DSAAS

DSAAS

A Cloud Service for Persistent Data Structures

Pierre le Roux, Steve Kroon and Willem Bester

Sunday 24th April, 2016

Stellenbosch University
http://cs.sun.ac.za/~kroon/dsaas

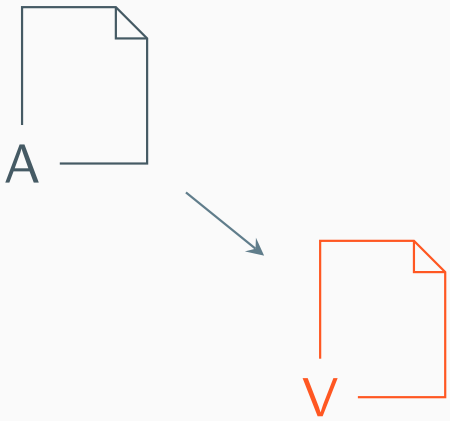
PROBLEM

Alice's structured data sets

Bob's structured data sets

PROBLEM

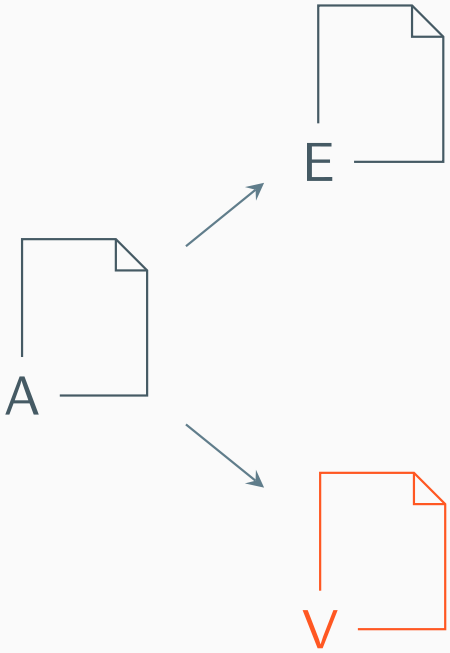
Alice's structured data sets



Bob's structured data sets

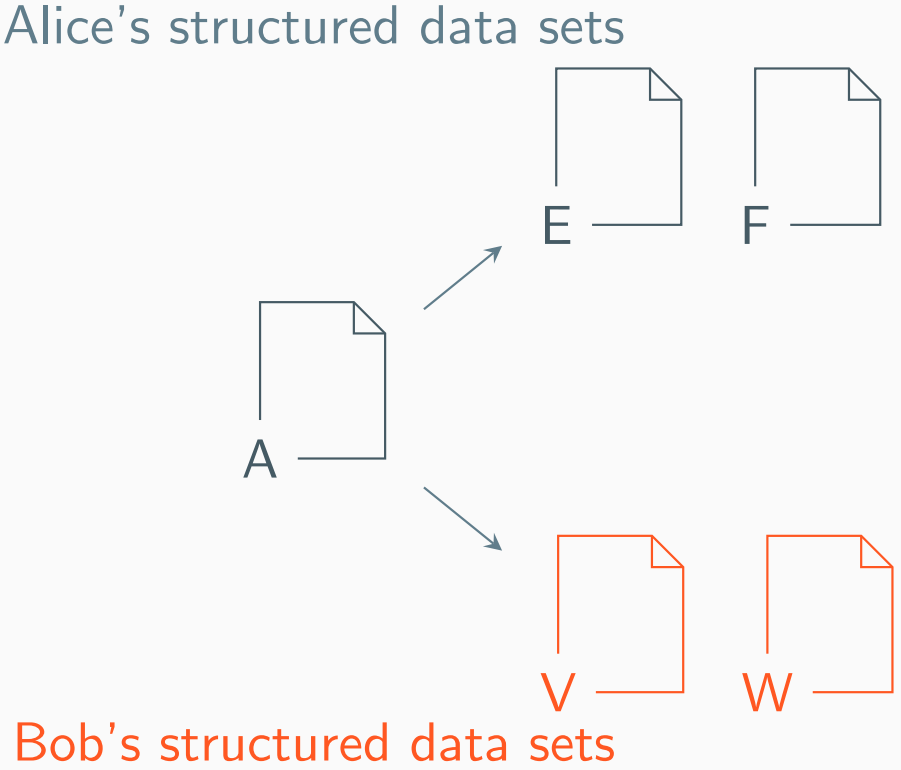
PROBLEM

Alice's structured data sets

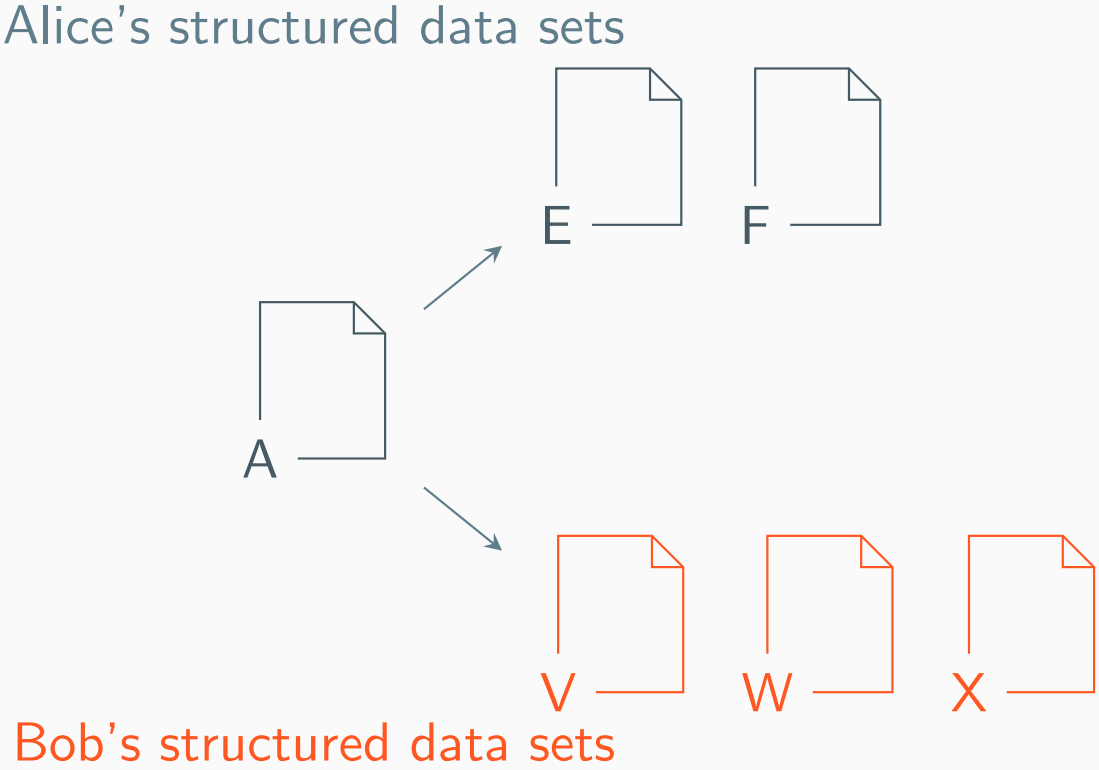


Bob's structured data sets

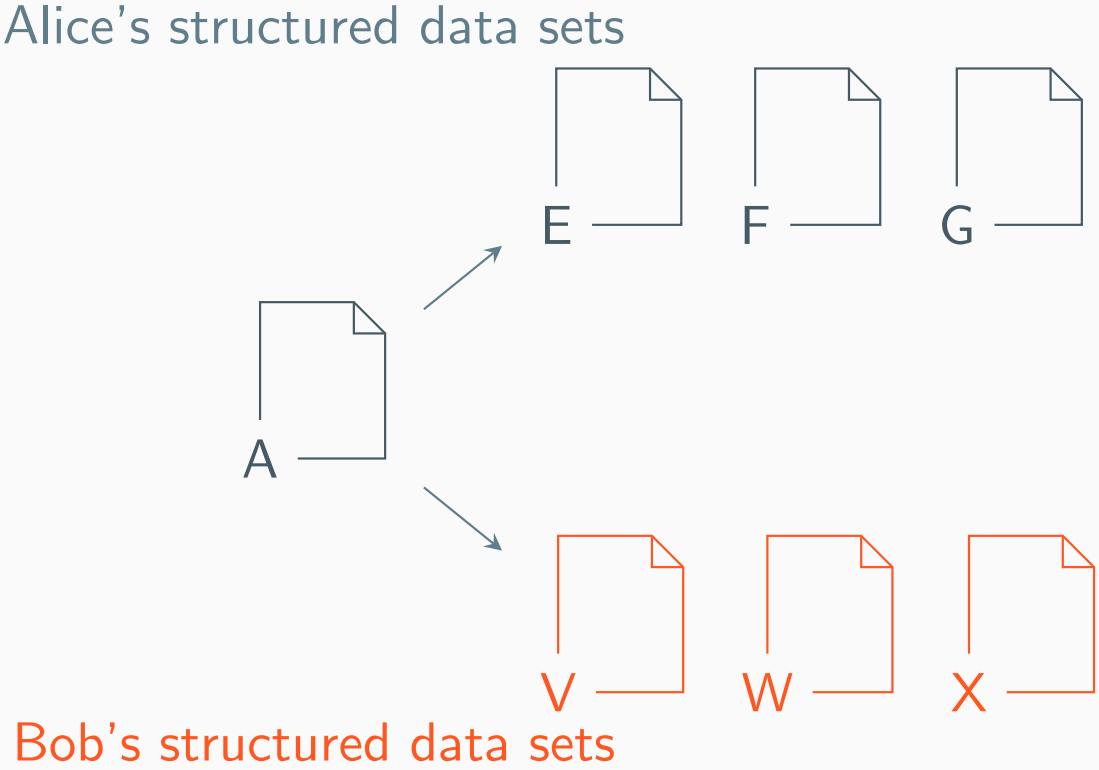
PROBLEM


Alice's structured data sets
E
F
A
V
W
Bob's structured data sets

