AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Inżynierii Metali i Informatyki Przemysłowej

PROJEKT MAGISTERSKI

pt.

„Implementacja solwera frontalnego zrównoleglonego w wielowęzłowym heterogenicznym środowisku sprzętowym”

Imię i nazwisko dyplomanta:	**Paweł Wal**
Kierunek studiów:	**Informatyka Stosowana**
Nr albumu:	**240202**
Opiekun:	dr inż. Łukasz Rauch

Podpis dyplomanta: Podpis opiekuna:

Kraków 2015

Oświadczam, świadomy odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszy projekt magisterski wykonałem osobiście i samodzielnie i że nie korzystałem ze źródeł innych niż wymienione w pracy.

Kraków, dnia . . . . . Podpis dyplomanta . . . . . .

Spis treści

1 WPROWADZENIE

Ilość dostępnych zasobów - takich jak moc obliczeniowa, pamięć dyskowa i operacyjna oraz przepustowość łączy - stale się zwiększa. Dzięki temu ciągłemu wzrostowi możliwości problemy które wydawały się zbyt duże by dało się je rozwiązać w rozsądnym czasie stają się możliwe do rozwiązania. Naturalnym jest jednak, że kiedy nierozwiązalne problemy stają się banalne, pojawiają się nowe problemy - jeszcze bardziej złożone, wynikające z próby odpowiedzi na jeszcze bardziej ambitne pytania.

Rozsądek nakazuje sądzić, iż wzrost wydajności pojedynczych urządzeń nie może trwać w nieskończoność. Prędzej czy później wzrost złożoności urządzeń zostanie zahamowany, choćby przez wzgląd na niemożność pomieszczenia większej ilości tranzystorów na pojedynczej płytce.

W przypadku CPU zahamowanie tego procesu nie daje się jeszcze zauważyć. W roku 2014 firma Intel wypuściła procesory Broadwell oparte o proces technologiczny 14nm [8]. Na początku roku 2015 firma Taiwan Semiconductors zapowiedziała ambitne plany wprowadzenia na rynek procesorów w technologii 10nm w roku 2016 i 7nm w 2017 [12].

W przypadku GPU producenci mają jednak większy problem z miniaturyzacją elementów składowych swoich urządzeń. Wystarczy spojrzeć na to, jaką miniaturyzację planowała przeprowadzić NVIDIA. W 2009 roku na temat planów firmy w tej kwestii mówił na 46th Design Automation Conference William J Dally, wówczas szef działu naukowego w NVIDIA [11]. Jego ambitne projekcje przewidywały, iż do 2015 roku NVidia wprowadzi na rynek urządzenia wykonane w technologii 11nm, zawierające 5,000 mikrordzeni obliczeniowych i taktowane zegarami o częstotliwości 3 GHz. Specyfikacja urządzenia NVIDIA GeForce GTX TITAN X [10], jednej z najnowszych i najwydajniejszych kart graficznych wyprodukowanych przez firmę, dowodzi iż projekcje Dally'ego były mocno przesadzone: urządzenie wykonane jest w technologii 28nm, ma 3072 mikrordzenie i jest taktowane zegarem o częstotliwości 1000 MHz (1089 MHz w trybie Boost).

Alternatywą dla stworzenia urządzeń o większej wydajności jest wykorzystanie większej ilości urządzeń obliczeniowych w jednej maszynie. Jednakże części pozwalające na zamontowanie więcej niż jednego GPU w danej obudowie, takie jak płyty główne, wciąż jeszcze są urządzeniami specjalnego zastosowania a nie standardem. Wraz z tym jak sprzęt jest specyficzny rośnie oczywiście cena. Dobrym przykładem jest tutaj płyta główna ASUS Rampage IV Black Edition, pozwalająca na zamontowanie aż 4 GPU; w trakcie pisania niniejszej pracy jej cena wynosi ponad 2000 złotych.

