Streaming Graph Processing & Analytics

M. Tamer Özsu

University of Waterloo
David R. Cheriton School of Computer Science
https://cs.uwaterloo.ca/~tozsu

Graphs have become ubiquitous

Forbes | Billionaires | Innovation | Leadership | Money | Business | Small Busin

12,073 views | Jul 18, 2019, 06:17pm EDT

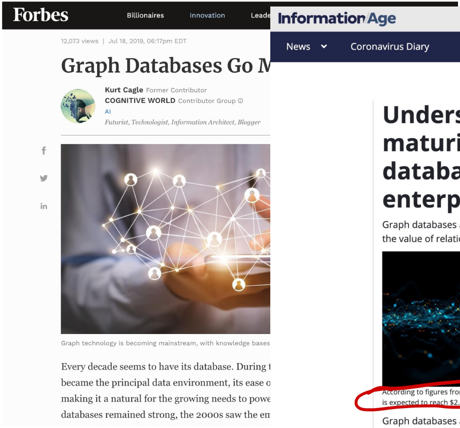
Graph Databases Go Mainstream

Kurt Cagle Former Contributor
COGNITIVE WORLD Contributor Group
AI
Futurist, Technologist, Information Architect, Blogger

Graph technology is becoming mainstream, with knowledge bases leading the way. GETTY

Every decade seems to have its database. During the 1990s, the relational database became the principal data environment, its ease of use and tabular arrangement making it a natural for the growing needs to power the data web. While relational databases remained strong, the 2000s saw the emergence of XML databases, and

Graphs have become ubiquitous



Forbes Billionaires Innovation Leade

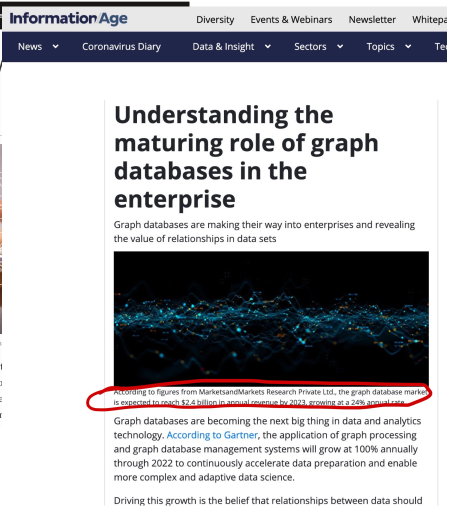
12,073 views | Jul 18, 2019, 06:17pm EDT

Graph Databases Go M

Kurt Cagle Former Contributor
COGNITIVE WORLD Contributor Group
AI
Futurist, Technologist, Information Architect, Blogger

Graph technology is becoming mainstream, with knowledge bases

Every decade seems to have its database. During t
became the principal data environment, its ease o
making it a natural for the growing needs to powe
databases remained strong, the 2000s saw the em



Information Age Diversity Events & Webinars Newsletter Whitepa

News Coronavirus Diary Data & Insight Sectors Topics Te

Understanding the maturing role of graph databases in the enterprise

Graph databases are making their way into enterprises and revealing the value of relationships in data sets

According to figures from MarketsandMarkets Research Private Ltd., the graph database market is expected to reach $2.4 billion in annual revenue by 2023, growing at a 24% annual rate

Graph databases are becoming the next big thing in data and analytics technology. According to Gartner, the application of graph processing and graph database management systems will grow at 100% annually through 2022 to continuously accelerate data preparation and enable more complex and adaptive data science.

Driving this growth is the belief that relationships between data should

Graphs have become ubiquitous

Forbes Billionaires Innovation Leade

12,073 views | Jul 18, 2019, 06:17pm EDT

Graph Databases Go M

Kurt Cagle Former Contributor
COGNITIVE WORLD Contributor Group
AI
Futurist, Technologist, Information Architect, Blogger

Graph technology is becoming mainstream, with knowledge bases

Every decade seems to have its database. During t
became the principal data environment, its ease o
making it a natural for the growing needs to powe
databases remained strong, the 2000s saw the em

Information Age Diversity Events

News Coronavirus Diary Data & Insight

Understanding maturing role databases in t enterprise

Graph databases are making their way
the value of relationships in data sets

According to figures from MarketsandMarkets Rese
is expected to reach $2.4 billion in annual revenue

Graph databases are becoming the ne
technology. According to Gartner, the
and graph database management syst
through 2022 to continuously accelera
more complex and adaptive data scier

Driving this growth is the belief that re

ENTERPRISE MANAGEMENT 360 CXO OF THE WEEK TOP 10 TECH PODCASTS WHITEPAPERS VID

AI IN THE ENTERPRISE BUSINESS AGILITY BUSINESS CONTINUITY UNIFIED C

Home › Why do experts say graph databases are headed for mainstream use?

Share

Why do experts say graph databases are headed for mainstream use?

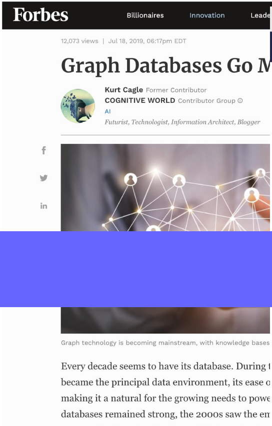
Graph databases are becoming mainstream, but how can this technology improve your data management?

24th July 2019

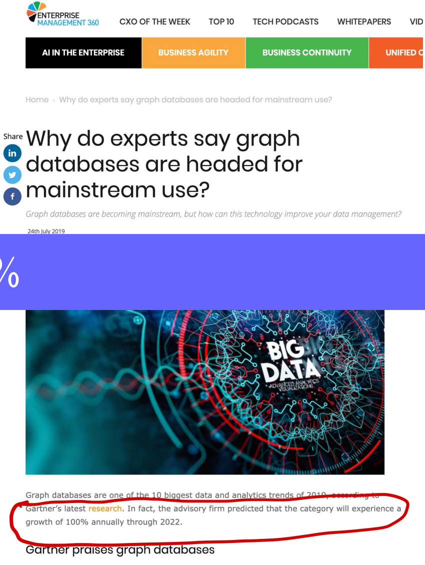
Graph databases are one of the 10 biggest data and analytics trends of 2019, according to Gartner's latest research. In fact, the advisory firm predicted that the category will experience a growth of 100% annually through 2022.

Gartner praises graph databases

Graphs have become ubiquitous


Forbes
Billionaires Innovation Leade
12,073 views | Jul 18, 2019, 06:17pm EDT
Graph Databases Go M
Kurt Cagle Former Contributor
COGNITIVE WORLD Contributor Group
AI
Futurist, Technologist, Information Architect, Blogger
Graph technology is becoming mainstream, with knowledge bases
Every decade seems to have its database. During t
became the principal data environment, its ease o
making it a natural for the growing needs to powe
databases remained strong, the 2000s saw the em


Information Age
Diversity Events
News Coronavirus Diary Data & Insight
Understanding
maturing role
databases in t
enterprise
Graph databases are making their way
According to figures from MarketsandMarkets Rese
is expected to reach $2.4 billion in annual revenue
Graph databases are becoming the ne
technology. According to Gartner, the
and graph database management syst
through 2022 to continuously accelera
more complex and adaptive data scien
Driving this growth is the belief that re


ENTERPRISE MANAGEMENT 360
CXO OF THE WEEK TOP 10 TECH PODCASTS WHITEPAPERS VID
AI IN THE ENTERPRISE BUSINESS AGILITY BUSINESS CONTINUITY UNIFIED C
Home › Why do experts say graph databases are headed for mainstream use?
Share
Why do experts say graph databases are headed for mainstream use?
Graph databases are becoming mainstream, but how can this technology improve your data management?
24th July 2019
BIG DATA
Graph databases are one of the 10 biggest data and analytics trends of 2019, according to Gartner's latest research. In fact, the advisory firm predicted that the category will experience a growth of 100% annually through 2022.
Gartner praises graph databases

CAGR $>$ 20%

Graphs – When Relationships are Important

Recent COVID-19 pandemic

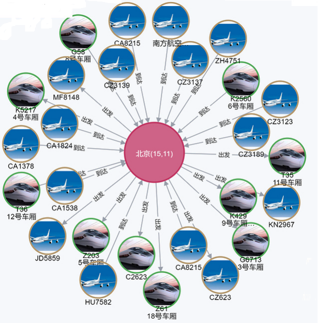
- Model how people interact and influence each other, and how ideas and behaviours travel along social pathways

(A. Woodie, "Tracking the Spread of Coronavirus with Graph Databases", https://bit.ly/2UuScbM)

Graphs – When Relationships are Important

Recent COVID-19 pandemic

- Model how people interact and influence each other, and how ideas and behaviours travel along social pathways
- Epidemic search
  - Self assessment by checking connections
  - {Place, flight, train, license plate} $\rightarrow$ {known cases}
  - {Source loc, Target loc} $\rightarrow$ {“edges” that connect them, flights, trains, vehicle license plates}



(A. Woodie, “Tracking the Spread of Coronavirus with Graph Databases”, https://bit.ly/2UuScbM)

Graphs – When Relationships are Important

Recent COVID-19 pandemic

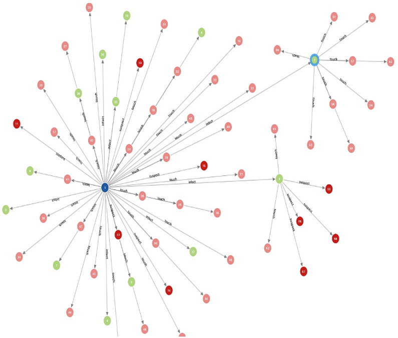
- Model how people interact and influence each other, and how ideas and behaviours travel along social pathways
- Epidemic search
- Complex COVID-19 pathways
  - Looking at propagation in social networks [Kempe et al., 2003]
  - Linear threshold model
  - Independent cascade model

(A. Woodie, "Tracking the Spread of Coronavirus with Graph Databases", https://bit.ly/2UuScbM)

