International Journal of Engineering and Management Sciences (IJEMS) Vol. 2. (2017). No. 2.
DOI: 10.21791/IJEMS.2017.2.9.

Információ visszakeresésekor szöveg típusú állományoknál használt indexelési technikák

M. PRINCZ[1], F. KRAUSZ[2]

[1] Debreceni Egyetem, pmaria@unideb.hu
[2] Budapesti Műszaki és Gazdaságtudományi Egyetem , krausz.fannii@gmail.com

Absztrakt. A jellemzően nem, vagy kevéssé strukturált szöveges dokumentumokból való információ-visszakeresés az Internet elterjedésével vált különösen fontossá. Különböző módszerek állnak rendelkezésre. Ezek közül az egyik az invertált fájlok alkalmazása, amely a web tartalmában való kereséskor általános. E dolgozat a strukturálatlan szöveges dokumentumok közötti keresés eszközeit tekinti át.

Abstract. Information retrieval from a typically not structured text became greatly important already more than two decades with the appearance of the Internet. Various methods are available for prosperous information retrieval. One of them is the use of inverted files that the general when searching for specific web content. This paper provides an overview of search unstructured text documents.

Bevezetés

Szöveges dokumentumokban való keresés egyik módja, hogy a keresett szó, kifejezés, szövegrészlet előfordulását szekvenciálisan, karakterről karakterre ellenőrizzük az állományokban (pl. grep). A szekvenciális vagy online keresés nem igényli a szöveg előkészítését a keresésre, azonban csak kisebb, vagy gyorsan változó szöveges állományok esetén megfelelő, sok időt követel, és a dokumentumoknak is elérhetőnek kell lenniük a keresés végrehajtásakor. Az esetek többségében azonban a dokumentum-gyűjtemény nem érhető el a kereséskor, így a keresés csak a dokumentumokat leíró információk között történhet.

Szöveges állományokban való keresésnek egy másik lehetséges módja, ha a szöveg fölé adatstruktúrát – pl. indexet - építünk a keresés meggyorsítása érdekében. Az indexek létrehozása különösen nagy és nem túl gyakran változó szöveges gyűjtemények esetén célszerű - mint amilyen a web -, de természetesen a kétféle keresés kombinálható is.

Indexelés és keresés

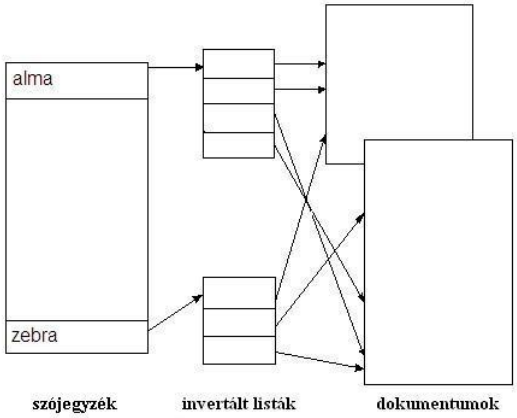
Több különböző indexelési technikát fejlesztettek ki, hogy hatékonyan támogassák a szöveges dokumentumok között az információ-visszakeresést. Ezek közé tartoznak az invertált fájlok, a szuffixek tömbje és a szignatúra fájlok. A három technika közül az invertált fájlok módszere a leggyakrabban alkalmazott technika, ezt használják a legnépszerűbb webes keresőrendszerek is. A szuffixek tömbje gyorsabb a kifejezések keresésénél, de fáradságos megépíteni és fenntartani. A

DOI: 10.21791/IJEMS.2017.2.9.

szignatúra fájlok népszerűek voltak az 1980-as években, de napjainkban az invertált fájlok módszere kiszorította ezt a technikát.

Invertált fájlok

A web tartalmában való keresésre az invertált fájlok alkalmazása vált általánossá, amely a kulcsszó alapú keresést teszi lehetővé. Az invertált fájlok struktúrájának két összetevője van: a szójegyzék vagy lexikon, valamint az előfordulások listája (invertált listák).



