Tensor Train Global Optimization: Application to Docking in the Configuration Space with a Large Number of Dimensions

A.V. Sulimov[1,2], D.A. Zheltkov[3], I.V. Oferkin[1], D.C. Kutov[1,2], E.V. Katkova[1,2], E.E. Tyrtyshnikov[2,3,4,5], ✉ V.B. Sulimov[1,2]

[1] Dimonta, Ltd, Moscow 117186, Russia
{as,io,dk,katkova}@dimonta.com,vladimir.sulimov@gmail.com
[2] Research Computer Center, Lomonosov Moscow State University, Moscow 119992, Russia
eugene.tyrtyshnikov@gmail.com
[3] Institute of Numerical Mathematics of Russian Academy of Sciences, Moscow, 119333, Russia
dmitry.zheltkov@gmail.com
[4] Faculty of Computational Mathematics and Cybernetics of Lomonosov Moscow State University, Moscow 119992, Russia
[5] Siedlce University of Natural Sciences and Humanities, woj. Mazowieckie, Poland

Abstract. The novel docking algorithm is presented and it is applied to the docking problem with flexible ligand and moveable protein atoms. The energy of the protein-ligand complex is calculated in the frame of the MMFF94 force field in vacuum. The conformation space of the system coordinates is formed by translations and rotations of the ligand as a whole, by the ligand torsions and also by Cartesian coordinates of the selected target protein atoms. The algorithm is realized in the novel parallel docking SOL-P program and results of its performance for a set of 30 protein-ligand complexes are presented. It is shown that mobility of the protein atoms improves docking positioning accuracy. The SOL-P program is able to perform docking of a flexible ligand into the active site of the target protein with several dozen of protein moveable atoms – up to 157 degrees of freedom.

Keywords: Docking · Tensor train · Protein-ligand complex · Protein moveable atoms · Flexible ligand · Drug design.

1 Introduction

Search of molecules-inhibitors of a given target protein is the key stage of the new drug development. Inhibitors block the active site of the protein associated with a disease and the disease is cured. Molecular modeling on the base of supercomputer simulation by docking and molecular dynamics programs should increase effectiveness of new inhibitors development [1, 2]. On the base of such calculations it is possible to predict inhibition activity of new compounds. The reliable prediction is defined by the accuracy of these programs. Docking programs perform positioning of a compound (a ligand) in the active site of the target protein. Computed poses of the ligand are used for the calculation of the protein-ligand binding free energy which is directly connected with the inhibition constant. Compounds with higher binding energy are better inhibitors because the same inhibition effect can be reached with smaller concentration of the compound. The accuracy of binding energy calculations should be better than 1 kcal/mol [3] for the reliable prediction of the inhibitory activity. However, the accuracy of binding energy calculations for arbitrary target proteins and ligands is too bad now. This accuracy depends on many factors and simplifications: the force field choice for modeling intra- and inter-molecular interactions instead of the use of quantum chemical methods, the solvent model, target protein and ligand models, the docking algorithm, the free energy calculation method, respective approximations and computer resources required for docking of one ligand. Main simplifications of many existing docking programs, e.g. the SOL [4] program, is the rigid protein approximation and the use of the grid of preliminary calculated potentials of ligand probe atoms interactions with the protein (the grid approximation) which restrict strongly performance of docking programs and make worse the docking accuracy. However proteins are flexible and some protein atoms near the ligand binding pose relax from their initial positions in the process of protein-ligand binding – a difference between bound and unbound protein's structures is often observed [5]. In this study we describe the novel docking algorithm which makes it possible to reject the rigid protein as well as the grid approximations, to take into account many proteins' degrees of freedom and to increase the docking accuracy.

The protein-ligand binding free energy ΔG_{bind} can be calculated as the difference between the free energy of the protein-ligand complex G_{PL} and the sum of free energies of the unbound protein G_P and the unbound ligand G_L:

$$\Delta G_{bind} = G_{PL} - G_P - G_L \quad (1)$$

Free energies of the protein, the ligand and their complex are described by respective energy landscapes and they can be calculated through the configuration integrals over the respective phase space. In the thermodynamic equilibrium the molecular system occupies its low energy minima. The configuration integral will come to the sum of configuration integrals over the separate low energy minima if these minima are separated by sufficiently high energy barriers [6, 7]. So, the docking accuracy is defined by the completeness of the low energy minima finding and by the accuracy of the configuration integral calculation in each of these minima.

Docking without the preliminary calculated energy grid requires much more computational resources because the protein-ligand energy has to be computed in the frame of the whole given force field for each system conformation appearing in the minima search algorithm. Such docking programs, FLM [7] and SOL-T [8], have been developed for the rigid target protein and the flexible ligand. The parallel FLM program can perform the comprehensive minima search either in vacuum or with the rigorous implicit solvent model [7] but at the expense of too large supercomputer resources – about 20000 CPU*h per one complex. The parallel SOL-T program employs the novel tensor train global (TT) optimization algorithm and it requires much less supercomputer resources than FLM. The docking positioning accuracy of FLM and SOL-T in vacuum for the rigid protein are comparable with one another at least for some test complexes [8]. Also it is demonstrated [7] that the ligand positioning accuracy is much better when the recent quantum chemical semiempirical methods, PM7 [9] and PM6 [10], are used instead of classical force fields and water solvent is taken into account.

Algorithms of most modern docking programs are based on the docking paradigm [7, 8, 9]. This paradigm assumes that the ligand binding pose in the active site of the target protein corresponds to the global minimum of the protein-ligand energy or is near it and the docking problem is reduced to the global optimization problem on the multi-dimensional protein-ligand energy surface. The dimensionality of this surface (d) is defined by the number of protein-ligand system degrees of freedom and commonly used docking algorithms, e.g. the genetic algorithm, are not able to perform docking for $d \geq 25$. Therefore docking of a flexible ligand into a flexible target protein requires more effective global optimization algorithms. The present study demonstrates that it is possible to perform successfully such docking employing the novel tensor train global optimization algorithm [8], [11]. We describe here main features of this novel algorithm, the respective program SOL-P for docking flexible ligands into target proteins with moveable atoms [12, 13] and the results of validation of the ligand positioning accuracy for a test set of 30 protein-ligand complexes [8].

