

How to Ruin the Career of a Ph.D. Student in Computer Science and Engineering: Precise Guidelines

Milutinović, Veljko; Tomažič, Sašo

In our profession, one widely utilized Ph.D. thesis structure implies the following elements¹:

- (1) Introduction: motivation and environment
- (2) Problem statement and why it is important
- (3) Existing solutions and their criticism
- (4) Proposed solution that is both application and technology aware
- (5) Details of the solutions to be compared
- (6) Conditions and assumptions of the analyses to follow
- (7) Mathematical analysis
- (8) Simulation analysis to compare performance
- (9) Implementation analysis to compare complexity
- (10) Conclusion from the performance/complexity ratio point of view
- (11) Acknowledgements
- (12) References

In order to have these elements ready when writing the thesis and papers (after the research and development are completed), one has to perform specific work (before the research and development are completed). Working on each one of the above defined elements, Ph.D. advisors can and often do make errors that force students to undertake activities and create habits that could form irreversible damages to their research mentality and ruin forever the chances that they become real scientists in the future. The text to follow discusses the major

pitfalls of each methodological element in the environment typical of Computer Science and Engineering (CSE), using the following template: (a) an advice leading to a pitfall (in italic), (b) a short explanation shedding an extra light, and (c) an illustrative example.

(1) *Tell the student that the Ph.D. thesis is the crown of a research career.*
Since things in CSE change so rapidly, the main motivation for a Ph.D. should not be to create her/his lifework; it should only be treated as a proof that she/he is able to solve scientific problems using scientific methodologies when the real research starts after the Ph.D. thesis is defended. It is the fact that many Ph.D. research activities, for a variety of reasons, take too long; at some universities, especially in East Europe, in the past, some researchers would obtain the Ph.D. degree not long before they retire.

(2) *Tell the student to select the hardest problem and to go after the optimal solution.*
In computer engineering there are many unsolvable problems, however, the things still work; therefore, the candidates do not always need to find the optimal solution; a suboptimal solution that works is also good. The fact is that some problems have many elegant and easily understandable, but unfortunately wrong solutions; do not select such a problem for a Ph.D. research; also, do not choose a problem that is not important for the present day technology and applications (it is the responsibility of the Ph.D. thesis advisor that the student selects an important problem to work on, and the advisor, rather than a student, is to be blamed for missing directions).

¹ V. Milutinovic, "The Best Method for Presentation of Research Results," IEEE TCCA Newsletter, September 1997, pp. 1-6.

- (3) *Tell the student that one has to master all existing solutions to the problem before one makes an attempt to create something novel.*

Knowing too much about existing solutions may mislead the student and will minimize the chances that she/he generates a breakthrough (it is good that in CSE people are less prone to following the opinions of recognized authorities); reading too much will definitely lock the student inside the avenues already taken by others. A well known example is that Nobel Laureate Marconi discovered that short waves do bounce off the ionosphere, because he dared to do the related experiments in conditions when nobody else dared, because a guru of the field published a paper 'proofing' that something like that is not possible; in this case, the inventor did not know about all existing work in the field, and that certainly helped.

- (4) *Tell the student that he, as a newcomer into the field, will never create good new ideas by herself/himself alone (without the help of the major professor); only experienced experts can create breakthrough ideas.*

Actually, the fact is that the accumulated knowledge (which may not be relevant any more) could create blocking obstacles in the process of our creative thinking and decision making; well educated newcomers into a field do not have this type of problem. Fortunately, in CSE, things change rapidly, and it is not difficult to be a well educated newcomer, e.g., operating system Linux was created by a student and turbo codes were invented by a mechanical engineer

- (5) *Tell the student that one should never share the details of an invention with others, because they will steal it and abuse it.*

Some researchers do not go to conferences (time waste), and publish their work only in journals (that brings the SCI credit, which is typically a formal requirement for oral defense); the fact is that one obtains the best ideas when trying to explain the initial ideas to others.

Fortunately, in CSE, people tend to be more open to sharing, which is clearly demonstrated by large open source and freeware communities.

