e-Science Journal
Global Scientific Information and Computing Center
Vol. 2

TSUBAME ESJ.

TSUBAME 2.0 Begins

The long road from TSUBAME1.0 to 2.0 (Part One)

Multi-GPU Computing for Next-generation Weather Forecasting

- 145.0 TFlops with 3990 GPUs on TSUBAME 2.0 -

Computer prediction of protein-protein interaction network using MEGADOCK

- application to systems biology -

TSUBAME 2.0 Begins

The long road from TSUBAME1.0 to 2.0 (Part One)

Satoshi Matsuoka*
* Global Scientific Information and Computing Center

The long-awaited TSUBAME2.0 will finally commence production operation in November, 2010.
However, the technological evolutionary pathway that stems from times even earlier than TSUBAME1, its direct predecessor, was by no means paved smoothly.
In the Part one of this article, we discuss the pros and cons of TSUBAME1,
and how they have been addressed to achieve a 30-fold speedup in mere 4.5 years in TSUBAME2.0.

Introduction 1

Early October 2010—Opening the door to a room on the ground floor of our center, where our admin staff was kept busy handling paperwork, a brand new scenery from another world would immediately jump into sight (Figure 1). To my ear that had been accustomed to the roaring and violent noise of TSUBAME1.0 solidly claiming its mighty presence, this time I could only hear the gentle sound of sound of air circulation in a new sealed water-cooled rack. TSUBAME2.0's soaring presence with deep rows of racks would not initially present itself as the fastest supercomputer in Japan containing the world's most advanced supercomputing technologies, but would rather resemble rows of office filing cabinets. But once one opens the door of the rack, pristine rows of TSUBAME2.0 compute nodes that had been newly developed with extreme computational density becomes visible, augmented with mere few cables protruding from each of its front panels. Overall performance contained in only a single rack that looks more like a large refrigerator is 50 Teraflops, a performance that would have had it ranked number one in the world mere 8 years ago, comparable to the actual machine the Earth Simulator that had occupied the entire facility of a large gymnasium, consisting of more than 600 racks that resembled groups of skyscrapers when viewed from its observatory. Overall TSUBAME2.0 performance of 2.4 petaflops is faster than the total of all the supercomputer performances of all such owned by public institutions in Japan circa 2010 — although we had intentionally designed it so dense due to various technical reasons, it is still is very amazing and exciting when one actually sees the result of such a technological feat.

Before even becoming operational in November, there seems to be almost never ending stream of visitors wishing for a tour of TSUBAME2.0. But just by observing the machine from

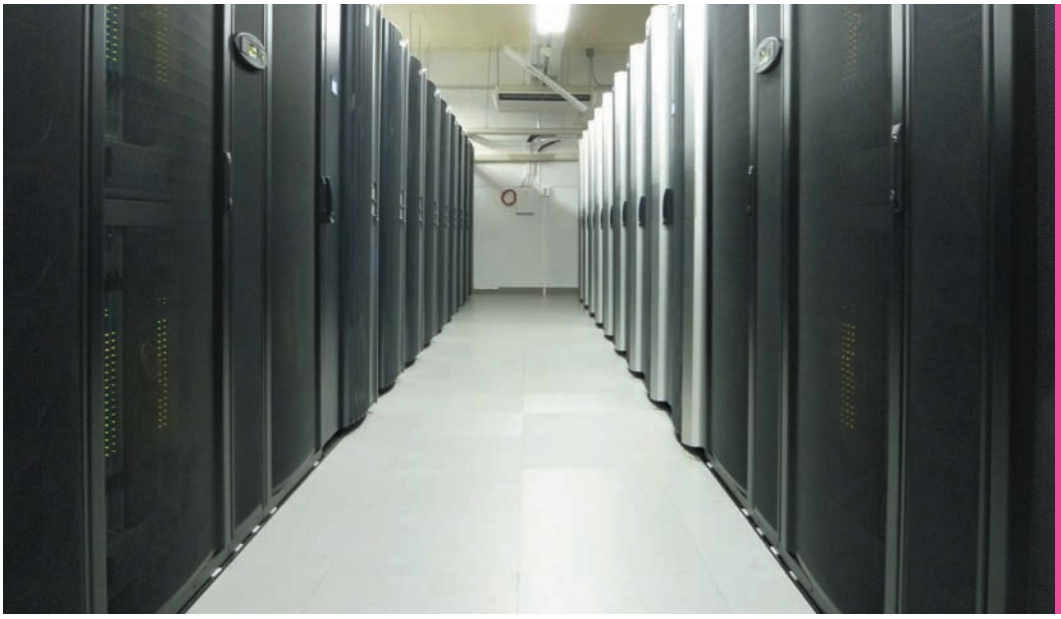


Figure 1
Newly Created Computer Room GSIC105 for TSUBAME2.0

its exterior does not reveal its contents; moreover just learning the machine as is does not reveal the true nature of the design decisions that had been made. The important question is, "why was it designed so as a supercomputer", i.e., or what were the rationale of a particular component selection, how and why they were combined, and how the experiences and knowledge of TSUBAME1 had been technically reflected in the design. For users as well as for the future development of the supercomputers in general, these questions must be answered clearly, and furthermore, whether we have actually had conducted proper design, engineering, and construction of TSUBAME2.0 given such technical challenges, need to be assessed. Simple stable operation as an infrastructure would not be sufficient for a machine to which the taxpayers will be investing more than 3 billion yen for the next 4 years as a machine in a national supercomputing center for academia and the industry nationwide—its advance technologies as well as their utility must be properly evaluated.

Given such a perspective, the predecessor TSUBAME1.0 had a unique standpoint as one of Japan's leading supercomputer. In particular, it was a machine that our center did not merely buy out of a vendor product catalog; rather, similar to the Earth Simulator or the follow-ons of TSUBAME, such as the so-called T2K supercomputers, it was a procurement of a machine that had been effectively jointly designed and developed with supercomputing manufacturers. Moreover, the cycle consisting of, (1) basic research in supercomputing and system software design, (2) experimental production of nascent hardware and software at GSIC and their assessment, (3) development and determining the specifications of a production machine, as well as feedback to a new cycle re-starting with (1), had proven to be critical over the years. Such "waterfall" development model was exactly our strategy for TSUBAME1.0: basic research of system software in clusters as well as various highly-parallel applications at various laboratories throughout Tokyo Institute of Technology constituted (1), while more than four years of operating a total of 400 nodes in the "Titech Campus Grid Project" in experimental production as group of remotely-managed clusters provided the experiences for (2), and based on those experiences we were able to determine the TSUBAME1 specifications (3). For TSUBAME2.0, basic research (1) was often conducted not only on TSUBAME1 but on other research platforms, but as will be described later, TSUBAME1 was retrofitted with additional equipments, and integrated experimental operation was achieved alongside normal production as (2). The experiences drawn resulted in the specifications of the successor model TSUBAME2.0 (3). Next, it is TSUBAME2.0's turn to serve experimental roles for its future successor, TSUBAME3.0!

In the last volume of the Tsubame E-Science Journal, our article described the characteristics of each component of TSUBAME2.0 [1]. In this article, we attempt to provide the behind-the-scenes technical rationalization of the design. Part 1 of this article will first cover the design of TSUBAME1, and briefly describe that experience that affected the TSUBAME2.0 design. In part 2, latest technology trends from petascale towards exascale will be discussed, and how TSUBAME2.0 serves as a prototype, how such requirements are reflected in the design. More complete introduction and evaluation of TSUBAME2.0 technologies would require a separate book, perhaps published after its operation begins. For the moment we hope that the this article will serve to deepen users' understanding of TSUBAME2.0 as a reference to its effective usage.

Looking Back at TSUBAME 1.0 2

The predecessor of TSUBAME2.0, i.e., TSUBAME1.0, (Figure 2) was inaugurated in April 2006. It was a combined fruition of a series of various research on cluster computing at Tokyo Tech over the years, as well experiences in fielding production supercomputers for over 20 years, in addition to experimental production operation projects such as Titech campus grid as mentioned earlier (April 2002 till March 2006). Based on the experiences, various design studies were conducted, resulting in a 655 node supercomputer that embodied over 10,000 CPU cores, exhibiting 50 Teraflops in total with 21 terabytes of memory, in addition to 1.1 Petabytes of hard disk storage, interconnected by Infiniband. The compute nodes were augmented with 360 ClearSpeed Accelerator cards (later expanded to 648 cards) well-suited for dense matrix operations, providing additional 30 Teraflops of compute power. TSUBAME in a way became a template architecture for large-scale cluster-based supercomputers for its follow-ons, as it possessed

Figure 2 The old Computer Room that housed TSUBAME1

architectural parameters had not been seen in clusters those days, and matched the best supercomputers in the world. In particular, circa 2006, a typical cluster might have sported 2 to 4 CPU cores per node with 1-4 Gigabytes of memory, whereas Tsubame1 utilized the "glue-less" multi-socket capability of the latest AMD Opteron 800 series to implement a so-called "fat node" with 16CPU cores and 32-128Gigabytes of memory each, with (80+80) gigaflops peak performance from the CPUs and the ClearSpeed respectively, interconnected by a dual-rail Infiniband fabric for 20Gbps bandwidth. Such a configuration allowed for both stability (smaller number of nodes) as well as provided extremely high ease-of-use as a supercomputer, as well as offering an environment where one's applications might not execute otherwise. The evidence of this is that TSUBAME1 is still highly competitive and in very high demand, even as it nears its retirement well beyond its initial designed shutdown in the Spring of 2010.

