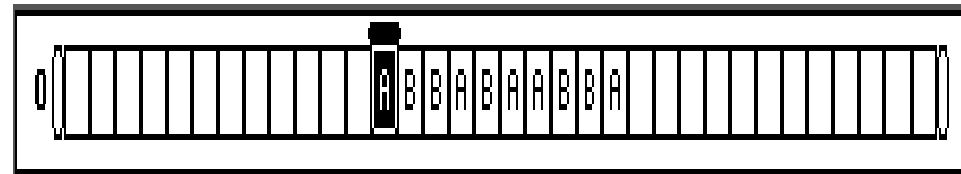


Beyond the Turing Limit?



Topics

- These notes present in part the work of Hava Siegelmann. Her research deals with the computing power of (neural) networks, resp. of analog computation generally.
- Also the super-computability of oracle and some non-standard machines is presented and compared to standard Turing-computation.
- Is there computation *beyond the Turing Limit*? What about the *Church-Turing Thesis*?

Analog Computation –
Beyond the Turing Limit?

Computation

- A computable function f is a rule between objects which can be explicitly presented by *finite* means (usually [coded by] natural numbers) that specifies how to get the second element from the first.
- If the objects are built from a finite stock and $\text{range}(f)$ is binary, f is a *characteristic function* and $\text{domain}(f)$ is a *language*.

Analog Computation –
Beyond the Turing Limit?

Noncomputable Functions

- A noncomputable function is an infinite set of ordered pairs for which no ‘reasonable’ rule can be provided. ‘reasonable’ obviously depends on an intuitive notion of computation or algorithm.
- A function can be uncomputable because there aren’t any resources (reasonable or not) to express a rule of pairing.

Analog Computation –
Beyond the Turing Limit?

Automata

- Automata are *theoretical machines*.
- Automata that cannot be built by our finite resources are *notional* machines only.
- One type of automata is *stronger* than another if the first can compute a set of functions the second cannot compute.
- The concept *machine* should be kept apart from the concepts *algorithm* and *finite*.

Analog Computation –
Beyond the Turing Limit?

Automata Described

- Abstractly an automaton is quintuple

$$M = \langle Q, I, O, f, h \rangle$$

consisting of the sets of states Q , the input space I , output space O , and the functions

$f: Q \times I \rightarrow Q$ (the next-state map) and
 $h: Q \rightarrow O$ (the output map)

Automata Described (II)

- Let Σ be a set of possible input letters, then $I = \Sigma \cup \{\$ \}$, “\$” designating the end of a string. Σ^* is the set of words. Any input $x \in \Sigma^*$. $\varepsilon \in \Sigma$. $\Sigma^+ = \Sigma \setminus \{\varepsilon\}$. $\Sigma^{\leq n} = \bigcup_{k \leq n} \Sigma^k$.
- $x(t)$ designates the state of all variables at t , $i(t)$ summarises the content of the inputs at t , so that $x(t+1) = f(x(t), i(t))$
- $y(t)$ is the total output at t , $y(t) = h(x(t))$

Complexity

- Automata depend on time and space resources. (We only consider time resources here.)
- With each x of the domain of computation we associate a measure of length $|x|$.
- We then define a partial function $T:N \rightarrow N$. $T(|x|)$ is defined only when the computation of all domain elements of this length halts.

Analog Computation –
Beyond the Turing Limit?

Complexity (II)

- A machine M *computes in time* T , if for all inputs x for which $T(|x|)$ is defined, M halts after performing not more than $T(|x|)$ steps of computation.
- We get *time complexity classes*. The class $A\text{-Time}(T(n))$ consists of all functions that are computed by some automaton in A in time $T(n)$. (“ A ” refers to some class of automata.)

Analog Computation –
Beyond the Turing Limit?

Turing Machines

- The standard automaton of computation is the (deterministic) Turing machine (TM).
- The input is binary. Q is finite. At each step the symbol “a” under the head is read, the state is checked and one of the elementary operations (moving [$m \in \{L, R\}$], writing [$b \in \{0, 1, \varepsilon\}$], changing control state [$q' \in Q$]) is executed. Transitions are described by a **function** $g(a, q) = (b, m, q')$. If the control reaches the halting state, the machine stops, the binary sequence up to the first “ ε ” right of the head being the output.

Analog Computation –
Beyond the Turing Limit?

Universal Turing Machines

- TMs can be arithmetically encoded.
- TMs are countable (say by a standard numbering τ). $\tau(i)$ is the i^{th} TM, $\varphi_{\tau(i)}$ the function computed by it.
- The existence of a universal TM M which computes the function $\varphi(i,x) = \varphi_{\tau(i)}(x)$ is an existence theorem for a *general purpose digital computer*.

