INTerface Builder: a fast protein-protein interface reconstruction tool

Chloé Dequeker, Elodie Laine, Alessandra Carbone

JOURNAL OF CHEMICAL INFORMATION AND MODELING

Application Note

INTerface Builder: a fast protein-protein interface reconstruction tool

Chloé Dequeker, Elodie Laine, and Alessandra Carbone

J. Chem. Inf. Model., **Just Accepted Manuscript** • DOI: 10.1021/acs.jcim.7b00360 • Publication Date (Web): 09 Oct 2017

Downloaded from http://pubs.acs.org on October 11, 2017

INTerface Builder: a Fast Protein-Protein Interface Reconstruction Tool

Chloé Dequeker[1], Elodie Laine[1] and Alessandra Carbone[1,2*]

[1] *Sorbonne Universités, UPMC-Univ P6, CNRS, IBPS, Laboratoire de Biologie Computationnelle et Quantitative - UMR 7238, 4 place Jussieu, 75005 Paris, France*

[2] *Institut Universitaire de France, Paris 75005, France*

E-mail: chloe.dequeker@upmc.fr,elodie.laine@upmc.fr,alessandra.carbone@lip6.fr

Abstract

Summary: INTerface Builder (INTBuilder) is a fast easy-to-use software to compute protein-protein interfaces. It is designed to retrieve interfaces from molecular docking software outputs in an empirically determined linear complexity. INTBuilder directly reads the output formats of popular docking programs like ATTRACT, HEX, MAXDo and ZDOCK, as well as a more generic format and Protein Data Bank (PDB) files. It identifies interacting surfaces at both residue and atom resolutions.

Availability and implementation: INTerface Builder is an open source software written in C and freely available for non-commercial use (CeCILL licence) at `https://www.lcqb.upmc.fr/INTBuilder`.

Contact: chloe.dequeker@upmc.fr, elodie.laine@upmc.fr or alessandra.carbone@lip6.fr

Introduction

Protein-protein interactions (PPIs) are essential to all biological processes and their mis-regulation is associated to many human diseases[1,2]. Targeting PPIs with small molecule drugs has become increasingly popular in the treatment of diseases[3–6]. Hence, it is important to determine which protein interacts with which one in the cell and in what manner.

The increasing amount of computing resources and the development of efficient molecular docking algorithms[7–9] have made possible large-scale studies of PPIs, where tens to thousands of proteins are docked to each other[8,10,11]. These cross-docking calculations generate millions of conformations that must be screened in order to extract pertinent information. Several types of analysis can be performed, among which the calculation of the residues propensity to be found at the interface in the docking poses. This property can be exploited toward protein binding sites[8,11,12] and functions[13] prediction. Also, docking interfaces can be analysed to select those that resemble the most known or predicted protein interfaces toward the identification of the cellular partners[8,10,11]. Both types of analysis require the fast and accurate detection of interacting residues in the docking conformations.

State-of-the-art approaches identify interacting residues based on inter-atomic distances, changes in residue Solvent Accessible Surface Area (SASA) upon binding[14] or a Voronoi model of the interface[15]. These methods suffer issues stemming from the large amount of data they need to handle. The first one is the speed of their algorithm. Since the number of conformations can go up to several millions, the algorithm used should be both fast and accurate in its computation of the interface. On the one hand, approaches based on grid-boxing or zoning[16,17] efficiently detect interactions between particles based on a distance criterion in linear complexity. On the other hand, Voronoi model provides a more detailed description of the interface at the expense of more computation time. Another bottleneck is the input/output (I/O) required. To be able to analyse docking ensembles with current tools, one has to write and read the PDB file corresponding to each docking pose before actually computing the interface with the various software available today, the whole process resulting in a very high I/O.

Both issues are crucial to the analysis of large docking ensembles. To specifically address them, we have developed INTerface Builder (INTBuilder), which combines a new, efficient algorithm with the ability to directly read the output of rigid-body docking software. Indeed, the algorithm of INTBuilder (detailed below) can achieve a complexity of $\mathcal{O}(n)$ by drastically reducing the search space when scanning protein surfaces for interface residue. INTBuilder explicitly considers the description of the docking pose by a scalar and a set of Euler angles representing the translation and rotations to be applied to the ligand relative to the receptor. To facilitate the usage of the rotating feature, the output of several rigid-body docking algorithm (ATTRACT[18], HEX[7], ZDOCK[19] and MAXDo[8]) is directly read with the effect of bypassing the I/O need. This allows INTBuilder to treat millions of conformations in a few hours. Other software (Rosetta[20], GRAMM-X[21]) directly outputs the resulting PDB files corresponding to each conformation, which allows INTBuilder analyse them without performing the rotations.

