Scaling RDF Systems

Jiewen Huang, *et. al*.
VLDB 2011

Michael Abebe
CS 848 (Mar 2019)

UNIVERSITY OF WATERLOO

Google

University of Waterloo

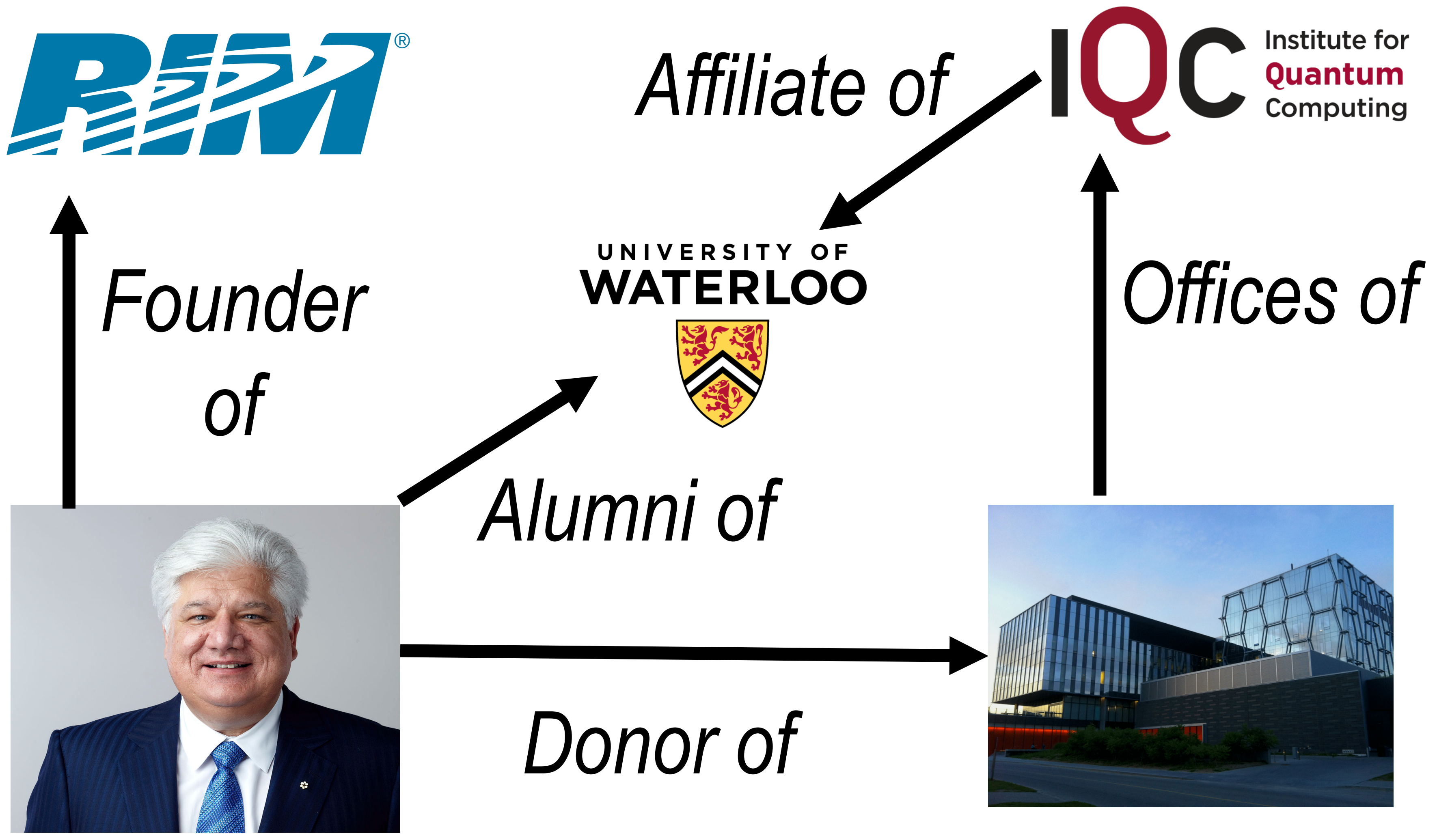
Notable Alumni

People also search for:

UNIVERSITY OF TORONTO

Western UNIVERSITY · CANADA

UNIVERSITY OF WATERLOO


RIM
Affiliate of
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
Founder of
Offices of
Alumni of
Donor of

Casey Newton

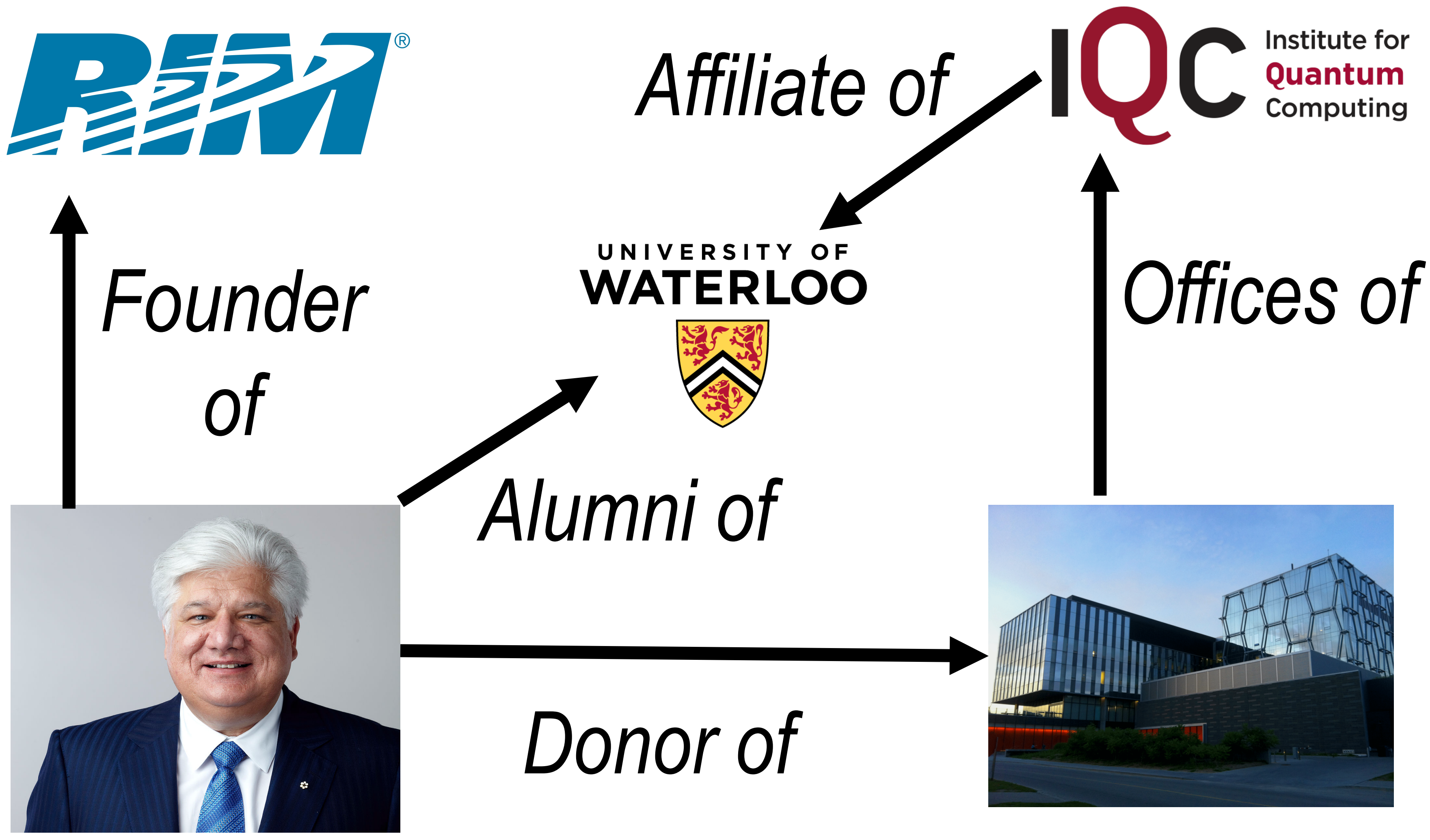
Dec 2012

Google's **Knowledge Graph tripled in size** in seven months

At launch, a database of the relationships among **500 million objects and 3.5 billion facts.**

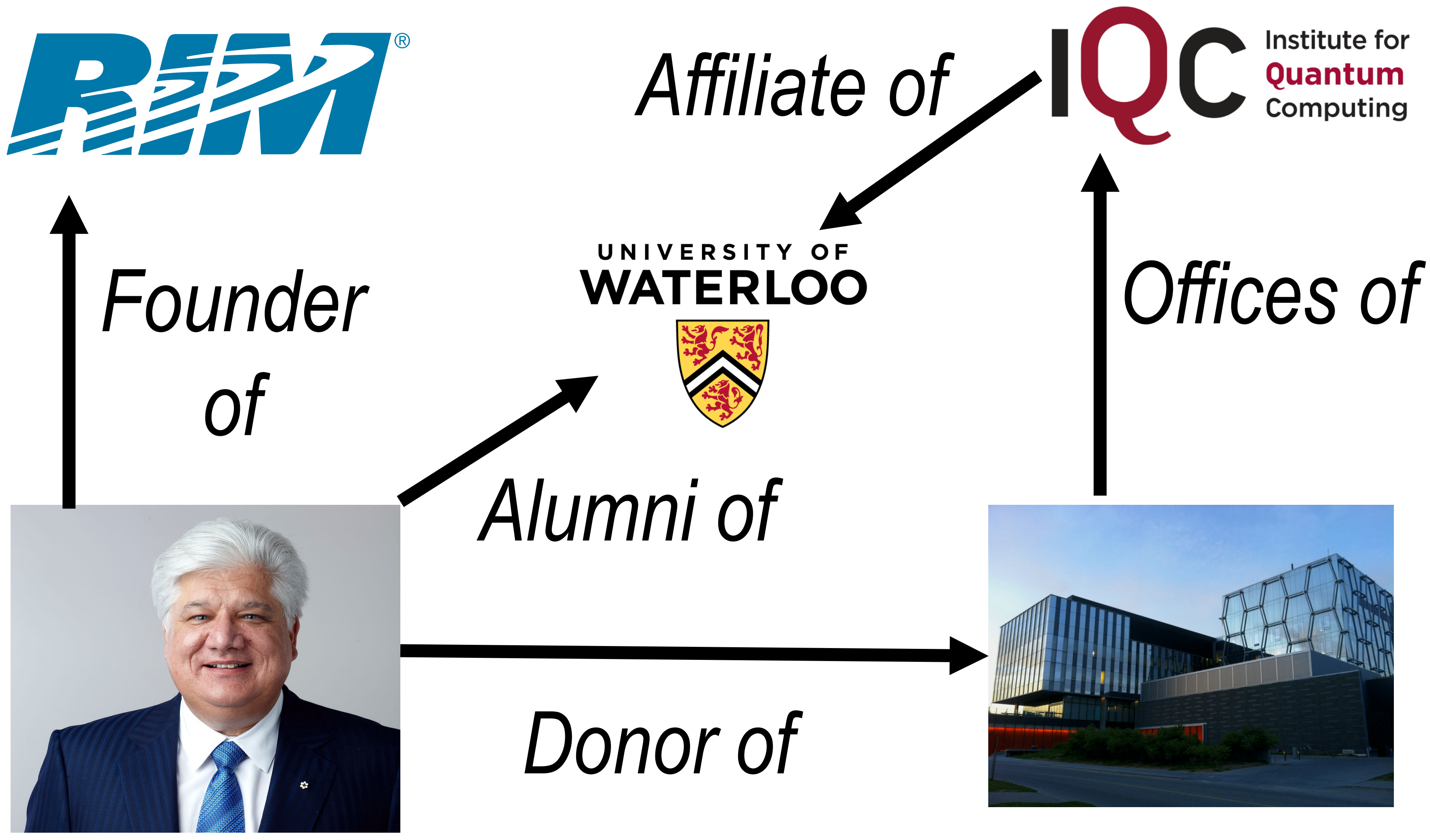
UNIVERSITY OF WATERLOO

Knowledge Graph

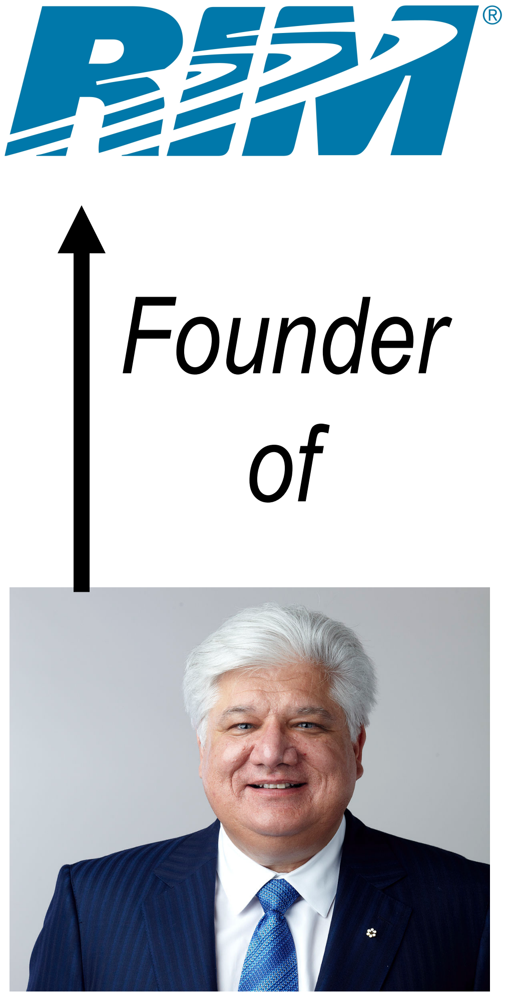


UNIVERSITY OF WATERLOO

RDF


RIM
IQC Institute for Quantum Computing
Affiliate of
UNIVERSITY OF WATERLOO
Founder of
Offices of
Alumni of
Donor of

How to scale **RDF** systems?

