Modeling and Simulation of the Blood Coagulation Cascade

Matteo Bellini[†], Daniela Besozzi[†], Paolo Cazzaniga[‡], Giancarlo Mauri[§] and Marco S. Nobile[§]

[§]Università di Milano-Bicocca – Dipartimento di Informatica, Sistemistica e Comunicazione, Milano, Italy
[†]Università di Milano – Dipartimento di Informatica, Milano, Italy
[‡]Università di Bergamo – Dipartimento di Scienze Umane e Sociali, Bergamo, Italy

Abstract

In the last years, the use of Graphical Processing Units (GPUs) has witnessed ever growing applications for different computational analyses in the field of Life Sciences. In this work we present a CUDA-powered computational tool for the simulation of biological models defined as systems of mixed ordinary differential equations, based on both mass-action kinetics and Hill functions. The tool, named coagSODA, was purposely developed and applied for the analysis of a large model of the blood coagulation cascade. We present the biological results of a parameter sweep analysis of this model, and show that GPU can boost the computational performances up to 177× speedup.

Keywords: *GPU Computing, modeling, simulation*

1. Introduction

Since the introduction of general-purpose Graphics Processing Units (GPUs) and of CUDA, the Nvidia's GPU programming language, the adoption of graphics engines experienced a great boost in scientific applications related to Bioinformatics, Systems Biology and Computational Biology (see [1], [2], [3] and references therein). Indeed, the use of GPUs for *in silico* investigations of complex biological systems – where Central Processing Units (CPUs) traditionally represented the standard workhorses – is motivated by the need of performing large numbers of simulations to analyze the systems' behavior in different conditions, which necessitate a computing power that usually overtakes the capability of standard desktop computers, therefore requiring high-performance computing solutions.

Despite the considerable advantages in terms of computational speedup, scientific applications on GPUs risk to remain a niche for few specialists, since GPU-based programming substantially differs from CPU-based programming and, therefore, GPU computing requires the development and the implementation of *ad hoc* algorithms. To avoid such limitations, several packages and software tools were released (see, e.g., [3], [4], [5]), so that also users with no knowledge of GPUs hardware and programming can access the computing power of graphics engines.

In this work we present coagSODA, a simulator of biological systems based on the GPU-powered tool cupSODA [6], which allows to run deterministic simulations of a given

mass-action based system of biochemical reactions, using the LSODA algorithm [7]. coagSODA was specifically designed for the analysis of the *blood coagulation cascade* (BCC), and it allows to efficiently execute a large number of parallel deterministic simulations at a considerable reduced computational cost with respect to CPUs.

Blood is an essential component in human life, whose primary functions are to feed cells by delivering a multitude of nutrients, and to carry away the cellular wastes, such as carbon dioxide. Thanks to its key role in making diagnosis of numerous diseases, blood is the subject of an intense scientific research [8]. All blood components are kept within appropriate concentration ranges by means of fine-tuned regulatory mechanisms, ruled by several feedback controls; the constancy of blood composition is maintained thanks to the circulation through an intricate network of vessels. In particular, humans have evolved a complex hemostatic system that, under physiologic conditions, is able to maintain blood in a fluid state; however, in response to any vascular injury, this system is able to rapidly react and seal the defects in the vessels wall in order to stop the blood leakage [9]. Indeed, the circulatory system is self-sealing, otherwise a continuous blood flow from even the smallest wound would become a threaten for the individual's life.

These regulation processes are governed by means of the BCC, a complex network of cellular reactions which, under physiological conditions *in vivo*, are inhibited by the presence of intact endothelium [10]. In order to allow blood coagulation, in humans there exist 13 blood clotting proteins, called *coagulation factors*, which are usually designated by Roman numerals I through XIII. Following a vascular injury, platelets become active and the tissue factor (TF, or factor III) is exposed in the subendothelial tissue, starting the BCC. The ultimate goal of BCC is to convert prothrombin (factor II) into thrombin (factor IIa - that is, the activated factor II), the enzyme that catalyzes the formation of a clot. Traditionally, the BCC is divided into the *extrinsic* and *intrinsic pathways*, both of which lead to the activation of factor X [11]. The last part of the cascade, downstream of this factor, is called the *common pathway* and leads to the formation of fibrin monomers, whose polymers finally constitute the backbone of the clot.

Excluding thrombin, all the enzymes involved in blood clotting are characterized by a low activity, which increases upon binding to a specific protein cofactor (e.g., factors V and VIII) or to appropriate phospholipid surfaces (e.g., the plasma membranes of active platelets) [11]. In BCC pathways, there exist also inhibitory factors, which limit the activity of the various active proteases and, therefore, allow to regulate the whole reactions cascade. When the hemostatic system is unregulated, thrombosis (i.e., the formation of a blood clot obstructing the blood flow in vessels) may occur due to an impairment in the inhibitory pathway, or because the functioning of the natural anticoagulant processes is overwhelmed by the strength of the hemostatic stimulus [9].

