Porovnanie exekučnej efektívnosti programovacích jazykov Python a Java pri spracovaní dát

Comparison of the code execution efficiency of programming languages Python and Java used in data processing

Pavol Sojka[1]

Abstrakt
Článok sa zameriava na efektívnosť vykonávania operácií v programovacích jazykoch Python a Java z hľadiska časovej náročnosti. Oba jazyky sa používajú v bežnom programovaní aj v dátovej vede. Dátová veda sa v súčasnosti rozrástla do veľkého rozsahu a tento článok by mal poskytnúť lepší prehľad o tom, ktorý z vybraných jazykov má vyšší výkon pri bežných činnostiach v základnom spracovaní dátových setov. Pripravili sme experiment a v praxi bežne používaný scenár v oboch jazykoch, ktorý zahŕňa prípravu dát, ich spracovanie a výpočet a následnú interpretáciu vrátených výsledkov.

Kľúčové slová
Python, Java, množiny údajov, spracovanie údajov

Abstract
The article focuses on the efficiency of execution of the programming languages computations in Python and Java in terms of time consumption. Both languages are used in common programming and also in data science. Data science has grown to a large scale nowadays and this article should provide a better overview of which of the selected languages has better performance in terms of basic data handling. We have prepared an experiment and a scenario commonly used in practice in both languages, which includes data preparation, data processing and calculation, and subsequent interpretation of the returned results.

Key words
Python, Java, datasets, data processing

JEL classification
C8

1 Úvod

V súčasnosti je veľký dopyt po dátových analytikoch a programátoroch, pretože dátová veda a priemysel v posledných rokoch výrazne narástli. S týmto trendom sa objavujú otázky, ako napríklad, ktorý programovací jazyk by sme mali použiť na splnenie požiadaviek na

[1] Ekonomická univerzita v Bratislave, Fakulta hospodárskej informatiky, Katedra aplikovanej informatiky, Dolnozemská cesta 1, 852 35 Bratislava, Slovakia, pavol.sojka@euba.sk.

spracovanie dát. Mnohé odpovede možno nájsť na internete alebo v literatúre. Môžeme zvážiť tieto odporúčania na základe rozšírenosti jazyka, krivky učenia, dostupnosti zdrojov poznatkov a podobne. V našom článku sme si vybrali jedno kritérium na porovnanie a toto kritérium je založené striktne na meraní efektívnosti na základe hardvérových parametrov s prednastaveným softvérovým scenárom.

2 Príprava prostredia

Ako primárny operačný systém, na ktorom budú vykonávané experimenty, sme si vybrali Linux Ubuntu 22.04. Naše prostredie bolo umiestnené na platforme Google Cloud na virtuálnom stroji s hardvérovými parametrami začínajúcimi na ôsmich CPU (centrálna procesorová jednotka, Intel Xeon @ 2,20 GHz, family 6, model 79) a šestnástich gigabajtoch RAM (operačná pamäť). Počet CPU sa postupne menil a merali sme dopad na efektívnosť vykonávania kódu z hľadiska času. Môže sa zdať, že výsledky sú ľahko predvídateľné, ale naším cieľom nebolo len dokázať, že zníženie počtu CPU má negatívny vplyv na výkon, ale tiež presne merať hodnoty vo zvolenom prostredí a skúmať, či tento vplyv je rovnaký alebo podobný u oboch programovacích jazykov.

Po úspešnom nasadení operačného systému sme nainštalovali jazyk Python (verzia Python 3.10.12) a jazyk Java (verzia OpenJDK 21.0.6) z repozitárov operačného systému. Spolu s Pythonom sme nainštalovali aj knižnicu Pandas. Táto knižnica je navrhnutá na efektívnu prácu s dátovými množinami uloženými vo formátoch CSV, XLSX a ďalších. Na úspešnú inštaláciu kompilátora Java je potrebné nainštalovať Java framework a jeho vývojové nástroje.

Naším cieľom bolo otestovať rýchlosť nášho naprogramovaného kódu v týchto krokoch:

- Otvoriť súbor CSV
- Načítať stĺpce dátovej sady do pamäte
- Vypočítať súčet celého stĺpca „Tržby“ a „Zisk“
- Vytlačiť názov súboru a výsledok na obrazovku
- Vytlačiť výsledný čas vykonávania a ukončiť aplikáciu

Rozhodli sme sa generovať syntetické dátové sady vo formáte CSV (hodnoty oddelené čiarkami). Na tento účel sme použili aplikáciu naprogramovanú v jazyku Java pôvodne pre iné experimenty vykonané v minulosti. Vygenerovali sme 500 testovacích súborov s údajmi o 20 riadkoch a 15 stĺpcoch. Náš program môže generovať dátové sady so stovkami tisíc riadkov, ale pre naše experimenty by malo byť 500 súborov x 20 riadkov dostatočných. Vygenerované súbory boli skopírované do adresárov Python a Java.

