

## GNU Octave-Cli + GNU Parallel

- 12) run GNU Parallel for deploying parallel work on 6 cores in local cluster ( 2 local cores and 4 remote cores ):  
`#touch log.txt && rm log.txt && parallel --slf clusterList.txt --progress < parallelCommands.txt &>> log.txt;`  
`cat log.txt`
- 13) please note that data, instructions network deployment, and run of octave-cli provide overheads. As a result processing should be big enough for obtaining any benefits from such model. For small data and instructions communication please use Message Passing Interface, which is better for such communication,
- 14) some High Performance Computing tricks:  
 -create local high throughput RAMDISK for data deployment to workers ( best efficiency will be provided with usage of PCIe multiple 1GE / 10GE network Host Bus Adapters HBA's, and getting benefits from fastest possible star network topology ) . For details please refer to RAMDISK tutorial. Single session mount of volatile 4GB RAMDISK:  
`#mkdir ~/RAMDISK && sudo mount -t tmpfs -o size=4096M,mode=777 tmpfs ~/RAMDISK/`  
 check if RAMDISK is mounted, and running:  
`#mount;`  
`#echo;echo RAM sequential writes;;for ((i=0;i<3;i++)); do dd if=/dev/zero of=~/RAMDISK/a.txt conv=fdatasync bs=1512M count=1;done; rm ~/RAMDISK/a.txt;echo;echo HDD sequential writes;for ((i=0;i<3;i++)); do dd if=/dev/zero of=~/a.txt conv=fdatasync bs=1512M count=1;done; rm ~/a.txt;`  
 -tmpfs RAMDISK has sequential read and write efficiency of about 6-10 SSD's RAID0, but pseudorandom read of small data latency is about order of magnitude smaller. Please note that overall storage size is about two orders of magnitude smaller than Solid State Drive disk array,  
 -make Samba file sharing of RAMDISK on local computer with HBA's. Mount file share with project folders on workers – there will be no need for rsync instructions, and data will be reasonably deployed to workers. For details please refer to Samba tutorial.  
`#sudo apt-get install samba cifs-utils; sudo smbpasswd -a userName`  
`#sudo vim /etc/samba/smb.conf`  
`[parallelProjectFileShare]`  
`path = ~/RAMDISK/`  
`force user = userName`  
`force group = root`  
`create mask = 0777`  
`directory mask = 0777`  
`hosts allow = IPv4 IPv4_1`  
`read only = no`  
`public = yes`  
`guest ok = yes`  
`writable = yes`  
`#sudo service smbd restart`  
 workers file share mount:  
`#ssh IPv4`  
`#mkdir ~/remoteRAMDISK && sudo mount -t cifs //IPv4/parallelProjectFileShare /home/userName/remoteRAMDISK -o username=userName`

Post Scriptum: it is simplified tutorial providing very basis of algorithms prototyping with GNU Octave-Cli and GNU Parallel packages in many core parallelism scheme. End program must be developed efficiently after completion of vast set of algorithm tests. Please do note, that programming language is less important ( for example Bash scripts efficiency ), in opposition to popular stereotypes. Program efficiency is mainly correlated with computations complexity, and efficiency of hardware use ( General Purpose Graphical Processing Units provides best results for majority of current problems in 2018y. - efficient execution time, but with price of longer development time ).

Post Post Scriptum: please do note, that some embedded platforms ( \*Pi ) might not provide full support for some functionalities – such problem appeared within Orange Pi One and Ubuntu 16.04. Check your hardware with Unit Tests before you will make bigger purchase order.