Sprawozdanie z projektu 2 MOM Dawid Bartosiak

1. Wstęp

Celem projektu jest opracowanie modelu optymalizacyjnego, który minimalizuje dzienne koszty dystrybucji produktów od zakładów produkcyjnych do punktów sprzedaży detalicznej. Transport odbywa się z wykorzystaniem magazynów hurtowych. W sprawozdaniu przedstawiono zarówno analizę problemu, jak i implementację modelu w języku AMPL. Model został rozwiązany przy użyciu dwóch solverów – CPLEX i Gurobi dając takie same rezultaty z zastrzeżeniem na dokładność numeryczną uzyskanego rozwiązania.

2. Opis problemu

Mamy dwa zakłady produkcyjne:

- **Zakład W1** – maksymalna produkcja: 61 jednostek P1 i 51 jednostek P2
- **Zakład W2** – maksymalna produkcja: 113 jednostek P1 i 108 jednostek P2

Produkty są transportowane do czterech punktów sprzedaży detalicznej: S1, S2, S3 i S4 przy pomocy trzech magazynów hurtowych:

- **Magazyn M1** – dostępny w dwóch wariantach: mały (pojemność 5 jednostek, koszt 20 tys. zł) lub duży (pojemność 125 jednostek, koszt 448 tys. zł)
- **Magazyn M2** – możliwy wariant: nie budowany (0 jednostek, koszt 0), mały (7 jednostek, koszt 24 tys. zł) lub duży (144 jednostek, koszt 672 tys. zł)
- **Magazyn M3** – budowa modułowa, gdzie każdy moduł ma pojemność 14 jednostek, a koszt operacyjny jednego modułu wynosi 18 tys. zł

Jednostkowe koszty transportu są podane w tabelach:

c_{ki}	M1	M2	M3
W1	5	3	6
W2	9	6	2

t_{ij}	S1	S2	S3	S4
M1	10	15	15	5
M2	15	1	10	2
M3	2	2	2	7

Wymagania dotyczące zapotrzebowania dla każdego produktu w punktach sprzedaży przedstawione są w tabeli:

b_{rj}	S1	S2	S3	S4
P1	25	40	36	31
P2	36	36	36	29

3. Oznaczenia

3.1. Dane

Zbiory:

plants – zbiór zakładów wytwórczych (W1, W2)

products – zbiór produktów (P1, P2)

warehouses – zbiór magazynów (M1, M2, M3)

stores – zbiór punktów sprzedaży (S1, S2, S3, S4)

$sizes_i$– zbiór możliwych rozmiarów magazynów i

dane liczbowe:

$max\_production_{i,j}$ – maksymalna produkcja towaru j w zakładzie i ≥ 0

$warehouse\_capacity_{i,j}$ – maksymalna pojemność magazynu i dla rozmiaru j ≥ 0

$warehouse\_cost_{i,j}$ – koszt rozmiaru j dla magazynu i ≥ 0

$demand_{i,j}$ – zapotrzebowanie na towar i w punkcie j ≥ 0

$plant\_to\_warehouse\_cost_{i,j}$ – koszt transportu z zakładu i do magazynu j ≥ 0

$warehouse\_to\_store\_cost_{i,j}$ – koszt transportu z magazynu i do punktu j ≥ 0

module_capacity – pojemność pojedynczego modułu (dla magazynu M3) ≥ 0

module_cost – koszt pojedynczego modułu (dla magazynu M3) ≥ 0

3.2. Zmienne decyzyjne

W modelu wykorzystano następujące zmienne decyzyjne:

x_{ijk} – przepływ między zakładem i a magazynem j produktu k (zmienna ciągła) ≥ 0

y_{ijk} – przepływ między magazynem i a punktem sprzedaży j produktu k (zmienna ciągła) ≥ 0

z_{ij} – decyzji inwestycji w magazyn i o rozmiarze j (zmienna binarna) ∈ $\{0, 1\}$

$modules$ – ilość modułów dla magazynu M3 (zmienna całkowita) ≥ 0

4. Model matematyczny

4.1. Ograniczenia modelu

W modelu znajdują się następujące grupy ograniczeń:

Ograniczenia produkcyjne w zakładach:

$$\bigwedge_{i}^{plants} \bigwedge_{j}^{products} \sum_{k}^{warehouses} x_{ikj} \leq max\_production_{ij}$$

Ograniczenia przepływu w magazynach:

$$\bigwedge_{i}^{warehouses} \bigwedge_{j}^{products} \sum_{k}^{plants} x_{kij} \leq \sum_{k}^{stores} y_{ikj}$$

