Journal of Information Engineering and Applications
ISSN 2224-5782 (print) ISSN 2225-0506 (online)
Vol.5, No.1, 2015
www.iiste.org

Self-Aware Monitoring Based Adaptive Systems

Fahad Ahmad
School of Computer Sciences, National College of Business Administration & Economics, Lahore, Pakistan
fahadahmad84@gmail.com

Abstract
Monitoring based computer system should be self-aware first because self-aware computer systems will be capable of adjusting and adapting their behavior and resources thousands of times in a second automatically to find the best way for the achievement of an assigned goal. Such an ability can benefits a wide range of computer systems from embedded systems to supercomputers and is especially helpful for meeting power, performance, and resource-metering challenges in monitoring and many other fields like mobile computing, cloud computing, multicore computing and parallel operating systems.Here a framework is being proposed, called Application Pulse-rate that offers a general, standardized approach for applications to observe their performance and provides that information to external observers. Through a deep view of a self-optimizing synchronization library called Smartlocks, a controlling technique can be determined that systems use to decide which optimization actions should be taken for monitoring of environment.

1. Introduction

Resources like number of transistors and memory, the level of incorporation and the speed of components have amplified dramatically over the years. The technologies have improved but we still continue to apply outmoded approaches to our usage of these resources. The operating systems, languages, etc. we use today, were planned for a different age. Thus this is the time for a new attitude to the way systems are designed and used. The Self-Aware computing research forces the new balance of resources to expand performance, utilization, reliability and programmability [1][2].

In this background, think about a computing system that will be able to observe its own execution and optimize its behavior according to a user's or application's needs. Consider a programming ability by which users can identify their desired goals instead of how to perform a task, along with constraints in terms of an energy budget, time, or simply an inclination for an approximate answer over an exact answer. Self-aware computer systems will be capable to configure, heal, optimize and protect themselves without the need for human intervention [3].

Capabilities that permit them to automatically find the best way to achieve a specific given goal like monitoring and detection with the resources at hand. Considering the resemblance to organic nature this new class of computer systems can be referred as Organic Computing System (OCS). In order to meet this idea of monitoring based Organic Computing System, i) Allow applications to specify their goals, ii) Allow system services to regulate whether these goals are met, and iii) Allow adaptive systems to make learned decisions among numerous possible actions.

The first two tasks can be handled by using the Application Pulse-rate framework, which gives a general interface for applications to express their performance goals and progress towards those goals [4]. Using Pulse-rate, applications can express their performance using the concept of Pulse-rate; however, the API also permits applications to express their goals as a desired Pulse-rate between Pulse-rates. The Pulse-rate within an application permits adaptive systems to decide using a direct measure of an application's performance rather than trying to infer it from underlying hardware counters. By stating application goals, the Pulse-rate framework permits adaptive systems to accomplish constraint-based optimizations like minimizing power for a minimum performance target.

The third challenge, that of making learned decisions, is to use machine learning in Monitoring Based Adaptive Systems. A method can be determined in an adaptive, self-aware spinlock library called Smartlocks [5]. Smartlocks use an adaptation called lock gaining scheduling to define the optimal policy for permitting access to a critical section within an application. Moreover, Smartlocks are capable to adapt their behavior in the response of environmental changes, like changes in clock frequency.

2. Self-Aware Monitoring Based Adaptive Systems

To attain the idea of a self-aware monitoring based system, it should be capable to monitor its behavior to update one or more of its components (hardware architecture, operating system and running applications), to attain its goals. The idea of organic computation in which an organic computer is assigned a goal and a set of resources and their availability, then it should find the best way to achieve the goal while optimizing concerned constraints which are helpful in monitoring because our resources are limited so this system can perform well within the provided resources. An organic computer has four major properties:

- It is goal oriented i.e., it takes actions automatically to meet the given application goals.
- It is adaptive i.e. it observes itself, reflects on its behavior to learn, computes the delta between the goal and observed state, and finally takes actions to optimize its behavior towards the goal.
- It is self-healing i.e., it constantly monitors for faults and continues to function through them, taking corrective action as needed.
- It uses the least amount of computation or energy to meet accuracy goals and accomplish a given task.

It is important that much like other biological organisms, an organic computer can go fine outside the traditional measures of goodness like performance and can adapt accordingly to the diverse environments and even develop itself over time. To adapt itself accordingly what the organic computer is performing or how it is performing a given task at run time, A control system is required as part of the system that observes execution, measures thresholds and compares them to goals, and then adapts the architecture, the operating system or algorithms as needed. The main task is to detect what fragments of a computer need to be adapted and to quantify the degree to which adaptation can be affordable. Examples of mechanisms that can be adapted include recent researches on memory controller [6].

