Applying Fast/Slow Asynchrony and Boolean Minimalism to the Computational Modeling of *C. elegans* Signaling Pathways

Chelsea S. Voss
November 13, 2010

Introduction.

The soil-dwelling nematode *Caenorhabditis elegans* is a model organism frequently studied as an example of animal development. The worm is transparent, easily grows in petri dishes, and has exactly 959 somatic cells in its adult form, making it a useful organism for developmental studies.[2] During *C. elegans'* late development, six precursor cells develop into the cells of the *C. elegans* vulva upon activation of intracellular and intercellular signaling pathways triggered by a signal from an anchor cell within the nematode gonad. Mutations in any of the genes involved in these signaling pathways can produce altered phenotypes which, along with the genetic mechanisms involved in these six cells' development, have been well researched.[1][2][8][9][10]

In the wild-type nematode, the anchor cell produces an inductive signal, causing the precursor cell nearest to the anchor cell to adopt a cell fate denoted as 1°. This 1° precursor cell produces a lateral signal, causing its neighbor precursor cells to adopt a cell fate denoted as 2°. The remaining three precursor cells, receiving neither signal, adopt the 3° cell fate (**figure 1**). The different cell fates yield different patterns of cell division for the lineage of cells descended from each precursor cell: the 1° fate produces a lineage of four cells with the last round of division along the transverse axis; the 2° fate produces a lineage of four cells with the last round of division partially along the transverse and partially along the longitudinal axes; and the 3° fate produces a lineage of two cells within the hypoderm of the nematode.[9] The precise placement of each of these cell fates determines whether the *C. elegans* vulva develops correctly. Mutations in the genes relevant to this development process produce altered patterns of cell fates, resulting in

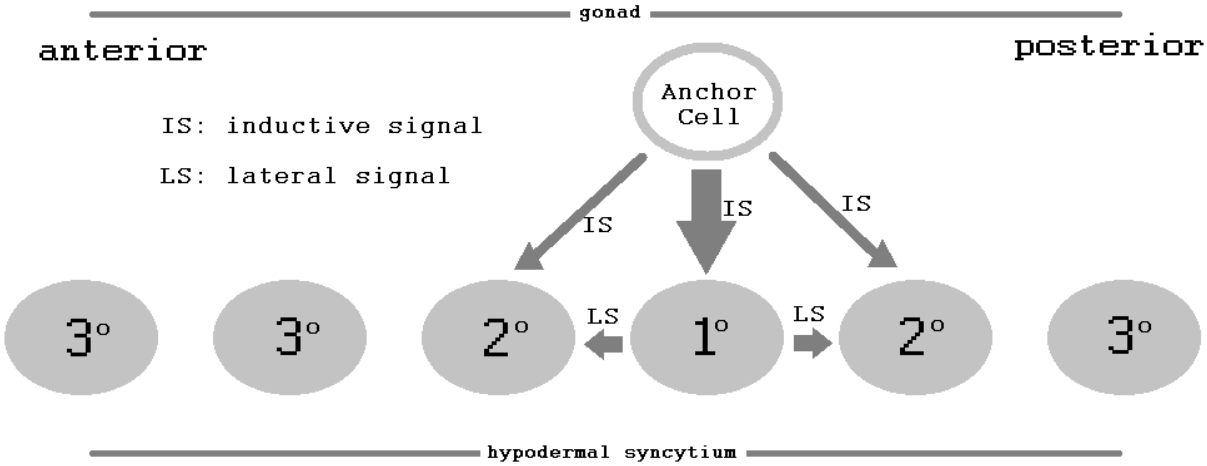

Figure 1. *Basic signaling mechanisms involved in development of the wild-type nematode. The anchor cell within the nematode gonad produces an inductive signal, which varies in strength along a gradient depending on a precursor cell's distance from it. The inductive signal causes the center-most cell to receive the 1° cell fate, and to generate a lateral signal. The surrounding cells receive this signal, which overrides the inductive signal and produces the 2° cell fate.*[6][8]

novel phenotypes.

The tools of systems biology can be used to understand the properties of such a signaling system. Computational models, describing the states and interactions of the components of a system such as the one involved in producing the fate patterns of the *C. elegans* vulva, allow predictions of model interactions to be made *in silico* when constraints prevent such interactions from being observed *in vivo*. As biological knowledge moves forward, computational models that allow such vast quantities of information to be simplified and understood grow more significant.

Of particular importance is the development of models that store a minimal amount of information while still accurately describing the system to be modeled, so that models of complicated systems may be analyzed within a reasonable capacity of computing power. Wherever possible, boolean models, in which the system's components may be either on or off and reactions between components either activate or inhibit, are thus preferable to non-boolean models, which describe a system's components and reactions with varying numerical degrees of

