

Multiobjective Differential Evolution for Mapping in a Grid Environment

Ivanoe De Falco¹, Antonio Della Cioppa², Umberto Scafuri¹,
and Ernesto Tarantino¹

¹ ICAR–CNR, Via P. Castellino 111, 80131 Naples, Italy

{ivanoe.defalco,ernesto.tarantino,umberto.scafuri}@na.icar.cnr.it

² DIIE, University of Salerno, Via Ponte don Melillo 1, 84084 Fisciano (SA), Italy
adellacioppa@unisa.it

Abstract. Effective and efficient mapping algorithms for multisite parallel applications are fundamental to exploit the potentials of grid computing. Since the problem of optimally mapping is NP–complete, evolutionary techniques can help to find near–optimal solutions. Here a multiobjective Differential Evolution is investigated to face the mapping problem in a grid environment aiming at reducing the degree of use of the grid resources while, at the same time, maximizing Quality of Service requirements in terms of reliability. The proposed mapper is tested on different scenarios.

1 Introduction

Grid [1] is a decentralized heterogeneous multisite system which aggregates geographically dispersed and multi–owner resources (CPUs, storage system, network bandwidth,...) and can be profitably used to execute MPI computational intensive applications [2]. In fact, a grid represents a collaborative computational–intensive problem–solving environment, in which MPI applications, each made up of multiple message–passing subtasks, can be submitted without knowing where the resources are or even who owns these resources.

When the execution of an MPI grid application must satisfy user–dependent requirements [3], such as performance and Quality of Service (QoS) [4], single sites resources could be insufficient for meeting all these needs. Thus, a multisite mapping tool, able to obtain high throughput while matching applications needs with the networked grid computing resources, must be conceived.

In this work we deal with the mapping problem from the grid manager’s point of view, so our aim is to find the solution which uses at the minimum the grid resources it has to exploit at the most and respects the user QoS requests. Naturally, in absence of information about the communication timing, the job execution on grid is possible only if the co–scheduling of all its subtasks is guaranteed [5]. In this hypothesis, we distribute the application subtasks among the nodes minimizing execution time and communication delays, and at the same time optimizing resource utilization. QoS issues are investigated only in terms

of reliability. This means that our mapper maximizes reliability by preferring solutions which make use of devices, i.e. processors and links connecting the sites to internet, which only seldom are broken.

As the mapping is an NP-complete problem [6], several evolutionary-based techniques have been used to face it in a heterogeneous or grid environment [7,8,9,10,11]. To provide the user with a set of possible mapping solutions, each with different balance for use of resources and reliability, a multiobjective version of Differential Evolution (DE) [12] based on the Pareto method [13] is here proposed. Unlike the other existing evolutionary methods which simply search for one site onto which map the application, we deal with a multisite approach. Moreover, as a further distinctive issue with respect to other approaches proposed in literature [14], we consider the nodes making up the sites as the lowest computational unit taking into account its reliability and its actual load.

Paper structure is as follows: Section 2 illustrates our multiobjective mapper, Section 3 reports on the test problems experienced and shows the results achieved and Section 4 contains conclusions and future works.

2 DE for Mapping Problem

The technique. Given a minimization problem with q real parameters, DE faces it starting with a randomly initialized population consisting of \mathcal{M} individuals each made up by q real values. Then, the population is updated from a generation to the next one by means of different transformation schemes. We have chosen a strategy which is referenced as *DE/rand/1/bin*. In it for the generic i -th individual in the current population three integer numbers r_1 , r_2 and r_3 in $[1, \dots, \mathcal{M}]$ differing one another and different from i are randomly generated. Furthermore, another integer number s in the range $[1, q]$ is randomly chosen. Then, starting from the i -th individual a new trial one i' is generated whose generic j -th component is given by $x_{i'_j} = x_{r_{3j}} + F \cdot (x_{r_{1j}} - x_{r_{2j}})$, provided that either a randomly generated real number ρ in $[0.0, 1.0]$ is lower than the value of a parameter CR , in the same range as ρ , or the position j under account is exactly s . If neither is verified then a simple copy takes place: $x_{i'_j} = x_{i_j}$. F is a real and constant factor which controls the magnitude of the differential variation $(x_{r_{1j}} - x_{r_{2j}})$. This new trial individual i' is compared against the i -th individual in the current population and, if fitter, replaces it in the next population, otherwise the old one survives and is copied into the new population. This basic scheme is repeated for a maximum number of generations g .

