Fakulta riadenia a informatiky

Informatika

Zadanie H11

# Duálna vkladacia heuristika

Mgr. Peter Czimmermann, Phd.   
STREDA 3, 4 Maroš Gorný, 5ZYI24

2021/2022

Obsah

[Duálna vkladacia heuristika 1](#_Toc72094833)

[Zadanie H11 3](#_Toc72094834)

[Popis riešeného algoritmu na konkrétnej úlohe 4](#_Toc72094835)

[Úvod 4](#_Toc72094836)

[Cieľ 4](#_Toc72094837)

[Postup riešenia 5](#_Toc72094838)

[Popis jednotlivých tried programu 6](#_Toc72094839)

[*Main()* 6](#_Toc72094840)

[Heuristika(String nazovSuboruHmotnost, String nazovSuboruCena) 7](#_Toc72094841)

[Záverečné zhodnotenie 7](#_Toc72094842)

[Účelová funkcia 7](#_Toc72094843)

[Konečný stav batohu 7](#_Toc72094844)

[Zhodnotenie 7](#_Toc72094845)

[Bonusová úloha – vylepšenie heuristiky 7](#_Toc72094846)

[Zadanie bonusovej úlohy 7](#_Toc72094847)

[Popis riešeného algoritmu na konkrétnej úlohe 7](#_Toc72094848)

[Úvod 7](#_Toc72094849)

[Cieľ 8](#_Toc72094850)

[Postup riešenia 8](#_Toc72094851)

[Popis jednotlivých tried programu, vylepšeného 10](#_Toc72094852)

[*Main()* 10](#_Toc72094853)

[DualnaHeuristika(String nazovSuboruHmotnost, String nazovSuboruCena) 10](#_Toc72094854)

[Záverečné zhodnotenie 16](#_Toc72094855)

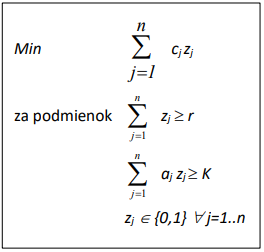
[Účelová funkcia 16](#_Toc72094856)

[Konečný stav batohu 16](#_Toc72094857)

[Zhodnotenie 16](#_Toc72094858)

## Zadanie H11

Duálnou vsúvacou heuristikou riešite úlohu danú modelom (obrátená úloha o batohu, kde hmotnosť batohu musí byť **aspoň K** a počet predmetov v batohu **aspoň r**).

  
Riešte úlohu pre **n=500**, **r=200, K=14000** a pre **lokálne kritérium** „Vlož prvok z dosiaľ nespracovaných prvkov, ktorý **má najväčšiu hmotnosť**“. Východiskové riešenie položte rovné prázdnemu batohu (neprípustné riešenie).

Súčasťou zadania sú súbory H1\_a.txt a H1\_c.txt, ktoré obsahujú n údajov koeficientov aj a cj pre j=1..n potrebných pre riešenie zadanej úlohy.

## Popis riešeného algoritmu na konkrétnej úlohe

### Úvod

Na začiatok je potrebné upresniť úlohu a určiť cieľ aby som presne vedel, že čo mám robiť, akým postupom a za akých podmienok sa ku tomu cieľu dostanem.

### Cieľ

Cieľ pri zadanej úlohe je účelová funkcia

Kde minimalizujeme cenu v batohu

#### za podmienok

Počet prvkov v batohu musí byť väčší alebo rovný ako r.

Hmotnosť všetkých prvkov v batohu musí byť väčšia alebo rovná ako K.

Zj môže nadobúdať len hodnoty 0 a 1, pre všetky prvky danej úlohy.  
0 znamená, že prvok v batohu nie je, naopak 1 znamená, že prvok v batohu je.

#### lokálne kritérium

*„Vlož prvok z dosiaľ nespracovaných prvkov, ktorý má najväčšiu hmotnosť.“*  
To znamená, že do batohu budem pridávať prvok, ktorý ešte nie je v batohu a budem sa riadiť podmienkou, že vyberám prvok ktorý má najväčšiu hmotnosť.

#### Východiskové riešenie

*„Východiskové riešenie položte rovné prázdnemu batohu (neprípustné riešenie).“*

To znamená, že úlohu začnem riešiť s prázdnym batohom, teda v batohu nebude žiadny prvok. Čo síce nie je prípustné riešenie, ale zadaná úloha ma takýto postup.

#### Vysvetlivky

***n*** *je počet všetkých predmetov (zatiaľ nespracovaných),*

***r*** *je minimálny počet predmetov v batohu,****K*** *je minimálna hmotnosť akú musí mať batoh,*

***cj*** *je cena j-teho predmetu,****aj*** *je hmotnosť j-teho predmetu,*

***z****j je j-ty predmet.*

### Postup riešenia

1. Určím si kritérium
   1. Prvok ktorý vložím do batohu musí mať maximálnu hmotnosť z doposiaľ nevložených prvkov
2. Určím si počet všetkých prvkov
   1. Počet všetkých prvkov je 500.
3. Všetky prvky vyložím z batohu.
4. Určím si podmienky
   1. Prvok môže byť len vložený alebo nevložený v batohu (1/0).
   2. Počet prvkov v batohu musí byť aspoň 200.
   3. Hmotnosť batohu musí byť aspoň 14 000.
5. Vložím doposiaľ nevložený prvok do batohu, podľa kritéria a za daných podmienok
   1. Ak sú splnené všetky podmienky, skočím na krok 6
   2. Ak nie sú splnené všetky podmienky, opakujem krok 5
6. Končím s algoritmom a účelová funkcia sa rovná sume ceny všetkých prvkov v batohu.

