STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LIV**, Number 1, 2009

UNCONVENTIONAL COMPUTING: A SHORT INTRODUCTION

MIHAI OLTEAN

Abstract. Physics imposes some limits to the computations that we can perform. Because of these limits we cannot solve problems in almost no time and with almost zero energy consumption. The main problem is that we are asking too much from a single device: we want computers that play chess, solve equations, navigate on the internet etc. Because of this great generality we are paying some big prices (in time and energy consumption). Some problems can be efficiently solved by using other concepts. Some materials have intrinsic properties which makes them very suitable for solving some kind of problems. In this paper we provide a short survey to the field of Unconventional Computing. We review and compare the most popular problems and methods belonging to this field.

1. Introduction

The quest for problem solvers has started since the humans have appeared on Earth. It seems that there is something in the human nature which push us to create machines for replacing our work.

Maybe the most prominent effort in this direction are the computers - which try to replace to human intellectual effort. Standard computers are able to perform faster computations than the human brain (for some problems). Take for instance the multiplication of 2 numbers. A computer can do this operation with several orders of magnitude faster and better than humans.

However, standard computer have a lot of limitations too. If you show them a picture they are not able to tell you which person is an old-friend of yours. The also cannot speak very well. They have difficulties in recognizing spoken words. These are just some of the limitations that a computer has.

Fortunately standard (digital) computers are just a part of the set of the possible computation devices. Recent years have shown a growing interest in

Received by the editors: April 2, 2009.

2000 *Mathematics Subject Classification.* 68Q05, 68Q10, 68Q17, 68Q99, 68W10, 81-xx, 78Axx.

1998 *CR Categories and Descriptors.* F [**Theory of Computation**]: General; F.1.2 [**Modes of Computation**]: Parallelism and concurrency.

Key words and phrases. unconventional computing, natural computing, survey.

Unconventional Computing (UC) devices. These types of computers are called unconventional mainly because they have been only recently invented, operate with some exotic principles, and because they have not been yet introduced on the market.

In this paper we make a short survey of the unconventional computation field. We have mainly focused our attention on NP-complete [13] problems only because these are the most difficult problems for standard computer. There is no digital computer which can solve NP-complete problems in polynomial time. Some researchers hope that unconventional computers are capable of solving NP-complete problems in polynomial time. There are already some examples which support this claim.

Another direction is given by some problems for which unconventional devices can perform much better (in both running time and energy consumption) than digital computers. Recently [29] some basic problems (such as sorting) have been shown to be solved in $O(1)$ time on an unconventional device. With such improvements over more basic problems we can get important benefits (especially in energy consumption).

The paper is structured as follows: We start in section 2 by defining what computing (both conventional and unconventional) is. The relationship between Natural Computing, Artificial Intelligence and Unconventional Computing is described in sections 2.1 and 2.2. The motivation for studying and researching unconventional computing is given in section 3. Problems that we want to solve are briefly described in section 4 and 5. Section 6 investigates if UC can solve NP-complete problems in polynomial time. The question if analog devices are better in all cases than their digital counterparts is negatively answered in section 7. Whether physical implementation is possible is discussed in 8. A table showing a comparison between UC devices is displayed in section 9. Advantages and weaknesses of the unconventional methods are discussed in 10. The main journals and conferences in the UC community are given in section 11. Finally, section 12 concludes our paper.

2. Conventional and unconventional computing

Before starting our talk about unconventional computing we have to see what computing (either conventional or unconventional) means. Most people believe that computing is performed only by standard computers such as desktops or laptops. This is completely incorrect.

All objects around us perform some kind of computing. Imagine a box full with sand. One might say that it is not a computing device because it does not have a standard configuration like we know it: it does not have processor, memory, screen, keyboard etc. But, this is wrong, because even without all these components one might still perform some great computations. Another one might say that a glass of water is unable to perform computations. This is again completely wrong.