Based on these characteristics, TSUBAME1.0 [2] was the first supercomputer in Japan to supersede the Earth Simulator on the Top500, a former world champion, and sustained Asia's No.1 supercomputer position for a year and a half. Moreover, the number of user base both inside and outside the Tokyo Institute of Technology, quadrupled to 2,000, including the industry users. TSUBAME1.0 retained its popularity as "Everybody's Supercomputer" being utilized by a highly diverse user base.

TSUBAME1 can thus can be objectively regarded as a "success". Not only on for the "grand-challenge" performance benchmarks such as the Top500, compared to conventional supercomputers that had been specially designed, similar high compute and storage capacity were achieved maintaining the properties of a large-scale supercomputer, but in a much more user-friendly environment. Not only these resulted in high utilization as well as a significant increase in the number of users, it had brought innumerable benefits to GSIC and to the university, including being designated as one of the Ministry of Education's Global Center of Excellence Program "CompView"; officially becoming one of the so called coalition of Major National Universities' Supercomputer Centers along with other 7 centers; officially being approved as one of the leading program members to promote industrial usage of advanced research facilities, with nearly 50 well-known companies being selected to use TSUBAME1.0 for future industrial applications; and finally being selected to be major research collaborators by major US IT companies such as Microsoft and NVIDIA, as "Center of Technical Innovation" and "CUDA Center of Excellence" respectively, demonstrating our new status as one of the leading supercomputing centers being recognized not only within Japan but also internationally.

TSUBAME1.0's behavior was comprehensively recorded in detail over its 4+ years of operational experience, not only for large-scale applications and benchmarks that imposed heavy load on the system, but also "normal" activities involving hundreds of concurrent users launching over 1 million jobs per year, leading to the TSUBAME2.0 design. Below, we discuss the pros and cons of TSUBAME1, that is "what went well" and "what had problems". What went well in design obviously carried over to TSUBAME2.0, while the problems were the subject of R&D to be solved for TSUBAME2.0.

TSUBAME1 – "The Bright Side" 3

By all means we lack the paper real-estate to cover every detailed aspect of TSUBAME1's strong points, but here we list the technological points that lead to its success, and carried over to TSUBAME2.0:

(A) **Significant adoption of industry standard technologies such as high-performance x86 processor and Linux-OS:** Advantages of PC clusters due to the adoption of x86 as well as its PC and server industry "ecosystem" have been reported a number of times, such as just-in-time adoption of high-performance and cost-effective technologies, as well as stability of quality achieved in the quantity due to mass production. In addition, continuity of the software and programming models from the usual laboratory PCs, workstations, and small clusters all the way up to supercomputing scale is a major benefit, as the increasing complexity of simulation software has transcended simulation work from being done on a single supercomputer, but rather, the mainstream usage model is for the same software to run seamlessly in a variety of environments from personally small to extremely large ones. Finally, it would be desirable to offer a easy path for a typical novice PC simulation user to their eventual use of supercomputers by allowing such step-by-step advances in a most transparent fashion. To achieve these goals of "Everybody's Supercomputer" was the main objective of TSUBAME1, and the most critical of the qualities to achieve such goals were actually achieved through the aggressive adoption of both hardware and software standards as much as possible.

(B) **Implementation of a "fat-node" architecture with mainstream x86 CPUs:** almost being an antithesis to the previous point, supercomputers are meaningless without being associated with the unique value of its use. In particular, a supercomputer would be a big attraction in solving not just when it would be able to achieve acceleration in time-to-solution, but also being able to solve problems that had been previously capacity-constrained. Thus, not only are total processor count and aggregated memory capacity for the entire system important, but also core counts and shared memory capacity inside a node would also be of significance if problems were constrained at that level as they often would be. In the past such "fat-node" design called for customized and expensive supercomputer design; for TSUBAME1, we were able to exploit the first generation of the latest technology x86 at the time that sported high-bandwidth processor-to-processor interconnect as well as to memory in a glueless fashion, namely the AMD Opteron 800 series processors, which allowed up to 8 sockets or 16 CPU cores, largely matching the numbers for dedicated supercomputer design of the time.

Fat node design not only benefits the users, but also helps to reduce the number of "moving parts" in the system allowing for higher reliability. Moreover it eases the redundancy and safety in the design, such as redundant power supply and fans, numerous thermal as well as other sensors as well as comprehensive IPMI-based monitoring and control network intended for large-scale server farms, etc. Augmented with adoption of efficient cooling for the time with clever configuration of servers versus the CRC units, as well as well-attuned operations, it was possible to build and operate a 80 teraflop supercomputer, that became the 7th fastest supercomputer in the world at the time. The TSUBAME technology gave birth to many follow-ons with their unique, further improvements, such as the "Ranger" cluster at the University of Texas TACC Supercomputer Center , the "Europa" cluster at the Julich supercomputer center in Germany, and the T2K supercomputers that were facilitated at centers in Universities of Tokyo, Kyoto, and Tsukuba respectively in 2008.

(C) **Accelerators for dense matrix computation:** Based on the technology at the time of TSUBAME1, it was already difficult to achieve 100 Teraflops given the constraints of space, power, costs, etc. As a result, we had decided to experiment acceleration technologies that had demonstrated reasonable results within the research lab of our center. Considering several candidates for actual production usage, ClearSpeed SIMD accelerator that was just being productized by a UK company of the same name was chosen, due to its favorable performance in dense matrix-type operations at very low power levels. For large dense matrix operations on using the BLAS library, user performance nearly doubled with a mere command line switch, without modifying the source code, or adding significant power or space to the infrastructure. However, although it also contributed significant performance gains for our Linpack it was necessary to conduct research and development of a new heterogeneous algorithm to maximize their contributions [3].

(D) **Multi-rail InfiniBand-based Fat-tree network using large switches, and integration of I/O storage network.:** Another important element of the supercomputer is the high-speed network interconnections between the nodes. In particular, in addition to bandwidth available to a single node (typically called the injection bandwidth), latency between the nodes must be very low, in the order of several microseconds, while the bisection bandwidth, or the bandwidth available to the entire sets of nodes when they conduct all-to-all communication, needs to be extremely high. By all means, low space overhead, low number of cables, low cost, and high reliability are required simultaneously, and many of these elements are sometimes contradictory in nature. TSUBAME1 employed a set of eight large 288-port Infiniband switches configured in a two-tier fat-tree, six switches in the lower edge layer and two on the top layer. Also, each node had two rails of network ports. This allowed each node to achieve 20Gbps bandwidth with approximately 5ns end-to-end network latency, matching the performance of supercomputers with dedicated networks. Moreover, the fat-tree configuration allowed the network to be symmetrical, making the operation to be very flexible. For example, if one of the nodes would fail, it would have been possible to use an alternate node without affecting the performances of other nodes.

(E) **High-performance storage achieving high-performance, high density, and low cost, with a parallel file system:** the oft-forgotten part of supercomputers is storage; for TSUBAME1-level simulation and processing capabilities in the 100 Teraflops, it was judged from early on that sub-petabyte class data handling capabilities will be of absolute necessity. As was with compute nodes, storage subsystem required high-performance, low cost and low power consumption, high reliability and scalability that coincided the parallelism in the system. It became clear that traditional enterprise IT systems

technology-based storage systems were inadequate. Rather, a cluster-based approach was also taken, where a powerful "fat" storage server that embedded 48 HDDs (24 Terabytes raw capacity) along with a powerful storage and network controller in a mere 4-U chassis, namely the Sun x4500 "Thumper" served as the basic building block, and 42 of them (later 20 more were added) were clustered and also connected directly to the main Infiniband network of the compute nodes allowing gigabyte level data transfer capability per each node. On top of this we implemented the Lustre parallel file system so that the accesses can be done in parallel and very large files could be handled. This allowed us to achieve over 10 gigabytes per second I / O performance, in 1.0 (later 1.5) petabyte storage of amazingly compact dimensions.