## Popis jednotlivých tried programu

V programe sa nachádzajú dve triedy, trieda ***Main*** a trieda ***Heuristika***

### *Main()*

Trieda ***Main()*** slúži na spustenie programu, vytváram v nej objekt ***Heuristika*** s názvom *„heu“*, ktorý si vypýta dva textové súbory: prvý je textový súbor, v ktorom je vložená hmotnosť prvkov a druhý je textový súbor, v ktorom je vložená cena prvkov.  
Následne len vyvolávam metódy triedy Heuristika.

#### Zdrojový kód triedy Main

public class Main **{**

public static void main**(**String**[]** args**)** **throws** FileNotFoundException **{**

heu **=** **new** Heuristika**(**"H1\_a.txt"**,**"H1\_c.txt"**);**

heu**.**dualnaVsuvacaHeurestika**();**

heu**.**vypisVysledkov**();**

heu**.**zapisVysledokDoSuboru**(**"Výstup pred vylepšením.txt"**);**

**}**

**}**

### Heuristika(String nazovSuboruHmotnost, String nazovSuboruCena)

Trieda ***Heuristika(String nazovSuboruHmotnost, String nazovSuboruCena)*** slúži na vykonávanie všetkých procesov ohľadom algoritmu pre danú heuristiku, taktiež na výpis na konzolu ale tiež na zápis a čítanie z alebo do textového súboru.

#### Premenné

private int**[]** a**;**

private int**[]** c**;**

private boolean**[]** z**;**

private int m**;**

private int aktualnaHmotnost**;**

private int aktualnaCena**;**

private int aktualnyPocetPrvkov**;**

private static final int r **=** 200**;**

private static final int K **=** 14000**;**

* private int[] a – Pole hmotnosti prvkov
* private int[] c – Pole ceny prvkov
* private boolean[] z – Pole, ci sa prvok v batohu nachádza*(TRUE)* alebo nenachádza*(FALSE)*
* private int m – Počet všetkých prvkov
* private int aktualnaHmotnost – Aktuálna hmotnosť batohu
* private int aktualnaCena – Aktuálna cena batohu
* private int aktualnyPocetPrvkov – Aktuálny počet prvkov v batohu
* private static final int r = 200 - Minimálny počet prvkov v batohu
* private static final int K = 14000 – Minimálna hmotnosť batohu

#### Konštruktor

public Heuristika**(**String nazovSuboruHmotnost**,** String nazovSuboruCena**)** **throws** FileNotFoundException **{**

aktualnaCena **=** 0**;**

aktualnaHmotnost **=** 0**;**

aktualnyPocetPrvkov **=** 0**;**

m **=** getPocetRiadkov**(**nazovSuboruHmotnost**);**

a **=** **new** int**[**m**];**

c **=** **new** int**[**m**];**

z **=** **new** boolean**[**m**];**

nacitajZoSuboruHmotnost**(**nazovSuboruHmotnost**);**

nacitajZoSuboruCenu**(**nazovSuboruCena**);**

**}**

* AktualnaCena, AktualnaHmotnost, AktualnyPocet prvkov nastavím na 0
* m nastavím na počet riadkov zo súboru
* Polia *a, c* vytvorím a nastavím na nuly
* Pole *z* vytvorím a nastavím na FALSE
* Načítam hodnoty zo súboru s cenami aj s hmotnosťou

#### Metódy

public void vypisVysledkov**()**

public void dualnaVsuvacaHeurestika**()**

public void vlozPrvokDoBatohuSMaxHmotnostou**(**int indexPrvku**)**

public int getIndexMaximaHmotnosti**()**

public int getPocetRiadkov**(**String nazovSuboru**)**

public void nacitajZoSuboruHmotnost**(**String nazovSuboru**)**

public void nacitajZoSuboruCenu**(**String nazovSuboru**)**

public void zapisVysledokDoSuboru**(**String nazovSuboru**)**

##### public void vypisVysledkov()

* Vypíše konečnú cenu, hmotnosť a počet prvkov v batohu

public void vypisVysledkov**()** **{**

System**.**out**.**println**(**"Aktuálna cena batohu:\t" **+** aktualnaCena**);**

System**.**out**.**println**(**"Aktuálna hmotnosť:\t\t" **+** aktualnaHmotnost**);**

System**.**out**.**println**(**"Aktuálny počet prvkov: \t" **+** aktualnyPocetPrvkov**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Hodnota účelovej funkcie:\t" **+** aktualnaCena**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Index vybraného predmetu... (1 až " **+** m **+** ")"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **((**i **+** 1**)** **%** 100 **==** 0**)** **{**

System**.**out**.**println**();**

**}**

**if** **(**z**[**i**]** **==** **true)** **{**

System**.**out**.**print**(**i **+** 1 **+** ", "**);**

**}**

**}**

**}**

##### public void dualnaVsuvacaHeurestika()

* Pokúsi sa vložiť 500 krát prvok s najväčšou hmotnosťou do batohu až pokiaľ nebude batoh spĺňať všetky podmienky

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**aktualnyPocetPrvkov **>=** r **&&** aktualnaHmotnost **>=** K**)** **{**

