# A Hybrid Genetic Algorithm for Task Allocation in Multicomputers

*Nashaat Mansour*
*Geoffrey Fox*

**CRPC-TR91125**
**April 1991**

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

SCCS - 36
CRPC-TR91125

# "A Hybrid Genetic Algorithm for Task Allocation in Multicomputers"

by:

Nashaat Mansour
School of Computer and Information Science

Geoffrey C. Fox
Northeast Parallel Architectures Center
Syracuse Center for Computational Science
School of Computer and Information Science
Physics Department

April 1991

Will Appear in:

*International Conference on Genetic Algorithms and Applications*

July 1991

**Syracuse Center for Computational Science**
**Syracuse University**
**111 College Place**
**Syracuse, New York 13244-4100**
**<sccs@npac.syr.edu>**
**(315) 443-1723**

# A Hybrid Genetic Algorithm for Task Allocation in Multicomputers

Nashat Mansour
School of Computer Science
Center for Computational Science
Syracuse University
Syracuse New York 13244

Geoffrey C. Fox
School of Computer Science
Department of Physics
Center for Computational Science
Northeast Parallel Architectures Ctr.
Syracuse University

## Abstract

A hybrid genetic algorithm is proposed for the task allocation problem (HGATA) in parallel computing. It includes elitist ranking selection, variable rates for the genetic operators, the inversion operator and hill-climbing of individuals. Hill-climbing is done by a simple heuristic procedure tailored to the application. HGATA minimizes the possibility of premature convergence and finds good solutions in a reasonable time. It also makes use of problem-specific knowledge to evade some computational costs and to reinforce some favorable aspects of the genetic search. The experimental results on realistic test cases support the HGATA approach for task allocation.

## 1 INTRODUCTION

Parallel computers offer a high computational power which makes them useful for many problems in science, engineering and other areas. Generally, they are classified as Single Instruction Multiple Data and Multiple Instruction Multiple Data (MIMD). Distributed-memory MIMD computers will henceforth be called multicomputers. While offering a high raw computational power, parallel computers can suffer from low utilization and, hence, show poor performance if the load is not distributed as equally as possible among the processors. This crucial issue leads to the task allocation problem. In multicomputers, task allocation aims at the minimization of the total execution time of a problem by balancing the calculations among the processors and minimizing the interprocessor communication. Task allocation may be based on partitioning the operations in the algorithm or the data set. In this work, data partitioning is considered.

The task allocation problem is a computationally intractable combinatorial optimization problem. Several heuristic methods have been proposed, such as mincut-based heuristics, orthogonal recursive bisection, scattered decomposition, neural networks and simulated annealing [7, 9, 10, 11, 12, 13, 19]. The deterministic methods are strong methods with predictable and low execution time. However, they, naturally, either make restrictive assumptions or tend to be biased towards particular instances of the problem. The stochastic methods make no assumptions about the problem considered. However, their execution time is currently unpredictable and is still an open question. A parallel version of the classical genetic algorithm has been suggested in [21] for the specific case of allocating matrix rows to a hypercube for the Gaussian elimination problem. The work in [21] is difficult to generalize to other allocation problems and assumes a computational model that is different from the one dealt with in this paper. The theory of complex systems has been suggested as a framework within which concurrency issues such as task allocation can be studied [14]. It should be emphasized here that all the approaches mentioned above, as well as our approach, aim at producing good sub-optimal solutions, and not necessarily the optimal, in a reasonable time.

In this work, we propose a hybrid genetic algorithm (HGATA) for the task allocation problem. HGATA enhances the classical genetic algorithm (GA) with a number of features in order to alleviate the problem of premature convergence and to improve the search efficiency. These features include a combination of design choices for the selection scheme, the genetic operators and the rates of the operators. The incorporation of a problem specific hill-climbing procedure is also an essential feature and is responsible for the hybrid adjective.

This paper is organized as follows. Section 2 defines the task allocation problem and presents an objective function. Section 3 presents HGATA and explains the choices involved. The experimental results are reported and discussed in section 4. In section 5, conclusions are given.

## 2 THE TASK ALLOCATION PROBLEM

Task allocation consists of partitioning the problem into tasks, i.e. subproblems, and allocating these tasks to the processors of the multicomputer such that an objective function is minimized. An objective function associated with the total execution time required for solving a problem is given below. The computational model is explained first, then exact and approximate objective functions are presented and discussed. Some parameters which will be utilized by HGATA are also given.

The model of computation considered here is that of loosely synchronous parallel algorithms [14], where calculation and communication do not overlap. Processors run the same code (algorithm) and repeat a calculate-communicate cycle, where each processor performs calculations on its subproblem (task) and then communicates with other processors to

exchange necessary boundary information.

