NI-KOP

Experimentální hodnocení kvality algoritmů

Úloha 3

Matej Šutý

Zadanie:

Navrhnite a vykonajte experimentálne vyhodnotenie závislostí kvality riešenia a výpočtovej náročnosti algoritmov na nasledujúce parametry instancií:

1. Pomer kapacity batohu k sumárnej váhe
2. Korelácia cena/váha
3. Rozloženie váh a granularita

Samostatnou kategóriou sú testy *robustnosti* algoritmu - schopnosť algoritmu riešiť problém s rozdielnym poradím predmetov v rovnakom/porovnateľnom čase.

Implementované algoritmy:

Algoritmus pri načítaní instancií zoradí všetky predmety zostupne podľa pomeru cena/váha.

Brute force (**BF)**:

Algoritmus vkladá predmety do batohu až do momentu, kým neprekročí kapacitu batohu.

Branch & bound (**BNB)**:

Algoritmus vytvára binárny strom, kde v úrovni $i > 0$ v jednej vetvi pridá predmet p_i a v druhej ho nepridá. Následne overí kapacitu, vypočíta skutočný zisk v uzli a vypočíta hornú hranicu možného maximálneho zisku. Uzly, ktoré prekročia kapacitu uzatvorí. Uzly, pri ktorých je horná hranica nižšia ako najvyšší dosiahnutý zisk tiež uzatvorí.

Price decomposition (**PD)**:

1. Nájsť maximálnu cenu ci predmetu, ktorý môže byť v batohu (wi <= M).
2. Vytvoriť tabuľku s rozmermi (počet predmetov + 1) x ci
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-ci+1)+wi+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

Simple greedy algorithm (**SIMPLE)**:

Hladový algoritmus vyberá predmety podľa najvyššieho pomeru cena/váha až kým pridanie ďalšie predmety neprekročí kapacitu batohu.

Redux greedy algorithm (**REDUX)**:

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.

Experimenty

Rozloženie parametrov:

Pri testovaní som používal generátor instancií, ktorý generuje charakteristické inštancie na základe rôznych vstupných parametrov.
Predpokladom testovania, v krátkosti som ho overil aj počas pilotného testovania, je vzájomná nezávislosť parametrov. Z toho vyplýva, že parametre stačí zafixovať a sledovať správanie algoritmu v závislosti na jednom parametri, ktorý sa mení. V tabuľke nižšie sú popísané použité hodnoty parametrov.

Popis parametru	**Hodnoty**
Počet predmetov **N**	{10, 15, 20} pre **BF, BNB** 100 pre **PD, SIMPLE, REDUX**
Maximálna hmostnosť predmetu **W**	500
Maximálna cena predmetu **C**	1000
Pomer kapacity batohu k sumárnej váhe **m**	0.8
Prevaha ľahkých / ťažkých predmetov **w**	bal
Korelácia s váhou **c**	uni
Exponent granularity **k**	1
Počet permutácií (pri testovaní robustnosti)	50
Počet instancií	10 pre pilotné testovanie 50, 200 pre detailné testovanie

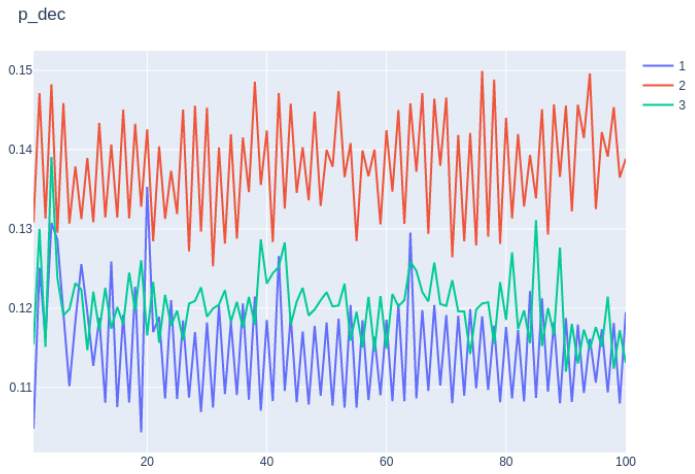
Robustnosť algoritmu

Testovanie robustnosti som vykonal pomocou merania času behu programu. Každá instancia bola prepočítaná trikrát (*attempts=3)* a za čas behu som považoval priemer týchto troch behov. Pri rýchlych výpočtoch pomocou greedy heuristík som na začiatku spustil metódy opakovanie 10x bez merania rýchlostí aby sa vo výsledkoch neobjavili spomalenia kvôli načítaniu kódu.

