Non-Classical Turing Machines: Extending the Notion of Computation

Jiří Wiedermann
Institute of Computer Science, Prague
Czech Academy of Sciences

Jan van Leeuwen
Department of Information and Computing Sciences
Utrecht University, NL

Questions

1. Does the classical Turing machine paradigm suit to modern computing?
2. Are interactive computations more powerful than algorithms?
3. How can we make interactive computations more powerful?
4. What are the main characteristics of contemporary computing?
5. What computational models correspond to such computing?
6. Is it necessary to change the notion of computation?

Main message

Computations should no longer be seen as finite processes whose parameters are fixed before the start of processing.

Rather, computations are potentially infinite evolutionary processes whose parameters can change during the processing in an unpredictable way in interaction with their environment.

The Turing machine paradigm: Any algorithmic process can be simulated by a Turing machine

?

?

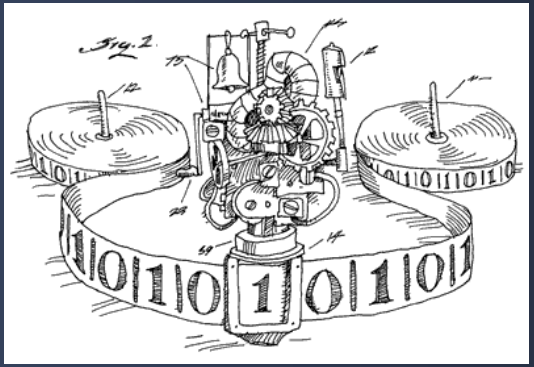
The Internet

Brain

Sensory nets

Cell

Ad-hoc nets



Turing machine

The Universe

Cognitive systems

DNA processing

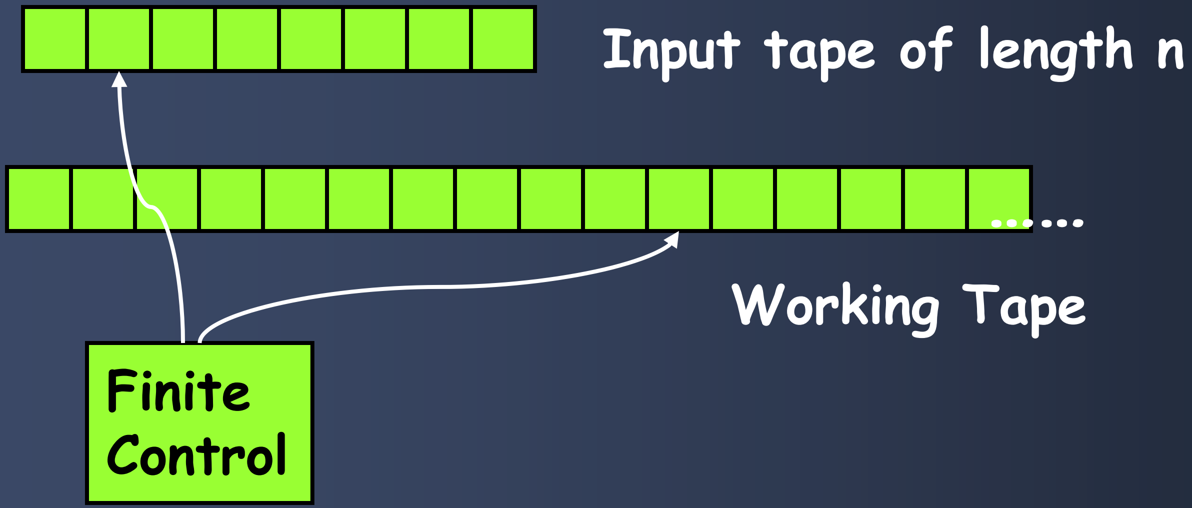
Relativistic Computing

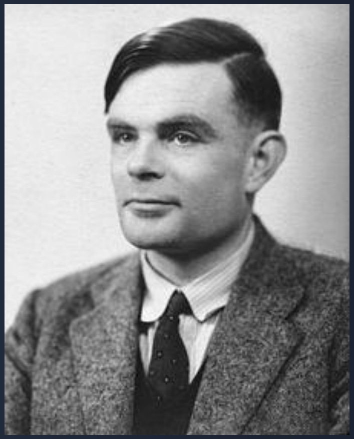
Artificial gadgets

“natural” computing

A classical view of computing

Classical Turing machine:



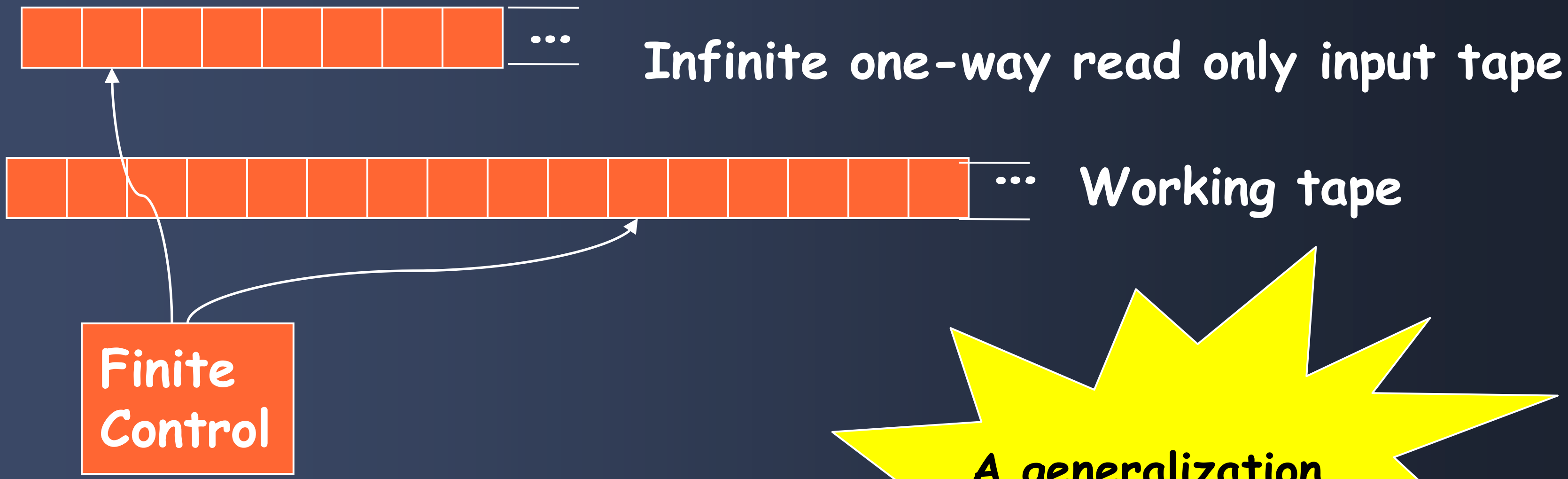


Computational scenario:

- finite input data present prior to the start of computation
- no new data added or changed during a computation
- the result only after the termination, in a finite time
- no data transferred to a future computation

“Always on” computing

ω - Turing machine:



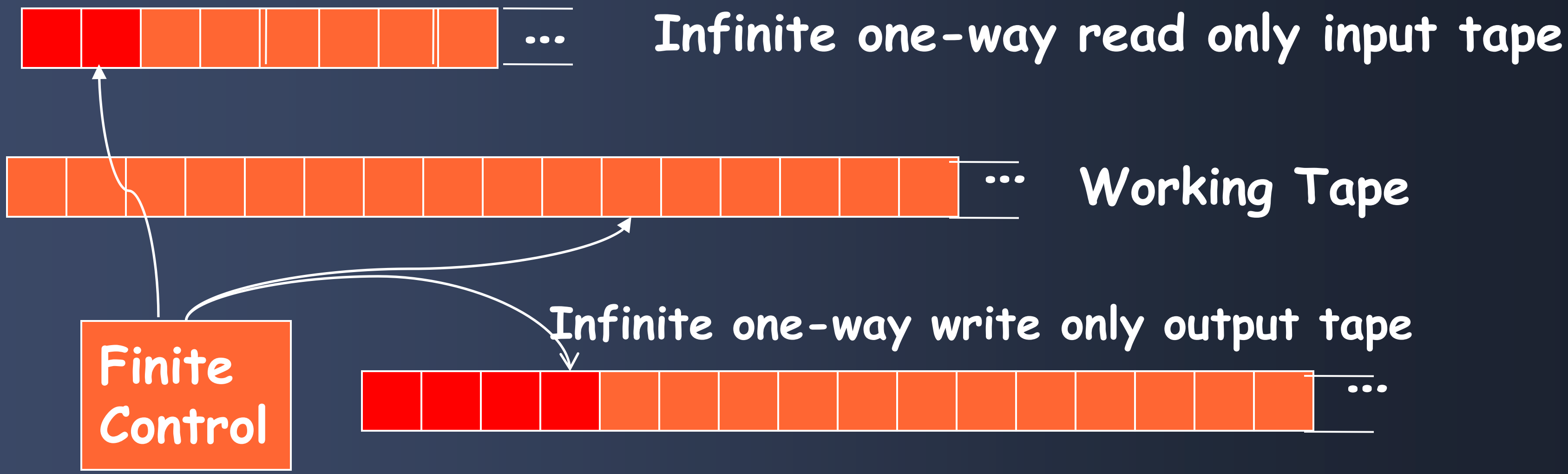
A generalization of Büchi automaton

Computational scenario:

- an infinite stream of input symbols
- a single infinite computation possibly entering and re-entering accepting states
- acceptance: an ω -machine accepts its infinite input iff it enters an accepting state infinitely many times

Interactive computing

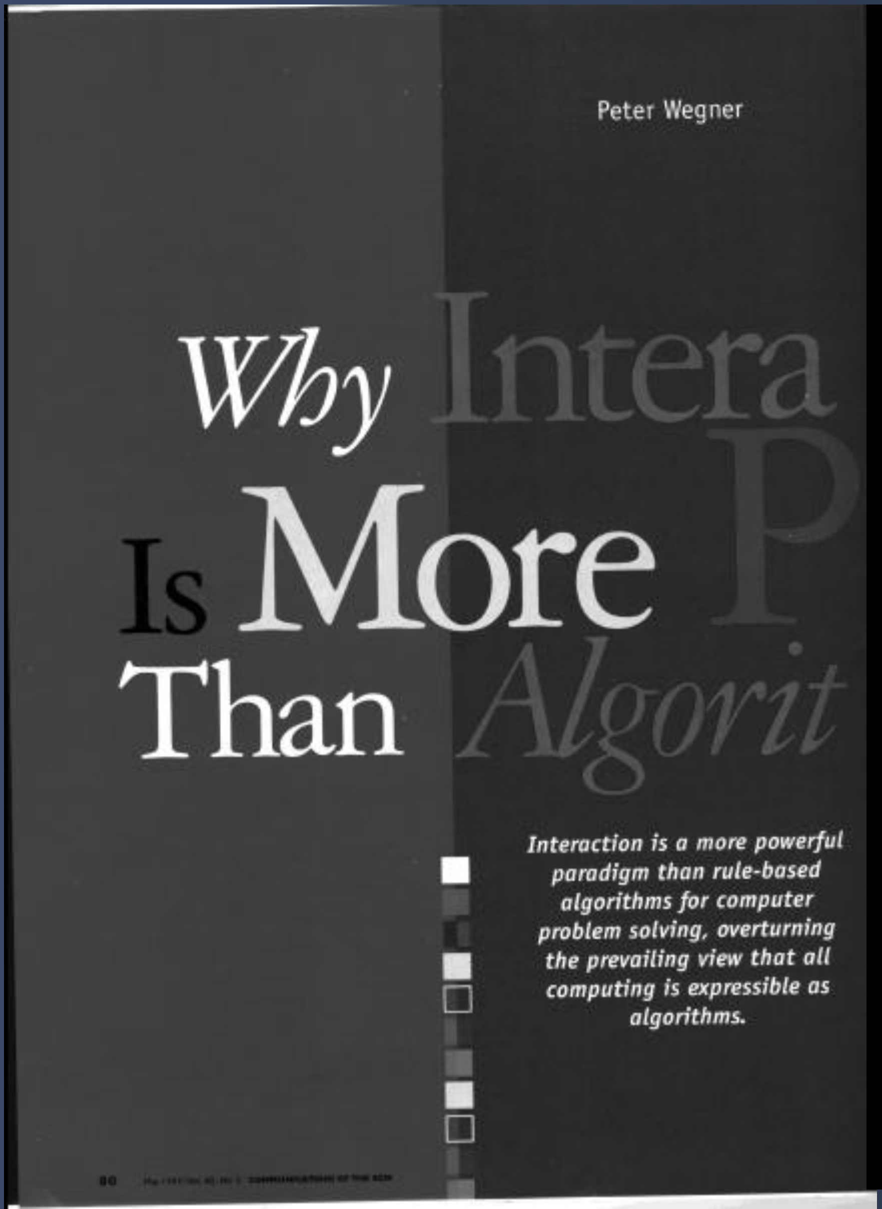
interactive Turing machine (JvL& 2001):

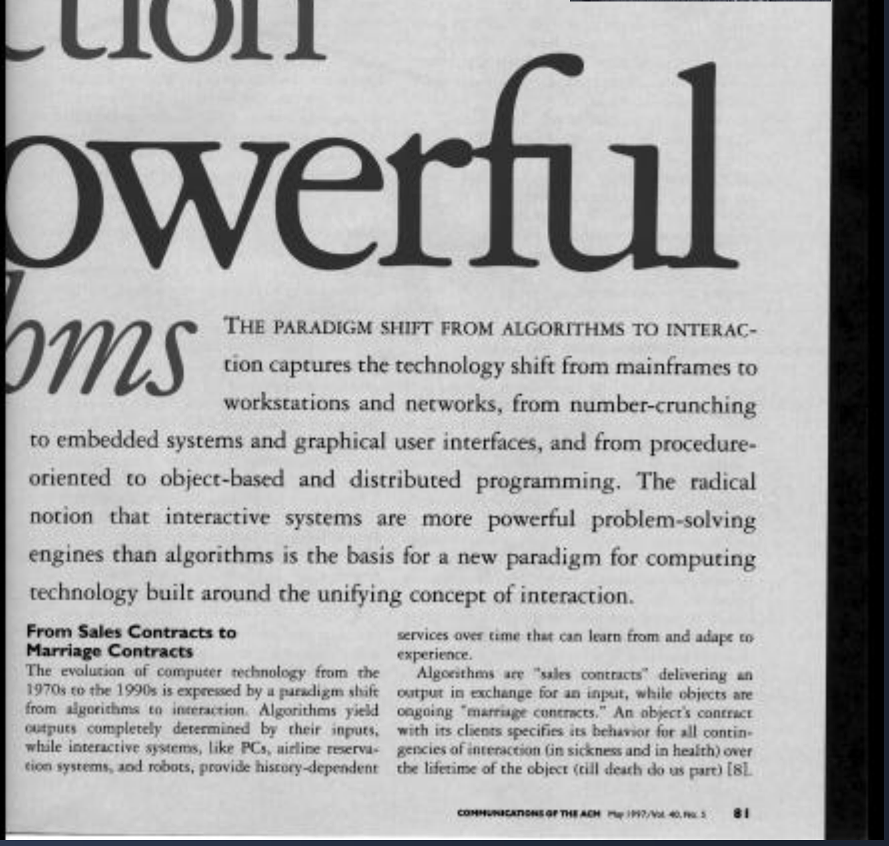

…
Infinite one-way read only input tape
…
Working Tape
Infinite one-way write only output tape
Finite Control
…

Computational scenario:

- a potentially infinite stream of finite inputs
- a sequence of individual computations
- data can be kept for future computations
- the result is an infinite sequence of results of individual computations
- in general, there is a dependence between the past outputs and the next inputs ("environmental feedback"): interaction

Is interaction really more powerful than algorithms?





Peter Wegner

Why Interaction Is More Powerful Than Algorithms

Interaction is a more powerful paradigm than rule-based algorithms for computer problem solving, overturning the prevailing view that all computing is expressible as algorithms.

