There are no new good ideas in computing; everything was done in the 50s, 60s, and 70s : a counter-argument

Abstract

In this paper, I make the argument that there have been and continue to emerge, good new ideas in computing, since the 70s; some more revolutionary than others. I argue that most if-not-all ideas in computing and science in general have emerged from a "parent-idea" and what makes some of them new is the number and complexity of new challenges they have to offer.

What is a "new good idea in computing" ?

> " "[W]e so readily assume that discovering, like seeing or touching, should be unequivocally attributable to an individual and to a moment in time. But the latter attribution is always impossible, and the former often is as well. [...] discovering [...] involves recognizing both that something is and what it is."
> -- *Thomas S. Kuhn, The Structure of Scientific Revolutions, 2nd Ed, 1970.* [Kuh70]

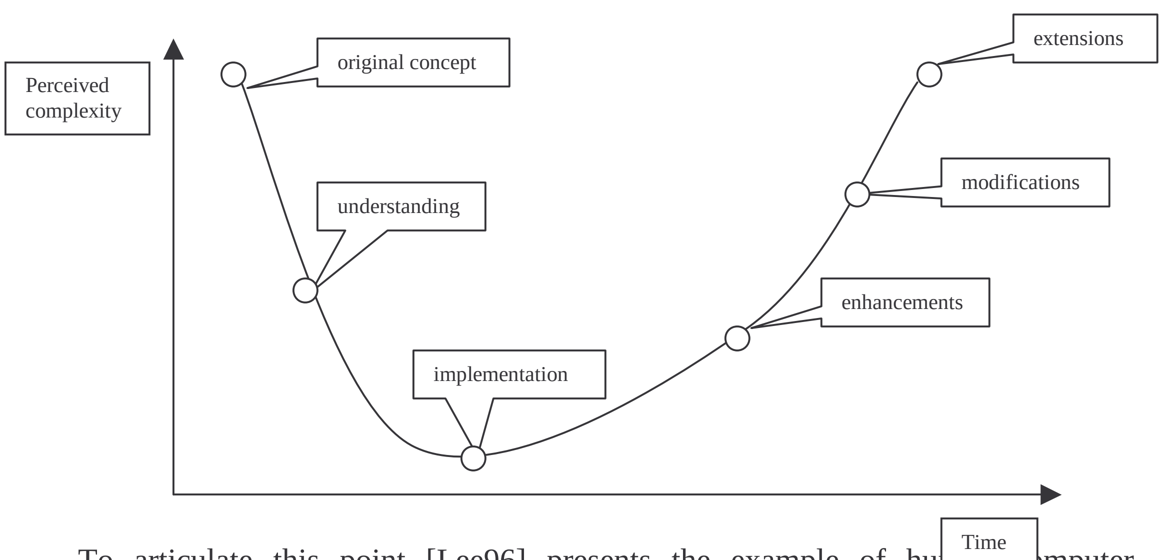
In the same manner, there is no particular time or person who should be credited with the discovery or creation of theoretical computer science. However, there are important steps along the way towards the consolidation of the study of systematic resolution of problems as a science. "

-- Comp.Theory FAQ (Alex Lopez-Ortiz and Daniel Jimenez) [Lop]

We have to begin by defining what is a "new good idea". Is it an idea that just drops out of the sky, has no background to prior developments in computing and provides a new perspective to computing ? Or is it an idea derived from and based on previous work in computing ? If it is derived than is it only a minor improvement over the parent or is it the spawning of a whole new field as complex as the parent itself ? There is also the question of time. Can we identify distinctly when an idea came into existence or does good idea develop over time. Take the example of *fuzzy logic*. In 1964, Dr. Lofti Zadeh came up with the term "fuzzy logic". So is that the year the idea came into being ? One can however argue, that it emerged for the first time when Indian scholar Buddha, in 500 BC, proposed that almost everything contains some of its opposite, or in other words, that things can be A and not-A at the same time. Like any good study, let us lay down some assumptions/constraints/ground-rules, which will address the above questions and also make the problem-space more tractable.

