Paralelný web crawler*

Peter Kapusta
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xkapustap@stuba.sk

16. decembra 2023

Abstrakt

V nasledujúcom článku sa budeme zaoberať využitím paralelných web crawlerov, ako aj problémom optimalizácie (spolu s metódou, ako odľahčiť dopad web crawlerov na jednotlivé webstránky), postupne uvedieme niekoľko riešení, ktoré sa momentálne využívajú a ich spôsoby komunikácie medzi jednotlivými inštanciami. Na konci preberieme dopad paralelných web crawlerov na spoločnosť.

1 Úvod

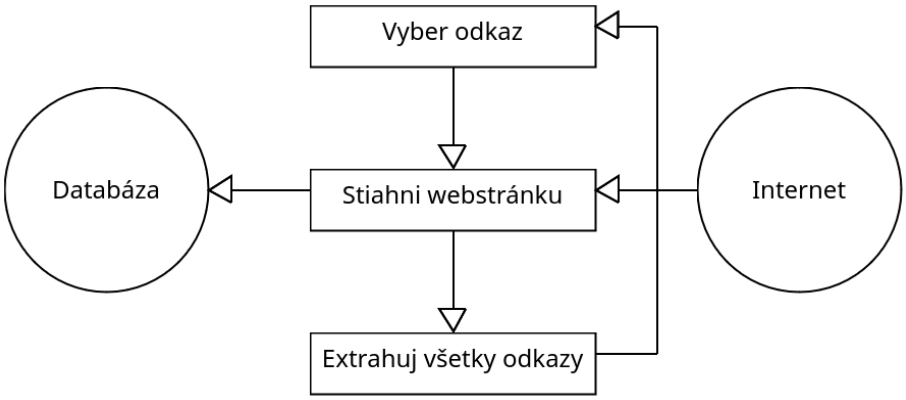
Vo všeobecnosti, web crawler je program, ktorý pomocou začiatočného zoznamu odkazov systematicky skenuje web, objavuje webstránky a následne z nich extrahuje informácie a nové odkazy, ktoré vedú k ďalším webstránkam. Cieľom je teda čo najefektívnejšie a najrýchlejšie objaviť čo najväčšie množstvo informácií, ktoré môže neskôr využiť napríklad prehliadač. Hlavnými ťažkosťami sa pri jednoduchých web crawleroch stávajú prehľadávanie HTML kódu a získavanie využiteľných odkazov, čo však pri prehľadávaní a indexovaní celého internetu nie sú jediné komplikácie - vzniká potreba pre sťahovanie a spracovanie masívneho množstva informácií. Pre vyriešenie tohto problému je treba použiť viac ako jeden web crawler. Tu prichádzajú na rad paralelizované web crawlery, ktoré sú výnimočné ich schopnosťou súčasne získavať odkazy z rôznych zdrojov, pomocou osobitných inštancií, bez toho, aby sa prekrývali – tým sa čo najoptimálnejšie využije maximálna rýchlosť sťahovania na danej sieti.

Pri práci s paralelnými web crawlermi je vhodné mať jasný obraz o tom, čo sú vlastne inštancie, ako vyzerajú vzťahy medzi nimi a ako sa medzi sebou prepájajú webstránky, ktorými musia inštancie prechádzať. Pre uľahčenie tohto cieľa si definujeme nasledovné pojmy: Internet môžeme vnímať ako orientovaný graf, ktorého vrcholy sú reprezentované jednotlivými webstránkami a hrany tvoria odkazy medzi nimi [1]. Inštancia je proces, ktorý asynchrónne prehľadáva internet a snaží sa nájsť prepojenia, ktoré ho dovedú k novým, ešte neobjaveným informáciám. Úlohou paralelného web crawleru je teda optimálne prechádzať komplexným internetovým grafom, a to pomocou viacerých inštancií, bez toho, aby sa ich cesty krížili.

*Semestrálny projekt v predmete Metódy inžinierskej práce ak. rok 2023/24, vedenie: Ing. Richard Marko, PhD.

