FPGA Accelerated Phased Array Design Using the Ant Colony Optimization

Ozlem Kilic*[†], Dave Barger[††]
[†]The Catholic University of America, Washington DC 20064
[††]Mikira Systems, Potomac MD, 20854

There is an increasing need in faster computations without sacrificing the accuracy of full wave analysis in the electromagnetics community as the applications demand ambitious on-site performance within the operational environment of these devices, such as smart antennas, advanced rf materials, etc. This growing interest in simulating the performance of components and platforms as a system requires modeling large scale structures with fine features. Conventional full wave simulation techniques often are not capable of solving such problems due to limitation in computational resources. Often, researchers resort to asymptotic or hybrid techniques in order to obtain a "reasonably accurate" solution. The supercomputing systems which can potentially handle such problems are not commonly available due to their cost. Recently, hardware accelerated computing has been gaining momentum due its applicability to parallel computations while using a fraction of the power requirements of the conventional microprocessors and requiring much less cost in comparison to supercomputers.

This paper investigates the use of field programmable gate arrays (FPGAs) in electromagnetic simulations. The amplitudes of a phased array antenna will be optimized to reduce interference in multi-beam satellite communication systems by implementing the optimization algorithm and the antenna pattern calculations on FPGAs. The ant colony optimization algorithm, which is a bio-inspired heuristic search method based on the survival skills of ants, will be applied to a phased array antenna such that nulls can be placed in the array factor to reduce interference from beams operating at the same frequency band. The ant colony optimization is an inherently parallel search method based on a number of ants simultaneously sampling the field (i.e. the optimization space) for the closest path between their nest and food (i.e. best cost function that fits the optimization criteria). In order to simulate a path in the search for the optimal values, the solution set is described as a binary string, where the probability of a one or zero at each bit defines different possible paths.

The algorithm is implemented on SGI's reconfigurable computational platform, Altix 450 where two Xilinx Virtex 4.0 FPGAs are utilized. Many processing elements will be created within the FPGAs in order to achieve the anticipated performance improvements. Linear Feedback Shift Registers (LFSRs) are used to generate the pseudo-random numbers used in the ACO algorithm. Multiple instantiations of these and other processing elements (e.g. multipliers, dividers, adders) and memory elements are used in this pipelined approach. Within the pipeline, a fan-out to parallel Multiply-Accumulators (MAC) provides simultaneous updates to the cost function. The back-end of the pipeline updates the probability that describes "pheromone" distribution for "ant" n-1 while the front-end of the pipeline proceeds with "ant" n cost function processing. The pipeline architecture avoids the Van Neumann bottle neck of funneling data and instructions through a shared interface. Functions of the processing elements are fixed and data is passed from one processing element to the next, eliminating the need to move data in and out of memory as a shared processing resource steps through the processing sequence of the algorithm.