2. Exaktní a heuristická řešení konstruktivního problému batohu

Matej Šutý
NI-KOP 2021
Praha

Popis implementovaných metod **2**
Branch and bound: **2**
Dekompozícia podľa ceny: **2**
Greedy a redukovaný greedy algoritmus: **3**
Greedy algoritmus: 3
Redukovaný greedy algoritmus: 3
FPTAS, Fully Polynomial Time Approximation Scheme: **4**

Srovnání výpočetních časů metody větví a hranic, dynamického programování a heuristiky **5**

Porovnání relativních chyb (průměrných a maximálních) obou heuristik. **7**
Absolútna chyba 8

FPTAS: závislost chyby a výpočetního času algoritmu na zvolené přesnosti zobrazení **9**
Závisloť chyby a výpočtového času: 10
Porovnanie maximálnej chyby s teoretickou očakávanou chybou: 11

Zhodnocení naměřených výsledků **13**
Odpovídají obě závislosti (kvality a času) předpokladům? 13
Je některá heuristická metoda systematicky lepší (tzv. dominance) v některém kritériu? 13
Jak se liší obtížnost jednotlivých sad z hlediska jednotlivých metod? 14
Jaká je závislost maximální chyby (ε) a času FPTAS algoritmu na zvolené přesnosti? Odpovídá předpokladům? 16

Popis implementovaných metod

Branch and bound:

Popísané v predchádzajúcom reporte, vrátane konštruktívnej verzie, na mojom fakultom gitlabe.

V krátkosti:

Algoritmus vytvára binárny strom, kde v úrovni d_i pridá/nepridá predmet *i* do batohu. V každom liste ukladá hornú hranicu hodnoty batohu a súčasnú hodnotu. Ak niektorý list má hornú hranicu nižšiu ako súčasná hodnota iného listu, znamená to, že podstrom tohto listu určite nebude mať vyššiu hodnotu ako sme doteraz našli a preto sa tento list zatvorí a nepokračuje v ňom expanzia. Tento algoritmus je systematický a vždy prináša optimálne riešenie. Jeho zložitosť môže byť rovnako veľká ako zložitosť hrubej sily $O(2^n)$.

Dekompozícia podľa ceny:

1. Nájsť maximálnu cenu c_i predmetu, ktorý **môže** byť v batohu (w_i <= *M*).
2. Vytvoriť tabuľku s rozmermi (*počet predmetov* + *1*) *x* c_i
3. Vyplniť tabuľku a vypĺňať dopredným spôsobom
    a. *W(0,0) = 0*
       W(0,c) = ∞ pro všechna c > 0
       $W(i+1, c) = \min(W(i, c), W(i, c-c_{i+1})+w_{i+1})$ *pro všechna i > 0.* *
4. Nájsť najvyššiu hodnotu batohu
    a. *Výsledné řešení je určeno polem v posledním sloupci, které obsahuje váhu menší než M a má maximální index. Jestliže pole (n-1, c) obsahuje váhu stejnou jako pole (n, c), pak n-tá věc není součástí optimálního řešení Takto se dá rekonstruovat vektor X z tabulky.* *

* **prevzaté z cvičenia NI-KOP**

Dekompozícia vždy nájde optimálne riešenie (môže byť nulové). Algoritmus je pseudopolynomiálny so zložitosťou $O(n^2\, c_{max})$. Zložitosť závisí na maximálnej cene predmetu.

Greedy a redukovaný greedy algoritmus:

Greedy algoritmus:

Hladový algoritmus vyberá predmety podľa najvyššieho pomeru *cena/váha* až kým pridanie ďalšie predmety neprekročí kapacitu batohu.
Relatívna kvalita tohto algoritmu je ľubovoľne veľká ako vidno na ukážke prevzatej z cvičenia NI-KOP:

Konstrukce příkladu selhání

- položíme $M = kn$, $k > 0$ libovolné
- položíme $v_1 = v_2 = ...= v_n-1 = k$ a $v_n = M$.
- položíme $c_1 = c_2 = ...= c_n-1 = k$ a $c_n = M$-1.

Heuristika vezme nejprve věci 1 ... n-1, protože mají poměr cena/váha rovný 1, zatímco n-tá věc má poměr cena/váha (M-1)/M. Dále, n-tá věc se do batohu již nevejde. Dosařená cena je proto n-1. Optimální řešení je pouze n-tá věc. Relativní kvalita je tedy

$$\frac{M-1}{n-1} > \frac{M}{n} = k$$

Redukovaný greedy algoritmus:

Algoritmus vyberie najdrahší predmet, ktorý sa vojde do batohu a jeho cenu porovná s cenou batohu, ktorý vyrátal pomocou bežného greedy algoritmu a vyberie lepšie riešenie.

FPTAS, Fully Polynomial Time Approximation Scheme:

Knapsack, ktorý je riešiteľný algoritmom s $O(n^2 c_{max})$, môžeme riešiť aproximačným algoritmom, pre ktorý výpočtová zložitosť závisí na požadovanej relatívnej chybe.

Hlavná myšlienka aproximácie spočíva v znížení presnosti cien predmetov, ktoré sa snažíme vložiť do batohu. Možné spôsoby sú :

a) odstrániť posledných ***b*** bitov v cene každého predmetu.

Zložitosť bude $O(n^2.\frac{C_{max}}{2^b})$, v závislosti na požadovanej chybe ε volíme

počet bitov $b = floor(\,log\frac{\varepsilon\,.\,C_{max}}{n})$, n je počet všetkých predmetov.

b) celočíselne vydeliť cenu predmetov v závislosti na **ε**.

Nájsť C_{max} a vypočítať $K = \frac{\varepsilon\,.\,C_{max}}{n}$.

Prepočítať cenu každého predmetu ako $C_i = floor(\frac{C_i}{K})$. Zložitosť bude

$O(n^2.floor(\frac{n}{\varepsilon}))$ *

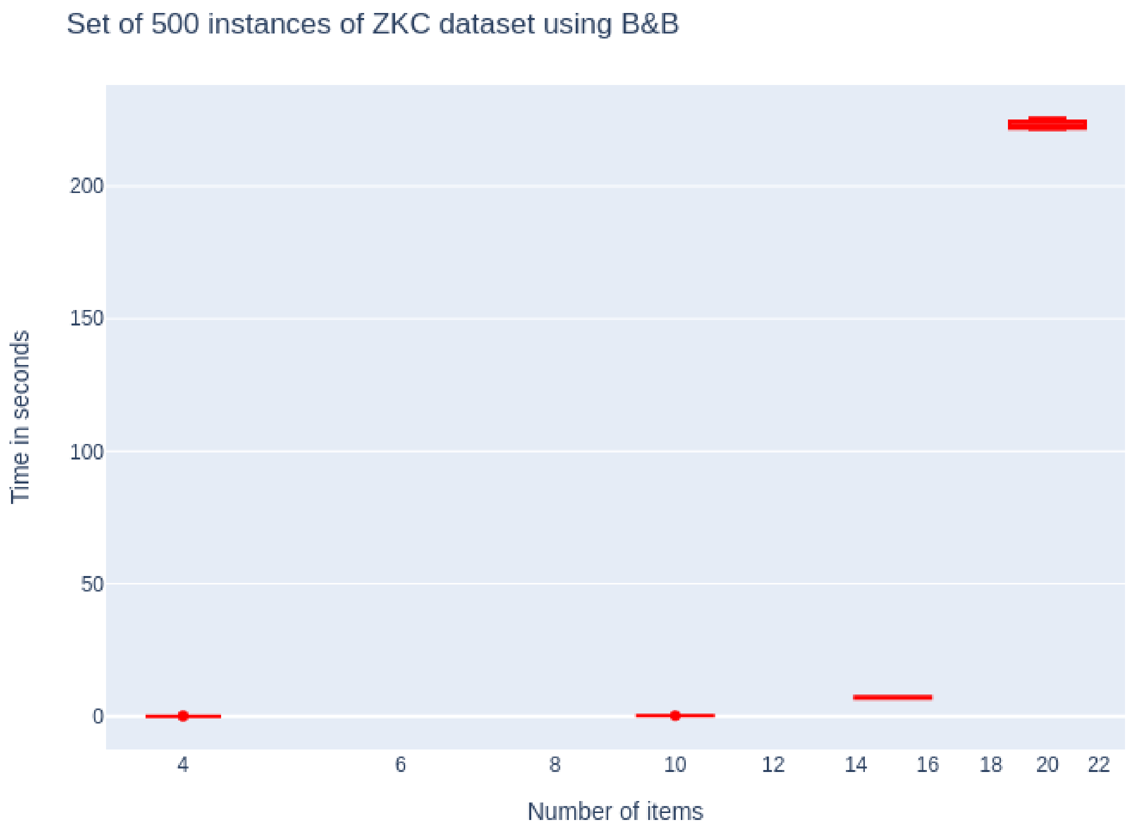
*** prevzaté z**
https://math.mit.edu/~goemans/18434S06/knapsack-katherine.pdf

