

NovoalignMPI User Guide

NovoalignMPI is a messaging passing version of Novoalign that allows the alignment process to be spread across multiple servers in a cluster or other network of computers

1. Multiple servers can be used to align one read file.
2. Novoalign on each server is multi-threaded to maximise server utilisation.
3. Linux and MAC OS-X servers with X86-64 CPUs can be included in the network.

NovoalignMPI takes the same command line arguments and options as Novoalign and operation is almost identical except that reads are being distributed to multiple servers for alignment.

The differences are:

1. The command line option **-o Sync** has no effect. Read alignments are always reported in the order that alignments complete rather than in the order they appear in the read files.
2. When using Quality calibration with the **-k** option without a calibration file (i.e. not **-k <file>**) the accumulation of mismatch counts and calibration is done per slave process. This does not affect the **-K <file>** option which will still record total mismatch counts for all alignments.

MPI API Version

NovoalignMPI uses MPICH version 3.1, an open source high-performance and widely portable implementation of the Message Passing Interface (MPI) standard which is available for download <http://www.mpich.org>

Setting Up MPICH

This is basic introduction to running MPI jobs using Hydra process manager, if you get stuck please refer to the MPICH Installation Guide.

Setting up host names

Every computer to be used must have a hostname that can be resolved via DNS server and/or /etc/hosts.

The IP addresses must resolve identically on every server! One common problem is that a server may have a /etc/hosts entry that uses the local loopback IP address

ie. This can cause problems....

/etc/hosts

```
#
# hosts          This file describes a number of hostname-to-address
#                mappings for the TCP/IP subsystem.  It is mostly
#                used at boot time, when no name servers are running.
#                On small systems, this file can be used instead of a
#                "named" name server.
# Syntax:
#
# IP-Address    Full-Qualified-Hostname  Short-Hostname
#
127.0.0.1      localhost
127.0.0.1      colinspc colinspc
```

The entry for colinspc should have the IP address of the server, not the local loop-back IP address.

Also check **/etc/hostname** and ensure it matches the server name

Any mismatches in names and IP addresses will surely mess up MPI even if the names can be resolved correctly via DNS and via **/etc/hosts** on other servers.

You also need to create a file consisting of a list of server names, one per line. Name this file **hosts.txt** and save it into the folder you will be running **NovoalignMPI** from.

hosts.txt

```
server1
server2
server3
```

These host names will be used as targets for **ssh** or **rsh**, so include full domain names if necessary. Check that you can reach these machines with **ssh** or **rsh** without entering a password. You can test by doing

```
ssh othermachine date
```

Testing Hydra

To test that **MPICH2** is installed and working correctly

```
mpiexec -np <number to start> -f hosts.txt hostname
```

This will run **hostname** on each server and you can easily see that MPI is working on the servers in your hostfile.

Running NovoalignMPI

Preparation

For the above **NovoalignMPI** command to work you have to ensure **novoalignMPI** and **novoalign.lic** are available on the **PATH** on all servers and that the **novoindex** is available using the same file string. There's no need for the read files to be available on all servers as they are only read by the master process.

If using quality calibration and the **-k <infile>** option then *infile* must be available on all servers in the ring.

Execution

Run NovoalignMPI:

```
mpiexec -f hosts.txt -np <number of jobs> novoalignMPI -d <novoidindex> -f  
<readfiles> <options> >report.novo 2>run.log </dev/null
```

The *number of jobs* must be at least 2. The first job is the master process and is responsible for file-io and sending reads to the slave processes for alignment. The master process does not do any alignments and does not use a lot of processor time. Other processes are slaves, they accept reads from the master process, align them and then returning the report lines to the master for output.

You can have more or less jobs than the number of hosts specified in the hosts file. If you have more jobs then some servers will have more than one NovoalignMPI process.

This makes it possible to have a slave process running on the same server as the master by starting two processes on the master server. One way to do this is to specify the master server first in the **hosts.txt** file and then start one more job than there are hosts. As jobs are started round robin on the servers the first server will have two jobs, the master and a slave.

Note. That novoalignMPI is multi-threaded and will use all CPU cores on a server unless the **-c** option is used to restrict the number of threads.

Note. If you place a slave process on the same node as the master process it's possible that CPU contention on this node can affect the master process and reduce overall performance. To avoid this you can use the **-mCPU** option to reduce the number of threads created for a slave on the same node as the master. You should reserve enough CPUs so that this node is not 100% used. Factors to take into account include: Are read files deflated? Are you piping SAM output into samtools view to convert directly to BAM? Do you then pipe to Novosort? These things add to CPU load on master node and can affect throughput of the master node and hence work delivered to all the slaves.

Threading and CPU Affinity

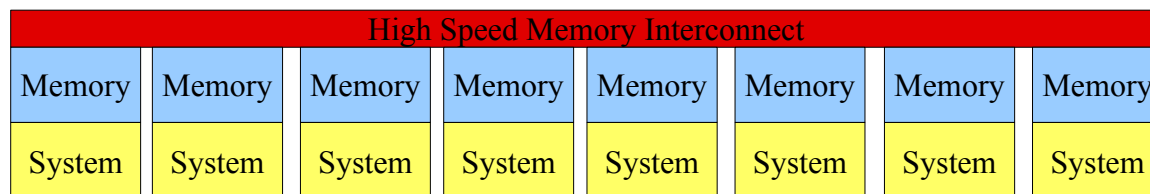
NovoalignMPI has several features to improve maximum throughput.