Definition. *A computer is any physical object that can be reconfigured to solve multiple problems, that is, to answer many different questions.* [21]

If the device cannot be reconfigured it is usually called experiment [21].

Having this said we can define unconventional computing as being computing without standard digital computers. We use DNA, quantum properties, light, water, chemical substances, mechanical devices, nano-technologies, etc for performing computations, but not standard computers.

2.1. **Natural computing vs. unconventional computing.** There is a strong relationship between Natural Computing (NC) and Unconventional Computing. Natural computing is a large field contain all possible techniques and devices using principles inspired from nature or using nature's materials (exception being made for standard computers which - even if are made from nature's materials - are not considered unconventional).

The following facts differentiate UC from NC:

- Unconventional computing is a subset of natural computing.
- Unconventional computing is mainly focused on physical devices whereas NC includes algorithms too (such as Genetic Algorithms [17], Particle Swarm Optimization, etc),
- Unconventional computing also includes algorithms, but these algorithms are specially designed for unconventional machines (for instance algorithms for solving Hamiltonian Path Problem on a DNA computer [2], Integer factorization on a Quantum computer [31], Steiner tree on a machine with soap bubbles [6] etc) .

2.2. **Artificial Intelligence and Unconventional Computing.** There is also a strong relationship between Unconventional Computing and Artificial Intelligence (AI). This connection is given by one of the research direction in AI. Actually, the research in the field of AI is divided into 2 main directions:

- Weak AI - whose purpose is to develop some intelligent algorithms for solving some particular problems. Weak AI can run on standard computers. Weak AI is everywhere. Neural networks [36], Evolutionary Algorithms [17], Fuzzy Systems [35] are all techniques belonging to weak AI field. These methods help us to solve some problems, but they cannot operate without human control.
- Strong AI - whose purpose is to develop universal intelligence capable to match human performances. Currently there is no computer capable of supporting strong AI. The computer's architecture is too different from the requirements of strong AI paradigm. This is why is widely accepted that a new, unconventional architecture is required

for achieving this level of intelligence. (For instance human brains are unconventional devices running strong AI algorithms.)

3. Motivation

Currently no unconventional device can replace standard computers because the technology is at beginning. More work is needed on all directions. However, in the near future we could expect to see special devices operated by principles totally different from those of standard computers.

There are a large number of reasons for which UC field requires a special attention. Here are some of them:

- Standard computers have a limit. It will not be long until we cannot fit more transistors on a square unit because we cannot decrease the size of the components forever. At that moment of time we have to search for alternate modes for performing computations. Some of these methods could be the unconventional methods of today.
- Standard computers are too slow for some problems. For instance sorting is a critical operation inside most of the programs, but no general algorithms operating in less than $O(nlog_2(n))$ is know for this problem. However, Rainbow sort [29] can do the sorting in O(1), which is unimaginable fast.
- Standard computers consume too much energy. In California, the second consumer of electrical energy are the computers (first one being electrical bulbs) [7]. Something must be done here too because in the near future the number of computers will increase several times worldwide. Note that with one liter of DNA we can perform much more computations that all computers in the world have performed so far. The reduction in energy usage is significant in this case.
- The discovery of a new algorithm requires a huge work. In some cases it can take years to develop a good algorithm. The nature has solved a lot of problems with its own algorithms. (for instance problems with requires sorting or the computation of the shortest path). Taking these natural algorithms as the source of inspiration could lead to a faster development of practical algorithms for today's problems.

4. Problems to solve

There are 2 main categories of problems that we want to solve with unconventional devices.

- Those which are difficult for standard computers. I am refereeing here mainly to NP-complete problems [13] because most of these problems are very practical. More discussion over the NP-complete problems is given in section 5.

- We are also interested in some devices offer direct advantages over digital computers for some particular problems. This aspect is only interesting if the benefits are huge (see for instance Rainbow sort [29] which does sorting in O(1)).