In order to investigate the individuals' variations in blood coagulation components, and the corresponding response to perturbed conditions, we consider here a Systems Biology perspective to analyze the alterations (prolongation or reduction) of the time necessary to form the clot *in vivo* (i.e., the *clotting time*). To this aim, we exploit a mathematical model of the BCC [12], built upon a previous model [13], that describes the intrinsic, extrinsic and common pathways and, more importantly, it accounts for the platelet activation, as well as the

presence of several inhibitors (e.g., the tissue factor pathway inhibitor (TFPI), antithrombin III and C1-inhibitor). Numerous mathematical models of blood coagulation have been developed in the last years, as they represent a useful tool for systematic studies of the intricate structure of the coagulation cascade, and allow to obtain a suitable reconstruction of empirical observations (see, e.g., [14], [15], [16]).

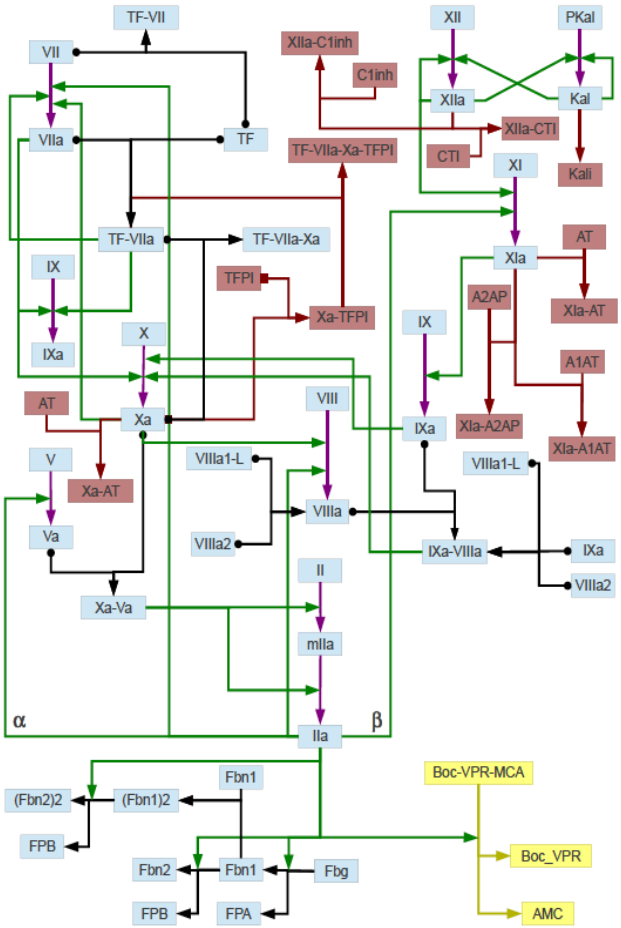

Fig. 1. Reaction network of the blood coagulation cascade model considered in this work. *Legenda*. Light blue boxes: coagulation factors; brown boxes: inhibitors and related complexes; yellow boxes: fluorogenic detectors. Black arrows: complex formation; green arrows: catalytical activation; violet arrows: activation; red arrows: inhibition.

The BCC model we consider is a slightly reduced version of the deterministic model given in [12], where we exclude a small set of reactions downstream of the clot formation – that is, the interactions between the fibrin polymers, thrombin and antithrombin III – since they do not have effect on the clotting time. A graphical sketch of the BCC model, which overall consists in 101 reactions among 81 molecular species, is given in Figure 1. The system of ordinary differential equations (ODEs), needed to carry out the simulations and the parameter sweep analysis presented in what follows is based on mass-action kinetics, except for 14 reactions that are formalized by means of Hill functions. The rationale behind the use of Hill functions relies on the need of modeling the physiological levels of thrombin concentration as a function of platelet activation, as described in [12].

coagSODA, the GPU-accelerated simulator that we present and exploit in this work for the analysis of the BCC model, is a user-friendly and efficient tool that circumvents the need of manually defining the system of ODEs that describe the blood coagulation. More precisely, coagSODA is able to automatically derive the system of (mass-action and Hill functions based) ODEs – and then to perform their numerical integration – starting from the given set of 101 biochemical reactions, which fully describe the molecular interactions between all the species involved in the BCC.

The paper is structured as follows. In Section 2 we describe the GPUs and CUDA architecture, the simulation method at the basis of cupSODA tool, and then introduce the coagSODA simulator. In Section 3 we present the results obtained from the parameter sweep analysis of the BCC model, as well as a comparison of the performance of coagSODA with respect to a CPU-based implementation of LSODA. Finally, in Section 4 we conclude the paper with a discussion of the presented work.

