# Algorithmic Arguments in Physics of Computation

Paul Vitányi*
CWI and University of Amsterdam

Ming Li**
University of Waterloo

**Abstract.** We show the usefulness of incompressibility arguments based on Kolmogorov complexity in physics of computation by several examples. These include analysis of energy parsimonious 'adiabatic' computation, and scalability of network architectures.

## 1 Introduction

In [Shannon, 1948] C. Shannon formulated information theory dealing with the average number of bits required to communicate a message produced by a random source from a sender to a receiver who both agree on the ensemble of possible messages. In this theory, if the universe of messages consists of a two elements, a sentence "let's go drink a beer" and Homer's Illiad, both elements equally likely, then the Illiad can be transmitted by a single bit. This illustrates that, as Shannon points out, this theory does not say anything about the information content of individual objects, but only says something about the required information exchange for communication.

In [Kolmogorov, 1965] A.N. Kolmogorov formulated a theory of information contents (Kolmogorov complexity) of individual finite objects. Since this theory deals with a stronger notion, namely information contents of *individual* objects instead of *average* information to communicate objects from *probabilistic ensembles*, it is not *a priori* obvious that properties of Shannon's notion would hold for Kolmogorov's new notion. Remarkably, it turns out that various properties, such as the 'symmetry of information' stating that the information in a random source $X$ about another random source $Y$ is precisely equal to the information in $Y$ about $X$, holds also approximately for Kolmogorov complexity of individual objects, that is, up to a logarithmic additive term.

* Partially supported by the European Union through NeuroCOLT ESPRIT Working Group Nr. 8556, and by NWO through NFI Project ALADDIN under Contract number NF 62-376 and NSERC under International Scientific Exchange Award ISE0125663. Address: CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. Email: paulv@cwi.nl

** Supported in part by NSERC operating grant OGP-046506, ITRC, and a CGAT grant. Address: Computer Science Department, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. Email: mli@math.uwaterloo.ca

Information theory *à la* Shannon has been shown applicable in a large range of areas ranging from combinatorics to communication and computation technologies. The special feature of Kolmogorov complexity is that it deals with individual objects. This allows topics of application were Shannon's theory is ostensibly powerless, albeit using slightly weaker laws. We and others have been able to find new simpler proofs for known results, like the Razborov – Fortnow – Laplante version of Hastad's Switching Lemma or Ian Munro's version of Russel Schaffer's exact average running time of Heapsort, to find new incompressibility arguments for combinatorial theory, a basis for inductive learning, or to resolve (formerly well known) old open problems in the theory of computation like Turing machine and PRAM time complexity. The basic theory and many of these applications are treated in our textbook [Li & Vitányi, 1993]. Recently, we have extensively used the symmetry of information law where application to individual objects seems absolutely crucial, [Jiang *et al.*].

Our purpose here is to point out that applicability of incompressibility arguments based on Kolmogorov complexity is not restricted to the platonic realm of mathematics and theory of computation, but can profitably be extended to the real world of physical phenomena.

### 1.1 Kolmogorov complexity

The Kolmogorov complexity, [Kolmogorov, 1965], of $x$ is the length of the *shortest* effective description of $x$. That is, the *Kolmogorov complexity* $C(x)$ of a finite string $x$ is simply the length of the shortest program, say in FORTRAN[3] encoded in binary, which prints $x$ without any input. A similar definition holds conditionally, in the sense that $C(x|y)$ is the length of the shortest binary program which computes $x$ given $y$ as input. It can be shown that the Kolmogorov complexity is absolute in the sense of being independent of the programming language, up to a fixed additional constant term which depends on the programming language but not on $x$. We now fix one canonical programming language once and for all as reference and thereby $C()$.

For the theory and applications, see [Li & Vitányi, 1993]. Let $x, y, z \in \mathcal{N}$, where $\mathcal{N}$ denotes the natural numbers and we identify $\mathcal{N}$ and $\{0,1\}^*$ according to the correspondence $(0, \epsilon), (1, 0), (2, 1), (3, 00), (4, 01), \ldots$. Hence, the length $|x|$ of $x$ is the number of bits in the binary string $x$. Let $T_1, T_2, \ldots$ be a standard enumeration of all Turing machines. Let $\langle \cdot, \cdot \rangle$ be a standard invertible effective bijection from $\mathcal{N} \times \mathcal{N}$ to $\mathcal{N}$. This can iterated to $\langle \langle \cdot, \cdot \rangle, \cdot \rangle$.

**Definition 1.** Let $U$ be an appropriate universal Turing machine such that $U(\langle \langle i, p \rangle, y \rangle) = T_i(\langle p, y \rangle)$ for all $i$ and $\langle p, y \rangle$. The *Kolmogorov complexity* of $x$ given $y$ (for free) is

$$C(x|y) = \min\{|p| : U(\langle p, y \rangle) = x, p \in \{0,1\}^*, i \in \mathcal{N}\}.$$

[3] Or in Turing machine codes.

## 2 Energy Parsimonious Computation

All computations can be performed logically reversibly, [Bennett, 1973], at the cost of eventually filling up the memory with unwanted garbage information. This means that reversible computers with bounded memories require in the long run irreversible bit operations, for example, to erase records irreversibly to create free memory space. The minimal possible number of irreversibly erased bits to do so is believed to determine the ultimate limit of heat dissipation of the computation by Landauer's principle, [Landauer, 1961, Bennett, 1973, Bennett, 1982, Proc. PhysComp, 1981, 1992, 1994]. In reference [Bennett *et al.,* 1993] we and others developed a mathematical theory for the unavoidable number of irreversible bit operations in an otherwise reversible computation. A precursor to this line of thought is [Zurek, 1989]. Here we present the operational proof in [Li & Vitányi, 1994] for the known exact expression of the number of irreversible bit operations in an otherwise reversible computation proved differently in [Bennett *et al.,* 1993].

Many currently proposed physical schemes implementing adiabatic computation reduce irreversibility by using longer switching times. This is done typically by switching over equal voltage gates after voltage has been equalized slowly. This type of switching does not dissipate energy, the only energy dissipation is incurred by pulling voltage up and down: the slower it goes the less energy is dissipated, [Proc. PhysComp, 1981, 1992, 1994]. If the computation goes infinitely slow, zero energy is dissipated. Clearly, this counteracts the purpose of low energy dissipation which is faster computation.

