SEVENTH FRAMEWORK PROGRAMME

GreenDataNet

D2.6 - Forecasting Tool for IT Energy Consumption

Final
EPFL
Rev 0.4
EPFL: Pablo Garcia, Ali Pahlevan, David Atienza

TABLE OF CONTENTS

REVISION SHEET

Revision Number	Date	Brief summary of changes
Rev 0.1	26/10/2015	Draft
Rev 0.2	11/01/2016	Adding content
Rev 0.3	13/01/2016	Adding content
Rev 0.4	14/01/2016	Minor corrections

KEY REFERENCES AND SUPPORTING DOCUMENTATIONS

[1] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi. "Clearing the clouds: a study of emerging scale-out workloads on modern hardware," in ACM SIGARCH Computer Architecture News, vol. 40, no. 1, pp. 37-48. ACM, 2012.

[2] D. Meisner, C. M. Sadler, L. A. Barroso, W.-D. Weber, and T. F. Wenisch, "Power management of online data-intensive services," in Computer Architecture (ISCA), 2011 38th Annual International Symposium on, pp. 319-330. IEEE, 2011.

[3] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," ACM SIGCOMM Computer Communication Review 40, no. 1 (2010): 92-99.

[4] D. Wang, B. Ganesh, N. Tuaycharoen, K. Baynes, A. Jaleel, and B. Jacob,"DRAMsim: a memory system simulator," ACM SIGARCH Computer Architecture News 33, no. 4 (2005): 100-107.

[5] Micron's system power calculators, [online available] http://www.micron.com/products/support/power-calc.

[6] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan, "Full-system power analysis and modeling for server environments," in Proceedings of Workshop on Modeling, Benchmarking, and Simulation, pp. 70-77. 2006.

[7] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A Comparison of High-Level Full-System Power Models," HotPower 8 (2008): 3-3.

[8] M. Pedram and I, Hwang,"Power and performance modeling in a virtualized server system," in Parallel Processing Workshops (ICPPW), 2010 39th International Conference on, pp. 520-526. IEEE, 2010.

[9] M. K. Patterson, "The effect of data center temperature on energy efficiency," in Thermal and Thermomechanical Phenomena in Electronic Systems, 2008. ITHERM 2008. 11th Intersociety Conference on, pp. 1167-1174. IEEE, 2008.

[10] D. Shin, et al., "Energy-optimal dynamic thermal management for green computing," in Proc. ICCAD, 2009.

[11] J. Kim, et al., "Program phase-aware dynamic voltage scaling under variable computational workload and memory stall environment," in IEEE TCAD, 2011.

[12] D. Meisner, et al. "Power management of online data-intensive services," in Proc. ISCA, 2011.

[13] J. Kim, et al., "Correlation-aware virtual machine allocation for energyefficient datacenters," in Proc. DATE, 2013.

[14] R, Ayoub, et al., "Temperature-aware dynamic workload scheduling in multisocket cpu servers," in IEEE TCAD, 2011.

[15] J. L. Hellerstein, et al., "Feedback control of computing systems," in Wiley. com, 2004.

[16] Y. Okuyama, "Robust stabilization and PID control for nonlinear discretized systems on a grid pattern," in Proc. ACC, 2008.

[17] R. Ayoub, et al.,"JETC: joint energy thermal and cooling management for memory and CPU subsystems in servers," in Proc. HPCA 2011.

[18] D. Economou, et al., "Full-system power analysis and modeling for server environments," in Proc. WMBS,2006.

[19] M. Pedram and I. Hwang, "Power and performance modeling in a virtualized server system," in Proc. ICPPW,2010.

[20] W. Huang, et al., "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," in IEEE TVLSI, 2006.

[21] A. K. Coskun, et al., "Utilizing predictors for efficient thermal management in multiprocessor SoCs," in IEEE TCAD, 2009.

[22] A. Bhattacharya, et al., "The Need for Speed and Stability in Data Center Power Capping," in Proc. IGCC, 2012.

2. INTRODUCTION

2.1 DOCUMENT PURPOSE

In order to perform an efficient management of the DC, the algorithm that decides how to allocate workloads to servers uses a forecast of the IT energy consumption. In order to get an accurate forecast, we need to model not only the IT HW infrastructure of the DC, but also the workload of the servers.

This deliverable is directly related to D2.2 – Analytical Models for DC, where we showed all the equations that accurately describe the behaviour of the components of the DC. While D2.2 was more focused on the equations, D2.6 focuses more on the tool.

The associated task is Task 2.5: IT Energy Consumption Need Forecasting, where the objective is to forecast the global activity of computer servers. Depending on the load distribution between data centres, a module will forecast for each data centre the profile of computer/server energy consumption.

