Bachelor-/Master-Forum 2021

Utilizing Emergent Technologies for Big Data Analytics

Institut für Informationssysteme (IFIS)

15.12.2021

Professor Dr. rer. nat. habil. Sven Groppe

https://www.ifis.uni-luebeck.de/index.php?id=groppe
Institut für Informationssysteme (IFIS)

- **Professoren**
  - Prof. Dr. rer. nat. Sven Groppe
  - Prof. Dr. rer. nat. Ralf Möller (Direktor)

- **Wissenschaftliche/Projekt-Mitarbeiter*innen**
  - Magnus Bender, M.Sc.
  - Marcel Gehrke, M.Sc.
  - Dr. rer. nat. Jinhua Groppe
  - Tobias Groth, M.Sc.
  - Mena Leemhuis, M.Sc.
  - Malte Luttermann, M.Sc.
  - Dr. rer. nat. Sylvia Melzer
  - **PD Dr. rer. nat. habil. Özgür Özçep**
  - Simon Schiff, M.Sc.
  - Benjamin Warnke, M.Sc.

- **Weitere Doktorand*innen**
  - Fabian Bormann, M.Sc.
  - Nils Finke, M.Sc.
  - Mattis Hartwig, M.Sc.
  - Marisa Mohr, M.Sc.
  - Simon Paasche, M.Sc.
  - Maurice Sambale, M.Sc.
  - Mareike Stender, M.Sc.
Contacts for Bachelor/Master Thesis

- For topics please check:  
  - [https://www.ifis.uni-luebeck.de/index.php?id=studentische-arbeiten](https://www.ifis.uni-luebeck.de/index.php?id=studentische-arbeiten)

- **Prof. Ralf Möller** (his topics are not in this talk, please contact him)  
  - Dynamic Probabilistic Relational Models  
  - Probabilistic Computing  
  - other topics (artificial intelligence/machine learning)

- **PD Özgür L. Özçep**  
  - Logical and mathematical Modeling for Knowledge Representation and Processing and for Machine Learning  
  - Ontology-based Data Access (OBDA)

- **Prof. Sven Groppe**  
  - Utilizing Emergent Technologies (e.g., GPU, Quantum Computing) for Big Data Analytics  
  - Databases  
  - Semantic Web  
  - IoT Data Management
Zoo of **Data Formats**, for example:

- relational data  
  - in relational databases
- XML  
  - for exchange
- JSON  
  - web data
- Resource Descr. Framework (RDF)  
  - Semantic Web
- graph data  
  - from social networks
- unstructured data  
  - of social media like wikis

▶ Parallel use of different Data Models for storing and processing

---

**Relational:**

<table>
<thead>
<tr>
<th>Primary</th>
<th>Secondary</th>
<th>Primary</th>
</tr>
</thead>
</table>

**XML:**

```xml
<root>
  <child>
    <first>hello</first>
    <sibling>sibling</sibling>
  </child>
</root>
```

**JSON:**

```json
{root:
  child:
    first:hello,
    sibling:sibling
}
```

**RDF/Graph Data:**

```
:article rdfs:subclassOf :bench:doc
:article1 rdf:type :article .
:article1 dc:creator :person1 .
:person1 foaf:name 'Martin' .
:person2 foaf:name 'Jennifer' .
```

**Ontology of Semantic Web**

- Inferred Fact
- Graph

**Unstructured Data:**

# Title #
The following issues are important:
1. Very Important Persons (VIPs)
2. Very Important Data (VID)
Every data model (here Semantic Web) has its own set of languages (data, query, rule, ...)

