

Realizacja frontalnego solwera MES z wykorzystaniem technologii OpenCL

Paweł J. Wal

Wydział Inżynierii Metali i Informatyki Przemysłowej

Sesja Kół Naukowych AGH, 2014

Agenda

- 1 Solwer frontalny
 - Historia i inspiracja
- 2 Cele projektu
 - Główne założenia
 - Stworzenie rozwiązania uniwersalnego
- 3 Ogólny algorytm
 - Metoda wydzielania frontów rozwiązania
 - Równoległy wariant metody Gaussa
 - Przykład
- 4 Realizacja projektu
 - Problemy równoległości masowej
 - Przykład funkcjonowania jednego frontu
 - Paradygmat czarnej skrzynki
- 5 Badania wydajności
 - Optymalna liczba wątków dla badanych urządzeń
 - Przyspieszenie w zależności od globalnej ilości wątków
 - Skalowalność
- 6 Podsumowanie projektu
 - Wykonana praca
 - Kontynuacja projektu

Solwer frontalny

- Bruce Irons, 1970
- Motywacja jego pracy:
 - Relatywnie niewielka moc obliczeniowa
 - Ograniczona pamięć operacyjna (96kB)
 - Rosnący rozmiar problemów do rozwiązania
- Analogie z problematyką GPU
 - Ograniczony rozmiar pamięci operacyjnej
 - Kosztowny transfer między gospodarzem a urządzeniem

Cele projektu

Główne założenia

- Wykorzystanie ducha pracy Ironsa
 - Rozwiązanie współbieżne
 - Wykorzystanie możliwości równoległości masowej w GPGPU
- Wykorzystanie możliwości urządzeń obliczeniowych
 - Rozłożenie rozwiązania układu równań liniowych na szereg mniejszych, częściowo zależnych podproblemów

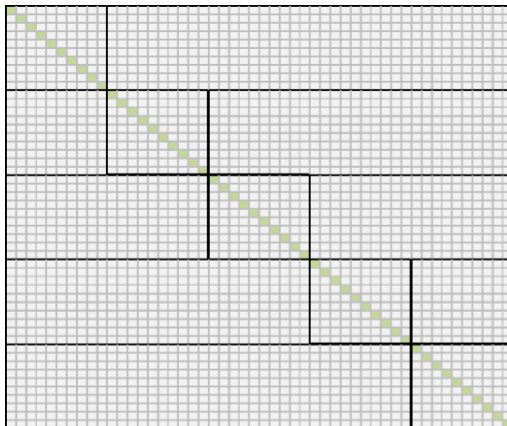
Cele projektu

Stworzenie rozwiązania uniwersalnego

- Czarna skrzynka
- Brak konieczności integracji z programem MES
- Możliwość rozwiązywania układów równań z różnych klas problemów
- Przenośność między systemami operacyjnymi
- Przenośność między urządzeniami obliczeniowymi

Algorytm

Metoda wydzielenia frontów rozwiązania



Algorytm

Równoległy wariant metody Gaussa

- Operacje elementarne na macierzach
 - Mnożenie i dodawanie wierszy
 - Zamiana wierszy
- Koncepcja mapy
 - Uniknięcie kosztownej, fizycznej zamiany wierszy
- Przywracanie formy macierzy schodkowej
 - Unikalny pierwszy wyraz niezerowy w wierszu
 - Mapa:
 - Pozwala na szybką weryfikację unikalności
 - Informuje względem którego wiersza prowadzić eliminację

Algorytm

Równoległy wariant metody Gaussa

	x1	x2	x3	x4	x5
→ r1	1	2	3	4	5
→ r2		1	2	4	5
→ r3			1	2	3
→ r4		2	3	4	5
→ r5			2	3	4

?	x1
?	x2
?	x3
?	x4
?	x5

	x1	x2	x3	x4	x5
→ r1	1	2	3	4	5
→ r2		1	2	4	5
→ r3			1	2	3
→ r4		2	3	4	5
→ r5			2	3	4

