Optymalizacja wielokryterialna

Metody klasyczne

1 Wstęp

Wiele rzeczywistych problemów optymalizacji jest opisanych przez więcej niż jedną funkcję celu, tj. kryterium optymalizacji. Funkcje te winny być optymalizowane jednocześnie. Przykładowo:

- w logistycznym problemie konstrukcji łańcucha dostaw kryteriami mogą być **całkowity koszt** realizacji dostaw oraz **całkowity czas** ich realizacji;
- w inżynieryjnym problemie konstrukcji silnika optymalizowana może być jego **wydajność** oraz **zużycie paliwa**.

W wymienionych przykładach wydajność jest kryterium typu **zysk**, natomiast pozostałe przypadki to kryteria typu **koszt**. Jeżeli problem optymalizacji jest opisany przez więcej niż jedno kryterium, jest on nazywany **problemem optymalizacji wielokryterialnej**.

Fundusze Europejskie Polska Cyfrowa | Rzeczpospolita Polska | Unia Europejska Europejski Fundusz Rozwoju Regionalnego

„Akademia Innowacyjnych Zastosowań Technologii Cyfrowych (AI Tech)”, projekt finansowany ze środków Programu Operacyjnego Polska Cyfrowa POPC.03.02.00-00-0001/20

Optymalizowane kryteria są często niestety w pewnym stopniu sprzeczne. Z tego powodu, niemożliwym może okazać się konstrukcja takiego rozwiązania, które było by najlepsze z punktu widzenia wszystkich kryteriów równocześnie. Na pewnym etapie optymalizacji, rozwiązanie może już nie być w stanie być poprawione z punktu widzenia pewnego podzbioru kryteriów, bez jednoczesnego pogorszenia z punktu widzenia innych kryteriów. Rozwiązania takie nazywane są **optymalnymi w sensie Pareta (Pareto optymalne)**, a zbiór wszystkich takich rozwiązań problemu nazywany jest **frontem Pareta**. Rozwiązania Pareto optymalne stanowią najlepsze możliwe do uzyskania kompromisy pomiędzy kryteriami, stanowiąc tym samym możliwe najlepsze warianty, tj. potencjalne rozwiązania problemu, dla decydenta.

$$\min \operatorname{lub} \max \mathrm{f}_1\,(\mathrm{x}) = \mathrm{s}_1,$$

$$\min \operatorname{lub} \max \mathrm{f}_2\,(\mathrm{x}) = \mathrm{s}_2,$$

...

$$\min \operatorname{lub} \max \mathrm{f}_M\,(\mathrm{x}) = \mathrm{s}_M,$$

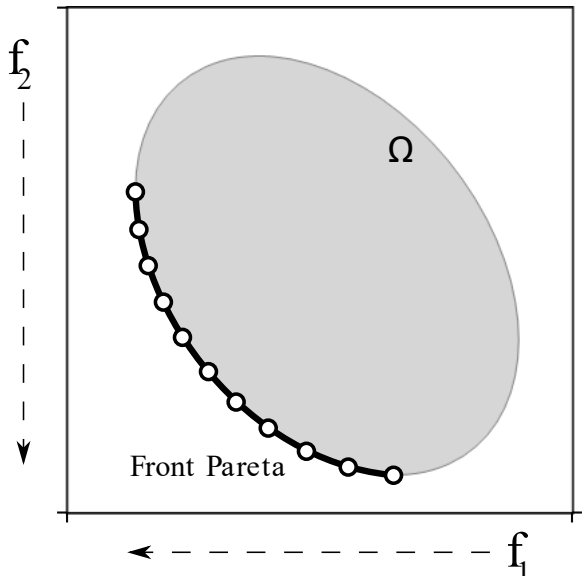
$$\text{przy ograniczeniach } x \in \mathcal{X}, s = [s_1, s_2, \dots, s_M] \in \Omega,$$

gdzie f_{1-M} to M funkcji celu, które winny być maksymalizowane bądź minimalizowane, $\boldsymbol{x}$ to rozwiązanie w przestrzeni decyzji, a $\boldsymbol{s}$ to rozwiązanie w przestrzeni ocen. Pareto front tak sformułowanego problemu jest zdefiniowany w następujący sposób:

$$PF = \left\{\boldsymbol{s}^j \in \Omega: \nexists_{\boldsymbol{s}^k \in \Omega, \boldsymbol{s}^k \neq \boldsymbol{s}^j} \boldsymbol{s}^k \Delta \boldsymbol{s}^j\right\},$$

