IT4BI

eBISS Summer School 2017

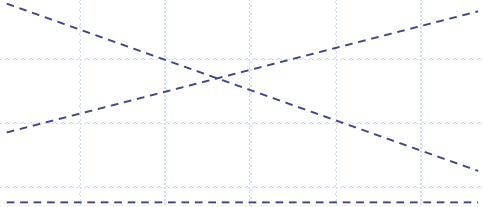
Active Business Intelligence – Compact and Efficient Query Processing under Updates

Presenter: Muhammad Idris

Advisors:

Prof. Stijn Vansummeren

Prof. Wolfgang Lehner



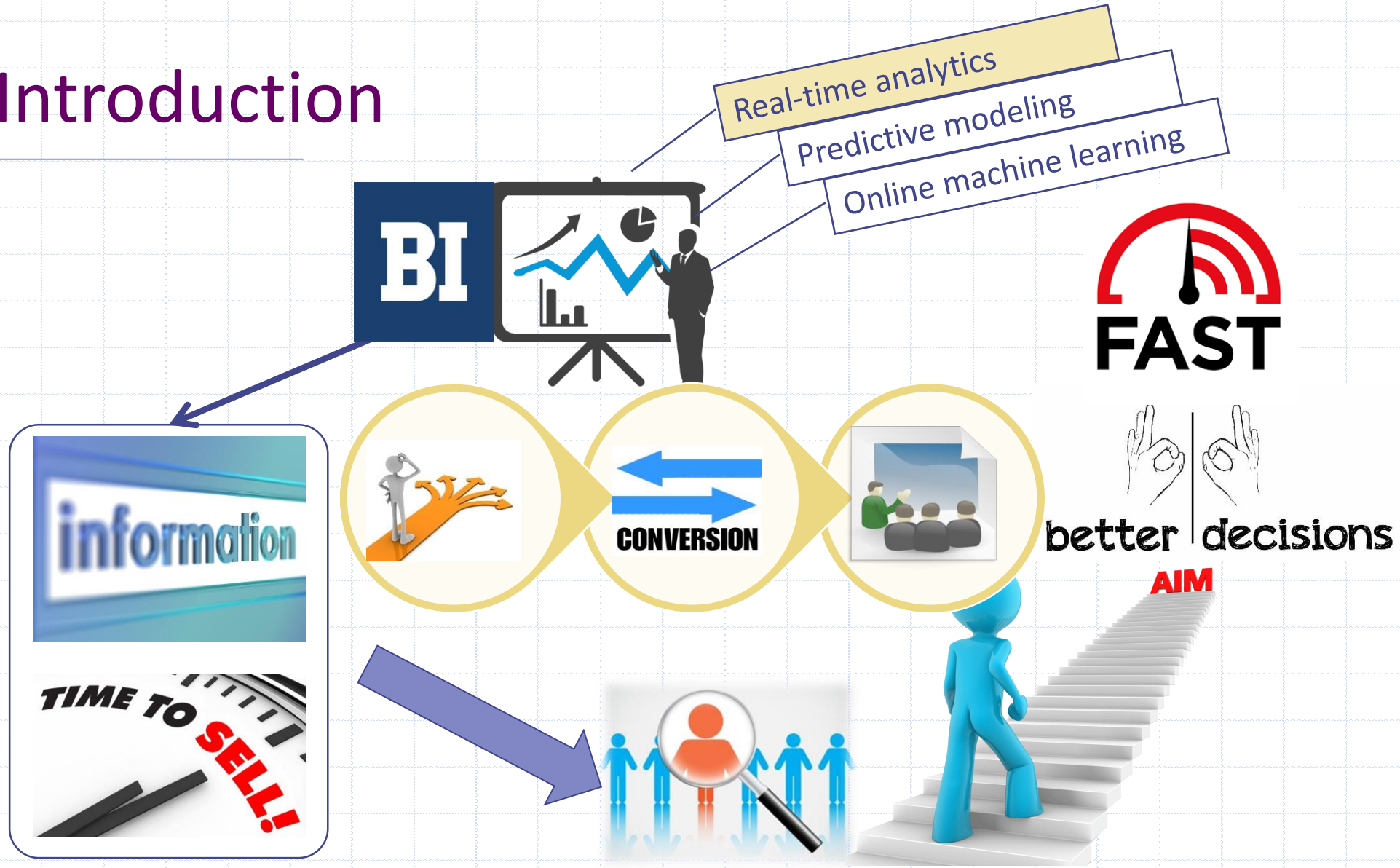
ULB UNIVERSITÉ LIBRE DE BRUXELLES

TECHNISCHE UNIVERSITÄT DRESDEN

Thursday, July 6, 2017

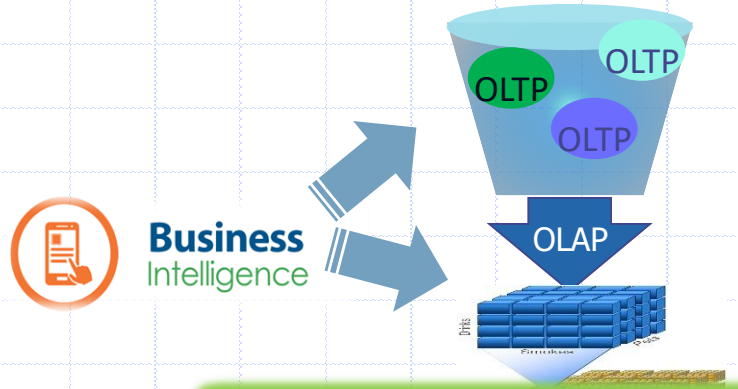
ULB UNIVERSITÉ LIBRE DE BRUXELLES

Introduction



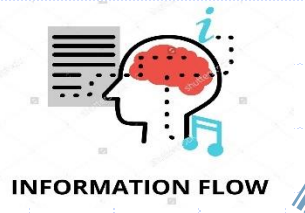
Motivation

Real-time Analytics



Business Intelligence

- Operational data
- Reacting to updates



Continuous Query

Complex Event Processing (CEP)

