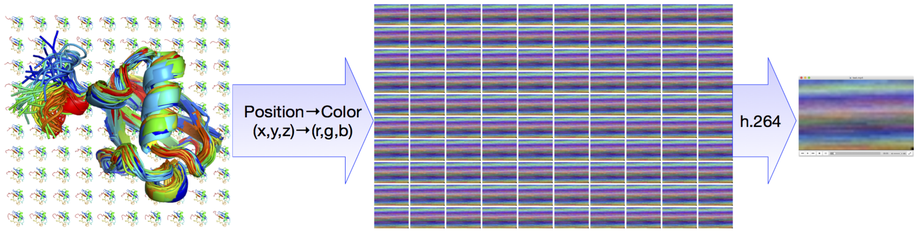
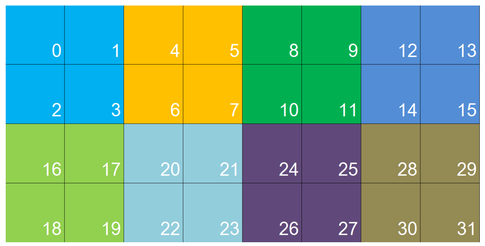
Data Compression of Molecular Dynamics Trajectories

Haiyang Huang **Prof. David Koes**

First Experiences in Research, Dietrich School of Arts & Sciences, University of Pittsburgh



Main idea of the Project



The idea of Blocking

Introduction

Molecular dynamics simulation is a valuable tool for understanding the dynamics and function of proteins, nucleic acids, and small molecules. As computational power increases, our ability to simulate longer and longer time scales has correspondingly increased, and the resulting trajectory files are gigabytes to terabytes in size. However, no application-specific compression algorithms have been created for these files and general-purpose compression tools (e.g. gzip) perform poorly. This project focuses on creating a compression and decompression algorithm for the molecular dynamics (MD) trajectories, which simulate the motions of biomolecules at an atomistic resolution.

Methods

Our approach is specifically designed to enhance online streaming of files: compression increases the streamed frame rate and the use of the built-in video codec enables fast hardware-accelerated decompression. First we record all the coordinates of the atoms at different time from the MD trajectories, and find the minimum and range of coordinates on three axes. Then we tried to calculate their relative coordinates using the minimum and range, and round these relative coordinates into a number of 8 bits. Because we have three different axes, coordinate of a point at a certain time can also be interpreted as a 24-bit color depth pixel. Using Python Imaging Library (PIL) and video streaming software FFmpeg, we transfer all the coordinates at a certain time into an image, and consists a video using images from different time losslessly. Blocking method is used in allocating the location of a pixel in the image. We can easily compress and decompress this video using the additional information, which is the minimum and the range of coordinates on the three axes, that is saved in a small text file.

Results

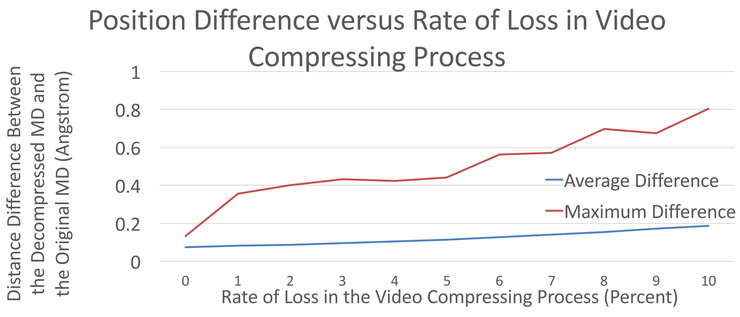
Results shown that our original algorithm, has a compression rate near to one eighth with an average position difference less than 0.1 Angstrom and the maximum difference less than 0.15 Angstrom for an ordinary protein MD trajectory. Rounding of the coordinates is the only resource of this difference. This distance is relatively small, compared to the size of the protein we studied in the MD trajectory.

Attempts of Improvement

Attempts to make improvement of our methods are in two directions:

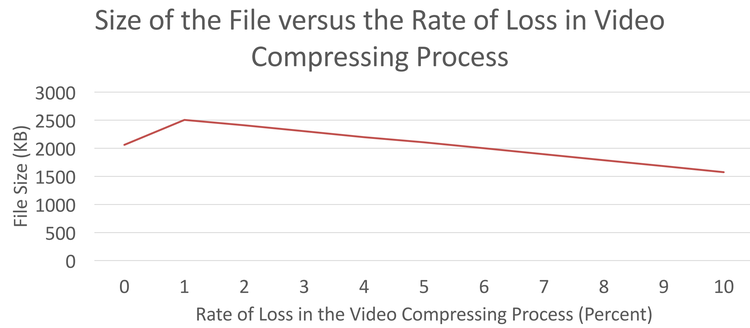
- Changing the rate of loss in the video compressing process to have a smaller file and an acceptable loss of data accuracy;
- Using blocking, sorting, or switching the origin to rearrange the pixel to have a smaller file.

To improve our methods, we changed the algorithm and make some tests to our sample MD file.



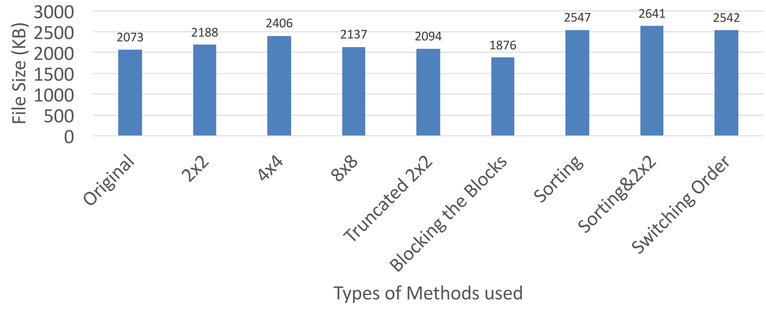
As shown in the first chart, increasing the rate of loss in video compressing process won't make a big increase on the average distance between the compressed data and the original data. But the maximum position distance will increase greatly if the video compressing method is not lossless.

Attempts of Improvement



As shown in the second chart, increasing the rate of loss in video compressing method will increase the file size at first. After the loss rate is higher than 5 percent, the file size become smaller than the original one.

Because the maximum difference and the average difference become too high after the loss rate is higher than 5 percent, we believe that it is useless to do so.



As shown in the third chart, different blocking methods will slightly increase the file size. If we truncated the last columns of block to make the area wasted smaller, the file size is still bigger than the original one by 20 KB. However, if we block the 8x8 blocks again in 4x4 bigger blocks, the file size will be 9.5% smaller than the original one. We suppose that this is because width and height of the video is a multiple of 16, so that the compression rate of the video codec can be higher.

Sorting doesn't make improvement to the file sizes. In fact, all the sorting methods will create an about 20% bigger file size, compared to the file size we got without sorting. Switching the origin also increase the file size by 20%.

Acknowledgements

This work was supported by the National Institute of Health [R01GM108340].

The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.