

JĘZYKI I NARZĘDZIA PROGRAMOWANIA II

CUDA

Witold Rudnicki

ZASADY

- 20% - 30% czasu wykłady
- 70% - 80% czasu laboratorium
- Zaliczenie
 - projekt końcowy
 - zadania bieżące

ZASADY

- Zadania i Projekt końcowy - praca samodzielna!
 - TAK - dyskusje, pytania,
 - NIE - praca zespołowa, ^C^V, przepisywanie cudzego kodu
 - Plagiat = OUT

ZASADY

- Punktacja i terminy
 - Zadania bieżące:
 - Zadania oddajemy przed czwartkiem poprzedzającym kolejne zajęcia (do 23:59 w środę).
 - Punktacja
 - 10 punktów bazowych za oddane w terminie zadanie poprawne.
 - Odejmujemy jeden punkt za każdy przekroczony termin.
 - Dodajemy/odejmujemy punkty za efektywność - logarytm dwójkowy (zaokrąglony w górę) ze wzrostu wydajności w stosunku do rozwiązania referencyjnego

ZASADY

- Punktacja i terminy
 - Projekt zaliczeniowy - oddajemy tydzień przed ostatnimi zajęciami (środa 23:59 w tygodniu przed ostatnimi zajęciami)
 - Punktacja
 - 100 punktów bazowych za oddane w terminie zadanie poprawne.
 - Odejmujemy jeden punkt za każdą godzinę przekroczonego terminu (pierwsza doba) i pięć za każdy kolejny dzień.
 - Dodajemy/odejmujemy punkty za efektywność - 5* logarytm dwójkowy (zaokrąglony w górę) ze wzrostu wydajności w stosunku do rozwiązania referencyjnego

ZASADY

- Warunkiem zaliczenia jest
 - a) dostarczenie poprawnych rozwiązań 10 zadań bieżących
 - b) dostarczenie poprawnego rozwiązania zadania końcowego
- Ocena
 - min 100 punktów - dobry
 - 120 punktów i więcej - bardzo dobry
 - min. 60 punktów - zaliczone

MATERIAŁY

NVIDIA CUDA C Programming Guide Version 5.0

CUDA By Example - wybrane rozdziały (dostarczę)

GPU Computing Gems part I. - wybrane rozdziały (dostarczę)

Slajdy z wykładu - strona w przygotowaniu

TOP 500

- Lata 1980-1990 – procesory wektorowe (Cray)
- Lata 1990-2000 – maszyny masywnie równolegle RISC (IBM SGI)
- Lata 2000-2010 – klastry Intel (IBM HP)
- Lata 2010- Klastry i komputery hybrydowe

TOP 500

Rok	#	Wydajność teoretyczna (GFLOPS)	Dostawca	Suma wydajności (GFLOPS)
1993	Thinking Machines 1024 CPU SuperSparc	60	Cray 205 maszyn	399
1996	Hitachi SR2xxx 1024 CPU PA RISC,	232	Cray 104 maszyny	1388
1999	Intel Paragon ASCI Red 9472 CPU IA-32	2 121	Cray 78 maszyn	12300
2002	NEC Earth Simulator 5120 CPU NEC,	36 000	IBM 164 maszyny	74 000
2005	IBM BlueGene 65536 CPU PowerPC	136 000	IBM 259 maszyn	985 000
2008	IBM Roadrunner Red 9472 6480 + 12960 CPU AMD Opteron + PowerXCell 8i	1 002 000	IBM 209 maszyn	5 600 000
2012	Titan - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x	17 590 000	IBM 193 maszyny	162 000 000

TOP 500

TOP 10 Sites for November 2015

For more information about the sites and systems in the list, click on the links or view the [complete list](#).

RANK	SITE	SYSTEM	CORES	RMAX (TFLOP/S)	RPEAK (TFLOP/S)	POWER (KW)
1	National Super Computer Center in Guangzhou China	Tianhe-2 (MilkyWay-2) - TH-IVB-EP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express 2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
2	DOE/SC/Oak Ridge National Laboratory United States	Titan - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
3	DOE/NNSA/LLNL United States	Sequoia - BlueGene/Q, Power BQC 16C 1.60 GHz Custom IBM	1,572,864	17,173.2	20,132.7	7,890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
5	DOE/SC/Argonne National Laboratory United States	Mira - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
6	DOE/NNSA/LANL/SNL United States	Trinity - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	301,056	8,100.9	11,078.9	
7	Swiss National Supercomputing Centre (CSCS) Switzerland	Piz Daint - Cray XC30, Xeon E5-2670 8C 2.600GHz, Aries interconnect , NVIDIA K20x Cray Inc.	115,984	6,271.0	7,788.9	2,325
8	HLRS - Höchstleistungsrechenzentrum Stuttgart Germany	Hazel Hen - Cray XC40, Xeon E5-2680v3 12C 2.5GHz, Aries interconnect Cray Inc.	185,088	5,640.2	7,403.5	
9	King Abdullah University of Science and Technology Saudi Arabia	Shaheen II - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	196,608	5,537.0	7,235.2	2,834
10	Texas Advanced Computing Center/Univ. of Texas United States	Stampede - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P Dell	462,462	5,168.1	8,520.1	4,510



THE LANDSCAPE OF PARALLEL COMPUTING RESEARCH: A VIEW FROM BERKELEY

*Krste Asanović, Rastislav Bodik, Bryan Catanzaro,
Joseph Gebis, Parry Husbands, Kurt Keutzer, David
Patterson, William Plishker, John Shalf, Samuel
Williams, and Katherine Yelick*

PROGRAMOWANIE - KIEDYŚ



PROGRAMOWANIE - TERAZ



WYZWANIA

- W 2005 roku Intel poszedł w ślady IBMa i SUNa i zdecydował się zrezygnować z drogi rozwoju mikroprocesorów polegającej na zwiększaniu częstotliwości zegara. Wzrost szybkości obliczeniowej ma następować przez zastąpienie pojedynczego procesora przez kilka/kilkanaście procesorów (rdzeni) w jednym układzie scalonym.
- Zwiększenie liczby procesorów bez zmiany modelu obliczeniowego nie zaowocuje wzrostem wydajności proporcjonalnym do ilości rdzeni
- Potrzebne jest zupełnie nowe podejście do projektowania procesorów i narzędzi programistycznych



ODWRÓCONE WIĘZY

TRADYCYJNE

- Moc jest za darmo, tranzystory kosztują
- Jeżeli moc jest problemem to jest to moc dynamiczna
- Procesory nie zawierają błędów, błędy mogą się zdarzać jedynie na połączeniach
- Możemy w nieskończoność zwiększać poziom abstrakcji i rozmiar logiczny układów
- Nowe pomysły mogą być testowane w postaci układów scalonych
- Wzrost wydajności pozwala na wzrost pasma i zmniejszenie latencji
- Mnożenie jest powolne, load & store są szybkie



ODWRÓCONE WIĘZY

TRADYCYJNE

- Wzrost wydajności można uzyskać przez równoległość na poziomie instrukcji (ILP) dzięki rozwojowi architektury i kompilatorów (przewidywanie rozgałęzień, wykonanie „out-of-order”, zgadywanie wyników instrukcji warunkowych, układy VLSI).
- Wydajność pojedynczego procesora podwaja się co 18 miesięcy
- Nie ma sensu zrównolegać aplikacji – wystarczy poczekać
- Zwiększanie wydajności odbywa się przez zwiększenie częstotliwości zegara
- Słabsze niż liniowe zwiększenie wydajności dzięki zrównoleglaniu jest porażką.



ODWRÓCONE WIĘZY

TRADYCYJNE

- Moc jest za darmo, tranzystory kosztują
- Jeżeli moc jest problemem to jest to moc dynamiczna
- Procesory nie zawierają błędów, błędy mogą się zdarzać jedynie na połączeniach
- Możemy w nieskończoność zwiększać poziom abstrakcji i rozmiar logiczny układów
- Nowe pomysły mogą być testowane w postaci układów scalonych
- Wzrost wydajności pozwala na wzrost pasma i zmniejszenie latencji
- Mnożenie jest powolne, load & store są szybkie

NOWE

- Moc jest droga, tranzystory są za darmo
- Moc spoczynkowa to do 40% mocy maksymalnej. **(POWER WALL)**
- Procesory produkowane w procesie poniżej 65 nm będą miały duży poziom błędów (miękkich i twardych)
- Opóźnienia sygnału, szумy, sprzężenia elektryczne i magnetyczne, niedokładność zegarów uniemożliwiają w praktyce dowolne zwiększanie rozmiarów układów scalonych
- Koszty produkcji układów $< 65\text{nm}$, oprogramowania, i projektowania sprawiają, że prototypy stają się zawodne
- Wzrost pasma jest co najmniej kwadratowy jako funkcja spadku latencji
- Load & store są wolne, mnożenie jest szybkie **(MEMORY WALL)**

ODWRÓCONE WIĘZY

TRADYCYJNE

- Wzrost wydajności można uzyskać przez równoległość na poziomie instrukcji (ILP) dzięki rozwojowi architektury i kompilatorów (przewidywanie rozgałęzień, wykonanie „out-of-order”, zgadywanie wyników instrukcji warunkowych, układy VLSI).
- Wydajność pojedynczego procesora podwaja się co 18 miesięcy
- Nie ma sensu zrównoleglać aplikacji – wystarczy poczekać
- Zwiększanie wydajności odbywa się przez zwiększanie częstotliwości zegara
- Słabsze niż liniowe zwiększenie wydajności dzięki zrównoleglaniu jest porażką.

NOWE

- Wzrost wydajności dzięki równoległości na poziomie instrukcji się wyczerpał.
(ILP WALL)
- Podwajanie się wydajności pojedynczego procesora to co najmniej 5 lat.
POWER WALL + MEMORY WALL + ILP WALL = BRICK WALL.
- Nie doczekasz się
- Zwiększanie wydajności odbywa się przez zwiększanie równoległości.
- Każde zwiększenie wydajności dzięki zrównoleglению jest sukcesem

MEMORY WALL

Problemy

- Przepustowość
- Opóźnienia
- Pojemność

Przepustowość = częstotliwość × szerokość szyny

Potencjalnie duże możliwości wzrostu – ograniczone technologicznie ale nie przez prawa przyrody.

Opóźnienie – szybkość ładowania kondensatora

Niewielkie możliwości zmniejszenia – prawa przyrody

Pojemność – szybkość wzrostu pojemności układów pamięci DRAMatycznie spadła w połowie ubiegłej dekady.

