Adding Counting Quantifiers to Graph Patterns

Wenfei Fan[1,2] Yinghui Wu[3] Jingbo Xu[1,2]

[1]University of Edinburgh [2]Beihang University [3]Washington State University

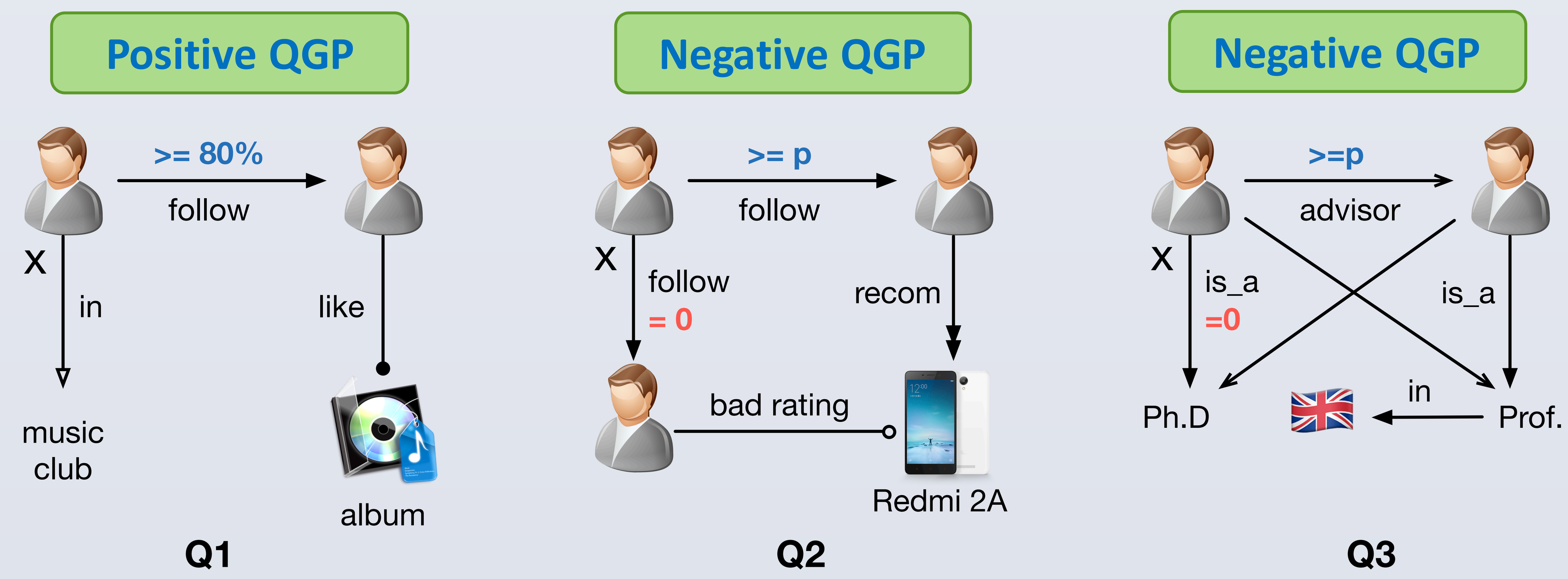
INTRODUCTION

- We propose ***Quantified Graph Patterns*** (QGP). Using simple counting quantifiers, QGPs uniformly support numeric and ratio aggregates, universal and existential quantification, and negation.
- We formalize quantified matching and establish its **complexity**.
- As an application of QGPs, we introduce *quantified graph association rules* (QGAR)
- We develop parallel algorithms for quantified matching, and experimentally verify the effectiveness of QGPs and the scalability of our algorithms.

QUANTIFIED GRAPH PATTERNS

◦ If (a) person x is in a music club, and (b) among the people whom x follows, at least 80% of them like an album y, then the chances are that x will buy y.

◦ If among the people followed by x, (a) at least p of them recommend Redmi 2A, and (b) no one gives Redmi 2A a bad rating, then x may buy Redmi 2A.



A quantified graph pattern $Q(x_o)$ is defined as (V_Q, E_Q, L_Q, f), where f is a function such that for each edge $e \in E_Q$, $f(e)$ is a predicate of

◦ a positive form $\sigma(e) \odot p\%$ for a real number $p \in (0, 100]$, or $\sigma(e) \odot p$ for a positive integer p, (Here $\odot$ is either = or ≥)

◦ $\sigma(e) = 0$, where e is referred to as a negated edge.

