PROBLEMATIKA INTELIGENTNÍHO AUTOMATICKÉHO MAPOVÁNÍ WEBOVÝCH STRÁNEK

ŘIMNÁČ MARTIN [1], ŠUSTA RICHARD [2], ŽIVNŮSTKA JIŘÍ[3]

Katedra řídicí techniky, ČVUT-FEL, Technická 2, Praha 6,
tel. +420 224 357 359, fax. + 420 224 918 646

email[1]: rimnacm@cs.cas.cz, email[2]: susta@control.felk.cvut.cz, email[3]: zivik@volny.cz

Klíčová slova: webové stránky, mapování, webmapa, automatické prohledávání

Abstrakt: Příspěvek zobecňuje dvouleté zkušenosti autorů získané při řešení problematiky automatického mapování webových stránek, což představuje problém velmi podobný identifikaci struktury neznámého systému. Vzájemnou provázanost stránek pomocí hypertextových odkazů lze sice reprezentovat grafy, avšak jejich vytváření není zdaleka triviální. Prosté pokusy o zmapovaní webu pomocí automatického prohledávání odkazů vedou často na nepoužitelné grafy s extrémně složitými strukturami, a případně i s nekonečnou velikostí, zejména při analýze dynamicky generovaných stránek.

Článek popisuje realizovanou metodu pro také automatické vytváření hierarchických map - ty lze použít Obecně navržený klient/server nástroj je určen pro generování map struktury webových serverů. Struktura webu je pak vnitřně reprezentována jako orientovaný graf. Článek se zabývá vedle problematiky konstrukce tohoto grafu z dat webových serverů hlavně algoritmem pro generování hierarchických map. Ty mohou být využity pro orientaci návštěvníků webových stránek nebo pro webmasterskou činnost, dále pro usnadnění orientace návštěvníků webu nebo pro detekci chyb ve struktuře stránek.

1 ÚVOD

Na obsah webových serverů můžeme pohlížet jako na rozsáhlou znalostní bázi dat s velkou neurčitostí. Tuto bázi potřebujeme strukturovat a prořezávat, ať na globální úrovni pomocí internetových fulltextových strojů nebo katalogů, tak na úrovni jednotlivých webových serverů.

Popisovaná metoda chápe strukturu webových stránek jako orientovaný graf. Webové stránky jsou reprezentovány uzly, hypertextové a hypermediální odkazy pak orientovanými hranami. Uvážíme-li, že stránky mohou být generovány dynamicky, hloubka takového grafu může být i nekonečná.

Metoda nejprve data získá, webové stránky stáhne z webového serveru a pomocí prohledavače (crawleru) [1] převede stránku na uzel s danými vlastnostmi a orientované hrany reprezentující odkazy mezi stránkami. Tato vytvořená struktura je uložena do databáze. Pojem webová stránka zde označuje dokument libovolného, textem reprezentovatelného, formátu získaný z webového serveru.

Nashromážděná data jsou užívána k výpočtu ohodnocení stránek, tzv. PageRank [3], pro člověka jsou však velmi nepřehledná, navíc zobrazení všech odkazů mezi stránkami je výpočetně náročné (srovnatelné s úlohou obchodního cestujícího, která je NP-úplná).

Naopak mapa vytvořená pomocí popisované metody [2] je podgrafem tohoto grafu, obsahuje minimální počet hran při zachování počtu uzlů, navíc požaduje mapu ve formě stromu.

Systém je doplněn o webové rozhraní, které vedle administrace generování map umožňuje fulltextové vyhledávání.

2 ZÍSKÁVÁNÍ INFORMACÍ

2.1 Stažení

V první fázi potřebujeme získat data z webových stránek umístěných na obecném webovém serveru. Systému explicitně zadáme URL adresu libovolné stránky umístěné na serveru, pro jednoduchost uvažujme domovskou stránku webové prezentace (*homepage*). Systém pomocí GNU nástroje *wget* v rekurzivním módu stáhne tuto stránku a následně provede analýzu odkazů. Pokud některá z odkazovaných stránek ještě nebyla stažena, je zařazena do fronty a stažena v některém dalším průchodu cyklu. Cyklus je ukončen kritériem na maximální hloubku grafu, tedy dosažení zadaného počtu stránek mezi analyzovanou a zadanou stránkou. Během procesu stahování detekujeme neplatné odkazy – odkazy na neexistující (či nedostupné) stránky.