Oczywistym rozwiązaniem w tej sytuacji wydaje się przetwarzanie rozproszone. Dzięki jego zastosowaniu można uniknąć problemu coraz wolniej zwiększającej się wydajności pojedynczych urządzeń, gdyż do rozwiązania problemu można wykorzystać więcej urządzeń o mniejszej jednostkowej wydajności. Dzięki połączeniu odrębnych fizycznie maszyn w abstrakcyjny „klaster" nie ma konieczności umieszczania wszystkich posiadanych urządzeń obliczeniowych w jednej maszynie. Maszyny obliczeniowe nie muszą nawet być w jednej lokalizacji dzięki połączeniu ich za pośrednictwem sieci Internet. Warto zauważyć, iż przetwarzanie rozproszone towarzyszy idei Internetu od samego jej zarania: możliwość stworzenia ogólno-

krajowej sieci przetwarzania danych była jednym z argumentów za budową sieci ARPANET [4].

Należy zdać sobie sprawę, iż podstawowa siła w przetwarzaniu rozproszonym jest możliwość zwiększania ilości urządzeń pracujących nad danym problemem (bądź zestawem problemów). W konsekwencji - zamiast wykorzystywać nieliczne urządzenia o bardzo wysokiej wydajności - i co za tym zazwyczaj idzie, cenie - można wykorzystać znacznie więcej urządzeń klasy konsumenckiej, tańszych i o mniejszej jednostkowej wydajności.

Oczywistym kierunkiem jest też skupienie się na architekturach heterogenicznych. Kompozycje składające się z urządzeń różnego typu, takich jak CPU i GPU, nie tylko osiągają coraz wyższą wydajność (drugie miejsce na liście TOP500 z listopada 2014 [9] zajmuje heterogeniczny superkomputer Titan), ale też stają się coraz popularniejsze wśród konsumentów. W istocie większość typowych komputerów osobistych i biurowych to maszyny heterogeniczne; trudno patrzyć na to inaczej w sytuacji gdy niemal każda karta graficzna nadaje się również do celów obliczeniowych.

W przypadku komputerów osobistych szczególnie często występuje zjawisko marnowania cykli. Z jednej strony karta graficzna jest zasilana. Z drugiej strony, wyświetlanie przeglądarki internetowej czy pakietu biurowego nie wykorzystuje pełni jej możliwości. W sytuacji gdzie efektywność energetyczna urządzeń zaczyna być coraz istotniejsza (czego dowodem może być powstawanie takich list jak GREEN500, klasyfikujących urządzenia według wskaźnika FLOP/wat), takie marnowanie cykli jest trudne do przyjęcia.

Między innymi w tym należy zapewne dopatrywać się popularności platform przetwarzania rozproszonego przy użyciu komputerów wolontariuszy. Platformy takie szerzej omawiane są w następnych rozdziałach. Charakteryzuje je nastawienie na heterogeniczny sprzęt oraz odporność na dużą rotację hostów. Trudno przecenić ich funkcje: wykorzystują zmarnowany potencjał urządzeń klasy konsumenckiej i zarazem służą popularyzacji konkretnych projektów naukowych.

Jednakże nie do każdego rodzaju zastosowań takie platformy się nadają. Trudno wyobrazić sobie firmę z branży przemysłowej która prowadzi badania nad swoimi produktami w publicznie otwartym środowisku. Ze względu na tą otwartość publiczne projekty przetwarzania rozproszonego muszą rozwiązywać też takie problemy jak złośliwi użytkownicy próbujący „zatruć” bazy danych. Ponadto by wykorzystać większość z takich platform należy utrzymywać własną infrastrukturę i stworzyć komplet oprogramowania działającego wewnątrz platformy, co oczywiście angażuje zasoby - tyleż ludzkie co finansowe.