(F) **A Batch scheduler that was fair and easy to understand, assuming simultaneous use by hundreds of both Capacity and Capability users:** TSUBAME's user base of 2000 typically had 100 or more users being logged into the system and running jobs at the same time. The annual number of jobs exceed a million in count, and varied greatly in their usage patterns — number, memory size, time (makespan), parallelism, I / O performance, required QoS, I / Operformance, etc. Moreover, there was a need to support both the experts with highly-parallel, "capability" jobs, versus novice users sometimes with parameter survey "capacity" jobs. Despite the large computational capacity of Tsubame1, it possessed nowhere near the amount of resources to satisfy all of these requirements. So, it was essential to adopt a easy-to-understand and seemingly-fair scheduling policy that would satisfy the above goals, such as co-existence and QoS control of pay-per-use versus flat-rate usage models, pay-for-priority-QoS, as well as later introduction of job reservation for very large jobs, etc. All such requirements were incorporated as customization modules of the Sun GridEngine batch scheduler, and was continuously updated and improved due to user feedback.

Besides these issues, there were a variety of technical and operational ideas we had incorporated into the system. In fact every year we reviewed the hardware, software, and operational aspects of the system and when necessary conducted additional procurements to augment the system. These were not merely constrained to simple issues such as additional software licenses, but some were directly relevant to the experimental operation we had mentioned as item (2) earlier, such as the addition of GPUs.

The Dark Side of TSUBAME1 and their Reflections in TSUBAME 2.0 4

Some of the problems identified in TSUBAME1 in its performance and operations were more fundamental, and found not to be solvable with only superficial improvements described above. Moreover, supercomputing is a field where the average performance of a machine increases by approximately 180% each year, well beyond the Moore's law. As such it was obvious that there would be a need for continuous research and development of various technologies merely to sustain such a growth level. Although some of the technological choices were done to the best of our knowledge circa 2004-5 when TSUBAME1 was being designed, it became apparent not all choices we made were the best leading to TSUBAME2.0's operations in 2010. Below we give such shortcomings, and how they were analyzed and attempts were made to solve the problem for TSUBAME2.0:

(A) **Drastic improvement of power/performance:** TSUBAME1.0 consumed at the peak of about 1MW electrical power, or 10% or more of the entire power usage of the Tokyo Institute of Technology's Oo-okayama main campus, costing about 100 million yens per year. Since TSUBAME2.0 was initially planned to be deployed in the early part of 2010, its target performance would be $(1.8)^4$, or about 10 times speedup at 1 petaflop while maintaining the power usage. However, recall that the power/performance ratio for TSUBAME1.0 was already very efficient for the supercomputer of the time, thanks to the ClearSpeed card for dense linear algebra operations. The University of Tokyo's T2K supercomputer that came out two years later but without using any type of accelerators, consumed about the same power as TSUBAME2.0, while in Linpack being only about 20% faster, instead of being over 3 times faster as the 180% growth should indicate. Thus, In order for TSUBAME1 to maintain international competitiveness, it became necessary to improve the power performance ratio significantly. Fortunately, the our project sponsored by the JST-CREST program, called "ULP-HPC: High Performance Ultra Low-Power" was approved, with the aggressive goal of improving the power efficiency of supercomputers by x1000 instead of x100 as is with the Moore's law, in 10 years. Such boost in basic research accelerated our understanding of low power design of supercomputers, and some of the research results were applied to TSUBAME2.0.

(B) **Widening the applicability and use of Accelerators (GPUs):** On related terms, we found that acceleration using GPUs were the key to low power and high performance, however, early accelerators lacked the applicability to the wide range of existing applications. Some were resulting from the lack of algorithms and software, but some were from the hardware architecture itself. In fact, early accelerators were quite limited in their applicability due to their hardware limitations; for example, ClearSpeed excelled in dense matrix type problems, but found to be poor at bandwidth-hungry problems. These were principally due to the lack of memory capacity and a shortage of available hardware memory bandwidth as well as programming difficulties; and as a result narrowed its applicability substantially. Fortunately, GPUs sporting both high computational density and high memory bandwidth, and being low cost, were becoming rapidly general purpose in terms of both its hardware and software, and in the labs we had already demonstrated its advantages and flexibilities in many real-world HPC situations. Therefore, the question was how much of GPU technology would be applicable to TSUBAME2.0; fortunately, we had initiated technical partnerships and discussions in 2007 with two key companies, namely NVIDIA and Microsoft, and we were able to conduct projects which resulted initially in a 128GPU prototype cluster in late 2007, and based on the results from that machine, we were able to augment TSUBAME1 with 680 cards of the latest NVIDIA Tesla GPU (Figures 3) in October 2008. The resulting supercomputer, TSUBAME1.2 formed a platform which allowed a variety of experiments in our preparations for TSUBAME2.0 which was to become extremely GPU centric (Figure 4) due to many positive results we were obtaining and the problems we encountered and solved in its operations.

Figure 3 NVIDIA Tesla s1070 GPU retrofitted to TSUBAME1

Figure 4 NVIDIA Tesla M2050 GPU used in TSUBAME2.0, 3 cards per node, over 4000 cards in total.

(C) **Lack of memory and network bandwidth in the node:** Not only the local memory bandwidth that the GPU resolved, another problem was the overall bandwidth of the system being somewhat deficient in TSUBAME1 then we initially anticipated. CPU memory bandwidth was affected by two negative factors, the decline of the memory bus clock with a larger number of multi-channel memory; and the inherent limit for the Socket 940 and Socket F generations of AMD CPU design, where the total memory bandwidth of the shared memory was limited to be approximately 20GB/s due to memory coherency traffic. As for the network, Infiniband theoretical channel performance of 1GB/s was seeing strong decrease to be half or less due to overlap with other activities within the node. As will be discussed in the part 2 of this article, bandwidth is necessary not only to significantly improve acceleration of the current applications, but will also be an important factor for ensuring future scalability of the system. Thus, in TSBAME2.0's design, increase of bandwidth exceeding the increase in compute FLOPS was absolutely necessary, and deemed to be the most important technological improvement; this is one of the major reasons why an entirely new node design was mandated, instead of using existing products.

(D) **Lack of network bisection bandwidth:** In addition, bisection bandwidth in TSUBAME1 was lacking; whereas the aggregate injection bandwidth at the endpoint was endpoint was 13 terabits/second, due to the constrained fat-tree bisection bandwidth was approximately 2.8 terabits/second, or about 5 to 1 oversubscription ratio. This has resulted in the weakness (e.g., in comparison to the Earth Simulator) in supporting various global-style algorithms such as global FFT, in which the communication bottleneck would become the dominant performance inhibitor. Therefore, it was imperative to achieve high bandwidth across the network. A new network design was needed to achieve full bisection in the same manner as the Earth Simulator, while sporting more than more than twice the number of nodes. As a result of various design studies, combined use of a smaller leaf switches

with very large centralized core switch fabric that were physically clustered, with optical fiber connection in-between comprising a full fat-tree configuration, was deemed to be the smallest, most cost effective, and the most reliable (Figure 5). One requirement however was to make the system as small as possible to shorten the fiber cable length.

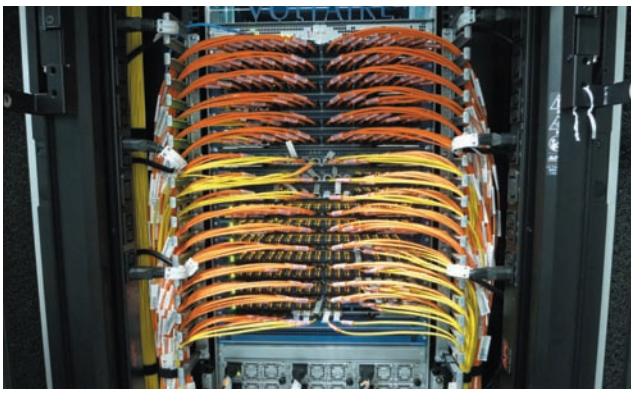


Figure 5 Fiber optic cables are aggregated into the large Infiniband core switch fabric from the the lower-tier edge switches

Figure 6 Individual compute nodes in TSUBAME2.0 are 1/4th the size of TSUBAME1 nodes, despite being 10 times more computationally powerful.

(E) **Cooling efficiency issues:** TSUBAME1.0 employed advanced cooling strategy of the time after over a year of study and given the physical confines of relative small space we had for our computer room, such as achieving the hot-row vs. cold-row separation using specially placed CRC units as well as ducts in the ceiling instead of floor cooling etc. The achieved PUE (Power Usage Effectiveness) was approximately 1.44, which was a reasonably good measure at the time. However, it was a fact that the cooling power consuming 44% beyond the machine power was not ideal indeed, and moreover, we knew that some of the recent advances will allow us to achieve much lower PUE. As a result, hybrid water- and air-cooled technology, where the individual rack would be cooled by water, and the circulatory air being sealed and contained to cool the nodes was employed, with an aim to achieve PUE as good as 1.2.