Graphs – When Relationships are Important

Recent COVID-19 pandemic

- Model how people interact and influence each other, and how ideas and behaviours travel along social pathways
- Epidemic search
- Complex COVID-19 pathways
- Contact tracing
    - Figuring out exactly how 5 people became infected in Tianjin
    - Vertices: people and places they traveled
    - Edges: people-people contact or travel
    - Paths: how infections link to known cases

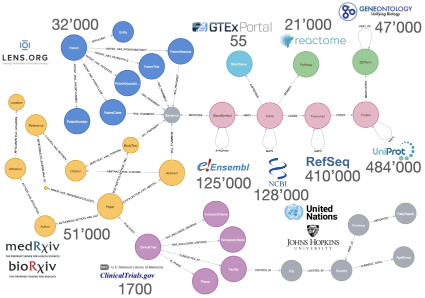


(A. Woodie, "Tracking the Spread of Coronavirus with Graph Databases", https://bit.ly/2UuScbM)

Graphs – When Relationships are Important

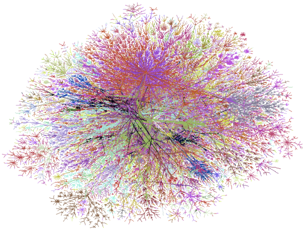
Recent COVID-19 pandemic

- Model how people interact and influence each other, and how ideas and behaviours travel along social pathways
- Epidemic search
- Complex COVID-19 pathways
- Contact tracing
- Covid knowledge graph

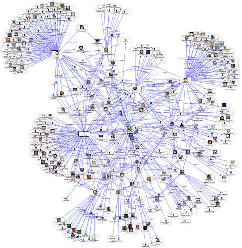


(https://covidgraph.org)

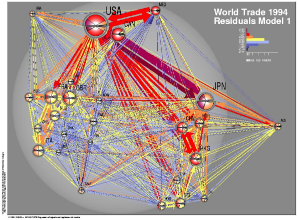
Modern graphs are different and diverse



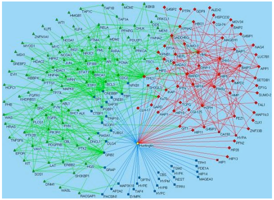
Internet



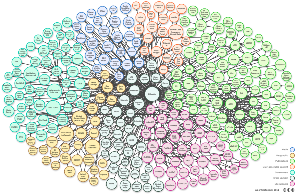
Social networks



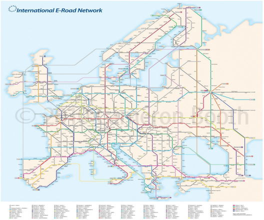
Trade volumes & connections



Biological networks



Linked data



Road network

Graph Usage Study

The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing

Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, M. Tamer Özsu
David R. Cheriton School of Computer Science
University of Waterloo
{s3sahu,amine.mhedhbi,semih.salihoglu,jimmylin,tamer.ozsu}@uwaterloo.ca

ABSTRACT

Graph processing is becoming increasingly prevalent across many application domains. In spite of this prevalence, there is little research about how graphs are actually used in practice. We conducted an online survey aimed at understanding: (i) the types of graphs users have; (ii) the graph computations users run; (iii) the types of graph software users use; and (iv) the major challenges users face when processing their graphs. We describe the participants' responses to our questions highlighting common patterns and challenges. We further reviewed user feedback in the mailing lists, bug reports, and feature requests in the source repositories of a large suite of software products for processing graphs. Through our review, we were able to answer some new questions that were raised by participants' responses and identify specific challenges that users face when using different classes of graph software. The participants' responses and data we obtained revealed surprising facts about graph processing in practice. In particular, real-world graphs represent a very diverse range of entities and are often very large, and scalability and visualization are undeniably the most pressing challenges faced by participants. We hope these findings can guide future research.

PVLDB Reference Format:
Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, and M. Tamer Özsu. The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing. *PVLDB*, 11(4): xxxx-yyyy, 2017.
DOI: https://doi.org/10.1145/3164135.3164139

1. INTRODUCTION

Graph data representing connected entities and their relationships appear in many application domains, most naturally in social networks, the web, the semantic web, road maps, communication networks, biology, and finance, just to name a few examples. There has been a noticeable increase in the prevalence of work on graph processing both in research and in practice, evidenced by the surge in the number of different commercial and research software for managing and processing graphs. Examples include graph database systems [3, 8, 14, 35, 48, 53], RDF engines [38, 64, 67], linear algebra software [6, 46], visualization software [13, 16], query languages [28, 52, 55], and distributed graph processing systems [17, 21, 27]. In the academic literature, a large number of publications that study numerous topics related to graph processing regularly appear across a wide spectrum of research venues.

Despite their prevalence, there is little research on how graph data is actually used in practice and the major challenges facing users of graph data, both in industry and research. In April 2017, we conducted an online survey across 89 users of 22 different software products, with the goal of answering 4 high-level questions:

(i) What types of graph data do users have?
(ii) What computations do users run on their graphs?
(iii) Which software do users use to perform their computations?
(iv) What are the major challenges users face when processing their graph data?

Our major findings are as follows:

- *Variety:* Graphs in practice represent a very wide variety of entities, many of which are not naturally thought of as vertices and edges. Most surprisingly, traditional enterprise data comprised of products, orders, and transactions, which are typically seen as the perfect fit for relational systems, appear to be a very common form of data represented in participants' graphs.
- *Ubiquity of Very Large Graphs:* Many graphs in practice are very large, often containing over a billion edges. These large graphs represent a very wide range of entities and belong to organizations at all scales from very small enterprises to very large ones. This refutes the sometimes heard assumption that large graphs are a problem for only a few large organizations such as Google, Facebook, and Twitter.
- *Challenge of Scalability:* Scalability is unequivocally the most pressing challenge faced by participants. The ability to process very large graphs efficiently seems to be the biggest limitation of existing software.
- *Visualization:* Visualization is a very popular and central task in participants' graph processing pipelines. After scalability, participants indicated visualization as their second most pressing challenge, tied with challenges in graph query languages.
- *Prevalence of RDBMSes:* Relational databases still play an important role in managing and processing graphs.

Our survey also highlights other interesting facts, such as the prevalence of machine learning on graph data, e.g., for clustering vertices, predicting links, and finding influential vertices.

We further reviewed user feedback in the mailing lists, bug reports, and feature requests in the source code repositories of 22 software products between January and September of 2017 with two goals: (i) to answer several new questions that the participants' responses raised; and (ii) to identify more specific challenges in different classes of graph technologies than the ones we could iden-

[Sahu et al., 2017, 2020]

The VLDB Journal
https://doi.org/10.1007/s00778-019-00548-x

SPECIAL ISSUE PAPER

The ubiquity of large graphs and surprising challenges of graph processing: extended survey

Siddhartha Sahu[1] · Amine Mhedhbi[1] · Semih Salihoglu[1] · Jimmy Lin[1] · M. Tamer Özsu[1]

Received: 21 January 2019 / Revised: 9 May 2019 / Accepted: 13 June 2019

Abstract
Graph processing is becoming increasingly prevalent across many application domains. In spite of this prevalence, there is little research about how graphs are actually used in practice. We performed an extensive study that consisted of an online survey of 89 users, a review of the mailing lists, source repositories, and white papers of a large suite of graph software products, and in-person interviews with 6 users and 2 developers of these products. Our online survey aimed at understanding: (i) the types of graphs users have; (ii) the graph computations users run; (iii) the types of graph software users use; and (iv) the major challenges users face when processing their graphs. We describe the participants' responses to our questions highlighting common patterns and challenges. Based on our interviews and survey of the rest of our sources, we were able to answer some new questions that were raised by participants' responses to our online survey and understand the specific applications that use graph data and software. Our study revealed surprising facts about graph processing in practice. In particular, real-world graphs represent a very diverse range of entities and are often very large, scalability and visualization are undeniably the most pressing challenges faced by participants, and data integration, recommendations, and fraud detection are very popular applications supported by existing graph software. We hope these findings can guide future research.

Keywords User survey · Graph processing · Graph databases · RDF systems

1 Introduction

Graph data representing connected entities and their relationships appear in many application domains, most naturally in social networks, the Web, the Semantic Web, road maps, communication networks, biology, and finance, just to name a few examples. There has been a noticeable increase in the prevalence of work on graph processing both in research and in practice, evidenced by the surge in the number of different commercial and research software for managing and processing graphs. Examples include graph database systems [13,20,26,49,65,73,90], RDF engines [52,96], linear algebra software [17,63], visualization software [25,29], query languages [41,72,78], and distributed graph processing systems [30,34,40]. In the academic literature, a large number of publications that study numerous topics related to graph processing regularly appear across a wide spectrum of research venues.

Despite their prevalence, there is little research on how graph data are actually used in practice and the major challenges facing users of graph data, both in industry and in research. In April 2017, we conducted an online survey across 89 users of 22 different software products, with the goal of answering 4 high-level questions:

(i) What types of graph data do users have?
(ii) What computations do users run on their graphs?
(iii) Which software do users use to perform their computations?

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00778-019-00548-x) contains supplementary material, which is available to authorized users.

Siddhartha Sahu
s3sahu@uwaterloo.ca

Amine Mhedhbi
amine.mhedhbi@uwaterloo.ca

Semih Salihoglu
semih.salihoglu@uwaterloo.ca

Jimmy Lin
jimmylin@uwaterloo.ca

M. Tamer Özsu
tamer.ozsu@uwaterloo.ca

[1] University of Waterloo, Waterloo, Canada

Published online: 29 June 2019

Springer

Graph Usage Study

[Sahu et al., 2017, 2020]

The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing

Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, M. Tamer Özsu
David R. Cheriton School of Computer Science
University of Waterloo
{s3sahu,amine.mhedhbi,semih.salihoglu,jimmylin,tamer.ozsu}@uwaterloo.ca

The VLDB Journal
https://doi.org/10.1007/s00778-019-00548-x

SPECIAL ISSUE PAPER

The ubiquity of large graphs and surprising challenges of graph processing: extended survey

Siddhartha Sahu[1] · Amine Mhedhbi[1] · Semih Salihoglu[1] · Jimmy Lin[1] · M. Tamer Özsu[1]

Objectives

1. What kind of graph data, computations, software, and major challenges real users have in practice?
2. What types of graph data, computations, software, and major challenges researchers target in publications?

future research.

