Turing Incomputable Computation

Turing Centenary Conference
CIE 2012, University of Cambridge

Michael Stephen Fiske
June 20, 2012

Cybersecurity Motivation

Current methods of malware detection are inadequate.

Malware (virus) detection is Turing undecidable. [6]

Malware is using NP hard methods to encrypt and hide. [17]

Alternative Approach.

1. Hide the computational steps of the program computation.
2. Make it more difficult to hijack program execution.

[6] Fred Cohen. Computer Viruses Theory and Experiments. Computers and Security. 6(1), 22–35. February (1987).

[17]. Eric Filiol. Malicious Cryptology and Mathematics. Cryptography and Security in Computing. Intech. 23–50 (2012).

New machine called the Active Element Machine (AEM).

Procedure 2: Using quantum randomness, an AEM program executes a Universal Turing Machine (UTM) program η with active element firing patterns that are Turing incomputable.

Table 1: Boolean Universal Turing Machine program $\eta = (\eta_0\eta_1\eta_2,\ \eta_3\eta_4,\ \eta_5)$

	10	**00**	**01**	**11**
001	**(001, 00, 0)**	**(001, 00, 0)**	**(010, 01, 0)**	**(001, 01, 0)**
010	**(001, 00, 0)**	**(010, 10, 1)**	**(010, 11, 1)**	**(110, 10, 1)**
011	**(011, 10, 0)**	**(000, 00, h)**	**(011, 11, 0)**	**(100, 01, 0)**
100	**(100, 10, 0)**	**(101, 10, 1)**	**(111, 01, 0)**	**(100, 01, 0)**
101	**(101, 10, 1)**	**(011, 10, 0)**	**(101, 11, 1)**	**(101, 01, 1)**
110	**(110, 10, 1)**	**(011, 11, 0)**	**(110, 11, 1)**	**(110, 01, 1)**
111	**(111, 00, 1)**	**(110, 10, 1)**	**(111, 01, 1)**	**(010, 00, 1)**

States $Q = \{001, 010, 011, 100, 101, 110, 111\}$.
Alphabet $\mathcal{A} = \{10, 00, 01, 11\}$.

Quantum Randomness Axioms

1. *No bias.*

A single outcome x_i of a bit sequence (x_1 x_2 . . .) generated by quantum randomness is unbiased.

$P(x_i = 1) = P(x_i = 0) = 1 / 2$.

2. *History has no effect on the next event.*

Each outcome x_i is independent of the history.

No correlation exists between previous or future outcomes.

$P(x_i = 1 \mid x_1 = b_1 \ \ldots \ x_{i-1} = b_{i-1}) = P(x_i = 0 \mid x_1 = b_1 \ \ldots \ x_{i-1} = b_{i-1}) = 1/2$ for each $b_k \in \{0, 1\}$.

Definition 1. Let $g : \mathrm{N} \to \{0, 1\}$, $f : \mathrm{N} \to \{0, 1, \ldots m\}$ be functions. A Turing machine maps g back to f if conditions A and B hold.

A) The initial Turing machine tape has $g(k)$ stored on tape square k for each k.

B) The Turing machine begins execution at tape square 1 and after a finite number of steps writes $f(1)$ on tape square 1. For each k, the Turing machine visits tape square k and after a finite number of steps, it writes $f(k)$ on tape square k and then moves to tape square $k + 1$.

Incomputable Firing Patterns

Unbounded number of computable UTM steps.

Intuitive Definition: Just the UTM steps if the UTM doesn't halt.

If the UTM halts, then the UTM needs to have its tape reinitialized after each halt in a computable way. The steps are concatenated as infinite sequence.

For each UTM computational step, there are 93 active elements that collectively compute the UTM program $\eta = (\eta_0\eta_1\eta_2, \eta_3\eta_4, \eta_5)$.

On the kth UTM computational step, whether each of these 93 elements fires or does not fire is denoted as $(f_{1,k} \ldots f_{93,k})$.
Define g: N $\rightarrow$ {0, 1} as $g = (f_{1,1} \ldots f_{93,1} \ldots f_{1,k} \ldots f_{93,k} \ldots)$.

