Turing machines and computer viruses

Luite van Zelst

Institute for Logic, Language and Computation
University of Amsterdam

3$^{\mathrm{rd}}$ International Workshop on the
Theory of Computer Viruses, 2008

Outline

1. Motivation
2. Turing machines
   - Modern Turing machines
   - Turing's original machine
3. Computer viruses
   - Interpreted Sequences
   - Cohen's viruses

What is a computer virus?

“A virus may be loosely defined as a sequence of symbols which, upon interpretation, causes other sequences of symbols to contain (possibly evolved) virus(es).” (Fred Cohen)

Cohen's viruses
The definition

Let M be a Turing machine and $V \subseteq \Sigma^*$ then $\langle M, V\rangle \in \mathrm{VS}$ if

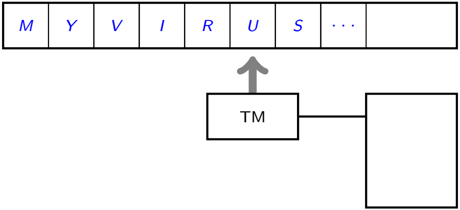
$$
\begin{aligned}
&\forall v \in V,\ h \in \mathbb{H}_M && (1)\\
&\quad \text{if } \exists\ n_1 < \omega && (2)\\
&\qquad \wedge\ h(0) = \langle s_0, \_, \_\rangle && (3)\\
&\qquad \wedge\ h(n_1) = \langle s_0, t_1, p_1\rangle && (4)\\
&\qquad \wedge\ t_1[p_1, |v|] = v && (5)\\
&\quad \text{then } \exists\ v' \in V,\ n_2 < \omega,\ pos < \omega && (6)\\
&\qquad \wedge\ h(n_2) = \langle \_, t_2, \_\rangle && (7)\\
&\qquad \wedge\ t[pos, |v'|] = v' && (8)\\
&\qquad \wedge\quad \vee\ pos \geq p_1 + |v| && (9)\\
&\qquad\qquad \vee\ p_1 \geq pos + |v'| && (10)\\
&\qquad \wedge\ \exists\ n_3 < \omega && (11)\\
&\qquad\qquad \wedge\ n_1 < n_3 < n_2 && (12)\\
&\qquad\qquad \wedge\ h(n_3) = \langle s_3, t_3, p_3\rangle && (13)\\
&\qquad\qquad \wedge\ pos \leq p_2 \leq pos + |v'| && (14)
\end{aligned}
$$

Sequences & machines

$M \quad Y \quad V \quad I \quad R \quad U \quad S \quad \cdots$

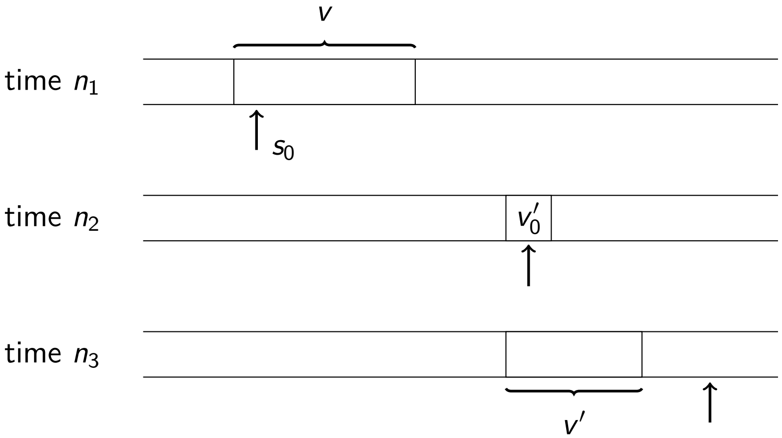
- Contiguous sequences (strings)
- Any substring on the tape
- Uses a special flavour of Turing machines

Sequences & machines



- Contiguous sequences (strings)
- Any substring on the tape
- Uses a special flavour of Turing machines

Cohen's viruses
Depicted



Are Turing machines appropriate?
Literature

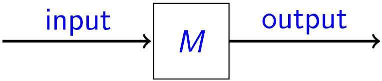
- Thimbleby et al. in 1998: *A Framework for Modelling Trojans and Computer Virus Infection*
- Mäkinen in 2001: *Comment on 'A Framework for Modelling . . . '*

Are Turing machines appropriate?