2.2 DEFINITION, ACRONYMS AND ABBREVATIONS

2.2.1 KEY ACRONYMS AND ABBREVATIONS

ADC	Analog-to-Digital Converter
CMP	Chip Multi-Processor
CPU	Central Processing Unit
DC	Data Centre
DTM	Dynamic Thermal Management
HPC	High Performance Computing
DVFS	Dynamic Voltage and Frequency Scaling
IT	Information Technology
NIC	Network Interface Controller
PDU	Power Distribution Unit
ePDU	Rack Power Distribution Unit
PID	Proportional-Integral-Derivative
PUE	Power Usage Effectiveness
PSU	Power Supply Unit
RC	Rack Controller, synonym of GC – GreenDataNet Controller
SW	Software
UPS	Uninterruptible Power Supply
VM	Virtual Machine

2.3 DOCUMENT OVERVIEW

This deliverable describes the forecasting tool for IT energy consumption. The first part, Sections 3 and 4, is dedicated to explain the DC workload characteristics and how the DC components are modelled. Then, in Sections 5 and 6, an example is presented where, thanks to the forecasting capabilities, we are able to optimize the fan speed in the cooling system of a server with dynamic task allocation.

3. DATACENTER WORKLOAD CHARACTERISTICS

Many types of applications are running on datacenters, ranging from high-performance computing (HPC) to large scale services, e.g., web search, streaming service, etc. Recently, due to the big advancements on cloud service providers (e.g., Amazon, Microsoft, Google, etc.), it becomes easier to deploy large-scale services, which leads to the drastic increase on servers hosting large-scale applications. The common characteristics of the large-scale services are that they are unprecedentedly parallel as it uses big chunk of data by splitting into small chunks. Figure 3.1 illustrates the overall operation which manipulates big chunk of dataset. In [1], Ferdman et al., examined applications running on today's clouds and presented top 6 most commonly found applications as follows:

- *Data serving*: serving as the backing store for large-scale web applications, e.g., Facebook inbox, Google Earth, etc.

- *MapReduce*: large-scale data analysis by first performing filtering and transformation of the data (namely, map procedure) and then aggregate the results (namely, reduce procedure)
- *Media streaming*: streaming services by packetizing and transmitting media files ranging from megabytes to gigabytes
- *SAT solver*: large-scale computations for solving complex algorithms, e.g., symbolic execution
- *Web frontend*: web services which schedule independent client requests across a large number of stateless web servers
- *Web search*: web search engines such as those powering Google and Microsoft Bing, which indexes terabytes of data obtained from online sources.

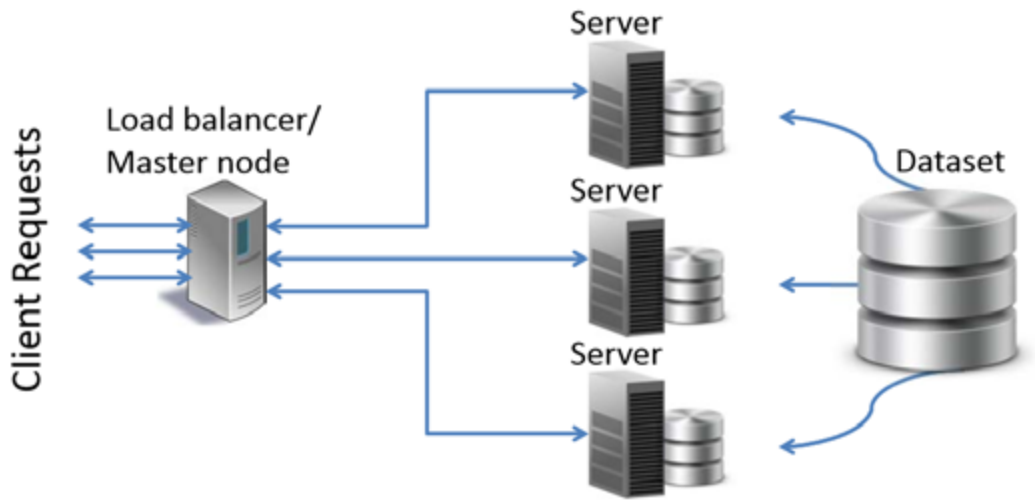

Figure 3.1 An example of scale-out applications

Up to now, most of the control solutions have been developed by targeting HPC workload characteristics. However, the workload characteristics of such large-scale applications are quite different from traditional HPC applications in both macroscopic and microscopic scales [1], which mandates us to develop the control solutions for the large-scale applications.

