Knowing when to give up: early-rejection stratagems in ligand docking

Gwyn Skone · Irina Voiculescu · Stephen Cameron

Received: 6 March 2009 / Accepted: 25 June 2009 / Published online: 8 July 2009

Abstract Virtual screening is an important resource in the drug discovery community, of which protein–ligand docking is a significant part. Much software has been developed for this purpose, largely by biochemists and those in related disciplines, who pursue ever more accurate representations of molecular interactions. The resulting tools, however, are very processor-intensive. This paper describes some initial results from a project to review computational chemistry techniques for docking from a non-chemistry standpoint. An abstract blueprint for protein–ligand docking using empirical scoring functions is suggested, and this is used to discuss potential improvements. By introducing computer science tactics such as lazy function evaluation, dramatic increases to throughput can and have been realized using a real-world docking program. Naturally, they can be extended to any system that approximately corresponds to the architecture outlined.

Keywords Ligand docking · Empirical scoring function · Structure-based drug design · Virtual screening · Early-rejection

Introduction

As the power of computers has increased through the last few decades, their application to the search for new medicinal compounds has exploded. It is now possible to perform tasks which were completely implausible only a few years ago. One field that has emerged in this new world of research has been that of High-Throughput Virtual Screening.

Screening a database of ligands for those showing a binding affinity with a target molecule may be done in many ways; one commonly used technique is docking. This searches a protein's spatial and chemical environment using some scoring function to finally produce some manageable number of filtered results as the final output. Many such algorithms have been developed, all with their own specialities and relative benefits, and various comparative assessments of these exist [1–4]. The repetitiveness of this task is what makes it ideal for computation. However, the convenience of a machine does not mean that careful analysis of the task is not necessary. A brute force implementation will still not provide answers within a useful amount of time and resources. The work can be minimized by being thoughtful about the actual requirements from the task. This paper discusses some important examples of this principle.

When discussing the mechanism of docking, a bipartite system is generally described (see Fig. 1): the *search method* that generates placements of ligands in the environment of the receptor, and the *scoring function* that evaluates each of these placements. For greater detail when considering the practicalities of execution, the overall process may be broken down into layers that correspond approximately to nested loops in any likely software implementation. A useful description has six levels, shown in Table 1.

For each of these layers, an estimated order of magnitude has been given to provide a sense of the problem's scale. These may vary slightly according to the particular molecules and algorithms used, but can help to sense where bottlenecks might occur in the process. In particular, it is clear that making the scoring function execute as quickly and efficiently as possible is crucial, since this will be

G. Skone (✉) · I. Voiculescu · S. Cameron
Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD, UK
e-mail: gwyn.skone@comlab.ox.ac.uk

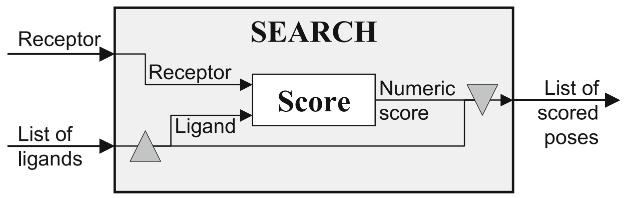

Fig. 1 Major components of the docking process

Table 1 Abstract layers of the naïve docking process

	For each…	*Order*
SEARCH	1. ligand	
	2. conformation	10^2
	3. pose	$(10^2)^6 = 10^{12}$
Score	4. ligand atom	10^1
	5. receptor atom	10^3
	6. interaction	10^1
	…calculate contribution	

Whilst myriad poses are possible, few search methods would consider them all exhaustively. In practice, the number of poses considered will be closer to the order 10^4

needed for each of the vast number of placements. The scale of layer 3 is quite variable and depends on the range of values to be covered by the search, but as a baseline estimate will span at least a six-dimensional space (three each of translation and rotation).

This generalization has assumed that empirical scoring functions [5] are used. These model several different interactions between a receptor (R) and a ligand (L), weighted according to observed example cases. Equation 1 shows the general form of such a function: it accumulates a total score over all pairs of atoms and the kinds of interaction considered (bond potential, hydrophobic burial, etc.). The independent interaction functions are denoted by $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_{N_{\text{int}}}$. This model does not preclude single physical functions, however, since those may be regarded as cases where $N_{\text{int}} = 1$.

$$\mathbf{Score}(R, L) = \sum_{l \in L_{\text{atoms}}} \sum_{r \in R_{\text{atoms}}} \sum_{i=1}^{N_{\text{int}}} \mathbf{F}_i(r, l) \qquad (1)$$

Any optimizations that can be made to this generic architecture should be applicable in some way to many implementations of docking systems.

Methods

Our approach to docking in general begins by giving some consideration to the information sought from the process—a scoring of the suitability of a ligand for binding to a receptor, and some poses that demonstrate the fit. One important observation that should be exploited is the relative vagueness of this requirement. These scores (particularly when using empirical functions) are inevitably approximate, even when they estimate binding free energies, and so should not be interpreted with a high degree of precision. Taking this into account, a less accurate score might be equally serviceable and quicker to obtain when this value is needed only to demonstrate the impossibility of a pose.

Put concisely, the qualitative *meaning* of a score—the interpretation as being plausibly docked or not—is often more important than its numerical value, regardless of the function that produced it. This should be borne in mind when designing and/or implementing software based on scoring functions.

Scoring function evaluation

The values returned by a scoring function fall generally on an unbounded scale of real numbers. These represent assessments of a ligand's placement from various degrees of 'abysmal' through to a relatively narrow band of 'good' scores. One can select a threshold on this scale to cap a bottomless pit of values that represent indubitably unsuitable positions—if a ligand is buried or clashes significantly, it does not matter *how* bad the score is, merely that it *is* beyond acceptability.