5. NP-complete problems

The main category of problems that we want to solve with UC devices are NP-complete problems [13]. Unfortunately no polynomial-time is known for them. Nor the exponential algorithms has been proved to be optimal. This field lives in a great uncertainty.

This is why the Clay Mathematics Institute offers a 1 million prize for a solution to the P=?NP problem. Even if the answer is negative we still have some great benefits: a big number of intelligent people will focus their attention on other problems.

The NP field was initiated by Stephen Cook which has made one of the most important contribution to computer science [8]. Since then the field has grown exponentially.

5.1. **Formal definition.** A problem C is said to be NP-complete if:

- Any given solution to the problem can be verified quickly (in polynomial time). The set of problems with this property is called NP.
- Every problem in NP is reducible to C in polynomial time.

Another well-know category is composed of NP-hard problems for which the second condition holds, but not necessarily the first one.

Some examples of NP-complete problems together with some practical applications are:

- Travelling Salesman Problem - with applications to path planning,
- Scheduling - with applications to machines and tasks,
- Subset sum - with applications to cutting and packing.

5.2. **How do we solve NP-complete problems ?** There are 3 major ways in which NP-complete problems can be solved:

- Brute force - which takes an exponential time because the size of the search space increases exponentially with the problem size.
- Heuristics - which are not optimal for all cases and even worse : require human intelligence.
- Unconventional devices - this is the aspect that we investigate in this paper.

5.3. **What benefits we get if we solve NP-complete problems?** Eloquent for this question is the answer given in 1956 by *Gödel* to $JohnvonNeumann$. We reproduce here this well-know phrase:

If there actually were a machine with [running time] $K * n$ (or even only with $K * n^2$), this would have consequences of the greatest magnitude. That is to say, it would clearly indicate that, despite the unsolvability of the Entscheidungsproblem, the mental effort of the mathematician could be completely (apart from the postulation of axioms) replaced by machines.

5.4. **Do we have to solve all NP-Complete problems?** The answer is simple: NO. One problem is enough because there is a polynomial time reduction between them [13]. Thus is enough to find a solution to TSP and all other problems all solved instantly.

6. Can Unconventional Computers solve NP-Complete problems in Polynomial time ?

This is a critical question. We already have evidence on small instances (see the Steiner tree with Soap Bubbles [6]). When solving larger instances of the Steiner tree problem (with soap bubbles) we get a lot of errors. Advocates of this method say that errors are due to imperfect experimental conditions.

Anyway, it is difficult to derive general statements because it is difficult to analyze the complexity of UC devices.

Some say that this is the perpetuum mobile of the modern times and what we should expect is to get solutions for particular problems.

7. Can analog computers do better than digital computers in all cases?

Unfortunately the answer to this question is negative. One major problem is related to the digital to analog conversion [33]. All unconventional devices are analog, thus this conversion is unavoidable if we want to make a fair comparison between conventional and unconventional computers.

Take for instance a simple problem: *Compare 2 numbers A_1, A_2 represented over n bits.*

One possible way to solve this problem in an analog is to create 2 analog objects having masses A_1 and A_2. Because we need to represent any mass between 0 and $2^n - 1$ we will need an exponential (in n) quantity of matter .

For these kinds of problems the standard (digital) computers will always perform better than their analog counterparts.

8. Physical implementation

A physical implementation is required for a better understanding of the mechanisms behind unconventional devices. For a computer scientist this could be very difficult because most of the phenomena not visible at macro-scale (take for instance a tube with DNA strings !).

Most of the UC methods requires a lot of equipments whereas mostly are expensive (for optical, DNA or quantum computers). Very few methods can be implemented with home-made tools (soap bubbles, linear programming machine etc).

Most of the work is done at theoretical level plus some simulations. Complicated experiments are rarely repeated (take as an example the Adleman experiment).

