Making Drug Discovery More Sustainable by Reducing the Electricity Consumption and Cost of Virtual Screening

David Toth[1], Forrest Kamperman[1], and Evan Paige[2]
[1]Centre College, [2]University of Mary Washington
[1]600 West Walnut St., Danville, KY 40422, [2]1301 College Ave., Fredericksburg, VA 22401
david.toth@centre.edu, forrest.kamperman@centre.edu, epaige@mail.umw.edu

ABSTRACT

ARM processors consume less electricity than the traditional x64 CPUs used in laptop and desktop computers, clusters, and supercomputers. However, ARM processors are less computationally powerful than the traditional CPUs. As concerns over electricity consumption in data centers have risen, alternatives that consume less electricity than systems with traditional CPUs have become desirable. However, people do not want to sacrifice performance in the area of virtual screening, where high throughput is critical. A comparison of the performance of two molecular docking programs used for virtual screening, AutoDock Vina and DOCK 6, on several computers with ARM processors and two computers with traditional x64 CPUs reveals the potential of computers with ARM-based CPUs for virtual screening. For both docking programs, one of the computers with an ARM-based CPU consumed less than half of the electricity per compound screened than the computers with traditional x64 CPUs did. The same computer costs much less per compound screened in 24 hours than the computers with the x64 CPUs.

KEYWORDS

Sustainable computing, electricity consumption, virtual screening, drug discovery, ARM-based processors.

1 INTRODUCTION

Virtual screening has become an indispensable part of the drug discovery process, but it is extremely computationally intensive and requires large amounts of expensive computer hardware to perform large scale screens in a reasonable amount of time [1]. In order to speed up virtual screening, people have attempted to decrease the amount of time it takes on the supercomputers and large compute clusters that are typically used [2][3]. One additional negative aspect of virtual screening is that the computers used to do it consume vast quantities of electricity. These computers use traditional x64 CPUs, which were designed for general purpose computing.

In recent years, the proliferation of mobile devices such as smart phones and tablets has spurred development of more energy-efficient CPUs that use the ARM CPU architecture instead of the traditional x64 CPU architecture. These ARM-based CPUs, which are used in many mobile devices, not only consume less electricity, but they are much cheaper, too. In fact, one can now purchase an entire computer with an ARM-based CPU for around $50. While the ARM-based CPUs are cheaper and consume less electricity, they are not as fast as the traditional x64 CPUs, so in a fixed amount of time, a single ARM-based CPU can perform less computational work than an x64 CPU. Because of the significant cost difference between computers with the different types of CPUs, we speculated that given a fixed amount of money to spend on hardware, the group of computers with ARM-based CPUs that could be purchased might outperform computers with one or more traditional x64 CPUs. We also hypothesized that the group of computers with ARM-based CPUs would consume less electricity for virtual screening.

2 METHODOLOGY

To test our two hypotheses, we selected several computers with ARM-based CPUs to compare against our computers with x64 CPUs. We measured the number of compounds each computer could screen with each of two molecular docking programs over a 24-hour period. We also recorded the amount of electricity consumed by each molecular docking program during the 24-hour period. We used the molecular docking programs AutoDock Vina 1.1.2 and DOCK 6.6 to benchmark the computers [4, 5]. AutoDock Vina and DOCK are very heavily used docking programs that are free for academic use and each have hundreds or thousands of citations in Google Scholar [6, 7]. In contrast, licenses for commercial molecular docking software like GLIDE cost thousands of dollars and thus are likely used by far fewer researchers. The lead author works with several drug discovery research groups that use either DOCK or AutoDock Vina for virtual screening.