It is also worth remembering that suitable binding states form a very small subset of the search space—this is, after all, why docking is like hunting through the proverbial haystack—and so it is a minority of calculations that need a complete and detailed evaluation. Consequently, much time can be wasted on the execution of scoring functions for arrangements that will, upon human assessment or aggregation in a best-*n* list, be promptly discarded again. It would be efficient to predict whether a score will fall into the pit, and produce an estimated indicative result instead of a fully calculated value.

In order to avoid spending time and resources pursuing work that will lead nowhere useful, an early exit point is introduced into the computation loops. Specifically, after the accumulation of each ligand atom's contribution (i.e. each iteration of the outermost summation in Eq. 1), the running total is used to decide whether it is still possible to reach an interesting final score. If this incomplete sum has passed a certain threshold which indicates that it is beyond the realm of suitability, then it is not calculated any further and the current total is returned immediately as the final result.

The prospect of abandoning outright some cases without fully evaluating them might seem to jeopardize good results, but provided a sensible rejection threshold is used the method can be justified. This is a principle well-established in computer science, where the term ‘lazy evaluation’ is applied to systems that ignore calculations until their results are required, if ever. Almost all scoring functions include a simple potential term—typically based on Van der Waals radii—and these have by far their largest values at short, clashing distances. Hence, an unacceptably buried ligand will aggregate quickly a very poor total whose magnitude dwarfs any favourable interactions that might still occur.

To identify such a value, find $B = b_1 + \cdots + b_{N_{\mathrm{int}}}$: the sum of the best-possible contributions of each $\mathbf{F}_i$ term to a score (accounting for the likely maximum number of receptor atoms that might influence any single ligand atom). Choosing a rejection threshold of $-B(|L_{\mathrm{atoms}}|-1)$ for a ligand L will then be absolutely safe, since upon reaching this point it can be concluded that the total cannot return to (or pass) zero even with wholly ideal interactions. In practice, this is a rather conservative cut-off, but it is still quite possible to abort scoring after assessing only the first ligand atom if this is badly buried.

For efficiency, scoring calculations that survive the first few iterations might be allowed to complete in full without further interference. This would eliminate the test required after each atom and might shrink the already small cost of watching for early-rejection opportunities.

A further improvement is to look for the worst atomic contributions first in the hope that the rejection threshold might be reached as soon as possible. The commutativity of summation allows us to accumulate the terms of the scoring function in any order. It would make sense to enumerate them arranged by exposure: the relative likelihood of clashing with a receptor’s surface. A simple implementation could use the distance of each atom from the ligand’s centroid, placing the farthest first. By doing this sorting stage when each ligand conformation is introduced (the beginning of step 2 in Table 1), any slightly buried placements should be identified as quickly as possible.

It might seem more effective to sort the ligand’s atoms by their distance from the receptor’s centroid, thus ensuring that those closest to the interaction surface would be seen first, whereas sorting relative to the ligand’s centroid merely delays assessment of the atoms that are least likely to clash. This more comprehensive approach has one major disadvantage, however: it would necessitate a sorting operation for every placement (step 3 in Table 1), rather than every molecule as in the design suggested.

Search quotas

Having established a means of using quickly calculated approximate scores to abbreviate steps in a search method, the overall exploration process may also be improved. The same principle of stopping as soon as an unfavourable outcome is predicted can be reapplied here, albeit with less precision.

The search method generates placements of a ligand conformation, passes them to the scoring function, and collects the best-scoring results into a list. Several techniques for this task have been published, including genetic algorithms, probabilistic roadmaps, and molecular dynamics [6–8]. Whilst an exhaustive search of the six-dimensional space (assuming completely rigid docking) could also be used, it would be quite impractical for high-throughout use.

Figure 2 shows the basic execution loop employed in a search, corresponding to step 3 of Table 1. The issue of pose selection has been well-studied already under the umbrella of search algorithms. However, the selection of a useful termination condition also warrants consideration. In general, this condition is usually the completion of a predefined number of cycles or the emergence of some quality or consistency in the results.

The results will be aggregated with those from every other conformation being docked. If a quota is imposed on the final number of results that may be returned from *all* cases, then it is possible that several of the results from an individual search will fall off the bottom of the list and be discarded. Since the set of results will almost always need further detailed analysis, the quantity of output data is often limited to a certain number of docked poses, and so the notion of a quota is justifiable. Rather than doggedly pursuing a docking of a relatively unsuitable ligand conformation, only to produce results that will ultimately be omitted from any output (or else ignored in later analysis), the termination condition could take into account the results already present in the final list.

One simple way to do this would be to track the worst score present in that list (after its quota has been filled), and abandon searches that have not found any placements that scored at least as well after some number of cycles—perhaps around two-thirds of the usual count. This uses a failure to find any slightly interesting poses after a shorter search as the basis for deducing a dearth of good poses in the whole space. That deduction, although not certain, may be rational if made only when the search has been afforded opportunity to make a reasonable exploration of the space.