Do 2002 roku podwajanie co 18 miesięcy

2002 – 1Gbit 2005 – 2Gbit 2009 – 4Gbit

Od 2010 przełom (2010 – 8Gbit 2012 – 16Gbit) ale granice wzrostu są już widoczne



POWER WALL

Energia jako podstawowy czynnik ograniczający

- Moc szczytowa: gęstość mocy $\sim 100 \text{ W/cm}^2$
 - Więcej niż w elementach grzejnych kuchenek elektrycznych
 - Porównywalna z ilością ciepła odprowadzaną przez ścianki cylindrów w silniku spalinowym ($T \sim 1000\text{K}$)
 - Odprowadzenie ciepła
 - Doprowadzenie mocy
 - Koszty chłodzenia
 - tradycyjne superkomputery to niezwykle wyrafinowane urządzenia chłodzące zbudowane wokół niekonwencjonalnego źródła ciepła

POWER WALL

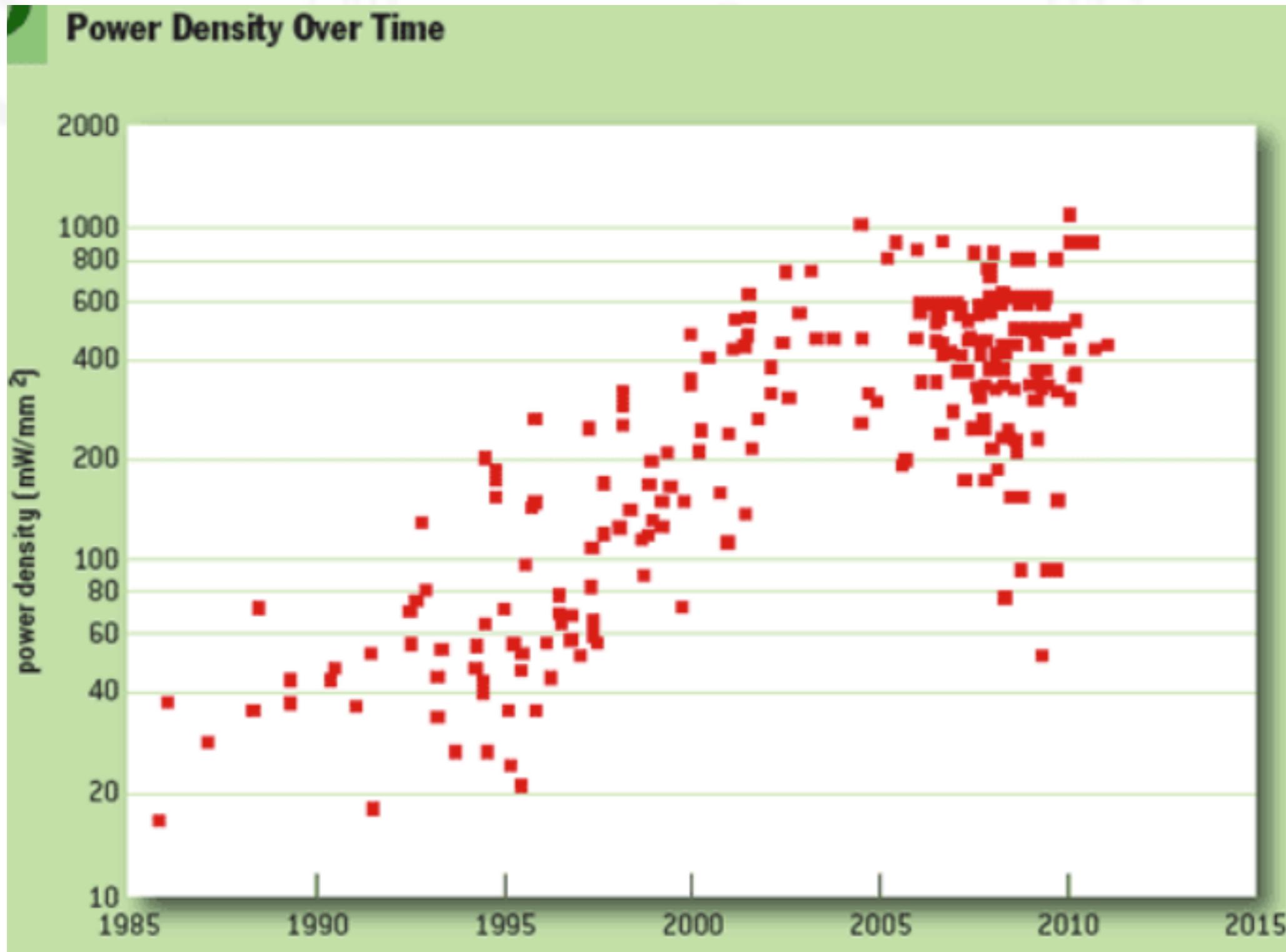
Energia jako podstawowy czynnik ograniczający

- Całkowite zużycie energii
 - Czas życia na bateriach
 - Koszty eksploatacyjne
 - Względy ekologiczne

POWER WALL

- Równoległość jest efektywną energetycznie formą zwiększenia wydajności.
- Wiele małych rdzeni daje większą wydajność na powierzchnię chipu dla kodów równoległych
- Dużo małych rdzeni pozwala na elastyczną politykę zarządzania energią
- Defekty w małych rdzeniach mało kosztują (układy z wadami w pojedynczych rdzeniach są nadal funkcjonalne – SP3 używa 7 z 8 rdzeni procesora Cell)
- Małe rdzenie jest łatwiej zaprojektować, zweryfikować i przetestować a ich charakterystyka elektryczna jest łatwiejsza do przewidywania

POWER WALL

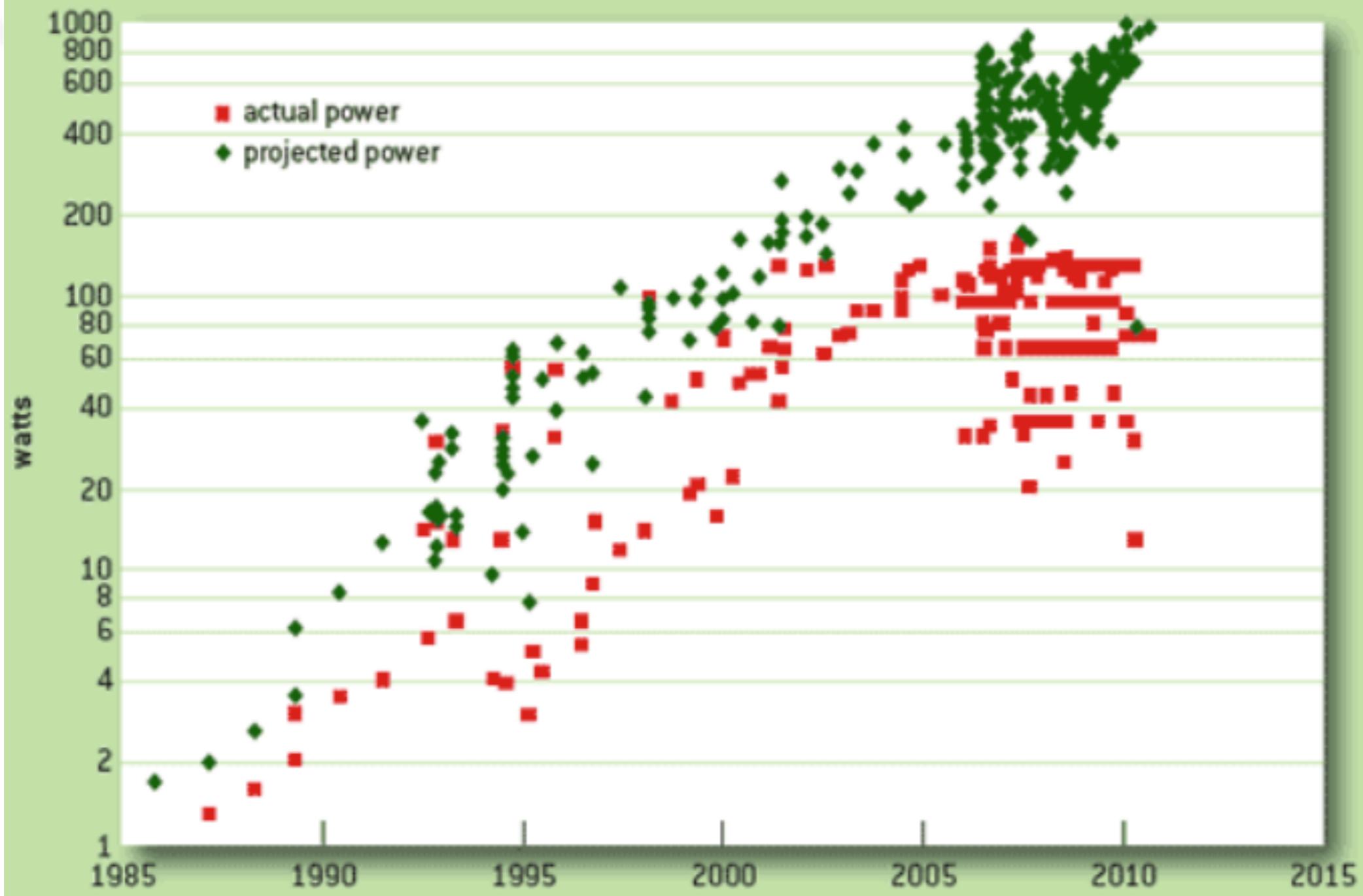


Andrew Danowitz, Kyle Kelley, James Mao, John P. Stevenson, Mark Horowitz (2012) CPU DB: Recording Microprocessor History. ACM QUEUE, 2012