2 Materials and Methods

For the realization of the novel docking algorithm we use the MMFF94 force field [14] in vacuum. The results will be much better, if either MMFF94 is used with the solvent model or PM7 is used with the solvent model [7, 9]. While looking for low-energy minima, ligands are considered to be fully flexible and some of protein atoms are moveable. The force field determines energy of the protein-ligand complex for its every conformation. The MMFF94 force field combines sufficiently good parameterization based on ab initio quantum-chemical calculations of a broad spectrum of organic molecules and the well-defined procedure of atom typification applicable to an arbitrary organic compound. MMFF94 is implemented in the SOL docking program [4] used successfully for new inhibitors development, e.g. see [15].

2.1 TT-docking

The novel docking algorithm (TT-docking) [8, 11] utilizes the TT global optimization method. It is based on the novel methods of tensor computations.

If d is the number of degrees of freedom of the protein-ligand complex, then we can introduce a grid in the configuration space with n_i nodes in each direction $i = 1,2 \ldots d$. If the grid is fine enough, then the solutions of continuous and discrete problems are expected to be close.

The basis of this consideration is the Tensor Train (TT) decomposition [16, 17] of a tensor $A \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ in the form:

$$A(i_1,\dots,i_d) \approx \sum_{\alpha_1=1,\dots,\alpha_{d-1}=1}^{r_1,\dots,r_d} G_1(i_1,\alpha_1)G_2(\alpha_1,i_2,\alpha_2)\dots G_{d-1}(\alpha_{d-2},i_{d-1},\alpha_{d-1})G_d(\alpha_{d-1},i_d) \quad (2)$$

The numbers $r_1,\dots,r_{d-1}$ are called TT-ranks of the tensor; for convenience, dummy ranks $r_0 \equiv r_d \equiv 1$ are also introduced. The 3-dimensional tensors $G_i \in \mathbb{R}^{r_{i-1}\times n_i \times r_i}$ are called cores or carriages of the tensor train. If TT-ranks are reasonably small, then the TT decomposition possesses several very useful properties [16, 17]. However, we cannot afford computing or storing all the elements for large tensors. Therefore, it becomes crucial to have for tensors a fast approximation method utilizing only a small number of their elements. Such a method was proposed and called the TT-Cross method [18]. It heavily exploits the matrix cross interpolation [19, 20, 21, 22, 23] algorithm applied cleverly, although heuristically, to selected submatrices in the unfolding matrices of the given tensor. The matrix $A_k \in \mathbb{R}^{n^k\times n^{d-k}}$, $A_k(i_1\dots i_k, i_{k+1}\dots i_d) = A(i_1,i_2,\dots,i_d)$ is called the k-th unfolding matrix of the tensor A. Such matrices are intrinsically linked with the TT-decomposition, TT-rank r_k is just the rank of the matrix A_k.

To explain the idea of the global optimization method consider a rank-1 matrix $A = uv^T \in \mathbb{R}^{m\times n}$. It is evident that the largest magnitude element of the matrix could be easily found in $m+n$ operations: if i and j are positions of the largest magnitude element in vectors u and v, respectively, then the required element is A_{ij}. Moreover even if factors u and v are unknown, such element could be found in $m+n$ evaluations of matrix elements. For this purpose, select any nonzero column of the matrix and find its largest magnitude element. Then select the row containing that element. The largest magnitude element of the matrix is the largest magnitude element of that row.

It was noticed, that latter strategy finds the largest magnitude element of the matrix with high probability even if matrix is not a rank-one matrix (though in this case more than $m+n$ elements should be evaluated, the search continues until the element is of the largest magnitude in both its row and column). Of course, this is evident if matrix is very close to a rank-one matrix. But such a strategy works with high probability even if the error in the optimal rank-one approximation of the matrix is quite large (as is proved by A.Osinsky, a good approximation exists if the error is even $1/8$ of the matrix Frobenius norm).

Moreover, consider a rank-2 matrix, for which a rank-one approximation is not very accurate. Apply the above strategy to the original matrix, perform the Gauss elimination with the selected element as a pivot and then apply the search strategy again. The largest in magnitude element with high probability is within evaluated elements of the matrix.

This is just how the matrix cross approximation method [20, 21] works. This method performs the search of the largest in magnitude matrix element, uses the found element to perform the Gauss elimination (constructing its factors but not performing elimination for all matrix elements) and repeats operations with the obtained matrix until the stopping criteria is met. Great advantage of the method is that it does not evaluate all matrix elements but only $O\big((m+n)r\big)$ of them, where r is the approximation rank. Also it has low complexity: $O\big((m+n)r^2\big)$ arithmetic operations. Moreover, the approximation obtained by this method is quasioptimal, *i.e.* its accuracy is close (by a not very large factor) to the accuracy of the optimal rank-r approximation, especially when the rank is small.

So, the matrix cross interpolation method could be used as a simple global optimization method as it finds the largest in magnitude element among all evaluated elements. More sophisticated variants of such optimization method use the local optimization of interpolation points because these points with high probability are near to local optima with large values. Such methods find in practice a global optimum with the ranks much less than those which are needed for a good approximation. Usually the parameter r_{max} limiting the rank from above is introduced to reduce the number of operations. Complexity of the global optimization is $O\big((m+n)r\big)$ function evaluations, $O\big((m+n)r^2\big)$ arithmetic operations and $O(r)$ local optimizations.

The TT-Cross approximation method applies to tensors (multi-dimensional arrays) [18]. It uses the matrix cross approximation method and pursues the same goal, which is to construct the approximation of a tensor in such a way that only small number of its elements are picked up. For tensors this is even by far more important than for matrices because the number of elements of many practical tensors is so huge that it cannot be computed or stored in any memory we may have at our disposal. If approximation ranks are reasonably low, the method evaluates only the logarithmic number of the total amount of tensor elements.

The idea of TT-cross approximation method is based on the following fact. Let a set I consist of r row indicies of A and a set J contain r column indicies, and let $A(I,J)$ be a submatrix of volume (modulus of the determinant) that is close to the maximal one among all submatrices of order r. Then a sufficiently good approximation is as follows:

$$A \approx A(:,J)A(I,J)^{-1}A(I,:). \quad (3)$$

Here $A(:, J)$ means the matrix consisting of columns of the A matrix with indicies from J, similarly A(I, :) means the matrix consisting of rows of the A matrix with indices from I.