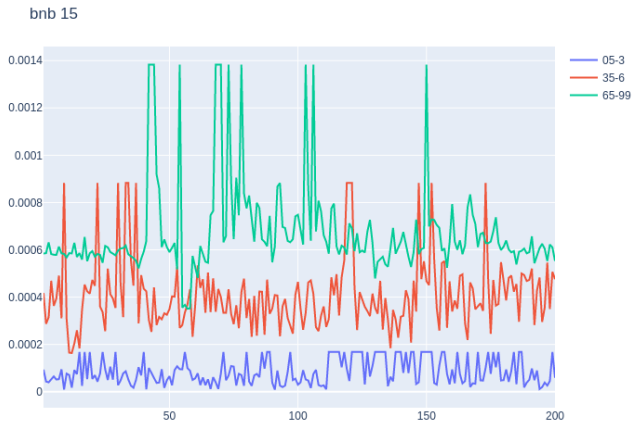
Na overenie robustnosti pre **BF, BNB** som použil 50 permutácií (maximálny počet je 10!), na metódy **PD, SIMPLE, REDUX** som použil 100 permutácií (maximálny počet je 100!).

BF, BNB, PD, SIMPLE, REDUX:

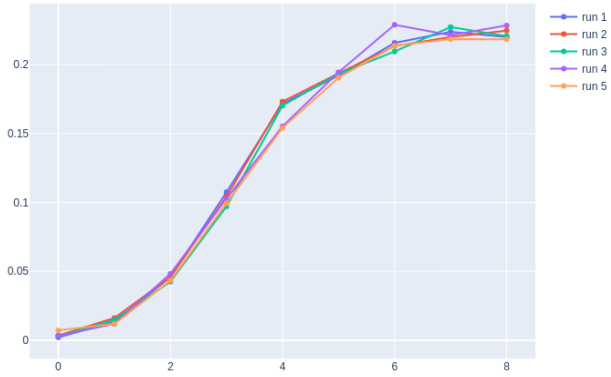
Algoritmus je robustný, vďaka predspracovaniu, pri ktorom sa zoraďujú všetky predmety podľa pomeru, nezávisí na permutácii predmetov. Na grafe nižšie je vidno pílovitý efekt, ktory je podľa mňa spôsobený réžiou procesoru pri dlhých výpočtoch (stotiny sekúnd). Rozdiely nie sú zásadné (rádovo sú rovnaké), podobný priebeh je aj u iných metód. Pri rýchlych metódach heuristík je naopak vidno niekoľko outlierov, ktorí sú tiež spôsobení réžiou, inak je rýchlosť výpočtu takmer rovnaká. O algoritmoch tvrdím, že sú robustné vzhľadom ku permutácii zadania.



Pomer kapacity batohu k sumárnej hmotnosti všetkých vecí



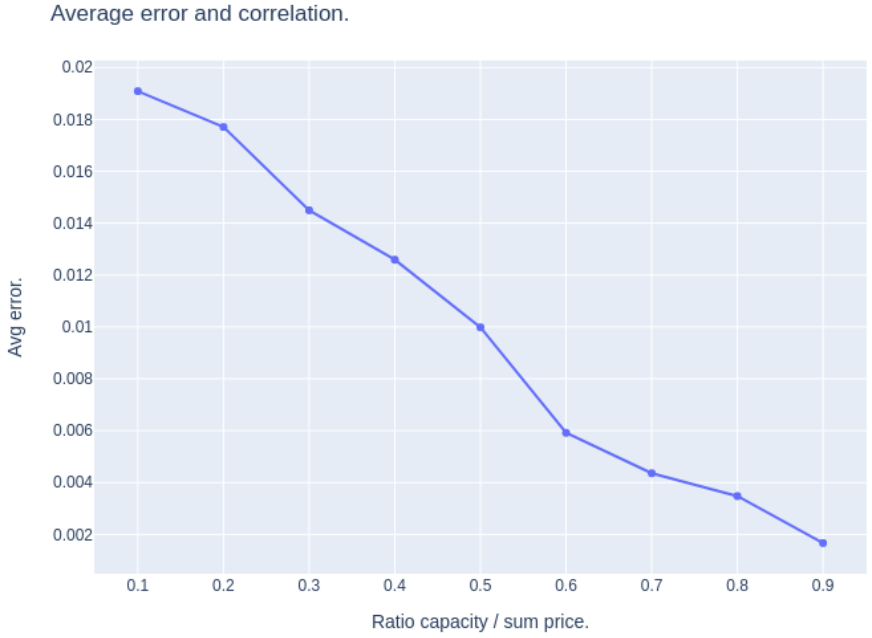
Ratio som nastavil ako tri intervaly: {[0.05, 0.3], [0.35, 0.6], [0.65, 0.99]}. Vplyv na výpočtový čas som zaznamenal u **BRUTE** a **BNB** metódach.



