Információ visszakeresés

Góth Júlia, Skrop Adrienn

2014

A tananyag a TÁMOP-4.1.2.A/1-11/1-2011-0104 "A felsőfokú informatikai oktatás minőségének fejlesztése, modernizációja" c. projekt keretében a Pannon Egyetem és a Szegedi Tudományegyetem együttműködésében készült.

1. fejezet

Bevezetés az információ-visszakeresésbe

1.1. Az információ-visszakeresés története

A XX. század vitathatatlanul az információ jegyében telt el. Ahogyan a XVIII. századot az ipari forradalom korának nevezik, úgy a XX. századra az információs forradalom megnevezés vetődött fel. A számítógépek mára már átkerültek az ipari és tudományos területekről a mindennapi életbe.

Napjaink rohanó világában az információ, és annak minél gyorsabb megtalálása nagy jelentőséggel bír kulturális, szociális és gazdasági szempontból is. Míg régen az emberek könyvtárakba jártak szakmai és szépművészeti irodalomért, addig mostanra, az Internet megjelenésével egyre jellemzőbb, hogy az információt a világhálón keresik.

A World Wide Web (WWW, világháló, vagy röviden web) közismert szó lett, és használata széles körben elterjedt. A világháló az információk kiapadhatatlan tárházává vált, amit az emberiség nagy része használ nap mint nap. Gyakorlatilag beépült mindennapi életünkbe. Innen tájékozódunk már nemcsak a legújabb tudományos eredményekről, a világban történt legfrissebb hírekről, de pl. a buszmenetrendről, időjárásról, receptekről is. A világhálót az emberiség egyik legnagyobb fordulópontjának nevezik. A World Wide Web (Web) az információ publikálásának legális és járható útja. A tapasztalatok azt mutatják, hogy a különböző jellegű információkat sok esetben hatékonyabban találhatjuk meg a Weben, mint nyomtatott forrásokban. A széleskörű alkalmazási lehetőségek, a felhasználóbarát grafikus környezet hatására napról napra több információ jelenik meg a Weben. Ebből a hatalmas dokumentumhalmazból meglehetősen nehéz kikeresni a számunkra megfelelő információt tartalmazó oldalakat. Az információ-keresés egyik és sokszor egyetlen módja a szörfözés (surfing): egy adott weboldalról indulva, oldalról-oldalra linkeket követünk, és azt reméljük, hogy előbb vagy utóbb megtaláljuk a keresett információt. Ez a módszer megfelelő, ha van elég időnk (és pénzünk) a felfedezésre. A szörfözés kevésbé hatékony,

ha gyorsan, vagy egymás után többször akarunk megtalálni egy speciális információt. A szörfözés tulajdonképpen nem más, mint barangolás (browsing) eszközök nélkül. A problémák leküzdésére nagyszámú keresőeszköz készült. A felhasználók szemszögéből a keresőeszköz is egy weboldal, még pedig egy speciális weblap: egy meghatározott helyre beírják a keresett kifejezést, ráklikkelnek egy ikonra, rövidebb hosszabb ideig várakoznak és a rendszer megad egy listát, amely többé-kevésbé megfelel a keresési feltételeknek.

Az internet és a világháló megjelenése, majd egyre szélesebb körű elterjedése mind gyakorlati, mind elméleti szempontból jelentős mértékben növelte az információ-visszakeresés (information retrieval, IR) fontosságát. Mára az információ-visszakeresés kikerült a könyvtártudomány kizárólagosságából és az információ-tudományok egyik főszereplőjévé vált.

1.1.1. Az információ-visszakeresés alapjai

Az információ-visszakeresés és annak megvalósítása korántsem tekinthető napjaink problémájának. Bizonyos értelemben már az ókorban megtalálhatjuk a gyökereket.

- *Tartalomjegyzék*: Az ókori görögök és rómaiak papirusztekercsre írtak. Már ők rájöttek arra, hogy a különböző szövegrészekben könnyebb eligazodni, ha jegyzéket készítenek, hogy mely fontos rész hol található. A tartalomjegyzék módszerét elsőként Valerius Soranus alkalmazta. Ez a módszer az inverz fájl struktúra volt, amelyet még napjainkban is széles körben alkalmaznak a számítástudomány, és az információ-visszakeresés területén.

  PÉLDA: Egy különböző költők verseit tartalmazó kötetben az egész könyvet végig kellene lapozni egy adott költő verseinek a megtalálásához, ha nem lenne tartalomjegyzék a végén.

- *Betűrend*: A szavak ilyen típusú rendezését görög tudósok alkalmazták először az alexandriai könyvtárban hogy a nagy mennyiségű görög irodalmi műveket könnyebb legyen rendszerezni. Az ábécé szolgáltatta az alapot a számítástudomány egyik legfontosabb és legszélesebb körben használt algoritmusához a rendezéshez.

  PÉLDA: Egy telefonkönyvben nagyon időigényes feladat lenne megtalálni az ismerősünk telefonszámát, ha a nevek nem lennének betűrendbe szedve.

- *Hierarchia*: Egy írott dokumentumban könnyebb eligazodni, ha fejezetekre, alfejezetkre tagoljuk. Római tudósok (pl. Valerius Maximus, Marcus Julius Frontinus, Aulus Gellius) osztották fel először műveiket alfejezetekre, és használtak fejlécet. Az írott szöveg hierarchikus szervezése az őse a számítástudomány, és a gráfelmélet egyik legfontosabb adatstruktúrájának a fastruktúrának.

  PÉLDA: Több száz receptet tartalmazó szakácskönyvben a diótorta receptjének megtalálása, nagyon időigényes feladat lenne, ha a szakácskönyv

nem lenne fejezetekre, alfejezetekre bontva a különböző ételtípusoknak megfelelően. Ennek segítésével elég a sütemények, és azon belül a torták alfejezetet megkeresni, és csak az ottlévő recepteket végignézni.

- *Index*: Napjainkban ha valaki egy adott szerző adott művét keresi a könyvtárban, akkor természetesen már számítógéphálózatot használ. Azonban a számítógépek megjelenése és elterjedése előtt, sőt bizonyos kisebb könyvtárakban még napjainkban is a felhasználó számára a legegyszerűbb megoldás a könyvtáros megkérdezése. Ezen kívül van még egy másik megoldás is az indexek használata. Ez azt jelenti, hogy van egy fiókokból álló szekrény, amit katalógusszekrénynek neveznek. A fiókokon betűrendben az ábécé betűi állnak. A szerző nevének kezdőbetűjéhez tartozó fiók kihúzása után a szerző művének katalóguskártyája gyorsan megtalálható, mivel a fiókon belül is betűrendben vannak a szerzők és műveik. A katalóguskártya tartalmazza a mű fontosabb paramétereit, és azt is, hogy merre található a könyvtárban.

  Az indexek használata is az ókori Rómába vezethető vissza, amikor minden papirusztekercshez csatoltak egy kis kártyát, ami tartalmazta a mű címét. Ezáltal minden papirusztekercs könnyen beazonosítható volt a polcról való levétel nélkül, így nem kellett a tekercseket feleslegesen mozgatni. Idővel az index a mű rövid kivonatát is tartalmazta. A papirusztekercseken nem volt sem oldalszámozás, sem sorszámozás.

  Ugyanakkor ha több példányt akartak csinálni egy műből, akkor azok mindig különböztek. A nyomtatás felfedezése tette lehetővé az oldalszámozást, és a teljesen egyforma másolatokat. Az első indexeknél a betűrendbe szedés még csak a szavak első betűje alapján történt. A teljes betűrendbe szedés csak a 18. században vált szabállyá. Az indexálás egy olyan eljárás ami megadja egy azonosítónak (szó, kifejezés, név, tárgy, kód), egy egységben (könyv, adatbázis) elfoglalt pontos helyét (fejezet, oldal, sor). Ez a módszer szolgált alapul a számítástudomány, adatbáziskezelés és az információ-visszakeresés egyik legfontosabb adatsruktúrájának az inverz fájl struktúrának.

1.1.2. Az információ-visszakeresés kialakulása

Az információ-visszakeresés kialakulásának legfontosabb mérföldkövei:

- A hidegháborús években az USA-ban került napvilágra először, hogy a kutatások hatékonyabbá tételének céljából az adatbázisokat hatásosabban lehetne kezelni információ-visszakeresési eljárásokkal. A kutatásra szánt pénzre akkoriban nem kellett sokáig várni, hiszen a Szovjetunió és az USA közti versengésben a tudomány szerepe jelentős volt. Miután az oroszok fellőtték az első műholdat, az USA a folyamatban levő kutatásokra jelentős pénzeket áldozott, köztük az információ-visszakeresésre is.

- A World Wide Web 1989-es megjelenése (ami T. Berners-Lee nevéhez fűződik) óta a visszakeresési eljárás már nem versengés célja, hanem a

Web használhatóságának a létkérdése. A világháló az interneten működő egymással összekötött számítógépeken tárolt elektronikus dokumentumok rendszere. Az eljárások és a különféle technikai megoldások a növekvő információáradat káosza ellen nyújtanak védőgátat, amelyet napról napra erősíteni kell.

- Az 1990-es évekre már több információ-visszakereső modell, algoritmus volt ismert, sok kutatás indult az információ-visszakeresés területén. Laboratóriumokban már több kísérleti információ-visszakereső rendszert használtak és teszteltek, és nagy mennyiségű kísérleti adat gyűlt össze.

