Analiza Algorytmów - Laboratorium 4

Wojciech Sęk

8 maja 2023

1 Symulator

1.1 Kod źródłowy

```
pub fn simulation(n: u64, q: f64, max_iter: u64) -> bool {
    let rand = fastrand::Rng::new();
    // liczba bloków adwersarza
    let mut adversary = 0;
    // liczba uczciwych bloków
    let mut legitimate = 0;

    // czekamy na moment aż uczciwi gracze nadbudują n bloków
    while legitimate < n {
        if rand.f64() <= q {
            adversary += 1;
        } else {
            legitimate += 1;
        }
    }

    // dla dużej liczby iteracji prowadzimy grę adwersarza
    for _ in 0..max_iter {
        // jeśli adwersarz dogoni uczciwych to wygrywa
        if adversary >= legitimate {
            return true
        }
        if rand.f64() <= q {
            adversary += 1;
        } else {
            legitimate += 1;
        }
    }

    // adwersarzowi nie udało się dogonić uczciwych graczy w dużym czasie
    false
}
```

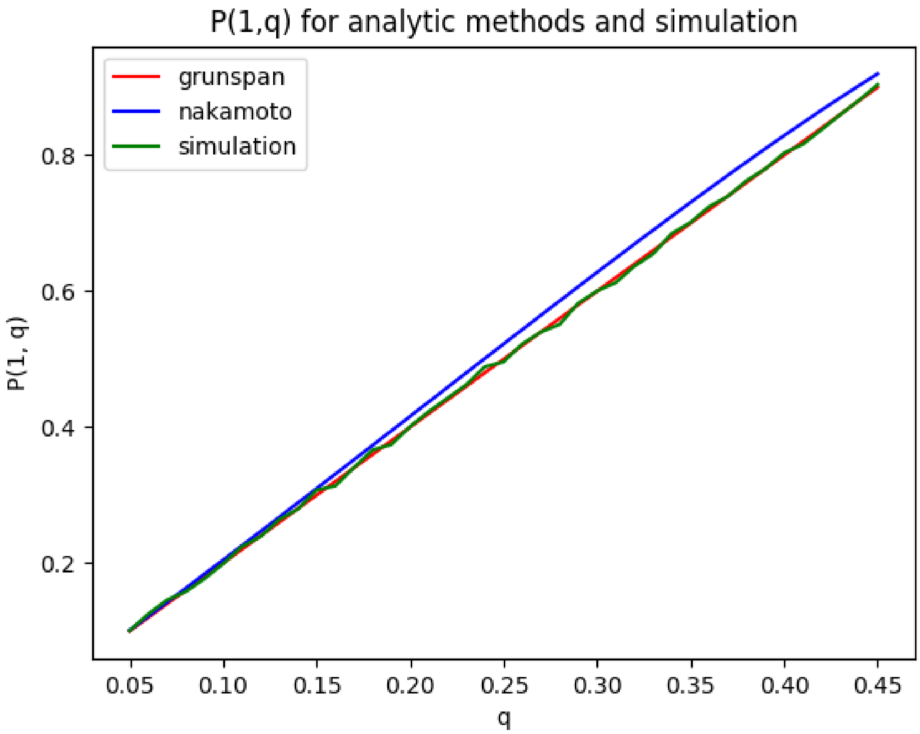
1.2 Idea

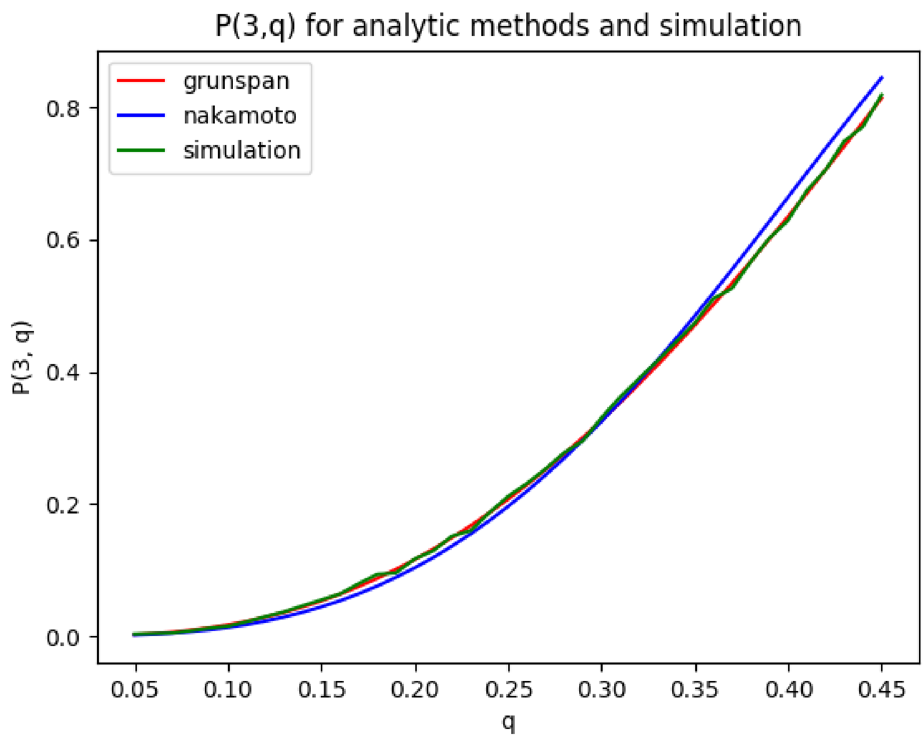
Idea symulatora polega na tym, że startujemy w pewnym momencie życia blockchaina, w którym nikt nie ma przewagi. Następnie prowadzimy symulację (nadbudowanie bloku przez adwersarza z prawdopodobieństwem q i przez uczciwych graczy z $1 - q$) aż uczciwi gracze nadbudują n bloków.
Szukamy czy w tym momencie lub później adwersarz dogoni uczciwych graczy. Symulowanie w nieskończoność nie ma racji bytu, zatem iterujemy dużą liczbę razy wydobywanie kolejnego bloku. Jeżeli w tym długim czasie adwersarz dogonił uczciwych graczy to odnosimy sukces i zwracamy TRUE. W

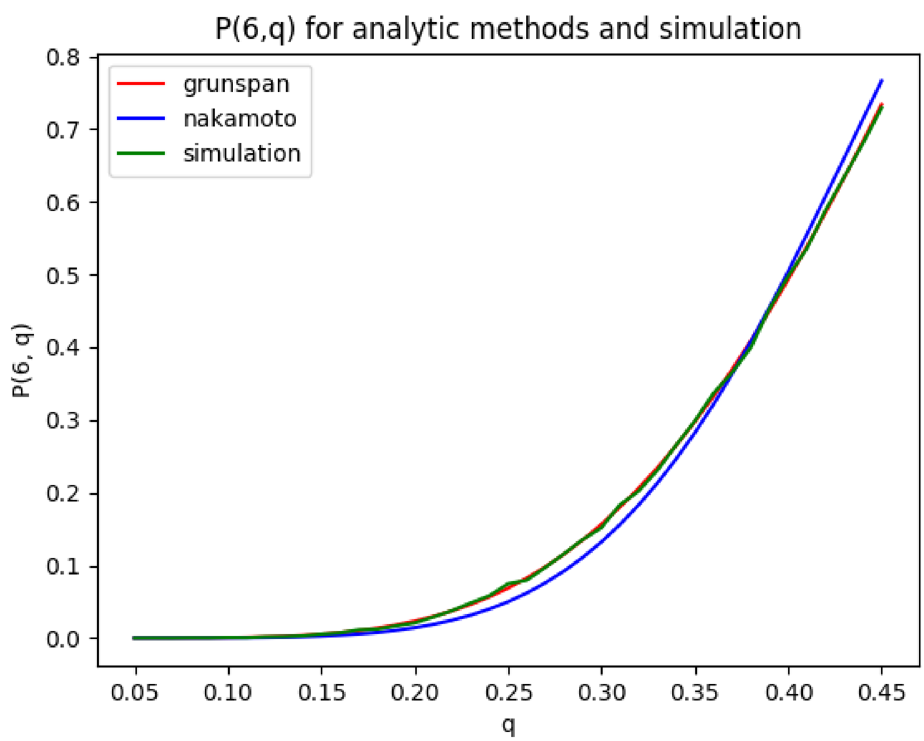
przeciwnym przypadku zakładamy, że adwersarzowi nie uda się dogonić uczciwych graczy i zwracamy FALSE.