Je jasne viditeľné rozdelenie najvyššieho pomeru v zelenom ako najzložitejšieho na výpočet, modrého ako najjednoduchšieho. Malá kapacita batohu znamená navštívených menej konfigurácií, naopak vysoká kapacita znamená veľa možností a konfigurácií = vyšší čas.

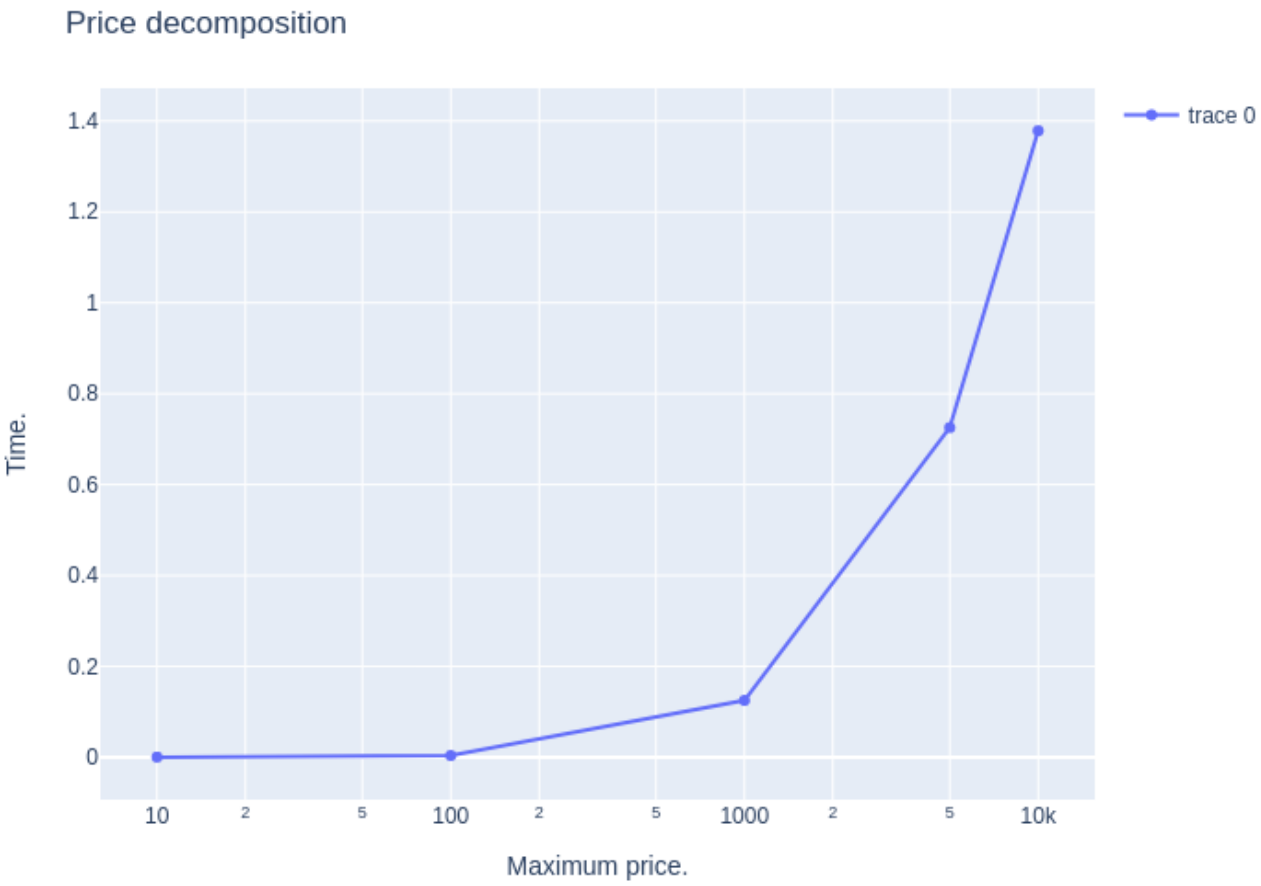
Pre jemnejšie pomery 0.1, 0.2, …, 0.9 je vidno nárast zložitosti. Na ose x je (pomer*10 - 1).

