Simulation of Infinite Time Turing Machines on a Classical Turing Machine.

Alexey Nikolaev, Computer Science Program, CUNY Graduate Center, New York, NY.
email: anikolaev@gc.cuny.edu

Abstract

The concept of Infinite Time Turing Machine (ITTM) introduced by Hamkins and Lewis [1] is an extension of conventional finite time Turing Machines (TM) to transfinite ordinal running times. This computational model is strictly more powerful than classical TMs, because the Halting Problem for classical TMs is decidable by ITTM. Interestingly, ITTMs that are using only finite tape can be simulated on a conventional finite time computer. We show how loops of repeating configurations can be identified and reduced. Using this reduction algorithm, we implement an actual ITTM interpreter. Using this method, some non-trivial ITTM computations can be simulated in finite time. For example, such tasks as clocking ordinals can be simulated in finite time for any ordinal less than ω^ω. Also, we show that the Halting Problem for ITTMs with finite tape is decidable by classical finite time TM.

ITTMs run in transfinite ordinal time:

$0 \to 1 \to 2 \to 3 \to 4 \to \cdots$

$\to$ (intinifely many steps of computation) $\to$

$\omega \to \omega+1 \to \omega+2 \to \omega+3 \to \cdots$

$\to$ (intinifely many steps of computation) $\to$

$\omega\cdot 2 \to \cdots \to \omega\cdot 3 \to \cdots \to \omega^2 \to \cdots \omega^3 \to \cdots \to$

$\omega^\omega \to \cdots \to \omega^{\omega^\omega} \to \cdots$

Limit ordinals

Non-zero ordinals that are not successors of any other ordinal: ω, $\omega\cdot 2$, $\omega\cdot 3$, $\ldots \omega^\omega$, $\ldots \omega^\omega+\omega$, $\omega^\omega+\omega\cdot 2$, $\ldots \omega^{\omega}\cdot 2$, $\ldots$, ω^{ω^ω}, $\ldots$, $\omega^{\omega^{\omega^\omega}}$, $\ldots$.

Successor ordinals

Can be represented in the form $\alpha+1$ (as a set, the maximum element of a successor ordinal $\alpha+1$ is α).

Counting time & Strong repeats

Definition 1. *Let δC be the transition time from a (possibly compound) configuration C to $\tau(C)$.*

Lemma 1. *(Loop Reduction). A loop of ITTM configurations $[C_s \to \ldots \to C_t]^\omega$ s.t. $\tau(C_t) = C_s$, can be reduced to a compound configuration L, and its transition time is*

$$\delta L = (\delta C_s + \ldots + \delta C_t)\cdot\omega.$$

The order, in which δC_i are added up is insignificant. It is clear that loops may occure inside other loops, that is how compound configurations can be nested one into another. Similar looping behavior is also discussed in [3].

Lemma 2. *("Strong" Repeat). ITTM does not halt if $\tau(L) \in L$.*

Proof. Since transition $L \dashrightarrow \tau(L)$ repeats all configurations of L infinitely many times, and the resulting limit configuration $\tau(L)$ is nevertheless in L, nothing new can happen to the machine any more. All possible changes to the tape have happened infinitely many times already, and repeating them again does not change limit configuration, thus this is a "strong" repeat indeed. □

Using these two lemmas, we can construct a reduction algorithm, which can simulate the ITTMs that do not need an infinite tape for computation. "Strongly" repeating non-terminating configurations can be identified. In fact, the Halting problem of the ITTMs with finite tape is decidable by a finite time TM.

ITTM at succesor ordinal times is equivalent to Classical TM

Instructions are quintuples $(q,s) \to (q',s',a)$.
States $q \in Q$, Q is finite. Symbols $s \in \Sigma = \{0,1\}$.
Actions $a \in A = \{L,S,R\}$.
The tape is infinite to the right. At moment t, the i^{th} cell is $T_i(t)$ and the head position is $H(t)$.

ITTM at limit ordinal time t

As defined by J. D. Hamkins and A. Lewis in [1]:

$$\text{head position } H(t) = 0,$$

$$\text{state } Q(t) = \textsc{Limit},$$

$$i^{th} \text{ cells of the tape } T_i(t) = \limsup_{s\to t} T_i(s).$$

So, after infinitely many steps:
1) the head goes to the beginning of the tape,
2) the machine is in the special Limit state,
3) unstabilized cells of the tape become equal to 1.

