Towards hardware-driven design of low-energy algorithms for data analysis

Indre Zliobaite, Jaakko Hollmen,
Aalto university and HIIT
Espoo, Finland
firstname.lastname@aalto.fi

Lauri Koskinen, and Jukka Teittinen
Technology Reserch Center, University of Turku
Turku, Finland
firstname.lastname@utu.fi

ABSTRACT

In the era of "big" data, data analysis algorithms need to be efficient. Traditionally researchers would tackle this problem by considering "small" data algorithms, and investigating how to make them computationally more efficient for big data applications. The main means to achieve computational efficiency would be to revise the necessity and order of subroutines, or to approximate calculations. This paper presents a viewpoint that in order to be able to cope with the new challenges of the growing digital universe, research needs to take a combined view towards data analysis algorithm design and hardware design, and discusses a potential research direction in taking an intreated approach in terms of algorithm design and hardware design. Analyzing how data mining algorithms operate at the elementary operations level can help do design more specialized and dedicated hardware, that, for instance, would be more energy efficient. In turn, understanding hardware design can help to develop more effective algorithms.

1. INTRODUCTION

More and more data is being generated every day by people, enterprises and smart devices. It is estimated [5] that the digital universe is growing by 40% a year, and by 2020 the data we create and copy annually will reach 44 zettabytes, or 44 trillion gigabytes. While currently less than 5% of the information in the digital universe is actually analyzed, it is estimated that at least 20% would be a candidate for analysis. Hence, there are more opportunities, but at the same time, more challenges for data analysis, and not only human resources, but also new technologies are needed to make the best use of these opportunities.

Data generated by smart devices is particularly interesting, since it presents opportunities for analysis that did not exist a few years ago. Currently data from embedded systems (Internet of Things) accounts for only about 10% of all the analyzable data, it is estimated [5] that by 2020 data from embedded systems will account for half of all the analyzable data. Such devices typically operate using autonomous power sources, therefore, research needs to anticipate how to process such data in the most energy efficient way.

This presents an interesting dilemma from the data mining perspective. Ideally, all of the data gathered in the wireless sensors would be tranferred wirelessly to be centrally processed. And while the energy consumption of data processing continually falls with Moore's Law [7, 6], the energy consumption of transmitting a bit across a given distance does not follow the Law as advantageously as the digital processing. Therefore, the energy cost of wireless transmission will proportionally grow when compared to digital processing. This means we will have to do more and more data analysis on smart devices, instead of sending it around.

Due to power constraints, it appears that the current general processor multi-core model is unlikely to scale beyond a limited number of cores; it will no longer be possible to use all cores simultaneously (a notion called Dark Silicon by ARM). This will result in heterogeneous multi-cores where only a few accelerators are used at any given time. An example can be seen in current mobile phone application processors which consist of many specialized units, for instance graphics processing accelerators, radio baseband accelerators etc. For example, Qualcomm's new Snapdragon S4, marketed as a dual-core processor, is actually a 10-core unit with a GPU, several DSPs, modem, etc. Processors will further develop into consisting of more subsystems and will therefore be heterogeneous units specialized for particular purposes.

In this article we present a viewpoint that in order to be able to cope with the new challenges of the growing digital universe, research needs to take a combined view towards data analysis algorithm design and hardware design. Analyzing how data mining algorithms operate at the elementary op-

erations level can help do design more specialized and dedicated hardware, that, for instance, would be more energy efficient. In turn, understanding hardware design can help to develop more effective algorithms.

The remainder of the paper is organized as follows. Section 2 overviews the challenges and desired properties from the hardware side. Section 3 overviews the challenges and desired properties from the algorithmic side. Section 4 analyses the existing most popular data mining/machine learning algorithms in terms of elementary operations, and relates algorithmic and hardware side. Section 5 discusses open research directions in energy efficient data mining.

