12. Mersenne Number Finding and Collatz Hypothesis Verification in the Comcute Grid System

Jerzy Balicki, Waldemar Korłub, Jan Masiejczyk*, Jacek Paluszak

Gdansk University of Technology

Faculty of Electronics, Telecommunications and Informatics

Computer System Architecture Department

e-mail: balicki@eti.pg.gda.pl, stawrul@gmail.com, jacekpaluszak@gmail.com

**The Naval University of Gdynia*

e-mail: j.masiejczyk@amw.gdynia.pl

Abstract

In this chapter, some mathematic applications have been described to test scalability of the Comcute grid system. Especially, a verification of the Collatz hypothesis and finding Mersenne numbers were applied to prove the scalability and high performance of this grid system. Results were compared with outcomes obtained by the other grid systems.

Keywords: *grid computing, Collatz hypothesis, volunteer computing, Mersenne numbers, system scalability.*

12.1. Introduction

In the Comcute system, an application for the Collatz hypothesis and another for finding Mersenne numbers were applied to prove the scalability and high performance. The Collatz problem was not solved up till now and is continuously calculated in BOINC grid system due to the large weight of a potential scientific discovery involving the natural number sequence that is not finished by three 4, 2, 1. Therefore, in one phase of the project involving the implementation and transferring some application from PC to the grid environment, we attempted to verify the effectiveness of hypothesis $3n+1$ by the Comcute application.

In 1952, Raphael Robinson applied a computer to discover five Mersenne primes that were the largest prime numbers known at the time. Then the Great Internet Mersenne Prime Search started as a distributed computing project on the Internet based on the grid architecture system. Nowadays, 47 Mersenne primes are known. Since 1997, all newly-found Mersenne primes have been discovered by the GIMPS.

12.2. Mersenne numbers

Small Mersenne primes like 3, 5, 31 and 127 were discovered by some ancient Greek mathematicians and Euclid, who proved a relation with among the perfect numbers. The next Mersenne prime 8191 was calculated in 1456 by a trial method that can be treated as the first integer factorization algorithm. Some asteroids discovered and named during the nineteenth century were related to the Mersenne primes, eg. *3 Juno, 7 Iris, 31 Euphrosyne* and *127 Johanna*. Another asteroid with minor planet number 8191 is named *8191 Mersenne* [10].

To prove that an integer number x is not prime, it can be divided by any integer greater than one but less than x. An effort can be reduced by selecting prime numbers as some candidate factors, only. Furthermore, the tested factors don't need to go further than $\sqrt{x}$ because, if x is divisible by a, then $x = a \times b$. If b is smaller than a, x would have previously been detected as being divisible by b or b is a prime factor [13].

The next two numbers 131071 and 524287 were calculated by Pietro Cataldi by a trial method. Similarly, 2147483647 was proved as Mersenne number by Leonhard Euler in 1972.

Marin Mersenne, a French mathematician who studied these numbers in the 17th century, prepared an incorrect list of Mersenne primes with exponents up to 257. However, he developed a theory of some positive integers (called Mersenne numbers) that are one less than a power of two, as below:

$$M_p = 2^p - 1. \tag{1}$$

A Mersenne prime is a Mersenne number that is prime. M_p is composite if p is composite. Moreover, if M_p is prime then p is prime. In computer science, unsigned p-bit integers can be used to express numbers up to M_p. Signed p+1-bit integers can implement values between $-M_p$ and M_p. In the problem of Hanoi Towers, solving a puzzle with a p-disc tower requires M_p iterations. In some NP-hard problems, the complexity of them is $O(M_p)$.

The Lucas test is a test for Mersenne numbers to be prime. This test was originally developed by French mathematician Édouard Lucas in 1856, what permitted him to find the new M_{127}. What is more, smaller numbers M_{61}, M_{89}, M_{107} were discovered by using Lucas test till 1911. The Lucas sequences $S_i(n,m)$ and $Z_i(n,m)$ are some integer sequences that satisfy the recurrence relation:

$$x_i = nx_{i-1} - mx_{i-2}, \qquad i=1,2,\ldots \tag{2}$$

where n, m are fixed integers,

Some examples of Lucas sequences are the Mersenne numbers, Fibonacci numbers, Pell numbers, Lucas numbers, Jacobsthal numbers, and a superset of Fermat numbers [12].

Derrick Lehmer in the 1930s improved the Lucas test and the Lucas–Lehmer test works as follows. Let p be an odd prime that can be tested by a trial division algorithm for establishing if it is prime. Define a sequence $\{x_i\}$ for all $i \geq 0$ by the following formula [13]:

$$x = \begin{cases} 4 & \text{if } i = 0 \\ x_{i-1}^2 & \text{otherwise} \end{cases} \tag{3}$$

The first few terms of this sequence are 4, 14, 194, 37634, 2005956546822746114,.... Then M_p is prime if:

repeat p − 2 times:
   x = ((x × x) − 2) mod M$_p$
if x = 0 return PRIME else return COMPOSITE

The number x_{p-2} *mod* M_p is called the Lucas–Lehmer residue of p. By performing the `mod M` at each iteration, we ensure that all intermediate results are at most p bits (otherwise the number of bits would double each iteration).

In 1952, Raphael Robinson used a computer SWAC to discover five Mersenne primes that were the largest prime numbers known at the time, M_{521}, M_{607}, M_{1279}, M_{2203}, M_{2281} with 157, 183, 386, 664, and 687 digits [14]. 35 Mersenne numbers have been discovered till 1996 by computers. Then the *Great Internet Mersenne Prime Search* (GIMPS) started as a distributed computing project on the Internet based on the grid architecture systems. Nowadays, 47 Mersenne primes are known. The largest known prime number ($2^{43,112,609}$ – 1) is a Mersenne prime. Since 1997, all newly-found Mersenne primes have been discovered by the GIMPS.

The Comcute grid system has processed the Lucas-Lehmer procedure to test positive integers as Mersenne primes for three months. We exam smaller integers and then we try to extend the upper bounder beyond the 47$^{\text{th}}$ Mersenne prime.

12.3. The Collatz problem in volunteer computing systems

The Collatz problem is calculated in BOINC system due to the large weight of a potential scientific discovery involving the natural number sequence that is not finished by three 4, 2, 1 [2]. On the other hand, it is important to test the numbers greater than 10^{18} to overthrow the hypothesis, but the probability of Collatz is very low.

Therefore, in one phase of the project involving the implementation and transferring some application from PC to the grid environment, we attempted to verify the effectiveness of hypothesis $3n+1$ by the Comcute application.