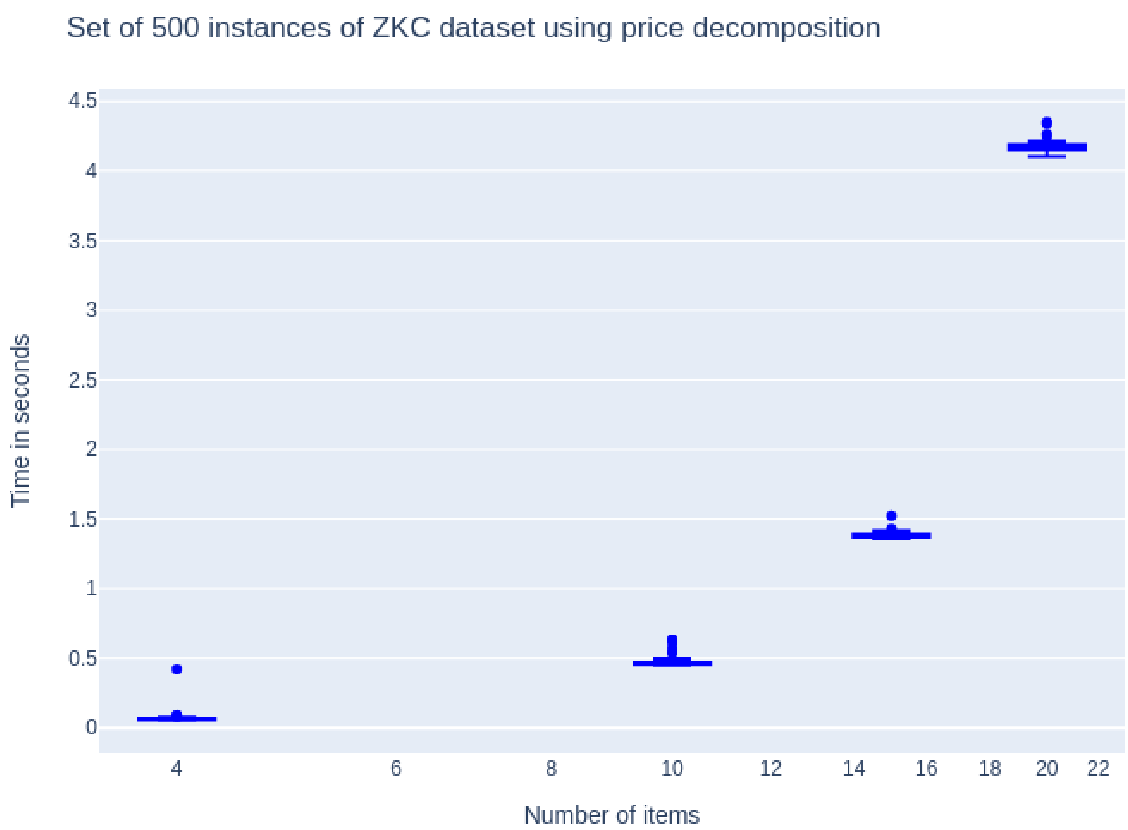
Srovnání výpočetních časů metody větví a hranic, dynamického programování a heuristiky

Metódy *B&B* a *dekompozicía podľa ceny* prinášajú optimálne riešenie narozdiel od greedy heuristík.

Metóda *B&B* je výpočetne náročná a preto som ju testoval na sadách s počtom predmetov pod 25 predmetov. Na výpočtovom čase v sadách s {4, 10, 15, 20} predmetmi je viditeľná dominancia výpočetného času metódy *dekompozicía podľa ceny*. Pre obe metódy trvajú výpočty sekundy až stovky sekúnd, čo sa nedá vizuálne porovnávať s greedy heuristikami, ktoré majú výpočet v stotinách sekúnd.

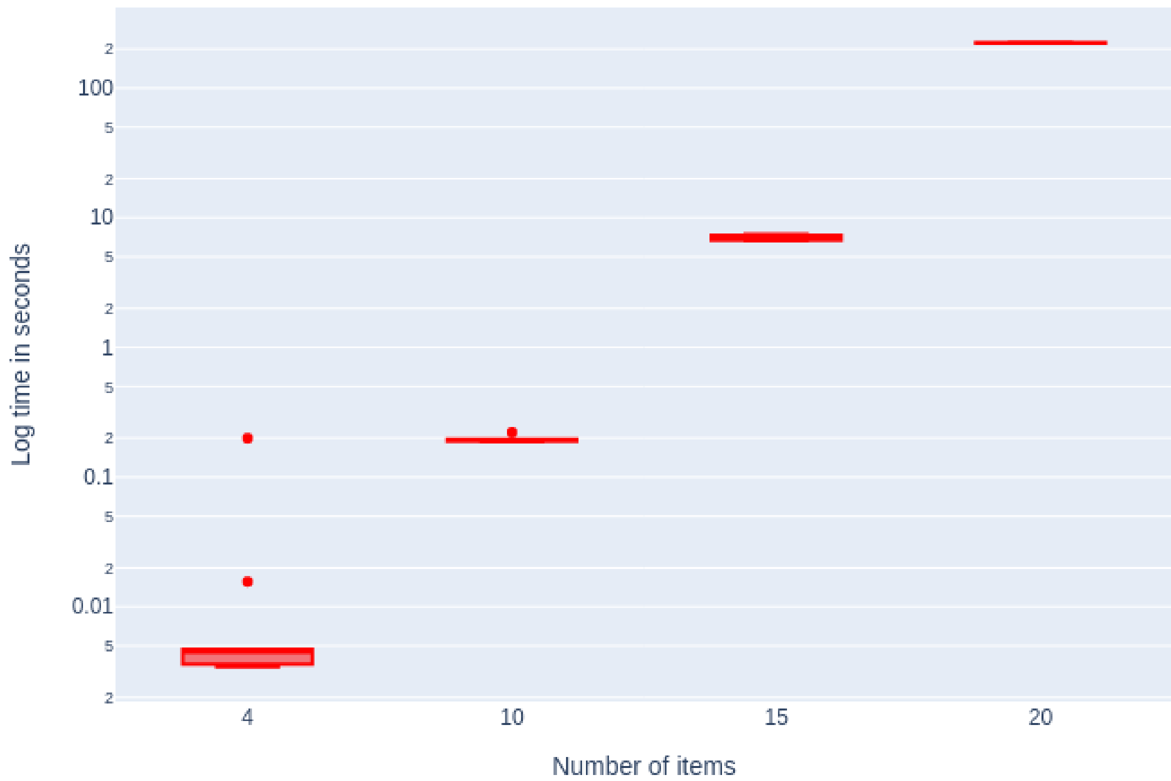
ZKC


Set of 500 instances of ZKC dataset using B&B
Time in seconds
200
150
100
50
0
4
6
8
10
12
14
16
18
20
22
Number of items

