Name: ___________________________

Directions: MAKE SURE TO COPY YOUR ANSWERS TO A SEPARATE SHEET FOR SENDING ME AN ELECTRONIC COPY LATER.

1. (60) The code below uses R Snow to implement a bucket sort similar to the OMP one in Sec. 1.4.2.6. See the comments at the beginning of the code. Fill in the blanks.

```
# bucket sort with sampling; sort vector x
# on cluster cls; data assumed to be fairly
# uniformly distributed between a and b,
# exclusive; return value is sorted x

bsort <- function(cls,x,a,b) {
   ncls <- length(cls)
   intwidth <- (b - a) / ncls
   # ship needed objects to workers
   clusterExport(cls, ________  // blank (a)
      envir=environment())
   # have all workers set their ID
   clusterApply(cls,
      ________ )  // blank (b)
   # have all workers set their intervals
   clusterEvalQ(cls, ________ ) // blank (c)
   # sort locally at workers
   sortedchunks <-
      clusterEvalQ(cls,
         ________ )  // blank (d)
   # wrap up
   ________  // blank (e)
}

setmyid <- function(i) {
   myid <<-  i
}

setmyinterval <- function() {
   mylow <<- a + (myid-1) * intwidth
   myhigh <<- a + myid * intwidth
}

sortmine <- function() {
   myx <-  ________  // blank (f)
   sort(myx)
}
```

2. Fill in the blanks with terms from our course.

(a) (10) The term used for a parallel application that presents no coding challenge, due to being easily parallelizable, it is called ______________.

(b) (10) When we are worried whether a certain parallel algorithm will work well on very large hardware (e.g. many cores), we ask whether it is ______________.

(c) (10) Associating each thread with a specific core is called ______________.

3. (10) Consider a *ring network*. Here the nodes are arranged in a circle, with serial links connecting successive nodes. When a node receives a packet, it checks whether this node is the intended destination. If so, it accepts the packet, but if not, it forwards to the next node. Packets can be transmitted simultaneously on the various links. Packet motion is one direction, so counterclockwise. There is a processing delay at each node. Which is true of the following when an extra node is added?

(i) Both latency and bandwidth will increase.

(ii) Latency will increase but bandwidth will decrease.

(iii) Latency will decrease but bandwidth will increase.

(iv) Both latency and bandwidth will decrease.

Solutions:

1.

```
# bucket sort with sampling; sort vector x
# on cluster cls; data assumed to be
# fairly uniformly distributed between
# a and b, exclusive; return value is sorted x

bsort <- function(cls,x,a,b) {
   ncls <- length(cls)
   intwidth <- (b - a) / ncls
   # ship needed objects to workers
   clusterExport(cls,
      c("x","a","b","intwidth",
      "setmyid","setmyinterval","sortmine"),
      envir=environment())
   # have all workers set their ID
   clusterApply(cls,1:ncls,setmyid)
   # have all workers set their intervals
   clusterEvalQ(cls,setmyinterval())
   # sort locally at workers
   sortedchunks <- clusterEvalQ(cls,sortmine())
   # wrap up
   Reduce(c,sortedchunks)
}

setmyid <- function(i) {
   myid <<- i
}

setmyinterval <- function() {
   mylow <<- a + (myid-1) * intwidth
   myhigh <<- a + myid * intwidth
}

sortmine <- function() {
   myx <- x[x > mylow & x <= myhigh]
   sort(myx)
}
```

2a. embarrassingly parallel

2b. scalable

2c. processor affinity

3. (i)