9. Comparing various UC devices

Comparing some poorly understood methods is a difficult task. In this section (see Table 1) we make a comparison based on the following aspects:

- Speed - how fast the solution of the problem is obtained.
- Size - the quantity of materials involved in the experiments. Note that the size of most devices is exponential (see section 7 and paper [33]).
- Price - the total cost of the materials involved in the experiments. Since prices can vary depending on the producer we have used some general labels: low - for few dollars, and high for more than hundreds or thousands dollars. The price includes the equipment for reading the output which can be sometimes the most expensive part (see for instance Rainbow sort [29] where sorting is very cheap, but reading the output is very expensive).
- Know how - how much knowledge is required for building such device.
- Number of problems - the number of problems that can be solved by the device.
- Approx. solution - if the device generates an approximate solution (e.g. has a heuristic behaviors) or an exact solution.

10. Advantages and Weaknesses

Unconventional methods have a huge number of advantages over the standard methods, but also have some weaknesses which is the main reason for which they cannot be seen yet in real-world applications.

Some of the advantages are:

- Parallel Computing - can solve problems in parallel, exploring multiple solutions in the same time.

Table 1. A comparison of the major unconventional computing paradigms.

Method	Speed	Size	Price	Know how	Number of problems	Approx solution
DNA computing [2, 27, 34]	Constant	Exponential (see [20])	High	High	Many	NO
Quantum computing [10, 31, 32]	sub-exponential (hopefully)	Polynomial	High	High	Many	NO
Optical devices [19, 18, 24, 25, 28, 30, 37, 38]	Exponential (see HPP)	Exponential (see HPP)	High	High	Many	NO
Bubble soap [1, 6]	No known	Polynomial	Low	Low	1 (Steiner tree)	YES
Rainbow sort [29]	Very fast	Exponential	High	Low	1 (sorting)	NO
Spaghetti sort [14]	Polynomial	Exponential	Low	Low	1 (sorting)	NO
Bead sort [3, 12]	Polynomial	Exponential	Low	Low	1 (sorting)	NO
Protein folding machine [5, 15, 21, 22, 26]	Not known	Polynomial	High	High	1 (protein folding)	Not known
Smart glass [23]	Polynomial	Exponential	Medium	Medium	Multiple	NO
Time travel computing [1]	Instant	Polynomial	Not known	Not known	Multiple	NO
Linear programming machine [33]	Exponential	Exponential	Low	Low	1 (linear programming)	NO

- Light weight - in some cases the UC methods operate at nanoscale level.
- Low power - working on nanoscale means a very low power consumption.
- Solves Complex Problems quickly - in some cases the solution is find in an extremely small time (see Rainbow sort [29]).

Major weaknesses are:

10.1. Weaknesses.

- Preprocessing time - A lot of time is spent for preparing the input for the unconventional device.
- Sometimes slower - Simple problems are solved much faster on electronic computers.
- Reading the output - It can take longer to read the answer to a problem than it takes to solve the problem itself. Thus the advantage of speed is virtually eliminated.
- Reliability - can have errors (see DNA computing). These error require extra corrections or too advanced tools which makes the computation impractical.

11. Unconventional Computing community

The UC community is growing every year. More and more papers are published on UC topics. The technology is improving, thus allowing us to perform larger and larger experiments with an increasing precision. In the next sections we give a short (but significant) list of journals and conferences focused on UC topics.

11.1. **Journals.** Main journals publishing papers in the field of unconventional computing are given in Table 2. Other journals focused on theoretical computing or on physics, chemistry, biology can also publish papers about UC topics.

11.2. **Conferences.** There is a huge number of conferences and workshops on Natural Computing topics. All of them accept papers about UC paradigms. Also, most of the Artificial Intelligence conferences accept papers about unconventional computing machines due to the strong connection between AI and UC (see sectionaiuc). It is impossible to list here all these conferences.

However, the leading conference in this field is Unconventional Computation which was started in 1998.

TABLE 2. A short list of journals publishing papers on Unconventional Computing.

