Notre Dame Journal of Formal Logic
Volume IX, Number 4, October 1968

APPROXIMATION TO A DECISION PROCEDURE FOR THE HALTING PROBLEM*

MICHAEL ANDERSON

Both the Halting Problem and the Immortality Problem for Turing machines (to provide an algorithm for determining if a given Turing machine will stop on a given input or, respectively, any input) have been shown to be unsolvable. In this paper we consider the problem of determining if a machine is mortal (stops on any input) solely from the machine's state instructions.

The machines considered here are two symbol machines (0 and 1) with right and left shift. Particular machines will be given in the form of machine tables. At each entry of the table there are three instructions--the print instruction, the shift instruction (1 for right and -1 for left), and the state instruction (q_i for the i^{th} state and, in particular, q_0 for the stop state). The machines will operate on two way infinite tapes on which appear only a finite number of strokes.

At many points in the work that follows it will be unnecessary to consider the complete machine table. When this occurs the instructions will still be referred to as a machine although they actually determine a set of machines.

The State Tree of a Machine. With each machine we can associate a dyadic tree[1], called the state tree of the machine, constructed in the following way:

---

*This research was done during the school year 1966-67 in conjunction with the Seminar in Symbolic Logic of Prof. B. Sobociński. I would like to thank Prof. B. Sobociński and Prof. V. Vučković for their kind encouragement and assistance.

1) The reader is referred to [1], [2], and [5] for the terminology connected with trees. We assume the concepts of length and order ordinarily associated with trees. For a development of the structural propositions in this section of the paper see [5] pp. 181-197, in particular pp. 190-191. The state tree of this paper is nothing more than that of [5] with some of the structure stripped off and paths cut down.

Received October 9, 1967

(1) The origin of the tree is the initial state of the machine.
(2) If a q_i in the tree is a q_0 or if the path from the origin to this q_i has an occurrence of q_i other than the q_i under consideration, this q_i is a terminal point. The path from the first occurrence of q_i to the second occurrence is called the cycle of the terminal point q_i.
(3) If a q_i satisfies neither of the conditions of (2), then from q_i there issue two edges: one, called an upper edge, connecting q_i to the state the machine will go to if in state q_i it observes a stroke and the other, called a lower edge, connecting q_i to the state the machine will go to if in state q_i it observes a blank. The point connected to q_i by an upper (lower) edge is called the upper (lower) point of q_i.

Notice that two points of the tree may be designated by the same state symbol. We call such points identical points. Also, there may be some states of a machine which do not appear in the state tree. Such states can never be reached by the machine in a computation and so, for our purposes, are superfluous. We eliminate from consideration machines with superfluous states. We associate with the empty machine, the machine consisting only of the stop state, the tree with single point q_0.

A tree is called a state tree iff it is a state tree of some machine after an appropriate renumbering under which q_0 remains q_0 and non-identical points remain non-identical. In what follows we will often transfer from a machine or tree to a renumbering of the machine or tree. The q_0 will always remain fixed and q_1 will denote the initial state or origin of a machine or tree.

Proposition 1: *The state tree of any machine is finite.*

Proposition 2: *Any dyadic tree in which each point is denoted by a q_i is a state tree iff it satisfies the following conditions*:
(1) *The two points following a junction point are distinguished, one being called upper and the other lower. Identical junction points have identical upper points and identical lower points.*
(2) *Any terminal point is either a q_0 or the path connecting the origin of the tree to the terminal point has precisely one point different from the terminal point and identical to it.*
(3) *Within any path there is not a pair of identical points.*

We now proceed to introduce further terminology. A path from the origin of a state tree to one of its terminal points will be called a complete path. A path consisting solely of upper edges will be called an upper path and a path consisting of lower edges, a lower path. The unique complete upper path will be called the main path. A path with an odd (even) number of edges will be called an odd (even) path.

Any point of a state tree T which is not a terminal point is the origin of a number of subtrees of T. We call the largest such tree (the one with the greatest number of edges) the subtree of this point. If the subtree of a point in T is a state tree, then it is called a state subtree of T.

Proposition 3: *If the path connecting the origin of a state tree T to a point*

q_i contains no point identical to a point of the subtree of q_i other than q_i itself, then the subtree of q_i is a state subtree.

