A non-Turing model of computation

Nitin Saxena (CSE@IIT Kanpur, India)

(*Thanks to the artists for their images.)

2019, IIT Delhi
Contents

- Church & Turing
- Valiant: Algebraic circuits
- Warmup: Determinant
- Valiant's question
- Zero or nonzero
- Applications
What's computing?

- The first response was by Alonzo Church (1935-6). Using effective computability based on his λ-calculus.
- Soon, Alan Turing (1936) postulated a simple, most general, mathematical model for computing – Turing machine.

Church (1903-1995)

Turing (1912-1954)

100th bday!
Church & Turing

- **Computation** is: whatever can be simulated on a Turing machine (TM).

- TM invented to resolve Hilbert (1928)'s question--

  “design an algorithm to decide whether a given statement is provable from the axioms using logic”.

- The answer first requires defining 'algorithm'.
  - hence, 'computation' requires a new mathematical framework.

- Algorithm is very much like a program.
  - TM is like a *real computer*-- highly iterative.
Church & Turing

- TM defines **time & space complexity** of a problem.
  - Asymptotics: as a function of input size \( n \).

- **OPEN Qn:** Is there a problem that **requires** \( 2^n \) time on a TM?
  - Consider SATisfiability problem. Boolean formulas.
  - You get the famous \( P \neq NP \) question.

- It's not clear how to prove the impossibility of a **faster** algorithm.
  - Since math proofs need an **algebraic** or **geometric** structure.

- So, let us look at an algebraic analogue---
Contents

- Church & Turing
- **Valiant: Algebraic circuits**
- Warmup: Determinant
- Valiant's question
- Zero or nonzero
- Applications
Valiant: Algebraic circuits

- An arithmetic circuit $\Phi$ has gates $\{+,*\}$, variables $\{x_1, \ldots, x_n\}$ and constants from some field $F$.
  - Defines size, depth, fanin, fanout.

- An arithmetic circuit is an algebraically neat model to capture real computation.
  - It's complete for polynomials!

- Valiant (1977) formalized computation & resources using algebraic circuits.
  - Giving birth to his $VP \neq VNP$ question!

Leslie Valiant (1949-)
Valiant: Algebraic circuits

- In an $n$-variate $n$-degree polynomial $f(x)$, there are at most $\binom{n+n}{n} \approx 2^n$ monomials.
  - “Usually”, that's the circuit-size complexity of $f(x)$ as well.
  - What's the usual circuit-depth of $f(x)$?

- OPEN Qn: Is there an explicit polynomial $f(x)$ that requires $2^n$ size algebraic circuits?
  - Valiant's VP $\neq$ VNP question even pin-points this $f(x)$.

- To better understand this, let's start with a familiar polynomial--- Determinant.
Contents

- Church & Turing
- Valiant: Algebraic circuits
- Warmup: Determinant
- Valiant's question
- Zero or nonzero
- Applications
Warmup: Determinant

- **Determinant** is an $n=m^2$-variate $m$-degree polynomial defined as, eg. for $m=3$,

$$\text{Det} \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} = x_{11}x_{22}x_{33} - x_{12}x_{21}x_{33} - x_{13}x_{22}x_{31} - x_{11}x_{23}x_{32} + x_{12}x_{23}x_{31} + x_{13}x_{21}x_{32}.$$ 

- It has $m! \approx m^{m/2} \approx n^{\sqrt{n}/4}$ monomials.

- **Qn:** What is its circuit-size complexity?
  - Is it as large as the number of monomials??
Warmup: Determinant

Theorem [Csanky 1976]: $\text{Det}_m$ has a circuit of size $< m^7$.

How's this possible?
- Algebraic ideas/ identities!

**Idea 1:** $\text{Det}_m(X) =$ product of the $m$ eigenvalues of $X =: \prod \lambda_i$.

**Idea 2:** $\prod \lambda_i$ is an expression in $p_k := \sum_{1 \leq i \leq m} \lambda_i^k$ ($1 \leq k \leq m$).
- Newton's identity.

**Idea 3:** In turn, $p_k = \text{tr}(X^k)$.

Thus, $\text{Det}(X)$ computation reduces to matrix-powering $X^k$.
- Easy to implement in a circuit.
Contents

- Church & Turing
- Valiant: Algebraic circuits
- Warmup: Determinant
- Valiant's question
- Zero or nonzero
- Applications
Valiant's permanent

- **Valiant (1977)** posed a related polynomial—*Permanent*.

\[
\text{Per} \begin{bmatrix}
 x_{11} & x_{12} & x_{13} \\
 x_{21} & x_{22} & x_{23} \\
 x_{31} & x_{32} & x_{33}
\end{bmatrix} = x_{11} x_{22} x_{33} + x_{12} x_{21} x_{33} + x_{13} x_{22} x_{31} +
\]

\[
x_{11} x_{23} x_{32} + x_{12} x_{23} x_{31} + x_{13} x_{21} x_{32}.
\]

**OPEN Qn:** Given a matrix \( X \) compute \( \text{Per}_m(X) \)?

- It counts the *number of satisfying assignments* of a given boolean formula.
  - Or, *number of matchings* in a graph.

- Solving this would solve all our previous *NP-hard* problems.

- **Valiant's \( VP \neq VNP \) question:**
  \( \text{Per}_m \) requires \( 2^m \) size algebraic circuits?
Valiant's permanent

- How could we show that permanent affords no ultra-clever algebraic identities that help in circuit computation?

- Classical algebra is not developed enough to answer this question.
  - Permanent, circuits are both recent constructs.
  - A specialized theory is missing.

- We need to study circuit identities---
Contents

- Church & Turing
- Valiant: Algebraic circuits
- Warmup: Determinant
- Valiant's question
- Zero or nonzero
- Applications
Zero or nonzero

- **Question:** Test whether a given circuit is zero.
  - Polynomial identity testing (PIT).

- PIT has a simple randomized fast algorithm:
  - Evaluate $\Phi(p)$ for a random point $p$.

- **OPEN Qn:** Is PIT in deterministic polynomial time?

- Work in last decades show:
  - **Meta-Theorem:** A solution of identity testing would answer the permanent question.

  **Idea:** Hitting sets.
Contents

- Church & Turing
- Valiant: Algebraic circuits
- Warmup: Determinant
- Valiant's question
- Zero or nonzero

Applications
Applications-- Graph theory

- Is there a \textit{perfect matching} in a given graph?
- First solution by \textit{Dinitz (1970), Edmonds, Karp (1972)}.
  - Deterministic polynomial time using \textit{flows}.
- Is there a fast \textit{parallel algorithm}?
- Relates to \textit{PIT}.
  - Write it as a determinant, circuit, randomized \textit{PIT}....
Applications-- Number theory

- Circuits can compute high-degree polynomials.
  - Eg. $f(x) = x^{2^s}$ requires only circuit-size $s$.
  - Repeated squaring.

- Testing primality of $n$ reduces to testing---
  \[(x+1)^n = x^n + 1 \mod n.\]

  - First deterministic polynomial time primality test.

- Relates to derandomizing PIT.
  - Fix $x$ to special values!
Applications-- Learning theory

- Areas like *Artificial Intelligence / Machine Learning* model decision-making using circuits.
  - Artificial Neural Networks (ANN).

- ANN is a *specialized* algebraic circuit.

- *Backpropagation* methods are used to *modify the edges*, and their weights, to improve the output.
  - General *optimization* algos used; “deep learning”.....

- Relates to better understanding of algebraic circuits?
  - Circuit complexity of learning problems...

Thank you!