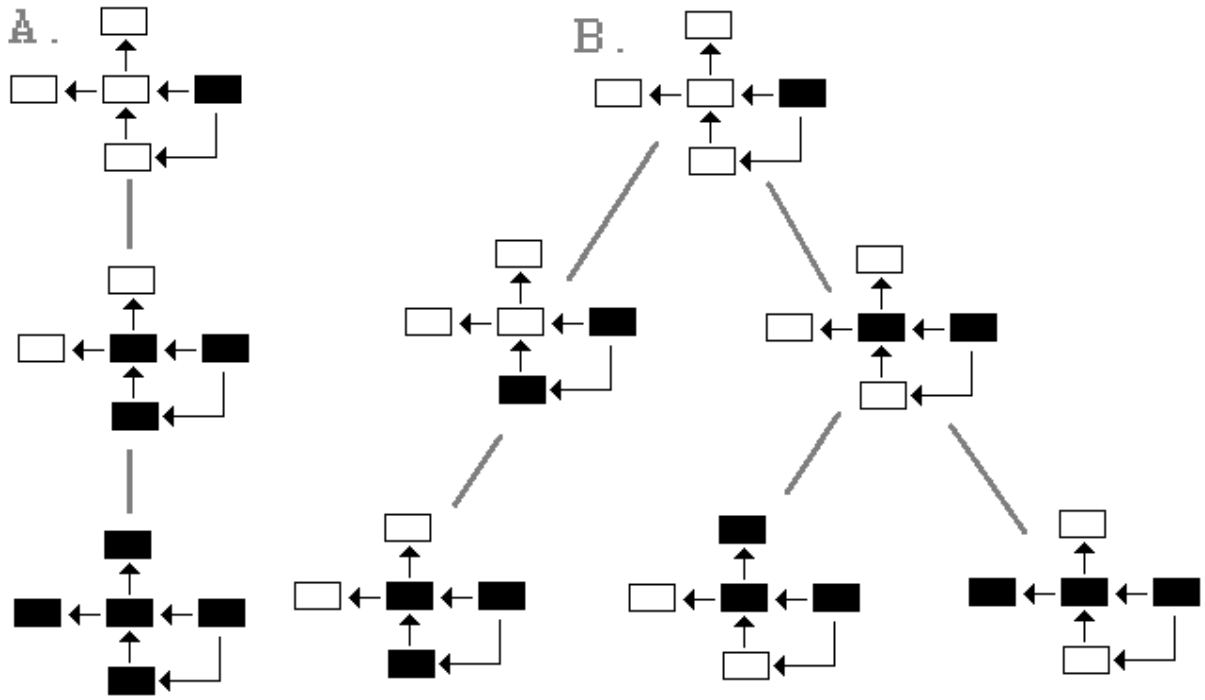

Figure 2: *Comparison of Kripke structures (diagrams of a model's possible outcomes) for asynchrony versus synchrony. a) Kripke structure of a synchronously-updating model. The process is deterministic; it can only lead to one possible outcome, and so the Kripke structure is a straight line. b) Kripke structure of the same model, but with asynchronous updating. Only one reaction is allowed to occur, of the many possible at each step, leading non-deterministically to a branched Kripke structure and multiple possible outcomes. This work will utilize asynchrony.*

strength or speed. Constructing such models is also more reasonable in the absence of concrete quantitative data about the reaction kinetics of all the system's components.[7]

The model constructed in this project makes fewer unnecessary assumptions about timing than other boolean models of *C. elegans* development, improving its simplicity and realism. During the execution of a boolean model, the status of each component in the system is usually updated according to a "synchronous update rule," in which all reactions that have the capacity to occur are updated simultaneously, or according to an "asynchronous update rule," in which one reaction among all possible reactions is chosen arbitrarily to happen[7] (**figure 2**). Synchronous updating results in deterministic outcomes for model executions, producing consistent and easily-analyzed results, though such outcomes may often be unrealistic for

biological systems. Asynchronous updating is useful for modeling systems where little is known about the rates of reactions, or where reaction rates may vary due to stochastic effects, as is often the case with biological systems.[7] Partially asynchronous models are also possible: a previous model of *C. elegans* vulval development[6] utilized the concept of "bounded asynchrony," in which processes are allowed to update independently up to a certain extent, such that no one process significantly outpaces the others.[5] This project's model uses a variation on asynchronous updating, in which reactions are defined as "fast" or "slow." Processes proceed at completely independent rates, in a non-deterministic manner, except that no "slow" reactions may occur until all available "fast" reactions have been allowed to update, proceeding to completion. Describing reactions as "fast" or "slow" allows the rates of processes to freely vary as they do in most biological systems.

The purpose of this project was to determine whether a minimalistic model, utilizing boolean variables and reactions and adding the concept of fast and slow reactions to the asynchronous update rule, could accurately describe the intricate interactions producing the *C. elegans* vulval development pattern, and to further determine whether such a model could be capable of predicting the novel phenotypes resulting from various combinations of mutations in the nematode genome. Extending the results of previous computational studies[6] of *C. elegans* vulval development, this work will determine the viability of a boolean and partially-asynchronous model and will generate new predictions about the mechanisms behind this aspect of development.

Methods and Results.