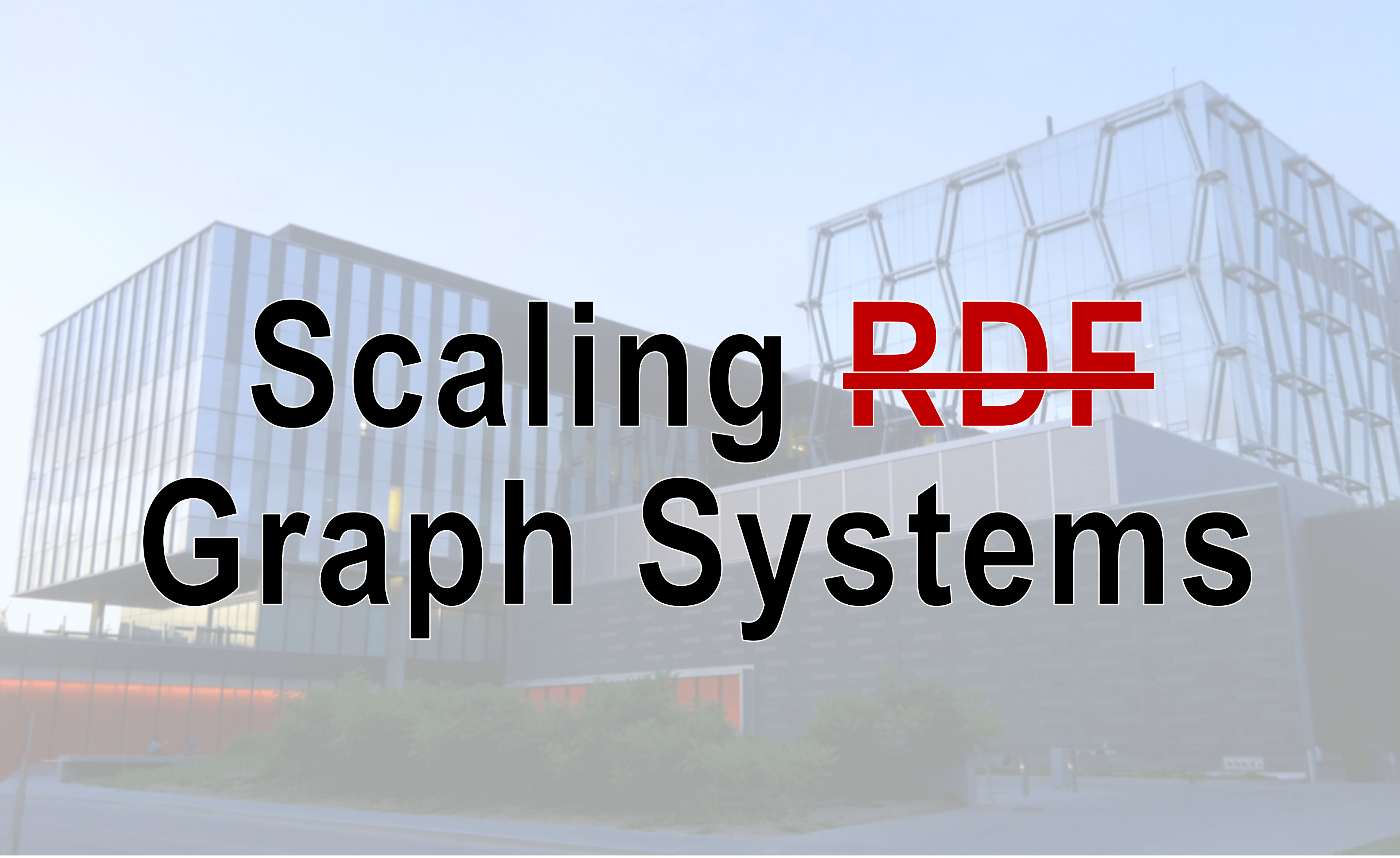


$$\begin{tikzcd}
\text{Object} & \text{Vertex} \\
\text{Subject} \arrow[u, "\textit{Predicate}"'] & \text{Vertex} \arrow[u, "\textit{Labelled Edge}"']
\end{tikzcd}$$

How to scale RDF systems?

Scale graph systems

UNIVERSITY OF WATERLOO



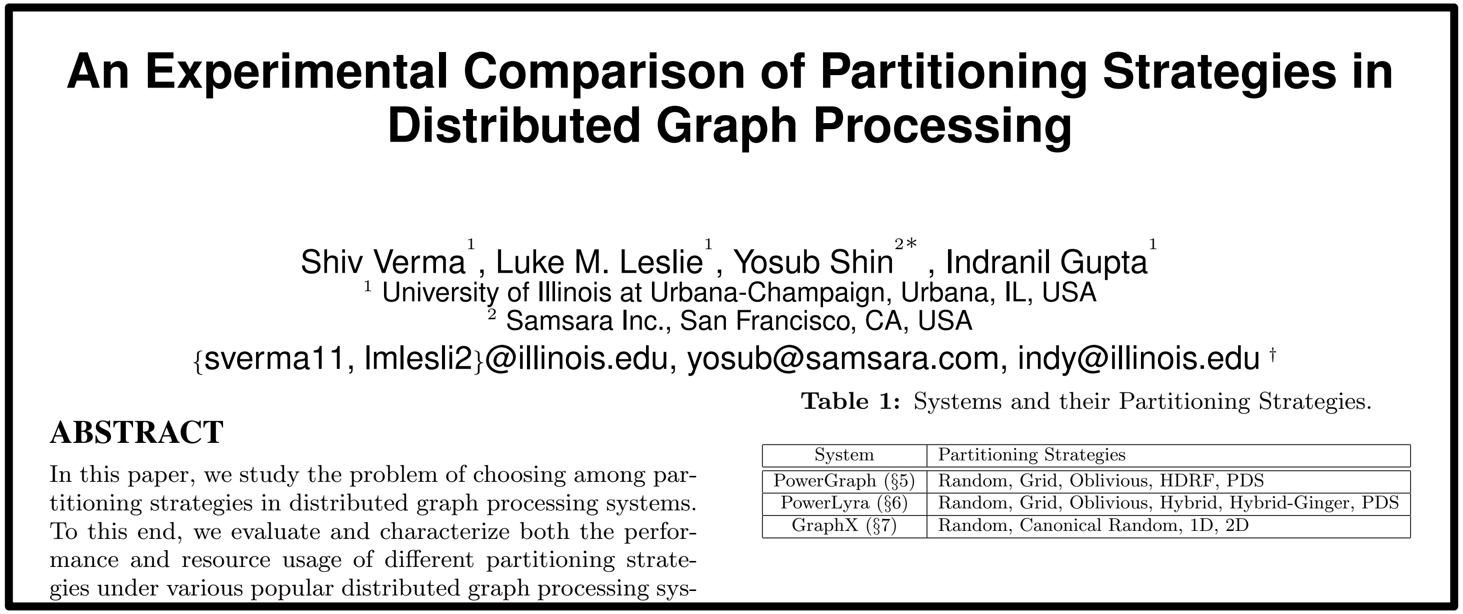
Scaling ~~RDF~~ Graph Systems

Jiewen Huang, *et. al.*
VLDB 2011

Michael Abebe
CS 848 (Mar 2019)

UNIVERSITY OF WATERLOO

How to scale graph systems?



An Experimental Comparison of Partitioning Strategies in Distributed Graph Processing

Shiv Verma[1], Luke M. Leslie[1], Yosub Shin[2*], Indranil Gupta[1]
[1] University of Illinois at Urbana-Champaign, Urbana, IL, USA
[2] Samsara Inc., San Francisco, CA, USA
{sverma11, lmlesli2}@illinois.edu, yosub@samsara.com, indy@illinois.edu [†]

ABSTRACT

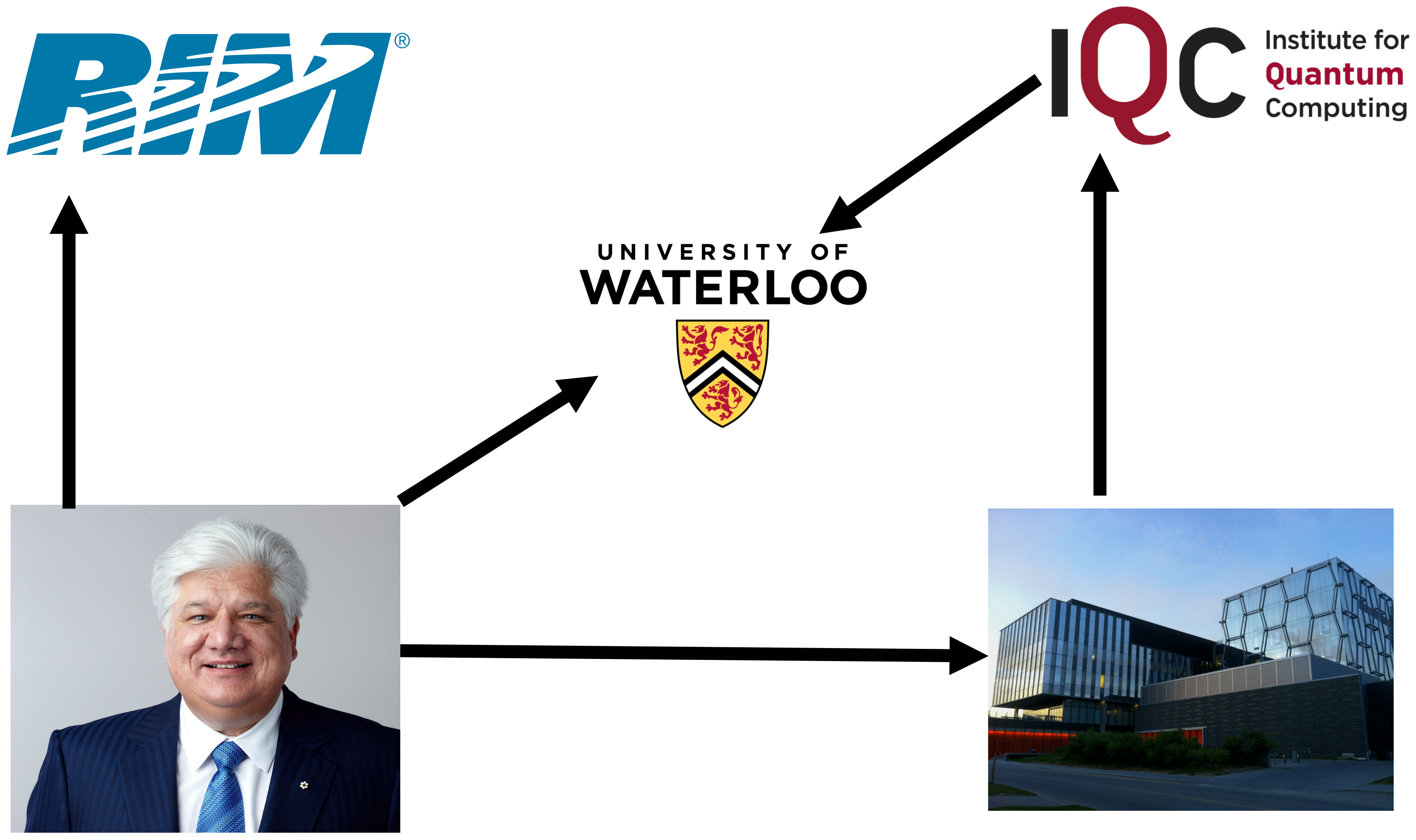
In this paper, we study the problem of choosing among partitioning strategies in distributed graph processing systems. To this end, we evaluate and characterize both the performance and resource usage of different partitioning strategies under various popular distributed graph processing sys-

Table 1: Systems and their Partitioning Strategies.

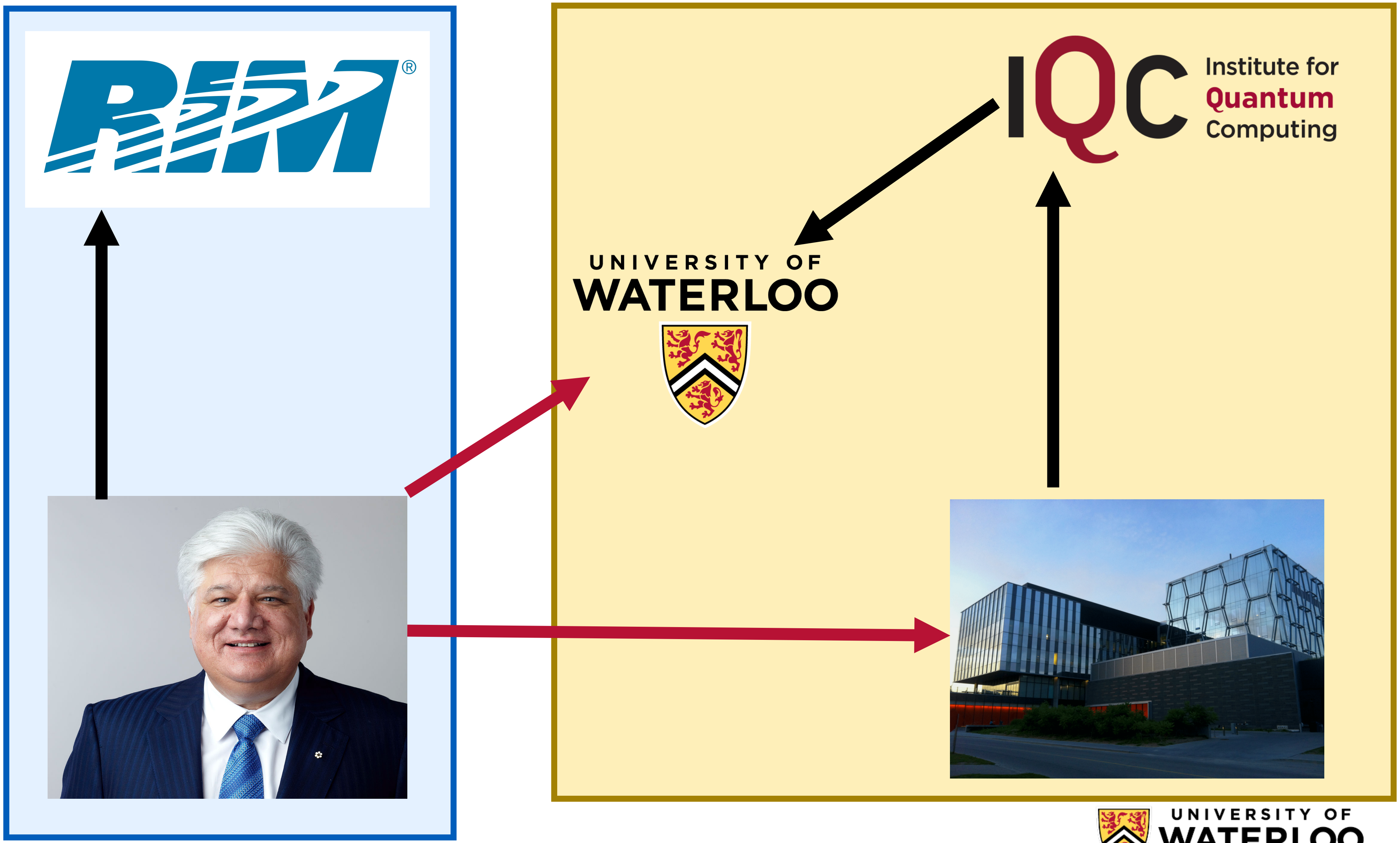
System	Partitioning Strategies
PowerGraph (§5)	Random, Grid, Oblivious, HDRF, PDS
PowerLyra (§6)	Random, Grid, Oblivious, Hybrid, Hybrid-Ginger, PDS
GraphX (§7)	Random, Canonical Random, 1D, 2D