2. OPPORTUNITIES AND CHALLENGES FROM THE HARDWARE PERSPECTIVE

Moore's Law has promised doubling of processing power every 18 months due to the scaling of CMOS[1] transistors, which are at the heart of all processors. While the scaling of transistors still continues, extraneous circumstances have made achieving the higher processing power increasingly difficult. About ten years ago, UC Berkeley's David Patterson defined the "Three Walls". While these design complexities (aka. "Walls") were known previously, Patterson combined them in a forward-looking equation:

Power Wall+Memory Wall+ILP Wall = Brick Wall.

These Three Walls could be shortly defined as follows:

Power Wall
If processor speed (clock frequency) would follow Moore's Law, the ensuing heat generated by the power consumption would destroy the processor. As the processor clock frequency is linearly (and superlinearly in extreme cases) related to power consumption, this can be seen in the plateauing of processor clock frequencies to c. 3-5 GHz.

Memory Wall
Ideally, all of the data to be processed would be stored adjacent to the processing element. However, with current memory technology, only small amounts can be stored beside the processor (e.g. L1 cache) and the ensuing data transfer delay stalls the processing. This can be seen in hiding memory latency with increasing on-chip cache sizes and levels and also the increasing amount of concurrently processed threads.

Instruction-Level Parallelism Wall (ILP)
The single-thread performance has been traditionally increased by parallelizing the sequential part of computer programs. This was achieved with better compilers or more complex architectures (out-of-order instruction processing, branch speculation, VLIW[2]).

To compound the problem, tackling one wall will make the other two worse.

The ultimate solution to break the Brick Wall is moving away CMOS technology to a novel technology. As such a mature technology does not yet exist, other tricks have been devised to mitigate the problem and lower the Wall. In the early 2000s processors moved from single core to multi core, but this made the Memory Wall worse. Cache coherence reduces the Memory Wall, but aggravates the Power Wall. The Power Wall currently also manifests itself in cloud computing via the electricity bill.

As smaller transistors allow cramming more cores into a single chip, Dark silicon, described in Section 1 is truly the main way to lower the Power Wall. However, two other problems ensue: the Programmer Wall and the Communication Wall.

The Programmer Wall basically relates to abstracting the heterogeneous nature of Dark Silicon as algorithms and software are not ready for computing on thousands of processors.

The Communication Wall Communication here means either moving data between levels of a memory hierarchy (sequential case), over a network connecting processors (parallel case), or wirelessly between processing nodes (sensor network case).

While creating an efficient programming model and associated compilers for Dark Silicon is out of scope for this discussion, the Communication Wall can already be tackled at the algorithm level (see e.g. [2]). Previously, the cost of communicating was divided into bandwidth and latency. However, the energy cost of communication is proportionally growing when compared to computation. As mentioned previously, due to the physics of electromagnetic signalling, wireless communication is the extreme

[1]Complementary metal oxide semiconductor (CMOS) is a technology for constructing integrated circuits.

[2]Very long instruction word (VLIW) is a processor architecture designed to take advantage of instruction level parallelism.

case. But even on-chip, the delay of wiring grows quadratically with scaling and as a result increasing amount of energy will have to be used to achieve sufficient bandwidth and delay.

To summarize, the algorithm designer should think of specialized functions that can be turned into Dark Silicon processing elements and minimize communication (memory accesses and data transfer between cores). When designing algorithms, or choosing existing algorithms for implementation, data analysis should be aware of energy costs of elementary operations at different hardware designs, and much energy which operation consumes. That would allow to prioritize. Algorithm and elementary function complexity and algorithm execution time are rudimentary estimates on energy consumption, which, in some cases, can be mitigated with Dark Silicon. Amount of memory required by the algorithm should also be minimized. Communication can be minimized by, for example, using small data sets that fit in a processor cache (and thereby avoid extra communication with the main memory) or using hierarchical algorithms (where some levels of the hierarchy can be performed in wireless nodes).

3. LOW-ENERGY ALGORITHMS

All data analysis algorithms can be broadly categorized as descriptive or predictive modeling.