2.1 Hardware

We used two computers with x64 CPUs and three computers with ARM-based CPUs. One computer with a traditional x64 CPU was an off-the-shelf ASUS CM1831, which has an 8-core 3.1 GHz AMD FX-8120 CPU and 8 GB of RAM [8]. The other computer was a custom built system from Xi Computer Corporation designed to be used for virtual screening. The custom system contained four AMD Opteron 6378 CPUs, each with 16 CPU cores, running at 2.4-3.3 GHz, and 64 GB of RAM. The three systems with ARM-based CPUs were the Raspberry Pi Model B, the Cubieboard2 and the ODROID U3 [9, 10, 11]. The Raspberry Pi has a single core ARM1176JZF-S CPU with a clock speed of 700 MHz. The Cubieboard2 has a dual core ARM Cortex-A7 MPCore CPU with a clock speed of 1 GHz [12]. The ODROID U3 has a quad core Exynos 4412 Prime CPU with a clock speed of 1.7 GHz. All three of the systems with ARM-based CPUs used 8 GB class 10 microSD cards to store the operating system and all of the files for the tests. We used a P3 International Corporation P4400 Kill A Watt energy usage meter to measure the energy consumption of the computers [13]. The costs of the computers including the SD and microSD cards are shown in Table 1. The costs do not include shipping charges.

Table 1. Computer Costs

Computer	Cost
ASUS CM 1831 8-Core	$650.00
Xi Computer 64-Core	$5015.00
Raspberry Pi	$51.00
Cubieboard2	$78.50
ODROID U3	$76.50

2.2 Software

The computers with the x64 CPUs and the ODROID U3 ran the Ubuntu Linux version 13.10 operating system. The Raspberry Pi ran the Raspbian operating system, a variant of Debian Linux. The Cubieboard2 ran the Cubian operating system, a variant of Debian Linux. All five of the systems had the graphical desktop disabled, so they were only accessible through the command line interface as systems used for virtual screening would be. With the exception of the Raspberry Pi, all the computers used the xjobs version 2014-0125 program to ensure that one job was always being run per CPU core, to maximize the use of the CPU cores [14]. Because the Raspberry Pi has only a single CPU core, it did not need to run xjobs to maximize the utilization of its CPU. The computers ran AutoDock Vina 1.1.2 and DOCK 6.6.

2.3 Installing the Necessary Software

We used the same procedure to test the systems'

performance with AutoDock Vina and with DOCK. First, we set up those two virtual screening programs on all of the systems. For AutoDock Vina, a pre-compiled binary is available for the x64 systems running Linux. However, for the systems with ARM-based CPUs, we needed to first install Boost 1.41.0 and compile it [15]. Once we had compiled Boost, we were able to compile AutoDock Vina from the source code. For all of the systems, we had to compile DOCK from source. Then we installed and compiled xjobs on all of the systems except the Raspberry Pi.

2.4 Preparing For the Virtual Screens

Once we had installed the two virtual screening programs, we retrieved a protein to screen (dock). For AutoDock Vina, we used the protein Cbp1 as our target. We wanted to use a single molecule to screen the protein against as a control for the experiments so the different times that different molecules take to screen wouldn't impact the performance measurements. For the AutoDock Vina tests, we wanted to select a molecule that would be screened relatively quickly, so we could easily and accurately compare the performance of the systems when we ran the full tests. We randomly selected the molecule ZINC00000048.pdbqt in the SetOf10k_0000 folder from the full_nci_ALL_TAUTOMERS_2011 library of ZINC database [16]. We screened this molecule using the following specifications: --center_x -2.952 --center_y -0.432 --center_z -3.154 --size_x 46 --size_y 32 --size_z 30 --exhaustiveness 8 --cpu 1 --seed 21. Since this docking test ran quickly (about 30 seconds on the fastest system and about 10 minutes on the slowest system), we decided that it would be adequate to allow us to see the performance differences between the computers clearly. In order to use this molecule as a control, we made enough copies of it on each system so that none of the systems would be able to screen all the copies in 24 hours. By having a separate copy of the molecule for each docking task, we simulated the way the file system on the computers would come into play during a normal virtual screen, with a different file having to be read from disk for each molecule to be tested. For DOCK, we used the protein 2QNX [17].

2.5 Running the Virtual Screens

Once we completed the preparations, we ran 10 trials on each computer with AutoDock Vina and 10 trials on each computer with DOCK. Each trial ran for 24 hours, using xjobs and a shell script to make sure each CPU core was constantly busy screening a molecule. Xjobs launched a number of screens equal to the number of CPU cores on the system and starting to screen another molecule as soon as one finished. We measured the electricity consumption of each computer during that time as well as the number of molecules that the program completed screening. After 24 hours, we stopped the script, recorded the electricity consumption and number of molecules that the computer had screened, and reset the computers to the pre-trial state by deleting the output files they had produced. Then we powered off the computers, reset the electricity monitors, powered on the computers, and started the next trials. Because AutoDock Vina writes a great deal of information to the console window that is not necessary and this impacts the performance of systems with a good number of CPU cores, we forced it to write it to /dev/null instead of the screen so it wouldn't affect the performance of the systems and we recommend that others else adopt this method to increase the productivity of AutoDock Vina. We also had DOCK write its output that normally goes to the console window to /dev/null instead of the screen. The practice of writing this output to /dev/null has no negative impact on the usefulness of the virtual screen because all of the output is also written to a file by both programs. Therefore, no information is lost with this practice, but the time to perform the

