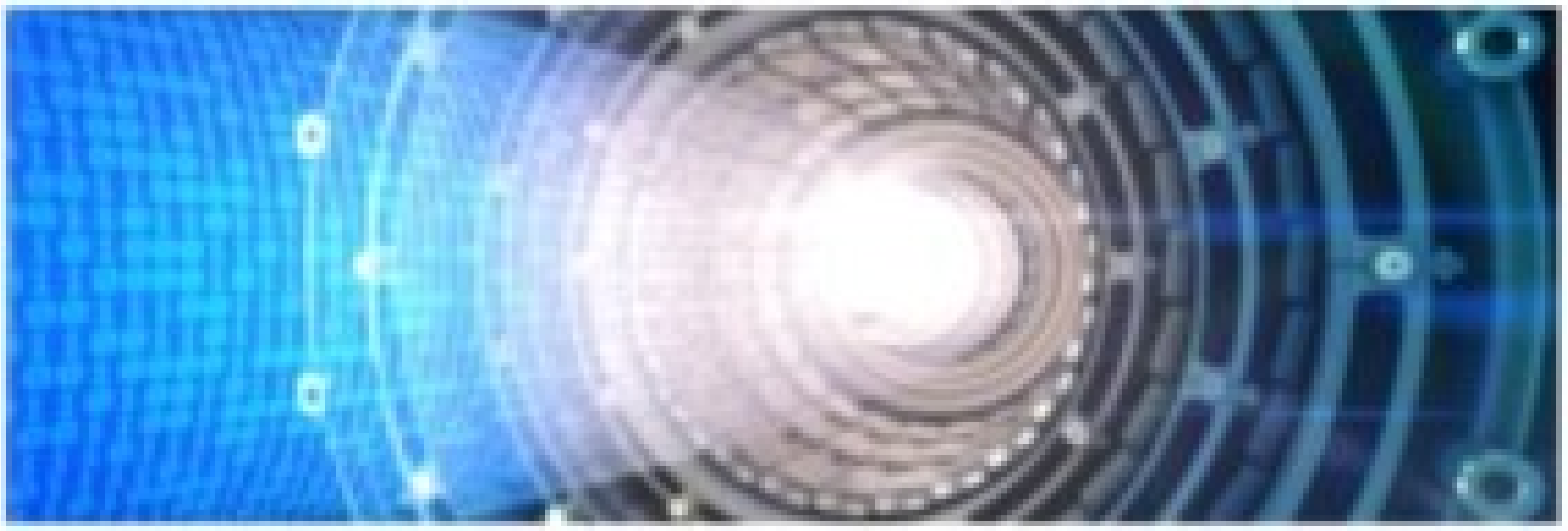
From Hilbert's Entscheidungsproblem to Valiant's counting problem

Nitin Saxena (Indian Institute of Technology Kanpur)



2014

Contents

- Hilbert
- Church & Turing
- Cook & Levin
- Valiant's permanent
- Zero or nonzero
- Fundamental goals

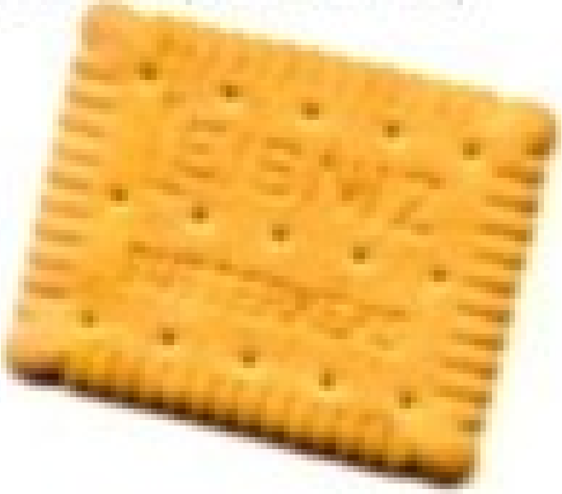
Hilbert

- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.

Hilbert

- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.

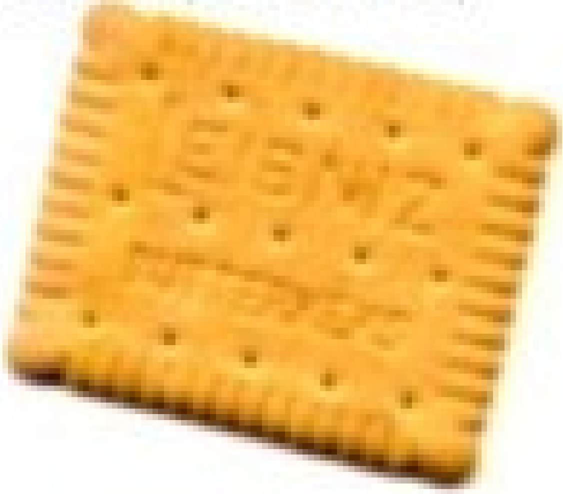
Leibniz (1646-1716)



Hilbert

- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.
- He was the first to design a machine that could do all the *four* arithmetic operations.

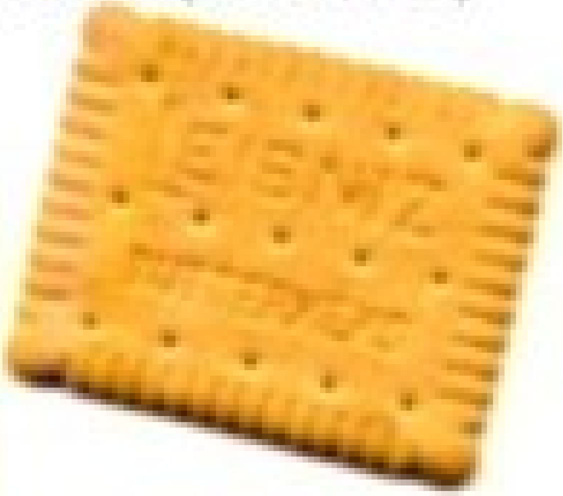
Leibniz (1646-1716)



Hilbert

- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.
- He was the first to design a machine that could do all the *four* arithmetic operations.

Leibniz (1646-1716)

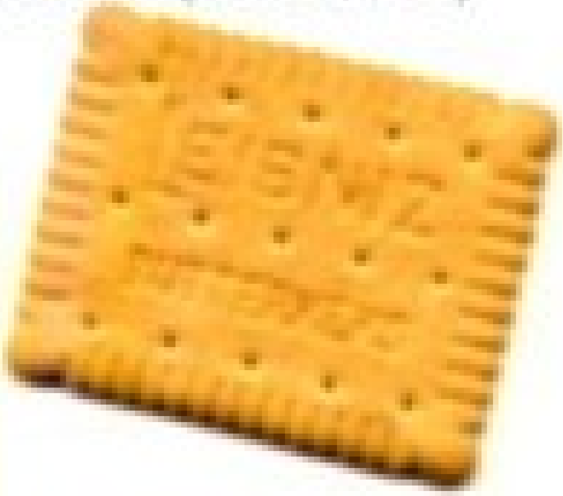


Leibnizrechenmaschine ~1694

Hilbert

- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.
- He was the first to design a machine that could do all the *four* arithmetic operations.
- This led to his optimism that machines might also prove theorems.

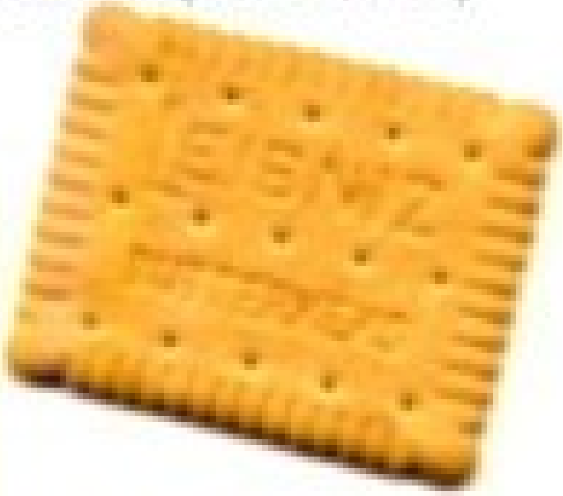
Leibniz (1646-1716)



Leibnizrechenmaschine ~1694

Hilbert

Leibniz (1646-1716)



- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.
- He was the first to design a machine that could do all the *four* arithmetic operations.
- This led to his optimism that machines might also prove theorems.

  Example 1: Angles of a triangle sum to 180°.

Leibnizrechenmaschine ~1694

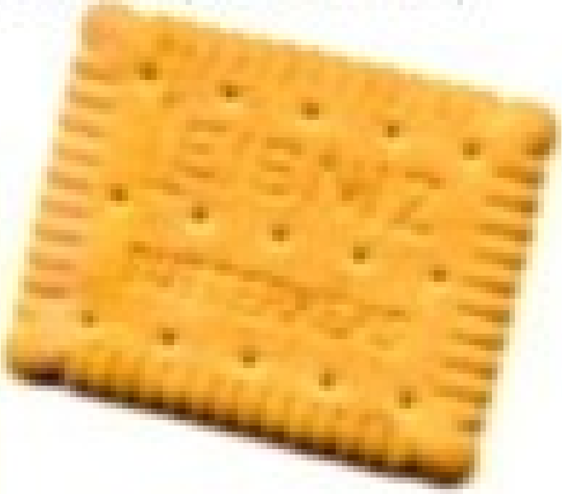
Hilbert

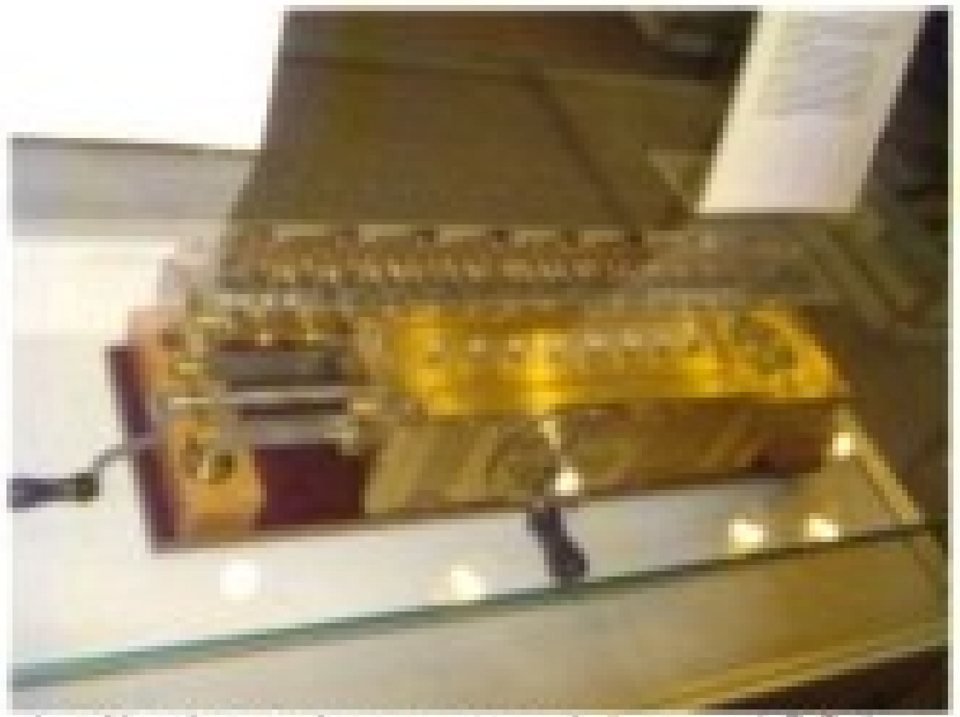
- Gottfried Leibniz dreamt of building a machine that could check the truth of math statements.
- He was the first to design a machine that could do all the *four* arithmetic operations.
- This led to his optimism that machines might also prove theorems.

