Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Feasibility Study of Multi-Agent Simulation at Cellular Level with FLAME GPU

Alcione de Paiva Oliveira[*]
Universidade Federal de Viçosa
Departamento de Informática,
365700-900 Viçosa, Brazil

Paul Richmond
The University of Sheffield
Western Bank, Sheffield S10 2TN, UK

Abstract

Multi-Agent Systems (MAS) are a common approach to simulating biological systems. Multi-agent modelling provides a natural method for describing individual level behaviours of cells. However, the computation cost of simulating behaviours at an individual level is considerably larger than top down equation based modelling approaches. A recent possibility to improve computational performance is the use of Graphics Processing Units (GPUs) to provide the necessary parallel computing power. In this paper we show that multi-agent models describing biological systems at cellular level are well suited to GPU acceleration. Cellular level systems are characterised by vast numbers of agents that intensively communicate, indirectly through diffusion of chemical substances, or directly, through connection of chemical receptors. We present a study which utilises the FLAME GPU software to target a MAS model of a generic pathogen induced infection to validate the suitability of the GPU for simulation of a broader class of cellular level systems.

Keywords:Multi-Agent Systems, Simulation at cellular level, GPU, Flame GPU

Introduction

Multi-agent modelling is a natural technique for describing and simulating complex systems, which are characterised by interactive autonomous individuals. However, multi-agent systems are computationally expensive, when compared with a more traditional top down approach of systems modelling. This is especially true when simulating phenomena that have a huge numbers of elements such as biological cellular systems or systems containing swarm behaviour. Parallel and distributed simulation methods offer a potential solution to the computational scale required to simulate multi-agent systems. Modern graphics cards (GPUs-Graphics Processing Units) originally designed to accelerate the rendering of real time computer graphics have previously been shown to offer a potential solution for efficient execution of a limited subset of complex multi-agent systems (Richmond et al. 2010; D'Souza, Lysenko, and Rahmani 2007). In such cases simulation performance scales well as the number of agents increases.

This paper reviews previous literature, comparing various studies which consider the applicability of GPU implementations of multi-agent systems within the context of cellular level biological modelling. In order to give further insight into the relevance of GPUs towards biological simulation we then propose a model which extends the biological complexity of previous work by capturing common but necessary biological mechanisms. Representing these mechanisms ensures that the results give a fair insight into how this disruptive technology can impact on the broader class of cellular level biological multi-agent systems simulation.

Multi-agent models for simulating biological systems at a cellular level have specific and unique requirements that may not appear in other kinds of MAS. For example, vast numbers of agents that intensively communicate both indirectly through diffusion of chemical substances, or directly, through connection of chemical receptors. Other relevant features include agents (cells) moving in a 3D continuous space environment; zones that communicate representing different organs; complex behaviour and agent rules; and intensive removal and addition of agents.

The immune system is an example of a complex biological system which demonstrates all of the above challenges. Its application to GPUs therefore demonstrates the applicability of GPUs towards a broader class of cellular systems which utilise the same complex mechanisms and behaviours. Furthermore, Li et al. (hua Li et al. 2009) state that computational models of the immune system might help researchers to understand its mechanisms and to verify their hypothesis in a very effective way, avoiding the slowness and high cost of *in-vivo* investigations.

The study described in this paper investigates the feasibility of exclusively using Graphics Processing Units (GPUs) in the simulation of large scale biological cellular systems through a case study of the immune system. The immune system simulation is implemented using the Flame GPU framework (Richmond 2011) which has been designed for the execution of multi-agent systems on NVIDIA GPUs supporting the Compute Unified Device Architecture (CUDA). The simulation consists of a generic pathogen induced infection.

The main contributions of this work are to show the viability of capturing the complex mechanisms and behaviours associated with biological cellular-level simulation using

[*]The author receives a grant from CAPES, process n.0449/15-6.

GPUs. As part of this contribution we present a first pilot implementation of a immune system model entirely within the GPU architecture and report on its performance characteristics.

