A Unified Relationship Graph for Explainable Multi-Hop Connection Discovery

Sanika Shailesh Mhatre[1], Avinash Santosh Nikam[2], Aditya Janardhan Gaikwad[3]
[1,2,3]*Pillai College of Arts Commerce and Science Autonomous*

I. RESEARCH CLASSIFICATION FROM THE SYLLABUS

The supplied course material distinguishes descriptive, analytical and empirical research. This paper combines all three, but its primary identity is analytical and empirical: relationships and graph structure are analyzed, while algorithmic behavior is measured through repeatable experiments. The descriptive component is used to characterize relationship categories and the proposed network. [S1] People maintain relationships across family, friendship, education, neighbourhoods and work, yet these connections are rarely stored in one structure. A person may know a relative, a friend, a former teacher, a neighbour or a colleague but still be unable to see how that network connects to a specific individual or organization. This paper proposes a unified, user-controlled heterogeneous relationship graph in which people, organizations and other relevant entities are represented as nodes and typed relationships are represented as edges. The central query is simple: given a source person and a target person or organization, can the system discover and explain a valid multi-hop connection between them? Bidirectional Breadth-First Search (Bi-BFS) is studied as the baseline path-discovery technique because the source and target are both known before the search begins. A relationship interpreter then converts the discovered edge sequence into a human-readable explanation, while the interface provides graph navigation, profile popups and living/deceased status handling subject to permission rules. A controlled benchmark using synthetic sparse graphs from 100 to 50,000 nodes found lower median runtime and substantially fewer expanded nodes for Bi-BFS than one-sided BFS under the tested conditions. The work does not claim that shortest paths are always the most useful real-world connections; instead, it establishes a research framework for cross-domain connectivity, explainability and future relationship-aware search.

Keywords: heterogeneous relationship graph, family network, social network, professional network, graph search, Bidirectional BFS, relationship inference, multi-hop connection, graph database, explainability, privacy.

A traditional family tree usually begins with a main person and expands upward to parents and grandparents or sideways to siblings, uncles, aunts and cousins. That structure is useful, but everyday social life is larger than a family tree. A person can simultaneously be a child, sibling, spouse, friend, neighbour, student, mentor, employee, alumnus and colleague. Each of these roles creates another set of connections, and a useful path to a distant person may cross several of them.

Modern digital platforms demonstrate that relationship structure can be used to make discovery easier. LinkedIn organizes users into first-, second- and third-degree connections and provides network-building features. FamilySearch allows users to view relationships within a family tree and provides different privacy mechanisms for living people. Relationship-intelligence products such as Affinity explicitly describe finding warm introduction paths using relationship signals such as strength, recency and shared context. [2][3][4]

These examples motivate a broader research question. Instead of keeping family, social, educational and professional information in isolated structures, what would happen if a person controlled one graph containing all permitted relationships and could ask for a path to another person, a company, a university or another target? The answer would not be a promise of employment, acceptance or a successful

introduction. It would be an explainable map of the relationships that connect the two endpoints.

1.1 Motivation

Imagine a student searching for a role at a company where they know no employee personally. A possible route may be student -> friend -> friend's sibling -> employee -> company. Another route may be student -> professor -> former student -> employee -> company. A graph makes both possibilities visible. The user can then decide whether to pursue any of them through a voluntary introduction.

1.2 Problem Statement

The problem is to represent heterogeneous personal relationships in one connected graph and to discover explainable multi-hop paths between a source and a target while accounting for incomplete information, relationship semantics and access permissions. The research does not assume that every available relationship is public, current or equally strong.

1.3 Objectives

- Create a unified graph model covering family, social, educational and professional relationships.
- Represent entities and relationships with explicit types and relevant properties.
- Evaluate Bidirectional BFS against conventional BFS for two-terminal shortest-path queries.
- Translate path sequences into human-readable explanations.
- Design profile views for living and deceased people with permission-aware information fields.
- Measure runtime and search effort on controlled synthetic graphs.
- Identify research directions for weighted search, temporal data, entity resolution and privacy-preserving traversal.

1.4 Research Questions

- RQ1: Can heterogeneous personal relationships be represented in one graph without losing semantic meaning?
- RQ2: Can Bidirectional BFS reduce search effort for two-terminal relationship queries?
- RQ3: Can a discovered path be explained clearly using relationship labels at every step?
- RQ4: How do graph size and structure influence runtime and node exploration?
- RQ5: How should privacy, incomplete information and conflicting relationships be incorporated into a practical design?

1.5 Scope and non-claims

The proposed system is a research framework, not a claim that private social-media or messaging data can be imported freely. The paper does not scrape personal accounts and does not infer a hiring outcome. A discovered path represents structural connectivity only. Human decisions, introductions and opportunities remain outside the algorithm.

II. FAMILYSEARCH AND GENEALOGICAL RELATIONSHIP DISCOVERY

FamilySearch provides a strong example of relationship-centric information management. Its Family Tree allows people to work with genealogical relationships, and its relationship-viewing feature can show how users are related when relationship viewing is enabled. The platform also distinguishes living people from deceased people and provides private-tree and family-group options for sharing information about living relatives. [1][2]

For this research, the important lesson is not that the proposed system should reproduce FamilySearch. Instead, FamilySearch shows that relationship discovery can be a direct user feature and that living-person privacy must be modeled alongside the graph.

2.1 LinkedIn and Degree-Based Networking

LinkedIn defines first-degree connections as direct connections and second-degree connections as people connected to a user through a first-degree contact. The platform also supports connection suggestions and network expansion. [3] Graph distance therefore has a familiar interpretation for users: one hop, two hops, three hops and so on. The proposed framework generalizes this idea beyond professional contacts.

2.2 Relationship Intelligence

Affinity describes relationship intelligence around the discovery of warm paths into target organizations and emphasizes relationship strength, recency and shared context. [4] This is important because it demonstrates that shortest path alone may not be sufficient in practical systems. The proposed paper therefore treats

plain Bi-BFS as an experimental baseline and leaves relationship-aware ranking for later work.

2.3 Heterogeneous Information Networks

Research on heterogeneous information networks shows that real systems can contain multiple object types and multiple link types. Surveys describe applications in social, academic, recommendation and other domains and emphasize that typed structure carries useful semantic information. [5][6] The proposed network is therefore modeled as heterogeneous rather than as a single undifferentiated friend graph.