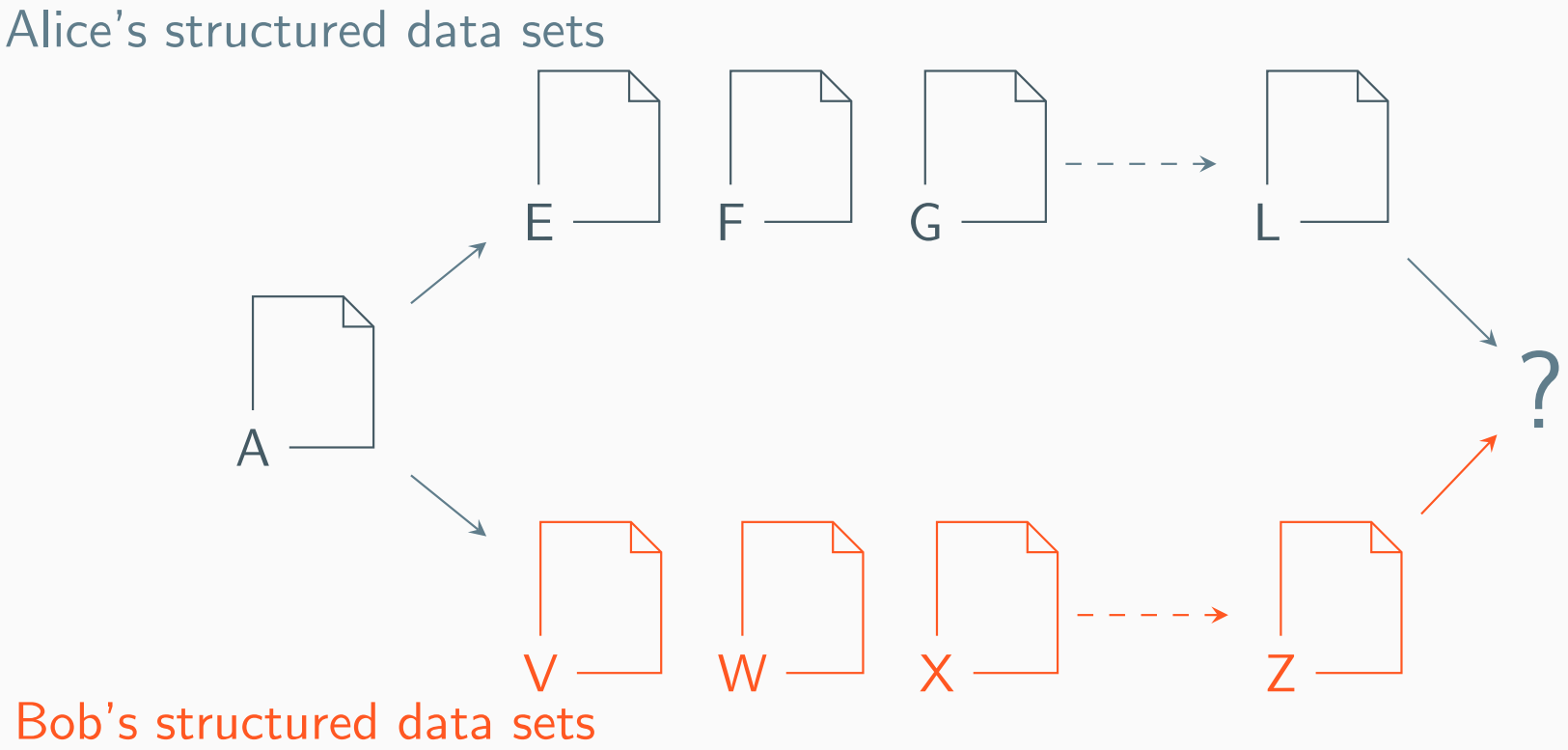
PROBLEM


Alice's structured data sets
E
F
A
V
W
X
Bob's structured data sets

PROBLEM


Alice's structured data sets
E
F
G
A
V
W
X
Bob's structured data sets

PROBLEM


Alice's structured data sets
E
F
G
L
A
?
V
W
X
Z
Bob's structured data sets

PROBLEM

Collaboration on structured data can be *difficult*, *time-consuming*, *error-prone* and *frustrating*.

OVERVIEW

A prototype *cloud* service for using automatically version controlled data structures.

(version, key) => value

OVERVIEW

A prototype *cloud* service for using automatically version controlled data structures.

(version, key) => value

Layers of Service

Language Bindings
API
