Implement a tool for simple Petri Net

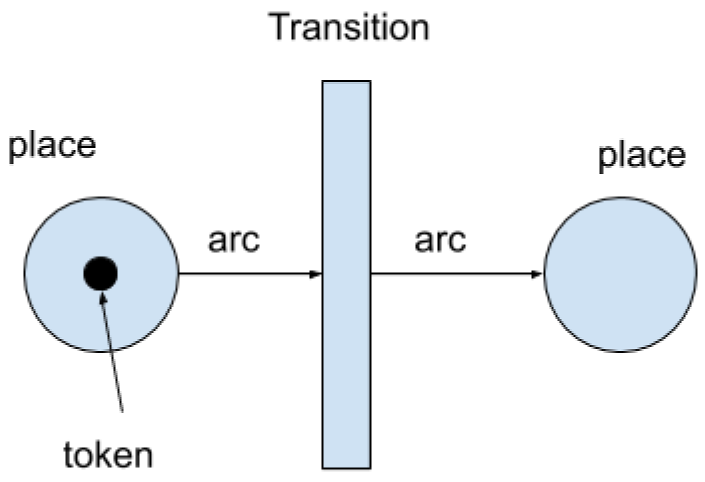
-Kavyansh Gangwar (18075027) and Kayala Nithin Sai (18075028)

Petri Net

A Petri net, also known as place transition net, is one of the mathematical modeling languages for distributed systems. It is a class of discrete event dynamic systems. In simple words, a Petri net is a directed bipartite graph that has two types of elements: Place and Transition.
Place: A place in the Petri net is graphically represented as a circle. A place is connected to another place with the help of a transition. A place contains tokens that add dynamicity to the whole entity.
Transition: A transition in the Petri net is graphically represented as a rectangle. The main aim of a transition is to connect two sets of places and transfer tokens from one set to another when all the places of the input set have tokens by the process of firing. The connection of place to transition and transition to place is represented by arcs. The place from which the arc runs towards a transition is said input place and is contained in the input set of that transition and the place to which the arc runs from transition is said to be the output place of the transition and is included in the output set of that transition.
Tokens: Tokens are the features of Petri net that are used to add dynamicity to the whole model. Tokens are contained in places and are generally used to represent the switch-on state of the place. There can be many tokens in a place.
Firing: Firing is the procedure of transfer of tokens from the input set of transition to the output set of transition. It is an atomic process. When all the places in the input set of a transition contain tokens then the transition fires. In this process, the transition takes the minimum amount of tokens from the input set contained by a single input place from all the input places and sends the same amount of tokens to all the output places.
Let the set of input places be represented as I and the set of output places be represented as O and tokens inside the place represented as tok(P_i) where P_i is a place. Then in a firing process

$$Tokens fired = min(tok(I_i)) \ \forall \ I_i \ \mathrm{E} \ I$$

So after firing all the places on the input side will have $tok(I_i) - Tokens fired$ amount of tokens and all the output places will have $tok(O_i) + Tokens fired$. This is of course the maximum number of tokens for the output places as there can be more than one transition from the same set of input places (a subset of input places) as firing is a non-deterministic process.
Petri nets are well suited for modeling the concurrent behavior of distributed systems. They are also used in Boolean differential calculus, Computational Biology, Concurrent Programming, Reachability Engineering, etc.



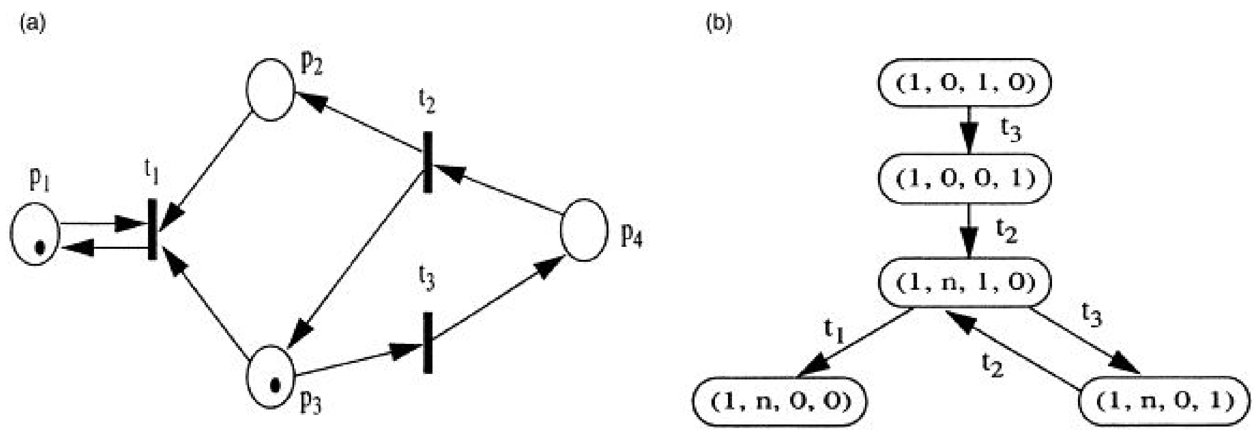
A simple Petri Net

Reachability Graph

A reachability graph is a way of representing the states of the Petri net. A marking M is reachable from the initial state, M_0 if there exists a firing sequence that brings us from the initial state of the Petri net to the state corresponding to the M net. A state of Petri net can be theoretically thought of as a representation that describes whether a place has tokens or not. In a reachability graph, nodes correspond to the reachable markings and edges correspond to the relations.

Reachability graph is used for behavioral analysis of Petri net. The reachability graph is a directed graph that tells all the states that can be reached from the initial state M_0 by the set of transitions represented as edges. The reachability graph can be infinite even for a finite Petri net. In such cases, a Coverability Tree is generated.

A state in the reachability graph can be represented as an m-tuple of the form (n_1, n_2, n_3, ..., n_m) where m is the number of places in the Petri net and n_i is the tokens at a place p_i. A state can be reachable from the initial state in the Petri net from more than one path of transitions moreover there can also be a cycle in the reachability graph.



(a) Petri net.
(b) Reachability Graph

How we will proceed with the project

In this project next, we will try to learn an algorithm to find the reachability graph and the coverability tree. Since we are dealing with a simple Petri net we are assuming the Petri net will not contain any cycle and will be unidirectional though there can be branches in the Petri net. We will implement the algorithm in code and use python as the programming language.