Descriptive modelling aims to extract information from large amount of data and summarize it into a model. The main goal is to describe a dataset at hand, and the main result of an algorithm is data summary (e.g. clustering traces of rental bicycles into groups in order to develop new bicycle roads).

Predictive modelling aims to extract summarizing information from large amount data into a model as well, however, the main goal is not to describe the data at hand, but rather to be able to generalize to new unseen data (e.g. recognizing the model of transportation based on accelerometer data captured by a mobile device).

While in descriptive modeling the modeling algorithms need to be as efficient as possible, in predictive modeling there needs to be a tradeoff between efficient modeling and efficient generalization. For example, producing a regression model requires much more computation resources than later applying the model to new data. On the other hand, for instance, k-nearest neighbour algorithm would require much more computing resources at the application stage.

There are many strands of research related to energy-efficiency, rather scattered across different research areas. The authors in [15] insist that a paradigm shift towards energy-efficiency is needed, if we want to use thousands of battery-powered low-cost sensors for long-term surveillance, for instance. The shift means that we need to focus on maximizing resource efficiency rather than just maximizing performance. They emphasize that in order to achieve efficient use of energy and communications, systems should be distributed, co-operative, and opportunistic (e.g. power-down mechanisms, or systems with multiple states [1]).

Sensor networks are inherently distributed systems. They have many areas of application, including tracking by multiple sensors [4], and structural health monitoring [9], to mention a few. In these applications, there is a concern whether the long-term operation of such systems is feasible at all, considering that there is a large cost for maintenance, to change the batteries, for instance. In the area of structural health monitoring, there are solutions to seek for for parsimonious models in order to minimize the amount of communication [10] between the sensor nodes.

Mobile sensing has become commonplace in everyday smart phone applications, however, the sensing consumes a fair amount of energy [16]. Since energy consumption may become an issue, researchers [3] created a simulation library to simulate a real-world deployment scenario in order to predict resource use in smart phones and wireless sensor networks.

A more theoretical perspective on computation and the energy needed to to compute (if at all) is discussed in [13], where the trade-off between time to perform a computation and energy needed to compute is discussed.

One line of work towards more resource-aware computation is to use lower precision of floating point numbers than the standard [8] used in digital computers. One such work makes use of limited-precision floating point numbers in the context of naive Bayes classification [11]. The research questions are then to quantify the overall compromise in the accuracy of the algorithm in the classification setting.

Overall, most of the research on energy efficient algorithms consider optimizations and savings that are algorithm specific. It means that optimizing every next algorithm starts from the beginning.

4. ANALYSIS OF ALGORITHMS AND ELEMENTARY OPERATIONS

From the data analysis algorithm design perspective energy efficiency may be improved in three main ways:

1. approximate computation,
2. revising the necessity of operations within an algorithm,
3. more efficient implementation of operations.

The first two approaches would need to be tailored for concrete algorithms. For example, if we want to optimize k-means clustering algorithm, we would need to decide, which computations within this algorithm can be done approximately, and which operations may not be necessary. Moreover, optimizing single algorithms is tedious, and the gains in energy efficiency may vary from algorithm to algorithm. For some algorithms it may be difficult or impossible to come up with further computational optimization. Hence, we find the third approach to optimization to be promising.

Instead of optimizing individual algorithms, we can try to optimize elementary operations that are commonly used by many algorithms. These operations could be optimized at the algorithmic level and the corresponding processing elements at the hardware level. For example, k-means requires computing distances between two vectors many times. Optimizing the distance computation operation would contribute to a wide range of data analysis algorithms, not only k-means.

The first step towards such an approach would be to identify the most common elementary operations used in data analysis. For a rough approximation of such ranking we conduct the following survey.

We analyze the top 10 data mining algorithms [14], recently identified by a panel of research leaders: C4.5, k-Means, SVM, Apriori, EM, PageRank, AdaBoost, kNN, Naive Bayes, and CART. In addition, we consider three more algorithms, commonly used in practical data mining tasks: the logistic regression (classification), the linear regression, and the Fast Fourier Transform (FFT) commonly used in signal processing.