2. GPU-powered Simulation of Biological Systems

2.1. CUDA Architecture and cupSODA Simulator

Introduced by Nvidia in 2006, Compute Unified Device Architecture (CUDA) is a parallel computing platform and programming model that provides programmers with a framework to exploit GPUs in general-purpose computational tasks (GPGPU computing). GPGPU computing is a low-cost and energy-wise alternative to the traditional high-performance computing infrastructures (e.g., clusters of machines), which gives access to the tera-scale computing on common workstations of mid-range price. However, due to the innovative architecture and the intrinsic limitations of GPUs, a direct porting of sequential code on the GPU is most of the times unfeasible, therefore making it challenging to fully exploit its computational power and massive parallelism [17]. CUDA combines the Single Instruction Multiple Data (SIMD) architecture with a flexible multi-threading, in which the conditional divergence between threads is automatically handled. Under CUDA's naming conventions, the programmer implements the *kernel*, that is, a C/C++ function, which is loaded from the host (the CPU) to the devices (one or more GPUs), and replicated in many copies named *threads*. Threads can be organized in three-dimensional structures named *blocks* which, in turn, are contained in three-dimensional *grids* (a schematic description is given in Figure 2, left side). Whenever the host runs a kernel, the GPU creates the corresponding grid and automatically schedules each block on one free streaming multi-processor available on the GPU, allowing a transparent scaling of performances on different devices. Threads within a block are executed in groups of 32 threads named *warps*.

The GPU is equipped with different kinds of memory. In this work, we exploit the *global memory* (accessible from all threads), the *shared memory* (accessible from threads of the same block), the *local memory* (registers and arrays, accessible from owner thread), and the *constant memory* (cached and not modifiable). A schematic representation of this memory hierarchy is shown in Figure 2, right side. To achieve the best performances, the shared memory should be exploited as much as possible; however, it is very limited (i.e., 49152 bytes for each multi-processor) and introduces restrictions on the blocks' size. On the other hand, the global memory is very large (thousands of MBs) but suffers of high latencies. A solution to this problem was implemented on the Fermi architecture, where the global memory is equipped with a L1 cache.

The systematic analyses of models of biological systems often consist in the execution of large batches of simulations. One of the standard analyses that can be executed on such kind of models regards an intensive search within the parameters space, which requires large numbers of independent simulations. In order to reduce the computational burden, we previously implemented on the CUDA architecture one of the most efficient numerical integration algorithms for ODEs, LSODA [7]. This GPU-powered tool is called cupSODA [6], it exploits CUDA's massive parallelism to execute different and independent simulations in each thread, thus reducing the computational time required by a standard CPU counterpart of LSODA.

Besides being very efficient for the execution of many independent simulations, cupSODA is also user- friendly. As a matter of fact, LSODA was designed to solve ODEs systems written in the canonical form(i.e., defined as a system of equations of the form $d[X]/d[t] = f([X], t)$, where $[X]$ represents the concentration of the chemical species X), but the user is supposed to specify the system of ODEs by implementing a custom function that is passed to the algorithm. Moreover, in order to speed up the computation when dealing with stiff systems, in LSODA the Jacobian matrix associated to the ODEs system must be implemented as a custom function as well. On the contrary, cupSODA was conceived as a *black-box* simulator, that can be easily used without any programming skills. cupSODA consists in a tool to automatically convert a generic mechanistic reaction-based model of a biological system into the corresponding set of ODEs, complied with the mass-action kinetics [19], and to directly encode the obtained system – along with the corresponding Jacobian matrix – as C arrays.

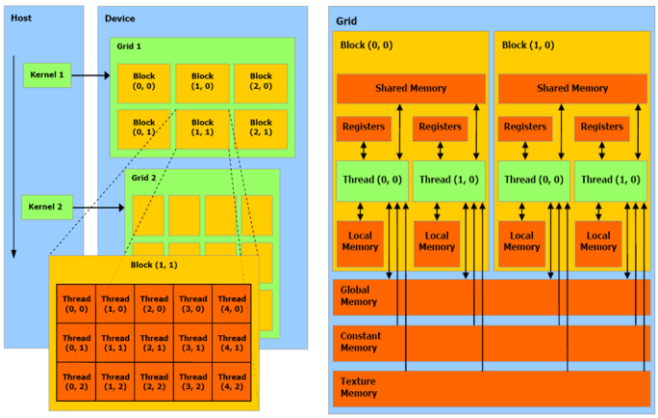

Fig. 2. Schematic description of CUDA's architecture, in terms of threads and memory hierarchy. *Left side*. Thread organization: a single kernel is launched from the host (the CPU) and is executed in multiple threads on the device (the GPU). Threads can be organized in three-dimensional structures named blocks which can be, in turn, organized in three-dimensional grids. The dimensions of blocks and grids are explicitly defined by the programmer. *Right side*. Memory hierarchy: threads can access data from different memories with different scopes. Registers and local memories are private for each thread. Shared memory let threads belonging to the same block communicate, and has low access latency. All threads can access the global memory, which suffers high latencies. Texture and constant memory can be read from any thread and are equipped with a cache as well; in this work we exploit the constant memory. Figures are taken from Nvidia's CUDA programming guide [18].

This fully automatic methodology can be exploited only for the simulation of models whose chemical kinetics is based on the *mass-action law*, that is, the fundamental empirical law that governs biochemical reaction rates which states that, in a dilute solution, the rate of an elementary reaction is proportional to the product of the concentration of its reactants raised to the power of the corresponding stoichiometric coefficient [20]. The BCC model that we investigate, though, includes a set of reactions that do not follow the mass-action kinetics [12]. In particular, the platelet activity is not explicitly modeled, but it is realized in the model by modulating the rate of the dissociation of the complexes formed on a platelet's surface by means of a specific factor ε, which is calculated with a special equation during the integration steps. For this reason, cupSODA was purposely modified in order to consider the *run-time* computation of the specific kinetics of this set of reactions.