This computational modeling project began with describing the structure of the wild-type signaling pathways, incorporating sufficient information to describe the state of every signaling component and every reaction between components. Conceptually the model contains 'nodes,'

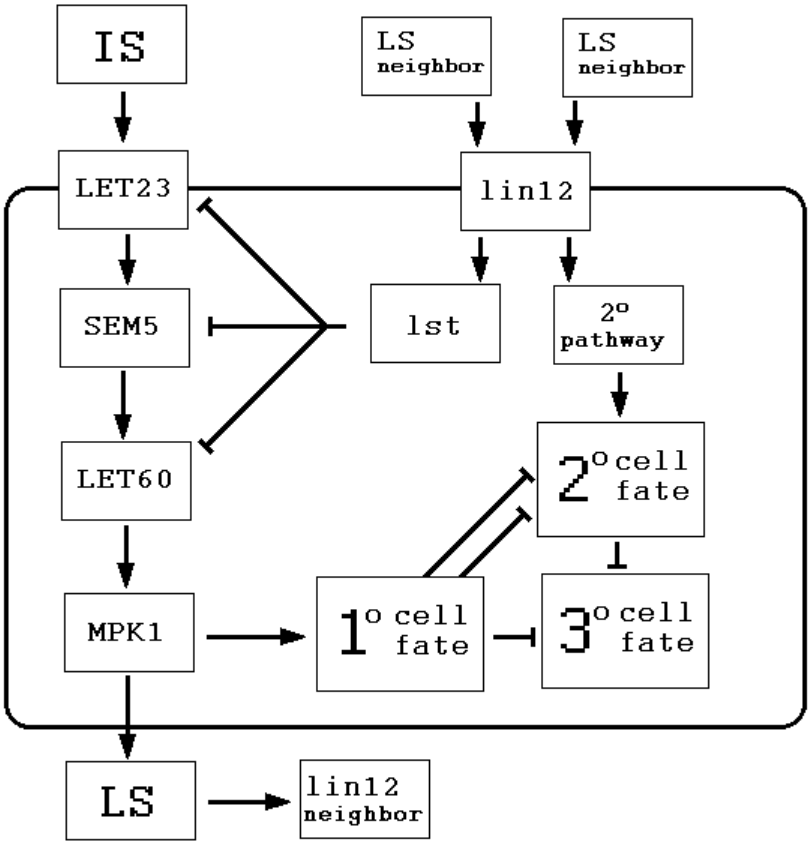

Figure 3. *Nodes and reactions of the wild-type computational model within a single vulval precursor cell. Pointed arrows denote activation, flat arrows denote inhibition.*

variables possessing a value of 1 or 0 and representing molecules, signals and cells, as well as 'reactions,' which represent a node activating or inhibiting another node. Twelve nodes are included per cell to describe the signaling processes relevant to each cell's fate determination (**figure 3**). The arrangement of these nodes and reactions was outlined to reflect known data[2][6][9] about their molecules' interactions. The inductive signal, node *IS*, activates node *let23*, an epidermal growth factor receptor, which activates a signal transduction pathway that leads to the 1° cell fate and the production of the lateral signal, *LS*, which may be received by neighboring cells. Reception of a neighboring cell's *LS* by another receptor, *lin12*, activates inhibitor molecules encoded by *'lst'*, the lateral signal targets, which inhibit the action of the 1° pathway, and activates a second mechanism which leads to the 2° cell fate. For the purpose of observing

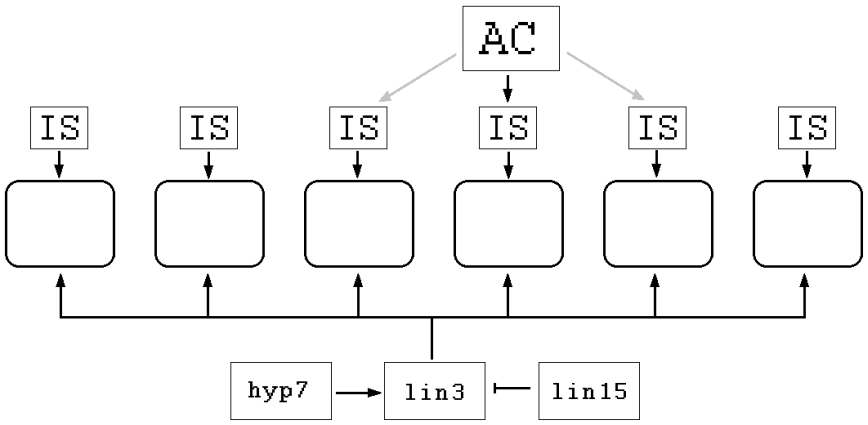

Figure 4. *Other nodes and reactions of the entire wild-type model, across all six precursor cells. Six times as many precursor cell nodes and reactions, from figure 3, are required to form the full model. The two gray arrows denote slow reactions; all others are fast.*

each cell's computed fate, three nodes are included in this model to represent the three individual cell fates. The interactions between these nodes are designed to ensure that each of the six precursor cells' fates are properly predicted, depending upon the nodes activated within the cell. The default cell fate is 3°, which may be inhibited by the activation of either cell fate 1° or cell fate 2°. Cell fate 1° doubly inhibits 2°, so that the cell fate received first will remain activated; cell fate 2° indirectly inhibits 1° in return, through the action of the *lst* inhibitors.

Four other nodes, related to the other signals produced by the anchor cell and the hypodermal syncytium, are also included in this model (**figure 4**). The hypodermal syncytium, node *hyp7*, secretes an epidermal growth factor (identical to the inductive signal produced by the anchor cell), encoded by *lin-3*, which activates all six precursor cells unless inhibited by *lin-15*, the product of the synthetic multivulva genes.[6] The anchor cell, node *AC*, only produces an inductive signal for the third, fourth, and fifth cells. Biologically, the fourth precursor cell is nearest in proximity to the anchor cell; therefore the inductive signal acts along a gradient, reaching the central cell first, the third and fifth cells with a significant time delay, and the remaining three cells not at all.

