MC64: A web platform to test bioinformatics algorithms in a many-core architecture

Francisco José Esteban [1], David Díaz [2], Pilar Hernández [3], Juan Antonio Caballero [4], Gabriel Dorado [5], Sergio Gálvez [2]

[1] Servicio de Informática, Universidad de Córdoba, 14071 Córdoba (Spain)

[2] Dep. Lenguajes y CC. de la Computación, Universidad de Málaga, 29071 Málaga (Spain)

[3] Instituto de Agricultura Sostenible (IAS-CSIC), Al. del Obispo, 14080 Córdoba (Spain)

[4] Dep. Estadística, Universidad de Córdoba, 14071 Córdoba (Spain)

[5] Dep. Bioquímica y Biología Molecular, Universidad de Córdoba, 14071 Córdoba (Spain)

<fjesteban, ma1camoj, bb1dopeg>@uco.es, <david.diaz, galvez>@lcc.uma.es, phernandez@ias.csic.es

Abstract. New analytical methodologies, like the so-called "next-generation sequencing" (NGS), allow the sequencing of full genomes with high speed and reduced price. Yet, such technologies generate huge amounts of data that demand large raw computational power. Many-core technologies can be exploited to overcome the involved bioinformatics bottleneck. Indeed, such hardware is currently in active development. We have developed parallel bioinformatics algorithms for many-core microprocessors containing 64 cores each. Thus, the MC64 web platform allows executing high-performance alignments (Needleman-Wunsch, Smith-Waterman and ClustalW) of long sequences. The MC64 platform can be accessed via web browsers, allowing easy resource integration into third-party tools. Furthermore, the results obtained from the MC64 include time-performance statistics that can be compared with other platforms.

1 Introduction

The MC64 is a web platform allowing researches to test the performance of a 64-core technology with bioinformatics algorithms. Although the term "many-core" is usually applied to General-Purpose Graphics Processing Units (GPGPU), where each core has very limited resources, we will use this name to designate a processor with tens of Central Processing Unit (CPU) cores, each of them being able to independently execute a different operating system. Amongst them are the

Intel Single-chip Cluster Computer (SCC) with 48 cores (x86) [1] and the Tilera Tile64 with 64 Reduced Instruction Set Computing (RISC) cores [3] without floating point support. Other manufacturers focus on hybrid solutions using about tens of cores besides multi-threading, like the Intel Knights Ferry with 32 cores (x86) and four threads per core [2], the NetLogic XLP832 with eight MIPS64 cores and four threads per core [4] and the Sun UltraSPARC T2 with eight cores and a total of 64 threads [5]. The Tile64 processor is the only commercially available many-core System on Chip (SoC). It has been deployed in a Peripheral Component Interconnect Express (PCIe) card that can be boarded inside a conventional Personal Computer (PC) running a Linux operating system. This paper focuses on a TilExpress-20G PCIe card with 8 GB of DDR2 RAM memory and a Tile64 running at 866 MHz, hosted by a Dell T5400 workstation.

We have tested the performance of the Tile64 when running pairwise alignment algorithms like Needleman-Wunsch (NW) [6] and Smith-Waterman (SW) [7] to align DNA, RNA and peptide/protein sequences. Results show that the speedup can be up to 20x when compared with similar algorithms running in a Xeon Quad-core workstation with 8 GB of RAM [8]. This ultra-performance is achieved when the length of the sequences to align is very long; e.g., from 300 kilobases (kb) to 1,000 kb. On the other hand, the pairwise alignment is the basis of ClustalW [9], which is a multiple alignment algorithm divided in three stages. When aligning n sequences, the first of these stages requires the execution of $n(n–1)/2$ pairwise alignments (quadratic complexity) so, a decrease in the pairwise alignment time execution significantly decreases as well the total ClustalW execution time.

We have developed and parallelized the NW, SW and ClustalW algorithms (among others) to evaluate the bioinformatics potential of the Tile64 processor. They are available via web at <http://www.sicuma.uma.es/manycore>.

2 Architecture

The MC64 web platform has been developed in JavaServer Pages (JSP) and deployed in Tomcat 5.5 on a low-power PC running Windows XP SP3 and MySQL 5.0.67. The MC64 uses a Front Controller pattern design, allowing the user to select the required algorithm and launch a customized execution, by introducing the chosen parameters.

