MAGICAL - Genetic Algorithms for More Efficient In-memory Computation

Andey Robins
Dept. of Electrical and Computer Engineering
University of Central Florida
Orlando, USA
andey.robins@ucf.edu

Abstract— Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus animo, cum corpore dolemus, fieri tamen permagna accessio potest, si aliquod aeternum et infinitum impendere malum nobis opinemur. Quod idem licet transferre in voluptatem, ut postea variari voluptas distinguique possit, augeri amplificarique non possit. At etiam Athenis, ut e patre audiebam facete et urbane Stoicos irridente, statua est in quo a nobis philosophia defensa et collaudata est, cum id, quod maxime placeat, facere possimus, omnis voluptas assumenda est, omnis dolor repellendus. Temporibus autem quibusdam et aut officiis debitis aut rerum necessitatibus saepe eveniet, ut et voluptates repudiandae sint et molestiae non recusandae. Itaque earum rerum defuturum, quas natura non depravata desiderat. Et quem ad me accedis, saluto: 'chaere,' inquam, 'Tite!' lictores, turma omnis chorusque: 'chaere, Tite!' hinc hostis mi Albucius, hinc inimicus. Sed iure Mucius.

Index terms—in-memory computation, artificial intelligence, genetic algorithms, computer aided design

I. Introduction

Data and the calculations performed on data are physically separated in modern computing devices. Data is moved from a storage media to closer and closer locations before it is finally used for computation before being places back into memory. This paradigm of computing, the Von-Neumann paradigm, has been instrumental in the development of modern computing solutions. As processing has sped up though, this transfer of information from storage to processing has begun to form a bottleneck which limits the computational speeds which can be achieved by state-of-the-art devices. Specialized processing units, such as Graphics Processing Units and other purpose-built hardware, often attempts to side-step the problem by increasing the potential throughput of information; however, the theoretical problem of moving data around is not solved, only mitigated, by these solutions. An alternative paradigm to the Von-Neumann architecture could be performing the computation at the same place the data is stored. This computing paradigm is aptly referred to as "in-memory computation."

Memristor Aided Logic (MAGIC) is an emerging computing paradigm making use of parallel, write-based systems to perform calculation in-memory [1]. This requires scheduling operations for the computation; however, the scheduling order, upon execution, may have substantially differing memory footprint requirements. State-of-the-art solutions model this dependence as a graph problem and perform scheduling as a graph covering problem. In this work, we characterize a number of properties of these evaluations graphs and apply those observations to the development of a genetic algorithm which produces reductions in the memory footprint of execution between 14% in the worst case and 26% in the best case when compared to standard algorithmic approaches

II. Prior Works

III. Problem Specification

In-memory computation can be modeled as an execution graph. Translated from traditional combinational logic, each vertex in the graph corresponds with a boolean logic gate. Within the memristor array, the input values to the logic gate can be selected by the write signal before being written to an empty location in the memristor array. An edge exists in the graph from a vertex to another if they have this depednece relation. As an example, for evaluating the boolean function $f = ab' + c$, Figure 1 details the transformation from function to circuit to graph to in-memory computation. Beginning with a boolean function f, existing processes are highly capable of mapping this to a minimal circuit. This circuit can then be transformed into the graphs under discussion in this work by assigning each gate a vertex and making each wire an edge in the graph. The final step illustrated in Figure 1 is the evaluation using in-memory computation and illustrates

the requirement of more memory cells than inputs to the function. The objective of this work is to minimize the extra memory needed.

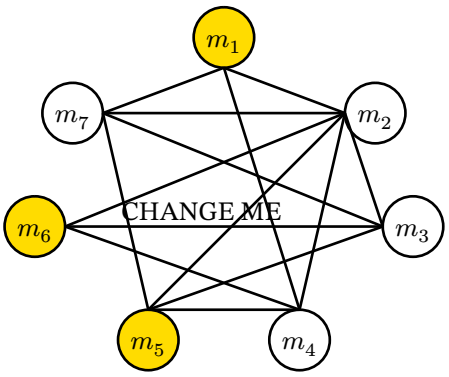

Figure 1: The MIS graph constructed from the covering table presented in the assignment.

Formally, the interdependence between computational nodes in the execution graph is a directed acyclic graph (DAG) in which the children of a vertex must not be executed until that vertex is executed. Thus, for any vertex, it can be viewed as both the root of

A. Data Set

IV. Genetic Algorithm

V. Evaluation

VI. Results

VII. Discussion

VIII. Conclusion

References

[1] S. Kvatinsky *et al.*, "MAGIC—Memristor-aided logic", *IEEE Transactions on Circuits and Systems II: Express Briefs*, no. 11, pp. 895–899, 2014.