  Example 1: Angles of a triangle sum to 180°.

  Example 2: Prime numbers are infinitely many.

Leibniz (1646-1716)





Leibnizrechenmaschine ~1694

Hilbert

- Leibniz's dream was generalized by Hilbert (1928), who asked for

Hilbert

- Leibniz's dream was generalized by Hilbert (1928), who asked for

  "***an <u>algorithm</u> to decide whether a given statement is provable from the axioms using the rules of logic***".

Hilbert (1862-1943)

Hilbert

- Leibniz's dream was generalized by Hilbert (1928), who asked for

  "***an <u>algorithm</u> to decide whether a given statement is provable from the axioms using the rules of logic***".

  - Known as the *Entscheidungsproblem*.

Hilbert (1862-1943)

Hilbert

- Leibniz's dream was generalized by Hilbert (1928), who asked for

  "***an <u>algorithm</u> to decide whether a given statement is provable from the axioms using the rules of logic***".

  - Known as the *Entscheidungsproblem*.

- Like Leibniz, he "believed" that there exists no *undecidable* problem!

Hilbert (1862-1943)

Hilbert

Hilbert (1862-1943)

- Leibniz's dream was generalized by Hilbert (1928), who asked for

  "***an <u>algorithm</u> to decide whether a given statement is provable from the axioms using the rules of logic***".

  - Known as the *Entscheidungsproblem*.

- Like Leibniz, he "believed" that there exists no *undecidable* problem!

- The answer first requires defining 'algorithm'.

Hilbert

Hilbert (1862-1943)

- Leibniz's dream was generalized by Hilbert (1928), who asked for

  "***an <u>algorithm</u> to decide whether a given statement is provable from the axioms using the rules of logic***".

  - Known as the *Entscheidungsproblem*.
- Like Leibniz, he "believed" that there exists no *undecidable* problem!
- The answer first requires defining 'algorithm'.
  - hence, 'computation' requires a new mathematical framework.

Contents

- Hilbert
- Church & Turing
- Cook & Levin
- Valiant's permanent
- Zero or nonzero
- Fundamental goals

Church & Turing

- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).

Church & Turing

- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).

Church (1903-1995)

Church & Turing

- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).
  - Using effective computability based on his λ-calculus.

Church (1903-1995)

Church & Turing

Church (1903-1995)

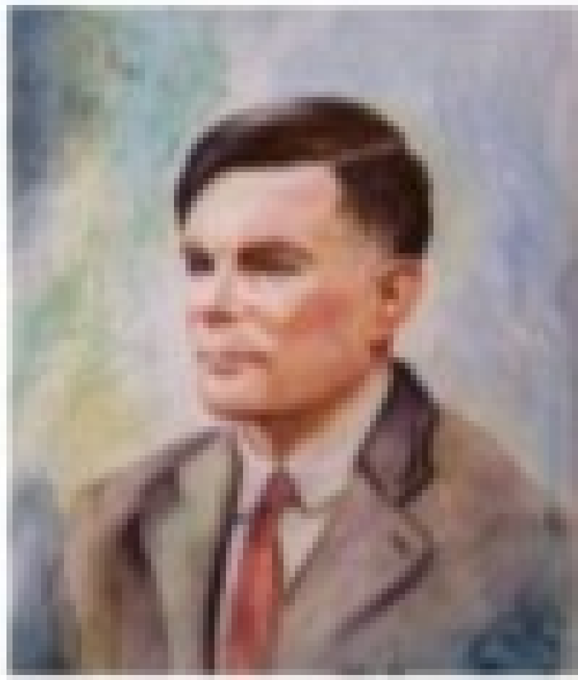
- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).
  - Using effective computability based on his λ-calculus.
  - Gave a negative answer!

Church & Turing

- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).
  - Using effective computability based on his λ-calculus.
  - Gave a negative answer!


Church (1903-1995)

- Alan Turing (1936) postulated a simple, most general, mathematical model for computing – Turing machine (TM).

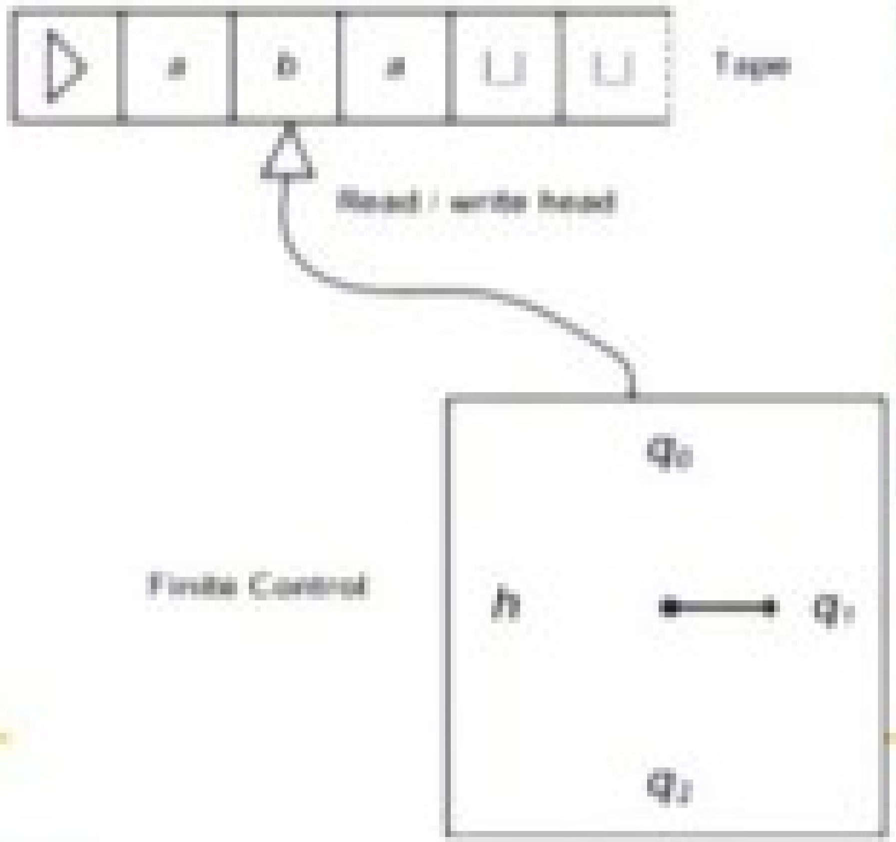

Turing (1912-1954)

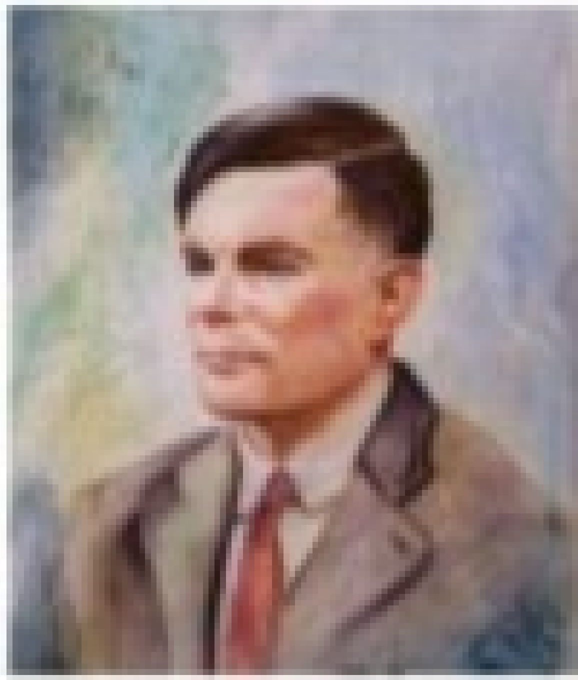
Church & Turing

- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).
  - Using effective computability based on his λ-calculus.
  - Gave a negative answer!

Church (1903-1995)

- Alan Turing (1936) postulated a simple, most general, mathematical model for computing – Turing machine (TM).





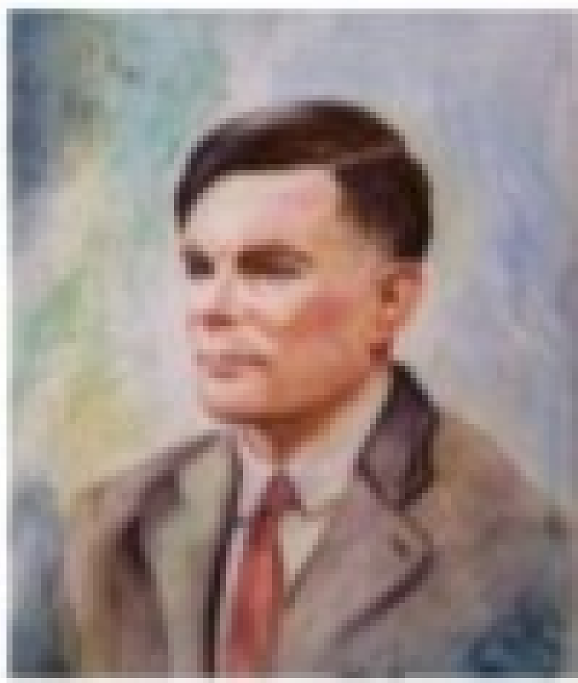
Turing (1912-1954)

Church & Turing

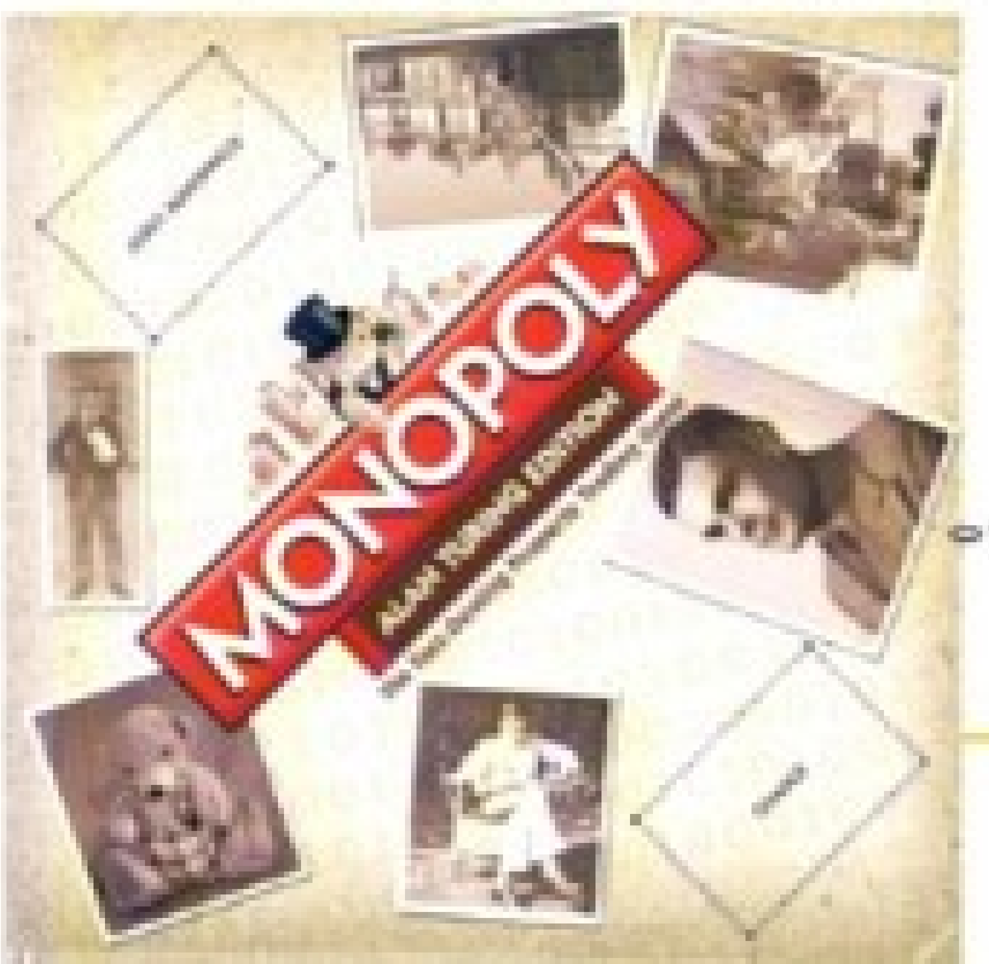
- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).
  - Using effective computability based on his λ-calculus.
  - Gave a negative answer!