Definitions. To focus the mapping problem in a grid we need information on the number and on the status of both accessible and demanded resources. We assume to have an application task subdivided into P subtasks (demanded resources) to be mapped on n nodes (accessible resources) with $n \in [1, \dots, N]$, where P is fixed *a priori* and N is the number of grid nodes. We need to know *a priori* the number of instructions α_i computed per time unit on each node i . Furthermore, we assume to have cognition of the communication bandwidth β_{ij} between any couple of nodes i and j . Note that β_{ij} is the generic element of an $N \times N$

symmetric matrix β with very high values on the main diagonal, i.e., β_{ii} is the bandwidth between two subtasks on the same node. This information is supposed to be contained in tables based either on statistical estimations in a particular time span or gathered tracking periodically and forecasting dynamically resource conditions [15,16]. In the Globus Toolkit [4], which is a standard grid middleware, the information is gathered by the Grid Index Information Service (GIIS) [16].

Since grids address non dedicated-resources, their own local workloads must be considered to evaluate the computation time. There exist several prediction methods to face the challenge of non-dedicated resources [17,18]. For example, we suppose to know the average load $\ell_i(\Delta t)$ of the node i at a given time span Δt with $\ell_i(\Delta t) \in [0.0, 1.0]$, where 0.0 means a node completely discharged and 1.0 a node locally loaded at 100%. Hence $(1 - \ell_i(\Delta t)) \cdot \alpha_i$ represents the power fraction of the node i available for the execution of grid subtasks.

As regards the resources requested by the application task, we assume to know for each subtask k the number of instructions γ_k and the number of communications ψ_{km} between the k -th and the m -th subtask $\forall m \neq k$ to be executed. Obviously, ψ_{km} is the generic element of a $P \times P$ symmetric matrix ψ with all null elements on the main diagonal. All this information can be obtained either by a static program analysis, or by using smart compilers or by other tools which automatically generate them. For example the Globus Toolkit includes an XML standard format to define application requirements [16].

Finally, information must be provided about the degree of reliability of any component of the grid. This is expressed in terms of fraction of actual operativity π_z for the processor z and λ_w for the link connecting to internet the site w to which z belongs. Any of these values ranges in $[0.0, 1.0]$.

Encoding. In general, any mapping solution should be represented by a vector μ of P integers ranging in the interval $[1, N]$. To obtain μ , the real values provided by DE in the interval $[1, N + 1[$ are truncated before evaluation. The truncated value $\lfloor \mu_i \rfloor$ denotes the node onto which the subtask i is mapped.

For all mapping problems in which also communications ψ_{km} must be taken into account, the allocation of a subtask on a given node can cause that the optimal mapping needs that also other subtasks must be allocated on the same node or in the same site, so as to decrease their communication times and thus their execution times, taking advantage of the higher communication bandwidths existing within any site compared to those between sites. Such a problem is a typical example of *epistasis*, i.e. a situation in which the value taken on by a variable influences those of other variables. This situation is also *deceptive*, since a solution μ_1 can be transformed into another μ_2 with better fitness only by passing through intermediate solutions, worse than both μ_1 and μ_2 , which would be discarded. To overcome this problem we have introduced a new operator, named *site mutation*, applied with a probability p_m any time a new individual must be produced. When this mutation is to be carried out, a position in μ is randomly chosen, let us suppose its contents refers to a node in site C_i . Then another site, say C_j , is randomly chosen. If this latter has N_j nodes, the position chosen and its next $N_j - 1$ are filled with values representing consecutive nodes

of C_j , starting from the first one of C_j . If the right end of the chromosome is reached, this process continues circularly. If $N_j > P$ this operator stops after modifying the P alleles. If *site mutation* does not take place, the classical transformations typical of DE must be applied.

Fitness. Given the two goals described in Section 1, we have two fitness functions, accounting one for the time of use of resources and the other for their reliability.