Analog Computation –
Beyond the Turing Limit?

Nondeterministic TMs

- A nondeterministic TM (NTM) differs from a deterministic TM (DTM) in that in at least one step of the computation there is a *choice* regarding what to write, how to move and to proceed to which state.
- NTMs have a transition *relation*.
- An input is accepted by a NTM if it is accepted by *some* transition.

Analog Computation –
Beyond the Turing Limit?

DTM and NTM

- *Theorem*

There is a DTM which computes f
 $\Leftrightarrow$ there is a NTM which computes f .

- Nondeterministic TM *are not stronger* than DTM. (“ $\Leftarrow$ ” is the interesting result.)

- But nondeterministic TMs are more efficient than deterministic TMs.

(By using choices they can ‘guess’ results.)

Efficient Computation

- An *efficient computation* is defined as one that requires polynomial time on a DTM. This complexity class is **P** (halting on x in time $c|x|^k$ for some constants c, k).
- **EXP** is the class of functions taking exponential computation time on DTMs.
- Although *solving* some problem might be in **EXP**, *recognising a solution* can be in **P**.

Analog Computation –
Beyond the Turing Limit?

Efficient Computation (II)

- **NP** (*Nondeterministic Polynomial time*) is the class of functions computed by a NTM in polynomial time.
- Some functions *supposedly* in **EXP** are in **NP**. It's open whether **P** = **NP**.
- Comparing TMs to other types of computation one can consider the *speed up* even if no new functions are computed.

Analog Computation –
Beyond the Turing Limit?

Non TM-Computable Functions

- Most functions cannot be computed by TMs (either DTMs or NTMs).
- There are only countably many TMs but uncountably many functions $f:\mathbb{N}\rightarrow\mathbb{N}$.
- Irrational numbers cannot be finitely represented, so cannot be a finite input.
- Some problems can be proven to be unsolvable (e.g. the Halting Problem [HP]).

Analog Computation –
Beyond the Turing Limit?

The Church-Turing Thesis

- The *Church-Turing Thesis* (CT) can be expressed in different ways, for example:
(CT') Everything that is computable is recursive(ly computable).
(CT'') Everything that is computable is computable by a Turing machine.
- The first expression refers to the recursive *functions*, the second to Turing *machines*.

Analog Computation –
Beyond the Turing Limit?

The Church-Turing Thesis (II)

- The *Church-Turing Thesis* is generally seen as outlining an upper limit on computability. Nothing seems to be computable that isn't Turing computable.
- Seen from a naive point of view –the naivety of which will be explained shortly– the existence of super-computability would falsify the Church-Turing Thesis.

Analog Computation –
Beyond the Turing Limit?

Effective Steps

- (CT) identifies the *intuitive* notion of computability with a *formally explicated* notion (being computable by a TM).
- The cornerstone of this is the idea of an *algorithm* M on *discrete* symbols ω :
 - executing steps each of which is mindless,
 - where each computation of $M(\omega)$ ends after finitely many steps (if $M(\omega)$ is defined at all),
 - M is implementable by different devices.

Analog Computation –
Beyond the Turing Limit?

The Church-Turing Thesis (II)

- To avoid misunderstanding CT therefore can be expressed properly as:

(CT) Everything that is intuitively computable is computable by a deterministic Turing machine.

- CT is not talking about any computability that could be nor about any type of machine that could be – keep that in mind!

Analog Computation –
Beyond the Turing Limit?

Aspects Of (CT)

- There are at least three aspects of (CT):
 - (i) reconstruction: DTM is said to characterise the *intuitive* notion of computability
 - (ii) conceptual: by proving equivalences between formal models of algorithms we grasp an *absolute* concept of computation
 - (iii) epistemological: (CT) outlines (some) structures and limits of the *mind*.
(being a bridge to meta-logical results)

Analog Computation –
Beyond the Turing Limit?

Aspects Of (CT) (II)

- So, controversies about (CT) and super-computability can be seen as having immediate effects in the philosophy of mind.
- Since the brain –which needn't be identical to the mind (!)– is said to be an analog net we have to consider the nets' computing power.

Analog Computation –
Beyond the Turing Limit?

Computation?

- So (CT) is phrased in terms of the standard notions of computability (i.e., recursive functions or Turing machines). These define steps of computation which are discrete and effective (e.g. writing a symbol, moving one to the left).
- But couldn't there be a different concept of computation altogether?

Analog Computation –
Beyond the Turing Limit?

The Turing Limit

- Asking for the Turing limit means asking whether there are other concepts of computability and whether a machine corresponding to such a concept of computability might compute functions which cannot be computed by a deterministic TM.
- If there is such a machine, it is *beyond* the Turing limit!