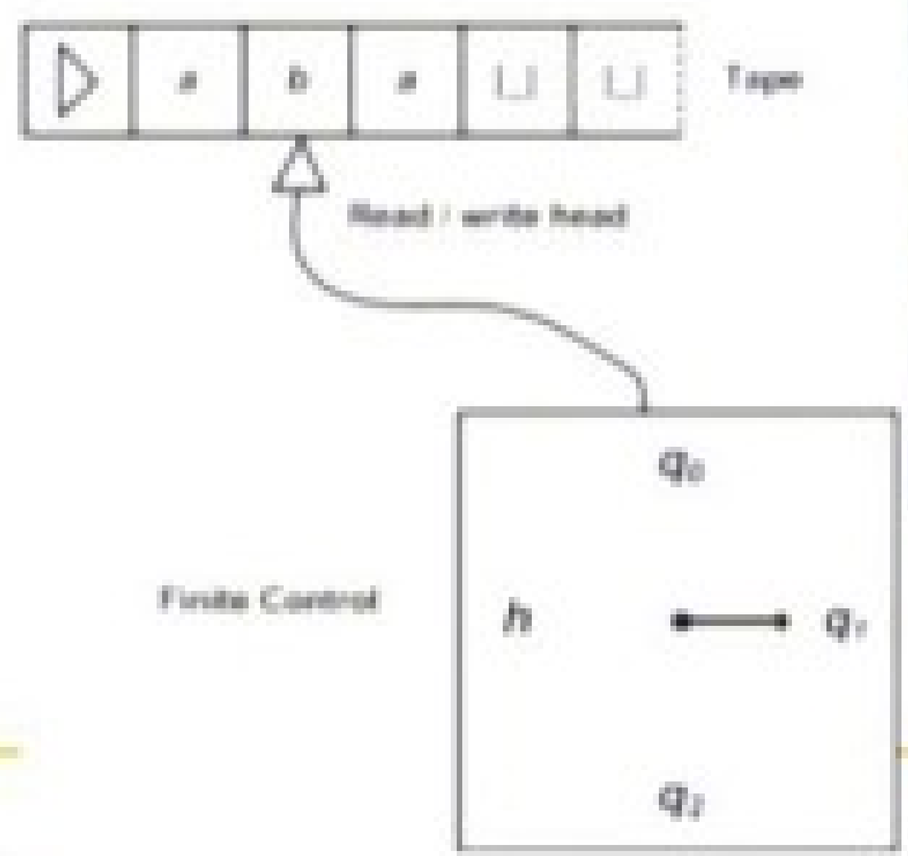
Church (1903-1995)

- Alan Turing (1936) postulated a simple, most general, mathematical model for computing – Turing machine (TM).



Turing (1912-1954)





Church & Turing

- Turing machines first appeared in the paper:

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. Turing.

[Received 28 May, 1936.—Read 12 November, 1936.]

[*Extracted from the Proceedings of the London Mathematical Society, Ser. 2, Vol. 42, 1937.*]

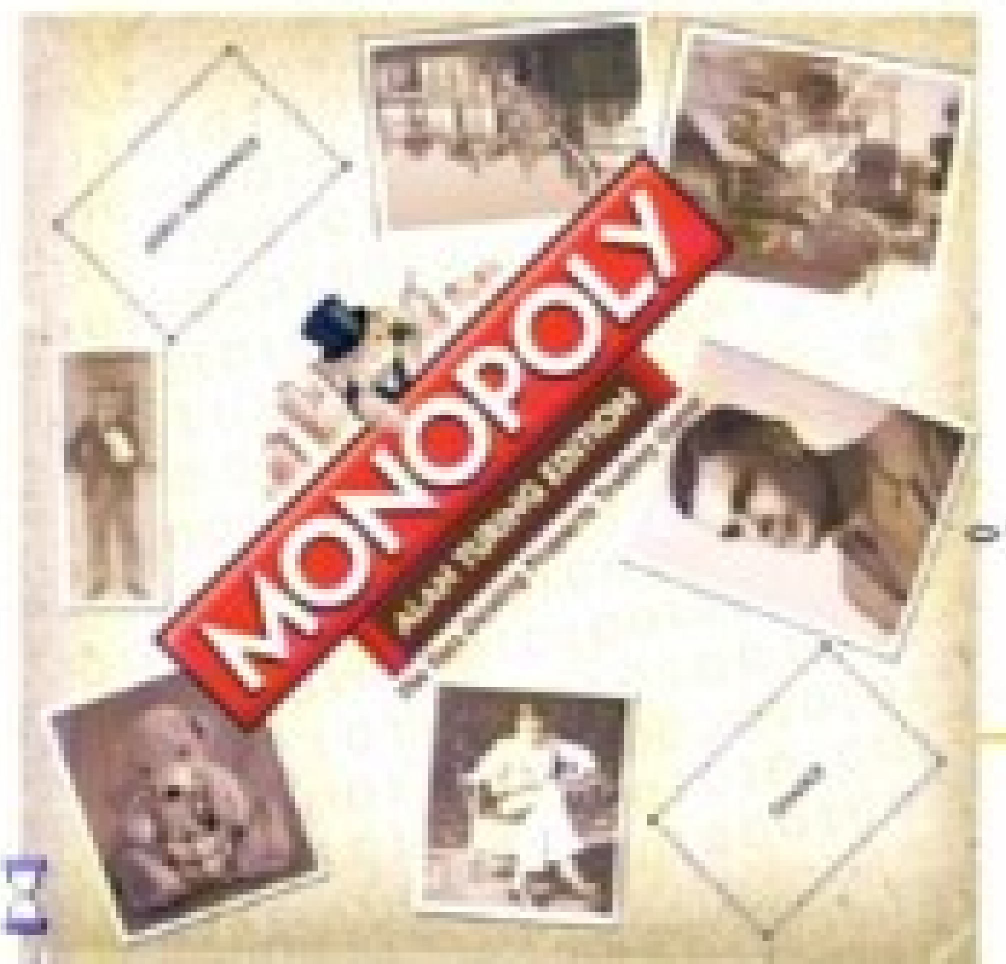
The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers

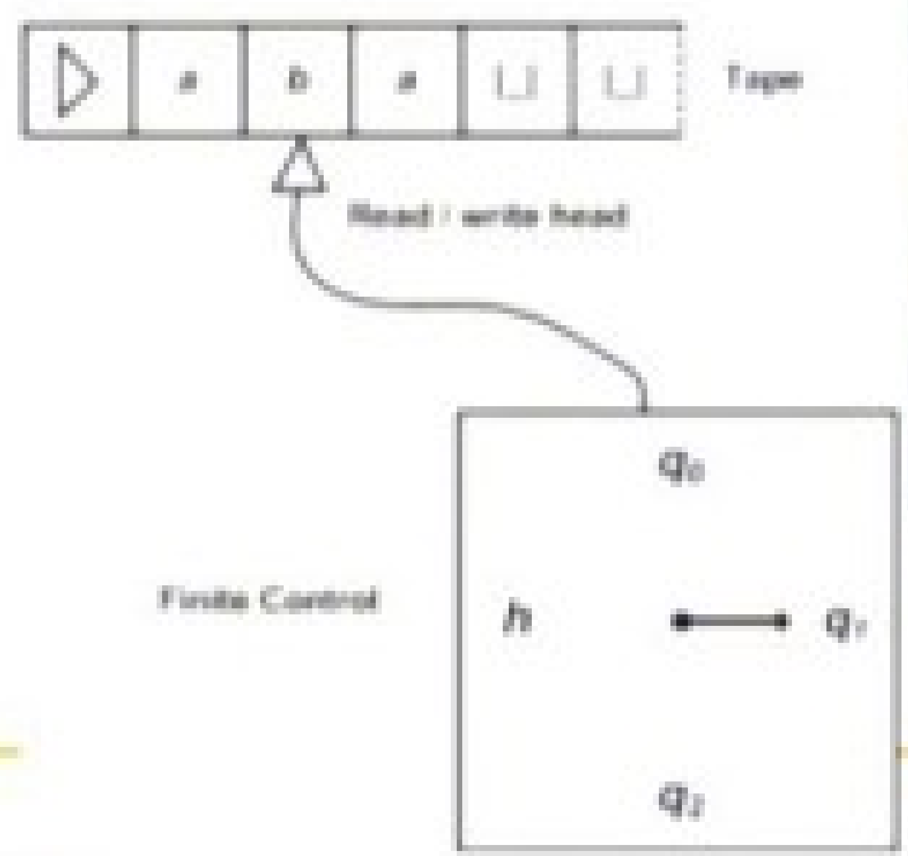
Church & Turing

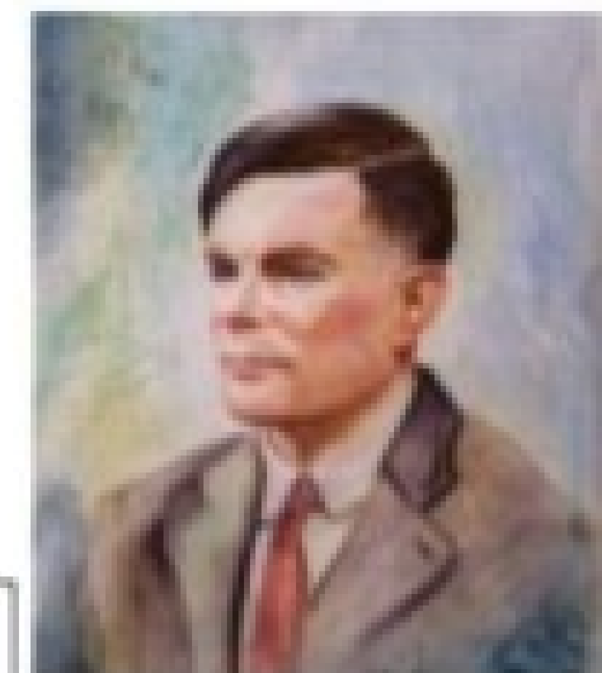
- The first response to the Entscheidungsproblem was by Alonzo Church (1935-6).
  - Using effective computability based on his λ-calculus.
  - Gave a negative answer!

Church (1903-1995)

- Alan Turing (1936) postulated a simple, most general, mathematical model for computing – Turing machine (TM).







Turing (1912-1954)

Church & Turing

- Turing machines first appeared in the paper:

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

[*Extracted from the Proceedings of the London Mathematical Society, Ser. 2, Vol. 42, 1937.*]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers

Church & Turing

- Both the proofs were motivated by Kurt Gödel.

Church & Turing

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.

Gödel (1906-1978)

Church & Turing

- Both the proofs were motivated by Kurt Gödel.