- How are Turing machines defined *precisely*?
- How are ‘interpreted sequences’ defined?

What is a Turing machine ?

Davis (1958), Minsky (1967), Hopcroft et al. (1979):
Turing machine computes a function:

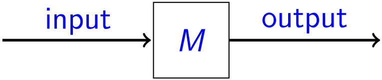


On computable numbers, with an application to the Entscheidungsproblem, Turing, 1936
Machine that computes an infinite sequence

1 1 0 0 1 1 1 1 0 0 0 0 1 1 1 1 ···

What is a Turing machine ?

Davis (1958), Minsky (1967), Hopcroft et al. (1979):
Turing machine computes a function:



On computable numbers, with an application to the Entscheidungsproblem, Turing, 1936
Machine that computes an infinite sequence

Modeling a modern Turing machine
Infinite tape

- Infinite tape: $t : \omega \to \Sigma$
- Finite content: $\Box \notin \Sigma$ represents an empty square
- Infinite tape: $t : \omega \to \Sigma \cup \{\Box\}$
- Pure content: $\mathbb{P}_\Sigma$

$$\overbrace{\begin{array}{|c|c|c|c|c|}\hline 0 & 1 & 0 & 1 & 0 \end{array}}^{\Sigma}\begin{array}{|c|c|c|c|c|}\hline \Box & \Box & \Box & \Box & \cdots \\ \hline \end{array}$$

- $\mathbb{P}_\Sigma$ in one-one correspondence with Σ^*

Modeling a modern Turing machine

Infinite tape

- Infinite tape: $t : \omega \to \Sigma$
- Finite content: $\square \notin \Sigma$ represents an empty square
- Infinite tape: $t : \omega \to \Sigma \cup \{\square\}$
- Pure content: $\mathbb{P}_\Sigma$

$$\overbrace{\boxed{0}\boxed{1}\boxed{0}\boxed{1}\boxed{0}}^{\Sigma}\boxed{\square}\boxed{\square}\boxed{\square}\boxed{\square}\boxed{\cdots}$$

- $\mathbb{P}_\Sigma$ in one-one correspondence with Σ^*

Modeling a modern Turing machine

Infinite tape

- Infinite tape: $t : \omega \to \Sigma$
- Finite content: $\square \notin \Sigma$ represents an empty square
- Infinite tape: $t : \omega \to \Sigma \,\dot\cup\, \{\square\}$
- Pure content: $\mathbb{P}_\Sigma$

$$\overbrace{\begin{array}{|c|c|c|c|c|}\hline 0 & 1 & 0 & 1 & 0 \\ \hline\end{array}}^{\Sigma}\begin{array}{|c|c|c|c|c|}\hline \square & \square & \square & \square & \cdots \\ \hline\end{array}$$

- $\mathbb{P}_\Sigma$ in one-one correspondence with Σ^*

Modeling a modern Turing machine

Infinite tape

- Infinite tape: $t : \omega \to \Sigma$
- Finite content: $\square \notin \Sigma$ represents an empty square
- Infinite tape: $t : \omega \to \Sigma \cup \{\square\}$
- Pure content: $\mathbb{P}_\Sigma$

$$\overbrace{\boxed{0}\boxed{1}\boxed{0}\boxed{1}\boxed{0}}^{\Sigma}\boxed{\square}\boxed{\square}\boxed{\square}\boxed{\square}\boxed{\cdots}$$

