration of the FaRS algorithm, the candidate pool* $\mathbf{CP}_k^\Gamma$ *can be represented as:* $\mathbf{CP}_k^\Gamma = \mathbf{RS}_k^\Gamma(\mathbf{out}, \boldsymbol{p}^l) \quad (l = K-k+1, \quad 1 < k \le L)$.

Here, $\mathbf{RS}_k^\Gamma$ *denotes the role similarity score matrix of all node pairs in the graph generated by the FaRS algorithm, and* $\mathbf{CP}_k^\Gamma$ *is the candidate pool for the kth iteration.*

It can be observed that during the computation of $\mathbf{RS}_k^\Gamma$, only those similarity values from $\mathbf{CP}_k^\Gamma$ would be updated at each iteration. The size of $\mathbf{CP}_k^\Gamma$ is typically much smaller than $\mathbf{RS}_k^\Gamma$ because the number of nodes with out-neighbors (i.e., the candidate pool's row) is smaller or equal to the total number of nodes in the graph, and the length of each element in the track path (i.e., the candidate pool's column) is much smaller than the total number of nodes in the graph (i.e., $|\mathbf{out}| \le n$ and $|\mathbf{p}^l| \ll n$, where n is the total number of nodes in the graph). In previous studies on RoleSim, all node pairs' $(n \times n)$ role similarity scores had to be computed at each iteration. In contrast, our proposed Opt_FaRS algorithm leverages the candidate pool concept to reduce the computation cost of each iteration to the information retrieval range $(|\mathbf{out}| \times |\mathbf{p}^l|)$.

Based on the computation formula of FaRS (Eq. 6) and the candidate pool definition, we propose an efficient single-source similarity search algorithm called Opt_FaRS, which can be expressed mathematically as follows:

Theorem 6.1. *Let* $G = (\mathbf{V}, \mathbf{E})$ *be a connected graph, and let q be a random query with corresponding track path* $\boldsymbol{P} = < \boldsymbol{p}^1, \boldsymbol{p}^2 \ldots, \boldsymbol{p}^L >$. *Then, the candidate pool at iteration k can be updated as follows:*

$$\begin{aligned} &\mathbf{RS}_k^\Gamma(\mathbf{out}, j) = \mathbf{CP}_k^\Gamma(:, j) = \\ &(1-C)(\boldsymbol{M}_{k-1}^1(:, j) + \lambda \boldsymbol{M}_{k-1}^2(:, j) + \cdots + \lambda^{\Gamma-1} \boldsymbol{M}_{k-1}^\Gamma(:, j)) \\ &\oslash (1 + \lambda + \cdots + \lambda^{\Gamma-1}) \mathbf{MAXDEG}^-(:, j) + C_{|\mathbf{out}| \times 1} \qquad j \in \boldsymbol{p}^{L-k+1} \end{aligned} \tag{9}$$

Here, j represents a node in track path $\boldsymbol{p}^{L-k+1}$. *The track path* $\boldsymbol{p}^{L-k+1}$ *determines the column index of the candidate pool, so the candidate pool at iteration k is given by* $\mathbf{RS}_k^\Gamma(\mathbf{out}, \boldsymbol{p}^{L-k+1}) = \mathbf{CP}_k^\Gamma$. *The vector* $\mathbf{MAXDEG}^-(:, j)$ *represents the maximum in-degree between node j and each node in* **out**, *respectively. The vector* $\boldsymbol{M}_{k-1}^\Gamma(:, j)$ *represents the top* Γ *maximum weighted matching of the node pair* (i, j) *in-neighbour similarity matrix, where* $i \in$ **out** *at iteration* $(k-1)$.

