Řešení problému batohu metodou hrubé síly a jednoduchou heuristikou

Popis problému:

1) Naprogramujte řešení problému batohu hrubou silou (tj. exaktně). Na zkušebních datech pozorujte závislost výpočetního času na n.
2) Naprogramujte řešení problému batohu heuristikou podle poměru.
    a. Pozorujte závislost výpočetního času na n. Grafy jsou vítány (i pro exaktní metodu).
    b. Průměrnou a maximální relativní chybu (tj. zhoršení proti exaktní metodě) v závislosti na n.

Rozbor možných řešení a algoritmus

Principiálně je řešení zadané popisem problém, záleží tedy spíše na implementačních detailech. Pro implementaci jsem zvolil jazyk Python (což nejspíš není nejvhodnější jazyk, ale je mi aktuálně nejbližší).

1) První část úkolu poskytuje největší možnost kreativity ve volbě generování kombinací, které jsou potřeba pro projití všech validních řešení problému batohu. Dva základní způsoby jsou vnořené for cykly a rekurze. Já si vybral rekurzi, protože se snad bude snáze přepisovat do dalších, více optimalizovaných řešení. Algoritmus tedy rekurzivně generuje množinu možných řešení a ty validní přidává do pole výsledků. V dalším kroku už se následně jenom vybere nejdražší řešení a řešení je na světě.
2) Řešení pomocí heuristiky podle poměru cena/výkon je přímočaré: Každé dvojici cena a váha je spočten koeficient výhodnosti (cena/váha). Následně jsou dvojice sestupně setříděny a do výsledku postupně zařazovány ty s nejlepším koeficientem. Pokud se nějaký předmět do batohu už nevejde, jednoduše se přeskočí a zkouší se další, dokud nejsou vyčerpány všechny.

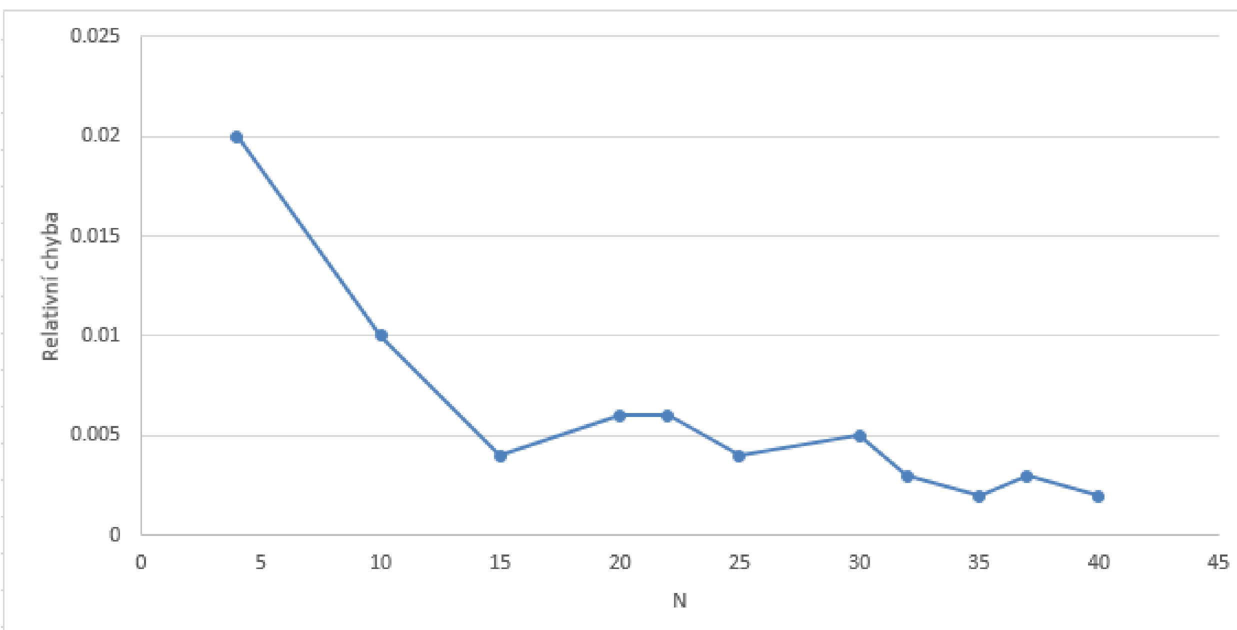
Rozbor výsledků:

Jak ukazuje první tabulka, řešení pomocí heuristiky je velmi rychlé, ve všech prováděných testech se vešlo pod jednu vteřinu. Rozdíly mezi jednotlivými měřeními tedy na konfiguraci s procesorem i5-2410M, Windows 8 a prostředím Python 2.7 nemá cenu dále zkoumat z důvodu jejich malých absolutních hodnot. Rozlišovací schopnost měření by sice šla dále navyšovat, ale to není předmětem tohoto úkolu.

