# Model bomby atomowe
# Specyfikacja projektu na Programowanie obiektowe

Jacek Strzałkowski Paweł Polak

Marzec 2020

## 1 Opis projektu

Symulacja pokazująca wybuch bomby atomowej. Działaniem programu można sterować modyfikując odpowiednio warunki początkowe: objętość próbki, która determinuje ilość atomów uranu(w zakresie od 0, $10^6$), prawdopodobieństwo wychwytu neutronu prowadzące do rozszczepienia jądra, ilość neutronów wyrzucanych podczas niego $N$. Każdy neutron porusza się w losowym kierunku [z odpowiednią porcją energii], a po przebyciu odcinka ma: i) szansę A na wywołanie kolejnego rozpadu oraz ii) szansę B na losową zmianę kierunku.

Dzięki programowi użytkownik może zobaczyć symulację ruchu neutronów podczas rozpadów jąder atomowych. Dodatkowo, dla każdej symulacji liczona jest moc wydzielona podczas serii rozpadów, którą użytkownik może wyeksportować do pliku tekstowego. Symulacja odbywa się w trzech wymiarach, a wizualizacja ograniczona jest do dwóch.

## 2 Model matematyczny

Rozpady promieniotwórcze jąder atomowych prowadzące do emisji neutronu/ów są zjawiskiem choć naturalnym, to niezwykle rzadkim. Załóżmy, że mamy próbkę zawierającą masę M atomów określonego izotopu promieniotwórczego o prawdopodobieństwie p samoistnego rozpadu.

Program będzie w każdym przedziale czasowym generował parametry symulacji, długość przedziału $dt$ będzie ustalana przez użytkownika.

1. Jądra promieniotwórcze $^{235}_{92}\mathrm{U}$ będą umieszczone na tetraedrycznej siatce o wymiarach 100x100x100. W trakcie trwania programu atom nie zmieni swojej pozycji na siatce. Udział punktów, które będą zapełnione przez aktywne jądra, będzie i) zależał od łącznej masy wszystkich atomów ii) zawierał się w przedziale[1] $(0, 1000000)$ czyli wszystkie pozycje w siatce zostaną zapełnione, gdy łączna masa M
$$M = 1000000 * 235u \approx 3.902269469 * 10^{-19} kg$$
2. Określenie jak wiele jąder promieniotwórczych samoistnie rozpadnie się: $p*N_{pr}$, gdzie $N_{pr}$ - obecna ilość jąder promieniotwórczych przed rozpadem. Rozpad spowoduje wyemitowanie N neutronów.
3. Samorzutny rozpad z utratą N neutronów będzie się odbywał według reakcji:
$$^{235}_{92}\mathrm{U} \longrightarrow {}^{235-\mathrm{N}}_{92}\mathrm{U} + \mathrm{N}\,{}^{1}_{0}\mathrm{n}$$
4. Reakcja rozpadu wymuszonego przez neutron:
$$^{235}_{92}\mathrm{U} + {}^{1}_{0}\mathrm{n} \longrightarrow {}^{93}_{36}\mathrm{Kr} + {}^{140}_{56}\mathrm{Ba} + 3\,{}^{1}_{0}\mathrm{n}.$$

Dalsze rozpady zgodnie z szeregiem promieniotwórczym nie będą uwzględniane w symulacji.

[1] Jest to wartość o wiele mniejsza od wartości masy krytycznej. Rozwiązanie tego problemu w punkcie 7.

5. Stała ilość energii E wydzielona podczas wybuchu zostaje przekazana każdemu z N neutronów: $\frac{E}{N} = E_n$. Prędkość całkowita z jaką porusza się neutron po wybuchu: $|v_n| = \sqrt{\frac{2E_n}{m}}$, gdzie masa neutronu $m_n = 1,67492721 * 10^{-27} kg$.

6. Kierunek ruchu neutronów powstałych po rozpadzie będzie określony losowo.

7. Masa krytyczna dla danego izotopu [w naszym przypadku $^{235}_{92}$U], to taka masa nagromadzonych w określonej objętości atomów, że neutron z jednego z rozpadów promieniotwórczych propaguje dokładnie jeden następny rozpad.

