Date: **September 29, 2013**	# ACG Software Install Guide

LLNS Subcontract No.	B599860
Subcontractor Name	Intel Federal LLC
Subcontractor Address	2200 Mission College Blvd. Santa Clara, CA 95052

ACG Software Install Guide

HAL (HDF5 Adaptation Layer)

In this installment, we advanced our HDF5 adaptation layer; To install the new hal, do the following :

Prerequisite: install HDF5 (v1.8.10 or above) on all nodes.
For our tests, we copied all hdf5 libraries in /usr/lib and hdf5 header files in /usr/include. If you put them in other location, the lib paths must be appropriately adjusted.

- Unzip the package and extract the folder hal, and copy it to a suitable location

```
-   # cd hal
-   # rm –rf build
-   #mkdir build
-   # cd build
-   # cmake ..   (this should create a Makefile in the build folder)
-   # make
```

Next step

```
-   copy hal/build/src/libhal.so to /usr/lib
-   copy hal/include/hal.h to  /usr/include
```

To test the hal library – you can go to hal/build/test – and run

```
        # ./haltest  --help
```

You should see the following prompt, describing the use of this sample unit-test suite.

```
for rev0test:  ./haltest 1 e for edgelist
for rev0test:  ./haltest 1 a for adjlistlist
for rev1test:  ./haltest 2  - gets partition count in TG1.h5
for SnapToHDF:  ./haltest 3 <infilename> <numpartitions>  [optional <replication-factor>]
for adding edges in chunks:  ./haltest 4
get max. vertex id in a tsv(edgelist) file:  ./haltest 5 <filename>
readhaltest - reads all edges in partition 0 of file TG1.5 and prints them  ./haltest 6
```

Graph Generator

A new graph generator is rolled into the distribution. This runs as a haoop map/reduce system, but is written using C++, and thus fed into the Hadoop processes through the use of Hadoop pipes. The default install of Hadoop doesn't usually come ready for Hadoop pipes, and therefore Hadoop libraries need to be recompiled to enable pipes. The steps for recompiling for Hadoop pipes are given in the appendix.
However that's a pre-requisite for running the graph-generator

Pre-requisites

- HDF5 and its prerequisites are installed (version 1.8.10 and above)
- New HAL (HDF5 Adaptation Library) as described before
- Hadoop is installed (we tried with Hadoop 1.0.3, but presumably everything should run on higher versions as well. For basic Hadoop install, you can consult many different tutorials online, or ACG software install guide for milestone 4.6)
- Hadoop is recompiled for Hadoop pipes – please check the Appendix of this document.

Graph Generation Install and Usage

Installing

The Graph generator is located in a folder called skggen_pipes.

The following steps are used to build the binary (standard cmake build)

- Unzip the package and extract the folder skggen_pipes, and copy it to a suitable location

```
-   # cd skggen_pipes
-   # rm –rf build
-   # cd build
-   # cmake ..   (this should create a Makefile in the build folder)
-   # make
```

You should see the binary (skggen) is created in the folder ../skggen_pipes/build/src

Usage

To run this you can modify a sample script provided with the software; once you untar and unzip the package you can locate it in …/skggen_pipes/scripts/skggen.sh

The script takes the following parameters

```
    ------------------------------------------------------------------------------
    hadoop dfs -rmr skggen/output       # specify output directory in HDFS to be removed before
execution.

    hadoop.pipes.java.recordreader=true # do not change
    hadoop.pipes.java.recordwriter=true # do not change
    edge_factor=15                      # num_edges = num_vertices * edge_factor
    scale=10                            # num_vertices = 2^scale
    seed=11378848                       # seed for random number generator
    spk_noise_level=20 \                # noise level under experiment
    offset=1000                         # num_edge_to_be_generated_by_one_reduce_task =
num_edge/offset
    h5_file="/home/tester/test.h5"       # path+filename for h5 file (if you remove this line,
skggen will store graphs in Hadoop distributed file system)

    reduces 1                           # number of reducers running per node
    input skggen/input.txt              # input.txt, a non-empty file,contains fake input data,
for example, 0
    output skggen/output                # Output directory in HDFS
    program skggen/bin/skggen           # path+binary name in HDFS
     ------------------------------------------------------------------------------------------
```

To run the script, do the following:

- Recompile Hadoop for pipes (steps in Appendix)
- Install Hadoop across your cluster
- Create the input directory needed by the script as shown above
- Set the size of the graph by the parameter **scale;** the size of the graph is 2^(scale). The edge factor is used to determine how many edges the graph will have (#edges = edge_factor * 2^(scale))
- After setting all the params and running ./skggen.sh at the command prompt you'll see Hadoop prompting for completion of map and reduce jobs. Once they are done, collect the output hdf5 file as mentioned in the parameter.

GraphLab adaptation and usage

In this milestone, we adapted the publicly available version of graphlab library (http://graphlab.org/) to accept graphs in the form of hdf5 files for running graph computation.

Pre-requisite

1. New HAL library must be installed as before on all nodes of the system
2. (Optional) Hadoop is installed, else you cannot use hdfs as the location for Graphlab's input and output files.

Install

1. First you need to install graphlab on your system. You can download execute the steps from GraphLab website (http://graphlab.org/downloads).
2. Apply FastForward patch:
   - unzip and untar the fastforward tarball for milestone 5.8
   - copy the file from FastForward folder graphlab/src/graphlab/graph/distributed_graph.hpp into the corresponding folder of original Graphlab install directory.