Although INTBuilder was designed to detect protein-protein interfaces, it can also readily be employed to identify the binding sites of small molecules (chemical compounds) from conformations obtained by virtual screening.

Algorithm

INTBuilder defines interfaces as sets of atoms or of residues, depending on the chosen scale, that are close to each other in a protein complex. It uses only one parameter (customisable by the user), that is the threshold distance under which two particles (residues or atoms) will be considered as interacting; we refer to this distance as d_{thresh}. A naive algorithmic approach would be to consider the two sets of particles $\mathcal{P}_1$ and $\mathcal{P}_2$ of each partner respectively and compute all the inter-atomic distances, thus leading to an $\mathcal{O}(n^2)$ complexity, n being the number of particles.

The idea behind the INTBuilder algorithm is to reduce the search space of particles before actually computing the inter-atomic distances (Fig. 1 and Algorithm 1). To do so, INTBuilder first selects the geometric center p_I of the ensemble of particles from the partner 1, $\mathcal{P}_1$. It then selects the farthest particle from it among of the ensemble of particles for the partner 2, $\mathcal{P}_2$, and name it p_I. From p_I, it computes the minimum distance to any particle belonging to $\mathcal{P}_1$ and subtracts to it d_{thresh}. We call the result of this subtraction d_{cut}. Any particle of $\mathcal{P}_2$ that is strictly closer to p_I than d_{cut} is removed from $\mathcal{P}_2$. Next, the algorithm selects the farthest particle of $\mathcal{P}_1$ from p_I, names it p_I in turn and operates the same process. These steps are looped over while at least one particle has been removed with each iteration. The second step of the algorithm simply consists in computing all inter-atomic distances between the remaining candidate particles. We define two sets $\mathcal{I}_1$ and $\mathcal{I}_2$ representing interface particles of partner 1 and partner 2 respectively. As such, any pair of particles from partner 1 and partner 2 are added to $\mathcal{I}_1$ and $\mathcal{I}_2$ respectively if they are separated by a distance lower than d_{thresh}. To ascertain that the algorithm does not erroneously remove any interface particle, we reason as follows.

We want to show that at each iteration (cycle `do` at line 4 in Algorithm 1), INT-Builder reduces the number of particles in $\mathcal{P}_1, \mathcal{P}_2$ while keeping those lying at the interface. We denote $d_{i,j}$ the distance between particles p_i and p_j.

Each iteration comprises two "internal iterations" (cycles `for` at line 8 and 16 in Algorithm 1), the first eliminating some particles in $\mathcal{P}_2$ and the second in $\mathcal{P}_1$. At the beginning of each internal iteration, INTBuilder defines a particle p_I (lines 6 and 14 in Algo 1). At the first iterative step, INTBuilder takes, as p_I, the farthest particle of the partner 2 from the center of mass of the partner 1.

If p_I belongs to the interface, notice that $min\{d_{I,j} - d_{thresh} \,|\, p_j \in \mathcal{P}_2\} < 0$ by definition. This implies that no particles' deletion will be realised by INTBuilder at the first internal iteration step, and the algorithm will go on by considering the particle in $\mathcal{P}_1$ that is most distant from p_I and will take this particle to be the new p_I.

If p_I does not belong to the interface, then let p_o be any particle of $\mathcal{P}_2$ belonging to the interface. We want to prove that p_o cannot be removed by INTBuilder. INTBuilder chooses a particle $p_m \in \mathcal{P}_1$ that is the closest to p_I. Then, it removes from $\mathcal{P}_2$ all particles p_j satisfying the equation:

$$d_{I,j} < d_{I,m} - d_{thresh} \tag{1}$$

Since p_o belongs to the interface of partner 2, by definition of particles at the interface, there is a particle $p_k \in \mathcal{P}_1$ belonging to the interface of partner 1 such as $d_{o,k} \leq d_{thresh}$. In order to show that p_o does not satisfy equation (1), we show:

$$d_{I,o} \geq d_{I,m} - d_{thresh} \tag{2}$$

Notice that $d_{I,m} \leq d_{I,k}$ because of the way p_m was chosen, and since $d_{I,k} \leq d_{I,o} + d_{o,k}$, we have

$$d_{I,m} \leq d_{I,o} + d_{o,k} \tag{3}$$

Since $d_{o,k} \leq d_{thresh}$ then, by (3), we derive $d_{I,m} - d_{thresh} \leq d_{I,o}$, that is (2), as claimed above. To show that particles in the interface are not removed in $\mathcal{P}_1$ by the second internal iteration of the algorithm, we proceed in a similar way.

