МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Информационные технологии и прикладная математика»

Кафедра 804 «Теория вероятностей и компьютерное моделирование»

**Лабораторная работа №1**

**по курсу «Математическая экономика»**

| Выполнил: | М.А.Трофимов |
| --- | --- |
| Группа: | М8О-408Б-18 |
| Преподаватель: | С.В.Иванов |

Москва, 2021

Текст программы:

| library(Matrix)  library(gurobi)  library(ramify) # для argmax  # =========  # p two  # =========  l <- 8  k <- 21  n <- 10 + k%/%4  #n <- 2  m <- 30 - k%/%4  #m <- 3  cat("l =", l, '\n')  cat("k =", k, '\n')  cat("n =", n, '\n')  cat("m =", m, '\n')  tmp <- 0  b <- c()  for ( i in 1:m){  tmp = 70 + l + k + i  b = append(b, tmp)  }  cat("b :", b, "\n")  c <- c()  for ( i in 1:n){  tmp = 50 + l + k - i  c = append(c, tmp)  }  cat("c :", c, "\n")  A <- c()  for (i in 1:m)  for ( j in 1:n){  tmp = 1 + ( (j+k)\*i + j\*j + i\*i\*i + 3\*(i+l) ) %% (30 + k%/%5)  #tmp = (i-1)\*n + j  A = append(A, tmp)  }  A = matrix(A, ncol=n, byrow=T)  cat("A :\n")  print(A)  # =========  # p three  # =========  model=list()  model$A = A  model$obj = c  model$modelsense = 'max'  model$rhs = b  model$vtype = 'C'  result = gurobi(model)  print('objective value')  print(result$objval) # выведем оптимальное значение целевой функции  print('x=')  print(result$x) # выведем решение задачи  print('y=')  print(result$pi) # выведем решение двойственной задачи  value1 = result$objval  # =========  # p four  # =========  idx = argmax(matrix(result$pi), rows=F)  result$pi[idx]=0  idx2 = argmax(matrix(result$pi), rows=F)  cat("\n\nBest resource to increase is", idx, "th resource\n\n")  b[idx] = b[idx] + 1  model$rhs = b  b[idx] = b[idx] - 1  result = gurobi(model)  print('objective value')  print(result$objval) # выведем оптимальное значение целевой функции  print('x=')  print(result$x) # выведем решение задачи  print('y=')  print(result$pi) # выведем решение двойственной задачи  value2 = result$objval  # =========  # p five  # =========  b[idx2] = b[idx2] + 1  model$rhs = b  b[idx2] = b[idx2] - 1  cat("\n\nchange resource", idx2, "\n\n")  result = gurobi(model)  print('objective value')  print(result$objval) # выведем оптимальное значение целевой функции  print('x=')  print(result$x) # выведем решение задачи  print('y=')  print(result$pi) # выведем решение двойственной задачи  value3 = result$objval  cat("\n\nprevious value", value2, "better then", value3, "\n\n")  # =========  # p six  # =========  model$rhs = b  model$vtype = 'I'  result = gurobi(model)  print('objective value')  print(result$objval) # выведем оптимальное значение целевой функции  print('x=')  print(result$x) # выведем решение задачи  cat("\n\n")  value4 = result$objval  # =========  # p seven  # =========  vtypes = c()  for ( i in 1:(n%/%2) ){  vtypes = append(vtypes,'I')  }  for ( i in (n%/%2 + 1):n ){  vtypes = append(vtypes,'C')  }  model$vtype = vtypes  result = gurobi(model)  print('objective value')  print(result$objval) # выведем оптимальное значение целевой функции  print('x=')  print(result$x) # выведем решение задачи  value5 = result$objval  # =========  # p eight  # =========  cat("\n\nvalue \\w all integers", value4, "worse then", value5, "\n\n")  rm(list=ls()) |
| --- |