Gödel (1906-1978)

Church & Turing

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.

Gödel (1906-1978)

Church & Turing

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.
  - Deciding whether a given TM *halts* or not.

Gödel (1906-1978)

Church & Turing

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.
  - Deciding whether a given TM *halts* or not.
- Turing's proof idea for Entscheidungsproblem:
  - Enumerate the TMs as $\{M_1, M_2, M_3, \ldots\}$.

Gödel (1906-1978)

Church & Turing

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.
  - Deciding whether a given TM *halts* or not.
- Turing's proof idea for Entscheidungsproblem:
  - Enumerate the TMs as $\{M_1, M_2, M_3, \ldots\}$.
  - Let M_i be the one solving the Halting problem.

Gödel (1906-1978)

Church & Turing


Gödel (1906-1978)

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.
  - Deciding whether a given TM *halts* or not.
- Turing's proof idea for Entscheidungsproblem:
  - Enumerate the TMs as $\{M_1, M_2, M_3, \ldots\}$.
  - Let M_i be the one solving the Halting problem.
  - Consider the TM M:

On input x, if M_i rejects x(x) then ACCEPT else NOT(x(x)).

Church & Turing


Gödel (1906-1978)

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.
  - Deciding whether a given TM *halts* or not.
- Turing's proof idea for Entscheidungsproblem:
  - Enumerate the TMs as $\{M_1, M_2, M_3, \ldots\}$.
  - Let M_i be the one solving the Halting problem.
  - Consider the TM M:

    On input x, if M_i rejects $x(x)$ then ACCEPT else NOT($x(x)$).
  - What is $M(M)$??

Church & Turing

- Both the proofs were motivated by Kurt Gödel.
- Turing showed the undecidability of the Halting problem.
  - Deciding whether a given TM *halts* or not.
- Turing's proof idea for Entscheidungsproblem:
  - Enumerate the TMs as $\{M_1, M_2, M_3, \ldots\}$.
  - Let M_i be the one solving the Halting problem.
  - Consider the TM M:

    On input x, if M_i rejects $x(x)$ then ACCEPT else NOT$(x(x))$.

  - What is $M(M)$?? ↯
  - Thus, Halting problem is undecidable.

Gödel (1906-1978)

Contents

- Hilbert
- Church & Turing
- Cook & Levin
- Valiant's permanent
- Zero or nonzero
- Fundamental goals

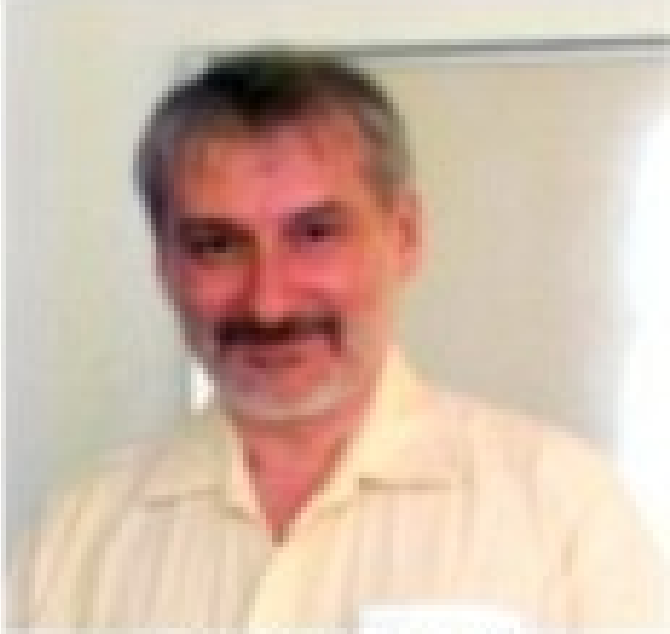
Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.

Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.

Stephen Cook (1939-)



Leonid Levin (1948-)

Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.
  - Truth of a boolean formula?

Stephen Cook (1939-)

Leonid Levin (1948-)

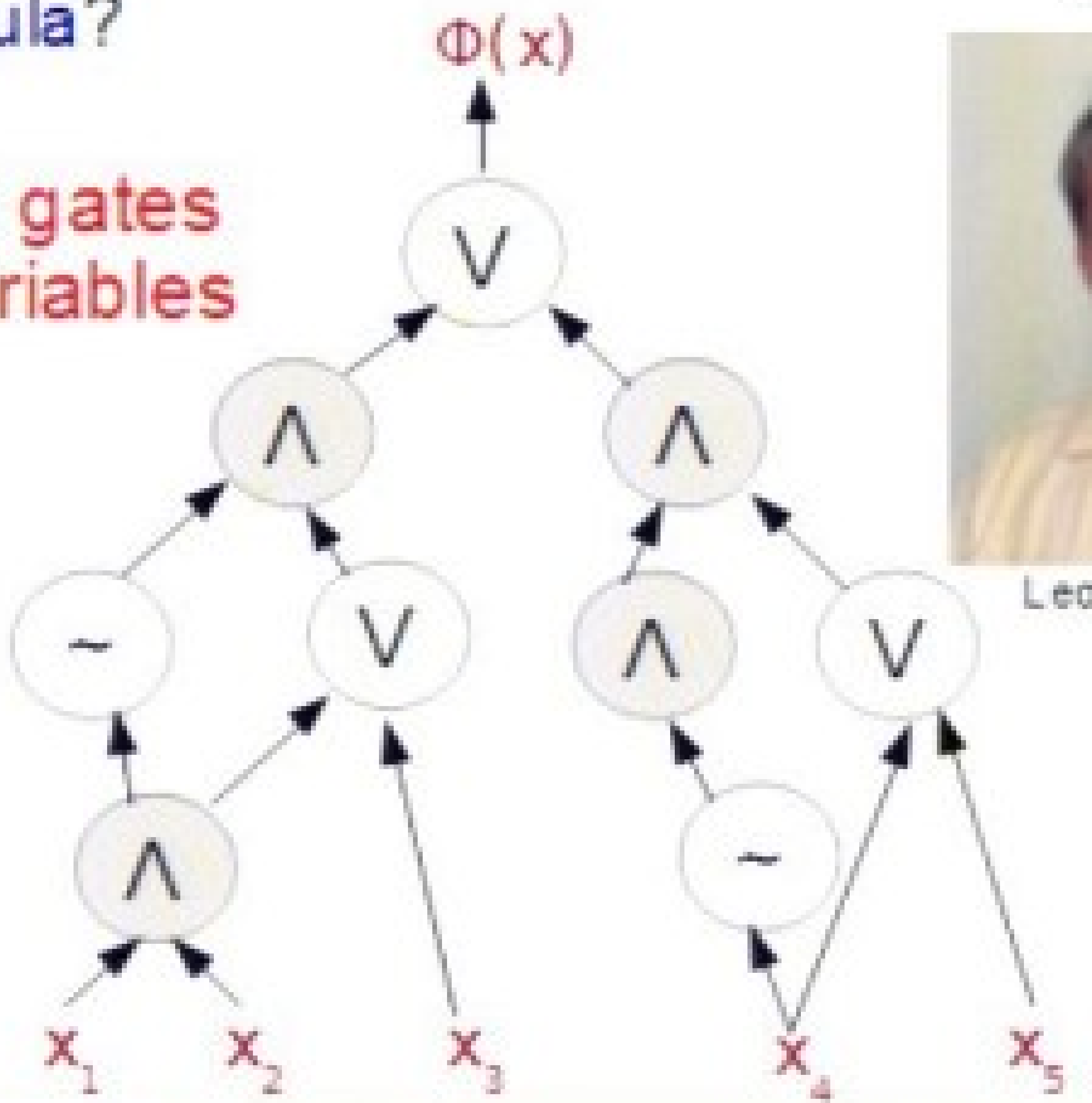
Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.
  - Truth of a boolean formula?

- A boolean formula Φ has gates {AND, OR, NOT}, and variables $\{x_1, \ldots, x_n\}$.

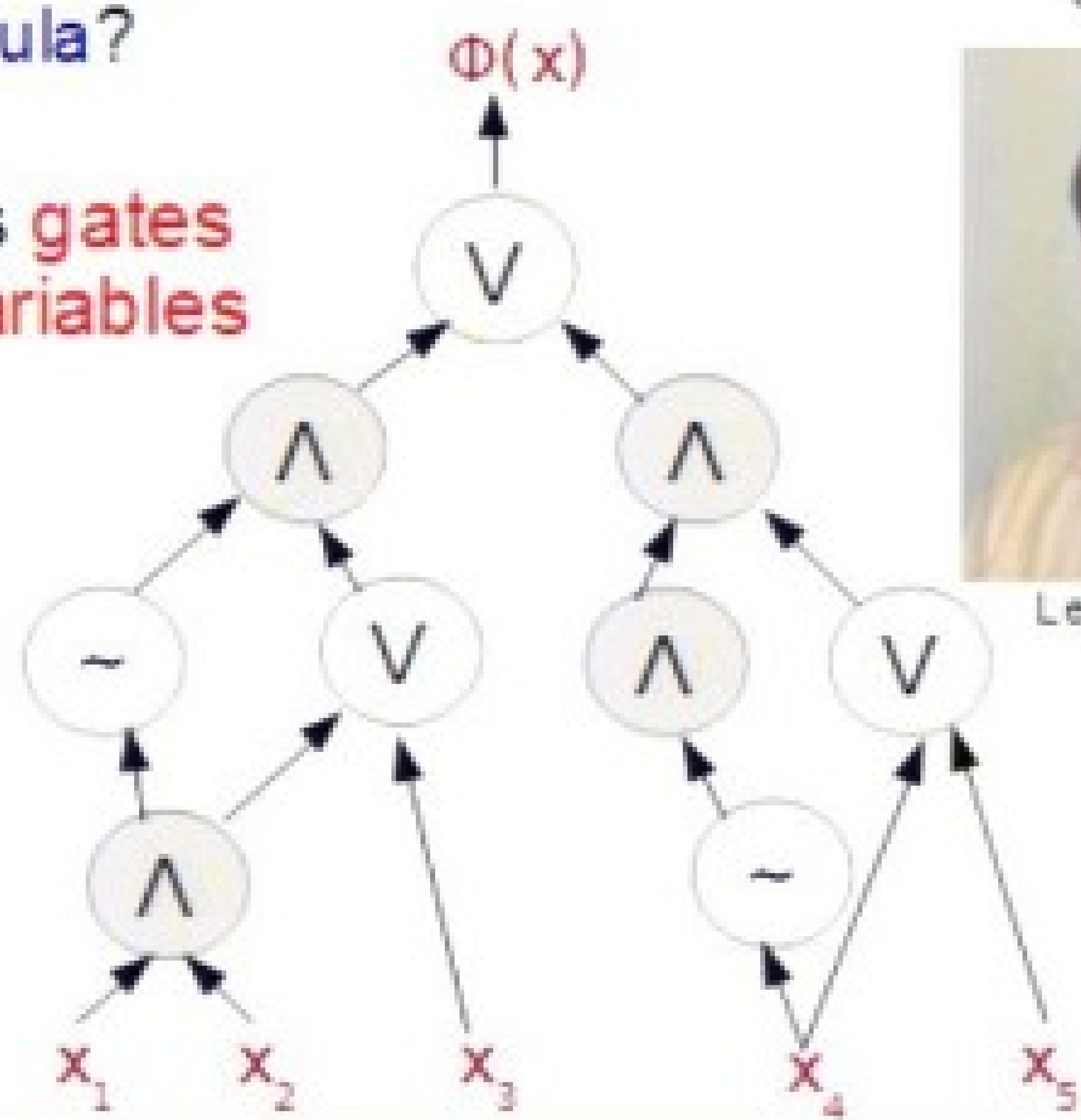
Stephen Cook (1939-)

