The Simple Roots of Real-Time Computation Hierarchies*
(Preliminary version)

Paul M.B. Vitányi

Centre for Mathematics and Computer Science (C.W.I.), Amsterdam†

SUMMARY

A BLAH machine is any memory device that can be simulated in real-time by a multitape Turing machine and such that a multiBLAH machine can real-time simulate a pushdown store. A multiBLAH machine consists of a finite control connected to an input terminal and an output terminal and one or more copies of the BLAH memory unit. It is shown that a $(k+1)$-BLAH machine is more powerful in real-time than a k-BLAH machine, for each k. Thus the hierarchies, within the real-time definable computations, are *proper* and *smooth*, that is, adding a device *always* increases power. It also turns out that all real-time hierarchy results in this vein are simple corollaries of a single root: the real-time hierarchy of multipushdown store machines. As examples of such new results we mention that in real-time, $k+1$ tape-units with a fast rewind square are more powerful than k such units; that $(k+1)$-head tape-units with fast rewind squares are more powerful than k-head tape-units with fast rewind squares; that $(k+1)$-dequeue machines are more powerful than k-dequeue machines; and that $(k+1)$-concatenable-dequeue machines are more powerful than k-concatenable-dequeue machines.

1. Introduction

It is known that $(k+1)$-tape Turing machines cannot be simulated in real-time by k-tape Turing machines [Aa, Pa]. We shall show that the same property holds for types of memory units, other than single-head tape units, if they satisfy the following condition.

(*) *Proper Power Increase.* For each assemblage of k units there is a finite number l such that an assemblage of $k+l$ units is more powerful in real-time than the assemblage of k units.

Hierarchy results concerning #tapes, #heads, with and without head-to-head jumps have been demonstrated in [Aa, Vi, PSS] and elsewhere. They are all corollaries to Theorem 1 below which in its turn rests on the exploitation of the (*) observation.

Theorem 1. *Let a k-BLAH machine consist of k copies of BLAH connected to a common finite control which is attached to an input- and output terminal. Let a BLAH be a memory device which can real-time simulate a pushdown store (or is such that a multiBLAH machine can real-time simulate a pushdown store) and which in its turn can be real-time simulated by a multitape Turing machine. In real-time,* $(k+1)$*-BLAH machines are more powerful than k-BLAH machines.*

Note that Limited Random Access Turing machines [FR], deques, concatenable dequeues, single-head tapes, multihead tapes, multihead tapes with head-to-head jumps, stacks, and what have you, are BLAH machines. Consequently, we first list some immediate *new* corollaries of Theorem 1 which represent hitherto unknown hierarchies. The reader may think of a few others on his own.

Limited Random Access Turing machines were introduced in [FR]. They consist of a multitape Turing machine with *fast rewind squares*. That is, the machine can drop a marker on a scanned square and henceforth can reset the head concerned in one step to the marked tapesquare, regardless of the distance in between. It is shown in [FR] that such machines are not more powerful than multitape Turing machines in real-time. In fact, 14 tapes suffice to simulate 1 fast rewind tape unit in real-time. Since additional tapes increase power (follows trivially from the 14 versus 1 relation above together with that according to [Aa] $14k+1$ tapes are more powerful in real-time than $14k$ tapes, $k \geqslant 0$) we satisfy (*). More in particular, as required by Theorem 1 the devices are capable of real-time simulating a pushdown store and a finite number of pushdown stores is capable of simulating the device. Hence,

* This work was supported by the *Stichting Mathematisch Centrum*.

† Full Address: Centre for Mathematics and Computer Science (C.W.I.), Kruislaan 413, 1098 SJ Amsterdam, The Netherlands.

Corollary. $(k+1)$-tape Turing machines with fast rewind squares are more powerful in real-time than k-tape Turing machines with fast rewind squares. Similarly, $(k+1)$-head Turing machines with fast rewind squares are more powerful in real-time than k-head Turing machines with fast rewind squares.