POWER WALL

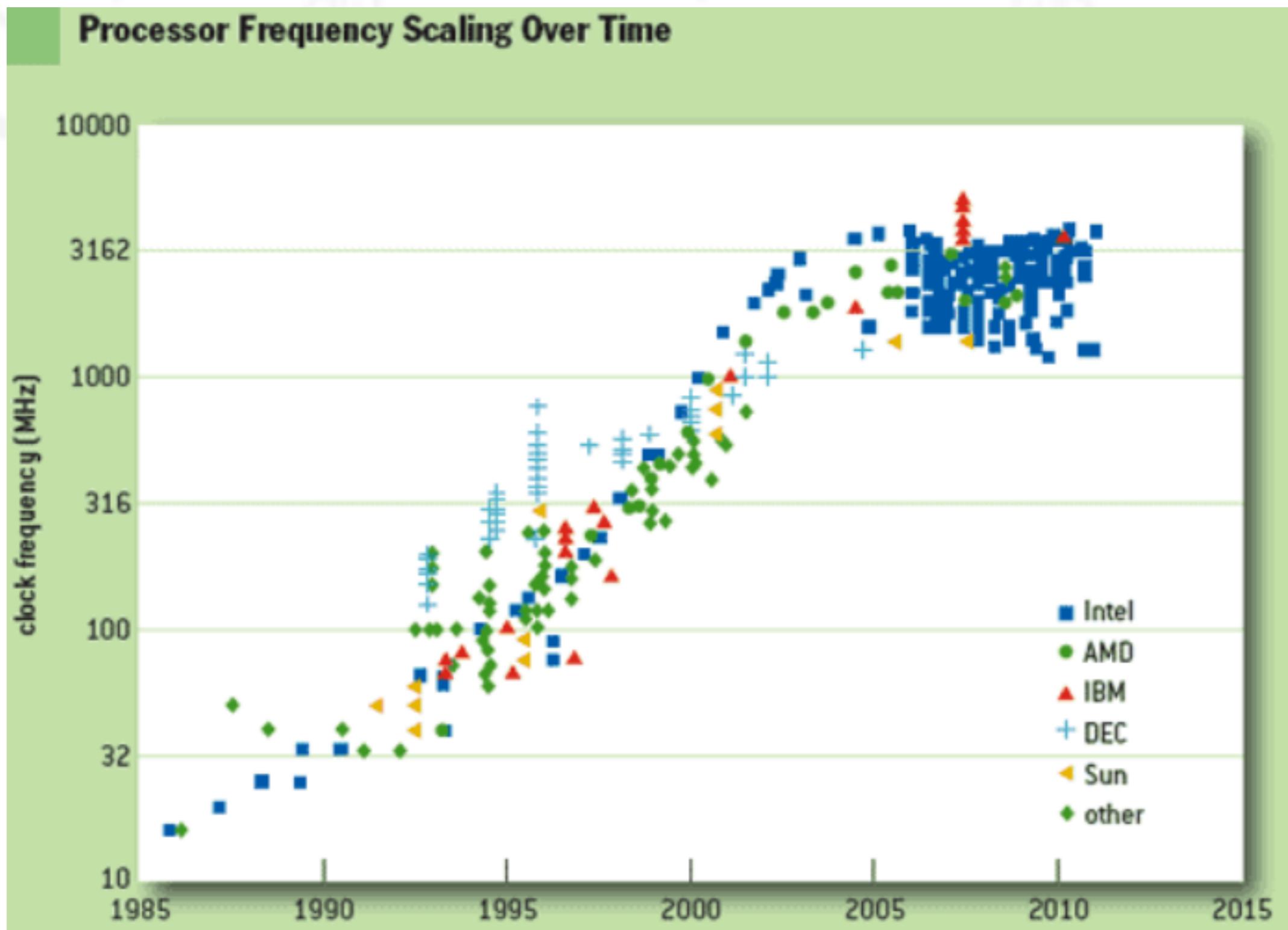
How Power Should Have Scaled



Andrew Danowitz, Kyle Kelley, James Mao, John P. Stevenson, Mark Horowitz (2012) CPU DB: Recording Microprocessor History. ACM QUEUE, 2012



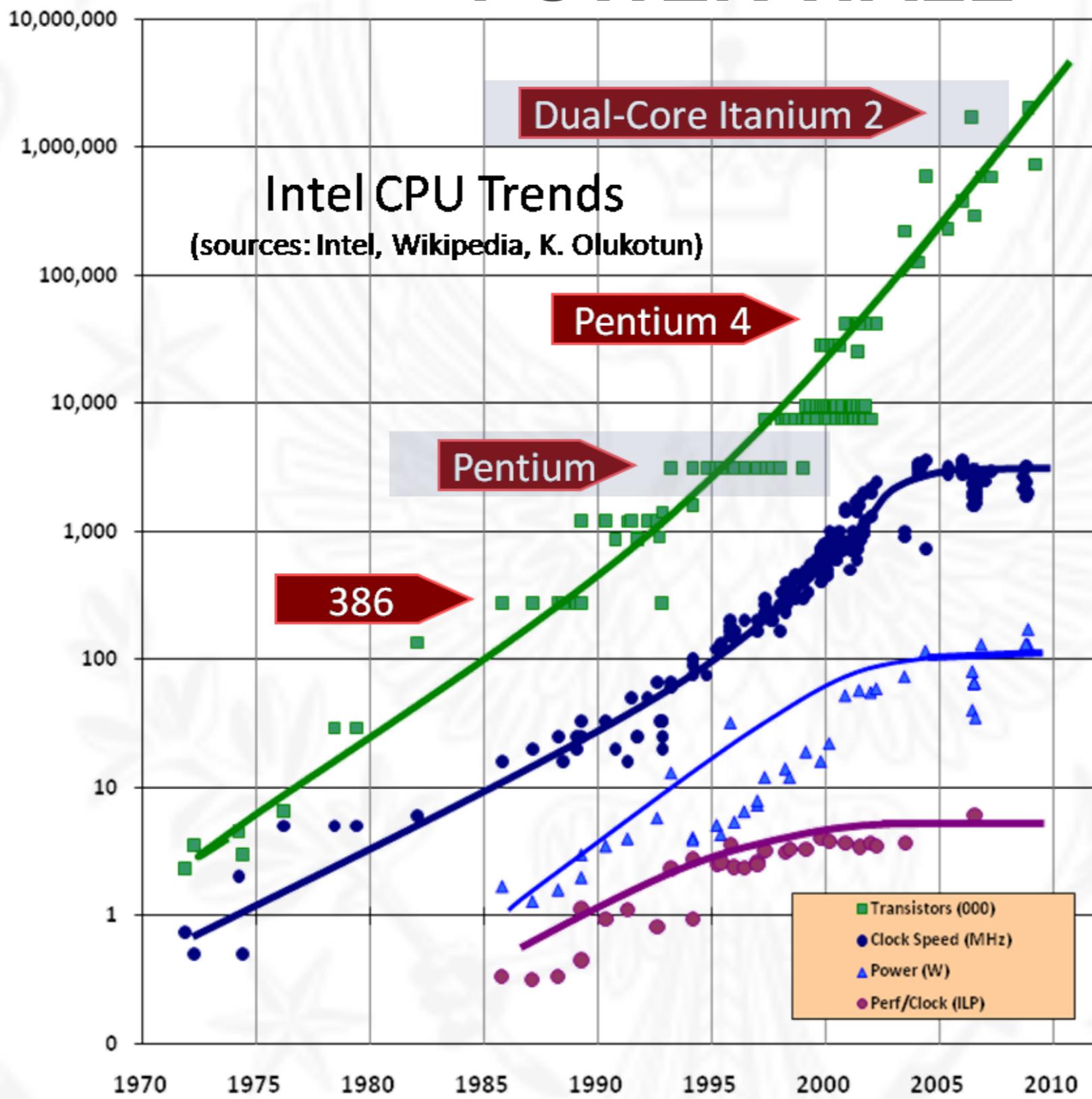
POWER WALL



Andrew Danowitz, Kyle Kelley, James Mao, John P. Stevenson, Mark Horowitz (2012) CPU DB: Recording Microprocessor History. ACM QUEUE, 2012



POWER WALL



<http://www.gotw.ca/publications/concurrency-ddj.htm>

JEST JESZCZE NADZIEJA

Prawo Moore'a obowiązuje:

- Setki rdzeni obliczeniowych w jednym układzie scalonym
- Brak latencji i duża przepustowość dla komunikacji wewnętrz układu scalonego



JEST JESZCZE NADZIEJA

Małe jest piękne

- Rozwój procesorów polagający na zwiększaniu liczby tranzystorów trwał 30 lat i 5 rzędów wielkości
 - Intel 4004 (1971) 2300 tranzystorów
 - Intel Core (2006) ~140 mln tranzystorów
- Od momentu w którym liczba tranzystorów jest wystarczająca do uzyskania wydajności rzędu **I operacja / I cykl zegara**
zyski z superskalarności (wzrost wydajności) są mniejsze niż nakłady (liczba tranzystorów)
 - R3000 (1992) **7 MFlop @ 1 mln tranzystorów @ 25 MHz**
 - Pentium Quad Core **100 000 MFlop @ 1000 mln tranzystorów @ 25 00 Mhz**
 - ?
- Zwiększenie wydajności jest możliwe przez zwiększenie liczby małych układów



JEST JESZCZE NADZIEJA

Małe jest piękne

- Rozwój procesorów polagający na zwiększaniu liczby tranzystorów trwał 30 lat i 5 rzędów wielkości
 - Intel 4004 (1971) 2300 tranzystorów
 - Intel Core (2006) ~140 mln tranzystorów
- Od momentu w którym liczba tranzystorów jest wystarczająca do uzyskania wydajności rzędu **I operacja / I cykl zegara**
zyski z superskalarności (wzrost wydajności) są mniejsze niż nakłady (liczba tranzystorów)
 - R3000 (1992) **7 MFlop @ 1 mln tranzystorów @ 25 MHz**
 - Pentium Quad Core **100 000 MFlop @ 1000 mln tranzystorów @ 25 00 Mhz**
 - $(100\ 000\ / 7) / (1000 \times 2500 / 25)$
- Zwiększenie wydajności jest możliwe przez zwiększenie liczby małych układów

JEST JESZCZE NADZIEJA

Małe jest piękne

- Rozwój procesorów polagający na zwiększaniu liczby tranzystorów trwał 30 lat i 5 rzędów wielkości
 - Intel 4004 (1971) 2300 tranzystorów
 - Intel Core (2006) ~140 mln tranzystorów
- Od momentu w którym liczba tranzystorów jest wystarczająca do uzyskania wydajności rzędu **I operacja / I cykl zegara**
zyski z superskalarności (wzrost wydajności) są mniejsze niż nakłady (liczba tranzystorów)
 - R3000 (1992) **7 MFlop @ 1 mln tranzystorów @ 25 MHz**
 - Pentium Quad Core **100 000 MFlop @ 1000 mln tranzystorów @ 25 00 Mhz**
 - $(100\ 000 / 7) / (1000 \times 2500 / 25) = (100\ 000 / 7) / (100\ 000)$
- **Zwiększenie wydajności jest możliwe przez zwiększenie liczby małych układów**

JEST JESZCZE NADZIEJA

Małe jest piękne

- Rozwój procesorów polagający na zwiększaniu liczby tranzystorów trwał 30 lat i 5 rzędów wielkości
 - Intel 4004 (1971) 2300 tranzystorów
 - Intel Core (2006) ~140 mln tranzystorów
- Od momentu w którym liczba tranzystorów jest wystarczająca do uzyskania wydajności rzędu **I operacja / I cykl zegara**
zyski z superskalarności (wzrost wydajności) są mniejsze niż nakłady (liczba tranzystorów)
 - R3000 (1992) **7 MFlop @ 1 mln tranzystorów @ 25 MHz**
 - Pentium Quad Core **100 000 MFlop @ 1000 mln tranzystorów @ 25 00 Mhz**
 - $(100\ 000 / 7) / (1000 \times 2500 / 25) = (100\ 000 / 7) / (100\ 000) = 1/7$
- Zwiększenie wydajności jest możliwe przez zwiększenie liczby małych układów

Procesor	Tranzystory	Process (um)	Pow. (mm2)	Tranz./Rdzeń	Opracowany	Rdzenie
4004	2 300	10	12	2 300	1971	1
8008	3 500	10	14	3 500	1972	1
8080	6 000	6	20	6 000	1974	1
8086	20 000	3	33	20 000	1978	1
80286	133 000	2	49	133 000	1982	1
80386	275 000	1	104	275 000	1985	1
80486	1 200 000	1	160	1 200 000	1989	1
Pentium	3 100 000	0	294	3 100 000	1993	1
Pentium II	7 500 000	0	195	7 500 000	1997	1
Pentium III	9 500 000	0	128	9 500 000	1999	1
Pentium IV	42 000 000	0	217	42 000 000	2003	1
Intel Core	291 000 000	0	143	145 500 000	2006	2
Penryn	410 000 000	0	107	205 000 000	2009	2
Nehalem	731 000 000	0	263	182 750 000	2010	4
Westmere	2 600 000 000	0	512	260 000 000	2011	10
Sandy Bridge	995 000 000	0	216	248 750 000	2011	4
Ivy Bridge	1 400 000 000	0,022	160	350 000 000	2012	

