Eurographics Workshop on Visual Computing for Biomedicine (2008)
C. P. Botha, G. Kindlmann, W. J. Niessen, and B. Preim (Editors)

Computation of more channels in protein molecules

P. Beneš[1] and P. Medek[1] and J. Sochor[1]

[1]Faculty of informatics, Masaryk University, Czech republic

Abstract

In the process of designing drugs it is crucial to perform various analyses of cavities and channels in protein molecules. Chemists also require that more than one ideal channel be computed in a static protein molecule. Three basic approaches for computation of more than a single channel were introduced in recent publications. However, these approaches have several disadvantages. In this paper we propose a new adaptive method for computation of more channels. This new method is piloted on a real data and results are compared with channels identified by chemists as relevant. The comparison indicates that this method is a significant improvement over previous methods, as the method computes less number of similar and biochemically insignificant channels.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Geometric algorithms, languages, and systems

1. Introduction

Protein structure is rather complicated. There are many pockets, clefts and cavities, and also some specific cavities – *channels* (formerly referred to as tunnels) – connecting a certain location inside the molecule – the active site – with the molecule surface. These channels can be used by a small molecule to penetrate into the protein molecule and invoke a chemical reaction there. In the process of protein analysis chemists need to compute and analyze these channels in protein molecules. It is crucial to compute not only the single widest channel from the active site to the molecule surface. In some cases when biochemical modifications happen in the widest channel, the knowledge of other channels that are present in the molecule is substantial.

After computing possible channels, chemists have to verify the biochemical relevance of computed channels. So far, this verification was accomplished only by visual inspection of computed channels. We provide a modification to the process of channel computation by involving the evaluation of channels. This allows us to omit the channels that should be identified as geometrically similar.

Recent methods that allow computing more channels [POB*06, MBS07] use a simple technique to modify a molecule and repeat the computation without any evaluation of channels. Typically a part of a molecule is blocked by a sphere and the next channel is computed in unblocked rest of the molecule. However, as was verified by chemists, this blocking is not sufficient to ensure that the next computed channel is different from the previous one.

2. Related work

Many papers have investigated the issue of automatic detection of cavities in protein molecules. An important approach is presented in [LEW98]. The alpha-shape theory is utilised and potentially important cavities are detected. Other advanced techniques were proposed in [BS00, LJ05]. Nevertheless, these cavities are not treated as paths or channels connecting interior of the protein with its surface.

The channel analysis is usually performed from the geometrical point of view only. The protein is simplified to a set of spheres which represent atoms. The small molecule that penetrates into the protein is represented by its bounding sphere, therefore a channel could be defined as a centerline and a volume (see Fig. 1), which is formed by a union of spheres inserted into each point of the centerline. The radius of such a sphere is always maximal, so that it does not intersect any atom. The size of the smallest sphere along the centerline, the bottleneck sphere, is usually the main criterion by which channels are compared. This value confines the size of the molecule able to pass through the tunnel. An

extension which takes more criteria into account is proposed in [MBS08]. A function which combines several factors into one value could be also used for this purpose [POB*06].

To the best of our knowledge, there are two major approaches to channel computation. An approach based on space discretization was introduced in [POB*06]. The protein is sampled into three-dimensional grid. Each sample is evaluated with its distance from the nearest atom and the grid is processed by the Dijkstra's algorithm to find the best channel [Dij59]. The second approach uses basic structures of computational geometry to compute a channel [MBS07]. The Delaunay triangulation for a set of atom centers is computed using the QuickHull algorithm [BDH96]. The Delaunay triangulation is converted to a graph then. Nodes of this graph correspond to tetrahedra in the triangulation. An edge between two nodes exists if two adjacent tetrahedra share a face. The weight of an edge is equal to the size of the bottleneck sphere of this edge. Finally, the graph is processed by the Dijkstra's algorithm and the widest path leading from some specified point inside the molecule to its surface is located. Thus, regarding the conversion of the Delaunay triangulation to the weighted graph, channels could also be seen as a sequence of neighbouring tetrahedra. A very similar method based on Voronoi diagram was presented in [PKKO07]. A method proposed in [YFW*08, YH08] combines both these approaches by computing the Voronoi diagram not for atom centers but for points sampled on the surface of atoms.