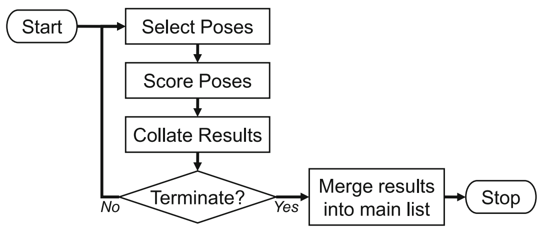

Fig. 2 General form of a docking search method

Table 2 Illustration of quota-based early-rejection

Example scored pose lists after ligand conformations…

L_1	L_2	L_3	L_4
(Pose1a, 9)	(Pose1a, 9)	(Pose1a, 9)	**(Pose4a, 10)**
(Pose1b, 8)	(Pose1b, 8)	(Pose1b, 8)	(Pose1a, 9)
(Pose1c, 4)	**(Pose2a, 7)**	(Pose2a, 7)	(Pose1b, 8)
(Pose1d, 1)	**(Pose2b, 6)**	(Pose2b, 6)	(Pose2a, 7)
....	(Pose1c, 4)	(Pose1c, 4)	(Pose2b, 6)
....	**(Pose2c, 2)**	(Pose2c, 2)	(Pose1c, 4)

Assuming for brevity a search method producing 4 poses and a quota of 6: after L_1 and L_2 have been docked and results merged into the list, the quota of 6 poses has been filled. Now, the worst-case score is 2: the search termination condition (see Fig. 2) has stopped after some specified cycle since no pose of L_3 had **Score**$(R, L_3) \geq 2$. After L_4, the worst-case score has risen to 4

Bold entries are the newly-inserted entries in each cycle (column)

It is assumed that the scores of the poses chosen in each cycle tend to improve over time, or at least remain of a similar standard. This requirement should be acceptable, since even a completely random selection method will, in general, maintain a roughly constant quality of pose set.

Table 2 illustrates this with an artificial example: the third conformation L_3 was rejected before its search completed because it did not appear likely to produce any poses highly ranked in the result list with a quota of 6. Since the worst score recorded in the main list can only improve with each batch of results added, the likelihood of a conformation being rejected after a particular number of cycles increases for each case. Ideally, then, the order in which conformations are presented could be chosen so as to dock the most promising cases first. How to identify a more promising case, however, is beyond the scope of this paper.

It should be noted that it may be helpful to extend the quota specification slightly, by allowing the output count to differ from the quota used for tracking worst-cases. For example, to use the illustration of Table 2 again, the result list of 6 might be cut to the top 4 for the final output. A larger quota than the final result list size could be used to offer the individual conformations a better chance of reaching the worst-case score, and thus not be discarded.

Results

A reference program is required to investigate the effect of these stratagems. The tests described below were conducted using a system called *DOX*. This implements a variation on Lamarckian genetic algorithms [9, 10]; whereas the usual definition includes local searches from some proportion of the population at every generation, *DOX*'s method uses a conventional genetic algorithm to search the translation/rotation space and then applies a local simplex optimization [11] to each member of the final population. The program takes pre-generated ensembles of ligand conformations and docks them using one of two scoring functions: the simple Piecewise Linear Potential (*PLP*) [12, 13] or the more sophisticated *XScore* [14].

Docking using the *PLP* function has not been used for testing the enhancements here since its relatively fast execution makes it harder to observe the benefits of early-rejection. *XScore*, being a more complex function, is a much more appropriate candidate for improvement.

DOX was a commercial docking tool already implemented in C++. To develop and test various editions applying the techniques discussed, preprocessor directives have been used to selectively enable the newly added sections of source code. Several configurations exist, denoted by the codes listed in Table 3.

The scoring function calculations employ lookup tables for speed; these are generated when a receptor is first presented, and saved for reuse. A separate four-dimensional table is calculated for each interaction kind: three dimensions for a ligand atom's position and one for its radius (12 values are considered by *XScore*). Hence, when calculating **Score**(R,L) for a pose, the middle summation of Eq. 1 is eliminated and $\mathbf{F}_i(r, l)$ becomes the lookup $\mathbf{T}_{R,i}(l) = \sum_{r \in R_{\text{atoms}}} \mathbf{F}_i(r, l)$.

Molecular inputs to the system are the receptor's PDB file, the ligand conformations in Symyx SDF format, and a search box SDF molecule (the crystal structure, for these tests) defining the approximate cuboid extents of the pocket region. This box is doubled in each dimension to establish the range of the lookup tables and thus the limits of the search space. Other inputs are the filename for docking results (which are saved in SDF format), the number of results required, and the genetic algorithm parameters such as population size and number of generations.

Ordering and/or rejection

For the scoring function rejection work, atom ordering was implemented by creating a new atom collection class based on one already present in *DOX*. The existing class permitted the use of standard iterator patterns over the vectors of the atoms in a molecule; the modified version simply

Table 3 Configurations of *DOX* to be evaluated

	B	O	R	RL	OR	ORL	N	ORN	ORLN
Atom ordering		•			•	•		•	•
Scoring thresholds			•	•	•	•		•	•
—Limited				•		•			•
Search quotas							•	•	•

added the ability to impose an ordering on this iteration. Only one ordering has been implemented: that of non-increasing distance from the mean of the atomic centres. The search procedures were amended to record when a scoring function terminated early and after which atom the threshold was reached.

Behaviour: single-case tests

Cytidine deaminase was docked with uridine (PDB code 1AF2) many times using different parameters to illustrate the behaviour of this technique. This was intended to show how the parameters affect the method; much broader tests are discussed later.

Figure 3 shows the percentage of all poses scored that were rejected after each ligand atom, both with (case 'OR') and without ('R') the ordering applied. These graphs show the reduction in rejections as the threshold score is relaxed, as well as the smoothing effect of sorting the atoms. Limiting the application of threshold-based rejection to the first eight atoms has the expected effect of flattening these results after their peaks, as seen in cases 'RL' and 'ORL'.

For a range of thresholds, the number of poses evaluated (calls to the scoring function) and total number of atoms considered were measured for docking runs with 1AF2. Figure 4 shows these statistics, which support the expectation that larger thresholds or limited application of early-rejection results in more atoms being surveyed. In general, relaxing the threshold also causes more poses to be scored. Since the GA evaluates a fixed number of poses, this must be explained by considering the local optimization search. This refines a pose within a small range by assuming that the scoring function will be locally smooth. The approximation effect of early-rejection may nullify this assumption, coarsening the function and making it appear to converge sooner than a more detailed evaluation would—and so fewer poses must be scored in the whole process.

