SENTENCE PARSING E.COLI

2009

Our Team

Our team inaugurates Sweden into IGEM for the first time. We are a bunch of students from Chalmers University of technology and Gothenburg Univesity.

Team: Sweden

The Abstract

Language is an essential part of our civilization. But making sense out of a series of words can only be achieved by certain rules that underlie the language. This set of rules is called a grammar. A grammar tells us how to order words in a meaningful way. These rules can be implemented as a Finite State Automaton (FSA), which for every new word input moves from the current state to the next until it reaches the end of input. We propose in our project a biological model which is based on this concept of language parsing from computational linguistics.

Our goal is to create an finite state automaton in the cell which parses sentences according to their parts-of-speech. We are currently working on the mathematical modeling of the parsing automation.

The Project Aim

We want to implement an automaton in the E-Coli cell. It is based on a few simple grammer rules on how to parse a simple sentence like **The little girl plays ball** or **Boys stroke the little dog**

These simple rules are :

- S --> NP VP
- NP --> (det) (adj) N
- VP --> V (NP)

We target only the parts of speech (POS) tags which in these above grammars are:

- NP:Noun Phrase VP: Verb Phrase
- (det:Determinent) (adj:adjective) N: Noun V: Verb

This way the grammar can be implemented as a finite state automaton (FSA (Fig 1)

The sentence in our project is taken as a string of different reagents which will be introduced to the cell one by one. As soon as a wrong input is detected the cell will light up red. A sentence is finished by a stop reagent and then the cell

Acknowledgements

1. Per Sunnerhagen

2. Sven Nelander

3. Olle Nerman

4. Carl Johan Franzén

UNIVERSITY OF GOTHENBURG

CHALMERS

iMPACT

Mathematical Modeling

Several problems needed to be taken care of when implementing a biological model.

- Repeated input (e.g. det det adj)
- Wrong input (e.g. det adj det)

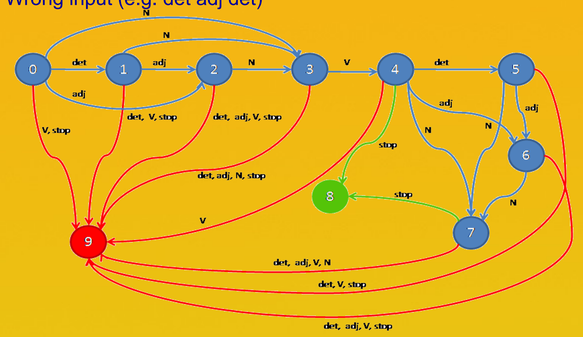

Fig 1. A Finite state Automaton designed with our chosen grammar

How the model works described in words

For **Repeated Input Challenge,** the same PoS we used a counter (with courtesy of ETH iGEM 2005) which counts how many times an input occurs in a row. It uses an interval state and an intermediate input to move the automaton from one state to the other.

For the **Wrong Input,** we set inhibitions in the model in a specific way. so, the interval states (I1-I5 in the figure 2) only get activated by the interval input **X0** and they inhibit all the other interval states. Interval states are also inhibiting some of the S-states, depending on what can be the next correct input.

As for the errors, they are inhibited like the S-states from the I-states just in reverse.

Figure 2 shows half of the model. We apologize for that, but this model has already 17 states and 6 inputs (the corresponding FSA stops at state 4, according to Figure 1). So this figure shows 4 states and the stop.

A complete example

Let's go through the whole model with a **correct** sentence.

English: The dog bites. Model: det N V stop

The model has always state I1 active in the beginning. So, we start with input X1 = det. This activates S1. Next the interval input X0, which moves the system to S2. Input N = X3, activates S3 from I2. Why? The grammar tells us that this is possible and we don't have an inhibition from I2 to S2 (input X2 = adj) or to S3 (input X3 = N). So, we have S3 active and we have to input the interval input X0. The system moves to I4. Next input is then V = X4. S4 is activated. And again the interval input X0. The system is moved to I5. To end the sentence we input the stop signal X5. The system activates S5 which leads to a correct sentence signal.

Now we will go through an **incorrect** sentence.

English: Dog bites hurt. Model: N N V stop

So, again we start with an activate I1 state. We input N = X3. The system moves to S3. This is possible because I1 only fails to activate S4 and S5 with a X4 and X5 input. Now, we are in S3 and we have to input the interval input. The system shifts to I4. Now we can observe how the system behaves when we input another N = X3. I4 inhibits the activation of S3, but not the activation of the error 3. So the error is activated instead of the S3. This leads to an incorrect output signal. The model doesn't even bother with