virtual screen is reduced. In fact, we use this practice of redirecting the output to /dev/null when we run the virtual screens normally.

3 RESULTS

We ran 10 trials of each molecular docking program on each computer being tested. We found that there were only minor variations in results throughout the trials and concluded that 10 trials would be enough for accurate results. The results of the trials are shown in Tables 2-5. Tables 2 and 3 show the number of molecules screened by AutoDock Vina and DOCK respectively for each computer in 24 hours. Tables 3 and 4 show the amount of electricity AutoDock Vina and DOCK consumed respectively in a 24-hour period. As expected, the computers with the x64 CPUs, which have much higher clock speeds, were able to screen more compounds in 24 hours and consumed significantly more electricity in the process. On each computer, the number of runs DOCK completed differs significantly from the number runs that AutoDock Vina completed. This was expected, as the two docking programs are very different. The electricity consumed by a given computer differed much less between the two docking programs, likely because the computers' CPUs were kept running at nearly full utilization the entire time. We believe that the difference in electricity consumption between the two docking programs was likely a result of DOCK writing to the SD or microSD card more frequently than AutoDock Vina, since it screened more molecules in the 24 hours, which would allow the CPU of the computer to use less electricity over time. We suspect that running the CPU consumes more electricity per unit of time than reading from and writing to the SD or microSD card.

Table 2. Completed Runs of AutoDock Vina in 24 Hours

Trial	ASUS 8-Core	Xi Computer 64-Core	ODROID U3	Cubieboard2	Raspberry Pi
1	11,493	82,560	3,409	768	146
2	11,475	82,549	3,405	769	146
3	11,491	82,541	3,397	767	146
4	11,493	82,564	3,400	767	148
5	11,492	82,552	3,404	768	147
6	11,490	82,552	3,410	770	147
7	11,491	82,560	3,408	770	147
8	11,494	82,559	3,403	768	148
9	11,490	82,551	3,401	769	147
10	11,492	82,555	3,400	768	145
Average	11,490.1	82,554.3	3,403.7	768.4	146.7

Table 3. Completed Runs of DOCK in 24 Hours

Trial	ASUS 8-Core	Xi Computer 64-Core	ODROID U3	Cubieboard2	Raspberry Pi
1	144,390	960,116	23,257	5,583	999
2	144,446	958,307	22,393	5,548	1,005
3	144,313	957,443	23,101	5,570	997
4	144,334	957,672	22,950	5,558	992
5	144,389	956,928	23,333	5,549	993
6	144,338	957,438	23,761	5,568	1,005
7	144,300	957,772	23,135	5,558	1,002
8	144,298	958,251	23,316	5,530	1,000
9	144,325	957,791	23,235	5,577	1,007
10	144,290	957,814	23,499	5,544	993
Average	144,342.3	957,953.2	23,198.0	5,558.5	999.3

Table 4. Electricity Consumed (kWh) by AutoDock Vina in 24 Hours

Trial	ASUS 8-Core	Xi Computer 64-Core	ODROID U3	Cubieboard2	Raspberry Pi
1	3.88	14.14	0.16	0.06	0.06
2	3.95	14.02	0.16	0.06	0.06
3	3.96	14.07	0.16	0.06	0.06
4	3.94	13.96	0.16	0.06	0.06
5	3.93	13.85	0.17	0.06	0.06
6	3.88	13.85	0.16	0.06	0.06
7	3.87	13.85	0.16	0.06	0.06
8	3.87	13.85	0.17	0.06	0.06
9	3.87	13.86	0.16	0.06	0.06
10	3.87	13.80	0.17	0.07	0.06
Average	3.902	13.925	0.163	0.061	0.06