To formulate an objective function representing the cost of task allocation, both the problem domain and the multicomputer are represented by graphs. The vertices of the problem graph are the data elements and the edges refer to the calculation dependency. The vertices of the multicomputer graph are the processors and the edges are given by the interconnections. Task allocation becomes a mapping of subsets of the vertices of the problem graph to vertices in the multicomputer graph. Let W(p) and C(p) denote the amount of calculation and communication for processor p, respectively. W(p) is proportional to the number of data elements allocated to p. C(p) is a function of the amount of information communicated by p and the distance it travels. The total execution time, T, for a parallel program is determined by the processor with the greatest load of calculation and communication, that is

$$T = \max_p \{W(p) + C(p)\} \ldots\ldots\ldots (1)$$

Equation (1) represents the exact objective function to be minimized in task allocation and is the basis for evaluating the results of HGATA. However, the use of this minimax criterion is computationally expensive because the calculation of a new T caused by any change in the mapping of elements to processors requires the recalculation of the load of all processors.To avoid this complexity, a quadratic objective function has been proposed [9, 12] to approximate the cost of task allocation. It can be expressed as

$$r^2 \sum_p N^2(p) + v\left(\frac{tcomm}{tcalc}\right) \sum_{p,q} d(p,q) \ldots.(2)$$

where r is the amount of calculation per data element, N(p) is the number of elements allocated to processor p, (tcomm/tcalc) is a machine dependent communication to calculation time ratio, $v$ is a constant scaling factor expressing the relative importance of communication with respect to calculation, and d(p,q) is the Hamming distance between processors p and q. The main advantage of using this quadratic cost function is that it enjoys the locality property. Locality means that a change in the cost due to a change in the allocation of elements to processors is determined by the reallocated elements only. Since HGATA employs a hill-climbing scheme based on incremental reallocation of elements, the locality property becomes very important for keeping hill-climbing as fast as possible. Another important consideration in using the objective function in (2) is the choice of the weight $v$. In this work, values for $v$ are chosen in harmony with the behavior of HGATA for the purpose of generating better quality solutions. This is elaborated in the next section within the HGATA context.

Two parameters which can be derived from the objective function in (2) are utilized by HGATA. The first is the degree of clustering (DOC) of the data elements in a task allocation instance. The maximum DOC, *DOC(max)*, corresponds to an optimal allocation. The second parameter is an estimate of the value of the optimal objective function. This estimate involves the problem size, the multicomputer size, and the scaling factor $v$. It is henceforth referred to as *OBJ(opt)*. The derivation for both parameters is omitted here. However, we note that DOC and *OBJ(opt)* are employed by HGATA for evading some computational costs and reinforcing some aspects of the search.

## 3 HYBRID GENETIC ALGORITHM

Genetic algorithms represent powerful weak methods for solving optimization problems, such as task allocation. However, the implementation of an efficient GA often encounters the problem of premature convergence to local optima, otherwise a long time may be required for the GA search to reach an optimal or a good suboptimal solution. Methods for overcoming the two problems of premature convergence and inefficiency would be conflicting and a compromise is usually required. The incorporation of problem specific knowledge has been proposed to direct the blind GA search to the fruitful regions of the search space for improving the efficiency [15, 16]. The resulting schemes are referred to as hybrid schemes. To address the problem of premature convergence, a number of techniques have been suggested. Some selection schemes have been proposed for reducing the stochastic sampling errors [2, 15]. Other techniques have been incorporated into the reproduction scheme in order to control the level of competition among individuals and to maintain diversity. Examples of these are prescaling, ranking and the use of sharing functions or crowding factors [1, 5, 6, 15]. Reduced-surrogate crossover and two-point crossover operators have been suggested for enhancing exploration and improving the search [3]. Adaptive rates for crossover and mutation have been found useful [3, 4]. The variation in these rates is usually inversely proportional to the level of diversity in the population.