3. Build graphlab again using steps as described in (1) above.

Running Graphlab on hdf5 graphs

Once compiled, you are ready to run graphlab binaries to do graph computation. At this point the current version requires the following:

- Each node has a copy of the input file located in an identical location
- The graph, presented as edge-list, is partitioned into as many partitions as there are number of nodes

- Each graphlab process running on each node, gets a mpi-rank- say ***r*** – the process with rank r will load partition# ***r*** in its own memory. That places requirement on the .h5 file structure. The partitions are to be named like P_0, P_1, P_2 …… and so on.
- To test run ..
  - # cd <parent_folder>/graphlab/release/demoapps/pagerank/

You should see a file runpr.sh (there are other example variation of this file too) that looks like the following –

```
mpiexec \

-n 16 \

   -hostfile ~/machines \

 -x CLASSPATH=`hadoop classpath` \

   /home/tester/dev/graphlab/release/demoapps/pagerank/simple_pagerank \

   --graph /home/tester/dev/graphdata/1m_edge.h5 \

   --saveprefix hdfs://ffmaster:9000/user/tester/graphlab/pagerank/output/1m_edge_h5_pr_output \
```

- Modify the parameters
  - --graph to include the appropriate input hdf5 file
  - --saveprefix to include the appropriate place for placing the output file
  - If the output is in hdfs the Hadoop must be running. You can specify a non-hadoop location as well (in that case the param –x CLASSPATH is not needed).

Appendix – Recompiling for Hadoop Pipe

- Install OpenSSL library

```
sudo apt-get install libssl1.0.0
```

- Check if `libssl` and `libcrypto` are installed properly.

```
ls -al /usr/lib/x86_64-linux-gnu/libssl*

ls -al /usr/lib/x86_64-linux-gnu/libcrypto*
```

- Manually modify the following lines:

```
Index: $(HADOOP_INSTALL)/src/c++/utils/m4/hadoop_utils.m4

===================================================================

--- $(HADOOP_INSTALL)/src/c++/utils/m4/hadoop_utils.m4     (revision 1136761)

+++ $(HADOOP_INSTALL)/src/c++/utils/m4/hadoop_utils.m4     (working copy)

@@ -51,8 +51,8 @@

   AC_MSG_ERROR(Please check if you have installed the pthread library))

 AC_CHECK_LIB([pthread], [pthread_create], [],

   AC_MSG_ERROR(Cannot find libpthread.so, please check))

-AC_CHECK_LIB([ssl], [HMAC_Init], [],

-  AC_MSG_ERROR(Cannot find libssl.so, please check))

+AC_CHECK_LIB([crypto], [HMAC_Init], [],

+  AC_MSG_ERROR(Cannot find libcrypto.so, please check))

 ])



 # define a macro for using hadoop pipes
```

- In `$(HADOOP_INSTALL)/src/c++/pipes/impl/HadoopPipes.cc`, add `#include <unistd.h>`

- In `$(HADOOP_INSTALL)/src/contrib/gridmix/src/java/org/apache/hadoop/mapred/gridmix/Gridmix.java`, apply the following patch

```
Index: src/contrib/gridmix/src/java/org/apache/hadoop/mapred/gridmix/Gridmix.java

===================================================================
--- src/contrib/gridmix/src/java/org/apache/hadoop/mapred/gridmix/Gridmix.java         (revision 1340233
+++ src/contrib/gridmix/src/java/org/apache/hadoop/mapred/gridmix/Gridmix.java         (working copy)
@@ -613,10 +613,10 @@
     }
   }
-  private <T> String getEnumValues(Enum<? extends T>[] e) {
+  private String getEnumValues(Enum<?>[] e) {
     StringBuilder sb = new StringBuilder();
     String sep = "";
-    for (Enum<? extends T> v : e) {
+    for (Enum<?> v : e) {
       sb.append(sep);
       sb.append(v.name());
       sep = "|";
```

- In `$(HADOOP_INSTALL)/src/c++/utils`

```
./configure
make install
```

- In `$(HADOOP_INSTALL)/src/c++/pipes`, run

```
# autoreconf -f
# ./configure
# make install
```

- In the new Makefile, use

```
-I$(HADOOP_INSTALL)/src/c++/install/include
-L$(HADOOP_INSTALL)/src/c++/install/lib -lhadooputils -lhadooppipes -lcrypto -lssl -
lpthread
```

Appendix – Profiling for Performance studies

For this round, we have used a publicly available profiling tool called oprofile, to understand the performance of our software. It can be downloaded from http://oprofile.sourceforge.net/news/

The basic steps of running oprofile can be found in http://ssvb.github.io/2011/08/23/yet-another-oprofile-tutorial.html

To collect profiling statistics for the software that we are rolling out, the profiling daemon must be running on all nodes.

1. Start the profiler, execute the following steps.

```
# sudo opcontrol --deinit
# sudo opcontrol --separate=kernel
# sudo opcontrol --init
# sudo opcontrol --reset
# sudo opcontrol --start
```

2. Run the ACG software (skggen or graphlab) while the profiler daemon is on

3. To stop the profiler, do the following

```
sudo  opcontrol --stop
sudo  opcontrol --deinit
sudo  opcontrol --reset
```

4. Collect performance profile on each node by typing :
   # opereport or #opreport –l