Table 5. Electricity Consumed (kWh) by DOCK in 24 Hours

Trial	ASUS 8-Core	Xi Computer 64-Core	ODROID U3	Cubieboard2	Raspberry Pi
1	4.04	14.91	0.13	0.06	0.06
2	4.01	14.87	0.13	0.06	0.06
3	4.04	14.89	0.13	0.06	0.06
4	4.03	14.79	0.13	0.06	0.06
5	3.97	14.91	0.13	0.06	0.06
6	3.96	15.00	0.13	0.06	0.06
7	4.01	14.62	0.13	0.06	0.06
8	4.00	14.62	0.13	0.06	0.06
9	3.96	14.63	0.13	0.06	0.06
10	3.95	14.56	0.13	0.06	0.06
Average	3.997	14.78	0.13	0.06	0.06

To determine which computer would consume the least electricity to perform a virtual screen, for each docking program, we divided the electricity consumed by a computer in 24 hours by the number of compounds the computer screened in that time with the given docking program. Because the resulting values were so small, to enable us to more easily compare them, we multiplied each value by 10,000. This had the effect of giving us the amount of electricity each computer would consume with each docking program to screen 10,000 compounds. These results are shown in Figures 1 and 2. For both AutoDock Vina and DOCK, the ODROID U3 with its ARM-based CPU consumed the least amount of electricity, consuming less than half of the electricity than the computers with the x64 CPUs. We suspect that the ODROID U3 consumes significantly less electricity because the ARM-based CPUs are designed to consume less electricity than x64 CPUs. They are able to do this in part due to the lower clock speed and in part because they do not need to support all the instructions that the x64-based CPUs must, which lowers the number of transistors needed for them and thus, their electricity consumption.

While it was clear that the ODROID U3 consumed the least electricity to conduct a virtual screen, we recognize that people are often unwilling to accept lower performance and having it take longer to get their results in order to use a more sustainable solution. Therefore, we felt it was important to also compare the amount of time it would take to complete a virtual screen with the different types of computers. While it is clear from Tables 2 and 3 that a single ODROID U3 will take much longer to screen as many compounds as a computer with an x64 CPU, we note that the computers with ARM-based CPUs are significantly cheaper than the computers with x64 CPUs. Since a scientist purchasing hardware for conducting virtual screens will have a given amount of money to spend on hardware, we felt that a fair comparison would be achieved by assuming that the scientist would spend the same amount of money on either computers with x64 CPUs or computers with ARM-based CPUs. Therefore, the fairest comparison in terms of performance is to determine which computers can screen the most compounds in a fixed amount of time if one spent the same amount of money on each of the possible types of hardware. That means that the person might be able to purchase multiple computers with ARM-based CPUs instead of a

single computer with one or more x64 CPUs. This meant we needed to determine which computer provided the greatest computational performance for the cost. To determine this, for each docking program, we divided the cost of the computer by the number of compounds a computer could screen in 24 hours. As with the electricity consumption, because the resulting values were so small, we then multiplied all the results by 10,000 to give the hardware cost to screen 10,000 compounds in 24 hours. The results of this calculation are shown in Figures 3 and 4. For both docking programs, the ODROID U3 delivered the lowest hardware cost to screen 10,000 compounds in 24 hours by a large margin. For AutoDock Vina to screen 10,000 compounds in 24 hours, one would have to spend $565.70 on a computer with an x64 CPU, whereas one could spend only $224.76 on ODROID U3 computers (about 40% of the cost) to achieve the same result. For DOCK, one would have to spend $45.03 on a computer with an x64 CPU, whereas one could spend only $32.98 on ODROID U3 computers (about 75% of the cost) to achieve the same result. Therefore, the ODROID U3 computers are not only the most sustainable computer we tested, but also the fastest solution if one spends the same amount of money on them as one would on another type of computer. That means one does not have to sacrifice performance to use the more sustainable solution.