1. ábra: Az invertált fájlok felépítése

A szójegyzék tartalmazza a szöveggyűjteményben előforduló szavakat. A szójegyzék minden egyes szavához készül egy lista, amely rendezett sorrendben tartalmazza a szó minden előfordulását a dokumentumokban. Az előfordulás megadja a tartalmazó oldal azonosítóját és az oldalon belül az előfordulás helyének pozícióját (postings). Amennyiben nem szükséges feltüntetni minden egyes előfordulást az oldalon belül, a hely megadás csak az oldal azonosítóját tartalmazza, valamint opcionálisan az előfordulások számát az oldalon belül. Így egy adott szót tartalmazó dokumentum előkeresésére csak a szót kell megkeresni a lexikonban, és az előfordulások listája alapján a szót tartalmazó dokumentumok előkereshetők. A Boole lekérdezéseknél a keresési szavaknak megfelelő előfordulások metszetét vagy unióját kell venni, az értelmezésnek megfelelően.

Az indexelő technikák tökéletesítésével (pl változó hosszúságú azonosítók) az invertált fájlok méretét a szöveg méretének 30%-áig lehet csökkenteni, de további tömörítő technikákkal az index mérete a szöveg méretének 10%-ig is csökkenthető [3]. E megközelítés hátránya, hogy az invertált listákat dekódolni kell a visszakereséseknél, de az ilyen kitömörítések gyorsan végrehajthatók. [4]

Az előfordulások listájának helyigénye csökkenthető a blokk címzéses technika használatával. A szövegeket ekkor blokkokra osztják, és az előfordulások listája a pontos előfordulás felsorolása helyett csupán azon blokkok azonosítóját tartalmazza, amelyekben a szöveg feltűnik. E technika alkalmazásánál azonban a szövegnek a keresés idejében elérhetőnek kell lennie, ezért a webes keresések esetében nem használatos. Gyakran alkalmazzák azt a technikát, amikor a szavak előfordulásának feltüntetésére egy dokumentumon belül a pointerek helyett a szó sorszámát

DOI: 10.21791/IJEMS.2017.2.9.

tartalmazza az invertált lista. Ez a módszer elősegíti a kifejezések, és a közelségi, szomszédsági keresések megvalósítását is.

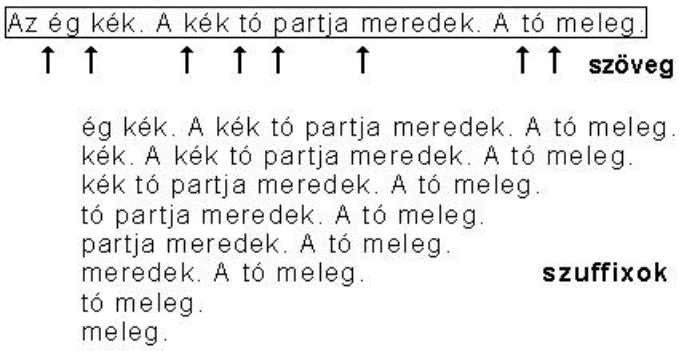
A szójegyzék tárolására hatékony struktúra a B-fa, de tömbök és hash táblák is alkalmasak rá. A szójegyzék általában kevesebb helyet igényel az invertált listáknál:

Szuffixek tömbje

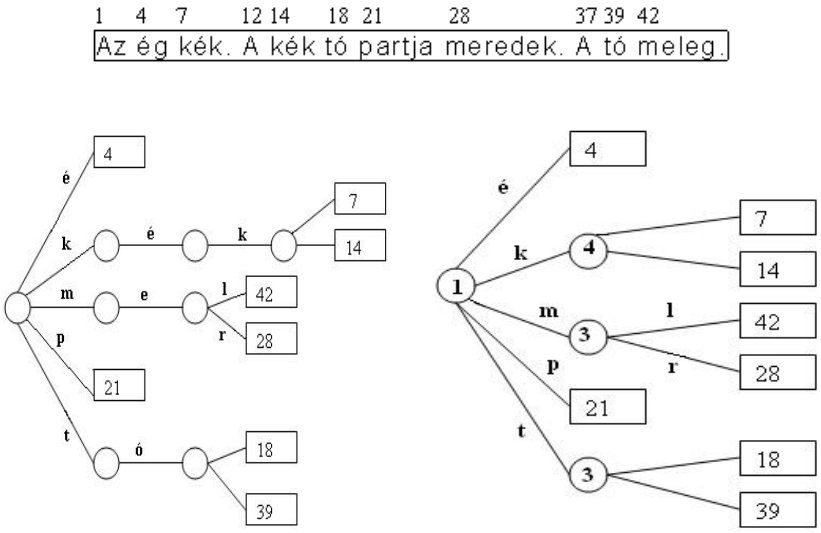
Az indexelésnek ez a típusa lehetővé teszi bonyolultabb lekérdezések (pl. kifejezések keresése) hatékony megvalósítását is, de hátránya a költséges megvalósíthatósága (különösen hosszú szöveges állományok esetén), s az a tény, hogy a szövegnek a lekérdezés idején olvashatónak kell lennie.

