

Faginův Top-K Algoritmus

BI-VWM Semestrální práce

Jan Černý
cernyj87@fit.cvut.cz

Obsah

1	Popis projektu.....	2
2	Způsob řešení.....	2
2.1	Databáze produktů	2
2.2	Zpracování dat	2
3	Implementace	3
3.1	Použité technologie	3
3.2	Rozdělení aplikace.....	3
3.3	Top-K algoritmus.....	3
3.4	Sekvenční průchod (naivní algoritmus).....	3
4	Spuštění aplikace.....	4
5	Ukázka aplikace.....	5
6	Experimenty.....	6
6.1	Počet přístupů do databáze	6
6.2	Doba běhu	7
6.3	Počet parametrů	8

1 Popis projektu

Cílem bylo implementovat Faginův Top-K algoritmus na vyhledání nejlepších K výsledků v databázi produktů. Uživatel si pomocí webového rozhraní zvolí, podle jakých parametrů chce produkty řadit. Výstupem je tabulka seřazených výsledků, změřený čas běhu algoritmu a počet, kolik bylo provedeno požadavků na databázi.

Projekt obsahuje jak Faginův Top-K, tak i naivní algoritmus sekvenčním průchodem. Je tak možné vyzkoušet rozdílné přístupy a porovnat jejich efektivitu.

2 Způsob řešení

2.1 Databáze produktů

Pro zjednodušení práce s daty jsou využívány CSV soubory, není tak potřeba spouštět žádný databázový engine nebo vytvářet SQL dotazy. Na ukázkou jsem si jako produkt vybral počítačové myši, které mají parametry jako název, cenu, váhu, DPI nebo přesnost senzoru.

Ukázková data jsou uložena ve složce `~/data``

- `~/data/mouse.csv`` - Ukázková reálná data (cca 40 položek)
- `~/data/test.random.csv`` - Náhodně generovaná data (cca 1 milion položek)

2.2 Zpracování dat

Data jsou při načtení uložena do paměti RAM a předtříděna podle parametrů produktu. Jsou znormalizována do rozsahu 0–1, aby každý parametr produktu měl při vyhledávání stejnou váhu. Pokud bychom například využili jako agregační funkci sumu a měli parametry cenu (s rozdíly tisíců Kč) a váhu (s rozdíly několika jednotek gramů), ovlivňoval by rozsah cen výsledek více než rozsah vah.

3 Implementace

3.1 Použité technologie

Aplikace je napsaná v programovacím jazyce Python. Ten jsem zvolil kvůli jeho jednoduchosti, a protože bych se s ním chtěl naučit více pracovat.

Pro webové rozhraní používám Pythonní balíček Gradio, se kterým jsem se setkal jako uživatel primárně u různých aplikací využívající strojové učení. Vybral jsem si ho, protože mě zaujal svojí jednoduchostí na programování, bohužel realita byla spíše opak. Dokumentace obsahuje zatím neimplementované nebo nefunkční vlastnosti a příliš materiálů/návodů online není. Příště bych zvolil spíše Flask na vytvoření REST API a frontend tvořil zvlášť například pomocí NuxtJS, se kterým mám zkušenosti.

3.2 Rozdělení aplikace

Hlavním vstupním bodem je soubor `~/main.py`, který spouští webové rozhraní ze souboru `~/src/app.py`, ve kterém je definované UI aplikace.

Prvním krokem je načtení a předzpracování dat, o to se stará `~/src/Database.py`. Nabízí metody na získání předtříděných seznamů parametrů či celou datovou sadu.

Soubor `~/src/Algorithm.py` obsahuje statické metody (`@classmethod`) pro naivní a top-k algoritmy, agregační funkce a další podpůrné funkce.

3.3 Top-K algoritmus

Top-K algoritmus využívá předtříděné seznamy parametrů ze třídy `Database` a množinu (set) produktů. Python pro množiny používá hashování, díky čemuž je zajištěn konstantní $O(1)$ přístup.

