Embedding Graphs in the Cubic Lattice and Another Construction of a Universal Turing Machine

Min Hyuk "Daniel" Jang

December 18, 2025

Contents

1 Introduction

The Turing machine, invented by Alan Turing in 1936, was crucial in igniting the field of theoretical computer science. Perhaps the most important reason for this was because certain Turing machines can simulate all Turing machines. We call such Turing machines universal Turing machines. Universal Turing machines were the crux of Turing's 1936 proof that there was no answer to David Hilbert and William Ackermann's *Entscheidungsproblem*.

There are many ways one may construct a universal Turing machine. This thesis proposes another one, primarily based on the natural way a reader of a text about Turing machines may reason about them.

2 Prerequisites

We begin with some prerequisites. Much of the following exposition is based on [1].

2.1 Turing Machines

Turing machines are mathematically a jumble of components.

Definition 1 (Turing machines)**.** *A Turing machine* $M := (q_0, Q, H, \Sigma, \#, \Gamma, \delta)$ *is a 7-tuple where, with* $Q' := Q - H$*,*

$$\begin{aligned} & q_0 \in Q \supseteq H \\ \textit{and } & \Sigma \sqcup \{\#\} \subseteq \Gamma \\ \textit{and } & \delta : Q' \times \Gamma \to Q \times \Gamma \times \{\mathsf{L}, \mathsf{S}, \mathsf{R}\}. \end{aligned}$$

Given an input $w := w_1 w_2 \ldots \in \Sigma^*$, a Turing machine will *run*, and then perhaps *halt*. (Note that a Turing machine may not halt, in which case it keeps running.) While running, a Turing machine will be in exactly one state at any given time; Q is the set of such states. q_0 is the starting state.

A Turing machine has a *tape*. The tape is (countably-)infinitely long and has *boxes* in which symbols from Γ can be written. Initially, the infinitely long tape is filled from the left with (countably-)infinitely many #s, then the contents of the input $w := w_1 w_2 \ldots \in \Sigma^* \subsetneq \Gamma$, and the another (countably-)infinitely long sequence of #s.

A Turing machine also has a *head*. The head is a pointer which points to a certain box on the tape (which will contain a symbol from Γ). Initially, the head points to the box of the tape containing the first symbol w_1 of the input $w := w_1 w_2 \ldots$.

Another way of conceptualizing the tape and head is by a function $t : \mathbb{Z} \to \Gamma$ and an integer $h \in \mathbb{Z}$. Initially, given input $w := w_1 w_2 \ldots \in \Sigma^*$,

$$\begin{aligned} t : \mathbb{Z} &\to \Gamma \\ \{\ldots, -1, 0\} &\to \{\#\} \\ \forall i \in \{1, \ldots, |w|\}, i &\mapsto w_i \\ \{|w| + 1, |w| + 2, \ldots\} &\to \{\#\} \end{aligned}$$

and $h := 0$ such that $t(h) = w_1$.

A Turing machine may

1. Change states,
2. Change the symbol $t(h)$ in the box at the current head position h, and
3. Move its head left (L) or right (R) or not (S)

as it runs, depending on

1. The current state, and
2. The symbol in the box at the current head position.

We call such changes *transitions*. The *transition function* $\delta : Q' \times \Gamma \to Q \times \Gamma \times \{\mathsf{L}, \mathsf{S}, \mathsf{R}\}$ is the description of all possible transitions.

We can represent δ as a directed, edge-labeled (or edge-"colored") graph $G_\delta := (V, \{E_{\gamma_1,\gamma_2,m}\}_{\gamma_1,\gamma_2,m})$[1] where

$$V := Q$$
$$E_{\gamma_1,\gamma_2,m} := \{(p, q) \in V^2 \mid \delta(p, \gamma_1) = (q, \gamma_2, m)\}.$$

The edges here are labelled (or "colored") with the necessary information to edit the tape and move the head, while the edge itself describes the change of state.

$H \subseteq Q$ is the set of halting states. A Turing machine halts if (and only if) it transitions into some halting state $\in H$. We may as well call $Q' := Q - H$ the set of "running states."

The *output* of a Turing machine is the finite string of symbols of the tape after it has halted, stripped of the (countably-)infinite string of #s on either side. Note that by definition, there is no output if the Turing machine doesn't halt. The fact that the Turing machine indeed halted implies that its output will be finitely long.

2.1.1 Multi-tape Turing Machines

A Turing machine may have multiple tapes and multiple accompanying heads. Each head can move separately along its respective tape.

