Adatbázisok elmélete – 2019. május 2.

Gráfadatbázisok

Szárnyas Gábor
szarnyas@mit.bme.hu

MŰEGYETEM 1782

Magyar Tudományos Akadémia

NoSQL rendszerek

ADATBÁZISKEZELŐ HASZNÁLHATÓSÁGA

- Kifejezőerő
  - Adatmodell
  - Lekérdezőnyelv
- Hatékonyság
  - Lekérdezésoptimalizáló
  - Lekérdezéskiértékelő
- Tooling
  - Eszközök (CLI/GUI/web UI)
  - Integráció (driverek)
  - Üzemeltetés (backup, hot swap)
- A dokumentación túl…
  - Terméktámogatás (support)
  - Közösség (community): levelezőlisták, Stack Overflow, konferenciák

RELÁCIÓS ADATMODELL

- Közel 50 éves terület
- Kiforrott elmélet
  - Több tízezer publikáció, tankönyvek
  - Az oktatás része
- Kiforrott eszközök
  - Kifinomult optimalizáció
  - Hatékony végrehajtómotorok
  - Jól üzemeltethetők


Information Retrieval

A Relational Model of Data for Large Shared Data Banks

E. F. Codd

IBM Research Laboratory, San Jose, California

E. F. Codd,
A Relational Model of Data for Large Shared Data Banks,
Communications of the ACM, 1970

NOSQL RENDSZEREK

Tipikusan:

- Nemrelációs adatmodell
- Cél a skálázhatóság
- Nyílt forráskód
- Kiforratlanabb implementációk, de aktív közösségek

4 fő kategória:

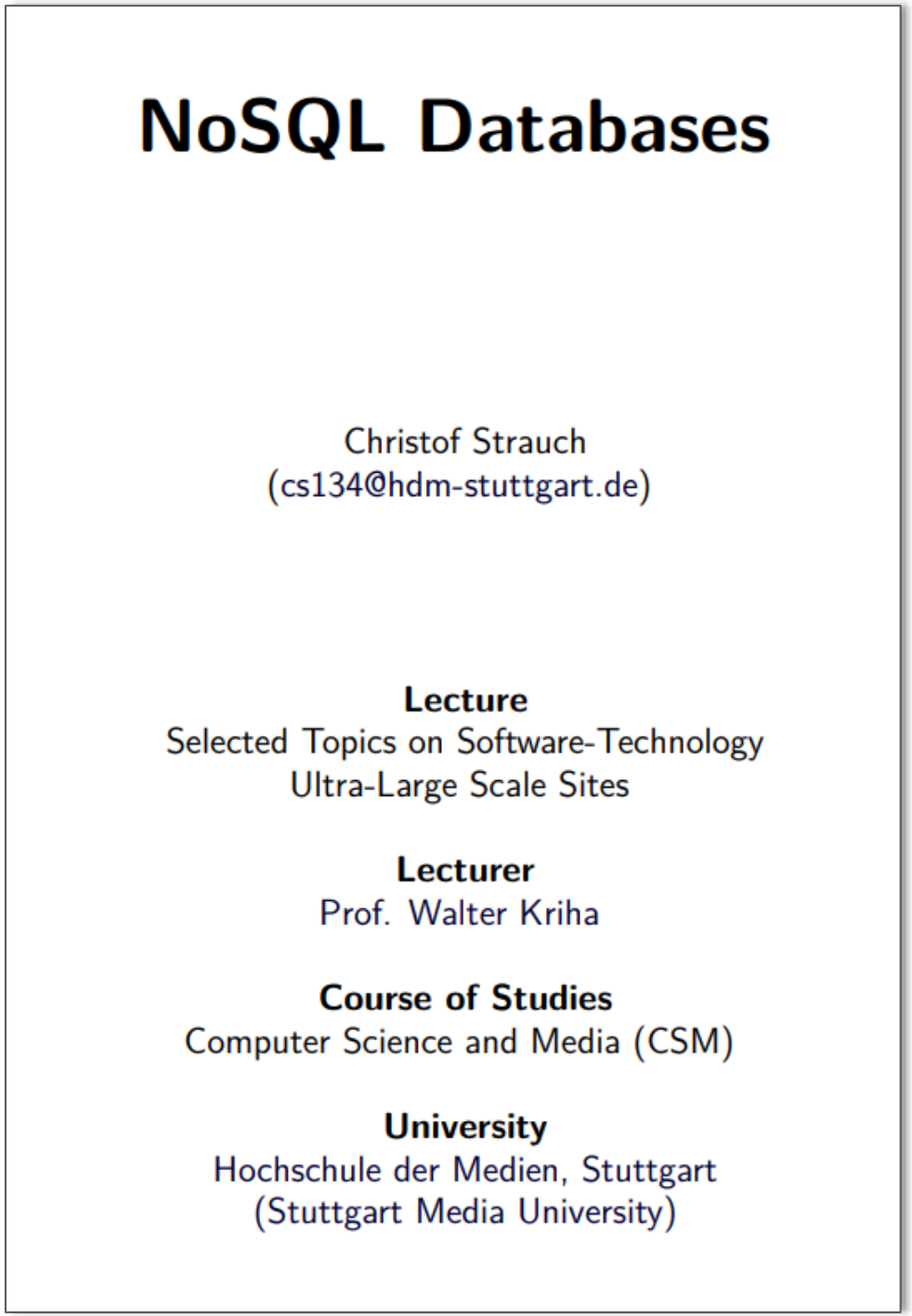
- **Dokumentumtárolók** MongoDB, CouchDB
- **Oszlopcsaládok** Cassandra, HBase
- **Kulcs-érték tárolók** BerkeleyDB, Redis, Memcached
- **Gráfadatbázisok** Neo4j

“NOSQL DATABASES” JEGYZET

Christof Strauch, *NoSQL Databases*,
Stuttgart Media University, 2009-2011

1.2 Uncovered topics

The class of graph databases is also left out of this paper but some resources are provided in the appendix A.



NoSQL Databases

Christof Strauch
(cs134@hdm-stuttgart.de)

Lecture
Selected Topics on Software-Technology
Ultra-Large Scale Sites

Lecturer
Prof. Walter Kriha

Course of Studies
Computer Science and Media (CSM)

University
Hochschule der Medien, Stuttgart
(Stuttgart Media University)

“NOSQL DATABASES” JEGYZET

	Performance	Scalability	Flexibility	Complexity	Functionality
Key-Value Stores	high	high	high	none	variable (none)
Column stores	high	high	moderate	low	minimal
Document stores	high	variable (high)	high	low	variable (low)
Graph databases	variable	variable	high	high	graph theory
Relational databases	variable	variable	low	moderate	relational algebra

Table 2.4.: Classifications – Categorization and Comparison by Scofield and Popescu (cf. [Pop10b], [Sco10])

Látni fogjuk, hogy a “functionality” besorolás nem teljesen pontos.

ADATBÁZISOK, NOSQL FEJEZET, 2012-

(Á. Barabás, G. Szárnyas, S. Gajdos)

Példák:

- Neo4j
- AllegroGraph
- HypergraphDB
- InfiniteGraph
- FlockDB

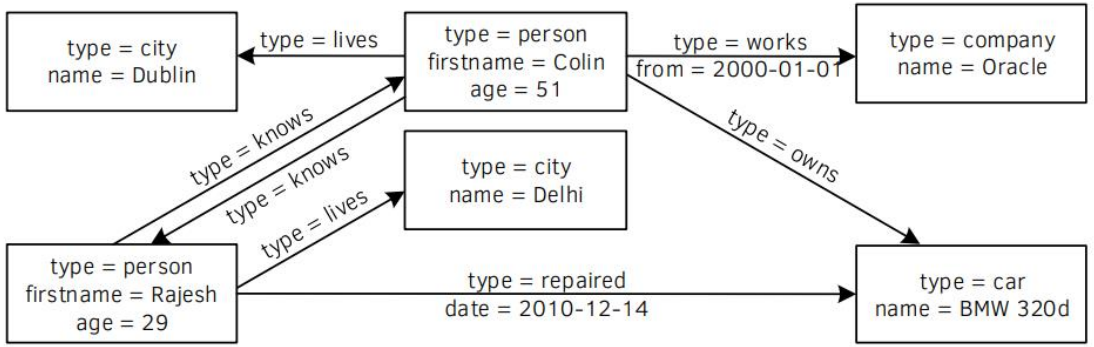
C.5.4. Gráfadatbázisok

Komplex, sok összefüggést tartalmazó adathalmazokat gyakran célszerű gráffal reprezentálni. Sajnos a gráfok relációs adatbázis-kezelőkben való tárolása esetén a gráfokon végzett műveletek rendkívül költségesek lehetnek: egy gráf bejárásához például több természetes illesztésre lehet szükség, ami köztudottan költséges művelet.

A *gráfadatbázisok* (graph database) gráfok hatékony tárolását és ezáltal gráfműveletek gyors végrehajtását teszik lehetővé [35].

A gráfadatbázisok jellemzően *tulajdonséggráfokat* (property graph) tárolnak, melyek csomópontjaihoz és éleihez tulajdonságok köthetők – rendszerint kulcs-érték párok formájában. Ezen tulajdonságok között jellemzően megtalálható az adott csomópontok és élek típusa is.

A C.13. ábrán látható tulajdonséggráfról például a következő állítást olvashatjuk le: az 51 éves Colin Dublinban lakik.



C.13. ábra. Tulajdonséggráf

A gazdag adatmodell miatt a gráfadatbázisok jellemzően kevésbé skálázódnak a többi NoSQL rendszernél, a legtöbb gráfadatbázis csak replikációt támogat.

Példák gráfadatbázisokra: Neo4j, AllegroGraph, HypergraphDB, InfiniteGraph, FlockDB.

Adatmodell

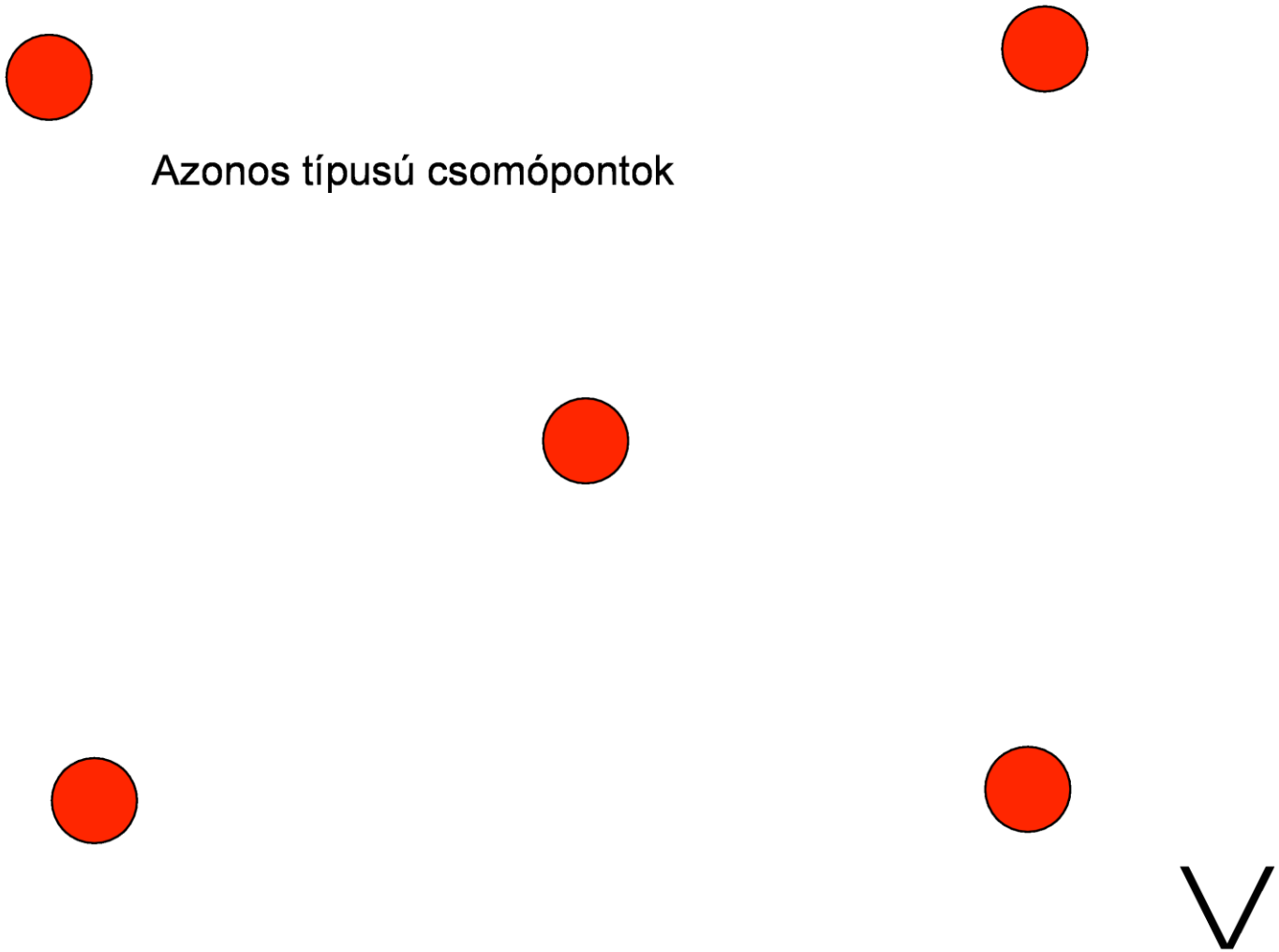
GRÁF ADATMODELL

$$G = (V, E)$$

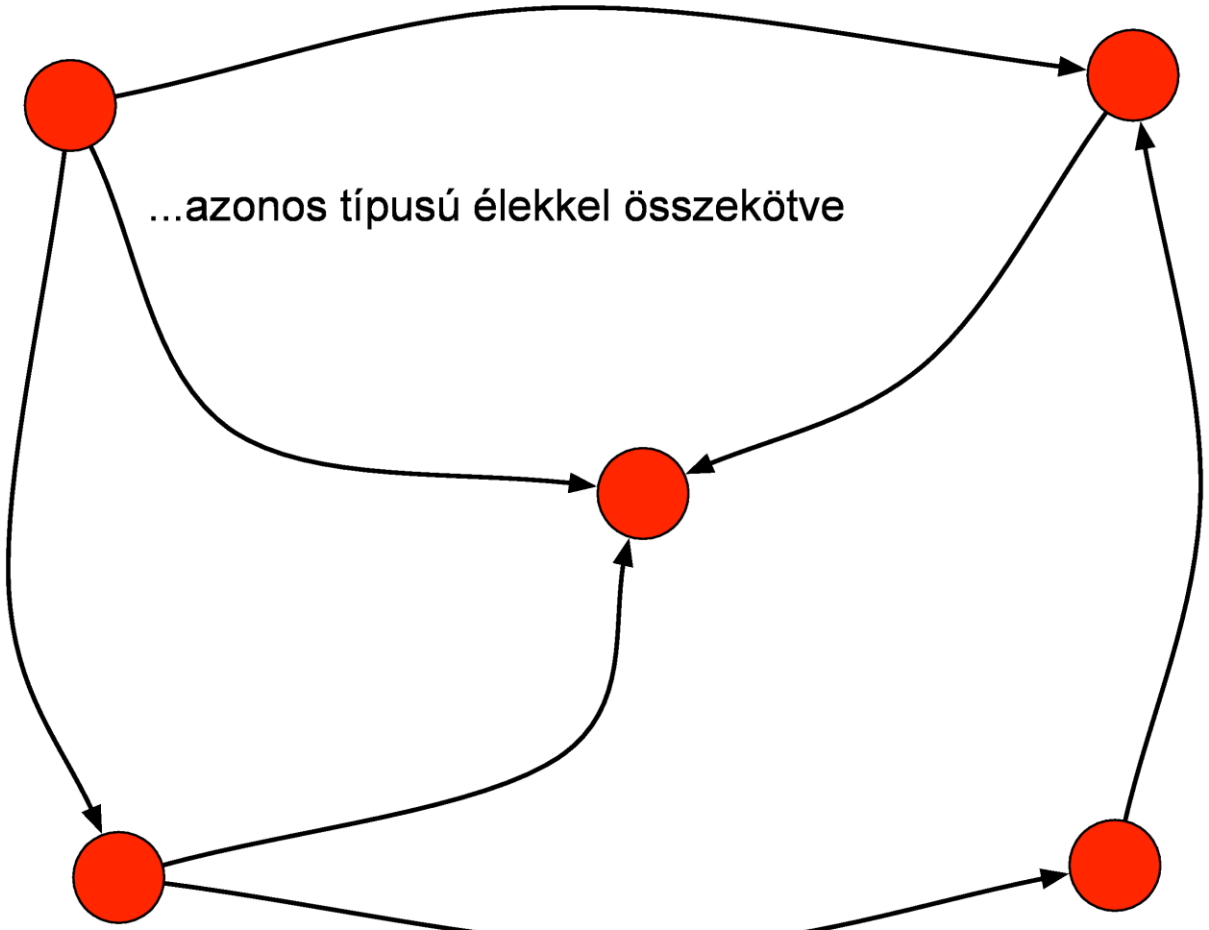
"textbook graph"

