On long-range planning for the CS department

There are two related, but distinct topics: the future of CS and the future of the CS department. They are distinct topics because the former can be discussed independently of local constraints, while the latter one cannot. Let us briefly discuss the future of CS first, because, without a clear view of that, I cannot meaningfully discuss the future of the department.

The future of Computing Science

To avoid misunderstanding, when I refer to CS I do not refer to its social aspects (such as how it can be sold, how it can supposedly be useful, what benefits are expected from it, etc.). Nor do I refer to it as viewed by the politicians, the majority of the laymen (or by the majority of the not-so-laymen for that matter). Instead, when I refer to CS I refer to an intellectual discipline as characterized by its internal structure (which gives it its coherence), its challenges, and its opportunities, to which list I should add - because we are discussing CS - its novelty.

As a further preliminary, I should point out two different ways of coping with novelty. Most "novelty" is not radical but gradual. To cope

with the latter, we can try to understand it in terms of what is familiar and to relate it to past experience: using metaphors and analogies we can plan today for tomorrow in yesterday's terminology. This is by far the most common situation, and therefore the situation most people are most comfortable with. In the case of radical novelty, however, this familiar way of "understanding" does not work, because our past experience is irrelevant and the analogies are so shallow that they become more misleading than illuminating. Radical novelty can only be approached with a "tabula rasa", by not trying to relate it to earlier experience but by trying to come to grips with it via its internal structure, properties, consistency, what have you. (The latter way of understanding is less popular because it requires hard work.)

Technology is expert at inflicting radical novelties upon mankind. Nuclear weapons and the pill are examples from the last 50 years; so is in my opinion the automatic computer, and this not so much in its capacity of a tool - in that capacity it causes only a ripple on the surface of our culture — as in its capacity of a challenge that is absolutely without precedent in the history of Mankind.

Radical novelties are, almost by definition, rather unsettling. There are people for whom gradual

change is the only paradigm of history and incremental improvement the only paradigm of progress. For them, radical novelty is most unsettling, because invisible.

The combined assumption that, for the majority of people, gradual change is their primary historical paradigm and that computers embody a radical novelty explains all sorts of strange phenomena. I mention a few examples.

(i) The popularity of "Software Engineering." Millions of dollars -and pounds, for that matter!- have been poured into it without any tangible benefit. It will never bear fruit, as analysis of the SE literature reveals that it has accepted as its chapter the self-contradicting target "how to program if you can't", and that, by calling it "software production", it treats programming as a production process -in which programmer productivity is measured in SLOC's (= Source Lines of Code) per month-. This grandiose idiocy, however, is popular because it has couched the frighteningly unfamiliar in reassuringly familiar terms, no matter how inadequate.

(ii) The mathematician's neglect of automatic computing. Naively one would have expected the mathematical community to have welcomed this new opportunity to show the indispensable value

of its well-honed intellects, but nothing of the sort! Instead, it regularly declares that CS is so trivial that it neither deserves the name of a science, nor any of their valuable attention. In fact, they skirt the issue because facing it would frighten the wits out of them.

(iii) The popularity of AI. One would expect people to feel threatened by "the giant brains". In fact, the frightening computer becomes less frightening if it is only used to simulate a familiar noncomputer (no matter how imperfectly).

So much for radical novelty and the general difficulty - including our own - of dealing with it. Let us turn our attention to our announced topic, the future of Computing Science.

When all is said and told, the only thing a computer can do for us is to manipulate symbols, but it does so on a scale that is without precedent. To do so, it has to be programmed, and the only known way of getting these programs is ..... by symbol manipulation, which, for a sizeable program, is of an uncommon amount. For me, this observation forces two conclusions, a qualitative and a quantitative one. The qualitative one is that CS is unavoidably concerned with the interplay between mechanized and human symbol manipulation. The quantitative one is that CS is unavoidably concerned with the effective application of formal tech-

niques on a scale very few people, as yet, have dreamt of. Whether we like it or not, here is the core challenge that Computing Science as intellectual discipline faces: how to apply – mechanized or humanly – formal techniques effectively and on a much larger scale than ever envisaged before. By its availability, the equipment provides both the challenge and the opportunity. The challenge is unavoidably ours because neither the mathematicians nor the logicians are inclined to pick up the gauntlet.

The mathematicians don't pick up the gauntlet because, by and large, the mathematical guild is very informal. Whole books have been devoted to the right of the "real mathematician" to wave his hands. They are the prisoners of a tradition of more than 25 centuries and continue to cling, but for a small minority, to the consensus model of mathematical truth.

The logicians don't pick up the gauntlet either. They are in the process of transferring logic from "the handmaiden of philosophy" and a mimicking of human reasoning to an alternative to human reasoning. Their focus, though, is on what logical systems can and can not achieve "in principle" and the effective application of formal techniques, "logic as engineering discipline", so to speak, is beyond the commitment of their community as a whole.

In short, I present a vision of CS as a branch of formal mathematics and of applied logic that will in due time transcend its parent disciplines by effectively realizing a significant part of Leibniz's Dream of providing symbolic calculation as an alternative to human reasoning.

Needless to say, this vision is not universally shared. On the contrary, it has met wide-spread -and sometimes even violent- opposition from all sorts of directions. I mention

- the mathematical guild that would rather continue to believe that the Dream of Leibniz is an unrealistic illusion
- the business community that, having been sold to the idea that computers would make life easier, is mentally unprepared to accept that they only solve the easier problems by creating much harder ones
- computer engineering that would rather continue to act as if it is all only a matter of bitrates and flops
- all soft sciences for which computing now acts as some sort of interdisciplinary haven
- the educational business that feels that, if it has to teach formal mathematics to CS students, it can as well close its schools.