SPRZĘT

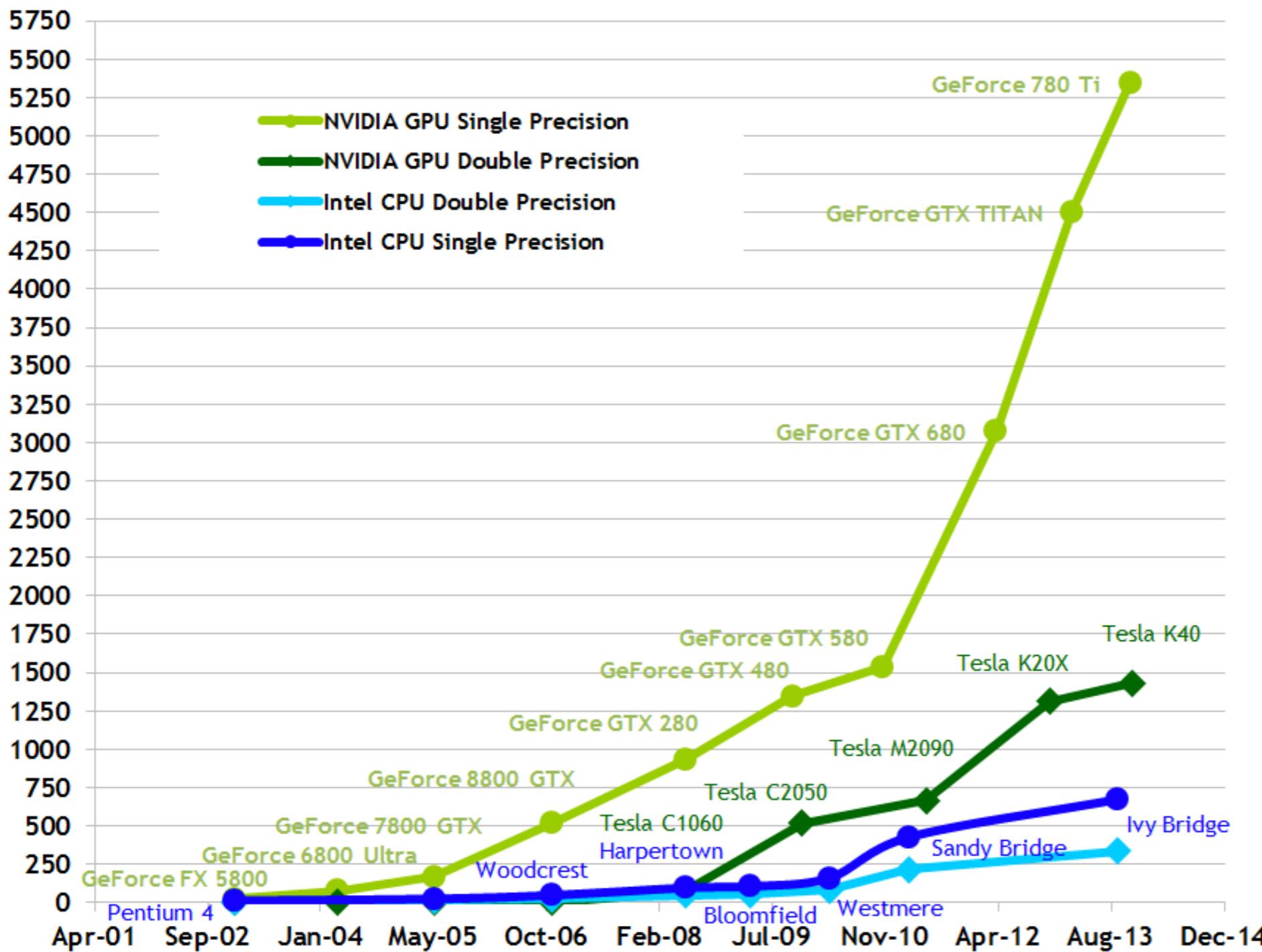
4.1.2 Will we really fit 1000s of cores on one economical chip

This significant reduction in the size and complexity of the basic processor building block of the future means that many more cores can be economically implemented on a single die; furthermore, this number can double with each generation of silicon. For example, the “manycore” progression might well be 128, 256, 512, ... cores instead of the current “multicore” plan of 2, 4, 8, ... cores over the same semiconductor process generations.

- 128 rdzeni w procesorze graficznym w 2007 roku
- 480 rdzeni w procesorze graficznym w 2009 roku
- 1536 rdzeni w procesorze graficznym w 2011 roku
- 2688 rdzeni w procesorze graficznym w 2013 roku
- 3072 rdzeni w procesorze graficznym w 2015 roku



