Parallel statistical computing

Why is this leading to a growing role for parallel computing?

"According to Wikipedia, power consumption by computers became a concern so parallel computing became more popular. Also, the alternative had physical restraints."

"Processors have developed much over the last decade and also libraries have been created which allows users to utilize all the cores in a machine or different nodes over a cluster to efficiently solve a computationally difficult problem. With the growing size of data in this era of big data, this improvement in computer hardware has driven computationally difficult challenges in statistics to be carried on rather easily by using parallel computing."

Define: process, thread, core, node

"Process: the execution of a set of instructions from a computer program

Thread: a subtask in a parallel program; the smallest subset of instructions that can be run on a core

Core: a processing unit on a computer chip which is able to run one thread

Node: a single machine in a parallel computing cluster."

"Process: Instance of a computing program that can be executed by one or many threads, possibly on different cores or even different nodes.

Thread: Within a core, it is possible to make your OS believe there are more than one CPU, called logical CPUs, which acts virtually in the same way as multiple cores. Inparticular, multi-threading allows running multiple processes on a single physical core.

Core: Within a physical processor chip, there can multiple processing units defined physically in the chip. Unless multi-threading is used, a core can only run one process at any given time.

Node: A node is the physical computer containing the essentials for it to run and possibly communicate with other nodes (processors, memory, storage, power, buses, etc.) As for computing, a node can contain multiple processors, each containing multiple cores which can contain multiple logical threads."

What common statistical computing tasks are embarassingly parallel?

"Bootstrap for confidence intervals. Power calculation in simulation."

"Running simulations of an experiment seems to be the most natural application, where you are doing the exact same task each time and recording the result with minimalcommunication needed."

"K-fold cross-validation is one common procedure in statistics that would benefit from using parallelization. Once the data is divided into k test/training splits, the procedure can run independently on each split."

Explain the relative speeds for `foreach`

"The quickest speed was time2. first one does all on one core, the second splits them up into ten threads, etc. The reason the middle one is the fastest is because although having more threads makes generating them faster, when there are more threads there is also more time taken for communication between them."

"We see that the time to generate 10^8 *standard normal random variables initially in-creased, when dividing the task among 10 processors, it takes longer, potentially due to just the added cost of communication between cores required for parallelization. When dividing into 100 or 1,000 independent tasks, the run time decreases, which makes sense if at least some of these can be done in parallel. Then the run time increases againwhen dividing into 10,000 independent tasks, potentially because my computer doesnthave the ability to run many tasks at the same time and again the communication time is taken into account."*

common statistical computing tasks benefitting greatly from `foreach`.

"I think it can accelerate the computing, but not so greatly."

"I always use foreach when doing simulations where I am evaluating performance of certain statistical methods by taking average of several multiple results."

"In deep learning statistical estimation related to convolution of neural networks would benefit deeply though parallel computing."

Previous experience you have had with computing on a cluster

none: 2
507: 8 (Cavium cluster, and a bit of Great Lakes)
Some cluster use outside of STATS 507: 5

"I run program on a cluster [as an undergraduate]. The learning process is slow, but after I get used to using it, it is really fast."

What parallel statistical computing tasks are more appropriate for Hadoop than for `foreach`?

"I think one of the differences is that when using foreach, each sub-task in the for loop would be independent with others; while if you are working on a multi-stage task, it will be a good idea to transform that one into a mapreduce procedure."

"Hadoop is more appropriate for those tasks that different nodes need to communicate with each other."

"MapReduce is more relevant for 'divide-and-conquer' where summaries are sufficient while foreach is more relevant is the output of each iteration is required. For example, if you need some average across multiple runs, then MapReduce is sufficient, but if you want to be able to more post-hoc analysis on the output, MapReduce loses most of the information so methods like foreach will be more appropriate."

"In Hadoop framework, the map combine and reduce operations can break down an embarrasingly parallel task to simpler ones and combine them in each core and finally reduce efficiently which can be better in terms of efficiency compared to a foreach."

"In Hadoop there is a inbuilt .combine() operation which reduces the data in several intermediate steps. Hence gathering data becomes efficient in Hadoop. Hence for all types of statistical computing in need of parallel computing, we should use Hadoop."

People argued in favor of Hadoop. However, `foreach` has a .combine() operation, and can implement a map-reduce paradigm. If you are working in R and don't have massive distributed data, `foreach` provides similar functionality to Hadoop MapReduce.