Forrás – Titan: The Rise of Big Graph Data

GRÁF ADATMODELL

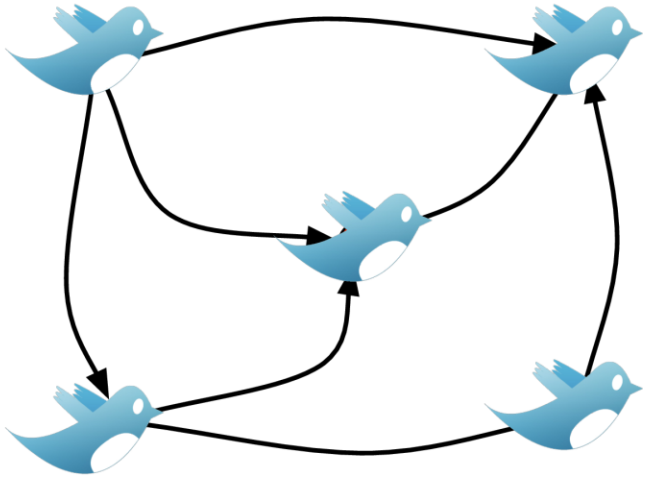


GRÁF ADATMODELL



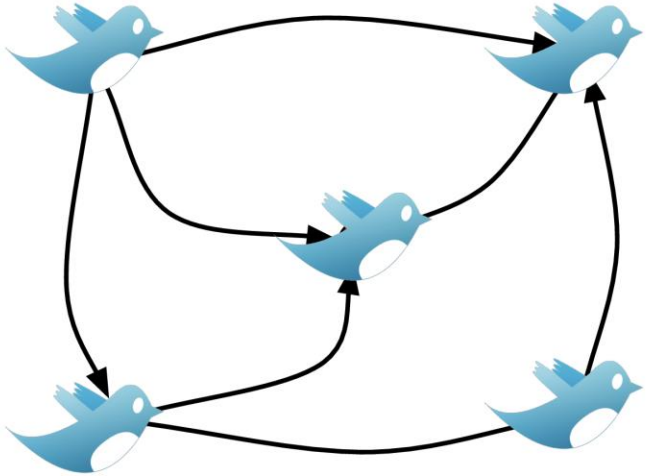
E

KORLÁTOZOTT KIFEJEZŐERŐ



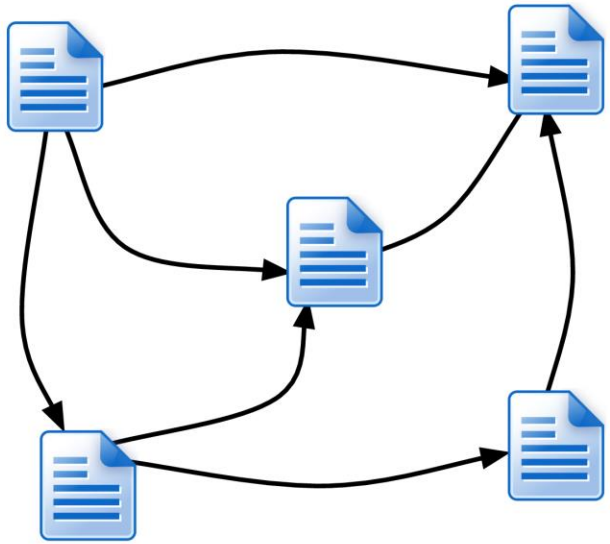
Twitter: ki kit követ

KORLÁTOZOTT KIFEJEZŐERŐ

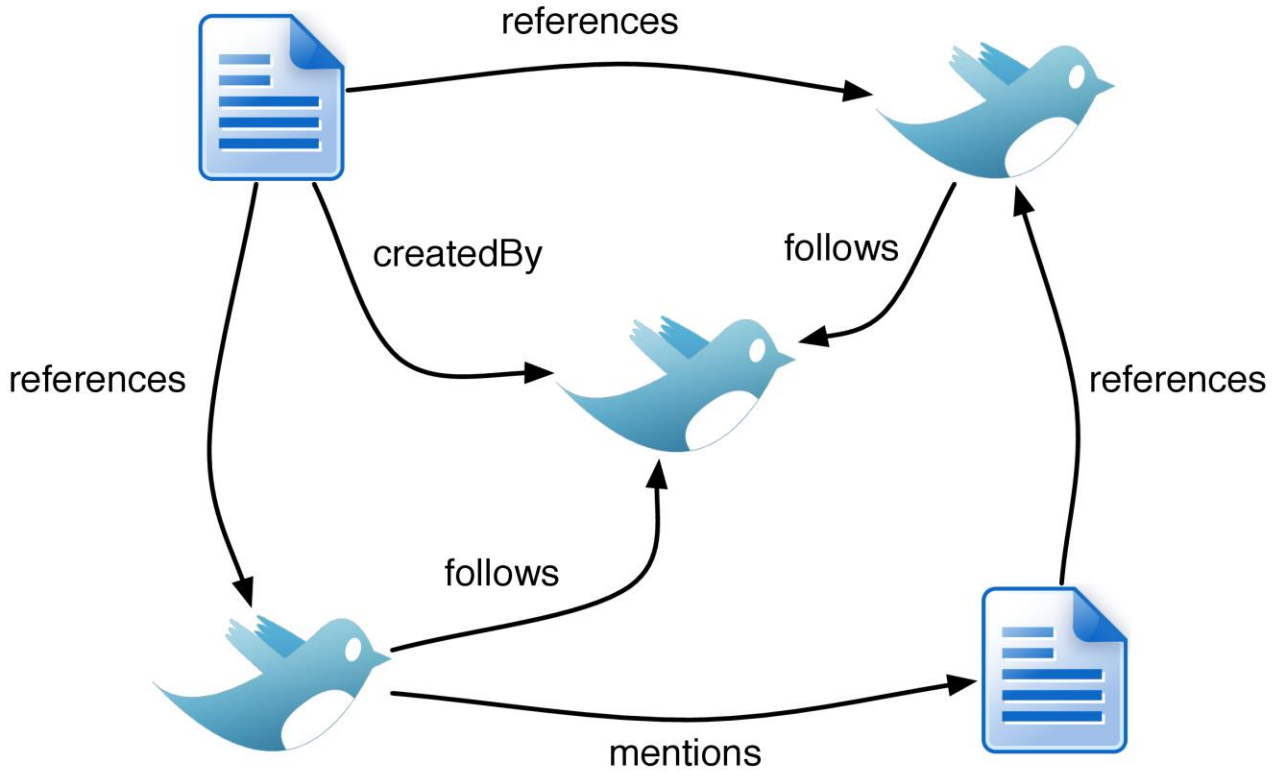


Twitter: ki kit követ

...weboldalak és linkek



KÜLÖNBÖZŐ KAPCSOLATOK EGYÜTT



TULAJDONSÁGGRÁF

- Property graph (PG)
- Labelled property graph (LPG)
- Attributed graph

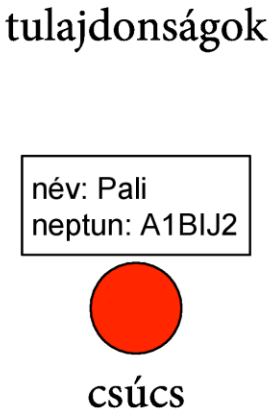
$$G = (V, E, \lambda)$$

* Irányított gráf, élek: címkék, csomópontok: tulajdonságok
* Többszörös élek lehetségesek

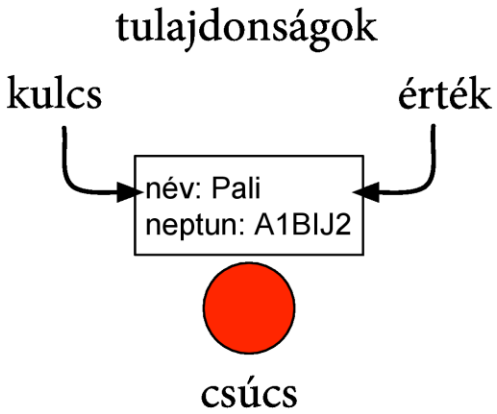
TULAJDONSÁGGRÁF

csúcs

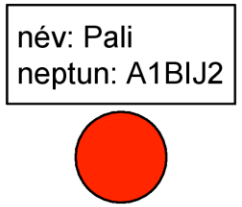
TULAJDONSÁGGRÁF



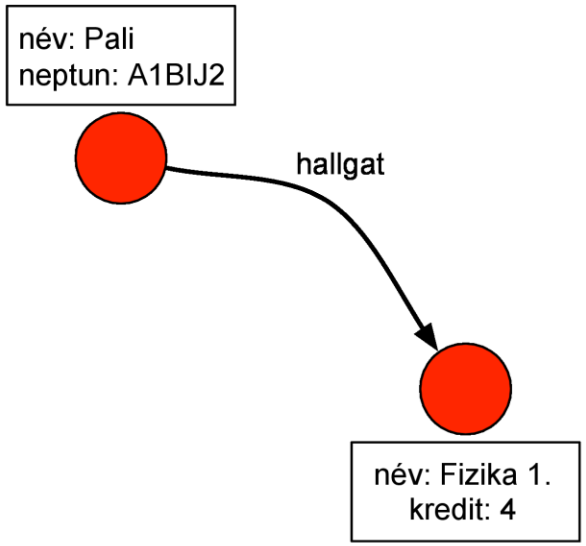
TULAJDONSÁGGRÁF



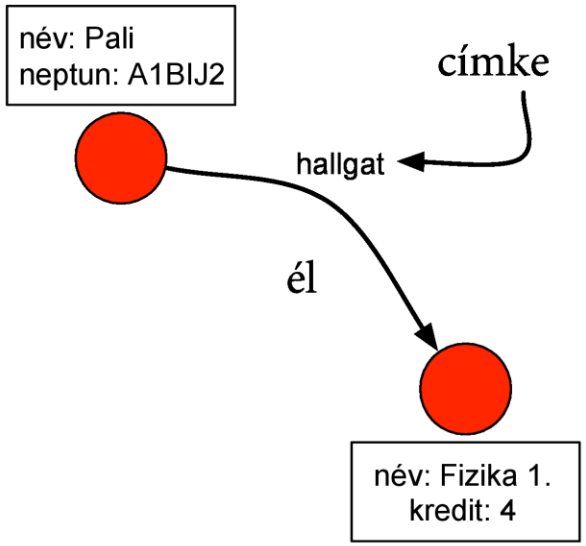
TULAJDONSÁGGRÁF



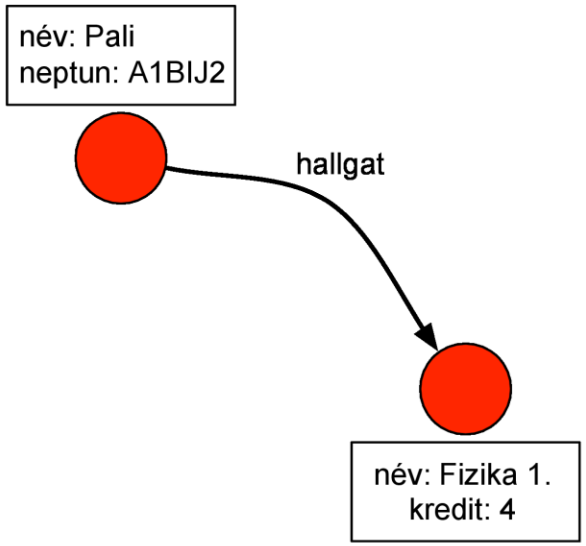
TULAJDONSÁGGRÁF



TULAJDONSÁGGRÁF



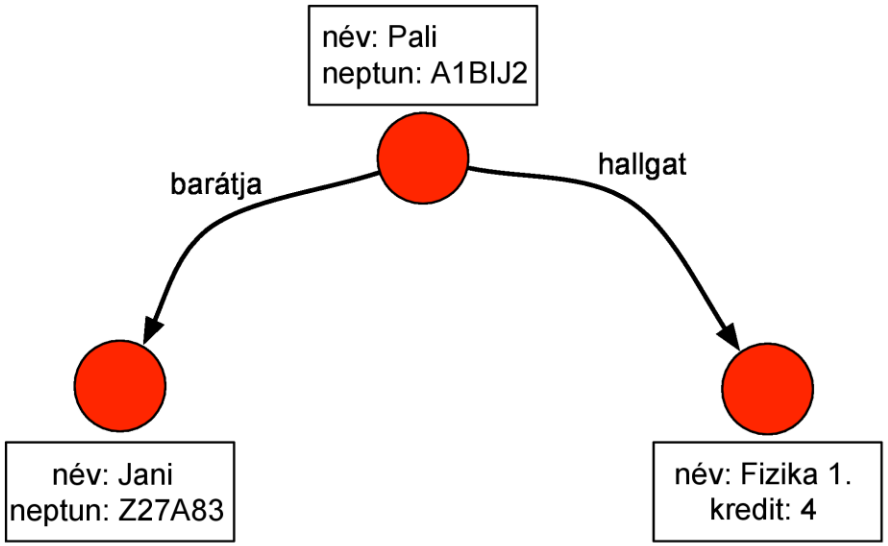
TULAJDONSÁGGRÁF



TULAJDONSÁGGRÁF

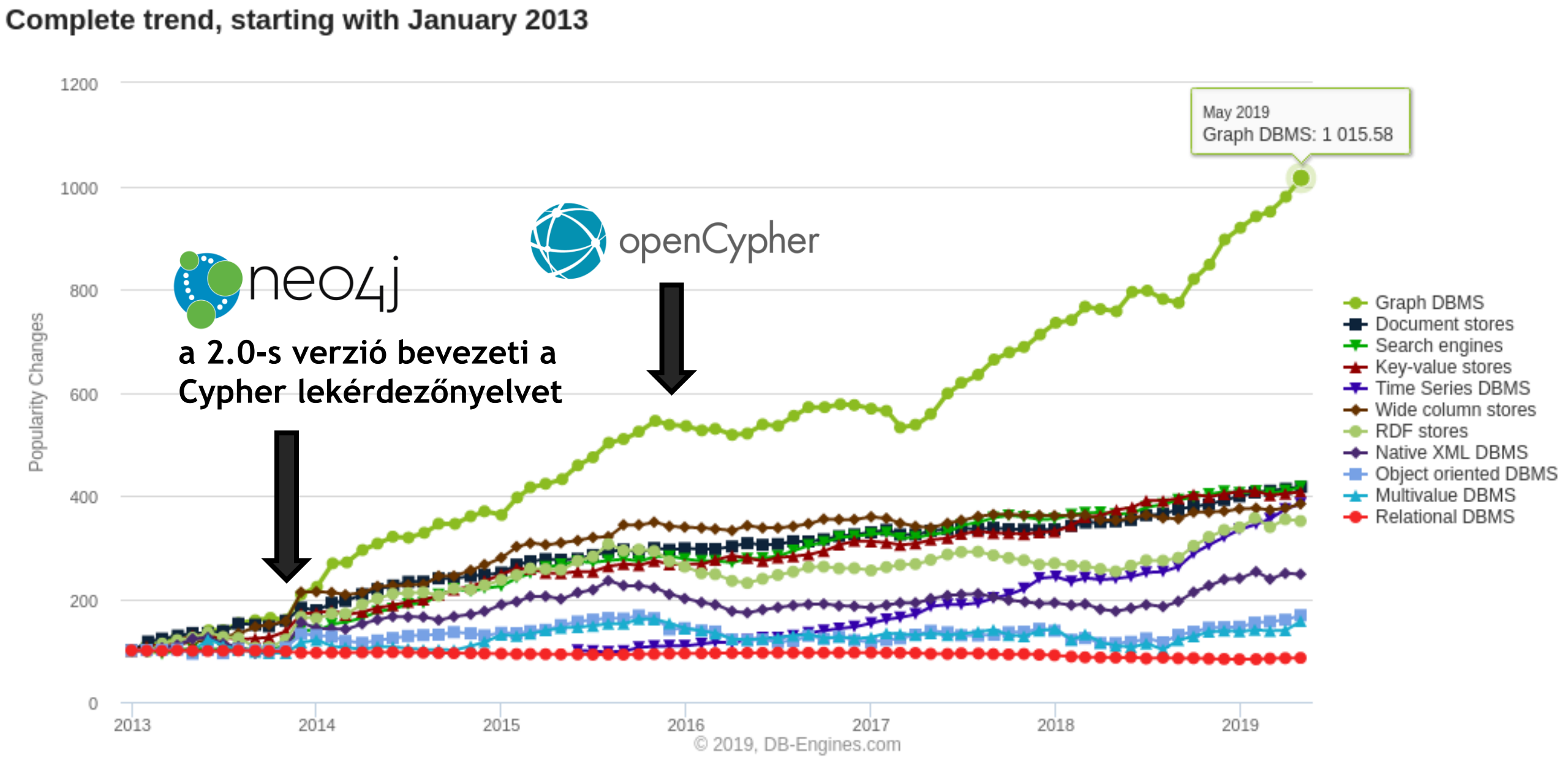
Filozófia:

- Miért jó ez?
- Akkor nem elterjedtebb?



Sőt: kollekciók (lista, map) is lehet tulajdonság értéke

ADATMODELLEK RELATÍV NÉPSZERŰSÉGE



GRÁF ADATMODELL FOGALMAI

- Csomópont vertex (node)
- Élek edge (relationship)
- Attribútum attribute (property)
- Útvonal simple path (path)
- Élsorozat walk (path)
- Séta trail (path)

TL;DR:
definiáljuk pontosan a fogalmakat

3.14. Definíció *A G gráf* élsorozata *egy olyan* $(v_1, e_1, v_2, e_2, \ldots, v_k)$ *sorozat, amire* $e_i \in E(G)$ *és* $e_i = v_i v_{i+1}$ $(\forall 1 \leq i < k)$. *A* séta *olyan élsororzat, aminek minden éle különböző.*

Séták [szerkesztés]

Egy **séta** csúcsok és élek váltakozó sorozata, mely csúccsal kezdődik és csúcsban végződik, és minden csúcs szomszédos az őt megelőző és őt követő éllel, illetve minden él két végpontja az őt megelőző és az őt követő csúcs.