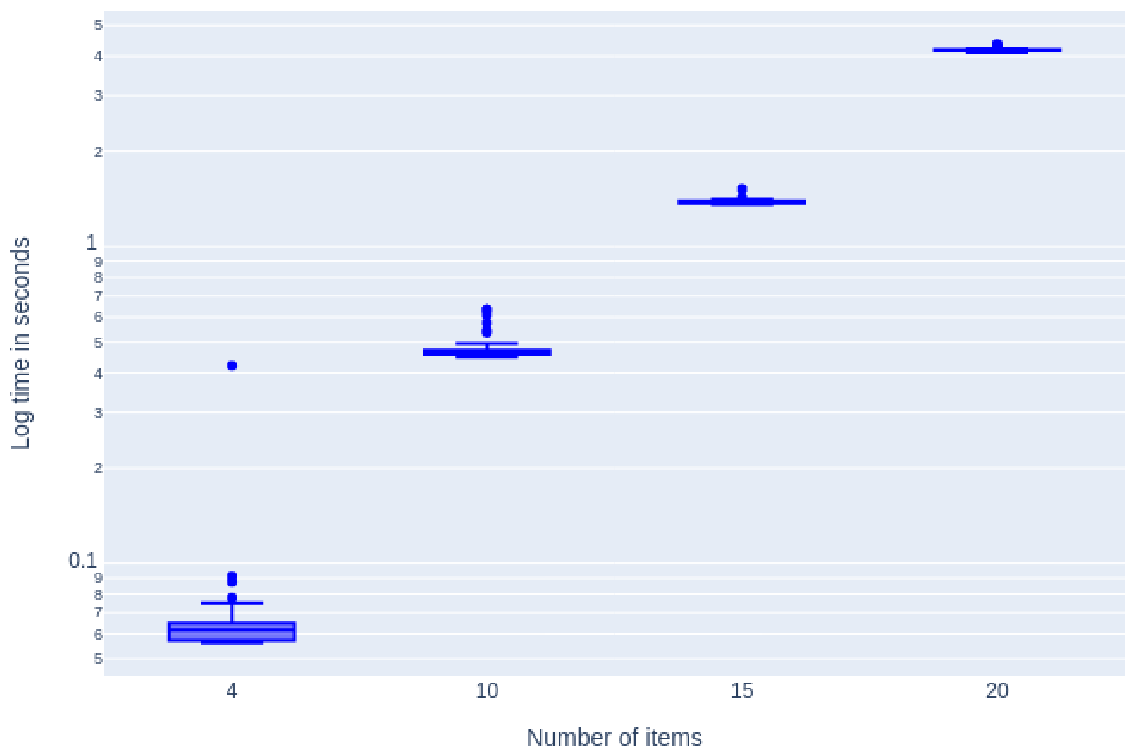

Set of 500 instances of ZKC dataset using price decomposition
Time in seconds
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
4
6
8
10
12
14
16
18
20
22
Number of items