To facilitate explanation, let us introduce some special matrices associated with tensors. For the tensor $T \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ the matrix $T_k \in \mathbb{R}^{n_1 \cdots n_k \times n_{k+1} \cdots n_d}$ is k-th *unfolding matrix* of this tensor if its elements are just reordered elements of the given tensor:

$$T_k(i_1 \dots i_k, i_{k+1} \dots i_d) = T(i_1, \dots, i_d). \tag{4}$$

Let us consider a T_1 matrix. If we know r_1 rows I_1 and columns J_1 for which $T_1(I_1, J_1)$ has large enough volume then using (3) we obtain:

$$T_1 \approx T_1(:, J_1) T_1(I_1, J_1)^{-1} T_1(I_1, :), \tag{5}$$

Denote $T_1(:, J_1) T_1(I_1, J_1)^{-1}$ as a matrix G_1 of size $n_1 \times r_1$ and rewrite equation (5) elementwise taking in account that elements of matrices T_1, T_2 and tensor T are the same:

$$T(i_1, \dots, i_d) \approx \sum_{\alpha_1=1}^{r_1} G_1(i_1, \alpha_1) T(\alpha_1, i_2, \dots, i_d) = \sum_{\alpha_1=1}^{r_1} G_1(i_1, \alpha_1) T_2(I_1(\alpha_1) i_2, i_3 \dots, i_d). \tag{6}$$

Note that $T_2(I_1(\alpha_1) i_2, i_3 \dots, i_d)$ are elements of the T_2 submatrix with rows selected in a special way. Denote this submatrix by $\tilde{T}_2$. Assuming good enough sets of rows I_2 and columns J_2 of size r_2, for this matrix we can obtain:

$$\tilde{T}_2 \approx \tilde{T}_1(:, J_2) \tilde{T}_2(I_2, J_2)^{-1} \tilde{T}_2(I_2, :). \tag{7}$$

Denote by $G_2 \in \mathbb{R}^{r_1 \times n_2 \times r_2}$ the tensor for which the matrix $\tilde{T}_1(:, J_2) \tilde{T}_2(I_2, J_2)^{-1}$ is the first unfolding matrix and substitute (7) to (6) using those elements of matrices $\tilde{T}_2$, T_3 which are elements of tensor T:

$$\begin{aligned} T(i_1, \dots, i_d) &\approx \sum_{\alpha_1=1, \alpha_2=1}^{r_1, r_2} G_1(i_1, \alpha_1) G_2(\alpha_1, n_2, \alpha_2) T_2(I_2(\alpha_2), i_3 \dots i_d) \\ &= \sum_{\alpha_1=1, \alpha_2=1}^{r_1, r_2} G_1(i_1, \alpha_1) G_2(\alpha_1, n_2, \alpha_2) T_3(I_2(\alpha_2), i_3, i_4, \dots, i_d). \end{aligned} \tag{8}$$

Now $T_3(I_2(\alpha_2), i_3, i_4, \dots, i_d)$ are elements of the submatrix of the T_3 matrix with rows selected in a special way. Denote it by $\tilde{T}_3$ and continue the procedure.

After repeating the procedure described above for $\tilde{T}_3$, …$\tilde{T}_{d-1}$ and denoting $\tilde{T}_d$ by G_d the approximation of the tensor in the TT format is obtained:

$$T(i_1, \dots, i_d) \approx \sum_{\alpha_1=1, \dots, \alpha_{d-1}=1}^{r_1, \dots, r_d} G_1(i_1, \alpha_1) G_2(\alpha_1, n_2, \alpha_2) \dots G_{d-1}(\alpha_{d-2}, n_{d-1}, \alpha_{d-1}) G_d(\alpha_{d-1}, n_d) \tag{9}$$

Note that the approximation error may grow exponentially with d, but in practice it is not large even for d of several hundreds.

The problem of the procedure described above is that matrices T_1, $\tilde{T}_k$ have a lot of columns, especially those considered first. So, finding a submatrix of the large volume in this matrix is a nontrivial task. But, if some small sets of columns which contain the large volume are known (at the start such columns could be selected by some mathematical assumptions or randomly), then a submatrix of large volume could be found in a fast way by the matrix cross approximation method. When the whole procedure is completed, we obtain a tensor approximation in the TT-format (may be not good enough) and rows containing submatrices of large enough volume. Next we can start this

procedure in the reversed order (or, equivalently, for the tensor with the reversed order of indicies). After the completion of this procedure the new approximation of the tensor and columns containing large volume submatrices are obtained. Such iterations could be repeated, for example, until the difference between two consequent approximations (fast calculation of this difference is possible in the TT format) becomes sufficiently small.

The TT-Cross method is transformed into a global optimization strategy by the same way as it is done for the matrix cross approximation method, in the result the largest in magnitude evaluated element is close to the largest in magnitude element of tensor. More fast convergence could be obtained using the local optimization of pivots obtained by the matrix cross method. To reduce the number of evaluations, the maximal rank is bounded by r_{max}. After the rank limitation iterations could possibly never converge and the *maximal iterations number* parameter is introduced.

Note that such a global optimization strategy allows us to find only the largest in magnitude element. For other optimization problems, e.g. for the docking problem which is the global minimization problem, the problem should be transformed into an equivalent problem of the largest magnitude search. It could be easily done by some monotonic continuous function. Selection of such a function is a nontrivial task as this function must separate optimums as good as possible. For the docking problem it is convenient to apply the TT magnitude maximization to the functional $f(x, E_*) = exp\{100arccot[E(x) - E_*]\}$, where $E(x)$ is the dimensionless MMFF94 energy for the given configuration x of the protein-ligand complex, E_* is the global minimum found on previous iteration.

For continuous problems, such as the docking problem, at first the grid must be introduced to obtain a tensor. For such problems, some additional artificial tensorisation might be very useful: instead of applying the method to the d-dimensional tensor with size n in each direction, the $D = md$-dimensional tensor with the size of 2 in every dimension is used. In this case, since the method complexity depends linearly on the tensor dimensionality and the size in each direction, the complexity grows logarithmically with the grid size and it is possible to use very fine grids. Note that artificial tensorisation may increase the number of parameters of the TT-optimization method (r_{max} and the maximal iterations number) which are needed to find optimum robustly. However in practice for most of global optimization problems these parameters stay almost the same.