Collatz problem is still unresolved problem from a mathematical number theory. The German mathematician Lothar Collatz hypothesized in 1937 that a string of numbers assigned according to the following formula [4]:

$$c_{n+1} = \begin{cases} \frac{1}{2}c_n, & \text{if } c_n \text{ is even,} \\ 3c_n + 1, & \text{if } c_n \text{ is odd.} \end{cases} \tag{4}$$

tends to 1 for each initial value c_n.

This issue was also considered by the Polish mathematician Stanislaw Ulam (hence is also known as a problem of Ulam or problem $3n+1$) [5, 6, 7].

Analyzing the above relationship, it can be provisionally concluded that the sequence of natural numbers should strive to infinity, because it is increased three times for every odd number, and it is reduced only twice for even numbers. In addition, each odd number is even increased by 1 after multiplying by 3 [8, 9].

On the other hand, Collatz claimed that regardless of what number we choose the initial cycle finally achieve 4, 2, 1 [1, 18]. The problem remains open to this day. Paul Erdős said that mathematics is not ready for such problems [10]. The contrast between simplicity and complexity of the formulation of the problem itself is fascinating [11].

This hypothesis has been empirically verified for numbers up to 10^{18} [12.17]. So far we have not found the initial number, which does not lead to the aforementioned series of 4, 2, 1 In the system Comcute, we took the natural numbers greater than 10^{18} to empirical verification of this hypothesis. Although the experiments carried only ones confirmed the convergence of the Collatz sequence of numbers in the range to 10^{30}, it is also possible to test the scalability of grid computing system, which was the practical benefit of this test experiment [20].

12.4. Parallelization in a laboratory application CJEE

The Collatz hypothesis states that there is no matter from what number the calculation process is started, because we finally get to number 1 [13, 16]. During a verification whether any number satisfies the hypothesis Collatz that takes place in the system Comcute JEE, some web users on their computers test different numbers, independently.

In practice, there were situations where hundreds of Internet users launched a test program at the same time and number of processes reached several thousand. With the independence of calculation the various execution units must not expect to complete the calculations on other nodes. There is therefore no time dependence between operations carried out by the various nodes of a grid system. So if a user’s computer starts calculating the value of 97, it will appoint 118 consecutive sequence of numbers, as below.

```
292 146 73 220 110 55 166 83 250 125 376 188 94 47 142 71 214 107 322 161 484 2
2 121 364 182 91 274 137 412 206 103 310 155 466 233 700 350 175 526 263 790 39
 1186 593 1780 890 445 1336 668 334 167 502 251 754 377 1132 566 283 850 425 12
6 638 319 958 479 1438 719 2158 1079 3238 1619 4858 2429 7288 3644 1822 911 273
 1367 4102 2051 6154 3077 9232 4616 2308 1154 577 1732 866 433 1300 650 325 976
488 244 122 61 184 92 46 23 70 35 106 53 160 80 40 20 10 5 16 8 4 2 1
```

Then, a web program returns a value of 0 (no success) to the data and code distributing node *S*. As the interaction between the user's computer and the server *S* do not too often, the computer receives a web data packet of thousands of natural numbers defined by the pair <*a,b*>. In this way, the user's computer processes the data during several seconds. Of course, in the case of breaking the Collatz hypothesis a web browser returns the number.

In the case of increasing the initial value to 1097, 137 transformations are needed as follow:

```
3292 1646 823 2470 1235 3706 1853 5560 2780 1390 695 2086 1043 3130 1565 4696 2
48 1174 587 1762 881 2644 1322 661 1984 992 496 248 124 62 31 94 47 142 71 214 1
07 322 161 484 242 121 364 182 91 274 137 412 206 103 310 155 466 233 700 350 1
5 526 263 790 395 1186 593 1780 890 445 1336 668 334 167 502 251 754 377 1132 5
6 283 850 425 1276 638 319 958 479 1438 719 2158 1079 3238 1619 4858 2429 7288 
644 1822 911 2734 1367 4102 2051 6154 3077 9232 4616 2308 1154 577 1732 866 433
1300 650 325 976 488 244 122 61 184 92 46 23 70 35 106 53 160 80 40 20 10 5 16 
 4 2 1
```

Thus, if an Internet user receives a packet of data to test the hypothesis Collatz range from 2 to 10 000, it is estimated that not more than 2 million arithmetic operations per second are needed.