CPU A GPU



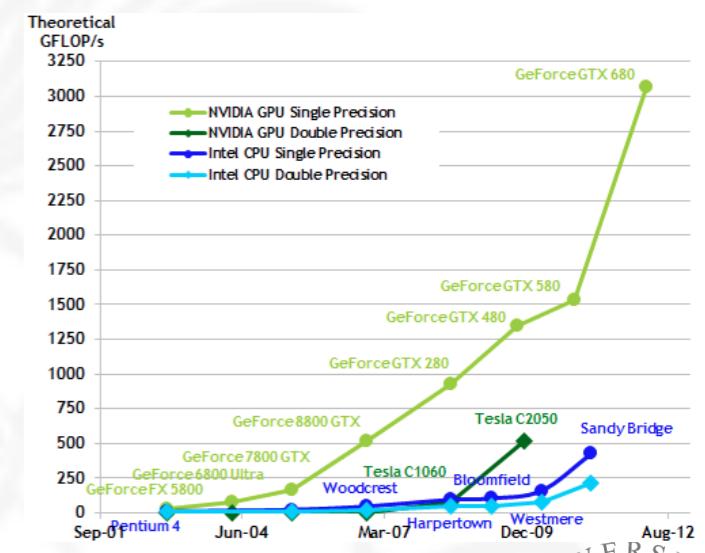
GeForce GTX Titan

4.5 TFLOPS
Single Precision

1.5 TFLOPS
Double Precision

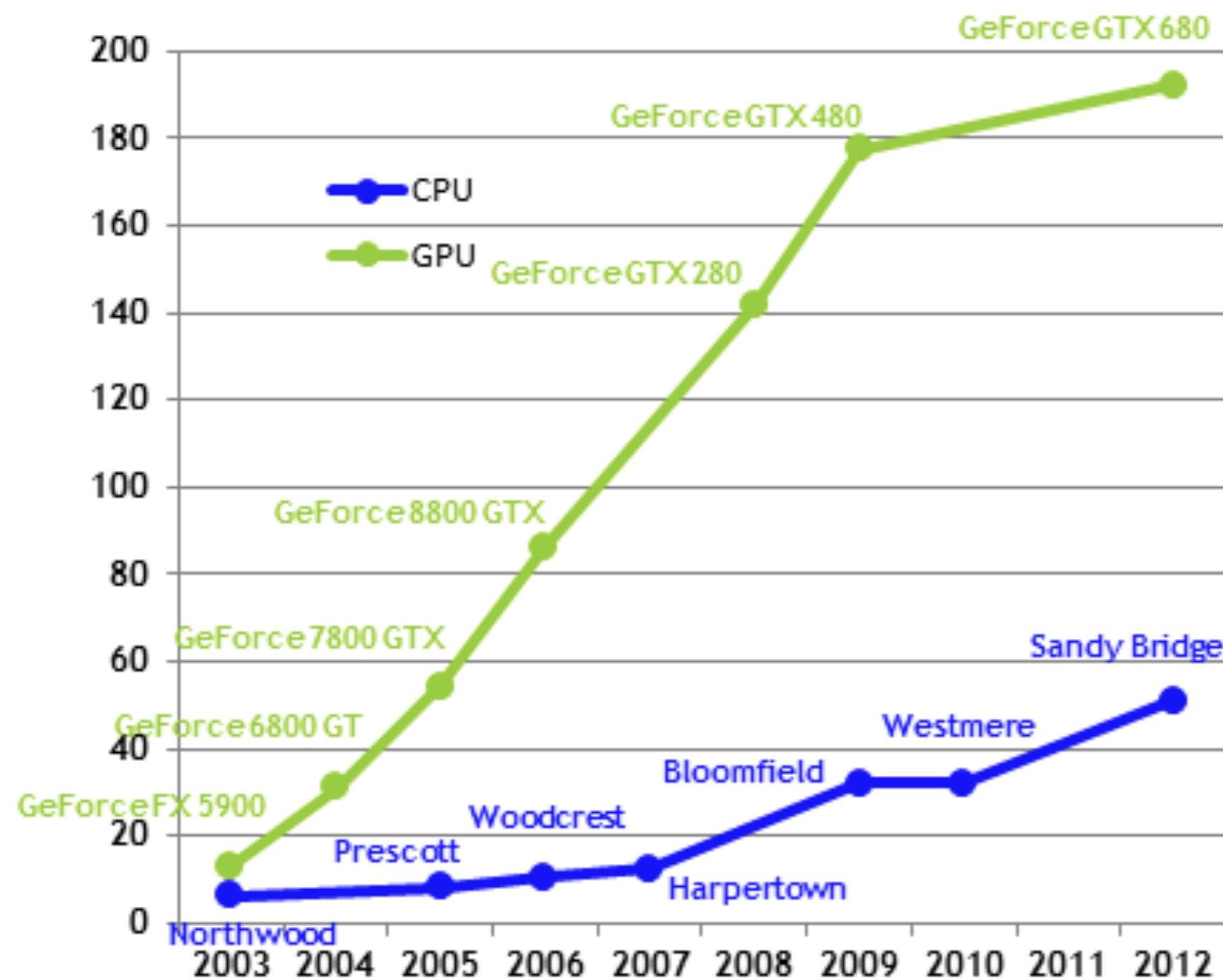
GeForce GTX 680

3.1 TFLOPS
Single Precision



CPU A GPU

Theoretical GB/s



GeForce GTX Titan

288 GB/s

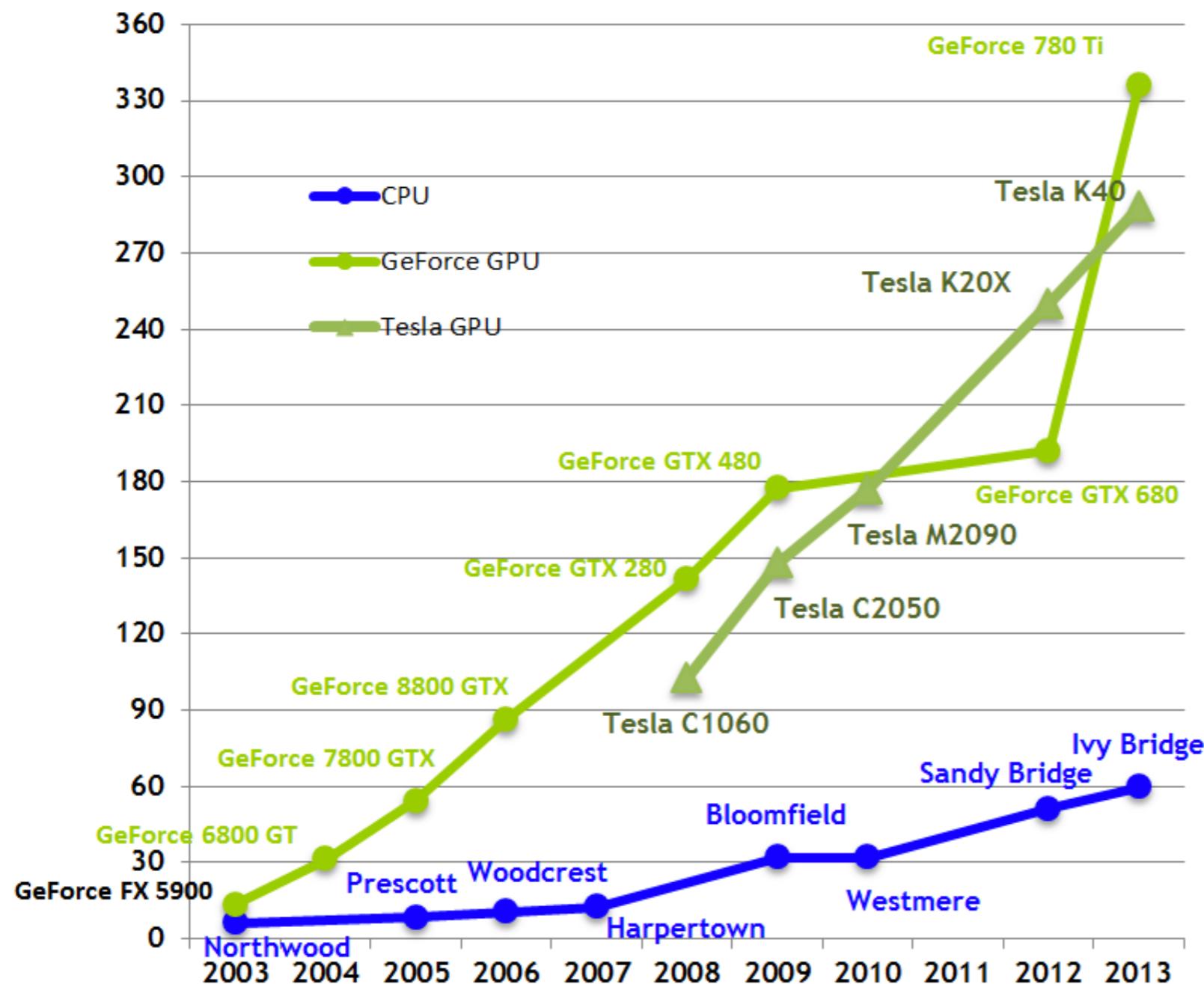
GeForce GTX 680

188GB/s

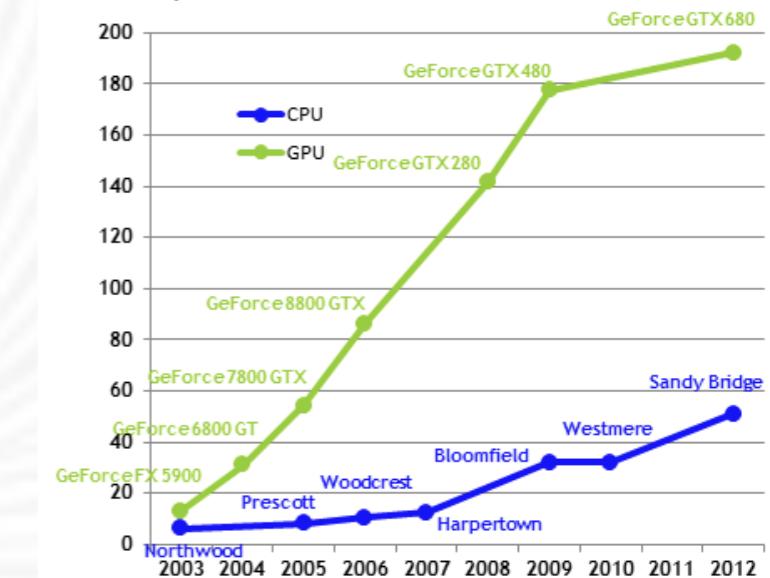


CPU A GPU

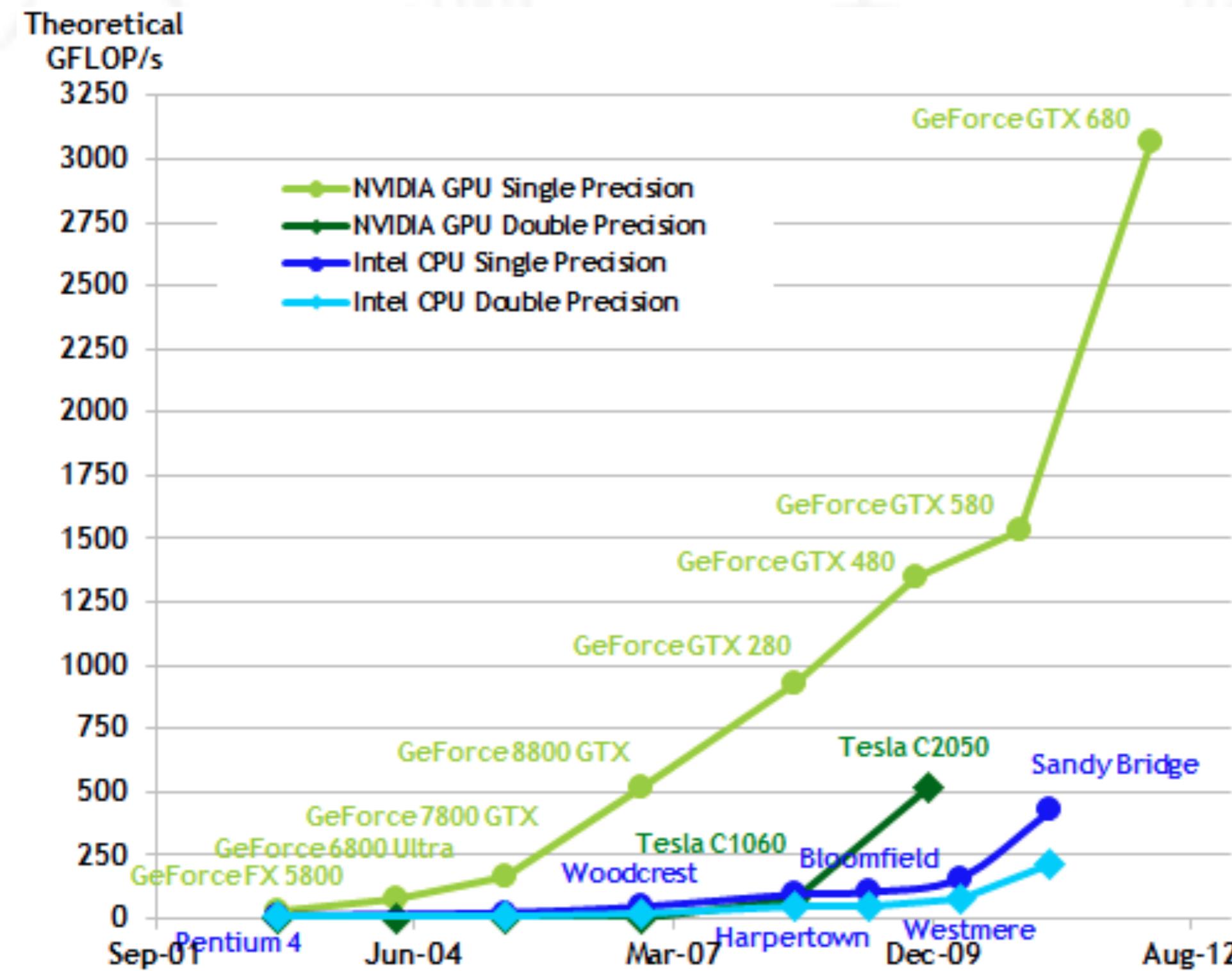
Theoretical GB/s



Theoretical GB/s



CPU A GPU



GeForce GTX Titan

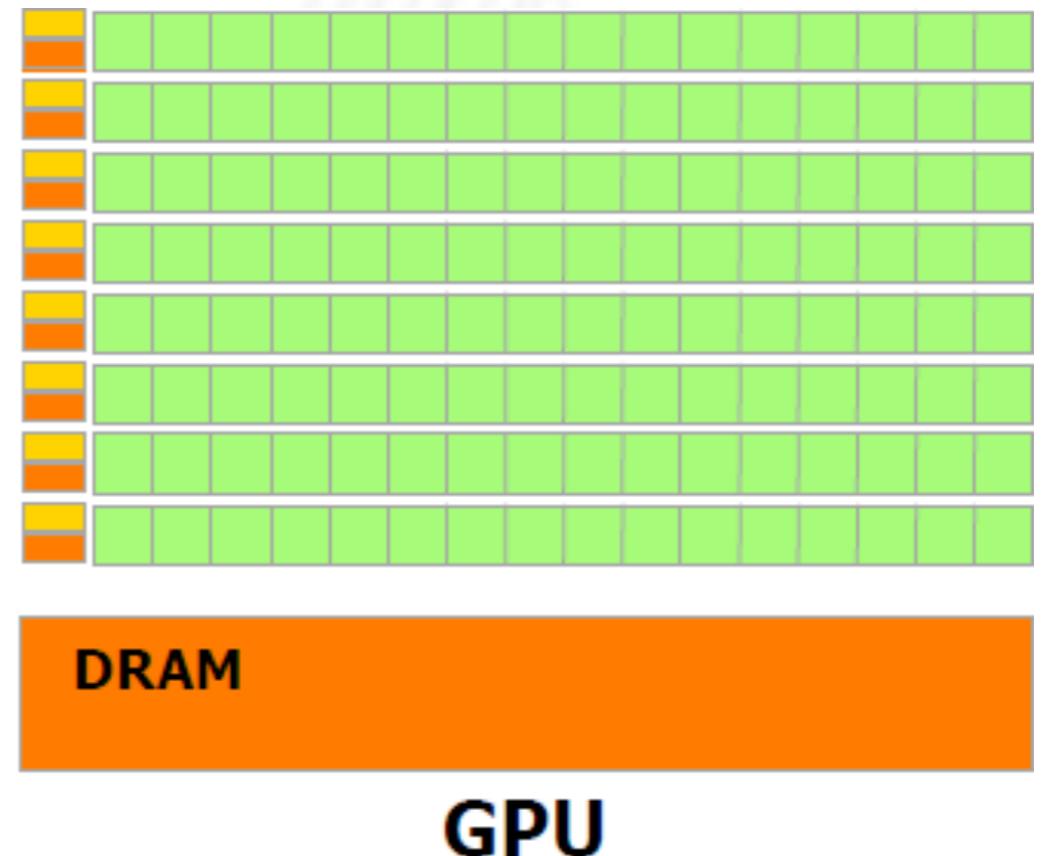
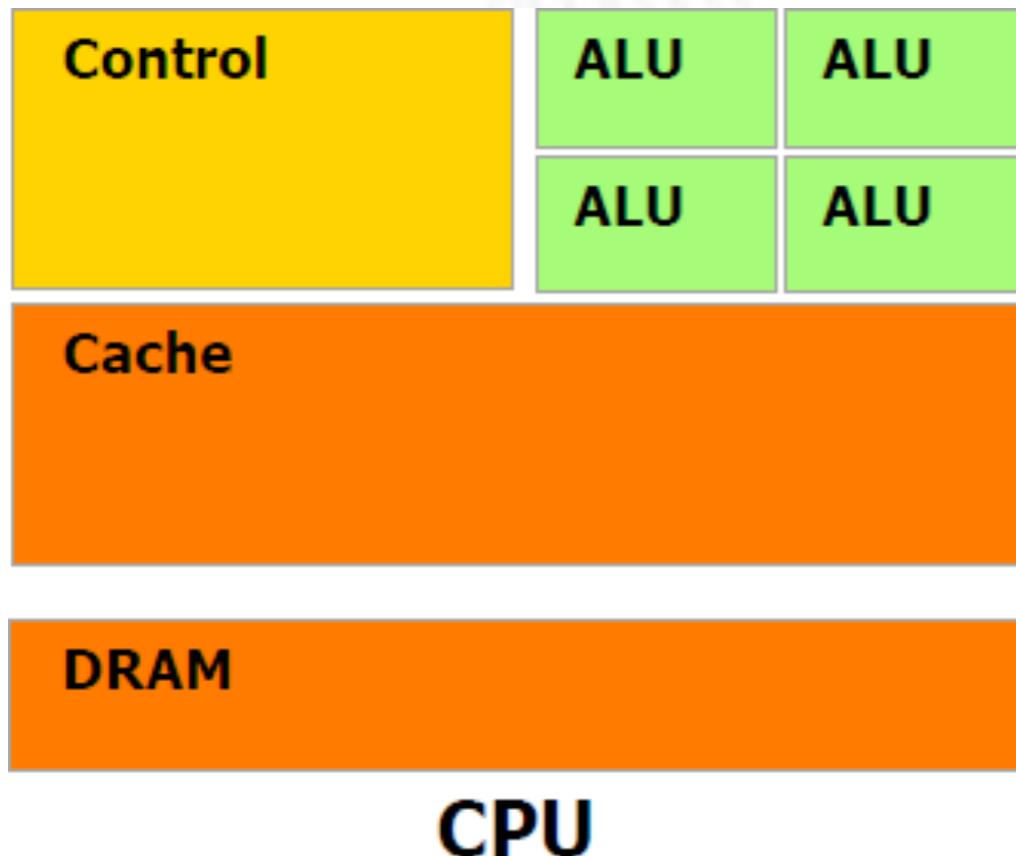
4.5 TFLOPS
Single Precision