(F) **Machine size and weight problems:** TSUBAME1 nearly occupied 80 racks to their entirety, occupying most of the computer room floorspace of GSIC's facilities building. This seriously compromised our ability to scale further, as well as complicating the wiring and control logistics spanning two floors. These and for other reasons we have mentioned constructing a machine that was dense with very small resulting footprint was becoming an important issue. Making a machine smaller not only reflects in cost, but would improve our ability to construct full bisection and fat networks, as the machines need to be located to the central switch fabric as close as possible. Fortunately, in order to accommodate multiple GPU for high bandwidth fat-node configuration necessitated an entirely new node design, and major technological goal of node implementations was set to achieve unprecedented levels of density (Figure 6). The result achieved is 50 Teraflops per rack performance density, i.e. each rack being approximately the compute performance of the entire Earth Simulator, while the entire TSUBAME2.0 occupying about 60 racks, or about 3/4th of TSUBAME1 despite the 30-fold increase in computational performance.

(G) **Lack of operational storage capacity:** The storage capacity TSUBAME1 was significantly improved over its predecessor, but still the actual production capacity was always lacking. This depended on various factors: tertiary storage systems of tapes or MAIDs were lacking, and moreover, all the storage was uniformly high-performance and somewhat expensive as a result, despite some being purely used for backups. Secondly, there were various single points of failures in the storage, including the Lustre parallel file system, necessitating

duplication in various parts of the storage system. However, this resulted in an entire Thumper box used up for metadata management, being expensive in terms of required capacity and sacrifices in performance. Thumpers were consumed for other service requirements; moreover, we had to deploy RAID6 with relatively small number of stripes, further constraining space. As a result of all of these requirements, the capacity of high performance Lustre storage area shrunk down to be miniscule, less than 100 Terabytes out of the original 1 Petabytes. In 2007 we alleviated the problem somewhat by adding 20 more Thumper units, and the total raw physical capacity was raised to 1.6 petabytes; still the total Lustre capacity including the scratch space for Gaussian was only about 200 terabytes in total. Therefore, TSUBAME2.0 storage was designed to retain as much space as possible for operational Lustre and GPFS storages by extensive streamlining of the storage management, and improving the reliability through redundancy and elimination of single points of failures without sacrificing storage capacity. In particular, we had started improving TSUBAME1's storage fabric in 2009 in preparation, incorporating dedicated storage management servers, as well as introducing a large capacity tape system that had the ability to be extended to beyond 10 petabytes, to better match the planned operational disk capacity of 7 petabytes for TSUBAME2.0. (Figures 7, 8, 9)

(H) **Lack of storage bandwidth:** In order to achieve the initially determined performance goal of 2-3 petaflops, it was deemed that we would require I/O speeds of several hundred gigabytes per second. However, implementing such I/O performance would require a huge number of spinning disks and storage servers. Fortunately, studies indicates that major portions of I/O writes are stream writes for checkpoints, or local scratch writes for simple scratch files or more serious data structures for out-of-core algorithms [4][5]. Assuming that approximately 80-90% of I/O workload would be of such nature, the achieved speed of hundreds of MB/s by a new breed of SSDs with their improved capacity, cost performances, and reliability seemed perfect for the task to relieve the parallel file systems of such loads and effectively attaining x5-x10 effective improvement in I/O performance. In practice, the aggregate I/O performances of two or more SSDs equipped in every node of TSUBAME2.0 (Figure 10) totals more than 660 gigabytes / second; work is underway to achieve reliability of local writes while maintaining performance using redundant coding techniques.

A **Figure 7** Various storage servers in TSUBAME 2.0 for Lustre, GPFS, NFS, CIFS, etc.

B **Figure 8** Each disk enclosure contains 60 units of 2 terabyte SATA HDDs.

C **Figure 9** TSUBAME2's SL8500 tape drive system, embodying over 10,000 tape slots and located in a separate building.

D **Figure 10** TSUBAME 2.0 embeds two or more SSD drives on each node 60GB, or 120GB depending on the node memory capacity.

(1) **Further improvements in reliability:** TSUBAME1 was already designed with high reliability in mind. In fact, the entire history of failures and repairs log was made constantly accessible via a web page of the GSIC center, and the only time the system was entirely down was twice in 4.5 years, once when a major power outage occurred around a small area South of Tokyo where Tokyo Tech, just happened to be affected for two hours. However localized faults that affect sizeable portions of the system did occur occasionally, including those that would affect one of the batch queues entirely. Significant lessons had been learned for TSUBAME2.0 design as a result, eliminating wherever possible single points of failures in batch queues, storage/parallel file system, various service nodes, etc. Despite the costs incurred by such redundancies, it was deemed that the user's lost time due to failures outweighed the costs, especially as the faults themselves would become bigger hindrances as we move on to future machines that will be more error prone.

TSUBAME 2.0's will now Begin — Towards Part 2 5

We have outlined the evolutionary path from TSUBAME1 to TSUBAME2.0. As of this writing TSUBAME1.0 continues to be shrunk down due to the lack of power in the building, while TSUBAME2.0 benchmarking and burn-in continues for services to begin on Nov. 1, 2010, whence TSUBAME1.0 will be totally shut down on Oct. 25th, 2010. Since much of the software stack will be carried over from TSUBAME1, users should feel quite at home with TSUBAME2.0, just bigger, better, and somewhat faster when one would run non-GPU programs. When a user would port his application to GPUs and exploit other performance capabilities such as the intra-node SSDs, the applications would fully be able to utilize the 30-fold speedup it had achieved, perhaps to compute in petaflops, along with other new features TSUBAME2.0 newly offers compared to TSUBAME1, some of which had been outlined in this article. Users should then should realize we would be in the dawn of petaflops, moving towards exaflops in the coming years.

Part 2 of this article will evaluate TSUBAME2.0 from the viewpoint of scaling towards exaflops, plus benchmarks to observe the trends thereof. Please stay tuned.

References

[1] Satoshi Matsuoka, Toshio Endo, Naoya Maruyama, Hitoshi Sato, Shin'Ichiro Takizawa. "The Total Picture of TSUBAME2.0", The TSUBAME E-Science Journal, Tokyo Tech. GSIC, Vol. 1, pp. 16-18, Sep. 2010

[2] Satoshi Matsuoka. Petascale Computing Algorithms and Applications --- Chapter 14 The Road to TSUBAME and Beyond, Chapman & Hall CRC Computational Science Series, pp.289-310, 2009.

[3] Toshio Endo and Satoshi Matsuoka. "Massive Supercomputing Coping with Heterogeneity of Modern Accelerators", In Proc. 22nd IEEE International Parallel & Distributed Processing Symposium (IPDPS 2008), The IEEE Press, Miami, FL, April 2008, pp.1-10, DOI: 10.1109/IPDPS.2008.4536251

[4] Weikuan Yu, Jeffrey S. Vetter, H. Sarp Oral. "Performance Characterization and Optimization of Parallel I/O on the Cray XT", In Proc. 22nd IEEE International Parallel & Distributed Processing Symposium (IPDPS 2008), The IEEE Press, Miami, FL, April 2008, pp.1-10, DOI: 10.1109/IPDPS.2008.4536277

[5] Julian Borrill, Leonid Oliker, John Shalf, Hongzhang Shan, Andrew Uselton. "HPC Global File System Performance Analysis Using A Scientic-Application Derived Benchmark", Parallel Computing, Volume 35 , Issue 6 (June 2009), pp. 358-373.

Multi-GPU Computing for Next-generation Weather Forecasting

- 145.0 TFlops with 3990 GPUs on TSUBAME 2.0 -

Takashi Shimokawabe* **Takayuki Aoki****

* Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology
**Global Scientific Information and Computing Center, Tokyo Institute of Technology

In order to drastically shorten the runtime of the weather-prediction code ASUCA , developed by the JMA (Japan Meteorological Agency) for the purpose of the next-generation weather forecasting service, the entire parts of the huge code were rewritten for GPU computing from scratch. By introducing many optimization techniques and several new algorithms, very high performance of 145 TFlops has been achieved with 3990 GPUs on TSUBAME 2.0 Supercomputer. It is quite meaningful to show that the GPU supercomputing is really available for one of the major applications in the HPC field.

1 Introduction

Weather forecasting is an indispensable part in our daily lives and business activities, needless to say for natural disaster preventions. The atmosphere has a very thin thickness compared with the Earth diameter. In the previous atmosphere code, the force balance between the gravity and the pressure gradient in the vertical direction was used to produce a hydrostatic model. Recently it is widely recognized that the vertical dynamical processes of the water vapor should be taken into consideration in cloud formations. A three-dimensional non-hydrostatic model describing up-and-down movement of air has been developed in weather research.

For weather simulations, the initial data is produced by assimilating many kinds of observed data and simulation results based on the four-dimensional variational principle. Since the weather phenomena are chaotic, the predictability period is less than several days for one set of initial data hence the jobs run sequentially updating the initial data.

In recent years, it is highly demanded to forecast detailed weathers such as unexpected local heavy rain, and high resolution non-hydrostatic models are desired to be carried on fine-grained grids.

2 GPU computing for weather simulations

A computational heavy load is required to run the high-resolution weather models. As a reference the WRF [1] has been scored at 50 TFlops on the current fastest supercomputer in the world [2]. WRF is a next-generation atmosphere simulation model (Weather Research and Forecasting), a world standard code developed at the NCAR (National Center for Atmospheric Research), UCAR (University Corporation for Atmospheric Research) and others in the United States and supported by worldwide researchers.