Use of resources. Denoting with τ_{ij}^{comp} and τ_{ij}^{comm} respectively the computation and the communication times requested to execute the subtask i on the node j it is assigned to, the generic element of the execution time matrix τ is computed as:

$$\tau_{ij} = \tau_{ij}^{\text{comp}} + \tau_{ij}^{\text{comm}}$$

In other words, τ_{ij} is the total time needed to execute the subtask i on the node j . It is evaluated on the basis of the computation power and of the bandwidth which remain available once deducted the local workload. Let τ_j^s be the summation on all the subtasks assigned to the j -th node for the current mapping. This value is the time spent by node j in executing computations and communications of all the subtasks assigned to it by the proposed solution. Clearly, τ_j^s is equal to zero for all the nodes not included in the vector μ .

Considering that all the subtasks are co-scheduled, the time required to complete the application execution is given by the maximum value among all the τ_j^s . Then, the fitness function is:

$$\Phi_1(\mu) = \max_{j \in [1, N]} \{\tau_j^s\} \tag{1}$$

The goal of the evolutionary algorithm is to search for the smallest fitness value among these maxima, i.e. to find the mapping which uses at the minimum, in terms of time, the grid resource it has to exploit at the most.

Reliability. In this case the fitness function is given by the reliability of the proposed solution. This is evaluated as:

$$\Phi_2(\mu) = \prod_{i=1}^P \pi_{\lfloor \mu_i \rfloor} \cdot \lambda_w \tag{2}$$

where $\lfloor \mu_i \rfloor$ is the node onto which the i -th subtask is mapped and w is the site this node belongs to.

It should be noted that the first fitness function should be minimized, while the second should be maximized. We face this two-objective problem by designing and implementing a multiobjective DE algorithm based on the Pareto-front approach. It is very similar to the DE scheme described in Sect. 2, apart from the way the new trial individual i' is compared to the current individual i . In this case i' is chosen if and only if it is not worse than i in terms of both the fitness functions, and is better than i for at least one of them. By doing so, a set

Algorithm 1

```

randomly initialize population
evaluate fitness  $\Phi_1$  and  $\Phi_2$  for any individual  $x$ 
while (maximal number of generations  $g$  is not reached) do
  begin
    for  $i = 1$  to  $\mathcal{M}$  do
      begin
        choose a random real number  $p_{sm} \in [0.0, 1.0]$ 
        if ( $p_{sm} < p_m$ ) apply site mutation
        else
          begin
            choose three integers  $r_1, r_2$  and  $r_3$  in  $[1, \mathcal{M}]$ , with  $r_1 \neq r_2 \neq r_3 \neq i$ 
            choose an integer number  $s$  in  $[1, q]$ 
            for  $j = 1$  to  $q$  do
              begin
                choose a random real number  $\rho \in [0.0, 1.0]$ 
                if ( ( $\rho < CR$ ) OR ( $j = s$ ) )  $x_{i'_j} = x_{r_{3j}} + F \cdot (x_{r_{1j}} - x_{r_{2j}})$ 
                else  $x_{i'_j} = x_{i_j}$ 
              end
            if ( ( $\Phi_1(x_{i'}) < \Phi_1(x_i)$ ) AND ( $\Phi_2(x_{i'}) \geq \Phi_2(x_i)$ ) ) OR
              ( ( $\Phi_1(x_{i'}) \leq \Phi_1(x_i)$ ) AND ( $\Phi_2(x_{i'}) > \Phi_2(x_i)$ ) )
              insert  $x_{i'}$  in the new population
            else insert  $x_i$  in the new population
          end
        end
      end
    end
  end
end

```

of “optimal” solutions, the so-called *Pareto optimal set*, emerges in that none of them can be considered to be better than any other in the same set with respect to all the single objective functions. The pseudocode of our DE for mapping is shown in the following **Algorithm 1**.

3 Experiments and Results

We assume to have a multisite grid architecture composed of $N = 116$ nodes divided into five sites (Fig. 1). Hereinafter the nodes are indicated by means of the external numbers, so that 37 in Fig. 1 is the fifth of 16 nodes in the site B . Without loss of generality we suppose that all the nodes belonging to the same site have the same power α expressed in terms of millions of instructions per second (MIPS). For example all the nodes belonging to B have $\alpha = 900$.