**break;**

**}** **else** **{**

vlozPrvokDoBatohuSMaxHmotnostou**(**getIndexMaximaHmotnosti**());**

**}**

**}**

**}**

##### public void vlozPrvokDoBatohuSMaxHmotnostou(int indexPrvku)

* Vloží do batohu doposiaľ nevložený prvok s najväčšou hmotnosťou a pripočíta cenu, hmotnosť a daný prvok do batohu

public void vlozPrvokDoBatohuSMaxHmotnostou**(**int indexPrvku**)** **{**

z**[**indexPrvku**]** **=** **true;**

aktualnaHmotnost **+=** a**[**indexPrvku**];**

aktualnaCena **+=** c**[**indexPrvku**];**

aktualnyPocetPrvkov **++;**

**}**

##### public int getIndexMaximaHmotnosti()

* Vráti index prvku ktorý ešte nebol vložený do batohu a má najväčšiu hmotnosť

public int getIndexMaximaHmotnosti**(){**

int docasneMaximum **=** **-**1**;**

int indexMaxima **=** **-**1**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**a**[**i**]** **>** docasneMaximum **&&** z**[**i**]** **!=** **true)** **{**

docasneMaximum **=** a**[**i**];**

indexMaxima **=** i**;**

**}**

**}**

**return** indexMaxima**;**

**}**

##### public int getPocetRiadkov(String nazovSuboru)

* Vráti počet riadkov z daného súboru

public int getPocetRiadkov**(**String nazovSuboru**){**

Path cesta **=** Paths**.**get**(**nazovSuboru**);**

int riadky **=** 0**;**

**try** **{**

riadky **=** **(**int**)**Files**.**lines**(**cesta**).**count**();**

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**return** riadky**;**

**}**

##### public void nacitajZoSuboruHmotnost(String nazovSuboru)

* Načíta hodnoty z textového súboru do poľa s hmotnosťou

public void nacitajZoSuboruHmotnost**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException **{**

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i**=**0**;** i**<**m**;** i**++)** **{**

a**[**i**]=**citac**.**nextInt**();**

**}**

citac**.**close**();**

**}**

##### public void nacitajZoSuboruCenu(String nazovSuboru)

* Načíta hodnoty z textového súboru do poľa s hmotnosťou

public void nacitajZoSuboruCenu**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException **{**

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i**=**0**;** i**<**m**;** i**++)** **{**

c**[**i**]=**citac**.**nextInt**();**

**}**

citac**.**close**();**

**}**

##### public void zapisVysledokDoSuboru(String nazovSuboru)

* Zapíše výsledok do textového súboru. V zápise je počet predmetov, hmotnosť a cena batohu spolu s účelovou funkciou a index prvkov ktoré sa nachádzajú v batohu

public void zapisVysledokDoSuboru**(**String nazovSuboru**)** **{**

**try** **{**

FileWriter zapis **=** **new** FileWriter**(**nazovSuboru**);**

zapis**.**write**(**"Počet predmetov = " **+** aktualnyPocetPrvkov **+** "\nHmotnosť batohu = " **+** aktualnaHmotnost **+** "\nÚčelová funkcia = " **+** aktualnaCena **+** "\n\n"**);**

zapis**.**write**(**"Index prvku\tHmotnosť\tCena\n"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**z**[**i**]** **==** **true)** **{**

zapis**.**write**(**String**.**valueOf**(**i **+** 1**)** **+** "\t\t" **+** a**[**i**]** **+** "\t\t" **+** c**[**i**]** **+** "\n"**);**

**}**

**}**

zapis**.**close**();**

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**}**

#### Zdrojový kód triedy Heuristika, aj s komentármi

package com**.**company**;**

**import** java**.**io**.**File**;**

**import** java**.**io**.**FileNotFoundException**;**

**import** java**.**io**.**FileWriter**;**

**import** java**.**io**.**IOException**;**

**import** java**.**nio**.**file**.**Files**;**

**import** java**.**nio**.**file**.**Path**;**

**import** java**.**nio**.**file**.**Paths**;**

public class Heuristika **{**

private int **[]** a**;** //hmotnost

private int **[]** c**;** //cena

private boolean **[]** z**;** // vybrany/nevybrany prvok

private int m**;** // pocet riadkov v textovom subore

private int aktualnaHmotnost**;**

private int aktualnaCena**;**

private int aktualnyPocetPrvkov**;**

private static final int r**=**200**;** //minimalny pocet prvkov

private static final int K**=**14000**;** //minimalna kapacita

public Heuristika**(**String nazovSuboruHmotnost**,** String nazovSuboruCena**)** **throws** FileNotFoundException **{**

aktualnaCena **=** 0**;**

aktualnaHmotnost **=** 0**;**

aktualnyPocetPrvkov **=** 0**;**

m **=** getPocetRiadkov**(**nazovSuboruHmotnost**);** //Ziskam pocet riadkov z textoveho suboru

a **=** **new** int**[**m**];** //inicializacia s nulami

c **=** **new** int**[**m**];** //inicializacia s nulami

z **=** **new** boolean**[**m**];** //batoh je naplneny s false

nacitajZoSuboruHmotnost**(**nazovSuboruHmotnost**);** // do a[] priradi dane hodnoty z .txt

nacitajZoSuboruCenu**(**nazovSuboruCena**);** // do c[] priradi dane hodnoty z .txt

**}**

public void vypisVysledkov**()** **{**

System**.**out**.**println**(**"Aktuálna cena batohu:\t" **+** aktualnaCena**);**

System**.**out**.**println**(**"Aktuálna hmotnosť:\t\t" **+** aktualnaHmotnost**);**

System**.**out**.**println**(**"Aktuálny počet prvkov: \t" **+** aktualnyPocetPrvkov**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Hodnota účelovej funkcie:\t" **+** aktualnaCena**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Index vybraného predmetu... (1 až " **+** m **+** ")"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **((**i **+** 1**)** **%** 100 **==** 0**)** **{** //po stovkach spravi new line

System**.**out**.**println**();**

**}**

**if** **(**z**[**i**]** **==** **true)** **{**

System**.**out**.**print**(**i **+** 1 **+** ", "**);** //vypise index predmetu ktory je v batohu

**}**

**}**

**}**

public void dualnaVsuvacaHeurestika**()** **{**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**aktualnyPocetPrvkov **>=** r **&&** aktualnaHmotnost **>=** K**)** **{** //Ak splnam dane podmienky zadania, tak koncim