Numerical weather models consist of a dynamical core and physical processes. In the dynamical core, forecast variables such as winds, atmospheric pressure and humidity are calculated by solving fluid dynamics equations. The physical processes strongly depend on parametrizations related to such microphysics as condensation of water vapor, cloud physics, and rain. In the computation of the dynamics core, the memory access time is a major part of the elapsed time compared with the floating point calculations part and therefore is hard to get close to the peak performance in any computer. On the other hand, the physical processes include some computationally intensive parts demanding high performance of floating point calculations.

Graphics Processing Units (GPUs) have been developed for the rendering purpose of computer graphics. The request for high-level computer visualization makes the GPUs to have high performance of floating point calculation and wide memory bandwidth. Recently, exploiting GPUs for general-purpose computing, i.e. GPGPU, has emerged as an effective technique to accelerate many applications. After CUDA [3] was released by NVIDIA as the GPGPU-programming framework in 2006, it allowed us to use the GPU easily as an accelerator. In the area of high performance computing (HPC), it has been reported a lot of successful applications in Computational Fluid Dynamics (CFD), Fast Fourier Transform (FFT), molecular dynamics, astrophysics, bio-informatics and some others ran on GPUs dozens of times faster than on a conventional CPU.

In the numerical weather prediction, it was reported that a computationally expensive module of the WRF model was accelerated by means of a GPU[4, 5]. These efforts, however, only result in a minor improvement (e.g., 1.3× in [4]) in the overall application time due to the partial GPU porting of the entire code. Since the other functions (subroutines) run on the CPU and all the variables were allocated on the CPU main memory, it is necessary

to transfer the data between the host CPU memory and the GPU video memory through the PCI Express bus for every GPU kernel-function call. They reported that the acceleration for the microphysics module itself achieved a twenty times speedup. Physical processes are composed of small and relatively independent modules and are able to be easily replaced with other modules while a dynamical core computes time-integration of interdependent forecast variables thus GPU porting of parts of the dynamical core does not contribute to improvement of the performance.

As the successor of TSUBAME 1.2, TSUBAME 2.0 supercomputer equipped with more than 4000 GPUs has started operating in November 2010 and has become the first petascale supercomputer in Japan. Since TSUBAME 2.0 owes most of its computing performance to GPUs, it is a key issue to achieve high performance on GPU in many applications. In this article, we show the process of porting an operational weather prediction code to the GPU on TSUBAME 2.0 and demonstrate the performance for a practical operation size.

Next-generation weather prediction code ASUCA 3

ASUCA (Asuca is a System based on a Unified Concept for Atmosphere) is a next-generation high resolution mesoscale atmospheric model being developed by the Japan Meteorological Agency (JMA)[6]. ASUCA succeeds the Japan Meteorological Agency Non-Hydrostatic Model (JMA-NHM) as an operational non-hydrostatic regional model at the JMA.

First, we have implemented the dynamical core as the first step toward developing the full GPU version of ASUCA. In ASUCA, a generalized coordinate and flux-form non-hydrostatic balanced equations are used for the dynamical core. The time integration is carried out by a fractional step method with the horizontally explicit and vertically implicit (HE-VI) scheme [7]. One time step consists of short time sub-steps and a long time step. The horizontal propagation of sound waves and the gravity waves with implicit treatment for the vertical propagation are computed in the short time step with the second-order Runge-Kutta scheme. The long time step is used for the advection of the momentum, the density, the potential temperature, the water substances, the Coriolis force, the diffusion and other effects in the physical processes with the third-order Runge-Kutta method. The above matters are almost as same as those employed in the WRF model. In the present ASUCA, the physical core is still being developed and a Kessler-type warm-rain model has been implemented for the cloud-microphysics parameterization describing the water vapor, cloud water, and rain drops.

Single GPU Implementation and Performance 4

Although our final destination is to develop the multi-GPU version of ASUCA, we start from the single GPU implementation and show its performance. Figure 1 illustrates the computational flow diagram. In the beginning of the execution, the host CPU reads the initial data from the input files onto the host memory, and then transfers it to the video memory on the GPU board. The GPU carries out all the computational modules inside the short time-step and long time-step loops. When the forecast data is completed on the GPU, a minimal data is transferred to the host CPU memory to reduce the communication between CPU and GPU.

4-1 Optimizations

In order to improve the performance on the GPU, we have introduced several optimizations when implementing the code in CUDA. We focus on two components as examples: (a) the advection computation (b) the 1-dimensional Helmholtz-type equation for the pressure.

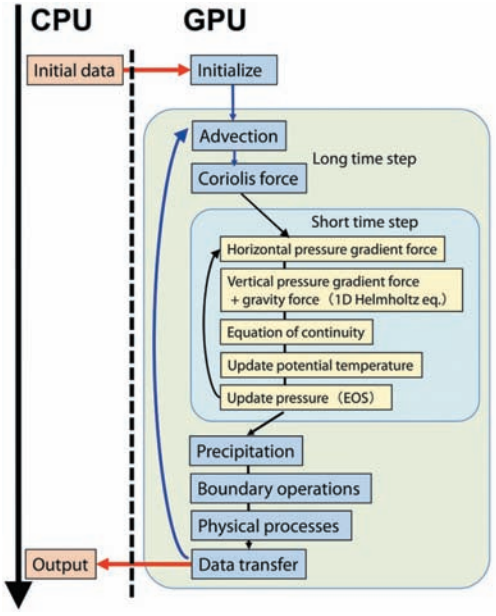


Figure 1 Computational flow diagram of the ASUCA. All the modules inside the short time-step and the long time-step loops are executed on the GPU.

(a) Implementation of the advection computational modules

The 3-dimensional advection computation is strongly memory bound and it is very effective to reduce the access to the video memory (called global memory in CUDA) in order to improve the performance. We make use of the shared memory as a software-managed cache, which is shared among threads in a block in the CUDA programming. For a given grid size (nx, ny, nz) of the computational domain, the GPU kernel function is invoked with (nx/64, nz/4, 1) blocks with (64, 4, 1) threads. The z direction in physical space is mapped to the y direction in the CUDA code. Each thread specifies a point (x, z) and calculate the advection equation on this grid point from j = 0 to j = ny − 1 marching in the y direction as shown in Fig. 2(a). The block size of (64, 4, 1) threads is derived from performance optimization.

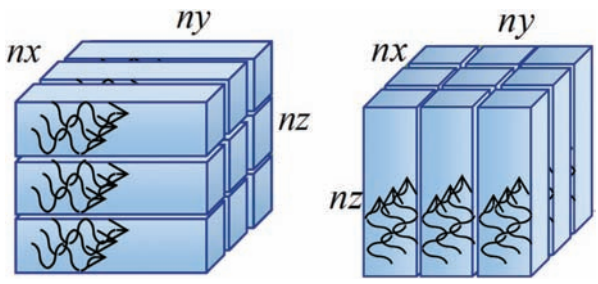
The discretization of the advection equation has a four-point stencil in each direction on the mesh. Each block holds an array of (64+3) x (4+3) elements on the shared memory. When a block computes a xy plane of the computational domain, the variable data on the global memory is copied to the shared memory to be shared among threads in the block. On the other hand, the stencil access in the y-direction is closed in the thread by marching the computation in the y-direction. The variable data of the global memory are stored in the registers (temporal variables). When we compute the j+1-th plane, the data have been already stored in the registers from the global memory in computing the j-th plane. In our implementation, we copy the data in the registers to the shared memory and reuse them without same accesses to the global memory [8].

(b) Implementation of the Helmholtz-type pressure equation

Due to the HE-VI splitting, the pressure equation reduces to a 1-dimensional Helmholtz-type elliptic equation in the vertical (z) direction. The discretization of the equation is expressed by a tri-diagonal matrix. It is possible to apply the TDMA algorithm to solve the matrix however we have to calculate the elements sequentially in the z-direction. Figure 2(b) shows the data parallel by marching the sequential calculation in the z-direction.

4-2 Performance

Since ASUCA is being developed in FORTRAN language at the JMA, the GPU code has to be developed from scratch in CUDA. Before implementing ASUCA on GPU, we rewrote the Fortran ASUCA code to C/C++ language because we changed the element order of the 3-dimensional array to improve the memory access performance of the GPU computing. In order to measure the performance of floating-point operations on a GPU, we count the number of floating-point operations of the CPU-based ASUCA by running it on a CPU with a performance counter provided by PAPI



(a) Advection computation (b) Helmholtz-type pressure equation

Figure 2 Configuration of the CUDA blocks and marching directions of the threads.

