Proponowane projekty zaliczeniowe

Uwagi ogólne:

Idealny program rozwiązujący problem powinien zawierać część obliczeniową napisaną w C/C++. Ten program powinien zapisywać dane do pliku tekstowego. Część wizualizacyjna powinna być napisana w Pythonie, pobierać dane z pliku tekstowego i następnie tworzyć odpowiednie wykresy/wizualizację.

Zaliczenie projektu:

Zaliczenie projektu na zajęciach będzie składało się z 3 części:
1)Prezentacja max. 15 minut w której grupa omawia problem, który modeluje/symuluje/wizualizuje. Należy wyjaśnić słuchaczom teorię leżącą u podstaw problemu i jego rozwiązanie.
2) Prezentacja programu. Należy uruchomić program/y i zaprezentować jego działanie.
3)„Przejście przez kod" (tzw. Code Walkthrough). Omówienie kodu ze zwróceniem uwagi na ciekawe fragmenty.

Grupa również odpowiada na pytania od publiczności.

Kolejna ocena zostanie wystawiona za raport, który zawiera rozszerzoną wersję prezentacji(prezentacja teorii na której opiera się problem, opis sposobu rozwiązania problemu i ogólny opis dołączonych programów, a także podsumowanie) oraz dołączone do niego programy oraz prezentację. Prezentacja, raport i programy zostaną zmieszczona na stronie internetowej do wglądu przez inne grupy. Proszę je przesyłać w paczce tar.gz.

W katalogu programu powinien znajdować się plik Readme.txt zawierający informację o autorach, opis oraz sposób uruchamiania. Każdy program musi być opatrzony plikiem Makefile. Program powinien się uruchamiać i prezentować wyniki po uruchamianiu komendą *make run*.

Trzecia ocena zostanie wystawiona zwycięzcom głosowania na najlepszą prezentację i rozwiązanie problemu.

Wybór projektu:

Każda grupa pracuje nad innym projektem. Wybór projektu odbywa się wg zasady pierwszeństwa - decyduje data zgłoszenia/listu ze zgłoszeniem grupy. Projekty z poniższej listy są tylko **sugestiami.** Zalecane są projekty, które są powiązane z Państwa pracami licencjackimi/magisterskimi lub hobby. Takiego programu mogą Państwo później użyć w przy pisaniu pracy. Mogą to być również rozwinięcia tematów omawianych na zajęciach. Wszystkie projekty o poziomie skomplikowania podobnym do poniższych są możliwe. Proszę przesłać prowadzącemu opis proponowanego projektu i zamierzony efekt końcowy. Prowadzący akceptuje projekt i ustala szczegóły lub odrzuca go. Dlatego proszę nie rozpoczynać pracy nad projektem przed jego zaakceptowaniem – list o potwierdzeniu.

Projekty

(W nawiasach podano maksymalną liczbę osób w grupie)

1) **Model Isinga 2D**
Proszę napisać program symulujący 2D lub 3D model Isinga z periodycznymi warunkami brzegowymi. Proszę zastosować algorytm Metropolisa dla zespołu kanonicznego. Należy pamiętać o termalizacji. Siatka spinów powinna być rysowana, np. w postaci strzałek umieszczonych w określonych węzłach. Symulację powinien wykonywać program C/C++, a wizualizację program w Pythonie. (3 os.)

[1]Dieter Heerman „*Podstawy symulacji komputerowych w fizyce”* WNT 1997
[2] http://www.physics.udel.edu/~jim/PHYS460_660_13S/statmech/The%20Ising%20model%20-%20simulation.htm

2) **Ewolucja**
Proszę zaimplementować grę w ewolucję. Wbrew pozorom jest to symulacja złożonego układu fizycznego i model układu biologicznego. Opis i kod w języku Lisp jest dostępny w [1]. Proszę odtworzyć wykres z [1] ilustrujący podział osobników na „wolne” i „szybkie”. (4 os.)

