Instructions for Authors of SBC Conferences Papers and Abstracts

Luciana P. Nedel[1], **Rafael H. Bordini**[2], **Flávio Rech Wagner**[1], **Jomi F. Hübner**[3]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

[2]Department of Computer Science – University of Durham
Durham, U.K.

[3]Departamento de Sistemas e Computação
Universidade Regional de Blumenal (FURB) – Blumenau, SC – Brazil

{nedel,flavio}@inf.ufrgs.br, R.Bordini@durham.ac.uk, jomi@inf.furb.br

Abstract. *This meta-paper describes the style to be used in articles and short papers for SBC conferences. For papers in English, you should add just an abstract while for the papers in Portuguese, we also ask for an abstract in Portuguese ("resumo"). In both cases, abstracts should not have more than 10 lines and must be in the first page of the paper.*

1. Introduction

Parallel computing is hard to define, but intuitively, it is a computation method that allows data to be distributed and processed simultaneously. In Flynn's taxonomy [Pacheco 2011], there are two types of parallel computer archictectures:

1. Single Instruction, Multiple Data (SIMD); a processor that allows a chunk of data to be loaded and a single instruction to be used to process it. Intel's SSE and Graphics Processing Units (GPU) are examples it.
2. Multiple Instruction, Multiple Data (MIMD); a system consisting of multiple independent processing units, executing asynchronously. Here are located all multicore Shared-memory systems and Distributed-memory systems.

GPU was originally designed to renderize graphics in real time, however, researchers realized that GPU could be used for general purpose applications. OpenGL and DirectX were designed to access GPUs resources, both are specialized to produce graphics. These libraries were used by engineers and scientists in their specific problems, as a result, they had to convert their problem into a graphical domain.

NVIDIA noticed a new demand for their products and created an API called CUDA with the goal enable the use of GPUs in general purpose situation.

CUDA consists of kernels, which are functions that run in a GPU in a parallel fashion. A kernel is organized into a set of blocks. A block is a set of threads that cooperate between themselves [Patterson and Hennessy 2007].

A kernel accesses data from GPU's memory. Its memory is divided between global memory, a memory that all threads can access; a local memory, a memory that is private to a thread; and shared memory, a low-latency memory that is shared between all threads in the same block [Patterson and Hennessy 2007].

The Bondary Elements Method (BEM) is a computational method of solving linear partial differentials equations that have been formulated as integral equations. This is used in many areas of engineering, but this article presents a Graphical Unit Processor (GPU) accelerated implementation of BEM with the purpose of analysing waves propagation on the ground and it's effects on nearby structures.

Given a sequential implementation of such method by [Carrion 2002], the main objectives of this project includes the creation of automated tests to check if the modified program results are numerically compatible with the original version; somewhat modernize the legacy code, which was written in Fortran 77; optimize the code, removing repeated unnecessary calculations and managing a better usage of the Central Processing Unit (CPU) resources; identify the most time-consuming subroutines and paralellize them.

This project was supported by CAPES and **BLABLABLABLABLABLA** (não sei muito bem o que colocar aqui)

2. Norms

In order to check if the final result obtained by the parallel program is numerically compatible with the original, the concept of vector and matrix norms are necessary. Let $x \in \mathbb{C}^n$ and $A \in \mathbb{C}^{m \times n}$. [Watkins 2004] defines a vector ∞-norm, matrix ∞-norm and matrix 1-norm as:

$$\|x\|_\infty = \max_{1 \le k \le n} |x_k| \qquad \|A\|_\infty = \max_{1 \le i \le m} \sum_{j=1}^{n} |a_{ij}| \quad \|A\|_1 = \max_{1 \le j \le n} \sum_{i=1}^{m} |a_{ij}| \tag{1}$$

All vector norms have the propierty that $\|x\| = 0$ if and only if $x = 0$, and the same result asserts for matrix norms. Let f and g be two numerical algorithms that solves the same problem, but in a different fashion. Let now y_f be the result computed by f and y_g be the result computed by g. The *error* between those two values can be measured computing $\|y_f - y_g\|$.