Select a good graph partitioning

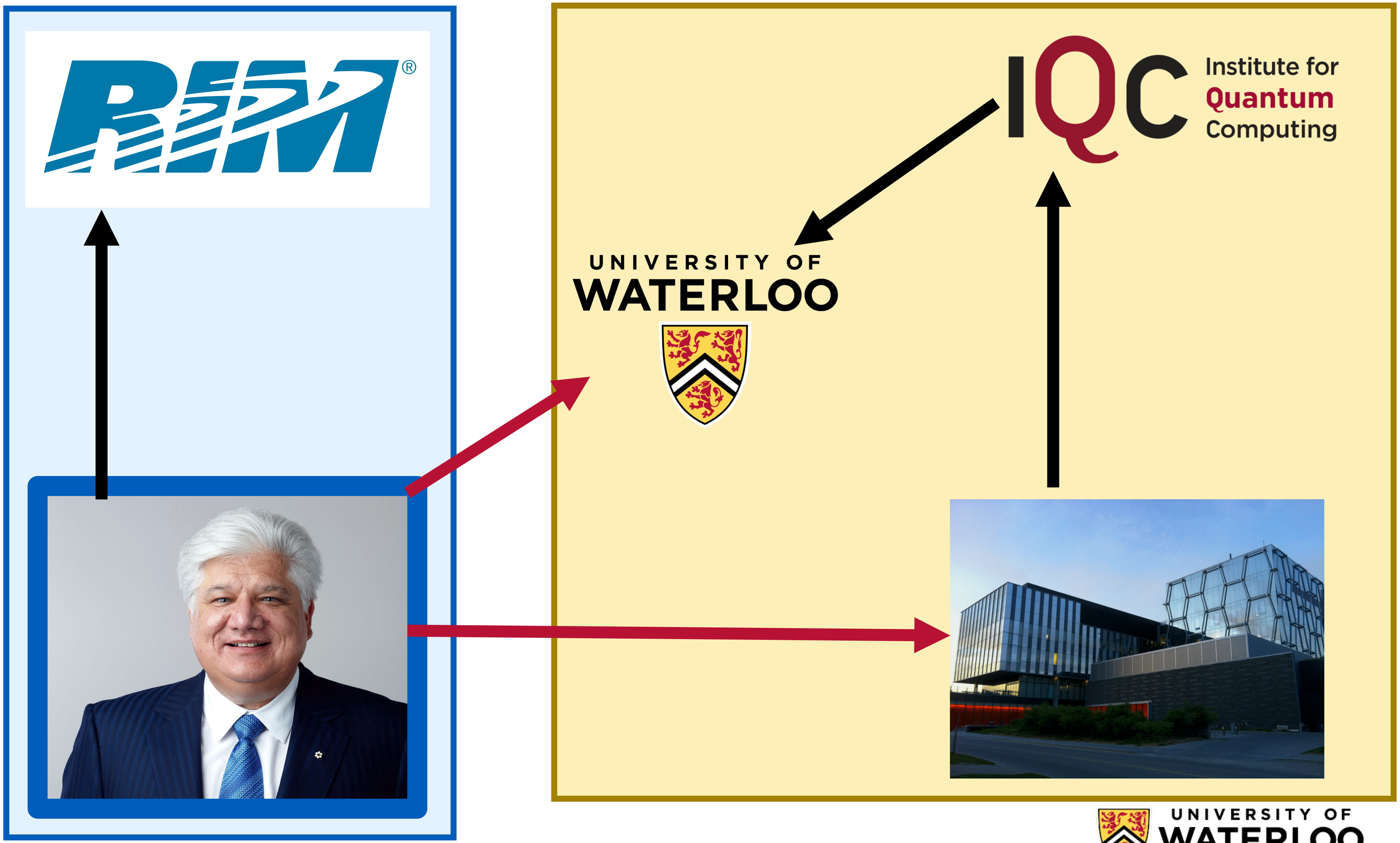
Partitioning Graphs



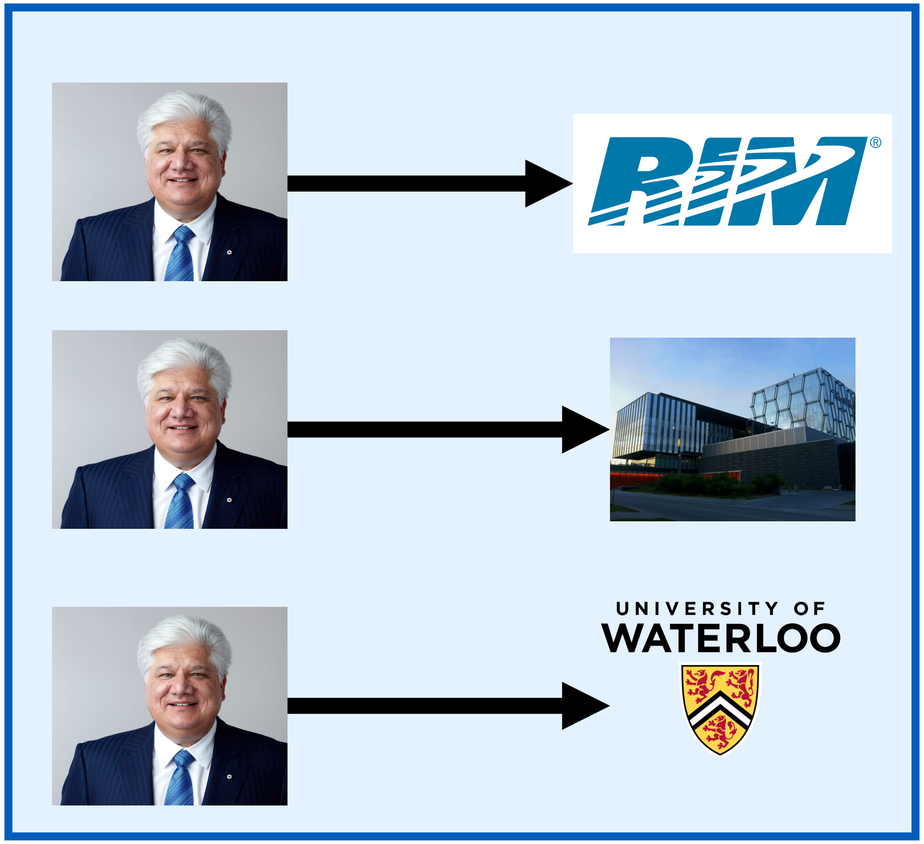
Partitioning Graphs



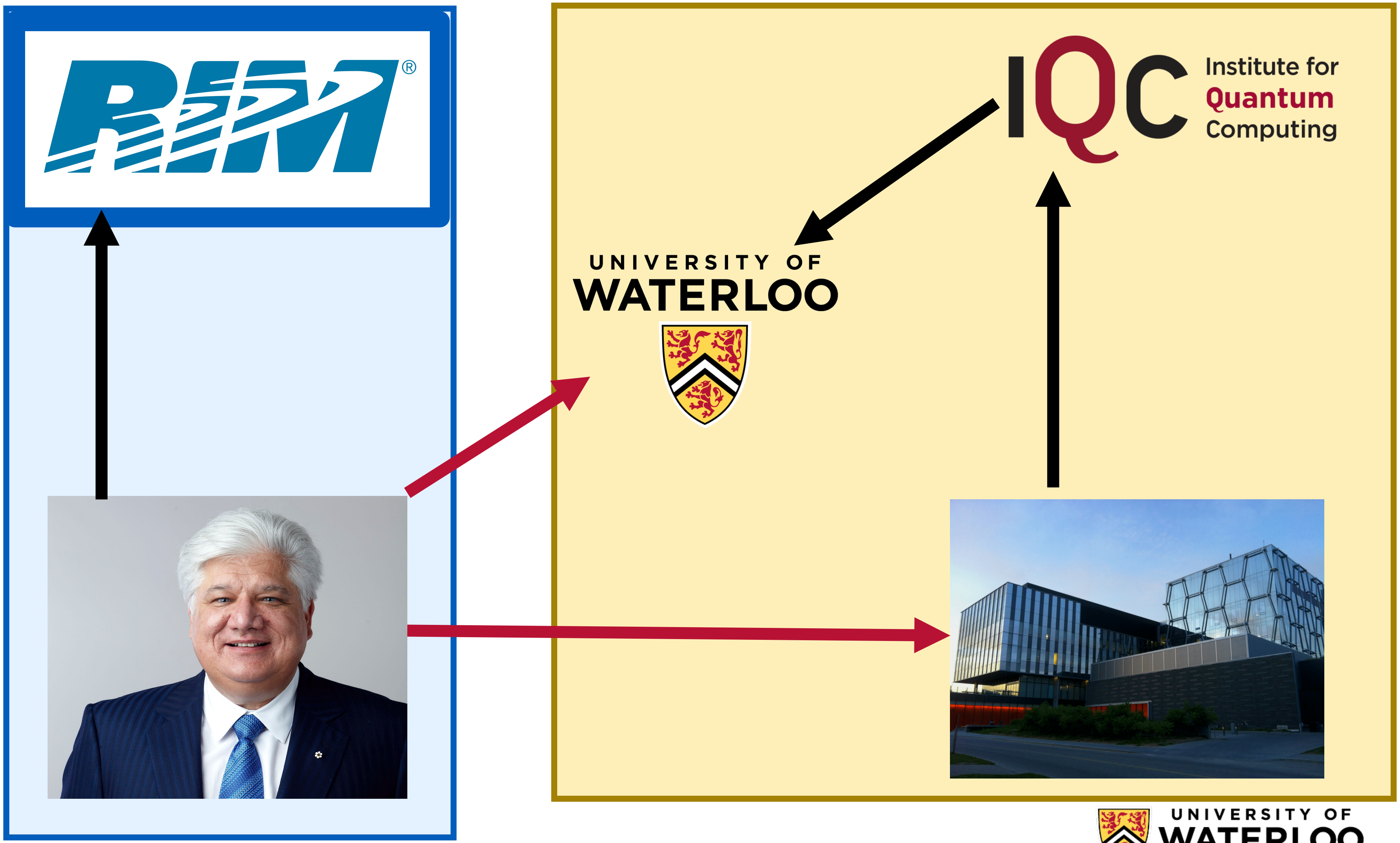
Placing Triples



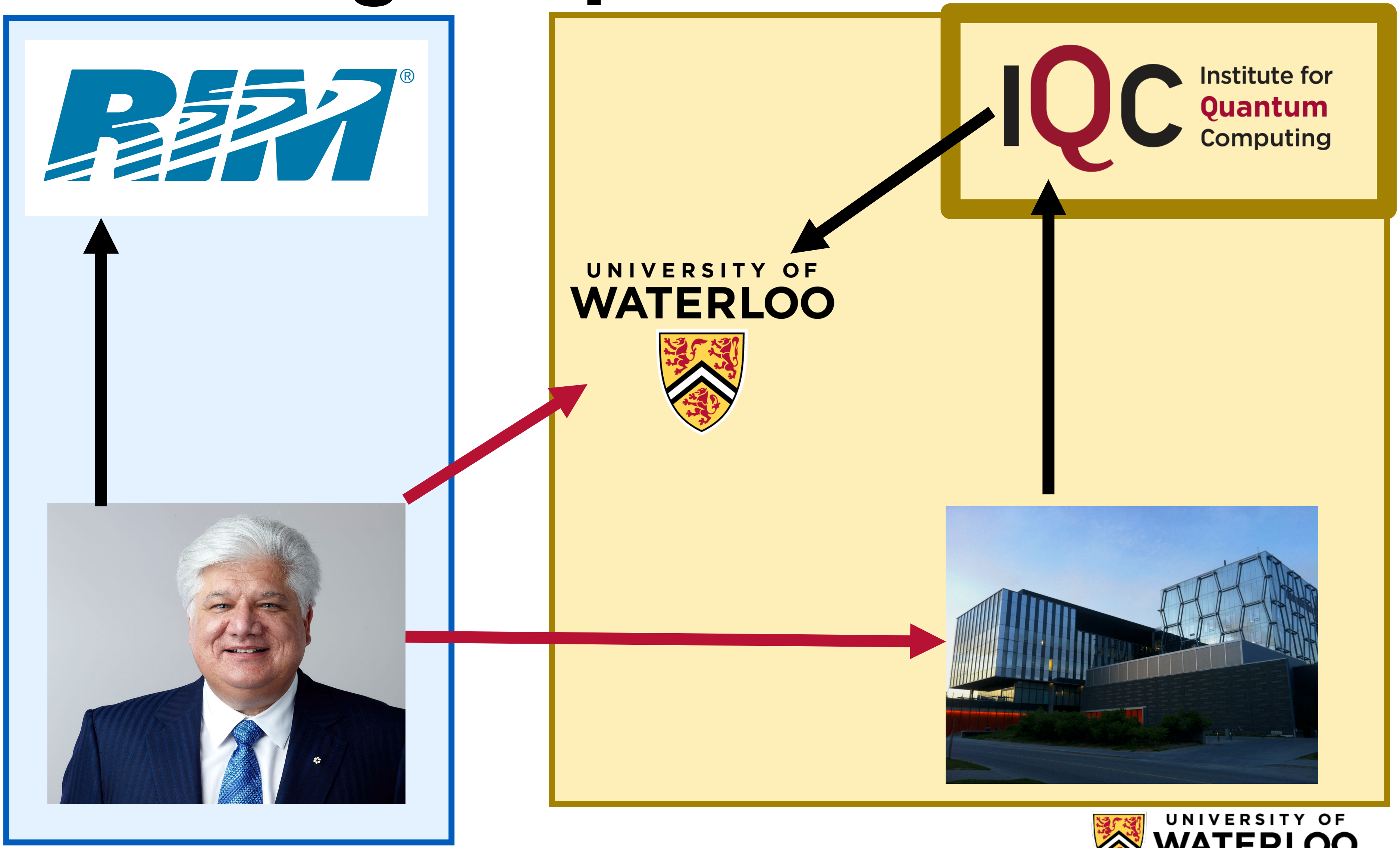
Placing Triples



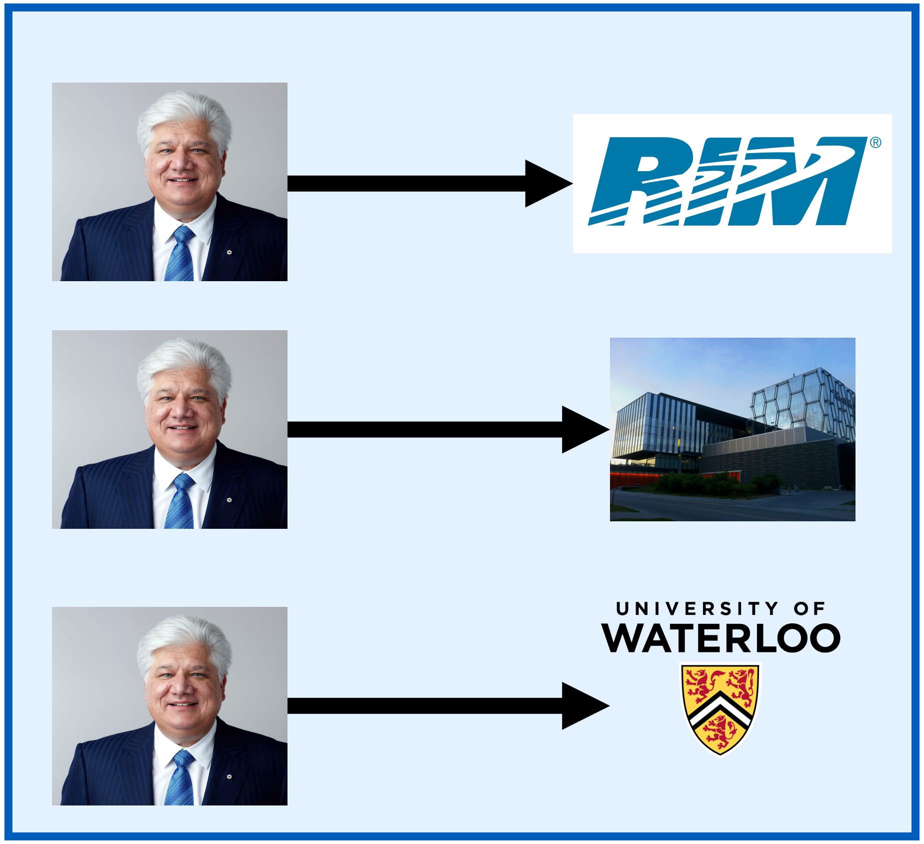
Placing Triples

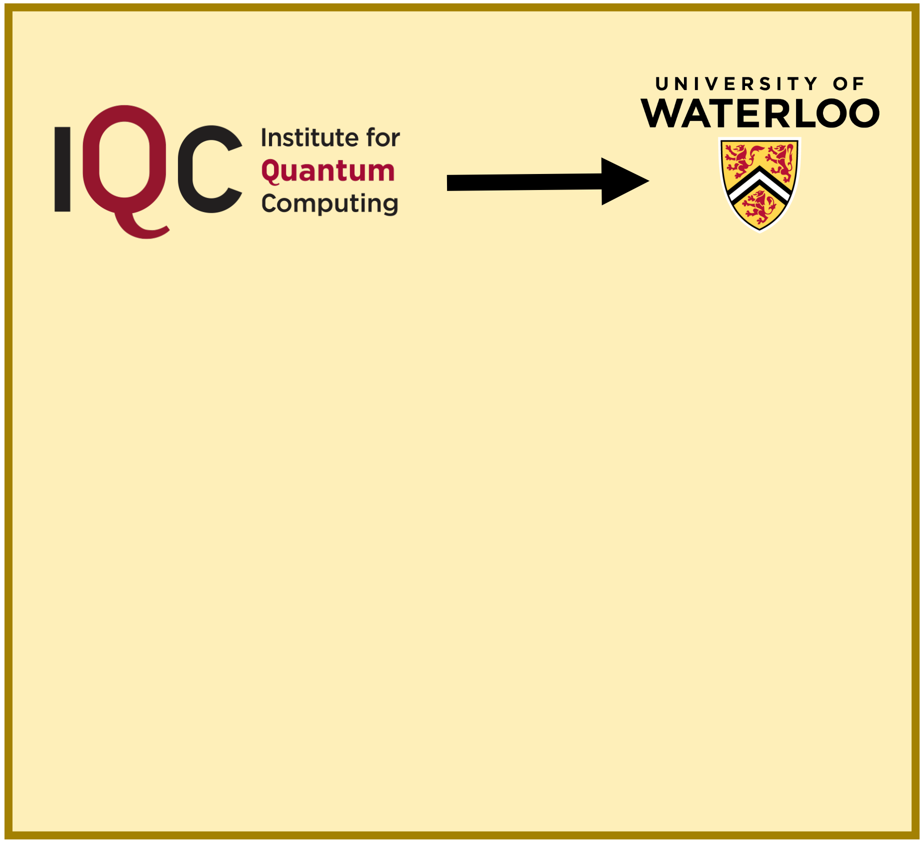


Placing Triples

