VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

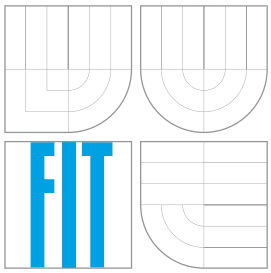
AUTOMATIZOVANÁ ANALÝZA WWW STRÁNEK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE NIKITA VAŇKŮ
AUTHOR

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

AUTOMATIZOVANÁ ANALÝZA WWW STRÁNEK

AUTOMATED WEB PAGE ANALYSIS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — NIKITA VAŇKŮ
AUTHOR

VEDOUCÍ PRÁCE — Ing. RADEK BURGET, Ph.D.
SUPERVISOR

BRNO 2015

Abstrakt

Výtah (abstrakt) práce v českém jazyce.

Abstract

Výtah (abstrakt) práce v anglickém jazyce.

Klíčová slova

Klíčová slova v českém jazyce.

Keywords

Klíčová slova v anglickém jazyce.

Citace

Nikita Vaňků: Automatizovaná analýza WWW stránek, bakalářská práce, Brno, FIT VUT v Brně, 2015

Automatizovaná analýza WWW stránek

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana ...

.....
Nikita Vaňků
3. února 2015

Poděkování

Zde je možné uvést poděkování vedoucímu práce a těm, kteří poskytli odbornou pomoc.

Obsah

Kapitola 1

Úvod

World Wide Web (dále jen WWW) dokumenty jsou stále více využívanějšími sdělovacími prostředky. Jejich počet neustále roste [zdroj]. Využití WWW dokumentů se nachází. WWW dokumenty se vuyžívají na širokou škálu účelů. Můlžeme narazit na dokumenty se spíše statickým obsahem jako jsou blogy, firemní reprezentace atd. Můžeme ale také narazit na velice dynamické dokumenty jako jsou internetové vyhledavače. informační systémy, webové aplikace aj.

Množství webových stránek na doméně je téměř neomezený a na internetu lze najít domény o několika stránek, ale i domény jejichž obsah roste po více jak deset let. Takové domény pak cyklicky procházejí refactoringem kodu a někdy i celého systému. Může se pak stát, že se řada stránek zapomene či vytratí. Či naopak, že systém roste a nalepují se na něj další a další časti mnohdy zbytečně. Stejně tak se staré neodstraní, třeba i z důvodu, že v programátorském týmu "nikdo nemá tušení co to dělá a co se stane, pokud se to dá pryč".

Z tohoto a dalších důvodů využívaji týmy mnoho nástrojů, které analyzují zpracování požadavku serveru. Jedná se o měření rychlosti zpracování backendu a frontendu. Velikost stahovaných klientských zdrojů jako jsou například soubory kaskádových stylů, obrázky či Javascriptové knihovny. Jednim z těchto nástrojů jsou Webové crawlery, neboli pavouci.

Pavouci našli největší využití hlavně u internetových vyhledavačů. Jsou navrženi tak taby v krátkém čase dokázali zpracovávat najednou velké množství stránek z různých domén. Při zpracovávání hledá odkazy na další stránky a ty také analyzuje. Vzhledem k velikosti a neústalého růsta pavouci mnohdy svou práci nikdy nekončí, nebo jsou spouštěni pravidelně. Úloha pavouků může být různá.

Mým cílem je vytvořit pavouka, který bude analyzovat pouze jednu doménu. Předpokládá se, tedy že bude zpracovávat pouhé stovky až tisíce stránek. Výstupem analýzy mají být programátorům užitečná data jako grafické zobrazení provázanosti stránek. Měření rychlosti a analýza stahování stránek. Rendering stránek. Součástí analýzy je možno také sledování chyb Javascriptu či validace DOM objektu.

V této práci se budu věnovat několika hlavních částí této problematiky. První část bude věnována analýze problematiky pavouků. V druhé části zpracuji návrh vlastního pavouka. Ve třetí části budu řešit implementaci vyhotoveného návrhu a poslední část je věnována experimentům s pavoukem a výsledky jeho analýz.

Kapitola 2

Teorie

Doplnit.

Kapitola 3

Současní pavouci

Dnes se pavouci využívají převážně k indexování internetu pro internetové vyhledavače. Mezi ty nejvíce známe můžeme zařadit Google či Bing. Detaily implementace těchto pavouků jsou pochopitelně utajeny. Kromě pavouků analyzující celý internet bez omezení existuje i celá řada úzce specializovaných pavouků pro určitou činnost. Například `skipfish` je OpenSource pavouk testující zabezpečení jedne domény a odhaluje slabá místa, které by mohli využít útočníci. `HTTrack` je pavouk, který po zadání domény prochází veškerý její obsah a stahuje ho do uložiště. Stránky lze potom prohlížet Offline.

3.0.1 Analýza URL

Uniform Resource Locator (dále jen URL, neboli též adresa) jsou definovány v (RFC 3986). Je potřeba rozlišit relativní a absolutní adresu. V případě relativní adresy se zohledňuje aktualní adresa dokumentu. Součástí URL může být též kotva, kotva je definována atributem `id` v elementu `<a>`, kotvy nemají pro pavouka žádný význam a jsou tedy ignorovány. V případě nalezení většího množství odkazů se stejnou adresou v dokumentu, pavouk se zachová jako by pracoval s adresou jedinou.

3.0.2 Extrakce URL

Extrakce adres je možné analýzou HTML webových dokumentů. Prozkoumáním vybraných elementů, můžeme získat potřebné odkazy. Mezi zmiňované elementy patří[1].