PVLDB Reference Format:
Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, and M. Tamer Özsu. The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing. *PVLDB*, 11(4): xxxx-yyyy, 2017.
DOI: https://doi.org/10.1145/3164135.3164139

1. INTRODUCTION

Graph data representing connected entities and their relationships appear in many application domains, most naturally in social networks, the web, the semantic web, road maps, communication networks, biology, and finance, just to name a few examples. There has been a noticeable increase in the prevalence of work on graph processing both in research and in practice, evidenced by the surge in the number of different commercial and research software for managing and processing graphs. Examples include graph database systems [3, 8, 14, 35, 48, 53], RDF engines [38, 64, 67], linear algebra software [6, 46], visualization software [13, 16], query languages [28,

form of data represented in participants' graphs.

- *Ubiquity of Very Large Graphs:* Many graphs in practice are very large, often containing over a billion edges. These large graphs represent a very wide range of entities and belong to organizations at all scales from very small enterprises to very large ones. This refutes the sometimes heard assumption that large graphs are a problem for only a few large organizations such as Google, Facebook, and Twitter.
- *Challenge of Scalability:* Scalability is unequivocally the most pressing challenge faced by participants. The ability to process very large graphs efficiently seems to be the biggest limitation of existing software.
- *Visualization:* Visualization is a very popular and central task in participants' graph processing pipelines. After scalability, participants indicated visualization as their second most pressing challenge, tied with challenges in graph query languages.
- *Prevalence of RDBMSes:* Relational databases still play an important role in managing and processing graphs.

Our survey also highlights other interesting facts, such as the prevalence of machine learning on graph data, e.g., for clustering vertices, predicting links, and finding influential vertices.

We further reviewed user feedback in the mailing lists, bug reports, and feature requests in the source code repositories of 22 software products between January and September of 2017 with two goals: (i) to answer several new questions that the participants' responses raised; and (ii) to identify more specific challenges in different classes of graph technologies than the ones we could iden-

1 Introduction

Graph data representing connected entities and their relationships appear in many application domains, most naturally in social networks, the Web, the Semantic Web, road maps, communication networks, biology, and finance, just to name a few examples. There has been a noticeable increase in the prevalence of work on graph processing both in research and in practice, evidenced by the surge in the number of different commercial and research software for managing and processing graphs. Examples include graph database systems [13,20,26,49,65,73,90], RDF engines [52,96], linear algebra software [17,63], visualization software [25,29], query languages [41,72,78], and distributed graph processing systems [30,34,40]. In the academic literature, a large number of publications that study numerous topics related to graph processing regularly appear across a wide spectrum of research venues.

Despite their prevalence, there is little research on how graph data are actually used in practice and the major challenges facing users of graph data, both in industry and in research. In April 2017, we conducted an online survey across 89 users of 22 different software products, with the goal of answering 4 high-level questions:

(i) What types of graph data do users have?
(ii) What computations do users run on their graphs?
(iii) Which software do users use to perform their computations?

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00778-019-00548-x) contains supplementary material, which is available to authorized users.

Siddhartha Sahu
s3sahu@uwaterloo.ca

Amine Mhedhbi
amine.mhedhbi@uwaterloo.ca

Semih Salihoglu
semih.salihoglu@uwaterloo.ca

Jimmy Lin
jimmylin@uwaterloo.ca

M. Tamer Özsu
tamer.ozsu@uwaterloo.ca

1 University of Waterloo, Waterloo, Canada

Published online: 29 June 2019

Springer

Graph Usage Study

[Sahu et al., 2017, 2020]

The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing

Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, M. Tamer Özsu
David R. Cheriton School of Computer Science
University of Waterloo
{s3sahu,amine.mhedhbi,semih.salihoglu,jimmylin,tamer.ozsu}@uwaterloo.ca

The VLDB Journal
https://doi.org/10.1007/s00778-019-00548-x
SPECIAL ISSUE PAPER

The ubiquity of large graphs and surprising challenges of graph processing: extended survey

Siddhartha Sahu[1] · Amine Mhedhbi[1] · Semih Salihoglu[1] · Jimmy Lin[1] · M. Tamer Özsu[1]

Objectives

1. What kind of graph data, computations, software, and major challenges real users have in practice?
2. What types of graph data, computations, software, and major challenges researchers target in publications?

Major Findings

1. Graphs are everywhere!
2. Graphs are very large!
3. ML on graphs is very popular ($>$ 85% of respondents have ML workloads)!
4. Scalability is the most pressing challenge (followed by visualization & query languages)!
5. Relational DBMSs still play an important role!

One particular type – streaming graphs

Streaming aspects

- Unbounded data ⇒ non-blocking algorithms & operators (one-pass)
- Usually at high speed ⇒ real-time constraints

One particular type – streaming graphs

Streaming aspects

- Unbounded data $\Rightarrow$ non-blocking algorithms & operators (one-pass)
- Usually at high speed $\Rightarrow$ real-time constraints

Graph aspects

- (Typically) edges streaming
- Graph "emerges"

One particular type – streaming graphs

Streaming aspects

- Unbounded data ⇒ non-blocking algorithms & operators (one-pass)
- Usually at high speed ⇒ real-time constraints

Graph aspects

- (Typically) edges streaming
- Graph "emerges"

Use case

Alibaba

- 500M active users, 2B catalog items
- 320K transactions/second (at peak)
- Need to process PB data in real-time in hours

Streaming Data Processing

Streaming Graph Processing

S-graffito Project

Concluding Remarks

Streaming Data Processing

Stream Systems

Inputs

One or more sources generate data continuously, in real time, and in fixed order (by timestamp)

- Sensor networks – weather monitoring, road traffic monitoring
- Web data – financial trading, news/sports tickers
- Scientific data – experiments in particle physics
- Transaction logs – point-of-sale purchases
- Network traffic analysis – IP packet headers

Stream Systems

Inputs

One or more sources generate data continuously, in real time, and in fixed order (by timestamp)

- Sensor networks – weather monitoring, road traffic monitoring
- Web data – financial trading, news/sports tickers
- Scientific data – experiments in particle physics
- Transaction logs – point-of-sale purchases
- Network traffic analysis – IP packet headers

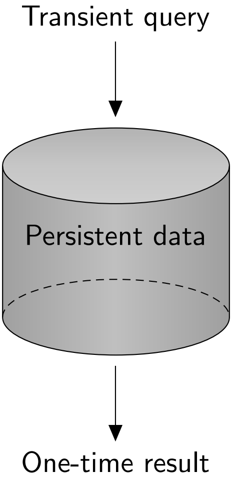
Outputs

Want to collect and process data in real-time; up-to-date answers generated continuously or periodically

- Environment monitoring
- Location monitoring
- Correlations across stock prices
- Denial-of-service attack detection

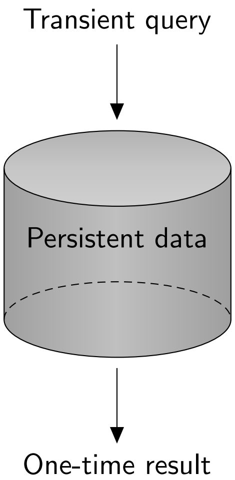
DBMS versus DSS

Traditional DBMS:

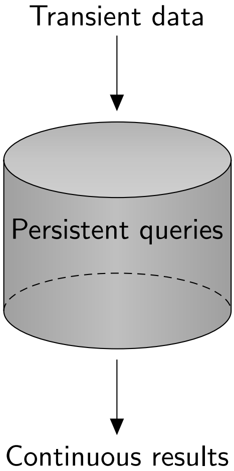


DBMS versus DSS

Traditional DBMS:



Data Stream System (DSS):



DBMS versus DSS

Traditional DBMS:

Transient query

One-time result

Data Stream System (DSS):

Transient data

Continuous results

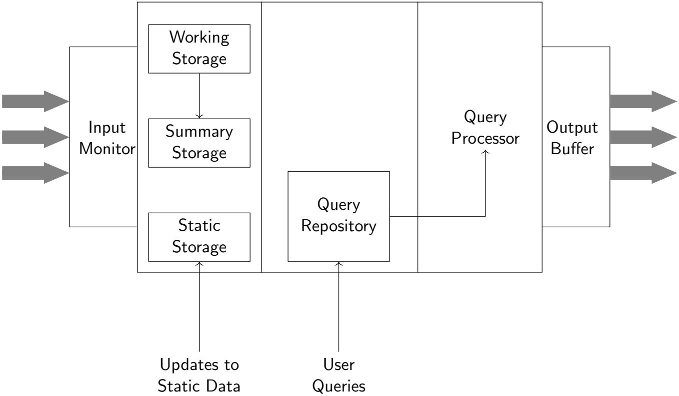
Other differences of DSS

- Push-based (data-driven)
- Persistent queries
- Unbounded stream; query execution as data arrives at the system – one look
- System conditions may not be stable – arrival rates fluctuate, workload may change

Old vs New

- Older systems: Data Stream Management Systems (DSMS) [Golab and Özsu, 2010]
  - Provide the functionalities of a typical DBMS
  - Examples: STREAM, Gigascope, TelegraphCQ, Aurora, Borealis
  - Mostly single machine systems
  - From early 2000s to late 2000s
- Newer systems: Data Stream Processing Systems (DSPS)
  - May not have full DBMS functionality
  - Examples: Apache Storm, Heron, Spark Streaming, Flink, MillWheel, TimeStream
  - Almost all are scale-out
  - From mid-2010s

DSMS System Architecture



Stream Data Model

Append-only sequence of timestamped items that arrive in some order.