- Az 1990-es évek végére megjelentek a világháló információ-visszakereső rendszerei, amelyeket web-keresőmotoroknak neveznek. Ezekben már megvalósították az információ-visszakeresés eredményeit, és lehetővé tették, hogy az elektronikus dokumentunok egyre növekvő halmazához mind több felhasználó férhessen hozzá.

- Ezáltal az internet, a világháló, a keresőgépek megváltoztatták a felhasználóknak az információ-visszakeresésről való fogalmát. A világháló megtanította őket arra, hogy a számítógépeken tárolt hatalmas, és továbbiakban is állandóan növekvő mennyiségű információhalmazban történő keresés sikeressége az információ-visszakereső módszerek hatékonyságán múlik. Informatikai cégek kezdtek újabb eljárások, keresőgépek fejlesztésébe mind nagyobb piaci sikereket elérve.

1.2. Adatbáziskezelő és információ-visszakereső rendszerek

Napjainkban az adatbáziskezelés akárcsak az információ-visszakeresés mindennapi tevékenységeink részévé vált, azonban a felhasználók számára gyakran e két dolog összemosódik, és ugyanazt értik mindkét fogalmon. Pedig a gyakorlatban van különbség köztük.

Az információt úgy kell reprezentálni és szervezni, hogy könnyen hozzáférhető legyen a felhasználók számára. A felhasználói információigény jellemzése azonban nem egyszerű probléma. A felhasználó rendszerint természetes nyelven fogalmazza meg információigényét, amit a legtöbb információ-visszakereső rendszer közvetlenül nem képes értelmezni. A felhasználónak a lekérdezési nyelv (query language) szintaktikájának megfelelően át kell alakítania információigényét olyan kérdéssé (query), keresőkifejezéssé (query expression), amely feldolgozható az információ-visszakereső rendszer számára. A legáltalánosabb esetben a transzformáció olyan kulcsszavakat (keyword), indexkifejezéseket (index term) eredményez, amelyek az információigény leírását tükrözik, összegzik. Az információ-visszakereső rendszer célja az, hogy olyan információt keressen az objektumok kollekciójában, ami az adott keresőkifejezés esetén hasznos, releváns (relevant) lehet a felhasználó számára. A gyakorlatban az információ-visszakeresést matematikai módszereken és formulákon alapuló algoritmusok valósítják meg.

A (tipikus, hagyományos) adatbáziskeresés jellemzője az információ- visszakereső rendszerekkel összehasonlítva a következő: a megtalált (visszakapott) dokumentumok azok, amely tartalmazzák a keresőkifejezést alkotó kulcsszavakat. Ez azonban a legtöbb esetben nem elegendő a felhasználó információigényének kielégítéséhez. Az információ visszakereső-rendszer felhasználóját jobban érdekli az, hogy a keresett témáról hasznos információt kapjon vissza, és ne a keresőkifejezést kielégítő adatot. Az adatbáziskeresés célja az előre definiált feltételeket kielégítő összes dokumentum visszakeresése. Egyetlen hibás dokumentum a több száz visszakeresett között teljes kudarcot jelenthet. Az információ-visszakereső rendszereknél lehetnek pontatlanok a visszakapott dokumentumok, és a kis hibák valószínűleg észrevétlenek maradnak. A különbség oka az, hogy míg az adatbáziskereső rendszerek (pl. relációs adatbázis) jól definiált struktúrájú és szemantikájú adatokat kezelnek, addig az információ-visszakereső rendszerek természetes nyelvű szövegeket, amelyek nem mindig jól strukturáltak és szemantikailag többértelműek lehetnek.

Az adatbáziskereső rendszerek csak az adatbázisrendszerek felhasználóinak nyújtanak megoldást, de általában nem oldják meg az információ-visszakeresést. Abból a célból, hogy az információ-visszakereső rendszerek hatékonyan tudják kielégíteni a felhasználók információigényét, értelmezniük kell a dokumentumok tartalmát, valamint rangsorolniuk (ranking) kell őket aszerint, hogy mennyire relevánsak a felhasználó által megadott keresőkifejezésre. A dokumentumok értelmezése magában foglalja szintaktikai és szemantikai információk kinyerését a szövegekből, és ezen információk felhasználását a relevanciáról való döntéshez. A relevancia központi fogalom az információ-visszakeresésben. Az információ-visszakereső rendszerek elsődleges célja az, hogy a felhasználói kérdésre lehetőleg az összes releváns, és a lehető legkevesebb irreleváns dokumentumot keresse vissza, vagyis nem minősíthetők csak az alapján, hogy egy kérdésre hány választ adnak.

1.3. Az információ-visszakeresés és a szövegbányászat

Az adatbányászat, és ennek egy speciális változata a szövegbányászat szintén napjaink egyik aktuális problémája. Felületes megfigyelők hajlamosak a szövegbányászatot a szöveges információ-visszakereséssel összemosni, pedig itt megint két különböző fogalomról van szó.

A szövegbányászat a struktúrálatlan szöveges állományokból történő ismeret kinyerésének tudománya. Olyan új ismeretek és információk kinyerése különböző dokumentumforrásokból amelyek előzőleg a felhasználó számára még ismeretlenek voltak. Ennek kinyerése gépi intelligencia felhasználásával történhet. Ezzel szemben az információ-visszakeresés során a felhasználó olyan információt keres az objektumok kollekciójában, ami számára hasznos, releváns. A felhasználónak van egy konkrét információigénye, tehát tudja, hogy mit akar találni. Az információigényét a keresőkérdésben próbálja megfogalmazni. A gyakorlatban az információ-visszakeresést matematikai módszereken és formulákon alapuló algoritmusok valósítják meg.

Míg az információ-visszakeresés során már meglévő információkra kívánunk kis időbefektetéssel rátalálni (nagy relevanciáju találatok által), addig a szövegbányászat során olyan tudásra, ismeretekre is szert kívánunk tenni, ami explicite nem volt benne a rendelékezésre álló dokumentum állományban (korpuszban), csak indirekt módon, rejtve.

1.4. Az információ-visszakeresés fogalma

1.4.1. Az információ-visszakeresés definíciójának alakulása

Az információ-visszakeresés fogalma, definíciója folyamatosan változott az évek során. A különböző szerzők műveikben mást és mást értenek információ-visszakeresés alatt. Azonban ezek a megfogalmazások lényegében nem tartalmaznak nagy különbségeket.

- Salton (1965)[**?**]: Az okos információ-visszakereső rendszer automatikusan megvalósítja a teljes tartalomanalízist, és az adott kérdésre a legmegfelelőbb dokumentumokat adja válaszul.

- Van Rijsbergen (1979)[**?**]: Az információ tárolása és visszakeresése egyszerű. Adott a dokumentunok egy halmaza, adott egy személy (aki ezt a dokumentumhalmazt használja), aki megfogalmazza a kérdést, amire a keresőkérdésben megfogalmazott információigényt kielégítő dokumentumok lesznek a válaszok.

- Salton (1986)[**?**] : Az automatikus szöveges visszakereső rendszer természetes nyelven megfogalmazott dokumentumokban keres és a felhasználó által feltett kérdésre csak bizonyos dokumentumokat ad válaszul. Az információ-visszakereső rendszer hatékonyságát a "pontosság" és a "felidézés" együttesen adják meg. A kérdés és a dokumentumot is olyan formában kell megadni, hogy a kívánt pontossági és felidézési szintet érjük el.

- Meadow, Boyce és Kraft (1999)[**?**] : az információ-visszakeresés azt jelenti, hogy az adatbázisban vagy az információhalmazban megtaláljuk a kívánt információt. Az információ visszaállítás nem ugyanaz, mint az információ-visszakeresés, ahogy egy teljes disk file másolása sem információ-visszakeresés.

- Berry és Browne (1999)[**?**]: a keresőgéppel szemben nagyok az elvárások. "Homályos" kérdéseket teszünk fel, és tömör jól megfogalmazott választ várunk. Általánosságban arra kérjük a számítógépet, hogy biztosítsa számunkra azt az információt, amit mi akarunk, ahelyett amit kérdeztünk.

- Baeza-Yates és Ribeiro-Neto (1999)[**?**]: Az információ-visszakereső rendszer célja, hogy a felhasználói kérdésre visszaadja az összes releváns választ, miközben a lehető legkevesebb nem-releváns választ ad még vissza.

- Belew (2000)[**?**]: Az információ-visszakeresés folyamata a következő: adott a felhasználói igény, amit egy kérdés formájában küld el a felhasználó egy adott dokumentumhalmazba, majd néhány eljárás alkalmazása után a doumentumok egy részhalmazát a felhasználó válaszként visszakap.

- Baeza-Yates (2003)[**?**]: Az információ-visszakeresés célja olyan rendszerek modellezése, tervezése és implementálása, amelyek gyors és hatékony tartalom-alapú hozzáférést biztosítanak nagy mennyiségű információhalmazhoz. Az információ-visszakereső rendszer célja mebecsülni a relevanciát a visszakapott információ és a keresőkérdésben megfogalmazott felhasználói információ között.

1.4.2. Az információ-visszakeresés definíciója

Az információ-visszakeresés a felhasználók információigényét kielégítő információ számítógépes reprezentálásával, tárolásával, szervezésével, visszakeresésével és annak az információigényre vonatkozó referenciafokának (hatékonyság) mérésével foglalkozik.[**?**]

Felhasználó lehet például:

- kutató
- turista
- háziasszony
- ingatlanügynök.

Felhasználói információigény lehet például:

- egy bizonyos tématerülethez tartozó folyóiratcikkek halmaza,
- utazási irodák nyári ajánlatai,
- diós sütemények receptjei,
- eladó veszprémi ingatlanok.

