Turing Machines are weak - philosophical discussion of computational hardness

Steven Meyer

Tachyon Design Automation, Boston, MA 02128
smeyer@tdl.com

Presented July 26, 2023 at CLMPST 2023 Buenos Aires

Slides will be posted on my web page www.tdl.com/˜smeyer

Introduction

- I argue here that the Cobham-Edmonds thesis that computational problems can be feasibly computed only if they can be computed in polynomial time is problematic because problems in the NP class can be computed in polynomially bounded time on von Neumann style computers we all use called MRAM model.
- I think the earliest paper discussing polynomial hardness by Alan Cobham was presented at the 2nd CMLPS in 1964. Ref: Cobham, A. "The intrinsic computational difficulty of functions." 2nd CLMPS, ed. Y. Bar Hillel, 24-30, 1965.
- Background is the artefact called "digital computer" is thought to be able to compute anything expressible as a Turing Machine (TM) "program". Called the Church-Turing thesis. TMs are universal but a TM program may take unfeasibly long time.
- I am presenting the physicist view of computation complexity.

P=NP

- The Cobham-Edmonds thesis is usually expressed as the P?=NP problem. If the class of problems computable in polynomial time on a TM is not the same as the class of problems that are only computable on a non deterministic TM (NDTM) in polynomial time, then the Cobham-Edmonds thesis is a good measure of problem hardness.
- The class NP is usually expressed as a problem is in the class if a yes/no question guessed solution can be checked on a deterministic TM in polynomial bounded time.
- A possible problem with the class NP is that the question "are two regular expression equivalent" is a "linguistic" yes/no question that is outside NP but maybe is feasibly computable.
- The reason why a difference is possible (TM P?=NP problem is unsolved) is that TMs are weak machines because they require unary tape encoding and lack random access.

MRAM model

- MRAM model has the same properties as modern physical computers. It contains a bounded number of unbounded size random access memory cells. Multiplication and selection are available and all have unit cost.
- In 1974 Hartmanis showed that P=NP in the MRAM model because NDTM can be simulated in deterministic polynomial time using normal computer random access tables. One Ref: Hartmanis, J. and Simon, J. "On the Power of Multiplication in Random Access Machines," Sep. 1974.
- The Hartmanis conceptualization of NDTMs assisted the proof. TMs are deterministic if there is only one copy of a branch to label. For a NDTM model, one or more labels will have multiple copies. This allows simulating NDTM MRAMs on a deterministic TM MRAM by constructing suitable look up tables (building a data base).
- This is similar to modern programming language switch statements implemented as indexed tables.

Neumann computer design

- I am arguing that Neumann studied abstract models and specifically rejected the TM model and chose the MRAM model.
- Neumann thinking in developing modern computers became available only in the 1990s partially because much of his work had been classified.
- Two books explain Neumann's thinking and quote his writing. Aspray, W. "John Von Neumann and The Origins of Modern Computing." 1990. Also: Redei, M. and Stoeltzner, M. (eds.) "John von Neumann and the Foundations of Quantum Physics." Vienna Circle Institute Yearbook 8, Kluwer, 2001.

Neumann thinking

- Applied mathematician Neumann had abandoned Quantum logic and Hilbert's logicism programme when he began designing a computer.
- Wolfgang Pauli told Neumann. "If a mathematical proof is what matters in physics, you would be a great physicist" (Redei[2001], p. 5).
- Neumann was automating the Manhattan Project physicist (Feynman?) organized calculator operator team plus punched card sorting machine method.
- Neumann argued that program complexity needed to be increased. Computers were needed for calculating complex partial differential equation models. Simple differential equation models were inadequate for the physics.

Neumann desired computer properties

Neumann made a list of desired computer properties as part of his work designing the Eniac (Aspray[1990]).

1. A computer would have a finite number of randomly accessed binary encoded memory locations.
2. Computers need to be built large enough for the given problem.
3. Instructions and data were mixed with indirect addressing, selects and indexing.

