Journal of Circuits, Systems, and Computers
Vol. 27, No. 4 (2018) 1850062 (13 pages)

DOI: 10.1142/S0218126618500627

World Scientific
www.worldscientific.com

Theoretical Approach to Performance Evaluation of Supercomputers*

Boris Ryabko†

Laboratory of Data Protection, Institute of Computational Technologies of SB RAS, Lavrentev Prospect, 6, Novosibirsk 630090, Russia

Novosibirsk State University, 630090, Novosibirsk, 2, Pirogova Str., Russia
boris@ryabko.net

Anton Rakitskiy

Department of App. Math and Cybernetics, Siberian State University of Telecommunications and Informatics, Kirova st., 86, Novosibirsk 630102, Russia
rakitsky.anton@gmail.com

Received 20 February 2017
Accepted 14 July 2017
Published 4 August 2017

In this paper, we extend an information-theoretic approach of computer performance evaluation to supercomputers. This approach is based on the notion of computer Capacity which can be estimated relying solely on the description of computer architecture. We describe the method of calculating Computer Capacity for supercomputers including the influence of the architecture of communication network. The suggested approach is applied to estimate the performance of three of the top 10 supercomputers (according to TOP500 June-2016 list) which are based on Haswell processors. For greater objectivity of results, we compared them relatively to values of another supercomputer which is based an Ivy Bridge processors (this microarchitecture differs from Haswell). The obtained results are compared with values of TOP500 LINPACK benchmark and theoretical peak and we arrive at conclusions about the applicability of the presented theoretical approach (nonexperimental) for performance evaluation of real supercomputers. In particular, it means that the estimations of the computer capacity can be used at the design stage of the development of supercomputers.

Keywords: Computer capacity; performance; supercomputer; processor; information theory.

1. Introduction

The performance evaluation of computers and supercomputers is very important to developers and users. For now, there are many benchmarks which can give some

*This paper was recommended by Regional Editor Piero Malcovati.
†Corresponding author.

estimations of computer performance for some specific programs, but an application of benchmarks requires a working model of computer or supercomputer to estimate the performance, so it cannot be used at the design stage. Furthermore, there is a theoretical method of performance evaluation called theoretical peak, but it uses the IPC (instructions per cycle) value which is given by the manufacturer and also cannot be estimated theoretically (without a working model).

In this paper, we present the theoretical method (i.e., it works without any experiments over the working model of examined computer) of estimating the performance of supercomputers that could be a good alternative for existing methods. First, this method was presented in Ref. 1 where theoretical basis was given. Later, it was applied to real computers with Intel processors.[2] This method is closed in spirit to the main concept of Shannon's information theory[3] and it uses the characteristic Computer Capacity as a measure of performance. In this paper, we describe the application of the suggested method to performance evaluation of Haswell processors and three supercomputers based on them. We describe the application of this method to the performance estimation of three of the top 10 supercomputers according to TOP500 june 2016 list. Examined supercomputers are Trinity (7 place), Hazel Han (9 place) and Shaheen II (10 place),[4–6] each of them is Cray XC40 model. In order to show the effectiveness of the suggested method, we compare the obtained results of those three supercomputers relatively to the values of supercomputer iDataPlex DX360M4 (66 place).[7] The architecture of iDataPlex DX360M4 is based on processors with Ivy Bridge microarchitecture and differs from the architecture of previously described supercomputers which are based on Haswell processors and have another network structure and manufacturer.

1.1. *Computer capacity of computing systems*

The concept of Computer Capacity (CC) was suggested in Ref. 1. This notation gives a possibility of the performance estimation of computers which is based on the description of architecture, without any experiments over them. Full detailed theory was presented in Ref. 1 so, in the Appendix A we present just a brief description paying the main attention on ideas.

Supercomputers and any similar computer systems can be considered as a set of computing nodes connected via network. It is expected that all calculations are performed on this system in parallel, so each node (furthermore, each core on each node) can operate independently except the communication via shared memory. Computing node is a set of processors, RAM and network processor (NP) (also it is called network interface controller (NIC) placed on the same board. In Ref. 1 it was shown that if there is a computing system with $N \geq 1$ computing nodes $I_1, \ldots, I_N$, where any node can be run individually and independently on other nodes, then the capacity of this system is the sum of capacities of individual nodes $I_1, \ldots, I_N$. Thus

the Computer Capacity (CC) of computing system is

$$C_{\text{cs}} = \sum_{i=1}^{N} C_i \,, \tag{1}$$

where N is the number of nodes, C_i is CC of ith computing node and C_{cs} is CC of computing system. Here, CC of computing node is considered as

$$C_i = C_{\text{np}_i} + \sum_{j=1}^{N_i} C_{\text{core}_j} \,,$$

where N_i is the number of cores at ith computing node, C_{core_j} is the computer capacity of jth computing core and C_{np_i} is CC of the network processor placed at ith node.

