COMPUTING TECHNIQUES USING HETEROGENOUS MULTICORE ARCHITECTURE

P.Bharath Kumar[1] C.Hari Krishna[2]
1. Assistant Professor, Anantha Lakshmi Institute of Tech & Sciences Anantapur.
1. Assistant Professor, Anantha Lakshmi Institute of Tech & Sciences, Anantapur.

ABSTRACT: The one of the power sufficient computing mechanism is the usage of heterogeneous multi-core processors. It has the ability to meet different resource requirements of various applications in a workload. The challenge of these heterogeneous multi-core processors is the scheduling of programs in a workload. For this purpose, it uses a scheduling mechanism that has a fuzzy logic to calculate suitability between programs and cores. This method achieves 15% average reduction in energy delay product (EDP) when compared to other scheduling mechanisms. Another one we use the Intel's Quick IA heterogeneous prototype platform for studying scheduling.

Keywords*: Heterogeneous, computing mechanism.*

1.1 INTRODUCTION*:*

The heterogeneous multi-core processors provide the architecture capability to accommodate diverse computation requirements of the applications. Then scheduling Techniques controls this architecture for energy efficient computing [1.3]. The program scheduling in these heterogeneous multi-core systems mainly focuses on scheduling of subtasks [2]. Here the program inherent characteristics shapes its hardware resource demands and used to guide the program scheduling.

As more and more core are integrated on chips due to increase in transistor costs [by MOORE's law] hence, heterogeneous multi-core processors are used to provide power [1.2] or performance tradeoffs.

1.2 LITERATIVE REVIEW:

Due to increase in performance and speed, processor power consumption and ejection of heat have become key challenge in the design of high performance systems. For example, P4 processor currently consumes 50W and processors in future are accepted to consume approximately 300W.

To overcome this problem the single-ISA heterogeneous multi-core architecture to reduce processor power dissipation.

For many applications, core diversity is of higher value than uniformity, offering much greater ability to adapt to the demands of the application for different applications have different resource requirements during their execution. Sometimes data have large amount of instruction-level

Parallelism which can be exploited by a core that can issue demand instructions per cycle.

Demands on execution architecture, but also that demand can vary between phases of the same program. We assume the ability to dynamically switch between cores. This allows the architecture to adapt to differences between applications, differences between phases in the same applications, or changing priorities of the processor or workload over time.

To provide an effective platform for a wide variety of application execution characteristics and system priority functions, the cores on the heterogeneous multi core processor should cover both a wide and evenly spaced range of the complexity design space.

1.2.1 WORKS RELATED TO IT:

Siegel presented static and hybrid heuristic to schedule the sub tables in heterogeneous systems.

Kumar et al discussed a dynamic scheduling approach and Chen et al did static application mapping in heterogeneous approach.

1.2.2 Dynamic Scheduling:

Till now, PIE scheduling was evaluated in a static setting, i.e., a workload is scheduled on a given core for its entire execution. There is a chance to improve PIE scheduling by dynamically adapting to workload phase behaviour.

For libquantum, overall performance can be approximately 10% of the instructions. However, the time-scale granularity is relatively fine-grained and much smaller than a typical OS time slice. This suggests that dynamic hardware scheduling might be beneficial provided that rescheduling overhead is low.

1.2.3 Heterogeneous multi-cores:

Heterogeneous multi-cores within a given power budget provides greater performance and reduces energy consumption.

Single-ISA heterogeneous multi-cores are

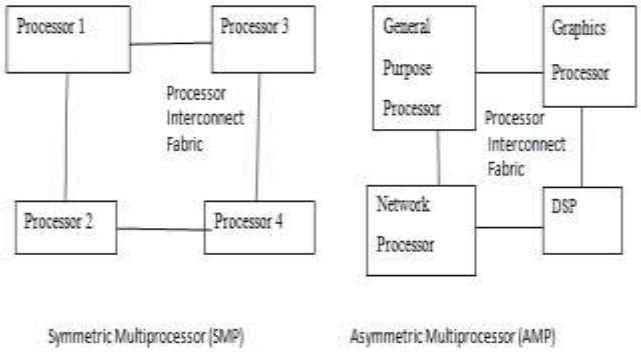


Figure 1: Comparison of SMP and Heterogeneous Multicore Architectures

the different core types implement same instruction-set architecture.

