Entropy Production in Insertion-sort Algorithm

Dominik Strzałka

Rzeszów University of Technology
Wincentego Pola 2, 35-959 Rzeszów, Poland
strzalka@prz.edu.pl
http://strzalka.sd.prz.edu.pl

Abstract. This paper explicitly states and formalizes a link between statistical mechanics and theory of algorithms based on the example of insertion sort algorithm. The link is established between microstate of the system and configuration of loop executions during insertion sorting. Entropy connected with specific microstate is related to the entropy production of the algorithm. Because the algorithms are considered as Turing machines, presented analysis can be used as a basis for the investigations of problem of entropy production during work of physical implementations of Turing machines (i.e., computer systems) giving the possibilities to develop the idea of physics of processing.

Keywords: simple insertion sort algorithm; computational complexity; statistical mechanics; entropy; physics of processing

1 Introduction

The brilliant idea of Turing machine [1] became a basis of mathematical considerations in computer engineering [2]. The quick development of computer science in the sixties and the seventies of the XX$^{\text{th}}$ century especially in the case of problems with algorithms analysis is a great achievement of mathematical approach used in computer systems analysis. Between Turing machines and algorithms the equivalence is usually put [3], thus the analysis of algorithms behavior can be considered as the analysis of Turing machines. However, it should be noted that the whole problem, despite the fact that in many cases is quite (or even sometimes very) difficult, can be even more complicated when one takes into account the physical limitations of environment, in which the implementations of Turing machines work.

As it is well-known Turing machines are the mathematical models, but not the physical devices [4]. However, their implementations are the physical ones – as it was noted by Ch. Bennett [5]: "*Computers may be thought of as engines for transforming free energy into waste heat and mathematical work*". The traditional and broadly accepted definition of the term *machine* assumes that it is regarded as a physical system working deterministically in basic well-defined, physical cycles, built by the man, and intended to concentrate the dispersion of the energy in order to do some physical work [6]. Because each physical machine needs energy for work immediately the problem of entropy production appears. The whole problem will be shortly explained in Section 2, however in this place let's assume that the nowadays computer systems are mainly considered as Turing machines implementations (and not only, because they also work in interactive mode – see very interesting discussion in [7–10]). In Turing machine, the energy consumption for computing is equal 0, thus the problem of its efficiency η, considered in the case of all machines, doesn't appear. This may suggest that a mathematical model indeed itself hasn't got any connections with physical world (here it will be omitted the obvious connection, i.e., the implementations), but the assumed lack of energy consumption doesn't exclude the problem of entropy production.

The first physical implementations of Turing machines that were built in the middle of forties of the XX$^{\text{th}}$ century were machines that need for work huge amount of energy. For example one of the first famous computers, ENIAC, needs for its normal work 174 kW of electrical energy [12]. For comparison the nowadays computers, as the desktop ones, needs for example less that 300 W. It is rather obvious that not necessary the whole consumed energy is used for effective processing. One can check this when the hand will be put over the ventilator in computer – the hot air can be felt; some part of energy that is used for computer work is dissipated and isn't transformed in an useful form, i.e., isn't used for calculations.

In presented paper the problem of computers efficiency won't be considered. But basing on the Ch. Bennett work [5] and the properties of Turing machines the problem of entropy production during processing will be analyzed taking into account the behavior of insertion sort algorithm for the optimistic, pessimistic and mean cases. The whole investigations will be based on the thermodynamical analysis of the number of internal loop executions for successive keys.

2 Entropy in physical systems

The understanding of term *entropy* is inherently connected with the conception of *energy*, which is omnipresent in our life. The law of conservation of energy states that the difference between interior energy in system must be equal to the exterior energy, which is given to the system minus energy, which leaks from the system during transformation [13]. Thus there is a possibility to do the energy balance in system but each decrease or increase of energy amount should be noted. The rule expressed in this manner is usually called the first law of thermodynamics. The total amount of energy in any isolated system remains constant and cannot be created, although it may change forms.