It should be emphasized that the number of operations does not grow as significantly as the numbers. For example, 303 operations are required to the convergence of the sequence for 10^{18}, as below:

```
5e+017 2.5e+017 1.25e+017 6.25e+016 3.125e+016 1.5625e+016 7.8125e+015 3.90625e+
015 1.95313e+015 9.76563e+014 4.88281e+014 2.44141e+014 1.2207e+014 6.10352e+013
 3.05176e+013 1.52588e+013 7.62939e+012 3.8147e+012 1.14441e+013 5.72205e+012 2.
86102e+012 8.58307e+012 4.29153e+012 1.28746e+013 6.4373e+012 3.21865e+012 1.609
33e+012 8.04663e+011 4.02331e+011 2.01166e+011 6.03497e+011 3.01749e+011 1.50874
e+011 7.54371e+010 3.77186e+010 1.88593e+010 9.42964e+009 4.71482e+009 2.35741e+
009 7.07223e+009 3.53612e+009 1.06083e+010 5.30417e+009 2.65209e+009 7.95626e+00
9 3.97813e+009 1.19344e+010 5.96719e+009 1.79016e+010 8.95079e+009 4.4754e+009 2
.2377e+009 1.11885e+009 5.59425e+008 1.67827e+009 8.39137e+008 2.51741e+009 1.25
871e+009 6.29353e+008 1.88806e+009 9.44029e+008 2.83209e+009 1.41604e+009 7.0802
2e+008 2.12406e+009 1.06203e+009 3.1861e+009 1.59305e+009 7.96524e+008 2.38957e+
009 1.19479e+009 5.97393e+008 2.98697e+008 8.9609e+008 4.48045e+008 1.34413e+009
 6.72067e+008 2.0162e+009 1.0081e+009 5.04051e+008 2.52025e+008 7.56076e+008 3.7
8038e+008 1.13411e+009 5.67057e+008 2.83528e+008 1.41764e+008 7.08821e+007 2.126
46e+008 1.06323e+008 3.1897e+008 1.59485e+008 4.78454e+008 2.39227e+008 7.17681e
+008 3.58841e+008 1.07652e+009 5.38261e+008 1.61478e+009 8.07392e+008 4.03696e+0
08 2.01848e+008 6.05544e+008 3.02772e+008 1.51386e+008 4.54158e+008 2.27079e+008
 1.13539e+008 5.67697e+007 1.70309e+008 8.51546e+007 2.55464e+008 1.27732e+008 3
.83196e+008 1.91598e+008 9.57989e+007 4.78995e+007 1.43698e+008 7.18492e+007 2.1
5548e+008 1.07774e+008 5.38869e+007 1.61661e+008 8.08303e+007 2.42491e+008 1.212
45e+008 6.06227e+007 3.03114e+007 1.51557e+007 7.57784e+006 2.27335e+007 1.13668
e+007 3.41003e+007 1.70501e+007 8.52507e+006 4.26254e+006 1.27876e+007 6.39381e+
006 3.1969e+006 9.59071e+006 4.79536e+006 1.43861e+007 7.19303e+006 2.15791e+007
 1.07896e+007 5.39478e+006 1.61843e+007 8.09216e+006 2.42765e+007 1.21382e+007 3
.64147e+007 1.82074e+007 9.10368e+006 4.55184e+006 2.27592e+006 6.82776e+006 3.4
1388e+006 1.70694e+006 5.12082e+006 2.56041e+006 1.28021e+006 640103 1.92031e+00
6 960155 2.88047e+006 1.44023e+006 4.3207e+006 2.16035e+006 1.08018e+006 3.24053
e+006 1.62026e+006 4.86079e+006 2.4304e+006 7.29119e+006 3.64559e+006 1.09368e+0
07 5.46839e+006 2.7342e+006 8.20259e+006 4.10129e+006 1.23039e+007 6.15194e+006
3.07597e+006 1.53799e+006 4.61396e+006 2.30698e+006 1.15349e+006 3.46047e+006 1.
73023e+006 865117 2.59535e+006 1.29768e+006 648838 324419 973258 486629 1.45989e
+006 729944 364972 182486 91243 273730 136865 410596 205298 102649 307948 153974
 76987 230962 115481 346444 173222 86611 259834 129917 389752 194876 97438 48719
 146158 73079 219238 109619 328858 164429 493288 246644 123322 61661 184984 9249
2 46246 23123 69370 34685 104056 52028 26014 13007 39022 19511 58534 29267 87802
 43901 131704 65852 32926 16463 49390 24695 74086 37043 111130 55565 166696 8334
8 41674 20837 62512 31256 15628 7814 3907 11722 5861 17584 8792 4396 2198 1099 3
298 1649 4948 2474 1237 3712 1856 928 464 232 116 58 29 88 44 22 11 34 17 52 26
13 40 20 10 5 16 8 4 2 1
```

However, the increasing the value to 10^{36} causes that 515 operations are required, as follow:

```
5e+031 2.5e+031 1.25e+031 6.25e+030 3.125e+030 1.5625e+030 7.8125e+029 3.90625e+
029 1.95313e+029 9.76563e+028 4.88281e+028 2.44141e+028 1.2207e+028 6.10352e+027
 3.05176e+027 1.52588e+027 7.62939e+026 3.8147e+026 1.90735e+026 9.53674e+025 4.
76837e+025 2.38419e+025 1.19209e+025 5.96046e+024 2.98023e+024 1.49012e+024 7.45
058e+023 3.72529e+023 1.86265e+023 9.31323e+022 4.65661e+022 2.32831e+022 1.1641
5e+022 5.82077e+021 2.91038e+021 1.45519e+021 7.27596e+020 3.63798e+020 1.81899e
+020 9.09495e+019 4.54747e+019 2.27374e+019 1.13687e+019 5.68434e+018 2.84217e+0
18 1.42109e+018 7.10543e+017 3.55271e+017 1.77636e+017 8.88178e+016 4.44089e+016
 2.22045e+016 1.11022e+016 5.55112e+015 1.66533e+016 8.32667e+015 2.498e+016 1.2
49e+016 6.245e+015 3.1225e+015 9.36751e+015 4.68375e+015 2.34188e+015 1.17094e+0
15 3.51282e+015 1.75641e+015 8.78204e+014 2.63461e+015 1.31731e+015 3.95192e+015
 1.97596e+015 9.87979e+014 2.96394e+015 1.48197e+015 7.40984e+014 2.22295e+015 1
.11148e+015 3.33443e+015 1.66721e+015 5.00164e+015 2.50082e+015 7.50247e+015 3.7
5123e+015 1.87562e+015 9.37808e+014 4.68904e+014 1.40671e+015 7.03356e+014 3.516
78e+014 1.75839e+014 8.79195e+013 2.63759e+014 1.31879e+014 3.95638e+014 1.97819
e+014 5.93457e+014 2.96728e+014 1.48364e+014 7.41821e+013 2.22546e+014 1.11273e+
014 3.33819e+014 1.6691e+014 5.00729e+014 2.50365e+014 1.25182e+014 3.75547e+014
 1.87773e+014 9.38867e+013 2.8166e+014 1.4083e+014 7.04151e+013 3.52075e+013 1.7
6038e+013 8.80188e+012 2.64056e+013 1.32028e+013 3.96085e+013 1.98042e+013 9.902
12e+012 2.97063e+013 1.48532e+013 7.42659e+012 3.71329e+012 1.11399e+013 5.56994
e+012 2.78497e+012 1.39249e+012 6.96243e+011 2.08873e+012 1.04436e+012 5.22182e+
011 1.56655e+012 7.83273e+011 3.91636e+011 1.95818e+011 9.79091e+010 4.89546e+01
0 2.44773e+010 7.34318e+010 3.67159e+010 1.10148e+011 5.50739e+010 1.65222e+011
8.26108e+010 2.47832e+011 1.23916e+011 3.71749e+011 1.85874e+011 5.57623e+011 2.
78811e+011 8.36434e+011 4.18217e+011 2.09109e+011 1.04554e+011 5.22772e+010 1.56
831e+011 7.84157e+010 3.92079e+010 1.17624e+011 5.88118e+010 2.94059e+010 8.8217
7e+010 4.41088e+010 1.32327e+011 6.61633e+010 3.30816e+010 1.65408e+010 4.96225e
+010 2.48112e+010 1.24056e+010 6.20281e+009 1.86084e+010 9.30421e+009 2.79126e+0
10 1.39563e+010 4.18689e+010 2.09345e+010 1.04672e+010 3.14017e+010 1.57009e+010
 4.71026e+010 2.35513e+010 7.06538e+010 3.53269e+010 1.05981e+011 5.29904e+010 2
.64952e+010 1.32476e+010 6.6238e+009 3.3119e+009 1.65595e+009 8.27975e+008 4.139
87e+008 2.06994e+008 6.20981e+008 3.10491e+008 1.55245e+008 7.76226e+007 2.32868
e+008 1.16434e+008 3.49302e+008 1.74651e+008 8.73255e+007 2.61976e+008 1.30988e+
008 6.54941e+007 1.96482e+008 9.82411e+007 2.94723e+008 1.47362e+008 7.36809e+00
7 3.68404e+007 1.10521e+008 5.52606e+007 1.65782e+008 8.2891e+007 2.48673e+008 1
.24336e+008 3.73009e+008 1.86505e+008 9.32523e+007 2.79757e+008 1.39879e+008 4.1
9636e+008 2.09818e+008 6.29453e+008 3.14727e+008 1.57363e+008 7.86817e+007 2.360
45e+008 1.18022e+008 5.90112e+007 1.77034e+008 8.85169e+007 4.42584e+007 1.32775
e+008 6.63877e+007 3.31938e+007 1.65969e+007 8.29846e+006 2.48954e+007 1.24477e+
007 6.22384e+006 1.86715e+007 9.33577e+006 2.80073e+007 1.40036e+007 7.00182e+00
6 3.50091e+006 1.75046e+006 875228 437614 218807 656422 328211 984634 492317 1.4
7695e+006 738476 369238 184619 553858 276929 830788 415394 207697 623092 311546
155773 467320 233660 116830 58415 175246 87623 262870 131435 394306 197153 59146
0 295730 147865 443596 221798 110899 332698 166349 499048 249524 124762 62381 18
7144 93572 46786 23393 70180 35090 17545 52636 26318 13159 39478 19739 59218 296
09 88828 44414 22207 66622 33311 99934 49967 149902 74951 224854 112427 337282 1
68641 505924 252962 126481 379444 189722 94861 284584 142292 71146 35573 106720
53360 26680 13340 6670 3335 10006 5003 15010 7505 22516 11258 5629 16888 8444 42
22 2111 6334 3167 9502 4751 14254 7127 21382 10691 32074 16037 48112 24056 12028
 6014 3007 9022 4511 13534 6767 20302 10151 30454 15227 45682 22841 68524 34262
17131 51394 25697 77092 38546 19273 57820 28910 14455 43366 21683 65050 32525 97
576 48788 24394 12197 36592 18296 9148 4574 2287 6862 3431 10294 5147 15442 7721
 23164 11582 5791 17374 8687 26062 13031 39094 19547 58642 29321 87964 43982 219
91 65974 32987 98962 49481 148444 74222 37111 111334 55667 167002 83501 250504 1
25252 62626 31313 93940 46970 23485 70456 35228 17614 8807 26422 13211 39634 198
17 59452 29726 14863 44590 22295 66886 33443 100330 50165 150496 75248 37624 188
12 9406 4703 14110 7055 21166 10583 31750 15875 47626 23813 71440 35720 17860 89
30 4465 13396 6698 3349 10048 5024 2512 1256 628 314 157 472 236 118 59 178 89 2
68 134 67 202 101 304 152 76 38 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10 5 1
6 8 4 2 1
```

