

Halting problem undecidability and infinitely nested simulation (V3)

We define Linz H to base its halt status decision on the behavior of its pure simulation of N steps of its input. If the simulated input cannot reach its own final state in any finite number of steps then H aborts the simulation of this input and transitions to H.qn. H determines this on the basis of matching an infinitely repeating behavior pattern. The copy of H embedded in $\hat{H}$ computes the mapping from its input $\langle \hat{H} \rangle \langle \hat{H} \rangle$ to $\hat{H}.qn$ on the basis of the above criteria.

The following simplifies the syntax for the definition of the Linz Turing machine $\hat{H}$, it is now a single machine with a single start state. A copy of Linz H is embedded at $\hat{H}.qx$.

$\hat{H}.q0 \langle \hat{H} \rangle \vdash^* \hat{H}.qx \langle \hat{H} \rangle \langle \hat{H} \rangle \vdash^* \hat{H}.qy \infty$
 $\hat{H}.q0 \langle \hat{H} \rangle \vdash^* \hat{H}.qx \langle \hat{H} \rangle \langle \hat{H} \rangle \vdash^* \hat{H}.qn$

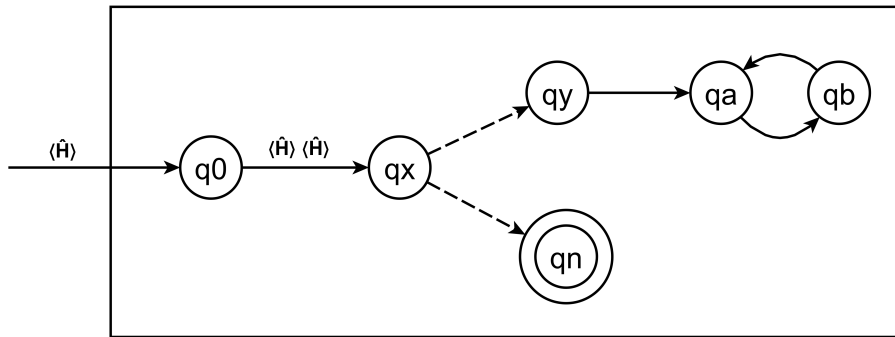


Figure 12.3 Turing Machine $\hat{H}$ applied to $\langle \hat{H} \rangle$

Because it is known that the UTM simulation of a machine is computationally equivalent to the direct execution of this same machine H can always form its halt status decision on the basis of what the behavior of the UTM simulation of its inputs would be.

When embedded_H simulates $\langle \hat{H} \rangle \langle \hat{H} \rangle$ these steps would keep repeating:
 $\hat{H}$ copies its input $\langle \hat{H} \rangle$ to $\langle \hat{H} \rangle$ then embedded_H simulates $\langle \hat{H} \rangle \langle \hat{H} \rangle \dots$

computation that halts ... the Turing machine will halt whenever it enters a final state.
(Linz:1990:234)

This shows that the simulated input to embedded_H $\langle \hat{H} \rangle \langle \hat{H} \rangle$ would never reach its final state conclusively proving that this simulated input never halts. This enables embedded_H to abort the simulation of its input and correctly transition to $\hat{H}.qn$.

if embedded_H does correctly recognize an infinitely repeating behavior pattern in the behavior of its simulated input: $\langle \hat{H} \rangle$ applied to $\langle \hat{H} \rangle$ then embedded_H is necessarily correct to abort the simulation of its input and transition to $\hat{H}.qn$.

A halt decider is a decider embedded_H is only accountable for computing the mapping from $\langle \hat{H} \rangle \langle \hat{H} \rangle$ to $\hat{H}.qy$ or $\hat{H}.qn$ on the basis of the behavior specified by these inputs. embedded_H is not accountable for any other behavior besides the behavior specified by its actual inputs.

Appendix: Peter Linz Halting Problem Proof

Definition 12.1

Let w_M describe a Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$, and let w be any element of Σ^+ . A solution of the halting problem is a Turing machine H , which for any w_M and w , performs the computation

$$q_0 w_M w \vdash^* x_1 q_y x_2,$$

if M applied to w halts, and

$$q_0 w_M w \vdash^* y_1 q_n y_2,$$

if M applied to w does not halt. Here q_y and q_n are both final states of H .