Leonid Levin (1948-)

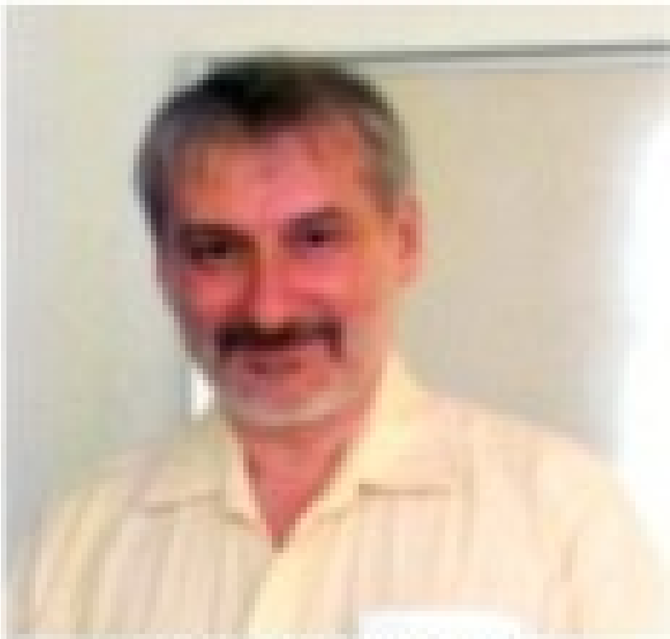


Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.
  - Truth of a boolean formula?
- A boolean formula Φ has gates {AND, OR, NOT}, and variables $\{x_1, \ldots, x_n\}$.
- We can try out all 2^n evaluations for truth.

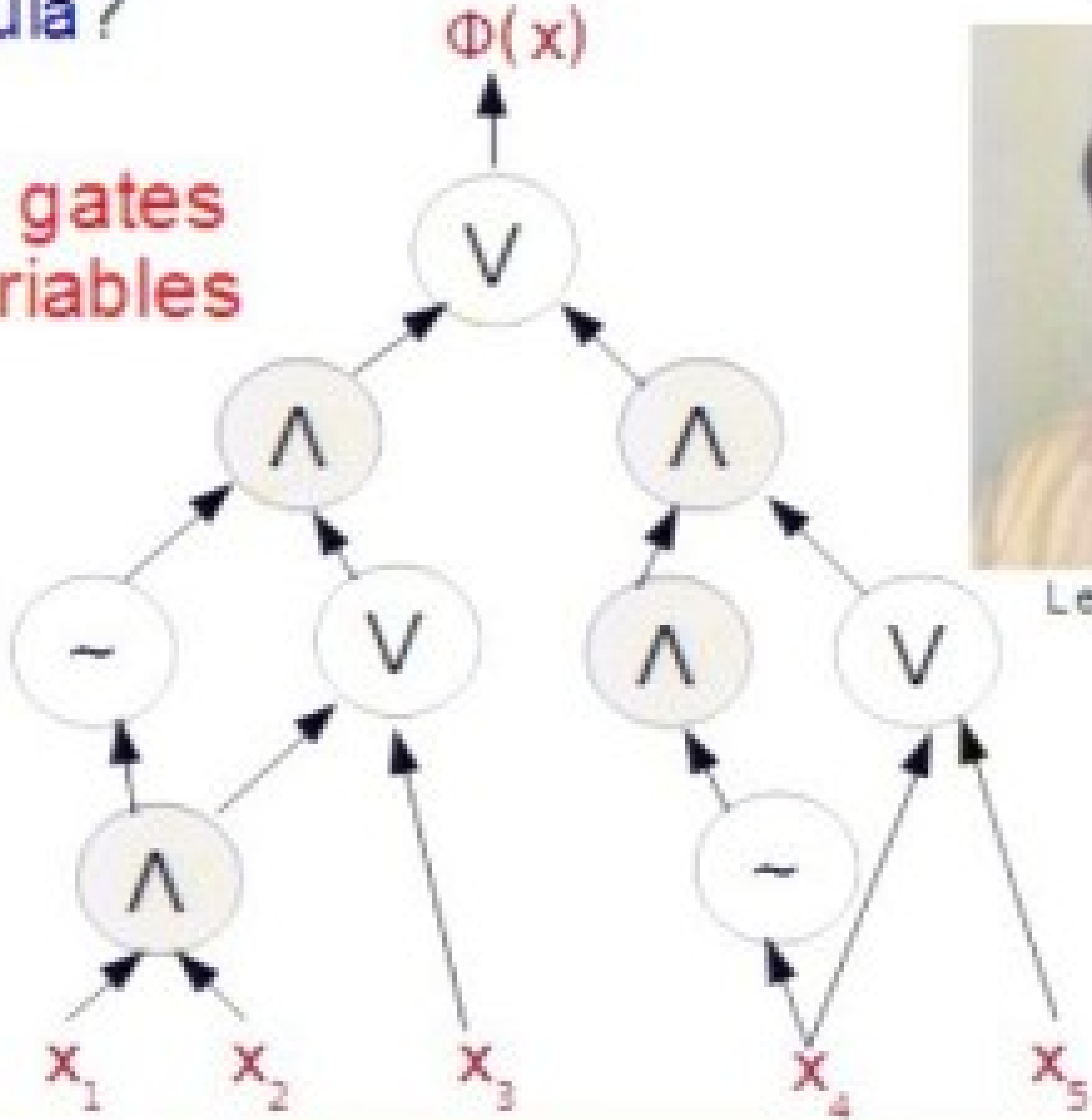



Stephen Cook (1939-)


Leonid Levin (1948-)

Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.
  - Truth of a boolean formula?
- A boolean formula Φ has gates {AND, OR, NOT}, and variables $\{x_1, \ldots, x_n\}$.
- We can try out all 2^n evaluations for truth.
  - Is there a faster way?

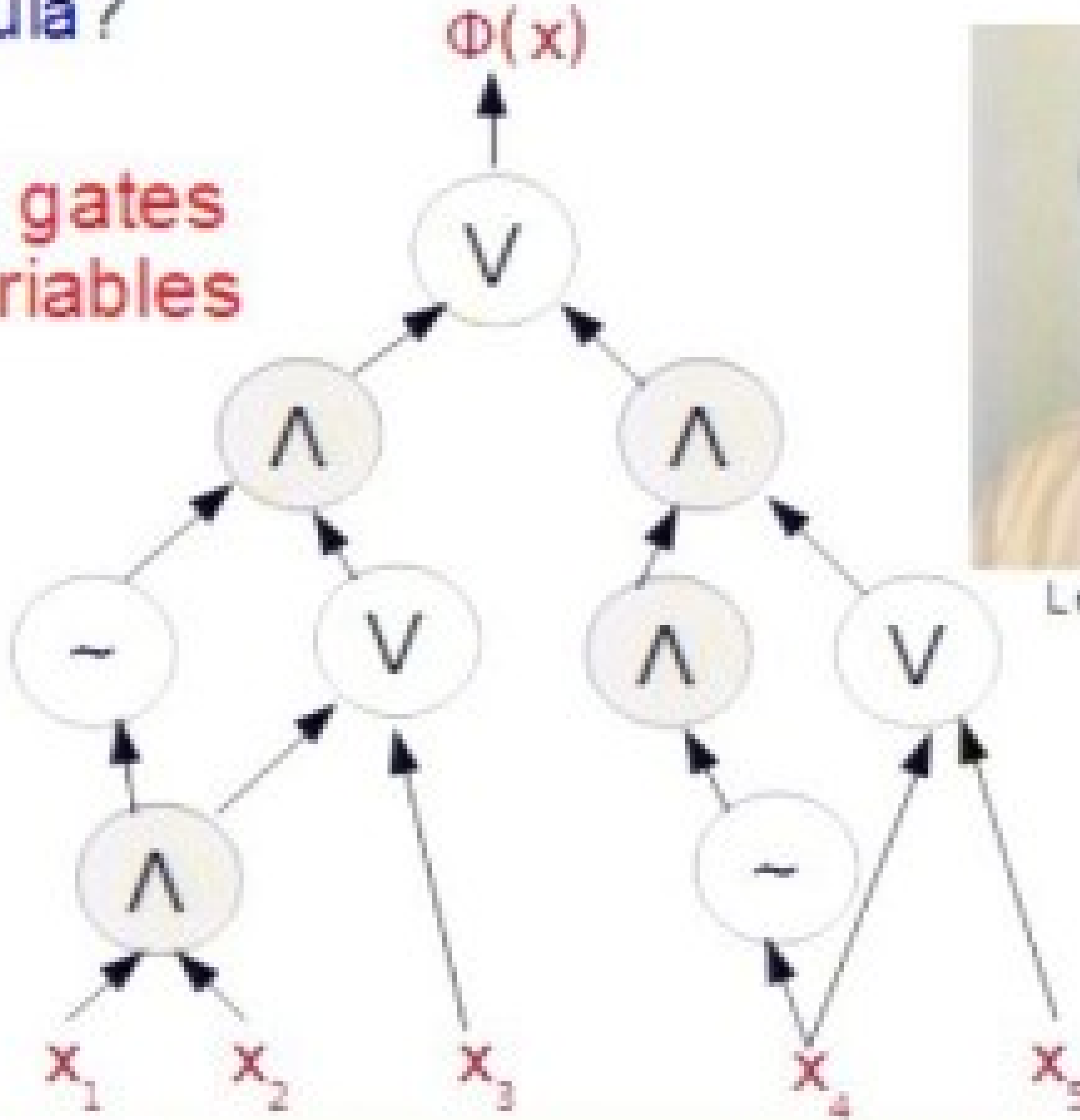


Stephen Cook (1939-)

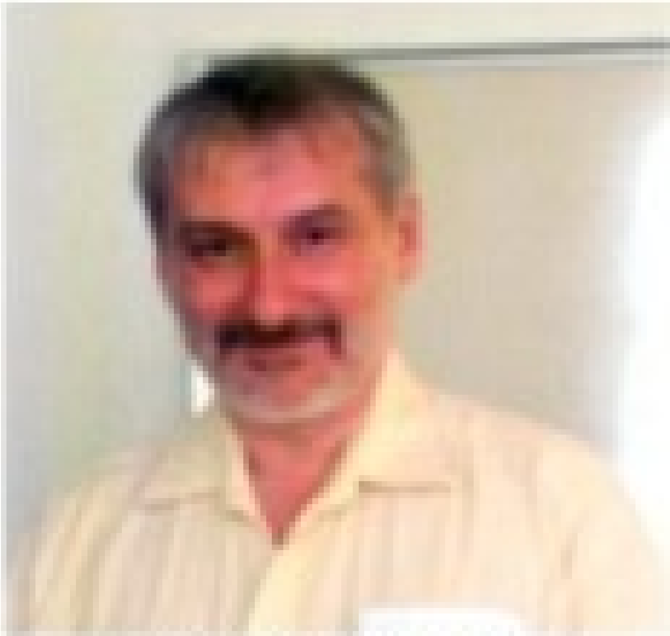
Leonid Levin (1948-)

Cook & Levin

- Cook & Levin (1971) studied a more tractable version of the *Entscheidungsproblem*.
  - Truth of a boolean formula?
- A boolean formula Φ has gates {AND, OR, NOT}, and variables $\{x_1, \ldots, x_n\}$.
- We can try out all 2^n evaluations for truth.
  - Is there a faster way?
- Move from decidability to <u>efficiency</u>……




Stephen Cook (1939-)


Leonid Levin (1948-)

Cook & Levin

- Intuitively, one cannot do any better than the *exponential*, i.e. 2^n, time.