In [Li & Vitányi, 1994] it is demonstrated that even if adiabatic computation technology advances to switching with no time loss, a similar phenomenon arises when we try to approach the ultimate limits of minimal irreversibility of an otherwise reversible computation, and hence minimal energy dissipation. This time the effect is due to the logical method of reducing the number of irreversible bit erasures in the computation irrespective of individual switching times. By computing longer and longer (in the sense of using more computation steps), the amount of dissipated energy gets closer to ultimate limits. Moreover, one can trade-off time (number of steps) for energy: there is a new time-irreversibility (time-energy) trade-off hierarchy. The bounds we derive are also relevant for quantum computations which are reversible except for the irreversible observation steps, [Deutsch, 1985, Benioff, 1995].

### 2.1 Background

The ultimate limits of miniaturization of computing devices, and therefore the speed of computation, are governed by unavoidable heating up attending rising energy dissipation caused by increasing density of switching elements in the device. On a basically two dimensional device linear speed up by shortening interconnects is essentially attended by squaring the dissipated energy per area unit per second because we square the number of switching elements per area unit, [Mead & Conway, 1980].

Therefore, the question of how to reduce the energy dissipation of computation determines future advances in computing power. Around 1940 a computing device dissipated about $10^{-2}$ Joule per bit operation at room temperature. Since that time the dissipated energy per bit operation has roughly decreased by one order of magnitude (tenfold) every five years. Currently, a bit operation dissipates about $10^{-17}$ Joule.[4] Extrapolations of current trends show that the energy dissipation per binary logic operation needs to be reduced below $kT$ (thermal noise) within 20 years. Here $k$ is Boltzamnn's constant and $T$ the absolute temperature in $^o$Kelvin, so that $kT \approx 3 \times 10^{-21}$ Joule at room temperature. Even at $kT$ level, a future laptop containing $10^{18}$ gates in a cubic centimeter operating at a gigahertz dissipates 3 million watts/second. For thermodynamic reasons, cooling the operating temperature of such a computing device to almost absolute zero (to get $kT$ down) must dissipate at least as much energy in the cooling as it saves for the computing. It is unlikely that this challenge can be met by other means than the use of reversible logic.

J. von Neumann [Burks, 1966] reputedly thought that a computer operating at temperature $T$ must dissipate at least $kT \ln 2$ Joule per elementary bit operation. Around 1960, R. Landauer [Landauer, 1961] analyzed this question and concluded that it is only 'logically irreversible' operations that dissipate energy. An operation is *logically reversible* if its inputs can always be deduced from the outputs. Erasure of information in a way such that it cannot be retrieved is not reversible. Erasing each bit costs $kT \ln 2$ energy, when computer operates at temperature $T$.

One should sharply distinguish between the issue of logical reversibility and the issue of energy dissipation freeness. The fact that some computer operates in a logically reversible manner says nothing about whether it dissipates heat. The only thing it says is that the laws of physics do not preclude that one can invent a technology in which to implement a logically similar computer to operate physically in a dissipationless manner. Computers built from reversible circuits, or the reversible Turing machine, [Bennett, 1973, Bennett, 1982, Fredkin & Toffoli, 1982], implemented with current technology will presumably dissipate energy but may conceivably be implemented by future technology in an adiabatic fashion. For nonreversible computers adiabatic implementation is widely considered impossible.

Thought experiments can exhibit a computer that is both logically and physically perfectly reversible and hence perfectly dissipationless. An example is the billiard ball computer, [Fredkin & Toffoli, 1982], and similarly the possibility of a coherent quantum computer, [Feynman, 1985, Deutsch, 1985].

Methods to implement (almost) reversible dissipationless computation using conventional electronic technologies appear in [Proc. PhysComp, 1981, 1992, 1994], often designated by the catch phrase 'adiabatic switching'. Our purpose is to determine the theoretical ultimate limits to which the irreversible actions in an otherwise reversible computation can be reduced.

---

[4] After R.W. Keyes, IBM Research.

## 2.2 Model of Computation

Energy free 'copying' of records, and cancelling of one record with respect to an identical record provided it is known that they are identical, is physically realizable (or almost realizable). This is the case when a program sets $y := x$ and later (reversibly) erases $x := 0$. We shall call reversible erasure 'cancelling'. Irrespective of the original contents of variable $x$ we can always restore $x$ by $x := y$. However, if the program has no copy of $x$ which can be identified by examining the program without knowing the contents of the variables, then after (irreversibly) erasing $x := 0$ we cannot restore the original contents of $x$ even though some variable $z$ may have by chance the same contents. 'Copying' and 'cancelling' are logically reversible, and their energy dissipation free execution gives substance to the idea that logically reversible computations can be performed with zero energy dissipation.

We have seen that the number of irreversibly erased bits in an otherwise reversible computation which replaces input $x$ by output $y$, each unit counted as $kT \ln 2$, represents energy dissipation. Complementary to this idea, if such a computation uses initially irreversibly provided bits apart from input $x$, then they must be accounted at the same negated cost as that for irreversible erasure. Because of the reversibility of the computation, we can argue by symmetry. Namely, suppose we run a reversible computation starting when memory contains input $x$ and additional record $p$, and ending with memory containing output $y$ and additional garbage bits $q$. Then $p$ is irreversibly provided, and $q$ is irreversibly deleted. But if we run the computation backward, then the roles of $x, p$ and $y, q$ are simply interchanged.

We can view any computation as consisting of a sequence of reversible and irreversible operation executions. We want the irreversibility cost to reflect all nonreversible parts of the computation. The irreversibility cost of an otherwise reversible computation is set to the *maximum* of the number of irreversibly provided and the number of irreversibly erased bits.

We consider the following axioms as a formal basis on which to develop a theory of irreversibility of computation.

**Axiom 1** Reversible computations do not incur any cost.

**Axiom 2** Irreversibly provided and irreversibly deleted bits in a computation incur unit cost each.

**Axiom 3** In a reversible computation which replaces input $x$ by output $y$, the input $x$ is not irreversibly provided and the output $y$ is not irreversibly deleted.

**Axiom 4** All physical computations are effective.

Axiom 4 is simply an extended form of *Church's Thesis*: the notion of physical computation coincides with effective computation which coincides with the formal notion of Turing machines computation. Deutsch, [Deutsch, 1985], and others have argued the possibility that this is false. If that turns out to be the case then either our arguments are to be restricted to those physical processes for which Axiom 4 holds, or, perhaps, one can extend the notion of effective computations appropriately.