The advantages of the techniques mentioned above have been demonstrated by comparing the resulting performance with that of the classical GA [17]. Often, the performance verification is carried out for DeJong's testbed of functions [6] or for other specific applications, such as the traveling salesperson problem. In this work, a number of techniques dealing with selection and genetic operators have been combined for improving the quality of the solutions for the task allocation problem. Also, a simple problem specific hill-climbing procedure is added for improving the efficiency of the search. The techniques and the procedure comprise HGATA which is outlined in Figure 1. Four objectives guide the design of HGATA. These are the minimization of the likelihood of premature convergence, increasing the search efficiency, keeping computational costs low, and utilizing domain knowledge wherever possible for satisfying the first three objectives. In the remainder of this section, HGATA is explained. An illustration of the stages of the genetic search is given first as a prelude to the description of some design choices in the following subsections.

```
Read (problem graph and multicomputer graph);
Random Generation of initial population P(0) of size POP;
Evaluate fitness of individuals in P(0);
For (gen = 1 to maxgen) OR until convergence do
  Set (v, operator rates, flags);
  Rank individuals in P(gen-1), and
      allocate reproduction trials stored in MATES[];
  /* produce new generation P(gen) */
  For (i = 1 to POP step 2) do
    Randomly select 2 parents from MATES [];
    Apply genetic operators (2-pt xover,mutation,inversion);
    Hill-climbing by new individuals;
  endfor
  Evaluate fitness of individuals in P(gen);
  Retain the better of {fittest(gen) , fittest(gen-1)};
endfor
```

Figure 1: An Outline of HGATA.

### 3.1 THREE STAGES OF HGATA SEARCH

In the beginning of the search, the allocation of data elements to processors is almost random and, thus, the communication among processors would be heavy and very far from optimal regardless of the distribution of the number of elements. In the successive generations, clusters of elements are expected to be gradually grown and allocated to processors such that the interprocessor communication is constantly reduced, at least in the fitter individuals in the population. Then, at some point in the search, the balancing of the calculational load becomes more significant for increasing the fitness. Therefore, two stages of the search can be distinguished. The first stage is the clustering stage which lays down the foundation of the basic pattern of the interprocessor communication. The second stage will be referred to as the calculation-balancing stage. Obviously, the two successive stages overlap.

A third stage in the search can also be identified when the population is near convergence. In this advanced stage, the average DOC of the population approaches *DOC(max)* and the clusters of elements crystallize. If these clusters are broken, the fitness of the respective individual would drop significantly and its survival becomes less likely. At this point, crossover becomes less useful for introducing new building blocks, mutation of elements in the middle of the clusters is useless and a fruitful search is that which concentrates on the adjustment of the boundaries of the clusters in the processors. This stage will henceforth be referred to as the tuning stage. Boundary adjustment can be accomplished mainly by the hill-climbing of individuals, which is explained below, aided by the probabilistic mutation of the boundary elements. The main responsibility of crossover becomes the propagation and the inheritance of high-performance building blocks and the maintenance of the drive towards convergence for the sake of search efficiency. For hill-climbing and boundary mutation to take on their role in this stage, it is necessary to increase the relative weight of the calculation term in the fitness function. This is elaborated below with the description of hill-climbing. It is worth noting here that the tuning stage constitutes a relatively small number of generations in comparison with the first two stages.

### 3.2 CHROMOSOMAL REPRESENTATION

An instance of task allocation is encoded by a chromosome whose length is equal to the number of data elements (vertices) in the problem graph. The value of an allele is an integer representing the processor to which a data element is allocated. The element is, therefore, the index (locus) of the processor (gene) to which it is assigned. For example, if we have a graph of four data elements and two processors, the genotype (1,1,2,1) indicates that elements 1,2 and 4 are allocated to processor 1 and element 3 to processor 2.

### 3.3 FITNESS EVALUATION

The fitness of an individual is evaluated as the reciprocal of the objective function in expression (2). As pointed out in section 2, the choice of $v$ is of particular interest. Its value should be chosen in accordance with the properties of the HGATA search illustrated above. $v$ should be so large that the communication term in the fitness function acquires sufficient importance in the clustering stage. But, $v$ should not be too large, otherwise it will swamp the effect of the calculational term in the later stages. In other words, $v$ is chosen to favor the fitness of the individuals whose structure involves nearest-neighbor interprocessor communication in the clustering stage. In the later phases of the search, the value of $v$ should allow the emphasis to shift to the calculation term in the fitness. A value which satisfies these requirements can be determined from the ratio of the calculation and communication terms of *OBJ(opt)*, which is defined in section 2. In subsection 3.7, it will be argued that $v$ has to be decreased in the tuning stage.

