Podstawy Podejmowania Decyzji Projekt (część 2)

Rozwiązanie problemu: Kakuro za pomocą IBM ILOG CPLEX

Bartosz Borkowski

Spis treści

1. Wstęp
2. Sformułowanie problemu optymalizacji
   2.1 Opis modelu
   2.2 Przedstawienie i uzasadnienie zmiennych decyzyjnych
   2.3 Przedstawienie i uzasadnienie funkcji celu i ograniczeń
3. Rozwiązanie:
   3.1 Wejścia dla każdej metody
   3.2 CPLEX ILOG
4. Podsumowanie

1. Wstęp

Cel Sprawozdania

Celem niniejszego sprawozdania jest przedstawienie metodologii oraz wyników rozwiązania problemu Kakuro przy użyciu optymalizacji matematycznej za pomocą CPLEX.

Definicja Problemu

Problem Kakuro polega na uzupełnieniu kwadratowej siatki o wymiarach n x n liczbami od 1 do 9 tak, aby w każdym wierszu i każdej kolumnie każda liczba wystąpiła dokładnie raz. Dodatkowo, w określonych komórkach siatki podane są sumy wartości znajdujących się w odpowiednich wierszach lub kolumnach. Jest to problem logiczny, który wymaga zastosowania kombinatorycznych technik rozwiązywania.

Tło

Podobnie jak w przypadku "Latin Square", nazwa "Kakuro" została zaczerpnięta z zastosowania w różnych dziedzinach, takich jak łamigłówki logiczne i gry umysłowe. Kakuro jest rozszerzeniem klasycznej łamigłówki "Krzyżówki", gdzie oprócz wypełniania liczbami od 1 do 9, należy także uwzględnić sumy wartości w poszczególnych sekwencjach komórek.

	5	19	
13	4	9	4
12	1	8	3
	3	2	1

Rys. 1 Przykładowy rozwiązany kakuro 4x4

2. Sformułowanie problemu optymalizacji

2.1 Opis modelu

Model ma na celu rozwiązanie problemu Kakuro na ustalonym kwadratowym gridzie o wymiarach n x n, gdzie pewne komórki mają już określone sumy wartości. Problem Kakuro obejmuje następujące zasady:

1. Co najwyżej jedno wystąpienie każdej liczby w każdej wskazówce poziomej
2. Suma liczb równa się wartości wskazówki poziomej dla każdej wskazówki poziomej
3. Co najwyżej jedno wystąpienie każdej liczby w każdej wskazówce pionowej
4. Suma liczb równa się wartości wskazówki pionowej dla każdej wskazówki pionowej
5. Każda początkowo pusta komórka musi być wypełniona dokładnie raz
6. Każda poczżatkowo zablokowana komórka nie może zostać wypełniona

Możemy więc wyróżnić 5 zmienych wejściowych, o których decyduje użytkownik:

Rozmiar kwadratu:

$$n$$

Wskazówki poziome, ich wartość, położenie i zasięg

$$Hc = [\ (wartość, wiersz, kolumna, granica\ dolna, granica\ górna), \dots\ ]$$

Wskazówki pionowe, ich wartość, położenie i zasięg

$$Vc = [\ (wartość, wiersz, kolumna, granica\ dolna, granica\ górna)\ \dots\ ]$$

Lista Koordynatów pustych okienek

E = [(wiersz1, kolumna1) , (wiersz2, kolumna2), …]

Lista Koordynatów zablokowanych okienek

P = [(wiersz1, kolumna1) , (wiersz2, kolumna2), …]

Zaprezentuje teraz przykładowe wejście dla Kakuro z Rys 1 (w poprzedniej sekcji)

Rozmiar kwadratu:

$$n_{rys1} = 4$$

Wskazówki poziome, ich wartość, położenie i zasięg

$$Hc_{rys\ 1} = [\ (13,2,1,2,3),\quad (12,3,1,2,4),\quad (3,4,2,3,4)]$$

Wskazówki pionowe, ich wartość, położenie i zasięg

$$Vc_{rys1} = [\ (5,1,2,2,3),\quad (19,1,3,2,4),\quad (4,2,4,3,4)]$$

Lista Koordynatów pustych okienek

E_{rys1} = [(2,2), (2,3), (3,2), (3,3), (3,4), (4,3), 4,4)]

Lista Koordynatów zablokowanych okienek

P_{rys1} = [(1,1), (1,2), (1,3), (1,4), (2,1), (2,4), (3,1), (4,1), (4,2)]

Co jest optymalizowane?