In a macroscopic scale, the application, first, is user-interactive, thereby, the amount of required computing capacity is highly variable and fast-changing [2] due to the dependence with external factors, i.e., number of clients/queries, etc. The characteristics of the workload traffic are well analyzed in [3]. In the coarse-grained time interval (few tens of minutes to hours), the characteristics of users' requests are distinctly different over time while the global pattern has a strong correlation with adjacent time periods as well as the same period in different days.

4. SERVER POWER MODELING

A server consists of various components, i.e., CPU, DRAM, disk, network interface (NIC), etc. As presented in Figure 4.1, a vast amount of the power is consumed by CPUs, memory, and disk, i.e., more than 70%. Extensive works have been presented to accurately model power consumption of each component: McPAT is a micro-architectural power model for chip multiprocessor (CMP), including in-order and out-of-order processor cores, networks-on-chips, shared caches, integrated memory controllers, and multiple-domain clocking, while taking into account various process characteristics, e.g., bulk CMOS, SOI, and double-gate transistors, based on the forecast in the ITRS roadmap. The accuracy is validated using various processor implementations, i.e.,

Niagara, Niagara2, Alpha 21364, and Xeon Tulsa, whose errors range 10.84~22.61%, compared to the measured values. DRAMSim [4] and Micron's System Power Calculator [5] provide accurate and detailed timing and power models of various types of DRAM, e.g., DDR, DDR2. DDR3, Mobile LPDRAM, etc., accounting for the operations.

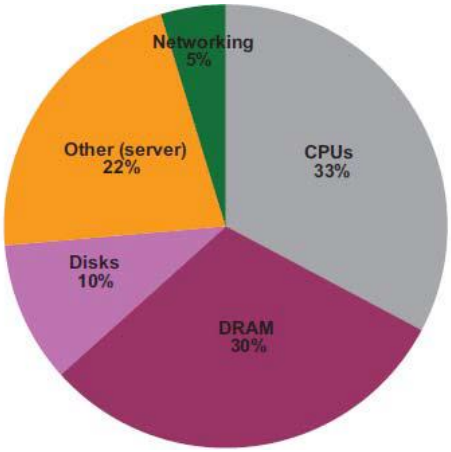

Figure 4.1 - Server power breakdown

Although such accurate power models exist to model individual component of servers, it is difficult to use all such accurate models together due to the speed of the simulation. It becomes more exacerbated when we target to simulate the large number of servers in datacenters. Thus, high-level power models are widely used to track and estimate the power consumption of servers based on the observation that the power consumption for a given server is highly correlated with distinctive workload characteristics, e.g., CPU-, memory-, or disk-intensive, stressed on servers. To capture the relationship, various works have presented high-level power model which estimates the power consumption based on the utilizations [6]–[8]. Among them, Economous et al. [6] present a linear regression power model which estimates the server power consumption with respect to utilizations of CPU (u_{cpu}), memory (u_{mem}), and disk (u_{disk}), and network interface (u_{net}) as follows.

(2) Pserver = C0 + C1ucpu + C2umem + C3udisk + C4unet

where {C0, C1, C2, C3} is a set of fitting parameters, which varies according to the target server system. This model is validated through two types of servers: 1) blade servers containing 2.2GHz AMD Turion processor, 512MB SDRAM, 40GB HDD, 10/100MBit Ethernet and 2) Itanium servers containing 4 Itanium2 chips, 1GB DDR, 36GB HDD, 10/100MBit Ethernet. According to their evaluations, the errors are within 10% in most of test cases using various benchmark suites, i.e., SPECcpu200, SPECjbb2000, SPECweb2005. Further evaluations for developing the high-level server power modeling have been conducted in [7] by comparing five different forms of power models as follows:

(3) Type1 : Pserver = C0

(4) Type2 : Pserver = C0 + C1ucpu

(5) Type3 : Pserver = C0 + C1urcpu

(6) Type4 : Pserver = C0 + C1ucpu + C2udisk

(7) Type5 : Pserver = C0 + C1ucpu + C2umem + C3udisk + C4unet

Type 1 modes the power consumption in a static value. Type 2 and 3 model the power consumption with respect to CPU utilization, i.e., ucpu, in linear and nonlinear manners, respectively. Type 3 and 4 add additional term to take into account the variations caused by disk (udisk), memory (umem), and network (unet). It concludes that Type 2 power model is enough for modeling CPU-intensive workload while Type 5 power model, using both of OS-reported component utilizations and CPU performance counters, is needed to cover broad workload characteristics, i.e., memory- and disk-intensive workloads, and aggressively power-managed servers.