### 3.4 REPRODUCTION SCHEME

The reproduction scheme adopted in HGATA is elitist ranking followed by random selection of mates from the list of reproduction trials, or copies, allocated to the ranked individuals. In ranking [1], the individuals are sorted by their fitness values and are allocated a number of copies according to a predetermined scale of equidistant values for the population, and not according to their relative fitness. In HGATA, the ranks assigned to the fittest and the least fit individuals are 1.2 and 0.8, respectively. Individuals with ranks bigger than 1 are first assigned single copies. Then, the fractional part of their ranks and the ranks of the lower half of individuals are treated as probabilities for assignment of copies. This scheme has been found to produce a percent involvement value of 92% to 98% in different generations. It offers a suitable way for controlling the selective pressure and, hence, the inversely related population diversity [23]. This results in the control of premature convergence, which is the main reason for using ranking-based reproduction in HGA-

TA. The control of premature convergence by ranking outweighs the loss due to ignoring knowledge about the relative fitness, especially that the expression used for the fitness in our application is only an approximation to the exact one anyway. Furthermore, ranking dispenses with prescaling which is usually necessary for fitness proportionate reproduction schemes. From efficiency point of view, ranking provides a computationally cheap method for controlling the population diversity in comparison with expensive methods needed with fitness proportionate selection, such as sharing functions or DeJong's crowding schemes [5, 6, 15].

Elitism in the reproduction scheme refers to the preservation of the fittest individual. In HGATA, the preceding fittest individual is passed unscathed to the new generation, but it is forced to compete with the new fittest and only the better of the two is retained. The purpose of elitism and its current implementation is the exploitation of good building blocks and ensuring that good candidate solutions are saved if the search is to be truncated at any point.To patch up a part of the loophole created by the use of the approximate objective function, the criterion for choosing between the current fittest and the preceding fittest individuals is changed in the tuning stage. The exact expression for fitness is used and has been found beneficial.

### 3.5 GENETIC OPERATORS

The Genetic operators employed in HGATA are crossover, mutation and inversion. The two-point ring-like crossover is used because it offers less positional bias than the one-point standard crossover without introducing any distributional bias [8]. Other more complex and presumably higher-performance crossover operators, such as shuffle crossover [8], are not used in this work in order to avoid excessive computations.

The standard mutation operator is employed throughout the search. In the tuning stage of the search, for the reason explained in subsection 3.1, mutation is restricted to elements at the boundaries of the clusters

Inversion is used in the standard biological way, where a contiguous section of the chromosome is inverted. In HGATA, the chromosome is considered as a ring. Inversion at a low rate helps in introducing new building blocks, into the population for an application such as task allocation.

### 3.6 OPERATOR RATES

It has become widely recognized that variable operator rates are useful for maintaining diversity in the population and, hence, for alleviating the premature convergence problem [3, 4]. Rates are varied in the direction that counteracts the drop in diversity. Several Measures have been suggested for the detection of diversity, such as lost alleles, entropy, percent involvement, and others [1, 4, 15, 16]. The evaluation of these measures invariably requires considerable computations. In HGATA, the cost of computing measures of diversity is not incurred. Instead, the degree of clustering of elements is used to guide the variation of the rates. This design decision is based upon the observation that diversity is reduced in the population as the DOC increases. The current implementation uses a simple stepwise change in the rates. The smallest and largest rates are associated with the DOC of the first generation and the *DOC (max)* estimate, respectively.

### 3.7 HILL CLIMBING

Knowledge about the application can direct the blind genetic search to more profitable regions in the adaptive space. In HGATA, individuals carry out a simple problem-specific hill-climbing procedure that can increase their fitness. The procedure is greedy and its inclusion improves the efficiency of the search significantly.

Hill-climbing for an individual is performed by considering only the boundary data elements allocated to the processors; one at a time. A boundary element e is an element that is allocated to a processor p1 and has at least one neighboring element (in the problem graph) allocated to a different processor p2. Such an element is transferred from p1 to p2 if and only if the transfer causes the objective function to drop or stay the same. It can be shown that the Change in Objective Function, COF, due to a transfer of element e is given by

$$2r^2[1+N(p2)-N(p1)]+2\nu R(CCD)$$

where N(x) is the number of elements allocated to processor x before the transfer, R is the (tcomm/tcalc) ratio, and CCD is the change in communication cost (sum of distances) for element e. From this expression, it can easily be seen that a transfer of an element can only take place from overloaded processors to underloaded processors. It should be emphasized here that the formulation of COF, which leads to a simple implementation of hill-climbing, is a direct result of the locality property of the approximate objective function; as mentioned in section 2.

In the tuning stage of the evolution, a procedure for removing isolated elements is invoked as a part of hill-climbing. This amounts to eliminating noise components, which manifest themselves as artificial additions to both the calculation and communication loads of processors in a task allocation instance.

Hill-climbing plays a distinctive role in the tuning stage, where it fine-tunes the structures by adjusting the boundaries of the sizeable clusters assigned to the processors. In this advanced stage, the basic pattern of interprocessor communication can not be significantly changed and the search ceases to offer significant gains. For these reasons, the emphasis upon balancing the calculational load should be artificially increased for the purpose of facilitating the boundary adjustment. This is achieved by decreasing the value of the weight $\nu$ in the objective function gradually

Figure 2: 551-Element Mesh1.



