WYDZIAŁ
MATEMATYKI
I FIZYKI STOSOWANEJ
POLITECHNIKI RZESZOWSKIEJ

Patryk Motyka, Gabriel Lichacz

Problem pokrycia zbioru

Rzeszów, 2021

Spis treści

1. Opis problemu

Problem pokrycia zbioru to problem optymalizacyjny związany z doborem zasobów do wielu zadań. Często stosowany jest przy wyborze personelu. Jest to jeden z problemów NP-zupełnych Richarda Karpa, opisanych w jego pracy z 1972 roku o tytule "Reducibility Among Combinatorial Problems".

$$zbiór\ U\ \ m - elementowy$$

$$S_i \subseteq U, i = 1,2,\dots,n$$

$$a_{ij} = 1\ jeśli\ element\ j\ należy\ do\ podzbioru\ S_i$$

$$koszt{:}\ c_i$$

Należy wybrać J podzbiorów S_{i_j}, 1,2,…,J o minimalnym koszcie, takich że

$$\bigcup_{j=1}^{J} S_{i_j} = U$$

(mamy dostęp do wszystkich elementów zbioru U).

2. Model matematyczny

Zmienne decyzyjne:

$$x_i = \begin{cases} 1\ jeśli\ podzbiór\ i\ jest\ wybrany, \\ \quad 0\ jeśli\ nie. \end{cases}$$

Funkcja:

$$min \sum_{i=1}^{n} c_i n_i$$

Ograniczenia:

$$\bigvee j = 1,2,\dots,m, \qquad \bigvee i = 1,2,\dots,n, \qquad \sum_{i=1}^{n} a_{i_j} n_i \geq 1, \qquad n_i \in \{0,1\}$$

3. Treść zadania

Firma JanuszTECH planuje poszerzyć swoją działalność o nowe usługi. W związku z tym w zakładzie pojawi się 100 nowych zadań. Na rynku dostępne jest 80 maszyn, spełniających rygorystyczne wymagania przedsiębiorstwa. Znany jest ich koszt oraz zakres wykonywanych zadań. Znaleźć optymalny koszt inwestycji.

4. Generowanie danych

Wygenerowana tabela, wcześniej zapisana w formacie .csv, została wczytana do programu Excel. Zostały w nim dodane zadania wykonywane przez maszyny. Dane zostały następnie wyeksportowane do pliku dane_do_wczytania.txt, który we właściwej części kodu jest wczytywany.

```
########################_generowanie_danych_######################
maszyna_big <- c(1:80)
maszyna_big <- paste('x', as.character(maszyna_big), sep='')

 ceny_1 <- sample(15:25, 15, replace=T)
ceny_2 <- sample(35:45, 30, replace=T)
ceny_3 <- sample(55:65, 20, replace=T)
ceny_4 <- sample(75:85, 15, replace=T)

ceny_big <- c(ceny_1, ceny_2, ceny_3, ceny_4)
ceny_big

df <- data.frame(maszyna_big, ceny_big)
colnames(df) <- c('Maszyna', 'Cena')

jobs <-data.frame(matrix(0, 80, 100))
jobs_names <- c(1:100)
jobs_names <- paste('j', as.character(jobs_names), sep='')
colnames(jobs) <- jobs_names

df <- cbind(df, jobs)
write.csv(df, 'dane.csv', row.names = FALSE)
##################################################################
```