The Figure 1 shows the workflow of an MC64 request. A Front Controller retrieves the parameters (1.a) and calls to a Launch Manager web service which validates them (2.a). This web service can be invoked directly as well by any third party program that fulfills its Web Services Description Language (WSDL) (1.b, 2.b). If the sequences to work with are specified by their accession numbers, then the Launcher Manager retrieves them from the National Center for Biotechnology Information (NCBI) nucleotide database (3). The Launcher Manager estimates the amount of memory required for operations and stores the job into an intermediate

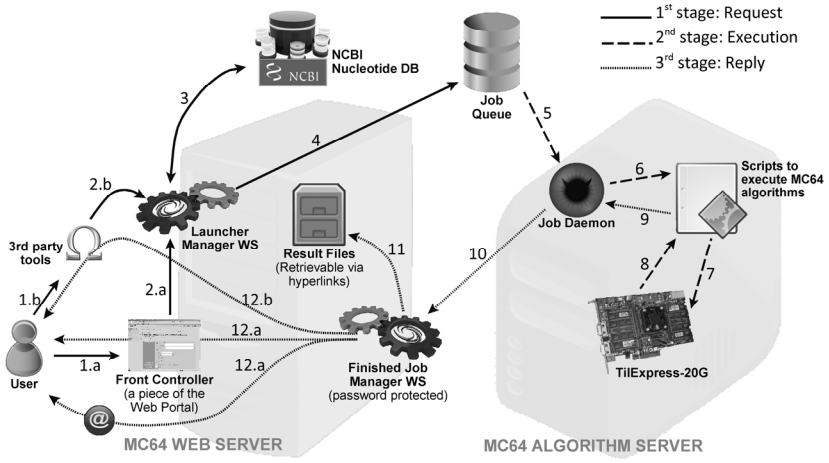

Figure 1. Architecture of the MC64 web platform. The data is processed in three main stages

Job Queue database (4) in order to be executed by the MC64 Algorithm Server. If an out-of-memory is foreseen or parameters are invalid, the job is rejected and the web service returns an error message to the user; otherwise, a control number is assigned to the job and the user receives an OK message via the Front Controller. At the same time, a Job Daemon pools the Job Queue (5) in the MC64 Algorithm Server, and invokes an associated script to each job type (6). These scripts communicate with a TilExpress-20G card to upload the required files, run the many-core algorithm (7) and download the resulting files (8). When a job is finished (9), the daemon calls a Finished Job Manager web service (10), which generates a results web page (11) in the Web Server (12.a,b). If an email address was specified, this manager sends a message to the user with the general performance information and the results as attachment files (12.a).

3 Supported algorithms and performance

The MC64 allows to execute three main bioinformatics algorithms for sequences alignments: Needleman-Wunsch global pairwise alignment (MC64-NW), Smith-Waterman local pairwise alignment (MC64-NW/SW) and ClustalW multiple alignment (MC64-ClustalW). All of them are based on a parallel FastLSA [10] implementation running on TilExpress-20G card (64 cores at 866 GHz) with 8 GB of RAM. Four of out of the 64 are reserved for host-PCIe communication purposes, and the rest are dedicated to run the FastLSA algorithm. Unfortunately, the lack of floating point support in the Tile64 tangles the implementation of other algorithms, including the widely used heuristic BLAST.

3.1 Pairwise alignments

Both the local and the global pairwise alignments are based on a parallel implementation of the FastLSA algorithm which has been developed from scratch. To allow very long sequence alignments, improving the time and memory usage, the FastLSA does not store the entire Dynamic Programming Matrix (DPM) in memory, but only the rows/columns at positions 0, k, $2k$, $3k$, etc., where the k-value can be adjusted to use all the available memory. This strategy takes advantage of the main memory resources to achieve better performance than other linear-space alignment algorithms, like Hirschberg [11].

The Figure 2 shows the execution times and memory requirements for pairwise local alignments ranging from 0.5 kb to 1,000 kb. To give the user an idea of the power of the many-core technologies, the MC64 web server provides an applet which simulates in real-time the alignment of two sequences of 400 kb (only the forward stage is simulated). Each dark pixel of the right 267x267 square (Figure 3) represents the calculus of 2,250,000 cells of the DPM.

The alignment algorithms require the sequences to align, which may be specified typing online text in the Fast Alignment Sequence Tools (FAST)-All (FASTA) format, by means of two NCBI GenInfo Identifier (GI) numbers, or uploading local FASTA files. Besides, the user can customize the open/extend gap cost values, select the scoring matrix to use (among many standard DNA and peptide matrices) and the cost of the match/replace operation. In fact, the MC64-NW/SW constitutes an extension of the MC64-NW, allowing both the local and the global alignments, as well as to obtain the alignment in FASTA format or only the similarity/homology score. However, the MC64-NW is, indeed, the very first Tile64 bioinformatics algorithm ever developed.

