

SumanSaurabh GSoC2015Nutch

Goal : Move Nutch 2.x to Hadoop 2.X from existing 1.x codebase.

The following page is a proposal for GSoC 2015 related to issue [Nutch-1936](#)

Introduction

- **1) About Nutch:**
- [Apache Nutch](#) is a flexible and powerful open source tool for web crawling. It builds on [Apache Solr](#) and comes with an integration of the highly popular [Apache Hadoop](#). Whole-web crawling is designed to handle very large crawls which may take weeks to complete, running on multiple machines. This also permits more control over the crawl process, and incremental crawling. It is important to note that whole web crawling does not necessarily mean crawling the entire world wide web. We can limit a whole web crawl to just a list of the URLs we want to crawl.
- **2) Basic Nutch Features:**
 - Runs on top of **Hadoop**
 - Scalable: billions of pages possible
 - Some overhead (if scale is not a requirement)
 - Not ideal for low latency
 - Customizable / extensible plug-in architecture * Pluggable protocols (document access)
 - URL filters + normalizers
 - Parsing: document formats + meta data extraction
 - Indexing back-ends
 - Mostly used to feed a search index
- **4) Nutch Work-flow?** (https://sites.google.com/site/nutch1936/_/rsrc/1427176500763/home/introduction/Nutch_Overview.png)*
- **5) Nutch Workflow execution with Hadoop**
- Every step is implemented as one (or more) **MapReduce** job.
 - Inject, generate, fetch, parse, updatedb, invertlinks, index.
 - local mode
 - works out-of-the-box (bin package)
 - useful for testing and debugging
 - (pseudo-)distributed mode
 - parallelization, monitor crawls with [MapReduce](#) web UI.
 - recompile and deploy job file with configuration changes.
 - In basic terms:
 - Map-reduce indexing
 - Map() just assembles all parts of documents.
 - Reduce() performs text analysis + indexing.
 - Sends assembled documents to Solr or – adds to a local Lucene index Nutch.
 - Nutch runs in two modes; namely **local** and **deploy**. When run in local mode e.g. running Nutch in a single process on one machine, then we use Hadoop as a dependency. This may suit if we have a small site to crawl and index.
- ◦ Nutch is mostly used because of its capability to run on in deploy mode, within a Hadoop cluster. This gives the benefit of a distributed file system (HDFS) and [MapReduce](#) processing style*.*
- *
- *
- **6) Why Hadoop 2.x over 1.x ?**
- ◦ The major difference between Hadoop 1.x and 2.x is the computation platform they use. (https://sites.google.com/site/nutch1936/_/rsrc/1427213159891/home/introduction/yarn.png)
- 1.x uses MRv1 whereas 2.x uses MRv2(aka YARN). MRv1: Master -> JobTracker
 - Slave -> TaskTracker
 - MRv2: Master -> Resource Manager
 - Slave -> Node Manager
 - And there is Application Specific Application Master .**Problems with 1.X 1.**
- ◦ Single Point of Failure for Name Node
- ◦ No Horizontal Scaling - V1
- ◦ Impossible to run Non Map Reduce tools because of tight coupling of JobTracker + MR
- ◦ Does not support Multi-tenancy
- ◦ Job Tracker overburdened because of too much work.
- The functionality of the JobTracker in 1.x split into 2 components :- **Application Specific Application Master** and **Global Resource Manager**. MRv2 introduced a concept of 'container'. Container is nothing but bunch of resources such as 'x amount of memory, y number of cores'.
- Allocating 'containers' for tasks is done by Resource Manager and tasks are actually launched by Application Master in the allocated containers.
- As a result,

- There are no more map/reduce dedicated slots on each tasktracker, instead there is a 'container' allocated for each task in an application.
- Hadoop 2.x scales better.
- NNHA: Name Node High availability for avoiding Single point of failure.
- Hadoop 2.x is more general than Hadoop 1.x.
- V2 now supports application other MR as well . You can do real time processing using **Apache Storm**.

