Evaluating Genetic Algorithms in Protein-Ligand Docking

Rafael Ördög and Vince Grolmusz

Protein Information Technology Group, Eötvös University, Budapest and
Uratim Ltd., Nyíregyháza, Hungary
{devill, grolmusz}@cs.elte.hu

Abstract. *In silico* protein-ligand docking is a basic problem in pharmaceutics and bio-informatics research. One of the very few protein-ligand docking software with available source is the Autodock 3.05 of the Scripps Research Institute. Autodock 3.05 uses a Lamarckian genetic algorithm for global optimization with a Solis-Wets local search strategy. In this work we evaluate the convergence speed and the deviation properties of the solution produced by Autodock with diverse parameter settings. We conclude that the docking energies found by the genetic algorithm have uncomfortably large deviations. We also suggest a method for considerably decreasing the deviation while the number of evaluations will not be increased.

1 Introduction

In silico protein-ligand docking methods are becoming more and more important in searching for new drug candidate molecules because of their speed, economy and increasing reliability. Acquiring one compound (or ligand) for wet-laboratory testing from compound manufacturers costs around $ 100, consequently, without counting the costs of labor, the additional reagents and the protein production, in vitro verifying of the binding of one million compounds against one protein may cost $ 100 million. *In silico* simulation of the binding by docking methods costs only a small fraction of that amount, and can be completed in 1-2 weeks on a computer cluster of moderate size. The key ingredient of the *in silico* docking is the docking algorithm. Each docking algorithm optimizes some scoring function for finding the best location and conformation of the ligand molecule near to the surface of the protein. As an input, one must use the three-dimensional coordinates of the protein (usually taken from the Protein Data Bank) and the ligand molecule. As the output, a docking algorithm returns one or more docked conformation of the ligand and the protein, and the corresponding values of the scoring function.

1.1 The AutoDock Docking Software

One of the most widely known docking software with acquirable source code is the AutoDock 3.05 of the Scripps Research Institute [1]. Note, that the source

I. Măndoiu, R. Sunderraman, and A. Zelikovsky (Eds.): ISBRA 2008, LNBI 4983, pp. 402–413, 2008.

code of such popular docking software as Dock[2], Gold[3], Fred[4], FlexX[5] and many others are not available at all. In AutoDock 3.05 the scoring function is the estimated docking energy of the ligand to the protein. The best docking is the one with the smallest energy. The derivation of the empirical binding free energy function is described in [1], along with a brief historical review of the topic. It is easy to see that the three-dimensional position of any rigid molecule can be described by 6 real variables (three Euler angles plus the position of one of its atoms). If ℓ torsion axes are also allowed, then the scoring function should be optimized in a real space of dimension $6 + \ell$.

To speed up evaluation of the energy function AutoDock uses a grid-based approach. First, the protein is preprocessed by a program called AutoGrid: probe atoms and charges are placed at grid points around the protein, and the energy function is calculated and stored for latter use. After that AutoDock uses trilinear interpolation between grid-points to determine energy terms for each the atoms of the ligand separately, and then sums them up to calculate the energy of the conformation. The strategy is to minimize the energy function above in the $6 + \ell$ dimensional real space, where each point of the space corresponds to a conformation of the ligand. The optimization procedure clearly distinguishes a *local optimization* and a *global optimization* strategy. The aim of the local minimization strategies (local search) is to find a local minimum of the function in the neighborhood of the starting point. The aim of the global minimization strategy is to find the minimum of the function on the whole domain.

We need to mention that although there exist a good number of local search techniques finding local optima reliably for any continuous function, the *No free lunch theorem* [6] states that it is impossible to find a general purpose algorithm for global optimization, that performs equally well on all functions. Hence it is not trivial – if possible at all – to choose a global strategy, that suits a class of functions well. In order to circumvent this problem, one need to utilize as much information about the function as possible.

The first versions of AutoDock were using Simulated Annealing (SA) as global optimization strategy [1]. Further investigations [7] showed, that Genetic Algorithms with Local Search (GA-LS) – discussed in the next section – outperform the SA strategy. AutoDock Version 3 and later has both SA and GA-LS implemented. Neither for SA nor for GA-LS convergence results are known in the case of the energy function above.

