Assignment 3A – Jorrit Kroes – CPS (Cache and Scheduling)

Cache analysis

I filled in the analysis with the provided excel structure:

Name: Jorrit Kroes
Student id: 11690399

Must analysis		
Node	**Before**	**After**
1	[{}, {}, {}, {}]	[{a}, {}, {}, {}]
2	[{a}, {}, {}, {}]	[{b}, {a}, {}, {}]
3	[{a}, {}, {}, {}]	[{c}, {a}, {}, {}]
4	[{}, {a}, {}, {}]	[{d}, {}, {a}, {}]
5	[{c}, {a}, {}, {}]	[{a}, {c}, {}, {}]
6	[{d}, {}, {a}, {}]	[{b}, {d}, {}, {a}]

May analysis		
Node	Before	After
1	[{}, {}, {}, {}]	[{a}, {}, {}, {}]
2	[{a}, {}, {}, {}]	[{b}, {a}, {}, {}]
3	[{a}, {c}, {}, {}]	[{c}, {a}, {}, {}]
4	[{b, c}, {a}, {}, {}]	[{d}, {b, c}, {a}, {}]
5	[{c}, {a}, {}, {}]	[{a}, {c}, {}, {}]
6	[{d}, {b, c}, {a}, {}]	[{b}, {d}, {a, c}, {}]

Scheduling analysis

T = Period (time between consecutive intervals

D=Deadline (Deadline relative to arrival times of instances)

C=Computation time

Utilization analysis

Task-set A set has a utilization of 2/8+2/16+7/20= 29/40 (=72.5 %)

Task-set B has a utilization of 2/3+1/5+1/8=119/120 (= close to 99 %)

a)

For **EDF** we can assess schedulability by first looking at the relation between period and deadline. In this case deadlines are all the same as the periods, which means there is no restriction on the upper bound for utilization (meaning its capable of reaching 100%).

We know that task combinations that are below this upperbound are always schedulable under EDF. Therefore, we can conclude that both A and B are schedulable under EDF, since as shown in the utilization calculations above, they are both lower than 100%.

b)

When assessing **RM** (Rate Monotonic) scheduling this can less easily be guaranteed.

Task set A

Since the utilisation formula U <= n(2^(1/n) – 1) has been proven for RMS and both tasks sets contain 3 tasks this means a task set of 3 with a utilization lower than 0.78 will be schedulable. Therefore we can conclude for task-set **A** that it is schedulable.

Task set B

To get to a final verdict for task-set **B** we need to look at the critical instance, meaning the situation in which all other (higher prio) tasks are ready to execute when the lowest priority task arrives. That involves waiting the absolute worst-case amount of time and only then executing. This is calculated using fixed-point iteration. Since for RM the only thing that matters for prioritization is the 'release rate' (= 1/T) so for both task sets they are already given in order from highest priority to lowest (1-3).

For task-set **B** we do the same (even if it seems difficult based on the high utilization, shorter task computation times means it might be easier to schedule).

Starting the process, we have R_1 = 2 + 0 = 2 < D1 = 3 so task 1 is schedulable.

Then for task 2, [l=1] => R_2 = 1 + [1/3] * 2 = 1 + 2 -> 3, following iterations will then be:

- [l=2] => R_2=1 + [3/3] * 2 = 1 + 2-> 3 < D2 = 5 ,so we have stabilized

For R_3 we start with C3 = 1 as well.

Then we iterate R_3(l)=1 + [R_3(l-1)/3] * 2 + [R_3(l-1)/5)]* 1

- [l=1] => 1 + [1/3] * 2 + [1/5] * 1 -> 4
- [l=2] => 1 + [4/3] * 2 + [4/5] * 1 -> 6
- [l=3] => 1 + [6/3] * 2 + [6/5] * 1 = 1 + 4 + 2-> 7
- [l=4]=> 1 + [7/3]*2 + [7/5] * 1 = 1 + 6 + 2 -> 9 > D3 = 8

- Just for good measure we do another iteration
- [l=4]=> 1 + [8/3]*2 + [8/5] * 1 = 1 + 6 + 2 -> 9 (so after 9 sec we can be sure Task 3 has been executed, but this is too late since D3 = 8)

So, we can conclude task-set **B** is not schedulable under RM