Table 1 lists elementary operations on the left hand side, and each column corresponds to one of the 13 data mining algorithms. A bullet sign indicates the presence of an elementary operation in that algorithm. The bottom line indicates to which data mining task an algorithm belongs: CLA - classification, REG - regression, C&R - classification and regression, CLU - clustering, ASS - association rule discovery, DSP - digital signal processing. The right column gives a weighted score, that we will describe in more detail.

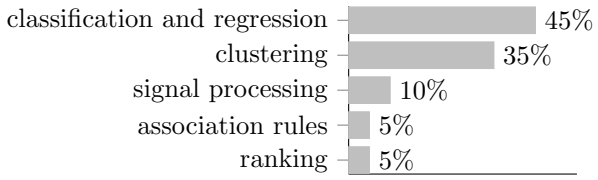

Figure 1: Estimated shares of data mining/machine learning tasks on autonomous devices.

We could simply sum the appearances of each elementary operation and prioritize the operations, that get the largest number of points, assuming that these are the most frequent operations. However, this does not take into account that each of the 13 algorithms is not equally to be used, some algorithms may be used more often than the others. To take the frequency of usage into account we first consider what is the share of each data mining task in practical sensor data analysis using autonomous devices. This allocation is completely subjective, and is based on our personal experiences; however, it provides a baseline for a start. An alternative way to estimate the relative prevalences of different algorithms would be to use an experiment database, see [12].

If we think of a data mining/machine learning processor, optimized for energy efficiency when analyzing data, we allocate shares of usage to different activities as given in Figure 1. Each of the 13 algorithms belongs to one of the categories. We assume that within each category any of the considered algorithms is equally likely to be used. For example, all clustering is assumed to take 35% of the processor time, we have two clustering algorithms (k-means and EM), hence, each of them is allocated 17.5%, which is half of the total allocation. The weighted scores are obtained by the following equation: $S_{weighted} = N \sum_{i=1}^{N} w_i r_i$, where $N = 13$ is the number of algorithms, w_i is the weight of each algorithm, $r_i = 1$ indicates the presence or $r_i = 0$ indicates the absence of a given elementary operation in algorithm i. The weighted scores in Table 1 indicate the frequency of each operation in the presumed general purpose autonomous data mining device, the higher the score - the more important it is to optimize this operation for making it energy efficient. This is a simplified approach to analysis, since we assume that each algorithm requires the same number of operations.

Table 1: Analysis of elementary computations.

Operation	C4.5	SVM	Adaboost	k-nn	Naive Bayes	CART	logistic regression	Lin.reg.	PageRank	k-means	EM	Apriori	FFT	Weighted score
Sum of the vector elements		•	•	•	•		•	•	•	•	•	•	•	11.5
Average of the vector elements			•	•	•		•	•	•	•	•		•	10.2
Sub-setting of values, windowing, find	•			•	•	•			•	•	•	•	•	10.1
Normalization to unity		•		•	•				•	•	•	•	•	9.3
Euclidean distance			•	•			•	•		•				5.2
Logarithm	•		•		•	•					•			5.2
Histogram					•				•		•	•		4.3
Inner product		•	•				•	•						2.9
Square root	•		•		•	•								2.9
Discretization	•				•	•						•		2.8
Entropy calculation, conditional entropy	•		•			•								2.2
Gini index	•		•			•								2.2
Cosine, Sine (due to exp())		•											•	2.0
Sigmoid function (non-linearity due to exp())		•						•						1.5
SVD, Matrix inversion		•	•											1.5
Absolute value														0
Weight (w)	5.6%	5.6%	5.6%	5.6%	5.6%	5.6%	5.6%	5.6%	5.0%	17.5%	17.5%	5.0%	10.0%	
Task	CLA	C&R	C&R	C&R	REG	C&R	CLA	REG	RAN	CLU	CLU	ASS	DSP	

We can see from the table that vector operations take the priority. The first four operations receive by a large margin higher weighted score than the rest. These should have priority for optimization on the hardware level with application specific components and memory addressing.

