BAM: BioAmbient Machine

Vinod A. Muganthan
Department of Computing
Imperial College
South Kensington Campus
SW7 2AZ, London, UK
vinod.aravind@gmail.com

Andrew Phillips
Microsoft Research
7 J J Thomson Avenue
CB3 0FB, Cambridge, UK
andrew.phillips@microsoft.com

Maria G. Vigliotti
Department of Computing
Imperial College
South Kensington Campus
SW7 2AZ, London, UK
mgv98@doc.ic.ac.uk

Abstract

In recent years it has become clear that techniques developed for concurrent programming are in fact very useful for the analysis of complex systems in biology. To some level of abstraction, it is possible to regard biological phenomena, such as gene transcription, protein interaction and communication among cells, as concurrent entities that compute according to the rules of chemical reactions. In this paper we present a tool for representing and stochastically simulating complex biological systems with mobile compartments. BAM, our tool, implements a stochastic version of the BioAmbient calculus and helps in understanding and reasoning about the multiple interconnections between components in bioregulatory networks.

1. Introduction

The need for formal models in order to better understand complex systems in biology has been advocated by both computer scientists [6, 19] and biologists [11, 12]. Kohn [11] states ”Depicting the molecular networks involved in signaling pathways that regulate cell function has proven challenging, due to the enormous amount of information that needs to be conveyed for each participant in the network and the cross-connections between pathways. This challenge must nevertheless be addressed in order to understand the underlying design of such networks, and to utilize the findings of modern biology most effectively to combat diseases... Another difficulty is that bioregulatory networks are replete with interconnections and loops that make intuition about network function unreliable”. Kohn depicts the problem of understanding the behaviour of complex biological networks in a similar way that computer scientists depict issues in understanding concurrent computer networks. To address the problem of understanding and analysing the interconnections among different elements in biological networks, Kohn invented *molecular interaction maps* (MIMs) [12], which can be regarded as a graphical concurrent language. Compositionality is the main advantage of MIMs, yet their greatest limitation is the lack of an operational semantics, which largely limits any analysis of the behaviour of biological networks. This shortcoming is not present in process algebra, which has been augmented with stochastic metrics and successfully used to simulate biological networks and signalling pathways [14, 4, 5, 8, 2]. It naturally represents – up to some level of abstraction – simple synchronisation of independent events acting in parallel. While standard process algebra has been of great help in understanding the intricate relationship among components in signalling pathways, it is generally known that interacting molecules often show location-dependent behaviour [16]. It was argued [16] that signalling of proteins can behave differently depending on their position in different compartments of the cell, e.g. late endosome, lysosome or other vesicles. One typical example of this kind of pathway is receptor-mediated endocytosis [15, 20]. Modelling membrane interactions can be a step towards producing more accurate models of biological networks. The BioAmbient calculus [18] has been developed for the purpose of modelling membrane interactions [20], and in this paper we present a tool based on a stochastic version of BioAmbients. Simulation tools are often necessary in systems biology when trying to understand very large models, since analysis by hand of such models is often unfeasible. Although we have emphasised the use of our tool for systems biology, it should not be forgotten that the Ambient Calculus was initially devised to represent mobile computation [7]. The calculus was later augmented with a stochastic metric [21], while an alternative stochastic metric was later presented for BioAmbients in [3]. The BioAmbient Machine implements essentially a superset language of the Stochastic Ambient Calculus, making it viable for simulation of large distributed mobile systems.

2 Stochastic BioAmbients

In this section we briefly introduce our stochastic version of the BioAmbient calculus. We modify the syntax of the calculus by using explicit recursion as opposed to replication, by introducing the delay operator τ_δ and by allowing ambients to have a name.

We assume the existence of a set of names $\mathcal{N}$, and let the meta-variables $n, m, \ldots$ range over this set.

