Energy Management in Mobile Systems

Lab 1: DPM

Andrea Floridia - 224906, Robert Margelli - 224854

Contents

1. Purpose of the lab sessions

The purpose of these lab sessions was to experience basic DPM policies by using a C++ simulator. A sample PSM was given and it was our task to implement some advanced policies and adding new features. By default, an example workload file was provided and five more were generated by us. These files were then the basis for testing the efficiency of different policies. We here analyze and present considerations and results obtained during our experiments.

2. Assignment 1

2.1 Workload generator

As a first step we implemented a workload generator by means of a C++ program (`workload_generator.cpp`). This program takes in input the number of samples to produce (i.e. the number of $T_{idle-start}$ and $T_{idle-end}$ couples) and the kind of distribution. In total five different workloads were generated according to the given requirements on the idle times distribution:

- Uniform distribution, defined on the values [1-100]us;
- Uniform distribution, defined on the values [1-400]us;
- Normal distribution, with mean value of 100us and standard deviation of 20us;
- Exponential distribution, with mean value of 50us;
- Trimodal distribution, with means of 50us, 100us, 150us and standard deviation of 10us;

The active times were instead modeled on a single uniform distribution on values [1-500]us.

Overall, the code is based on a loop that prints the idle times couples at each iteration. This is repeated for the requested number of samples. In order to recreate the above distributions the C++11 `<random>` system library [1] was used.

2.2 Obtained waveforms

The generated workload files were processed in MATLAB to reproduce their histogram representation (`hist()` command). Idle times are on the x-axis while on the y-axis the number of their occurrences. After some tests we established that 10000 samples for each workload were enough to provide a reasonable result.

- High activity uniform distribution: in this kind of distribution each idle time value is equally probable. An example PDF (probability density function) of such a distribution is shown in fig.1, along with the histogram derived from the generated workload. We can here see how the values between 1 and 100 are almost uniformly distributed, by increasing the number of samples we could achieve a higher resolution.

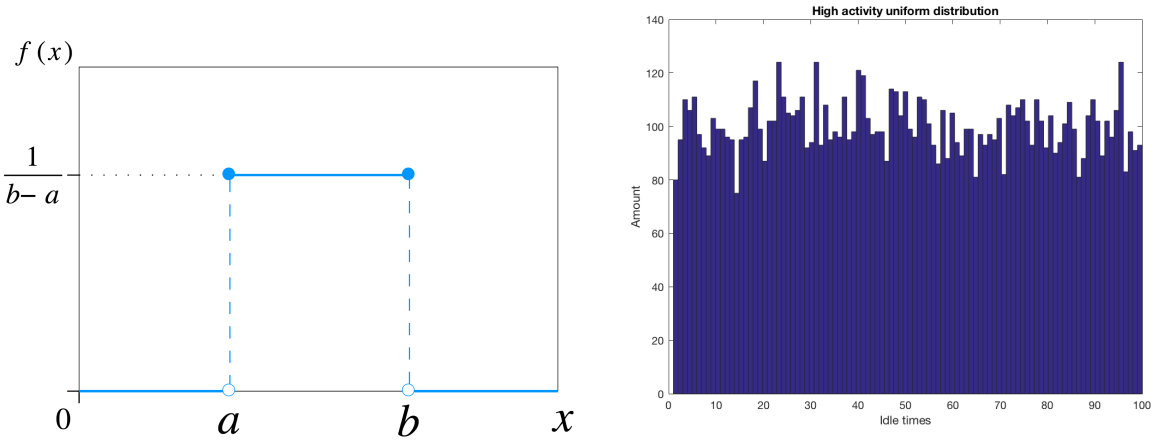

Figure 1 - PDF of a uniform distribution between a and b and histogram

- Low activity distribution: here the only difference with the above is that the idle times are spread between 1 and 400us.

