Computer Physics Communications 184 (2013) 2856–2859

Contents lists available at ScienceDirect

Computer Physics Communications

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

g_contacts: Fast contact search in bio-molecular ensemble data[☆,☆☆]

Christian Blau, Helmut Grubmuller[*]

Max Planck Institute for Biophysical Chemistry, Am Fassberg 11, 37077 Göttingen, Germany

ARTICLE INFO

Article history:
Received 6 June 2013
Received in revised form
1 July 2013
Accepted 13 July 2013
Available online 29 July 2013

Keywords:
Range search
Molecular dynamics
Decomposition scheme
Gromacs

ABSTRACT

Short-range interatomic interactions govern many bio-molecular processes. Therefore, identifying close interaction partners in ensemble data is an essential task in structural biology and computational biophysics. A contact search can be cast as a typical range search problem for which efficient algorithms have been developed. However, none of those has yet been adapted to the context of macromolecular ensembles, particularly in a molecular dynamics (MD) framework. Here a set-decomposition algorithm is implemented which detects all contacting atoms or residues in maximum $O(N\log(N))$ run-time, in contrast to the $O(N^2)$ complexity of a brute-force approach.

Program summary

Program title: g_contacts
Catalogue identifier: AEQA_v1_0
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AEQA_v1_0.html
Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland
Licensing provisions: Standard CPC licence, http://cpc.cs.qub.ac.uk/licence/licence.html
No. of lines in distributed program, including test data, etc.: 8945
No. of bytes in distributed program, including test data, etc.: 981604
Distribution format: tar.gz
Programming language: C99.
Computer: PC.
Operating system: Linux.
RAM: ≈Size of input frame
Classification: 3, 4.14.
External routines: Gromacs 4.6[1]
Nature of problem: Finding atoms or residues that are closer to one another than a given cut-off.
Solution method: Excluding distant atoms from distance calculations by decomposing the given set of atoms into disjoint subsets.
Running time: $\leq O(N\log(N))$
References:
[1] S. Pronk, S. Pall, R. Schulz, P. Larsson, P. Bjelkmar, R. Apostolov, M. R. Shirts, J.C. Smith, P. M. Kasson, D. van der Spoel, B. Hess and Erik Lindahl, Gromacs 4.5: a high-throughput and highly parallel open source molecular simulation toolkit, Bioinformatics 29 (7) (2013).

1. Introduction

Molecular dynamics (MD) integrators allow simulations of large bio-molecular systems comprising millions of atoms on nanosecond to millisecond time scales [1,2]. These simulations produce a substantial amount of trajectory data, which typically consist of 10^4–10^6 structure "snapshots" (frames). The computational effort to generate the trajectory data scales with $O(N\log(N))$, where N is the number of simulated particles. Efficient analysis tools to extract certain observables from these data are required that exhibit a comparable scaling behavior to the algorithms that generate the trajectory data.

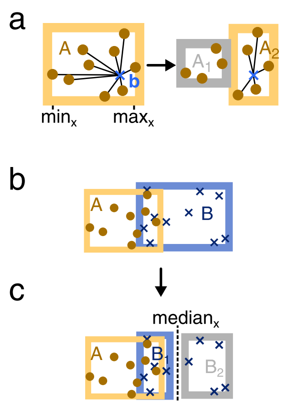
Identifying all atoms of a solute molecule which interact with the solvent, or all close atoms from different subunits of a molecular complex, is a recurring task. From a computational perspective, these tasks require one to identify all pairs of atoms that are closer to one another than a defined minimum contact distance. This task has been described as a spherical range search problem [3].

[☆] This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

[☆☆] This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

[*] Corresponding author.
E-mail addresses: cblau@gwdg.de (C. Blau), hgrubmu@gwdg.de (H. Grubmuller).