Inconsistently, an increasing to 10^{72} generates only 496 operations For 10^{84}, 622 instruction runs are required, and for 10^{168} - 880 operations. Number of operations required thus grows more slowly than log n, where n is the number of digits in the initial number.

A relationship between the number of iterations from the initial numbers for the "small" numbers in the problem $3n$+1 are shown in Fig. 12.1.

An important consequence of the independence of the calculation is that there is no need to synchronize the state between various machines involved in the task. This is particularly important for grid systems because some individual nodes can be located in geographically different locations. The communication link between such nodes is characterized by relatively low-speed data transmission and long delays. Thus, all communication involves a significant overhead in time. This is in contrast to cluster computing, where we have a fast connection between nodes, which are characterized by low latency. The Infiniband network is characterized by a rate of gigabit per second and a delay of microseconds. For the transmission between nodes on grid computing system, it is expected that a rate is megabits per second and a delay can be reached hundreds of milliseconds.

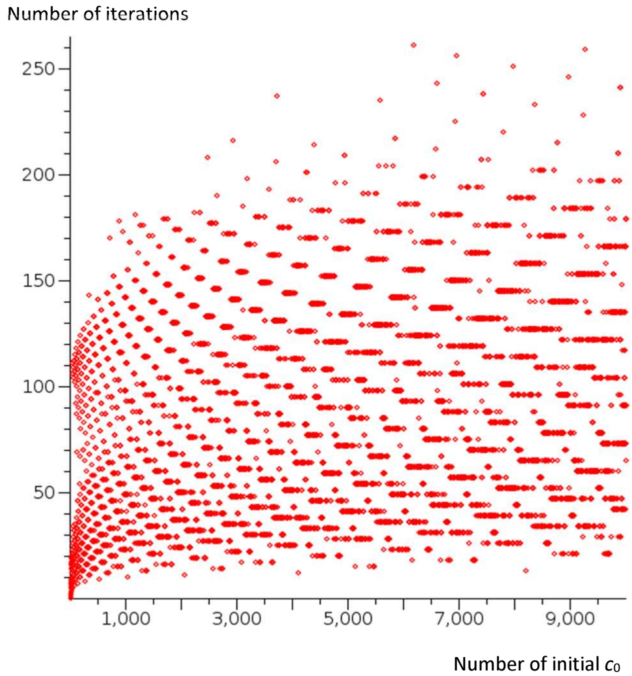

Fig. 12.1. A relationship between the number of iterations and the initial numbers (that are not more than 10 000) in the Collatz problem [19]

Another important feature in terms of calculations in a grid system is the ratio of computation time to communication time. As shown earlier, data grid systems are expensive. In case of problems needing to send large amounts of data and relatively simple calculations, the acceleration due to multiply calculations are limited by communication overhead and speed up the whole system will be small in comparison to the system operating sequentially. In testing the Collatz hypothesis, an input data for computing node is the range of numbers to be tested. We can represent it as an ordered pair (initial value of the range, number of consecutive numbers to be tested).

A linear increase in the size of the input data is associated with the exponential increase in the amount of calculation. Thus, we can arbitrarily extend the computation time on a single node, without increasing significantly the amount of input data. It should be noted that the time extension to process a single packet of input data increases the risk that we do not get a result for this data parcel because of a communication link failure or disconnection of a computational node (i.e., in a system based on the idea of volunteer computing). A computation time should therefore be chosen to be sufficiently long in relation to the communication time, but short enough to keep the probability of receiving a response still high. In the case of the one-time calculation of BOINC was set at 15 seconds [3, 14].