Definition 2.1 *The set of processes of BioAmbients is given by the following syntax:*

$$\begin{array}{rcl} P & ::= & \mathbf{0} \mid P \mid P \mid (\mathsf{new}\, n)\, P \mid a\,[P] \mid \mathsf{A}\langle\widetilde{x}\rangle \\ & & \mid \sum_{i\in I} M_i.P_i \\ M & ::= & \mathbf{enter!}\,n \mid \mathbf{exit!}\,n \mid \mathbf{enter?}\,n \mid \mathbf{exit?}\,n \mid \tau_r \\ & & \mid \mathbf{merge!}\,n \mid \mathbf{merge?}\,n \mid \$n?(x) \mid \$n!\langle m\rangle \\ \$ & ::= & \mathbf{s2s} \mid \mathbf{local} \mid \mathbf{p2c} \mid \mathbf{c2p} \end{array}$$

We assume that each name has a unique rate associated to it, and that there is an environment $\rho : \mathcal{N} \to \mathbb{R}^+ \cup \{\infty\}$ that formally keeps track of the rate associated to names. In BioAmbients communication happens on a channel n by sending and receiving on this channel. $\$n?(x)$ stands for the input, while $\$n!\langle m\rangle$ stands for the output. There are three ways of communicating: channels in the same ambient perform *local communication* $\mathbf{local}\,n?(y)$ for the input on channel n and $\mathbf{local}\,n!\langle m\rangle$ for the output on channel n of m. Inputs and outputs located in sibling ambients respectively perform *sibling communication* ($\mathbf{s2s}\,n?(y)/\mathbf{s2s}\,n!\langle m\rangle$). Parent to child communication happens when outputs $\mathbf{p2c}\,n!\langle m\rangle$ and inputs $\mathbf{p2c}\,n?(m)$ are located in parent/child ambients respectively. Similarly for the child to parent communication. The capabilities such as $\mathbf{exit!}\,n$ or $\mathbf{enter!}\,n$ give the ambient the ability to become mobile. $\mathbf{enter!}\,n/\mathbf{enter?}\,n$ allow an ambient to move into a sibling. $\mathbf{exit!}\,n/\mathbf{exit?}\,n$ allow a child ambient to leave its parent, while $\mathbf{merge!}\,n/\mathbf{merge?}\,n$ fuse two sibling ambients into a single ambient.

As far as processes are concerned, *Nil* represents the inactive process. Delay $\tau_r.P$ is a process that delays for an amount of time that is exponentially distributed by rate r before becoming P. *Local sum* $\sum_{i\in I} M_i.P_i$ represents the standard competitive choice (race condition). Given a set of indices I and a permutation p on it, we write $\sum_{p(i)\in I} M_{p(i)}.P_{p(i)}$ to represent a reordering of the terms of the summation. We reserve the letters G, C to represent summation as in $\sum_{i\in I} M_i.P_i = M_j.P_j + G$ where $G = \sum_{i\in I\ i\neq j} M_i.P_i$. In general, inputs are binding operators on the arguments. This means that in the process $\mathbf{local}\,n?(y).P$ the name y is bound in P, and not accessible from outside P. A similar argument applies to the other inputs in the communication primitives. *Ambient* $a\,[P]$ represents a compartment named a with an active process P.

$$\begin{array}{lcll} P \mid \mathbf{0} & \equiv & P & \\ P \mid Q & \equiv & Q \mid P & \\ (P \mid Q) \mid R & \equiv & P \mid (Q \mid R) & \\ (\mathsf{new}\,n)\,\mathbf{0} & \equiv & \mathbf{0} & \\ (\mathsf{new}\,m)\,(\mathsf{new}\,n)\,P & \equiv & (\mathsf{new}\,n)\,(\mathsf{new}\,m)\,P & \\ (\mathsf{new}\,n)\,(P \mid Q) & \equiv & P \mid (\mathsf{new}\,n)\,Q & \text{if } n \notin \mathsf{fn}(P) \\ (\mathsf{new}\,m)\,a\,[P] & \equiv & a\,[(\mathsf{new}\,m)\,P] & \text{if } a \neq m \\ \sum_{i\in I} M_i.P_i & \equiv & \sum_{p(i)\in I} M_{p(i)}.P_{p(i)} & \\ \mathsf{A}\langle\widetilde{m}\rangle & \equiv & P\{\widetilde{m}/\widetilde{x}\} & \text{if } A(\widetilde{x}) = P \end{array}$$