Výsledkem tohoto kroku je kopie všech ze zadané stránky přístupných stránek na lokální souborový systém včetně adresářové struktury.

2.2 Korespondence URL a obsahu stránky

U dynamicky generovaných webových stránek je nutné navíc vyřešit netriviální úlohu korespondence mezi URL adresou stránky a jejím o obsahem. URL adresa takových stránek se skládá ze jména skriptu, které následuje znak ‚?' a znakem ‚&' oddělený seznam předávaných atributů a jejich hodnot.

Ne každý atribut však má vliv na obsah dynamicky vygenerovanou stránku. Mějme např. URL adresu *http://example.com/tisk.php?jazyk=cz&clanek=123&uzivatel=987*, pak atributy *clanek* a *jazyk* udávají číslo a jazykovou mutaci tisknutého článku a tedy zásadně ovlivňují obsah stránky. Naopak atribut *uzivatel* specifikuje číslo návštěvníka, ten žádným způsobem obsah neovlivňuje.

O vlivu parametrů URL adresy na obsah generované stránky systém nemá žádnou informaci. Sofistikované metody detekují tento vliv pomocí různých heuristik. Pro potřeby zákaznického mapování stránek postačuje možnost přidání takovéto apriorní informace pomocí voleb systému.

Parametry každého skriptu rozdělíme na *povinně uvedené*, *povinně nabývající* jedné z množiny hodnot a *nepovinné*. Speciálním případem nepovinného atributu je *doplnění atributu* na předdefinovanou hodnotou, není-li ve výčtu parametrů URL adresy uveden. Vliv neuvedených atributů na obsah stránky považujeme za zanedbatelný. Pokud URL stránky nevyhovuje zadanému výčtu podmínek na parametry, je stránka označena jako *ignorovaná*. V našem příkladu by např. parametr *clanek* byl povinný, *jazyk* nepovinný s předdefinovanou hodnotou *cz*. Parametr *uzivatel* by v seznamu nebyl uveden.

V poslední fázi operací s URL adresou seřadíme parametry podle abecedy a tím získáme jednoznačnou transformaci URL adresy stránky na odpovídající obsah. Na úrovni souborového systému nahradíme soubor s původní URL adresou symbolickým odkazem na soubor s adresou upravenou.

2.3 Analýza obsahu stránky

Nyní můžeme přikročit k samotné analýze stránek. Každá stránka je popsána informacemi z HTTP hlavičky a samotným obsahem stránky. Jedním z atributů HTTP hlavičky je i typ obsahu stránky, podle jehož hodnoty zavoláme dílčí filtr na samostatné zpracování obsahu stránky. U filtru pro (X)HTML formát navíc musíme uvažovat možnost explicitního předefinování hodnot atributů HTTP hlavičky.

HTTP hlavička poskytuje informace jak o stránce, tak o serveru, případně i o spojení mezi klientem a serverem. Pro naše účely zmíníme vedle typu obsahu (*content-type*) i délku obsahu (*content-length*), dobu poslední modifikace obsahu (*last-modified*), dobu platnosti (*expires*), příp. hlavička může obsahovat i jazykovou mutaci stránky.

Systém umožňuje obsah stránek rozdělit do textových bloků a ty fulltextově zaindexovat. Příkladem takového bloků je např. titulek stránky, text na odkazu na danou stránku, odstavec atd.

Výhodou tohoto přístupu je ovlivnění výpočtu celkové relevance odkazu podle typu textového bloku, tedy výskyt klíčového slova v titulku stránky je pro vyhledavač významnější nežli výskyt slova v odstavci.

