Trial and Error-Based Necessary Conditions for General Intelligence

Francisco J. Arjonilla

Yuichi Kobayashi

Graduate School of Science and Technology (Shizuoka University)

We show that self-referential Turing Machines (TM) are restricted to a subset of all possible transition functions and therefore self-learning is restricted. We propose a work-around based on random trial-and-error.

ESS 2018/09/09-11

1. Background

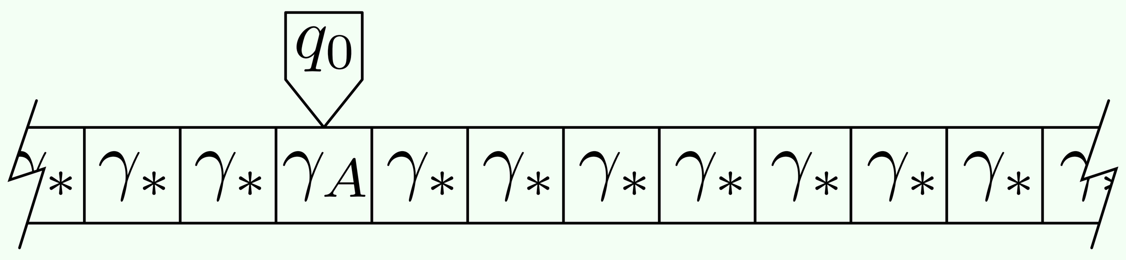
Turing Machines (TM) are the standard framework to study computation theory. All computer functions have an equivalent TM.

A Turing Machine is defined by:

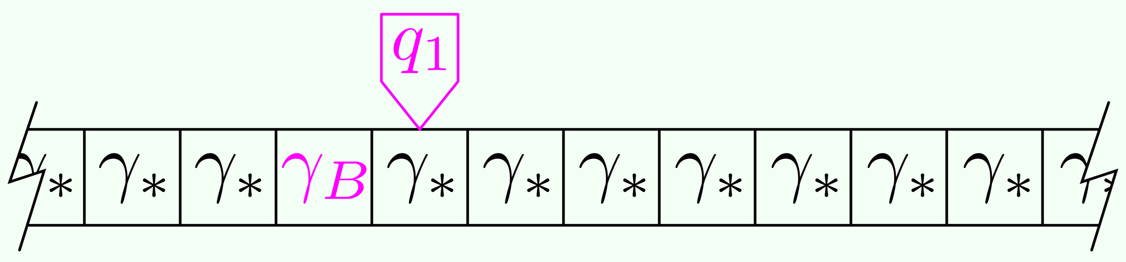
- A finite set Q of machine states. $q_0 \in Q$ is the initial state.
- A finite set Γ of tape alphabet symbols. $b \in \Gamma$ is the blank symbol and $\Sigma \subseteq \Gamma \setminus \{b\}$ the input symbols.
- We do not consider final states.
- A tape, which consists of a 2-sided infinite sequence of blank symbols with a finite number of non-blank symbols for input.
- The TM reads and writes one symbol $\gamma_* \in \Gamma$ at a time, changes its internal state $q_* \in Q$, and then moves either left (L) or right (R) as instructed by its transition function δ: ($*$ denotes "any")

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$$

E.g.:



After execution of $\delta(q_0, \gamma_A) = (q_1, \gamma_B, R)$:



2. Setting

Self-learning is a requisite for General Intelligence [1] and defined here as the capability to modify itself in order to arrive at the appropriate transition function for the desired goal. Let us represent the TM in the tape (i.e. a self-referencing Universal Turing Machine) by encoding δ with the alphabet Γ: $\delta = decode(\tau_1, \ldots, \tau_n)$ where $n \in \mathbb{N}$.
Assume that the header is in position $j : 1 \leq j \leq n$. Then,

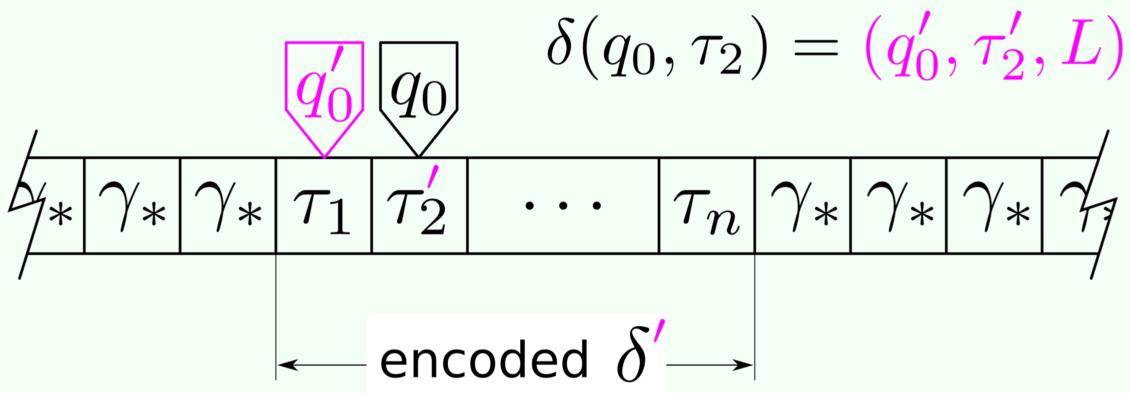
$$\delta(q_*, \tau_j) = decode(\tau_1, \ldots, \tau_j, \ldots, \tau_n)(q_*, \tau_j) = (q_*, \tau_j', *)$$

yields $\tau_1, \ldots, \tau_j', \ldots, \tau_n$ which decodes to $\delta' \neq \delta$.
We denote it as:

$$\delta(\delta) = \delta'$$

such that we ignore what happens in other regions on the tape by assuming $\delta(\delta) = id(\delta) = \delta$ when the header is outside $\tau_1, \ldots, \tau_n$.