Although the worst case scenario could theoretically lead the algorithm to a complexity of $\mathcal{O}(n^2)$, that only happens if the whole surface of the protein is interacting (the complexity of INTBuilder is mainly linked with the size of the interacting surface itself more than the size of the protein).

To estimate the empirical complexity of the algorithm, we computed the interfaces of about 50 million complex structure predictions, obtained from a complete cross-docking of 168 proteins[22] using the docking algorithm MAXDo[8]. Overall, we found that the do-while loop (Algorithm 1, lines 4-21) had an average of 5.8 iterations and a maximum number of iterations N_{max} of 23. Thus, the reduction of the search space algorithm is realised in $\mathcal{O}(n \times N_{max})$. Since N_{max} is constant, this step has a time complexity of $\mathcal{O}(n)$. The last part of the INTBuilder algorithm (from line 23 on) computes all the distances between the remaining candidate particles of $\mathcal{P}_1$ and $\mathcal{P}_2$

Algorithm 1 Reducing the search space and pairwise detection

```
1:  let $\mathcal{P}_1$ be the ensemble of particles for the partner 1
2:  let $\mathcal{P}_2$ be the ensemble of particles for the partner 2
3:  compute the geometric center of $\mathcal{P}_1$ and call it $p_I$
4:  do
5:      choose $p_2$ such that $d_{p_2,p_I} \geq d_{p_j,p_I}$ for all $p_j \in \mathcal{P}_2$
6:      let $p_2$ be called $p_I$
7:      compute $d_{cut}$ as $\min(d_{p_I,p_i} - d_{thresh})$ for all $p_i \in \mathcal{P}_1$
8:      for $p_j \in \mathcal{P}_2$ do
9:          if $d_{p_I,p_j} < d_{cut}$ then
10:             remove $p_j$ from $\mathcal{P}_2$
11:         end if
12:     end for
13:     choose $p_1$ such that $d_{p_1,p_I} \geq d_{p_i,p_I}$ for all $p_i \in \mathcal{P}_1$
14:     let $p_1$ be called $p_I$
15:     compute $d_{cut}$ as $\min(d_{p_I,p_j} - d_{thresh})$ for all $p_j \in \mathcal{P}_2$
16:     for $p_i \in \mathcal{P}_1$ do
17:         if $d_{p_I,p_i} < d_{cut}$ then
18:             remove $p_i$ from $\mathcal{P}_1$
19:         end if
20:     end for
21: while at least an element is removed in $\mathcal{P}_1$ or $\mathcal{P}_2$
22:
23: let $\mathcal{I}_1$ be the set of interface particles for the partner 1
24: let $\mathcal{I}_2$ be the set of interface particles for the partner 2
25: for $p_i \in \mathcal{P}_1$ do
26:     for $p_j \in \mathcal{P}_2$ do
27:         if $d_{p_i,p_j} \leq d_{thresh}$ then
28:             add $p_i$ to $\mathcal{I}_1$
29:             add $p_j$ to $\mathcal{I}_2$
30:         end if
31:     end for
32: end for
```

and stores them in $\mathcal{I}_1$ and $\mathcal{I}_2$ respectively if they are in contact with one another. Although the complexity of this last step is $\mathcal{O}(n^2)$, n holds only for roughly a quarter of its original value after the space reduction obtained in the first part of the algorithm (Fig. S1).