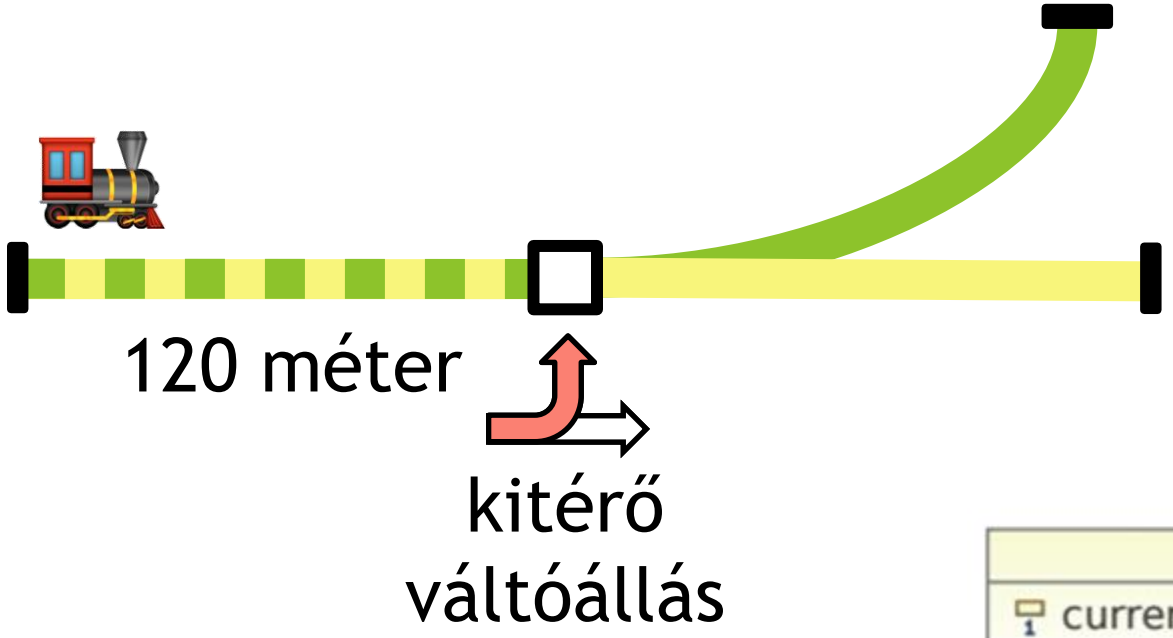
ADATMODELLEK LEKÉPEZÉSE

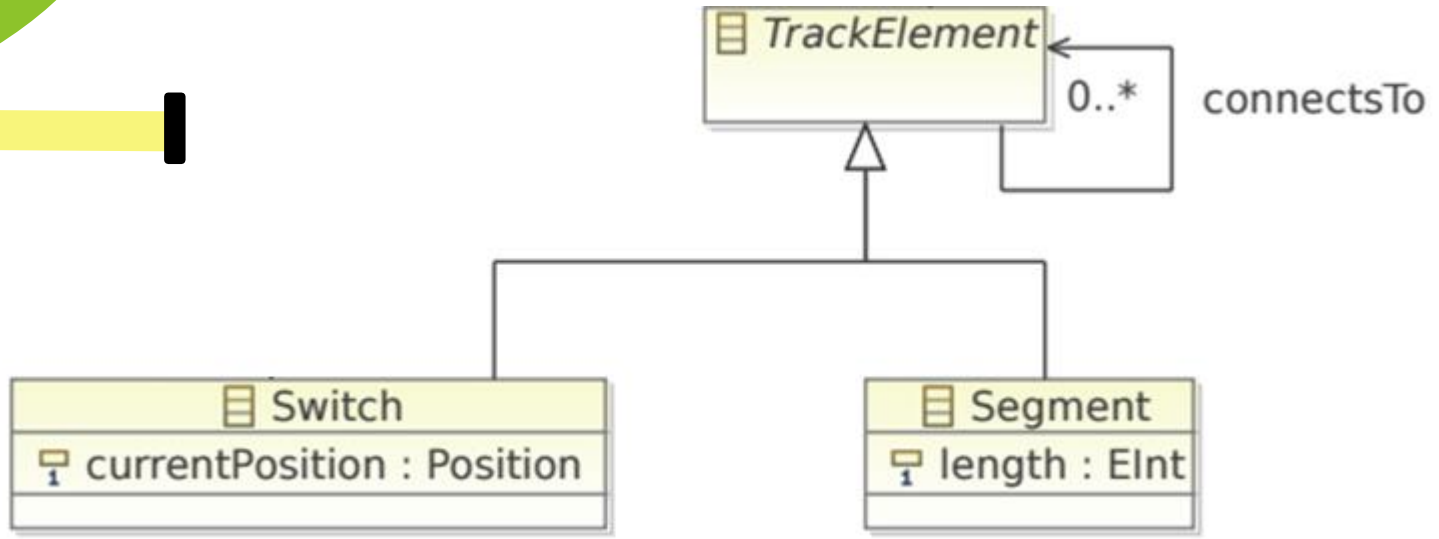
OO	tulajdonsággráf	RDF	SQL
osztály def.	csomópont címke	rdfs:class	tábla
referencia def.	élcímke	rdf:Property, owl:ObjectProperty	idegen kulcs
attribútum def.	tulajdonság név	rdf:Property, owl:DataTypeProperty	oszlop def.
attribútum típusa	tulajdonság típusa	rdfs:domain	oszlop típusa
ősosztály	OGM probléma	rdfs:subClassOf	ORM probléma

- $O(n^2)$ nagyságrendnyi leképezés lehetséges
- Köztes nyelvvel levihető $O(n)$-re ($\sim 2n$), de ez ritkán hatékony

Antal János Benjamin, Elekes Márton:
Gráf információs rendszerek összehasonlító teljesítménymérése.
OTDK 1. hely

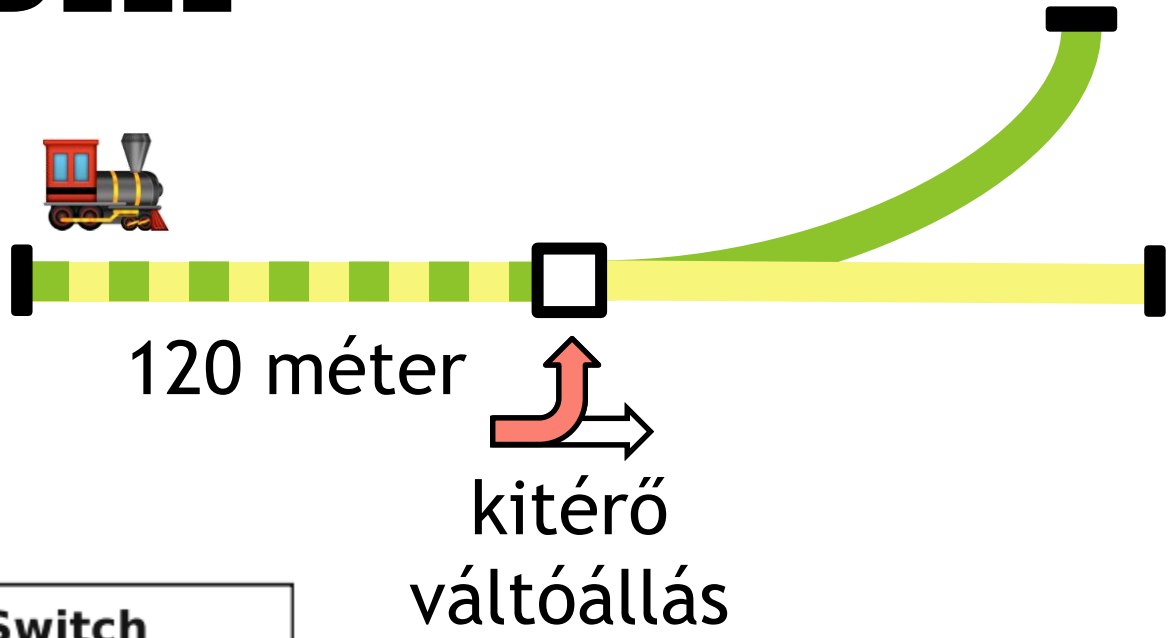
ÖRÖKLÉS ÉS ATTRIBÚTUMOK MODELLEZÉSE



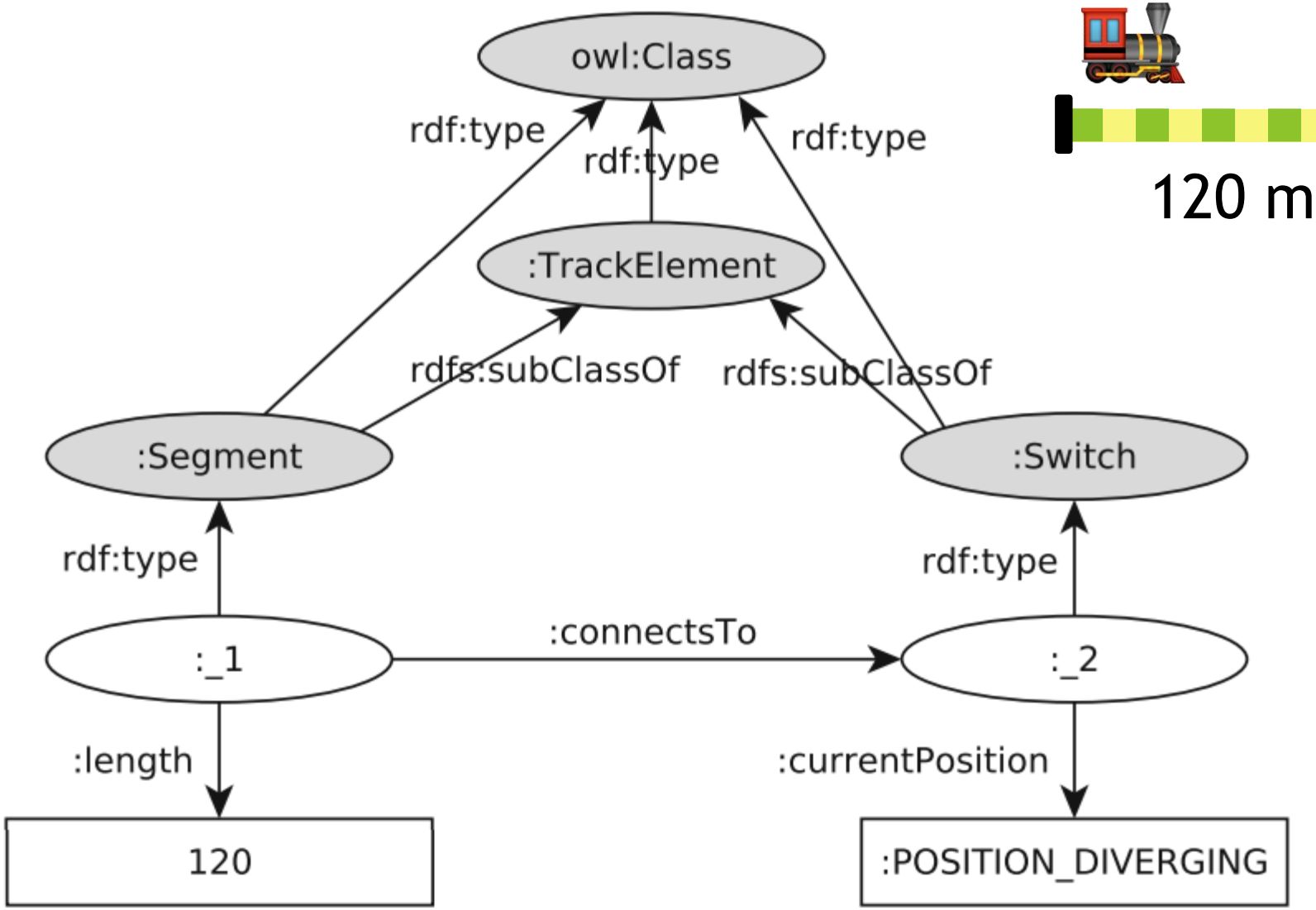


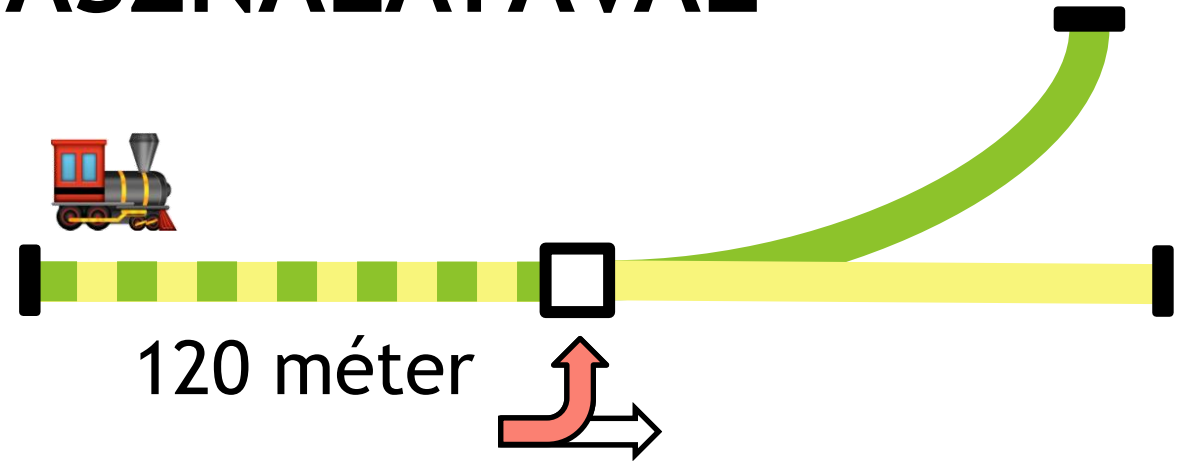
G. Szárnyas, B. Izsó, I. Ráth, D. Varró,
The Train Benchmark: cross-technology performance evaluation of continuous model queries,
Software and Systems Modeling, 2017

