Weryfikacja „słabej” hipotezy Goldbacha do 10^{31}

Verifying the „weak” Goldbach conjecture up to 10^{31}

Łukasz Świerczewski[1]

Treść. Praca prezentuje aspekt numerycznej weryfikacji „słabej” hipotezy Goldbacha dla wartości mniejszych niż 10^{31}. Do obliczeń, które zajęły w sumie ok. 50 000 godzin czasu pojedynczego CPU wykorzystano klaster wydajnościowy złożony z procesorów AMD Opteron 4284. Podczas sprawdzania pierwszości zastosowano test Millera-Rabina. Przetestowano także możliwe zastosowanie testu ECPP. Jak się okazało przy założeniu dodatkowych warunków poprawności testu Millera-Rabina „słaba” hipoteza Goldbacha w badanym zakresie jest prawidłowa.

Słowa kluczowe: teoria liczb, hipoteza Goldbacha, liczby pierwsze

Abstrackt. This paper presents aspect of the numerical verification a „weak” Goldbach's conjecture for values less than 10^{31}. For calculations, that took about 50 000 hours of a single CPU performance, there was used an performance cluster consisting of the AMD Opteron 4284 processors. During the primality check, there was used Miller-Rabin test. There was also tested the possiblity of ECPP test usage. As it turned out, when there were added some additional conditions of correctness of Miller-Rabin test, the „weak” Goldbach's conjecture occurs correct in researched range.

Key words: number theory, Goldbach conjecture, primes

1. Wprowadzenie

Hipoteza Goldbacha zakłada, że każdą liczbę parzystą większą od 2 można przedstawić jako sumę dwóch liczb pierwszych. Problem ten pierwotnie sformułował Goldbach w liście do Eulera w 1742 roku. Istnieje także tzw. „słaba” hipoteza Goldbacha mówiąca że każda liczba naturalna nieparzysta większa od 7 jest sumą trzech nieparzystych liczb pierwszych *(niekoniecznie różnych).* W 1937 roku Iwan Vinogradov udowodnił [1], że każdą dostatecznie dużą liczbę nieparzystą można przedstawić w postaci sumy trzech liczb pierwszych. Wynik ten poprawili w 2002 roku Liu Ming-Chit i Wang Tian-Ze z Uniwersytetu w Hong Kongu [2]. Udowodnili oni, że „dostatecznie duża” liczba oznacza większa od e^{3100} (w przybliżeniu 10^{1346}).

2. Algorytm

Wykorzystany algorytm został szczegółowo opisany w publikacji Yannick'a Saouter'a z 1998 roku [3]. Wygląda on następująco:

1. Do p_0 wpisz 100000000209366024193 i przejdź do kroku 2.
2. Jeżeli p_i jest liczbą pierwszą to inkrementuj i oraz p_i zwiększ o $95367431640 \cdot 2^{22}$ i przejdź do kroku 4. W przeciwnym wypadku przejdź do kroku 3.
3. Inkrementuj i oraz p_i zmniejsz o $10 \cdot 2^{22}$ i przejdź do kroku 2.
4. Jeżeli $p_i < 10^{31}$ wykonuj dalej obliczenia i przejdź do kroku 2. W przeciwnym wypadku przerwij działanie programu.

W pierwotnym artykule w kroku pierwszym algorytmu do p_0 została przypisywana wartość 138412033, a w kroku drugim p_i było zwiększane o $95360 \cdot 2^{22}$. Dane te jednak zaktualizowano biorąc pod uwagę najnowsze osiągnięcia. Wartość 138412033 została zwiększona ze względu na rozpoczęcie obliczeń tam gdzie zostały one w publikacji [3] zakończone. Zastąpienie $95360 \cdot 2^{22}$ wartością $95367431640 \cdot 2^{22}$ wynika z tego, że na czas pisania artykułu „mocną” hipotezę Goldbacha dwukrotnie sprawdzono już do $4 \cdot 10^{17}$ [4] (jednokrotnie aż do $4 \cdot 10^{18}$). $95367431640 \cdot 2^{22}$ jest największą wielokrotnością liczby $10 \cdot 2^{22}$ mniejszą od $4 \cdot 10^{17}$.

3. Implementacja algorytmu

Do obliczeń wykorzystano algorytm zaimplementowany z wykorzystaniem zoptymalizowanej pod kątem wykorzystanych procesorów biblioteki GMP (GNU Multiple Precision Arithmetic Library) [5]. Podczas testowania pierwszości zastosowano test Millera-Rabina [6] z określonymi świadkami pierwszości - wybrano w tym celu 20 najmniejszych liczb pierwszych. Tego typu rozwiązanie zostało teoretycznie rozpatrzone przez Zhang'a [7] i w połączeniu z testem Millera-Rabina można w ten sposób uzyskać bardzo szybki test pierwszości dla liczb mniejszych niż 10^{36}.