1.5 TFLOPS
Double Precision

GeForce GTX 680

3.1 TFLOPS
Single Precision

CPU A GPU



CPU A GPU

Procesor	Tranzystory	Process (um)	Pow. (mm2)	Tranz./Rdzeń	Opracowany	Rdzenie
4004	2 300	10	12	2 300	1971	1
8008	3 500	10	14	3 500	1972	1
8080	6 000	6	20	6 000	1974	1
8086	20 000	3	33	20 000	1978	1
80286	133 000	2	49	133 000	1982	1
80386	275 000	1	104	275 000	1985	1
80486	1 200 000	1	160	1 200 000	1989	1
Pentium	3 100 000	0	294	3 100 000	1993	1
Pentium II	7 500 000	0	195	7 500 000	1997	1
Pentium III	9 500 000	0	128	9 500 000	1999	1
Pentium IV	42 000 000	0	217	42 000 000	2003	1
Intel Core	291 000 000	0	143	145 500 000	2006	2
Penryn	410 000 000	0	107	205 000 000	2009	2
Nehalem	731 000 000	0	263	182 750 000	2010	4
Westmere	2 600 000 000	0	512	260 000 000	2011	10
Sandy Bridge	995 000 000	0	216	248 750 000	2011	4
Ivy Bridge	1 400 000 000	0,022	160	350 000 000	2012	4



Procesor	Tranzystory	Process (um)	Pow. (mm2)	Tranz./Rdzeń	Opracowany	Rdzenie
4004	2 300	10	12	2 300	1971	1
8008	3 500	10	14	3 500	1972	1
8080	6 000	6	20	6 000	1974	1
8086	20 000	3	33	20 000	1978	1
80286	133 000	2	49	133 000	1982	1
80386	275 000	1	104	275 000	1985	1
80486	1 200 000	1	160	1 200 000	1989	1
Pentium	3 100 000	0	294	3 100 000	1993	1
Pentium II	7 500 000	0	195	7 500 000	1997	1
Pentium III	9 500 000	0	128	9 500 000	1999	1
Pentium IV	42 000 000	0	217	42 000 000	2003	1
Intel Core	291 000 000	0	143	145 500 000	2006	2
Penryn	410 000 000	0	107	205 000 000	2009	2
Nehalem	731 000 000	0	263	182 750 000	2010	4
Westmere	2 600 000 000	0	512	260 000 000	2011	10
Sandy Bridge	995 000 000	0	216	248 750 000	2011	4
Ivy Bridge	1 400 000 000	0,022	160	350 000 000	2012	4



CPU A GPU

Procesor	Tranzystory	Process (um)	Pow. (mm ²)	Tranz./Rdzeń	Opracowany	Rdzenie
G80	681 000 000	0,09	480 mm ²	5 320 313	2006	128
GT200	1 400 000 000	0,055	576 mm ²	5 833 333	2008	240
1st GF100	3 200 000 000	0,04	526 mm ²	6 666 667	2010	480
2nd GF100	3 000 000 000	0,04	520 mm ²	5 859 375	2010	512
GK104	3 540 000 000	0,028	294 mm ²	2 304 688	2012	1536
GK110	7 080 000 000	0,028	521 mm ²	2 633 929	2012	2688
GM200	8 100 000 000	0,028	601 mm ²	2 636 719	2015	3072

Procesor	Tranzystory	Process (um)	Pow. (mm ²)	Tranz./Rdzeń	Opracowany	Rdzenie
4004	2 300	10	12	2 300	1971	1
8008	3 500	10	14	3 500	1972	1
8080	6 000	6	20	6 000	1974	1
8086	20 000	3	33	20 000	1978	1
80286	133 000	2	49	133 000	1982	1
80386	275 000	1	104	275 000	1985	1
80486	1 200 000	1	160	1 200 000	1989	1
Pentium	3 100 000	0	294	3 100 000	1993	1
Pentium II	7 500 000	0	195	7 500 000	1997	1
Pentium III	9 500 000	0	128	9 500 000	1999	1
Pentium IV	42 000 000	0	217	42 000 000	2003	1
Intel Core	291 000 000	0	143	145 500 000	2006	2
Penryn	410 000 000	0	117	205 000 000	2009	2
Nehalem	731 000 000	0	263	182 750 000	2010	4
Westmere	2 600 000 000	0	512	260 000 000	2011	10
Sandy Bridge	995 000 000	0	216	248 750 000	2011	4
Ivy Bridge	1 400 000 000	0,022	160	350 000 000	2012	4

17.03.2016



ONE SIZE FITS ALL?

Prawo Amdahla:

Wydajność programu składającego się z części sekwencyjnej i części równoległej jest limitowana czasem wykonania części sekwencyjnej

$$T[1] = T_{sek}[1] + T_{równ}[1] \quad T[1] = \alpha + (1 - \alpha)$$

$$T[N] = T_{sek}[N] + T_{równ}[N] = \alpha + \frac{(1 - \alpha)}{N}$$

$$P[N] = \frac{T[1]}{T[N]} = \frac{1}{\alpha + \frac{(1 - \alpha)}{N}} = \frac{N}{N\alpha + (1 - \alpha)} \xrightarrow{N \rightarrow \infty} \frac{1}{\alpha}$$

ONE SIZE FITS ALL?

Przykład:

Rdzeń skalarny 10 mln tranzystorów

Rdzeń superskalarny 100 mln tranzystorów

Chip 1000 mln tranzystorów:

- 100 prostych rdzeni
- 10 rdzeni superskalarnych
- Dowolny miiks



ONE SIZE FITS ALL?

Przykład:

Kod 50% równoległy 50% sekwencyjny

$$P[100s] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.5 + \frac{0.5}{100}} = \frac{1}{0.505} = 1.98$$

$$P[90s+1ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.25 + \frac{0.5}{90}} = \frac{1}{0.256} = 3.91$$

$$P[10ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.25 + \frac{0.25}{10}} = \frac{1}{0.275} = 3.64$$

ONE SIZE FITS ALL?

Przykład:

Kod 90% równoległy 10% sekwencyjny

$$P[100s] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.1 + \frac{0.9}{100}} = \frac{1}{0.109} = 9.17$$

$$P[90s + 1ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.05 + \frac{0.9}{90}} = \frac{1}{0.06} = 16.7$$

$$P[10ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.05 + \frac{0.45}{10}} = \frac{1}{0.095} = 10.5$$



ONE SIZE FITS ALL?

Przykład:

Kod 99% równoległy 1% sekwencyjny

$$P[100s] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.01 + \frac{0.99}{100}} = \frac{1}{0.0199} = 50.3$$

$$P[90s+1ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.005 + \frac{0.99}{90}} = \frac{1}{0.016} = 62.5$$

$$P[10ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.005 + \frac{0.495}{10}} = \frac{1}{0.0545} = 18.3$$

ONE SIZE FITS ALL?

Przykład:

Kod 99.9% równoległy 0.1% sekwencyjny

$$P[100s] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.001 + \frac{0.999}{100}} = \frac{1}{0.01099} = 90.1$$

$$P[90s+1ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.0005 + \frac{0.999}{90}} = \frac{1}{0.0116} = 86.2$$

$$P[10ss] = \frac{1}{\alpha + \frac{1-\alpha}{N}} = \frac{1}{0.0005 + \frac{0.4995}{10}} = \frac{1}{0.05045} = 19.8$$

SPEKTAKULARNE EFEKTY

- Fast k Nearest Neighbor Search using GPU
 - Przyspieszenie **470 razy**
- Efficient Computation of Sum Products on GPUs
 - **270 x**
- Graphic-Card Cluster for Astrophysics (GraCCA)
 - **250 x**
- A Fast Similarity Join Algorithm
 - **100 x**
- Molecular Dynamics Simulations
 - **40-100 x**
- Fast Exact String Matching
 - **35 x**
- Smith Waterman Sequence Alignment
 - **30 x**
- Atomistic Monte Carlo Simulations
 - **60 x**



SPEKTAKULARNE EFEKTY

- Fast k Nearest Neighbor Search using GPU
 - Przyspieszenie 470 razy
- Efficient Computation of Sum Products on GPUs
 - 270 x
- Graphic-Card Cluster for Astrophysics (GraCCA)
 - 250 x
- A Fast Similarity Join Algorithm
 - 100 x
- Molecular Dynamics Simulations
 - 40-100 x
- Fast Exact String Matching
 - 35 x
- Smith Waterman Sequence Alignment
 - 30 x
- Atomistic Monte Carlo Simulations
 - 60 x

Potraktujmy z odrobiną sceptyczmu:

- Porównujmy kod zoptymalizowany na CPU (wektorowy, instrukcje SSE, zmodyfikowany algorytm) z kodem zoptymalizowanym na GPU
- Porównujmy wydajność całego CPU (np. 8 rdzeni) z wydajnością GPU

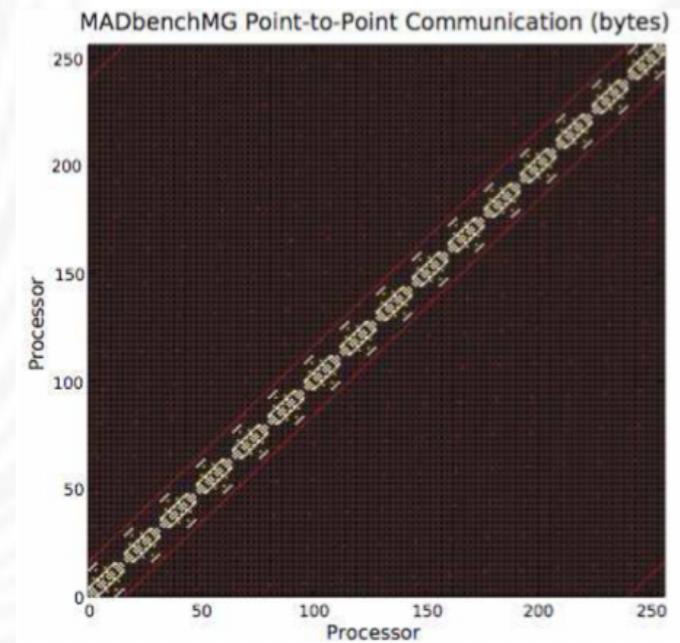
OBSZARY APLIKACJI RÓWNOLEGŁYCH

Algorytmy o podobnej strukturze komunikacji i organizacji danych można pogrupowane w klasy równoważności.

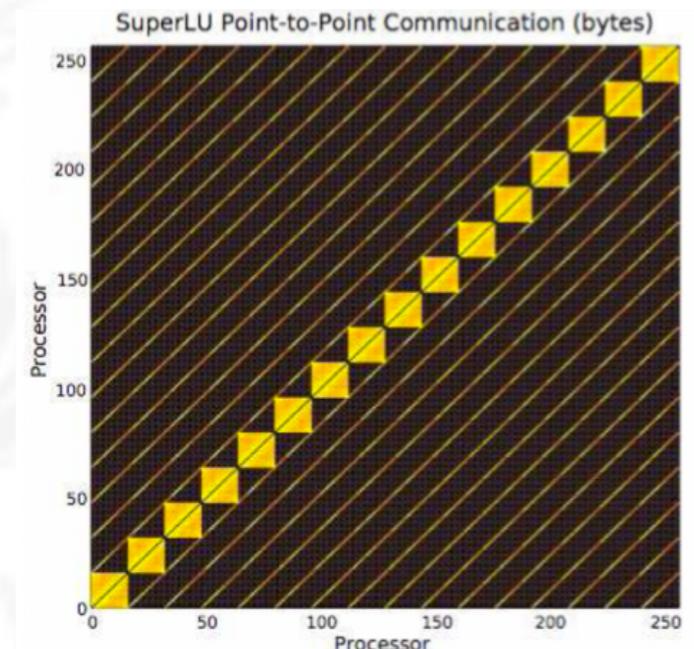
Programy mogą składać się z wielu problemów każdy z nich należeć do innej klasy.

SIEDEM KLAS PODSTAWOWYCH

1. Algebra liniowa dla układów z gęstymi macierzami – klasyczne metody rozwiązywania równań wielu zmiennych

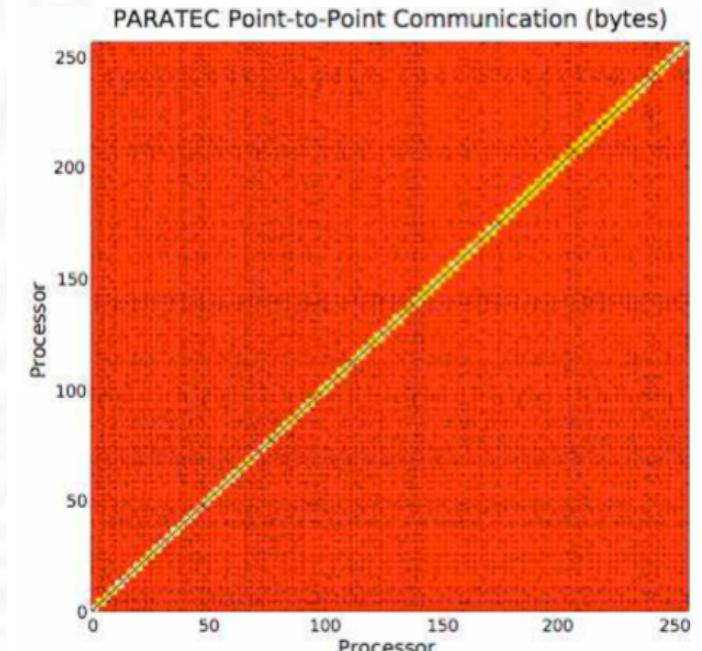


2. Algebra liniowa dla układów z rzadkimi macierzami – formalnie podobne problemy, ale układy zawierające dużo zer, w związku z tym dane są przechowywane w postaci skompresowanej

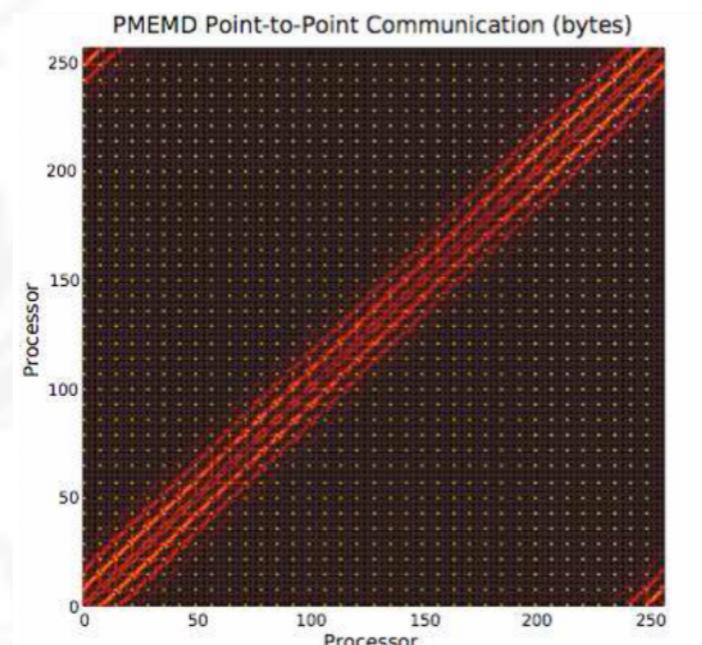


SIEDEM KLAS PODSTAWOWYCH

3. Metody spektralne: dane w domenie częstotliwości oszczędne obliczeniowo wymagają intensywnej komunikacji, często każdy z każdym (FFT) - przykład komunikacja dla 3D FFT

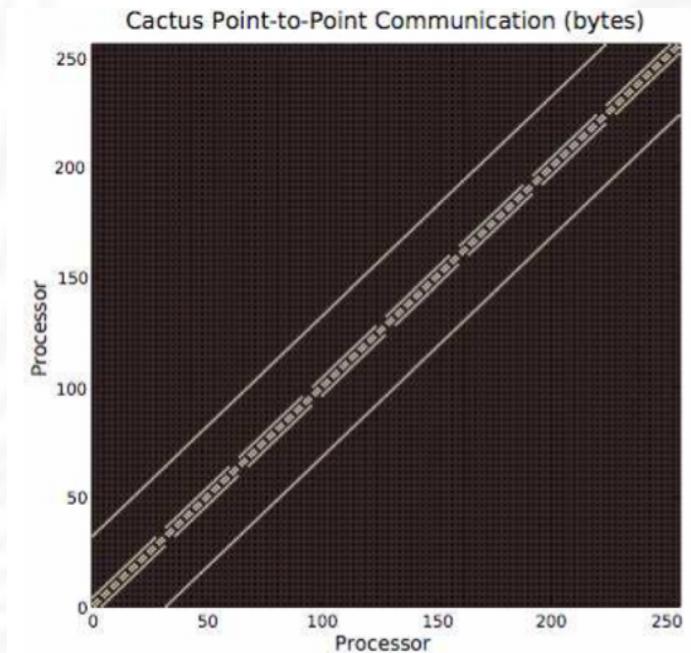


4. Problemy N-ciał: oddziaływanie między dyskretnymi punktami w przestrzeni $O(N^2)$. Metody hierarchiczne $O(N \ln N)$ i $O(N)$ (komunikacja dla metody particle – mesh Ewald)



SIEDEM KLAS PODSTAWOWYCH

5. Siatki regularne (równania różniczkowe na siatkach) dane zlokalizowane przestrzennie
6. Siatki nieregularne – listy sąsiedztwa, równania różniczkowe na nieregularnych siatkach przestrzennych (triangulacja, obiekty inżynierskie)
7. Symulacje Monte Carlo / MapReduce – embarrassingly parallel, komunikacja nie jest dużym problemem



