Evolving Concurrent Petri Net Models of Epistasis

Michael Mayo\(^1\) and Lorenzo Beretta\(^2\)

\(^1\) Dept. of Computer Science, University of Waikato, New Zealand
mmayo@cs.waikato.ac.nz
\(^2\) Referral Center for Systemic Autoimmune Diseases,
Fondazione IRCCS Ospedale Maggiore Policlinico di Milano, Italy
lorberimm@hotmail.com

Abstract. A genetic algorithm is used to learn a non-deterministic Petri net-based model of non-linear gene interactions, or statistical epistasis. Petri nets are computational models of concurrent processes. However, often certain global assumptions (e.g. transition priorities) are required in order to convert a non-deterministic Petri net into a simpler deterministic model for easier analysis and evaluation. We show, by converting a Petri net into a set of state trees, that it is possible to both retain Petri net non-determinism (i.e. allowing local interactions only, thereby making the model more realistic), whilst also learning useful Petri nets with practical applications. Our Petri nets produce predictions of genetic disease risk assessments derived from clinical data that match with over 92% accuracy.

Keywords: Petri net, genetic algorithm, epistasis, concurrency, systemic sclerosis, digital ulcers.

1 Introduction

Petri nets [13] are widely used abstract computational models of concurrent processes. Recently, they have found application as useful modeling tools in biochemistry, genetics and medicine (e.g. [2,6]).

They are best described as executable graphs with two different types of node: places and transitions. In a biochemical modeling situation, a place usually represents a substance and a transition stands for a reaction or process in which one or more input substances are transformed over time into one or more output substances. Petri nets have potential to realistically model what could be happening in real world situations because they are inherently concurrent. For example, in a net, two pathways of multiple transitions may fire simultaneously, thus simulating two concurrent processes.

Figure 1 depicts a simple Petri net with three places and two transitions. The places, \(P_0\), \(P_1\) and \(P_2\), represent three different chemical substances, and the transitions, \(T_0\) and \(T_1\), represent two different reactions that can occur between them. Petri nets represent the concentration of a substance at a particular point in time by “marking” each place with an integer number of tokens. These tokens move around the net as the transitions fire.
For example, suppose in Figure 1 that there are 10 tokens at $P_0$, and no tokens at $P_1$ and $P_2$. The overall marking of the entire net is the vector $<10,0,0>$. The arcs indicate either transition inputs or outputs, depending on the directionality. They are labeled with a quantity of tokens consumed or produced. $T_0$, for example, represents a chemical process in which $P_0$ is being converted into $P_2$, with one unit of $P_0$ being consumed for every three units of $P_2$ being produced. If $T_0$ fires once, the marking of the net will become $<9,0,3>$. If it fires twice, it will become $<8,0,6>$. $T_1$, on the other hand, represents an entirely different reaction with $P_0$ and $P_2$ as inputs, and $P_1$ as output. Because $T_1$ requires three units of $P_2$ as an input, it cannot fire until $T_0$ has fired at least once. If this happens, the marking will change from $<10,0,0>$ to $<9,0,3>$ (after $T_0$ fires) and then to $<7,5,0>$ (after $T_1$ fires).

Transitions can only fire if there are sufficient input tokens available (i.e. the number of tokens at an input place cannot fall below zero), and if they are not inhibited. An example of an inhibitor in Figure 1 is the arc from $P_1$ to $T_0$: if ever $P_1$ has a non-zero quantity of tokens present, then $T_0$ is effectively turned off.

The only other time that a transition cannot fire is if one of its output places has insufficient capacity. For example, suppose the maximum capacity of all places in Figure 1 is 10 tokens, and the current marking is $<7,6,9>$. Although $T_1$ has sufficient inputs available at $P_0$ and $P_2$, there is insufficient capacity at the output place $P_2$, so $T_1$ cannot fire.

It should be evident by now that Petri nets are concurrent and non-deterministic models. Transitions may fire in any order, and if they do not share common inputs or outputs, they can fire concurrently.

Non-determinism does have some issues when models are to be executed on serial computers. If there are two or more transitions enabled, which one should fire first? The simplest answer to this question is to enforce an arbitrary priority amongst the transitions [13]. For example, in Figure 1, $T_0$ may have a higher priority and therefore always fire before $T_1$, if they are both enabled at the same time. This strategy simplifies a non-deterministic Petri net into a deterministic model.

An alternative answer is to make the transitions fire stochastically. Of those that are enabled, one of them is selected to fire at random; and sometimes, in order to give all enabled transitions a fair chance of firing, those that have recently fired are not permitted to fire again until a certain amount of time has elapsed.

A significant issue with both of these solutions is that they require global coordination. In other words, in order to select the next transition to fire, all transitions must be examined globally. Nature, however, is unlikely to employ this level of global coordination; natural systems are more likely to evolve gradually with many local, concurrent interactions. The issue is therefore how to relax the requirement of global coordination from our Petri net models in order to make them more realistic and therefore more interesting.

In this paper, we address this specific problem in the context of modeling disease-causing epistatic interactions between genes. Our solution is to convert the Petri net