A Genetic Algorithm Approach to Scheduling Communications for a Class of Parallel Space-Time Adaptive Processing Algorithms

Jack M. West and John K. Antonio

School of Computer Science
University of Oklahoma
200 Felgar Street
Norman, OK 73019
Phone: (405) 325-4624
{west, antonio}@ou.edu

Abstract. An important consideration in the maximization of performance in parallel processing systems is scheduling the communication of messages during phases of data movement to reduce network contention and overall communication time. The work presented in this paper focuses on off-line optimization of message schedules for a class of radar signal processing techniques know as space-time adaptive processing on a parallel embedded system. In this work, a genetic-algorithm-based approach for optimizing the scheduling of messages is introduced. Preliminary results indicate that the proposed genetic approach to message scheduling can provide significant decreases in the communication time.

1 Introduction and Background

For an application on a parallel and embedded system to achieve required performance given tight system constraints, it is important to efficiently map the tasks and/or data of the application onto the processors to the reduce inter-processor communication traffic. In addition to mapping tasks efficiently, it is also important to schedule the communication of messages in a manner that minimizes network contention so as to achieve the smallest possible communication time.

Mapping and scheduling can both – either independently or in combination – be cast as optimization problems, and optimizing mapping and scheduling objectives can be critical to the performance of the overall system. For parallel and embedded systems, great significance is placed on minimizing execution time (which includes both computation and communication components) and/or maximizing throughput.

The work outlined in this paper involves optimizing the scheduling of messages for a class of radar signal processing techniques known as space-time adaptive processing (STAP) on a parallel and embedded system. A genetic algorithm (GA) based approach for solving the message-scheduling problem for the class of parallel STAP algorithms is proposed and preliminary results are provided. The GA-based optimization is performed off-line, and the results of this optimization are static

schedules for each compute node in the parallel system. These static schedules are then used within the on-line parallel STAP implementation. The results of the study show that significant improvement in communication time performance are possible using the proposed approach for scheduling. Performance of the schedules were evaluated using a RACEway network simulator [6].

2 Overview of Parallel STAP

STAP is an adaptive signal processing method that simultaneously combines the signals received from multiple elements of an antenna array (the spatial domain) and from multiple pulses (the temporal domain) of a coherent processing interval [5]. The focus of this research assumes STAP is implemented using an element-space post-Doppler partially adaptive algorithm, refer to [5, 6] for details. Algorithms belonging to the class of element-space post-Doppler STAP perform filtering on the data along the pulse dimension, referred to as Doppler filtering, for each channel prior to adaptive filtering. After Doppler filtering, an adaptive weight problem is solved for each range and pulse data vector.

The parallel computer under investigation for this work is the Mercury RACE® multicomputer. The RACE® multicomputer consists of a scalable network of compute nodes (CNs), as well as various high-speed I/O devices, all interconnected by Mercury's RACEway interconnection fabric [4]. A high-level diagram of a 16-CN RACEway topology is illustrated in Figure 1. The interconnection fabric is configured in a fat-tree architecture and is a circuit switched network. The RACEway interconnection fabric is composed of a network of crossbar switches and provides high-speed data communication between different CNs. The Mercury multicomputer can support a heterogeneous collection of CNs (e.g., SHARC and PowerPCs), for more details refer to [6].

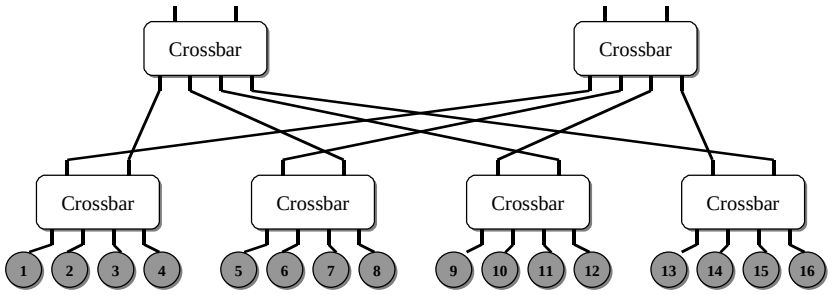

Fig. 1. Mercury RACE® Fat-Tree Architecture configured with 16 CNs.

Achieving real-time performance requirements for STAP algorithms on a parallel embedded system like the Mercury multicomputer largely depends on two major issues. First is determining the best method for distributing the 3-D STAP data cube across CNs of the multiprocessor system (i.e., the mapping strategy). Second is

determining the scheduling of communications between phases of computation. In general, STAP algorithms contain three phases of processing, one for each dimension of the data cube (i.e., range, pulse, channel). During each phase of processing, the vectors along the dimension of interest are distributed as equally as possible among the processors for processing in parallel. An approach to data set partitioning in STAP applications is to partition the data cube into sub-cube bars. Each sub-cube bar is composed of partial data samples from two dimensions while preserving one whole dimension for processing. The work here assumes a sub-cube bar partitioning of the STAP data cube, for further details refer to [6]. Figure 2 shows an example of how sub-cube partitioning is applied to partition a 3-D data cube across 12 CNs.