Historické súvislosti Na začiatku boli sekvenčné web crawlery štandardom. Fungujú tak, že prechádzajú hypertextovými odkazmi z jednej webstránky na druhú sériovým spôsobom, pričom získavajú a zbierajú informácie. S exponenciálnym rastom veľkosti a zložitosti internetu sa sekvenčné web crawlery stali neadekvátnymi na spracovanie obrovského množstva webstránok a zvládanie zvyšujúcej sa záťaže na systémy. Ako riešenie tohto problému sa zaviedlo paralelné prehľadávanie webu, ktoré umožňuje rýchlejšie a efektívnejšie vyhľadávanie informácií. Namiesto prechádzania webstránok po jednej, paralelné web crawlery rozdeľujú záťaž prehľadávania medzi viacero inštancií, čo umožňuje súčasné prehľadávanie rôznych webstránok súčasne. Prvým paralelným web crawlerom sa v roku 1994 stal program WebCrawler, ktorý dokázal sťahovať 15 webstránok súčasne. Počet indexovaných webstránok náhle vzrástol z 110 000 na 2 milióny. V roku 1998 vznikol paralelný web crawler Google, ktorý sa odlišoval schopnosťou ohodnotiť relevantnosť webstránok, čo bolo užitočné pri ich následnom zobrazovaní webovým prehliadačom [2].

Obr. 1: Orientovaný graf reprezentujúci vzťah medzi webstránkami



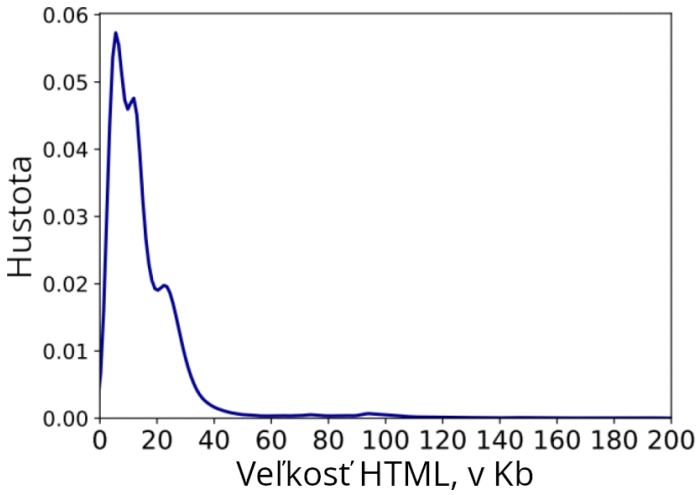
Obr. 2: Princíp získavania informácií web crawlerom [3]

2 Porovnanie rýchlosti

2.1 Podmienky pri testovaní

Princíp fungovania paralelných web crawlerov vyžaduje implementáciu, ktorá je rýchla a nenáročná na zdroje. Táto požiadavka sa stáva nevyhnutnosťou, vzhľadom na to, že indexovaný web obsahuje minimálne 5,27 miliárd webstránok [4]. Exituje mnoho spôsobov, ako implementovať samotné procesy jednolivých web crawlerov. V tejto časti porovnáme rozdiely vo výkone web crawlerov, ktoré boli vytvorené pomocou procesov, threadov (vlákien) a systémového volania `epoll` [1], napísané v programovacom jazyku C++ (vybraný pre rýchlosť a knižnicu `pthreads`, ktorá je v ňom dostupná) [4].

Pri testovaní implementácií, parametre ako kompilátor (GCC 9.3.0), webserver (operačný systém Ubuntu 20.04 LTS a serverový softvér NGINX), programovací jazyk (C++) a optimizácie pri kompilácii (`O3` [2]) zostávajú nezmenené. Testy budú spočívať v rýchlosti stiahnutia 132 094 HTML súborov, ktoré sú uložené na serveri, s rozložením veľkostí podľa nasledujúceho grafu [4]:



Obr. 3: Graf distribúcie veľkosti HTML súborov. [4]

2.2 Podrobnosti implementácií

Implementácia, ktorá využíva procesy, ich vytvára pomocou systémového volania `fork` [3] - zároveň využíva aj funkcie `mmap` a `pthreads` pre zjednodušenie synchronizácie procesov. Počet inštancií pre jednotlivé implementácie bol vybraný tak, aby každá implementácia dovŕšila jej najväčší možný výkon [4].