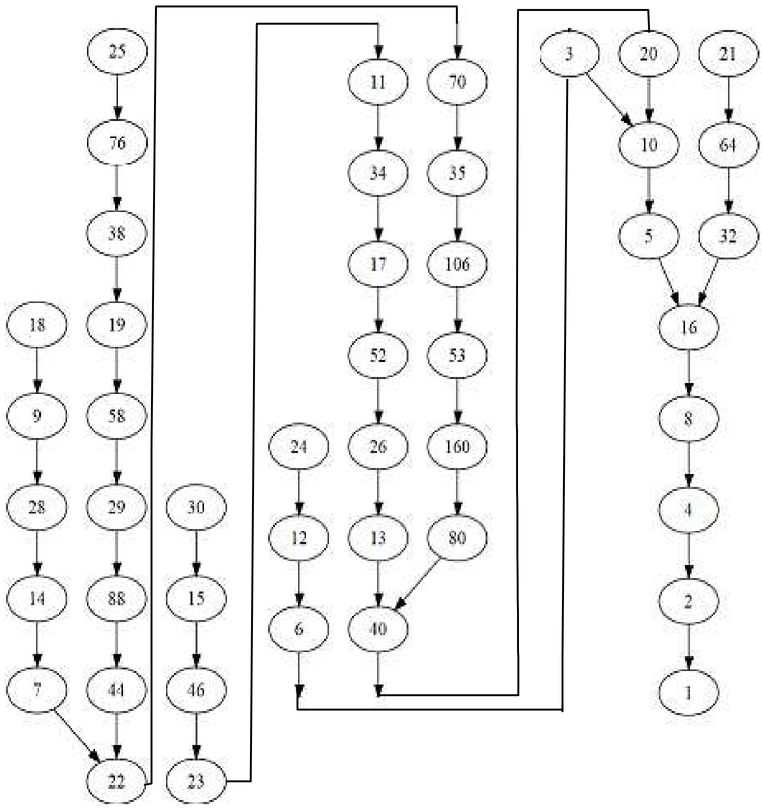

Fig. 12.2. Some paths in the directed graph representing computational processes in the Ulama problem for the initial numbers not greater than 30 except 27 [18]

Internet users involved in the program BOINC $3x$ +1 @ home have the aim to determine the counterexample to the hypothesis Collatz. The website has already closed the project, but we can find a list of candidate numbers for which the string length before reaching the final sequence {16,8,4,2,1} was 1000 times [15].