IBM ILOG CPLEX jest środowiskiem, gdzie ważne są ograniczenia problemu i w tym wypadku decyzje będą podejmowane zgodnie z ograniczeniami

2.2 Przedstawienie i uzasadnienie zmiennych decyzyjnych

Zmienne decyzyjne w modelu optymalizacyjnym reprezentują decyzje, które muszą być podjęte, aby uzyskać poprawnie wypełniony kwadrat łaciński. W kontekście CPLEX zmienną decyzyjną będzie macierz (binarna) X, która informuje o zawartości danej cyfry w danej komórce
gdzie:

i – indeks wiersza, i ∈ $\{1, 2, \dots, n\}$,

j – indeks kolumny, j ∈ $\{1, 2, \dots, n\}$,

k – indeks cyfry, k ∈ $\{1, 2, \dots, 9\}$,

$x_{ijk} \in \{0,1\}$ – informacja o tym czy dana cyfra k została wpisana przez algorytm w i-tym wierszu i j-tej kolumnie.

2.3 Przedstawienie i uzasadnienie funkcji celu i ograniczeń

Ograniczenia

Jak wspomniałem w Opisie Modelu (2.1) w tym problemie mamy do czynienia z sześcioma ograniczeniami:

1. Co najwyżej jedno wystąpienie każdej liczby w każdej wskazówce poziomej
2. Suma liczb równa się wartości wskazówki poziomej dla każdej wskazówki poziomej
3. Co najwyżej jedno wystąpienie każdej liczby w każdej wskazówce pionowej
4. Suma liczb równa się wartości wskazówki pionowej dla każdej wskazówki pionowej
5. Każda początkowo pusta komórka musi być wypełniona dokładnie raz
6. Każda poczatkowo zablokowana komórka nie może zostać wypełniona

Przedstawmy je zatem matematycznie:

1. $\sum_{j=h_{lb}}^{h_{ub}} x_{ijk} \leq 1, \quad \forall\, k \in \{1, \dots, 9\}$ oraz i $= i_{hc}$

2. $\sum_{k=1}^{9} \sum_{j=h_{lb}}^{h_{ub}} x_{ijk} * k = c_h$, i $= i_{hc}$

3. $\sum_{i=v_{lb}}^{v_{ub}} x_{ijk} \leq 1, \quad \forall\, k \in \{1, \dots, 9\}$ oraz j $= j_{vc}$

4. $\sum_{k=1}^{9} \sum_{i=v_{lb}}^{v_{ub}} x_{ijk} * k = c_v$, j $= j_{vc}$

5. $\sum_{k=1}^{9} x_{ijk} = 1, \quad \forall\, (i,j) \in$ E

6. $\sum_{k=1}^{9} x_{ijk} = 0, \quad \forall\, (i,j) \in$ P

Gdzie

h_{ub} – wartość j górnej granicy poziomej wskazówki

h_{lb} – wartość j dolnej granicy poziomej wskazówki

i_{hc} – wartość i na której jest pozioma wskazówka

c_h – wartość sumy poziomej wskazówki

v_{ub} – wartość i górnej granicy pionowej wskazówki

v_{lb} – wartość i górnej granicy pionowej wskazówki

i_{vc} – wartość j na której jest pozioma wskazówka

c_v – wartość sumy poziomej wskazówki

Funkcja Celu

Jak wytłumaczyłem wcześniej w ***„Co jest optymalizowane?”*** W środowisku CPLEX nie jest potrzebna funkcja celu dla tego problemu.