Theorem 4.2. If an unbounded number of computable UTM steps are executed by the AEM according to Procedure 2 and the two quantum randomness axioms hold, then g is an incomputable function.

One-way Computation: Anti-Panopticon

Theorem 4.3. If an adversary can only eavesdrop on the firing activity of $g = (f_{1,1}\ , f_{2,1}\ \ldots f_{93,1}\ , \ldots\ , f_{1,j}\ \ldots f_{93,j}\ \ldots)$ then the AEM execution, described in Procedure 2, of the UTM is perfectly secret.

In other words, $P(q = q_k) = P(q = q_k \mid f_{1,j} = b_1\ \ldots\ f_{m,j} = b_m)$ and
$P(T_k = a_k) = P(T_k = a_k \mid f_{1,j} = b_1\ \ldots\ f_{m,j} = b_m)$ for each $b_i \in \{0, 1\}$
where q is the current state of the UTM and
T_k is the contents of the kth tape square.

Corollary 1. Consider an unbounded number of computable steps generated by the AEM in procedure 2, where the sequence of UTM instructions is $I_1\ , I_2\ \ldots\ I_k\ \ldots$ and $I_k \in Q \times A$. Define $f : \mathrm{N} \rightarrow Q \times A$ as $f(k) = I_k$. If an adversary can only eavesdrop on g (the AEM firing activity computing η), then no Turing machine exists that can map g back to f.

Active Element Machine computation

1. **Active Elements and Connections**.

   All elements compute simultaneously.

2. **Connections connect Elements**.

   Connections specify the pulses that are sent between elements.

3. **Elements fire and send pulses along Connections**.

   An element **E** fires at time **t** if the sum of **E**'s input pulses is greater than **E**'s threshold θ and **E**'s refractory period **r** has expired i.e. $\mathbf{t} \geq \mathbf{r} + \mathbf{s}$ where **s** is **E**'s most recent firing time.

   If element **E** fires at time **s** and a connection from **E** to **B** exists, then a pulse reaches element **B** at time $\mathbf{s} + \tau_{EB}$ where τ_{EB} is the transmission time from **E** to **B**.

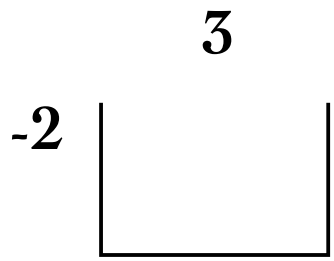
AEM computation - Outgoing Pulses

4. **(Connection (Time 4) (From E) (To Y) (Amp -2) (Width 3) (Delay 5)**
   (Connection (Time 4) (From E) (To Z) (Amp 4) (Width 2) (Delay 3)

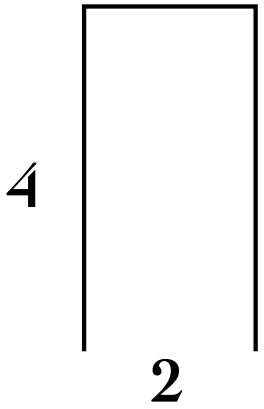
If element **E** fires at time 4, then:

A pulse of time width 3 and amplitude -2 (height is - 2) arrives at element **Y** at time 9.

A pulse of time width 2 and amplitude (height 4) arrives at element **Z** at time 7.



is sent to **Y**



is sent to **Z**

Element and Connection commands

Extended integers {$m + k$ dT: m, k are integers & dT is a fixed infinitesimal}.

