# Divide-and-conquer strategy for large-scale Eulerian solvent excluded surface 

Rundong Zhao,<br>Department of Computer Science and Engineering, Michigan State University, MI 48824, USA<br>Menglun Wang,<br>Department of Mathematics, Michigan State University, MI 48824, USA<br>Yiying Tong, and<br>Department of Computer Science and Engineering, Michigan State University, MI 48824, USA<br>Guo-Wei Wei<br>Department of Mathematics, and Department of Electrical and Computer Engineering, and Department of Biochemistry and Molecular Biology, Michigan State University, MI 48824, USA


#### Abstract

Motivation: Surface generation and visualization are some of the most important tasks in biomolecular modeling and computation. Eulerian solvent excluded surface (ESES) software provides analytical solvent excluded surface (SES) in the Cartesian grid, which is necessary for simulating many biomolecular electrostatic and ion channel models. However, large biomolecules and/or fine grid resolutions give rise to excessively large memory requirements in ESES construction. We introduce an out-of-core and parallel algorithm to improve the ESES software.

Results: The present approach drastically improves the spatial and temporal efficiency of ESES. The memory footprint and time complexity are analyzed and empirically verified through extensive tests with a large collection of biomolecule examples. Our results show that our algorithm can successfully reduce memory footprint through a straightforward divide-and-conquer strategy to perform the calculation of arbitrarily large proteins on a typical commodity personal computer. On multi-core computers or clusters, our algorithm can reduce the execution time by parallelizing most of the calculation as disjoint subproblems. Various comparisons with the state-of-the-art Cartesian grid based SES calculation were done to validate the present method and show the improved efficiency. This approach makes ESES a robust software for the construction of analytical solvent excluded surfaces.


Availability and implementation: http://weilab.math.msu.edu/ESES.

## 1. Introduction

As a principal tool to study the biomolecular world, molecular modeling and analysis have an increasing impact in computational biology. The accuracy and efficiency of molecular modeling and analysis are often crucial in enabling more sophisticated downstream research.

[^0]Researchers have made persistent efforts in reconstructing and visualizing the details of biomolecules through various simplifications, including the ball-and-stick model by von Hofmann, dated back to 1865 , and the ribbon diagram by Richardson for illustrating protein structures. However, in order to simulate physical phenomena like the electrostatic distribution of macromolecules in a cellular environment, a much more elaborate model is needed to describe the interface between solvent and solute regions. The van der Waals surface (i.e., "atom and bond" model by Corey and Pauling in 1953) was introduced to describe such interfaces, where each type of atoms was described by a sphere with the corresponding van der Waals radius. For various simulations and geometric smoothness, concepts of solvent accessible surface (SAS) [7, 18] and solvent excluded surface (SES) [12, 17] were built on top of the van der Waals radii. SAS captures the trajectory of the center of a probe atom rolling on the van der Waals surface as the interface delineating the boundary of regions accessible by the center of any solvent molecule. SES is defined by the boundary of the union of all possible outside probe balls, and thus consists of three types of patches. Specifically, convex patches, where the probe touches one of the atoms of the molecule, saddle patches, where the probe touches two atoms, and concave patches, where the probe touches three or more atoms, are parts of an SES for a biomolecule.

All of these models still fail to guarantee the interface smoothness, as singularities and sharp edges cannot be completely avoided for the aforementioned geometry models for biomolecules. Minimal molecular surface (MMS) based on the mean curvature flow was introduced to resolve this issue [2, 3]. Various Gaussian surfaces [4, 5, 8, 9, 23, 25], skinning surface [6] and flexibility-rigidity index (FRI) surface [16, 22] have been proposed to achieve a similar goal. Another limitation for these models is that they only reflect the static or instantaneous shape in vacuum. In practice, solvent and solute interactions, making a static interface inaccurate for certain biophysical analysis. Thus, various solvent-solute interactive boundaries were proposed [10, 21]. However, despite its weaknesses, SES remains the most favorable model among biophysicists, due to its simplicity and effectiveness in capturing the interface of solvent and solute through its definition, with which various physical phenomena can be described with a reasonable accuracy.

Many software packages were developed to calculate SES [19]. Among them, MSMS is of considerable influence [20]. Built on top of MSMS, there are various software packages for different purposes. For the Lagrangian representation, a triangle mesh can be directly constructed for the three different types of patches followed by a concatenation. Nevertheless, MSMS is known for its efficiency and robustness issues, which often occur when large protein molecules and fine resolutions are required [14]. Moreover, many biophysical phenomena are happening not only on the surface, but inside the encapsulated volume of the molecules. To address these issues and meet the requirements of volumetric output, Liu et al. [14] introduced an Eulerian solvent excluded surface (ESES) approach as an alternative for surfaces represented as intersections and normals with a regular Cartesian grid. The ESES algorithm starts with a list of atoms describing the molecule enclosed by a regular Cartesian grid. Based on the three different types of patches for SES, all grid points are classified as either inside or outside with respect to SES. Finally, intersection points are computed on each mesh line with two ends on opposite side of the interface. It is also straightforward to be converted into the Lagrangian representation, i.e., a triangle mesh,
through the marching cubes algorithm. Although high accuracy and robustness are well addressed with this method, it often suffers from the lack of efficiency as well as overly large memory requirements, as a full regular grid has to be maintained. The ESES algorithm is sequential, which results in long execution time especially when the grid resolution increases due to a fine grid spacing or large protein complexes with many atoms [13].

In this work, we propose an out-of-core parallelizable version of ESES, in which we divide the bounding box of the molecule into tiled sub-blocks based on the localized nature of the problem. By performing the computation based on local information, one can avoid keeping the whole grid and all the atoms in memory, and at the same time, distribute the computation to multiple processors. Thus, for large molecules or fine grids, both space and time efficiency can be substantially improved. By restricting the active subdomains that are being executed, the whole procedure can always be done on a personal computer (PC) with a fixed memory, e.g., 2GB. Testing and comparison are done on the 2016 core set of PDBbind database (http://www.pdbbind.org.cn/), with additional validation by users worldwide through the authors' website for the project.

The rest of the paper is organized as follows. Section 2 discusses design of the improved algorithm with locality. Section 3 is devoted to the space and time complexity analysis of the proposed algorithm. The validation and comparison of our results are carried out in Section 4. Section 5 concludes the paper with a comment on future work.

## 2. Algorithm

### 2.1. Recap of ESES

As we aim at improving the efficiency of ESES [14], we assume the same input, a list $\boldsymbol{A}$ of atoms, represented by the center location $\boldsymbol{c}_{\boldsymbol{i}}$ and the corresponding van der Waals radius $r_{i}$ for each atom,

$$
\begin{equation*}
\mathscr{A}=\left\{\left(c_{i}, r_{i}\right)\right\}_{i=1 \ldots N}, \tag{1}
\end{equation*}
$$

where $N$ is the number of atoms.
We also assume the same output: first, a 3D array of Boolean indicating whether each grid point is inside the molecule surface,

$$
\text { Inside }[\mathrm{i}, \mathrm{j}, \mathrm{k}]= \begin{cases}1, & (i h, j h, k h) \in M  \tag{2}\\ 0, & \text { otherwise },\end{cases}
$$

where $h$ is the grid spacing, and $M$ is the volume enclosed by SES; and second, a set of intersection points between grid edges and the SES

$$
\begin{equation*}
\mathscr{I}=\left\{s, t, \lambda_{s t}\right\}_{s t \in \mathscr{E}} \tag{3}
\end{equation*}
$$

where $\boldsymbol{s}$ and $\boldsymbol{t}$ are grid points with $\boldsymbol{s} \boldsymbol{t}$ representing the corresponding edge, and Inside $[\boldsymbol{s}] \neq$ Inside $[t]$ is the grid edge set with two grid points adjacent to each other and Inside $[s] \neq$ Inside $[t]$. The location of the intersection point can be computed through

$$
\begin{equation*}
\boldsymbol{p}=\lambda_{\boldsymbol{s} t} s+\left(1-\lambda_{s t}\right) \boldsymbol{t} \tag{4}
\end{equation*}
$$

Note that, using the connectivity construction procedure in the standard marching cubes algorithm, we can also output a triangle mesh based on the Eulerian output.

### 2.2. Overview

The main idea for reducing the main memory requirement is through a simple domain decomposition without the need to explicitly handle the boundary matching problem. Owing to the locality of the ESES algorithm, we can straightforwardly decompose the computational domain into many non-overlapping subdomains. With each subdomain retaining only a small number of atoms that are less than the largest van der Waals radius away from the subdomain boundary, we have all the information necessary to determine the inside and the outside information, as well as the intersections of the SES surface and the grid edges within the subdomain. Thus, we can successfully reduce the memory footprint by controlling the size of each subdomain so as to fit within the main memory limit of a typical PC . The memory storage for the list of atoms relevant for the subdomain is negligible compared to the storage requirement for the subgrid, since the grid spacing in practical applications would typically be smaller than the van der Waals radius of the smallest atom.

As shown in Figure 1, patches rendered with different color belong to different subdomains, and they can be independently constructed by intersection detection locally within the corresponding subdomain, followed by using the marching cubes algorithm. With a direct concatenation of all the output, we can construct the whole molecular surface. It is also possible for the downstream applications to choose only the subdomains relevant for the calculation that they perform.

When designing a parallelizable out-of-core algorithm, the first and foremost problem to deal with is to analyze the dependence among different steps of the procedure or different parts of the data. In this section, we examine the four main stages of the ESES algorithm [14]:

- construction of the grid and the analytical expression for patches of the SES,
- lassification of the grid points to the inside points or the outside points of the SES,
- calculation of the intersection between grid edges and the SES,
- and assembly of the output.

When performed in a subdomain of the entire domain, the first three steps need to performed in the given order, but they do not have data dependence to the calculation done on any other subdomain. For instance, the classification of any grid point can be locally determined by
the nearby atoms, more precisely, atoms at a distance less than the sum of the probe radius and the largest van der Waals radius. Similarly, where the intersection is on a grid edge depends only on the SES analytical patch expressions determined by nearby atoms. Thus we can set up one thread per subdomain, without any possibility of race conditions, i.e., our output is independent on the timing of each thread.