Результаты исчислений:

| schizophrenia@home:~/labs/4kurs/MathEc/1lab$ Rscript main.R  Загрузка требуемого пакета: slam  Присоединяю пакет: ‘ramify’  Следующие объекты скрыты от ‘package:Matrix’:  tril, triu  Следующий объект скрыт от ‘package:graphics’:  clip  l = 8  k = 21  n = 15  m = 25  b : 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124  c : 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64  A :  [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]  [1,] 18 22 28 2 12 24 4 20 4 24 12 2 28  [2,] 16 21 28 3 14 27 8 25 10 31 20 11 4  [3,] 26 32 6 16 28 8 24 8 28 16 6 32 26  [4,] 20 27 2 13 26 7 24 9 30 19 10 3 32  [5,] 4 12 22 34 14 30 14 34 22 12 4 32 28  [6,] 18 27 4 17 32 15 34 21 10 1 28 23 20  [7,] 34 10 22 2 18 2 22 10 34 26 20 16 14  [8,] 24 1 14 29 12 31 18 7 32 25 20 17 16  [9,] 28 6 20 2 20 6 28 18 10 4 34 32 32  [10,] 18 31 12 29 14 1 24 15 8 3 34 33 34  [11,] 34 14 30 14 34 22 12 4 32 28 26 26 28  [12,] 14 29 12 31 18 7 32 25 20 17 16 17 20  [13,] 32 14 32 18 6 30 22 16 12 10 10 12 16  [14,] 26 9 28 15 4 29 22 17 14 13 14 17 22  [15,] 2 20 6 28 18 10 4 34 32 32 34 4 10  [16,] 34 19 6 29 20 13 8 5 4 5 8 13 20  [17,] 26 12 34 24 16 10 6 4 4 6 10 16 24  [18,] 18 5 28 19 12 7 4 3 4 7 12 19 28  [19,] 16 4 28 20 14 10 8 8 10 14 20 28 4  [20,] 26 15 6 33 28 25 24 25 28 33 6 15 26  [21,] 20 10 2 30 26 24 24 26 30 2 10 20 32  [22,] 4 29 22 17 14 13 14 17 22 29 4 15 28  [23,] 18 10 4 34 32 32 34 4 10 18 28 6 20  [24,] 34 27 22 19 18 19 22 27 34 9 20 33 14  [25,] 24 18 14 12 12 14 18 24 32 8 20 34 16  [,14] [,15]  [1,] 22 18  [2,] 33 30  [3,] 22 20  [4,] 29 28  [5,] 26 26  [6,] 19 20  [7,] 14 16  [8,] 17 20  [9,] 34 4  [10,] 3 8  [11,] 32 4  [12,] 25 32  [13,] 22 30  [14,] 29 4  [15,] 18 28  [16,] 29 6  [17,] 34 12  [18,] 5 18  [19,] 16 30  [20,] 5 20  [21,] 12 28  [22,] 9 26  [23,] 2 20  [24,] 31 16  [25,] 34 20  Gurobi Optimizer version 9.5.1 build v9.5.1rc2 (linux64)  Thread count: 4 physical cores, 4 logical processors, using up to 4 threads  Optimize a model with 25 rows, 15 columns and 375 nonzeros  Model fingerprint: 0xf0d710f4  Coefficient statistics:  Matrix range [1e+00, 3e+01]  Objective range [6e+01, 8e+01]  Bounds range [0e+00, 0e+00]  RHS range [1e+02, 1e+02]  Presolve time: 0.00s  Presolved: 25 rows, 15 columns, 375 nonzeros  Iteration Objective Primal Inf. Dual Inf. Time  0 1.5987500e+32 1.290625e+32 1.598750e+02 0s  18 4.0298182e+02 0.000000e+00 0.000000e+00 0s  Solved in 18 iterations and 0.00 seconds (0.00 work units)  Optimal objective 4.029818182e+02  [1] "objective value"  [1] 402.9818  [1] "x="  [1] 0.581818182 1.490909091 0.218181818 1.009090909 0.009090909 1.218181818  [7] 0.154545455 0.000000000 0.045454545 0.263636364 0.418181818 0.000000000  [13] 0.000000000 0.000000000 0.000000000  [1] "y="  [1] 0.00000000 0.67727273 0.14204545 0.00000000 0.41931818 0.23863636  [7] 0.00000000 0.00000000 0.00000000 0.40000000 0.65795455 0.49659091  [13] 0.40000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000  [19] 0.00000000 0.23863636 0.00000000 0.00000000 0.00000000 0.05795455  [25] 0.00000000  Best resource to increase is 2 th resource  Gurobi Optimizer version 9.5.1 build v9.5.1rc2 (linux64)  Thread count: 4 physical cores, 4 logical processors, using up to 4 threads  Optimize a model with 25 rows, 