Karp/Cook problem Reducibility to 3-SAT

- Back to discussion of Cobham-Edmonds thesis. The thesis became plausible because of Karp/Cook reducibility.
- In 1971 Stephen Cook showed that the problem called 3-SAT that goes back to Tarski definition of truth is in the class NP, i.e. it is requires a NDTM to be computed in polynomial bounded time.
- A formula is T (satisfied) if there exists an assignment of true or false to each variable that makes the formula true.
- Richard Karp then showed that many (all?) interesting graph theory and combinatorics problem can be reduced to 3-SAT in TM polynomial bounded time. (ref. Karp, R. "Reducibility among combinatorial problems," 1972).
- This was viewed as either positive evidence for the Cobham-Edmonds thesis because it explained why no polynomial algorithms were being found or as negative evidence because it implies there is only one combinatorial problem.

Knuth's approach - count number of steps

- Problem with Knuth's concrete complexity is that a given program may not be efficient.
- But even worse, how to count is sometimes unclear. The graph pre dominator problem shows the lack of counting clarity.
- Pre dominator is polynomially bounded so Cobham-Edmonds thesis does not have the concrete complexity number of operations counting problem.
- Claimed fastest algorithm is the Lengauer Tarjan tree building algorithm (Ref. Lengauer, T and Tarjan R. "A Fast Algorithm for Finding Dominators in a flowgraph," 1979).
- I think the Cooper-Harvey algorithm (Ref. Cooper, K. et. al. "A simple, fast dominance algorithm," 2001) is not just faster in practice but faster using concrete complexity counting.
- Problem is that the Cooper algorithm is heaped base. A heap is both a priority tree stored in array cells (leaf index is 2x parent index) and a randomly accessible array so there may be

MRAM unbounded RAM cell problem

- TMs seemingly are better models of computation because machine cells are finite (0 or 1 only) size while MRAM cells are unbounded in size and cell sizes can often grow to exponential size.
- The unboundedness infinity is also present in TMs because the number of tape cells is unbounded and can often grows to exponential size.
- People who write programs to solve application problems follow Neumann's advice. Namely, build a wider computer with more RAM so that a given application fits on the computer that runs a program.

Miscellaneous comment on Neumann

- Neumann literature is somewhat unclear.
- I think Neumann terminology needs to be replaced by different modern meaning shifted words:.Automaton should be read as 'computer', self-reproducing Automata meant programs generating other programs (Godel numbering and TM simulation), 'axiomatics' should be read as algorithm. I am not sure of this but it is consistent with Neumann as an applied mathematician.
- Neumann may have rejected the TM model because he was competing with Alan Turing.

Neumann attitude toward computing

- Neumann was skeptical of the theory that the brain was a simple computer (automaton), but as an applied mathematician he worked on the brain as computer analogy. He wrote this:
- **The insight that a formal neuron network can do anything which you can describe in words is a very important insight and simplifies matters enormously at low complication levels. It is by no means certain that it is a simplification on high complication levels. It is perfectly possible that on high complication levels the value of the theorem is in the reverse direction, namely, that you can express logics in terms of these efforts and the converse may not be true. (quoted in Aspray[1990], note 94, p. 32)**

Quantum algorithms

- It is not clear that quantum computers (QC) parallelism is faster than polynomial unit time operation with random access.
- Shor in his algorithm uses P!=NP. It is not clear if this effects run time.
- Maybe the QC Fourier transform relaxation is really just analog computing.
- Literature is filled with improvements to MRAM speeds compared to QCs.
- Maybe physical artefact Neumann computer properties should be the new logic.
- Maybe quantum mechanics and modern logic needs a third unknown (X) value. I am thinking of Hans Reichenbach three value quantum theory.

Conclusions

- Neumann's physical computer design and Hartmanis P=NP proof for MRAMs implies that the Cobham-Edmonds thesis is useless.
- The Cobham-Edmonds thesis implies that probability (guessing) improves algorithm efficiency. The MRAM model shows that guessing is not needed because on actual physical computers table look up is better.
- This is maybe a way of explaining the term "big data".
- Neumann was a strong advocate of Monte Carlo style high level simulation using randomly changed actual program parameters.
- Complexity seems to be similar to the concept infinity. It is difficult to say much about it except in the specific problem specific case. Approximate solutions need to be replaced by application specific efficient algorithms.