The conception of energy and the rule of its conservation allows to write an equation, which will reflect the first law of thermodynamics, but they don't make any constraints on quantities used in such an equation. Moreover, they don't give any hints how the energy should be delivered or taken from the system or even what kind of laws (if they exist) should govern the energy transformation. The energy is not a some kind of substance or there is no possibility to calculate, in absolute values, the amount of energy inside the system. The meaning have the differences of transformed energy. But there are some laws that rule the energy transformations. The conception of entropy and the notions connected with it concerns these rules.

The term entropy was introduced by German physicist Rudolf Clausius around 1865, however it should be noted that this idea was developed during years and for the first time appeared in a little bit different form in 1854 in [14] where Clausius used the term "*equivalence-value*". This term is considered as a first name of entropy. The term entropy means "*a turning toward*". When Clausius proposed this term he stated that [15]: "*I propose to name the quantity S the entropy of the system, after the Greek word [$\tau\rho o\pi\epsilon$ – trope], the transformation. I have deliberately chosen the word entropy to be as similar as possible to the word energy: the two quantities to be named by these words are so closely related in physical significance that a certain similarity in their names appears to be appropriate.*" Clausis didn't explained why he chose the symbol S, but probably this was in honor of Sadi Carnot, who in article from 1824 [16] as a first man gave the first successful theoretical account of heat engines.

The entropy, similarly as the energy is a parameter that describes the system's state. The system feature, which is described by entropy, is a strictly abstract quantity and similarly to the energy it can't be "*seen*" or "*felt*". But entropy gives the description of energy distribution inside the system and about its possible transformations. Due to its connections with statistics and theory of probability the term entropy was generalized and is very useful in description of degree of organization or the level of order in system. The conception of entropy is closely connected with the second law of thermodynamics [17]. This law shows the way of each isolated system, which can be in states that do not decrease its entropy. British scientist Artur Edington in 1927 called this principle "*the arrow of time*" [18].

The successive properties of entropy won't be considered here – they can be found in literature (see for example [13, 17]). However, it should be noted that despite the fact that the conception of entropy was introduced in 1865 for many years there hadn't been any equation, which allow the direct calculations of entropy levels basing on molecular properties of analyzed system (for example the number of particles in gas). Because the entropy describes similarly to other parameters a some feature of system it was assumed that this property should be extensive (additive). In other words, if one considers two statistically independent systems **A** and **B**, which have the entropies $S(A)$ and $S(B)$, the entropy of system **A**+**B** should be equal [19]: $S(A+B) = S(A) + S(B)$.

To understand this assumption let's consider a very simple example – the process of possible distribution of particles in divided box. In this process there will be N particles, which will be inserted into a box. This box is divided into two equal parts and at the beginning all particles will be on its left side, while on the right side there will be no particles (obviously the arrangement can be reversed). The particles can freely move between parts and the whole problem is: find all possible configurations W of particles distribution. To solve this problem let's see that: if N_1 particles will be on the right side, the rest, i.e. $N_2 = N - N_1$, will be on the left side. Let, for example, $N = 10$; the first possibility is $N_1 = 0$ and $N_2 = 10$ and there is only one, $W = 1$ such a configuration. For $N_1 = 1$ and $N_2 = 9$ there are $W = 10$ configurations, while for $N_1 = 2$ and $N_2 = 8$ there are $W = 45$ configurations. As it turn-out the whole problem can be solved by the observation that in each case there is a need to calculate the combination of N_1-element subsets from N-element set, which can be given by

$$W = C_N^{N_1} = \frac{N!}{N_1!(N-N_1)!} = \frac{N!}{N_1!N_2!}. \tag{1}$$

Basing on the equation (1) the number of configurations W for other values of N_1 can be found in Table 1.

