Computer Physics Communications 190 (2015) 224–227

Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

CAVE-CL: An OpenCL version of the package for detection and quantitative analysis of internal cavities in a system of overlapping balls: Application to proteins

Ján Buša Jr.[a,b], Ján Buša[b], Shura Hayryan[a], Chin-Kun Hu[a,*], Ming-Chya Wu[a,c]

[a] Institute of Physics, Academia Sinica, Nankang, Taipei 11529, Taiwan

[b] Faculty of Electrical Engineering and Informatics, Technical University of Košice, 040 01 Košice, Slovak Republic

[c] Research Center for Adaptive Data Analysis, National Central University, Chungli 32001, Taiwan

A R T I C L E I N F O

Article history:
Received 8 December 2014
Accepted 23 December 2014
Available online 7 January 2015

Keywords:
Protein
Cavity
OpenCL

A B S T R A C T

Here we present the revised and newly rewritten version of our earlier published CAVE package (Buša et al., 2010) which was originally written in FORTRAN. The package has been rewritten in C language, the algorithm has been parallelized and implemented using OpenCL. This makes the program convenient to run on platforms with Graphical Processing Units (GPUs). Improvements include also some modifications/optimizations of the original algorithm. A considerable improvement in the performance of the code has been achieved. A new tool called `input_structure` has been added which helps the user to make the data input and conversion more easier and universal.

New version program summary

Program Title: CAVE-CL, CAVE C

Catalogue identifier: AEHC_v2_0

Program summary URL: http://cpc.cs.qub.ac.uk/summaries/aehc_v2_0.html

Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland

Licensing provisions: Standard CPC license, http://cpc.cs.qub.ac.uk/licence/licence.html

No. of lines in distributed program, including test data, etc.: 32646

No. of bytes in distributed program, including test data, etc.: 444248

Distribution format: tar.gz

Programming language: C, C++, OpenCL.

Computer: PC with GPU.

Operating system: OpenCL compatible systems.

Has the code been vectorized or parallelized?: Parallelized using GPUs. A revised serial version (non GPU) is included in the package as well.

Keywords: Proteins, Solvent accessible area, Excluded volume, Cavities, Analytic method, Stereographic projection, GPGPU, OpenCL.

PACS: 82.20.Wt, 02.60.Cb, 02.70.Ns.

Classification: 16.1.

Catalogue identifier of previous version: AEHC_v1_0.

Journal reference of previous version: Comput. Phys. Commun. 181 (2010) 2116.

Does the new version supersede the previous version?: Yes

Nature of problem: Molecular structure analysis.

* Corresponding author.
E-mail addresses: jan.busa.2@tuke.sk (J. Buša Jr.), jan.busa@tuke.sk (J. Buša), shura@phys.sinica.edu.tw (S. Hayryan), huck@phys.sinica.edu.tw (C.-K. Hu), mcwu@ncu.edu.tw (M.-C. Wu).

http://dx.doi.org/10.1016/j.cpc.2014.12.017
0010-4655/© 2015 Elsevier B.V. All rights reserved.

Solution method:
Analytical method, which uses the stereographic transformation for exact detection of internal cavities in the system of overlapping balls and numerical algorithm for calculation of the volume and the surface area of cavities.

Reasons for the new version:

This work is in line with our global efforts to modernize the protein structure related algorithms and software packages developed in our research group during last several years [1–8]. These tools are keeping to receive considerable attention from researches and they have been used in solving many interesting research problems [9,10]. Among many others, one important application has been found by the members of our team [11].

Therefore, we think that there is a demand to revise and modernize these tools and to make them more efficient. Here we follow the approach used earlier in [8] to develop a new version of the CAVE package [7]. The original CAVE package was written in FORTRAN language. One of the reasons for the new version is to rewrite it in C in order to make it more friendly to the young researchers who are not familiar with FORTRAN. Another, a more important reason, is to use the possibilities of the contemporary hardware (for example, the modern graphical cards) to improve the performance of the package. We also want to allow the user to avoid the re-compiling of the program for every molecule during multiple calculations of the array of molecules. For this purpose we are providing the possibility to use general `pdb` files as an input. After compiling one time, the program can receive any number of input files successively. Also, we found it necessary to go through the algorithm and to optimize, where it is possible, the memory usage and to make the algorithm more efficient.

Summary of revisions:

1. **Memory usage and language.** The whole code has been ported into C and the static arrays have been replaced with dynamic memory allocation. This allows to load and handle the proteins of arbitrary size.
2. **Changes in the algorithm.** Like in [8], the original method of *North Pole test and molecule rotation* [4] has been changed. The details of implementation and the benefits from this change are properly described in [8] and we find it not necessary to repeat it here.
3. **New tool.** A module called `input_structure` which takes as an input a protein structure file in the format compatible with Protein Data Bank (`pdb`) [12] has been adopted from [8]. Using external tool allows users to create their own mappings of atoms and radii without re-compiling the module `input_structure` itself or the CAVE.