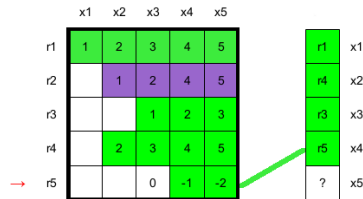
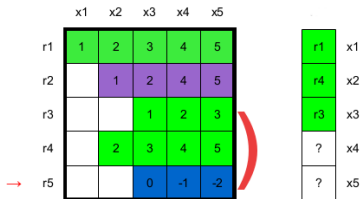
r1	x1
?	x2
?	x3
?	x4
?	x5

	x1	x2	x3	x4	x5
→ r1	1	2	3	4	5
→ r2		1	2	4	5
→ r3			1	2	3
→ r4		2	3	4	5
→ r5			2	3	4

r1	x1
r4	x2
r3	x3
?	x4
?	x5

Algorytm

Równoległy wariant metody Gaussa



Algorytm

Równoległy wariant metody Gaussa

→

	x1	x2	x3	x4	x5
r1	1	2	3	4	5
r2		0	0,5	2	2,5
r3			1	2	3
r4		2	3	4	5
r5			0	-1	-2

r1	x1
r4	x2
r3	x3
r5	x4
?	x5

→

	x1	x2	x3	x4	x5
r1	1	2	3	4	5
r2		0	0	1	1
r3			1	2	3
r4		2	3	4	5
r5			0	-1	-2

r1	x1
r4	x2
r3	x3
r5	x4
?	x5

→

	x1	x2	x3	x4	x5
r1	1	2	3	4	5
r2		0	0	0	-1
r3			1	2	3
r4		2	3	4	5
r5			0	-1	-2

WO

r1	x1
r4	x2
r3	x3
r5	x4
?	x5

	x1	x2	x3	x4	x5
r1	1	2	3	4	5
r2		0	0	0	-1
r3			1	2	3
r4		2	3	4	5
r5			0	-1	-2

WO

r1	x1
r4	x2
r3	x3
r5	x4
r2	x5

Realizacja projektu

Problemy równoległości masowej

- Wewnątrz części macierzy (frontu) wydzielane są grupy robocze
 - Wynika to z architektury urządzeń obliczeniowych
- Przedstawiony wcześniej algorytm działa w obrębie grupy
 - Pewność, iż nie ma konfliktujących wierszy w obrębie grupy
 - Co z konfliktami w obrębie całego frontu?
 - Co z konfliktami w obrębie całej macierzy?
- Rozwiązanie
 - Dodatkowy kernel na urządzeniu obliczeniowym
 - Dodatkowa faza przetwarzania na CPU

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	RHS	M1	M2	M3	M4
1	x	x	x														x				
2	x	x	x	x													x				
3	x	x	x	x	x												x				
4		x	x	x	x	x											x				
5				x	x	x	x	x									x				
6					x	x	x	x	x								x				
7						x	x	x	x	x							x				
8							x	x	x	x	x						x				
9								x	x	x	x	x					x				
10									x	x	x	x	x				x				
11										x	x	x	x	x			x				
12											x	x	x	x	x		x				
13												x	x	x	x	x	x				
14													x	x	x	x	x				
15														x	x	x	x				
16															x	x	x				

Realizacja projektu

Przykład funkcjonowania jednego frontu

- Konflikty w obrębie grup zostały rozwiązane
- Istnieją konflikty w obrębie frontu

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	x	x	x													
2		x	x	x												
3			x	x	x											
4				x	x	x										
5		x	x	x	x	x										
6			x	x	x	x	x									
7				x	x	x	x	x								
8					x	x	x	x	x							
9							x	x	x	x	x					
10								x	x	x	x	x				
11									x	x	x	x	x			
12										x	x	x	x	x		
13											x	x	x	x	x	
14												x	x	x	x	x
15													x	x	x	x
16														x	x	x

RHS	M1	M2	M3	M4
x	1			
x	2			
x	3	5		
x	4	6		
x	7			
x	8			
x		9		
x		10		
x		11		
x		10		
x				13
x				14
x				15
x				16
x				
x				

Realizacja projektu

Przykład funkcjonowania jednego frontu

- Konflikty w obrębie frontu zostały rozwiązane
- Poczyniono zmiany: czy nie powstały nowe konflikty w obrębie grup roboczych?