Unified Model

- Data model
- Query language
- Processing model

Dynamic Query Evaluation

- Dynamic query evaluation
  - Given a database db and an update u:
    - Compute $Q(db + u)$
    - $Q(db)$ is already computed (**Materialized**)

Materialized Views

Applications

Compute $Q(DB)$

Read-only view

Incremental View Maintenance

Source data

OrderID	CustomerID
1	2
2	3

PartID	Name	Stock
1	Front door	30
2	Back Glass	10

OrderID	PartID	Qty
1	1	2
2	2	1
1	1	2

1	Front door	4
2	Back Glass	2

Incremental view maintenance (IVM)

Query: $Q = R(A,B) \bowtie S(B,C) \bowtie T(C,D)$

D =

A	B
1	5

B	C
5	2

C	D
2	4

u_R =(A:2,B:5) $\bowtie$ | B | C | $\bowtie$ | C | D | = | 2 | 5 | 2 | 4 |

A	B
1	5

$\bowtie$ u_S =(B:3,C:2) $\bowtie$

C	D
2	4

A	B
1	5

$\bowtie$ | B | C | $\bowtie$ u_T =(C:2,D:3)

Q(D)

A	B	C	D
1	5	2	4

+

A	B	C	D
2	5	2	4

=

$Q(D + \Delta D)$

A	B	C	D
1	5	2	4
2	5	2	4

Higher order IVM (DBToaster)

Query: $Q = R(A,B) \bowtie S(B,C) \bowtie T(C,D)$

D =

A	B
1	5

B	C
5	2

C	D
2	4

Q(D)

A	B	C	D
1	5	2	4

=

$Q(D + \Delta D)$

A	B	C	D
1	5	2	4
2	5	2	4

Can we do better?

u_R =(A:2,B:5)

$\bowtie$ $\Delta_R Q$ =

2	5	2	4

$\bowtie$ S $\Rightarrow$ $\Delta_T Q + (u_R \bowtie S)$ =

A	B	C
1	5	2
2	5	2

$\Delta_T Q$

Ahmad, Yanif, et al. "Dbtoaster: Higher-order delta processing for dynamic, frequently fresh views." *Proceedings of the VLDB Endowment* 5.10 (2012): 968-979.

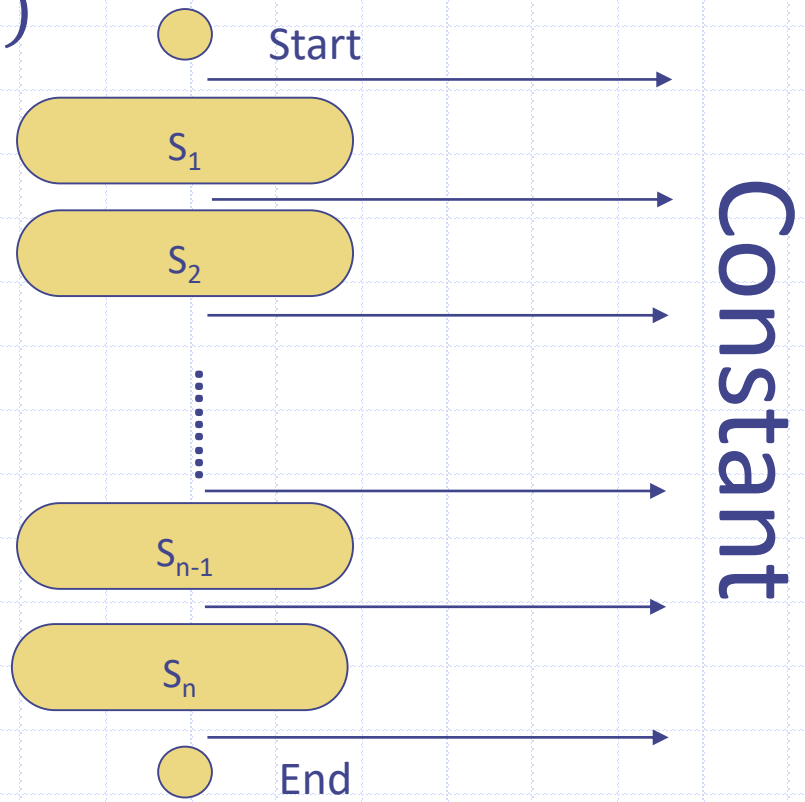
What we would like to?

- Desirable approach to:
  - View Enumeration: To have **constant delay(ideally)** – without materialization
  - Maintain in-memory data structure D- ***linear space*** *in the size of database db*
  - D is **maintainable** under updates easily (**constant time complexity- ideally**)
  - D provides constant time lookups.

Constant delay enumeration

- Enumerate S with constant delay from a data structure D

$Enumerat(D)$



$$S = \{S_1, S_2, \ldots, S_n\}$$

Research questions

Query Classes?

How close to these goals, if not achievable?

Data structures and algorithms?

Does this work for IFP and BI both?

Queries

- **Aggregation over Acyclic Conjunctive Queries(ACQ)**
  - **SELECT** sum(A), B

    FROM R,S,T

    WHERE R.B = S.B **AND** S.C = T.C

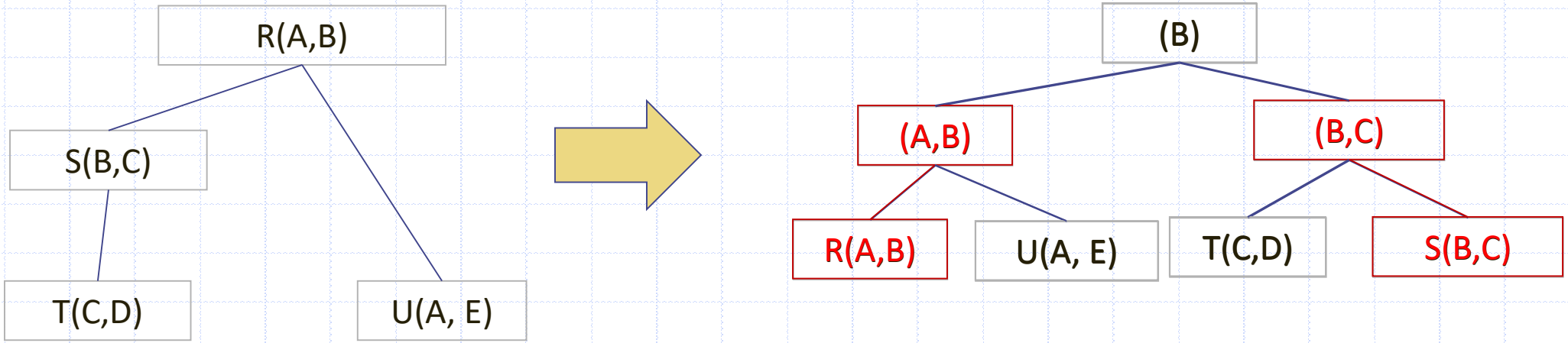
  - $\Sigma_A(\pi_{A,B}(R(A,B) \bowtie S(B,C) \bowtie T(C,D)))$