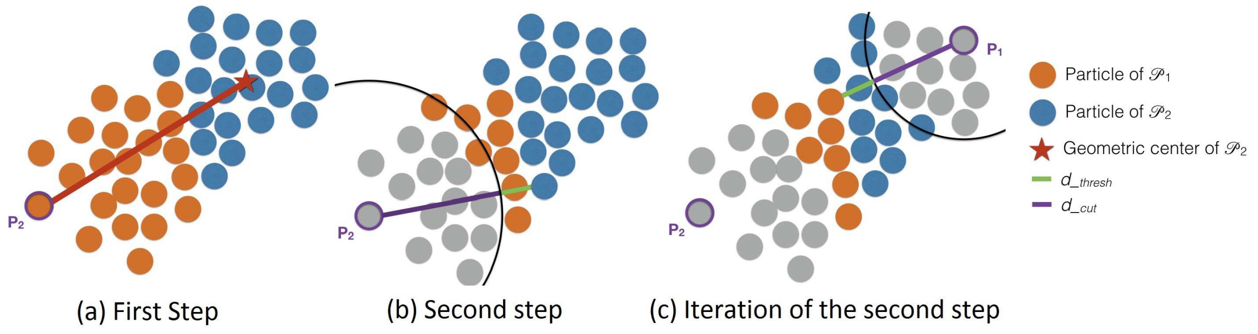

P₂
P₂
P₂
P₁
Particle of $\mathscr{P}_1$
Particle of $\mathscr{P}_2$
Geometric center of $\mathscr{P}_2$
d_thresh
d_cut
(a) First Step
(b) Second step
(c) Iteration of the second step

Figure 1: Scheme of the search space reduction algorithm. **(a)** The geometric center of the blue partner (red star) is chosen as a starting point and the farthest particle p_2 of the orange partner is selected. **(b)** The minimum distance between p_2 and the blue partner is computed and d_{thresh} is subtracted to it to obtain d_{cut}. All the particles closer than d_{cut} (in grey) are removed from the orange partner. **(c)** The particle p_1 of the blue partner that is the farthest from p_2 is chosen and the reduction step is repeated.

Comparison with other methods

INTBuilder is distance-based, and as other similar methods its main challenge consists in reducing the search space before computing all pairwise distances between remaining candidates particles. As INTBuilder, boxing approaches[16] focus on reducing the search space and do so with a complexity of $\mathcal{O}(n)$. An important part of the boxing approaches consists in defining the grid size, which adds another parameter to the program. To the best of our knowledge, no tool is available to specifically detect protein-protein interfaces using a boxing approach. In contrast, INTBuilder has the advantage of its algorithmic simplicity, ease of implementation and of a single defined parameter (threshold distance). Overall, boxing approaches are applied to more general issues (Discrete Element Method, Molecular Dynamics) while INTBuilder focuses on a specific issue. We have measured the computation time required by INTBuilder and a naive approach (computing every inter-atomic distances) and specifically evaluated the computation time of INTBuilder's algorithm compared to the naive approach in Table S1. The results show a decrease of the computation time of the interface determination by a factor from ten to one hundred over the naive algorithm, depending on the size of the protein. INTBuilder's efficiency was also compared with Naccess[23] and the

Voronoi model[15] when computing the interface for a single complex (Table S3). Since we do not read from a docking output, we do not use INTBuilder's perk of bypassing the I/O. This permits us to focus on the algorithm speed itself in its comparison to other software. When looking at several conformations however, INTBuilder's ability to bypass the I/O and allows it to outshine the other software in terms of computation speed. Indeed, both software require to write the PDB file corresponding to each conformation, which proved to be extremely hindering for treating the 50 million conformations of our set. Both tables show that INTBuilder is consistently faster than the other two software, its increase in speed ranging from twenty to more than one hundred times faster. In Table S3, we computed the interface for five hundreds conformations computed with HEX[7] and showed the importance of the I/O ability implemented in INTBuilder (also present in the Naive approach). Naccess and Voronoi give a computation time in the same order of magnitude as the docking time itself. The naive approach, while benefiting from the I/O ability of INTBuilder, also shows its lack of scalability when considering bigger complexes.