**break;**

**}** **else** **{** //ak podmienky nesplnam, pokracujem dalej

vlozPrvokDoBatohuSMaxHmotnostou**(**getIndexMaximaHmotnosti**());**//vlozim do batohu prvok s najvacsou hmotnostou

**}**

**}**

**}**

public void vlozPrvokDoBatohuSMaxHmotnostou**(**int indexPrvku**)** **{**

z**[**indexPrvku**]** **=** **true;** //oznacim dany prvok, ze uz som ho vlozil do batohu

aktualnaHmotnost **+=** a**[**indexPrvku**];** //pripocitam hmotnost prvku

aktualnaCena **+=** c**[**indexPrvku**];** //pripocitam cenu prvku

aktualnyPocetPrvkov **++;** //pripocitam jednu jednotku k aktualnemu poctvu prvkom

**}**

public int getIndexMaximaHmotnosti**(){**

int docasneMaximum **=** **-**1**;**

int indexMaxima **=** **-**1**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**a**[**i**]** **>** docasneMaximum **&&** z**[**i**]** **!=** **true)** **{** //ak dany prvok este nie je v batohu a je zatial najvacsi

docasneMaximum **=** a**[**i**];** //zvolim ho ako maximalny prvok

indexMaxima **=** i**;** //priradim index maximalneho prvku

**}**

**}**

**return** indexMaxima**;** //vratim index maximalneho prvko

**}**

public int getPocetRiadkov**(**String nazovSuboru**){**

Path cesta **=** Paths**.**get**(**nazovSuboru**);**

int riadky **=** 0**;**

**try** **{**

riadky **=** **(**int**)**Files**.**lines**(**cesta**).**count**();** //pretypovanie na INT, vypocita kolko tam je riadkov

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**return** riadky**;**

**}**

public void nacitajZoSuboruHmotnost**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException

**{**

// vytvorenie novej instancie triedy Scanner pre citanie z textoveho suboru

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i**=**0**;** i**<**m**;** i**++)** **{** //pre vsetky riadky

a**[**i**]=**citac**.**nextInt**();** // nacitanie hodnoty prvku zo suboru

**}**

citac**.**close**();**

**}**

public void nacitajZoSuboruCenu**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException

**{**

// vytvorenie novej instancie triedy Scanner pre citanie z textoveho suboru

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i**=**0**;** i**<**m**;** i**++)** **{** //pre vsetky riadky

c**[**i**]=**citac**.**nextInt**();** // nacitanie hodnoty prvku zo suboru

**}**

citac**.**close**();**

**}**

public void zapisVysledokDoSuboru**(**String nazovSuboru**)** **{**

**try** **{**

FileWriter zapis **=** **new** FileWriter**(**nazovSuboru**);**

zapis**.**write**(**"Počet predmetov = " **+** aktualnyPocetPrvkov **+** "\nHmotnosť batohu = " **+** aktualnaHmotnost **+** "\nÚčelová funkcia = " **+** aktualnaCena **+** "\n\n"**);**

zapis**.**write**(**"Index prvku\tHmotnosť\tCena\n"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**z**[**i**]** **==** **true)** **{**

zapis**.**write**(**String**.**valueOf**(**i **+** 1**)** **+** "\t\t" **+** a**[**i**]** **+** "\t\t" **+** c**[**i**]** **+** "\n"**);**

**}**

**}**

zapis**.**close**();**

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**}**

**}**

## Záverečné zhodnotenie

### Účelová funkcia

= 21060

### Konečný stav batohu

Aktuálna cena batohu: 21060

Aktuálna hmotnosť: 14831

Aktuálny počet prvkov: 200

### Zhodnotenie

Daná heuristika nám zabezpečila cenu batohu rovnej 21060, pričom prvkov bolo 200 a hmotnosť batohu bola 14831.

Výsledok je prípustný, ale určite nie je najoptimálnejší. Vzhľadom na to, že náš cieľ bol minimalizovať cenu za daných podmienok, tento algoritmus raz môže byť lepší a pri iných prvkoch môže byť naopak horší.  
  
Pokiaľ by napríklad platilo to, že čím väčšia hmotnosť, tým väčšia cena, mohlo by sa nám stať, že by sme mali splnené všetky podmienky okrem podmienky, že v batohu musí byť minimálne 200 prvkov. Tým pádom by cena aj hmotnosť narastala, no čakalo by sa na to, aby bolo v batohu aspoň 200 prvkov.  
  
Ak by to ale bolo naopak, teda, že čím väčšia hmotnosť , tým menšia cena, tak by bol tento algoritmus rozhodne lepší. Pretože pri daných podmienkach by sa batoh naplnil s čo najťažšími prvkami, tým pádom by podmienka pre hmotnosť bola určite splnená a hneď ako by batoh obsahoval 200 prvkov, algoritmus by sa ukončil a cena batohu by teda mala byť menšia ako pri predošlom príklade.

# Bonusová úloha – vylepšenie heuristiky

## Zadanie bonusovej úlohy

Navrhnite a implementujte vlastný heuristický algoritmus na zlepšenie výsledkov zadanej heuristiky. Algoritmus a dosiahnuté výsledky popíšte v dokumentácii.

## Popis riešeného algoritmu na konkrétnej úlohe

### Úvod

Vylepšený algoritmus som sa rozhodol robiť na pôvodnej úlohe. Rozhodol som sa teda konečný výsledok zadania **H11** vylepšiť, bez toho, aby som menil pôvodne zadanie a pridal algoritmus, ktorý mi vymení prvky z batohu tak, aby bola účelová funkcia menšia. Jediná zmena v pôvodnom algoritme je taká, že pri výbere maxima z doposiaľ nevybraných prvkov sa pozerám aj na to, či tam nie je prvok, ktorý ma takú istú hmotnosť, ale zato má menšiu cenu[[1]](#footnote-1).