Na przeciwnym biegunie w stosunku do przetwarzania na komputerach wolontariuszy stoją architektury gridowe. Zazwyczaj jest to własność instytucji, takich jak duże firmy czy uczelnie. Instytucje te zapewniają zasoby niezbędne, by grid mógł funkcjonować, oraz konieczną obsługę. Wykorzystanie zasobów gridowych należących do instytucji pozwala pozbyć się wielu problemów które dotyczą platform opartych na wolontariacie (nie do pomyślenia jest na przykład sytuacja zatruwania wyników badań). Nie ma też konieczności utrzymywania infrastruktury. Nadal jednak pozostaje problem stworzenia oprogramowania działającego w ramach

architektury gridowej. Nie rozwiązuje to również sytuacji w której instytucja chcąca prowadzić badania nie posiada własnej infrastruktury, a umieszczenie danych dotyczących badań na komputerach będących pod kontrolą innego podmiotu nie wchodzi w grę.

W niniejszej pracy zostanie zaproponowane rozwiązanie pośrednie między tymi dwoma kierunkami. Z jednej strony stworzone w podobnym duchu co oprogramowanie do przetwarzania na zasobach wolontariuszy: odporne na rotację i awarie hostów, nie przyjmujące zbędnych założeń w stosunku do hostów, ich mocy obliczeniowej, położenia geograficznego czy dostępności w ciągu doby. Z drugiej strony jest to rozwiązanie nakierowane bardziej na rozwiązanie problemu przetwarzania dużych zadań w małych i średnich organizacjach, nie posiadających własnych architektur gridowych. Jego instalacja, utrzymanie i wykorzystanie mają być możliwie uproszczone, tak by nie było konieczne tworzenie własnego oprogramowania bądź korzystanie z usług specjalistów. Rozwiązanie ponadto ma się dobrze skalować, tak by wykorzystanie wszystkich dostępnych w danym momencie urządzeń obliczeniowych dawało realne przełożenie na czas uzyskania wyniku.

W dalszych sekcjach niniejszej pracy przeprowadzona zostanie analiza dostępnych rozwiązań, w tym ich mocnych oraz słabych stron z punktu widzenia rozpatrywanego przypadku. Zostaną również przedstawione założenia projektowe rozwiązania własnego, jego implementacja oraz wyniki przeprowadzonych na stworzonym oprogramowaniu badań wydajności.

2 PRZEGLĄD ISTNIEJĄCYCH ROZWIĄZAŃ

2.1 ENTROPIA

Projekt Entropia [3] był komercyjnym rozwiązaniem, tworzonym w Entropia, Inc. od 1997 roku. Głównym celem tego oprogramowania było zapewnienie możliwości tworzenia wewnętrznych, organizacyjnych architektur gridowych.

Entropia pozwalała na wykorzystywanie wyłącznie maszyn działających pod kontrolą systemu Windows. W momencie tworzenia tego oprogramowania przetwarzanie rozproszone było stosowane raczej w środowiskach akademickich, na maszynach działających pod kontrolą systemów z rodziny *NIX, na co Entropia miała być odpowiedzią.

Entropia, Inc. zakończyła działanie komercyjne w 2004 roku bez oficjalnego ogłoszenia tego faktu, ani powodu podjęcia takiej decyzji.

2.1.1 Architektura

Centralnym elementem sieci w Entropii jest serwer, pełniący kilka funkcji. Serwer służy zarówno do komunikacji z systemem użytkownika zlecającego zadanie, jak i do porozumiewania się z maszynami klienckimi.

Pierwszym etapem komunikacji użytkownika z systemem jest stworzenie zadania. Zadanie zawiera aplikację która ma być uruchamiana na komputerach klienckich, zbiór wejść dla aplikacji oraz innego rodzaju danych. Po otrzymaniu od użytkownika zadania, mechanizm zarządzania zadaniami w serwerze dzieli je na mniejsze podzadania oraz uruchamia narzędzia

przetwarzania wstępnego jeżeli użytkownik takie zdefiniował.