Figure 1. Structural congruence

The name of the ambient has no semantic role, but it is useful for identifiying ambients in a simulation. *Parallel composition* $P \mid Q$ means that P and Q are running in parallel. *Restriction* $(\mathsf{new}\,a)\ P$ of the name a makes that name private and unique to P: the name a becomes bound in P. *Recursion* $\mathsf{A}\langle\widetilde{x}\rangle$ models infinite behaviour by assuming the existence of a set of equations of the form $A(\widetilde{x}) \overset{\text{df}}{=} P$ such that $\{\widetilde{x}\} \subseteq \mathsf{fn}(P)$, where $\mathsf{fn}(P)$ stands for the usual free names of P. The definition of $\mathsf{fn}(P)$ is standard taking into account that the only binding operators are inputs and restriction. We write $P\{m/y\}$ to mean the substitution of every free occurrence of the name y by m in P. Steps of computation are defined by the *reduction relation* which is defined in Figure 2, while *structural congruence* $\equiv$ is defined in Figure 1. The reduction relation specifies how terms evolve syntactically, while the rates given by the environment ρ are sufficient to determine the probability and duration of each reduction. We assume that each transition is exponentially distributed with parameter given by $\rho(n)$ for the name n involved in the transition. We allow a special rate *infinity* which means, in practical terms, that we allow immediate transitions. This is useful for expressing actions that always take priority, as well as actions that can change the state space without being delayed. We omit in this paper the formal description of the derivation of Continuous Time Markov Chains given a stochastic process algebra (for the fragment of the language without immediate transition) since it is standard [10, 1, 17, 21].

3 Tool overview

The BioAmbient Machine is a simulator for stochastic BioAmbients written in Java 1.5. The core functionality of the tool is to simulate the behaviour of the stochastic BioAmbients according to the Gillespie algorithm [9]. The tool automatically produces both a simulation graph and a debugging trace of the program, which is a sequence of triples consisting of the following: a real number that

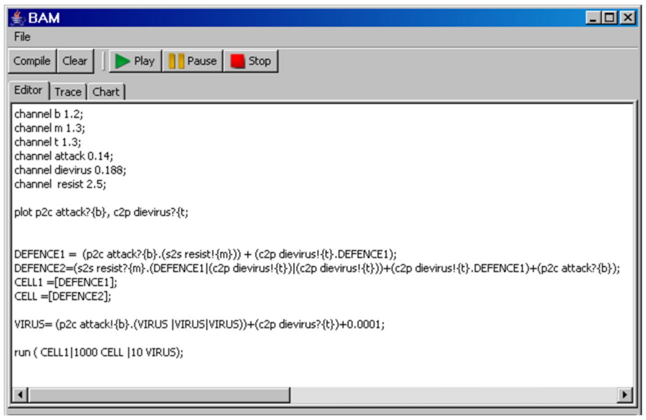
$$
\begin{array}{rcl}
G + \tau_r.P & \xrightarrow{r} & P \\
a\,[(G + \mathbf{enter}!\,n.P) \mid Q] \mid b\,[(G' + \mathbf{enter}?\,n.R) \mid S] & \xrightarrow{\rho(n)} & b\,[a\,[P \mid Q] \mid R \mid S] \\
b\,[a\,[(G + \mathbf{exit}!\,n.P) \mid Q] \mid (G' + \mathbf{exit}?\,n.R) \mid S] & \xrightarrow{\rho(n)} & a\,[P \mid Q] \mid b\,[R \mid S] \\
a\,[(G + \mathbf{merge}!\,n.P) \mid Q] \mid b\,[(G' + \mathbf{merge}?\,n.R) \mid S] & \xrightarrow{\rho(n)} & a\,[P \mid Q \mid R \mid S] \\
a\,[(C + \mathbf{local}\,n?(y).P) \mid (C' + \mathbf{local}\,n!\langle m\rangle.Q) \mid R] & \xrightarrow{\rho(n)} & a\,[P\{m/y\} \mid Q \mid R] \\
a\,[(C + \mathbf{s2s}\,n?(y).P) \mid R] \mid b\,[(C' + \mathbf{s2s}\,n!\langle m\rangle.Q) \mid S] & \xrightarrow{\rho(n)} & a\,[P\{m/y\} \mid R] \mid b\,[Q \mid S] \\
a\,[b\,[(C + \mathbf{p2c}\,n?(y).P) \mid R] \mid (C' + \mathbf{p2c}\,n!\langle m\rangle.Q) \mid S] & \xrightarrow{\rho(n)} & a\,[b\,[P\{m/y\} \mid R] \mid Q \mid S] \\
a\,[b\,[(C + \mathbf{c2p}\,n!\langle m\rangle.Q) \mid S] \mid (C' + \mathbf{c2p}\,n?(y).P) \mid R] & \xrightarrow{\rho(n)} & a\,[b\,[Q \mid S] \mid P\{m/y\} \mid R] \\
P \xrightarrow{r} P' & \Rightarrow & P \mid R \xrightarrow{r} P' \mid R \\
P \xrightarrow{r} P' & \Rightarrow & (\mathsf{new}\,n)\;P \xrightarrow{r} (\mathsf{new}\,n)\;P' \\
P \xrightarrow{r} P' & \Rightarrow & a\,[P] \xrightarrow{r} a\,[P'] \\
P \equiv P' \xrightarrow{r} Q' \equiv Q & \Rightarrow & P \xrightarrow{r} Q
\end{array}
$$