Analog Computation –
Beyond the Turing Limit?

Three Questions

We can ask at least three questions now:

(1) What other types of computability are there, if there are any.

And with respect to some such type:

(2) Does the existence of this type of computability put (CT) in doubt or even falsify it?

(3) Can there *really be* such machines?

Analog Computation –
Beyond the Turing Limit?

A Physical Church-Turing Thesis

- The last question asks whether some type of theoretical machine could ever be built.
- David Deutsch introduced a physical interpretation of the Church-Turing Thesis:

(DCT) No *realisable physical device* can compute functions that aren't TM-computable.

Analog Computation –
Beyond the Turing Limit?

Probabilistic Turing Machines

Probabilistic TMs (PTM) work as follows:

- every step of the computation can have two outcomes one with *probability* p , the other with probability $1-p$
- all computations on the same input require the same number of steps
- every computation ends with *reject* or *accept*.

Analog Computation –
Beyond the Turing Limit?

Probabilistic Turing Machines (II)

- The error probability of a PTM M is the function $e_M(x)$ defined by the ratio of computations on input x resulting with the wrong answer to the total number of computations on x (which is $2^{T(|x|)}$).
- **PP** is the class of languages accepted by polynomial PTMs with $e_M < 1/2$. **BPP** resp. the class of languages accepted by PTMs in polynomial time whose error probability is bounded by some positive $c < 1/2$.

Analog Computation –
Beyond the Turing Limit?

PTM and DTM

- *Theorem:* (p the used probability in a computation step)

If p is rational, then a language is recognised by a PTM iff it is recognised by a DTM.

- That means PTMs working with *rational* probabilities *are not stronger* than DTMs.
- What if p can take any real value? This will be discussed talking about stochastic networks corresponding to such PTMs.

Analog Computation –
Beyond the Turing Limit?

Oracle Machines

- An oracle machine (OTM) is a TM supplemented by a further move:
looking up for any input x whether for a given set A : $x \in A$ or $x \notin A$.
- The set A being any set, an OTM can decide *any* set! (How it is done does not matter.)
- OTMs can be counted; there is a complexity measure on OTM computations.

Analog Computation –
Beyond the Turing Limit?

Relativized Computation

- Given an OTM we can introduce the functions that are computable *relative to* an OTM including an oracle on a set A .
- A function $f:\mathbb{N}\rightarrow\mathbb{N}$ is A -computable iff there is some oracle computable function $g:2^{\mathbb{N}}\times\mathbb{N}\rightarrow\mathbb{N}$ such that $g(A,x)=f(x)$, $x\in\mathbb{N}$.
[OTMs can be used to introduce a preorder of *Turing reducibility*, the equivalence classes of which are *Turing degrees*. Turing reducibility is weaker than reducibility.]

Analog Computation –
Beyond the Turing Limit?

Oracles and DTM

- *Theorem:*

If the set A of an OTM M is recursive φ_M can be computed by a DTM M' . If the set A of an OTM M is non-recursive, there is no DTM M' such that $\varphi_M = \varphi_{M'}$.

- So OTM is really *stronger* than DTM.

OTMs solve HP, decide First Order Logic
(The Turing degree of a nonrecursive, r.e. set A contains sets which aren't even r.e.)

Analog Computation –
Beyond the Turing Limit?

Do Oracles Matter?

- Oracles are stronger than DTMs. They are *beyond* the Turing Limit. But they are mysterious – by definition.

(ad Q1) OTM defines a new type of computation. (In fact a hierarchy of degrees.)

(ad Q2) Are oracles related in any way to the claim of (CT)?

At least one author has claimed that:

Analog Computation –
Beyond the Turing Limit?

Copeland's 'Fallacy'

Jack Copeland has argued against a “wide spread” *Church-Turing fallacy* (a supposed misuse of (CT)). Polemics and Copeland's sophistic reading of Church and Turing aside the first of the central questions is:

(a) What is meant by “machine” (in CT)?

If (CT) was the claim that any machine could be simulated by a DTM it would be false. (Since oracles can't.)

But that's definitely not the claim made!

Analog Computation –
Beyond the Turing Limit?

Copeland's 'Fallacy' (II)

(CT) deals only with machines that are algorithmic (and therefore intuitively computable) Oracles *aren't* – by definition.

(b) What does *mechanism* claim?

If mechanism only claimed the mind to be *some* machine, the mind could be an oracle! To give mechanism some bite it has to claim that the mind is computable.

Analog Computation –
Beyond the Turing Limit?

The Oracle's Upshot

What can we learn from Copeland's attack? We should keep in mind:

- (1) (CT) is falsified only if some of the functions which are computed by a new type of computation are *intuitively* computable.