Filtr pro (X)HTML formáty rovněž registruje odkazy na stránky, přičemž na URL adresy odkazovaných stránek je aplikována výše popsané transformace URL adres.

Analýza probíhá postupně pro všechny stažené stránky reprezentované soubory, symbolické linky přeskakujeme. Analýzu provádíme hierarchicky, tedy první analyzujeme úvodní stránku, pak všechny stránky odkazované ze stránky úvodní atd. až do maximální hloubky grafu. Stránky, na které je odkazováno z maximální hloubky a dosud nebyly zaindexovány, označíme jako stránky *registrované*.

Každou vrstvu (uzly se shodnou hloubkou) uložíme do databáze zvlášť, což umožňuje řízení zpracování stránek z databáze. Výhodou tohoto přístupu je i možnost paralelního generování map až do hloubky vrstvy před aktuálně analyzovanou vrstvou stránek.

2.4 Aktualizace

Použitý nástroj *wget* umožňuje jednou již stažené stránky pouze aktualizovat, tedy znovu stáhnout pouze ty stránky, které byly změněny. Využijeme-li v souborovém systému symbolických odkazů a znalostí o statusu stránek (především ignorované stránky) z databáze, další aktualizace praxi jsou podstatně méně časově náročné. Vlastní systém obsahuje prostředky pro automatickou aktualizaci stránek jednou za specifikovanou periodu.

2.5 Ohodnocení stránek

Pro potřeby vyhledavače a generování map potřebujeme každou stránku ohodnotit. V systému je použita základní verze metody *PageRanku* [3]. Její modifikace využívají i komerční vyhledávače včetně Google, který ji poprvé implementoval. Předpokladem správné funkčnosti metody je, že dobrá stránka odkazuje na jiné, stejně dobré stránky. Měřítkem kvality stránky je tedy počet na stránku odkazujících stránek vážený jejich kvalitou. Zaveďme následující značení. Množinu všech stránek p označme P, tato množina reprezentuje uzly grafu. Definujme množinu odkazů L, jednotlivé prvky množiny deklarujme jako $< a, b >; a, b \in P$ ve významu odkazu ze stránky a na stránku b. Množina L tedy reprezentuje hrany grafu. Zaveďme zobrazení $PR(p)$, které každé stránce p přiřadí její ohodnocení.

Vzorec pro výpočet ohodnocení (Page-Rank) je

$$PR(p) = (1-k) + k \sum_{<q_i, p> \in L} \frac{PR(q_i)}{|< q_i, x >|}; x, q_i, p \in P \qquad \text{(V1)}$$

Symbol k je konstanta algoritmu, $k \in (0,1)$, doporučená hodnota k=0,85 zajišťuje optimální konvergenci. Součet je přes všechny stránky q_i odkazující na stránku p. Symbolem $|< q_i, x >|$ rozumíme počet stránek odkazovaných ze stránky q_i. Příklad výpočtu je na obrázku 1.

Ze vzorce vyplývá, že pro $\forall p \in P : PR(p) \geq 1 - k$. Této vlastnosti lze zneužít ke zvýhodňování stránek [4]. Tyto metody ale sofistikované vyhledavače detekují a penalizují zvýhodňované stránky.

3 GENEROVÁNÍ MAP

V souladu s předchozím textem zjemníme množinu P na množinu zaindexovaných stránek P_I, množinu registrovaných stránek P_R, množinu ignorovaných stránek P_G a množinu neplatných stránek (přesněji URL adres) P_N.

3.1 Kontrola platnosti odkazů

Nejprve zmíníme automatický mechanizmus kontroly platnosti odkazů. Výsledkem operace je výčet odkazů (hran v grafu), které vedou na neplatné stránky. Formálně tedy

$$L_{inv} = \{< p, q >, p \in P_I, q \in P_N\} \tag{A1}$$

Jednoduše lze nahlédnout, že algoritmus A1 je složitosti *o(n)*.

Tento algoritmus využijí webmasteři rozsáhlých webových prezentací. Systém bude periodicky aktualizovat svoji bázi dat stále se vyvíjející webové prezentace a upozorní webmastera na stránku, která obsahuje neplatný odkaz včetně informace o URL adrese odkazované stránky.