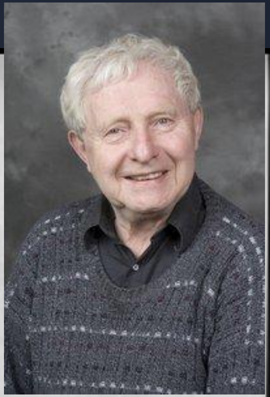
THE PARADIGM SHIFT FROM ALGORITHMS TO INTERACTION captures the technology shift from mainframes to workstations and networks, from number-crunching to embedded systems and graphical user interfaces, and from procedure-oriented to object-based and distributed programming. The radical notion that interactive systems are more powerful problem-solving engines than algorithms is the basis for a new paradigm for computing technology built around the unifying concept of interaction.

From Sales Contracts to Marriage Contracts

The evolution of computer technology from the 1970s to the 1990s is expressed by a paradigm shift from algorithms to interaction. Algorithms yield outputs completely determined by their inputs, while interactive systems, like PCs, airline reservation systems, and robots, provide history-dependent services over time that can learn from and adapt to experience.

Algorithms are "sales contracts" delivering an output in exchange for an input, while objects are ongoing "marriage contracts." An object's contract with its clients specifies its behavior for all contingencies of interaction (in sickness and in health) over the lifetime of the object (till death do us part) [8].

Peter Wegner
1932-2017



Interactive Turing machines

- they make use of the same machinery as classical TM do, but they compute under a different scenario;
- they compute translations of infinite streams to infinite streams;
- they are not "more powerful", they merely compute with different entities

---

How can we make computers "more powerful"? What computational resource must be added to interactive Turing machines in order to compute qualitatively "more" than the standard interactive Turing machine?

What about allowing a modification of their hardware or software in the course of their computation?

Modeling the evolution of finite interactive computing systems

Model 1 – hardware evolution m el

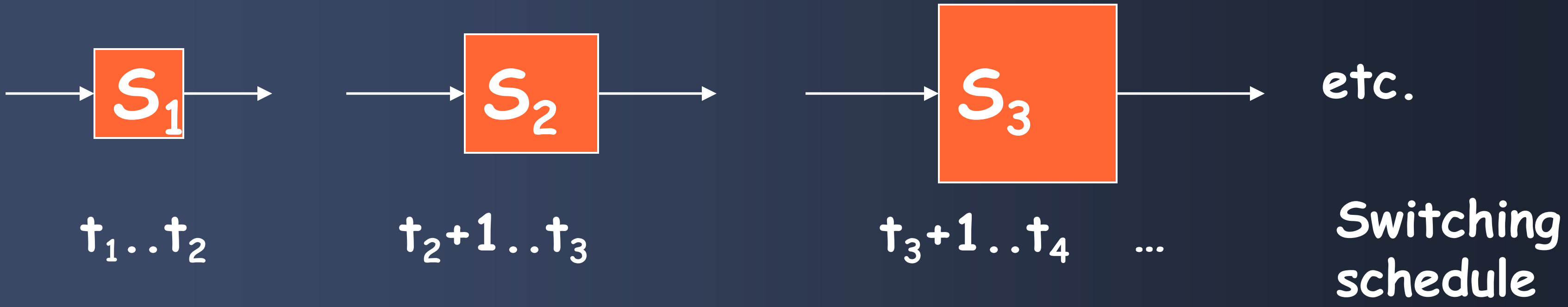
by a finit

Observation: in between a hardw changes of system's components, the s automaton

Modeling interactivness and non-uniform evolution by finite automata

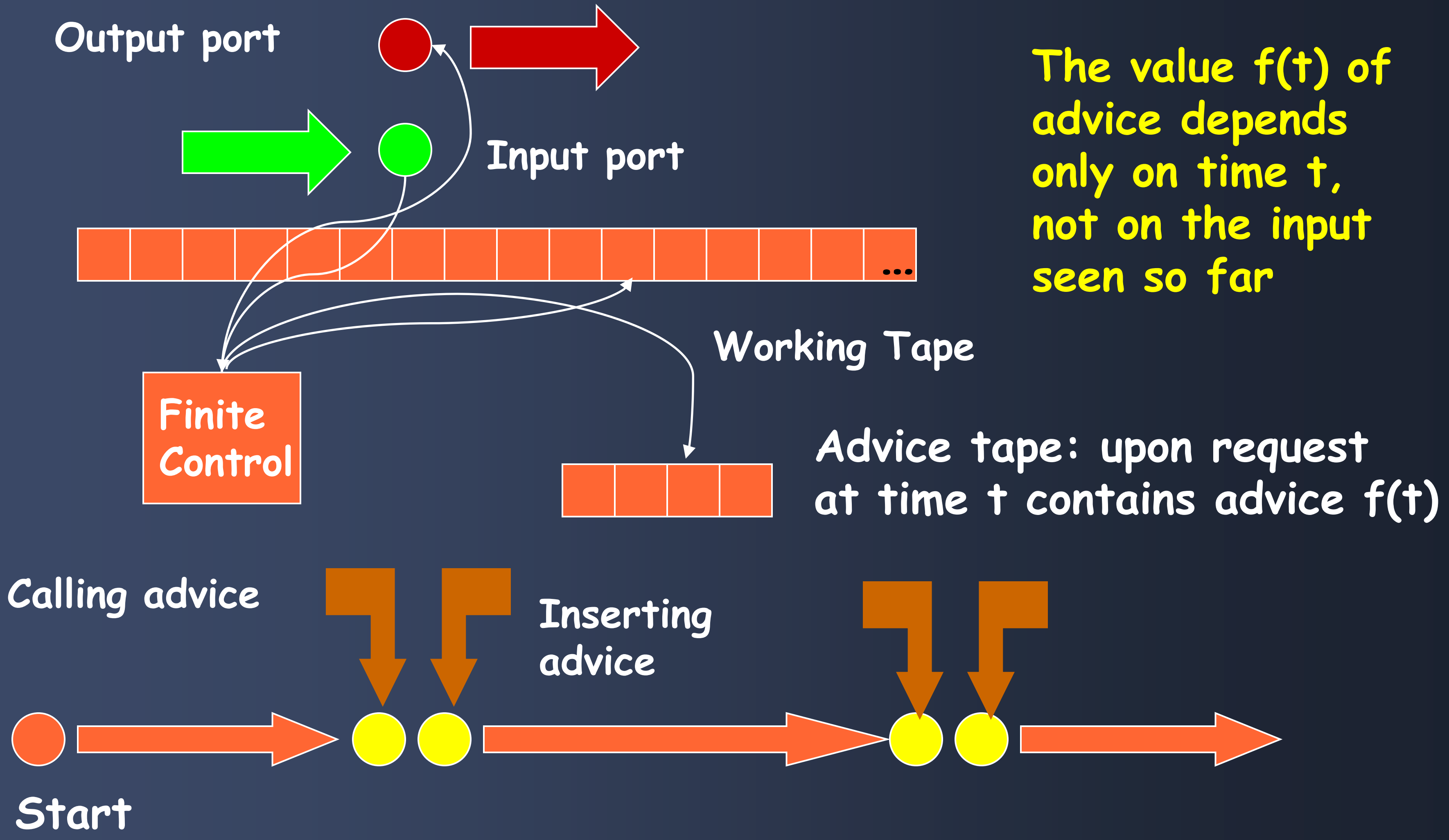
Definition: (van Leeuwen, Wied ton (EA) is a (non-computable) infinite ite automata with information tran c ers via states at times determined by E switchi sche

$Q_1 \subseteq Q_2 \subseteq Q_3 \subseteq \dots$ and $Q_i \subseteq S_i$ = {set of states of A_i}



Model 2: software evolution model

Computing with advice – interactive TM with advice (JvL&W 2001)



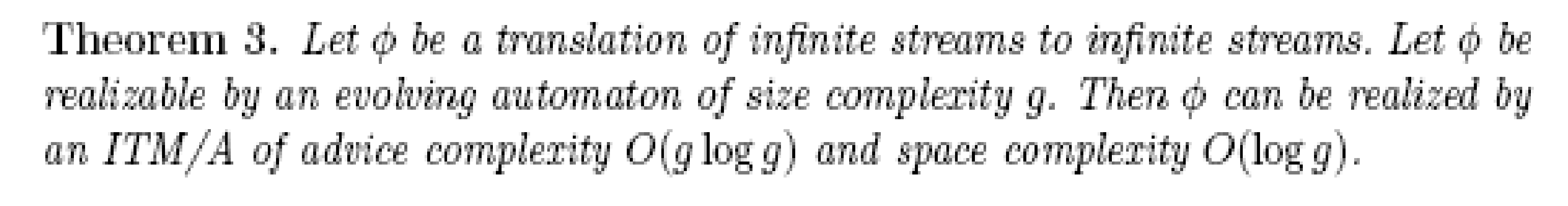
The value f(t) of advice depends only on time t, not on the input seen so far