A *dequeue* is a double-ended queue, and at least as powerful as a pushdown store, see e.g. [LS, Ko]. A multidequeue machine with the additional option of instantaneous concatenation of any current dequeue contents onto another dequeue's contents, thus emptying the first dequeue in a single step, is called a multi*concatenable* dequeue machine. Such machines can be real-time simulated by multitape Turing machines [Ko]. Consequently we have, by Theorem 1,

Corollary. $(k+1)$-dequeue machines are more powerful in real-time than k-dequeue machines. Also, $(k+1)$-concatenable dequeue machines are more powerful in real-time than k-concatenable dequeue machines.

2. The smoothness of real-time memory hierarchies

Theorem 1 establishes many real-time hierarchies within the real-time definable computations (computations which can be performed in real-time by multitape Turing machines), a small sample of which was exhibited in the Corollaries above. The novelty of the present approach is that the arguments used are surprisingly simple and uniform for a great variety of Turing machine like models. It will also appear that there is a master problem here.

Master Problem. For each k there exists an l such that $(k+l)$-pushdown store machines are more powerful in real-time than k-pushdown store machines.

> If we find a simple proof for this master problem (as opposed to the complicated proofs [Aa, PSS] for seemingly stronger problems) then this proof implies all real-time hierarchies by the simple argument below. The proofs in [Aa, PSS] have the appearance, possibly misleading, of the use of a sledge hammer to kill a fly. Thus, using the intricate arguments more effectively, [Pa] has strengthened the consequences of the methods in [Aa] to a nonlinear lower bound on the on-line simulation time required to simulate the effect of an additional tape.

For definitions of the considered devices we direct the reader to the references. For convenience, we consider the machines as transducers; the results can then be transferred to language recognizers at will. To lead up to the general statement we first discuss some lemma's. It will be at once apparent to the reader that (the proof of) Lemma 1 is a particular instance of a general argument applicable to virtually any transducer whatever. Lemma 2 is another such instance. Rather than confusing the issue by unnecessary definitions and formalisms we give representative instances and trust that the general case is self-evident.

Lemma 1. *Let k be an integer greater than 0. Either $(k+1)$-pushdown store machines are more powerful in real-time than k-pushdown store machines or, for all $i \geq 1$, $(k+i)$-pushdown store machines are equally powerful in real-time as k-pushdown store machines.*

Proofsketch. Suppose the Lemma is false for some k. Then there exists an l, l minimal and $l > k+1$, such that l-pushdown store machines are more powerful in real-time than k-pushdown store machines. Let M_l be any l-pushdown store machine. Decompose M_l in a $(l-1)$-pushdown store machine M_{l-1} and a separate pushdown store P. Multiplex the input of M_{l-1} with the current top symbol on P and the resulting output of M_{l-1} with the replacing new top string on P. More precisely, if M_l contains a transition

$$\begin{aligned}&(\text{input, state,topsymbol store } 1, \ldots, \text{topsymbol store } l)\\ &\qquad \to (\text{newstate, topstring store } 1, \ldots, \text{topstring store } l, \text{output})\end{aligned}$$

then M_{l-1} has the corresponding transition

$$\begin{aligned}&((\text{input, topsymbol store } l), \text{state,topsymbol store } 1, \ldots, \text{topsymbol store } l-1)\\ &\qquad \to (\text{newstate, topstring store } 1, \ldots, \text{topstring store } l-1, (\text{output, topstring store } l)) .\end{aligned}$$

In M_{l-1} we have an $(l-1)$-pushdown transducer which transforms transductions from the input (over a new input alphabet) to output (over a new output alphabet). By the contradictory assumption we can replace M_{l-1} by a k-pushdown transducer M_k performing the same transduction from the relevant strings in $(I \times T)^*$ into $(O \times T^*)^*$, where I is the input alphabet, T is the stack alphabet and O is the output alphabet of the original M_l. Replacing the transducer M_{l-1} by the transducer M_k, in the combination M_{l-1} and P, makes no difference. The resultant combination however, viewed as a transducer, is a $(k+1)$-pushdown transducer M_{k+1} performing the same transduction as the original M_l. Since by the contradictory assumption (viz., the minimality of l) it follows that $(k+1)$-transducers are equally powerful in real-time as k-transducers, we can replace M_{k+1} by a k-transducer M'_k, performing the same transduction as the original M_l, which yields the required contradiction. □ □