The TT-docking iteratively performs the following steps:

1. Generation of submatrices of unfolding matrices using sets of tensor elements.
2. Interpolation of submatrices using TT-Cross method with the rank ≤r_max.
3. A set of interpolation points for each submatrix contains elements with large values in modulus.
4. Rough local optimization of interpolation points (protein-ligand poses) by the simplex method, addition of optimized point projections to the tensor and to the interpolation point sets.
5. Updating of each set of interpolation points of the unfolding matrix by merging the interpolation points of the previous unfolding matrix and ones of the subsequent unfolding matrix.
6. Addition of the best points (protein-ligand poses) to the interpolation point set of the unfolding matrix, and transition to step 1 using the obtained point set as the tensor elements.

The complexity of the TT global optimization method is O(dnr_{max}^2) functional evaluations, O(dr_{max}) local optimizations and O(dnr_{max}^3) arithmetic operations.

2.2 SOL-P Docking Program

The parallel SOL-P docking program is constructed on the base of the TT-docking algorithm (see above). The SOL-P program is developed for finding the low energy local minima spectrum of protein-ligand complexes, proteins or ligands including the respective global energy minimum. The energy of each molecule conformation is calculated directly in the frame of the MMFF94 force field [14] in vacuum without any simplification or fitting parameters. The conformation space of the system coordinates is formed by translations and rotations of the ligand as a whole, by the ligand torsions and also by Cartesian coordinates of the selected target protein atoms. The parallel MPI (message passing interface) based SOL-P program is written on C++ with usage of BLAS and LAPACK libraries. Main SOL-P parameters are: the maximal rank r_{max} of the TT-Cross approximation method, the power m of the discretization degree of the search space (there are $n = 2^m$ nodes along one dimension) and the number of iterations of the TT global optimization algorithm.

As it is mentioned in the previous section there is a rough local energy optimization in the TT-docking algorithm by the Nelder-Mead simplex method [24] within the Subplex algorithm [25] implemented as Sbplx program in NLOpt library [26].

2.3 Moveable Atoms

In the present consideration a protein atom is moveable when it is close to at least one of reference ligand poses. The protein atom is close to a ligand pose when the distance between this protein atom and at least one ligand atom is less than a given threshold. In the present work we took three ligand poses as reference ones: the ligand pose corresponding to the global protein-ligand energy minimum found by the FLM program [8] for the rigid protein, the locally optimized native ligand pose and the nonoptimized native ligand pose. Such choice of the reference ligand poses is taken here only for the uniformity of the consideration of all different proteins and ligands of the test set. Determination of moveable protein atoms is carried out by the specially written our original program Mark-PMA (Mark Protein Moveable Atoms) with the MLT (Moveable Layer Thickness) parameter defining the threshold distance. The MLT parameter is taken up to 3 Å in the present investigation.

2.4 Docking Procedure

The molecular data of the ligand and the protein with the marked moveable atoms are the input of the SOL-P program (shown in I stage in Fig. 1). The SOL-P program uses a cube centered in the geometrical center of the native ligand position in the crystallized protein-ligand complex as the spatial region for the low-energy minima search: all found ligand positions have their geometrical centers inside this cube (the docking cube). Each of moveable protein atoms can move inside its own small cube centered in the initial atom position taken from the crystallized protein-ligand complex. In this work we set the docking cube edge equal to 10 Å and the small cube edge equal to 1 Å. The SOL-P program performs MPI-parallelized search for the low-energy minima of protein-ligand complexes by TT-docking algorithm containing the rough local optimization by the simplex method. The ligand has six rotational-translational degrees of freedom as a whole rigid body plus torsional degrees of freedom for each single non-cyclic bond; each of the protein moveable atoms has three degrees of freedom – its Cartesian coordinates. Data about all found low-energy minima including protein-ligand configurations is too large to be saved in the molecular data format. These configurations are saved as the binary data (shown in Fig. 1 as "Binary data of all non-optimized minima").

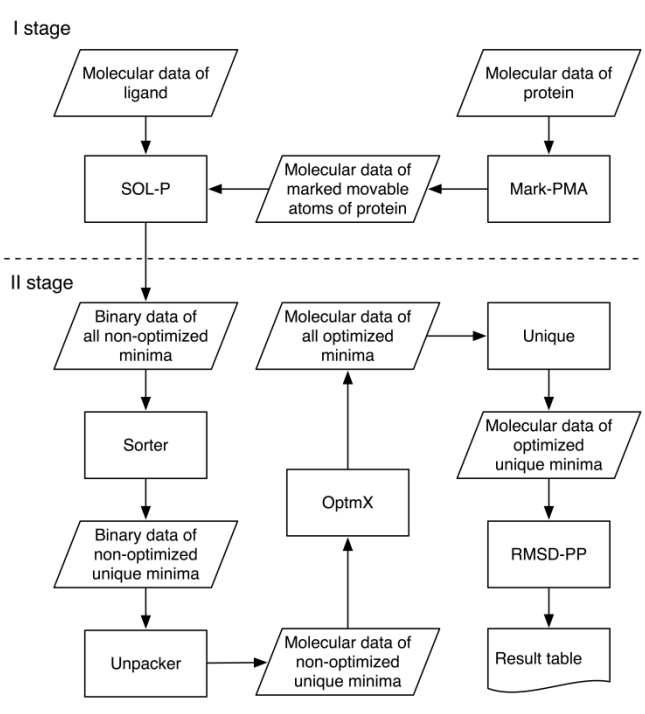

Fig. 1. Flowgraph of the program complex for low energy local minima search with flexible ligand and moveable target protein atoms. I stage: the data preparation and TT global energy minima search with the SOL-P program. II stage: the analysis of binary data with the "non-optimized minima" obtained from the SOL-P program and preparation of the table with the results and the final minima set.