3 Výsledky

Každé meranie sme vykonali trikrát za sebou v oboch jazykoch a zaznamenané časy sme uložili do tabuľky. Po troch cykloch sme vypli server a postupne menili počet používaných CPU z ôsmich na dve. Pozorovali sme malé zmeny v časoch vykonávania kódu, ale vyskytli sa len menšie rozdiely, až pri dvoch CPU bola zmena času významná. Ako vidno z tabuliek nižšie, môžeme povedať, že program v Jave bol efektívnejší ako v jazyku Python, pretože Java je kompilovaná do bajtkódu spusteného na Java Virtual Machine, ktorý je vždy rýchlejší ako interpretovaný jazyk – Python. Oba programy bežali v konzolovom okne Linux bash. Prvý meraný cyklus v oboch jazykoch bol vždy pomalší ako ďalšie dva. Tento efekt možno vysvetliť prednačítaním systémových knižníc do pamäti po prvom behu a preto sme merali časy až po

prvom spustení, aby sme tento efekt eliminovali. Časy v tabuľkách sú už matematicky zaokrúhlené kvôli prehľadnosti, pôvodne boli počítané rozdiely pri vyššej presnosti.

Tab. 1: Časy vykonávania (8 CPU)

	CPU: 8, Pamäť: 16 GB		
	Python (s)	**Java (s)**	**Rozdiel (python/Java)**
1. meranie	0,65	0,15	4,28
2. meranie	0,63	0,17	3,66
3. meranie	0,62	0,16	3,91
priemer	**0,63**	**0,16**	3,94

Zdroj: vlastné spracovanie

Tab. 2: Časy vykonávania (6 CPU)

	CPU: 6, Pamäť: 16 GB		
	Python (s)	**Java (s)**	**Rozdiel**
1. meranie	0,63	0,16	3,97
2. meranie	0,68	0,15	4,50
3. meranie	0,63	0,16	3,91
priemer	**0,65**	**0,16**	4,12

Zdroj: vlastné spracovanie

Tab. 3: Časy vykonávania (4 CPU)

	CPU: 4, Pamäť: 16 GB		
	Python (s)	**Java (s)**	**Rozdiel**
1. meranie	0,68	0,18	3,72
2. meranie	0,68	0,21	3,28
3. meranie	0,67	0,17	3,88
priemer	**0,68**	**0,19**	3,61

Zdroj: vlastné spracovanie

Tab. 4: Časy vykonávania (2 CPU)

	CPU: 2, Pamäť: 16 GB		
	Python (s)	**Java (s)**	**Rozdiel**
1. meranie	0,70	0,33	2,09
2. meranie	0,71	0,28	2,56
3. meranie	0,72	0,30	2,43
priemer	**0,71**	**0,30**	2,34

Zdroj: vlastné spracovanie

Ako môžeme vidieť z údajov v tabuľkách, jazyk Java je vždy efektívnejší ako jazyk Python. Efektívnosť oboch jazykov mierne klesá vždy, keď sa zníži počet jadier (CPU). Tabuľky 1–3 ukazujú, že efektívnosť Pythonu je podobná a v jazyku Java rýchlo klesá v poslednom scenári iba s dvoma CPU. Ako bolo spomenuté vyššie, prvé meranie je pomalšie ako ďalšie a tento efekt ovplyvňujú interné procesy riadené operačným systémom. Rozdiely v jazykoch Python a Java sú na prvý pohľad zrejmé. V porovnaní s Pythonom a knižnicou Pandas bude Java vždy efektívnejšia, pretože Java je staticky typovaný programovací jazyk a kompilátor pozná každú premennú a výraz pri behu (Adhikari, 2024). Na druhej strane Python je dynamicky typovaný programovací jazyk a preto sa typ premennej odvodzuje na základe jej použitia (Kohli, 2021). Ďalší rozdiel je, že Java je kompilovaná do bajtkódu, ktorý sa následne vykonáva na Java Virtual Machine (Adhikari, 2024). Python s jeho najbežnejším použitím je interpretovaný vždy zo zdrojového kódu (interpreter následne už volá skompilované knižnice), čo je jasne viditeľné z časov vykonávania (nižší čas spracovania = vyššia efektívnosť aplikácie pri ostatných nezmenených podmienkach). Na optimalizáciu procesov nášho konceptu by sme mali zahrnúť špecifické algoritmy vhodné pre daný typ úlohy, hlbší vhľad do konkrétneho programovacieho jazyka, výber vhodnejších programovacích jazykov pre strojové učenie alebo výber výkonnejších knižníc jazyka Python použitých pri strojovom učení (Esposito & Esposito, 2020). Náš článok má za cieľ základné predstavenie často používaných jazykov strojového učenia, ktoré zahŕňajú jazyky ako Java, Python a R (Kumar, 2024). Často používaným nástrojom v praxi je tiež rámec Spark (Mehrotra & Grade, 2019), ktorý využíva efektívnosť Java a knižnice Spark používaných na manipuláciu s obrovským množstvom dát, ako aj platformy Hadoop (Nordeen, 2020) používané na udržiavanie takýchto veľkých dátových sád. Nasledujú časti kódu, ktoré sme použili v našom testovacom prostredí.

