Portland State University

PDXScholar

11-2014

Unconventional Computing Catechism

Christof Teuscher
Portland State University, teuscher@pdx.edu

Citation Details

Unconventional computing catechism

Christof Teuscher *

Department of Electrical and Computer Engineering, Portland State University, Portland, OR, USA

What makes a new paradigm or technology promising? What should science, research, and industry invest money in? Is there a life after CMOS electronics? And will the vacuum tube be back? While one cannot predict the future, one can still learn from the past. Over the last decade, unconventional computing developed into a major new research area with the goal to look beyond existing paradigms. In this Perspective, we reflect on the current state of the field and propose a set of questions that anyone working in unconventional computing should be able to answer in order to assess the potential of new paradigms early on.

Keywords: unconventional, non-classical, computing, technology, promise, catechism

New technologies typically go through cycles. A good example is neural networks, a movement that was pioneered by McCulloch and Pitts, Rosenblatt, Hebb, and others some 60 years ago. The new field started off with great promise, if not hype, before it received a major setback when it was shown by Minsky and Paper in 1969 that a single-layer perceptron cannot even solve all simple logic functions. Despite the initial setback, neural networks, and more generally machine learning, are used today very successfully in many real-world applications. Gartner's technology hype cycle (Fenn, 2008) has five key phases that describe the maturity of a given technology: (1) Technology Trigger, (2) Peak of Inflated Expectations, (3) Trough of Disillusionment, (4) Slope of Enlightenment, and (5) Plateau of Productivity. The graphical hype cycle representation provides a tool to view how a technology will evolve over time. Using Gartner's terminology, mainstream neural networks have nowadays reached the Plateau of Productivity. "When new technologies make bold promises, how do you discern the hype from what's commercially viable? And when will such claims pay off, if at all?" (Fenn, 2008).

Unconventional computing (also *UCOMP*, *non-classical, non-standard, alternative*, or *next-generation computing*) (Stepney et al., 2005; de Castro, 2006; Amos et al., 2012; Cerf, 2014) is an emerging research field with the goal to go beyond traditional computing technologies and paradigms, such as the von Neumann architecture or the Turing model. Examples of unconventional computing paradigms include quantum computing, optical computing, molecular computing, and chemical computing. While these approaches may be able to perform classical computations, it is often not a natural way to do so. As opposed to seeking incremental (or evolutionary) changes of current paradigms, unconventional computing seeks revolutionary changes by using novel substrates, formalisms, and paradigms. Naturally, the more revolutionary a technology is, the more risk it bears. Critics of unconventional computing commonly argue that (1) the community has not produced a useful paradigm that beats a conventional approach and (2) that the real, practical challenges to be solved are always kept comfortably far away, e.g., in the 10- to 20-year time frame. Using Gartner's terminology again, unconventional

computing technology seems to have trouble crossing the Trough of Disillusionment and ending on the Plateau of Productivity.

However, supporters of unconventional computing argue that comparing unconventional with current state-of-the-art technology is simply not a fair comparison. After all, state-of-the-art technology typically is the result of decades of multi-billion dollar efforts while most unconventional approaches have been a few years in the making, often on a shoestring budget at best.

The best example is probably the CMOS transistor. For example, in 2009 the semiconductor industry spent $200 billion on research (Apte and Scalise, 2009). In the same year, the US National Science Foundation (NSF) received $6.49 billion only. One can only wonder how much any emerging computing technology could have been advanced with $200 billion. But what technology would you have picked?