- `<a></a>` atribut `href`
- `<form></form>` atribut `action`
- `<link>` atribut `href`
- `<script></script>` atribut `src`
- `<style></style>` atribut `src`

[1] Je potřeba brát v potaz, že URL mohou být v dokumentu vloženy také jako pouhý text či při zpracování události javascriptem. Tyto odkazy pavouk pochopitelně nenajde.

3.0.3 Zaznamenání časových událostí

Pro uživatele je důležitý co nejkratší čas mezi odesláním požadavku na server a zobrazení stránky. Tento čas je možné rozdělit na dvě hlavní části a to:

- čas stráveny zpracováním požadavku na serveru - Backend
- čas strávený stažení a zobrazení zdrojových dokumentů - Frontend

Webová stránka, která se zobrazí uživateli v prohlížeči je mnohdy složena z více souborů. Jedná se jednak o HTML/XML dokument, ale také o obrázky, Kaskádove styly, Javascript a další soubory. Moderní prohlížeče dnes dokáží stahovat více souborů z jednoho serveru najednou aby čas strávený stahováním co nejvíce urychlily. Množství vláken se však u každého prohlížeče liší a uživatel může s touto hodnotou manipulovat. [Zdroj]

Kapitola 4

Návrh aplikace

Aplikace je složena z několika částí. Jedná se o algoritmus pavouka, který stahuje a analyzuje dokumenty. Databázovým uložištěm, které uchovává analyzované data a grafická nadstavba (GUI) přes kterou je aplikace řízena. Pro vykreslování dokumentů jsou využity technologie Headless Browseru, v tomto případě byl využit PhantomJS, který pracuje na vykreslovacím jádře WebKit.

4.0.4 Java

Byla využita platforma Java s platformou JavaFX, která umožňuje tvorbu Rich Internet Applications (dále jen RIA). JavaFX aplikace je možno využít i jako desktopové aplikace. Aplikace vytvořena součástí této práce byla naprogramována pro Java 8, JavaFX 8.

4.0.5 Omezení prohledání

Cílem aplikaci je analyzovat pouze jedinou doménu. Při prohledávání je tedy nutno zamezit zabloudění pavouka na cizí domény a prohledávání zcela jiné části internetu. Toto je dosaženo kontrolou domény nejvyššího soukromého řádu společně s veřejným suffixem. Pokud se doména tohoto řádu nebo suffix liší, adresa se neprohledává.

Je třeba také počítat s tím, že množství dokumentů a odkazů na doméně není předem známo. Doména může obsahovat několik desítek dokumentů, ale i několik stovek. Aplikace musí být na tuto skutečnost připravena a v případě procházení takto velké domény zareagovat. Toto je uskutečněno stanovením maximální hloubky prohledávání. Před prohledáním prvního dokumentu je aktuální hloubka 0 a každým dalším dokumentem se hodnota zvýší o 1. V případě dosažení maximální hloubky prohledávání se z tohoto dokumentu neextraktují další adresy.

4.0.6 Analýza DOM objektu

Extrakce adres z HTML dokumentů je možné za pomocí prohledání Document Object Modelu (dále jen DOM). V objektu se hledají elementy a atributy zmíněne v **??**. Vytažené adresy se musí dále zpracovat jak bylo popsáno v **??**, k adrese se také přídá záznam na kterém dokumentu byl nalezen a následně se vloží do fronty ke zpracování.

Pro práci s DOM objektem a extrakci adres je využit HTML parser `jsoup`.

4.0.7 Databáze

Jako databáze byla zvolena `OrientDB 2.0.1 Community Eddition`, jedná se o hybridní NoSQL databáze umožňujíci pracovat jak v Dokument (MongoDB) tak Grafovém (Neo4j) modu.

Aby aplikace nemusela skladovat v paměti příliš velké množství dat, data se co nejdříve po zpracování přesunou do databáze. Jelikož zpracované data ve své podstatě vytváří orientovaný graf webových dokumentů, byla pro tuto potřebu zvolena grafová databáze. Způsob uložení dat v databázi tedy co nejvíce připomíná způsobu uložení dat na doméně.

Grafová databáze je sestavena stejně jako graf ze dvou základních stavebních prvků. Vrcholy a hrany mezi vrcholy. Jako vrcholy si můžeme například představit samotné dokumenty. Hrany by poté mohly být odkazy mezi těmito dokumenty. Kompletní popis hiearchie v databázi lze vidět na následujícím diagramu a tabulce.

Název	**Typ**	**Popis**
Resource	Vrchol	Webový dokument
Form	Vrchol	Formulář
LinkTo	Hrana	Odkaz mezi dokumenty
Includes	Hrana	Připojení formuláře k dokumentu

[Diagram]

4.0.8 Vykreslení dokumentu

Pro vykreslování webových dokumentu byl využit Headless browser `PhantomJS 2.0`. PhantomJS obsahuje vykreslovací jádro WebKit, lze s ním tedy emulovat vykreslování podobné prohlížečům se stejným jádrem jako je např Google Chrome, Safari či Opera.

PhantomJS je plně ovladatelný Javascriptem.[Doplnit]

[Diagram]

Kapitola 5

Implementace aplikace

Kapitola 6

Experimenty

Kapitola 7

Závěr

Kapitola 8

Slovníček pojmů

- Frontend - Klientská část zpracování požadavku
- Backend - Serverová část zpracování požadavku
- Crawler - pavouk
- Resource - zdroj/webový objekt
- Headless browser - Prohlížeč běžící v pozadí