ACQ – Generalized Join Tree(GJT)

- A CQ is acyclic if its has a generalized join tree

Internal nodes have *guards*

Each variable must *induce* a subtree

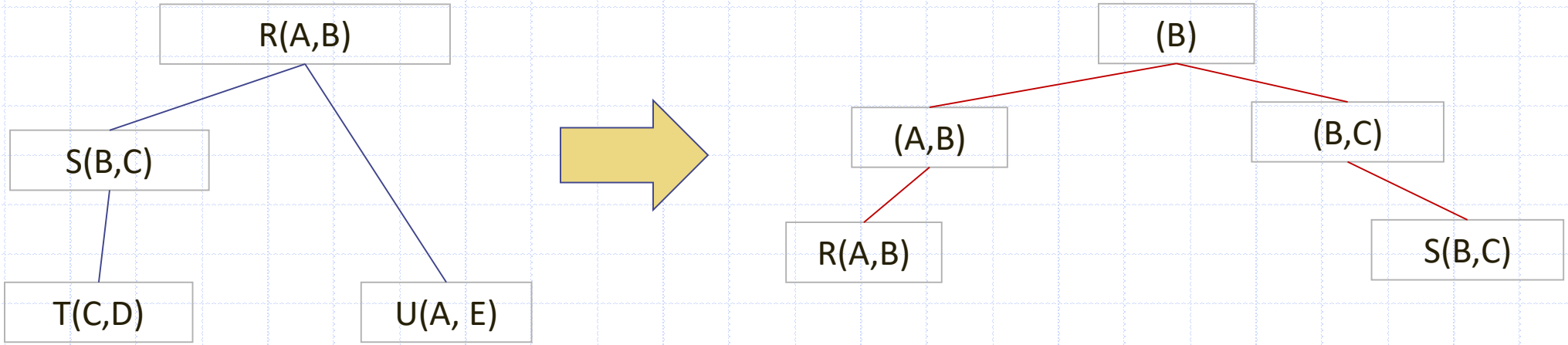


ACQ – Generalized Join Tree

- A CQ is acyclic if its has a generalized join tree

Internal nodes have *guards*

Each variable must *induce* a subtree



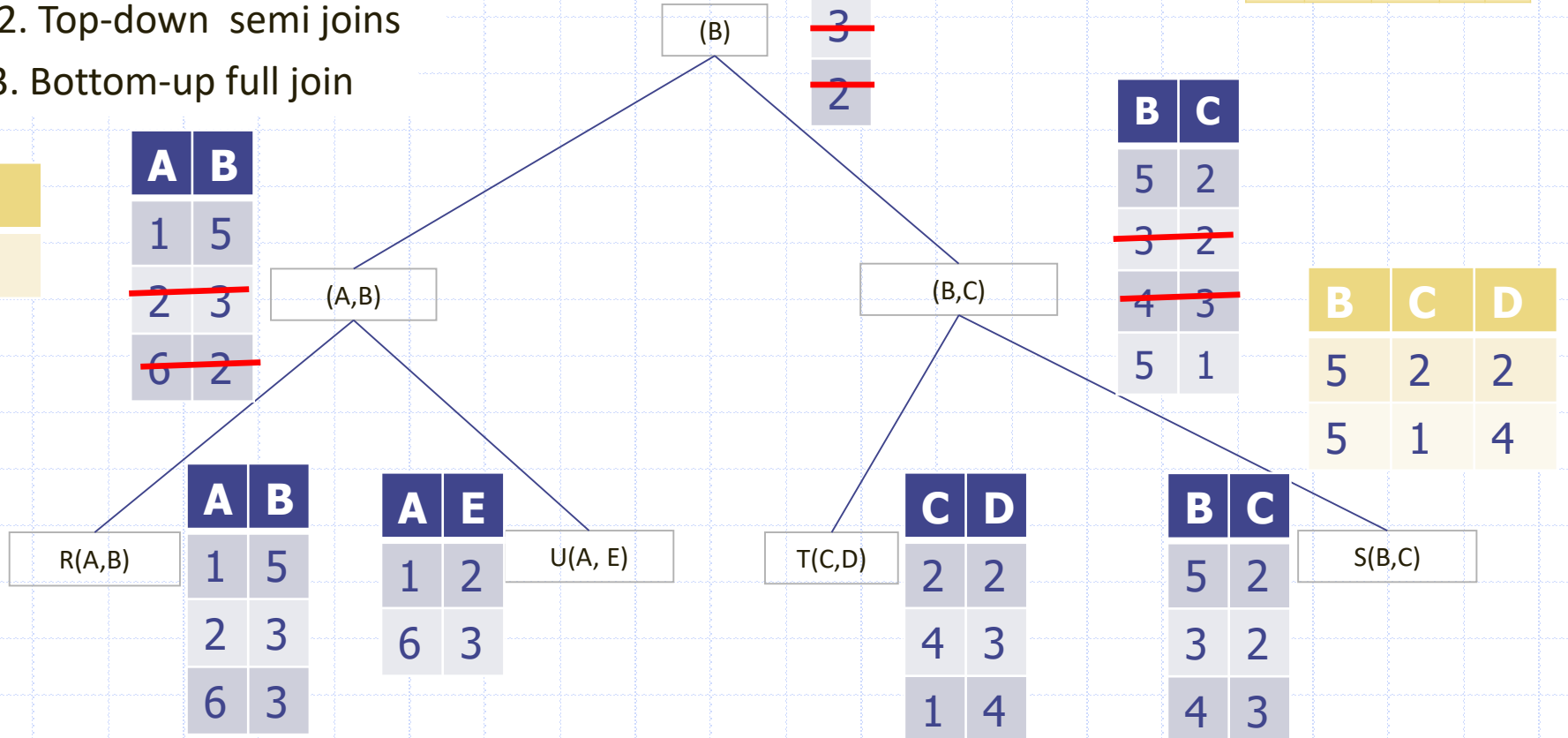
Yannakakis algorithm (1981)