Gdzie Δ to relacja dominacji, która zachodzi dla pary $\left(\boldsymbol{s}^j, \boldsymbol{s}^k\right)$, gdy $\forall_{i=1,\dots,M} s_i^j \leq s_i^k$ oraz $\exists_{i\in\{1,\dots,M\}} s_i^j < s_i^k$. (dla kryterium typu zysk relacje powinny być

odpowiednio $\geq$ oraz $>$). Jako że rozwiązań Pareto optymalnych może być w ogólności nieskończenie wiele, przyjmuje się, że ogólnym zadaniem optymalizacji wielokryterialnej jest konstrukcja zbioru rozwiązań **możliwie najlepiej przybliżających front Pareta**. Rysunek 1 przedstawia przykładową przestrzeń rozwiązań Ω dla problemu 2-kryteriowego. Front Pareto został zaznaczony przy użyciu pogrubionej linii, a jego przykładowe przybliżenie przy użyciu kropek.



Rysunek 1: Przykładowa przestrzeń Omega i front Pareta

2 Programowanie liniowe

Programowanie liniowe jest sformalizowanym sposobem opisu problemu optymalizacji **przy wykorzystaniu funkcji liniowych**. Zakłada się, że rozwiązania są opisane przy użyciu n zmiennych decyzyjnych $\boldsymbol{x} = [x_1, x_2, \dots, x_n]$, które mogą być uwikłane w funkcjach liniowych do określenia funkcji celu bądź narzucenia ograniczeń na przestrzeń decyzyjną. Przykładowe sformułowanie problemu

optymalizacji przy wykorzystaniu programowania linowego może wyglądać następująco:

$$\max 2x_1 + 3x_2$$

przy ograniczeniach:

$$x_1 - 2x_2 \leq 15$$

$$3x_1 + x_2 \leq 10$$

$$x_1, x_2 \geq 0$$

Powyższy przykład maksymalizuje funkcję (1), narzucając ograniczenia (1-4) na zmienne decyzyjne x_1 oraz x_2. Do rozwiązywania tak sformułowanych problemów wykorzystuje się metody oparte o algorytm Simpleks. Poza ograniczeniami typu$\leq$, możliwe jest również wykorzystanie ograniczeń typu $=$ oraz $\geq$. Dodatkowo można również ograniczyć dziedzinę zmiennych do liczb całkowitych. Mówi się wtedy o programowaniu mieszanym liniowym bądź całkowitoliczbowym. Zawarcie zmiennych całkowitoliczbowych na ogół znacząco zwiększa złożoność obliczeniową algorytmu.

Programowanie liniowe zakłada optymalizację **jednej funkcji celu**. By móc wykorzystać tę metodę do przybliżenia frontu Pareta problemu wielokryterialnego, wykorzystuje się najczęściej podejścia sekwencyjne, które przy każdym jednorazowym uruchomieniu algorytmu starają się odkryć inne rozwiązanie Pareto optymalne. Dwoma klasycznymi takimi algorytmami są Metoda Sumy Ważonej oraz Metoda Epsilon-Ograniczeń.

Metoda Sumy Ważonej

Metoda ta agreguję M funkcji celu w jedną funkcję przy użyciu wag $\boldsymbol{w}$, które określają istotność każdego kryterium. Innymi słowy:

$$f^S = w_1 f_1(x) + w_2 f_2(x), \dots, w_M f_M(x).$$

Tak sformułowana funkcja f^S jest wykorzystana w modelu programowania liniowego jako funkcja celu. Przy zachowaniu takich samych ograniczeń, problem ten jest rozwiązywany **dla różnych kombinacji wag celem odnalezienia różnych rozwiązań Pareto optymalnych**. Wagi te powinny sumować się do 1. Ponadto ważne jest to, aby funkcje celu f były znormalizowane, chociaż lepszym podejściem jest uwzględnienie takiej normalizacji przy definiowaniu samych wag.

Zaletą Metody Sumy Ważonej jest jej stosunkowo prosta implementacja. Wady są natomiast następujące:

- Trudno jest uzyskać równomierny rozkład uzyskanych rozwiązań Pareto optymalnych, ponieważ to w jaki sposób wektory wag powinny być wygenerowane by uzyskać taki efekt, silnie zależy od samego kształtu frontu Pareta (który nie jest znany *a priori*).
- Metoda ta nie jest w stanie odnaleźć rozwiązań na płaskich bądź wklęsłych obszarach frontu Pareta.
- Gdy problem jest dyskretny, wiele różnych wektorów wag może wskazywać to samo rozwiązanie jako optymalne, tym samym istotnie wzrasta liczba uruchomień algorytmu celem uzyskania zadanej liczby różnych rozwiązań Pareto optymalnych.