3. Parallelization Technique

A parallel implementation of BEM began by analyzing and modifying a sequential code provided by Carrion. Gprof, a profiling tool by GNU, showed that the most time-consuming routine was Nonsingd, a subroutine designed to solve small parts of the non-singular dynamic problem. Since most calls to Nonsingd were from Ghmatecd, most of the parallelization effort was focused on that last routine.

3.1. Parallelization of Ghmatecd

Ghmatecd works in the following way: It assembles two complex matrices H and G by computing smaller 3×3 matrices returned by Nonsingd and Sigmaec. Sigmaec is very similar to Nonsingd, but it is designed to solve the singular part of the dynamic problem. Let n be the number of **ELEMENTOS DE MALHA** and m **ELEMENTOS DE CONTORNO** The following pseudocode illustrates how Ghmatecd works:

Algorithm 1 Creates $H, G \in \mathbb{C}^{(3m)\times(3n)}$

```
1: procedure GHMATECD
2:     for $j := 1, n$ do
3:         for $i := 1, m$ do
4:             $ii := 3(i-1)+1; jj := 3(j-1)+1$
5:             if $ii == jj$ then
6:                 $Gelement, Helement \leftarrow$ Sigmaec$(i, j)$ ▷ two $3 \times 3$ complex matrices
7:             else
8:                 $Gelement, Helement \leftarrow$ Nonsingd$(i, j)$
9:             end if
10:            $G[ii : ii+2][jj : jj+2] \leftarrow Gelement$
11:            $H[ii : ii+2][jj : jj+2] \leftarrow Helement$
12:        end for
13:    end for
14: end procedure
```

In the algorithm above, Nonsingd and Sigmaec parameters are not displayed for readability. Notice that there is no interdependency between iteractions, so the two nested loop in the algorithm can be computed in parallel, hence, both matrices H and G can be built in a parallel fashion. If the number of processors is small, then computing that nested loop in parallel is enough because even for small instances of the problem, $n \times m$ will be bigger than the number of processors. By the other hand, GPUs have a huge parallel capability, and the above strategy alone would generate a waste of computational resources. A careful analysis of both Nonsingd and Sigmaec codes concluded that a certain degree of parallelism can be explored in those procedures in order to take advantage of GPUs. Both procedures have a two nested loop that generates various 3x3 matrices. Before returning, those procedures do a reduction of these matrices. It should be highlighted that Sigmaec and Nonsingd are almost identical, so both can be merged into a single routine with a special logic to check if it needs to solve the singular or the nonsingular problem. Let g be the number of Gauss quadrature points. If this new procedure is unrolled in Ghmatecd, then there would be:

The algorithm2 can be implemented in CUDA kernel in the following way: Inside a block, create $g \times g$ threads to compute in parallel the two nested loop in lines 6 to 7, allocating *Hbuffer* and *Gbuffer* in shared memory. Since these buffers contain matrices of size 3×3, 9 of these $g \times g$ threads can be used to sum all matrices. Notice that $g \geq 3$ is necessary for that condition, and that g is also upper-bounded by the amount of shared memory available in the GPU. Launching $m \times n$ blocks to cover the two nested loops in lines 2 to 3 will generate the entire H and G.