We refer to $f(e)$ as the counting quantifier of e, which express logic quantifiers as follows:

◦ **negation** when $f(e)$ is $\sigma(e) = 0$ (e.g., $Q2$);

◦ **existential quantification** if $f(e)$ is $\sigma(e) \geq 1$; and

◦ **universal quantifier** if $f(e)$ is $\sigma(e) = 100\%$.

THE COMPLEXITY OF QUANTIFIED MATCHING

Quantified matching problem:

◦ Input: A QGP $Q(x_o)$ and a graph G.

◦ Output: $Q(x_o, G)$, the set of all matches of query focus x.

Theorem: The quantified matching problem remains NP-complete for positive QGPs, and it becomes DP-complete for (possibly negative) QGPs.

PARALLEL QUANTIFIED MATCHING

An algorithm is parallel scalable if

$$T(|A|, |G|, n) = O\left(\frac{t(|A|, |G|)}{n}\right) + (n|A|)^{O(1)}$$

- $T(|A|, |G|, n)$: worst case running time for solving problem A over graph G using n processors;
- $t(|A|, |G|)$: worst case running time of sequential algorithm.

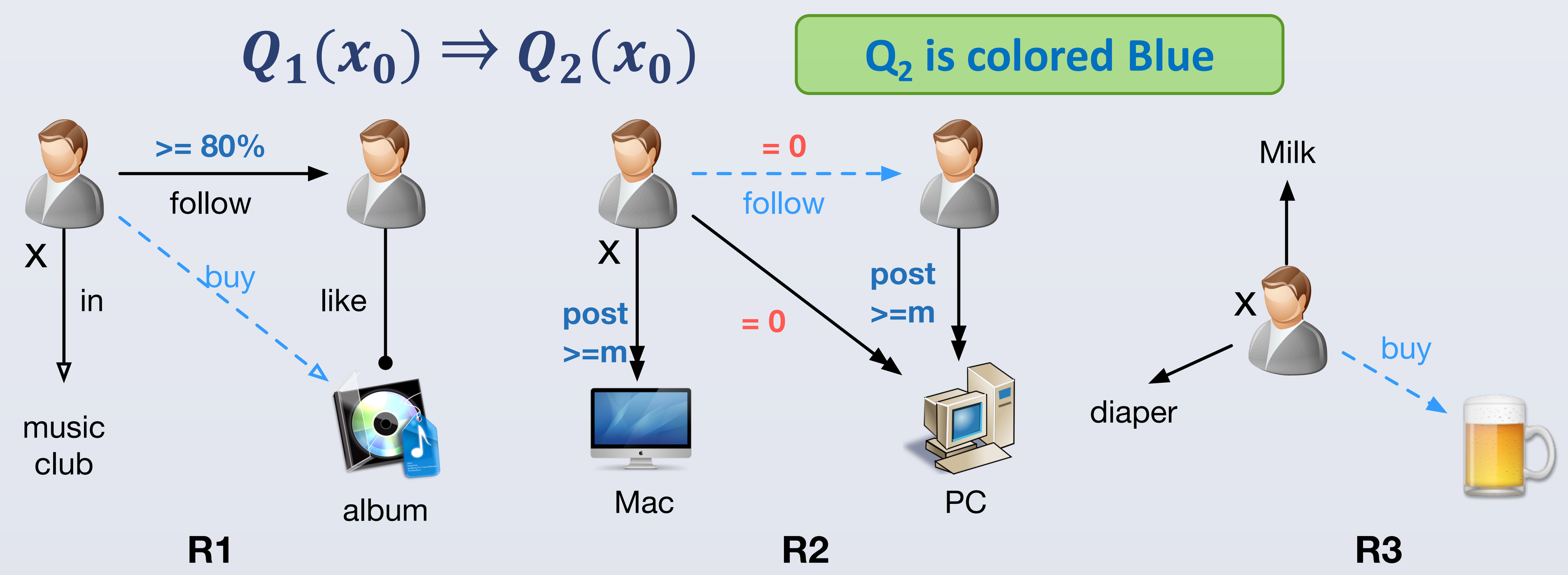
A parallel scalable QGPs matching algorithm `PQMatch` contains:

- **Partition** (*d-hop preserving partition* for candidates of x)
- **Local Matching** Compute local matches $Q(x, F_i)$ at each fragment F_i in parallel and check the satisfaction for quantifiers.
- **Assembling** Return matches of R(x, G)