Note: That something can be *expressed finitely* doesn't make it *effective*!

(e.g. "Divide this by the largest prime.")

Analog Computation –
Beyond the Turing Limit?

The Oracle's Upshot (II)

- (2) If analog machines aren't machines in the sense of (CT), we could try to express an analog to (CT) for this kind of device. (something Siegelmann does)
- (3) If analog machines aren't machines in the sense of (CT) we have to reconsider mechanism only if we consider what they compute to be part of the mind (in distinction to the brain).

Analog Computation –
Beyond the Turing Limit?

Mind And Brain

- **Note:** Even if the brain was of a non-standard computability type and further on that type was beyond the Turing Limit, this would *not* falsify (CT) or narrow mechanism, since the mind needn't be the brain.
- Maybe the brain is an analog device that *implements on some level* a TM. (cf. Dennett)
- If the analog processes aren't *cognitively penetrable* they needn't belong to the mind.

Analog Computation –
Beyond the Turing Limit?

What Happened In Delphi?

Now – are there any oracles? Otherwise OTMs would be notional machines only.

If there are irrational quantities in nature there could be a quantity $0,101010001\dots$ coding the Halting function! If some mechanism M could measure this quantity to *any* specified number, M is an oracle.

But even that doesn't mean that M could be *built* by us – (DCT) needn't be violated.

Analog Computation –
Beyond the Turing Limit?

More Nonfinite Machines

- There are other machine models that are beyond the power of the (universal) Turing machine, e.g. *Coupled Turing Machines* or Copeland's *Accumulator Machines*.
- But they don't have finite input, so are beyond what is intuitively computable.
- These machines are notional machines.
- Networks on the other hand are real – but what can they do?

Analog Computation –
Beyond the Turing Limit?

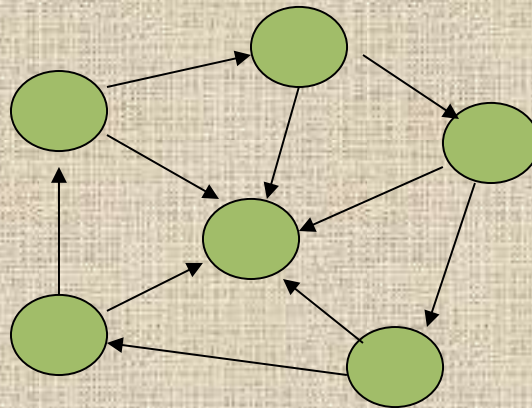
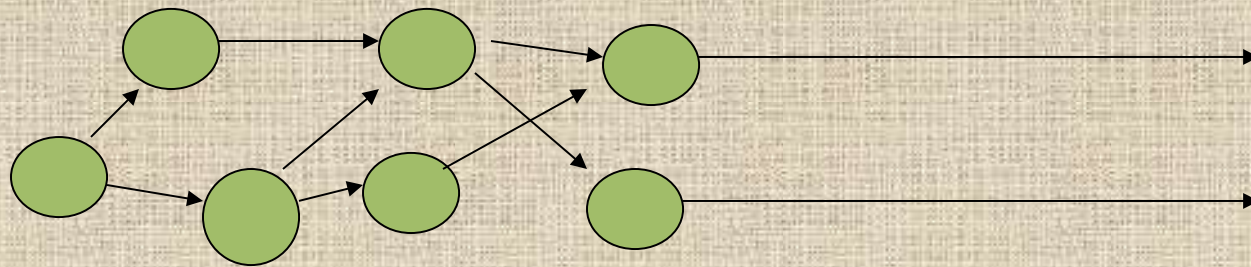
Neural Networks

- Neural Networks are interconnections of simple processors or *neurons*. The directed connections are characterized by a positive or negative real (a *weight*)
- Each neuron computes a scalar function of its weighted inputs (the *response*).
- The output of each neuron is sent to other neurons in the network.

Analog Computation –
Beyond the Turing Limit?

Feedforward and Recurrent

- Feedforward and Recurrent Network



Analog Computation –
Beyond the Turing Limit?

Recurrent Networks

- Feedforward networks are arranged in multiple layers, such that the output of one layer is the input of another layer.
- Their computation ends in a fixed number of steps.
- Recurrent networks on the other hand allow loops and thus memory of past computations can be present in the network

Analog Computation –
Beyond the Turing Limit?

Coding A Language In A Net

- Output neurons take binary values only.
- Input arrives on two binary input lines: the data line D and the validation line V (indicating whether the data line is active).
So: $u(t) = (D(t), V(t)) \in \{0, 1\}^2$ for each t
- There are two output processors: G for data and H for validation
- Now we encode each input string ω :

Analog Computation –
Beyond the Turing Limit?