3. Rozwiązanie

3.1 Wejścia dla każdej metody

Wykonam 4 wejścia dla każdej z metod.
Będą one wyglądać następująco:

Rozmiar kwadratu:

$$n_1 = 3$$

Wskazówki poziome, ich wartość, położenie i zasięg

$$Hc_1 = [\,(14,2,1,2,3),\qquad (8,3,1,2,3)]$$

Wskazówki pionowe, ich wartość, położenie i zasięg

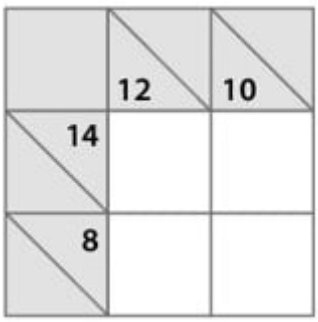
$$Vc_1 = [\,(12,1,2,2,3),\qquad (10,1,3,2,3)]$$

Lista Koordynatów pustych okienek

E_1 = [(2,2), (2,3), (3,2), (3,3]

Lista Koordynatów zablokowanych okienek

P_1 = [(1,1), (1,2), (1,3), (2,1), (3,1)]



Rys 3.1.1 zwizualizowany kakuro 1

Rozmiar kwadratu:

$$n_2 = 4$$

Wskazówki poziome, ich wartość, położenie i zasięg

$$Hc_2 = [\,(20,2,1,2,4),\quad (18,3,1,2,4),\quad (7,4,1,2,4)]$$

Wskazówki pionowe, ich wartość, położenie i zasięg

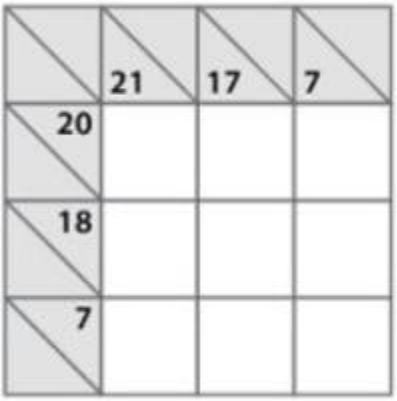
$$Vc_2 = [\,(21,1,2,2,4),\quad (17,1,3,2,4),\quad (10,1,4,2,4)]$$

Lista Koordynatów pustych okienek

E_2 = [(2,2), (2,3), (2,4), (3,2), (3,3), (3,4), (4,2), (4,3), (4,4)]

Lista Koordynatów zablokowanych okienek

P_2 = [(1,1), (1,2), (1,3), (1,4), (2,1), (3,1), (4,1)]



Rys 3.1.2 zwizualizowany kakuro 2

Rozmiar kwadratu:

$$n_3 = 5$$

Wskazówki poziome, ich wartość, położenie i zasięg

$Hc_3 = [\ (9,2,2,3,4),\quad (28,3,1,2,5),\quad (22,4,1,2,5),\quad (7,5,2,3,4)]$

Wskazówki pionowe, ich wartość, położenie i zasięg

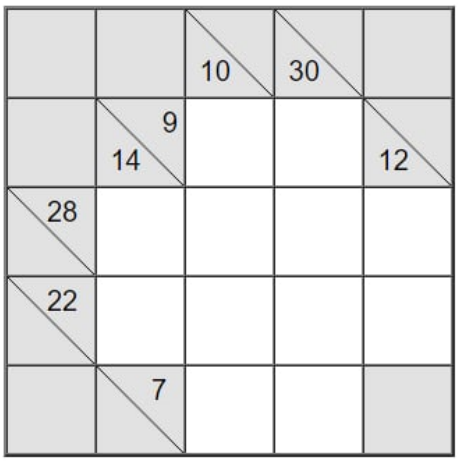
$Vc_3 = [\ (14,2,2,3,4),\quad (10,1,3,2,5),\quad (30,1,4,2,5),\quad (12,2,5,3,4)]$

Lista Koordynatów pustych okienek

E_3 = [(2,3), (2,4), (3,2), (3,3), (3,4), (3,5), (4,2), (4,3), (4,4), (4,5), (5,3), (5,4)]

Lista Koordynatów zablokowanych okienek

P_3 = [(1,1), (1,2), (1,3), (1,4), (1,5) (2,1), (2,2), (2,5), (3,1), (4,1), (5,1), (5,2), (5,5)]