Table 1. Number of possible microstates W for $N = 10$

N_1	0	1	2	3	4	5	6	7	8	9	10
N_2	10	9	8	7	6	5	4	3	2	1	0
W	1	10	45	120	210	252	210	120	45	10	1

As it can be seen when $N_1 = N_2$ the number of possible microstates is the highest and this is the most probable state (configuration) of all. If one divides the box into two separate parts (subsystems **A** and **B**) for which the number of possible microstates will be computed, will see that for each configuration of states (W_A) in the system **A** there are possible all configurations (W_B) in the system **B** thus the total number of configurations W for system **A**+**B** can be computed as $W = W_A \cdot W_B$, i.e., W is a multiplicative quantity. Because the entropy was assumed as an additive quantity, while the number of microstates is multiplicative, the extensivity in entropy calculations can be guaranteed by $\ln(\cdot)$ because $\ln(W_A \cdot W_B) = \ln(W_A) + \ln(W_B)$ and at the end of XIX$^{\text{th}}$ Planck proposed the following formula for entropy calculations [19, 20] $S = k\ln(W)$, where k is a some positive constant. This formula is usually called Boltzmann-Gibbs entropy[1]. In the XX$^{\text{th}}$ century this approach was further developed by Shannon [21].

Because the state when $N_1 = N_2$ is the most probable it can be assumed that this is a natural, equilibrium state for the system. In this state the number of particles in both parts is equal and the symmetry appears. But from the other hand in this state one can also see the total disorder because particles are in both parts of box, while the state when all of them are on the right or left side (the asymmetry and order) is almost improbable (see Fig. 1). For systems with many particles all states with unequal distribution of particles are improbable. Because the state when $N_1 = N_2$ is the most probable it also expresses the natural limit of system evolution thus is also connected with the maximum of entropy.

[1] it is even engraved on Boltzmann's gravestone in Vienna

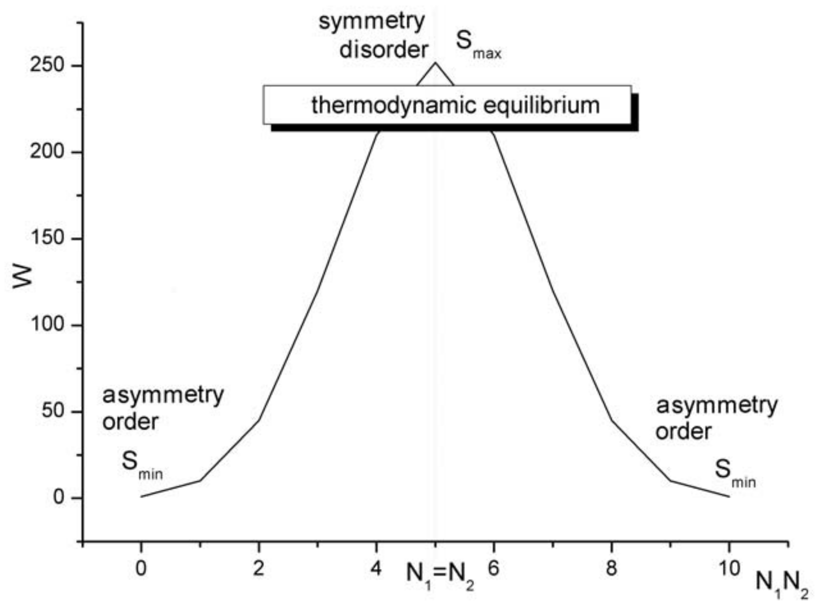

Fig. 1. Number of microstates W for $N = 10$

To introduce the order that was at the beginning (all particles on the left side) there is a need to do some work in order to move half of the particles from right to left. This obviously will decrease the level of entropy in a box, but something/someone must do this work. This requires a some portion of energy and as a result it will increase the entropy of environment outside the box. In other words to ensure that in a box there will be the order one needs increase the entropy elsewhere. The same situation can be related to the sorting problem. Because the sorting itself introduces a some kind of order into the sorted set it decreases the entropy in set but increases the entropy outside (see Section 4).

3 Insertion sort algorithm

A sorting is one of the most frequently used operation in computer systems [11]. This problem is algorithmically closed and there are well-known algorithms, which have the computational complexity equal $O(n \log n)$. However, the sorting is interesting itself and in some books usually is used as an introduction to the algorithms analysis (see for example [22]). There are many sorting algorithms and some of them are the direct adaptations of well-known in our life methods that were used for many years.

