Research Report – Summer 2004

Converting Benchmarking Programs

Donya Shirzad
August 20, 2004

Introduction

The research work this quarter involved converting benchmarking programs to first use Java Sockets and then GridSockets. The MonteCarlo benchmarking program was converted successfully from MPJ to both Sockets and GridSockets. The Sugarscape program is in the process of being converted from C++ to Java so that it may also be converted to Sockets and GridSockets.

The original assignment was to convert MonteCarlo and two more benchmarking programs completely to Sockets and GridSockets, however the MonteCarlo program had out-of-memory errors which had to be debugged before it could be successfully converted. Debugging took more time than planned, and so the conversion of a second program began much later. Also, the conversion of Sugarscape from C++ to Java has been slow, as it is 2151 lines of code which must each be converted.

MonteCarlo

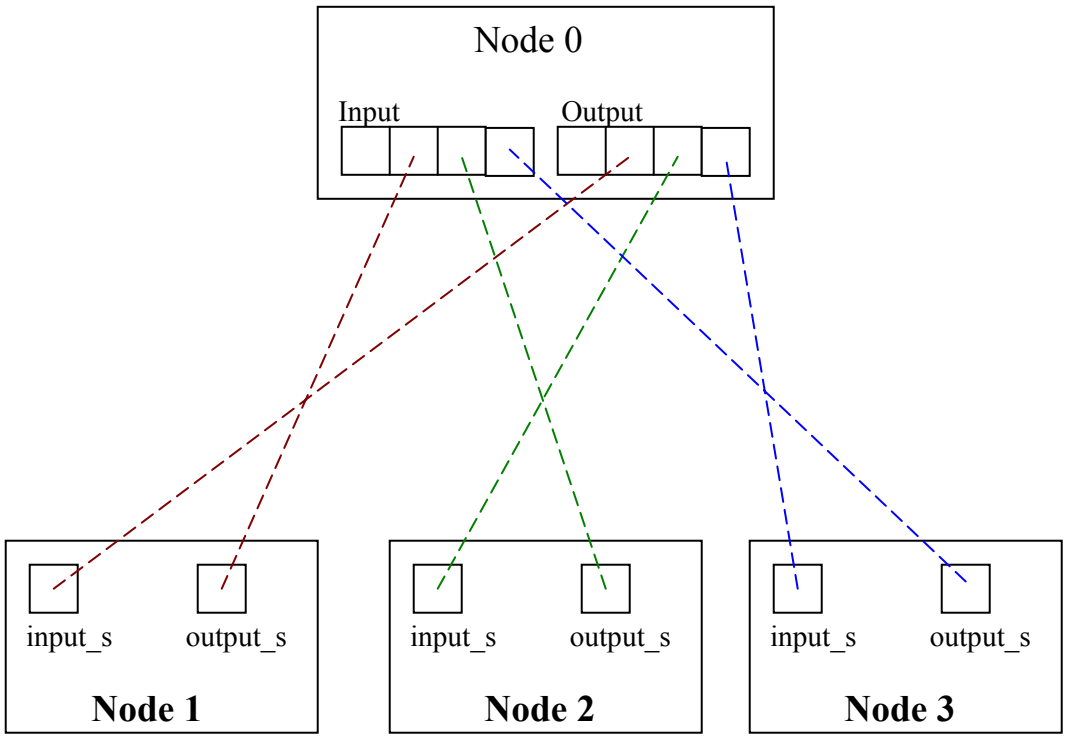
MonteCarlo is a financial simulation that prices products according to Monte Carlo techniques using historical data. It is part of an entire MPJ suite that has been converted from MPJ (which uses MPI) to Java Sockets, and then to GridSockets. Since the other programs in the suite had already been converted, MonteCarlo was converted by using the other programs as the model in order to be as similar in performance as possible.

Implementation

Sockets

Each node in the cluster of computers is given a ranking, starting at 0. Node 0 is the root node which collects all the data. Converting MPJ into Sockets required the use of two arrays of Sockets at Node 0, named Input and Output. Each subsequent node has two ServerSockets named input_s and output_s. When a non-root node (Node i) wants to

write data to the root, it writes to output_s. That data is received at the root node in the Input array at the index corresponding to the node's rank (Input[i]). When Node 0 wants to write data to another node, it writes to Output[i], where i is the ranking of the node. Node i then reads that data from its input_s socket.



The communication between the nodes had to be serialized into a byte array in order to be streamed across the socket. When the byte stream was received, it would be reassembled into an object. All the communication between nodes took place in the makeConnection() function of the AppDemo class.

