EDA项目汇报

基于动态并行的混合仿真设计

第17组 祝彦翔 22307130073

Content

1. 项目背景

2. 三种项目实现方案

3. 项目效果

4. 项目总结

1. 项目背景

PJ1 背景

目标:
实现概念性混合仿真器（M仿真器），使用***“锁步”同步机制***。
抽象数字（D）和模拟（A）仿真器行为，强调同步机制。
锁步机制：D与A仿真时间完全对齐，空闲时不超前。

事件类型:
D事件（数字）：V，dT，T，bM
A事件（模拟）：V，Vth，dT，T，bM
M事件（混合）：虚拟事件，仅用于同步，不输出

初始：T=0时发生首个A事件
触发：D事件生成A事件，或A事件V跨Vth生成D事件。

同步规则:
D-to-A：D事件生成同时间A事件（V=D的V）。
A-to-D：A的V跨Vth，生成D事件（V=前D的V*int(A的V≥Vth)）。
优先级：A-to-D优先于D-to-A

接口设计:
仿真器（A/D）:
推进仿真：输入当前事件，输出下一事件（bM=0）。
被同步：接受同步请求，生成下一事件（bM=-1）。
仿真底板:
同步判断：根据D/A事件判断是否同步及方向。
推进仿真：执行同步（调用被同步）或推进两仿真器。

PJ1 实现核心逻辑

数据结构管理

为了存储事件，对不同的事件类型进行了结构化管理，以及为了维护队列将不同的事件结构变形而放入一个专门的事件结构中进行调度。事件结构包括AEvent、DEvent、Event存储事件数据。**在锁步式下，Event数组大小为2，存取当前运行目标A和D事件。主代码核心即为维护Event结构数组。**

事件生成（simulator.c/.h）

事件生成是利用generate_next_A和generate_next_D基于前一事件、当前时间和随机值生成新事件，确保新事件的各项参数符合要求。

文件读取(config.c/.h)