We have considered for each node three communication bands: the bandwidth β_{ii} available when subtasks are mapped on the same node (*intranode communication*), the bandwidth β_{ij} between the nodes i and j belonging to the same site (*intrasite communication*) and the bandwidth β_{ij} when the nodes i and j belong to different sites (*intersite communication*). For the sake of

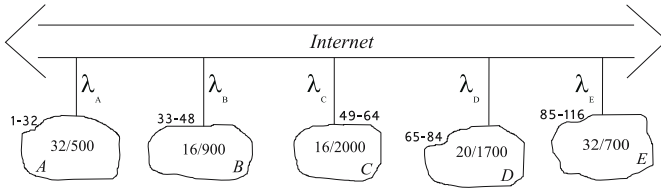


Fig. 1. The grid architecture

simplicity we presume that $\beta_{ij} = \beta_{ji}$ and that all the β_{ii} s have the same very high value (100 Gbit/s) so that the related communication time is negligible with respect to intrasite and intersite communications (Table 1).

Table 1. Intersite and intrasite bandwidths expressed in Mbit/s

	A	B	C	D	E
A	10				
B	2	100			
C	6	3	1000		
D	5	10	7	800	
E	2	5	6	1	100

Moreover we assume to know the local average load $\ell_i(\Delta t) \cdot \alpha_i$ of available grid nodes and, as concerns the reliability, we suppose that $\lambda_w = 0.99 \forall w \in \{A, B, \dots, E\}$, while in any site the nodes have different π values (Table 2).

Table 2. Reliability for the nodes

Sites	A	B	C	D	E					
nodes	1-12	13-32	33-40	41-48	49-58	59-64	65-72	73-84	85-100	101-116
π	0.99	0.96	0.97	0.99	0.97	0.99	0.99	0.97	0.98	0.96

Since a generally accepted set of heterogenous computing benchmarks does not exist, to evaluate the effectiveness of our DE approach we have conceived and explored four scenarios of increasing difficulty: one without communications among subtasks, one in which communications are added, another in which local node loads are also considered and a final in which reliability of a link is decreased. To test the behavior of the mapper also as a function of P , two cases have been dealt with: one in which $P = 20$ and another with $P = 35$. Given the grid architecture, $P = 20$ has been chosen to investigate the ability of the mapper when the number of subtasks is lower than the number of nodes

of some of the sites, while $P = 35$ has been considered to evaluate it when this number is greater than the number of nodes in any site.

After a preliminary tuning phase, DE parameters have been set as follows: $\mathcal{M} = 100$, $g = 1000$, $CR = 0.8$, $F = 0.5$, $p_m = 0.2$. All the tests have been made on a 1.5 GHz Pentium 4. For each problem 20 DE runs have been performed. Each execution takes about 1 minute for $P = 20$ and 2 minutes for $P = 35$. Note that these times are negligible since the mapping is related to computationally intensive grid applications which require several hours to be executed.

Henceforth we shall denote by μ_{Φ_1} and μ_{Φ_2} the best solutions found in terms of lowest maximal resource utilization time and of highest reliability, respectively.

Experiment 1. It has regarded an application of $P = 20$ subtasks with $\gamma_k = 90$ Giga Instructions (GI), $\psi_{km} = 0 \forall k, m \in [1, \dots, P]$, $\ell_i(\Delta t) = 0$ for all the nodes and $\lambda_w = 0.99 \forall w \in \{A, B, \dots, E\}$. Any execution of the DE mapper finds out several solutions, all being non-dominated in the final Pareto front. The solutions at the extremes of the achieved front are:

$$\mu_{\Phi_1} = \{62, 63, 64, 65, 66, 78, 68, 69, 81, 70, 71, 72, 74, 75, 76, 57, 58, 59, 60, 61\}$$

which has fitness values of $\Phi_1 = 52.94$, $\Phi_2 = 0.579$ and uses, as expected, the most powerful nodes of the sites C and D , and

$$\mu_{\Phi_2} = \{47, 43, 63, 63, 64, 59, 60, 61, 62, 59, 64, 61, 66, 62, 68, 69, 70, 71, 48, 46\}$$

which has $\Phi_1 = 100.00$, $\Phi_2 = 0.668$ and uses the most reliable nodes contained in the sites B , C and D . Furthermore, the system proposes some other non-dominated solutions better balanced in terms of the two goals, like for instance:

$$\mu = \{71, 59, 68, 61, 76, 60, 60, 70, 69, 61, 63, 65, 51, 64, 63, 75, 62, 72, 56, 67\}$$

with $\Phi_1 = 90.0$, $\Phi_2 = 0.616$. It is up to the user to choose, among the proposed solutions, the one which best suits his/her needs.