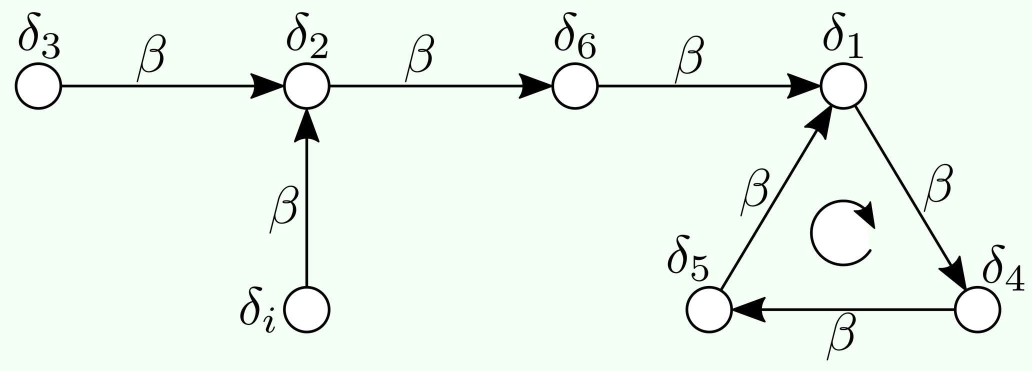
E.g.: $\delta(q_0, \tau_2) = (q_0', \tau_2', L)$



3. Development

Let $\Delta := \{\delta_i | i = 1, 2, \ldots, |\Gamma|^n\}$ be the set of all functions that $\tau_1, \ldots, \tau_n$ can encode.
Then, $\delta_i : \Delta \to \Delta$ and we define a monoid action $\alpha : \Delta \times \Delta \to \Delta$ (i.e. $\alpha(\delta_A, \delta_B) = \delta_A(\delta_b)$) with the peculiarity that $\alpha(\delta_A, \delta_B)$ only happens when $\delta_A = \delta_B$.
Then, we define $\beta(\delta_i) := \alpha(\delta_i, \delta_i)$ and observe that $\beta \circ \beta \circ \cdots \circ \beta(\delta_i)$ produces a path for each δ_i that inevitably ends up in a cycle:

E.g.:

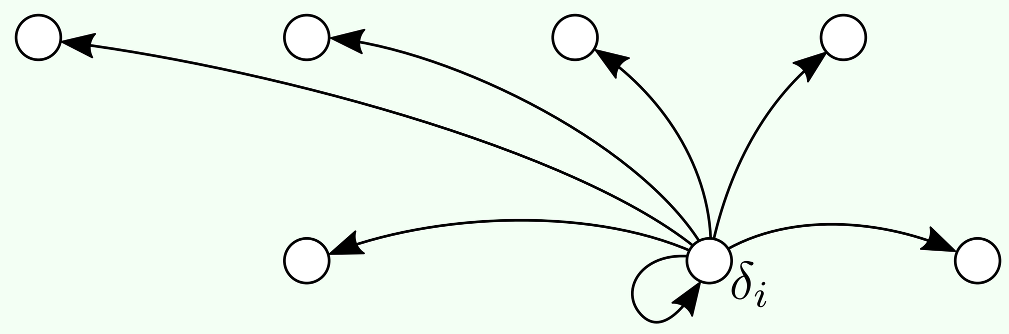


As observed in the previous quiver, there is no path that connects a δ_i with all the other δs.
Consequently, any TM-equivalent machine, including computers, have limited self-learning capabilities. Therefore, any program written with deterministic functions lack the capability of General Intelligence.

4. Proposal

In order to escape cycles, we propose using transition relations instead of transition functions in the region $\tau_1, \ldots, \tau_n$ which enable the TM to randomly / stochastically travel through all $\delta_i \in \Delta$,
effectively transforming the TM to a non-deterministic / probabilistic TM.

E.g.:



This behavior is equivalent to adding an instruction to non-self-referential deterministic TMs that randomizes the transition function, as in:

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, \text{"}randomize \ \ \delta\text{"}\}$$

Now, all $\delta_i \in \Delta$ are reachable at the expense of not always arriving at the correct δ. Hence, the transition function must undergo a process of trial and error [2] until it reaches the desired δ .
Remark: Biological evolution, which is increasingly thought of as an intelligent agent [3], does indeed show stochastic modifications to DNA.

5. Conclusions

- A necessary condition for General Intelligence in computation is that CPUs require true random number generators.
- Mistakes by trial and error are unavoidable in general self-learning.

6. Future Works

- Explore the potential of self-writing proggrams by genetic algorithms.
- Initiate a scientific debate on whether Evolution is actually intelligent.
- Explore other necessary or sufficient conditions for General Intelligence.

7. References

[1] Laird J, Wray R (2010) Cognitive architecture requirements for achieving AGI. *Proc. of the Third Conference on General Intelligence*. pp: 79-84
[2] Arjonilla, F. J. (2015). *A three-component cognitive theory.* MSc. Thesis, Universiteit Utrecht.
[3] Watson R., Szathmary E. (2016) *How can evolution learn?* Trends in Ecology and Evolution. vol:31 (2) pp: 147-157