

# Dokumentacja

## Systemy równoległe i rozproszone

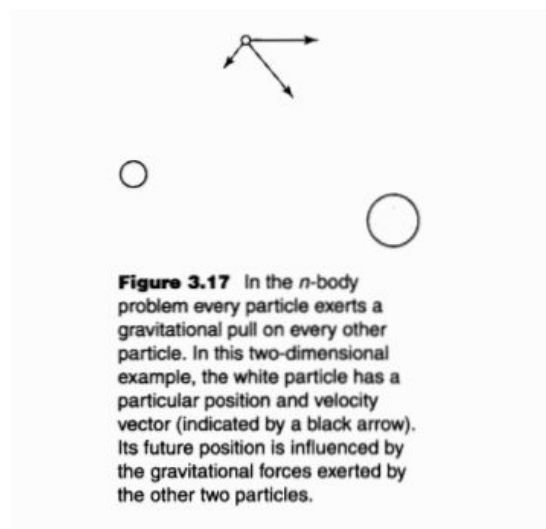
### Projekt 1 – Aplikacja równoległa MPI

#### Temat: Problem oddziaływania wielu ciał (N-body problem)

Karol Rojek, Michał Zbrożek

#### 1. Wprowadzenie teoretyczne i cele

Celem projektu było stworzenie aplikacji równoległej w technologii MPI, która będzie rozwiązywać problem wielu ciał (N-body problem). Jest to zagadnienie mechaniki klasycznej polegające na obliczaniu toru ruchu ciał pewnego układu znając ich masy, prędkości i położenia początkowe. Zależą one od działających w układzie sił grawitacyjnych pomiędzy ciałami. Uproszczony schemat takiego układu 2D przedstawia Rys. 1. Aplikacja będzie obliczać analogiczny problem dla trzech wymiarów.



Rys. 1 - Uproszczony schemat problemu wielu ciał 2D. Źródło: M. J. Quinn, *Parallel Programming in C with MPI and OpenMP*, 2001

Aby znaleźć położenia wszystkich ciał należy obliczyć działającą nań siłę wypadkową zgodnie z prawem powszechnego ciążenia (1).

$$(1) \quad F^i = G \frac{m_1 m_2}{r^2} e^i,$$

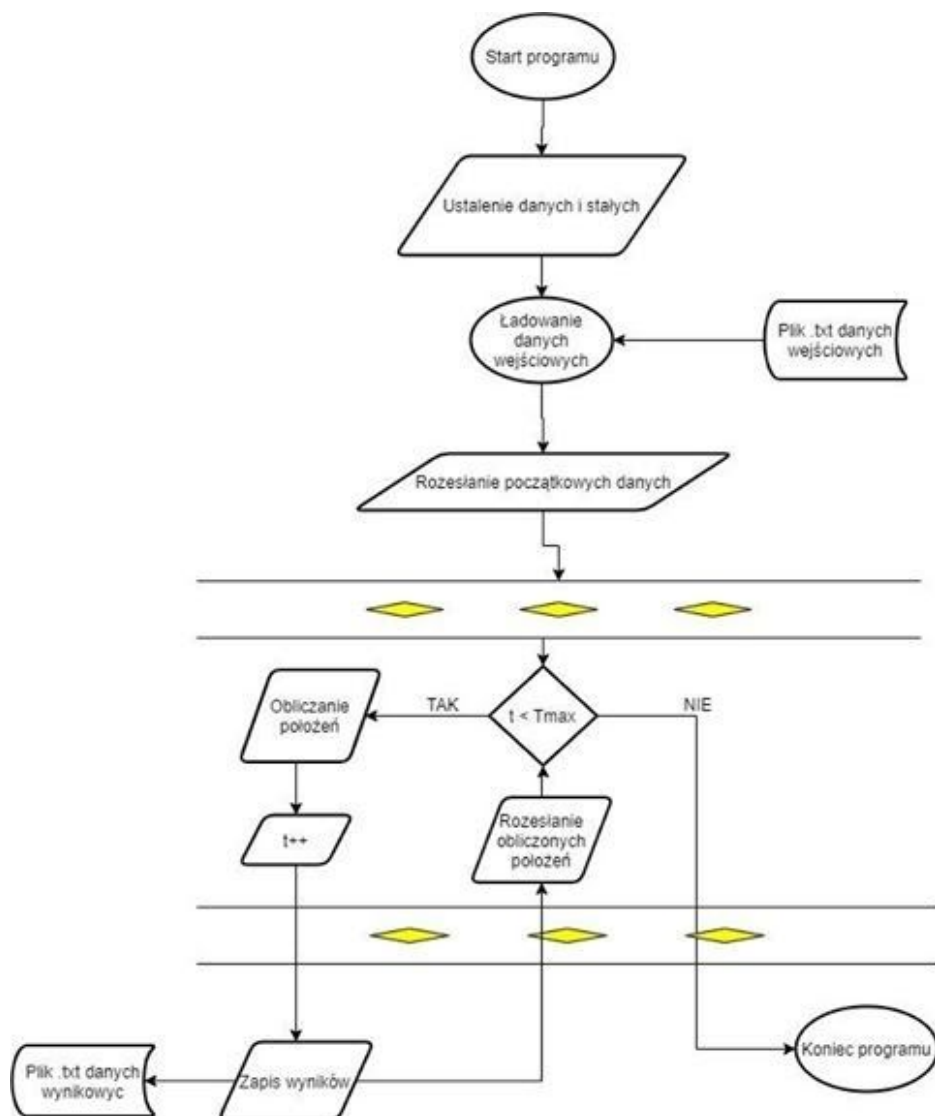
Następnie na podstawie tej siły obliczane jest przyspieszenie ciała, modyfikowany jest jego wektor prędkości, a na końcu położenie.