Postupně prochází předtříděné seznamy parametrů a doplňuje ostatní hodnoty parametrů produktu, dokud nezíská „k“ produktů se všemi parametry doplněnými. Následně spočítá výsledné hodnocení pomocí agregační funkce a podle něj seřadí výstupní seznam.

3.4 Sekvenční průchod (naivní algoritmus)

Sekvenční průchod prochází celou databází produkt po produktu a každému počítá hodnocení pomocí agregační funkce. Seznam následně pomocí něj seřadí a vrátí prvních „k“ výsledků.

4 Spuštění aplikace

Pro spuštění je potřeba mít nainstalovaný Python.

Aplikace využívá další podpůrné balíčky, proto je vhodné si nejdříve vytvořit virtuální prostředí, například do složky `~/ .venv` příkazem:

```
python -m venv .venv
```

A aktivovat jej

(PowerShell)	<code>.venv\Scripts\Activate.ps1</code>
(CMD)	<code>.venv\Scripts\activate.bat</code>
(Bash)	<code>source .venv/Scripts/activate</code>

Nainstalovat balíčky (závislosti)

```
pip install -r requirements.txt
```

Samotnou aplikaci lze spustit příkazem

```
python main.py
```

Webové rozhraní poté běží na adrese

<http://127.0.0.1:7860/>

5 Ukázka aplikace

BI-VWM Semestral Project

Fagin Top-K Algorithm

Jan Cerný, 2023

Select database

CSV File

Select CSV file to load into database

mouse.csv

Loaded database

mouse.csv

Load database

Select query parameters

Algorithm

Algorithm used for querying top-k

Fagin

Aggregation function

Select aggregation function to use

Sum

K

Number of top-k results to return

10

Fields

Select fields to aggregate

Weight

Accuracy

DPI

Price

Submit

Normalized

Results

Name	Weight	Accuracy	DPI	Price	Aggregation
Logitech G102	85	0.8	8000	40	1.5
Logitech G203	85	0.8	8000	40	1.5
Logitech G305	99	0.8	12000	50	1.4523809523809523
Logitech G502	121	0.8	12000	50	1.4523809523809523
Logitech G502 Hero	121	0.8	16000	50	1.4523809523809523
Logitech G402	108	0.8	4000	50	1.4523809523809523
Logitech G403	87	0.8	12000	50	1.4523809523809523
Logitech G302	87	0.8	4000	50	1.4523809523809523
Logitech G403 Hero	87	0.8	16000	50	1.4523809523809523
Logitech G Pro	83	0.8	12000	50	1.4523809523809523