Semantic Web (Core) "Standards"

- Query: SPARQL
- Ontology: RDFS, OWL (2)
- Rule: RIF
- Data Format: RDF
Semantic Web: Ontology

- **Ontology as additional abstraction layer**
  - More than schema descriptions:
    - Specification of background knowledge (based on which new facts can be derived)
      ⇒ avoids storing of redundant data
      ⇒ supports re-use of data
      ⇒ supports data integration
      ⇒ increases computational complexity
Semantic Web: Ontology

- Ontology as additional abstraction layer
  - More than schema descriptions:
    - Specification of background knowledge
      (based on which new facts can be derived)
      ⇒ avoids storing of redundant data
      ⇒ supports re-use of data
      ⇒ supports data integration
      ⇒ increases computational complexity

- However, ontologies do not pose strict restrictions on RDF graphs
  - SHACL Shapes Constraint Language: for validating RDF graphs
    against a set of conditions
St. Arbeiten durch PD Özçep betreut

Wissenschaftl. Mitarbeiterin: Mena Leemhuis

- angesiedelt im Rahmen des BMBF-Verbundprojekts "Digitalisierung smarter Materialien und ihrer Herstellprozesse (SMADI)"
- Ontologie-basierter Zugriff auf Daten und Modelle (OBDMA)
  - Validierung von SHACL-Integritätsbedingungen
  - Mapping-Generierung auf der Basis von SHACL-Integritätsbedingungen
  - Optimierung des OBDMA-Transformationsalgorithmus mit SHACL-Integritätsbedingungen
  - Optimierung des OBDMA-Transformationsalgorithmus mit dem Magic-Set-Verfahren
  - Rollenkomposition
  - Anfrageupdate
  - Bedingte Anfragen
Thesis supervised by Prof. Groppe
COVID-19 Knowledge Graph (KG)

Some issues of knowledge graph quality:
- contradictions in (evolving) facts
- checking vague formulations and compare them with other given information
- errors with ambiguity: E484K mutation of B.1.1.7 is a SARS-CoV-2 virus, but reinfection with E484K is only possible for patients recovered from SARS-CoV-2 virus not mutating E484K (→ introduce class “SARS-CoV-2 without E484K mutation“)

[1] https://jamanetwork.com/journals/jamanetworkopen/fullarticle/2774102
[4] https://www.bmj.com/content/372/bmj.n359
High Quality Knowledge Graphs from recent English, French and German Emergent Trends with the example of COVID-19

- Project (DFG/ANR) starting soon (partners: universities of Paris, Toulouse and Lübeck)
- Thesis in the areas of
  - data capturing,
  - visualization & analysis,
  - detection of contradictions in KG,...
Definitions of Internet-of-Things (IoT)

- Internet is extended by adding things as network nodes.
  - connect those physical objects that are not yet connected with the help of the Internet and
  - thus create significant added value
    - a better collaboration between humans and machines through new types of applications in various areas.
    - Application domains like Smart City, Smart Healthcare, Smart Farming, Smart Factory, ...
- "IoT allows people and things to be connected Anytime, Anyplace, Anything and Anyone ideally using any path/network and Any service."

CISCO: *Age of IoT emerged between 2008 and 2009*
Things in IoT are heterogeneous

- **Different** kinds of **devices**: Smartphone/-watch-/tv/..., sensors, routers, tablets, raspberry pi (and similar), ...
- **Hardware** (e.g. **CPU**)
- **Operating system** (**OS**)
- **Network** protocols
- **Capabilities**
  - Performance of **CPU**
  - Storage capacity
  - Access to networks (wireless, stationary, ...)
  - ...
- ...

...
Multi-Platform Development (e.g. of DBMS)

- **Native Binaries via C/C++**
  - support of a new platform: **porting code** is necessary
  - code **close to hardware, fast execution**
  - direct access to **native libraries**
  - **doesn't run in browser**
  - most server DBMS: C/C++ code

- **Java/Java Virtual Machine (JVM)**
  - runs on **many platforms (without porting code)**
  - interpreted bytecode, via Just-In-Time compilation **comparable speed to native execution**
  - **no direct access to native libraries**
  - does **neither run on iPhone nor in browser**
  - many NoSQL/NewSQL/Cloud DBMS: Java (or JVM language like Scala) code

- **Code generation for query processing** via C/C++ or Janino-Compiler (JVM)
Multi-Platform Development with Kotlin

Targets:

- JVM
  - Desktop
  - Server
  - Android
- JS
  - Browser
  - Server NodeJS
- LLVM/Native
  - Win (X64 / X86)
  - Linux (arm64/32 / 32Hfp / Mips(e)l32 / X64)
  - MacOS (X64)
  - iOS (arm32/64 / X64)
  - watchOS (arm32/64 / X86)
  - Android (arm32/64)
  - WebAssembly (wasm32)

- Most target platforms are supported
- Splitting the project in platform-independent and platform-dependent code
  - Platform-dependent code can be partly coded in the programming language of the target platform (e.g., Java for JVM, JS for Web)
- Enables one code repository for various target platforms
  - Sharing of code between server & (various) clients
- Avoids efforts to port code (into other programming languages)
The Power of Multi-Platform: LUPOSDATE3000

- ultra-fast in jvm...

...but also enabling web demos running completely in the browser!

B. Warnke, M.W. Rehan, S. Fischer, S. Groppe: Flexible data partitioning schemes for parallel merge joins in semantic web queries in: BTW'21

Internet-of-Things (IoT) Architectures
Example of an IoT Scenario: Parking Slots
Network Topologies to be considered for IoT query processing

- **Scenario:** Subnets of sensors of parking slots are connected via different network topologies

**Random:**

**Ring:**

**Full:**

**Uniform:**
Properties of Routing Algorithms

- Where is the routing performed?
  - **Centralized**
    - one or few central nodes as routers
    - other nodes ask the routers
  - **Decentralized**
    - each and every node is a router
    - nodes exchange information

- How fast do routes change?
  - **Static**
    - routes change slowly
    - manual configuration or restart of the routing algorithm
  - **Dynamic**
    - routes change more quickly
    - proactive or reactive

- How much information do the routers have?
  - **Global**
    - router node has knowledge of the entire network
    - all router nodes have the same view of the network (in a stable state)
  - **Local**
    - no node has knowledge of the entire network
    - node never knows the complete route from a source to a sink

- When is the routing performed?
  - **In Advance**
    - routing is done before forwarding
  - **On Demand**
    - routing and forwarding is one mixed process
Embed Query Execution Plan in Topology

\[ \text{Select } p \]
\[ \text{From } (G \text{ Join } H \text{ Join } J) \]

Projection \( p \)

Join

Picture of a network with nodes labeled G, H, J, and WSN, showing the flow of data and the execution plan.
Inserting Data replicated due to indexing scheme

State-of-the-Art:

- IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL)
  → All-Shortest-Paths (ASP): 17-48%
- multi-cast: 24%
- additional devices: 3%

Multi-Cast:

- network traffic reduction during insert by using IPv6 RPL
  → multi-cast: 24%
  → additional devices: 3%
Querying Data in IoT (4 ⊗ 13 → 3)

State-of-the-Art:

Combining Routing & Processing:
Thesis supervised by Prof. Groppe

(S)IoT Database LUPOS

Wissenschaftl. Mitarbeiter: Benjamin Warnke/Simon Paasche

- Project **BigSloT** (DFG, together with Stefan Fischer/ITM) (Warnke)
- **External Promotion Bosch** (Paasche)
- Combining (e.g., geo) **routing protocols with query processing**
- Compress data (in storage/messages) according to SHACL definitions, reformulate queries to directly work on compressed data
- **Continuous Queries** (e.g., redundant processing, efficient recovery after crashes, new types of windows for consistency checks)
- **Distributed RDFS Inference**
- Digital twins of machines in production
- **Automatic Testing of Databases**: randomized testing, generation of test queries, automatic minimization of test cases, ...
- **Semantic Web Layer for NebulaStream** (co-supervision with DIMA/Berlin [D])
IoT and Cloud as Platforms for Processing the IoT DATA
IoT and Cloud as Platforms for Processing the IoT DATA

- ...but there are also **other platforms**!
Hybrid Multi-Model Multi-Platform (HM3P) Database

+ full and uniform data integration at database level
+ performance: fully optimized across different data models
+ transparent fault-tolerance
+ SQL standards: relational ('87), XML ('03), temporal ('11), JSON ('16), Multi-dimensional Arrays ('19), schemaless ('19), streams ('20?), property graphs ('21?)
+ features of different types of databases running on different platforms can be used
Variant: **Semantic HM3P (SHM3P) DB**