   It is the user's responsibility to assign proper radii to each type of atoms. One can use any of the published standard sets of radii (see for example, [13–17]). Alternatively, the user can assign his own values for radii immediately in the module `input_structure`. The radii are assigned in a special file with extension `pds` (see the documentation) which consists of lines like this: `ATOM CA ALA 2.0` which is read as "the C_α atom of Alanine has radius 2.0 Å".
4. **Some computational tricks.** In several parts of the program square roots were replaced by second powers and calls of sin and cos functions were replaced by calls to sincos allowing for further speed-up (in comparison to original FORTRAN version).

   The typical value of the relative error between results obtained by original (FORTRAN), C, and OpenCL versions was between 10^{-8} and 10^{-10} and it never exceeded 10^{-5}. Small differences in results can be due to the implementation of compiler and specially in case of OpenCL also in the implementation of arithmetic by the GPU vendor.
5. **OpenCL implementation and testing results.** OpenCL [18] is an open standard for parallel programming in heterogeneous systems. It is becoming increasingly popular and has proved to be an efficient tool for computations in different fields (see, for example, the most recent [19,20] and the references therein).

   Table 1 shows the speedup of the C and OpenCL implementations of CAVE as compared to the FORTRAN version. We compare both results obtained using free GNU FORTRAN (g77) and commercial (and faster) ifort. Speedup is calculated as a ratio between the original time obtained by FORTRAN and C or OpenCL version of program. Times of execution are measured in seconds.

   One could expect greater speed-ups but the problem is that not the whole algorithm could be parallelized. Only about 1/3 of the whole program was parallelized and the effect of this is visible for the proteins with 2000 atoms and more if the calculation time of FORTRAN version is higher than approximately 10 s. The rest of the code is sequential and its parallelization will require entirely new algorithm which might be the future work. Fig. 1 shows the speed-up as a function of number of neighbors. This clearly indicates, that the effect of parallelization is stronger for proteins with many neighbors. This is also the reason, why the effect is not so strong for proteins with 0 testing sphere radius. Most of the cavities in such case are enclosed only in few (around 4–8) spheres, while in the case of 1.2 testing sphere radius we have easily 35 or more enclosing spheres.

   In global, we can see that C version is a good choice for general proteins (and testing sphere radius of 0), OpenCL is proper for larger proteins and larger computational times. 0 in the name of protein means that no probe radius has been added to the atomic radii. In other cases 1.2 Å was added to all atomic radii.

Table 1
Speed-up of C and OpenCL versions of the program CAVE when compared to the original (FORTRAN) version calculated using GNU Fortran (gfort) and Intel® FORTRAN (ifort).

Protein PDB ID	Number of atoms	Time gfort (s)	Speed-up ifort	Speed-up C	Speed-up OpenCL
1AUW0	13 872	345.98	1.56	1.94	1.97
1AUW	13 872	235.10	1.70	1.75	3.32
1DJ30	6 580	61.37	1.55	1.95	1.97
1DJ3	6 580	68.80	1.81	1.70	3.12
1I0A0	13 675	358.88	1.57	1.97	1.97
1I0A	13 675	254.81	1.67	1.85	3.44
1LD40	15 648	305.43	1.55	1.87	1.91
1LD4	15 648	551.28	1.58	1.47	2.53
1M3Z0	11 292	181.31	1.54	1.92	1.95
1M3Z	11 292	141.04	1.75	1.74	3.06
1OJX0	19 368	503.02	1.62	1.90	1.90
1OJX	19 368	339.28	1.69	1.60	3.12
1OK40	19 358	541.84	1.67	1.92	1.93
1OK4	19 358	414.96	2.32	2.39	4.10
1QI60	14 384	292.94	1.55	1.90	1.93
1QI6	14 384	237.80	1.67	1.82	3.15
1QNW0	7 168	72.24	1.55	1.90	1.93
1QNW	7 168	71.35	1.85	1.67	3.11
1S3Q0	15 358	376.79	1.57	1.95	1.98
1S3Q	15 358	349.48	1.76	2.03	3.27
1UPA0	16 272	390.91	1.55	1.90	1.93
1UPA	16 272	289.55	1.67	1.84	3.09
1YLO0	15 318	306.27	1.55	1.88	1.92
1YLO	15 318	326.21	1.61	1.90	3.14
2MYS0	6 287	51.37	1.55	1.98	1.97
2MYS	6 287	62.21	1.78	1.72	2.65