Ez a fajta indexelés a szöveget egy hosszú karaktersorozatnak tekinti. A szöveg minden pozíciója egy szuffixnek (a pozíciótól a szöveg végéig tartó karaktersorozatnak) tekinthető, s minden szuffix egyértelműen meghatározható a pozíciójával.

A szöveg nem minden pozícióját szükséges indexelni: elegendő csupán azon pozíciók indexelése, amelyek a visszakereshető szavak vagy kifejezések kezdő pozíciójával egyeznek meg.



2. ábra: Mintaszöveg az indexelési pontok jelölésével



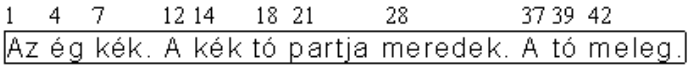
3. ábra: Szuffixfa és a tömörített szuffixfa a mintaszövegre

DOI: 10.21791/IJEMS.2017.2.9.

A szuffixfa a szöveg valamennyi szuffixa fölé épített adatstruktúra, ahol a szuffixokra mutató pointerek a fa végpontjaiban találhatók. A jobb térkihasználás végett a fa tömöríthető: azon csomópontokon átvezető utak, amelyekből csak egy további csomópontba lehet jutni, a 3. ábr szerint tömöríthetők.

A szuffixfával kapcsolatos probléma, hogy nagyon sok helyet igényel: Megvalósítástól függően egy-egy csomópont tárigénye 12 és 24 byte közötti érték, s ha csak a szavak kezdőpontját indexeljük, akkor is az eredeti szöveg többszöröse lehet az index nagysága (120% és 240% közötti érték [1]).

A szuffixtömb a szuffixfához hasonlóan működik, de jóval kevesebb a tárigénye. Ha a szuffixfa végpontjait (felülről lefelé haladva) egy tömbbe írjuk, akkor a szöveg szuffixai között abc szerint rendezve kereshetünk.



4	7	14	42	28	18	39

szuffixtömb

4. ábra: Szuffixtömb a mintaszövegre

A szuffixek tömbje tehát egy olyan tömb, amely lexikografikus sorrendben tartalmazza a szöveg szuffixaira mutató valamennyi pointert. A szuffixek tömbjében bináris kereséseket lehet végrehajtani, összehasonlítva a pointerek tartalmát a keresett kifejezéssel.

Szignatúra fájlok

Szignatúra fájlok egy valószínűségi módszer a dokumentumok indexelésére. A dokumentum minden egyes szavához egy szignatúrát kapcsolunk, ami azonos számú bitekből álló vektor. Ezeknek a hozzákapcsolt vektoroknak a tartalma véletlenszerű, előállításuk hash függvény alkalmazásával történik.

A dokumentum deszkriptorát úgy kapjuk, hogy a dokumentumban lévő szavak szignatúráinak bitenként a logikai VAGY kapcsolatát vesszük. Ha a dokumentum elég hosszú, akkor a szöveget blokkokra osztjuk, és a blokkban található szavak szignatúráit bitenként összeadva elkészítjük a blokkok szignatúráit is.

Ha ellenőrizni akarjuk, hogy egy *S* szó egy adott *D* dokumentumban (blokkban) megtalálható-e, akkor meg kell néznünk, hogy valamennyi bit, amely az *S* szignatúrájában be van állítva, be van-e állítva a *D* deszkriptorában is, azaz igaz-e, hogy $S \& D = S$, ahol & a bitenkénti AND (ÉS) kapcsolatot jelöli *S* és *D* szignatúrái között. Ha nem igaz az egyenlőség, akkor *S*–t nem tartalmazza a dokumentum. Ha igen, akkor *S* valószínűleg megtalálható a dokumentumban, de előfordulhat más kifejezések szignatúrájának kombinációjaként is, ezért nem lehet megmondani biztosan, hogy egy adott szó ténylegesen előfordul-e az adott dokumentumban.

DOI: 10.21791/IJEMS.2017.2.9.

