MAY 1968
IEEE Computer Group Conference (p. 1) “Large Scale Integration [LSI] and its impact upon the Information Processing Profession is the theme for the 1968 IEEE Computer Group Conference. This single topic will allow the conference to focus entirely on this one area of the semiconductor and computer arts. It will be possible at this conference to explore the field of LSI in much greater detail than would be possible in one of the larger, more general conferences. … [LSI] is four to five years old for many of us, yet we must continue to ask some very fundamental questions: Can we reach the substantial cost reductions projected? Where is the market and how will it evolve? Will implementation problems force radical system designs? Does the key rest with automation, for example, with simulation and testing, or rather will it depend upon our ability to design with repetitive structures?” [Editor’s note: There’s been tremendous progress in LSI in the last 50 years, which very few people in 1968 predicted. Interestingly enough, the successors of this conference are now part of the IEEE Circuits and Systems Society and the IEEE Council on Electronic Design Automation and, therefore, moved away from the Computer Society, so many of us are still involved.]

Directions in Time-Sharing Terminals (p. 12) “One class of CRT terminal that has recently flooded the market is the alphanumeric display. These can be summed as clever ‘electronic teletypewriters.’ … They contain a digital memory that holds the encoded form of the message displayed on the screen; by allowing the user to alter the memory through the keyboard, dynamic editing features can be provided. … These electronic typewriters have not had the impact on time-sharing systems that many expected. For one thing, they are more expensive than teletypewriters. … A second drawback to these units is that they seldom can display more than 20 to 25 lines of text. … The second major category of CRT terminal is the graphic display. Until recently these have been far more expensive than the alphanumeric terminal. The justification for use of these terminals has come from their highly intimate interaction with the human user. … Few who have seen the movie of Ivan Sutherland’s ‘Sketchpad’ program have failed to be impressed with the powerful tool that a graphic CRT display can be. … The application of this type of display will generally be limited to problems whose solutions justify the expense involved to provide this level of interaction. Typical prices for these consoles range from $65,000 to $200,000.” [Editor’s note: Just think what a smartphone now provides, and at what price!]

“Unthinking Man And His Thinking Machines” by Ray Bradbury (p. 19) “Nietzsche said, ‘We have art that we may not perish from truth,’ and I think you called me here today because you are the truth collectors, the data collectors of the world. And on a morning like this morning, passing among your machines, I must admit I came out with my head swimming. I had seen too much; I had tried to gather too much in too short a period of time. Suddenly you find yourself smothered by too many facts; how do you make sense of it all? So you called me here because I am prolific with art, and sometimes short on truth, and you have too much truth and you need my art. … The main fact is the hydrogen bomb, which prevents us from destroying each other. That’s the most beautiful irony and paradox in the history of all the wars we fought; we can’t have them anymore. What is the other fact? The other fact is that in about thirty years we are going to make do about the Cold War. On what day, when what happens? When Mao Tse-tung’s grandson drives his car into Peking and can’t find a damned parking place.” [Editor’s note: Here, a science fiction writer was invited to provide an article. How amazing and true, in a humanistic sense, his thoughts were, especially his vision of Peking, which was accurate even in terms of timescale.]

MAY 1993
www.computer.org/csdl/mags/co/1993/05/index.html

Adaptive Routing Protocols for Hypercube Interconnection Networks (p. 12) “Multipath networks and adaptive routing protocols dynamically adapt to network conditions
such as communication bottlenecks, thus lifting a major impediment to the development of massively parallel architectures. In this article, we propose a taxonomy for characterizing adaptive routing protocols for HINs [hyper-cube interconnection networks]. The taxonomy is based on classes of routing decisions common to any HIN.”

Example-Based Graphical Database Query Languages (p. 25) “In example-based database query languages, users can formulate a query by specifying a graphically aided example on the screen. The authors survey and compare a number of these user-friendly languages. … One of the earliest user-friendly graphical database languages is Zloof’s Query-by-Example, named after its style of forming a query. This approach graphically specifies an example output of the query being formed. … Among the 12 example-based languages discussed, only Query-by-Example and Picquery are commercially available. Query-by-Statistical-Relational-Table, Time-by-Example, and System-for-Business-Automation are in the design cycle. The rest of the languages are in the prototype stage.” [Editor’s note: With the exception of the original Query-by-Example, none of the described approaches made it into commercial use. That is despite the fact that many graphical/pictorial languages for database manipulations are used today.]

Easy-to-Use Object-Oriented Parallel Processing with Mentat (p. 39) “Lack of appropriate abstractions makes programming for parallel architectures more difficult than writing sequential software. Mentat addresses this problem by extending C++ to include parallelism encapsulation. … What distinguishes Mentat from other distributed object-oriented systems is the combination of its objectives: easy-to-use high performance via parallelism and reliance on compiler and runtime techniques to transparently exploit parallelism.”

Efficient Program Tracing (p. 52) “A program trace lists the addresses of instructions executed and data referenced when a program runs. Detailed program traces support many simulations used in computer science and engineering, for example, in the design of processor instruction sets and memory systems, the study of storage reclamation and virtual memory page-replacement algorithms, and the analysis of input to parallelizing compilers. … The difficulties in obtaining a complete program trace stem from the high cost of recording every instruction and data address as the application program executes and from the large size of the resulting trace files. … The tracing overhead can be reduced by modifying either the computer hardware or the application software to record addresses. … The first system, called AE, demonstrated a technique called abstract execution, which greatly reduces the cost of tracing. The second system uses abstract execution along with another technique, optimal control tracing, to further reduce the cost of tracing. These efficient tracing techniques reduce the time overhead to a fraction of a program’s execution cost and the file size by a factor up to 250 times.” [Editor’s note: Tracing is still an important technique in the investigation of program behavior. It should not be confused with logging, which usually concentrates on recording “events” and not instruction executions.]

Open Channel: All at C (p. 112) “I am concerned about an unprofessional outlook toward programming languages that seems to have reached epidemic proportions in the computing profession—both in industry and academia. I believe that this threatens the integrity of the profession. Perhaps I am the one who is perverse, but it seems that a language should be selected on the basis of its suitability to the application and the design paradigm. It seems wrong to select a language merely because it is one’s favorite. … I have been deeply involved in creating a new computer science degree program, and I am convinced that such a program can only impart the skills needed to benefit from academic courses and should never be considered adequate training for professional programming. That requires an apprenticeship—several years working closely with experienced and successful professionals. However, the term ‘apprenticeship’ seems now to be considered dirty, perhaps because it appears to imply spending money. What it actually implies is investing money.” [Editor’s note: What a true statement, and what little effect it had. “Apprenticeship” is still a bad word; “entrepreneurship” is now the right one—just have an idea and do it, without much consideration for design, sustainability, maintainability, security, portability, and so on.]