Pomer kapacity a sumárnej ceny má vplyv na priemernú chybu aproximačných heuristík **SIMPLE** a **REDUX.** Môžeme to vysvetliť princípom heuristiky, ktorá sa snaží vkladať všetky predmety, ktoré sa do nej vojdu podľa ich pomeru.



Závislosť na maximálnej cene predmetu

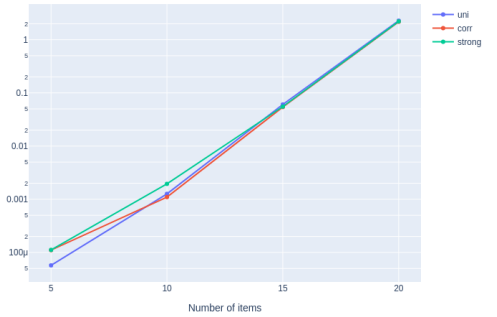
Táto závislosť je prítomná v metóde **PD**, čo je očakávaná vlastnosť pseudopolynomiálneho algoritmu, kde výpočtová zložitosť závisí na parametri najvyššej ceny.



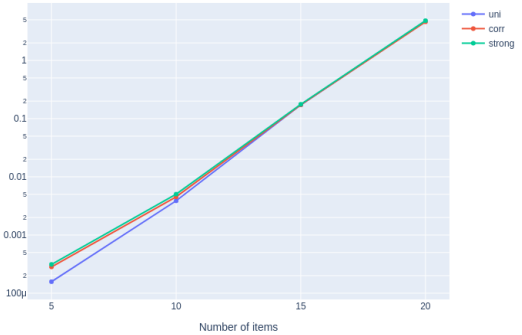
Korelácii ceny a hmotnosti

Závislosť výpočtového času (os Y) som nenašiel v žiadnej optimálnej metóde. Y os je logaritmická.

BF:

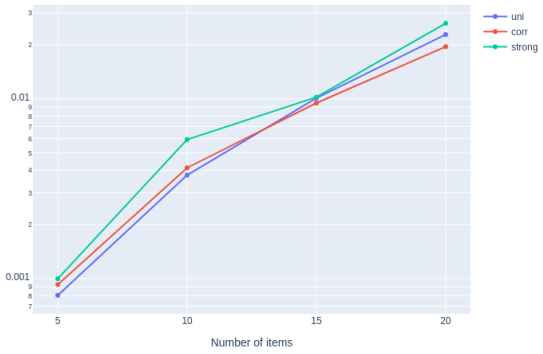


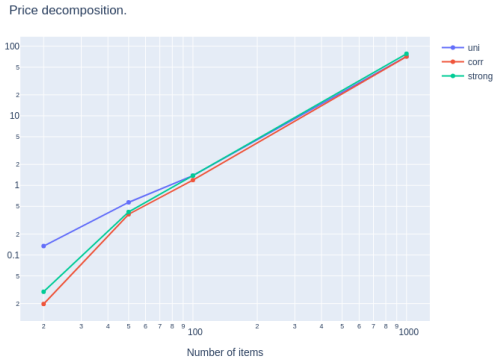
BNB:



PD:

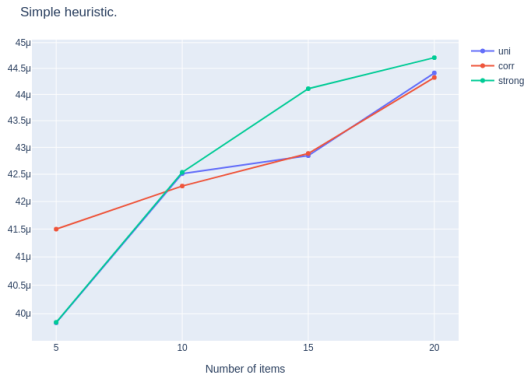
Na grafe vidno, že závislosť výpočtového času je zanedbateľná sa stráca pre veľký počet predmetov (100, 1000).