Methodology

Phase 1(Learning & Experimenting):

- **1.1) Explore Nutch Documentation:**

- Since I have less knowledge about Nutch codebase, I will likely cover Nutch documentation [1].

- **1.2) Workspace Setup:**

- Nutch workspace it built on Ant+Ivy. I have experience with Ant build framework, so workspace setup would be relatively easier. I have forked the Nutch codebase to my Git [2] and after successful completion I will provide the patch. Nutch dependency on Hadoop: *hadoop-core.1.x.jar* is changed in *Hadoop 2.x*

```
<dependency org="org.apache.hadoop" name="hadoop-core" rev="1.2.0" conf="*->default">
  <exclude org="hsqldb" name="hsqldb" />
  <exclude org="net.sf.kosmosfs" name="kfs" />
  <exclude org="net.java.dev.jets3t" name="jets3t" />
  <exclude org="org.eclipse.jdt" name="core" />
  <exclude org="org.mortbay.jetty" name="jsp-*" />
  <exclude org="ant" name="ant" />
</dependency>
```

- Following dependency needs to be added for Hadoop 2.6 support instead of above.

```
<dependency org="org.apache.hadoop" name="hadoop-common" rev="2.6.0" conf="*->default" />
<dependency org="org.apache.hadoop" name="hadoop-mapreduce-client-core" rev="2.6.0" conf="*->default" />
```

- Dependency *hadoop-test-1.2.0.jar* needs to be removed.

```
<dependency org="org.apache.hadoop" name="hadoop-test" rev="1.2.0" conf="test->default" />
```

- **1.3) Experimental setup with of Nutch with Hadoop and their result:**

- I have been using Hadoop 2.3 for my MapReduce application and while trying to setup Nutch 1.9 with Hadoop 2.3. I ran into following error:

```
Injector:
java.lang.UnsupportedOperationException: Not implemented by the !DistributedFileSystem !FileSystem
implementation
at org.apache.hadoop.fs.FileSystem.getScheme(!FileSystem.java:214)
at org.apache.hadoop.fs.FileSystem.loadFileSystems(!FileSystem.java:2365)
at org.apache.hadoop.fs.FileSystem.getFileSystemClass(!FileSystem.java:2375)
at org.apache.hadoop.fs.FileSystem.createFileSystem(!FileSystem.java:2392)
at org.apache.hadoop.fs.FileSystem.access$200(!FileSystem.java:89)
at org.apache.hadoop.fs.FileSystem$Cache.getInternal(!FileSystem.java:2431)
at org.apache.hadoop.fs.FileSystem$Cache.get(!FileSystem.java:2413)
at org.apache.hadoop.fs.FileSystem.get(!FileSystem.java:368)
at org.apache.hadoop.fs.FileSystem.get(!FileSystem.java:167)
at org.apache.nutch.crawl.Injector.inject(Injector.java:297)
at org.apache.nutch.crawl.Injector.run(Injector.java:380)
at org.apache.hadoop.util.ToolRunner.run(!ToolRunner.java:70)
at org.apache.nutch.crawl.Injector.main(Injector.java:370) .
```

- May be I will start looking at this point onwards?

Phase 2 (Coding):

- **2.1) Migrating from Hadoop 1.x to Hadoop 2.x**

- **Binary Compatibility' _*:**
- First, we ensure binary compatibility to the applications that use old_ 'mapred*' APIs. This means that applications which were built against MRv1 **mapred** APIs can run directly on YARN without recompilation, merely by pointing them to an Apache Hadoop 2.x cluster via configuration.
- **Source Compatibility:**
- One cannot ensure complete binary compatibility with the applications that use **mapreduce** APIs, as these APIs have evolved a lot since MRv1. In other words, users should recompile their applications that use **mapreduce** APIs against MRv2 jars. One notable binary incompatibility break is **Counter** in

```
Package: crawl
Class: CrawlDbUpdateUtil
```