Data Structures
Version Control
Data storage

EXAMPLE

http://dsaas.pbit.co.za/workbench/graph/pierre/SimpleFriendsGraph/

BACKGROUND

VERSION CONTROL

Ephemeral vs Persistent Data Structures

VERSION CONTROL

Ephemeral vs Persistent Data Structures

Types of persistence:

VERSION CONTROL

Ephemeral vs Persistent Data Structures

Types of persistence:

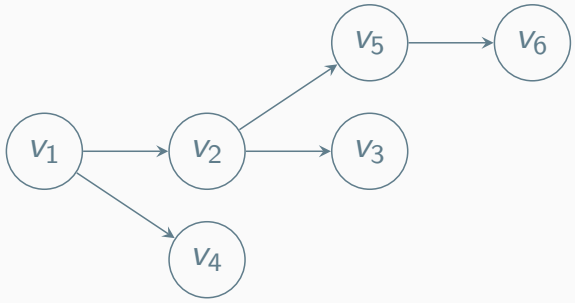
Partial Persistence

Ephemeral vs Persistent Data Structures

Types of persistence:

Partial Persistence

Full Persistence



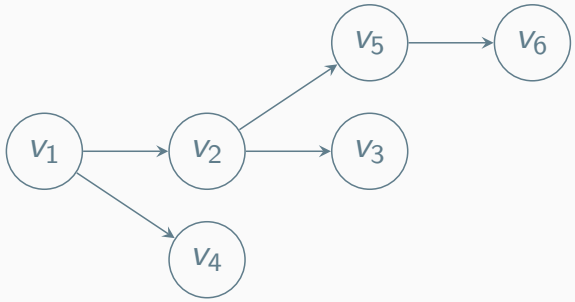
VERSION CONTROL

Ephemeral vs Persistent Data Structures

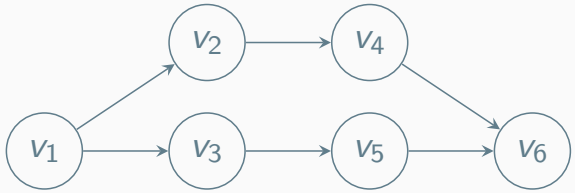
Types of persistence:

Partial Persistence

Full Persistence

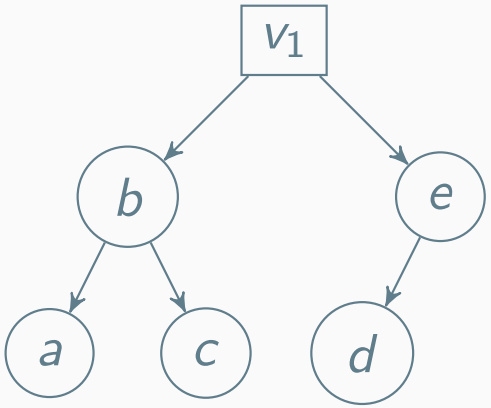


Confluent Persistence



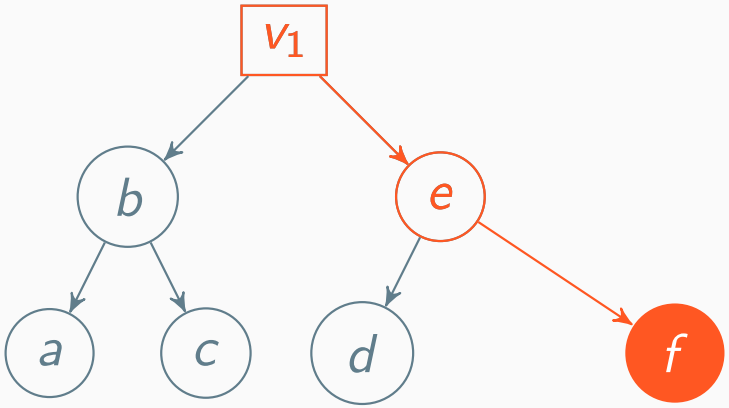
PATH-COPYING

Achieve full persistence using a technique called *path-copying*



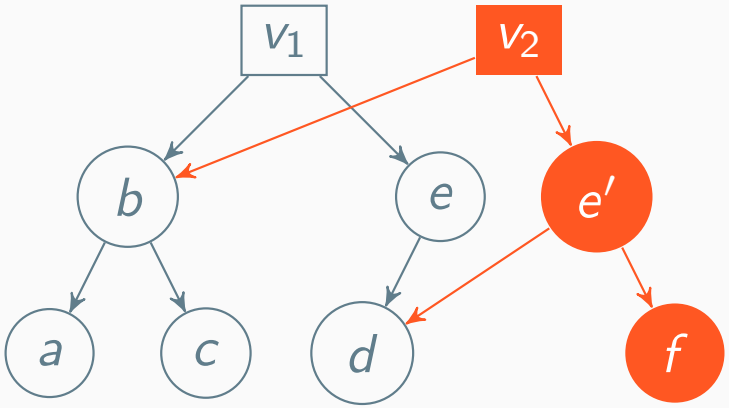
PATH-COPYING

Achieve full persistence using a technique called *path-copying*



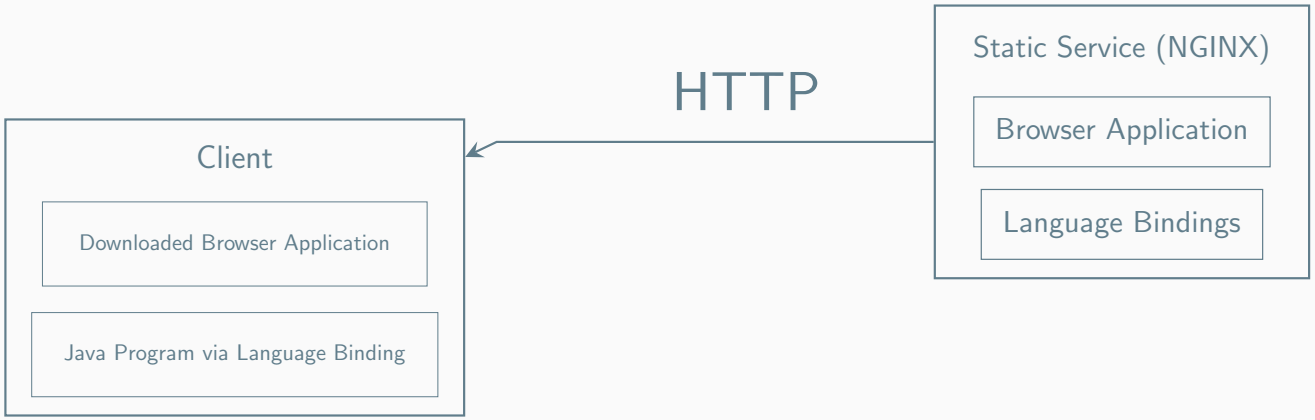
PATH-COPYING