Az információt objektumokban - hagyományos nevükön dokumentumokban - keressük.

Az objektum lehet:

- szöveg
- kép
- hang
- multimédia.

A dokumentumok kollekcióját adatbázisokban, számítógép diszkeken tároljuk. Az információigényt keresőkérdésként adjuk meg az információ-visszakeresést megvalósító számítógépprogram által definiált formában.

A visszakeresés azt jelenti, hogy egy adott kérdésre (keresőkérdésre) keressük a választ egy adott objektumhalmazban. A gyakorlatban a visszakeresést matematikai módszereken és képleteken alapuló algoritmusok valósítják meg az objektumok és kérdések számítástechnikailag megfelelő reprezentációit alkalmazva.

A kérdésre válaszként visszaadott információ lehet például:

- folyóiratcikk,
- weboldal.

Ezek felhasználásával az információ-visszakeresés az alábbi formában adható meg:

1. Definíció. *Legyen az információ-visszakeresés(IR)az alábbi:*

$$IR = (U, IN, Q, O) \rightarrow R,$$

ahol

- U = *felhasználó (user),*
- IN = *információigény (information need),*
- Q = *keresőkérdés (query),*
- O = *keresendő objektumok halmaza,*
- R = Q *keresőkérdésre válaszként visszaadott dokumentumok halmaza.*

Az információigény (IN) mindig több mint amennyi a Q keresőkérdésben megfogalmazódik. Ez az információtöbblet felhasználó specifikus, és a felhasználó számára olyan nyilvánvaló ismereteket is jelent, ami a számítógépes információ-visszakereső rendszer számára nem az. Ezáltal az információigény az alábbi formában adható meg:

2. Definíció. *A felhasználói információigény (IN) az alábbi:*

$$IN = (Q, I),$$

ahol I azt a felhasználó- specifikus információtöbbletet jelenti, amely nem fogalmazódik meg a Q keresőkérdésben.

Ezáltal az információ-visszakeresés szigorúbb definíciója a következő:

3. Definíció. *Az információ-visszakeresés az információigény (IN) és az O keresendő objektumok közti releváns kapcsolat megtalálása, formálisan:*

$$IR = \Re(O, IN) = \Re(O, (Q, I))$$

.

A megfelelő kapcsolat megtalálása azt jelenti, hogy olyan objektumokat nyerünk, amelyek a Q keresőkérdésre válaszok, és kielégítik az I implicit információigényt is.

Az információ-visszakereső rendszerek célja az, hogy olyan információt keressenek az objektumok halmazában, ami adott keresőkifejezés esetén hasznos, releváns lehet a felhasználók számára. A gyakorlatban az információ-visszakeresést különböző matematikai módszereken és formulákon alapuló algoritmusok valósítják meg.

2. fejezet

Klasszikus információ-visszakereső módszerek

2.1. Információ-visszakereső modellek

Az információ-visszakeresésben négy alapvető elemet különböztetünk meg:

- dokumentum,
- kérdés,
- relevancia és a
- visszakeresés.

Attól függően, hogy a dokumentumokat, a kérdéseket és a visszakeresést hogyan modellezzük (reprezentáljuk), különböző formális információ-visszakereső modellt (information retrieval models) különböztetünk meg. A visszakeresés nem befolyásolja a dokumentumok között fellépő kapcsolatokat. A relevanciát bizonyos előzetes információk alapján határozzák meg, vagy egyáltalán nem is veszik figyelembe.

Az információ-visszakeresés első modelljei matematikai módszereken alapulnak. Ezekben a modellekben a visszakeresés a kérdés és a dokumentum (a gyakorlatban ezek reprezentációja) 'távolságának' (hasonlóság, illeszkedés) matematikai mérésén alapul. E modellek különböző speciális eseteit alkalmazzák a jelenlegi kereskedelmi információ-visszakereső rendszerek. Ezeket a modelleket más néven klasszikus információ-visszakereső modelleknek nevezik. Ezek előnye, hogy könnyű implementálási lehetőséget nyújtanak, ami a gyakorlati alkalmazásban nagyon fontos.

A klasszikus információ-visszakeresési modellek a következők:

- Boole modell (Boolean Model), ami a matematikai logikán és a halmazelméleten alapszik,
- Vektortér modell (Vector Space Model), aminek a lineáris algebra szolgál alapul,
- Valószínűségi modell (Probabilistic Model), ami a valószínűségszámítás, és a Bayes statisztikán alapszik.

Az 1980-as, 90-es években más alapelveken nyugvó újabb modelleket is kifejleszettek. Így a hagyományos, vagy klasszikus modellek mellett megjelentek a nem-klasszikus, illetve az alternatív módszerek is [**?**].

Nem-klasszikus modellek a következők:

- Információs logika (information logic) alapú információ-vissakereső modell [**?**].
- Szituációelmélet alapú (situation theory) információ-visszakereső modell.
- Kölcsönhatás alapú információ-visszakereső modell (associative interaction model), ami a kvantummechanika Koppenhágai értelmezésén alapszik [**?**].

Alternatív modellek közül néhány:

- Klaszter alapú információ-visszakereső modell.
- Fuzzy alapú információ-visszakereső modell (a fuzzy halmazelméletből, fuzzy modellből indul ki)[**?**].
- Mesterséges neurális hálózat (artificial neural network) alapú információ-visszakereső modell.
- Genetikus algoritmus (genetic algorithm) alapú információ-visszakereső modell.
- Tudásbázis (knowledge base) alapú információ-visszakereső modell.

Bár ezek a modellek különböznek egymástól, azonban bizonyos modellek közös tulajdonságokkal is rendelkeznek:

- A dokumentumokat az őket leíró, vagy bennük megtalálható indexkifejezésekkel adjuk meg, és ezen indexkifejezések előfordulási száma alapján (súlyszámítással) reprezentálható numerikusan egy dokumentum.
- A kérdés is mint egy adott dokumentum tekinthető, amit a benne szereplő kifejezések határoznak meg.
- A dokumentum tartalmán kívül egy weboldal fontosságát az is befolyásolja, hogy a weboldalak egymással linkeken keresztül összekötöttek.

- A kérdés és a dokumentum hasonlósági fokát a súlyszámítás és az összekötöttség határozza meg.

Attól függően különböztetünk meg többféle formális információ-visszakereső modellt hogy a dokumentumokat, kérdéseket és a visszakeresést hogyan modellezzük. Továbbá a dokumentumokat reprezentáló indexkifejezések megadására is többféle módszer létezik.

Például a Boole-féle módszer a hagyományos halmazelméleten és a matematikai logikán alapszik.

2.2. Boole modell

Az első és széles körben alkalmazott klasszikus modell a Boole modell [**?**] . Napjainkban gyakorlatilag minden kereskedelmi keresőrendszer ezt használja, ill. ezen alapszik. Boole logikára (Boolean Logic) és klasszikus halmazelméletre (classical Set Theory) épül abban az értelemben, hogy mind a dokumentumokat, mind a felhasználók kérdéseit szavaknak (kifejezéseknek) tekinti. A visszakeresés azon alapul, hogy a dokumentumok tartalmazzák-e vagy sem a keresőkérdést alkotó kifejezéseket.

A Boole modell formális leírása a következő:

Adottak:

$T = \{t_1, \ldots, t_i, \ldots, t_n\}$ indexkifejezések, amelyek jellemzik, leírják a dokumentumokat, valamint a

$D = \{D_1, \ldots, D_j, \ldots, D_m\}$ dokumentumok, ahol $D_j \in \wp(T)$.

A Boole modellben a dokumentumokat formálisan indexkifejezések halmazának tekintjük. A valóságban a dokumentum és a reprezentációja két külön entitás, matematikailag viszont ekvivalensnek tekinthetők, hiszen a dokumentumokat a rájuk jellemző indexkifejezésekkel reprezentáljuk, valamint visszakeresés során is a dokumentumok reprezentációját használjuk és nem magukat a dokumentumokat.

A Q keresőkérdés egy Boole kifejezés, amely konjunktív normál formában a következő módon adható meg:

$$Q = \bigwedge_{k \in K} (\bigvee_{i \in I} \Theta_i), \Theta_i \in \{t_i, \neg t_i\}$$

A keresőkérdést azért célszerű normál alakban felvenni a matematikai modellben, mert bármely Boole-kifejezés átalakítható normál alakká.

A Q kérdésre válaszként visszaadott dokumentumokat a logikai műveleteknek megfelelő halmazműveletek adják meg az alábbi formában:

$\bigcap$ (metszet) megfelel a $\bigwedge$ (logikai ÉS) kapcsolatnak,

$\bigcup$ (unió) megfelel a $\bigvee$ (logikai VAGY) kapcsolatnak.