2.5 Analysis of Local Minima

Docking of a flexible ligand into the target protein with moveable protein atoms differs strongly for docking into the rigid protein. In the former case we obtain after docking much larger volume of information about low energy minima than in the latter case. Different minima found in docking with moveable protein atoms are described by different protein-ligand conformations containing different ligand poses as well as different protein conformations. When docking is performed with a large number of degrees of freedom, e.g. with a flexible ligand and moveable protein atoms, the local energy optimization is too laborious and it is performed in the TT-docking not very precisely by the simplex method. All these peculiarities of the docking with a large number of degrees of freedom lead to importance of post-docking processing (post-processing): elimination of equivalent minima, more accurate local energy optimization and elimination of equivalent minima again. All these operations are performed at the stage II (see Fig. 1).

At the stage II in Fig. 1 the post-processing of low energy configurations stored in the binary data is performed with the Sorter program. The Sorter program sorts the “nonoptimized minima” by their MMFF94 energies in vacuum and excludes minima with equal ligand positions – only one minimum with the lowest energy is being kept. Two ligand positions are considered equal if RMSD between them is less than a given threshold (0.1 Å), where RMSD is calculated atom-to-atom without chemical symmetry accounting. Thus, all the remaining low-energy configurations (“unique non-optimized minima” in Fig. 1) have different ligand positions. Then, the Unpacker program performs exporting all unique low-energy configurations from the binary file to the file with the MOL2 molecular format. The post-processing of low energy protein-ligand configurations consists of performance of two programs: OptmX and Unique (Fig. 1). The OptmX program locally optimizes all of the “unique non-optimized minima”. For these purposes, the OptmX program uses L-BFGS algorithm [27, 28] applied to the local optimization of the MMFF94 energy function in vacuum with variations of Cartesian coordinates of all ligand atoms and moveable protein atoms. Each local optimization stopped when the energy change at several steps was less than 10^{-8} kcal/mol. Optimization of different minima is MPI-parallelized. After this optimization, the “all optimized minima” (Fig. 1) set is obtained. But many of these minima may become equal again. Therefore, we need to re-exclude similar minima. The Unique program excludes equal minima from the “all optimized minima” set as follows. Among several equal configurations only the minimum with the lowest energy is being kept as it is made in the binary data file post-processing by the Sorter program. However, in contrast to the Sorter program the protein moveable atoms are also taken into account in RMSD calculation, and the RMSD is calculated with chemical symmetry analysis. The decrease of the number of minima at the post-processing stage can be very large comparing with the number of minima found at the docking stage. For example, after the processing with the Sorter program there are 30365 and 28166 minima for the protein-ligand complexes with PDB ID 1MRW (with 30 moveable atoms) and 5BT3 (with 27 moveable atoms), respectively; however, after the precise local energy optimization with the OPTM-X program and filtering the obtained minima with the Unique program the numbers of different local energy minima decrease down to 7580 and 5891 minima for 1MRW and 5BT3, respectively.

Analysis of the local minima remaining after post-processing is carried out by the RMSD-PP program which calculates RMSD (with respect to all ligand atoms) between the ligand pose in a certain energy minimum of the protein-ligand complex and the ligand pose in the energy minimum corresponding to the native ligand position obtained after the local optimization from its configuration in the crystallized complex. The RMSD here is calculated taking into account the approximate chemical symmetry analysis [13] and it is a good metric to estimate geometrical difference between two configurations of a protein-ligand complex; it can correctly discard geometrical pseudo-differences such as phenyl residue flip, comparing to the native atom-to-atom RMSD calculation.

As a result the RMSD-PP program creates in its output (Fig. 1) the resulting table containing: the minimum index, the minimum energy, RMSD from the optimized native configuration and the distance from the ligand geometric center in the given minimum to the ligand geometric center in the optimized native configuration. The energy minima are sorted by their energy in the ascending order; that is, every minimum gets its own index equal to its number in this sorted list of minima. The lowest energy minimum has the index equal to 1.

Some minima from the list might be close in space to the optimized native ligand position. We designate the index of the minimum having RMSD from the optimized native ligand position less than 2 Å as “Index of the minimum Near Optimized Native” or “INON.” If there are several such minima which are close to the optimized native ligand position, we will choose the minimum with the lowest energy (with the lowest index) as “INON”. When INON=1 the docking paradigm is satisfied: the global minimum of the protein-ligand energy is near the native configuration. If there are no minima with the ligand pose near the optimized native configuration among all minima found by the SOL-P program, we use notation INON=inf.

In the present consideration we compare the energy minima found by the SOL-P program with ones obtained by the FLM program [7] with the same target function – energy in the frame of the MMFF94 force field in vacuum, for the same test set of 30 protein-ligand complexes [8] which are taken from the Protein Data Bank [29].

2.6 Parallel Performance of SOL-P

In docking problem (and many others) the evaluation of any tensor element has almost the same complexity. So, the parallelization is considered for this case. The parallel implementation of the matrix cross method is available [30] for such case. However, matrices which are used by TT global optimization strategy are relatively small, especially in the case when the additional artificial tensorisation is used. So, this parallel resource is very limited.

But for the TT global optimization and even for TT-cross approximation methods submatrices of different unfolding matrices are not necessary to be considered consequently and rows or columns used for the approximation are not necessary to be nested. In the case of the approximation this will lead to some additional (but independent for all unfolding matrices with the single communication between unfoldings number k and $k+1$ prior) computations for the construction the tensor approximation. In the global optimization strategy the approximation of the tensor is not constructed explicitly, so such additional computations are not needed.

To balance computations, all submatrices of unfolding matrices are selected of the same size (maximal amongst all original submatrices) and the approximation is performed till the same rank. As a positive side effect this leads to faster convergence and better robustness. Moreover, in the case of the global optimization, especially when the additional tensorisation is used, original sizes of these submatrices are very close to each other due to the rank limitation and to the equal size of each dimension.

Finally, in the parallel algorithm such operations are done for every unfolding submatrix independently at every iteration:

1. From the set of tensor elements P_k, which are obtained on the previous iteration, construct the set of unfolding matrix columns and rows. Rows are constructed in the same way as in the TT-cross procedure in the normal order, columns – as in the reversed order. The number of rows and columns will be approximately n times larger than the number of elements in P_k.
2. Using random rows and columns extend their number to $7nr_{max}$ each.
3. Perform the parallel matrix cross interpolation of the unfolding matrix submatrix of the order $7nr_{max}$ with the rank bounded by r_{max}.
4. Perform in parallel a small number of local optimization steps for obtained pivots and project them back to tensor elements.
5. Every unfolding has approximately $2r_{max}$ tensor elements now – matrix cross pivots and elements obtained by the projection of locally optimized points. Send positions of these elements to unfoldings number $k-1$ and $k+1$ and receive positions from them. Also, by the parallel reduction find r_{max} points with the best functional values amongst all unfoldings. After these operations each unfolding has about $7r_{max}$ elements which are used at the next iteration.