Con.txt文件读取，确保不符合要求的输入文件被报错处理，读入变量为全局变量。

PJ2 背景

- **目标**：实现高效并行化混合仿真器（M仿真器），优化同步机制以减少运行时间。
  - 相较期中PJ（串行锁步），重点提升仿真效率。

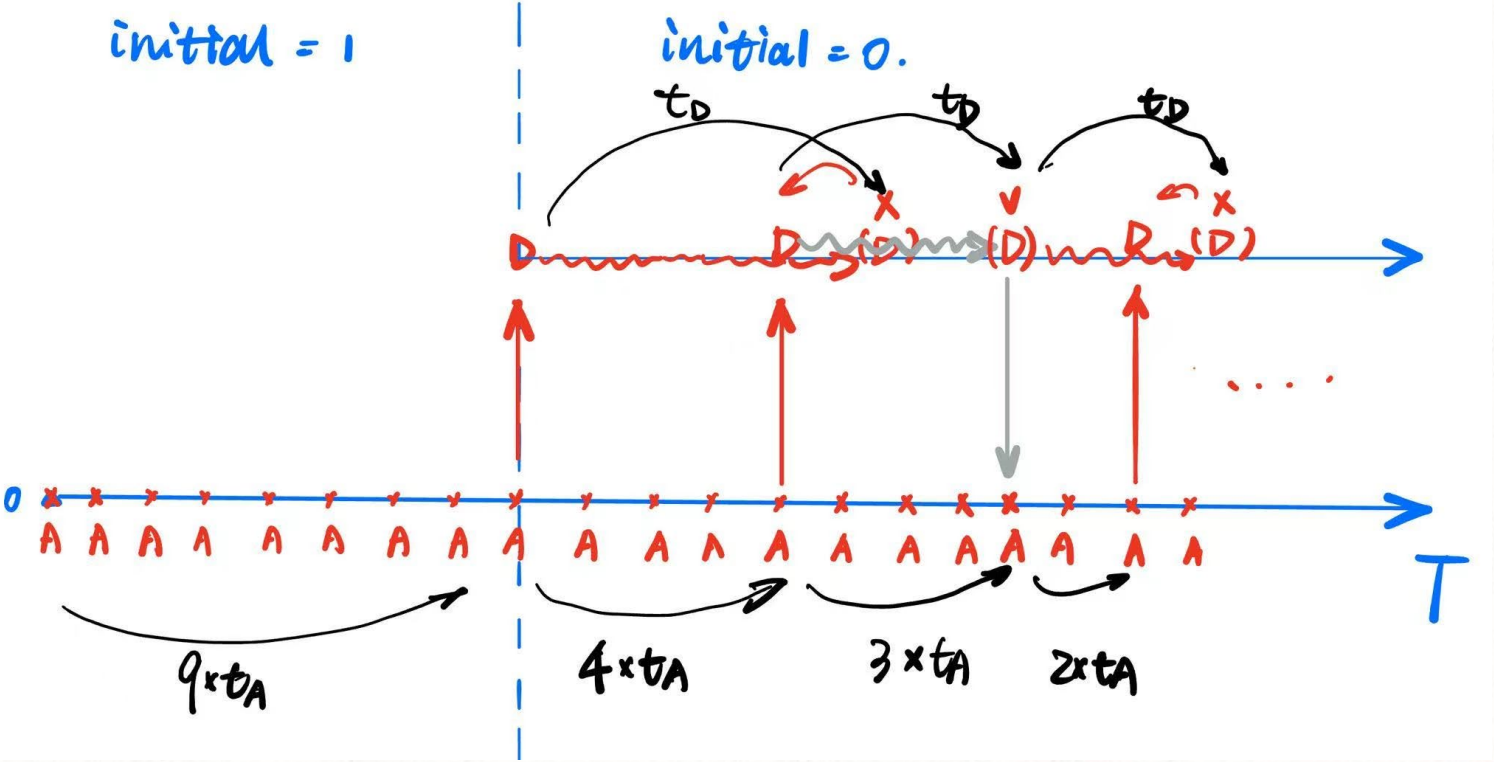
- **工作要求**：
  - **tA**（整数，[1,10]）：A事件（bM≠-1）前停顿tA ms，模拟A仿真器计算耗时，用户在con.txt中指定。
  - **tD**（整数，[1,10]）：D事件（bM≠-1）前停顿tD ms，模拟D仿真器计算耗时，随机生成。
  - **输入文件（con.txt）**：新增tA字段

- **效率分析**：
  - **串行锁步**：T = T_A + T_D + T_1。
  - **A/D事件并行锁步**：T = Σmax(t_A, t_D) + T_2。
  - **高度并行化仿真**