2.2. coagSODA: a GPU-powered Simulator for the Blood Coagulation Cascade

The coagSODA simulator is an extension of cupSODA, designed and implemented for the CUDA architecture, and specifically tailored for the simulation of the BCC model given in [12]. coagSODA realizes the runtime calculation of the ε value required to correctly simulate the activity of platelets; the variable ε depends on a Hill function that was fit against experimental data which quantify the platelet activation status (see details in [12]). The value of ε influences the formation of some complexes within the BCC, occurring on the platelets' surface; namely, ε intervenes on their dissociation constants by modifying the activity of reactions of the form

$$A + B \underset{k_2}{\overset{k_1}{\rightleftharpoons}} AB$$

so that the corresponding standard ODEs are changed to yield new equations of the form:

$$\frac{d[AB]}{dt} = k_1 \cdot [A] \cdot [B] - \frac{k_2 \cdot [A] \cdot [B]}{\varepsilon}. \qquad (1)$$

The set of reactions of the BCC model involved in the evaluation of ε is shown in Table I.

Table I. Reactions in the BCC model influenced by the ε value for the simulation of the activity of platelets

IXa-VIIIa → VIIIa + IXa
IXa-VIIIa-X → IXa-VIIa + X
VIIIa → $VIIIa_1$-L + $VIIIa_2$
Xa-Va → Xa + Va
Xa-Va-II → Xa-Va + II
Xa-TFPI → Xa + TFPI
XIIa-XII → XIIa + XII
XIIa-PKal → XIIa + PKal
XII-Kal → XII + Kal
XIIa-XI → XIIa + XI
XIa-IX → XIa + IX
IXa-X → IXa + X
Xa-VIII → Xa + VIII
VIIa-X → VIIa + X

The value of ε in Equation 1 depends on the following Hill function F, which quantifies the state of platelets' activation according to the thrombin concentration (denoted as [*IIa*]), that is, the factor catalyzing the formation of the fibrin clot:

$$F([IIa^*(t)]) = \frac{[IIa^*(t)]^{1.6123}}{[IIa^*(t)]^{1.6123} + (2.4279 \cdot 10^{-9})^{1.6123}}$$

where $[IIa^*(t)] = \max_{t' \in [0,t]}\{[IIa(t')]\}$, for a chosen time interval [0, t] of simulation. The value $[IIa^*(t)]$ represents the maximum transient thrombin concentration, it is needed to simulate the fact that in physiological conditions the thrombin concentration starts decreasing after monotonically rising to a peak (Figure 3). So doing, $[IIa^*(t)]$ never decreases once it reaches its maximum magnitude: $[IIa^*(t)]$ is equivalent to $[IIa(t)]$ until the concentration of factor IIa reaches the peak, while thereafter it remains constant at that value, which is the maximum in the considered time interval [0, t]. Function F allows to simulate the physiological condition whereby platelets remain active also when the thrombin concentration decreases.

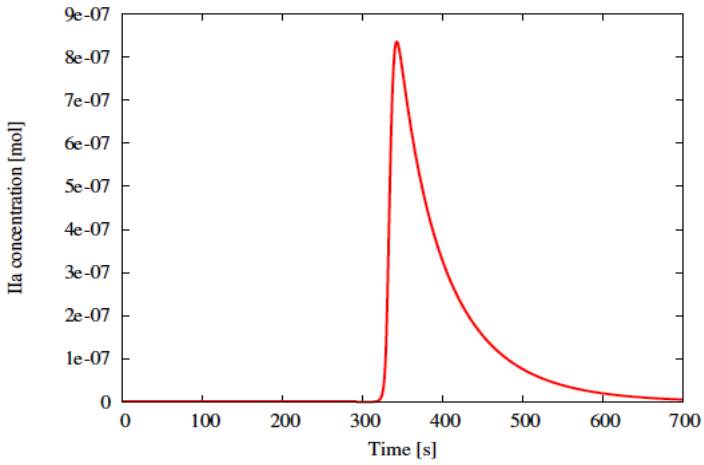

Fig. 3. Dynamics of thrombin in physiological condition

For a given concentration [*IIa*], the maximum platelet activation state ε_{max} is defined as:

$$\varepsilon_{max} = \varepsilon_{max0} + (1 - \varepsilon_{max0}) \cdot F([IIa^*(t)]) \qquad (2)$$

where ε_{max0} defines the basal activation state of the platelet at simulation time $t = 0$. The value ε_{max0} is initially set to 0.01, assuming a basal 1% binding strength of coagulation factors to the resting platelet surface. When the full activation of platelets is reached, ε_{max} is equal to 1 and the complex dissociation constants are minimized [12].

coagSODA calculates at each time instant the platelets activation state ε by solving the following differential equation:

$$\frac{d\varepsilon}{dt} = k(\varepsilon_{max} - \varepsilon)$$

Where the constant k is inversely proportional to the time scale of platelets activation and is set to 0.005. This is consistent with the fact that platelets do not instantly achieve their maximum attainable activation state (ε_{max}), but they reach it on a physiologically relevant timescale.