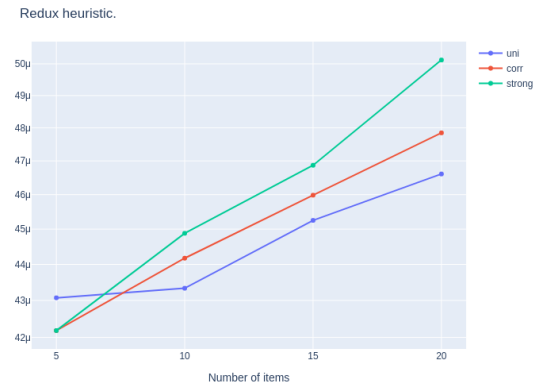


Pre aproximačné metódy sa závislosť výpočtového času (os Y) na korelácii najprv neprejavila veľmi silno.

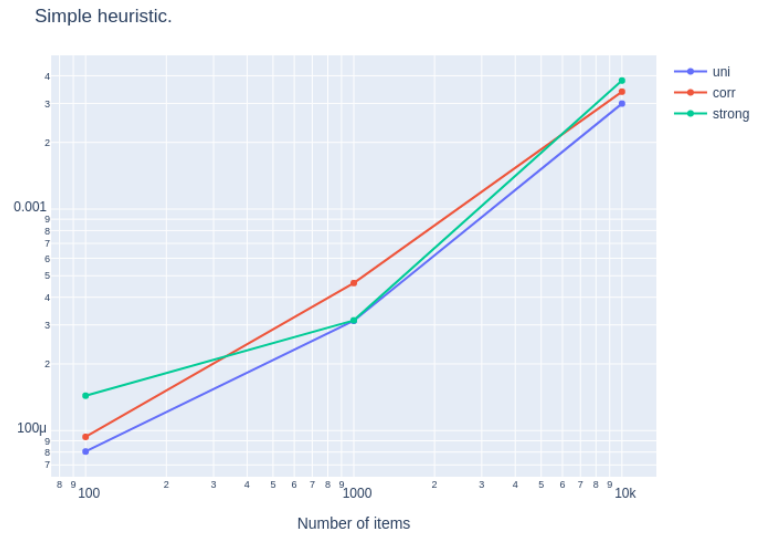
Simple:

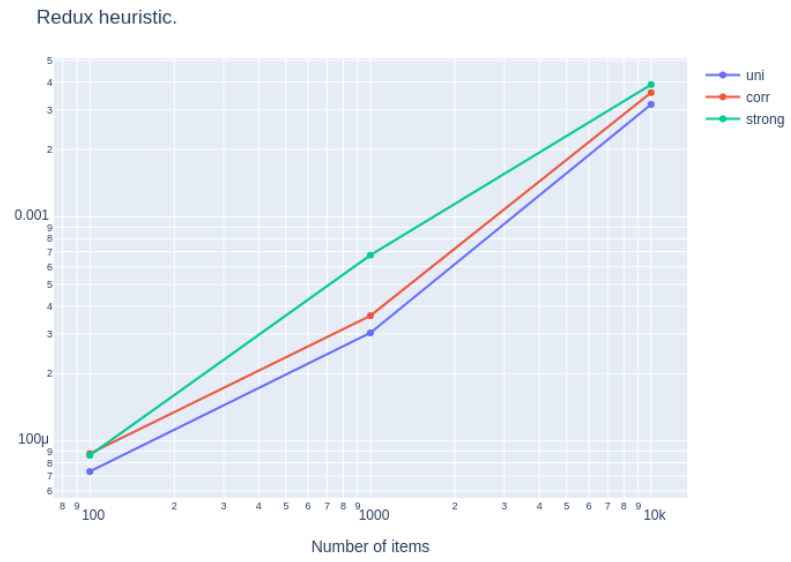


Redux:

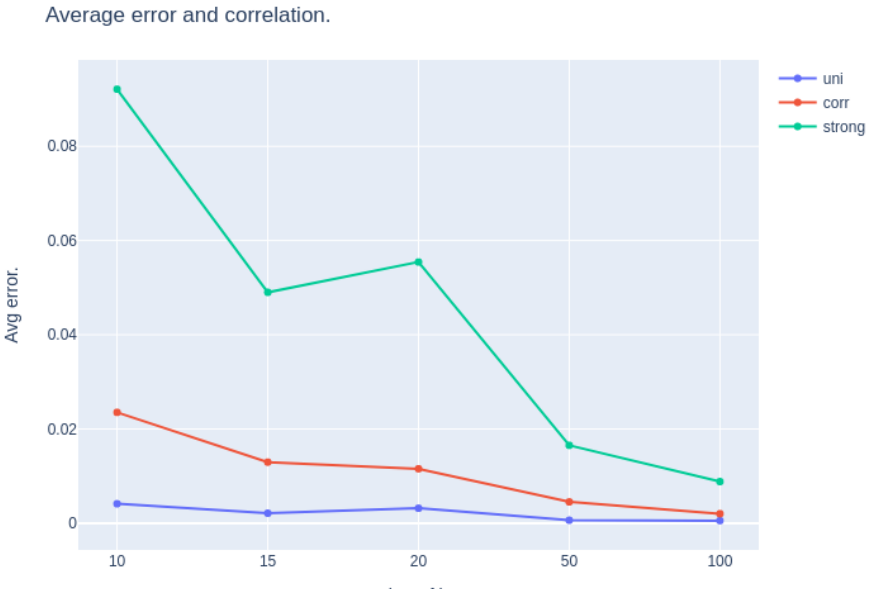


Pre väčší počet predmetov som už zvolil aj os X ako logaritmickú.



