Theoretical Computer Science

Jittat Fakcharoenphol

Kasetsart University

December 20, 2025

A little bit about history

- When did research in computer science start?

A little bit about history

- When did research in computer science start?
  - Long before the first working computer was constructed.
- Theorists worked on “research in computation”.

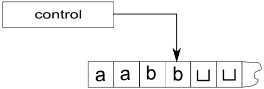
Theory $\Rightarrow$ Practice

Turing machine

[1] http://en.wikipedia.org/wiki/Image:Alan_Turing_Memorial_Closer.jpg

Turing machine

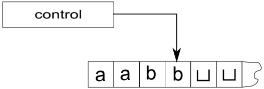
- In 1931, defined Turing machine, a really simple machine that works on an infinite tape.



[1] http://en.wikipedia.org/wiki/Image:Alan_Turing_Memorial_Closer.jpg

Turing machine

- In 1931, defined Turing machine, a really simple machine that works on an infinite tape.

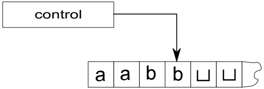


- Turing also proved that

[1]http://en.wikipedia.org/wiki/Image:Alan_Turing_Memorial_Closer.jpg

Turing machine

- In 1931, defined Turing machine, a really simple machine that works on an infinite tape.



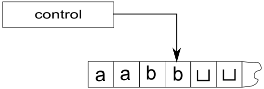
- Turing also proved that
  (a) one can construct a Turing machine that can performs any computation performed by any Turing machine, and

[1] http://en.wikipedia.org/wiki/Image:Alan_Turing_Memorial_Closer.jpg

Turing machine

[1]

- In 1931, defined Turing machine, a really simple machine that works on an infinite tape.



- Turing also proved that
  (a) one can construct a Turing machine that can performs any computation performed by any Turing machine, and
  (b) there exists some problem that cannot be solved by Turing machines.

[1] http://en.wikipedia.org/wiki/Image:Alan_Turing_Memorial_Closer.jpg

Universal Turing machines

Turing machines shifted the focus from building machines to perform specific tasks to building general-purpose machines.

Universal Turing machines

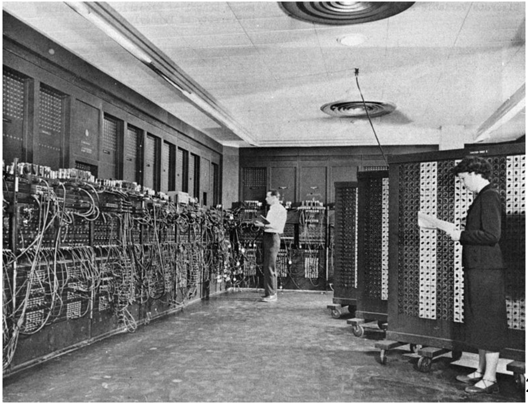
Turing machines shifted the focus from building machines to perform specific tasks to building general-purpose machines.

Historical developments:

- 1945: Von Neumann's *First Draft of a Report on the EDVAC* proposed the idea of a stored-program computer.
- 1940s: First electronic computers (ENIAC, EDVAC)
- 1960s: High-level programming languages (FORTRAN, COBOL)

Practice $\Rightarrow$ Theory

Working programmable digital computers

[2]

- Designed by John Mauchly and J. Presper Eckert from University of Pennsylvania. (Completed 1946)

[2] http://en.wikipedia.org/wiki/Image:Eniac.jpg

Reliability

- Some people predicted that the ENIAC won't run.

Reliability

- Some people predicted that the ENIAC won't run.
- It had 17,468 vacuum tubes. The tube failures would stop the machine so frequently, too frequently.

Reliability

- Some people predicted that the ENIAC won't run.
- It had 17,468 vacuum tubes. The tube failures would stop the machine so frequently, too frequently.
- However, failures usually occur when turning the machine on and turning it off. So the engineers solved this reliability problem by never turning the machine off.

Dealing with errors

- von Neumann studied this problem and proposed a solution based on majority gates in 1956.