Ograniczenia pojemności magazynowych:

$$\bigwedge_{i}^{warehouses:\, i!="M3"} \sum_{j}^{plants} \sum_{k}^{products} x_{jik} \leq \sum_{j}^{sizes} warehouse\_capacity_{ij} \cdot z_{ij}$$

$$\sum_{i}^{plants} \sum_{j}^{products} x_{i"M3"j} \leq module\_capacity \cdot modules$$

Ograniczenia popytowe:

$$\bigwedge_{i}^{stores} \bigwedge_{j}^{products} \sum_{k}^{warehouses} y_{kij} = demand_{ji}$$

Ograniczenia wyboru opcji budowy magazynów:

$$\bigwedge_{i}^{warehouses:\, i!="M3"} \sum_{j}^{sizes} z_{ij} = 1$$

4.2. Funkcja celu

Funkcja celu polega na minimalizacji łącznych dziennych kosztów dystrybucji, które są sumą:

Kosztów transportu: od zakładów do magazynów oraz od magazynów do punktów sprzedaży:

$$\sum_{i}^{plants} \sum_{j}^{warehouses} \sum_{k}^{products} plant\_to\_warehouse\_cost_{ij} \cdot x_{ijk}$$

$$\sum_{i}^{warehouses} \sum_{j}^{stores} \sum_{k}^{products} warehouse\_to\_store\_cost_{ij} \cdot y_{ijk}$$

Kosztów operacyjnych magazynów zależnych od wybranej pojemności lub liczby modułów:

$$\sum_{i}^{warehouses:\, i!="M3"} \sum_{j}^{sizes} warehouse\_cost_{ij} \cdot z_{ij}$$

$$modules\_cost \cdot modules$$

Całość (suma powyższych składników) jest minimalizowana.

5. Implementacja modelu w AMPL

Poniżej znajduje się przykładowy kod w AMPL, który implementuje powyższy model.

Kod uruchomieniowy:

```
model main.mod;
data main.dat;

option solver gurobi;
solve;

display TotalCost;

display z;
display modules;

display x;
display y;

printf "Całkowity koszt magazynowania: %g\n",
    (sum {w in WAREHOUSES, s in SIZES[w]: w != "M3"} warehouse_cost[w,s] * z[w,s]) +
    (module_cost * modules);

printf "Całkowity koszt transportu: %g\n",
    (sum {p in PLANTS, w in WAREHOUSES, r in PRODUCTS} plant_to_warehouse_cost[p,w] * x[p,w,r]) +
    (sum {w in WAREHOUSES, s in STORES, r in PRODUCTS} warehouse_to_store_cost[w,s] * y[w,s,r]);

printf "\nUżycie magazynu:\n";
for {w in WAREHOUSES} {
    printf "Magazyn %s: ", w;
    if w == "M3" then {
        printf "%g jednostek (pojemność: %g)\n",
            sum {p in PLANTS, r in PRODUCTS} x[p,w,r],
            module_capacity * modules;
    } else {
        printf "%g jednostek (pojemność: %g)\n",
            sum {p in PLANTS, r in PRODUCTS} x[p,w,r],
            sum {s in SIZES[w]} warehouse_capacity[w,s] * z[w,s];
    }
}
```