An example of such a number, which was able to determine by means of grid Comcute, is the value recorded below. A sequence of numbers is 1004 long.

```
Wprowadz liczbē calkowita jako poczatek ciagu Collatza:2827272377374747474747474
84847474464647747474774747474747477474747474774747474848484848484848477473773738388474
74747474774747474747474774747474747474747747474747474774848484847477474763673747477474
774727272727727737373737
1.41364e+206 7.06818e+205 3.53409e+205 1.76705e+205 8.83523e+204 4.41761e+204 2.
20881e+204 1.1044e+204 5.52202e+203 2.76101e+203 1.3805e+203 6.90252e+202 3.4512
6e+202 1.72563e+202 8.62815e+201 4.31408e+201 2.15704e+201 1.07852e+201 5.39259e
+200 2.6963e+200 1.34815e+200 6.74074e+199 3.37037e+199 1.68519e+199 8.42593e+19
8 4.21296e+198 2.10648e+198 1.05324e+198 5.26621e+197 2.6331e+197 1.31655e+197 6
.58276e+196 3.29138e+196 1.64569e+196 8.22845e+195 4.11422e+195 2.05711e+195 1.0
2856e+195 5.14278e+194 2.57139e+194 1.28569e+194 6.42847e+193 3.21424e+193 1.607
12e+193 8.03559e+192 4.0178e+192 2.0089e+192 1.00445e+192 5.02224e+191 2.51112e+
191 1.25556e+191 6.27781e+190 3.1389e+190 1.56945e+190 7.84726e+189 3.92363e+189
 1.96181e+189 9.80907e+188 4.90454e+188 2.45227e+188 1.22613e+188 6.13067e+187 3
.06533e+187 1.53267e+187 7.66334e+186 3.83167e+186 1.91583e+186 9.57917e+185 4.7
8959e+185 2.39479e+185 1.1974e+185 5.98698e+184 2.99349e+184 1.49675e+184 7.4837
3e+183 3.74186e+183 1.87093e+183 9.35466e+182 4.67733e+182 2.33866e+182 1.16933e
+182 5.84666e+181 2.92333e+181 1.46167e+181 7.30833e+180 3.65416e+180 1.82708e+1
80 9.13541e+179 4.5677e+179 2.28385e+179 1.14193e+179 5.70963e+178 2.85482e+178
1.42741e+178 7.13704e+177 3.56852e+177 1.78426e+177 8.9213e+176 4.46065e+176 2.2
3032e+176 1.11516e+176 5.57581e+175 2.78791e+175 1.39395e+175 6.96976e+174 3.484
88e+174 1.74244e+174 8.71221e+173 4.3561e+173 2.17805e+173 1.08903e+173 5.44513e
+172 2.72256e+172 1.36128e+172 6.80641e+171 3.40321e+171 1.7016e+171 8.50801e+17
0 4.25401e+170 2.127e+170 1.0635e+170 5.31751e+169 2.65875e+169 1.32938e+169 6.6
4689e+168 3.32344e+168 1.66172e+168 8.30861e+167 4.1543e+167 2.07715e+167 1.0385
8e+167 5.19288e+166 2.59644e+166 1.29822e+166 6.4911e+165 3.24555e+165 1.62277e+
165 8.11387e+164 4.05694e+164 2.02847e+164 1.01423e+164 5.07117e+163 2.53559e+16
3 1.26779e+163 6.33896e+162 3.16948e+162 1.58474e+162 7.9237e+161 3.96185e+161 1
.98093e+161 9.90463e+160 4.95232e+160 2.47616e+160 1.23808e+160 6.19039e+159 3.0
952e+159 1.5476e+159 7.73799e+158 3.869e+158 1.9345e+158 9.67249e+157 4.83625e+1
57 2.41812e+157 1.20906e+157 6.04531e+156 3.02265e+156 1.51133e+156 7.55663e+155
 3.77832e+155 1.88916e+155 9.44579e+154 4.7229e+154 2.36145e+154 1.18072e+154 5.
90362e+153 2.95181e+153 1.47591e+153 7.37953e+152 3.68976e+152 1.84488e+152 9.22
441e+151 4.6122e+151 2.3061e+151 1.15305e+151 5.76525e+150 2.88263e+150 1.44131e
+150 7.20657e+149 3.60328e+149 1.80164e+149 9.00821e+148 4.5041e+148 2.25205e+14
8 1.12603e+148 5.63013e+147 2.81507e+147 1.40753e+147 7.03766e+146 3.51883e+146
1.75942e+146 8.79708e+145 4.39854e+145 2.19927e+145 1.09963e+145 5.49817e+144 2.
74909e+144 1.37454e+144 6.87272e+143 3.43636e+143 1.71818e+143 8.5909e+142 4.295
45e+142 2.14772e+142 1.07386e+142 5.36931e+141 2.68466e+141 1.34233e+141 6.71164
e+140 3.35582e+140 1.67791e+140 8.38955e+139 4.19477e+139 2.09739e+139 1.04869e+
139 5.24347e+138 2.62173e+138 1.31087e+138 6.55434e+137 3.27717e+137 1.63858e+13
7 8.19292e+136 4.09646e+136 2.04823e+136 1.02411e+136 5.12057e+135 2.56029e+135
1.28014e+135 6.40072e+134 3.20036e+134 1.60018e+134 8.0009e+133 4.00045e+133 2.0
0022e+133 1.00011e+133 5.00056e+132 2.50028e+132 1.25014e+132 6.2507e+131 3.1253
5e+131 1.56268e+131 7.81338e+130 3.90669e+130 1.95334e+130 9.76672e+129 4.88336e
+129 2.44168e+129 1.22084e+129 6.1042e+128 3.0521e+128 1.52605e+128 7.63025e+127
 3.81513e+127 1.90756e+127 9.53781e+126 4.76891e+126 2.38445e+126 1.19223e+126 5
.96113e+125 2.98057e+125 1.49028e+125 7.45142e+124 3.72571e+124 1.86285e+124 9.3
1427e+123 4.65714e+123 2.32857e+123 1.16428e+123 5.82142e+122 2.91071e+122 1.455
35e+122 7.27677e+121 3.63839e+121 1.81919e+121 9.09597e+120 4.54798e+120 2.27399
e+120 1.137e+120 5.68498e+119 2.84249e+119 1.42124e+119 7.10622e+118 3.55311e+11
8 1.77656e+118 8.88278e+117 4.44139e+117 2.2207e+117 1.11035e+117 5.55174e+116 2
.77587e+116 1.38793e+116 6.93967e+115 3.46984e+115 1.73492e+115 8.67459e+114 4.3
373e+114 2.16865e+114 1.08432e+114 5.42162e+113 2.71081e+113 1.3554e+113 6.77702
e+112 3.38851e+112 1.69426e+112 8.47128e+111 4.23564e+111 2.11782e+111 1.05891e+
111 5.29455e+110 2.64727e+110 1.32364e+110 6.61819e+109 3.30909e+109 1.65455e+10
9 8.27273e+108 4.13637e+108 2.06818e+108 1.03409e+108 5.17046e+107 2.58523e+107
1.29261e+107 6.46307e+106 3.23154e+106 1.61577e+106 8.07884e+105 4.03942e+105 2.
01971e+105 1.00986e+105 5.04928e+104 2.52464e+104 1.26232e+104 6.3116e+103 3.155
8e+103 1.5779e+103 7.88949e+102 3.94475e+102 1.97237e+102 9.86187e+101 4.93093e+
101 2.46547e+101 1.23273e+101 6.16367e+100 3.08183e+100 1.54092e+100 7.70458e+09
9 3.85229e+099 1.92615e+099 9.63073e+098 4.81536e+098 2.40768e+098 1.20384e+098
6.01921e+097 3.0096e+097 1.5048e+097 7.52401e+096 3.762e+096 1.881e+096 9.40501e
+095 4.7025e+095 2.35125e+095 1.17563e+095 5.87813e+094 2.93907e+094 1.46953e+09
4 7.34766e+093 3.67383e+093 1.83692e+093 9.18458e+092 4.59229e+092 2.29614e+092
1.14807e+092 5.74036e+091 2.87018e+091 1.43509e+091 7.17545e+090 3.58773e+090 1.
79386e+090 8.96932e+089 4.48466e+089 2.24233e+089 1.12116e+089 5.60582e+088 2.80
291e+088 1.40146e+088 7.00728e+087 3.50364e+087 1.75182e+087 8.7591e+086 4.37955
e+086 2.18977e+086 1.09489e+086 5.47444e+085 2.73722e+085 1.36861e+085 6.84305e+
084 3.42152e+084 1.71076e+084 8.55381e+083 4.2769e+083 2.13845e+083 1.06923e+083
 5.34613e+082 2.67306e+082 1.33653e+082 6.68266e+081 3.34133e+081 1.67067e+081 8
.35333e+080 4.17666e+080 2.08833e+080 1.04417e+080 5.22083e+079 2.61041e+079 1.3
0521e+079 6.52604e+078 3.26302e+078 1.63151e+078 8.15755e+077 4.07877e+077 2.039
39e+077 1.01969e+077 5.09847e+076 2.54923e+076 1.27462e+076 6.37308e+075 3.18654
```

A continuation of the completed project 3*x*+1@home is a Collatz Conjecture program that uses the BOINC infrastructure.

Another important aspect is the size of the output, which must be returned after the calculation. In the case of Collatz problem, it can be assumed that the output is one of two forms:

1. single, one-byte tag, indicating that at the interval could not be found a number that discredit a hypothesis;
2. number on which it was discovered that contradicts the hypothesis.

Finding numbers, contradicting the hypothesis Collatz, completes the calculation. Thus, the amount of output for the problem is small.

12.5. Optimization of the calculation grid in the system

While testing the correctness of the hypothesis of the Collatz sequence, the two possible optimizations can be introduced. First of these is associated with the observation that if it is generating another sequence for a number of initial *n*, then it will achieve the number *m,* which the previously have been demonstrated that it met the hypothesis, the conclusion is the number n also satisfies it.

This follows from the fact that after obtaining the number m, every subsequence is the same as for the sequence that began with the numbers m and reached a value of 1. Based on this observation, we can shorten the testing time and stop the calculation when we meet a previously tested number. However, it requires the archive with information on previously verified numbers. In the grid system, some compute nodes would send all the numbers that do not contradict the hypothesis. This would increase significantly the amount of communication between nodes and this wouldlead to a drastic reduction in performance.

Second possible optimization is based on the observation that to obtain the number of c_n= 1 in the sequence $c_{1,}c_{2,}c_3$ ... c_n defined according to formula (4) not only the number of c_1 does not contradict the hypothesis, but also all the numbers $c_{2,}c_3$... c_{n-1} do not contradict it. Using this observation also requires a central repository of verified numbers. Amount of data needed to transfer via the grid system would be even more than in the previous case. Computing nodes, in addition to the initial numbers, would also send all the items on the way to the element with a value of 1. Formula (4) explicitly shows that the number of such elements is at least equal to the logarithm with base 2 of the initial number. Thus also this modification is not optimized for computing on the grid system.