SZEŚĆ KLAS DODATKOWYCH

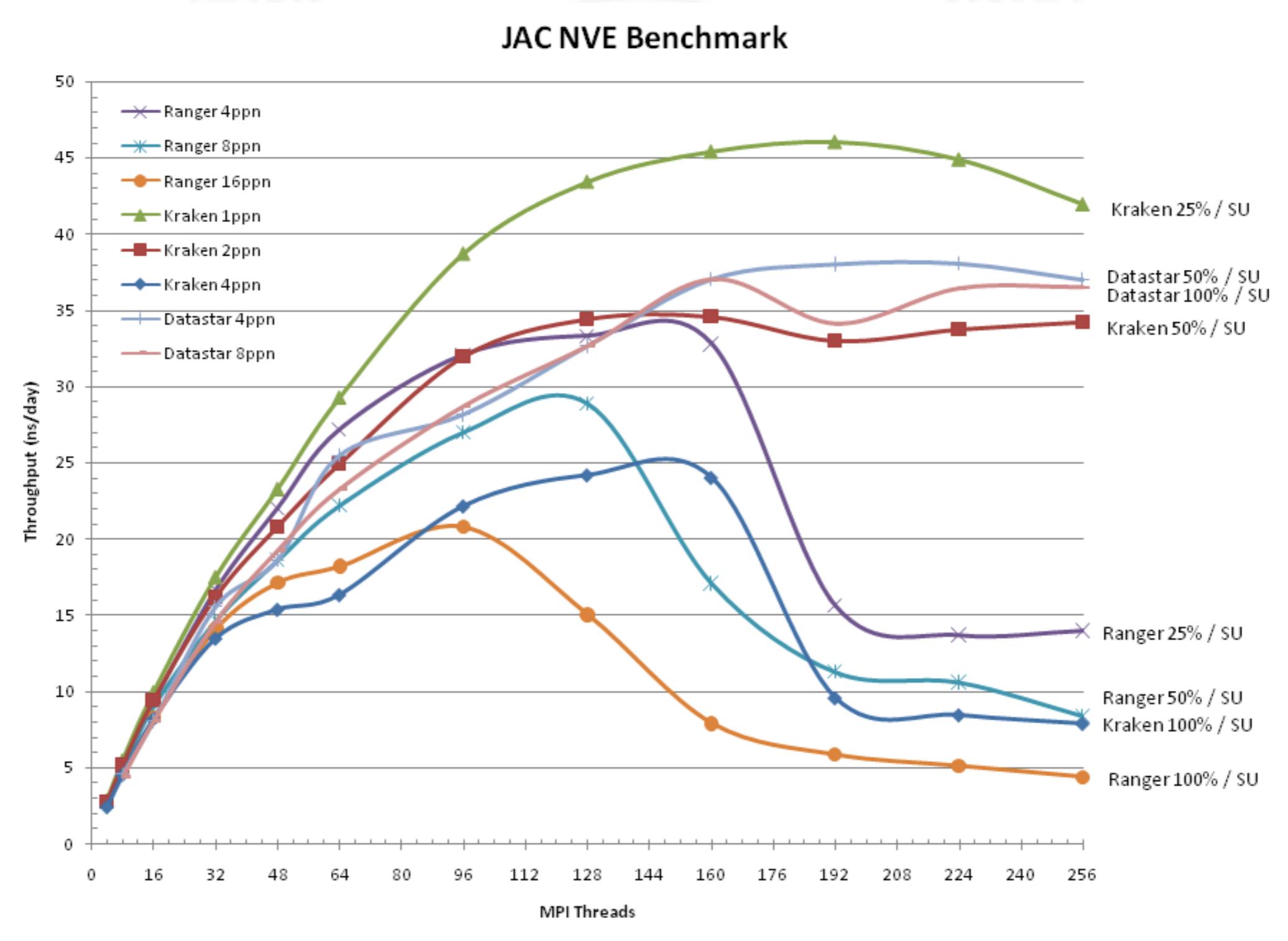
- 8.** Logika kombinacyjna (np. szyfrowanie/deszyfrowanie)
- 9.** Przeszukiwanie grafów (dużo nieuporządkowanych odwołań do pamięci, mało obliczeń)
- 10.** Modele grafowe (modele, w których węzły reprezentują zmienne losowe a krawędzie warunki – Sieci Bayesowskie, Ukryte Modele Markowa)
- 11.** Automaty skończone (układy, których zachowanie zdefiniowane jest przez stany i reguły przejścia między nimi)
- 12.** Programowanie dynamiczne (poszukiwanie optymalnego rozwiązania przez rozwiązywanie pokrywających się prostszych problemów)
- 13.** Algorytmy podziału i ograniczeń (Podział problemu na prostsze i wyrzucanie nieoptymalnych rozwiązań)



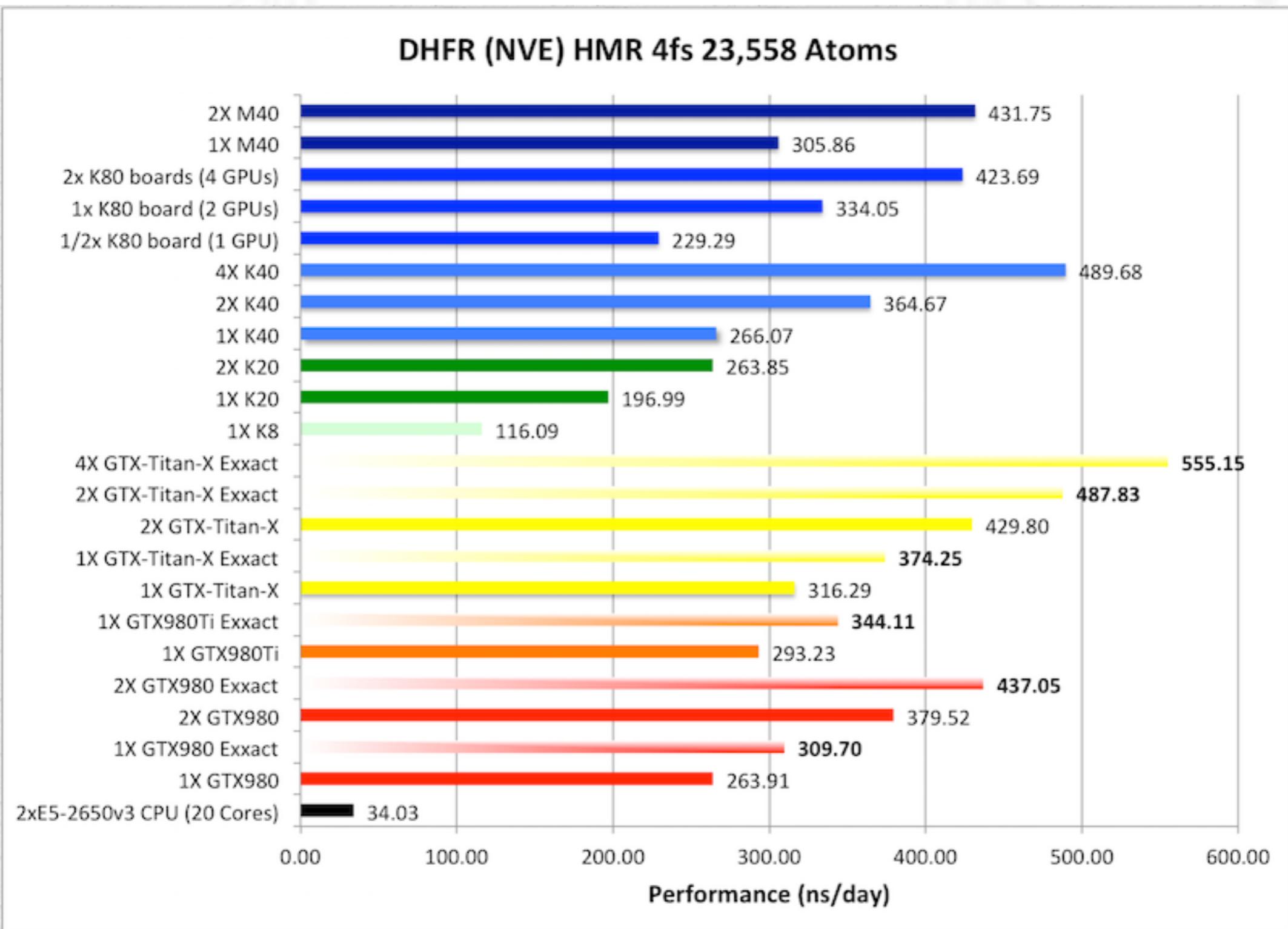
PODSTAWOWE OGRANICZENIA WYDAJNOŚCI ALGORYTMÓW

- Komunikacja między węzłami obliczeniowymi
- Latencji dostępu do pamięci
- Przepustowość pamięci
- Rozmiar pamięci
- Wydajność instrukcji

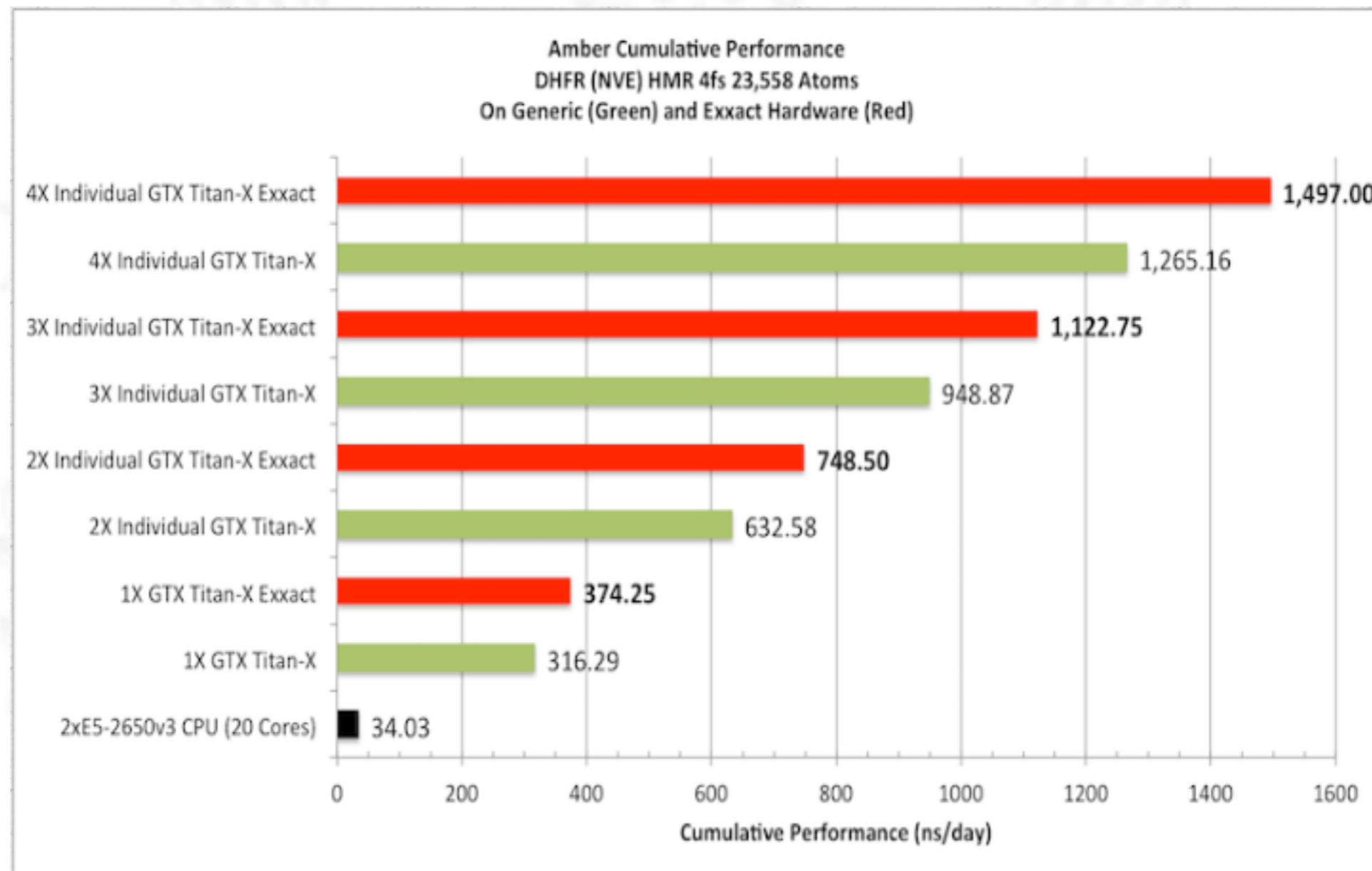
KOMUNIKACJA MIĘDZY WĘZŁAMI



KOMUNIKACJA MIĘDZY WĘZŁAMI



KOMUNIKACJA MIĘDZY WĘZŁAMI



PODSTAWOWE OGRANICZENIA WYDAJNOŚCI ALGORYTMÓW

- ~~Komunikacja między węzłami obliczeniowymi~~
 1. Latencja dostępu do pamięci
 2. Przepustowość pamięci
 3. Rozmiar pamięci
 4. Wydajność instrukcji