In dealing with the 14 reactions in the BCC model that are influenced by this Hill function, the value of [*IIa**] must be stored in the GPU because, during each integration step, coagSODA recalculates Equation 2, which exploits the value [*IIa**] to determine a new ε value. Since the CUDA architecture does not offer static variables, the information for each thread has to be memorized in global memory. The accesses to the global memory and the computational costs due to the additional calculations slow down the integration process, with respect to the integration of ODEs performed according to a strictly mass-action based kinetics (as in the cupSODA simulator). Nevertheless, in Section 3.2 we show that our parallel implementation largely outperforms a sequential counterpart.

3. Results

3.1. A. Parameter Sweep Analysis of the BCC Model

In this section we discuss the results of parameter sweep analysis (PSA) carried out on the BCC model, to investigate the prolongation and the reduction of the clotting time in response to perturbed values of some reaction constants, chosen according to their meaning within the whole pathway. PSA was performed by generating a set of different initial conditions – corresponding to different parameterizations of the model – and then automatically executing the deterministic simulations with coagSODA.

The sweep analysis for single parameters (PSA-1D) was performed considering a logarithmic sampling of numerical values of each reaction constant within a specified range (with respect to its physiological reference value). The sweep analysis over pairs of parameters (PSA-2D) was performed by simultaneously varying the values of two reaction constants within a specified range, considering a logarithmic sampling on the resulting lattice. The logarithmic sampling allows to uniformly span different orders of magnitude of the parameters value using a reduced and fine-grained set of samples, therefore efficiently analyzing the response of the system in a broad range of conditions.

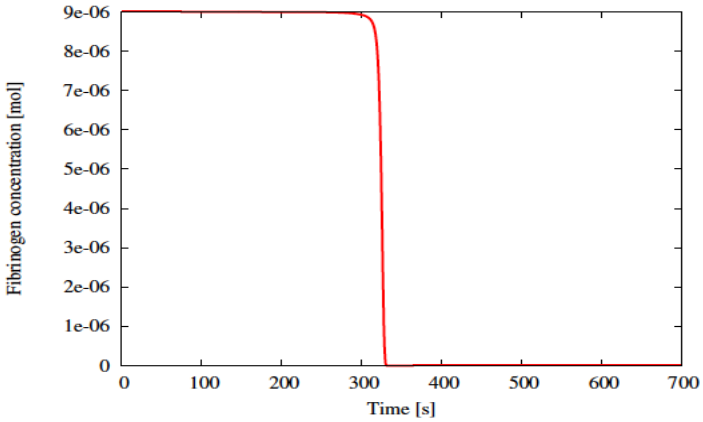

Fig. 4. Dynamics of fibrinogen in physiological condition

The output of the PSA is the *prothrombin time* (PT), generally used in laboratory tests for monitoring the therapy with anticoagulant drugs, which is defined as the time necessary to convert the 70% of the fibrinogen in fibrin (see Figure 4), conventionally assumed to correspond to the time necessary to form the clot [21]. According to the model defined in [12], the PT *in vivo* is around 300 seconds in physiological conditions. We investigate here the response of the BCC by evaluating the clotting time in different conditions – namely, we carry out PSA-1D over the constant of two reactions (hereafter called reactions α and β) taken separately, and PSA-2D over the two constants together – in order to monitor both their individual effects and any possible coupled effect on the robustness of the BCC model. In each PSA, the value of the constants (hereafter denoted as k_α and k_β) was varied of six orders of magnitude with respect to their physiological values, that is, three below and three above the reference value.

Reaction α, which describes the catalytical activation of factor V by factor IIa (IIa + V → IIa-Va), was chosen because it represents the main positive feedback within the extrinsic pathway; reaction β, which is involved in the activation of factor XI by binding to factor IIa (IIa + XI → IIa-XI), was chosen because it represents the main positive feedback in the intrinsic pathway of the BCC model (see Figure 1). Moreover, preliminary PSA over all reaction constants of the BCC model evidenced that these two reactions are among the most relevant steps of the coagulation network.

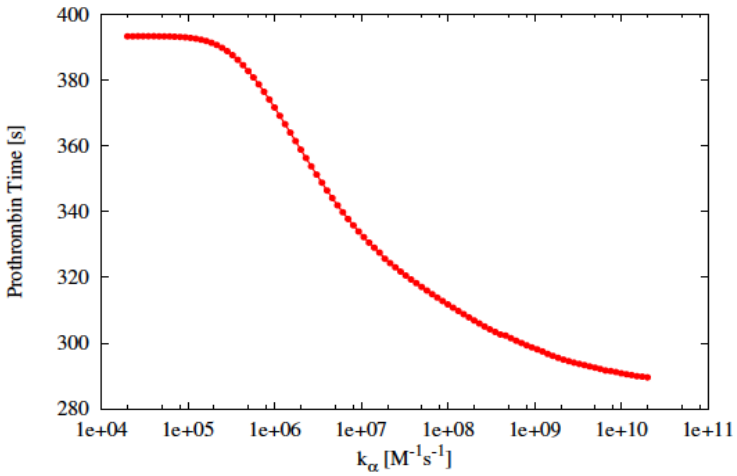

Fig. 5. Variation of prothrombin time (PT) according to PSA-1D over constant k_α of reaction α : IIa + V → IIa + Va. The reference value of k_α is $2 \cdot 10^7$, the sweep range is $[2 \cdot 10^4, 2 \cdot 10^{10}]$.