Related works

There are several studies that analyse the application of GPUs to biological multi-agent systems. Chen et al. presented a simulation of the blood coagulation system (Chen et al. 2011). They implemented the simulation using a GPU specific implementation, a C specific implementation and via two MAS platforms: NetLogo, Repast in order to perform direct comparisons. In their experiments, it was demonstrated that the computational speed of the GPU implementation of the million-level scale of agents was over 10 times faster than the direct C version, over 100 times faster than the Repast version, and over 300 times faster than the NetLogo simulation. The simulation presented by Chen et al. differs considerably from our own, in the sense that the substances do not undergo diffusion. The implications of this are that in the blood coagulation system model, the information that agents access remains static, i.e. the environment is not modified during execution of the simulation. Additionally, the biological behaviour of the agents is homogeneous creating conditions ideally suited to efficient implementation on a GPU.

D'souza et al. (D'Souza, Lysenko, and Rahmani 2007) also investigated Graphics Processing Units (GPUs) as an alternative platform for agent based simulations. They implemented a simulation of the problem called Sugarscape (Epstein and Axtell 1996). According to them, they were "able to achieve over 50 updates per second with agent populations exceeding 2 million on an environment with a resolution of 2560x1024 with visualization." While this is a significant result, the study has no relationship with the problem addressed here, because the problem Sugarscape is implemented as a cellular automata and not as agents that move in a continuous space.

Richmond et al. (Richmond et al. 2010) developed a Keratinocyte (cell) model of the *in vitro* behaviour of skin epithelial cells. As described by (Richmond et al. 2010), the model includes behaviour representing progression of the cell cycle, including cell growth and division (proliferative behaviour), leading to cell birth and the addition of agents to the model. The performance of the Keratinocyte model indicates that a GPU is highly suitable to cellular level simulation however the interactions between cells are only over short distances. In order to understand the impact of mid range or long range interactions (such as that of hormones or cytokines) further investigation is required. Our paper builds upon this work by presenting a more complex modelling example which capture these essential mechanisms.

A technique which has investigated the use of the GPU in substance diffusion calculations is the work of Romão et al. (Romão et al. 2012) which compares the performance against a counterpart CPU implementation. Whilst capturing the diffusion behaviours, the individual cellular behaviour is neglected giving no indication of the computational cost of the overhead of communication between cells and substances. The aim of out paper is to combine both the diffusion and cellular level interactions to give necessary insight into the performance implications of complex biological models on GPUs.

Our immune system model is based on the work of Possi et al. (Possi et al. 2011) which presents a comprehensive model of the human immune system, addressing both the innate immunity as adaptive immunity (the one that is developed after a first infection). The original model is implemented on the Repast framework (North et al. 2013) and is only suitable for execution within a serial processing environment. Our model differs from the original in that, in we implement a specific subset of the entire model (i.e. an immune system with innate immunity). This subset demonstrates the implementation of key biological mechanisms and behaviours which are persistent in a wider class of biological simulation. In addition, we must deal with the lack of functionality to implement a substances diffusion calculation within the Repast framework.The reported performance of the original model vs our own gives significant insight into the performance implications of simulation with the CPU in Repast vs. with the GPU with FLAME.

Bacteria-macrophage-antibiotic interaction model

A complex, challenging biological system that possess all the features mentioned on the Introduction section namely is the human immune system. Due to its variety of cell types and the complex interactions between them, it has been considered to be even more challenging than the human neural system (DasGupta 1999). As such, to confirm the viability of GPU acceleration for complex biological systems at cellular level, we consider a model of reproductive pathogen behaviour (e.g. bacteria) controlled by an immune response agent (e.g. macrophage). In addition, we simulate the diffusion of a substance that could interfere in the behaviour of the agents. For example, an amount of an antibiotic substance injected in an arbitrary position of the simulated environment. As a result our model demonstrates how bacteria attack of tissue cells would cause the appearance of substances that attract immune cells to the site of the infection. Thus, the model requires the simulation of four listed fundamental requirements for biological simulation: 1) near communication through receptors; 2) remote communication through cytokines; 3) random and directed movement; 4) creation and removal of agents dynamically during simulation.

A bacteria is a unicellular organism that can live harmlessly in the human body or can act as a pathogen, causing diseases and, potentially, death. In our immune system model, the bacteria agent randomly moves one unit position within a space at each iteration, periodically dividing. According to Tortora et al. (Tortora, Funke, and Case 1995), as a bacterium can typically move about 50 $\mu m/sec$, and one unit in our model is $10\mu m$, then one iteration corresponds about 1/5sec. Bacteria infects tissue cells present in the system. Tissue cells are static representing the biological environment and do not move, if they are attacked by bacteria they emit a substance which attracts macrophages to the site of the infection. The macrophage agent moves in the same way as bacteria. When it encounters a bacteria agent at, it kills the bacteria by phagocytosis. When a macrophage detects the occurrence substances issued by tissue cell they move towards the infection site. The number of macrophage agents is constant throughout the simulation.