The Opt_FaRS algorithm comprises two key steps. Firstly, the algorithm retrieves the tracking path $\mathbf{P}$ of the connected graph G starting from the query node q. Secondly, the algorithm generates the candidate pool for the tracking path at iteration k. Finally, the role similarity scores with respect to the query node q are computed as follows: $\text{Opt\_FaRS}(:, q) = \mathbf{RS}_K^\Gamma(:, q) = \mathbf{CP}_K^\Gamma$. Here, the size of $\mathbf{CP}_{K-1}^\Gamma$ is $|\mathbf{out}| \times \mathbf{p}^2$, and the size of $\mathbf{CP}_K^\Gamma$ is $n \times 1$ $(|\mathbf{out}| \le n)$. Next, we set the value of the difference index between n and **out** to C to ensure that the sizes of $\mathbf{CP}_K^\Gamma$ and $\mathbf{RS}_K^\Gamma(:, q)$ are consistent. It is important to note that when $\mathbf{MAXDEG}^-(\mathbf{out}, j) = 0$, the result of Eq. 9 is equal to $\mathbf{C}_{\mathbf{out} \times 1}$. The proof of this method is omitted here to save space.

In addition to Theorem 6.1, in the next, we present exceptional cases that contribute positively to the speed-up of the Opt_FaRS algorithm. We introduce two speed-up approaches to reduce the computational complexity of Opt_FaRS for candidate pool computation. One approach optimizes column generation, while the other optimizes row generation.

6.2 P-Speedup Approach

Based on Eq. 9, the column indexes of candidate pools are determined by the corresponding track path. Therefore, we have named our speed-up approach on column generation the **P-Speedup** Approach.

There are two exceptional cases of the Fast Role Selection (FRS) algorithm, where FRS can retrieve

the role similarity scores without the need to use the maximum matching algorithm.

Definition 4 (**Exceptional cases**). *Given a graph $G(\mathbf{V},\mathbf{E})$, we define two special sets of nodes:*

- ***One-hop.*** *This set consists of all the nodes in G with in-degrees equal to zero. We denote this set as $\mathbf{V}^{(1)}$, and it can be formally defined as: $V^{(1)} = \{i \mid deg_i^- = 0, i \in V\}$.*
- ***Two-hop.*** *This includes all the nodes in G where all their in-neighbors have in-degrees equal to zero. We define the two-hop node set (denoted as $\boldsymbol{V}^{(2)}$) mathematically as: $V^{(2)} = \{i \mid deg^-_{I(i)} = 0, i \in V\}$, where $I(i)$ is the set of in-neighbours of node i in the graph.*

We observe that in the presence of exceptional cases in a graph node, such as belonging to the one-hop set or two-hop set, the candidate pool value can be generated directly.

Lemma 6.2. *Given a graph G is a connected graph, and let j be a query column ($j \in \boldsymbol{p}^k$).*

- *If node j belongs to the one-hop set $\mathbf{V}^{(1)}$, then the candidate pool value $\mathbf{CP}_k^{\Gamma}(:,j)$ can be generated as follows: $\mathbf{CP}_k^{\Gamma}(:,j) = \mathbf{C}_{\mathbf{out}\times 1}$, where $\mathbf{C}_{\mathbf{out}\times 1}$ is a column vector (with length equal to the number of $|\mathbf{out}|$) containing Cs in all entries. Note that the role similarity scores of $\mathbf{CP}_k^{\Gamma}(:,j)$ will not change in the subsequent iteration.*
- *If node j belongs to $\mathbf{V}^{(2)}$, then the candidate pool value $\mathbf{CP}_k^{\Gamma}(:,j)$ can be generated as follows: $\mathbf{CP}_k^{\Gamma}(:,j) = (1-C)(C \cdot \mathbf{MINDEG}^-(\mathbf{out},j) \oslash \mathbf{MAXDEG}^-(\mathbf{out},j) + \mathbf{1}_{\mathbf{out}\times 1})$. Here, $\mathbf{1}_{\mathbf{out}\times 1}$ is a column vector (with size equal to $|\mathbf{out}|$, containing all 1s. In this exceptional case, the role similarity scores converge at iteration $k = 2$.*