A visszakeresés két fő lépésből áll, amelyek a következők:

1. Meghatározzuk a dokumentumok azon S_i halmazát, amely tartalmazza vagy nem a t_i kifejezést: ($\Theta_i = t_i$, ha $t_i \in D_j$)

$$S_i = \{D_j | \Theta_i \in D_j\}$$

2. A válaszként visszaadandó objektumokat a következő halmazművelettel határozzuk meg:

$$\bigcap_{k\in K}(\bigcup_{i\in I} S_i)$$

PÉLDA (BOOLE-féle kérdésre, válaszadásra):
Az A, B, C, D indexkifejezések esetén a Q keresőkérdés megadására néhány példa:

- $Q_1 = \neg B \bigvee \neg A$
- $Q_2 = (A \bigvee C) \bigwedge (B \bigvee C)$
- $Q_3 = (A \bigvee (B \bigwedge C)$

A visszakeresés azon alapszik, hogy az adott dokumentum tartalmazza-e, avagy sem a kérdésben megadott keresőkifejezéseket. Ezáltal a visszakeresés az alábbi módon adható meg:

1. A dokumentumok S_i halmazát úgy nyerjük, hogy tartalmazzák-e vagy nem az adott indexkifejezést.

   A indexkifejezésre: $S_i = \{D | A \in D\}$

   A indexkifejezés negáltjára: $S_i = \{D | Anot \in D\}$

2. A logikai műveleteknek megfelelő halmazműveletek adják meg a Q kérdésre a válaszokat az alábbiakban:

   $\bigcap$ (metszet) megfelel a $\bigwedge$ (logikai ÉS) kapcsolatnak,

   $\bigcup$ (unió) megfelel a $\bigvee$ (logikai VAGY) kapcsolatnak.

Tekintsük a

$$Q_3 = A \bigvee (B \bigwedge C)$$

keresőkérdésre a választ:
S_1 adja az A kifejezésre az eredménydokumentumokat, azaz azokat a dokumentumokat, amelyek tartalmazzák a A indexkifejezést,
S_2 adja a B kifejezésre az eredménydokumentumokat, azaz azokat a dokumentumokat, amelyek tartalmazzák a B indexkifejezést,
S_3 adja a C kifejezésekre az erdménydokumentumokat, azaz azokat a dokumentumokat, amelyek tartalmazzák a C indexkifejezést.
Tehát a Q_3 keresőkérdésre válaszként adott dokumentumok:

$$S_1 \bigcup (S_2 \bigcap S_3)$$

PÉLDA (BOOLE-modellre):
Legyen O az eredeti (azaz valós) objektumok halmaza az alábbi:
$O = \{O_1, O_2, O_3\}$

- $O_1 =$
  Még nyílnak a völgyben a kerti virágok,
  még zöldell a nyárfa az ablak előtt,
  de látod amottan a téli világot?
  Már hó takará el a bérci tetőt.
- $O_2 =$
  Fenyő ága Hósubában,
  Mire vársz a Hófúvásban?
  Hideg az a Kristálybunda,
  Gyere haza Kis házunkba.
- $O_3 =$
  Fekete pont fehér fákon.
  Varjú károg:
  Fázom, fázom.

Legyen T indexkifejezések halmaza az alábbi: $T = \{t_1, t_2, t_3, t_4, t_5, t_6\}$, ahol:

- $t_1 =$ virág,
- $t_2 =$ tél,
- $t_3 =$ hó,
- $t_4 =$ fenyő,
- $t_5 =$ bunda,
- $t_6 =$ varjú.

A $T = \{t_1, t_2, t_3, t_4, t_5, t_6\}$ indexkifejezéssekkel az
$O = \{O_1, O_2, O_3\}$ objektumok az alábbi formában reprezentálhatók:
$D = \{D_1, D_2, D_3\}$, ahol

- $D_1 =$\{virág, tél, hó\}$= \{t_1, t_2, t_3\}$
- $D_2 =$\{hó, fenyő, bunda\}$= \{t_3, t_4, t_5\}$,
- $D_3 =$\{varjú\}$= \{t_6\}$.

Legyen Q keresőkérdés az alábbi: hó $\bigwedge$ fenyő, azaz

$$Q = t_3 \bigwedge t_4$$

Ekkor az
$S_3 = \{D_1, D_2\}$, azon dokumentumok halmaza, amelyek tartalmazzák a
$t_3 =$ hó indexkifejezést,

$S_4 = \{D_2\}$, azon dokumentumok halmaza, amelyek tartalmazzák a
$t_4 =$ fenyő indexkifejezést.
A Q keresőkérdésre válaszként adott dokumentumhalmaz a következő:

$$S_3 \bigcup S_4 = \{D_2\}$$

Tehát a Q keresőkérdésre a válasz az O_2 objektum lesz.

2.3. Vektortér modell

Attól függően, hogy a dokumentumokat, kérdéseket és a visszakeresést hogyan modellezzük, többféle formális információ-visszakereső modellt különböztetünk meg, amelyek közül az egyik leginkább elterjedt és használatos modell a vektortér modell [**?**], [**?**], [**?**], [**?**]. A modell neve onnan származik, hogy a dokumentumokat és a kérdéseket számok sorozataként fogjuk fel, mintha vektorok volnának. A visszakeresés azon alapszik, hogy a kérdés és a dokumentum vektor mennyire hasonlít egymásra.

Legyen
$D = \{D_1, \ldots, D_j, \ldots, D_m\}$ dokumentumok véges halmaza, ahol $D_j \in \wp(T)$, valamint
$T = \{t_1, \ldots, t_i, \ldots, t_n\}$ dokumentumokat leíró indexkifejezések véges halmaza.

Minden egyes D_j dokumentumhoz hozzárendelhető egy n hosszúságú valós számokból álló $\mathbf{v}_j$ vektor az alábbi formában:

$$\mathbf{v}_j = (w_{ij})_{i=1\ldots n} = (w_{1j}, \ldots w_{ij}, \ldots w_{nj}),$$

ahol általában $0 \leq w_{ij} \leq 1$ (w_{ij} normalizált)

A $\mathbf{v}_j$ vektort másnéven a D_j dokumentum súlyvektorának nevezik, ahol a w_{ij} súly azt mutatja meg, hogy a t_i indexkifejezés mennyire határozza meg a D_j dokumentumot.

A vektortér modell esetében a súlyvektorokból megadható a TD (term-by document) kifejezés-dokumentum mátrix, amelynek elemei a súlyok:

TD= $(w_{ij})_{n\times m}, i = 1, \ldots, n, j = 1, \ldots, m$, ahol:
w_{ij} a t_i indexkifejezés súlya a D_j dokumentumban
m: a dokumentumok száma (mátrixnak m oszlopa van)
n: az indexkifejezések száma (a mátrixnak n sora van)
TD:=

$$\begin{pmatrix} w_{11} & \ldots & w_{1j} & \ldots & w_{1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{i1} & \ldots & w_{ij} & \ldots & w_{im} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{n1} & \ldots & w_{nj} & \ldots & w_{nm} \end{pmatrix}$$

Az indexkifejezések, és a hozzájuk tartozó súlyok meghatározása nagyon nehéz elméleti (nyelvészeti, szemantikai), és gyakorlati probléma. Az információ-visszakeresésben a dokumentumokat indexkifejezésekkel reprezentálhatjuk, amelyeket vagy automatikusan a dokumentumból nyerünk, vagy szakértők határoznak meg. Attól függően, hogy melyik módszert használjuk indexkifejezések megállapítására, más és más hatékonyságú lesz az információ-visszakereső rendszer. A vektortér modell abból a feltételezésből indul ki, hogy a szövegben találhatók meg a szöveget leíró indexkifejezések. Továbbá feltételezi azt is, hogy a szövegben előforduló kifejezések gyakorisága meghatározza, hogy mely szavak legyenek indexkifejezések [**?**]. A szöveget leíró indexkifejezésekhez tartozó súlyok meghatározásának, majd a visszakeresés megvalósításának lépései a következők:

1. Szövegfeldolgozási műveletek során a dokumentumokból az indexkifejzések kinyerése.

2. Az indexkifejezések felhasználásával minden D_j dokumentumra minden w_{ij} súly meghatározása, amire többféle számítási módszer is létezik [**?**]. Ezek közül a leggyakrabban használt módszerek az alábbiak:

   - Bináris súlyszámítás (az adott indexkifejezés benne van, vagy nincs benne az adott dokumentumban):
     $w_{ij} = 1$, ha $t_i \in D_j$, és
     $w_{ij} = 0$, ha $t_i nem \in D_j$,
   - f (előforulási gyakoriság): a t_i indexkifejezés előfordulásainak a száma a D_j dokumentumban
     $$w_{ij} = f_{ij}$$
   - maxNorm: az előfordulási gyakoriság maxnormalizált változata:
     $$w_{ij} = \frac{f_{ij}}{\max_k f_{kj}}$$
   - tf-idf (inverz dokumentum gyakoriság):
     $$w_{ij} = -\log_2 \frac{df_i}{m}$$
     , ahol a df_i azon dokumentumok száma, amelyekben a t_i indexkifejezés előfordul.
   - tfn (normalizált előfordulási gyakoriság):
     $$w_{ij} = \frac{f_{ij}}{\sqrt{\sum_{i=1}^{n} f_{ij}^2}}$$

3. Keresőkérdés megadása. A felhasználó által feltett Q_k keresőkérdésből is egy $\mathbf{v}_k$ vektor generálható az előbbiekhez megegyező módon. A kérdés is úgy tekinthető, mint egy dokumentum.