Bacteriostatic substances play an additional role within the model. Bacteriostatic substances are ‘injected’ at a random locations and, at each interaction, the substance concentration diffuses to adjacent positions. The presence of a bacteriostatic substance prohibits the division of cells.

Equilibrium of the model can result on a number of possible outcomes which are 1) the elimination of all bacteria agents; 2) a stabilisation of the number of bacteria agents; or 3) an explosion of number of bacteria agents.

For the simulation we adopt the parameters displayed in the table .

According to Folcik et al. (Folcik, An, and Orosz 2007), if one assumes the average diameter of a cell to be approximately 0.01 mm and as we’re using 0.02 mm as intercellular space, then the simulation represents an area of about 560×0.03 ≈ 17.0 mm^2 of tissue.

The FLAME GPU implementation

The FLAME GPU framework (Richmond 2011) has the following features required for implementing the immune system model: 1) continuous and discrete agents; 2) messages sent by agents with a fixed distance to reach; and 3) global constants. According to (Richmond 2011) an agent in FLAME GPU is a represented as a X-Machine (a form of state machine) and consists primarily of a name, an internal memory set (M in the formal definition), a set of agent functions (or next state partial functions, F, in the formal definition) and a set of states (Q in the formal definition). X-Machine agents communicate only through messages. This communication allows dependencies of agents to be calculated and displayed as a directed graph. The representation of agents as a state machine is shown in Fig. 1.

In the diagram the ellipses represent the agent states; simple rectangles denote state transition functions (operations on memory which represent blocks of agent behaviour); and diamonds represent messages. Functions are unable to send and receive messages of the same type to avoid race conditions during execution.

The diagram is divided into three sections, each comprising a machine-X for each of the three modelled agents.

The bacteriostatic substance was implemented as a set of global variables. A global variable is a static value (or array of values) which is mapped to CUDA constant memory. The constant memory cache makes access of global functions exceptionally fast for all agents.

Within FLAME GPU each spatially located message sent by an agent has a fixed distance radius for agents that can read it. As the radius increases so too does the global cost of message reading (as more messages will be considered by each agent). It is therefore preferable to use the minimum possible interaction radius to obtain high performance. This presents somewhat of a problem for the diffusion of substances such as pro and anti-inflammatory substances which require variable radius due to its gradual diffusion.