15 columns and 375 nonzeros  Model fingerprint: 0x16fb0b73  Coefficient statistics:  Matrix range [1e+00, 3e+01]  Objective range [6e+01, 8e+01]  Bounds range [0e+00, 0e+00]  RHS range [1e+02, 1e+02]  Presolve time: 0.00s  Presolved: 25 rows, 15 columns, 375 nonzeros  Iteration Objective Primal Inf. Dual Inf. Time  0 1.5987500e+32 1.290625e+32 1.598750e+02 0s  19 4.0365226e+02 0.000000e+00 0.000000e+00 0s  Solved in 19 iterations and 0.00 seconds (0.00 work units)  Optimal objective 4.036522593e+02  [1] "objective value"  [1] 403.6523  [1] "x="  [1] 0.57147810 1.51363689 0.21252745 0.99653299 0.00000000 1.23500520  [7] 0.14642321 0.00000000 0.03669248 0.28209869 0.42505489 0.00000000  [13] 0.00000000 0.00000000 0.00000000  [1] "y="  [1] 0.0000000 0.6601179 0.1866405 0.0000000 0.4400786 0.2534381 0.0000000  [8] 0.0000000 0.0000000 0.4066798 0.6935167 0.4734774 0.4066798 0.0000000  [15] 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.2200393 0.0000000  [22] 0.0000000 0.0000000 0.0000000 0.0000000  change resource 11  Gurobi Optimizer version 9.5.1 build v9.5.1rc2 (linux64)  Thread count: 4 physical cores, 4 logical processors, using up to 4 threads  Optimize a model with 25 rows, 15 columns and 375 nonzeros  Model fingerprint: 0xf85e5677  Coefficient statistics:  Matrix range [1e+00, 3e+01]  Objective range [6e+01, 8e+01]  Bounds range [0e+00, 0e+00]  RHS range [1e+02, 1e+02]  Presolve time: 0.00s  Presolved: 25 rows, 15 columns, 375 nonzeros  Iteration Objective Primal Inf. Dual Inf. Time  0 1.5987500e+32 1.290625e+32 1.598750e+02 0s  16 4.0363977e+02 0.000000e+00 0.000000e+00 0s  Solved in 16 iterations and 0.00 seconds (0.00 work units)  Optimal objective 4.036397727e+02  [1] "objective value"  [1] 403.6398  [1] "x="  [1] 0.58241979 1.46510695 0.24919786 1.01099599 0.04040775 1.19037433  [7] 0.15384358 0.00000000 0.04395053 0.26373663 0.41758021 0.00000000  [13] 0.00000000 0.00000000 0.00000000  [1] "y="  [1] 0.00000000 0.67727273 0.18068182 0.00000000 0.41931818 0.20000000  [7] 0.00000000 0.00000000 0.00000000 0.40000000 0.65795455 0.49659091  [13] 0.40000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000  [19] 0.00000000 0.20000000 0.00000000 0.00000000 0.03863636 0.05795455  [25] 0.00000000  previous value 403.6523 better then 403.6398  Gurobi Optimizer version 9.5.1 build v9.5.1rc2 (linux64)  Thread count: 4 physical cores, 4 logical processors, using up to 4 threads  Optimize a model with 25 rows, 15 columns and 375 nonzeros  Model fingerprint: 0x1896e981  Variable types: 0 continuous, 15 integer (0 binary)  Coefficient statistics:  Matrix range [1e+00, 3e+01]  Objective range [6e+01, 8e+01]  Bounds range [0e+00, 0e+00]  RHS range [1e+02, 1e+02]  Found heuristic solution: objective 234.0000000  Presolve time: 0.00s  Presolved: 25 rows, 15 columns, 375 nonzeros  Variable types: 0 continuous, 15 integer (0 binary)  Root relaxation: objective 4.029818e+02, 22 iterations, 0.00 seconds (0.00 work units)  Nodes | Current Node | Objective Bounds | Work  Expl Unexpl | Obj Depth IntInf | Incumbent BestBd Gap | It/Node Time  0 0 402.98182 0 10 234.00000 402.98182 72.2% - 0s  H 0 0 306.0000000 402.98182 31.7% - 0s  H 0 0 369.0000000 402.98182 9.21% - 0s  0 0 386.10640 0 9 369.00000 386.10640 4.64% - 0s  0 0 383.80000 0 9 369.00000 