6.3 Out-Speedup Approach

Lemma 6.2 indicates that optimizing the computation of $\mathbf{CP}^{\Gamma}k$ involves considering column index nodes that belong to exceptional cases. In this subsection, we further enhance the computation of $\mathbf{CP}^{\Gamma}k$ by focusing on specific rows of the $\mathbf{CP}_k^{\Gamma}$ matrix. The row indices for each candidate pool are determined by **out**, and thus we refer to the method of accelerating the row nodes as the **out-speedup** approach. This optimization is based on the observation that it is unnecessary to compute a similarity score using the computationally expensive maximum weighted matching algorithm if a node can reach a root node of the graph (a node with no incoming edges) within two hops of traversal. This observation enables us to propose an optimization strategy that accelerates the computation of FaRS by avoiding unnecessary calls to the maximum weighted matching algorithm for certain node pairs, as described in Lemma 6.3.

Lemma 6.3. *Given a graph $G(\mathbf{V},\mathbf{E})$, an iteration number k, a query column j ($j \in \boldsymbol{p}^k$), and any node i($i \in \mathbf{out}$), we have the following:*

1. *If node i belongs to the one-hop set $\mathbf{V}^{(1)}$, then $\mathcal{M}_k^{\Gamma}(i,j) = 0$.*
2. *If node i belongs to the two-hop set $\mathbf{V}^{(2)}$, then $\mathcal{M}_k^{\Gamma}(i,j) = C$.*
3. *Otherwise, the maximum weighted matching of $\mathbf{CP}_{k-1}^{\Gamma}$ is generated. Before introducing the computation method, we define several notions. The maximum matching result of $\mathbf{CP}_{k-1}^{\Gamma}$ is denoted as $\mathcal{M}[\mathbf{CP}_{k-1}^{\Gamma}]$, and the matched set of the maximum matching on $\mathbf{CP}_{k-1}^{\Gamma}$ is represented by $\widetilde{\mathcal{M}}[\mathbf{CP}_{k-1}^{\Gamma}]$. The in-neighbor similarity matrix of the node pair (i,j) is defined as $\mathcal{B}ij$. The maximum matching result of $\mathcal{B}ij$ is denoted as $\mathcal{M}[\mathcal{B}ij]$, and the matched set of the maximum matching on $\mathcal{B}ij$ is denoted as $\widetilde{\mathcal{M}}[\mathcal{B}ij]$. The matched values of $\mathbf{CP}_{k-1}^{\Gamma}$ in $\mathcal{B}ij$ are defined as $\mathcal{M}[\mathcal{B}_{ij}, \mathbf{CP}_{k-1}^{\Gamma}]$, and the number of matched values of $\mathbf{CP}_{k-1}^{\Gamma}$ in $\mathcal{B}_{ij}$ is denoted as $|\mathcal{M}[\mathcal{B}_{ij}, \mathbf{CP}_{k-1}^{\Gamma}]|$.*
   - *If the minimum value between deg_i^- and deg_j^- is equal to $|\mathcal{M}[\mathcal{B}_{ij}, \mathbf{CP}_{k-1}^{\Gamma}]|$, then we can assert that $\mathcal{M}_k^{\Gamma}(i,j) = \mathcal{M}[\mathbf{CP}_{k-1}^{\Gamma}]$.*
   - *If the minimum value between deg_i^- and deg_j^- exceeds $|\mathcal{M}[\mathcal{B}_{ij}, \mathbf{CP}_{k-1}^{\Gamma}]|$, then we can decompose this into two cases:*
     - *The matched values in $|\mathcal{M}[\mathcal{B}_{ij}, \mathbf{CP}_{k-1}^{\Gamma}]|$ are the maximum value of both the column and the row of the bipartite matrix $\mathcal{B}_{ij}$. We can then calculate the value of $\mathcal{M}_k^1(i,j)$ by $\mathcal{M}_k^1(i,j) = sum(\mathcal{M}[\mathcal{B}_{ij}, \mathbf{CP}_{k-1}^{\Gamma}]) + \mathcal{M}(\mathcal{B}_{ij}^{rem})$. Here, the operation sum denotes the sum of the matched values between the bipartite graph $\mathcal{B}_{ij}$ and $\mathcal{M}[\boldsymbol{CP}_{k-1}^{\Gamma}]$. We then eliminate the matched elements' rows and columns from the bipartite graph, which we define as $\mathcal{B}_{ij}^{rem}$. The value $\mathcal{M}(\mathcal{B}_{ij}^{rem})$ represents the maximum matching score of the remaining bipartite graph.*
     - *otherwise, the value of $\mathcal{M}_k^1(i,j)$ is as follows: $\mathcal{M}_k^1(i,j) = \mathcal{M}[\mathcal{B}_{ij}]$, where $\mathcal{M}[\mathcal{B}_{ij}]$ is the maximum matching value of the in-neighbour similarity matrix $\mathcal{B}_{ij}$.*

