Paper SAS10940-2016

How to Move Data among Client Hard Disk, Hadoop File System and LASR™ in SAS®

Yue Qi, SAS Institute Inc.

ABSTRACT

In SAS LASR Analytics Server, the data can reside in three types of environments: client hard disk (e.g. laptop), Hadoop File System (HDFS) and memory of the LASR Analytic Server. How to move the data efficiently among these is critical to get the insights behind data on time. In this paper, we will illustrate all the possible ways to move the data including 1) moving data from client hard disk to HDFS; 2) moving data from HDFS to client hard disk; 3) moving data from HDFS to LASR Analytic Server; 4) moving data from LASR Analytic Server to HDFS; 5) moving data from client hard disk to LASR Analytic Server; 6) moving data from LASR Analytic Server to client hard disk.

INTRODUCTION

Leveraging great power of parallel and distributed in-memory computing to handle the problem of analytics in big data, SAS LASR Analytic Server is shown to have exceptional performance in terms of computing speed and model accuracy by benchmarking result and users' feedback. LASR Analytic Server gains tremendous popularity after it was released. Its programming interface, SAS In-Memory Statistics covers all necessary functions for an entire modeling process including data integration, data exploration, statistics modeling and data mining. It incorporates capabilities such as recommendation engine and social network analysis as well. There are also many applications built on top of LASR Analytic Server. For example, SAS Visual Analytics and Visual Statistics, SAS Visual Scenario Designer and so on.

Since at least three types of environment are involved in LASR Analytic Server: client hard disk (e.g. your laptop), Hadoop File System (HDFS) and memory of the LASR Analytic Server, a critical problem is how to transfer data among them. Data transferring is a simple, but not trivial, yet sometimes disturbing job for beginners to start using LASR Analytic Server. How to transfer data among those environments freely and easily is often the first aspect of knowledge that many users eager to learn. Besides HDFS, LASR Analytic Server also supports other data sources provide by third-party vendors such as Greenplum, MapR, Teredata, and so on. But how to communicate with these data providers is not in the scope of this paper.

The purpose of the paper is helping users effortlessly overcome the steep initial learning curve of data movement in LASR Analytic Server by summarizing all possible data transferring directions and providing examples of how to transfer data in those directions. Note that for each data transferring direction, there could be multiple methods to accomplish the same task. The intention of this paper is not to include exhaustive methods of data transferring in LASR Analytic Server, but to provide methods that are good enough to fulfill the tasks efficiently and timely.

TRANSFER DATA BETWEEN HDFS AND LASR ANALYTIC SERVER

LOAD DATA FROM HDFS TO LASR ANALYTIC SERVER

Suppose you have a data set stored in a Hadoop Files System (HDFS) which is co-located with LASR Analytic Server, in other words, it is on the same set of hardware with LASR Analytic Server. The first method is you can load the data set into a LASR Analytic Server when you create it using CREATE statement in PROC LASR. You can add more data sets using ADD statement in PROC LASR later. The following code shows the complete process of the method.

```
option set=GRIDHOST="grid001.example.com";                    /* 1 */
option set=GRIDINSTALLLOC="/opt/TKGrid";                      /* 2 */

libname hdfs1 sashdat path="/path1";                          /* 3 */
```

```
proc lasr create path="/tmp/" data=hdfs1.organics_test_1;       /* 4 */
   performance nodes = all;
run;

proc lasr add port=&lasrport data=hdfs1.organics_test_2;        /* 5 */
run;
```

In line [1], GRIDHOST environment variable specifies the control node for both SASHDAT engine which is defined in line [3] and PROC LASR in line [4].

In line [2], GRIDINSTALLLOC environment variable specifies the install location of LASR Analytic Server for PROC LASR in line [4].

In line [3], path of the data set in HDFS is specified by LIBNAME statement and SASHDAT engine.

In line [4], a LASR Analytic Server is created and a data set, *organics_test_1* is loaded into it from HDFS. You can use PORT option to determine a port number manually. If you do not specify a port number manually, a random integer will be assigned as port number and saved to a macro variable with predefined name, *lasrport,* which can be used later. The port number is also shown in log window which you need to write down so that you can use it when you want to connect to the LASR Analytic Server from another SAS session.

In line [5], another data set, *organics_test_2* is loaded to LASR Analytic Server. Built-in marco variable *lasrport* is used to indicate the port number of LASR Analytic Server you just created.

The second method is 1) Creating an empty LASR Analytic Server without loading any data set into it. 2) Adding data sets to it using ADD option in PROC LASR.

```
option set=GRIDHOST="grid001.example.com";
option set=GRIDINSTALLLOC="/opt/TKGrid";

libname hdfs1 sashdat path="/path1";

proc lasr create path="/tmp/";                                  /* 6 */
   performance nodes = all;
run;

proc lasr add port=&lasrport data=hdfs1.organics_test_1;        /* 7 */
run;

proc lasr add port=&lasrport data=hdfs1.organics_test_2;        /* 8 */
run;
```

In line [6], an empty LASR Analytic Server is created.

In line [7] and [8], two data sets are loaded to the LASR Analytic Server you just created.

Note that both methods requires LASR Analytic Server is created using all the nodes (nodes = all), because data is stored in a distributed way as data blocks in HDFS and could be distributed to any node. If you create a LASR Analytic Server on only part of the nodes, it might not be able to find all the data blocks which leads to loading failure because some blocks are missing.

SAVE DATA FROM LASR ANALYTIC SERVER TO HDFS

Suppose you generated a new in-memory table or modified an existing in-memory table in a LASR Analytic Server, which you want to save into a co-located HDFS. You can do it using SAVE statement in PROC IMSTAT. The following code saves table *organics_test_3* in LASR Analytic Server to subdirectory */path1/example* in HDFS, and replaces the file with the same name in the same subdirectory, if such file exists:

```
proc imstat;
   table mylasr.organics_test_3;
      save path="/path1/example/organics_test_3" replace fullpath;
run;
```

TRANSFER DATA BETWEEN CLIENT HARD DISK AND LASR ANALYTIC SERVER

You can use SASIOLA engine to transferring data between client hard disk and LASR Analytic Server. SASIOLA functions in both two directions which means it can load data from client hard disk to LASR Analytic Server and save data from LASR Analytic Server to client hard disk. SASIOLA is short for **SAS** **I**nput/**O**utput to **LA**SR.

LOAD DATA FROM CLIENT HARD DISK TO LASR ANALYTIC SERVER

The following code snippet load data set *organics_test_1.sas7bdat* from a folder in local C: drive to an existing LASR Analytic Server. If you already specified GRIDHOST environment variable in the same SAS session, you do not have to set it again here.

```
option set=GRIDHOST="grid001.example.com";


libname client1 'C:/example/sasdemo';
libname mylasr sasiola port = &exampleport;

data mylasr.organics_test_1;
   set client1.organics_test_1;
run;
```

SAVE DATA FROM LASR ANALYTIC SERVER TO CLIENT HARD DISK

The following code snippet saves in-memory table *organics_test_1* from LASR Analytic Server to a folder in local C: drive.

```
option set=GRIDHOST="grid001.example.com";


libname client1 'C:/example/sasdemo';
libname mylasr sasiola port = &exampleport;

data client1.organics_test_1;
   set mylasr.organics_test_1;
run;
```

TRANSFER DATA BETWEEN CLIENT HARD DISK AND HDFS

UPLOAD DATA FROM CLIENT HARD DISK TO HDFS

You can use SASHDAT engine to upload data from client hard disk to HDFS through simple LIBNAME statement and DATA step code. Note that both GRIDHOST and GRIDINSTALL environment variables are necessary for SASHDAT engine. The following code snippet uploads data set *organics_test_1.sas7bdat* from a folder in local C: drive to HDFS:

```
option set=GRIDHOST="grid001.example.com";
option set=GRIDINSTALLLOC="/opt/TKGrid";


libname client1 'C:/example/sasdemo';
libname hdfs1 sashdat path="/path1";
```

```
data hdfs1.organics_test_1;
   set client1.organics_test_1;
run;
```

DOWNLOAD DATA FROM HDFS TO CLIENT HARD DISK

SASHDAT engine is a one-way engine, in other words, you can use it to upload data from client hard disk to HDFS, but cannot use it to download data from HDFS to client hard disk. However there are other solutions to bypass the problem. For example, you can load data from HDFS to LASR Analytic Server first, then save data from LASR Analytic Server to client hard disk. Alternatively, you can use PROC HPDS2 to complete the task as shown in the following code snippet:

```
option set=GRIDHOST="grid001.example.com";
option set=GRIDINSTALLLOC="/opt/TKGrid";


libname client1 'C:/example/sasdemo';
libname hdfs1 sashdat path="/path1";


proc hpds2 in= hdfs1.organics_test_1 out= client1.organics_test_1;
   performance details;
   data DS2GTF.out; method run(); set DS2GTF.in; end; enddata;
run;
```

CONCLUSION

The above code snippets are in the parsimonious manner to help you get started quickly. There are many other useful options in DATA step, SASIOLA and SASHDAT engine and PROC LASR that you can add to better fit your needs. Please check reference guide of LASR Analytic Server for details.

REFERENCES

SAS® LASR™ Analytic Server 2.7: Reference Guide. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/inmsref/67629/HTML/default/viewer.htm

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Yue Qi
100 SAS Campus Drive
Cary, NC 27513
SAS Institute Inc.
Yue.Qi@sas.com
http://www.sas.com