   Ze względu na złożoność obliczeniową niemożliwe jest symulowanie takiej ilości atomów, która odpowiadałaby masie krytycznej. Z tego względu posłużymy się proporcją: całkowite zapełnienie siatki będzie oznaczało, że w układzie jest $\approx$ 1,2 masy krytycznej, a pusta siatka będzie odpowiadała zerowej masie.

8. Działanie programu dla wybranej masy mniejszej niż krytyczna byłoby możliwe: użytkownik będzie mógł zobaczyć rozpady, które jednak nie spowodują wybuchu.

9. Wybuch nie będzie wizualnie specjalnie spektakularny. Kluczową będzie wartość wydzielonej mocy na kilogram ładunku. Jeśli r - ilość rozpadów w jednostce czasu, t - czas od początku symulacji, to
$$P * \text{masaCalkowita} = \frac{E}{t} = \frac{rtm_nc^2}{t} = rm_nc^2 \tag{1}$$

**Wariant 1 - ruch neutronów ograniczony do siatki** Neutrony mogą poruszać się jedynie po punktach na siatce. W każdym momencie programu, neutron będzie znajdował się na określonym polu. Jeśli pole będzie zajęte przez atom, którego jądro nie uległo jeszcze wybuchowi, to będzie miał prawdopodobieństwo A, na spowodowanie takiego. W przeciwnym wypadku, będzie miał B na losową zmianę kierunku.

**Wariant 2 - swobodny ruch neutronów** Neutrony, które traktujemy jak punkty materialne, poruszają się po całej objętości sześcianu 100x100x100. Kierunek ruchu neutronów zostaje znaleziony poprzez takie wylosowanie procentowych wartości $v_x, v_y, v_z$, że $v_x^2 + v_y^2 + v_z^2 = \frac{2E_n}{m}$. Jeśli neutron znajdzie się w odległości nie większej niż
$$R = \sqrt{\frac{\sigma}{\pi}},$$
gdzie $\sigma$ - przekrój czynny atomu $^{235}_{92}$U na rozszczepienie i $\sigma = 582b$, to może nastąpić jego wychwyt i rozszczepienie jądra.

# 3 Algorytm działania programu

Algorytmy obrazujące działanie programu. Algorytm 2 w zasadzie będzie działał równolegle do 1, wykorzystuje on bowiem zmienną *newNeutron*, która będzie jednym z parametrów działania algorytmu 1.

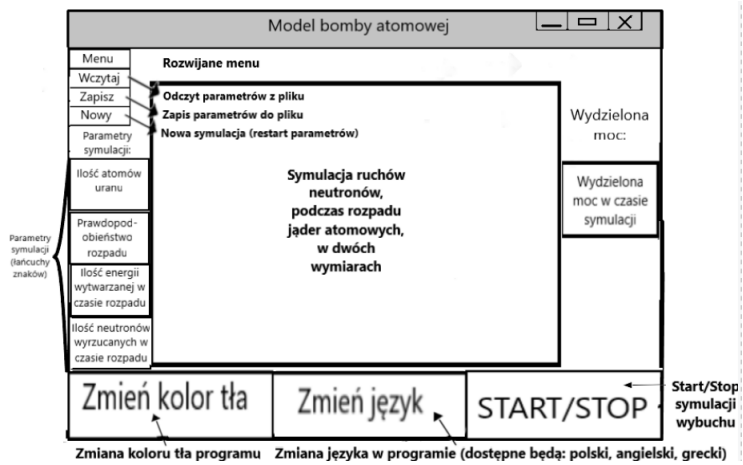
**Algorithm 1** Algorytm na ruch atomów "w symulacji"

```
1: dt ← 0.1, t ← 0
2: allAtom ← UserInput
3: Atomy[allAtom] ← 0, 0, 0
4: isWorking ← UserInput
5: Neutrony* ← 0
6: Określenie, które punkty siatki będą zajęte przez aktywne jądra.
7: zapełnij(Atomy[])
8: while isWorking = true do
9:     Pętla po wszystkich atomach, żeby określić, które ulegą samoistnemu rozpadowi oraz generacja
   neutronów.
10:    for i: Atomy doi + +
11:        if Atomy[i].czyRozpadl = false oraz czyRozpadnie()=true then Atomy[i].czyRozpadl = true
12:            for i: 1-N do
13:                new neutron=generuj(Atom[i])
14:                Neutrony.add(neutron)
15:            Energia += energiaRozpadu()
16:            newNeutron+ = N
17:    liczMoc(newNeutron)
18:    Czyścimy: newNeutron = 0
19:    ruszajAndWybuchaj(neutrony,dt)
20:    t+ = dt
```