$$Q = R(A,B) \bowtie S(B,C) \bowtie T(C,D) \bowtie U(A,E)$$

1. Bottom-up semi joins
2. Top-down semi joins
3. Bottom-up full join

A	B	C	D	E
1	5	2	2	2
1	5	1	4	2

A	B	E
1	5	2



B
5
~~3~~
~~2~~

A	B
1	5
~~2~~	~~3~~
~~6~~	~~2~~

B	C
5	2
~~3~~	~~2~~
~~4~~	~~3~~
5	1

B	C	D
5	2	2
5	1	4

A	B
1	5
2	3
6	3

A	E
1	2
6	3

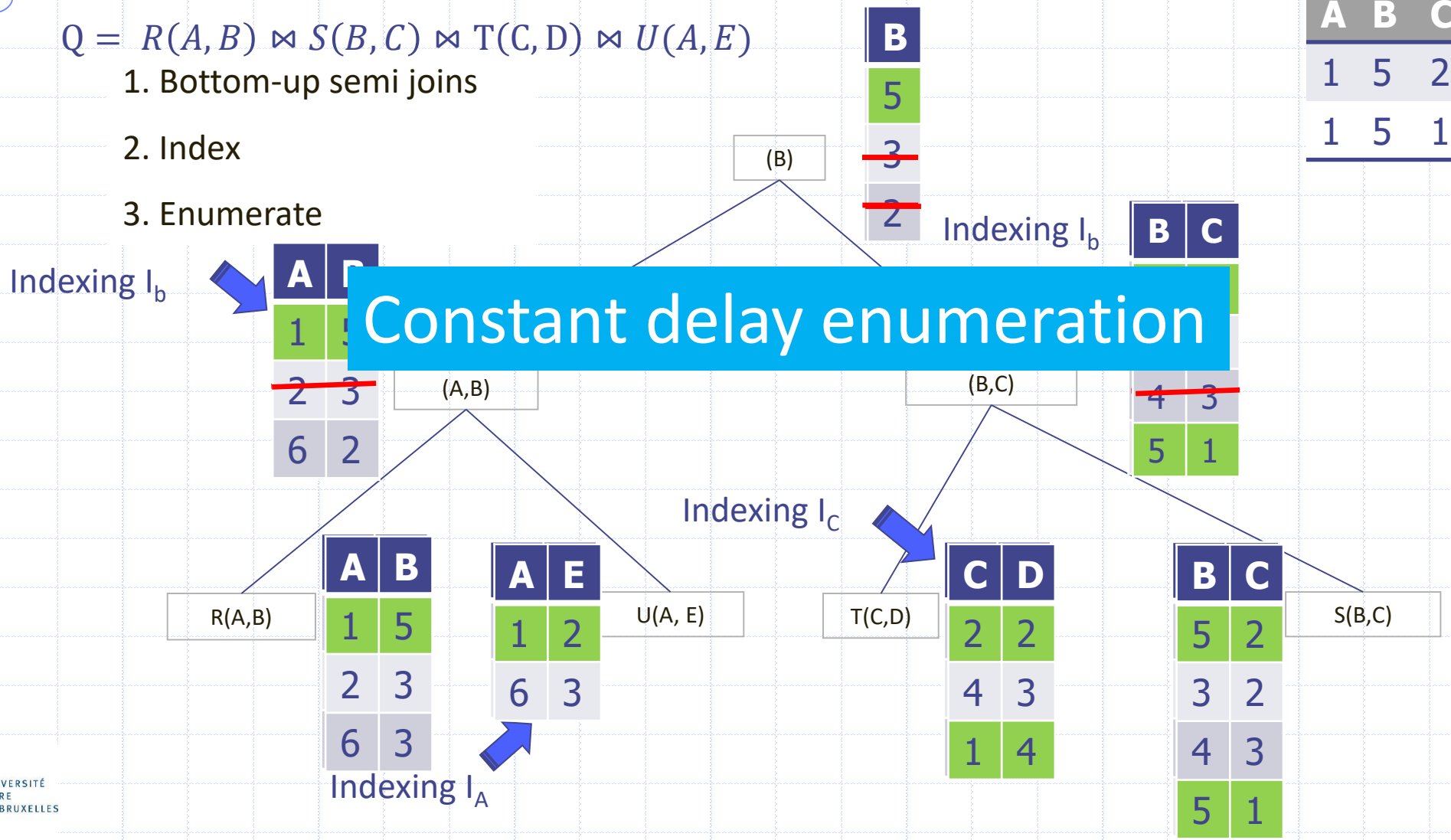
C	D
2	2
4	3
1	4

B	C
5	2
3	2
4	3
5	1

Constant delay Yannakakis- modified

$Q = R(A,B) \bowtie S(B,C) \bowtie T(C,D) \bowtie U(A,E)$

1. Bottom-up semi joins
