# Ultrafilter and non-standard Turing machines

Petrus H Potgieter
Department of Decision Sciences
University of South Africa (Pretoria)
&
Elemér E Rosinger
Department of Mathematics and Applied Mathematics
University of Pretoria

UC 2008 :: Vienna :: 2008-08-27

## Outline

We consider several kinds of non-finitary computation, using ordinary Turing machines, as usual, as the reference case.

We are interested in defining the output, or final message, of a machine which has run for a countably infinite number of steps.

Non-finitary machines are the work-horses of hypercomputation. We attemt to the question of how to deal with the machine and output description when we consider a realised non-finitary "computation".

Needless to say, this is just the problem of Thomson's lamp, which is

**on** at time $1 - 2^{2n}$ and

**off** at time $1 - 2^{2n+1}$.

## Convention and caveats

We shall consider only Turing machines with unbounded input and output tapes, with an unbounded working tape, over a finite alphabet.

Our approach bears only a superficial resemblance to the infinite time Turing machines introduced by Hamkins and Lewis (2000); Hamkins and Seabold (2001).

Logical difficulties for supertask machines arise, in our view, *not* from the identification of logical with computational time but were picked up in Hilbert's Non-Smoking Hotel.

Recall that a *filter* $\mathcal{F}$ on $X$ is a non-empty collection of subsets of $X$,

- being closed under the taking of finite intersections, and
- containing $B$ whenever $B \supseteq A \in \mathcal{F}$.

## A Turing machine “run”

A computation on a Turing machine $T$ can be described by a sequence of natural numbers

$$(k_n)_{n=0}^{\infty}$$

where each $k_i$ describes the condition – including the tapes, head position and internal state – after the $i$-th step in the computation. Call such a sequence a *run* of $T$.

The sequence $(k_n)_{n=0}^{\infty}$ is fully determined by $k_0$ and the rules that determine the machine $T$.

The input is, naturally, encoded by $k_0$ and if the input leads to an accepting computation, then the sequence is eventually constant.

## Accepting computations

Consider the Fréchet filter

$$\mathcal{F} = \{L \subseteq \mathbb{N} \mid \mathbb{N} \setminus L \text{ is finite}\}$$

consisting of all co-finite subsets of the natural numbers.

**Definition 1** *A sequence* $(k_n)_{n=0}^{\infty}$ *is an* accepting computation *whenever it is eventually constant, i.e. whenever it is constant on some element of the Fréchet filter* $\mathcal{F}$.

We have de-emphasised the notion of explicitly defined accepting state, without loss of generality in this recasting of the usual definition.

**Definition 2** *A sequence* $(k_n)_{n=0}^{\infty}$ *is a* $\mathcal{G}$-accepting computation *whenever it is constant on some element of* $\mathcal{G}$.

Definition 2 is especially interesting when we consider $\mathcal{U}$-accepting computations, where $\mathcal{U}$ is an *ultrafilter*.

## Ultrafilter-accepting computations

Recall that an *ultrafilter* on $X$ is a filter with the property that for each $A \subseteq X$, either $A$ or its complement belongs to the filter.

Subject to the Axiom of Choice, there exist *ultrafilters* $\mathcal{U}$ on $\mathbb{N}$ which contain $\mathcal{F}$. Choose one such $\mathcal{U}$.

The subsets belonging to a specific filter are often seen as the *large* sets. The $\mathcal{F}$-large sets are the cofinite subsets of $\mathbb{N}$.

The $\mathcal{U}$-large sets therefore generalise of the cofinite sets.

Furthermore, $\mathcal{U}$-accepting computation generalises $\mathcal{F}$-accepting computation in a clean and obvious way.

**Accepting computation** There exists an $\mathcal{F}$-large set of points in time where the tape content as well as the internal state of the machine remain constant.

**$\mathcal{U}$-accepting computation** There exists a $\mathcal{U}$-large set of points in time where the tape content as well as the internal state of the machine remain constant.