In our analysis we will consider the simple insertion sort algorithm (which main idea reflects the behavior of bridge player who sorts his cards) according the pseudocode presented in [22]. In this algorithm exist two loops: the external one, which guarantees that all elements in input set will be sorted and the internal one, which is responsible for finding a right place for each sorted key. The number of this loop executions is used for calculations of computational complexity, which for optimistic case (i.e. all elements in input set are in the right order) equals $\Omega(n)$, while for pessimistic case (i.e. all elements in input set are in the inverse order) equals $O(n^2)$, where n denotes the size of input set. Detailed analysis of computational complexity for the best and the worst cases can be found for example in [22]. Although it is known that for this algorithm the average complexity is $O(n^2)$ here the example of this analysis will be presented again to show some existing connections with the problem of entropy production during sorting.

The average case complexity will be defined as the amount of time that is needed for "*typical*" input data for a given n [23]. In calculations the following notation will be used:

- D_n – a set of input data instances with size n;
- d – one instance of input data;
- $T_d(n)$ – the number of needed operations for instance d with size n (usually the dominant operations are considered);
- X_n – a random variable with values $T_d(n)$ for $d \in D_n$;
- p_{nk} – a probability distribution of X_n (probability that the algorithm will do k dominant operations or equivalently $T_d(n) = k$ for input set with size n);

If the set D_n is finite each input instance is equiprobable. The calculations of average complexity are possible when one knows the probability distribution of X_n, which is defined as [23]

$$A(n) = \sum_{k \geq 0} k p_{nk}, \tag{2}$$

i.e., the average value of random variable X_n.

In the case of sorting the set D_n consist of all possible $n!$ permutations (instances) of n-element sets. If $n = 2$, then D_n will have two elements ($d = 2$), which will represent two cases: A[1] > A[2] and A[2] > A[1].

If the sorting is done according to the relation $\leq$ the first case denotes that in input set there is the inverse order (pessimistic case), while the second one that there is the optimistic one. Both cases can appear with $p = 0.5$ and the total time $T_d(n)$ for each case is equal: $T_1(2) = 1$ and $T_2(2) = 0$. This means that for input set with size $n = 2$ the algorithm will do $k = 0$ or $k = 1$ dominant operations with the same probability $p_{20} = p_{21} = 0.5$. Thus the average computational complexity according to the equation (2) is equal: $A(n = 2) = 0.5 \cdot 0 + 0.5 \cdot 1 = 0.5$.

For $n = 3$ the set D_n consists of $d = 6$ elements that can appear with the probability $p_n = 1/6$ and the average complexity equals: $A(n = 3) = 1.5$.

Nothing can be said about the behavior of $A(n)$ having the information about two cases. Thus for $n = 1 \ldots 10$ a Table 2 was made where one can also find: $A'(n) = A(n) - A(n-1)$ for $n = 2, 3, \ldots$, $A''(n) = A'(n) - A'(n-1)$ for $n = 3, 4, \ldots$ and also $B(n) = \max\{T_d(n)\}$, $B'(n) = B(n) - B(n-1)$ for $n = 2, 3, \ldots$, $B''(n) = B'(n) - B'(n-1)$ for $n = 3, 4, \ldots$.

Table 2. The values of $A(n)$, $A'(n)$, $A''(n)$, $B(n)$, $B'(n)$, $B''(n)$

n	$A(n)$	$A'(n)$	$A''(n)$	$B(n)$	$B'(n)$	$B''(n)$
1	0	–	–	0	–	–
2	0.5	0.5	–	1	1	–
3	1.5	1	0.5	3	2	1
4	3	1.5	0.5	6	3	1
5	5	2	0.5	10	4	1
6	7.5	2.5	0.5	25	5	1
7	10.5	3	0.5	31	6	1
8	14	3.5	0.5	38	7	1
9	18	4	0.5	46	8	1
10	22.5	4.5	0.5	55	9	1

Basing on information from Table 2 one can see that $A'(n)$ is a linear function ($A''(n) = const$). Thus one may expect that $A(n)$ will be given by $a \cdot n^2 + b \cdot n + c$. The coefficients a, b, c can be calculated basing on method that was given in [24] and for the insertion sort algorithm $A(n)$ is given by

$$A(n) = \frac{n^2 - n}{4} \tag{3}$$

and this function is $O(n^2)$.