2. Computer Capacity of Modern Processors

First, we consider some basic features of modern computers, which must be taken into account in the construction of Eq. (A.2).

2.1. *Cache memory*

A present-day processor includes the memory with fast access in the core. This memory (called cache memory) is used to reduce the average time to access data from the main memory. This mechanism works as follows: when CPU tries to access some data from the memory it looks in the cache memory first, if it was not found there, it then looks in the main memory. Requested data is stored in the cache memory in accordance with the caching algorithm (it may vary for different processors). We need to take this feature into account in a process of building Eq. (A.2). Let us consider a computer with instruction set I, main memory M and cache memory L. If there is an instruction $s \in I$ that has single operand which is a memory cell, $\tau(s)$ is the basic execution time of this instruction (excluding the memory access), $\tau(M)$ is the main memory access time and $\tau(L)$ is the cache memory access time then it would be presented in Eq. (A.2) as follows:

$$\cdots + \frac{|L|}{X^{\tau(s)+\tau(L)}} + \frac{|M|}{X^{\tau(s)+\tau(L)+\tau(M)}} + \cdots \,, \tag{2}$$

where $|L|$ is the size of cache memory and $|M|$ is the size of main memory related to instruction operand size. It is necessary to explain terms of the presented part of Eq. (2). The first term describes the situation when memory operand is located in cache, so execution time of instruction is the sum of the cache memory access time and the basic execution time. The number of such instruction types is equal to the size of cache memory related to the operand size, so after collecting similar terms we get single term with the numerator equal to $|L|$. The second term describes the situation when memory operand is located in main memory. Processor sends a

request to the cache memory first and after receiving the answer it sends the second request to the main memory, so the execution time of this type of instructions is the sum of the cache access time, the main memory access time and the basic execution time. The value of numerator is obtained the same way as for the first term. It is important to note that latest processors could have up to three levels of cache memory. But we can easily expand the described case for multiple cache levels.

2.2. *Pipeline*

The next feature which needs to be described is a pipeline. All present-day processors have a pipeline whose task is to reduce the execution time of instructions. This is achieved through the execution of instructions by parts. A pipeline is a sequence of data processing elements where the output of one element is the input of the next one. Each element works independently and, for example, when the instruction runs at the decoding stage then another one can run at the computing stage. The execution of instruction at the stage of pipeline takes much less time than complete execution, so for single instruction the execution time grows, but for a sequence of instructions (a processor task) the execution time significantly reduces. For our method, it is necessary to take into account this feature, when we consider the definition of execution time. In view of the described feature, we define the execution time of instruction as the maximum latency on the pipeline which is caused by this instruction. For example, let us consider a pipeline with four stages: the instruction decoder (stage 1), the register renaming stage (stage 2), the execution unit (stage 3) and the retirement stage (stage 4). And let there be two instructions MOV r1,r2 and ADD r1,r2 ('r1,r2' means that both operands are registers). Instruction MOV would execute for 1 clock cycle at the first stage, 1 at the second, 2 cycles at the third stage and 1 cycle at the fourth. Instruction ADD would have the following execution times: 1, 1, 3 and 1 cycles. Then, if there is the sequence of instructions, for example, MOV AX, CX (1); MOV BX, DX (2); ADD AX, BX (3); MOV CX, AX (4) then it executes in the following way (each step is one clock cycle):

(1) Instruction (1) is decoded at stage 1.
(2) Instruction (1) renames their registers at stage 2 and simultaneously instruction (2) is decoded.
(3) Instruction (1) starts to execute at stage 3, instruction (2) comes at stage 2 and instruction (3) comes at stage 1.
(4) Instruction (1) is still at stage 3 so instructions (2) and (3) are waiting.
(5) Instruction (1) copies the values of the temporary registers into the permanent registers at stage 4, (2) comes at stage 3, (3) comes at stage 2 and (4) comes at stage 1.
(6) Instruction (1) has already left the pipeline, instruction (2) is still at stage 3, so instructions (3) and (4) are waiting.