In [8], Pedram et al. further enhance the accuracy of the power model by adjusting the fitting parameters according to various operating voltage and frequency and the number of active cores. It used Intel Xeon E5410

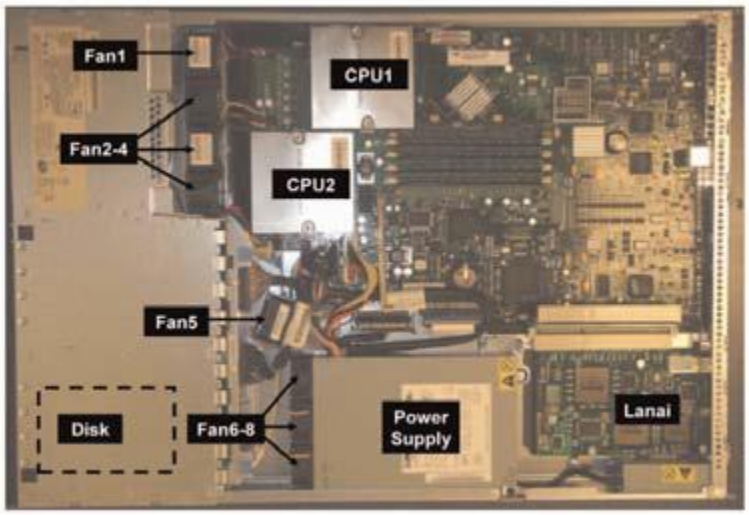

Fig. 4.2. Layout of IBM X335 server

processor for the validation with various test cases, i.e., combination of the number of active cores and operating voltage and frequency level. Recently, Joulemeter is provided to automatically tune the parameters in power models by measuring battery usage in laptop or measuring power consumption in servers.

Fans also consume significant amount of power in servers. Indeed, it is well known that the fan power consumption has a cubic relationship with fan speed [9], as follows:

(8) $$P_{fan} = C_0 + C_1 s_{fan}^3$$

where $\{C_0;C_1\}$ is a set of fitting parameters and s_{fan} represents fan speed. Thus, lowering the fan speed enables us to reduce drastic amount of power consumption, as we will see during the example of this deliverable.

5. EXAMPLE OF THE FORECASTING TOOL

Among various solutions to reduce the power consumption of computing servers, a variable fan speed control scheme is promising as it can save a significant amount of energy consumption by lowering the fan speed, as the power consumed by fans has a cubic relationship with fan speed, i.e., $P_{fan} \alpha s^3_{fan}$ [10].

In addition to the foregoing stability challenges with variable fan speed controller, a more important challenge is arising now due to the existence of multiple local controllers in enterprise servers, e.g., CPU power management via dynamic voltage and frequency scaling (DVFS) and power gating [11], [12], and temperature-aware workload scheduling in the operating system (OS) [13], [14]. As an example, thermal designers use CPU temperatures as an input to fan control algorithm to keep these temperatures inside a comfort zone window for reliability assurance. Meanwhile, processor designers implement P-state power management for the CPUs with CPU temperatures as input variables to achieve thermal capping. At the same time, OS developers work on intelligent thermally-aware workload scheduling to control the CPU temperature. If two or all three of these local controllers are active simultaneously in future servers, dynamic instability can (and most certainly will) ensue.

In this context, we present a global control scheme for servers equipped with a variable fan speed control which assures stability while jointly optimizing the performance and the power consumption. In the next sections we explain the new fan controller design, which is robust to non-ideal temperature measurement effects; then, we detail the global coordinating solution and, finally, we validate the proposed control scheme. For this approach to be successful, we need to correctly predict the IT energy consumption (server load), as well as the evolution of the temperature.

5.1 DESIGNING A SERVER FAN CONTROLLER

In order to make the control solution stable accounting for the non-ideal effects, we present a fan speed control solution based on Proportional-Integral-Derivative (PID) control [15] that is robust with respect to the challenges of signal quantization and system bus latencies. PID is one of the most widely used control solutions due to its simplicity while guaranteeing stability, accuracy, settling time, and overshoot (SASO) criterion by simply adjusting PID parameters. However, simply applying the PID control solution is not sufficient to a variable fan speed control as the relationship between temperature and fan speed is highly nonlinear and vulnerable to the measurement quantization [16].

Multiple control knobs in servers are jointly manipulated to achieve further power savings while satisfying thermal limits. In [17], [14], they present proactive thermal-aware workload management solutions to distribute the workload among cores while minimizing the cooling energy costs of fan subsystems. They compare the effectiveness of multiple control actions, i.e., ratio of temperature reduction to energy increase against multiple control actions (e.g., migrating workload vs. increasing fan speed when a thermal emergency happens), and then, select the control action that yields the best efficiency. However, it can lead to huge performance degradation as it does not take into account the impact to the performance degradation because each controller locally decides the action.