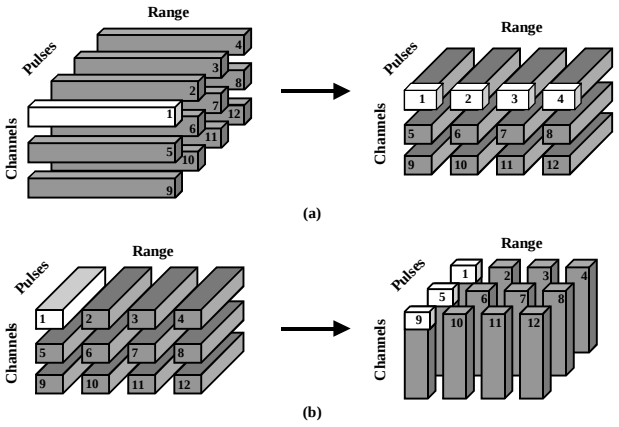

Fig. 2. Illustration of the sub-cube bar mapping technique for the case of 12 CNs. The mapping of the sub-cube bars to CNs defines the required data communications. (a) Example illustration of the communication requirements from CN 1 to the other CNs (2, 3, and 4) after completion of the range processing and prior to Doppler processing. (b) Example illustration of the communication requirements from CN 1 to other CNs (5 and 9) after the completion of Doppler processing and prior to adaptive weight processing.

During phases of data redistribution (i.e., communication) between computational phases, the number of required communications and the communication pattern among the CNs is dependant upon how the data cube is mapped onto the CNs. For example, in Figure 2(a) the mapping of sub-cube bars to CNs dictates that after range processing, CN 1 must transfer portions of it data sub-cube bar to CNs 2, 3, and 4. (Each of the other CNs, likewise, is required to send portions of their sub-cube bar to CNs on the same row.) The scheduling (i.e., ordering) of outgoing messages at each CN impacts the resulting communication time. For example, in Figure 2(a) note CN 1 could order its outgoing messages according to one of 3! = 6 permutations (i.e., [2,3,4], [3,2,4], etc.). Similarly, a scheduling of outgoing messages must be defined for each CN. Improper schedule selection can result in excessive network contention and thereby increase the time to perform all communications between processing phases. The focus in this paper is on optimization of message scheduling, for a fixed mapping, using a genetic algorithm methodology.

3 Genetic Algorithm Methodology

A GA is a population-based model that uses selection and recombination operators to generate new sample points in the solution space [3]. A GA encodes a potential solution to a specific problem on a chromosome-like data structure and applies recombination operators to these structures in a manner that preserves critical information. Reproduction opportunities are applied in such a way that those chromosomes representing a better solution to the target problem are given more chances to reproduce than chromosomes with poorer solutions. GAs are a promising heuristic approach to locating near-optimal solutions in large search spaces [3]. For a complete discussion of GAs, the reader is referred to [1, 3].

Typically, a GA is composed of two main components, which are problem dependent: the *encoding problem* and the *evaluation function*. The *encoding problem* involves generating an encoding scheme to represent the possible solutions to the optimization problem. In this research, a candidate solution (i.e., a chromosome) is encoded to represent valid message schedules for all of the CNs. The *evaluation function* measures the quality of a particular solution. Each chromosome is associated with a fitness value, which in this case is the completion time of the schedule represented by the given chromosome. For this research, the smallest fitness value represents the better solution. The "fitness" of a candidate is calculated here based on its simulated performance. In previous work [6, 7], a software simulator was developed to model the communication traffic for a set of messages on the Mercury RACEway network. The simulation tool is used here to measure the "fitness" (i.e., the completion time) of the schedule of messages represented by each chromosome.

Chromosomes evolve through successive iterations, called generations. To create the next generation, new chromosomes, called offspring, are formed by (a) merging two chromosomes from the current population together using a crossover operator or (b) modifying a chromosome using a mutation operator. Crossover, the main genetic operator, generates valid offspring by combining features of two parent chromosomes. Chromosomes are combined together at a defined crossover rate, which is defined as the ratio of the number of offspring produced in each generation to the population size. Mutation, a background operator, produces spontaneous random changes in various chromosomes. Mutation serves the critical role of either replacing the chromosomes lost from the population during the selection process or introducing new chromosomes that were not present in the initial population. The mutation rate controls the rate at which new chromosomes are introduced into the population. In this paper, results are based on the implementation of a position-based crossover operator and an insertion mutation operator, refer to [1] for details.

