# Projekt podstawy podejmowania decyzji - Hitori solver

Andrzej Dąbrowski

June 25, 2023

## 1 Wprowadzenie

Hitori to logiczna gra łamigłówkowa, która została stworzona w Japonii prze firmę Nikoli. Celem gry jest wykreślanie liczb na planszy w taki sposób, aby żadna liczba nie powtarzała się w wierszu i kolumnie, a także aby żadne dwie wykreślone liczby nie były sąsiadującymi polami - zarówno w poziomie, jak i w pionie.

Pełna lista zasad Hitori:

1. Żadna liczba nie może się powtarzać w wierszu lub kolumnie.
2. Żadne dwie takie same liczby nie mogą być sąsiadującymi polami (dotykać się bokami).
3. Wszystkie puste pola muszą być dostępne z pozostałymi polami planszy poprzez ruchy w pionie lub poziomie. Nie mogą istnieć odizolowane grupy pustych pól.

Podczas rozwiązywania Hitori trzeba analizować układ liczb na planszy i strategicznie wykreślać odpowiednie liczby, aby spełnić powyższe zasady. Rozwiązanie jest zawsze unikalne dla każdej planszy.

## 2 Rozwiązanie

### 2.1 Dane:

$u$ - liczba unikalnych cyfr zapisabnych w tablicy gry
$j$ - wymiary tablicy gry

$D_{j\times j}$ - macierz reprezentująca plansze gry

$B_{j\times j\times u}$ - lista macierzy binarych; w jedenej macierzy zakodowano miejsce występowania danej liczby w tablicy gry.

$X_{3j\times 3j} = \begin{bmatrix} x_{11} & \cdots & x_{13j} \\ \vdots & \ddots & \vdots \\ x_{3j1} & \cdots & x_{3j3j} \end{bmatrix}$ - binarna macierz decyzji, gdzie 1 - pole zacieniowane, 0 - pole nie zacieniowane.

Macierz $X$ ma wymiary 3 razy większe od wymiarów macierzy $B$ z powodu implementacji ograniczenia związanego z zasadą nr.3 w CPLEX. $j$ pierwszych i $j$ ostatnich kolumn oraz wierszy stanową kolumny pomocnicze, które są zapełnione wartościami 1. Kolumny i wiersze od $j$ do $2j$ stanowią aktualną macierz decyzji o zacieniowaniu komórki.

### 2.2 Funkcja celu:

$$F(X) = \sum_{n=1}^{3j}\sum_{i=1}^{3j} x_{in} \tag{1}$$

- całkowite pokrycie zacieniowanymi polami

$$X^* = \arg\min_X \sum_{n=1}^{3j}\sum_{i=1}^{3j} x_{in} \tag{2}$$

- minimalne całkowite pokrycie zacieniowanymi polami

## 2.3 Ogranicznia:

Dla każdego y $\in \{1, 2, ..., j\}$:

$$\sum_{i=1}^{3j} x_{iy} = 3j \tag{3}$$

- ograniczenie związane z zapełnieniem lewych kolumn pomocniczych wartością 1
Dla każdego y $\in \{1, 2, ..., j\}$:

$$\sum_{i=1}^{3j} x_{iy} = 3j \tag{4}$$

- ograniczenie związane z zapełnieniem prawych kolumn pomocniczych wartością 1
Dla każdego w $\in \{1, 2, ..., j\}$:

$$\sum_{i=1}^{3j} x_{wi} = 3j \tag{5}$$

- ograniczenie związane z zapełnieniem górnych rzędów pomocniczych wartością 1
Dla każdego w $\in \{1, 2, ..., j\}$:

$$\sum_{i=1}^{3j} x_{wi} = 3j \tag{6}$$

- ograniczenie związane z zapełnieniem dolnych rzędów pomocniczych wartością 1
Dla każdego f $\in \{1, 2, ..., u\}$: oraz dla każdego w $\in \{1, 2, ..., j\}$:

$$\sum_{i=2j}^{3j} (1 - x_{wi}) * b_{fwi} \leq 1 \tag{7}$$

- ograniczenie związane z powtarzającymi się liczbami w tablicy gry w rzędach
Dla każdego f $\in \{1, 2, ..., u\}$: oraz dla każdego y $\in \{1, 2, ..., j\}$:

$$\sum_{i=2j}^{3j} (1 - x_{iy}) b_{fiy} \leq 1 \tag{8}$$

- ograniczenie związane z powtarzającymi się liczbami w tablicy gry w kolumnach
Dla każdego r $\in \{1, 2, ..., j\}$: oraz dla każdego y $\in \{1, 2, ..., j-1\}$:

$$x_{ry} + x_{r(y+1)} \leq 1 \tag{9}$$

- ograniczenie związane z eliminacją sąsiadujących ze sobą zacieniowanych pól w wierszach
Dla każdego c $\in \{1, 2, ..., j\}$: oraz dla każdego w $\in \{1, 2, ..., j-1\}$:

$$x_{wc} + x_{(w+1)c} \leq 1 \tag{10}$$

- ograniczenie związane z eliminacją sąsiadujących ze sobą zacieniowanych pól w kolumnach
Dla każdego y $\in \{1, 2, ..., j-1\}$ , dla każdego w $\in \{j+1, j+2, ..., 2j\}$ oraz dla każdego n $\in \{j+1, j+2, ..., 2j\}$

$$x_{(y-1+p)(w+1+n-p)} + x_{(y-1+p)(w-1-n+p)} + x_{(y+1-p)(w+1+n-p)} + x_{(y+1-p)(w-1-n+p)} \leq 4n + 4 - 1 \tag{11}$$

- ograniczenie związane z zasadą numer 3.

Powyższe ograniczenie ma pewne restrykcje i nie jest w stanie wykrywać pewnych zakreśleń łamiących zasadę nr.3. Jest to związane z ograniczeniami solvera CPLEX. Przykładowe zacieniowanie nie wykrywane przez wspomniane ograniczenie:

$$\begin{bmatrix} 3 & 3 & \blacksquare & 3 & 3 \\ 3 & 3 & 3 & \blacksquare & 3 \\ 3 & 3 & \blacksquare & 3 & 3 \\ 3 & 3 & 3 & \blacksquare & 3 \\ 3 & 3 & \blacksquare & 3 & 3 \end{bmatrix}$$

# 3 Wyniki

Dla u = 5 i j = 5 oraz planszy

$$D_{5\times5} = \begin{bmatrix} 2 & 2 & 1 & 5 & 3 \\ 2 & 3 & 1 & 4 & 5 \\ 1 & 1 & 1 & 3 & 5 \\ 1 & 3 & 5 & 4 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{bmatrix}$$

Poprawne rozwiązanie:

$$\begin{bmatrix} \blacksquare & 2 & \blacksquare & 5 & 3 \\ 2 & 3 & 1 & 4 & \blacksquare \\ \blacksquare & 1 & \blacksquare & 3 & 5 \\ 1 & \blacksquare & 5 & \blacksquare & 2 \\ 5 & 4 & 3 & 2 & 1 \end{bmatrix}$$

Solver zwrócił wynik (zmienne pomocnicze pominięto):

$$X^* = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Wizualizacja:

$$\begin{bmatrix} \blacksquare & 2 & \blacksquare & 5 & 3 \\ 2 & 3 & 1 & 4 & \blacksquare \\ \blacksquare & 1 & \blacksquare & 3 & 5 \\ 1 & \blacksquare & 5 & \blacksquare & 2 \\ 5 & 4 & 3 & 2 & 1 \end{bmatrix}$$

Solver poprawnie zacieniował 7 komórek, przez co rozwiazał testową instancję puzzli Hitori poprawnie.

Do drugiego rozwiązania użyto metaheurytsyki Tabu Search o długości tablicy Tabu = 10 oraz maksymalnej ilości iteracji 1000.

Algorytm uruchomiono 20 razy. Dla żadnego z uruchomień algorytm nie wyznaczył wszystkich 7 komórek do zacieniowania.

Statystyki zacieniowanych komórek:
Średnia: 1.2
Mediana: 1.0
Max: 3.0
Min: 0.0
Odchylenie standardowe: 0.89

Przykładowe rozwiązanie wyznaczone przez Tabu Search (zmienne pomocnicze pominięto):

$$X^* = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Wizualizacja:

$$\begin{bmatrix} \blacksquare & 2 & 1 & 5 & 3 \\ 2 & 3 & 1 & 4 & 5 \\ \blacksquare & 1 & \blacksquare & 3 & 5 \\ 1 & 3 & 5 & 4 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{bmatrix}$$

# 4 Wnioski

Próbę rozwiązania puzzli Hitori przeporowadzono na dwa sposoby, poprzez solver oraz metaheurystykę. Przy użyciu solvera udało się osiągnąć poprawne rozwiązanie (rozwiązanie działa poprawnie z wyłączeniem pewnych szczególnych przypadków), rozwiązanie przy użyciu metaheurystyki nie osiągneło zadawalających wyników.

Link do kodów źródłowych rozwiązania: https://github.com/11jolek11/Hitori