Rys 3.1.3 zwizualizowany kakuro 3

Tym razem kakuro będzie prostokątem 9x8, jako przykład zaawansowanego kakuro

Rozmiar **Prostokątu**:

$$a = 9 \qquad b = 8$$

Wskazówki poziome, ich wartość, położenie i zasięg

$Hc_4 = [\ (8,2,3,4,5),\ (28,3,1,2,7),\ (12,4,1,2,4),\ (6,4,5,6,7),\ (7,5,4,5,6),\ (9,6,3,4,5),\ (7,7,2,3,4),\ (17,7,6,7,8),\ (26,8,2,3,8),\ (16,9,4,5,6)]$

Wskazówki pionowe, ich wartość, położenie i zasięg

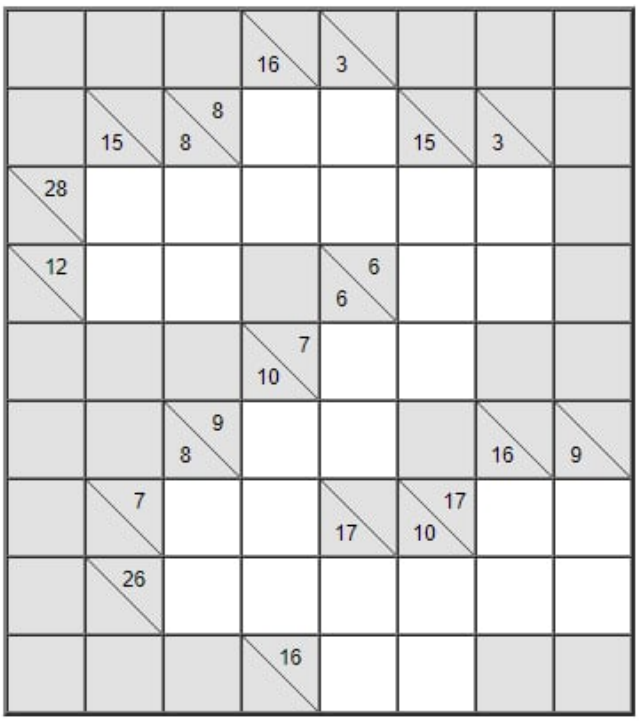
$Vc_4 = [\ (16,1,4,2,3),\ (3,1,5,2,3),\ (15,2,2,3,4),\ (8,2,3,3,4),\ (15,2,6,3,5),\ (3,2,7,3,4),\ (6,4,5,5,6),\ (10,5,4,6,8),\ (8,6,3,7,8),\ (16,6,7,7,8),\ (9,6,8,7,8),\ (17,7,5,8,9),\ (10,7,6,8,9)]$

Lista Koordynatów pustych okienek

E_4 =
[(2,4),(2,5),(3,2),(3,3),(3,4),(3,5),(3,6),(3,7),(4,2),(4,3),(4,6),(4,7),(5,5),(5,6),(6,4),(6,5),(7,3),(7,4), (7,7), (7,8), (8,3),(8,4),(8,5),(8,6),(8,7),(8,8),(9,5),(9,6)]

Lista Koordynatów zablokowanych okienek

P_4 = [(1,1), (1,2), (1,3), (1,4), (1,5),(1,6),(1,7),(1,8), (2,1), (2,2), (2,3) (2,6),(2,7),(2,8), (3,1), (3,8) (4,1), (4,4),(4,5),(4,8), (5,1), (5,2), (5,3), (5,4), (5,7), (5,8), (6,1), (6,2), (6,3), (6,6),(6,7),(6,8), (7,1),(7,2), (7,5),(7,6), (8,1),(8,2), (9,1),(9,2),(9,3),(9,4),(9,7),(9,8)]



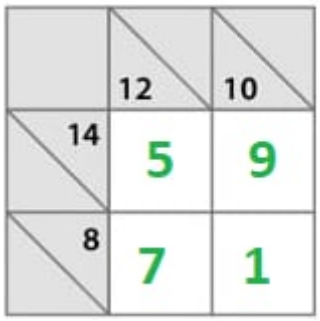
Rys 3.1.4 zwizualizowany kakuro 4

3.2 CPLEX ILOG

Tutaj znajdują się rozwiązane 4, wcześniej przedstawione wejścia, przedstawie je graficznie.
X_z^* będzie oznaczał wynikową macierz rozwiązania numer z

