J. Parallel Distrib. Comput. 122 (2018) 36–50

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

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

Towards high performance data analytic on heterogeneous many-core systems: A study on Bayesian Sequential Partitioning

Bo-Cheng Lai [a,*], Tung-Yu Wu [b], Tsou-Han Chiu [c], Kun-Chun Li [d], Chia-Ying Lee [c], Wei-Chen Chien [e], Wing Hung Wong [b]

[a] Nation Chiao-Tung University, 1001 Da-Hsueh Rd., Hsinchu, 30010, Taiwan
[b] Stanford University, USA
[c] MediaTek Incorporation, No.1, Dusing 1st Rd., Hsinchu Science, 300, Taiwan
[d] HTC Corporation, No. 23, Xinghua Rd., Taoyuan Dist., Taoyuan City, 330, Taiwan
[e] Skymizer Corporation, No.408, Ruiguang Rd., Neihu Dist., Taipei City, 114, Taiwan

HIGHLIGHTS

- Techniques to speedup Bayesian Sequential Partitioning by 48x on a heterogeneous many-core system.
- Proposes a series of techniques, for both data structures and execution management policies.
- Achieve 106x average runtime enhancement while the maximum speedup can reach 197.96x.

ARTICLE INFO

Article history:
Received 24 March 2017
Received in revised form 23 June 2018
Accepted 7 July 2018
Available online 25 July 2018

Keywords:
Data processing
Heterogeneous system
Many-core system
Performance analysis
Design and optimization

ABSTRACT

Bayesian Sequential Partitioning (BSP) is a statistically effective density estimation method to comprehend the characteristics of a high dimensional data space. The intensive computation of the statistical model and the counting of enormous data have caused serious design challenges for BSP to handle the growing volume of the data. This paper proposes a high performance design of BSP by leveraging a heterogeneous CPU/GPGPU system that consists of a host CPU and a K80 GPGPU. A series of techniques, on both data structures and execution management policies, is implemented to extensively exploit the computation capability of the heterogeneous many-core system and alleviate system bottlenecks. When compared with a parallel design on a high-end CPU, the proposed techniques achieve 48x average runtime enhancement while the maximum speedup can reach 78.76x.

1. Introduction

The challenges of information technologies have rapidly shifted from computation bound to memory bound in the past few years [27]. Vast amount of data is being generated and collected due to the drastically growing number of sensors, connected devices, and online users. As the emerging of the massive data [7], the rich information buried in the data samples can be transformed into critical intelligence for individuals [15,26], businesses [2,4], and societies [1,21]. Uncovering this information from the sampled data in a timely manner, therefore, has been an imperative task in the next wave of computing technologies.

The properties of high volume and normally unstructured organization make the sampled data extremely difficult to analyze with human intuition [13]. Machine learning has been demonstrated as an effective method to identify the characteristics of data and automatically extract useful information [5]. Due to the prevalent of feature-rich data content, such as videos [3] or genome libraries [10], high dimensional density estimation has become an important and effective machine learning analytic to comprehend the collected data. By constructing the density function estimated from the data, the abundant and various features of the data can be effectively explored and analyzed [7]. A recent work from Lu et al. [19] demonstrated an effective statistical model, Bayesian Sequential Partitioning (BSP), for density estimation on large volume of high dimensional data. Unlike many of the previous density estimation methods, BSP adopts non-parametric model that assumes no pre-known distribution of the data, and returns the intrinsic density distribution of the observed data. Relying on a presumed distribution when analyzing the target data could easily lead to inappropriate conclusions [19] since the collected data do not necessarily follow a certain statistic distribution. When

* Corresponding author.
E-mail address: bclai@mail.nctu.edu.tw (B.-C. Lai).

https://doi.org/10.1016/j.jpdc.2018.07.011
0743-7315/

compared with previous density estimation methods, BSP has demonstrated better Kullback–Leibler divergence on analyzing high dimensional non-parametric data [16,18,19]. In some examples
discussed in [19], BSP even demonstrates superior classification rates than the powerful and widely used approach of SVM (Support Vector Machine) [6].

Although being statistically effective, there still exist three main challenges to BSP when analyzing the future dataset with large number of samples and growing dimensions. First, in order to accurately estimate the density in a high dimensional space, BSP applies statistic models that require complex computing, such as logarithm and exponentiation. Second, it is data intensive while BSP needs to iteratively count the number of data samples within specific hyper-regions inside a high dimensional space. The above challenges become greater with more features (number of dimensions) and growing volume (number of samples) of future data content [9]. The stringent demand on both computation and data access has caused long execution time. According to our experiment, a reference implementation on a high-end CPU takes several hours to complete the BSP analysis of one million samples with 128 dimensions. The third challenge stems from the changing runtime behavior along the BSP analysis flow. Different stages along the analysis flow would require different design techniques to avoid system bottlenecks and attain superior performance. Together with the first two challenges, BSP poses dynamic characteristics for both execution behavior and data access patterns. These attributes make a heterogeneous many-core system an appropriate platform for performing BSP analysis.