GridSockets

After converting to Sockets, MonteCarlo was converted to GridSockets. The classes were no longer associated with a montecarlo package. GridTcp.jar and UserProgramWrapper.class were copied into the same directory as all the other files. All Sockets were changed to GridTcpSockets and ServerSockets to GridTcpServerSockets. References to rank had to be updated to myRank, as the UserProgramWrapper used with GridTcp automatically initializes the myRank variable. Three other variables also had to be declared for the UserProgramWrapper. These variables were also automatically initialized.

```
GridTcp tcp;
GridIpEntry ipTable[];
int funcId;
```

The UserProgramWrapper also requires that some functions be renamed and given an int return value. Instead of running main() or run(), the UserProgramWrapper starts at func_0 and when that finishes with a return value of -1, it continues to func_1, etc. Only when a function returns the value –2 will it stop incrementing to the next function. This meant that all the functions that were called by main() or run() in the Sockets version had to be moved into the same class file.

GridTcpMonteCarlo.java was created from AppDemo.java, CallAppDemo.java and MonteCarloBench.java. Changing the way in which objects were created and used was necessary in order to make them a single class for the UserProgramWrapper. After they were in a single file, the names could be changed to func_0, func_1, etc.

The function names that were changed are:

func_0 <- main(), first half of run()
func_1 <- initialise()
func_2 <- application()
func_3 <- validate()
func_4 <- tidyup()
func_5 <- second half of run()

The function numbers correspond to the order in which they were originally called in main() and run(). Immediately before the last function returns a –2 to end the program, a call to tcp.disconnect() must be made.

Issues

Out of Memory

After converting the MPJ version of MonteCarlo to Java Sockets, the program gave an out-of-memory error. Unfortunately, the medusa cluster was not set up to run the original MPJ version, and so many hours of debugging and code rewrites were wasted when it was found that the original version also had the out-of-memory error. The medusa cluster was never set up to run MPJ correctly, but prunjava was installed and allowed the program to run in a fashion. After the MPJ version could be run and the out-of-memory error found to be within the original code, some profiling of the error was accomplished.

The original MPJ version of MonteCarlo included two datasizes to be tested, 10000 and 60000. The size could be chosen by running either MonteCarloBenchSizeA.java or MonteCarloBenchSizeB.java. Both sizes caused out-of-memory errors on the medusa cluster.

By changing the first datasize in the datasizes array (CallAppDemo.java) to a lower number, and running only MonteCarloBenchSizeA.java, the out-of-memory error could be avoided.

The limits of the datasize were profiled with possible explanations. Results gathered on the Bothell Medusa cluster, July 22, 2004.

In testing with 1, 2, 4 and 8 processors, they all have different data sizes that they can handle, with 2 processors being the lowest at 4445. All the others were higher, probably because there are three places that the out-of-memory error can occur:

a) when creating the vector to hold the data
b) when sending the data through the Socket
c) when gathering the data back to node 0

It looks like the vector can handle a data size of around 7500.
The Socket can handle 4445.
And the system can gather data around 6500.

Expected Return Rate

The expected return rate was being compared to a constant number based on the original datasize, so after changing the datasize to avoid the out-of-memory error, the results were reported as incorrect by the program. Although the results do not match the program's expected result, as it is always a consistent number, it was considered to be correct.

Using the lowest common denominator for the out-of-memory error, I tested with 1,2,4,8 processors using a data size of 4445. The validation number is the same across all processors:

expectedReturnRate= 0.0 0.0333976656762814 0

I also changed the MPJ version to use a datasize of 4445 as well and ran it using prunjava. Unfortunately, I got different results. I tested 1, 2, and 8 processors. 2 still ran out of memory, so MPJ must use more in communicating, but 1 and 8 finished. Their numbers agreed with each other, but not with the numbers I got.

expectedReturnRate= -0.03335313452664269 4.4531149638711576E-5 0

As the first number for the expectedReturnRate should always be 0.0 according to the code, I believe the discrepancy is due to prunjava running it incorrectly.

Total Time

There may be a small bug in RayTracerBench.java on line 156, which was duplicated in MonteCarlo, in case it was not a bug, but the preferred behavior. The Total Time is printing out the runtime / 1000.0. I would expect it be totaltime / 1000.0.

Running Instructions

Sockets

The Sockets version of MonteCarlo can be run on 1 or more systems. If run on more than 1, the system with rank 0 must be started last, otherwise the communication will fail. The syntax is:

```
java MonteCarloBenchSizeA #processors rank [serverNames]
```

where:

#processors = the number of processors in the cluster (starts at 1)
rank = the rank of the system the program is being run on (starts at 0)
serverNames = names of the other servers in the cluster (only required for rank 0)

Examples:
Using the medusa cluster, using mnode0 as rank 0, mnode1 as rank 1, etc.

