

A White Paper

Author: D Goldsworth

The Limitations of Computers

Computers are excellent at handling large data spaces where the rules are well known, but are unable to handle large data spaces where the rules are either not fully understood or are constantly changing, as is the case in virtually all the more complex business data systems.

For conventional computing to attempt these latter data scenarios it is necessary for them to simplify the problem by 'throwing away' 'irrelevant' data. This 'irrelevant' data can often subsequently become quite significant. A good example of this would be in financial risk management where probability distributions are simplified in order to analyse the vast bulk of the existing risk. However, in extreme market scenarios such as a default of a major country, these discarded data points can become quite relevant indeed. Some banks are able only to compute a risk profile for 20% of each days trading overnight, so they compute only what is already considered high risk, what if there is a trend in the 80%?

This data is thrown away because computers are unable to handle large numbers of variables in a reasonable time regardless of the numbers of computers deployed. This is amply demonstrated in the 'travelling salesman' conundrum. This is where a computer can solve a 10 variable problem in 5 seconds but to solve a 20 variable problem will take the same computer 100,000 years to solve. The conundrum is that the salesman can still find the shortest route around 20 towns in about 5 seconds.

The difference between the approach of the computer and the human is simply that the computer has no initial knowledge and therefore has to compute every possibility. The human however has built up knowledge over time and is simply comparing the problem against that existing knowledge and is able to make an instant and correct decision. The human can no longer do this however when he is presented with too many variables and the following quote from Ian Stewart, Professor of Mathematics at Warwick University shows the result:

"The human brain is wonderful at spotting patterns. It's an ability that is one of the foundation stones of science. When we notice a pattern, we try to pin it down mathematically, and then use the maths to help us understand the world around us. And if we can't spot a pattern, we don't put its absence down to ignorance. Instead we fall back on our favourite alternative. We call it randomness."

The Store and Throw-away Data Society

Regulatory requirements demand that organizations store their data in case it is required at a later date. As a result organisations around the world are producing data faster than there is any meaningful way of analysing it. A study released by researchers at the University of California, Berkeley, estimates that 1.5 billion gigabytes of information are created each year. That's equivalent to every man, woman and child on Earth churning out a novel the size of 'Moby Dick' each week. Just managing this historical data has fuelled the expansion of the computer industry at the expense of existing businesses.

Then when we come to analyse our historical data in order to better understand the future what do we do? Because of the sheer complexity and volume we throw away and ignore the data that we don't consider necessary to analyse. Does this mean that data that was irrelevant 5 minutes ago will never be relevant? We know that is not true.

If our computers are looking for direct matches they can trawl huge amounts of data fast and find the direct match, but as anyone who has ever received two mailings from the same company will know, just one letter in the wrong place can make your address a separate logical entity as far as the computer is concerned.

The programmes that throw away this data have a vast array of impressive sounding names. Let's just take a look at a few of them from a common sense perspective.

Neural Nets, with a name made to sound like it has something of a human characteristic, popular a few years ago and now largely ignored are characterized by needing huge amounts of specific training data to learn, then they 'set' and from then on they can make predictions but unfortunately without *all* the input data and the information from the time of setting. The result is they are about 5% to 10% accurate on prediction and slow. It makes tossing a coin look like a good option.

Genetic Algorithms sound like something that might be derived from evolution. They start out by running some 'experiments' and produce an output that is a spectrum of bad to good results. Then the best results are taken for the next experiments. The problem is that if we are selecting the results we know where we are trying to get to. This has nothing to do with evolution that constantly adapts to change. In reality the overall best result might be derived from the initial worst outcomes.

Fuzzy Logic has the most realistic name and is extremely useful and fast when it comes to balancing just a few variables such as in a car ABS system. Expand out the problem space however and this approach suffers immediately from the computer variable problem and then takes a long time to find any solution.

Simulated Annealing is used widely in a variety of industry applications from Finite Element Analysis to Printed Circuit Board or Microchip layout. It is able to come up with a solution in a reasonable amount of time by seeding a start point in a multidimensional problem landscape and looking for a minimum or deepest valley that represents a

solution. The limitation is in terms of time or energy to find alternative deeper valleys that represent the best or real solution. So once again most of the problem space is ignored.

Bayesian Networks are producing some good results in very controlled environments. Named after the Presbyterian minister and 16th century mathematician Thomas Bayes who posthumously had his paper published in 1763 and who was largely ignored until recent years. He produced a variant of probability theory in that he noticed that not all probability outcomes made sense from a human perspective and some outcomes could be discarded. It suffers with too much information and a cynical but humorous example would be: If a cat is thrown in the air it always lands on its feet, never on its back so we can discard the back outputs. Now when you drop your bread and butter it always lands butter side down so we can discard the butter up outputs. But with more data what happens if the bread and butter is stuck to the cat's back, which way up will it land? As the reference data points are diluted with additional data it loses its accuracy.

All the above software technologies have been produced to try and overcome the limitations of the computer to be able to check large numbers of variables against each other.