1.2 Genetic Algorithms

Algorithms - performing function optimization - based on the principles of Darwinian Evolution, are called Evolutionary Algorithms. These algorithms maintain a collection of possible solutions (individuals) and select certain individuals for further processing depending on their fitness, i.e., the function value at the point represented by the individual. The most widely used of these algorithms are the Genetic Algorithms. In this section we sketch the basic properties of Genetic Algorithms. A more detailed description is given by Whitley [8].

An *Individual* is a point in the search space, and its *Genotype* is the string of numbers (or vector) that describes it. The *Phenotype* is the collection of attributes of the individual, and its *Fitness* is the function value corresponding to the individual. A *Population* is simply a collection of individuals. The algorithm first selects a population of (usually random) individuals that form the first *Generation*, then enters a cycle of deriving the n^{th} generation of individuals from the preceding. Every generation has the same fixed size. A cycle of the algorithm performs a *selection*, and applies *variation operators* to the individuals in the current population. We perform a *selection* by randomly choosing each individual of the

Genetic Algorithms with Local Search. (GA-LS) introduce a further unary variation operator, that is usually not considered a mutation operator, and are typically applied at the end of a cycle. This variation operator is a type of *Local Search*, typically Solis & Wets random local search.

The GA-LS implementation in AutoDock. To initialize the first generation, AutoDock chooses a population randomly. Population size is a parameter that is set to 50 as default. Then it enters the generation-deriving process, and starts the global strategy. Selection is either proportional, or a probabilistic binary tournament. We applied proportional choices for our experiments. Before the crossover step, individuals are being permuted, and then subsequent $2i^{th}$ and $(2i+1)^{th}$ individuals are being applied the crossover operator with a given probability, that is $\frac{4}{5}$ by default. AutoDock is using either one or two point block crossover (latter being our choice) where the blocks are real values for translation, rotation, and torsions. Cauchy mutation is being applied with default parameters mean 0, variance 1, and probability of applying the mutation operator is $\frac{1}{50}$. Elitism was set to keep the best individual alive. As a default every individual of a generation is applied a local search with the probability $\frac{2}{3}$, maximal number of steps is 300. All further parameters are left as default in AutoDock.

1.3 Previous Work

Thomsen's remarkable paper [9] analyzes genetic algorithm parameterization in the AutoDock 3.05 software. Six protein-ligand complexes were chosen for the experiments, and the effect of different population sizes, mutation operators, recombination operators, different local search strategies were tested for these six complexes. Based on the findings, in [9] a new algorithm, called DockEA was introduced and compared with the original AutoDock solutions. The results were quite different for different complexes: for some of the complexes it was not too difficult to find the near-optimum solutions, and for some others it was a more challenging task. Note also, that the resolution of the complexes examined were in the range 1.63-3.1 Å. The termination criterion was set to 250,000 evaluations and each experiments were repeated with 30 random seeds. We, in contrast, docked 48 randomly chosen ligands to the same protein: the

Mycobacterium tuberculosis dUTPase protein with PDB code of 2PY4, and each docking was repeated 100 times with different random seeds. The main point of our analysis was to find the number of evaluations which already gives reliable results, but it is as low as possible to facilitate screening large ligand libraries. Consequently, we perform our experiments with higher evaluations upper bound than 250,000; in some cases the upper bound was set to 2 million.

1.4 Concept of Our Analysis

Clearly, an "idealistic" randomized algorithm that performs automated docking need to satisfy some natural requirements. Our main motive was to optimize a database screening task with a single protein against two million ligand molecules, without changing the energy function of AutoDock. In what follows we list our requirements and give methods to quantitatively evaluate them.

Capability of finding optimum: This requirement means that with high enough probability the algorithm approximates the minimum of the function with a small enough error. Evaluating this requirement is difficult, since we do not have reliable measurement data on the values of the energy-minima for multiple ligands in question. To circumvent these difficulties, we compared different variants of the GA algorithm based on a fixed number of runs each with the same energy function.

Mean approximates best run - (Low deviation). It is reasonable to expect that a result, yielded by a given run of the algorithm is generally close to the optimum.