2.4 Graph Databases

Property-graph systems represent entities as nodes and connections as relationships, with properties attached to both. Neo4j documentation describes this representation as a natural fit for richly connected data and graph traversal. [7] The paper does not lock the final implementation to one database, but a graph database is a natural design choice for a path-centric application.

2.5 Search Research

Bidirectional search has a long history in shortest-path research. Pohl's work in Machine Intelligence described bidirectional search for two-terminal path problems, and recent research continues to study when Bidirectional BFS gains an advantage on real-world networks. [8][9] This literature supports using Bi-BFS as a baseline while also warning that the gain is data-dependent.

The review suggests that the basic building blocks are established: genealogy systems model family relationships, professional platforms expose connection degrees, relationship-intelligence systems search for introduction paths, heterogeneous-graph research studies multiple node and edge types, and bidirectional algorithms solve two-terminal path problems. The gap examined in this paper is therefore not “inventing a relationship graph.” It is investigating a unified, user-controlled framework in which heterogeneous personal relationships can be searched across domains and the resulting path can be explained to the user.

2.6 Proposed Contribution

- A unified representation for family, social, educational and professional relationship domains.
- A separation between graph traversal and relationship interpretation, enabling transparent explanations.
- A controlled empirical comparison of BFS and Bidirectional BFS for two-terminal queries.
- Explicit treatment of living/deceased profile states, permission boundaries and incomplete data.
- A UI design in which every person node can be selected to reveal biodata and the relationship path without leaving the graph context.

2.7 Hypotheses

H1: In the controlled sparse synthetic graphs used in this study, Bidirectional BFS will expand fewer nodes than conventional BFS for the tested source-target pairs. H2: Under the same conditions, the measured runtime advantage of Bidirectional BFS will generally increase as the graph becomes larger, although the magnitude of the benefit will depend on graph structure and implementation details.

2.8 Graph Representation

The core graph is written as G = (V, E), where V is the node set and E is the edge set. For a heterogeneous relationship graph, we extend the representation to G = (V, E, T, R), where T denotes node types and R denotes relationship types. This is aligned with standard heterogeneous-information-network formulations. [5][6]

2.9 Node and Edge Types

- Person, organization, college/university, place and event can be modeled as node types.
- Edges may represent parent_of, child_of, sibling_of, spouse_of, friend_of, neighbour_of, mentor_of, studied_at, alumnus_of, colleague_of and works_at.
- Node and edge properties can store permitted attributes such as dates, status, source, verification state and recency.

III. PATH, HOP AND RELATIONSHIP INTERPRETATION

A path P = (v0, v1, ..., vk) contains k edges. The value k is the hop count. For example, user -> friend -> sibling -> employee

-> company is a five-node path with four edges. The traversal algorithm returns the sequence; a separate interpreter turns that sequence into a readable explanation.

3.1 Privacy-Aware Graph Thinking

Relationship privacy is not identical to attribute privacy. Research on social-network data shows that the existence or structure of relationships can itself disclose information. [10]

[11] The proposed design therefore treats visibility and consent as graph-level constraints. A path should not be allowed to use an edge that the current query is not permitted to traverse.

The study combines descriptive, analytical and empirical elements. Descriptively, it categorizes relationship and node types. Analytically, it formalizes graph structure and compares algorithmic strategies. Empirically, it constructs controlled synthetic graphs, runs repeatable search queries and records quantitative measurements. This combination follows the research-methodology categories in the supplied course material. [S1]

IV. RESEARCH DESIGN

The benchmark graphs are synthetic because real family and employment relationships are private and would introduce unnecessary ethical and reproducibility problems. Each graph is connected and sparse. Connectivity is created through a random recursive tree; a controlled number of additional random edges increases network density without making the graph fully connected. For each size, node 0 is the source and the farthest reachable node is the target.

4.1 Data Controls

- Graph sizes: 100, 500, 1,000, 5,000, 10,000, 20,000 and 50,000 nodes.
- Additional edges: 0.25 per node in addition to the connected backbone.
- Search repetitions: five runs for each graph; median runtime is reported.
- Implementation: adjacency-list graph representation; the same runtime and graph are used for BFS and Bi-BFS.
- No real personal, contact, social-media or employment records are included in the experiment.

4.2 Evaluation Metrics

The study uses structural, efficiency and correctness measures so that the algorithmic comparison can be read from more than one angle. Structural measures describe graph size and connectivity; efficiency measures describe the computational effort needed to find a path; correctness measures describe whether the returned path agrees with the known synthetic graph.

Metric	Definition	Purpose
N = |V|	Number of nodes	Graph scale
M = |E|	Number of edges	Connectivity
k	Path length in edges	Hop distance
T	Elapsed search time	Runtime
X	Nodes expanded	Search effort
Speedup	T_BFS / T_BiBFS	Relative runtime
Reduction	(X_BFS-X_BiBFS)/X_BFS x 100	Search-effort reduction

4.3 Experimental Integrity

The benchmark results are intended to support the hypotheses only within the stated experimental conditions. The target-selection rule favors longer search, which is useful for exposing the value of two-ended traversal but is not a universal model of user behavior. Real relationship graphs may contain hubs, short paths, directionality, temporal changes and permission constraints that alter performance.