Wide-spread as it is, the opposition is no ground for concluding that there is something

wrong with the vision in question. The opposition merely reflects that there is a definite upper bound for the rate at which society as a whole can absorb intellectual progress (if it can do so at all). Intellectual revolutions are as unwelcome as political ones.

I should add that the vision is by no means universally rejected. In some form or another, it is a source of inspiration for quite a few colleagues —here and elsewhere—, who act accordingly, both in their teaching and in the direction of their research. (Admittedly, they are a minority, but so was Galileo Galilei.)

Educational policy

Educational policy is trivial if the needs of society coincide with what society ask for. Such coincidence, regrettably, is extremely rare. In the case of computing science —and in view of the topic's radical novelty, this is not amazing at all— the discrepancy between needs and requests is truly dramatic. I have seen requests for more "advanced COBOL" in the curriculum, general complaints that the undergradute curriculum is "too innovative" and neglects "fundamentals" such as "basic business practice and awareness, the invoice cycle and the nominal ledger";

we should pay more attention to the teaching of "communication skills", "teamwork", "group dynamics", "human psychology" and "innovation management". If these noises come primarily from computer users, those from computer manufacturers are equally ignorable. (For instance, in 1975, IBM has explained at length to the -Canadian- universities that they should not teach operating systems principles, because "work on OS is not an entry-level job". IBM was clearly afraid of employees knowing how operating systems should be constructed. I trust that the rest of the computer industry is somewhat more enlightened, but have all reason to assume that on the whole it is not significantly so. Certainly where programming is concerned, the computer industry -with the possible exception of INMOS- has not yet fathomed the consequences of being involved in a high-technology industry.) So, by and large, we are on our own!

We are faced with the traditional teacher's dilemma that whenever we succeed in doing our job well, we have delivered misfits. We may be faced with the dilemma in extreme form; yet I call the dilemma "traditional", because no one can found a new "school of thought" without delivering graduates that differ.

One of the ways of solving the dilemma is by avoiding it : don't rock the boat! An extreme example is provided by the ultraconservative university of Cervera, Spain, whose Rector reassured the visiting monarch Ferdinand with the words "Far be from us, Sire, the dangerous novelty of thinking.". (This was in the previous century, around the time Texas gained its independence from Mexico. Cervera did not become a leading university and, were UT to follow its example, it could be argued that the war with Mexico has been fought in vain.)

Another way of trying to solve the dilemma is by joining an established "school of thought" that still shows signs of vigour, and by trying to find recognition in that league. In our industry I think it is known as "the Amdahl strategy". For an American CS department, the analogue would be cloning, say, Stanford/MIT/CMU/Berkeley, all rolled into one. If we wish to consider the Amdahl strategy for the CS department at UT we must consider at least two issues, a tactical one and more fundamental one.

The tactical one is concerned with our chance of success, in particular: is there any chance of successfully competing with those four (i) as long as Texas is viewed in the USA as Down Under is viewed in the world, and (ii) when, competing for

the same competence -both in faculty and students-, as those four- we can expect to have to recruit from their leftovers ? (I am still a stranger to these shores and, hence, without such a thing as a considered opinion on this question. My impression is that the answer might very well turn out to be negative.)

The more fundamental point is: do we really believe that those four have faced the dilemma, do provide what society needs and serve computing science as it deserves to be served ? If not, there is little point in just offering more of the same.

Finally, if the best we can reach via the Amdahl strategy is becoming the leading clone, it raises the question whether we can thus meet our charter of becoming "a leading department".

If the conditions are satisfied -or seem satisfiable- we can try the complement of the Amdahl strategy; we might call it the anti-clone strategy. Instead of trying to imitate the big four, one tries to distinguish oneself from them, the clearer, the better. To be successful, it has to be done clearly, for it boils down to a "redefinition" of CS as popularly conceived.

Two technical conditions have to be satisfied.

Firstly, there are one or more areas or aspects that are neglected by the big four while deserving a better fate. Secondly, our department must have the corresponding strengths or it must be conceivable to build them up. (Cornell seems to have opted for such a strategy when it decided to focus on "robotics"; I have second thoughts about that choice: the topic is a bit small and rather peripheral, and -cf. MIT and CMU- only mildly different. I am afraid that time will show that Cornell has been too timid.) We should realize that such areas -if they exist- are of necessity unfashionable and unpopular. (Note that the unpopularity would also be our protection: it will delay the moment others join the bandwaggon, and when they do, we know, in a way, that we have succeeded.)

Off hand, I can think of one candidate that seems to meet the technical conditions, viz. "the effective, large-scale application of formal methods": it is neglected by the big four; it lies at the core of computing science; UT's department has considerable strength in this direction; it is sufficiently unpopular —it is in fact more unpopular in the USA than elsewhere, and that is good: it will delay the big four in joining the bandwaggon while our graduate population is sufficiently international not be hurt by it,

probably on the contrary.)

We would not be alone either: it would put is in league with Oxford and Edinburgh (and later, as the case may be, Groningen or perhaps CalTech). None of those I consider deadly competitors.

Whatever area is chosen, the adoption of an anti-clone strategy will cause internal problems in any department that adopts it. Individuals can take more extreme positions than whole departments can, and it is therefore a safe assumption that only a part of the faculty will be affected by or involved in such an anti-clone strategy. The danger that others feel left out in the cold is only too real. The challenge is how to avoid needless pain in the implementation of such a strategy without sabotaging it.

Austin, 7 November 1988

prof. dr. Edsger W. Dijkstra
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712 - 1188