Ponieważ celem jest stworzenie aplikacji równoległej aby przyspieszyć wykonywane obliczenia, konieczna jest modyfikacja algorytmu problemu. Obliczenie nowego położenia danego ciała wymaga znajomości danych wszystkich pozostałych ciał - każdy z procesów musi posiadać cały zestaw danych, nie można go podzielić. Procesy mogą natomiast obliczać nowe pozycje tylko pewnego wybranego podzbioru ciał, a następnie wymieniać się wynikami. W ten sposób po jednym wykonaniu pętli algorytmu każdy z procesów zawiera dane o położeniu wszystkich ciał w następnym kroku czasowym.

## 2. Budowa i opis programu

### Algorytm działania

Aplikacja obliczająca problem n-ciał stworzona jest w języku C++ z wykorzystaniem biblioteki mpi.h w implementacji MPICH. Schemat blokowy (Rys. 2) przedstawia uproszczony przebieg algorytmu.



Rys. 2 - Schemat blokowy aplikacji równoległej problemu n-ciał.

Program rozpoczyna działanie od ustawienia zmiennych potrzebnych do realizacji i określających działanie, takich jak krok czasowy i czas symulacji, oraz stałych matematycznych. Zainicjalizowane zostają też mechanizmy MPI za pomocą funkcji *MPI\_Init*, *MPI\_Comm\_rank* i *MPI\_comm\_size*.

Następnie proces główny programu odczytuje ilość wpisów zawartych w pliku danych wejściowych oraz rozsyła tą wartość do wszystkich procesów poleceniem *MPI\_Bcast*. Na tej podstawie każdy proces oblicza dla których ciał będzie prowadził symulację. Kolejnym krokiem jest wczytanie oraz rozesłanie samych danych wejściowych.

W każdym z węzłów wywoływana jest pętla symulacji, która oblicza siły wypadkowe, przyspieszenia, oraz nowe położenia ciał. Krokiem czasowym pętli jest 1 minuta. Dane o położeniu są następnie przesyłane do wszystkich pozostałych procesów, a co 10 minut po otrzymaniu całego zestawu danych proces główny zapisuje je do pliku tekstowego. W ten sposób w pliku wyjściowym tworzy się historia położzeń każdego z ciał, czyli tak zwana efemeryda.

Pętla kończy się po upływie jednego miesiąca, a po jej zakończeniu program zwalnia zasoby i kończy działanie funkcją *MPI\_Finalize*.

### **Funkcje zdefiniowane w programie**

#### *getDataSize*

Oblicza oraz zwraca ilość wpisów danych zawartych w pliku wejściowym.

#### *readData*

Czyta plik z danymi wejściowymi i zapisuje startowe położenia, prędkości i masy wszystkich ciał do przechowujących te dane tablic.

#### *splitData*

Określa podział załadowanych danych pomiędzy węzły, umożliwia zrównoleglenie obliczeń.

#### *broadcastInitialData*

Rozsyła wczytane z pliku dane początkowe między wszystkie węzły.

#### *broadcastData*

Rozsyła zmodyfikowane w tym kroku czasowym dane położenia ciał pomiędzy wszystkie węzły.

#### *saveData*

Dodaje aktualne dane o położeniu każdego ciała do pliku wynikowego.

### **Dane wejściowe**

Dane wejściowe programu to zestaw położzeń, prędkości oraz mas wybranych 36 ciał niebieskich z dnia 01.01.2020 o godzinie 00:00. Położenie ciał zostało zaczerpnięte w postaci sferycznej z serwisu HORIZONS organizacji NASA z heliocentrycznego punktu widzenia, a następnie transformowane do postaci kartezjańskiej. Aby uniknąć sytuacji dzielenia przez 0 podczas konwersji Słońce zostało przesunięte o 1 metr, co nie ma znaczącego wpływu na obliczenia symulacji. Prędkości ciał zostały obliczone na

podstawie różnicy w położeniu pomiędzy wymienioną datą a punktem dziesięć minut później.