Figure 3: 301-Element Mesh2.

from the fixed value used throughout the search to a small suitable value determined by the COF expression. The smallest useful value for $\nu$ is that which makes COF negative or zero when the following conditions coexist. The first condition is that an overloaded processor has two elements more than the underloaded processor. The second condition is that the transfer of an element e does not increase the sum of communication distances of e by more than one.

# 4 EXPERIMENTAL RESULTS

The experiments describe the solutions that can be obtained by HGATA for realistic problems. They also illustrate the design choices and parameters of HGATA. The experimental set-up is presented first, then the results are given and discussed.

## 4.1 EXPERIMENTAL DESIGN

A genetic algorithm is considered to be a 6-tuple of variables GA = (REP, XOV, INV, OPRATE, POP, MRANK), where REP and XOV refer to the reproduction scheme and the crossover operator, respectively, INV indicates whether inversion is included, OPRATE indicates either variable or fixed rates for the genetic operators, POP is the population size, and MRANK is the maximum rank for the ranking-based reproduction scheme. Other parameters are assumed to be the same as in the classical GA. POP has been empirically determined by extrapolation from small test cases. It has been found that a population size approximately equal to the size of the problem graph is adequate for HGATA as long as the multicomputer graph is much smaller. Fixed rates for the genetic operators are 0.6 for crossover, 0.002 for mutation and 0.02 for inversion. Variable rates vary in a stepwise fashion as follows. Crossover rate increases from 0.5 to 1.0, mutation rate increases form 0.002 to 0.004, and inversion rate decreases from 0.03 to 0.0.

Several test cases have been used. For small and regular problems, HGATA has always found an optimal task allocation efficiently. These results are not be presented here. Instead, two irregular problems with realistic sizes are considered. These are shown in Figures 2 and 3 and are henceforth referred to as Mesh1 and Mesh2, respectively. Most of the results presented below are the averages of three runs. This small number of runs is satisfactory to illustrate HGATA's features, except for paragraph (iii) where 20 runs have been carried out.

In all experiments, a solution obtained at a certain point in the search refers to the fittest individual in the respective generation. The performance measures are the (exact) multicomputer's efficiency and the average fitness of the population. The efficiency is defined as the ratio of the sequential execution time to the product of T (equation 1) and the number of processors in the multicomputer, Both measures are plotted below with respect to the number of generations, which, in its turn, is used to assess the efficiency of the search. For clarity, the results are given as ratios, where efficiency is normalized with respect to the (exact) optimum, and fitness is normalized with respect to the (approximate) fitness of the optimal solution. It should be understood that the use of exact efficiency and approximate fitness for expressing the quality of the solutions will obviously exhibit a discrepancy in the results for the two measures.

## 4.2 RESULTS

The first experiment only refers to Mesh1. All the following experiments refer to allocating Mesh2 to an 8-processor hypercube.

(i) For Mesh1 and a 16-processor hypercube, HGATA1 = (ranking, 2-point, yes, var, 500, 1.2) yields the allocation configuration depicted in Figure 4. The efficiency of this allocation is 0.93 of the optimum, and its fitness is 0.998 of the optimal fitness. This solution is obtained after 280 generations. Each generation takes about 30 seconds on a SPARC workstation.

(ii) HGATA2 = (ranking, 2-point, yes, var, 300, 1.2) applied to Mesh2 for a 3-cube finds a solution shown in Figure 5. The efficiency and fitness are shown in Figure 6, where the relative average loads of calculation and communication are also depicted. After generation 118, the search converges to a solution with efficiency and fitness ratios 0.97 and 0.998, respectively. Each generation takes about 12 seconds.

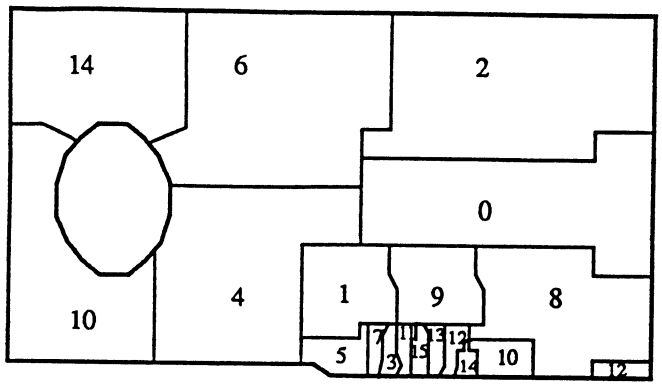
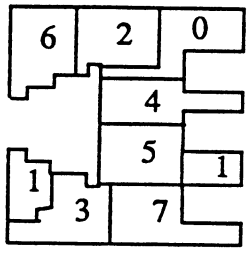



Figure 4: Allocation of Mesh1 to 4-Cube.




| Proc | Calc | Comm |
|---|---|---|
| 0 | 37 | 17 |
| 1 | 36 | 11 |
| 2 | 36 | 16 |
| 3 | 37 | 15 |
| 4 | 38 | 14 |
| 5 | 35 | 18 |
| 6 | 42 | 6 |
| 7 | 40 | 11 |

Figure 5: Allocation of Mesh2 to 3-Cube by HGATA2, and Processor Loads.