Let's note that the computational complexity for pessimistic case is given by [23]

$$B(n) = \sup\{T_d(n) : d \in D_n\}, \tag{4}$$

which, taking into account the last column in Table 2, gives [22]

$$B(n) = \frac{n(n-1)}{2}. \tag{5}$$

Thus the number of dominant operations in average case (given by the equation (3)) is half of the number in pessimistic one (given by the equation (5)).

4 Entropy production in insertion sort algorithm

It is obvious that sorting can be related to the introduction of some kind of order in input set, because sorting is based on one chosen ordering relation [11]. From physical point of view this means that the energy, which is delivered to the machine that sorts, is also used to decrease the existing in input set entropy, i.e., the entropy from input set is carried out to the environment. Because the whole analysis will be done for a mathematical model of algorithm, i.e. Turing machine, it can be assumed that the entropy production will be only caused by ordering. The insertion sorting is a very simple procedure and its main idea can be presented as follows: the successive keys are compared one by one in internal loop with the so far sorted elements (from the right to the left) and when the right place is found another key is sorted. This means that for each key the number of internal loop executions can be different and is governed by [25]:

- value of sorted key;
- number of so far sorted keys;
- values of so far sorted keys.

This leads to the possible dynamical behavior of this algorithm and the existing dynamics can be described in terms of entropy, which gives the statistical mechanics basis for any considerations. The whole analysis can be done as follows [26]:

Let n denotes a size of sorting set and n_i denotes the position of successively sorted elements in input set, $i = 1 \dots n$. The total number of all possible executions of internal and external loops needed for sorting the successive elements from input set will be denoted by M. For each key its maximal value can be obtained in the pessimistic case, thus for inverse order in input set $M = n_i$. The number of executions of external loop will be denoted by M_1 – in analyzed algorithm for each key there is only one execution of this loop thus $M_1 = 1$. M_2 will represent the number of necessary internal loop executions thus it can change from 0 in optimistic case to $n_i - 1$ in pessimistic case. Because the algorithm sorts not only pessimistic cases but also, for input size n, many other instances by M_3 it will be denoted the number of such possible internal loop executions that can appear but don't appear due to the some properties of input set. Obviously it is clear that $M = M_1 + M_2 + M_3$.

To define the amount of entropy that will be removed from input set for each sorted key it will be calculated the number W of possible configurations of loops executions that can appear during sorting. Similarly to the example presented in Section 2 this quantity can be calculated as the number of all possible combinations $C_M^{M_1}$ multiplied by $C_{M-M_1}^{M_2}$ [26], thus one has

$$W = C_M^{M_1} \cdot C_{M-M_1}^{M_2} = \frac{M!}{M_1!\,(M-M_1)!} \cdot \frac{(M-M_1)!}{M_2!\,(M-M_1-M_2)!} = \frac{M!}{M_1!M_2!M_3!}. \tag{6}$$

It seems that in the optimistic case the entropy production should be on the lowest possible level due to the fact that in input set there is the order that is required. Indeed in this case one has one execution of the external loop, thus $M_1 = 1$, no executions of the internal loop ($M_2 = 0$) but this also means that the internal loop could be executed, but it won't be, $M_3 = n_i - 1$ times. Thus in optimistic case for each key W_O equals

$$W_O = \frac{n_i!}{1!0!\,(n_i-1)!} = n_i. \tag{7}$$

In the pessimistic case one may expect that the situation should be different, but let's note that for each sorted key there will be one execution of external loop ($M_1 = 1$), $M_2 = n_i - 1$ executions of internal loop thus $M_3 = 0$. The number of microstates for each sorted key in pessimistic case W_P will be equal

$$W_P = \frac{n_i!}{1!\,(n_i-1)!0!} = n_i. \tag{8}$$

The results obtained above are little bit surprising, but the whole situation can be quickly explained. Let's note that from thermodynamical point of view in pessimistic case in input set there is the inverse order and this is still some kind of order. Moreover, at this place it should be also noted that taking into account the Onsager relations [17, 27], which stated that the system is in (quasi)equilibrium state when the entropy production is on the lowest possible level, in insertion sort algorithm exist such states for optimistic and pessimistic cases.