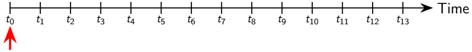
$$\langle \text{timestamp, payload} \rangle$$

What is the payload?

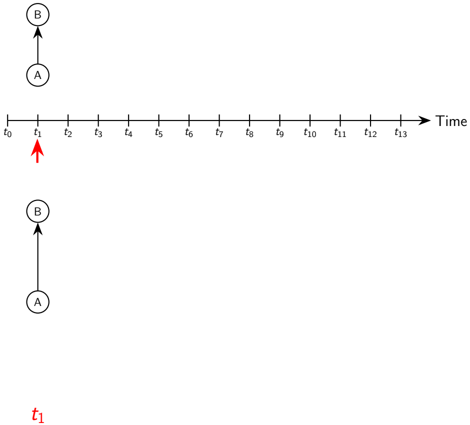
- Relational tuple
- Revision tuple
- Graph edge
- Sequence of events (as in publish/subscribe systems)
- Sequence of sets (or bags) of elements with each set storing elements that have arrived during the same unit of time

Streaming Graph Processing

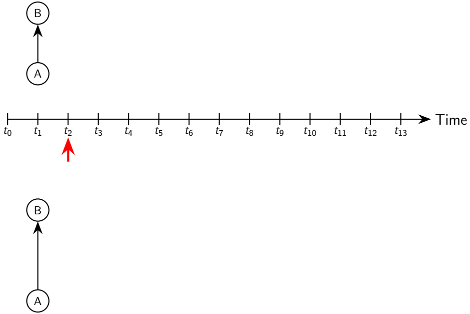
Streaming Graphs



Streaming Graphs

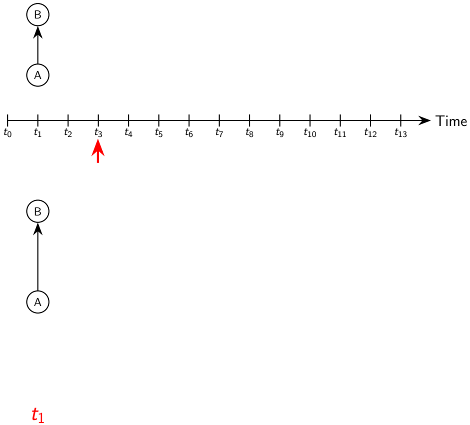


Streaming Graphs

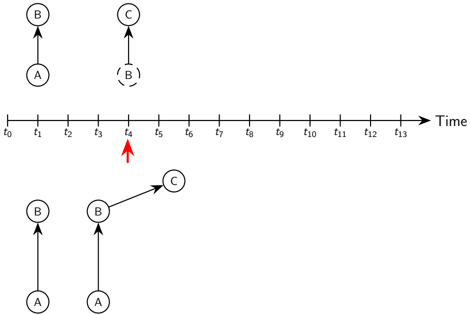


t_1

Streaming Graphs

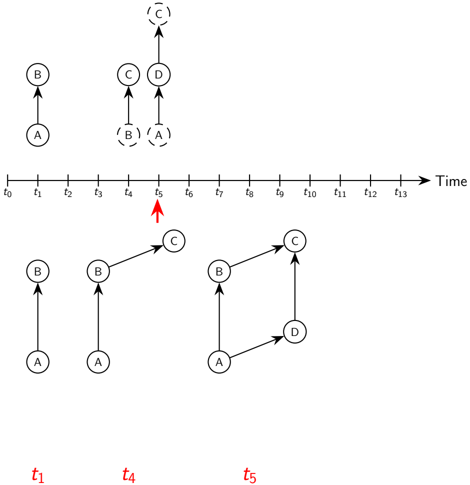


Streaming Graphs

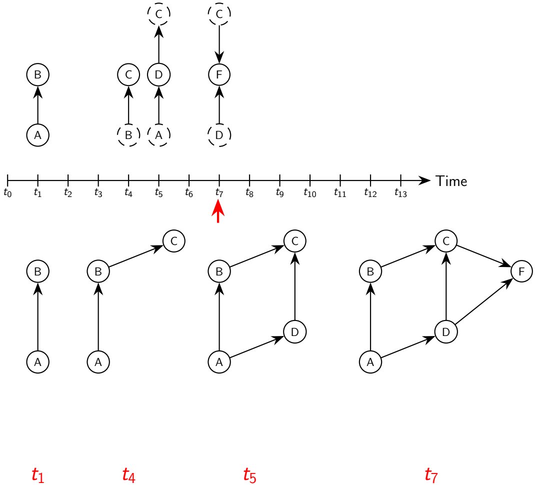


t_1 t_4

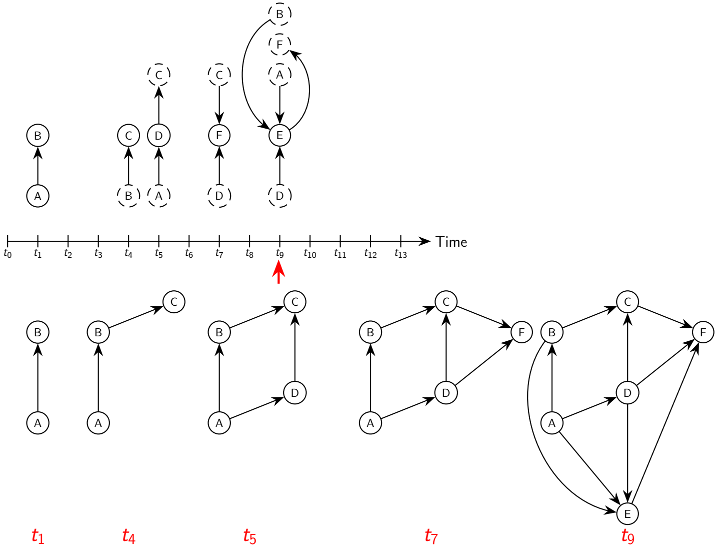
Streaming Graphs



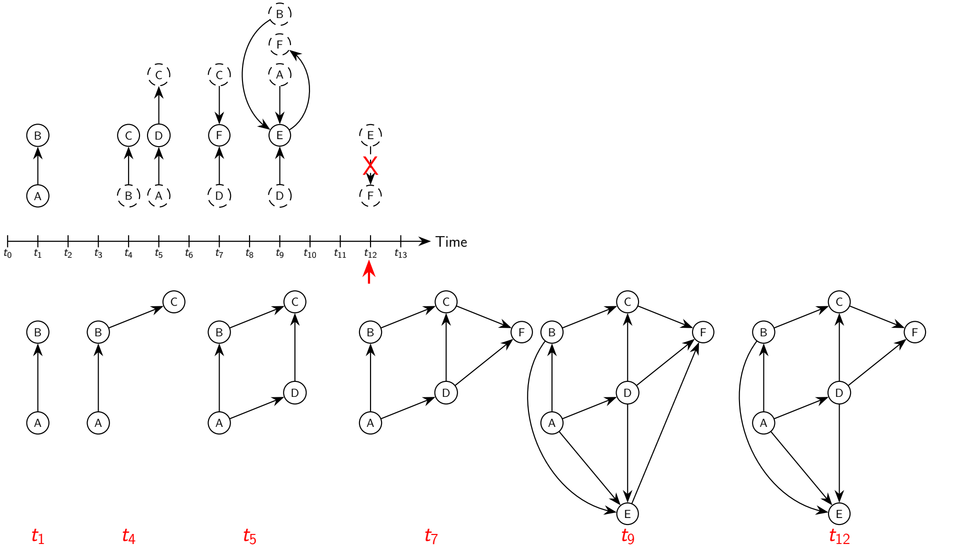
Streaming Graphs



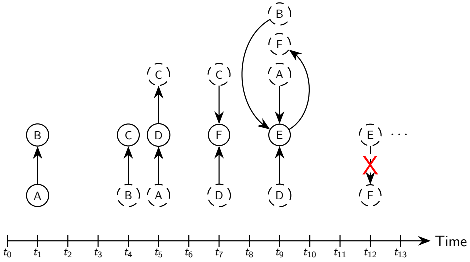
Streaming Graphs



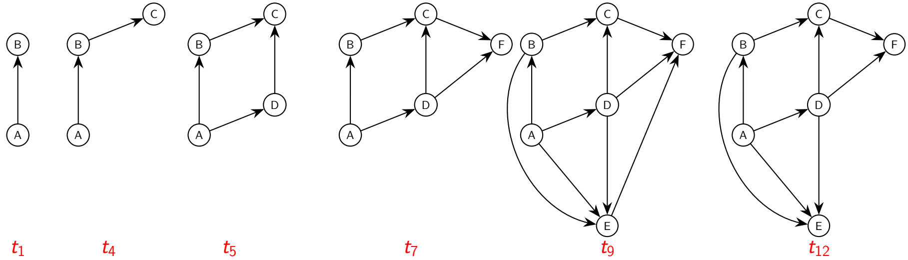
Streaming Graphs



Streaming Graphs



- Combines two difficult problems: streaming+graphs
- Unbounded $\Rightarrow$ don't see entire graph
- Streaming rates can be very high

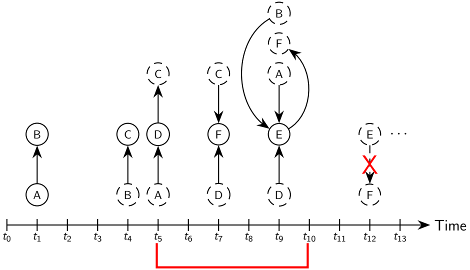


Streaming Graph Computation Models

- Continuous
  - Process each edge as it comes $\Rightarrow$ for simple transactional operations
  - Requires linear space $\Rightarrow$ unrealistic
    - Many graph problems are not solvable [McGregor, 2014]
  - Semi-streaming model $\Rightarrow$ sublinear space [Feigenbaum et al., 2005]
    - Sufficient to store vertices but not edges (typically $|V| \ll |E|$) $\Rightarrow$ dynamic graph model
    - Approximation for many graph algorithms exist