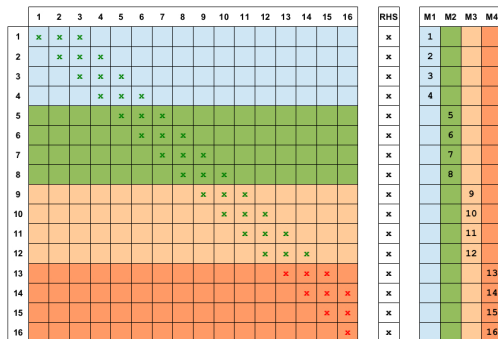
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	x	x	x													
2		x	x	x												
3			x	x	x											
4				x	x	x										
5					x	x	x	x								
6						x	x	x	x							
7							x	x	x	x	x					
8								x	x	x	x	x				
9									x	x	x	x	x			
10										x	x	x	x	x		
11											x	x	x	x	x	
12												x	x	x	x	x
13													x	x	x	x
14														x	x	x
15															x	x
16																x

RHS	M1	M2	M3	M4
x	1			
x	2			
x	3	5		
x	4	6		
x	7			
x	8			
x		9		
x		10		
x		11		
x		10		
x				13
x				14
x				15
x				16
x				
x				

Realizacja projektu

Przykład funkcjonowania jednego frontu

- Kernele wykonywane naprzemiennie dopóki drugi nie zgłosi zerowej ilości wykonanych operacji
- Kiedy wszystkie części skończą przetwarzanie analogiczna operacja jest powtarzana po stronie hosta



Realizacja projektu

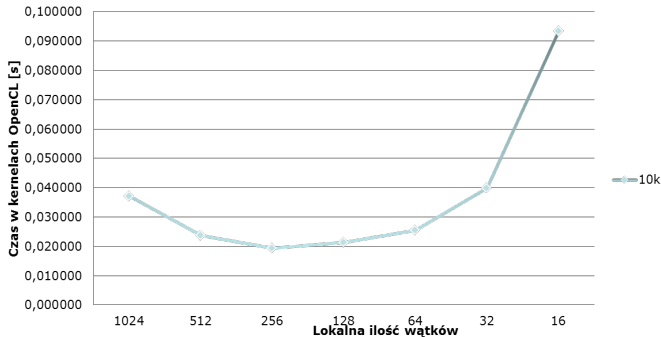
Paradygmat czarnej skrzynki

- Projekt zrealizowany jako biblioteka nagłówkowa
 - Nie wymaga dodatkowej kompilacji i linkowania ze strony użytkownika
 - Kompiluje się razem z kodem użytkownika
- Nie wymaga informacji o rozwiązywanym problemie
 - Nie integruje się z siatką MES
 - Może rozwiązywać dowolne problemy postawione jako układ równań liniowych
- Eksponuje wygodny interfejs
 - Dostarczany jest jeden wielofunkcyjny obiekt
 - Dostarczane są funkcje konwersji z macierzy użytkownika do wewnętrznych macierzy solwera
- Pozwala na wymianę kerneli już po skompilowaniu kodu

Badania wydajności

Optymalna liczba wątków dla badanych urządzeń

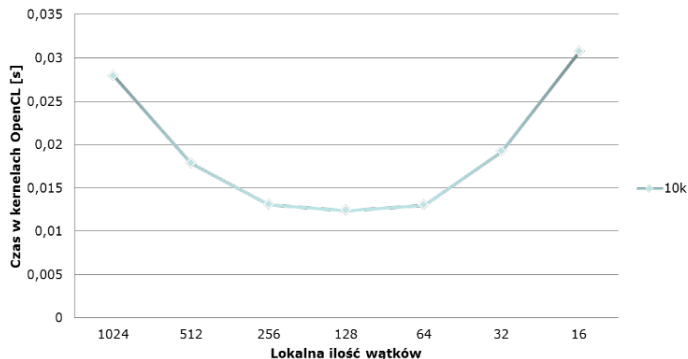
Czas w kernelach OpenCL od lokalnej ilości wątków na karcie Tesla M2090