Consider a machine $T_{\mathrm{TL}}$ à la Thomson (1955):

- $T_{\mathrm{TL}}$ has alphabet $\{-1, 1\}$;
- at time $n$ writes $(-1)^n$ to the first position of the output tape and
- has a minimal number of states.

Any run of this machine is a $\mathcal{U}$-accepting computation since either the set of odd points in time, or the set of even points, belongs to $\mathcal{U}$ and is equivalent to a machine outputting a constant bit on the tape.

## $T_{\mathrm{TL}}$ and $T_{\mathrm{ATL}}$

Consider also a $T_{\mathrm{ATL}}$ which at time $n$ writes $(-1)^{n+1}$ to the first position of the output tape and has a minimal number of states.

Clearly the "output" of $T_{\mathrm{ATL}}$ is also either $+1$ or $-1$, depending on the choice of $\mathcal{U}$.

Furthermore, considered as $\mathcal{U}$-accepting computations, the machines $T_{\mathrm{ATL}}$ and $T_{\mathrm{ATL}}$ have opposite outputs.

It is not extraordinarily liberal to consider a run of $T_{\mathrm{TL}}$ or of $T_{\mathrm{ATL}}$ an accepting computation since these machines simply oscillate between two global states.

## Comparison to the Hamkins approach

Hamkins and Lewis (2000) define the state of $T_{\mathrm{TL}}$ at the first limit ordinal $\omega$ and – if $\limsup$ has been chosen as the limiting operator for each cell of the tape – the content of the tape "at" the ordinal $\omega$ is $+1$.

Moreover, in the approach of Hamkins and Lewis (2000) the tape of $T_{\mathrm{ATL}}$ would *also* contain $+1$ at ordinal $\omega$.

In our approach, we do not know whether a run of $T_{\mathrm{TL}}$, being a $\mathcal{U}$-accepting computation, will "output" $+1$ or $-1$.

Howeber, considered as $\mathcal{U}$-accepting computations, the machines $T_{\mathrm{ATL}}$ and $T_{\mathrm{ATL}}$ have opposite outputs.

Should one, or should one note, distinguish between $T_{\mathrm{ATL}}$ and $T_{\mathrm{ATL}}$?

## All $\mathcal{G}$-accepting computations cycle

**Proposition 1** *If a filter $\mathcal{G} \supsetneq \mathcal{F}$ then every $\mathcal{G}$-accepting computation is either*

*(i) an $\mathcal{F}$-accepting computation, i.e. a usual accepting computation; or*

*(ii) a computation that ends in a finite cycle of global states of the machine, in the fashion of Thomson's lamp.*

This elementary result follows directly from the internal dynamics of the Turing machine and the requirement that this also stabilise on the "large" filter set.

## Ultrafilter machines

One can use the filters $\mathcal{F}$ and $\mathcal{U}$ to describe two further notions for machines with a dedicated output tape.

**Limit computation** For each $\mathcal{F}$-small set of positions on the output tape, there exists a $\mathcal{F}$-large set of points in time where those positions on the output tape do not change.

**$\mathcal{G}$-limit computation** For each $\mathcal{G}$-small set of positions on the output tape, there exists a $\mathcal{G}$-large set of points in time where those positions on the output tape do not change.

**Ultrafilter computation** There exists a $\mathcal{U}$-large set of points in time where the output tape is constant.

Ultrafilter computation still avoids some pathologies like the undefinability of the output of the Thomson's lamp machine $T_{\mathrm{TL}}$.

## A further toy example

Consider a Turing machine $T_{\mathrm{d}}$, operating with the alphabet $\{-1, 1\}$:

```
write -1 on the tape up to position 98;
n = 0;
while 1 > 0 do
     write "+1" on position 99;
     go back and write "-1" on position 99;
     write "-1" in the 2^n positions to
         the right of 99;
     move back to position 99;
     n = n + 1;
end while;
```

$T_{\mathrm{d}}$ has neither an accepting computation, nor a $\mathcal{U}$-accepting computation. It is also not limit-computable or ultrafilter computable.