How to run an ITTM in finite time?

Configuration of the machine at time t is

$$C(t) = \langle Q(t), H(t), T(t)\rangle.$$

Call such configurations simple.
Transition function

$$\tau : C(t) \mapsto C(t+1).$$

Compound configuration (loop) is a set of simple configuration.
Transition function for loops

$$\tau : L \mapsto C,$$

where C is a simple configuration.

Why is it possible?

- When you get into configuration $C_i \in L$, where L is a compound configuration (i.e. an infinite loop), then you will visit all $C_j \in L$ infinitely many times.
- The "entry point" into L does not matter.
- If a compound configuration L contains only finitely-many simple configurations, we can produce the corresponding limit configuration in finite time (just bitwise-OR the tapes of all $C_j \in L$ and you are done).

Definition 2. *An ordinal α is called ITTM-clockable, if there is an ITTM program, which on input 0 halts in exactly α many steps of computation, that is, the α^{th} step is the act of changing to the* Halt *state.*

Example. Clocking ω^4

$(Q_0, x) \to (Q_1, x, R)$ *(move to the right)*

$(Q_1, x) \to (Q_{10}, x, R)$ *(move to the right again)*

$(Q_{10}, 1) \to (Q_{10}, 0, L)$ *(while on a 1, zero it and move to the left)*

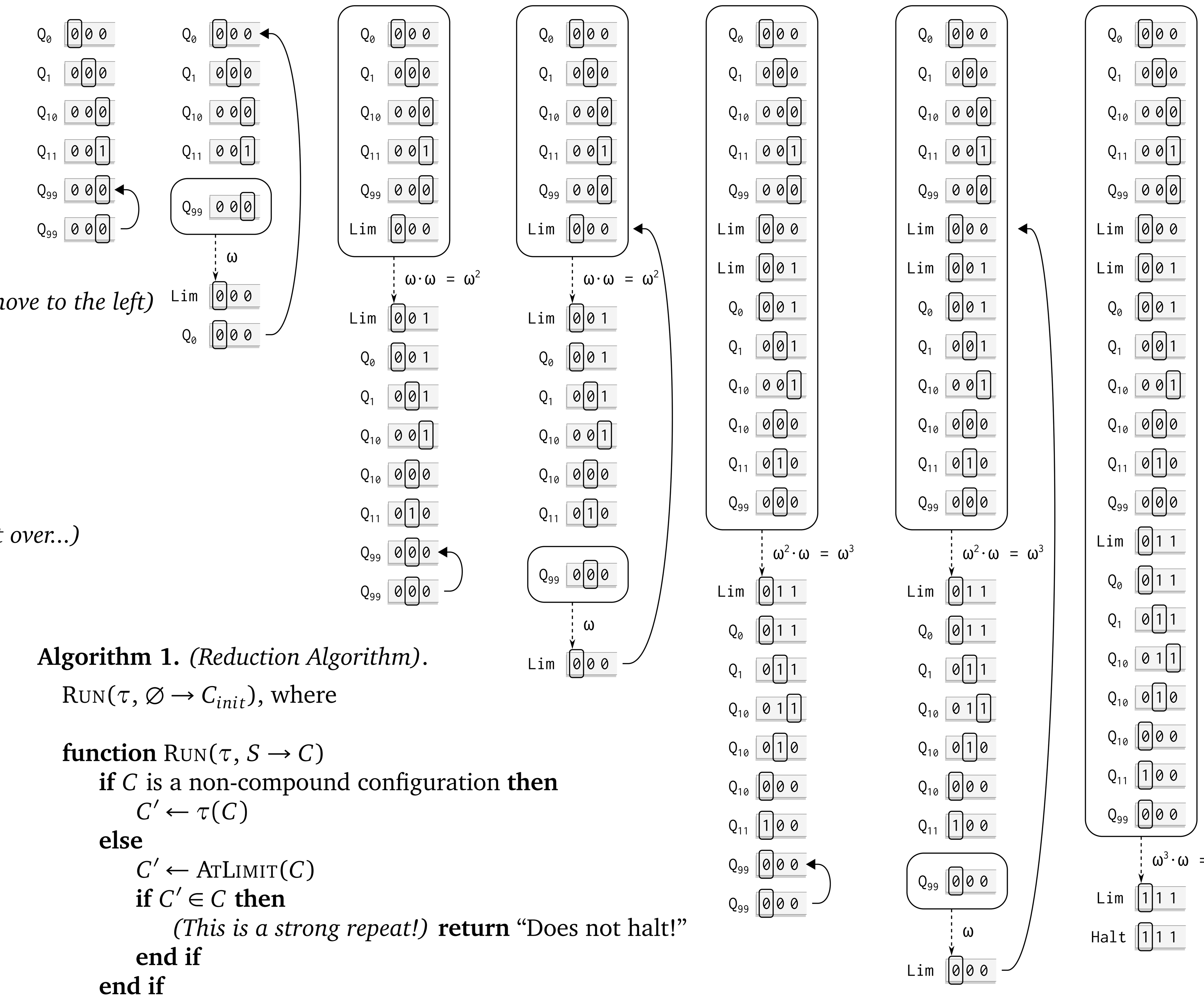
$(Q_{10}, 0) \to (Q_{11}, 1, S)$ *(flash on ...)*