Logarithmic time scale

Set of 500 instances of ZKC dataset using B&B

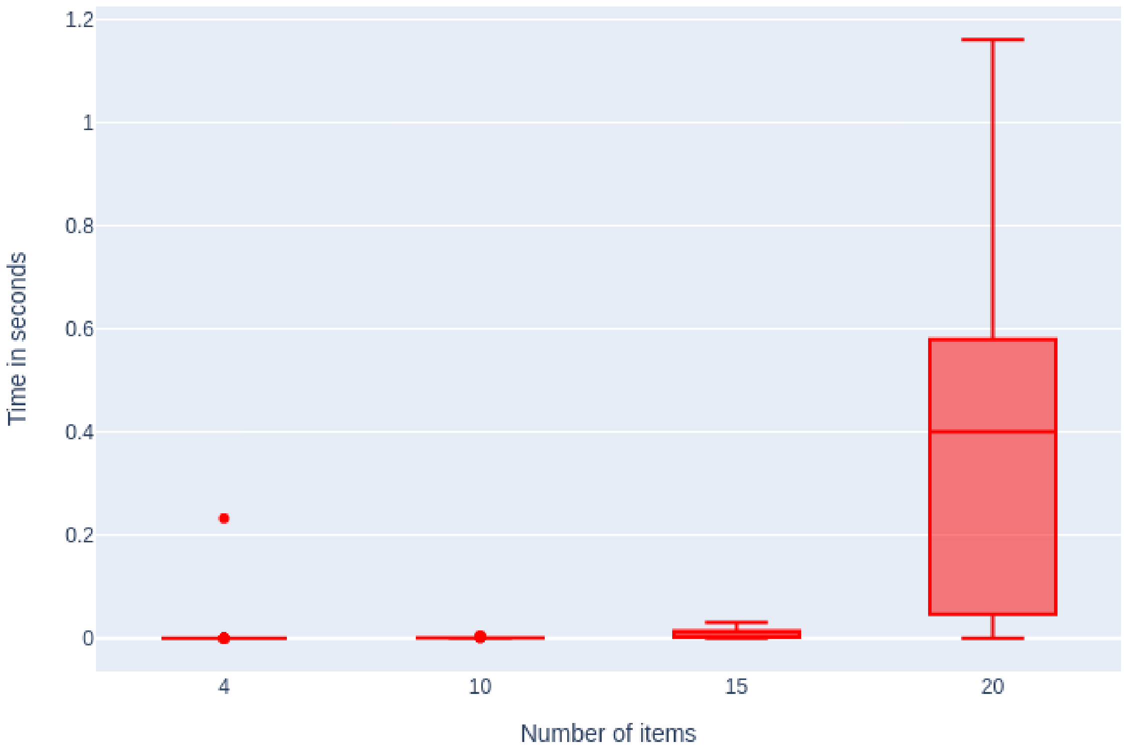


Set of 500 instances of ZKC dataset using price decomposition

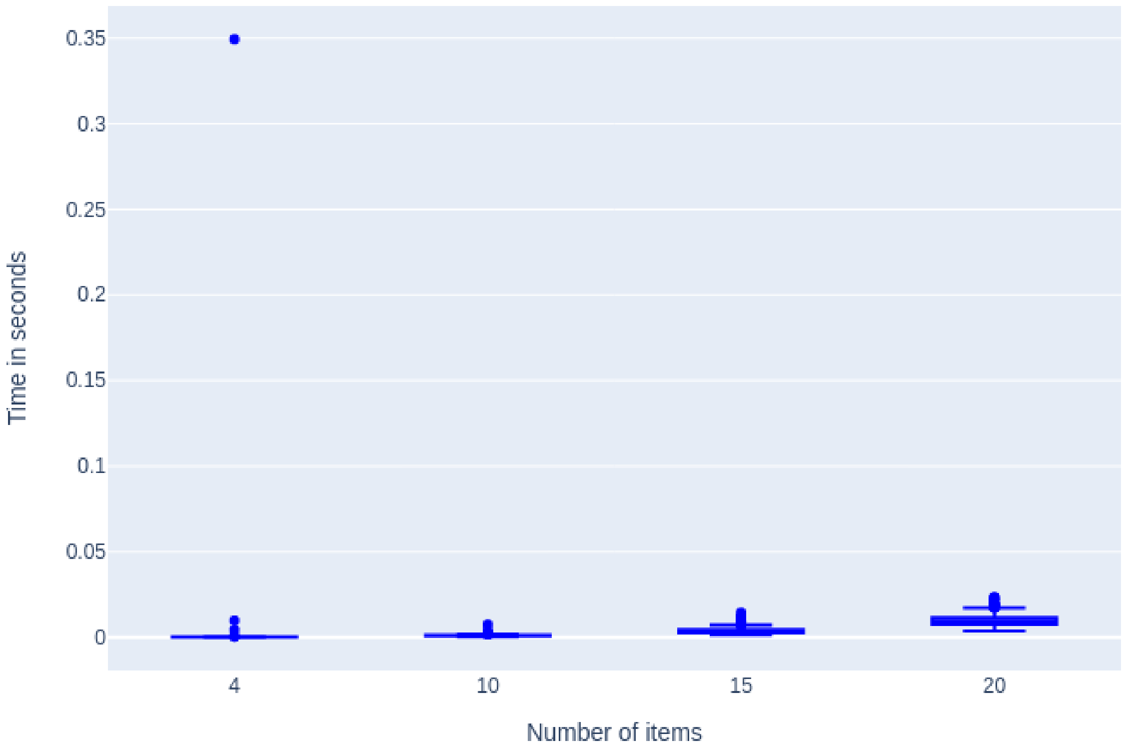


NK

Set of 500 instances of NK dataset using B&B

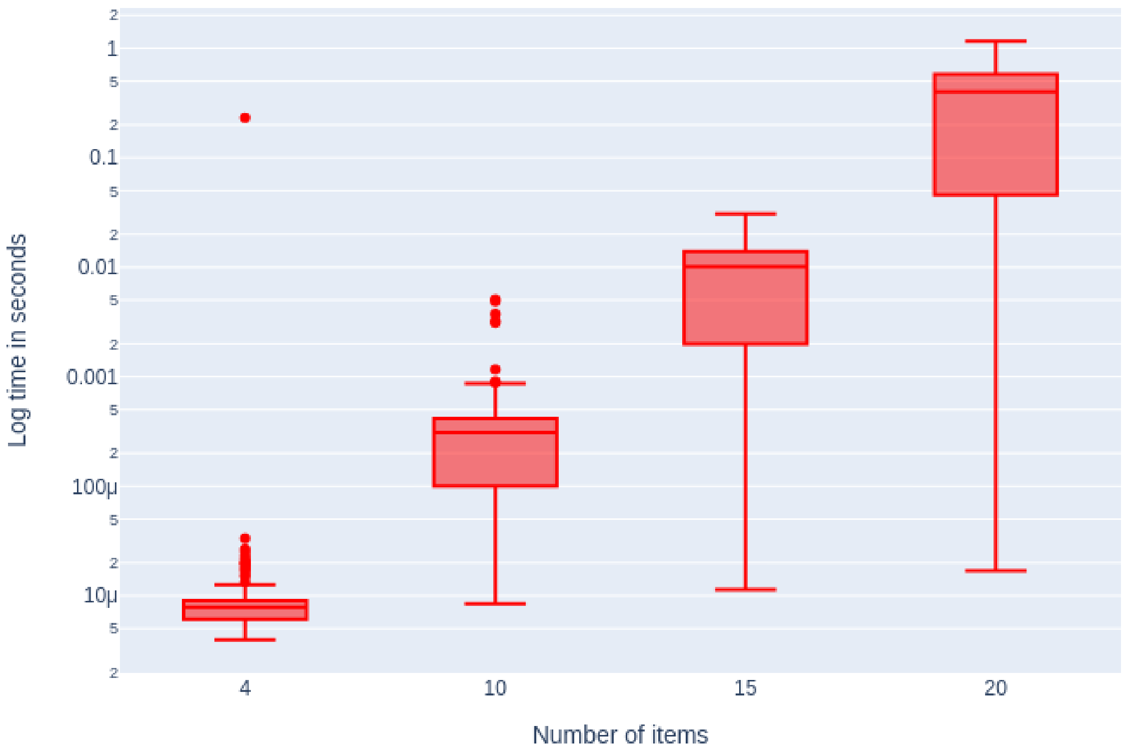


Set of 500 instances of NK dataset using price decomposition

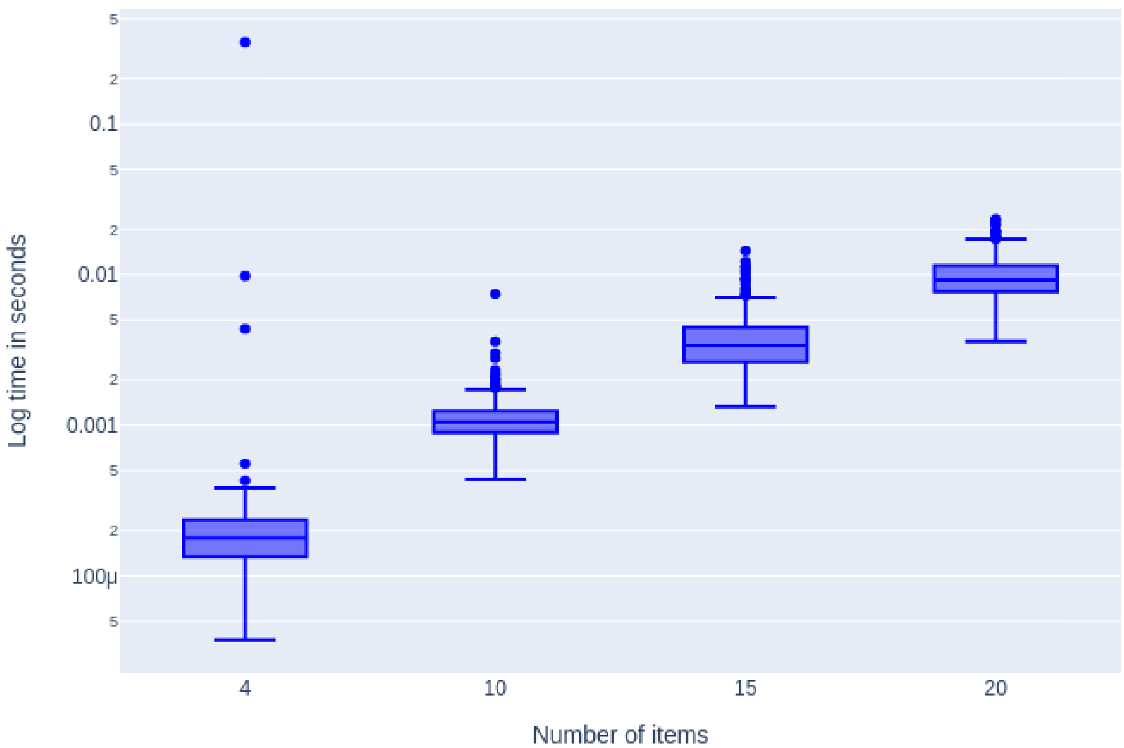


Logaritmic time scale

Set of 500 instances of NK dataset using B&B

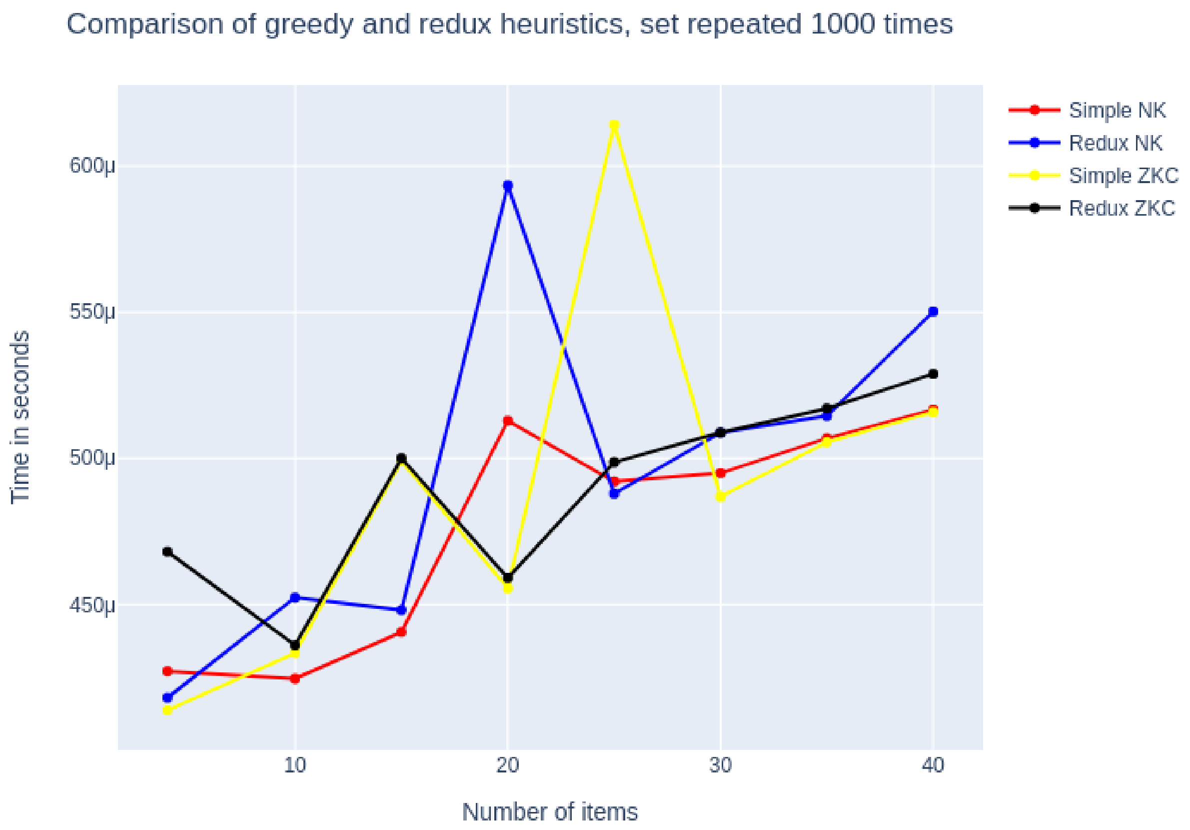


Set of 500 instances of NK dataset using price decomposition