This gradient is modeled by applying the fast-slow asynchrony concept, defining reactions as either “fast” or “slow.” The activation of *IS* at the fourth cell is an interaction represented as “fast,” whereas the signal's later arrival at the third and fifth cells is indicated as a “slow” process. The activations of *IS* at these two cells are the only “slow” reactions within the entire model; all other reactions are by default “fast,” meaning they have equal probability of selection by the asynchronous update rule and take precedence over the two “slow” reactions, which are these two inductive signals. “Slow” reactions will only proceed after all possible “fast” reactions have been allowed to occur. This serves to simulate the slowness with which the inductive signal arrives at these cells compared to its immediate effects upon the fourth cell, while maintaining a mostly asynchronous model.

Possible values for nodes are 0 and 1: 0 denoting “off” or “false” and 1 denoting “on” or “true.” All nodes start with a value of 0 except for *AC*, *hyp7*, *lin15*, and *cell-fate*-3^{o} in each cell, which have the value 1 at the beginning of the model's execution. Thus at the beginning of the model's execution, the anchor cell is present, the production of a second inductive signal by the hypodermal syncytium is blocked by *lin15*, and *cell-fate*-3^{o} is true by default until further signals arrive. Reactions, both “inhibition” and “activation,” may increase or decrease the value of a node to 1 or 0 depending upon the states of the nodes acting upon it. The sum of the values of activating reactions, minus the sum of the values of inhibiting reactions, is used to calculate whether a node can update its value: where x_a is the value (1 or 0) of any activating node and x_i is the value (1 or 0) of any inhibiting node, a node decreases in value if $\sum x_a - \sum x_i$ is negative, and increases in value if $\sum x_a - \sum x_i$ is positive. Note that if the value of a node is 0, it cannot have any activating influence upon other nodes; likewise with inhibition.

The model was analyzed using NuSMV, a symbolic model checker capable of analyzing

systems for certain properties.[3] With a description of the system's structure and a CTL specification formula describing the logic of what to check for, NuSMV searches through the entire set of possible states of the system and outputs whether or not it has the characteristics outlined in the CTL property. In this case, NuSMV was used to check that the cell fate pattern for the six vulval precursor cells had the pattern 3^o-3^o-2^o-1^o-2^o-3^o.

Since the complete wild-type model contains 76 nodes and 123 reactions, it was first described in a more manageable text file format before being converted into the format of NuSMV. For each node, data detailed in the file included the node's name, its initial value (0 or 1), and its speed (fast or slow); each reaction included the name of the node causing the reaction, the name of the node being reacted upon, and whether the interaction was activation or inhibition. A program of 172 lines was written in Java for the purpose of generating SMV files based on these simpler text descriptions.

The first few lines of the wild-type model are included [appendix 1], in its text format prior to conversion to SMV. The first line of the file is a CTL formula, defining the logical statement which NuSMV will check to determine its truth for the Kripke structure of all possible model states.[3][4] Since the model is non-deterministic, a consequence of utilizing the asynchronous update rule, the Kripke structure has a branched and possibly infinite topology (**figure 2**). The CTL formula in the top line of this file is written such that it will be true if and only if the Kripke structure has the property that for all possible pathways the model can take, the precursor cells eventually assume the wild-type fate pattern 3^o-3^o-2^o-1^o-2^o-3^o.

As a sample of the outputs produced by the Java program, the SMV file corresponding to the wild-type model is included [appendix 2]. The language of the SMV file describes all aspects of the model: it outlines the possible values of each node, sums the activations and inhibitions

influencing the value of each node, defines which nodes may update under which conditions (to ensure the definition of “fast” and “slow” is met), and sets the rules by which nodes' value increases or decreases. In addition to the nodes of the model, a variable called *'run'* was included to create asynchronous updating: a node will only update if *'run'* points to it, and *'run'* may point to any one in the set of all the model's reactions that is capable of occurring at that moment, allowing processes to occur non-deterministically as the next node to update is selected arbitrarily. This tactic for modeling asynchronous updating had been used in asynchronous models of other processes.[7]

Execution of the wild-type model with the appropriate CTL formula confirmed that the model produced the expected 3^o-3^o-2^o-1^o-2^o-3^o pattern of cell fates. The next step was to test the boolean model for a variety of mutations, and confirm that the model predicted similar results to those observed *in vivo*. Six different mutations were simulated, with a total of 48 possible mutation combinations. These mutations were removal of the anchor cell (node *AC*), deletion of the inductive signal receptor (node *let23*), deletion or constitutive activation of the lateral signal receptor (node *lin12*), deletion of the lateral signal targets (node *lst*), and deletion of the synthetic multivulva genes (node *lin15*). A second Java program, with 117 lines, was written to make the requisite changes to the wild-type model's topology in order to generate the text file model descriptions describing each possible combination of mutations, which were then converted to SMV files using the first Java program. Deletions were accomplished by removing all activating reactions to the deleted node, and by setting the node's initial value to 0; constitutive activation of *lin12* was accomplished by setting its initial value to 1. The resulting text-files were then entered into the first Java program to generate a SMV description of each mutated model.