For the computation of more channels in protein molecules there are several methods proposed in recent papers. A discretization method uses a large sphere to block the part of the channel at the surface of the molecule. In fact it disables all samples located in a sphere placed in the channel gorge so that they are not processed again. Methods based on computational geometry disable edges at the neighborhood of the bottleneck sphere or disable or penalize the weight of edges which are already processed. In [YFW*08], the issue of computing multiple channels is also addressed. The main difference against the other methods is that it is necessary to compute the values in the Dijkstra's algorithm for the whole molecule (in the other approaches the computation was terminated when the molecule surface was reached). Then, similar pathways are clustered into several main channels. These methods are described more thoroughly in the next section.

3. Possible approaches

Considering the channel computation based on computational geometry, there are two possibilities how to compute more channels in a protein molecule. When we compute a path in the graph constructed from the Delaunay triangulation, we could either stop the computation when we reach a node which is located on the surface of the molecule or continue in the computation and evaluate all nodes.

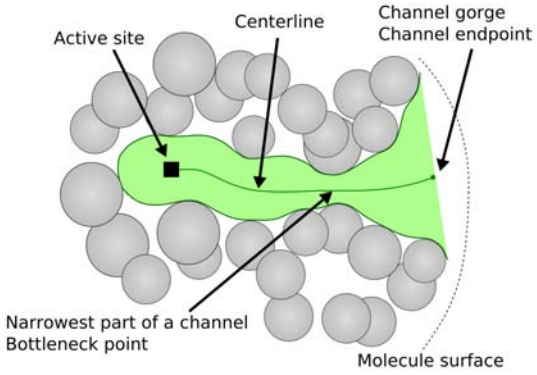


Figure 1: *Basic terms. Channel, its centerline and volume. The active site denoted by a square.*

3.1. Full evaluation of the Dijkstra's algorithm

Using the full evaluation we could determine for each node the ideal path from the node to the active site. Using some clustering algorithm we could identify the most significant channels. However, this approach suffers from many disadvantages. If the main criterion for the channel comparison is the radius of the bottleneck sphere, the typical behaviour of the Dijkstra's algorithm is that it utilizes the fact that the weight of edges at the surface is high due to the fact that the atom density near the surface is lower. Therefore, the best channel to any tetrahedra at the surface is the ideal channel which continues by traversing along the surface to the desired tetrahedra (see Fig. 2). This means, that all channels share the same bottleneck sphere and it is not possible to cluster them using this criterion. We could solve this problem by involving other criteria such as the length of the channel, however it is not possible to use the approach described in [MBS08]. The only possibility is to construct a function, which combines all these parameters into one value. However, if we use such a function as an evaluation function of the Dijkstra's algorithm, it is not guaranteed that we are able to find the ideal channel (according to the bottleneck sphere). Also the construction of this function could be complicated. Other disadvantage of this approach is a high time of the computation.

3.2. Partial evaluation of the Dijkstra's algorithm

As demonstrated in [MBS07], the computation time could be significantly decreased if we stop the progression of the Dijkstra's algorithm when we reach the surface of the protein. When we want to compute more channels in this situation, we have to modify the structure of the molecule appropriately after the computation of the ideal channel and repeat the whole process again. The two main concepts of the modification are as follows. We can either block the narrowest part of a channel or block the spherical neighbourhood of channel gorge near the molecule surface.

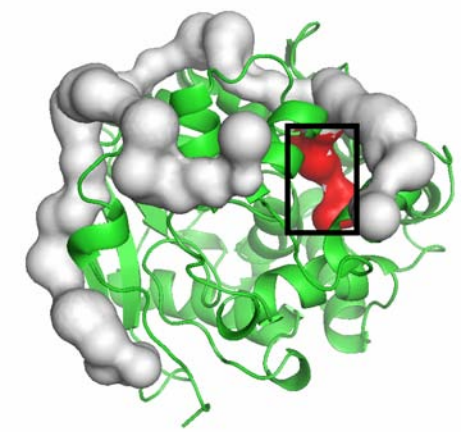


Figure 2: *Full Dijkstra's evaluation. Ideal channel (highlighted) and three randomly chosen channels. It can be seen that all these channels pass through the same locations as the ideal channel and continue along the molecule surface.*

