ADAM: Analysis of Discrete Models of Biological Systems Using Computer Algebra

Franziska Hinkelmann[de], Madison Brandon[a*], Bonny Guang[b*], Rustin McNeill[c*], Reinhard Laubenbacher[de]

[a]University of Tennessee - Knoxville, Knoxville, TN 37996-2513, USA, [b]Harvey Mudd College, Claremont, CA 91711-5901, USA, [c]University of North Carolina - Greensboro, Greensboro, NC 27402-6170, USA, [d]Department of Mathematics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0123, USA, [e]Virginia Bioinformatics Institute, Blacksburg, VA 24061-0477, USA . [*]These three authors contributed equally

Abstract

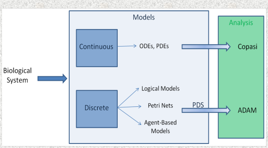
Many biological systems are modeled qualitatively with discrete models. Several different modeling types have established communities in the biological sciences, including probabilistic Boolean networks, logical models, bounded petri-nets, and agent-based models. These and other discrete model types can be translated into algebraic models. Using algebraic models as a representation for discrete models allows one to apply theory from algebraic geometry and tools from computational algebra to analyze the dynamic features of such systems. Simulation has become common practice for analyzing discrete models, but most real world biological systems are far too complex to be analyzed by simulation alone. We use various abstract algebra techniques to analyze discrete models in order to develop algorithms and software to analyze discrete models for key dynamic features of biological relevance. All algorithms and methods are available trough a web-interface <http://dvd.vbi.vt.edu/cgi-bin/git/adam.pl>. Analysis of Discrete Algebraic Models (ADAM) has a 'modeler friendly' interface that allows for fast analysis of large models while requiring no understanding of the underlying mathematics or installing software. By providing a user-friendly interface to fast analysis tools, we promote the use of discrete models to model large complex systems.

Introduction

In biological systems, we are concerned with how different elements in the system interact with one another. One way to describe such interactions is by creating a discrete model – a qualitative description of the system. Discrete models are playing an increasing role in Mathematical Biology.

Continuous Models:
•Rely on exact parameter rates and other possibly hard to obtain information
•Often not intuitive
•Many tools available for analysis

Discrete Models:
•Variables can only take on a finite number of states
•Intuitive
•Few mathematical tools available for analysis

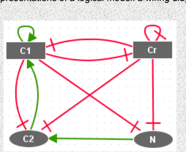


Different types of models used in biology along with examples of corresponding analytic software.

For discrete models, analysis is usually done by simulation which is inefficient and even impossible for some large networks. By translating discrete models into algebraic models, we are able to use algorithms from abstract algebra to analyze the dynamics of networks that are too large for simulation. We implemented several algorithms and made them available through a web-interface called Analysis of Discrete Algebraic Models (ADAM)[2]. Algebraic Models can be inferred from experimental time course data with *Polynome*[10].

Definitions

One type of model we consider is **Logical models**. A Logical model is a set of vertices or nodes connected by edges which have some biological significance. There are two representations of a logical model: a *wiring diagram* and *state space graph*.



Wiring Diagram of lysogenization by Lambda Phage virus. Model from GINsim[5] repository.

Wiring diagram – a graph representing the static relationship between nodes or genes by directed edges.

The wiring diagram represents the interaction of four genes involved in the lysogenic cycle of the Lambda Phage virus.
•red arrows signify inhibition
•green arrows signify activation.

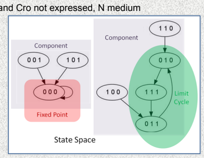
The Lambda Phage virus has been extensively studied as a model organism and is a useful tool in molecular biology.

Polynomial Dynamical Systems (PDS)
• The current state of the system is described by a vector of concentrations.
• Every gene can take on finitely many different states
• Example: Lambda Phage:
  - lowest concentration: 0
  - highest concentration: 4
  - a state is represented as (CI, CII, Cro, N)
  - For example (1,0,0,3) means CI low, CII and Cro not expressed, N medium

State Space – a graph representing the key dynamical features:
•**Fixed Points** – may give the researcher info about what combination of gene states is a steady state.

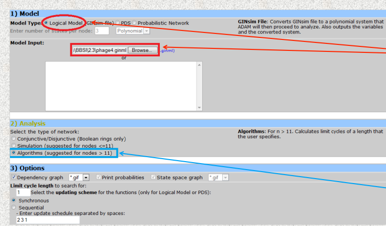
•**Limit Cycles** - limit cycles and their length can indicate recurring processes such as the cell cycle.

•**Components** - typically a modeler expects small limit cycles with large component sizes.