In his 2006 paper (Borkar, 2006), Borkar outlined three tenets that made electronics evolution successful: (1) gain, (2) signal to noise ratio, and (3) scalability. Applying these three tenets to past and present electronics, he concluded that "[.] there is nothing on the horizon that has promise to replace CMOS at least in the next ten to fifteen years." We believe that this is still true in general and perhaps, as Sery et al. (2002) stated, "Life is CMOS" and it is here to stay for the foreseeable future? While this is indeed a plausible option, it still leaves plenty of niches for unconventional computing approaches to excel. For example, not every unconventional technology needs to be scalable. Future biomolecular computers may use a few hundred logic gates only to perform, e.g., some basic functions to analyze blood sugar levels and to control the release of some drugs. First and foremost, such on-body systems need to be bio-compatible and highly reliable. Speed and scalability, on the other hand, are not an issue.

In this Perspective, we would like to propose a new set of questions that are inspired both by *Heilmeier's Catechism* (Shapiro, 1994) as well as observations of the unconventional computing field over the last decade. A "catechism" is a list of principles in the form of questions and answers that are used to educate people. George H. Heilmeier argued that anyone proposing a new project should try to answer those questions. Compared to Heilmeier's

catechism, the new questions we propose here are specific to unconventional computing. While we focus on physically realizable computing machinery, the field of unconventional computing is broad and also encompasses new theoretical models of computations that may not necessarily be grounded in physical reality, e.g., models that rely on infinite resources, time, or precision.

UNCONVENTIONAL COMPUTING CATECHISM

1. What challenge (or problem or application) are you trying to address with an unconventional computing approach?
2. What are the metrics for meeting that challenge?
3. What are the fundamental limits to computing you should be concerned about?
4. How is the system controlled and programed?
5. How do you interface with your unconventional system?

While there are certainly many other questions, one might ask, I consider – given the current state of the field – the five above the most important to ensure further progress.

In the following, we will briefly explain these questions. And as common with an actual catechism, we shall also provide some answers, or at least further issues to consider. Naturally, it is the task of the proposer of the unconventional computing approach to provide answers to these questions because general answers are hardly possible. Instead, they will be highly dependent on the challenge at hand.

THE CHALLENGE

Do something, just because it can be done, does rarely lead to breakthrough innovations. For example, we could easily find some strange device that can perform simple logical operations in some way. Yet, what insights have we gained by doing that beyond a simple proof-of-concept? That can admittedly be valuable, but the real challenges often become apparent only when one starts to integrate the components into a larger system. Instead of focusing on small proof-of-concepts, we believe that it is now time for community to tackle existing larger-scale challenges, problems, and applications in a very focused and goal-oriented way. A good example is high performance computing. Traditionally, supercomputers rely on thousands of processing units that crunch data. Of course, such an approach is the most efficient if the data are mostly independent. However, recent "big data" challenges have somewhat changed the game because a lot of that data is highly interrelated, and therefore, hard to crunch on that traditional type of parallel machine (Wright, 2014). In addition, there is a huge gap between the speed of the processing units and the memory, resulting in memory bottlenecks because data need to be constantly retrieved and stored. What is the best solution for that challenge? Can some unconventional computing approach become the solution?

METRICS

In order to compare unconventional computing approaches with other approaches, one needs well-defined metrics. Good metrics help to track progress and to define success. And as with any bold claim, bold evidence is needed. While you may propose a new device that operates five times faster than the state-of-the-art

device, it may not be useful if it consumes 20 times more power. The metrics question must also address such trade-offs and also address potential (killer) applications. With the chosen metrics, how would your approach perform on these applications? Can it outperform existing approaches in at least one aspect? And how do the metrics scale as a function of the system size? Your system might work well at a small scale, but hit fundamental scaling limits (e.g., combinatorial explosion) as you increase its size.

FUNDAMENTAL LIMITS

Trying to prove whether or not $P = NP$ is likely as hopeless as trying to physically realize a hypercomputer (Davis, 2004), a computer that can compute functions (such as the halting problem) that a classical Turing machine cannot. It is key to be aware of possible loose and tight theoretical and practical fundamental limits in order to weed out any "mission impossible" projects early on. A recent paper by Markov (2014) summarizes the relevant fundamental limits for emerging technologies. Lloyd (2000) introduced several *ab initio* limits and the "ultimate laptop" in his 2000 paper. While loose limits might be, well, loosened up by playing certain tricks, tight limits will not give in. For example, the accelerating Turing machine (Teuscher and Sipper, 2002) can solve the Halting Problem in linear time in theory, but it is obviously not possible to physically implement such a machine because that would eventually require operations that are infinitely fast, at least if the observer and the computer are in the same reference frame.