If file I/O exchanges are done through sequential devices such as a hard drive, the final output step would have to be done after finishing the previous three steps. On the other hand, if the final output is to reside in a random access memory, once the size of output for each subdomain is determined, it is possible to assemble the output in sublinear time. If the file system allows for concatenation without moving data blocks, it is also possible for each thread to write to a different file, and concatenate them in a time linear manner with respect to the number of subdomains.

As in ESES, for the robustness of the calculation, there are some grid points left undetermined as inside or outside in the second step, and only finalized in the third step after the intersections of nearby grid edges are determined [14]. Nevertheless, this procedure will only have a local data dependence. Thus, partitioning the whole grid into several subdomains does not change the final classification of such points. As confirmed by our large set of test results, the uncertain grids are rare as indicated by [14], and when they indeed exist, their classification in the subdomain based approach is identical to that in the original ESES.

In sum, we can safely assume that there is no communication of information between intermediate results from different subdomains. This implies that we can do out-of-core calculation by loading only one subdomain, and/or parallelizing the calculation by simultaneously initiating one thread per subdomain.

### 2.3. Decomposition to subdomains

The entire calculation domain in ESES is a regular Cartesian grid inside a cuboid. Typically, it is constructed as a tight bounding box of the list of the atoms, padded with a few additional layers of grid cells to provide some margin for easy handling of the boundary cells.

Therefore, it is a natural choice to design the subdomain as non-overlapping cubes with the same number of grid cells in each of the three dimensions. The domain can be extended slightly if the size of the original entire domain in any direction is not a multiple of the size of the subdomain. We will call a cubic subdomain as a block, following the similar term used in CUDA parallel thread mapping design.

By focusing on the local computation within each block, the memory footprint is mainly determined by the size of the block, since we only need to keep one block in the main memory at a time. Some memory storage is required to store the list of atoms relevant for the analytical SES patch construction. If we store the entire list of atoms, for large molecules, it can still require a large block of memory, and may require more time to perform the nearest neighbor search. Fortunately, due to the localized nature of the
calculation, it is possible to determine whether an atom is inside or within a small distance from a block. The rest of the atoms do not need to reside in the memory, and one can treat the calculation inside the block as if it were for a smaller molecule, without any unfavorable effect on the final output.

### 2.4. Index mapping

After the computation is done for each individual block, we need to map the result within that block to the original domain. The index mapping is similar to the CUDA parallel thread mapping design for 3D. We illustrate the basic idea in 2D through Figure 2. Each block has coordinates $\left(b_{x}, b_{y}, b_{z}\right)$ indicating the position of its top left back corner. Assuming that the grid cell count along one edge of each block is $\left(b_{s}\right)$, for a grid point with local coordinates ( $i$, $j, k)$ in the block, its corresponding global coordinates are found through the following function

$$
\begin{equation*}
\operatorname{LocalToGlobal}\left(i, j, k ; b_{x}, b_{y}, b_{z}\right)=\left(b_{s} b_{x}+i, b_{s} b_{y}+j, b_{s} b_{z}+k\right) \tag{5}
\end{equation*}
$$

### 2.5. Subdomain boundary treatment

While the subdomains are not overlapping with the outer boundary of the total computational domain, they may intersect at a zero-measure set, such as a common rectangle, a common line segment, or a common point. Thus, during the final assembly of the output from all the blocks, the boundary grid points and boundary grid edges are to be carefully handled. Otherwise, redundant information for grid point classification and intersection points on subdomain boundaries may appear in the output. For instance, whether a grid point is inside may be duplicated up to 8 times, if it is at the corner point shared by 8 subdomains. Similarly, the intersection information may also be duplicated up to 4 times, if a grid edge is shared by 4 subdomains.

One way to eliminate the redundancy is to add a post-processing step. However, using ideas commonly used for eliminating such redundancies, we can directly avoid the generation of redundant data. This more efficient approach is based on the partitioning of the domain into truly disjoint subdomains, each of which has the form of a half-open half-closed box, in other words, the Cartesian product of three half-open half-closed intervals:

$$
\left[b_{s} b_{x}, b_{s} b_{x}+b_{s}\right) \times\left[b_{s} b_{y}, b_{s} b_{y}+b_{s}\right) \times\left[b_{s} b_{z}, b_{s} b_{z}+b_{s}\right) .
$$

Thus, only the grid points and grid edges that lie on the left, top and back faces of the subdomain are considered for the output, while the front, right and bottom faces are ignored as shown in Figure 3. When taking the union of the output from the blocks, we eventually omit all the grid points and grid edges that are on the front, right and bottom boundary faces of the entire domain. Fortunately, by leaving the sufficient margin as mentioned earlier, the whole domain is a bounding box of the molecule, and no inside points or intersection points exist on those faces.

### 2.6. Pipeline

Algorithm 1 provides the pseudocode of the main procedures for the parallelized version of ESES. The function ESES is identical to the original procedure introduced in [14]. Each block (subdomain) uses a unified data structure to store all the necessary information for the part of ESES computation within that block, including the subgrid, the list of relevant atoms, and the output-inside/outside information for grid points and intersection locations on grid edges.

## 3. Spatial and temporal complexity analysis

Our treatment does reduce the total memory requirement but not the total amount of computation. The number of grid point classification operations to perform, and the number of edge and SES intersection tests are not reduced. Nevertheless, as we divide these task into subdomains of the entire domain, we can either handle the previously intractable problem on a commodity PC with few gigabytes of memory, or greatly reduce the amount of time on a computer cluster or a multi-core computer.

## Algorithm 1

ParaESES Algorithm

| 1: | function ParaESES (data, probeRadius, gridSize, margin) |
| :--- | :---: |
| $2:$ | CreateBlocks (data, blocks) |
| $3:$ | for all $b$ E blocks do in parallel |
| $4:$ | AssembleRelevantAtoms $(b)$ |
| $5:$ | GlobalToLocalIndexMapping ( $b$, , bLocal) |
| $6:$ | ESES (bLocal, probeRadius, gridSize, margin) |
| $7:$ | RemoveDuplicates (bLocal) |
| $8:$ | LocalToGlobalIndexMapping $(b$, bLocal) |
| $9:$ | critical |
| $10:$ | OutputInfo $(b)$ |
| $11:$ | end critical |
| $12:$ | end for |

As the number of atoms grows larger or as the resolution becomes finer, the number of grid cells increases asymptotically at $O\left(w h d / s^{3}\right)$, where $s$ is the grid spacing, and $w, h$, and $d$ are the width, height, and depth of the bounding box of the molecule respectively. When dealing with large proteins in the original ESES [14] on fine grids, the memory storage is essentially cubic to the number of cells along an edge of the box domain. For instance, a grid with the size of $1000 \times 1000 \times 1000$ would require roughly $n$ Giga-Byte if each grid point or grid cell requires $n$ Byte storage. As ESES requires the data to reside in the main memory for the grid point classification and grid edge intersection, it cannot fit in the memory of a regular PC, even with virtual memory.

Our straightforward domain decomposition into blocks can effectively shrink the memory footprint, i.e., the maximum memory requirement at any point of the calculation, since all the information required for the localized calculation is associated with the block (or
subdomain). No matter how large the original grid size is, we can treat the problem as if it is for the subgrid of size $b_{s}^{3}$, as long as we choose each block to be of size $b_{s} \times b_{s} \times b_{s}$. Specifically, there will be $b_{s}^{3}$ grid points to classify, and $3\left(b_{s}-1\right) b_{s}^{2}$ grid edges to check for intersections when processing one block. The overhead introduced to handle the boundary of the blocks is negligible, since for each block, there will be $3 b_{s}^{2}$ duplicate grid points and 6 ( $b_{s}$ -1) $b_{S}$ duplicate grid edges to check. So in terms of each grid block, there will be an approximate ratio of $O\left(1 / b_{s}\right)$ overhead. As mentioned in the previous section, we introduced a procedure to determine which atoms can influence a particular block, which brings an overhead that is also negligible to the dominating time and space requirement for the grid. This part of the overhead can be further reduced by any spatial data-structure such as a kdtree, since the construction of the list of atoms within a block is just a query for spatial database entries within a certain spatial range. However, an overly small block size can increase the proportion of the memory the overhead requires, so we do not recommend to aggressively reduce the block size. Fortunately, in practice, even the memory of any modern smart phone can easily accommodate the block size with $b_{s}=32$.

If multi-core machines or clusters with multiple computers are available, one can use our block-based design to achieve essentially a speedup factor controlled by the number of available cores, assuming each core has access to a memory space that can store one block. In an ideal case with infinitely available cores, the time complexity is dropped to $O\left(b_{s}^{3}\right)$, which is entirely determined by a single block size.

## 4. Validation and application

We performed our tests on a PC with Intel(R) Xeon(R) CPU E5-1630 v3 @ 3.70GHz and 8GB memory. For parallel computing, we used OpenMP (https://computing.llnl.gov/ tutorials/openMP/). We first tested our algorithm on an extremely large multiprotein complex to verify the capability of our algorithm. Then, we analyze the impact of using different block sizes and numbers of threads in terms of the execution time and memory footprint at various grid spacings. Based on the resulting statistics with varying parameter settings, we have confirmed empirically that the memory footprint and execution time indeed behaved as predicted in our analysis above.

All molecular structures used in our validation were downloaded from Protein Data Bank (PDB, https://www.rcsb.org/). The protein-ligand complexes used in our application were obtained from PDBbing (http://www.pdbbind.org.cn/). All structures were processed with pdb-to-xyzr (https://github.com/Pymol-Scripts/Pymol-script-repo/blob/master/modules/ MSMS/i64Linux2/pdb_to_xyzr) to assume appropriate van der Waals radii in addition to atomic coordinates.

### 4.1. Validation on multiprotein complex

In our tests, the algorithm was able to produce the SES successfully for multiprotein complexes with an arbitrary list of atoms at a very high resolution. For instance, Figure 4 shows a multiprotein complex consisting of tubulin, Drosophila melanogaster kinesin-13

KLP10A and microtubule [1] constructed by protein 3 j 2 u with 15575 atoms shown in Figure 5 as the building block. There are 42 such blocks plotted in this multiprotein complex. This typical protein assembly is crucial for investigating the recognition and deformation of tubulins in a microtubule. Due to its excessively large size, such a complex is always an obstacle to handle in theoretical modeling. However, with our software, by only assigning 8 threads to perform the block-based tasks in parallel, the combined memory footprint is controlled to the reasonable amount of 2 GB . The whole procedure took about 10 minutes to generate the SES output, including the grid point classification and the intersection information. We were also able to mark different chains in the large protein assembly in the process, with auxiliary information provided by our algorithm, which is the nearby atoms of intersections points. This demonstrates the versatility of our algorithm when used in downstream applications, such as solving the Poisson-Boltzmann equation for electrostatic analysis [13].

