Genetic Algorithm based Automatic Data Partitioning Scheme for HPF

Sunil Kumar Anand & Y. N. Srikant
Department of Computer Science and Automation
Indian Institute of Science, Bangalore
asunil@csa.iisc.ernet.in & srikant@csa.iisc.ernet.in

Abstract *The performance of a parallel program depends largely on its data partitions. So a good data partitioning scheme is the need of the time. However it is very difficult to arrive at a good solution as the number of possible data partitions for a given real life program is exponential in the size of the program. We present a heuristic technique for automatic data partitioning for HPF. Our approach is based on Genetic Algorithms and is very simple, yet very efficient to quickly find appropriate data partitions even for large programs with large number of alternatives for data distribution.It makes use of both static as well as dynamic data distribution with the main aim of reducing the overall execution time of the entire program.*

Problem Description

In *automatic data partitioning,* the compiler is responsible for deciding the partitioning of data. It analyzes the array access patterns in a program to arrive at the best partition by taking into consideration the architectural parameters of the parallel machine.

We view the solution to the automatic data partition problem as a search for finding the best data partition in the search space of all possible data partitions.

We view the programs as a collection of procedures each of which is comprised of various execution paths and each execution path is comprised of the computational phases.

To find the complete solution to the automatic data partition problem, we use genetic algorithms at different levels of the program as folows.

The *inter-procedure GA* is used at program level. The *intra-procedure GA* is used at procedure level and *intra-chain GA is used* at execution path level.

Genetic Algorithms at different levels

The chromosome encoding of the individuals at each level varies.

The *inter-procedure GA* starts with the startup/random distribution for each procedure. Then for each procedure, intra-procedure GA is called to find optimal data partition for that procedure. Then it evaluates the cost of the program. The GA stops if the minimum cost partition for the program has been found or there is no improvement even after few more generations, otherwise *reaching distribution analysis* for each procedure is done and in the next genration of intra-procedure GA is called with the distributions specified by reaching distribution analysis.

In each generation of the *intra-procedure GA*, we call intra-chain GA to optimize some of the execution paths. Then *crossover, mutation* and *fitness scaling* is done and the the cost of the procedure is evaluated to check whether next generation of the GA is needed or not.

In *intra-chain GA*, the data partitions for the execution paths are decided randomly. Then cossover, mutation and fitness scaling is done and the cost of the execution path is evaluated. GA stops in case minimum cost partition for the execution path is found or if there is no improvement even after few more generations.

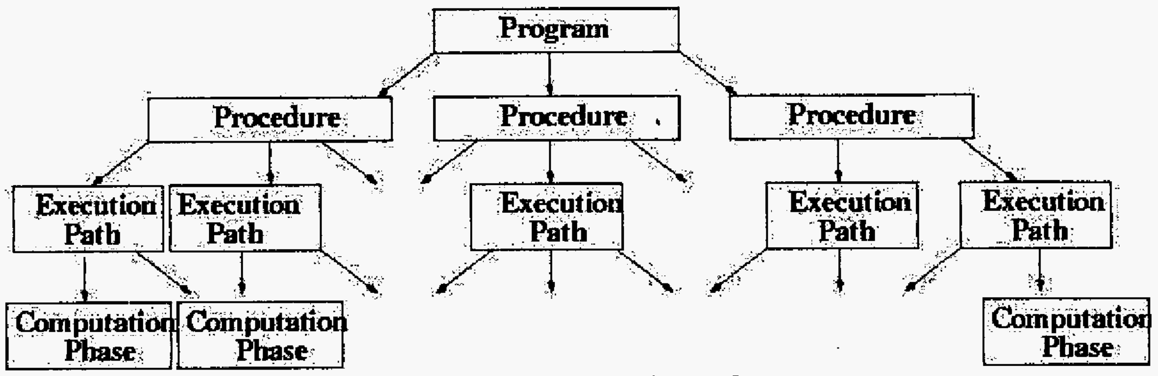

Fig: Program hierarchy

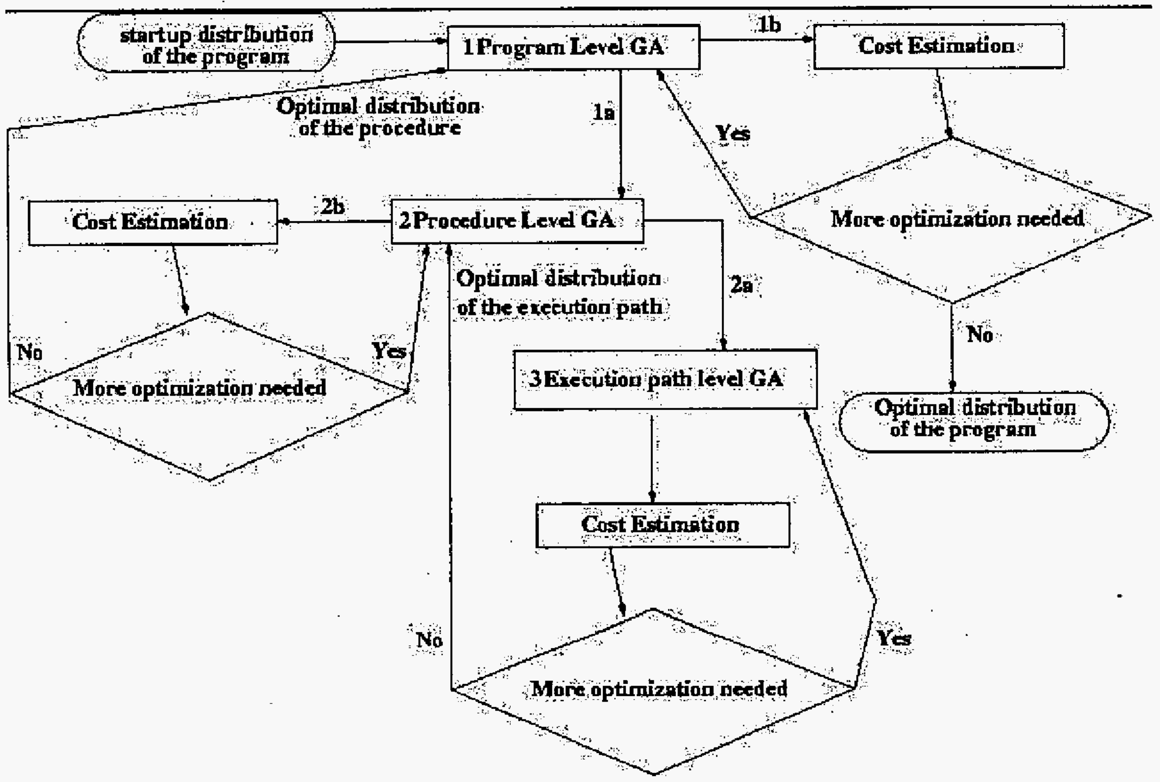

Fig: Flowchart of the GA at different levels of the program

Results	Conclusion
- The results on a 4-node and a 8-node cluster show that the GA based automatic data partitioning scheme gives better results as compared to that with some particular data distributions for the whole program. - The results of inter-procedure data partition are better than those with only intra-procedure data partition.	- By using Hierarchical Genetic Algorithms, we have explored and implemented a very simple algorithm to find a good solution for Automatic Data Partitioning problem. - Inter-procedural GA is required for a complete solution to the GA based automatic data partitioning problem.