AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział fizyki i informatyki stosowanej

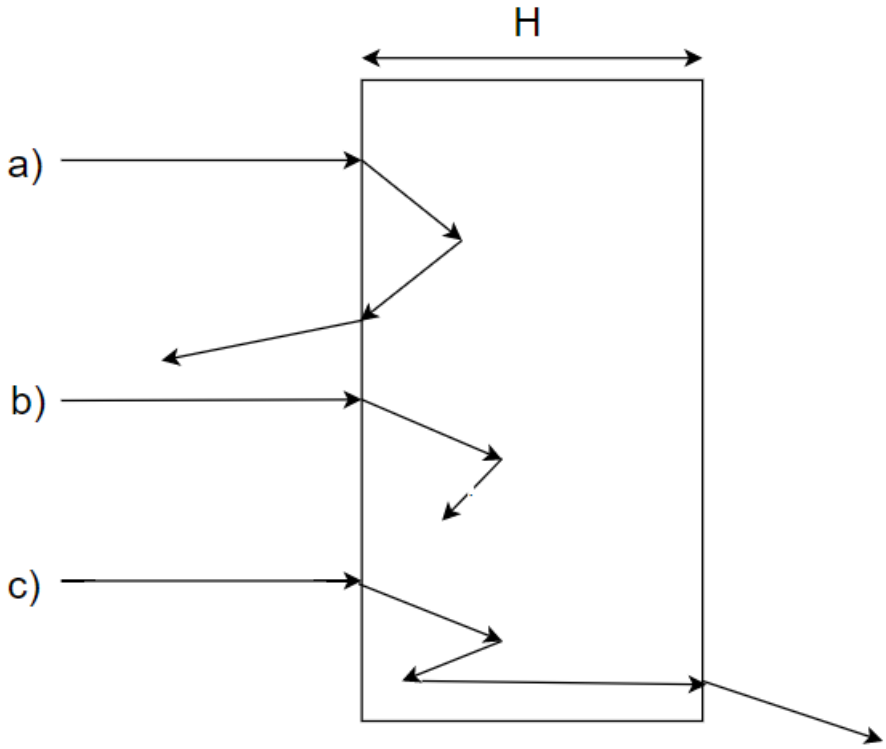
Przedmiot: Programowanie równoległe i rozproszone
Temat: Transport neutronów- metoda Monte Carlo
Autorzy: *Bartłomiej Leśnicki*, Mateusz Niepokój

Kraków, 2023

1. Wstęp

Celem projektu było rozwiązanie problemu transportu neutronów. Należało wyznaczyć prawdopodobieństwo odbicia, transportu i pochłonięcia neutronu dla zadanych parametrów.
Źródło emituje neutrony w kierunku jednorodnej płytki o grubości H i nieskończonej długości. Neutrony mogą:

a) odbić się od płytki
b) zostać zaabsorbowane
c) przejść przez płytkę



Rysunek 1 Możliwe zachowanie neutronu

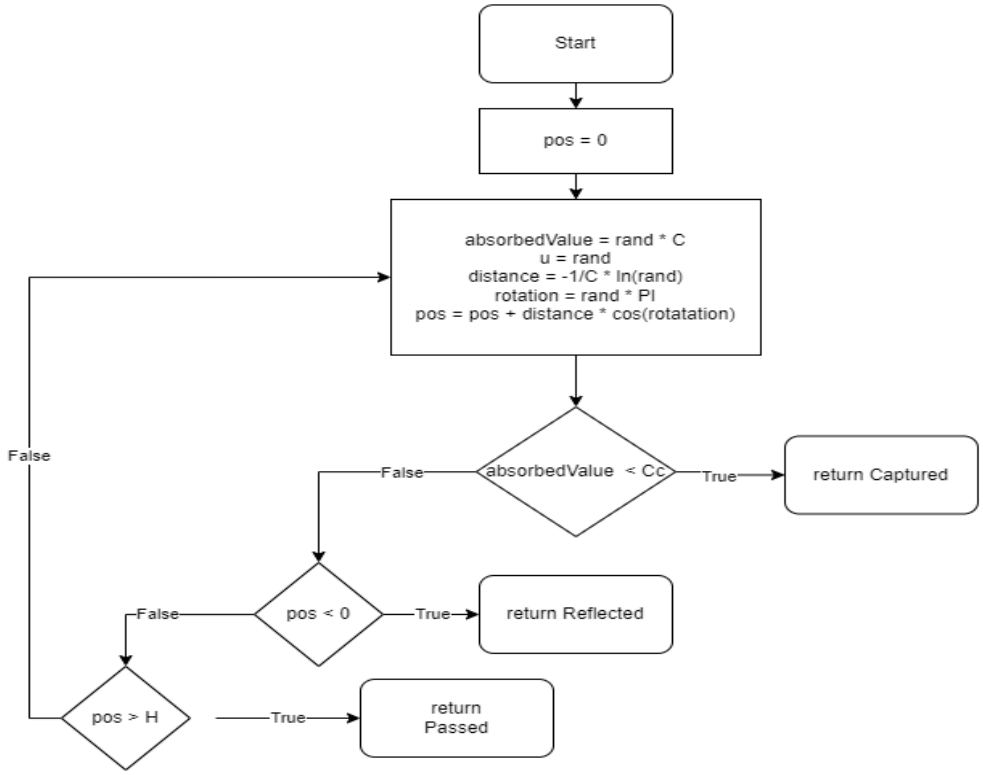
Dwie stałe opisują neutronów w płytce: Cc i Cs. Prawdopodobieństwo, że neutron zostanie zaabsorbowany wynosi Cc, natomiast że zostanie odbity wynoszą Cs. Całkowite prawdopodobieństwo wynosi C=Cs+Cc.
Dystans przebyty przez neutron w pojedynczym kroku jest zgodny z rozkładem:

$$L = \frac{-1}{c} lnu$$

Kiedy neutron oddziałuje z atomami w płytce, prawdopodobieństwo odbicia się od atomu wynosi Cs/C, natomiast prawdopodobieństwo rozproszenia Cc/C.

2. Implementacja

Problem został rozwiązany metodą monte carlo, symulując w każdej iteracji trasę pojedynczego neutronu.



Rysunek 2 Schemta blokowy obliczenia pozycji pojedynczego neutronu

Aplikacja została zaimplementowana jako aplikacja rozproszona w języku Java z wykorzystaniem interfejsu RMI (Remote Metod invocation). Podzielona jest na 3 główne części:

- Klient (klasa NeutronTransportClient)
- Serwer (klasa NeutronTransportServer)
- Interfejs Zdalny (interfejs NeutronTransportInterface)

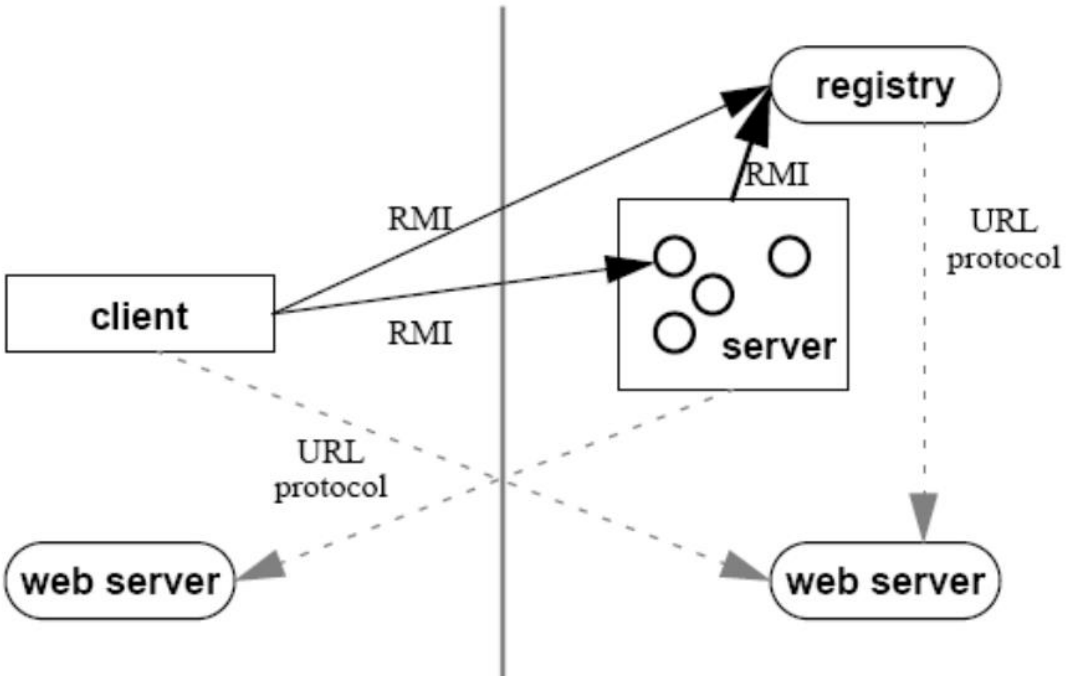
Klasy te odpowiadają za obsługę technologii RMI. Schemat działania interfejsu RMI został przedstawiony na rysunku 3. Tworzone są dwa serwery do których łączy się klient. Deklaracje metod udostępnianych przez serwer dla klienta znajdują znajdują się w interfejsie NeutronTransportInterface. Klienta oraz serwer należy traktować jako dwa osobne programy. Mogą być one wywołane na dowolnych komputerach połączonych siecią (spełniające standardowe wymagania uruchomieniowe javy).
Program można by łatwo rozszerzyć o większą liczbę serwerów dodając dodatkowe rejestry. Dla dużej ilości serwerów można wprowadzić pętlę zmieniając tylko kod Serwera i Klienta o dodatkowy parametr odpowiednio zmieniającym nazwy serwerów i obiektów do których zwracane są outputy metod Interfejsu Zdalnego.