Przetestowano także wydajność rozwiązania wykorzystującego dodatkowo test ECPP [9]. W tym algorytmie początkowo sprawdzano pierwszość z wykorzystaniem testu Millera-Rabina, a jeżeli okazało się, że dana liczba jest

1 Instytut Informatyki i Automatyki, Państwowa Wyższa Szkoła Informatyki i Przedsiębiorczości w Łomży, lswierczewski@pwsip.edu.pl

według niego pierwsza wykonywano dodatkowo całkowicie deterministyczny test ECPP. Program wykorzystujący także ECPP okazał się znacząco wolniejszy i z jego wykorzystaniem zweryfikowano hipotezę tylko do 10^{24}. Na Listingu 1 przedstawiono główną część kodu realizującego weryfikację „słabej" hipotezy Goldbacha. Kod ten został napisany w języku C i standardzie OpenMP umożliwiającym wykonywanie obliczeń na komputerach równoległych z pamięcią wspólną. Za pomocą makr MILLER_RABIN oraz ECPP w przypadku tego listingu zdefiniowane jest użycie różnych algorytmów testowania pierwszości. Ze względu na ograniczone miejsce w tej pracy nie zaprezentuję kodu odpowiedzialnego za przetwarzanie danych na klastrach komputerowych. Jest on analogiczny do tego działającego z wykorzystaniem OpenMP lecz zostały w nim użyte funkcje interfejsu MPI.

```
    #pragma omp parallel shared(S, add_1, add_2, number_of_threads) private(thread_id, number, upper_range, counter)
   {
      thread_id = omp_get_thread_num();

      mpz_set(number, table_down[thread_id]);
      mpz_set(upper_range, table_upper[thread_id]);

      while( mpz_cmp(number, upper_range) == -1 )
      {

   # ifdef MILLER_RABIN
         if(miller_rabin(number, S) == 1)
   # endif
   # ifdef ECPP
         if(gmp_ecpp(mpz_class(number)) == 1)
   # endif
         {
            mpz_add_ui(number, number, add_1);
         }
         else
         {
            mpz_sub_ui(number, number, add_2);
         }
      }
   }
```

Listing 1. Implementacja części głównej kodu realizującego weryfikację „słabej" hipotezy Goldbacha

Listing 1. The implementation of the main part of the code performing a „weak" Goldbach's Conjecture verification.

Na Listing 2 została zaprezentowana funkcja główna realizująca test Millera-Rabina. Jako argumenty przyjmuje ona liczbę, która jest testowana oraz tablicę liczb pierwszych. Funkcja ta wywołuje funkcję pomocniczą miller_rabin_pass(m, k), która wykonuje test liczby m względem świadka pierwszości k.

```
int miller_rabin(mpz_t n, char *S)
{

   mpz_t pom;
   mpz_init(pom);
   mpz_set_ui(pom, 2);

   if(miller_rabin_pass(pom, n) == 0)
   {
      mpz_clear(pom);
      return 0;
   }


   for(unsigned short int i = 3; i < 72; i += 2)
   {
      if(S[i] == 1)
      {
         mpz_set_ui(pom, i);

         if (miller_rabin_pass(pom, n) == 0)
         {
            mpz_clear(pom);
            return 0;
         }
      }
   }
   mpz_clear(pom);

   return 1;
}
```

Listing 2. Implementacja funkcji głównej realizującej test pierwszości Millera-Rabina

Listing 2. The implementation of the main function performing Miller-Rabin primality test.

4. Obliczenia

Obliczenia wykonano na klastrze wydajnościowym złożonym z procesorów AMD Opteron 4284 i zajęły one w sumie 50000 godzin czasu pracy pojedynczego CPU. W oprogramowaniu wykorzystano środowisko MPI [12]. Ostatnią liczbą pierwszą jaką wygenerował program używający testu Millera-Rabina była 100000000000003998 61783893901313. W Tabeli 1 przedstawiono 30 ostatnich liczb pierwszych jakie wygenerowało oprogramowanie wykorzystujące test Millera-Rabina. Analogiczną tabelą dla rozwiązania całkowicie deterministycznego opartego także na teście ECPP jest Tabela 2.

Tab. 1. Ostatnich 30 liczb jakie wygenerował algorytm wykorzystujący test Millera-Rabina.
Tab. 1. The last 30 numbers generated by the algorithm which uses Miller-Rabin test.