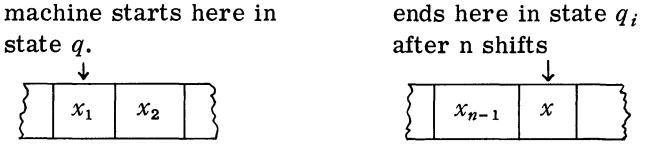
A submachine of a machine **M** is a set of columns of the machine table of **M** which is a machine[2]. If a computation of **M** enters into a submachine, it will remain there. In the state tree T of a machine **M**, state subtrees correspond to submachines of **M** and conversely. Two state subtrees which have identical origins correspond to the same submachine.

We call the subtree of T associated with the upper (lower) point of a point the upper (lower) tree of the point.

With each path in a state tree we associate a sequence $(q_{i_1}, \ldots q_{i_n})$, where q_{i_j} is the state associated with the j^{th} point of the path, and a sequence $(x_{i_1}, \ldots x_{i_{n-1}})$, where x_{i_j} is 1 if the j^{th}, edge of the path is an upper edge and 0 if it is a lower edge. These are called, respectively, the state sequence and the input sequence associated with the path. In what follows, whenever we consider a particular path we will renumber the state tree in order to obtain consecutive numbering for the sequences.

The significance of these two sequences is seen in that given a state tree T and a complete path of T with state sequence $(q_1, \ldots q_{n-1}, q_i)$ and input sequence $(x_1, \ldots x_{n-1})$, the machine with state tree T which shifts only to the right, if started on the tape as shown in fig. 1, will go through the state sequence of the path and end in state q_i.

Figure 1



A machine is said to cycle (to have cycled) when it returns to a state it was in before.

Lemma 1: *Given a state tree T with a state subtree T' having origin q_i and a machine* **M'** *with state tree T', there is a machine* **M** *with submachine* **M'** *and state tree T such that for any input I' for* **M'** *there is an input I for* **M** *which will put* **M** *in state q_i on the input I'.*

Proof: Consider the path connecting the origin of T to q_i. Say this path has state sequence $(q_1, \ldots q_i)$ and input sequence $(x_1, \ldots x_{i-1})$. Then none of the states $q_1, \ldots q_{i-1}$ appear in T' since T' is a state subtree. Thus assigning instructions to these states will not determine any of the instructions of the submachine associated with state subtree T'. Consider the input given as in fig. 2. Construct **M** so that, in state q_k observing x_k, **M** has instructions

2) Note that we have restricted ourselves to well defined machines with non-superfluous states.

$y_{k-i}1q_{k+1}$, for $k < i$. Start this machine on the tape in fig. 3 and it will go to the desired configuration.

Figure 2

Figure 3

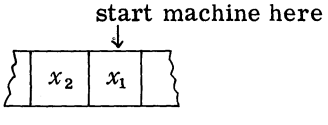
The Decision Procedure. We will now consider the problem of when a given state tree corresponds only to mortal machines. A state tree having this property will be called a mortal state tree.

Lemma 2: *Consider a state tree T in which the only terminal points are q_0 and q_1. T is mortal iff all terminal points other than the terminal point of an odd main path are q_0.*

Proof: (1) If all complete paths of T terminate in q_0, then the computation of any machine with state tree T cannot cycle and must stop.

(2) Suppose T has a complete even path which terminates in q_1. Consider the state sequence $(q_1, \ldots q_{2m}, q_1)$ and input sequence $(x_1, \ldots x_{2m})$ associated with this path. Construct the machine with the following instructions—in state q_j observing x_j the machine has instructions $x_{j+2}(-1)^j q_{j+1}$ where $j+1$ is taken modulo $2m$ and $j < 2m+1$. Then on the input of fig. 4 this machine doesn't stop.

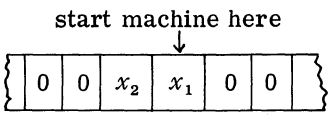
Figure 4



(3) Suppose there is a complete odd path other than the main path which ends in q_1. Consider the associated sequences of this path $(q_1, \ldots q_{2m-1}, q_1)$ and $(x_1, \ldots x_{2m-1})$. There is at least one x_i such that $x_i = 0$. Call this x_k. Construct the machine with the following instructions--when

observing x_j in state q_j the machine has the instructions $x_{j+2}(-1)^j q_{j+1}$ for $j<k-1$ and $x_{j+2}(-1)^{j+1} q_{j+1}$ for $j \geq k-1$. Then on the tape of fig. 5 this machine does not stop.