Advice tape: upon request at time t contains advice f(t)

Basic complexity results

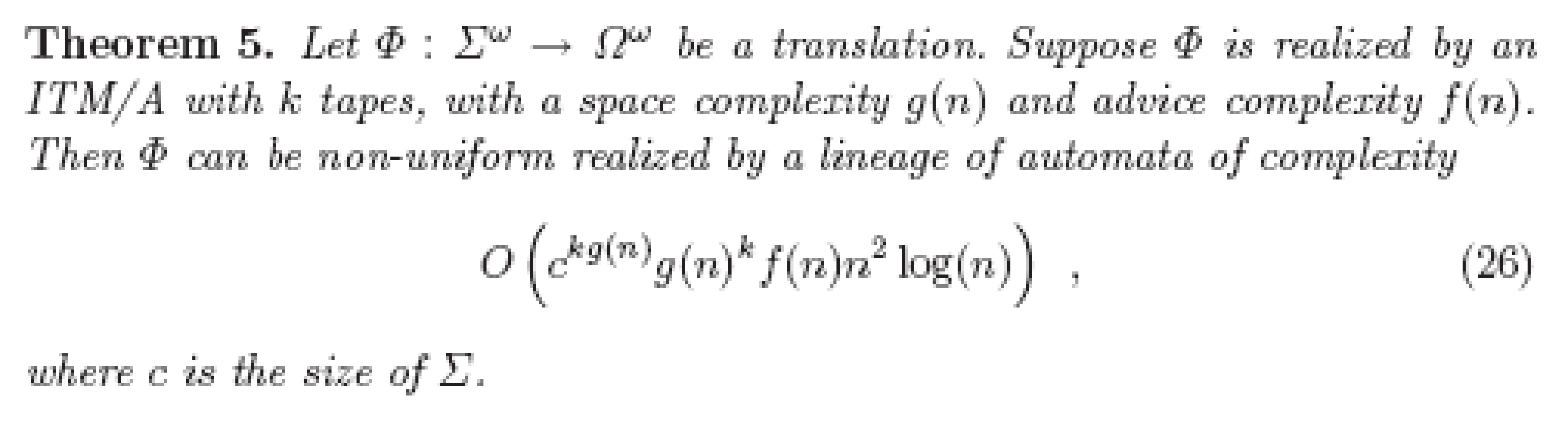
Model 1 and Model 2 are computationally equivalent:

1) EA $\subseteq$ ITM/A : use the description of EA as the advice for the ITM/A



Theorem 3. *Let ϕ be a translation of infinite streams to infinite streams. Let ϕ be realizable by an evolving automaton of size complexity g. Then ϕ can be realized by an ITM/A of advice complexity $O(g \log g)$ and space complexity $O(\log g)$.*

2) ITM/A $\subseteq$ EA: during periods of an ITM/A computations of changeless space complexity, with the same advice, this ATM/A is equivalent to a finite state automaton



Theorem 5. *Let $\Phi : \Sigma^\omega \rightarrow \Omega^\omega$ be a translation. Suppose Φ is realized by an ITM/A with k tapes, with a space complexity $g(n)$ and advice complexity $f(n)$. Then Φ can be non-uniform realized by a lineage of automata of complexity*

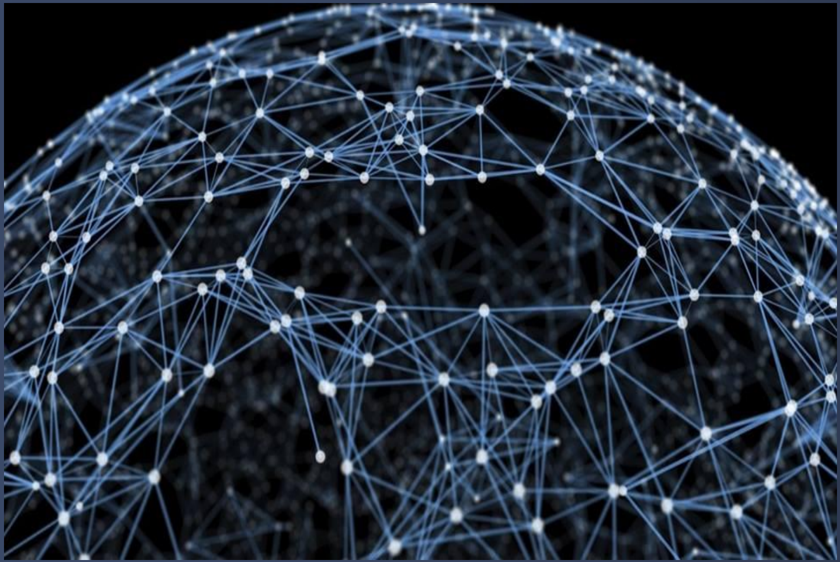
$$O\left(c^{kg(n)} g(n)^k f(n) n^2 \log(n)\right) \quad , \qquad (26)$$

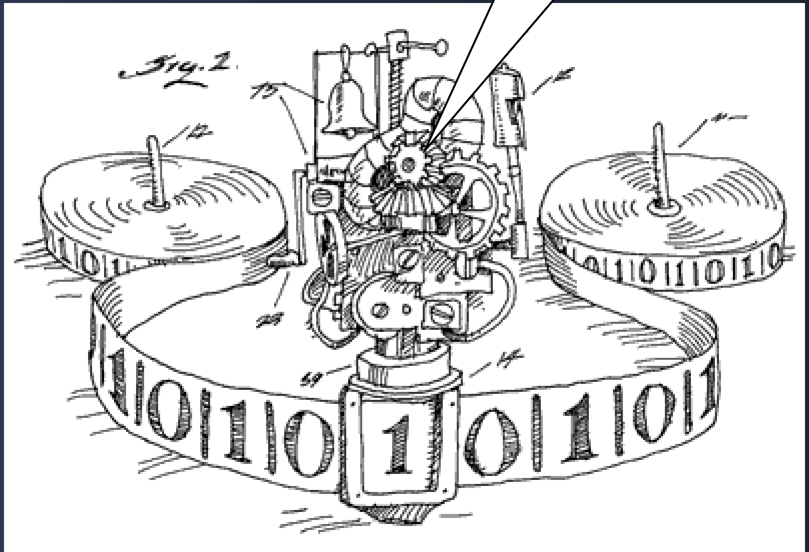
where c is the size of Σ.

A consequence:

- At each time the Internet can be seen as a giant finite automaton
- The computations of the Internet and its evolution over time can be captured by an evolving automaton, and hence,
- the Internet can be modeled by an ITM/A