OBJEKTUM-ORIENTÁLT MODELL

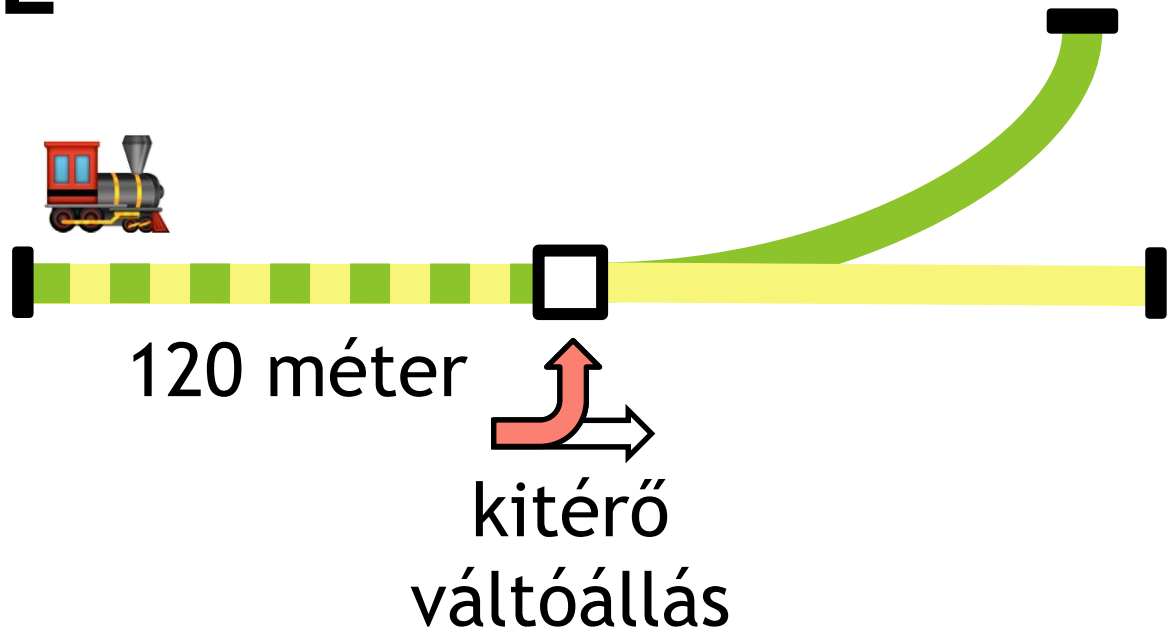


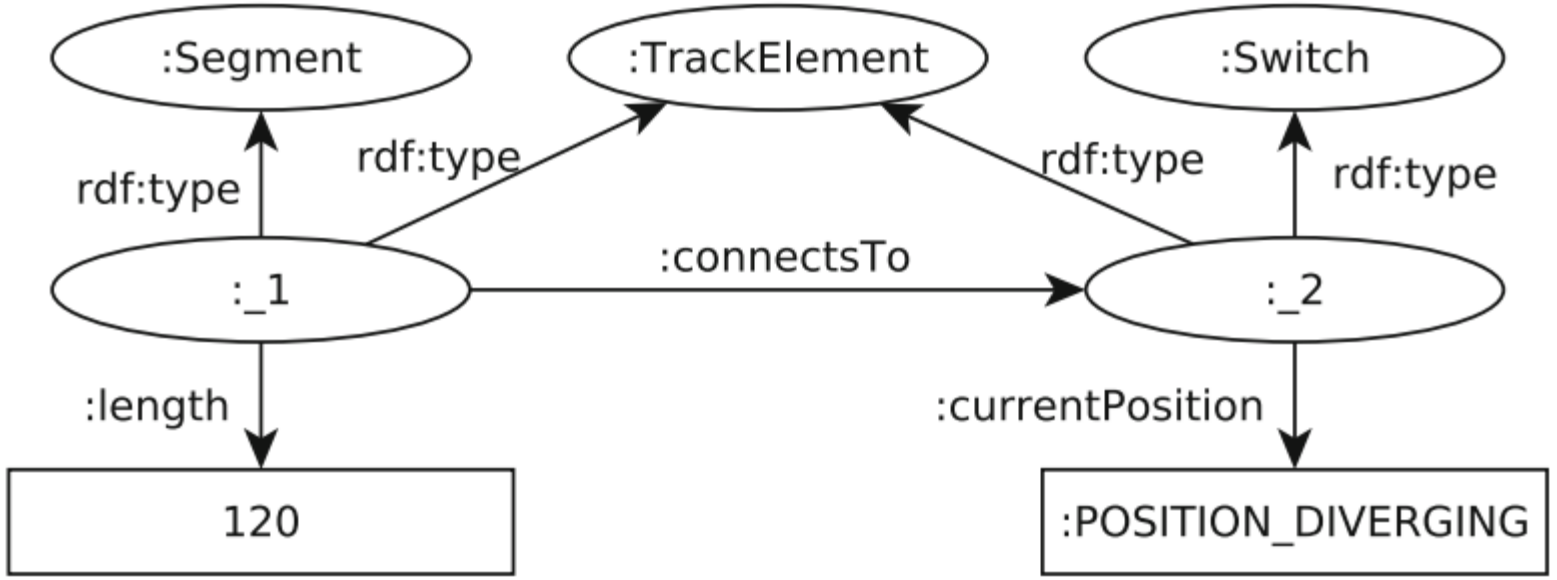
RDF / OWL METAMODELL HASZNÁLATÁVAL



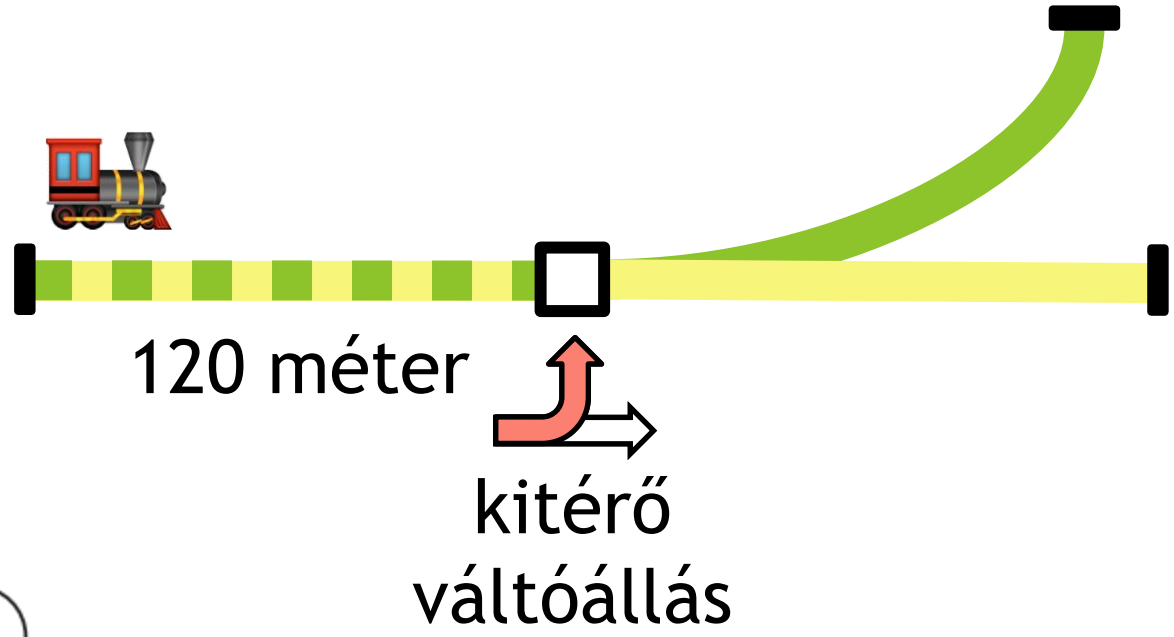


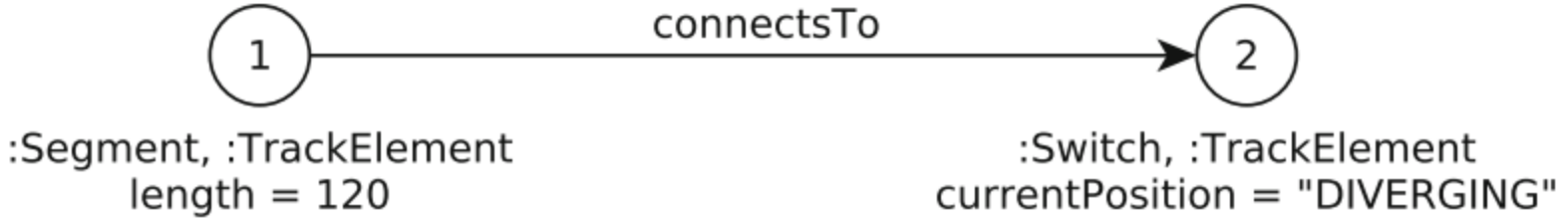
RDF / METAMODELL NÉLKÜL





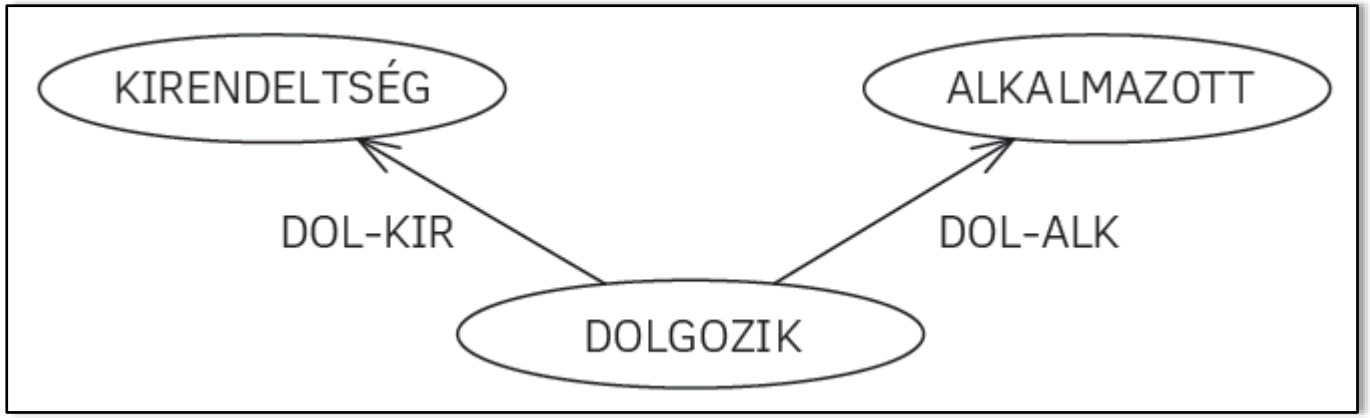
TULAJDONSÁGGRÁF





HÁLÓS ADATMODELL

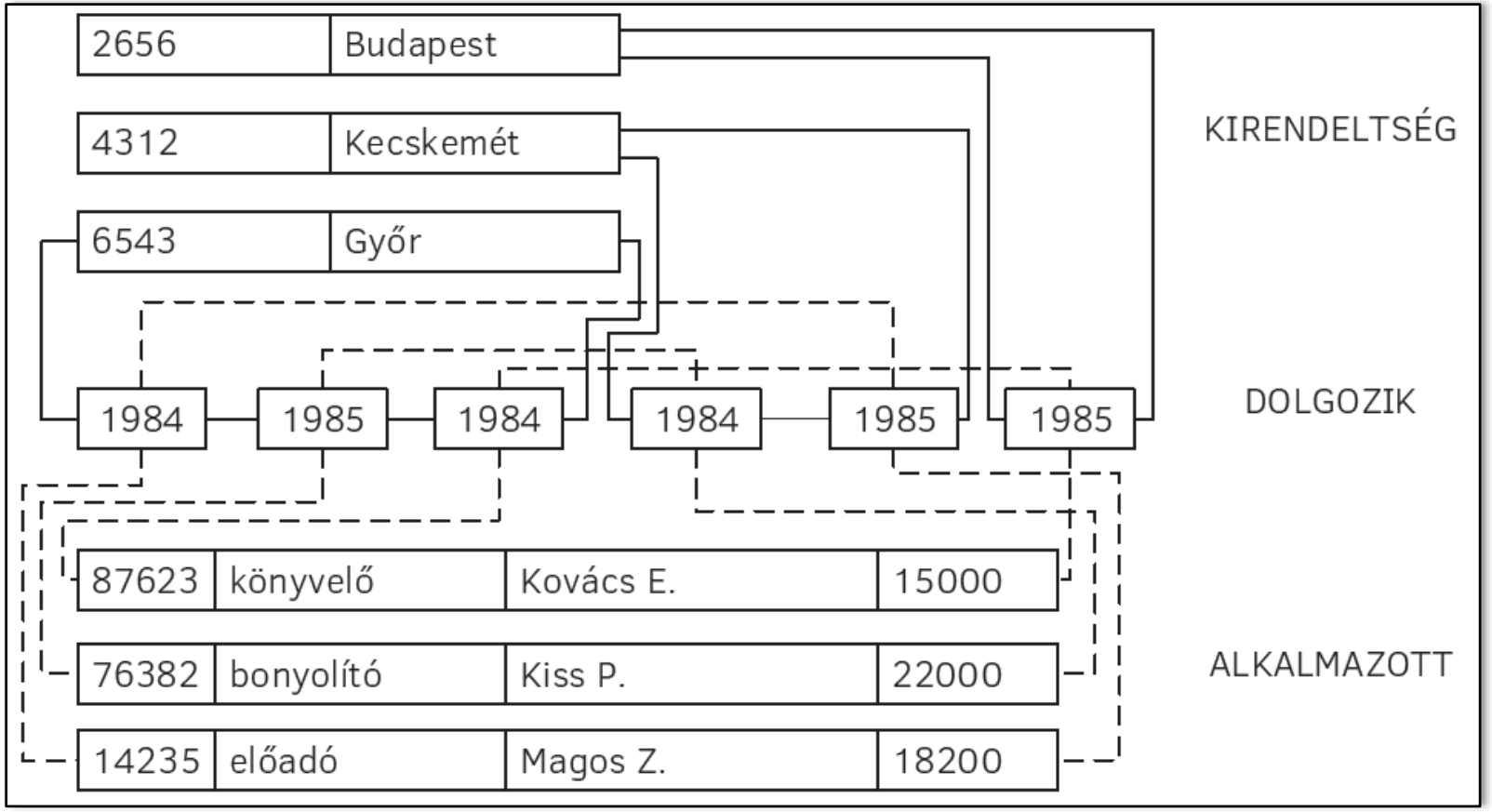
- 1969 CODASYL
- A tulajdonsággráfokhoz hasonló kifejezőerő
- Lekérdezés imperatív programozási nyelvvel



PG-hez képest:

- Rugalmatlan adatmodell
- Kollekciók hiánya
- Lekérdezőnyelv hiánya

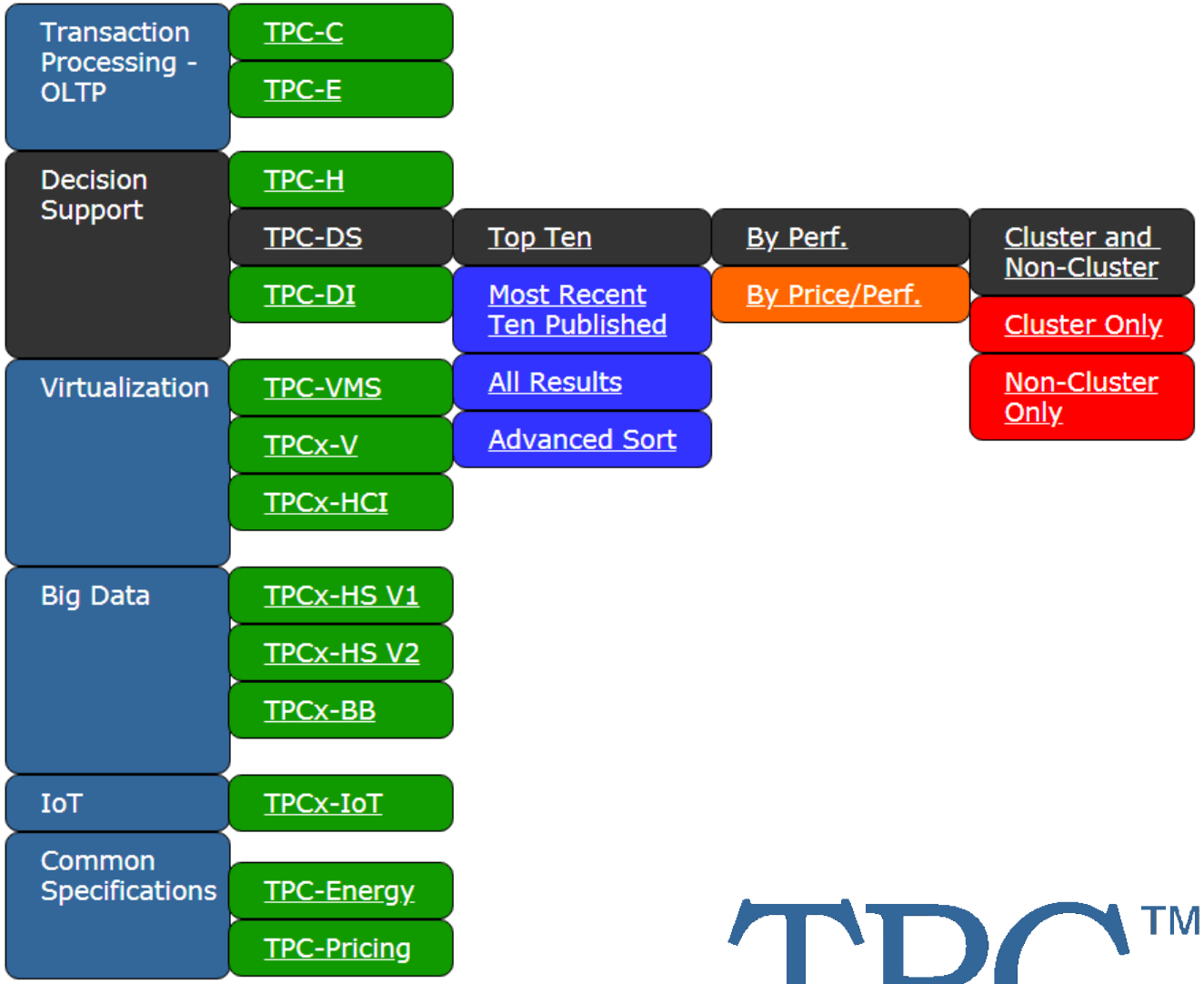
(n.b. ~40 év különbség)



Gráffeldolgozási megközelítések

TRANSACTION PROCESSING PERFORMANCE COUNCIL (1988-)

Szabványos specifikációk relációs adatbázisok teljesítményméréséhez



TPC-DS

- 99 lekérdezés
- Mérföldkövek
  - 2002: első publikáció
  - 2006: második publikáció
  - 2018: első auditált mérések

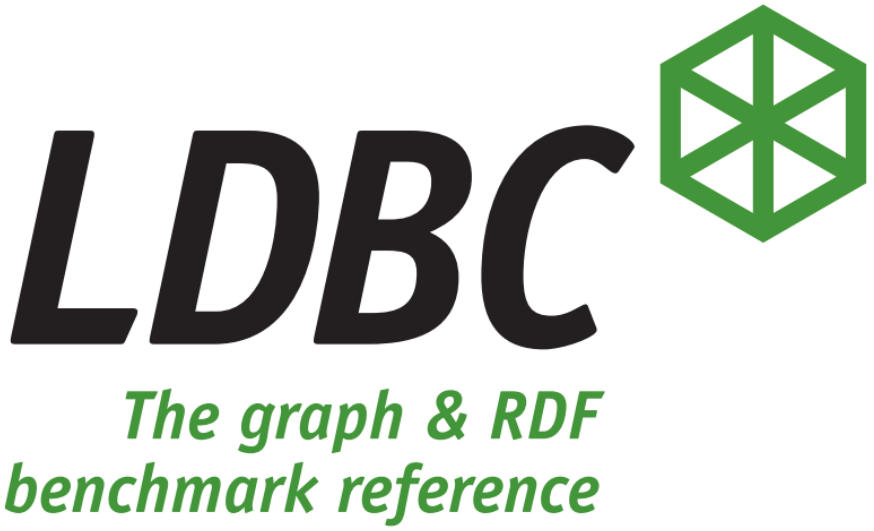
  -> 16 év

TPC™

LINKED DATA BENCHMARK COUNCIL (2012-)

LDBC is a non-profit organization dedicated to establishing benchmarks, benchmark practices and benchmark results for graph data management software.

LDBC's Social Network Benchmark is an industrial and academic initiative, formed by principal actors in the field of graph-like data management."



GRÁFFELDOLGOZÁSI MEGKÖZELÍTÉSEK

lokális lekérdezések	kevés adat

Példa: „ismerősök új lájkjai”

```
MATCH (u:User {id: $userId})-[:FRIEND]-
      (f:User)-[l:LIKES]->(p:Post)
RETURN f, p
ORDER BY l.timestamp DESC
LIMIT 10
```

globális lekérdezések
gráfanalitika

GRÁFFELDOLGOZÁSI MEGKÖZELÍTÉSEK

lokális lekérdezések	kevés adat
globális lekérdezések	sok adat

Példa: „egyoldalú barátságok”

```
MATCH (u1:User)-[:FRIEND]-(u2:User)-[l:LIKES]->(p:Post),
      (u1)-[:AUTHOR_OF]->(p)
WITH u1, u2, count(l) AS likes
WHERE likes > 10
  AND NOT (u1)-[:LIKES]->(:Post)<-[:AUTHOR_OF]-(u2)
RETURN u1, u2
```

gráfanalitika

GRÁFFELDOLGOZÁSI MEGKÖZELÍTÉSEK

lokális lekérdezések	kevés adat
globális lekérdezések	sok adat
gráfanalitika	minden adat

- PageRank
- Legrövidebb utak
- Klaszterezettség

Példa: „Találjuk meg a központi embereket.”

Külön tudományterület: hálózatkutatás

GRÁFFELDOLGOZÁSI MEGKÖZELÍTÉSEK

lokális lekérdezések	kevés adat
globális lekérdezések	sok adat
gráfanalitika	minden adat

Lekérdezések tulajdonsággráfokon

CYPHER LEKÉRDEZŐNYELV

Cypher: a Neo4j gráfadatbázis lekérdezőnyelve.

„Cypher is a declarative, SQL-inspired language for describing patterns in graphs visually using an ascii-art syntax.”

```
MATCH
  (p:Person)-[:LECTURER_OF]->(l:Lecture)-[:OF]->(c:Course)
WHERE l.date = '2019. május 2.'
  AND c.name = 'Adatbázisok haladóknak'
RETURN p
```

„The openCypher project aims to deliver a full and open specification of the industry’s most widely adopted graph database query language: Cypher.” (2015)

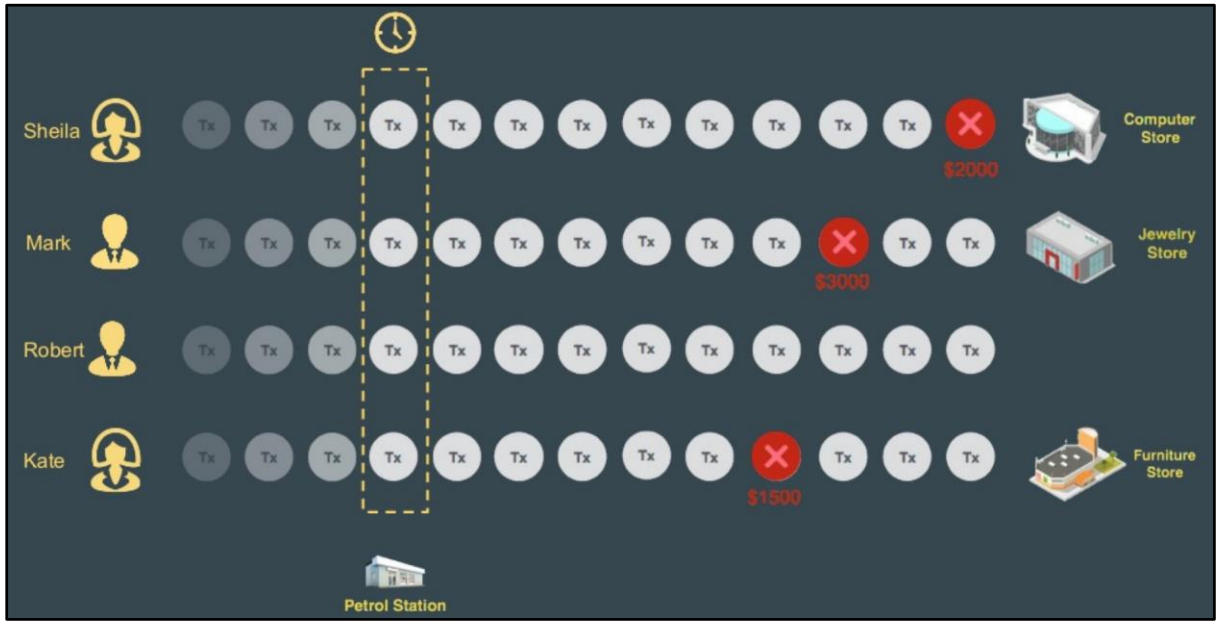
OPENCYPHER RENDSZEREK

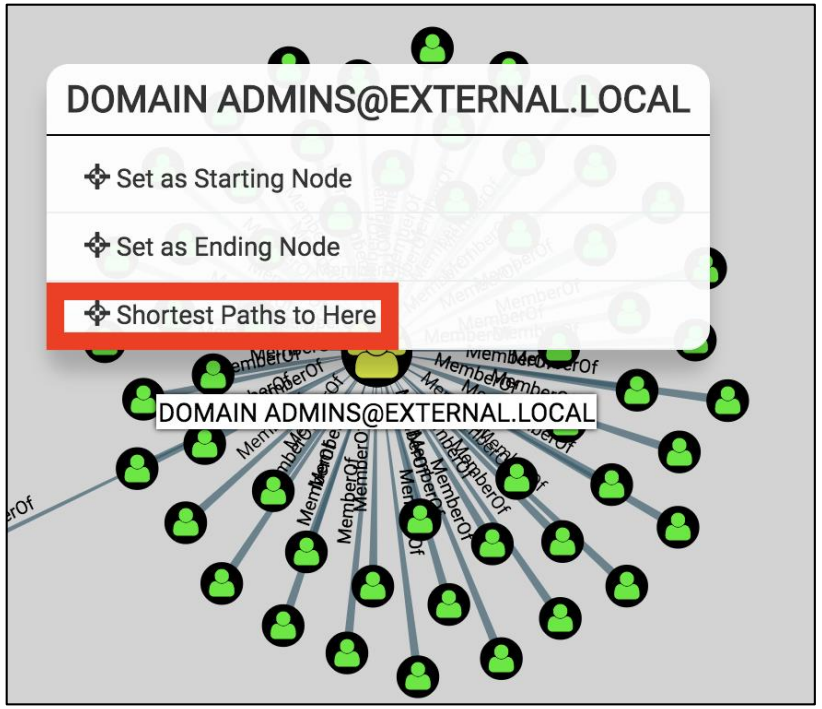
- Cél: teljes és nyílt specifikáció a Cypher nyelvhez
- Relációs adatbázisok:
  - SAP HANA
  - AGENS Graph (PostgreSQL alapokon)
- Kutatási prototípusok:
  - Graphflow (Univesity of Waterloo)
  - ingraph (incremental graph engine – BME, MTA)

