CS 2150 – miniGoogle
Project Requirements

To secure full credit, the following requirements must be met.

1. **The design and implementation of a fault-tolerant name server/port mapper**. A worker must register with the nameserver/portmapper as either mapper or reducer or both, depending on your design. The design must tolerate the failure of the main or standby server without disruption to the system. Careful considerations must be given to the replication of the name server, to ensure consistency in the presence of failure.

2. **The design and implementation of a socket-based infrastructure to support miniGoogle functionality**. It is advised to use UDP sockets with a thin layer of reliability for the transfer of data in bulk, and TCP sockets, for other communication needs, depending on your design.

3. **The design and implementation of an indexing component to index data items.** The indexing component of your design must strike an appropriate balance to achieve high levels of parallelism and efficient use of the computing cluster. A cluster of 16 servers will be available for this project. The unit of work per indexer can vary from a chapter to a page, or even a finer level of granularity. You need to also think how a distributed structure of directories/folders will be designed to support efficient work assignment and execution, while avoiding excessive contention among workers. Your system should detect and avoid duplicate indexing of the same item.

4. **The design and implementation of the distributed inverted index.** The design must enable a high level of parallelism and support both indexing and search queries efficiently.

5. **The design and implementation of search engine to handle search queries, using different patterns.** The system should support simultaneous indexing and searching queries, while preserving consistency of the distributed inverted index and guaranteeing correct results of the search query. You also need to develop a ranking system, including a set of criteria to rank-order hits and a display strategy to list this hits.

6. **The development and implementation of an interface to interact with the system.** A command line interface is sufficient for this project, as long as it allows submission of multiple queries simultaneously.

7. **The design and implementation of all the miniGoogle components, discussed above, on a Hadoop platform, using MapReduce.**

8. The students should include log files of their experiments on the Hadoop cluster. It is highly recommend to carry out extensive testing on the Hadoop cluster, before submission. Results obtained by running the program on your own computer will not be sufficient. It has been the case that programs that work on the student's local computing environment fail on an actual Hadoop cluster. It is, therefore, the student's responsibility to debug the program on the Hadoop cluster before final submission.

9. Grading of this type of projects is time consuming. To ensure fairness, while minimizing grading time, the project submission should include the project software, a compiled version of the project software that captures all functionalities implemented in this project, a make file and a "how-to?" instruction manual to compile and execute the project on a Hadoop cluster. Failure to provide this information will be heavily penalized.

10. Although not required, you may consider support for "speculative execution", whereby when tasks in a job are coming to a close, redundant copies of the remaining tasks are scheduled across several nodes which do not have other work to perform. This ensures that the same input can be processed multiple times in parallel, to exploit differences in machine capabilities.