A parallel implementation of Genetic Algorithms for parameters estimation of Molecular Surfaces similarity analysis

Cozzi P[1,2], Merelli I[1], D'agostino D[3], Milanesi L[1]

Motivation

The problem of identifying protein functions is an open issue in bioinformatics. Generally, this is achieved by finding homologies with well-annotated proteins relying on sequence similarity or structure similarity, which can be even more conserved than sequences. In this context, we are developing a methodology that identifies similarities relying on protein surfaces, since they are highly correlated with functional sites. Our strategy is to describe molecular surfaces through images of local description in order to identify similarities by finding and clustering similar images. Finally surfaces are realigned to inspect and evaluate the identified similarity matches. In order to optimize the algorithm, which depends on 15 parameters, we have to estimate it in different conditions. For this reason, we chose to exploit Genetic Algorithms to identify a good parameter configuration so that we could successfully apply the surface similarity evaluation.

Methods

The Genetic Algorithm is a computational model of biological evolution used to find approximate solutions to optimization problems. By using this model, all candidate solutions are represented by individuals composed by different set of genes (chromosomes). Starting from a population of randomly generated individuals, the evolution of the population happens in generations: in each step of the algorithm a fitness function is evaluated for all the individuals and only the best individuals are selected for the following generations. Those individuals are recombined and mutated in order to generate a new population with higher fitness. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the current population. Each model implemented with a genetic algorithm requires a genetic representation of the solution domain and a fitness function to evaluate the solution domain. In our case, the genetic representation of the problem is obtained by considering our parameters as genes, and the fitness of each individual is evaluated by inspecting the goodness of the similarity match achieved by each parameter configuration. The repeated evaluation of the fitness function for parameter estimation is very time consuming, since each similarity measure can take from minutes to hours on a single CPU. For this reason, we have developed a

[1] ITB-CNR, Segrate (MI) [2] Università degli studi di Milano [3] IMATI-CNR, Genova

parallel implementation of a genetic algorithm in order to execute it on a computer cluster to improve the fitness function evaluation. In our implementation the main process handles the population by generating new individuals and selecting the best configuration of parameters. The fitness evaluation is managed by the master process which acts as server in the master node of the cluster and sends individuals to clients running on the other nodes through socket connections. In this way the fitness evaluation of each individual can be done in parallel by each client process. When one client finishes its calculation, it writes a result file and asks to the server for a new individual to evaluate. After each generation, the master process reads all the results files produced by the clients and prepares a new population relying on the fitness to be evaluated by the client processes in the next step of the algorithm.

Results

With the help of our parallel implementation of this Genetic Algorithm, we tried to superimpose similar surfaces and to evaluate the similarity level by using the image based approach. Molecular surfaces were calculated and represented by triangular meshes: each client process evaluates the similarity level by inspecting the percentage of vertices nearer than 1A° from two aligned surfaces. In our model we chose to evaluate each parameter in 10 steps, to initialize a population of 100 individuals and to perform 200 generations by evolving individuals with a two points crossing over. After about 50 generations on average, solutions converged to the specific fitness values, therefore we decide to terminate the simulation after two weeks. During this period we computed about 100 CPU days, by running each simulation on 20 client processes. Even the time needed for the calculation of the slowest individual affects the time needed to calculate the entire generation, we estimate a speedup of about 7 times, proving that our approach is scalable. Finally we evaluated the goodness of the parameter configuration identified by searching for surface similarity for 6 proteins randomly chosen towards a database of patches derived from the extraction of prosite domains from the whole molecular surfaces of the non redundant chain set of protein data bank. In each case we correctly identified similarities with prosite patches in all the proteins used for this test. We concluded that the parameter configuration adopted for these test cases also applies to other cases and has been adopted as the default parameter configuration of our algorithm.

Contact e-mail

paolo.cozzi@itb.cnr.it