Xcos Profiler

Proposal for GSoC 2017

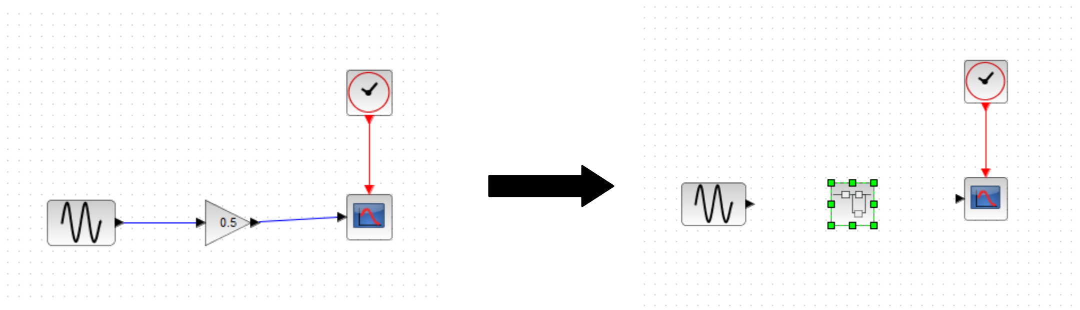
Student Name: Rui SHI
Mentor: Dr.Umut Durak

Description

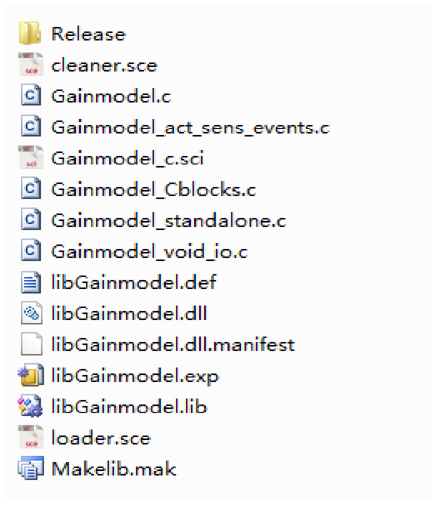
Since Scilab is an open source, various practitioners use it to simulate real-time applications and systems. However, Simulation cannot be achieved in zero time. Knowing execution time is a key to develop an accurate simulation in real-time areas. Especially in some areas about life safety, such as military equipments, automotive and aerospace industries, measuring execution time accurately is more important.
As we known, Matlab has a Profiler function which could measure where a program spends time. By using Matlab Profiler, we could evaluate and improve performance of programs. What I want to do is to make a similar function for Scilab.
We use Xcos to make models that have many blocks. In some cases, we want to know execution time of these blocks. A variety of literatures introduce the way of measuring execution time but they did not present directly time to users. Therefore, in this project, I will measure time and present time to users.

Ideas

First of all, Scilab/Xcos has a Code generation function that can generate c codes from a superblock.



There are a number of files in the specific folder. We need to choose the computational file and measure the execution time.

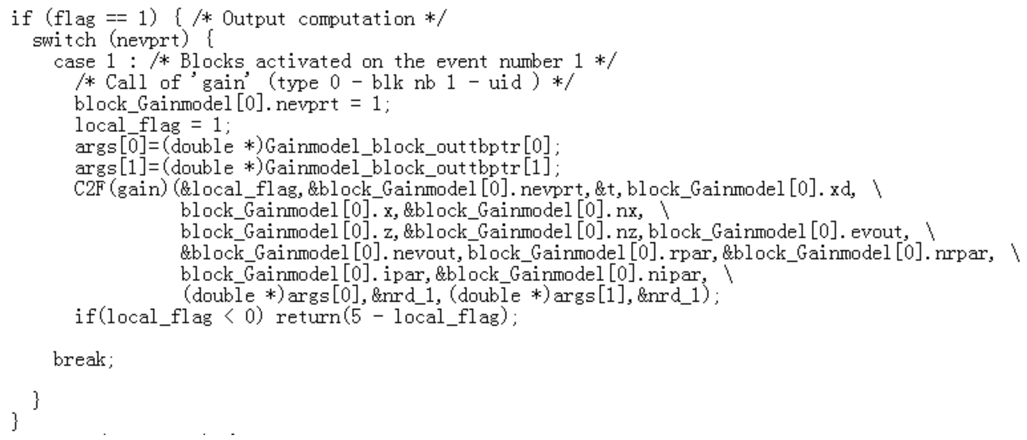


```
Release
cleaner.sce
Gainmodel.c
Gainmodel_act_sens_events.c
Gainmodel_c.sci
Gainmodel_Cblocks.c
Gainmodel_standalone.c
Gainmodel_void_io.c
libGainmodel.def
libGainmodel.dll
libGainmodel.dll.manifest
libGainmodel.exp
libGainmodel.lib
loader.sce
Makelib.mak
```