**Algorithm 2** Algorytm liczenia mocy wybuchu

1: **procedure** WYZNACZENIE ILOŚCI MOCY W JEDNOSTCE CZASU
2: atomowAll, masaAll $\leftarrow$ UserInput
3: masaNeutronu $\leftarrow 1,67492721 * 10^{-27} kg$
4: $c \leftarrow 3 * 10^8 \frac{m}{s^2}$
5: isWorking $\leftarrow$ UserInput
6: curPower $\leftarrow 0$
7: **while** isWorking $= Yes$ **do**
8: curPower+ = newNeutronmasaNeutronu $* c^2$
9: Output $\leftarrow \frac{\text{curPower}}{masaAll}$

# 4 Interfejs graficzny aplikacji




Rysunek 1: Szkic GUI głównego okna programu

# 5 Scenariusze użycia

Scenariusz: Symulacja wybuchu bomby atomowej

1. Ustawienie, opisanych powyżej, parametrów wybuchu.
   (a) Poprzez wczytanie z pliku (Menu → Wczytaj)
   (b) Poprzez wpisanie string'ów w odpowiednie okienka. Będzie możliwe wybranie jednostki fizycznej, która będzie reprezentować wartość wespół z wpisaną liczbą.
2. Włączenie symulacji następuje po naciśnięciu przycisku START. Zatrzymanie z kolei po naciśnięciu przycisku STOP, który zastąpi przycisk START, gdy symulacja będzie działała.
3. (opcjonalnie) Wybór kolorów tła animacji oraz poszczególnych typów cząstek.
4. (opcjonalnie) Zapisanie wykonanej symulacji poprzez wybranie Menu → Zapisz...
5. (opcjonalnie) Wyczyszczenie wszystkich danych poprzez wybranie Menu → Nowy...

# 6 Wymagania dodatkowe

1. Umieszczenie finalnego programu na serwerze w systemie kontroli wersji GIT
2. Użycie systemu kontroli wersji GIT podczas pisania projektu
3. Możliwy odczyt i zapis z(do) pliku tekstowego.
4. Program można zainstalować za pomocą java web start
5. Program można zainstalować za pomocą instalatora w systemie Windows
6. Skorzystanie z technik/bibliotek nie omawianych na wykładzie
7. Projekt jest dostępny na licencji open source.
8. Program dostępny jest w angielskiej i polskiej wersji językowej.

# 7 Terminarz realizacji projektu

I. Specyfikacja (3 zajęcia) 20.III.2020 r.

II. Prototype - User Interface - gotowy interfejs użytkownika - 3.IV.2020 r.

III. Release Candidate (10 zajęcia) - implementacja minimum połowy funkcjonalności

IV. Final (15 zajęcia) - ukończony projekt spełniający założone wymagania

# 8 Ocena oraz tabela zadań

| Funkcjonalność | Maksymalna ilość punktów | Uzyskana ilość punktów | Notatki |
|---|---|---|---|
| 1 | GUI | .... pkt | |
| 2 | Program znajduje się repozytorium GIT | obowiązkowo | |
| 3 | Poprawne zastosowanie teorii fizycznej do wykonywanych obliczeń | .... pkt | |
| 4 | Podstawowa funkcjonalność programu | .... pkt | |
| 5 | System kontroli wersji GIT | .... pkt | |
| 6 | Możliwa instalacja za pomocą Java web start | .... pkt | |
| 7 | Zapis i odczyt parametrów z pliku tekstowego | .... pkt | |
| 8 | Działający przycisk START/STOP | .... pkt | |
| 9 | Płynna animacja wybuchu | .... pkt | |
| 10 | Angielska wersja językowa | .... pkt | |
| 11 | Umieszczona informacja o licencji programu | .... pkt | |
| 12 | Użycie biblioteki nie omawianej na wykładzie | ... pkt | |
| | Ilość punktów: | ..... pkt | |

Za poprawnie wykonany projekt chcielibyśmy uzyskać ocenę 5.