- $\mathbb{P}_\Sigma$ in one-one correspondence with Σ^*

Structure $\langle Q, \Sigma, tr, q_0 \rangle$ where

- Q a finite set of states
- Σ a finite set of tape symbols
- q_0 starting state
- tr is a transition function such that

$$tr : Q \times (\Sigma \mathbin{\dot\cup} \{\square\}) \to Q \times \Sigma \times \{-1, 0, 1\}$$

- Configurations: $\langle s, t, p \rangle$ where
  - state: $s \in Q$
  - tape: $t : \omega \to \Sigma \mathbin{\dot\cup} \{\square\}$
  - position: $p < \omega$
- Moves: $\langle s, t, p \rangle \hookrightarrow \langle s', t', p' \rangle$

Modeling a modern Turing machine
Computation

- Computations: $\to_M$ binary relation on infinite tapes $t : \omega \to \Sigma \cup \{\Box\}$

$$t \to_M t' \iff \langle s, t, p \rangle \hookrightarrow^n \langle s', t', p' \rangle \not\hookrightarrow$$

- Condition 1: the machine start with$\langle s_0, t \in \mathbb{P}_B, 0 \rangle$
- Condition 2: The machine does not write $\Box$:
  - $\to_M \subseteq (\mathbb{P}_B)^2$
  - $\to_M \subseteq (\Sigma^*)^2$

- Computations: $\to_M$ binary relation on infinite tapes $t : \omega \to \Sigma \,\dot\cup\, \{\square\}$

$$t \to_M t' \iff \langle s, t, p \rangle \hookrightarrow^n \langle s', t', p' \rangle \not\hookrightarrow$$

- Condition 1: the machine start with $\langle s_0, t \in \mathbb{P}_B, 0 \rangle$
- Condition 2: The machine does not write $\square$:
  - $\to_M \subseteq (\mathbb{P}_B)^2$
  - $\to_M \subseteq (\Sigma^*)^2$

- Computations: $\rightarrow_M$ binary relation on infinite tapes $t : \omega \rightarrow \Sigma \,\dot\cup\, \{\Box\}$

$$t \rightarrow_M t' \iff \langle s, t, p\rangle \hookrightarrow^n \langle s', t', p'\rangle \not\hookrightarrow$$

- Condition 1: the machine start with$\langle s_0, t \in \mathbb{P}_B, 0\rangle$
- Condition 2: The machine does not write $\Box$:
  - $\rightarrow_M \subseteq (\mathbb{P}_B)^2$
  - $\rightarrow_M \subseteq (\Sigma^*)^2$

Modeling a modern Turing machine
Semantics

- Basic: $|M| : \Sigma^* \rightarrow \Sigma^*$
- Function on naturals: encode input and output
- Representing all functions: one extra 'erasure' symbol

Turing's Original Machine
Computable Numbers

In 1936, Turing wrote: *On computable numbers, with an application to the Entscheidungsproblem*.

- Real numbers
  - Binary expansion: $\pi = 11,0010010000111111 01\dots$
  - Non integer part: infinite sequence
- Computable numbers
  - Binary expansion written by a machine ?
- Computable sequences

Turing's Original Machine
Computable Numbers

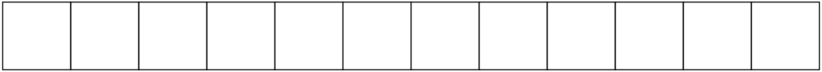
In 1936, Turing wrote: *On computable numbers, with an application to the Entscheidungsproblem*.

- Real numbers
  - Binary expansion: $\pi = 11,001001000011111101\ldots$
  - Non integer part: infinite sequence
- Computable numbers
  - Binary expansion written by a machine ?
- Computable sequences

In 1936, Turing wrote: *On computable numbers, with an application to the Entscheidungsproblem*.

