1. Vidal, David et. al. “A parallel workload balance and memory efficient lattice-Boltzmann algorithm with single unit BGK relaxation time for laminar Newtonian flows.” *Computers and Fluids.* Vol. 39 No. 8, 1411-1423, 2010.

**Summary:** This paper is useful because it proposes a method for fusing the lattice Boltzmann collision and propagation steps into one using shift algorithm, which will reduce computation time per time step.

1. Vidal, David et.al. “On improving the performance of large parallel lattice Boltzmann flow simulations in heterogeneous porous media.” *Computers and Fluids*. Vol. 39 No. 2, 324-337, 2010.

**Summary:** This paper talks about a scheme for the propagation step that reduces memory usage. It discusses a method for reducing inter-process communication by only transferring velocities per point which will be used by the recipient processors. This reduces inter-process communication, which could be expensive and complex as the lattice size increases.

1. Wang J, Zhang X, Bengough AG, Crawford JW. Domain-decomposition method for parallel lattice Boltzmann simulation of incompressible flow in porous media. *Phys Rev E* 2005;72:016706–11.

**Summary:** This paper suggests a smarter way of domain decomposition for processors by dividing domains based on the number of computation points to avoid load imbalance.

1. Latt, Jonas. “The Hydrodynamic Limit of the Lattice Boltzmann Equation.” *Thesis, University of Geneva.* 2007.

**Summary:** This is a well-rounded and in-depth examination of the mathematical theory and numerical execution of the lattice Boltzmann method, and has been used as an introduction and resource for the algorithm during the project.

1. Succi, Sauro. “Lattice Boltzmann Method.” *Scholarpedia*. <http://www.scholarpedia.org/article/Lattice_Boltzmann_Method>.

**Summary:** This resource was useful as a simple introduction to the lattice Boltzmann equation and the mathematical formalism in general behind the lattice Boltzmann algorithm.

1. Latt, Jonas. “Technical report: How to implement your DdQq dynamics with only q variables per node.” <http://www.palabos.org/palabos/downloads/plb-tr1.pdf>.

**Summary:** This report provides us with a proposed method for implementing the lattice Boltzmann method without the need for temporary memory storage. Although our algorithm must be revised because the suggested boundary conditions in this report are different from our own, the report provided an idea from us to develop from.

1. Vladimirov, Andrey and Vadim Karpusenko. “Parallel Programming and Optimization with Intel Xeon Phi Coprocessors.” Sunnyvale, CA: Colfax Internation, 2013.

**Summary:** This book gives a thorough description of optimization on Xeon Phi coprocessors. We intend to use this to better understand how to extend our hybrid OpenMP/MPI implementation to take advantage of the huge performance boosts that the Xeon Phi’s could possibly offer us. It will be an essential tool for developing our methodology as well as learning the programming models that are necessary to implement our code for these coprocessors.

1. G. Bella, Salvatore Filippone, Nicola Rossi, Stefano Ubertini. “Using OpenMP on a Hydrodynamic Lattice Boltzmann Code.” EWOMP 2002.

**Summary:** Implementing OpenMP in functions with dependencies and nested loops may result in an imbalance in computational load between the threads. This paper finds that such code only experiences speedup if the number of threads is an even divisor of 10, and suggests methods of improving the code to get speedup for all numbers of threads.

1. Qing, Quinlan, Whaley, Qasem. “A Multi-Language Environment for Programmable Code Optimization and Empirical Tuning.” Washington D.C: US Department of Energy, 2013. <http://libraries.colorado.edu/record=b7838754~S4>.

**Summary:** This report provided a basis through which we could analyze our code and determine when a structure (such as a for loop) could be better interpreted by a compiler.

1. Micheal K. Gschwind. “Performing aggressive code optimization with an ability to rollback changes made by the aggressive optimizations.” Washington D.C: US Department of Energy, 2013. <http://libraries.colorado.edu/record=b7838754~S4>.

**Summary:** This report demonstrates particular scenarios in which compiler optimization may not achieve the desired results, for example in attempting to parallelize interdependent data series. It was useful in analyzing our code to determine when it was appropriate to implement SIMD.

1. Magee, Glen. “Code Optimization Techniques.” Washington D.C: US Department of Energy, 2013.  <http://libraries.colorado.edu/record=b6982669~S4>.

**Summary:** This resource provides concrete, concise examples of modern optimization and error correction, including though-provoking examples like the Reed-Solomon code.

1. Federico Massaioli, Giorgio Amati. “Achieving high performance in a LBM code using OpenMP.” *Fourth European Workshop on OpenMP*. Roma, 2002.

**Summary:** This paper argues that hardware limitations present an obstacle to the scalability of the typical implementation of lattice Boltzmann method to large problem sizes on a large number of processors. They suggest several substantial changes to the implementation of the algorithm which mitigate the problem and which we may try if time allows.

1. Kevin Tubbs. “Lattice Boltzmann Modeling for Shallow Water Equations using High Performance Computing.” *Thesis, Louisiana State University*. 2010.

**Summary:** This thesis suggests a novel way to access data in the lattice Boltzmann method. They obtain good performance on cache-based architectures by dividing the data into blocks which can fit into cache and be utilized repeatedly.