Selection is the process of keeping and eliminating chromosomes in the population based on their relative quality or fitness. In most practices, a roulette wheel approach, either rank-based or value-based, is adopted as the selection procedure. In a ranked-based selection scheme, the population is sorted according to the fitness values. Each chromosome is assigned a sector of the roulette wheel based on its ranked-value and not the actual fitness value. In contrast, a value-based selection scheme assigns roulette wheel sectors proportional to the fitness value of the chromosomes. In this paper, a ranked-based selection scheme is used. Advantages of rank-based fitness

assignment is it provides uniform scaling across chromosomes in the population and is less sensitive to probability-based selections, refer to [3] for details.

4 Numerical Results

In the experiments reported in this section, it is assumed that the Mercury multicomputer is configured with 32 PowerPC compute nodes. For range processing, Doppler filtering, and adaptive weight computation, the 3-D STAP data cube is mapped onto the 32 processing elements based on an 8×4 process set (i.e., 8 rows and 4 columns), refer to [2, 6]. The strategy implemented for CN assignment in a process set is raster-order from left-to-right starting with row one and column one for all process sets. (The process sets not only define the allocation of the CNs to the data but also the required data transfers during phases of data redistribution.) The STAP data cube consists of 240 range bins, 32 pulses, and 16 antenna elements.

For each genetic-based scenario, 40 random schedules were generated for the initial population. The poorest 20 schedules were eliminated from the initial population, and the GA population size was kept a constant 20. The recombination operators included a position-based crossover algorithm and an insertion mutation algorithm. A ranked-based selection scheme was assumed with the angle ratio of sectors on the roulette wheel for two adjacently ranked chromosomes to be $1+1/P$, where P is the population size. The stopping criteria were: (1) the number of generations reached 500; (2) the current population converged (i.e., all the chromosomes have the same fitness value); or (3) the current best solution had not improved in the last 150 generations.

Figure 3 shows the simulated completion time for three genetic-based message scheduling scenarios for the data transfers required between range and Doppler processing phases. Figure 4 illustrates the simulated completion time for the same three genetic-based message scheduling scenarios for the data transfers required between Doppler and adaptive weight processing phases. In the first genetic scenario (GA 1), the crossover rate (P_{xover}) is 20% and the mutation rate (P_{mut}) is 4%. For GA 2, P_{xover} is 50% and P_{mut} is 10%. For GA 3, P_{xover} is 90% and P_{mut} is 50%. Figures 3 and 4 provide preliminary indication that for a fixed mapping the genetic-algorithm-based heuristic is capable of improving the scheduling of messages, thus providing improved performance. All three genetic-based scenarios improve the completion time for both communication phases. In each phase, GA 2 records the best schedule of messages (i.e., the smallest completion time).

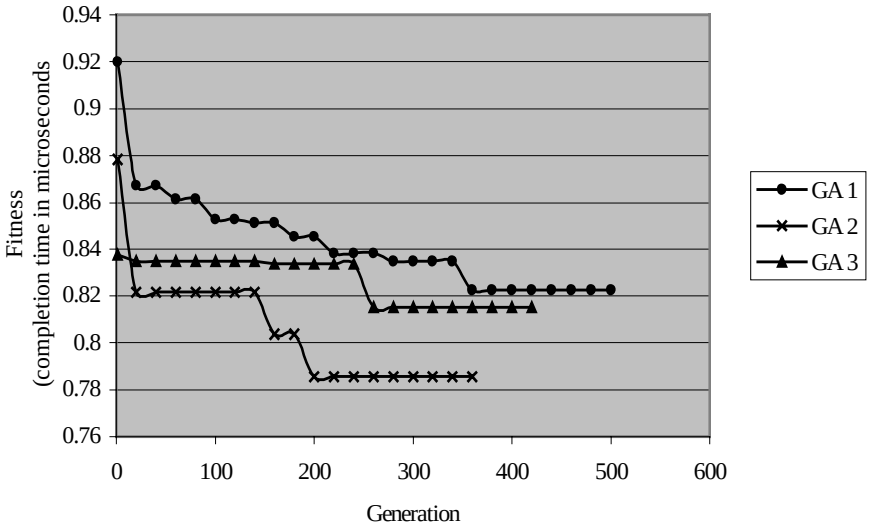

Fig. 3. Simulated completion time of the communication requirements for data redistribution after range processing and prior to Doppler processing for the parameters discussed in Section 4. For GA 1, the crossover rate (P_{xover}) = 20% and the mutation rate (P_{mut}) = 4%. For GA 2, P_{xover} = 50% and P_{mut} = 10%. For GA 3, P_{xover} = 90% and P_{mut} = 50%.