To see the number of microstates W in other cases let's consider the following example: only one excess dominant operation for key n_i will be needed, i.e.: $M_1 = 1$, $M_2 = 1$, $M_3 = n_i - 2$ thus

$$W = \frac{n_i!}{1!1!\,(n_i-2)!} = \frac{(n_i-2)!\,(n_i-1)\,n_i}{(n_i-2)!} = n_i\,(n_i-1) \tag{9}$$

and in this case the number W is greater than in W_O or W_P cases. Thus from thermodynamical point of view the pessimistic and optimistic cases are the same, while other ones leads to the greater entropy production. This can be also confirmed when one analyzes the increments of number of internal loop executions for each successive key n_i needed for their sorting. In optimistic and pessimistic cases they are constant indicating that there is no dynamics in algorithm behavior and processing can be compared to the "*laminar flow*", while for other cases the flow can be a "*turbulent*" one [26].

As it can be noted the maximum of entropy production will be obtained when $M_2 = M_3$, i.e., when the number of internal loop executions will be equal half of the all possible executions ($(n_i - 1)/2$). For some values of n this number can be found in Table 3 where the entropy was calculated using the Boltzmann-Gibbs formula, $S = k \ln W$ with $k = 1$.

Plotting S versus n_i it can be seen (Fig. 2) that in the average case (i.e. $M_2 = M_3$) the entropy growth tends to be the linear one for $n_i \to \infty$.

This can be confirmed if one notes that when $M_2 = M_3$ the number of possible loop configurations W for each key n_i can be obtained as

$$W = \frac{n_i!}{\left(\frac{n_i-1}{2}\right)! \cdot \left(\frac{n_i-1}{2}\right)!}$$

for $i = 1, 3, 5, \ldots$. Thus the entropy S can be calculated as

$$S = \ln\left(\frac{n_i!}{\left(\frac{n_i-1}{2}\right)! \cdot \left(\frac{n_i-1}{2}\right)!}\right) = ln\,(n_i!) - 2\ln\left(\left(\frac{n_i-1}{2}\right)!\right). \tag{10}$$

Table 3. Entropy production in average case, i.e. $M_2 = M_3$

key number n_i	number of microstates W	entropy $S = \ln(W)$
2	2	0.693147181
4	6	1.791759469
6	20	2.995732274
8	70	4.248495242
10	252	5.529429088
12	924	6.828712072
14	3432	8.140898461
16	12870	9.462654301
18	48620	10.79179025
20	184756	12.12679131
22	705432	13.46656566
24	2704156	14.81030041
26	10400600	16.15737405
28	40116600	17.50730077
30	155117520	18.85969358

Let's assume that $k = n_i - 1$, thus $n_i = k+1$ and the equation (10) will be given by

$$S = \ln\left((k+1)!\right) - 2\ln\left(\left(\frac{k}{2}\right)!\right). \tag{11}$$

Because

$$\ln\left((k+1)!\right) = \sum_{j=1}^{k+1}\ln(j) \approx \int_1^{k+1}\ln(x)dx = x\ln x - x\Big|_1^{k+1} = (k+1)\ln(k+1) - k$$

and also

$$\ln\left(\left(\frac{k}{2}\right)!\right) = \sum_{j=1}^{\frac{k}{2}}\ln(j) \approx \int_1^{\frac{k}{2}}\ln(x)dx = x\ln x - x\Big|_1^{\frac{k}{2}} = \frac{k}{2}\ln(\frac{k}{2}) - \frac{k}{2} + 1,$$

the equation (11) for $n_i \to \infty$ can be written as

$$\begin{aligned} S &\approx (k+1)\ln(k+1) - k - 2\cdot\left(\frac{k}{2}\ln\left(\frac{k}{2}\right) - \frac{k}{2} + 1\right) \\ &= (k+1)\ln(k+1) - k - k\cdot\ln\left(\frac{k}{2}\right) + k - 2 \\ &= k\ln(k+1) - k\ln\left(\frac{k}{2}\right) + \ln(k+1) - 2 \\ &= k\cdot\ln\left(\frac{2(k+1)}{k}\right) + \ln(k+1) - 2 \\ &= (n_i - 1)\ln\left(\frac{2n_i}{n_i - 1}\right) + \ln(n_i) - 2 \end{aligned}$$