2.三种项目实现方案

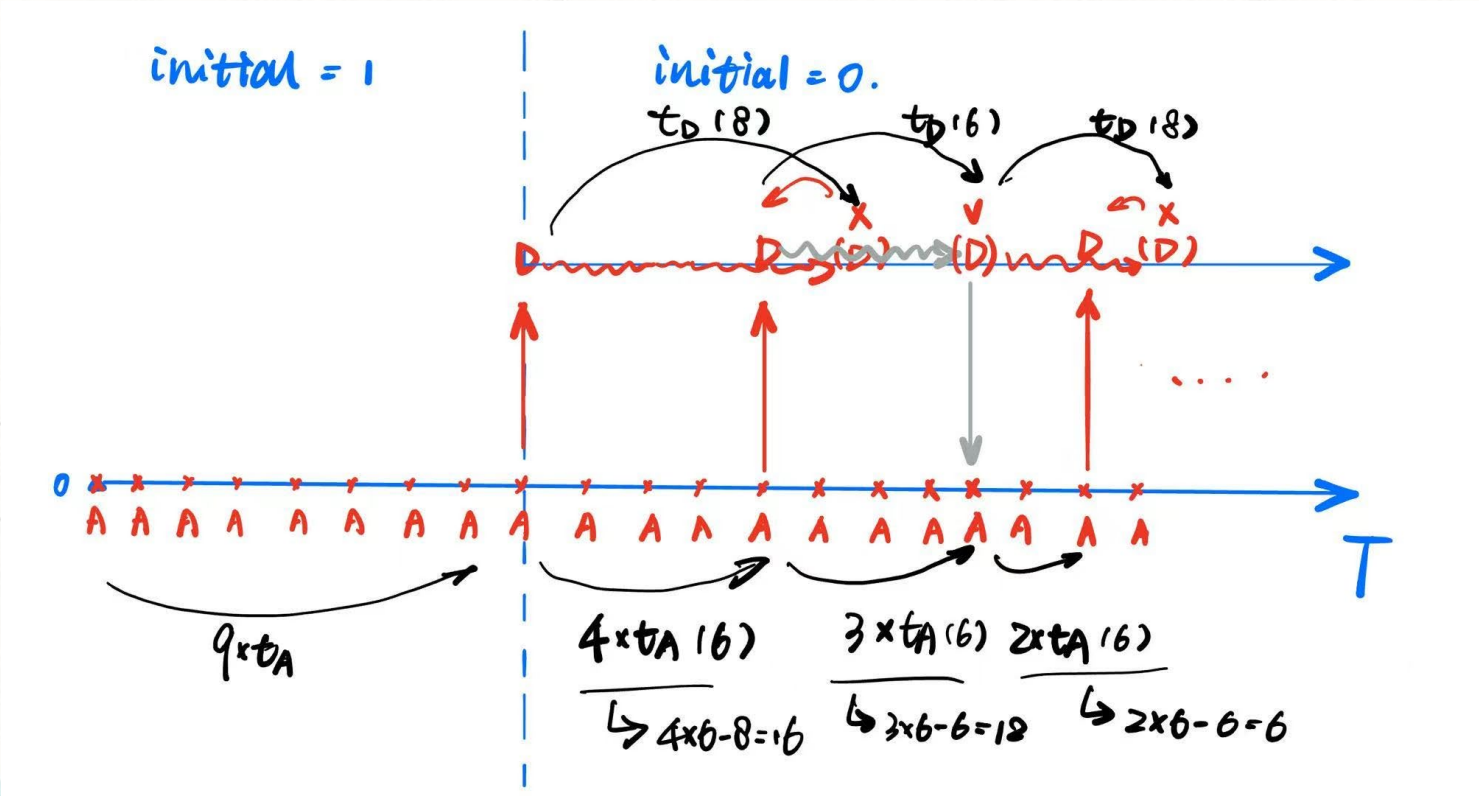
Version1: 简单串行递进式

•串行锁步：T = T_A + T_D + T_1。

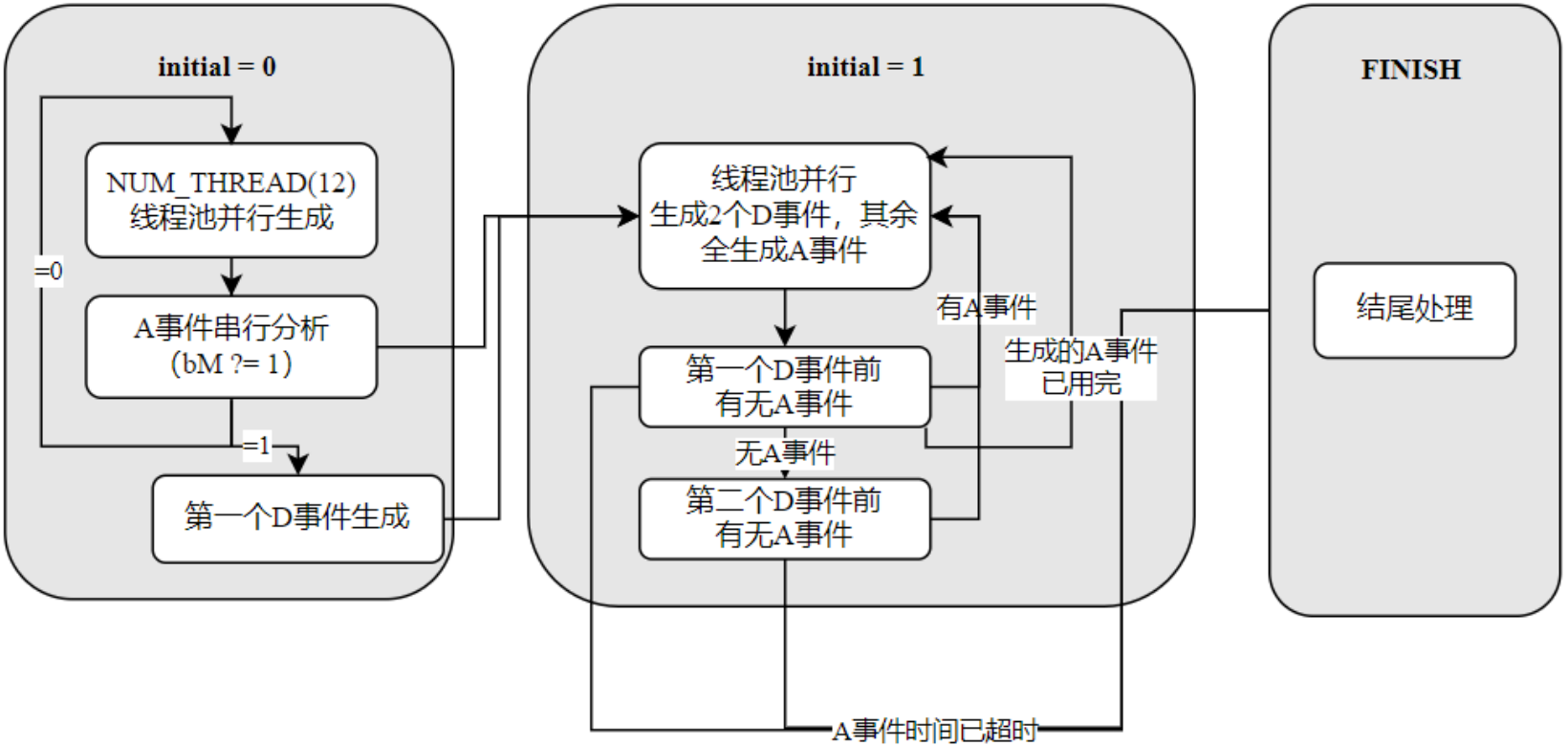


