AUTODOCK - MULTIPLE GPU IMPLEMENTATION

Duc Chi NGO, Nguyen Dat HO, Quoc Lan PHAN, Thanh Tam NGUYEN and Huu Nhan TRINH

Versalogies
Institute for Computational Science and Technology (HCMC)

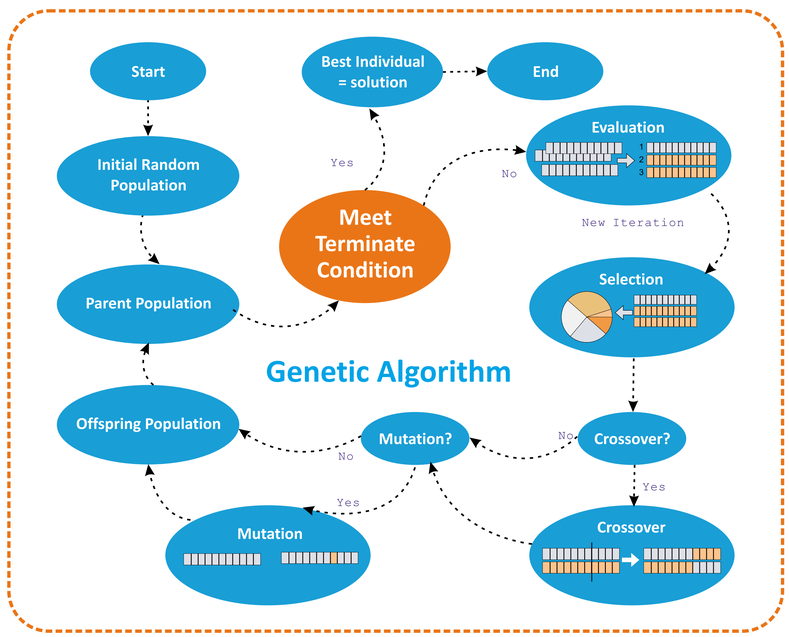
Motivation

AutoDock is one of the simulation software which creates 3D-models of molecular bindings which often used in pharmaceutical experiments. The software offers a suite of docking tools and is designed to predict how small molecules, such as substrates or drug candidates, bind to a receptor of a pre-defined 3D structure. With the particular strengths in simulation, it is widely used in thousands of research labs over the world and has been regarded as one of the most frequently cited molecular docking software in various scientific publications.

AutoDock is open source, freely distributed, and steadily further developed, until today. However, with its traditional serial programming model, AutoDock cannot utilize the parallel computing power of multi/many cores hardware models nowadays. This leads to the limitation in performance of AutoDock even when it runs on powerful workstation or super-computer. CUDA-fying AutoDock is one solution to let AutoDock run faster and utilize many cores hardware model of GPGPU, and there are existing CUDA versions [1][2] of AutoDock show that this is a potential and efficient solution. This report gives another view of utilizing CUDA for AutoDock: spending little effort to CUDA-fy 1 small part of AutoDock and run it on multiple GPUs to achieve 13x – 43.5x speedup comparing to running on 1 CPU.

AutoDock Main Algorithm - Genetic Algorithm

AutoDock uses Genetic Algorithm (GA) to find the optimal docking position of small molecule when it binds to a receptor. At the beginning, GA initializes a random population of individuals with pre-defined size. GA is an iterative process which produces new population in each iteration (often referred to as a 'generation'). In each generation, there are four main operations or phases: evaluation, selection, crossover and mutation. Firstly, the fitness of each and every individual in the population is evaluated (evaluation phase); then fitter individuals are stochastically selected from the current population (selection phase), and finally each individual's genome is modified (crossover and mutation phases) to form a new population for the next iteration. GA terminates when either a pre-defined number of generations has been completed, or a satisfactory fitness level of a population has been reached.



