Application of the Computer Capacity to the Analysis of Processors Evolution

Boris Ryabko[1,2] and Anton Rakitskiy[1,2,3]

April 1, 2019

Abstract

The concept of the so-called computer capacity was proposed in 2012 and applied for analysis of processors of different kinds. Here we analyze the evolution of processors using the computer capacity as the main tool of analysis. It is shown that during the transition "from old to new" the manufacturers change the parameters that affect the computer capacity. It allows us to predict the values of parameters of following processors. Intel processors are used as the main example due to their high popularity and the accessibility of detailed description of all the technical characteristics.

Keywords: Information theory, computer capacity, processors, evolution of computers, performance evaluation
This work was supported by Russian Foundation for Basic Research (grant 18-29-03005).

[1]Institute of Computational Technologies of the Siberian Branch of the RAS
[2]Novosibirsk State University
[3]Siberian State University of Telecommunication and Information Sciences

1 Introduction

Processors are widely used in computers, mobile phones and many other devices and their performance has grown a thousandfold for the last 20 years. In the context of such rapid development it is interesting to find the way of processors evolution. For this, one needs to be able to estimate the influence of changing some characteristics of a processor on its performance.

Nowadays designers must have a real model and set of benchmarks in order to estimate the performance of the developed processor. It require some time to prepare physic model of processor and spend time to calculations. A theoretical approach to estimating computer capacity was suggested in [1]. This approach uses only the description of the investigated computer architecture. The latter includes the set of instructions, features of their execution, sizes of all the memory types, etc. This, no experiments with a working model of the computer are necessary. This method of estimating the computer capacity was applied to the performance evaluation of a large number of different Intel and AMD processors [2, 3]. In [3] we show that this characteristic is consistent with the values achieved experimentally by using benchmarks [4, 5, 6] (some of the results are presented in the appendix 1). This method was also used to estimate the performance of some supercomputers from TOP500 list [7].

In this paper we consider the evolution of Intel processors for the last 15 years. We show that each processor can be represented as a set of parameters and changing some of these parameters has a significant effect on its performance. Our investigation shows that in new processors the manufacturers usually increase the characteristics which affect the computer capacity the most. This, in turn, allows us to predict the direction of changes in the evolution of computers.

Note that there are clear tendencies which are traced in the development of processors for the last 15 years. The first obvious tendency is the increase of the clock rate which causes the decrease of the task execution time. The second one is the widespread introduction of parallelism, in particular: by increasing the number of processors in computer; by increasing the number of computing cores in processor; by introducing threads and pipelines etc. It is clear that the effect of these parameters is huge, but it is also obvious.

The emphasis in our work is on the quantitative estimation of the impact of the parameters whose role in the performance is not so obvious. These parameters include: sizes and access times to different kinds of memory (including registers, cache-memory etc.); the number of different instructions of certain type (instruction types are divided by the number and kind of the operands). In addition, we show how the computer capacity can be applied at the design stage of the processor. In this paper, we consider the "evolutionary series" of Intel processors. These processors have been developed for several decades, the information on their architectures is completely open and well researched. We show that in fact, when releasing new processors, designers are changing exactly those parameters that significantly affect the computer capacity. This is why we consider the processors of one company, assuming that designers maintain continuity and release new processors improving previous.

For example, if we examine the Intel processor codenamed Wolfdale, we see that the increase of some parameters significantly affects performance, whereas changing the rest of the parameters has almost no effect. For example: if we

increase the size of level-1 cache memory in Wolfdale the value of the computer capacity almost does not change, but if the number of internal vector registers is increased, the growth of the computer capacity becomes perceptible. It turns out that in the processors of the succeeding microarchitectures (Sandy Bridge, Ivy Bridge etc.) these exact parameters were increased. This allows us to claim that using the suggested method may be useful for predicting which parameters will be changed in succeeding the future models. Moreover, it proves that the computer capacity can be used at the design stage to solve the problem of architecture optimization. Since the computer capacity is multi-variable function we can reduce the solution of the architecture optimization problem to the problem of finding the function maximum. In other words, we can easily change the parameters and estimate how it affects the computer capacity to improve the processor performance.