The major problem in design space of single-ISA heterogeneous multi-core processors is how best the workloads to be schedule on most appropriate core type.

Generally, small cores provide good performance for compute-intensive workloads whose subsequent instructions are in the dynamic instruction stream.

1.3 FUZZY INFERENCE SYSTEM (FIS):

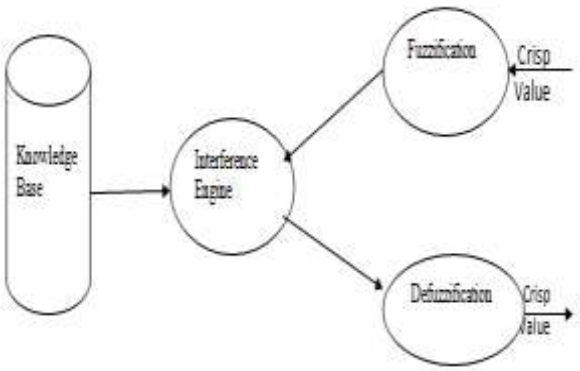
This combines all suitability metrics to produce a single metric. It uses IF-THEN rules. To combine these metrics it uses 4 steps **fuzzification, inference, composition, defuzzification, Fuzzification** transforms crisp input values to fuzzy degrees.

1. **SCHEDULING:**

Each instrument interval has 50 million instructions. Every function is assigned with a unique ID. It consists of four steps:

STEP1:

First one is for identifying boundaries through analysis. The static instrumentation is implemented in the **LLVM** intermediate representation (IR). IR is a static single-assignment based representation. Here we start



with the call and ret instructions.

STEP2:

A call graph is constructed by the program. The call graph is based on Ammos et al context tree.

Each node of call graph is labeled with the name of the function. To differentiate between these functions each are provided with a path ID and node name.

STEP3:

Major program phases can be identified by using the call graph created in the previous step. A function is said to be qualified for major problem phases, if the total number of instructions executed has to be >= th_{ins} and number of invocations should be greater than or equal to thinvoke.

STEP4:

Here it calculates the energy consumption.

The complexity of this scheduling mechanism is O(P*N) Where P-- > number of major program phases detected N-- > number of different types of processors on chip. If the number of cores increases then scalability issue will arise.

2. PIE:

Sampling-based selects the best performing mapping after scheduling dynamically samples different workload-to-core mappings at runtime. While such an approach can perform well, due to periodically migrating workloads between different core types. it introduces performance overhead. To overcome these drawbacks, we propose Performance Impact Estimation (PIE).

To select the appropriate workload-to-core mapping in a heterogeneous multi-core processor, a mechanism Performance Impact Estimation (PIE) can be used.

In PIE performance is estimated if the workload were to run on another core type.

Dynamic PIE scheduling collects profile information on basis of per-interval and adjusts the workload-to-core mapping dynamically, which exploits time-varying execution behaviour.

The major idea behind PIE is to estimate workload performance on a different core type.PIE does this by using CPI stacks. The two major components in the CPI stack are

1. The base component and the memory component.
2. The former lumps together all non-memory related components.

3. Dynamic PIE Scheduling:

Till now we have seen PIE model, let's now see Dynamic PIE scheduling. PIE scheduling

is applicable to any number of cores of any core type. Let assume one core of each type, we assume as many workloads as there are cores, and that workloads are initially randomly scheduled onto each core.

PIE scheduling requires hardware support for collecting CPI stack on each core, the number of misses, the number of dynamically executed instructions, and finally the inter-instruction dependency distance distribution on the big core.

PIE scheduling can be done both in hardware and software. PIE scheduling is applied in software if the time interval of scheduling workloads to cores coincides with a time slice, the hardware would collect the event counts and the software would make scheduling decisions.