(Kép forrása: Keynote előadás @ GraphConnect NYC 2017)

HASZNÁLATI ESETEK

- IT biztonság
- Oknyomozó újságírás
- Csalásdetekció
- Ajánlórendszerek





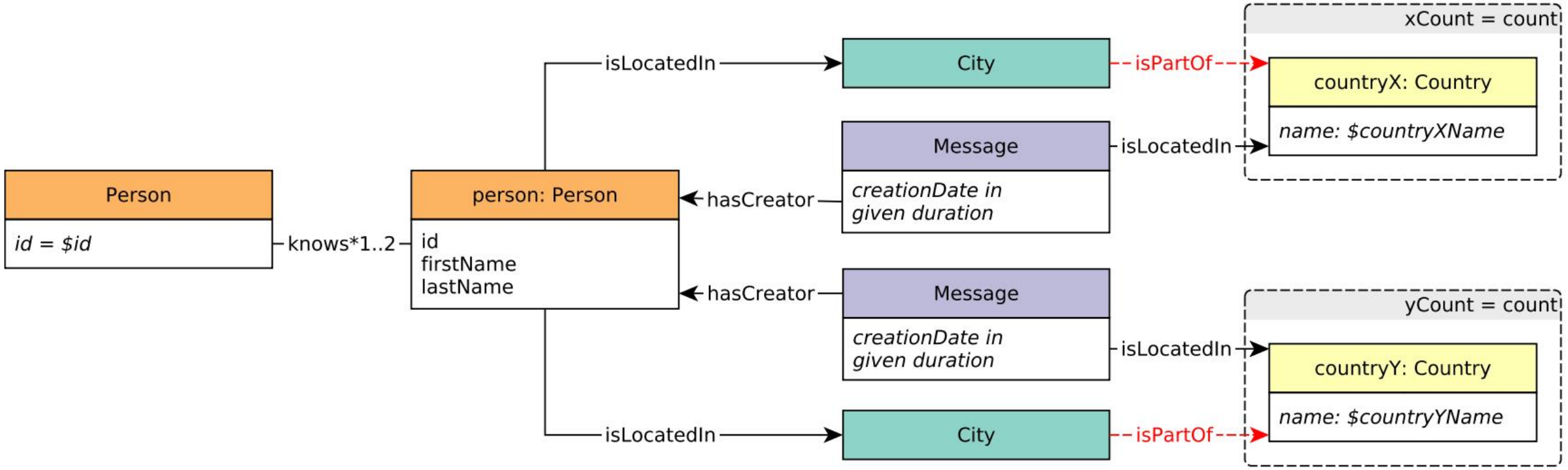
Walmart

Walmart uses Neo4j to optimize customer experience with personal recommendations

Minta lekérdezések

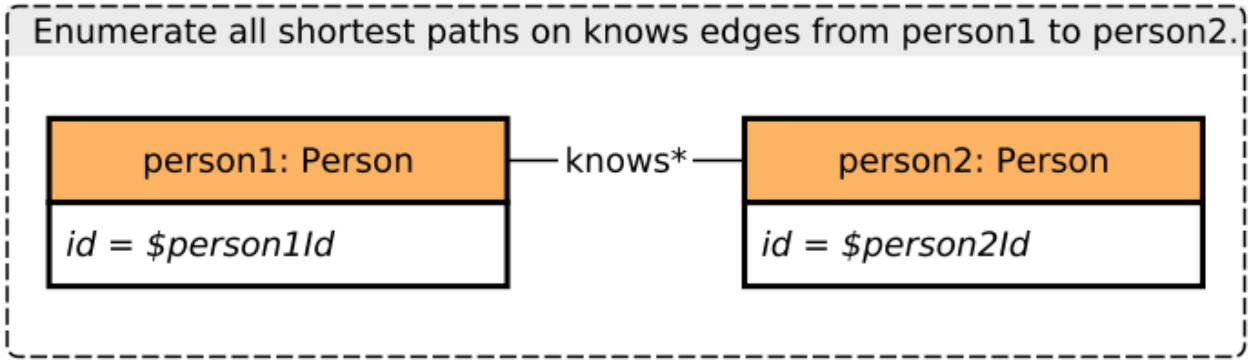
LDBC INTERACTIVE 3

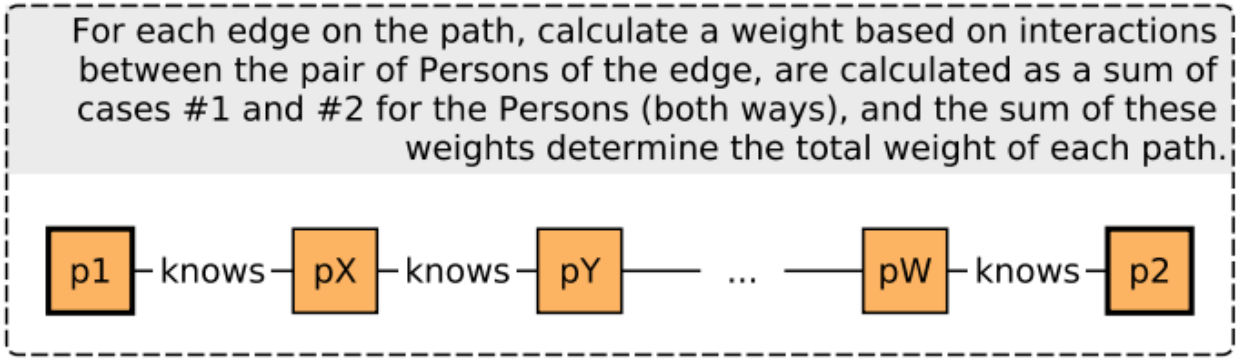
Friends and friends of friends that have been to countries X and Y

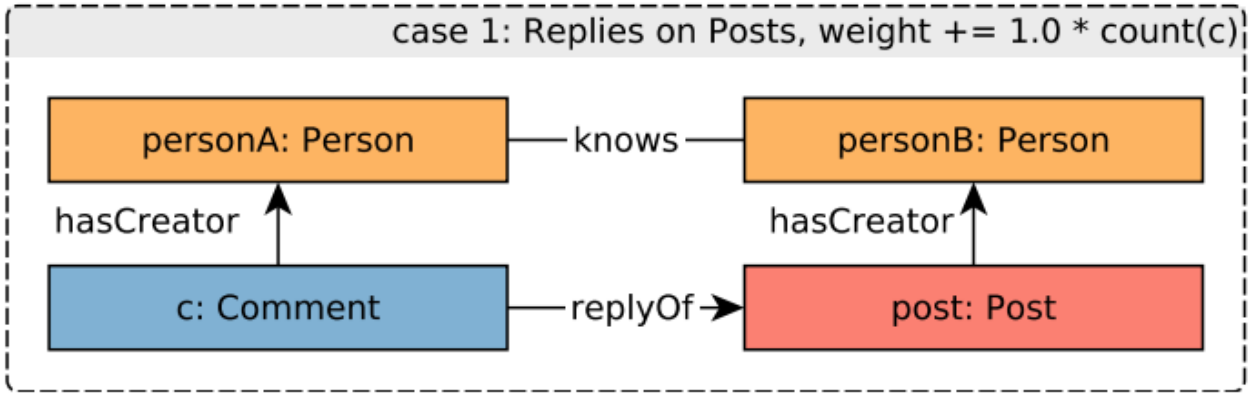


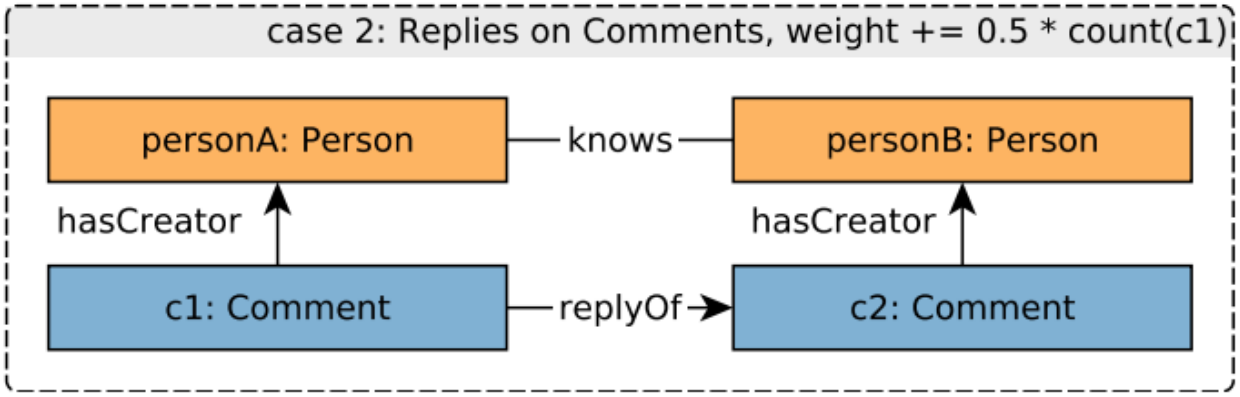
LDBC INTERACTIVE 14

Trusted connection paths

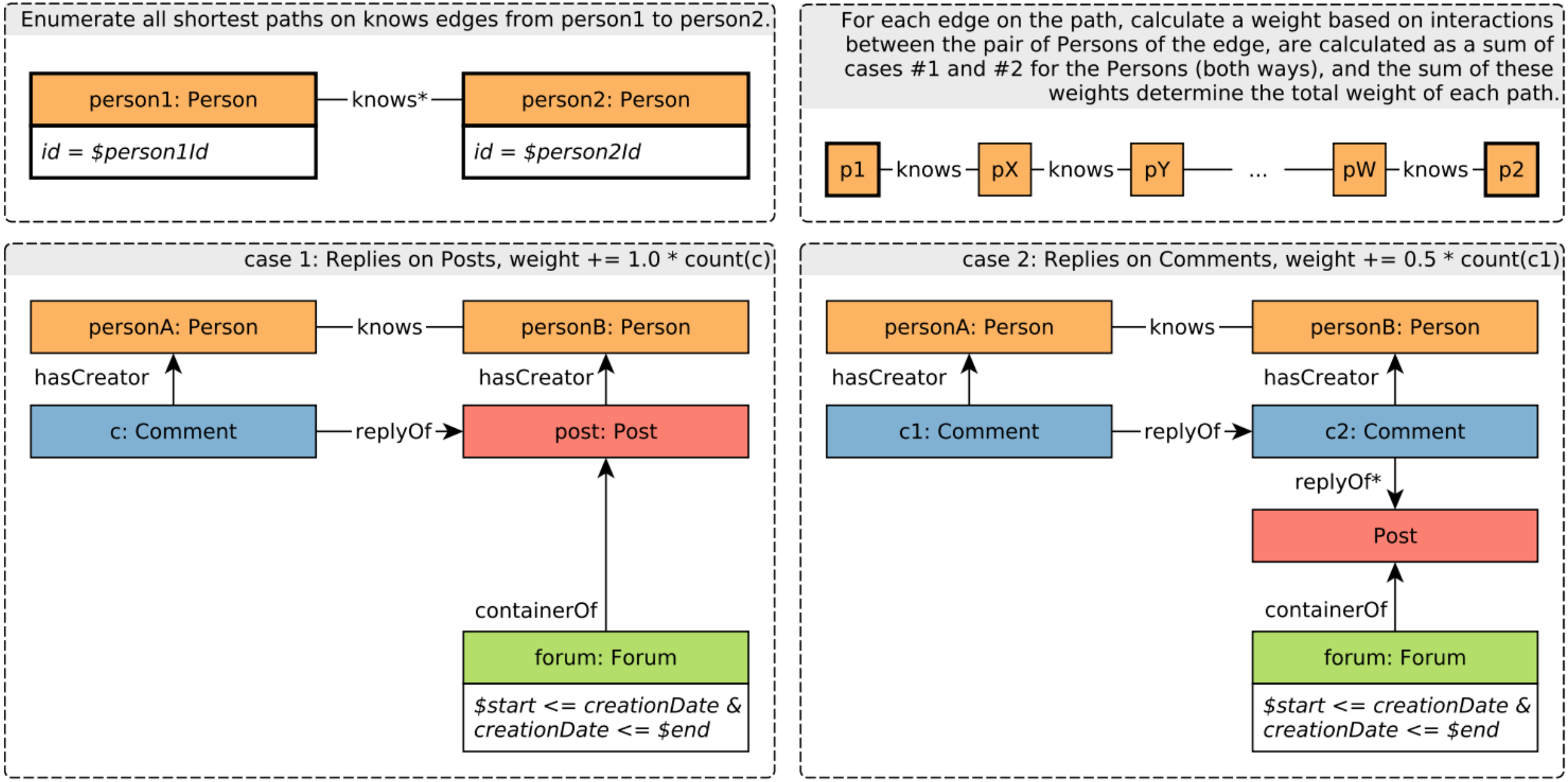








LDBC BI 25: TRUSTED CONNECTION PATHS #2



Gráflekérdezések relációalgebrával

RELÁCIÓALGEBRA

- Algebra
  - “A műveletek általános tudománya”
  - Tipikusan zárt rendszert alkot
  - Ekvivalenciaszabályok felírhatók
- Relációalgebra
  - A halmazelmélet bővítése
  - Reláció fogalom
  - Alapműveletek: $\sigma, \pi, \times, \cup, \cap$
  - Származtatott műveletek: $\bowtie$
  - Műveletek költsége becsülhető -> optimalizálás

LEKÉRDEZŐNYELVEK KIFEJEZŐEREJE

- Relációalgebra — relational algebra
- Sor-/oszlopkalkulus — tuple/domain relational calculus
- Elsőrendű logika — first-order logic

(Kifejezőerő szerinti növekvő sorrendben.)

Elérhetőség (reachability) nem fejezhető ki elsőrendű logikában.

Miért?

“You cannot express reachability in the particular language where the only relations available are the incidence relation on the graph and equality, and where quantification is only permitted over elements of the graph.” (Mathematics StackExchange)

ILLESZTÉS OPERÁTOROK

- Theta join $\bowtie_\theta$
- Equijoin $\bowtie$
- Semijoin $\ltimes$ $r \ltimes s = \pi_R r \bowtie s$
- Antijoin $\triangleright$ vagy $\overline{\ltimes}$ $r \triangleright s = r \setminus (r \ltimes s)$
- Left outer join $\rtimes$ $\sim(r \bowtie s) \cup (r \triangleright s)$, plusz nullok

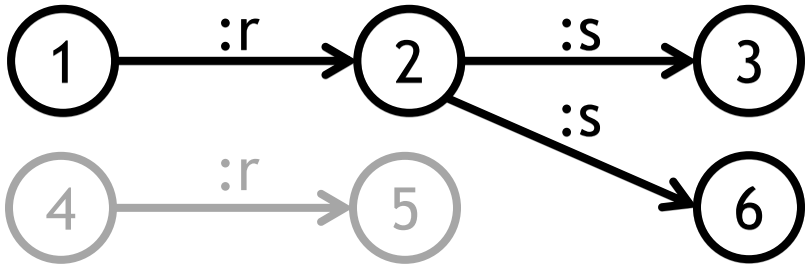
(Lásd még: right outer join $\ltimes$, full outer join $\rtimes\ltimes$.)

<u>Gondolkodtató feladat:</u> semi-/antijoin sor- és oszlopkalkulussal.