oraz klasy pomocnicze:

- NeutronTransportImpl - klasa odpowiedzialna za działanie metod zdalnych. Serwer wywołuje metody tej klasy. Klasa rozszerza klasę UnicastRemoteObject oraz implementuje interfejs NeutronTransportInterface. W ciele klasy znajduje się klasa prywatna MonteCarloWorker, która zawiera metody do obliczania cząstek algorytmem monte Carlo. Klasa NeutronTransportImpl zawiera metodę performMonteCarloSimulation(), która tworzy

16 wątków obliczeniowych. Na każdym wątku wywoływana jest metoda run klasy MonteCarloWorker.

- NeutronTransportParam - klasa pomocnicza zawiera pola odpowiadające parametrą symulacji.
- MonteCarloWorker zawiera metody do obliczania cząstek algorytmem monte Carlo. Implementuje interfejs Runnable. Zawiera pomocniczą klasę typu Enum do realizacji metody singleNeutronTransport. Klasa MonteCarloWorker zawiera metody:
  - singleNeutronTransport - oblicza trajektorię pojedynczej cząstki
  - calculateDistance - metoda pomocnicza oblicza dystans
  - run - wywołuje wykorzysuje funkcję singleNeutronTransport do obliczenia trajektorii cząstek dla określonej liczby iteracji



Rysunek 3 Schemat działania RMI -źródło: https://view.fis.agh.edu.pl/staff/gronek/parallel/wyklady/Wyklad-13.pdf

3. Obsługa programu

Aby uruchomić program należy najpierw uruchomić serwery, a następnie klienta.
Programy można skompilować komendami:

- javac NeutronTransportServer.java
- javac NeutronTransportClient.java

Skompilowane programy można uruchomić komendą java “nazwa programu”.

- java NeutronTransportServer
- java NeutronTransportClient

Klienta można uruchomić albo z parametrami domyślnymi albo z własnymi podając 4 argumenty programu:

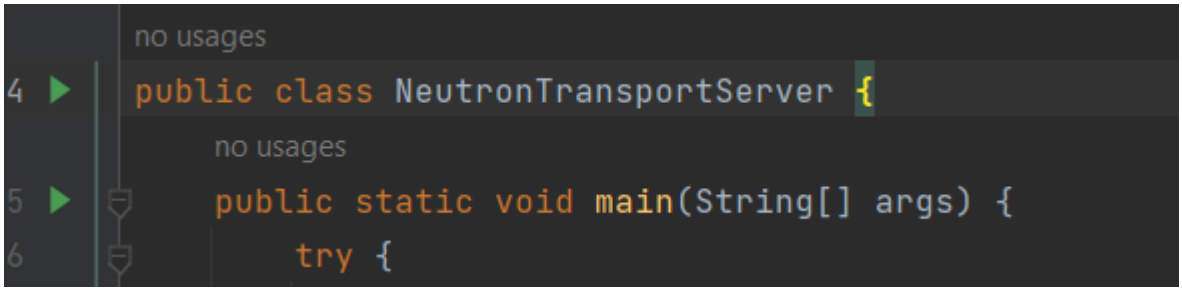
- java NeutronTransportClient iter Cc Cs H

gdzie:

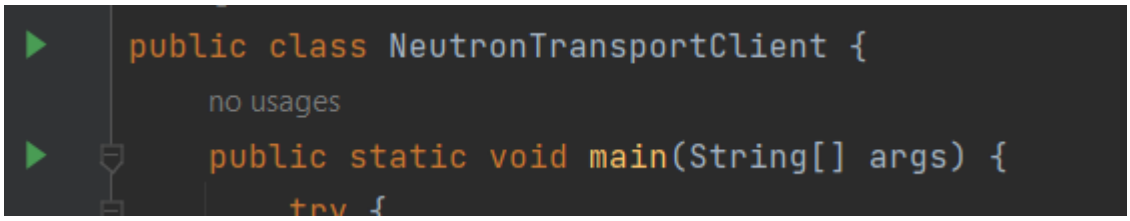
- iter - liczba iteracji dla jednego z 16 wątków danego serwera
- Cc - prawdopodobieństwo rozproszenia neutronu

- Cs - prawdopodobieństwo odbicia neutronu
- H - szerokość płytki

Z poziomu intelij programy można uruchomić za pomocą “zielonego trójkąta” przy nazwie klasy. Uruchamiając najpierw Serwer i potem Klienta rysunek 4 i 5.

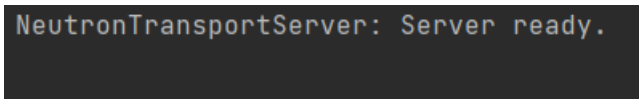


Rysunek 4 Uruchomienie serwera z poziomu Intellij



Rysunek 5 Uruchomienie Klienta z poziomu Intellij

Po poprawnym uruchomieniu Serwera na konsoli znajduje się napis:
„NeutronTransportServer: Server ready.„ przedstawiony na rysunku 6.



Rysunek 6 Napis wyświetlany przez poprawnie uruchomiony Serwer

Po prawidłowym uruchomieniu Klienta na konsoli zostają wypisane:

- Dla każdego serwera: Ilości neutronów odbitych, pochłoniętych, i tych które przeszły przez płytkę, suma wszystkich próbek, szansa odbicia, pochłonięcia i przejścia neutronu.
- Całkowita szansa odbicia, pochłonięcia i przejścia neutronu.

Przykładowy output Klienta przedstawiony został na rysunku 7.

4. Przykładowe wyniki

Poniżej zostały przedstawione przykładowe wyniki działania programów z parametrami domyślnymi rysunek 7 oraz z wybranymi przez użytkownika parametrami rysunek 8.

```
SERVER 1: Reflected particles: 4685929
SERVER 1: Captured particles: 11313658
SERVER 1: Passed particles: 413
SERVER 1: sum: 16000000
SERVER 1: Reflected particle chance: 0.2928705625
SERVER 1: Captured particle chance: 0.707103625
SERVER 1: Passed particle chance: 2.58125E-5
SERVER 2: Reflected particles: 4686743
SERVER 2: Captured particles: 11312831
SERVER 2: Passed particles: 426
SERVER 2: sum: 16000000
SERVER 2: Reflected particle chance: 0.2929214375
SERVER 2: Captured particle chance: 0.7070519375
SERVER 2: Passed particle chance: 2.6625E-5
------------TOTAL------------
Total: Reflected particle chance: 0.292896
Total: Captured particle chance: 0.70707778125
Total: Passed particle chance: 2.621875E-5
```

Rysunek 7 Przykładowy wynik działania Klienta z parametrami domyślnymi

```
9lesnicki@taurus:~/Desktop/SRiR_projekt_v2$ java NeutronTransportClient 3333333 0.3 0.7 1.0
SERVER 1: Reflected particles: 23292453
SERVER 1: Captured particles: 23026955
SERVER 1: Passed particles: 7013920
SERVER 1: sum: 53333328
SERVER 1: Reflected particle chance: 0.43673353742335375
SERVER 1: Captured particle chance: 0.43175544942554495
SERVER 1: Passed particle chance: 0.1315110131511013
SERVER 2: Reflected particles: 23283983
SERVER 2: Captured particles: 23035582
SERVER 2: Passed particles: 7013763
SERVER 2: sum: 53333328
SERVER 2: Reflected particle chance: 0.4365747249074725
SERVER 2: Captured particle chance: 0.4319172056917206
SERVER 2: Passed particle chance: 0.13150806940080695
------------TOTAL------------
Total: Reflected particle chance: 0.43665413116541313
Total: Captured particle chance: 0.43183632755863277
Total: Passed particle chance: 0.13150954127595413
```

Rysunek 8 Przykładowy wynik działania Klienta z wybranymi parametrami