Te podzadania są centralnie składowane i przekazywane do mechanizmu harmonogramowania zadań. Oprócz listy podzadań element ten utrzymuje też listę urządzeń na których może uruchamiać zadania, wraz ze szczegółowymi informacjami o urządzeniu. W skład tych informacji wchodzi dostępność urządzenia oraz statyczne cechy urządzenia - ilość pamięci czy typ systemu operacyjnego. Dane te są wykorzystywane do efektywnego rozdzielania zadań pomiędzy aktualnie dostępnych klientów.

Mechanizm harmonogramowania odpowiada także za to, by zadania które długo nie zostały przekazane do przetwarzania otrzymały wyższy priorytet, co przesuwa je w górę kolejki zadań do wysłania. Dba również o ponowne uruchamianie zadań których wykonanie nie udało się i ponowne zlecenie zadań których wyniki zbyt długo nie wróciły z klienta. Jeżeli wykonanie danego podzadania konsekwentnie kończy się błędem, zostaje ono oznaczone jako błędne, o czym informowany jest użytkownik który zlecił zadanie.

Po zakończeniu wszystkich podzadań w danym zadaniu serwer wykonuje na nich etap przetwarzania końcowego, ponownie przy użyciu narzędzi zdefiniowanych przez użytkownika systemu. Po jego zakończeniu wyniki udostępniane są użytkownikowi. Może on pobrać albo zagregowane wyniki dla całego zadania, lub wyniki zakończenia poszczególnych podzadań.

2.1.2 Oprogramowanie klienckie

W ramach Entropii dostarczane było tylko oprogramowanie dla systemów operacyjnych z rodziny Windows - zarówno serwerowe, jak i klienckie.

W założeniu oprogramowanie klienckie Entropii miało być uruchamiane na komputerach osobistych klasy konsumenckiej, jednocześnie służących do innych zadań. Przykładem typowego zastosowania mogłaby więc być sytuacja, w której klient Entropii jest uruchomiony na wszystkich komputerach należących do jednej firmy - niezależnie od tego, czy używają ich pracownicy, czy nie.

Entropia wstrzymuje przetwarzanie zadań na pięć minut jeśli wykryje aktywność użytkownika na komputerze (obserwuje ją monitorując myszkę i klawiaturę). Autorzy oprogramowania nie zdecydowali się na wymaganie od aplikacji uruchamianych wewnątrz Entropii możliwości wstrzymania obliczeń. Po pierwsze, zawężałoby to możliwość zastosowania Entropii do oprogramowania w którym istnieje możliwość ingerencji w kod źródłowy. Po drugie, aplikacje które - celowo lub przypadkiem - nie wstrzymałyby działania po otrzymaniu stosownego sygnału zaburzałyby działanie maszyny klienckiej z punktu widzenia jej użytkownika.

Problem ten - oraz niektóre inne problemy związane ze źle napisanymi aplikacjami - został przez autorów Entropii rozwiązany przy pomocy wirtualizacji. Każde zadanie jest uruchamiane wewnątrz „piaskownicy”. Działanie tejże jest wstrzymywane kiedy wykryta zostanie aktywność użytkownika. W ten sposób aplikacja nie może też uzyskać dostępu do większej ilości zasobów niż pozwoli na to środowisko wirtualizacyjne. Przechwycone mogą też być odwołania do systemowych API służących do zapisu plików; aplikacja może uważać, że zapisuje pliki w folderze `C:\Program Files`, podczas gdy w istocie zapisuje je w folderze umieszczo-

nym gdzieś wewnątrz struktury katalogów klienta Entropii.

Warto zwrócić uwagę, iż o ile jest to rozwiązanie podnoszące bezpieczeństwo całego układu - przy założeniu dużej dowolności uruchamianych wewnątrz Entropii aplikacji - jak każda wirtualizacja dodaje ono własny narzut obliczeniowy i pamięciowy. Jest to efekt niepożądany z punktu widzenia minimalizacji czasu do rozwiązania. Trudno również wyobrazić sobie skuteczną wirtualizację oprogramowania korzystającego z kart graficznych.