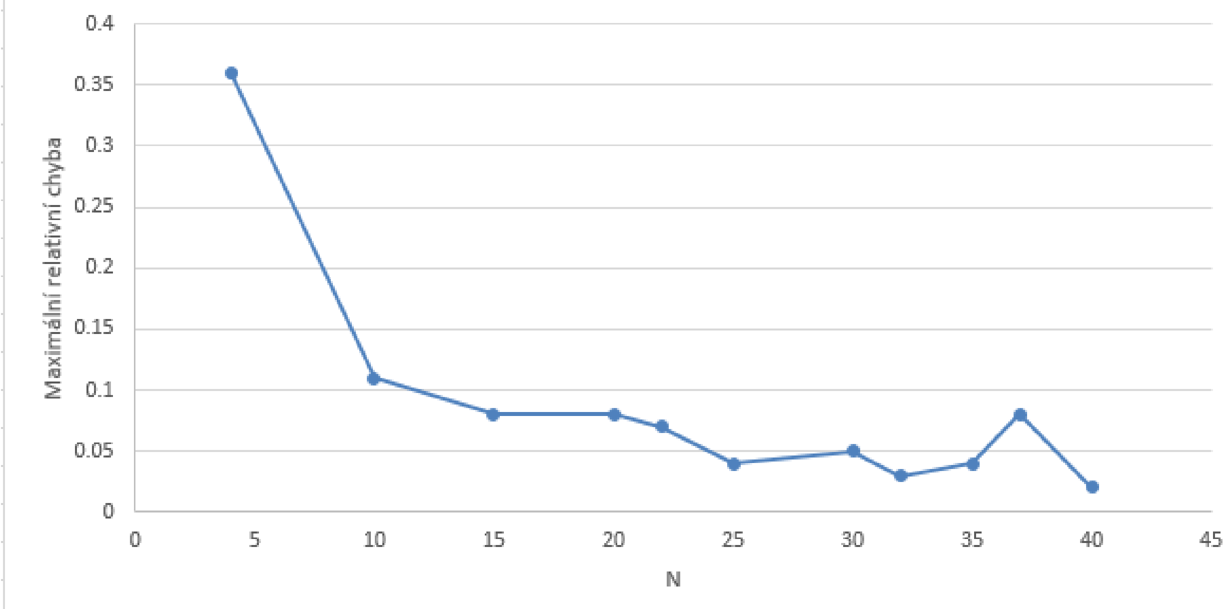
Přesná metoda je kvůli svojí asymptotické složitosti úplně jinde. Již při N=10 a 15 jsou patrné dramatické rozdíly v čase řešení, u N=20+ jsem se rozhodl měření ukončit, protože by na dané konfiguraci trvalo nesmyslně dlouho času a nic nového by nepřineslo. Řešení by určitě šlo optimalizovat, což by možná bylo vhodné pro naměření více hodnost, ale v zadání se jasně hovoří o brute force metodě bez jakékoli optimalizace.

Jak naznačuje poslední tabulka, řešení pomocí heuristiky řešení pouze „odhaduje“, tudíž výsledná konfigurace batohu nepřináší vždy tu nejlepší kombinaci, nicméně řešení s relativně malou relativní chybou je vypočítáno řádově rychleji.

N	presna metoda (s)	heuristika (s)
4	0.1	<0.001
10	3	<0.001
15	120	<0.001
20	n/a	<0.001
22	n/a	<0.001
25	n/a	<0.001
30	n/a	<0.001
32	n/a	<0.001
35	n/a	<0.001
37	n/a	<0.001
40	n/a	0.001



Graf relativní chyby



Graf maximální relativní chyby

N	Relativni chyba	Maximalni rel chyba
4	0.02	0.36
10	0.01	0.11
15	0.004	0.08
20	0.006	0.08
22	0.006	0.07
25	0.004	0.04
30	0.005	0.05
32	0.003	0.03
35	0.002	0.04
37	0.003	0.08
40	0.002	0.02

Závěr

Cvičení v praxi ověřilo, že při obětování absolutní přesnosti se k řešení můžeme dobrat řádově rychleji. Záleží tak na požadavcích, jaké na výsledek máme. V tomto případě vyšla relativní chyba vždy <= 2%, což by pro mnoho aplikací z reálného světa určitě stačilo. Maximální naměřená chyba pro jednotlivé datasety se pohybovala od 2% do 36%. Co se týče času řešení, řešení pomocí heuristiky je velmi rychlé, při N=40 byla naměřena tisícina vteřiny, při menších N byly naměřené časy nižší. U exaktní minuty N=15 trvalo dvě minuty, pro větší N už měření neprobíhalo z důvodu vysoké časové náročnosti.