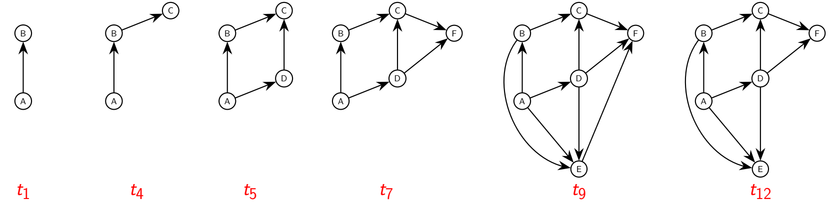
Streaming Graph Computation Models

- Continuous
  - Process each edge as it comes ⇒ for simple transactional operations
  - Requires linear space ⇒ unrealistic
    - Many graph problems are not solvable [McGregor, 2014]
  - Semi-streaming model ⇒ sublinear space [Feigenbaum et al., 2005]
    - Sufficient to store vertices but not edges (typically $|V| \ll |E|$) ⇒ dynamic graph model
    - Approximation for many graph algorithms exist
- Windowed
  - Use windows to batch edges
  - For more complex queries



Continuous Computation

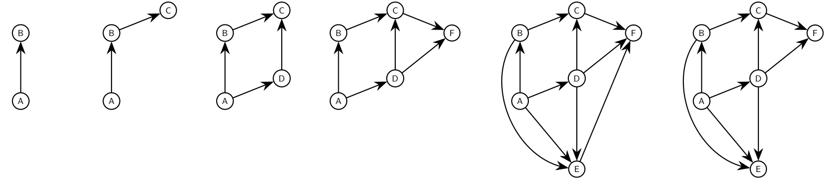
Query: Vertices reachable from vertex A



Time	Incoming edge	Results
t_1	⟨A,B⟩	{B}
t_2		
t_3		
t_4	⟨B,C⟩	{B,C}
t_5	⟨A,D⟩, ⟨D,C⟩	{B,C,D}
t_6		
t_7	⟨C,F⟩, ⟨D,F⟩	{B,C,D,F}
t_8		
t_9	⟨D,E⟩, ⟨A,E⟩, ⟨B,E⟩, ⟨E,F⟩	{B,C,D,F,E}
t_{10}		

Windowed Computation

Query: Vertices reachable from vertex A



t_1 t_4 t_5 t_7 t_9 t_{12}

(Window size=5)

Time	Incoming edge	Expired edges	Results
t_1	⟨A,B⟩		{B}
t_2			
t_3			
t_4	⟨B,C⟩		{B,C}
t_5	⟨A,D⟩, ⟨D,C⟩		{B,C,D}
t_6		⟨A,B⟩	{~~B~~,C,D}
t_7	⟨C,F⟩, ⟨D,F⟩		{C,D,F}
t_8			
t_9	⟨D,E⟩, ⟨A,E⟩, ⟨B,E⟩, ⟨E,F⟩	⟨B,C⟩	{C,D,F,E}
t_{10}		⟨A,D⟩, ⟨D,C⟩	{~~C,D~~,F,E}

Querying Graph Streams

- Graph query functionalities
  - Subgraph matching queries & reachability (path) queries
  - Doing these in the streaming context
  - This is querying beyond simple transactional operations on an incoming edge
    - Edge represents a user purchasing an item $\rightarrow$ do some operation
    - Edge represents events in news $\rightarrow$ send an alert
- Subgraph pattern matching under stream of updates
  - Windowed join processing
  - Graphflow [Kankanamge et al., 2017], TurboFlux [Kim et al., 2018]
  - These are not designed to deal with unboundedness of the data graph
- Path queries under stream of updates

Analytics on Graph Streams

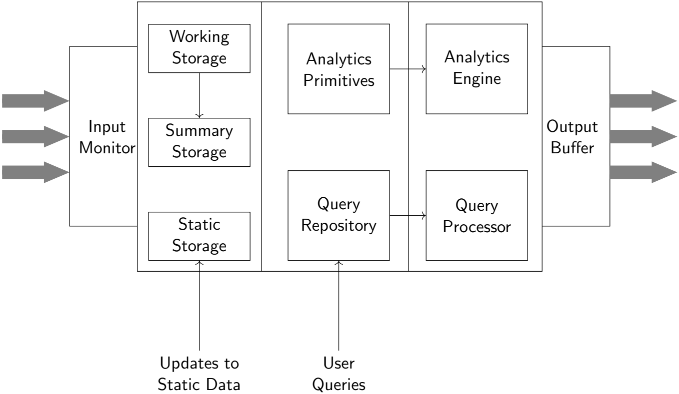
- Many use cases
  - Recommender systems
  - Fraud detection [Qiu et al., 2018]
  - ...
- Existing relevant work
  - Snapshot-based systems
    - Aspen [Dhulipala et al., 2019], STINGER [Ediger et al., 2012]
    - Consistent graph views across updates
  - Snapshot + Incremental Computations
    - Kineograph [Cheng et al., 2012], GraPu [Sheng et al., 2018], GraphIn [Sengupta et al., 2016], GraphBolt [Mariappan and Vora, 2019]
    - Identify and re-process subgraphs that are effected by updates
  - Designed to handle high velocity updates
  - Cannot handle unbounded streams
    - Similar to dynamic graph processing solutions

S-graffito Project

https://dsg-uwaterloo.github.io/s-graffito/

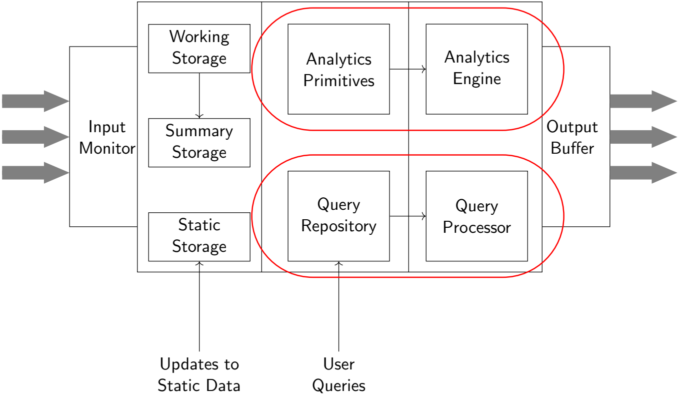
S-Graffito project

Processing of transactional (OLTP) and and analytical (OLAP) queries on high streaming rate, very large graphs.

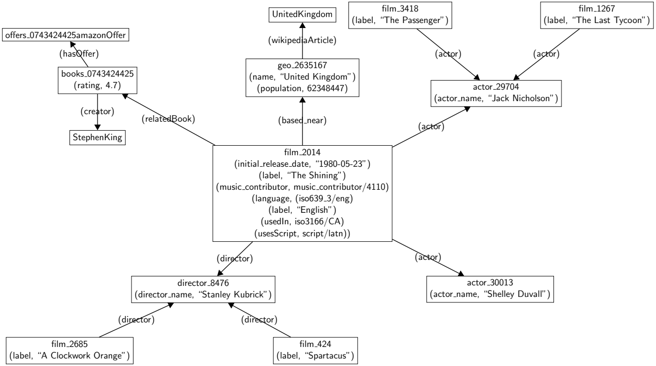


S-Graffito project

Processing of transactional (OLTP) and and analytical (OLAP) queries on high streaming rate, very large graphs.



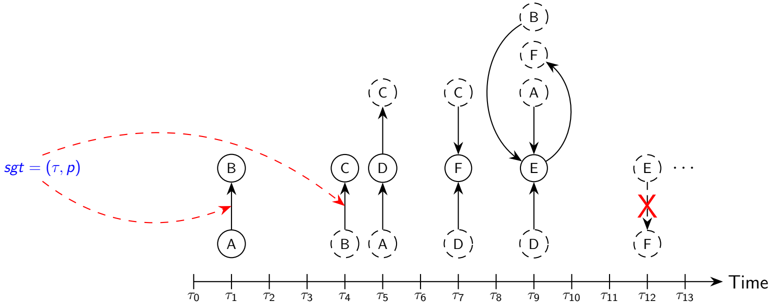
Working on Property Graphs



Property Graph

A *property graph* is an attributed graph $G = (V, E, \Sigma, \psi, \phi, \mathcal{K}, \mathcal{P})$ where V is a set of vertices, E is a set of edges, $\psi : E \to V \times V$ is a function that maps each edge to an ordered pair of vertices, Σ is a set of labels and ϕ is a labelling function, $\phi : (V \cup E) \to \Sigma$, $\mathcal{K}$ is a set of property keys, $\mathcal{P}$ is a set of values, and $\nu : (V \cup E) \times \mathcal{K} \to \mathcal{P}$ is a partial function assigning values for properties to objects.

Arrivals are Streaming Graph Tuples



Streaming Graph Tuple

A *streaming graph tuple* (sgt) is a streaming tuple where is a pair (τ, p) where τ is the event (application) timestamp of the tuple assigned by the data source, p defines the payload of the tuple that indicates an edge $e \in E$ or a vertex $v \in V$ of the property graph G, and an operation $op \in \{insert, delete, update\}$ that defines the type of the tuple.

Time-based Window & Snapshot

τ	p
τ_1	$\langle$(A,B), *insert*$\rangle$
τ_4	$\langle$(B,C), *insert*$\rangle$
τ_5	$\langle$(A,D), *insert*$\rangle$
τ_5	$\langle$(D,C), *insert*$\rangle$
τ_7	$\langle$(C,F), *insert*$\rangle$
τ_7	$\langle$(D,F), *insert*$\rangle$
τ_9	$\langle$(B,E), *insert*$\rangle$
τ_9	$\langle$(E,E), *insert*$\rangle$
τ_9	$\langle$(E,F), *insert*$\rangle$
τ_{12}	$\langle$(E,F), *delete*$\rangle$

$W(\tau_5 - \tau_{10}]$

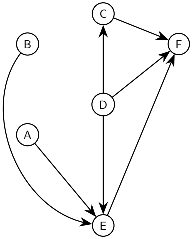
Time-based Window

A *time-based window* W over a streaming graph S is a time interval $(W^b, W^e]$ where W^b and W^e are the beginning and end times of window W and $W_e - W_b = |W|$. The window contents $W(c)$ is the multiset of sgts where the timestamp τ_i of each record t_i is in the window interval, i.e., $W(c) = \{t_i \mid W_b < \tau_i \leq W_e\}$. When it is clear from context, W is used interchangeably to refer to window interval or its contents.