ILLESZTÉS JELLEGŰ OPERÁTOROK GRÁFOKON

Natural join: $r \bowtie s$

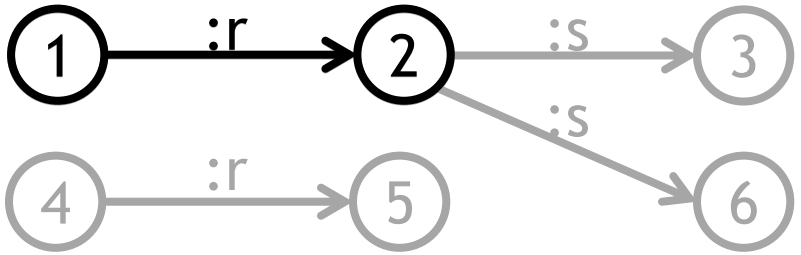
```
MATCH (v1)-[:r]->(v2)-[:s]->(v3)
RETURN *
```



v1	v2	v3
1	2	3
1	2	6

Semijoin: $r \ltimes s$

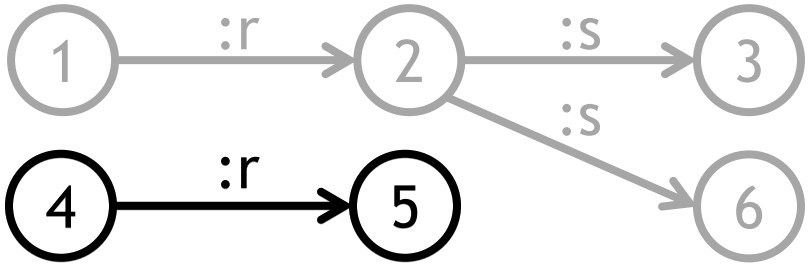
```
MATCH (v1)-[:r]->(v2)
WHERE (v2)-[:s]->()
```



v1	v2
1	2

Antijoin: $r \bar{\ltimes} s$

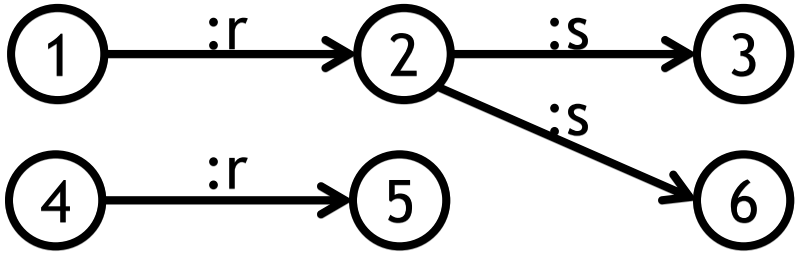
```
MATCH (v1)-[:r]->(v2)
WHERE NOT (v2)-[:s]->()
```



v1	v2
4	5

Left outer join: $r \bowtie s$

```
MATCH (v1)-[:r]->(v2)
OPTIONAL MATCH (v2)-[:s]->(v3)
```



v1	v2	v3
1	2	3
1	2	6
4	5	null

GRÁF RELÁCIÓALGEBRA

- Relációalgebra
  - $\sigma, \pi, \times, \cup, \cap, \bowtie, \rhd$
- Gyakori kiterjesztések
  - aggregáció (γ), duplikátum-elimináció (δ), rendezés (τ), limit (λ)
- Gráf-specifikus kiterjesztések
  - get-vertices ($\bigcirc$)
  - expand-out ($\uparrow$), expand-in ($\downarrow$), expand-both ($\updownarrow$)

Formalising openCypher Graph Queries in Relational Algebra

József Marton[1], Gábor Szárnyas[2,3], and Dániel Varró[2,3]

[1] Budapest University of Technology and Economics, Database Laboratory
marton@db.bme.hu
[2] Budapest University of Technology and Economics
Fault Tolerant Systems Research Group
MTA-BME Lendület Research Group on Cyber-Physical Systems
{szarnyas, varro}@mit.bme.hu
[3] McGill University
Department of Electrical and Computer Engineering

Abstract. Graph database systems are increasingly adapted for storing and processing heterogeneous network-like datasets. However, due to the novelty of such systems, no standard data model or query language has yet emerged. Consequently, migrating datasets or applications even between related technologies often requires a large amount of manual work or ad-hoc solutions, thus subjecting the users to the possibility of vendor lock-in. To avoid this threat, vendors are working on supporting existing standard languages (e.g. SQL) or standardising languages.
In this paper, we present a formal specification for openCypher, a high-level declarative graph query language with an ongoing standardisation effort. We introduce relational graph algebra, which extends relational operators by adapting graph-specific operators and define a mapping from core openCypher constructs to this algebra. We propose an algorithm that allows systematic compilation of openCypher queries.

1 Introduction

Context. Graphs are a well-known formalism, widely used for describing and analysing systems. Graphs provide an intuitive formalism for modelling many real-world scenarios, as the human mind tends to interpret the world in terms of objects (vertices) and their respective relationships to one another (edges) [15].

The property graph data model [17] extends graphs by adding labels/types and properties for vertices and edges. This gives a rich set of features for users to model their specific domain in a natural way. Graph databases are able to store property graphs and query their contents with complex graph patterns, which otherwise would be are cumbersome to define and/or inefficient to evaluate on traditional relational databases [21].

Neo4j[1], a popular NoSQL property graph database, offers the Cypher query language to specify graph patterns. Cypher is a high-level declarative query

[1] https://neo4j.com/

J. Marton, G. Szárnyas, D. Varró,
Formalising openCypher Graph Queries in Relational Algebra,
ADBIS, Springer, 2017

NESTED RELÁCIÓALGEBRA (NRA)

- 0NF adatszerkezetek, azaz NF²: Non-First Normal Form
- Operátorok
  - Nest (ν) ~ `collect`
  - Unnest (μ) ~ `UNWIND`

name	works	
	year	company
John	1982	Big Biz, Inc.
	2010	Fusion Power Plant, Ltd.

μ ν

name	works.year	works.company
John	1982	Big Biz, Inc.
John	2010	Fusion Power Plant, Ltd.

E. Botoeva, D. Calvanese, B. Cogrel, G. Xiao,
Expressivity and Complexity of MongoDB Queries,
ICDT 2018

TULAJDONSÁGGRÁFOK MINT NF² RELÁCIÓK

- Tulajdonságok:

id	name	age	favColours	beerRatings
1	John	32	[blue, green]	{lager: 5, ale: 3}

- Lista:

id	name	age	favColours	beerRatings
1	John	32	id | value 0 | blue 1 | green	{lager: 5, ale: 3}

- Map:

id	name	age	favColours	beerRatings
1	John	32	id | value 0 | blue 1 | green	key | value lager | 5 ale | 3

NRA -> FRA*

- NRA kifejezések kiteregethetők Flat Relational Algebrába
- Papíron…
  - Előre ismerni kell a sémát
  - A rendezés megtartása problémás

Converting Nested Algebra Expressions into Flat Algebra Expressions

JAN PAREDAENS
University of Antwerp
and
DIRK VAN GUCHT
Indiana University

1. INTRODUCTION

J. Paredaens, D. Van Gucht:
Converting nested algebra expressions into flat algebra expressions.
ACM Transactions on Database Systems, 1992

OPENCYPHER -> RELÁCIÓALGEBRA LEKÉPZÉS*

Result and subresult operations. Rules for `RETURN` also apply to `WITH`.		
⟦r⟧ `RETURN «x1» AS «y1», ⋯`	$\pi_{x1 \to y1, \dots}(r)$	(15)
⟦r⟧ `RETURN DISTINCT «x1» AS «y1», ⋯`	$\delta\left(\pi_{x1 \to y1, \dots}(r)\right)$	(16)
⟦r⟧ `RETURN «x1», «aggr»(«x2»)`	$\gamma^{x1}_{x1, aggr(x2)}(r)$ (see Sec. 3.1)	(17)
⟦r⟧ `WITH «x1»` ⟦s⟧ `RETURN «x2»`	$\pi_{x2}\left(\left(\pi_{x1}(r)\right) \bowtie s\right)$	(18)
Unwinding and list operations		
⟦r⟧ `UNWIND «xs» AS «x»`	$\omega_{xs \to x}(r)$	(19)
⟦r⟧ `ORDER BY «x1» ASC, «x2» DESC, ⋯`	$\tau_{\uparrow x1, \downarrow x2, \dots}(r)$	(20)
⟦r⟧ `SKIP «s» LIMIT «l»`	$\lambda^{s}_{l}(r)$	(21)
Combining results		
⟦r⟧ `UNION` ⟦s⟧	$r \cup s$	(22)
⟦r⟧ `UNION ALL` ⟦s⟧	$r \uplus s$	(23)

Table 2: Mapping from openCypher constructs to relational algebra. Variables, labels, types and literals are typeset as «v». The notation (p) represents patterns resulting in a relation p, while ⟦r⟧ denotes previous query parts resulting in a relation r. To avoid confusion with the ". ." language construct (used for ranges), we use ⋯ to denote omitted query parts.

Language construct	Relational algebra expression	
Vertices and patterns. (p) denotes a pattern that contains a vertex «v».		
(«v»)	$\bigcirc_{(\mathsf{v})}$	①
(«v»:«l1»:⋯:«ln»)	$\bigcirc_{(\mathsf{v}:\ \mathsf{l1}\wedge\cdots\wedge\mathsf{ln})}$	②
(p)-[«e»:«t1»|⋯|«tk»]->(«w»)	$\uparrow \begin{smallmatrix}(\mathsf{w})\\(\mathsf{v})\end{smallmatrix} [\mathsf{e}:\ \mathsf{t1}\vee\cdots\vee\mathsf{tk}]\,(p)$, where e is an edge	③
(p)<-[«e»:«t1»|⋯|«tk»]-(«w»)	$\downarrow \begin{smallmatrix}(\mathsf{w})\\(\mathsf{v})\end{smallmatrix} [\mathsf{e}:\ \mathsf{t1}\vee\cdots\vee\mathsf{tk}]\,(p)$, where e is an edge	④
(p)<-[«e»:«t1»|⋯|«tk»]->(«w»)	$\updownarrow \begin{smallmatrix}(\mathsf{w})\\(\mathsf{v})\end{smallmatrix} [\mathsf{e}:\ \mathsf{t1}\vee\cdots\vee\mathsf{tk}]\,(p)$, where e is an edge	⑤
(p)-[«e»*«min»..«max»]->(«w»)	$\uparrow \begin{smallmatrix}(\mathsf{w})\\(\mathsf{v})\end{smallmatrix} [\mathsf{e}*^{\mathsf{max}}_{\mathsf{min}}]\,(p)$, where e is a list of edges	⑥
Combining and filtering pattern matches		
MATCH (p1), (p2), ⋯	$\not\equiv_{\text{edges of p1, p2, ⋯}}\,(p1 \bowtie p2 \bowtie \cdots)$	⑦
MATCH (p1) MATCH (p2)	$\not\equiv_{\text{edges of p1}}\,(p1)\ \bowtie\ \not\equiv_{\text{edges of p2}}\,(p2)$	⑧
OPTIONAL MATCH (p)	$\{\langle\rangle\} \ltimes \not\equiv_{\text{edges of p}}\,(p)$	⑨
OPTIONAL MATCH (p) WHERE (condition)	$\{\langle\rangle\} \ltimes_{\text{condition}} \not\equiv_{\text{edges of p}}\,(p)$	⑩
[r] OPTIONAL MATCH (p)	$\not\equiv_{\text{edges of r}}\,(r) \ltimes \not\equiv_{\text{edges of p}}\,(p)$	⑪
[r] WHERE «condition»	$\sigma_{\text{condition}}(r)$	⑫
[r] WHERE («v»:«l1»:⋯:«ln»)	$\sigma_{v.l=\mathsf{l1}\wedge\cdots\wedge n.l=\mathsf{ln}}(r)$	⑬
[r] WHERE (p)	$r \bowtie p$	⑭

Gráfanalitika

GRÁFANALITIKA / LDBC GRAPHALYTICS

6 népszerű algoritmus:

- BFS mélységi bejárás
- CDLP közösségdetekció
- LCC lokális klaszterezettségi együttható
- PR PageRank
- SSSP egy forrásból számított legrövidebb utak
- WCC gyengén összefüggő komponensek (v.ö. SCC)

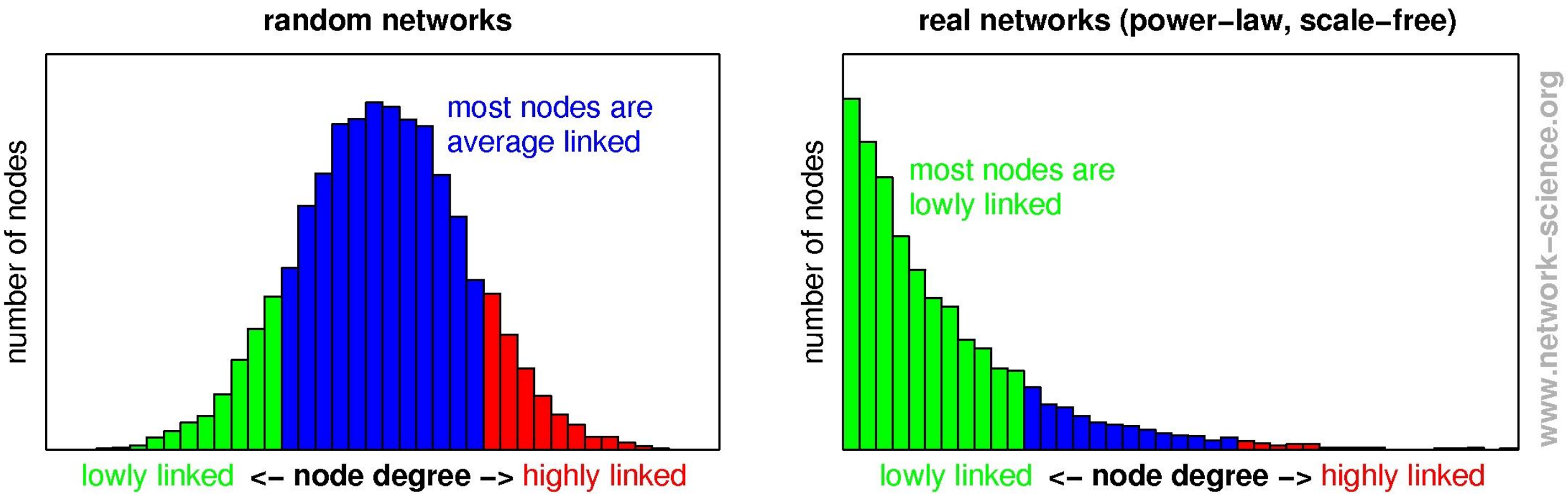
Hatékony megoldásokat adni nehéz:

algoritmuselmélet + adatbázis-kezelés + system programming és High-Performance Computing (HPC) technikák

GRÁFANALITIKA

- Tipikus megoldás programozási modellek alkalmazása
  - Mint máshol a MapReduce
  - Pregel (“a portmanteu of Parallel, Graph, and Google”)
  - Scatter-Gather
  - Gather-Apply-Scatter
- Megkötik a programozó kezét
- Cserébe
  - párhuzamosíthatók
  - elosztottan futtathatók

HÁLÓZATOK STRUKTÚRÁJA



- Fokszámeloszlások vizsgálata
- Hálózatkutatás (network science)
- Barabási-Albert László et al.

Gráffeldolgozó eszközök és kihívásaik

GRÁFFELDOLGOZÓ ESZKÖZÖK

Jelenleg éles kettéválás.

gráfminta-illesztés

neo4j

JanusGraph

MEM GRAPH

Gremlin

$G = (V, E)$

gráfanalitika

APACHE Spark™

GraphX

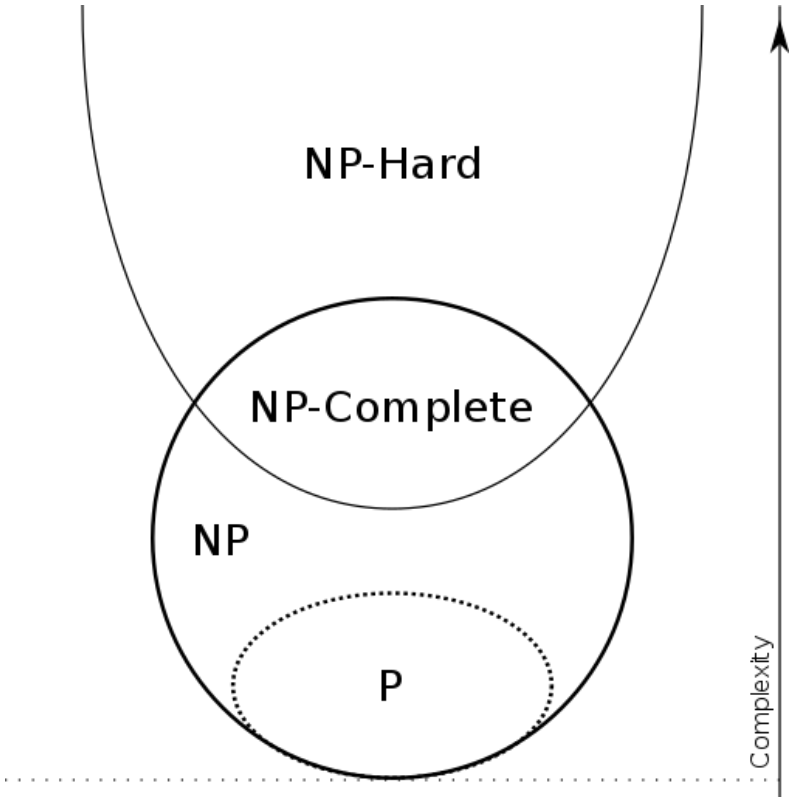
APACHE GIRAPH

GraphLab

Flink

GRÁFMŰVELETEK KOMPLEXITÁSA

- RÉSZGRÁFIZO: NP-teljes
- GRÁFIZO: nem ismert
  - NP-nehéz?
  - P-beli?
  - NP-Intermediate?
- Egy konkrét részgráf keresése -> P-beli



ha P≠NP

- Legrövidebb utak keresése -> P-beli
- Tranzitív lezárás / összes út felsorolása:
  - *intractable* (polinom idő alatt nem megoldható)

Ábra forrása: Wikipedia

GRÁFFELDOLGOZÁS KIHÍVÁSAI / TOPOLÓGIA / 1

- […] large graph processing has some unique characteristics, which make the systems that do not respect them in their design suffer from the “curse of connectedness” when processing big graphs.
- Graph data is inherently complex. The contemporary computer architectures are good at processing linear and simple hierarchical data structures, such as *Lists*, *Stacks*, or *Trees*.


Trinity Graph Engine and its Applications

Bin Shao, Yatao Li, Haixun Wang† Haozhou Xia
Microsoft Research Asia, †Facebook
{binshao, yatli, haixy}@microsoft.com, †haixun@gmail.com

Abstract

1 Introduction

The complex nature of graph. Graph data is inherently complex. The contemporary computer architectures are good at processing linear and simple hierarchical data structures, such as Lists, Stacks, or Trees. Even when the data scale becomes large and is partitioned over many distributed machines, the divide and conquer

Copyright 2017 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

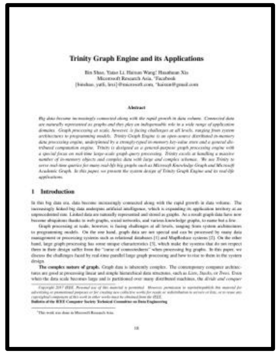
† This work was done in Microsoft Research Asia.