Version2: A/D事件相互并行

•A/D事件并行锁步：T = Σmax(t_A, t_D) + T_2

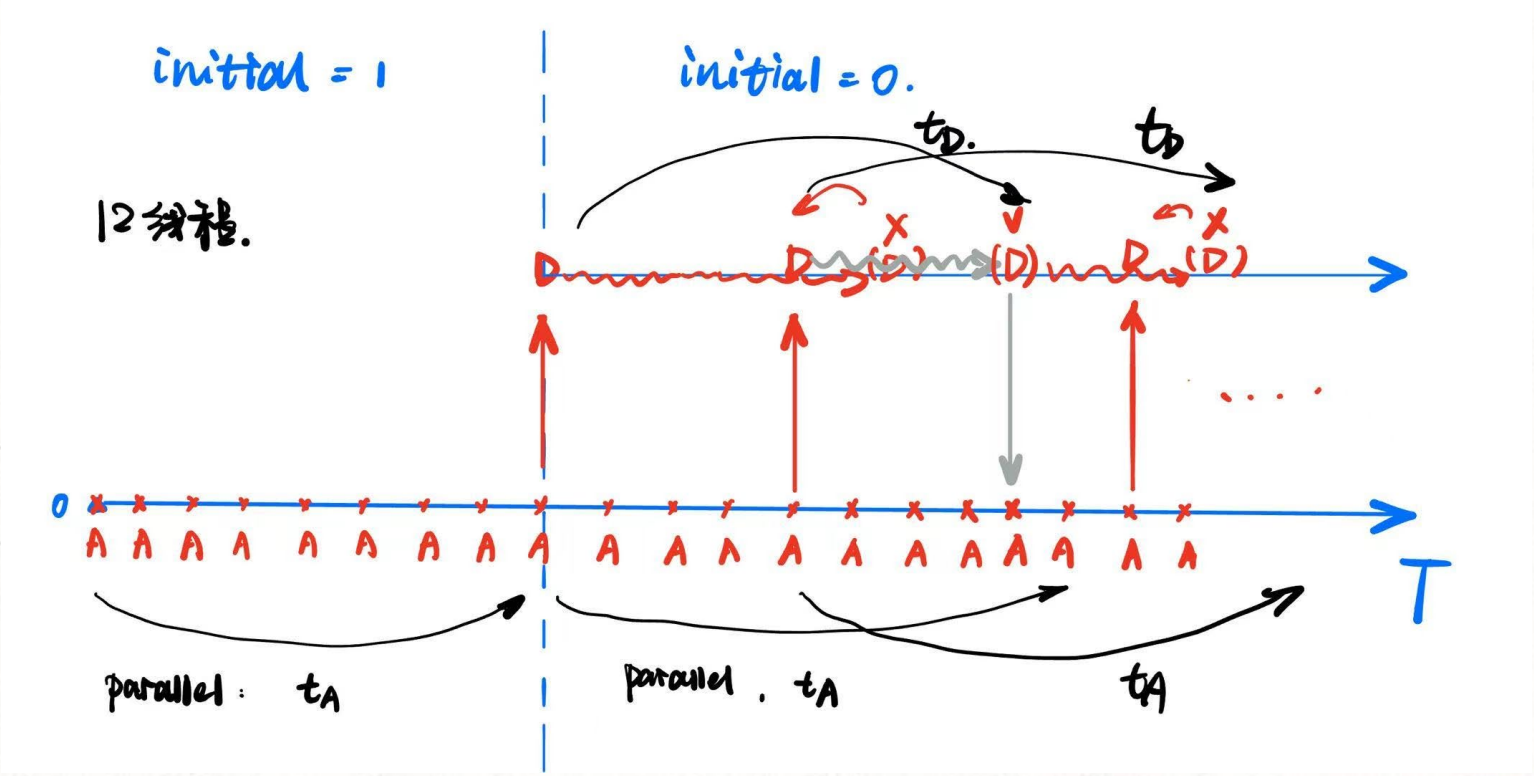


Version3: A/D事件交互动态并行



Version3: A/D事件交互动态并行

线程时间 = max(tD, tA) = max(tD_1, tD_2, tA)
线程数 = max(NUM_THREADS, 3+dT_D1/dT_A)