We will be talking about the ultimate limits of energy dissipation by computation. Since these limits will be expressed in the number of bits in the irreversibly erased records, we consider compactification of records. Rather as in analogy of garbage collection by a garbage truck: the cost is less if we compact the garbage before we throw it away.

The ultimate compactification which can be effectively exploited is expressed in terms of Kolmogorov complexity. This is a recursively invariant concept, and to express the ultimate limits *no other notion will do.* Consequently, this mundane matter of energy dissipation of physical computation is linked to, and expressed in, the pristine theoretical notion of Kolmogorov complexity.

### 2.3 Irreversibility Cost of Computation

Axioms 1—4 lead to the definition of the irreversibility cost of a computation as the number of bits we added plus the number of bits we erased in computing one string from another. Let $\mathbf{R} = R_1, R_2, \ldots$ be a standard enumeration of reversible Turing machines, [Bennett, 1973]. We define $E(\cdot,\cdot)$ as in [Bennett *et al.,* 1993] (where it is denoted as $E_3(\cdot,\cdot)$).

**Definition 2.** The *irreversibility cost* $E_R(x,y)$ of computing $y$ from $x$ by a reversible Turing machine $R$ is is

$$E_R(x,y) = \min\{|p| + |q| : R(\langle x,p\rangle) = \langle y,q\rangle\}.$$

We denote the class of all such cost functions by $\mathcal{E}$.

We call an element $E_Q$ of $\mathcal{E}$ a *universal irreversibility cost function*, if $Q \in \mathbf{R}$, and for all $R$ in $\mathbf{R}$

$$E_Q(x,y) \leq E_R(x,y) + c_R,$$

for all $x$ and $y$, where $c_R$ is a constant which depends on $R$ but not on $x$ or $y$. Standard arguments from the theory of Turing machines show the following.

**Lemma 3.** *There is a universal irreversibility cost function in $\mathcal{E}$. Denote it by $E_{UR}$.*

*Proof.* In [Bennett, 1973] a universal reversible Turing machine $UR$ is constructed which satisfies the optimality requirement.

Two such universal (or optimal) machines $UR$ and $UR'$ will assign the same irreversibility cost to a computation apart from an additive constant term $c$ which is *independent* of $x$ and $y$ (but does depend on $UR$ and $UR'$). We select a reference universal function $UR$ and define the *irreversibility cost* $E(x,y)$ of computing $y$ from $x$ as

$$E(x,y) \equiv E_{UR}(x,y).$$

Because of the expression for $E(x,y)$ in Theorem 4 below it is called the *sum cost* measure in [Bennett *et al.,* 1993].

In physical terms this cost is in units of $kT \ln 2$, where $k$ is Boltzmann's constant, $T$ is the absolute temperature in degrees Kelvin, and ln is the natural logarithm.

Because the computation is reversible, this definition is *symmetric*: we have $E(x,y) = E(y,x)$.

In our definitions we have pushed all bits to be irreversibly provided to the start of the computation and all bits to be erased to the end of the computation. It is easy to see that this is no restriction. If we have a computation where irreversible acts happen throughout the computation, then we can always mark the bits to be erased, waiting with actual erasure until the end of the computation. Similarly, the bits to be provided can be provided (marked) at the start of the computation while the actual reading of them (simultaneously unmarking them) takes place throughout the computation).

**Computing Between $x$ and $y$** Now let us consider a general computation which outputs string $y$ from input string $x$. We want to know the minimum irreversibility cost for such computation. This leads to the following theorem, first proven as below and also established in [Bennett *et al.,* 1993] by a different more indirect proof, which is the basis of our theory.

**Theorem 4 Fundamental theorem.** *Up to an additive logarithmic term*

$$E(x,y) = C(x|y) + C(y|x).$$

*Proof.* We prove first an upper bound and then a lower bound.

**Claim 1** $E(x,y) \leq C(y|x) + C(x|y) + 2[C(C(y|x)|y) + C(C(x|y)|x)]$.

*Proof.* We start out the computation with programs $p, q, r$. Program $p$ computes $y$ from $x$ and $|p| = C(y|x)$. Program $q$ computes the value $C(x|y)$ from $x$ and $|q| = C(C(x|y)|x)$. Program $r$ computes the value $C(y|x)$ from $y$ and $|r| = C(C(y|x)|y)$. To separate the different binary programs we have to encode delimiters. This takes an extra additional number of bits logarithmic in the two smallest length of elements $p, q, r$. This extra log term is absorbed in the additive log term in the statement of the theorem. The computation is as follows. Everything is executed reversibly apart from the final irreversible erasure.

1. Use $p$ to compute $y$ from $x$ producing garbage bits $g(x,y)$.
2. Copy $y$, and use one copy of $y$ and $g(x,y)$ to reverse the computation to $x$ and $p$. Now we have $p, q, r, x, y$.
3. Copy $x$, and use one copy of $x$ and $q$ to compute $C(x|y)$ plus garbage bits.
4. Use $x, y, C(x|y)$ to dovetail the running of all programs of length $C(x|y)$ to find $s$, a shortest program to compute $x$ from $y$. Doing this, we produce more garbage bits.

5. Copy $s$, and reverse the computations in Steps 4, 3, canceling the extra copies and all garbage bits. Now we have $p, q, r, s, x, y$.
6. Copy $y$, and use this copy to compute the value $C(y|x)$ from $r$ and $y$ producing garbage bits.
7. Use $x, y, C(y|x)$, to dovetail the running of all programs of length $C(y|x)$ to obtain a copy of $p$, the shortest program to compute $y$ from $x$, producing more garbage bits.
8. Delete a copy of $p$ and reverse the computation of Steps 7, 6 cancelling the superfluous copy of $y$ and all garbage bits. Now we are left with $x, y, r, s, q$.
9. Compute from $y$ and $s$ a copy of $x$ and cancel a copy of $x$. Reverse the computation. Now we have $y, r, s, q$.
10. Erase $s, r, q$ irreversibly.

We started out with additional shortest programs $p, q, r$ apart from $x$. We have irreversibly erased the shortest programs $s, q, r$, where $|s| = C(x|y)$, leaving only $y$. This proves the claim.

Note that all bits supplied in the beginning to the computation, apart from input $x$, as well as all bits irreversibly erased at the end of the computation, are *random* bits. This is because we supply and delete only shortest programs, and a shortest program $p$ satisfies $C(p) \geq |p|$, that is, it is maximally random.

**Claim 2** $E(x, y) \geq C(y|x) + C(x|y)$.

*Proof.* To compute $y$ from $x$ we must be given a program to do so to start out with. By definition the shortest such program has length $C(y|x)$.