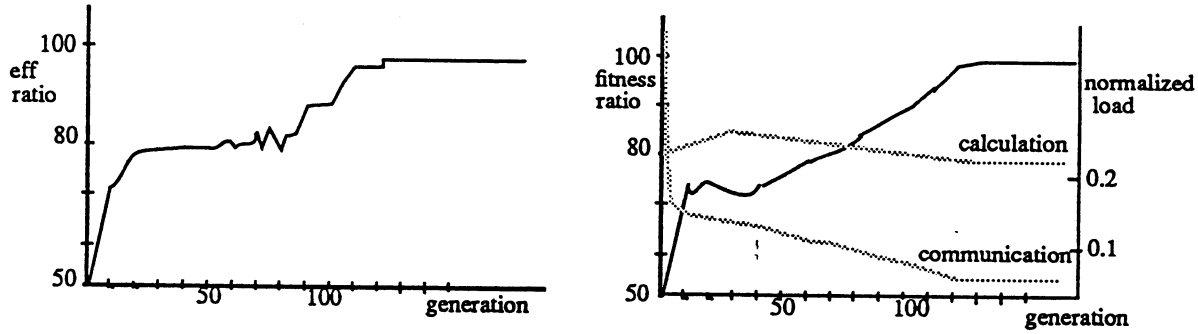



Figure 6: Efficiency and Fitness Ratios for HGATA2.

The three stages of the search can be identified in the fitness and load curves in Figure 6. Roughly, their overlapping points are generations 50 and 100. It can be seen that in the first stage, the communication load drops steadily regardless of the calculation load which happens to increase. In the second stage, both loads decrease and the fitness rises. Decreasing $\nu$ in the tuning stage enhances HGATA's tendency to reduce the calculation load. If $\nu$ had not been decreased at this advanced stage, the efficiency would have been trapped at 89%.

(iii) The averages of 20 runs are shown in Figure 7 for comparing HGATA2 with a classical GA1 = (RSIS, 1-point, no, fixed, 300, -). GA1, however, still includes hill climbing, for speed, and the problem-specific features in the tuning stage, for improving the final solution. RSIS is Remainder Stochastic Independent Sampling [2] implemented here with prescaling. Figure 7 shows that GA1 converges before generation 80 to a fitness of 0.99% and an efficiency of 89%. The efficiency is later improved to 92.5% in generation 130; under the effect of mutation and tuning. HGATA2 takes 45 more generations to converge to 0.99% fitness and 96% efficiency in generation 125. The best solutions found in the 20 runs are 94.2% and 97.2% efficiency by GA1 and HGATA2, respectively. The worst is 90.4% and 93.5% for GA1 and HGATA2, respectively. The mean square deviation of the efficiency results are 1.18 for HGATA2 and 1.1 for GA1. Clearly, GA1 (without expensive sharing functions or crowding factors) results in a higher selection pressure and lacks the capability of controlling convergence. This explains the lower quality solutions produced by the classical GA1 and highlights the advantages of the combination of choices adopted in HGATA.

(iv) The effect of increasing the selection pressure is explored by increasing MRANK in HGATA3 = (ranking, 2-point, yes, var, 300, 2.0). This results in an early convergence as shown in Figure 8. HGATA3 finds a good solution (96% efficiency ratio) in only 66 generations, which in 61% of the time required by HGATA2 to find a solution of the same quality. However, the large percentage of individuals (up to 20%) that die every generation, makes a maximum rank of 2.0 too high to be reliable in general for producing good solutions. This highlights the trade-off that exists between the solution quality and the search efficiency.

(v) Without hill-climbing, the efficiency of the search deteriorates tremendously. HGATA2, for example, becomes more than a hundred times slower.