Version3: A/D事件交互动态并行

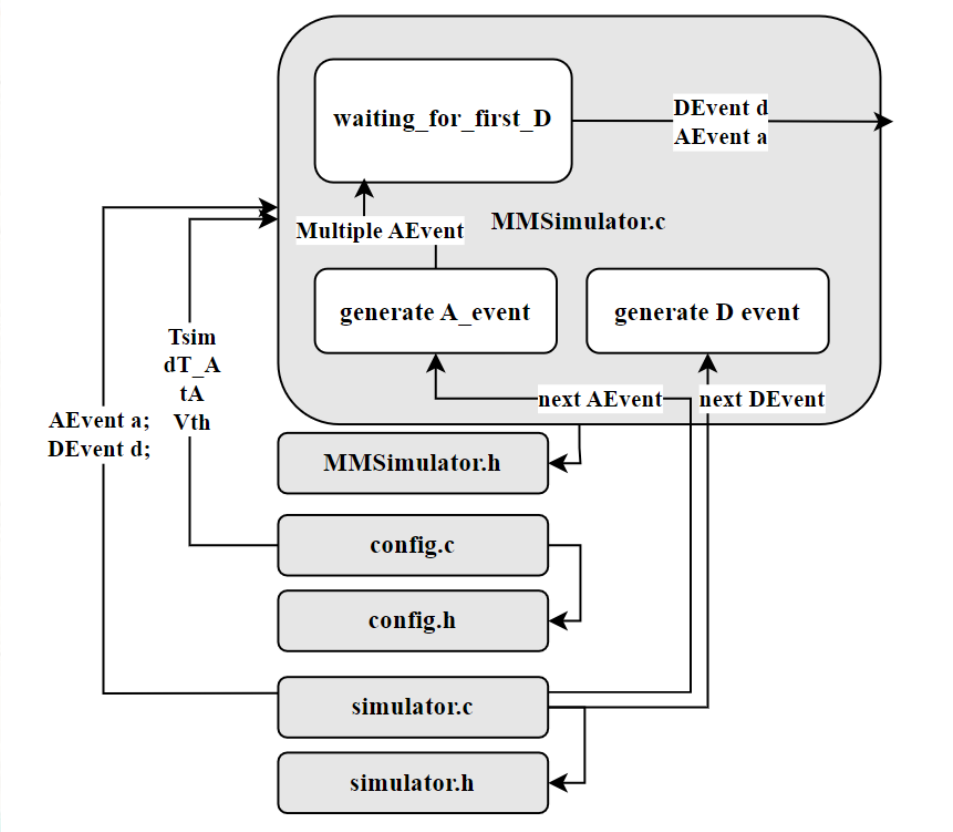
高度并行后的数据处理：
依次生成 A事件的实际T及bM、D事件的T及bM

进入逻辑分支 循环产生的A事件（分析进入不同状态）

1. 这个A的时间超出Tsim （打印所有积压的A事件）
2. 这个A时间第一次越过第一个D事件时间 （打印所有积压A事件，打印M[D]事件，改变后续A事件的时间后继续推进）
3. 这个A时间第一次越过第二个D事件时间 （打印所有积压A事件，打印M[D]事件）
4. 这个A是线程产生的最后一个A事件（清空积压A事件）
5. 这个A事件的bM=1产生了M事件 （打印所有积压A事件，打印M[A]事件）
6. 其它（无任何操作）--A事件被积压，暂不打印

→ 下次多线程任务：继续生成A/D事件

Version3: A/D事件交互动态并行



逻辑级加速

运行总时间 = 项目（强制要求）停顿时间 + **逻辑级处理时间**

优化项	优化手段	效果说明
I/O 输出优化	fprintf → fwrite	提高写入效率，减少格式开销
编译器优化	使用 -O2	提升整体执行性能
内存管理优化	减少 malloc 使用	降低内存碎片和分配开销
全局变量优化	封装、替换为局部变量	减少耦合、提升可维护性