Cook & Levin

- Intuitively, one cannot do any better than the *exponential*, i.e. 2^n, time.
  - This is the $\mathcal{P}$ *vs* $\mathcal{NP}$ question.

Cook & Levin

- Intuitively, one cannot do any better than the *exponential*, i.e. 2^n, time.
  - This is the $\mathcal{P}$ *vs* $\mathcal{NP}$ question.
  - Worth at least a *million* $$!

Clay Mathematics Institute (1999-)

Cook & Levin

- Intuitively, one cannot do any better than the *exponential*, i.e. 2^n, time.
  - This is the $\mathcal{P}$ vs $\mathcal{NP}$ question.
  - Worth at least a *million* $$!

Clay Mathematics Institute (1999-)

- This is an extremely important problem because 100s of *practical* problems are known to be equivalent to it.

Cook & Levin

- Intuitively, one cannot do any better than the *exponential*, i.e. 2^n, time.
  - This is the $\mathcal{P}$ vs $\mathcal{NP}$ question.
  - Worth at least a *million* $$!

Clay Mathematics Institute (1999-)

- This is an extremely important problem because 100s of *practical* problems are known to be equivalent to it.
  - Karp (1972) himself showed 21 such problems!

Richard Karp (1935-)

Cook & Levin

- Intuitively, one cannot do any better than the *exponential*, i.e. 2^n, time.
  - This is the $\mathcal{P}$ *vs* $\mathcal{NP}$ question.
  - Worth at least a *million* $$!

Clay Mathematics Institute (1999-)

- This is an extremely important problem because 100s of *practical* problems are known to be equivalent to it.
  - Karp (1972) himself showed 21 such problems!

Richard Karp (1935-)

Integer programming, set packing, vertex cover, feedback node set, hamiltonian cycle, chromatic number, clique, steiner tree, 3-dimensional matching, knapsack, job sequencing, partition, Max cut, independent set problem, Travelling salesmen problem

Contents

- Hilbert
- Church & Turing
- Cook & Levin
- Valiant's permanent
- Zero or nonzero
- Fundamental goals

Valiant's permanent

- Valiant (1977) asked a related question.

Valiant's permanent

- Valiant (1977) asked a related question.

Leslie Valiant (1949-)

Valiant's permanent

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.

Leslie Valiant (1949-)

Valiant's permanent

Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.

Valiant's permanent

Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.
- Solving this would solve all our previous NP-hard problems.

Valiant's permanent

Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.
- Solving this would solve all our previous NP-hard problems.
- More interestingly, the counting problem reduces to a simple matrix question – Permanent.

Valiant's permanent

Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.
- Solving this would solve all our previous NP-hard problems.
- More interestingly, the counting problem reduces to a simple matrix question – Permanent.

$$Per\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33} +$$

Valiant's permanent

Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.

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

- More interestingly, the counting problem reduces to a simple matrix question – Permanent.

$$Per \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{21}a_{33} +$$

Valiant's permanent


Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.

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

- More interestingly, the counting problem reduces to a simple matrix question – Permanent.

$$Per \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{21}a_{33} + a_{13}a_{22}a_{31} +$$

Valiant's permanent


Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.
- Solving this would solve all our previous NP-hard problems.
- More interestingly, the counting problem reduces to a simple matrix question – Permanent.

$$Per\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33}+a_{12}a_{21}a_{33}+a_{13}a_{22}a_{31}+a_{11}a_{23}a_{32}+a_{12}a_{23}a_{31}+a_{13}a_{21}a_{32}.$$

Valiant's permanent

Leslie Valiant (1949-)

- Valiant (1977) asked a related question.
  - Count the number of *good* evaluations of a given boolean formula.
  - Valiant's counting problem.

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

- More interestingly, the counting problem reduces to a simple matrix question – Permanent.

$$Per\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33} + a_{12}a_{21}a_{33} + a_{13}a_{22}a_{31} + a_{11}a_{23}a_{32} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32}.$$

Given a matrix A compute Per(A) ?

Valiant's permanent

- Notice that permanent looks very much like a determinant.

Valiant's permanent

- Notice that permanent looks very much like a determinant.

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

Valiant's permanent

- Notice that permanent looks very much like a determinant.

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

Valiant's permanent

- Notice that permanent looks very much like a determinant.

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

- It is an old question of Pólya (1913): Can permanent be computed using the determinant?

Valiant's permanent

- Notice that permanent looks very much like a determinant.

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

- It is an old question of Pólya (1913): Can permanent be computed using the determinant?

George Pólya (1887-1985)

Valiant's permanent

- Notice that permanent looks very much like a determinant.

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


George Pólya (1887-1985)

- It is an old question of Pólya (1913): Can permanent be computed using the determinant?

- Valiant's study suggests that permanent is a much harder sibling of determinant!

Valiant's permanent

- Notice that permanent looks very much like a determinant.

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

- It is an old question of Pólya (1913): Can permanent be computed using the determinant?

George Pólya (1887-1985)

- Valiant's study suggests that permanent is a much harder sibling of determinant!

- *Algebraic P vs NP question:*

  Is permanent efficiently computable ?

Zero or nonzero

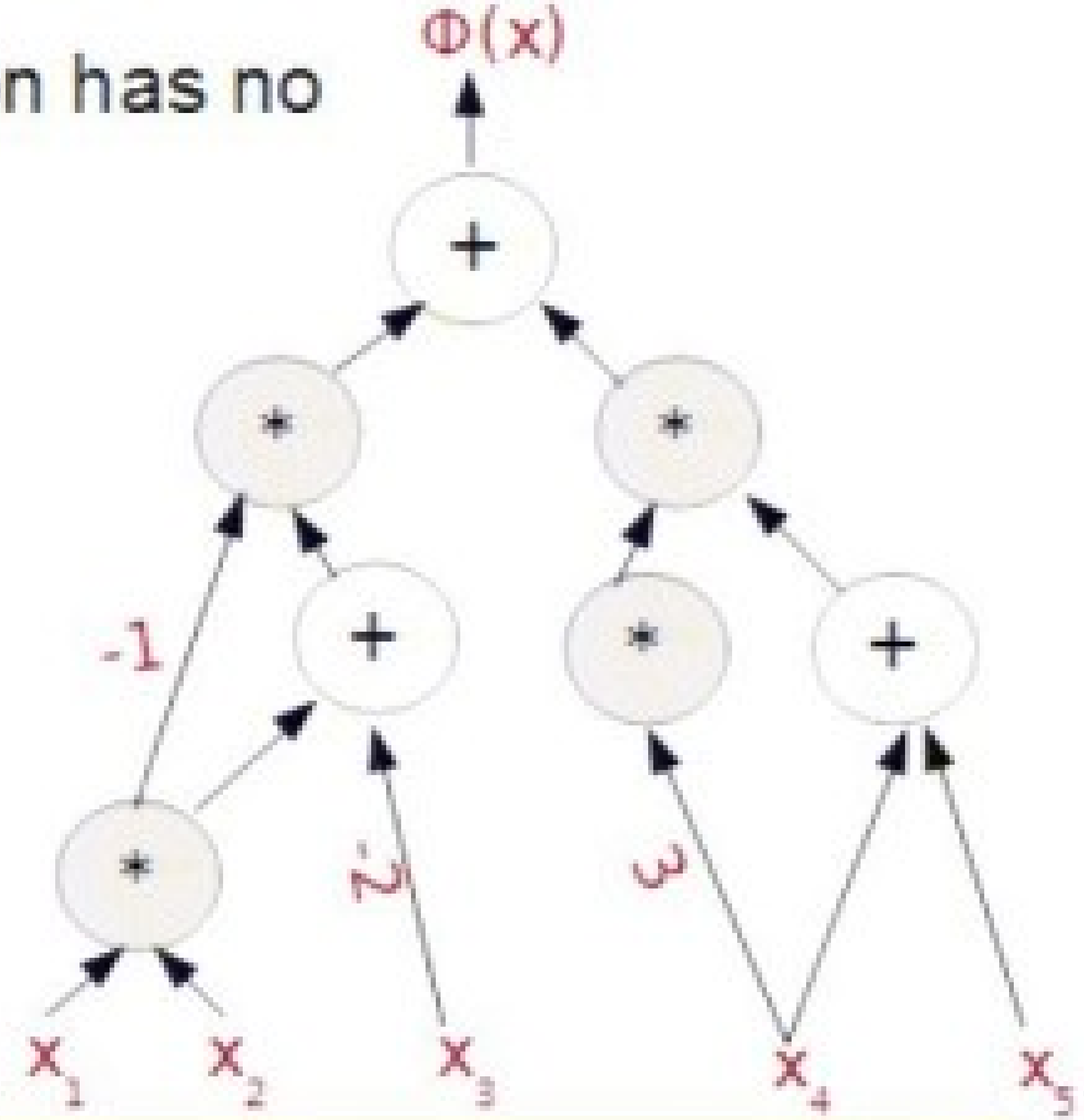
- The current research focuses on proving permanent's hardness – by algebraic means.

Zero or nonzero

- The current research focuses on proving permanent's hardness – by algebraic means.
- I.e. show that the permanent function has no *small* arithmetic circuit.

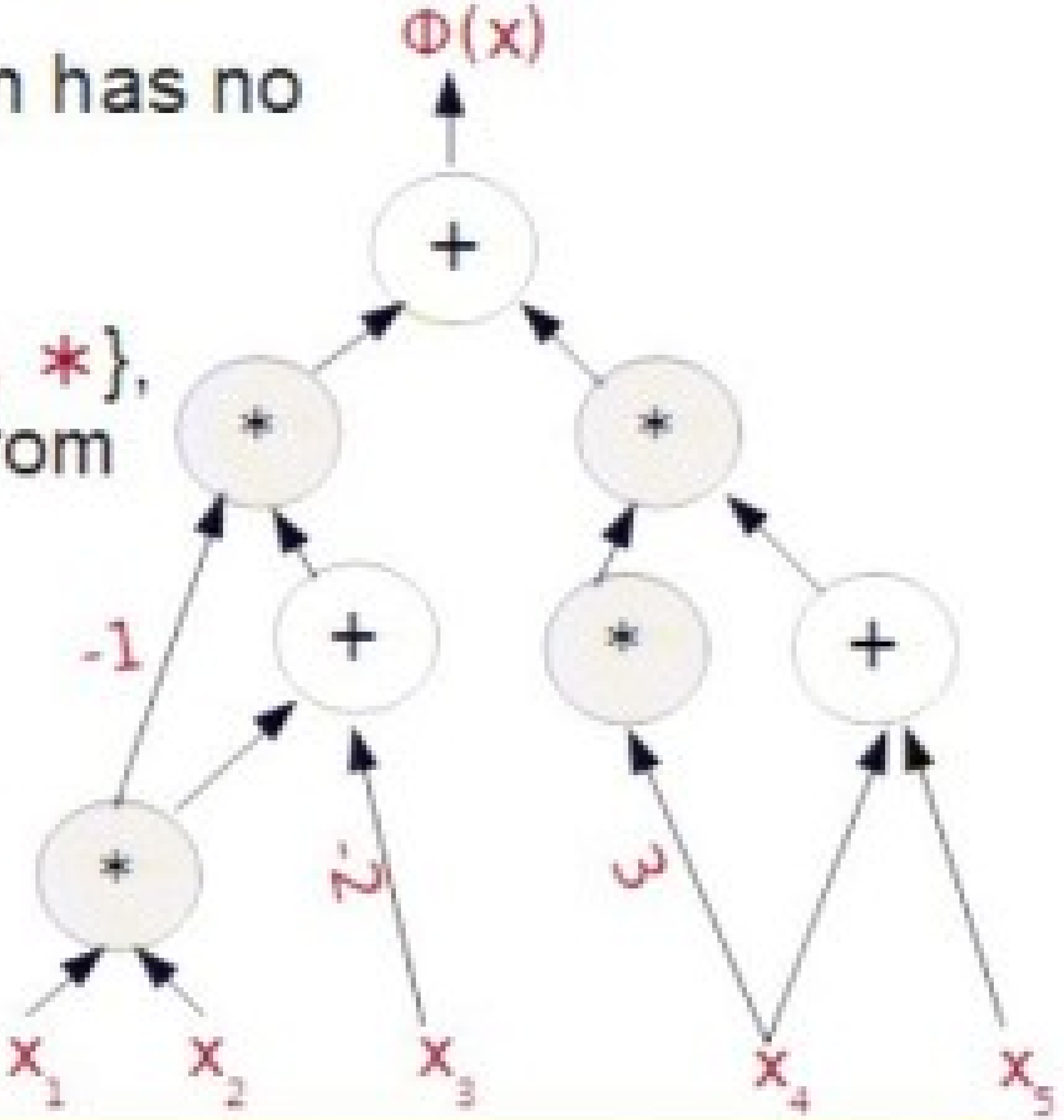
Zero or nonzero

- The current research focuses on proving permanent's hardness – by algebraic means.
- I.e. show that the permanent function has no *small* arithmetic circuit.