383.80000 4.01% - 0s  0 0 383.80000 0 10 369.00000 383.80000 4.01% - 0s  0 0 377.62187 0 9 369.00000 377.62187 2.34% - 0s  0 0 377.62187 0 9 369.00000 377.62187 2.34% - 0s  0 0 377.62187 0 9 369.00000 377.62187 2.34% - 0s  0 0 377.24353 0 9 369.00000 377.24353 2.23% - 0s  0 0 373.27948 0 10 369.00000 373.27948 1.16% - 0s  0 0 371.82759 0 10 369.00000 371.82759 0.77% - 0s  0 0 371.82759 0 10 369.00000 371.82759 0.77% - 0s  0 0 371.82759 0 10 369.00000 371.82759 0.77% - 0s  Cutting planes:  Gomory: 3  MIR: 3  StrongCG: 1  Explored 1 nodes (48 simplex iterations) in 0.02 seconds (0.01 work units)  Thread count was 4 (of 4 available processors)  Solution count 3: 369 306 234  Optimal solution found (tolerance 1.00e-04)  Best objective 3.690000000000e+02, best bound 3.690000000000e+02, gap 0.0000%  [1] "objective value"  [1] 369  [1] "x="  [1] 0 1 1 1 0 1 0 0 0 0 1 0 0 0 0  Gurobi Optimizer version 9.5.1 build v9.5.1rc2 (linux64)  Thread count: 4 physical cores, 4 logical processors, using up to 4 threads  Optimize a model with 25 rows, 15 columns and 375 nonzeros  Model fingerprint: 0xdc208417  Variable types: 8 continuous, 7 integer (0 binary)  Coefficient statistics:  Matrix range [1e+00, 3e+01]  Objective range [6e+01, 8e+01]  Bounds range [0e+00, 0e+00]  RHS range [1e+02, 1e+02]  Found heuristic solution: objective 262.4000000  Presolve time: 0.00s  Presolved: 25 rows, 15 columns, 375 nonzeros  Variable types: 8 continuous, 7 integer (0 binary)  Root relaxation: objective 4.029818e+02, 21 iterations, 0.00 seconds (0.00 work units)  Nodes | Current Node | Objective Bounds | Work  Expl Unexpl | Obj Depth IntInf | Incumbent BestBd Gap | It/Node Time  0 0 402.98182 0 7 262.40000 402.98182 53.6% - 0s  H 0 0 352.8090909 402.98182 14.2% - 0s  H 0 0 388.2413793 402.98182 3.80% - 0s  Cutting planes:  Gomory: 1  Explored 1 nodes (21 simplex iterations) in 0.01 seconds (0.00 work units)  Thread count was 4 (of 4 available processors)  Solution count 2: 388.241 262.4  Optimal solution found (tolerance 1.00e-04)  Best objective 3.882413793103e+02, best bound 3.882413793103e+02, gap 0.0000%  [1] "objective value"  [1] 388.2414  [1] "x="  [1] 1.00000000 1.00000000 0.00000000 1.00000000 0.00000000 1.00000000  [7] 0.00000000 0.27203065 0.00000000 0.19540230 0.65900383 0.00000000  [13] 0.07662835 0.00000000 0.04022989  value \w all integers 369 worse then 388.2414 |
| --- |

Краткая сводка

При решении исходной задачи было получено значение целевой функции равному 402.9818.   
 Решение двойственной задачи имело наибольшее значение во втором элементе, что означает, что наиболее полезный ресурс, за счёт увеличения запаса которого увеличение целевой функции будет наибольшим. Увеличим на единицу сначала запас второго ресурса, а затем для сравнения 11ый ресурс, т.к. в двойственном решении 11 элемент второй по величине. Первое увеличение дало значение 403.6523, а второе - 403.6398. Как видно, разница не слишком большая между полученными значениями, но увеличение целевой функции было больше в первом увеличении, чем во втором.

Теперь будем искать целочисленные решения и решения, которые наполовину целые, а наполовину действительные. В первом случае итоговое решение - 369, во втором - 388.2414. Как видно, целочисленные решения достаточно сильно ухудшают целевую функцию.