2 Computer Capacity

All the theory behind the computer capacity and its estimation was described in details in the previous work [1], so here we only present the main definitions and a brief summary of the theory required.

Let us consider a computer with the set of instructions I and memory M. An instruction $x \in I$ is formed as the combination of its name and the values of its operands (two instructions with the same names and different values of operands are both included in I). For example, instructions *mov eax, ebx* and *mov eax, ecx* are different and included in I independently. A computer task X is a sequence of instructions $X = x_1, x_2, \ldots, x_i \in I$. It is important to note that if there is a loop in task which is repeated m times, the body of this loop included m times in X. We denote $\tau(x)$ the execution time of instruction x. So the execution time of a computer task X is given by $\tau(X) = \sum_{i=1}^{n} \tau(x_i), X = x_1, x_2, \ldots, x_n$. Let consider the number of all possible computer tasks which execution times equal to T as $N(T) = |\{X : \tau(X) = T\}|$.

Furthermore, let there be a processor which has exactly N_1 different tasks with execution times equal to, for example, 1 hour. In this case we can say that it can execute N_1^2 different tasks in 2 hours because if X_1 and X_2 are 1-hour tasks, the combined task S_1S_2 is the 2-hour one (we did not take into account the 2-hour tasks with instruction starts at the end of the first hour and finishes and the beginning of the second). In this way, the considered processor has $\approx N_1^k$ tasks with execution times k hours. So we claim that the number of possible tasks grows exponentially as a function of time ($N(T) \approx 2^{CT}$). Thereby, $C = \frac{\log(N(T))}{T}$ (or rather the limit of this value) is the adequate measure of the computer capacity and it defines as follows:

$$C(I) = \lim_{T \to \infty} \frac{\log N(T)}{T}. \tag{1}$$

The main question here is how to estimate the value of $C(I)$ from (1). The direct calculation of the limit is impossible, but there exists the method of calculation the capacity $C(I)$ in combinatorial analysis. In this case we consider the set of instructions I as an alphabet and assume that all words (sequences of instructions) over this alphabet are possible (can be executed). This assumption allows us to estimate an upper-bound of the computer capacity, because for any

processor the set of its permissible tasks is the subset of all possible tasks. Here, all execution times are integers (this statement is valid for the most of processors). The way of estimation of the capacity was suggested by C. Shannon [8], who showed that the capacity $C(I)$ is equal to the logarithm of the largest real solution Y_0 of the following characteristic equation:

$$Y^{-\tau(x_1)} + Y^{-\tau(x_2)} + \cdots + Y^{-\tau(x_n)} = 1. \quad (2)$$

In [1] it was also shown that the computer capacity of multi-core processing unit can be defined as the sum of capacities of the cores.

3 The computer capacity of Intel processors

The present work is based on the analysis of Intel processors for the last 15 years, because the information about these processors architectures (with the full description of instruction set) is public and easily accessible. In our previous works we have shown that the computer capacity correlates well with the values of benchmarks [9, 6] and can be used as a measure of computer performance [3].

We have identified the list of processors to analyze: Pentium M (Dothan processor), Intel Core (Wolfdale), Ivy Bridge, Haswell and Skylake. In this paper we analyze these processors in details, evaluate their computer capacity values and perform some investigations about the effect of parameters on performance. Pentium M and Wolfdale architecture differs strongly from Ivy Bridge, Haswell and Skylake, so the comparison of this architectures would be quite interesting. Skylake is considered here because it is the latest Intel microarchitecture with the published detailed description (at the moment of this paper preparation). Here, we present the details of the calculation of the computer capacity for Intel Pentium M, Core, Ivy Bridge, Haswell and Skylake microarchitectures. The structure of the described processors pipelines are similar to each other and the features of building the equation (1) for that structure are presented in [2, 3].