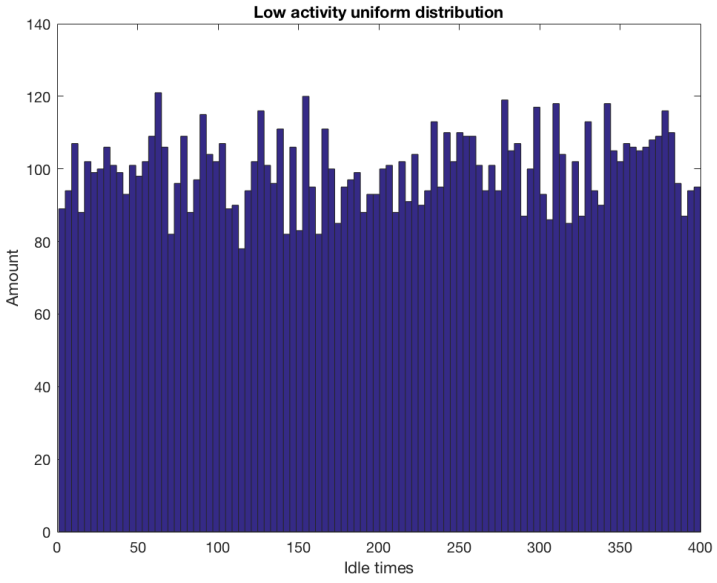

Figure 2 - Low activity uniform distribution

- Normal distribution: as the resulting *bell-shaped* histogram shows values are spread around a mean value (100 in our case).

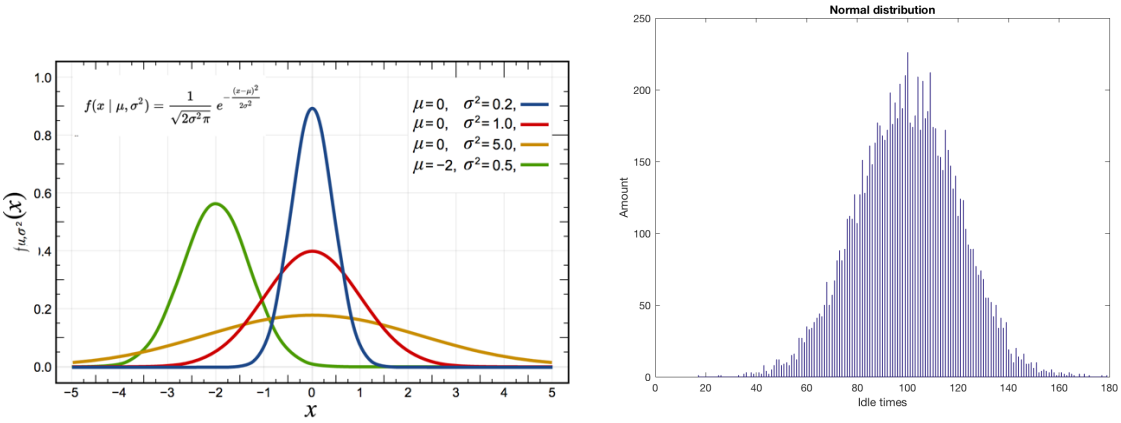

Figure 3 - PDF of a normal distribution and histogram

- Exponential distribution: given the mean value of 50us it's clear from the histogram that after such value the amount of idle times decrease drastically.

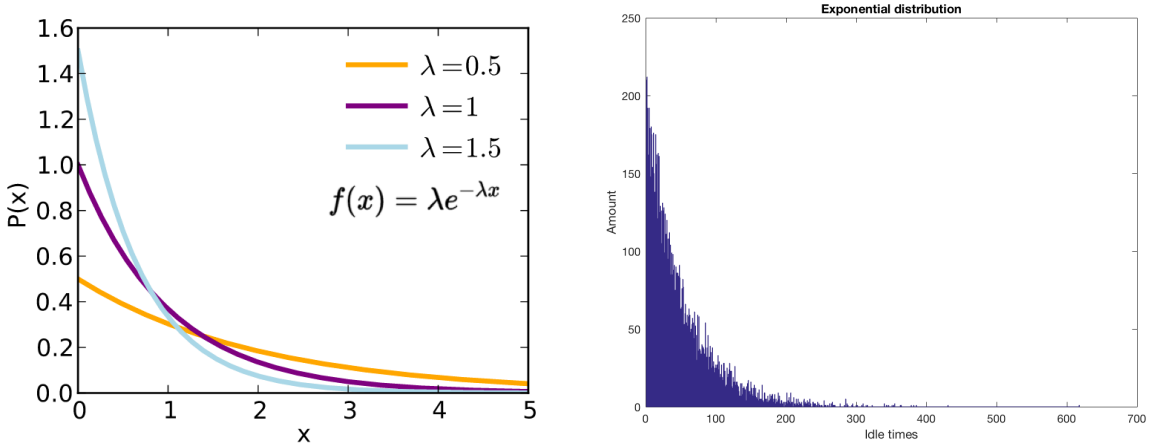

Figure 4 - PDF of exponential distribution and histogram

- Trimodal distribution: this is the result of 3 normal distributions. As expected the histogram shows 3 distinct peaks, one for each mean value (50, 100, 150).