- (6) *Tell the student to keep simplifying the problem until it becomes tractable.*

It is the fact that narrowing the assumptions and conditions of the research increases the probability that one creates something novel, but narrowing beyond the absurd line turns the underlying assumptions into wrong assumptions, since the contact with reality gets lost; if one lives 24 hours with the Ph.D. thesis problem, and is obsessed with it, one will create original solutions without introducing any technology and application restrictions.

Researchers in CSE tend to oversimplify, especially when doing analytical work, since mathematical modeling works there not nearly as efficiently as in other fields, like physics or similar.

- (7) *Tell the student to avoid any simplification of the problem.*

Do not be overambitious, since it is impossible to take all relevant details into account. Some simplification is necessary in order to arrive to a solution. What is important is to distinguish between important, not so important, and unimportant.

By being too ambitious, one typically creates a useless result, or does not create the result at all, as is still the case with the attempts to create artificial intelligence comparable to human intelligence.

- (8) *Tell the student that one does not have necessarily to be a good programmer, if doing a Ph.D. in CSE; software tools will do the necessary job.*

Some researchers advocate that the purpose of Ph.D. research is to create ideas, not programs. The fact is, however, that one has to touch and feel the problem (e.g., by mastering the programming related details), before being able to create an effective simulation environment.

If one wants to find an efficient solution to a problem, she/he has to understand the essence and limitations of programs underlying the tools she/he uses.

- (9) *Tell the student that she/he should produce an ideal implementation, since academic implementations (those including bugs, errors, and stupidities of the un-experienced) are worthless.*

Actually, PoC type (proof of concept)

implementations are the best enablers of extremely efficient market oriented industrial implementations. Trying is the best catalyst for breakthroughs.

PoCs are of special importance for CSE; some of the products of major software companies stay in the PoC phase throughout their lifetime cycle, until the new successful release comes (actually, the new release becomes successful only the concept was proved by previous versions).

- (10) *Tell the student that price and performance are the only important criteria for evaluation of the novel ideas.*

Performance and complexity do not represent the full set of important issues; sometimes other issues like availability, reliability, and feasibility are of greater importance.

Researchers in CSE do not care enough for these other issues; they are often treated as of secondary importance. Actually, “abilities” are typically much more important in technology and application considerations, and notoriously omitted. Only holistic approaches and solutions they create will survive technology and application revolutions.

- (11) *Tell the student that it is not recommended to apply for funds apart from doing it jointly with the major professor, since no funding agency would give the research money to a Ph.D. student alone; they will do it only if they see the major professor name on the application, too.*

It makes sense to give research money directly to PhD students (one can not teach an old dog new tricks). Also, Ph.D. students who rely on the exclusive guidance from the advisor will never become creators of breakthroughs (note however that an advice from an old dog can be very useful, indeed). Since CSE is changing so rapidly, some computer research sponsoring agencies do recognize the importance of this issue.

- (12) *Tell the students to include into the thesis text as many references as possible, and before completing the thesis, also to publish as many own papers as possible (for inclusion into the list of references and the CV).*

Knowing about all existing solutions takes lifetime and minimizes the chances to generate anything, while putting in more than one knows is dangerous. Quantity can never compensate for lack of quality; even worse, too much quantity can produce bad quality.

Actually, in some of the best universities of the World, researchers in computer area are judged for promotion based on only the best 3 papers: in such conditions, a researcher with 300 papers on the CV is judged based on only 3 she/he selects, and is obviously handicapped in comparison with another researcher who created only 3 papers using the same amount of time and creative energy (all of them superb, because she/he did not care to waste time on non-breakthrough ideas).

A general truth is that a fool would never agree to exchange his brain with the brain of a genius, since, fool as he is, he believes that his brain is more valuable. In computer engineering this non-trading might be a good decision to make, because things change so rapidly and one must be open minded (empty brain) to be able to accept and generate new ideas. Of course, this is an oversimplification to underline the fact that an unbiased mind is often times also a more effective mind, when it comes to computer engineering.