3(a).Application Pulse-rate Framework

The Application Pulse-rate framework provides a simple, standardized way for applications to report their performance and goals like monitoring or detection to external observers. As shown in Figure 1, this development can be observed by either the application itself or an external system (such as the OS or another application). Development of a simple, standardized interface makes it possible for programmers to include Pulse-rate to their applications. Registration of the goals of an application with external systems enables adaptive systems to make optimization decisions at the time of monitoring the program's performance directly rather than having to suppose that performance from low-level details. If performance is found to be undesirable, then information obtained can help to decide why and what should be reformed.

The Application Pulse-rate framework measures application progress towards goals of monitoring using the idea of Pulse-rate. At substantial points, applications make an API call to indicate a Pulse-rate. The Pulse-rate API lets applications to communicate their goals by setting a target Pulse-rate (i.e., number of Pulse-rate per second). Adaptive system services, such as the operating system, runtime systems, hardware, or the application itself, monitor progress through supplementary API calls and can use these facts to alter their behavior and aid the application achieve the best monitoring performance goals.

As an example, a CamWizard application used in CCTV cameras used for security purposes can use Application Pulse-rate to identify a target throughput of 30 video frames a second [7].

Figure. 1 indicates Self-optimizing application using the Application Pulse-rate framework. Figure. 2 indicates Optimization of machine parameters by an external observer.

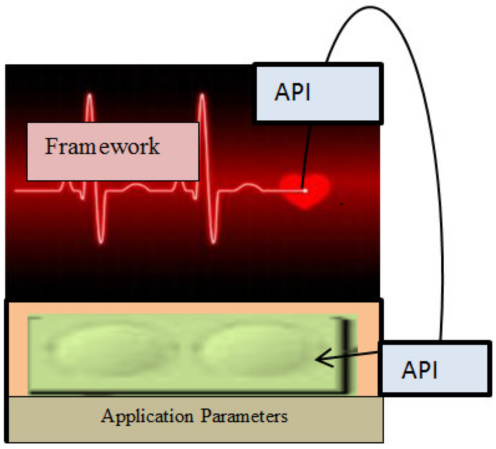

Figure 1

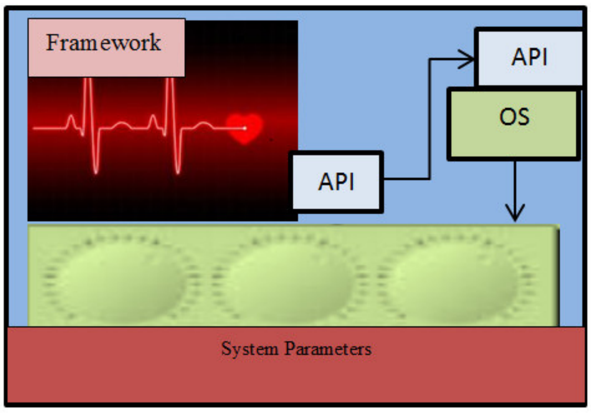

Figure 2

3(b).Smartlocks

Smartlocks is a self-aware system designed to support in reducing the programming complexity of extracting high performance from today's multicores and asymmetric multicores. Smartlocks is a self-optimizing spin-lock library that can be used as the basis for synchronization, resource sharing, or programming models in multicore software. As it runs, Smartlocks uses Application Pulse-rate together with a Machine Learning (ML) engine to

optimize the performance of monitoring application by adapting Smartlock's internal behaviors [8].
The vital adaptable behavior within Smartlocks is the lock gaining scheduling policy. Lock gaining scheduling is picking which thread or process among those waiting should get the lock next (and thus get to execute the critical section next) for the best long-term effect.

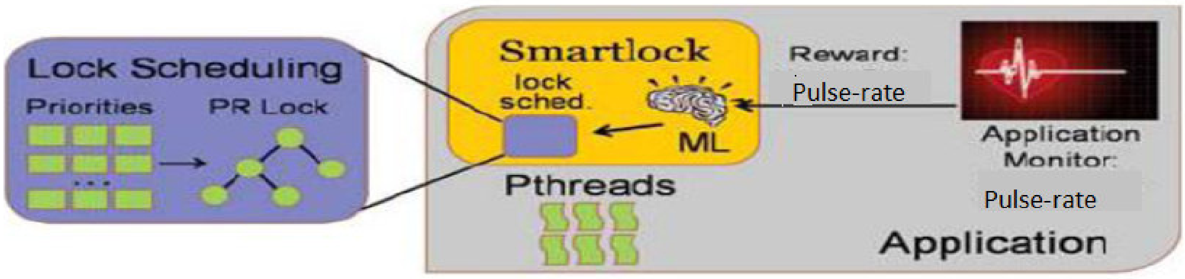

Figure 2. Smartlocks Architecture

As an example C/C++ pthread applications are Smartlock applications that use pthreads for thread producing, Smartlocks for spin-lock synchronization, and the Pulse-rate framework for performance monitoring.