Figure 2. Reduction Relation

represents the time at which an action happens, the name of the action that has been selected and the residual term. The facility of a GUI interface is provided, where an editor (Figure 3) allows the user to easily edit BioAmbient programs and to perform standard I/O facilities, including loading and saving programs stored in files. Compilation is executed using the GUI interface by pressing the 'Compile' button. In the absence of syntactic mistakes, the program is simulated by clicking on the 'Play' button, while the 'Chart' button displays a graph of the simulation. The graph depicts the variation over time of the population of the species chosen to be monitored (Figure 4). Only prefixes ($\mathbf{enter}!\,n$, $\mathbf{local}\,n!\langle m\rangle$ etc.) and ambients can be plotted in the graph. For example in the following process $P = a\,[]|M.a\,[]$ we count only one occurrence of the ambient 'a', since the one occurring after the guard M is not currently being executed and is therefore ignored. Note that the use of ambient names in our syntax is required in order to plot an ambient. The algorithm for counting occurrences in a process is defined inductively on the syntax of BioAmbients. The simulation in BAM continues until no more transitions are possible, or until the user pauses or stops the simulation. A paused simulation can be resumed at a later stage. The simulation graph can be easily manipulated in the GUI interface, e.g. it can be enlarged or reduced for customised viewing of the simulation results.

The main motivation for developing BAM was to produce a BioAmbient simulation tool that is more accessible to users. In contrast to BioSPI [18], BAM features a graphical user interface for entering program code and visualising simulation results. BAM is implemented in Java and requires only the Java runtime to install and run.

The architecture of the tool is composed of the GUI interface that feeds into a parser, which translates the syntax



```
channel b 1.2;
channel m 1.3;
channel t 1.3;
channel attack 0.14;
channel dievirus 0.188;
channel  resist 2.5;

plot p2c attack?{b}, c2p dievirus?{t;


DEFENCE1 =  (p2c attack?{b}.(s2s resist!{m})) + (c2p dievirus!{t}.DEFENCE1);
DEFENCE2=(s2s resist?{m}.(DEFENCE1|(c2p dievirus!{t})|(c2p dievirus!{t}))+(c2p dievirus!{t}.DEFENCE1)+(p2c attack?{b});
CELL1 =[DEFENCE1];
CELL =[DEFENCE2];

VIRUS= (p2c attack!{b}.(VIRUS |VIRUS|VIRUS))+(c2p dievirus?{t})+0.0001;

run ( CELL1|1000 CELL |10 VIRUS);
```