5.1.1 SYSTEM OVERVIEW

Figure 5.1 shows the target system and its controller. We focus on the computing and the cooling subsystems of servers, i.e., CPU and fan. For the sake of simplicity, we target a server consisting of Ncore cores assuming that running workloads are perfectly balanced among the cores, which implies that multiple fans in a server run at the same speed. Temperature sensors are located at each core and deliver measured

temperatures to the dynamic thermal management (DTM) unit, quantized and time-lagged, due to the underlying signal acquisition standards (i.e., Analog-to-Digital Converter (ADC) and I2C bus). The target server model is developed on the basis of a presently shipping enterprise server. All temperature sensors in the server use an 8-bit ADC. According to our measurements, the time lag on the temperature measurement in this system amounts to 10 sec.

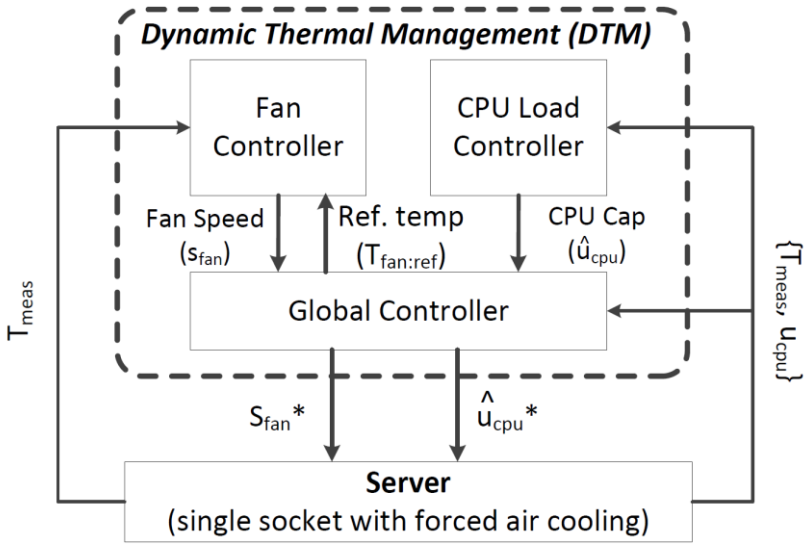

Figure 5.1 - Proposed target computing/server system

The role of the proposed controller is to jointly determine the optimal fan speed, i.e., s^*_{fan}, and maximum allowable CPU utilization (so called CPU cap), i.e., $\hat{u}^*_{cpu}$, so as to maintain the operating temperature of CPU within a safe operating region, e.g., <80°C, while minimizing performance degradation which happens when the required performance level is higher than the CPU cap. The controller largely consists of two parts: 1) multiple local and 2) a global controllers. In the target architecture, we have two local controllers, namely fan speed (s_{fan}) and CPU cap ($\hat{u}_{cpu}$) controllers. In this work, we focus on developing a stable fan controller robust to the non-ideal temperature measurement while simply using low-complexity CPU capper using a deadzone-like scheme. In a deadzone-like CPU capper, there are two threshold values, i.e., T^{low}_{th} and T^{high}_{th}. The CPU cap, i.e., $\hat{u}_{cpu}$, is only increased when the measured temperature, Tmeas, is higher than T^{high}_{th} while lowering $\hat{u}_{cpu}$ when Tmeas is lower than T^{low}_{th}. Regarding
the fan speed controller, we adopt a PID control scheme to make the junction temperature track a reference trajectory, i.e., $T^{ref}_{fan}(t)$. The independent local decisions are fed into a global controller which gives an optimal control action by coordinating the local decisions. We explain the designs of the local fan speed and the global controllers in the next sections.

5.1.2 POWER AND TEMPERATURE MODELING

Following the server model described in Section 4, the server's total power consumption (Ptot) can be modeled as the sum of CPUs (P_{cpu}) and fans (P_{fan}) power consumptions, i.e., $P_{tot} = P_{cpu} + P_{fan}$. P_{cpu} is proportional to CPU utilization ($u_{cpu} \in [0; 1]$) and modeled as follows [18], [19]:

(1) $$P_{cpu} = P^{static}_{cpu} + P^{dyn}_{cpu} * u_{cpu}$$

where P^{static}_{cpu} and P^{dyn}_{cpu} are the static and the maximum dynamic power consumption of CPU. P_{fan} has a cubic relationship with the fan speed, i.e., $P_{fan} \alpha S^3_{fan}$.