In table 1 we present the summary of technical characteristics and the values of the computer capacity for all the chosen processors. Hereinafter L1,L2,L3 - the size of level-1, level-2 and level-3 caches, M - the size of RAM, $L1_t, L2_t, L3_t, M_t$ - the memory latencies for level-1, -2, -3 and RAM respectively, c.c. - clock cycles, R_i - the number of integer registers, R_v - the number of vector (floating point) registers. The detailed descriptions and the lists of instructions of the described microarchitectures are presented in [10]. The characteristic equations of all the described processors and all the tools for evaluation of the computer capacity can be found in [11].

4 Analysis of Intel processors evolution

To analyze the evolution of Intel processors we divide them into 5 groups: (Pentium M, Intel Core), (Intel Core, Ivy Bridge), (Ivy Bridge, Haswell), (Haswell, Skylake) and (Skylake and the prediction of its successor). The investigated characteristics are the following:

- Physical characteristics of processor: the size of all memory types; the access time of all memory types; the number of different registers. Here,

Table 1: The summary of characteristics for selected processors

Processor	Pentium M	Intel Core	Ivy Bridge	Haswell	Skylake
L1, KB	32	64	64	64	64
L2, KB	2048	6144	256	256	256
L3, MB per core	-	-	2.5	2	1.375
M, GB	1	16	16	16	16
$L1_{time}$, c.c.	3	3	4	4	4
$L2_{time}$, c.c.	10	15	12	12	12
$L3_{time}$, c.c.	-	-	30	36	42
M_{time}, c.c.	70	24	30	36	42
R_i	8	16	160	168	180
R_v	8	16	144	168	168
Computer capacity, bits/c.c.	51.217	70.898	108.587	115.86	116.208

the memory types are the 1-level cache, the 2-level cache, the 3-level cache (if processor contains this level) and the main memory (RAM). Registers are also of two types: integer and vector.

- The instructions set is also a characteristic which affects the performance. It can be observed that the instruction set changes from one processor to another. Here, we isolate the fastest instructions from instruction set of each examined processor and calculate the amounts of them. The fastest instruction is the one whose execution time equals 1 and does not have memory cells in the list of operands (access to the memory cell greatly increases the execution time of an instruction).

We group all instructions by the number of operands and present the results in table 2. The names of columns signify the number of operands in the investigated instructions: the number of instructions with single operand presented in second column (for example, instruction "PUSH R"), with two operands (example is "MOV R1 R2") and with three operands ("MULX R32 R32 R32"). In most cases, these operands are registers of different types.

Table 2: Number of different instructions

	1	2	3
Pentium M (Dothan)	53	91	-
Intel Core (Wolfdale)	67	328	-
Ivy Bridge	21	99	10
Haswell	31	113	44
Skylake	35	134	48

We examined the influence of changing the value of a single characteristic (from the physical group) and identify those of them which have a nonzero influence on the value of the computer capacity. We also examined all the possible pairs of different characteristics, but only one pair (R_i,R_v) was included

because all other pairs either have no effect on the value of the computer capacity or their effect equals the effect of single characteristic. The changes of values are performed as $\times 0.5, \times 2, \times 5, \times 10, \times 20$ of the original value.

Next we try to add instructions in the instruction set and to show the influence of this addition on the value of the computer capacity. The number of instructions in the modified instruction set is obtained as the original value $\times 1.1, \times 1.25, \times 1.5, \times 2$. The last step is the combination of the two previous steps. Here, we want to show the influence of increasing the number of registers and adding instructions of a new type simultaneously. It is shown above that in Ivy Bridge processors a new type of instructions appeared instructions with three register operands, and at the same time the number of registers is increased almost 10 times in relation to Intel Core (Wolfdale) processors. So, in this part we add the instructions of new type: for Pentium M and Intel Core we add instructions with 3 operands, for the remaining processors we add instructions with four operands. In the experiments we add 8, 16, 32 and 64 instructions. Increasing the number of registers is performed in the following way: $\times 2, \times 5, \times 10$ of the original value. Following is the list of added instructions types in the second and third steps:

1. cmd r 1 / cmd x 1 - the instruction with the name "cmd", with a single integer register operand (r) or vector register (x) and with the execution time equals to 1;

2. cmd r,r 1 / cmd x,x 1- the instruction with two integer or two vector register operands;

3. cmd r,r,r 1 / cmd x,x,x 1 - the instruction with three integer or three vector register operands, codenamed as cmd1 and cmd2 respectively in the last step.