Metoda Epsilon Ograniczeń

W przeciwieństwie do Metody Sumy Ważonej, Metoda Epsilon-Ograniczeń optymalizuje wybraną – oryginalną – funkcję celu. By uzyskać różne rozwiązania Pareto optymalne, stopniowo ogranicza ona przy każdym kolejnym uruchomieniu pozostałe kryteria.

Dla przykładu, jeżeli problem jest opisany przy użyciu dwóch kryteriów, które winny być minimalizowane, metoda może optymalizować tylko funkcję f_1. Przy pierwszym uruchomieniu algorytmu, znalezione rozwiązanie będzie możliwie najlepiej realizowało właśnie tę funkcję. Natomiast zakładając, że f_1 i f_2 są w silnym konflikcie, znalezione rozwiązanie powinno być przeciętne z punktu widzenia f_2. Wiedząc, że rozwiązanie Pareto optymalne nie może zostać poprawione na wybranych kryteriach bez jednoczesnego pogorszenia na innych, ograniczenie $f_2(\boldsymbol{x}) \leq \varepsilon$ może zostać dodane do modelu, co z jednej strony ograniczy możliwą realizację tego kryterium, a z drugiej strony spowoduje pogorszenie realizacji uzyskanego rozwiązania z punktu widzenia f_1. Tym samym stopniowo narzucając coraz większe ograniczenia ε na f_2, metoda może odnaleźć różne rozwiązania Pareto optymalne.

Do zdefiniowania ε-ograniczeń dla powyższego przykładu, można odnaleźć minimum oraz maksimum dla wartości f_2 rozwiązań Pareto optymalnych. Można to zrobić minimalizują raz f_1 a raz f_2. Dla minimalizacji f_1 można założyć, że odnaleziona zostanie najgorsza wartość dla f_2. Następnie tak zdefiniowany przedział może zostać podzielony na kilka równych części, które wyznaczą kolejne wartości ograniczeń dla tej funkcji.

Zaletą Metody ε -ograniczeń jest to, że jest wstanie odnaleźć dowolne rozwiązanie Pareto optymalne. Ponadto na ogół wymaga ona znacznie mniej uruchomień do

odnalezienia zbioru o – zadanej wielkości – różnych rozwiązań Pareto optymalnych, w porównaniu do Metody Sumy Ważonej. Wadą natomiast jest to, że często uzyskane rozwiązania nie są równomiernie rozłożone w przestrzeni funkcji celu.

3 Zadanie Domowe

Firma Pfitzer jest jednym z wiodących producentów i dystrybutorów leków na świecie. Bezpośrednimi klientami firmy są lekarze, których regularnie odwiedzają przedstawiciele handlowi Pfitzera. Zadanie domowe skupia się tylko na małym skrawku działań operacyjnych firmy, jakim jest rejon Istambułu w Turcji. W tym obszarze operuje 4 przedstawicieli firmy, a sam obszar jest podzielony na 22 mniejsze jednostki geograficzne. Każdy przedstawiciel ma swoją stałą siedzibę w centrum jednego z regionów. Dla każdej takiej jednostki geograficznej utrzymywane są dane dotyczące sprzedaży, liczba lekarzy oraz ich profile, i na podstawie tych danych obliczany jest indeks, który odzwierciedla „pracochłonność" związaną z działaniami operacyjnymi w tym rejonie.

Firma Pfitzer stara się wyznaczyć swoim przedstawicielom obszary działań operacyjnych w taki sposób, aby, po pierwsze, każdy region miał przypisanego swojego przedstawiciela i po drugie, aby całkowita pracochłonność związana z regionami przedzielonymi każdemu z przedstawicieli była podobna i bliska 1. Ostatni taki przydział miał miejsce wiele lat temu i obecnie firma zastanawia się, czy po sukcesywnych zmianach wskaźników dotyczących regionów, które miały miejsce na przestrzeni czasu, możliwa jest lepsza alokacja swoich przedstawicieli. Ogólne zadania, kryteria, rozważane w tym problemie są następujące:

- **Minimalizacja odległości f_1**: Aby podróże przedstawicieli firmy do przypisanych im działań operacyjnych były efektywne, rozwiązanie powinno w pewien sposób minimalizować odległości między siedzibami przedstawicieli, a przypisanymi im regionami. Dane dotyczące odległości pomiędzy siedzibami swoich przedstawicieli a każdym z regionów są dane w Tabeli 1. Funkcja f_1 powinna zostać zdefiniowana jako suma odległości pomiędzy każdą z siedzib przedstawiciela firmy a przypisanymi im regionami.
- **Minimalizacja zmian f_2**: Firma Pfitzer chciała by uniknąć drastycznych zmian w przydziałach obszarów operacyjnych przedstawicieli handlowych. Modli oni np. nawiązać już dobre relacje z klientami i przeciwwskazanym było by burzyć te kontakty. Obecny przydział regionów do przedstawicieli jest dany w Tabeli 2. By zdefiniować miarę „zmian", model może zliczać dla ilu regionów nastąpiła zmiana przedstawiciela firmy (Uwaga: należy uwzględnić tylko te regiony, które nie były do tej pory przypisane do danego przedstawiciela a w nowym rozwiązaniu zostały przypisane). Ponadto składniki tej sumy mogły by być ważone „pracochłonnością" regionów. Jest to uzasadnione w ten sposób, że dla regionów słabo pracochłonnych ewentualna zmiana przydzielonego przedstawiciela nie jest tak istotna jak w sytuacji, w której region ten byłby silnie pracochłonny. Współczynniki pracochłonności regionów są dane w Tabeli 3. Ostatecznie więc f_2 jest zdefiniowana jako procentowy – po przemnożeniu przez 100% – odchyłek od obecnego przydziału.

Zakładając, że obecny przydział nie jest idealny, kryteria f_1 oraz f_2 są sobą do pewnego stopnia sprzeczne. W związku z tym, problem ma naturę wielokryterialną. Narzucone są tutaj trzy ograniczenia:

- Każdy region musi mieć przydzielonego dokładnie jednego przedstawiciela firmy.
- Suma pracochłonności regionów przydzielonych do każdego przedstawiciela powinna być w przedziale$[0.9,\ 1.1]$. Obecnie całkowita suma pracochłonności wszystkich regionów równa jest 4 więc dodatkowa normalizacja nie jest wymagana.
- Sugerowanym sposobem reprezentacji rozwiązania problemu jest macierz binarna, w której 1 oznacza, że dany pracownik (kolumna) ma przydzielony dany region (wiersz). Zmienne decyzyjne winny mieć więc dziedzinę $\{0, 1\}$.

Przedstawiony problem można rozwiązać wykorzystując Metodę Epsilon-Ograniczeń. Sugerowane jest optymalizowanie f_1 a ograniczanie f_2. Należy odnaleźć i wykreślić na wykresie 10 rozwiązań Pareto optymalnych znalezionych przy użyciu tej metody. Do implementacji Metody Epsilon-Ograniczeń należy wykorzystać jedno z dwóch narzędzi:

Program Excel

W programie Excel dostępny jest dodatek Solver, który pozwala rozwiązywać problemy opisane językiem programowania liniowego oraz całkowitoliczbowego. Dodatek powinien być domyślnie zainstalowany. Aby go aktywować, należy wejść w opcje programu i zakładkę „dodatki”. Po aktywacji odpowiednia ikona powinna się pojawić w zakładce „dane”.

Okno główne dodatku jest bardzo intuicyjne. Można w nim wskazać komórki w skoroszycie, które utożsamiane będą z zmiennymi decyzyjnymi. Podczas procesu optymalizacji, dodatek będzie zmieniał wartości w tych komórkach starając się odnaleźć optimum. Dodatkowo należy wskazać komórkę zawierającą funkcję celu oraz kierunek optymalizacji. Komórka ta winna zawierać liniową formułę. W

podobny sposób można dodać ograniczenia. W odpowiednim oknie dialogowym można wskazać komórkę zawierającą funkcje liniową oraz typ ograniczenia i stałą wartość na ograniczenia. W przypadku istnienia wielu ograniczeń, nie trzeba każdej komórki uwzględniać z osobna. Jeżeli funkcje ograniczające mają układ kolumnowy lub wierszowy, można po prostu zaznaczyć całą kolumnę/wiersz w oknie dialogowym.

W przypadku zadania domowego należy uwzględnić w dodatku Solver ograniczenia na dziedzinę zmiennych (zmienne binarne) oraz wykorzystać metodę LP Simplex. **Szablon rozwiązania znajduje się w pliku Pfitzer.xlsx**.