Experiment 2. In it all the parameters remain unchanged, but we have added a communication $\psi_{km} = 10$ Mbit $\forall k, m \in [1, \dots, P]$. In this case we have:

$$\mu_{\Phi_1} = \{72, 73, 74, 75, 66, 77, 78, 69, 80, 81, 82, 83, 84, 65, 76, 67, 68, 79, 70, 71\}$$

with $\Phi_1 = 53.17$ and $\Phi_2 = 0.523$. It is worth noting that the presence of communications yields that all the subtasks are mapped on the same site D , differently from the previous test case. In fact, C is the site with the fastest processors, but if the mapper used the 16 nodes of C only, then on some of them two subtasks should be placed, which would require a computation time on those nodes of $(90000 \cdot 2)/2000 = 90s$, higher than that needed to execute one subtask on a node of D , i.e. $(90000/1700) = 52.94s$. Furthermore, communications between C and D nodes are quite slower than those within the same site due to communication bandwidths. Moreover, we have:

$$\mu_{\Phi_2} = \{70, 71, 64, 65, 62, 67, 71, 69, 70, 66, 72, 68, 72, 65, 66, 67, 68, 69, 60, 61\}$$

with $\Phi_1 = 117.66, \Phi_2 = 0.668$. Also in this case many intermediate solutions are provided, which cannot be shown here for the sake of brevity.

Experiment 3. It is based on the same scenario as in the previous one, but we have taken into account the node loads as well, namely $\ell(\Delta t) = 0.9$ for all the nodes of the sites B and D , while for the site C we assume $\ell_i(\Delta t) = 0.8$ for $i \in [49, \dots, 52]$ and $\ell_i(\Delta t) = 0.6$ for $i \in [53, \dots, 64]$. The best mappings are:

$$\mu_{\Phi_1} = \{85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104\}$$

with $\Phi_1 = 130.47, \Phi_2 = 0.502$, and

$$\mu_{\Phi_2} = \{63, 1, 5, 4, 60, 5, 1, 7, 3, 12, 2, 6, 3, 6, 2, 8, 9, 10, 8, 11\}$$

with $\Phi_1 = 398.66, \Phi_2 = 0.668$. As concerns μ_{Φ_1} it is to remark that all subtasks have been placed on the nodes of E . This might seem strange, since nodes 53 – 64 in C , being loaded at 60%, are capable of providing an available computation power of 800 MIPS, so they are anyway more powerful than those in E . Nonetheless, since $P = 20$, those nodes would not be sufficient and four less powerful would be needed. The most powerful after those in C are those in E , and, in terms of Φ_1 , these two solutions are equivalent, since the slowest resources utilized are in both cases nodes in E , yielding a resource use due to computation of 128.57s. Moreover, the communications within E are faster than those between C and E , hence the μ_{Φ_1} proposed. As regards μ_{Φ_2} , instead, it correctly chooses among the most reliable nodes in the grid, and strongly uses the nodes 1 – 12 of A .

Experiment 4. It is as the third, but we have supposed that $\lambda_A = 0.97$. This should cause that, as far as reliability is concerned, solutions containing nodes of A should not be preferred, unlike the previous experiment. The best solutions found are:

$$\mu_{\Phi_1} = \{85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104\}$$

with $\Phi_1 = 130.47, \Phi_2 = 0.502$ and

$$\mu_{\Phi_2} = \{59, 60, 69, 60, 62, 64, 67, 59, 63, 61, 61, 63, 62, 65, 61, 71, 60, 66, 60, 72\}$$

with $\Phi_1 = 549.47, \Phi_2 = 0.668$. We can see that μ_{Φ_1} is the same as before, while μ_{Φ_2} has optimally shifted to the most reliable nodes in C and D . This confirms that our mapper correctly deals with low reliability issues.