This paper aims to attain a high performance BSP design on a heterogeneous platform [11] that consisting CPU as the host, and GPGPU as the device. In the heterogeneous CPU/GPGPU system, the host CPU takes charge of sophisticated algorithm flows while the device GPGPU performs massively parallel data processing. The heterogeneous system in this paper applies high-end CPUs as the host, and a Kepler K80 GPGPU [24] as the device. Based on the observed characteristics of BSP, the essential design principle is to execute the complex analysis flow on the high-end CPU while performing the highly parallel sample counting on the GPGPU. However, simply dispatching the corresponding tasks to the host and device could easily hit various performance pitfalls while the BSP poses changing execution behavior when analyzing high volume data. Attaining superior performance of the BSP analysis on the heterogeneous many-core system involves thorough understanding of the algorithm behavior as well as efficient management of the large dataset. This paper comprehensively discusses the design phases and corresponding performance bottlenecks when porting a BSP analysis flow from a CPU to a heterogeneous many-core system. Along the design phases, this paper proposes a series of design techniques, on both data structures and execution management policies, to leverage the computation capability and alleviate performance bottlenecks. With the proposed approaches, the overall speedup of the BSP analysis can reach up to 78.76x when compared with the runtime of the reference design on a high-end CPU.

BSP algorithm performs iterative computation that features high dimensional massive data, complex computation, and hybrid and dynamic execution behavior. These properties can be found in many data analytics. This paper not only addresses the design issues in the case of BSP algorithm, the comprehensive analysis and corresponding techniques also provide effective guidelines when designing other data processing schemes with similar attributes. This paper first demonstrates significant performance enhancement by performing the massively parallel sample counting on a GPGPU. A two-layer indexing (TLI) scheme is proposed to enable efficient management of the large volume of data samples during the BSP analysis. This indexing technique greatly reduces the overhead of the data operations. Then, this paper proposes two techniques, TLI extension and TLI defragmentation, to integrate the TLI data structure with a tree-like structure that are adopted in the reference BSP implementation to manage the information of regions generated along the BSP analysis. By complying with the tree-like structure, the information in the data structure can be efficiently shared among BSP partitioning processes and reduce the redundant counting of samples. According to empirical observations, the most time consuming part in the later iterations of BSP analysis is shifted from counting data samples to the complex calculation of statistic parameters. This paper further implements analysis status boards to track the dynamic analysis behavior of BSP and remove the unnecessary calculations of the statistic model. By sharing these parameters between BSP partitioning processes, the newly generated regions in a process can reuse these parameters without recalculation.

The rest of this paper is organized as follows. Section 2 introduces the Bayesian Sequential Partitioning algorithm, including fundamental theories and a reference analysis flow. Section 3 shows the heterogeneous many-core computing system, and details the implementation and design techniques of BSP algorithm. The performance analysis will be discussed in Section 4. Section 5 concludes the design principles learned from previous sections. Section 6 discusses related works and Section 7 draws the conclusions and future work.

2. Background

This section introduces the Bayesian Sequential Partitioning (BSP) analysis scheme. In general, BSP is an efficient way of analyzing high dimensional non-parametric data [16]. It has demonstrated superior Kullback–Leibler divergence and has been demonstrated to achieve better classification rates than SVM [19]. BSP also serves as a general data exploration tool and is readily applicable to many important learning tasks [18], such as finding good initializations for k-means. BSP is also applicable to facilitate other applications including mode seeking, data visualization via level set tree and data compression [18].

In general, a BSP analysis can be decomposed into three execution layers, including BSP partitioning process, BSP algorithm, and BSP-based analysis flow. BSP partitioning process is the core part that applies the effective statistical model proved by rigorous BSP theories [18,19]. BSP algorithm initiates multiple BSP partitioning processes to increase the chance of finding a partitioning solution that can well approximate the distribution of data in the sample space. BSP-based analysis flow preprocesses the sample space and integrates the BSP algorithm to efficiently and effectively analyze a high dimensional sample space. While this paper focuses on a high performance design of BSP analysis, this section will introduce the essential background behind the BSP models, algorithms, and an analysis flow. The complete statistical model and theoretic derivations will not be elaborated in this paper. Readers can refer the work of Lu and et al. [19] for more detailed discussions.

2.1. Algorithm of Bayesian Sequential Partitioning

Bayesian Sequential Partitioning (BSP) is a statistically effective method to estimate the density distribution of a high dimensional dataset. Unlike parametric methods, BSP assumes no pre-known distribution of the analyzed data. Instead, BSP aims to build a data-driven density distribution by sequentially partitioning the sample space into sub-divisions that can be applied to create an adaptive histogram.

BSP estimates the distribution from the samples in the sample space (Ω) of R^d, where d represents the number of dimensions. The sample space Ω is then iteratively partitioned into sub-divisions.