Zero or nonzero

- The current research focuses on proving permanent's hardness – by algebraic means.
- I.e. show that the permanent function has no *small* arithmetic circuit.
- An arithmetic circuit Φ has gates $\{+, *\}$, variables $\{x_1, \ldots, x_n\}$ and constants from some field F.
- An arithmetic circuit is an algebraically neat model to capture real computation.

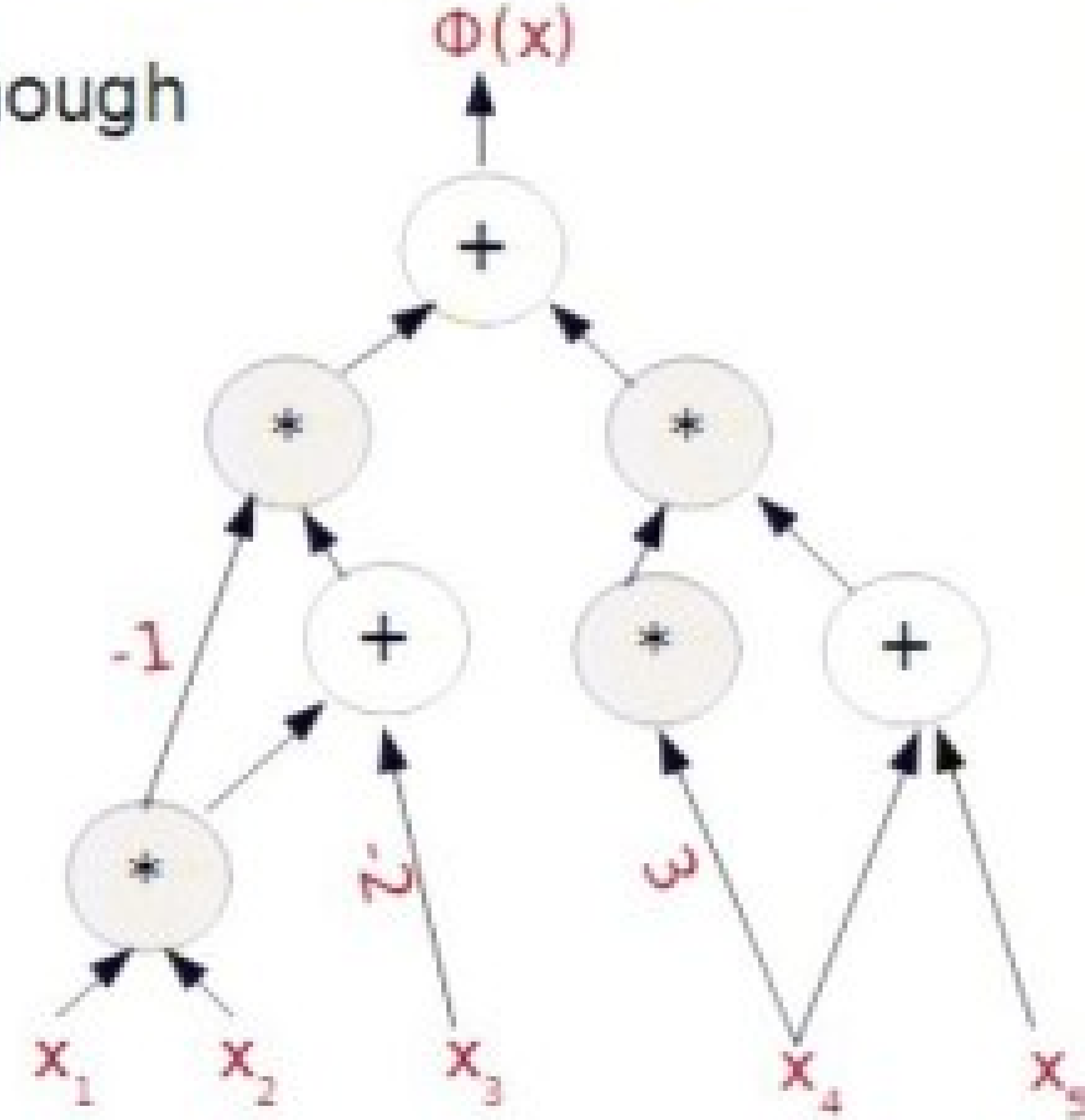


Zero or nonzero

- *Conjecture: Permanent has no small arithmetic circuits.*

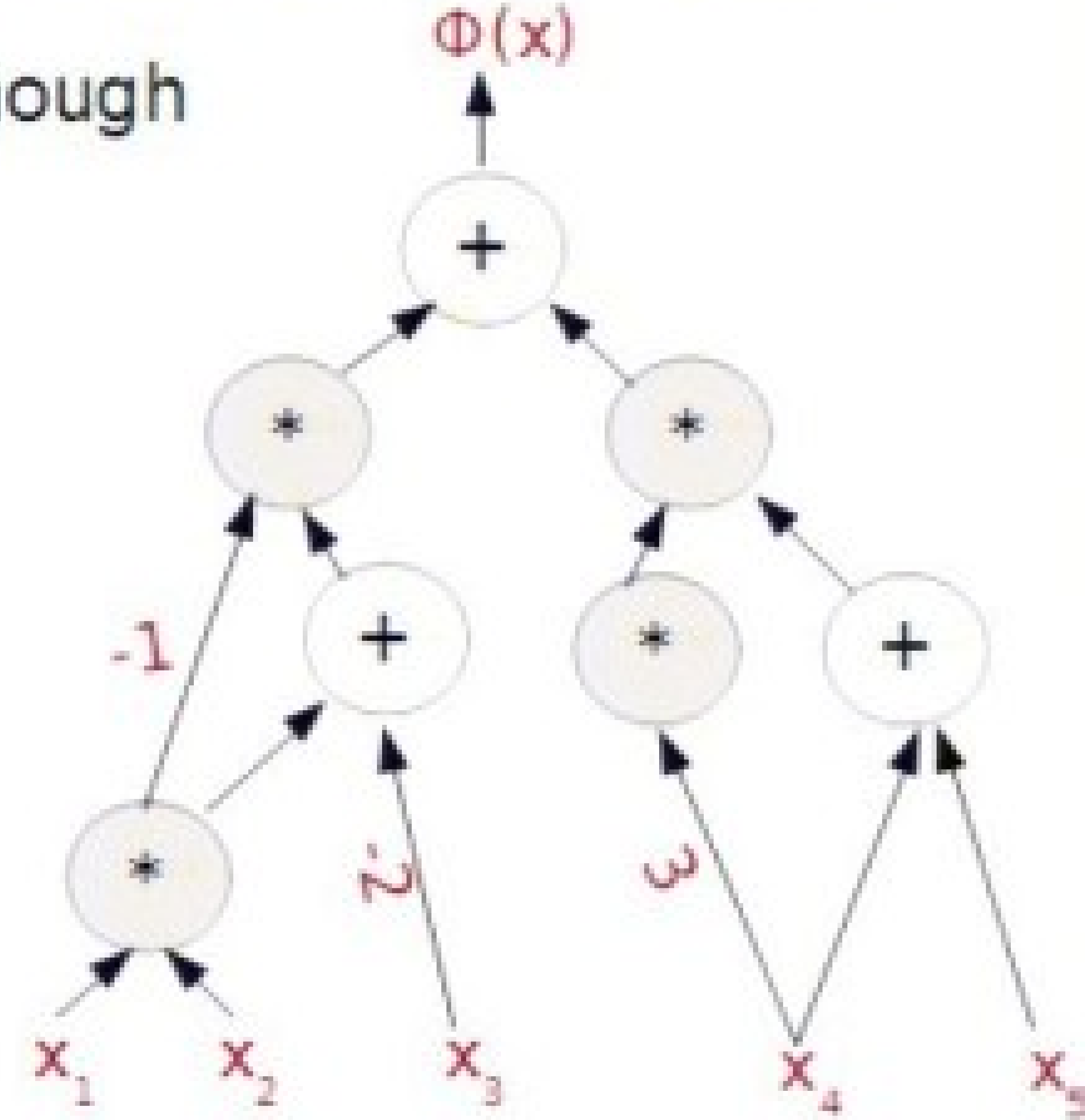
Zero or nonzero

- *Conjecture: Permanent has no small arithmetic circuits.*
- Classical algebra is not developed enough to answer this question.