Coding A Language In A Net (II)

Encode the word $\omega = \omega_1 \dots \omega_k \in \{0,1\}^+$ by

$$u_\omega(t) = (D_\omega(t), V_\omega(t)), \quad t \in \mathbb{N}$$

where

$$D_\omega(t) = \begin{cases} \omega_k & \text{if } t=1 \dots k \\ 0 & \text{otherwise} \end{cases}$$

and

$$V_\omega(t) = \begin{cases} 1 & \text{if } t=1 \dots k \\ 0 & \text{otherwise} \end{cases}$$

(Remember: ω is binary coded previously.)
Analog Computation –
Beyond the Turing Limit?

Classifying A Word

- A word $\omega \in \{0,1\}^+$ is *classified in time r* by a formal net starting from the initial state $x(1)=0$ if the input lines (D_ω, V_ω) take the values $D_\omega = \omega 0^\infty$ and $V_\omega = 1^{|\omega|} 0^\infty$, and the output line $H_\omega(t)=0$ for $t < r$ and $H_\omega(r)=1$.
- If $G_\omega(r)=1$ then ω is *accepted*, if $G_\omega(r)=0$ ω is *rejected*.

Analog Computation –
Beyond the Turing Limit?

Accepting A Language

- A language $L \subseteq \{0,1\}^+$ is *accepted* by a formal network N if every $\omega \in L$ is accepted by N and every $\omega \notin L$ is rejected or not classified by N .
- L is *decided* by N if L is accepted by N and its complement is rejected by N .
- L is *decided in time T* by N if $\omega \in \{0,1\}^+$ is correctly classified in time $t < T(|\omega|)$.

Computing A Function

Let $\psi: \{0,1\}^+ \rightarrow \{0,1\}^+$ be a partial function and N be a formal net with input (D,V) and output (G,H) . ψ is *computable by N* if for every $\omega \in \{0,1\}^+$:

1. If $\psi(\omega)$ is undefined, then $H_\omega = 0^\infty$
2. If $\psi(\omega)$ is defined, then $(\exists r \in \mathbb{N})$ (*response time*) such that:

$$G(t) = \begin{cases} \psi(\omega)[t-r+1] & \text{if } t=r, \dots, (r+|\psi(\omega)|-1) \\ 0 & \text{otherwise} \end{cases}$$

$$H(t) = \begin{cases} 1 & \text{if } t=r, \dots, (r+|\psi(\omega)|-1) \\ 0 & \text{otherwise} \end{cases}$$

Analog Computation –
Beyond the Turing Limit?

Computing A Function (II)

- The response time is the time the net needs to start delivering output (bit by bit).
- The function $\psi(\omega)$ is *computable in time* T if for every $n \in \mathbb{N}$ $T(n)$ is defined iff $\psi(\omega)$ is defined on all inputs of length n , and there is a net N such that for every $\omega \in \{0,1\}^+$: if $\psi(\omega)$ is defined, the response time is at most $T(|\omega|)$.

Analog Computation –
Beyond the Turing Limit?

Non-Finite Networks

- If the response function is binary a *finite network* cannot accomplish more complex computations than a *finite automation* (i.e. it can compute only *regular* languages).

(McCulloch introduced such networks in the 1940s.)

- So either we need *infinite* networks or the activation functions must allow for *non-binary* – maybe even continuous – *values*.

Analog Computation –
Beyond the Turing Limit?

Response in Recurrent Nets

- Recurrent Nets (RN) have n neurons, the i^{th} has an activation value $x_i(t)$ at t , there are m external binary inputs u_i and the connections carry weights a_{ij} , b_{ij} , c_{ij} .
- The dynamics is a map (*net function*):

$$F: \mathbb{R}^n \times \{0,1\}^m \rightarrow \mathbb{R}^n$$

component-wise, with *response* σ :

$$x_i(t+1) = (\sum^n (a_{ij} x_j(t)) + \sum^m (b_{ij} u_i(t)) + c_i)$$

Analog Computation –
Beyond the Turing Limit?

Networks And Determinism

- These networks don't work like NTMs or PTMs. Once weights, the output processors and σ have been specified, the behaviour is completely defined for a input coding.
- A net is *nondetermined* if it additionally contains a *guess line* (guessing a symbol like a NTM), accepting x if *some* computation accepts x .

Analog Computation –
Beyond the Turing Limit?

Real Weights In Recurrent Nets

- The weights of the nets may assume different kinds of values. If the weights can assume arbitrary *real* values we have *Real-weighted Recurrent Networks* (RRN).
- RRN correspond to *Advice Turing Machines* (ATMs).