2.3 Výsledky

Z experimentov napokon vyplývajú nasledujúce výsledky (v poradí od najrýchlšej implementácie po najpomalšiu) [4]:

[1] Rozhranie poskytované linuxovým jadrom, určené pre oznamovanie I/O udalostí [5]
[2] Stupeň optimalizácie kompilátora GCC určený pre zvýšenie rýchlosti programu [6]
[3] Funkcia, ktorá vytvorí z rodičovského procesu nový proces (program sa rozdvojí) [5]

Implementácia	Priemerný čas (s)
Epoll	47.9
Vlákna	49.0
Procesy	50.1

Môžeme konštatovať, že systémové volanie `epoll` je najrýchlejšie v porovnaní s ostatnými testovanými implementáciami. Má aj nevýhody - `epoll` je dostupný iba na linuxovom jadre (existujú aj alternatívy dostupné v iných jadrách, ako napríklad `kqueue` na BSD systémoch), rovnako je ho ťažké implementovať, čo môže mať za následok chyby a bezpečnostné problémy. Vlákna majú síce nižší výkon, ale je ich ľahšie naprogramovať. Nakoniec implementácia pomocou procesov je síce najpomalšia, ale pre využitia, kde najvyššia možná rýchlosť nie je na prvom mieste, je postačujúca [4].

3 Zníženie záťaže na servery

3.1 Princíp a implementácia

40% prenosu dát internetom je spôsobených web crawlermi, preto by bolo vhodné nájsť riešenie nielen pre sťahovanie čo najväčšieho množstva dát, ale aj pre minimalizáciu opätovného sťahovania webstránok, ktoré už máme v databáze, za účelom aktualizovania ich obsahu. Riešenie spočíva v pridaní súboru UPDATE do každej webstránky na internete. Tento súbor by obsahoval odkazy na adresáry webstránky, kde nastala zmena [7].

Algoritmus crawlovania [7]:

1. Web crawler po prvýkrát navštívi webstránku a stiahne celý adresár (podľa súboru `robots.txt` [4])
2. Pre aktualizáciu databázy web crawler najskôr navštívi súbor UPDATE
3. Porovná ho s UPDATE súborom z poslednej návštevy webstránky
4. Ak sú súbory rôzne, web crawler stiahne aktualizované webstránky a uloží ich do databázy

Súbor môže samozrejme pridať len sám vlastník webstránky, čo by vyžadovalo veľkú úroveň koordinácie. Zo strany vlastníkov to má však výhodu - ich server nie je zaplavený žiadosťami o stiahnutie celého adresára, čo znižuje nároky na údržbu. Namiesto toho, aby web crawlery zakaždým sťahovali celý adresár, stačí im overiť si UPDATE súbor. Zo strany web crawlerov vzniká výhoda, že nedochádza k časovo náročnému prehľadávaniu webstránky, ktorej najnovšiu verziu už majú v databáze [7].

Existencia súboru `robots.txt` vo veľkom množstve webstránok dokazuje, že je možné takéto riešenie implementovať. Každý web crawler by mal byť schopný prehľadávať HTML kód, a každý vlastník webstránky by mal byť schopný upravovať súbor s HTML kódom, obsahujúci aktualizácie jeho webstránky. Preto by sa mal aj UPDATE súbor skladať z HTML [7].

[4] Súbor, ktorý oznamuje web crawlerom, ktoré časti webstránky môže indexovať

3.2 Simulácia a výhody

V experimente bola otestovaná teória stiahnutím najprv celého adresára webstránky, potom pomocou súboru UPDATE s jednou, dvomi a tromi aktualizáciami. Z výsledkov vyplýva, že so súborom s jednou aktualizáciou sa dokopy stiahne 6,5-krát menej webstránok ako bez neho [7].

Implementácia UPDATE súboru nevyžaduje sťahovanie žiadnych nových programov zo strany vlastníkov a úprava softvéru web crawlerov je minimálna. Je to celkovo veľmi optimálne riešenie problému nadmerného toku web crawlerov na internete [7].