Consistency of the above quantities means that if the number of evaluations is increased, then the discovered optima get ever closer to the real optimum, and the deviation drops as well.

Order of hits. When thousands or millions of different ligands are docked against the same protein, identifying the order of the ligands, sorted by the docking energies, is sometimes much more important than the absolute values of the docking energies themselves.

To see the progress made by different algorithms we created run logs similar to the ones in Hart's thesis [7]. After every evaluation we checked if a new optimum was discovered, and recorded the number of evaluations and the new optima. On plots like the one on Figure 1 we can see number of evaluations on the horizontal axes, and the approximation of minimal energy in kcal/mol calculated after the given number of evaluations by different runs of the algorithm. Note that the most time-consuming step is the evaluation of the energy function, so the number of evaluations is proportional to the running time of the algorithm.

2 Methods

Our test-runs were performed on a cluster of 48 PC's equipped with 2.40GHz Intel Celeron^{TM} processors and 256 MB RAM. We have randomly chosen 48 molecules from the ZINC database [10], and docked them to the same *Mycobacterium tuberculosis* dUTPase molecule, with PDB code 2PY4, as a protein target. The algorithm ran with 100 different random seeds for every ligand to determine

deviation and seed dependency. The diagrams derived from the run logs by the procedure described at the end of the previous section were similar to Figure 1, and in what follows we will describe the common properties we have found:

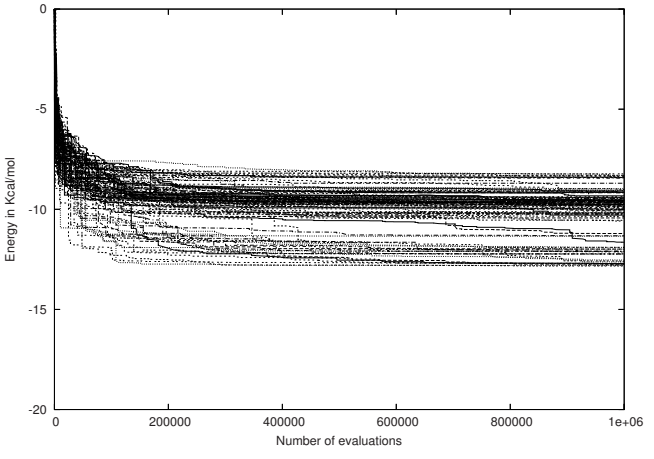

Fig. 1. Evolution of discovered minima over 100 runs with different seeds for ligand with ZINC code ZINC01208228

- One should note that near to the beginning the algorithm comes across unreasonably high positive energies due to collisions, but in several thousand steps it reaches negative values.
- Shortly after reaching negative energies, the best of the 100 runs finds values close to the minimum. For example, on Figure 1, the minimal run changes less than 0.5 kcal/mol after 250,000 steps. For the 71% of the 48 ligands we were testing this changed less than 0.5 kcal/mol after 250,000 evaluations, and 88% changed less than 1 kcal/mol. The worst case was 2 kcal/mol.
- As a generalization of the above observation one can see on Figure 1, that even after 10,000,000 evaluations, most of the runs stay close to the value reached after 250,000 evaluations. Note, however, that there are seeds producing large jumps at random positions (e.g., on Figure 1, just after 400,000 evaluations, where one descends from -10 kcal/mol to -12 kcal/mol.)

  Figure 3 shows 4,800 different runs (100 for each ligands), ordered by the amount of decrease of the docking energy found in kcal/mol from evaluation 250,000 to evaluation 10,000,000. The 50% of all runs to change less than 0.5 kcal/mol, and only 25% of the runs decreased more than 1 kcal/mol in this interval.
- Motivated from our analysis above, we define the "high confidence" interval of energies by the following properties:
  - Every discovered minimum is located in this interval independently of seed.
  - It is the smallest interval that satisfies the above.