Assume the computation from $x$ to $y$ produces $g(x, y)$ garbage bits. Since the computation is reversible we can compute $x$ from $y$ and $g(x, y)$. Consequently, $|g(x, y)| \geq C(x|y)$ by definition [Zurek, 1989]. To end the computation with $y$ alone we therefore must irreversibly erase $g(x, y)$ which is at least $C(x|y)$ bits.

Together Claims 1, 2 prove the theorem.

**Corollary 5.** *Erasing a record $x$ is actually a computation from $x$ to the empty string $\epsilon$. Therefore, up to a logarithmic additive term, the irreversible cost (also thermodynamic cost) of erasure is $E(x, \epsilon) = C(x)$.*

### 2.4 Trading Time for Energy

In order to erase a record $x$, Corollary 5 actually requires us to have, apart from $x$, a program $p$ of length $C(C(x)|x)$ for computing $C(x)$, given $x$. The precise bounds are $C(x) \leq E(x, \epsilon) \leq C(x) + 2C(C(x)|x)$. This optimum is not effective, it requires that $p$ be given in some way. But we can use the same method as in the proof of Theorem 4, by compressing $x$ using some time bound $t$.

First we need some definitions. Because now the time bounds are important we consider the universal Turing machine $U$ to be the machine with two work tapes which can simulate $t$ steps of a multitape Turing machine $T$ in $O(t \log t)$

steps, see for example [Li & Vitányi, 1993]. If some multitape Turing machine $T$ computes $x$ in time $t$ from a program $p$, then $U$ computes $x$ in time $O(t \log t)$ from $p$ plus a description of $T$.

**Definition 6.** Let $C^t(x|y)$ be the *minimal length* of binary program (not necessarily reversibly) for the two work tape universal Turing machine $U$ computing $x$ given $y$ (for free) *in time* $t$. Formally,

$$C^t(x|y) = \min_{p \in \mathcal{N}} \{|p| : U(\langle p, y \rangle) = x \text{ in } \leq t(|x|) \text{ steps}\}.$$

$C^t(x|y)$ is called the *t-time-limited conditional Kolmogorov complexity* of $x$ given $y$. The unconditional version is defined as $C^t(x) := C^t(x, \epsilon)$. A program $p$ such that $U(p) = x$ in $\leq t(|x|)$ steps and $|p| = C^t(x)$ is denoted as $x_t^*$.

Note that with $C_T^t(x|y)$ the conditional $t$-time-limited Kolmogorov complexity with respect to Turing machine $T$, for all $x, y$, $C^{t'}(x|y) \leq C_T^t(x|y) + c_T$, where $t' = O(t \log t)$ and $c_T$ is a constant depending on $T$ but not on $x$ and $y$.

This $C^t(\cdot)$ is the standard definition of time-limited Kolmogorov complexity, [Li & Vitányi, 1993]. However, in the remainder of the paper we always need to use reversible computations. Fortunately, in [Bennett, 1989] the following is shown.

**Lemma 7.** *For any $\epsilon > 0$, ordinary multitape Turing machines using $T$ time and $S$ space can be simulated by reversible ones using time $O(T)$ and space $O(ST^\epsilon)$.*

To do effective erasure of compacted information, we must at the start of the computation provide a time bound $t$. Typically, $t$ is a recursive function and the complexity of its description is small, say $O(1)$. However, in Theorem 8 we allow for very large running times in order to obtain smaller $C^t(\cdot)$ values. (In the theorem below $t$ need not necessarily be a recursive function $t(|x|)$, but can also be used nonuniformly. This leads to a stronger result.)

**Theorem 8 Irreversibility cost of effective erasure.** *If $t(|x|) \geq |x|$ is a time bound which is provided at the start of the computation, then erasing an $n$ bit record $x$ by an otherwise reversible computation can be done in time (number of steps) $O(2^{|x|} t(|x|))$ at irreversibility cost (hence also thermodynamic cost) $C^t(x) + 2C^t(t|x) + 4 \log C^t(t|x)$ bits. (Typically we consider $t$ as some standard explicit time bound and the last two terms adding up to $O(1)$.)*

*Proof.* Initially we have in memory input $x$ and a program $p$ of length $C^t(t, x)$ to compute reversibly $t$ from $x$. To separate binary $x$ and binary $p$ we need to encode a delimiter in at most $2 \log C^t(t|x)$ bits.

1. Use $x$ and $p$ to reversibly compute $t$. Copy $t$ and reverse the computation. Now we have $x$, $p$ and $t$.
2. Use $t$ to reversibly dovetail the running of all programs of length less than $x$ to find the shortest one halting in time $t$ with output $x$. This is $x_t^*$. The computation has produced garbage bits $g(x, x_t^*)$. Copy $x_t^*$, and reverse the computation to obtain $x$ erasing all garbage bits $g(x, x_t^*)$. Now we have $x, p, x_t^*, t$ in memory.

3. Reversibly compute $t$ from $x$ by $p$, cancel one copy of $t$, and reverse the computation. Now we have $x, p, x_t^*$ in memory.
4. Reversibly cancel $x$ using $x_t^*$ by the standard method, and then erase $x_t^*$ and $p$ irreversibly.

More practical compression methods are surveyed in [Storer, 1988].

By spending more time we can reduce the thermodynamic cost of erasure of $x_t^*$ to its absolute minimum. In the limit we spend the optimal value $C(x)$ by erasing $x^*$, since $\lim_{t \to \infty} x_t^* = x^*$. This suggests the existence of a trade-off hierarchy between time and energy. The longer one reversibly computes to perform final irreversible erasures, the less bits are erased and energy is dissipated. This intuitive assertion will be formally stated and rigourously proved below. We proceed through a sequence of related 'irreversibility' results.