3.2 Seznam externích odkazů

Při SEO optimalizaci stránek je žádoucí znát všechny externí stránky, na které je z analyzované webové prezentace odkazováno. Zda-li se jedná o externí zdroj poznáme testem shodnosti jmen domén. Formálně doménu stránky označme symbolem $D(p)$, pak mapu externích odkazů vygenerujeme jako výčet z množiny L dle

$$L_{ext} = \{< p, q >, D(p) \neq D(q), p \in P_I, q \in P_I \cup P_R\} \tag{A2}$$

Povšimněme si, že podle A2 neplatná URL adresa, byť s externí doménou, není součástí externích odkazů, ale považuje se za chybu na straně analyzované webové prezentace. Tak tomu je i u jiných automaticky řízených prohledavačů.

3.3 Hierarchická mapa

Základní motivací popisovaného systému je generování hierarchické mapy. Požadujme graf H se stromovou strukturou, $H = (\overline{P}, \overline{L})$. Symbol $\overline{P}^i$ reprezentuje uzly v i-té vrstvu stromu H.

Specifikujme podmínky na graf H.

$$\overline{P}^0 = \{p_0; p_0 \in P_I\} \tag{P1}$$

$\forall p \in P_I \cup P_R, D(p) = D(p_0)$ odpovídá právě jeden uzel v $\overline{P}$. (P2)

$$\overline{L}^i = \left\{ l =< p, q >; p \in \overline{P}^{i-1}; q \notin \bigcup_{j=0}^{i} \overline{P}^j; l \in L - L_{ext} \right\} \tag{P3}$$

$$\exists l_1, l_2; l_1 =< p_1, q >, l_2 =< p_2, q >; PR(p_1) \geq PR(p_2); p_1, p_2, q \in P_I \cup P_R : l_1 \in \overline{L}, l_2 \notin \overline{L} \tag{P4}$$

Interpretujme tyto podmínky:

1. Hierarchická mapa má ve svém kořeni jedinou stránku a to homepage webové prezentace. Ta musí být zaindexována.
2. Hierarchická mapa obsahuje všechny zaindexované a registrované stránky přístupné z homepage. Každá stránka je v mapě uvedena právě jednou. Mapa obsahuje stránky pouze z domény, která je určena homepage.

3. Z množiny odkazů L vybíráme pouze ty odkazy $\overline{L}$, které vedou na v mapě dosud neuvedené stránky. Z podmínek 1 až 3 je možné nahlédnout, že se jedná o algoritmus generující kostru grafu.
4. V praxi se běžně vyskytují případy, kdy ze dvou různých stránek patřících do stejné vrstvy (vyplývá z předchozích podmínek) je odkazováno na jednu stránku, kostra grafu tedy není jedinečná. Pak je upřednostněna ta stránka, která má větší hodnotu ohodnocení.

Se 4. podmínkou souvisí problém možné neexistence maxima ohodnocení stránek, zde existuje pouze suprémum. V takovém případě povolujeme nedeterministické řešení, vybíráme náhodně jednu ze stránek, jejíž ohodnocení nabývá hodnoty supréma.

Algoritmus je implementován pomocí SQL funkce přímo v prostředí databáze *PostgreSQL* v jazyce *pl/pgSQL*, funkce vrací XML-validní řetězec popisující danou mapu. Výpočet rozdělujeme do dvou částí.

Nejprve generujeme hierarchickou mapu, jedná se o algoritmus postavený na klasickém prohledávání do šířky. Algoritmus startujeme s libovolnou hranou vedoucí do homepage p_0 odpovídajícímu uzlu. Hrany grafu, které vyhovují podmínce P3, vložíme do dočasné tabulky. Ta vedle počátečního a koncového uzlu hrany obsahuje i označení vrstvy (abychom mohli testovat jen aktuální vrstvu).