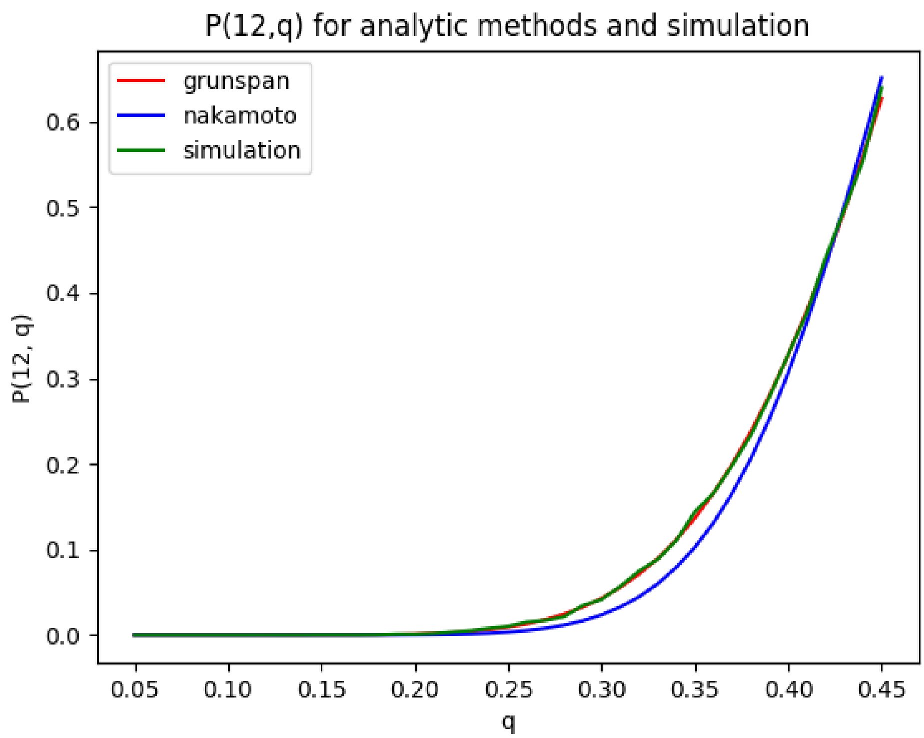
2 $P(n, q)$ dla ustalonego n

2.1 Wykresy

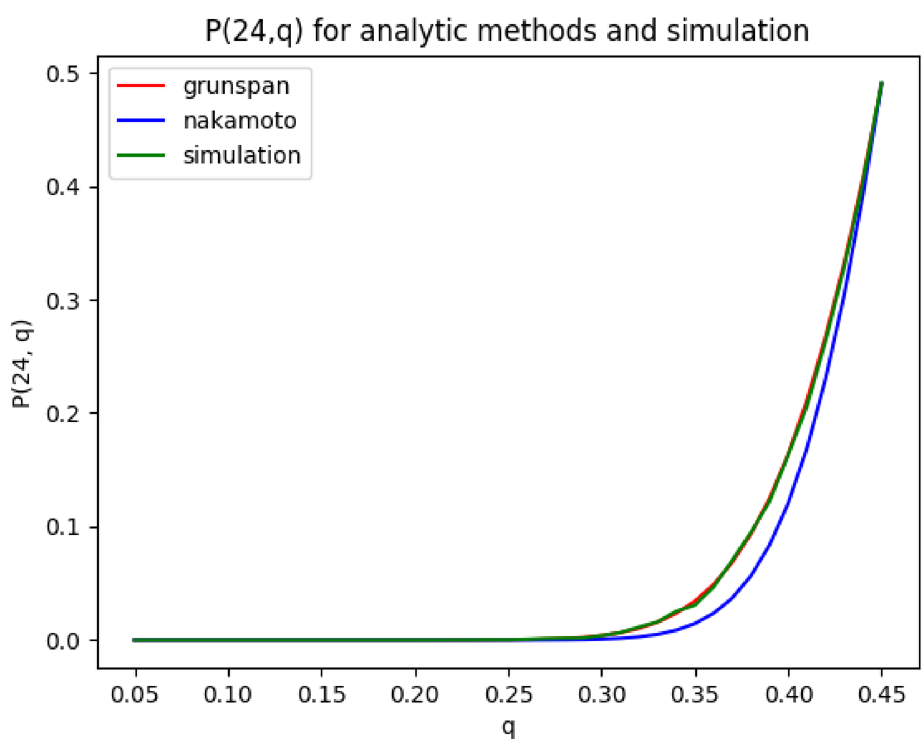



P(3,q) for analytic methods and simulation
grunspan
nakamoto
simulation
P(3, q)
0.0
0.2
0.4
0.6
0.8
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
q

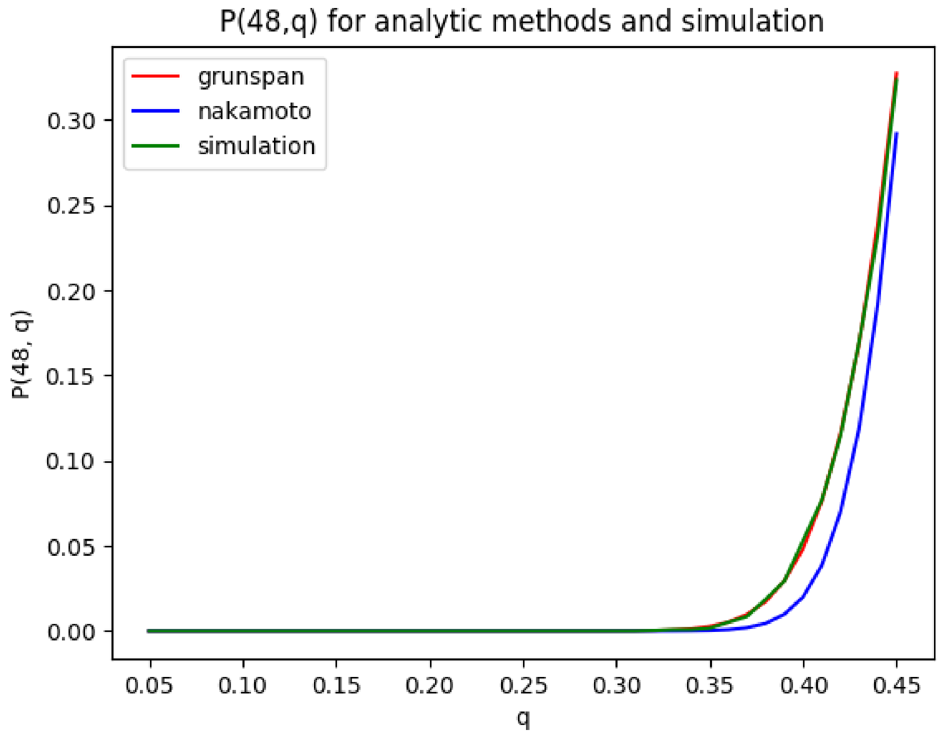

P(6,q) for analytic methods and simulation
grunspan
nakamoto
simulation
P(6, q)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
q



P(12,q) for analytic methods and simulation
grunspan
nakamoto
simulation
P(12, q)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
q



P(24,q) for analytic methods and simulation
grunspan
nakamoto
simulation
P(24, q)
0.0
0.1
0.2
0.3
0.4
0.5
0.05
0.10
0.15
0.20
0.25
0.30
0.35
0.40
0.45
q

