Implementing Elections as Networks
Mathematical Modeling & Computing for Scientists
Dr. Allali & Dr. Waldrop

Jeremy Wayland

Fall 2020

Introduction

Systems that describe how inputs transition into outputs are ubiquitous in science and engineering. We draw networks and diagrams to capture these processes and solve complex problems. Examples of networks include electrical circuits, Petri Nets, Feynman diagrams, and many more. How do we reason about such systems? Mathematically, how do we represent an open network (one with inputs and outputs)? Is there a formalism to confirm our intuition for how these networks should combine and/or run in parallel? Probing these questions is an active field of research in network theory and applied category theory.

Background

Currently, two of the most prolific formalisms for open networks were developed by Dr. John Baez and his cohort at the University of California, Riverside; the first is named structured cospans (Courser and Baez) and the second decorated cospans (Fong and Baez). Both begin with the outlook that a network can be seen as a set equipped with some additional structure and diverge in their specification of this structure. Together, they have been applied to study markovian processes, chemical reaction networks, electrical circuits, and Petri Nets (with and without rates) in [3],[4],[2],[5]. Structured cospans is a newer, sleeker framework for understanding open networks that ties together some of the loose ends left behind by decorated cospans. Structured cospans gives us an ability to represent these types of networks in a much more general setting. When using structured cospans to specify Petri Nets, the formalism provides the infrastructure to generate tensor products and composition of networks. This allows for the production of the following more complex systems as illustrated in the example below.

A Brief Example

Consider a concrete example from [1].

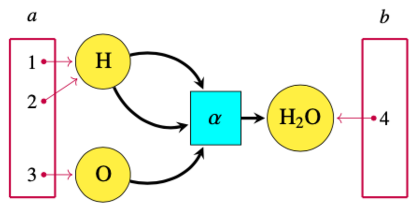

Figure 1: The formation of H_2O in an open petri net.

In the figure above, a, b are the sets of inputs and outputs, respectively and α is called a transition. This idea of "openness" allows you to think of this reaction as a belonging to some larger, more complex series of events. We could also consider a decomposition of H_2O, given by the open petri net below:

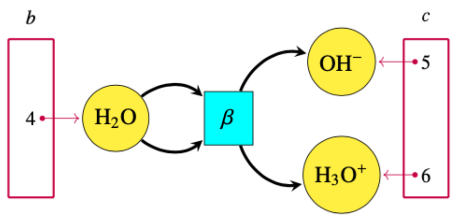

Figure 2: Ionization of H_2O.

