The Main GeoGig Team

David Blasby
@Boundless

Hannah Bristol
@Boundless

Erik Merkle
@Boundless

Gabriel Roldan
@Boundless

Johnathan Garrett
@Prominent Edge
# GeoGig Releases

<table>
<thead>
<tr>
<th>Year</th>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016 Q4</td>
<td>1.0</td>
<td>First official LocationTech release! Core functionality complete.</td>
</tr>
<tr>
<td>2017 Q2</td>
<td>1.1</td>
<td>Optimization for direct GeoGig Querying. Spatial-with-attributes index.</td>
</tr>
<tr>
<td>2017 Q3</td>
<td>1.1.1</td>
<td>More optimization for GeoGig inside GeoServer GeoServer Clustering Support</td>
</tr>
</tbody>
</table>
1.1.x Series

GeoServer

GeoGig

PostgreSQL
1.1.x Series

GeoServer

GeoGig

PostgreSQL
1.1.x Series

Geoserver

GeoGig

Network Encoding

- 1.1 - twice as small as 1.0
- 1.1.1 - another 25% smaller

Almost 3* smaller!
1.1.x Series

GeoServer → GeoGig → Cache → PostgreSQL

In-Memory Cache
- lots of performance improvements!
- easy to configure
- view usage statistics

<table>
<thead>
<tr>
<th>Status</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of objects</td>
<td>872284</td>
<td>Approximate number of revision objects in the cache</td>
</tr>
<tr>
<td>Size in MB</td>
<td>475.483</td>
<td>Approximate cache size in MB</td>
</tr>
<tr>
<td>Maximum size in MB</td>
<td>2,730</td>
<td>Currently configured maximum cache size in MB</td>
</tr>
<tr>
<td>Maximum size %</td>
<td>0.75</td>
<td>Configured maximum cache size as a ratio of the JVM max heap memory</td>
</tr>
<tr>
<td>Absolute maximum in MB</td>
<td>3,276.9</td>
<td>Absolute maximum size the cache can be configured to</td>
</tr>
<tr>
<td>Default cache size in MB</td>
<td>2,730.75</td>
<td>Default cache size as given by configuration parameters</td>
</tr>
<tr>
<td>Eviction count</td>
<td>0</td>
<td>Number of times an entry has been evicted</td>
</tr>
<tr>
<td>Hits count</td>
<td>1210228</td>
<td>Number of times lookup methods have returned a cached value</td>
</tr>
<tr>
<td>Hits rate</td>
<td>0.577</td>
<td>Ratio of cache requests which were hits</td>
</tr>
<tr>
<td>Miss count</td>
<td>891956</td>
<td>Number of times a cache lookup resulted in a non cached value</td>
</tr>
<tr>
<td>Miss rate</td>
<td>0.423</td>
<td>Ratio of cache requests which were misses</td>
</tr>
</tbody>
</table>

Refresh  Clear cache
1.1.x Series

Rendering Performance Improvements
- took a deep look at the rendering process
- made many optimizations
Clustered GeoServer Deployments

- made sure GeoGig functioned with GeoServer’s Resource API
- watch out for the size of the memory cache
1.1.x Series

Spatial-With-Attributes Index
- Biggest task in 1.1.x
- **Huge** performance improvements
GeoGig holds most of its information in two places
• A revision-sharing hash (Merkle) index tree
• Actual feature data
When you want to Query features in a revision;

a) Scan through the tree
b) Request the features
Quad-Tree

Includes sub-tree bounds

Features in Same Quad

Includes feature bounds
What about attribute Queries?
• Just draw *Highways* from a roads dataset
• Just draw a *time-slice* of data from a time-enabled layer
We add some of the feature’s attribute data to the leaf nodes. This results in a huge performance boost.
Spatial-with-Attributes Index

Include feature bounds and some attribute values

Optimize queries and retrieval
Spatial-with-Attributes Index

Revision A

Revision B

GeoServer
GeoGig
Cache

PostgreSQL
1.1.x - Where are we?

The performance improvements in this release open the door to many more use cases for GeoGig implementations. Boundless is excited to have a newly sharpened version of this powerful tool.

Zach Rouse - Boundless Exchange

GeoGig spatial indexing, specifically the flexibility to materialize any spatial or non-spatial attribute is a key performance enhancement yielding immediate value for spatio-temporal queries.

Clarence Davis - StoryScapes

1.0 - Core Functionality, LocationTech Graduation
1.1.1 - Performance meeting/beating expectations
Where are we going?

1.2  - Changes required for GeoServer 2.12 release (REST)
     - Improving performance of clone/push/pull/fetch
Where do we need to go next?
Where do we need to go next?

We really really want people to **actually** collaborate.
Where do we need to go next?

Need to make things easy to use!
Making Things Easier

Concentrating on User Experience
Making Things Easier

Fundamental GeoGig Ops

High-level Primitives

Workflows
Making Things Easier

GeoServer ➔ GeoGig Collaboration Engine ➔ QGIS

GeoGig
Enabling Collaboration

We want to hear how *you* could use GeoGig
geogig-dev@locationtech.org
Discover, Learn, Collaborate, and Share
With GIS Professionals
connect.boundlessgeo.com
Check out our booth #103
Q&A

Please come talk to us during the conference or on-line at geogig.org - geogig-dev@locationtech.org

David Blasby

Johnathan Garrett

Prominent Edge