To run MonteCarlo on 1 system:

```
[uwagent@mnode0]$ java MonteCarloBenchSizeA 1 0
```

To run MonteCarlo on 2 systems:

```
[uwagent@mnode1]$ java MonteCarloBenchSizeA 2 1

[uwagent@mnode0]$ java MonteCarloBenchSizeA 2 0 mnode1
```

To run MonteCarlo on 4 systems:

```
[uwagent@mnode3]$ java MonteCarloBenchSizeA 4 3

[uwagent@mnode2]$ java MonteCarloBenchSizeA 4 2

[uwagent@mnode1]$ java MonteCarloBenchSizeA 4 1

[uwagent@mnode0]$ java MonteCarloBenchSizeA 4 0 mnode1 mnode2 mnode3
```

GridSockets

The GridSockets version of MonteCarlo can be run on 1 or more systems. If run on more than one, they may be started in any order. The syntax is:

```
java -cp GridTcp.jar:. UserProgWrapper - #processors myRank
myName [- nodeRank nodeName ] - GridTcpMonteCarlo [#processors] 0
```

where:

#processors = the number of processors in the cluster (starts at 1)
myRank = the rank of the system the program is being run on (starts at 0)
myName = the name of the system the program is being run on
- nodeRank nodeName = the rank and name of other systems in the cluster (must have a nodeRank nodeName specification for all systems the program will be running on)

Examples:
Using the medusa cluster, using mnode0 as rank 0, mnode1 as rank 1, etc.

To run MonteCarlo on 1 system:

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 1 0
mnode0 -  GridTcpMonteCarlo 1 0
```

To run MonteCarlo on 2 systems:

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 2 0
mnode0 - 1 mnode1 - GridTcpMonteCarlo 2 0

[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 2 1
mnode1 - 0 mnode0 - GridTcpMonteCarlo 2 0
```

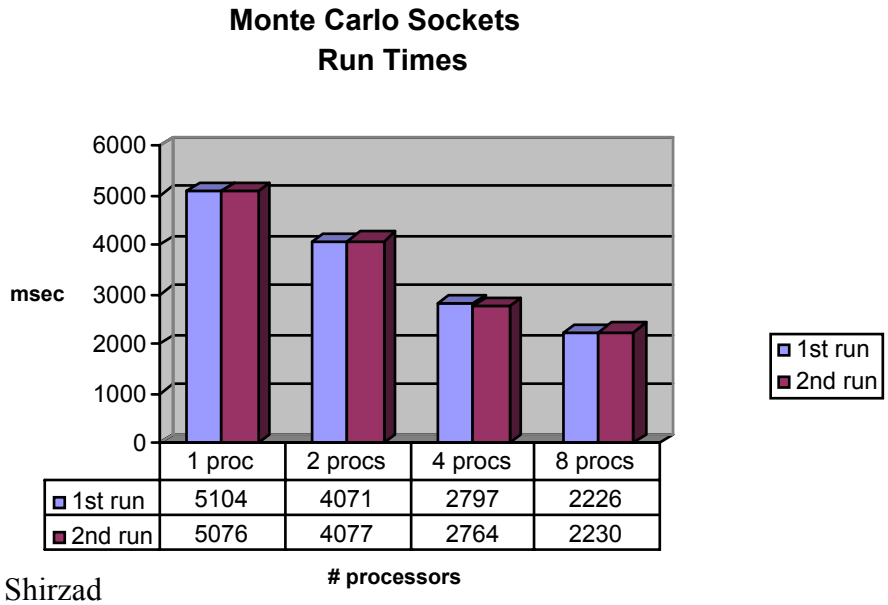
To run MonteCarlo on 4 systems:

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 0
mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - GridTcpMonteCarlo 4 0

[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 1
mnode1 - 0 mnode0 - 2 mnode2 - 3 mnode3 - GridTcpMonteCarlo 4 0

[uwagent@mnode2 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 2
mnode2 - 0 mnode0 - 1 mnode1 - 3 mnode3 - GridTcpMonteCarlo 4 0