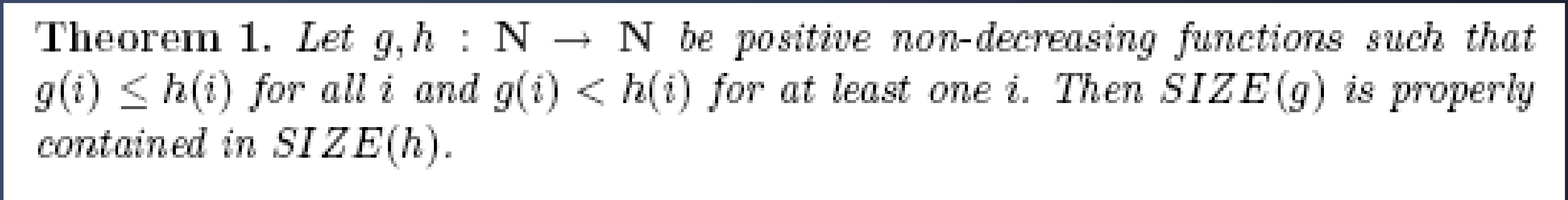
Advice





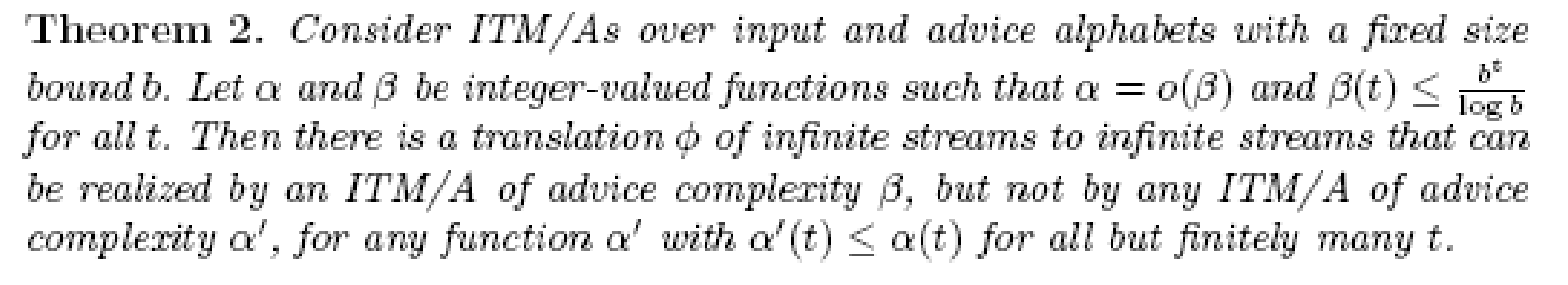
Hierarchies

Faster growing evolving automata compute strictly more then slower growing automata



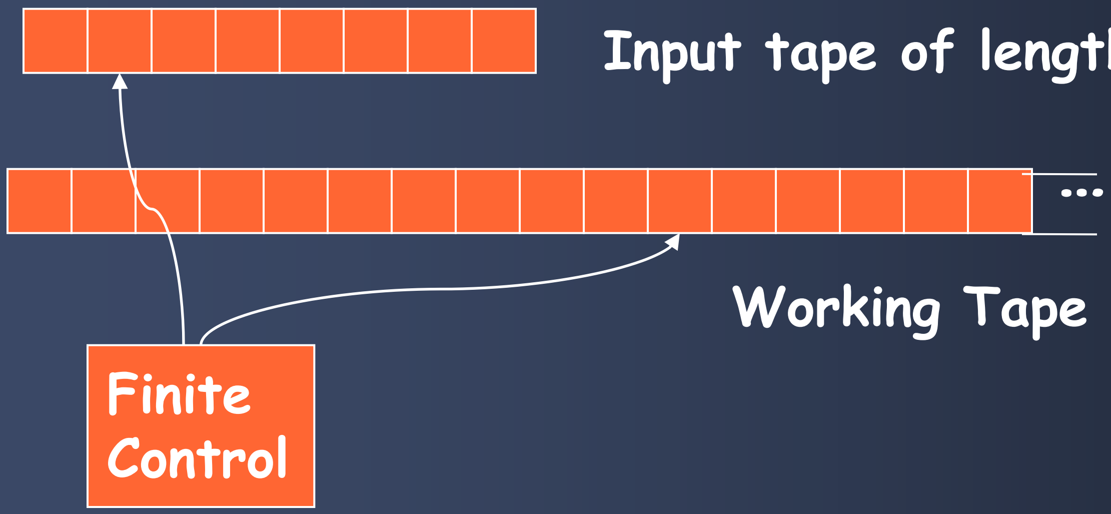
Theorem 1. *Let $g, h : \mathrm{N} \rightarrow \mathrm{N}$ be positive non-decreasing functions such that $g(i) \leq h(i)$ for all i and $g(i) < h(i)$ for at least one i. Then $SIZE(g)$ is properly contained in $SIZE(h)$.*

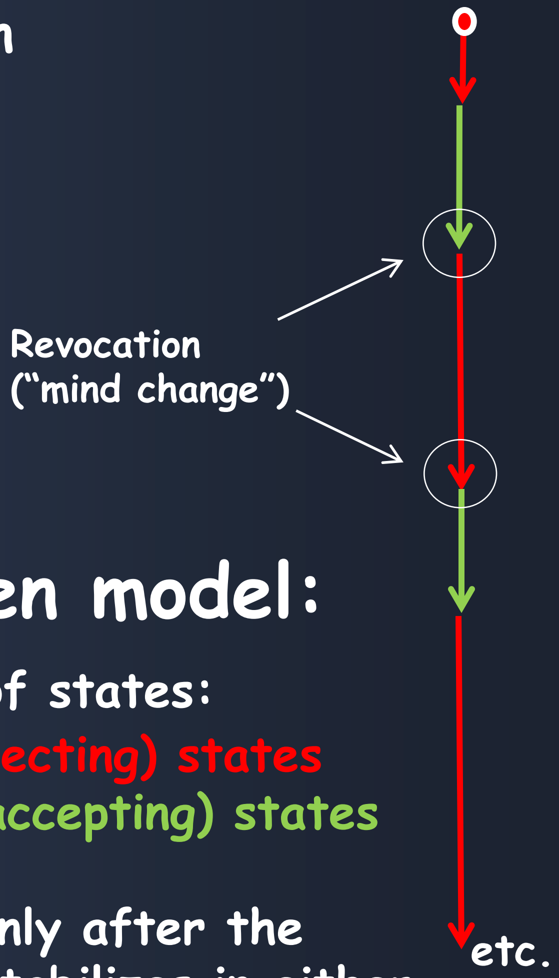
Interactive TMs with a faster growing advice function compute strictly more than ITMs with a slower growing advice function



Theorem 2. *Consider ITM/As over input and advice alphabets with a fixed size bound b. Let α and β be integer-valued functions such that $\alpha = o(\beta)$ and $\beta(t) \leq \frac{b^t}{\log b}$ for all t. Then there is a translation ϕ of infinite streams to infinite streams that can be realized by an ITM/A of advice complexity β, but not by any ITM/A of advice complexity α', for any function α' with $\alpha'(t) \leq \alpha(t)$ for all but finitely many t.*

Red-green Turing machines





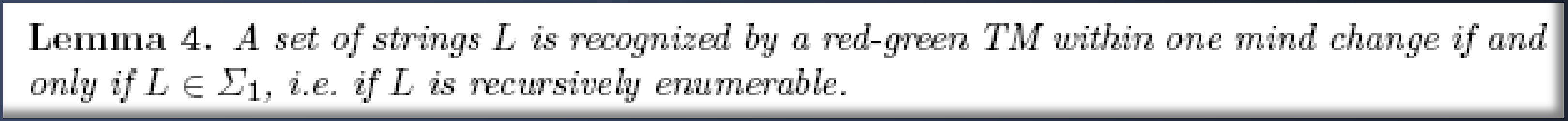
Classical model:

- two kinds of states:
  - working states
  - accept/reject state
- finite input
- the result only after the termination, in a finite time