Figure 5 compares the run times for each of the thresholds and editions. A clear trend towards longer run times can be seen as the threshold is relaxed, as would be expected from the reduced proportion of cases that will be abandoned. The limited early-rejection methods (dotted lines) are slower than their unlimited counterparts (solid lines), and the unordered method is slightly faster by what appears to be a constant factor in the unlimited case. The limited cases, as they approach a threshold of 2000, level off towards a more constant run time because the number of poses rejected is tending to zero. The unlimited cases would show the same pattern, but at a higher threshold still. The dark horizontal line shows the run time of the non-

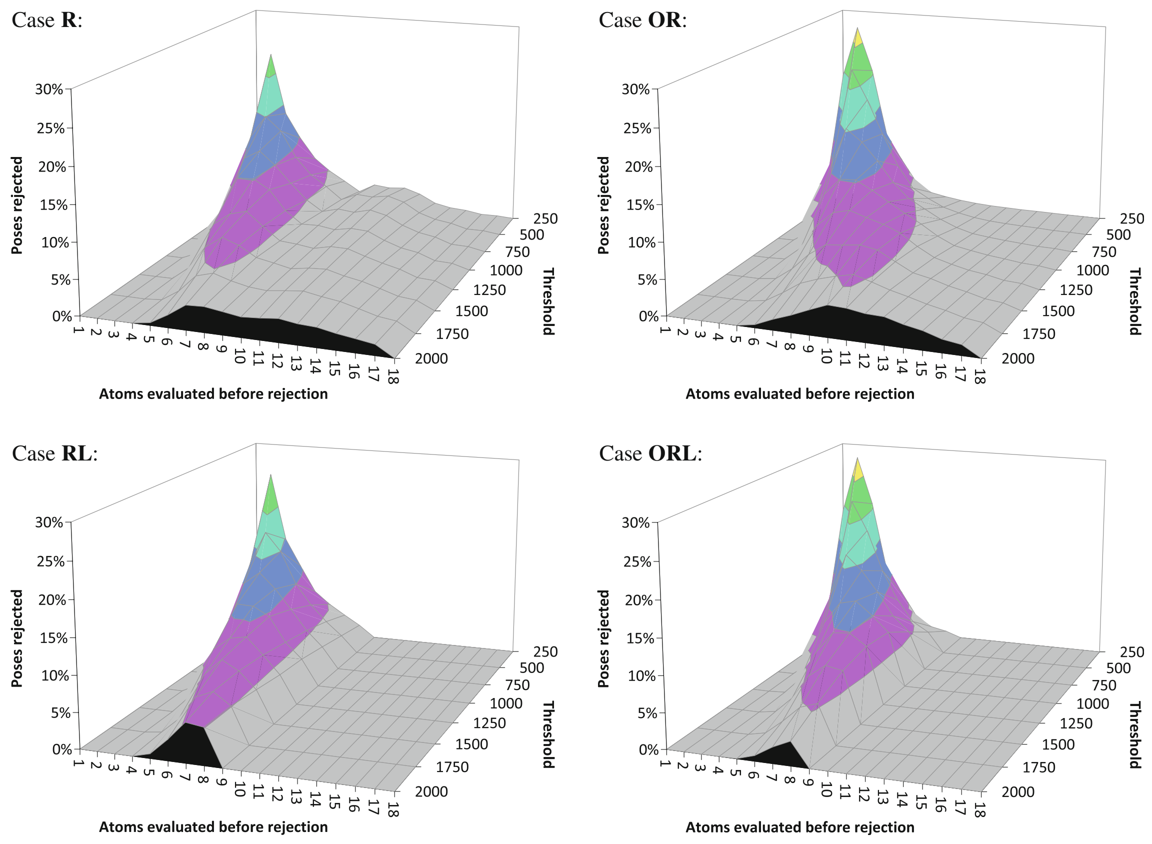

Fig. 3 Effect of scoring threshold on early-rejection when docking 1AF2

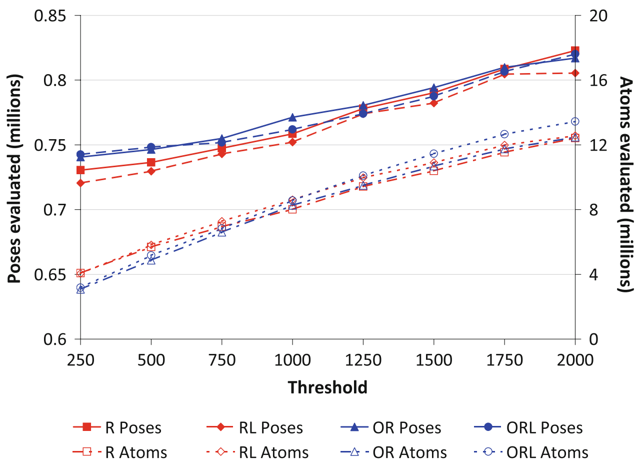

Fig. 4 Effect of scoring threshold on total pose and atom counts

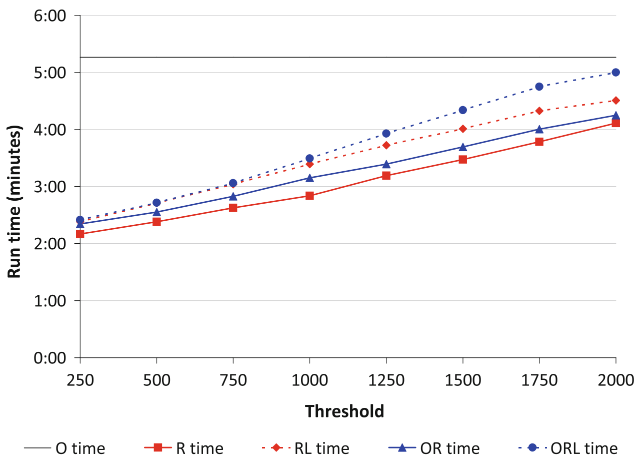

Fig. 5 Effect of scoring threshold on run time

rejecting ‘O’ edition for comparison: the rejecting editions show reductions of up to 59% depending on threshold.

