INTEGRATION, the VLSI journal 47 (2014) 408–416

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Efficient architecture and hardware implementation of hybrid fuzzy-Kalman filter for workload prediction

Shiann-Rong Kuang*, Kun-Yi Wu, Bao-Chen Ke, Jia-Huei Yeh, Hao-Yi Jheng

Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan

ARTICLE INFO

Article history:
Received 13 May 2013
Received in revised form
10 October 2013
Accepted 26 November 2013
Available online 4 December 2013

Keywords:
Workload prediction
Kalman filter
Fuzzy controller
Low area

ABSTRACT

In modern systems, many well-known techniques (e.g., dynamic voltage and frequency scaling, job scheduling etc.) have been developed to achieve low power, high performance, appropriate quality-of-service or other specific purposes. Workload prediction is an extremely critical factor for bringing these techniques into full play. However, it is very difficult to accurately predict the workloads of upcoming tasks if they are varying drastically. In this paper, we propose a new hybrid fuzzy-Kalman filter and the corresponding area-efficient hardware architecture to accurately and quickly predict the workload with large variation. To decrease the hardware complexity while maintaining sufficient accuracy, the computation of Kalman Gain is simplified with a lookup table method. In addition, the workload and covariance values in Kalman filter are properly normalized and truncated to significantly reduce the bit length of hybrid workload predictor. Furthermore, a simplified fuzzy controller is developed to adaptively adjust the measurement noise covariance of Kalman filter so that the prediction error can be further lowered. Experimental results of real applications exhibit that the proposed hybrid fuzzy-Kalman filter can achieve lower prediction error and smaller hardware area when compared to previous workload predictors.

1. Introduction

With the rapid development in the modern computer systems, more and more researches attempt to employ famous techniques (e.g., dynamic voltage and frequency scaling (DVFS) [1–3], job scheduling [4–6], task allocation [7–9], or other management systems) to achieve specific purposes. For instance, power management strategies are usually adopted to lengthen the lifetime of batteries in portable systems. DVFS is a popular power management strategy that assigns a suitable voltage and frequency for each task to save the power consumption under performance constraints. In addition, the grid computing environment, multi-core architecture and real-time system focus on keeping the quality-of-service (QoS), improving performance, or meeting deadline guarantees by resource selection, task allocation, job scheduling, or other management mechanisms.

Most of the above-mentioned techniques confront an important problem: how to exactly and quickly predict the workloads of upcoming tasks for bringing them into full play [3]. In general, there are three categories to estimate the workload of the next upcoming task: workload profiling, workload model, and workload prediction (or called workload forecast). Profiling-based approaches [10,11] use statistical estimation techniques to extract reliable workload statistics, but they may not be very suitable for predicting the workload with large variation. On the other hand, many researches [12,13] build the workload model to compute the predicted workload of the upcoming task by observing the characteristics of specific applications. Workload model probably predicts the workload more accurately, but it can only be utilized in some specific applications. In the case of workload prediction category, it adopts some specific strategies to predict the workload of upcoming task by previous information. Workload prediction is more suitable to be implemented by hardware for real-time systems with violent variation in workload.

Homeostatic workload prediction [7] is one of the most intuitive workload prediction methods. The next workload predicted by this method is likely to increase or decrease a “certain” value to the current workload according to the difference between the current workload and the mean of previous workloads. However, it is difficult to decide the proper quantity of “certain” value for different task in various applications. History-based workload prediction [14,15] is another simple and the popular workload prediction method. The next workload predicted by this method is equal to the sum of weighted workloads of previous tasks. Signature-based workload prediction [12,16,17] constructs the unique workload prediction mechanism based on the characteristics of tasks in the specific application. In addition, recent researches [18–24] have proved that many famous controllers or

* Corresponding author.
E-mail address: srkuang@cse.nsysu.edu.tw (S.-R. Kuang).

0167-9260/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.vlsi.2013.11.006

filters, such as proportional-integral-derivative controller [18,19], fuzzy controller [20,21], or Kalman filter [22–24], could efficiently predict the workload of upcoming tasks exactly and stably. Especially when the workload is varying drastically, these methods can obtain better performance at workload prediction. More details about different workload prediction methods will be explained in Section 2.