Experiment 5. It has regarded an application of $P = 35$ subtasks with $\gamma_k = 90$ Giga Instructions (GI), $\psi_{km} = 0 \forall k, m \in [1, \dots, P]$, $\ell_i(\Delta t) = 0.0$ for all the nodes and $\lambda_w = 0.99 \forall w \in \{A, B, \dots, E\}$. The best mappings provided by our tool are:

$$\mu_{\Phi_1} = \{72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71\}$$

with $\Phi_1 = 52.94s$, $\Phi_2 = 0.32$ and uses the most powerful nodes in C and D , and

$$\mu_{\Phi_2} = \{66, 59, 60, 56, 59, 41, 61, 62, 69, 71, 65, 63, 62, 72, 67, 68, 53, 72, \\ 44, 61, 62, 66, 71, 47, 74, 66, 63, 65, 63, 68, 71, 70, 43, 61, 65\}$$

with $\Phi_1 = 158.82s$, $\Phi_2 = 0.465$ and picks the most reliable nodes among the sites B , C and D . Furthermore, the system proposes some other non-dominated solutions which are better balanced in terms of the two goals.

Experiment 6. It is like the fifth but we have added a communication $\psi_{km} = 10$ Mbit $\forall k, m \in [1, \dots, P]$. The best solutions are:

$$\mu_{\Phi_1} = \{81, 82, 83, 84, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, \\ 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80\}$$

with $\Phi_1 = 74.60s$, $\Phi_2 = 0.322$ and

$$\mu_{\Phi_2} = \{60, 66, 73, 61, 61, 66, 69, 59, 66, 65, 69, 72, 70, 48, 60, 76, 61, 42, \\ 71, 67, 59, 41, 67, 65, 70, 65, 65, 62, 70, 68, 64, 64, 60, 69, 54\}$$

with $\Phi_1 = 293.14s$, $\Phi_2 = 0.465$. The mapping μ_{Φ_1} uses only nodes in sites C and D , i.e., the most powerful ones. In fact, in this case the increase in execution time due to communications between sites is lower than that which would be obtained by using just one site and mapping two subtasks on a same node.

Experiment 7. It is like the sixth but we have taken into account the node loads as well, namely $\ell(\Delta t) = 0.9$ for all the nodes of the sites B and D , while for the site C we assume $\ell_i(\Delta t) = 0.8$ for $i \in [49, \dots, 52]$ and $\ell_i(\Delta t) = 0.6$ for $i \in [53, \dots, 64]$. This time we have:

$$\mu_{\Phi_1} = \{93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, \\ 110, 111, 112, 113, 114, 115, 116, 59, 60, 61, 85, 86, 87, 88, 89, 90, 91, 92\}$$

with $\Phi_1 = 165.85s$, $\Phi_2 = 0.257$ and

$$\mu_{\Phi_2} = \{1, 6, 63, 8, 9, 2, 5, 1, 2, 6, 7, 5, 6, 7, 8, 9, 10, 24, 12, 1, 2, \\ 3, 9, 57, 4, 5, 6, 7, 8, 9, 10, 11, 64, 65, 66\}$$

with $\Phi_1 = 860.01s$, $\Phi_2 = 0.47$. In this case μ_{Φ_1} maps 32 subtasks on the site E and 3 on the nodes 59, 60 and 61 of C . This might seem strange, since nodes 53 – 64 in C , being loaded at 60%, are capable of providing an available computation power of 800 MIPS, so they are anyway more powerful than those in E . Nonetheless, since $P = 35$, those nodes would not be sufficient and 23 less powerful would be needed. The most powerful after those in C are those in E , and, in terms of Φ_1 , these two solutions are equivalent since the slowest resources utilized are in both cases nodes in E , yielding a resource use due to computation of $(90000/700) = 128.57s$. Moreover, the intrasite communications of E are faster than the intersite between C and E , hence the μ_{Φ_1} proposed.

It is to note that a number of subtasks on E greater than 32 would imply that more than one subtask should be placed per node. This would heavily increase the computation time. On the contrary, a number of subtasks on C greater than 3 and lower or equal to 12 would leave the computation time unchanged but would increase the amount of intersite communication time.

As regards μ_{Φ_2} , instead, the suboptimal solution chooses all the nodes with $\pi = 0.99$ apart from node 24 with $\pi = 0.96$ and node 57 with $\pi = 0.97$.