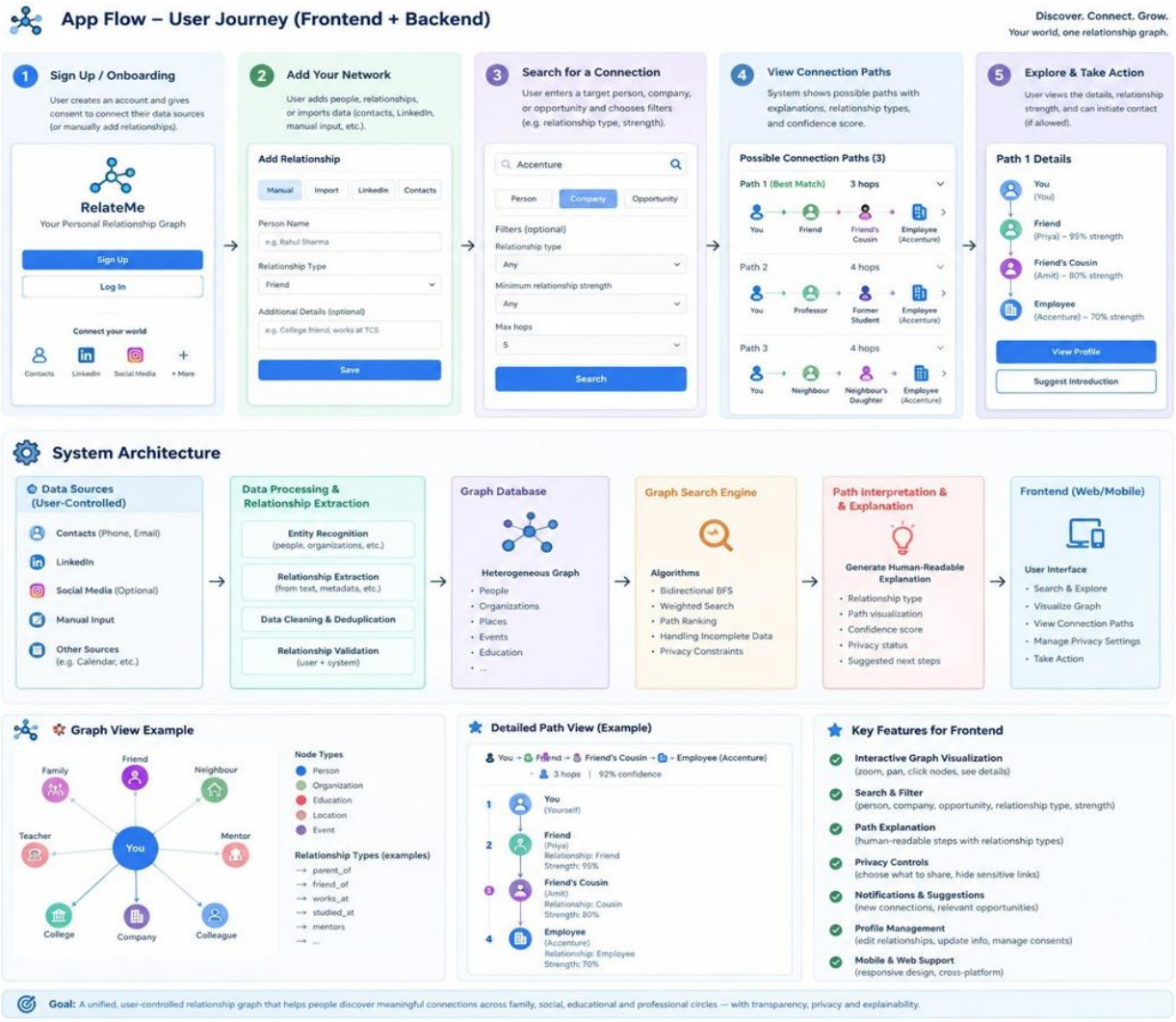
4.4 Controlled Comparison Principle

Both algorithms receive the same graph, the same source node, the same target node and the same adjacency-list representation. The only intentional difference is the traversal strategy. This control reduces confounding when interpreting runtime and search-effort differences. The measured values therefore describe the implementations under the stated benchmark rather than the performance of an entire production platform.

4.5 Ethical and Privacy Boundary

The proposed application is designed for permissioned relationship information. The paper does not advocate collecting private messages, hidden contact lists or employment records without consent. Synthetic

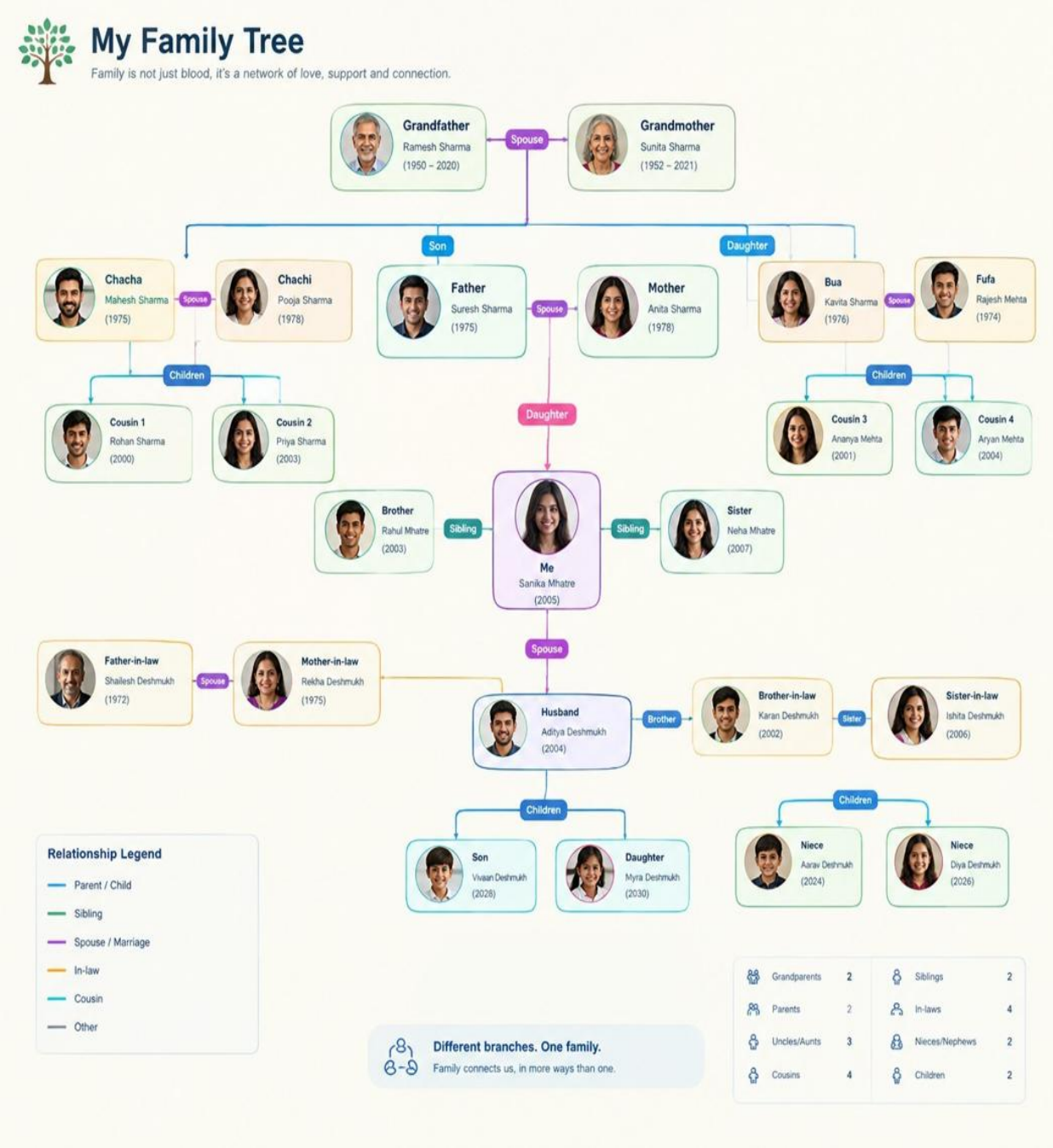
graphs are used for algorithm evaluation so that the research can be reproduced without exposing personal relationship data. A real deployment would require explicit consent, access control, data minimization and procedures for correcting or deleting relationship records.