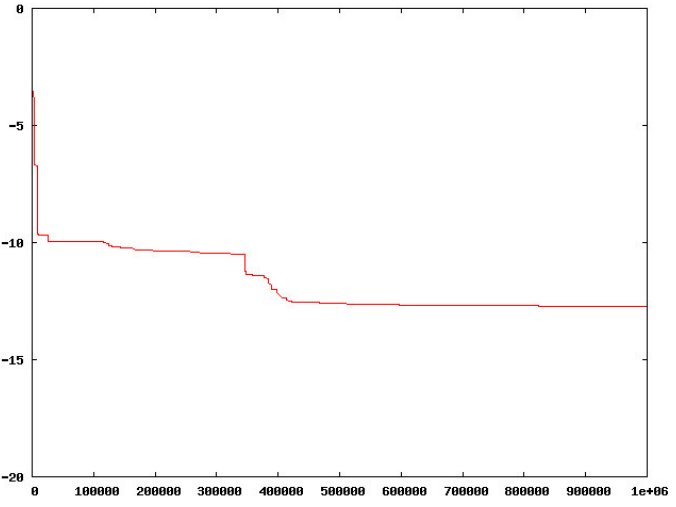

Fig. 2. Evolution of discovered minima with a seed for ligand with ZINC code ZINC01208228. Notice the sudden fall just before 350 000 evaluations.

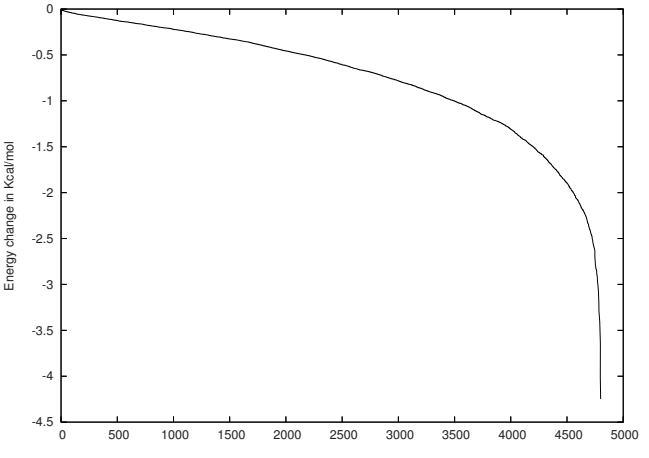

Fig. 3. 4,800 different runs ordered by amount of decreasing in kcal/mol from the 250,000th evaluation to the 10,000,000th evaluation

- It is worth mentioning that high confidence intervals can be approximated by considering only a small number of runs, as it can be seen on Figure 1. Figure 5 represents the "high confidence" intervals for different ligands with vertical bars, and the mean of 100 runs with a dot on the corresponding bar.
- Based on our 100 run approximations:
  - Usually the minima gained in 100 runs cover the high confidence interval almost uniformly. Figure 1 is a counterexample of this phenomenon, since

it has a gap between -10 and -12 kcal/mol. From the 48 ligands another 3 ligand also have similar gaps.
  - After 250,000 evaluations the endpoints of the high confidence intervals will not be changed much. More exactly, for the 71% of the 48 ligands we were testing the interval changed less than 0.5 kcal/mol after 250,000 evaluations, and for 88% it changed less than 1 kcal/mol. The worst case was 4 kcal/mol, but the second worst was only 1.5 kcal/mol.
  - The approximate size of the high confidence intervals is between 2 and 5 kcal/mol. (For example the energies for the ligand with ZINC code ZINC02958278 were between -11 and -7 kcal/mol.)
- Neither the standard deviation, nor the mean of 100 runs show consistent decrease after 250,000 evaluations. (See Figure 4.)

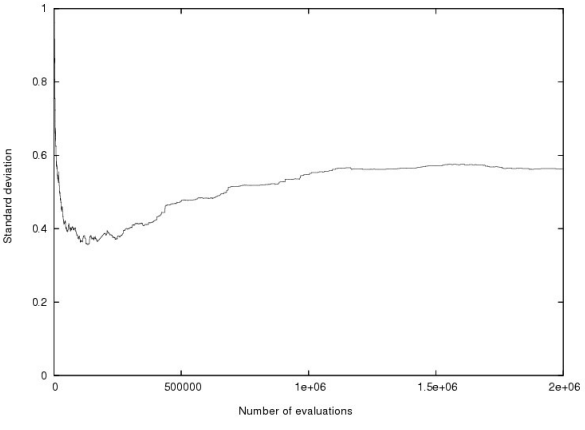

Fig. 4. Increasing deviation of different runs on the ligand with ZINC code ZINC01106466

One of the main conclusions of ours is that if ligands are characterized by the minimal energies discovered over 100 different runs, the difference between the best and worst ligands was less than 6.5 kcal/mol for this 48 random choices, i.e., we have got values between -13.5 and -7. Note that the length of that interval is clearly comparable with the lengths of high confidence intervals.