## Non-standard Turing machine outputs

In this part simply define the *sequence* of output tape content, at each discrete moment in time, as the *output* of the machine.

If $p(k)$ denotes the content of the output tape of a machine when its global description is $k$ then for each run $(k_n)$ of the machine, $(p(k_n))$ will be the sequence of output tape contents.

For the classically accepting computations, we identify the output sequence $(p(k_n))$ – which will be constant after a finite number of terms – with the limit of the sequence, which is exactly the classical output of the machine.

If $\mathcal{U}$ is the ultrafilter discussed earlier, we proceed to use the notions of non-standard analysis.

## Equivalence classes of outputs

**Definition 3** *For each sequence of natural number $(a_n)$ we set*

$$(a_n)_{\mathcal{U}} = \{(b_n) | \{m | a_m = b_m\} \in \mathcal{U}\}$$

*which is the equivalence class of all sequences that agree with $(a_n)$ on a $\mathcal{U}$-large set of indices.*

**Theorem 1** *If a run $(a_n)$ of a classical Turing machine $T$ is an accepting computation with output $k$ then*

*(i) for some $\ell \in \mathbb{N}$ we have $(a_n)_{\mathcal{U}} = \ell$; and*

*(ii) $(p(a_n))_{\mathcal{U}} = k$.*

Here $k$ is also the equivalence class of a constant sequence.

## Non-standard inputs/outputs

We can now see the output of a Turing-type machine as a, possibly infinite, non-standard natural number $(c_n)_{\mathcal{U}}$ where $c_n = p(k_n)$ for some run $(k_n)$ of the machine.

The *input* of the machine can also be made a non-standard natural number. Suppose $(a_i)$ is a sequence of natural numbers and let $(k^i_n)$ denote a run of Turing machine $T$ on input $a_i$ – in the classical sense. We simpy *define* a run of $T$ on $(a_i)$ to be the sequence $(k^i_i)$.

**Remark 1** *If $(a_n)_{\mathcal{U}} = (b_n)_{\mathcal{U}}$ and $(k_n)$ and $(\ell_n)$ are runs of $T$ on the two respective non-standard numbers, then*

$$(k_n)_{\mathcal{U}} = (\ell_n)_{\mathcal{U}}.$$

**Definition 4** *The output of $T$ on input $(a_n)_{\mathcal{U}}$ is the class $(p(k_n))_{\mathcal{U}}$ where $(k_n)$ is a run of $T$ on $(a_n)$.*

## NSTMs are unconventional

Within this framework the halting problem for ordinary Turing machines can be solved by a machine that outputs $(1)_{\mathcal{U}}$ if the machine halts, and $(0)_{\mathcal{U}}$ otherwise.

It is clear, of course, what the concatenation of two NST machines would compute as the output of an NST is *always* a valid input for another NSTM.

How this concatenation would be implemented on an NSTM – and whether this would be possible at all – is not so clear.

The non-standard approach to computability has been investigated before, i.a. by Richter and Szabo (1988).

## Conclusion

We have explored – modestly and briefly – how the filter and ultra-filter concepts can be used to characterise the behaviour of certain non-classical computation schemes based on Turing machines.

A fully non-standard scheme w.r.t. the input, output and run length is proposed as only one way to overcome the problem of defining the output or final global state of the machine.

The authors regard this as a tentative proposal with for extending the vocabulary of hypercomputation by accelerated Turing machines.

## References

Hamkins, J. D. and Lewis, A. (2000). Infinite time turing machines. *The Journal of Symbolic Logic*, 65:567–604.

Hamkins, J. D. and Seabold, D. E. (2001). Infinite time turing machines with only one tape. *MLQ. Mathematical Logic Quarterly*, 47:271–287.

Richter, M. M. and Szabo, M. E. (1988). Nonstandard methods in combinatorics and theoretical computer science. *Polish Academy of Sciences. Institute of Philosophy and Sociology. Studia Logica. An International Journal for Symbolic Logic*, 47:181–191.

Thomson, J. (1954–1955). Tasks and Super-Tasks. *Analysis*, 15:1–13.