### 4.2. Single-thread analysis

Protein 5z10, shown in Fig. 6 left, reported by [24] is tested as an example for single-thread performance. This protein is a typical mechanosensitive ion channel constructed by three identical blade-like subunits. It is found that by probing the state of surrounding membrane, the channel opens with the distortion of these three blades.

We provide some memory footprint statistics in Figure 7 with only a single thread launched. As the plot indicates, if we do not incorporate the block design and use the original method with the entire grid residing in the memory at all time, the memory footprint increases for fine grids, shown by the curve with circular nodes. If we use blocks with size $b_{s}=128$, the memory footprint drops significantly, simply because only a single block needs to reside in the memory for the single thread, in addition to other auxiliary information, as shown by the curve with square nodes. For the block size $b_{s}=64$, the curve with diamond nodes shows that the memory footprint is further reduced. By assigning blocks, the memory footprint is dominated and restricted by the information stored within a single block, which is controlled only by the block size and is independent of the grid sizes. Therefore, memory footprint of our approach is well-controlled.

The statistics of execution time is provided in Figure 8 for molecule 5 z 10 with only a single thread. The execution time for different block sizes did not vary significantly. This behavior is expected, since we did not change the total amount of the calculation, for the single-thread version, only the memory footprint is reduced. Note that change was made to the grid point classification or grid edge intersection detection parts. Stated differently, there is no change in analytical nature of the original ESES algorithm.

### 4.3. Multi-thread analysis

Protein 5vkq, shown in Fig. 6 right, reported by [11] is used as an example to test multithread performance. This protein is a typical mechanosensitive ion channel in bacteria. Its long and spring shaped domains are tethered with microtubules, which will open when it senses the motion of the cytoskeleton environment.

If we initiate $N$ threads in parallel, we expect that the memory footprint is roughly $N$ times
the numbers when the number of $N$ is large. In practice, we observed that it is actually smaller due to that some of the overhead is shared by the threads, especially when $N$ is small. In Figure 9, we present the statistics of memory footprint for protein 5 vkq with 8 threads launched in parallel. The curve with circular nodes serves as a baseline when we stick to the original ESES algorithm, which shows an excessive memory requirement. When launching 8 threads, obviously the memory footprint shifted higher compared to launching a single thread simply because we need to load the execution context for all 8 blocks. Nevertheless the memory footprint is still significantly reduced compared to the baseline, unless the number of threads matches the number of blocks. In addition, we still control the memory footprint by the number of threads launched. The curve with square nodes shows the memory usage with block size $b_{s}=128$, and the curve with diamond nodes shows the case with block size $b_{s}=64$.

The execution time statistics for the same 8-thread experiment for protein 5 vkq are shown in Figure 10. The curve with circular nodes gives us a baseline when we stick to the original ESES algorithm. In this example, the execution time was reduced significantly simply by launching several threads at the same time. It is not a perfect 8 -times improvement, as predicted by Amdahl's Law [15], because there are always critical sections that need serial execution such as file I/O. We also found that a smaller block size (curve with diamond nodes) also brings some additional improvements as in the single-thread mode. It is most likely due to the same reason that the memory allocation is easier for smaller blocks. Taking into account both the spatial and temporal statistics, we observed that by reducing the block size, we can significantly reduce the memory footprint without any negative impact on the time performance and algorithm accuracy.

Finally, we apply the present approach to a large set of protein-ligand complexes. We consider the PDBbing v2016 core set of 290 protein-ligand complexes. Our results in terms of grid dimension, block dimension, surface area, and surface enclosed volume are given for each protein in Appendix A1. These results can be used by independent researchers to validate their own surface generations. The computational parameters are set to probe size $1.4 \AA$, grid spacing $0.4 \AA$, grid extension $0.8 \AA$ and block size 64 . Note that the proposed method has no effect on the ESES generation quality. The proposed method can thus be used as an efficient replacement to ESES, and be applied to any solvent excluded surface based molecular modeling and analysis.

## 5. Conclusion

We present a divide-and-conquer approach to solve the memory explosion issue when dealing with large macromolecules at high grid resolutions. The approach is based on the localized nature of the computations involved in Eulerian solute-excluded surfaces (ESESs) [14]. In the present approach, we partition the entire computational domain into subdomains (blocks) that can fit into a given size of memory space. The memory requirement is determined by the data in the block(s) used in the current calculation. In this manner, we can control the upper bound of the memory footprint, and allow the user to run our software on a typical commodity personal computer (PC). Taking the advantage of the locality, we also
incorporate the power of parallel computing to further enhance the performance. With such a practical implementation, we can significantly extend the applicability of the earlier ESES algorithm by lifting its constraints of memory requirements and running on a single CPU. The present improvement does not change the analytical nature of the original ESES algorithm. The proposed method is validated on the ESES generation of an excessively large protein complex and a couple of large proteins. Application is considered to 290 proteinligand complexes.

There is still a room for further improvements. In the potential analytical patch construction, especially for the saddle and concave patches, we simply consider all possible pairs of atoms which are at a distance below a threshold determined by the van der Waals radii. Apparently, there is a redundancy in such an approach, since some patches are buried inside the molecular surface and may be pre-culled to save computation. As future work, we wish to explore fast calculations that can eliminate such patches before classifying grid points. Another direction to explore is to consider GPU computing, since a similar parallelized design can be applied when mapping them to GPU threads and blocks instead of CPU cores. A central issue in carrying out a GPU implementation is how the analytical SES patch construction and the associated high order polynomial root finding at grid edge and surface intersection can be efficiently adapted to the less powerful ALU units on GPUs. Further simplifications may be desirable to harness the power of GPU for this problem.

## Acknowledgments

This work was supported in part by NSF Grants DMS-1721024, DMS-1761320, and CCF-1655422, and NIH grant 1R01GM126189.

## Appendix A.: Surface generation test

Table A1:
Surface generation results of the PDBbing v2016 core set sorted by the number of atoms.

| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | Area <br> $\left(\mathbf{A}^{\mathbf{2}}\right)$ | Volume <br> $\left(\mathbf{A}^{3}\right)$ | time(s) |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| 3 dxg | 1856 | $105 \times 101 \times 117$ | $2 \times 2 \times 2$ | 5777.88 | 16569.2 | 13.0162 |
| 3 d 6 q | 1856 | $105 \times 102 \times 117$ | $2 \times 2 \times 2$ | 5793.47 | 16601.3 | 19.8521 |
| 1w4o | 1856 | $105 \times 99 \times 116$ | $2 \times 2 \times 2$ | 5776.42 | 16575.4 | 14.2584 |
| 1o0h | 1856 | $104 \times 108 \times 117$ | $2 \times 2 \times 2$ | 5753.33 | 16456.4 | 14.2019 |
| 1ulb | 1856 | $103 \times 129 \times 94$ | $2 \times 3 \times 2$ | 5790.07 | 16722.8 | 12.3264 |
| 4 lzs | 2121 | $95 \times 132 \times 127$ | $2 \times 3 \times 2$ | 6585.37 | 18644.5 | 19.2717 |
| 3 u5j | 2121 | $95 \times 135 \times 127$ | $2 \times 3 \times 2$ | 6686.26 | 18677.5 | 17.7138 |
| 4 wiv | 2121 | $138 \times 85 \times 125$ | $3 \times 2 \times 2$ | 6749.24 | 18694.1 | 16.3995 |
| 4 ogj | 2121 | $129 \times 110 \times 110$ | $3 \times 2 \times 2$ | 6710.66 | 18851.7 | 16.8964 |
| $3 p 5 \mathrm{o}$ | 2121 | $96 \times 135 \times 125$ | $2 \times 3 \times 2$ | 6718.7 | 18764.6 | 17.1206 |
| 31 ka | 2408 | $110 \times 112 \times 101$ | $2 \times 2 \times 2$ | 7040.19 | 21685.7 | 16.9318 |
| 3 ehy | 2408 | $110 \times 112 \times 100$ | $2 \times 2 \times 2$ | 7039.49 | 21574.7 | 15.2638 |
| $3 n x 7$ | 2408 | $112 \times 112 \times 101$ | $2 \times 2 \times 2$ | 6960.15 | 21681.9 | 14.4231 |
| 3 tsk | 2425 | $109 \times 100 \times 119$ | $2 \times 2 \times 2$ | 7057.27 | 22082.1 | 14.6475 |

Zhao et al.

| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA_{\AA^{2}}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time( $s$ ) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $4 \mathrm{gr0}$ | 2425 | $109 \times 101 \times 115$ | $2 \times 2 \times 2$ | 6962.62 | 21868.1 | 15.4103 |
| $3 \mathrm{nq9}$ | 2584 | $120 \times 120 \times 112$ | $2 \times 2 \times 2$ | 7368.2 | 23464 | 22.7301 |
| 5aba | 3025 | $111 \times 126 \times 136$ | $2 \times 2 \times 3$ | 8747.75 | 27568.7 | 25.6786 |
| 4agq | 3025 | $113 \times 126 \times 138$ | $2 \times 2 \times 3$ | 8885.8 | 27571.7 | 19.5389 |
| 5a7b | 3047 | $111 \times 130 \times 136$ | $2 \times 3 \times 3$ | 8943.39 | 27637.8 | 27.9725 |
| 4agp | 3047 | $114 \times 126 \times 138$ | $2 \times 2 \times 3$ | 8877.19 | 27718.7 | 27.5211 |
| 4agn | 3069 | $112 \times 126 \times 137$ | $2 \times 2 \times 3$ | 8966.61 | 27884.4 | 29.8699 |
| 2 qnq | 3128 | $104 \times 139 \times 143$ | $2 \times 3 \times 3$ | 8577.43 | 27855.4 | 20.697 |
| 3 cyx | 3128 | $106 \times 134 \times 142$ | $2 \times 3 \times 3$ | 8566.31 | 27591.5 | 21.0516 |
| 1eby | 3134 | $107 \times 136 \times 147$ | $2 \times 3 \times 3$ | 8604.66 | 28389.9 | 21.5876 |
| 309i | 3134 | $152 \times 103 \times 111$ | $3 \times 2 \times 2$ | 8526.46 | 27838 | 19.3083 |
| 1a30 | 3138 | $107 \times 136 \times 140$ | $2 \times 3 \times 3$ | 8459.42 | 27734.2 | 32.6073 |
| 4abg | 3204 | $109 \times 118 \times 135$ | $2 \times 2 \times 3$ | 8104.51 | 29113.2 | 27.3888 |
| 1uto | 3220 | $123 \times 99 \times 132$ | $2 \times 2 \times 3$ | 8037.53 | 29010 | 17.8095 |
| 3 gy 4 | 3220 | $123 \times 107 \times 133$ | $2 \times 2 \times 3$ | 7993.9 | 29061.3 | 20.5542 |
| 1k1i | 3220 | $124 \times 108 \times 131$ | $2 \times 2 \times 3$ | 8127.1 | 29250 | 20.144 |
| 103f | 3220 | $121 \times 132 \times 111$ | $2 \times 3 \times 2$ | 8059.72 | 29193.8 | 19.3656 |
| 3kr8 | 3248 | $140 \times 132 \times 106$ | $3 \times 3 \times 2$ | 9356.53 | 29196.6 | 20.4359 |
| 2yki | 3259 | $122 \times 124 \times 126$ | $2 \times 2 \times 2$ | 9171.29 | 29287.1 | 19.6484 |
| 4kzq | 3278 | $137 \times 130 \times 103$ | $3 \times 3 \times 2$ | 9356.27 | 29733.6 | 19.8373 |
| 4kzu | 3278 | $139 \times 131 \times 108$ | $3 \times 3 \times 2$ | 9308.97 | 29802.4 | 20.4075 |
| 4 j 21 | 3292 | $149 \times 125 \times 107$ | $3 \times 2 \times 2$ | 9533.27 | 30100.4 | 20.4142 |
| 4 j 31 | 3292 | $125 \times 150 \times 106$ | $2 \times 3 \times 2$ | 9398.54 | 29849.4 | 19.9642 |
| 1 yc 1 | 3313 | $119 \times 110 \times 129$ | $2 \times 2 \times 3$ | 9072.47 | 29432.3 | 20.9225 |
| 3 zt | 3357 | $200 \times 148 \times 114$ | $4 \times 3 \times 2$ | 10015 | 30240.5 | 40.6189 |
| 3 zzs | 3357 | $196 \times 150 \times 114$ | $4 \times 3 \times 2$ | 9996.25 | 30152.6 | 26.3215 |
| $3 \mathrm{oe5}$ | 3357 | $193 \times 149 \times 113$ | $4 \times 3 \times 2$ | 9925.33 | 30107 | 27.2306 |
| 3 oe 4 | 3357 | $193 \times 149 \times 114$ | $4 \times 3 \times 2$ | 9980.64 | 30163 | 27.2095 |
| 3nw9 | 3365 | $107 \times 115 \times 144$ | $2 \times 2 \times 3$ | 8131.95 | 30170.6 | 19.38 |
| 3b27 | 3388 | $128 \times 126 \times 159$ | $3 \times 2 \times 3$ | 9731.4 | 30503.3 | 24.2988 |
| 2 fxs | 3409 | $151 \times 128 \times 132$ | $3 \times 3 \times 3$ | 9203.98 | 30434.6 | 23.9637 |
| 2yge | 3420 | $151 \times 130 \times 134$ | $3 \times 3 \times 3$ | 9285.46 | 30724.4 | 27.1157 |
| 2iwx | 3426 | $155 \times 129 \times 133$ | $3 \times 3 \times 3$ | 9345.14 | 30760.3 | 27.0977 |
| 2vw5 | 3426 | $157 \times 129 \times 134$ | $3 \times 3 \times 3$ | 9341.02 | 30901.6 | 28.3237 |
| 3rlr | 3449 | $121 \times 121 \times 156$ | $2 \times 2 \times 3$ | 9914.87 | 30865.4 | 23.8025 |
| 11pg | 3665 | $125 \times 136 \times 114$ | $2 \times 3 \times 2$ | 9706.24 | 32892.4 | 20.5021 |
| 4 crc | 3711 | $127 \times 120 \times 141$ | $2 \times 2 \times 3$ | 9907.48 | 33608.9 | 22.8678 |
| $4 \times 6 \mathrm{p}$ | 3715 | $140 \times 116 \times 126$ | $3 \times 2 \times 2$ | 10131.4 | 33827.4 | 20.8315 |
| 4 cra | 3724 | $145 \times 128 \times 115$ | $3 \times 3 \times 2$ | 9972.72 | 33624 | 22.1204 |
| 4ty7 | 3727 | $133 \times 138 \times 144$ | $3 \times 3 \times 3$ | 10136.1 | 33908.5 | 24.43 |
| 3 kgp | 3737 | $144 \times 112 \times 123$ | $3 \times 2 \times 2$ | 9519.34 | 33612.6 | 23.194 |

Commun Inf Syst. Author manuscript; available in PMC 2019 July 19.
Zhao et al.

| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA_{\AA^{2}}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time( $s$ ) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 105b | 3810 | $146 \times 127 \times 120$ | $3 \times 2 \times 2$ | 9445.63 | 34582.3 | 22.1093 |
| 1 c 5 z | 3811 | $146 \times 129 \times 120$ | $3 \times 3 \times 2$ | 9531.12 | 34644.1 | 33.7694 |
| 1sqa | 3818 | $128 \times 157 \times 125$ | $3 \times 3 \times 2$ | 10141.1 | 35162.1 | 23.7716 |
| lowh | 3820 | $127 \times 156 \times 125$ | $2 \times 3 \times 2$ | 10149.2 | 35375.6 | 23.6066 |
| 4qd6 | 3849 | $134 \times 147 \times 211$ | $3 \times 3 \times 4$ | 12029.2 | 35089.4 | 33.5549 |
| 3qgy | 3896 | $133 \times 135 \times 159$ | $3 \times 3 \times 3$ | 10853.7 | 35113.4 | 28.1767 |
| 4de2 | 3900 | $138 \times 109 \times 147$ | $3 \times 2 \times 3$ | 9295.39 | 34595.6 | 35.9985 |
| 4de3 | 3900 | $138 \times 111 \times 144$ | $3 \times 2 \times 3$ | 9241.18 | 34660.7 | 23.651 |
| 1z95 | 3917 | $141 \times 120 \times 156$ | $3 \times 2 \times 3$ | 10169.7 | 35038.8 | 27.3216 |
| 4de1 | 3923 | $139 \times 108 \times 152$ | $3 \times 2 \times 3$ | 9427.62 | 34862.4 | 23.8837 |
| 3 g 2 z | 3925 | $140 \times 109 \times 152$ | $3 \times 2 \times 3$ | 9285.38 | 34891.5 | 22.2739 |
| 3 g 31 | 3925 | $140 \times 110 \times 147$ | $3 \times 2 \times 3$ | 9326.88 | 34989 | 26.6228 |
| 4 rfm | 3942 | $138 \times 128 \times 154$ | $3 \times 3 \times 3$ | 10899.5 | 35830 | 25.4256 |
| 3kwa | 4032 | $127 \times 125 \times 150$ | $2 \times 2 \times 3$ | 10003.9 | 36326.8 | 35.8207 |
| 4jsz | 4048 | $127 \times 124 \times 149$ | $2 \times 2 \times 3$ | 10121.1 | 36590 | 22.4405 |
| 3ryj | 4054 | $129 \times 123 \times 148$ | $3 \times 2 \times 3$ | 10333.2 | 36435.3 | 23.7567 |
| 3b68 | 4068 | $149 \times 120 \times 158$ | $3 \times 2 \times 3$ | 10827.6 | 36233.7 | 29.5425 |
| 3 b 5 r | 4068 | $149 \times 120 \times 159$ | $3 \times 2 \times 3$ | 10866.6 | 36325.1 | 28.6952 |
| 3b65 | 4068 | $149 \times 120 \times 158$ | $3 \times 2 \times 3$ | 10801 | 36251.7 | 28.1824 |
| 2weg | 4071 | $128 \times 121 \times 147$ | $3 \times 2 \times 3$ | 10146 | 36513.5 | 24.3336 |
| 3dd0 | 4071 | $130 \times 123 \times 149$ | $3 \times 2 \times 3$ | 10322.6 | 36951.8 | 23.7389 |
| 3 gbb | 4086 | $123 \times 149 \times 157$ | $2 \times 3 \times 3$ | 11431.4 | 36652.7 | 27.8599 |
| 3 g 0 w | 4099 | $153 \times 124 \times 150$ | $3 \times 2 \times 3$ | 10647.3 | 36712.6 | 27.9932 |
| 3fv2 | 4101 | $145 \times 134 \times 121$ | $3 \times 3 \times 2$ | 11330.7 | 36688 | 24.859 |
| 3 fv 1 | 4101 | $145 \times 140 \times 122$ | $3 \times 3 \times 2$ | 11270.2 | 36526.1 | 24.8372 |
| 3 fur | 4161 | $133 \times 110 \times 172$ | $3 \times 2 \times 3$ | 11796.3 | 37539 | 25.4867 |
| 3 myg | 4169 | $156 \times 145 \times 126$ | $3 \times 3 \times 2$ | 11063 | 37469.2 | 27.555 |
| 4 m 0 y | 4177 | $131 \times 152 \times 148$ | $3 \times 3 \times 3$ | 11990.4 | 38728.9 | 29.0143 |
| 3 u 9 q | 4179 | $124 \times 131 \times 154$ | $2 \times 3 \times 3$ | 11740 | 37330.8 | 33.5672 |
| 3jvs | 4186 | $139 \times 118 \times 183$ | $3 \times 2 \times 3$ | 11445 | 37468.2 | 27.3559 |
| 4 m 0 z | 4199 | $128 \times 148 \times 149$ | $3 \times 3 \times 3$ | 11655.2 | 38361.3 | 30.2577 |
| 3ao4 | 4210 | $137 \times 133 \times 144$ | $3 \times 3 \times 3$ | 10992 | 38287.1 | 26.372 |
| 3 jvr | 4218 | $138 \times 116 \times 181$ | $3 \times 2 \times 3$ | 11495.1 | 37840.2 | 24.8808 |
| 3 b 1 m | 4229 | $116 \times 129 \times 169$ | $2 \times 3 \times 3$ | 11722.6 | 37809.4 | 27.7153 |
| 3 mss | 4263 | $131 \times 118 \times 172$ | $3 \times 2 \times 3$ | 12047.8 | 38810.3 | 38.2454 |
| 2c3i | 4274 | $154 \times 144 \times 123$ | $3 \times 3 \times 2$ | 11301.2 | 38876.8 | 26.4624 |
| 2 yfe | 4281 | $139 \times 156 \times 153$ | $3 \times 3 \times 3$ | 11923.7 | 38510.9 | 31.6776 |
| 3 pyy | 4289 | $111 \times 128 \times 166$ | $2 \times 3 \times 3$ | 11972.6 | 38784.4 | 25.2828 |
| 1 nvq | 4291 | $139 \times 124 \times 181$ | $3 \times 2 \times 3$ | 11909.9 | 38394.9 | 28.3015 |
| 2 xbv | 4291 | $149 \times 135 \times 131$ | $3 \times 3 \times 3$ | 11794.8 | 39406.4 | 27.6058 |
| 4twp | 4292 | $165 \times 147 \times 113$ | $3 \times 3 \times 2$ | 11511.7 | 38978.3 | 27.6603 |