QUANTIFIED ASSOCIATION RULES

QGARs: A quantified graph association rule R is defined as

$$Q_1(x_0) \Rightarrow Q_2(x_0)$$

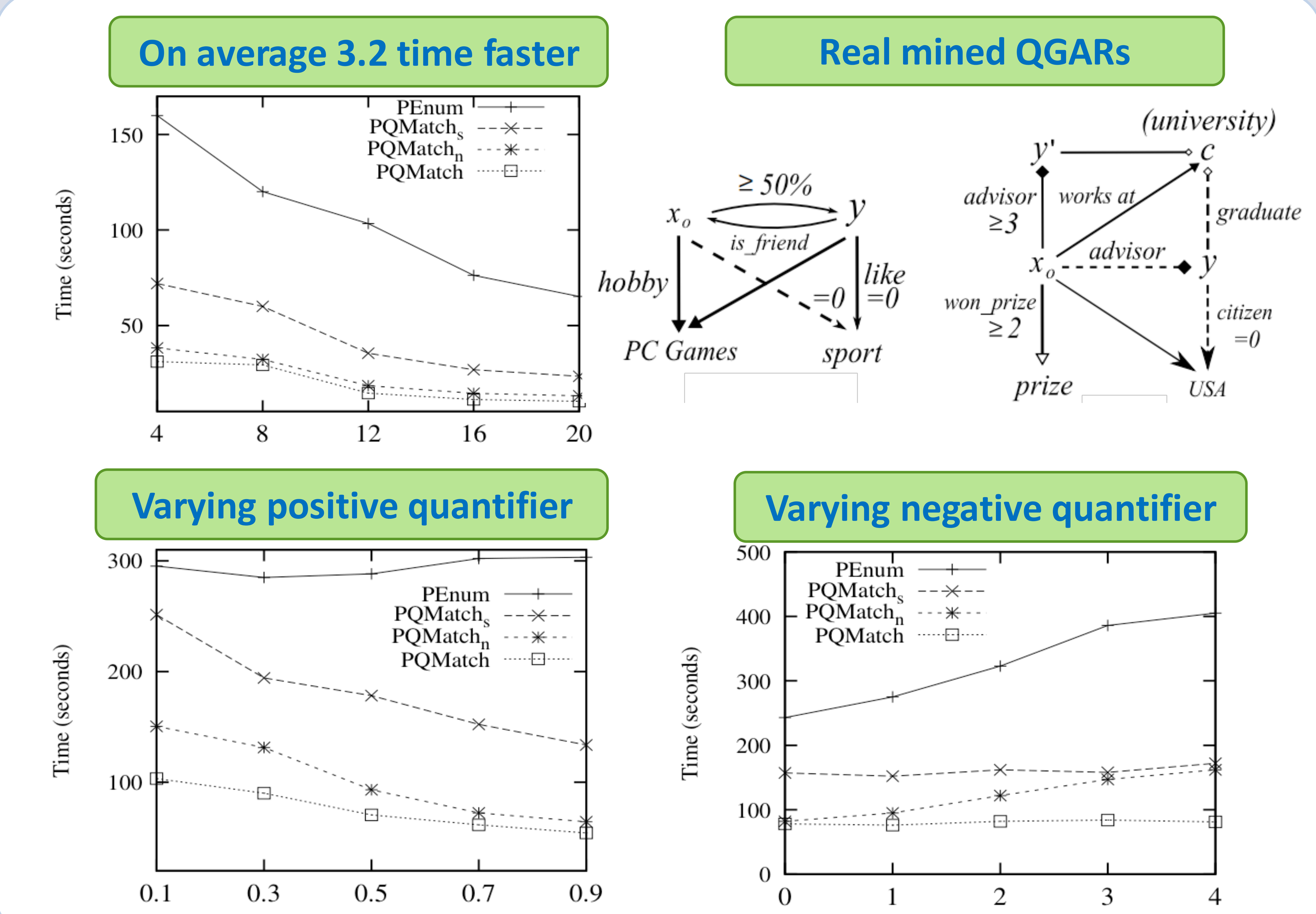


◦ If x_o is in a music club and if 80% of people whom x_o follows like an album y, then x_o will likely buy y.

◦ If x_o and y actively (≥ m) tweet on competitive products (e.g., "Mac" vs "PC"), then x_o is unlikely to follow y.

EXPERIMENTAL STUDY

- Pokec: 1.63M nodes of 269 types, and 30.6M edges of 11 types;
- Yago: 1.99M entities and 5.65M links of 36 types.



CONCLUSION

The novelty of this work consists in quantified patterns (QGPs), quantified graph association rules (QGARs), and algorithms with provable guarantees. Our experimental study has verified the effectiveness of QGPs and the feasibility of quantified matching.