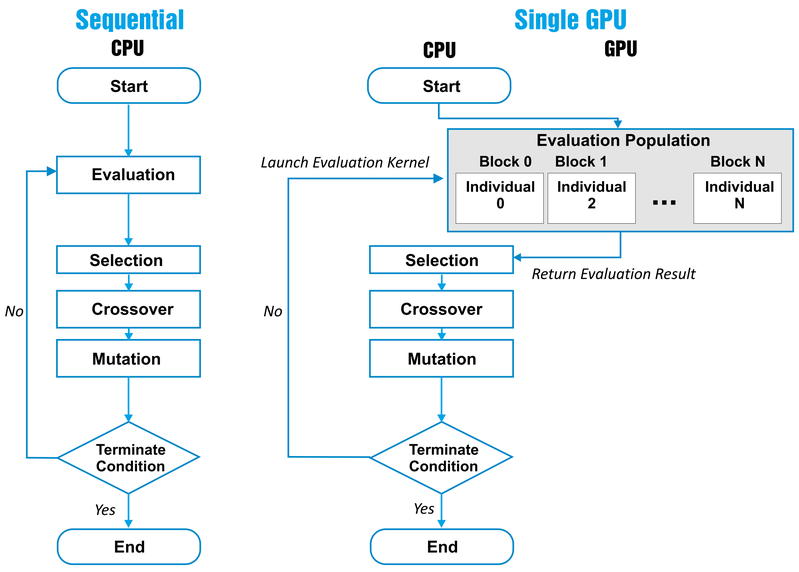
Approach

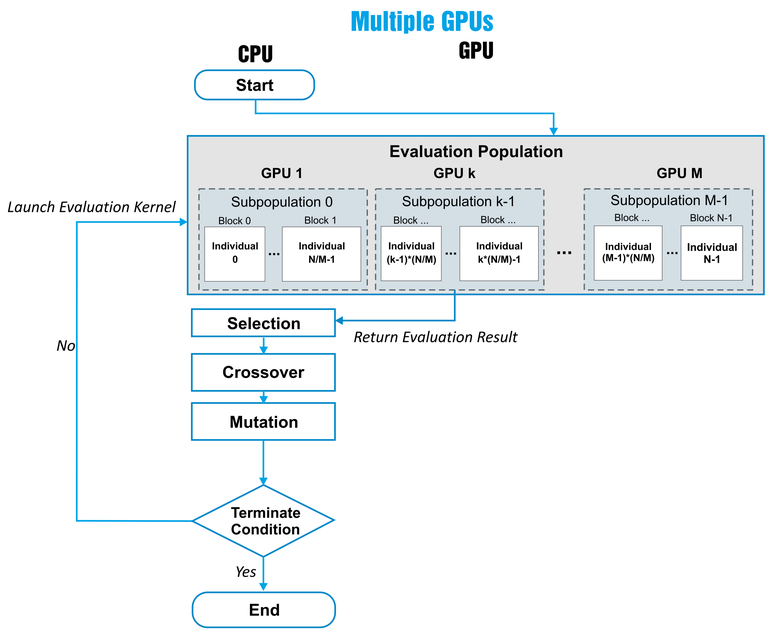
Through profiling, the time to execute evaluation phase consumes more than 70% total execution time of AutoDock, this is considered as a big bottleneck of AutoDock. With idea to run CUDA version of AutoDock on multiple GPUs, our approach to CUDA-fy AutoDock is below:

a. Develop 1 kernel for evaluation phase.
b. This kernel evaluate all individuals of population in 1 generation.
c. Each block of kernel evaluate 1 individual.

This approach has some advantages:

- CUDA-fy only small part of AutoDock code ⇒ time to develop CUDA code is very short.
- This CUDA code of AutoDock is data parallel and can be easily run on single or multiple GPUs.
- Be able to run with problem has very large population.





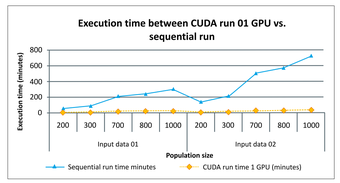
Result

First of all, correctness of result is verified carefully: result of CUDA version of AutoDock (runs on single or multiple GPUs) is the same with result of original version of AutoDock.

Performance of CUDA version of AutoDock is shown below:

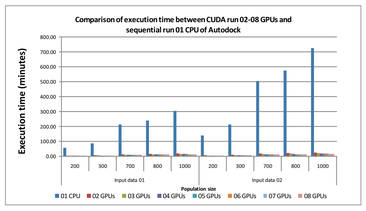
- Single GPU vs. CPU: 10.3x – 19.17x speedup
- Multiple GPUs vs. CPU: 13x – 43.5x speedup