[1] C. Barski „Land of Lisp”, No Starch Press. (rozdział 10).

3) **Równanie przewodnictwa cieplnego w 2D przy użyciu przetwarzania równoległego MPI**
Proszę napisać program do symulacji 2D rozwiązań równania przewodnictwa cieplnego w oparciu o MPI [1]. Kod symulujący powinien być napisany w C/C++ lub w Fortranie, natomiast kod wizualizujący w Pythonie. (3 os.)

[1]https://computing.llnl.gov/tutorials/parallel_comp/#ExamplesHeat
[2]Maciej Matyka „*Symulacje komputerowe w fizyce”,* rozdz. 3 Helion 2011

4) **Perkolacja na przykładzie pożarów lasów**
Proszę napisać model obrazujący rozprzestrzenianie się pożaru lasu przy użyciu perkolacji. Symulacja powinna być wykonana estetycznie, np. drzewa powinny być podobne do drzew, a nie do stożków. Należy odtworzyć wyniki z publikacji [2], a szczególnie wykres 4.0.4. Część symulacyjną proszę napisać w C/C++, a wizualizacyjną w Pythonie. (3 os.)
[1]Materiały do wykładu.
[2]http://lucc.ess.inpe.br/lib/exe/fetch.php?media=percolation_experiments_on_a_2d_lattice.pdf

5) **Odwzorowanie logistyczne i powstawanie chaosu**
Proszę opracować wizualizację odwzorowania logistycznego [1], rozdział 2 pt „Bifurcation”. Należy omówić zachowanie odwzorowania logistycznego przy zmianie parametru μ. Należy skonstruować program, który pokazuje graficzną iterację [2] (slajd 65-67, „Iteration for kindergarden”). Następnie proszę skonstruować program tworzący diagram bifurkacyjny zamieszczony w [3] na slajdzie 26 – 28. Proszę wyjaśnić na czym polega chaos i napisać program, który pokazuje czułość odwzorowania logistycznego na warunku początkowe w obszarze chaotycznym. (3 os.)

[1]Shlomo Sternberg „Dynamical Systems”:
http://www.math.harvard.edu/~shlomo/docs/dynamical_systems.pdf
[2]http://www.math.harvard.edu/library/sternberg/slides/lec2.pdf
[3]http://www.math.harvard.edu/library/sternberg/slides/1180904.pdf

6) **Fraktale Newtona**
Proszę napisać program ilustrujący tworzenie fraktali Newtona. Program generujący fraktal proszę napisać w C++, natomiast program wizualizujący w Pythonie. Sposób generacji opisany jest w [2] (slajd 57). (3 os.)

[1]https://en.wikipedia.org/wiki/Newton_fractal
[2]http://www.math.harvard.edu/library/sternberg/slides/lec1.pdf

7) **Układ Lotki-Volterry dla większej niż 2 liczby gatunków**
Proszę omówić model Lotki-Volterry dla WIĘKSZEJ NIŻ 2 liczby gatunków, tzw. „food chain equations” [1] (slajd 40). Proszę dopasować parametry tak, aby otrzymać sensowne rozwiązania i proszę wyjaśnić ich sens. (3 os.)

[1] http://www.math.harvard.edu/library/sternberg/slides/11809LV.pdf

8) **Oryginalny algorytm Google - „miniGoogle”**
Proszę zamodelować strukturę/graf (liczba węzłów > 3) przedstawiającą powiązane odnośnikami strony internetowe. Strony powinny mieć przykładową zawartość. Tę strukturę należy przedstawić graficznie. Następnie proszę zaimplementować prosty algorytm wyszukiwania oparty na algorytmie Google [1] w takiej prostej sieci. (3 os.)