Corollary. If, for each k there is an l, $l > k$, such that l-pushdown machines are more powerful in real-time than k-pushdown machines, then $(k+1)$-pushdown machines are more powerful in real-time than k-pushdown machines, for each k.

Lemma 2. *Analogous results to Lemma 1 plus Corollary, with "-pushdown store machines" replaced by "-tape Turing machines", can be derived with M_{l-1} performing the obvious transduction from $(I \times T)^*$ into $(O \times T \times M)^*$ where I and O are as before, T is the tape alphabet and $M = \{\text{left}, \text{nomove}, \text{right}\}$.*

Theorem 2 below establishes a particular instance of a wide variety of equivalences between real-time hierarchies within the class of real-time definable computations.

Theorem 2. *If, for all $k \geqslant 0$ we can find an $l > k$ such that l-pushdown store machines are more powerful in real-time than k-pushdown store machines then $(t+1)$-tape Turing machines are more powerful in real-time than t-tape Turing machines for all $t > 0$. The same statement holds with "-pushdown store machines" and "-tape Turing machines" interchanged.*

Proof. It is obvious that by breaking each tape of a t-tape Turing machine around the head position we can simulate such machines by $2t$-pushdown store machines in real-time. If the condition in the Theorem is satisfied then $(2t+1)$-pushdown store machines are more powerful than $2t$-pushdown store machines by Lemma 1. The former, in their turn, are trivially simulatable in real-time by $(2t+1)$-tape Turing machines. So $(2t+1)$-tape Turing machines are more powerful in real-time than t-tape Turing machines, which gives the required result by Lemma 2. The second statement in the theorem follows because if l-tape Turing machines are more powerful in real-time than k-tape Turing machines $(l > k)$ then $2l$-pushdown store machines are more powerful than k-pushdown store machines in real-time, which gives the result by Lemma 1. □

The argument is quite general and is used for the proof below.

Proof of Theorem 1. Let it be established that there is no k such that $(k+l)$-pushdown store machines are equally powerful to k-pushdown store machines for all $l \geqslant 1$, e.g. [Aa]. Let a BLAH be a memory unit which can simulate a pushdown store in real-time (or such that a multiBLAH machine can real-time simulate a pushdown store) and can itself be simulated by a multitape Turing machine in real-time. If k-BLAH machines can be simulated in real-time by $f(\text{BLAH}, k)$-pushdown store machines then we can also assume that $f(\text{BLAH}, k)$ is minimal. By assumption, there is an $l_{\text{BLAH},k}$ such that $(f(\text{BLAH}, k) + l_{\text{BLAH},k})$-pushdown store machines are more powerful in real-time than $f(\text{BLAH}, k)$-pushdown store machines and therefore more powerful than k-BLAH machines. Also by assumption, a c-BLAH machine can real-time simulate a pushdown store, $c \geqslant 1$, so there is a minimal integer m such that m-BLAH machines are more powerful in real-time than k-BLAH machines. Either $m = k+1$ and we have established what we want or $m > k+1$. In the latter case a $(m-1)$-BLAH machine can be simulated in real-time by a k-BLAH machine and, following the method of proof of Lemma 1, we show that an m-BLAH machine can be real-time simulated by a k-BLAH machine: contradiction. Therefore, $(k+1)$-BLAH machines are more powerful in real-time than k-BLAH machines. □

> The situation is slightly more general. If we have a transducer of type X, which can be real-time simulated by a multitape Turing machine transducer, and we plug in an extra memory unit of type BLAH satisfying the conditions of Theorem 1, then we obtain a new transducer type Y which is more powerful in real-time than transducers of type X.