Figure 5



(4) Consider a state tree in which the main path is odd and terminates in q_1 and all other paths terminate in q_0. The only chance for this machine to be non-mortal is if it cycles on the main path. But, since the main path is odd, any cycle must leave the machine at least one space to the right or left of its starting space. Thus after n cycles it must be at least n spaces away from its starting space. Therefore the machine will eventually be thrown into a computation off the main path. But these all terminate in q_0. Thus the machine will eventually stop.

Corollary: *Any state tree which has an occurrence of q_1 as a terminal point other than on an odd main path is non-mortal.*

Theorem 1: *Given a mortal state tree T, any state subtree of T is mortal.*

Proof: Suppose there is a state subtree of T which is not mortal. Then there is a machine corresponding to this state subtree which has an endless computation. But, by Lemma 1, this implies that there is a machine with state tree T which has an endless computation and thus T is not mortal. Thus any state subtree of T is mortal.

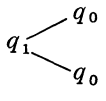
We define the order of a finite tree to be the maximum length of its paths.

Theorem 2: *A state tree T is mortal iff every terminal point of T is either a q_0 or a q_i whose cycle is an upper odd path.*

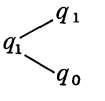
(1) Assume there is a terminal point q_k whose cycle is not an upper odd path. Consider the path from the origin of T to this terminal point and the associated sequences $(q_1, \ldots q_k, \ldots q_{n-1}, q_k)$ and $(x_1, \ldots x_{n-1})$. Combining the construction of Lemma 1 with the constructions of (2) and (3) of Lemma 2, we get a machine with state graph T which has an endless computation. Thus T is not mortal.

(2) Consider the set of all state trees such that every terminal point is either a q_0 or has a cycle which is an upper odd path. We proceed by induction on the order of the trees.

(i)



and



are mortal.

(ii) Assume that for all state trees of order $k < n$ the theorem holds. Let T be a state tree of order n satisfying the condition. Consider the upper and lower trees of T. The lower tree has no occurrence of q_1 and thus is a state subtree of T. It satisfies the condition and is of order less than n. Thus it is a mortal state tree. With the upper tree we must distinguish two cases:
Case 1: If the upper tree has no occurrence of q_1, it is, as with the lower tree, a mortal state tree. This implies that T is mortal.
Case 2: If the upper tree has an occurrence of q_1, it must be as the terminal point of the main path and this path must be odd. Consider the state sequence of the main path, $(q_1, \ldots q_{2m+1}, q_1)$. None of the q_i on the main path have points off the main path identical to them for this would imply the existence of q_1 as a terminal point other than on the main path or the existence of a non-upper cycle. Thus the lower trees of all the points on the main path are state subtrees of T. Since they satisfy the hypothesis and have order less than n, they are all mortal. Thus the only chance for a machine with state tree T to cycle indefinitely is to stay on the main path for its computation. But as in (4) of the proof of Lemma 2, the machine can't have an endless computation on the main path. Thus T is mortal.

Formal Results. We now present a formal system representing the preceding results. To this end, we associate with each state tree a formula.

First consider all cycles of a state tree T with origin q_i. A point of a state tree is said to occur under another point if there is a point with subtree T' which contains both points and such that the first point is contained in the lower tree of T' and the second is contained in the upper tree of T'. We consider the lowest[3] occurrence of q_i and eliminate the subtrees, save for the origin itself, of all the other occurrences of q_i. The remaining tree, called the reduced tree of T, is dyadic. Different state trees have different reduced trees. We define upper and lower trees etc. as with state trees.

From the reduced tree for T we construct a formula for T in the following manner:

(1) With $q_i \lessdot^{q_j}_{q_k}$, q_j the upper point of q_i and q_k the lower point, we associate $Kq_iq_jq_k$.

(2) With $q_i \lessdot^{A}_{B}$ where A and B are dyadic trees, we associate $Kq_i\alpha\beta$ where α is the associated formula of A and β is the associated formula of B.

Then for any tree there is a unique formula and a formula corresponds to no more than one tree

The Formal System

Variables: $q_1, \ldots q_i$
Constant: q_0

3) Among any set of identical junction points, there is a lowest.

Connective: K

Wffs: (1) Any constant or variable is a wff.
(2) If α, β, and γ are wffs, then $K\alpha\beta\gamma$ is a wff.
(3) Nothing else is a wff.

Normal Numbering: If in a wff an occurrence of q_{i+1} immediately following a K is preceded in the formula by an occurrence of q_i, immediately following a K for $i \geq 1$; then the wff is said to be normally numbered.