Figure 3. BAM editor

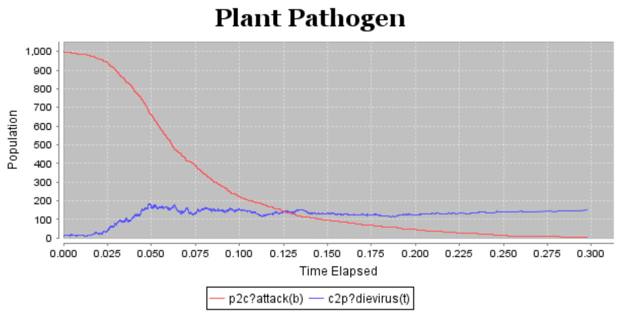


Figure 4. Plant Pathogen 1

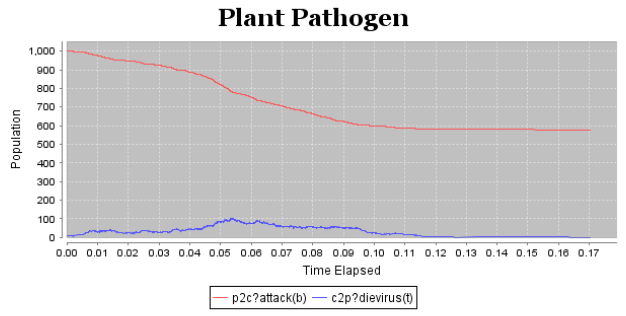


Figure 5. Plant Pathogen 2

of BioAmbients into machine terms. Terms are built as lists and consist of a list of actions $M_1, \ldots, M_N$, together with a list of ambients, where each ambient can contain a nested term, recursively. The reduction relation written in Figure 2 is therefore implemented as operators on these lists. Additional data structures are used to count different types of interactions on different channels, and the generation of new names $(\mathsf{new}\, a)$ is handled using a top-level renaming function applied to the lists. Structural congruence is essentially implemented as a reordering of lists. At each step in the simulation a term is selected to be reduced according to the rules in Figure 2, based on the Gillespie algorithm [9]. This algorithm is performed on a machine term by first choosing an ambient with probability proportional to the total rate of all the reactions inside the ambient, and then choosing a local reaction inside this ambient with probability proportional to the rate of the reaction.

BAM has been devised to simulate large models of biological systems with mobile compartments. This is the main motivation for choosing a simulation algorithm as opposed to other analysis methods such as steady state probability. In our simulations the state space and reactions do not need to be constructed beforehand, allowing us to effectively deal with models that have a large number of states and reactions.

For future work we aim to further improve the user interface by allowing a graphical representation of our code to be input by the user for simulation.

3.1. Example

We provide a simple biological example to show the use of BAM, based on the attack of pathogens in plants [13]. We represent a simple model of the behaviour of a group of cells. In the presence of pathogens each cell has two kinds of responses:(1) the cell tries to kill the pathogen; (2) the cell sends a signal to a neighbouring cell, which is believed to trigger better defences. The model can be readily described in BioAmbients as follows;

$$\begin{array}{rcl}
PTH & = & \mathbf{p2c}\, attck!\langle y\rangle.(PTH \mid PTH \mid PTH) \\
 & & +\mathbf{c2p}\, die?(y) + \tau_\delta \\
DEF1 & = & \mathbf{p2c}\, attck?(b).\mathbf{s2s}\, res!\langle m\rangle + \\
 & & \mathbf{c2p}\, die!\langle t\rangle.DEF1 \\
DEF2 & = & \mathbf{s2s}\, res?(m).(DEF1 \mid \mathbf{c2p}\, die!\langle t\rangle \mid \\
 & & \mathbf{c2p}\, die!\langle t\rangle) + \\
 & & \mathbf{c2p}\, die!\langle t\rangle.DEF1 + \mathbf{p2c}\, attck?(b) \\
CELL1 & = & cell[DEF1] \\
CELL & = & cell[DEF2] \\
SYST & = & CELL1 \mid \underbrace{CELL \mid \ldots \mid CELL}_{1000} \mid \\
 & & \underbrace{PTH \mid \ldots \mid PTH}_{10}
\end{array}$$

The idea behind the model is that when the pathogen attacks the cell, either it gets stronger and multiplies, or it gets attacked by the cell and dies. Alternatively, it can simply die for independent reasons. The cell can either be attacked, in which case it signals other cells, or it can attack the pathogen. If a cell receives a signal from a neighbouring cell then it can increase its attack on the pathogen, otherwise it triggers the ordinary response. In the cell there is a race between attacking, signalling to a neighbouring cell and being attacked. We run simulations with the following parameters $attck = 0.1, die = 0.188, res = 2.5$ with an initial population of 1000 cells and 10 pathogens. After repeated simulation runs we observe that the plant usually survises the pathogen attack, as shown in Figure 5. We can run additional simulations to determine the number of pathogens and the attack strength needed to destroy the plant. We observe that an increased attack rate of 40 percent $attck = 0.14$ is sufficinet to kill the plant, as shown in Figure 4.

