The case for dynamic efficiency

The shift from vacuum tubes to transistors revolutionised the optimum approach to software design – and banks' risk technology strategy should be formulated with this lesson in mind, argues ***David Rowe***

Everyone knows the story of the silicon-based transistor and the impact it had on digital computer design – enabling engineers to dispense with the large, expensive vacuum tubes that were the first binary logic device and triggering a decades-long process of miniaturisation. What is often forgotten is the impact this revolution had on programming logic.

In the vacuum tube era, binary logic units were a scarce resource – indeed a binding constraint on computer performance – so a lot of effort was devoted to minimising the number of physical logic gates required to solve a given problem.[1] After the advent of the transistor, logic gates became cheap and plentiful, growing ever more so with the march of Moore's Law, which states that the number of transistors in computer chips doubles more or less every two years. While logic gate minimisation continued to play a role in the theory of computer science, its practical importance declined dramatically.

This is an early example of a recurring phenomenon, namely a hardware breakthrough revolutionising optimal system design. Two more recent examples come to mind. In the early to mid-1990s, I was involved in the development of a global trade database at Bank of America. At the time, global network communication was still a comparatively scarce and expensive resource, so the system relied on highly structured files in which the rigid format carried significant implicit information on the meaning of the data they contained. It is no coincidence that self-describing messages using XML-based protocols only emerged in parallel with the massive growth of global communication capacity during the dotcom boom. These protocols are highly flexible and ideally suited for real-time event-driven updates but they are relatively verbose – requiring considerable metadata to clarify the meaning of the numerical values being transmitted.

A still more recent example is the rise of multi-core processors and the deployment of server farm configurations in the middle of the first decade of this century. Until then, virtually all mainstream computing involved sequential calculations on a single core processor. While some simple ways of leveraging multiple processors offered material performance improvements, obtaining the full benefit of this new architecture demanded fundamental rewrites of most applications. Today, risk management applications are responding to two further hardware advances. The first is the dramatic decline in both the absolute and the relative cost of volatile memory. Traditionally, the massive volume of detail produced by a large Monte Carlo simulation could only be stored on mechanical external devices. Physical constraints, such as disc rotation speed, made certain types of program architecture impractical. In many cases, such intermediate data was discarded once its contribution to the ultimate end result was extracted. This made it difficult or impossible to examine the extreme tails of the distribution to isolate what was driving these results. Today, it is far more commercially practical to hold this massive volume of data in active memory, allowing both sophisticated analysis and ad hoc queries of the extreme results to support scenario analysis and stress testing.

Another hardware innovation is the emergence of the graphics processing unit (GPU), which is used to power modern computer games. As a light-hearted quip among gamers puts it, "Why waste good technology on science and medicine?" In fact, of course, it should be the other way around – this hardware is only produced in volume because of the tens of billions of dollars generated by the computer gaming industry. Five years ago, the state of the art in parallel processing was to use a server farm with multiple central processing units (CPUs) on each machine, but the number of arithmetic/logic units (ALUs) per CPU is actually quite small – an important constraint, since it is ALUs that handle arithmetical calculations. A GPU, by contrast, sacrifices flexibility for raw computational power and can provide thousands of ALUs at far less cost than is possible with multiple CPUs. Once again, hardware advances are rendering our existing software configurations obsolete.

The lesson for risk management software applications is that much more attention must be paid to what I call dynamic rather than static efficiency. Dynamic efficiency is the ability to incorporate new technology smoothly and efficiently as it becomes available. In my experience, the institutional difficulty in accomplishing this is often the Achilles heel of in-house risk systems. Overcoming such obstacles will take a combination of disciplined adherence to a flexible architecture and intelligent integration of relevant vendor software. Lacking this, firms will continue to confront the problem so eloquently voiced by that wise cartoon character Pogo, when he said, "We are surrounded by insurmountable opportunities". ■

David Rowe is president of David M Rowe Risk Advisory, a risk management consulting firm. Email: davidmrowe@dmrra.com

[1] *While there appears to be no general solution to the logic gate minimisation problem, various tools such as Veitch diagrams (1952) or a refinement known as Karnaugh maps (1953) can assist in deriving optimal solutions for specific logical requirements (see* http://en.wikipedia.org/wiki/Karnaugh_map*)*