The PSA-1D over k_α shows that the PT is very sensitive to the perturbation of this reaction, when the physiological value of its reaction (i.e., $k_\alpha = 2 \cdot 10^7$ M^{-1} s^{-1}) is either increased or decreased; in particular, when k_α is very low, a plateau in PT is reached since the strength of the positive feedback exerted by factor IIa is largely reduced, a condition where the contribution of the amplification of the hemostatic stimulus, due by the common pathway, to the formation of the clot is basically not effective (Figure 5). On the other side, the PSA-1D over k_β shows that while a decrease of the constant from its physiological value (i.e., $k_\beta = 1 \cdot 10^8$ M^{-1} s^{-1}) does not have a substantial effect on the clotting time, an increase of its value leads to a progressive increase in the PT (Figure 6). This increase is due to the fact that factor IIa is sequestered in the formation of a complex with factor XI, and hence it is no longer

available as a free component in blood to participate in other reactions, especially those reactions of the extrinsic pathway which principally lead the clot formation *in vivo*. This behavior demonstrates that the intrinsic pathway has a secondary role in blood coagulation *in vivo*, compared with the extrinsic pathway, as also evidenced by various experimental observations [22].

Figure 7 shows the effect of the simultaneous variation of k_α and k_β (over the same sweep ranges considered in the two PSA-1D). This result remarks a synergistic interplay between the two analyzed reactions, especially when the value of constant $k\alpha$ is low and constant $k\beta$ is high, since in this condition both the extrinsic and common pathways are inhibited and the PT is higher with respect to the values it reaches when only a single constant is changed. The anticoagulant effect achieved in this condition is due to the lack of the positive feedback of factor IIa, especially on factor V, which results in a reduced conversion rate of the fibrinogen into fibrin.

3.2. CPU vs GPU Performance Comparison

We present here the comparison of the computational effort required by GPU and CPU for the simulation of the BCC model, to the aim of showing the relevant speedup achieved by coagSODA. The performances of our GPU simulator were compared with those obtained using the LSODA algorithm implemented in the software COPASI [23], executing on the CPU the same set of simulations that were run on CUDA. COPASI is single-threaded and does not exploit the physical and logical cores of the CPU, therefore it represents a good benchmark as a single-node CPU-bound simulator of biological systems.

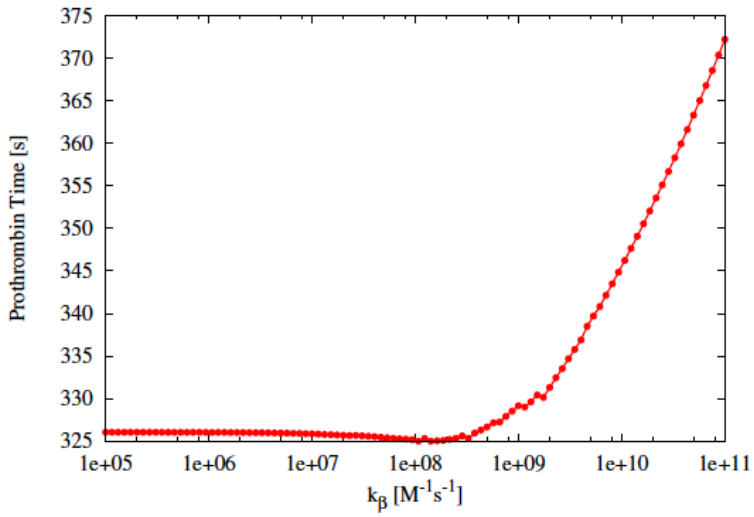

Fig. 6. Variation of prothrombin time (PT) according to PSA-1D over constant k_β of reaction β : IIa + XI → IIa-XI. The reference value of k_β is $1 \cdot 10^8$, the sweep range is $[1 \cdot 10^5, 1 \cdot 10^{11}]$.

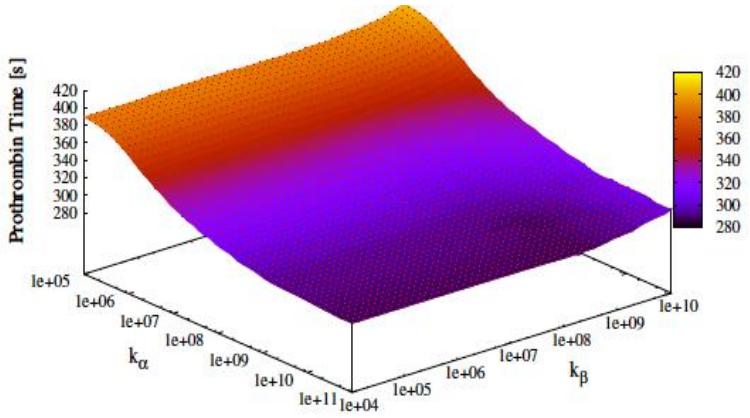

Fig. 7. Variation of prothrombin time (PT) according to PSA-2D over reaction constants k_α and k_β.