(7) Instruction (2) is still at stage 3 and insturctions (3) and (4) are still waiting.
(8) Instruction (2) comes at stage 4, (3) comes at stage 3 and (4) comes at stage 2.
(9) Instruction (3) is still at stage 3, (4) is waiting at stage 2.
(10) Instruction (3) comes at stage 4, (4) comes at stage 3.
(11) Instruction (4) is still at stage 3.
(12) Instruction (4) comes at stage 4.
(13) The pipeline is empty.

As we can see, when some sequence of instructions executes in the pipeline, the execution time of each instruction is equal to the number of clock cycles that runs between finishing this instruction and the previous. So, in this case, the execution time of instruction MOV equals 2 clock cycles and, analogously, the execution time of instruction ADD equals 3 clock cycles.

3. Haswell Processors

In this section we describe how to find the Computer Capacity of Haswell processors. First of all we need to describe inherent features of these processors. Haswell processors have the register file with 168 integer registers and 168 vector registers. It means, for example, that the instruction can use 168 integer registers as the operand instead of 16 registers (this is made automatically at the register renaming stage). The next important feature is the translation of instructions into microoperations (so-called μops). Due to this mechanism, each instruction is translated into several μops which are executed significantly faster and often could be executed independently. In most cases μops are executed for 1 clock cycle except for special cases (for example, loading data from memory). It is important to note that in most cases the execution time of instruction equals to the number of μops generated by this instruction. Also the number of dependency chains which can be executed at the pipeline simultaneously must be taken into account. In Haswell processors this number is 4. In our method it means that we have 4 independent pipelines, so after calculating the Computer Capacity of Haswell processor, we need to multiply the obtained value by 4.

Now we can describe the process of building the characteristic equation in Eq. (A.2) of Haswell processors. First, we need to make the file with the list of all instructions in the primary format. The primary format means that each instruction is presented by the instruction name, the list of types of operands and the basic execution time. The obtained file contains 681 entries. The next step is to transform the obtained file to the file with the list of instructions in the final form. This form is obtained by substituting real numbers in place of operands according to their types. For example, instruction "MOV r,r 1" is transformed to "MOV 28224 1". The number 28224 is the number of different instructions "MOV r,r 1" in I because each of "r" operands can take one of 168 different values, so the number of different

combinations is $168 \times 168 = 28224$. In order to transform the file with the list in the primary format automatically, we use the program which is based at the recursive descent parsing algorithm. This algorithm is used for the correct parsing of the complex instructions (for example, the instruction MOV r8/r16,r8/r16 is represented as the set of instructions: {MOV r8,r8; MOV r8,r16; MOV r16,r8; MOV r16,r16}). The file with the final form of instruction list can be found at Ref. 8. Also it is necessary to use the file with technical characteristics (sizes of all cache levels, the size of RAM, access times for all types of memory and the number of registers). When all these files are obtained, the program for calculating the Computer Capacity of the pipeline of Haswell core can be used, so $C_p(I) \approx \log_2 524014684.672 \approx 28.965$ bits per clock cycle. As we considered before, each core has 4 pipelines, so $C = C_p \times 4 \approx 28.965 \times 4 \approx 115.86$ bits per clock cycle. To estimate the Computer Capacity of real processors we need to multiply the obtained value by the value of the clock rate in order to transform the measurement units to bits per second. Next, we need to multiply the obtained result by the number of computing cores (these values may differ for different processors) as explained in Ref. 1 for multi-core processors. Equation (A.2) for Haswell processors after collecting similar terms is presented in Appendix.

4. Network Processor

The main task of the network processor is to form a packet with data and send it through the network to another network processor or to receive a packet from different network processor. So the set of instructions for this processor will consist of instructions which send some packet from one node to another (in our model each node contains a single network processor and each network processor corresponds to its node) or receive it. Taking into account the described features of network processors, we consider the characteristic equation of the network processor from the node with number k as follows:

$$C_{\text{np}} = \sum_{i=\text{minSize}}^{\text{maxSize}} \sum_{j=1, j\neq k}^{N} \frac{M_{i,k} + M_{i,j}}{X^{T_{i,j}}}, \tag{3}$$

where minSize, maxSize are the minimum and the maximum possible sizes of a packet in network, N is the number of nodes in supercomputer, $M_{i,j}$ is the number of different possible packets with size i which can be formed in node j, and T_j is the transmission time of the packet with size i between the node k and node j. Here, the numerator $M_{i,k} + M_{i,j}$ is built in such a way because it includes both sending instructions ($M_{i,k}$) and receiving instructions ($M_{i,j}$).