As a result, it was assumed that if the number in the calculation by dividing by 2 is less than 10^{18} the hypothesis will no longer be questioned and further transform this sequence does not make sense.

Kurt Gödel formulated a theorem in 1930, as follows. If the theory T consists of the arithmetic sentences related to natural numbers, then there is a sentence $A(x)$ such that, although all sentences

$$A(0), A(1), A(2), ...$$

are theorems in T, but this general statement

for every natural number x there is A(x)

nor its negation cannot be deduced.

An example of such a sentence is a stop problem in computer software run. Suppose we are given a program P. Does this program run in a finite time? Do an universal algorithm exist deciding whether to stop the other algorithms?

Of course, if we could call into question the hypothesis Collatz, it would not be a proof of Gödel's theorem were false.

12.6. Scalability verification

The scalability of the Comcute grid system were tested in accordance with the Collatz hypothesis (Fig. 12.3).

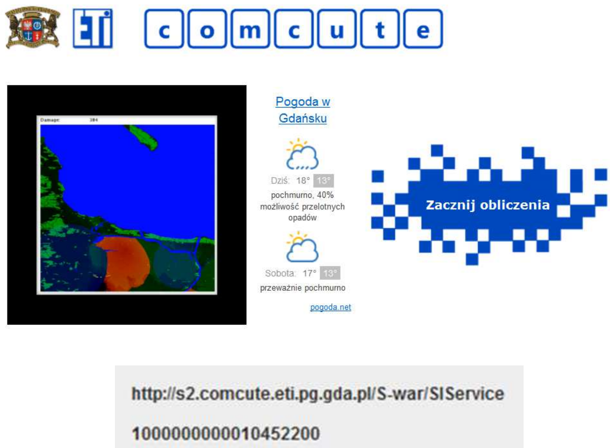

Fig. 12.3. Graphical user interface

After testing some data a packet of the limited value 1,000,000,000,010,452,200, is returned (0 - failure) to the server *S1,* which sends it to the supervising server *W1.* In the case that the hypothesis has been disproved, the value of a number is sent. The code does not increase memory on the web computer what is presented in Fig. 12.4 in the *history of the use of physical memory*.

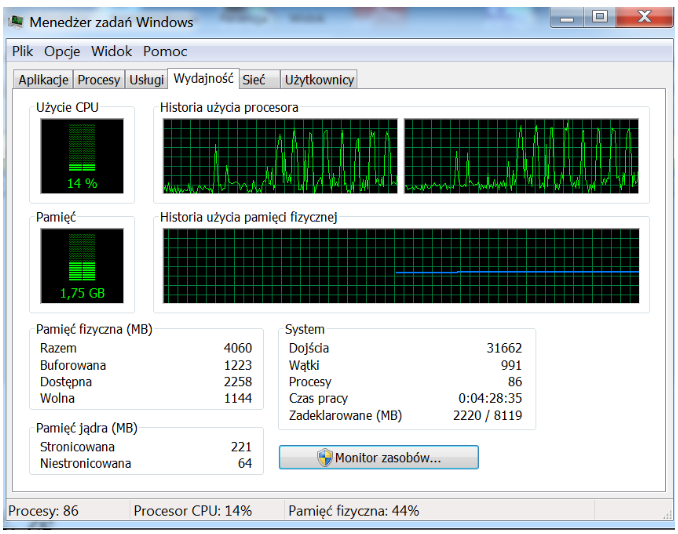

Fig. 12.4. The workload of computer resources

The user may experience loss of calculation power, which shows *CPU usage history* in Fig. 12.5. The maximum CPU usage equal to 90% is achieved in the periods when the processor is processing some data packet from the server *S*.

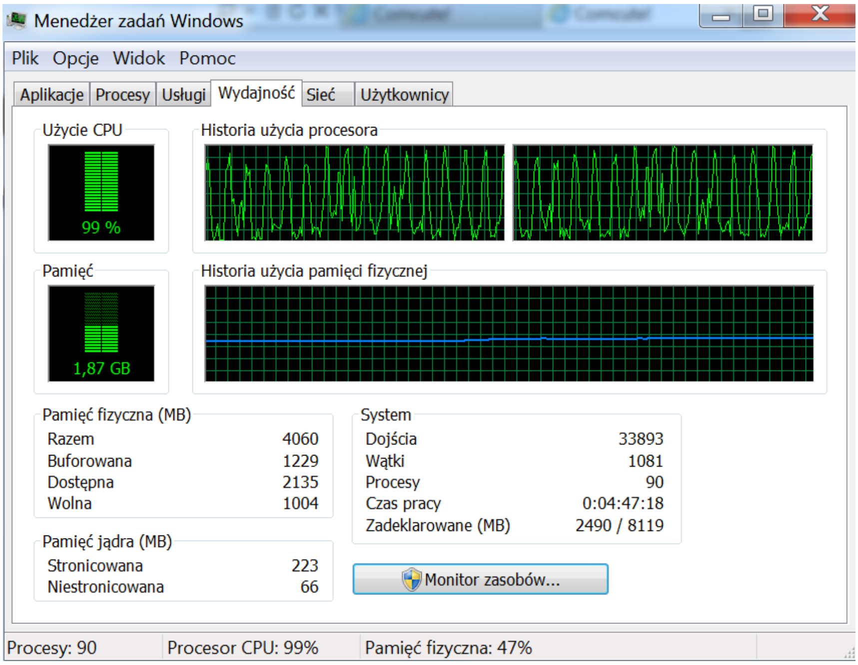

Fig. 12.5. The workload of computer resources at the two parallel grid calculations

In contrast, the minimum value corresponds to approximately 10% situations while the transmission takes place from the server. Thus, the faster the network, the greater the efficiency of calculations. In addition, the data packet should be sent in the size that dependents on bandwidth connections. They can be larger for greater bandwidth. Moreover, a packet size depends on the performance of user's computer (the more powerful, the greater the packet data). In the case of two Comcute tasks at a web computer, the CPU usage slightly increased (up to 99%), but also regularly decreased to the amount of a few percent (Fig. 12.5). Some popular activities on this computer such as preparing documents and presentations, or surfing via the Internet can be done without important delays.

These two tasks were made using Internet Explorer version 9. In the event of a third call from Firefox and running the third and fourth tasks, the CPU usage also slightly increased (up to 99%), but also regularly decreased (Fig. 12.6).