In all the simulations, executed both on GPU and CPU, we stored 100 samples – uniformly distributed in the time interval considered for each simulation, i.e., [0, 700] seconds – of the dynamics of all the chemical species involved in the BCC model. The settings for the LSODA algorithm were the following: relative error $1 \cdot 10^{-7}$, absolute error $1 \cdot 10^{-13}$, maximum number of internal steps set to 20000. The GPU used for the simulation is a Nvidia Tesla K20c, with 2496 cores, equipped with 5 GB of DDR5 RAM. The performance of the GPU were compared against a personal computer equipped with a dual-core CPU Intel Core i5, frequency 2.3 GHz, 4 GB of DDR3 RAM, running the operating system Mac OS X Lion 10.7.5.

In Table II we report the running times required to run several batches of simulations, executed to carry out the PSA over the reaction constant k_α, along with the speedup achieved on the GPU. The choice of comparing the performances of the GPU and CPU implementations by executing batches of simulations that are related to a PSA – instead of running *n* independent but identical simulations (i.e., all characterized by the same parameterization of the model) – is due to the fact that these results represent a more realistic scenario in the computational analysis of biological systems, whereby it is useful to investigate large search spaces of parameters, corresponding to different perturbed conditions of the model. Moreover, for the evaluation of the running time, the execution of a batch of *n identical* deterministic simulations would be futile. The figure clearly shows that coagSODA always performs better than the CPU counterpart. In particular, these results highlight that the advantage of exploiting the GPUs for the simulation of the BCC model becomes more evident as the number of simulations increases, with a 177× speedup when a PSA with 10000 different parameterizations is executed: while the CPU performance linearly increases with the number of simulations, the running times are strongly reduced on the GPU, since in this case the overall running time roughly corresponds to the time required for the execution of the slowest simulations. Indeed, the different running times are due to the effort to manage the execution of blocks of threads on the GPU, as well as to the different parameterizations used in the PSA, both factors that can greatly affect the computational performances. Therefore, the GPU accelerated analysis of the BCC model with coagSODA represents a novel, relevant computational mean to investigate the behavior of this complex biological system under non-physiological conditions, and could be exploited to efficiently determine the response of the BCC to different therapeutical drugs.

Table II. CPU vs GPU performance comparison

Number of simulations	CPU time (sec)	GPU time (sec)	Speedup
10	1060.4	120.81	8.7 ×
100	11652.8	902.12	12.9 ×
1000	107358.2	2070.89	51.8 ×
10000	1069981.8*	6027.69	177.5 ×

* Estimated running time

4. Conclusion

Thanks to the high-performance computing capabilities and to the very low costs, GPUs nowadays represent a suitable technology for the development and the application of parallel computational methods for *in silico* analysis of complex biological systems, to the aim of carrying out fast simulations and analyzing their emergent behaviors in different conditions. However, the implementation of efficient computational tools able to fully exploit the large potentiality of GPUs is still challenging, since good programming skills are required to implement GPU-based algorithmic methods, and to handle specific features of GPU computing, as the efficient usage of memory or the communication bandwidth between GPU and CPU. Moreover, algorithms cannot be directly ported on the GPU because of the limited programming capabilities allowed by GPU kernels; as a matter of fact, they need to be redesigned to meet the requirements of the underlying SIMD architecture.

In this work we presented coagSODA, a GPU-powered simulator specifically developed for the efficient simulation of the BCC model firstly introduced in [12]. coagSODA was designed to offer a black-box solution usable by any final user in an easy way. It relies on cupSODA, a numerical integrator for ODEs that we previously implemented for the GPU architecture, based on the LSODA algorithm and capable of automatically translating a reaction-based model into a set of coupled ODEs. coagSODA implements specific functions to compute the kinetics of mass-action and Hill function- based reactions, such as those involved in the platelets activity of the BCC model. coagSODA exploits the massive parallel capabilities of modern GPUs, and our results demonstrated that it can achieve a relevant reduction of the computational time required to execute many concurrent and independent simulations of the BCC dynamics. The mutual independence of the simulations allows to fully exploit the underlying SIMD architecture; moreover, coagSODA benefits from an additional speedup, thanks to our choice of storing each system state into the low-latency shared memory (a solution that was already implemented in cupSODA). Since the BCC model is large (101 reactions, 81 chemical species), a large amount of shared memory was assigned to each thread, strongly reducing the theoretical occupancy of the GPU, i.e., the ratio of active warps with respect to the maximum number of warps supported by each streaming multi-processor of the GPU.