4.1. *Aries interconnect*

Three of examined supercomputers are Cray XC40 models and use Aries interconnect network. Let us consider the network structure: Cray XC40 systems composed

of blades with four nodes (each node includes two processors and memory). Each blade has a single Aries router (each NIC is connected to a single node). Such 16 blades (64 nodes) form the chassis and three chassis forms a cabinet (192 nodes). The considered network is constructed from two-cabinet electrical groups with 384 nodes per group. All connections inside the group are electrical and operate at 14 Gbps per lane. All groups are connected with 12X active optical cables operating at 12.5 Gbps. More details about technical characteristics are presented in Ref. 9. Next we consider the important characteristics that affect Eq. (3):

- NIC packetizes all requests and sends packets to the network. Each packet contains up to 64 bytes of data.
- Operating speed of an electrical link is 14 Gbps.
- Operating speed of an optical link is 12.5 Gbps.
- Communication between processors is performed through NICs (except the processors which are located at the same node).

Now our aim is to build Eq. (A.2) which is the same for each NIC in a supercomputer. The equation for NIC is as follows:

$$\sum_{i=1}^{64}\left(\frac{4\times M_i}{X^{T_{\mathrm{el}_i}\times 2}}+\frac{(N_{\mathrm{gr}}-3)\times M_i}{X^{T_{\mathrm{el}_i}\times 3}}+\frac{(N_{\mathrm{nd}}-N_{\mathrm{gr}}+1)\times M_i}{X^{T_{\mathrm{el}_i}\times 2+T_{\mathrm{op}_i}}}\right)=1\,, \tag{4}$$

where i is size of the packet in bytes, M_i is number of different packets with size i that can be formed in one node, T_{el_i} is the time of transmission of the i-byte packet through the electrical link, T_{op_i} is the time of transmission of the i-byte packet through the optical link, N_{gr} is number of nodes in group, N_{nd} is number of nodes in the supercomputer.

Let us take a closer look at Eq. (4). We need to consider three cases of the packet transmission through the network as follows:

(1) The data is transmitted between two nodes at the same blade. There are two transmissions, from the source node to the router and from the router to the destination node.
(2) The data is transmitted between two nodes from the same group, but with different blades. There are three transmissions here: from the source node to its router, from the source router to the router of the destination node (by electrical link) and from the router to the node.
(3) The data is transmitted between two nodes from different groups. There are also three transmissions: from the source node to its router, from the source router to the router of the destination node (by optical link) and from the router to the node.

In this way, case (1) is presented in Eq. (4) by the term $\frac{4\times M_i}{X^{T_{\mathrm{el}_i}\times 2}}$ (here 4 is the number of nodes at the blade), case (2) by the term $\frac{(N_{\mathrm{gr}}-3)\times M_i}{X^{T_{\mathrm{el}_i}\times 3}}$ and case (3) by the term $\frac{(N_{\mathrm{nd}}-N_{\mathrm{gr}}+1)\times M_i}{X^{T_{\mathrm{el}_i}\times 2+T_{\mathrm{op}_i}}}$.

4.2. *InfiniBand*

Supercomputer iDataPlex DX360M4 uses InfiniBand FDR[10] as interconnection. The method of calculation Computer Capacity of NP for InfiniBand is similar to the one previously discussed. Here some technical characteristics which are important to Eq. (3) are as follows:

- The minimum possible size of the transport packet in InfiniBand is 32 bytes (according to Ref. 10).
- The maximum possible size of the transmitted data is 4096 bytes.
- Theoretical effective throughput for 1x link is 13.64 Gbit/s.

CC of network processor is estimated for InfiniBand FDR interconnect of the examined supercomputer iDataPlex is ≈ 9.24 Gbits per second.

