The Master-Slave Architecture for Evolutionary Computations Revisited

Christian Gagné, Marc Parizeau, and Marc Dubreuil

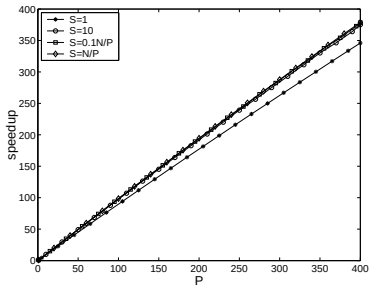
Laboratoire de Vision et Systèmes Numériques (LVSN),
Département de Génie Électrique et de Génie Informatique,
Université Laval, Québec (QC), Canada, G1K 7P4.
{cgagne,parizeau,dubreuil}@gel.ulaval.ca

The recent availability of cheap Beowulf clusters has generated much interest for Parallel and Distributed Evolutionary Computations (PDEC) [1]. Another often neglected source of CPU power for PDEC are networks of PCs, in many case very powerful workstations, that run idle each day for long periods of time. To exploit efficiently both Beowulfs and networks of heterogeneous workstations we argue that the classic master-slave distribution model is superior to the currently more popular island-model [1].

The key features of a good PDEC capable of exploiting networks of heterogeneous workstations are *transparency* for the user, *robustness*, and *adaptivity*. Transparency is essential to both the user of the PDEC and the user of the workstation, as none want to deal with the other. One way to implement such a PDEC is as a screen-saver. Robustness is very important because evolutionary computations may execute over long periods of time during which different types of failures are expected: *hard failures* caused by network problems, system crashes or reboots, and *soft failures* that stem from the use of the workstation for other tasks (e.g. when the user deactivates the screen-saver). Finally, adaptivity refers to the capability of the PDEC to exploit new or compensate for lost computing resources (dynamical network configuration). The classical island-model is not designed to deal with these features, essentially because populations (demes) are tightly coupled with processing nodes.

In contrast, the master-slave model has all required features. One issue that needs to be addressed, however, is its ability to scale with a large number of slave nodes, knowing that there is a communication bottleneck with the master node. In the rest of this short paper, we build a mathematical model of the master-slave and show that, given current Local Area Network (LAN) technologies, a quite large PDEC can be built before reaching this bottleneck.

For real world applications, assuming that the time needed for fitness evaluation is the dominant time factor for evolutionary algorithms, the speedup of a master-slave system over that of a single processor can be modeled by NT_f/T_p, where N is the population size, T_f is the time needed to evaluate the fitness of a single individual, and T_p is the time needed to evaluate all individuals using P processors. Possible distribution policies range from separating the population into P sets and sending each of them to a different slave, or sending the individuals one-by-one to available slaves until all are evaluated. Let S designate the average size of the sets that are sent to processing nodes, and $C = \lceil N/PS \rceil$ the

(a) 

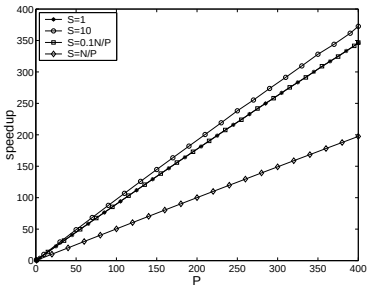
(b) 

Fig. 1. Speedup for $S = \{1, 10, \frac{0.1N}{P}, \frac{N}{P}\}$ as a function of number of processors $P \in [1, 400]$, (a) with no failure ($K = 0$); and (b) with five failures ($K = 5$). Other parameters are $N = 500000$, $T_f = 1$ s, $T_c = 1.4 \times 10^{-4}$ s, and $T_l = 0.1$ s.

number of communication cycles needed to complete a generation. Then:

$$T_p = \underbrace{CST_f}_{\text{computation}} + \underbrace{CPST_c}_{\text{communication}} + \underbrace{CT_l}_{\text{latency}} + \underbrace{T_k}_{\text{failure}} \tag{1}$$

where T_c is the transmission time for one individual, T_l the average connection latency, and T_k the delay associated to the observation of $K \in [0, P]$ failures:

$$T_k = \underbrace{(1 - 0.5^K)ST_f}_{\text{synchronization}} + \underbrace{KST_c}_{\text{communication}} + \underbrace{\lceil K/P \rceil T_l}_{\text{latency}} \tag{2}$$

The synchronization term assumes that each failure follows a Poisson process and occurs on average at half-way time during an evaluation cycle. Figure 1 presents the speedup curves of different S values in a plausible scenario, with (a) no failure, and (b) exactly five failures per generation. With no failure, the figure shows linear speedup close to optimal for all S except when $S = 1$, where performance starts to degrade given the relatively large latency T_l. However, when failures occur, the figure shows that a value $S = \frac{N}{P}$ no longer achieves linear speedup, and that the intermediary value of $S = 10$ (for this scenario) makes a good compromise between efficiency and robustness. These curves thus globally show that parameter S should be adjusted dynamically in order to optimize performance.

For the above model, using the conservative parameters of Figure 1 (that assumes 7 MB/sec peek throughput for 100Mb/sec LANs), it can be shown that a master-slave system of 7K processing nodes could be build, at least in theory, with a speedup of about 50% of the optimal value (i.e. 3.5K). Even if this result may be overly optimistic in practice, it goes to show that the scalability limitation of the master-slave model is a relative matter that needs to be put in perspective with its advantages. Moreover, if one needs to experiment with an island-model, it can always be simulated using a multidemic framework (e.g. Open BEAGLE; `http://www.gel.ulaval.ca/~beagle`) over a master-slave.

References

1. E. Cantú-Paz. *Efficient and accurate parallel genetic algorithms.* Kluwer Academic Publishers, Boston, MA, USA, 2000.