4 Moderné riešenia

4.1 C-proc

Paralelné web crawlery sa skladajú z viacerých komponentov, ktoré sa dajú zrealizovať viacerými spôsobmi. Jeden z takých spôsobov je vytvorený pomocou crawlovacích inštancií C-proc. Každý C-proc je oddelený proces, ktorý nezávislo prehľadáva web. Výhodou takejto architektúry je zníženie záťaže na lokálne siete, pretože každý C-proc môže bežať na inej sieti. Jednotlivé inštancie sú ovládané centrálnym programom na správu web crawlerov [8].

Paralelné web crawlery typu C-proc sa rozdeľujú nasledovne [8, 9]:

1. Viacvláknový server a jeho časti - Koordinujú činnosť jednotlivých inštancií.
   (a) URL Dispatcher
   (b) DNS Resolver
   (c) URL Mapper
2. Inštancie web crawlerov - Jednotlivé programy C-proc, ktoré sťahujú webstránky

4.2 Ergate

Paralelný web crawler Ergate funguje na podobnom princípe ako C-proc, ale odstraňuje problém spoliehania sa na jeden program, ktorý riadi všetky inštancie. Tento cieľ je dosiahnutý pomocou distribuovanej architektúry, zloženej z kontrolného systému a systému na sťahovanie webstránok - každý z týchto systémov sa môže skladať z viacerých inštancií ovládačov a web crawlerov. Jednotlivé časti komunikujú medzi sebou cez MPI [5] a celok sa nazýva Cluster. Jednou z kľúčových vlastností Ergate je jeho schopnosť rozložiť záťaž na viacero počítačov, čo zvyšuje jeho efektivitu a umožňuje rýchlejšiu extrakciu údajov, čo vedie k výraznej úspore času v porovnaní s tradičnými paralelnými web crawlermi. Využitím distribuovaného prístupu dokáže Ergate spracovať veľké množstvo údajov, vďaka čomu je ideálny na extrahovanie informácií z webstránok s veľkým adresárom [10].

[5] Message Passing Interface - rozhranie na komunikáciu medzi procesmi

5 Paralelné web crawlery a spoločnosť

Technológia a ľudia V nekonečne sa rozširujúcom internete, paralelné web crawlery hrajú rolu navigátorov, bez ktorých by bolo oveľa ťažšie v globalizovanom svete fungovať. Mohli by sa teda paralelné web crawlery považovať za najdôležitejšiu časť internetu? Technológia poskytuje nástroje a infraštruktúru potrebnú na paralelizáciu, čo umožňuje rozsiahle zhromažďovanie údajov. Ľudia sú však zodpovední za navrhovanie, programovanie a monitorovanie týchto nástrojov, robia kritické rozhodnutia o ich správaní a zabezpečujú, že sú v súlade s etickými zásadami. Ľudská účasť je rozhodujúca pri vytváraní algoritmov, ktoré efektívne rozdeľujú úlohy prehľadávania, spravujú oneskorenia pri extrakcií odkazov s cieľom dodržiavať pravidlá webstránok a vyhýbajú sa nadmernej spotrebe dát na sieti. Paralelné web crawlery môžeme teda považovať za nevyhnutný nástroj na využitie niečoho, čo vzniklo vďaka ľuďom, ktorí vytvorili internet. Sú súčasťou internetu, ale nie sú jeho najdôležitejšou časťou - tvoria jeden celok, ktorého časti sú rovnako dôležité.

Spoločenské súvislosti Paralelné web crawlery sa dajú používať aj na osobnú zábavu. Jednotlivci môžu vytvárať paralelné web crawlery na získavanie webstránok a zhromažďovanie mediálneho obsahu, ako sú obrázky, videá alebo súbory GIF, do svojich osobných zbierok. Toto použitie je bežné najmä medzi umelcami, dizajnérmi alebo nadšencami vizuálnych médií, ktorí hľadajú inšpiráciu alebo jednoducho radi budujú osobné archívy [11]. Ďalším využitím je zhromažďovanie konkrétnych informácií z viacerých zdrojov. Napríklad vedci a študenti môžu využívať paralelné web crawlery na získavanie relevantných údajov pre svoje štúdie. Umožňujú im analyzovať veľké množstvo údajov a vyvodzovať zmysluplné závery bez toho, aby museli manuálne navštevovať veľký počet webstránok, čo šetrí čas aj námahu.