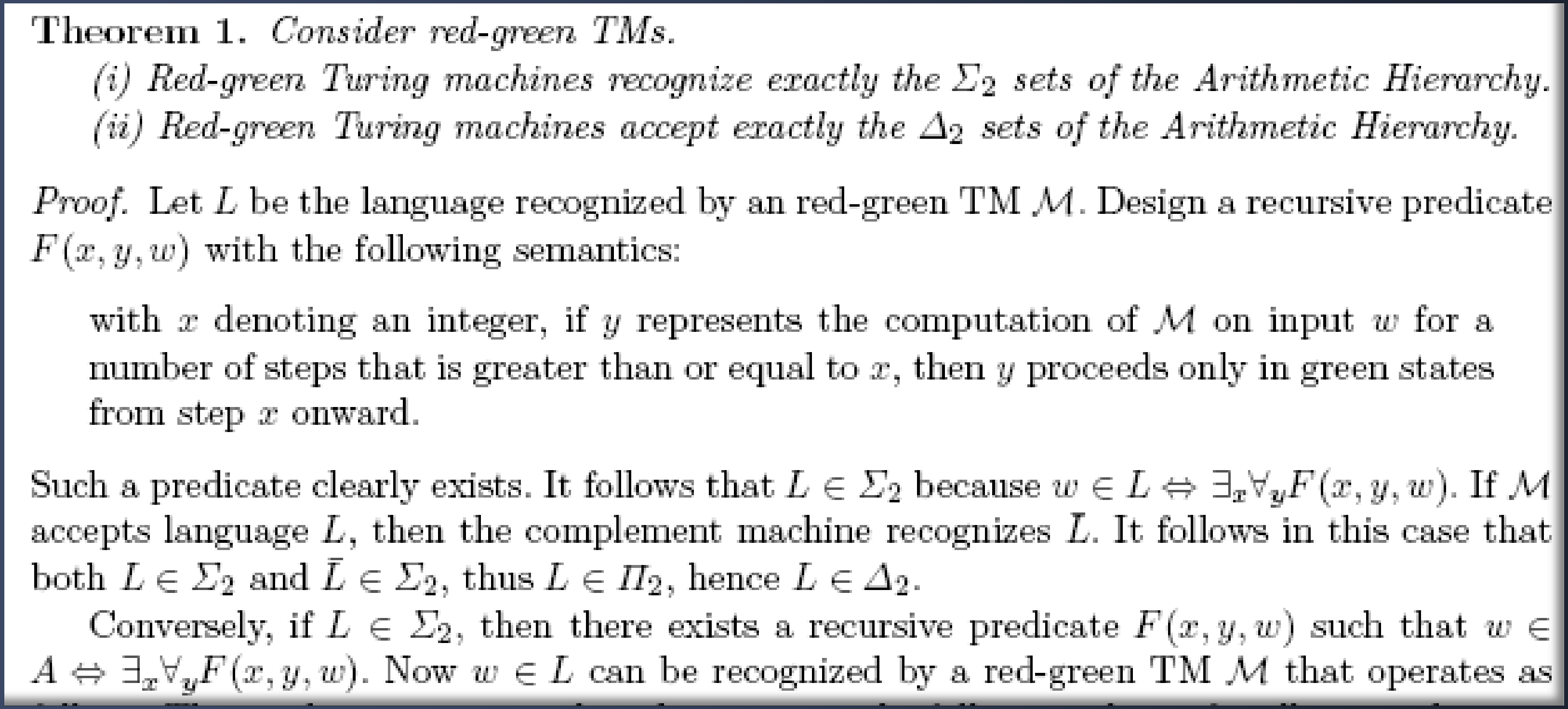
Red-green model:

- two kinds of states:
  - red (rejecting) states
  - green (accepting) states
- finite input
- the result only after the computation stabilizes in either red or green states

Computational power of red-green machines:



Lemma 4. *A set of strings L is recognized by a red-green TM within one mind change if and only if $L \in \Sigma_1$, i.e. if L is recursively enumerable.*



Theorem 1. *Consider red-green TMs.*

(i) Red-green Turing machines recognize exactly the Σ_2 sets of the Arithmetic Hierarchy.

(ii) Red-green Turing machines accept exactly the Δ_2 sets of the Arithmetic Hierarchy.

Proof. Let L be the language recognized by an red-green TM $\mathcal{M}$. Design a recursive predicate $F(x, y, w)$ with the following semantics:

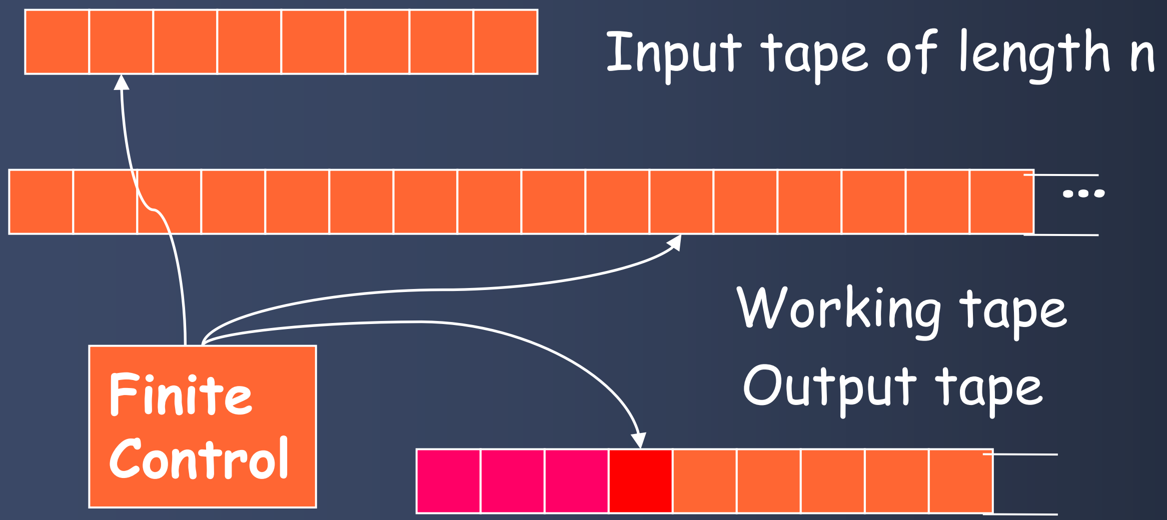
> with x denoting an integer, if y represents the computation of $\mathcal{M}$ on input w for a number of steps that is greater than or equal to x, then y proceeds only in green states from step x onward.

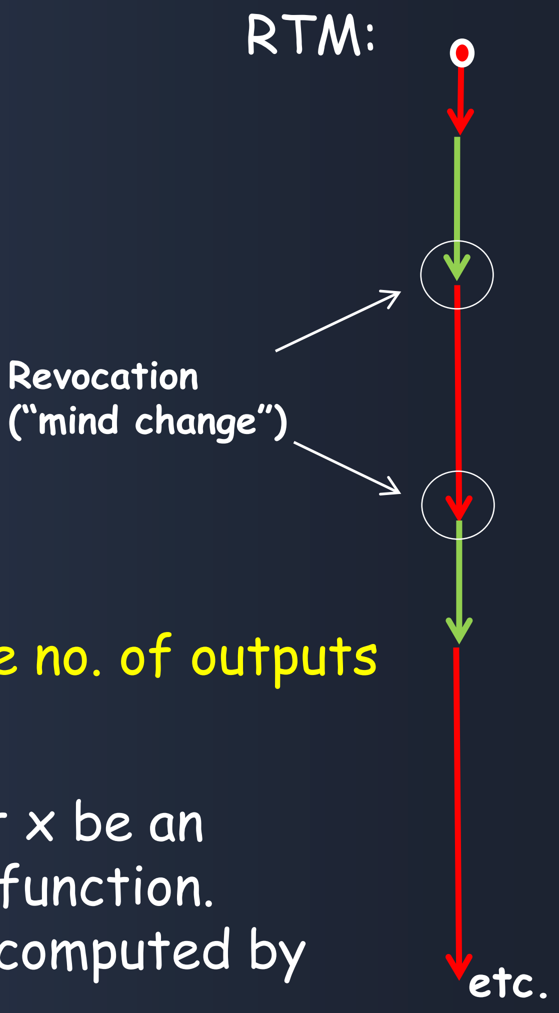
Such a predicate clearly exists. It follows that $L \in \Sigma_2$ because $w \in L \Leftrightarrow \exists_x \forall_y F(x, y, w)$. If $\mathcal{M}$ accepts language L, then the complement machine recognizes $\bar{L}$. It follows in this case that both $L \in \Sigma_2$ and $\bar{L} \in \Sigma_2$, thus $L \in \Pi_2$, hence $L \in \Delta_2$.