Definition 2 (Multi-tape Turing machines)**.** *An n-tape Turing machine $M := (q_0, Q, H, \Sigma, \#, \Gamma, \delta)$ is a 7-tuple where, with $Q' := Q - H$,*

$$q_0 \in Q \supseteq H$$
$$\textit{and } \Sigma \sqcup \{\#\} \subseteq \Gamma$$
$$\textit{and } \delta : Q' \times \Gamma^n \to Q \times \Gamma^n \times \{\mathsf{L}, \mathsf{S}, \mathsf{R}\}^n.$$

Surprisingly (or not), all multi-tape Turing machines can be simulated by some single-tape Turing machine. This can be achieved by stacking all the tapes together (so that they become one "thick" tape) and writing the head positions of each tape on the tape itself.

Fact 1. *$\forall n \in \mathbb{N}$, $\forall$ n-tape Turing machines M, $\exists$ a 1-tape Turing machine M' which can simulate M.*

2.1.2 Multidimensional Turing Machines

A Turing machine can have a multidimensional[2] tape. For instance, the tape could instead be a two-dimensional *sheet*.

[1] m stands for "movement."

[2] Unlike "multi-tape," I chose to not hyphenate "multidimensional," as it is a more common word compared to "multi-tape."

Definition 3 (Multidimensional Turing machines)**.** *A d-dimensional Turing machine $M := (q_0, Q, H, \Sigma, \#, \Gamma, \delta)$ is a 7-tuple where, with $Q' := Q - H$,*

$$\begin{aligned} & q_0 \in Q \supseteq H \\ \textit{and } & \Sigma \sqcup \{\#\} \subseteq \Gamma \\ \textit{and } & \delta : Q' \times \Gamma \to Q \times \Gamma \times \{\mathsf{L}_1, \mathsf{L}_2, \ldots, \mathsf{L}_d, \mathsf{S}, \mathsf{R}_1, \mathsf{R}_2, \ldots, \mathsf{R}_d\}. \end{aligned}$$

Again, surprisingly (or not), all multidimensional Turing machines can be simulated by some one-dimensional Turing machine. This is possible because the boxes of a d-dimensional "tape" (or "sheet" or ...) can still be counted; it is only a matter of enumerating the boxes in a convenient way and translating each d-dimensional movement $m \in \{\mathsf{L}_1, \mathsf{L}_2, \ldots, \mathsf{L}_d, \mathsf{S}, \mathsf{R}_1, \mathsf{R}_2, \ldots, \mathsf{R}_d\}$ to some one-dimensional movement accordingly.

Fact 2. *$\forall d \in \mathbb{N}$, $\forall$ d-dimensional Turing machines M, $\exists$ a one-dimensional Turing machine M' which can simulate M.*

2.1.3 Universal Turing Machines

Certain Turing machines can simulate all other Turing machines (including themselves). We call these *universal Turing machines* or UTMs. Specifically, given as input an encoding $\langle M \rangle$ of some Turing machine $M := (q_0, Q, H, \Sigma, \#, \Gamma, \delta)$ and an encoding $\langle w \rangle$ of an input string $w \in \Sigma^*$, a UTM will output the output of M run with input w.

The Classical Construction The classical construction of a UTM is based on a three-tape Turing machine. Fact 1 guarantees that we can convert this into a one-tape Turing machine; we opt for a three-tape Turing machine merely for ease of exposition. Let us call this UTM M_u.

The first tape of M_u contains the encoding $\langle M \rangle$ of $M := (q_0, Q, H, \Sigma, \#, \Gamma, \delta)$. The second tape is where we will simulate the tape of M; hence it initially contains the encoding of w, $\langle w \rangle$. The third tape keeps track of the simulated state of M; it initially contains the encoding of state $q_0 \in Q$.

With this set up, the simulation of a single transition is straightforward:

1. We read the entirety of the third tape to obtain the current state $q \in Q$;
2. We read the symbol under the head of the second tape $\gamma \in \Gamma$;
3. We search the first tape (which contains the encoding of M which contains the encoding of δ) for $\delta(q, \gamma)$;
   (a) If it exists, let $(q', \gamma', m) := \delta(q, \gamma)$; we update the third tape to store our new state $q' \in Q$, replace the symbol γ under the head of the second tape with γ', and finally move the head of the second tape according to $m \in \{\mathsf{L}, \mathsf{S}, \mathsf{R}\}$;
   (b) Otherwise, this is a halting state; we halt.

By simulating transitions, we can simulate the running of M on input w.

2.2 Embedding Graphs

By an *embedding* of a graph we mean a particular placement of vertices and drawing of edges such that neighboring edges only meet at their endpoints and nowhere else. A graph *can be embedded in* or *is embeddable in* some topological space iff there exists an embedding of it in that (topological) space.