Assessment: many-case tests

Figure 6 compares the effect of three thresholds when applied to the Astex Diverse Set [15] of 85 receptor–ligand complexes. It shows again the gradual rise in the time taken to run the docking procedure as the threshold is relaxed, but also the great reduction in time still when a large docking task is undertaken. It also shows that the ability of the method to select the correct conformation is reduced at low thresholds, but may be retained with good speed if the threshold is increased.

Quotas

The main loop in *DOX*, which iterates through the ligand conformations, was modified to check whether the result container is full and, if so, report the worst score recorded to the genetic algorithm. That code, in turn, had a test inserted at the end of each generation to terminate the procedure if no genome in the population has scored as well as the worst case, provided at least half the generations have been completed. If a conformation was rejected in this manner, the subsequent local optimization stage would also be skipped; hence, rejecting a conformation after the final generation still reduces the work to be done. To explore this method, an ensemble of 100 conformations plus the crystal structure were redocked to tryptophan synthase (1K3U) to see whether the right conformation was still returned in the final result list. Three different quotas were tested: 100, 300, and 500.

Figure 7 shows the run time and number of results returned (in a set of 100) which were the correct conformation for 11 versions of the input database; each input set had the crystal structure of the ligand at a different point in the list. Note that the population size was 96, and thus for any quota q the first $\lceil q/96 \rceil$ input conformations must be accepted. The graph shows that decreasing the quota does speed up the docking process, as expected. The quota of 100 is consistently faster than one of 500, although there is fluctuation between input examples because of the

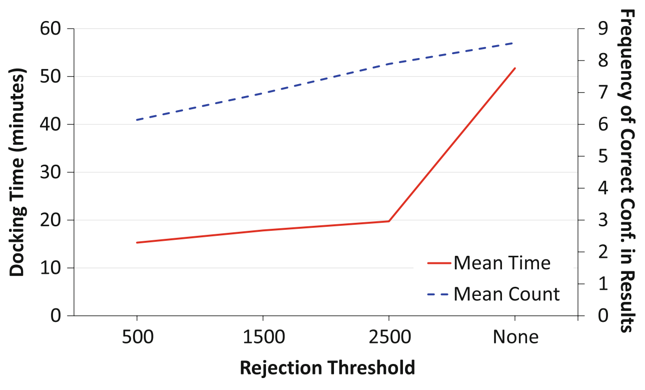

Fig. 6 Scoring thresholds evaluated with 85 complexes (‘OR’ case)

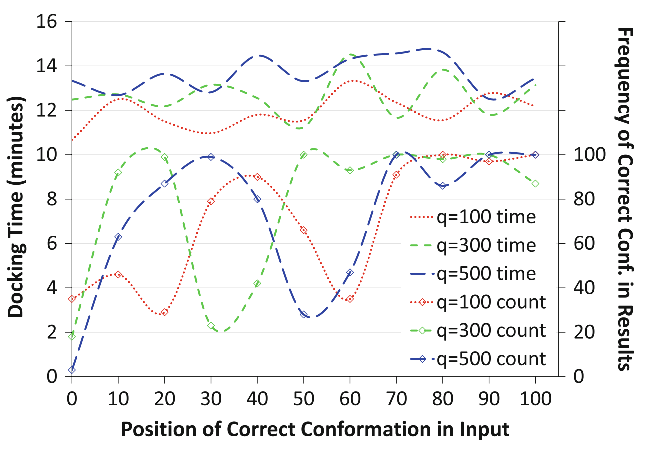

Fig. 7 Effect of ligand’s input index on quota-based rejection

unpredictability of both the worst-case score set for each conformation and whether it will reach it. There is, desirably, no significant trend based on the input database ordering.

However, the frequency of the correct conformation in the results appears to be much more influenced by its rank in the input than by the quota. Counterintuitively, the correct conformation was selected much more reliably when it was amongst the last to be considered, when the minimum score required to avoid rejection would be greatest. This would suggest that a well-suited ligand is likely to be recognized as such swiftly enough to avoid being discarded in most situations, but when it is one of the first to be added to the result list it can easily be 'diluted' with other cases that meet the more lenient worst-case cut-off (although this is not at all certain to happen). Although the first few entries are always guaranteed to be allowed through to the results, this slight preference is apparently not especially substantial.

Comparison

Each of the *DOX* editions listed in Table 3 was used to re-dock each of the 85 complexes in the Astex Diverse Set [15] with the *XScore* function. Input conformation databases were generated using the FROG server [16], 9,596 in total, and the crystal structure was present in each set. Table 4 shows the PDB codes used and the numbers of conformations docked for each. The docking results included RMSD values calculated from the native pose and the highest ranking of the crystal conformation in the output. Those editions featuring threshold-based scoring function early-rejection (with codes containing an 'R') were run using a threshold of 1500; if the threshold was limited ('L') then it was to the first 8 atoms. Those with quota-based conformation rejection ('N') used a quota of 300, and all editions returned 100 poses per execution. All the tests were run using a 2.2 gigahertz Athlon64 3500+ processor (in 32 bit mode), with 1 gigabyte of memory, and the genetic algorithm evolved a population of 96 for 240 generations. Timing values to the nearest second were recorded in the program's output (around 0.25% accuracy at worst). These account only for the conformation docking stage of execution, not the loading of data or writing of output files. The results were averaged over all the complexes, and these statistics are shown in Table 5.