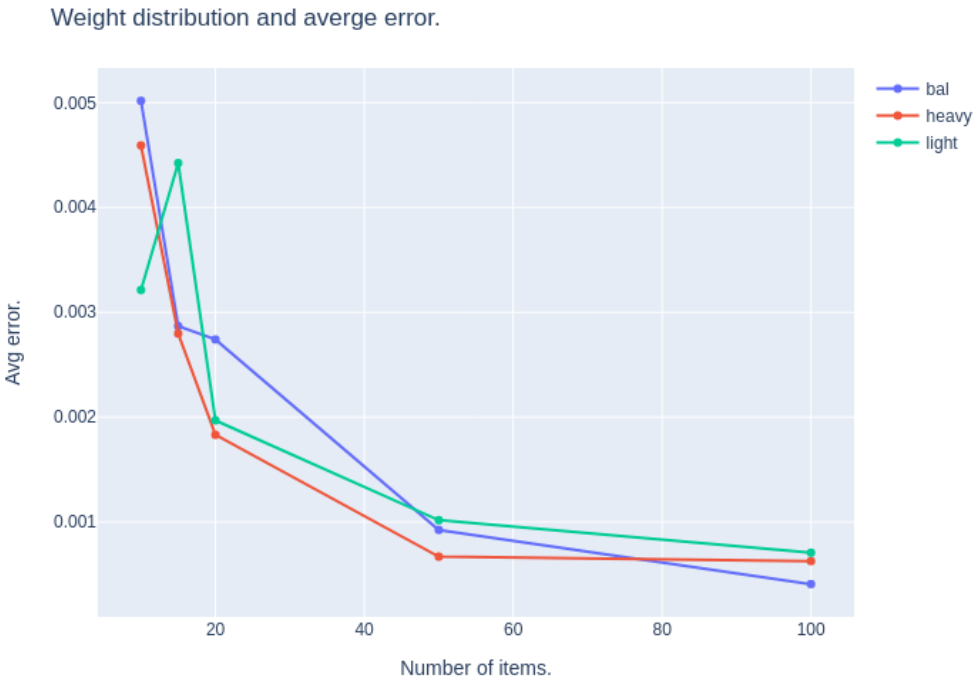


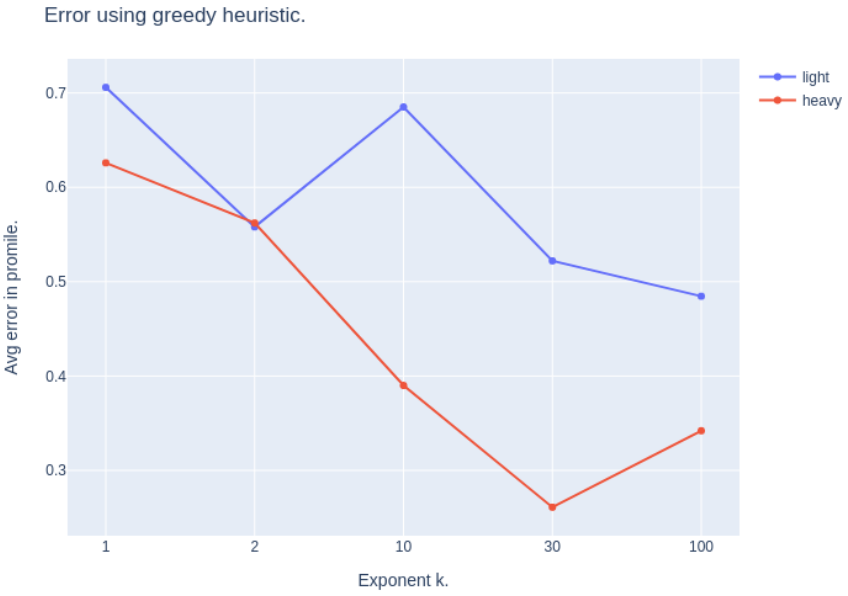
Na grafe je vidno, že v inštanciách s koreláciou, resp. silnou koreláciu trvá výpočet dlhšie, resp. oveľa dlhšie (osa Y je logaritmická).
Tiež môžeme pozorovať závislosť priemernej chyby aproximačných heuristík. S vyššou koreláciou medzi cenou a hmotnosťou je aj priemerná chyba vyššia. Znovu sa to dá odôvodniť vlastnosťou heuristík, ktoré vyberajú predmety podľa ich pomeru hmotnosti a ceny.



Rozloženie váh a vplyv granularity

Rozloženie váh predmetov na prevažne ťažšie alebo ľahšie nemalo vplyv na výpočtový čas optimálnych metód.
Rozloženie váh ovplyvnilo priemernú chybu aproximačných heuristík **SIMPLE** a **REDUX**. Na grafe nižšie je vidno, že vyvážený dataset má niekedy vyššie, inokedy nižšie priemerné chyby oproti nevyváženým datasetom. Ale medzi datasetom, kde sú v prevahe ťažké predmety je (s výnimkou prvého datasetu) vidno nižšia chyba ako pri datasete s prevahou ľahkých predmetov.





Zvyšovaním exponentu sa udržuje vlastnosť nižšej chyby pre datasety s ťažkými predmetmi. Je dôležité si všimnúť veľmi malú chybu, ktorá je v desatinách promile, takže je náročné merať závislosti. Datasety mali 100 predmetov. Túto závislosť by som pripísal rozloženiu predmetov na ľahké a ťažké a nie vplyvu granularity

Súhrn

Brute force (**BF)**:

Algoritmus je robustný voči permutáciám. Pomer kapacity batohu k sumárnej hmotnosti všetkých vecí ovplyvňuje výpočtový čas keď sa pomer blíži k jednej. Je to z dôvodu, že algoritmus musí navštíviť takmer všetky konfigurácie aby naplnil kapacitu.

Branch & bound (**BNB)**:

Algoritmus je robustný voči permutáciám. Z rovnakého dôvodu ako **BF** je algoritmus ovplyvnený pomerom kapacity a sumárnej ceny.

Price decomposition (**PD)**:

Algoritmus je robustný voči permutáciám. Experimenty potvrdili očakávanú závisloť výpočtového času na maximálnej cene predmetov.

Simple greedy algorithm (**SIMPLE)**, Redux greedy algorithm (**REDUX)**:

Algoritmus je robustný voči permutáciám. Experimenty ukázali, že výpočtový čas algoritmu je závislý na korelácii ceny a váhy, čo je pre mňa prekvapivé, pretože tieto heuristiky sú veľmi rýchle a ich zložitosť spočíva len v zoradení predmetov (zložitosť povedzme n.logn) a výberu do momentu naplnenia (zložitosť maximálne n). Ak je veľa malých predmetov s vysokou cenou, algoritmus ich bude vyberať viac.
Algoritmus preukázal závislosť chyby a) *na pomere kapacity batohu a sumárnej cene,* b) *na korelácii ceny a váhy,* c) *na rozložení váhy a granularite.*