Bloc: Binary Large Objects with Concurrency

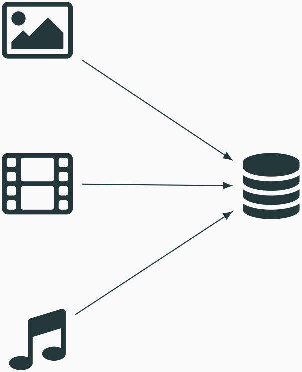
Anshu Avinash
Thesis supervisor: Piyush Kurur

June 15, 2015

Department of CSE, IIT Kanpur

What is a blob?

BLOB stands for “**B**inary **L**arge **OB**ject”.



How to store blobs? I

Store the entire content of the blob in the database.

- PostgreSQL breaks large objects into “chunks” and these chunks are stored in rows in the database.
- MongoDB also divides large objects using the GridFS specification and stores them in the “chunks” collection.

How to store blobs? II

Store the blob in a file and store the file name in the database.

- Number and size of blobs are now limited only by the file system.
- Requires to develop an interface to keep track of all the blobs of a database.

Here comes Bloc!

We provide a library which will keep track of all the blobs stored in a database.

Design

- Our design is inspired from the maildir format[1].

[1] Daniel J Bernstein. *Using maildir format.* 1995.

Design

- Our design is inspired from the maildir format[1].
- All the blobs of a database are stored inside a directory which we also call a BlobStore.

[1] Daniel J Bernstein. *Using maildir format.* 1995.

Design

- Our design is inspired from the maildir format[1].
- All the blobs of a database are stored inside a directory which we also call a BlobStore.
- Each blob is stored as a different file inside the BlobStore.

[1] Daniel J Bernstein. *Using maildir format.* 1995.

Design

- Our design is inspired from the maildir format[1].
- All the blobs of a database are stored inside a directory which we also call a BlobStore.
- Each blob is stored as a different file inside the BlobStore.
- We provide an incremental interface for writing to a blob as well as reading from a blob.

[1] Daniel J Bernstein. *Using maildir format.* 1995.

Creating a new blob

```
openBlobStore :: FilePath -> IO BlobStore
newBlob       :: BlobStore -> IO WriteContext
```

```
blobstore
└── tmp
    ├── 1a4c5091-1295-4c9c-b8d3-8e6123a51b41
    └── 4af0ac03-0b79-43fd-a2b0-2cc7c62947bc
```

Adding contents to a blob

```
writePartial :: WriteContext
             -> Blob
             -> IO WriteContext
endWrite     :: WriteContext -> IO BlobId
```

```
📂 blobstore
├── 📂 tmp
│   └── 🗋 1a4c5091-1295-4c9c-b8d3-8e6123a51b41
└── 📂 curr
    └── 🗎 sha512-861844d6704e8573fec34d967e20b..
```

Reading from a blob

```
startRead   :: BlobId -> IO ReadContext
readPartial :: ReadContext -> Int -> IO Blob
skipBytes   :: ReadContext -> Integer -> IO ()
endRead     :: ReadContext -> IO ()
```

Deleting blobs

- A blob can be shared by multiple “records” of the database.

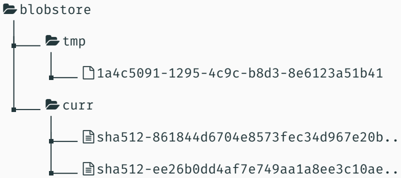
Deleting blobs

- A blob can be shared by multiple “records” of the database.
- We provide an interface for garbage collection of deleted blobs.

Starting a GC

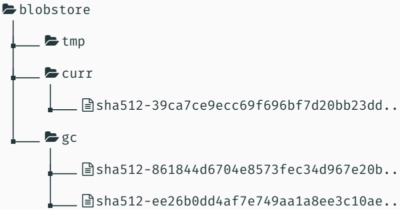
```haskell
startGC :: BlobStore -> IO ()
```

Before startGC



```
blobstore
├── tmp
│   └── 1a4c5091-1295-4c9c-b8d3-8e6123a51b41
└── curr
    ├── sha512-861844d6704e8573fec34d967e20b..
    └── sha512-ee26b0dd4af7e749aa1a8ee3c10ae..
```

After startGC

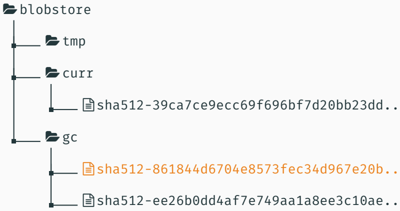


```
blobstore
├── tmp
├── curr
│   └── sha512-39ca7ce9ecc69f696bf7d20bb23dd..
└── gc
    ├── sha512-861844d6704e8573fec34d967e20b..
    └── sha512-ee26b0dd4af7e749aa1a8ee3c10ae..
```

Marking a blob as accessible

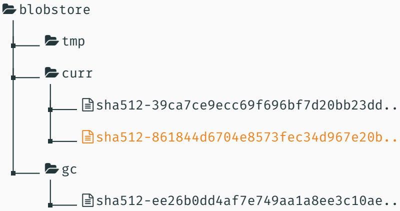
```
markAsAccessible :: BlobId -> IO ()
```

Before



```
blobstore
├── tmp
├── curr
│   └── sha512-39ca7ce9ecc69f696bf7d20bb23dd..
└── gc
    ├── sha512-861844d6704e8573fec34d967e20b..
    └── sha512-ee26b0dd4af7e749aa1a8ee3c10ae..
```

After

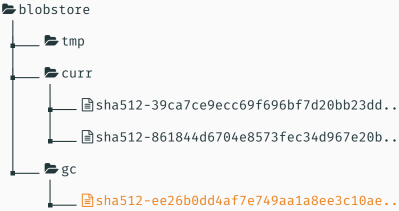


```
blobstore
├── tmp
├── curr
│   ├── sha512-39ca7ce9ecc69f696bf7d20bb23dd..
│   └── sha512-861844d6704e8573fec34d967e20b..
└── gc
    └── sha512-ee26b0dd4af7e749aa1a8ee3c10ae..
```

Ending a GC

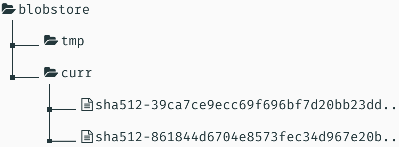
```haskell
endGC :: BlobStore -> Int -> IO ()
```

Before



```
blobstore
├── tmp
├── curr
│   ├── sha512-39ca7ce9ecc69f696bf7d20bb23dd..
│   └── sha512-861844d6704e8573fec34d967e20b..
└── gc
    └── sha512-ee26b0dd4af7e749aa1a8ee3c10ae..
```

After



```
blobstore
├── tmp
└── curr
    ├── sha512-39ca7ce9ecc69f696bf7d20bb23dd..
    └── sha512-861844d6704e8573fec34d967e20b..
```