**CS 686 - Final Project Report**
**Chronicle Tree**
**Ali Alnajjar**

## 1 Abstract:

Debugging is the most complex and time consuming step in the development process. Especially with the increasing complexity in modern programming languages. Many developers don't realize the complexity they created in their projects until they try to debug it. On the other hand many new programmers and new computer science students find it hard to visualize (trace) the variables in programs. In this project I tries build an analysis tool to help understanding the program behavior. This tool will help to speed up the learning and debugging process by showing a visual analysis of the program.

## 2 Introduction:

In modern programming languages we have many concepts, tools and techniques to helps us in solving complex problems. That helps in developing efficient, quick, and user-friendly applications in an ease way. Unfortunately that increased the complexity of the application and made it harder to debug an application. For example using multi threads helps in perform many tasks almost the same time but on the other hand debugging multi thread is one of the most complex and long steps in development.

There are many resources and tools can help interested people to learn programming. Programming itself is not hard to learn Especially with the support provided my IDEs. The hardest part in the learning process is to learn how to trace programs. Tracing a program needs an imagination to predict how the program will behave during execution. This is the point where many people need some help.

ChronicalTree a tool to analysis python program. This tool will trace a cretin object in the program and provide his full history. It will point out all the states where the object changed. Find all the other objects and elements that effects this object and get their history. Finally the tool will draw a map showing the object full history.

## 2 Design:

### 2.1 State Saving:

In this project Copy State Saving (CSS) was used to save full state after each event execution. Using stackless python a pickled version of the tasklet is begging saved to a sqllite3 database. In order to decrease overhead and avoid delaying program execution multiprocessing was used. The syntax of record instruction is:

```
spython ../src/sa.py record <state file>
<python code>
```

### 2.2 Analyzing:

In the analysis phase the tool find the state where the object was created. Then Find the states where the object modified by comparing the id and value of the last state with the ones from the current state. In these stated the tool first get the bytecode of the state using a function called GetBytecode then get the objects and constants appeared in this state and add them to a list called ObjList. For each object in this state the tool will get the history of it until this state using the function called GetHistory. GetHistory function return a state number where the object was last modified. Using this information the connection between objects is being added to a list called ObjRObj.

### 2.3 Tree Drawing:

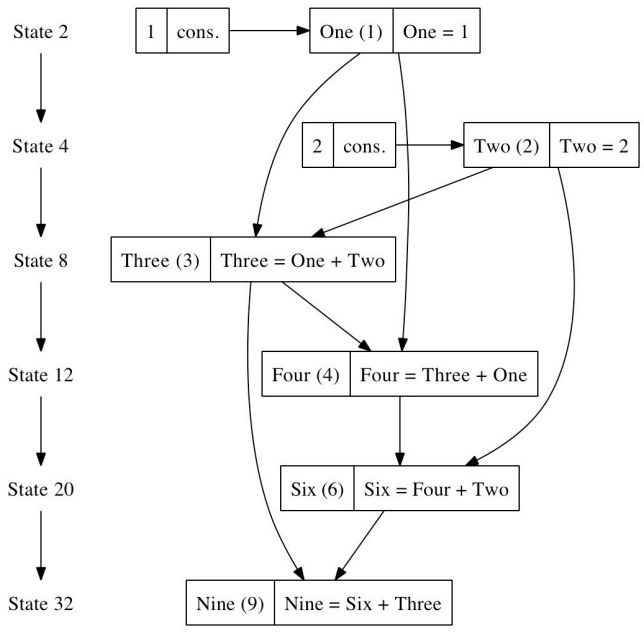
After completing the analysis phase we have two list the first list ObjList includes the objects,constants, functions, and modules and the other list is ObjRObj which includes the relations between the entries in the first list. Using these list we can generate the graph using dot. a dot file will be generate and from this file a ps file will be generated.
The syntax of drawing instruction is :

```
spython ../src/sa.py history <state file>
<object> <beginning state (optional)>
```

## 3 Example:

Here is an example of a python program and how the graph will look like.

```
 #python sample
One = 1
Two = 2
Three = One + Two
Four = Three + One
Five = Four + One
Six = Four + Two
Seven = Four + Three
Eight = Seven + One
Nine = Six + Three
```




## 4 Conclusion:

Many developers have a hard time trying to find bugs in programs. With the current programing techniques programs became more complex and debugging became more painful. In this project I tried to find a way to allow the developer have a blueprint shows the history of a cretin object that includes all changes happened to his object and all the elements that effects the value of it.