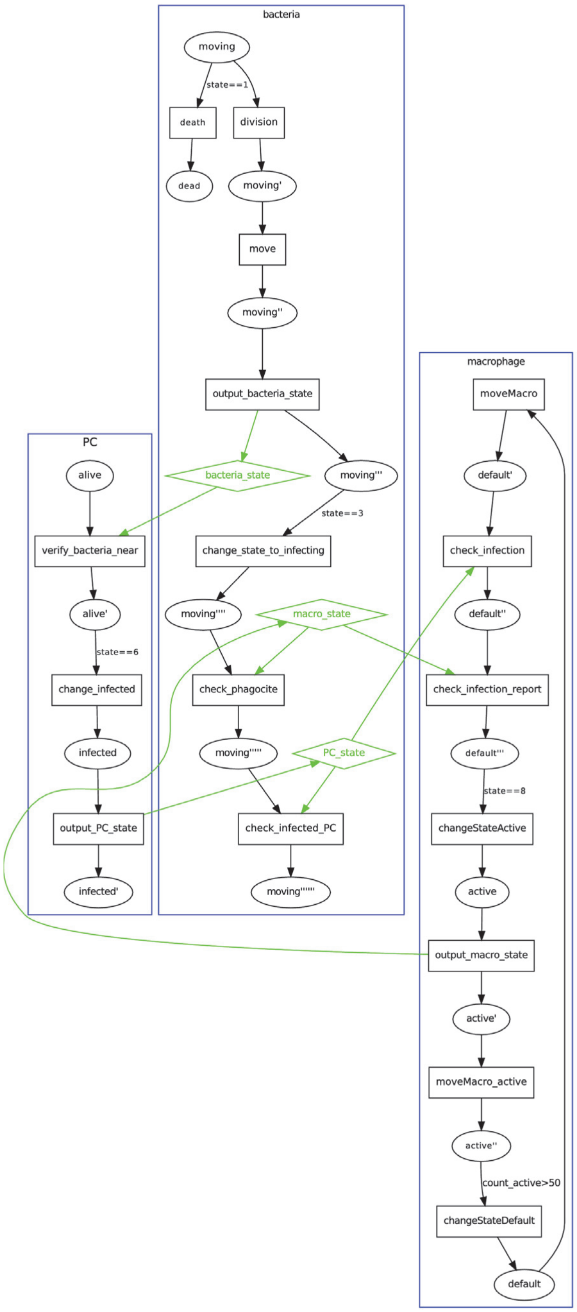

Figure 1: Diagram showing each immune system agent modelled as a X-machine and their interactions via messages (green). The model diagram shows the Macrophage × bacteria × tissue cell (PC - Parenchyma cells) interaction. In other words, an aspect of innate immunity.

To circumvent the requirement of a variable range message it would be possible to specify a maximum possible

No. tissue cells	*No. Bacteria*	*divide time*	*bacteriostatic efect*	*No. Macrophage*	*Total No. agents*
32000	10	every 10 iterations	reduces the chance of bacteria division in 95%	160000	560010

Table 1: Initial parameters adopted in the simulations. The simulation area corresponds to approximately 1.7 cm^2 of tissue.

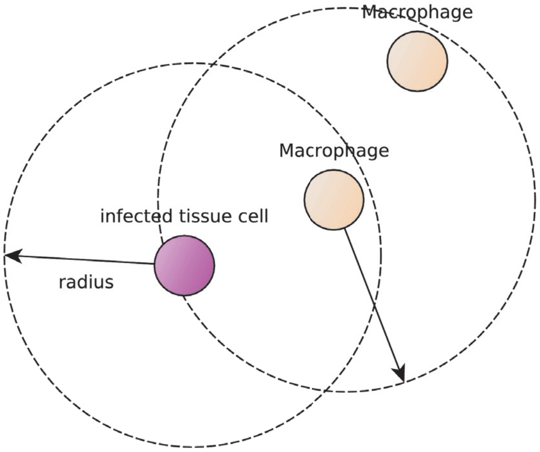

Figure 2: Illustration of how messages are forwarded to simulate diffusion. When a tissue cell is infected it sends a short range message. After that, if the message is received by a macrophage in the local neighbourhood, it forwards the message over a wider area to reach more distant macrophages.

range, however this would lead to significant performance degradation. The solution employed within our model is to use a technique of message forwarding between the tissue and macrophage agents. For example, when a tissue cell is infected it sends a short range message. If the message is received by a macrophage in the local neighbourhood, it forwards the message over a wider area to reach more distant macrophages. This behaviour has correspondence with the biological reality and minimises the amount of long range interaction. The Fig. 2 illustrates the approach adopted.

Results

The purpose of benchmarking our model is to give insight into the expected performance of the relevant biological mechanisms on non specialist desktop hardware (representing a conceivable setup of a regular biological modeller). The simulation was performed on a notebook with the following configuration: i7 processor, 8GB Ram, NVIDIA GeForce 830M graphics card with 2GB dedicated memory. It is important to note that both simulations (of the proposed model and of the original model from Possi et al.) were performed on the same hardware confuguration. The Fig. 3 shows a snapshot of a simulation where the defence cells (macrophages) have not been able to contain the infection and the number of bacteria are growing exponentially to a power of 2 at each iteration.