Podmínku přidání hrany na q z P3 je pro potřeby databázového algoritmu přepíšeme podle

$$\exists q \in \bigcup_{j=0}^{m} \overline{P}^{j} \Leftrightarrow \exists < p,q > \in \overline{L} \qquad \text{(P3.a)}$$

Algoritmus končí, pokud neexistuje žádná další hrana na uzel, který ještě není součástí mapy. Hloubka hierarchické mapy je m. Tedy

$$\neg\exists < p,q > \in L : q \notin \bigcup_{j=0}^{m} \overline{P}^{j} \qquad \text{(P5)}$$

Uvažujeme, že z optimalizačních důvodů provádíme na jednu vrstvu mapy právě jeden výběr hran. Podmínku P3 upravíme na podmínku P3.b (resp. ekvivalentní podmínku podle P3.a), která neuvažuje stránky v aktuálním kroku, protože podmínku P3.a v této podobě nelze databázově interpretovat.

$$\overline{L}^{i} = \left\{ l =< p,q >; p \in \overline{P}^{i-1}; q \notin \bigcup_{j=0}^{i-1} \overline{P}^{j}; l \in L - L_{ext} \right\} \qquad \text{(P3.b)}$$

Pak ale musíme připustit, že v aktuální i-té vrstvě se objeví více uzlů téže stránky, každý spojený hranou z jiného uzlu, tedy situace

$$\exists l_1, l_2; l_1 =< p_1, q >, l_2 =< p_2, q >; p_1, p_2 \in \overline{P}^{i-1}, q \in \overline{P}^{i} \qquad \text{(P6)}$$

To je ale v rozporu s podmínkou P2. Proto zavádíme podmínku P4 a ze všech takových hran vybíráme právě jednu s maximální hodnotou ohodnocení (tupá nerovnost v podmínce P4). Tím splňujeme všechny podmínky a algoritmus generuje kostru grafu dle původních požadavků.

Předpokládejme, že samotný výběr hran z databáze je $o(n)$ a test na podmínku každé hrany je $o(n)$. Celková složitost výběru hran s ohledem na podmínku je $o(n^2)$.

Uvažme krajně nepříznivý případ, kdy pro každý uzel p_i existuje právě jedna hrana do uzlu p_{i+1}. Pak prohledávání proběhne celkem $(n-1)$-krát. Celková složitost této části algoritmu generování mapy je tedy $o(n^3)$.

Nyní máme v dočasné tabulce uloženy všechny hrany $\overline{L}$ potřebné pro vygenerování hierarchické mapy *H*.

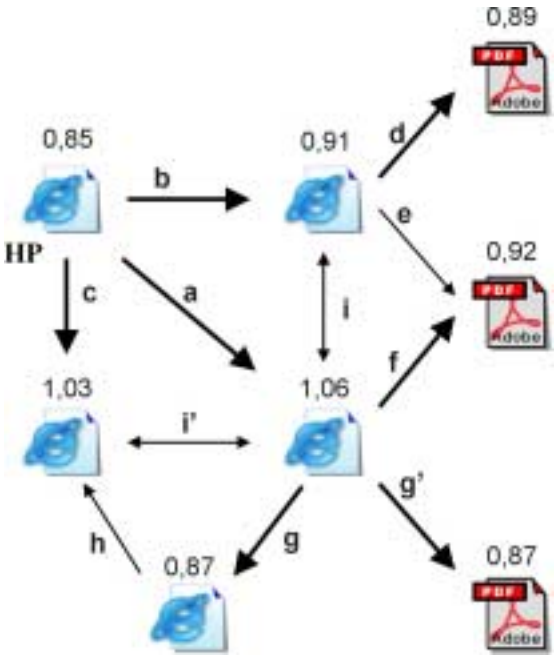
Pokračujeme druhou fází, generováním řetězce reprezentujícího hierarchickou mapu. Jedná se opět o prohledávání, tentokráte do hloubky. Složitost této fáze je *n*-krát *o(n)* výběr z dočasné tabulky, tedy $o(n^2)$.

Celková složitost celého algoritmu včetně vygenerování výstupu ve formátu XML je $o(n^3)$. Efektivní složitost je však podstatně nižší.