X_1^* =

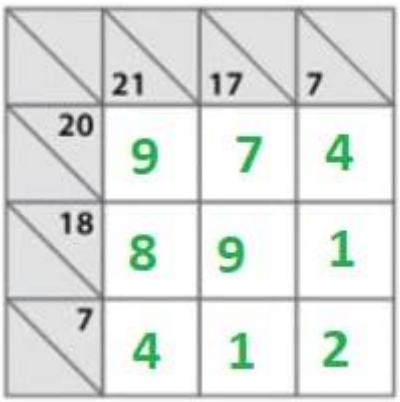
```
[[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]]
 [[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]]
 [[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0 0]]]
```



Rys 3.2.1 zwizualizowana odpowiedź 1

X_2^* =

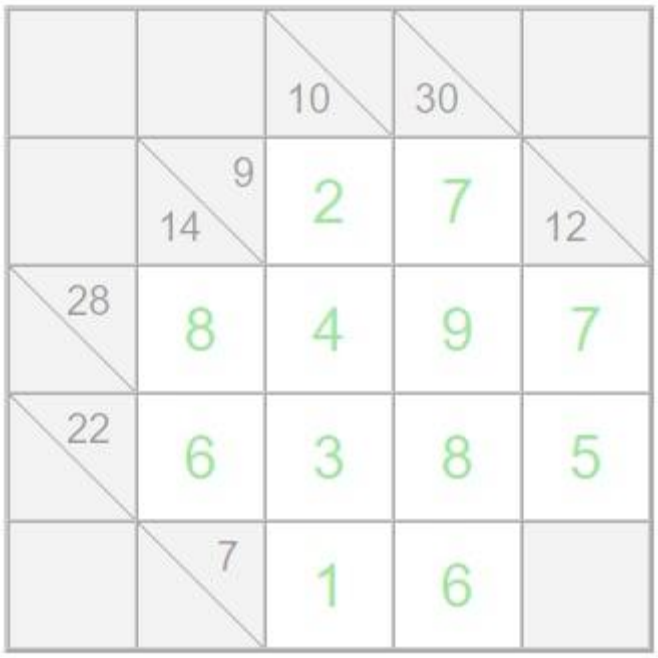
```
[[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]]
 [[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 1 0 0]
 [0 0 0 1 0 0 0 0 0]]
 [[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 1]
 [1 0 0 0 0 0 0 0 0]]
 [[0 0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 0 0 0]]]
```



Rys 3.2.2 zwizualizowana odpowiedź 2

X_3^* =

```
[[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0]
 [0 0 0 0 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 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 1 0]
 [0 0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 1]
 [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 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 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 1 0 0 0]
 [0 0 0 0 0 0 0 0 0]]]
```



Rys 3.2.3 zwizualizowana odpowiedz 3

			16	3			
	15	8 8	7	1	15	3	
28	8	3	9	2	5	1	
12	7	5		6 6	4	2	
			7 10	1	6		
		9 8	4	5		16	9
	7	6	1	17	17 10	9	8
	26	2	5	8	3	7	1
			16	9	7		

Rys 3.2.4 zwizualizowana odpowiedź 4

4. Podsumowanie

Kakuro jest problemem w postaci gry logiczej, w którym mamy ograniczony zbiór cyfr a same cyfry muszą się sumować w wierszu i w kolumnie do określonych liczb, podanych jako wskazówki.

W tym sprawozdaniu przedstawiłem rozwiązanie kakuro za pomocą CPLEX ILOG, program poradził sobie bardzo dobrze z tym problemem, poprzez rozwiązanie każdego z zadanych mu wejść. Ostatnie bardziej skomplikowane wejście zajęło mu o więcej czasu ale dalej było to relatywnie mało czasu (6 min).

Na koniec tego sprawozdania jak i poprzedniego można wyciągnąć pare wniosków na temat samego narzędzia CPLEX:

- Bardzo dobrze rozwiązuje gry logiczne z jasno ustalonymi zasadami
- Potrafi znaleźć poprawne rozwiązania bez podawania jakiegokolwiek kryterium jakości (funkcji celu).
- Bardzo dobrze odnajduje błędnie zainicjowane wejścia
- Znajduje odpowiedzi na problemu równie szybko co metaheurystyki, a jednocześnie według ustalonego w poprzednim sprawozdaniu kryterium jakości robi to lepiej.