**Definition 9.** Let $UR$ be the reversible version of the two worktape universal Turing machine, simulating the latter in linear time by Lemma 7. Let $E^t(x,y)$ be the *minimum irreversibility cost* of an otherwise reversible computation from $x$ to $y$ *in time* $t$. Formally,

$$E^t(x,y) = \min_{p,q \in \mathcal{N}} \{|p| + |q| : UR(\langle x,p \rangle) = \langle y,q \rangle \text{ in} \leq t(|x|) \text{ steps}\}.$$

Since $E(x,\epsilon)$ is about $C(x)$, one is erroneously led to believe that $E^t(x,\epsilon) = C^t(x)$ up to a log additive term. However, the time-bounds introduce many differences. To reversibly compute $x_t^*$ we may require (because of the halting problem) at least $O(2^{|x|}t(|x|))$ steps after having decoded $t$, as indeed is the case in the proof of Theorem 8. In contrast, $E^t(x,\epsilon)$ is about the number of bits erased in an otherwise reversible computation which uses at most $t$ steps. It is not difficult to show that for each $x$ and $t(|x|) \geq |x|$,

$$E^t(x,\epsilon) \geq C^t(x) \geq E^{t'}(x,\epsilon)/2, \tag{1}$$

with $t'(|x|) = O(t(|x|)$, [Li & Vitányi, 1994]. Moreover, Theorem 8 can be restated in terms of $E^t(\cdot)$ as

$$E^{t'}(x,\epsilon) \leq C^t(x) + 2C^t(t|x) + 4 \log C^t(t|x),$$

with $t'(|x|) = O(2^{|x|}t(|x|))$. Comparing this to the righthand inequality of Equation 1 we have improved the upper bound on erasure cost at the expense of increasing erasure time. However, these bounds only suggest but do not actually prove that we can exchange irreversibility for time. But the following result, shown by incompressibility arguments in [Li & Vitányi, 1994], definitely establishes the existence of a trade-off.

**Theorem 10 Irreversibility-time trade-off hierarchy.** *For every large enough $n$ there is a string $x$ of length $n$ and a sequence of $m = \frac{1}{2}\sqrt{n}$ time functions $t_1(n) < t_2(n) < \ldots < t_m(n)$, such that*

$$E^{t_1}(x,\epsilon) > E^{t_2}(x,\epsilon) > \ldots > E^{t_m}(x,\epsilon).$$

In the cost measures like $E^t(\cdot,\cdot)$ we have counted both the irreversibly provided and the irreversibly erased bits. But Landauer's principle only charges energy dissipation costs for irreversibly erased bits. It is conceivable that the above results would not hold if one considers as the cost of erasure of a record only the irreversibly erased bits. However, in [Li & Vitányi, 1994] it is shown that the above results hold under these considerations as well.

## 3 Scalability of Multiprocessor Architectures

In many areas of the theory of parallel computation we meet graph structured computational models which encourage the design of parallel algorithms where the cost of communication is largely ignored. Yet it is well known that the cost of computation - in both time and space - vanishes with respect to the cost of communication latency in parallel or distributed computing. It turns out that symmetric low diameter networks do not scale well; and random networks (and hence almost all networks) do not scale at all. This confirms that meshes are the way to go.

Models of parallel computation that allow processors to randomly access a large shared memory, such as PRAMs, or rapidly access a member of a large number of other processors, will necessarily have large latency. If we use $2^n$ processing elements of, say, unit size each, then the tightest they can be packed is in a 3-dimensional sphere of volume $2^n$. Assuming that the units have no "funny" shapes, e.g., are spherical themselves, no unit in the enveloping sphere can be closer to all other units than a distance of radius $R$,

$$R = \left( \frac{3 \cdot 2^n}{4\pi} \right)^{1/3} \tag{2}$$

Because of the bounded speed of light, it is impossible to transport signals over $2^{\alpha n}$ $(\alpha > 0)$ distance in polynomial $p(n)$ time. In fact, the assumption of the bounded speed of light says that the lower time bound on *any* computation using $2^n$ processing elements is $\Omega(2^{n/3})$ outright. Or, for the case of computations on networks which use $n^\alpha$ processors, $\alpha > 0$, the lower bound on the computation time is $\Omega(n^{\alpha/3})$.

In previous theoretical analysis, often a wire did not take room, did not dissipate heat, and did not cost anything - at least, not enough to worry about. This was realistic when the number of wires was low, somewhere in the hundreds. Current designs use many millions of wires (on chip), or possibly billions of wires (on wafers). In a computation of parallel nature, most of the time seems to be spent on communication - transporting signals over wires. The present analysis allows us to see that any reasonable model for multicomputer computation must charge for communication. The communication cost will impact on both physical time and physical space costs.

### 3.1 Regular Low Diameter Networks

At present, many popular multicomputer architectures are based on highly symmetric communication networks with small diameter. Like all networks with small diameter, such networks necessarily contain *some* long interconnects (embedded edges). We have shown in [Vitányi, 1986, Vitányi, 1988] that the desirable fast permutation properties of symmetric networks don't come free, since they require that the average of *all* interconnects is long. (Note that 'embedded edge,' 'wire,' and 'interconnect' are used synonymously.) To preclude objections that the results hold only asymptotically (and therefore can be safely ignored for practical numbers of processors), or that processors are huge and wires thin (idem), we calculated without hidden constants and assume that wires have length but no volume and can pass through everything. It is consistent with the results that wires have *zero* volume, and that *infinitely* many wires pass through a unit area. The lower bound obtained hlolds for the *average* edge length for *any* graph, in terms of certain symmetries and diameter. The lower bound deteriorates when the graph is irregular. For each regular graph topology we have examined, the resulting lower bound turned out to be sharp. It turns out that for symmetric networks like binary $d$-cube, cube-connected cycles, star graphs, complete graphs, the average edge length is as bad as can be. An extension of the argument shows the same for related networks like the Bruijn networks, shuffle-exchange graphs, and so on, [Koppelman, 1995].

### 3.2 Irregular Networks

Since low-diameter symmetric network topologies lead to high average interconnect length, it is natural to ask what happens with irregular topologies. In fact, it is sometimes proposed that since symmetric networks of low diameter lead to high interconnect length, one should use random networks where the presense or absence of a connection is determined by a coin flip. We report on some work in [Vitányi, 1994] that such networks will also have impossibly high average interconnect length.

Concretely, the problem is posed as follows. Let $G = (V, E)$ be a finite undirected graph, without loops or multiple edges, *embedded* in 3-dimensional Euclidean space. Let each embedded node have unit *volume*. For convenience of the argument, each node is embedded as a sphere, and is *represented* by the single point in the center. The *distance* between a pair of nodes is the Euclidean distance between the points representing them. The *length* of the embedding of an edge between two nodes is the distance between the nodes. How large does the *average* edge length need to be?

One way to express irregularity or *randomness* of an individual network topology is by a modern notion of randomness like Kolmogorov complexity. A simple counting argument shows that for each $y$ in the condition and each length $n$ there exists at least one $x$ of length $n$ which is *incompressible* in the sense of $C(x|y) \geq n$, 50% of all $x$'s of length $n$ is incompressible but for 1 bit ($C(x|y) \geq n-1$), 75%th of all $x$'s is incompressible but for 2 bits ($C(x|y) \geq n-2$)

and in general a fraction of $1-2^{-c}$ of all strings cannot be compressed by more than $c$ bits, [Li & Vitányi, 1993].