Journal	Publisher	Starting year	Issues / year
Natural Computing	Springer	2002	4
New Generation Computing	Springer	1982	4
International Journal of Unconventional Computing	Old City Publishers	2005	4
Theoretical Computer Science	Elsevier	1975	61 (in 2008)
Fundamenta Informaticae	IOS Press / EATCS	1977	20
Journal of Universal Computer Science	Graz University of Technology and Universiti Malaysia Sarawak	1994	22 (in 2008)
International Journal of Foundations of Computer Science	World Scientific	1990	6
New Mathematics and Natural Computing	World Scientific	2005	3

12. Conclusions and further work

In this paper we have investigated some of the most important analog devices capable of performing computations. Future work will be focused on:

- Comparing the proposed paradigms from a practical perspective.
- Checking the current limits of the compared methods. These limits can be advanced once the technology improves.
- Finding killer applications [9] for each of the paradigms. A killer application proves the value of the technology and pushes for more development of the corresponding technique. An example of killer application is *VisiCalc* for Apple II, which has generated huge sales for those platforms. Currently, for quantum computers we have a killer application: integer factorization [31] which runs in sub-exponential time. This is why big players (such as IBM and other private companies) have tried to implement quantum computers. For DNA computing a potential killer application has been proposed in [34]: a nano-scale robot which can fix diseases inside of a cell. For other paradigms more investigation should be performed in order to discover killer applications.

Acknowledgment

This work was supported from grant IDEI-543 from CNCSIS, Romania.

References

[1] S. Aaronson, *NP-complete Problems and Physical Reality*, $arXiv : quant - ph0502072v2$, 2005
[2] L. Adleman, *Molecular computation of solutions to combinatorial problems*, Science, 266, (1994), 1021-1024
[3] J. J. Arulanandham, C. S. Calude, M. J. Dinneen, *Bead-Sort: A natural sorting algorithm*, EATCS Bull, 76 (2002), 153-162
[4] C. H. Bennett, *Logical reversibility of computation*, IBM Journal of Research and Development, 17:6 (1973), pp. 525-532.
[5] B. Berger, T. Leighton, *Protein folding in the hydrophobic-hydrophilic (HP) is NP-complete*, Proceedings of the second annual international conference on Computational molecular biology, (1998) pp. 30-39.
[6] S. Bringsjord, J. Taylor, *P=NP*, arXiv:cs.CC/0406056 v1, 2004
[7] H. J. Caulfield, *Optics Goes Where No Electronics Can Go: Zero-Energy-Dissipation Logic*, in Optical SuperComputing Workshop, S. Dolev, T. Haist, M. Oltean (Eds.), LNCS 5172 (2008) pp. 1-8.
[8] S.A. Cook, *The complexity of theorem proving procedures*, Proceedings, Third Annual ACM Symposium on the Theory of Computing, ACM, New York, (1971) pp. 151-158,
[9] P. Dasgupta, *Killer Applications (overview)*, Arizona State University in Tempe, webpage: http://cactus.eas.asu.edu/partha/Columns/2002/07-01-killer-app.htm, last accessed on April 16, 2009.
[10] D. David, *Quantum Computation*, Science 270 (1995), pp. 255-261.
[11] Iwamura, H., Akazawa, M., Amemiya, Y. (1998), *Single-electron majority logic circuits*, IEICE Trans. Electronics E81-C(1), (1998) pp. 42-48.
[12] A. J. Jeyasooriyan, *Ball Sort - A natural algorithm for sorting*, Sysreader (1995), pp. 13-16.
[13] M. Garey, D. Johnson, *Computers and Intractability*, W.H. Freeman, New York, NY, 1979.
[14] A. K. Dewdney, *On the spaghetti computer and other analog gadgets for problem solving*, Scientific American 250:6 (1984), pp. 19-26
[15] A.S., Fraenkel, *Protein folding, spin glass and computational complexity*, Theoretical Computer Science, 48 (1999), pp. 101–121.
[16] M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* New York: W.H. Freeman, 1979.
[17] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.
[18] J.W. Goodman, *Architectural development of optical data processing systems*, Aust. J. Electr. Electron. Eng. 2 (1982), pp. 139-149
[19] T. Haist, W. Osten, *An Optical Solution For The Traveling Salesman Problem*, Optics Express, 15, (2007), pp. 10473-10482
[20] J. Hartmanis, *On the weight of computations*, Bulletin of the EATCS 55 (1995), pp. 136-138
[21] J. W. Mills, *Introduction to the EAC with Protein Folding Example*, PPT presentation, accessed 4 April 2009