1. The Master process uses multiple threads to read, inflate and build the MPI messages. These threads are set to have affinity to all CPUs.
2. If a slave is running on the same node as the master process then it sets nice(12) so that the master process gets the CPU time it needs.
3. If **-c** option uses all cores then slaves set CPU affinity for the alignment threads.

NUMA Servers

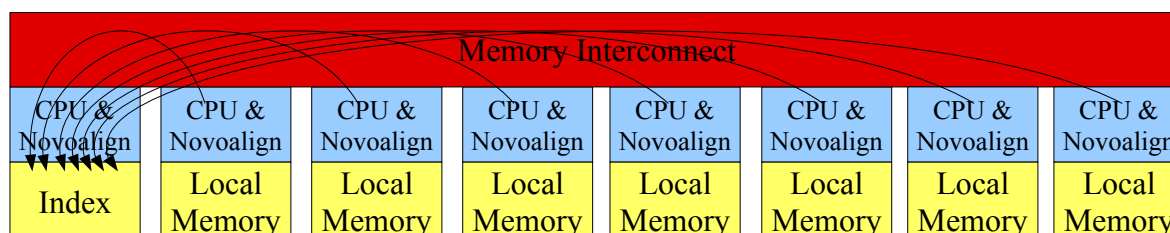
When using Novoalign on servers with Non Uniform Memory Access there can be contention for the index.

NUMA servers have some memory directly attached to each processor chip and an interconnect that



connects memory between the processors. Accessing the memory directly attached to a processor chip is many times faster than accessing memory attached to a different processor.

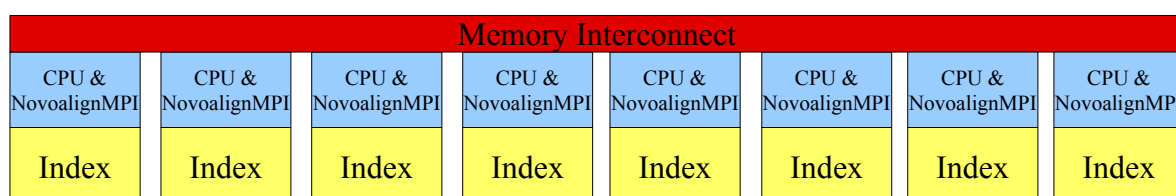
It's also possible to overload the interconnect bus. Consider case of using running Novoalign with 32 threads on 8 quad core processors. The indexed reference genome will be loaded into physical RAM of the first process that accesses it.



This can lead to overloading of the memory interconnect and reduced performance as the index is used very heavily.

Keeping a copy of the index in each processors local memory can improve performance.

With NovoalignMPI you can force an index to be loaded into the local memory of each processor chip.



To do this we start one slave MPI job per processor chip and multi-thread each Novoalign task using -c option to the number of cores in each processor. We also have to disable memory mapping of the index with the option -mmmapoff

Note. We have seen some systems where this hasn't worked and all the slaves started on the same CPU so we suggest testing to see if it improves performance.

Example.

For 8 quad core processors we might use:

```
mpiexec -np 9 -f hosts novoalignMPI -c4 -mmapoff ...
```

This assumes each processor has enough local memory to have it's own copy of the index.

Resolving Problems

1. Try running **mpiexec -f hosts.txt -np <number of hosts> pwd** to see what folder your MPI jobs are running in.
2. Check novoalignMPI is in PATH and all other files are accessible on all servers

```
mpirun -f hosts.txt -np <number of hosts> which novoalignMPI
```

Linking NovoalignMPI

As of Novoalign V3.02.03 we have included in the release the files necessary to compile and link your own Novoalign[CS]MPI. This should allow you to use different release of MPICH, MPICH2, Cray-MPICH and perhaps OpenMPI and Intel MPI. It also allows you to configure MPI to use your high speed interconnects.

Note. We have reports of novoalignMPI with openMPI hanging so please check and if you have a problem use MPICH.

So if you are having trouble using the pre-linked versions of the MPI programs you can try linking your own.

In the release folder there is a sub-folder buildMPI with the following files...

README	Documentation on how to compile and link the programs
Makefile	Makefile
mpidriver.cc	MPI specific source code files.
mpidriver.h	
trace.cc	Other system specific routines.
lib/novoalignCSMPI.a	Object library for NovoalignCSMPI
lib/novoalignMPI.a	Object library for NovoalignMPI
lib/bamtools.LIC	Bamtools license
lib/bamtools.a	Object library for Bamtools
lib/tcmalloc.LIC	Tcmalloc License
lib/libtcmalloc.a	Object library for tcmalloc
lib/unwind.LIC	libunwind License

Requirements

1. Installed and working MPI development environment.
2. BZIP2 development libraries
3. OpenSSL development libraries
4. HDF5 libraries for novoalignCSMPI