NI-KOP, 1. úkol

Jan Bittner (bittnja3)

Zadání úlohy

Úkolem je vytvořit program, který řeší rozhodovací verzi problému batohu hrubou silou (**BF**) a pomocí metody větví a hranic (**B&B**).

Spuštění programu

Program lze zkompilovat pomocí `make compile` a následně spustit pomocí `./program.out <method> <file>`, tj. např. `./program.out bnb data/NR/NR4_inst.dat`.

1. `metodu` – `bf` pro metodu hrubou silou, `bnb` pro metodu větví a hranic
2. `file` – soubor nebo soubory ke zpracování

Použité prostředky

Programovací jazyky a software

Úloha byla řešena v jazyce C++ na operačním systému Windows 10.

Měření bylo spuštěno z **Bashe** prostřednictvím prostředí WSL 2 (**Windows Subsystem for Linux 2**), které využívá **Ubuntu 18.04.01 LTS**, nebylo tedy pro spuštění použito IDE.

Na zachycení aktuálního času bylo využito `std::chrono::high_resolution_clock::now()`.

Konfigurace testovacího stroje

Testování bylo provedeno na **ASUS F555UB-DM035T**. Stroj obsahuje CPU **Intel® Core™ i5-6200 @ 2.30Ghz** a RAM **DDR3 SODIMM 8.00 GB**.

Rozbor možných variant řešení

Podle zadání mělo být implementováno řešení hrubou silou a pomocí metody **B&B**.

Řešení šlo provést jak iterativním přístupem, tak pomocí rekurze. Pro výhody rekurzivního řešení byly obě mětody implementovány právě rekurzí.

U metody **B&B** byla implementována pravidla ořezu dle zadání.

Předpokladem je, že metoda **B&B** díky svému ořezávání bude efektivnější. Zároveň je však předpoklad, že bude asymptoticky stejné s **BF** metodou.

Rámcový popis postupu řešení

Obě metody jsou impementovány rekurzivní funkcí, která mění své chování podle zvolené metody. Metody jsou implementovány pomocí strategy patternu, a tedy se rekurzivní funce přizpůsobuje zvolené metodě bez nutnosti ověřování, která metoda je zrovna použita.

Cílem je řešit rozhodovací problém batodu, a tedy při prvním nalezení vyhovující konfigurace se rekurzivní funkce pro obě metody ukončí.

Je měřen čas a počet navštívených nodů. Obě hodnoty jsou průměrovány.

Popis kostry algoritmu

Řešení se opírá o nápad implementovat zadaný problém jako průchod grafem, kde z každého nodu (kromě listů) vychází 2 potomci popisující, zda-li se daná věc do batohu přidá, nebo nepřidá. V každém momentě algoritmus ví kolik aktuální batoh váží, jakou má cenu a jakou cenu ještě může do batohu přidat.

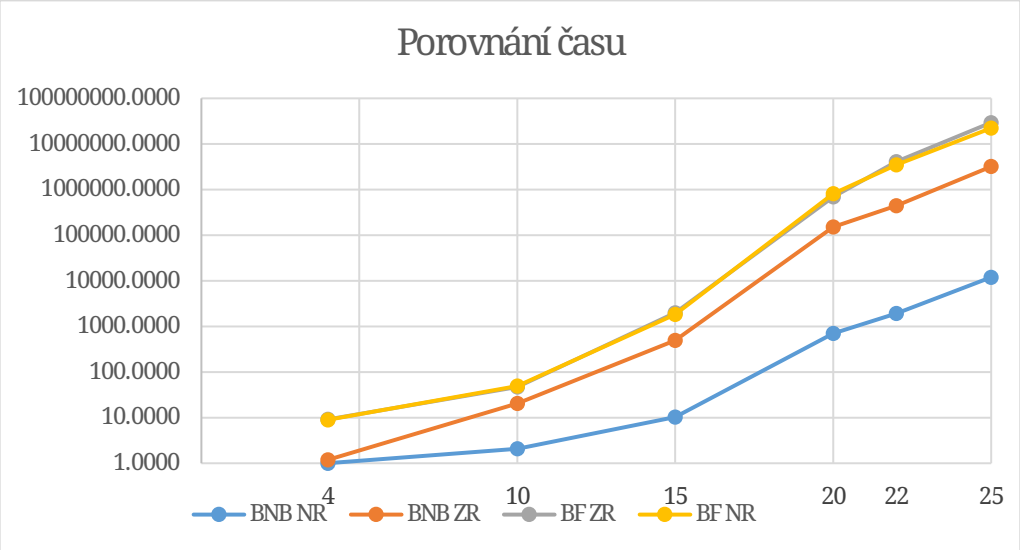
U řešení hrubou silou se prochází všechny možné možnosti.