4. cmd r,r,r,r 1 / cmd x,x,x,x 1 - the instruction with four integer or four vector register operands, codenamed as cmd3 and cmd4 respectively in the last step.

All the results in tables are presented in percent relative to the original value of computer capacity. It is important to note that we merge the characteristics with the same results into a single row.

4.1 Pentium M and Intel Core

The results of the analysis of Pentium M are presented in tables 3,4 and 5, one for each step of investigation. The first row is filled with 100 and it means that the characteristics from this row have no influence on the computer capacity. So the role of the size and the access time of cache-memory and RAM are insignificant for the computer capacity. In the following subsections we exclude from the tables those characteristics that do not affect the value of computer capacity. We can also observe that the increase in the number of instructions has no significant effect (for the instructions of existing types). However, there are some characteristics which change the value of the computer capacity by more than 1%. Obviously, to increase the capacity, we need to increase the number of registers (integer or vector) and add some instructions of a new type. As we can see in Tables 1 and 2 , the number of registers was doubled in Intel

Core, the size of memory of different kinds was also increased, but the access times are almost unchanged. Increase in the Intel Core computer capacity is also explained by the improvement of the throughput of its pipeline (it grew from 3 μops per cycle to 4).

Table 3: Pentium M step 1

Variable parameter	0.5	2	5	10	20
$L1, L2, M,$ $L1_t, L2_t, M_t$	100	100	100	100	100
R_i	99.92	100.24	101.58	104.95	111.91
R_v	99.78	100.78	104.75	111.87	121.88
R_i & R_v	99.72	101.03	105.97	113.95	124.39

Table 4: Pentium M step 2

Instruction type	1.1	1.25	1.5	2
"cmd r 1", "cmd x 1"	100.002	100.006	100.013	100.026
"cmd r,r 1", "cmd x,x 1"	100.035	100.086	100.175	100.35

Table 5: Pentium M step 3

Combination of instructions and additional registers	8	16	32	64
$r \times 2$ cmd1, $x \times 2$ cmd2	102.644	103.999	106.191	109.372
$r \times 5$ cmd1, $x \times 5$ cmd2	114.767	118.985	123.885	129.219
$r \times 10$ cmd1, $x \times 10$ cmd2	130.027	135.229	140.739	146.417

4.2 Intel Core and Ivy Bridge

In tables 6,7 and 8 the results are close to those for the previous processor, except for the influence of adding the instructions with 2 operands. We noted that the 10-20 fold increase of the number of registers and adding the instructions with three operands gives the best effect on the value of computer capacity. So the manufacturer change exactly this characteristics in the succeeding processor. In Ivy bridge the number of integer and vector registers increased tenfold and 10 fast instructions with 3 register operands were added.

4.3 Ivy Bridge and Haswell

The results for Ivy Bridge analysis are presented in tables 9, 10 and 11. Here, we observe tendencies similar to those in the previous processors. It is interesting to observe that starting from Ivy Bridge the characteristics related with cache-memory and RAM are almost unchanged. We can also notice that starting with Ivy Bridge the effect of the number of commands with one register becomes insignificant and we exclude them from the tables of the following subsections.

Table 6: Intel Core step 1

Variable parameter	0.5	2	5	10	20
R_i	99.73	100.79	104.36	110.67	119.84
R_v	97.48	105.81	118.08	128.87	140.01
R_i & R_v	97.18	106.3	119	129.91	141.09

Table 7: Intel Core step 2

Instruction type	1.1	1.25	1.5	2
"cmd r 1", "cmd x 1"	100.004	100.01	100.02	100.04
"cmd r,r 1", "cmd x,x 1"	100.302	100.753	101.442	102.667

Table 8: Intel Core step 3

Combination of instructions and additional registers	8	16	32	64
$r \times 2$ cmd1, $x \times 2$ cmd2	109.908	112.401	115.849	120.124
$r \times 5$ cmd1, $x \times 5$ cmd2	127.474	131.534	136.253	141.391
$r \times 10$ cmd1, $x \times 10$ cmd2	142.739	147.493	152.651	158.04

4.4 Haswell and Skylake

Haswell processors (tables 12,13 and 14) were improved by increasing the number of registers and by adding some instructions of the existing types. We can observe that most of investigated characteristics are unchanged for Ivy Bridge, Haswell and Skylake. The main tendencies is for increasing the number of registers (but not as much as for Intel Core), changing the instruction set and making some improvements of the processor pipeline.