In summary, our research has revealed that not all node pairs in each iteration have a significant impact on the final role similarity scores of column q. Building upon this insight, the Opt_FaRS algorithm efficiently extracts the influential information

during each iteration, eliminating the calculation of unnecessary scores. These optimization techniques greatly reduce the number of calls to the maximum matching algorithm, which is particularly beneficial for large graphs where this algorithm can be computationally expensive. Furthermore, Opt_FaRS captures and reuses the "shared" information, avoiding redundant computations. The key advantage of the Opt_FaRS algorithm is its significant improvement in computational efficiency without compromising accuracy.

7 EXPERIMENTAL EVALUATION

We empirically evaluate the performance of our proposed algorithms, FaRS and Opt_FaRS, on real-world datasets. We compare them with baseline algorithms and assess their efficiency using three metrics: the impact of coefficient choices on FaRS accuracy, as well as accuracy and time efficiency.

7.1 Experimental Settings

Datasets. We evaluate our algorithms using the publicly available email-Eu-core-temporal dataset (EU) obtained from SNAP (https://snap.stanford.edu/index.html). This dataset consists of anonymized email data from a research organization, representing the network of incoming and outgoing emails between members. The dataset sizes are summarized in Table 1.

Table 1: Description of Datasets.

Datasets	#-Nodes	#-Edges	Type
email-Eu-core (EU)	986	24,929	Directed
Department 1 (Dept-1)	309	3,031	Directed
Department 2 (Dept-2)	162	1,772	Directed
Department 3 (Dept-3)	89	1,506	Directed
Department 4 (Dept-4)	142	1,375	Directed

Compared Algorithms. We evaluated FaRS and Opt_FaRS on the aforementioned real-life datasets, comparing them with three state-of-the-art similarity search competitors (CSR, RoleSim, and FaRS_N). CSR (Rothe and Schütze, 2014) calculates the CoSimRank score using the dot product of Personalized PageRank vectors. RoleSim (Lee, 2012) is a state-of-the-art role similarity search algorithm based on average maximum matching. FaRS_N is an alternative version of FaRS that computes the average maximum matching of the remaining in-neighbor similarity matrix instead of using the top Γ maximum matching.

Parameters. We use the following default parameters: (a) the decay factor $C = 0.2$; (b) the number of iterations $K = 5$; (c) the order of maximum matching $\Gamma = 3$; and (d) the relative weight $\lambda = 0.7$.