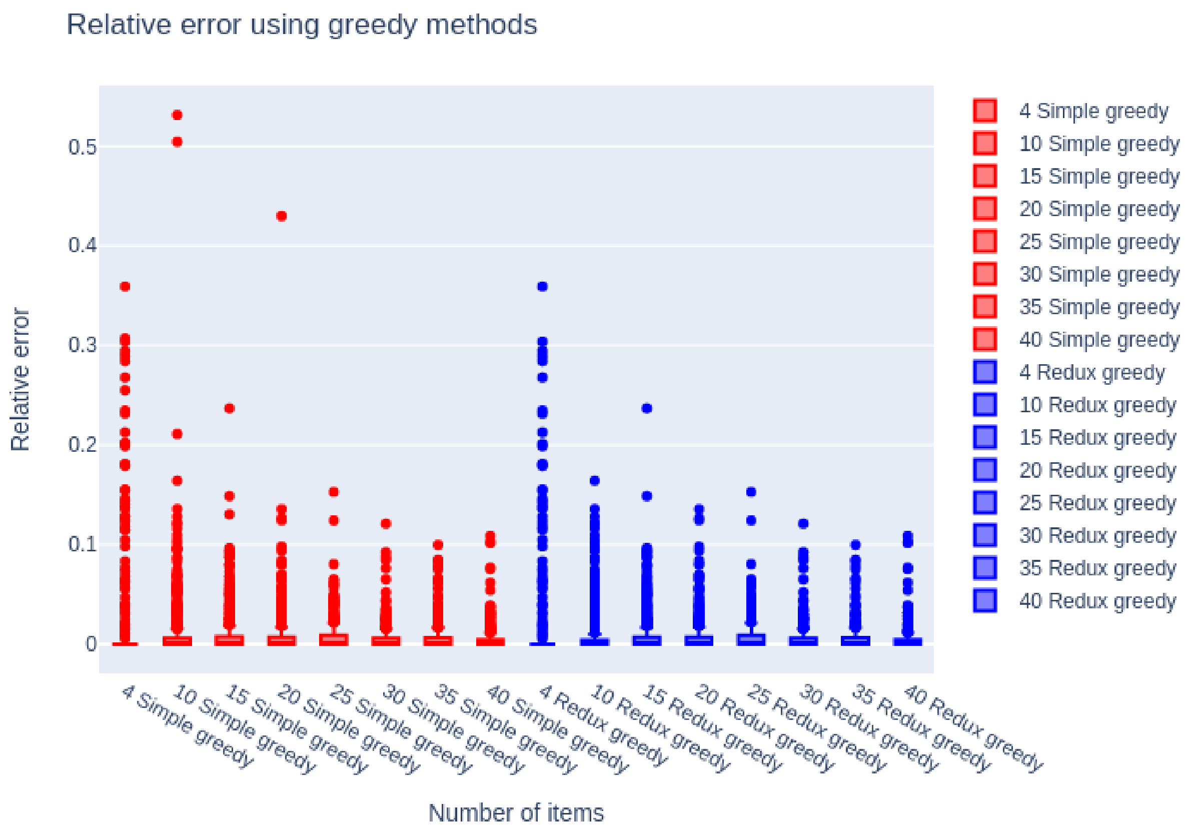
Porovnanie datasetov NK a ZKC na B&B a price decomposition:
ZKC bol výpočetne(časovo) oveľa náročnejší.

Greedy heuristiky nemusia prinášať optimálne riešenie a ich výpočet je veľmi rýchly.
Na nižšie je vidno porovnanie výpočtového času bežnej greedy heuristiky a redukovanej greedy heuristiky na sadách ZKC a NK.Výpočtovo sú obe datasety podobne náročné, je to dané zložitosťou greedy heuristiky, ktorá zoradí predmety podľa pomeru cena/váha (zložitosť podľa radiaceho algoritmu, n.logn) a postupne v lineárnom čase nájde riešenie.