On Figure 5 the dots on the intervals indicate the average of runs. One can see that they are usually in the middle of the confidence interval, hence we can not expect a single randomly chosen run to be at least near to the optimum. What's more disturbing, if we accept the rather optimistic assumption of having chosen the "average run" for all the ligands by luck, we are still far from a relevant order, as — due to the large variance in the size of intervals — there seems to be no relationship between the minimal and average runs.

Another surprising result was, that deviation sometimes was increased after a number of evaluations instead of the anticipated decreasing. (See figure 4.) This pathological behavior arises when there is an easy to discover optimum,

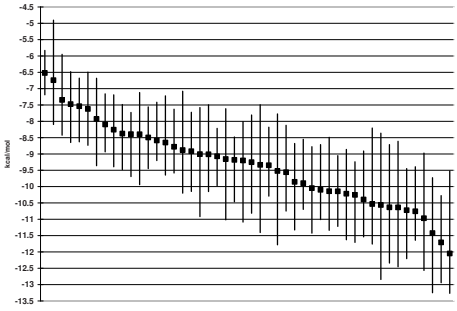

Fig. 5. High confidence intervals for 48 ligand molecules after 2,000,000 evaluations

and most of the runs find it within the first few thousand steps, and afterward they start to explore better solutions at random positions in time.

3 Results and Discussion

3.1 Multi-run Tests

In the case of high deviation it is a quite common approach to take the average, or the minimum of multiple samples. As we pointed out in the previous section averages are not relevant, consequently we tried multi-run algorithms with common minima, i.e., an n-run strategy with k evaluations would run the algorithm k times through $\frac{k}{n}$ evaluations, and take the minimum of the results. Our test described below showed that optimal n for a given k is highly protein-ligand dependent, though we believe that for a group of similar ligands and fixed protein it might be possible to find a common n. If ligand classes would be large enough it might be possible to save time by preprocessing ligand classes for every protein in order to identify the optimal n. The question whether is this possible remains open.

Our tests were performed as follows: we ran the algorithm 100 times, and divided it into 10 equal-sized group. For the average n run strategy the minimum of the first n out of every group was taken, and after that the mean of them to get an average n run strategy.

It is obvious, that for small number of evaluations any multi-run strategy is no match for the one-run strategy, and for significantly large number of evaluations in multi-run strategies will overcome the one-run strategy. (The latter is obvious considering the results of the previous Section, i.e., most runs get stuck at a point.) The question remains when does an n run strategy overcome an k run strategy, and more importantly which is the best for a given number of evaluations.

On Figure 6 the average of the n-run strategies plotted for ZINC entry ZINC00342090. In this example, the 3-run strategy is the first to overcome the one-run strategy at approximately 700,000 evaluations, but there is not enough

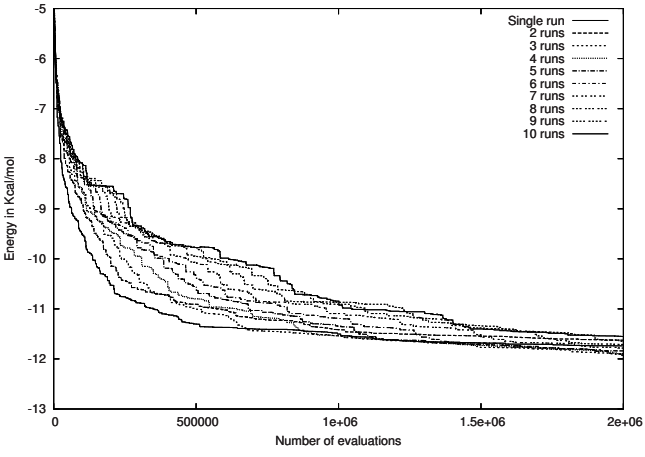

Fig. 6. Different multi run strategies for ZINC entry ZINC00342090

difference between them to ensure that the 3 run strategy is definitely better. Other ligand showed examples where the 1 run strategy turned out to be far better than any other strategy even at 2,000,000 evaluations, and there were examples, where the 2 and 3 run strategies overcame the single run version pretty well.