2.1.3 Great Internet Mersenne Prime Search

Oryginalny projekt GIMPS, stworzony w 1996 przez George'a Woltmana, wykorzystywał proces manualny. Konieczne było wysłanie wiadomości e-mail pod wskazany adres by uzyskać pakiet wejść dla programu, pobranie programu, uruchomienie go i odesłanie - również e-mailem - wyników. Wraz ze wzrostem ilości autoignition złożoność i brak odporności na błędy tego procesu stały się niedostateczne. Mimo tego, jeszcze w 1996 roku odało się odkryć trzydziestą piątą liczbę Mersenne'a - $2^{1,398,269} - 1$.

Scott Kurowski - założyciel Entropia, Inc. - stworzył w oparciu o oprogramowanie tworzone w swojej firmie PrimeNet, co pozwoliło zautomatyzować wszystkie zadania dotyczące rozsyłania zadań i odbierania ich wyników. Umożliwiło to łatwiejsze zarządzanie siecią tysięcy wolontariuszy oraz bazą milionów zadań. Mimo zamknięcia Entropia, Inc., PrimeNet działa nadal; ostatnie odkrycie w ramach projektu nastąpiło 25 stycznia 2013, kiedy to zidentyfikowano 48 liczbę Mersenne'a [6].

2.2 SETI@HOME i BOINC

SETI@home zostało oryginalnie stworzone w celu przetwarzania danych z projektu SERENDIP [5]. W ramach tego projektu zbierane były sygnały radiowe odbierane przez radioteleskop w Arecibo, w Puerto Rico. Następnie w tych danych wyszukiwane były charakterystyczne struktury, których pojawienie się mogło świadczyć o odebraniu sygnału radiowego powstałego sztucznie - potencjalnie pochodzącego od pozaziemskiej cywilizacji. Stąd pierwszy człon nazwy projektu - SETI, czyli Search For Extraterrestial Intelligence.

Z punktu widzenia przetwarzania rozproszonego bardziej interesujący jest drugi człon nazwy - @home. W ramach eksperymentu SETI@home po raz pierwszy zostało stworzone oprogramowanie dzięki któremu posiadacze zwykłych domowych komputerów i połączeń z internetem mogli uczestniczyć - a przynajmniej w pewnym sensie kontrybuować - w dużym projekcie naukowym. Oczywistą zachętą dla uczestników - wolontariuszy, pozwalających oprogramowaniu pracować kiedy nie korzystają z komputera - była minimalna szansa, że to właśnie ich komputery mogą znaleźć dowód istnienia technologicznie zaawansowanej cywilizacji pozaziemskiej.

Projekt SETI@home został stworzony do rozwiązania problemu którego nie dało się wówczas rozwiązać przy użyciu dostępnych urządzeń do obliczeń wysokiej wydajności; kłócić się można, że nadal nie istnieje możliwość przeprowadzenia tak dokładnych badań na takiej ilości danych w rozsądnym czasie nie korzystając z przetwarzania rozproszonego. Dane radiowe

spływały bowiem w tempie 257 GB na tydzień.

2.2.1 Obróbka danych wejściowych

Architektura projektu była stosunkowo skomplikowana, a jego utrzymanie wymagało serii skomplikowanych operacji. Przede wszystkim, kiedy projekt zaczynał pracę - w 1997 roku - nie istniała techniczna możliwość przesłania takiej ilości danych przez internet z Puerto Rico do USA. W związku z tym dane były przesyłane w formie taśm magnetyczych, przesyłką kurierską.

Na miejscu taśmy oczywiście należało zgrać przy użyciu odpowiednich urządzeń, oraz dodać do zapisanych na nich informacji metadane - takie jak punkt na niebie z którego zostały zebrane czy czas kiedy były nagrane. Był to czasochłonny, w dużej mierze ręczny proces wymagający odpowiedniej wiedzy technicznej od osób odpowiedzialnych za badania.