Conceptual mock-up produced for this research; it describes the intended workflow and is not a deployed interface.

The system begins with user registration and permission settings, followed by relationship entry or other permitted data sources. Processing identifies people, organizations and relationship statements, removes duplicates and validates the records. The resulting graph is stored in a relationship-oriented database. A search engine then receives a source and target, executes path discovery, reconstructs the path and passes it to a relationship interpreter. The frontend finally renders the graph, path explanation and profile information.

The architecture deliberately separates three responsibilities: graph storage, path search and presentation. This separation is important because the same graph search engine can support a family question, a social question or a professional connection question without changing the underlying data model. Privacy checks can be applied before and during traversal so that a hidden edge is never treated as an available bridge.



Synthetic identities are used only to demonstrate how the family layer can be represented as a connected subgraph.

4.6 Family Subgraph

The family layer contains parents, children, siblings, grandparents, uncles, aunts, cousins, spouses and in-laws. Every person is linked by explicit edges, allowing the system to compute relationships without asking the user to mentally translate a long chain. A family graph can still be visualized in a tree-like layout for readability, even though the underlying model remains a graph.

This distinction matters when the graph expands outside the family. A person who appears in the family branch may also be connected to a school, a friend, a company or a neighbour. The same node can therefore participate in several contexts without being duplicated into separate trees.

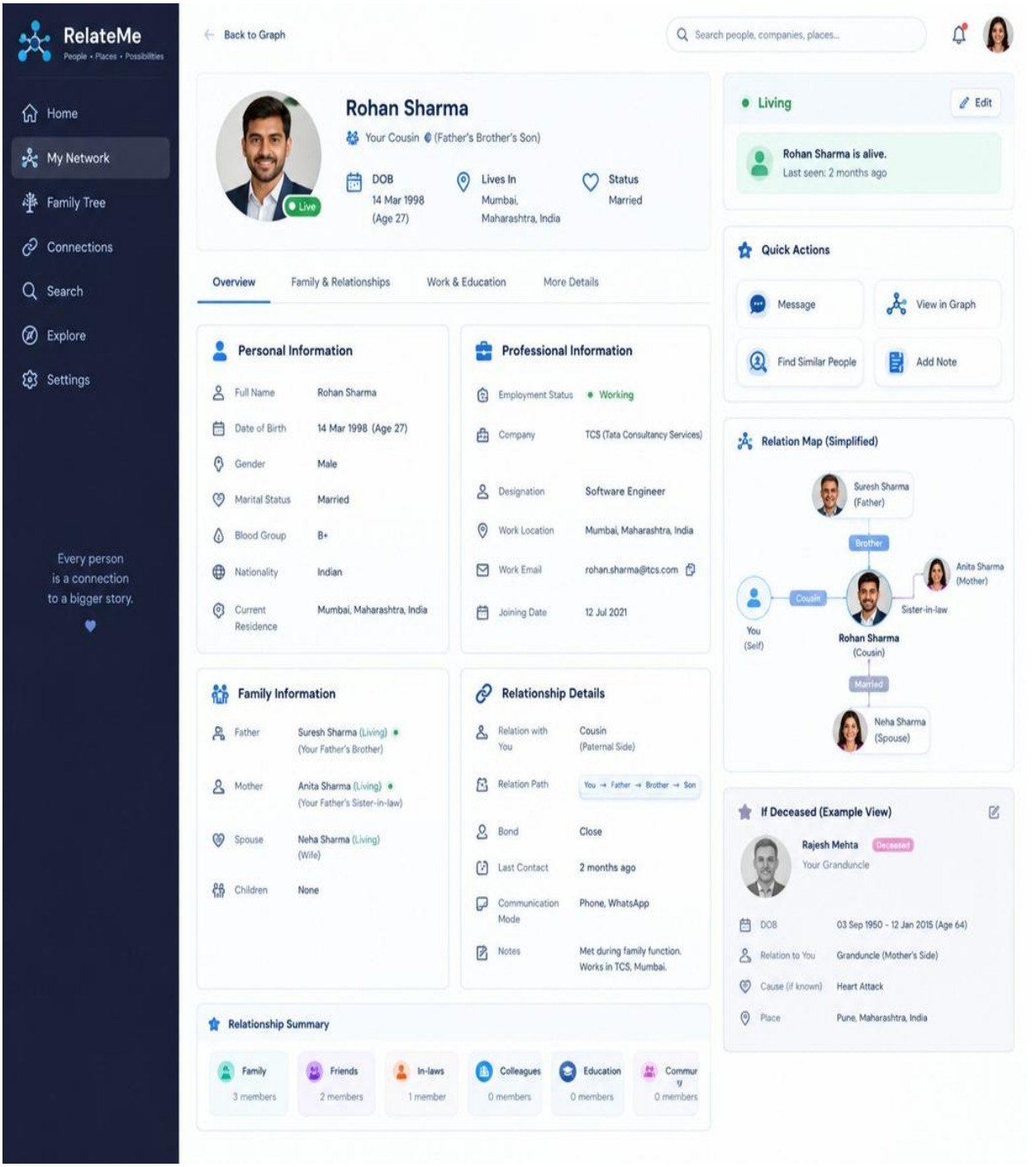


The figure illustrates the central research idea: multiple relationship domains coexist in one graph.