(So before we continue talking about different kinds of recurrent networks we take a look at the computing power of ATMs.)

Analog Computation –
Beyond the Turing Limit?

Advice Turing Machines

- An ATM is an extended TM. It is a TM supplemented by a further input tape with *advice* on the input which is computed at the moment.
- All inputs ω of some length $|\omega|$ have the same advice sequent.
- Let $\omega \in A \subseteq \Sigma^*$ and $v: \mathbb{N} \rightarrow \Sigma^*$ be the advice function. Then $v(|\omega|)$ is the advice.

ATM Complexity

- The length of the advice is bounded as a function of the input and measures the level of noncomputability.
- Given a class of languages C , bounding functions h_i in H , that $L_i \in C/H$ means that language L_i can be computed in complexity class C given advice computed by $h_i \in H$.
- **P/poly** is accepted by ATMs with polynomial advice computing in polynomial time.

Analog Computation –
Beyond the Turing Limit?

ATM Nonuniformity

- Be $A_v = \{\omega \$ v(|\omega|) | \omega \in A\}$ the input of an ATM. A_v can be translated back given homomorphism h^* , so $\langle \omega, v(|\omega|) \rangle \in \Sigma^*$. So $A \in C/H \Leftrightarrow v \in H \wedge h^*(A_v) \in C$.
- In a *prefix nonuniformity class* (Pref-C/H) advice $v(|n|)$ must be useful for all input up to n , ($v(|n_1|)$ is the prefix of $v(|n_2|)$ for $n_1 < n_2$).

$$A'_v = \{\omega_1 \$ v(|\omega|) | \omega, \omega_1 \in A \wedge |\omega_1| < |\omega|\}$$

Analog Computation –
Beyond the Turing Limit?

ATM Super-Computability

- *Theorem:*

If *exponential* advice is allowed to an ATM any language L is computable.

Proof: L is decided by fixing the advice for an input x_i of length n to have “1” in location i (given locations $1, 2, \dots, 2^n$.) iff $x_i \in L$.

- Obviously ATMs are beyond the Turing Limit.

Analog Computation –
Beyond the Turing Limit?

ATM And DTM

- Let α be the advice of an ATM. If α cannot be generated from a finite rule (an algorithm) call α „nonuniform“.

- *Theorem:*

If the advice α of an ATM M is nonuniform M accepts sets that are not accepted by any DTM (i.e. are not computable).

An ATM Hierarchy

- The Kolmogorov complexity of a string α is the length of the shortest string γ which as input of a universal TM computes α .
- Resource bounded Kolmogorov complexity is constrained by the amount of time and advice in a computation (e.g. $K[\log, P]$).
- Increasing complexity we get a hierarchy of computational classes. ($K[\log, P] \subseteq K[\text{poly}, P]$)

Analog Computation –
Beyond the Turing Limit?

RRN And ATM

- *Theorem:*

$$\text{RRN} = \text{ATM}$$

- *Proof* (Outline):

Let the concatenation of the advice be $r = v(1) v(2) \dots$ and $\delta(r)$ a Cantor encoding of r (an irrational). Fix $\delta(r)$ as a weight of net N . Subnet N_1 receives input ω , measures $|\omega|$, and retrieves $v(|\omega|)$ from $\delta(r)$. ω and $v(|\omega|)$ are output for the other subnet N_2 which simulates a 2 tape TM on that finite input.

The decisive step is being able to fix the real weight $\delta(r)$.

Analog Computation –
Beyond the Turing Limit?

RRN Computing Power

- Since RRNs are equivalent to ATMs they compute functions that aren't Turing computable. RRN are *beyond* the DTM limit.
- If exponential computing time is allowed, one can specify a RRN for each binary language: The class of languages recognised by a RRN in exponential time, $\text{NET}_R(\text{EXP})$, includes *all discrete language*.

Analog Computation –
Beyond the Turing Limit?

Real Weights And (DCT)

- The weights of RRNs assume arbitrary real values. So it is practically impossible to *built* a net having such weights (this requires *infinite precision*). Updates also require infinite precision.
- There might *be* such nets (if there are irra-tional quantities in nature) but—at least—we cannot realise them. (DCT) is still in force!

Analog Computation –
Beyond the Turing Limit?

Variants Of Recurrent Nets

- RRNs have to be considered, therefore, as ideal, *notional* machines. Variants that don't rely on infinite precision are *boun-ded precision neurons* and *stochastic nets*.
- Changing the response function σ to a linear or treshold function results in a *different computational model*.

(As mentioned: Nets with integer weights compute the regular languages over the alphabet $\{0,1\}$ [Kleene 1956].)

Analog Computation –
Beyond the Turing Limit?