In his landmark work on the history of science entitled *The Structure of Scientific Revolutions* [Kuh70], Thomas Kuhn defines *paradigms* as "universally recognized scientific achievements that for a time provide model problems and solutions to a community of practitioners". He suggests that paradigms are sufficiently unprecedented to attract an enduring group of practitioners away from competing modes of scientific activity and also sufficiently open-ended to leave all sorts of problems for the redefined group of practitioners to resolve. This definition of a "paradigm" will be considered as the definition of a "good new idea" for the rest of the discussion and used as such in the rest of the text.

An examination of many of the central concepts of science from birth to maturity shows a perceived complexity versus time curve, which is roughly parabolic [Lee96]. In [Lee96] the author argues, that in the beginning, the initial idea is often expressed in such complex terms that few can grasp it. Those who do then express it in a manner that can lead to implementation and application. As the resulting system is used and explored, it is again modified and enhanced to achieve a new level of applicability and complexity. At this point the original concept can be fragmented into *child-concepts* that each offer levels of complexity similar to the original. We can label these as *new paradigms*.



To articulate this point [Lee96] presents the example of human-computer interaction :

- The emergence of the concept (mid-1970s)
- The emergence of a set of related disciplines around the central theme (late-1970s)
- A multidisciplinary approach (1980s)
- Interdisciplinary integration (1990s)
- Specialization and fragmentation (ongoing ?)

For the purpose of discussion, we can now identify two types of paradigms in computing. A *hard paradigm* is one that brings a brand-new insight and direction to computing based on expertise in other fields such as chemistry or biology. A *soft paradigm* can be one that is an extension of an existing paradigm in computing but by itself satisfies the definition of a paradigm. In other words, it is at the tail end of the parabola and has substantial perceived complexity and following.

We therefore say that to qualify as a "new good idea in the post-70s era" , an idea has to:

- meet the definition of *paradigm* (hard or soft) described earlier
- be published/presented for the first time in post-70s era for peers to review and accept.

The following sections will give examples of such ideas to articulate the point that such ideas are emerging.

Case study of a *Hard Paradigm*: DNA Computing

On December 29th 1959, at the annual meeting of the American Physical Society at the California Institute of Technology (Caltech), Richard Feynman, in his visionary talk described the possibility of building sub-microscopic computers [Fey59]. He did however make it clear, that he did not know how to build something on that small scale in a practical way. Decades later in November of 1994, Leonard Adleman published his seminal paper on DNA computing titled, *Molecular Computation of Solutions to Combinatorial Problems* [Adl94]. In this paper Adleman used the tools for molecular biology to solve an instance of the directed Hamiltonian path problem. A small graph was encoded in the molecules of DNA and *operations* of the computation were carried out in a test tube with standard protocols and enzymes. This experiment successfully demonstrated the feasibility of carrying out computations at the molecular level and opened up a whole new field of research. He showed that not only were molecular computers highly energy efficient, they also have potential for enormous storage density. He followed it up with another paper [Adl95] wherein he gave a view of how molecular computer may be constructed. It was still an open problem to elucidate all the kinds of algorithms, which are possible using molecular computers and the kinds of problems these algorithms can solve. In [Lip94] Lipton showed that DNA computing could be used to solve any NP problem. He further illustrated this technique in detail in [Lip95] by solving SAT using biological computing. To speak in terms of the model of *idea-evolution* described earlier, we can say that as our understanding of DNA computing deepened, at least two new fields have emerged from it; one that deals with the construction of a commercially practical DNA computer [Oly02] and the other dealing with algorithmic research in DNA computing.