Achieve full persistence using a technique called *path-copying*

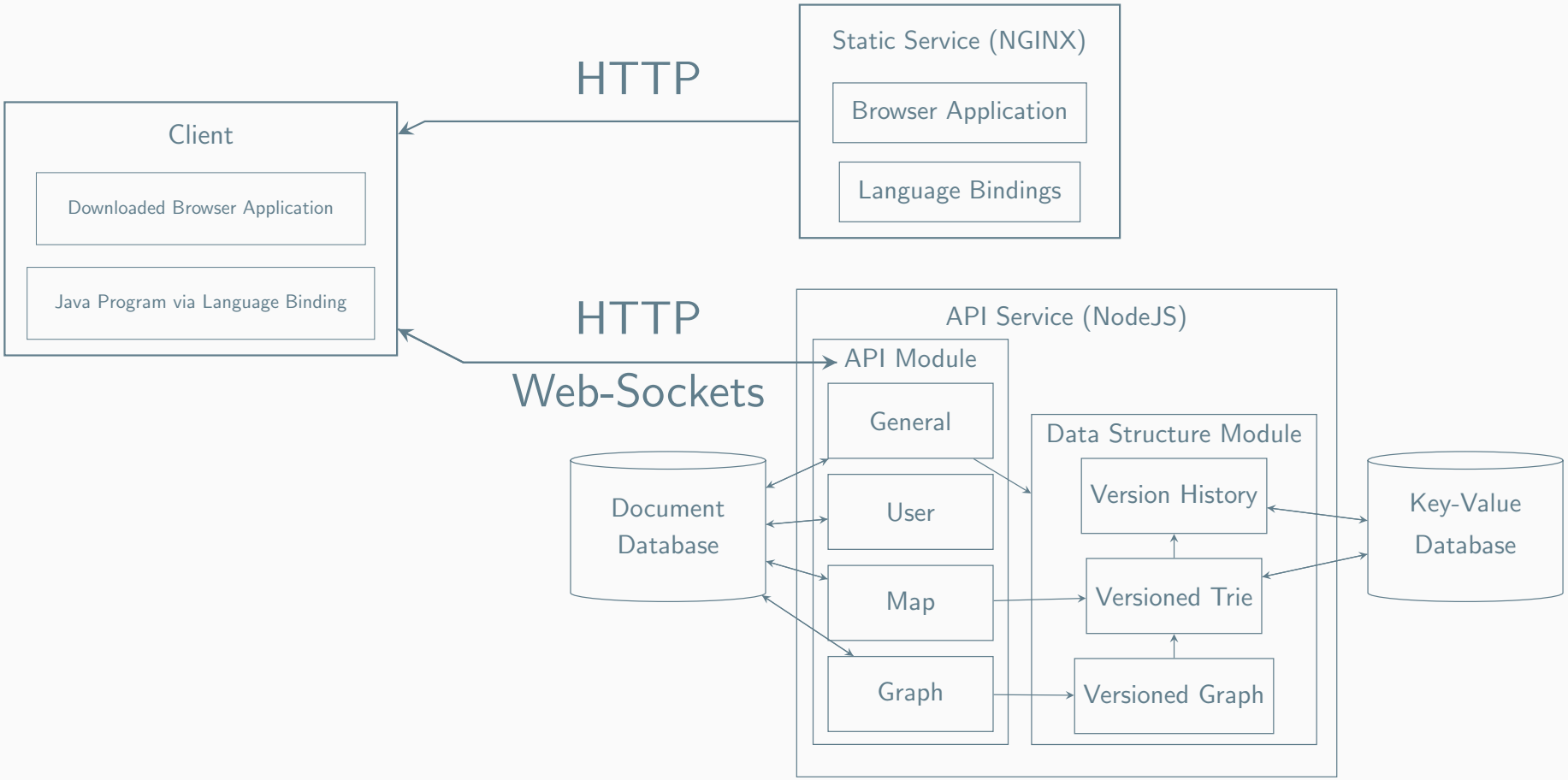


DEVELOPMENT

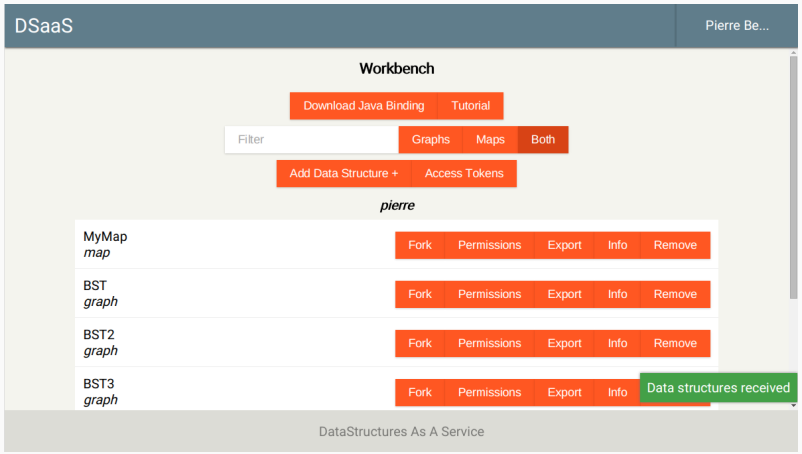
ARCHITECTURE OVERVIEW

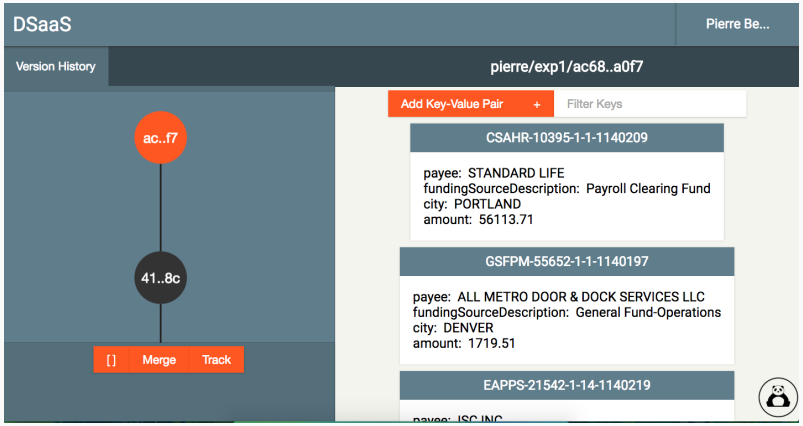


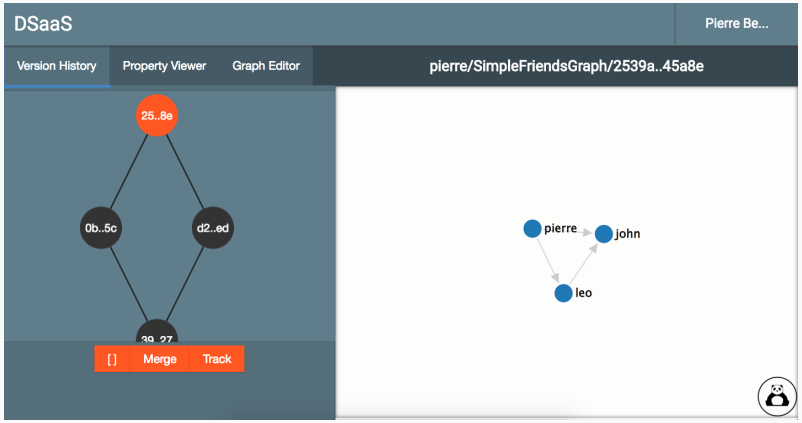
ARCHITECTURE OVERVIEW

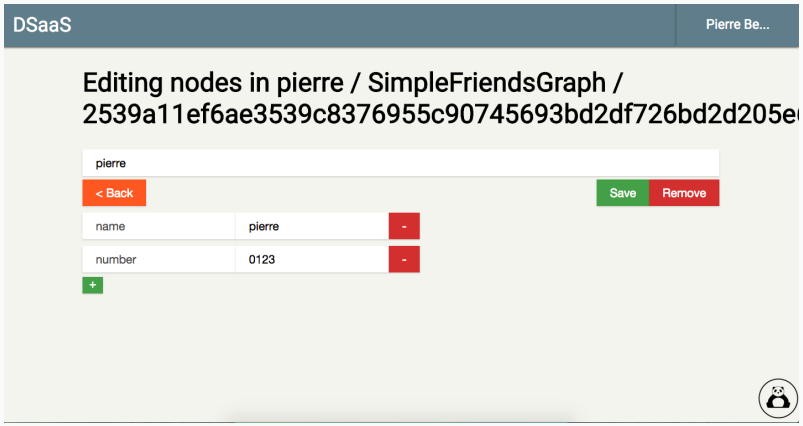


BROWSER APPLICATION

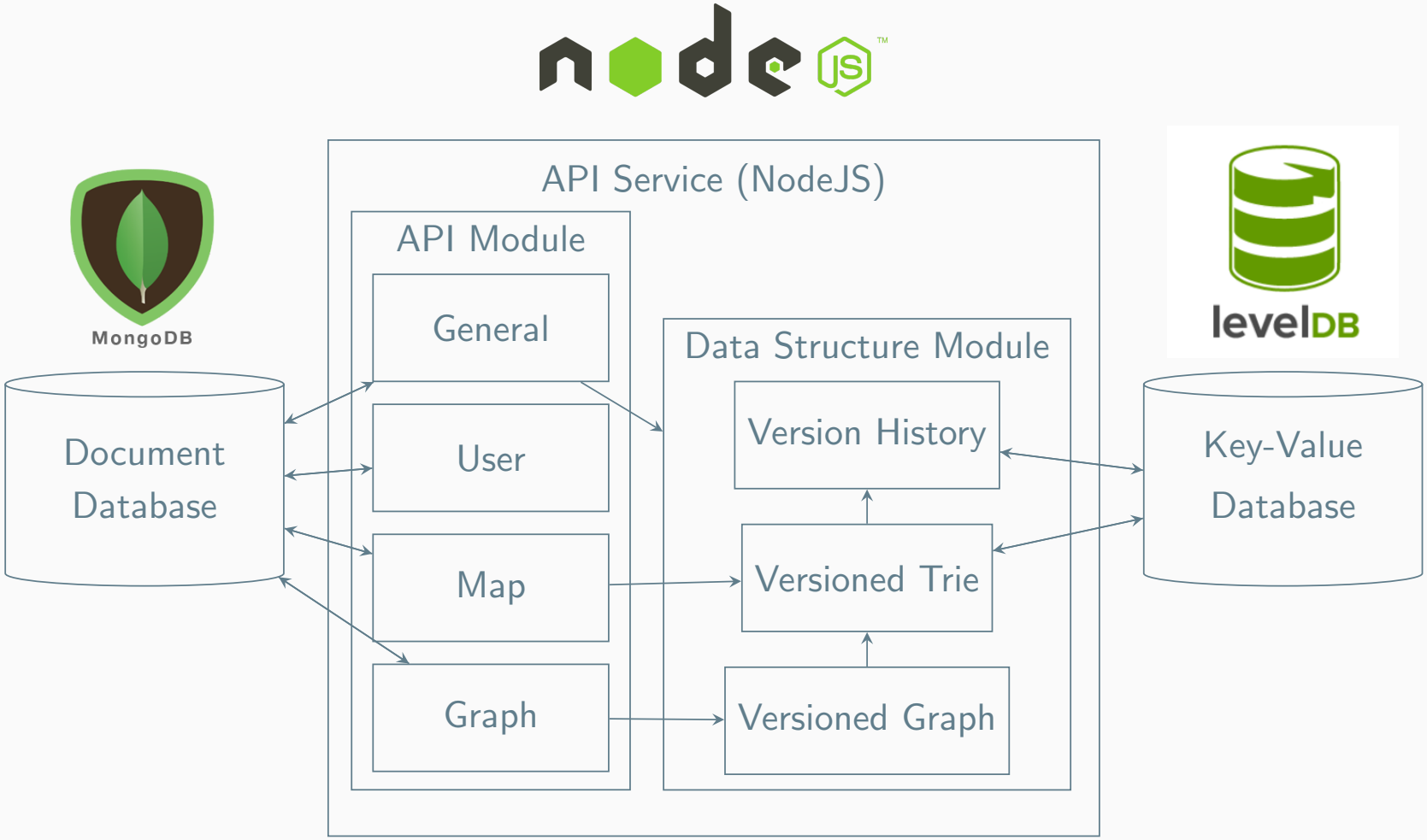








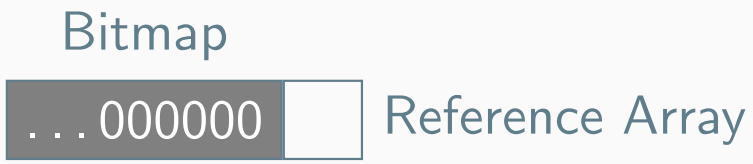
BACK END


MongoDB
Document Database
API Service (NodeJS)
API Module
General
User
Map
Graph
Data Structure Module
Version History
Versioned Trie
Versioned Graph
levelDB
Key-Value Database

VERSIONED TRIE

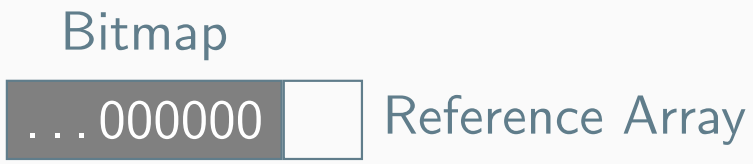
- Based on the Hash Array Mapped Trie (HAMT).
- Implemented on storage instead of in memory.
- Three-way merge operation for confluent persistence.
- Detects transpositions using Zobrist hashing.