The results of the analysis presented in this work, performed on the BCC model, show that coagSODA allows a relevant boost with respect to a reference CPU implementation. For instance, in the case of 10000 simulations to carry out the PSA of the model, we achieved a

noticeable 177× speedup. Interestingly, the performances of coagSODA are better than COPASI even for small batches of simulations. These results indicate that, for this kind of biological models, consisting in many reactions and many species, our GPU implementation becomes more profitable than the CPU counterpart as the number of concurrent simulations increases, making it suitable especially when performing demanding computational analysis such as, e.g., parameter sweep, parameter estimation or sensitivity analysis.

References

[1] L. Dematté and D. Prandi, "GPU computing for systems biology," *Briefings in Bioinformatics*, vol. 11, no. 3, pp. 323–33, 2010.

[2] J. Payne, N. Sinnott-Armstrong, and J. Moore, "Exploiting graphics processing units for computational biology and bioinformatics," *Interdisciplinary Sciences, Computational Life Sciences*, vol. 2, no. 3, pp. 213–20, 2010.

[3] M. J. Harvey and G. De Fabritiis, "A survey of computational molecular science using graphics processing units," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 2, no. 5, pp. 734–742, 2012.

[4] Y. Zhou, J. Liepe, X. Sheng, M. P. H. Stumpf, and C. Barnes, "GPU accelerated biochemical network simulation," *Bioinformatics*, vol. 27, no. 6, pp. 874–876, 2011.

[5] M. Vigelius, A. Lane, and B. Meyer, "Accelerating reaction-diffusion simulations with general-purpose graphics processing units," *Bioinformatics*, vol. 27, no. 2, pp. 288–290, 2011.

[6] M. S. Nobile, D. Besozzi, P. Cazzaniga, G. Mauri, and D. Pescini, "cupSODA: a CUDA-powered simulator of mass-action kinetics," in Proceedings of 12th International Conference on Parallel Computing Technologies (PaCT 2013), V. Malyshkin, Ed., vol. LNCS 7979, 2013, pp. 344–357.

[7] L. Petzold, "Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations," *SIAM Journal of Scientific and Statistical Computing*, vol. 4, no. 1, pp. 136–148, 1983.

[8] K. Rogers, *Blood: Physiology and Circulation*, ser. The Human Body. Britannica Educational Pub., 2010.

[9] R. Colman, J. Hirsh, V. Marder, A. Clowes, and J. George, Eds., *Hemostasis and thrombosis: basic principles and clinical practice*. Lippincott Williams & Wilkins, 2006.

[10] A. Michelson, *Platelets*, 2nd ed. Elsevier, 2007.

[11] D. Voet and J. Voet, *Biochemistry*. Wiley, 2011.

[12] M. S. Chatterjee, W. S. Denney, H. Jing, and S. L. Diamond, "Systems biology of coagulation initiation: Kinetics of thrombin generation in resting and activated human blood," *PLoS Computational Biology*, vol. 6, no. 9, p. e1000950, 2010.

[13] M. F. Hockin, K. C. Jones, S. J. Everse, and K. G. Mann, "A model for the stoichiometric regulation of blood coagulation," *Journal of Biological Chemistry*, vol. 277, no. 21, pp. 18 322–18 333, 2002.

[14] T. Wajima, G. Isbister, and S. Duffull, "A comprehensive model for the humoral coagulation network in humans," *Clinical Pharmacology & Therapeutics*, vol. 86, no. 3, pp. 290–298, 2009.

[15] K. Lo, W. S. Denney, and S. L. Diamond, "Stochastic modeling of blood coagulation initiation," *Pathophysiology of Haemostasis and Thrombosis*, vol. 34, no. 2-3, pp. 80–90, 2006.

[16] C. M. Danforth, T. Orfeo, K. G. Mann, K. E. Brummel-Ziedins, and S. J. Everse, "The impact of uncertainty in a blood coagulation model," *Mathematical Medicine and Biology*, vol. 26, no. 4, pp. 323–336, 2009.

[17] R. Farber, "Topical perspective on massive threading and parallelism," *Journal of Molecular Graphics and Modelling*, vol. 30, pp. 82–89, 2011.

[18] Nvidia, "Nvidia CUDA C Programming Guide v5.0," 2012.

[19] O. Wolkenhauer, M. Ullah, W. Kolch, and C. Kwang-Hyun, "Modeling and simulation of intracellular dynamics: choosing an appropriate framework," *IEEE Transactions on Nanobiosciences*, vol. 3, no. 3, pp. 200–7, 2004.

[20] D. Nelson and M. Cox, *Lehninger Principles of Biochemistry*. W. H. Freeman Company, 2004.

[21] B. H. Esteridge, A. P. Reynolds, and N. J. Walters, *Basic medical laboratory techniques*, 4th ed. Cengage Learning, 2000.

[22] T. Renne´, A. H. Schmaier, K. F. Nickel, M. Blombaeck, and C. Maas, "In vivo roles of factor XII," *Blood*, vol. 120, no. 22, pp. 4296–4303, 2012.

[23] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer, "COPASI - a COmplex PAthway SImulator," *Bioinformatics*, vol. 22, no. 24, pp. 3067–3074, 2006.