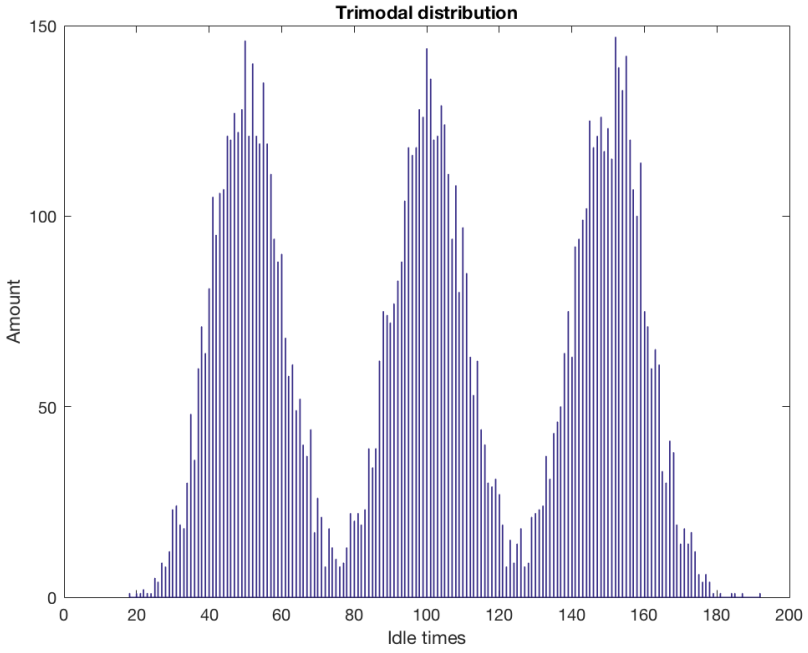

Figure 5 - Histogram of the trimodal distribution

2.3 Experiments

In order to perform a realistic analysis, we first computed the power consumed in each transition ACTIVE-IDLE (P_{tr}):

$$P_{tr} = \frac{(E_{ACTIVE-IDLE} + E_{IDLE-ACTIVE})}{T_{tr}} = 1\ W$$

Since $P_{tr} > P_{ON}$, to compute T_{BE} we have to use the following formula:

$$T_{BE} = T_{tr} + T_{tr}\left(\frac{P_{tr} - P_{ON}}{P_{ON} - P_{OFF}}\right) \approx 55\mu s$$

From the theory, we know that there are three options for setting the timeout value:

1. Greater than T_{BE}
2. Equal to T_{BE}
3. Less than T_{BE}

Table 1 summarizes the experiments performed using the three options listed above. T_{BE} represents the minimum amount of idle period which is necessary to amortize the cost of the transition (from the power consumption viewpoint). Since we are using a timeout based policy, we expect to gain in terms of energy savings when there are idle periods long enough such that (1) $T_{idle} \geq Timeout + T_{BE}$.

We chose the following timeout values:

1. 80μs (120 μs in case of the uniform distribution with a low utilization factor)
2. 55 μs
3. 25 μs

	Timeout < T_{BE}		Timeout = T_{BE}		Timeout > T_{BE}	
	Overhead[ms]	Energy Saved[%]	Overhead[ms]	Energy Saved[%]	Overhead[ms]	Energy Saved[%]
Uniform Low	19.93	23.9	18.31	19.7	15.33	11.5
Uniform High	24.92	-4.6	14.04	-4.3	5.88	-2.4
Normal	28.33	4.2	28.05	-3.1	23.39	-6.5
Exponential	19.57	-1.4	10.56	-0.8	6.54	-0.6
Trimodal	27.93	4.2	21.29	0.8	18.24	-1.9

