An Architecture for Massive Parallelization of the Compact Genetic Algorithm

Fernando G. Lobo, Cláudio F. Lima, and Hugo Mártires

ADEEC-FCT, Universidade do Algarve
Campus de Gambelas, 8000 Faro, Portugal.
{flobo,clima}@ualg.pt, hmartires@myrealbox.com

Abstract. This paper presents an architecture which is suitable for a massive parallelization of the compact genetic algorithm. The approach is scalable, has low synchronization costs, and is fault tolerant. The paper argues that the benefits that can be obtained with the proposed methodology is potentially higher than those obtained with traditional parallel genetic algorithms.

1 Introduction

With a traditional parallel genetic algorithm (GA) implementation, population members need to be sent over a computer network, and that imposes a limit on how fast they can be [1]. In this paper, we investigate the parallelization of the compact genetic algorithm (cGA) [2], and take advantage of its compact representation of the population do develop a parallelization scheme which significantly reduces the communication overhead.

The cGA uses a probability vector as a model to represent the population. The vector can be stored with $\ell \times \log_2(N+1)$ bits (ℓ is the chromosome length, N is the population size), a different order of magnitude than the $\ell \times N$ bits needed to represent a population in a regular GA. Since communication costs can be drastically reduced, it makes sense to clone the probability vector to several computers, and let each computer work independently on solving the problem by running a separate cGA. Then, the different probability vectors need to be consolidated (or mixed) once in a while.

2 Massive Parallelization of the Compact GA

We have developed an asynchronous parallelization scheme which consists of a manager processor, and an arbitrary number of worker processors. Initially, the manager starts with a probability vector with 0.5 in all positions, just like in a regular cGA. After that, it sends the vector to all workers who are willing to contribute with CPU time.

Each worker processor runs a cGA on its own based on a local copy of the probability vector. Workers do their job independently and only interrupt the

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 412–413, 2004.

manager once in a while, after a predefined number of m fitness function evaluations have elapsed.

During the interruption period, a worker sends the accumulated results of the last m function evaluations as a vector of probability fluxes with respect to the original probability vector. Subsequently, the manager adds the probability fluxes to its own probability vector, and resends the resulting vector back to the worker. Meanwhile, other worker processors can continue doing their job nonstop, even though some of them are working with a slightly outdated vector.

Each worker processor can start and finish at any given point in time making the whole system fault tolerant. When a worker starts, it receives a copy of the manager's probability vector, which already contains the accumulated results of the other cGA workers. On the other hand, when a worker quits, we simply loose a maximum of m function evaluations, which is not a big problem.

We have conducted computer simulations to validate the proposed approach and observed a linear speedup with a growing number of processors (see Figure 1). Additional details of the simulations and a longer description of this work can be found elsewhere [3].

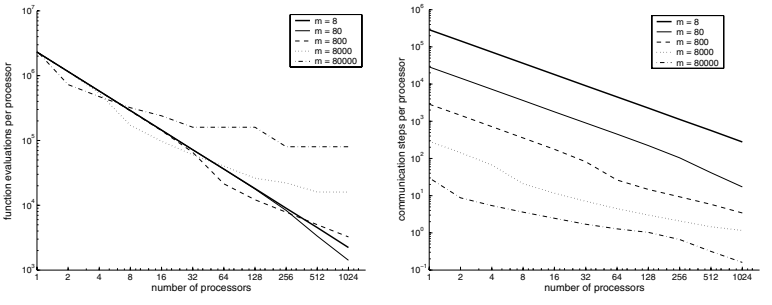

Fig. 1. On the left, we see the average number of function evaluations per processor. On the right, we see the average number of communication steps per processor.

Acknowledgements. This work was sponsored by FCT/MCES under grants POSI/SRI/42065/2001 and POCTI/MGS/37970/2001.

References

1. Cantú-Paz, E.: Efficient and accurate parallel genetic algorithms. Kluwer Academic Publishers, Boston, MA (2000)
2. Harik, G.R., Lobo, F.G., Goldberg, D.E.: The compact genetic algorithm. IEEE Transactions on Evolutionary Computation **3** (1999) 287–297
3. Lobo, F.G., Lima, C.F., Mártires, H.: An architecture for massive parallelization of the compact genetic algorithm. arXiv Report cs.NE/0402049 (2004) (Available at http://arxiv.org/abs/cs.NE/0402049).