5. Algorytm zachłanny w języku R

Algorytm zachłanny polega na zliczeniu sumy zadań wykonywanych przez każdą maszynę oraz podzieleniu przez nią kosztu danej maszyny. Następnie należy wybrać maszynę dla której iloraz ten jest najmniejszy. Dalej usuwane są zadania, które wykonywała dana maszyna. Następnie usuwany jest wiersz odpowiadający maszynie o najmniejszym aktualnie ilorazie. Algorytm jest powtarzany dopóki nie zostaną usunięte wszystkie kolumny odpowiadające zadaniom.

```
#############################_biblioteki_##########################
if (!("here" %in% rownames(installed.packages()))) install.packages("here")
if (!("dplyr" %in% rownames(installed.packages()))) install.packages("dplyr")
library(here)
library(dplyr)

setwd(here()) # sciezka do pliku
getwd()
Sys.setenv(LANG = "en")
```

```
#######################_algorytm_#######################
vector_match <- list()  # stworzenie pustych list
ilorazy <- list()
koszt <- 0
nr <- 1

while (ncol(dane_3) > 4){  # petla wykonuje sie dopoki pozostaja niewykonane zadania
  colnames(dane_3) <- c(1:ncol(dane_3)) # zmiana nazw kolumn

  suma <- 0
  for (j in 1:nrow(dane_3)){  # przejscie po wszystkich wierszach
    for (i in 5 : ncol(dane_3)){  # przejscie po wszystkich kolumnach we wczesniej wybranych wierszu
      suma <- dane_3[j,i] + suma  # sumowanie jedynek w wierszu
    }
    dane_3[j,2] <- suma  # wpisanie policzonej sumy do kolumny "suma jednynek"
    dane_3[j,1] <- dane_3[j,4]/suma  # wpisanie ilorazu ceny przez sume jedynek do kolumny "iloraz"
    suma <- 0  # wyzerowanie sumy w celu przygotowania jej do nastepnego przejscia petli
  }

  match <- match(min(dane_3[,1]),dane_3[,1])  # wyszukanie pozycji najmniejszej wartosci z kolumny
"iloraz"
  print(paste0("Wybor nr ",nr,", Iloraz: ",dane_3[match,1]))  # wyswietlenie najmniejszej wartosci z kolumny
"iloraz"
  ilorazy[nr] <- dane_3[match,1]
  nr<-nr+1

  sum_usun <- 0  # suma usunietych zadan
  for (k in 5 : ncol(dane_3)){  # przejscie po kolumnach z zadaniami
    if (dane_3[match,k - sum_usun] == 1 && sum_usun != dane_3[match,2]) {  # jesli zadanie bylo
wykonywane przez wybrana maszyne z aktualnie
                                                    # najmiejszym ilroazem i suma usunietych nie jest rowna sumie
jedynek
                                                    # wybranej maszyny
      dane_3[,k - sum_usun] <- NULL  # usuniecie kolumn, ktorych zadania sa wykonane
      sum_usun <- sum_usun + 1

    }
  }

  koszt <- dane_3[match,4] + koszt  # sumuje koszt wybranych juz maszyn
  vector_match <- append(vector_match, dane_3[match,3])  # dodaje do listy nazwe wybranej maszyny

  dane_3 <- dane_3[-match,] # usuwa wiersz z wybrana maszyne z aktualnie najmiejszym ilroazem
  row.names(dane_3) <- c(1:nrow(dane_3))  # nadanie nazw wierszy
}
```

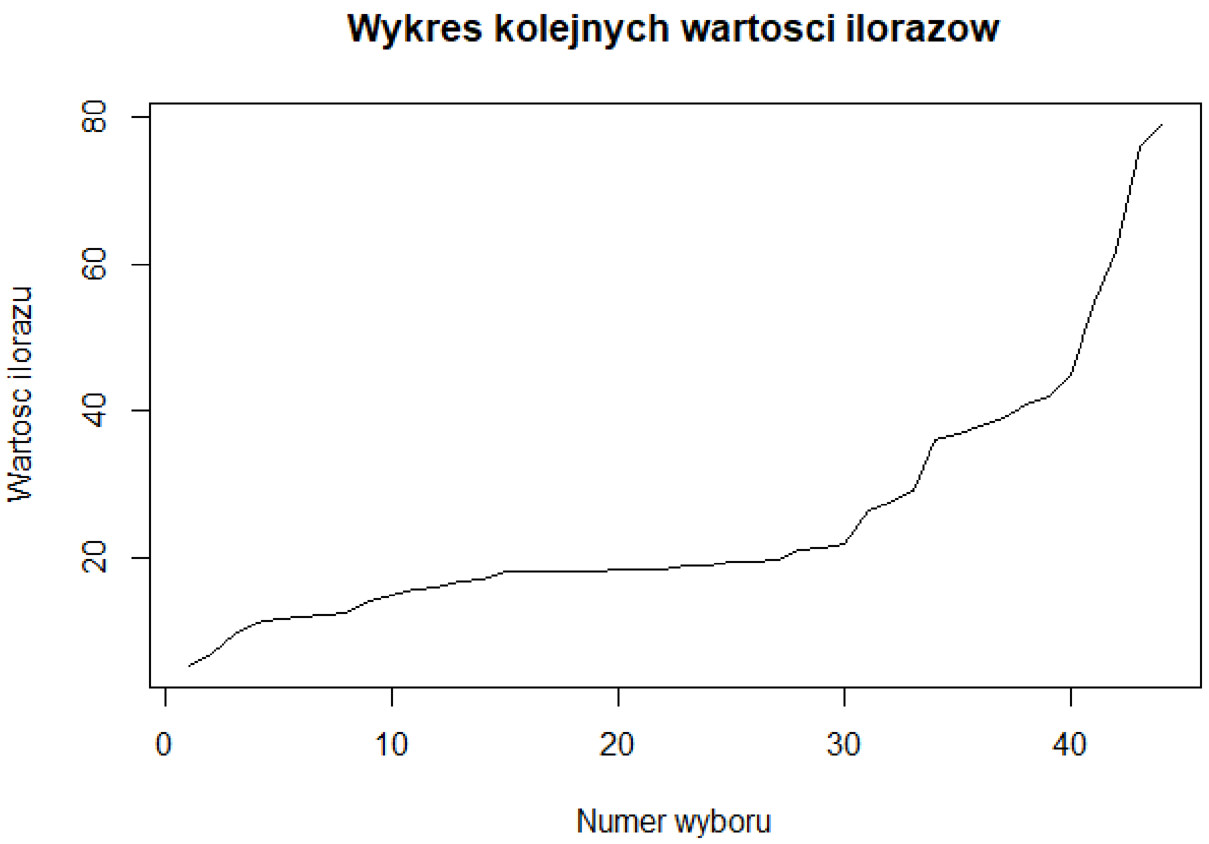
```
vector_match_2 <- setdiff(dane_2[,1], dane_3[,3])  # sprawdza ktore maszyny sa wybrane porownujac z oryginalna tabela

remove(i,j,k,match,sum_usun,suma, dane_2) # usuniecie niepotrzebnych wartosci

plot(1:length(ilorazy), ilorazy, type="l", main="Wykres kolejnych wartosci ilorazow", xlab="Numer wyboru", ylab="Wartosc ilorazu")  # wykres kolejnych wartosci ilorazow
```