Udržateľnosť a etika Napriek svojim výhodám, paralelné web crawlery predstavujú určité výzvy z etického hľadiska. Niektorí jednotlivci ich využívajú na zhromažďovanie osobných údajov, ako sú e-mailové adresy, telefónne čísla alebo profily sociálnych médií, na marketingové alebo dokonca škodlivé účely. Rozsiahlosť zhromaždených informácií teda ohrozuje súkromie používateľov, čo si vyžaduje bezpečnostné opatrenia a právne rámce na ich ochranu (ako napríklad GDPR v Európskej Únií). Okrem toho je kritické predísť neobjektívnemu obsahu a zabezpečiť presnosť a dôveryhodnosť získaných informácií. S cieľom riešiť tieto problémy, mnohé webstránky zaviedli opatrenia na obmedzenie alebo kontrolu prístupu web crawlerov, ako napríklad súbory robots.txt, CAPTCHA a rôzne časové limity [12].

6 Záver

Tento článok poskytol informácie o používaní paralelných web crawlerov a ich optimalizácii, pričom sme sa venovali aj obavám súvisiacim s vplyvom tejto technológie na jednotlivé webstránky. Celkovo sa v článku zdôraznilo, že pri využívaní tejto technológie je dôležité zvážiť jej vplyv na jednotlivé webstránky, ako aj na širšiu spoločnosť.

Literatúra

[1] Shruti Sharma and Parul Gupta. The anatomy of web crawlers. In *International Conference on Computing, Communication & Automation*, pages 849–853, 2015.

[2] Seyed M Mirtaheri, Mustafa Emre Dinçktürk, Salman Hooshmand, Gregor V Bochmann, Guy-Vincent Jourdan, and Iosif Viorel Onut. A brief history of web crawlers. *arXiv preprint arXiv:1405.0749*, 2014.

[3] Md Abu Kausar, VS Dhaka, and Sanjeev Kumar Singh. Web crawler: a review. *International Journal of Computer Applications*, 63(2):31–36, 2013.

[4] Andriy Sultanov, Maksym Protsyk, Maksym Kuzyshyn, Daria Omelkina, Vyacheslav Shevchuk, and Oleg Farenyuk. Comparison of performance of the popular approaches to implementing parallel crawlers. In *2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)*, volume 1, pages 349–352, 2021.

[5] M. Kerrisk. *The Linux Programming Interface: A Linux and UNIX System Programming Handbook*. No Starch Press, 2010.

[6] Richard M Stallman. *Using the gnu compiler collection: a gnu manual for gcc version 4.3. 3*. CreateSpace, 2009.

[7] Shekhar Mishra, Anurag Jain, and AK Sachan. Smart approach to reduce the web crawling traffic of existing system using html based update file at web server. *International Journal of Computer Applications*, 11(7):34–38, 2010.

[8] Shruti Sharma, AK Sharma, and JP Gupta. A novel architecture of a parallel web crawler. *International Journal of Computer Applications*, 14(4):38–42, 2011.

[9] Tithi Dhar, Sayan Mazumder, Susnigdha Dhar, Susovan Karak, and Debraj Chatterjee. An approach to design and implement parallel web crawler. In *2021 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*, pages 1–4, 2021.

[10] Min Wu and Junliang Lai. The research and implementation of parallel web crawler in cluster. In *2010 International Conference on Computational and Information Sciences*, pages 704–708, 2010.

[11] Maureen Pennock. Web-archiving. *DPC technology watch report*, 13(1):1–45, 2013.

[12] Mike Thelwall and David Stuart. Web crawling ethics revisited: Cost, privacy, and denial of service. *Journal of the American Society for Information Science and Technology*, 57(13):1771–1779, 2006.