Obr. 1: Ukážka kódu v jazyku Python

```
for f in csv_files:
        df = pd.read_csv
(f, sep=';')
        print('Location:', f)
        print('File Name:', f.split("\\")[-1])
                print(f)
```

```
print('Total profit:', df['profit'].sum())
print('Total sales:', df['sale'].sum())
print("------------------------------")
```

Zdroj: vlastné spracovanie

Obr. 2: Ukážka kódu v jazyku Java

```
public static void compute(String filename) {
    try (BufferedReader bufferedSource = new BufferedReader(new FileReader(filename))) {
        int start = 1;
        double hodnota = 0.0;
        double hodnota2 = 0.0;
        double sum = 0;
        double sum2 = 0;
        String line;

        while ((line = bufferedSource.readLine()) != null) {
            String[] cols = line.split(";");
            for (int i = 0; i < cols.length; i++) {
                cols[i] = cols[i].trim();
            }

            if (start != 1) {
                hodnota = Double.parseDouble(cols[4]);
                sum += hodnota;
                hodnota2 = Double.parseDouble(cols[5]);
                sum2 += hodnota2;
            }
            start = 0;
        }
        System.out.println(filename);
```

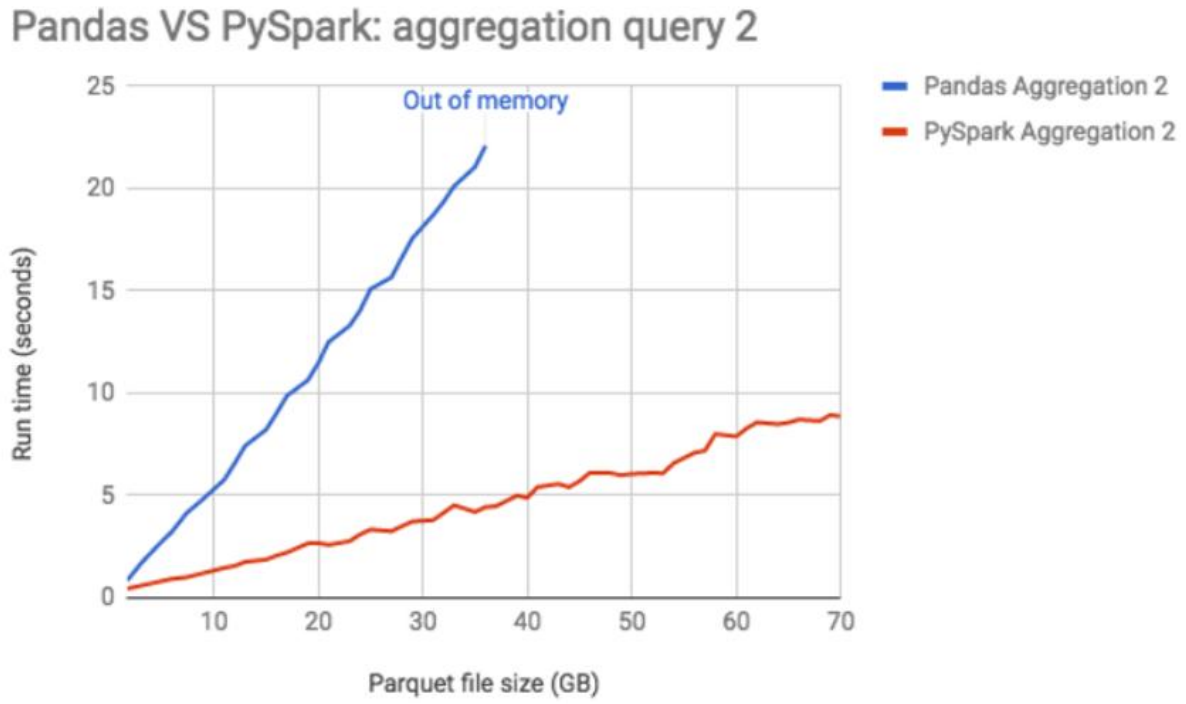
```
System.out.println("Celkove trzby:" + sum);

System.out.println("Celkovy zisk:" + sum2);

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

Zdroj: vlastné spracovanie

4 Diskusia

Náš prístup by mohol byť optimalizovaný pre lepšiu výkonnosť pomocou mnohých nástrojov a knižníc, ako sú Hadoop, Spark a špecializované rámce používané oboma jazykmi. Môžu sa objaviť ďalšie otázky, aký jazyk použiť, keďže mnohé iné jazyky (napr. Mojo, Julia a iné) zaznamenali v posledných rokoch veľký rozvoj. Python sa môže používať aj s rámcom Apache Spark. V tomto rámci je Python optimalizovaný pre špecializované knižnice, ktoré sú oveľa efektívnejšie ako samotný Python (Aven, 2018). Ako ukazuje nasledujúci graf, čas behu niekoľkých dopytov v PySpark (jazyk Python bežiaci spolu s Apache Spark) v porovnaní s rovnakými dotazmi v knižnici Pandas. Je zrejmé, že (Py)Spark skutočne umožňuje pracovať s väčšími údajmi efektívnejším spôsobom (Element 61, 2020). V grafe je znázornená (rovnako ako v našom článku) celková efektívnosť funkcie „sum“ v knižnici Python Pandas v porovnaní s knižnicou PySpark.

Obr. 3: Agregačný dopyt Pandas vs PySpark



Zdroj: Element (Element 61, 2020)

5 Záver

V našom článku sme sa zamerali na porovnanie dvoch jazykov aplikovaných na jednoduchú výpočtovú úlohu, aby sme ukázali efektívnosť použitých jazykov. Na základe našich výsledkov sme preukázali, ktorý z vybraných jazykov je z hľadiska výkonnosti lepší.