[22] J. T. Ngo, J. Marks, *Computational complexity of a problem in molecular-structure prediction*, Protein Engineering, 5:4 (1992), pp. 313-321.

[23] M. Oltean, *Switchable Glass: A possible medium for Evolvable Hardware*, NASA conference on Adaptive Hardware Systems, Stoica A., et al. (Eds), IEEE CS Press (2006), pp. 81-87.

[24] M. Oltean, *A light-based device for solving the Hamiltonian path problem*, Unconventional Computing, Calude C. (et al.) (Eds), LNCS 4135, Springer-Verlag (2006), pp. 217-227

[25] M. Oltean, *Solving the Hamiltonian path problem with a light-based computer*, Natural Computing, Springer-Verlag, 8:1 (2008), pp. 57-70.

[26] P. Crescenzi, D. Goldman, C. Papadimitriou, A. Piccolboni, M. Yannakakis, *On the complexity of protein folding*, Journal of Computational Biology, 5:3 (1998), pp. 423-65.

[27] R. S. Braich, N. Chelyapov, C. Johnson, P. W. K. Rothemund, L. Adleman, *Solution of a 20-Variable 3-SAT Problem on a DNA Computer*, Science, 296 (2002), 499-502.

[28] J.H. Reif, A. Tyagi, *Efficient parallel algorithms for optical computing with the discrete Fourier transform primitive* Applied optics 36(29) (1997), pp. 7327-7340.

[29] D. Schultes, *Rainbow Sort: Sorting at the speed of light*, Natural Computing, Springer-Verlag, 5(1) (2005), pp. 67-82.

[30] N.T. Shaked, S. Messika, S. Dolev, J Rosen, *Optical solution for bounded NP-complete problems*, Applied Optics, 46 (2007), pp. 711-724.

[31] P. Shor, *Algorithms for Quantum Computation: Discrete Logarithms and Factoring*, IEEE Symposium on Foundations of Computer Science, IEEE Press (1994), pp. 124–134.

[32] M. Steffen, M.K. Lieven, I.L. Vandersypen Chuang, *Toward quantum computation: a five-qubit quantum processor* IEEE Micro, 21 (2001) pp. 24-34.

[33] A. Vergis, K. Steiglitz, B. Dickinson, *The complexity of analog computation* Mathematics and Computers in Simulation, 28 (1986), pp. 91-113.

[34] Y. Benenson, B. Gi, U. Ben-Dor, R. Adar, E. Shapiro, *An autonomous molecular computer for logical control of gene expression*, Nature 429 2004, pp. 423-429.

[35] L. Zadeh, *Fuzzy sets and systems* In: Fox J, editor. System Theory. Brooklyn, NY: Polytechnic Press, (1965), pp. 29-39.

[36] J. Zurada, *Introduction to artificial neural systems*, West Publishing Co. St. Paul, MN, USA, 1992.

[37] D. Woods, T.J. Naughton, *An optical model of computation* Theoretical Computer Science, 334(1-3) (2005), pp. 227-258

[38] *Thoughts on the Subset Sum Problem (P vs. NP)* $http://idrone.net/2006/06/11/thoughts-on-the-subset-sum-problem-p-vs-np/$ (accessed 2006)

Department of Computer Science, Faculty of Mathematics and Computer Science, Babeş-Bolyai University, Kogălniceanu 1, Cluj-Napoca, 400084, Romania

E-mail address: moltean@cs.ubbcluj.ro