mrnative – Run your native code in an Hadoop environment

Yunhui Fu

Nov 25, 2015

The mrnative stack

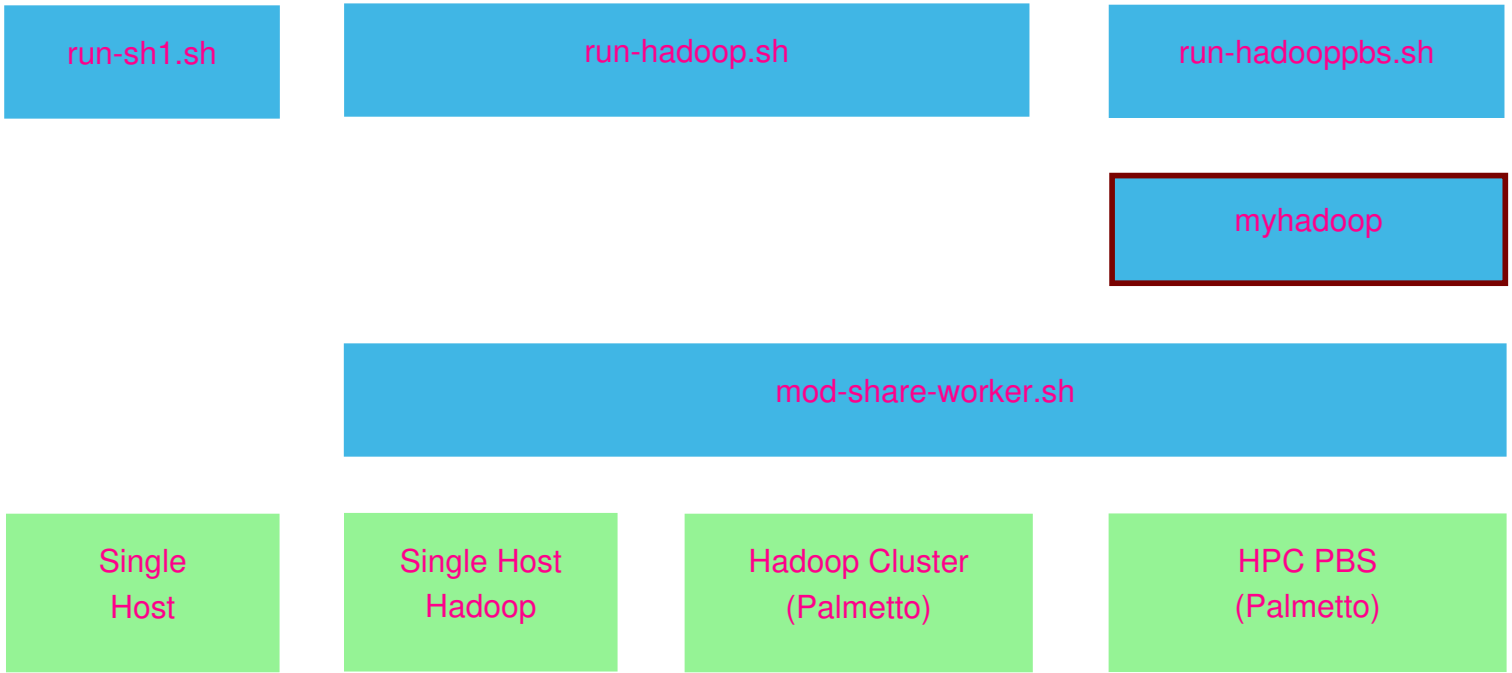


Figure: The stack of the mrnative

- write map/reduce code in native code
- run in various system settings
- supports single host, single host Hadoop, Hadoop cluster, and HPC PBS over myhadoop

single host

- emulate the streaming mode map/reduce in bash;
- test the correctness of your process flows;
- run a small amount of tasks which can use all of the cores in the system;
- get the results in a subdir `output-*`

```
1 cd projtools
2 ./run-sh1.sh
```

single host Hadoop

- simulate a Hadoop environment;
- to find out Hadoop environment related problems;
- get the results in a HDFS `hdfs://tmp/$USER/output-*`

```
1 cd projtools
2 ./run-hadoop.sh
```

Hadoop Cluser

- run your program in a productive environment;
- run a huge amount of small tasks;
- you need to set two variables in your `mrsystem.conf`: (b/c we can't set the vcore mem parameters)
  - `HDFF_NUM_CLONE` to 1.
  - `HDFF_TOTAL_NODES` to the number of vcores (eg. 400); and
- get the results in a HDFS `hdfs://tmp/$USER/output-*`

```
1 ssh $USER@user.palmetto.clemson.edu
2 ssh resourcemgr.palmetto.clemson.edu
3 cd projtools
4 ./run-hadoop.sh
```

HPC PBS with myHadoop

- run your program in a productive environment;
- run a huge amount of small tasks;
- automatically detects the available resources, and setup the required config files for Hadoop;
- get the results in the `/scratch1/$USER/output-*`

```
1 ssh $USER@user.palmetto.clemson.edu
2 cd projtools
3 ./run-hadooppbs.sh
```

Three type of config files

- `mrsystem.conf` – the configs for the software, such as the number of nodes, the scratch directory etc;
- `input/` or `input.conf` – the configs for input data;
- `output/` or `output.conf` – the configs for output data;

Compile 3rd party tools

- using the compile scripts in folder `3rd`;
- current version supports compiling `ns2`, `ffmpeg`, `opencv`, and gnu tools such as `gawk`, `gnuplot` etc.;
- the compiled binaries are packed in a `tar.gz` file;
- the `tar.gz` package contains the required libraries and support files, strip other redunant binaries or files generated during compiling;

Compile NS-2

- in a Linux machine, type

```
cd 3rd
make dist-gzip-ns2
```

- compile `ns-2` and required tools `gawk, gnuplot` etc.
- the file `ns-*.tar.gz` contains the binaries;

Compile ffmpeg

- in a Linux machine, type

```
cd 3rd
make dist-gzip-ffmpeg
```

- compile `ffmpeg`, `gpac`, `mediametrics`, `opencv` with or without GPU supports.
- the file `ffmpeg-*.tar.gz` contains the binaries;

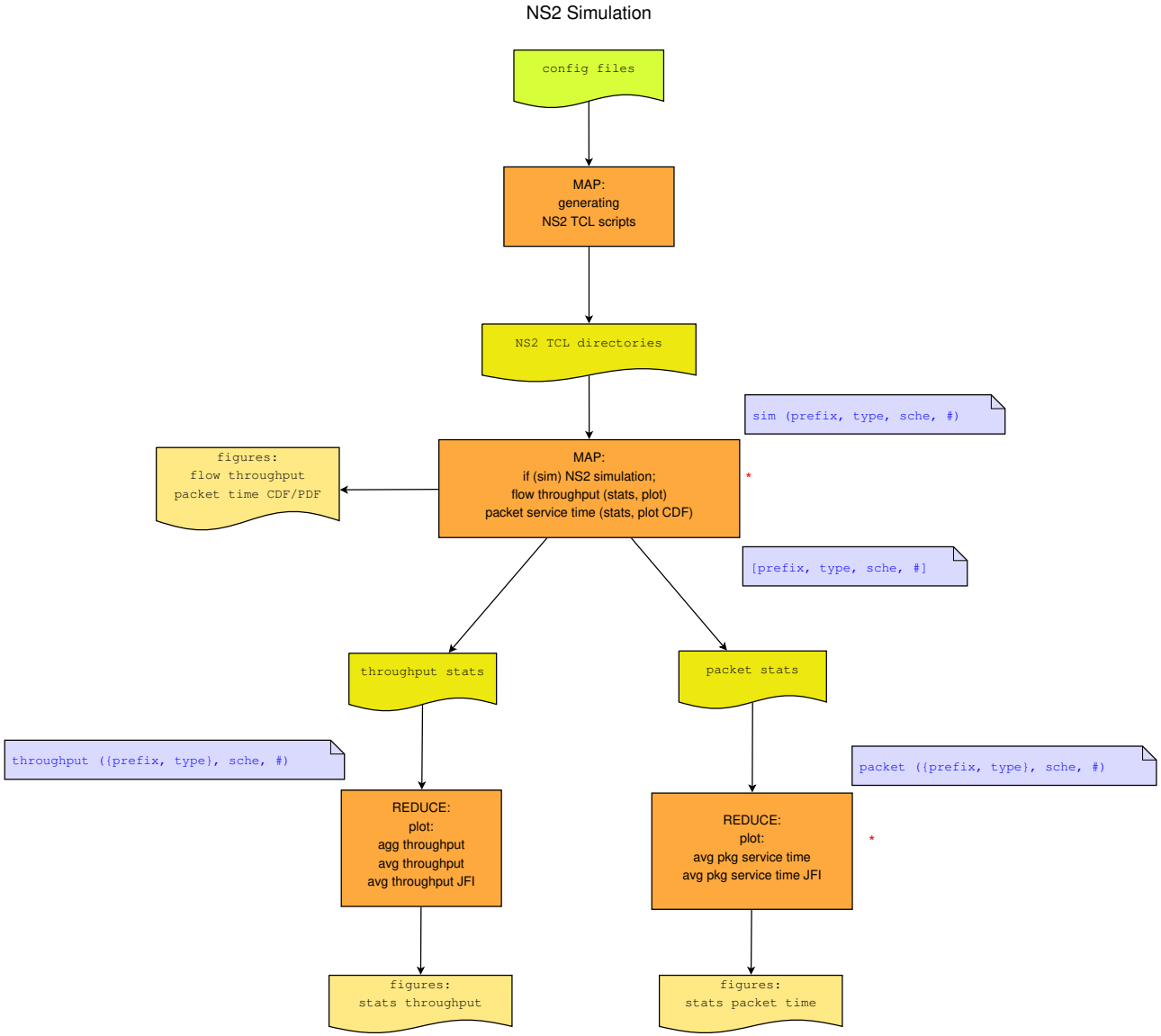
NS2 network simulator introduction

- NS-2 supports only single host simulating
- it's time-consuming simulating

NS2 simulating scenario

- Compare multiple packet scheduler algorithms
- Deploy various number of nodes, such as 1, 3, 6, 12, 24, 48, 96 competing packet flows each from one node;
- Construct different competing applications, such as UDP flows competing with several TCP flows.

NS2 simulating in MapReduce programming model



Media trans-coding introduction

- supports single host processing
- parallelize it by processing small chunk of video and merge them at the end of processing stage.

Group Of Pictures (GOP)

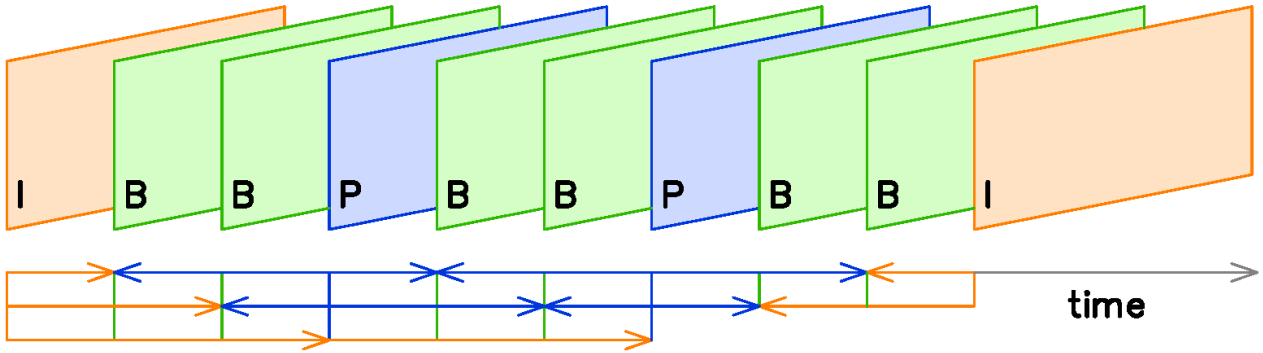


Figure: Example of a GOP structure (Wikipedia).

An MPEG Group Of Pictures (GOP), starts with an "I" frame, follows with multiple "P" or "B" frames.

- "I" frame (Intra-coded picture, key frame) like a conventional static image file;
- "P" frame (Predicted picture) holds only the changes from the *previous* frame;
- "B" frame (Bi-predictive picture) holds the differences between the current frame and both the preceding and following frames.

Media trans-coding scenario

- The input source, such as PNG sequences files, lossless video files, etc.
- The output format, such as DASH, H.264 etc.

Media tran-scoding in MapReduce programming model