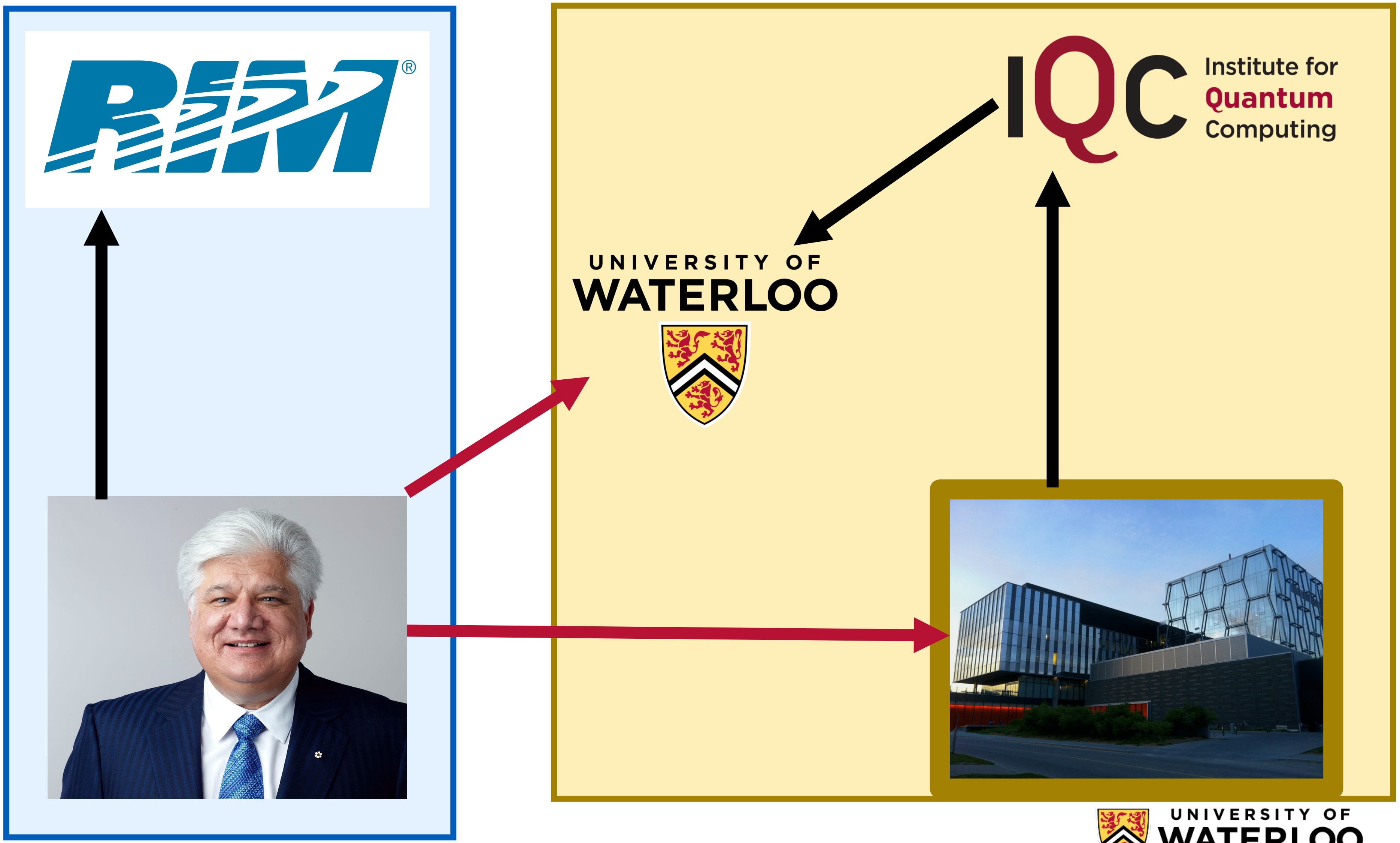


Placing Triples

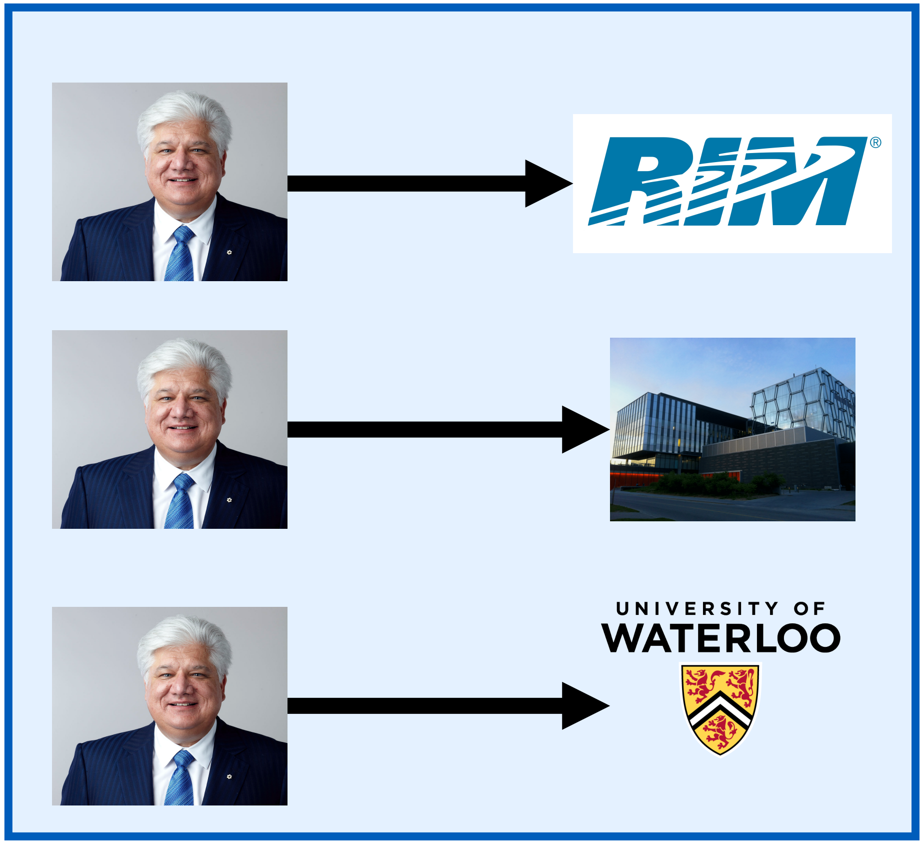


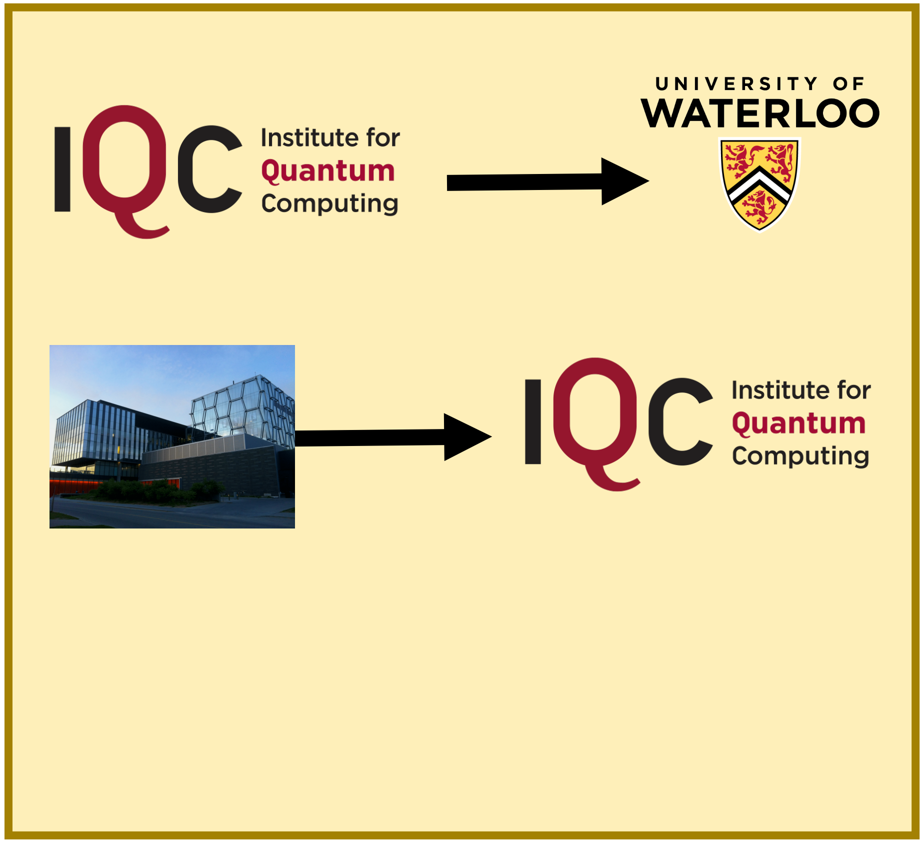


Placing Triples



Placing Triples



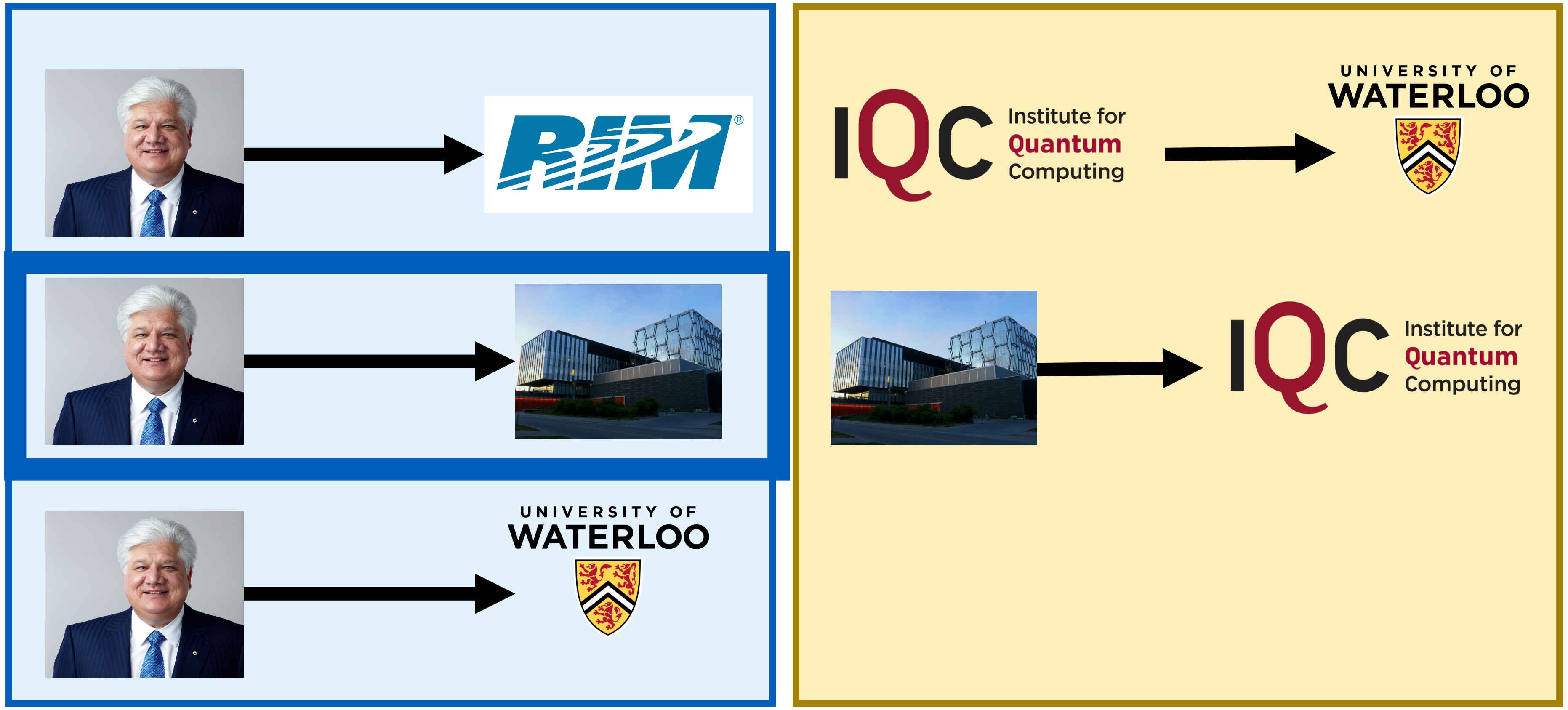

IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing

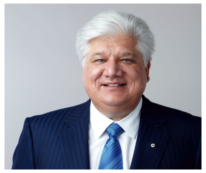
Placing Triples

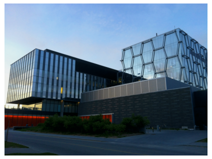
Partition RDF graph (Metis)