1

B. Shao, Y. Li, H. Wang, H. Xia
(Microsoft Research Asia),
Trinity Graph Engine and its Applications,
IEEE Data Engineering Bulleting 2017

GRÁFFELDOLGOZÁS KIHÍVÁSAI / TOPOLÓGIA / 2

- […] the adjacent nodes of a graph node cannot be accessed without "jumping" in the data store no matter how we represent a graph […] a massive amount of random data access is required.
- Many modern program optimizations rely on data reuse. Unfortunately, the random data access nature of graph processing breaks this premise. […], this usually leads to poor performance since the CPU cache is not in effect for most of the time.
- From the perspective of programming, parallelism is difficult to extract because of the unstructured nature of graphs


Trinity Graph Engine and its Applications

B. Shao, Y. Li, H. Wang, H. Xia
(Microsoft Research Asia),
Trinity Graph Engine and its Applications,
IEEE Data Engineering Bulleting 2017

GRÁFFELDOLGOZÁS KIHÍVÁSAI / ATTRIBUTÚMOK

- The existing graph querying methods […] focus on querying the topological structure of the graphs and very few have considered attributed graphs.
- In practice, […] applications of large graph databases would involve querying the graph data (attributes) in addition to the graph topology.
- Answering queries that involve predicates on the attributes of the graphs in addition to the topological structure is more challenging as it requires extra memory for building indices over the graph attributes in addition to the structural indices [and] it makes evaluation and optimization more complex.

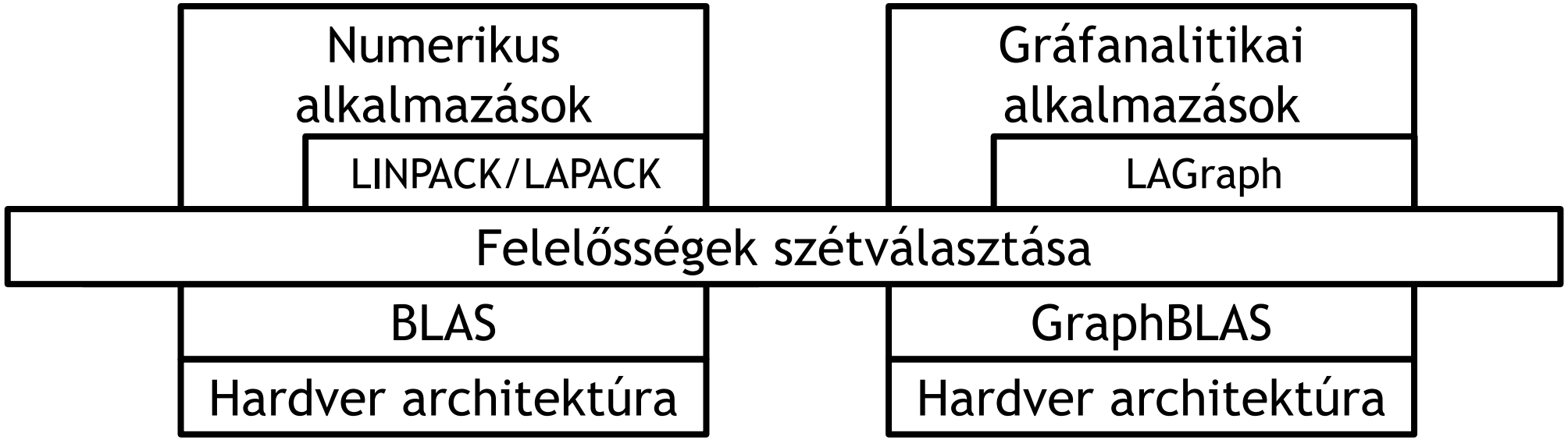

G-SPARQL: A Hybrid Engine for Querying Large Attributed Graphs

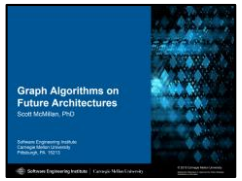
S. Sakr, S. Elnikety, Y. He
(Microsoft Research),
G-SPARQL: A Hybrid Engine for Querying Large Attributed Graphs,
CIKM 2012

GraphBLAS: egységes elmélet a lineáris algebrára építve

A GRAPHBLAS MEGKÖZELÍTÉS

- GraphBLAS: „an effort to define standard building blocks for graph algorithms in the language of linear algebra”
- 1979: BLAS (Basic Linear Algebra Subprograms)
- 2013: GraphBLAS
- Absztrakció, felelősségek szétválasztása (*separation of concerns*)

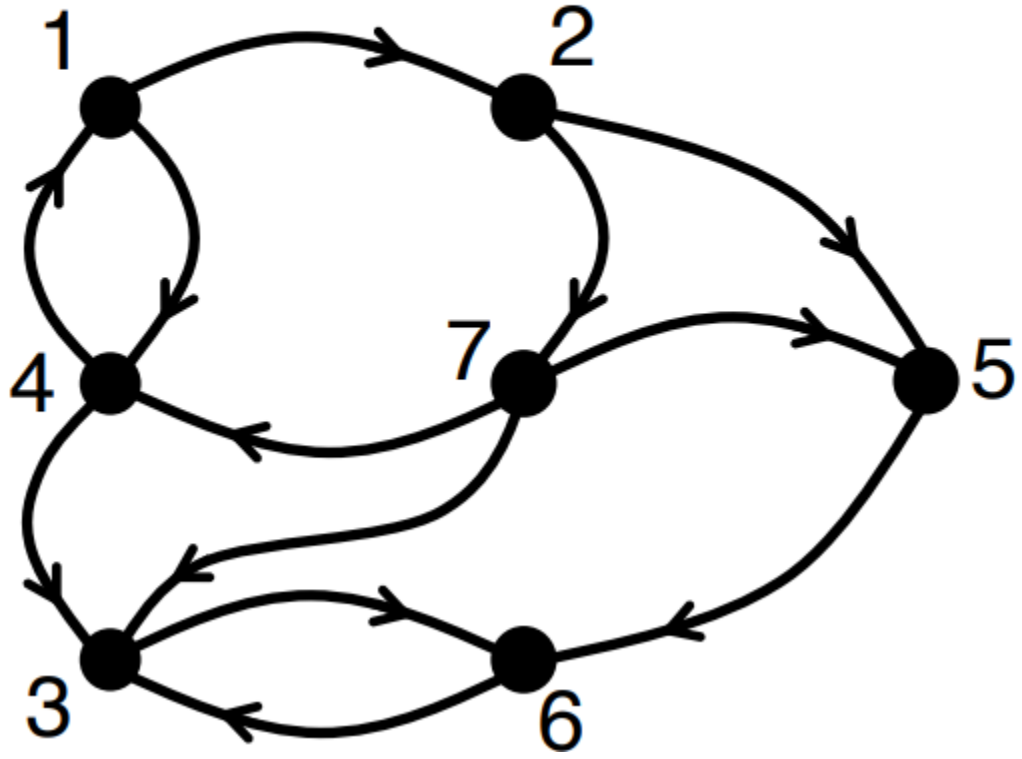


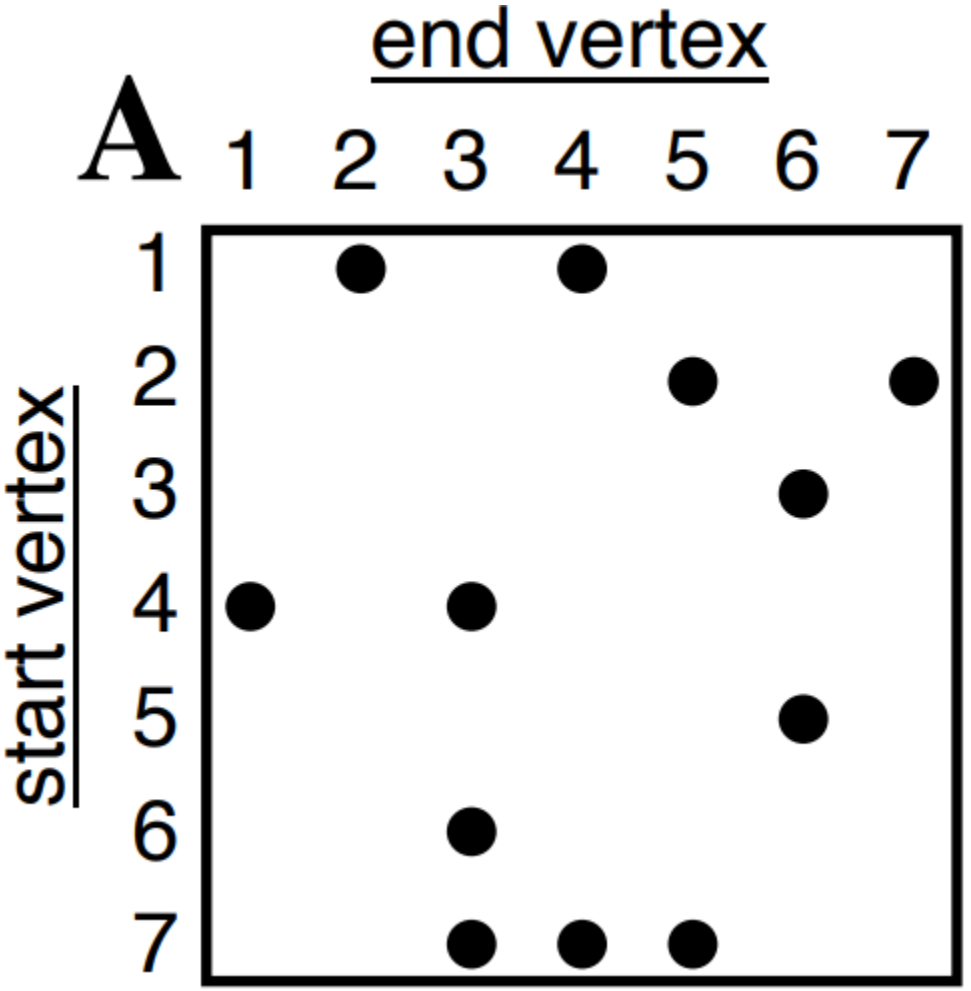


S. McMillan @ SEI Research Review (Carneghie Mellon University, 2015)
Graph algorithms on future architectures

SZOMSZÉDOSSÁGI MÁTRIX

Forrás:
GraphBLAS Mathematics
v1.0, 2017

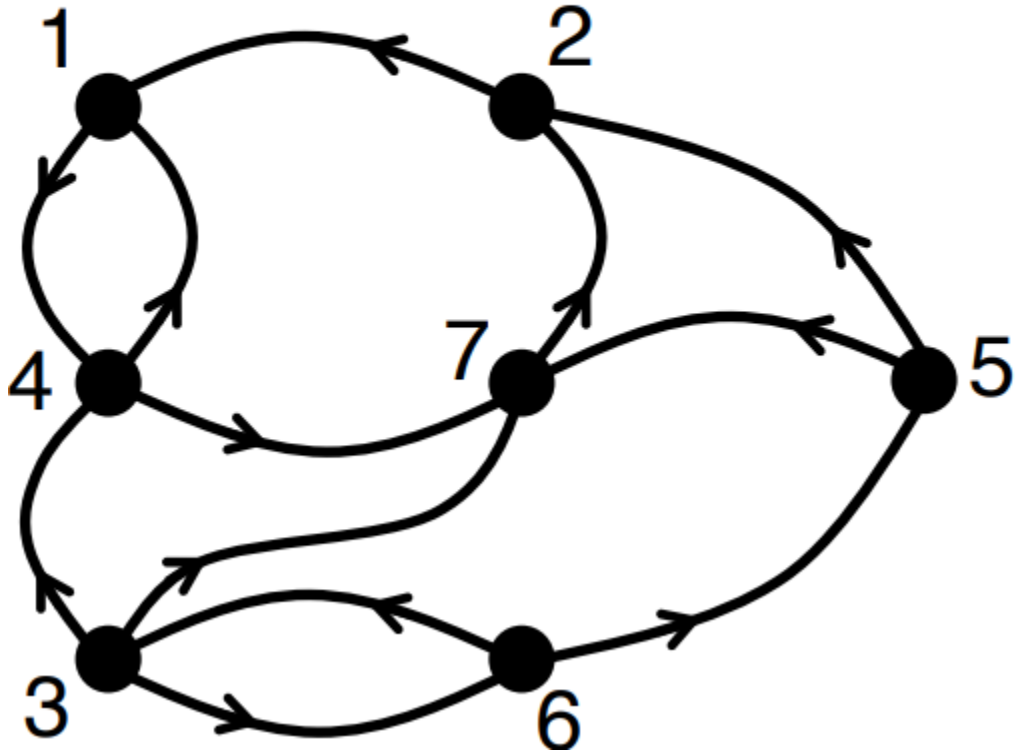


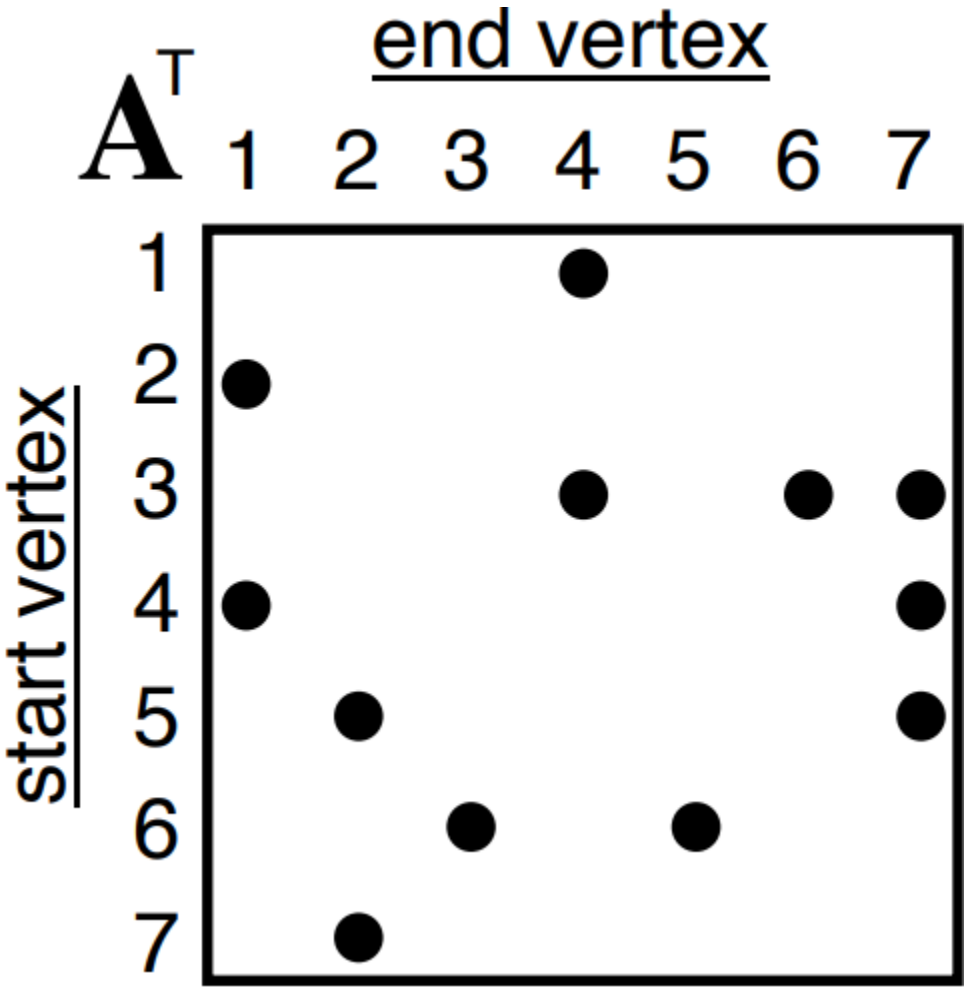


A gyakorlatban ritka a legtöbb elem 0 -> a mátrixok nagyon ritkák.

SZOMSZÉDOSSÁGI MÁTRIX: A^T

Forrás:
GraphBLAS Mathematics
v1.0, 2017





A gyakorlatban ritka a legtöbb elem 0 -> a mátrixok nagyon ritkák.

MÁTRIXSZORZÁS

- A szokásos alak:

$$C = A \times B$$

$$C(i,j) = \sum_k A(i,k) \times B(k,j)$$

- Általánosított alak:

$$C = A \oplus . \otimes B$$

$$C(i,j) = \oplus_k \ A(i,k) \otimes B(k,j)$$

- $\oplus$ kiválasztás (*summary*)
- $\otimes$ kiterjesztés (*extension*)
- Keressünk olyan algebrai struktúrát, amin a mátrixszorzás értelmezhető -> félgyűrű (*semiring*)

GYŰRŰ (*RING*), DEF1

$\langle S, \oplus, \otimes \rangle$ algebrai struktúra *gyűrű*, ha $\forall a, b, c \in S$-re

- $\langle S, \oplus \rangle$ Abel-csoport
  - Kommutatív $a \oplus b = b \oplus a$
  - Asszociatív $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Van egységeleme $a \oplus 0 = a$ (nullelem)
  - Van inverze $a \oplus (-a) = 0$
- $\langle S, \otimes \rangle$ félcsoport
  - Asszociatív $(a \otimes b) \otimes c = a \otimes (b \otimes c)$

- Teljesül a disztributivitás:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
  - $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$

GYŰRŰ (*RING*), DEF2

$\langle S, \oplus, \otimes \rangle$ algebrai struktúra egységelemes *gyűrű*, ha $\forall a, b, c \in S$-re

- $\langle S, \oplus \rangle$ Abel-csoport
  - Kommutatív $a \oplus b = b \oplus a$
  - Asszociatív $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Van egységeleme $a \oplus 0 = a$ (nullelem)
  - Van inverze $a \oplus (-a) = 0$
- $\langle S, \otimes \rangle$ egységelemes félcsoport (=monoid)
  - Asszociatív $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - Van egységeleme $a \otimes 1 = a$
- Teljesül a disztributivitás:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
  - $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$

FÉLGYŰRŰ (*SEMIRING*)

$\langle S, \oplus, \otimes \rangle$ algebrai struktúra *félgyűrű*, ha $\forall a, b, c \in S$-re

- $\langle S, \oplus \rangle$ kommutatív monoid
  - Kommutatív $a \oplus b = b \oplus a$
  - Asszociatív $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Van egységeleme $a \oplus 0 = a$ (nullelem)
  - ~~Van inverze $a \oplus (-a) = 0$~~
- $\langle S, \otimes \rangle$ monoid
  - Asszociatív $(a \otimes b) \otimes c = a \otimes (b \otimes c)$
  - Van egységeleme $a \otimes 1 = a$
- Teljesül a disztributivitás:
  - $a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes c)$
  - $(a \oplus b) \otimes c = (a \otimes c) \oplus (b \otimes c)$

FÉLGYŰRŰK

	alaphalmaz	$\oplus$	nullelem	$\otimes$	egységelem
valós számok	$a \in \mathbb{R}$	$+$	0	$\cdot$	1
max-plus	$a \in \mathbb{R} \cup \{-\infty\}$	max	$-\infty$	$+$	0
min-plus	$a \in \mathbb{R} \cup \{+\infty\}$	min	$+\infty$	$+$	0
lor-land	$a \in \{F, T\}$	$\vee$	F	$\wedge$	T
hatványhalmaz	$a \subset \mathbb{Z}$	$\cup$	$\emptyset$	$\cap$	$\emptyset$
Galois mező	$a \in \{0,1\}$	xor	0	$\wedge$	0

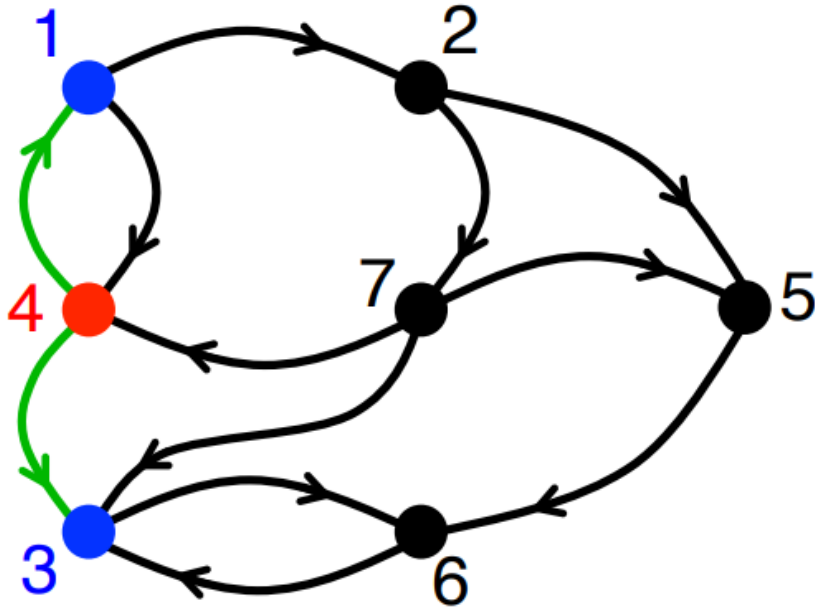
A max-plus/min-plus esetek *trópusi félgyűrűk* néven is ismertek. Ezek a trópusi geometria (tropical geometry) területén használatosak, melyet Simon Imre magyar származású Brazíliában alkotó matematikus nyomán neveztek el.

