Can specialized autonomic performance managers cooperate to optimize power consumption and, at

the same time, satisfy the requirements of SLAs? This is the question examined by a group from IBM

Research in a 2007 paper [187]. The paper reports on actual experiments carried out on a set of blades

mounted on a chassis (see Figure 6.3 for the experimental setup). Extending the techniques discussed

in this report to a large-scale farm of servers poses significant problems; computational complexity is

just one of them.

Virtually all modern processors support dynamic voltage scaling (DVS) as a mechanism for energy

saving. Indeed, the energy dissipation scales quadratically with the supply voltage. The power management

controls the CPU frequency and, thus, the rate of instruction execution. For some compute-intensive

workloads the performance decreases linearly with the CPU clock frequency, whereas for others the

effect of lower clock frequency is less noticeable or nonexistent. The clock frequency of individual blades/servers is controlled by a power manager, typically implemented in the firmware; it adjusts the

clock frequency several times a second.

The approach to coordinating power and performance management in [187] is based on several ideas:

• Use a joint utility function for power and performance. The joint performance-power utility function,

Upp(R, P), is a function of the response time, R, and the power, P, and it can be of the form

Upp(R, P) = U(R) −     P or Upp(R, P) = U(R)

P

, (6.18)

withU(R) the utility function based on response time only and   a parameter to weight the influence

of the two factors, response time and power.

• Identify a minimal set of parameters to be exchanged between the two managers.

• Set up a power cap for individual systems based on the utility-optimized power management policy.

• Use a standard performance manager modified only to accept input from the power manager regarding

the frequency determined according to the power management policy. The power manager

consists of Tcl (Tool Command Language) and C programs to compute the per-server (per-blade)

power caps and send them via IPMI5 to the firmware controlling the blade power. The power manager

and the performance manager interact, but no negotiation between the two agents is involved.

• Use standard software systems. For example, use the WebSphere Extended Deployment (WXD),

middleware that supports setting performance targets for individual Web applications and for the

monitor response time, and periodically recompute the resource allocation parameters to meet the

targets set. Use the Wide-Spectrum Stress Tool from the IBM Web Services Toolkit as a workload

generator.

For practical reasons the utility function was expressed in terms of nc, the number of clients, and pκ ,

the powercap, as in

U

(pκ , nc) = Upp(R(pκ , nc), P(pκ , nc)). (6.19)

The optimal powercap popt

κ is a function of the workload intensity expressed by the number of

clients, nc,

popt

κ (nc) = argmaxU

(pκ , nc). (6.20)

The hardware devices used for these experiments were the Goldensbridge blades each with an Intel

Xeon processor running at 3GHzwith 1GBof level 2 cache and 2GB of DRAM and with hyperthreading

enabled. A blade could serve 30 to 40 clients with a response time at or better than a 1,000 msec limit.

When pk is lower than 80 Watts, the processor runs at its lowest frequency, 375 MHz, whereas for pk

at or larger than 110 Watts, the processor runs at its highest frequency, 3 GHz.

Three types of experiments were conducted: (i) with the power management turned off; (ii) when

the dependence of the power consumption and the response time were determined through a set of

exhaustive experiments; and (iii) when the dependency of the powercap pκ on nc was derived via

reinforcement-learning models.

The second type of experiment led to the conclusion that both the response time and the power

consumed are nonlinear functions of the powercap, pκ , and the number of clients, nc; more specifically,

the conclusions of these experiments are:

• At a low load the response time is well below the target of 1,000 msec.

• At medium and high loads the response time decreases rapidly when pk increases from 80 to

110 watts.

• For a given value of the powercap, the consumed power increases rapidly as the load increases.

The machine learning algorithm used for the third type of experiment was based on the Hybrid

Reinforcement Learning algorithm described in [349]. In the experiments using the machine learning

model, the powercap required to achieve a response time lower than 1,000 msec for a given number of

clients was the lowest when   = 0.05 and the first utility function given by Eq. (6.18) was used. For

example, when nc = 50, then pκ = 109 Watts when   = 0.05, whereas pκ = 120 when   = 0.01.