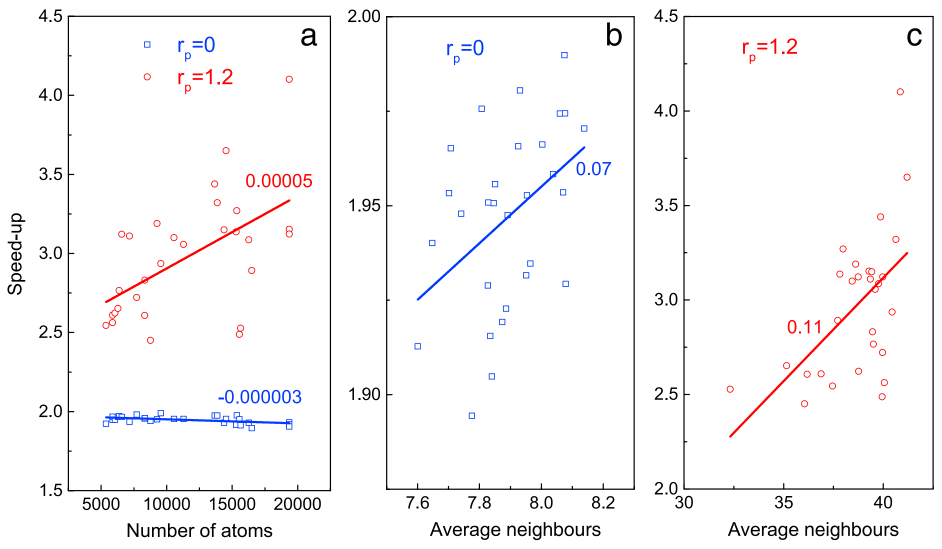

Fig. 1. Dependence of speed-up of OpenCL CAVE on (a) the number of atoms, (b) the number of neighbors for testing sphere radius $r_p = 0$, and (c) $r_p = 1.2$. The numbers indicate the slopes of linear fits.

All results were obtained on computer with Intel Core 2 Duo E8500 CPU running at 3.16 GHz with 4 GB RAM and GPU NVIDIA GTX470 and computer with Intel Xeon X5450 CPU running at 3.00 GHz with 32 GB RAM and dedicated NVIDIA C1060 GPU card.

When considering which GPU to use, it is important to watch its double precision performance. Consumer oriented GPUs have usually intentionally decreased double precision performance and because of that results can be similar even if newer generation of GPUs is used. For instance in 2010 the performance in double precision of NVIDIA GPUs (except for highly specialized GPUs for scientific computing) was 1/8 of the performance in single precision. Nowadays (2014) this ratio is 1/24, meaning that GPUs from 2010 are as fast as current GPUs (except for special editions of GPUs or dedicated cards).

Restrictions: None

Running time:
Depends on the size of the molecule under consideration. All test examples run under 1 min, usually under 30 s.

The work was supported by Grants MOST 103-2120-M-001-005.

References:

[1] S. Hayryan, C.-K. Hu, S.-Y. Hu, R.-J. Shang, J. Comput. Chem. 22 (2001) 1287.
[2] F. Eisenmenger, U.H.E. Hansmann, S. Hayryan, C.-K. Hu, Comput. Phys. Commun. 138 (2001) 192.
[3] F. Eisenmenger, U.H.E. Hansmann, S. Hayryan, C.-K. Hu, Comput. Phys. Commun. 174 (2006) 422.
[4] S. Hayryan, C.-K. Hu, J. Skřivánek, E. Hayryan, I. Pokorný, J. Comput. Chem. 26 (2005) 334.
[5] J. Buša, J. Džurina, E. Hayryan, S. Hayryan, C.-K. Hu, J. Plavka, I. Pokorný, J. Skřivánek, M.-C. Wu, Comput. Phys. Commun. 165 (2005) 59.
[6] J. Buša, S. Hayryan, C.-K. Hu, J. Skřivánek, M.-C. Wu, J. Comput. Chem. 30 (2009) 346.
[7] J. Buša, S. Hayryan, M.-C. Wu, J. Skřivánek, C.-K. Hu, Comput. Phys. Commun. 181 (2010) 2116.
[8] J. Buša Jr., S. Hayryan, M.-C. Wu, J. Buša, and C.-K. Hu, Comp. Phys. Comm. 183 (2012) 2494-2497.
[9] H. L. Chen, et al., Proteins: Structure, Function, and Bioinformatics 78 (2010) 2973.
[10] P. Kota, et al., Bioinformatics 27 (2011) 2209-2215.
[11] M.-C. Wu, M. S. Li, W.-J. Ma, M. Kouza, C.-K. Hu, EPL 96 (2011) 68005.
[12] http://www.rcsb.org.
[13] B. Lee, F. M. Richards, J. Mol. Biol. 55 (1971) 379.
[14] F. M. Richards, Annu. Rev. Bipohys. Bioeng. 6 (1977) 151.
[15] A. Shrake, J. A. Rupley, J. Mol. Biol. 79 (1973) 351.
[16] A. A. Rashin, M. Iofin, B. Honig, Biochemistry 25 (1986) 3619.
[17] C. Chotia, Nature 248 (1974) 338.
[18] http://www.khronos.org/opencl/.
[19] M. Molero-Armenta, U. Iturraran-Viveros, S. Aparicio, et al., Comp. Phys. Commun. 185 (2014) 2683.
[20] M. Bach, V. Lindenstruth, O. Philipsen, et al., Comp. Phys. Commun. 184 (2013) 2042.