Secondly, I choose one of techniques to measure execution time which is clock() function. Here is an example of a program that uses *clock()*.

```
#include <time.h>
clock_t start,end;
double total;
start = clock();
end = clock();
total = (double) (finish - start) / (double) CLK_TCK;
printf("Total = %f\n",total);
```

Besides, the computational file includes various parts. For example, when flag is equal to one ,it is output computation. We could use clock() function to calculate execution time of output computation.



```
if (flag == 1)  { /* Output computation */
  switch (nevprt)  {
    case 1 : /* Blocks activated on the event number 1 */
      /* Call of 'gain' (type 0 - blk nb 1 - uid ) */
      block_Gainmodel[0].nevprt = 1;
      local_flag = 1;
      args[0]=(double *)Gainmodel_block_outtbptr[0];
      args[1]=(double *)Gainmodel_block_outtbptr[1];
      C2F(gain)(&local_flag,&block_Gainmodel[0].nevprt,&t,block_Gainmodel[0].xd,  \
                block_Gainmodel[0].x,&block_Gainmodel[0].nx,  \
                block_Gainmodel[0].z,&block_Gainmodel[0].nz,block_Gainmodel[0].evout,  \
                &block_Gainmodel[0].nevout,block_Gainmodel[0].rpar,&block_Gainmodel[0].nrpar,  \
                block_Gainmodel[0].ipar,&block_Gainmodel[0].nipar,  \
                (double *)args[0],&nrd_1,(double *)args[1],&nrd_1);
      if(local_flag < 0) return(5 - local_flag);

    break;

  }
}
```

```
clock_t start,end;
double total;
start = clock();

if (flag == 1) { /* Output computation */
  switch (nevprt) {
    case 1 : /* Blocks activated on the event number 1 */
      /* Call of 'gain' (type 0 - blk nb 1 - uid ) */
      block_Gainmodel[0].nevprt = 1;
      local_flag = 1;
      args[0]=(double *)Gainmodel_block_outtbptr[0];
      args[1]=(double *)Gainmodel_block_outtbptr[1];
      C2F(gain)(&local_flag,&block_Gainmodel[0].nevprt,&t,block_Gainmodel[0].xd, \
                block_Gainmodel[0].x,&block_Gainmodel[0].nx, \
                block_Gainmodel[0].z,&block_Gainmodel[0].nz,block_Gainmodel[0].evout, \
                &block_Gainmodel[0].nevout,block_Gainmodel[0].rpar,&block_Gainmodel[0].nrpar, \
                block_Gainmodel[0].ipar,&block_Gainmodel[0].nipar, \
                (double *)args[0],&nrd_1,(double *)args[1],&nrd_1);
      if(local_flag < 0) return(5 - local_flag);

    break;
total = (double) (finish - start) / (double) CLK_TCK;
printf("Total = %f\n",total);
```

After this small step succeeding, we can try to make programs to add clock() function automatically in computational file.

Finally, what is most special is to present execution time to users. In Matlab, users can choose Profiler function to get information about execution time. Therefore, I plan to make a similar function. When users want to know execution time, the numerical data will present below every block.

Schedule

Time	Main Work
First week	Add clock() function to computational file and compile it successfully
Second to Third week	Automatically add clock() function to computational file
Forth to Eighth week	Find a way and make programs to present numerical data of execution time below blocks
Ninth to Twelfth week	Optimise this project

Self-Introduction

I am 24 years old from China. I got my bachelor degree(Electronic Information Engineering) in China. And now I am studying Master program(Internet Technologies and Information Systems) in Clausthal University of Technology in Clausthal-Zellerfeld in Germany.

I am good at C programming and also learned Java, Matlab, Simulink.

1. We measured moisture by using moisture sensor ,transferred data by using Arduino and presented data in an Android app which was designed by ourselves. It was a team work and I was responsible for some parts of designing Android App and designing shell. The result video could be watched in https://www.youtube.com/watch?v=tG3SWIxuJBE&feature=youtu.be
2. I used X2C blocks to substitute Xcos blocks in model and generate c codes.
3. I used C language to make a small game. Users can play game and chat with each other

which is C/S basing on socket.

The first time I know Scilab is in my Master course “Graph models and simulation”. I used X2C to substitute Xcos models and generated c codes. Then I tried to use Cblock which contains c codes to substitute original blocks.

In addition, In my bachelor, Matlab was a compulsory course. There is something similar between Matlab and Scilab. I also used Simulink to simulate communication models.

At last, I promise I will do full time work and spend at least 36 hours a week in GSoC.

Here are my contact information:

Name: Rui SHI
Email: rui.shi@tu-clausthal.de
Mobile: +49 015166073414
Postal Address: Leibnizstr. 22, Zimmer 19,38678 Clausthal-Zellerfel, Niedersachsen, Germany