4.7 Unified Relationship Network

The cross-domain view places a user at the center of a larger personal network. Nodes can represent people and organizations; edges can represent family, friendship, neighbourhood, education or work. A target company can therefore be treated as a graph node, while an employee becomes an intermediate person node.

The same graph supports different queries. “Who is my cousin?” follows family semantics. “Who is connected to this company?” follows mixed semantics. “Can I reach a researcher through my academic network?” can focus on mentor, alumni and institutional edges. The search engine remains generic while the relationship interpreter changes the natural-language explanation.

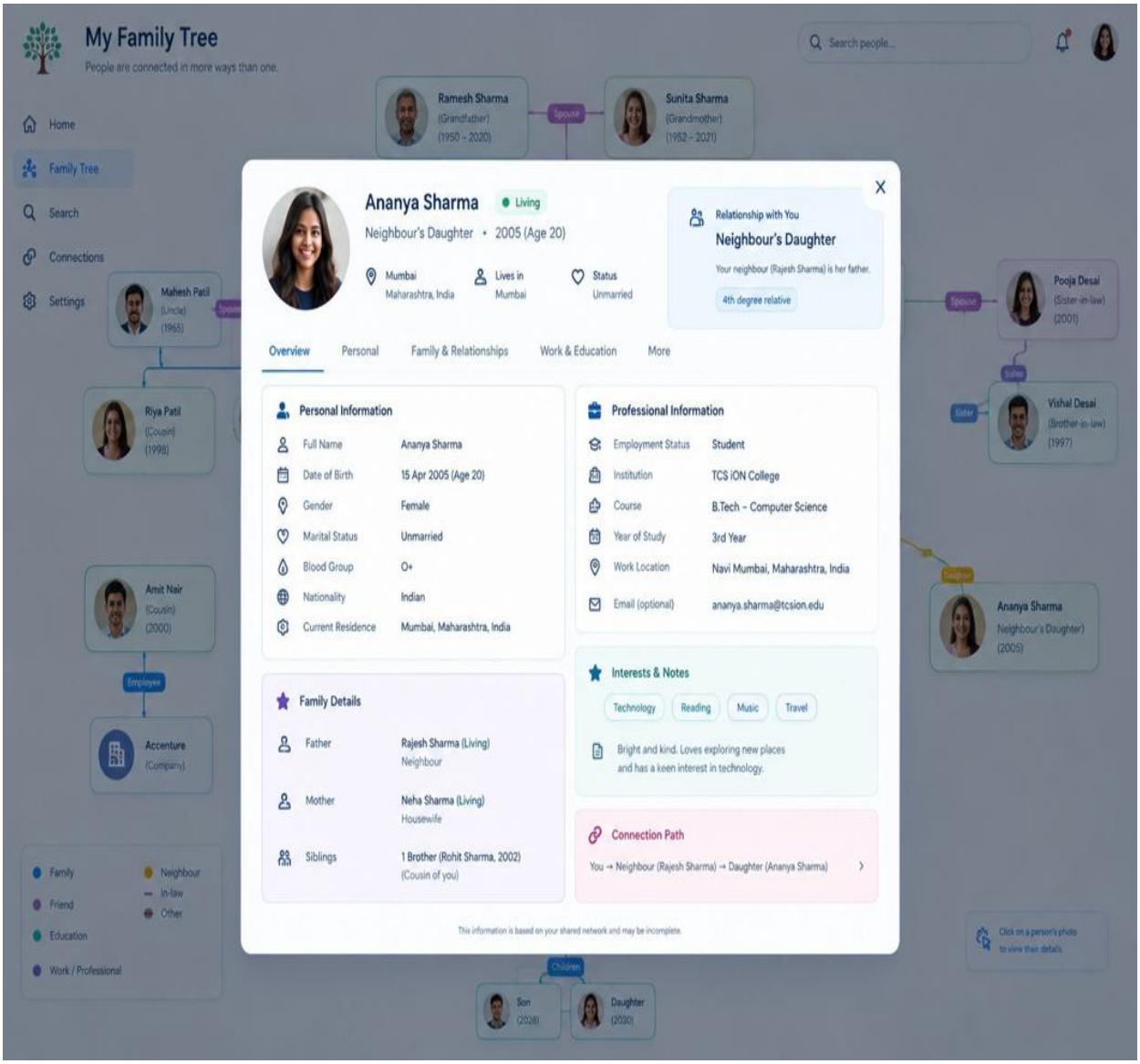


Conceptual UI mock-up using synthetic data.

4.8 Frontend Dashboard

The graph is the primary navigation surface. Users can search for people, organizations and places, zoom or filter the graph, select a node and return to the connection path. A dashboard view keeps the relationship structure visible while exposing a focused profile panel for the selected person.

The dashboard also separates “connectivity” from “biodata.” The graph answers how entities connect; the profile answers who the selected entity is. This prevents a dense graph from becoming a substitute for a conventional profile and keeps each interface element responsible for a clear task.



Conceptual mock-up; sensitive fields should be shown only with appropriate permission.

4.9 Living-Person Profile

Selecting a living person opens a modal profile over the graph rather than taking the user to a separate page. The proposed information hierarchy includes name, date of birth, living status, current residence, marital status, work or education, spouse, parents, children, relationship to the current user and the path that established that relationship.

The interface is intentionally permission-aware. FamilySearch provides a useful real-world precedent by limiting access to living-person records and supporting private and family-group sharing. [1][10] The proposed framework follows the same general principle: visibility should be controlled rather than assumed.