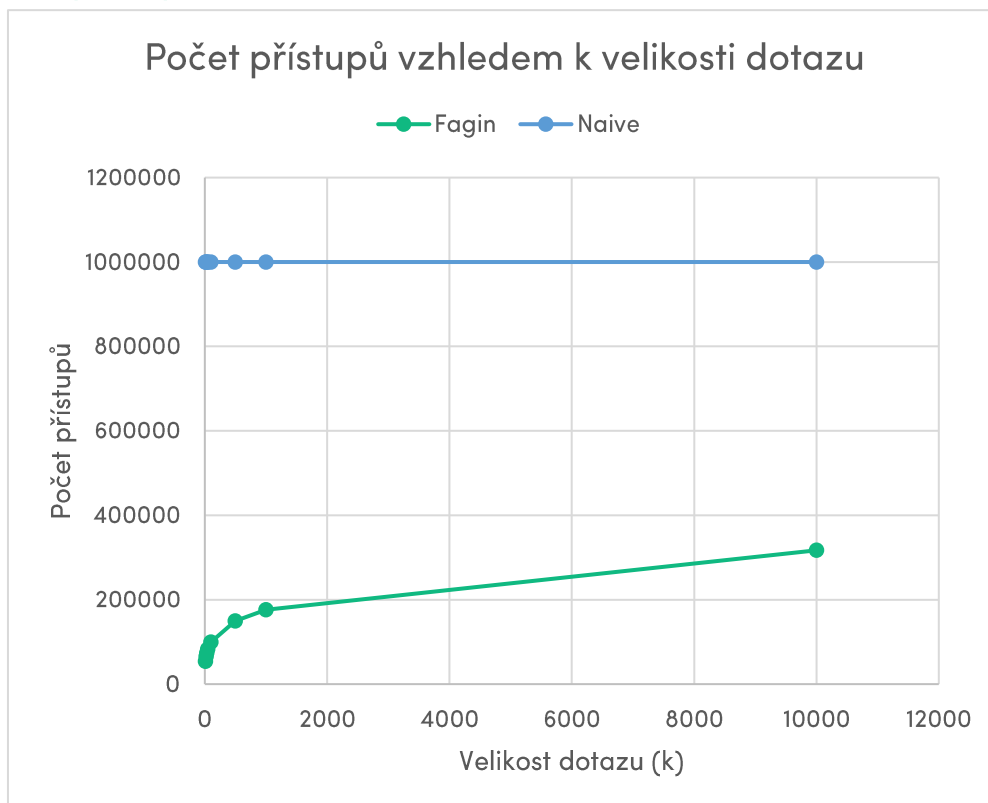
Time spent: 0.0000000000s

Access count: 15

6 Experimenty

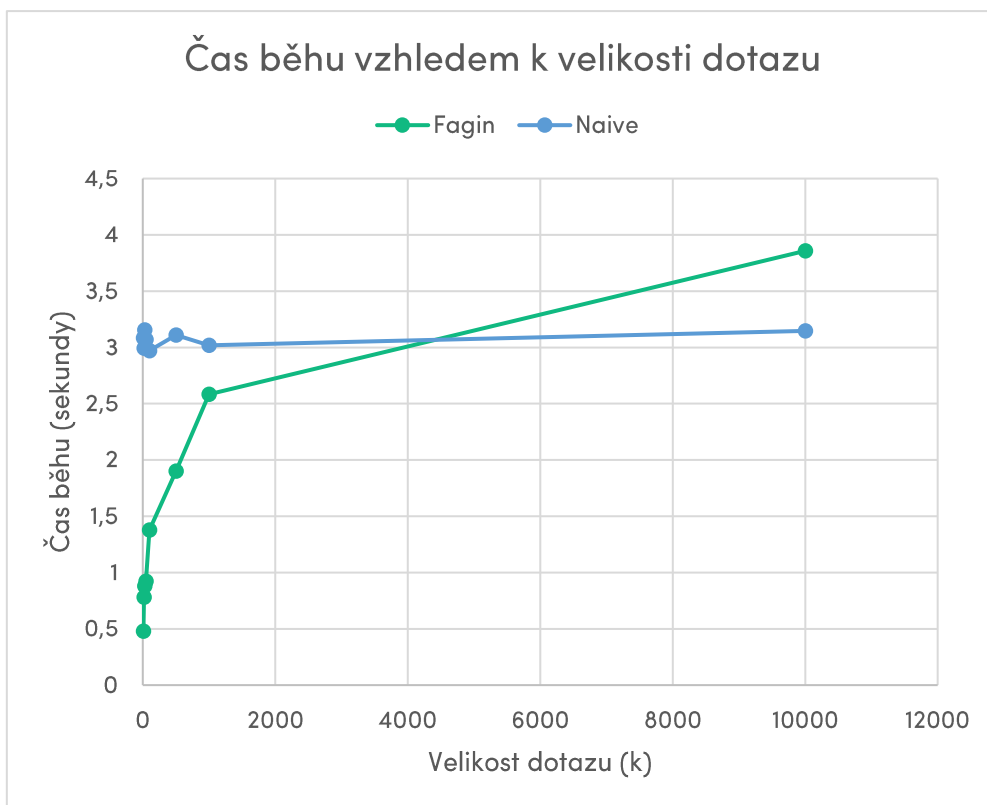
V následujících experimentech se pokusím porovnat Faginův algoritmus oproti sekvenčnímu průchodu. Testy byly prováděny na datové sadě o velikosti 1 milion produktů s vybranými všemi čtyřmi parametry.