[uwagent@mnode3 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 3
mnode3 - 0 mnode0 - 1 mnode1 - 2 mnode2 - GridTcpMonteCarlo 4 0
```

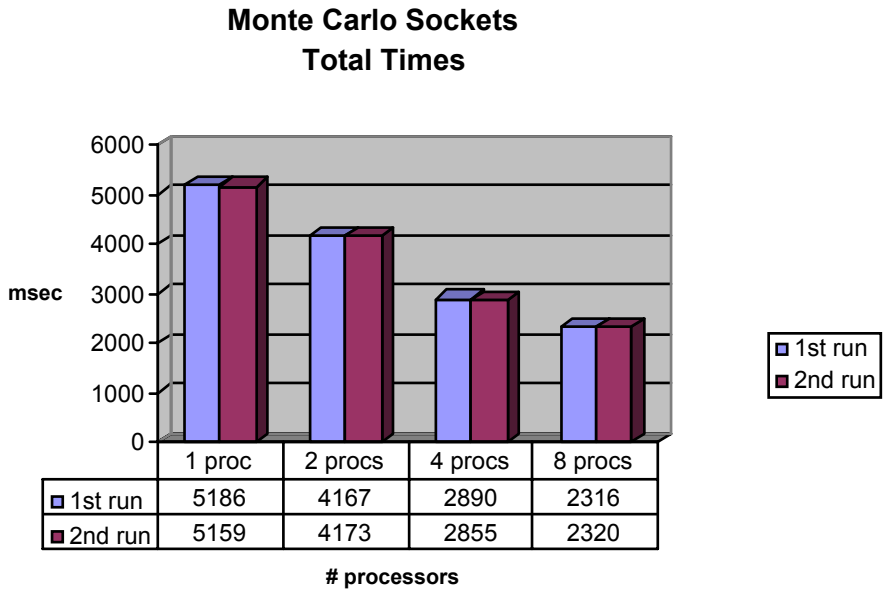
Results

Results were taken with 1, 2, 4 and 8 processors in two separate runs. The full text of the results is in the Appendix.

Sockets

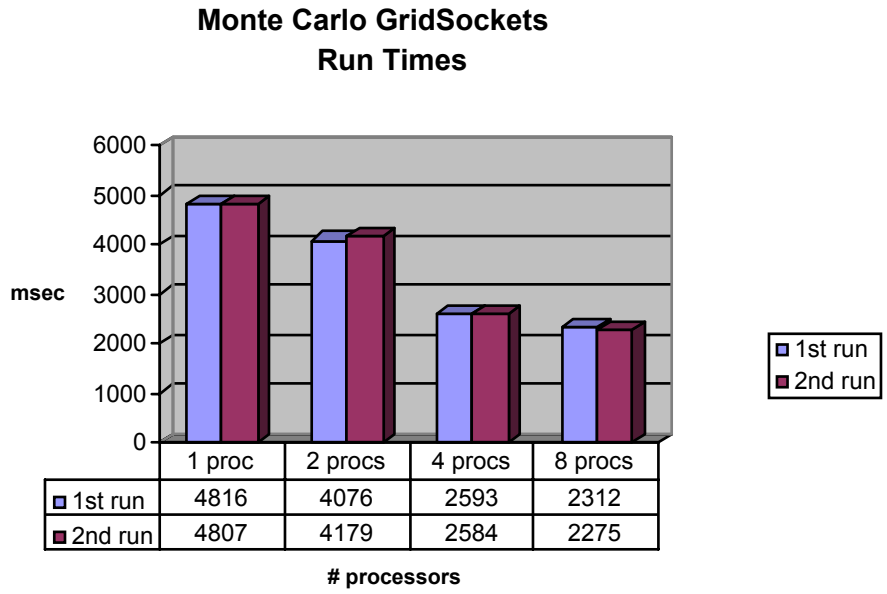


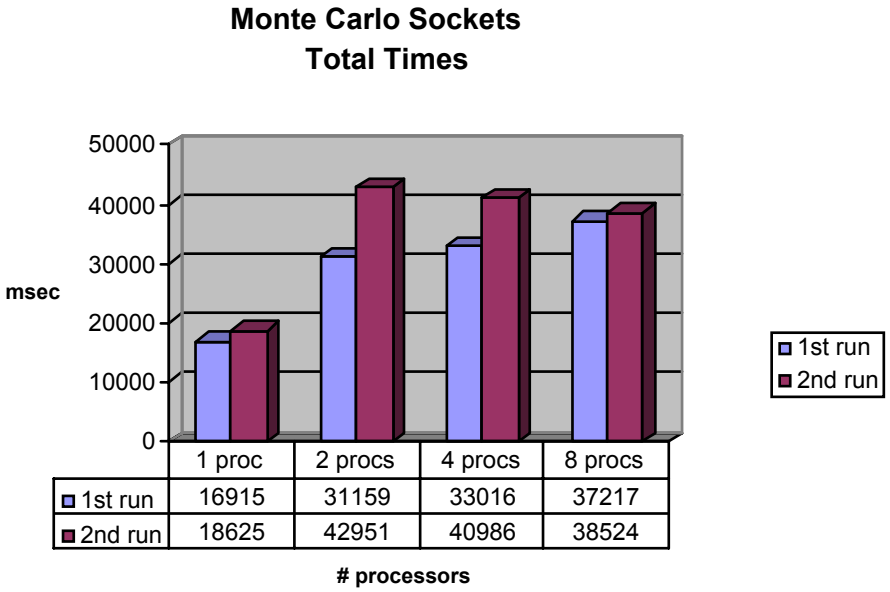
	1 proc	2 procs	4 procs	8 procs
1st run	5104	4071	2797	2226
2nd run	5076	4077	2764	2230



GridSockets

The Total Time results on the GridSockets version of MonteCarlo was much higher than expected. The timers were started in exactly the same places as in the Sockets version, and in the same places as in the GridTcpRayTracer class. RayTracer's times (both Sockets and GridTcpSockets) are all higher than the MonteCarlo GridSockets Total Time.





Sugarscape

Implementation

All class files have been converted from C++ to Java, except sim.cpp and the packAgent and unpackAgent methods in Mpi.cpp. When Sugarscape is converted to Sockets, those methods and sim.cpp are likely to be completely different, which is why I left them to the last. I converted around 1300 lines of code with approximately 600 left unconverted. Most of those 600 lines are for MPI communication.

Instead of using compilation options for Profiling, Debug and Message, global variables of those names have been declared in the Global.java class that can be set to true or false before compilation.

Issues

C++ Compilation

The C++ version of Sugarscape wouldn't compile immediately when moved to medusa. The following changes had to be made before it would compile.

