CPU Torrent – Quantitative Analysis

Initial Case with Assumptions

Let,

$$t = Total\ Time\ Required\ for\ a\ Job$$

$$Therefore,\ t = t_q + t_p$$

where,

$$t_q = Time\ spent\ waiting\ on\ the\ queue$$

$$t_p = Actual\ Processing\ Time$$

$$So,\ t_p = t_o + t_{no}$$

where,

$$t_o = Processing\ Time\ of\ offloadable\ components\ like\ Blast,\ Clustal,\ etc$$

$$t_{no} = Processing\ Time\ of\ non-offloadable\ components$$

With CPU Torrent, we can improve t by using the time spent on the queue, to process the offloadable components elsewhere. Thus, let $t_{torrent}$ be the time taken to complete an entire process with the CPU Torrent.

$$Thus,\ t_{torrent} = t_q + t_{no}\ (Assuming\ t_q > t_o)$$

$$Percentage\ Time\ Improvement,\ t_i$$

$$t_i = \frac{(t - t_{torrent})}{t} * 100$$

$$t_i = \frac{(t_p - t_{no})}{(t_q + t_p)} * 100$$

$$t_i = \frac{t_o}{(t_q + t_p)} * 100$$

From Queuing Theory, we can assume the M/M/1 Model, with

- 1 Server with,
  Service Time as $m\mu\,(\mu\,is\,the\,Service\,Time\,of\;Each\,Individual\,Node\,,m\,is\,the\,number\,of\;nodes)$
- Poisson Arrivals
- Exponential Service Time Distribution

The Average Waiting Time $\overline{W}$ is given by,

$$\overline{W}=\frac{\lambda}{m\mu(m\mu-\lambda)}$$

$$where\,,\lambda=Rate\,of\;Incoming\,Jobs$$

Thus, from the above equations, we have,

$$t_q=\frac{\lambda}{\frac{m}{t_p}(\frac{m}{t_p}-\lambda)}$$

$$t_q=\frac{t_p^{\,2}\cdot\lambda}{m(m-\lambda\cdot t_p)}$$

$$t_i=\frac{t_o}{t_p(\frac{t_p\cdot\lambda}{m(m-\lambda\cdot t_p)}+1)}*100$$

$$t_i=\frac{\alpha}{\frac{t_p\cdot\lambda+m^2-m\cdot t_p\cdot\lambda}{m^2-m\cdot t_p\cdot\lambda}}*100$$

$$t_i=\frac{\alpha\cdot m(m-t_p\cdot\lambda)}{t_p\cdot\lambda(1-m)+m^2}*100$$

$$where\,,\alpha=\frac{t_o}{t_p}\,,Percentage\,of\;the\,entire\;process\,which\,can\,be\,offloaded$$

We know that,

$m = 46$

PredictProtein received about 37000 *requests during the first* 83 *days of* 2007, *so*