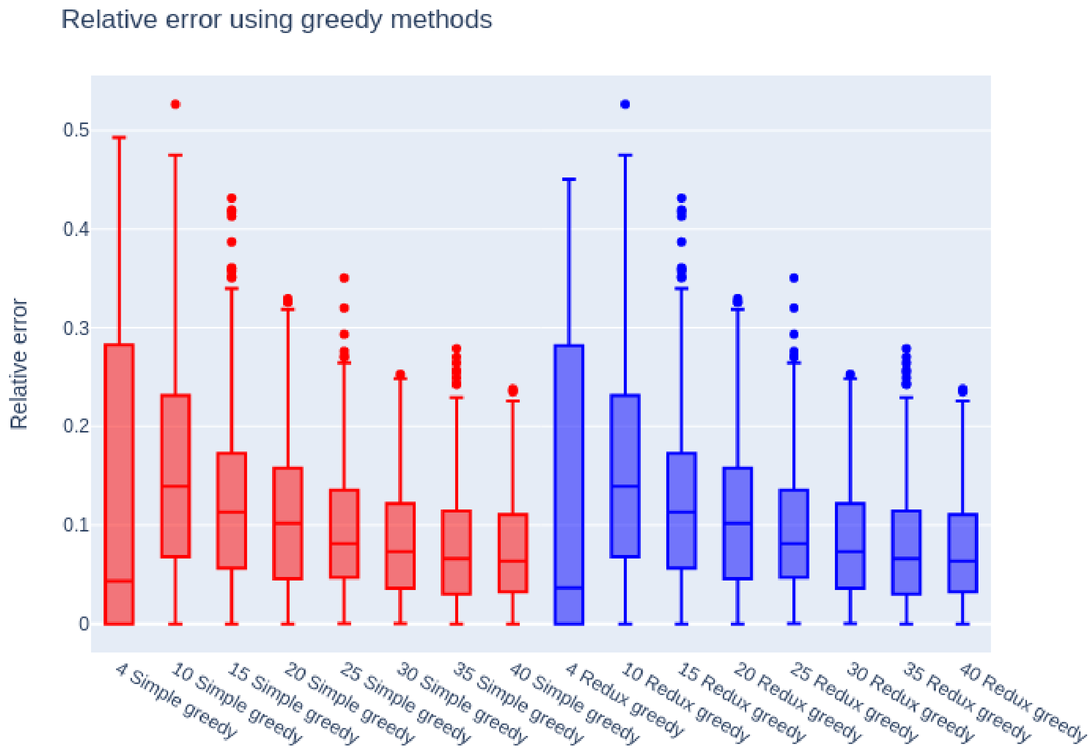


Porovnání relativních chyb (průměrných a maximálních) obou heuristik.

Na grafe nižšie vidno rozdiel v priemerných relatívnych chybách jednoduchej a redukovanej greedy heuristiky. Jednoduchá greedy heuristika mala vyššiu priemernú aj maximálnu chybu v sade NK.

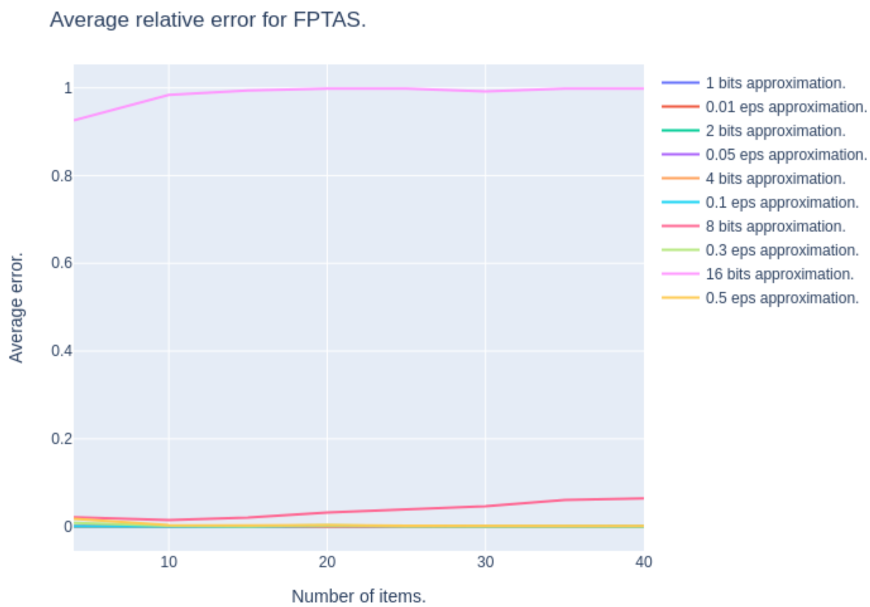


Pre sadu ZKC sú grafy porovnateľné, nie je vidno žiadny rozdiel medzi chybovosťou jednoduchej a redukovanej heuristiky.

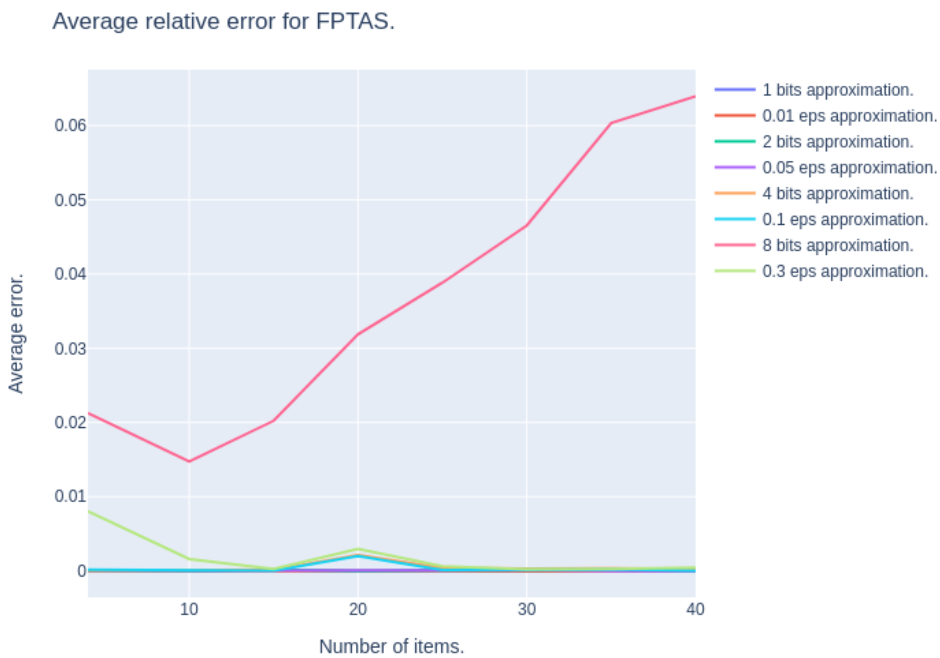


FPTAS: závislost chyby a výpočetního času algoritmu na zvolené přesnosti zobrazení

Na obrázku nižšie sú grafy priemerných relatívnych chýb pre FPTAS aproximácie pomocou bitov aj požadovanej presnosti. Je vidno, že pri zanedbaní posledných 16 bitov je chyba príliš veľká, preto už nebudem zobrazovať tento priebeh. Chybu som počítal na datasete ZKC.

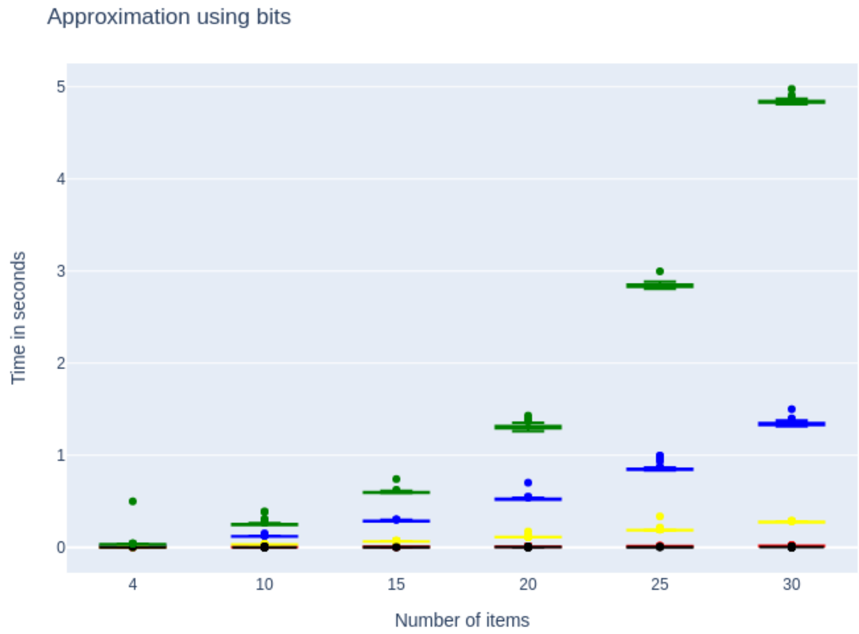


Pre aproximácie pomocout zanedbania {1, 2, 4, 8} bitov a požadovanú presnosť {0.01, 0.05, 0.1, 0.3} vidíme, že priemerná chyba je veľmi nízka.

