Algorytmy - projekt drugi

Dominik Karmoliński
Piotr Karolak

Projekt polega na ukazaniu różnic w działaniu dwóch algorytmów, gdzie stosunkowo niewielkie ilości dodanego kodu przekładają się na wielokrotnie skuteczniejsze przeliczenia. Sprawdzamy jak duży skok wydajności można osiągnąć dzięki dobremu dobraniu algorytmu. Do demonstracji posłużą nam wyselekcjonowane liczby pierwsze, które zostaną potwierdzone jako takowe przez algorytmy sprawdzające i następnie je porównamy.

Używane narzędzia:

- Microsoft Visual Studio 2019
- Google Sheets
- Microsoft Excel
- Google Docs
- Notatnik
- Procesor Intel Core i5-6400 2.70 GHz

1. Algorytm bazowy vs lepszy algorytm - wyniki

Algorytm 1		Algorytm 2	
Operacje 1	Czas 1	Operacje 2	Czas 2
25227	29113	316	3118
252283	227658	1003	590
2522849	2409843	3175	1992
25228514	23752069	10044	4857
252285151	217744566	31765	12590
2522851532	5142795885	100452	32039
25228515333	22549595765	317661	78556
252285153349	264987464863	1004531	206718

Tabela 1. Dane w liczbach - operacje i czas obu algorytmów - dwie ostatnie liczby wyestymowane.

Dominik Karmoliński
Piotr Karolak

Za nasze próbne liczby obraliśmy wskazaną w poleceniu grupę testową:

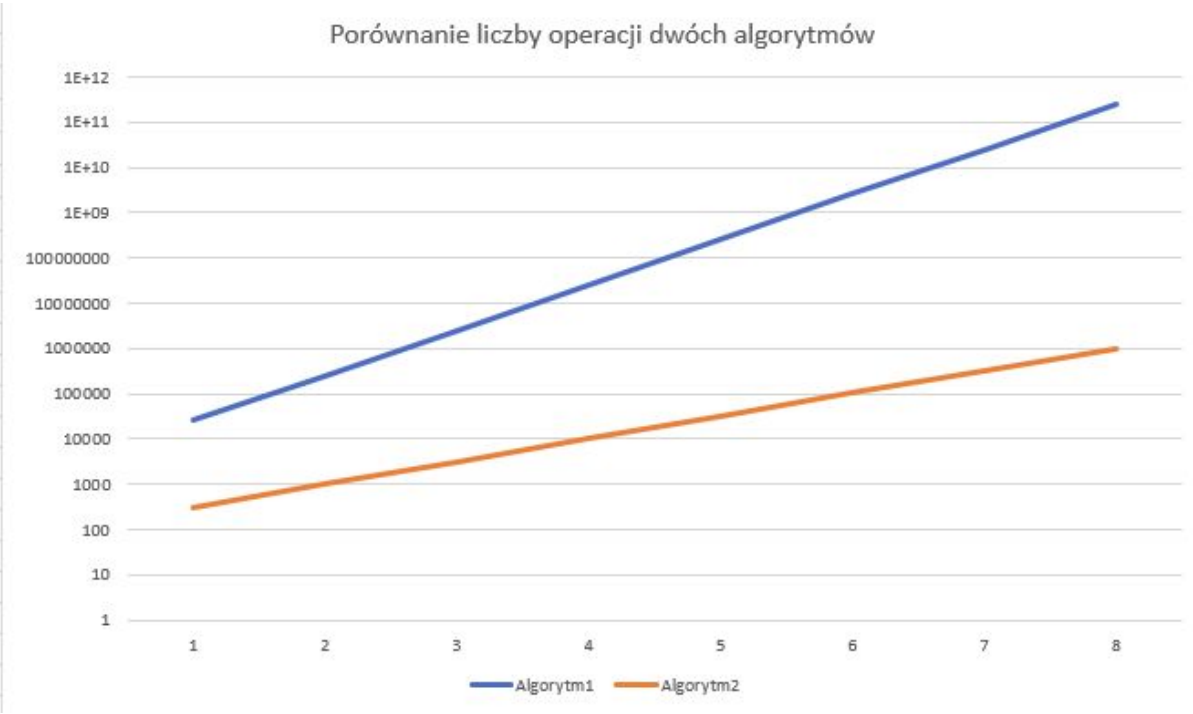
100913
1009139
10091401
100914061
1009140611
10091406133
100914061337
1009140613399

Tabela 2. Liczby testowe - potwierdzone liczby pierwsze.

2. Wyszukiwanie binarne - operacje

Dla wskazanych liczb uruchomiliśmy oba porównywane algorytmy, w dwóch wersjach - zliczającej operacje oraz zliczającej czas wykonania w tickach procesora.