5. CONCLUSION

Our position is that in order to be able to cope with the new challenges of the growing digital universe, research needs to take a combined view towards data analysis algorithm design and hardware design. A way to do it is to iteratively analyze the algorithms at an elementary operation level, optimize the performance of the relevant elementary operations at the hardware level. As the first step in this direction, we presented a preliminary survey of such operations.

When designing algorithms, or choosing existing algorithms for implementation, data analysis should be aware of energy costs of elementary operations at different hardware designs, and much energy which operation consumes. That would allow to prioritize elementary operations in the hardware design.

This integrated approach to algorithm design opens an interesting and important research directions. The focus should be on understanding the big picture of the available data analysis algorithms, which will allow to develop low-energy solutions for the big data problems in a systematic way.

6. ACKNOWLEDGEMENTS

This research is supported by Academy of Finland grant 118653 (ALGODAN), and ARTEMIS joint undertaking under grant agreement no 641439 (ALMARVI).

7. REFERENCES

[1] S. Albers. Energy-efficient algorithms. *Commun. ACM*, 53(5):86–96, May 2010.

[2] G. Ballard, J. Demmel, O. Holtz, and O. Schwartz. Minimizing communication in numerical linear algebra. *SIAM J. Matrix Analysis Applications*, 32(3):866–901, 2011.

[3] M. Buschhoff, J. Streicher, B. Dusza, C. Wietfeld, and O. Spinczyk. MobiSIM: A simulation library for resource prediction of smartphones and wireless sensor networks. In *Proceedings of the 46th Annual Simulation Symposium*, ANSS, pages 8:1–8:8, 2013.

[4] V. S. Cheng and E. H.-C. Lu. Energy-efficient real-time object tracking in multi-level sensor networks by mining and predicting movement patterns. *The Journal of Systems and Software*, 82:697–706, 2009.

[5] IDC. Emc digital universe, 2014. http://www.emc.com/leadership/digital-universe/2014iview/index.htm.

[6] G. Moore. Progress in digital electronics. *Technical Digest of the Int'l Electron Devices Meeting*, page 13, 1975.

[7] G. E. Moore. Cramming more components onto integrated circuits. *Electronics Magazine*, pages 114–117, 1965.

[8] M. L. Overton. *Numerical Computing with IEEE Floating Point Arithmetic*. SIAM Publishing, 2001.

[9] J. Toivola and J. Hollmén. Feature extraction and selection from vibration measurements for structural health monitoring. In *Proceedings of the 8th International Symposium on Intelligent Data Analysis*, IDA, pages 213–224, 2009.

[10] J. Toivola and J. Hollmén. Collaborative filtering for coordinated monitoring in sensor networks. In *Proceedings of the 2011 11th IEEE International Conference on Data Mining Workshops*, ICDMW, pages 987–994, 2011.

[11] S. Tschiatschek, P. Reinprecht, M. Mücke, and F. Pernkopf. Bayesian network classifiers with reduced precision parameters. In *Proceedings of the 2012 European Conference on Machine Learning and Knowledge Discovery in Databases*, ECMLPKDD, pages 74–89, 2012.

[12] J. Vanschoren, H. Blockeel, B. Pfahringer, and G. Holmes. Experiment databases. *Machine Learning*, 87(2):127–158, May 2012.

[13] P. Vitanyi. Time, space and energy in reversible computing. In *2005 ACM International Conference on Computing Frontiers*, pages 435–444, May 2005.

[14] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg. Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37, Dec. 2007.

[15] Q. Zhao, R. Manohar, R. Ulman, and V. V. Veeravalli. Resource-constrained signal processing, communications, and networking. *IEEE Signal Processing Magazine*, 24(3):12–14, May 2007.

[16] I. Zliobaite and J. Hollmén. Mobile sensing data for urban mobility analysis: A case study in preprocessing. In *Proceedings of the Workshops of the EDBT/ICDT 2014 Joint Conference*, pages 309–314, 2014.