```
Changed line 164 CommonVar.cpp
double d( sqrt(( sx - x ) * ( sx - x ) + ( sy - y ) * ( sy - y ) ) );

to
double d( sqrt((double)( ( sx - x ) * ( sx - x ) + ( sy - y ) * ( sy -
y )) ) );
```

```
Changed line 291 and 201 sim.cpp
for( int i = 0 ; i < int( log( mpi->getSize() ) / log( 2 ) ) * 2 ; i++
)

to
for( int i = 0 ; i < int( log( double(mpi->getSize() )) / log( 2.0 ) )
* 2 ; i++ )

lines 207 and 209
int( log(mpi->getSize() ) / log( 2 ) ) * 2;

to
int( log( double(mpi->getSize() )) / log( 2.0 ) ) * 2;
```

C++ Execution

The C++ version of Sugarscape ran fine on medusa with only one processor. The full results are shown in Appendix B. However, it would not run on more than one system in the medusa cluster. It gives the following errors:

```
[uwagent@mnode1 Sugarscape]$ ./run 2
mnode3: Connection refused
p0_16371:  p4_error: Child process exited while making connection to
remote process on mnode3: 0
/usr/local/bin/mpirun: line 1: 16371 Broken pipe
/home/uwagent/MA/applications/mpi/Sugarscape/sugar "20000" "900" "288"
-p4pg /home/uwagent/MA/applications/mpi/Sugarscape/PI16291 -p4wd
/home/uwagent/MA/applications/mpi/Sugarscape
```

Sugarscape was able to run on the uw1-320-lab systems on more than one system, and so I began converting it to java despite it not working on medusa.

Appendix A– MonteCarlo Results

MPJ Results (prunjava)

1 Processor

```
[uwagent@mnode0 section3]$ prunjava 1 JGFMonteCarloBenchSizeA
Java Grande Forum MPJ Benchmark Suite - Version 1.0 - Section 3 - Size
A
Executing on 1 process

Validation failed
 expectedReturnRate= -0.03335313452664269  4.4531149638711576E-5  0
Section3:MonteCarlo:Run:SizeA   5.138 (s)        865.1226
(Samples/s)
Section3:MonteCarlo:Total:SizeA 5.337 (s)        0.18737118
(Solutions/s)
```

8 Processors

```
[uwagent@mnode0 section3]$ prunjava 8 JGFMonteCarloBenchSizeA
Java Grande Forum MPJ Benchmark Suite - Version 1.0 - Section 3 - Size
A
Executing on 8 processes

Validation failed
 expectedReturnRate= -0.03335313452664269  4.4531149638711576E-5  0
Section3:MonteCarlo:Run:SizeA   10.176 (s)       436.8121
(Samples/s)
Section3:MonteCarlo:Total:SizeA 11.201 (s)       0.089277744
(Solutions/s)
Timeout in waiting for processes to exit, 1 left.  This may be due to a
defective
rsh program (Some versions of Kerberos rsh have been observed to have
this
problem).
This is not a problem with P4 or MPICH but a problem with the operating
environment.  For many applications, this problem will only slow down
process termination.
```

Sockets Results

Results were collected in two separate runs.

First Run

1 Processor

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 1 0
nprocess = 1 rank = 0
```

```
start totalTimer
start runTimer
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 5.104(s), 870.8855799373041(Calculations/s)
TotalTime: 5.104(s), 0.19592476489028213(Solutions/s)
Run Time = 5104
Total Time = 5186
```

2 Processors

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 2 0 mnode1
serverNames[0] =mnode1
nprocess = 2 rank = 0
server[0] = mnode1
MonteCarloBench.serverNames[0] = mnode1
socket[1] to mnode1 established
start totalTimer
start runTimer
rank 0 <- rank 1 completed
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 4.071(s), 1091.8693195774995(Calculations/s)
TotalTime: 4.071(s), 0.24563989191844757(Solutions/s)
Run Time = 4071
Total Time = 4167
```

4 Processors

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 4 0 mnode1
mnode2 mnode3
serverNames[0] =mnode1
serverNames[1] =mnode2
serverNames[2] =mnode3
nprocess = 4 rank = 0
server[0] = mnode1
server[1] = mnode2
server[2] = mnode3
MonteCarloBench.serverNames[0] = mnode1
socket[1] to mnode1 established
MonteCarloBench.serverNames[1] = mnode2
socket[2] to mnode2 established
MonteCarloBench.serverNames[2] = mnode3
socket[3] to mnode3 established
start totalTimer
start runTimer
rank 0 <- rank 1 completed
rank 0 <- rank 2 completed
rank 0 <- rank 3 completed
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 2.797(s), 1589.2027171969967(Calculations/s)
TotalTime: 2.797(s), 0.3575259206292456(Solutions/s)
Run Time = 2797
Total Time = 2890
```

