TriplePlace
A flexible triple store for Android with six indices

Natanael Arndt

arndtn@gmail.com
http://people.comiles.eu/natanael

December 2, 2011
TriplePlace
A flexible triple store for Android with six indices
1 Introduction
   - Motivation
   - Goal

2 Basics
   - Resource Description Framework (RDF)
   - Triple Store
   - Hexastore
   - Tokyo Cabinet

3 Implementation and Evaluation
   - Foundation
   - Dictionary
   - Indexing Structure
   - API
   - Benchmark

4 Conclusion
   - Future Work
Introduction
Motivation

Figure: LOD cloud two years ago
Motivation

Figure: LOD cloud as of September 2011 [CJ11]
Motivation

- Semantic Web is growing
Motivation

- Semantic Web is growing
- Mobile applications become more important
  - e.g. Mobile Social Semantic Web Client [Arn10]
Motivation

- Semantic Web is growing
- Mobile applications become more important
  - e.g. Mobile Social Semantic Web Client [Arn10]
- Only one triple store for Android
Motivation

- Semantic Web is growing
- Mobile applications become more important
  - e.g. Mobile Social Semantic Web Client [Arn10]
- Only one triple store for Android
  - Originally meant for servers
Motivation

- Semantic Web is growing
- Mobile applications become more important
  - e.g. Mobile Social Semantic Web Client [Arn10]
- Only one triple store for Android
  - Originally meant for servers
  - High memory usage
Motivation

- Semantic Web is growing
- Mobile applications become more important
  - e.g. Mobile Social Semantic Web Client [Arn10]
- Only one triple store for Android
  - Originally meant for servers
  - High memory usage
  - Large in the APK (∼ 4MB extra)
Goal

Create a better triple store for Android!
Goal

Create a better triple store for Android!

What is better?
Goal

Create a better triple store for Android!

What is better?

- Speed
- Memory
- Storage space
- APK size
Goal

Create a better triple store for Android!

What is better?

- Memory
- APK size < 1MB
Goal

Create a better triple store for Android!

What is better?

- Speed
- Memory
- APK size < 1MB
Resource Description Framework (RDF)

- W3C standard
- Framework for representing information on the Web
- Information are represented as statements
- These statements consist of:
  - (subject, predicate, object)
- also known as **Triple**
Resource Description Framework (RDF)

Figure: A Triple
Resource Description Framework (RDF)

Figure: RDF graph

subject node (<http://aksw.org>)
predicate edge
  (<http://purl.org/dc/elements/1.1/title>)
object node or leaf node ("AKSW")
A **Triple Store** is a database designed to store and retrieve Triples. Triple Stores can be built on top of an existing RDBMS or with an own on-disk layout.
Hexastore

- Indexing scheme for Triple Stores
- Dictionary to translate URIs and literals to IDs
- Six indices to materialize all permutations of a Triple
  (spo, sop, pso, pos, osp and ops)
- Efficient data querying with cheap merge joins
  (compare [WKB08])
Hexastore
Indexing Structure

Type 1 \rightarrow Type 2 \rightarrow Type 3

\begin{align*}
\text{Type} &= \{\text{subject, predicate, object}\} \\
\text{(compare [WBB08])}
\end{align*}
Tokyo Cabinet

- Key/value database
- Successor of DBM
- Space and time efficient
- Provides $B^+$-tree and hash table data structures
- Open source

(compare [10T10])
Implementation and Evaluation
Foundation

- Android provides an NDK to run native C/C++ code
- Tokyo Cabinet provides a Java API
- Porting Tokyo Cabinet and its Java API to Android NDK
Dictionary

- Mapping of RDF-nodes to 64bit IDs
- Implemented with two hash tables in Tokyo Cabinet
- Random hashing algorithm
Indexing Structure

Similar to Hexastore

- 6 indices (one for each permutation)
- Implemented with $B^+$-Trees in Tokyo Cabinet
API

Classes

Node represents a RDF-node

Triple represents a RDF-statement, consists of three nodes

TriplePlace abstracts the underlaying store functions
API
Methods of TriplePlace

- **getNode()** Node factory method
- **addTriple()** adds a triple to the indexing structure
- **query()** retrieves a list of triples which match a given pattern
- **removeTriple()** not yet implemented
Benchmark
Loading 2000 Triples

Memory Footprint:
- 10,000KB (Androjena)
- 4,000KB (TriplePlace)

Speed:
- 4.4s (Androjena)
- 5.5s (TriplePlace)
Conclusion
Conclusion

- Successfully designed a Triple Store for Android
- Economically using the limited main memory
- Moderate speed
- Only $\sim 800\text{KB}$ storage in the final APK
Future Work

- Evaluation of complex graph pattern queries
- Complete evaluation and comparison of TriplePlace with Androjena
- Usage in the Mobile Social Semantic Web Client [Arn10]
Thank you!

The source code of TriplePlace is available on github (white-gecko) and licensed under the terms of the GPL.


http://lod-cloud.net/.

Cathrin Weiss, Abraham Bernstein, and Sandro Boccuzzo.  

Cathrin Weiss, Panagiotis Karras, and Abraham Bernstein.  
*Hexastore: Sextuple Indexing for Semantic Web Data Management.*  
In *Proc. of the 34th Intl Conf. on Very Large Data Bases (VLDB)*, February 2008.