We compared the accuracy with which the different methods were able to define interfaces. All three of them yield similar interfaces (Table 1 and Fig. S2). On average, the detected interfaces comprise the same number of particles (atoms or residues), and they share more than 79% of particles in common (Table 1). We further evaluated the impact of the small differences between the interfaces detected by INTBuilder, Naccess and Voronoi (Table 1) on the discrimination of binding partners. We considered the 14 196 possible protein pairs of our dataset of 168 proteins and the goal was to single out the 84 experimentally validated pairs of interactors. The docking interfaces detected by INTBuilder, Naccess and Voronoi were compared to the experimentally known interfaces. For each protein pair, the docking pose with the interface resembling the experimental interface the most was selected, and the overlap between docking and experimental interfaces was used to compute an interaction index for the protein pair. All protein pairs were then ranked based on their interaction indices (see[10] for a detailed

Table 1: Statistical values obtained when comparing INTBuilder with a 5Å distance cut-off to Naccess and Voronoi model. For the INTBuilder-Voronoi comparison, 4 750 938 conformations were treated and the interfaces were detected at the atomic scale. For the INTBuilder-Naccess comparison, 49 192 401 conformations were treated and the interfaces were detected at the residue scale. PPV stands for Positive Predictive Value.

	Atom	Residue
	Voronoi	Naccess
Recall	0.79	0.90
PPV	0.80	0.83
Accuracy	0.98	1.00
Specificity	0.99	1.00
F1-score	0.79	0.86
Naccess/Voronoi average interface size	78	16
INTBuilder average interface size	78	17

description of the protocol). The discrimination power of the approach was estimated by the Area Under the Curve (AUC). The AUC values obtained on the whole dataset and on the different functional classes are very similar between the three detection methods (Table 2). In other words, no significant advantage over INTBuilder could be gained from using another method. These results show that INTBuilder is accurate enough to be used in the context of partner discrimination.

Conclusion

We have presented INTBuilder, a new, easy-to-use and very efficient software which computes the interface between two proteins. The speed of its algorithm comes from a new way to reduce the search space before computing the interacting distances between remaining particles and is able to achieve an $\mathcal{O}(n)$ complexity. INTBuilder itself has been implemented in such a way that it can process millions of different conformations coming from docking software in a limited amount of time. Specifically, it can directly read the output of known rigid-body docking software. This feature allows it to avoid

Table 2: AUC values for the identification of interacting partners in the Protein-Protein Docking Benchmark v2[22]. The complete cross-docking experiment is described in[10]. The AUCs were obtained by using experimental interfaces and docking interfaces computed according to the method described in the column. The dataset is divided into 8 functional classes: Antibody-Antigen (AA), Bound Antibody-Antigen (ABA), Enzyme-Inhibitor (EI), Enzyme-Regulator (ER), Enzyme-Substrate (ES), Other linked to G-protein (OG), Other regulatory (OR) and Other (OX).

	Atom		Residue	
	INTBuilder	Voronoi	INTBuilder	Naccess
AA (20)	0.83	0.84	0.86	0.83
ABA (24)	0.86	0.91	0.92	0.92
EI (38)	0.84	0.88	0.81	0.82
ER (6)	0.78	0.72	0.78	0.74
ES (12)	0.87	0.90	0.83	0.87
OG (24)	0.93	0.95	0.90	0.87
OR (14)	0.81	0.82	0.79	0.87
OX (30)	0.87	0.92	0.88	0.84

any excess of I/O and thus brings a valuable gain of time when considering large set of docking conformations.

The data obtained from the interfaces of large-scale docking calculations can be exploited to identify cellular partners and/or compute propensities of residues to be found at the interface. Although INTBuilder was designed for PPIs, it can also be readily applied to small-molecule docking. The simplicity of INTBuilder's usage makes it a valuable tool to identify the binding sites of small molecules from conformations obtained by virtual screening.

Acknowledgment

We thank Thom Vreven for providing his source code that is used in INTBuilder for reconstructing PDBs from the ZDOCK output files.

Fundings

This work was supported by the MAPPING project (ANR-11-BINF-0003, Excellence Programme "Investissement d'Avenir") (AC), the Institut Universitaire de France (AC), the Ministère de l'Enseignement Supérieur et de la Recherche (CD).

Supporting Information

Figures: search space reduction graph (Figure S1), comparison of performances distributions (Figure S2). List of Tables: benchmark of INTBuilder algorithm versus Naive approach (Table S1), benchmark of INTBuilder versus Naccess and Voronoi (Table S2), benchmark comparison with HEX docking algorithm (Table S3).

References

(1) Betzi, S.; Restouin, A.; Opi, S.; Arold, S. T.; Parrot, I.; Guerlesquin, F.; Morelli, X.; Collette, Y. Protein Protein Interaction Inhibition (2P2I) Combining High Throughput and Virtual Screening: Application to the HIV-1 Nef protein. *Proc. Natl. Acad. Sci. U.S.A.* **2007**, *104*, 19256–19261.