3.2 Modified GA-LS Algorithms

The choice of the initial population is an important phase of Genetic Algorithms. Its optimization seems to be neglected by the literature. This is not surprising in a certain sense, as the algorithm usually leaves those individuals behind at a quite early stage, hence a possible bad initial choice seems to have only a negligible effect on later generations. Our investigations show that this intuition is far from being correct. In the case of a function with such high complexity as our energy function, the choice of initial population can have strong positive effect on the speed of convergence to the actual optimum, by more or less restricting the search to interesting areas.

On figure 7 one can observe the average of 100 runs with different initialization strategies, for ZINC entry ZINC01208228. Again horizontal axes is number of evaluations, while vertical axes is energy in kcal/mol. Different strategies are described below:

Rigid start. In the ZINC database [10], ligands are stored in a conformation with minimal internal energy. Hence it is a natural idea to fix the torsions when choosing the individuals randomly. Our aim was to reduce the first "many collisions" phase of the algorithm, but the diversity of the population - as we expected - turned out to be too small, and that slowed down the algorithm significantly.

First population selected from a larger random population is another natural idea. To understand the behavior of random individuals we have created plots

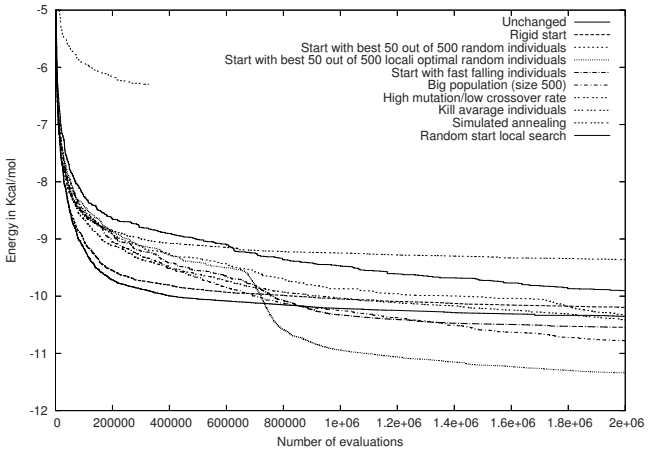

Fig. 7. Comparing the modified algorithms (x-axis) by average run and confidence interval for ZINC entry ZINC00342090 in kcal/mol (y-axis)

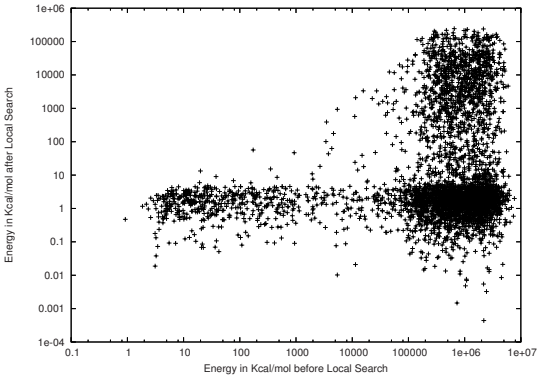

Fig. 8. Energies of random individuals, and of the nearest local optima found by local search

like the one on Figure 8 that show the energies (horizontal axes) for 5,000 random individuals, and the energies for the nearest local optimum found by local search (vertical axes). One can see that points form 3 classes, formed by low energy individuals, high energy individuals near low local optimum, and high energy individuals near high local optimum.

Selecting the best 50 individuals out of 200,000 turns out to perform more or less the same way as the unchanged algorithm. Depending on the ligand it can perform better or worse just as many times.

