Interim Report

Milestones
Week 1 :
Complete the literature survey and study about existing work
(Neha and Xiaoqing) -**Done**

Selected papers with their contribution to our work:

A New Quality of Service Metric for Hard/Soft Real-Time Applications

Our goal is to minimize the energy consumption using DVFS while providing the best effort QoS. We were in search of a metric that would track the QoS as various decisions (viz. completing a soft task after its deadline or dropping a job) are made by our algorithm. We consider soft jobs where execution of a job after its deadline is not wasteful though there is some penalty associated with missing a deadline. There exists dependencies among tasks and the decision of dropping a job may affect varying number of other tasks. This effect also needs to be captured as we observe the QoS. This paper provides us with a metric to consider all of these factors while allowing us to tune various parameters like the penalty associated with finishing late or dropping a job, significance of finishing hard RT jobs vs soft RT jobs.

$$Q(S) = \frac{\alpha_s K_s + \alpha_h K_h}{N} - \frac{\beta}{N}\sum \frac{\delta_i}{d_i - a_i} - \frac{\gamma}{N}\sum 1_i \Delta_i$$

Kh: hard-deadline tasks and
Ks:soft-deadline tasks
N:Total tasks
αs and αh : weights for soft-deadline and hard-deadline tasks
β : penalty parameter or the tolerance factor for deadline missing of soft-deadline tasks
δ_i : difference between the task's deadline and completion time when the (soft) deadline is missed
di- ai : life time of the task
γ : penalty parameter for task dropping
Δi : number of tasks that will be affected if the i-th task is dropped(1i=1)

Frame-Based Dynamic Voltage and Frequency Scaling for a MPEG Decoder
This paper present an approach that can predict the future decoding time by maintaining a moving-average of the decoding time for I, P, B frame type based on the history. It separates the decoding process into two parts: frame-dependent (FD) and frame-independent (FI) portion. It makes predictions for the FD part which is seen to be varying and uses the optimal frequency. It uses FI part (which is nearly constant) as a buffer zone to compensate for incorrect prediction of the FD part. They compared the results with FI buffer zone and

without FI buffer zone, and conclude that better QoS was obtained in the former case. We plan to use this approach in our algorithm in some way.

Other papers listed in the references:

Week 2 :

Build vanilla scheduler simulator (Neha) - **Done**
Basic framework for simulation is in place. We input the task parameters (Period, WCET, BCET, deadline) and the set of timings corresponding to the different frequencies on Intel Xeons. Based on these values we initialize the TCB. This code generates a timeline considering the worstcase execution upto a hyperperiod. It maintains an absolute timer (abs_time) of it own that is incremented in every iteration of the simulation.

Come up with task sets that would be fed into the simulator and perform static timing analysis considering all the possible dvfs frequencies (Xiaoqing) - **Done**

Xiaoqing performed experiments to see how the task execution time varies with the frequency on Intel Xeons (Xeons because towards the end of the project we may use rapl to read power meters to get the power requirement for our scheme). We intended to feed these approximate values into our simulator. For each task we have a best case execution time (BCET) and the worst case execution time (WCET). We also intend to use a taskset with realistic release times (based on frame arrival rates) and deadlines (based on frame arrival and display rates).

F	task1		task2	
	BCET	WCET	BCET	WCET
GHz	sec	sec	sec	sec
3.3	2.002	8.001	1.0009	5.003
3.1	2.129563	8.509334	1.064999	5.321862
3	2.200975	8.676035	1.099353	5.499497
2.8	2.358575	9.30724	1.178011	5.890263
2.7	2.444442	9.758761	1.220769	6.106557
2.5	2.637789	10.526856	1.318816	6.605802
2.4	2.75143	10.776295	1.37483	6.871095
2.2	3.004155	11.965871	1.500768	7.508472
2.1	3.147915	12.54564	1.573465	7.864525
1.9	3.47816	13.843834	1.736777	8.69109
1.8	3.66875	14.634803	1.832217	9.162818
1.6	4.126418	16.374298	2.06442	10.320506
1.5	4.400673	17.501274	2.202601	10.992312
1.3	5.076952	20.218109	2.538735	12.694765
1.2	5.503854	21.861601	2.751146	13.73953

Week 3 : Design and implement QoS tracking based on the formula mentioned earlier in the paper and prediction based on history in the simulation framework (Neha)

Design and implement the reaction mechanism that makes decisions about the defensive frequency rocovery and drop/execute a job (Xiaoqing)

Week 4: Run the tasks sets developed by Xioaqing on the simulator and interpret the results (Xiaoqing, Neha)

Week 5 : Debugging and further experimentation (Xiaoqing, Neha)

Week 6 : Presentation and final report submission (Xiaoqing, Neha)

References:

- Dynamic Voltage and Frequency Scaling in Multimedia Servers, Alaa Brihi, Waltenegus Dargie, Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference
- Power-Aware CPU Management in QoS-Guaranteed Systems, Saowanee Saewong, PhD. Thesis, Carnegie Institute of Technology
- http://moss.csc.ncsu.edu/~mueller/rt/rt09/readings/projects/g5/