4.3. *iDataPlex DX360M4*

Now we can present the results of the supercomputer iDataPlex DX360M4. This supercomputer is constructed with processors Intel Xeon E5-2680 v2, which have Ivy Bridge microarchitecture. Computer Capacity for Ivy Bridge is estimated closely to Haswell and its value is $C(I) \approx 108.582$ bits per clock cycle. We estimate the upper bound of Computer Capacity, so the maximum value of the clock rate is used for processors. For the single core of E5-2680 v2 processor this value is $C(I) \approx 390.9$ Gbits per second. Each node of this supercomputer contains two processors (each processor is 10-core). The number of cores for iDataPlex is 65320 so the number of nodes is 3266. With all presented values we can calculate the total Computer Capacity of supercomputer which is set by Eq. (1) and this value is ≈ 25563.52 Tbits per second.

5. Analysis of Results

In order to show the effectiveness of Computer Capacity, we examined and compared four supercomputers: iDataPlex, Shaheen II, Hazel Han and Trinity. Let us emphasize that we consider the supercomputers of two different types: iDataPlex corresponds to the first type (InfiniBand FDR interconnect, Ivy Bridge processors) and the other supercomputers correspond to the second one (Aries interconnect, Haswell

Table 1. Characteristics of supercomputers.

	Trinity	Hazel Han	Shaheen II	iDataPlex DX360M4
CC of NIC, Gbit/s	1065.46	1064.56	1063.54	9.24
CC of core, Gbit/s	532.96	579.3	532.96	390.9
CC of supercomputer, Gbit/s	170473598.5	115431469.22	111317928.7	25563517.67
Number of cores	301056	185088	196608	65320
Clock rate of core, MHz	3600.00	3300.00	3600.00	3600.00
LINPACK, TFlops/s	8100.90	5640.17	5536.99	1283.31
Theoretical peak, TFlops/s	11078.90	7403.52	7235.17	1463.17

processors). All characteristics and results of examined supercomputer are presented in Table 1. Following are some clarifications for Table 1:

- CC of NIC is Computer Capacity of a single network processor.
- CC of core is Computer Capacity of a single computing core.
- CC of supercomputer is the total value of supercomputer's Computer Capacity set by Eq. (1).
- LINPACK is the name of benchmark presented in TOP500 list as Rmax.
- Theoretical peak is the name of characteristic called Rpeak in TOP500 list.

Measurement units of examined characteristics are different so it is impossible to compare Computer Capacity with LINPACK and Theoretical peak directly. First, we sort the supercomputers in ascending order of Computer Capacity. Next, we divide each value by the corresponding value of iDataPlex supercomputer. In this case we get a new Table 2, where the values of characteristics corresponding to iDataPlex supercomputer are equal to 1. The obtained values in Table 2 are relative and have no measurement units so we can compare them with each other directly. In order to present the results more visually, we build the graph presented in Fig. 1. This graph shows us that the obtained results of the suggested method are closer to the experimental results of LINPACK than the results of Theoretical peak. It is important to note that the value of benchmark LINPACK is the major characteristic in TOP500 rating and the places are assigned to supercomputers according to this value (Theoretical peak is used only in the case of equality). Let us take a closer look at Fig. 1. For example, if we take a look at the position corresponding to Shaheen II supercomputer, we can see that the points representing the values of LINPACK and Computer Capacity are almost fused in one point, unlike the value of the theoretical

Table 2. Comparison results of supercomputers.

	iDataPlex DX360M4	Shaheen II	Hazel Han	Trinity
Computer Capacity	1.00	4.35	4.52	6.67
LINPACK	1.00	4.31	4.40	6.31
Theoretical peak	1.00	4.94	5.06	7.57

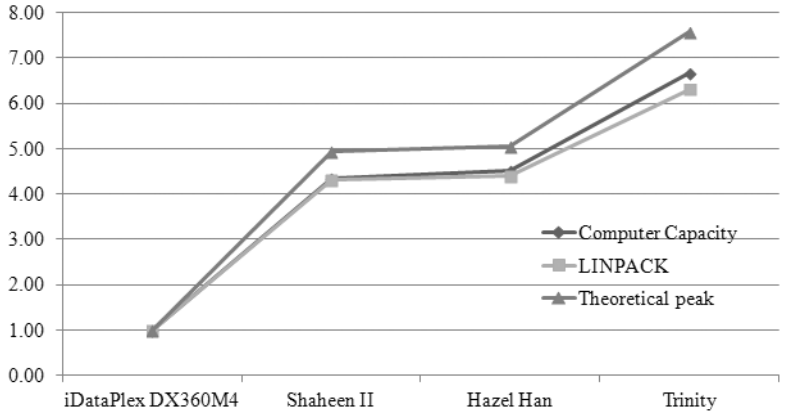

Fig. 1. The graph of Table 2.

peak which is placed far enough from them. We can watch the same situation in all other positions (except the position of iDataPlex). This allows us to talk about high affinity of the obtained values for considered characteristics. The obtained results show that the method of theoretical estimation of Computer Capacity (without any experimental researches over the model of investigated computer) is accurate and can be used for performance evaluation of supercomputers. Moreover, since our method is entirely theoretical, it can be used at the design stage of new supercomputers.