8 Processors

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 8 0 mnode1
mnode2 mnode3 mnode4 mnode5 mnode6 mnode7
serverNames[0] =mnode1
serverNames[1] =mnode2
serverNames[2] =mnode3
serverNames[3] =mnode4
serverNames[4] =mnode5
serverNames[5] =mnode6
serverNames[6] =mnode7
nprocess = 8 rank = 0
server[0] = mnode1
server[1] = mnode2
server[2] = mnode3
server[3] = mnode4
server[4] = mnode5
server[5] = mnode6
server[6] = mnode7
MonteCarloBench.serverNames[0] = mnode1
socket[1] to mnode1 established
MonteCarloBench.serverNames[1] = mnode2
socket[2] to mnode2 established
MonteCarloBench.serverNames[2] = mnode3
socket[3] to mnode3 established
MonteCarloBench.serverNames[3] = mnode4
socket[4] to mnode4 established
MonteCarloBench.serverNames[4] = mnode5
socket[5] to mnode5 established
MonteCarloBench.serverNames[5] = mnode6
socket[6] to mnode6 established
MonteCarloBench.serverNames[6] = mnode7
socket[7] to mnode7 established
start totalTimer
start runTimer
rank 0 <- rank 1 completed
rank 0 <- rank 2 completed
rank 0 <- rank 3 completed
rank 0 <- rank 4 completed
rank 0 <- rank 5 completed
rank 0 <- rank 6 completed
rank 0 <- rank 7 completed
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 2.226(s), 1996.8553459119496(Calculations/s)
TotalTime: 2.226(s), 0.44923629829290207(Solutions/s)
Run Time = 2226
Total Time = 2316
```

Second Run

1 Processor

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 1 0
nprocess = 1 rank = 0
start totalTimer
```

```
start runTimer
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 5.076(s), 875.6895193065407(Calculations/s)
TotalTime: 5.076(s), 0.19700551615445233(Solutions/s)
Run Time = 5076
Total Time = 5159
```

2 Processors

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 2 0 mnode1
serverNames[0] =mnode1
nprocess = 2 rank = 0
server[0] = mnode1
MonteCarloBench.serverNames[0] = mnode1
socket[1] to mnode1 established
start totalTimer
start runTimer
rank 0 <- rank 1 completed
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 4.077(s), 1090.2624478783418(Calculations/s)
TotalTime: 4.077(s), 0.24527839097375523(Solutions/s)
Run Time = 4077
Total Time = 4173
```

4 Processors

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 4 0 mnode1
mnode2 mnode3serverNames[0] =mnode1
serverNames[1] =mnode2
serverNames[2] =mnode3
nprocess = 4 rank = 0
server[0] = mnode1
server[1] = mnode2
server[2] = mnode3
MonteCarloBench.serverNames[0] = mnode1
socket[1] to mnode1 established
MonteCarloBench.serverNames[1] = mnode2
socket[2] to mnode2 established
MonteCarloBench.serverNames[2] = mnode3
socket[3] to mnode3 established
start totalTimer
start runTimer
rank 0 <- rank 1 completed
rank 0 <- rank 2 completed
rank 0 <- rank 3 completed
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 2.764(s), 1608.1765557163533(Calculations/s)
TotalTime: 2.764(s), 0.36179450072358904(Solutions/s)
Run Time = 2764
Total Time = 2855
```

8 Processors

```
[uwagent@mnode0 MonteCarlo]$ java MonteCarloBenchSizeA 8 0 mnode1
mnode2 mnode3 mnode4 mnode5 mnode6 mnode7
serverNames[0] =mnode1
serverNames[1] =mnode2
serverNames[2] =mnode3
serverNames[3] =mnode4
serverNames[4] =mnode5
serverNames[5] =mnode6
serverNames[6] =mnode7
nprocess = 8 rank = 0
server[0] = mnode1
server[1] = mnode2
server[2] = mnode3
server[3] = mnode4
server[4] = mnode5
server[5] = mnode6
server[6] = mnode7
MonteCarloBench.serverNames[0] = mnode1
socket[1] to mnode1 established
MonteCarloBench.serverNames[1] = mnode2
socket[2] to mnode2 established
MonteCarloBench.serverNames[2] = mnode3
socket[3] to mnode3 established
MonteCarloBench.serverNames[3] = mnode4
socket[4] to mnode4 established
MonteCarloBench.serverNames[4] = mnode5
socket[5] to mnode5 established
MonteCarloBench.serverNames[5] = mnode6
socket[6] to mnode6 established
MonteCarloBench.serverNames[6] = mnode7
socket[7] to mnode7 established
start totalTimer
start runTimer
rank 0 <- rank 1 completed
rank 0 <- rank 2 completed
rank 0 <- rank 3 completed
rank 0 <- rank 4 completed
rank 0 <- rank 5 completed
rank 0 <- rank 6 completed
rank 0 <- rank 7 completed
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
runTime: 2.23(s), 1993.273542600897(Calculations/s)
TotalTime: 2.23(s), 0.4484304932735426(Solutions/s)
Run Time = 2230
Total Time = 2320
```