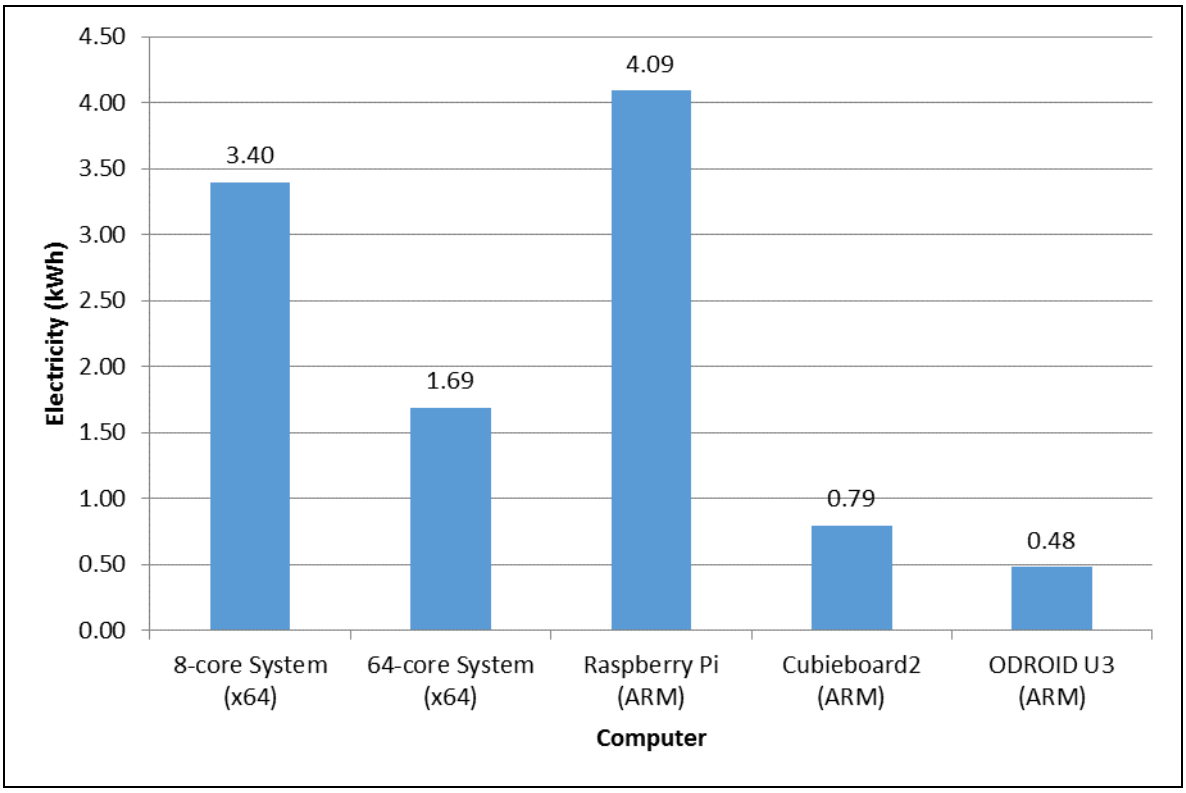

Figure 1. Electricity Consumed to Screen 10,000 Compounds with AutoDock Vina

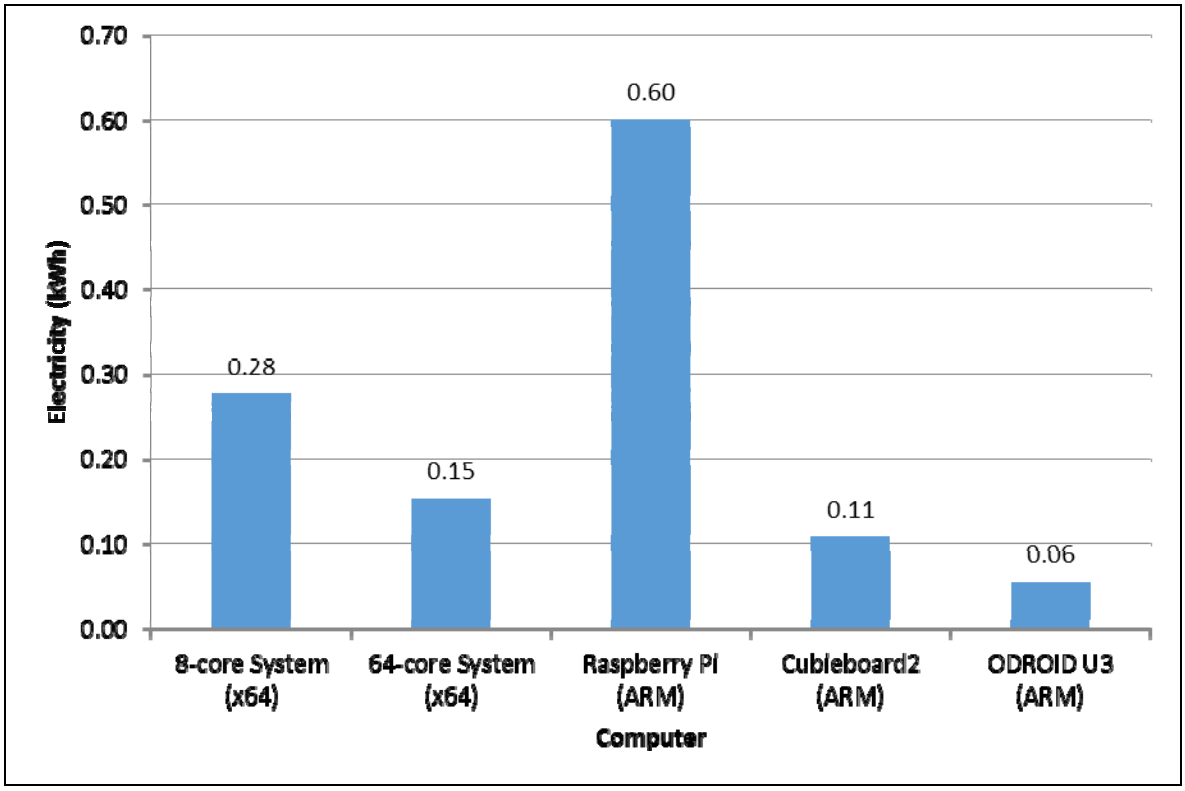

Figure 2. Electricity Consumed to Screen 10,000 Compounds with DOCK

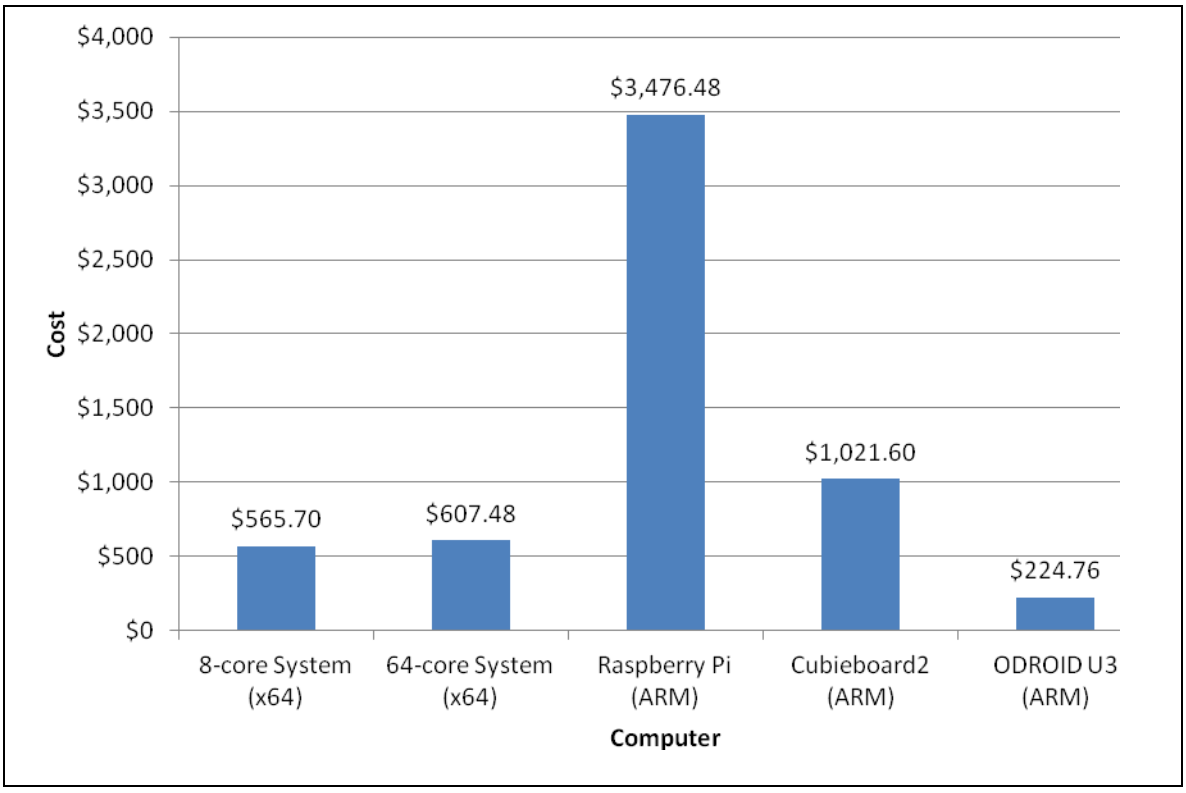

Figure 3. Hardware Costs to Screen 10,000 Compounds in 24 Hours for AutoDock Vina

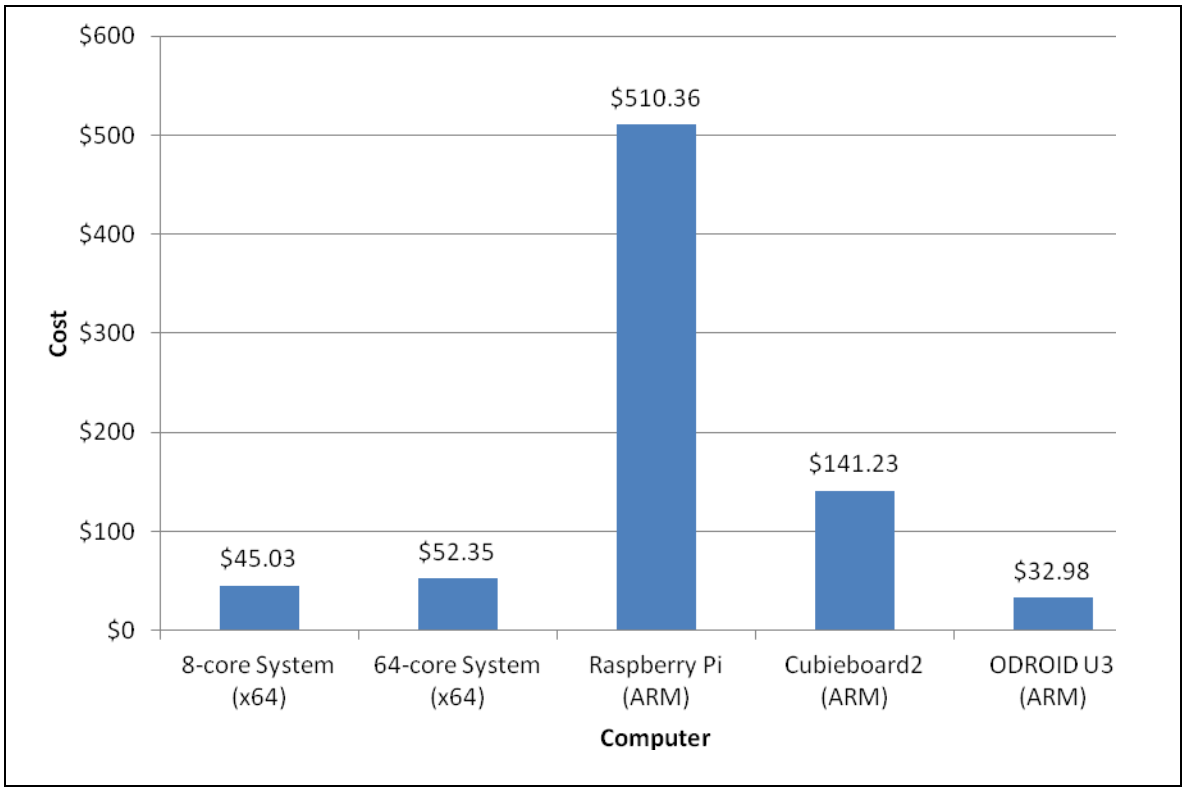

Figure 4. Hardware Costs to Screen 10,000 Compounds in 24 Hours for DOCK

4 CONCLUSIONS

Our results have shown that a set of computers with ARM-based CPUs can provide a more sustainable way to conduct the computational portion of the drug discovery process (virtual screening) than computers with traditional x64 CPUs without increasing the time required to get results and without costing more money. Not only can a group of ODROID U3 computers with ARM-based CPUs screen a given number of compounds in the same amount of time as a number of x64-based computers, but the group of ODROID U3 computers required to accomplish this task uses less than half of the electricity and costs less than the group of computers with x64 CPUs required to accomplish this. Therefore, using the ARM-based ODROID U3 computers can both lower the energy consumption and the cost of virtual screening.