Kod modelu:

```
# Sets
set PLANTS;         # Plants (W1, W2)
set PRODUCTS;       # Products (P1, P2)
set WAREHOUSES;     # Warehouses (M1, M2, M3)
set STORES;         # Retail stores (S1, S2, S3, S4)
set SIZES {WAREHOUSES}; # Possible sizes for each warehouse

# Parameters
param max_production {PLANTS, PRODUCTS} >= 0;
param warehouse_capacity {w in WAREHOUSES, s in SIZES[w]} >= 0;
param warehouse_cost {w in WAREHOUSES, s in SIZES[w]} >= 0;
param module_capacity > 0;
param module_cost > 0;
param demand {PRODUCTS, STORES} >= 0;
param plant_to_warehouse_cost {PLANTS, WAREHOUSES} >= 0;
param warehouse_to_store_cost {WAREHOUSES, STORES} >= 0;

# Variables
var x {p in PLANTS, w in WAREHOUSES, r in PRODUCTS} >= 0;
var y {w in WAREHOUSES, s in STORES, r in PRODUCTS} >= 0;
var z {w in WAREHOUSES, s in SIZES[w]} binary;
var modules >= 0, integer;

# Objective function: Minimize total daily cost
minimize TotalCost:
    # Transportation costs from plants to warehouses
    sum {p in PLANTS, w in WAREHOUSES, r in PRODUCTS}
        plant_to_warehouse_cost[p,w] * x[p,w,r] +
    # Transportation costs from warehouses to stores
    sum {w in WAREHOUSES, s in STORES, r in PRODUCTS}
        warehouse_to_store_cost[w,s] * y[w,s,r] +
    # Warehouse operational costs for M1 and M2
    sum {w in WAREHOUSES, s in SIZES[w]: w != "M3"}
        warehouse_cost[w,s] * z[w,s] +
    module_cost * modules;

# Constraints
subject to ProductionCapacity {p in PLANTS, r in PRODUCTS}:
    sum {w in WAREHOUSES} x[p,w,r] <= max_production[p,r];

subject to DemandSatisfaction {s in STORES, r in PRODUCTS}:
    sum {w in WAREHOUSES} y[w,s,r] = demand[r,s];

subject to FlowConservation {w in WAREHOUSES, r in PRODUCTS}:
    sum {p in PLANTS} x[p,w,r] = sum {s in STORES} y[w,s,r];

subject to WarehouseCapacity {w in WAREHOUSES: w != "M3"}:
    sum {p in PLANTS, r in PRODUCTS} x[p,w,r] <=
    sum {s in SIZES[w]} warehouse_capacity[w,s] * z[w,s];

subject to WarehouseCapacityM3:
    sum {p in PLANTS, r in PRODUCTS} x[p,"M3",r] <= module_capacity * modules;

subject to OneSizePerWarehouse {w in WAREHOUSES: w != "M3"}:
    sum {s in SIZES[w]} z[w,s] = 1;
```

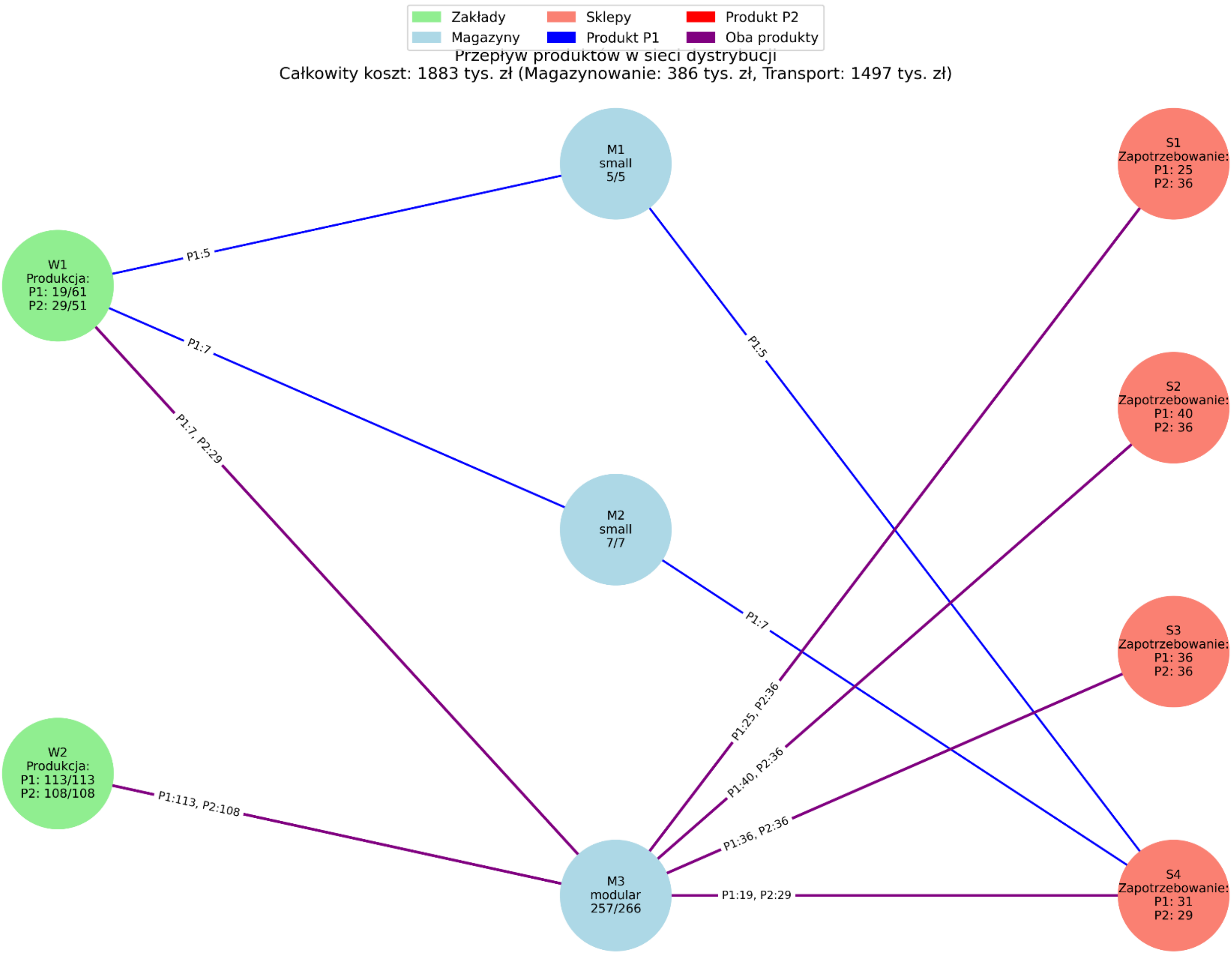
Kod danych:

```
set PLANTS := W1 W2;
set PRODUCTS := P1 P2;
set WAREHOUSES := M1 M2 M3;
set STORES := S1 S2 S3 S4;

set SIZES[M1] := small large;  # 5, 125 jedn
set SIZES[M2] := none small large;  # 0, 7, 144 jedn
set SIZES[M3] := dummy;  # Dummy bo M3 używa modułów

param max_production:
      P1   P2 :=
  W1  61   51
  W2  113  108 ;

param warehouse_capacity:
          small  large  none  dummy :=
  M1      5      125    .     .
  M2      7      144    0     .
  M3      .      .      .     0      ;  # M3 używa modułów

param warehouse_cost:
          small  large  none  dummy :=
  M1      20     448    .     .
  M2      24     672    0     .
  M3      .      .      .     0      ;  # M3 używa modułów

param module_capacity := 14;

param module_cost := 18;

param demand:
      S1   S2   S3   S4  :=
  P1  25   40   36   31
  P2  36   36   36   29  ;

param plant_to_warehouse_cost:
      M1   M2   M3  :=
  W1  5    3    6
  W2  9    6    2   ;

param warehouse_to_store_cost:
      S1   S2   S3   S4  :=
  M1  10   15   15   5
  M2  15   1    10   2
  M3  2    2    2    7   ;
```