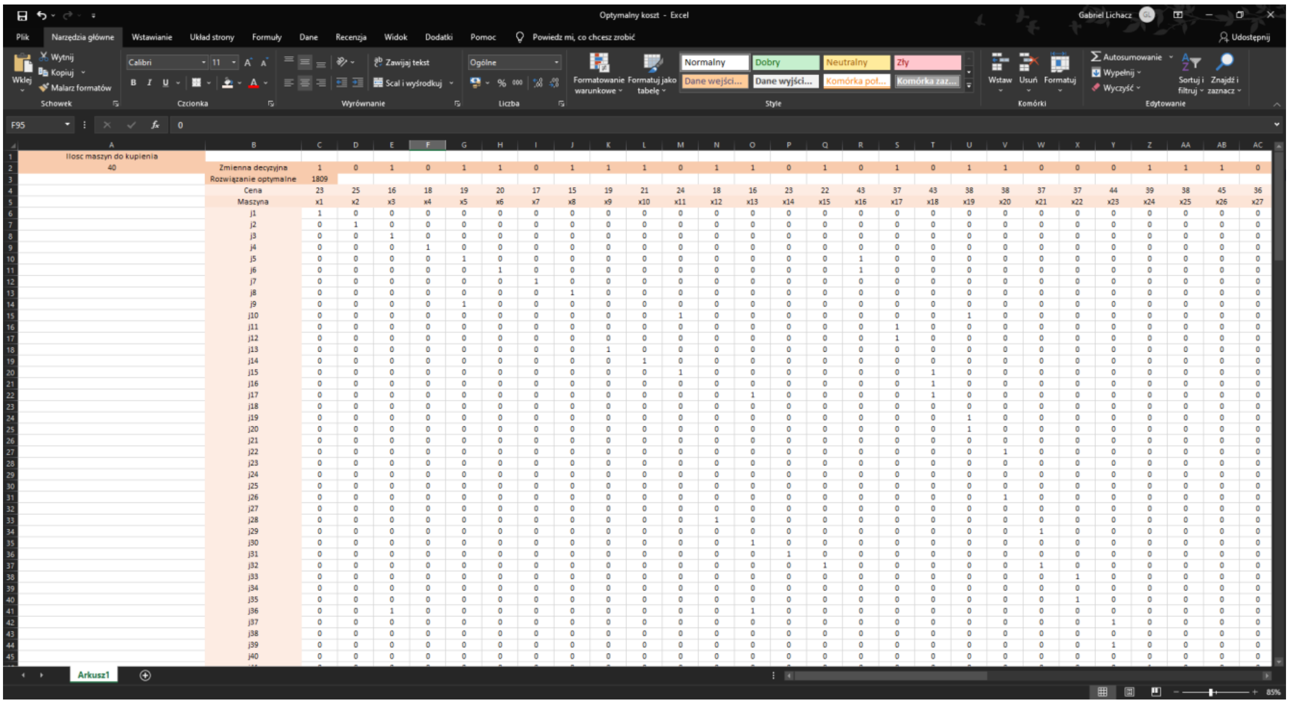
6. Rozwiązanie zachłanne

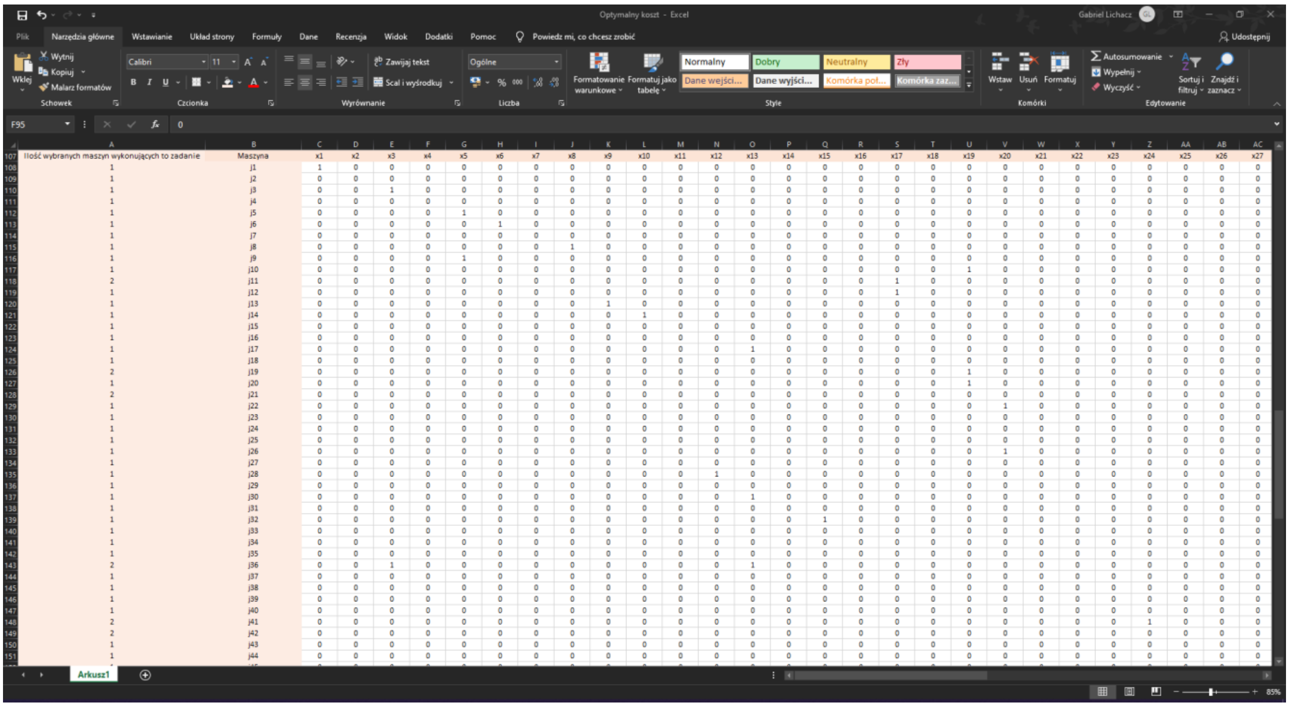
Algorytm zachłanny znalazł wynik wynoszący 1997 tys. zł.



7. Rozwiązanie optymalne w pakiecie Excel przy pomocy solvera

Macierz wyboru została wczytana z pliku .txt. Do istniejącej tabeli dodaliśmy opisy ułatwiające identyfikacje danych oraz niezbędne dla pakietu solver pola takie jak zmienne decyzyjne oraz pole celu („Rozwiązanie optymalne”). Wprowadziliśmy polecenie suma.iloczynow w polu celu, działające na polach zmiennych decyzyjnych oraz ceny, dzięki czemu możliwe było użycie solvera. Stworzyliśmy macierz o takich samych wymiarach, zawierającą iloczyny wartości zmiennej decyzyjnej danej maszyny i odpowiadającego pola z macierzy podstawowej. Następnie zsumowaliśmy wartości z wierszy jako kolumnę o nazwie „Ilość wybranych maszyn wykonujących to zadanie”. Posłużyła jako jedno z dwóch ograniczeń w solverze (wartości muszą być większe lub równe jeden). Drugim była binarność zmiennych decyzyjnych.





Solver zwrócił wynik 1809 tys. zł.

8. Wnioski

Algorytm zachłanny wyznacza nieznacznie gorsze rozwiązanie niż pakiet solver. Różnica wynosi 188 tys. zł.