Zhao et al.

| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA_{\AA^{2}}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time( $s$ ) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 3 bgz | 4318 | $158 \times 141 \times 121$ | $3 \times 3 \times 2$ | 11265.8 | 39021.6 | 28.2811 |
| 2wtv | 4337 | $118 \times 161 \times 160$ | $2 \times 3 \times 3$ | 11713 | 38572.4 | 27.7694 |
| 2v7a | 4340 | $129 \times 136 \times 166$ | $3 \times 3 \times 3$ | 12063.1 | 39932 | 28.3806 |
| 1mq6 | 4372 | $167 \times 139 \times 112$ | $3 \times 3 \times 2$ | 11625.3 | 39651.5 | 26.7912 |
| 3uo4 | 4392 | $157 \times 142 \times 139$ | $3 \times 3 \times 3$ | 11569.5 | 39341.1 | 32.1715 |
| 3up2 | 4392 | $158 \times 145 \times 139$ | $3 \times 3 \times 3$ | 11612.7 | 39524.3 | 31.3933 |
| 5dwr | 4397 | $135 \times 151 \times 134$ | $3 \times 3 \times 3$ | 11467.8 | 40084.1 | 27.2752 |
| 3 jya | 4408 | $134 \times 155 \times 133$ | $3 \times 3 \times 3$ | 11571 | 40232.4 | 27.4488 |
| 1z6e | 4413 | $164 \times 142 \times 109$ | $3 \times 3 \times 2$ | 11656.5 | 39643.9 | 25.5687 |
| 2 brb | 4425 | $141 \times 133 \times 182$ | $3 \times 3 \times 3$ | 12414 | 40414.9 | 31.3865 |
| 2br1 | 4425 | $138 \times 135 \times 183$ | $3 \times 3 \times 3$ | 12257.1 | 39755.9 | 32.0098 |
| 3utu | 4434 | $128 \times 130 \times 139$ | $3 \times 3 \times 3$ | 11221.5 | 39829.2 | 24.9517 |
| 2 y 5 h | 4440 | $140 \times 168 \times 116$ | $3 \times 3 \times 2$ | 12126 | 40601.4 | 26.7111 |
| 1 bcu | 4446 | $128 \times 129 \times 140$ | $3 \times 3 \times 3$ | 11130.7 | 40337.1 | 37.4077 |
| 4k18 | 4453 | $161 \times 141 \times 132$ | $3 \times 3 \times 3$ | 11652.3 | 40461.1 | 28.9929 |
| 1oyt | 4479 | $129 \times 132 \times 140$ | $3 \times 3 \times 3$ | 11485.2 | 40270.1 | 26.678 |
| 2zda | 4513 | $131 \times 131 \times 143$ | $3 \times 3 \times 3$ | 11344.7 | 40606 | 27.9352 |
| 4k77 | 4549 | $146 \times 142 \times 150$ | $3 \times 3 \times 3$ | 12696.8 | 41470.2 | 30.1223 |
| 4cig | 4564 | $144 \times 125 \times 152$ | $3 \times 2 \times 3$ | 12304.1 | 41078.4 | 40.9304 |
| 3 k 5 v | 4588 | $143 \times 184 \times 124$ | $3 \times 3 \times 2$ | 13284.3 | 41730.7 | 29.958 |
| 3bv9 | 4621 | $138 \times 138 \times 136$ | $3 \times 3 \times 3$ | 11636.2 | 41530.9 | 27.8745 |
| 3zt2 | 4642 | $142 \times 131 \times 155$ | $3 \times 3 \times 3$ | 12440.7 | 41627.9 | 32.2067 |
| 3zsx | 4642 | $144 \times 130 \times 153$ | $3 \times 3 \times 3$ | 12556.8 | 41830 | 45.4165 |
| 2zy1 | 4646 | $149 \times 144 \times 159$ | $3 \times 3 \times 3$ | 12283.4 | 41678.4 | 31.8607 |
| 3uri | 4650 | $147 \times 152 \times 134$ | $3 \times 3 \times 3$ | 11386.5 | 42463.2 | 29.2076 |
| 2 fvd | 4653 | $140 \times 114 \times 170$ | $3 \times 2 \times 3$ | 12820 | 41544.9 | 30.1102 |
| 2v00 | 4669 | $139 \times 127 \times 166$ | $3 \times 2 \times 3$ | 11462.2 | 42410.6 | 42.6454 |
| 3wz8 | 4669 | $140 \times 127 \times 167$ | $3 \times 2 \times 3$ | 11460.9 | 42536.5 | 27.7162 |
| 3pww | 4669 | $138 \times 126 \times 166$ | $3 \times 2 \times 3$ | 11500.2 | 42377.7 | 30.2387 |
| 3 prs | 4669 | $139 \times 125 \times 166$ | $3 \times 2 \times 3$ | 11499.9 | 42426.3 | 28.767 |
| 3zso | 4670 | $143 \times 132 \times 153$ | $3 \times 3 \times 3$ | 12690 | 41991.9 | 31.6446 |
| 4ea2 | 4670 | $148 \times 144 \times 161$ | $3 \times 3 \times 3$ | 12343.4 | 42173 | 32.4862 |
| 2 zcr | 4670 | $149 \times 147 \times 160$ | $3 \times 3 \times 3$ | 12337 | 41938.9 | 33.0923 |
| 4hge | 4674 | $161 \times 156 \times 143$ | $3 \times 3 \times 3$ | 13043.1 | 42660 | 33.0706 |
| 4ivd | 4698 | $214 \times 143 \times 158$ | $4 \times 3 \times 3$ | 13788.6 | 42619.9 | 40.4818 |
| 3 fcq | 4700 | $173 \times 131 \times 132$ | $3 \times 3 \times 3$ | 10956.1 | 42376.4 | 34.77 |
| $1 \mathrm{z9g}$ | 4700 | $175 \times 132 \times 133$ | $3 \times 3 \times 3$ | 11019.2 | 42836.4 | 29.4944 |
| $1 \mathrm{qf1}$ | 4700 | $173 \times 132 \times 132$ | $3 \times 3 \times 3$ | 10908.5 | 42623.9 | 28.7241 |
| 4ivb | 4713 | $215 \times 143 \times 158$ | $4 \times 3 \times 3$ | 13772.8 | 42947.1 | 41.5465 |
| 4ivc | 4713 | $213 \times 142 \times 158$ | $4 \times 3 \times 3$ | 13712.4 | 42361 | 40.4934 |
| 3 cw | 4714 | $150 \times 146 \times 161$ | $3 \times 3 \times 3$ | 12637.4 | 42325.7 | 48.4493 |


| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA_{\AA^{2}}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time( $s$ ) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 2zcq | 4714 | $145 \times 143 \times 157$ | $3 \times 3 \times 3$ | 12308.2 | 42281.4 | 31.2949 |
| 4f09 | 4739 | $143 \times 159 \times 169$ | $3 \times 3 \times 3$ | 13163 | 42812.4 | 35.2086 |
| 4 gfm | 4762 | $145 \times 148 \times 129$ | $3 \times 3 \times 3$ | 12925.4 | 43402.4 | 28.8006 |
| 1 pxn | 4788 | $142 \times 114 \times 170$ | $3 \times 2 \times 3$ | 13001.5 | 42799.9 | 30.9308 |
| 2hb1 | 4811 | $170 \times 145 \times 117$ | $3 \times 3 \times 2$ | 11635.3 | 43180.8 | 45.4613 |
| 1bzc | 4811 | $173 \times 142 \times 115$ | $3 \times 3 \times 2$ | 11505.4 | 43489.1 | 30.1936 |
| 2 qbq | 4811 | $170 \times 145 \times 118$ | $3 \times 3 \times 2$ | 11678.6 | 43351.1 | 31.5649 |
| 2 qbp | 4811 | $169 \times 145 \times 117$ | $3 \times 3 \times 2$ | 11655.4 | 43424.7 | 31.878 |
| 2 xnb | 4819 | $141 \times 111 \times 171$ | $3 \times 2 \times 3$ | 12885.2 | 43258.1 | 29.5643 |
| 2 qbr | 4830 | $177 \times 145 \times 120$ | $3 \times 3 \times 2$ | 11737.8 | 43724.4 | 30.8117 |
| 3 e 5 a | 4850 | $152 \times 161 \times 145$ | $3 \times 3 \times 3$ | 13072.8 | 43625.6 | 32.6783 |
| 4 e q | 4869 | $168 \times 138 \times 181$ | $3 \times 3 \times 3$ | 13820.9 | 43831.2 | 37.587 |
| 4jia | 4878 | $142 \times 183 \times 165$ | $3 \times 3 \times 3$ | 14222.9 | 44267.3 | 37.1466 |
| 3 pxf | 4908 | $157 \times 110 \times 180$ | $3 \times 2 \times 3$ | 13841.6 | 43820.6 | 43.9436 |
| 3 uno | 4922 | $145 \times 150 \times 157$ | $3 \times 3 \times 3$ | 12998.8 | 45572.7 | 31.3175 |
| 3 ueu | 4966 | $123 \times 130 \times 203$ | $2 \times 3 \times 4$ | 13255.2 | 44763.8 | 32.4677 |
| 3uew | 5000 | $123 \times 128 \times 213$ | $2 \times 3 \times 4$ | 13666.4 | 45435.3 | 33.8787 |
| 3twp | 5009 | $164 \times 127 \times 133$ | $3 \times 2 \times 3$ | 12462.3 | 45286.2 | 45.8088 |
| 3 qqs | 5009 | $134 \times 166 \times 147$ | $3 \times 3 \times 3$ | 12412.7 | 45349.6 | 31.8109 |
| 3 uev | 5068 | $125 \times 128 \times 201$ | $2 \times 3 \times 4$ | 13640.4 | 45616.3 | 31.0619 |
| $3 \mathrm{ui7}$ | 5091 | $142 \times 151 \times 157$ | $3 \times 3 \times 3$ | 13135.4 | 46772.9 | 32.3145 |
| 3 uex | 5096 | $124 \times 129 \times 209$ | $2 \times 3 \times 4$ | 13975.3 | 46266 | 32.1042 |
| 5c2h | 5191 | $151 \times 147 \times 149$ | $3 \times 3 \times 3$ | 13072.6 | 47126.9 | 35.0963 |
| 5c1w | 5235 | $147 \times 156 \times 157$ | $3 \times 3 \times 3$ | 13216.2 | 47644.6 | 36.0221 |
| 5c28 | 5235 | $148 \times 154 \times 154$ | $3 \times 3 \times 3$ | 13159.3 | 47569.4 | 32.2369 |
| 3ag9 | 5267 | $137 \times 174 \times 143$ | $3 \times 3 \times 3$ | 13473.3 | 47423.8 | 36.549 |
| 4w9h | 5271 | $174 \times 143 \times 199$ | $3 \times 3 \times 4$ | 14645.7 | 47401.8 | 41.345 |
| 4 msc | 5280 | $147 \times 158 \times 158$ | $3 \times 3 \times 3$ | 13787.5 | 48439.7 | 34.8642 |
| 4w9c | 5282 | $175 \times 145 \times 200$ | $3 \times 3 \times 4$ | 14685.8 | 47924.2 | 58.7774 |
| 4w91 | 5282 | $174 \times 145 \times 197$ | $3 \times 3 \times 4$ | 14812.9 | 47975.6 | 41.5329 |
| 4w9i | 5282 | $177 \times 143 \times 199$ | $3 \times 3 \times 4$ | 14614.3 | 47854.5 | 40.2372 |
| 4bkt | 5296 | $141 \times 179 \times 201$ | $3 \times 3 \times 4$ | 14714.9 | 48335 | 57.7965 |
| 411x | 5299 | $150 \times 159 \times 157$ | $3 \times 3 \times 3$ | 13540.6 | 48052.3 | 38.9411 |
| 4 mrw | 5299 | $150 \times 159 \times 157$ | $3 \times 3 \times 3$ | 13525.4 | 48193.6 | 50.8482 |
| 4 mrz | 5299 | $149 \times 160 \times 156$ | $3 \times 3 \times 3$ | 13504.9 | 47956.9 | 52.8705 |
| 4 msn | 5299 | $150 \times 159 \times 155$ | $3 \times 3 \times 3$ | 13688.1 | 48625.8 | 51.905 |
| 4dli | 5411 | $185 \times 140 \times 133$ | $3 \times 3 \times 3$ | 15130.5 | 49108.4 | 36.0334 |
| 4f9w | 5433 | $185 \times 140 \times 129$ | $3 \times 3 \times 3$ | 14761.8 | 48989.7 | 33.0079 |
| 2zb1 | 5575 | $155 \times 149 \times 155$ | $3 \times 3 \times 3$ | 15142.3 | 50414.7 | 37.3719 |
| 3 gv 9 | 5581 | $153 \times 144 \times 145$ | $3 \times 3 \times 3$ | 12928.2 | 50191.6 | 33.2439 |
| 3 gr 2 | 5581 | $153 \times 140 \times 144$ | $3 \times 3 \times 3$ | 12961.9 | 50214 | 35.9617 |

Zhao et al.

| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA_{\AA^{2}}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time( $s$ ) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 4kz6 | 5581 | $151 \times 156 \times 133$ | $3 \times 3 \times 3$ | 12812.9 | 49828.8 | 48.0046 |
| 4jxs | 5581 | $151 \times 157 \times 134$ | $3 \times 3 \times 3$ | 13053.9 | 50220.6 | 47.9802 |
| 2r9w | 5581 | $154 \times 141 \times 143$ | $3 \times 3 \times 3$ | 13174 | 49905.5 | 34.4525 |
| 3 e 93 | 5584 | $156 \times 156 \times 159$ | $3 \times 3 \times 3$ | 15401.5 | 50252.8 | 39.2194 |
| 1r5y | 5595 | $146 \times 134 \times 181$ | $3 \times 3 \times 3$ | 13256.5 | 50372 | 36.9078 |
| 3 e 92 | 5609 | $156 \times 152 \times 158$ | $3 \times 3 \times 3$ | 15307 | 50559.2 | 38.9401 |
| 1s38 | 5650 | $146 \times 132 \times 179$ | $3 \times 3 \times 3$ | 13268 | 50828.2 | 36.4457 |
| 1ydr | 5813 | $136 \times 178 \times 129$ | $3 \times 3 \times 3$ | 14353.4 | 52608.3 | 34.2593 |
| 1ydt | 5813 | $136 \times 177 \times 129$ | $3 \times 3 \times 3$ | 14423 | 52672.4 | 33.32 |
| 3rsx | 5833 | $154 \times 178 \times 132$ | $3 \times 3 \times 3$ | 14770.9 | 54165.6 | 54.6174 |
| 1q8t | 5875 | $141 \times 186 \times 138$ | $3 \times 3 \times 3$ | 15349.5 | 54420.4 | 53.2045 |
| 1q8u | 5928 | $141 \times 187 \times 134$ | $3 \times 3 \times 3$ | 15888.7 | 55111.5 | 36.2449 |
| 4 gid | 5997 | $160 \times 173 \times 150$ | $3 \times 3 \times 3$ | 14746.3 | 54907.2 | 41.5304 |
| 2vkm | 6019 | $160 \times 175 \times 149$ | $3 \times 3 \times 3$ | 14775.8 | 54738.4 | 41.6591 |
| 4djv | 6034 | $166 \times 149 \times 174$ | $3 \times 3 \times 3$ | 14975.7 | 55246.1 | 42.4239 |
| 3udh | 6111 | $166 \times 182 \times 131$ | $3 \times 3 \times 3$ | 15607.2 | 55890.3 | 42.4351 |
| 3 wtj | 6504 | $197 \times 116 \times 177$ | $4 \times 2 \times 3$ | 18210.3 | 58882.8 | 39.4775 |
| 2 xdl | 6560 | $131 \times 191 \times 125$ | $3 \times 4 \times 2$ | 17346.2 | 59278 | 48.6823 |
| 2qe4 | 6651 | $172 \times 149 \times 145$ | $3 \times 3 \times 3$ | 16615.2 | 59904.7 | 39.4594 |
| 2wer | 6846 | $129 \times 223 \times 147$ | $3 \times 4 \times 3$ | 17262.1 | 61418.2 | 42.2958 |
| 4f3c | 6854 | $200 \times 136 \times 128$ | $4 \times 3 \times 3$ | 14854 | 62181.4 | 40.3111 |
| 1 nc 3 | 6896 | $167 \times 108 \times 189$ | $3 \times 2 \times 3$ | 15249.6 | 61913 | 39.4786 |
| 1 nc 1 | 6905 | $168 \times 106 \times 190$ | $3 \times 2 \times 3$ | 15224.4 | 62500 | 37.7971 |
| 1 y 6 r | 6905 | $169 \times 107 \times 191$ | $3 \times 2 \times 4$ | 15335.1 | 62804.4 | 37.6239 |
| 4f2w | 6979 | $109 \times 145 \times 201$ | $2 \times 3 \times 4$ | 15692.2 | 63726.8 | 37.0809 |
| 2cet | 7025 | $156 \times 147 \times 148$ | $3 \times 3 \times 3$ | 14613.8 | 63522.8 | 41.3852 |
| 4 jfs | 7054 | $157 \times 181 \times 185$ | $3 \times 3 \times 3$ | 16911.8 | 63804.5 | 48.7066 |
| 4j28 | 7054 | $157 \times 184 \times 184$ | $3 \times 3 \times 3$ | 17110.9 | 64082 | 46.2038 |
| 2xii | 7054 | $160 \times 148 \times 195$ | $3 \times 3 \times 4$ | 16914 | 64003.5 | 44.4223 |
| 2j7h | 7066 | $155 \times 149 \times 149$ | $3 \times 3 \times 3$ | 14854.5 | 64039.6 | 42.2341 |
| 4 pcs | 7076 | $188 \times 174 \times 177$ | $3 \times 3 \times 3$ | 17301.3 | 64169.7 | 49.6077 |
| 2cbv | 7141 | $162 \times 161 \times 150$ | $3 \times 3 \times 3$ | 14978.2 | 64682.5 | 46.0519 |
| 2j78 | 7142 | $163 \times 153 \times 156$ | $3 \times 3 \times 3$ | 15058.2 | 64684 | 47.1593 |
| 2pog | 7210 | $161 \times 174 \times 166$ | $3 \times 3 \times 3$ | 18032.6 | 64766.2 | 46.4577 |
| 4 cr 9 | 7448 | $213 \times 142 \times 171$ | $4 \times 3 \times 3$ | 19486.5 | 67552.5 | 67.7259 |
| 2p4y | 7688 | $192 \times 177 \times 185$ | $4 \times 3 \times 3$ | 20497.6 | 69126.1 | 55.2999 |
| 4 mgd | 7695 | $167 \times 173 \times 185$ | $3 \times 3 \times 3$ | 19584.2 | 69673.7 | 75.4981 |
| 1vso | 7794 | $173 \times 213 \times 149$ | $3 \times 4 \times 3$ | 20774.4 | 69624.7 | 71.1788 |
| 2p15 | 7849 | $172 \times 166 \times 189$ | $3 \times 3 \times 3$ | 19652.8 | 71004.1 | 50.9455 |
| 1qkt | 8026 | $184 \times 199 \times 165$ | $3 \times 4 \times 3$ | 21676 | 72259.2 | 56.2978 |
| 4mme | 8118 | $157 \times 182 \times 161$ | $3 \times 3 \times 3$ | 17939 | 73566.5 | 52.593 |

