Computer Science and Global Economic Development: Sounds Interesting, but is it Computer Science?

Tapan S. Parikh
UC Berkeley School of Information
Berkeley, CA, USA
parikh@berkeley.edu

OVERVIEW

Computer scientists have a long history of developing tools useful for advancing knowledge and practice in other disciplines. More than fifty years ago, Grace Hopper said the role of computers was "freeing mathematicians to do mathematics." [5] Fred Brooks referred to a computer scientist as a *toolsmith*, , making "things that do not themselves satisfy human needs, but which others use in making things that enrich human living." [4]. Computational biologists have applied algorithmic techniques to process and understand the deluge of data made possible by recent advances in molecular biology.

The proper way to approach this kind of research has never been clear within Computer Science. The refrain "Sounds interesting, but is it Computer Science?" is frequently heard. In this paper I argue that it is crucial take an expansive view of what Computer Science is. To do otherwise is to cede a great deal of exciting, high-impact research territory to others - research that is technically interesting and has the potential to impact many lives. I believe the tension felt by our community in conducting "technical research" for global economic development provides a critical opportunity to re-examine our conceptions of our discipline.

CS AND GLOBAL DEVELOPMENT

My students and I are focused on solving development problems through the innovative use of computing technologies. What gets us out of bed in the morning is the opportunity to have impact on individuals and communities living in poverty in the developing world, and on the institutions that serve them. This is not to say we are uninterested in technical problems — in fact, that is what we bring to the table — the ability to design, implement and deploy computing systems within constraints not often felt in other contexts — including limited user literacy, physical infrastructure and organizational capacity. However, for us, solving technical problems is a means to an end, not an end in itself. If we can solve the latter without contributing anything hugely novel to the former we would do (and have done) so.

Information and communication technologies (or, ICTs) carry great promise for development practitioners and researchers alike. Governments, NGOs and businesses see them as tools to communicate with target communities and their staff, to document and learn from their own interventions, and from those working in the same region or on similar issues. Recently, development economists have recognized the paucity of applicable theory, turning their focus on designing, identifying, and evaluating the impact of new interventions from the bottom up, usually applying experimental methods [1]. This includes prominent use of ICTs, both as the focus of new interventions (mobile phones for making markets more efficient [6], digital cameras to monitor teacher attendance [2]), and as tools for understanding their impact (PDAs and smartphones to conduct extensive in-field surveys [8]). It is a wonderfully timely moment for computer scientists to engage with the state-of-the-art in development research and practice.

WHY ACADEMIA? WHY CS?

Why do this work in academia, and within the discipline of Computer Science? There are several motivations. Academia allows us to be more free, and take greater risks, than other institutions. A commercial approach must (eventually, at least) be accountable to the tyranny of revenue and profits. A government approach is constrained by bureaucracy, and by an inability to take risks, including learning from (often, small-scale) experiments. In academia we are able to experiment with new ideas, without being guaranteed of their popularity or success, and without knowing a priori the long-term sustainability model.

The most compelling alternative is the non-profit sector — non-governmental organizations, or NGOs. NGOs probably have the best record of ICT innovation in support of rural economic development. The fact that NGOs have outdistanced academia in an area with such potential for important technological innovation, with just a fraction of the financial and especially human capital, stands as much as an indictment of our record, as a justification of theirs. However, there are important limitations to their approach as well. NGOs are usually not rewarded for nor capable of methodological and empirical rigor. It is important to develop tested theory and methods to inform technology design and implementation, and that allow us to generalize to new applications and operating contexts.

Why not do this work within other disciplines — such as Public Health, Economics or Education? After all, aren't we solving their problems? Simply put — researchers in these fields are most comfortable working with existing technologies, and not designing and implementing new ones. They have neither the skills nor the training to pursue that agenda, which could lead to entirely new opportunities and innovations. Moreover, CS students can learn a lot from doing this work, and are demanding opportunities to do so. Freedom, innovation, scientific rigor and opportunities for students — sounds like a good fit for academia!

BUT, IS IT COMPUTER SCIENCE?

It is time to return to that refrain “Sounds interesting, but is it Computer Science?” The ontological (what objects and phenomena we study) and epistemological (how we study them) bases of the field have been the subject of recent reflection. Ammon Eden distinguishes three paradigms of computer science [3]. Of fundamental importance is the phenomenon to be studied. The mathematics branch holds that the objects to be studied are algorithms — abstract mathematical objects, properties of which can be determined through deductive reasoning. The engineering branch holds that what we study are running computing programs, which are more complex, such that their properties can only be determined by seeing how they perform after the fact.