Time-based Window & Snapshot

τ	p
τ_1	$\langle$(A,B), *insert*$\rangle$
τ_4	$\langle$(B,C), *insert*$\rangle$
τ_5	$\langle$(A,D), *insert*$\rangle$
τ_5	$\langle$(D,C), *insert*$\rangle$
τ_7	$\langle$(C,F), *insert*$\rangle$
τ_7	$\langle$(D,F), *insert*$\rangle$
τ_9	$\langle$(B,E), *insert*$\rangle$
τ_9	$\langle$(E,E), *insert*$\rangle$
τ_9	$\langle$(E,F), *insert*$\rangle$
τ_{12}	$\langle$(E,F), *delete*$\rangle$

$W(\tau_5 - \tau_{10}]$



Time-based Window

A *time-based window* W over a streaming graph S is a time interval $(W^b, W^e]$ where W^b and W^e are the beginning and end times of window W and $W_e - W_b = |W|$. The window contents $W(c)$ is the multiset of sgts where the timestamp τ_i of each record t_i is in the window interval, i.e., $W(c) = \{t_i \mid W_b < \tau_i \leq W_e\}$. When it is clear from context, W is used interchangeably to refer to window interval or its contents.

Streaming Graph Snapshot

A *streaming graph snapshot* $G_{W,\tau}$ is the graph formed by the tuples in the time-based window W at time τ.

S-graffito Project

Streaming Graph Querying

Anil Pacaci

Streaming Graph Querying Objectives

Persistent query processing over streaming graphs

1. Path navigation queries
   - Non-blocking operators for path queries
   - Regular path queries (RPQ)
     - Regular expressions that are matched against directed, labelled paths
2. A query subsystem for persistent graph queries over streaming graphs
   - Combining graph patterns & path navigation
   - Treating paths as first-class citizens
3. Querying streaming graphs with data
   - Attribute-based predicates for property graphs

Persistent RPQ Evaluation

[Pacaci et al., 2020]

- Design space for persistent RPQ algorithms

	Result semantics	
Path semantics	Simple Append-only	Simple Explicit delete
	Arbitrary Append-only	Arbitrary Explicit delete

Persistent RPQ Evaluation

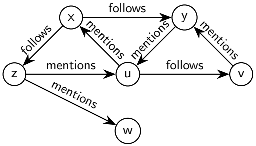
[Pacaci et al., 2020]

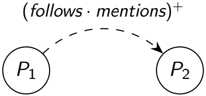
- Design space for persistent RPQ algorithms

	Result semantics	
Path semantics	Simple Append-only	Simple Explicit delete
	Arbitrary Append-only	Arbitrary Explicit delete

- Path semantics used in practice
  - Simple paths (*no repeating vertex*): navigation on road networks

$Q_1 = (\mathit{follows} \cdot \mathit{mentions})^+$





Persistent RPQ Evaluation

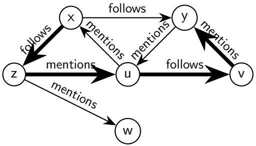
[Pacaci et al., 2020]

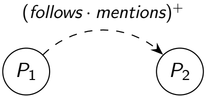
- Design space for persistent RPQ algorithms

Path semantics \ Result semantics		
	Simple Append-only	Simple Explicit delete
	Arbitrary Append-only	Arbitrary Explicit delete

- Path semantics used in practice
  - Simple paths (*no repeating vertex*): navigation on road networks

$Q_1 = (\mathit{follows} \cdot \mathit{mentions})^+$





Persistent RPQ Evaluation

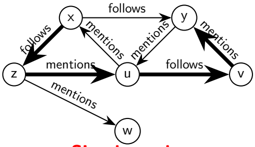
[Pacaci et al., 2020]

- Design space for persistent RPQ algorithms

Path semantics \ Result semantics		
	Simple Append-only	Simple Explicit delete
	Arbitrary Append-only	Arbitrary Explicit delete

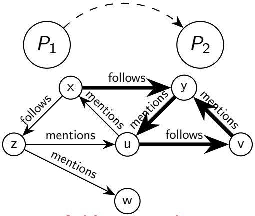
- Path semantics used in practice
  - Simple paths (*no repeating vertex*): navigation on road networks
  - Arbitrary paths: reachability on communication networks

$Q_1 = (\textit{follows} \cdot \textit{mentions})^+$



Simple paths

$(\textit{follows} \cdot \textit{mentions})^+$



Arbitrary paths

Persistent RPQ Evaluation

[Pacaci et al., 2020]

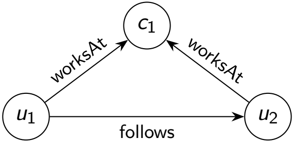
- Design space for persistent RPQ algorithms

Path semantics \ Result semantics		
	Simple Append-only	Simple Explicit delete
	Arbitrary Append-only	Arbitrary Explicit delete

- Path semantics used in practice
  - Simple paths (*no repeating vertex*): navigation on road networks
  - Arbitrary paths: reachability on communication networks
- Result semantics & stream types
  - Append-only streams with fast insertions
  - Support for explicit deletions

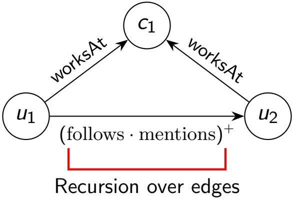
Beyond Path Navigation

Combining subgraph matching & path navigation



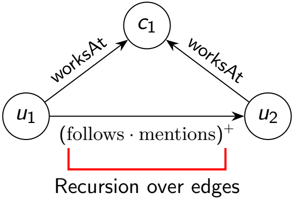
Beyond Path Navigation

Combining subgraph matching & path navigation



Beyond Path Navigation

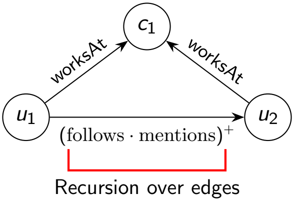
Combining subgraph matching & path navigation



- Unions of Conjunctive RPQs (UCRPQ)
  - SPARQL v1.1, Cypher9 (limited form), Oracle PGQL [van Rest et al., 2016]

Beyond Path Navigation

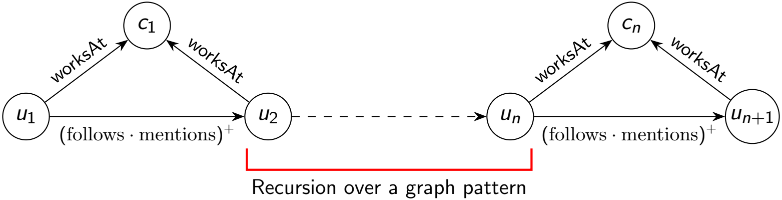
Combining subgraph matching & path navigation



- Unions of Conjunctive RPQs (UCRPQ)
  - SPARQL v1.1, Cypher9 (limited form), Oracle PGQL [van Rest et al., 2016]
- No algebraic closure

Beyond Path Navigation

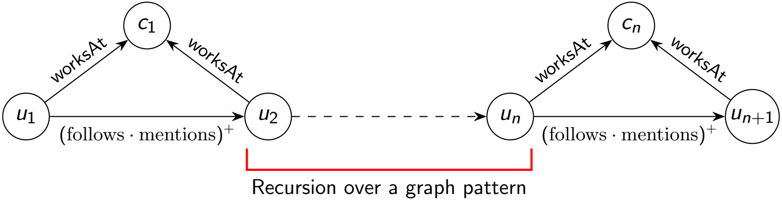
Combining subgraph matching & path navigation



- Unions of Conjunctive RPQs (UCRPQ)
  - SPARQL v1.1, Cypher9 (limited form), Oracle PGQL [van Rest et al., 2016]
- No algebraic closure

Beyond Path Navigation

Combining subgraph matching & path navigation

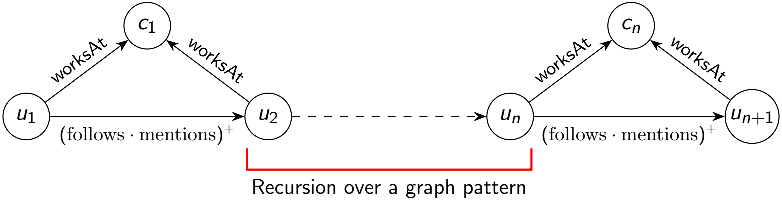


- Unions of Conjunctive RPQs (UCRPQ)
  - SPARQL v1.1, Cypher9 (limited form), Oracle PGQL [van Rest et al., 2016]
- No algebraic closure

- Regular Queries (RQ) [Reutter et al., 2017]
  - A subset of Datalog with algebraic closure
  - Computationally well-behaved
- The basis of G-CORE [Angles et al., 2018]

Beyond Path Navigation

Combining subgraph matching & path navigation



- Unions of Conjunctive RPQs
- No algebraic closure

- Regular Queries (RQ)
- The basis of G-CORE [Angles et al., 2018]

Our work

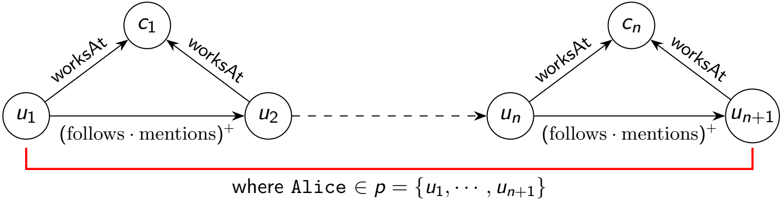
- An algebra for RQ on streaming graphs
- Concrete implementation of this algebra