Evaluation Metrics. The evaluation of role similarity ranking on real-life datasets was performed using k-means clustering (Arthur and Vassilvitskii, 2006; Lloyd, 1982; Bock, 2007). To establish the ground truth, we initially computed the role similarity score matrix using our algorithms and other baseline algorithms. Subsequently, k-means clustering was applied to the various role similarity score matrices to group the data into multiple clusters. According to the inherent characteristics of k-means clustering, nodes within the same cluster exhibit higher role similarity scores. For each query, we extracted the top 20 nodes that displayed the greatest similarity to the query node in each algorithm. The cluster to which the query node belonged was identified through k-means clustering. Finally, we determined the number of nodes that were common between the top 20 nodes most similar to the query in each algorithm and the nodes within the query cluster. A higher overlap ratio signifies greater accuracy.

Experiments were carried out on a PC with an Intel Core i7-6700 3.40GHz CPU, 64GB of memory, and Windows 10. Each experiment was repeated five times, and the average results are reported.

7.2 Experimental Results

The experimental results consist of three parts: parameter evaluation, accuracy assessment, and time efficiency analysis.

Hyper-Parameter Evaluation. For demonstrate the effectiveness of the FaRS approach, we first apply it to a real-life dataset to determine the optimal values of two parameters, namely λ and Γ. We implement our algorithm on the Dept-3 dataset and evaluate its accuracy using different parameter values. To assess accuracy, we generate a role similarity score matrix for the graph and utilize the k-means clustering method.

Figure 5 shows the accuracy of the FaRS algorithm on the Dept-3 dataset for various λ values. The y-axis represents the accuracy ratio, calculated based on the number of duplicate nodes between two sets. One set consists of the top 20 nodes with the high-

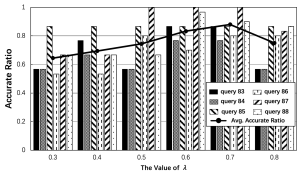


Figure 5: Hyper-Parameter Evaluation(λ).

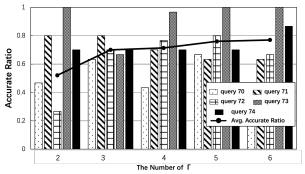


Figure 6: Hyper-Parameter Evaluation(Γ).

est role similarity scores to the query node, while the other set consists of nodes in the same k-means group as the query node. We selected six different λ values ($\lambda = [0.3, 0.4, 0.5, 0.6, 0.7, 0.8]$) and query nodes ($Q = [83, 84, 85, 86, 87, 88]$). Figure 5 shows that $\lambda = [0.6, 0.7]$ yield relatively high accuracy ratios, particularly for query 87. The line chart in Figure 5 represents the average accuracy ratio for different λ values. It can be observed the FaRS algorithm achieves the highest accuracy ratio when $\lambda = 0.7$. Similar results were obtained for other datasets, confirming the selection of $\lambda = 0.7$ for future experiments.

Similarly, Figure 6 presents the accuracy of the FaRS algorithm on the Dept-3 dataset for different Γ values. We selected five Γ values ($\Gamma = [2, 3, 4, 5, 6]$) and query nodes $Q = [70, 71, 72, 73, 74]$. The accuracy for each Γ value is the average of five trials due to the varying k-means clustering groups. The bar chart in Figure 6 displays the accuracy of different queries for each Γ value, while the line chart shows the average accuracy ratio for the five queries. The algorithm achieves improved accuracy as Γ increases, with a significant enhancement at $\Gamma = 3$. Beyond $\Gamma = 3$, the accuracy improvement becomes less pronounced. Hence, the optimal performance is achieved with $\Gamma = 3$, providing high accuracy with minimal time consumption.