2. Index
3. Enumerate



Enumerate

A	B	C	D	E
1	5	2	2	2
1	5	1	4	2

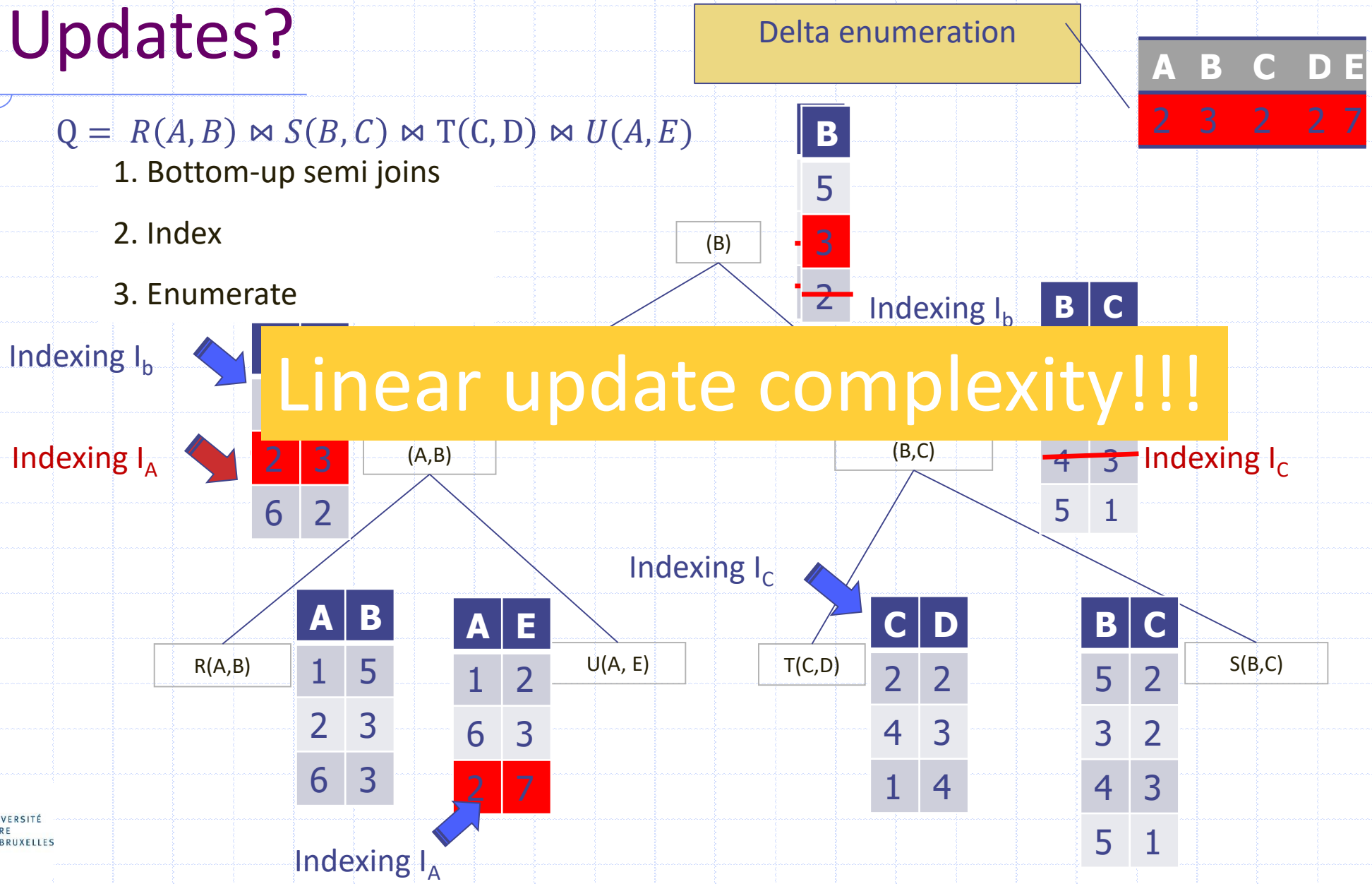
What we would like to?

- Desirable approach to:
  - View Enumeration: To have **constant delay(ideally)** – without materialization
  - Maintain in-memory data structure D- ***linear space*** *in the size of database db*
  - D is **maintainable** under updates easily (**constant time complexity- ideally**)
  - D provides constant time lookups.

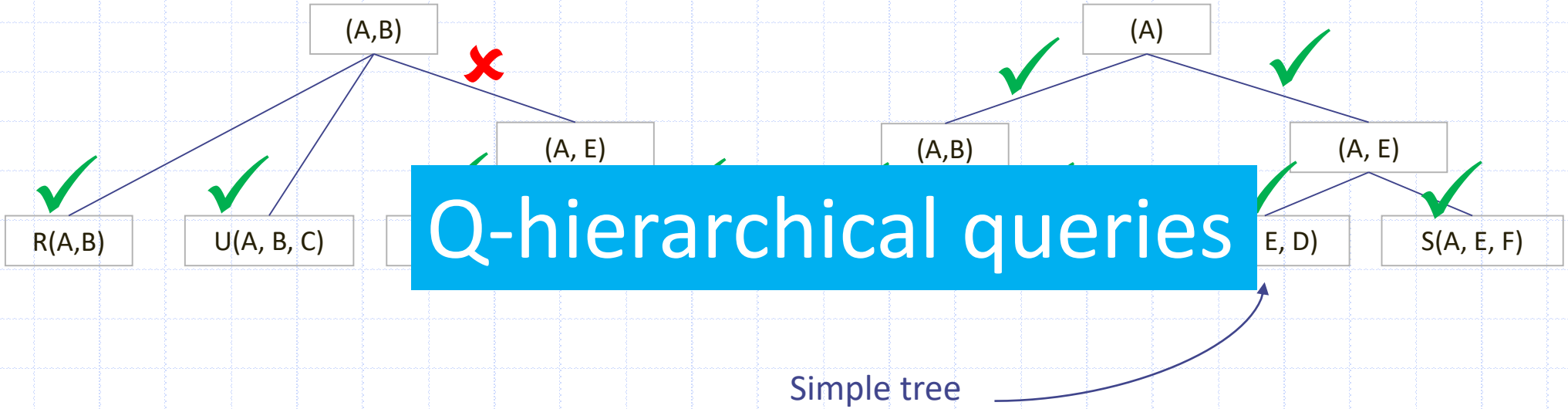
Updates?