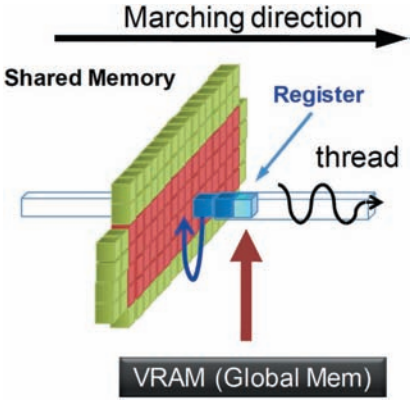


Figure 3 Usage of the shared memory and the register to reduce the access to the global memory.

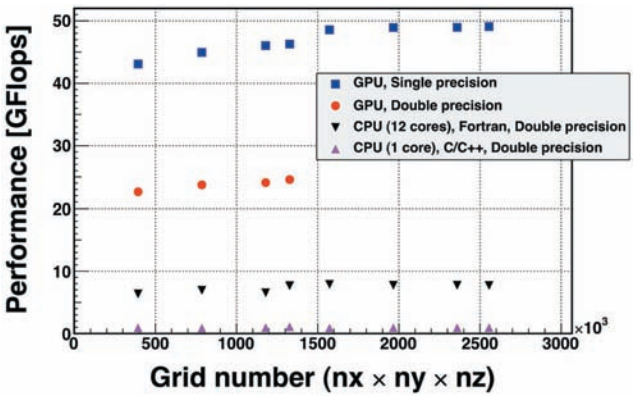


Figure 4 Single GPU performance on a NVIDIA Tesla M2050 of TSUBAME 2.0 and CPU performance of the Intel Xeon X5670.

(Performance API) [9]. By using the counts and the GPU elapsed time, the performance of the GPU computing is evaluated.

In all the cases, we fixed the grid number nx = 256 and nz = 48 of the computational domain and varied the number ny 32 to 208. The performance was measured in both single- and double- precision floating-point calculation using a NVIDIA Tesla M2050 in TSUBAME 2.0. The results of the GPU performance are shown in Figure 4. We achieved 49.1 GFlops in single precision for 256 × 208 × 48 mesh on a single GPU. In the double precision, the performance has about half of that of the single precision. As references, the performances on an Intel CPU (Xeon X5670 (Westmere-EP) 2.93 GHz 6 core x2: total 12 cores and 1 core) are plotted on the same graph. The original FORTRAN code was compiled by the Intel ifort compiler. It is found that the single-GPU performance achieved a six times speedups in comparison with 2 sockets of the CPU performance of the Intel Xeon X5670 in double precision.

Multi-GPU Computing 5

A GPU Tesla M2050 card on TSUBAME 2.0 has only 3-GB on-board video memory, which can hold up to a grid size 256 × 208 × 48 when running ASUCA. For large-size problems, it is necessary to use multiple GPUs beyond the video memory on a single GPU. The current operation for the weather forecast at the JMA utilizes the grid of size 721 x 577 x 50.

We decompose the given computational domain in both the x- and y-directions (2-D decomposition) and allocate the sub domain to each GPU since the z-directional mesh size is relatively small. Because GPUs cannot directly access the data stored on the global memory of other GPUs, the host CPUs are used to bridge GPUs for the exchange of the boundary data between the neighbor GPUs. The process is composed of the following three steps: (1) the data transfer from GPU to CPU using the CUDA runtime library, (2) the data exchange between nodes with the MPI library, and (3) the data transfer back from CPU to GPU with the CUDA runtime library. In the case of multi-GPU computing, the data communication time with the neighbor GPUs is not ignored in the total execution time. The overlapping technique with the computation is available to hide the communication costs and achieves better performance with a large number of GPUs [10].

5-1 Performance of Multi-GPU Computing

Each node of TSUBAME 2.0 has three NVIDIA GPU Tesla M2050 attached to the PCI Express bus 2.0x16, two QDR Infiniband and two sockets with Intel CPU Xeon X5670 (Westmere-EP) 2.93 GHz 6-core. The nodes are connected to the fat-tree interconnection with 200 Tbps bi-section bandwidth. Each GPU handles a domain of 256 x 108 x 48 in double precision and 256 x 208 x 48 in single precision, respectively.

The multi-GPU performance of ASUCA on TSUBAME 2.0 is shown in Figure 5. Using 3990 GPUs we achieved an extremely high performance of 145.0 TFlops for the domain of 14368 x 14284 x 48 in single precision. The double precision performance is 76.1 TFlops for the domain of 10336 x 9988 x 48. It is confirmed to maintain a good weak scalability. To compare with the CPU performance, the performance of 3990 is compatible with 3990x50 CPU cores.
Figure 6 demonstrates a real case of ASUCA operation with both the real initial and the boundary data used for the current weather forecast at the JMA. This simulation was performed with a 4792 × 4696 × 48 mesh with horizontal mesh resolution of 500 meters using 437 GPUs of TSUBAME 2.0 in single precision.

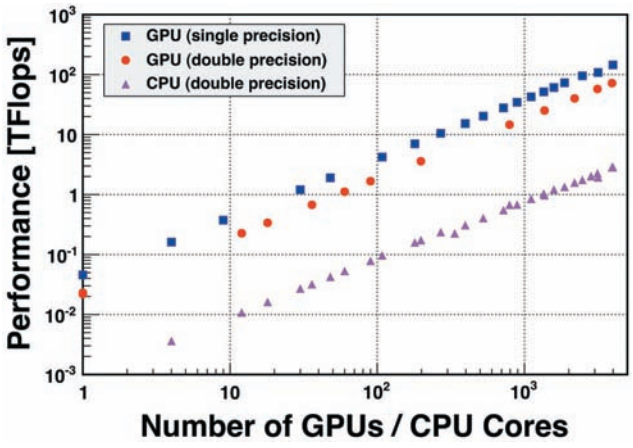


Figure 5 Multi-GPU performance of ASUCA on TSUBAME 2.0 comparing with the CPU.

Figure 6 ASUCA real operation to describe a typhoon with 4792 × 4696 × 48 mesh using 437 GPUs of TSUBAME 2.0

Summary 6

The full GPU implementation of the next-generation production weather code ASUCA was carried out on the TSUBAME 2.0 supercomputer in the Tokyo Institute of Technology. GPU offers not only extremely huge computational performance but also big advantages in respect of the cost and power consumption. It is really meaningful that the numerical weather prediction, one of the typical applications for a practical purpose, is provided as a successful example of the GPU supercomputing. In China and other countries, large-scale GPU computing on supercomputers is about to begin. TSUBAME 2.0 in the Tokyo Tech Global Scientific Information and Computing Center (GSIC) is the first over-petaflops supercomputer in Japan. In the GPU supercomputing era, it is expected that research communities become large and active to achieve great outcomes on GPU supercomputers.

Acknowledgements

We would like to thank Dr. Chiashi Muroi, Dr. Junichi Ishida and Dr. Kohei Kawano at the Japan Meteorological Agency for providing the original ASUCA code and helping us develop the GPU version. We are grateful to Prof. Satoshi Matsuoka, Prof. Toshio Endo, Dr. Akira Nukada and Dr. Naoya Maruyama at the Tokyo Institute of Technology for helping us to use TSUBAME 2.0.

This research was supported in part by the Global Center of Excellence Program "Computationism as a Foundation for the Sciences" and KAKENHI, Grant-in-Aid for Scientific Research (B) 19360043 from the Ministry of Education, Culture, Sports, Science and Technology (MEXT) of Japan, and in part by the Japan Science and Technology Agency (JST) Core Research of Evolutional Science and Technology (CREST) research program "ULP-HPC: Ultra Low-Power, High-Performance Computing via Modeling and Optimization of Next Generation HPC Technologies".

References

[1] W. C. Skamarock, J. B. Klemp, J. Dudhia, D. O. Gill, D. M. Barker, M. G. Duda, X. Y. Huang, W. Wang, and J. G. Powers, "A Description of the Advanced Research WRF Version 3," National Center for Atmospheric Research, (2008)

[2] A. S. Bland, R. A. Kendall, D. B. Kothe, J. H. Rogers, and G. M. Shipman, "Jaguar: The world's most powerful computer," in 2009 CUG Meeting, pp. 1–7. (2009)

[3] "CUDA Programming Guide3.2," http://developer.download.nvidia.com/compute/cuda/ 3_2/toolkit/docs/CUDA_C_Programming_Guide.pdf, NVIDIA, (2010)

[4] J. Michalakes and M. Vachharajani, "GPU acceleration of numerical weather prediction." in IPDPS. IEEE, pp. 1–7, (2008)

[5] J. C. Linford, J. Michalakes, M. Vachharajani, and A. Sandu, "Multi- core acceleration of chemical kinetics for simulation and prediction," in SC '09: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis.