(vi) The amount of improvement in the solution quality acquired in the tuning stage of the evolution has been found somewhat sensitive to the parameter that triggers this stage. If tuning is triggered too early, the time allowed for the first two stages of the evolution might be insufficient for producing near-optimal building blocks. If the tuning stage is invoked too late, convergence to a local optimum might have

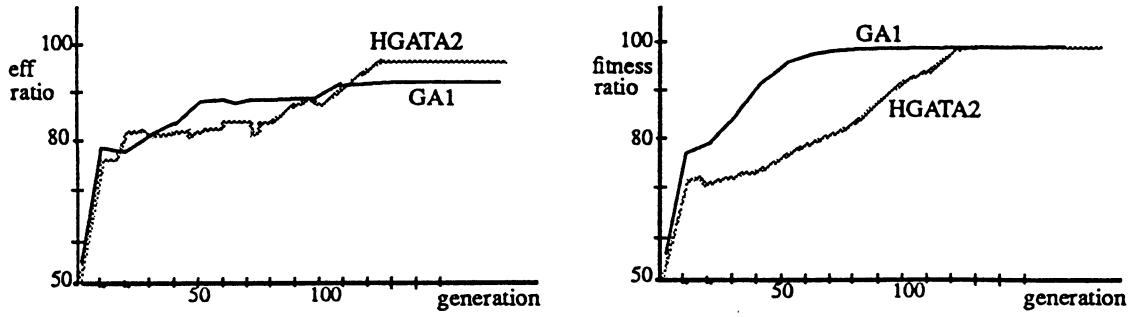



Figure 7: Comparison of HGATA2 and GA1.

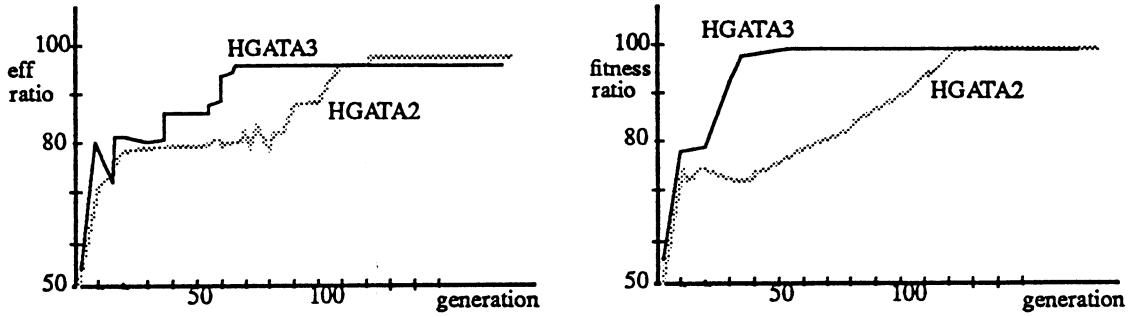



Figure 8: Comparison of HGATA2 and HGATA3.

already prevailed in the population as a result of the first two stages.

### 4.3 DISCUSSION

The results obtained for Mesh1 and Mesh2 are good suboptimal task allocations. They are considerably better than the results obtained by other faster task allocation techniques. For example, recursive bisection [11] produces a solution for Mesh2 whose efficiency is 87% of the optimum. Scattered decomposition [19] yields an efficiency 61% of the optimum. The best result of 500 runs of the hill-climbing algorithm, each starting with different initial random configuration, has been found to be 83%.

HGATA is not restricted to the loosely synchronous model of computation described in section 2. It can be easily adapted to other models by modifying the objective function module. Furthermore, most of the constituents of HGATA can be employed for solving related combinatorial optimization problems such as graph partitioning and quadratic assignment.

The trade-off between the solution quality and the computational cost is worthwhile emphasizing. The search can be made less expensive by resorting to measures such as, for example, increasing the selection pressure by some proportion as in HGATA3. But, in such cases the solution quality is likely to be sacrificed, although at a smaller proportion. The determination of a suitable population size is another important and difficult issue affecting the solution quality and amount of computations. The use of theoretically derived estimates makes the search time impractical. Since we are interested in suboptimal results, heuristic estimates would be adequate. In this work, a population size of the order of the size of the problem graph has been found satisfactory; when the multicomputer size is many times smaller.

The hill-climbing procedure enables qualified individuals to rapidly climb the adaptive peaks, which speeds up the evolution. This improvement in the efficiency of the search may seem to cause the exploitation feature to gain an upper hand over exploration; contributing to premature convergence. However, although hill-climbing does fuel the exploitation aspect of the search, the experimental results do not reveal any negative effects. Hill-climbing enables exploration to be carried out in the space of genotypes representing local fitness optima. Further, It seems that it plays a role similar to that of a knowledgeable mutation operator and does not lead the search to be trapped in local optima.

