Database Theory Column

Victor Vianu
University of California San Diego
vianu@cs.ucsd.edu

The last database theory column initiated a series of articles on the relationship between database theory and practice with a guest column by David Harel: *Will I be Pretty, Will I be Rich? – Some Thoughts on Theory vs. Practice in Systems Engineering*. We continue with an article by Christos H. Papadimitriou, based on his invited talk at *PODS '95*.

Database Metatheory: Asking the Big Queries

Christos H. Papadimitriou
University of California San Diego
christos@cs.ucsd.edu

Is "database theory" an oxymoron? Or is it a platitude? What is the fitness measure that decides the survival of ideas (and areas) in mathematics, in applied science, and in computer science? Which ideas from database theory during the past twenty-five years have influenced research in other fields of computer science? How many were encapsulated in actual products? Was the relational model the only true paradigm shift in computer science? Is applicability the only and ultimate justification of theoretical research in an applied science? Are applicability pressures really exogenous and unwelcome? Are negative results appropriate goals of theoretical research in an applied science —or are they the only possible such research goals? If scientific theories must be refutable, what are the "hard facts" that provide the possibility of refutation in the case of computer science?

1 Introduction

The phrase *theoretical computer science* often elicits a puzzled smile from a well-intended layman. I suspect that *database theory* has the same effect. If it is the field to which you have dedicated your life's work, this makes you stop and think. This paper is my one-time attempt to organize and register these thoughts.

2 Theory and its Function

The historical justification of theoretical computer science is well-known: After all, theoreticians have founded computer science (Turing, Gödel, von Neumann), and contributed crucially to its most celebrated triumphs (understanding how to write compilers, how to lay out chips, and how to design databases, to name three). However, historical arguments of this sort have pitfalls, because situations change and yesterday's truths are no more. Here I will concentrate on a less inductive argument.

In the context of an applied science, *theory in the broad sense* is the use of significant abstraction in scientific research, the suppression of low-level details of the object or artifact being studied or designed. This abstraction usually requires insight and ingenuity; for example, in economics it is challenging to capture the abstract nature of markets. In contrast, as Herbert Simon observed ([Si] p. 22), the high-level behavior of the computer is the only aspect that is directly observable, and in fact it is so in a most deliberate way. Computer theory, in the broad sense, is more than justified, possible, or desirable: It is *inevitable*. Virtually all computer research, including experimental research, is "theoretical" in this weak sense. This is even more true in database research, because abstraction is the essence and *raison d'être* of databases.

However, here we are concerned with *theory in the narrow sense,* the sense usually employed in computer science: the use of sophisticated mathematical techniques for developing, using, and analyzing mathematical models of the (possibly already significantly abstracted) artifact under consideration. Theory is needed to cure the *complexity* that plagues the artifact being studied or, more typically, designed. Simon [Si] observes that this complexity is not innate, but it is the result of the adaptation of the artifact to the complexities of the environment. For example, the complexity of an algorithm is an adaptation to the complexity of the problem posed to it, and the complexity of an operating system is the result of its adaptation to its usage pattern and the components' performance and cost. Of course, nowhere is this adaptation to the environment more prevalent and complexity-inducing than in databases, whose purpose is to *represent* parts of the environment, as well as to *interact* with other parts.

Theoreticians attack the complexity of the artifact in several distinct ways.

(a) They *develop mathematical models* of the artifact. Turing machines, formal languages, and the relational model come to mind as particularly successful (at least in the sense explored in Section 4) examples of such models from computer science.

(b) Since ours is a science of the artificial, abstract models can become reality: Theoreticians *propose complexity-reducing solutions* (typically, algorithms and representational schemes) that are derived from the mathematical models. This function of theory is what we usually mean by "synthesis" or "positive results." Such results must be actually verified by *experiments.* The Berkeley-IBM experiment in the middle 1970s [As, SWKH], which established the feasibility of relational databases, was implicitly suggested in database theory's most celebrated positive result, the formulation of the relational model [Co1].

(c) Theoreticians *analyze* the mathematical models to predict the outcome of the experiments (and calibrate the models). For example, the need and importance of normalization in relational databases, and the role played by dependencies in it, were amply predicted; the difficulty of query optimization, on the other hand, came as a surprise, and necessitated new model development, synthesis, analysis, and experiments.

(d) Finally, theoreticians *explore.* They develop and study extensions and alternative applications of the model, and they fathom its ultimate limitations. They introduce and apply more and more sophisticated mathematical techniques. They build a theoretical body of knowledge and an edifice of mathematical methodology that transcend the motivating artifact and model (presumably in anticipation of higher complexity, stemming from more complex environments yet to come). Exploration is usually guided by (individual or collective) aesthetics, taste, and sense of what is "important" and "relevant".

Model building, synthesis, and analysis are obviously and uncontroversially necessary parts of the research and discovery process in any science of the artificial. But exploration is what theoreticians do most often (and most gladly); predictably, it is also the aspect that is criticized most viciously. As it will become clear in the next sections, I believe that there are three distinct powerful arguments in defense of exploration: (1) *It has been historically beneficial to computer science;* (2) *in reasonable doses, it promotes the field's health and connectivity;* (3) *exploration and proving elegant theorems are natural and attractive activities, and so it would be wrong and futile to repress them.*

On the other hand, I believe that exploratory theoretical research activity (1) *can disorient the field and lead it into crisis, when it is disproportionately extensive in comparison to model building, synthesis, and analysis;* (2) *will not thrive if it consistently ignores practice;* and (3) *requires true discipline and honesty in its exposition, especially in avoiding frivolous and unchecked claims of relevance and applicability.*

3 On Negative Results

In mathematics, theorems are judged by their elegance and depth, as well as by their place in mathematics' long-term research program. In computer science there is an additional criterion: whether the result advances the complexity-reducing program of theoretical computer science in the specific application, or points out a setback in this regard.[1] This valid distinction between *positive and negative results* has been exaggerated and abused (normative terms have this potential). I want to make only a few disjointed observations on the subject.

I will start by noting that negative results are *the only possible* self-contained theoretical results. I mean this in the following sense: Positive results —complexity-reducing solutions such as algorithms and representation schemes— must be validated experimentally and can therefore be considered as mere invitations to experiment. I am aware that not all positive results are followed up by such experimental validation, but I think that such absence should be considered as a form of falsification.[2]

A related point is that successful exploratory theoretical research is bound to produce predominantly negative results. After all, delimitation (discovering that "that's all there is!") is the ultimate success in exploration. Identifying the limitations of a model is valuable information for further model-building, and for crystalizing the subject. For example, the prevalence of a few simple algorithms in concurrency control [GR, BHG] is supported by negative results severely delimiting the feasibly implementable solutions and establishing lower bounds on their informational cost [Pa1]. It would be interesting to know whether, in the case of concurrency control, the negative theoretical results influenced the direction of practice, or simply chronicled and justified *ex post facto* its choices. I strongly suspect the latter —theory often plays this role.

Is Cook's Theorem [Cook] a negative result? It makes an ingenious and unexpected connection between two theretofore unrelated ideas: nondeterministic polynomial-bounded computation and Boolean satisfiability (the related result due to Ron Fagin [Fa] makes such a connection between computation and logic even more directly). Therefore, it is positive as a *metatheorem,* in that it reduces the complexity not of the artifact, but of the mathematical landscape. Of course, seen as a result in the study of algorithms for satisfiability, it is a definite setback, although still valuable as a warning against futile research directions. So, negativity is to a large extent in the eye of the beholder. In fact, in *cryptography* the

[1] I believe that such a normative distinction between positive and negative results exists to a smaller degree even in pure mathematics, where disproof of an important conjecture can represent a setback to the research program as perceived by the community at the time. In other applied sciences this distinction is less prevalent than in ours, essentially because computer science is unique in its development of mathematical methodology for proving negative results. In my view this is one of the major intellectual achievements that sets computer science apart from other applied sciences. There is a growing body of research [BPT, PY] whose goal is to *export* this "negative methodology" to applied sciences (such as applied mathematics, game theory, and mathematical economics) that are less so endowed.

[2] I realize that this sounds like a very sweeping statement. As it will hopefully become much more clear later in this paper, my use of the term *falsification* is not a scientific (certainly not a moral) condemnation. Such "operational falsification" can be the result of being ahead of one's time, expositional style, timing, mediocrity in the experimental community, or luck. But I highly recommend the obvious prevention: doing your own experiments.

issue of negative vs. positive results is confused in a most deliberate and ingenious way. In contrast, Codd's Theorem [Co2], another celebrated result identifying two important concepts (this time relational algebra and relational calculus) is solidly positive because of its double implication that the calculus is implementable and the algebra expressive.

4 What is "Good Theory"?

The reader who expects to find in this section rules to be obeyed by anyone who wants to do good theory will be disappointed immediately, for on this subject I am influenced by the thought of Paul Feyerabend [Fe]:

> *"Science is an essentially anarchic enterprise.* [...] *There is no idea that is not capable of improving our knowledge.* [...] *The only principle that does not inhibit progress is* 'anything goes'*."*

Feyerabend's argument is that such major scientific advances as the defense of the Copernician model by Galileo broke even the most self-evident rules of scientific conduct (like "never introduce a new theory if the old is not in crisis" and "never introduce a new theory that contradicts available evidence predicted by the old").

Galileo was successful because of his propagandistic craftiness (being right probably helped too). For, although there is no such thing as "bad science",[3] *successful* is an important category for science. It is not an intrinsic category depending on the methods and results, but a much more complex predicate of the social dynamics of the field and its environment, and of course open to circumstance and chance. In fact, in describing successful science we are better off adopting metaphors not from sociology, but from *microbiology:* Scientific ideas blossom and thrive by invading and affecting other ideas, fields, and of course practice —influencing the practical milieu is the ultimate test of honesty and relevance in computer science research, and greatly enhances an idea's prestige and propagandistic value.

What does this all mean for theoreticians? First, free-style exploratory theoretical research is legitimate (essentially because nothing in science isn't). But its success will depend mainly on its propagandistic value, on its ability to contaminate its environment, especially on its potential to influence practice. The theoretician who aspires to do successful science should be a tireless and meticulous expositor and popularizer, and must strive to bring his or her results to the attention of the experimentalist and the practitioner, to convince them of their value[4] (by arguments that are measured, rigorous, and credible, see Section 8 for a discussion of this point). And, of course, doing your own experiments helps a lot.

[3] Assuming that dishonest, anti-intellectual, superstitious, or sloppy "science" is no science at all.

[4] Physics is often mentioned (although not by physicists) as a model of harmonious hand-in-hand collaboration of theory and experiment. Should we aspire to emulate this model, we

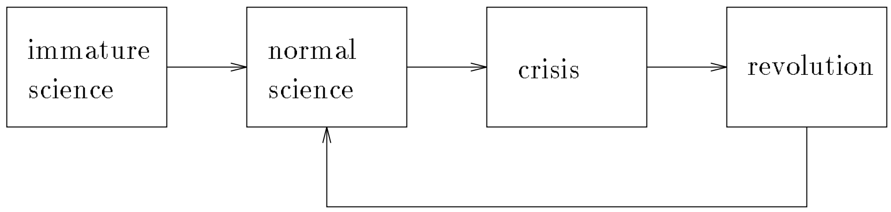

Figure 1: The stages of the scientific process according to Thomas Kuhn.

But the ultimate success of a scientific idea is, of course, the launching of a victorious *scientific revolution;* this is my next subject.

5 On Paradigms and Revolutions

Perhaps the most influential idea in the philosophy of natural science has been that of a *paradigm*, proposed by Thomas Kuhn [Ku]. Kuhn argues that natural science evolves roughly as shown in Figure 1. There are long periods of "normal science," in which the field progresses incrementally within a broadly accepted framework that includes not only scientific assumptions and theories, but also conventions about what are appropriate questions to ask and how further development should proceed. Such a framework is called a *paradigm.* The term was intentionally left by Kuhn with no further definition; Copernicus' model and Einstein's general theory seem to be the most frequently mentioned paradigms. During normal science, scientists consider it their duty to defend the paradigm and show that it works (such behavior would today be perhaps hammered in by such instruments as graduate education and program committees). But cruel facts that do not fit in the paradigm accumulate, despite the community's ingenious efforts to sweep them under the rug; the paradigm creaks and staggers, and we enter a stage of "science in crisis".

During crisis, the accumulated anomalies slow down progress and dispirit the scientific community. The framework of rules that was the prevailing paradigm weakens, and new kinds of ingenuity and imagination develop and compete. Eventually, and typically, one of them triumphs and becomes the next paradigm; this is the stage of "scientific revolution".

---

must not forget a crucial fact: Communication between theoretical and experimental physicists is relatively easy, because *experimental physicists are usually surprisingly good theoreticians.* In contrast, the attitude of most experimental computer scientists towards theory typically ranges from ignorant hostility to naive curiosity (the healthily growing ranks of ex-theoreticians who are doing experimental work —thus propagandizing with their feet— are the main exceptions). I am pointing this out not as an excuse for theoreticians to neglect their educational duty towards practitioners, but as a warning that the task will not be easy.

Kuhn's view was explicitly intended for the natural sciences. To my knowledge, there has been very little discussion of its adaptation to applied science and the sciences of the artificial. It is very tempting to contemplate how the idiosyncrasies of, say, computer science and its subfields would affect the structure and characteristics of Kuhn's stages.[5]

With no intention of belittling the dynamism of natural science, I must now observe that its object is fairly static —or should I say eternal? In contrast, in the sciences of the artificial we study *artifacts*, which keep changing while studied. In fact, in many ways our object changes *exactly because it is being studied* (after all, the ultimate goal of our study is precisely to improve the artifact). We have thus a tight closed-loop interaction between a science and its object. Furthermore, at least in the case of computer science, the artifact interacts with (both deeply affects and is deeply affected by) entities that are themselves fantastically dynamic: the computer industry, the computer market, society. One would expect that the stages of Figure 1 are much accelerated in the case of computer science.

The ever-changing nature of our science's object is of relevance for another, more fundamental reason: Crises in natural science are caused by the accumulation of unsolved problems or "anomalies", observations of the objective reality that cannot fit the current paradigm. Theories in the natural sciences are, necessarily, falsifiable, and it is exactly manifestations of this falsifiability that bring about scientific crises. In contrast, in computer science we have no objective reality against which to judge our scientific work. What aspect of the scientific process brings about crises in our science —in other words, *what is the operational analog of falsifiability in computer science?*

This seems to me a challenging question which I have no ambition of answering in a convincing way.[6] However, I do have a plausible model to propose. Visualize applied science as the interaction of "research units", as in the graphs in Figure 2. These units (whose precise nature and granularity I want to leave unspecified, although you may think of them as researchers, papers, research groups, results, or subfields) are to a varying degree theoretical (from product development to absurdly abstract theory), and influence each other in various ways (conscious, documented, or otherwise), denoted by the edges. The top snapshot in Figure 2

[5]There is a logical gap here, of course, in assuming that these stages exist in computer science; after all, Kuhn's argument was intended for natural science, and, as we shall see, was very specific to that domain. The real and appropriate defense here is that Kuhn's ideas have themselves paradigmatic power, and it is hard to think outside their framework; indeed, most philosophers of science since Kuhn work largely within his framework.

[6]Lakatos [La] briefly discusses the same question for the case of mathematics, arguably also a science of the artificial; my thoughts have been somewhat influenced by his argument. Incidentally, there is disagreement whether the object of mathematics is an ideal reality as existent as universe and life, or an artifact consisting of artificial axioms, definitions, and their consequences (or, even more intriguingly, the result of complex interactions between innate ideas and stimuli from social life and the natural sciences).

depicts a rather healthy situation: As in any decent random graph [ER] there is a *giant component* (in fact, one with reasonably small diameter) that spans most of the practical-theoretical spectrum. Autistic theories and introverted products do exist, but they are the salutary exception that tests the wisdom of the rule. Most of theory is within a few hops from practice, and vice-versa. Theoretical explorations are absorbed by more applied strata and brought back into the mainstream. Incidentally, the value of a modest level of exploratory activity is seen clearly here: It can help fill previously uncharted regions of the space by nodes and, more importantly, edges in all directions.

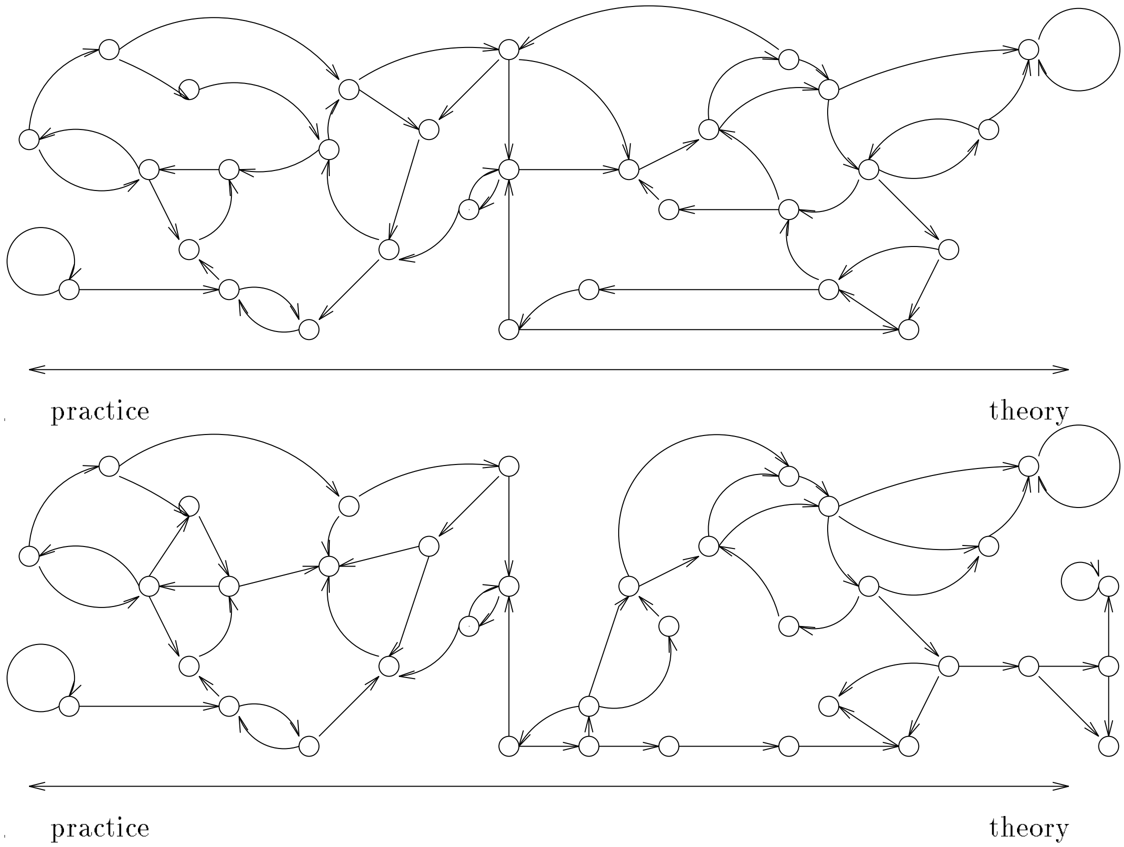

Figure 2: Normal applied science (top), and applied science in crisis.

The snapshot in the bottom of Figure 2 differs from the one on top only in subtle *global* aspects (thus the differences can escape detection for a long time). Although the local situation seems unchanged (say, the average degree is the same as before), connectivity is low. Tangents and introverted components are the rule. The little connectivity that exists is via *long paths*, as theoreticians iterate posing and answering their own questions that bring them further and further from the original motivation [Ul2]. Practitioners of various degrees and shades have stopped listening to theory —even to the parts to which they should listen, presumably because even relevant theory is now done and communicated in a unfriendly, defensive style. Morale is low and interaction unpleasant, with sparks flying at every panel discussion and recruiting committee meeting. The field is in crisis.

But there is hope. Having given up on theory, practitioners develop and use their own abstractions, models, and mathematical techniques, while theoreticians make their own attempts to reconnect to practice (perhaps responding to "applicability pressures" from within their community and outside). The uninspiring practical problems and the unresponsive theoretical work that triggered the crisis become less central, and new small research traditions blossom. Well-targeted exploratory theory connects several of them, and a new healthy state emerges from the ashes. A successfully championed new research paradigm may then take over.

Should all this be familiar to a database researcher? I think so. The introduction of the relational model was as clear a paradigm shift as we can hope to find in computer science.[7] It fulfills the requirements: (1) It was a powerful and attractive proposal (whose plausibility was expertly supported by theoretical arguments such as Codd's Theorem [Co2]); (2) it was explicitly open-ended, a whole framework for research problems, applications, and experiments; (3) it came as the result of a crisis [DBTG] (or was it "immature science", recall Figure 1?); (4) it was indeed followed by a period of normal science. Whether this period has ended as we speak, and we are now in the blues of a crisis, or even in the flames of an on-going revolution, is an important question for each one of us to ponder.[8] A look back at the fourteen years of database theory's most prestigious conference is an interesting and helpful exercise in this regard.

6 A PODS Retrospective

PODS started in 1982, when database theory was already a well-developed field —at least by computer science standards. The relational model was almost a

[7]High-level programming languages were arguably the most important paradigm shift in computer science. I can think of the object-oriented model as another true paradigm in the sense of Kuhn, although the "software crisis" that produced it was not exactly scientific in nature. VLSI and parallel computation, to mention two other contenders, were responses to opportunities, rather than crises (but perhaps this is a valid adaptation of the concept of paradigm to the realities of computer science). Polynomial-time and NP-completeness is another example of a powerful, open-ended research idea with a long wake that sprang out of something like a crisis —the broadly perceived inability of computability and formal language theories to discern between feasible and infeasible computation— and it is sometimes called a paradigm [Pa2].

[8]But remember that it is very difficult to answer such queries on-line. During several famous political and social upheavals, very few of the participants (sometimes only one) had the right answer. For my personal perspective, see the last section.

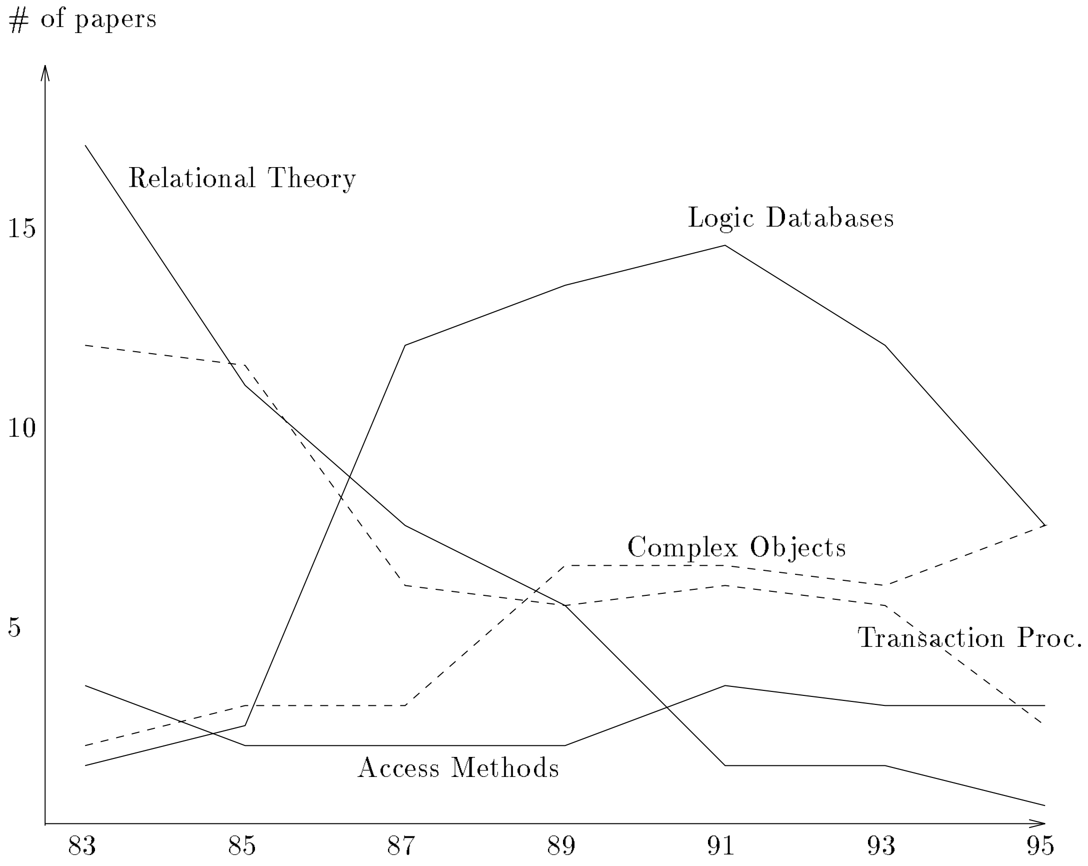

Figure 3: The number of PODS papers in five areas, averages for the two-year period ending in the year indicated.

teenager, the synonymous book by Jeff Ullman [Ul1] had appeared, and two major research traditions were dominant, almost to the exclusion of anything else. The first was *relational theory*, including, among others, the closely interacting subjects of dependencies, normalization, views, query optimization, universal relation assumptions, and acyclicity. The second is what can be called *transaction processing*, encompassing topics such as concurrency control and schedulers, reliability and recovery, distributed concurrency control and systems (including some almost purely PODC material), transaction theory, and concurrency specialized to data structures and to transactions with known semantics. In the first two years there was very little else: Some security (previously a major theme in database research), and timid and scattered representation of two issues that were precursors of explosions to come: incomplete information (basically null values, and then disjunctive databases and closed-world assumptions, which later developed into deductive databases and DATALOG), and non-flat data models (which evolved into the currently important "complex objects" category, including object-oriented, spatial and constraint databases). Data structures and access methods already had the modest presence they would maintain throughout the fourteen years (in this category I am also including sampling and statistical modeling aspects of query optimization).

DATALOG, and its two main issues of query optimization and negation, took the field by storm (possibly because they had been brewing in other communities for some time). In the first conference with a significant presence of this topic (1986) there was a block of *ten* papers, and the number increased to *fourteen* the following year (including an invited talk[9]). This tradition has been by far the largest in terms of volume in PODS, but it now shows definite signs of waning.

Figure 3 is a picture of extreme dynamism.[10] Actually the graphs very much recall solutions to Volterra equations for an isolated ecosystem with very aggressive predators [Sig]. The decline of the prey brings about the decline of the predator, who then becomes the prey of the next species. But of course, our ecosystem was far from isolated: Much of the movement in Figure 3 partly reflects what was happening in computer science and the database practice at that time (or a couple of years earlier). And there is no real predator and prey (although relational theory and logic databases arguably behaved that way). A more accurate analogy would involve species competing for space but depending on different food sources,

[9]PODS invited talks coincide in three distinct instances with the maximum derivative in the volume of the corresponding area. There are many possible explanations, based on different flows of causality; I dare not believe the one that says that people actually *listen* to invited talks.

[10]Notice that the curves represent two-year averages; single-year data would be too jerky to display, mostly because of a strong *two-year harmonic*. For example, the time series for Logic Databases between 1986 and 1992 is $(\ldots, 10, 14, 9, 18, 13, 16, 14, \ldots)$. This bizarre phenomenon is also present in the decline of transaction processing. I have a theory for this: What has a one-year memory in science? Program committees! I think we are seeing here the work of committees trying to correct "excesses" (in one direction or the other) of the previous committee...

that are to a varying degree extensive, renewable, and externally controlled. For example, concurrency control was a problem that was to a large extent solved as satisfactorily as it could be —and this was confirmed by both theoretical exploration and feedback from practice. And the intellectual content of relational theory proved to be *very* large, but still finite.

The declining curves in Figure 3 also bring to mind the following question: Does our community make the mistake of holding for too long on traditions that are past their peak? Should we be more responsive to the winds of change? In my opinion, Figure 3 shows that we are, if anything, too responsive and subject to fashion and fad; if we were any more responsive, the field would lack the diversity necessary for covering the motivating application and attracting researchers.[11]

Figure 3 suggests that database theory has not only high dynamism, but also very high connectivity. But how about its connectivity with other regions of the research graph (Figure 2), such as applied database research and database products? Clearly, the field seems to be responding quite well to changes in both applied research and the artifact, that is to say, there seem to be plenty of incoming arcs. But of course it is outgoing arcs that we are most interested in. Here the situation is mixed. Certainly the relational data model has had tremendous impact on computational practice. Normalization and dependency theory, for all its innumerable tangents, has reached practice in the form of database design tools ([BCN] mentions more than twenty database design tools that do some form of normalization). Concurrency control is of course inevitable, but most database products seem to have adopted the simplest solutions [GR] (two-phase locking, and occasionally optimistic methods or tree-based locking). And the PODS community contributed much to the dissemination of object-oriented database ideas and systems [BDK]. The major disappointment is perhaps the absence of database products that incorporate some of the beautiful ideas our community has developed for the implementation of recursive queries. But there are reports of *prototypes* that are useful to actual applications [Ra1], and of recursive query evaluation methods that were useful for *non-recursive* query optimization [Ra2].[12]

[11]Natural scientists are known to hold on to paradigms even after they have been undeniably falsified; Philip Kitcher [Ki] uses a simple population genetics model to argue that such diversity is beneficial and inevitable.

[12]Note the analogy with arguments for the space program based on the byproducts of its technology that have applications on earth. But I believe that direct impact on actual products is a very cruel criterion. I am wondering how many ideas from SIGMOD and VLDB would make it.

7 A Brief History of Practice

Respect for practice is so universal today,[13] that it is easy to forget what a recent development it is. The ancient Greek tradition[14] strongly favors theory (from $\theta\epsilon\omega\rho\epsilon\iota\nu$, to contemplate) over practice (from $\pi\rho\alpha\tau\tau\epsilon\iota\nu$, to act). It was self-evident to Aristotle that thinking for the sole purpose of achieving knowledge and wisdom is superior to thinking for achieving more worldly advantages such as wealth and power [Ar]. Before the last century, an inventor could become famous only if he[15] was a moonlighting major theoretician or artist (Archimedes, Aristarchus, Leonardo da Vinci) or if his invention helped in the spreading of theoretical knowledge (Gutenberg).[16] Practice starts obtaining a measure of respectability with Galileo (1564-1642) (and later under the influence of the British empiricist philosophers) by outfitting theoreticians in their experiments. However, only after James Watt (1736-1819) did sophisticated theoretical knowledge come to the assistance of practice and invention, thus launching the industrial age and the traditions of applied science and engineering. Theory and practice collaborated gloriously if uneasily for two centuries, with theory dominating important domains in applied science due to its academic prowess and prestige (borrowed from the natural sciences, see the introduction of [Si] for a discussion of this topic). Serious and systematic ideological attack against the value and necessity of theory in applied science seems to be a novel and disturbing phenomenon of the last decade or so.

As for computer science, its history in this respect is a miniature of the history of science. The strongest founding influence for computer science came from mathematics (and less from electrical engineering and physics), and thus its early history was largely dominated by theory and theoreticians. Practice earned respectability in the late 1960s with successes in multitasking operating systems.[17] Certain areas, such as cryptography and databases, thrived on the creative coexistence of theory and practice. And it has recently become fashionable in computer science to criticize theory and belittle its contributions. My last two sections examine certain aspects of this tension.

[13] In the literature on the philosophy of technology, theory is sometimes personified by Plato, and practice by Odysseus (Ulysses) [Fer]. Theoreticians who are deeply suspicious that applied research will pollute, invade, and pillage our platonic city of philosophers, will be terrified to recall that Odysseus was the inventor of, among other things, the Trojan horse.

[14] This is an excuse for the mandatory hellenocentric parenthesis, not a serious historical treatment of this important subject. For a lovely discussion of theory and practice by a man who knows see [Kn].

[15] Embarrassingly, no "she" comes to mind; I will be delighted to be corrected on this one.

[16] An exception is Hero of Alexandria (ca. 50 AD), a remarkable precursor of Watt.

[17] The other epic research program of that age, compilers, was led to a large extent by theoreticians; in fact, theoreticians of Edsger Dijkstra's kind were also prominent in the early stages of the operating systems program.

8 On Applicability

A talk like this is a one-time opportunity to influence the field, give advice, sound an alarm, issue a call to arms. Looking back to this paper, I can see now that I was perhaps a little too "scholarly", even-handed, and measured in pushing my own view. In these last two sections I want to be a little bolder, take sides, even articulate some advice that is, I believe, concrete, modest, realistic, and beneficial.

Applicability pressures are ever present these days, ranging from the mild and friendly to the hideously self-righteous, and they seem to be coming from all directions: experimentalists, fellow theoreticians, media, grant monitors, politicians, deans. My first point about them is perhaps obvious: Applicability pressures do have a place in the scientific debate, even if we believe that in science 'anything goes' —in fact, especially if we do. Remember Feyerabend's other aphorism *"there is no idea that is not capable of improving our knowledge."* Applicability views — as long as they are not dogmatic and oppressive— can be valuable research stimuli and wake-up calls.

And, of course, we should feel free to ignore them at any time. A theoretical paper can be proud in its elegance, beauty, and purely theoretical interest and motivation.[18] *What I consider unacceptable (and regret having done) is to obscure theoretical work with a cloak of applicability claims that are frivolous, far-fetched, and non-rigorous.* Phony applicability claims come in many forms:

Recursive applicability: "The last paper on the subject starts with a claim that the problem is practically important, so it must be." The author probably spent much energy understanding and checking the proofs of that paper. In my view, the applicability claims are worth checking with equal vigor. We should follow up citations and references, and read and understand the relevant applied literature.

Remote applicability: Applications of computer science to other sciences must walk a thin line. It is easy to become the algorithms RA of a physicist or chemist. And it is easy to massage a problem in biology, say, until it succumbs to the tricks of our trade —and is of no use to biologists. It is much harder, but very rewarding and worthwhile, to do work that genuinely contributes to the advancement of another science, while at the same time being proud of its position in computer science [Ka].

Applicability by association. "Circular arc graphs are important in compilers." This could very well be a true statement, but it is irrelevant if we go on to solve an obscure and alien to the compiler application problem (my favorite: minimum cover by circular arc graphs). The argument should be whether our paper's contribution is of practical interest, not whether its keywords lend it an aura of practicality.

[18]And its authors should suffer with pride and courage the consequences: limited appreciation in computer science conferences, departments, and funding agencies. This is no sadistic remark, I consider myself as one who endures some suffering of this sort.

Applicability by pun. The classic style here is the following (made-up example): "Planarity is practically important, parallel computation is practically important, so parallel planarity algorithms got to be *very* important." The author here should have investigated whether the applicability domains of the two concepts intersect, and how extensive and important this intersection is.

I believe that theoreticians should check the validity of their applicability claims as carefully as they check their proofs[19] —and so should reviewers.[20] I think that this will improve the quality of our science. It will make our practically relevant work more focused and better argued, and will let our purely theoretical results shine even more brilliantly, free from any façade of false applicability. I also hope that this exercise in rigor and scholarship will bring theory and theoreticians closer to practice, its literature, and its real problems.

9 Theory in the Time of Crisis

Severe applicability pressure is only one symptom of the present tension within computer science. In searching for the roots of the crisis we must look at three places: The social milieu, the artifact, and the science.

It seems clear that our society is at an important juncture regarding its relationship with human intellect in general.[21] De-intellectualization is the order of the day in many aspects of life; research and academia are logical and strategic targets.[22] Computer science, lacking the serene self-confidence that comes with age and political entrenchment (enjoyed by physics and economics, say) is over-reacting insecurely. And what would be a more natural reaction than to harass its own intellectual vanguard —for being just this?

Secondly, we have always taken pride on how all-pervasive our artifact is; we are now paying the price. The galloping globalization of computation has produced novel and challenging research issues, as well as a globalization and intensification of the debate concerning these issues (ranging in level from the scientific to

[19] Honest speculation of the form "I think that this may be of practical interest because..." is acceptable —although I would much prefer a more thoroughly researched and documented claim; after all, replacing speculation by fact is what science is all about. Naturally, speculation about *future* application opportunities is valuable and welcome. But it should be taken for what it is: a scientific claim open to cruel falsification by life.

[20] Of course, flakey *inapplicability* arguments (such the one familiar from program committees "I showed it to a practitioner and he didn't like it") are even more unacceptable, as they unfairly handicap applicable theory. A result does not have to solve everybody's problems and satisfy everybody's aesthetic criteria to qualify for applicability.

[21] Some would go as far as predicting that humanity as a whole is sliding towards a future that is dark, sinister, and decidedly anti-intellectual.

[22] Actually, the scientific community may have provoked or facilitated this attack by promoting "big science" —the kind that needs public and political support, and therefore necessarily broadens and trivializes the scientific debate.

the popular, with many layers of marginal and shallow science in between). As all scientists feel the centrality of computation in their research program, they voice their opinion about the research agenda of computer science; the result is a cacophonous and off-tempo chorus. Such an environment is not conducive to composed contemplation of the foundations —that is to say, theory.

It is also true that theoretical computer science is coming of age.[23] It now seems that the easy observations have all been made, and the ready-made mathematical techniques have all been applied. The big problems are still with us (and those that have been solved seem smaller from a distance). The basic models have been explored exhaustively, and the new models have not had the experimental attention and practical impact that they deserved. The new technological challenges do not seem to lead to good theory —and the practitioners are not listening anyway. Our research graph (Figure 2) is falling apart.

What are theoreticians to do at times such as these? I believe that the ideas that will deliver us from the crisis (and which most likely will be to a large extent theoretical) will necessarily develop in a reasonable isolation from it. Theoreticians must pay limited attention to the voices of the crisis. We should not feel obliged to coordinate our research goals with current applied research, such as it is. We should also question and challenge the prevailing ideology within theory, be iconoclastic and irreverent to established ideas, trends, traditions, and leaders —after all, they too are voices of the crisis. We should be even more independent, bold, imaginative, exploratory, anarchistic. But we should constantly have in mind the complexity-reducing program of computer science, and the connectivity-increasing function of theory within it; altogether ignoring these issues only feeds the crisis. Incidentally, a crisis is a most opportune time for theoreticians to do their own experiments, and to get involved first-hand in applied research.

I do not think that this is a time for self-pity and despair. *"It is darkest before the dawn."* After all, a crisis is often the precursor of the ultimate scientific experience: a beautiful, brilliant, exciting scientific revolution.[24]

> *"Those whose acquaintance with scientific research is derived chiefly from its practical results, easily develop a completely false notion of the mentality of the men who, surrounded by a skeptical world, have shown the way."*
>
> Albert Einstein

Acknowledgment: I am indebted to many friends for the ideas, feedback, references, and quotations they contributed to this paper: Serge Abiteboul, Costas

[23]The correct analogy here is, one hopes, not mid-life crisis or senility, but teething.

[24]In my papers I use footnotes very sparingly. However, influenced by the topic of this paper (and the literature I had been reading while working on it) I thought I should have at least two dozens.

Callias, Stefano Ceri, Vassilis Christofidis, Jim Gray, Joe Hellerstein, Yannis Ioannidis, Paris Kanellakis, Philip Kitcher, Phokion Kolaitis, Elias Koutsoupias, Paul Kube, Mike Luby, Jeff Ullman, Moshe Vardi, Umesh Vazirani, Victor Vianu, and Mihalis Yannakakis.

References

[Ar] Artistotle *Ethics* bk. X, ch. 7; *Metaphysics* ch. 2.

[As] Astrahan, M. M., *et al.* "System R: A relational approach to data management," *ACM Transactions on Database Systems, 1*, 2, pp. 97–137, 1976.

[BHG] Bernstein, P., V. Hadzilacos, and N. Goodman *Concurrency control and recovery in database systems,* Addison-Wesley, 1987.

[BCN] Batini, Carlo, Stefano Ceri, and Shamkant Navathe *Conceptual database design: an entity-relationship approach* Benjamin Cummings, 1992.

[BDK] Bancilhon, François, Claude Delobel, and Paris C. Kanellakis *Building an object-oriented database system: the story of* O_2, Morgan Kaufman, 1992.

[BPT] Buss, S., C. H. Papadimitriou, J. N. Tsitsiklis "On the predictability of coupled finite automata: an allegory on chaos," *Proc. 31st FOCS Conference*, pp. 788–793, 1990. Also, to appear in *Complex Systems.*

[Co1] Codd, E. F., "A relational model for large shared data banks," *C.ACM, 13*, 6, pp. 377–387, 1970.

[Co2] Codd, E. F., "Relational completeness of data base sublanguages," in *Data Base Systems* (R. Rustin, ed.), pp. 65–98, Prectice-Hall, 1972.

[Cook] Cook, S. A., "The complexity of theorem-proving procedures," *Proc. 3rd FOCS,* pp. 151–158, 1971.

[DBTG] *CODASYL Data Base Task Group April 1971 Report,* ACM, 1971.

[ER] Erdős, P., and A. Rényi "On the evolution of random graphs," *Magyar Tud. Akad. Mat. Kut. Int. Kőzl, 5*, pp. 17–61, 1960.

[Fa] Fagin, R., "Generalized first-order spectra and polynomial-time recognizable sets," in *Complexity of Computation*, R. M. Karp (editor), SIAM-AMS Proceedings vol. 7, pp. 43–73 1974.

[Fer] Ferré, Frederick *Philosophy of technology,* Prentice Hall, 1988.

[Fe] Feyerabend, Paul *Against method,* Verso, 1993 (third edition).

[GR] Gray, J., and A. Reuter *Transaction processing: concepts and techniques,* Morgan Kaufman, 1993.

[Ka] Karp, R. M., "Mapping the genome: Some combinatorial problems arising in molecular biology," *Proc. 25th STOC Conference*, pp. 278–285, 1993.

[Ki] Kitcher, Philip, "The division of cognitive labor (conflicts between individual and collective rationality in science)," *J. of Philosophy, 87*, 1, pp. 1–18, 1990.

[Kn] Knuth, Donald E., "Theory and practice," *EATCS Bulletin, 27*, pp. 14–21, Oct. 1985. See also *Theoretical Computer Science, 90,* 1, pp. 1-15, Nov. 1991.

[Ku] Kuhn, Thomas S, *The structure of scientific revolutions,* The University of Chicago Press, 1962.

[La] Lakatos, Imre, *Mathematics, science, and epistemology,* Cambridge University Press, 1978.

[LB] Lekkerkerker, C. B., and J. Boland "Representation of a finite graph by a set of intervals on the real line," *Fund. Math, 51*, pp. 45–64, 1962.

[Pa1] Papadimitriou, Christos H., *The theory of database concurrency control,* Computer Science Press, 1986.

[Pa2] Papadimitriou, Christos H., *Computational complexity,* Addison-Wesley, 1994.

[PY] Papadimitriou, C. H., and M. Yannakakis "Complexity as bounded rationality, *Proc. 26th STOC Conference,* pp. 726–733, 1994.

[Ra1] Ramakrishnan, Raghu (editor), *Aplications of logic programming,* Kluwer Academic Publishers, 1995.

[Ra2] Ramakrishnan, Raghu, private e-mail communication to Moshe Vardi, September 1994.

[Sig] Sigmund, Karl, *Games of life: explorations in ecology, evolution, and behaviour,* Oxford University Press, 1993.

[Si] Simon, Herbert, *The sciences of the artificial,* MIT Press, 1981 (second edition).

[SWKH] Stonebreaker, M., E. Wong, P. Kreps, and G. Held "The design and implementation of INGRES," *ACM Transactions on Database Systems, 1*, 3, pp. 189–222, 1976.

[Ul1] Ullman, Jeffrey, *Principles of database systems,* Computer Science Press, 1982 (second edition).

[Ul2] Ullman, Jeffrey, "The role of theory today," manuscript, 1994.