Note, that only steps 3 and 4 have high computational cost.

Parallel efficiency of such algorithm is highly dependent on the number of unfoldings (denote it as K) and the number of processors (p). If p is less than K then the parallel efficiency is the best when K is divisible by p. In the other case, the parallel efficiency is higher when p is divisible by K and it is even more better when additionally Kr_{max} or $7Knr_{max}$ is divisible by p. The maximal number of processors which is reasonable to use is $7Knr_{max}$.

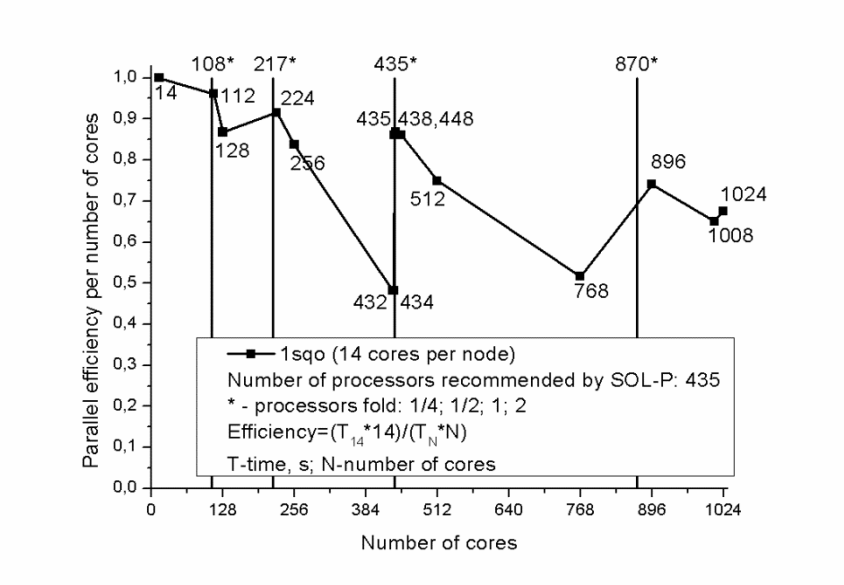

Fig. 2. Parallel efficiency of the SOL-P program for different numbers of core of the Lomonosov-2 supercomputer [31] for the 1SQO test complex with 6 protein moveable atoms, the ligand consisting of 34 atoms (4 torsions).

The multi-processor performance of SOL-P is investigated at the Lomonosov-2 supercomputer [31]. The results of SOL-P performance for the first set of TT-docking parameters ($r_{max} = 4$, $n = 2^{16}$ and the number of iterations equal to 15) are demonstrated in Fig. 2 for the 1SQO complex with 6 protein moveable atoms, the ligand consisting of 34 atoms (4 torsions) and with different numbers of cores. We see the non-monotonic behavior of the parallel efficiency $(T_{14} \times 14)/(T_N \times N)$ on the number of cores which are used for the calculations (at the nodes containing 14 cores per node). The detailed plots are different for different complexes but their general behavior is the same. For example, for the 1SQO complex $K = 435$ unfolding matrices were considered for constructed tensor. It is easy to see, that efficiency is maximal when the number of used cores p is close to the number which is divisible by K (for numbers of cores larger than K) or when the fractional part of K/p is close to zero.

3 Results

Performance of SOL-P is investigated for different values of the maximal rank $r_{max} = \{4, 8, 16\}$, the initial grid size $n = \{2^8, 2^{12}, 2^{16}\}$ and the number of iterations. Results of this testing demonstrate that for the higher initial grid size even the lowest tested maximal rank $r_{max} = 4$ is enough to find the optimum reliably and precisely. However, the increase of the initial grid size leads to slower convergence of the method and the iteration number must be larger (for $n = 2^{16}$ from 10 to 15 iterations are needed). The high grid size $n = 2^{16}$ for ranks 8 and 16 makes computations significantly slower, thus the initial grid size of $n = 2^{12}$ is used for ranks 8 and 16. For such initial grid size the computation time is reduced by 1.5 times and the number of iterations decreases. SOL-P with three parameter sets: $\{r_{max} = 16, n = 2^{12}\}$, $\{r_{max} = 8, n = 2^{12}\}$ and $\{r_{max} = 4, n = 2^{16}\}$ demonstrates similar ability to find the global energy minimum near the optimized native ligand position for several test complexes but the fastest performance is observed for $\{r_{max} = 4, n = 2^{16}\}$, and we choose the latter set with 15 iterations as the optimal parameters for the present investigation.

It is found that for some complexes (e.g. 1SQO: 4 ligand internal torsions and 34 ligand atoms) the docking paradigm is satisfied for the rigid protein as well as for the protein up to 35 moveable atoms. For some complexes (e.g. 3CEN: 7 torsions and 50 ligand atoms) the docking paradigm is satisfied only for sufficiently large number (13, 26, 48) of protein moveable atoms when INON is equal to 1 or 2. SOL-P finds the global energy minimum for this complex when 48 protein atoms are moveable and the dimensionality of the energy surface is equal to 157=144 (protein) + 13 (ligand). For such docking SOL-P uses about 9 hours at 512 core of the Lomonosov supercomputer [31, 32]. For other complexes (e.g. 4FT9: 5 torsions and 32 ligand atoms) the MMFF94 force field energy in vacuum is not adequate and the energy surface is so complicated that for the too large number of protein moveable atoms (42) SOL-P is not able to find minima near the native configuration.

The validation shows that SOL-P finds either the global minimum or one of low energy minima corresponding to the ligand pose being near the optimized native ligand pose for the rigid protein and/or for the protein with moveable atoms for more than two thirds of the whole test set of protein-ligand complexes (for 22 out 30) for these 22 complexes INON=1 or INON $\leq$ 25 and the docking paradigm is fulfilled for them in the frame of the MMFF94 force field in vacuum. The test complexes are collected in groups in respect with values of their INON index in Fig. 3.

