INFO-F404: Global vs Partitioned DM Scheduler

Raymond Lochner Najim Essakali

1 Introduction

This project deals with the study of Deadline Monotonic scheduling in two strategies, global and partitioned, on a multiprocessor system. The tasks on the systems are periodic with constrained deadlines.

2 Program description

2.1 taskGenerator

The task generator can be executing in the following form:
./taskGenerator -u <UtilizationFactor>-n <tasksNumber>-o <OutputFile>[-hp <HyperPeriod>]

- -u Total utilization of the system as an integer number
- -n Number of how many tasks the described system should have
- -o Path to the file which will include the generated system
- (Optional) -hp Soft limit for the HyperPeriod of the system

The taskGenerator creates n-random utilization values and then multiplies each randomly generated utilization value by the sum of all generated utilization values multiplied by the wanted utilization value. This gives n-random utilization values which add up to the wanted utilization factor.
Then a random period is chosen and a wcet is calculated by looking at which wcet will match the utilization the most. The deadline is then randomly chosen between the wcet and the period.
Because of the random period, the final reached utilization by the program will vary from the target utilization. The program will hence look at all tasks again and add/subtract wcet to see if the target optimization can be better matched.
The program allows to give a soft maximum value for the total lcm of the system, as multiple tasks can have a very high lcm indeed. When the soft limit is reached, new periods will only be chosen by already existing periods in the system to not increase in total lcm. The default of this is 20000.

2.2 simDM

The simulation can be executed in the following form:
./simDM [-g , -p] <tasksFile><processorsNbr>[-d]

- -p Partitioned scheduling strategy
- -g Global scheduling strategy
- <tasksFile>File describing the system
- <processorNbr>Number of processors contained in the system
- (Optional) -d Flag to show the scheduling of the system in the terminal

In both cases, global and partitioned scheduling, the simulator checks if the system is schedulable with the given parameters of the user. If yes, the possibility of a lower processor count is calculated by iterating every processor value from the total utilization of the system (rounded up) until the entered processor count is reached. The user is then informed if a lower count was found, however the displayed scheduling stays the same.
If the system was not schedulable by the parameters, the program will increase the processor count by one, starting also at the total system utilization rounded up, until a processor count is found where the system is schedulable.
To see if a system is schedulable, a simulation of the system is being undergone.

2.3 studyDM

The study comes in the following form:
./simDM <SystemFile><OutputFile>

- <SystemFile>Path to the file that describes the to be studied system
- <OutputFile>Path to the file which will contain the obtained results

The SystemFile file will accept systems in the following form:

```
Number of how many different task configurations
Number of how many different utilization configurations
Task configuration
Utilization configuration
```

For example:

```
2
2
3 6
50 100
```

Describes systems with the configurations of 3 or 6 tasks with the utilization of 50% or 100%. The study will hence calculate the required processors for the systems

```
3 tasks, 50% utilization
3 tasks, 100% utilization
6 tasks, 50% utilization
6 tasks, 100% utilization
```

in the global and partitioned strategy by doing a simulation of the system described by the values. In the output file, we can find the minimum processor count of each strategy for each configuration.
The input file

```
4
4
3 4 6 8
50 100 150 200
```

gives an output of

```
global scheduling required processors:
1 2 3 3
1 2 2 3
1 2 2 3
1 2 2 3
global scheduling distribution load:
10.1939 58.9928 32.6078 78.0392
0.150372 31.0127 91.351 98.8636
```

```
4.03372 50.2278 71.4077 98.2068
3.06372 23.4169 98.6653 52.809
partitioned scheduling required processors:
1 2 3 3
1 2 2 3
1 2 2 3
1 2 2 3
partitioned scheduling distribution load:
10.1939 49.2469 43.1069 65.8139
0.150372 44.4518 91.351 83.6887
4.03372 62.178 26.4732 36.5023
3.06372 52.6407 81.0066 84.9762
```

which represents the minimum required processors for the described system. As each system is generated with the help of the modules created in the taskGenerator program, the results will never be the same. The distribution load is calculated by adding the overflow or underflow of each utilization factor of each processor. A lower value is better as the tasks are evenly distributed. A -1 in the distribution load means that it is impossible for the system to be scheduled with the specified values.