CPU	P_{max}	160W
	P_{idle}	96W
	Die thermal time constant	0.1 sec
Fan	Fan power per socket	29.4W
	Max fan speed per socket	8500rpm
	Fan sample interval	1 sec
	Heat sink thermal resistance in K/W	$R_{hs} = 0.141 + \frac{132.51}{V^{0.923}}$ V: fan speed (rpm)
	Heat sink thermal time constant at max air flow	60 sec

Table 5.1 - Design parameters used in power and temperature modeling

The temperature of the target system can be modeled using a well known duality between thermal and electrical phenomena [20]. Hence, we use the temperature model presented in [17]. Using the model, the temperature of the heat sink at the time (t + Δt), i.e., $T_{hs}(t + \Delta t)$, is calculated as follows:

(2) $$T_{hs}(t + \Delta t) = T^{ss}_{hs} + \left(T_{hs}(t) - T^{ss}_{hs}\right) \cdot e^{-\frac{\Delta t}{R_{hs} C_{hs}}}$$

where R_{hs} and C_{hs} represent the heat sink thermal resistance and capacitance. R_{hs} is inversely proportional to the fan speed. T^{ss}_{hs} is the steady-state T_{hs} which is calculated as follows:

(3) $$T^{ss}_{hs} = T_{amb} + R_{hs} * P_{cpu}$$

where T_{amb} is the ambient temperature. The thermal time constant of the heat sink is much larger than that of the CPU die, even at the highest fan speed (i.e., smallest heat sink time constant). Thus, we can calculate the junction temperature of the CPU die, T_j by solving the differential equation for the thermal RC network assuming that T_{hs} is constant. Table 5.1 summarizes the parameters and corresponding values for modeling the power and the temperature in the target system.

5.1.3 ROBUST FAN SPEED CONTROLLER DESIGN

In this section, we present a robust fan speed controller design based on a PID control scheme which assures control stability even under the non-ideal effects associated with the temperature measurement subsystem. We enhance a conventional PID-based fan speed controller to make it resilient to the time lag and the quantization error while reducing server performance degradation:

A set of PID parameters (KP , KI , and KD, the coefficients of proportional, integral and derivative gains, respectively) obtained by careful tuning is only optimal for a linear target system. However, the target system (temperature and the fan speed relationship) is nonlinear as the thermal resistance also varies nonlinearly with respect to fan speed as shown in Table 5.1. Thus, a set of PID parameters obtained in one fan speed region may not work well in other fan speed regions.

In order to solve the problem of using a single set of PID parameters in a target server system, we introduce a new adaptive PID control scheme which dynamically adjusts the set of the parameters according to the operating fan speed: First, we obtain the sets of parameters in multiple fan speed regions. Note that the number of regions depends on the error of the piecewise linearization. In our work, two regions, i.e., 2000 and 6000 rpm, are enough to linearize the relationship within 5% error for the considered enterprise server systems. Then, at runtime, a set of PID parameters is adjusted according to the measurement time latency and the operating fan speed at every fan speed decision period.

Finally, to eliminate the oscillation caused by the quantization of the temperature measurement, we propose a quantization elimination scheme that enforces no change in fan speed (i.e., sfan) when the temperature measurement error is less than the size of the quantization step.

5.1.3.1 GLOBAL CONTROLLER

A global control scheme coordinates multiple local control actions to guarantee server system operation stability while jointly minimizing the performance degradation and the energy consumption.

A. Rule-based global coordination approach

In order to guarantee the system stability when multiple local controllers are running together in a system, we propose a global control scheme that dynamically selects only one control action at a time affecting the system because the stability of each controller has been proven at the local controller design. The suitable selection among multiple local control actions varies according to operating conditions. To deal with the issue, we propose a rule-based coordination scheme as presented in Table 5.2, which adjusts the control variables, i.e., $\{\hat{u}_{cpu}; s_{fan}\}$, by considering performance as the primary system behavior concern.

		Fan speed		
		$s_{fan}(k+1) < s_{fan}(k)$	$s_{fan}(k+1) = s_{fan}(k)$	$s_{fan}(k+1) > s_{fan}(k)$
CPU cap	$u_{cpu}(k+1) < u_{cpu}(k)$	$s_{fan} \downarrow$	$u_{cpu} \downarrow$	$s_{fan} \uparrow$
	$u_{cpu}(k+1) = u_{cpu}(k)$	$s_{fan} \downarrow$	-	$s_{fan} \uparrow$
	$u_{cpu}(k+1) > u_{cpu}(k)$	$u_{cpu} \uparrow$	$u_{cpu} \uparrow$	$s_{fan} \uparrow$

Table 5.2 – A rule-based coordination

B. Predictive adjustment of the set-point temperature of fan controllers

The performance degradation can be reduced by lowering the reference temperature of a fan controller ($T^{fan}ref$) because it makes the CPU junction temperature lowered by setting the fan speed higher while it increases the power consumed by fans. Thus, we need a solution to judiciously adjust $T^{fan}ref$ to jointly reduce both performance degradation and power consumption. Our studies outline two observations:

- When CPU utilization is low, attenuate T^{fan}ref to cope with any unexpected abrupt increase of the CPU utilization.
- When CPU utilization is high, amplify T^{fan}ref to lower the temperature increase caused by the unexpected increase on CPU utilization.