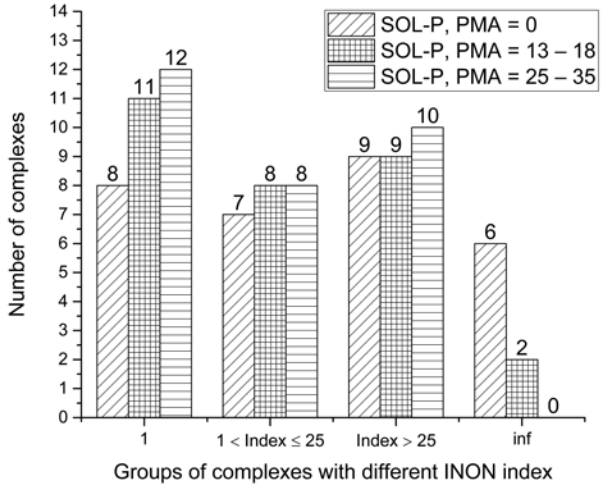

Fig. 3. Numbers of complexes with different values of INON index. PMA indicates the range of protein moveable atoms for the SOL-P program. INON is the index of the minimum having RMSD from the optimized native ligand position less than 2 Å; if there are several such minima, the minimum with the lowest energy (with the lowest index) should be taken.

Protein atoms mobility is crucial for 4 complexes (1J01, 1K1J, 1MQ6 and 3CEN) out of 30 ones: SOL-P does not find any minima near the optimized native ligand pose for docking into the rigid protein (INON=inf) but, when mobility of protein atoms is taken into account, docking finds near the optimized native ligand pose either the global minimum (INON=1) or one of the lowest energy minima (INON≤25). On the other hand, for rigid proteins SOL-P and FLM cannot find such minima (INON=inf) for 6 and 5 complexes, respectively. It is worth to note that SOL-P is able to find the minimum near the optimized native ligand pose for all 5 complexes where FLM is not able to do this. In tote, we can say that SOL-P, with and without protein moveable atoms, works not worse than the FLM program and much faster than the latter.

Our observation that neither SOL-P nor FLM can find any minimum near the optimized native ligand pose for 11 complexes (out of 30) is connected with inadequacy of the energy target function calculated in the frame of the MMFF94 force field in vacuum. It has been previously demonstrated [7] that protein-ligand energy calculation in the frame of the MMFF94 force field in solvent (with an implicit model) improves docking performance of the FLM program for the rigid proteins and with such target energy function SOL-P should also work better.

4 Conclusions

The novel algorithm is realized in the supercomputer SOL-P docking program where protein and ligand atoms mobility is taken into account simultaneously and equally. Energies of low-energy minima found in the docking procedure and respective ligand poses are carefully analyzed.

It is shown that the program is able to perform docking of a flexible ligand into the active site of the target protein taking mobility of assigned protein atoms into account: up to 157 degrees of freedom in the conformation space using about 9 hours at 512 core of the Lomonosov supercomputer [31, 32]. This is the first time when the docking program is able to perform successfully the global energy minimum search in the conformational space with such a large dimensionality. This result is achieved due to the novel docking algorithms (TT-docking) which is based on the so-called tensor train decomposition of multi-dimensional arrays and the TT global optimization method [8], [11].

The SOL-P docking performance is comparable with one of the FLM program [7] which executes the massive parallel local energy minima for rigid target proteins due to employment of much larger computing resources.

It is demonstrated that the docking paradigm is fulfilled for the target energy function calculated in the frame of the MMFF94 force field in vacuum for a flexible ligand and for a target proteins with 25 – 35 moveable atoms for two thirds of the whole test set of protein-ligand complexes. Interaction with solvent should increase this number. It is demonstrated that in some cases docking results are being improved even when small movements of protein atoms is taken into account in the docking procedure.

The present investigations became possible due to computing resources of M.V. Lomonosov Moscow State University supercomputer Lomonosov [32].

Acknowledgements

The work was financially supported by the Russian Science Foundation, Agreement no. 15-11-00025.