Język Python i biblioteka PuLP

Biblioteka PuLP jest darmową biblioteką do rozwiązywania problemów opisanych językiem programowania liniowego bądź całkowitoliczbowego.**W pliku Pfitzer.py znajduje się szablon programu**. Do wykreślenia uzyskanych punktów Pareto optymalnych można wykorzystać bibliotekę matplotlib. Przykładowy kod programu wykorzystującego bibliotekę PuLP znajduje się poniżej:

```
from pulp import *
import numpy as np


# Utworzenie instancji problemu
model = LpProblem(name="jakis-problem", sense=LpMaximize)


# Utworzenie dwoch zmiennych decyzyjnych
x1 = LpVariable(name="x1", lowBound=0, cat='Continuous')
x2 = LpVariable(name="x2", lowBound=0, cat='Continuous')
```

```
# Ograniczenia problemu
model += (1 * x1 + 1*x2 >= 1, "#1 constraint")
model += (2 * x1 + 1*x2 <= 6, "#2 constraint")
model += (-1 * x1 + 1*x2 == 1, "#3 constraint")

# Funkcja celu
obj_func = 4*x1 + 2 * x2
model += obj_func

# Uruchomienie solvera
status = model.solve()

# Wypisanie statusu
print(f"status: {model.status}, {LpStatus[model.status]}")
# WYNIK: status: 1, Optimal

# Wypisanie realizacji funkcji celu
print(f"objective: {model.objective.value()}")
# WYNIK: objective: 12.000000199999999

# Wypisanie wartosci zmiennych decyzyjnych
print(x1.value())
print(x2.value())
# WYNIK:
#   1.6666667
#   2.6666667
```

```
### Inny sposob -- macierzowy -- na utworzenie listy zmiennych
### oraz ograniczen

var_names = ["First", "Second"]
x = LpVariable.dicts("x", var_names, 0)
x
# WYNIK: {'First': x_First, 'Second': x_Second}

const_names = ["GE", "LE", "EQ"]
sense = [1, -1, 0] # GE, LE, EQ
coefs = [[1,1],[2,1],[-1,1]] # Matrix coefs
rhs = [1, 6, 1]

for c, s, r, cn in zip(coefs, sense, rhs, const_names):
    expr = lpSum([x[var_names[i]] * c[i] for i in range(2)])
    model += LpConstraint(e=expr, sense = s, name = cn, rhs = r)

obj_coefs = [4,2]
model += lpSum([x[var_names[i]] * obj_coefs[i] for i in range(2)])
```

Tabela 1: Odległości pomiędzy siedzibami przedstawicieli firmy a regionami (km)

	Siedziba			
Region	1	2	3	4
1	16.16	24.08	24.32	21.12
2	19.00	26.47	27.24	17.33
3	25.29	32.49	33.42	12.25
4	0.00	7.93	8.31	36.12
5	3.07	6.44	7.56	37.36
6	1.22	7.51	8.19	36.29
7	2.80	10.31	10.95	33.50
8	2.87	5.07	5.67	38.80
9	3.80	8.01	7.41	38.16
10	12.35	4.52	4.35	48.27
11	11.11	3.48	2.97	47.14
12	21.99	22.02	24.07	39.86
13	8.82	3.30	5.36	43.31
14	7.93	0.00	2.07	43.75
15	9.34	2.25	1.11	45.43
16	8.31	2.07	0.00	44.43
17	7.31	2.44	1.11	43.43
18	7.55	0.75	1.53	43.52
19	11.13	18.41	19.26	25.40
20	17.49	23.44	24.76	23.21

Tabela 2: Obecny przydział przedstawicieli firmy do regionów

Siedziba	**Region siedziby**	**Przydzielone regiony**
1	4	4, 5, 6, 7, 8, 15
2	14	10, 11, 12, 13, 14
3	16	9, 16, 17, 18
4	22	1, 2, 3, 19, 20, 21, 22

Tabela 3: Tabela zawierająca informacje o współczynnikach pracochłonności regionów

Region	Pracochłonność
1	0.1609
2	0.1164
3	0.1026
4	0.1516
5	0.0939
6	0.1320
7	0.0687
8	0.0930
9	0.2116
10	0.2529
11	0.0868
12	0.0828
13	0.0975
14	0.8177
15	0.4115
16	0.3795
17	0.0710
18	0.0427
19	0.1043
20	0.0997
21	0.1698
22	0.2531

Fundusze Europejskie Polska Cyfrowa
Rzeczpospolita Polska
Unia Europejska Europejski Fundusz Rozwoju Regionalnego

„Akademia Innowacyjnych Zastosowań Technologii Cyfrowych (AI Tech)", projekt finansowany ze środków Programu Operacyjnego Polska Cyfrowa POPC.03.02.00-00-0001/20