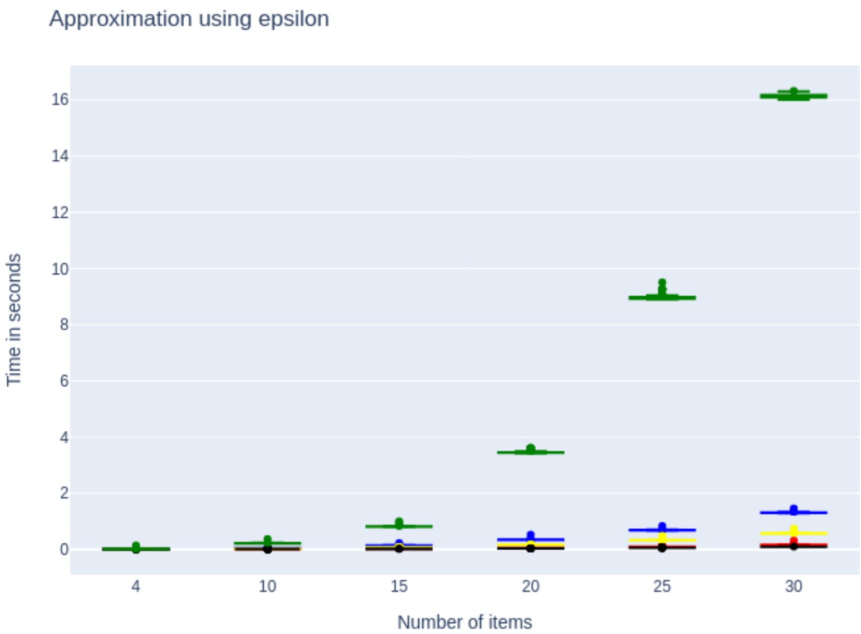


Závisloť chyby a výpočtového času:

Súvisí to aj s výpočtovou náročnosťou modelu - na grafe vidno, že so stúpajúcou presnosťou rastie aj výpočetný čas.



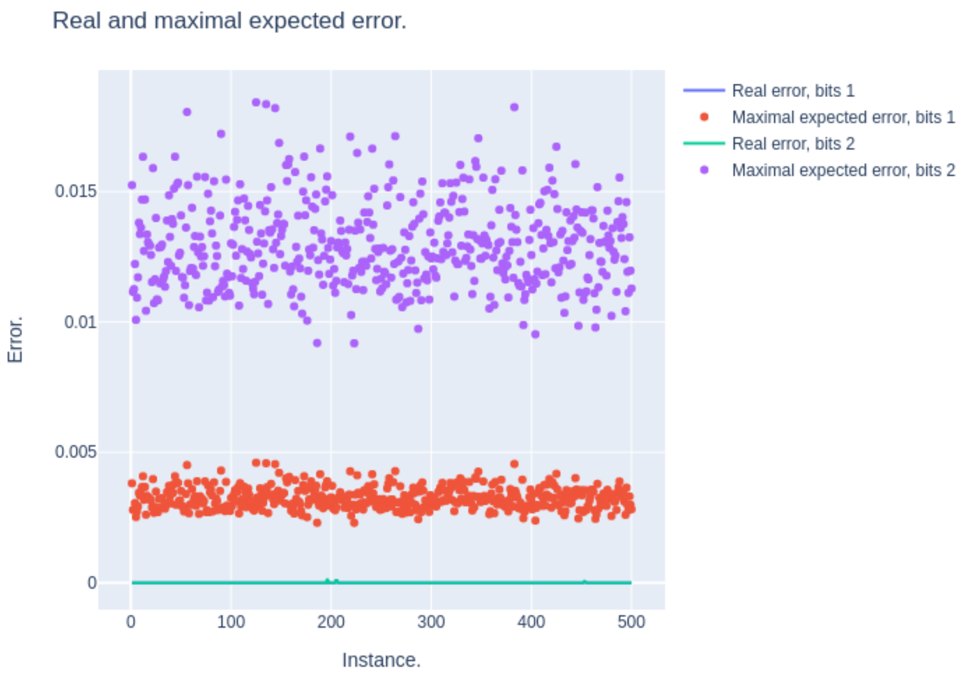
Počet bitov: zelená - 1 bit, modrá 2 bity, žltá 4 bity, červená 8 bitov, čierna 16 bitov



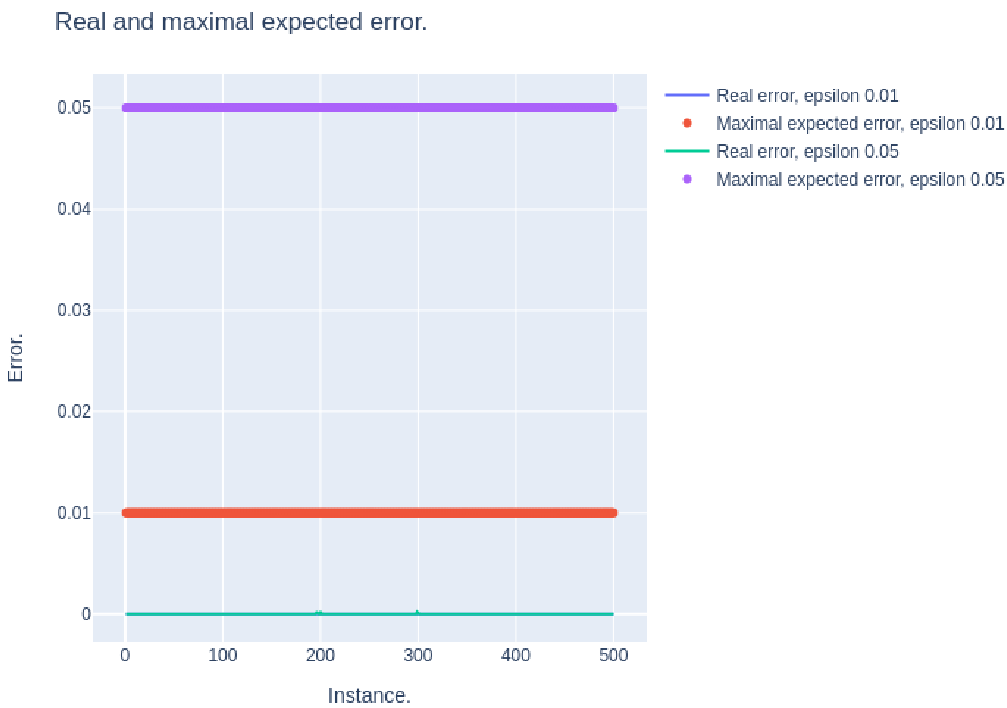
Epsilon: zelená - 0.01, modrá 0.05, žltá 0.1, červená 0.3, čierna 0.5

Porovnanie maximálnej chyby s teoretickou očakávanou chybou:

Na grafe nižšie je vidno skutočnú nameranú chybu (čiara) porovnanú s maximálnou predpokladanou chybou (body). Pre obe hodnoty aproximácie pomocou {1, 2} bitov je skutočná chyba vždy nižšia ako maximálna predpokladaná.



Rovnako aj pre aproximáciu priamo cez maximálnu povolenú chybu.



Zhodnocení naměřených výsledků

1. Odpovídají obě závislosti (kvality a času) předpokladům?

Experimenty ukazujú, že pri znižovaní požadovanej presnosti (kvality) je znížená aj výpočtová náročnosť a čas behu. Ukázalo sa, že skutočná chyba je často oveľa nižšia ako maximálna očakávaná chyba.