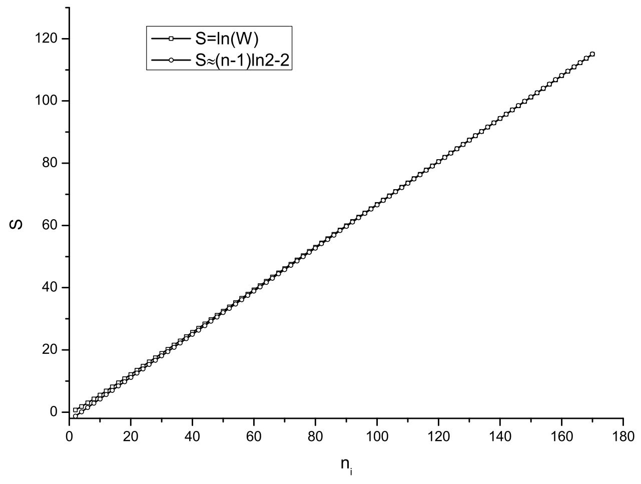

Fig. 2. Entropy S production for successive keys n_i (continues line with open squares) with the approximation given by the equation (13) – line with open circles

$$\approx (n_i - 1)\ln\left(\frac{2n_i}{n_i - 1}\right) - 2. \tag{12}$$

Due to the fact that

$$\lim_{n_i \to \infty} \frac{2n_i}{n_i - 1} = 2,$$

the equation (12) can be written as

$$S \approx (n_i - 1)\ln 2 - 2, \tag{13}$$

which is the approximation of entropy production given by the equation (10) and as it can be seen (Fig. 2) the level of entropy production for each key n_i is approximately represented by the linear function $a \cdot n + b$. It should be noted that this is again consistent with Onsager reciprocal relations [17, 27] and indicates that in the system exists the thermodynamic non-equilibrium state with maximum but linear entropy production thus the algorithm behavior in this case can be analyzed by the classical extensive thermodynamics.

The equation (6) shows the number of possible microstates for each successive sorted key. Assuming that between the successive keys there aren't any dependencies, especially those ones, which can indicate the existence of correlations between values of successive keys, for each key n_i the produced entropy S is extensive. For average case it grows linear thus the total amount of produced entropy during sorting such a case by insertion sort can be expressed as: $a \cdot n^2 + b \cdot n + c$, which indicates the direct connection between the problem of entropy production and the average case computational complexity of insertion sort algorithm. It is a very interesting problem itself because the question arises: do such connections exist for different sorting algorithms, however the answer requires further investigations.

5 Conclusions

In the paper it has been shown that between the mathematical analysis of computational complexity for insertion sort algorithm and the statistical mechanics in physics exist direct connections. It can be rather a surprising fact, but it should be noted that despite the fact that Turing machine is a mathematical model, while the statistical mechanics is usually related to the physical systems, such connections can be made even when the assumption about zero energy consumption by Turing machines is assumed. Moreover, the presented analysis also confirms that in an "*average*" case, when the behavior of analyzed algorithm is considered basing on given probability distributions as a some kind "*expected value*" for $n \to \infty$, the connections with classical statistical mechanics based on Boltzmann-Gibbs entropy are very strong.