Page 16

Commun Inf Syst. Author manuscript; available in PMC 2019 July 19.
Zhao et al.

| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA_{\AA^{2}}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time(s) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 1p1q | 8134 | $188 \times 174 \times 146$ | $3 \times 3 \times 3$ | 20746 | 72655.6 | 52.0299 |
| 1 p 1 n | 8134 | $191 \times 188 \times 149$ | $4 \times 3 \times 3$ | 21060.8 | 72683.9 | 55.3808 |
| 4dld | 8158 | $172 \times 162 \times 182$ | $3 \times 3 \times 3$ | 21683.9 | 73727 | 50.1094 |
| 4u4s | 8162 | $194 \times 214 \times 155$ | $4 \times 4 \times 3$ | 21407.1 | 73234.4 | 68.8114 |
| 1syi | 8170 | $159 \times 151 \times 182$ | $3 \times 3 \times 3$ | 21317.4 | 73091.7 | 47.9248 |
| 2al5 | 8186 | $148 \times 199 \times 191$ | $3 \times 4 \times 4$ | 21975.7 | 73709.8 | 54.9643 |
| 1h23 | 8286 | $174 \times 168 \times 164$ | $3 \times 3 \times 3$ | 18128.9 | 75779.4 | 49.7802 |
| 1h22 | 8296 | $174 \times 168 \times 159$ | $3 \times 3 \times 3$ | 17949.9 | 76064.8 | 50.8137 |
| 1 gpk | 8301 | $170 \times 171 \times 166$ | $3 \times 3 \times 3$ | 18075.1 | 75389.9 | 52.4765 |
| 1 gpn | 8303 | $171 \times 171 \times 161$ | $3 \times 3 \times 3$ | 18205.4 | 75497 | 51.2762 |
| 3 coy | 8382 | $171 \times 162 \times 236$ | $3 \times 3 \times 4$ | 21615.4 | 75471.6 | 56.5176 |
| 3ivg | 8393 | $171 \times 164 \times 225$ | $3 \times 3 \times 4$ | 21473.8 | 75420.7 | 77.236 |
| 3 coz | 8472 | $171 \times 163 \times 236$ | $3 \times 3 \times 4$ | 21908.6 | 76264.1 | 55.5484 |
| 3 aru | 8474 | $258 \times 145 \times 182$ | $5 \times 3 \times 3$ | 19817.1 | 77814.7 | 90.0987 |
| 4ddh | 8495 | $168 \times 163 \times 226$ | $3 \times 3 \times 4$ | 21761.8 | 76872.6 | 83.0679 |
| 4ddk | 8519 | $169 \times 164 \times 236$ | $3 \times 3 \times 4$ | 21986.4 | 76989.4 | 54.1794 |
| 3 arp | 8549 | $249 \times 158 \times 177$ | $4 \times 3 \times 3$ | 20026.1 | 78534 | 58.3026 |
| 3arv | 8563 | $249 \times 158 \times 178$ | $4 \times 3 \times 3$ | 20188.7 | 78669.3 | 56.6699 |
| 3 ary | 8563 | $249 \times 159 \times 176$ | $4 \times 3 \times 3$ | 20110.6 | 78665.2 | 58.0375 |
| 3 arq | 8563 | $249 \times 158 \times 177$ | $4 \times 3 \times 3$ | 20159.2 | 78685.2 | 59.7228 |
| 4eo8 | 8692 | $161 \times 174 \times 194$ | $3 \times 3 \times 4$ | 21052 | 78285.7 | 55.5408 |
| 4ih7 | 8701 | $164 \times 173 \times 198$ | $3 \times 3 \times 4$ | 20699.9 | 79327.2 | 53.9337 |
| 4ih5 | 8703 | $168 \times 173 \times 196$ | $3 \times 3 \times 4$ | 20942.1 | 79392 | 77.9905 |
| 3cj4 | 8720 | $174 \times 168 \times 193$ | $3 \times 3 \times 4$ | 21129.3 | 78945.2 | 56.0903 |
| 3gnw | 8738 | $162 \times 166 \times 201$ | $3 \times 3 \times 4$ | 20977.2 | 78570.2 | 53.2703 |
| 4eor | 8949 | $165 \times 203 \times 190$ | $3 \times 4 \times 3$ | 21622.7 | 82242.2 | 57.4209 |
| 4e5w | 9243 | $234 \times 238 \times 156$ | $4 \times 4 \times 3$ | 25469.5 | 84609.8 | 69.0594 |
| 2wca | 9295 | $232 \times 207 \times 176$ | $4 \times 4 \times 3$ | 21978.8 | 84461.4 | 67.2198 |
| 2w4x | 9306 | $230 \times 207 \times 177$ | $4 \times 4 \times 3$ | 21814.6 | 83862 | 64.4236 |
| 5 tmn | 9400 | $213 \times 184 \times 178$ | $4 \times 3 \times 3$ | 21020.7 | 86484.7 | 58.6322 |
| 4 tmn | 9400 | $213 \times 184 \times 178$ | $4 \times 3 \times 3$ | 21224.8 | 85989 | 58.9291 |
| 3 r 88 | 9948 | $203 \times 241 \times 214$ | $4 \times 4 \times 4$ | 23584.2 | 90457.6 | 81.9614 |
| 4 gkm | 10018 | $234 \times 134 \times 241$ | $4 \times 3 \times 4$ | 23776.8 | 90996.8 | 65.9909 |
| 4owm | 10046 | $217 \times 157 \times 239$ | $4 \times 3 \times 4$ | 23757.2 | 90913.7 | 92.6124 |
| 2w66 | 10441 | $198 \times 259 \times 184$ | $4 \times 5 \times 3$ | 24986.9 | 93973 | 111.381 |
| 2 vvn | 10458 | $199 \times 259 \times 185$ | $4 \times 5 \times 3$ | 24967.8 | 94680.3 | 72.0086 |
| 3 ge 7 | 11112 | $212 \times 130 \times 232$ | $4 \times 3 \times 4$ | 24214.1 | 100795 | 68.4804 |
| 3 gc 5 | 11402 | $210 \times 131 \times 233$ | $4 \times 3 \times 4$ | 24774 | 102959 | 65.0512 |
| 3 rr 4 | 11416 | $210 \times 132 \times 231$ | $4 \times 3 \times 4$ | 24687.2 | 102504 | 94.0242 |
| 3 g 2 n | 13194 | $212 \times 195 \times 213$ | $4 \times 4 \times 4$ | 29288.2 | 122132 | 114.925 |
| 2 wvt | 14130 | $181 \times 167 \times 298$ | $3 \times 3 \times 5$ | 32575.7 | 128126 | 86.4621 |


| Protein <br> ID | \#Atoms | Grid Dim | Block <br> Dim | $\begin{aligned} & \text { Area } \\ & \left(\AA^{2}\right) \end{aligned}$ | Volume $\left(\AA^{3}\right)$ | time(s) |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 2wbg | 14288 | $253 \times 208 \times 176$ | $4 \times 4 \times 3$ | 28560.6 | 129288 | 118.505 |
| 3 zdg | 15709 | $206 \times 200 \times 214$ | $4 \times 4 \times 4$ | 38866 | 144437 | 85.0419 |
| 3 u 8 n | 15960 | $189 \times 211 \times 224$ | $3 \times 4 \times 4$ | 40327.3 | 147464 | 88.1512 |
| 3u8k | 16004 | $178 \times 208 \times 212$ | $3 \times 4 \times 4$ | 41707.7 | 149056 | 87.3808 |
| 2xys | 16045 | $233 \times 229 \times 215$ | $4 \times 4 \times 4$ | 40437 | 146928 | 100.344 |
| 1 ps 3 | 16157 | $205 \times 213 \times 246$ | $4 \times 4 \times 4$ | 33802.4 | 148038 | 94.0591 |
| 3 dx 1 | 16185 | $204 \times 212 \times 251$ | $4 \times 4 \times 4$ | 33359.3 | 147934 | 152.277 |
| 3 d 4 z | 16185 | $205 \times 211 \times 248$ | $4 \times 4 \times 4$ | 33016.8 | 147051 | 100.968 |
| 3 dx 2 | 16185 | $206 \times 215 \times 246$ | $4 \times 4 \times 4$ | 33246.5 | 147399 | 106.006 |
| 3ejr | 16185 | $205 \times 213 \times 248$ | $4 \times 4 \times 4$ | 33829.1 | 148567 | 106.539 |
| 3f3a | 16252 | $232 \times 181 \times 258$ | $4 \times 3 \times 5$ | 34144.4 | 144819 | 151.338 |
| 3f3c | 16364 | $233 \times 194 \times 252$ | $4 \times 4 \times 4$ | 34288.8 | 147783 | 104.871 |
| 3f3d | 16364 | $235 \times 194 \times 254$ | $4 \times 4 \times 5$ | 34513.1 | 149136 | 104.439 |
| 3f3e | 16364 | $233 \times 193 \times 253$ | $4 \times 4 \times 4$ | 34231.4 | 147296 | 102.979 |
| 2wn9 | 16453 | $209 \times 212 \times 199$ | $4 \times 4 \times 4$ | 42401.8 | 150607 | 92.4032 |
| 4 qac | 16464 | $197 \times 215 \times 214$ | $4 \times 4 \times 4$ | 41359.3 | 151072 | 90.1083 |
| 2wnc | 16628 | $231 \times 210 \times 223$ | $4 \times 4 \times 4$ | 42586.5 | 153057 | 99.1218 |
| $2 \times 00$ | 16661 | $212 \times 214 \times 201$ | $4 \times 4 \times 4$ | 43256.1 | 154206 | 90.0651 |
| 1 e 66 | 16692 | $288 \times 256 \times 222$ | $5 \times 5 \times 4$ | 36312.4 | 153168 | 124.091 |
| 2xj7 | 20502 | $207 \times 307 \times 309$ | $4 \times 5 \times 5$ | 48182.4 | 186415 | 146.431 |
| 3 n 7 a | 25763 | $233 \times 248 \times 260$ | $4 \times 4 \times 5$ | 54932.7 | 236584 | 211.186 |
| 4ciw | 25848 | $266 \times 266 \times 266$ | $5 \times 5 \times 5$ | 55675.4 | 236737 | 164.345 |
| 2xb8 | 25848 | $263 \times 263 \times 263$ | $5 \times 5 \times 5$ | 56332.5 | 236229 | 160.274 |
| 3n86 | 25889 | $233 \times 239 \times 237$ | $4 \times 4 \times 4$ | 54431 | 236562 | 126.98 |
| 3syr | 26110 | $217 \times 217 \times 311$ | $4 \times 4 \times 5$ | 56471.3 | 243244 | 146.197 |
| 317b | 26188 | $218 \times 218 \times 312$ | $4 \times 4 \times 5$ | 56457.9 | 244430 | 149.303 |
| 4eky | 26216 | $217 \times 217 \times 311$ | $4 \times 4 \times 5$ | 56276.4 | 244775 | 217.654 |
| 3ebp | 26216 | $216 \times 216 \times 311$ | $4 \times 4 \times 5$ | 55922.4 | 243389 | 143.218 |
| 3n76 | 26220 | $264 \times 264 \times 264$ | $5 \times 5 \times 5$ | 54813.3 | 237610 | 161.143 |
| 2 ymd | 32988 | $357 \times 324 \times 204$ | $6 \times 6 \times 4$ | 79086.5 | 303394 | 314.233 |