Based on the observations above, we linearly scale T^{fan}Ref according to the predicted CPU utilization. Furthermore, in order to filter out the noise term in the CPU utilization, we used a moving average filter for the prediction [21].

C. Single-step fan speed scaling

To further reduce the performance degradation, especially, caused by abrupt spikes on required CPU utilization, we present a single-step fan speed scaling solution which sets the fan speed to the maximum when the measured performance degradation is higher than a predefined threshold value. As analyzed in [22], the performance spike in server workloads is much faster than the settling time of controllers. In adjusting the fan speed, it takes $N^{fan}_{trans} * t^{fan}_{interval}$ where N^{fan}_{trans} is the number of decision periods until the fan speed reaches to a steady-state. Thus, the performance of the servers can be severely degraded during the long transient time period. The single-step fan speed scaling scheme can guarantee that the performance degradation during the settling time is no larger than the predefined value. Once the maximum fan speed is set, we lower the fan speed to reach the lowest possible fan speed which enables to run required CPU utilization without any temperature violation.

6. EXPERIMENTS

We developed our simulation environment to model the system characteristics of actual commercial enterprise servers. Table 5.1 summarizes the parameters used in this simulation. We used synthetic workload traces which alternates between 0.1 and 0.7 while imposing a random Gaussian noise to further validate the robustness of the propose control scheme in realistic CPU variation characteristics. Considering the normal control interval in commercial enterprise servers, we set the CPU and fan control time constants ($\Delta t^{cpu}_{control}$ and $\Delta t^{fan}_{control}$) to 1 sec and 30 sec, respectively.

First, we demonstrate the stability of the proposed control scheme under the dynamic workload scenario, and then, compare the performance and the energy consumption with the following solutions:

- w/o coordination: baseline which uses the fan speed and CPU load controllers without any coordination
- E-coord: energy-aware coordination scheme in [17]
- R-coord(@ T^{fan}ref = 75°C): rule-based coordination while using a fixed reference temperature with 75°C, for the fan speed controller
- R-coord+A-T^{fan}ref : rule-based coordination with adaptively changing the T^{fan}ref from 70 to 80°C according to the CPU utilization
- R-coord+A-Tref+SS^{fan}: additively applying the single-step fan speed control scheme

For fair comparison, we use the proposed fan speed control scheme in all solutions. Note that the system becomes unstable if we directly adopt the fan speed control scheme presented in [17] as it is designed without the long I2C delay concern.

7. RESULTS

In order to evaluate the quality of our algorithm for fan speed control, and the correctness of the forecasted information, we compare the stability, performance and energy against alternative solutions.

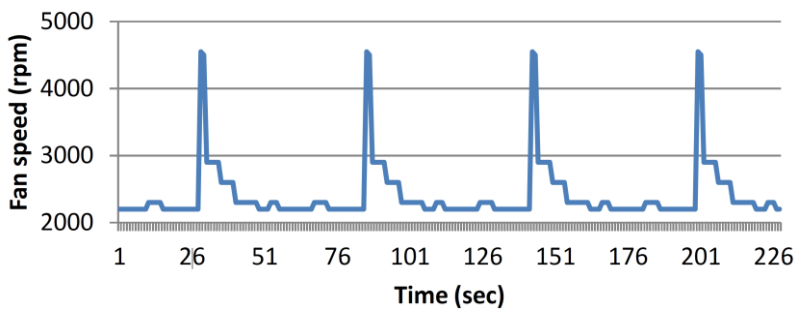

Figure 7.1 - Measured of fan speed and temperature under a stable workload

Figure 7.1 shows the measured fan speed in the target server adopting a deadzone fan speed control scheme under a fixed workload. It demonstrates that the fan speed becomes oscillatory due to the effects caused by the non-ideal temperature measurement. To validate the stability of the proposed global coordination scheme, we performed a simulation while running the proposed fan speed control scheme along with the CPU load controller. Figure 7.2 shows the varying CPU utilization (solid line and left Y-axis) and the fan

speed (dotted line and right Y-axis). As shown in the figure, even with time-varying CPU utilization, the proposed control solution provides stable fan speed control.