PIE scheduling requires hardware support for collecting CPI stack. Collecting CPI stacks on in-order cores is fairly straightforward and is implemented in commercial systems.

PIE scheduling requires some profile information that cannot be collected on existing hardware. For example while we are running on big core, PIE requires the ability to measure the inter-instruction dependency distance distribution for estimating small-core MLP and ILP.

The PIE model requires that the average dependency distance D be computed over the dynamic instruction stream. This can be done by requiring a table with as many rows as there are architectural registers. The table keeps track of which instruction last wrote to an architectural register.

The cost of computing plan can be determined by complexity of scheduling algorithm; however the actual cost of determining the computation plan is not fixed during the simulation.

CONCLUSION:

Hence, here it represents that how a multi fuzzy logic approach is used to schedule programs for every efficient computing using the program characteristics and how scheduling, simulators are used to reduce the power consumption and heat generation results.

The future work includes employing more program characteristics to determine suitability; considering the effects of resource sharing and inter-core communication.

REFERENCES

[1].Multi-core architectures BY Jernej Barbic15-213, Spring 2007 May 3, 2007.

[2]. Noel Eisley,Vassos Soteriou,Li-Shiuan Peh High-Level power Analysis for Multi-core chips NJ08554.

[3]. Jihong Kim Power -Aware Resource Management Techniques for Low-Power Embedded Systems.

[4]. Baniasadi A.,Moshovos A.,"Instruction flow-Based Frontend Throttling for Power-Aware High performance Processors" proceedings of the International symposium on Low Power Electronics and Designs ISLPED01,August 2001.

[5]. Brooks D.Tiwari v.Martonosi M. "Wattch:A Frame work for Architectural -Level Power Analysis and Optimizations " proceedings of the International Symposium on computer Architecture, ISCA,JUNE 2000.

[6]. M.Hamada,y.Ooteguro "Utilizing Surplus Timing for Power Reduction" proceedings of IEEE custom Integrated Circuits Conference 2001,pp.89-92.

[7]. R.Brodersen M.Horowitz, D.Markovic, B.Nikolic and V.Stojanovic "Methods for True power Minimization" proceedings ICCAD,San jose,CA,November 2002.pp 35-42.

[8]. Anand,M.Nightingale and Flinn J.2004.Ghosts in the Machine:Interfaces for better power management. proceedings of the International Conference 23-35.

[9]. Bahar,R.I and Manne 2001.Power and Energy reduction Via pipeline balancing proceedings of

the International Symposium on Hardware/Software Code design.
[10]. ACM Computing Surveys, VOL.37, No.3.September 2005.

P.BHARATH KUMAR **M.Tech [SE], IRPM (HR), MISTE, IAENG, UACEE, CSTA**
As an Assistant Professor in the department of computer science and engineering, **ALTS**, Anantapuramu. He received from **B.Tech** Degree in the Department of Computer Science and Engineering, **JNTU-Anantapuramu** from **2007-2011**. He received from **M.Tech** degree in Software Engineering Specialization from **SVIST**, Madanapalle from **2012-2014**. He received from **IRPM (HR)** Diploma, **SV University** from **2014-2015**.

Technical Memberships

1. Member of Computer Science Teachers Association **(MCSTA)**
2. Member of International Association of Engineers **(MIAENG)**
3. Associate Member of Universal Association of Computer and Electronics Engineers **(UACEE)**

He Received **10** International Journals **1** UGC Approved Journal and **3** International Conferences From various Organizations. His **Research** area is **Cryptography and Network Security, Networking using simulation tool**

C.HARI KRISHNA **M.Tech [AI]**
As an Assistant Professor in the department of computer science and engineering, **ALTS**, Anantapuramu. He received from **B.Tech** Degree in the Department of Computer Science and Engineering, **JNTU-Anantapuramu** from **2007-2011**. He received from **M.Tech** degree in Artificial Intelligence Specialization from **JNTU Anatapuramu** from **2012-2014**.