PROGRAMING

Many unconventional substrates, i.e., a soup of DNA strands, *E. coli*, or ant colonies offer interesting dynamics, but it is unclear how we can harness them for a useful purpose, i.e., to compute a function in the sense of a Turing machine. Crutchfield et al. (2010) described this issue as bridging the gap between *intrinsic* and *designed* computation. The deeper issue is related to control and programing of intrinsic dynamics, which would then lead to designed (or useful) computation. As Stepney states in Stepney (2012), "[o]ur ability to exploit unconventional computing is partly hampered by a lack of corresponding programing formalisms: we need models for building, composing, and reasoning about programs that execute in these substrates."

INTERFACING

Interfacing conventional with unconventional computers can be both challenging and be a show-stopper. For example, the output signals of biomolecular computers are typically relying on fluorescence. On the other hand, the input signals may be represented by certain chemical concentrations that need to be injected into the system at very specific instants. Needless to say that interfacing such an unconventional computer with a conventional digital computer is non-trivial. Interfacing also commonly adds significant overhead, which might nullify any possible advantage of an unconventional approach as one moves to a more integrated system level. Worse, a significant portion of the overall computational effort could be performed in the interfacing part. This is one of the reasons why the community should not stop with simple proofs-of-concept. We need complete solutions, a simple logic gate that performs a universal NAND operation is a great start, yet it is not sufficient.

In the semiconductor industry "[r]evolutionary innovation has been missing in action for about 40 years as the industry instead focused on incremental advances" (Apte and Scalise, 2009). What we truly – and possibly even desperately – need at this point is a new "transistor" or "landing on the moon" moment that would get computers *as we know them* to the next level. Most people would agree that the "next big thing" is not going to be some incremental improvement, but it will be something radically different. Yet, precisely what secret sauce of devices, architectures, and compute paradigms will get us there is speculation at best. How do we make the right decision in the right direction?

First and foremost, the unconventional computing community needs a willingness of the funding agencies to continue to invest in bold and radically different ideas. As Gros (2012) argued, "[s]upporting a range of small to medium research projects instead of a few large ones will be, as a corollary, a more efficient use of resources for science funding agencies." Second, the community needs to focus on the key issues and address the important questions above with focus and determination. Too often, research projects stop after a proof-of-principle is obtained, e.g., we can compute a NAND function, so know we can compute any function. While this is true in theory, the practical engineering obstacles are often significant. Building an actual computing architecture from simple NAND gates may prove to be practically infeasible. Or, it may result in a system that is terribly inefficient.

We would like to advocate for the community to go beyond simple proofs-of-concept. To make an unconventional approach successful, we need to ultimately cross the *valley of death* between government funding for research and industry support for prototypes and products. That is easier said than done, yet, with answers to the above questions, the mission can be kept focused.

While performance measures and comparisons are undoubtedly needed to evaluate new technologies, "[t]he most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" (Weiser, 1991). When was the last time you thought about your car's *Anti-lock braking system* (ABS)? Who remembers the times when new drivers were told to "pump the brakes" in order to stop fast and to avoid skidding at the same time? Today, we have sophisticated electronics taking care of just that, including many other functions in your car you never even thought about. Is ABS electronics based on unconventional approaches? Not in today's terms. It is conventional electronics combined with well-established algorithms that do not even have to particularly fast or sophisticated. Yet, we expect the system to be extremely reliable.

The point we would like to make here is that what we consider unconventional is ultimately a matter of perspective. The ultimate metric for an unconventional technology to consider successful may simply be when that technology becomes conventional.