999999999998879986181739852595 3	999999999999479986179764597555 3
999999999998919986181676675891 3	999999999999519986179697226547 3
999999999998959986181655442227 3	999999999999559986179487249203 3
999999999998999986181290275635 3	999999999999599986179319214899 3
999999999999039986181147407155 3	999999999999639986179247649587 3
999999999999079986181126173491 3	999999999999679986179234804531 3
999999999999119986180949750579 3	999999999999719986178991272755 3
999999999999159986180932711219 3	999999999999759986178970039091 3
999999999999199986180731122483 3	999999999999799986178969776947 3
999999999999239986180546310963 3	999999999999839986178818519859 3
999999999999279986180546048819 3	999999999999879986178654679859 3
999999999999319986180407374643 3	999999999999919986178654417715 3
999999999999359986180352586547 3	999999999999959986178498966323 3
999999999999399986180176163635 3	999999999999999986178389652275 3
999999999999439986180092015411 3	100000000000003998617838939013 13

Tab. 2. Ostatnich 30 liczb jakie wygenerował algorytm wykorzystujący test Millera-Rabina oraz ECPP.
Tab. 2. The last 30 numbers generated by the algorithm which uses Miller-Rabin and ECPP tests.

999988699999803877097473	999994699999792429268993
999989099999802658127873	999950999997920072171 53
999989499999802110246913	999954999997916690513 93
999989899999802107625473	999958999997891498475 53
999990299999801853345793	999996299999789105283073
999990699999801599066113	999996699999788767117313
999991099999801009242113	999997099999787380375553
999991499999800754962433	999997499999786706665473
999991899999800249024513	999997899999785697411073
999992299999800246403073	999998299999785065644033
999992699999799950180353	999998699999784685535233
999993099999795669368833	999999099999783969882113
999993499999794408456193	999999499999783422001153
999993899999793399201793	999999899999781406113793
999994299999792515776513	100000029999978140349235 3

5. Wnioski

Osiągnięcie granicy 10^{1346} jest całkowicie nierealne przy zastosowaniu współczesnych algorytmów i technik przetwarzania danych. Udowodniono, że przy założeniu poprawności uogólnionej hipotezy Riemmana granica ta może zostać obniżona do $3.2 \cdot 10^{49}$ [8] co jednak także nie będzie w naszym zasięgu w ciągu najbliższych lat. Postęp w badaniach dotyczących hipotezy Goldbacha jest jednak ogromny. W sierpniu 2012 roku Agostino Prástaro opublikował dowód „mocnej” hipotezy Goldbacha [10], a w maju 2013 roku H. A. Helfgott w swoim artykule [11] zamieścił dowód „słabej” hipotezy Goldbacha dla wartości większych niż 10^{29}. Obydwa dowody jednak na czas pisania niniejszej publikacji nie przeszły pozytywnie weryfikacji. Niniejsza praca jest jest drobnym krokiem ku potwierdzeniu poprawności hipotezy Goldbacha. Zaimplementowane na potrzeby artykułu kody programów mogą w przyszłości umożliwić weryfikację hipotezy w jeszcze większym zakresie.

Literatura (References)

[1] I.M. Vinogradov, Representation of an odd number as the sum of three primes, Dokl. Akad. Nauk SSSR (1937), no. 15, 169-172.
[2] MC Liu, T. Z. Wang, On the Vinogradov bound in the three primes Goldbach conjecture, Acta Arithmetica 105 (2002): 133-175.
[3] Y. Saouter, Checking the odd Goldbach conjecture up to 10^{20}, Mathematics of Computation of the American Mathematical Society 67.222 (1998): 863-866.
[4] Tomás Oliveira e Silva, http://sweet.ua.pt/tos/goldbach.html
[5] T. Granlund. The GNU Multiple Precision Arithmetic Library. TMG Datakonsult, Boston, MA, USA, 2.0.2 edition, June 1996.
[6] J. Hurd, Verification of the Miller–Rabin probabilistic primality test, The Journal of Logic and Algebraic Programming 56.1 (2003): 3-21.
[7] Z. Zhang, Two Kinds of Strong Pseudoprimes up to 10^(36), Math. Comput. 76, 2095-2107, 2007.
[8] T.Z. Wang, J.R. Chen, On odd Goldbach problem under general Riemann hypothesis ,Sci. China Ser. A 36 (1993), no. 6, 682-691. MR 95a:11090.
[9] J. Franke, et al., Proving the Primality of Very Large Numbers with fastECPP, Algorithmic Number Theory. Springer Berlin Heidelberg, 2004. 194-207.
[10] A. Prástaro, The Goldbach's conjecture proved, *arXiv preprint arXiv:1208.2473* (2012).
[11] H. A. Helfgott, Major arcs for Goldbach's theorem, *arXiv preprint arXiv:1305.2897* (2013).
[12] W. D. Gropp, E. Lusk and A. Skjellum, *Using MPI: portable parallel programming with the message-passing interface*. Vol. 1. the MIT Press 1999.