Turing, Church, Gödel, Computability, Complexity and Logic, a Personal View

Michael O. Rabin
Hebrew University, Harvard University

Alan M. Turing Conference – Boston University

Origins

- Hilbert 1928: Find an automatic computational procedure to determine if **S** is a theorem.

- Mathematical logic begets computability

- Turing 1936:

What does automatically computable mean?

- Church: Lambda Calculus.

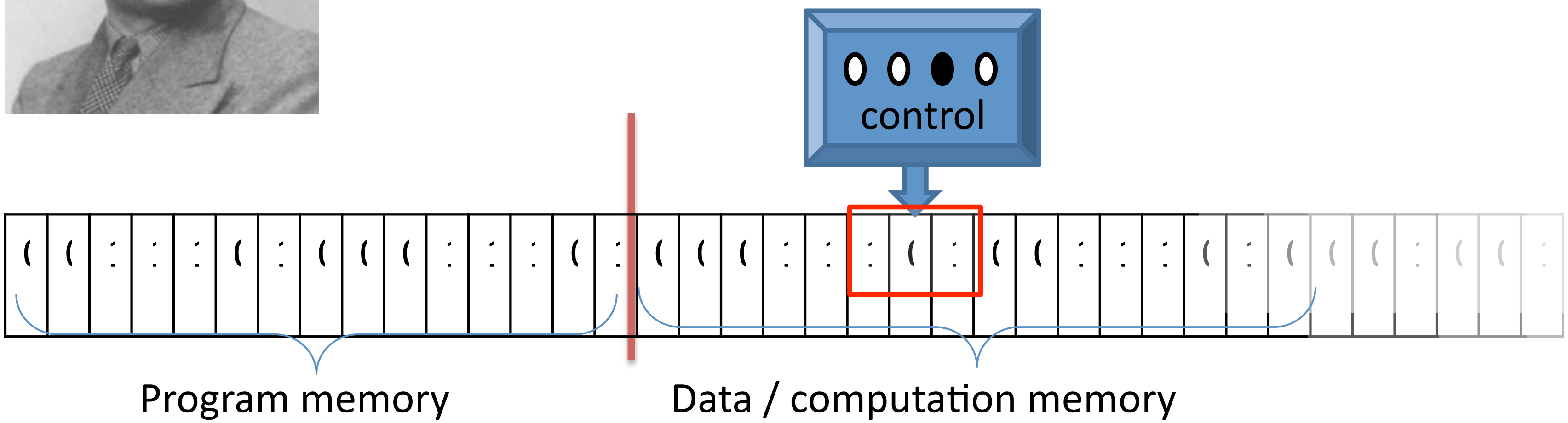
- Gödel: (Primitive) Recursive Functions.

Mystery Of The Little Engine That Could

How can one build a machine performing 10^9 different operations per second?

The instruction cycle

Turing machine



Instruction cycle:

- Read memory cell
- Change state
- Read instruction
- Change state
- Write memory cell

repeat

<u>Turing – Church thesis</u>:

$f:N \rightarrow N$ computable

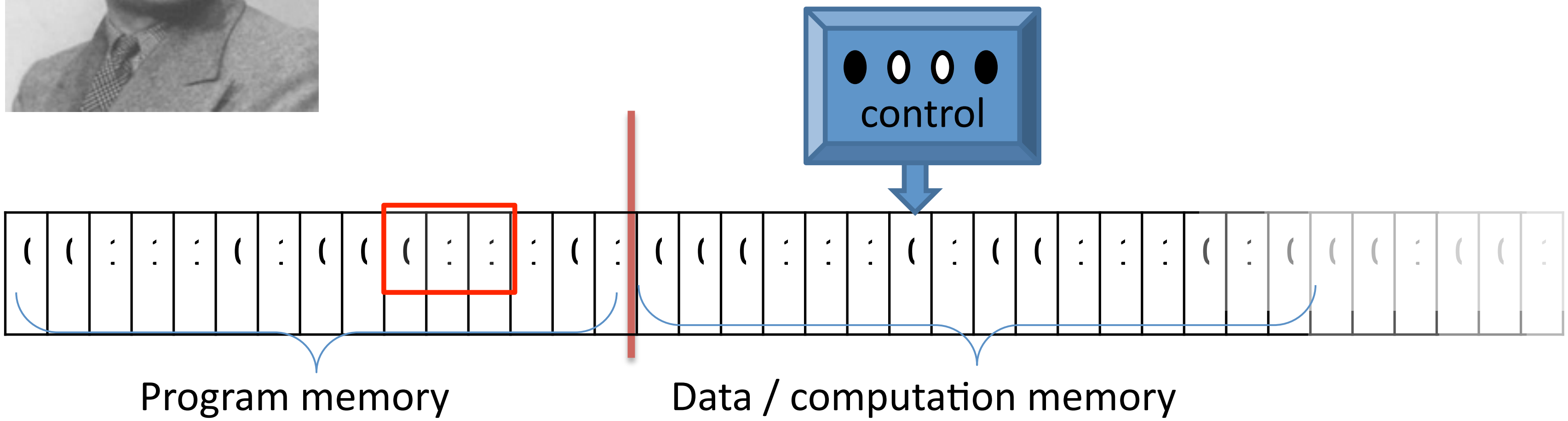
$\Leftrightarrow$

$\exists$ TM computing f.

Elgot Robinson ~1960:

Address register TMs

Turing machine



Instruction cycle:

- **Read memory cell**
- **Change state**
- Read instruction
- Change state
- Write memory cell

repeat

<u>Turing – Church thesis</u>:

$f: N \rightarrow N$ computable

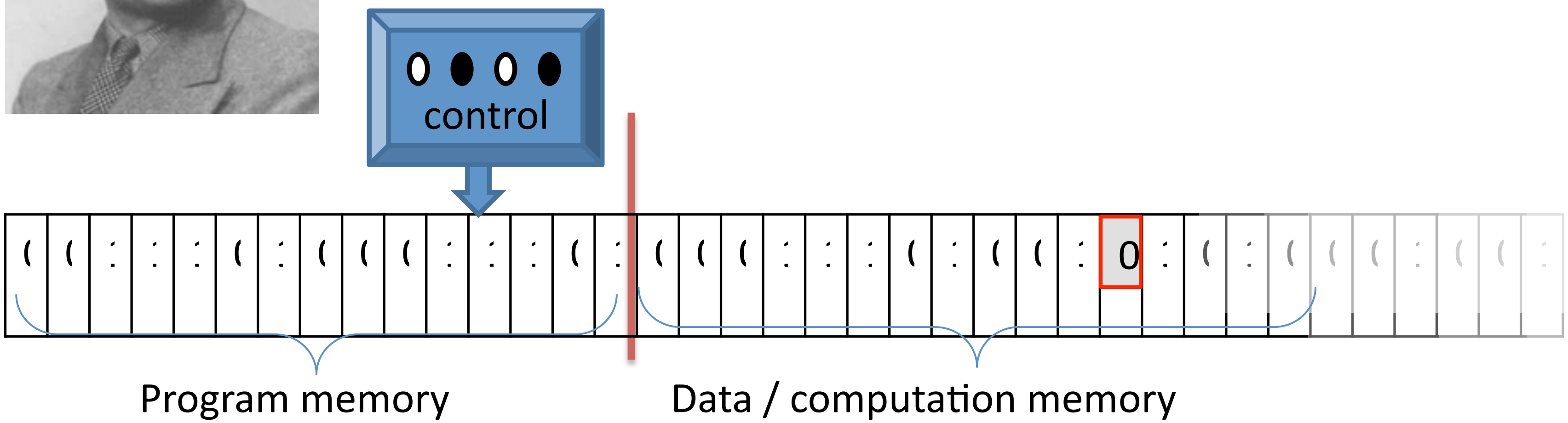
$\Leftrightarrow$

$\exists$ TM computing f.

Elgot Robinson ~1960:

Address register TMs

Turing machine



Instruction cycle:

- **Read memory cell**
- **Change state**
- **Read instruction**
- **Change state**
- Write memory cell

repeat

Turing – Church thesis:

f:N→ N computable

⇔

∃ TM computing f.

Elgot Robinson ~1960:

Address register TMs

Basic features of TMs

1. Memory tape / Alphabet
2. Finite / small control (# of states <100 suffice)
3. Instruction Cycle
4. ***Stored*** program device
5. ***Universal*** computing machine: one machine programmable to compute every computable function

Kolmogorov Complexity and Proofs of Gödel's First and Second Incompleteness Theorems (Chaitin 1971 Kritchman, Raz 2010)

String x = 011011100…10 Length(x) = n

TM fixed Universal Turing Machine

K(x) = length of shortest program P written in 0,1 such that TM programmed by P prints out x.

By counting: Most strings x of length n have

$K(x) \geq n$.

Chaitin's First Incompleteness Theorem. No Liar's Paradox

- Let AX be a rich axiom system, sufficient to express arithmetic and Gödel numbering
- Let M be size of a TM program that recognizes strings which are formal proofs in AX. We may assume M = 9,000.
- ***Theorem***. If AX is consistent then for no string x is the statement $K(x) \geq 10{,}000$ provable in AX.

Computing ➔ New Proof Concepts

- Proof by Randomization
- Non-Transferable Proofs
- Interactive Proofs

Randomized Proofs of Polynomial Identities

$6(x_1^2 + x_2^2 + x_3^2 + x_4^2)^2 = (x_1 + x_2)^4 + (x_1 + x_3)^4 + (x_2 + x_3)^4 + (x_1 + x_4)^4 + (x_2 + x_4)^4 + (x_3 + x_4)^4 + (x_1 - x_2)^4 + (x_1 - x_3)^4 + (x_2 - x_3)^4 + (x_1 - x_4)^4 + (x_2 - x_4)^4 + (x_3 - x_4)^4$

- Teacher: Prove the above identity!
- Naïve Student: Substitute $x_1 = 37$, $x_2 = 9211$, $x_3 = 590$, $x_4 = 103$. Use Notebook Computer:

  7259482876354801 = 7259482876354801

  QED
- Student does not understand example is not a proof!
- Grade: F

Randomized Proof Continued

- Theorem: Let F be a field. $f(t_1, . . ., t_k)$ polynomial of total degree d.
- Let S subset F, card(S) finite.

  If $f \neq 0$, then

  $$\Pr[f(a_1, . . ., a_k) = 0] \leq_R d/\mathrm{card}(S)$$

  where $a_1, . . ., a_k$ <----- S
- Student used S = {1, 2, . . ., 10007}
- $\Pr[f(a) = 0] \leq 16 / 10007 < 0.0016$

Randomized Proof Continued

- Theorem: Let F be a field. $f(t_1, . . ., t_k)$ polynomial of total degree d.
- Let S subset F, card(S) finite.

  If $f \neq 0$, then

  $$\Pr[f(a_1, . . ., a_k) = 0] \leq_R d/\text{card}(S)$$

  where $a_1, . . ., a_k$ <----- S
- Student used S = {1, 2, . . ., 10007}
- $\Pr[f(a) = 0] \leq 16 / 10007 < 0.0016$

Simplified Computation

- Actually, 10007 is prime, so Z mod 10007 is a field of 10007 elements. Theorem hold for Z mod p, p prime.
- Now clever student computes mod 10007, gets same probabilistic proof for the identity, without computing with long integers.
- Method applicable to identities as yet not provable by classical methods. For such identities, only non-transferable proofs.
- Open question.

Back to Mathematical Logic

- Language L $\subseteq$ N has solvable decision problem if f:N→{0,1}, $\forall$n$\in$L f(n)=1 and $\forall$n$\notin$L f(n)=0 is Turing/Church/Gödel computable/solvable/recursive
- Turing: Language HALT (halting problem) is unsolvable

  → word problem for semi-groups, *unsolvable*.
- Turing/Church: Decision problem for First-order logic, *unsolvable*.
- Turing/Church/Gödel: Decision problem for almost any axiomatic theory, *unsolvable*.

From Unsolvability to Complexity

- Turing degrees of unsolvability.
- Reduction: Let $R_1, R_2 \subseteq N$ be Recursively enumerable, unsolvable (non-recursive) sets.
- $R_1 \prec R_2$ if $\exists$ $g: N \rightarrow N$ recursive function s.t.

  $$n \in R_1 \text{ iff } g(n) \in R_2$$

- deg $R_1 <$ deg R_2 if $R_1 \prec R_2$ but $R_2 \not\prec R_1$.
- Friedberg, Mucnik 1957: $\exists$ r.e. R_1, R_2 s.t. deg $R_1 <$ deg R_2

Degrees of Difficulty of Computing a Function (R. 1958)

- Responding to a question by John McCarthy about passwords, R. asked:
  - What does it mean that computable function $g: N \rightarrow \{0,1\}$ is more difficult to compute than computable function $f: N \rightarrow \{0,1\}$?
- *Theorem:*

For every recursive set $R_1 \subseteq N$, $\exists$ recursive set $R_2 \subseteq N$ s.t. decision problem for R_2 absolutely more difficult than decision problem for R_1 .

Complexity of Computations enables Modern cryptography

Complexity of Theorem Proving

<u>Presburger Arithmatic</u>

- Alphabet 0, 1, +, =, ˜, $\wedge$, $\vee$, $\exists$, $\forall$, x, y,…
- Domain N = {0,1,2,…}
- All true sentences:

$\forall x \forall y[x+y=y+x]$, $\forall x \forall y \exists z\, [x+z=y \vee y+z = x]$, etc.

- *<u>Theorem</u>* [Presburger, 1929]: **PA**- The set of all true first-order sentences about addition of natural numbers, is *decidable*.

Presburger Arithmatic is Double Exponentially Hard

Theorem [M. Fischer, R., 1973]

$\exists\, \alpha\ (\geq 0.1)$ such that:

for every decision algorithm *AL* for **PA**,

$\exists\, n_0 = n_0(AL) = O(|AL|),\ \forall\, n > n_0,\ \exists\, S,\ |S| = n,$

$STEPS_{AL}(S) \geq 2^{2^{\alpha n}}$

Theorem. For every axiomatic theory AX for PA

$\exists\, n_0 = n_0(AX),\ \forall\, n > n_0,\ \exists$ true $S,\ |S| = n,$

$LengthShortestProof(S) \geq 2^{2^{\alpha n}}$

Beyond Turing Computability

- R.S. 1957 : Non-Deterministic computation
- Non-Deterministic → Cook, Karp, Levin (1971) P=NP?
- R. 1963, R. 1976, Solovay, Strassen 1977: Randomized Algorithms
- Parallel and Distributed computing
- Computation and Communication networks
- Quantum Computing (?)