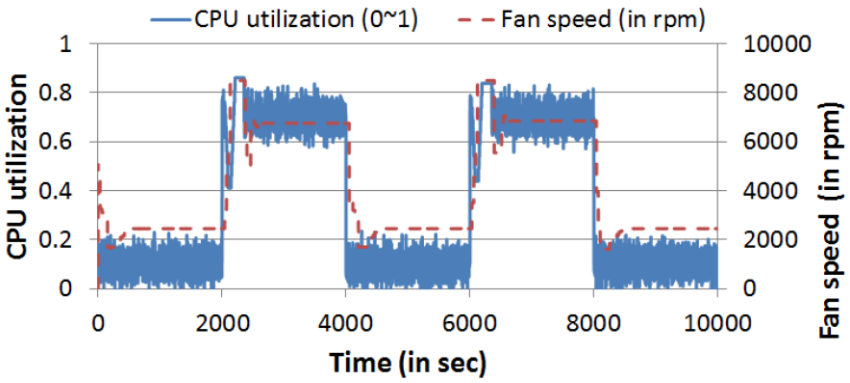

Figure 7.2 - Traces of fan speed with the dynamic CPU load and noise (standard deviation is set to 0.04).

The second column of Table 7.1 shows the performance comparisons in terms of the fraction of the deadline violations caused by the thermal emergency. As shown in the table, E-coord can cause a huge increase in the deadline violation, as it does not take into account the impact of the performance degradation when it decides the control action when a thermal emergency happens. On the contrary, R-coord(@ T^{fan}_{ref} = 75°C) can reduce the percentage of the deadline violation by up to 12% compared to the baseline scheme as the rule-based coordination scheme is designed to minimizing the performance degradation by increasing the fan speed first when multiple control actions are conflicted when the thermal emergency happens. Further improvement can be obtained by using the predictive adjustment of T^{fan}_{ref} because the scheme lowers T^{fan}_{ref} when a predicted CPU load is low, which enables to reduce the performance degradation caused by unexpected CPU load spike. The single-step fan speed scaling provides additional 4.5% reduction in the terms of the performance degradation by further reducing the performance degradation until the fan speed reaches its desired point by setting the fan speed is set to its maximum when the measured performance degradation is higher than a certain threshold value.

Solution	Deadline violation (%)	Norm. Fan energy consumption
w/o coordination (***baseline***)	26.12	1
E-coord [6]	44.44	0.703
R-coord(@ $T^{fan}_{ref} = 75°C$)	14.14	1.075
R-coord+*A*-T^{fan}_{ref}	11.42	0.801
R-coord+*A*-T^{fan}_{ref} +SS^{fan}	6.92	0.804

Table 7.1 - comparisons of the performance and the power consumption

The third column of Table 7.1 shows the energy consumed by fans when adopting four different solutions. The values are normalized with respect to the uncoordinated case. As we use the rule-based coordination scheme with a fixed T^{fan}ref , the energy consumption is slightly increased compared to the baseline case, as we set T^{fan}ref to low value in order to prevent the performance degradation caused by unexpected spike in the CPU load. The energy consumption can be reduced as we adaptively adjust T^{fan}ref according to the predicted CPU load by up to 20% as we increase T^{fan}ref when the predicted CPU load is high. The single-step fan speed scaling scheme leads to the slight increase in the energy consumption as it set the fan

speed to its maximum when the performance degradation is higher than a threshold value. E-coord can yield even 10% lower energy consumption compared to the proposed solution as it is originally designed to minimize the energy consumption. However, the performance degradation is unacceptably high.

To summarize, in this example, we have first presented a stable PID fan speed controller that is robust to non-ideal temperature effects, i.e., measurement time lag and signal quantization. Then, we have presented a global control scheme which coordinates multiple local control actions via a low-complexity rule-based management scheme while minimizing the performance degradation caused by the variable fan speed control. We have validated the proposed control scheme by developing simulation environment modeling presently shipping commercial enterprise servers. The experimental results show that, thanks to the forecasting tool for IT energy consumption, the fan speed control scheme has been successfully tuned to be robust to the long measurement time lag and the temperature quantization while reducing the performance degradation and the power consumption by up to 19.2% and 19.06%, respectively, compared to a conventional fan speed control solution.

8. CONCLUSIONS

This deliverable describes the forecasting tool for IT energy consumption. After introducing the modelling of the workload and of the DC components, an example of use is presented where the tool assists a system designer to optimize the power consumption of the servers running dynamic workloads by efficiently managing their on-board fans.