HASH ARRAY MAPPED TRIE (HAMT)



An example of an HAMT. The white cells represent the array of references to key–value pairs stored in this trie.

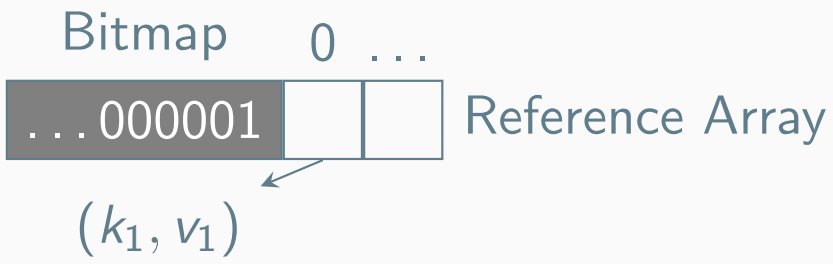
HASH ARRAY MAPPED TRIE (HAMT)



$h(k_1) = 00000\ldots$

An example of an HAMT. The white cells represent the array of references to key–value pairs stored in this trie.

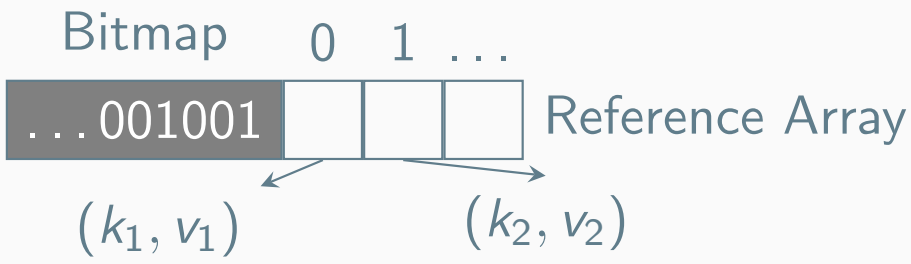
HASH ARRAY MAPPED TRIE (HAMT)



$h(k_1) = 00000\ldots$

An example of an HAMT. The white cells represent the array of references to key–value pairs stored in this trie.

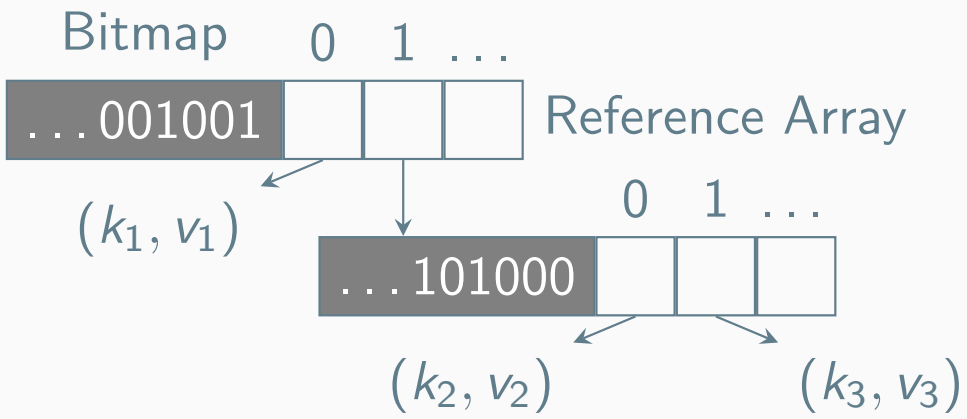
HASH ARRAY MAPPED TRIE (HAMT)



$$h(k_1) = 00000\ldots$$
$$h(k_2) = 00011\ 00011\ldots$$

An example of an HAMT. The white cells represent the array of references to key–value pairs stored in this trie.

HASH ARRAY MAPPED TRIE (HAMT)

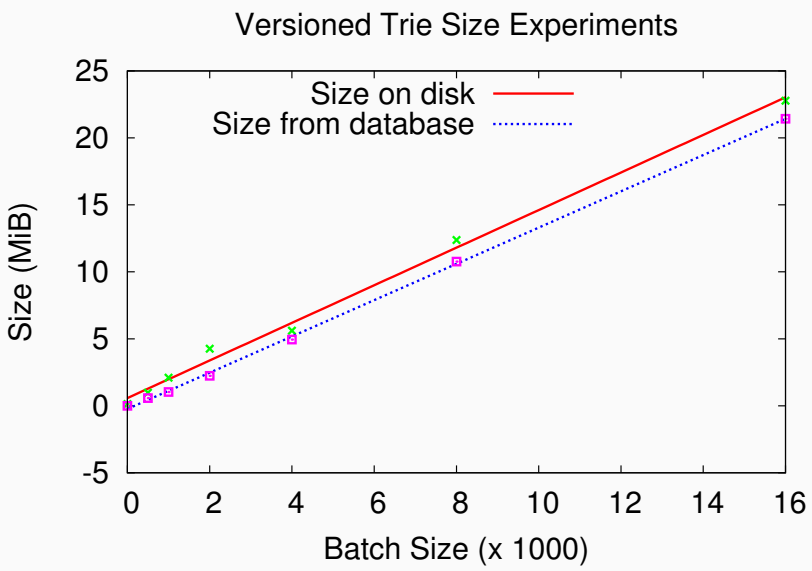


$$h(k_1) = 00000\ldots$$
$$h(k_2) = 00011\ 00011\ldots$$
$$h(k_3) = 00011\ 00101\ldots$$

An example of an HAMT. The white cells represent the array of references to key–value pairs stored in this trie.