PROBABILISTIC LOGICS AND THE SYNTHESIS OF RELIABLE ORGANISMS FROM UNRELIABLE COMPONENTS

J. von Neumann

1. INTRODUCTION

The paper that follows is based on notes taken by Dr. R. S. Pierce on five lectures given by the author at the California Institute of Technology in January 1952. They have been revised by the author but they reflect, apart from minor changes, the lectures as they were delivered.

The subject-matter, as the title suggests, is the role of error in logics, or in the physical implementation of logics — in automata-synthesis. Error is viewed, therefore, not as an extraneous and misdirected or misdirecting accident, but as an essential part of the process under consideration — its importance in the synthesis of automata being fully comparable to that of the factor which is normally considered, the intended and correct logical structure.

Our present treatment of error is unsatisfactory and ad hoc. It is the author's conviction, voiced over many years, that error should be treated by thermodynamical methods, and be the subject of a thermodynamical theory, as information has been, by the work of L. Szilard and C. E. Shannon [Cf. 5.2]. The present treatment falls far short of achieving this, but it assembles, it is hoped, some of the building materials, which will have to enter into the final structure.

The author wants to express his thanks to K. A. Brueckner and M. Gell-Mann, then at the University of Illinois, to whose discussions in 1951 he owes some important stimuli on this subject; to Dr. R. S. Pierce at the California Institute of Technology, on whose excellent notes this exposition is based; and to the California Institute of Technology, whose invitation to deliver these lectures combined with the very warm reception by the audience, caused him to write this paper in its present form, and whose cooperation in connection with the present publication is much appreciated.

$\epsilon + 3\eta^2 < \eta$ is possible. Hence, the chance of keeping the error under control lies in maintaining the conditions of the special case throughout the construction. We will now exhibit a method which achieves this.

8.3 Synthesis of Automata

8.3.1 THE HEURISTIC ARGUMENT. The basic idea in this procedure is very simple. Instead of running the incoming data into a single machine, the same information is simultaneously fed into a number of identical machines, and the result that comes out of a majority of these machines is assumed to be true. It must be shown that this technique can really be used to control error.

Denote by O the given network (assume two outputs in the specific instance picture in Figure 26). Construct O in triplicate, labeling the copies O^1, O^2, O^3 respectively. Consider the system shown in Figure 26.

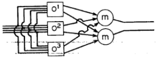


FIGURE 26

For each of the final majority organs the conditions of the special case considered above obtain. Consequently, if η is an upper bound for the probability of error at any output of the original network O, then

(9) $$\eta^* = \epsilon + (1 - 2\epsilon)(3\eta^2 - 2\eta^3) = f_\epsilon(\eta)$$

is an upper bound for the probability of error at any output of the new network O*. The graph is the curve $\eta^* = f_\epsilon(\eta)$, shown in Figure 27.

Consider the intersections of the curve with the diagonal $\eta^* = \eta$: First, $\eta = 1/2$ is at any rate such an intersection. Dividing $\eta - f_\epsilon(\eta)$ by $\eta - 1/2$ gives $2((1 - 2\epsilon)\eta^2 - (1 - 2\epsilon)\eta + \epsilon)$, hence the other inter-

von Neumann's analysis (1)

Suppose that each component fails with probability at most η. Let's make three copies of each component and use majority gates to decide the output of each component.

von Neumann's analysis (1)

Suppose that each component fails with probability at most η. Let's make three copies of each component and use majority gates to decide the output of each component. However, the majority gate itself may fail. Suppose that the majority gate fails with probability at most ϵ.

von Neumann's analysis (1)

Suppose that each component fails with probability at most η. Let's make three copies of each component and use majority gates to decide the output of each component. However, the majority gate itself may fail. Suppose that the majority gate fails with probability at most ϵ.
Therefore the probability that the output is wrong is at most

$$\epsilon + 3\eta.$$

von Neumann's analysis (2)

With more assumptions, we can do better. We assume that errors are independent. There are two cases. When the input to the majority gate is "good", i.e., at most one of the three inputs is wrong. This happens with probability at most

$$E = 3\eta^2(1-\eta) + \eta^3 = 3\eta^2 - 2\eta^3.$$

However, the majority gate itself may fail. Suppose that the majority gate fails with probability at most ϵ. Then, the output is correct with probability at least

$$(1-\epsilon)(1-E).$$

von Neumann's analysis (3)