6. Wyniki

Po rozwiązaniu modelu przy użyciu wybranego solvera (np. CPLEX lub Gurobi) uzyskaliśmy następujące wyniki:

```
TotalCost = 1883

z :=
M1 large   0
M1 small   1
M2 large   0
M2 none    0
M2 small   1
M3 dummy   0
;

modules = 19

x :=
W1 M1 P1     5
W1 M1 P2     0
W1 M2 P1     7
W1 M2 P2     0
W1 M3 P1     7
W1 M3 P2    29
W2 M1 P1     0
W2 M1 P2     0
W2 M2 P1     0
W2 M2 P2     0
W2 M3 P1   113
W2 M3 P2   108
;

y [*,*,P1] (tr)
:    M1   M2    M3     :=
S1    0    0    25
S2    0    0    40
S3    0    0    36
S4    5    7    19

 [*,*,P2] (tr)
:    M1   M2    M3     :=
S1    0    0    36
S2    0    0    36
S3    0    0    36
S4    0    0    29
;

Całkowity koszt magazynowania: 386
Całkowity koszt transportu: 1497

Użycie magazynu:
Magazyn M1: 5 jednostek (pojemność: 5)
Magazyn M2: 7 jednostek (pojemność: 7)
Magazyn M3: 257 jednostek (pojemność: 266)
```



7. Podsumowanie

W opracowanym modelu stworzono model mieszany liniowy-całkowitoliczbowy, którego celem jest minimalizacja łącznych kosztów dystrybucji przy jednoczesnym spełnieniu ograniczeń produkcyjnych, popytowych i magazynowych. Model odzwierciedla rzeczywiste problemy logistyczne, w których konieczne jest podjęcie decyzji zarówno o przepływach produktów, jak i wyborze optymalnych rozwiązań magazynowych. Zastosowane podejście umożliwia elastyczne modyfikowanie parametrów co ułatwia zmianę założeń początkowych. Wyniki osiągnięte przez obydwa solvery (CPLEX i GUROBI) są takie same, z zastrzeżeniem, że CPLEX przedstawia część zerowych wyników jako wyniki bardzo bliskie zeru ($1e^{-9}$), a GUROBI lepiej pozbywa się tego rodzaju szumu.

Ostateczne wyniki (wartość funkcji celu oraz wartości zmiennych) pokazują, że model został rozwiązany optymalnie, co potwierdzają zarówno wyniki obliczeniowe, jak i zgodność z warunkami zadania.