Zaawansowane Techniki Programowania
Praca domowa 2
Dominik Freicher
27.10.2016

OPIS ZADANIA

Zadanie polega na implementacji servletu, który jako dane wejściowe otrzymuje parametry o nazwach *r* oraz *h*, które przekazywane są w url. Parametry określają odpowiednio promień podstawy oraz wysokość walca. Środek podstawy walca umieszczony jest w punkcie (0,0) płaszczyzny (x,y) a oś symetrii walca pokrywa się z dodatnią półosią współrzędnej z. Wymiary walca podane są w metrach. Walec zbudowany jest z materiału o gęstości właściwej *c* wyrażonej w [kg]/[m3]. W materiale występują defekty, które mają charakter przestrzeni o kształcie kulistym i gęstości materiału *g*. Wyniki badania struktury materiału zawierające opis defektów – z których każdy zapisywany jest w postaci równania kuli o środku w punkcie *(x, y, z)* oraz promieniu *r* przekazywane są do servletu w postaci kolejnych żądań typu GET. Żądanie typu POST z parametrami : *r* oraz *h* określającymi rozmiary walca, *c* charakteryzującym gęstość materiału, z którego kostka została wykonana oraz *g* charakteryzującym gęstość materiału w obszarach defektów. Żądanie POST protokołu http winno zwrócić wyznaczoną przez komponent masę rzeczywistą analizowanego bloku materiału. Dla rozwiązania zadania wykorzystać należy metodę Monte Carlo.

ZASTOSOWANY ALGORYTM

Do rozwiązania problemu wykorzystana została metoda Monte Carlo. Jest to metoda symulacyjnego rozwiązywania zagadnień poprzez losowanie wartości zmiennych według określonych rozkładów prawdopodobieństwa.

Założenia:

- Pojedynczy defekt zapisywany jest w postaci równania kuli o środku w punkcie (x, y, z) oraz promieniu r.
- Analizowany blok materiału ma kształt walca o parametrach kolejno r oraz h określających rozmiary walca, c charakteryzującym gęstość materiału.
- Liczba iteracji: 10 000 000 (dokładność wyniku nie mniejsza niż 0.01)

Lista kroków:

1. W celu wyznaczenia objętości pojedynczego defektu należy wygenerować n losowych punktów o współrzędnych x, y, z.
2. Dla każdego wylosowanego punktu należy sprawdzić czy znajduje się w defekcie. Sprawdzenie takie wykonujemy z użyciem wzoru na odległość pomiędzy dwoma punktami w przestrzeni trójwymiarowej. Jeżeli odległość ta jest mniejsza bądź równa w stosunku do promienia defektu można stwierdzić, że wylosowany punkty należy do defektu.

$$|AB| = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2} \leq r$$

gdzie: A – środek defektu, B – wylosowany punkt, r – promień defektu

3. Należy wyznaczyć objętość defektów korzystając z ilorazu wyznaczonych punktów i wygenerowanych n losowych punktów oraz wzoru na objętość walca.

$$V_d = \frac{k}{n} * \pi r^2 H$$

gdzie: k – ilość wyznaczonych punktów, n – ilość wygenerowanych punktów, r – promień podstawy walca, H – wysokość walca

4. Kolejnym krokiem jest wyznaczenie całkowitej masy wszystkich defektów, korzystając z iloczynu gęstości i objętości defektów.

$$m_d = \rho * V_d$$

gdzie: p – gęstość defektu

5. Następnie należy wyznaczyć objętość walca nie uwzględniając defektów korzystając z poniższego wzoru.

$$V_w = \frac{n-k}{n} * \pi r^2 H$$

gdzie: k – ilość wyznaczonych punktów, n – ilość wygenerowanych punktów, r – promień podstawy walca, H – wysokość walca

6. Kolejno wyznaczamy masę walca, korzystając z iloczynu gęstości i objętości walca.

$$m_w = \rho * V_w$$

gdzie: p – gęstość materiału walca

7. Ostatni krok to obliczenie masy całkowitej szukanej bryły, masę taką możemy zdefiniować jako łączną sumę masy walca oraz masy defektów.

$$m = m_w + m_d$$

gdzie: m_w – masa walca, m_d – suma mas wszystkich defektów

Zaawansowane Techniki Programowania
Praca domowa 2
Dominik Freicher
27.10.2016

STRUKTURY DANYCH

Projekt składa się z klasy `Block`. Poniżej wymienione zostały najważniejsze struktury danych:

```
Class Block
  List<Damage> damages
  double result
  void doGet(HttpServletRequest req, HttpServletResponse res)
  void doPost(HttpServletRequest req, HttpServletResponse res)
  void performAnalysis(double r, double h, double c, double g)
  double calculateMonteCarloMethod(double points)
  double getRandomNumber(Random r)
  void printResult(PrintWriter writer)
  void createDamage(String x, String y, String z, String r)
  double convertParameter(String parameter)
  class Damage
    double x
    double y
    double z
    double r
```

Klasa `Block` dziedziczy po klasie `HttpServlet`, w klasie `Block` nadpisane zostały kolejno metody `doGet` oraz `doPost`. Zadaniem pierwszej z nich jest obsługa żądań GET protokołu HTTP. Podczas wywołania żądania GET z przekazanymi parametrami *x, y, z, r* wywołana zostaje metoda pomocnicza `createDamage` której zadaniem jest stworzenie obiektu typu `Damage` oraz uzupełnienie jego pól zgodnie z przekazanymi parametrami. Klasa `Damage` jest reprezentacją defektu, który opisany został jako kula o atrybutach kolejno x, y, z, r. Każdorazowo po utworzenia obiektu typu `Damage` zostaje on zapisany do listy `damages`, w której przechowywane są wszystkie defekty. Za obsługę żądań typu POST odpowiedzialna jest metoda `doPost`. Otrzymanie żądania typu POST z parametrami *r, h, c, g* oznacza, że przekazano komplet danych i należy rozpocząć analizę. W procesie analizy wywołana zostaje metoda `performAnalysis`, do której przekazane zostają wspomniane parametry. W metodzie tej obliczona zostaje łączna masa defektów oraz masa walca. Kluczowym elementem realizacja algorytmu jest metoda `calculateMonteCarloMethod` przekazany do niej argument zawiera liczbę iteracji jaką należy wykonać, metoda ta realizuje główną logikę algorytmu. Ostatnim elementem jest metoda `printResult` odpowiada za wypisanie poprawnie sformatowanego rezultatu przeprowadzonych obliczeń.

URUCHOMIENIE

Uruchomienie programu możliwe jest w środowisku serwera aplikacyjnego z wykorzystaniem plików:

```
120525/WEB-INF/classes/Block*.class
120525/WEB-INF/web.xml
```