0010-4655/$ – see front matter © 2013 The Authors. Published by Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.cpc.2013.07.018



d
```
read sets of atoms A and B
for A and B
    create bounding box (bbox)
if bboxes of A and B overlap
    make A,B neighbour sets
else
    stop
while any set contains > $n_{min}$ atoms
    split subset  P with > $n_{min}$ atoms
    for subsets $P_1$, $P_2$ of P
        create bbox
        for all neighbour sets N of P
            if bboxes of N and P overlap
                store neighbour pair
for all leftover sets P
    for all neighbour sets N of P
        for all pairs of atoms (i,j) in N and P
            if atom distance < d
                store atom indices (i,j)
stop
```

Fig. 1. *Decomposition reduces the number of distance calculations during the contact search between two sets of atoms.* (a) The number of required distance calculations (black lines) is reduced by decomposing the set of atoms A (left) into disjoint subsets A_1, A_2 (right). The distances to $\mathbf{b}$ are only calculated if $\mathbf{b}$ lies within the bounding box of the atoms in A_i or is closer to the bounding box than the minimum contact distance d, indicated by the width of the yellow and gray frames. (b) Generalization of (a) to an arbitrary set B. Another bounding box of the set B is determined and checked for overlap with A. (c) In the next step, the biggest remaining subset is split along the median. The resulting subsets of B are again checked for overlap with A. (d) The implemented set-decomposition scheme given in pseudo-code.

Here, we describe the efficient implementation of an algorithm to obtain contacting atom pairs of two sets of atoms and respective trajectory contact frequencies. The modified k-dimensional tree approach employed has a worst-case run-time of $\propto O(N\log(N))$ for two sets of size N compared with a run-time $\propto O(N^2)$ of a brute-force approach. This high efficiency is achieved by excluding sets of distant atoms from the distance calculation. Combined with the excellent scaling properties of the method on parallel machines, this advantage will be particularly pronounced in future exascale computing applications.

The routine is implemented within GROMACS [4]. Due to the versatile implementation, it can also be applied to other three-dimensional contact searches. Extension to higher dimensions is straightforward.

2. Methods

2.1. Task

Given two sets of labeled atoms, $A = \{\mathbf{a}_i\}$ and $B = \{\mathbf{b}_j\}$, and a minimum contact distance d, the task of the algorithm described here is to identify all contacting atom pairs, i.e., all pairs of atom indices $\{(i,j)\}$ with $\|\mathbf{a}_i - \mathbf{b}_j\| < d$. A brute-force approach would require the calculation of the Euclidean distance between all possible pairs of atoms. The set decomposition scheme implemented here drastically reduces the number of necessary distance calculations and therefore the run-time.

2.2. Algorithm

For simplicity of presentation, we first assume the special case where one of the two sets, B, contains only one atom $\mathbf{b}$ (Fig. 1(a)). This case will subsequently be generalized to arbitrary sets A, B (Fig. 1(b), (c)).

As a first step, the minimum bounding box (bbox, yellow) of set A with sides aligned to the x, y, and z axes is determined. If the distance of $\mathbf{b}$ to the box boundary along the direction of the three coordinates exceeds a given contact distance d, $\mathbf{b}$ is not in contact with A, and the contact search terminates. Otherwise, A is decomposed into two subsets $A = A_1 \cup A_2$ [3]. If the distance of the two child bboxes (gray, yellow) of subset A_1 to $\mathbf{b}$ or A_2 to $\mathbf{b}$ exceeds d, the respective subset is discarded. Alternatively, the child bbox is further decomposed into two disjoint subsets (not shown), and so on. Decomposition is terminated when all subsets contain less than a given minimum number of atoms. As a final step, the distances of $\mathbf{b}$ to all atoms in the remaining subsets are determined, and the indices i are stored for which $\|\mathbf{a}_i - \mathbf{b}\| < d$.

Fig. 1(b) generalizes the above decomposition procedure to a set B comprising more than one atom. In this case, the bbox is also determined for B (blue), and B is also recursively split into subsets. For each subset B_q, all sets A_p overlapping with B_q are stored. When the decomposition terminates, only the distances for atom pairs i, j in stored pairs of sets A_p, B_q need to be calculated.

2.3. Application to ensemble data

The algorithm is applied to each frame of a given trajectory. Atom pair contacts are counted each frame. From these, the contact frequency is calculated by dividing the contact count by the number of frames analyzed. In addition to atom contact frequencies, residue contact frequencies are determined by defining two residues to be in contact if any of their respective atoms are in contact.

