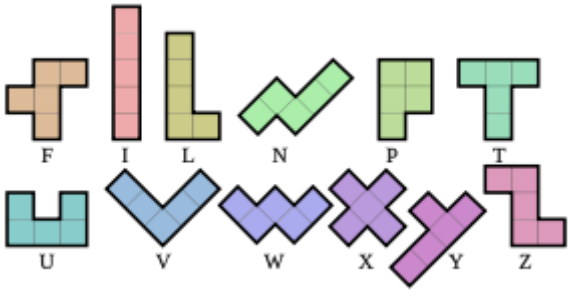
Sprawozdanie projekt PPD

1. Temat

Temat projektu to maximum cover problem z zastosowaniem zapełnienia poprzez figury pentomino. Optymalne rozwiązanie to takie, które maksymalizuje pokrycie, z zachowaniem ograniczenia, że wszystkie figury użyjemy co najwyżej raz. Rozwiązanie pozwala na użycie dowolnej rotacji figur.



Rys. 1. Figury Pentomino

2. Zmienne

r – liczba wierszy macierzy wynikowej

c – liczba kolumn macierzy wynikowej

p – liczba pentomino (12)

f – liczba możliwych pokryć planszy danym pentomino

P – macierz możliwych pokryć każdej komórki *p* × *f* × *r* × *c* wskazująca:

- 1, jeśli pokrycie *f*, przy użyciu pentomino *p* zapełnia komórkę *r*, *c*.
- 0 wpp.

3. Zmienne decyzyjne

X_{pf} - macierz decyzyjna *p* × *f* przyjmująca wartości:

- 1, jeśli wybieramy pokrycie *f* pentomino *p*
- 0 wpp.

Y_{rc} – macierz decyzyjna *r* × *c* reprezentująca zapełnienie max cover, przyjmuje wartości:

- 1, jeśli komórka *r*, *c* jest zapełniona
- 0 wpp.

4. Funkcja Celu

$F(X) = \sum_{i=1}^{r}\sum_{j=1}^{c} Y_{i,j}$ – ilość zapełnionych komórek.

Wyznaczamy:

$X = \arg max_X F(X)$

5. Ograniczenia

$\forall_{r=1}^{r}\forall_{c=1}^{c}\sum_{p=1}^{p}\sum_{f=1}^{f} P_{pfrc} \cdot X_{pf} \leq 1$ – każda komórka jest zapełniona co najwyżej raz,

$\forall_{r=1}^{r}\forall_{c=1}^{c}\sum_{p=1}^{p}\sum_{f=1}^{f} P_{pfrc} \cdot X_{pf} = Y_{rc}$ – każda komórka zapełniona odpowiada komórce macierzy *Y*,

$\forall_{p=1}^{p}\sum_{f=1}^{f} X_{pf} \leq 1$ – każde pentomino jest użyte co najwyżej raz.

6. Cplex

```
1. int row = 6;
2. int column = 10;
3. int f = 50;
4. int p = 12;
5. int P[1..p][1..f][1..row][1..column] = ...;


6. dvar boolean x[1..p][1..f];
7. dvar boolean y[1..row][1..column];


8. maximize sum(r in 1..row, c in 1..column)y[r][c];

9. subject to {
10.forall(r in 1..row, c in 1..column) sum(p in 1..p, f in 1..f)
    P[p][f][r][c] * x[p][f]) <= 1;

11.forall(r in 1..row, c in 1..column) sum(p in 1..p, f in 1..f)
    P[p][f][r][c] * x[p][f]) == y[r][c];

12.forall(p in 1..p) sum(f in 1..f) x[p][f] <= 1;
13.}
```

7. Metaheurystyka

Jako metaheurystyka użyty został algorytm symulowanego wyżarzania. Parametry symulowanego wyżarzania:

T = 100 – temperatura początkowa,

α = 0.9 – tempo schładzania,

max_iter = 10000 – warunek stopu,

warunek akceptacji: 1 gdy $F(x_{i+1}) > F(x_i)$, $e^{\frac{F(x_{i+1})-F(x_i)}{T}}$ gdy $F(x_{i+1}) \leq F(x_i)$.

Sposób wyznaczania sąsiada: Losujemy liczbę pentomino, a następnie z każdego pentomino losujemy jedną figurę, jeśli ograniczenia nie są spełnione przechodzimy do następnej iteracji.