3.2. Resources

The tool is available for download from aesop.doc.ic.ac.uk/tools/bam and simply requires Java 1.5. Both the binary files and the source code are available, together with examples and a manual.

4. Acknowledgments

We would like to thank the Aesop Group at Imperial College and Luca Cardelli for useful discussion on the development of BAM. Maria G. Vigliotti was supported by the EPSRC grant Spartacos (EP/D047587/1).

References

[1] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov

chains. *IEEE Transactions on Software Engineering*, 29(6):524–541, 2003.

[2] R. Blossey, L. Cardelli, and A. Phillips. A compositional approach to the stochastic dynamics of gene networks. *Transactions on Computational Systems Biology IV*, 3939:99–122, 2006.

[3] L. Brodo, P. Degano, and C. Priami. A stochastic semantics for bioambients. In *Proceedings PaCT'07*, volume 4671 of *LNCS*. Springer-Verlag, 2007.

[4] M. Calder, S. Gilmore, and J. Hillston. Modelling the influence of RKIP on the ERK signalling pathway using the stochastic process algebra PEPA. In A. Ingolfsdottir and H. R. Nielson, editors, *Proceedings of the BioConcur Workshop on Concurrent Models in Molecular Biology*, London, England, Aug. 2004.

[5] L. Cardelli. Bioware languages. *Monographs in Computer Science: A Tribute to Roger Needham*, pages 59–65, 2004.

[6] L. Cardelli. Abstract machine for system biology. *Transactions on Computational Systems Biology.*, 3737:145–168, 2005.

[7] L. Cardelli and A. Gordon. Mobile ambients. *Theoretical Computer Science*, 240(1):177–213, 2000.

[8] V. Danos and C. Laneve. Formal molecular biology. *Theor. Comput. Sci.*, 325(1):69–110, 2004.

[9] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[10] J. Hillston. *A Compositional Approach to Perfomance Modelling*. PhD thesis, Department of Computer Science, Edinburgh, 1994.

[11] K. W. Kohn and M. I. Aladjem. Circuit diagrams for biological networks. *Molecular Systems Biology*, 2006.

[12] J. N. W. Kurt W. Kohn, Mirit I. Aladjem and Y. Pommier. Molecular interaction maps of bioregulatory networks: A general rubric for systems biology. *Molecular Biology of the Cell*, 17:1–13, 2006.

[13] l Tainz and E. Zeiger. *Plant Physiology*. Sinauer Associated Inc., 2006.

[14] P. Lecca and C. Priami. Cell cycle control in eukaryotes: A biospi model. *Electron. Notes Theor. Comput. Sci.*, 180(3):51–63, 2007.

[15] H. Lodish, A. Berk, P. Matsudaira, C. A. Kaiser, M. Krieger, M. Scott, S. L. Zipursky, and J. Darnell. *Molecular Cell Biology*. W.H. Freeman and Company, 2003.

[16] M. Miaczynska, L. Pelkmans, and M. Zerial. Not just a sink: endosomes in control of signal transduction. *Curr. Opin. Cell.*, 16:400–4006, 2004.

[17] C. Priami. Stochastic π-Calculus. *The Computer Journal*, 38(7):579–589, 1995.

[18] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Y. Shapiro. Bioambients: an abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 2004.

[19] A. Regev and E. Y. Shapiro. Cellular abstractions: Cells as computation. *Nature*, 419(343), 2002.

[20] S. van Bakel, I. Khan, J. K. Heath, and M. Vigliotti. Modelling intracellular fate of FGF receptors with BioAmbients. In *Proc. 6th Int. Workshop Quantitative Aspects of Programming Languages (QAPL 2008)*, 2008.

[21] M. Vigliotti and P. Harrison. Stochastic mobile ambients. In A. D. Pierro and H. Wiklicky, editors, *Proc. 4th Int. Workshop Quantitative Aspects of Programming Languages (QAPL 2006)*, volume 164 (issue 3) of *Electronic Notes in Theoretical Computer Science*, pages 169–186. Elsevier, 2006.