Appendix A

Let us consider a computer with instruction set I and memory M. Each instruction $s \in I$ is described by instruction name and values of its operand. It means that the same instructions with different values of operands are included in I independently. For example, we consider that MOV R0 R1 and MOV R0 R2 (where R0, R1 and R2 are names of registers) are different instructions and both included in I. We denote $\tau(s), s \in I$ as the execution time of instruction s and $S = s_1, s_2, \ldots, s_n, s_i \in I$ as the computer task. Then the execution time of task S is considered as $\tau(S) = \sum_{i=1}^{n} \tau(s_i)$. We also suppose that all execution times $\tau(s)$ are integers and the greatest common divisor of $\tau(s), s \in I$ equals 1 (this assumption is valid for most of processors because there are instructions with execution time equal to one time unit, i.e., $\tau(s) = 1$). We consider the number of different problems with execution times equal T as $\nu(T)$ and note that it is equal to the size of the set of all sequences of instructions with execution times equal T, i.e., $\nu(T) = N(T)$, where $N(T) = |\{S = s_1, s_2, \ldots, s_n : s_i \in I, \tau(S) = T\}|$. Let there be a processor which can execute N different sequences of instructions during 1 min. We can say that this processor can execute N^2

sequences of instructions during 2 min because if S_1 and S_2 are 1-minute sequences, the combined sequence S_1S_2 is a 2-minute one (we did not take into account a few extra 2-minute sequences with instruction which starts at the end of first minute and finishes at the beginning of the second one). Analogously, $\approx N^k$ sequences can be executed during k minutes. So the number of possible sequences grows exponentially as a function of the time T ($N(T) \approx 2^{\mathrm{CT}}$), thus $\log N(T)/T$ (or the limit of this value) is adequate measure of processor capacity and CC defines as follows:

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

It is worth noting that this situation is typical for Information Theory, where, for example, the capacity of a lossless channel is defined by the rate of asymptotic growth of the number of allowed sequences of basic symbols (letters), whose length can be different.

The important question is how to calculate (or at least estimate) $C(I)$ in (A.1). Obviously, the direct calculation of the limit is impossible, but in combinatorial analysis there exists a method of calculation the capacity $C(I)$. This method was used by Shannon[3] when he estimated a channel capacity. In this case we consider instruction set I as an alphabet and assume that all words (sequences of symbols) from that alphabet can be executed. This assumption allows to estimate the upper-bound of the processor capacity, because for any processor the set of admissible sequence of instructions is a subset of all possible sequences. Considering the above, $C(I)$ is equal to the logarithm of the largest real solution X_0 of the following characteristic equation:

$$X^{-\tau(s_1)} + X^{-\tau(s_2)} + \cdots + X^{-\tau(s_n)} = 1\,. \tag{A.2}$$

It is shown in Ref. 1 that CC of multi-core processors is defined as a sum of CCs of the cores.

Appendix B

$$\begin{aligned}
&\frac{524014680}{X^1} + \frac{2448358122}{X^2} + \frac{81258594}{X^3} + \frac{4863059}{X^4} + \frac{19968745476}{X^5} + \frac{7936385025}{X^6}\\
&+ \frac{118510993664}{X^7} + \frac{4187392636}{X^8} + \frac{4198458}{X^9} + \frac{232644384}{X^{10}} + \frac{6978}{X^{11}} + \frac{14300162}{X^{12}}\\
&+ \frac{3440640}{X^{13}} + \frac{19117056}{X^{14}} + \frac{9516033}{X^{15}} + \frac{32768}{X^{16}} + \frac{79874982059}{X^{17}} + \frac{31745540097}{X^{18}}\\
&+ \frac{474046726144}{X^{19}} + \frac{16768421479}{X^{20}} + \frac{16678913}{X^{21}} + \frac{949315584}{X^{22}} + \frac{2778726}{X^{23}}\\
&+ \frac{19267584}{X^{24}} + \frac{13762561}{X^{25}} + \frac{39911425}{X^{26}} + \frac{131072}{X^{27}} + \frac{2752513}{X^{28}} + \frac{11010049}{X^{31}}
\end{aligned}$$