- **Reasoning:**
  - Lightweight reasoning on large data sizes of IoT devices
  - Heavyweight reasoning on moderate data sizes
  - Heavyweight reasoning on large data sizes
  - Reasoning on small data sizes of mobile devices

- **SHM3P Database**
  - Single instance
  - Offers (fully cross-platform optimized) functionality of & replaces

- **How to integrate the different reasoning capabilities and requirements into one transparent global reasoner?**
  - **Semantic Layer as glue** between other models and platforms
  - **new challenges** like integrating different types of reasoners in a transparent global reasoner

- **Features of HM3P databases**
  - Easier data integration

- **Performance issues** may occur due to semantic layer
## Architectures of Emergent Hardware

<table>
<thead>
<tr>
<th>Core Complexity</th>
<th>Multi-Core CPU</th>
<th>Many-Core CPU</th>
<th>Graphics Processing Unit (GPU)</th>
<th>Field Programmable Gate Arrays (FPGA)</th>
<th>Quantum Computer/Annealer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cores</td>
<td>~10</td>
<td>~100</td>
<td>~1000</td>
<td>~100 000</td>
<td>~50 / ~5000 qubits</td>
</tr>
<tr>
<td>Core Complexity</td>
<td>Complex (optimized for single thread performance)</td>
<td>Simple</td>
<td></td>
<td></td>
<td>Complex</td>
</tr>
<tr>
<td>Computational Model</td>
<td>MIMD + SIMD</td>
<td>SIMD</td>
<td>Data-Flow</td>
<td>Universal/Adiabatic Quantum computing</td>
<td>Quantum Logic Gates/Fluctuation (glob. Min.)</td>
</tr>
<tr>
<td>Parallelism</td>
<td>Thread and Data Parallel</td>
<td>Data Parallel</td>
<td>Arbitrary</td>
<td></td>
<td>Quantum Superposition</td>
</tr>
<tr>
<td>Memory Model</td>
<td>Shared</td>
<td>Distributed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>150 W</td>
<td>200 W</td>
<td>250 W</td>
<td>50 W</td>
<td>25 KW</td>
</tr>
<tr>
<td>Database Op.</td>
<td>Query Optimization (Enumeration of Plans), Concurrency Control</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Efficient Processing Of</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Query Processing</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Legend:**
- Purple: Computational Unit
- Green: Execution Controller
- Orange: Interconnection Network
- Pink: On-Chip Memory

---

Institut für Informationssysteme | Prof. Dr. habil. S. Groppe
Architectures for Hybrid Indices (Variant 1)
Architectures for Hybrid Indices (Variant 2)
Good News

- **Framework** for Hybrid Indices including Benchmark Suite available

  ~⇒ Tobias Groth

- Offered topics deal with
  - integrating suitable/develop new indices in this hybrid approached
Thesis supervised by Prof. Groppe
GPU Accelerated Index Structures

Wissenschaftl. Mitarbeiter: Tobias Groth

- Project Hybrid² (DFG, together with Prof. Pionteck (Magdeburg))
- Filter based on Hashing
  - Bloom: [Paper Tutorial Code](#)
  - Cuckoo: [Paper Code 1](#) [Code 2](#)
  - Morton: [Paper Code](#)
  - Vacuum: [Paper Code](#)
- Search Trees
  - B/B+-tree: [Paper Code](#)
  - CSB/CSB+: [Paper Code](#)
  - B^{ed}-Tree: [Paper Code](#)
  - Masstree: [Paper Code](#)
- ...
Quantum Mechanics

- Very small particles and light behave differently from objects in normal life
- Mechanics of light and matter at the atomic and subatomic scale are described by quantum theory - forming the underlying principles of chemistry and most of physics
- Quantum theory has brought us the information age with its disruptive technologies of - transistors,
  - lasers,
  - nuclear power, and
  - superconductivity...
  - ...and now also quantum computers!