Experiment 8. The scenario is the same as in the previous case, but we have supposed that the reliability $\lambda_A = 0.97$. The best solutions found are:

$$\mu_{\Phi_1} = \{108, 109, 110, 111, 112, 113, 114, 115, 116, 60, 61, 63, 85, 86, 87, \\ 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107\}$$

with $\Phi_1 = 165.85s$, $\Phi_2 = 0.257$ and

$$\mu_{\Phi_2} = \{65, 57, 58, 59, 79, 61, 68, 62, 65, 65, 66, 67, 63, 69, 70, 71, 72, 73, \\ 60, 62, 58, 48, 60, 61, 71, 63, 64, 41, 66, 38, 68, 69, 70, 71, 72\}$$

with $\Phi_1 = 1653.55s$, $\Phi_2 = 0.437$. Here μ_{Φ_1} uses heavily site E and some nodes from C . This solution is slightly different from that in the previous experiment, but has the same fitness value. The solution μ_{Φ_2} instead uses the most reliable nodes in sites B , C and D and, as it should be, avoids nodes from A which has $\lambda_A = 0.97$. Moreover, the five non optimal nodes (38, 57, 58, 73 and 79) in the solution are not in E because E , at parity of reliability, has inferior performance due to its minor intrasite bandwidths.

In Fig. 2 we report the final Pareto front of the alternative mapping solutions, equivalent from an evolutionary point of view, found out for one of the runs for this last test. In this case, the solutions range between those with a good reliability Φ_2 and with a high time of use of resources Φ_1 to those with a lower reliability but with a minor use time. As it can be easily perceived from the figure, the solutions which yield better balance in satisfying both goals are those in the intermediate region of the front. In fact such solutions are geometrically closer to the theoretically optimal solution for which $\Phi_1 \rightarrow 0$ and $\Phi_2 \rightarrow 1$.

Table 3 shows for each test (*Exp. no*) for both fitnesses the best final value Φ^b , the average of the final values over the 20 runs $\langle \Phi \rangle$ and the variance σ_Φ . The tests have evidenced a high degree of efficiency of the proposed model in terms of goodness for both resource use and reliability. In fact, efficient solutions have been provided independently of work conditions (heterogenous nodes diverse in terms of number, type and load) and kind of application tasks (computation or communication bound). Moreover, we have that often, especially as regards the problems with $P = 20$ subtasks, $\sigma(\Phi_1) = 0$ which means that in all the 20 runs the same final solution, i.e. the globally best one, has been found.

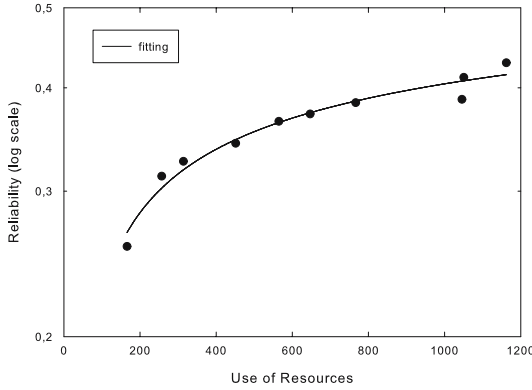


Fig. 2. The solutions on the Pareto front for experiment 8

Table 3. Findings for each experiment

Exp. no.	P=20				P=35			
	1	2	3	4	5	6	7	8
Φ_1^b	52.94	53.17	130.47	130.47	52.94	74.60	165.85	165.85
$\langle \Phi_1 \rangle$	52.94	53.17	130.47	130.47	93.61	94.93	224.33	165.85
σ_{Φ_1}	0	0	0	0	17.86	15.02	31.44	0
Φ_2^b	0.668	0.668	0.668	0.668	0.465	0.465	0.470	0.437
$\langle \Phi_2 \rangle$	0.668	0.668	0.660	0.652	0.443	0.448	0.436	0.419
σ_{Φ_2}	0	0	0.008	0.008	0.009	0.008	0.013	0.011

4 Conclusions and Future Works

This paper faces the grid multisite mapping problem by means of a multiobjective DE considering two goals: the minimization of the use degree of the grid resources and the maximization of the reliability of the proposed mapping. The results show that DE is a viable approach to the important problem of grid resource allocation.