Table 1 - Results gathered from the experiments

From Table 1 we find that the best performances are in the case of a workload with a uniform distribution (low utilization). This is reasonable given the range [1-400]us of possible idle periods available and the fact that they are equally probable. Indeed, by setting the timeout equal to 25us, we gain a considerable percentage of energy saved because there is a non-negligible amount of idle periods that fall in the range [25-400]us. Most importantly, the majority of these idle periods are long enough to satisfy relation (1) and it is worth switching to IDLE state. On the other hand, since there will be many ACTIVE-IDLE transitions, the time overhead is the worst. By increasing the timeout to 55us and then to 120us, we progressively gain in terms of time overhead and execution time while losing something from the percentage of energy saved viewpoint. Again, this can be explained through the fact that we are reducing the number of ACTIVE-IDLE transitions. This means that the systems will remain even more time in the ACTIVE state with respect to the previous experiment, reducing the effective idle periods and consequently wasting more energy.
Considering the remaining workloads, the energy savings are minimal while the timeout overhead is increasing.
Workload modelled as a uniform distribution (high utilization) yields negative results since the range of possible value is now shrunk to [1-100]us. Only a few idle periods last long enough to be fully exploitable and since their number is small, the energy wasted during the transition has a negative impact on the total energy consumption. The same reasoning applies also for the other workloads. Since idle periods seldom last more than 100us, the system saves energy only for small values of the timeout. On the other hand, it is worth noting that gaining something in terms of energy savings always leads in an increasing of the timeout overhead and so of the final execution time.
As a final consideration, given the specification of the PSM used during the experiments, it is worth adopting this kind of timeout-based policy only for workloads with long idle periods (namely the low utilization factor uniform distribution). Alternatively, it can also be used with

other workloads (always taking into account that there is a non-negligible overhead due to the timeout itself).

3. Assignment 2

We here extended the timeout policy by adding the IDLE-SLEEP transition (`new_dpm_simulator.cpp`). A new PSM file (*new_psm.txt*) was created to support it and values for the consumed energy and transition times were obtained by simply summing the values on the edges of the PSM from IDLE to ACTIVE and to SLEEP and vice-versa. Furthermore, the simulator was also modified and now takes 2 timeout values when using the timeout policy (*-t* option), one for the transition to idle timeout and the other for the transition to sleep timeout. The following code snippet shows the new policy in the `decide_state` function:

```
switch (policy){
// timeout policy
    case TIMEOUT:
    {
        if(curr_time < idle_period_start){
      // in active period
          psm_state = ACTIVE;
        }
        else if (curr_time - idle_period_start > timeout){
            /*  Lab2 */
            if (curr_time - idle_period_start > sleep_timeout)
                psm_state = SLEEP;
            else
                // after timeout point
                psm_state = IDLE;
        }
        break;
    }
```

3.1 Analysis and considerations

Again the TBE and the selected workload file play a key role in the experiments. We here present the obtained results and explain how they matched our expectations.

	Time-out < T_{BE}		Time-out = T_{BE}		Time-out > T_{BE}	
	Overhead[ms]	Energy Saved[%]	Overhead[ms]	Energy Saved[%]	Overhead[ms]	Energy Saved[%]
Uniform Low	225.780	10.6	115.060	12.6	85.920	11.2
Uniform High	24.920	-4.6	14.040	-4.3	5.880	-2.4
Normal	194.180	-8.8	28.050	-3.1	23.390	-6.5
Exponential	71.070	-5.3	11.810	-0.9	7.040	-0.6
Trimodal	194.430	-8.5	21.290	0.8	18.240	-1.9

Table 2 - Results of the experiments after introducing the SLEEP state

In analogy to paragraph 1.3, we first computed the power consumed in each transition IDLE-SLEEP-ACTIVE (P_{tr}):

$$P_{tr} = \frac{(E_{IDLE-SLEEP} + E_{SLEEP-IDLE})}{T_{tr}} = 0.19\ W$$

In this case $P_{tr} < P_{ON}$, so T_{BE} of the SLEEP state is given by:

$$T_{BE} = T_{tr} \approx 260\ \mu s$$

While the IDLE state T_{BE} remains 55us as before.
Table 2 summarizes the performed experiments. In each one both timeouts were chosen to be less, equal or greater than their respective T_{BE} value at the same time. For example, for Time-out < T_{BE} we chose $T_{TO-idle} = 25\ \mu s$ and $T_{TO-sleep} = 100\ \mu s$.

Again, we can confirm that the time overhead is inversely proportional to the given timeout. Moreover, in some cases this overhead is significant with respect to the overall profile times, as in the low activity uniform distribution where the overhead is 225.780ms while the active and idle times in the profile were 279.565ms and 220.389ms respectively.
The percentage of saved energy mostly increases with the timeout except for two cases. In fact, in the low activity uniform distribution we observe a decrease of this parameter as the timeout exceeds the T_{BE}, this is mainly due to the fact that by imposing $T_{TO-sleep} = 300\ \mu s$ we reduce the opportunity to exploit the SLEEP state and in turn the IDLE state is mostly reached during simulation, giving lower savings. For the trimodal distribution instead, the SLEEP state is never reached with $T_{TO-sleep} = T_{BE}$ or $T_{TO-sleep} > T_{BE}$ since the sleep timeout values are drastically larger than the average idle times of the workload. The decrease in energy saving is then solely due to the idle timeout which for $T_{TO-sleep} > T_{BE}$ is equal to 80μs leaving less opportunities for saving.