Part: We define a part of a wff inductively as follows--
(1) If $K\alpha\beta\gamma$ is a wff, where α, β, and γ are wffs; then β and γ are parts of $K\alpha\beta\gamma$.
(2) A part of a part of a wff is a part of that wff.
(3) Nothing else is a part.

System **K**

Rules:

Substitution: *For any occurrence of q_0 in a thesis θ we may substitute a thesis Δ where θ and Δ are renumbered in order not to identify distinct variables and to keep the numbering normal.*

Reference: *For any occurrence of q_0 in a thesis θ we may substitute a q_i where q_i occurs in a part of θ which is a thesis after renumbering and this part of $\bar{\theta}$ occurs after the q_0 to be substituted for.*

Axioms:

A_0: q_0

A_1: $Kq_1q_0q_0$

A_2: $Kq_1Kq_2 \ldots Kq_{2i-1}q_1q_0 \ldots q_0$ for all $i > 0$

Note that system **K** is trivially a decidable system since there is no rule by which a thesis yields a shorter thesis.

Theorem 3: *A state tree is mortal iff its associated formula, after renumbering, is a thesis of the* **K** *system* (semantic completeness).

Proof: (1) The reader may verify that all the theses of the **K** system correspond to mortal state trees.

(2) Consider the set of all mortal state trees. We proceed by induction on the order of the reduced trees.

(i) The formula associated with the state tree q_0 is in the **K** system.

(ii) Assume for all $k < n$ the formula of a state tree with reduced tree of order k has its associated formula in the **K** system. Consider T', of order n, the reduced tree of T.

The lower tree of T is of lower order than T and, since it is the reduced tree of a state tree, its formula is in the **K** system. Call it α_1.

We distinguish two cases for the upper tree of T.

Case 1: If there is no occurrence of q_1 in the upper tree of T, then by substituting for any q_i which appears in the lower graph of T a q_0 we get the reduced tree of a state tree whose order is less than n and which is mortal. Thus its formula is in the **K** system. Call it α_2.

We have $Kq_1q_0q_0$ as a thesis and thus, by substitution, we have $Kq_1\alpha_2\alpha_1$, renumbering α_1 and α_2. By the reference rule we may substitute for any q_0 in α_2 a q_i appearing in α_1. In particular wherever we substituted into the original tree a q_0 for a q_i, we may substitute this q_i back (actually its appropriate renumbering). Thus the formula of T is in the system.

Case 2: If q_1 does occur in the upper tree of T, it only occurs as terminal point of the main path. Consider the state sequence of the main path, $(q_1, \ldots q_{2m+1}, q_1)$. As in the proof of Theorem 2, the lower trees of the points on the main path correspond to state subtrees. By substituting in the lower tree of q_i a q_0 for any occurrence of a q_j which appears in the lower tree of a q_u, where $u < i$, we get a reduced tree which is a reduced state subtree and is of order less than n. Call the formula of the **K** system associated with the state subtree of q_i α_i.

We have as a thesis of the system $Kq_1Kq_2(K \ldots Kq_{2m+1}q_1q_0 \ldots q_0$ and thus we have $Kq_1Kq_2K \ldots Kq_{2m+1}q_1\alpha_{2m+1} \ldots \alpha_1$. But $\alpha_{2m+1} \ldots \alpha_1$ are theses of the **K** system. Thus any q_0 in α_{2m+1} can be changed to a q_i appearing in $\alpha_{2m+1} \ldots$ or α_1. Thus we can substitute back the q_j which we eliminated before. Continuing the process we substitute back for the q_0's appearing in $\alpha_{2m}, \ldots \alpha_2$ and get the formula of the original tree.

BIBLIOGRAPHY

[1] Ore, *Theory of Graphs*, American Mathematical Society Publications, 1962.

[2] Smullyan, "Trees and Nest Structures," *The Journal of Symbolic Logic*, Vol. 31, no. 3, Sept. 1966.

[3] Hooper, "The Undecidability of the Turing Machine Immortality Problem," *The Journal of Symbolic Logic*, Vol. 31, no. 2, June 1966.

[4] Kleene, *Introduction to Metamathematics*, Van Nostrand, 1952.

[5] Kobrinskii and Trakhtenbrot, *The Theory of Finite Automata*, North Holland, 1965.

[6] Shannon, "A Universal Turing Machine with Two Internal States," *Automata Studies*, Annals of Mathematics Studies, No. 34, 1956.

Seminar in Symbolic Logic
University of Notre Dame
Notre Dame, Indiana