Szó	**Szignatúra**
boci	1000 0000 0000 0100
tarka	0100 0010 0010 0000
lepke	1000 1000 0000 0001
kis	0000 0000 0001 0001
mese	0000 0001 1000 0010

Dokumentum	**Szöveg**	**Deszkriptor**
1	Boci, boci tarka	1100 0010 0010 0100
2	Tarka lepke, kis mese…	1100 1011 1011 0011

1. táblázat: Példa szignatúra fájlokra

A szignatúra fájlok lekérdezése néha azt eredményezi, hogy a kifejezés a dokumentumban található, noha a dokumentum ténylegesen nem tartalmazza azt. A szignatúra fájlok tervezésének legkényesebb része a hamis egyezések valószínűségét minél alacsonyabban tartani a szignatúrák hosszának a lehető legrövidebb megválasztása mellett.

Ha *S* a szignatúrában szereplő bitek száma, és *X* a szignatúrákban beállított bitek száma (értékük =1), akkor általában

1000 <= S <=10 000 és 6 <= X <= 20 javasolt [2, 3].

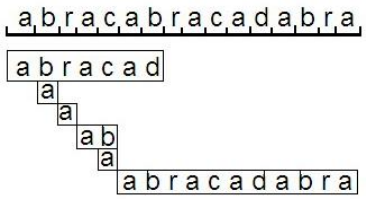
A szignatúra fájlok legfőbb előnye, hogy – az invertált fájlokkal szemben - nem igényelne, k lexikont, így a lekérdezés alatt nem igénylik a lexikon memóriában való jelenlétét sem. Szignatúra fájlok alkalmazására példa a TREC (Text REtrieval Conference) adatbázis.

Szekvenciális keresés

A szekvenciális keresés a szöveges dokumentumokban való kereséseknek egy olyan módja, amelynél semmiféle adatstruktúra nem épül a szövegre. Ez a módszer számos indexelő technika része.

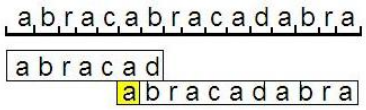
Karaktersorozatok pontos egyezésének vizsgálatára szolgál, alkalmazásával megadható egy hosszú szövegben az összes olyan pozíció, amelytől kezdve a minta előfordul. Klasszikus keresési probléma, amelyre számos elméleti megoldást kifejlesztettek már. Az algoritmusok közül a legegyszerűbb megoldás a nyers erő alkalmazása (Brut-force algorithm), azaz végig kell próbálni az összes lehetséges esetet, amikor a szöveg megegyezhet a mintával. Ez a szövegben az összes olyan pozíció vizsgálatát jelenti, amely a minta kezdőkarakterével megegyezik. Az alábbi példákban az abracadabra szó előfordulását keressük a szövegben:

DOI: 10.21791/IJEMS.2017.2.9.



5. ábra: Keresés a "Brut-force" algoritmus használatával

Átlagban egy kissé gyorsabb keresést tesz lehetővé a Knuth-Morris-Pratt algoritmus. Ez annyiban különbözik a "Brut-force" algoritmustól, hogy a megelőző egyezési vizsgálatok eredményét felhasználva kizárja eleve azokat a pozíciókat a szövegben, ahol a minta nem egyezhet.



6. ábra: Keresés a Knuth-Morris-Pratt algoritmus használatával

A szekvenciális keresés megvalósítására született számos modell közül az összehasonlításoknál az ismertetett két modell nyújtja a legjobb eredményt [1].

Összefoglalás

E dolgozat a strukturálatlan szöveges dokumentumok közötti keresés lehetőségeit tekintette át.

A dokumentumokból való információ-visszakeresés az Internet elterjedésével vált különösen fontossá. A szöveges dokumentum automatizált, gépi feldolgozásához szükség van a dokumentum logikai struktúráját is leírni. Ennek a megoldására fejlesztették ki az XML nyelvet, melyek segítségével olyan szöveges formátumokat állíthatunk elő, amelyek alkalmasak adatok strukturált, a számítógép által is értelmezhetően feldolgozható leírására.

Hivatkozások

[1] Baeza-Yates,R., Ribeiro-Neto,B., (1999) Modern Information Retrieval, Addison Wesley

[2] Blelloch,G. (1997) Algorithms in the Real World http://www.cs.cmu.edu/~guyb/real-world/indexing/

[3] Witten,I., Moffat,A., Bell,T. (1999) Managing Gigabytes: Compressing and Indexing Documents and Images, Morgan Kaufmann Publishing, San Francisco, ISBN 1-55860-570-3.

[4] Zobel,J.,Moffat,A.,Ramamohanarao,K.. (1998) Inverted Files Versus Signature Files for Text Indexing