2.2.2 Tworzenie i wysyłka zadań

Dane z teleskopu opatrzone metadanymi były następnie dzielone na niewielkie paczki - o rozmiarze 250 kB. Tak mały rozmiar był konieczny po pierwsze dlatego, by czas pobierania na połączeniu modemowym był akceptowalny (według wyliczeń autorów projektu, na modemie o przepustowości 14400 baudów zajmowało to 2.3 minuty), a po drugie by czas przetwarzania zadania na komputerze klasy konsumenckiej był do zaakceptowania przez obie strony. Na typowych procesorach w 1997 roku było to zazwyczaj kilka dni.

Warto zauważyć, że ze względu na publiczną naturę projektu, dane do przetwarzania musiały być redundantne, to znaczy ten sam pakiet danych musiał być przetworzony przez dwie lub więcej niezależnych maszyn. Pozwalało to uniknąć przypadkowych przekłamań w wynikach (powstałych na przykład w skutek niepoprawnego działania komputera wolontariusza) lub celowego zatruwania danych przez złośliwych użytkowników. Kopie danego zadania były więc przetwarzane aż do uzyskania kworum wyników zgadzających się ze sobą (z dopuszczalnymi pewnymi różnicami wynikającymi na przykład z różnych architektur procesorów w maszynach).

Wynikiem przetwarzania danego zadania była krótka wiadomość zawierająca położenie i częstotliwości sygnałów-kandydatów w danym wycinku danych radiowych. Informacja taka była prezentowana użytkownikowi, oraz wysyłana z powrotem do serwera, gdzie oczekiwała na kworum. Ostatecznie rezultat był zapisywany w bazie danych do późniejszych dokładniejszych badań.

2.2.3 Podsumowanie

W przeciwieństwie do oprogramowania Entropia, nie został wykorzystany żaden rodzaj wirtualizacji ani zabezpieczania systemu operacyjnego klienta przed działaniami klienta SETI@home. Jest to zrozumiałe - ostatecznie odpowiedzialni za jego stworzenie byli też odpowiedzialni za jego dystrybucję. Można zatem uznać, że mieli kontrolę nad jego funkcjonowaniem i mogli zapewnić wolontariuszy o tym, iż jego uruchomienie jest bezpieczne.

Warto zauważyć, że dla wolontariusza wykorzystanie klienta było maksymalnie uproszczone. Wystarczyło, że pobierze on pakiet oprogramowania odpowiedni dla jego systemu operacyjnego i go uruchomi. SETI@home automatycznie wykrywało momenty kiedy użytkownik nie korzystał z komputera i uruchamiało swoje przetwarzanie; wolontariusz więc nie miał ani potrzeby, ani możliwości ingerencji w jego działanie. Z drugiej strony, wytworzenie zadań oraz zarządzanie serwerową częścią projektu było trudnym zadaniem, wymagającym głębokiej technicznej wiedzy. Równie trudna była interpretacja wyników pochodzących ze zautomatyzowanego oprogramowania.

2.2.4 BOINC

2.3 XTREMWEB

3 PROJEKT ROZWIĄZANIA WŁASNEGO

3.1 PODZIAŁ PROBLEMU NA PODPROBLEMY

Dzielenie na części celem składowania.

Umieszczanie ich w jednym katalogu/repozytorium (na serwerze).

3.2 ROZWIĄZANIE PODPROBLEMÓW

Proces przez który przechodzi każde zadanie: wydzielenie, zapis, przesył, rozwiązanie, przesył.

3.3 ZEBRANIE WYNIKÓW

Rozwiązanie zależności między falami. Jeśli zaszła konieczność ich rozwiązania, stworzenie kolejnej fali rozwiązania.

3.4 ROZWIĄZANIE UKŁADU RÓWNAŃ

4 IMPLEMENTACJA

4.1 WYBÓR NARZĘDZI

Tu wstaw dlaczego Railsy. Dlaczego Ruby. Dlaczego C++. Dlaczego OpenCL.