Badania wydajności

Optymalna liczba wątków dla badanych urządzeń

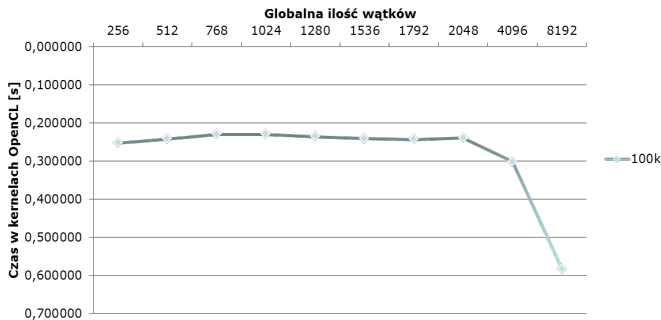
Czas w kernelach OpenCL od lokanej ilości wątków na procesorze Intel Xeon X5650



Badania wydajności

Przyspieszenie w zależności od globalnej ilości wątków

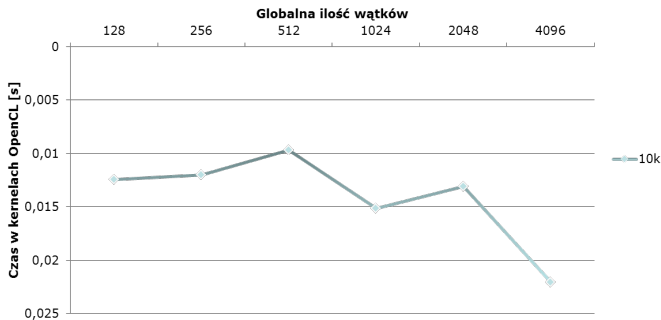
Czas w kernelach OpenCL od globalnej ilości wątków przy lokalnej ilości wątków 256 na karcie Tesla M2090



Badania wydajności

Przyspieszenie w zależności od globalnej ilości wątków

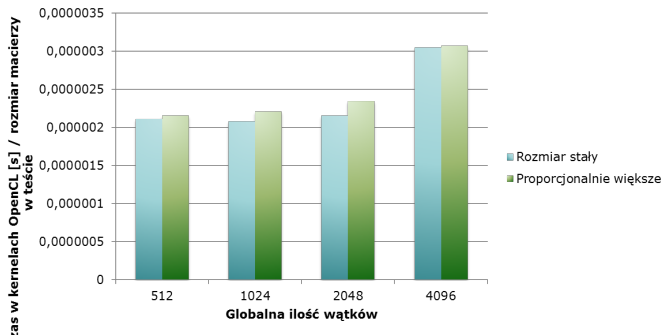
Czas w kernelach OpenCL od globalnej ilości wątków przy lokalnej ilości wątków 128 na procesorze Intel Xeon X5650



Badania wydajności

Skalowalność

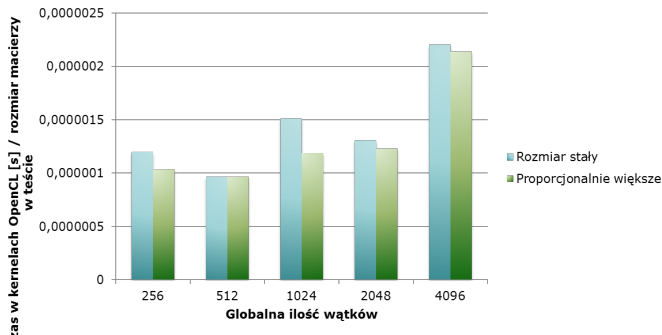
**Stosunek czasu rozwiązania do rozmiaru
macierzy dla optymalnej lokalnej ilości
wątków na karcie Tesla M2090**



Badania wydajności

Skalowalność

Stosunek czasu rozwiązania do rozmiaru macierzy dla optymalnej lokalnej ilości wątków na procesorze Intel Xeon X5650



Podsumowanie projektu

Wykonana praca

- Stworzono równoległy algorytm rozwiązywania układów równań w oparciu o metodę Gaussa
- Zaproponowano masowo równoległy, frontalny solwer MES z wykorzystaniem technologii OpenCL
- Oprogramowanie powstało zgodnie z paradygmatem czarnej skrzynki
 - Łatwe w przeniesieniu między systemami operacyjnymi i urządzeniami
 - Łatwe w implementacji w innych projektach

Podsumowanie projektu

Kontynuacja projektu

- Kontynuowana jest praca nad projektem
 - Wykorzystanie wielu urządzeń na jednym węźle obliczeniowym
 - Rozproszenie obliczeń na wiele węzłów