KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Zaawansowane zastosowania kart graficznych	**1010512321010519522**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**1 / 2**
Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
Technologie przetwarzania danych	**polski**	**obieralny**

Stopień studiów:	**Forma studiów** (stacjonarna/niestacjonarna)
II stopień	**niestacjonarna**

Godziny	Liczba punktów
Wykłady: **16** Ćwiczenia: **-** Laboratoria: **16** Projekty/seminaria: **-**	**3**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
kierunkowy	**z danego kierunku**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**
nauki techniczne	**3 100%**

Odpowiedzialny za przedmiot / wykładowca:

dr inż. Witold Andrzejewski
email: Witold.Andrzejewski@cs.put.poznan.pl
tel. 61 6652965
Instytut Informatyki
ul. Piotrowo 2, 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę o architekturze komputerów, o programowaniu obiektowym i proceduralnym oraz o metodach oceny złożoności algorytmów.
2	**Umiejętności:**	Powinien posiadać umiejętność rozwiązywania podstawowych problemów algorytmicznych, programowania w jezyku C/C++, oraz umiejętność pozyskiwania informacji ze wskazanych źródeł.
3	**Kompetencje społeczne**	Powinien również rozumieć konieczność poszerzania swoich kompetencji i mieć gotowość do podjęcia współpracy w ramach zespołu. Ponadto w zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu:

1. Przekazanie studentom podstawowej wiedzy z programowania procesorów kart graficznych (GPU), w zakresie:

a. Wybranych aspektów architektury sprzętowej popularnych kart graficznych.

b. Logicznego modelu współbieżności kart graficznych w kontekście API CUDA i OpenCL.

c. Optymalizacji wykonania programów na kartach graficznych.

d. Wybranych podstawowych algorytmów równoległych, w tym: map, reduce, compact, sort, scan, search, generowania ntych wyrazów ciągów.

e. Wybranych struktur danych, w tym: macierzy, tablic hashowych i drzew CSS i algorytmów ich przetwarzania.

f. Oceny złożoności algorytmów równoległych (w tym model PRAM) i ich związku z faktyczną złożonością programów wykorzystujących karty graficzne.

g. Zastosowania kart graficznych do rozwiązywania praktycznych problemów związanych z przetwarzaniem i wizualizacją danych.

2. Rozwijanie u studentów umiejętności:

a. Rozwiązywania problemów algorytmicznych z ukierunkowaniem na zrównoleglenie operacji przetwarzania danych.

b. Optymalizacji programów wykorzystujących procesory kart graficznych.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma uporządkowaną, podbudowaną teoretycznie wiedzę ogólną w zakresie algorytmów równoległych i ich złożoności. - [K_W4]

2. ma podbudowaną teoretycznie szczegółową wiedzę związaną z wybranymi zagadnieniami z zakresu informatyki, takimi jak: algorytmy równoległe i ich ocena złożoności, architektura kart graficznych, API i paradygmaty programowania programów równoległych wykorzystujących GPU. - [K_W5]

3. ma wiedzę o trendach rozwojowych i najistotniejszych nowych osiągnięciach w informatyce, w szczególności w dziedzinie sprzętowego wsparcia dla obliczeń równoległych. - [K_W6]

4. zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu złożonych zadań inżynierskich związanych z projektowaniem i optymalizacją niskopoziomową programów wykorzystujących procesory kart graficznych. - [K_W8]

Umiejętności:

1. potrafi pozyskiwać informacje z literatury, baz danych oraz innych źródeł (w języku ojczystym i angielskim), integrować je, dokonywać ich interpretacji i krytycznej oceny, wyciągać wnioski oraz formułować i wyczerpująco uzasadniać opinie - [K_U1]

2. potrafi określić kierunki dalszego uczenia się i zrealizować proces samokształcenia - [K_U5]

3. potrafi wykorzystać do formułowania i rozwiązywania zadań inżynierskich i prostych problemów badawczych metody analityczne oraz eksperymentalne - [K_U9]

4. potrafi - przy formułowaniu i rozwiązywaniu zadań inżynierskich - integrować wiedzę z różnych obszarów informatyki. - [K_U10]

5. potrafi formułować i testować hipotezy związane z problemami inżynierskimi i prostymi problemami badawczymi. - [K_U12]

6. potrafi ocenić przydatność i możliwość wykorzystania nowych osiągnięć (metod i narzędzi) oraz nowych produktów informatycznych w dziedzinie programowania równoległego.. - [K_U13]