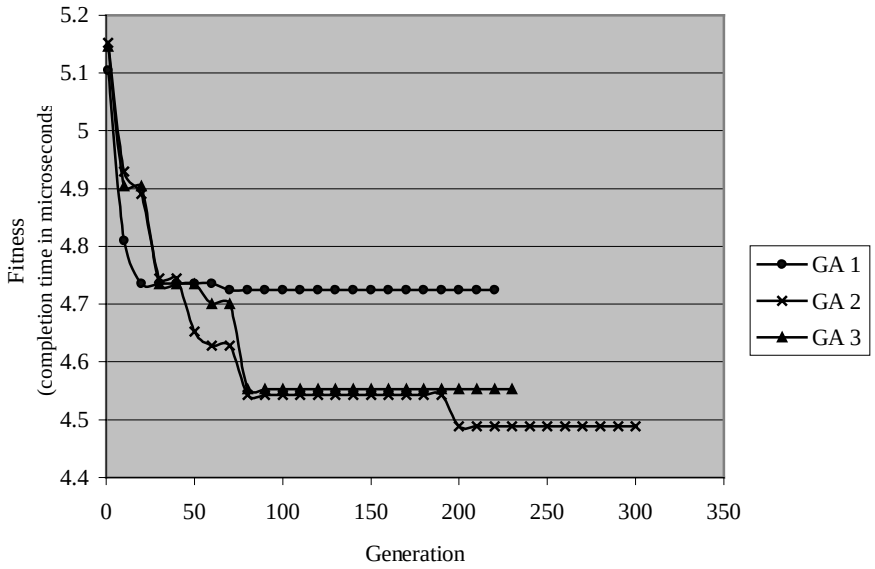

Fig. 4. Simulated completion time of the communication requirements for data redistribution after Doppler processing and prior to adaptive weight computation for the parameters stated in Section 4. For GA 1, the crossover rate (P_{xover}) = 20% and the mutation rate (P_{mut}) = 4%. For GA 2, P_{xover} = 50% and P_{mut} = 10%. For GA 3, P_{xover} = 90% and P_{mut} = 50%.

5. Conclusion

In conclusion, preliminary data demonstrates that off-line GA-based message scheduling optimization can provide improved performance in a parallel system. Future work will be conducted to more completely study the effect of changing parameters of the GA, including crossover and mutation rates as well as the methods used for crossover and mutation. Finally, future studies will be conducted to determine the performance improvement between a randomly selected scheduling solution and the one determined by the GA. In Figures 3 and 4, the improvements shown are conservative in the sense that the initial generations' performance on the plots represents the best of 40 randomly generated chromosomes (i.e., solutions). It will be interesting to determine improvements of the GA solutions with respect to the "average" and "worst" randomly generated solutions in the initial population.

Acknowledgements

This work was supported by DARPA under contract no. F30602-97-2-0297.

References

1. M. Gen and R. Cheng, *Genetic Algorithms and Engineering Design*, John Wiley & Sons, Inc., New York, NY, 1997.
2. M. F. Skalabrin and T. H. Einstein, "STAP Processing on a Multicomputer: Distribution of 3-D Data Sets and Processor Allocation for Optimum Interprocessor Communication," *Proceedings of the Adaptive Sensor Array Processing (ASAP) Workshop*, March 1996.
3. L. Wang, H. J. Siegel, V. P. Roychowdhury, and A. A. Maciejewski. "Task Matching and Scheduling in Heterogeneous Computing Environments Using a Genetic-Algorithm-Based Approach," *Journal of Parallel and Distributed Computing*, Special Issue on Parallel Evolutionary Computing, Vol. 47, No 1, pp. 8-22, Nov. 25, 1997.
4. The RACE Multicomputer, Hardware Theory of Operation: Processors, I/O Interface, and RACEway Interconnect, Volume I, ver. 1.3.
5. J. Ward, Space-Time Adaptive Processing for Airborne Radar, Technical Report 1015, Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, MA, 1994.
6. J. M. West, *Simulation of Communication Time for a Space-Time Adaptive Processing Algorithm Implemented on a Parallel Embedded System*, Master's Thesis, Computer Science, Texas Tech University, 1998.
7. J. M. West and J. K. Antonio, "Simulation of the Communication Time for a Space-Time Adaptive Processing Algorithm on a Parallel Embedded System," *Proceedings of the International Workshop on Embedded HPC Systems and Applications (EHPC '98)*, in *Lecture Notes in Computer Science 1388: Parallel and Distributed Processing*, edited by Jose Rolim, sponsor: IEEE Computer Society, Orlando, FL, USA, Apr. 1998, pp. 979-986.