Each graph $G=(V,E)$ on $n$ nodes $V=\{0,\ldots,n-1\}$ can be coded (up to isomorphism) by a binary string of length $n(n-1)/2$. We enumerate the $n(n-1)/2$ possible edges in a graph on $n$ nodes in standard order and set the $i$th bit in the string to 1 if the edge is present and to 0 otherwise. Conversely, each binary string of length $n(n-1)/2$ encodes a graph on $n$ nodes. Hence we can identify each such graph with its corresponding binary string.

We shall call a graph $G$ on $n$ nodes *random* if it satisfies

$$C(G|n) \geq n(n-1)/2 - cn, \tag{3}$$

where $c$ is an appropriate constant ($c=1/16$ suffices for our purpose). Elementary counting shows that *a fraction* of at least

$$1-1/2^{cn}$$

of all graphs on $n$ nodes has that high complexity.

**Lemma 11.** *The degree $d$ of each node of a random graph satisfies $|d-(n-1)/2| < n/4$.*

*Proof.* Assume that the deviation of the degree $d$ of a node $v$ in $G$ from $(n-1)/2$ is at least $k$. From the lower bound on $C(G|n)$ corresponding to the assumption that $G$ is random, we can estimate an upper bound on $k$, as follows.

Describe $G$ given $n$ as follows. We can indicate which edges are incident on node $v$ by giving the index of the connection pattern in the ensemble of

$$m = \sum_{|d-(n-1)/2|\geq k} \binom{n}{d} \leq 2^n e^{-k^2/(n-1)} \tag{4}$$

possibilities. The last inequality follows from a general estimate of the tail probability of the binomial distribution, with $s_n$ the number of successful outcomes in $n$ experiments with probability of success $0<p<1$ and $q=1-p$. Namely, Chernoff's bounds, [Li & Vitányi, 1993], pp. 127-130, give

$$\Pr(|s_n - np| \geq k) \leq 2e^{-k^2/4npq}. \tag{5}$$

To describe $G$ it then suffices to modify the old code of $G$ by prefixing it with

- the identity of the node concerned in $\lceil \log n \rceil$ bits,
- the value of $d$ in $\lceil \log n \rceil$ bits, possibly adding nonsignificant 0's to pad up to this amount,
- the index of the interconnection pattern in $\log m + 2\log\log m$ bits in self-delimiting form (this form requirement allows the concatenated binary subdescriptions to be parsed and unpacked into the individual items: it encodes a separation delimiter, at the cost of adding the second term, [Li & Vitányi, 1993]),

followed by the old code for $G$ with the bits in the code denoting the presence or absence of the possible edges which are incident on the node $v$ deleted.

Clearly, given $n$ we can reconstruct the graph $G$ from the new description. The total description we have achieved is an effective program of

$$\log m + 2\log\log m + O(\log n) + n(n-1)/2 - (n-1)$$

bits. This must be at least the length of the shortest effective binary program, which is $C(G|n)$ satisfying Equation 3. Therefore,

$$\log m + 2\log\log m \geq n - 1 - O(\log n) - cn.$$

Since we have estimated in Equation 4 that

$$\log m \leq n - (k^2/(n-1))\log e,$$

it follows that, with $c = 1/16$,

$$k < n/4.$$

The lemma shows that each node is connected by an edge with about 25% of all nodes in $G$. Hence $G$ contains a subgraph on about 25 % of its nodes of diameter 1. This is all we need. For completeness, we derive the following lemma, using an idea due to Harry Buhrman.

**Lemma 12.** *Random graphs have diameter 2.*

*Proof.* The only graphs with diameter 1 are the complete graphs which can be described in $O(1)$ bits, given $n$, and hence are not random. It remains to consider $G$ is a random graph with diameter greater than 2. Let $i, j$ be a pair of nodes with distance greater than 2. Then we can describe $G$ by modifying the old code for $G$ by prefixing it with

- The identities of $i < j$ in $\lceil \log n \rceil$ bits,

followed by the old code of $G$ with all bits representing an edge $(j, k)$ between $j$ and each $k$ with $(i, k)$ an edge of $G$ deleted. We know that all the bits representing such edges must be 0 since the existence of any such edge shows that $(i, k), (k, j)$ is a path of length 2 between $i$ and $j$ contradicting the assumption that $i$ and $j$ have distance $> 2$. Since we know the identities of $i$ and $j$, and the nodes adjacent to $j$, we can reconstruct $G$ from this discussion and the new description, given $n$. Since by Lemma 11 the degree of $i$ is at least $n/4$, the new description of $G$, given $n$, has at most

$$n(n-1)/2 + 2\lceil \log n \rceil - n/4 + O(1),$$

which contradicts Equation 3 from some $n$ onwards.

**Theorem 13.** *A fraction of at least $1 - 1/2^{cn}$ ($c = 1/16$) of all graphs on $n$ nodes (the incompressible, random, graphs) have average interconnect length of $\Omega(n^{1/3})$ in each 3-dimensional Euclidean space embedding (or $\Omega(n^{1/2})$ in each 2-dimensional Euclidean space embedding).*

*Proof.* By lemma 11 we know that in a random graph $G$ each node $x$ is at distance 1 of $(n-1)/2 \pm n/4$ other nodes $y$, and 7/8th of these nodes $y$ (in 3 dimensions) is at distance $\Omega(n^{1/3})$ of $x$ by Equation 2. The argument for 2 dimensions is analogous.

By Lemma 11 we know that a random graph $G$ on $n$ nodes has $\Omega(n^2)$ edges since each node has about $n/2$ incident edges. Therefore, we have the following.

**Corollary 14.** *A fraction of at least* $1 - 1/2^{cn}$ $(c = 1/16)$ *of all graphs on* $n$ *nodes (the incompressible, random, graphs) have total interconnect length of* $\Omega(n^{7/3})$ *in each 3-dimensional Euclidean space embedding (or* $\Omega(n^{5/2})$ *in each 2-dimensional Euclidean space embedding).*

Since both the very regular symmetric low diameter graphs and the random graphs have high average interconnect length which sharply rises with $n$, the only graphs which will scale feasibly up are symmetric fairly high diameter topologies like the mesh—which therefore will most likely be the interconnection pattern of the future massive multiprocessor systems.