### Cieľ

Cieľ je teda skúsiť vymeniť každý prvok, ktorý je v batohu s prvkom ktorý v batohu nie je, tak aby boli stále splnene podmienky a aby cena nového prvku bola menšia ako cena starého prvku.

[Účelová funkcia](#_Cieľ), [podmienky](#_za_podmienok), [lokálne kritérium](#_lokálne_kritérium) a [východiskové riešenie](#_Východiskové_riešenie) sú rovnaké ako v pôvodnej úlohe.   
  
Avšak keď dosiahnem optimálny výsledok, spúšťam nový algoritmus, ktorý mi začne vymieňať prvky tak, aby sa pokúsil účelovú funkciu zmenšiť.

### Postup riešenia

Začiatok postupu riešenia je taký istý ako v [pôvodnej úlohe](#_Postup_riešenia).

Preto postup bonusovej úlohy začína krokom 7 kde sa napája nový algoritmus na ten starý.

1. Vyberiem si n-ty prvok z batohu n=(1..500)
2. Nájdem ďalší prvok, ktorý nie je v batohu
   1. Ak taký prvok nenájdem skočím na krok 11
3. Skúsim ho vymeniť
   1. Ak sú aj po výmene splnené všetky podmienky a nová cena je menšia ako pôvodná cena, tak skočím na krok 10
   2. Ak by po výmene neboli splnené podmienky, alebo by nová cena bola väčšia alebo rovná ako pôvodná cena, tak skočím na krok 8
4. Zapamätám si vybraný prvok a skočím na krok 8
5. Zoberiem si posledný prvok, ktorý som si zapamätal
   1. Ak som si nezapamätal žiadny prvok, inkrementujem n-ko a skočím na krok 7
   2. Ak som si zapamätal nejaký prvok, skočím na krok 12
6. Vyhodím n-ty prvok z batohu a posledný prvok ktorý som si zapamätal vložím do batohu.
7. Aktualizujem cenu a hmotnosť batohu, vymažem si posledný zapamätaný prvok z batohu a inkrementujem n-ko.
8. Ak n = 500, algoritmus sa končí, inak skočím na krok 7.

## Popis jednotlivých tried programu, vylepšeného

V programe sa nachádzajú dve triedy, ***Main*** a  ***DualnaHeuristika***

### *Main()*

Trieda **Main()** slúži na spustenie programu, vytváram v nej objekt **DualnaHeuristika** s názvom *„heu“*, ktorý si vypýta dva textové súbory: prvý je textový súbor, v ktorom je vložená hmotnosť prvkov a druhý je textový súbor, v ktorom je vložená cena prvkov.  
Následne len vyvolávam metódy triedy DualnaHeuristika.

Najprv sa spustí pôvodná úloha, vypíšu sa jej výsledky na konzolu a taktiež sa dané výsledky zapíšu do .txt súboru a následne sa spustí algoritmus ktorý danú úlohu optimalizuje a taktiež sa vypíšu výsledky na konzolu a aj do .txt súboru.

#### Zdrojový kód triedy Main

public class Main **{**

public static void main**(**String**[]** args**)** **throws** FileNotFoundException **{**

DualnaHeuristika heu;

heu = new DualnaHeuristika("H1\_a.txt","H1\_c.txt");

heu.dualnaVsuvacaHeurestika();

heu.vypisVysledkov();

heu.zapisVysledokDoSuboru("Výstup pred vylepšením.txt");

System.out.println();

System.out.println("--------------------------------------------");

heu.vylepsenieHeurestiky();

heu.vypisVysledkov();

heu.zapisVysledokDoSuboru("Výstup po vylepšení.txt");

System.out.println();

**}**

**}**

### DualnaHeuristika(String nazovSuboruHmotnost, String nazovSuboruCena)

Trieda ***DualnaHeuristika*** je rovnaká ako trieda ***Heuristika*** pri pôvodnej úlohe. Zmena sa spravila len v metóde ***getIndexMaximaHmotnosti()*** a pridala sa metóda ***vylepsenieHeurestiky().***

#### [Premenné](#_Premenné) (rovnaké ako v pôvodnej úlohe)

#### Metódy - Zmenené

public int getIndexMaximaHmotnosti**()**

##### public int getIndexMaximaHmotnosti()

* Vráti index prvku ktorý ešte nebol vložený do batohu a má najväčšiu hmotnosť, kontroluje sa však aj to, či neexistuje aj ďalší prvok, ktorý ma rovnakú hmotnosť, ale má nižšiu cenu

public int getIndexMaximaHmotnosti**()** **{**

int docasneMaximum **=** **-**1**;**

int indexMaxima **=** **-**1**;**

int cenaMaxima **=** **-**1**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**a**[**i**]** **>=** docasneMaximum **&&** z**[**i**]** **!=** **true)** **{**

**if** **(**a**[**i**]** **==** docasneMaximum **&&** cenaMaxima **<** c**[**i**])** **{**

**continue;**

**}** **else** **{**

docasneMaximum **=** a**[**i**];**

indexMaxima **=** i**;**

cenaMaxima **=** c**[**i**];**

**}**

**}**

**}**

**return** indexMaxima**;**

**}**

#### Metódy - Pridané

public void vylepsenieHeurestiky**()** **{**

##### public int vylepsenieHeurestiky()

* Metóda sa snaží vymeniť každý vložený prvok, s každým prvkom ktorý nie je v batohu, za podmienky, že sa zmenší cena batohu a že zostanú splnené aj základne podmienky

public void vylepsenieHeurestiky**()** **{**

int indexNahradnehoPrvku **=** **-**1**;**

int novaCena **=** aktualnaCena**;**

int novaHmotnost **=** aktualnaHmotnost**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**z**[**i**]** **==** **true)** **{**