Element command parameters:

(Element (Time 3-2dT) (Name E) (Threshold 9) (Refractory 4) (Last 0))

- **(Refractory 4)** The refractory period is a positive integer.
- **(Threshold 9)** The threshold is an integer.
- **(Last 0)** The most recent firing time is an integer.
- **(Time 0)** Time is an extended integer ***m* + *k* dT** i.e. 3-2dT

Connection command parameters:

(Connection (Time -1) (From A) (To E) (Amp -7) (Width 4+2dT) (Delay 1-dT)

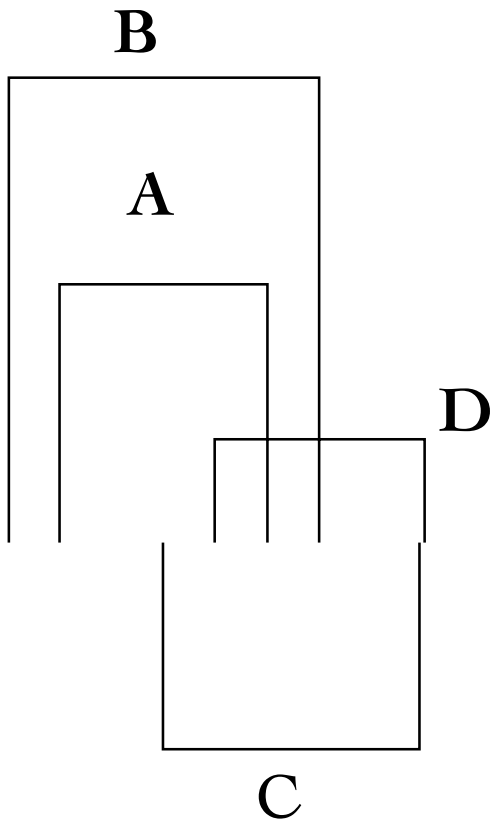
- **(Amp -7)** Amplitude is an integer.
- **(Width 4+2dT)** Pulse width is ***m* + *k* dT** where the standard part ***m*** ≥ 1.
- **(Delay 1-dT)** Transmission time is ***m* + *k* dT** where ***m*** ≥ 1.

AEM computation - Input Pulses

(Connection (Time -1) (From A) (To E) (Amp 5) (Width 4+2dT) (Delay 3-dT)
(Connection (Time -1) (From B) (To E) (Amp 9) (Width 6) (Delay 2+dT)
(Connection (Time -1) (From C) (To E) (Amp -4) (Width 5) (Delay 5-dT)
(Connection (Time -1) (From D) (To E) (Amp 2) (Width 4+2dT) (Delay 6-dT)

(Fire (Time 0) (Name A))
(Fire (Time 0) (Name B))
(Fire (Time 0) (Name C))
(Fire (Time 0) (Name D))

Time	Sum of Input to E
2	0
3	14
5	10
10	2

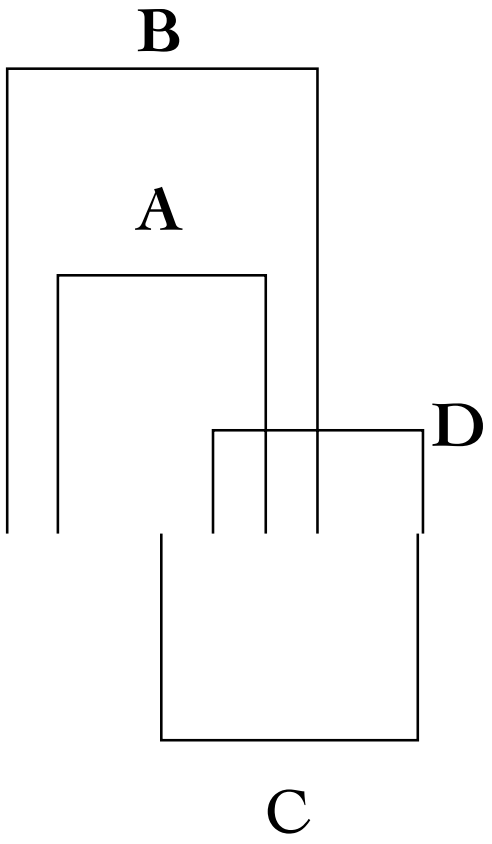


AEM computation - Element E Fires

(Pulse (Name A) (Window 3-dT 7+dT) (Amp 5))
(Pulse (Name B) (Window 2+dT 8+dT) (Amp 9))
(Pulse (Name C) (Window 5-dT 10-dT) (Amp -4))
(Pulse (Name D) (Window 6-dT 10+dT) (Amp 2))

(Element (Time 0) (Name E) (Threshold 9) (Refractory 4) (Last 0))

Element E fires at time 4.