When the molecule is blocked at the narrowest part of a channel, it is certain that the next computed channel will be narrower (assuming there is only one edge with this value in the molecule). In this case, all other channels passing through the blocked narrowest part are treated as the same channel and are omitted. Therefore, side branches of a channel that splits somewhere between the narrowest part and the molecule surface are not considered. However, if branches are biochemically relevant, chemists are interested in these side branches as well. In addition, when blocking the narrowest part, it is possible that the trajectory of the newly computed channel will be similar to the trajectory of the previously computed channel. Channels with similar trajectory may be identical and only representative of the set of identical channels should be reported.

The blocking of a channel near the molecule surface allows to recognize the branching of a channel. If we block locations near the channel gorge appropriately, all channels computed would be relevant. A simple approach was used for this purpose when a spherical neighbourhood of a channel endpoint was blocked. The parameter of the radius of this neighbourhood was set experimentally. However, this approach is not general and for some molecules identical channels were reported and on the other hand some channels were omitted as their gorge was blocked during the blocking of previous channels. This method is very sensitive to the arrangement of atoms forming the surface of the molecule.

The examples of these drawbacks on real protein molecules are presented in section 5.

The partial evaluation of the Dijkstra's algorithm seems to be more suitable for most situations.

4. Proposed solution: algorithm

The method that blocks the molecule near the channel gorge can be improved by involving adaptive evaluation of computed channels. We introduce a comparison of computed channels and identification of similar channels during the computation. For the computation of tunnels we use the approach described in [MBS07].

The process of computation of a next channel is divided into two phases. After a computation of the best channel, we have to block the molecule near the channel endpoint, so that we would not compute the same channel again. The blocking is accomplished by disabling edges in the graph obtained from the Delaunay triangulation or its dual, the Voronoi diagram. We call this step a blocking phase. After the blocking phase is performed, we compute the channel best in the actual situation and compare it with the last best channel to decide whether this two channels are so similar that we could consider them as the same channel. We call this phase a comparison phase. The comparison phase is not present in the previous methods where it is assumed that each newly computed channel is different. If the channels are identified as similar, we perform a further blocking of the original channel (and eventually the newly computed channel as well), so that the next computed channel would differ more significantly. On the other hand, if the channels are identified as different, we output the new channel as the next channel and perform the blocking phase only on this new channel. In general, if we block the channel that is not similar to the last best channel, we call the blocking phase an initial blocking phase. This allows the initial blocking phase to be different from the main blocking phase. For instance, in this case the initial blocking may be accomplished by using larger sphere than the main blocking. The smaller sphere assures that not large region will be blocked if there are many identical channels. The whole process is summarized in the algorithm in Fig. 3.

Possible implementations of the comparison and blocking phase are discussed in following sections.

4.1. Comparison phase

The key step is to compare channels and evaluate their match. The match of 0.0 would indicate that channels are completely different whereas the match of 1.0 would state that channels are identical. For channel matching we may use several metrics. More detail on metrics used is provided below. An additional parameter of threshold has to be set. Channels whose match is below the threshold are considered to be different. We expect this parameter to be determined via empirical testing on real protein molecules.

We propose the following metrics for the evaluation of the match of two channels:

- **Atom match.** Each atom has associated a unique identification number. This is why we can determine which

```
INPUT: Molecule m, int numTunnels
OUTPUT: reported tunnels
/* compute first tunnel */
Tunnel t = compute_tunnel(m);
initialBlock(m,t);        B
output (t);  /* report first tunnel */
for (i = 1; i < numTunnels; )
{
   previous = t;
   t = compute_tunnel(m);
   /* match to previous tunnel */
   if  (t.match(previous) > threshold)   C
         mainBlock(m,previous,t);   B
   else /* tunnels different */
   {
      initialBlock(m,t);   B
      output (t);
      i++;
   }
}
```

Figure 3: *The algorithm. Blocking phase (B) and comparison phase (C) highlighted. For more detail on* `compute_tunnel(m)` *see [MBS07].*

atoms are located near each channel centerline. These atoms can be considered as the atoms forming the channel. The match is computed as a number of atoms that are located around both channels divided by the total number of atoms forming the larger of the two channels. The value represents the percentage of identical atoms shared by both channels.

