# Ray

## Korištenje

Dostupne verzije i pripadajući moduli:

| Verzija | Modul |
|---|---|
| 1.10.0 | ray/1.10.0 |

Primjer skripte za opis posla:

**ray-multinode.sge**

```
#$ -N ray-multinode
#$ -q p28.q
#$ -pe *mpi 14
#$ -cwd

module load ray/1.10.0

ray_isabella_start.sh

python cluster_test.py
```

**cluster_test.py**

```
import ray

ray.init(address='auto')


print('''This cluster consists of
    {} nodes in total
    {} CPU resources in total
'''.format(len(ray.nodes()), ray.cluster_resources()['CPU']))
```

**Važno**

Prije pozivanja vaše Python skripte, obavezno je u skriptama za opis posla **prvo pozvati skriptu** ray_isabella_start.sh kako je navedeno u primjeru.

**Napomena**

Ray je instaliran i konfiguriran u Python virtualnom okruženju kojeg korisnik može proširiti sa svojim proizvoljnim Python aplikacijama. Nakon učitavanja modula, korisnik dodatne aplikacije instalira s:

```
conda create --prefix $LOCALPKGS python=3.8
conda install --prefix $LOCALPKGS potrebni-python-pkg
ili
pip install --prefix $LOCALPKGS potrebni-python-pkg
```

## Instalacija

Framework Ray olakšava paralelizaciju Python aplikacija i na klasteru je pripremljen u Conda okruženju s Python 3.8. Ray ima vlastitu head & worker node arhitekturu pa je potrebno "ručno" pripremiti Ray klaster jednom kad raspoređivač poslova dodijeli slobodne resurse. U tu svrhu je pripremljena skripta `ray_isabella_start.sh`.

**Naredbe instalacije**

```
source /apps/miniforge3/bin/activate
conda create --prefix /apps/virtenv/ray1.10 python=3.8
conda activate /apps/virtenv/ray1.10
pip3 install -U ray
pip3 install -U 'ray[tune]'
pip3 install -U 'ray[rlib]'
pip3 install -U 'ray[rllib]'
pip3 install -U 'ray[server]'
pip3 install -U 'ray[serve]
```

ray_isabella_start.sh

```
 #!/bin/bash

jobid=$JOB_ID
machinefile=$TMPDIR/machines
head_node=''
password=''
portnum=0
declare -a list_machines
# ports are used by ray
declare -a skip_ports=(11123 10001 38717 44006)

while true
do
        # assign random port in range 20000 - 52767
        portnum=$(($jobid % $RANDOM + 20000))
        if [[ ! "${skip_ports[*]}" =~ "$portnum" ]]
        then
                break
        fi
done

# build uniq list of machines assigned by scheduler
for machine in $(cat $machinefile | uniq)
do
        list_machines[${#list_machines[@]}]=$machine
done

# first node is head node
master_node=${list_machines[0]}

# head node bootstrap
if [[ "x$(hostname)" == "x$master_node" ]]
then
        numcpus=$(grep $master_node $machinefile | wc -l)

        echo "Isabella Ray head - $numcpus cores @ $master_node port=$portnum"
        head_start_log=$(ray start --num-cpus $numcpus --port=$portnum --head | grep "ray start")
        head_start_log=${head_start_log#*ray start}

        head_node=$(echo $head_start_log | awk '{print $1}' | awk -F'=' '{print $2}')
        head_node="${head_node%\'*}'"
        head_node="'${head_node#*\'}"

        password=$(echo $head_start_log | awk '{print $2}' | awk -F'=' '{print $2}')
        password="${password%\'*}'"
        password="'${password#*\'}"
fi

sleep 10

# worker nodes bootstrap
for machine in ${list_machines[@]:1}
do
        numcpus=$(grep $machine $machinefile | wc -l)

        echo "Isabella Ray worker - $numcpus cores @ $machine"
        master_arg="${head_node//\'/}"
        password_arg="${password//\'/}"
    ssh $machine "eval `/usr/bin/modulecmd bash load ray/1.10.0` ray start --num-cpus $numcpus --
address=$master_arg --redis-password=$password_arg --block"&
done

sleep 10
```