Knowing what we know now and being surrounded by limitless amounts of digital stored data would we design a computer today the same way as before or would we do it differently?

The Inspiration for The Uncertainty Engine

So why don't we design a computer that builds up knowledge first like a human and then compare it with real time information? The answer is in computing history, and our inspirations for the 'Uncertainty Engine' are outlined below:

At the end of World War II and 1945 when Winston Churchill ordered the destruction of the code breaking technology at Bletchley Park, he believed that the power of the machines at Bletchley should not fall into the hands of a world not ready to receive it. What came forward to become the modern day computer was purely the calculator, what was left behind was the ability to find and verify patterns in data.

(Luckily the Canadian government was entrusted with the safekeeping of the only surviving circuit diagrams and had not destroyed them as requested. This enabled the Colossus replicas to be rebuilt by employees of Quantel Ltd in the UK, they can now be seen in the Bletchley Park museum and more recently in the London Science Museum).

These reconstructions are still purely the antecedents of the modern computer. However even back then they had identified the need to calculate and remember the patterns in the data first, (the bombs).

A more modern example of this approach is in Multistate Electronics Design Simulation. The output of an electronic logic gate cannot be determined at the outset because its actual state will be the result of delays in the system that will subsequently define its actual input conditions. This can be seen in 'set top boxes' that 'crash' and the only way to reinstate them to a working condition is to turn them off and start again to get back to a known state. As a result in order to 'simulate' this logic on a computer, the system has to keep all the possible output states in memory (and uncertain) until the actual state is determined.

Additional thoughts for performance are based on the principles of closed loop control systems engineering such as used in the world's first 'asynchronous digital framestore' designed by employees of Quantel Ltd in 1974 enabling live TV pictures to be rebuilt and transmitted in real time around the world from the Montreal Olympics. Every evening on their television, people around the world watch pictures being totally rebuilt from one TV standard to another in real time and faster than any computer can achieve even today.

The main and triggering inspiration comes from Organisational Cybernetics and was detailed in the six radio broadcasts given in the autumn of 1973 as part of the thirteenth series of Massey Lectures established by the Canadian Broadcasting Corporation. Stafford Beer describes the cybernetic principles required to design a prototype 'Liberty Machine' that is able to handle all the complex relationships in data. Whilst he was unable to achieve this in his lifetime due to insufficient technology, his concepts appeared absolutely logical. A booklet entitled 'Designing Freedom' is a summary of his notes for the lectures and the catalyst for our technology development.

The Uncertainty Engine Development

Building on the ideas in 'Designing Freedom' and conforming to 'Ashby's Law of Requisite Variety' the Uncertainty Engine has a multidimensional structure, receives input in binary and is able to process 4000 binary elements in each of up to 4000 dimensions.

It reads in the data, learns about the data patterns and remembers the 'pattern rules' in computer memory in a very compact form. The next piece of data to arrive updates *all* the previous knowledge giving the engine its real time performance and highly efficient, high 90's predictive capability even with small amounts of input data. Additional data improves the accuracy of prediction.

In decision making it retains information in an uncertain state until such time as it is certain. When making decisions it looks at all the positive and negative assertions before deciding in much the same way as a human makes decisions. Questions of the data can be posed in an English format such as 'tell me what's interesting'.

This structure of remaining uncertain, finding the patterns then verifying the patterns and finally becoming certain is in stark contrast to conventional computing that starts with a

computer that has to be certain (0 or 1) from the outset and then giving it rules that are already known.

Depending on the industry that the data is coming from, the Uncertainty Engine can make predictions for example about customer behaviour (CRM), fraud, real time Basel II financial risk analysis, enterprise and internet security.

The Uncertainty Engine is currently implemented in a software format that is still limited by the computer host but is in itself a hardware design. When the correct structure is implemented in software then it can be compiled into FPGA's, a microchip hardware design implementation that then unleashes phenomenal performance. Many applications run sufficiently fast in software obviating the need to progress to the hardware phase. It has within its structure a proper deployment of Requisite Variety.

Additional Reading

Designing Freedom by Stafford Beer – This text is a good primer to thinking about the subject. It provides terminology and applies it clearly. The writing discusses Ashby's Law of Requisite Variety and continues with expansion beyond this principle of balanced variety into the realms of attenuation and amplification of variety.

An Introduction to Cybernetics by W Ross Ashby

Cybernetics and Second-Order Cybernetics by Francis Heylighen of the Free University of Brussels and Cliff Joslyn of the Los Alamos National Laboratory

Pdf's of the above publications can be obtained from Sinus Iridum Ltd

About the Author

David was an engineering pioneer in the Digital Broadcast Television market where they developed the world's first asynchronous framestore to allow real-time satellite global television transmission and won the Queen's Award for Technology and Industry on five occasions.

David was part of the team that started the International Operations of Mentor Graphics Inc where they developed and brought to market Multistate Digital Logic Simulation and subsequently changed the way that engineers around the world designed electronics from simulation to full on chip systems design.

David holds a degree in Electrical, Electronics and Control Engineering from the University of Liverpool.