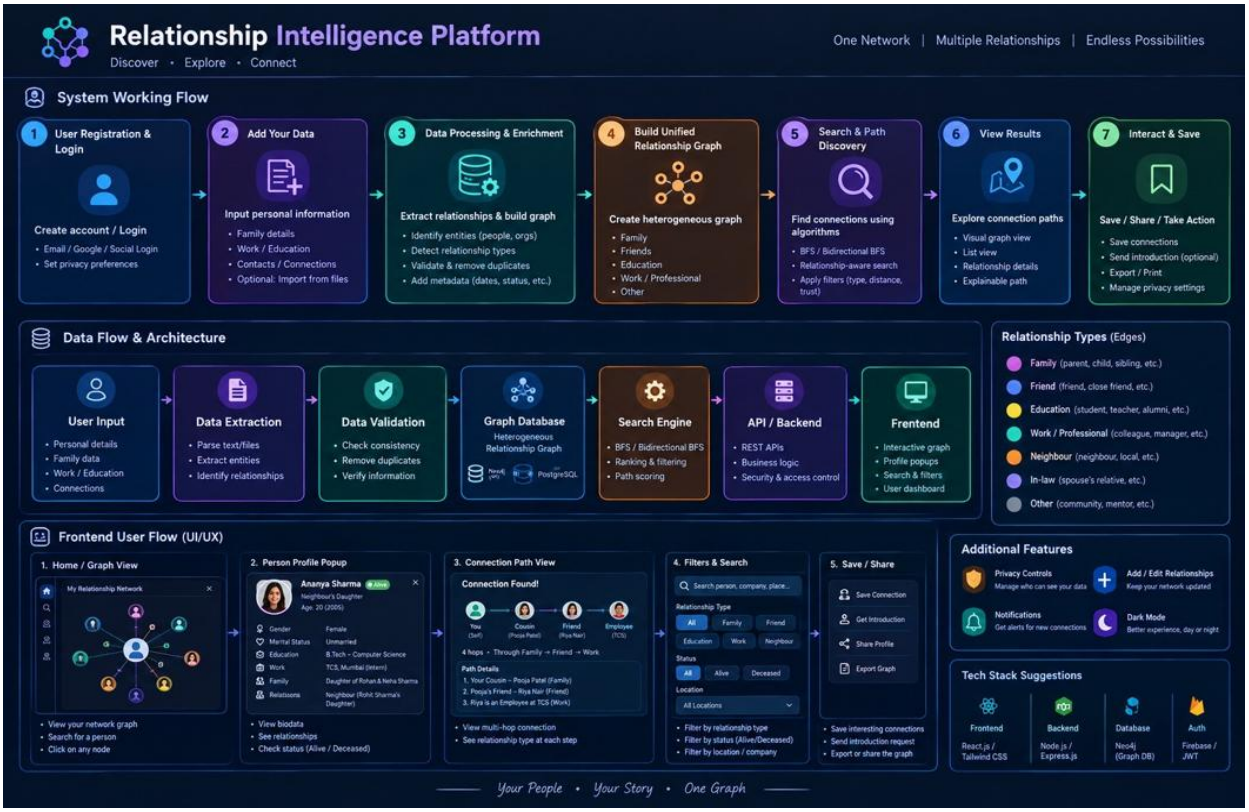


Conceptual mock-up using a synthetic deceased profile; historical fields should be sourced or verified in real use.

4.10 Deceased-Person Profile

A deceased profile can provide historical and family context, including date of birth, date of death, age at death, spouse, parents, children, former occupation, former organization and relationship to the current user. A remembrance or notes area can capture contextual information when appropriate.

The structural graph does not need a different algorithm for deceased people. Instead, the node status changes which properties are available to the interface and which permissions apply. This is an example of separating graph structure from policy and presentation.



The hierarchy separates extraction, validation, graph storage, search, explanation and frontend interaction.

The operational sequence is: register and define permissions -> add relationship information -> extract entities and relationships -> validate and deduplicate -> create or update the heterogeneous graph -> select a target -> run graph search -> reconstruct the path -> interpret the relationship sequence -> display the graph and profile -> optionally save or request an introduction when permitted.

The algorithm is only one layer of the solution. It should not decide who is trustworthy, whether a relationship is socially appropriate, or whether a job opportunity is suitable. Its technical responsibility is to discover graph paths. The interface and policy layers control what the user may see and what actions can be taken.

4.11 Core Equations

$G = (V, E)$ Basic graph representation.

$N = |V|$ Number of nodes.

$M = |E|$ Number of edges.

$\bar{d} = 2M/N$ Average degree for an undirected graph.

$D = 2M/[N(N-1)]$ Density of a simple undirected graph.

$L(P) = |P|-1$ Path length in edges.

$T_{BFS} = O(V+E)$ Standard BFS time complexity with adjacency lists.

$T_{BiBFS} \approx O(b^{(\bar{d}/2)})$ Simplified branching-factor intuition for bidirectional search; actual behavior depends on graph structure and termination conditions.

$Speedup = T_{BFS}/T_{BiBFS}$ Measured runtime ratio.

$Reduction = (X_{BFS}-X_{BiBFS})/X_{BFS} \times 100$ Percentage reduction in expanded nodes.

V. BIDIRECTIONAL BFS PSEUDOCODE

A. Input: graph G, source s, target t

1. Put s in the forward frontier and t in the backward frontier.
2. Record parent maps for both search directions.
3. Expand a frontier, preferably the smaller frontier.
4. When a new node appears in the opposite visited set, record the meeting node.
5. Reconstruct s -> meeting node and meeting node -> t.
6. Return the path and search statistics.

B. Why the Algorithm Matches the Query

The application query always provides two endpoints: the current user and a selected target. This is a natural

setting for two-ended search. In the unweighted case, BFS and correctly terminated Bidirectional BFS can return shortest paths. The research benchmark therefore starts with the cleanest possible comparison before introducing relationship weights or semantic constraints. [8][9]

C. Baseline Limitation

Plain Bi-BFS does not distinguish between a close friend and a weak acquaintance. Both may be represented by one edge. It also does not know whether a relationship is recent or verified. Consequently, shortest path is used here as a measurable baseline, not as a complete definition of relationship usefulness.

D. Path-Quality Measures for Future Experiments

Once relationship semantics are introduced, path quality can be evaluated using hop count, relationship-strength evidence, recency, verification and permission status. A simple path score may be written as S(P) = w1*strength + w2*recency + w3*verification - w4*risk - w5*hop_count. This equation is a research model rather than a final production formula; the weights must be justified empirically.

E. Inference Accuracy

For a labelled test set, relationship interpretation can be evaluated using true positives (TP), false positives (FP) and false negatives (FN). Precision = TP/(TP+FP), recall = TP/(TP+FN), and F1 = 2*Precision*Recall/(Precision+Recall). These measures are relevant when the system begins translating multi-edge paths into human-readable kinship labels or cross-domain connection descriptions.