The first observation to make is that, as intended, all the early-rejection techniques discussed reduced the run time for the docking system. Figure 8 compares the timings graphically. The only edition that increased the time taken to process the input database was the 'O' case, which does not implement any rejection schemes but used additional time to sort the ligands' atoms. The use of a scoring

Table 4 Astex Diverse Set test cases used for comparison of editions

			Time per conf. (s)			Freq. of correct		
PDB	HAC	#Conf	B	OR	N	B	OR	N
1g9v	25	51	23.9	10.1	11.0	2	0	3
1gkc	22	201	20.0	7.7	6.9	0	0	10
1gm8	24	201	22.0	8.2	7.8	2	6	17
1gpk	18	3	15.7	7.0	17.3	29	16	27
1hnn	14	3	11.7	5.7	13.0	31	26	31
1hp0	19	49	15.2	6.8	5.8	2	0	26
1hq2	14	4	11.8	5.8	11.8	4	46	33
1hvy	32	201	42.2	13.1	14.3	1	0	15
1hwi	30	279	31.5	11.4	12.2	0	0	8
1hww	12	9	9.2	4.1	6.2	21	31	44
1ia1	19	5	13.6	6.6	14.2	11	16	27
1ig3	18	27	15.8	6.8	7.8	4	0	5
1j3j	16	3	16.7	7.3	18.7	24	30	21
1jd0	13	13	13.5	5.5	12.6	4	13	10
1jje	25	201	20.6	7.9	8.2	3	1	2
1jla	27	51	26.6	8.2	11.6	0	9	9
1k3u	21	101	19.7	7.4	8.3	7	1	9
1ke5	23	281	21.2	8.2	7.0	0	0	2
1kzk	41	401	42.3	12.4	12.1	1	0	5
1l2s	18	101	16.5	7.4	6.6	7	3	20
1l7f	23	210	17.8	7.9	7.3	1	0	3
1lpz	30	51	31.0	9.7	13.8	1	3	5
1lrh	14	7	10.6	5.1	10.0	38	16	31
1m2z	28	65	24.0	8.8	10.5	2	0	5
1meh	23	101	20.1	8.6	8.5	0	0	13
1mmv	15	401	16.5	7.0	5.9	1	0	7
1mzc	35	401	37.3	11.3	12.9	0	0	8
1n1m	12	36	10.8	5.6	4.7	6	6	36
1n2j	10	25	7.3	4.3	4.1	1	0	12
1n2v	15	28	11.9	6.0	5.5	0	6	24
1n46	27	8	26.1	8.3	21.3	8	6	31
1nav	21	39	18.9	8.1	8.2	1	0	5
1of1	18	37	18.9	7.9	8.5	11	6	4
1of6	13	37	11.2	5.9	5.1	0	0	11
1opk	27	72	28.6	8.9	10.8	7	0	20
1oq5	26	25	23.0	8.9	13.3	9	0	0
1owe	22	17	18.4	7.6	8.8	8	1	15
1oyt	30	70	27.7	9.2	10.0	2	0	19
1p2y	12	7	8.6	5.0	7.9	28	18	30
1p62	18	49	13.4	6.6	5.2	1	0	5
1pmn	31	201	35.8	11.4	11.1	1	0	1
1q1g	20	74	18.5	8.2	7.5	4	1	7
1q41	21	1	16.0	7.0	13.0	96	96	96
1q4g	17	25	15.3	7.1	7.8	0	0	3
1r1h	29	401	36.8	12.2	12.4	1	0	2
1r55	23	401	19.6	7.9	6.9	1	1	4
1r58	22	201	26.7	8.7	9.7	0	0	7

Table 4 continued

PDB	HAC	#Conf	Time per conf. (s)			Freq. of correct		
			B	OR	N	B	OR	N
1r9o	18	25	14.4	7.0	7.1	1	1	35
1s19	30	218	26.7	9.3	9.1	0	0	0
1s3v	27	191	24.8	9.0	8.2	1	0	14
1sg0	17	9	14.3	7.1	9.3	31	20	36
1sj0	33	201	31.7	10.4	10.4	2	2	34
1sq5	15	101	13.8	6.7	5.6	0	0	1
1sqn	22	3	17.7	7.3	17.3	45	21	68
1t40	28	51	26.3	9.3	9.8	2	0	19
1t46	37	101	39.9	12.0	14.2	1	0	18
1t9b	22	136	23.3	8.8	8.6	2	0	12
1tow	19	51	13.3	6.5	6.1	3	0	4
1tt1	15	71	10.5	5.9	4.4	0	0	0
1tz8	20	72	16.1	7.6	6.7	5	1	8
1u1c	20	51	18.6	7.7	8.6	4	0	8
1u4d	18	3	15.3	7.0	16.3	29	30	28
1uml	33	101	34.6	11.7	12.1	4	2	8
1unl	26	401	30.4	9.7	10.3	0	0	3
1uou	15	5	11.6	5.8	11.2	6	15	8
1v0p	29	401	35.2	10.8	12.2	1	0	9
1v48	22	51	19.1	7.8	8.5	3	9	0
1v4s	23	31	19.9	7.8	8.9	0	4	28
1vcj	25	201	23.1	8.7	8.6	0	0	0
1w1p	11	2	8.5	4.0	9.5	27	69	29
1w2g	17	49	13.0	6.4	5.6	4	0	25
1x8x	13	37	10.2	5.2	4.6	6	6	21
1xm6	19	138	17.1	7.7	6.2	3	0	1
1xoq	24	51	26.4	9.8	9.7	4	0	12
1xoz	29	7	22.6	8.0	20.1	14	6	33
1y6b	34	201	36.0	11.0	11.0	0	0	0
1ygc	38	201	42.4	12.6	13.2	0	0	74
1yqy	23	401	22.8	8.3	8.1	0	0	1
1yv3	22	3	19.3	6.3	22.7	35	49	34
1yvf	27	101	27.2	9.7	10.6	0	0	2
1ywr	35	401	35.5	12.0	11.6	0	0	8
1z95	29	101	25.1	8.9	9.0	0	0	0
2bm2	30	101	33.9	11.5	12.8	0	0	21
2br1	29	101	27.0	9.2	10.3	1	0	11
2bsm	26	51	22.5	9.1	12.0	3	0	8
Averages			21.5	8.2	10.1	7.3	7.0	15.9