$Q = R(A,B) \bowtie S(B,C) \bowtie T(C,D) \bowtie U(A,E)$

1. Bottom-up semi joins
2. Index
3. Enumerate



Constant update time (hierarchical queries)



Simple tree T

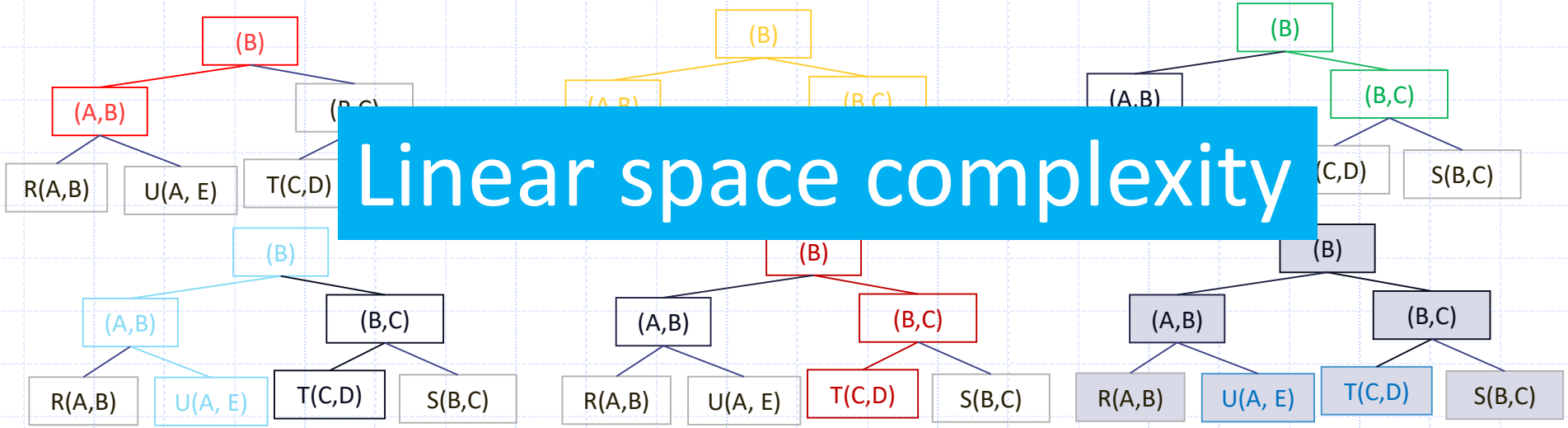
- Constant update time
- Constant delay enumeration

Dynamic Yannakakis (DYN)

- Generalization of the previous idea to:
  - Multiple tuple updates
  - Generalized Multiset Relations (GMRs)

Projections

$$Q(AB|ABC|BC|ABE|BCD|ED) = R(A,B) \bowtie S(B,C) \bowtie \mathrm{T}(\mathrm{C},\mathrm{D}) \bowtie U(A,E)$$



Linear space complexity

A CQ is ***free-connex*** acyclic iff it has a tree $T = (V, E)$ for which $\boldsymbol{N} \in$ V forms a rooted subtree in T.

