

Algorytmy dla Problemów Trudnych  
Obliczeniowo.  
Rozwiązanie zadania: **Potęga hetmanów**  
Informatyka 2015/16

Autor: Zbigniew Królikowski

13 lipca 2016

Prowadzący: dr hab. inż. Piotr Faliszewski

## 1 Uwagi

Moja filozofia zapisu: **hetmany bijące nie zmieniają miejsca tylko znikają na rzecz zbitego** - taki obraz przyjąłem gdyż łatwiej było go rozpisać, mniej dynamiki. Często zapisuję bicie jako redukcję. Nie zmienia to w żaden sposób istoty algorytmu tylko jego opis, który mógłby być mylący bez tej notki.

## 2 Reprezentacja problemu

Plansza została przedstawiona jako **graf nieskierowany**, w którym każdy węzeł połączony jest z innymi co najwyżej 8 krawędziami.

### 2.1 Zapis wartości hetmanów

W celach uproszczenia obliczeń wartości są reprezentowane jako wykładniki liczby 2.

### 2.2 Krótki słownik moich pojęć - w razie wątpliwości

- osie - pion, poziom i skosy

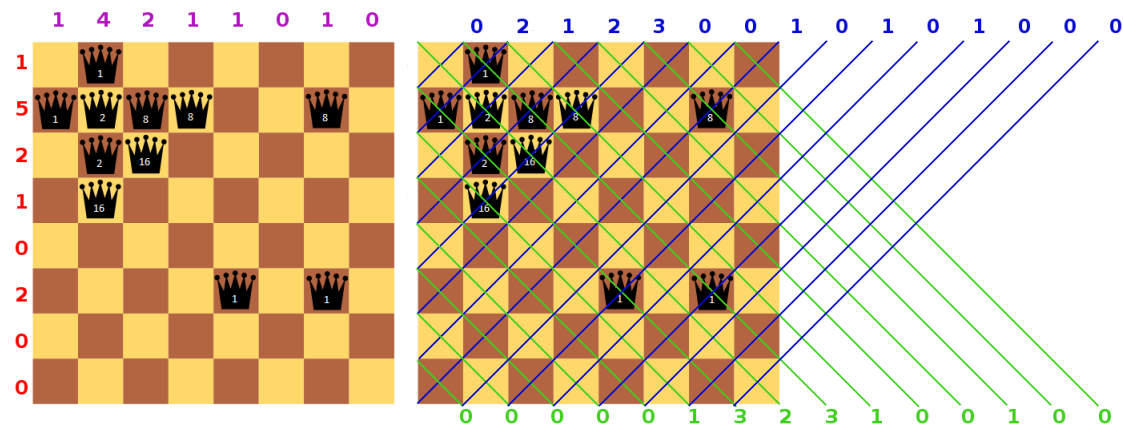
### 3 Wstępne obserwacje

### 4 Ograniczenie maksymalnej wartości hetmana

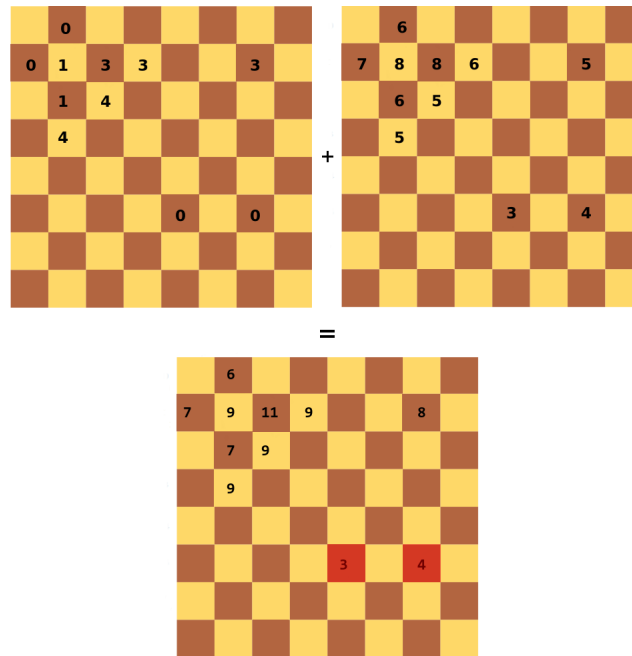
Na wartość końcową hetmana da się nałożyć pewne ograniczenie:

**Końcowa wartość hetmana  $h$  nie może być wyższa od jego aktualnej wartości  $V(h) * 2^n$  gdzie  $n$  - liczba hetmanów w pionie, poziomie i na skosach (osiach).**

Jest to łatwo udowodnić - każde bicie skierowane przeciwko hetmanowi podnosi jego wartość dwukrotnie. Bić może być *w najlepszym wypadku* tyle ile hetmanów będących na tych samych osiach co dany hetman.



Rysunek 1: Zliczenie ilości hetmanów na poszczególnych osiach.



Rysunek 2: *Lewo*: Wartość hetmanów w mojej notacji *Prawo*: maksymalna ilość biał skierowana ku konkretnemu hetmanowi. *Dół*: Górne ograniczenie na wartość końcową hetmanów. Na czerwono hetmany, które nie mogą być końcowe dla  $K=1$ .

Jak zademonstrowano na powyższym przykładzie możemy **wykluczyć** pewne kombinacje (o wielkości  $K$ ) hetmanów z bycia końcowymi. Warunkiem wykluczenia jest niemożliwość uzyskania przez tą kombinację wartości będącej sumą wartości wszystkich hetmanów na planszy. Jest to pewne uproszczenie problemu, część postaci rozwiązania.

**Bezpośrednio nieużyteczne** - liczba tych kombinacji wynosi  $\binom{N}{K}$  gdzie  $N$  to liczba hetmanów na planszy a  $K$  to ilość hetmanów końcowych.

#### 4.0.1 Heurystyka

Możemy się zatem spodziewać, że pewne hetmany zostaną zredukowane "wcześniej niż później", a więc mamy nałożoną jakąś **kolejność przeszukiwania** przestrzeni rozwiązań. **Skoro mniejszy potencjał hetmana wiąże się z niemożnością bycia końcowym hetmanem to może powinniśmy ruszać się wcześniej tymi z mniejszym potencjałem?**

## 5 Algorytm

Pominałem opis wczytywanie wejścia oraz wypisywania rozwiązania.

```
bool recursiveCheck():
    if achievedTarget:
        true
    else:
        sortQueens()
        for queen in board:
            connections = queen.possibleConnections()
            sortConnections()
            for connection in connections:
                if recursiveCheck():
                    return true
        // Nie znaleziono rozwiązania
        undoMove()
        false
```

## 6 Generator scenariuszy

Dodatkowo wykonałem w języku Python generator który dla danej wielkości planszy, końcowej ilość hetmanów i docelowej ilości ruchów tworzy losowy scenariusz wraz z przykładowym rozwiązaniem.

Okazał się on bardzo przydatny w testowaniu algorytmu.