It follows from the above that the unsatisfactory complicated proofs for the real-time tape hierarchy in [Aa, PSS] may possibly be replaced by a proof for the fact that for no k we have that $(k+l)$-pushdown store machines are equally powerful to k-pushdown store machines for all $l \geqslant 1$. This is the *master* problem for the real-time hierarchies and finding a neat proof for it would simplify a great deal.

Different tape architectures and computation modes. The main result established is Theorem 1 which follows, in the realm of real-time definable computations, from [Aa] together with Lemma 1. We like to point out, however, that the principle enunciated in Lemma 1 has a far larger scope. The argument, and the Lemma, seems to hold for all types of transducers. Thus, like intuition tells us, the real-time computation hierarchies are *smooth*. For various reasons people like to consider tape *architectures* which are not linear lists but trees, more dimensional arrays or graphs. *Mutatis mutandis* Lemma 1 holds for each such class of machines too. A useful *computation mode* which is often considered is that of an *oblivious* computation. A computation is *oblivious* if the sequence of accessed storage cells is a fixed function of time, independent of the inputs to the machine. See e.g. [PF]. One of the attractive features of oblivious Turing machine computations is that they can be simulated by combinational logic networks at the cost in logic gates of the latter in the order of the time complexity of the former. Oblivious *real-time* computations translate in combinational logic networks with a response time of O(1) in between processing the i-th input at the i-th input port and producing the i-th output at the i-th output port, which enables the $i+1$-th input port. The oblivious real-time computations are the computations which can be performed by oblivious real-time multitape Turing machines. Notice that *linear* oblivious computations, that is, those performed by oblivious linear time multitape Turing machines, may translate in combinational logic networks with an *unbounded* response time. Other computation modes are *nondeterminism* or *alternation*. As a general, intuitively clear statement, Lemma 1 does hold for all BLAH-transducers in BUH mode, and not just for pushdown transducers in deterministic mode, using the same proof outline in each case. Thus, each transducer hierarchy either stops at some point or proceeds by proper inclusion according to computing power with *each* added unit.

REFERENCES

Aa Aanderaa, S.O., On k-tape versus $(k-1)$-tape real-time computation. In: *SIAM-AMS Proceedings, Vol. 7 (Complexity of Computation),* 1974, 75 - 96.

FMR Fischer, P.C., A.R. Meyer and A.L. Rosenberg, Real-time simulation of multihead tape-units, *J. Ass. Comp. Mach.* **19** (1972) 590 - 607.

FR Fischer, M.J., and A.L. Rosenberg, Limited random access Turing machines, Proceedings 9-th IEEE Conference on Switching and Automata Theory, 1968, 356 - 367.

Ko Kosaraju, S.R., Real-time simulation of concatenable double-ended queues by double-ended queues. Proceedings 11-th ACM Symposium on Theory of Computing, 1979, 346 - 351.

LS Leong, B., and J.I. Seiferas, New real-time simulations of multihead tape units, Proceedings 9-th ACM Symposium on Theory of Computing, 1977, 239 - 248.

Pa Paul, W.J., On-line simulation of $k+1$ tapes by k tapes requires nonlinear time. Proceedings 22-nd IEEE Conference on Foundations of Computer Science, 1982, 53 - 56.

PSS Paul, W.J., J.I. Seiferas and J. Simon, An information-theoretic approach to time bounds for on-line computation. Proceedings 12-th ACM Symposium on Theory of Computing, 1980, 357 -367.

PF Pippenger, N., and M.J. Fischer, Relations among complexity measures, *Journal ACM* **26** (1979) 361 - 384.

Vi Vitányi, P.M.B., On the power of real-time Turing machines under varying specifications. Proceedings of the 7-th International Colloquium on Automata, Languages and Programming, *Lecture Notes in Computer Science* **85**, Springer Verlag, Berlin, 1980, 658 - 671.