Existujú aj ďalšie aspekty, ktoré by sa mohli zohľadniť, a to je napríklad rozšírenosť konkrétneho jazyka a krivka učenia. Vzhľadom na tieto možnosti je Python víťazom z dôvodu jeho rozšíreného používania a známosti. Jazyk Java je pre nováčikov trochu komplikovaný na rýchle pochopenie. Na druhej strane Python je celosvetovo uznávaný jazyk s veľkou používateľskou základňou a rozsiahlou znalostnou bázou na riešenie problémov. Naučiť sa tento jazyk nie je pre nováčikov také náročné ako jazyk Java. Z našich úryvkov kódu možno pozorovať, že jazyk Python je ľahko čitateľný a jednoduchý. Tieto výhody a nevýhody možno podporiť alebo vyvrátiť kombináciou nástrojov, ako je Apache Spark, ktorý implementuje podporu pre Javu aj Python. Špeciálne pripravené knižnice by mohli zvýšiť efektivitu jazyka Python, ale stále nebude taký výkonný ako čisto kompilované jazyky. Na záver pridávame obrázok najpoužívanejších jazykov v dátovej vede do roku 2024. Index popularity programovacieho jazyka bol vytvorený na základe analýzy frekvencie vyhľadávania príručiek a tutoriálov pre jednotlivé jazyky vo vyhľadávacej službe Google (PYPL, 2024).

Obr. 4: Programovacie jazyky používané v dátovej vede

Worldwide, Jul 2024 :

Rank	Change	Language	Share	1-year trend
1		Python	29.35 %	+1.5 %
2		Java	15.6 %	-0.2 %
3		JavaScript	8.49 %	-0.8 %
4		C#	6.9 %	+0.1 %
5		C/C++	6.37 %	-0.1 %
6	↑	R	4.73 %	+0.3 %
7	↓	PHP	4.49 %	-0.5 %
8		TypeScript	2.96 %	-0.1 %
9		Swift	2.78 %	+0.2 %
10	↑	Rust	2.55 %	+0.4 %

Zdroj: PYPL (PYPL, 2024)

Literatúra

1. Adhikari, S. (2024). Exploring Scientific Computing with Java: A Practical Guide for Logic and Application Building (English Edition). India: Bpb Publications.
2. Aven, J. (2018). Data Analytics with Spark Using Python. Pearson Education.
3. Element 61. (2020). How to use Python and Pandas while embracing the power of Spark. Element 61. https://www.element61.be/en/resource/how-use-python-and-pandas-while-embracing-power-spark
4. Esposito, D., Esposito, F. (2020). Introducing Machine Learning. Pearson Education.
5. Kohli, M. (2021). Basic Core Python Programming. BPB Publications.
6. Kumar, A. (2024). Ultimate Java for Data Analytics and Machine Learning: Unlock Java's Ecosystem for Data Analysis and Machine Learning Using WEKA, JavaML, JFreeChart, and Deeplearning4j (English Edition). India: Orange Education Pvt Ltd.
7. Mehrotra, S., Grade, A. (2019). Apache Spark Quick Start Guide. Packt Publishing.

8. Nordeen, A. (2020). Learn Hadoop in 24 Hours. Guru99.
9. PYPL (2024). PYPL PopularitY of Programming Language. PYPL. https://pypl.github.io/PYPL.html