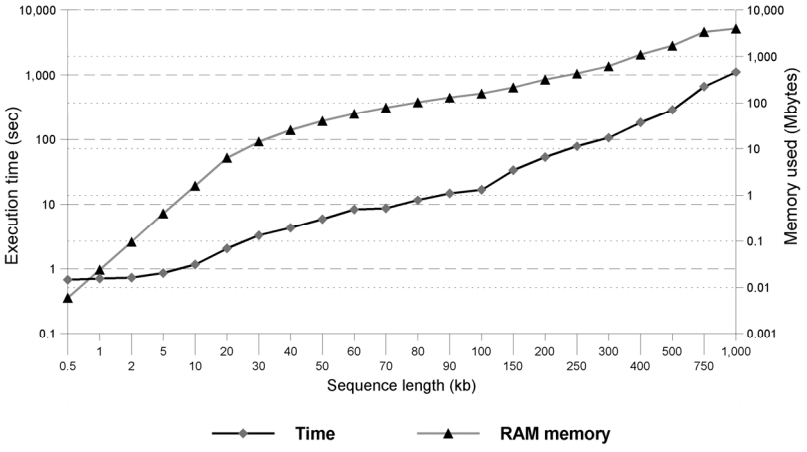

Figure 2. Smith-Waterman alignment performance. The plot shows the execution time and the required memory versus the sequence length with optimal k value in each case.

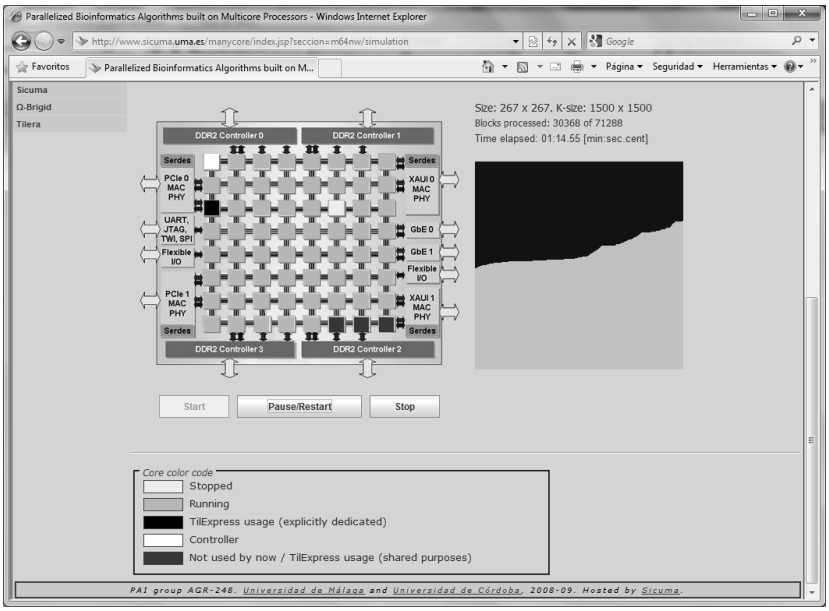

Figure 3. Parallel alignment simulation. Web page with an applet simulating in real time the pairwise alignment between two sequences of 400 kb length. The core's grid shows four reserved cores (dark grey) and 60 dedicated to align: one is the controller (white) and 59 are workers (grey).

3.2 Multiple alignments: ClustalW

The multiple sequences alignment is one of the most bioinformatics algorithms used by life-science researchers. It allows comparing two or more sequences at once, in order to determine identities and differences. The result of multiple alignment algorithms can be used to generate a phylogenetic tree of the aligned sequences (dendrogram). That may be particularly useful to sort the sequences (and hence the individuals, varieties, cultivars, strains, breeds, species, etc) taking into account the evolutionary (or domestication) point of view.

The ClustalW is a multiple alignment algorithm divided in three main stages. At the time ClustalW is invoked with n sequences, its first stage calculates a pairwise alignment between any unordered pair of sequences s_i and s_j with $1 \leq i \neq j \leq n$. Though the former implementations of ClustalW relied on heuristics to do this calculus, the latest ones use optimal alignments. Thus, if the input sequences are large (>200 kb), the time consumed by this stage may be prohibitive, even with a small number of sequences. For instance, the multiple alignment of 10 large sequences requires 45 pairwise alignments, which is costly in time.