New York, NY, USA: ACM, pp. 1–11, (2009)

[6] J. Ishida, C. Muroi, K. Kawano, and Y. Kitamura, “Development of a new nonhydrostatic model “ASUCA” at JMA,” CAS/JSC WGNE Reserch Activities in Atomospheric and Oceanic Modelling, (2010)

[7] W. C. Skamarock and J. B. Klemp, “Efficiency and Accuracy of the Klemp-Wilhelmson Time-Splitting Technique,” Monthly Weather Review, vol. 122, pp. 2623–+, (1994)

[8] P. Micikevicius, “3D finite difference computation on GPUs using CUDA,” in GPGPU-2: Proceedings of 2nd Workshop on General Purpose Processing on Graphics Processing Units. New York, NY, USA: ACM, pp. 79–84, (2009)

[9] S. Browne, J. Dongarra, N. Garner, G. Ho, and P. Mucci, “A portable programming interface for performance evaluation on modern processors,” Int. J. High Perform. Comput. Appl., vol. 14, no. 3, pp. 189–204, (2000)

[10] T. Shimokawabe, T. Aoki, C. Muroi, J. Ishida, K. Kawano, T. Endo, A. Nukada, N. Maruyama, and S. Matsuoka, "An 80-Fold Speedup, 15.0 TFlops Full GPU Acceleration of Non-Hydrostatic Weather Model ASUCA Production Code", in SC ’10: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis. New York, NY, USA: ACM (2010) in press

Computer prediction of protein-protein interaction network using MEGADOCK

- application to systems biology -

Yuri Matsuzaki* Masahito Ohue* Nobuyuki Uchikoga*
Takashi Ishida* Yutaka Akiyama*

* Graduate School of Information Science and Engineering, Tokyo Institute of Technology

We have developed a high throughput and ultra-fast protein-protein interaction (PPI) prediction system "MEGADOCK" on TSUBAME. MEGADOCK showed significant speed-up in the rigid-body docking process that leads a way of full utilization of protein tertiary structure data for large-scale and network-level problems in systems biology. We then have explored biological PPI networks with the method of bioinformatics. This is a grand challenge to solve the mega-order problems in medicinal science and drug discovery.

Introduction 1

Living matters are maintained and developed by various molecular interactions. Elucidation of the regulatory relations among the thousands of protein species working in a human cell is crucial for understanding the mechanisms underlie diseases and for development of drugs. We are working on the problem of predicting Protein-Protein Interaction (PPI) network (Figure 1) which is one of the main topics in systems biology, by using bioinformatics methods.

Conventionally, computational methods have been used mainly to analyze the mechanism of individual known protein interactions in detail. Those methods are not applicable to a large-scale analysis such as in systems biology field. In this work, we developed a method that can be applied to PPI prediction problems of mega-order data.

We proposed a novel score function to reduce calculations required for protein docking that incorporates shape complementarity and electrostatic interaction between the target proteins. Using this simple model, we designed a PPI prediction system "MEGADOCK" as to be capable of running on large-scale parallel computing systems like TSUBAME.

The proposed method can be used as one of the basic bioinformatics tools given the situation where we can use thousands to ten thousands of CPU cores.

Docking calculation 2

MEGADOCK includes a rigid-body docking system that searches relevant interacting protein pairs among the target protein tertiary structures. The rigid-body docking process is mainly based on the shape complementarity of the target proteins without considering conformational changes under the protein complex formation event.

Docking calculation part, the core of MEGADOCK system, calculates docking scores by an equation consisting of the shape complementarity term *G* and the electrostatic interaction term *E*. The target protein pair R (receptor) and L (lignad) are first allocated on the 3-D voxel space, 1.2 Å × 1.2 Å × 1.2 Å. Then the scores are assigned to each voxel according to the location in a protein, such as surface and core. We use our original scoring function called "real Pairwise Shape Complementarity" (rPSC) for the shape complementarity term *G* as follows [1]:

$$G_R(l,m,n) = \begin{cases} \#\text{ of R atoms within } (3.6\text{Å} + \text{R atom } r_{vdW}) \\ -27 \quad \text{inside of the R} \end{cases}.$$

$$G_L(l,m,n) = \begin{cases} 0 & \text{solvent accessible surface layer of the L} \\ 1 & \text{solvent excluding surface layer of the L} \\ 2 & \text{core of the L} \\ 0 & \text{open space} \end{cases}.$$

The advantage of rPSC is that it is a real number representation (Figure 2), thus we can put a physicochemical parameter into the imaginary part. Then it is possible to calculate a score with only one complex number for each voxel.

In order to search best docking poses, a ligand molecule is moved through all voxel space (scores under translation are effectively calculated as the convolution sum), and possible ligand

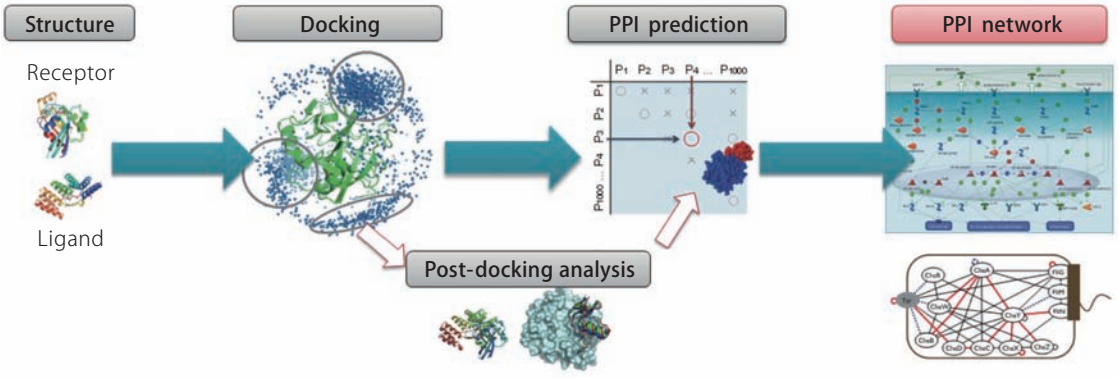


Figure 1 PPI network prediction.

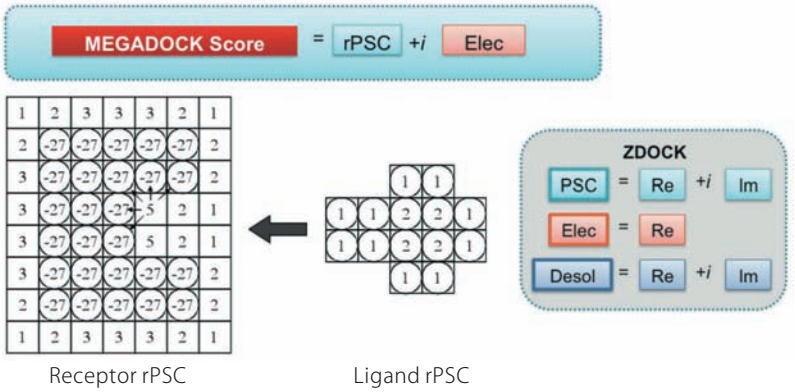


Figure 2 rPSC and docking score.

orientations are exhaustively examined with 3,600 rotation angles with 15 degree steps in default.

For a physicochemical parameter, we use electrostatic interaction of each amino acid, $E_R(l, m, n)$ and $E_L(l, m, n)$ in Figure 2, which are calculated using the interaction energy in each voxel, $q(l, m, n)$, with using CHARMM19[2] for the electrostatic charge of each atom.

Considering these terms, the overall docking score S is calculated as follows:

$$R(l,m,n) = G_R(l,m,n) + iE_R(l,m,n).$$
$$L(l,m,n) = G_L(l,m,n) + iwE_L(l,m,n).$$
$$S(\alpha,\beta,\gamma) = \mathcal{R}\left[\sum_{l=1}^{N}\sum_{m=1}^{N}\sum_{n=1}^{N} R(l,m,n)L(l+\alpha, m+\beta, n+\gamma)\right].$$

In a direct execution of simple convolution sum, $O(N^6)$ calculations are required. On the other hand, the calculation order using fast Fourier transform (FFT) algorithm both for discrete Fourier transform (DFT) and inverse discrete Fourier transform (IFT) is reduced to $O(N^3 \log N)$ [3]. The score S for FFT is:

$$S(\alpha,\beta,\gamma) = \mathrm{IFT}[\mathrm{DFT}[\mathrm{R}(l,m,n)]^* \mathrm{DFT}[L(l,m,n)]].$$

By decreasing the number of required DFT/IFT operations with rPSC, MEGADOCK performs the docking process about four times faster than well-known protein-protein docking tool, ZDOCK[4,5], which uses three complex numbers for the score function, as shown in Figure 2.

Large-scale parallelization 3

MEGADOCK is parallelized with the MPI library. Because calculations for each pair are almost independent, we can parallelize an all-to-all exhaustive PPI prediction task in several ways on hundreds or thousands of processors. User can specify number of receptor and ligand protein data assigned for a single processor with considering memory capacity. When a processor is assigned for m receptors and n ligands data it then calculates FFT for the first

ligand with each possible rotation. The FFT results are repeatedly used for docking with all *m* receptors, in order to avoid redundant calculation. Then the process is repeated for *n* times totally.