### 3.3 Interpretation of the Results

An effect that becomes increasingly important at the present time is that most space in the device executing the computation is taken up by the wires. Let's make the very conservative estimates that the unit length of a wire has a volume which is a constant fraction of that of a component it connects.

**Regular Networks.** We have shown in [Vitányi, 1988] that in 3-dimensional layouts for binary $d$-cubes, the volume of the $n = 2^d$ components (nodes) performing the actual computation operations is an asymptotic fastly vanishing fraction of the volume of the wires needed for communication:

$$\frac{\text{volume computing components}}{\text{volume communication wires}} = o(n^{-1/3})$$

If we charge a constant fraction of the unit volume for a unit wire length, and add the volume of the wires to the volume of the nodes, then the volume necessary to embed the binary $d$-cube is $\Omega(n^{4/3})$. However, this lower bound ignores the fact that the added volume of the wires pushes the nodes further apart, thus necessitating longer wires again. How far does this go? A rigorous analysis is complicated, and not important here. The following intuitive argument indicates what we can expect well enough. Denote the volume taken by the nodes as $V_n$, and the volume taken by the wires as $V_w$. The total volume taken by the embedding of the cube is $V_t = V_n + V_w$. The total wire length required to lay out a binary $d$-cube as a function of the volume taken by the embedding is, substituting radius $R$ obtained from $V_t = 4\pi R^3/3$ in the formula for the total wire length obtained in [Vitányi, 1988],

$$L(V_t) \geq \frac{7n}{32} \left( \frac{3V_t}{4\pi} \right)^{1/3}$$

Since $\lim_{n \to \infty} V_n/V_w \to 0$, assuming unit wire length of unit volume, we set the total interconnect length $L(V_t)$ at $L(V_t) \approx V_t$. This results in a better estimate of $\Omega(n^{3/2})$ for the volume needed to embed the binary $d$-cube. When we want to investigate an upper bound to embed the binary $d$-cube under the current assumption, we have a problem with the unbounded degree of unit volume nodes. There is no room for the wires to come together at a node. For comparison, therefore, consider the fixed degree version of the binary $d$-cube, the Cube Connected Cycles (CCC) topology (see [Vitányi, 1988]), with $n = d2^d$ trivalent nodes and $3n/2$ edges. The same argument yields $\Omega(n^{3/2} \log^{-3/2} n)$ for the volume required to embed CCC with unit volume per unit length wire. It is known, that every small degree $n$-vertex graph, e.g., CCC, can be laid out in a 3-dimensional grid with volume $O(n^{3/2})$ using a unit volume per unit wire length assumption, [Mead & Conway, 1980, Ullman, 1984]. This neatly matches the lower bound.

Because of current limitations to layered VLSI technology, previous investigations have focussed on embeddings of graphs in 2-space (with unit length wires of unit volume). We observe that the above analysis for 2 dimensions leads to $\Omega(n^2)$ and $\Omega(n^2 \log^{-2} n)$ volumes for the binary $d$-cube and the cube-connected cycles, respectively. These lower bounds have been obtained before using *bisection width* arguments, and are known to be optimal, [Ullman, 1984]. In [Mead & Conway, 1980] it is shown that we cannot always assume that a unit length of wire has $O(1)$ volume. (For instance, if we want to drive the signals to very high speed on chip.)

**Irregular Networks.** Just like for the complete graph, the situation for the random graph which we consider here, is far worse. For a random graph we have, under the assumption that the wires have unit volume per unit length, that the total wire length in 3 dimensional embeddings is $\Omega(n^{7/3})$ by Theorem 13, and that

$$\frac{\text{volume communication wires}}{\text{volume computing components}} = \Omega(n^{4/3})$$

The proof of Theorem 13 actually shows that the total interconnect length of an embedded random graph is $L(V_t) = \Omega(n^2 V_t^{1/3})$, where the radius of an as tight as possibly packed 3-dimensional sphere of the total volume $V_t$ of nodes and wires together is $\Omega(V_t^{1/3})$. Considering that the larger volume will cause the average interconnect length to increase, as above for the binary $d$-cube, setting the total interconnect length $L(V_t) \approx V_t$ since the volume of the computing nodes add a negligible term, we find for a random graph that on $n$ nodes that the total volume satisfies

$$V_t = \Omega(n^3).$$

Here we have not yet taken into account that longer wires need larger drivers and have a larger diameter, that the larger volume will again cause the average interconnect length to increase, and so on, which explosion may make embedding altogether impossible with finite length interconnects as exhibited in related contexts in [Vitányi, 1985].

The arguments we have developed are purely geometrical, apply to any graph and any technology, and give optimal lower bounds in all cases we have examined. Our observations are mathematical consequences from the noncontroversial assumptions on 3 dimensional space and the Laws of Physics.

## 4 Algorithmic Entropy, Chaos, Biology

**Algorithmic Entropy.** In [Gács, 1994, Li & Vitányi, 1993] an application of Kolmogorov complexity in statistical thermodynamics due to Péter Gács is reported. One can explain the classical theory of thermodynamics by statistical and information-theoretic analysis of an underlying deterministic model. It turns out that a complexity analysis using the powerful methods as explained in [Li & Vitányi, 1993] gives a basis of an algorithmic theory for entropy. Some applications include a proof of an 'entropy nondecrease over time' property, and 'entropy stability' property, 'entropy increase' for certain systems, and an analysis of Maxwell's demon.

**Chaos and Predictability.** Given sufficient information about a physical system, like the positions, masses and velocities of all particles, and a sufficiently powerful computer with enough memory and computation time, it should be possible in principle to compute all of the past and all of the future of the system. This view, eloquently propagated by P.S. Laplace, can be espoused both in classical mechanics and quantum mechanics. In classical mechanics one would talk about a single 'history', while in quantum mechanics one would talk about probability distributions over an ensemble of 'possible histories.'

Nonetheless, in practice it is impossible to obtain all parameters precisely. The finitary nature of measurement and computation requires truncation of real valued parameters; there are measuring errors; and according to basic quantum mechanics it is impossible to measure certain types of parameters simultaneously and precisely. Altogether, it is fundamental that there are minute uncertainties in our knowledge of any physical system at any time.

This effect can been combined with the consistent tradition that small causes can have large effects exemplified by the metaphor "a butterfly moving its wing in tropical Africa can eventually cause a cyclone in the Caribbean." Minute perturbations in initial conditions can cause, mediated by strictly computable functions, arbitrary large deviations in outcome. In the mathematics of nonlinear deterministic systems this phenomenon has been described by the catch term 'chaos'.