To overcome this problem, we have developed a preliminary version of the ClustalW algorithm, where this stage has been parallelized by using the MC64-NW/SW in the first stage: MC64-ClustalW. Although our ultimate goal is a full parallelization of the ClustalW (including second and third stages, guide tree calculus and progressive alignment, respectively), the performance obtained with the simple substitution of this first stage is considerable. Thus, the Table 1 shows the time consumed by the ClustalW-MPI [12] when is executed in a multi-core system (Xeon Quad-core at 2.0 GHz) with 10 sequences of different lengths, as compared with the MC64-ClustalW. The performance gain achieved by MC64-ClustalW increases as the sequences are longer.

Table 1. Time performance comparison between many- and multi-core implementations.

Implementation	Sequence length (kb)			
	200	**150**	**100**	**50**
MC64-ClustalW	2,860 s.	1,651 s.	898 s.	311 s.
ClustalW-MPI	9,715 s.	3,572 s.	1,503 s.	427 s.
Gain (%)	339%	216%	167%	137%

4 Third party tools integration: Omega-Brigid

The Omega-Brigid [13] is a framework designed to unify and integrate different bioinformatics resources from both local resources and the Internet. From a user point of view, the Omega-Brigid tool allows managing workflows; i.e., diagrams with a chain of cells connected by arrows, which represent the data flow. The user can create such diagrams and execute the corresponding workflows, so that the data is retrieved from the input cells, redirected to the processing cells and the output is displayed or stored by viewer or storage cells, respectively. Most importantly, the Omega-Brigid can be used for any life-science researcher, without any computing code programming required. Each cell corresponds to a local or remote resource operation. For instance, reading a local FASTA file, extracting sequences from their NCBI accession numbers, executing a local or remote Basic Local Alignment Search Tool (BLAST) [14], generating the reverse complement of a sequence, executing ClustalW, displaying alignments in viewers like Jalview [15], etc. The Omega-Brigid is used to illustrate how a third-party tool can benefit of the open architecture of the MC64 web platform.

The Omega-Brigid architecture allows including new functionalities straightforwardly by means of Java plugins. Therefore, to use the MC64 platform resources, we have developed a plugin to request the execution of any available MC64 algorithm to the Launcher Manager web service. Each algorithm is represented by a different processing cell. However, the output of such cells is not the final result of the algorithm execution, but a Job number (or an error message)

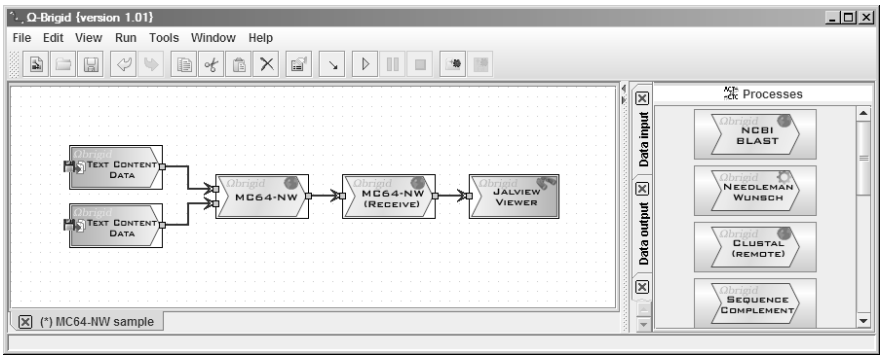

Figure 4. Omega-Brigid workflow. This workflow example takes two sequences in FASTA text format, calls the MC64-NW and, finally, displays the alignment in the Jalview viewer.

that must be used by a next "receiving" cell to retrieve the actual result, when available. This second cell polls the user email account for the resulting message or polls the MC64 Web Server for the results page whose number corresponds to the Job code. When the second cell retrieves the result, this is redirected to the next cell, e.g. an appropriate viewer, so the workflow continues. The Figure 4 shows a very simple workflow to execute the MC64-NW.

5 Conclusions and further work

The many-core technologies are evolving in two main different directions: i) hundreds and even thousands of cores with minimal resources (GPGPU); and ii) tens of cores, each of them being capable of executing a whole operating system (SoC). The first ones are readily available because they are widely used in the video game market. In this paper we have focused on the latter ones, which constitute an emerging technology with great potential for bioinformatics. The MC64 web platform allows any life-science researcher to execute basic bioinformatics algorithms in the Tile64 architecture and to test the relative performance when the same algorithms are executed in a usual x86 multi-core architecture.

