TAPTICS: A Taxonomy of Proof Techniques In Computer Science.

Arun Vasan
Department of Computer Science
University of Maryland
College Park, MD-20742

Abstract— **Computer Science is often criticized as lacking scientific proof techniques. However, our findings show that Computer Science has a plethora of techniques, which have been accepted by the *community* as proofs. Firstly, we characterize the Computer Science research community in a graph-theoretic framework. Then, we present a taxonomy of techniques accepted as valid proofs by this community. We illustrate an identifying feature of each classification wherever possible.**

I. Introduction

A typical comment made by researchers outside CS (abbreviations are considered the norm in Computer Science.) is, “ Fine, you folks build Quadrillium-9 processors with 1.9 TeraHz frequency and write obnoxiously junky and clunky software that doesn’t work, but are you really a science ? Do you have a structured research community ? How do you prove something in Computer Science ? ”

We answer this question with a structural characterization of the research community in Computer Science and a taxonomy of techniques accepted as proof in this community.

In addition to defending the honor of CS in the assembly of sciences, this work is bound to be of use to researchers in CS, who wonder how “ to be known in the community ”, as well. The taxonomy will also serve as a veritable tool-chest of sorts to budding CS researchers.

II. The CS Research Community (CSRC)

Intuitively, the CSRC bears remarkable resemblance to the mafia in its organizational structure. Like the mafiosi, it is a very closely knit set of people, who are hierarchically organized. For a new member to be accepted by the CSRC, the researcher first tries to join one sub-community in CS (like say Artificial Intelligence or Networks) and then works his way up. Every step of her career, she is reminded in a Godfatheresque manner: “Never go against the community”.

We now present a mathematical characterization of the CSRC. The CSRC can be represented as a directed graph $G = (V, E)$, whose vertex set V represents members of the community and edge set E represents the **Kisses-The-Ass-Of** (K-TAO or $k - \tau$) relationship. Therefore, if a member X kisses Y’s ass, there is an edge from X to Y in the graph.

When a student begins grad school, an edge is added from him to his advisor. A senior member of the community is characterized by many incoming edges, i.e, gets her ass kissed by many. The $k - \tau$ relationship is transitive, is not reflexive, and may or may not be symmetric.

The $k - \tau$ graph typically forms closely connected cliques, which are then inter-connected by leading researchers. By lead, we mean that they lead the newer members of the community like dogs on a leash.

Each of these cliques holds an annual conference, where all papers accepted will be those of members only. The total weight of a clique in the broader graph is directly proportional to the number of papers the conference rejects. Some cliques get together and publish journals too, where again the authors are mainly the editors themselves.

Given such a CSRC graph, research information has to flow between nodes for the community to thrive. CS Research has evolved several techniques, which the community accepts as a valid proof of an idea. We now present a taxonomy of such research techniques and illustrate their use in the community wherever possible.

III. Proof Techniques Used in CS

We classify techniques primarily used by computer systems researchers, although definite contributions from other communities are acknowledged wherever possible.

A. *Proof By Intimidation:*

Often used by senior members of the community in their papers. When author names and biographies are known to refreees, they typically don't pass judgements that are against members of the conference committee or editorial board of the journal. If the reviewing is blind, the authors cite atleast a dozen of their own papers as references and make a subtle statement: ”Be Warned. This is **my** paper. I am on the committee and don't forget a rejection easily.”

B. *Proof By Simulation:*

The darling of computer system designers, this proof technique saves them the trouble of actually building a system or measuring an existing system. The technique can be recognized by a statement which is lexicographically close to “We demonstrate the validitity of our idea with extensive simulations. Our results show -whatever we claim- to be true 98.765% of the time with 123.45% confidence”

C. *Proof by Obfuscation:*

A standard technique often used by the “theoretical”, “analytical”, and “formal” members of the community, it is characterized by referees tearing whatever hair remains on their otherwise wise heads. Finally, the referee gives the author the benefit of doubt, by declaring, “If I can't understand it, then I don't know if it is wrong”.

D. *Proof by Implementation:*

The community recognizes this technique by the signature, “As proof of concept, we have implemented a prototype system”. Practised by Computer System builders, this technique's uniqueness lies in convincing one of the lack of need for something as trivial as reality.

E. *Proof by Assumptions:*

This key principle behind this technique is, if you can't solve a problem with your tools, resize the problem by making assumptions so that it fits your tools. It is recognized by a pair of sentences, which may be seperated by few other sentences:

- For sake of {tractability, brevity, ease of exposition, . . . }, we make the following assumptions.
- Assume that -what we claim- is true. Under these assumptions, -what we claim- is true

F. *Proof by Transformations:*

Transform the problem by making use of other techniques (assumptions, intimidation, etc.) to a problem in an area as far from Computer Science as possible. This can easily be identified by the following lines: “ We cast this problem in an *I'll-be-damned-if-you-know-this* framework from Gobbledygook theory. Once this is done, Gobbledygook theory [13] states that the following result holds.”

Reference [13] will helpfully point the referee to “Advances in Goobledygook Theory and its Applications - Editors: Nestle and Hershey”

G. *Proof by Measurement:*

The ”performance” sub-community in CS, who study the performance of their systems (It should be emphasized they don't measure their own performance), is characterized by all members swearing an oath of undying loyalty to measurement-based proofs. The proof technique relies on measuring a system's characteristics anywhere, anytime, and anyhow. A typical proof would be: ”We measured the number of packets in the link 24 hrs a day, 7 days a week, for one whole month, without leaving the room. The steep fall at time 11:45 AM on the 12th of December was caused by a technician accidentally pulling the network cable out of the measuring computer. However, the overall trends confirm our claim”.

IV. Conclusion

We characterized the structure of the CS research community in a graph-theoretic framework. We presented a taxonomy of techniques accepted as proofs in the CS research community.

It is hoped that this taxonomy will give rise to new and more powerful techniques, which will then be accepted as valid proofs by the community. It is also hoped that the community does not decide to excommunicate the author for daring to defend it.

V. Acknowledgements

The author asserts exclusive rights over any discredit for this work. He absolves his past and current colleagues, mentors, and professors of any association with this piece of research.