## 5 CONCLUSIONS

The combined constituents of HGATA have been shown to provide a good balance between exploratory forces and exploitation forces for the task allocation problem. HGATA greatly reduces the causes of premature convergence and has found near-optimal solutions in a reasonable time, although the objective function used is only an approximation to the exact one. The use of the degree of clustering of data elements has obviated expensive diversity detection mechanisms. Also, it has been found that setting the weighting factor $v$ in harmony with the properties of the search in dif-

ferent phases leads to better results.

The performance of HGATA can be improved in several ways. Firstly, the frequencies of the genetic operators can be adaptively varied according to a measure of the population diversity. Secondly, a more fruitful crossover operator, such as the reduced surrogate operator [3], can be used to enable the search to concentrate on useful work. However, it should be clear that additional costs will be incurred for both suggestions. Thirdly, a better heuristic estimate for the population size needs to be worked out for our specific application. Fourthly, the search efficiency is likely to increase and better solutions might be produced by starting the hill climbing procedure at a randomly chosen gene instead of the first gene in the chromosome. Fifthly, faster execution can be obtained by parallel algorithms based on HGATA [18, 20, 22]. The parallel algorithms can also reduce the sensitivity to design parameters.


### Acknowledgment

This work was supported by the Joint Tactical Fusion Program Office, and the National Science Foundation under Cooperative Agreement No. CCR-8809165.


### References


[1] J. E. Baker. Adaptive Selection Methods for Genetic Algorithms. *ICGA*'85, 101-111.

[2] J. E. Baker. Reducing Bias and Efficiency in the Selection Algorithm. *ICGA*'87, 14-21.

[3] L. Booker. Improving Search in Genetic Algorithms. In L. Davis, ed, *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, 1987, 61-73.

[4] L. Davis. Adapting Operator Probabilities on Genetic Algorithms. *ICGA*'89, 61-69.

[5] K. Deb, and D. E. Goldberg. An Investigation of Niche and Species Formation in Genetic Function Optimization. *ICGA*'89, 42-50.

[6] K. A. DeJong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. Doctoral Dissertation, University of Michigan, 1975.

[7] F. Ercal. *Heuristic Approaches to Task Allocation For Parallel Computing*. Doctoral Dissertation, Ohio State University, 1988.

[8] L. J. Eshelman, R. A. Caruana, and J. D. Schaffer. Biases in the Crossover Landscape. *ICGA*'89, 10-19.

[9] J. Flower, S. Otto, and M. Salama. Optimal Mapping of Irregular Finite Element Domains to Parallel Processors. *Caltech C3P* #292b, 1987.

[10] G. C. Fox. A Review of Automatic Load Balancing and Decomposition Methods for the Hypercube. In M. Shultz, ed., *Numerical Algorithms for Modern Parallel Computer Architectures*, Springer-Verlag, 1988, 63-76.

[11] G. C. Fox. A Graphical Approach to Load Balancing and Sparse Matrix Vector Multiplication on the Hypercube. *Caltech C3P* #327b, 1986.

[12] G. C. Fox, A. Kolawa, and R. Williams. The Implementation of a Dynamic Load Balancer. *Proc. 2nd Conf. Hypercube Multiprocessors*, 1987, 114-121.

[13] G. C. Fox and W. Furmanski. Load Balancing Loosely Synchronous Problems with a Neural Network. *Proc 3rd Conf. Hypercube Concurrent Computers, and Applications*, 1988, 241-278.

[14] G. C. Fox, M. Johnson, G. Lyzenga, S. Otto, J. Salmon, and D. Walker. *Solving Problems on Concurrent Processors*. Prentice Hall, 1988.

[15] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[16] J. J. Grefenstette. Incorporating Problem Specific Knowledge into Genetic Algorithms. In L. Davis, ed., *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, 1987, 42-60.

[17] J. H. Holland. *Adaptation in Natural and Artificial Systems*. Univ. of Michigan Press, 1975.

[18] N. Mansour and G.C. Fox. Parallel Genetic Algorithms with Application to Load Balancing. *In Preparation.*

[19] R. Morison and S. Otto. The Scattered Decomposition for Finite Elements. *Caltech C3P* #286, 1985.

[20] H. Muhlenbein. Parallel Genetic Algorithms, Population Genetics, and Combinatorial Optimization. *ICGA*'89, 416-421.

[21] C. C. Pettey and M.R. Leuze. Parallel Placement of Parallel Processes. *Proc. 3rd Conf.Hypercube Concurrent Computers, and Applications*, 1988, 232-238.

[22] R. Tanese. Distributed Genetic Algorithms. *ICGA*'89, 434-440.

[23] D. Whitley. The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best. *ICGA*'89, 116-123.