4.2 PODZIAŁ MACIERZY

Tu wstaw boleśnie dokładny opis splittera.

4.3 SERWER ZADAŃ

4.3.1 Struktura bazy danych

4.3.2 Umieszczenie zadania w serwerze

4.3.3 Autoryzacja klienta

4.4 KLIENT

4.4.1 Instancje oprogramowania obliczeniowego

4.4.2 Negocjacja ilości przydzielonych zadań

Na pewno mechanizm z ilością urządzeń i max GWS; może benchmark

4.4.3 Rozwiązanie zadań

4.4.4 Odesłanie rozwiązanych zadań

4.5 OPROGRAMOWANIE OBLICZENIOWE

4.5.1 Solwer po stronie klienta

4.5.2 Negocjator zależności po stronie serwera

5 WYNIKI

5.1 CHARAKTERYSTYKA MASZYN

Skalowalność rozwiązania w aspekcie zmiennej ilości urządzeń obliczeniowych.

6 PODSUMOWANIE

7 BIBLIOGRAFIA

[1] Anderson, D.P.: BOINC: a system for public-resource computing and storage. *Grid Computing. Proceedings. Fifth IEEE/ACM International Workshop on*. 2004

[2] Anderson, D.P., Cobb, J., Korpela, E., Lebofsky, M., Werthimer, D.: SETI@home: An Experiment in Public-Resource Computing. *Communications of the ACM*. 2002 Vol. 45, No. 11, pp. 56-61

[3] Chien, A., Calder, B., Elbert, S., Bhatia, K.: Entropia: architecture and performance of an enterprise desktop grid system. *Journal of Parallel and Distributed Computing*. 2003 Vol. 63, pp. 597–610

[4] Heart, F., McKenzie, A., McQuillian, J., Walden, D.: ARPANET completion report, Technical Report 4799. *BBN*. 1978

[5] Sullivan, III, W. T., Werthimer, D., Bowyer, S., Cobb, J., Gedye, D., Anderson, D.: A new major SETI project based on Project Serendip data and 100,000 personal computers. *Astronomical and Biochemical Origins and the Search for Life in the Universe, IAU Colloquium 161*. 1997 p. 729

[6] GIMPS Discovers 48th Mersenne Prime, $2^{57,885,161} - 1$ is now the Largest Known Prime [online]. *Great Internet Mersenne Prime Search*. [dostęp: 2015-06-01], Dostępny w Internecie: <http://www.mersenne.org/primes/?press=M57885161>

[7] GIMPS History - PrimeNet [online]. *Great Internet Mersenne Prime Search*. [dostęp: 2015-06-01], Dostępny w Internecie: <http://www.mersenne.org/various/history.php>

[8] Intel launches first Broadwell processors [online]. *CPU World*. [dostęp: 2015-06-01], Dostępny w Internecie: <http://www.cpu-world.com/news_2014/2014090701_Intel_launches_first_Broadwell_processors.html>

[9] November 2014 [online]. *TOP500 Supercomputer Sites*. [dostęp: 2015-06-01], Dostępny w Internecie: <http://www.top500.org/lists/2014/11/>

[10] NVIDIA GeForce GTX TITAN X [online]. *TechPowerUp*. [dostep: 2015-06-01], Dostępny w Internecie: <http://www.techpowerup.com/gpudb/2632/geforce-gtx-titan-x.html>

[11] Nvidia SVP Predicts Rapid GPU Performance Growth [online]. *Nikkei Technology*. [dostep: 2015-06-01], Dostępny w Internecie: <http://techon.nikkeibp.co.jp/english/NEWS_EN/20090731/173702/>

[12] TSMC Promises 10nm Production In 2016, 7nm in 2017 [online]. *WCCF Tech*. [dostęp: 2015-06-01], Dostępny w Internecie: <http://wccftech.com/tsmc-promises-10nm-production-2016-7nm-2017/>