Linear Precision

- RRN are defined by unbounded precision.
- *Linear precision* means that for up to q steps of computation *only* the first $O(q)$ bits of both the weights and attraction values (of the neurons) influence the result.
(O being a function giving the complexity boundary as in O -notation for space and time complexity measures.)
- For time bounded computation only bounded precision is required.

Analog Computation –
Beyond the Turing Limit?

Linear Precision (II)

- If a network computes σ in time $T(n)$ the *$T(n)$ -truncated version* computes the same response function σ on any input of length n (i.e. a truncated result.)
- Truncated values aren't irrational values:

Theorem:

Computation within linear precision
is not beyond DTM-computation.

Computable Real Weights

- Some irrationals are step-wise computable. *Recursive reals* are those the decimal expansion of which is the characteristic sequence of a recursive set.
- *Theorem:*
Networks having recursive real weights compute only DTM-computable functions.
- But even these reals give us speed up:

Analog Computation –
Beyond the Turing Limit?

Computable Real Weights (II)

- *Theorem:*

In polynomial time RRNs with recursive weights compute $P/\text{poly} \cap \text{REC}$.

(REC is the set of recursively computable languages.)

- Since $P \subset P/\text{poly} \cap \text{REC}$ these networks provide *speed up* without the use of non-recursive weights, although they *aren't* computationally stronger than DTM.

Stochastic Networks

- A stochastic network (SRN) has additional input (stochastic lines). For all $t > 0$ the stochastic line l_i has value 1 with probability p_i .
- The number of steps in all computations on ω is the same. (cf. PTM)
- The final decision on ω considers the ratio of *rejects* and *accepts* on ω .
- The probabilities are *real values* in $[0,1]$.

Analog Computation –
Beyond the Turing Limit?

Stochastic Language Recognition

- A language $L \subseteq \{0,1\}^+$ is ε -recognised in time T by a stochastic network N if every word $\omega \in \{0,1\}^+$ is classified by every computation path of N in time $T(|\omega|)$ with bounded error probability $e_N(\omega) < \varepsilon < 1/2$.
- So the decision on ω takes into account the ratio of *rejects* and *accepts* with an error probability of the whole computation.

Analog Computation –
Beyond the Turing Limit?

Stochastic Computing Power

- SRNs have real probabilities, but their weights can be integers, rationals or reals.
- *Theorem*: SRNs with integer weights are not computationally stronger than deterministic nets with integer weights (i.e. are *below* the Turing Limit, computing regular languages only).
- *Theorem*: Stochasticity does *not* add to RRNs computing power or speed up.

Analog Computation –
Beyond the Turing Limit?

Stochastic Computing Power (II)

- So adding stochastic lines does not matter in case of integer weights as DFA = FA for finite automata.
- Adding stochastic lines does not matter in case of irrational real weights as RRN is already way beyond the Turing Limit.
- The interesting case are nets with rational weights, which have to be introduced first:

Analog Computation –
Beyond the Turing Limit?

Networks With Rational Weights

- Some nets (integer weights) are below, some (real weights) are beyond the Turing Limit. Are some equivalent to it?
- Consider nets in which the neurons take *countably different values*; so do the weights. These values can be represented by $\mathbb{Q}$ (at most). ($\mathbb{Q}$ being countable, of course.)
- They are *Rational Recurrent Nets* (QRN).

Analog Computation –
Beyond the Turing Limit?

QRN And DTM

- *Theorem:*

A QRN N is finitely describable and can be simulated by a DTM M .

- *Theorem:*

It is possible to specify a QRN N that simulates a universal TM (in real time).

- *Theorem:* L is recognised by QRN $N \Leftrightarrow$
 L is recognised by a DTM M .

QRN And DTM (II)

- Put shortly:
(T) $QRN = DTM = NTM = NQRN$
- Nondeterministic QRN (NQRN) are *not* stronger than deterministic QRN.
- (T) rather *supports* (CT) by a new equivalence proof w.r.t. a concept of computation.
- QRNs *aren't even faster* than TMs. There is no gain in complexity reduction.

Analog Computation –
Beyond the Turing Limit?

QRN With Stochastic Lines

- As seen stochastic lines don't matter with integer or irrational weights, but they do in case of QRN yielding a class SQRN.
- *Theorem:* QRN with stochastic lines compute $BPP/\log^*$.

(Bounded error Probabilitic Polynom)

- $BPP/\log^* \subset P/\text{poly}$, already contains *noncomputable f* .
- *Theorem:* $DTM \subset SQRN \subset ATM$.

Analog Computation –
Beyond the Turing Limit?