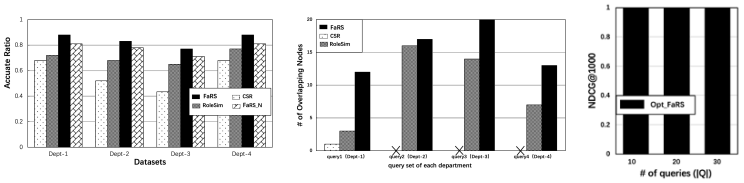
Accuracy Evaluation. The accuracy of the algorithms is assessed using two methods: k-means clustering and evaluation based on the network's characteristics. We evaluate the performance on the EU dataset, which represents the communication network within an organization. The dataset includes four departments: Department 1 (Dept-1), Department 2 (Dept-2), Department 3 (Dept-3), and Department 4 (Dept-4), and the remaining employees. Given the network structure, employees within the same department exhibit higher role similarity compared to those from different departments. Consequently, we divide the EU dataset into five partitions based on the departments and the remaining employees.

To evaluate the accuracy of the algorithms, we randomly select a node from each part of the dataset as a query and test the number of nodes in the corresponding department among the top 20 similar nodes to the query in different algorithms. This approach justifies our choice of datasets for evaluating the algorithms.

We evaluate the algorithms using k-means clustering on the mail exchange network of each department individually. With a partition number of $k = 6$ and a list of 20 ordered nodes for each query, we assess the accuracy ratios of the different algorithms on the four datasets. Figure 7a illustrates the results. We observe that the CSR algorithm exhibits relatively low accuracy in role similarity search on each dataset. The RoleSim algorithm achieves higher accuracy than CSR but falls short of the FaRS algorithm. Notably, the FaRS algorithm consistently outperforms all other algorithms in role similarity detection on the four datasets. The FaRS_N algorithm demonstrates better accuracy than RoleSim and CSR, but it does not match the performance of the FaRS algorithm.

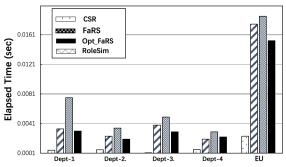
Next, we evaluate the accuracy of our algorithms and other baselines on the EU dataset. Four nodes are randomly selected from each department to form the query set, with each query corresponding to its respective department. The top 20 nodes are ranked, and we assess the highest role similarity scores of each algorithm with respect to the query. The number of nodes belonging to the query's department among the top 20 nodes is determined to measure the algorithm's accuracy. Figure 7b presents the results. The FaRS algorithm consistently achieves a high level of accuracy, followed by the RoleSim algorithm. Conversely, the CSR algorithm consistently exhibits lower accuracy in role similarity search. Considering the findings from Figure 7a and Figure 7b, we can conclude that the FaRS algorithm outperforms the well-known algorithms, CSR and RoleSim, on the five real-life datasets.

Lastly, we assess the accuracy of the Opt_FaRS algorithm on the EU dataset, which is an accelerated method based on FaRS. To compare its accuracy with that of FaRS, we focus on the Dept-4 dataset. Random query sets are selected, varying in size from 10 to 30. For each query set Q, we measure the similarity ranking results using Normalized Discounted Cumulative Gain (NDCG) (Wang et al., 2013) based on the role similarity scores obtained from Opt_FaRS. An NDCG score of 1 indicates that the results of the compared algorithm perfectly match those of FaRS, without any loss in accuracy. Figure 7c illustrates the NDCGs of



(a) FaRS(Dept Dataset). (b) FaRS(EU Dataset). (c) Opt_FaRS.

Figure 7: Accuracy Evaluation of FaRS & Opt_FaRS.



(a) Time Efficiency.

Figure 8: Efficiency.

Opt_FaRS for each query set Q, all of which are 1. This implies that achieves the same level of accuracy as FaRS, affirming the correctness of Lemma 6.3.

Efficiency Analysis. Figure 8a shows the efficiency of our proposed algorithms and baseline approaches across five authentic networks. The elapsed time results from computing the single-source role similarity scores for each query, where $|Q| = 20$ queries are randomly selected for each dataset. Our proposed FaRS and RoleSim algorithms exhibit comparable efficiency performance on all five datasets. While the CSR algorithm entails relatively lower time costs, its role classification accuracy is suboptimal. Notably, the Opt_FaRS algorithm demonstrates significant efficiency gains, outperforming both the FaRS and RoleSim algorithms across all five real-life datasets.