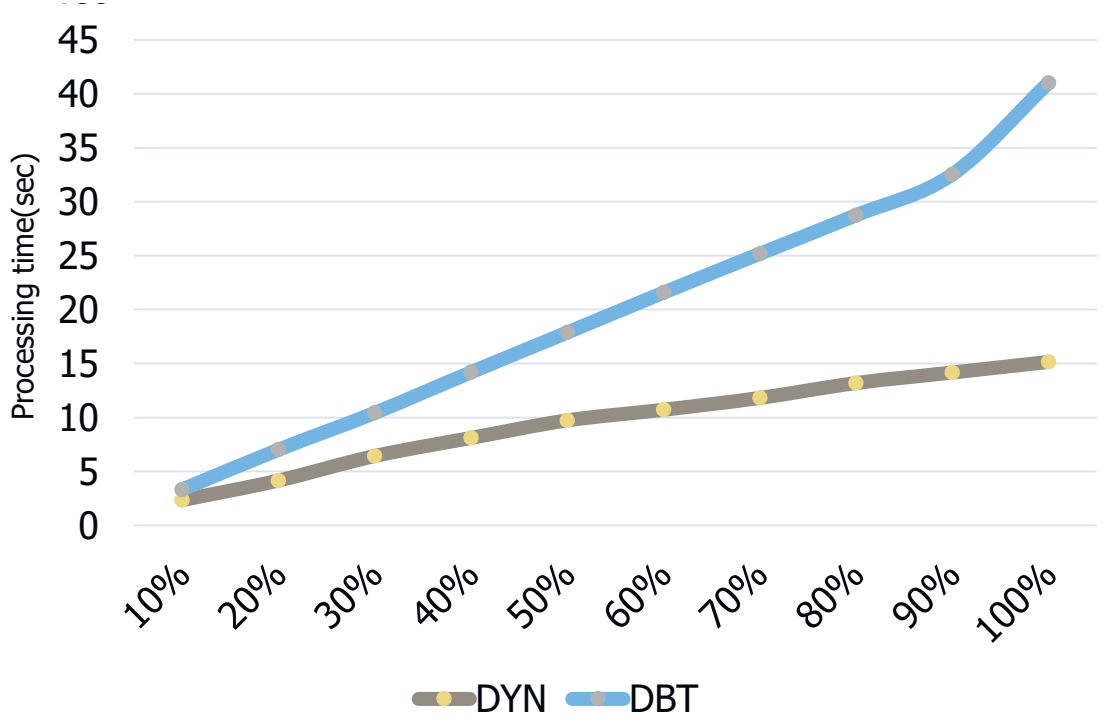
Compatible join tree

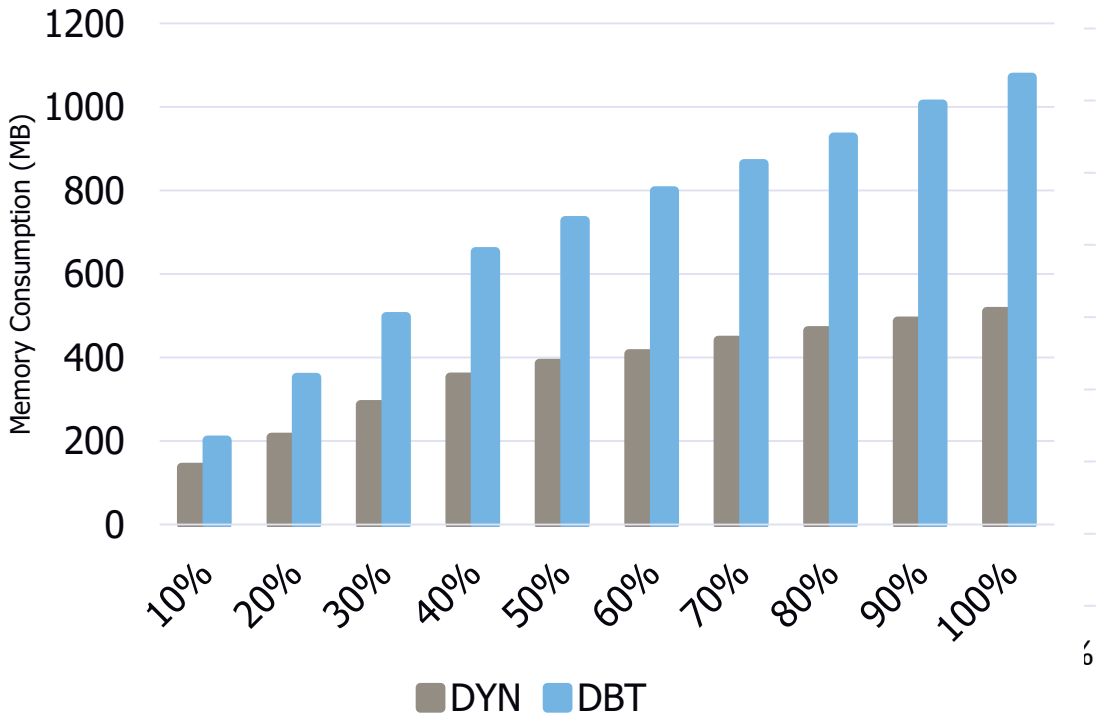
Acyclic Join Queries (ACQ)

Query class	Updates	Enumeration	Space	
			Free-connex	No Free-connex
Hierarchical	$O(1)$	$Constant\ delay$	$O(\|\|db\|\|)$	$O(\|\|db\|\| + \|\|Q(db)\|\|)$
Non-hierarchical	$O(\|\|db\|\| + \|\|u\|\|)$	$Constant\ delay$	$O(\|\|db\|\|)$	$O(\|\|db\|\| + \|\|Q(db)\|\|)$

Experiments- DYN vs H(IVM)

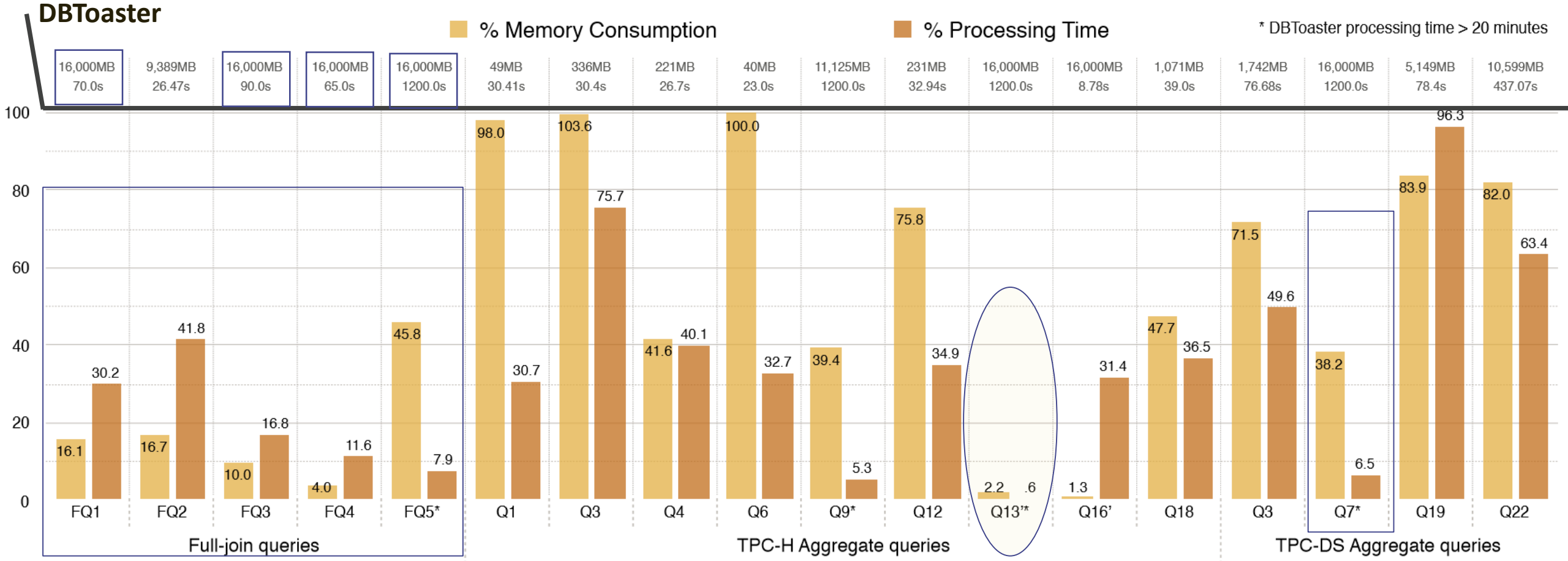
$TPCHQuery18 = LineItem \bowtie Orders \bowtie Customer \bowtie Supplier$