(2) Hakes, L.; Pinney, J. W.; Robertson, D. L.; Lovell, S. C. Protein-Protein Interaction Networks and Biology–What's the Connection? *Nat. Biotechnol.* **2008**, *26*, 69–72.

(3) Arkin, M. R.; Tang, Y.; Wells, J. A. Small-Molecule Inhibitors of Protein-Protein Interactions: Progressing Toward the Reality. *Chem. Biol.* **2014**, *21*, 1102–1114.

(4) Wells, J. A.; McClendon, C. L. Reaching for High-Hanging Fruit in Drug Discovery at Protein-Protein Interfaces. *Nature* **2007**, *450*, 1001–1009.

(5) Hartwell, L. H.; Hopfield, J. J.; Leibler, S.; Murray, A. W. From Molecular to Modular Cell Biology. *Nature* **1999**, *402*, 47–52.

(6) Zhao, L.; Chmielewski, J. Inhibiting Protein-Protein Interactions Using Designed Molecules. *Curr. Opin. Struct. Biol.* **2005**, *15*, 31–34.

(7) Ghoorah, A. W.; Devignes, M. D.; Smail-Tabbone, M.; Ritchie, D. W. Protein Docking Using Case-Based Reasoning. *Proteins* **2013**, *81*, 2150–2158.

(8) Sacquin-Mora, S.; Carbone, A.; Lavery, R. Identification of Protein Interaction Partners and Protein-Protein Interaction Sites. *J. Mol. Biol.* **2008**, *382*, 1276–1289.

(9) Pierce, B. G.; Wiehe, K.; Hwang, H.; Kim, B. H.; Vreven, T.; Weng, Z. ZDOCK Server: Interactive Docking Prediction of Protein-Protein Complexes and Symmetric Multimers. *Bioinformatics* **2014**, *30*, 1771–1773.

(10) Lopes, A.; Sacquin-Mora, S.; Dimitrova, V.; Laine, E.; Ponty, Y.; Carbone, A. Protein-Protein Interactions in a Crowded Environment: an Analysis via Cross-Docking Simulations and Evolutionary Information. *PLoS Comput. Biol.* **2013**, *9*, e1003369.

(11) Laine, E.; Carbone, A. Protein Social Behavior Makes a Stronger Signal for Partner Identification than Surface Geometry. *Proteins* **2017**, *85*, 137–154.

(12) Fernandez-Recio, J.; Totrov, M.; Abagyan, R. Identification of Protein-Protein Interaction Sites From Docking Energy Landscapes. *J. Mol. Biol.* **2004**, *335*, 843–865.

(13) Vamparys, L.; Laurent, B.; Carbone, A.; Sacquin-Mora, S. Great Interactions: How Binding Incorrect Partners Can Teach Us About Protein Recognition and Function. *Proteins* **2016**, *84*, 1408–1421.

(14) Lensink, M. F.; Wodak, S. J. Docking and Scoring Protein Interactions: CAPRI 2009. *Proteins* **2010**, *78*, 3073–3084.

(15) Cazals, F.; Proust, F.; Bahadur, R. P.; Janin, J. Revisiting the Voronoi Description of Protein-Protein Interfaces. *Protein Sci.* **2006**, *15*, 2082–2092.

(16) T. Iwai, C.-W. H.; Greil, P. Fast Particle Pair Detection Algorithm for Particle Simulations. *Int. J. Mod. Phys. C* **1999**, *10*.

(17) Mishra, B. K. A Review of Computer Simulation of Tumbling Mills by the Discrete Element Method: Part IContact Mechanics. *Int. J. Min. Proc.* **2003**, *71*, 73–95.

(18) Zacharias, M. Protein-Protein Docking With a Reduced Protein Model Accounting For Side-Chain Flexibility. *Protein Sci.* **2003**, *12*, 1271–1282.