Due to the lack of systems which operate in the same conditions as ours, at the moment, a comparison to ascertain the effectiveness of our multisite mapping approach is difficult. In fact some of these algorithms, such as Min–min, Max–min, XSuffrage [14], are related to independent subtasks and their performance are affected in heterogenous environments. In case of dependent tasks, the classical approaches apply the popular model of Direct Acyclic Graph differently from ours in which no assumptions are made about the communication timing among the processes since we have hypothesized the subtasks co–scheduling.

In addition to reliability, we intend to enrich our tool to manage multiple QoS requirements (performance, bandwidth, cost, response time and so on).

References

1. Buyya, R., Abramson, D., Giddy, J., Stockinger, H.: Economic models for resource management and scheduling in grid computing. *Journal of Concurrency and Computation: Practice and Experience* 14(13–15), 1507–1542 (2002)
2. Snir, M., Otto, S., Huss-Lederman, S., Walker, D., Dongarra, J.: *MPI: The Complete Reference – The MPI Core*, vol. 1. The MIT Press, Cambridge, MA (1998)
3. Khokhar, A., Prasanna, V.K., Shaaban, M., Wang, C.L.: Heterogeneous computing: Challenges and opportunities. *IEEE Computer* 26(6), 18–27 (1993)
4. Foster, I.: Globus toolkit version 4: Software for service-oriented systems. In: Jin, H., Reed, D., Jiang, W. (eds.) *NPC 2005*. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005)
5. Mateescu, G.: Quality of service on the grid via metascheduling with resource co-scheduling and co-reservation. *International Journal of High Performance Computing Applications* 17(3), 209–218 (2003)
6. Fernandez-Baca, D.: Allocating modules to processors in a distributed system. *IEEE Transaction on Software Engineering* 15(11), 1427–1436 (1989)
7. Wang, L., Siegel, J.S., Roychowdhury, V.P., Maciejewski, A.A.: Task matching and scheduling in heterogeneous computing environments using a genetic-algorithm-based approach. *Journal of Parallel and Distributed Computing* 47, 8–22 (1997)
8. Kwok, Y.K., Ahmad, I.: Efficient scheduling of arbitrary task graphs to multiprocessors using a parallel genetic algorithm. *Journal of Parallel and Distributed Computing* 47(1), 58–77 (1997)
9. Braun, T.D., Siegel, H.J., Beck, N., Bölöni, L.L., Maheswaran, M., Reuther, A.I., Robertson, J.P., Theys, M.D., Yao, B.: A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing* 61, 810–837 (2001)
10. Kim, S., Weissman, J.B.: A genetic algorithm based approach for scheduling decomposable data grid applications. In: *International Conference on Parallel Processing (ICPP 2004)*, Montreal, Quebec, Canada, pp. 406–413 (2004)
11. Song, S., Kwok, Y.K., Hwang, K.: Security-driven heuristics and a fast genetic algorithm for trusted grid job scheduling. In: *IPDP 2005*, Denver, Colorado (2005)
12. Price, K., Storn, R.: Differential evolution. *Dr. Dobbs's Journal* 22(4), 18–24 (1997)
13. Fonseca, C.M., Fleming, P.J.: An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation* 3(1), 1–16 (1995)
14. Dong, F., Akl, S.G.: Scheduling algorithms for grid computing: State of the art and open problems. Technical Report 2006–504, School of Computing, Queen's University Kingston, Ontario, Canada (2006)
15. Fitzgerald, S., Foster, I., Kesselman, C., von Laszewski, G., Smith, W., Tuecke, S.: A directory service for configuring high-performance distributed computations. In: *Sixth Symp. on High Performance Distributed Computing*, Portland, OR, USA, pp. 365–375. IEEE Computer Society, Los Alamitos (1997)
16. Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid information services for distributed resource sharing. In: *Tenth Symp. on High Performance Distributed Computing*, pp. 181–194. IEEE Computer Society, San Francisco, CA, USA (2001)
17. Wolski, R., Spring, N., Hayes, J.: The network weather service: a distributed resource performance forecasting service for metacomputing. *Future Generation Computer Systems* 15(5–6), 757–768 (1999)
18. Gong, L., Sun, X.H., Waston, E.: Performance modeling and prediction of non-dedicated network computing. *IEEE Trans. on Computer* 51(9), 1041–1055 (2002)