4.5 Skylake and the prediction of its successor

The results obtained for this last processor from our list (tables 15,16 and 17) present the direction for making predictions for succeeding processors. We can note that as far as the characteristics from technical group are concerned, the biggest effect on the computer capacity is reached by increasing the number of vector registers. To achieve the effect close to the Intel Core – Ivy Bridge capacity increase we need to increase tenfold the number of registers and to add a large number of new fast instructions with four registers..

The first step of our research shows that some parameters starting from the first examined processor do not affect the performance. These parameters are sizes of all types of memory and theirs latencies. Certainly, we do not want to claim that these parameters are useless and can be freely removed, but in the previous models the level of saturation was reached for these parameters, so increasing them more is inefficient. Also we isolate the parameters which have a significant influence on the value of computer capacity and that have been changed substantially during the evolution process. In fact, registers of a processor is the fastest memory (they can be accessed nearly instantly), so it

Table 9: Ivy Bridge step 1

Variable parameter	0.5	2	5	10	20
R_i	99.96	100.17	101.16	103.66	108.57
R_v	89.3	111.16	126.05	137.35	148.66
R_i & R_v	89	111.18	126.06	137.36	148.67

Table 10: Ivy Bridge step 2

Instruction type	1.1	1.25	1.5	2
"cmd r 1","cmd x 1"	100	100	100	100
"cmd r,r 1"	100.013	100.035	100.072	100.144
"cmd x,x 1"	100.011	100.0297	100.058	100.117
"cmd r,r,r 1"	100	100.431	101.049	101.945
"cmd x,x,x 1"	100	100.334	100.799	101.496

Table 11: Ivy Bridge step 3

Combination of instructions and additional registers	8	16	32	64
$r \times 2$ cmd3	136.298	140.419	144.358	148.213
$x \times 2$ cmd4	134.032	138.137	142.07	145.921
$r \times 5$ cmd3	156.223	160.361	164.309	168.168
$x \times 5$ cmd4	153.939	158.072	162.016	165.874
$r \times 10$ cmd3	171.31	175.454	179.404	183.265
$x \times 10$ cmd4	169.02	173.162	177.111	180.97

Table 12: Haswell step 1

Variable parameter	0.5	2	5	10	20
R_i	99.71	101.85	110.89	120.75	131.04
R_v	91.57	110.03	123.66	134.01	144.37
R_i & R_v	89.72	110.32	123.99	134.34	144.7

Table 13: Haswell step 2

Instruction type	1.1	1.25	1.5	2
"cmd r,r 1","cmd x,x 1"	100.003	100.008	100.015	100.03
"cmd r,r,r 1","cmd x,x,x 1"	100.177	100.473	100.904	101.669

Table 14: Haswell step 3

Combination of instructions and additional registers	8	16	32	64
$r \times 2$ cmd3, $x \times 2$ cmd4	125.817	129.492	133.053	136.558
$r \times 5$ cmd3, $x \times 5$ cmd4	143.936	147.683	151.277	154.798
$r \times 10$ cmd3, $x \times 10$ cmd4	157.7	161.471	165.076	168.603

Table 15: Skylake step 1

Variable parameter	0.5	2	5	10	20
R_i	99.66	102.08	111.51	121.41	131.67
R_v	91.83	109.95	123.53	133.85	144.17
R_i & R_v	89.76	110.29	123.92	134.24	144.56

Table 16: Skylake step 2

Instruction type	1.1	1.25	1.5	2
"cmd r,r 1"	100.004	100.01	100.019	100.039
"cmd x,x 1"	100.003	100.008	100.017	100.034
"cmd r,r,r 1"	100.201	100.585	101.111	102.021
"cmd x,x,x 1"	100.166	100.483	100.924	101.702