5 FUTURE WORK

We are looking to do further research in this field by testing out other ARM based systems. We hope to test out AMD's soon-to-be-released ARM based system sometime in the future. We also plan to test the recently released ODROID XU3-Lite computer, which may provide even better results than the ODROID U3. We also expect to replicate these tests on a much larger scale, using a cluster of many ODROID U3s to determine if the energy savings and improved performance hold for clusters of these systems.

REFERENCES

[1] A. S. Reddy, S. P. Pati, P. P. Kumar, H. N. Pradeep, G. N. Sastry, "Virtual screening in drug discovery -- a computational perspective.," J. Curr Protein Pept Sci. vol. 8, pp. 329-351, August 2007.

[2] X. Zhang, S. E. Wong, F. C. Lightstone, "Message Passing Interface and Multithreading Hybrid for Parallel Molecular Docking of Large Databases on Petascale High Performance Computing Machines," J. Comp. Chem., vol. 34, pp. 915-927, January 2013.

[3] S. R. Ellingson, J. C. Smith, J. Baudry, “VinaMPI: Facilitating multiple receptor high-throughput virtual docking on high-performance computers,” J. Comp. Chem., vol. 34, pp. 2212-2221, June 2013.

[4] O. Trott, A. J. Olson, “AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading,” J. Comp. Chem., vol 31, pp. 455–461, June 2009.

[5] P. T. Lang, S. R. Brozell, S. Mukherjee, E. F. Petterson, E. C. Meng, V. Thomas, R. C. Rizzo, D. A. Case, T. L. James, and I. D. Kuntz, “DOCK 6: combining techniques to model RNA-small molecule complexes,” RNA, vol. 15, pp. 1219-1230, June 2009.

[6] J. Biesiada, A. Porollo, P. Velayutham, M. Kouril, and J. Meller. Survey of public domain software for docking simulations and virtual screening. Hum Genomics. 2011; 5(5): 497–505. Published online 2011 Jul 1. doi: 10.1186/1479-7364-5-5-497.

[7] Google Scholar. http://scholar.google.com/ (accessed May 10, 2015).

[8] Tiger Direct, ASUS CM1831-US-3AA Desktop PC - AMD FX-8120 3.1GHz, 8GB DDR3, 2TB HDD, DVDRW, Windows 7 Home Premium 64-bit at TigerDirect.com. http://www.tigerdirect.com/applications/SearchTools/item-details.asp?EdpNo=1858586 (accessed May 10, 2015).

[9] Raspberry Pi Foundation, Raspberry Pi, http://www.raspberrypi.org/ (accessed May 13, 2014).

[10] Cubieboard, Cubieboard | A series of open sorce hardware, http://cubieboard.org/ (accessed September 13, 2014).

[11] Hardkernel co., Ltd., ODROID | Hardkernel, http://www.hardkernel.com/main/main.php (accessed May 10, 2014).

[12] linux-sunxi.org, Cubietech Cubieboard2, http://linux-sunxi.org/Cubieboard2 (accessed October 4, 2014).

[13] P3 International Corporation, P3 - Kill A Watt, http://www.p3international.com/products/p4400.html (accessed September 28, 2014).

[14] T. Maier-Komor, ~xjobs/ - the home of the xjobs utility, http://www.maier-komor.de/xjobs.html (accessed June 15, 2014).

[15] Boost.org, Boost C++ Libraries, http://www.boost.org/ (accessed June 8, 2014).

[16] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad and R. G. Coleman, “ZINC: A free tool to discover chemistry for biology.,” J. Chem. Inf. Model. vol. 52, pp. 1757-1768, July 2012.

[17] RCSB Protein Data Bank, RCSB Protein Data Bank - RCSB PDB - 2QNX Structure Summary, http://www.rcsb.org/pdb/explore.do?structureId=2qnx (accessed October 4, 2014).