F. Algorithm Comparison

Property	BFS	Bidirectional BFS
Search direction	One-sided	Forward + backward
Known endpoints	Source	Source and target
Typical unweighted role	Shortest-path baseline	Two-ended shortest-path baseline
Per-edge cost	Uniform	Uniform
Main output	Path + search effort	Path + search effort
Research variable	Runtime and nodes expanded	Runtime and nodes expanded

G. Worked Relationship-Path Example

Consider a query in which the source is a student and the target is a company. A valid path may be student -> friend -> friend's cousin -> employee -> company. The search algorithm identifies the connected sequence, while the interpreter labels each edge and presents the complete route. Because the intermediate nodes remain visible, a user can inspect the path instead of receiving an unexplained connection claim. For an unweighted graph, a valid shortest-path comparison requires that each traversal records visited states, maintains sufficient parent information for reconstruction, and stops only when the two searches have established a connection that satisfies the termination rule. For the empirical benchmark, both implementations use the same graph representation and target pair so that correctness and performance differences can be attributed to traversal strategy.

A controlled synthetic benchmark was executed to test the algorithmic component of the research. The graph-generation process creates connected sparse networks with additional random edges. For every graph size, the same graph and source-target pair are given to both algorithms. Five runs are executed and the median runtime is retained.

Table 2. Controlled synthetic benchmark results used in this paper.

Nodes	Edges	Target hops	BFS median ms	Bi-BFS median ms	BFS expanded	Bi-BFS expanded
100	124	6	0.0190	0.0038	99	7
500	624	10	0.0896	0.0084	500	18
1,000	1,249	10	0.2008	0.0123	999	25
5,000	6,249	12	1.4751	0.0404	4,999	80
10,000	12,499	14	2.6702	0.0623	10,000	129
20,000	24,999	14	5.9189	0.0903	19,998	178
50,000	62,499	15	22.7654	0.1503	49,995	212

The benchmark values are direct outputs of the accompanying Python experiment. They are not results taken from an external paper. Because graph structure, target choice, implementation and hardware all affect runtime, the values are used for within-experiment comparison rather than as universal speed claims.

H. Reproducibility and Ethical Boundary

The graph generator uses a fixed seed pattern based on graph size, adjacency lists and the same search implementation for both algorithms. No real family, contact, social-media or employment records were used. A future participant-based evaluation should add a consent process, data minimization rules and a formal privacy review.

I. Software and Execution Environment

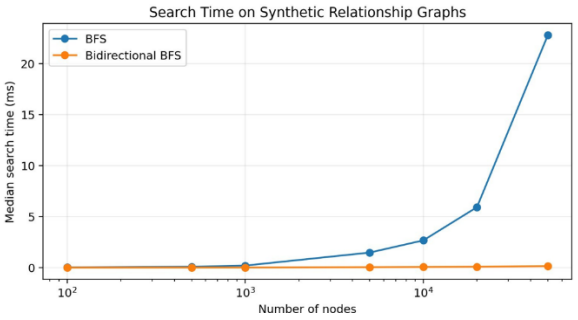
The benchmark is implemented in Python using an adjacency-list representation so that traversal overhead can be compared directly. Both algorithms run against the same graph objects and use the same timing procedure. Results are reported as medians across five runs to reduce sensitivity to a single transient measurement.

J. Experimental Procedure

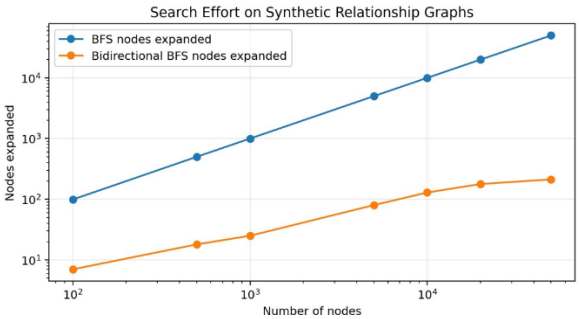
For each graph size, a connected backbone is generated first, controlled additional edges are added, a farthest reachable target is identified from node 0, and then BFS and Bidirectional BFS are run on the identical graph. Runtime and expanded-node counts are collected for every repetition before computing the reported median and derived ratios.

K. Benchmark Boundaries

The experiment evaluates graph traversal, not the complete application. It does not measure database latency, network latency, privacy-policy evaluation, user-interface rendering, entity matching or whether a person would accept an introduction. Those components require separate experiments with appropriate data and consent.

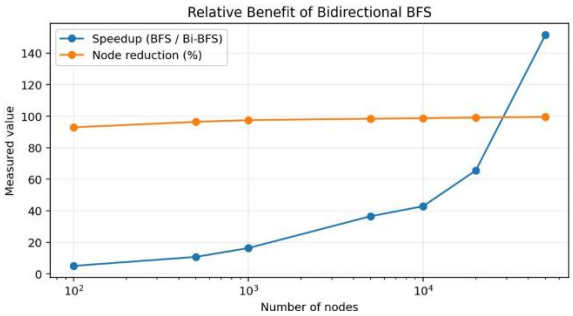


The x-axis is logarithmic; both algorithms operate on identical graphs.



Lower values represent less frontier exploration in the benchmark.

For every tested graph size, Bidirectional BFS recorded a lower median runtime and expanded substantially fewer nodes than one-sided BFS. The difference becomes larger in the 20,000- and 50,000-node cases under the selected farthest-target condition. The result supports H1 and H2 within the benchmark.



Speedup = BFS runtime / Bi-BFS runtime; node reduction is the percentage reduction in expanded nodes.

VI. MEASURED RESULTS

At 100 nodes, the median runtime was approximately 0.0190 ms for BFS and 0.0038 ms for Bidirectional BFS, corresponding to a measured ratio of about 5.0x. At 50,000 nodes, the medians were approximately

22.7654 ms and 0.1503 ms, corresponding to about 151.5x in this particular benchmark. Expanded nodes fell from 49,995 for BFS to 212 for Bidirectional BFS, a reduction of about 99.6%.

These ratios are striking but must be interpreted carefully. The synthetic graphs are sparse, the target is deliberately chosen as a farthest node, and the implementation cost is measured in a controlled Python environment. A real relationship platform may show smaller gains when paths are already short, when hubs create many alternative routes, or when database queries and permission checks add overhead.

6.1 Interpretation

The key finding is not that Bidirectional BFS will always be faster. It is that two-terminal graph search is a technically plausible core for the proposed "Who connects me to X?" query, and the benchmark provides measurable evidence that the strategy can reduce search effort under specified conditions. The next research stage should therefore compare unweighted shortest paths with relationship-aware paths.