Conversely, if $L \in \Sigma_2$, then there exists a recursive predicate $F(x, y, w)$ such that $w \in A \Leftrightarrow \exists_x \forall_y F(x, y, w)$. Now $w \in L$ can be recognized by a red-green TM $\mathcal{M}$ that operates as

A special case of red-green Turing machines: (1936)

Turing's non-terminating circular a-machines

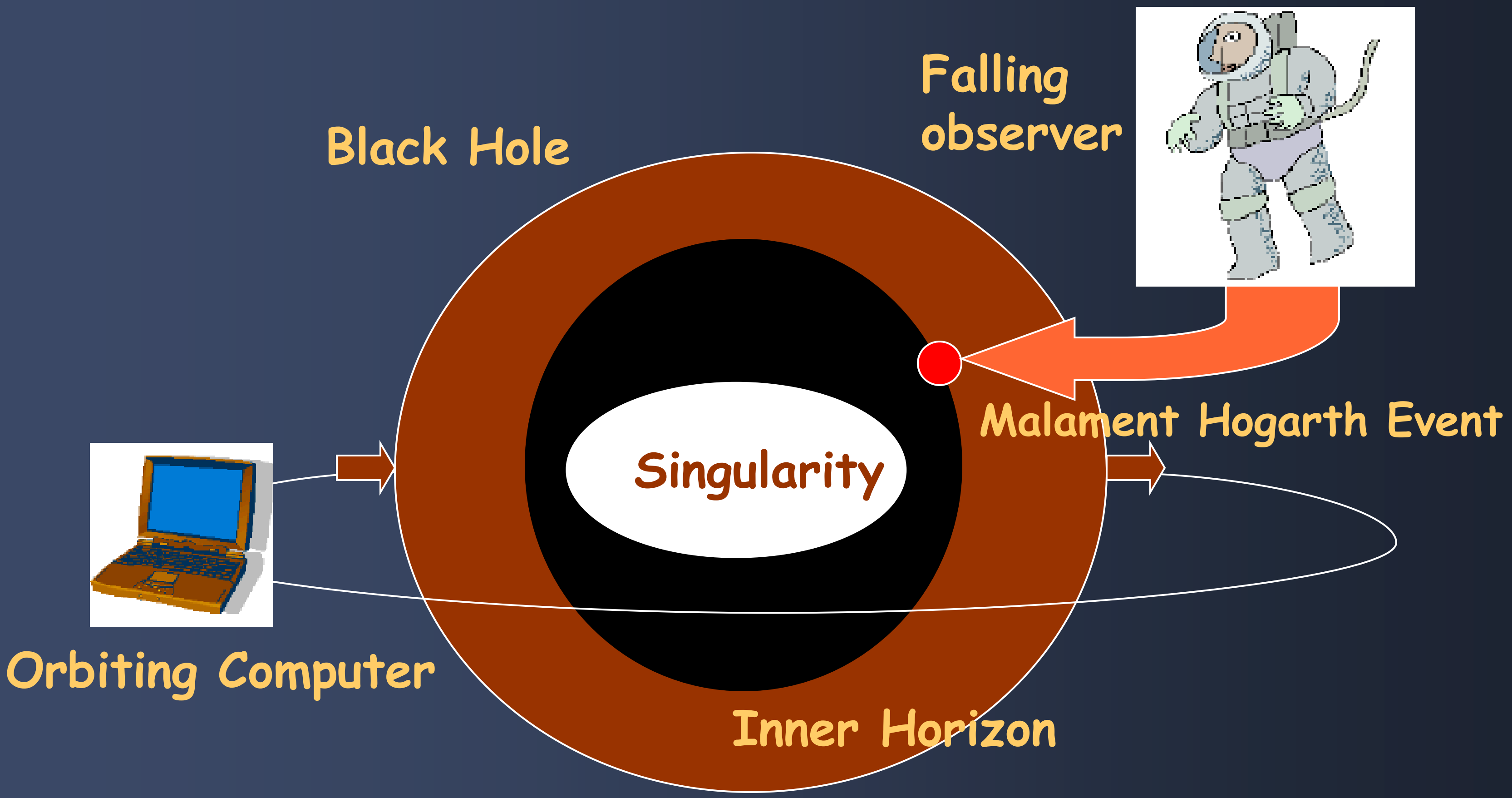




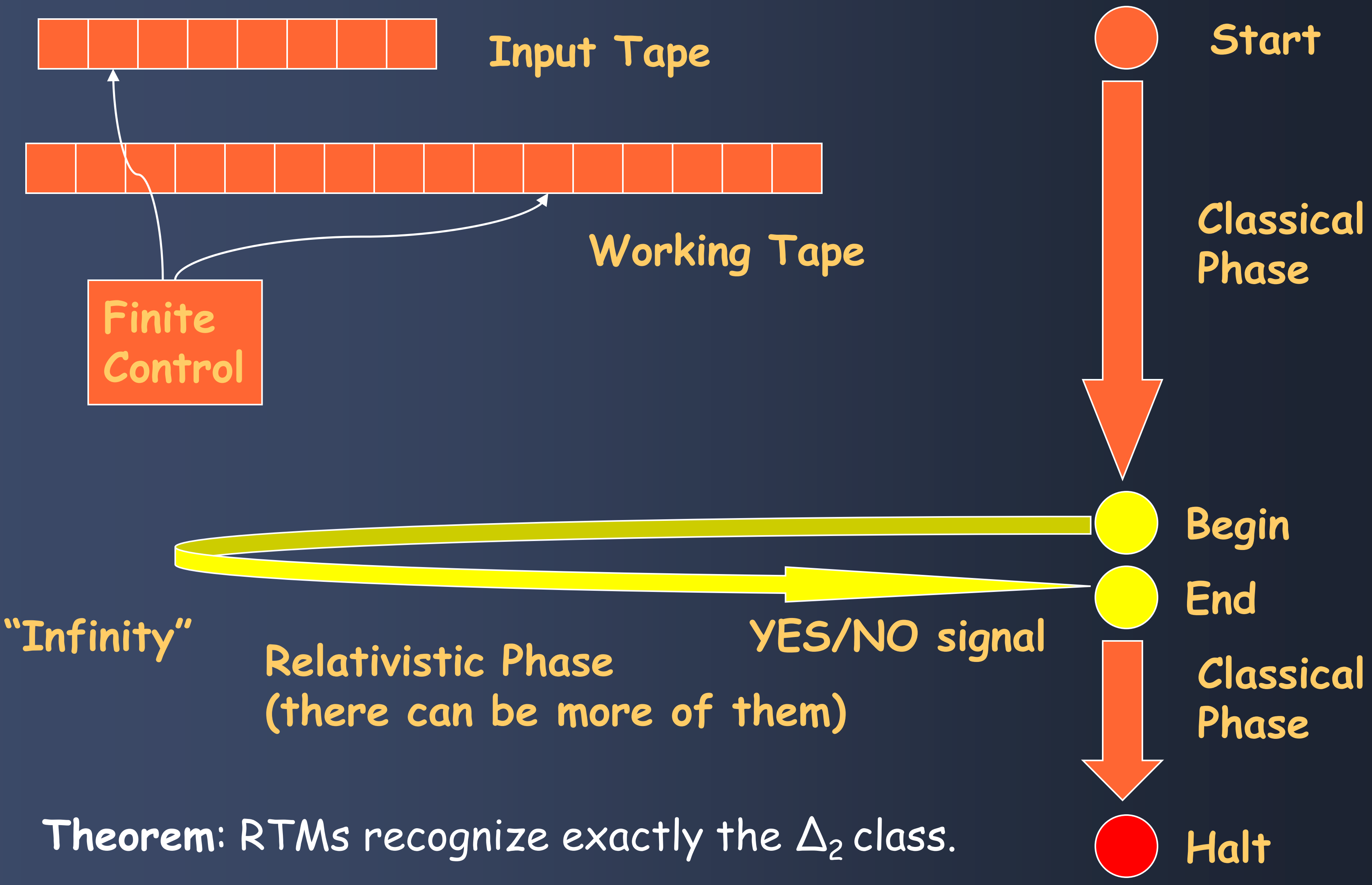
A ntcTM never halts while producing but a finite no. of outputs

Theorem: Let A be a ntcTM, let R be a rgTM, let x be an input string and let f: $\{0,1\} \to \{0,1\}^*$ be a partial function. Then f(x) is computed by A if and only if f(x) is computed by R with 2 f(x) mind changes.