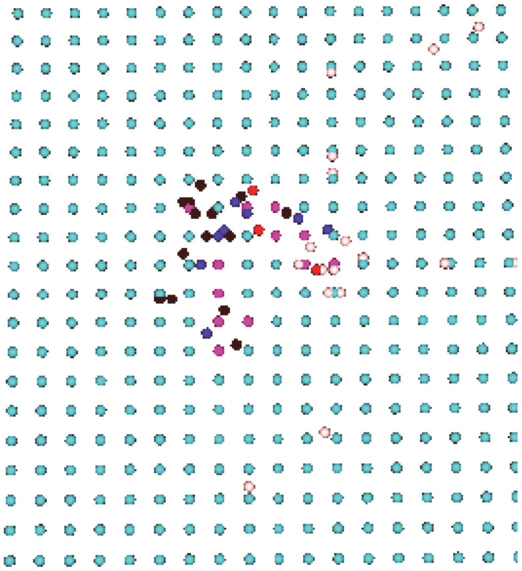

Figure 3: Snapshot of a simulation. The pink circles denote macrophages; the blue circles denote the tissue cells; the violet circles denote the infected tissue cells; brown circles denote bacteria; and red circles denote dead bacteria.

Using our modest hardware configuration, the simulation results without the bacteriostatic substance are shown in Table . In this case the population of bacteria grew exponentially and the infection progresses exponentially. The execution was interrupted in step 302 due to exceeding the maximum user defined agent population size.

Table shows the simulation results with the bacteriostatic substance. In this case the bacteria was completely eliminated, even using a larger initial number of bacteria agents, and the iteration was stopped at the 1000 iteration only because there was no more changes in the simulation.

Our results show a five times improvement in scaling performance compared to the comparable simulation performed by Possi et al. (Possi et al. 2011). Additionally the size of the simulated area differs greatly. While Possi et al. used a discrete grid of 150X150, the continuous space used in our simulations was 2000x2000 suggesting that our FLAME GPU implementation is memory efficient. Secondly there is the population size agents. While Possi et al. simulated around 10,000 cells we were able to exceed one million agents. Finally there is a performance gain. In 35 seconds the Possi simulator was able to accomplish only 34 iterations. Even neglecting the size of the simulated area and the number of actors involved the performance increase is around 10 times.

No. iterations	*time (seconds)*	*No. Bacteria*	*No. Tissue cells*	*No. Macrophage*	*Total No. agents*
0	0	10	320000	160000	482000
302	35	679092	320000	160000	1159902

Table 2: Model execution results without bacteriostatic substance. The population of bacteria grew exponentially.

No. iterations	*time (seconds)*	*No. Bacteria*	*No. Tissue cells*	*No. Macrophage*	*Total No. agents*
0	0	100	320000	160000	480100
1000	30	0	320000	160000	480000

Table 3: Model execution results with bacteriostatic substance. In this case the bacteria was completely eliminated.

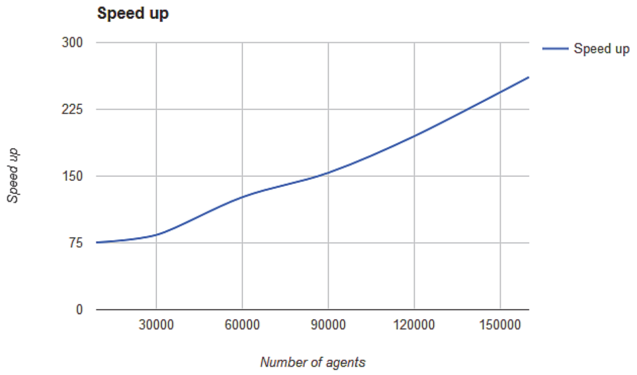

Figure 4: Speedup. It indicates how many times our approach was faster than the baseline version.

It should be noted that the limiting factor of our FLAME GPU implementation in this case is the mamximum agent population size (which is ultimately dictated by the available GPU memory) when the number of agents grows exponentially, in which case the simulation outcome is defined.

A more appropriate comparison should be made with an approximate number of agents. A simulation with an identical number of agents is difficult to obtain due to the different rates of reproduction and death of the agents in each model. For this, we changed the original size of AutoSimmune tissue cell grid to get 160,000 tissue cells. The result of this simulation, the speedup obtained with Flame GPU can be seen in Table . The speedup is a measure of performance improvement from the baseline version. Note that in this case the GPU Flame version was 261 times faster than the version using the CPU based simulator, which clearly demonstrates the performance advantage in the GPU hardware use.

We also carry out various simulations increasing gradually the number of agents in order to verify how the performance gain behaved. The result can be seen in Fig. 4. It may be noted that as the number of agents increases the relative performance gain increases. This can be explained by the fact that the implementation of multi-agent systems in sequential hardware degrades much faster than the version that uses parallel hardware.