7. potrafi ocenić złożoność algorytmów równoległych - [K_U16]

8. potrafi ocenić przydatność metod i narzędzi służących do rozwiązania zadania inżynierskiego, polegającego na implementacji algorytmu wykorzystującego procesory kart graficznych uwzględniając ich ograniczenia.. - [K_U24]

9. potrafi - stosując m.in. koncepcyjnie nowe metody - rozwiązywać złożone zadania informatyczne, w tym zaprojektować nowe algorytmy równoległe wykorzystujące GPU. - [K_U25]

10. potrafi wybrać język programowaniai API odpowiednie do danego zadania związanego z zaprogramowaniem karty graficznej.. - [K_U26]

Kompetencje społeczne:

1. rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe - [K_K1]

2. zna przykłady i rozumie przyczyny wadliwie działających systemów informatycznych, które doprowadziły do poważnych strat finansowych - [K_K4]

3. potrafi odpowiednio określić priorytety służące realizacji określonego przez siebie lub innych zadania - [K_K6]

Sposoby sprawdzenia efektów kształcenia

Ocena formująca

a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez odpowiedzi na pytania dotyczące materiału omówionego na poprzednich wykładach

b) w zakresie laboratoriów / ćwiczeń weryfikowanie założonych efektów kształcenia realizowane jest przez: okresową ocenę przygotowania studenta do poszczególnych sesji zajęć laboratoryjnych (sprawdzian wejściowy) oraz ocenę umiejętności związanych z realizacją ćwiczeń laboratoryjnych

Ocena podsumowująca:

a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez ocenę wiedzy i umiejętności wykazanych na zaliczeniu pisemnym o formie testu jednokrotnego wyboru składającego się z ok. 30 pytań, łączna liczba punktów za prawidłowe odpowiedzi: 30, minimalna liczba punktów umożliwiających zaliczenie: 16

b) w zakresie laboratoriów / ćwiczeń weryfikowanie założonych efektów kształcenia realizowane jest przez ocenę i obronę przez studenta sprawozdania z realizacji projektu,

Uzyskiwanie punktów dodatkowych za aktywność podczas zajęć, a szczególnie za:

- efektywność zastosowania zdobytej wiedzy podczas rozwiązywania zadanego problemu,
- uwagi związane z udoskonaleniem materiałów dydaktycznych,
- wskazywanie trudności percepcyjnych studentów umożliwiające bieżące doskonalenia procesu dydaktycznego.

Treści programowe

Program wykładu obejmuje następujące zagadnienia:

Wykład wstępny przedstawiający motywację stojącą za wykorzystaniem kart graficznych do obliczeń. Dodatkowo: przedstawienie różnych rozwiązań technologicznych umożliwiających przetwarzanie równoległe, wprowadzenie podstawowych pojęć stosowanych w kolejnych wykładach. Omówienie podstaw modelu programistycznego wykorzystywanego w programach wykorzystujących procesory kart graficznych.

Wykład omawiający architekturę sprzętową kart graficznych. Omówienie związków pomiędzy architekturą sprzętową a modelem programistycznym. Przedstawienie trików bitowych.

Omówienie hierarchii pamięci i metod wydajnego dostępu dla różnych architektur sprzętowych. Przedstawienie przykładowych metod optymalizacji wykorzystujących różne poziomy pamięci operacyjnej. Omówienie rozwiązań pozwalających na zapewnienie równoległego przesyłu danych pomiędzy komputerem a kartą graficzną i wykonywania obliczeń.

Wprowadzenie podstaw teoretycznych oceny złożoności algorytmów równoległych. Omówienie obliczeń w modelu maszyny PRAM.

Podstawowe algorytmy wykorzystywane podczas konstrukcji bardziej złożonych rozwiązań, w tym: map, gather, scatter, reduce, compact, search, scan. Szczegółowe omówienie algorytmu scan. Ocena złożoności wprowadzonych algorytmów w ich wersji sekwencyjnej i równoległej.

Tematyka algorytmów równoległych. Szczegółowe omówienie algorytmów : compact i reduce oraz algorytmów przeszukiwania i sortowania danych. Ocena złożoności przedstawionych algorytmów.

Algorytmy generowania elementów w sekwencji kombinacji, permutacji bądź w wyniku iloczynu kartezjańskiego wraz z ich przykładowymi zastosowaniami.

Problemy związane z wykorzystywaniem standardowych struktur danych. Omówienie wydajnych metod przetwarzania macierzy (w tym macierzy rzadkich) oraz tablic haszowych. Oszacowanie złożoności algorytmów przetwarzania tych struktur.

