Incorporating Blockchain into RDF Store at The Lightweight Edge Devices

Anh Le-Tuan, Darshan Hingu, Manfred Hauswirth and Danh Le Phuoc | Technische Universität Berlin
RDF4LED - An RDF Engine for Edge Devices

Incorporating Blockchain into RDF4Led
Why Edge Devices ???

**Edge Computing - Internet of Things**

- reduce network overhead.
- reduce latency for real-time applications.
- improve scalability.
- better privacy control.

**Linked Data**

- enable data integration of heterogeneous sources.
- enable data federation over edge nodes.
Missions

Moving semantic data processing task away from centralised cloud for the IoT

- How much semantic data on small devices?
- How to scale data federation over small devices on edge systems?
- How to encourage people sharing the data from their edge devices?
System Design

- **Physical storage:** Introduce a *two-layers index schema* for triples to adapt the *flash I/O behaviors*.

- **Buffer Manager:** Introduce a *buffer replacement policy* to reduce the *number of overwritten*.

- **Query Executor:** Introduce an *adaptive strategy* for *iterative join execution* to avoid caching *intermediated results* that *minimizing memory usage*. 
**Buffer Layer:**
- managing triples in memory.
- acting as an index for the Physical Layer by keeping the first triple of a page and mapping it to the physical address of the page.
- organising the atomic triples into pages.
- grouping the dirty pages into writing blocks.

**Physical Layer:**
- Managing triples on flash.
- Using molecule-based data structure to avoid storing repetitive values.
- Keeping the molecules continuously sorted in pages, and keeping the pages sorted in blocks.
### Targeted Small Devices

<table>
<thead>
<tr>
<th>Devices</th>
<th>Pi0</th>
<th>BBB</th>
<th>GII</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>ARM 11, 1.0 GHz, 1-core</td>
<td>ARM A8, 1.0 GHz, 1-core</td>
<td>x86 Quark, 0.4 GHz, 1-core</td>
</tr>
<tr>
<td>RAM</td>
<td>512 MB</td>
<td>512 MB</td>
<td>256 MB</td>
</tr>
<tr>
<td>Storage</td>
<td>Transcend MicroSD 16GB class 10 (40MB/s)</td>
<td>Debian 7.0</td>
<td>Yocto</td>
</tr>
<tr>
<td>OS</td>
<td>Raspbian</td>
<td>Debian 7.0</td>
<td>Yocto</td>
</tr>
</tbody>
</table>
Incorporating Blockchain into RDF4Led

System Overview
Incorporating Blockchain into RDF4Led

System Workflow

SPARQL Query Processor

Validation Service

Smart Contract Manager

Smart Contracts Storage

Distributed RDF Storage

1. Index Entries
2. Triple Pattern Request
3. Smart Contracts
4. Payment Request
5. Payment Confirmation
6. Trigger Contracts
7. Index Entries
8. Request with Index Entries
9. Matched Triples
Incorporating Blockchain into RDF4Led

Physical Organisation

Transaction 1:
- Owner: ClientA
- Price: 0.1
- IPFS:hash1

Transaction 2:
- Owner: ClientB
- Price: 0.1
- IPFS:hash2

Transaction N:
- Owner: ClientN
- Price: 0.1
- IPFS:hashN

Distributed RDF Storage

Blockchain

Smart Contracts Storage
Incorporating Blockchain into RDF4Led

System Deployment
Incorporating Blockchain into RDF4Led

Evaluation - Input (1) Acc. Throughput on Static Cluster Sizes

Dataset Size (in millions)

Throughput (triples/sec)

0 200 400 600 800 1,000

1,000 1,500 2,000 2,500

10 nodes Inserting
Incorporating Blockchain into RDF4Led

Evaluation - Input (2) Acc. Throughput on Varying Cluster Sizes

![Graph showing throughput in thousands for 100 million triples dataset vs number of nodes.]
Incorporating Blockchain into RDF4Led

Evaluation - Query

![Chart showing the relationship between the number of blocks returned in query patterns and the time in seconds (log scale). The x-axis represents the number of blocks (0.5, 1, 10, 50, 100, 500, 1000), and the y-axis represents time in seconds (0.5, 5, 30, 60, 350, 1000).]
Thank You !!!
Write Management (1): Clustering example
Replacement policy:
- data access frequency.
- probability of data being changed in future.