The result for each mutation was determined first by testing with a CTL formula

Mutations (AC+)				Predicted (*in silico*)						Actual (*in vivo*)						Mutations (AC–)				Predicted (*in silico*)						Actual (*in vivo*)					
lin12	lin15	let23	lst													lin12	lin15	let23	lst										
+	+	+	+	3	3	2	1	2	3	3	3	2	1	2	3	+	+	+	+	3	3	3	3	3	3	3	3	3	3	3	3
+	−	+	+	½	½	½	½	½	½	½	½	2	1	2	½	+	−	+	+	½	½	½	½	½	½	½	½	½	½	½	½
+	−	+	−	1	1	1	1	1	1	1	1	1	1	1	1	+	−	+	−	1	1	1	1	1	1						
+	−	−	+	3	3	3	3	3	3	3	3	3	3	3	3	+	−	−	+	3	3	3	3	3	3						
+	−	−	−	3	3	3	3	3	3							+	−	−	−	3	3	3	3	3	3						
+	+	+	−	3	2	1	1	2	3	3	3	1	1	1	3	+	+	+	−	3	3	3	3	3	3	3	3	3	3	3	3
+	+	−	+	3	3	3	3	3	3	3	3	3	3	3	3	+	+	−	+	3	3	3	3	3	3						
+	+	−	−	3	3	3	3	3	3							+	+	−	−	3	3	3	3	3	3						
−	+	+	+	3	3	1	1	1	3	3	3	1	1	1	3	−	+	+	+	3	3	3	3	3	3	3	3	3	3	3	3
−	−	+	+	1	1	1	1	1	1	1	1	1	1	1	1	−	−	+	+	1	1	1	1	1	1	1	1	1	1	1	1
−	−	+	−	1	1	1	1	1	1							−	−	+	−	1	1	1	1	1	1	1	1	1	1	1	1
−	−	−	+	3	3	3	3	3	3							−	−	−	+	3	3	3	3	3	3	3	3	3	3	3	3
−	−	−	−	3	3	3	3	3	3							−	−	−	−	3	3	3	3	3	3	3	3	3	3	3	3
−	+	+	−	3	3	1	1	1	3	3	3	1	1	1	3	−	+	+	−	3	3	3	3	3	3						
−	+	−	+	3	3	3	3	3	3	3	3	3	3	3	3	−	+	−	+	3	3	3	3	3	3						
−	+	−	−	3	3	3	3	3	3							−	+	−	−	3	3	3	3	3	3						
d	+	+	+	2	2	2	½	2	2	2	2	2	1	2	2	d	+	+	+	2	2	2	2	2	2	2	2	2	2	2	2
d	−	+	+	½	½	½	½	½	½	½	½	2	1	2	½	d	−	+	+	½	½	½	½	½	½	½	½	½	½	½	½
d	−	+	−	1	1	1	1	1	1							d	−	+	−	1	1	1	1	1	1						
d	−	−	+	2	2	2	2	2	2							d	−	−	+	2	2	2	2	2	2						
d	−	−	−	2	2	2	2	2	2							d	−	−	−	2	2	2	2	2	2						
d	+	+	−	2	2	1	1	1	2							d	+	+	−	2	2	2	2	2	2	2	2	2	2	2	2
d	+	−	+	2	2	2	2	2	2	2	2	2	2	2	2	d	+	−	+	2	2	2	2	2	2	2	2	2	2	2	2
d	+	−	−	2	2	2	2	2	2							d	+	−	−	2	2	2	2	2	2						

Figure 5: *Results of the test of each combination of mutations. "½" indicates that "1 or 2" are both possible cell fates. "d" indicates a component is constitutively activated. The cases shaded in gray indicate predicted outcomes which disagreed with published observations.*

describing the predicted outcomes, based on existing studies of *C. elegans* vulval development. The output of NuSMV is 'true' or 'false' with a counterexample, based on the truth or falsity of the CTL formula. Each combination of mutation was tested to find the final pattern of cell fates that matched it, such that the CTL formula output from NuSMV was 'true.' In certain cases, cells

could take either a 1 or 2 fate, which was an allowable possibility with this non-deterministic, asynchronous model. The model's predictions are summarized in **figure 5**, and compared with the *in vivo* observations documented in the literature.[1][8][9][10] Except for the four cases highlighted in gray, these results consistently matched up with the *in vivo* results observed in the literature, as well as the predictions of an earlier model constructed with separate methods.[6]

Of the four discrepancies, three are in cases where precursor cells may non-deterministically receive either fate 1° or fate 2°; these are highlighted in light gray. For each of these, the computational model predicts more flexible phenotypes than have been observed in actual mutant nematodes. A previous model of *C. elegans* development based on "bounded asynchrony" correctly predicted these mutations' *in vivo* phenotypes,[6] differing from the results of this project. This difference may be due to the nature of the fast-slow asynchronous update rule, which considers all possible sequences of reaction times, even those that may occur rarely; the other concept of "bounded asynchrony" disallows improbable reaction timings. *In vivo*, only the cell fate patterns ½½212½, 222122, and ½½212½ have thus far been observed as potential phenotypes for each of these mutation combinations[9] (the mutations are deleted-*lin15*, constitutively-activated-*lin12*, and both of these, respectively). According to this project's fast-slow asynchronous model, the predicted cell fate patterns ½½½½½½, 222½22, and ½½½½½½ may be possible (if improbable) outcomes. For example, while 222122 may be the only phenotype observed *in vivo* for constitutive activation of *lin12*, the computational model predicts that rare circumstances may produce a mutant worm with the fate pattern 222222. Random chance may have prevented these possibilities from being observed *in vivo*, as most of the time these mutations' development would follow the more probable path.