Fact 3. *Any graph can be embedded in the three-dimensional topological space* $\mathbb{R}^3$.

3 Embedding Graphs in the Cubic Lattice $\mathbb{Z}^3$

A natural generalization of Fact 3 one might consider is the following lemma:

Lemma 1. *Any directed graph* $G := (V, E)$ *can be embedded in the cubic lattice* $\mathbb{Z}^3$*; that is, we can draw* G *such that each vertex coincides with some lattice point* $(x, y, z) \in \mathbb{Z}^3$ *and each edge is a concatenation of steps from* $S := \{(\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1)\}$. *Moreover, this embedding requires no more than* $|E|^2(|V| + |E|)$ *or* $O(|E|^2(|V| + |E|)) \subseteq \mathrm{O}(|V|^6)$ *lattice points.*

Proof. Refer to Figures 1 and 2.

WLOG, let $V := [|V|]$. Stack the vertices along the z-axis, with smaller vertices higher up. Ensure the distance between vertex v and $v + 1$ is at least $\deg_+(v) + 1$, so that there are at least $\deg_+(v)$ lattice points in between vertex v and $v + 1$, so that each edge originating from v can leave the z-axis on its own z-level. Let z_e be this z-level for each $e \in E$.

Draw the rest of each edge to its destination vertex. First, each edge e must reach its "orbit"—that is, the Manhattan-distance circle about $(0, 0, z_e)$. There will be a total of $|E|$ orbits. At some point, the edge can *float* or *sink* to the z-level of its destination vertex, and then exit its orbit. Note that edges with the same destination collide only as they exit their respective orbits.

This completes the embedding. Our tower of vertices along the z-axis can be as short as $|V| + |E|$, although in Figure 1 it is $(2|V| + |E|)$-tall due to the extra lattice points we add after each "vertex block." The orbits can be confined to the first xy-quadrant, although in Figures 1 and 2 we use two xy-quadrants for ease of illustration. Hence, this embedding requires no more than $|E|^2(|V| + |E|)$ lattice points.

Even if we space our vertices with s lattice points and use all four xy-quadrants, we need no more than $(2|E|)^2((1 + s)|V| + |E|) \in \mathrm{O}(|E|^2(|V| + |E|)) \subseteq \mathrm{O}(|V|^6)$ lattice points. □

4 The Construction

With this, we may now consider our alternative construction of a universal Turing machine. The main idea is to replace the first and third tapes of the classical construction with a single three-dimensional tape. The resulting two-tape Turing machine has one three-dimensional tape (the first tape) and one one-dimensional tape (the second tape). Let us call this UTM M'_u.