Table 17: Skylake step 3

Combination of instructions and additional registers	8	16	32	64
$r \times 2$ cmd3	126.767	130.454	134.014	137.514
$x \times 2$ cmd4	125.454	129.111	132.657	136.151
$r \times 5$ cmd3	144.858	148.603	152.19	155.703
$x \times 5$ cmd4	143.511	147.243	150.825	154.335
$r \times 10$ cmd3	158.589	162.354	165.95	169.467
$x \times 10$ cmd4	157.23	160.989	164.582	168.098

is expected that this value has the highest influence among all the presented parameters. Nevertheless, it is important to note that the really significant effect at the value of computer capacity (more than 10%) is obtained at 10-20 fold increase.

The second step of research shows the influence of instruction set on the performance of processors. All the obtained results show the growth of the computer capacity in the limits of 1-2% even with a double increase of the number of fastest instructions. Considering the complexity of designing new instructions, we can speak of the saturation of the instruction set with the existing types of instructions. It is also clearly seen that this exact way has been chosen by the manufacturers of the processors. Starting from Ivy Bridge processor a new type of fast instructions with three register operands was added. This statement is proved by the results from the third step of research where we simultaneously increase the number of registers and add instructions with three operands. As we can see in Table 2, 10 instructions with three register operands were added, and in Table 1 we see that the number of integer registers increased from 16 to 160 (10 times). So we can look at the results from Table 8 and find the result of 10 times increase in registers and add 16 instructions. The obtained value is 147.49% and if we compare the original values of computer capacity for processors Intel Core and Ivy Bridge from table 1 we can see the value 153.16% which is close to previously obtained.

5 Conclusions

After presenting all the obtained results we can make some conclusions. The last considered microarchitecture is Skylake (Kaby Lake and Coffee Lake are both used the same microarchitecture and are only the optimization of Skylake so there is no interest in evaluating them separatly) and its successor, which characteristics we try to predict, is not released yet. The investigations show that just a little set of Skylake parameters affects the computer capacity. In the way of achieving greater performance the manufacturer need to increase the number of internal registers (as we can see in table 15, vector registers have stronger effect on the computer capacity value then integer registers) and add new processor instructions (for example, some instructions for the specific tasks may be added). On the other hand, it is clearly seen that it makes no sense to change the values of RAM and cache-memory sizes or access times to improve the performance. This can be done for some technical needs but such a change will not affect the performance. We can also observe that for the last 5 years the number of registers has increased insignificantly, so we suppose that in new processor this tendency will continue and the number of registers will be increased insignificantly (for example, the number of vector registers will be increased from 168 to 180). There was shown in Table 17 that to achieve the significant growth of computer capacity the manufacturer need to design and add fast instructions with 4 register operands. The design of new instructions is very complicated task, but it is the most promising direction. In the context of presented work we do not consider the other parameters which influence is obvious and linear (number of cores, clock rate etc.). So, we can predict that in next processor microarchitecture the manufacturer will increase the number of internal registers and add some new fast instructions with three or more operands.

In this paper we successfully applied the computer capacity for investigation and prediction of processor evolution. Indeed, in our experiments we try to optimize the architecture of processor to achieve the maximal value of its performance and this work is very close to one which manufacturers do. But the use of computer capacity allows us to perform the optimization analytically, without building the working model of processor and running the benchmarks on it.

APPENDIX 1

In table 18 we present the values of Computer Capacity and PassMark benchmark for all processor microarchitectures investigated in paper. Here, the processor Ivy Bridge was presented twice just to show how Computer Capacity would behave with different numbers of cores (i5-3579 has 4 cores and E5-2660v2 has 10 cores). At figure 1 we can see the results of comparison these values. As the measurement units are different we took the value of the first processor in list (Intel Pentium M) as 1 and build the graph in relation to the values of this processor. For example, we divide the value of Computer Capacity for i5-6600K (1673398.77) by 102434.84 and get the result 16.336, and analogously we divide the value of PassMark benchmark (7884) by 464 and get 16.991.