3.2. Parallelization of Ghmatece

Ghmatece is a routine designed to create two real matrices H and G associated with the static problem, and it is very similar to Ghmatecd. That routine can be implemented in CUDA in almost the same way as Ghmatecd, but caution is needed because the singular case has a lot of ifs that can cause branch divergence, thus affecting performance in a negative way. Since the singular case is dimensionally smaller than the non-singular case,

Algorithm 2 Creates $H, G \in \mathbb{C}^{(3m)\times(3n)}$

```
 1: procedure GHMATECD
 2:     for $j := 1, n$ do
 3:         for $i := 1, m$ do
 4:             $ii := 3(i-1)+1; jj := 3(j-1)+1$
 5:             Allocate *Hbuffer* and *Gbuffer*, buffer of matrices $3 \times 3$ of size $g \times g$
 6:             for $y := 1, g$ do
 7:                 for $x := 1, g$ do
 8:                     *params* ← ComputeParameters$(i, j, x, y)$
 9:                     *Hbuffer*$(x, y)$ ← GenerateMatrixH$(x, y, params)$
10:                     *Gbuffer*$(x, y)$ ← GenerateMatrixG$(x, y, params)$
11:                     if $ii == jj$ then
12:                         SpecialLogicForSingularCase(*Hbuffer*$(x, y)$, *Gbuffer*$(x, y)$)
13:                     end if
14:                 end for
15:             end for
16:             $Gelement$ ← SumAllMatricesInBuffer(*Gbuffer*)
17:             $Helement$ ← SumAllMatricesInBuffer(*Hbuffer*)
18:             $G[ii : ii+2][jj : jj+2] \leftarrow Gelement$
19:             $H[ii : ii+2][jj : jj+2] \leftarrow Helement$
20:         end for
21:     end for
22: end procedure
```

it can be computed in the CPU and the result merged with the non-singular case calculated in the GPU.

4. General Information

All full papers and posters (short papers) submitted to some SBC conference, including any supporting documents, should be written in English or in Portuguese. The format paper should be A4 with single column, 3.5 cm for upper margin, 2.5 cm for bottom margin and 3.0 cm for lateral margins, without headers or footers. The main font must be Times, 12 point nominal size, with 6 points of space before each paragraph. Page numbers must be suppressed.

Full papers must respect the page limits defined by the conference. Conferences that publish just abstracts ask for **one**-page texts.

5. First Page

The first page must display the paper title, the name and address of the authors, the abstract in English and "resumo" in Portuguese ("resumos" are required only for papers written in Portuguese). The title must be centered over the whole page, in 16 point boldface font and with 12 points of space before itself. Author names must be centered in 12 point font, bold, all of them disposed in the same line, separated by commas and with 12 points of space after the title. Addresses must be centered in 12 point font, also with 12 points of space after the authors' names. E-mail addresses should be written using font Courier New, 10 point nominal size, with 6 points of space before and 6 points of space after.

The abstract and "resumo" (if is the case) must be in 12 point Times font, indented 0.8cm on both sides. The word **Abstract** and **Resumo**, should be written in boldface and must precede the text.

6. CD-ROMs and Printed Proceedings

In some conferences, the papers are published on CD-ROM while only the abstract is published in the printed Proceedings. In this case, authors are invited to prepare two final versions of the paper. One, complete, to be published on the CD and the other, containing only the first page, with abstract and "resumo" (for papers in Portuguese).

7. Sections and Paragraphs

Section titles must be in boldface, 13pt, flush left. There should be an extra 12 pt of space before each title. Section numbering is optional. The first paragraph of each section should not be indented, while the first lines of subsequent paragraphs should be indented by 1.27 cm.

7.1. Subsections

The subsection titles must be in boldface, 12pt, flush left.

8. Figures and Captions

Figure and table captions should be centered if less than one line (Figure 1), otherwise justified and indented by 0.8cm on both margins, as shown in Figure 2. The caption font must be Helvetica, 10 point, boldface, with 6 points of space before and after each caption.

Figure 1. A typical figure

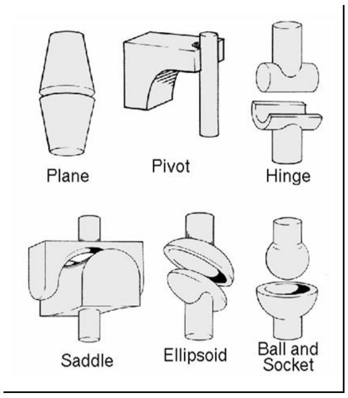


Figure 2. This figure is an example of a figure caption taking more than one line and justified considering margins mentioned in Section 8.