References

1. Sliwoski, G., Kothiwale, S., Meiler, J., Lowe, E.W.: Computational Methods in Drug Discovery. Pharmacol. Rev. 66, 334–395 (2014). doi: 10.1124/pr.112.007336
2. Sadovnichii, V.A., Sulimov, V.B.: Supercomputing technologies in medicine. In: Supercomputing technologies in science, education, and industry. Voevodin, V.V., Sadovnichii, V.A., Savin, G.I. (eds.) Moscow University Publishing, 2009, pp. 16–23 (in Russian).
3. Mobley, D.L., Dill, K.A.: Binding of small-molecule ligands to proteins: "what you see" is not always "what you get." Structure. 17(4), 489–498 (2009). doi:10.1016/j.str.2009.02.010.
4. Sulimov, A.V., Kutov, D.C., Oferkin, I.V., Katkova, E.V., Sulimov, V.B.: Application of the Docking Program SOL for CSAR Benchmark. Journal of Chemical Information and Modeling. 53(8), 1946-1956 (2013). doi:10.1021/ci400094h.
5. Antunes D.A., Devaurs D., Kavraki L.E.: Understanding the challenges of protein flexibility in drug design. Expert Opinion Drug Discovery. 10(12), 1301–1313 (2015). http://dx.doi.org/10.1517/17460441.2015.1094458.
6. Chen, W., Gilson, M.K., Webb, S.P., Potter, M.J.: Modeling protein-ligand binding by mining minima. Journal of Chemical Theory and Computation. 6(11), 3540–3557 (2010)
7. Oferkin, I.V., Katkova, E.V., Sulimov, A.V., Kutov, D.C., Sobolev, S.I., Voevodin, V.V., Sulimov, V.B.: Evaluation of docking target functions by the comprehensive investigation of protein-ligand energy minima. Advances in Bioinformatics. Vol. 2015, Article ID 126858, 12 pages. http://dx.doi.org/10.1155/2015/126858.
8. Oferkin, I.V., Zheltkov, D.A., Tyrtyshnikov, E.E., Sulimov, A.V., Kutov, D.C.: Evaluation of the docking algorithm based on tensor train global optimization. Bulletin of the South Ural State University, Ser. Mathematical Modelling, Programming & Computer Software. 8(4), 83–99 (2015). doi:10.14529/mmp150407.
9. Sulimov, A.V., Kutov, D.C., Katkova, E.V., Sulimov, V.B.: Combined docking with classical force field and quantum chemical semiempirical method PM7. Advances in Bioinformatics. Vol. 2017 (2017), Article ID 7167691, 6 pages. https://doi.org/10.1155/2017/7167691
10. Pecina, A., Meier, R., Fanfrlík, J., Lepšík, M., Řezáč, J., Hobza, P., Baldauf, C.: The SQM/COSMO filter: reliable native pose identification based on the quantum-mechanical description of protein–ligand interactions and implicit COSMO solvation. Chem. Commun. 52, 3312-3315 (2016).
11. Zheltkov, D.A., Oferkin, I.V., Katkova, E.V., Sulimov, A.V., Sulimov, V.B.: TTDock: Docking Method Based on Tensor Train. Vychislitelnie metody i programmirovanie (Numerical methods and programming). 14, 279–291 (2013). (in Russian) Available: http://num-meth.srcc.msu.ru/english/zhurnal/tom_2013/v14r131.html. Accessed 2017 April 10.
12. Sulimov, A., Zheltkov, D., Oferkin, I., Kutov, D., Tyrtyshnikov, E.: Novel gridless program SOL-P for flexible ligand docking with moveable protein atoms. In: 21st EuroQSAR Where molecular simulations meet drug discovery, September 4–8, 2016. Aptuit Conference Center, Verona Italy, Abstract book, OC15, p.52, www.euroqsar2016.org.
13. Sulimov, A.V., Zheltkov, D.A., Oferkin, I.V., Kutov, D.C., Katkova, E.V., Tyrtyshnikov, E.E., Sulimov, V.B.: Evaluation of the novel algorithm of flexible ligand docking with moveable target protein atoms. Computational and Structural Biotechnology Journal. 15, 275-285 (2017). doi: 10.1016/j.csbj.2017.02.004
14. Halgren, T.A.: Merck Molecular Force Field. I. Basis, Form, Scope, Parameterization and Performance of MMFF94. Journal of Computational Chemistry, 17, 490–519 (1996).

15. Sinauridze, E.I., Romanov, A.N., Gribkova, I.V., Kondakova, O.A., Surov, S.S.: New Synthetic Thrombin Inhibitors: Molecular Design and Experimental Verification. PLoS ONE. 6(5), e19969 (2011). doi:10.1371/journal.pone.0019969.
16. Oseledets, I.V., Tyrtyshnikov, E.E.: Breaking the curse of dimensionality, or how to use SVD in many dimensions. SIAM Journal on Scientific Computing. 31(5), 3744–3759 (2009). doi:10.1137/090748330.
17. Oseledets I.V.: Tensor-Train Decomposition. SIAM Journal on Scientific Computing. 33(5), 2295–2317 (2011). doi:10.1137/090752286.
18. Oseledets I.V., Tyrtyshnikov E.E.: TT-Cross approximation for multidimensional arrays. Linear Algebra and its Applications. 432(1), 70–88 (2010). doi:10.1016/j.laa.2009.07.024.
19. Goreinov, S.A., Tyrtyshnikov, E.E., Zamarashkin, N.L.: Pseudo-skeleton approximations of matrices. Reports of Russian Academy of Sciences, 342(2), 151–152 (1995). http://dx.doi.org/10.1016/S0024-3795(96)00301-1.
20. Goreinov, S.A., Tyrtyshnikov, E.E., Zamarashkin, N.L.: A theory of pseudo-skeleton approximations. Linear Algebra Appl. 261, 1–21 (1997). http://dx.doi.org/10.1016/S0024-3795(96)00301-1.
21. Tyrtyshnikov, E.E.: Incomplete cross approximation in the mosaic-skeleton method. Computting. 64(4), 367–380 (2000). doi:10.1007/s006070070031.
22. Goreinov, S.A., Tyrtyshnikov. E.E.: The maximal-volume concept in approximation by low-rank matrices. Contemporary Mathematics. 208, 47–51 (2001)
23. Goreinov, S.A., Oseledets, I.V., Savostyanov, D.V., Tyrtyshnikov, E.E., Zamarashkin, N.L.: How to find a good submatrix. In: Research Report 8-10, ICM HKBU, Kowloon Tong, Hong Kong, 2008. doi:10.1142/9789812836021_0015.
24. Nelder, J.A., Mead, R.: A simplex method for function minimization. The Computer Journal. 7, 308-313 (1965).
25. Rowan, T.: Functional Stability Analysis of Numerical Algorithms. In: Ph.D. thesis, Department of Computer Sciences, University of Texas at Austin, 1990.
26. Steven, G.J.: The NLopt nonlinear-optimization package. http://ab-initio.mit.edu/nlopt
27. Byrd, R.H., Lu, P., Nocedal, J., Zhu, C.: A limited memory algorithm for bound constrained optimization. SIAM J. Sci. Comput. 16(5), 1190–1208 (1995). http://dx.doi.org/10.1137/0916069.
28. Zhu, C., Byrd, R.H., Lu, P., Nocedal, J.: Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Transactions on Mathematical Software. 23(4), 550–560 (1997). doi:10.1145/279232.279236.
29. Berman, H.M., Westbrook, J., Feng, Z.: The protein data bank. Nucleic Acids Research. 28(1), 235–242 (2000). http://www.rcsb.org/pdb/home/home.do
30. Zheltkov, D.A., Tyrtyshnikov, E.E.: Parallel Implementation of Matrix Cross Method. Vychislitelnye metody i programmirovanie (Numerical Methods and Programming). 16, 369–375 (2015). (in Russian)
31. MSU Supercomputers: Lomonosov-2. URL: http://hpc.msu.ru/?q=node/159. Accessed 2017 May 30.
32. Sadovnichy, V.A., Tikhonravov, A.V., Voevodin, V.V., Opanasenko, V.: “Lomonosov”: Supercomputing at Moscow State University. In: Contemporary High Performance Computing: From Petascale toward Exascale. CRC Press. pp. 283–307 (20013).