**for** **(**int j **=** 0**;** j **<** m**;** j**++)** **{**

**if** **(**j **==** i **||** **(**z**[**j**]** **==** **true))** **continue;**

int hmotnostPoVymene **=** aktualnaHmotnost **-** a**[**i**]** **+** a**[**j**];**

int cenaPoVymene **=** aktualnaCena **-** c**[**i**]** **+** c**[**j**];**

**if** **(**hmotnostPoVymene **>=** K **&&** cenaPoVymene **<** aktualnaCena**)** **{**

**if** **(**cenaPoVymene **>** novaCena**)** **continue;**

indexNahradnehoPrvku **=** j**;**

novaCena **=** cenaPoVymene**;** novuCenu kandidata

novaHmotnost **=** hmotnostPoVymene**;**

**}**

**}**

**if** **(**indexNahradnehoPrvku **!=** **-**1**)** **{**

z**[**i**]** **=** **false;**

z**[**indexNahradnehoPrvku**]** **=** **true;**

aktualnaHmotnost **=** novaHmotnost**;**

aktualnaCena **=** novaCena**;**

indexNahradnehoPrvku **=** **-**1**;**

**}**

**}**

**}**

**}**

#### Metódy - [Pôvodné](#_Metódy)

public void vypisVysledkov**()**

public void dualnaVsuvacaHeurestika**()**

public void vlozPrvokDoBatohuSMaxHmotnostou**(**int indexPrvku**)**

public int getPocetRiadkov**(**String nazovSuboru**)**

public void nacitajZoSuboruHmotnost**(**String nazovSuboru**)**

public void nacitajZoSuboruCenu**(**String nazovSuboru**)**

public void zapisVysledokDoSuboru**(**String nazovSuboru**)**

##### public void vypisVysledkov()

* Vypíše konečnú cenu, hmotnosť a počet prvkov v batohu

public void vypisVysledkov**()** **{**

System**.**out**.**println**(**"Aktuálna cena batohu:\t" **+** aktualnaCena**);**

System**.**out**.**println**(**"Aktuálna hmotnosť:\t\t" **+** aktualnaHmotnost**);**

System**.**out**.**println**(**"Aktuálny počet prvkov: \t" **+** aktualnyPocetPrvkov**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Hodnota účelovej funkcie:\t" **+** aktualnaCena**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Index vybraného predmetu... (1 až " **+** m **+** ")"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **((**i **+** 1**)** **%** 100 **==** 0**)** **{**

System**.**out**.**println**();**

**}**

**if** **(**z**[**i**]** **==** **true)** **{**

System**.**out**.**print**(**i **+** 1 **+** ", "**);**

**}**

**}**

**}**

##### public void dualnaVsuvacaHeurestika()

* Pokúsi sa vložiť 500 krát prvok s najväčšou hmotnosťou do batohu až pokiaľ nebude batoh spĺňať všetky podmienky.

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**aktualnyPocetPrvkov **>=** r **&&** aktualnaHmotnost **>=** K**)** **{**

**break;**

**}** **else** **{**

vlozPrvokDoBatohuSMaxHmotnostou**(**getIndexMaximaHmotnosti**());**

**}**

**}**

**}**

##### public void vlozPrvokDoBatohuSMaxHmotnostou(int indexPrvku)

* Vloží do batohu doposiaľ nevložený prvok s najväčšou hmotnosťou a pripočíta cenu, hmotnosť a daný prvok do batohu.

public void vlozPrvokDoBatohuSMaxHmotnostou**(**int indexPrvku**)** **{**

z**[**indexPrvku**]** **=** **true;**

aktualnaHmotnost **+=** a**[**indexPrvku**];**

aktualnaCena **+=** c**[**indexPrvku**];**

aktualnyPocetPrvkov **++;**

**}**

##### public int getPocetRiadkov(String nazovSuboru)

* Vráti počet riadkov z daného súboru

public int getPocetRiadkov**(**String nazovSuboru**){**

Path cesta **=** Paths**.**get**(**nazovSuboru**);**

int riadky **=** 0**;**

**try** **{**

riadky **=** **(**int**)**Files**.**lines**(**cesta**).**count**();**

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**return** riadky**;**

**}**

##### public void nacitajZoSuboruHmotnost(String nazovSuboru)

* Načíta hodnoty z textového súboru do poľa s hmotnosťou

public void nacitajZoSuboruHmotnost**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException **{**

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i**=**0**;** i**<**m**;** i**++)** **{**

a**[**i**]=**citac**.**nextInt**();**

**}**

citac**.**close**();**

**}**

##### public void nacitajZoSuboruCenu(String nazovSuboru)

* Načíta hodnoty z textového súboru do poľa s hmotnosťou

public void nacitajZoSuboruCenu**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException **{**

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i**=**0**;** i**<**m**;** i**++)** **{**

c**[**i**]=**citac**.**nextInt**();**

**}**

citac**.**close**();**

**}**

##### public void zapisVysledokDoSuboru(String nazovSuboru)

* Zapíše výsledok do textového súboru. V zápise je počet predmetov, hmotnosť a cena batohu spolu s účelovou funkciou a index prvkov ktoré sa nachádzajú v batohu.