Example **state space** of a model with three genes created with ADAM.

ADAM Web-interface



ADAM Analysis options for networks too large to simulate completely.

Lambda Phage

•You can upload the Lambda Phage Logical Model (from *GINsim*) into ADAM for analysis.

•Besides uploading a logical model, the user has the option of selecting PDS (polynomial dynamical system) or PBN (probabilistic Boolean network) and then either uploading a text file or inputting functions into the provided text box.

Analysis:
•Conjunctive/Disjunctive – for systems in which all functions involve only ANDs / ORs respectively.
Analysis is done via algebra.
•***Algorithms*** – Uses algebra to solve for dynamics.
•***Simulation*** – An initial configuration of the system is iterated until all dynamics are found, i.e., checking every possible state as a solution.

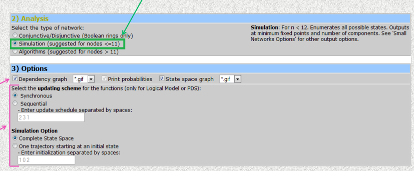
•If ***Algorithms*** is selected, the user must specify a limit cycle length to search for.

•We may choose to simulate the Lambda Phage example since it only involves four nodes.

•If ***Simulation*** is selected more additional options appear.

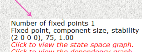
•The user may choose to output the state space graph in addition to the dependency graph.

•The user may also specify other options such as updating schedule or generating single trajectories.

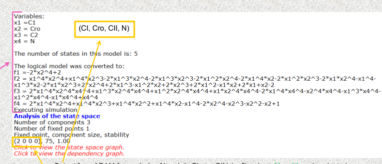


ADAM analysis options.

ADAM's Outputs:
•All fixed points / number of fixed points
•Limit cycles of the specified length when ***Algorithms*** is selected
•All limit cycles when ***Simulation*** is selected
•Number of Components
•Number of states in the model
•PDS for GINsim models with variable descriptions
•Dependency graph
•State space when ***Simulation*** is selected



ADAM generates a graph of the wiring diagram and the state space.



Output from ADAM for analysis of Lambda Phage GINsim file when ***Algorithms*** is selected.

From ADAM's output the lysogenic cycle of Lambda Phage virus has one fixed point at *(2,0,0,0)*. Therefore when CI is at some medium concentration and the three other genes are off, the process is in a fixed or steady state.

TCR Signalization Pathway[11]:

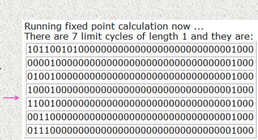
•GINsim Boolean logical model with 40 nodes => 2^{40} = 1,099,511,627,776 states.

•For a state space this large simulation is particularly inefficient. In fact, running the simulation on GINsim takes over an hour.

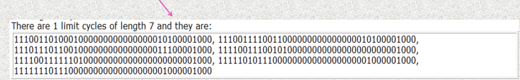
•ADAM computes all 7 steady states in less than half a second.

•The system was analyzed for limit cycles up to length 20 with a total runtime of approximately 63 seconds.

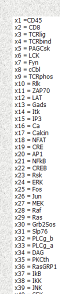
•The results show a limit cycle of length 7, which was not found in the original analysis[3].



Steady states of the TCR Signaling Pathway.



Output from ADAM for limit cycles of length 7 of the TCR Signaling Pathway.



Variables and corresponding genes for TCR model.

Methods

Any discrete model can be written as a polynomial dynamical system (PDS)[6][13]. Hence we can use computational algebra to solve systems of polynomial equations to analyze the model's key dynamic features. Specifically, we simplify the analysis of large polynomial systems of equations by computing the Gröbner basis. The Gröbner basis is useful because its triangular structure allows one to solve the system via back substitution.

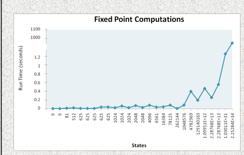
The algorithms were coded in the computer algebra system Macaulay2[9]. Specifically we used Macaulay2 to compute the Gröbner basis of input functions within a quotient ring.

Benchmark tests on Gröbner basis computations. Benchmark tests were run on randomly generated Boolean functions with different ranges of variables, terms per function, and a maximum number of variables per term.