Paths as First-class Citizens

So far we focused on *existence* of a path, i.e., reachability

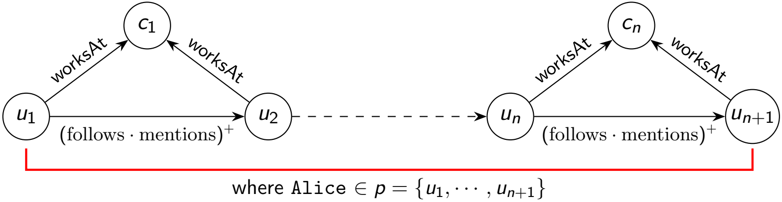
Paths as First-class Citizens

So far we focused on *existence* of a path, i.e., reachability



Paths as First-class Citizens

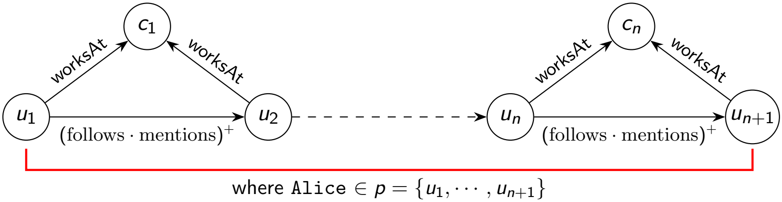
So far we focused on *existence* of a path, i.e., reachability



- Ability to store, return and compare paths

Paths as First-class Citizens

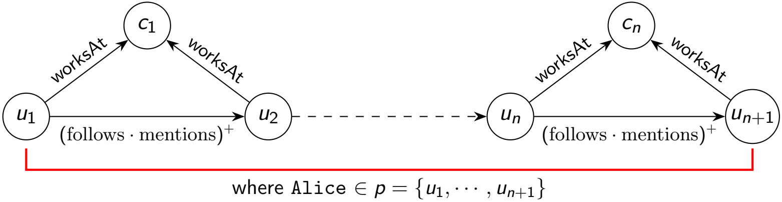
So far we focused on *existence* of a path, i.e., reachability



- Ability to store, return and compare paths
- Enumerate all paths
  - High complexity, FPT for certain classes [Martens and Trautner, 2019]

Paths as First-class Citizens

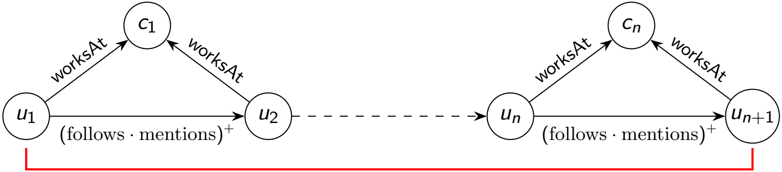
So far we focused on *existence* of a path, i.e., reachability



- Ability to store, return and compare paths
- Enumerate all paths
  - High complexity, FPT for certain classes [Martens and Trautner, 2019]
- Structural restrictions on path operations
  - Length predicates [Barceló et al., 2012]
  - Closed semi-ring aggregates [Cruz and Norvell, 1989]

Paths as First-class Citizens

So far we focused on *existence* of a path, i.e., reachability



Our work

- Data model and query language that treats paths as *first-class citizens*
- Streaming, sliding-window algorithms for common path operations

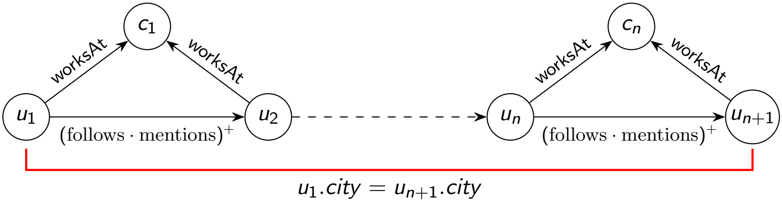
- Structural restrictions on path operations
  - Length predicates [Barceló et al., 2012]
  - Closed semi-ring aggregates [Cruz and Norvell, 1989]

Querying Graphs with Data

Real-world graphs have data, so as queries

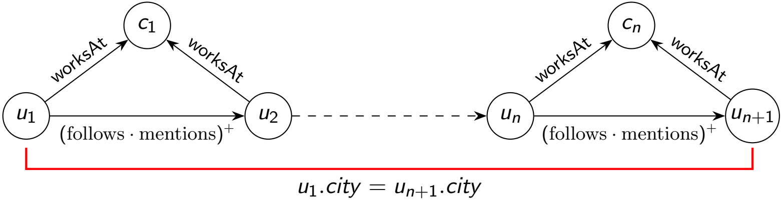
Querying Graphs with Data

Real-world graphs have data, so as queries



Querying Graphs with Data

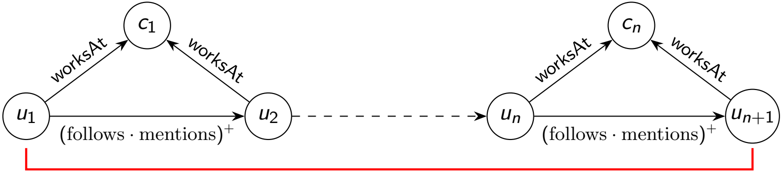
Real-world graphs have data, so as queries



- Support for attribute-based predicates on property graphs
- Regular Property Graph Queries (RPGQ) [Bonifati et al., 2018]
  - RQ on property graphs
- Non-trivial query planning [Mulder et al., 2020]
  - Structure-based vs structure&attribute-based planning
  - Up to 30× performance differences

Querying Graphs with Data

Real-world graphs have data, so as queries



Our work

- Support for property graphs & attribute-based predicates
- Non-blocking implementation of RPGQ for streaming graphs

- Non-trivial query planning [Mulder et al., 2020]
  - Structure-based vs structure&attribute-based planning
  - Up to 30× performance differences

S-graffito Project

Streaming Graph Analytics

Aida Sheshbolouki

Streaming Graph Analytics Objectives

Building a generic analytics engine based on window semantics and vertex embeddings

1. Exploratory analysis of real-world streaming graphs
2. Representation learning over streaming graphs
3. Prediction-based analytics over streaming graphs

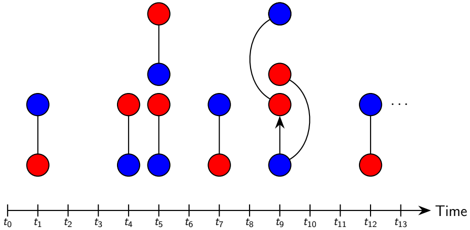
Exploratory Analysis of Real-world Streaming Graphs

Exploratory Analysis of Real-world Streaming Graphs

1. Identifying streaming graph patterns

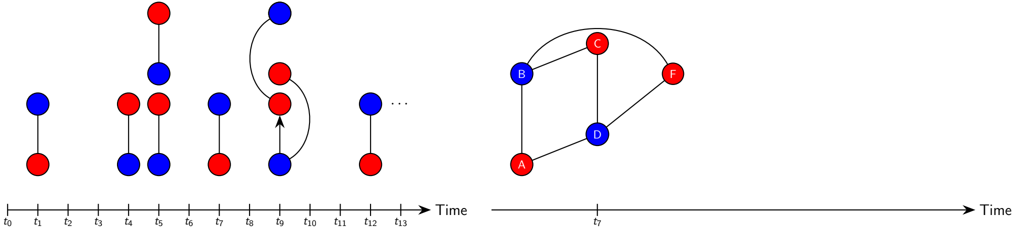
Exploratory Analysis of Real-world Streaming Graphs

1. Identifying streaming graph patterns
   - The emergence patterns of edges ⇒ attachment rules
     - "Rich-get-richer" conjecture



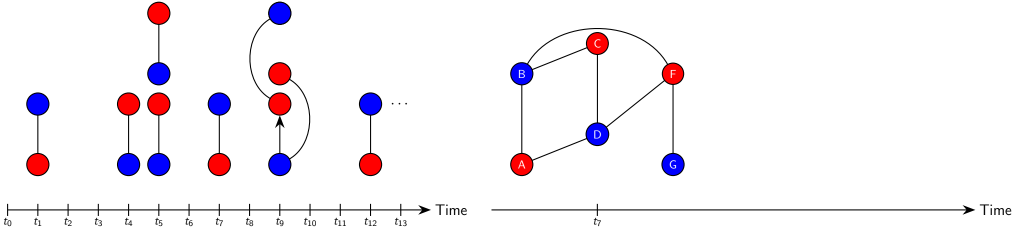
Exploratory Analysis of Real-world Streaming Graphs

1. Identifying streaming graph patterns
   - The emergence patterns of edges ⇒ attachment rules



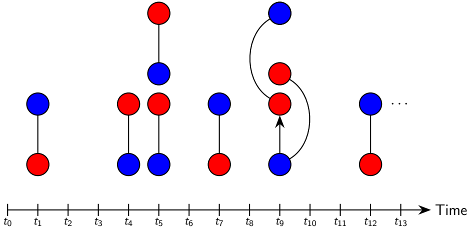
Exploratory Analysis of Real-world Streaming Graphs

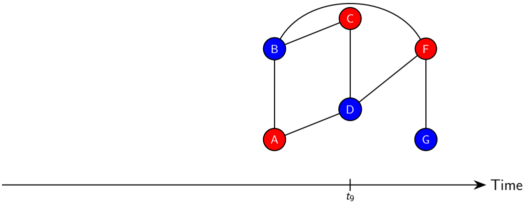
1. Identifying streaming graph patterns
   - The emergence patterns of edges ⇒ attachment rules