Place triples at **subjects partitions**

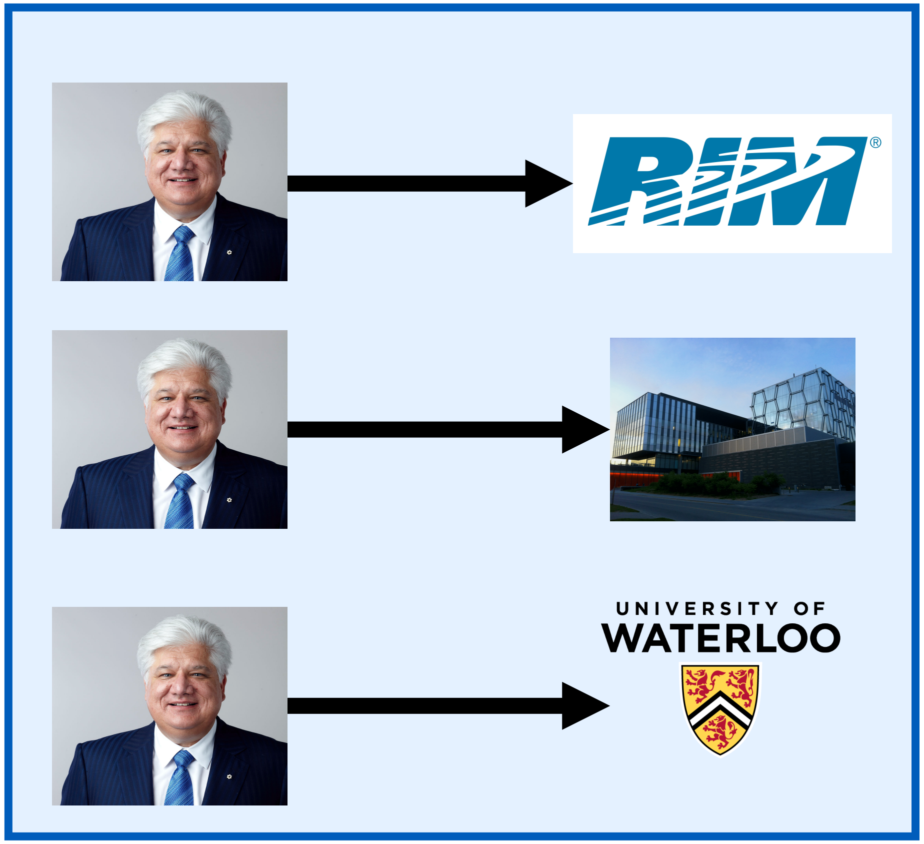
Placing Triples

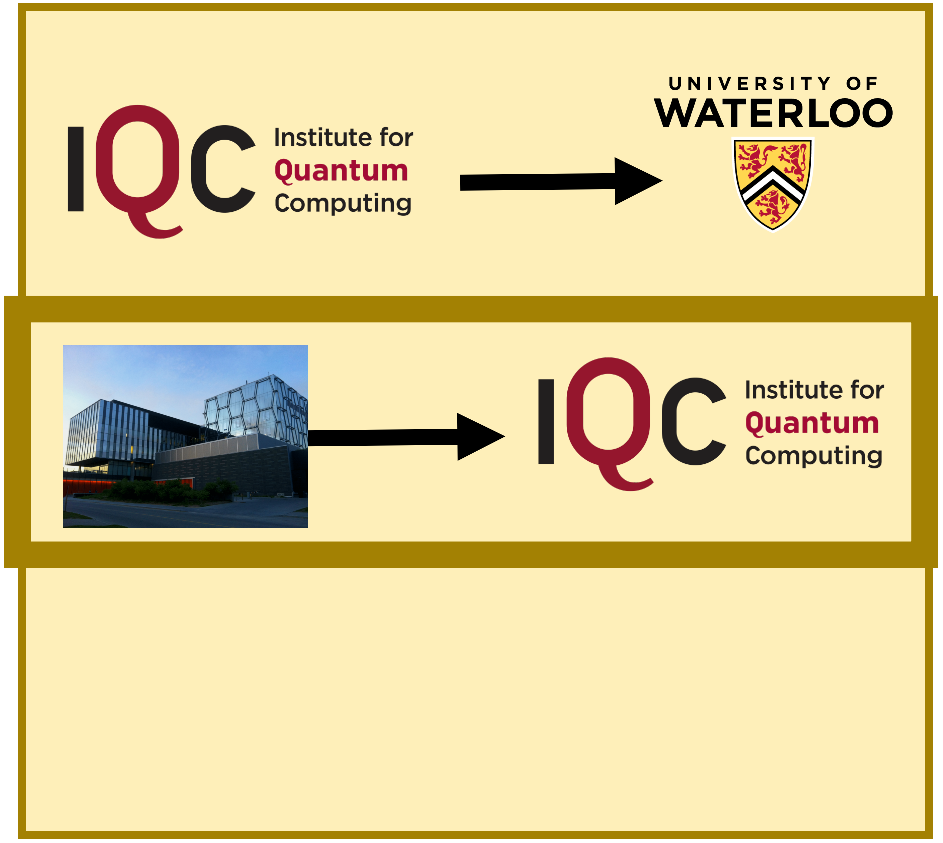

RIM
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing

Q: Find **buildings** donated by 

A: 

Placing Triples



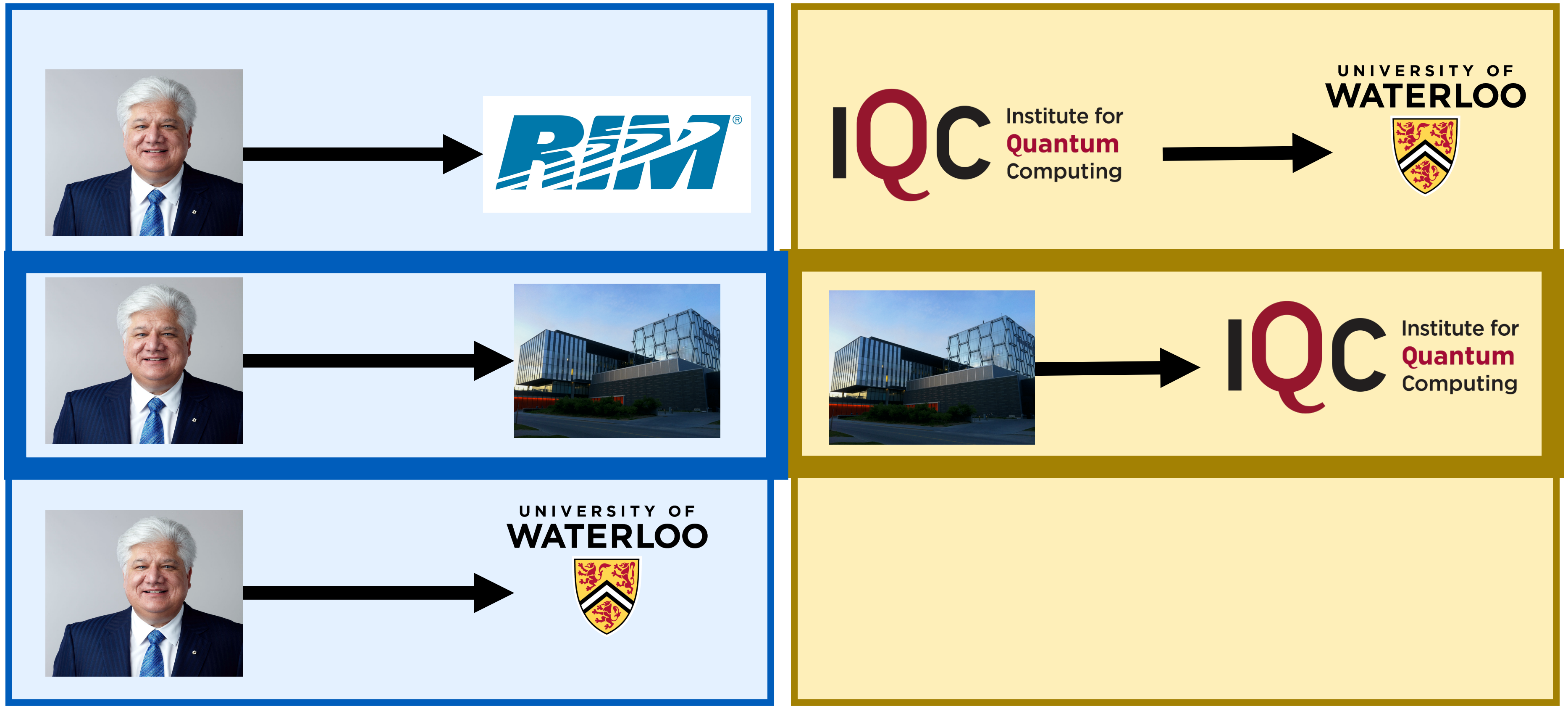

IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing

Q: Find offices in

A: IQC Institute for Quantum Computing

UNIVERSITY OF WATERLOO

Placing Triples


RIM
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO

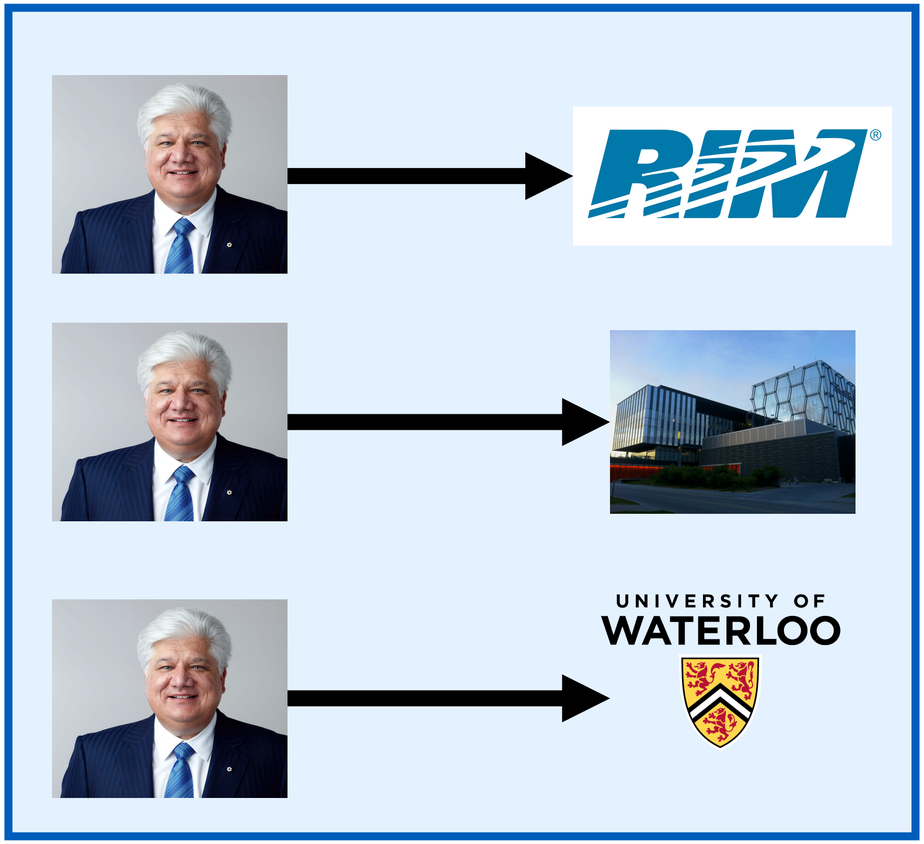
Q: Find **offices** in **buildings donated** by

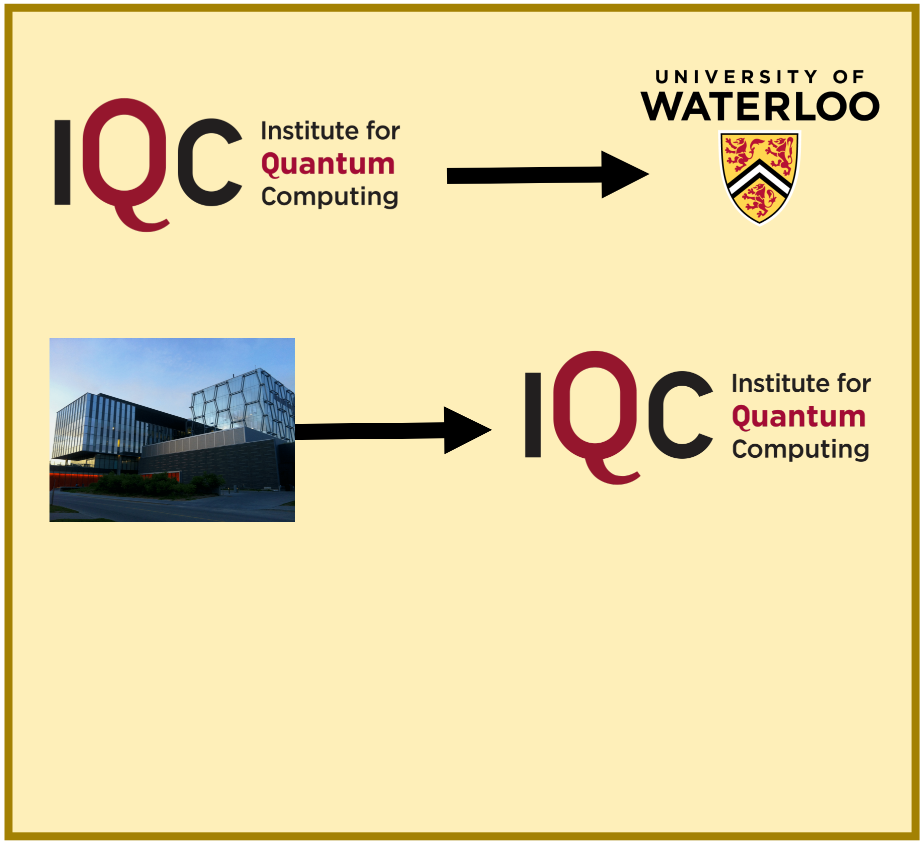
A: IQC Institute for Quantum Computing

UNIVERSITY OF WATERLOO

Distributed queries are expensive

Replicating Triples




IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing

Replicating Triples

→ RIM®

→

→ UNIVERSITY OF WATERLOO

→ IQC Institute for Quantum Computing

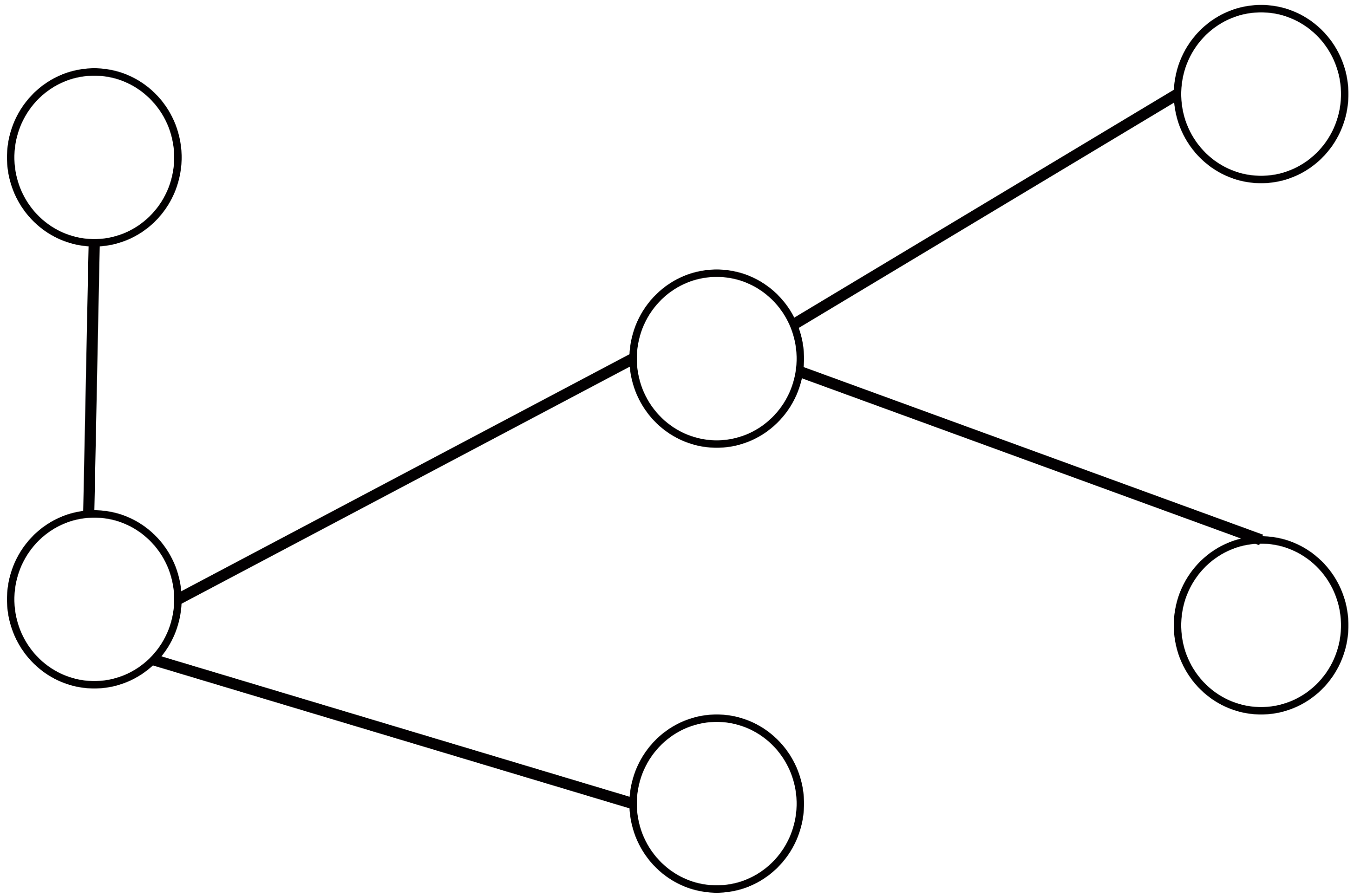
IQC Institute for Quantum Computing → UNIVERSITY OF WATERLOO