Table 18: The values of Computer capacity and PassMark benchmark

Name	PassMark	Computer capacity, Mbit/s
Intel Pentium M	464	102434.84
Core 2 Duo T7300 (Intel Core)	1232	283593.72
Intel i5-3570 (Ivy Bridge)	6978	1610797.70
Intel Core i5-6600K (Skylake)	7884	1673398.77
Intel Xeon E5-2660v2 (Ivy Bridge)	13659	3179205.98
Intel Xeon E5-2640v3(Haswell)	14036	3151395.48

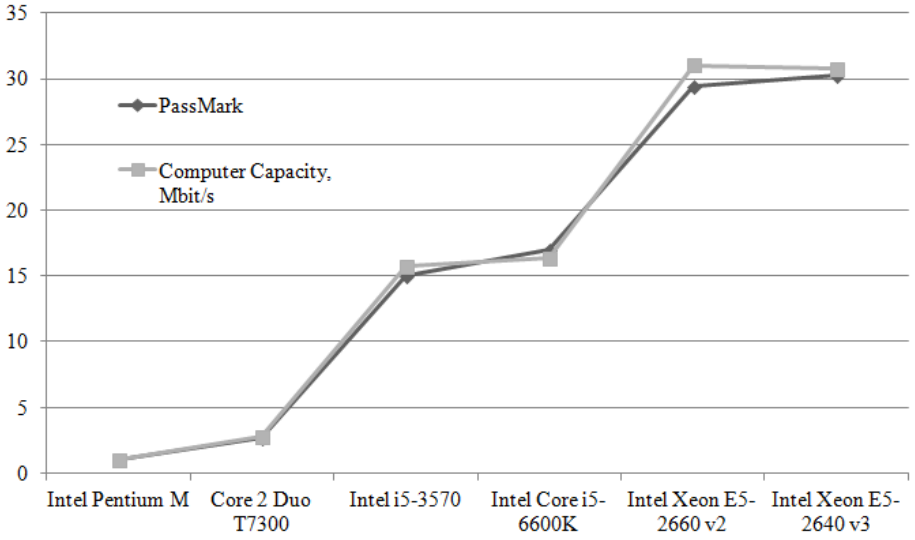

Figure 1: The results of processors comparison

References

[1] B. Ryabko, "An information-theoretic approach to estimate the capacity of processing units," *Performance Evaluation*, vol. 69, pp. 267–273, 2012.

[2] A. Rakitsky, B. Ryabko, and A. Fionov, "Evaluation of computer capacity for p5 intel processors," in *Problems of Redundancy in Information and Control Systems (RED), 2012 XIII International Symposium.* St. Petersburg: IEEE, 2012, pp. 70–73.

[3] B. Ryabko and A. Rakitskiy, "An analytic method for estimating the computation capacity of computing devices," *Journal of Circuits, Systems and Computers*, vol. 26, no. 05, p. 1750086, 2017. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0218126617500864

[4] R. P. Weicker, "A detailed look at some popular benchmarks," *Parallel Computing*, vol. 17, no. 10-11, pp. 1153–1172, 1991.

[5] D. J. Lilja, *Measuring computer performance: a practitioner's guide.* Cambridge university press, 2005.

[6] PassMark_Software, "Passmark performance test description," 2016. [Online]. Available: http://www.passmark.com/support/performancetest/interpreting_test_results.htm

[7] A. Rakitskiy and B. Ryabko, "An information-theoretic approach to performance evaluation of supercomputers," in *2016 XV International Symposium Problems of Redundancy in Information and Control Systems (REDUNDANCY)*, Sept 2016, pp. 125–128.

[8] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, 1948.

[9] A. Phansalkar, A. Joshi, L. Eeckhout, and L. K. John, "Measuring program similarity: Experiments with spec cpu benchmark suites," in *Performance Analysis of Systems and Software, 2005. ISPASS 2005. IEEE International Symposium on.* IEEE, 2005, pp. 10–20.

