HST on steroids

15 sep 2009

Ard Schrijvers • a.schrijvers@onehippo.com

The HST is very fast without caching

The HST is very fast without caching

Archetype 7.7.4 created site benchmark for the homepage /site/

On my machine (4 proc)

10.000 requests
50 threads

Requests per second: 1565.46 [#/sec] (mean)

The HST is very fast without caching

The HST scales almost linearly UP and OUT

The HST is very fast without caching

But pages can be slow

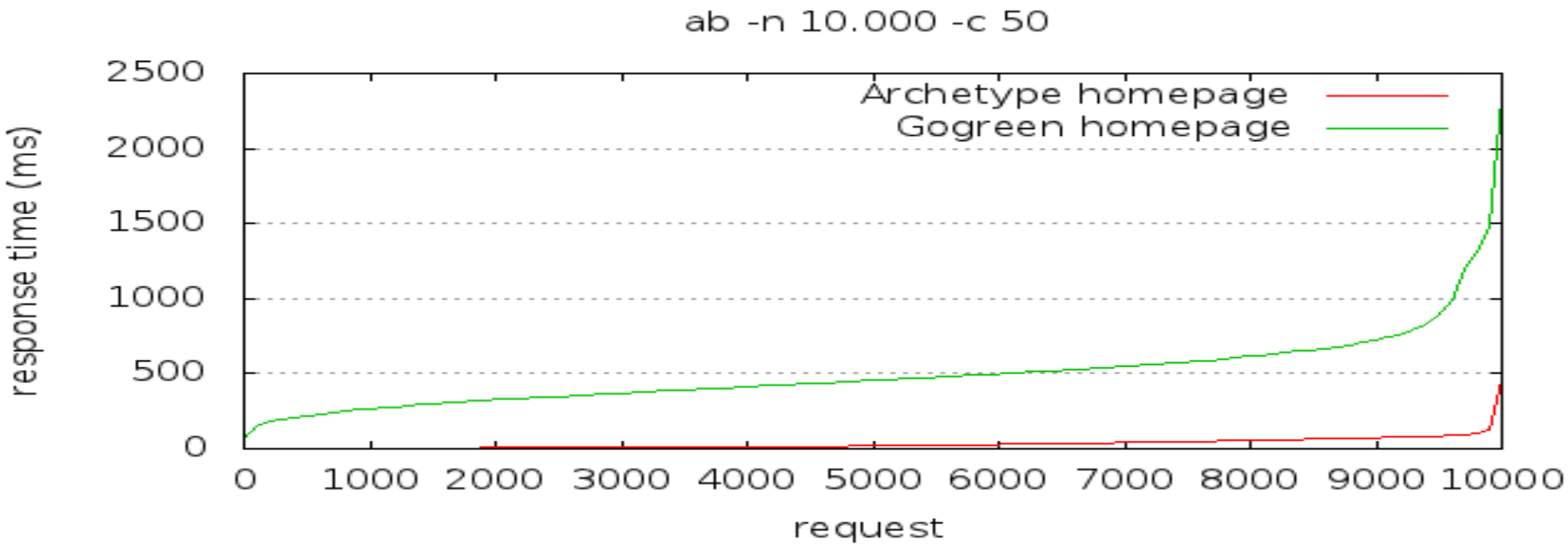
The HST is very fast without caching

Reasons:

1. Heavy searches
2. Searches with many hits
3. Many searches on a single page
4. Accessing several thousands of JCR Nodes for a page
5. Accessing many JCR Nodes that are not in the Jackrabbit bundle caches
6. Accessing external services (of course can always better be avoided)
7. Faceted navigation with large resultsets

The HST is very fast without caching

Archetype homepage compared to gogreen homepage



HIPPO

With targeting/personalization, we need to run faster

Targeting / Personalization

==

NO mod_cache
NO squid
NO any caching proxy

Howto?

Howto?

```
<dependency>
    <groupId>net.sf.ehcache</groupId>
    <artifactId>ehcache-web</artifactId>
</dependency>
```

Howto?

And rewrite ehcache

SimplePageCachingFilter

to

PageCachingValve

Howto?

Add it to the default site pipeline (also usable for REST though)

```
<property name="processingValves">
            <list>
              <ref bean="contextResolvingValve" />
              <ref bean="localizationValve" />
              <ref bean="securityValve" />
              <ref bean="subjectBasedSessionValve" />
              <ref bean="jcrSessionStatefulConcurrencyValve"/>
              <ref bean="siteMenusResolvingValve" />
              <ref bean="actionValve" />
              <ref bean="pageCachingValve"/>
              <ref bean="resourceServingValve" />
              <ref bean="componentRenderingValve" />
              <ref bean="aggregationValve" />
            </list>
</property>
```

