|  |
| --- |
| EFREI |
| Mini project : implementation of a CTL Model checker |
| Formal modelling and verification of critical systems |

|  |
| --- |
| Victor DALIBOT  05/02/2022 |

## Table of contents

[Table of contents 1](#_Toc95070539)

[Technical choice 2](#_Toc95070540)

[Data Input 2](#_Toc95070541)

[Discussion 5](#_Toc95070542)

[Conclusion 5](#_Toc95070543)

## Technical choice

I chose python because it is an object-oriented language, everything in python is an object so it is easier to deal with class and structure such as a CTL formula and a Kripke structure.

## Data Input

Kripke structure

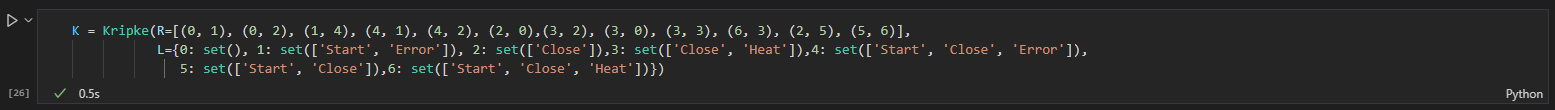
The input for the kripke structure is:

* R : a set of finite set of states
* L : a labeling function

The structure is composed of:

* S : a collection of states
* S0 : a collection of initial states
* R : a collection of edges
* L : a labelling function
* Functions

Ex: Definition



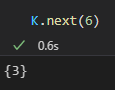
Ex: Functions

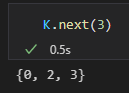
A function to check the states of a Kripke Structure

Une image contenant texte

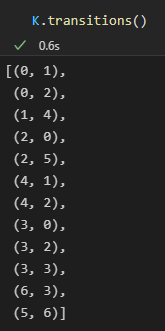
Description générée automatiquement

A function to check the successor of a state

 For state 6

 For state 3

A function to check all the transition



A function to check the labels on the transition

Une image contenant texte

Description générée automatiquement

CTL formula

The parser parses the formula into a suite of recursive functions calling another one until the formula reaches its end.

For example:

Une image contenant texte

Description générée automatiquement

The term EX(psi) becomes E(X(psi)), E is called with X(psi) as parameter.

The function is divided until it cannot be divided anymore and will be resolved one after another.

For example:



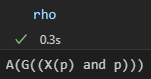
This formula Φ when printed will be understood as:

Une image contenant texte

Description générée automatiquement

Sub-formula can be used together to form another formula:

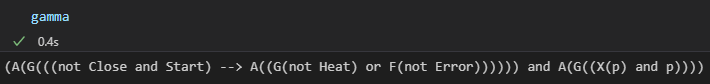




Associated with Φ and “And” we have a new formula:



Which when printed will give us:



## 

## Discussion

Encountered difficulties

Among the difficulties encountered, one was tougher than the others and it was the parsing, before thinking of a recursive call of different functions I intend to make one function that will solve an entire formula. But this solution implies a massive tree of possibilities of “if, elif”.

Possible Improvements

To provide a graphical web interface, which is doable in python with libraries.

To make a random generator of CTL formulas and Kripke structure.

To build a plugin allowing the generation of the kripke structure from a formal model such as Petri net or states machines.

## Conclusion

The goal was to make an algorithm able to solve CTL formula for a given Kripke structure. Despite some difficulties, I was able to achieve the project by implementing the algorithm and the parser.