The other discrepancy in the model's predictions occurs for a deletion in *lst*, the lateral

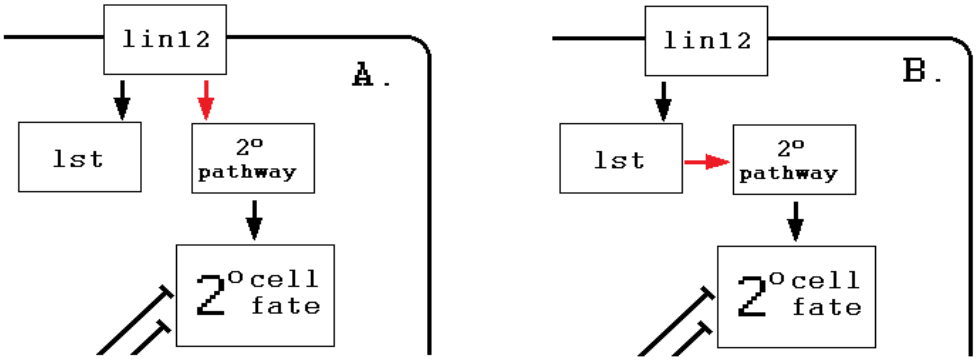

Figure 6. *a) Original structure of model for reception of lateral signal. b) Modification made to structure of model in order correct discrepancy after testing the outcome for a deletion in lst. These results predict a different tactic for modeling the pathway by which the lateral signal produces the 2º cell fate.*

signal target pathway. The expected outcome based on experiments done *in vivo* is 331113;[1][10] the cell fate pattern predicted by this model's results is 321112. Analyzing the computational model suggested this discrepancy could be due to an error whereby the model's structure did not properly reflect the molecular interactions within a live nematode. In the original model, the lateral signal activates both *lst* and a separate, independent pathway leading to the 2º cell fate; as a consequence, when a mutant possesses a deletion in *lst*, the 2º pathway is left intact, so two cells may still assume the 2º fate, yielding the 321112 pattern predicted by the computational model. However, in an actual nematode possessing this mutation, the precursor cell fate pattern is 331113:[9][10] the two neighboring cells were unable to assume the 2º cell fate. This discrepancy between *in silico* and *in vivo* results suggests that the *lst* inhibitor proteins and the 2º pathway were not independent, but that they were linked in some way: specifically, since a deletion in *lst* prevented the 2º cell fate, the 2º pathway could be more accurately modeled as activated indirectly by *lst* rather than directly by the lateral signal receptor *lin12*. The updated version of this model, based on this inference, is shown in **figure 6**. When NuSMV was executed once more using this updated model description, the resulting prediction exactly matched the 331113 cell fate pattern observed *in vivo* in mutated nematodes.

Discussion.

This work successfully demonstrated that even a minimalistic boolean model with partially asynchronous timing could accurately predict the cellular signaling interactions governing *C. elegans* vulval development, in both wild-type and mutant nematodes. The results of the wild-type model, yielding the expected 3^o-3^o-2^o-1^o-2^o-3^o cell fate pattern outcome, confirmed that a system representing molecular values as boolean and lacking detailed timing information but for a notion of "fast" or "slow" was capable of depicting the signaling pathways involved in this process.

Results from testing various combinations of mutations in the genes controlling *C. elegans* vulval development further demonstrated the boolean, asynchronous model's viability, while suggesting other hypotheses about these cellular signaling pathways. Subsequent execution of all 48 possible combinations of mutations in the model using NuSMV accurately predicted the mutant phenotypes of 44 combinations. Analysis of a discrepancy in the mutation where *lst* is deleted suggested a modification to the original model's concept and structure, showing a different pathway for the action of the lateral signal and removing the discrepancy. In addition to amending current knowledge about the mechanisms of this signaling pathway, correction of this detail of the computational model's structure improved its accuracy to 45 out of 48. The remaining three discrepancies between the model's predictions and published observations suggested a greater degree of flexibility in these three mutations' phenotypes. The predicted phenotypes may not have yet been observed in studies of mutant worms, as a result of the rarity of the random timing variations that would lead to these unobserved fate patterns. Possible future work would involve taking this model's hypotheses and testing them with *in vivo* observations of mutant nematode worms, to see if the occurrence of these improbable phenotypes can be verified.