4. A visszakeresés megvalósítása. A visszakeresés egy hasonlóság meghatározásán alapul. A dokumentumoknak megfelelő $\mathbf{v}_j$ vektorokat "összevetjük" a $\mathbf{v}_k$ kérdés vektorral (hasonlóság-mérés). Ennek eredménye sík. Ha ez a mérési eredmény egy küszöbértéknél nagyobb, akkor $\mathbf{v}_j$ vektorral azonosított dokumentum válasz a kérdésre.

$$s_{jk} = s(\mathbf{v}_j, \mathbf{v}_k) > K$$

Hasonlóság mérésére néhány példa:

- Pont (skalár szorzat):

$$s_{jk} = (\mathbf{v}_j, \mathbf{v}_k) = \sum_{i=1}^{n} w_{ij} \cdot w_{ik}$$

- Koszinusz mérték (c_{jk}):

$$s_{jk} = c_{jk} = \frac{(\mathbf{v}_j, \mathbf{v}_k)}{\|\mathbf{v}_j\| \cdot \|\mathbf{v}_k\|} = \frac{\sum_{i=1}^{n} w_{ij} \cdot w_{ik}}{\sqrt{\sum_{i=1}^{n} w_{ij}^2 \cdot \sum_{i=1}^{n} w_{ik}^2}}$$

- Dice együttható (d_{jk}):

$$s_{jk} = d_{jk} = \frac{2(\mathbf{v}_j, \mathbf{v}_k)}{\sum_{i=1}^{n} w_{ij} + w_{ik}}$$

- Jaccard együttható (J_{jk}):

$$s_{jk} = J_{jk} = \frac{\sum_{i=1}^{n} w_{ij} \cdot w_{ik}}{\sum_{i=1}^{n} \frac{w_{ij}+w_{ik}}{2^{w_{ij} \cdot w_{ik}}}}$$

5. Találati lista megadása, ami a válaszként megadott dokumentumokat tartalmazza a hasonlósági értékük alapján felállított csökkenő sorrendben.

PÉLDA:

Legyen $D = \{D_1, \ldots, D_j, \ldots, D_m\}$ dokumentumok véges halmaza, ahol $D_j \in \wp(T)$, valamint a

$T = \{t_1, \ldots, t_i, \ldots, t_n\}$ dokumentumokat leíró indexkifejezések véges halmaza.

TD mátrix az alábbiakban adható meg:

TD= $(w_{ij})_{n \times m}, i = 1, \ldots, n, i = 1, \ldots, m$,ahol:

w_{ij} a t_i indexkifejezés súlya a D_j dokumentumban

m: a dokumentumok száma (mátrixnak m oszlopa van)

n: az indexkifejezések száma (a mátrixnak n sora van)

f_{ij} a t_i indexkifejezés előfordulási gyakorisága a D_j dokumentumban

F_i azon dokumentumok száma, amelyekben előforul a t_i indexkifejezés

TD:=

$$\begin{pmatrix} w_{11} & \dots & w_{1j} & \dots & w_{1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{i1} & \dots & w_{ij} & \dots & w_{im} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{n1} & \dots & w_{nj} & \dots & w_{nm} \end{pmatrix}$$

Legyen

m=7 dokumentum (mátrixnak 7 oszlopa van), azaz D = $\{D_1, D_2, D_3, D_4, D_5, D_6, D_7\}$ és

n=9 dokumentumot leíró indexkifejezés (a mátrixnak 9 sora van), azaz T = $\{t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9\}$,

a dokumentumok álljanak a következő indexkifejezésekből:

$D_1 = \{t_6, t_9\}$

$D_2 = \{t_1, t_2, t_5\}$

$D_3 = \{t_2, t_5, t_8\}$

$D_4 = \{t_1, t_4, t_6, t_8, t_9\}$

$D_5 = \{t_1, t_7\}$

$D_6 = \{t_3, t_7\}$

$D_7 = \{t_1, t_2\}$

- Bináris súlyszámítási sémát alkalmazva:

  $w_{ij} = 1$, ha $t_i \in D_j$, és $w_{ij} = 0$, ha $t_i nem \in D_j$,

  (az adott indexkifejezés benne van, vagy nincs benne az adott dokumentumban) ekkor a TD mátrix a következő:

  TD:=

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

  ,ahol $w_{11} = 0$ azt jelenti, hogy a t_1 indexkifejezés egyszer sem fordul elő a D_1 dokumentumban.

A Q keresőkérdés legyen az alábbi:$Q = \{t_2, t_5, t_6, t_7, t_8\}$ Q:=

$$\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

- Hossznormalizált súlyszámítási sémát alkalmazva:

$$w_{ij} = \frac{f_{ij}}{\sqrt{\sum_{k=1}^{n} f_{kj}^2}}$$

TD:=

$$\begin{pmatrix} 0 & 0.577 & 0 & 0.447 & 0.707 & 0 & 0.707 \\ 0 & 0.577 & 0.577 & 0 & 0 & 0 & 0.707 \\ 0 & 0 & 0 & 0 & 0 & 0.707 & 0 \\ 0 & 0 & 0 & 0.447 & 0 & 0 & 0 \\ 0 & 0.577 & 0.577 & 0 & 0 & 0 & 0 \\ 0.707 & 0 & 0 & 0.447 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.707 & 0.707 & 0 \\ 0 & 0 & 0.577 & 0.447 & 0 & 0 & 0 \\ 0.707 & 0 & 0 & 0.447 & 0 & 0 & 0 \end{pmatrix}$$

A Q keresőkérdés legyen az alábbi: Q:=

$$\begin{pmatrix} 0 \\ 0.447 \\ 0 \\ 0 \\ 0.447 \\ 0.447 \\ 0.447 \\ 0.447 \\ 0 \end{pmatrix}$$

A hasonlósági értékek (Koszinusz, Jaccard, Dice) így alakulnak:

Dokumentumszám	Koszinusz	Jaccard	Dice
D_1	0.316	0.092	0.174
D_2	0.516	0.142	0.26
D_3	0.775	0.224	0.39
D_4	0.4	0.094	0.178
D_5	0.316	0.092	0.174
D_6	0.316	0.092	0.168
D_7	0	0	0

Tehát a válaszként visszaadott találati lista:

Koszinusz mérték esetében: D_3, D_2, D_4, D_1, D_5, D_6

Jaccard mérték esetében: D_3, D_2, D_4, D_1, D_5, D_6

Dice mérték esetében: D_3, D_2, D_4, D_1, D_5, D_6

Az adott példában mindegyik hasonlósági mérték esetében ugyanaz lesz a rangsor, tehát ugyanazon a dokumentumok lesznek a válaszok. Egy adott küszöbérték megadásával a találati lista hossza csökkenthető.

2.4. Valószínűségi modell

A valószínűségi modellek (probabilistic models) feltételes valószínűségi értékek figyelembevételével végzik a dokumentumok felett végrehajtott információ- visszakeresési műveleteket [**?**]. A valószínűségi modellek kidolgozásának indíttatása az volt, hogy a ugynazon dokumentum esetében a különböző szakértők által megadott indexkifejezések sok esetben mások.

Amikor a felhasználó megad egy keresőkérdést, amelyre a releváns dokumentumok listáját várja válaszként, akkor a dokumentumok egy részhalmaza lesz az a találati lista, amely csak és kizárólag a releváns dokumentumokat tartalmazza. Ezt nevezzük ideális találati listának. Az ideális találati listát (válaszhalmazt) ismerve könnyedén meghatározhatnánk azt az ideális lekérdezést, amely ezt az ideális dokumentum részhalmazt eredményezi. A probléma az, hogy nem ismerjük teljes mértékben az ideális válaszhalmazt. A lekérdezéskor azonban nem ismerjük teljes mértékben a dokumentumhalmazt, így a kezdeti lekérdezés egy kezdeti próbálkozásként fogható fel. Ezt követően egy iterációs folyamat során lehet eljutni a megfelelő válaszhalmazhoz. Ráadásul minden felhasználó más és más kezdeti keresőkérdést használ ugyanazon dokumentumok megtalálására. Ha a felhasználó kiválasztja a számára releváns dokumentumokat, akkor az információ-visszakereső rendszer tovább tudja finomítani a keresést az ideális találati lista felállítására. A rendszer a felhasználó által adott visszajelzésekből tudja finomítani ezt a listát. Ez történik az iterációs lépések során.

A valószínűségi modell azt feltételezi, hogy a relevancia valószínűsége kizárólag a keresőkérdéstől, és a dokumentumhalmaztól függ. A találati listába kerülő dokumentumoknál a relevancia becslése során azt jósoljuk, hogy az adott keresőkérdés mellett a dokumentum releváns lesz-e a felhasználó számára. A kihívás abban rejlik, hogy hogyan számítsuk ki a relevancia becslésekor annak valószínűségét. A valószínűségi modell esetében az információ-visszakereső

rendszer minden dokumentumhoz egy valószínűségi értéket rendel annak megfelelően, hogy mennyire jósolja annak valószínűségét, hogy az majd a felhasználó számára releváns lesz. A visszaadott dokumentumokat megjeleníti a felhasználónak, és ő dönt, hogy melyik releváns, és melyik nem. Ezt hívják relevancia visszacsatolásnak. Majd ezután a rendszer már egy pontosabb, a felhasználó preferenciáira építő listát képes készíteni a relevánsnak ítélt dokumentumokról.

A relevancia becslésekor a valószínűség számítására többféle megközelítés létezik.

Az egyik modell matematikai leírása a következő:

Adottak:

- $D = \{D_1, \ldots, D_j, \ldots, D_m\}$ dokumentumok véges halmaza,
- $T = \{t_1, \ldots, t_i, \ldots, t_n\}$ index kifejezések véges halmaza
- bináris súlyszámítási sémát alkalmazunk, tehát csak az fontos, hogy egy adott indexkifejezés szerepel-e egy adott dokumentumban vagy sem:
- felhasználói kérdés q.

A valószínűségi modell megbecsüli annak a valószínűségét, hogy a felhasználó számára az adott dokumentum releváns-e.