We have talked about technology cycles above. Things come and go, and things get re-invented periodically. Here is a recent example: "The vacuum transistor could one day replace traditional silicon" (Han and Meyyappan, 2014). Is the idea of combining transistor with vacuum-tube technology unconventional? And does it matter? It is a cool idea no matter what, which has shown some promise. As with most other new approaches, it needs to be shown that it can scale up, which the authors say is the next step.

So unconventional computing may simply not be the wisest choice of words for exciting new computing technology. As Apte and Scalise conclude: "the challenge today is in finding sources of disruptive scientific innovation" (Apte and Scalise, 2009). Whether it is unconventional or not ultimately does not matter. Now let us all go out and seek new adventures in computing!

REFERENCES

Amos, M., Stepney, S., Doursat, R., and Vico, F. J. (2012). Truce: a coordination action for unconventional computation. *Int. J. Unconv. Comput.* 8, 333–337.

Apte, P., and Scalise, G. (2009). The recession's silver lining. *IEEE Spectr.* 46, 46–51.

Borkar, S. (2006). "Electronics beyond nano-scale CMOS," in *Proceedings of the Design Automation Conference*, (New York: The Association for Computing Machinery), 807–808.

Cerf, V. G. (2014). Unconventional computing. *Commun. ACM* 57, 7. doi:10.1145/2656433

Crutchfield, J. P., Ditto, W. L., and Sinha, S. (2010). Introduction to focus issue: intrinsic and designed computation: information processing in dynamical systems-beyond the digital hegemony. *Chaos* 20, 037101. doi:10.1063/1.3492712

Davis, M. (2004). "The myth of hypercomputation," in *Alan Turing: Life and Legacy of a Great Thinker*, ed. C. Teuscher (Berlin: Springer-Verlag), 195–212.

de Castro, L. N. (2006). *Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications*. Boca Raton, FL: CRC Press.

Fenn, J. (2008). *Understanding Gartner's Hype Cycles*. Available at: http://www.gartner.com/technology/research/methodologies/hype-cycle.jsp

Gros, C. (2012). Pushing the complexity barrier: diminishing returns in the sciences. *Complex Syst.* 21, 183–192.

Han, J.-W., and Meyyappan, M. (2014). The device made of nothing. *IEEE Spectr.* 51, 30–35.

Lloyd, S. (2000). Ultimate physical limits to computation. *Nature* 406, 1047–1054. doi:10.1038/35023282

Markov, I. M. (2014). Limits on fundamental limits to computation. *Nature* 512, 147–154. doi:10.1038/nature13570

Sery, G., Borkar, S., and De, V. (2002). "Life is CMOS: why chase the life after?," in *Proceedings of the 39th Design Automation Conference*, (New York: The Association for Computing Machinery), 78–83.

Shapiro, J. (1994). George H. Heilmeier. *IEEE Spectr.* 31, 56–59. doi:10.1109/6.284787

Stepney, S. (2012). Programming unconventional computers: dynamics, development, self-reference. *Entropy* 14, 1939–1952. doi:10.3390/e14101939

Stepney, S., Braunstein, S. L., Clark, J. A., Tyrrell, A., Adamatzky, A., Smith, R. E., et al. (2005). Journeys in non-classical computation I: a grand challenge for computing research. *Int. J. Parallel Emergent Distrib. Syst.* 20, 5–19. doi:10.1080/17445760500033291

Teuscher, C., and Sipper, M. (2002). Hypercomputation: hype or computation? *Commun. ACM* 45, 23–24. doi:10.1145/545151.545170

Weiser, M. (1991). The computer of the 21st century. *Sci. Am.* 265, 66–75. doi:10.1038/scientificamerican0991-94

Wright, A. (2014). Big data meets big science. *Commun. ACM* 57, 13–15. doi:10.1145/2617660