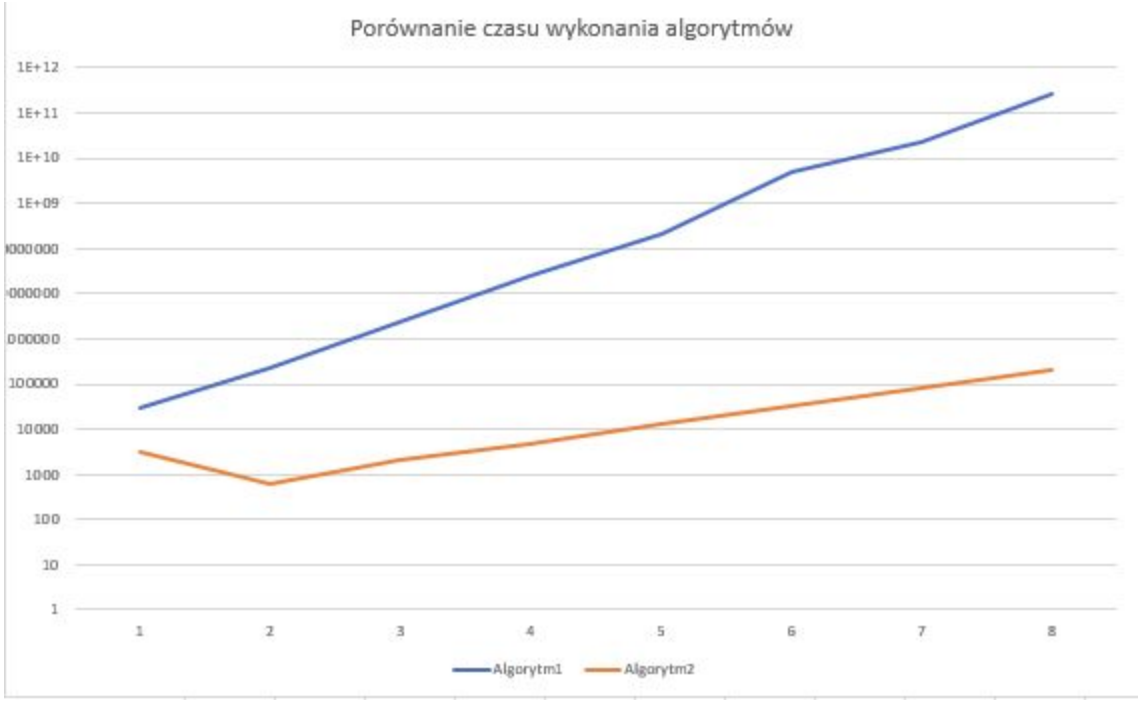
Poniżej prezentujemy wykres porównania obu algorytmów w operacjach.



Wykres 1. Porównanie liczby operacji dla Algorytmu 1 (słabszego) oraz Algorytmu 2 (lepszego) dla wskazanych ośmiu liczb. Dwa ostatnie wyniki oszacowano. Skala logarytmiczna.

Dominik Karmoliński
Piotr Karolak

Drugi wykres prezentuje analogiczne porównanie czasu wykonania.



Wykres 2. Porównanie czasu trwania Algorytmu 1 (słabszego) oraz Algorytmu 2 (lepszego) dla poszczególnych liczb. Dwa ostatnie wyniki oszacowano. Skala logarytmiczna.

Wahania czasu są stosunkowo znaczące dla lewej części wykresu, jednak całokształt skutecznie prezentuje tendencję całościową.

Algorytmy - projekt drugi

Dominik Karmoliński
Piotr Karolak

Podsumowanie

W przypadku naszych algorytmów, zauważyliśmy pewne konkretne proporcje przy kolejnych wzrostach liczb - dla Algorytmu 1 każda kolejna liczba daje mniej więcej czterokrotny wzrost liczby operacji oraz niemal regularny czterokrotny wzrost czasu; dla Algorytmu 2 wzrost nie jest liniowy, ale widać zależność od sprawdzanej liczby, za to czas jest niemal dokładnie czterokrotnością czasu poprzedniej liczby.

Niewątpliwie drugi, ulepszony algorytm jest lepszą opcją co do wyszukiwania liczb pierwszych tak pod względem operacji, jak i czasu wykonania w każdym badanym przypadku. Jednak przy liniowym, regularnym wzroście łatwiej przewidzieć czas działania pierwszego algorytmu.

Warto zwrócić uwagę na to, że nie staraliśmy się dobrać maksymalnie optymalnego algorytmu do zadania, a wyniki i tak są kolosalnie lepsze w stosunku do przykładowego algorytmu. Gdyby spędzić nad tym więcej czasu, niewątpliwie można osiągnąć jeszcze lepsze wyniki.

Przy wielkiej skali obliczania danych prawidłowy dobór algorytmu wiąże się ze znaczącymi oszczędnościami tak w formie czasu, jak i eksploatowania posiadanej mocy obliczeniowej. Nawet przy stosunkowo prostych operacjach na małych bazach danych czy tablicach podobnych do naszego przykładu obniżamy czas wykonania o godziny w stosunku do gorszego algorytmu, co w skali całego cyklu życia przeciętnego PC czy wykupionej chmury może przekładać się na tysiące złotych oszczędności w celu osiągnięcia analogicznych wyników kosztem inwestowania w lepszy sprzęt.

Czasami kompensacja większą mocą obliczeniową jest niepraktyczna, w szczególności gdy mowa o urządzeniach mobilnych czy wszelakich komponentach “internetu rzeczy”. Jedynym realnym sposobem podniesienia efektywności takich urządzeń (na krótką metę) jest dbałość o tworzenie skutecznego kodu pozbawionego niepotrzebnych elementów oraz oszczędzającego maksymalnie dużo mocy obliczeniowej, a algorytmy umieszczone w kodzie są kluczową częścią tego procesu.

Istotnym wnioskiem końcowym naszej analizy jest to, że dobry programista w rzeczywistych warunkach pracy w realnym świecie jest w stanie oszczędzić bardzo znaczące sumy dla zatrudniających go firm nawet przy tak prostych aspektach jak dobór prawidłowego algorytmu do odpowiedniego zadania.