The final branch is the scientific one, best captured by Allen Newell and Herbert Simon — “Computer science is the study of the phenomena surrounding computers... an empirical discipline... an experimental science... like astronomy, economics, and geology.” [7] These phenomena are not limited to algorithms, or even running programs (though they traditionally have been within mainstream CS), but can include the human, social and physical processes surrounding them. This interpretation is well-established within Human-Computer Interaction (HCI). By accepting it across Computer Science, especially as other areas are becoming increasingly user and application-driven, we bring a consistent philosophical basis to the field, including for engaging with other disciplines. In this paradigm, we advance hypotheses (often in the form of new computing technologies or applications, and/or their variants), and validate them using experimental methods. Note that it is not only the novelty of the technology that is relevant, but the importance and generalizability of the knowledge derived about them. CS research that advances the goals of global development, conducted in a scientific manner, for understanding the appropriate design, implementation, cost, impact, and usage of new computing technologies, clearly fits the bill.

One possible complaint is that if we stray too far from the *algorithm*, the theory will be insufficient for explaining the phenomena we study. I find this critique unsatisfying, and in fact already a fait accompli. Is there only one set of theories or concepts that underlies economics, biology or sociology? Naturally, the study of complex systems requires drawing upon multiple strands of theory, and drawing from other disciplines. The proper study of computing can (and does) draw upon theory from economics, sociology, psychology, cognitive science, neuroscience, and other areas. We should embrace this complexity, and learn from other disciplines, or doom ourselves to decreasing relevance by ignoring the panoply of interesting computing phenomena surrounding us.

BROADER RAMIFICATIONS

New Research Problems

We can create opportunities to attack a number of important, high-impact research problems for which we have the necessary expertise and tools.

New Publishing Models

Doing good science requires conducting studies that can take years to plan, implement and analyze. This is suitable for publishing in journal format, as opposed to the conference format common in Computer Science. By publishing more journal articles, we can improve our academic standing (for example, in tenure cases). By addressing problems important to other domains, we can publish articles in their conferences and journals also.

New Job Prospects

We can improve the job prospects of our students, both within academia and outside of it. For example, economics PhDs work in economics departments, business schools, schools of public policy, for NGOs, multi-lateral agencies, or governments; depending on their interests and the kind of dissertation they have published.

New Funding Opportunities

We can approach new funding opportunities, in collaboration with those disciplines that have experience with them. Computer scientists increasingly collaborate with medical researchers to apply for funding programs sponsored by NIH.

CONCLUSION

All these outcomes require a deep engagement with problems considered outside the realm of traditional Computer Science. It is pointless to go halfway — our new community must commit to the problems we are addressing, instead of trying to “squeeze” technical nuggets out of them. In this paper, I have argued that this shift is completely possible, by choosing a broader ontological and epistemological basis appropriate for the range of problems that Computer Science is now addressing. By taking this expansive view, we will not only contribute to other disciplines, but also to our understanding of computing, and to its advancement as a mature professional discipline.

REFERENCES

[1] A. V. Banerjee and E. Duflo. The Experimental Approach to Development Economics. *SSRN eLibrary*, 2008.

[2] E. Duflo, R. Hanna, and S. Ryan. Monitoring Works: Getting Teachers to Come to School. *SSRN eLibrary*, 2008.

[3] A. H. Eden. Three Paradigms of Computer Science. *Minds Mach.*, 17(2):135–167, 2007.

[4] J. Frederick P. Brooks. The computer scientist as toolsmith II. *Commun. ACM*, 39(3):61–68, 1996.

[5] G. M. Hopper. The education of a computer. In *ACM '52: Proceedings of the 1952 ACM national meeting (Pittsburgh)*, pages 243–249, New York, NY, USA, 1952. ACM.

[6] R. Jensen. The Digital Provide: Information (Technology), Market Performance, and Welfare in the South Indian Fisheries Sector. *Quarterly Journal of Economics*, 122(3):879–924, 2007.

[7] A. Newell and H. A. Simon. Computer science as empirical inquiry: symbols and search. *Commun. ACM*, 19(3):113–126, 1976.

[8] K. Shirima et al. The use of personal digital assistants for data entry at the point of collection in a large household survey in southern Tanzania. *Emerging Themes in Epidemiology*, 4(1):5, 2007.