Research-Related Projects for Graduate Students as a Tool to Motivate Graduate Students in Classes Outside Their Direct Interest Areas

Vladik Kreinovich
Department of Computer Science
University of Texas at El Paso
El Paso, Texas 79968, USA
vladik@utep.edu

1. Formulation of the Problem

- Research is an important part of graduate studies.
- Usually, students select a research topic about which they feel passionate.
- The students willingly (and usually successfully) study for the classes which are directly related to this topic.
- Students are also required to take "breadth" classes.
- The relation of these classes to the student's research topic is indirect – and may not be clear to the student.
- *Example:* Theory of Computation class in our graduate Computer Science (CS) programs.
- *Problem:* students often do not do their best in the "breadth" classes.
- This deficiency often affects them later on, when they need the corresponding skills in their research.

2. Possible Solution: General Idea

- To solve the above problem, we make a project an important part of the class (and of the class grade).
- Main option: perform class-relevant research related to the topic of their future thesis or dissertation.
- This win-win idea:
  - helps students master the class,
  - helps with their research – and
  - sometimes even (eventually) leads to publications.
- In the talk, we present examples of such projects and related publications.
- These examples come from the Theory of Computation classes taught in 2010 and 2011.

3. When Theory Is Useful: A General Description

- In many practical situations, we have empirically successful *heuristic* algorithms and methods.
- Their success has no clear theoretical explanation.
- Thus, there is no guarantee that the corresponding method will work well in new situations.
- Also, it is not clear whether a modification or generalization of this method will work.
- In such cases, a theoretical justification can help:
  - it can lead to a better understanding of when this method works and when it does not;
  - this helps avoid wasting time on using this method in situations where it does not work;
  - it helps understand when a proposed modification or a generalization of the method will work.

4. When Theory Is Useful (cont-d)

- In the above discussion, we assumed that the method either works or not.
- Often, we also need to select the values of several parameters.
- Usually, the quality of the result (e.g., whether a method works) depends on this parameter selection.
- Once we have a theoretical explanation for the method, we can:
  - not only use this theoretical description to predict the method's quality for given parameter values,
  - we can also find the values of the parameters which are *optimal* for a given practical problem.
- How we can do that: by using known optimization techniques.

5. Towards Practical Applications of Computing

- The ultimate objective of computing is to help in solving practical problems.
- Computations are very precise, they process well-defined data according to well-defined algorithms.
- Thus, to use computing, we need to formalize the problem, i.e., describe the problem in precise terms.
- This is an important *first stage* in solving the practical problem.
- Once this problem is formalized, we need to come up with *an* algorithm for solving this problem.
- Designing such an algorithm is an important *second stage* of solving a practical problem.

6. Practical Applications of Computing (cont-d)

- The algorithm designed on the second stage is not always the most efficient one.
- The next step is to come up with faster, more efficient algorithms for solving the problem.
- This optimization forms an important *third stage* of solving a practical problem.
- On all three stages:
  - we need to formalize heuristic methods, and
  - we need to find optimal values of the parameters of these methods.
- In this talk, we give examples of research-related student projects from all 3 stages of applied computing.

7. First Stage: How to Formalize the Problem (Case of Intelligent Control)

- *Intelligent control:* transforming imprecise knowledge of an expert controller into an algorithm. Stages:
  1. formalize the meaning of the words like "small";
  2. combine these meanings into the meaning of the corresponding rules;
  3. combine these rules and transform these combined rules into an exact control strategy.
- *Hernandez et al.* showed how to select the least sensitive "exclusive or" operations.
- On the third stage, *Bravo et al.* explained success of current heuristics.
- This led to a new heuristic based on "exclusive or".
- Applications: better estimates of system failure rates (*Ferregut, Campos, et al.*).

8. First Stage: How to Formalize the Problem (Reasoning in Physics)

- Similar problems occur when we try to formalize the skills of expert researchers; *Gutierrez et al.* showed:
  - that when we apply the usual techniques to physicists' intuition,
  - we get known equations of physics such as Newton's equations.
- These intelligent techniques, of course, go beyond justifying well-known equations.
- *Example:* in addition to equations, physicists use intuition to dismiss meaningless ("abnormal") solutions.
- It is desirable to formalize the notion of "abnormality".
- An important step towards such formalization is presented in *Jalal-Kamali et al.*

9. 2nd Stage: Designing Algorithms and Theoretical Justification of Heuristic Algorithms

- One of the main objectives of *geophysics* is to find $\rho(x, y, z)$.
- *Usual approach:* take values $\rho(x, y, z)$ on a grid as unknowns.
- *Better approach:* look for a combination of thin vertical line elements that indefinitely down.
- *Cardenas at al.* theoretically explained this heuristic.
- *Trade-off:* some parameters $\vec{p}$ lead to better accuracy, some to better resolution; which $\vec{p}$ to choose?
- *Heuristic idea:* find the point with largest curvature on accuracy-resolution curve.
- *Sosa at el.* explained this heuristic – and came up with a more general family of heuristics.

10. Third Stage: Making Computations Faster

- Main objective: make computations faster.
- First, we try to come up with a faster algorithm for solving the *general* problem.
- Once the general algorithm is close to optimal, we must speed up *individual* computations.
- Once this is achieved, a natural next step is to optimize the way this algorithm is implemented on a computer.
- Finally, when this is optimized, the natural next step is to speed up the computers themselves.
- The projects dealt with all the stages of this optimization process.

11. How to Make Computations Faster (cont-d)

- Some algorithms are designed by scientists.
- However, there are more problems than scientists, so an automatic program synthesis is used.
- *Example:* NASA used it for deep space missions into the unknown.
- *Reyna et al.* showed how to make program synthesis generate faster programs.
- In optimization, we start with a box or an ellipsoid, and subsequently halve it.
- *Portillo et al.* showed that half-ellipsoids speed up computations.
- *Implementation:* cloud computing *Lerma et al.*
- *Hardware:* speed up require miniaturization, which leads to quantum computing (*Cuellar et al.*, *Nava et al.*).

12. Acknowledgments

This work was supported in part by:

- the National Science Foundation grant HRD-0734825 (Cyber-ShARE Center of Excellence);
- the National Science Foundation grant DUE-0926721;
- Grant 1 T36 GM078000-01 from the National Institutes of Health; and
- a grant on F-transforms (fuzzy transforms) from the Office of Naval Research.

13. References

- S. Bravo and J. Nava, "Mamdani Approach to fuzzy control, logical approach, what else?", *Proc. 30th Conf. of the North American Fuzzy Information Processing Society NAFIPS'2011*, El Paso, Texas, March 18–20, 2011
- R. Cardenas and M. Ceberio, "Efficient Geophysical Technique of Vertical Line Elements as a Natural Consequence of General Constraints Techniques", *Journal of Uncertain Systems*, **6**(2), to appear
- C. Cuellar, E. Longpré, and V. Kreinovich, Vladik, "Why L^2 Topology in Quantum Physics", *Journal of Uncertain Systems*, **6**(2), to appear
- C. Ferregut, Carlos, F. J. Campos, and V. Kreinovich, "Reducing over-conservative expert failure rate estimates in the presence of limited data: a new probabilistic/fuzzy approach", *Proc. NAFIPS'2011*

14. References (cont-d)

- E. Gutierrez and V. Kreinovich, "Fundamental physical equations can be derived by applying fuzzy methodology to informal physical ideas", *Proc. 30th Conf. of the North American Fuzzy Information Processing Society NAFIPS'2011*, El Paso, Texas, March 18–20, 2011
- J. E. Hernandez and J. Nava, "Least sensitive (most robust) fuzzy 'exclusive or' operations", *Proc. NAFIPS'2011*
- A. Jalal-Kamali, O. Nebesky, M. H. Durcholz, V. Kreinovich, and L. Longpr'e, "Towards Towards a 'Generic' Notion of Genericity: From 'Typical' and 'Random' to Meager, Shy, etc.", *Journal of Uncertain Systems*, **6**(2), to appear

15. References (cont-d)

- O. Lerma, E. Gutierrez, C. Kiekintveld, and V. Kreinovich, "Towards Optimal Knowledge Processing: From Centralization Through Cyberinsfrastructure to Cloud Computing", *International Journal of Innovative Management, Information & Production (IJIMIP)*, **2**(2), 67–72 (2011)
- J. Nava, J. Ferret, V. Kreinovich, G. Berumen, S. Griffin, and E. Padilla, Why Feynman Path Integration?", *Journal of Uncertain Systems*, **5**(2), 102–110 (2011)
- P. Portillo, M. Ceberio, and V. Kreinovich, "Towards an Efficient Bisection of Ellipsoids", *Proc. ITEA Live-Virtual-Constructive Conference "Test and Evaluation"*, El Paso, Texas, January 24–27, 2011

16. References (cont-d)

- J. Reyna, "From program synthesis to optimal program synthesis", *Proc. 30th Conf. of the North American Fuzzy Information Processing Society NAFIPS'2011*, El Paso, Texas, March 18–20, 2011
- A. Sosa, M. Ceberio, and V. Kreinovich, "Why Curvature in L-Curve: Combining Soft Constraints", *Proc. 4th Int'l Workshop on Constraint Programming and Decision Making CoProD'11*, El Paso, Texas, March 17, 2011