Selecting the best after local search with 20,000 being the number of local searched random individuals, and 50 the population size. As one can check on the example Figure 7, this approach turns out to be the best from the ones inspected by us. First it works its way through the local search phase, and then

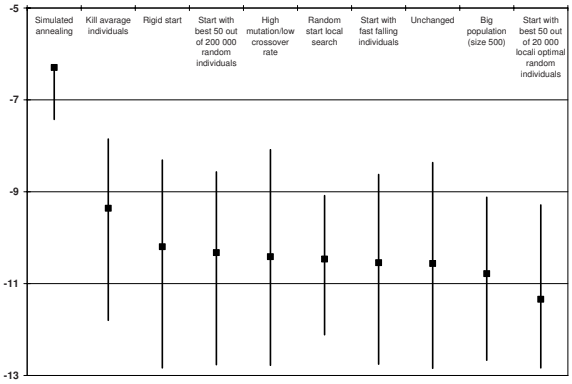

Fig. 9. Comparing the modified algorithms by high confidence interval after $2,000,000$ evaluations for ZINC entry ZINC00342090

soon after starting the genetic algorithm the average of the runs makes a sudden fall. At this point it outperforms all other approaches for most of the ligands.

Individuals changed by local search most dramatically seemed to be interesting, partly because they form a separate cluster (Figure 8), and partly because according to our notion optimal bindings are situated on the surface of the protein, hence changing it a little causes collisions, i.e., high energy conformations.

Non-initialization based modifications included higher population size of 500, modified mutation $\left(\frac{4}{5}\right)$ and crossover $\left(\frac{1}{5}\right)$ rates, and a modified version of proportional selection. The last modification aimed to exploit the remark mentioned earlier about optima being near to high energy conformations.

4 Conclusions

We performed an in-depth analysis of the settings of the GA-LS algorithm implemented in AutoDock 3.05, and concluded, that they tend to have high deviations when applied to energy functions of docking problems. The results were obtained using one single protein as a docking target; this fact yields consistency in the results, but may also bound the the generality of our results.

Consequently, one can not expect to reach exact energy bounds with this technique, nor to find a relevant order of ligands according to their bonding energies within a low number of evaluations. Multi-run strategies can help, but they are highly dependent on the protein-ligand pair in question.

Initialization can have a major effect on the speed of convergence. The best algorithm examined in this article was choosing the best 50 out of $20,000$ local searched individuals. The first phase of this algorithm is actually a random start local search, but after the genetic algorithm is started the average of runs makes a sudden fall, and outperforms all other variants we have examined. Figure 9 shows that this variant performs better than others in average.

Acknowledgment. The authors acknowledge the partial support of an OTKA grant NK 67867 and NKTH project "TB_INTER", and invaluable help from Zoltan Szabadka.

References

[1] Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. Journal of Computational Chemistry 19(14), 1639–1662 (1998)

[2] Ewing, T., Makino, S., Skillman, A., Kuntz, I.: Dock 4.0: Search strategies for automated molecular docking of flexible molecule databases. J. Comput. Aided. Mol. Des. 15(5), 411–428 (2001)

[3] Verdonk, M., Cole, J., Hartshorn, M., Murray, C., Taylor, R.: Improved protein-ligand docking using gold. Proteins 52(4), 609–623 (2003)

[4] Schulz-Gasch, T., Stahl, M.: Binding site characteristics in structure-based virtual screening: evaluation of current docking tools. Journal of Molecular Modeling 9(1), 47–57 (2003)

[5] Rarey, M., Kramer, B., Lengauer, T., Klebe, G.: A fast flexible docking method using an incremental construction algorithm. J. Mol. Biol. 261(3), 470–489 (1996)

[6] Wolpert, D.H., Macready, W.G.: No free lunch theorems for imization. IEEE Transactions on Evolutionary Computation (1997)

[7] Hart, W.E.: Adaptive Global imization with Local Search. PhD thesis, University of California, San Diego (1994)

[8] Whitley, D.: A genetic algorithm tutorial. Statistics and Computing 4, 65–85 (1994)

[9] Thomsen, R.: Flexible ligand docking using evolutionary algorithms. Investigating the effects of variation operators and local-search hybrids. BioSystems 72(1–2), 57–73 (2003)

[10] Irwin, J.J., Shoichet, B.K.: A free database of commercially available compounds for virtual screening. J. Chem. Inf. Comput. Sci. 45(1), 177–182 (2005)