- **Residue match.** Atoms form logical groups – so called residues. The match on the basis of percentage of identical residues can also be used. This approach minimizes the undesirable sensitivity of the atom-based approach mentioned above.
- **Length-distance match.** A ratio between the total length of a channel and the maximal distance between two tetrahedra that are shared by both considered channels. We would like to emphasize, that the sharing of two tetrahedra between channels does not necessarily mean that both of channels contain the same tetrahedra. We can also consider two tetrahedra whose distance is less than a specified value to be shared. The advantage of this metrics is that it also reflects the uniformity of sharing. If two channels share parts along the whole centerline, we would like to evaluate them with higher value than channels, which share the same percentage of parts but only in a small portion of the centerline. Notice that we always have to consider the first and the last tetrahedra in channels to be shared to obtain this behaviour.

4.2. Blocking phase

Depending on the fact if previously computed channels are needed for the blocking process we distinguish two main concepts of blocking phase.

- **Initial blocking.** This phase is performed on the first computed channel and on each channel that was identified as different from the previously computed channel. The previous channel is not required for this phase.
- **Main blocking.** This phase is applied when the previously computed channel was identified as identical to the currently computed channel. Both compared channels could be required.

The best method for the initial blocking seems to be blocking of the spherical neighbourhood of channel gorge, similar to the original approach. For main blocking we propose two alternatives.

The first is identical to the original method: we simply block the spherical neighbourhood of the molecule near channel gorge. Still, the number of blocked endpoints is large in some cases. Suppose there are many channels which have unimportant branching. Using the adaptive method, all endpoints of these branches have to be blocked before a different channel is found. First, the computation of these channels that are not important is wasting of the computation time. Second, if we block all endpoints of these branches, we block unacceptably large region around these branches. Imagine a channel that is different from all of these considered channels. If it leads through different parts of the molecule and its gorge is located near the blocked branches, this channel – although it may be important – would never be computed for the region near its gorge was blocked before (see Fig. 5, channel labeled B).

This issue can be solved by the second alternative. We perform the blocking only on the branches of identical channels. If a newly computed channel is identical with the last computed channel, we evaluate if there is some branching present. In the majority of cases, if the channels are similar, they have the same trajectory until the point where they branch and each of them reaches the surface in different gorge. We design the main blocking phase so that it would block both channels from the surface to their common branch-point (Fig. 4). By this blocking we assure that no channels passing through this branch-point will be computed. Using this strategy, all channels that should be identified as identical are excluded and due to the fact that we blocked only two branches of possible many branches, other different channels with the gorge near these branches can be computed (see Fig. 5, channel labeled B). Notice that if we would block only the branching point we could encounter the same difficulties as with blocking of the narrowest part.

However, there are other possibilities of how the blocking phase can be designed. We have considered the blocking of the percentage of distance along the channel trajectory (for

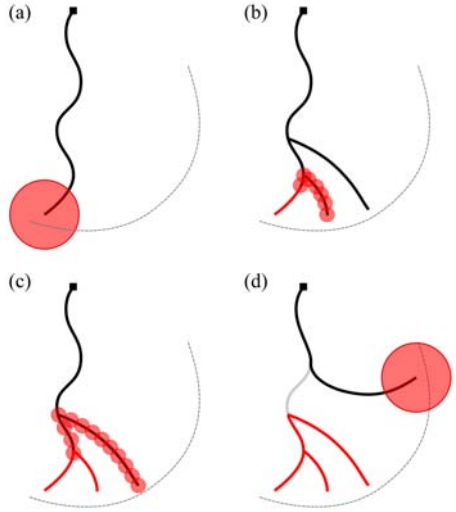


Figure 4: *The example of the branch blocking variant of the blocking method. (a) Initial blocking phase. (b), (c) Main blocking phase. (d) Tunnel classified as different – initial blocking phase.*