8. Wyniki

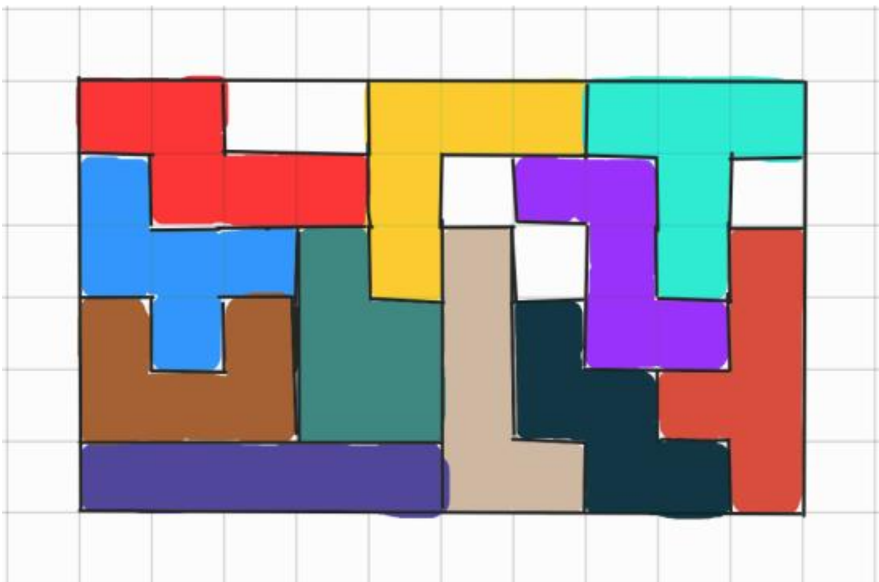
Optymalizacja została przeprowadzona przy wykorzystaniu IBM CPLEX OPL oraz PYTHON (symulowane wyżarzanie) dla następujących parametrów:
$r = 6$, $c = 10$, $p = 12$, $f = 50$.

CPLEX:

```
F(X) = 55
x = [
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
];
y = [
[1 1 1 1 1 1 1 1 1 1]
[1 1 1 1 1 0 1 0 1 0]
[1 1 1 1 1 1 1 1 1 1]
[1 1 1 1 1 1 1 0 0 1]
[1 1 1 1 1 1 1 1 1 1]
```

[1 1 1 1 1 1 1 1 1 1]];

Time = 4,52s



Rys 2. Rozwiązanie CPLEX

PYTHON:

F(X) = 30

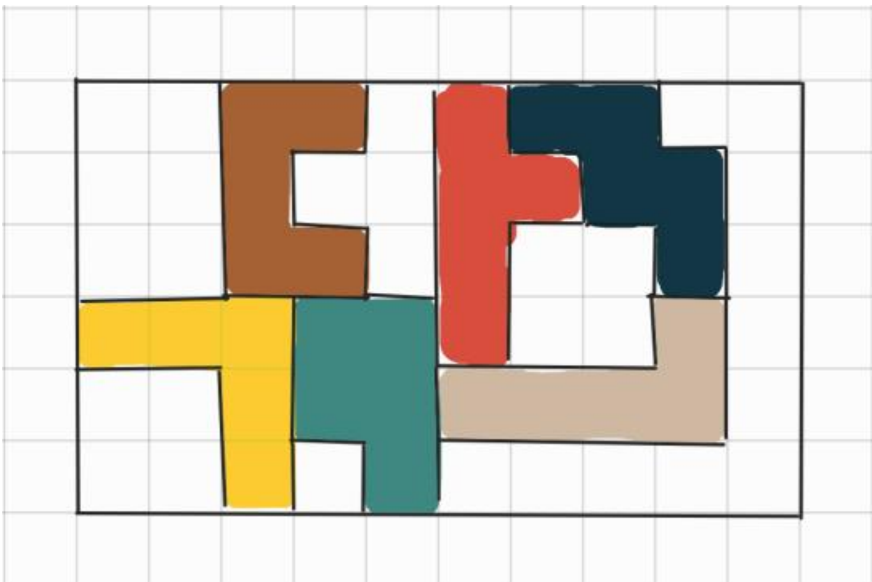
x =

[[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]

y =

[0, 0, 1, 1, 0, 1, 1, 1, 0, 0]
[0, 0, 1, 0, 0, 1, 1, 1, 1, 0]
[0, 0, 1, 1, 0, 1, 0, 0, 1, 0]
[1, 1, 1, 1, 1, 1, 0, 0, 1, 0]
[0, 0, 1, 1, 1, 1, 1, 1, 1, 0]
[0, 0, 1, 0, 1, 0, 0, 0, 0, 0]

Time = 1,38s



Rys 3. Rozwiązanie Symulowane Wyżarzanie

9. Wnioski

Udało się rozwiązać problem obiema metodami. Rozwiązanie zaprezentowane przez CPLEX okazało się wolniejsze (4,52s), ale za to skuteczniejsze, ponieważ F(X) = 55. Natomiast Symulowane Wyżarzanie w Python zajęło zaledwie 1,38s, kosztem słabszego rozwiązania F(X) = 30.

Bartłomiej Ruszaj