In tables, try to avoid the use of colored or shaded backgrounds, and avoid thick, doubled, or unnecessary framing lines. When reporting empirical data, do not use more decimal digits than warranted by their precision and reproducibility. Table caption must be placed before the table (see Table 1) and the font used must also be Helvetica, 10 point, boldface, with 6 points of space before and after each caption.

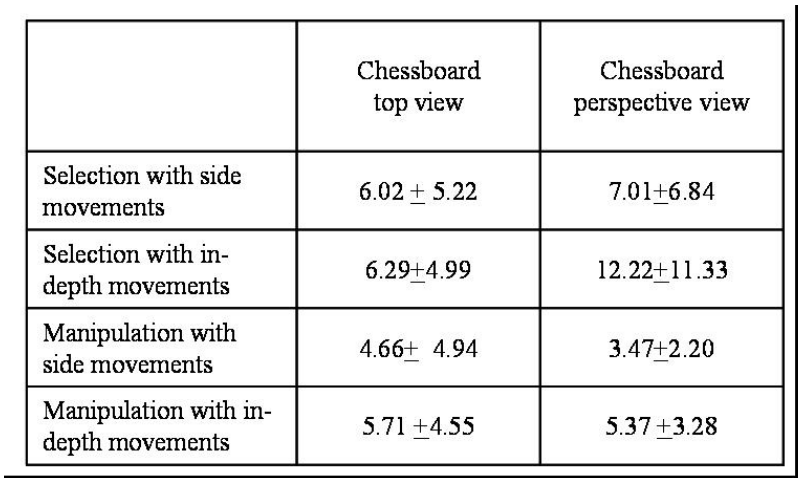
9. Images

All images and illustrations should be in black-and-white, or gray tones, excepting for the papers that will be electronically available (on CD-ROMs, internet, etc.). The image resolution on paper should be about 600 dpi for black-and-white images, and 150-300 dpi for grayscale images. Do not include images with excessive resolution, as they may take hours to print, without any visible difference in the result.

10. References

Bibliographic references must be unambiguous and uniform. We recommend giving the author names references in brackets, e.g. [Knuth 1984], [Boulic and Renault 1991], and [Smith and Jones 1999].

Table 1. Variables to be considered on the evaluation of interaction techniques



	Chessboard top view	Chessboard perspective view
Selection with side movements	6.02 ± 5.22	7.01±6.84
Selection with in-depth movements	6.29±4.99	12.22±11.33
Manipulation with side movements	4.66± 4.94	3.47±2.20
Manipulation with in-depth movements	5.71 ±4.55	5.37 ±3.28

The references must be listed using 12 point font size, with 6 points of space before each reference. The first line of each reference should not be indented, while the subsequent should be indented by 0.5 cm.

References

Boulic, R. and Renault, O. (1991). 3d hierarchies for animation. In Magnenat-Thalmann, N. and Thalmann, D., editors, *New Trends in Animation and Visualization*. John Wiley & Sons ltd.

Carrion, R. (2002). *Uma Implementação do Método dos Elementos de Contorno para problemas Viscoelastodinâmicos Estacionários Tridimensionais em Domínios Abertos e Fechados*. PhD thesis, Universidade Estadual de Campinas.

Knuth, D. E. (1984). *The TEX Book*. Addison-Wesley, 15th edition.

Pacheco, P. (2011). *An introduction to parallel programming*. Elsevier.

Patterson, D. A. and Hennessy, J. L. (2007). Computer organization and design. *Morgan Kaufmann*.

Smith, A. and Jones, B. (1999). On the complexity of computing. In Smith-Jones, A. B., editor, *Advances in Computer Science*, pages 555–566. Publishing Press.

Watkins, D. S. (2004). *Fundamentals of matrix computations*, volume 64. John Wiley & Sons.