A hasonlósági mérték az alábbi formában adható meg:

$$S(q|d_j) = \frac{P(R|d_j)}{P(I|d_j)}, ahol:$$

$P(R|d_j)$ annak a valószínűsége, a d_j dokumentum releváns-e a kérdésre

$P(I|d_j)$ annak a valószínűsége, a d_j dokumentum irreleváns-e a kérdésre.

A hasonlósági mérték a Bayes-szabály alkalmazásával az alábbi módon írható fel:

$$S(q|d_j) = \frac{P(R|d_j)}{P(I|d_j)} = \frac{\frac{P(d_j|R)\cdot P(R)}{P(d_j)}}{\frac{P(d_j|I)\cdot P(I)}{P(d_j)}} = \frac{P(d_j|R)\cdot P(R)}{P(d_j|I)\cdot P(I)}, ahol:$$

$P(R)$ jelöli a priori valószínűséget, azaz annak a valószínűségét, hogy a véletlenül kiválasztott d dokumentum releváns-e az adott kérdésre,

$P(I)$ jelöli a priori valószínűséget, azaz annak a valószínűségét, hogy a véletlenül kiválasztott d dokumentum irreleváns-e az adott kérdésre,

$P(d_j|R)$ annak a valószínűsége, hogy ha a releváns dokumentumot visszakaptuk válaszként, akkor az d_j.

Mivel $P(R)$ és $P(I)$ konsans D-re, ezért a hasonlósági mértéket az alábbi formában is megadhatjuk:

$$S(q|d_j) = \frac{P(d_j|R)\cdot P(R)}{P(d_j|I)\cdot P(I)} \approx \frac{P(d_j|R)}{P(d_j|I)}$$

Kifejezés- függetlenség:

Legyen

$\mathbf{t} = (t_1, \ldots, t_i, \ldots, t_n)$ a d_j dokumentum indexkifejezés-vektora, ahol $t_i = 1$, ha szerepel a $d_j$dokumentumban, és 0 egyébként. Így megadhatjuk a: $P(d_j|R)$-t a $P(\mathbf{t}|R)$-rel, ha feltételezzük, hogy az indexkifejezések egymástól függetlenek, akkor:

$$P(\mathbf{t}|R) = P(t_1 \bigcap R)P(t_2 \bigcap R) \ldots P(t_n \bigcap R) = P(t_1|R)P(t_2|R) \ldots P(t_n|R) = \prod_{i=1}^{n} P(t_i|R)$$

,

tehát az indexkifejezések függetlenségének alkalmazásával a hasonlósági mérték az alábbi formában írható fel:

$$S(q|d_j) = \frac{\prod_{i=1}^{n} P(t_i|R)}{\prod_{i=1}^{n} P(t_i|I)}$$

Felhasználva, hogy a q kérdés csak k indexkifejezésből áll, elég csak az kérdésben szereplő indexkifejezésekre alkalmazni:

$$S(q|d_j) = \frac{\prod_{t_i \in q} P(t_i|R)}{\prod_{t_i \in q} P(t_i|I)}$$

A visszakeresés lépései:

1. Feltesszük, hogy a priori valószínűség:
   $P(t_i|R) = 0.5$, és a
   $$P(t_i|I) = \frac{m_i}{m}, ahol$$
   $i = 1, \ldots, n$, és az
   m_i azon dokumentumok száma, amelyek tartalmazzák t_i indexkifejezést
2. Hasonlósági értékek kiszámítása: $S(q|d_j)$ minden d_j dokumentumra.
3. Rangsor felállítása.
4. A $P(t_i|R)$ valószínűségek újraszámolása az alábbiak alapján:
   $$P(t_i|R) = \frac{|V_i|}{|V|},$$
   és
   $$P(t_i|I) = \frac{m_i - |V_i|}{m - |V|},$$
   ahol
   $V \subseteq D$ a válaszként visszakapott dokumentumokat jelöli, a
   $V_i \subseteq V$ pedig azon dokumentumokat jelöli, amelyek tartalmazzák a q kérdés t_i indexkifejezését.
5. A Folytatás a 2. lépéstől.

3. fejezet

Információ-visszakereső tecnológiák

Technikai oldalról az információ-visszakereső rendszer egy adatbázisból és a kereső felületből áll. Az adatbázis például weboldalak gyűjteménye. A kereső felületen keresztül, lekérdezések végrehajtásával érhetjük el az adatbázisban tárolt dokumentumokat.

A felhasználói munka (az információigény megfelelő átalakítása keresőkifejezéssé), és a dokumentumok logikai nézete (logical view) együttesen befolyásolják a releváns információ hatékony visszakeresését.

3.1. Általános információ-visszakereső rendszer architektúrája

Az információ-visszakeresési folyamat több részfolyamatból áll.

1. Szöveges adatbázis definiálása, azaz:
   - a dokumentumok,
   - a szövegfeldolgozó műveletek,
   - és a szövegmodell (szöveg szerkezete, visszakereshető elemek) megadása.
2. A szövegfeldolgozó műveletek átalakítják a dokumentumokat, és elkészítik azok logikai nézetét.
3. Ezután indexelni kell a szöveget, amit már az indexelő modul végez. Az index kritikus paraméter: nagy terjedelmű szöveg gyors keresését teszi lehetővé. Többféle fájl struktúra (file structure) használható a tárolására, a legelterjedtebb az inverz fájl (inverted file).

4. A lekérdezés folyamán az indexek és a keresőkérdés kifejezéseinek összehasonlítása történik.

5. A kapott eredmények alapján végzi el a rangsoroló modul a hasonlósági értékek kiszámítását,

6. és adja vissza a felhasználónak a találati listát.

Tehát egy általános információ-visszakereső rendszer az alábbi elemekből állhat:

- ADATBÁZIS. Központi adatbázis tárolja a dokumentumok halmazát, amelyen a keresés történik. Ezek a dokumentumok az adatbázisba manuálisan, vagy speciális számítógépprogramok segítségével kerülnek.
- INDEXELŐ MODUL. Az adatbázisban lévő dokumentumokból az indexelő modul készíti az indexeket inverz fájl struktúrában. Ezeket a struktúrákat használja majd a lekérdező modul, hogy megtalálja a felhasználó által feltett keresőkérdésre a válasz dokumentumokat.
- LEKÉRDEZŐ MODUL. A lekérdező modul olvassa be a keresőkérdést, és alakítja át a felhasználható formára. A lekérdező modul az indexeket használja a felhasználói kérdésre megfelelő válaszok megtalálására. Ezután a talált dokumentumokat átadja a rangsoroló modulnak.
- RANGSOROLÓ MODUL. A rangsoroló modul kiszámítja a lekérdező modul által talált dokumentumokra a hasonlósági értékeket (az indexek felhasználásával). Majd a hasonlósági értékek alapján csökkenő sorrendbe rendezi a dokumentumokat, és ezt találati listaként megjeleníti a felhasználónak. Ennek a kiszámítására sokféle módszer került ismertetésre a korábbi fejezetekben.

3.2. Adatbázis

Az információt objektumokban keressük.

$O = \{O_1, \ldots, O_j, \ldots, O_m\}$ jelölje az objektumok véges halmazát, ahol az objektum lehet:

- szöveg (könyv, újságcikk, előadás jegyzet, címek, stb...)
- kép (fényképek, rajzok, stb...)
- hang (dalok, zenei művek, stb...)
- multimédia (szöveg kép és hang együttesen).

Az információ-visszakeresés céljából minden O_j objektum megadható egy szöveges D_j dokumentumként. A továbbiakban tekintsük a D_j-t az O_j objektum helyett, minden j esetben. Nyilvánvaló, hogy ezen a D_j dokumentumon az O_j objektumot értjük.

Példaként, ha az adott objektum nem éppen egy szöveges objektum, akkor ahhoz is hozzárendelhető egy szöveges dokumenetum. Ha O_j objektum egy képi objektum, ami például egy családi fénykép, akkor a hozzátartozó D_j dokumentum lehet egy szöveg, ami tartalmazza például:

- a fényképen szereplő személyek nevét,
- a fénykép milyen eseményen készült, az esemény részletes leírása,
- a fénykép mikor készült,
- a fénykép hol készült,
- a fényképet ki készítette.

3.3. Indexelő modul

A dokumentumok nemcsak, hogy strukturálatlan módon tárolják a szövegeket, hanem sok egyéb más jellemzővel is rendelkeznek, amelyek dokumentumról dokumentumra változhatnak. Ahhoz, hogy az információ-visszakeresés során a dokumentumokat egységesen és gördülékenyen tudjuk kezelni, át kell hidalni azokat a különbözőségeket, amelyek az egyes dokumentumok esetében adódhatnak. Szöveges dokumentumok esetében tehát tisztában kell lennünk azokkal a jellemzőkkel, amelyekkel egy tetszőleges szöveges dokumentum rendelkezhet. A dokumentumokat legtöbbször indexkifejezések bizonyos struktúráival reprezentáljuk. Az indexkifejezéseket vagy automatikusan, közvetlenül a dokumentumok szövegéből nyerjük, vagy szakértők határozzák meg. Az indexkifejezések adják a dokumentumok logikai nézetét.

A manuális adatbázis-feltöltést emberek végzik. Ezáltal pontosak, ám ez drága, nehezen fenntartható eljárás, és nem minden témát érint, továbbá nehezen frissíthetőek és szubjektívek az így előállított adatbázisok.