U metody **B&B** jsou navíc na začátku rekurzivní funkce odřezány ty větve grafu, které nemohou vyústit v úspěšnou konfiguraci (nedosatečná kapacita nebo cena se nedá uspokojit).

Naměřené výsledky

Dle předpokladů se podle měření se ukázalo, že metoda **B&B** je efektivnější pro dobrá data a lehce efektivnější pro špatná data. Metoda hrubé síly i metoda **B&B** nebyla naměřena pro instance s 27, 30, 32, 35, 37 a 40 prvky kvůli nedostatečnému výkonu testovacího stroje.

Naměřené časy a počty nodů ukazují následující grafy s logaritmickým měřítkem. První graf znázorňuje závislost času (v mikrosekundách) na počtu věcí v batohu:



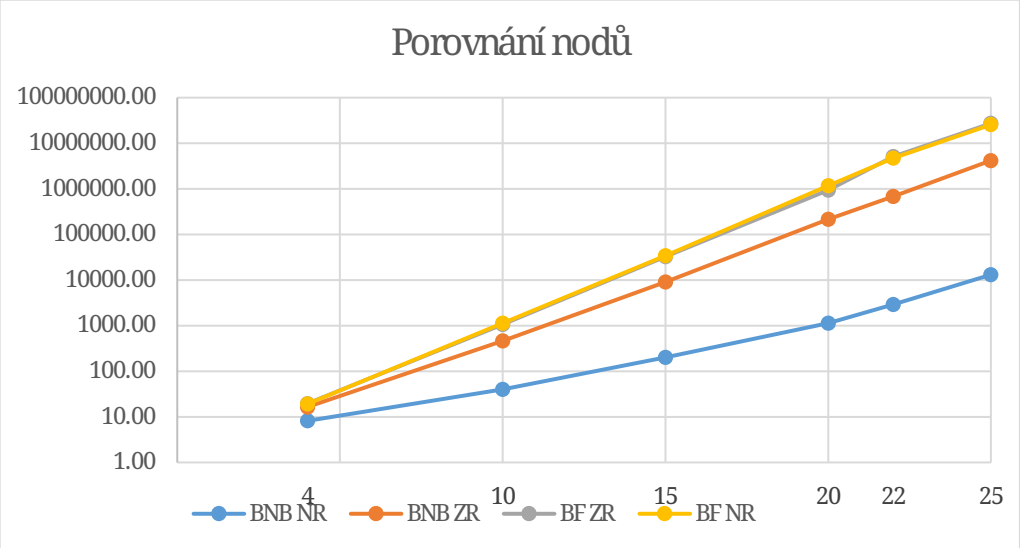
Přehled naměřených dat lze vidět i v tabulce:

	BF NR	BF ZR	BNB NR	BNB ZR
4	8.9568	9.1413	1.0031	1.1852
10	49.2447	47.3357	2.0680	20.3315
15	1822.5574	1975.0213	10.3024	496.7516
20	810784.0616	689940.1320	702.0960	151075.2740
22	3500385.0130	4085678.4555	1923.1958	445346.0930
25	22265312.3666	29423837.7800	11962.9706	3194298.1780

Přehled maximálních časů:

	BF NR MAX	BF ZR MAX	BNB NR MAX	BNB ZR MAX
4	36.3000	53.1553	1.2338	9.5500
10	842.1401	123.0000	2.7504	68.3909
15	7165.5903	8835.2112	15.5566	2561.5033
20	2805619.9000	819110.1930	1692.5400	367947.3200
22	5520834.4010	6733313.2801	13955.8500	605834.3029
25	52591190.2013	41795933.5100	18266.2500	5637184.9031

Druhý graf znázorňuje závislost nodů na počtu věcí v batohu:

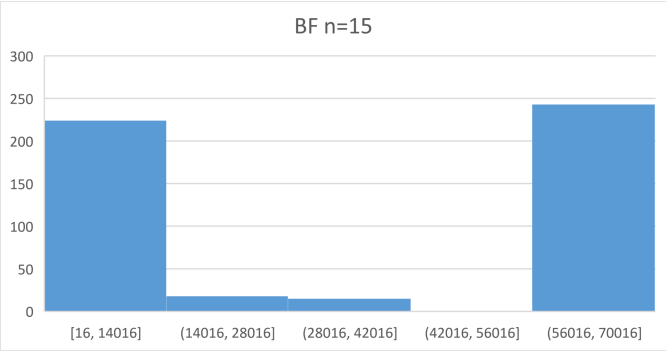


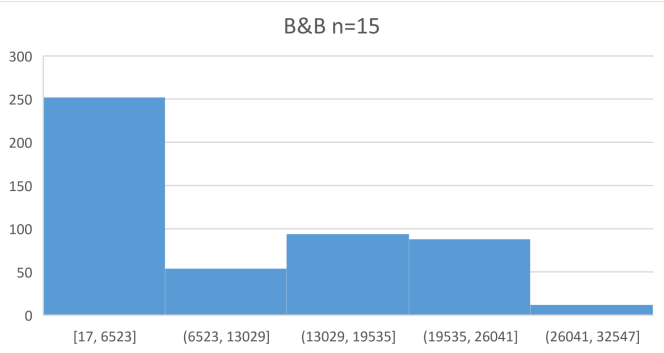
Přehled naměřených dat lze vidět i v tabulce:

	BF NR	BF ZR	BNB NR	BNB ZR
4	19.14	19.61	8.17	16.46
10	1129.94	1062.27	39.90	464.79
15	34164.18	32791.25	200.49	9075.87
20	1173016.43	934556.12	1130.11	214539.65

	BF NR	BF ZR	BNB NR	BNB ZR
22	4704648.14	5141662.45	2905.04	679568.88
25	25512688.80	27236501.30	13085.69	4177891.80

Z grafu je patrné, že počty nodů rostou stejně jako u porovnání časů. Poslední graf zobrazuje histogram metod (na NR datech) pro instance s 15 prvky:





Z histogramu lze vyčíst, že průběh metody **BF** se dostane v cca polovině případů do průchodu téměř celého grafu. Oproti tomu metoda **B&B** dle histogramu ořeže velkou část grafu a díky tomu, že metoda nezkoumá špatné větve, nalézá řešení velmi rychle.

Závěr

Podle zadání byly implementovány metody řešení hrubou silou a **B&B**. Obě metody byly řešeny rekurzivně.

Byly naměřeny jak časy, tak i počty nodů, pro porovnání řešení metod hrubou sílu a **B&B**. Z měření je zjevné, že metoda **B&B** je rychlejší, avšak na špatných datech může dosahovat až stejného počtu průchodu nodů jako metoda hrubé síly.

Z dat jde také vidět, že metoda hrubé síly hloupě prochází a testuje většinu konfigurací, proto má dle předpokladu horší čas zpracování. Díky tomu, že implementace metody **BF** ukončí algoritmus ihned poté, kdy je nalezena vyhovující konfigurace, je tato implementace rychlejší než naivnější implementace, která by prohledala vždy všechny možnosti.

I přes to, že metoda **B&B** je nepochybně efektivnější, podle naměřených dat a sestrojených grafů lze usoudit, že obě metody rostou exponenciálně rychle, a to jak časově, tak i počtem navštívených nodů, což je ovšem předpokládaný stav, jelikož počet navštívených nodů koreluje s vykonaným časem.