Time(min)	# of tests	Nodes	Nodes per term	# of terms	testing
Under 0.05	44	10-20	10-15	5-9	*High # of nodes per term*
Under 1	5	"	"	"	
Under 7	1	"	"	"	
Under 0.5	8	20-30	10-15	5-9	
Under 0.5	38	10-20	2-6	100-1,000,000	*High # of terms*
Under 6	5	"	"	"	
Under 10	1	"	"	"	
Over 20	3	"	"	"	
Under 0.05	50	10-20	2-6	1	*Monomial (Conjunctive) Networks*
"	10	20-30	"	"	
"	5	30-40	"	"	
Under 0.5	5	40-50	"	"	
Under 1	3	50-100	2-6	1-5	*High # of nodes*
Under 0.5	50	50-100	2	6-10	
Under 0.5	7	40-50	5	1-8	
Under 1	1	"	"	"	
10	1	"	"	"	
Over 10	2	"	"	"	

In biological systems, most nodes only have a few direct neighbors. For instance, in gene regulatory networks, genes are regulated by only a handful of regulators[7]. This means the PDSs representing such biological networks are sparse, i.e., the support of each polynomial is a small set of variables. It has been shown that computing Gröbner bases from sparse polynomials is more efficient in a finite field[4]. Based on benchmark tests on our algorithms which compute fixed points, we show that our algorithms are, in fact, fast on large sparse systems. All benchmark tests were run on a 2.27 GHz processor.

91% of Gröbner bases computations finished in less than 30 seconds.



Fixed point analysis for Logical Models in GINsim repository. Models have between 2 and 75 variables.

•We benchmarked 25 models from the GINsim repository.

•Fixed points and limit cycles up to length 20 were computed for all models.

•85 % of computations finished in less than 1 second.

•Collectively all computations finished in under 30 minutes.

For networks with a special structure, *conjunctive* and *disjunctive* Boolean networks, ADAM uses the algorithm described in Jarrah *et. al.*[1] which does not require a Gröbner basis computation and therefore can analyze larger systems. *Conjunctive* Boolean networks consist of functions containing only one monomial term, i.e. the functions use only the AND operator. Conversely, *disjunctive* Boolean Networks consist of functions which use only the OR operator. There is a closed formula to compute the cycle structure which depends solely on the wiring diagram.

Conclusions

•We can analyze ***discrete models***, such as logical networks, petri-nets, or agent-based models by converting them into ***polynomial dynamical systems (PDS)***.

•Once these models have an algebraic structure, we use tools from ***computational algebra*** to compute key dynamics.

•The algorithms we developed are ***fast*** for sparse systems, a structure maintained by most biological systems.

•All algorithms have been included in the ***software package ADAM***, which is user-friendly and available as a free web service.

•We hope to expand ADAM to an all-encompassing ***Discrete Toolkit*** which incorporates more analytical methods, better visualization, and automatic conversion for more model types.

Acknowledgements

This research was conducted during the Research Experience for Undergraduates (REU), Modeling and Simulation in Systems Biology (MSSB) at Virginia Tech University. This research was funded by the National Science Foundation, Fund Number 0755322.

References

1. Jarrah A., Laubenbacher R., Veliz-Cuba A. (2008) The Dynamics of Conjunctive and Disjunctive Boolean Network Models. Bioinformatics.
2. ADAM Home. <http://dvd.vbi.vt.edu/cgi-bin/git/adam.pl>.
3. Naldi A., Thieffry D., Chaouiya C. (2007) Decision Diagrams for the Representation and Analysis of Logical Models of Genetic Networks. Lecture Notes in Bioinformatics
4. Brown W.S. (1971) On Euclid's Algorithm and the Computation of Polynomial Greatest Common Divisors. Journal of the ACM.
5. GINsim Home. <http://gin.univ-mrs.fr/>.
6. Hinkelmann F. et al. (2010) A Mathematical Framework for Agent Based Models of Complex Biological Networks. Quantitative Biology.
7. Leclerc R. (2008) Survival of the Sparsest: Robust Gene Networks are Parsimonious. Ecology and Evolutionary Biology.
8. Lidl R., Niederreiter H. (1997) Finite Fields, Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge.
9. Macaulay2 Home. Grayson D. and Stillman M. , Macaulay2, a software system for research in algebraic geometry. Available at <http://www.math.uiuc.edu/Macaulay2/>.
10. Polynome. <http://polymath.vbi.vt.edu:3002/>.
11. Klamt S., Saez-Rodriguez J., Lindquist J., Simeoni L., Gilles E. (2006) A methodology for the structural and functional analysis of signaling and regulatory networks. BMC Bioinformatics
12. Chen T., He H., Church G. (1999) Modeling Gene Expressions with Differential Equations. Genetics.
13. Veliz-Cuba A., Salam J., Laubenbacher R. (2010) Polynomial Algebra of Discrete Models in Systems Biology. Bioinformatics.
14. Dimitrova E. et al. (2010) Parameter estimation for Boolean models of biological networks. Theoretical Computer Science.