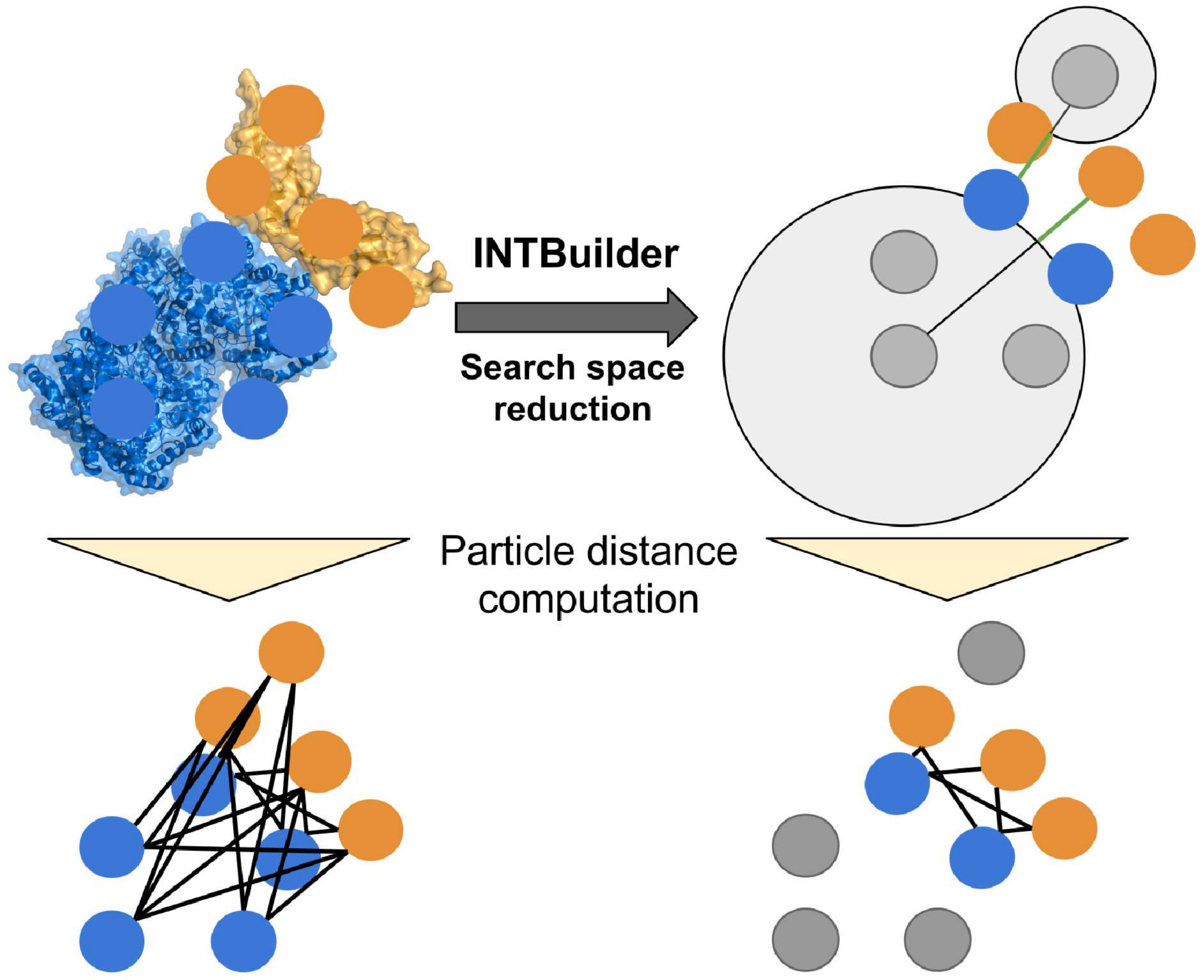
(19) Chen, R.; Weng, Z. Docking Unbound Proteins Using Shape Complementarity, Desolvation, and Electrostatics. *Proteins* **2002**, *47*, 281–294.

(20) Wang, C.; Bradley, P.; Baker, D. Protein-Protein Docking With Backbone Flexibility. *J. Mol. Biol.* **2007**, *373*, 503–519.

(21) Tovchigrechko, A.; Vakser, I. A. GRAMM-X Public Web Server For Protein-Protein Docking. *Nucleic Acids Res.* **2006**, *34*, W310–314.

(22) Mintseris, J.; Wiehe, K.; Pierce, B.; Anderson, R.; Chen, R.; Janin, J.; Weng, Z. Protein-Protein Docking Benchmark 2.0: an Update. *Proteins* **2005**, *60*, 214–216.

(23) Hubbard, S. J.; Thornton, J. M. Naccess. *Computer Program, Department of Biochemistry and Molecular Biology, University College London* **1993**, *2*.


INTBuilder
Search space reduction
Particle distance computation