3. 项目效果

时间停顿

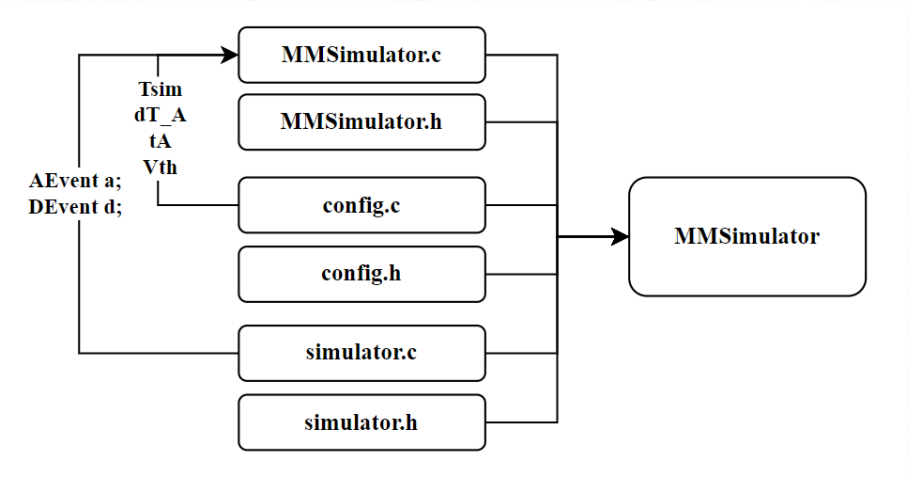
```c
void* generate_D (void* arg) {
    DThreadArgs* args = (DThreadArgs*)arg;

    usleep(args->tD * 1000);

    DEvent d_parallel = generate_next_D(d, d.T);

    pthread_mutex_lock(&lock);
    d_event_queue[args->idx] = d_parallel;
    pthread_mutex_unlock(&lock);
    free(arg);
    return NULL;
}
```

```c
void* generate_A(void* arg) {
    int index = *(int*)arg;

    usleep(tA * 1000);

    AEvent a_parallel = generate_next_A(a, a.T, Vth);

    pthread_mutex_lock(&lock);
    a_event_queue[index] = a_parallel;
    pthread_mutex_unlock(&lock);
    free(arg);
    return NULL;
}
```

编译方式及运行

使用gcc进行编译：

```
gcc -O2 config.c simulator.c MMSimulator.c -o MMSimulator -lpthread
```



使用Linux下的time对MMSimulation可执行文件计时：

```
time ./MMSimulation
```

```
eda@Ubuntu:~/eda-program/PJ2$ gcc -o2 config.c simulator.c MMSimulator.c -o MMS
mulator -lpthread
MMSimulator.c: In function ‘main’:
MMSimulator.c:215:31: warning: zero-length gnu_printf format string [-Wformat-z
ro-length]
  215 |                 fprintf(fout, "");
      |                                ^~
eda@Ubuntu:~/eda-program/PJ2$ time ./MMSimulator

real    0m4.341s
user    0m0.211s
sys     0m1.817s
```

由图所示，在该con.txt要求文件下运行时间约为4.3s

三种方法运行结果白盒测试

	白盒测试结果/s		
	Method1	Method2	Method3
dT=0.08, Vth=0.2, Tsim=1000.0, tA=8	113.3127	92.7469	49.9383
dT=0.06, Vth=0.2, Tsim=1000.0, tA=8	147.8556	110.8364	59.8554
dT=0.08, Vth=0.1, Tsim=1000.0, tA=8	114.3553	94.4532	44.5944
dT=0.08, Vth=0.03, Tsim=1000.0, tA=8	115.1708	92.3719	41.0342
dT=0.08, Vth=0.01, Tsim=1000.0, tA=8	114.7685	90.3283	39.5906

* Tsim = 1000ms下，测试时间误差约1%

4. 项目总结

结果很好 过程曲折

```
eda@Ubuntu:~/EDA/PJ2/submit/17$ time ./MMSimulator

*********TEST RESULT**********
Total Time:     0.0094 seconds

real    0m0.011s
user    0m0.002s
sys     0m0.002s
eda@Ubuntu:~/EDA/PJ2/submit/17$ cat sim_res.txt
0.000  A(0.736 0.200 0.080 0 )
0.080  A(0.416 0.200 0.080 0 )
0.160  A(0.527 0.200 0.080 0 )
0.240  A(0.625 0.200 0.080 0 )
0.320  A(0.555 0.200 0.080 0 )
0.400  A(0.897 0.200 0.080 0 )
0.480  A(0.584 0.200 0.080 0 )
0.560  A(0.892 0.200 0.080 0 )
0.640  A(0.200 0.200 0.080 1 )  D(0 0.161 -1)
1000.000  FINISH
```

成功完成了三种越来越有效的仿真方案！！
并在仿真时间上达到了比较快的程度。

改进：

1. 需要进一步探索线程的最大工作能力。
2. 线程如果回退，则消耗仍较大，有优化空间

Thank you