HAC heavy atom count; *#Conf* number of conformations in input database

threshold completed the dockings in around 35% of the original (‘B’) time, with the same pattern as seen in Fig. 5: limited use of rejection is slower than unlimited, and the application of atom ordering adds a small time cost. The quota-based rejection method steps down again to around 36% of the original run time. Combining this with threshold consideration speeds up the execution further to 13%; the fact that this is approximately 36% of 35% corroborating the independence of the two methods. Since it would seem that limited scoring function rejection is counter-productive, at least temporally, those cases might be disregarded. In that case, a reduction in time by more than a half can be claimed.

It is not good enough that the modifications result in a quicker computational system. It needs to remain capable of returning good docked poses. Figure 9 compares the proportion of results from each edition within four bands of closeness to the natural bound state, while Fig. 10 shows the ranking and frequency of the crystal structure conformation in the output. Editions ‘B’ and ‘O’ of the system should produce equivalent results since neither ever rejects any cases and, as discussed, the reordering of atomic contributions to a score should be mathematically insignificant. This prediction is supported by the data, allowing for a certain amount of variation as must be expected from such a stochastic search mechanism as a genetic algorithm. The ratio of results within 2 Å does vary a little, with the use of scoring thresholds damaging it slightly. However, referring to Fig. 10, it is reassuring to note that, even if the docking procedure is a little less precise in its placement of ligands, it is still able to select the correct conformation successfully.

The use of quotas also seems to improve the result quality. Figure 10 indicates that the correct conformation was more prevalent and ranked much higher in the final result set using the quota system. In these tests, the correct conformations were the first presented to the docking procedure in each case, so the bias due to its input ranking was effectively minimized. Applying thresholds to scoring function calculation in combination with search quotas reduced the ranking of the correct conformation back to the previous level, though, and slightly dropped their proportion in the results. This suggests that tracking the worst-recorded result score against partially evaluated search cases approximates too much, blurs the assessment of whether a conformation is ‘promising’ enough to pursue, and thus relaxes the rejection behaviour.

Conclusions

This paper has presented two stratagems for effectively and efficiently evaluating scoring function-based receptor-ligand docking processes.

The first—threshold-based function approximation—has proved beneficial by reducing the time taken to dock a database of ligand conformations by a significant amount. Since this uses an adjustable parameter, it may be used to

Table 5 Average result statistics from test executions of all editions on entire Astex set

	B	O	R	RL	OR	ORL	N	ORN	ORLN
Ligand docking only									
Run time (min)	51:05	51:44	18:04	17:49	30:19	23:04	18:11	6:36	7:37
Improvement		–1%	65%	65%	41%	55%	64%	87%	85%
Top-ranked result									
RMSD (Å)	5.2	5.3	5.5	5.7	5.3	5.6	4.9	6.1	5.5
Correct conformation									
Highest rank	16.0	19.0	19.4	16.0	16.1	19.5	8.9	18.4	15.0
Lowest rank	65.3	71.9	73.6	75.0	76.0	68.8	80.2	80.3	81.7
Appearances	7.3	8.6	7.4	7.0	7.8	7.6	15.9	13.8	11.7
RMSD of highest (Å)	3.3	4.0	3.6	4.8	4.4	4.1	4.5	5.9	5.3