3 Difficulties

The lack of enough C++ knowledge lead to multiple problems in the process of programming the different parts required for this project. As I have not much existing experience, everything took a lot longer to accomplish. Debugging with Valgrind and gdb were new to me, but I managed to find a memory leak that would only sometimes occur, even when given the same arguments multiple times in a row. Indeed, a logic error took multiple dozens of hours to find. Even now, during the studyDM program, sometimes the program has a memory error during the finding of the minimum number of required processes.

In the task generation, a problem occurred with too large lcm values of the period, making it impossible to undergo a simulation in reasonable time and some variables would be affected by overflowing problems. The decision was hence made to introduce a soft limit for the total lcm of the system. Further, if the soft limit is over-passed by 500000, a new system is generated.

To forbid optimization that takes too much time in order to reach the target utilization, in some parts of the program there is a counter of 50 which describes the maximum time the loop is allowed to be looped. With a higher amount of tasks, the wait is however at some systems of some seconds as the hyper period is rather large

4 Comparisons

Comparisons were made easily with the help of the programmed studyDM. Different systems are tested automatically in both the partitioned and global strategy. The input and results are shown in the tables 1 to 5.
Looking at the results, we can see that neither of the strategies is optimal as there are systems that work better in the partitioned strategy than the global strategy and vice versa. 6 tasks with 300 % for example is scheduled with only 4 processors with the partitioned system with an distribution factor of 67%, whereas the global strategy needs 5 processors and has an distribution factor of 100%.

Table 1: Comparisons input file

Tasks	5									
Utilizations	10									
Task description	3	4	6	8	10					
Utilization description	50	100	150	200	250	300	350	400	450	500

Table 2: Global Scheduling min. processors

	50%	100%	150%	200%	250%	300%	350%	400%	450%	500%
3	1	1	2	3	3	3	-1	-1	-1	-1
4	1	2	2	4	4	4	4	4	-1	-1
6	1	2	2	3	4	5	5	5	5	6
8	1	2	2	3	4	4	5	6	7	6
10	1	2	2	3	4	5	4	5	7	8

Table 3: Partitioned Scheduling min. processors

	50%	100%	150%	200%	250%	300%	350%	400%	450%	500%
3	1	1	2	3	3	3	-1	-1	-1	-1
4	1	2	2	3	3	3	4	4	-1	-1
6	1	2	2	3	4	4	5	5	5	6
8	1	2	2	3	4	4	6	5	5	7
10	1	2	2	3	4	4	4	6	7	7

Table 4: Global Scheduling utilization distribution

	50%	100%	150%	200%	250%	300%	350%	400%	450%	500%
3	13.3859	76.1537	69.8742	56.6038	150.549	224.219	-1	-1	-1	-1
4	1.48442	7.95918	93.8167	35.9375	101.347	174.157	288.746	349.091	-1	-1
6	6.41414	20.1576	95.4919	92.8231	100.847	111.929	323.654	322.987	386.419	393.549
8	11.5152	11.5334	98.6819	91.7781	90.292	199.019	193.833	133.896	204.12	373.407
10	1.0164	16.5203	88.0012	96.8188	94.5174	103.33	228.068	288.383	198.429	236.066

Table 5: Partitioned Scheduling utilization distribution

	50%	100%	150%	200%	250%	300%	350%	400%	450%	500%
3	13.3859	76.1537	69.2747	108.8	178.405	239.023	-1	-1	-1	-1
4	1.48442	19.9773	52.2468	42.7103	82.8253	117.234	75.4988	183.016	-1	-1
6	6.41414	43.1305	76.0168	78.041	67.6729	131.548	71.5828	79.6703	125.392	150.476
8	11.5152	62.3877	86.0902	101.436	83.0545	164.214	121.162	135.911	248.642	131.547
10	1.0164	73.3369	74.2903	87.7839	97.7609	142.385	236.78	139.636	148.453	231.529