Dane zapisane są w pliku *nbodydata.txt*, gdzie pierwsza linia stanowi opis danych, a każda kolejna to wpis mówiący o jednym ciele niebieskim.

### **Dane wyjściowe**

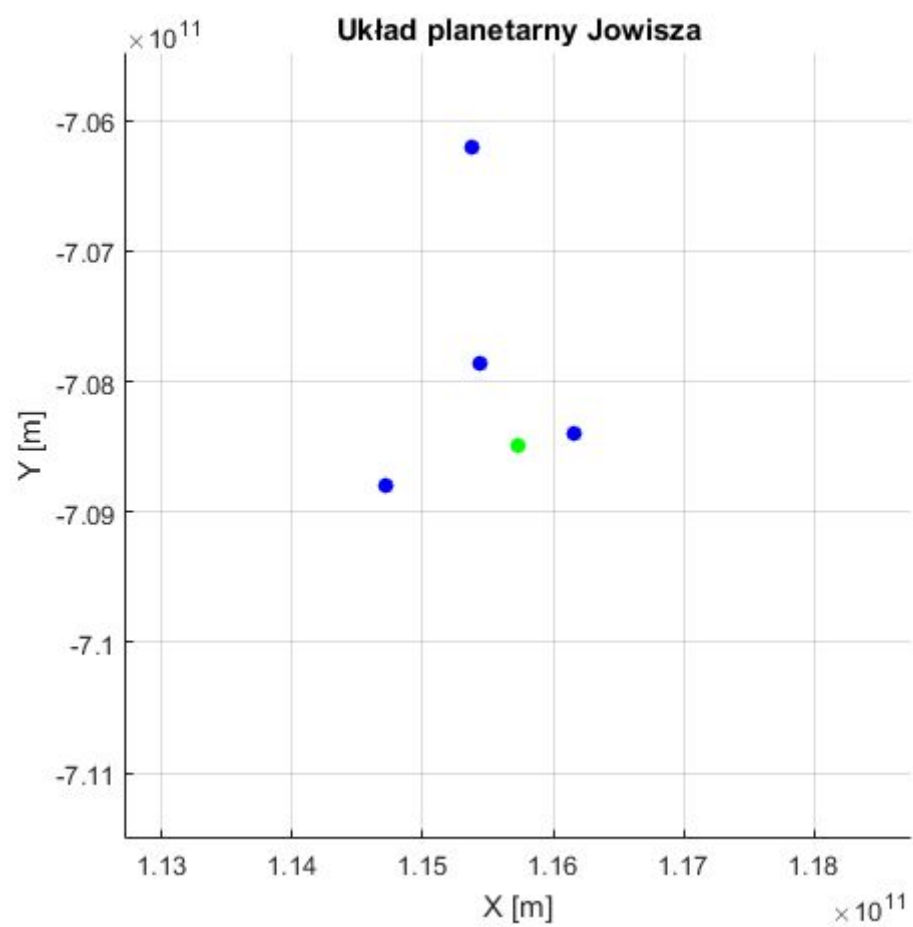
Dane wyjściowe zapisywane są w pliku *resultdata.txt* w postaci podobnej do wejściowych, ale ograniczone są tylko do pozycji ciał. Do ich podglądu zalecamy użycie załączonego skryptu wizualizującego wykonanego w MATLAB, *vizualization.m*.

Obserwacje danych wyjściowych wskazują na istnienie pewnej niedokładności ze stanem rzeczywistym układu słonecznego - orbity księżyców różnych planet wydają się być bardziej ekscentryczne niż być powinno. Może to wynikać z ograniczonej do 3 miejsc po przecinku dokładności pobranych współrzędnych sferycznych.

### **3. Obsługa programu**

Do programu dołączony jest plik *Makefile* udostępniający funkcjonalność kompilacji, uruchomienia i posprzątania po programie, za pomocą trzech poleceń, odpowiednio *make*, *make run*, *make clean*.

Dane zapisane w pliku *resultdata.txt* mogą być wykorzystane z dołączonym programem MATLAB *vizualization.m* do wizualizacji przebiegu orbit ciał poprzez stworzenie pliku wideo. Domyślnie skrypt pokazuje rzut układu planetarnego Jowisza na płaszczyznę XY, a krok czasowy to 100 minut. Rysunek 3 przedstawia zrzut ekranu z przykładowego wideo.



Rys. 3 - Prosta wizualizacja układu obliczonego w problemie wielu ciał.