→ IQC Institute for Quantum Computing

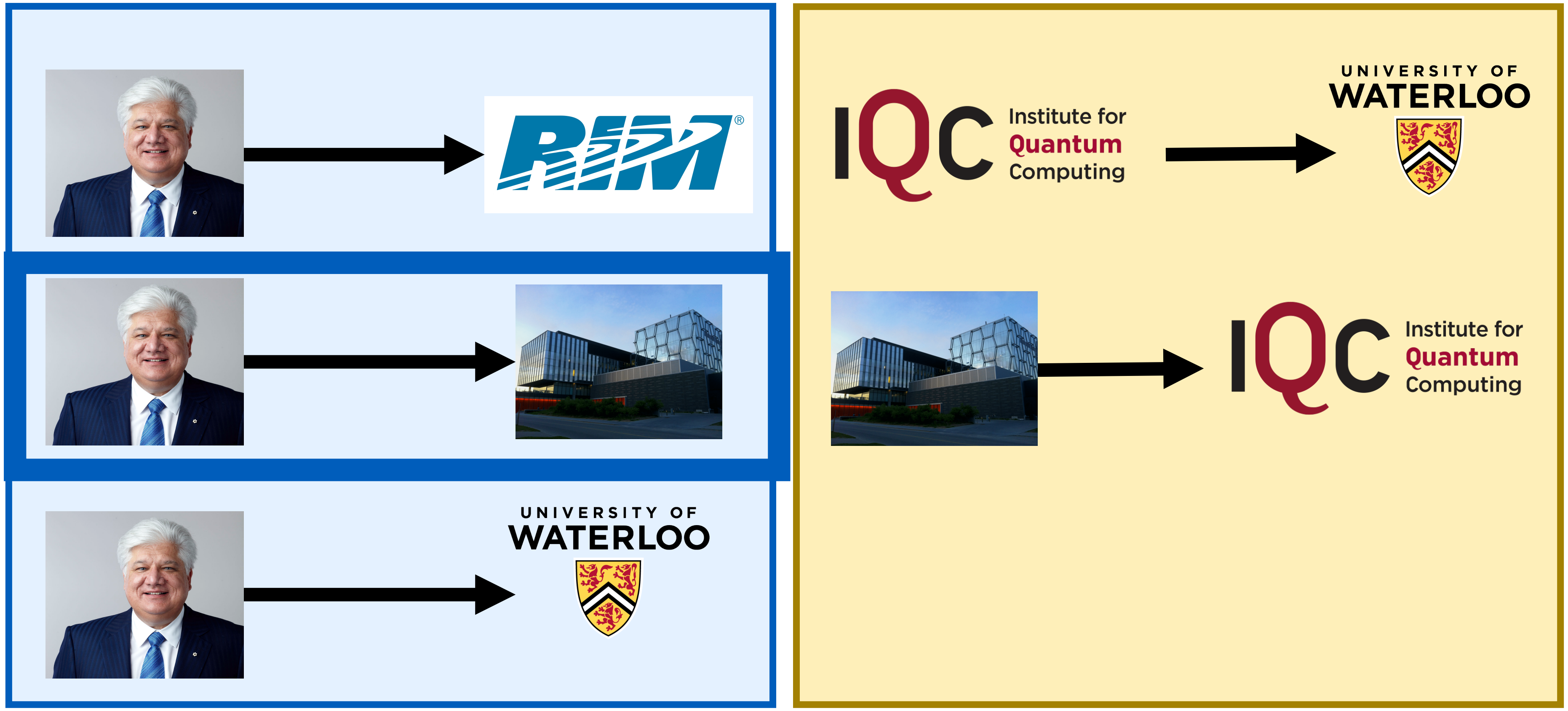
Distributed queries are expensive

Replicate (undirected) n-hop neighborhood

N-Hop Neighborhood

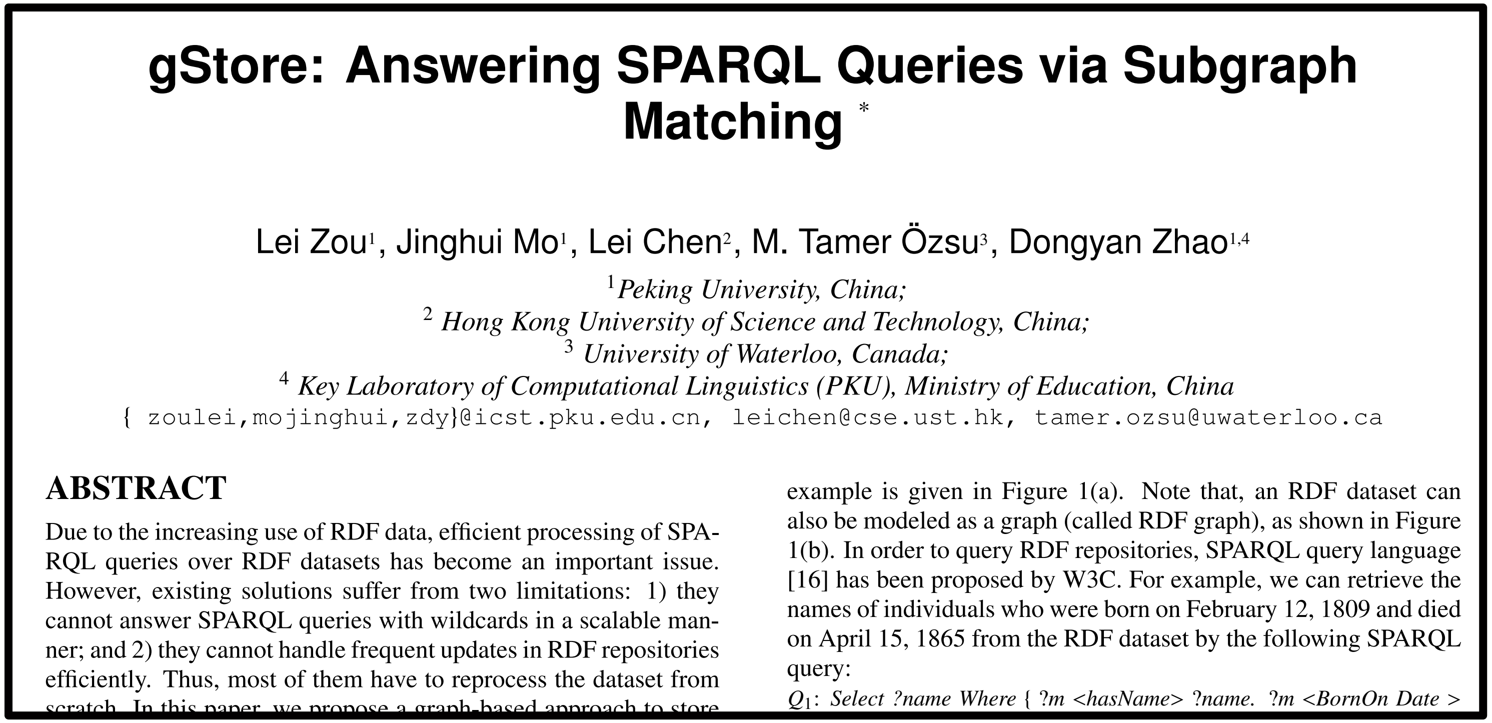


Executing Queries


RIM
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing

Execute with **existing** RDF engine

RDF query execution is subgraph matching



gStore: Answering SPARQL Queries via Subgraph Matching *

Lei Zou[1], Jinghui Mo[1], Lei Chen[2], M. Tamer Özsu[3], Dongyan Zhao[1,4]

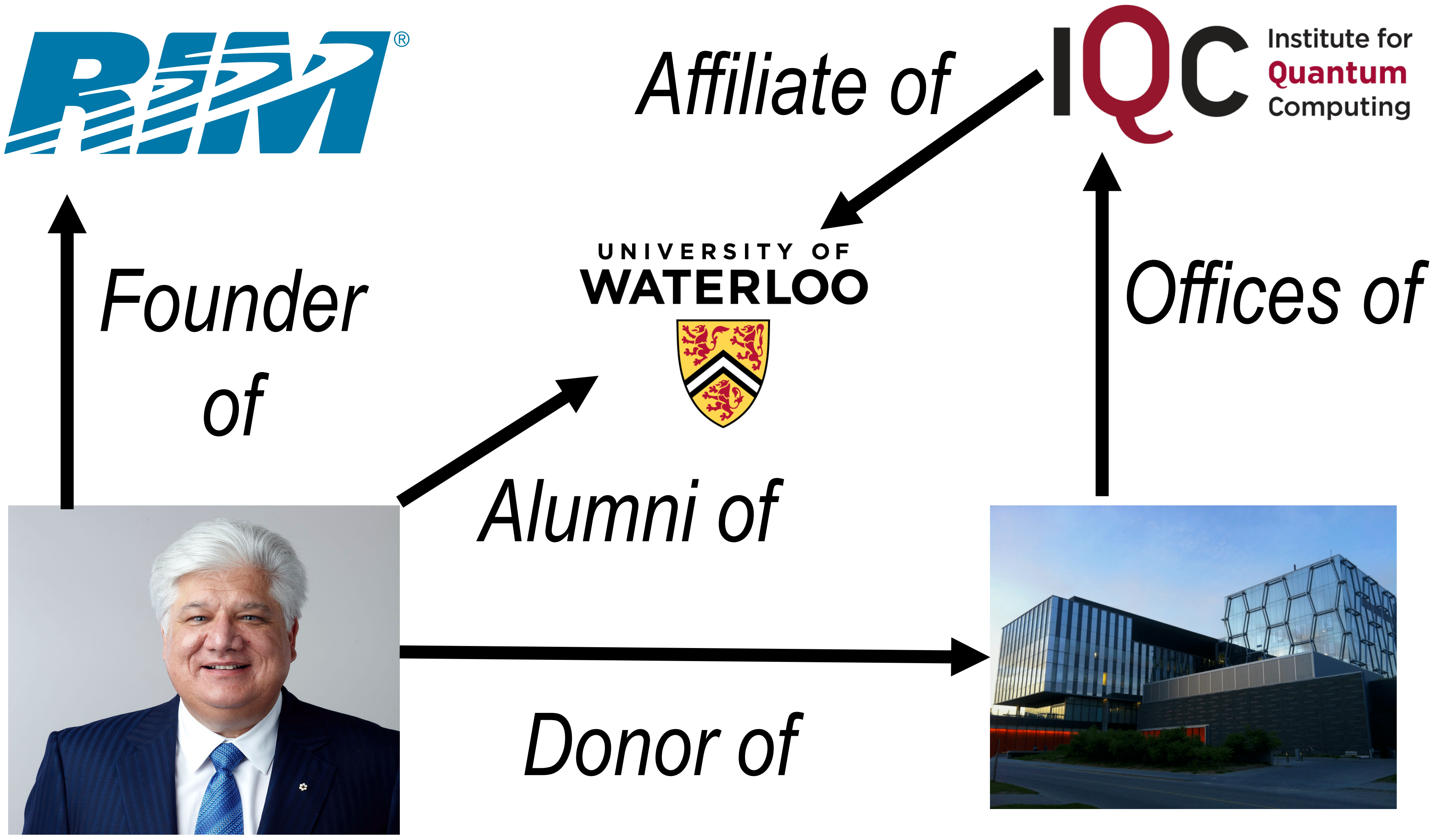
[1]*Peking University, China;*
[2] *Hong Kong University of Science and Technology, China;*
[3] *University of Waterloo, Canada;*
[4] *Key Laboratory of Computational Linguistics (PKU), Ministry of Education, China*
{ zoulei,mojinghui,zdy}@icst.pku.edu.cn, leichen@cse.ust.hk, tamer.ozsu@uwaterloo.ca

ABSTRACT

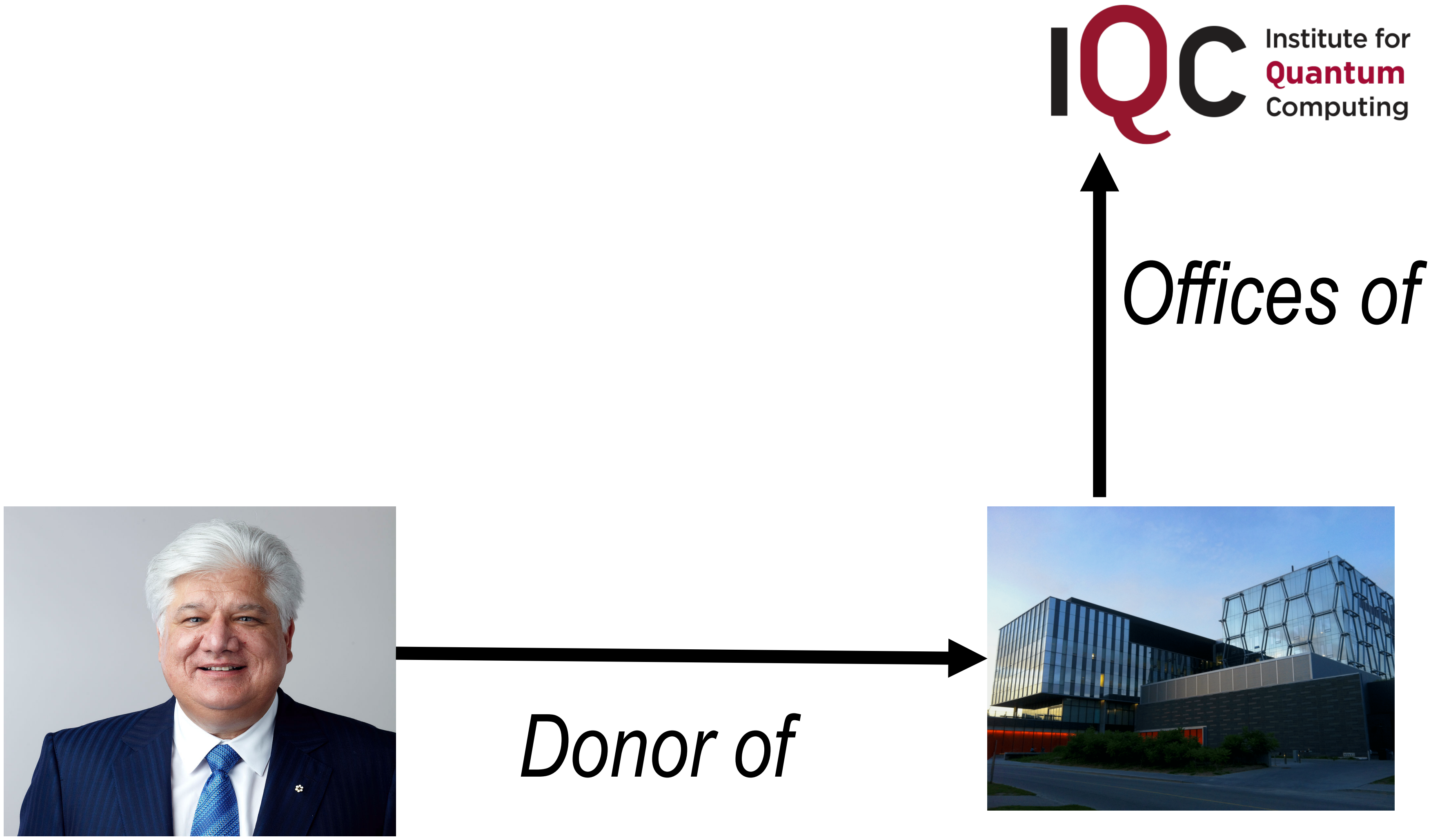
Due to the increasing use of RDF data, efficient processing of SPARQL queries over RDF datasets has become an important issue. However, existing solutions suffer from two limitations: 1) they cannot answer SPARQL queries with wildcards in a scalable manner; and 2) they cannot handle frequent updates in RDF repositories efficiently. Thus, most of them have to reprocess the dataset from scratch. In this paper, we propose a graph-based approach to store

example is given in Figure 1(a). Note that, an RDF dataset can also be modeled as a graph (called RDF graph), as shown in Figure 1(b). In order to query RDF repositories, SPARQL query language [16] has been proposed by W3C. For example, we can retrieve the names of individuals who were born on February 12, 1809 and died on April 15, 1865 from the RDF dataset by the following SPARQL query:

Q_1: *Select ?name Where { ?m <hasName> ?name. ?m <BornOn Date >*

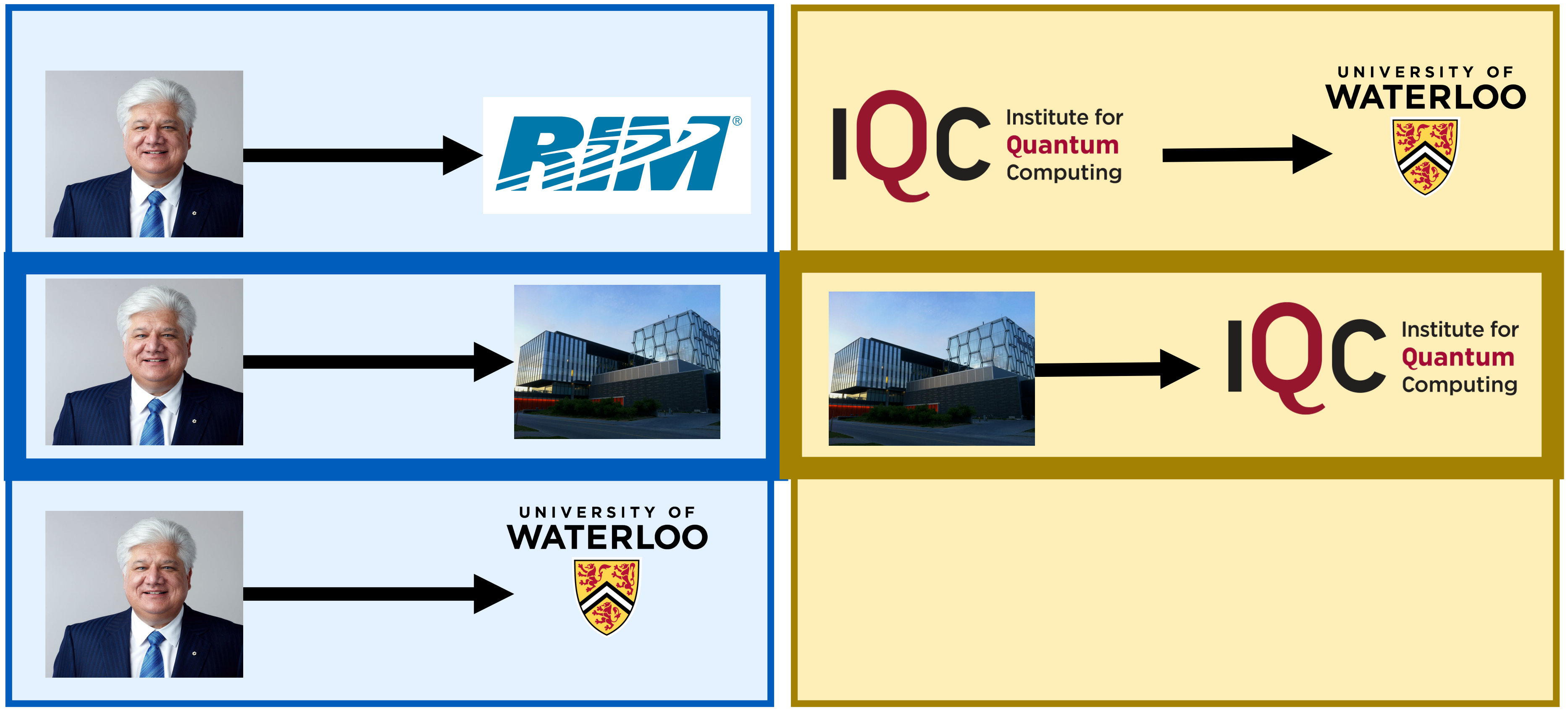


Q: Find **offices** in **buildings donated** by



Q: Find **offices** in **buildings donated** by

Executing Queries


RIM
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO
IQC Institute for Quantum Computing
UNIVERSITY OF WATERLOO

Distributed RDF query execution is distributed subgraph matching

Use Hadoop

UNIVERSITY OF WATERLOO

Executing Hadoop Queries

Decompose queries into **single node queries**

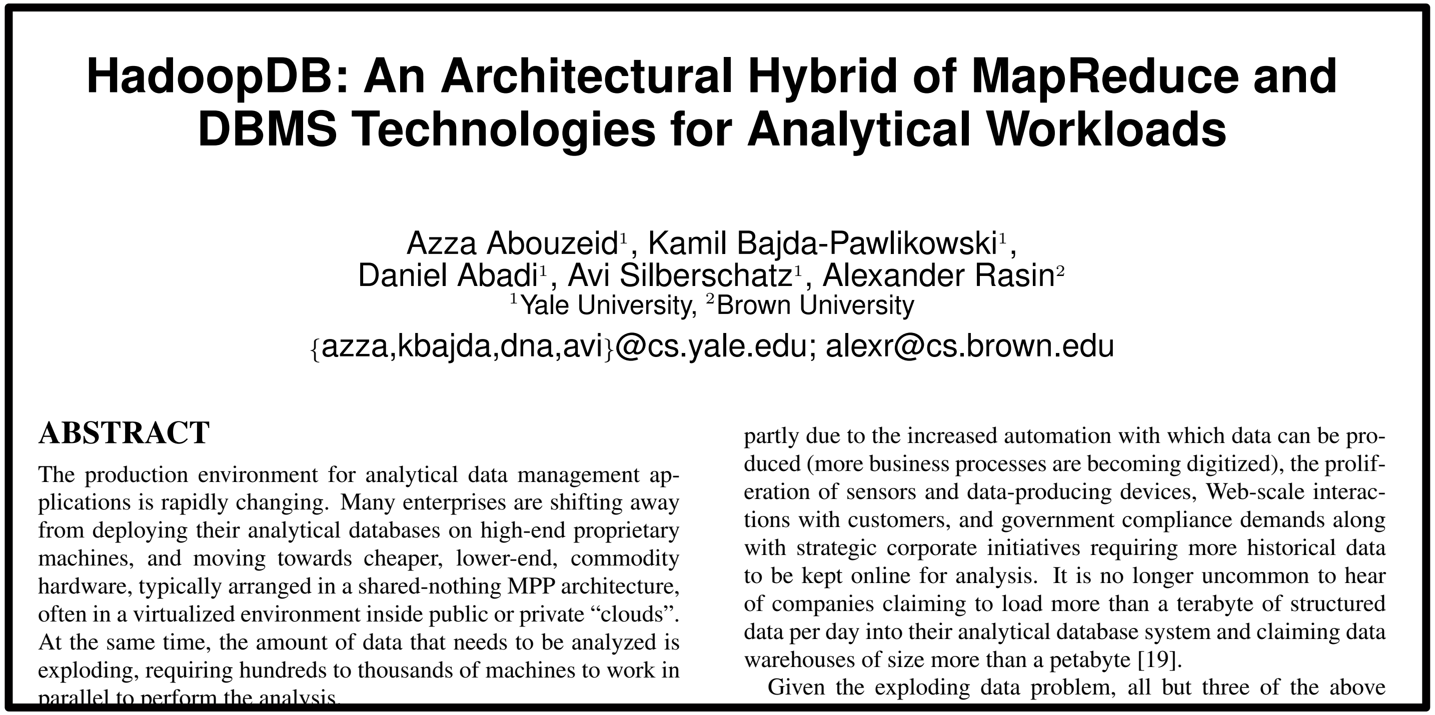
Return **non-replicated** triples from sites

Join results with Hadoop jobs

Hadoop usually has at least **20s overhead**

UNIVERSITY OF WATERLOO

Why use Hadoop?



HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads

Azza Abouzeid[1], Kamil Bajda-Pawlikowski[1],
Daniel Abadi[1], Avi Silberschatz[1], Alexander Rasin[2]
[1]Yale University, [2]Brown University
{azza,kbajda,dna,avi}@cs.yale.edu; alexr@cs.brown.edu

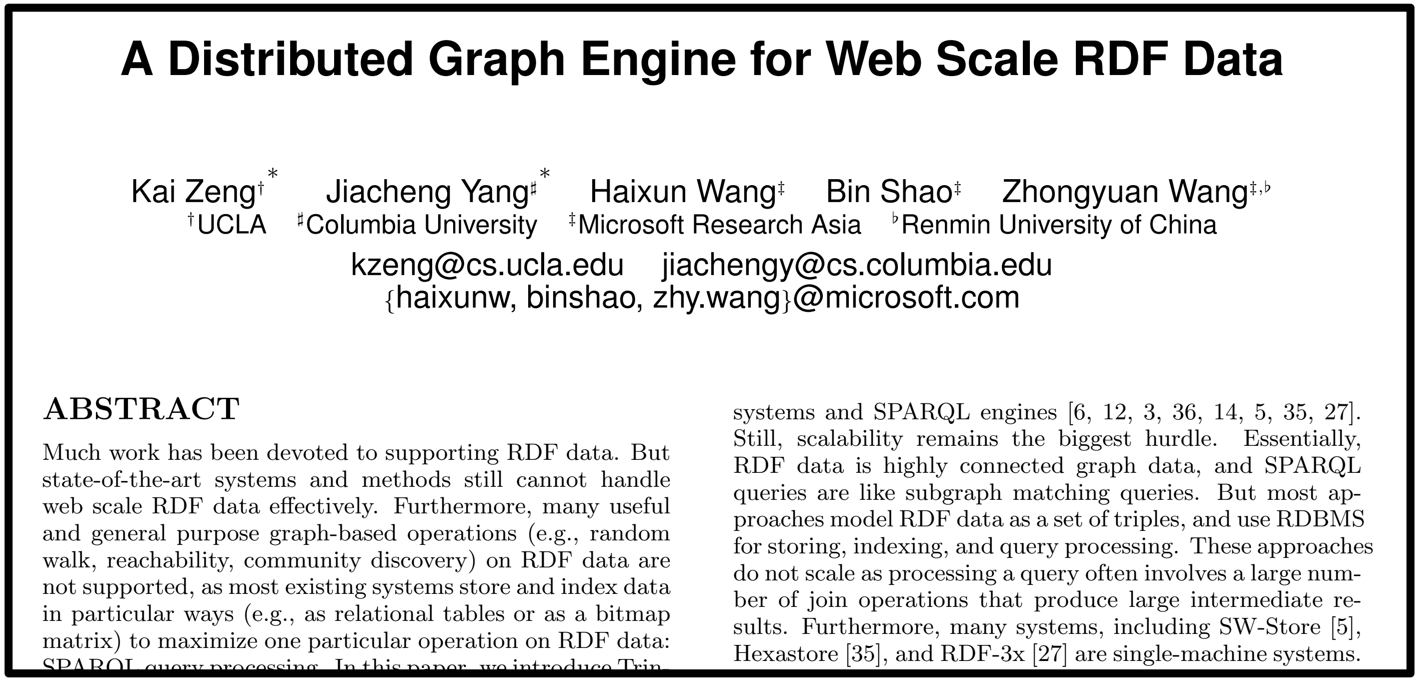
ABSTRACT

The production environment for analytical data management applications is rapidly changing. Many enterprises are shifting away from deploying their analytical databases on high-end proprietary machines, and moving towards cheaper, lower-end, commodity hardware, typically arranged in a shared-nothing MPP architecture, often in a virtualized environment inside public or private "clouds". At the same time, the amount of data that needs to be analyzed is exploding, requiring hundreds to thousands of machines to work in parallel to perform the analysis.

partly due to the increased automation with which data can be produced (more business processes are becoming digitized), the proliferation of sensors and data-producing devices, Web-scale interactions with customers, and government compliance demands along with strategic corporate initiatives requiring more historical data to be kept online for analysis. It is no longer uncommon to hear of companies claiming to load more than a terabyte of structured data per day into their analytical database system and claiming data warehouses of size more than a petabyte [19].