Theorem 12.1

There does not exist any Turing machine H that behaves as required by Definition 12.1. The halting problem is therefore undecidable.

Proof: We assume the contrary, namely that there exists an algorithm, and consequently some Turing machine H , that solves the halting problem. The input to H will be the description (encoded in some form) of M , say w_M , as well as the input w . The requirement is then that, given any (w_M, w) , the Turing machine H will halt with either a yes or no answer. We achieve this by asking that H halt in one of two corresponding final states, say, q_y or q_n . The situation can be visualized by a block diagram like Figure 12.1. The intent of this diagram is to indicate that, if M is started in state q_0 with input (w_M, w) , it will eventually halt in state q_y or q_n . As required by Definition 12.1, we want H to operate according to the following rules:

$$q_0 w_M w \vdash^* H x_1 q_y x_2,$$

if M applied to w halts, and

$$q_0 w_M w \vdash^* H y_1 q_n y_2,$$

if M applied to w does not halt.

Figure 12.1

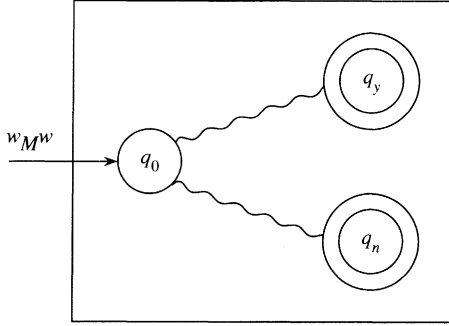
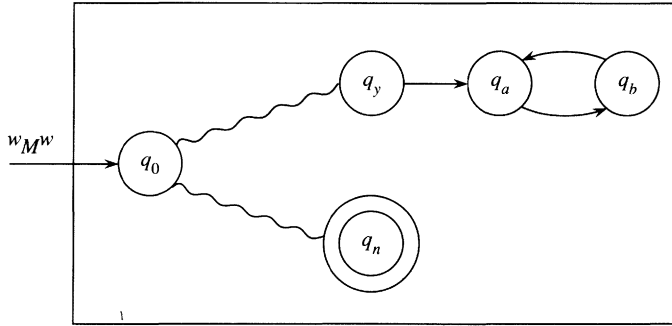


Figure 12.2



Next, we modify H to produce a Turing machine H' with the structure shown in Figure 12.2. With the added states in Figure 12.2 we want to convey that the transitions between state q_y and the new states q_a and q_b are to be made, regardless of the tape symbol, in such a way that the tape remains unchanged. The way this is done is straightforward. Comparing H and H' we see that, in situations where H reaches q_y and halts, the modified machine H' will enter an infinite loop. Formally, the action of H' is described by

$$q_0 w_M w \vdash^*_{H'} \infty,$$

if M applied to w halts, and

$$q_0 w_M w \vdash^*_{H'} y_1 q_n y_2,$$

if M applied to w does not halt.

From H' we construct another Turing machine $\hat{H}$. This new machine takes as input w_M , copies it, and then behaves exactly like H' . Then the action of $\hat{H}$ is such that

$$q_0 w_M \vdash^* \hat{H} q_0 w_M w_M \vdash^* \hat{H} \infty,$$

if M applied to w_M halts, and

$$q_0 w_M \vdash^* \hat{H} q_0 w_M w_M \vdash^* \hat{H} y_1 q_n y_2,$$

if M applied to w_M does not halt.

Now $\hat{H}$ is a Turing machine, so that it will have some description in Σ^* , say $\hat{w}$. This string, in addition to being the description of $\hat{H}$ can also be used as input string. We can therefore legitimately ask what would happen if $\hat{H}$ is applied to $\hat{w}$. From the above, identifying M with $\hat{H}$, we get

$$q_0 \hat{w} \vdash^* \hat{H} \infty,$$

if $\hat{H}$ applied to $\hat{w}$ halts, and

$$q_0 \hat{w} \vdash^* \hat{H} y_1 q_n y_2,$$

if $\hat{H}$ applied to $\hat{w}$ does not halt. This is clearly nonsense. The contradiction tells us that our assumption of the existence of H , and hence the assumption of the decidability of the halting problem, must be false. ■

Linz, Peter 1990. An Introduction to Formal Languages and Automata. Lexington/Toronto: D. C. Heath and Company. (317-320)