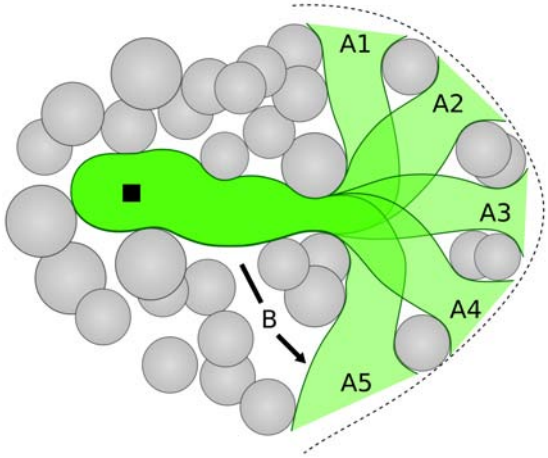


Figure 5: *If all branches A1 – A5 are blocked, the channel B will never be computed. On the other hand, if branches A1 and A2 are blocked from the surface till their branch-point, other branches A3 – A5 will not be computed nor blocked and in next step, channel B will be found.*

instance from the surface with the length of 30% length of the whole channel) or the decreasing value of the blocking sphere if nearer to the active site. These designs of blocking phase were not considered as substantial since their behaviour was similar to the simple spherical blocking in both initial and main blocking.

A complex comparison including examples on real data is presented in the following section.

All of the methods are dependent on the setting of radii of blocking spheres within. These radii define the "aggressiveness" of blocking. With the on-fly evaluation process the radius parameter could be set to a smaller value. In this case, when repeating the computation, we expect newly computed channels to be very similar to those previously computed as the blocking is not significant. After several steps we certainly find a channel that differs enough to be evaluated as different by our metric. With this evaluation we can increase the probability that we do not miss any important channel. It is essential that the number of temporary channels computed until a different channel is found is dependent on the setting of blocking radii within methods. The smaller is the value the more temporary channels will probably be computed. Therefore, the value has to be set carefully to keep the computation time acceptable. These parameters have to be set experimentally on the basis of massive testing on large data sets.

There is also a possibility to make the blocking more restrictive with each newly computed channel that is identified as identical with previous one. This may also help to save computational time. However, the increase in restrictiveness has to be carefully designed and bounded since we cannot block parts of a molecule where another possible channel may lead.

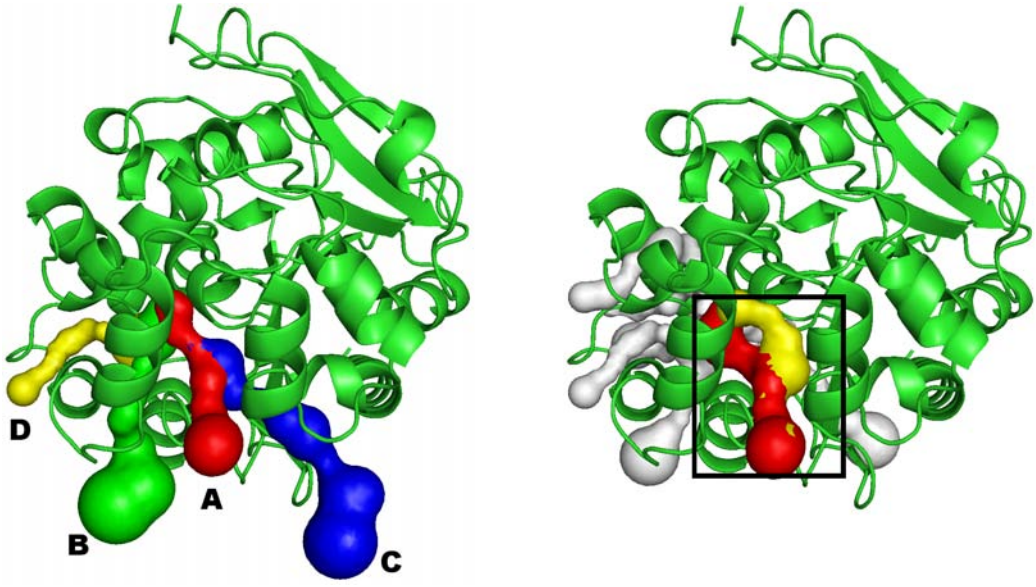
5. Results

The methods were tested on real protein molecules of haloalkane dehalogenases (on molecules with PDB codes 1b6g, 1mj5, 1edd, 2bfn). Biochemically relevant cavities inside these molecules were set as the active sites. Visualizations and screenshots were obtained using PyMOL software.