Conclusions

This paper presents the implementation of a model in order to test the feasibility of simulation exclusively for GPUs at the cellular level of complex biological systems, with massive number of agents and intensive exchange of messages. The implemented model was intended to capture biological mechanisms in a general way rather than represent a high degree of biological accuracy. The model extends the complexity of biological mechanisms represented beyond that of work presented in the literature, combining for example, individual cell behaviours and diffusion. The system showed higher scaling and simulation performance when compared to complex models implemented in CPUs.

Regarding the requirements for biological simulation, mentioned at the beginning of the paper, the model was capable of representing each of them. The near communication through receptors is simulated by means of short range messaging. The remote communication through cytokines is simulated by means of forwarding short range messaging. The random and directed movement is implemented by the movement of agents in the Flame GPU framework. Finally, the creation and removal of agents dynamically during simulation is already implemented in the Flame GPU framework.

The execution on GPUs is attractive once it allows the construction of systems with high performance at low cost. In addition, the biological behavior at the cellular level is simple enough to be simulated by X-machines. However, modeling is still quite complex and challenging, consuming much of the development time and keeping away users who do not master the architecture. Future versions of frameworks should focus on ease-of-use to popularize its use. The next step is the simulation of models that has a greater fidelity to the biological reality.

Acknowledgements. This research is supported in part by the funding agencies FAPEMIG, CNPq, and CAPES.

References

Chen, W.; Ward, K.; Li, Q.; Kecman, V.; Najarian, K.; and Menke, N. 2011. Agent based modeling of blood coagulation system: implementation using a gpu based high speed framework. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, 145–148. IEEE.

Simulator	*No. iterations*	*time (seconds)*	*seconds per ticks*	*No. cells*	*Speedup*
Flame GPU implementation	100	2,696	0,02	≈ 160000	261,01
AutoSimmune	100	730,69	7,307	≈ 160000	0

Table 4: Speedup table: how many times our approach was faster than the baseline version.

DasGupta, D. 1999. *An overview of artificial immune systems and their applications*. Springer.

D'Souza, R. M.; Lysenko, M.; and Rahmani, K. 2007. Sugarscape on steroids: simulating over a million agents at interactive rates. In *Proceedings of the Agent 2007 Conference*, volume 20.

Epstein, J. M., and Axtell, R. 1996. *Growing artificial societies: social science from the bottom up*. Brookings Institution Press.

Folcik, V.; An, G.; and Orosz, C. 2007. The Basic Immune Simulator: An agent-based model to study the interactions between innate and adaptive immunity. *Theoretical Biology and Medical Modelling* 4(1):39+.

hua Li, X.; xuan Wang, Z.; yang Lu, T.; and jiu Che, X. 2009. Modelling immune system: Principles, models, analysis and perspectives. *Journal of Bionic Engineering* 6(1):77 – 85.

North, M. J.; Collier, N. T.; Ozik, J.; Tatara, E. R.; Macal, C. M.; Bragen, M.; and Sydelko, P. 2013. Complex adaptive systems modeling with repast simphony. *Complex adaptive systems modeling* 1(1):1–26.

Possi, M. A.; Oliveira, A. P.; Chaves, C. M.; Cerqueira, F. R.; and Arroyo, J. E. 2011. An in-silico immune system model for investigating human autoimmune diseases. In *XXXVII Conferencia Latinoamericana de Informática (XXXVII CLEI)*. sn.

Richmond, P.; Walker, D.; Coakley, S.; and Romano, D. 2010. High performance cellular level agent-based simulation with flame for the gpu. *Briefings in bioinformatics* 11(3):334–347.

Richmond, P. 2011. Flame gpu technical report and user guide. Technical report, technical report CS-11-03. Technical report, University of Sheffield, Department of Computer Science.

Romão, O. C.; de Souza Amorim, L. E.; Ferreira, R. S.; Possi, M. A.; and Oliveira, A. P. 2012. Multiagent systems modeling using gpus–a case study of the human immune system. In *Computer Systems (WSCAD-SSC), 2012 13th Symposium on*, 234–241. IEEE.

Tortora, G.; Funke, B.; and Case, C. 1995. *Microbiology: An Introduction*. Redwood City, California: The Benjamin/Cummings Publishing Company.