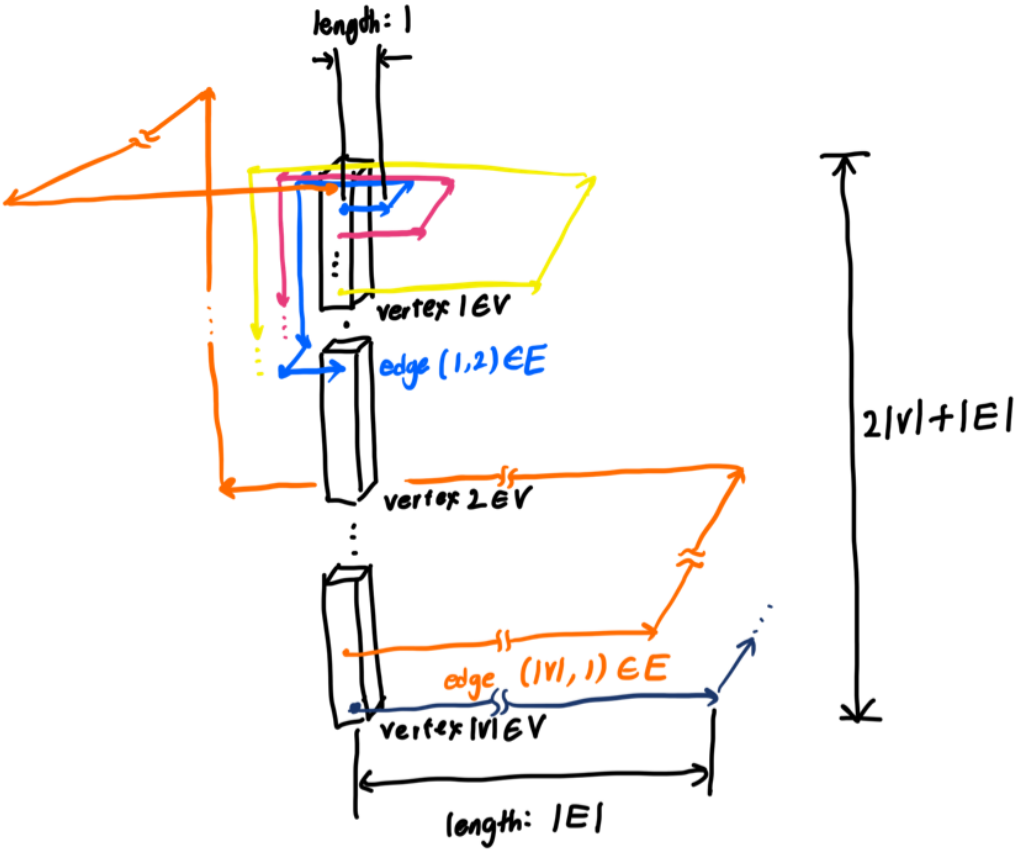

Figure 1: A perspective-projection view of our embedding.

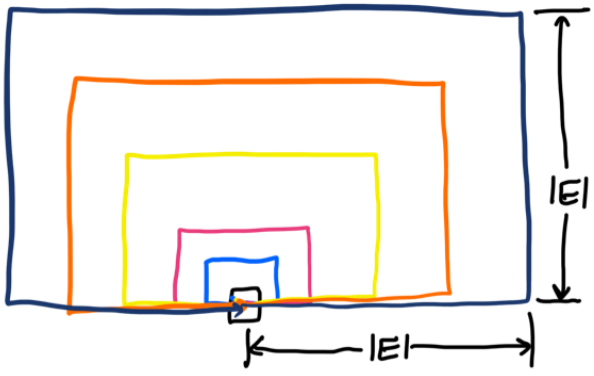

Figure 2: A top-down view of our embedding.

Given $M := (q_0, Q, H, \Sigma, \#, \Gamma, \delta)$, a Turing machine, and an input $w \in \Sigma^*$, we embed the graph representation G_δ of δ in the cubic lattice $\mathbb{Z}^3$ of the three-dimensional tape. Lemma 1 guarantees that this is possible in polynomial-in-$|\langle M \rangle|$ space. One way of doing this can be based on Figure 1: each "vertex block" corresponds to some state $q \in Q$; each point in the block is labelled with an element from Γ. This designates a separate z-level and orbit for each transition $(q', \gamma', m) := \delta(q, \gamma)$ that exists. We connect the vertex blocks with directed edges as in the proof of Lemma 1. We insert information about γ' and m at some point in the edge, before it merges with other edges. Halting states will have no outgoing edges. Note that a three-dimensional tape is isomorphic to a cubic lattice; we simply replace each lattice point with a box containing a symbol. The one-dimensional tape is identical to the second tape of the classical construction (it contains the simulated tape of M; hence it initially contains $\langle w \rangle$).

Again, with this set up, the simulation of a single transition is straightforward:

1. The head of the three-dimensional tape is currently at the upmost box of the "state block" corresponding to $q \in Q$.

2. We read the symbol under the head of the second tape $\gamma \in \Gamma$;

3. The head of the three-dimensional tape scans the state block from top to bottom to see whether the transition $\delta(q, \gamma)$ exists;

   (a) If it exists, the head of the three-dimensional tape follows the sequence of boxes containing step symbols which direct the head to the next state block q'. Along the way, there will be information about γ' and m, encoded in special step symbols such that $\delta(q, \gamma) = (q', \gamma', m)$; we replace the symbol γ under the head of the one-dimensional tape with γ' and move its head according to $m \in \{\mathsf{L}, \mathsf{S}, \mathsf{R}\}$;

   (b) Otherwise, this is a halting state; we halt.

We remark that this is a common way someone studying Turing machines would reason about one. We simulate the Turing machine, with the aid of a transition diagram drawn on paper, with the simulated Turing machine's tape in our head.

Note that the simulated Turing machine need not be a one-dimensional Turing machine. Thanks to the transition diagram being directly represented in one of the tapes, the encoding and subsequent simulation of multidimensional Turing machines is more straightforward. This is the primary advantage of this construction.

5 Conclusion

In this thesis, I proposed an alternate construction of a universal Turing machine, inspired by the way someone studying Turing machines may reason about them. This construction has the benefit that it makes the simulation of multidimensional Turing machines simpler. A proof that any directed graph can be embedded in the cubic lattice $\mathbb{Z}^3$ in polynomial space accompanies the construction.

References

[1] 박근수. 『오토마타 이론』. Unpublished lecture notes. Sept. 2023.