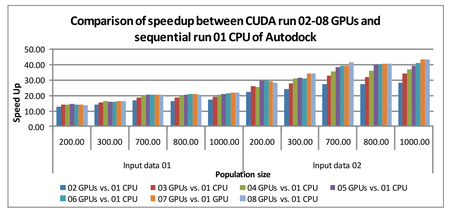
Input Data	Population size	Sequential run time (minutes)	CUDA run time 1 GPU (minutes)	CUDA run 1 GPU vs. sequential run speedup
Input data 01	200	56.81	5.5	10.33
	300	86.58	7.72	11.22
	700	212.95	17.34	12.28
	800	240.49	18.63	12.91
	1000	304.06	23.51	12.93
Input data 02	200	139.17	8.53	16.32
	300	212.62	12.32	17.26
	700	504.13	26.98	18.69
	800	574.74	30.12	19.08
	1000	726.63	37.91	19.17



Input Data	Population size	01 CPU	01 GPU	02 GPUs	03 GPUs	04 GPUs	05 GPUs	06 GPUs	07 GPUs	08 GPUs
Input data 01	200	56.81	5.50	4.37	4.04	4.04	3.90	3.96	4.04	4.20
	300	86.58	7.72	6.03	5.64	5.32	5.36	5.44	5.22	5.29
	700	212.95	17.34	12.57	11.26	10.70	10.32	10.32	10.38	10.32
	800	240.49	18.63	14.55	12.76	12.18	11.72	11.50	11.50	11.57
	1000	304.06	23.51	17.54	15.67	14.79	14.42	14.08	13.75	13.75
Input data 02	200	139.17	8.53	6.27	5.33	5.40	4.66	4.71	4.77	4.87
	300	212.62	12.32	8.80	7.65	6.88	6.73	6.81	6.22	6.19
	700	504.13	26.98	18.48	15.24	14.05	13.10	12.79	12.85	12.15
	800	574.74	30.12	20.92	17.82	15.94	14.27	14.27	14.14	14.07
	1000	726.63	37.91	25.79	21.30	19.64	18.49	17.55	16.70	16.70

Input Data	Population size	02 GPUs vs. 01 CPU	03 GPUs vs. 01 CPU	04 GPUs vs. 01 CPU	05 GPUs vs. 01 CPU	06 GPUs vs. 01 CPU	07 GPUs vs. 01 GPU	08 GPUs vs. 01 CPU
Input data 01	200.00	13.02	14.05	14.05	14.57	14.36	14.05	13.53
	300.00	14.36	15.37	16.27	16.16	15.93	16.61	16.38
	700.00	16.95	18.91	19.89	20.63	20.63	20.51	20.63
	800.00	16.52	18.85	19.75	20.53	20.91	20.91	20.79
	1000.00	17.33	19.40	20.56	21.08	21.59	22.11	22.11
Input data 02	200.00	22.20	26.11	25.79	29.87	29.54	29.21	28.56
	300.00	24.16	27.79	30.90	31.59	31.24	34.17	34.35
	700.00	27.29	33.08	35.88	38.50	39.44	39.25	41.49
	800.00	27.48	32.25	36.06	40.26	40.26	40.64	40.83
	1000.00	28.18	34.12	37.00	39.30	41.41	43.52	43.52





Hardware:

- *Sequential run: Intel CPU Core i7.*
- *CUDA run: Intel CPU Core i7 + NVidia Kepler K10.*

Data:

- *Input data 01: Small molecule has 14 torsions, 49 atoms. Receptor has 1686 atoms, in flexible residue mode.*
- *Input data 02: Small molecule has 33 torsions, 32 active torsions, 119 atoms. Receptor has 384 atoms.*

Funding:

- *This project is fully funded by the Institute for Computational Science & Technology (HCMC).*

Reference

[1] Sarnath Kannan, and Raghavendra Ganji, "Porting AutoDock to CUDA", WCCI 2010 IEEE World Congress on Computational Intelligence, Jul 18-23, 2010-CCIB, Barcelona, Spain.

[2] Timothy Blatter, Andrew St. Jean and Craig Struble, "AutoDock Software in Parallel with GPUs – Final Report", http://sourceforge.net/projects/gpuautodock/

Institute for Computational Science and Technology (HCMC)
SBI Building
Quang Trung Software City, Tan Chanh Hiep Ward, District 12, Ho Chi Minh City, Vietnam

Versalogies Company Limited
6A building, Quang Trung Software City, Dist.12, Ho Chi Minh City, Vietnam
Mr. Dat Ho
Email: ***dat@versalogies.com***
Cellphone: ***+84.907.979.055***