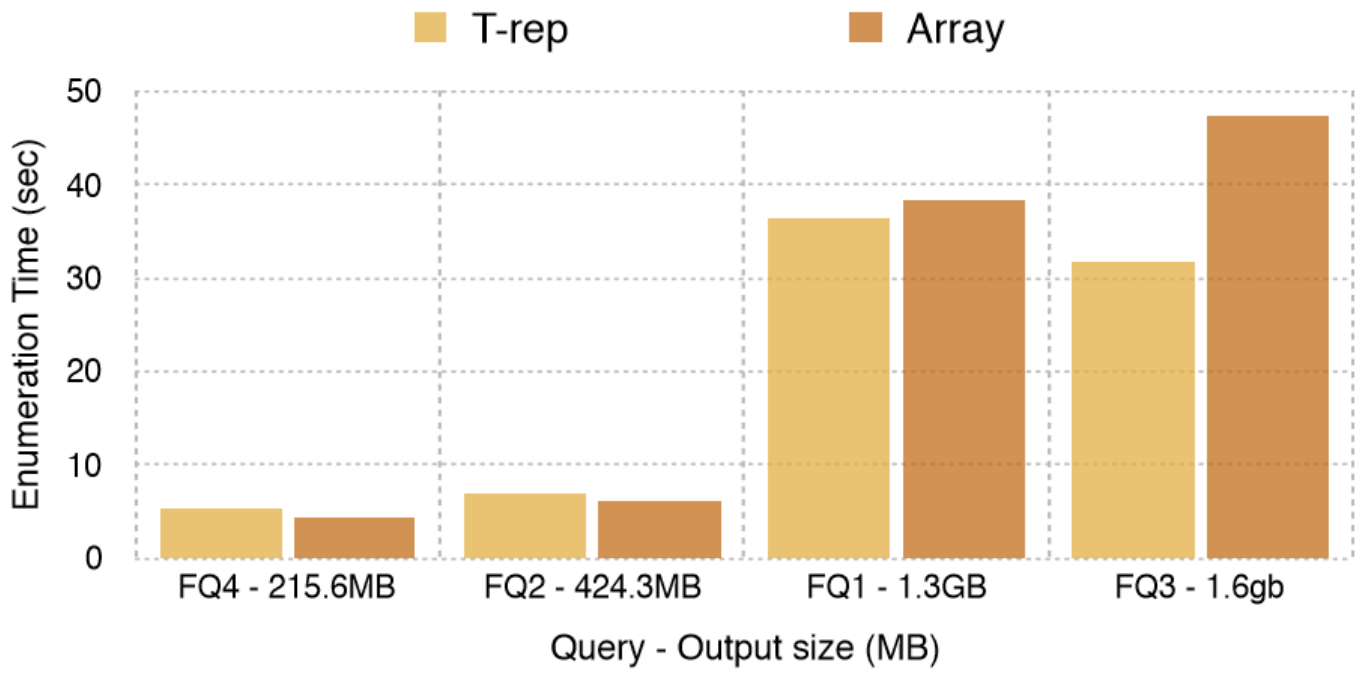


Experiments- DYN vs H(IVM)

100% = DBToaster, Max Memory: 16 GB, Max processing time: 20 minutes



Experiments- DYN vs H(IVM)



Enumerating the output from Array and T-representation

What we would like to?

- Desirable approach to:
  - View Enumeration: To have **constant delay(ideally)** – without materialization
  - Maintain in-memory data structure D- ***linear space*** *in the size of database db*
  - D is **maintainable** under updates easily (**constant time complexity- ideally**)
  - D provides **constant time** lookups.
- Published in **Sigmod 2017**

Questions answered!!!

CDE, Constant Update time, Linear space in database? → Hierarchical Free-connex queries

CDE, Linear update time, Linear space in database? → Non-hierarchical free-connex queries

What data structures and algorithms? → DCLR, DYN, IEDYN

Works for IFP and BI both? → Yes!!

Conclusion & Future work

- Conclusion
  - Algorithm is good in theory and practice
- Future work
  - Tackling multiple in-equalities between relations
  - Tackling outer joins
  - Implementation and evaluation on Complex event processing (CEP) queries

Publications

- **Published:**
  - Idris Muhammad, Martín Ugarte, and Stijn Vansummeren. "**The Dynamic Yannakakis Algorithm: Compact and Efficient Query Processing Under Updates."** *Proceedings of the 2017* ***ACM International Conference on Management of Data***. ACM, 2017.
- **Planned:**
  - **IEDyn: Compact and ecient non-equi join processing under updates**
    - **Authors:** Muhammad Idris, Stijn Vansummeren, Hannes Voigt, Martin Ugarte, Wolfgang Lehner
    - **Target venue and deadline:** VLDB 2018, November 2017
  - **Efficient query processing under updates on DCLRs and Factorizations (full paper on equi-joins)**
    - **Authors:** Muhammad Idris, Stijn Vansummeren, Hannes Voigt, Martin Ugarte, Wolfgang Lehner
    - **Target venue and deadline:** VLDB Journal

ECTS

Activity	Place/Organised by	ECTS	Type	Status
Workshop (DBDBD)	CWI	1	Informal	Completed (Fall 2015)
Workshop (DBDBD)	CWI	1	Informal	Completed (Fall 2016)
Technical writing and research ethics	ULB	1	General	Completed (Fall 2015)
French Language	ULB	2.5	General	Completed (Spring 2016)
German Language	TUD	2.5	General	Completed (Spring 2017)
Intellectual property, Copyright and Knowledge transfer	ULB	1	General	Completed (Spring 2016)
Academic Writing	ULB	3	General	Completed (Fall 2016)
IT4BI-DC Summer School	IT4BI	2	Project	Completed (Spring 2016/17)
IT4BI-DC Doctoral colloquium	ULB	3	Project	Completed (Spring 2016)
PhD seminar	TUD	2	Project	Enrolled (Spring 2017)
Winter school on Big Data	TUD	1	Project	Completed (Spring 2017)
Data systems architecture/coursera	ULB	2	Project	Planned (Fall 2017)
Transactional Information Systems (TIS)	TUD	3	Project	Planned (Fall 2017)
An introduction to good practices in research	ULB	1	General	Planned (Spring 2018)
Writing and reviewing papers	ULB/TUD	3.75	General	In-progress (Fall 2016)
Conference attendance	VLDB/TBD	2	Informal	Spring 2017

Total Credits: 31.75
General: 14.75
Project: 13 Informal: 4

THANK YOU!

Discussion and comments!