The unpredictability of this phenomenon is sometimes explained through Kolmogorov complexity. Assuming that the initial state is randomly drawn from $[0, 1)$ according to the uniform measure, [Ford, 1983] and other papers, use complexity arguments to show that the *doubling map*'s observable orbit cannot be predicted better than a coin toss. Namely, with $\lambda$-probability 1 the drawn initial state will be a Martin-Löf random infinite sequence. Such sequences by definition cannot be effectively predicted [Li & Vitányi, 1993] better than a random coin toss. But in this case we do not need to go to such trouble. The observed orbit essentially consists of the consecutive bits of the initial state. Selecting that initial state randomly from the uniform measure is isomorphic to flipping a fair coin to generate it. There emerges the question of a genuinely significant application of Kolmogorov complexity to unpredictability of chaotic trajectories.

**Compression by Ants.** In everyday life, we continuously compress information which is presented to us by the environment. Perhaps animals do this as well, as the following experiment reported by Zh.I. Reznikova and B.Ya. Ryabko [*Prob. Inform. Transm.*, 22:3(1986), 245-249, also reported in [Li & Vitányi, 1993]] suggests. It is claimed there that the transmission of information by ants using tactile code is a well-established fact. This led the researchers to probe both the information transmission rate and message compressing capabilities of ants. The experimental results suggest that, apparently, it takes a longer time for the scout ants to communicate 'random' sequences to the forager ants than to communicate 'regular' sequences.

## References


[Benioff, 1995] P. Benioff, Review of quantum computation, Argonne National Laboratories, manuscript 1995.

[Bennett, 1973] C.H. Bennett. Logical reversibility of computation. *IBM J. Res. Develop.*, 17:525–532, 1973.

[Bennett, 1982] C.H. Bennett. The thermodynamics of computation—a review. *Int. J. Theoret. Phys.*, 21(1982), 905-940.

[Bennett, 1989] C.H. Bennett. Time-space trade-offs for reversible computation. *SIAM J. Comput.*, 18(1989), 766-776.

[Bennett *et al.*, 1993] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi and W.H Zurek, Thermodynamics of computation and information distance *Proc. 25th ACM Symp. Theory of Computation.* ACM Press, 1993, 21-30.

[Deutsch, 1985] D. Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer. *Proc. Royal Society London.* Vol. A400(1985), 97-117.

[Feynman, 1985] R. Feynman. Quantum mechanical computers. *Foundations of Physics*, 16(1986), 507-531. (Originally published in *Optics News*, February 1985.)

[Ford, 1983] J. Ford, 'How random is a random coin toss?', *Physics Today*, Series A, 1983.

[Gács, 1974] P. Gács. On the symmetry of algorithmic information. *Soviet Math. Dokl.*, 15:1477–1480, 1974. Correction, Ibid., 15:1480, 1974.

[Gács, 1994] P. Gács, The Boltzmann entropy and randomness tests. In: Proc. 2nd IEEE Workshop on Physics and Computation (PhysComp'94), 1994, 209-216.

[Fredkin & Toffoli, 1982] E. Fredkin and T. Toffoli. Conservative logic. *Int. J. Theoret. Phys.*, 21(1982),219-253.

[Jiang *et al.*] T. Jiang, J. Seiferas and P.M.B. Vitányi, Two heads are better than two tapes, In: *Proc. 26th ACM Symp. Theory of Comput.*, 1994, 668-675.

[Kolmogorov, 1965] A.N. Kolmogorov, Three approaches to the definition of the concept 'quantity of information', *Problems in Information Transmission*, **1:1**(1965), 1-7.

[Koppelman, 1995] D.M. Koppelman, A lower bound on the average physical length of edges in the physical realization of graphs, Manuscript Dept ECE, Lousiana State Univ. Baton Rouge, 1995.

[Landauer, 1961] R. Landauer. Irreversibility and heat generation in the computing process. *IBM J. Res. Develop.*, 5:183–191, 1961.

[Li & Vitányi, 1993] M. Li and P.M.B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications.* Springer-Verlag, New York, 1993.

[Li & Vitányi, 1994] M. Li and P.M.B. Vitányi. *Irreversibility and Adiabatic Computation: Trading time for energy*, submitted.

[Mead & Conway, 1980] C. Mead and L. Conway. *Introduction to VLSI Systems.* Addison-Wesley, 1980.

[Proc. PhysComp, 1981, 1992, 1994] Proc. 1981 Physics and Computation Workshop. *Int. J. Theoret. Phys.*, 21(1982). *Proc. IEEE 1992 Physics and Computation Workshop.* IEEE Computer Society Press, 1992. *Proc. IEEE 1994 Physics and Computation Workshop.* IEEE Computer Society Press, 1994.

[Shannon, 1948] C.E. Shannon, A mathematical theory of communication, *Bell System Tech. J.*, **27**(1948), 379-423, 623-656.

[Storer, 1988] J. Storer. *Data Compression: Method and Theory.* Computer Science Press, 1988.

[Ullman, 1984] J. Ullman, *Computational Aspects of VLSI*, Computer Science Press, Rockville, MD, 1984.

[Vitányi, 1985] Area penalty for sublinear signal propagation delay on chip, *Proceedings 26th Annual IEEE Symposium on Foundations of Computer Science*, 1985, 197-207.

[Vitányi, 1986] P.M.B. Vitányi, Non-sequential computation and Laws of Nature, In: VLSI Algorithms and Architectures (Proceedings Aegean Workshop on Computing, 2nd International Workshop on Parallel Processing and VLSI), *Lecture Notes In Computer Science 227,* Springer Verlag, 1986, 108-120.

[Vitányi, 1988] P.M.B. Vitányi, Locality, communication and interconnect length in multicomputers, *SIAM J. Computing,* 17 (1988), 659-672.

[Vitányi, 1994] P.M.B. Vitányi, Multiprocessor architectures and physical law. In: Proc. 2nd IEEE Workshop on Physics and Computation (PhysComp'94), 1994, 24-29.

[Burks, 1966] J. von Neumann. *Theory of Self-Reproducing Automata.* A.W. Burks, Ed., Univ. Illinois Press, Urbana, 1966.

[Zurek, 1989] W.H. Zurek. Thermodynamic cost of computation, algorithmic complexity and the information metric. *Nature*, 341:119–124, 1989.