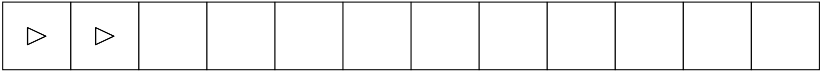
- Real numbers
  - Binary expansion: $\pi = 11,0010010000111111 01\ldots$
  - Non integer part: infinite sequence
- Computable numbers
  - Binary expansion written by a machine ?
- Computable sequences

Turing's original machine
Conditions



- Mark the left-hand side
- Figures: ‘output’
- Auxiliaries: ‘notes’
- *F*-squares
  - A contiguous sequence of figures
  - Not erasable (‘write-once’)
- *E*-squares
  - a kind of scratchpad

Turing's original machine
Conditions



- Mark the left-hand side
- Figures: 'output'
- Auxiliaries: 'notes'
- *F*-squares
  - A contiguous sequence of figures
  - Not erasable ('write-once')
- *E*-squares
  - a kind of scratchpad

Turing's original machine
Conditions

$\triangleright$	$\triangleright$	0		1		0		1		1	

- Mark the left-hand side
- Figures: ‘output’
- Auxiliaries: ‘notes’
- *F*-squares
  - A contiguous sequence of figures
  - Not erasable (‘write-once’)
- *E*-squares
  - a kind of scratchpad

Turing's original machine
Conditions

$\triangleright$	$\triangleright$	0	$*$	1	\$	0	$*$	1	$*$	1	\$

- Mark the left-hand side
- Figures: 'output'
- Auxiliaries: 'notes'
- F-squares
  - A contiguous sequence of figures
  - Not erasable ('write-once')
- E-squares
  - a kind of scratchpad

$\triangleright$	$\triangleright$	0	$*$	1	$	0	$*$	1	$*$	1	$

- Mark the left-hand side
- Figures: 'output'
- Auxiliaries: 'notes'
- *F*-squares
  - A contiguous sequence of figures
  - Not erasable ('write-once')
- *E*-squares
  - a kind of scratchpad

Turing's original machine
Conditions

$\triangleright$	$\triangleright$	0	$*$	1	$	0	$*$	1	$*$	1	$

- Mark the left-hand side
- Figures: ‘output’
- Auxiliaries: ‘notes’
- *F*-squares
  - A contiguous sequence of figures
  - Not erasable (‘write-once’)
- *E*-squares
  - a kind of scratchpad

Turing's original machine
Computed sequence

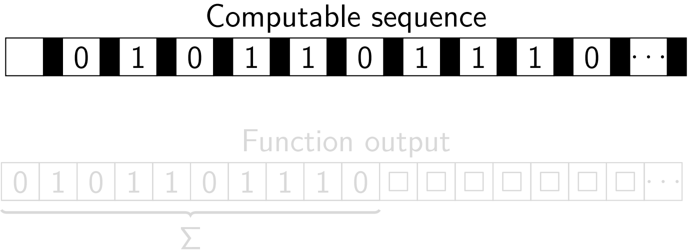
Computable sequence

Function output

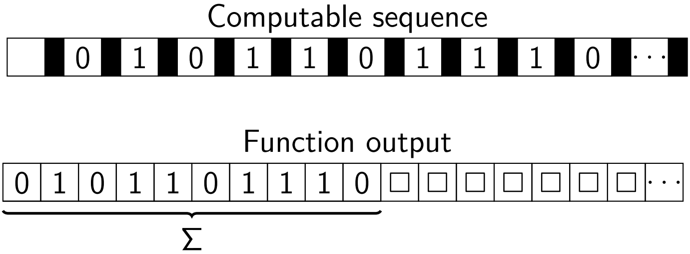
$$\underbrace{0\;1\;0\;1\;1\;0\;1\;1\;1\;0}_{\Sigma}\;\square\;\square\;\square\;\square\;\square\;\square\;\square\;\cdots$$

Turing's original machine

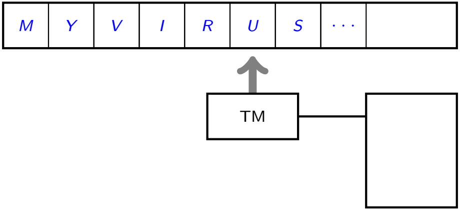
Computed sequence



Turing's original machine
Computed sequence

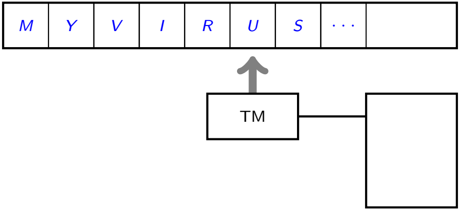


Computer viruses
On Turing machines



Can we define a virus as a (contiguous) 'sequence of symbols' that is 'interpreted' by a Turing machine?

Computer viruses
On Turing machines



Can we define a virus as a (contiguous) 'sequence of symbols' that is 'interpreted' by a Turing machine?

Interpreted Sequences

For modern Turing machines

- Turing machine: computes a function
  - fully determined by transition function
- Universal machine:
  - Computes the *universal function*
  - Computes the function of *some other machine*
- Defining a Universal machine:
  - Encode the transition function: ‘program’
  - Encode the input to this machine: ‘input’
  - Encodings: injective and therefore decodable

Encodings
For modern Turing machines

- Without specifying ‘valid’ encodings
  - Any machine ‘interprets’ any input
  - Empty string ‘encodes’ the machine itself
  - Entire input ‘encodes’ a constant function
- Just one program
- Interleaving of ‘program’, ‘input’ and simulated tape and temporary symbols
- Not every substring of the (total) input is interpreted

Interpreted Sequences

For modern Turing's original machines

- Turing's original machine:
  - no input
  - no interpreted sequences
- Turing's universal machine:
  - itself unlike Turing's original machines
  - the entire input is the encoding of exactly one machine
  - F- and E-squares: program is not a contiguous sequence on the tape

Interpreted Sequences

For modern Turing's original machines

- Turing's original machine:
  - no input
  - no interpreted sequences
- Turing's universal machine:
  - itself unlike Turing's original machines
  - the entire input is the encoding of exactly one machine
  - F- and E-squares: program is not a contiguous sequence on the tape

Viruses for Turing machines

- Model at least two programs
- Non-program cannot be a virus
- Standard models inadequate
  New (universal) Turing machine?
  - No benefit: non standard

Viruses for Turing machines

- Model at least two programs
- Non-program cannot be a virus
- Standard models inadequate
  New (universal) Turing machine?
    - No benefit: non standard

Cohen's Turing machine

- tape: $\omega \longrightarrow \Sigma$
  - infinite tape with infinite content
  - compare with $t : \omega \longrightarrow \Sigma \dot\cup \{\Box\}$
- starting state & position undefined
- transition function unrestricted
  - $tr : K \times \Sigma \;\; \longrightarrow \;\; K \times \Sigma \times \{-1, 0, 1\}$
  - even with $\Box \in \Sigma$ finite content undecidable

Cohen's Turing machine

- tape: $\omega \longrightarrow \Sigma$
  - infinite tape with infinite content
  - compare with $t : \omega \longrightarrow \Sigma \,\dot{\cup}\, \{\square\}$
- starting state & position undefined
- transition function unrestricted
  - $tr : K \times \Sigma \;\longrightarrow\; K \times \Sigma \times \{-1, 0, 1\}$
  - even with $\square \in \Sigma$ finite content undecidable

Cohen's Turing machine

- tape: $\omega \longrightarrow \Sigma$
  - infinite tape with infinite content
  - compare with $t : \omega \longrightarrow \Sigma \,\dot{\cup}\, \{\Box\}$
- starting state & position undefined
- transition function unrestricted
  - $tr : K \times \Sigma \ \longrightarrow \ K \times \Sigma \times \{-1, 0, 1\}$
  - even with $\Box \in \Sigma$ finite content undecidable

What is Cohen's machine?