Cache organisation:
- using 2 buffer queues (hot queue/cold queue).
- the recent active blocks are kept in the hot queue.
- the inactive blocks are kept in the cold part.

Blocks are sorted in queue with following priority:
(i) the clean/unmodified blocks;
(ii) the number of atomic triples in a block
(iii) the higher density blocks:
(iv) the last recently access block.
Push-based Join Algorithm

**Algorithm 1: Join propagation**

1. **Function** `propagate(μ, P)`
   - **input**: μ: mapping, P: set of triple query patterns
   - **output**: μ: mapping
2. **if** `isEmpty(P)` **then**
3.   `return μ;`
4. `p ← findNextPattern(μ, P);`
5. `pkey ← createKey(μ, p);`
6. `T ← indexScan(pkey);`
7. `P' ← P \ {p};`
8. **for** `t ∈ T` **do**
9.   `μ ← bindMapping(t, p);`
10. `propagate(μ, P');`
11. `μ ← resetMapping(t, p);`
12. `return pnext;`

**Algorithm 2: Find the next triple pattern**

1. **Function** `findNextPattern(μ, P)`
   - **input**: μ: mapping, P: set of triple query patterns
   - **output**: P: triple query pattern
2. `pnext ← null;`
3. `smin ← Integer.max;`
4. **for** `p ∈ P` **do**
5.   **if** `isShared(μ, p)` **then**
6.     `pkey ← createKey(μ, p);`
7.     `I ← indexLookUp(pkey);`
8.     `s ← sizeOf(I);`
9.     **if** `s < smin` **then**
10.       `smin ← s;`
11.       `pnext ← p;`
12. `return pnext;`
RDF4LED (cont.)

Evaluation Results - Input (1)

(a) Input throughput on Intel Galileo Gen II

Number of triples per second

Number of triples in total

1k

500

2m 5m 10m 15m 20m 25m 30m

RDF4Led
JENA-TDB
VIRTUOSO

Incorporating Blockchain into RDF Store at The Lightweight Edge Devices | Anh Le-Tuan | Seite 19
RDF4LED (cont.)

Evaluation Results - Input (2)

(b) Input throughput on Pi Zero

Number of triples per second

Number of triples in total
Evaluation Results - Input (3)

(c) Input throughput on BeagleBone Black

- Number of triples per second
- Number of triples in total
Evaluation Results - Query (1)

(a) Query response time on Intel Galileo Gen II

- Queries: F1, F2, F3, F4, F5, L1, L2, L3, L4, L5, S1, S2, S3, S4, S5
- Systems: VIRTUOSO, RDF4Led, JENA TDB
- Units: second (in log scale)
Evaluation Results - Query (2)

(b) Query response time on Raspberry Pi Zero

Incorporating Blockchain into RDF Store at The Lightweight Edge Devices

<table>
<thead>
<tr>
<th>Queries</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
<th>F5</th>
<th>L1</th>
<th>L2</th>
<th>L3</th>
<th>L4</th>
<th>L5</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIRTUOSO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RDF4Led</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JENA TDB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation Results - Query (3)

(c) Query response time on Beagle Bone Black

Incorporating Blockchain into RDF Store at The Lightweight Edge Devices | Anh Le-Tuan | Seite 24
Evaluation Results - Query (4)

(d) Query response time on Beagle Bone Black (20 mil)

 Queries

F1 F2 F3 F4 F5 L1 L2 L3 L4 L5 S1 S2 S3 S4 S5

second (in log scale)
Evaluation Results - Memory (1)

(a) Memory consumption of inserting of VIRTUOSO, JENA TDB, RDF4Led

Memory consumption (MB)

Number of triples in total
Evaluation Results - Memory (2)

(b) Memory consumption for querying of VIRTUOSO, JENA TDB, RDF4Led

- Memory boundary of the PI0 and BBB
- Virtuoso on PI0 and BBB
- Jena TDB
- Memory boundary of the GII
- Virtuoso on GII
- RDF4Led

Number of triples in total

Memory consumption (MB)