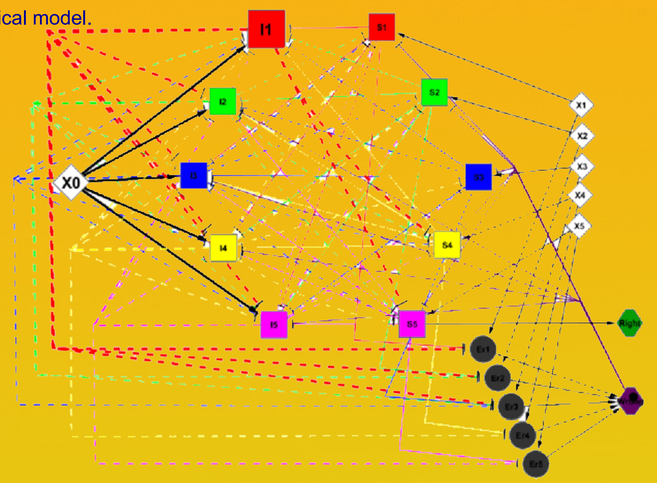

Fig 2. Our Mathematical Model Based on the finite state automaton

The boring theory

Several assumptions have been made to make the model work the way we want it to.

- I1 is always active at t=0, meaning that every sentence begins with an intermediate input to activate I1.
- The strength of every inhibition is the same on all the states.
- We introduce a stop signal, so we know when a sentence is finished

We used ODEs to model the system. ODEs are used for describing how a system changes over time. In general, our ODEs look like this:

State i = synthesis * (every incoming activation and repression) – degradation of i

For activation and repression we used a hill function: **act = ((conc*k)^m)/(1+(conc*k)^m)**

where k is the responsible for when the activation occurs and m for how fast it is. so the hill function tells you how long it takes for a protein to be synthesized and if it is synthesized how fast it is done. The repression is simply: **rep = 1 – act**

This way we set up a model as in the figure. This model does not represent the full automaton, but only the first four states. Our ODE equation is as followed:

$$dydt_i = syn_i \prod_{j \in i} \begin{cases} act_j, & \text{if } j \text{ activates } i \\ rep_j, & \text{if } j \text{ inhibits } i \end{cases} - \deg_i$$

The Results

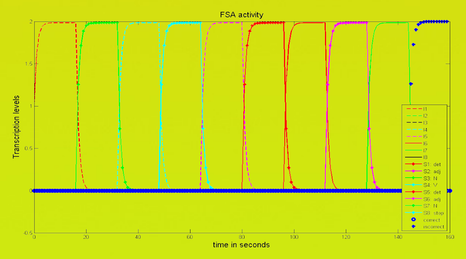

Fig 3. Shows the MATLAB simulation result when the correct sentence is tested with following order: “det, N, V, adj, N, Stop” all the protein levels for states and intervals represent good response in the simulated network and finally the correct protein level shows that the input sentence was right

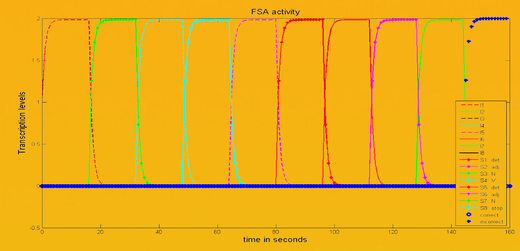

Fig 4. Shows the MATLAB simulation result for the incorrect sentence given with following order: “N, V, det, adj, adj” as seen before all protein levels are working fine before we reach the point which model detects an error in the order of inputs and automatically goes to incorrect state. By that point no more input is given as model is going back to initial state. This is also the case for the correct sentence

The Plasmid Suggestions

We designed suggestions for input and internal plasmids for the model devoloped. Designing input plasmid requires 5 exogenous activators which are labeled as X1…X5(fig 2) . Metal ion seems to be good candidates as well as pathways such as Methionine biosynthesis and Galactose degradation pathways. Let’s consider Iron ion as the first reagent .Induction of iron leads to regulation of fnr gene which acts as positive regulatory element for FdrA.

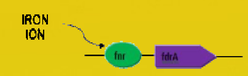

Fig. 5. This is the input plasmid involving one state which can be seen as S1 in graphical representation of our linguistic model. It should be able to inhibit 9 other genes selected for other 9 states. In order to implement internal plasmid based on our model each gene need to be inhibited by specific repressors for example, in case of fdrA the inhibitor NarI which needs specific binding site

Along with this, the gene should have the flexibility to be released from inhibition meaning that there must be a way to remove the repressor. Proteolysis could be a possible example. As we reach the last input reagent and it regulates the final input plasmid it should ultimately activate one of the 2 final GFP or RFP tagged genes, based on the semantically correct or incorrect sentence we pursue followed by the specific pathway. Implementation of such a genetic network needs extensive research upon existing genetic pathways in E.coli that would complement our mathematical model. But due to time constrain it was not possible to actually design such a synthetic network. Thus, it possibly be our future work!!

Literature Cited

1. ETH Zurich 2006
2. Modelling and Simulation of regulatory networks: A literature Review by Hidde de Jong
3. A deoxyribosome based molecular automaton by Milan N stojanovic and Danko Stefanovic