References

1. Turing, A. M.: On computable numbers, with an application to the Entscheidungsproblem, Proc. of the London Mathematical Society, Series 2, 42 (1936), pp 230–265. Errata appeared in Series 2, 43, pp. 544–546 (1937)
2. Wegner, P.: Research paradigms in computer science, Proc. of the 2nd Int. Conf. on Soft. Eng., San Francisco, California, pp. 322–330 (1976)
3. Papadimitriou, Ch.: Computational Complexity. Addison Wesley (1993)
4. Penrose, R.: The Emperor's New Mind, 2nd edition, Oxford University Press, New York (1990)
5. Bennett, Ch. H.: The Thermodynamics of Computation – a Review, Int. J. of Theor. Phys., vol. 21, No. 12, pp. 905–939 (1982)
6. Horàkowà, J., Kelemen, J., Čapek, J.: Turing, von Neumann, and the 20th Century Evolution of the Concept of Machine. Int. Conf. in Memoriam J. von Neumann, J. von Neumann Comp. Soc., p. 121, Budapešt́ (2003)
7. Wegner, P., Goldin, D.: Computation Beyond Turing Machines, Comm. of the ACM, vol.46, No. 4 (2003)
8. Eberbach, E., Goldin, D., Wegner, P.: Turing's Ideas and Models of Computation, in Alan. Turing: Life and Legacy of a Great Thinker, Teuscher, Ch. (ed.), Springer-Verlag,. Berlin, Heidelberg, New York (2005)
9. Goldin, D., Wegner, P.: The Church-Turing Thesis: Breaking the Myth, LNCS 3526, Springer, p. 152 (2005)
10. Wegner, P.: Why Interaction is More Powerful Than Algorithms, Comm. of the ACM 40(5), p. 81 (1997)
11. Knuth D.: The art of Computer Programming, vol.1-3, Addison-Wesley Innovations (1968-73)
12. Weik, M. W.: A Survey of Domestic Electronic Digital Computing Systems, Ballistic Research Laboratories, Report No. 971 (1955)

13. Sonntag, R. E., van Wylen, G. J.: Introduction to Thermodynamics, Classical and Statistical, Wiley (1991)
14. Clausius, R.: Fifth Memoir: On the Application of the Mechanical theory of Heat to the Steam-Engine in The Mechanical Theory of Heat – with its Applications to the Steam Engine and to Physical Properties of Bodies: p. 215, London: John van Voorst, Paternoster Row. (1865)
15. Laidler, K. J.: The Physical World of Chemistry, Oxford University Press (1995)
16. Carnot, S.: Réflexions sur la puissance motrice du feu et sur les machines propres a développer cette puissance, Bachelier, Paris, (1824); translated by R. H. Thurston: Reflections on the motive power of heat and on machimnes fitted to develop this power, New York, The American Society of Mechanical Engineers (1943)
17. Prigogine, I., Stengers, I.: Order out of Chaos: Man's new dialogue with nature, Flamingo (1984)
18. Eddington, A.: The Nature of the Physical World, MacMillan (1928)
19. Tsallis, C.: Nonextensive Statistics: Theoretical, Experimental and Computational Evidences and Connections, Brazilian Journal of Physics, vol. 29, Issue 1, p. 1 (1999)
20. Tsallis C., Baldovin F., Cerbino R., Pierobon P.: Introduction to Nonextensive Statistical Mechanics and Thermodynamics, arXiv:cond-mat/0309093v1 [cond-mat.stat-mech]
21. Shannon, C. E.: A mathematical theory of Communication, Bell System Technical Journal, vol. 27, pp. 379–423 (1948)
22. Cormen, T. H., Leiserson, Ch. E., Rivest, R. L., Stein, C.: Introduction to Algorithms, MIT Press (1994)
23. Banachowski, L., Diks, K., Rytter, W.: Algorithms and data structures, WNT, Warsaw (1996)
24. Sawyer, W. W.: The search for pattern, Penguin (1970)
25. Grabowski, F., Strzałka, D.: Dynamic behavior of simple insertion sort algorithm. Fundam. Inform. 72, p. 155 (2006)
26. Strzałka, D., Grabowski, F.: Towards possible non-extensive thermodynamics of algorithmic processing – statistical mechanics of insertion sort algorithm, Int. J. of Mod. Phys. C, vol. 19, n. 9, pp. 1443–1458, (2008)
27. Onsager, L.: Reciprocal Relations in Irreversible Processes. I., Phys. Rev., vol. 37(4), p. 405 (1931)