$$+\frac{9483265}{X^{34}}+\frac{11010048}{X^{35}}+\frac{168}{X^{36}}+\frac{5505024}{X^{38}}+\frac{11010048}{X^{40}}+\frac{1}{X^{41}}+\frac{6553}{X^{47}}$$
$$+\frac{3932160}{X^{50}}+\frac{9584997826560}{X^{51}}+\frac{3809464811520}{X^{52}}+\frac{56885276835840}{X^{53}}$$
$$+\frac{2009934594048}{X^{54}}+\frac{2001469442}{X^{55}}+\frac{111641886720}{X^{56}}+\frac{3145728}{X^{57}}+\frac{2312110081}{X^{58}}$$
$$+\frac{1651533582}{X^{59}}+\frac{4624220160}{X^{60}}+\frac{15728640}{X^{61}}+\frac{1321205760}{X^{65}}+\frac{1321205760}{X^{69}}+\frac{1}{X^{70}}$$
$$+\frac{1}{X^{71}}+\frac{660602880}{X^{72}}+\frac{1321205760}{X^{74}}+\frac{1}{X^{78}}+\frac{2147483648}{X^{80}}$$
$$+\frac{5234686813011968}{X^{81}}+\frac{2080475715731456}{X^{82}}+\frac{31066945855946752}{X^{83}}$$
$$+\frac{1097692279629414}{X^{84}}+\frac{1093069176832}{X^{85}}+\frac{60971355734016}{X^{86}}+\frac{1717986918}{X^{87}}$$
$$+\frac{1262720385024}{X^{88}}+\frac{901943132160}{X^{89}}+\frac{2525440770048}{X^{90}}+\frac{8589934592}{X^{91}}+\frac{3145728}{X^{93}}$$
$$+\frac{8192}{X^{94}}+\frac{721554505728}{X^{95}}+\frac{721554505728}{X^{99}}+\frac{360777252864}{X^{102}}+\frac{721554505728}{X^{104}}$$
$$+\frac{32768}{X^{106}}+\frac{2}{X^{110}}+\frac{128}{X^{120}}+\frac{1717986918}{X^{123}}+\frac{512}{X^{132}}+\frac{128}{X^{134}}+\frac{3932160}{X^{140}}+\frac{512}{X^{146}}$$
$$+\frac{8192}{X^{151}}+\frac{32768}{X^{163}}+\frac{61440}{X^{166}}+\frac{2147483648}{X^{170}}+\frac{1}{X^{173}}+\frac{61440}{X^{180}}+\frac{33554432}{X^{196}}$$
$$+\frac{3932160}{X^{197}}+\frac{33554432}{X^{210}}+\frac{1}{X^{224}}+\frac{2147483648}{X^{227}}$$
$$+\frac{6553}{X^{242}}+\frac{26214}{X^{254}}+\frac{3145728}{X^{288}}+\frac{1717986918}{X^{318}}=1$$

References

1. B. Ryabko, An information-theoretic approach to estimate the capacity of processing units, *Perform. Eval.* **69** (2012) 267–273.
2. B. Ryabko and A. Rakitskiy, An analytic method for estimating the computation capacity of computing devices, *J. Circuits Syst. Comput.* **26**(5) (2017) 1750086.
3. C. E. Shannon, A mathematical theory of communication, *Bell Syst. Tech. J.* **27** (1948) 379–423.
4. TOP500 official website, description of Trinity supercomputer, http://top500.org/system/178610.
5. TOP500 official website, description of Hazel Hen supercomputer, http://top500.org/system/178446.
6. TOP500 official website, description of Shaheen II supercomputer, http://top500.org/system/178515.
7. TOP500 official website, description of iDataPlex DX360M4 supercomputer, http://top500.org/system/178197.

8. Files with equations for processors, http://www.ict.nsc.ru/ru/structure/orgunits/lab-info-sys-security-page.
9. B. Alverson, E. Froese, L. Kaplan and D. Roweth, Cray XC Series Network, Cray official website, www.cray.com/sites/default/files/resources/CrayXCNetwork.pdf.
10. Tom Shanley, Joe Winkles, MindShare Inc., InfiniBand Network Architecture, 1208 p (Addison-Wesley Professional, 2003).