$(Q_{11}, 1) \to (Q_{99}, 0, S)$ *(... and off)*

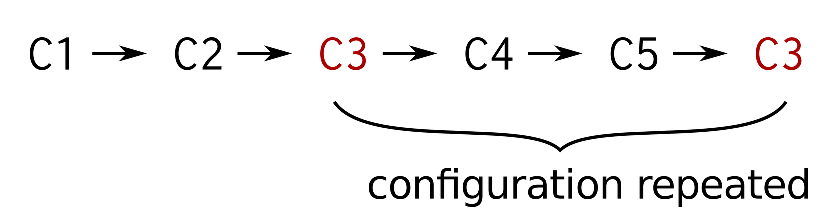
$(Q_{99}, x) \to (Q_{99}, x, S)$ *(just wait)*

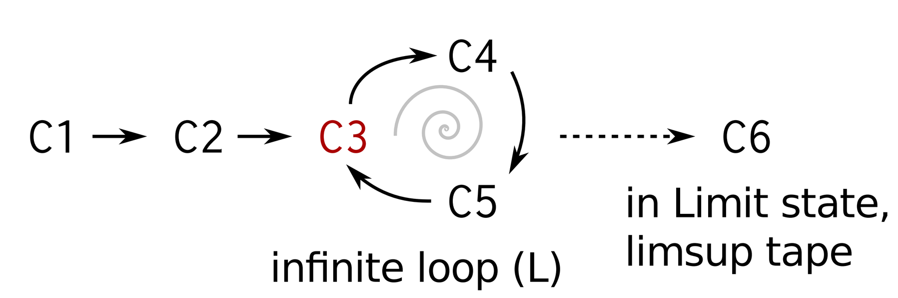
$(\textsc{Limit}, 0) \to (Q_0, 0, S)$ *(at the limit, you either start over...)*

$(\textsc{Limit}, 1) \to (\textsc{Halt}, 1, S)$ *(... or halt)*

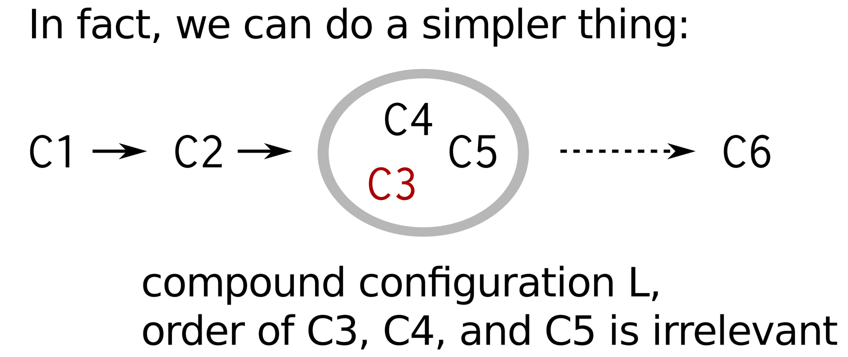


The Idea





In fact, we can do a simpler thing:

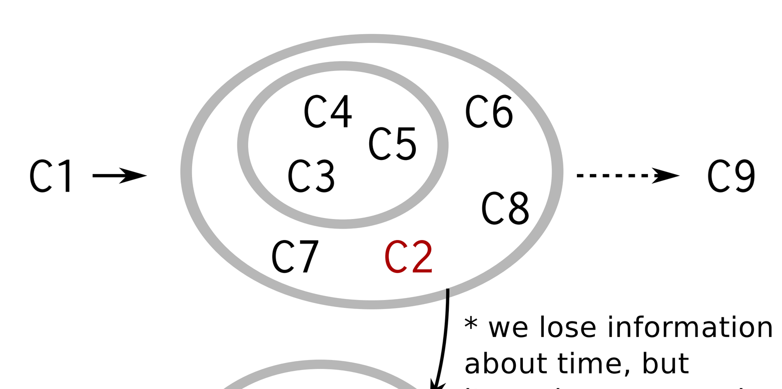


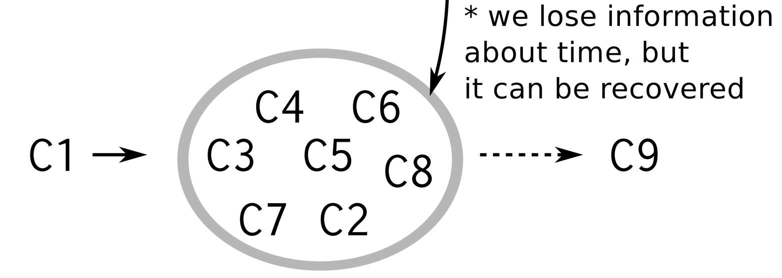
C1 → C2 → {C3,C4,C5} ⇢ C6

Continue execution:

C1 → C2 → {C3,C4,C5} ⇢ C6 →

→ C7 → C8 → C2





C1 → {C2,C3,C4,C5,C6,C7,C8} ⇢ C9

Algorithm 1. *(Reduction Algorithm).*

$\textsc{Run}(\tau, \varnothing \to C_{init})$, where

function $\textsc{Run}(\tau, S \to C)$
  if C is a non-compound configuration **then**
    $C' \leftarrow \tau(C)$
  else
    $C' \leftarrow \textsc{AtLimit}(C)$
    if $C' \in C$ **then**
      (This is a strong repeat!) **return** "Does not halt!"
    end if
  end if
  if $C' = C_{\textsc{Halt}}$ **then return** C'
  else
    $S' \leftarrow \textsc{Reduce}(S \to C \to C')$
    $\textsc{Run}(\tau, S')$
  end if
end function

function $\textsc{Reduce}(S \to C)$
  if C is found in the list S **then**
    Find sequences before and after the repeat:
    S_{before} and S_{after}, such that $S = (S_{\text{before}} \to R) + S_{\text{after}}$, and ($C \in R$ or $C = R$)
    $L \leftarrow \textsc{Merge}((\varnothing \to R) + S_{\text{after}})$
    return $S_{\text{before}} \to L$
  else
    return $S \to C$
  end if
end function

function $\textsc{AtLimit}(L)$
  return the limit configuration of the infinitely repeating compound configuration L
end function

← Notation

Note, that the sequence of configurations $C_0 \to C_1 \to \ldots \to C_n$ is in fact a list, with symbol ($\to$) being a left-associative list construction operator, (i.e. Tail $\to$ Head). Let $\varnothing$ be the empty list, and ($+$) be the operator of list concatenation.

Conclusion

We propose a method to simulate ITTMs in finite time, if they use only finite tape, i.e. its computation goes through only finite number of configurations. Computation time can be correctly counted by this simulation. The Halting problem of such ITTMs is decidable by finite time TM (we identify strongly reapeating configurations), so only relatively weak ITTMs can be simulated. Nevertheless, we are able to perform some intrinsically infinite-time programs, for example, we can simulate ITTMs clocking ordinals $\alpha < \omega^\omega$.

References

[1] J. D. Hamkins and A. Lewis, Infinite Time Turing Machines, *The Journal of Symbolic Logic*, **65**(2), 2000.
[2] A. M. Turing, On Computable Numbers, with an Application to the Entscheidungsproblem, *Proceedings of the London Mathematical Society*, 2 **42** (1), 1937.
[3] P. Koepke, Turing computations on ordinals, *Bull. Symbolic Logic*, **11**(3), 2005.