[1] S. Sternberg „Dynamical systems”(rozdział „The Perron-Frobenius Theorem” lub odpowiedni wykład z transparencji) http://www.math.harvard.edu/library/sternberg/

9) **Obliczenia symboliczne w C/C++**
Proszę omówić możliwość obliczeń symbolicznych w C/C++ przy użyciu SymbolicC++ oraz GiNac [2]. Programy powinien zawierać przykłady użycia tych bibliotek, a raport sposób instalacji bibliotek, przykładowe programy oraz wybrane funkcje do obliczeń symbolicznych. (3 os.)

[1]https://en.wikipedia.org/wiki/SymbolicC%2B%2B
[2]http://issc.uj.ac.za/symbolic/symbolic.html
[3] http://www.ginac.de/

10) **Całkowanie Monte Carlo przy użyciu klasy Tfoam z Pakietu ROOT**
Proszę omówić działanie adaptacyjnego integratora Tfoam [1] z pakietu ROOT [2]. Należy omówić sposób instalacji biblioteki ROOT, a następnie przykład użycia foam [3] do całkowania Monte Carlo dowolnej funkcji. Programy przykładowe NIE powinny być skryptami(jak w [3]), lecz programem kompilowanym z poziomu Makefile. (2 os.)

[1]http://arxiv.org/abs/physics/0203033
[2] https://root.cern.ch/

[3]https://root.cern.ch/root/html/tutorials/foam/index.html

11) **Serwer analizy danych w trybie online**
Proszę zaprojektować i zaimplementować aplikację do analizy wybranych danych i prezentacji ich na stronie internetowej. Należy zastosować Python, R oraz technologie sieciowe Apache, JavaScritp. Należy wzorować się na metodzie z [1] opisującej internetową aplikację do analizy danych komet. Do danych proszę dostosować odpowiednie wykresy. Grupa może zaimplementować pobieranie danych bezpośrednio z Internetu. (4 os.)

[1] http://www.admin-magazine.com/Archive/2015/25/Data-Analysis-with-R-and-Python

12) **Rozpoznawanie obrazów na przykładzie programu OCR do hieroglifów**
Proszę napisać program rozpoznający kilka wybranych hieroglifów i „tłumaczący je" na język polski lub angielski [1]. Program powinien pobierać obraz z hieroglifami (np. zeskanowany obrazek), a następnie rozpoznawać wybrane znaki i je wypisywać. W celu rozpoznawania hieroglifów można użyć biblioteki do uczenia maszynowego scikit-image [2], w szczególności proszę zwrócić uwagę na [3]. Przykładowym rozwiązaniem byłoby oznaczenie regionów na obrazku, w których znajdują się poszczególne hieroglify [4], a nastąpienie w każdym regionie próba odczytania hieroglif [3]. W celu poprawienia kontrastu można spróbować [5]. (4 os.)

[1] Wprowadzenie do hieroglifów egipskich:
https://en.wikipedia.org/wiki/Egyptian_hieroglyphs
http://www.ancientegyptonline.co.uk/hieroglyphs-tutorial.html
http://www.egyptianhieroglyphs.net/egyptian-hieroglyphs/
[2] http://scikit-image.org/
[3] http://scikit-image.org/docs/dev/auto_examples/plot_template.html#example-plot-template-py
[4] http://scikit-image.org/docs/dev/auto_examples/plot_label.html
[5] http://scikit-image.org/docs/dev/auto_examples/plot_threshold_adaptive.html

13) **Szyfrowanie transmisji sieciowej przy użyciu chaosu**
Zadanie polega na napisaniu programu w Pythonie szyfrującego i deszyfrującego wiadomości podczas transmisji przez sieć przy użyciu metody reverse interval mapping [1](str. 49) oraz varaible bit length encoding [1](str. 51) – sposób komunikacji opisany jest w [1] w rozdziale 1.18. Metody te wykorzystują odwzorowanie w którym pojawia się chaos. Dzięki temu dla osoby podsłuchujące widzą jedynie losowy/chaotyczny ciąg liczb/znaków. W prezentacji należy omówić teorię(chaos w układach dynamicznych) [2], która leży u podstaw tych metod kodowania, a następnie zaprezentować program w Pythonie, umożliwiający rozmowę pomiędzy dwoma użytkownikami połączonymi przez sieć używającymi tego szyfrowania. Szkic funkcji szyfrujących i deszyfrujących w Javie znajduje się w [1], rozdział 1. Komunikacja sieciowa może odbywać się w obrębie jednego komputera poprzez sieciową pętlę zwrotną(loopback – adres IP 127.0.0.1) [3] – rozmowa pomiędzy dwoma terminalami na tym samym komputerze.(4 os.)