Time	Sum of Input to E
4	14
8	7
9	-2
12	0

Fire command and Keyword dT

(Element (Time 0) (Name E) (Threshold 9) (Refractory 4) (Last 0))
(Connection (Time -1)(From A) (To E) (Amp 5) (Width 4+2dT) (Delay 3-dT))

A is an input element**:** **(Fire (Time 0) (Name A))**

Keyword dT

- **dT** is an infinitesimal amount of time. **dT** > 0 and **dT** is less than every positive rational. If $m < n$, then $m\mathbf{dT} < n\mathbf{dT}$.

- **dT** helps with concurrency. $3\text{-}2\mathbf{dT} < 3\text{-}\mathbf{dT} < 3 < 3+\mathbf{dT} < 3+2\mathbf{dT}$
  For every integer $m > 0$, $2 < 3\text{-}m\mathbf{dT}$ and $3+m\mathbf{dT} < 4$.

- $\mathit{st}(k + m\mathbf{dT}) = k$ is called the standard part of extended integer $k + m\mathbf{dT}$.

Keyword Clock and the Meta Command

Keyword clock

clock evaluates to an integer which is the standard part of the current time of the active element machine clock.

Meta command

Enables the AEM to change itself during program execution.

(**Meta** (**Name E**) (**Window b e**) (**C** $x_1 \ldots x_n$))

If element **E** fires at time **s** in window [**b**, **e**] where **b ≤ s ≤ e**, then command (**C** $x_1 \ldots x_n$) executes at time **s**.

If no window is specified and **E** fires at time **s**, then (**C** $x_1 \ldots x_n$) executes at time **s**. (No window restrictions on time **s**.)

1. Universal Turing Machine Computational Step

Step $\eta(101, 00)=(011, 10, 0)$. $\eta_3(101, 00) = 1$.

Table 1: Boolean Universal Turing Machine program $\eta = (\eta_0\eta_1\eta_2, \eta_3\eta_4, \eta_5)$

	10	00	01	11
001	(001, 00, 0)	(001, 00, 0)	(010, 01, 0)	(001, 01, 0)
010	(001, 00, 0)	(010, 10, 1)	(010, 11, 1)	(110, 10, 1)
011	(011, 10, 0)	(000, 00, h)	(011, 11, 0)	(100, 01, 0)
100	(100, 10, 0)	(101, 10, 1)	(111, 01, 0)	(100, 01, 0)
101	(101, 10, 1)	(011, 10, 0)	(101, 11, 1)	(101, 01, 1)
110	(110, 10, 1)	(011, 11, 0)	(110, 11, 1)	(110, 01, 1)
111	(111, 00, 1)	(110, 10, 1)	(111, 01, 1)	(010, 00, 1)

States $Q = \{001, 010, 011, 100, 101, 110, 111\}$.
Alphabet $\mathcal{A} = \{10, 00, 01, 11\}$.

2. AEM computation of the UTM based on level sets of η

η_3^{-1} (UWX, YZ){1} = {(111, 00), (110, 00), (110, 01), (110, 10), (101, 00), (101, 01), (101, 10), (100, 00), (100, 10),(011, 01), (011, 10), (010, 11), (010,01), (010, 00)}

3. Dynamic Level Sets Based on Quantum Randomness

Quantum random input to elements $R_0 \ldots R_{13}$ helps determine how the AEM uses level sets to compute the next UTM step.