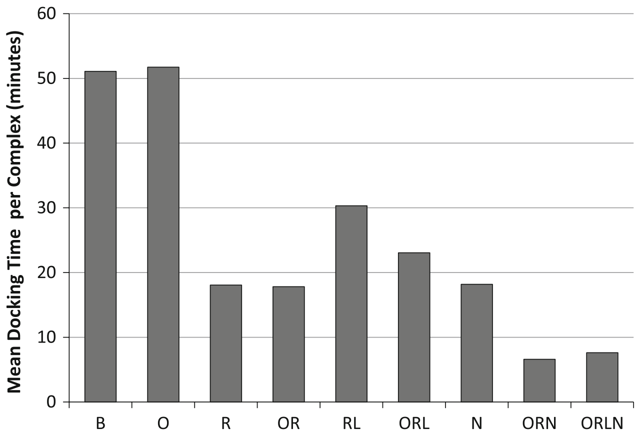

Fig. 8 Run time comparison of all editions

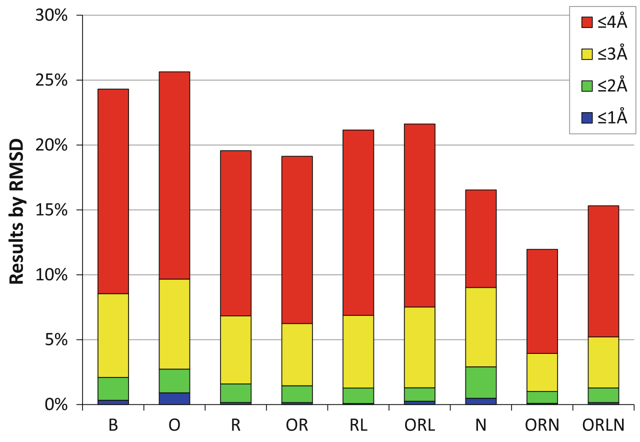

Fig. 9 Result quality comparison of all editions

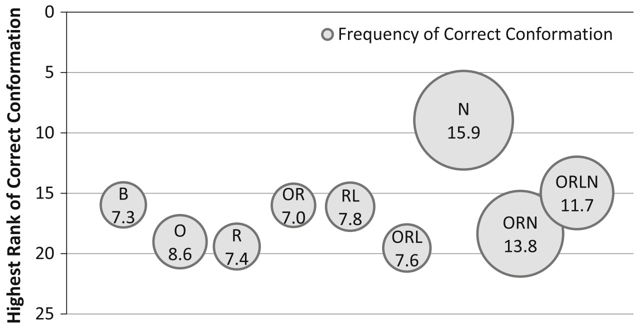

Fig. 10 Conformation ranking comparison of all editions (100 results)

trade greater speed for marginally less reliable results when precise pose placements are less important than swiftly filtering a preliminary compound database. In the multi-stage nature of many programs, this may be a convenient facility to have. The auxiliary technique of prioritizing ligand atoms by their importance for binding is not in itself harmful, but would be better done as a normalization on the input database, and not dynamically for each ligand as part of the docking procedure. The corresponding constant time cost seen in Fig. 5 would then be eliminated, provided no assumptions made by other parts of a wider screening architecture are invalidated by the pre-processing.

The second stratagem—quota-based ligand rejection—also offers a great speed gain without a substantial deterioration in placement. It is not so useful if threshold rejection is applied, but can still help to improve the selectivity of a docking process. From observing this methodology, the input conformation ensemble ordering is seen to bias slightly the make-up of a results set. Whilst it is not necessarily a feature of the quota scheme, this observation may have an application if some intelligent prioritization of ligands can be applied. Work is already in progress to investigate the use of shape and other molecular properties in this manner.

The foundation of this work, inspired by the lazy evaluation paradigm from computer science, is that calculations that make no contribution to the overall output of a system should be avoided. The practical implementations of this principle have been carefully framed to be generic enough that they might be reused in a wide range of software tools and with many scoring functions. This work continues to be investigated as part of a wider examination of strategies for optimizing docking programs using contributions from abstract computer science.

Acknowledgements This work was funded by InhibOx Ltd. (http://www.inhibox.com, 2009). The authors thank Daniel Robinson, Romesh Ranawana, and Garrett Morris for several helpful conversations there, and for the use of their source code as a foundation project.

References

1. Kitchen DB, Decornez H, Furr JR, Bajorath J et al (2004) Docking and scoring in virtual screening for drug discovery: methods and applications. Nat Rev Drug Discov 3:935–949
2. McGaughey GB, Sheridan RP, Bayly CI, Culberson JC, Kreatsoulas C, Lindsley S, Maiorov V, Truchon J-F, Cornell WD et al (2007) Comparison of topological, shape, and docking methods in virtual screening. J Chem Inf Model 47:1504–1519
3. Warren GL, Webster Andrews C, Capelli A-M, Clarke B, LaLonde J, Lambert MH, Lindvall M, Nevins N, Semus SF, Senger S, Tedesco G, Wall ID, Woolven JM, Peishoff CE, Head MS et al (2006) A critical assessment of docking programs and scoring functions. J Med Chem 49:5912–5931
4. Wang R, Lu Y, Wang S et al (2003) Comparative evaluation of 11 scoring functions for molecular docking. J Med Chem 46:2287–2303
5. Böhm H-J (1994) The development of a simple empirical scoring function to estimate the binding constant for a protein–ligand complex of known three-dimensional structure. J Comput Aided Mol Des 8:243–256
6. Taylor RD, Jewsbury PJ, Essex JW et al (2002) A review of protein–small molecule docking methods. J Comput Aided Mol Des 16:151–166
7. Campbell SJ, Gold ND, Jackson RM, Westheady DR et al (2003) Ligand binding: functional site location, similarity and docking. Curr Opin Struct Biol 13:389–395
8. Halperin I, Ma B, Wolfson H, Nussinov R et al (2002) Principles of docking: an overview of search algorithms and a guide to scoring functions. Proteins 47:409–443
9. Hart WE (1994) Adaptive global optimization with local search. PhD thesis, University of California, San Diego
10. Hart WE, Kammeyer TE, Belew RK et al (1994) The role of development in genetic algorithms. Foundations of genetic algorithms III, pp 315–332. Morgan Kauffman, San Mateo, CA
11. Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7:308–313
12. Gehlhaar DK, Verkhivker GM, Rejto PA, Sherman CJ, Fogel DB, Fogel LJ, Freer ST et al (1995) Molecular recognition of the inhibitor AC-1343 by HIV-1 protease: conformationally flexible docking by evolutionary programming. Chem Biol 2:317–324
13. Böhm H-J, Stahl M (2000) Structure-based library design: molecular modelling merges with combinatorial chemistry. Curr Opin Chem Biol 4:283–286
14. Wang R, Lai L, Wang S et al (2002) Further development and validation of empirical scoring functions for structure-based binding affinity prediction. J Comput Aided Mol Des 16:11–26
15. Hartshorn MJ, Verdonk ML, Chessari G, Brewerton SC, Mooij WTM, Mortenson PN, Murray CW et al (2007) Diverse, high-quality test set for the validation of protein-ligand docking performance. J Med Chem 50:726–741
16. Bohme Leite T, Gomes D, Miteva MA, Chomilier J, Villoutreix BO, Tufféry P et al (2007) Frog: a FRee Online druG 3D conformation generator. Nucleic Acids Res 35:W568–W572