QRN With Stochastic Lines (II)

- So SQRN are *beyond* the Turing Limit.
- $BPP/\log^*$ is one class in the ATM-hierarchy. It presents a further concept of computability [answering our (Q1)].
- If the probabilities are *rational*s stochastic QRN compute BPP which *is* computable!
- Again: the stochastic lines *cannot be built* requiring *infinite* precision, as says (DCT).

Analog Computation –
Beyond the Turing Limit?

What Net Is The Brain?

In their book *The Computational Brain* Patricia Churchland and Terrence Sejnowski aren't precise what *kind* of computation occurs in the brain. On the one hand they claim it to execute vector mappings that are finite and deterministic. Being such it wouldn't be beyond the Turing Limit. On the other hand they consider continuous valued units. These might be RNNs.

Analog Computation –
Beyond the Turing Limit?

What Net Is The Brain? (II)

- If the brain at the neuronal level can be finitely specified, it isn't a RRN that is beyond the Turing Limit.
- If the brain is a RRN with at least one irrational weight it cannot be rebuilt by us.
- Must the brain master some task which requires such an RRN? – It seems difficult to say what type of network the brain is!

Analog Computation –
Beyond the Turing Limit?

Analog Computation

- (CT) deals with *algorithmic* computation.
- Natural process might be considered computational processes which are *continous*.
- Taken thus there is a class of *analog computations*. RRNs being *one type* of these.
- Continous/“chaotic“ processes in nature can only be *nonapproximately simulated* by RRNs or other devices beyond DTM.

Analog Computation –
Beyond the Turing Limit?

Siegelmann's Thesis

- With respect to analog/continuous computation Hava Siegelmann claims:

(ST) No possible abstract analog device can have more computational capabilities up to polynomial time than RRNs.

- This means there is an *absolute concept* of analog computation.

(As (CT) claims for algorithmic computation.)

Analog Computation –
Beyond the Turing Limit?

Siegelmann's Thesis (II)

- Has (ST) the same status as (CT)?
- On the one hand as (CT) it cannot be proven. Proven equivalences between abstract analog devices and RRNs give inductive evidence for it. (Siegelmann gives a proof w.r.t. the *analog shift map* [which is able to simulate an ATM with polynomial advice and is simulated by an RRN].)
- On the other hand (ST) doesn't seem to relate an intuitive concept with a formal one.

Analog Computation –
Beyond the Turing Limit?

Non-Standard Computation

- Besides networks and oracles there are other forms of non-standard computation: *molecular computation*, *cellular automata* (CA) or *quantum computation* (QC).
- Each of them represents a new type of computation. But their relationship to (CT) and standard computation is somewhat like in the case of (analog) networks.

Analog Computation –
Beyond the Turing Limit?

Quantum Computation And DTM

- *Theorem:*

A function f is computable by a DTM M
 $\Leftrightarrow f$ is computable by a QC Q .

- *Theorem:*

There are functions f such that there is
an *exponential gap* in *computing time*
between any DTM M and some QC Q .

[The speed up is due to using *superpositions* of bit states in *qubits* and working on them (simultaneously)!!]

Analog Computation –
Beyond the Turing Limit?

Is QC Realisable?

- So QC is not stronger but *much more efficient* than Turing-computation.
- Realising a QC is of interest since it would allow to crack encryption in polynomial time! Unfortunately it seems to be very difficult to realise a quantum computer. None has been built yet.

(Deutsch and others think its parallel computations occur in different universes!)

Analog Computation –
Beyond the Turing Limit?

Summary

- There are different types of computability some of which are *beyond* the Turing limit.
- There could be an *absolute* concept of analog computation (Siegelmann's Thesis).
- (CT) is not falsified by super-computability since that isn't algorithmic in the usual sense.
- (DCT) is not falsified since the machines with super-computability cannot be built.

Analog Computation –
Beyond the Turing Limit?

Summary (II)

- Even if the brain was a super-computer the mind need not be.
- Not only super-computability devices, but *also* Turing-computable networks provide significant speed up. You might speculate whether this is a reason to assume that the mind is realised in such a structure.

Analog Computation –
Beyond the Turing Limit?

Sources

- For the standard notions of *Turing machine*, *oracle machine* etc. see a textbook on computability, e.g. Weihrauch, Klaus. *Computability*. Berlin, 1987.
- An overview of Hava Siegelmann's work is her book *Neural Networks and Analog Computation*. Beyond the Turing Limit. Boston et. al., 1999.
- On quantum computation or cellular automata see Gramß, Tino et. al. *Non-Standard Computation*. Weinheim et. al. , 1998.
- On Copeland's attack see his papers in *The Journal of Philosophy* (2000), *Australasian Journal of Philosophy* (1999), *Analysis* (1998).

Analog Computation –
Beyond the Turing Limit?



Manuel Bremer