Wavefunctions of the electron in a hydrogen atom at different energy levels. Brighter areas represent a higher probability of finding the electron.
Quantum Computer

- use of quantum-mechanical phenomena such as superposition and entanglement to perform computation
- Different types of quantum computer, e.g.
  - Universal Quantum Computer
    - uses quantum logic gates arranged in a circuit to do computation
    - measurement (sometimes called observation) assigns the observed variable to a single value
  - Quantum Annealing
    - metaheuristic for finding the global minimum of a given objective function over a given set of candidate solutions
    - i.e., some way to solve a special type of mathematical optimization problem
Classical versus Quantum Computing

<table>
<thead>
<tr>
<th>Information Unit</th>
<th>Classical</th>
<th>Quantum</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Binary Digit (Bit):</strong></td>
<td>- basis of a 2-level system</td>
<td>- basis of a 2-level quantum system</td>
</tr>
<tr>
<td></td>
<td>- can be in state 0 or 1</td>
<td>- can be in state $</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Classical</th>
<th>Quantum</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Logic Gate:</strong></td>
<td>- performs on 1 or more bits to produce a single bit output</td>
<td>- performs on 1 or more qubits to change the quantum state of a single qubit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example Operation</th>
<th>NOT/Inverter:</th>
<th>Quantum Logic Gate:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NOT/Inverter:</strong></td>
<td>$In \rightarrow Out$</td>
<td>$</td>
</tr>
<tr>
<td><strong>Quantum Circuit:</strong></td>
<td>$</td>
<td>In\rangle -</td>
</tr>
</tbody>
</table>

\[
\begin{array}{|c|c|}
\hline
In & Out \\
\hline
0 & 1 \\
1 & 0 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
In & Out \\
\hline
|0\rangle & |1\rangle \\
|1\rangle & |0\rangle \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
\text{Example Operation} & \text{NOT/Pauli}_{x}-\text{Gate:} \\
\hline
\text{Quantum Circuit:} & |In\rangle - X - |Out\rangle \\
\text{Alternatively:} & |In\rangle - |Out\rangle \\
\hline
\frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) & \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \\
\frac{3i}{5} |0\rangle + \frac{4}{5} |1\rangle & \frac{4}{5} |0\rangle + \frac{3i}{5} |1\rangle \\
\hline
\end{array}
\]
Digital versus Quantum Circuits

Digital Circuit

Logic Gates

Quantum Logic Gates

Consists of NAND gates

Consists of Toffoli and CNOT gates

In- and Output Full Adder

Table:

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

In- and Output Full Adder

Example

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C_in</th>
<th>C_out</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The dotted square marks a superfluous gate if uncomputation to restore the B output is not required. [Feynman, 1986]
Timeline of Quantum Computers

---

- **IBM Eagle**: 127 qubits (16.11.2021)

---

**Main Data Source**: IBM Eagle

**Roadmaps**: IBM, Google, Quantum Brilliance, D-Wave

---

*IBM*

*IBM Roadmap’20*

*Google*

*Google Roadmap’21*

*Intel*

*Rigetti*

*QuTech*

*USTC*

*Xanadu Quantum Technologies*

*Quantum Brilliance*

*Roadmap’21 (Room Temperature)*

*D-Wave (Quantum Annealing)*

*D-Wave Roadmap’21*
Potential of Quantum Algorithms

- **Quantum Algorithm Zoo** as example of collection of *important* quantum algorithms

<table>
<thead>
<tr>
<th>Covered Years</th>
<th>1974-today</th>
</tr>
</thead>
<tbody>
<tr>
<td>#Investigated References</td>
<td>430 (visited on October 2021)</td>
</tr>
<tr>
<td>#Algorithms</td>
<td>64</td>
</tr>
<tr>
<td>Speedups</td>
<td></td>
</tr>
</tbody>
</table>

- **Superpolynomial**: 31,
- **Polynomial**: 27,
- **Constant factor**: 1,
- **Varies**: 3,
- **Various**: 1,
- **Unknown**: 1