The computational model of *C. elegans* vulval development created in this project successfully modeled the complex cellular signaling mechanisms behind vulval precursor cell fate determination patterns, while maintaining partial asynchrony (except for two “slow” reactions) and boolean simplicity. Simplifying these signaling pathways to a handful of true-or-false values and minimally-synchronized reactions allowed an accurate model to be made, despite containing sparse information about the strengths of signals or the rates of reactions. The merits of creating such a model apply to both its construction and analysis. During biological research, computational models can be made when only a few details have been studied beforehand about *in vivo* reaction rates when it is assumed that they may behave sporadically or at random, as in an asynchronous model, or about the effects of varying concentrations of signal molecules when they are assumed to act on an all-or-nothing basis. During computational analysis of the model itself, while the branching Kripke structure of an asynchronous model's possible values may be time-consuming to interpret, the simplicity of nodes' values assists computation, since calculating whether or not reactions occur is made that much more simple.

Where accuracy can be maintained, this simplicity is an important feature of computational models. Computational biology is expanding to more and more complex systems as biological knowledge grows; thus, it is important that methods be developed for modeling vast and intricate systems, using the absolute minimum in complexity and data requirement for the system to still be modeled accurately. Boolean models require only a single bit to store the state of each molecule included in the model, allowing more complex biological systems to be modeled. The simplicity provided by a boolean model system expedites the process of analyzing the model's outcomes, a factor which becomes more important as the systems to be modeled increases in complexity.

Asynchronously updating models, as opposed to synchronous models or models with detailed description and simulation of each reaction's speed, thoroughly account for the randomness inherent in biological systems while requiring little knowledge about specific reaction times. While the non-determinism of an asynchronous model, due to the many possible model states that could be selected to occur at any stage, may be more of a challenge to interpret computationally, asynchrony allows broader flexibility in a model's outcomes than would synchrony, as demonstrated in this project's results. Appending the asynchronous update rule by defining reactions as “fast” or “slow” adds more realism to the model, while retaining the simplicity of that which must be known to develop the model's timing system. This work demonstrates the viability of using both boolean variables and fast-slow asynchrony as methods for constructing computational models, which can contribute to simpler and more realistic modeling of biological systems in the future.

Developing simplified models of complex biological systems will ultimately allow more knowledge to be discovered about these systems. Perhaps someday entire organisms may be modeled by computer, allowing the mechanisms of their development and physiology to be observed. Indeed, when the exact lineage and developmental ancestry of every one of *C. elegans*' 959 cells has been traced out, such a possibility might not be too far off. As biological studies increase understanding of life's processes, especially those of model organisms such as *Drosophila melanogaster*, *Arabidopsis thaliana*, and *Caenorhabditis elegans*, there is a growing repository of detailed information that could be used to model the genetics, development, and behavior of full organisms. Accurate biological data and efficient computational methods would both be critical to the construction of such ambitious models.

For *C. elegans*, a model organism of animal development, modeling work also yields

results that can be extended to the study of other organisms' development. Uncovering the details of its development leads directly to further understanding of the mechanisms underlying the development of all animals, including humans. Though an invertebrate, the nematode shares many molecular features with vertebrates: insulin as a signaling molecule, homeobox-containing homeotic genes, and programmed cell death,[2] to name a few. Even the epidermal growth factor family of proteins and their receptors, including the EGF molecule produced as an inductive signal from the anchor cell in the very signaling pathway explored by this project, are conserved evolutionarily in both nematodes and humans. Advances in the study of *C. elegans* development are therefore broadly applicable to the study of the development of humans and other animals. Improved computational modeling techniques, such as those tested in this project, have the capacity to facilitate such studies, ultimately leading to more complete knowledge of the molecular biology of human development.

References.

[1]. Berset T., Hoier E.F., Battu G., Canevascini S., Hajnal A. (2001). Notch Inhibition of RAS Signaling Through MAP Kinase Phosphatase LIP-1 During C. elegans Vulval Development. Science 291, 1055-1058. DOI: 10.1126/science.1055642

[2]. Campbell N.A., Reece J.B. (2004). *"Biology,"* 7th edition. Benjamin Cummings Publishing: San Francisco. pp. 412-427.

[3]. Cavada R., Cimatti A., Jochim C.A., Keighren G., Olivetti E., Pistore M., Roveri M., Tchaltsev A. (2010). "NuSMV 2.5 User Manual." Fondazione Bruno Kessler. Web. Accessed 12 Nov. 2010. <http://nusmv.fbk.eu/>.

[4]. Clarke E.M., Emerson E.A., Sistla A.P. (1986). Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. ACM Transactions on Programming Languages and Systems, vol. 8 no. 2, 244-263.

[5]. Fisher J., Henzinger T.A., Mateescu M., Piterman N. (2008). Bounded Asynchrony: Concurrency for Modeling Cell-Cell Interactions. LNBI 5054, 17-32.

[6]. Fisher J., Piterman N., Hajnal A., Henzinger T.A. (2007). Predictive Modeling of Signaling Crosstalk during C. elegans Vulval Development. PLoS Comput Biol 3(5): e92. doi:10.1371/journal.pcbi.0030092.

[7]. Mangla K., Dill D.L., Horowitz M.A. (2010). Timing Robustness in the Budding and Fission Yeast Cell Cycles. PLoS ONE 5(2): e8906. doi:10.1371/journal.pone.0008906

[8]. Sharma-Kishore R., White J.G., Southgate E., Podbilewicz B. (1999). Formation of the vulva in Caenorhabditis elegans: a paradigm for organogenesis. Development 126, 691-699.

[9]. Sternberg P.W., Horvitz H.R. (1989). The Combined Action of Two Intercellular Pathways Specifies Three Cell Fates during Vulva1 Induction in C. elegans. Cell 58, 679-693.