2. Je některá heuristická metoda systematicky lepší (tzv. dominance) v některém kritériu?

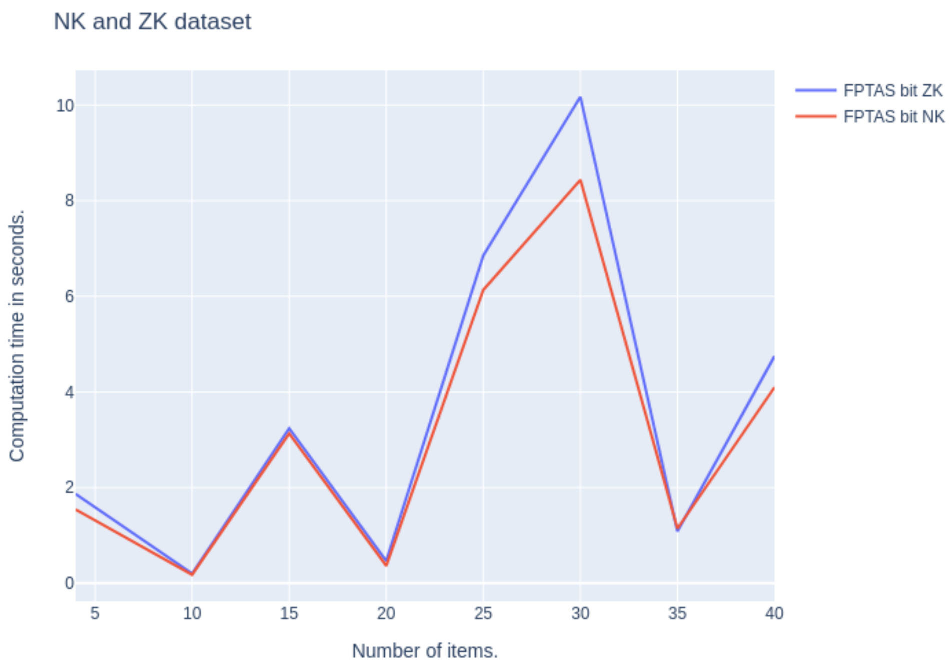
Jednoduchá greedy heuristika má podľa experimentov trend zlepšovať svoju presnosť pri zvyšovaní počtu predmetov. Môže to byť spôsobené tým, že väčší počet predmetov znižuje pravdepodobnosť vybrať nevhodný predmet na základe pomeru cena/váha. Pri redukovanej heuristike je vidno, že má často vysokú chybovosť ale v niektorých prípadoch ako to je vidno v grafe má lepšie výsledky ako jednoduchá heuristika. Výpočtová náročnosť oboch heuristík je porovnanteľná.

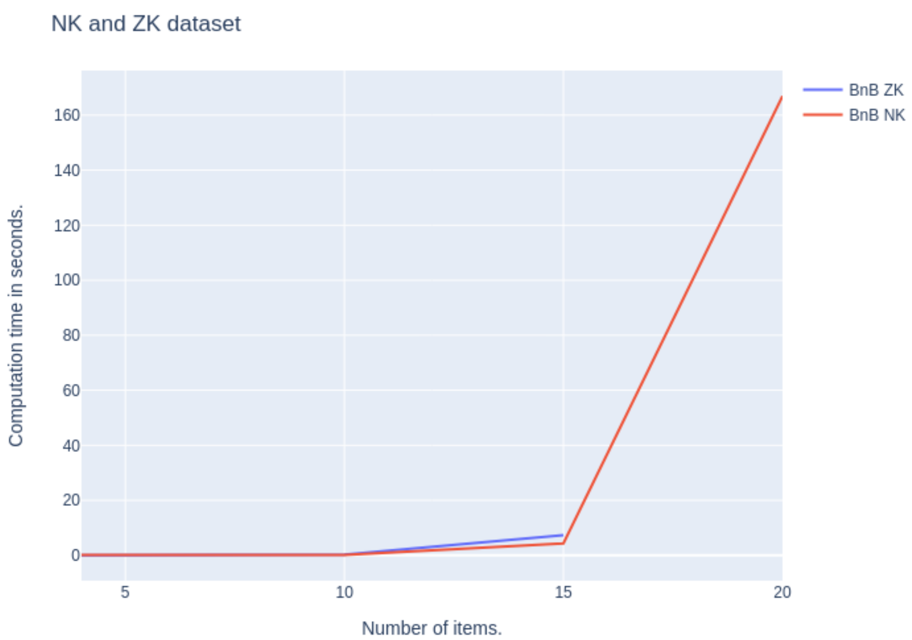
3. Jaká je závislost maximální chyby (ε) a času FPTAS algoritmu na zvolené přesnosti? Odpovídá předpokladům?

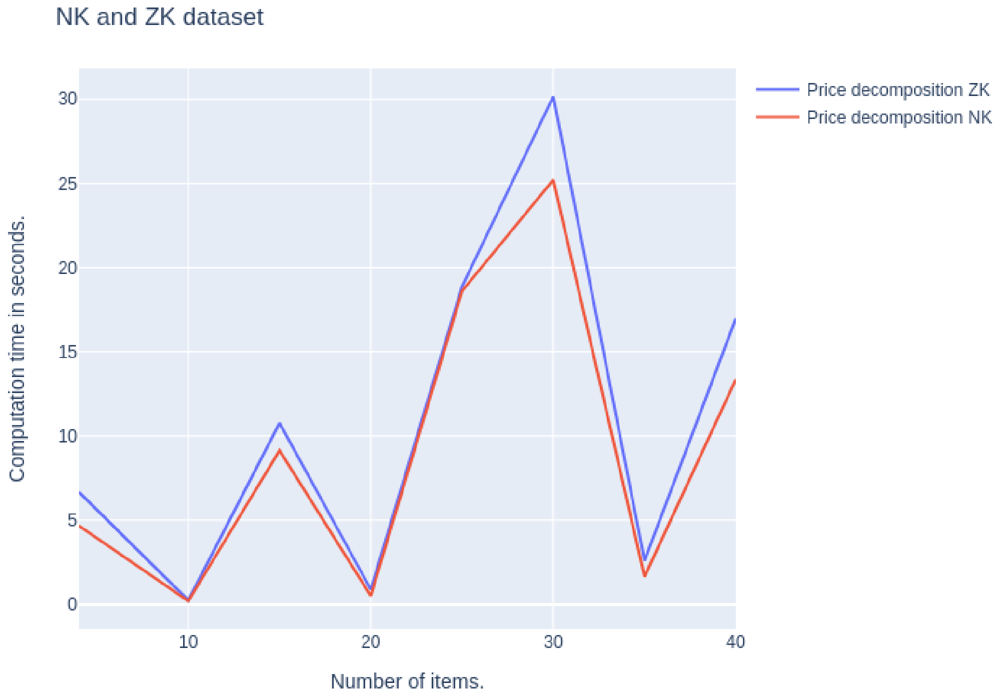
Čas behu algoritmu je polynomiálne závislý od maximálnej požadovanej chyby ako je to vidno na grafoch. Odpovedá predpokladom $O(n^2.floor(\frac{n}{\varepsilon}))$, kde je polynomiálna závisloť na $1/\varepsilon$aj na n.

4. Jak se liší obtížnost jednotlivých sad z hlediska jednotlivých metod?

Sada ZK je výpočtovo zložitejšia ale napriek tomu porovnateľná ku sade NK. Porovnanie chyby medzi sadami som nevykonal.






NK and ZK dataset
Price decomposition ZK
Price decomposition NK
Computation time in seconds.
0
5
10
15
20
25
30
10
20
30
40
Number of items.


NK and ZK dataset
FPTAS eps ZK
FPTAS eps NK
Computation time in seconds.
0
2
4
6
8
10
12
14
16
5
10
15
20
25
30
35
40
Number of items.