Given the exploding data problem, all but three of the above

Distributed RDF query execution is distributed subgraph matching



A Distributed Graph Engine for Web Scale RDF Data

Kai Zeng[†] [*] Jiacheng Yang[♯] [*] Haixun Wang[‡] Bin Shao[‡] Zhongyuan Wang[‡,♭]
[†]UCLA [♯]Columbia University [‡]Microsoft Research Asia [♭]Renmin University of China
kzeng@cs.ucla.edu jiachengy@cs.columbia.edu
{haixunw, binshao, zhy.wang}@microsoft.com

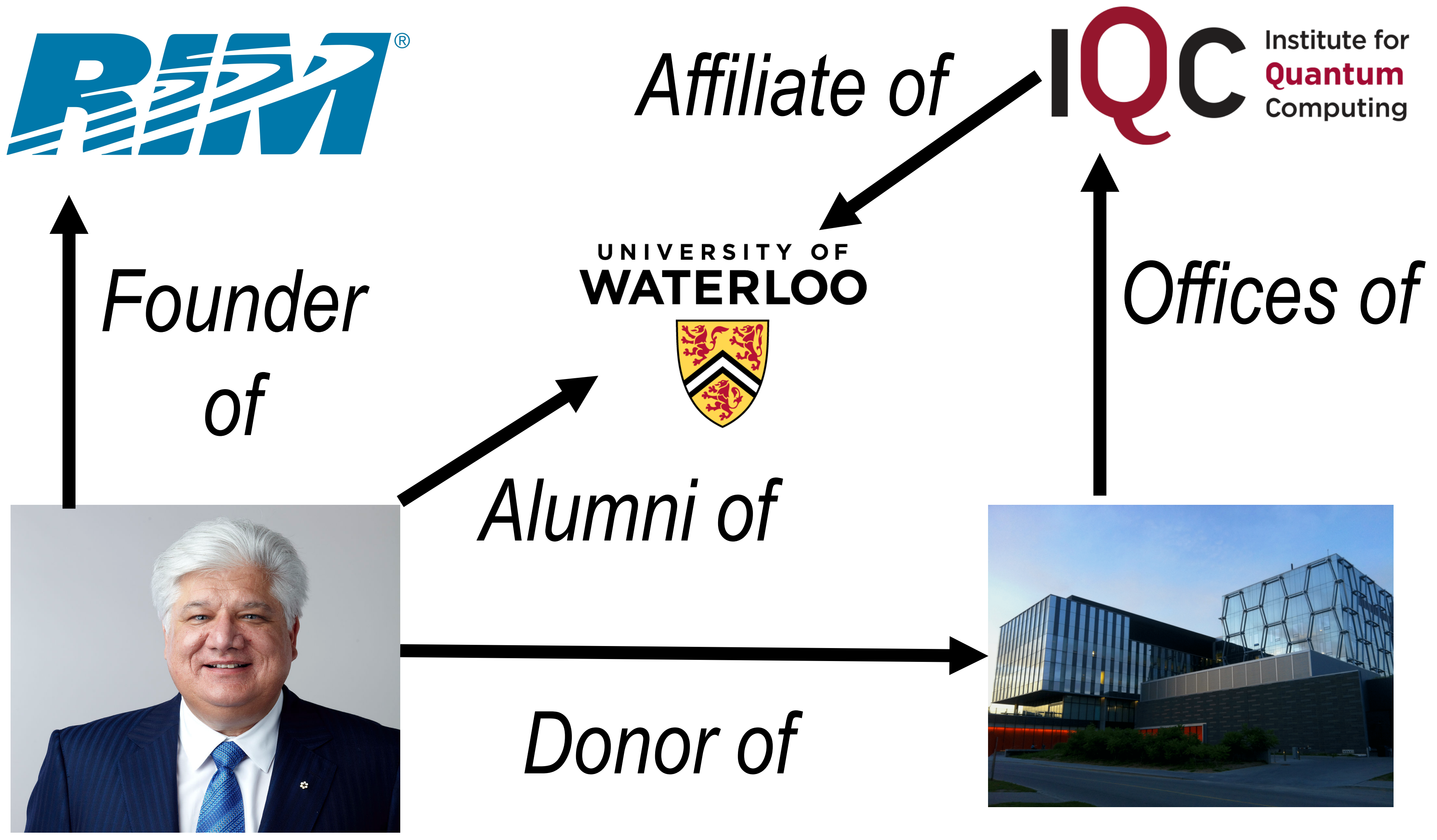
ABSTRACT

Much work has been devoted to supporting RDF data. But state-of-the-art systems and methods still cannot handle web scale RDF data effectively. Furthermore, many useful and general purpose graph-based operations (e.g., random walk, reachability, community discovery) on RDF data are not supported, as most existing systems store and index data in particular ways (e.g., as relational tables or as a bitmap matrix) to maximize one particular operation on RDF data: SPARQL query processing. In this paper, we introduce Trin-

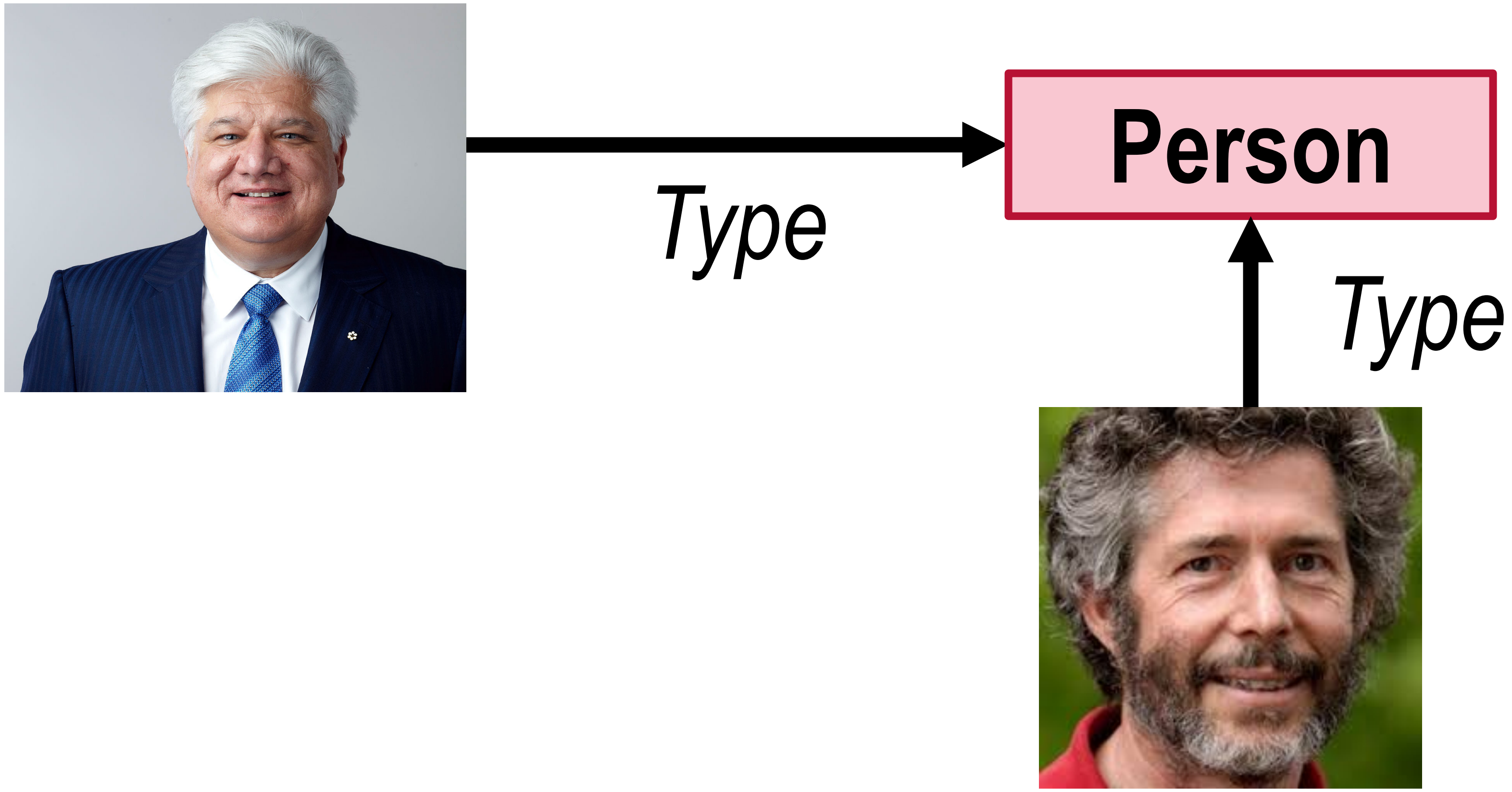
systems and SPARQL engines [6, 12, 3, 36, 14, 5, 35, 27]. Still, scalability remains the biggest hurdle. Essentially, RDF data is highly connected graph data, and SPARQL queries are like subgraph matching queries. But most approaches model RDF data as a set of triples, and use RDBMS for storing, indexing, and query processing. These approaches do not scale as processing a query often involves a large number of join operations that produce large intermediate results. Furthermore, many systems, including SW-Store [5], Hexastore [35], and RDF-3x [27] are single-machine systems.

Beats Hadoop

Optimizations

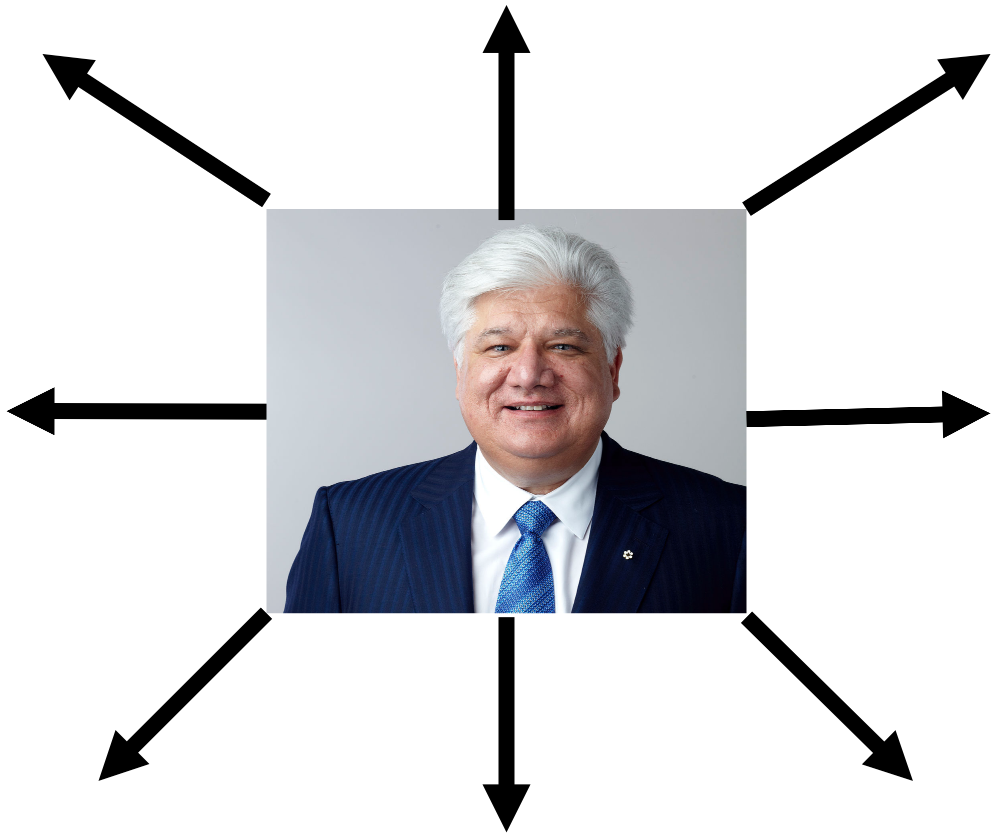


Optimizations



Remove *type* predicated triples from graph partitioning

Optimizations



Remove *high degree* vertices from graph partitioning

UNIVERSITY OF WATERLOO

Evaluation

Compare:

1 & 2 hop **graph partitioning**

Hash partitioning

Single node

Shard (Hadoop based)

Evaluation

1 & 2 hop graph partitioning:

Slows down fast queries

Speeds up slow queries (5x)

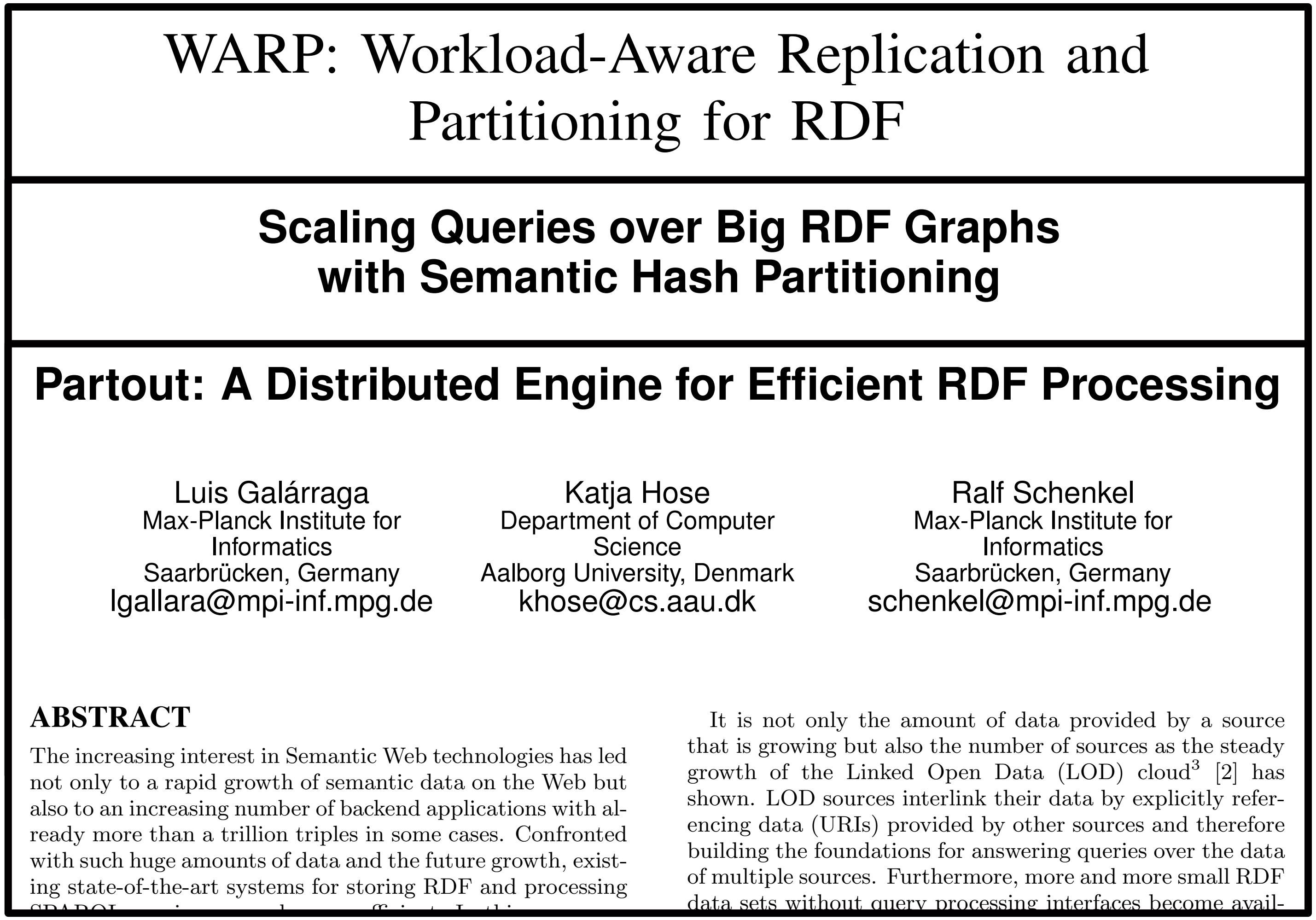
Takes longer to load (2-8x)

Increases space used (4x)

UNIVERSITY OF WATERLOO

Workload Aware RDF

Schism like partitioning & replication



WARP: Workload-Aware Replication and Partitioning for RDF

Scaling Queries over Big RDF Graphs with Semantic Hash Partitioning

Partout: A Distributed Engine for Efficient RDF Processing

Luis Galárraga
Max-Planck Institute for Informatics
Saarbrücken, Germany
lgallara@mpi-inf.mpg.de

Katja Hose
Department of Computer Science
Aalborg University, Denmark
khose@cs.aau.dk

Ralf Schenkel
Max-Planck Institute for Informatics
Saarbrücken, Germany
schenkel@mpi-inf.mpg.de

ABSTRACT

The increasing interest in Semantic Web technologies has led not only to a rapid growth of semantic data on the Web but also to an increasing number of backend applications with already more than a trillion triples in some cases. Confronted with such huge amounts of data and the future growth, existing state-of-the-art systems for storing RDF and processing [illegible]

It is not only the amount of data provided by a source that is growing but also the number of sources as the steady growth of the Linked Open Data (LOD) cloud[3] [2] has shown. LOD sources interlink their data by explicitly referencing data (URIs) provided by other sources and therefore building the foundations for answering queries over the data of multiple sources. Furthermore, more and more small RDF data sets without query processing interfaces become avail-

Summary

Scale RDF systems by using:

Graph partitioning of data

Vertex neighborhood **replication**

Scale RDF systems by scaling graph systems

UNIVERSITY OF WATERLOO

Discussion

Purpose and overhead of **Hadoop**

Space and maintenance **cost** of broad **replication** strategy

Can we use **general graph systems not RDF?**

UNIVERSITY OF WATERLOO