Praktyczne zastosowania GPU do przetwarzania danych relacyjnych. Omówienie algorytmów selekcji, projekcji i połączeń. Ocena złożoności, zalety i wady wykorzystania GPU do przetwarzania danych relacyjnych.

Praktyczne zastosowania GPU, tym razem w grafice komputerowej. W ramach wykładów omawiane są podstawy matematycznych grafiki 3D oraz algorytmów detekcji przecięcia promienia z różnymi figurami/bryłami geometrycznymi, podstawowe modele oświetlenia i równoległe algorytmy wizualizacji tekstur trójwymiarowych oraz raytracingu.

Praktyczne zastosowania GPU w eksploracji danych. Pokazanie algorytmów grupowania i odkrywania zbiorów częstych.

Ćwiczenia realizowane są przez studentów samodzielnie za wyjątkiem 3 ostatnich spotkań na których studenci w zespołach dwuosobowych będą realizować projekty zaliczeniowe. Program laboratorium obejmuje następujące zagadnienia:

Zapoznanie się z CUDA. Ćwiczenia związane z prawidłową konstrukcją siatki obliczeń. Implementacja prostych algorytmów równoległych. Omówienie sposobów debugowania programów wykorzystujących karty graficzne. Kontynuacja ćwiczeń z prostymi algorytmami. Realizacja ćwiczeń w których należy uwzględnić komunikację i synchronizację wątków. Testowanie wydajności dostępu do różnych dostępnych rodzajów pamięci operacyjnej na karcie graficznej. Zapoznanie się z biblioteką thrust. Omówienie podstawowych koncepcji stojących za API tej biblioteki. Przedstawienie podstawowych algorytmów w niej zaimplementowanych. Laboratorium 7. Omówienie biblioteki CURAND. Ćwiczenia wykorzystujące bibliotekę thrust. Budowa złożonych algorytmów z podstawowych algorytmów takich jak: map, reduce, gather, scatter, search, scan.

Cześć wymienionych wyżej treści programowych realizowana jest w ramach pracy własnej studenta.

Metody dydaktyczne:

1. wykład: prezentacja multimedialna, prezentacja ilustrowana przykładami podawanymi na tablicy, rozwiązywanie zadań.
2. ćwiczenia laboratoryjne: rozwiązywanie zadań, ćwiczenia praktyczne, pokaz multimedialny, demonstracja.

Literatura podstawowa:

1. Jason Sanders, Edward Kandrot: CUDA By Example, Addison-Wesley 2011.

2. Rob Farber: CUDA Application Design and Development, Morgan-Kauffman, 2011

3. NVIDIA: CUDA C Programming Guide: http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html

4. NVIDIA: CUDA C Best Practices Guide: http://docs.nvidia.com/cuda/cuda-c-best-practices-guide/index.html

5. NVIDIA: OpenCL Jumpstart Guide: http://www.cs.cmu.edu/afs/cs/academic/class/15668-s11/www/cuda-doc/OpenCL_Jumpstart_Guide.pdf

6. NVIDIA: OpenCL Best Practices Guide: http://www.nvidia.com/content/cudazone/CUDABrowser/downloads/papers/NVIDIA_OpenCL_BestPracticesGuide.pdf

Literatura uzupełniająca:

1. Jianlong Zhong* and Bingsheng He. Medusa: Simplified Graph Processing on GPUs. Accepted by TPDS 2013: IEEE Transactions on Parallel and Distributed System

2. Bingsheng He and Jeffrey Xu Yu. High-Throughput Transaction Executions on Graphics Processors. Proceedings of Very Large Data Bases (VLDB) 2011

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)

1. udział w zajęciach laboratoryjnych	16
2. przygotowanie do ćwiczeń laboratoryjnych:	8
3. udział w konsultacjach (mogą być realizowane drogą elektroniczną) związanych z realizacją procesu kształcenia, w szczególności ćwiczeń laboratoryjnych / projektu	3
4. przygotowanie do sprawdzianów / kolokwium	8
5. udział w wykładach	16
6. zapoznanie się ze wskazaną literaturą / materiałami dydaktycznymi (10 stron tekstu naukowego = 1 godz.), 100 stron	10
7. omówienie wyników zaliczenia	2
8. przygotowanie do zaliczenia wykładów 10 godz. i +2 godz. udział w kolokwium zaliczeniowym	12

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	75	3
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	39	1
Zajęcia o charakterze praktycznym	32	1