GridSockets Results

First Run

1 Processor

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 1 0
mnode0 -  GridTcpMonteCarlo 1 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
capture: funcId = 2
start runTimer
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 4.816(s), 922.9651162790698(Calculations/s)
TotalTime: 4.816(s), 0.20764119601328904(Solutions/s)
Run Time = 4816
Total Time = 16915
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

2 Processors

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 2 0
mnode0 - 1 mnode1 - GridTcpMonteCarlo 2 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
```

```
return : -1
testing snapshot
capture: funcId = 2
start runTimer
rank 0 <- myRank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 4.076(s), 1090.5299313052012(Calculations/s)
TotalTime: 4.076(s), 0.24533856722276745(Solutions/s)
Run Time = 4076
Total Time = 31159
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end

[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 2 1
mnode1 - 0 mnode0 - GridTcpMonteCarlo 2 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@b89838
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
```

```
Userprogclass#main: end
```

4 Processors

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 0
mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - GridTcpMonteCarlo 4 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
capture: funcId = 2
start runTimer
rank 0 <- myRank 1 completed
rank 0 <- myRank 2 completed
rank 0 <- myRank 3 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 2.593(s), 1714.2306209024296(Calculations/s)
TotalTime: 2.593(s), 0.3856536829926726(Solutions/s)
Run Time = 2593
Total Time = 33016
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 1
mnode1 - 0 mnode0 - 2 mnode2 - 3 mnode3 - GridTcpMonteCarlo 4 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
```

```
socket_s = GridSocket@b89838
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode2 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 2
mnode2 - 0 mnode0 - 1 mnode1 - 3 mnode3 - GridTcpMonteCarlo 4 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@b89838
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 2 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
```

```
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode3 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 3
mnode3 - 0 mnode0 - 1 mnode1 - 2 mnode2 - GridTcpMonteCarlo 4 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@b89838
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 3 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

8 Processors

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 0
mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
```

```
capture: funcId = 2
start runTimer
rank 0 <- myRank 1 completed
rank 0 <- myRank 2 completed
rank 0 <- myRank 3 completed
rank 0 <- myRank 4 completed
rank 0 <- myRank 5 completed
rank 0 <- myRank 6 completed
rank 0 <- myRank 7 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 2.312(s), 1922.5778546712804(Calculations/s)
TotalTime: 2.312(s), 0.43252595155709345(Solutions/s)
Run Time = 2312
Total Time = 37217
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 1
mnode1 - 0 mnode0 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 -  GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
```

```
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode2 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 2
mnode2 - 0 mnode0 - 1 mnode1 - 3 mnode3 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 2 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode3 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 3
mnode3 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
```

```
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 3 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode4 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 4
mnode4 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 4 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
```

```
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode5 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 5
mnode5 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@22c95b
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 5 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode6 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 6
mnode6 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5
mnode5 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@22c95b
server socket established
func_0 completed
funcId = 1
```

```
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 6 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode7 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 7
mnode7 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5
mnode5 - 6 mnode6 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@22c95b
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 7 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

Second Run

1 Processor

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 1 0
mnode0 -  GridTcpMonteCarlo 1 0funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
capture: funcId = 2
start runTimer
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 4.807(s), 924.6931558144372(Calculations/s)
TotalTime: 4.807(s), 0.20802995631370916(Solutions/s)
Run Time = 4807
Total Time = 18625
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

2 Processors

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 2 0
mnode0 - 1 mnode2 - GridTcpMonteCarlo 2 0funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
```

```
capture: funcId = 2
start runTimer
rank 0 <- myRank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 4.179(s), 1063.6515912897821(Calculations/s)
TotalTime: 4.179(s), 0.2392916965781287(Solutions/s)
Run Time = 4179
Total Time = 42951
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode2 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 2 1
mnode2 - 0 mnode0 - GridTcpMonteCarlo 2 0funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@22c95b
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

4 Processors

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 0
mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - GridTcpMonteCarlo 4 0funcsNum
= 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
capture: funcId = 2
start runTimer
rank 0 <- myRank 1 completed
rank 0 <- myRank 2 completed
rank 0 <- myRank 3 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 2.584(s), 1720.2012383900928(Calculations/s)
TotalTime: 2.584(s), 0.3869969040247678(Solutions/s)
Run Time = 2584
Total Time = 40986
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 1
mnode1 - 0 mnode0 - 2 mnode2 - 3 mnode3 - GridTcpMonteCarlo 4 0funcsNum
= 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@b89838
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
```

```
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode2 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 2
mnode2 - 0 mnode0 - 1 mnode1 - 3 mnode3 - GridTcpMonteCarlo 4 0funcsNum
= 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@93dcd
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 2 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