Gyorsabb információ- visszakeresést eredményez az index, amely kiválasztott szavakból és kifejezésekből áll. Mutatók (pointerek) jelzik, hogy mely dokumentum(ok)ban találhatók meg az indexelt szavak. A számítógépek lehetővé teszik nagy terjedelmű indexek automatikus létrehozását. Az automatikus indexeléssel az információ-visszakeresési probléma sokkal inkább magához a rendszerhez, mint a felhasználó igényéhez kapcsolódik: főként hatékony indexek létrehozásából, a felhasználói kérdés (keresőkifejezés) eredményes feldolgozásából és a találati lista minőségét javító rangsoroló algoritmusok fejlesztéséből áll.

Az automatizált kulcsszó meghatározást másnéven tartalomszűrésnek is nevezik, aminek egyik legfontosabb felfedezése a Zipf-törvény. Ennek az elméleti fontosságát az adja, hogy a nyelv teljes megértése felé vezető útnak egyik állomása mindenképpen a nyelvben fellelhető statisztikai szabályszerűségek felismerése kell, hogy legyen. A Zipf-törvény gyakorlati fontossága pedig abból adódik, hogy a számítógépes információ-visszakereső rendszerek által használt indexelési módszerek erre a törvényre épülnek. Zipf hipotézise kimondja, hogy egy kulcsszó előfordulásának száma a dokumentum összes kulcsszójának számához képest

(frekvencia) és a frekvencia által létrehozott rangsor szorzata egy állandó érték körűl mozog [**?**]. Ezt erősíti meg Luhn és Hayes, akik hasonló elmélettel általak elő a dokumentum összes szavát vizsgálva. Hipotézisük kimondja, hogy a jellemző szavak egy bizonyos sávban találhatóak meg, amit a nagyon sűrűn és a nagyon ritkán előforduló szavak határolnak. Ennek megbízhatósága eléggé nyelvspecifikus.[**?**]

A számítógépek lehetővé teszik azt, hogy a dokumentumokat az azokat alkotó összes szóval reprezentáljuk. Ebben az esetben azt mondjuk, hogy a visszakereső rendszer a dokumentumok teljes szövegű (full text) logikai nézetét (reprezentálásának módszerét) alkalmazza. Nagy dokumentumhalmazok esetén a rendszernek csökkentenie kell a reprezentált indexkifejezések számát. Különböző ún. szövegfeldolgozó (test operation) műveletekkel csökkenthető a dokumentumreprezentálás komplexitása. A műveletek lehetővé teszik azt, hogy a teljes szöveg helyett indexkifejezésekkel reprezentáljuk a dokumentumokat. Valójában a teljes szövegű reprezentálás a legteljesebb logikai nézete a dokumentumoknak, ugyanakkor a leginkább számításigényes a használata.

A szövegfeldolgozás lépései a következők:

1. A szavak, azaz a lexikai egységek azonosítása a szövegben (pl. írásjelek elhagyásával). A lexikai egység, azaz a szó karakterek sorozatából áll, amelyeket egymástól bizonyos speciális karakterek (pl. írásjelek, szünetjel) választanak el.

2. Stoplista alkalmazása. Stoplista egy olyan listát jelent ami azokat a szavakat tartalmazza, amelyek általában nem hordoznak jelentést egy dokumentumban

3. Szótőre visszavezető algoritmus alkalmazása. Ez az alkalmazás minden szót redukál, vagy áttranszformál nyelvi szótőre. (Angol nyelvterületen a Porter algoritmus a legismertebb). Gyakorlatilag a szótövek lesznek az indexkifejezések.

4. Minden t_i indexkifejezésre a D_j dokumentumban való előfordulási számának a meghatározása: f_{ij}.

5. Minden egyes t_i indexkifejezés összes elfordulásának kiszámítása:

$$tf_i = \sum_{j=1}^{m} f_{ij}$$

6. Kifejezések sorbarendezése tf_i szerint. A nagyon magas értékűeket kirekesztjük, mert az a kifejezés ami nagyon gyakran előfordul már nem mond semmit. Hasonlóan a nagyon alacsony előfordulásúakat is kihagyjuk a kifejezések halmazából, mert azok pedig nem meghatározó jellegűek.

7. A megmaradt kifejezéseket nevezzük a továbbiakban indexkifejezéseknek.

3.3.1. Stoplista

Egy dokumentum szavakból áll, amelyek:

- előforulhatnak többször is a dokumentumban,
- és vannak olyan szavak is amelyek csupán néhányszor
- vagy csak egyszer fordulnak elő a dokumentumban.

A dokumentumfeldolgozás abból indul ki, hogy a:

- túlságosan gyakran előforduló szavak (amelyek előfordulási gyakorisága egy bizonyos küszöbérték feletti) többnyire nem hordoznak információt, valamint a
- akárcsak a túl ritkán előforduló kifejezések (amelyek előfordulási gyakorisága egy bizonyos küszöbérték alatti), amelyek valószínűleg azért fordulnak elő ritkán, mert a dokumentum szerzője nem tartotta azokat fontosnak.

Ezen szavak kezelésére találták ki a stoplistát, amelyek a túlságosan gyakori, illetve ritka kifejezéseket tartalmazza. Stoplista elemei többek közt a kötőszavak, névelők. A magyar nyelvű stoplista néhány eleme:

- A
- És
- Az
- Van
- Is
- Mely
- Ez
- Hogy

A stoplista egy olyan listát jelent ami azokat a kifejezéseket tartalmazza, amelyek általában nem hordoznak jelentést egy adott dokumentumban. Azokat a szavakat, amelyeket a stoplista tartalmaz kihagyjuk a további vizsgálatkor. A stoplista általában terület, illetve applikációfüggő. Az angol nyelvű dokumentumok feldolgozására leggyakrabban használt stoplista a TIME stoplista. Egy stoplista megalkotása lehet automatizált. Újabb stoplisták is generálhatók, például egy adott témához tartozó dokumentumokra, amikor a stoplistába olyan kifejezések is bekerülhetnek, amelyek egy általános stoplistának nem az elemei. Rendszerint egy általános stoplista a kiindulópont, majd a felhasználó módosítja, bővíti az adott témához kapcsolódóan.

3.3.2. Szótövesítés

A keresők a felhasználó által megadott kulcsszavak alapján keresnek az adatbázisban. Arra is gondolnunk kell, hogy a kulcsszavak kinyerése a dokumentumból nem elegendő ahhoz, hogy összehasonlítást tudjunk végezni. A szavakhoz a mondatban különféle ragok, toldalékok csatlakoznak. Ezeket valamilyen módszerrel el kell távolítanunk, mert zavaróak lehetnek két szó egyezésének megállapításában. A toldalékok alakja úgyszintén erősen nyelvspecifikus. Az angol nyelv nyújt egy kitűnő algoritmust e probléma leküzdésére, melyet Porter algoritmusának nevezünk[**?**]. Ez az algoritmus az azonos szótövű angol nyelvű szavakat ugyanarra az alakra hozza. A weboldalak nagy hányada angol nyelvű, ezért ez a megoldás megfelelő a weboldalak többségére, tehát a rendelkezésünkre áll egy egyszerű módszer a probléma kivédésére. Több nyelvnek úgyszintén létezik szótövesítő algoritmusa, így például a magyarnak is. Ezen algoritmusok ismeretében már elmondhatjuk, hogy a keresett kulcsszó és az adatbázisban levő kulcsszavak között összehasonlítást tudunk végezni.

3.3.3. Inverz fájl struktúra

A szövegfeldolgozási lépések (lexikai egységek azonosítása, stoplista alkalmazása, szótövesítés) során elkészített indexkifejezésekből készíti el az indexelő modul az indexet inverz fájl struktúrában:

Legyen O a feldolgozandó objektumok halmaza,

$$O = \{O_1, \ldots, O_j, \ldots, O_m\}$$

és jelölje

$$D = \{D_1, \ldots, D_j, \ldots, D_m\}$$

az objektumoknak megfelelő dokumentumokat. Az első három lépés elvégzése után (szavak meghatározása, stoplista alkalmazása, majd szótövesítés) az alábbi indexkifejezéseket kapjuk:

$T = \{t_1, \ldots, t_i, \ldots, t_n\}$.

A T indexkifejezéshalmazt használjuk az inverz fájl struktúra megalkotásához az alábbi módon:

1. A $t_1, \ldots, t_i, \ldots, t_n$ indexkifejezéseket ábécé sorrendbe rendezzük. Ezen cél megvalósításának érdekében többféle gyors rendezési algoritmus ismert.

2. Az I index tábla megalkotása, amelyben minden r_i sorban a t_i indexkifejezés van D_j dokumentumokra vonatkozó kódjaival, amely D_j dokumentumokban a t_i indexkifejezés előfordul.

Tehát, a t_i indexkifejezések ábécé sorrendben vannak, és mellettük azon dokumentumok kódjai, amelyek tartalmazzák az adott t_i indexkifejezést:

Indexkifejezés	**Dokumentum kódja**
t_1	$\{D_{11}, \ldots, D_{1k}\}$
...	...
t_i	$\{D_{i1}, \ldots, D_{is}\}$
...	...
t_n	$\{D_{n1}, \ldots, D_{np}\}$

Az inverz fájl struktúra (IF Inverted file Structure) egy I index táblából, és a hozzá tartozó MF (master file) objektumhalmazból áll.