- Not a modern Turing machine
- Not Turing's original (universal) machine
  - All sequences trivially computable
  - No distinction between figures and auxiliaries
  - No distinction between F-squares and E-squares

Insurmountable difference

Can the difference between Cohen's machine and modern Turing machines be overcome?

- Viral equivalence: $M \equiv_{vir} N$ if: $\langle M, V \rangle \in \mathrm{VS} \iff \langle N, V \rangle \in \mathrm{VS}$
- Viral equivalence is incomparable to functional equivalence

No, it cannot.

Summary

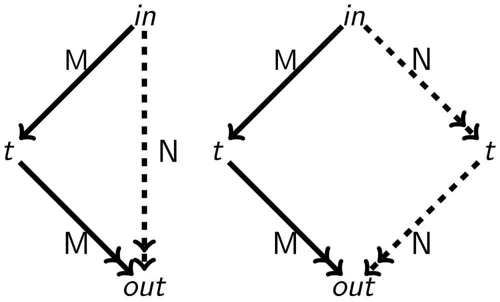
- Precise definitions essential for Turing machines
- Turing machines are inappropriate to model viruses
- Cohen's modelling non-standard

- Outlook
  - Open door for other modellings of computer viruses
  - Dissect Turing's machine and unify the two Turing machine models: 'Talkative Machine' (TM)

The end

Thank you for your attention!

Insurmountable difference

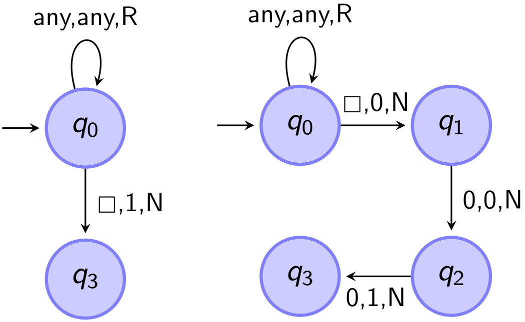


$\equiv_{vir} \not\subseteq \equiv_{\mathrm{FUNC}}$

$$M = \langle Q, \Sigma, tr, q_0 \rangle \qquad N = \langle Q, \Sigma, tr', q_0 \rangle$$
$$\Sigma = \{a, b\} \qquad Q = \{q_0, q_1\}$$
$$tr(q_0, \_) = \langle q_1, a, 0 \rangle \qquad tr'(q_0, \_) = \langle q_1, b, 0 \rangle$$

- Virally equivalent (trivially)
- No functionally equivalent:
  - $|M|_{func} = t \mapsto t[0 \mapsto a]$
  - $|M|_{func} = t \mapsto t[0 \mapsto b]$

Two machines that compute $x \mapsto x \cdot 1$.



For Further Reading I

beamericonbook M. Davis.
Computability and unsolvability.
McGraw-Hill, 1958.

beamericonbook M. Minksy.
Computation : finite and infinite machines.
Prentice-Hall, 1967.

beamericonbook J. Hopcroft and J. Ullman.
Introduction to Automata Theory, Languages, and Computation.
Addison Wesley, 1979.

For Further Reading II

beamericonbook B.J. Copeland.
The essential Turing: seminal writings in computing, logic, philosophy, artificial intelligence, and artificial life plus the secrets of enigma.
Oxford University Press, 2004.

beamericonarticle H. Thimbleby, S Anderson and P. Cairns
A Framework for Modelling Trojans and Computer Virus Infection.
Computer Journal, 41: 444-458, 1998.

For Further Reading III

beamericonarticle E. Mäkinen
Comment on 'A framework for modelling trojans and computer virus infection'.
Computer Journal, 44(4): 321-323, 2001.