Based on the above facts, I contend that DNA computing :

- is a hard paradigm since it was a very original approach to computing and enamored a large number of dedicated researchers and offers substantial and diverse future challenges
- originated after 1970s (first publication in 1994)

Case study of a *Soft Paradigm* : Grid Computing

Grid computing is a form of distributed computing that involves coordinating and sharing computing, application, data, storage, or network resources across dynamic and geographically dispersed organizations. The parent idea for grid computing is quite old. In the early 1970s when computers were first linked by networks, the idea of harnessing unused CPU cycles was born [GriH]. In 1973, the Xerox Palo Alto Research Center (PARC) installed the first Ethernet network and the first full-fledged distributed computing effort was underway. John Shoch and Jon Hupp wrote a program, moving from machine to machine using idle resources for beneficial purposes [GriH]. Even upto 1984 though, the total number of connected hosts was a little over 1000. So the paradigm of grid computing as seen today could not have been fully conceived before that time.

The term “Grid” was coined in the mid-90s to denote a proposed distributed computing infrastructure for advanced science and engineering [Fos99]. While Grid technologies are currently distinct from other trends such as Internet, enterprise, distributed and p2p computing, these trends can significantly benefit from research in Grid technologies [Fos01]. The specific problem underlying the Grid concept is coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations [Fos01]. There is a dedicated research and development community that is working on protocols, services and tools that address precisely this problem.

Miscellaneous

Affective computing, is another new area of research, which combines human psychology, AI and human-computer interaction. The basic theme of the research is to have the computer recognize the emotions of the user and interact accordingly [Pic95].

Dynamics Based Computation [Sin98] is yet another new paradigm that uses the ability of lattices of coupled chaotic maps to perform computations.

Conclusion

All ideas in computing have evolved from prior knowledge and research. Only some of them qualify to be *paradigms*. In this paper I have argued that such paradigms have definitely emerged in computing in the post-70s era. Although I have cited only a few examples here, my contention is that a lot of new ideas we see around us today will fit the definition of either a hard or soft paradigm; and that the examples should only serve as a guideline for our search.

References

[Kuh70] Thomas Kuhn (1970). *The Structure of Scientific Revolutions.* 2nd Edition. University of Chicago Press.

[Lee96] John Lee (1996). “*Those Who Forget the Lessons of History Are Doomed to Repeat It” or, Why I Study the History of Computing.* IEEE Annals of the history of Computing, Vol. 18, No. 2.

[Lop] Alex Lopez-Ortiz, Daniel Jimenez. *Comp.Theory FAQ*. http://db.uwaterloo.ca/~alopez-o/comp-faq/faq.html

[Ieee] IEEE Computer Society, *History of Computing*. http://www.computer.org/history/development/

[Adl94] Leonard M. Adleman (1994). *Molecular Computation of Solutions to Combinatorial Problems.* Science, Vol. 266.

[Adl95] Leonard M. Adleman (1995). *On constructing a Molecular Computer*. http://citeseer.nj.nec.com/cache/papers/cs/10/ftp:zSzzSzusc.eduzSzpubzSzcsinfozSzpaperszSzadlemanzSzmolecular_computer.pdf/on-constructing-a-molecular.pdf

[Fey59] Richard Feynman (1959). *There's Plenty of Room at the Bottom.* http://www.zyvex.com/nanotech/feynman.html

[Lip94] R. Lipton (1994). *Speeding up computation via molecular biology*. Unpublished manuscript, Princeton University.

[Lip95] R. Lipton (1995). *Using DNA to solve NP-complete problems.* Science, Apr 1995.

[Oly02] http://www.bio-itworld.com/news/030702_report26.html

[Sal84] J. Saltzer, D. Reed, D. Clark (1984). *End-to-end arguments in system design.* ACM Transactions on Computer Systems, Nov. 1984.

[GriH] http://www.grid.org/about/gc/evolution.htm

[Fos99] I. Foster, Carl Kesselman (eds.). *The Grid: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann, 1999.

[Fos01] I. Foster, Carl Kesselman, S. Tuecke (2001). *The Anatomy of the Grid.* International Journal Supercomputer Applications, 2001.

[Pic95] R. Picard (1995). *Affective Computing.* MIT Media Laboratory Technical Report 321. ftp://whitechapel.media.mit.edu/pub/tech-reports/TR-321-ABSTRACT.html

[Sin98] Sudeshna Sinha and W. Ditto (1998). *Dynamics Based Computation.* Physical Review Letters, Vol. 81, No. 10.