


50 & 25 YEARS AGO



EDITOR ERICH NEUHOLD 
University of Vienna
erich.neuhold@univie.ac.at



MAY 1973

<https://www.computer.org/csdl/mags/co/1973/05/index.html>

Artificial Intelligence; Bertram Raphael (p. 9): “Basically, AI is the study of how to make computers more competent. It is a research fringe of computer science, aimed at learning how to write programs or build systems that can perform tasks which have never been performed automatically before; usually, tasks that have been assumed to require human intelligence. ... In the following pages, you will find the recent history and current status of three major subfields: problem solving, perception, and language. Space does not permit exhaustive coverage of these subfields, or of other newer AI interests such as speech understanding systems and complete robot systems.” *[Editor’s note: It is interesting to note that 50 years later, these themes are still subject to intensive research despite the fact that undeniably tremendous progress has been made, largely due to the immense amount of data that are now available and can be processed, for example, in deep learning approaches.]*

Themes in Automatic Problem Solving; R.E. Fikes et al. (p. 11): “A major theme in artificial intelligence research has been the quest for methods for common-sense reasoning about actions and their effects. Such methods, when implemented as computer programs, are called automatic problem solvers.” (p. 12) **“A: Search Trees.** Perhaps the most straightforward and fundamental approach to automatic problem solving centers on the notion of searching among alternative courses of action to find one that achieves the goal. (p. 13) **“B: Theorem Proving.** Some form of logical deduction is an important element of virtually every problem-solving process. ... Unfortunately, the promised benefits of theorem proving have not been fulfilled.” *[Editor’s note: The article then concentrates on two approaches—the general problem solver and Stanford Research Institute Problem Solver—that apply the*

mentioned techniques. Even today, the techniques in this field have not overcome the rapid growth of the search space despite all the increases in processing power.]

Computer Vision; Thomas O. Binford (p. 19): “Teaching a computer to see will open up a host of applications for this ‘intelligent’ machine. But the problem of perception is extraordinarily complex. ... Computer vision can be thought of as the process of transforming this mass of numbers into a symbolic description. The translation process can be subdivided into segmentation, representation, and interpretation.” (p. 22) “Description of simple polyhedral scenes was difficult enough; with these added obstacles, is there hope for success in perceiving the real world? ... Recognition of a few prominent landmarks, combined with a priori knowledge of the environment can thus severely restrict what objects may be present in each part of the scene.” *[Editor’s note: The article recognizes the difficulty of real-world vision and proposes a priori knowledge to help manage the problem. As we know 50 years later, that knowledge is still essential for vision systems. Just think of the difficulty encountered in autonomous vehicles that have to deal with highly unrestricted real-world situations.]*

Research in Natural Language; Terry Winograd (p. 25): “Question answering systems are approaching a level useful in real applications areas. Advanced computing applications like automatic programming will base their interaction with the user on natural language capabilities. ... Traditionally, languages have been described by sets of ‘rules.’” (p. 26) “In studying language, AI researchers have directed attention toward its function. A language is studied, not as a mathematical object following certain regularities, but as a process by which one intelligent being communicates with another. ... If language understanding really involves extensive use of ‘real-world’ knowledge, we are faced with the prospect of having to give the program a ‘universal encyclopedia’ before it can understand. At this point, we have only the most rudimentary ideas about organizing such a mass of knowledge and the reasoning power which must go with it.” *[Editor’s note: The article*

then continues with a detailed description of the language understanding program, using both rules and artificial intelligence that the author developed for his 'block world.' Remember, this article was written 50 years ago, and only lately has the foreseen need for 'big data' processing via deep learning technology really brought the progress the author predicted.]

New Products; Milton G. Bienhoff, Ed. (p. 31): "A number of companies announced new computer models in March; the five reviewed here fall in the range of high-capability mini to low capability G.P. [Editor's note: general purpose]. Four of these new computers offer writeable microprogram memory; one (IBM's) is a system which includes several brand new peripherals as well as a new processor." [Editor's note: Just one example is the HP 2100S Microprogrammable Systems Computer. The system offered three writable control store cards and 16,000 words of memory and cost US\$16,000. Again, it is an example of the tremendous progress the computer field has made in the past 50 years.]

MAY 1998

<https://www.computer.org/csdl/mags/co/1998/05/index.html>

Are ATM, Gigabit Ethernet Ready for Prime Time? David Clark (p. 11): "ATM [Editor's note: asynchronous transfer mode] and Gigabit Ethernet, the subjects of intense speculation by industry observers and potential users of the technologies, appear as if they may be ready to begin competing for their place in the networking sun." (p. 13) "FUTURE: Many industry observers agree that ATM will thrive on the WAN [Editor's note: wide area network] and Gigabit Ethernet will dominate the LAN [Editor's note: local area network]. However, the two technologies will still compete directly in various areas." [Editor's note: The article explains, in some detail, the properties and advantages/disadvantages of the two. It totally ignores the already existing TCP/IP protocols that will, as we know 25 years later, eventually dominate the WAN and LAN world.]

News Briefs; David Clark, Ed. (p. 14): "Auction Sets Stage for New Broadband Wireless Service ... CHIP MAKERS DELAY MOVE TO 300-MM WAFERS ... EUROPEANS AREN'T READY FOR THE EURO ... The Object Management Group (OMG) takes steps to open up JAVA ... PROBLEMS FOR E-MAIL SECURITY STANDARDS ... W3C PUSHES NET ACCESS FOR THE DISABLED ..." [Editor's note: Here, I cite just the headlines of the news briefs to remind the reader what issues were being discussed 25 years ago. You may go to the articles, but just from the titles, you may already feel that some of the issues they involve have not yet been resolved.]

Experimental Models for Validating Technology; Marvin V. Zelkowitz et al. (p. 23): "Experimentation helps determine the effectiveness of proposed theories and methods. But

computer science has not developed a concise taxonomy of methods for demonstrating the validity of new techniques." (p. 25) "These categories and aspects of experimentation apply to science in general, but for effective experimentation in software engineering, we need to implement approaches specific to the characteristics of software. In the remainder of this article, we describe several such approaches and the results of a study examining how these approaches have been used." (p. 29) "To test whether the classification presented here reflects the software engineering community's idea of experimental design and data collection, we examined software engineering publications covering three different years: 1985, 1990, and 1995." [Editor's note: This is a very interesting article that, at first, identifies a large number of possible validation approaches and then uses these cases to evaluate about 550 articles in conference proceedings and journals. To summarize the conclusion, not enough "controlled" validation takes place. Some improvements over time can be seen, but much needs to be done. In my opinion, looking at recent (25 years later) publications, a lot still has to be done.]

Should Computer Scientists Experiment More? Walter F. Tichy (p. 32): "Computer scientists and practitioners defend their lack of experimentation with a wide range of arguments. Some arguments suggest that experimentation is inappropriate, too difficult, useless, and even harmful. This article discusses several such arguments to illustrate the importance of experimentation for computer science." (p. 33) "Experiments are also used where theory and deductive analysis do not reach. Experiments probe the influence of assumptions, eliminate alternative explanations of phenomena, and unearth new phenomena in need of explanation. In this mode, experiments help with induction: deriving theories from observation. Artificial neural networks are a good example of the explorative mode of experimentation. After having been discarded on theoretical grounds, experiments demonstrated properties better than predicted." (p. 40) "Experimentation is central to the scientific process. Only experiments test theories. Only experiments can explore critical factors and bring new phenomena to light so that theories can be formulated and corrected." [Editor's note: This again very interesting article covers essentially the same issues that are raised in the preceding article. It rejects the "lack of experimentation fallacies" with sound arguments and, like the previous article, is really worth reading, as its arguments are still valid 25 years later.]

Cover Feature: Performance Analysis and Its Impact on Design; Pradip Bose et al. (p. 41): "Consider general-purpose microprocessors: Gone are the days when one or two expert architects would use hunches, experience, and rules of thumb to determine a processor's features. ... The main advantage of a systematic process is that it produces a finely tuned design targeted at a particular market. At its core are

models of the processor's performance and its workloads. Developing and verifying these models is the domain now called performance analysis. ... **DEFINING PERFORMANCE:** In this article, our focus is on architectural performance, typically measured in cycles per instruction (CPI)." (p. 45) "After establishing working models of the workloads and proposed processor, it is natural to want to prove that they are correct. ... Calibrating the prehardware models against hardware constitutes the ultimate level of model validation that a design team attempts." [Editor's note: The article then continues to describe various important aspects that have to be considered for performance analysis describing various state-of-the-art approaches. In the following, I only very briefly describe the two articles that utilize these techniques and leave it to the interested reader to go directly to them.]

Performance Simulation of an Alpha Microprocessor; Matt Reilly et al. (pp. 50–57): "We are pleased with the range of experiments our performance model supports. It allowed us to conduct architectural explorations over a large range of processor organizations." [Editor's note: They used quite a number of ideas discussed in the article by Bose et al.]