6.1 Počet přístupů do databáze



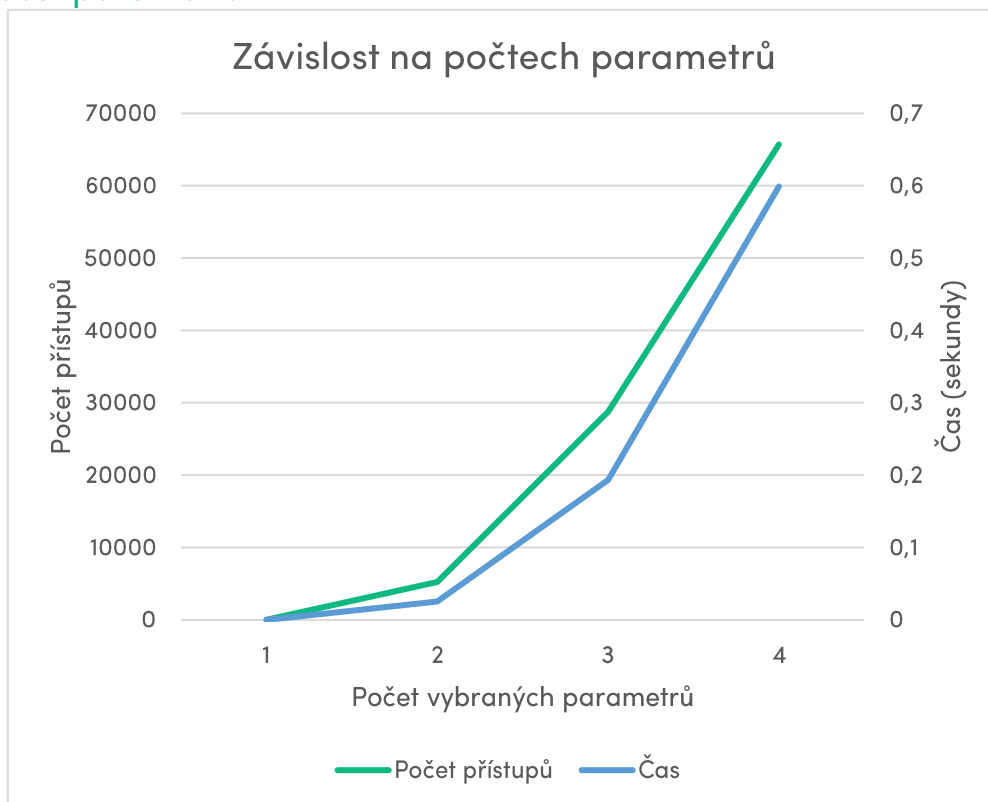
Počet přístupů Faginova algoritmu je závislý na velikosti dotazu (číslo „k“), protože nepotřebuje projít celý dataset, ale skončit, jakmile má všechny parametry pro dostatek produktů. Sekvenční průchod musí vždy projít celý dataset.

6.2 Doba běhu



Ačkoliv Faginův algoritmus vyžaduje nižší počet přístupů do databáze, je výpočetně náročnější dohledávat hodnoty zbývajících parametrů produktu. Od určité větší velikosti „k“ se více vyplatí použít sekvenční průchod. Pro nízké „k“ je ale velmi efektivní.

6.3 Počet parametrů



Na tomto grafu nejsou porovnávány algoritmy mezi sebou, ale porovnává pouze vlastnosti Faginova algoritmu. Zkoušel jsem, jak se bude chovat pro různý počet zvolených parametrů. Na grafu je vidět, že počet přístupů do databáze i čas běhu jsou závislé na tomto výběru, protože čím více parametrů je zvoleno, tím více musí algoritmus dohledávat další hodnoty parametrů produktu.

7 Závěr

Mým cílem bylo implementovat Faginův Top-K algoritmus. O tomto a dalších podobných algoritmech jsem dříve vůbec neslyšel, proto bylo zajímavé je poznat a něco se o nich naučit. Vyzkoušel jsem si práci s knihovnou Gradio, u které jsem byl z DX¹ lehce zklamán, ale nové znalosti mi přijdou velmi přínosné pro tvorbu web UI v Pythonu.

¹ Developer Experience