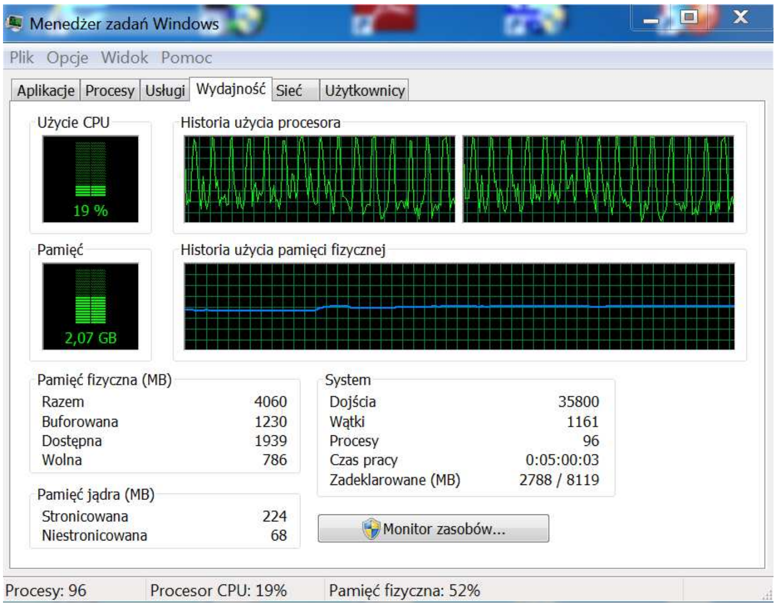

Fig. 12.6. The workload of computer resources on four parallel grid calculations

Each new task causes a slight increase in occupancy of memory, and a call for a task in a new browser increases its occupancy to approximately 120 MB. Fig. 12.7 shows the load on an Internet resources for ten tasks performed by browsers like IE, Firefox and Safari.

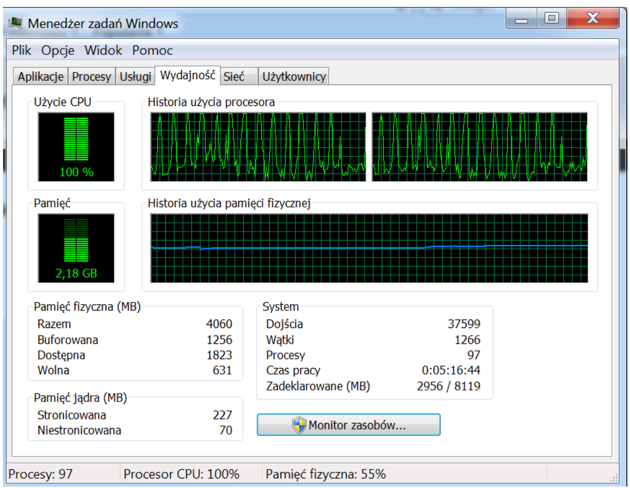

Fig. 12.7. The workload of computer resources at ten tasks performed by the three browsers

12.7. Summary

The Collatz hypothesis has features that allow for their effective processing by a grid system. The most important are: the independence of the calculation and the preferred ratio of computation time to the time of communication, which can be adjusted by modifying the input data. In addition, optimizations that reduce the computation time in the case of sequential execution may adversely effect on the performance of a grid..

Collatz problem can be treated with two scenarios. In the first scenario, we aim to overthrow the hypothesis, and in the second variant, we are looking for numbers that are characterized by a long string that leads to one.

The Comcute system has been tested for hundreds computers of web users in 2012 what made it possible to run at the same time about two thousand processes.

Our future works will focus on testing the other sets of tasks to develop this grid platform. Moreover, we will concern on new algorithms development for task and data management.

References

1. Andrei S., Masalagiu C.: *About the Collatz conjecture*. Acta Informatica 35 (2): 167. , 1998
2. Balicki J.: *Immune systems in multi-criterion evolutionary algorithm for task assignments in distributed computer system*. LNCS, Vol. 3528, 2005, pp. 51-56.
3. Belaga E. G., Mignotte M.: *Embedding the 3x+1 Conjecture in a 3x+d Context*, Experimental Mathematics, Volume 7, issue 2, 1998
4. Bruschi, M.: *A generalization of the Collatz problem and conjecture*. arXiv:0810.5169 [math.NT], 2008
5. Chamberland M.: *A continuous extension of the 3x + 1 problem to the real line*. Dynam. Contin. Discrete Impuls Systems 2:4 pp. 495–509, 1996
6. De Mol, Liesbeth: *Tag systems and Collatz-like functions*, Theoretical Computer Science, 390:1, 92–101, January 2008
7. Guy R. K.: *Unsolved problems in Number Theory,* E17: Permutation Sequences and the Collatz Conjecture, pp.336–337, Springer, 2004
8. Lagarias J. C.: *The 3x + 1 problem: An annotated bibliography*, II, arXiv:math.NT/0608208, 2006
9. Lagarias J. C: *Syracuse problem*, in Hazewinkel, Michiel, Encyclopedia of Mathematics, Springer, 2001
10. Lehmer D. H., *On Lucas's Test for the Primality of Mersenne's Numbers*. J. London Math. Soc. v. 10, 1935, p. 162-165

11. Letherman S., Schleicher D., Wood R.: *The (3n + 1)-Problem and Holomorphic Dynamics*. Experimental Mathematics 8:3, pp.241–252, 1999
12. Matthews K., *The 3x+1 Problem*: An Annotated Bibliography, II (2000-2009), Cornell University Library
13. Peterson I., *Priming for a Lucky Strike*. Science News. v. 133, no. 6, 1988, p. 85
14. Simons J., de Weger B.: *Theoretical and computational bounds for m-cycles of the 3n + 1 problem*, Acta Arithmetica, (online version 1.0, November 18, 2003), 2005
15. Sinyor J.: *The 3x+1 Problem as a String Rewriting System*, International Journal of Mathematics and Mathematical Sciences, Volume 2010
16. Steiner, R.P.: *A theorem on the Syracuse problem*, *Proceedings of the 7th Manitoba Conference on Numerical Mathematics*, pp. 553–559, 1977
17. Van Bendegem, J. P.: *The Collatz Conjecture: A Case Study in Mathematical Problem Solving*, Logic and Logical Philosophy, Volume 14 (2005), 7–23
18. Vance A.: *Sun and UC Berkeley are about to BOINC*. The Register, 2003
19. Wirsching G. J.: *The Dynamical System Generated by the 3n+1 Function.* Lecture Notes in Mathematics, No. 1681. Springer-Verlag, 1998
20. Wirsching G. J: The Dynamical System Generated by the 3n+1 Function. Number 1681 in Lecture Notes in Mathematics. Springer-Verlag, 1998