Takto vygenerovanou mapu lze pomocí XSL transformovat do PDF formátu nebo libovolného designu XHTML stránky, se zobrazením všech nebo jen některých hodnot získaných předchozím procesem z webových stránek. Seznam zobrazovaných hodnot může být rozšířen o další výpočty nad strukturou XML, např. je implementováno zobrazení grafu u každé stránky v mapě, který ukazuje velikost odkazovaných stránek (a obecně souborů) podle jejich typu obsahu. Tak lze např. bezpečně na první pohled poznat galerii obrázků v dané webové prezentaci.

3.4 Příklad

Ilustrujme algoritmus generování hierarchické mapy obrázkem 1, nad každým uzlem je uvedena hodnota PageRanku po první iteraci, úvodní stránka je označena *HP*.



Obrázek 1 - Příklad generování hierarchické mapy

V prvním kroku vybereme pouze ty hrany, které odkazují z úvodní stránky, tedy hrany *a*, *b*, *c* (pravidlo P2). V druhém kroku vybíráme ty hrany, na jejichž uzly odkazovaly hrany z předchozího kroku a zároveň na jejich cílové uzly nevede žádná hrana (pravidlo P3). Pozitivním případem jsou hrany *d, f, g* a *g′*, negativním pak hrany *i* a *i′*, které odkazují na uzly již odkazované z jiných hran. Hrana *e* není vybrána podle pravidla P4, neboť hrana *f* odkazuje na stejnou stránku ze stejné úrovně, ale odkazující stránka má vyšší hodnotu PageRank. Hrana *h* není v mapě uvedena, ukazuje podobně jako hrany *i* a *i′* na uzel, který již byl do mapy přidán v některém z předchozím kroku.

V mapě již jsou obsaženy všechny uzly (pravidlo P5), které byly součástí celé struktury webové prezentace, proto algoritmus končí. Výsledná hierarchická mapa je v obrázku naznačena tučnými hranami.

4 SHRNUTÍ

Příspěvek popisuje implementaci obecného nástroje pro práci s webovými stránkami. Součástí nástroje je i webové rozhraní, ze kterého je nástroj ovládán.

Nástroj umožňuje určitý stupeň paralelizace pomocí zavedeného balíčkového systému a prioritně řazené fronty dotazů. Řízení je ponecháno v režii databázového stroje, jednotlivé požadavky na systém jsou reprezentovány záznamy v databázi. Navržený systém umožňuje jednoduchou rekonfigurovatelnost, podporuje periodickou aktualizaci výsledků požadavků. Detailní popis řídicího systému je však mimo rámec tohoto příspěvku.

Popisovaná metodika byla testována na několika webových prezentací, od statických stránek s jednoduchou strukturou až po stránky dynamicky generované [5] s použitím apriorních informací i bez těchto informací. Nástroj pomohl i při částečné restrukturalizaci zmíněné webové prezentace odhalováním neplatných odkazů.

V praxi se při generování hierarchických map zatím vždy potvrdil předpoklad správné funkčnosti metody, kterým je strukturalizace webových prezentací.

5 LITERATURA

[1] ŘIMNÁČ, M. Mapa webové sítě. Katedra řídicí techniky, FEL, ČVUT, 2004. Diplomová práce.

[2] ŽIVNŮSTKA, J. Webové rozhraní pro mapu webové sítě. Katedra řídicí techniky, FEL, ČVUT, 2004. Diplomová práce.

[3] HENZINGER, M.R. Hyperlink Analysis for the Web. *IEEE Internet Computing*. 2001 (January/February). Stránky 45-50.

[4] ŘIMNÁČ, M.; ŠUSTA, R.; ŽIVNŮSTKA J. Automatické prohledávání webových stránek I. *Automatizace* 2004/4.

[5] DCE. Mapa webu. [on-line]. Katedra řídicí techniky, FEL, ČVUT. <*http://dce.felk.cvut.cz/pub/webmapa/cs/webmapa.html*>