The path itself also provides an explanation. A result such as "You -> friend -> friend's cousin -> employee -> company" is inspectable because every hop is visible. This is preferable to a black-box statement that a connection "exists," particularly when the user needs to decide whether to contact an intermediary.

6.2 Relationship-Aware Search as the Next Step

A future path model could combine hop count with measurable properties such as relationship strength, recency and verification. A candidate score can be written as S(P) = w1*strength + w2*recency + w3*verification - w4*risk - w5*hop_count, but the weights should be learned or justified through data and user studies rather than invented arbitrarily. The present paper intentionally keeps this model outside the primary benchmark.

6.3 Findings Against the Research Questions

RQ1 is addressed by the proposed heterogeneous representation, which allows family, social, educational and professional edges to coexist. RQ2 is examined through the BFS versus Bidirectional BFS benchmark, where the two-ended method expands fewer nodes under the tested conditions. RQ3 is addressed by separating path traversal from the relationship interpreter so that the intermediate edges remain visible. RQ4 is partly addressed by testing several graph sizes; denser and more varied graph structures remain future work. RQ5 is addressed at the design level through permission checks, explicit status fields and synthetic experimentation, but real user studies are still required.

6.4 Advantages

- A single graph can unify multiple relationship domains instead of forcing separate trees.
- Multi-hop paths can reveal connections that are difficult to remember manually.
- The path is explainable because the intermediate people and relationship types are visible.
- A person profile can be opened directly from the graph.
- The model is extensible to organizations, education, alumni and professional relationships.
- Synthetic experiments allow algorithm comparison without exposing private personal data.

6.5 Limitations

- Results depend on the completeness and accuracy of user-provided relationships.
- Plain BFS and Bi-BFS do not understand relationship strength or context.
- Living-person information is privacy-sensitive and cannot simply be made public.
- Duplicate or ambiguous identities can create incorrect graph paths.
- The shortest path is not necessarily the most meaningful or safest path.
- The experiment uses synthetic graphs and does not validate real-world social behavior.
- Natural-language relationship entry using entity and relation extraction.
- Weighted search using relationship strength, recency and verification.
- Comparison with Dijkstra, A* and multi-criteria search for richer path objectives.
- Entity resolution to merge duplicate people and organizations safely.
- Temporal graphs that capture changing jobs, locations and relationships.
- Privacy-preserving graph queries and selective disclosure of relationship paths.

- Human-subject evaluation to test comprehension, trust and usability of connection explanations.

This research proposes a unified heterogeneous relationship graph for explainable multi-hop connection discovery. The idea grows from the traditional family tree but extends the representation to friends, in-laws, neighbours, teachers, mentors, classmates, colleagues and organizations. The core query is a graph question: given two endpoints, what valid relationship path connects them?

Bidirectional BFS is a logical baseline because the query has two known endpoints. In the controlled synthetic experiment, it expanded far fewer nodes and recorded lower runtime than conventional BFS. At the same time, the research shows why algorithmic speed alone is not enough. A practical system must represent semantic relationship types, explain the discovered path, control access to living-person information, manage incomplete or conflicting data and distinguish structural connectivity from real-world outcomes.

VII. RESEARCH SIGNIFICANCE AND NEXT VALIDATION STEP

The main contribution of the study is a testable framework rather than a claim that one search algorithm solves relationship intelligence. The framework provides a common graph representation, an interpretable path model and an experimental baseline that can be extended to weighted, temporal and privacy-aware search. The next validation stage should use consented anonymized data and compare several path-selection strategies while preserving a clear audit trail of why each connection was shown.

VIII. PRACTICAL DESIGN IMPLICATIONS

A usable relationship platform should keep three questions separate: who is the person, how is the person connected, and what action is appropriate? The profile popup answers the first, graph traversal answers the second, and user choice answers the third. Keeping these responsibilities separate reduces the risk of presenting a technically valid graph path as a recommendation about a person's trustworthiness, suitability or expected outcome.

IX. REPRODUCIBILITY SUMMARY

The research can be reproduced from the stated graph sizes, edge-generation rule, source-target selection rule, repeated timing protocol and evaluation formulas. The accompanying benchmark data are therefore part of the experimental record. Reproduction on different hardware may change absolute runtime while leaving the comparison framework intact.

[S1] Types of Research, course material supplied for MSc. Data Analytics research-methodology work. User-provided course material, 2026.

REFERENCES

[1]. FamilySearch, "Privacy in Family Tree: How Does Family Tree Protect the Privacy of Living People?" 2025.
[2]. FamilySearch, "Can I Share Access to Living and Confidential People in Family Tree?" 2025.
[3]. LinkedIn Help, "Your Network and Degrees of Connection," current help documentation.
[4]. Affinity, "Which Relationships Actually Drive Warm Introductions?" 2026.
[5]. L. Outemzabet, N. Gaud, A. Bertaux, C. Nicolle, S. Gerart, and S. Vachenc, "Harnessing heterogeneous information networks: A systematic literature review," Computer Science Review, vol. 52, Art. no. 100633, 2024, doi: 10.1016/j.cosrev.2024.100633.
[6]. C. Shi, Y. Li, J. Zhang, Y. Sun, and P. S. Yu, "A survey of heterogeneous information network analysis," IEEE Transactions on Knowledge and Data Engineering, vol. 29, no. 1, 2017, doi: 10.1109/TKDE.2016.2598561.
[7]. Neo4j Documentation, "What Is a Graph Database?" and "Graph Database Concepts," official documentation.
[8]. I. Pohl, "Bi-directional search," in Machine Intelligence, vol. 6, pp. 127–140. Edinburgh, U.K.: Edinburgh University Press, 1971.
[9]. T. Bläsius and M. Wilhelm, "Deterministic performance guarantees for bidirectional BFS on real-world networks," Journal of Computer and System Sciences, vol. 159, Art. no. 103774, 2026, doi: 10.1016/j.jcss.2026.103774.
[10]. M. Rajaei, M. S. Haghjoo, and E. Khanjari

Miyaneh, “Ambiguity in social network data for presence, sensitive-attribute, degree and relationship privacy protection,” PLOS ONE, vol. 10, no. 6, Art. no. e0130693, 2015, doi: 10.1371/journal.pone.0130693.

[11]. “Privacy protection and utility trade-off for social graph embedding,” Information Sciences, vol. 676, Art. no. 120866, 2024, doi: 10.1016/j.ins.2024.120866.