. Tradeoffs between MRv1 Users and MRv2 Adopters:

- Unfortunately, maintaining binary compatibility for MRv1 applications may lead to binary incompatibility issues for early MRv2 adopters. Below is the list of **MapReduce** APIs which are incompatible with Hadoop 1.3.

```
org.apache.hadoop.mapreduce.Job#failTask <--> Return type changes from void to boolean
org.apache.hadoop.mapreduce.Job#killTask <--> Return type changes from void to boolean
org.apache.hadoop.mapreduce.Job#getTaskCompletionEvents <--> Return type changes from o.a.h.
mapred.TaskCompletionEvent to o.a.h.mapreduce.TaskCompletionEvent
```

- **2.2) Configuring and Running MRv2 Clusters**

- **Configuration Migration**

- Since MapReduce 1 functionality has been split into two components, MapReduce cluster configuration options have been split into YARN configuration options, which go in yarn-site.xml, and MapReduce configuration options, which go in mapred-site.xml.
- As JobTrackers and TaskTrackers no longer exist in MRv2, all configuration options pertaining to them no longer exist, although many have corresponding options for the ResourceManager, NodeManager, and JobHistoryServer.
- A minimal configuration required to run MRv2 jobs on YARN is **yarn-site.xml configuration**. Detailed configuration for *conf/mapred-site.xml*, *conf/core-site.xml*, *conf/hdfs-site.xml* will remain same.
- Configuration for *conf/yarn-site.xml* is:

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

2.3) Summary of Configuration Changes which I have observed

- **1) JobTracker Properties and ResourceManager Equivalents .**

```
mapred.job.tracker to yarn.resourcemanager.hostname
Package: crawl
Class: Generator
```

- **2) MRv1 Properties that have no MRv2 Equivalents .**

```
mapred.temp.dir has no MRV2 equivalent
Package: crawl
Class: DeduplicationJob
```

Phase 3 (Documentation):

1. Documentation leading to the detailed description of migration of Hadoop framework in Apache Nutch.
2. Detailed guide for setting up Hadoop 2.x on Nutch 2.x.

Timeline

_Phase 1: _ * *

* _ '27 March- 20 April: *I will acquaint myself comprehensively with Nutch Documentation and Hadoop Framework. Experimenting with Nutch by simple web crawling techniques will expose myself completely to Apache Nutch.

- My end semester exam is scheduled from 26th April to 2nd May. My 4 years B. Tech tenure will be over and will have complete May, June and July to dedicate to Nutch.
- - 3rd May - 30th of June: * Simple experimentation would have given me detailed insight. Further I would have contributed to the issue related to Nutch-1219.
- **Time to submit First Report**

Phase 2: _ * *

_ '1st June - 25th June: Coding work would be half over, Mid- Term evaluation report submission. Discussion with mentor regarding improvement.



- **Time to submit Second Report**
- **26th June - 25th July:** Coding work would complete and time for System Testing and Bug Fixing that will come up in the project.

Phase 3:

- **1st August - 7th August:** Documentation of the project and preparing tutorial guide for setting up Hadoop 2.x on Nutch 2.x.
- _ **8th August - 15th August**_: Scrubbing the documentation and other reports to make it more readable and appropriate. Final evaluation report submitted to Google.
- **Time to submit Final Report**
- *

References:

- [1] <http://wiki.apache.org/nutch/FrontPage>
- [2] <https://github.com/sumansaurabh/nutch>
- [3] <https://sites.google.com/site/nutch1936/home/3-methodology>
- [4] http://www.cloudera.com/content/cloudera/en/documentation/core/v5-2-x/topics/cdh_ig_mapreduce_to_yarn_migrate.html
- [5] <http://www.slideshare.net/tshooter/strata-conf2014>

- [6] http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduce_Compatibility_Hadoop1_Hadoop2.html
- [7] <http://www.slideshare.net/wattsteve/web-crawling-and-data-gathering-with-apache-nutch?related=2> *