3.2 Comparing energy saving w.r.t. the previous simulations (no sleep state)

A direct comparison of Table 1 and Table 2 indicates that, for the five workloads and the given PSM, there is absolutely no advantage in introducing the IDLE-SLEEP transition. Mainly, this is due to the high sleep timeout values which give little to no time to stay in the SLEEP state. Only with the low activity uniform distribution there is some saving as idle times are in the range [1-400 μs] hence there is some margin for saving.
We expect to obtain a better performance with more advanced policies, especially the history-based one.

5. Assignment 3

We decided to implement three different versions of the history-based policy:

1. n-degree single variable polynomial: the equation used to predict the length of the current *i-th* idle time is based on a single independent variable, that is the previous idle time length ($T_{idle}[i-1]$). The general equation used is

$$(2)\ p(x) = p_1 x^n + p_2 x^{n-1} + p_3 x^{n-2} + \cdots + p_{n+1}$$

   Coefficients have been derived from Matlab's `polyfit` built-in function.
2. Multi-variable polynomial: two equations have been used, with both a grade equal to two but a different number of independent variables. The equations are

$$(3) p(x) = p_1 x_1^2 + p_2 x_1 x_2 + p_3 x_1 + p_4 x_2^2 + p_5 x_2 + p_6$$

$$(4) p(x) = p_1 x_1^2 + p_2 x_1 x_2 + p_3 x_1 x_3 + p_4 x_1 + p_5 x_2^2 + p_6 x_2 x_3 + p_7 x_2 + p_8 x_3^2 + p_9 x_2 + p_{10}$$

   (3) is used to predict the current *i-th* idle time based on two independent variables, that is the previous idle times $T_{idle}[i-1]$ and $T_{idle}[i-2]$, whereas (4) uses three independent variables ($T_{idle}[i-1]$, $T_{idle}[i-2]$, $T_{idle}[i-3]$).
   In this case, coefficients have been generated using a function called `polyfitn` [2].

For the proposed experiments, we decided to select four workloads among those available.
The purpose of implementing history-based policies is to improve the energy efficiency to fully exploit idle periods by predicting their length in advance. Therefore, the two required thresholds have been set equal to T_{BE} values computed using the same approach seen in the previous paragraph. In details, the T_{BE} describing the transition from active to idle and vice-versa is equal to $55\mu s$, while the T_{BE} for the transition active to sleep and sleep to active is equal to $250\mu s$.
Our first experiment is focused on showing differences between the low activity uniform distribution, the high activity one and the normal distribution using equation (3). Figure 6 collects results from the experiment.