public void zapisVysledokDoSuboru**(**String nazovSuboru**)** **{**

**try** **{**

FileWriter zapis **=** **new** FileWriter**(**nazovSuboru**);**

zapis**.**write**(**"Počet predmetov = " **+** aktualnyPocetPrvkov **+** "\nHmotnosť batohu = " **+** aktualnaHmotnost **+** "\nÚčelová funkcia = " **+** aktualnaCena **+** "\n\n"**);**

zapis**.**write**(**"Index prvku\tHmotnosť\tCena\n"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**z**[**i**]** **==** **true)** **{**

zapis**.**write**(**String**.**valueOf**(**i **+** 1**)** **+** "\t\t" **+** a**[**i**]** **+** "\t\t" **+** c**[**i**]** **+** "\n"**);**

**}**

**}**

zapis**.**close**();**

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**}**

#### Zdrojový kód triedy DualnaHeuristika, aj s komentármi

package com**.**company**;**

**import** java**.**io**.**FileNotFoundException**;**

**import** java**.**io**.**FileWriter**;**

**import** java**.**io**.**IOException**;**

**import** java**.**nio**.**file**.**Files**;**

**import** java**.**nio**.**file**.**Path**;**

**import** java**.**nio**.**file**.**Paths**;**

public class DualnaHeuristika **{**

private int**[]** a**;** //hmotnost

private int**[]** c**;** //cena

private boolean**[]** z**;** // vybrany/nevybrany prvok

private int m**;** // pocet riadkov v textovom subore

private int aktualnaHmotnost**;**

private int aktualnaCena**;**

private int aktualnyPocetPrvkov**;**

private static final int r **=** 200**;** //minimalny pocet prvkov

private static final int K **=** 14000**;** //minimalna kapacita

public DualnaHeuristika**(**String nazovSuboruHmotnost**,** String nazovSuboruCena**)** **throws** FileNotFoundException **{**

aktualnaCena **=** 0**;**

aktualnaHmotnost **=** 0**;**

aktualnyPocetPrvkov **=** 0**;**

m **=** getPocetRiadkov**(**nazovSuboruHmotnost**);** //Ziskam pocet riadkov z textoveho suboru

a **=** **new** int**[**m**];** //inicializacia s nulami

c **=** **new** int**[**m**];** //inicializacia s nulami

z **=** **new** boolean**[**m**];** //batoh je naplneny s false

nacitajZoSuboruHmotnost**(**nazovSuboruHmotnost**);** // do a[] priradi dane hodnoty z .txt

nacitajZoSuboruCenu**(**nazovSuboruCena**);** // do c[] priradi dane hodnoty z .txt

**}**

public void vypisVysledkov**()** **{**

System**.**out**.**println**(**"Aktuálna cena batohu:\t" **+** aktualnaCena**);**

System**.**out**.**println**(**"Aktuálna hmotnosť:\t\t" **+** aktualnaHmotnost**);**

System**.**out**.**println**(**"Aktuálny počet prvkov: \t" **+** aktualnyPocetPrvkov**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Hodnota účelovej funkcie:\t" **+** aktualnaCena**);**

System**.**out**.**println**();**

System**.**out**.**println**(**"Index vybraného predmetu... (1 až " **+** m **+** ")"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **((**i **+** 1**)** **%** 100 **==** 0**)** **{** //po stovkach spravi new line

System**.**out**.**println**();**

**}**

**if** **(**z**[**i**]** **==** **true)** **{**

System**.**out**.**print**(**i **+** 1 **+** ", "**);** //vypise index predmetu ktory je v batohu

**}**

**}**

**}**

public void dualnaVsuvacaHeurestika**()** **{**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**aktualnyPocetPrvkov **>=** r **&&** aktualnaHmotnost **>=** K**)** **{** //Ak splnam dane podmienky zadania, tak koncim

**break;**

**}** **else** **{** //ak podmienky nesplnam, pokracujem dalej

vlozPrvokDoBatohuSMaxHmotnostou**(**getIndexMaximaHmotnosti**());**//vlozim do batohu prvok s najvacsou hmotnostou

**}**

**}**

**}**

public void vylepsenieHeurestiky**()** **{**

int indexNahradnehoPrvku **=** **-**1**;**

int novaCena **=** aktualnaCena**;**

int novaHmotnost **=** aktualnaHmotnost**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{** //prejde kazdy prvok

**if** **(**z**[**i**]** **==** **true)** **{** //ak je dany prvok zaradeny, skusim ho vymenit

**for** **(**int j **=** 0**;** j **<** m**;** j**++)** **{** //skusa dany prvok vymenit so vsetkymi ostatnymi prvkami

**if** **(**j **==** i **||** **(**z**[**j**]** **==** **true))** **continue;** //Ak chcem vymenit sam seba, alebo prvok, ktory je uz zaradeny, tak ho preskocim

int hmotnostPoVymene **=** aktualnaHmotnost **-** a**[**i**]** **+** a**[**j**];** //Vypocitana hmotnost po vymene

int cenaPoVymene **=** aktualnaCena **-** c**[**i**]** **+** c**[**j**];** //Vypocitana cena po vymene

**if** **(**hmotnostPoVymene **>=** K **&&** cenaPoVymene **<** aktualnaCena**)** **{** //Ak splna podmienku K a cena je lepsia