Calibration of Microprocessor Performance Models; Bryan Black et al. (pp. 59–64): "This article presents experimental results on calibrating a performance model against actual hardware, and based on these results suggests a systematic method for validating performance models." [Editor's note: They use a stepwise refinement process to improve their performance analysis model and describe, in detail, the difficulties encountered.]

BTU: A Host Communication Benchmark; Kurt J. Maly et al. (p. 66): "This fully automated benchmark measures concurrent activities on both the workstation and network to more realistically assess a workstation's communication performance under a particular load. It provides detailed reports to evaluate hardware and software configurations and identify their communication bottlenecks. ... We have developed the BTU (bits to the user) benchmark to take into account both concurrent activities within a workstation (such as CPU and I/O processes that compete for resources) and concurrent activities on the network." [Editor's note: The article introduces six commands around send/receive used in the benchmark suite in a three-workstation configuration (test, active responder, and nonresponder) and then does an extensive evaluation, with five workstations mentioned. It is interesting but limited in scope from today's point of view.]

Advances in Disk Technology: Performance Issues; Spencer W. Ng (p. 75): Although the computer industry has made regular, significant advances in magnetic recording technology for hard disk drives, some advances—such

as those in head design, media, and channel technology—are primarily concerned with increasing disk density and do not necessarily improve total performance. ... We measure disk drive performance by how fast a disk can complete a user request for reading or writing data. ... The time required by a disk drive to execute and complete a user request consists of four major components: command overhead, seek time, rotational latency, and data transfer time." (p. 81) "Since cost is always the number one concern, drive makers will continue to find ways to increase areal density—regardless of its impact on performance. They will also aim to increase speed to beyond 10,000 rpm and to reduce seek time and command overhead." [Editor's note: In this interesting article, the four performance components are investigated extensively, and some issues on improving them are discussed. As we now know, the disk density grew at such a rate that the other parameters could not keep pace, and quite a number of approaches around the redundant array of inexpensive/independent disks arose try to handle at least some of those problems.]

The Power of Process; Steve McConnell (p. 100): "Some people in the software development community think 'process' is a four-letter word. They think software processes are rigid, restrictive, and inefficient." (p. 101) "In a survey of about 50 companies, only 20% of the people in the least process-oriented companies rated their staff morale as 'good' or 'excellent'. ... And, in the most process-sophisticated organizations, 60% of the people rated their morale as 'good' or 'excellent'." [Editor's note: The author's arguments about process-oriented development, of course, have been proved correct over time. However, he did not foresee the innumerable developments of "apps," where processes frequently play a minor role and "power programming" and software libraries carry the time of the day.]

Open Channel: Success? Failure? Or Both? (p. 103): "I collect and periodically publish stories about computing failures. I like to tell people that whatever success I have achieved has been built on a foundation of failure. ... Many people think of it as the A-7 project, while others call it the Software Cost Reduction (SCR) project. ... Things went wrong from the outset. The first task—identifying requirements—proved difficult. ... The SCR team did a magnificent job overcoming the problem. In fact, in so doing they discovered and documented important new ways of recording and organizing software requirements. ... And in the end the SCR team produced an excellent statement of the project requirements. ... The project was canceled because it had burned its schedule and budget, not because it had learned most of what it set out to learn." [Editor's note: As the author states, the project was a success for the gain in methodology, but it was otherwise a failure, as those concepts were never tested in a realistic environment.]

Providing Trusted Components to the Industry; Bertrand Meyer et al. (p. 104): “The recent push for reuse and componentware has raised the hope that by relying on reusable components we can gain quality and reliability. But without excellent techniques to build the components themselves, this nirvana is a mirage. ... With the progress of incompletely designed approaches to reuse, more catastrophes are likely to happen. ... No single technique can produce completely trusted components. Trust is in fact a social phenomenon.”

(p 105) “The Trusted Component Project proposes to apply a mix of formal and informal approaches. It rests on six principal techniques • Design by contract • Formal validation • OO and reuse techniques • Global public scrutiny • Extensive testing • Metrics efforts.” [Editor’s note: A project, *Trusted Object*, is described in this article as a community effort to realize those “trust” principles. Since then, a number of companies have been founded that provide “trust” technology/behavior in the spirit of this article.]



IEEE COMPUTER SOCIETY
Call for Papers

Write for the IEEE Computer Society's authoritative computing publications and conferences.

GET PUBLISHED
www.computer.org/cfp

IEEE COMPUTER SOCIETY

IEEE