John Woodward: Lateral non-classical thinking

This is an informal positional paper which asks more questions than it answers (if it answers any questions at all!). The aim of this paper is purely to inspire discussion, encourage lateral thinking and brainstorming.

Turing machines, an abstraction of computation, have long been used to investigate what is and is not computable (the best example probably being the insolubility of the halting problem) and what is and is not practically computable (i.e. what can be computed within a polynomial amount of time). It is as amazing fact that many models of computation have been proposed, but they have all been shown to correspond exactly to the class of functions Mathematicians call the recursive functions [1].

While this branch of mathematics has been very successful, it may also be too restrictive a class of model of computation, and fail to capture some interesting Physics and Biology which may improve computation. Indeed as far as AI is concerned some researchers may even go so far as to say Turing Machines maybe irrelevant to AI (reference Aaron Sloman).

Possible physical properties which could be exploited are

- The physical universe is inherently parallel (Evolution could be considered as doing a type of Levin Search).
- Physical systems have a tendency to settle in a state having minimum energy.
- Light takes shortest path between two points.

This list is far from exhaustive.

2 Computing with straight lines and a compass

In this section we give examples of what can and cannot be computed in principle using an idealised mathematical device. A constructible number is one which can be constructed using a line segment of unit length and a compass [2]. It is interesting that some numbers can be readily computed. For example $\sqrt[2]{2}$ can be easily represented by constructing a right angled triangle, two sides of unit length and the hypotenuse will have length $\sqrt[2]{2}$. It has long been known that $\sqrt[2]{2}$ is irrational and therefore cannot be represented on a digital device exactly.

Not only can we construct numbers but we can also do operations on them. For example we can do addition, subtraction, multiplication and division (provided the denominator is not zero). However, there are operation which cannot be performed. For example, it is not always possible to construct a square with the same area as a given circle, and there are certain angles that cannot be divided into 3 equal angles.

3 Operations on numbers with analogy and digital devices

Let us consider the following example of multiplying two numbers. The time taken to multiply two numbers on a Turing Machine will depend on the length in binary (i.e the *precission*) of the two numbers. If the two numbers are precise, the computation will take longer.

Now consider an idealized analogy device. This could consist of numbers being represented by the lengths of lines. Multiplying two numbers could be done by scaling (see [2]) and the result can be measured directly. The advantage here is that the time complexity of multiplying is independent of the two numbers (I am making assumptions of the time it takes to do these operations, and I have not been precise about time complexity when we are talking about analogue machines). Similarly, the process of comparing two numbers illustrates differences between digital and analogy representations.

While analogue methods are open to noise, they maybe more robust when considering rounding errors!

4 Solving the travelling salesman problem in linear time

In this section we present the idea for a process to solve the Travelling Salesman Problem in linear time (linear, not in the number of cities, but linear in the length of the shortest tour). The method is based on using ant colonies.

Suppose we have n cities and the aim is to find a path which visits each city once and only once and has minimum path length. Starting at any city, $n-1$ ants walk to one of each of the other $n-1$ cities. When an ant arrives at its destination, it reproduces, producing a number of children (equal to the number of cities not visited by the parent ant or its direct ancestors). The parent then communicates to the children, which cites have been visited by its ancestors. Each of these children then walk to one of the remaining unvisited cities, the parent ant dies off, and the process is repeated. The first ant back to the starting city will have visited all cities once and only once and will have taken the shortest path and will have a record of the order of the cities it has visited. If there is more than one shortest path, the ants taking these paths will arrive back at the starting city simultaneously.

The immediate objection to this method is that it will require an impractical number of ants as at each city, each ant produces a number of children. (Maybe this is one of the reasons why ants live in large colonies? - so they can do large computations of this type in parallel). Of course we do not literally intend to use ants, but there may be physical properties which may be exploitable (e.g. the wave like nature of light displayed by the double slit experiment i.e. the light wave takes a number of paths simultaneously).

This algorithm is essentially doing a breath first search in the sense that at any point in time, all the ants currently alive have travelled the same accumulated distance (i.e. the distance travelled by it and its ancestors). It maybe possible to implement this type of algorithm on a non deterministic Turing Machine, however no one has built a truly non deterministic Turing Machine. (I believe this is why we put the dividing line between what is practically computable or not between polynomial and exponential complexity classes rather than between other complexity classes, as deterministic devices can simulate each other in polynomial time).

You may be familiar with sorting using the idea of sorting lengths of spaghetti.

While we may encounter problems with this method when the number of quantities being sorted is large (i.e. larger than the palm of one's hand), I feel we should be encouraged by the concept rather than being discouraged by impracticalities at this stage.

5 Conclusions

I envisage the early algorithms relating to non classical computation will be serendipitous discoveries. Instead of asking "how can we write an algorithm to do a given task" we may make more progress by asking "how can this interesting physical property X be used to create new algorithms".

What physical processes exist which we cannot modelled in linear time on a digital computer? I think what will become a classic example (if it has not already), is picking out the longest length of spaghetti in order $O(1)$ time rather than linear time on a digital computer. Some other physical properties are listed earlier. It may be worth cataloguing these physical properties (e.g. parallelism or light taking the shortest path w.r.t. time), so when a new method of doing a given process is discovered, it can be added to the list (this should certainly be a job of a grand challenge).

This will certainly be a time to put our *lateral thinking* caps on!

References

[1] N. J. Cutland. *Computability, An introduction to recursive function theory.* Cambridge University Press, 1997.

[2] John B. Fraleigh. *A First Course in Abstract Algebra.*