EVALUATION

EVALUATION



Insertion: 1 ≈ adding 12 ephemeral data items

Removal: 1 ≈ adding 10 ephemeral data items

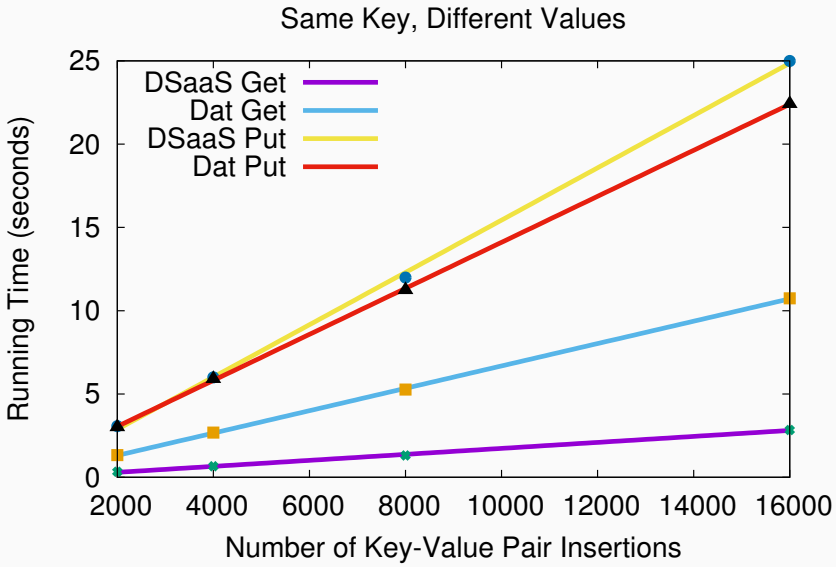
Merging: 16 000 x 16 000 elements ≈ increase of 650 KiB (≈ 6000 ephemeral items)

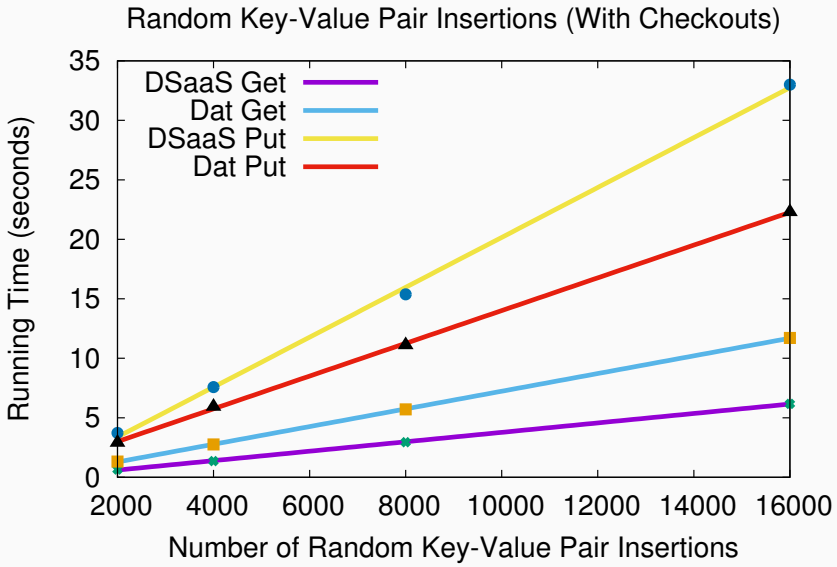
LATENCY

Remote Server (Library Binding)	206
Localhost (Library Binding)	7.6
Core (JavaScript)	2

The latency (in ms) for the *put* operation using the library binding to connect to a remote server and the localhost, and using JavaScript to test it on the core system.

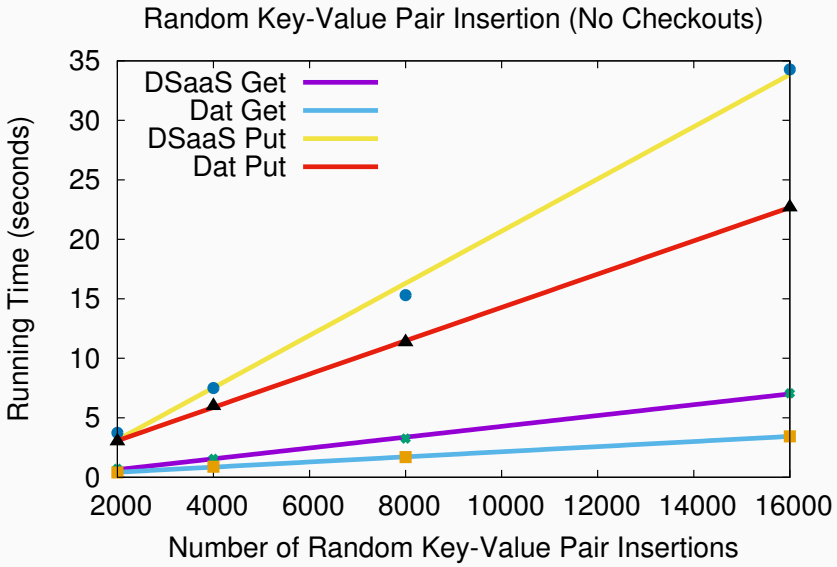
COMPARED TO DAT





Dat is available at `http://dat-data.com/`

COMPARED TO DAT



Dat is available at `http://dat-data.com/`

CONCLUSION

Questions?