2.4. Efficiency

Four particular properties of the implemented algorithm render it efficient. First, subsets are decomposed along the median atom coordinates, which allows for the application of the median sort algorithm [5] such that the number of atoms in each subset is balanced in minimum run-time. Second, the order of atoms is kept from the previously analyzed frame. Thus the sorting effort is reduced if similar frames are analyzed. Third, after splitting a set A_p into subsets A_{p_1}, A_{p_2}, overlap with subsets $B_q \subset B$ only needs to be checked if B_q overlapped with A_p in the previous step, thus saving a large fraction of overlap checks for newly generated sets. Fourth, decomposition is stopped as soon as the brute-force approach to identify contacts between subsets A_p, B_q becomes on average more efficient than further decomposition at an empirically determined upper boundary for the minimum set size $n_{\min}$.

3. Software structure

The contact search algorithm described here is implemented in C99. It uses the GROMACS application programming interface (API) provided with the MD package GROMACS 4.6 [4].

4. Run description

4.1. Data input

Input arguments are trajectory file names (flagged `-f`), a GROMACS index file name that contains two index groups specifying each set of atoms (`-n`), a floating-point number that holds the minimum contact distance in nm (`-d`, by default $d = 0.3$ nm), and the threshold for the largest number of atoms in any node (`-bsize`). If the option (`-resndx`) is chosen, a GROMACS structure file (`-s`) is read.

4.2. Optional switches

The optional switch `-nopbc` ignores periodic boundary conditions, speeding up the calculation; `-resndx` calculates the contacts between two groups of residues instead of two groups of atoms.

4.3. Data output

Contact frequencies are written to an output file (name given in `-o`). If the flag `-resndx` is set, an additional index file (name given in `-on`) is written, which contains one index group for each contacting residue and its atom indices.

4.4. Example runs

We performed example runs on a typical test case as well as on a worst-case scenario.

A typical case is provided by a simulation of adenosine triphosphate (ATP) molecules in solution which bind to ribonucleic acid (RNA) [6]. We used an MD simulation of a solvated RNA molecule comprising 1166 atoms and two ATP molecules in solution comprising 86 atoms. The atom pairs and contact frequencies of RNA and ATP that are closer than $d = 0.3$ nm were determined for 20 000 frames of the simulation. The index-file reads:

```
[ RNA ]
1 2 3 4 5 6 7 8 9 10 11 12 13
...
1161 1162 1163 1164 1165 1166
[ ATP ]
1167 1168 1169 1170 1171 1172
...
1248 1249 1250 1251 1252 1253   .
```

The command to analyze the given trajectory `traj.xtc` is:

```
g_contacts -f traj.xtc -n index.ndx  .
```

A worst-case scenario is provided by two highly overlapping sets of atoms, where many set decompositions are required, and only a few subsets can be excluded from the contact search. The example considered here is a contact search in trajectories of a 1 ns simulation of TIP3P water in a periodic cubic water box of 5, 6, 7, 8 and 9 nm length (i.e., 12 426, 21 483, 34 251, 51 393, and 72 768 atoms, respectively), which were screened for contacts between sets of N consecutively labeled atoms. Contacts were searched between $N = 1, 21, 41, \ldots, 981$ atoms. The default distance cut-off of $d = 0.3$ nm was applied.

The respective index-file for $N = 21$ and a 12 426 atom simulation reads:

```
[ group_1 ]
1 2 3 4 5 6 7 8 9 10 11 12 13
14 15 16 17 18 19 20 21
[ group_2 ]
6213 6214  6215 6216 6217 6218
...
6228 6229 6230 6231 6232 6233    .
```

The analysis of the trajectory stored in `traj.xtc` was performed issuing the following command:

```
g_contacts -f traj.xtc -n index.ndx  .
```

The residue-based contact search determines contacts between water molecules, and was performed using

```
g_contacts -s traj.gro -f traj.xtc -resndx  .
```

5. Comparison with other methods

To compare the set-decomposition algorithm and the brute-force approach, the CPU clock cycles were counted that were required for the respective contact search and the storage of the contacts, excluding trajectory-file reading routines.

For the system containing ATP and RNA, the required CPU cycles for contact search and storage were recorded for 101 analyzed frames, which were analyzed every 0.2 ns in a 20 ns trajectory. A speed-up of 9.1-fold was obtained for the implementation of the set decomposition algorithm over the brute-force approach.