AEM Separation Rules for Level Set ${\eta_3}^{-1}\{1\}$

Firing Pattern	Element	(U, D_i)	(W, D_i)	(X, D_i)	(Y, D_i)	(Z, D_i)	θ_{D_i}
111 00	D_0	2	2	2	−2	−2	5
110 00	D_1	2	2	−2	−2	−2	3
110 01	D_2	2	2	−2	−2	2	5
110 10	D_3	2	2	−2	2	−2	5
101 00	$\mathbf{D_4}$	**2**	**−2**	**2**	**−2**	**−2**	**3**
101 01	D_5	2	−2	2	−2	2	5
101 10	D_6	2	−2	2	2	−2	5
100 00	D_7	2	−2	−2	−2	−2	1
100 10	D_8	2	−2	−2	2	−2	3
011 01	D_9	−2	2	2	−2	2	5
011 10	D_{10}	−2	2	2	2	−2	5
010 00	D_{11}	−2	2	−2	−2	−2	1
010 01	D_{12}	−2	2	−2	−2	2	3
010 11	D_{13}	−2	2	−2	2	2	5

The firing pattern of active elements U, W, X, Y and Z represents the input to η_3

4. D_4 dynamically separates the firing pattern 101 00 with respect to the other firing patterns for UWX YZ.

Input element R_4 fires iff its current quantum random bit = 1.

If R_4 fires, then element D_4 fires when the firing pattern for UWX YZ is **101 00.** For all other firing patterns for UWX YZ, then D_4 doesn't fire**.**

If R_4 doesn't fire, then D_4 doesn't fire when the firing pattern for UWX YZ is 101 00. For all other firing patterns for UWX YZ, then D_4 fires.

Procedure 2 - The Meta Command

5. How? The Meta command dynamically changes the connections parameters from elements U, W, X, Y and Z to element D_4 based on whether R_4 fires or doesn't fire.

```
(Program  set_dynamic_C (Args s  t  f  xk  a  w  tau  rk)
   (Connection (Time s-dT) (From f) (To xk) (Amp -a) (Width w) (Delay tau))

  (Meta (Name rk) (Window s t)
    (Connection (Time t) (From f) (To xk) (Amp a) (Width w) (Delay tau))) )

(Program set_dynamic_E (Args  s  t  xk  h  r  L  rk)
    (Element (Time s-2dT) (Name xk) (Threshold -h) (Refractory r) (Last L))

    (Meta (Name rk) (Window s t)
          (Element (Time t) (Name xk) (Threshold h) (Refractory r) (Last L))) )
```

```
(set_dynamic_E (Args s t  D4 3 1 s-2 R4))      (set_dynamic_C  (Args s t U D4 2 1 1 R4))
(set_dynamic_C (Args s t W D4 -2 1 1 R4))      (set_dynamic_C  (Args s t X D4 2 1 1 R4))
(set_dynamic_C (Args s t  Y D4 -2 1 1 R4))     (set_dynamic_C  (Args s t Z D4 -2 1 1 R4))
```

Firing Representations and Interpretations

6. **Firing Representation.** Consider element $\mathbf{E_i}$'s firing times in the interval of time $\mathbf{W} = [\mathbf{a}, \mathbf{b}]$. Let $\mathbf{s_1}$ be the earliest firing time of $\mathbf{E_i}$ and $\mathbf{s_n}$ the latest firing time both lying in $\mathbf{W}$.

$\mathbf{F(E_i, W)} = [\mathbf{s_1}, \ldots, \mathbf{s_n}] = \{\mathbf{s} \in \mathbf{W}: \mathbf{E_i}$ fired at time $\mathbf{s}\}$ is called a firing sequence of the active element $\mathbf{E_i}$ over the window of time $\mathbf{W}$.

The tuple $(\mathbf{F(E_1, W)}, \ldots, \mathbf{F(E_n, W)})$ is called a *firing representation* of the elements $\mathbf{E_1} \ldots \mathbf{E_n}$ within window $\mathbf{W}$.

An AEM is an *interpretation* between two sequences of firing representations if the machine computes the output sequence from the input sequence.

Future Areas of Research

1. **Building *native* AEM algorithms.**

Understanding AEM procedures with quantum randomness and without. Gurevich's acyclic postulate doesn't appear to hold.

2. **Dynamic Firing Interpretations.**

What else can you do with elaborate types of dynamic level sets? Non-boolean level sets?

3. **Cybersecurity.**

Designing AEM computation that is robust to sabotage.

4. **Foundations of math using infinite AEMs.**