4. Proposed Model

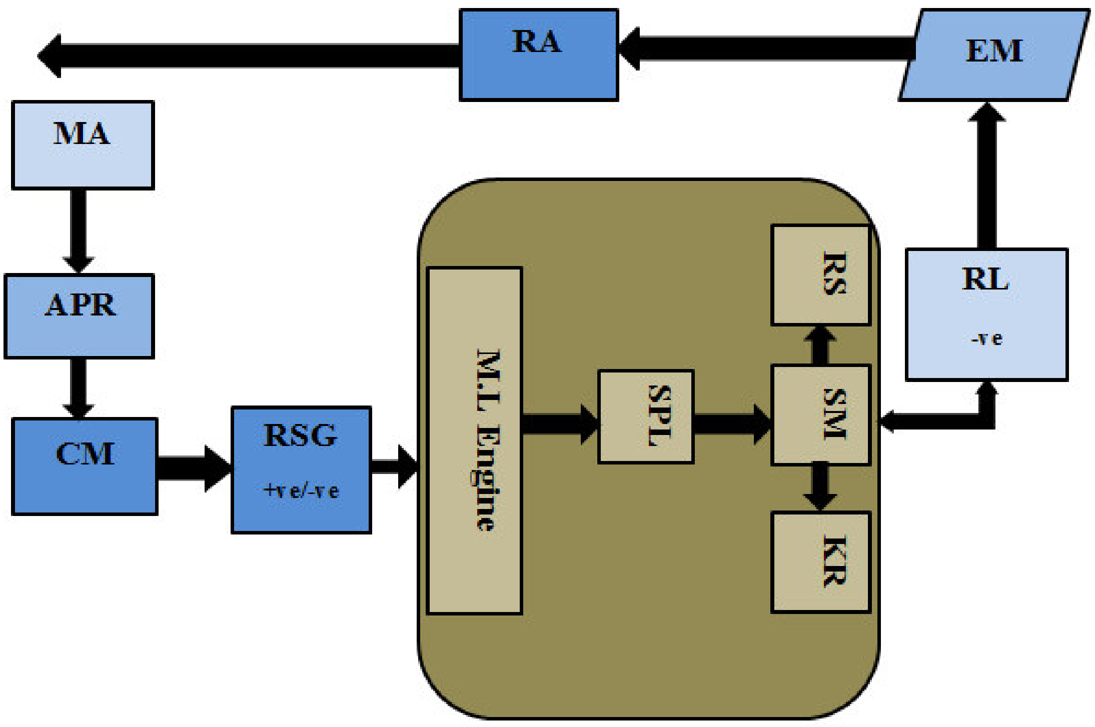

Figure3. Self-Aware Monitoring Based Adaptive Systems Model

5. Conclusion

Adaptive techniques as a whole (Hardware, Operating system and Application) promise to adapt itself according to the environmental changes. Adaptive techniques must be developed which are generally applicable to a wide range of applications and systems.

In this paper Pulse-rate API is discussed which provides a standard interface allowing applications to express their goals and progress to adaptive systems using a standard interface. With the Smartlocks library we have provided an example of using machine learning in combination with application performance measurements to adapt itself in a challenging computing environment. Both Pulse-rate and Smartlocks are built upon techniques which can be generalized to a wide range of adaptive systems. Pulse-rate consists low-overhead and easy to add to a variety of different applications which are used to monitor and detect the changes in the environment.

6. References

[1] Jeffrey O. Kephart and David M. Chess, "The Vision of Autonomic Computing," 2003.

[2] Mazeiar Salehie and Ladan Tahvildari, "Self-Adaptive Software:Landscape and Research Challenges," 2009.

[3] Petre Dini et al., "Internet, GRID, Self-adaptability and Beyond: Are We Ready?," 2004.

[4] Henry Hoffmann , Jonathan Eastep , Marco D. Santambrog, Jason E.Miller , and Anant Agarwal, "Application Heartbeats for Software Performance and Health," 2010.

[5] Jonathan Eastep, David Wingate, and Marco D. Santamb, "Self-Aware, Smartlocks: Lock Acquisition Scheduling for Self-Aware Synchronization," 2010.

[6] Engin Ipek , Onur Mutlu , Jose F. Mart, and ınez Ric, "Self-optimizing memory controllers:A Reinforcement Learning Approach," 2008.

[7] Ledset. [Online]. http://www.ledset.com/camwiz/download/index.htm

[8] Marco D. Santambrogio, Henry Hoffmann, Jonathan Eastep, and Anant Agarwal, "Enabling technologies for self-aware adaptive systems," , 2010.