$\lambda \approx 19\ requests/hour$

For Stability of the System, $\frac{m}{\lambda \cdot t_p} < 1$

Hence, for the current system, $\overline{t_p} < 2.42\ hours$

Plotting t_i *as a function of* t_p, *for different values of* α, *we get*

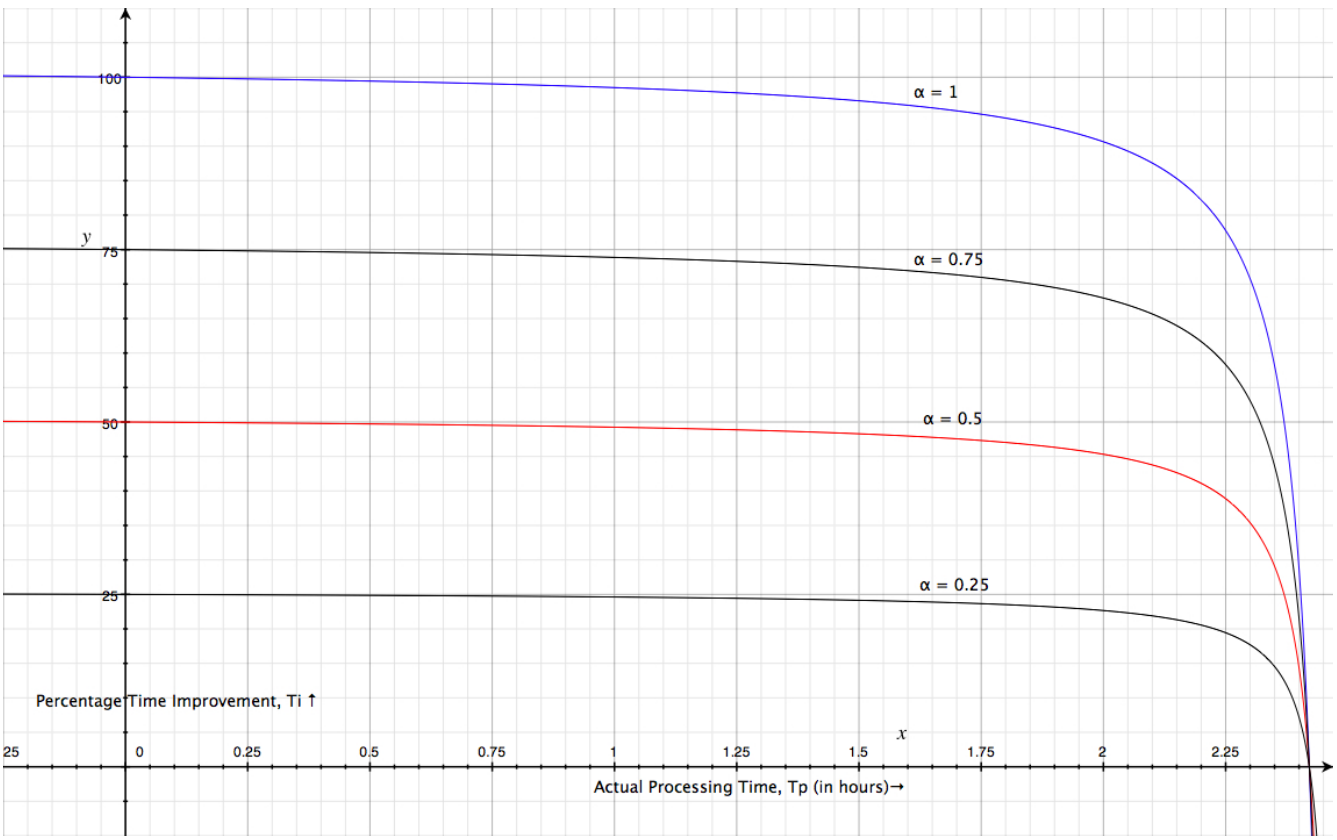

Illustration 1: Actual Processing Time, Tp vs Percentage Time Improvement, Ti

Interesting Observations

1 $t_i \propto \frac{1}{t_q}$

This basically means that as the time spent on the queue increases, the time improvement is of diminishing importance. This is even more so if $t_q \gg t_p$

e.g.
$$\begin{aligned} &If\ t_q=98, t_o=1, t_{no}=1 \\ &t_i=\frac{1}{98+(1+1)}*100 \\ &t_i=1 \end{aligned}$$

However,
$$\begin{aligned} &For\ say,\ t_q=5, t_o=20, t_{no}=30 \\ &t_i=\frac{20}{50+(20+30)}*100 \\ &t_i=20 \end{aligned}$$

2 If $t_q=0$

There have been many times during which the current PredictProtein System has had its queue empty

$$So,\ t_i=0$$

With an empty queue, there are no time improvements gained by offloading some computation, in fact, it may actually increase the time required due to the network latencies, etc. However, the load on the local system can still be reduced by the CPU Torrent at the cost of a marginally slower response time $(t_p+\delta,\ \delta=network\ delays,\ etc.)$ instead of t_p .

Let,

$$\begin{aligned} &l_o=CPU\ Load\ for\ offloadable\ components \\ &l_{no}=CPU\ Load\ for\ non-offloadable\ components \\ &l_p=CPU\ Load\ for\ the\ entire\ process \end{aligned}$$

$$Therefore,\ l_p=l_o+l_{no}$$

Let,

$$l_{torrent}=CPU\ Load\ for\ the\ entire\ process\ when\ using\ CPU\ Torrent$$

$$\begin{aligned} &l_{torrent}=l_{no}+\beta\cdot l_o \\ &where,\ \beta=Fraction\ of\ the\ offloadable\ components\ that\ we\ process\ locally \end{aligned}$$

$$Percentage\ Load\ Improvement,\ l_i$$

$$\begin{aligned} &l_i=\frac{l_p-l_{torrent}}{l_p}*100 \\ &l_i=\frac{l_o(1-\beta)}{l_o+l_{no}}*100 \end{aligned}$$

Let,

$$l_o = \alpha \cdot l_p$$

$$where\ ,\alpha = Fraction\ of\ the\ entire\ process\ which\ can\ be\ offloaded$$

$$Hence\ , l_i = \alpha(1-\beta) * 100$$

3 $\overline{t_p} < 2.42\ hours$

For the current PredictProtein system to be stable (from a Queuing Theory point of view),

$$Rate\ of\ incoming\ jobs < Total\ Service\ Rate$$

$$\lambda < m \cdot \mu$$

$$\lambda < \frac{m}{\overline{t_p}}$$

$$\overline{t_p} < \frac{m}{\lambda}$$

$$We\ know\ ,\ m = 46$$

$$Also\ ,\ PredictProtein\ received\ about\ 37000\ requests\ during\ the\ first\ 83\ days\ of\ 2007$$

$$So\ ,\ \lambda = 19\ requests / hour$$

$$Hence\ ,\ \overline{t_p} < \frac{46}{19}$$

$$\overline{t_p} < 2.42\ hours$$

$$So\ ,\ if\ \overline{t_p} > 2.42, the\ Job Queue\ will\ continuously\ increase$$

$$Thus\ ,\ the\ System\ will\ not\ be\ stable.$$