Computing the Incomputable: Relativistic Computing

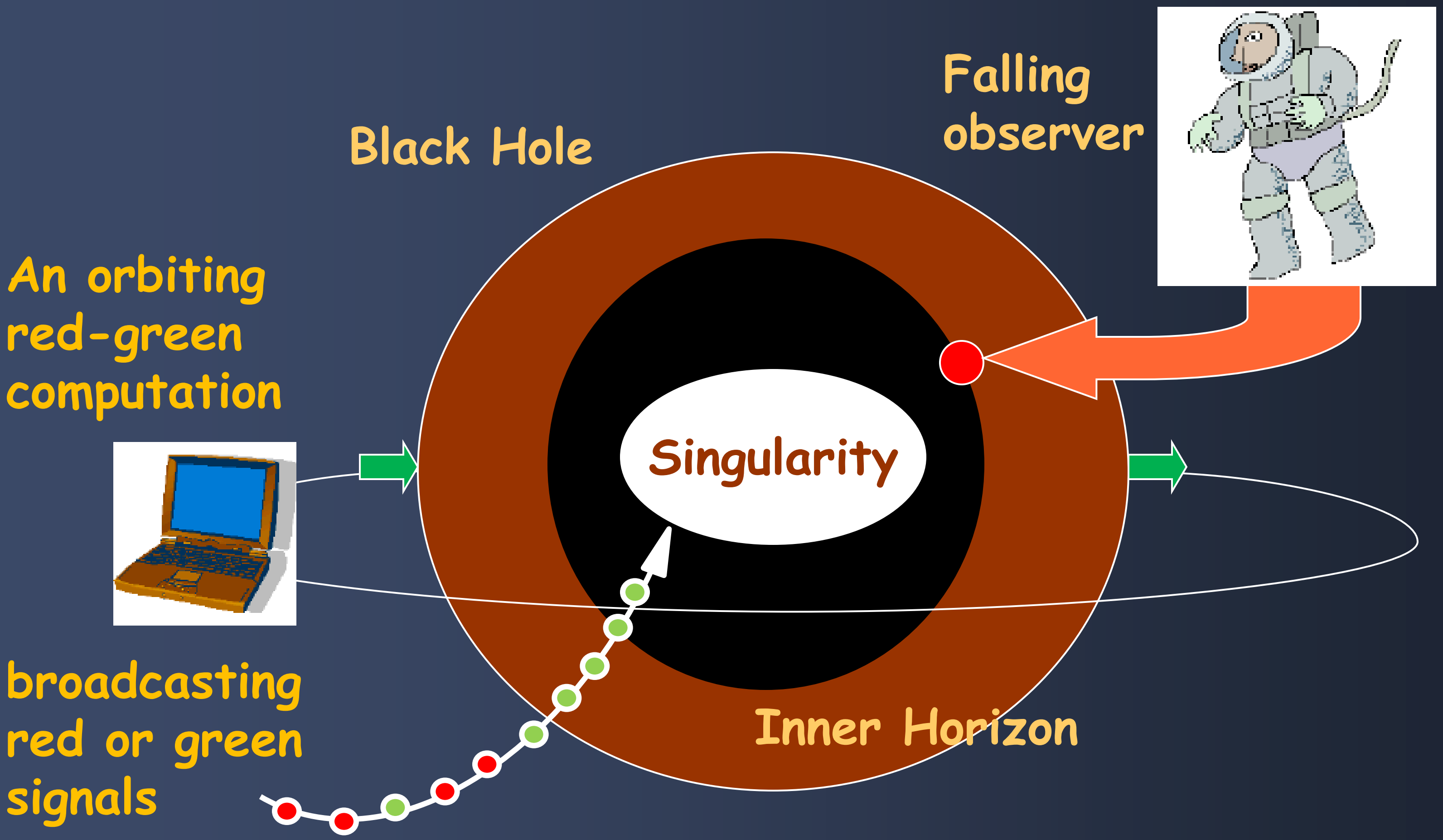


Relativistic Turing Machine



Theorem: RTMs recognize exactly the Δ_2 class.

A relativistic implementation of red-green computations



Black Hole

Falling observer

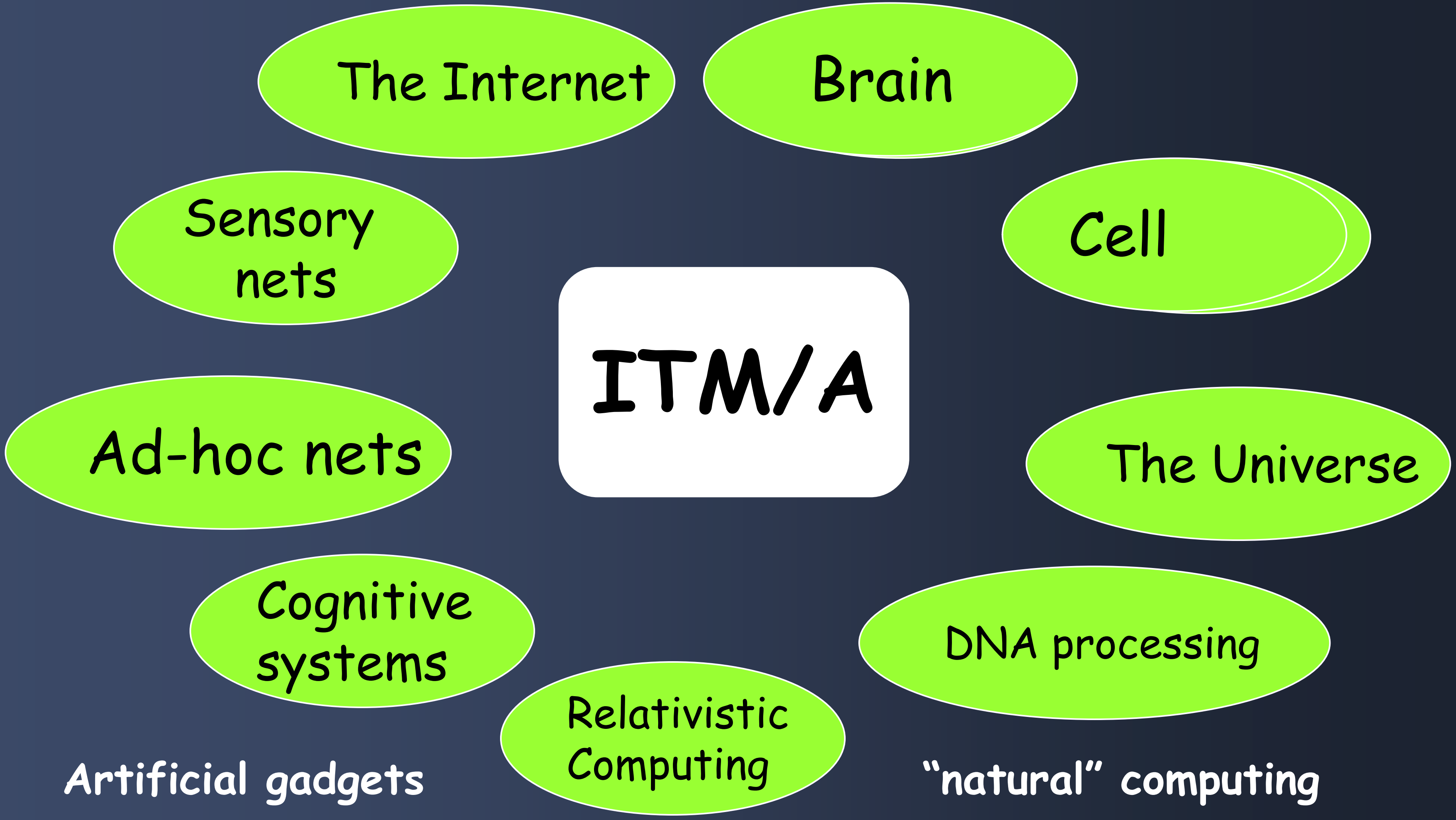
An orbiting red-green computation

Singularity

broadcasting red or green signals

Inner Horizon

Extended Turing machine paradigm: Any algorithmic process can be simulated by an ITM/A



ITM/A

The Internet

Brain

Sensory nets

Cell

Ad-hoc nets

The Universe

Cognitive systems

DNA processing

Relativistic Computing

Artificial gadgets

“natural” computing

Conclusions

Three important characteristics of contemporary computing:

- potentially infinite computations
- interaction
- non-uniform (unpredictable) hardware/software evolution

Extending the notion of computation:

Classical computation	Modern computation
A finite process whose parameters are fixed before the start of processing.	Potentially infinite evolutionary processes whose parameters can change during the processing in an unpredictable way in interaction with their environment

THANK YOU

COPYRIGHT DISCLAIMER

Texts, marks, logos, names, graphics, images, photographs, illustrations, artwork, copyrighted by their respective owners are used on these slides for non-commercial, educational and personal purposes only. Use of any copyrighted material is not authorized without the written consent of the copyright holder. Every effort has been made to respect the copyrights of other parties. If you believe that your copyright has been misused, please direct your correspondence to: jiri.wiedermann@cs.cas.cz stating your position and I shall endeavor to correct any misuse as early as possible.