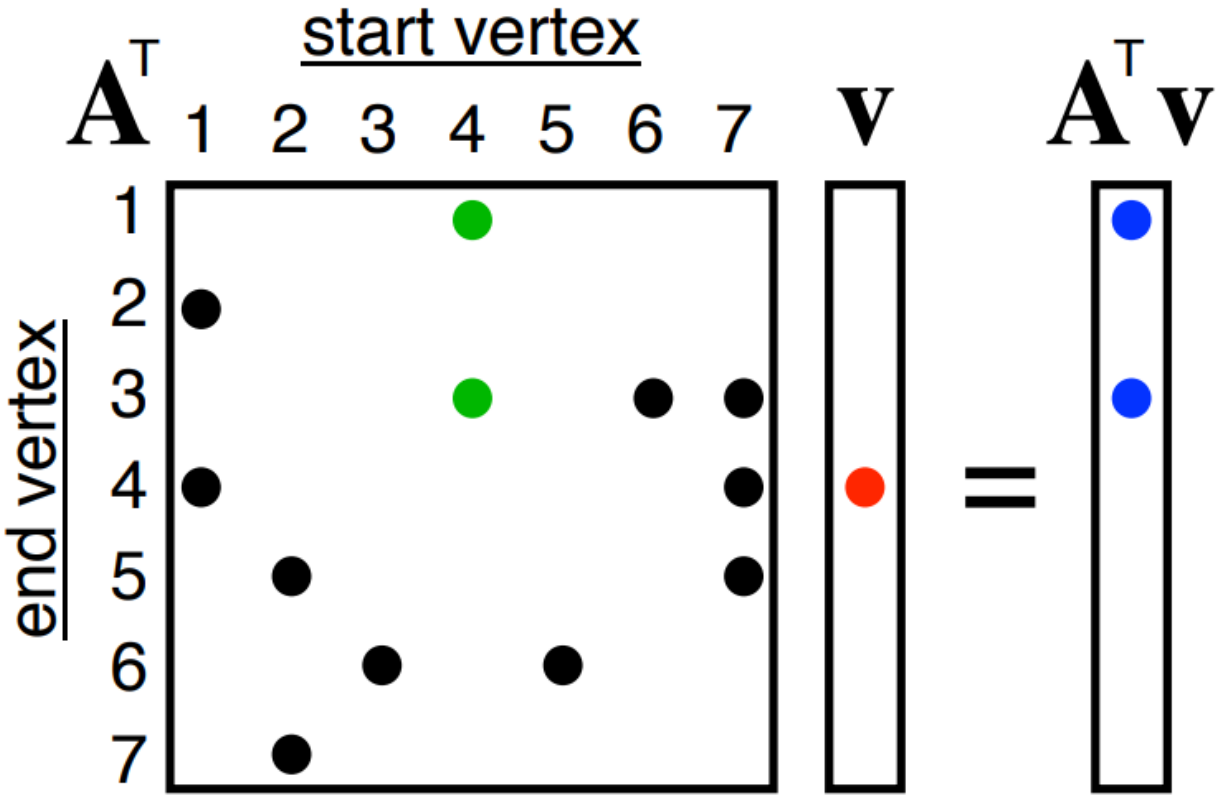
Gráfbejáró algoritmusok GraphBLAS-ban

BFS: BREADTH-FIRST SEARCH*

Forrás:
GraphBLAS Mathematics
v1.0, 2017

- Breadth-first search (BFS)





- Részletesebben a RedisGraph-ról szóló előadás 14-28. diáin.
- Megjegyzés: a BFS-t könnyű, a DFS-t nehéz párhuzamosítani.

SSSP: SINGLE SOURCE SHORTEST PATH*

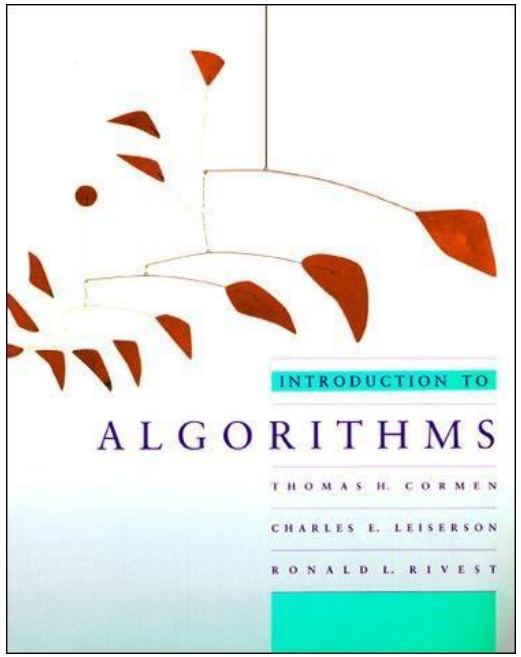
- Két pont közötti legrövidebb út
  - a következő problémával megegyező nehézségű
- Adott pontból vezető összes legrövidebb út
  - Dijkstra algoritmus $\mathcal{O}(n + e)$
  - Bellman-Ford algoritmus $\mathcal{O}(n \cdot e)$
    - negatív élsúlyokat is kezel
    - min.+ félgyűrű felett felírható ->

$$
\begin{array}{l}
\mathbf{1}\ d \leftarrow [\infty\ \infty\ \ldots\ \infty]^{\mathsf{T}} \\
\mathbf{2}\ d(s) \leftarrow 0 \\
\mathbf{3}\ \textbf{for}\ k \leftarrow 1\ \textbf{to}\ N-1\ \textbf{do} \\
\mathbf{4}\ \ \ d^{\mathsf{T}} \leftarrow d^{\mathsf{T}}\ \text{min.}+A \\
\mathbf{5}\ \textbf{end} \\
\mathbf{6}\ \textbf{return}\ d
\end{array}
$$

- Bármely két pont közötti legrövidebb út
  - Floyd algoritmus $\mathcal{O}(n^3)$
    - a kimenete egy sűrű mátrix $\mathcal{O}(n^2)$ nem-nulla elemmel

Kapcsolódó irodalom

CORMEN-LEISERSON-RIVEST ALGORITMUSOK KÖNYV



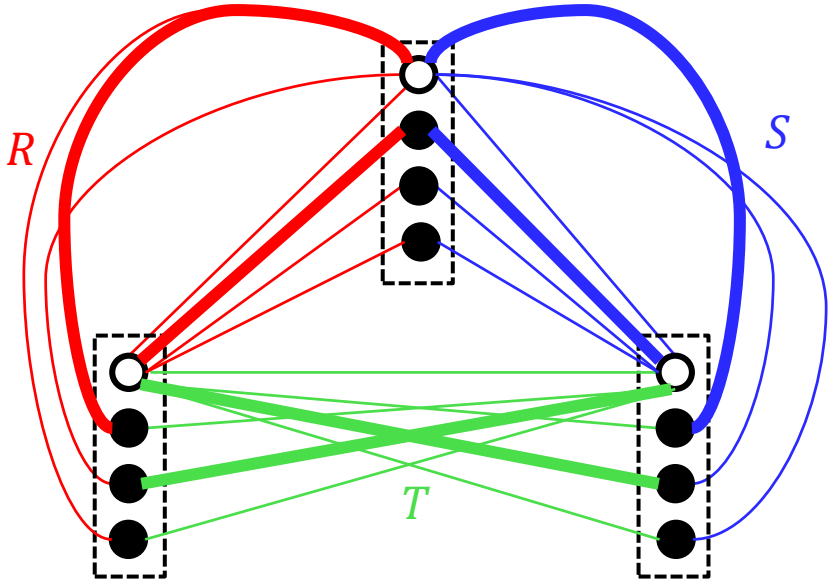
***26.4. Zárt félgyűrűk: az irányított utakkal kapcsolatos problémák algebrai szerkezete**

[1]A zárt félgyűrűkön alapuló algoritmusok, így a Floyd–Warshall és a tranzitív lezárás algoritmus jelentősége, hogy ***párhuzamos számítási hálózatokon*** alkalmazhatók. Leighton [135] kimerítően tárgyalja a párhuzamos számítási hálózatokon működő gráfalgoritmusokat és bemutatja, hogy a zárt félgyűrű ÚT-KIVÁLASZTÁS algoritmusa a Gauss-elimináció lépéseihez hasonlít. *A fordító.*

Párhuzamos számítási hálózatok -> ma ilyenek a GPU-k

- 1997: a fenti lábjegyzet
- 1999: Nvidia GeForce 256
- 2001: első GPU-alapú mátrixszorzó algoritmus (GPGPU)

„SKEW” PROBLÉMA BINÁRIS ILLESZTÉSEKKEL*

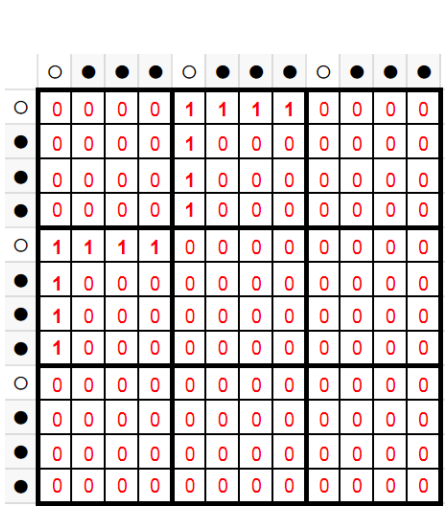


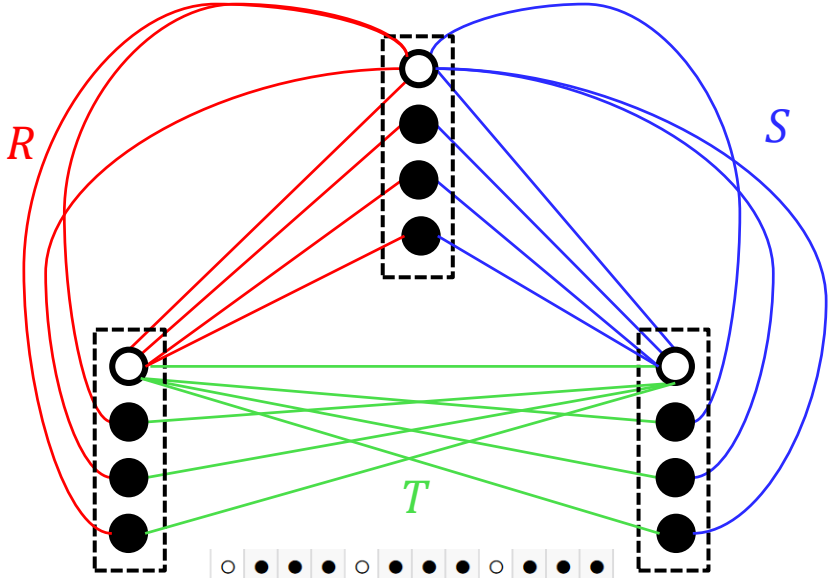
Soroljuk fel mindegyik háromszöget: $R \bowtie S \bowtie T$

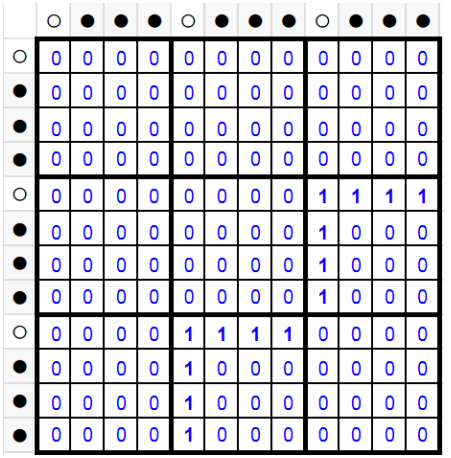
A *bináris természetes illesztést* használó megoldásoknak legalább $\mathcal{O}(n^2)$ lépésre van szüksége, miközben az elméleti alsó korlát $\mathcal{O}(n^{1.5})$.

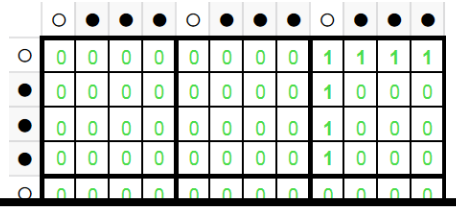
H.Q. Ngo et al. @ SIGMOD Record 2013
Skew strikes back: new developments in the theory of join algorithms.

„SKEW” MÁTRIXOKON*



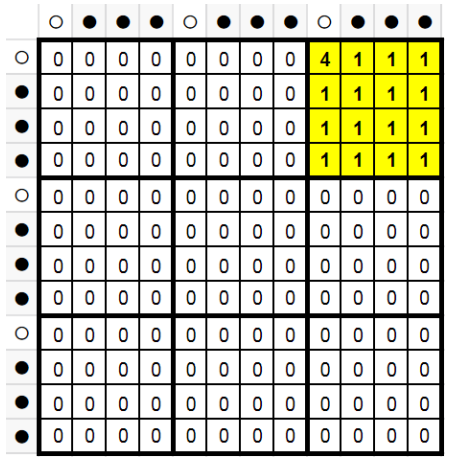






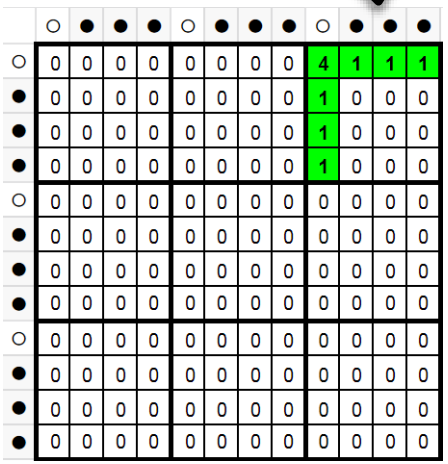
maszkolás

$R \cdot S =$



19 ék
$\mathcal{O}(n^2)$

$R \cdot S \cdot T =$



10 háromszög
$\mathcal{O}(n)$

„SKEW” AZ ADATELOSZLÁSBAN*

- **A worst-case optimal join algoritmusok** képesek multiway join művelettel $\mathcal{O}(n^{1.5})$ alatt kiszámítani az eredményt: $\bowtie (R, S, T)$
- **Előfordul-e ez a gyakorlatban?** Igen! Skálafüggetlen hálózatok.
- A probléma jól ismert a gráfanalitikában is, ezért van a GraphBLAS-ban API szinten maszkolt mátrix szorzás.

H.Q. Ngo @ Journal of the ACM 2018
Worst-case optimal join algorithms

T.M. Low, S. McMillan et al. @ HPEC 2017
First look: linear algebra-based triangle counting without matrix multiplication

Összefoglalás

KIHÍVÁSOK A GRÁFFELDOLGOZÁSBAN

- Véletlen hozzáférés (random access)
- NF^2 adatszerkezet
  - Tulajdonságok
  - Kollekciók: lista, map (dictionary)
- Halmaz/multihalmaz/lista szemantikájú eredményhalmazok
- Anti/semi/outer join műveletek kezelése
- Lekérdezések és CUD műveletek keveredése
- Analitikai és lekérdezés terhelési profilok keveredése
- Vizualizáció (!)
- Friss terület

GRÁFLEKÉRDEZŐNYELVEK ÉS FORMALIZÁLÁSUK


Formalising openCypher Graph Queries in Relational Algebra

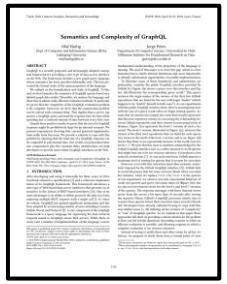
J. Marton, G. Szárnyas, D. Varró,
Formalising openCypher Graph Queries in Relational Algebra,
ADBIS 2017


Cypher: An Evolving Query Language for Property Graphs

N. Francis et al.,
Cypher: An Evolving Query Language for Property Graphs,
SIGMOD 2018


G-CORE
A Core for Future Graph Query Languages

R. Angles et al.,
G-CORE: A Core for Future Graph Query Languages,
SIGMOD 2018


Semantics and Complexity of GraphQL

O. Hartig, J. Pérez,
Semantics and Complexity of GraphQL,
WWW 2018