We compared the original solution with the blocking near the narrowest part of the channel (NP). We also tested both suggested version of main blocking phase: blocking of the channel gorge (G) and improvement by blocking channel branches (B). Computed channels were also compared against channels that were annotated by chemists. These annotated channels are certainly present in the molecule and are considered to be biochemically relevant.

The following example shows the computation of channels in the molecule 1b6g. Channels classified by chemists as A, B, C and D are depicted in Fig. 6 (a). Channels computed with NP blocking are visualized in Fig. 6 (b). It can be seen that many of these channels have similar trajectory and should be classified as identical. Notice that this method is not able to identify these channels as identical.

Both our new methods G and B have similar results (Fig. 7). However, it can be seen in Figure 7 and Table 1 that small differences are present. In most cases, the number of temporary channels which are identified as identical to the previous



(a) (b)

Figure 6: *Molecule 1b6g. (a) channels classified by chemists as A, B, C and D. (b) channels obtained by blocking the molecule near the narrowest part of the channel. Identical channels are highlighted.*

one are significantly lower in case of B blocking. Still, in a small number of cases some of the reported channels are visually identical. This is caused by the properties of the metric for comparing channels and setting the threshold for determining whether two channels are different or identical. The number of identical channels reported is not high and by using the B blocking the number of such unimportant channels even decreases (see Table 1). Except for one case, all channels annotated by chemists are reported by both of our methods. Due to the fact, that chemists annotated four present channels and we computed 10 different channels, there are also other channels. It is possible that after examination by chemists these channels could be also valuable.

6. Conclusion

We have reviewed existing methods for computation of more channels in protein molecules and presented their advantages and disadvantages. In addition, we proposed a general blocking scheme for computation of more channels in the protein molecules. The blocking is adaptive and results indicate that in typical protein molecules the most of computed channels are relevant and no important channels are excluded.

As for the future research, we intend to adjust this algorithm and blocking techniques to take additional biochemical parameters into account. Naturally, new possibilities of adjustment to be explored will arise from cooperation and discussion with biochemists. Fully automated blocking method is a great step towards all additional analyses, especially in the analysis of protein dynamics. Although it is not known yet whether it is possible to automatically find only biochemically relevant channels, the method we propose is an important step towards.

7. Acknowledgement

This work was supported by The Ministry of Education of The Czech Republic, Contract No. LC06008 and by The Grant Agency of The Czech Republic, Contract No. 201/07/0927.

References

[BDH96] BARBER C. B., DOBKIN D. P., HUHDANPAA H.: The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw. 22*, 4 (1996), 469–483.

[BS00] BRADY G. P., STOUTEN P. F.: Fast prediction and visualization of protein binding pockets with pass. *J Comput Aided Mol Des 14*, 4 (May 2000), 383–401.