MEGADOCK has an option to avoid DFT calculation and upload pre-calculated DFT results from "FFT protein structure library" on the hard disks. This approach is effective in a system with high I/O performance, and we recorded three times faster speed than simple exhaustive calculation on TSUBAME 1.2.

The FFT routine in MEGADOCK uses not only base-2 logarithm, but uses 2, 3, 5 as bases to minimize volume of a target 3-D cube. If we chose too many logarithm bases, we should prepare so many pre-calculated FFT models in the library, because protein paring is unknown a priori.

On the other hand, if we utilize GPU acceleration, it is better to simply repeat FFT calculation on GPU with most adequate combination of logarithm bases. We need to consider this point when we implement our system on TSUBAME 2.0 system.

Application to systems biology 4

We applied MEGADOCK system to a known set of 44 protein complex data adopted from a generally used benchmark of protein-protein docking, and evaluated the accuracy of the prediction. We predicted relevant interacting proteins from $44 \times 44 = 1{,}936$ combinations by docking and post-docking analysis. We obtained near-native complex structures as in the figure (Figure 3, above). And in the colored matrix (below), the red colored cells located on the diagonal line shows correctly predicted interacting pairs.

To further demonstrate the potential of MEGADOCK system, we evaluated our PPI prediction system by applying it to the data of bacterial chemotaxis pathway, which is one of the typical problems in systems biology (Figure 4) [6].

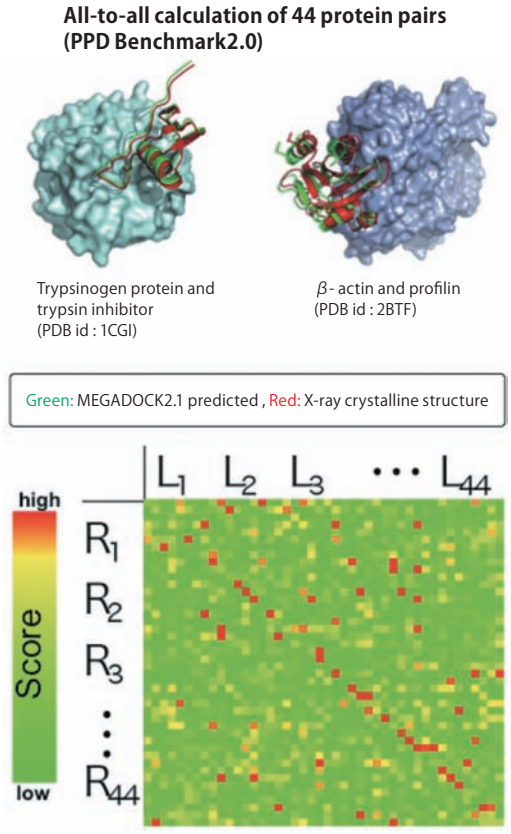


Figure 3 PPI prediction results: benchmark data

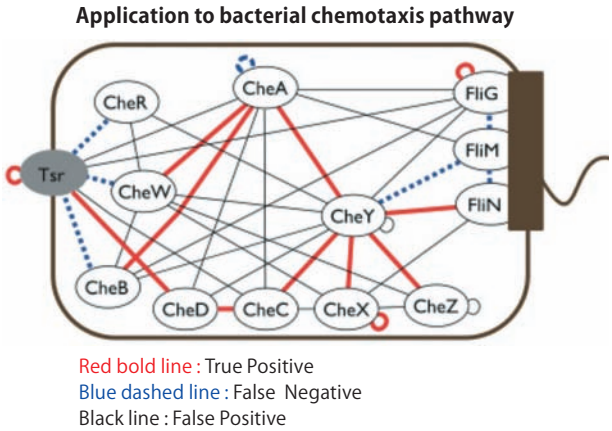


Figure 4 Application to systems biology: bacterial chemotaxis

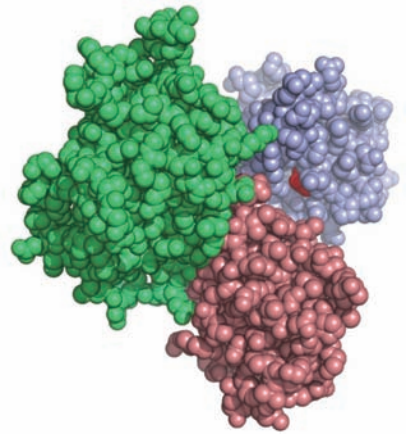


Figure 5 Predicted interactions among CheY-CheD-CheC

Enteric bacteria like *Escherichia coli* shows a behavior called chemotaxis that leads the cell towards nutritional substances. Because the signaling pathway of this system has been well-studied for decades, almost all PPIs needed to explain the functionality of the system are known. We used these known PPIs as relevant PPIs and evaluated the prediction performance of MEGADOCK by investigating chemotaxis pathway proteins. We conducted docking and post-docking analysis with MEGADOCK ver.2.1 using protein data (13 protein species, 89 structures) collected from a public protein structure database, PDB. We analyzed 89 × 89 = 7,921 combinations of 3-D structure data. We also discussed some unknown but interesting PPIs detected by the system (Figure 5) [6].

Conclusion 5

We implemented a high-throughtput and ultra-fast PPI network prediction system "MEGADOCK" on TSUBAME. MEGADOCK suits well with the large-scale computing environment. And we showed that our system could conduct network level analysis, which is common in systems biology research, and is faster than the conventional method. We will further exploit the system as to make mega-scale analyses easier, involving analyses of proteins in cancer cells and microbes. Now we are analyzing 500 × 500 data using protein data related to EGFR signaling which is important in human lung cancer processes.

Acknowledgments

This work was supported in part by a Grant-in-Aid for Research and Development of The Next-Generation Integrated Life Simulation Software, and a Grant-in-Aid for Scientific Research (B), 19300102, both from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT).

References

[1] Ohue M., Matsuzaki Y., Matsuzaki Y., Sato T., and Akiyama Y., MEGADOCK: an all-to-all protein-protein interaction prediction system using tertiary structure data and its application to systems biology study., IPSJ Transactions on Mathematical Modeling and its Applications (TOM), 3: 91-106, 2010.

[2] Katchalski-Katzir E, Shariv I, Eisenstein M, et al., Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques., Proc Natl Acad Sci U S A, 89:2195-9, 1992.

[3] Chen R., and Weng Z., Docking unbound proteins using shape complementarity, desolvation, and electrostatics., PROTEINS, 47:281-294, 2002.

[4] Chen R., Li L., and Weng Z., ZDOCK: an initial-stage protein-docking algorithm., PROTEINS, 52:80-87, 2003.

[5] Brooks BR, Bruccoleri RE, Olafson BD, et al., CHARMM: A program for macromolecular energy, minimization, and dynamics calculations., J Comput Chem, 4:187-217, 1983.

[6] Matsuzaki Y., Matsuzaki Y., Sato T and Akiyama Y., *In silico* screening of protein-protein interactions with all-to-all rigid docking and clustering: an application to pathway analysis., J Bioinform Comput Biol, 7:991-1012, 2009.

TSUBAME
e-Science Journal

● TSUBAME e-Science Journal No. 2

Published 11/10/2010 by GSIC, Tokyo Institute of Technology © ISSN 2185-6028
Design & Layout: Caiba & Kick and Punch
Editor: TSUBAME e-Science Journal - Editorial room
Takayuki AOKI, Toshio WATANABE, Masakazu SEKIJIMA, Thirapong PIPATPONGSA, Fumiko MIYAMA
Address: 2-12-1 i7-3 O-okayama, Meguro-ku, Tokyo 152-8550
Tel: +81-3-5734-2087 Fax: +81-3-5734-3198
E-mail: tsubame_j@sim.gsic.titech.ac.jp
URL: http://www.gsic.titech.ac.jp/

GSIC
Global Scientific Information and Computing Center

TSUBAME

International Research Collaboration

The high performance of supercomputer TSUBAME has been extended to the international arena. We promote international research collaborations using TSUBAME between researchers of Tokyo Institute of Technology and overseas research institutions as well as research groups worldwide.

Recent research collaborations using TSUBAME

1. Simulation of Tsunamis Generated by Earthquakes using Parallel Computing Technique
2. Numerical Simulation of Energy Conversion with MHD Plasma-fluid Flow
3. GPU computing for Computational Fluid Dynamics

Application Guidance

Candidates to initiate research collaborations are expected to conclude MOU (Memorandum of Understanding) with the partner organizations/ departments. Committee reviews the "Agreement for Collaboration" for joint research to ensure that the proposed research meet academic qualifications and contributions to international society. Overseas users must observe rules and regulations on using TSUBAME. User fees are paid by Tokyo Tech's researcher as part of research collaboration. The results of joint research are expected to be released for academic publication.

Inquiry

Please see the following website for more details.
http://www.gsic.titech.ac.jp/en/InternationalCollaboration

GSIC
Global Scientific Information and Computing Center