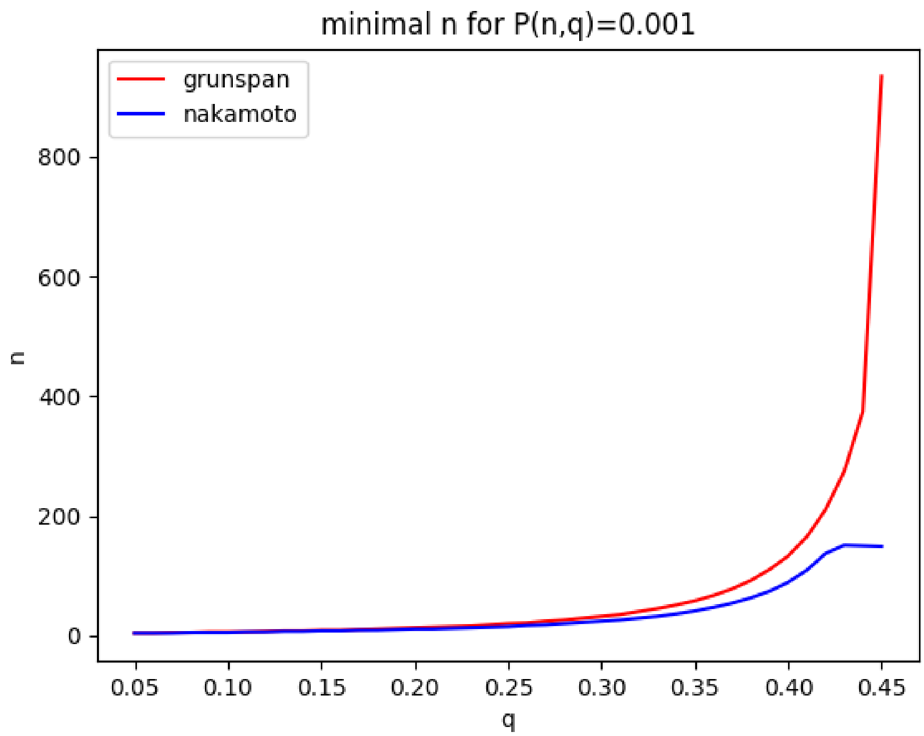


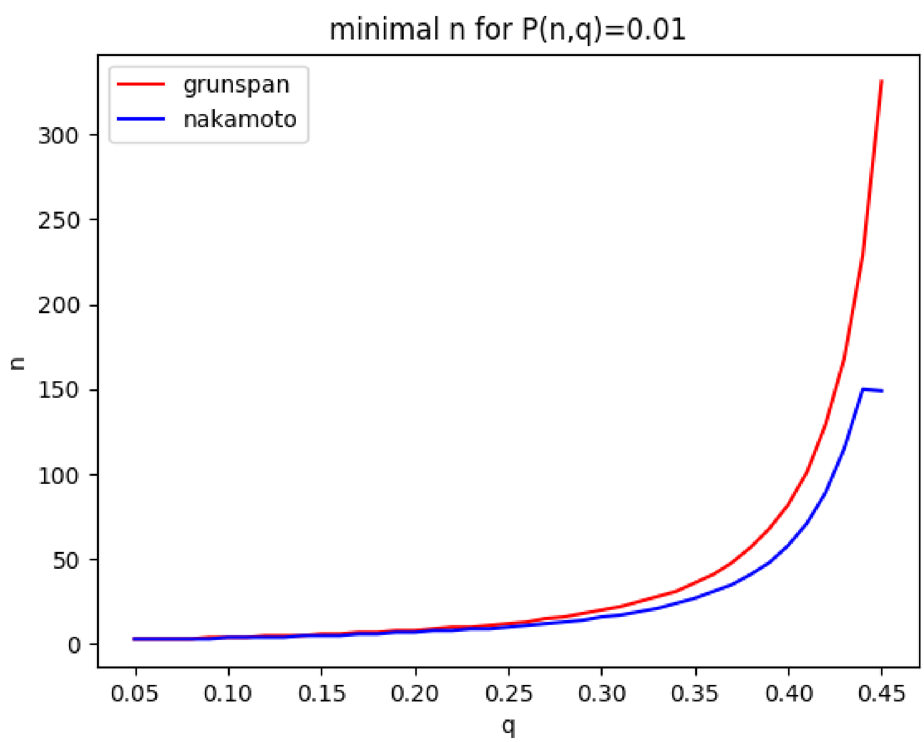
2.2 Obserwacje

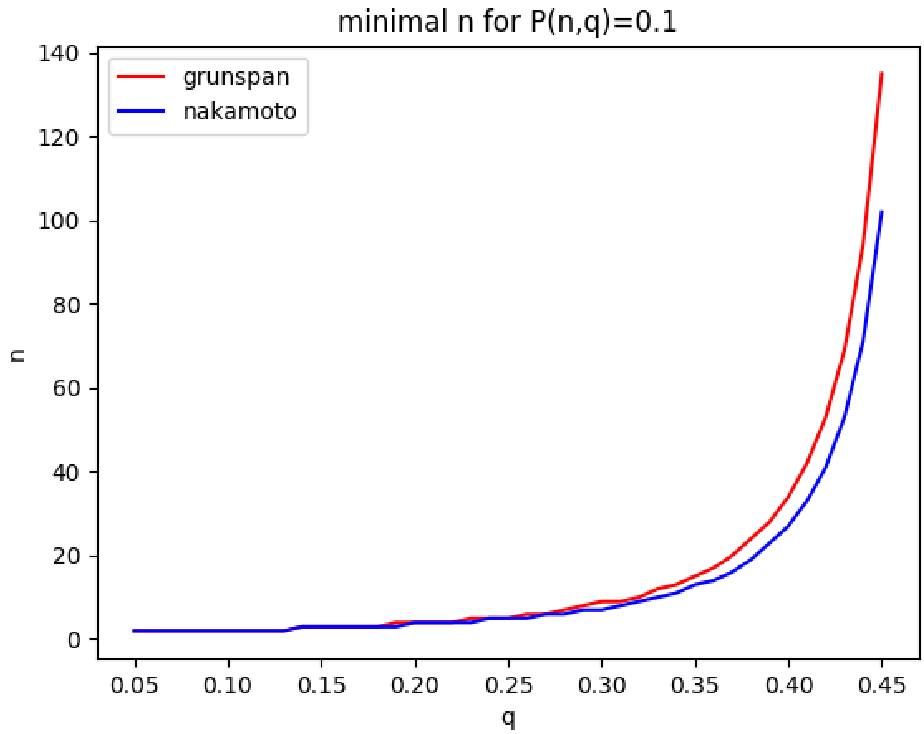
Dla $n = 1$ $P(n, q)$ jest funkcją liniową od q.
Dla większych n funkcja $P(n, q)$ przyjmuje wartości bliskie zero dla większych q, natomiast potem drastycznie rośnie.
Dla ustalonego q funkcja $P(n, q)$ jest malejąca względem n.
Symulacja praktycznie pokrywa się z analizą Grunspana, analiza Nakamoto przyjmuje, że faktyczny czas wydobywania bloków równy wartości oczekiwanej czasu wydobycia tych bloków, co jest zbyt dużym uproszczeniem.

3 n dla danego q by $P(n, q) = \pi$, gdzie π to stała

3.1 Wykresy







3.2 Obserwacje

Im większe $P(n, q)$ tym mniejsze n musimy brać dla ustalonego q, ponieważ ułatwiamy adwersarzowi wygraną.
Analiza Nakamoto zaniża wartość $P(n, q)$ wobec tej z analizy Grunspana, i dlatego wedle niej wystarczą mniejsze wartości n by zapewnić bezpieczeństwo blockchainowi.
Dla wartości $q > 0.4$ i małych $P(n, q) = 0.001$ analiza Nakamoto zaburza wartość n w dół szczególnie mocno wobec Grunspana.