8 CONCLUSION

This paper presents a precise algorithm for single-source role similarity search, namely FaRS, which is based on graph topology. The FaRS algorithm is capable of capturing more information from the node-pair in-neighbour role similarity scores matrix than the RoleSim algorithm, thus ensuring greater accuracy. Additionally, we provide proofs for the convergence, uniqueness, symmetry, boundedness, and triangular inequality of the FaRS algorithm. Furthermore, we propose an accelerated algorithm, Opt_FaRS, based on FaRS to enable more efficient computation. Lastly, we evaluate our algorithms and compare them with baseline algorithms using five real datasets. The experimental results indicate that FaRS algorithm yields a more precise role similarity value compared to the baseline algorithms. Furthermore, Opt_FaRS algorithm significantly enhances the computation speed of FaRS algorithm without compromising its accuracy.

REFERENCES

Arthur, D. and Vassilvitskii, S. (2006). k-means++: The advantages of careful seeding. Technical report, Stanford.

Bock, H.-H. (2007). Clustering methods: a history of k-means algorithms. *Selected contributions in data analysis and classification*, pages 161–172.

Chen, X., Lai, L., Qin, L., and Lin, X. (2020). Structsim: Querying structural node similarity at billion scale. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 1950–1953. IEEE.

Chen, X., Lai, L., Qin, L., and Lin, X. (2021). Efficient structural node similarity computation on billion-scale graphs. *The VLDB Journal*, 30(3):471–493.

Diao, L., Wang, H., Alsarra, S., Yen, I.-L., and Bastani, F. (2019). A smart role mapping recommendation system. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, volume 2, pages 135–140. IEEE.

Everett, M. G. (1985). Role similarity and complexity in social networks. *Social Networks*, 7(4):353–359.

Gabow, H. N., Kaplan, H., and Tarjan, R. E. (2001). Unique maximum matching algorithms. *Journal of Algorithms*, 40(2):159–183.

Lee, V. E. (2012). *RoleSim and RoleMatch: Role-based similarity and graph matching*. Kent State University.

Li, L., Qian, L., Lee, V. E., Leng, M., Chen, M., and Chen, X. (2015). Fast and accurate computation of role similarity via vertex centrality. In *International Conference on Web-Age Information Management*, pages 123–134. Springer.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Murty, K. G. (1968). An algorithm for ranking all the assignments in order of increasing cost. *Operations research*, 16(3):682–687.

Rao, P. N., Devi, T., Kaladhar, D., Sridhar, G., and Rao, A. A. (2009). A probabilistic neural network approach for protein superfamily classification. *Journal of Theoretical & Applied Information Technology*, 6(1).

Rothe, S. and Schütze, H. (2014). Cosimrank: A flexible & efficient graph-theoretic similarity measure. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1392–1402.

Shahabi, C., Banaei-Kashani, F., Chen, Y.-S., and McLeod, D. (2001). Yoda: An accurate and scalable web-based recommendation system. In *International Conference on Cooperative Information Systems*, pages 418–432. Springer.

Shao, Y., Liu, J., Shi, S., Zhang, Y., and Cui, B. (2019). Fast de-anonymization of social networks with structural information. *Data Science and Engineering*, 4(1):76–92.

Wang, Y., Lian, X., and Chen, L. (2018). Efficient simrank tracking in dynamic graphs. In *2018 IEEE 34th international conference on data engineering (ICDE)*, pages 545–556. IEEE.

Wang, Y., Wang, L., Li, Y., He, D., and Liu, T.-Y. (2013). A theoretical analysis of ndcg type ranking measures. In *Conference on learning theory*, pages 25–54. PMLR.

Yang, R. (2022). Efficient and effective similarity search over bipartite graphs. In *Proceedings of the ACM Web Conference 2022*, pages 308–318.