**if** **(**cenaPoVymene **>** novaCena**)** **continue;** //Ak som nasiel druheho kandidata na vymenu,tak

//Ak ma novy kandidat vacsiu cenu, tak ho preskocim

indexNahradnehoPrvku **=** j**;** //Zapamatam si index vymeneneho prvku

novaCena **=** cenaPoVymene**;** //Zapamatam si novuCenu kandidata

novaHmotnost **=** hmotnostPoVymene**;** //Zapamatam si novu hmotnost kandidata

**}**

**}**

**if** **(**indexNahradnehoPrvku **!=** **-**1**)** **{** //Ak sa nasiel nejaky kandidat, tak...

z**[**i**]** **=** **false;** //Predchodcu z batohu vyhodime

z**[**indexNahradnehoPrvku**]** **=** **true;** //Zaradime do batohu vymeneny prvok

aktualnaHmotnost **=** novaHmotnost**;** //Aktualizujem hmotnost batohu

aktualnaCena **=** novaCena**;** //Aktualizujem cenu batohu

indexNahradnehoPrvku **=** **-**1**;** //Resetujem index nahradneho prvku

**}**

**}**

**}**

**}**

public void vlozPrvokDoBatohuSMaxHmotnostou**(**int indexPrvku**)** **{**

z**[**indexPrvku**]** **=** **true;** //oznacim dany prvok, ze uz som ho vlozil do batohu

aktualnaHmotnost **+=** a**[**indexPrvku**];** //pripocitam hmotnost prvku

aktualnaCena **+=** c**[**indexPrvku**];** //pripocitam cenu prvku

aktualnyPocetPrvkov**++;** //pripocitam jednu jednotku k aktualnemu poctvu prvkom

**}**

public int getIndexMaximaHmotnosti**()** **{**

int docasneMaximum **=** **-**1**;**

int indexMaxima **=** **-**1**;**

int cenaMaxima **=** **-**1**;**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**a**[**i**]** **>=** docasneMaximum **&&** z**[**i**]** **!=** **true)** **{** //ak dany prvok este nie je v batohu a je zatial najvacsi

**if** **(**a**[**i**]** **==** docasneMaximum **&&** cenaMaxima **<** c**[**i**])** **{**

**continue;**

**}** **else** **{**

docasneMaximum **=** a**[**i**];** //zvolim ho ako maximalny prvok

indexMaxima **=** i**;** //priradim index maximalneho prvku

cenaMaxima **=** c**[**i**];**

**}**

**}**

**}**

**return** indexMaxima**;** //vratim index maximalneho prvko

**}**

public int getPocetRiadkov**(**String nazovSuboru**)** **{**

Path cesta **=** Paths**.**get**(**nazovSuboru**);**

int riadky **=** 0**;**

**try** **{**

riadky **=** **(**int**)** Files**.**lines**(**cesta**).**count**();** //pretypovanie na INT, vypocita kolko tam je riadkov

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**return** riadky**;**

**}**

public void nacitajZoSuboruHmotnost**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException **{**

// vytvorenie novej instancie triedy Scanner pre citanie z textoveho suboru

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{** //pre vsetky riadky

a**[**i**]** **=** citac**.**nextInt**();** // nacitanie hodnoty prvku zo suboru

**}**

citac**.**close**();**

**}**

public void nacitajZoSuboruCenu**(**String nazovSuboru**)** **throws** java**.**io**.**FileNotFoundException **{**

// vytvorenie novej instancie triedy Scanner pre citanie z textoveho suboru

java**.**util**.**Scanner citac **=** **new** java**.**util**.**Scanner**(new** java**.**io**.**File**(**nazovSuboru**));**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{** //pre vsetky riadky

c**[**i**]** **=** citac**.**nextInt**();** // nacitanie hodnoty prvku zo suboru

**}**

citac**.**close**();**

**}**

public void zapisVysledokDoSuboru**(**String nazovSuboru**)** **{**

**try** **{**

FileWriter zapis **=** **new** FileWriter**(**nazovSuboru**);**

zapis**.**write**(**"Počet predmetov = " **+** aktualnyPocetPrvkov **+** "\nHmotnosť batohu = " **+** aktualnaHmotnost **+** "\nÚčelová funkcia = " **+** aktualnaCena **+** "\n\n"**);**

zapis**.**write**(**"Index prvku\tHmotnosť\tCena\n"**);**

**for** **(**int i **=** 0**;** i **<** m**;** i**++)** **{**

**if** **(**z**[**i**]** **==** **true)** **{**

zapis**.**write**(**String**.**valueOf**(**i **+** 1**)** **+** "\t\t" **+** a**[**i**]** **+** "\t\t" **+** c**[**i**]** **+** "\n"**);**

**}**

**}**

zapis**.**close**();**

**}** **catch** **(**IOException e**)** **{**

System**.**out**.**println**(**"Vyskytla sa chyba!"**);**

e**.**printStackTrace**();**

**}**

**}**

**}**

## Záverečné zhodnotenie

### Účelová funkcia

= 18897

### Konečný stav batohu

Aktuálna cena batohu: 18897

Aktuálna hmotnosť: 14001

Aktuálny počet prvkov: 200

### Zhodnotenie

Daná heuristika nám zabezpečila cenu batohu rovnej 18897, pričom prvkov bolo 200 a hmotnosť batohu bola 14001.

Zlepšenie oproti prvému zadaniu je:  
Účelová funkcia (cena batohu) sa zmenšila o 2163 jednotiek.  
Hmotnosť batohu sa zmenšila o 830 jednotiek.

Výsledok je prípustný a mal by byť aj naoptimálnejší za daných podmienok.  
  
Malo by byť jedno aké lokálne kritérium by sme použili pri pôvodnej heurestike, pretože nakoniec sa mi aj tak vo vylepšenej úlohe vymení každý prvok s každým a vyberie sa len ten najoptimálnejší.

Preto považujem moje bonusové riešenie za najlepšie za daných podmienok.

Pri väčších dátach by však úloha mohla trvať dlhšie a ak by sme sa mali zamerať len na čas, s tým, že výsledok by mohol mať väčšiu odchýlku, tak by bola výhodnejšia prvá úloha, kde sme použili lokálne kritérium výberu maximálnej hmotnosti, bez pridaného algoritmu.

1. Dané vylepšenie zmenilo účelovú funkciu o 6 jednotiek. Váha a počet prvkov ostali rovnaké. [↑](#footnote-ref-1)