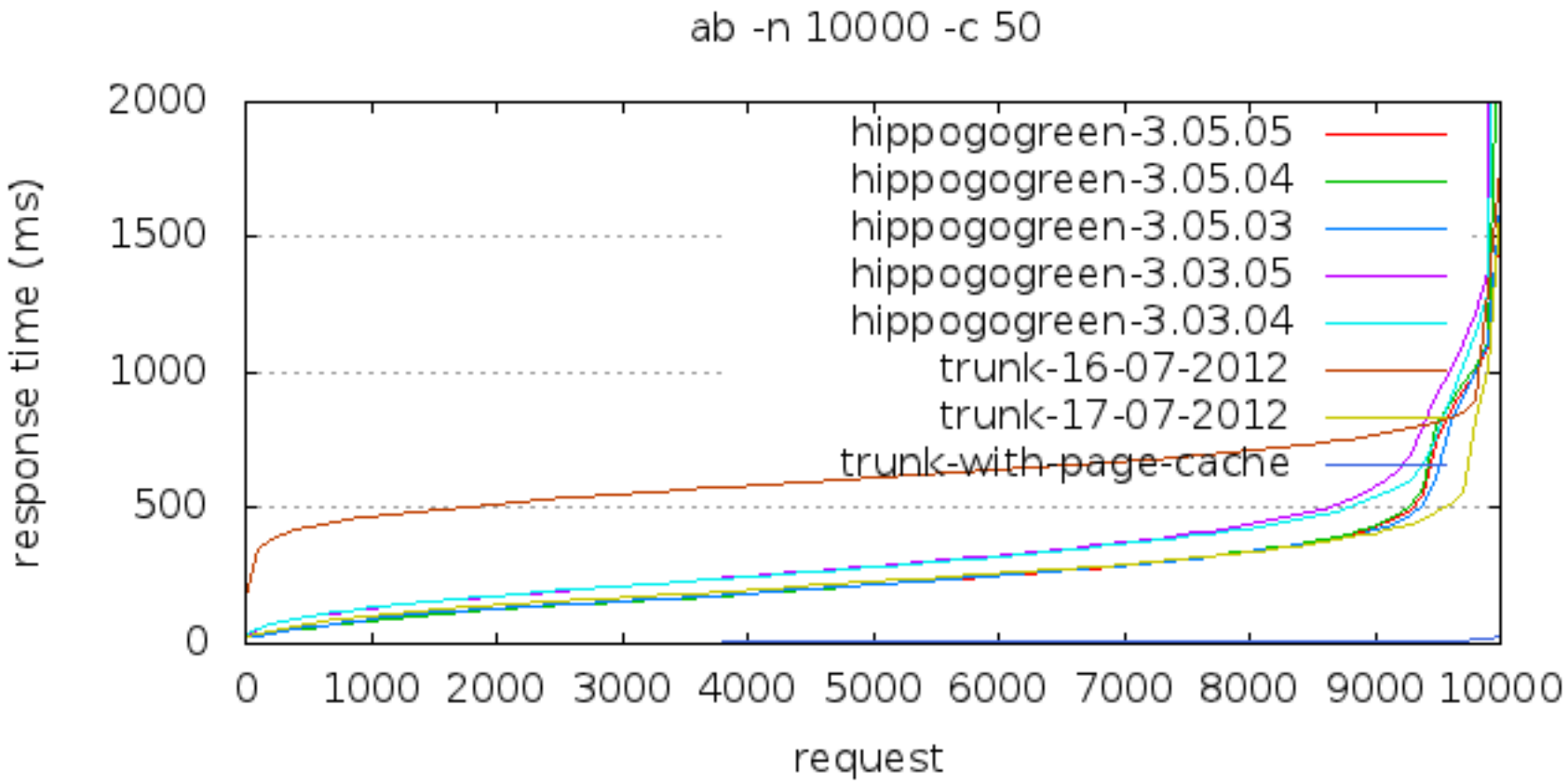
Howto?

Note that request matching (Host / mount / sitemap) is already done before pipelines are invoked

PageCachingValve

1. PageCachingValve is blocking : Stampeding herds less harmful
2. Optional : When page is invalidated, serve stale responses to all requests for same page until the single request we let through replaces the stale response

Result



HIPPO

Result

Gogreen homepage

10.000 requests
50 threads

Requests per second: 8921.19 [#/sec] (mean)

And with targeting / personalization?

```
<property name="initializationValves">
    <list>
      <ref bean="initializationValve" />
      <ref bean="cmsSecurityValve"/>
      <bean class="com.onehippo.cms7.behavioral.core.container.BehavioralUpdateValve"/>
    </list>
  </property>
  <property name="processingValves">
    <list>
      <ref bean="contextResolvingValve" />
      <ref bean="localizationValve" />
      <ref bean="securityValve" />
      <ref bean="subjectBasedSessionValve" />
      <ref bean="jcrSessionStatefulConcurrencyValve"/>
      <ref bean="siteMenusResolvingValve" />
      <ref bean="actionValve" />
      <ref bean="pageCachingValve"/>
      <ref bean="resourceServingValve" />
      <ref bean="componentRenderingValve" />
      <ref bean="aggregationValve" />
    </list>
  </property>
  <property name="cleanupValves">
    <list>
      <ref bean="cleanupValve" />
    </list>
  </property>
```

And with targeting / personalization?

During the BehavioralUpdateValve

1. Visitor data is harvested by targeting providers : Very lightweight
2. For the HST Component tree belonging to request, all different Persona's are SCORED by targeting engine : very lightweight
3. The ScorePersona[] array concatenated ==> Part of the page cache key: very lightweight

C'est tout

What about uncacheable components?

We could opt for

1. By default HST components are cacheable unless marked as uncacheable --> entire page becomes uncacheable when one component is uncacheable
2. Mark sitemap items as cacheable or uncacheable

Open for discussion

Development

Follow https://issues.onehippo.com/browse/HSTTWO-2215