Compressed Indexes for Fast Search of Semantic Data

(TKDE poster presentation)

Raffaele Perego
ISTI-CNR
Pisa, Italy

Giulio Ermanno Pibiri
ISTI-CNR
Pisa, Italy

Rossano Venturini
The University of Pisa
Pisa, Italy

37-th IEEE International Conference on Data Engineering (ICDE)

April 2021

Resource Description Framework (RDF)

“RDF is a standard model for data interchange on the Web.”
https://www.w3.org/RDF

Statements are encoded with **triples**:
Subject (**S**) - Predicate (**P**) - Object (**O**)

Resource Description Framework (RDF)

"RDF is a standard model for data interchange on the Web."
https://www.w3.org/RDF

Statements are encoded with **triples**:
Subject (**S**) - Predicate (**P**) - Object (**O**)

Storage space is an issue:
compression is mandatory.

How to support triple selection patterns (with wildcards) **efficiently**?

<Bob Smith> <knows> <???>

<???> <???> John Doe

<Bob Smith> <???> <Sara Parker>

1 wildcard:
SP?
S?O
?PO

2 wildcards:
S??
?P?
??O

3 wildcards:
???

0 wildcard:
SPO

The Permuted Trie Index

Map URI strings to integers to reduce space requirements:
we deal with datasets of integer triples.

The Permuted Trie Index

Map URI strings to integers to reduce space requirements:
we deal with datasets of integer triples.

S P O
S P ?
S ? ?
? ? ?

→ **S-P-O** order

? P O
? P ?

→ **P-O-S** order

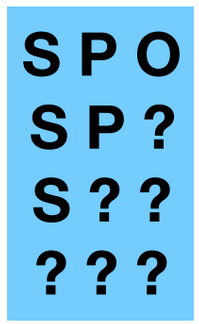
S ? O
? ? O

→ **O-S-P** order

Store an **integer trie**
data structure
for each permutation.

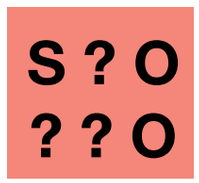
The Permuted Trie Index

Map URI strings to integers to reduce space requirements:
we deal with datasets of integer triples.



→ **S-P-O** order

→ **P-O-S** order



→ **O-S-P** order

Store an **integer trie**
data structure
for each permutation.

- **Common prefixes** are encoded once.
- Two integer **sequences** per level (nodes and pointers).

Allows effective compression

- Symmetrically support **all** selection patterns with 1 and 2 wildcards.
- **Cache-friendly** memory layout.

Fast retrieval

Refinements

1. Cross Compression

2. Permutation Elimination

Experiments: overall comparison

	Index	DBLP	Geonames	DBpedia	Freebase
		bits/triple	bits/triple	bits/triple	bits/triple
	2Tp	**51.99**	**48.98**	**54.14**	**52.17**
	HDT-FoQ	76.89 (+32%)	88.73 (+45%)	76.66 (+29%)	83.11 (+37%)
	TripleBit	125.10 (+58%)	120.03 (+59%)	130.07 (+58%)	—
		ns/triple	ns/triple	ns/triple	ns/triple
	2Tp	5	5	5	5
?PO	HDT-FoQ	12 (2.4×)	13 (2.6×)	14 (2.8×)	13 (2.6×)
	TripleBit	15 (3.0×)	13 (2.6×)	14 (2.8×)	—
	2Tp	**445**	**490**	**692**	**3736**
S?O	HDT-FoQ	1789 (4.0×)	2097 (4.3×)	3010 (4.3×)	0.7×10^7 (2057×)
	TripleBit	11872 (26.7×)	13008 (26.5×)	18023 (26.0×)	—
	2Tp	**197**	**347**	**11**	**3**
SP?	HDT-FoQ	640 (3.2×)	897 (2.6×)	30 (2.7×)	9 (3.0×)
	TripleBit	1222 (6.2×)	927 (2.7×)	42 (3.8×)	—
	2Tp	**28**	**40**	**10**	**3**
S??	HDT-FoQ	110 (3.9×)	154 (3.9×)	29 (2.9×)	9 (3.0×)
	TripleBit	2275 (81.2×)	3261 (81.5×)	490 (49.0×)	—
	2Tp	**9**	**8**	**6**	**4**
?P?	HDT-FoQ	108 (12.0×)	173 (21.6×)	32 (5.3×)	41 (6.8×)
	TripleBit	28 (3.1×)	28 (3.5×)	40 (6.7×)	—
	2Tp	**5**	**5**	**6**	**10**
??O	HDT-FoQ	17 (3.4×)	17 (3.4×)	18 (3.0×)	18 (1.8×)
	TripleBit	24 (4.8×)	60 (12.0×)	24 (4.0×)	—

Conclusions

The *triple indexing problem with pattern matching* can be solved efficiently in both time and space regards using a **permuted trie index**.

C++ code available at
https://github.com/jermp/rdf_indexes

Drop me a line at
giulio.ermanno.pibiri@isti.cnr.it
for questions and/or details!