On the other hand, if the input to the majority gate is "bad"; with probability ϵ, the output is (accidentally) correct; this happens with probability[3]

$$\epsilon \cdot E.$$

Let $f_\epsilon(\eta)$ be the probability that the output is wrong. Then,

$$f_\epsilon(\eta) = \epsilon + (1 - 2\epsilon)E = \epsilon + (1 - 2\epsilon)(3\eta^2 - 2\eta^3).$$

[3] This is wrong, as η is only the upperbound, but let's ignore this fact for now.

von Neumann's analysis (4)

When we perform this construction repeatedly, we want to control the error probability to always be below η.

von Neumann's analysis (4)

When we perform this construction repeatedly, we want to control the error probability to always be below η.
This leads to the following inequality:

$$\epsilon + 3f_\epsilon(\eta) \leq \eta,$$

or

$$4\epsilon + 3(1 - 2\epsilon)(3\eta^2 - 2\eta^3) \leq \eta.$$

von Neumann's analysis (4)

When we perform this construction repeatedly, we want to control the error probability to always be below η.
This leads to the following inequality:

$$\epsilon + 3f_\epsilon(\eta) \leq \eta,$$

or

$$4\epsilon + 3(1 - 2\epsilon)(3\eta^2 - 2\eta^3) \leq \eta.$$

Solving this leads to the condition $\epsilon < 0.0073$.

von Neumann's analysis (4)

When we perform this construction repeatedly, we want to control the error probability to always be below η.
This leads to the following inequality:

$$\epsilon + 3f_\epsilon(\eta) \leq \eta,$$

or

$$4\epsilon + 3(1 - 2\epsilon)(3\eta^2 - 2\eta^3) \leq \eta.$$

Solving this leads to the condition $\epsilon < 0.0073$. Also, if we want error probability η to be below 2%, we need $\epsilon < 0.0041$.

von Neumann's analysis (4)

When we perform this construction repeatedly, we want to control the error probability to always be below η.
This leads to the following inequality:

$$\epsilon + 3f_\epsilon(\eta) \leq \eta,$$

or

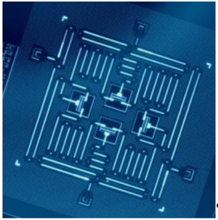
$$4\epsilon + 3(1 - 2\epsilon)(3\eta^2 - 2\eta^3) \leq \eta.$$

Solving this leads to the condition $\epsilon < 0.0073$. Also, if we want error probability η to be below 2%, we need $\epsilon < 0.0041$.
Remark: This construction is not practical, as the circuit size grows exponentially.

Practice $\Rightarrow$ Theory

Practice $\Rightarrow$ Theory
$\Rightarrow$ Future development

Quantum computation

[a]

[a] Gambetta, Chow, and Steffen, npj quantum information, 2017

- Reliability becomes an issue again when people try to build quantum computers.
- **The quantum threshold theorem:**
  A quantum circuit on n qubits and containing $p(n)$ gates may be simulated with probability of error at most ϵ using

$$O(\log^c(p(n)/\epsilon) \cdot p(n))$$

gates (for some constant c) on hardware whose components fail with probability at most p, provided p is below some constant threshold,

$$p < p_{th},$$

and given reasonable assumptions about the noise in the underlying hardware.

Backbone of theoretical computer science

- Upperbounds (prove that something is possible):

Backbone of theoretical computer science

- Upperbounds (prove that something is possible):
  - Algorithms (constructive upperbounds)
  - Data structures
  - Complexity classes (e.g., P, NP, BPP, BQP, etc.)

Backbone of theoretical computer science

- Upperbounds (prove that something is possible):
  - Algorithms (constructive upperbounds)
  - Data structures
  - Complexity classes (e.g., P, NP, BPP, BQP, etc.)
- Lowerbounds (prove that something is impossible):

Backbone of theoretical computer science

- Upperbounds (prove that something is possible):
  - Algorithms (constructive upperbounds)
  - Data structures
  - Complexity classes (e.g., P, NP, BPP, BQP, etc.)
- Lowerbounds (prove that something is impossible):
  - Hardness results (e.g., NP-completeness, etc.)
  - Lowerbounds for specific models of computation

But there are many other areas that study different aspects of computation, e.g., logics, information theory and cryptography, or studies of other computing models.