## References

[1]. Asenjo Ana B, Chatterjee Chandrima, Tan Dongyan, Vania DePaoli, Rice William J, Diaz-Avalos Ruben, Silvestry Mariena, and Sosa Hernando, Structural model for tubulin recognition and deformation by kinesin-13 microtubule depolymerases, Cell Reports 3 (2013), no. 3, 759-768. [PubMed: 23434508]
[2]. Bates PW, Chen Z, Sun YH, Wei GW, and Zhao S, Geometric and potential driving formation and evolution of biomolecular surfaces, J. Math. Biol 59 (2009), 193-231. [PubMed: 18941751]
[3]. Bates PW, Wei Guo-Wei, and Zhao Shan, Minimal molecular surfaces and their applications, Journal of Computational Chemistry 29 (2008), no. 3, 380-391. [PubMed: 17591718]
[4]. Blinn J, A generalization of algebraic surface drawing, ACM Transactions on Graphics 1 (1982), no. 3, 235-256.
[5]. Chen Minxin and Lu Benzhuo, Tmsmesh: A robust method for molecular surface mesh generation using a trace technique, J Chem. Theory and Comput 7 (2011), 203-212. [PubMed: 26606233]
[6]. Cheng Ho-Lun and Shi Xinwei, Quality mesh generation for molecular skin surfaces using restricted union of balls, Computational Geometry 42 (2009), 196-206.
[7]. Connolly Michael L., Solvent-accessible surfaces of proteins and nucleic acids, Science 221 (1983), no. 4612, 709-713. [PubMed: 6879170]
[8]. Decherchi S and Rocchia W, A general and Robust Ray-Casting-Based Algorithm for Triangulating Surfaces at the Nanoscale, PLoS ONE 8 (2013), e59744. [PubMed: 23577073]
[9]. Duncan BS and Olson AJ, Shape analysis of molecular surfaces, Biopolymers 33 (1993), 231-238. [PubMed: 8485297]
[10]. Dzubiella J, Swanson Jessica MJ, and McCammon JA, Coupling hydrophobicity, dispersion, and electrostatics in continuum solvent models, Physical Review Letters 96 (2006), no. 8, 087802. [PubMed: 16606226]
[11]. Jin Peng, Bulkley David, Guo Yanmeng, Zhang Wei, Guo Zhenhao, Huynh Walter, Wu Shenping, Meltzer Shan, Cheng Tong, Jan Lily Yeh, et al., Electron cryo-microscopy structure of the mechanotransduction channel nompc, Nature 547 (2017), no. 7661, 118. [PubMed: 28658211]
[12]. Lee Byungkook and Richards Frederic M, The interpretation of protein structures: estimation of static accessibility, Journal of Molecular Biology 55 (1971), no. 3, 379-IN4. [PubMed: 5551392]
[13]. Li Lin, Alper Joshua, and Alexov Emil, Multiscale method for modeling binding phenomena involving large objects: application to kinesin motor domains motion along microtubules, Scientific Reports 6 (2016), 23249. [PubMed: 26988596]
[14]. Liu Beibei, Wang Bao, Zhao Rundong, Tong Yiying, and Wei Guo-Wei, ESES: Software for Eulerian solvent excluded surface, Journal of Computational Chemistry 38 (2017), no. 7, 446466. [PubMed: 28052350]
[15]. McCool Michael D, Robison Arch D, and Reinders James, Structured Parallel Programming: Patterns for Efficient Computation, Elsevier, 2012.
[16]. Opron K, Xia KL, and Wei GW, Fast and anisotropic flexibility-rigidity index for protein flexibility and fluctuation analysis, Journal of Chemical Physics 140 (2014), 234105. [PubMed: 24952521]
[17]. Richards Frederic M, Areas, volumes, packing, and protein structure, Annual Review of Biophysics and Bioengineering 6 (1977), no. 1, 151-176.
[18]. Richmond Timothy J, Solvent accessible surface area and excluded volume in proteins: Analytical equations for overlapping spheres and implications for the hydrophobic effect, Journal of Molecular Biology 178 (1984), no. 1, 63-89. [PubMed: 6548264]
[19]. Rocchia W, Sridharan S, Nicholls A, Alexov E, Chiabrera A, and Honig B, Rapid grid-based construction of the molecular surface and the use of induced surface charge to calculate reaction field energies: Applications to the molecular systems and geometric objects, Journal of Computational Chemistry 23 (2002), 128-137. [PubMed: 11913378]
[20]. Sanner Michel F, Olson Arthur J, and Spehner Jean-Claude, Reduced surface: an efficient way to compute molecular surfaces, Biopolymers 38 (1996), no. 3, 305-320. [PubMed: 8906967]
[21]. Wei Guo-Wei, Differential geometry based multiscale models, Bulletin of Mathematical Biology 72 (2010), no. 6, 1562-1622. [PubMed: 20169418]
[22]. Xia KL, Opron K, and Wei GW, Multiscale multiphysics and multidomain models - Flexibility and rigidity, Journal of Chemical Physics 139 (2013), 194109. [PubMed: 24320318]
[23]. Yu ZY, Holst M, Cheng Y, and McCammon JA, Feature-preserving adaptive mesh generation for molecular shape modeling and simulation, Journal of Molecular Graphics and Modeling 26 (2008), 1370-1380.
[24]. Zhao Qiancheng, Zhou Heng, Chi Shaopeng, Wang Yanfeng, Wang Jianhua, Geng Jie, Wu Kun, Liu Wenhao, Zhang Tingxin, Dong Meng-Qiu, et al., Structure and mechanogating mechanism of the piezol channel, Nature 554 (2018), no. 7693, 487. [PubMed: 29469092]
[25]. Zheng Q, Yang SY, and Wei GW, Molecular surface generation using PDE transform, International Journal for Numerical Methods in Biomedical Engineering 28 (2012), 291-316. [PubMed: 22582140]


Figure 1: Illustration of subdomain based algorithm.
The bounding box of the whole molecular surface is divided into several non-overlapping subdomains, which can be computed independently with a small memory footprint. Finally, patches of the SES from each subdomain can be assembled into a watertight surface identical to the one constructed with ESES. Note that the patches form a watertight surface, the gaps between the adjacent patches from different subdomains are added for visualization only.


Figure 2: 2D Index mapping Example.
All the indices are 0 -based. Given a cell with global coordinates ( $i, j$ ) in block ( $k, I$ ) (green) and its local coordinates ( $m, n$ ) within the block, the one-to-one mapping is straightforward as illustrated. Note that ( $m, n$ ) would require fewer bits to store than $(i, j)$. This is typical in parallel processing such as CUDA threads.


Figure 3: Redundancy elimination.
For each subdomain, we only keep the information on faces with a normal along negative axis directions (red), the other faces (yellow) are omitted because they have already been accounted for by adjacent subdomains. After concatenation (dashed line), only the output for faces of the entire grid with normals along the positive axis directions is missing. However, with the margin padded to the molecule when constructing the domain, they do not contain any intersection information to begin with.


Figure 4: Illustration of the SES generation of multiprotein complex.
Here we show the assembly of 3 j 2 u proteins with different chains being marked by different colors. Inner ring: microtubule; intermediate ring: kinesin-13 head domain; and outer ring: curved tubulin protofilament.


Figure 5: Illustration of the building block protein 3j2u.
It shows Drosophila melanogaster kinesin-13 head domain (yellow) binding to tubulin protofilament (silver and blue) and microtubule (red and green).


Figure 6:
Two models, proteins 5 z 10 (left) and 5 vkq (right), on which tests and statistics of the present algorithm are performed.


Figure 7: Memory footprint comparison at various grid spacings (i̊) for protein 5z10 (singlethread).
Curves with circular, square and diamond nodes are corresponding to no block assigned, block size 128 assignment, and block size 64 assignment, respectively.


Figure 8: Execution time comparison at various grid spacings (i) for protein 5z10 (singlethread).
Curves with circular, square and diamond nodes are corresponding to no block assigned, block size 128 assignment, and block size 64 assignment, respectively.


Figure 9: Memory footprint comparison at various grid spacings (A) for protein 5 vkq (8threads).
Curves with circular, square and diamond nodes are corresponding to no block assigned, block size 128 assignment, and block size 64 assignment, respectively.


Figure 10: Execution time comparison at various grid spacings (i) for protein $\mathbf{5 v k q}$ (8-threads). Curves with circular, square and diamond nodes are corresponding to no block assigned, block size 128 assignment, and block size 64 assignment, respectively.


Figure 11: Execution time analysis.
The scatter plot of execution time vs the number of atoms is given for 290 proteins when 8 threads are used.


[^0]:    zhaorund@msu.edu.

