

## **Roche-454 / Newbler – Execution Guidelines for running applications in aggregated environment using ScaleMP’s vSMP Foundation**

### **Overview**

The Newbler is using a small number of cores in most phases of the run, and in addition has I/O intensive phases.

### **Job Placement**

As Newbler is using a small number of cores during the run, it is recommended to run Newbler while using the cpus on a single node. This can be done by using numactl to instruct the OS scheduler to place the job on the sockets (numa-nodes) on a single node.

For example:

```
numactl --cpunodebind=2,3 runAssembly...
```

Would place the job on numa-nodes 2 and 3; which on a dual socket system are on the 2<sup>nd</sup> board. While the job would still be able to use as much memory as is available on the entire aggregated system.

### **Fast I/O**

Newbler is using I/O extensively at certain phases of the run (especially the start and ending). As such, it is recommended to use the fastest option available for I/O.

### **Use of Ramfs**

In order to configure Ram-Drive, perform the following, in case /ramfs is not already mounted:

```
sudo mkdir /ramfs

sudo mount /dev/ram -t ramfs /ramfs -o noatime

sudo chmod 777 /ramfs
```

Once /ramfs is mounted, point Newbler to have the input files and the output to use /ramfs (see sample script).

### **Use of SW-RAID**

In case there is not enough RAM to run Newbler using /ramfs, it is recommended to use internal storage on the system. For that the best option would be a SW-RAID 0 .

Such volume would typically be configured already on the system. For instructions on how to configure one, please refer to the User’s guide or vSMP Productivity Pack,

\* It is recommended to avoid running Newbler over NFS, when internal storage is sufficient

## Sample run script

```
#!/bin/bash

# Create a ram-drive
sudo mkdir /ramfs
sudo mount /dev/ram -t ramfs /ramfs -o noatime
sudo chmod 777 /ramfs

# Copy the Newbler binaries
cp applicationsBin /ramfs

# Copy the Newbler job data
cp DataSets/Yeast /ramfs

cd /ramfs

ROOT=`pwd`
export PATH=$ROOT/applicationsBin:$PATH

OUTFILE=$ROOT/log-of-yeast-8-inhd-b0.log
mkdir -p Yeast

export MALLOC_TOP_PAD_=40000000
export MALLOC_TRIM_THRESHOLD_=536870912
export MALLOC_MMAP_MAX_=0

cd Yeast

# Use of a pre-load library we provide as part of the vSMP Productivity Pack, which would eliminate
unnecessary system calls (sched_yield in this case)
export LD_PRELOAD=/opt/ScaleMP/libvsmplib/0.1/lib64/libvsmplib.so

#numactl is used to bind the threads to the 2nd board (NUMA nodes 2 and 3 on a Nehalem based system)
time numactl --cpunodebind=2,3 runAssembly -cpu 8 -nrm -o yeastAssembly
$ROOT/DataSets/Yeast/*.sff >$OUTFILE
```