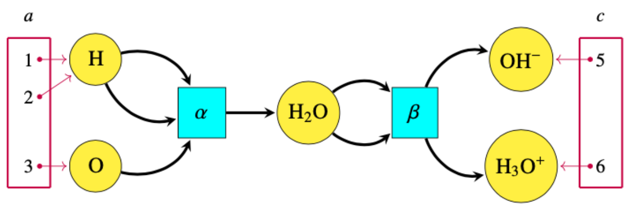

Figure 3: Composition of Figures 1 & 2

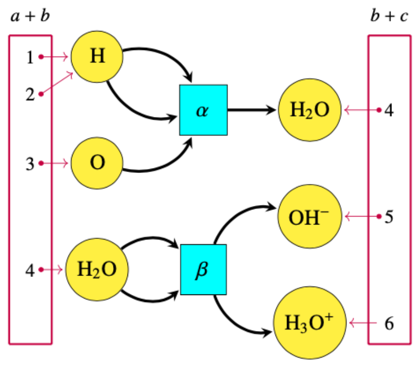

Figure 4: Tensor Product of Figures 1 & 2

Voting Webs

In my udergraduate thesis [6], I construct a categorical representation of a voting system which I call **VotingWeb** and then show that it has a commutative monoidal structure. This was motivated by the similarities I noticed between petri nets and voting systems. This formulation could pave the way for using high powered categorical machinery to describe, build, and analyze elections. Here are the basic definitions: Let V be a nonempty, finite set of n *voters* denoted by $\{1, \cdots, n\}$. Let C be the set of m *candidates* $\{c_1, \cdots, c_m\}$. A ballot then is a full set of preferences submitted by a given voter; this is nothing more than a linear order on C. Thus, consider the set $\mathcal{L}(C)$ which will denote the set of a all linear orders on C. The outcome of an election is determined by a *voting method*; this will be a function $f : \mathcal{L}(C)^n \to C$ that picks out a single winner. This is the only information necessary to describe a voting system, so I represent them with the following triple: $\langle C, n, f\rangle$ for $C \in$ **FinSet**, $m \in \mathbb{N}$, f is a voting method. One can envision the system as a collection of voters coming together with packaged information (a total order on candidates) that then undergoes a transition phase (votes are counted) and returns a winning candidate. What is this if not a network with inputs that generate prescribed outputs? In most cases, you are really transforming m strict linear orders on C, into a single (potentially non-strict) linear order on C that represents the rankings of the candidates. I choose to define the set of possible voting systems as T_{vote}, where each element in T_{vote} can be generated by one of the triples specified above. Without some of the details, this allows for the following definition of a voting web (an object in the category **VotingWeb**):

Definition. *A Voting web is a specialized Petri Net that is defined by a source and a target function that have the following form:*

$$\begin{tikzcd} T_{\text{vote}} \arrow[r, shift left, "s"] \arrow[r, shift right, "t"'] & \mathbb{O}[C] \end{tikzcd}$$

where $\mathbb{O}[C]$ represents the set of all preorders on a set C. These functions s, t interact with a *fixed* set of transitions, T_{vote} and are subject to the following restrictions:

- $|\text{Image}(s)| = n$ for $n \in \mathbb{N}$. So for a system with n voters the source function picks out precisely n preorders.
- $|\text{Image}(s)| = 1$. So the target function picks out a single preorder on C that represents the outcome of the election according to a well defined voting method f.

Project Description and Motivation

This class project will look to extend the work done on Voting Webs and contribute to a larger research initiative: to package the formalism of structured cospans and the subsequent results into a strongly typed language that can be leveraged by scientists using networks. The idea is to provide universal accessibility to the formalism via an open source interface, eliminating the need to parse through dozens of papers and advancing the use of category theory in modern applied sciences. The scope of this project will be as follows: first experiment with implementing voting webs in a more familiar language R. Then, take this experience and use it to implement voting webs in Haskell and exhibit the computation of deciding an election. This will allow me to experiment with the implementation of a network in Haskell in a regime that is familiar, mundane, and prevalent. I hope to gain insight on the interplay between syntax, formalism, and abstraction in a concrete setting before generalizing this idea to structured cospans. This will be my first experience using Haskell, and thus a large portion of the project will boil down to me understanding how the built in functionality leverages the concepts/structures in category theory that are relevant to voting webs. The next significant task will be choosing a specific implementation, of which there will most likely be many attempts and failures. I anticipate working quite closely with Dr. Moshier and maybe even Dr. Kurz to settle on an implementation that lays a good foundation for generalization to the underlying theory of structured cospans.

References

[1] John C. Baez and Kenny Courser. Structured Cospans. pages 1–43, 2019.

[2] John C. Baez, Brandon Coya, and Franciscus Rebro. Props in network theory. *Theory and Applications of Categories*, 33:727–783, 2018.

[3] John C. Baez, Brendan Fong, and Blake S. Pollard. A compositional framework for Markov processes. *Journal of Mathematical Physics*, 57(3):1–43, 2016.

[4] John C. Baez and Jade Master. Open Petri nets. *Mathematical Structures in Computer Science*, pages 1–30, 2020.

[5] John C. Baez and Blake S. Pollard. A compositional framework for reaction networks. *Reviews in Mathematical Physics*, 29(9), 2017.

[6] Jeremy Wayland. *Voting Systems : An Investigation into Strategic Voting and the Commutative Monoidal Structure of Elections*. 2019.