[10] A. Fog, "The microarchitecture of intel, amd and via cpus," 2016. [Online]. Available: www.agner.org/optimize/

[11] A. Rakitskiy, "Characteristic equations and evaluation programs," 2016. [Online]. Available: http://www.ict.nsc.ru/ru/structure/orgunits/lab-info-sys-security-page

APPENDIX 2

The characteristic equation (2) for Haswell processor:

$$\frac{524014680}{X^{1}}+\frac{2448358122}{X^{2}}+\frac{81258594}{X^{3}}+\frac{4863059}{X^{4}}+\frac{19968745476}{X^{5}}+\frac{7936385025}{X^{6}}+\frac{118510993664}{X^{7}}+\frac{4187392636}{X^{8}}+\frac{4198458}{X^{9}}+\frac{232644384}{X^{10}}+\frac{6978}{X^{11}}+\frac{14300162}{X^{12}}+\frac{3440640}{X^{13}}+\frac{19117056}{X^{14}}+\frac{9516033}{X^{15}}+\frac{32768}{X^{16}}+\frac{79874982059}{X^{17}}+\frac{31745540097}{X^{18}}+\frac{474046726144}{X^{19}}+\frac{16768421479}{X^{20}}+\frac{16678913}{X^{21}}+\frac{949315584}{X^{22}}+\frac{2778726}{X^{23}}+\frac{19267584}{X^{24}}+\frac{13762561}{X^{25}}+\frac{39911425}{X^{26}}+\frac{131072}{X^{27}}+\frac{2752513}{X^{28}}+\frac{11010049}{X^{31}}+\frac{3145728}{X^{288}}+\frac{9483265}{X^{34}}+\frac{11010048}{X^{35}}+\frac{168}{X^{36}}+\frac{5505024}{X^{38}}+\frac{11010048}{X^{40}}+\frac{1}{X^{41}}+\frac{6553}{X^{47}}+\frac{3932160}{X^{50}}+\frac{9584997826560}{X^{51}}+\frac{3809464811520}{X^{52}}+\frac{56885276835840}{X^{53}}+\frac{2009934594048}{X^{54}}+\frac{2001469442}{X^{55}}+\frac{111641886720}{X^{56}}+\frac{3145728}{X^{57}}+\frac{2312110081}{X^{58}}+\frac{1651533582}{X^{59}}+\frac{4624220160}{X^{60}}+\frac{15728640}{X^{61}}+\frac{1717986918}{X^{318}}+\frac{1321205760}{X^{65}}+\frac{1321205760}{X^{69}}+\frac{1}{X^{70}}+\frac{1}{X^{71}}+\frac{660602880}{X^{72}}+\frac{1321205760}{X^{74}}+\frac{1}{X^{78}}+\frac{2147483648}{X^{80}}+\frac{5234686813011968}{X^{81}}+\frac{2080475715731456}{X^{82}}+\frac{31066945855946752}{X^{83}}+\frac{1097692279629414}{X^{84}}+\frac{1093069176832}{X^{85}}+\frac{60971355734016}{X^{86}}+\frac{1717986918}{X^{87}}+\frac{1262720385024}{X^{88}}+\frac{901943132160}{X^{89}}+\frac{2525440770048}{X^{90}}+\frac{8589934592}{X^{91}}+\frac{3145728}{X^{93}}+\frac{8192}{X^{94}}+\frac{721554505728}{X^{95}}+\frac{721554505728}{X^{99}}+\frac{360777252864}{X^{102}}+\frac{721554505728}{X^{104}}+\frac{32768}{X^{106}}+\frac{2}{X^{110}}+\frac{128}{X^{120}}+\frac{1717986918}{X^{123}}+\frac{512}{X^{132}}+\frac{128}{X^{134}}+\frac{3932160}{X^{140}}+\frac{512}{X^{146}}+\frac{8192}{X^{151}}+\frac{32768}{X^{163}}+\frac{61440}{X^{166}}+\frac{2147483648}{X^{170}}+\frac{1}{X^{173}}+\frac{61440}{X^{180}}+\frac{33554432}{X^{196}}+\frac{3932160}{X^{197}}+\frac{33554432}{X^{210}}+\frac{1}{X^{224}}+\frac{2147483648}{X^{227}}+\frac{6553}{X^{242}}+\frac{26214}{X^{254}}=1$$