Thus, we have demonstrated the potential of the many-core technology for the next-generation bioinformatics. With this platform, we want to make this technology available to researchers. Our current developments with the Tile64 includes: i) optimal pairwise alignments of sequences with tens of megabases (Mb), ii) ClustalW improvements and iii) high performance pairwise alignments between a query sequence and a target database. These services will be publicly available by user request, in order to schedule long executions and optimize resources. Furthermore, the MC64 Web platform is constantly updated (user registration and job management is currently under development).

Acknowledgments. We are grateful to Tilera for providing hardware and software tools <http://www.tilera.com>. This work was supported by "Ministerio de Ciencia e Innovación" [BIO2009-07443-E and AGL2010-17316]; "Consejería de Agricultura y Pesca" of "Junta de Andalucía" [041/C/2007 & 75/C/2009]; "Grupo PAI" [AGR-248]; and "Universidad de Córdoba" ["Ayuda a Grupos"], Spain.

References

1. Howard J, Dighe S, Hoskote Y, Vangal S, Finan D, Ruhl G, Jenkins D, Wilson H, Borkar N, Schrom G et al: A 48-Core IA-32 Message-Passing Processor with DVFS in 45nm CMOS. In: International Solid-State Circuits Conference, 2010 ISSCC 2010 Digest of Technical Papers IEEE International. 2010: 19-21.
2. Skaugen K: Petascale to Exascale. In: International Supercomputing Conference. Hamburg, Germany; 2010.
3. Bell S, Edwards B, Amann J, Conlin R, Joyce K, Leung V, MacKay J, Reif M, Bao L, Brown J et al: TILE64 - Processor: A 64-Core SoC with Mesh Interconnect. In: Solid-State Circuits Conference, 2008 ISSCC 2008 Digest of Technical Papers IEEE International. 2008: 88-598.
4. Mike Tate RJ, Behrooz Abdi: NetLogic Microsystems, Inc. Q3 2010 Earnings Conference Call Transcript. In: Thomson StreetEvents. 2010.
5. Shah M, Barreh J, Brooks J, Golla R, Grohoski G, Gura N, Hetherington R, Jordan P, Luttrell M, Olson C et al: UltraSPARC T2: A highly-treaded, power-efficient, SPARC SOC Asian Solid-State Circuits Conference (ASSCC'07) 2007:22-25.
6. Needleman SB, Wunsch CD: A general method applicable to the search for similarities in the amino acid sequence of two proteins. J Mol Biol 1970, 48(3):443-453.
7. Smith TF, Waterman MS: Identification of common molecular subsequences. J Mol Biol 1981, 147(1):195-197.
8. Gálvez S, Díaz D, Hernández P, Esteban FJ, Caballero JA, Dorado G: Next-Generation Bioinformatics: Using Many-Core Processor Architecture to Develop a Web Service for Sequence Alignment. Bioinformatics 2010, 26(5):683-686.
9. Thompson JD, Higgins DG, Gibson TJ: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res 1994, 22(22):4673-4680.
10. Driga A, Lu P, Schaeffer J, Szafron D, Charter K, Parsons I: FastLSA: A Fast, Linear-Space, Parallel and Sequential Algorithm for Sequence Alignment. Algorithmica 2006, 45(3):337-375.
11. Hirschberg DS: A linear space algorithm for computing maximal common subsequences. Commun ACM 1975, 18(6):341-343.
12. Li K-B: ClustalW-MPI: ClustalW analysis using distributed and parallel computing. Bioinformatics 2003, 19(12):1585-1586.
13. Díaz D, Gálvez S, Falgueras J, Caballero JA, Hernández P, Claros G, Dorado G: Intuitive Bioinformatics for Genomics Applications: Omega-Brigid Workflow Framework. In: Proceedings of the 10th International Work-Conference on Artificial Neural Networks: Part II: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living. Salamanca, Spain: Springer-Verlag; 2009: 1084-1091.
14. Altschul S, Gish W, Miller W, Myers E-M, Lipman D: Basic local alignment search tool. J Mol Biol 1990, 215:403-410.
15. Waterhouse AM, Procter JB, Martin DM, Clamp M, Barton GJ: Jalview Version 2--a multiple sequence alignment editor and analysis workbench. Bioinformatics 2009, 25(9):1189-1191.