Profiling a Optimalizace Programu

team_not_found

30. dubna 2025

Shrnutí Profilování

Tento dokument obsahuje analýzu profilování programu pro různé velikosti vstupních dat: 10, 10^3 a 10^6 hodnot. Cílem je identifikovat místa, kde program tráví nejvíce času, a poskytnout doporučení pro optimalizaci kódu.

Výsledky Profilování

Vstupní velikost 10^6:

- **Celkový počet volání funkcí**: 14 000 027
- **Celkový čas**: 4.017 sekund
- Funkce `add` v souboru `calc_lib.py` je volána 3 000 000krát a přispívá k celkovému času 1.848 sekund.
- Funkce `expon` v souboru `calc_lib.py` je volána 1 000 001krát a přispívá 0.718 sekund.
- Další významné příspěvky pocházejí z vestavěných funkcí jako `isinstance` (0.888 sekund) a `uniform` (0.440 sekund).

Vstupní velikost 10^3:

- **Celkový počet volání funkcí**: 14 027
- **Celkový čas**: 0.005 sekund
- Funkce `calculate_stddev` stále zabírá nejvíce času, ale celkový čas je podstatně nižší (0.005 sekund).
- Funkce `add` je volána 3 000krát a funkce `expon` 1 001krát.
- Přítomnost overheadu z vestavěných funkcí je i zde minimální.

Vstupní velikost 10:

- **Celkový počet volání funkcí**: 167
- **Celkový čas**: 0.000 sekund
- Výkon je téměř zanedbatelný, přičemž rozdělení volání funkcí je stejné. Funkce `add` a `expon` jsou volány, ale s malými počty a minimálním dopadem.

Zjištěné Úzká Místa a Doporučení pro Optimalizace

1. celkový čas

Funkce jako taková vypadá poměrně dobře optimalizovaná a nezabírá nejvíce času spíš to vypadá na zpomalování ze strany knihovny a jejích funkcí.

- Použití knihoven, které mohou významně urychlit výpočty místo vlastních knihoven.

2. Funkce `add` (Velký Počet Volání)

Funkce `add` je volána milionkrát při větších vstupech, ale její časová náročnost na jedno volání je nízká. Optimalizace této funkce pravděpodobně nepřinese velké zlepšení, ale je možné:

- Zvážit sloučení operací nebo optimalizaci vnitřní implementace tak, aby se snížil počet nepotřebných výpočtů.

3. Vestavěné Funkce (`isinstance`, `uniform`)

Tyto funkce se vyskytují často, zejména u větších vstupů. I když jejich náklady na čas jsou malé, je možné je optimalizovat takto:

- Minimalizovat jejich používání tam, kde to není nutné.
- Zvážit efektivnější alternativy pro generování náhodných čísel nebo kontrolu typů.

Doporučení pro Další Optimalizaci

- Zaměřte se především na optimalizaci funkcí knihovny kde v takovém programu by byla lepší specifická knihovna, která nepotřebuje jakkoliv kontrolovat data při výpočtech.
- U funkce `add` zkontrolujte, zda lze snížit počet volání nebo sloučit některé operace.

- Zvažte použití jiné knihovny která se specializuje na takovéto mat fce aby se zefektivnilo a z rychlilo počítání jako takové.