**Terminology:**

- $\alpha$: positive constant
- $C(n)$: runtime of the best known classical algorithm
- $Q(n)$: runtime of the quantum algorithm

Superpolynomial Speedup: $C = 2^{\Omega(Q^\alpha)}$

Polynomial Speedup: otherwise
Very Important Quantum Algorithms 1/2: Shor's Algorithm

- factoring integers in polynomial time
  - Depth of quantum circuit$^2$ to factor integer $N$:
    $O((\log N)^2 (\log \log N) (\log \log \log N))$
  - superpolynomial speedup, i.e., almost exponentially faster than the most efficient known classical factoring algorithm (general number field sieve):
    $O(e^{1.9 (\log N)^{\frac{1}{3}} (\log \log N)^{\frac{2}{3}}})$

- Important for cryptography → Post-Quantum Cryptography
- Most quantum algorithms with superpolynomial speedup like Shor's algorithm are based on quantum Fourier transforms (quantum analogue of inverse discrete Fourier transform)
Very Important Quantum Algorithms 2/2: Grover's Search Algorithm

- **Black box** function (oracle) $f : \{0, \ldots, 2^b - 1\} \mapsto \{true, false\}$
- **Grover's search** algorithm finds one $x \in \{0, \ldots, 2^b - 1\}$, such that $f(x) = true$
  - if there is only **one solution**: $\frac{\pi}{4} \cdot \sqrt{2^b}$ basic steps each of which calls $f$
    Let $f'(b)$ be runtime complexity of $f$ for testing $x$ to be true:
    $\Rightarrow O(\sqrt{2^b} \cdot f'(b))$
  - if there are $k$ possible solutions: $O(\sqrt{\frac{2^b}{k}} \cdot f'(b))$
- **Basis of many other quantum algorithms and applications**
# Algorithms (used e.g. in Query Optimization) and their Quantum Counterparts

<table>
<thead>
<tr>
<th>Query Optimization Approach</th>
<th>Basic Algorithm</th>
<th>Quantum Computing Counterpart</th>
</tr>
</thead>
<tbody>
<tr>
<td>[S+79]</td>
<td>Dynamic Programming [E04]</td>
<td>[R19] [A+19]</td>
</tr>
<tr>
<td>[IW87], QA: [TK16]</td>
<td>Simulated Annealing [KGV83]</td>
<td>[J+11]</td>
</tr>
<tr>
<td>[MP18] [Y+20] [W+19] [O+19]</td>
<td>Reinforcement Learning [BSB81]</td>
<td>[S+21] [DCC05]</td>
</tr>
<tr>
<td>[GPK94]</td>
<td>Random Walk [BN70]</td>
<td>[ADZ93] [A+01]</td>
</tr>
<tr>
<td>[BFI91]</td>
<td>Genetic Algorithm [H92]</td>
<td>[W+13]</td>
</tr>
<tr>
<td>[TC19]</td>
<td>Ant Colony Optimization [CDM91]</td>
<td>[WNF07] [G+20]</td>
</tr>
</tbody>
</table>

This list is not complete...
Open Challenges for QC for Databases

- Project QC4DB (BMBF, together with Quantum Brilliance)
- Are QC counterparts of basic algorithms used in query optimizations suitable for speeding up databases?
- What should be the properties of a quantum computer (e.g. #qubits, latencies of gates) to achieve certain speedups?
- How to combine classical and quantum computing algorithms to achieve good speedups with few qubits? (...for running database optimizations on current available quantum computers...)
- What other database domains besides query optimization benefit from quantum computers? (In short: those based on mathematical optimization problems, but also other...?)
Summary and Conclusions

• Different **data models** and their special features
  - ➔ Multi-Model Databases

• Different **platforms and** a need for different types of databases
  - Different features
  - ➔ Multi-Platform Databases

• Databases spanning over different platforms in operation (supporting multiple data models)
  - ➔ Hybrid Multi-Model Multi-Platform (HM3P) Databases

• Emergent Technologies like IoT, GPU, Quantum Computing

• We are offering **many topics for bachelor/master thesis**…
  - Please contact me: groppe@ifis.uni-luebeck.de