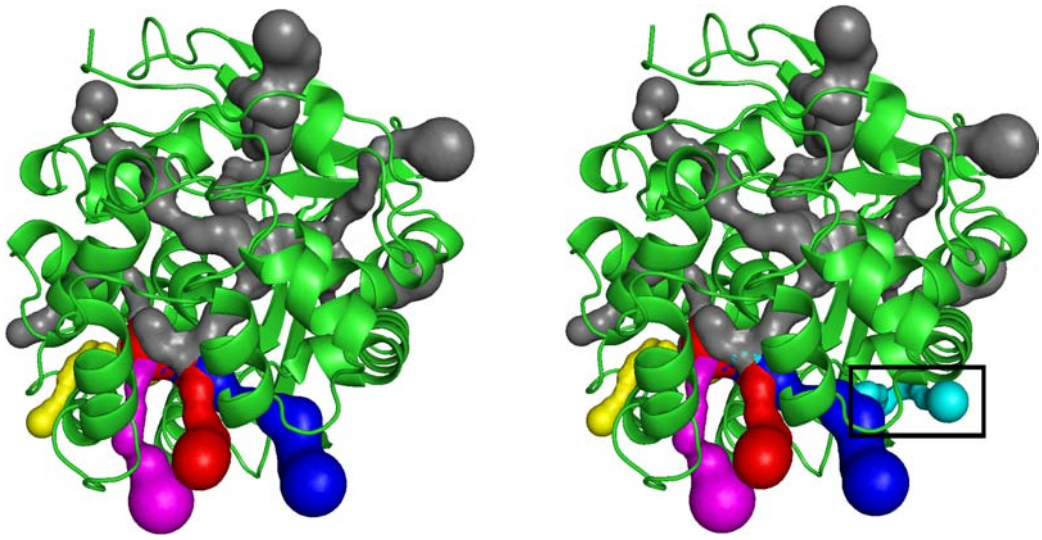
[Dij59] DIJKSTRA E. W.: A note on two problems in connexion with graphs. *Numerische Mathematik 1*, 1 (December 1959), 269–271.

[EM94] EDELSBRUNNER H., MÜCKE E. P.: Three-dimensional alpha shapes. *ACM Transactions on Graphics 13*, 1 (1994), 43–72.

[LEW98] LIANG J., EDELSBRUNNER H., WOODWARD

Table 1: *Comparison of adaptive blocking methods: near channel gorge (G) and blocking channel branches (B)*

Molecule	Number of channels to be computed	Temporary channels needed (blocking method G/B)	Identical channels reported (blocking method G/B)
1edd	10	18/9	3/0
2bfn	10	16/6	1/0
1b6g	10	24/15	1/1
1mj5	10	17/7	0/0
1bez	10	13/8	0/0
1bn7	10	18/9	1/0



(a) (b)

Figure 7: *Molecule 1b6g. (a) Spherical neighbourhood blocked near the channel gorge (b) Blocking of channel branches. It can be observed that there exists a channel that was blocked by other identical channels in case (b). However, the branch-blocking method was able to compute this particular channel (highlighted).*

C.: Anatomy of protein pockets and cavities: measurement of binding site geometry and implications for ligand design. *Protein Sci 7*, 9 (September 1998), 1884–1897.

[LJ05] LAURIE A. T., JACKSON R. M.: Q-sitefinder: an energy-based method for the prediction of protein-ligand binding sites. *Bioinformatics 21*, 9 (May 2005), 1908–1916.

[MBS07] MEDEK P., BENES P., SOCHOR J.: Computation of tunnels in protein molecules using delaunay triangulation. *Journal of WSCG 15*, 1–3 (2007), 107–114.

[MBS08] MEDEK P., BENES P., SOCHOR J.: Multicriteria tunnel computation. *Proceedings IASTED International Conference on Computer Graphics and Imaging (CGIM)* (2008).

[PKKO07] PETŘEK M., KOŠINOVÁ P., KOČA J., OTYEPKA M.: Mole: A voronoi diagram-based explorer of molecular channels, pores, and tunnels. *Structure 15*, 11 (November 2007), 1357–1363.

[POB*06] PETREK M., OTYEPKA M., BANAS P., KOSINOVA P., KOCA J., DAMBORSKY J.: Caver: a new tool to explore routes from protein clefts, pockets and cavities. *BMC Bioinformatics 7* (June 2006), 316+.

[PS85] PREPARATA F. P., SHAMOS M. I.: *Computational Geometry: An Introduction (Monographs in Computer Science)*. Springer, August 1985.

[YFW*08] YAFFE E., FISHELOVITCH D., WOLFSON H. J., HALPERIN D., NUSSINOV R.: Molaxis: Efficient and accurate identification of channels in macromolecules. *Proteins* (2008).

[YH08] YAFFE E., HALPERIN D.: Approximating the pathway axis and the persistence diagram of a collection of balls in 3-space. In *SCG '08: Proceedings of the twenty-fourth annual symposium on Computational geometry* (New York, NY, USA, 2008), ACM, pp. 260–269.