Workload prediction can be realized by software or hardware. Software approach predicts the workload by performing the workload prediction algorithm at a processor. Software approach is flexible and can use a sophisticated algorithm to enhance the accuracy of workload prediction. On the contrary, hardware approach uses extra hardware circuit to quickly predict the next workload so that no additional CPU execution time, program memory, and modification to software are required. In general, sophisticated prediction algorithms are unsuitable for hardware implementation. Therefore, few of the above-mentioned workload prediction methods have both advantages of high accuracy and low hardware complexity. To accurately and quickly predict the workload of real-time systems, this paper proposes a new hybrid fuzzy-Kalman filter and its low-area hardware architecture. Firstly, we simplify all multi-dimensional matrixes of Kalman filter into simple scalars to solve workload prediction problem and replace the division operation of Kalman Gain with a lookup table method. Subsequently, the workload and covariance values in Kalman filter are properly normalized and truncated to largely reduce the hardware complexity. Moreover, a simplified fuzzy controller is used to adaptively adjust the measurement noise covariance variable of Kalman filter for further reducing the prediction error. Experimental results of several applications exhibit that the proposed hybrid fuzzy-Kalman filter can achieve smaller prediction error than previous methods. Besides, its hardware area is much smaller than that of other existing methods. The remainder of this paper is organized as follows. Section 2 briefly reviews several famous workload prediction methods. Section 3 illustrates the main ideas of our prediction method and describes the detailed architecture of the proposed hybrid fuzzy-Kalman filter. Section 4 presents the implementation and experimental results. Finally, a conclusion of this paper is drawn in Section 5.

2. Workload prediction methods

In this section, several famous workload prediction methods, including homeostatic prediction, history-based prediction, controller-based prediction, filter-based prediction and hybrid prediction, are reviewed. Before introducing these prediction methods, the notations and basic terms used in the following sections are first described in Table 1.

2.1. Homeostatic prediction

According to the relationship (greater or less) between the current workload W_n and the mean of previous workloads $Mean_{(n, m)}$, homeostatic prediction predicts the workload of next task PW_{n+1} by decreasing or increasing a "certain" value [7]. The decrement or increment value may be independent on or proportional to the current workload W_n. Moreover, the decrement or increment value can be static (it is fixed for all predictions) or dynamic (it is adapted at each prediction). The algorithm of homeostatic prediction is shown in Fig. 1.

2.2. History-based prediction

History-based prediction [14,15,25] is a simple and the popular workload prediction method. Unlike homeostatic prediction that goes back to the mean of previous workloads, history-based prediction predicts the workload of next task through the tendency information of previous workloads. Fig. 2 shows the algorithm of this kind of prediction method which obtains the predicted workload of next task PW_{n+1} by computing the sum of the weighted workload of previous tasks. There are two main factors in history-based prediction: window size and weight of the nth task. Window size decides the amount of the weighted workloads in previous tasks which will be accumulated. In Fig. 2, the variable "k" denotes the window size of history-based prediction. Another important factor, the weight of the nth task (denoted by WR_n), will be used to control the affection from each previous task in the predicted workload of next task. UW_k (uniform window-size k) [14] and polynomial fitting [25] are two well-known methods in history-based prediction. The predicted workload of the next task from UW_k is the average value of the previous k workloads. That is, the window size of UW_k is equal to k and the weight of UW_k is equal to $1/k$. The weight of polynomial fitting can be generated by different methods (e.g., Burg's method [26]).

Table 1
Notations and basic terms used.

Symbol	Description
W_n	The workload of the nth task
$Mean_{(n,\ m)}$	The mean of previous workloads from the nth task to the m-th task
PW_n	The predicted workload of the nth task
WR_n	The weight of the nth task
ε_n	The difference between W_n and PW_n
PC_{n+1}	The output of the proportional controller
IC_{n+1}	The output of the integral controller
DC_{n+1}	The output of the derivative controller
K_P	The weight of the proportional controller
K_I	The weight of the integral controller
K_D	The weight of the derivative controller
x_n	
	The nth posteriori covariance estimation
P_n^-	The nth priori covariance estimation
A	State transition matrix
B	Control input matrix
u_n	The control input in the nth state
Q	Process noise covariance
H	Measurement matrix
R	Measurement noise covariance
KG_n	Kalman Gain
z_n	The nth state measurement
I	The matrix whose elements all are 1

Algorithm *Homeostatic Prediction*

if $(W_n > Mean_{(n,\ m)})$
$$PW_{n+1} = W_n - DecrementValue;$$
else if $(W_n < Mean_{(n,\ m)})$
$$PW_{n+1} = W_n + IncrementValue;$$
else
$$PW_{n+1} = W_n;$$
endif
return PW_{n+1};

Fig. 1. The algorithm of homeostatic prediction.

Algorithm *History-based Prediction*

$$PW_{n+1} = \sum_{i=0}^{k-1} W_{n-i} \times WR_{n-i}\ ;$$
return PW_{n+1};

Fig. 2. The algorithm of history-based prediction.