In the water box simulation, the required CPU cycles were averaged over the analysis of 45 frames each. Fig. 2 illustrates that our approach exhibits the expected $N\log(N)$ scaling, even in this

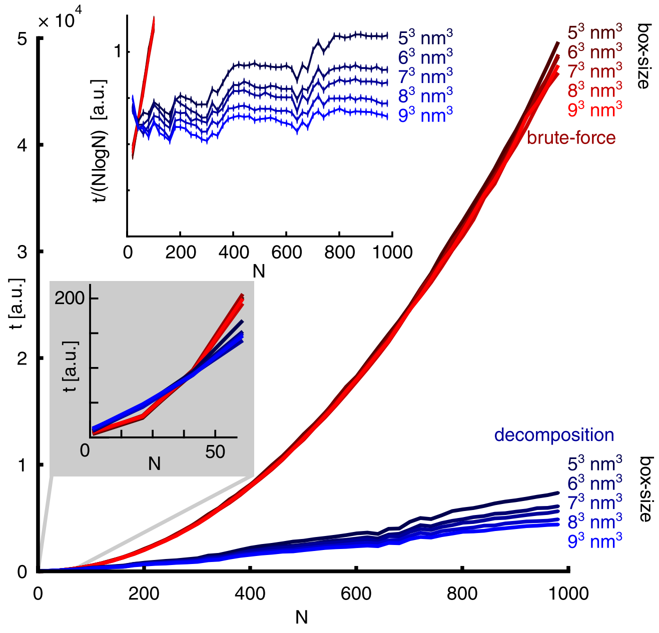


Fig. 2. *Scaling for the set-decomposition approach versus a brute-force contact search between two sets of* N *atoms for a cubic simulation box of* 5 nm, 6 nm, ..., 9 nm *and* $d = 0.3$ *nm containing water molecules.* The number of CPU cycles (rescaled) used exhibits an $N\log(N)$ scaling for the set-decomposition scheme (blue), while the brute-force algorithm (red) scales with N^2. The gray inset shows the "cross-over" region magnified. The inset above shows the number of CPU cycles used normalized to $N\log(N)$.

worst-case scenario. In contrast, the brute-force approach scales quadratically. For smaller boxes, a deviation from the ideal scaling behavior is observed. We attribute this deviation from $N\log(N)$ scaling to the fact that the number of contacts increases with decreasing box-size. The sorted list of lists approach employed for book-keeping of the contact pairs found results in a slightly worse overall complexity than $N\log(N)$ when very many contacts are found. Further, set decomposition works more efficiently if the subsets are less likely to overlap, which is the case for larger water boxes, explaining the different scaling offsets.

In the current implementation, the "cross-over" in efficiency between the set-decomposition algorithm and the brute-force algorithm (Fig. 2) is seen at set sizes of ≈40 atoms each, where the set-decomposition algorithm becomes faster. The set-decomposition algorithm reaches a speed gain of about ten-fold at ≈900 atoms per set.

Acknowledgments

We thank Benjamin von Ardenne, Andrea Vaiana, Lars Bock, and Petra Kellers for reading the manuscript. This work has been supported by the International Max-Planck Research School for Physics of Biological and Complex Systems and the DFG, grants FOR-1805 and EXC 171/A1. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

[1] M. Zink, H. Grubmuller, Mechanical properties of the icosahedral shell of southern bean mosaic virus: a molecular dynamics study, Biophysical Journal 96 (4) (2009) 1350–1363.
[2] J. Klepeis, K.L. Larsen, R. Dror, D. Shaw, Long-timescale molecular dynamics simulations of protein structure and function, Current Opinion in Structural Biology 19 (2) (2009) 120–127.
[3] J. Erickson, P. Agarwal, Geometric range searching and its relatives. advances in discrete and computational geometry, Contemporary Mathematics 223 (1999) 1–56.

[4] S. Pronk, S. Pall, R. Schulz, P. Larsson, P. Bjelkmar, R. Apostolov, M.R. Shirts, J.C. Smith, P.M. Kasson, D.V.D. Spoel, B. Hess, E. Lindahl, Gromacs 4.5: a high-throughput and highly parallel open source molecular simulation toolkit, Bioinformatics 29 (7) (2013) 845–854.

[5] D. Knuth, The Art of Computer Programming, Vol. 3: Sorting and Searching, Addison-Wesley, Reading, MA, 1973.

[6] T. Dieckmann, E. Suzuki, J. Feigon, G.K. Nakamura, Solution structure of an atp-binding rna aptamer reveals a novel fold, RNA 2 (7) (1996) 628–640.