```
[uwagent@mnode3 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 4 3
mnode3 - 0 mnode0 - 1 mnode1 - 2 mnode2 - GridTcpMonteCarlo 4 0funcsNum
= 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@b89838
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 3 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

8 Processors

```
[uwagent@mnode0 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 0
mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
func_0 completed
funcId = 1
progArgsNum = 2
start totalTimer
return : -1
testing snapshot
capture: funcId = 2
start runTimer
rank 0 <- myRank 1 completed
rank 0 <- myRank 2 completed
```

```
rank 0 <- myRank 3 completed
rank 0 <- myRank 4 completed
rank 0 <- myRank 5 completed
rank 0 <- myRank 6 completed
rank 0 <- myRank 7 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
runTime: 2.275(s), 1953.8461538461538(Calculations/s)
TotalTime: 2.275(s), 0.43956043956043955(Solutions/s)
Run Time = 2275
Total Time = 38524
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode1 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 1
mnode1 - 0 mnode0 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 -  GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 1 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
```

```
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode2 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 2
mnode2 - 0 mnode0 - 1 mnode1 - 3 mnode3 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 2 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode3 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 3
mnode3 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 4 mnode4 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
```

```
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 3 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode4 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 4
mnode4 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 5 mnode5 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 4 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

```
[uwagent@mnode5 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 5
mnode5 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 6
mnode6 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 5 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode6 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 6
mnode6 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5
mnode5 - 7 mnode7 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@19efb05
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
```

```
rank 0 <- rank 6 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end


[uwagent@mnode7 GridTcp]$ java -cp GridTcp.jar:. UserProgWrapper - 8 7
mnode7 - 0 mnode0 - 1 mnode1 - 2 mnode2 - 3 mnode3 - 4 mnode4 - 5
mnode5 - 6 mnode6 - GridTcpMonteCarlo 8 0
funcsNum = 6
funcId = 0
progArgsNum = 2
argClass = [Ljava.lang.Class;@e53108
func_0 = public int GridTcpMonteCarlo.func_0(java.lang.String[])
initArgs = [Ljava.lang.Object;@19189e1
initArgs.length = 1
socket_s = null
socket_s = GridSocket@22c95b
server socket established
func_0 completed
funcId = 1
progArgsNum = 2
return : -1
testing snapshot
capture: funcId = 2
rank 0 <- rank 7 completed
return : -1
testing snapshot
capture: funcId = 3
Validation failed
 expectedReturnRate= 0.0  0.0333976656762814  0
return : -1
testing snapshot
capture: funcId = 4
return : -1
testing snapshot
capture: funcId = 5
return : -2
UserProgWrapper#funcScheduler: end
Userprogclass#main: end
```

Appendix B – Sugarscape C++ MPI Results

These results are just included for informational purposes. When Sugarscape has been completely converted to Java Sockets and GridSockets, it may be compared to these results.

1 Processor

```
[uwagent@mnode2 Sugarscape]$ ./run 1
mnode2 : CPU[0] argc = 4 agentNum = 20000 maxSimTime = 900 simSize =
288
Network construction
Agent injection
Agent's moveToIxIy
Cycle simulation
ArtificialSocieties: gvt = 51
ArtificialSocieties: gvt = 101
ArtificialSocieties: gvt = 151
ArtificialSocieties: gvt = 201
ArtificialSocieties: gvt = 251
ArtificialSocieties: gvt = 301
ArtificialSocieties: gvt = 351
ArtificialSocieties: gvt = 401
ArtificialSocieties: gvt = 451
ArtificialSocieties: gvt = 501
ArtificialSocieties: gvt = 551
ArtificialSocieties: gvt = 601
ArtificialSocieties: gvt = 651
ArtificialSocieties: gvt = 701
ArtificialSocieties: gvt = 751
ArtificialSocieties: gvt = 801
ArtificialSocieties: gvt = 851
ArtificialSocieties: Time = 55.4852
ArtificialSocieties: terminated.
***** main() *****
Initialize       : 0.00350889
Create SimSpace  : 0.149523
Inject agents    : 0.0640266
Clocking         : 10.5116
Agent simulation : 44.7005
Total            : 55.4292

***** agentSimulation() *****
Agent queue-in & out : 10.2418
Agent execution      : 32.6394
Exchange agents      : 1.81929

***** exchangeAgents() *****
exchangeWith   : 0.000274712
recvFromMyself : 1.81601
Allreduce      : 0
Total          : 1.81628
```

```
***** Clocking *****
Node clocking        : 10.5116
clocking send & recv : 0

***** Total information *****
Send & recv      : 1.81929
Agent execution  : 32.6394
Agent management : 10.3059
Msg management   : 10.6646
Total            : 55.4292

***** Send/recv size *****
Agent size : 92
Link size  : 16
Total send size : 0
Total recv size : 0
```