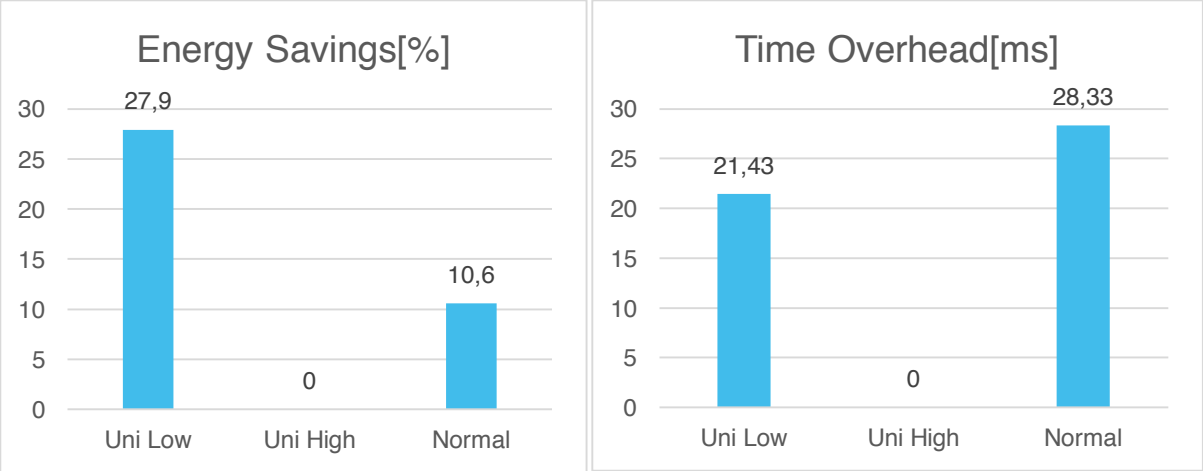

Figure 6 – First set of results of history based policy

It worth noting that in the case of a high activity uniform distribution the system never reaches the idle state. This is due to the fact that the predicted value is always less than the imposed threshold (equal to TBE). These mispredictions are caused by the distribution itself and the equation (2) used. In fact, idle periods are completely not correlated and (2) yields approximately an average value of the distribution. As a result most of the idle periods are wrongly predicted and there are no advantages in using this kind of policy. The same reasoning applies also for the low activity uniform distribution, but in this case the equation (2) generally yields an average value of 200us, which is greater than the threshold. Therefore, since idle periods last longer, there will be more energy savings. Finally, even in the normal distribution the average value is greater than the threshold. In this case the samples are strongly correlated each other, thus it is reasonable to apply a history-policy.

The second experiment uses only the workload modelled as a trimodal distribution. Here the intent is to analyze the performance by varying the number of coefficients of equation (1) (i.e. we experimented 3 different degrees) and the window sizes by using equations (3) and (4) which fix the degree to 2 and have window sizes of 2 and 3 respectively.

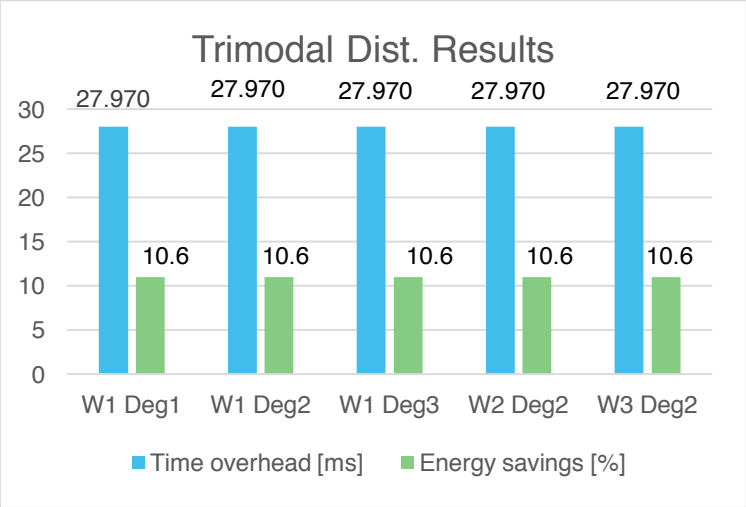

Figure 7 – Second set of results of history based policy

In this case, by trying several configurations and assuming thresholds equal to TBE values, there are no differences. This is because the predicted value is always much greater than the first threshold. For this reason, we did a third experiment, aiming at possible variations in the figures that could prove the benefits of having a higher number of windows or coefficients. We slightly modified the first threshold, setting it equal to 100 while all the other parameters remained untouched. Analyzing Table 3, it is clear that there are no advantages in using just one window (independently from the number of coefficients). In fact, samples are not enough correlated to use just one window. Looking at the third and the fourth rows, we can notice improvements in the prediction. By increasing the number of windows, we can use a larger set of past values to adjust the prediction, which yield in better performance.

	Idle time in profile[ms]	Idle time[ms]	Time overhead[ms]	Energy savings[%]
W1 g1	140.363	0	0	0
W1 g2	140.363	0	0	0
W1 g3	140.363	0	0	0
W2 g2	140.363	140.363	27.97	10.6
W3 g3	140.363	127.928	23.6	10.6

Table 3 – Final experiments with the history based policy

6. Overall comparisons and conclusions

By comparing the three types of policies implemented (two of which are timeout based), we see that the overall performance strongly depends on both the given workload and the PSM. Considering the available set of workloads, we can conclude that a timeout-based policy yields better results for a distribution with long idle periods (namely the low activity uniform distribution).

History-based policies best fit workloads with strong correlation (e.g. normal distribution, exponential distribution and partially the trimodal distribution).

It's clear how the workload drives most considerations on energy reduction in a system. In the case of single-purpose applications a profiler can be used in prototyping phases to understand what a typical workload is like. This information should then be used in order to decide a policy and implement it.

7. References

[1] http://www.cplusplus.com/reference/random/

[2] https://nl.mathworks.com/matlabcentral/fileexchange/34765polyfitn/content/PolyfitnTools/polyfitn.m