[10]. Yoo A.S., Bais C., Greenwald I. (2004). Crosstalk Between the EGFR and LIN-12/Notch Pathways in C. elegans Vulval Development. Science 303, 663-665. DOI: 10.1126/science.1091639

Appendix 1. *First few lines of the wild-type model, in its text format before conversion to SMV. The full version was too long to include here, at 228 lines. The CTL formula in the very first line is typical of the logical specifications used to test the mutated model structures later on.*

```
AF((cellfate3_a = 1) & (cellfate2_a = 0) & (cellfate1_a = 0) & (cellfate3_b =
1) & (cellfate2_b = 0) & (cellfate1_b = 0) & (cellfate2_c = 1) & (cellfate1_c
= 0) & (cellfate3_c = 0) & (cellfate1_d = 1) & (cellfate2_d = 0) &
(cellfate3_d = 0) & (cellfate2_e = 1) & (cellfate1_e = 0) & (cellfate3_e = 0)
& (cellfate3_f = 1) & (cellfate2_f = 0) & (cellfate1_f = 0))
76
AC 1 f
hyp7 1 f
lin3 0 f
lin15 1 f
IS_a 0 f
LET23_a 0 f
SEM5_a 0 f
LET60_a 0 f
MPK1_a 0 f
lst_a 0 f
LS_a 0 f
cellfate1_a 0 f
cellfate2_a 0 f
cellfate3_a 1 f
lin12_a 0 f
pathway2_a 0 f
IS_b 0 f
LET23_b 0 f
SEM5_b 0 f
LET60_b 0 f
MPK1_b 0 f
lst_b 0 f
LS_b 0 f
cellfate1_b 0 f
cellfate2_b 0 f
cellfate3_b 1 f
lin12_b 0 f
pathway2_b 0 f
IS_c 0 s
LET23_c 0 f
SEM5_c 0 f
LET60_c 0 f
MPK1_c 0 f
lst_c 0 f
LS_c 0 f
cellfate1_c 0 f
cellfate2_c 0 f
cellfate3_c 1 f
lin12_c 0 f
pathway2_c 0 f
IS_d 0
LET23_d 0 f
SEM5_d 0 f
LET60_d 0 f
MPK1_d 0 f
lst_d 0 f
```

***Appendix 2**. Abridged wild-type model written for NuSMV execution. The full version was too long to include here, numbering 825 lines. Excised portions are indicated with italicized notes.*

```
MODULE main
VAR
run : {run_choice, run_AC, run_hyp7, run_lin3, run_lin15, run_IS_a,
run_LET23_a, [...all possible nodes...] run_lin12_f, run_pathway2_f};
AC: {0,1};
hyp7: {0,1};
lin3: {0,1};
lin15: {0,1};
IS_a: {0,1};
[...all nodes, possible values 0 or 1...]
cellfate3_f: {0,1};
lin12_f: {0,1};
pathway2_f: {0,1};
DEFINE
AC_inc := 0;
hyp7_inc := 0;
lin3_inc := 0 + hyp7 - lin15;
lin15_inc := 0;
IS_a_inc := 0;
LET23_a_inc := 0 + IS_a + lin3 - lst_a;
[...formulas for all nodes' activators and inhibitors...]
cellfate3_f_inc := 0 - cellfate2_f - cellfate1_f;
lin12_f_inc := 0 + LS_e;
pathway2_f_inc := 0 + lin12_f;
ASSIGN
init(run) := run_choice;
next(run) := case
      ( ( (AC_inc > 0) & (AC = 0) ) | ( (hyp7_inc > 0) & (hyp7 = 0) ) |
( (lin3_inc > 0) & (lin3 = 0) ) | ( (lin15_inc > 0) & (lin15 = 0) ) |
( (IS_a_inc > 0) & (IS_a = 0) ) | [...includes all 'fast' nodes, excludes
IS_c and IS_e...] | ( (lin12_f_inc > 0) & (lin12_f = 0) ) | ( (pathway2_f_inc
> 0) & (pathway2_f = 0) ) | (0=1)) :{run_AC, run_hyp7, run_lin3, [...set of
all 'fast' nodes...] run_lin12_f, run_pathway2_f};
      1 :{run_AC, run_hyp7, run_lin3, run_lin15, run_IS_a, run_LET23_a,
[...set of all nodes, including two 'slow'...] run_cellfate3_f, run_lin12_f,
run_pathway2_f};
  esac;
init(AC) := 1;
next(AC) := case
      (run = run_AC) & ((AC_inc > 0) & (AC = 0)) : 1;
      (run = run_AC) & ((AC_inc < 0) & (AC = 1)) : 0;
      1 :AC;
  esac;
[...similar statements for all nodes...]
init(pathway2_f) := 0;
next(pathway2_f) := case
      (run = run_pathway2_f) & ((pathway2_f_inc > 0) & (pathway2_f = 0)) : 1;
      (run = run_pathway2_f) & ((pathway2_f_inc < 0) & (pathway2_f = 1)) : 0;
      1 :pathway2_f;
  esac;
FAIRNESS run = run_AC;
[...similar 'fairness' statements for all nodes...]
FAIRNESS run = run_pathway2_f;
```