Petri Net JSON Schema

Micah Halter

September 10, 2020

Formulation

We can define a Petri net to be the following diagram of finite sets

$$\begin{tikzcd}
Str & S \arrow[l, "sname"'] \arrow[r, "conc."] & \mathbb{R}^+ \\
I \arrow[ur, "is"] \arrow[dr, "it"'] & & O \arrow[ul, "os"'] \arrow[dl, "ot"] \\
Str & T \arrow[l, "tname"] \arrow[r, "rate"'] & \mathbb{R}^+
\end{tikzcd}$$

Here, T is the set of transitions, S is the set of states, I is the set of input edges from some state s to some transition t, and O is the set of output edges from some transition t to some state s where $s \in S$ and $t \in T$. Along with these definitions for transitions, we also have maps from $S \to \mathbb{R}^+$ that map states to their initial concentrations and $S \to Str$ that map states to labels. Similarly, we have maps from $T \to \mathbb{R}^+$ that map transitions to their transition rates and $T \to Str$ that map transitions to labels.

With this formulation, we can naturally convert the resulting structure into JSON schema:

```json
{
  "T": [
    { "rate": 0.0003, "tname": "inf" },
    { "rate": 0.2, "tname": "rec" }
  ],
  "S": [
    { "concentration": 990, "sname": "S" },
    { "concentration": 10, "sname": "I" },
    { "concentration": 0, "sname": "R" }
  ],
  "I": [
    { "it": 1, "is": 1 },
    { "it": 1, "is": 2 },
    { "it": 2, "is": 2 }
  ],
  "O": [
    { "ot": 1, "os": 2 },
    { "ot": 1, "os": 2 },
    { "ot": 2, "os": 3 }
  ]
}
```

Fields:

- `T`: an array of transitions each with both a `rate` and `tname` specified to represent the transition rate and the label of the transition respectively
- `S`: an array of states each with both a `concentration` and `sname` specified to represent the initial concentration and the label of the state respectively
- `I`: an array of input edges each with an input transition (`it`) and input state (`is`) whose values are indexes into the `T` and `S` arrays respectively. These represent an edge from state `is` to transition `it`.
- `O`: an array of output edges each with an output transition (`it`) and output state (`is`) whose values are indexes into the `T` and `S` arrays respectively. These represent an edge from transition `it` to state `is`.

Dynamic Transition Rates

In order to support dynamic transition rates, we will be providing raw Julia code blocks that defines a function as either a function of time (`t->begin ... end`) or both time and current state (`(u,t)->begin ... end`). This will allow us to provide the math the defines the change of the rate in regards to different parts of the system while maintaining the structure of the original model. For interoperability, these rates can either be transpiled between languages or you can focus on just extracting the model structure from the `json` file and write your own definitions for the rates. An example of this would be `{ "rate": ":((u,t)->((3/sum(u))/(t+1)))", "tname": "inf" }`. Here the rate of the transition `inf` changes as a function of the current state of the system (total population) and time, where infection rate decreases over time to represent a virus weakening over time. Another example can be found in the `chime.json` where the rate of infection is dependent on time to model changes in intervention policies.

Note: While this is how we plan to represent dynamic rates currently, we are investigating using a dataflow graph in the future to represent the math that defines dynamic rates in a more programming language/implementation agnostic way.

SIR Example

```json
{
  "T": [
    { "rate": 0.0003, "tname": "inf" },
    { "rate": 0.2, "tname": "rec" }
  ],
  "S": [
    { "concentration": 990, "sname": "S" },
    { "concentration": 10, "sname": "I" },
    { "concentration": 0, "sname": "R" }
  ],
  "I": [
    { "it": 1, "is": 1 },
    { "it": 1, "is": 2 },
    { "it": 2, "is": 2 }
  ],
  "O": [
    { "ot": 1, "os": 2 },
    { "ot": 1, "os": 2 },
    { "ot": 2, "os": 3 }
  ]
}
```

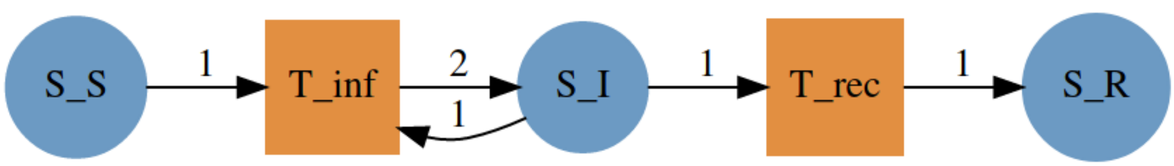

Figure 1: Resulting SIR Petri Net