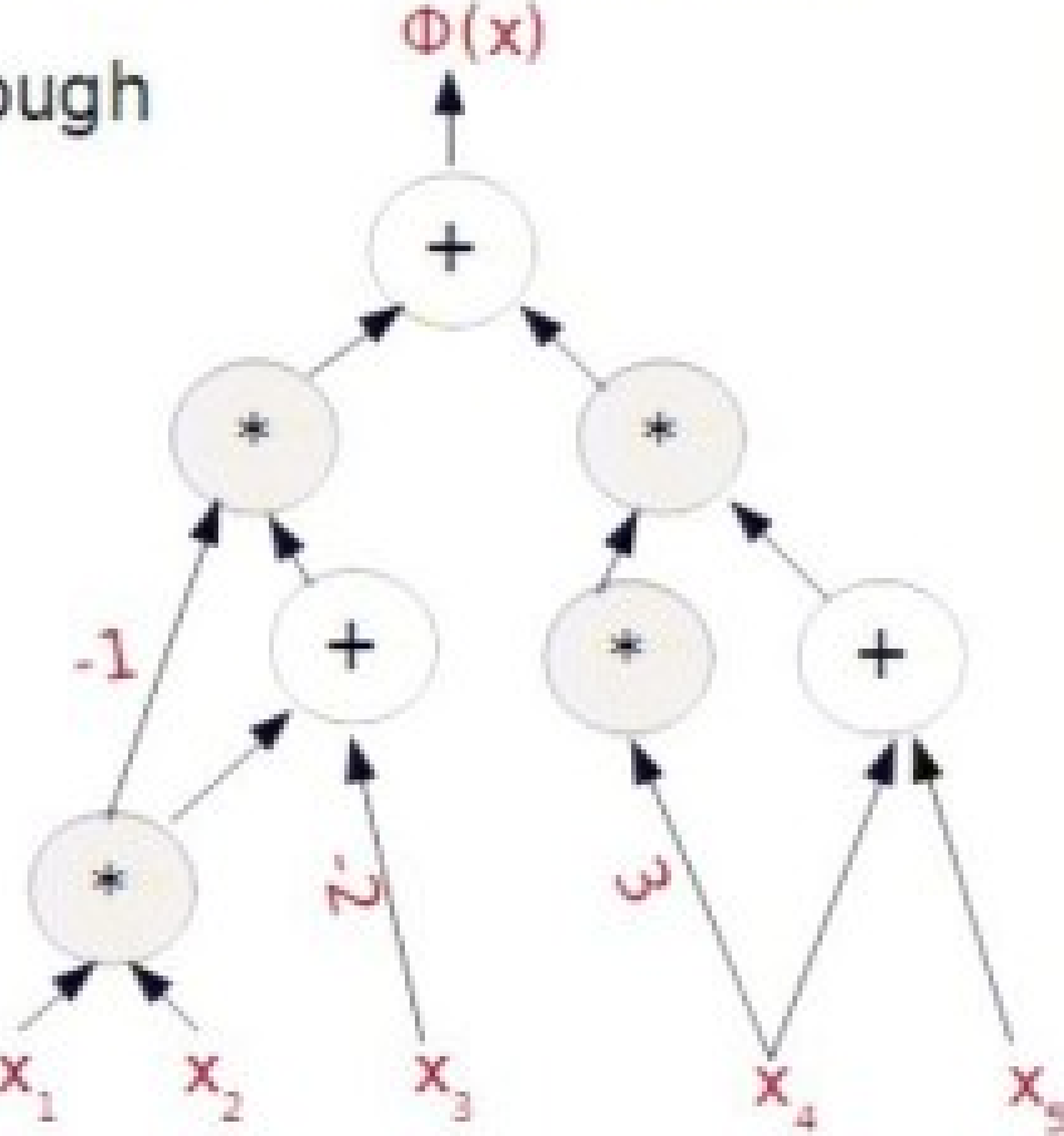
Zero or nonzero

- *Conjecture: Permanent has no small arithmetic circuits.*
- Classical algebra is not developed enough to answer this question.
  - Permanent, circuits are both recent constructs.



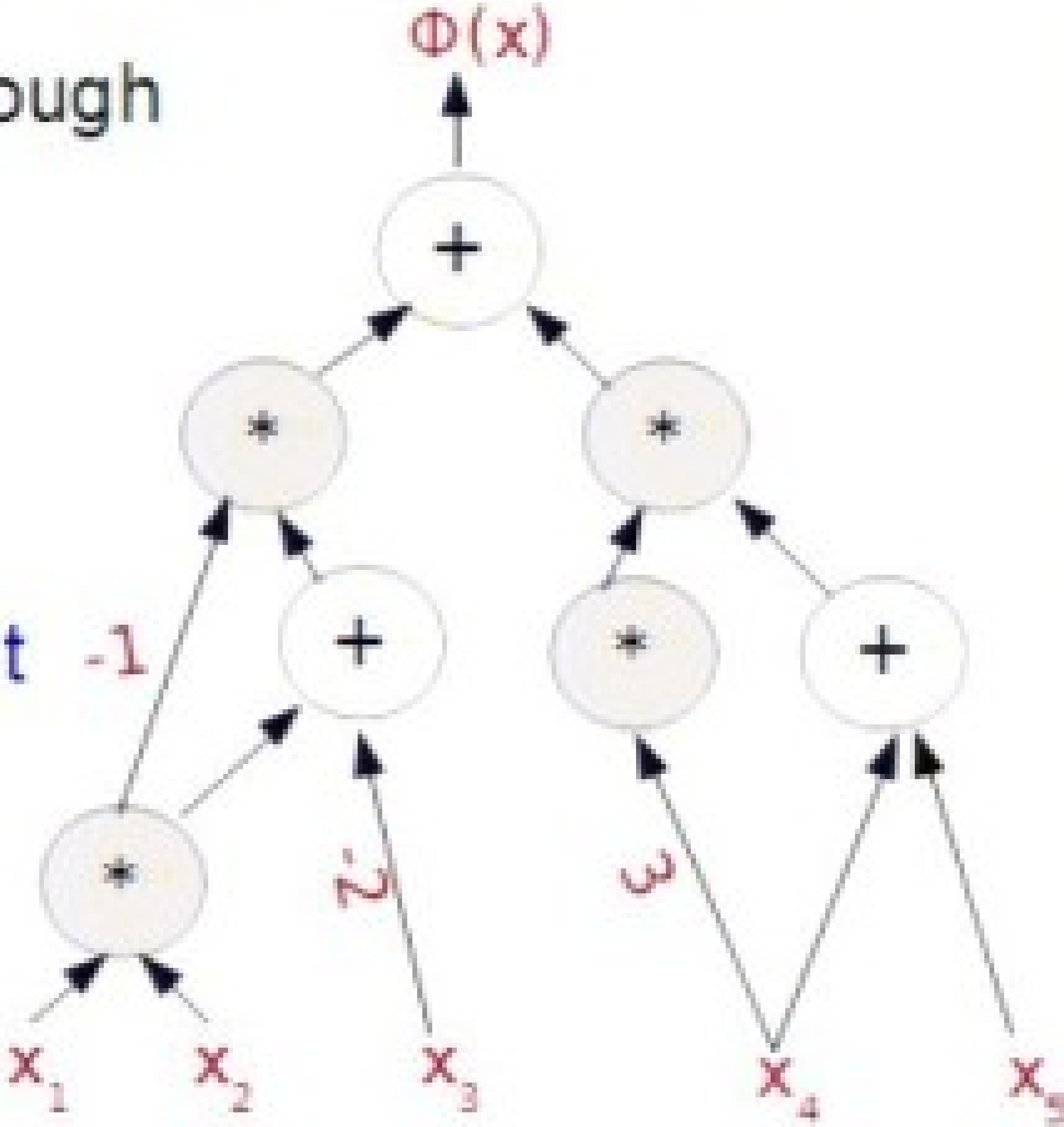
Zero or nonzero

- *<u>Conjecture</u>: Permanent has no small arithmetic circuits.*
- Classical algebra is not developed enough to answer this question.
  - Permanent, circuits are both recent constructs.
  - A specialized theory is missing.



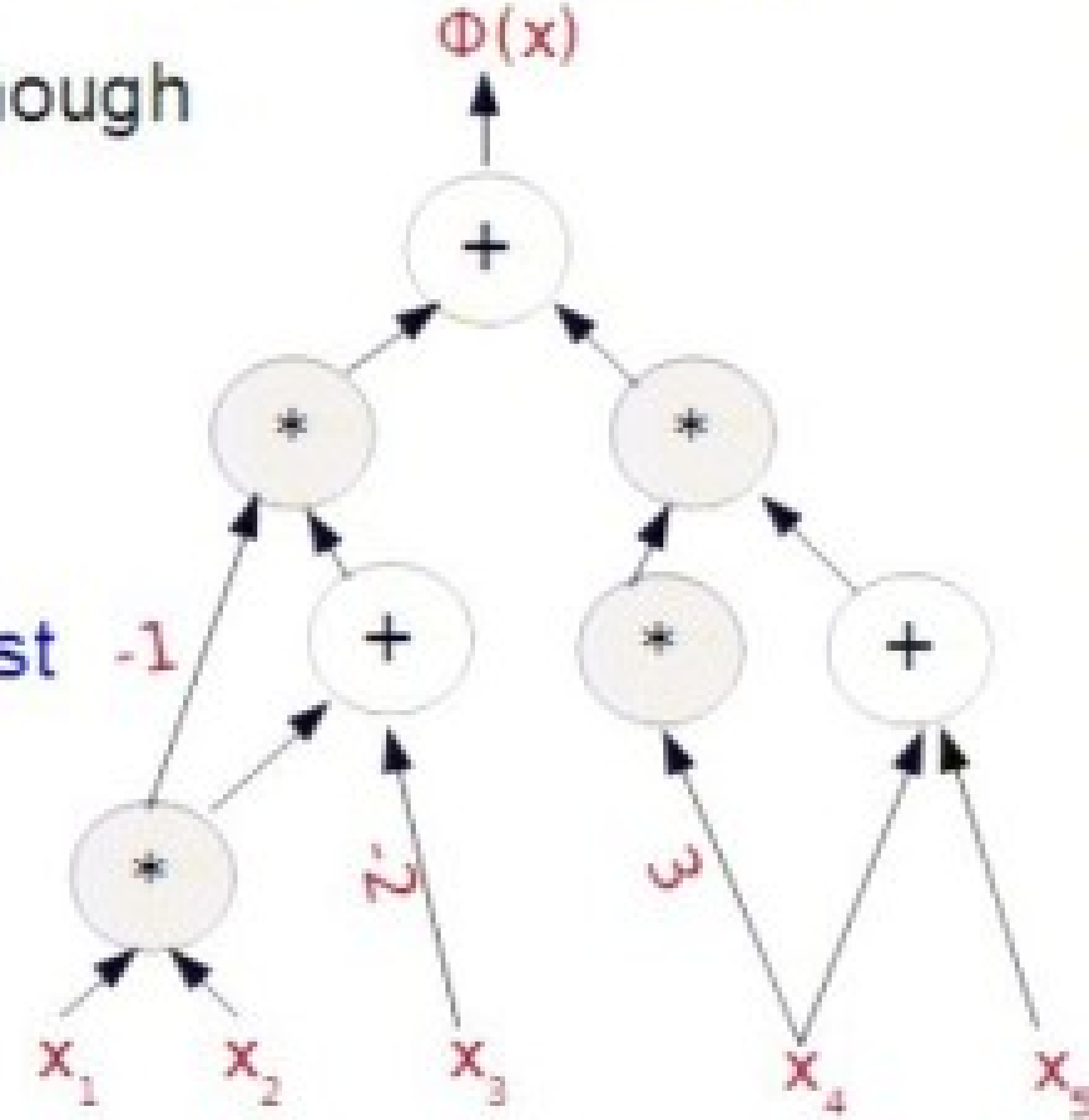
Zero or nonzero

- *Conjecture: Permanent has no small arithmetic circuits.*
- Classical algebra is not developed enough to answer this question.
  - Permanent, circuits are both recent constructs.
  - A specialized theory is missing.
- As a warmup: Find an algorithm to test whether a given circuit is *zero*.



Zero or nonzero

- *Conjecture: Permanent has no small arithmetic circuits.*
- Classical algebra is not developed enough to answer this question.
  - Permanent, circuits are both recent constructs.
  - A specialized theory is missing.
- As a warmup: Find an algorithm to test whether a given circuit is *zero*.
  - Identity testing.



Meta-Theorem: A solution of identity testing would answer the permanent question.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.
- Prove *permanent* hardness against circuits.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.
- Prove *permanent* hardness against circuits.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.
- Prove *permanent* hardness against circuits.
- Resolve *algebraic* P vs NP (Valiant's *counting* problem).

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.
- Prove *permanent* hardness against circuits.
- Resolve *algebraic* P vs NP (Valiant's *counting* problem).
- Resolve *P vs NP*.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.
- Prove *permanent* hardness against circuits.
- Resolve *algebraic* P vs NP (Valiant's *counting* problem).
- Resolve *P vs NP*.

The tools therein shall enrich our understanding.

Fundamental Goals

- Find a *proper* algorithm for circuit *identity testing*.
- Prove *permanent* hardness against circuits.
- Resolve *algebraic* P vs NP (Valiant's *counting* problem).
- Resolve *P vs NP*.

The tools therein shall enrich our understanding.

Thank you!