[1] Willi-Hans Steeb „The Nonlinear Workbook", WorldScientific
[2] S. Sternberg „Dynamical systems"(rozdziały opisujące odwzorowanie logistyczne - „logistic map") http://www.math.harvard.edu/library/sternberg/
[3]
http://www.bogotobogo.com/python/python_network_programming_tcp_server_client_chat_server_chat_client_select.php

14) **Krzywa Hilberta przy pomocy programu grafika żółwia w języku symbolicznym Scheme**
Krzywa Hilberta [1] jest krzywą pokrywającą dowolny kwadrat na płaszczyźnie. Pokazuje ona, iż liczba punktów na linii jest taka sama jak liczba punktów na płaszczyźnie. Powstaje ona w wyniku rekurencyjnego stosowania ustalonej reguły rysownia elementu bazowego. W celu narysowania tej i innych krzywych należy zaimplementować program realizujący grafikę żółwia – język LOGO [2]. Program rysujący fragment krzywej Kocha przy użyciu języka Scheme[3] podany jest tutaj [4]. Program ten wykorzystuje interpreter języka Scheme o nazwie Guile, który jest osadzony w programie języka C. Należy uruchomić ten przykład, następnie dodać wszystkie potrzebne funkcje do języka Scheme. W ostatnim kroku należy podać kod w języku Scheme, który rysuje krzywą Hilberta. Wskazówki można znaleźć w książce [5]. (4 os.)

[1] https://en.wikipedia.org/wiki/Hilbert_curve
[2] https://pl.wikipedia.org/wiki/Logo_%28j%C4%99zyk_programowania%29
[3] Abelson Harold, Sussman Gerald Jay, Sussman Julie „Struktura i interpretacja programów komputerowych", Helion
[4] http://www.gnu.org/software/guile/docs/guile-tut/tutorial.html
[5] Harold Abelson, Andrea di Sessa „Geometria Żółwia", WNT

15) **Całki po trajektoriach w klasycznych problemach kwantowomechanicznych**
Całki po trajektoriach do mechaniki kwantowej zostały wprowadzone prze Ryszarda Feynmana jako alternatywny opis mechaniki kwantowej [1]. Nie wszystkie ich aspekty mają jeszcze ścisłe matematyczne dowody, jednak są niezwykle użytecznym narzędziem obliczeniowym w wielu skomplikowanych problemach nierozwiązywalnych analitycznie. Zadaniem będzie napisanie programu w Pythonie i C/C++ rozwiązującego kwantowy oscylator harmoniczny przy użyciu całek po trajektoriach. Rozwiązanie należy porównać z wynikiem dokładnym(poziomy energetyczne, trajektorie) oraz przetestować zależność od parametrów i ustawień symulacji. Wprowadzenie do problemu znajduje się w klasycznym artykule [2], natomiast bardzo podstawowy kurs pisania tego typu programów można znaleźć tutaj [3]. Rozszerzenie zaganiania można znaleźć w [4],[5]. (2 os.)

[1] Ramamurti Shankar „Mechanika kwantowa", PWN
[2] M. Creutz, B. Friedmann „A Statistical Approach to Quantum Mechanics", Annals of Physics **132** 427-462 (1981)
[3] http://www.itkp.uni-bonn.de/~urbach/bnd/
[4] R. Feynman „Quantum Mechanics and Path Integrals", Dover
[5] M. Creutz „Kwarki, gluony i sieci", PWN

%%%