Az MF az $O = \{O_1, \ldots, O_j, \ldots, O_m\}$ objektumokat tartalamzza:

Objektumok
O_1
...
O_j
...
O_m

Az I indextáblában lévő kódok tartalmazhatják a megfelelő objektum MF-ben elfoglalt címét is. Az inverz fájl struktúra az alábbi módon alkalmazható: Jelölje t a keresőkérdést, majd a megfelelő keresőalgoritmus alkalmazása után az IF-ben megtaláljuk a t indexkifejezésre vonatkozó sort, azaz azon dokumentumok kódjait, amelyek tartalmazzák a t indexkifejezést. Felhasználva a dokumentumok kódjait a megfelelő O objektumok visszakereshetők az MF-ből. Az inverz fájl struktúrában egyéb más adatok is tárolhatók, mint például:

- t_i előfordulásainak a száma a D_j dokumentumban,
- t_i összes előforulásainak a száma az összes dokumentumban
- stb...

3.4. Lekérdezés

A lekérdezési (információ-visszakeresési) folyamat az indexelés után kezdhető meg, és rendszerint a következő lépésekből áll:

3.4.1. A lekérdezés lépései

1. Adott egy felhasználói információigény.
2. A felhasználó kiválaszt egy információ-visszakereső rendszert.
3. A felhasználó információigénye alapján megfogalmaz egy kérdést természetes nyelven.
4. A természetes nyelvű kérdést az információ-visszakereső rendszer értelmezhető keresőkifejezéssé alakítja.

5. A rendszer elemzi, átalakítja a szövegnél alkalmazott szövegfeldolgozó műveletekkel, majd feldolgozza a keresőkifejezést. Ezáltal a keresőkérdést is indexkifejezések halmazává alakítja.

6. A rendszer a keresőkifejezés alapján végrehajtja a visszakeresést. Majd a visszaadott dokumentumokat átadja a rangsoroló modulnak.

3.4.2. Keresőkérdés megadása

A különböző keresők egyes kereséseknél jobbak, mint más keresők. Általában elmondható, hogy a pontosság javítható, ha a keresőkérdés több kifejezést tartalmaz.

A jelenleg elérhető keresők nagy része kulcsszó alapú keresést használ, és az adatbázis dokumentumait indexekkel jellemzi. A felhasználó beírja a kulcsszavakat (kérdés), a kereső pedig azonosítja és felsorolja a kérdést kielégítő dokumentumokat. A keresők több tekintetben különböznek egymástól: az indextábla mérete, keresési lehetőségek, a válaszok megadásának ideje, a válaszok megjelenítése, a válaszok relevanciája. A legmegfelelőbb kereső kiválasztása részben attól függ, hogy alaposan megértsük és ismerjük, hogy kereső a dokumentum mely elemeit indexeli: egyes keresők a teljes dokumentumot, míg mások csak bizonyos részeit (cím, fejléc).

Ezen különbségek miatt a különböző keresők nagyon eltérő válaszokat adnak vissza ugyanarra a kérdésre. A keresési lehetőségek (opciók) sem egyformák. Némely esetekben az OR (logikai VAGY) az alapértelmezett opció, és relevancia rangsoroló algoritmus rangsorolja a releváns dokumentumokat, máskor pedig több lehetőség közül is választhatunk: AND, OR, NOT, NEAR, stb. Amennyiben a felhasználó nem ismeri a Boole-logikát, előfordulhat, hogy nem arra kap választ, amire szeretne. Éppen ezért, a keresések nagy százalékában nem szerepel az ÉS és a VAGY, ami segítené a keresőt leszűkíteni a keresést, ezáltal lehetővé tenné, hogy sokkal pontosabb eredményekhez jusson a felhasználó a kérdésével kapcsolatban.

A keresők hasonlóképpen működnek, mégis más-más eredményeket adnak vissza. Ez betudható az adatbázis-feltöltés különböző módszereinek, de annak is, ahogy a felhasználó által megadott kulcsszavakat értelmezik. Például egy egyszerű keresés két kulcsszóval, már bizonyos kérdéseket vet fel:

- A két kulcsszót egymás mellett kell keresni a dokumentumban?
- A két kulcs akármelyikének jelenléte a dokumentumban elegendő vagy mindkettő jelen kell-e legyen?
- Az első kulcs fontosabb, mint a második?

Egyes keresők ezt a problémát úgy oldják fel, hogy a kérdést más megfogalmazásban teheti fel a felhasználó, esetleg a felhasználói felületen ki lehet választani a kulcsok közötti kapcsolatot. Például a szokás szerint a két kulcsszó közé alulhúzás jelet téve, vagy macskakörmök közé helyezve a keresendő szöveget

a keresők egy darab kulcsszónak értelmezik és csak olyan válaszokat adnak eredményül, ahol ezek a kulcsszavak egymás mellett találhatóak.

Egyes információ-visszakereső rendszerek lehetővé teszik azt, hogy egyszerű mondatot írjunk be keresőkifejezésnek, és kísérletek folynak összetett mondatok feldolgozására, a legtöbb kereső használatakor azonban a keresőkifejezést a rendszer számára feldolgozható formában, kulcsszavakkal és operátorokkal kell megadni. Sok felhasználó nem ismeri az alkalmazható szövegfeldolgozó műveleteket és logikai operátorokat, ezért az általuk megadott keresőkifejezés gyakran nem kielégítő, nem tükrözi megfelelően információigényüket. A rosszul meghatározott keresőkifejezés alacsony visszakeresési hatékonysághoz (retrieval effectiveness) vezet.

A felhasználó gyakran nincs tisztában a kereső működési elvével, és a keresőkérdés megadásának lehetőségeivel, ezáltal azzal sem, hogy egy megfelelőképpen megfogalmazott keresőkérdéssel lényegesen javítható a kereső hatékonysága [**?**]. Ennek érdekében hasznos lehet egy rövid tájékoztatást adni (vagy egy rámutató linket) a keresőkérdés megadásának lehetőségeiről. A felhasználót meg kell tanítani a megfelelő kérdésfeltevésre. A kérdés feltevésének módja erősen befolyásolja a visszaadott eredmények minőségét. Egyes keresők kis és nagybetűre érzékenyek. Például, ha nagybetűvel írunk egy szót, akkor azt pontosan olyan formában keresi a kereső, és a kisbetűs vátozatával nem foglalkozik. Ugyanakkor a szó különböző alakjai miatt is a elveszhetnek releváns dokumentumok.

A mai keresőket gyakran ellátják ún. haladóknak (advanced) való kereséssel. Itt a felhasználó szűrési feltételeket tud megadni, ami alapján a keresést végrehajtja a szoftver.

A kereső kiválasztásánál a keresési lehetőségek kifinomultsága mellett az indextábla méretét is figyelembe kell venni. Egyszerűbb keresési opciókkal, de nagy számú indexkifejezéssel is sikeres lehet a keresés.

3.5. Rangsorolás

A rangsoroló modul készíti el a felhasználónak visszaadott találati listát, amelyhez az alábbi megelőző lépésekre van szükség:

1. Az indexelő modul által meghatározott indexkifejezésekre a kifejezés-dokumentum mátrix generálása, ami az indexkifejzésekhez tartozó súlyszámokat adja meg.

2. Keresőkérdés megadása. A felhasználó által feltett Q_k keresőkérdésből is egy $\mathbf{v}_k$ vektor generálható az előbbiekhez megegyező módon. A kérdés is úgy tekinthető, mint egy dokumentum. A keresőkérdés átalakítása súlyokból álló keresőkérdés-vektorrá, ahol a súlyszámok meghatározása ugyanazon módszerrel történik, mint a kifejezés-dokumentum mátrixban.

3. A lekérdező modul által visszaadott dokumentumokra a hasonlósági értékek kiszámítása.

4. A visszakeresett dokumentumok rangsorolása hasonlósági értékük alapján,

5. A találati lista megjelenítése

6. A felhasználó dönt a visszakapott dokumentumok (találati lista) relevanciájáról.

3.5.1. Kifejezés-dokumentum mátrix generálása

A kifejezés-dokumentum mátrix elemeinek, azaz a súlyszámok meghatározására többféle módszer is bemutatásra került a Vektortér modell című fejezetben.

Legyen

$D = \{D_1, \ldots, D_j, \ldots, D_m\}$ dokumentumok véges halmaza, ahol $D_j \in \wp(T)$, valamint

$T = \{t_1, \ldots, t_i, \ldots, t_n\}$ dokumentumokat leíró indexkifejezések véges halmaza.

a w_{ij} súly azt mutatja meg, hogy a t_i indexkifejezés mennyire határozza meg a D_j dokumentumot.

Megadható a TD kifejezés-dokumentum (term-by document) mátrix, az alábbi formában:

$$TD = (w_{ij})_{n \times m}, i = 1, \ldots, n, j = 1, \ldots, m,$$

ahol:

w_{ij} a t_i indexkifejezés súlya a D_j dokumentumban, ami azt mutatja meg, hogy a t_i indexkifejezés mennyire határozza meg a D_j dokumentumot.

m: a dokumentumok száma (mátrixnak m oszlopa van)

n: az indexkifejezések száma (a mátrixnak n sora van)

TD:=

$$\begin{pmatrix} w_{11} & \ldots & w_{1j} & \ldots & w_{1m} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{i1} & \ldots & w_{ij} & \ldots & w_{im} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ w_{n1} & \ldots & w_{nj} & \ldots & w_{nm} \end{pmatrix}$$

Az indexkifejezések, és a hozzájuk tartozó súlyok meghatározása nagyon nehéz elméleti (nyelvészeti, szemantikai), és gyakorlati probléma.

3.5.2. Találati lista megadása

A kiválasztott súlyszámítási séma használatával a keresőkérdéshez megadható a keresővektor. Majd ezután következik a hasonlósági értékek kiszámítása. Ennek kiszámítására is többféle módszer létezik, amit a Vektortér modell fejezet tárgyal részletesen. