Elastic parallel computing with R, Redis and the foreach package on Amazon EC2

The doRedis package makes it easy to scale out parallel R compute jobs using Amazon's elastic compute cloud. The R package includes example scripts for setting up a doRedis service that can be used with EC2.

Configure an Amazon Instance

Begin with a Linux-based Amazon instance. This example should work on RHEL/CentOS or Ubuntu-based AMIs. The step-by-step command-line process outlined below assumes Ubuntu.

Install R and Redis and prerequisites

The bash shell is required by the doRedis service.

Listing 1: Installing prerequisites

```
sudo apt-get build-dep r-base libxml2-dev libssl-dev \
     libcurl4-openssl-dev unzip  redis-server
wget https://www.cran.r-project.org/src/base/R-3/R-3.2.3.tar.gz
# (or most current version of R)
tar xf R-3.2.3.tar.gz
cd R-3.2.3
./configure --enable-R-shlib --enable-memory-profiling
make -j 2     # use an appropriate number of CPUs here
sudo make install
```

Configure Redis to listen on all interfaces

It is *important* to combine this step with EC2 security firewall rules. You need access on the Redis port 6379 between nodes in your compute cluster, *but not to nodes on the internet*. This is important. Fortunately, it's easy to achieve in two steps:

1. Selecting a VPC network when you instantiate nodes; the default one (something like vpc-xxxxxxxx 172.31.0.0/16) should work.
2. Edit your security group to add the following inbound rule, Protocol: TCP, Port Range: 6379, Source: Custom IP 172.31.0.0/16

Those steps can be performed with the Amazon EC2 GUI console, or from the EC2 CLI or API. Again, it's very important to carefully prevent outsiders from accessing your Redis port. It must only be available within a VPC network used by your compute nodes.

The following command will configure your Redis server to listen on all interfaces:

```
sudo sed -i.bak  "s/^bind 127.0.0.1/bind 0.0.0.0/"
/etc/redis/redis.conf && sudo /etc/init.d/redis-server restart
```

Install R packages

Finally, we need to install the foreach, rredis, and doRedis packages, plus whatever other R packages you might want to use. The following script installs rredis and doRedis from their source repository on GitHub using the devtools package. Soon, you will also be able to install the latest versions (early 2016) from CRAN.

```
sudo R --slave -e "install.packages(c('devtools', 'foreach'), /
                   repos='https://cran.r-project.org')"
sudo R --slave -e "devtools::install_github('bwlewis/rredis')"
sudo R --slave -e "devtools::install_github('bwlewis/doRedis')"
```

Install the doRedis service

The doRedis package includes an example script for setting up R workers as a service. The script works on generic Linux systems and also has special options to facilitate running on EC2. Setting up the service for EC2 is easy:

```
script=$(R --slave -e "cat(system.file(
         'scripts/redis-worker-installer.sh', package='doRedis'))")
sudo $script EC2
```

Make an AMI from your instance

Once you’ve got your instance and its R environment and packages configured to your liking, make an AMI from it. This can be done from the EC2 control panel web GUI by shutting down (stopping) your instance and selecting ”Image...Create image” from the instance menu.

Plan on including all the R packages you normally need in your AMI. If you run the doRedis service as the “nobody” user (the default), that user will not be able to easily install new packages other than in a temporary directory.

Re-start your instance and log back in

After creating an AMI, you're ready to play with doRedis on EC2. Start up your master instance and log back in, and note its IP address.

EC2 R/Redis/foreach workers are configured using the Amazon EC2 ”user data” service. We simply need to supply a valid doRedis configuration file as user data. An example minimum configuration file is, replacing “172.31.20.55” with the IP address of your master node:

```
n: 2
host: 172.31.20.55
```

This configures 2 R workers to run per Amazon instance, communicating with the Redis host at 172.31.20.55, (using a default job queue named “RJOBS” in Redis). You can supply that to newly launched nodes by pasting that in to the user data box in the EC2 control panel web GUI.

New workers can be added at any time to your R/Redis/foreach cluster at any time by simply turning them on and passing the configuration user data at startup.