Exploratory Analysis of Real-world Streaming Graphs

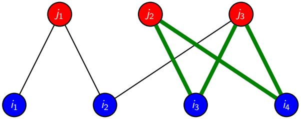
1. Identifying streaming graph patterns
   - The emergence patterns of edges $\Rightarrow$ attachment rules





Exploratory Analysis of Real-world Streaming Graphs

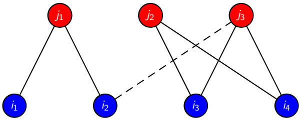
1. Identifying streaming graph patterns
   - The emergence patterns of edges ⇒ attachment rules
   - The emergence patterns of key subgraphs ⇒ subgraph densification power laws
     - The number of 2,2-bicliques (butterflies) follows a power law function of the number of the number of edges
     - Bursty butterfly densification – Butterflies emerge in a bursty fashion due to the existing hubs contribution
     - sGrapp: Streaming Graph Approximation Framework for Butterfly Counting



Exploratory Analysis of Real-world Streaming Graphs

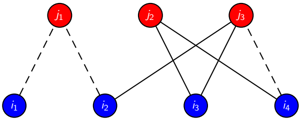
1. Identifying streaming graph patterns
    - The emergence patterns of edges ⇒ attachment rules
    - The emergence patterns of key subgraphs ⇒ subgraph densification power laws
    - The connectivity and robustness of the graph snapshots

Merging components



Robust against random edge removals
Not robust against targeted removals

A giant growing component



Robust against any edge removal

Exploratory Analysis of Real-world Streaming Graphs

1. Identifying streaming graph patterns
    - The emergence patterns of edges $\Rightarrow$ attachment rules
    - The emergence patterns of key subgraphs $\Rightarrow$ subgraph densification power laws
    - The connectivity and robustness of the graph snapshots
2. Modeling streaming graphs
    - Synthetic graph model that preserves realistic patterns
    - For pinpointing the performance of processing algorithms

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

1. Unbounded stream management and processing

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

1. Unbounded stream management and processing
2. Addressing structural evolutions

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

1. Unbounded stream management and processing
2. Addressing structural evolutions
3. Addressing streaming property graphs

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

1. Unbounded stream management and processing
2. Addressing structural evolutions
3. Addressing streaming property graphs
4. Addressing data sparsity

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

1. Unbounded stream management and processing
2. Addressing structural evolutions
3. Addressing streaming property graphs
4. Addressing data sparsity
5. Model optimizations
    - Heterogeneous embedding
    - Dynamic graph convolutions
    - Parameter training

Representation Learning over Streaming Graphs

Main issue: trade-off between effectiveness and efficiency

1. Unbounded stream management and processing
2. Addressing structural evolutions
3. Addressing streaming property graphs
4. Addressing data sparsity
5. Model optimizations
    - Heterogeneous embedding
    - Dynamic graph convolutions
    - Parameter training

Outcome

An embedding model based on window semantics to incrementally learn the graph evolutions and update the vertex embeddings.

Prediction-based Analytics over Streaming Graphs

1. Efficient windowed analytics
2. Window semantics
3. Graph versatility
4. Accurate predictions

Concluding Remarks

Some Take-home Messages

- Streaming graphs are real and occur in real-life applications
- We have not paid nearly sufficient attention to streaming graph challenges
- Streaming $\neq$ dynamic
  ... most "streaming" papers are not streaming
- Unboundedness in streams raises real challenges
- Most graph problems are unbounded under edge insert/delete
- The entire field is pretty much open...
  ... this area is tough and you are not likely to write as many papers

Thank you!

Aida
Sheshbolouki

Anil
Pacaci

Angela
Bonifati

NSERC CRSNG INNOVATION.CA Ontario HUAWEI Google BOSCH IBM Canada Lab CAS

References I

Angles, R., Arenas, M., Barceló, P., Boncz, P. A., Fletcher, G. H. L., Gutierrez, C., Lindaaker, T., Paradies, M., Plantikow, S., Sequeda, J. F., van Rest, O., and Voigt, H. (2018). G-CORE: A core for future graph query languages. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 1421–1432. Available from: http://doi.acm.org/10.1145/3183713.3190654.

Barceló, P., Libkin, L., Lin, A. W., and Wood, P. T. (2012). Expressive languages for path queries over graph-structured data. *ACM Trans. Database Syst.*, 37(4). Available from: https://doi.org/10.1145/2389241.2389250.

Bonifati, A., Fletcher, G., Voigt, H., and Yakovets, N. (2018). *Querying Graphs*. Synthesis Lectures on Data Management. Morgan & Claypool. Available from: https://doi.org/10.2200/S00873ED1V01Y201808DTM051.

Cheng, R., Hong, J., Kyrola, A., Miao, Y., Weng, X., Wu, M., Yang, F., Zhou, L., Zhao, F., and Chen, E. (2012). Kineograph: Taking the pulse of a fast-changing and connected world. In *Proc. 7th ACM SIGOPS/EuroSys European Conf. on Comp. Syst.*, pages 85–98. Available from: http://doi.acm.org/10.1145/2168836.2168846.

References II

Cruz, I. F. and Norvell, T. S. (1989). Aggregative closure: An extension of transitive closure. In *Proc. 5th Int. Conf. on Data Engineering*, pages 384–391. Available from: https://doi.org/10.1109/ICDE.1989.47239.

Dhulipala, L., Blelloch, G. E., and Shun, J. (2019). Low-latency graph streaming using compressed purely-functional trees. In *Proc. ACM SIGPLAN 2019 Conf. on Programming Language Design and Implementation*, pages 918–934. Available from: http://doi.acm.org/10.1145/3314221.3314598.

Ediger, D., McColl, R., Riedy, J., and Bader, D. A. (2012). STINGER: High performance data structure for streaming graphs. In *Proc. 2012 IEEE Conf. on High Performance Extreme Comp.*, pages 1–5.

Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., and Zhang, J. (2005). On graph problems in a semi-streaming model. *Theor. Comp. Sci.*, 348(2):207–216. Available from: http://www.sciencedirect.com/science/article/pii/S0304397505005323.

Golab, L. and Özsu, M. T. (2010). *Data Stream Systems*. Synthesis Lectures on Data Management. Morgan & Claypool.

References III

Kankanamge, C., Sahu, S., Mhedbhi, A., Chen, J., and Salihoglu, S. (2017). Graphflow: An active graph database. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 1695–1698. Available from: http://doi.acm.org/10.1145/3035918.3056445.

Kempe, D., Kleinberg, J., and Tardos, E. (2003). Maximizing the spread of influence through a social network. In *Proc. 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 137—146. Available from: https://doi.org/10.1145/956750.956769.

Kim, K., Seo, I., Han, W., Lee, J., Hong, S., Chafi, H., Shin, H., and Jeong, G. (2018). Turboflux: A fast continuous subgraph matching system for streaming graph data. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 411–426.

Mariappan, M. and Vora, K. (2019). GraphBolt: Dependency-driven synchronous processing of streaming graphs. In *Proc. 14th ACM SIGOPS/EuroSys European Conf. on Comp. Syst.*, pages 25:1–25:16. Available from: http://doi.acm.org/10.1145/3302424.3303974.

Martens, W. and Trautner, T. (2019). Dichotomies for evaluating simple regular path queries. *ACM Trans. Database Syst.*, 44(4). Available from: https://doi.org/10.1145/3331446.

McGregor, A. (2014). Graph stream algorithms: A survey. *ACM SIGMOD Rec.*, 43(1):9–20. Available from: http://doi.acm.org/10.1145/2627692.2627694.

References IV

Mulder, T., Yakovets, N., and Fletcher, G. H. L. (2020). Towards planning of regular queries with memory. In *Proc. 23rd Int. Conf. on Extending Database Technology*, pages 451–454. Available from: https://doi.org/10.5441/002/edbt.2020.55.

Pacaci, A., Bonifati, A., and Özsu, M. T. (2020). Regular path query evaluation on streaming graphs. In *Proc. ACM SIGMOD Int. Conf. on Management of Data*, pages 1415–1430. Available from: https://doi.org/10.1145/3318464.3389733.

Qiu, X., Cen, W., Qian, Z., Peng, Y., Zhang, Y., Lin, X., and Zhou, J. (2018). Real-time constrained cycle detection in large dynamic graphs. *Proc. VLDB Endowment*, 11(12):1876–1888.

Reutter, J. L., Romero, M., and Vardi, M. Y. (2017). Regular queries on graph databases. *Theory of Computing Systems*, 61(1):31–83. Available from: https://doi.org/10.1007/s00224-016-9676-2.

Sahu, S., Mhedhbi, A., Salihoglu, S., Lin, J., and Özsu, M. T. (2017). The ubiquity of large graphs and surprising challenges of graph processing. *Proc. VLDB Endowment*, 11(4):420–431.

References V

Sahu, S., Mhedhbi, A., Salihoglu, S., Lin, J., and Özsu, M. T. (2020). The ubiquity of large graphs and surprising challenges of graph processing. *VLDB J.*, 29:595—618. Available from: https://link.springer.com/article/10.1007/s00778-019-00548-x.

Sengupta, D., Sundaram, N., Zhu, X., Willke, T. L., Young, J., Wolf, M., and Schwan, K. (2016). GraphIn: An online high performance incremental graph processing framework. In *Proc. 22nd Int. Euro-Par Conf.*, pages 319–333. Available from: http://dx.doi.org/10.1007/978-3-319-43659-3_24.

Sheng, F., Cao, Q., Cai, H., Yao, J., and Xie, C. (2018). Grapu: Accelerate streaming graph analysis through preprocessing buffered updates. In *Proc. 9th ACM Symp. on Cloud Computing*, pages 301–312. Available from: http://doi.acm.org/10.1145/3267809.3267811.

van Rest, O., Hong, S., Kim, J., Meng, X., and Chafi, H. (2016). PGQL: a property graph query language. In *Proc. 4th Int. Workshop on Graph Data Management Experiences & Systems*, page 7. Available from: https://doi.org/10.1145/2960414.2960421.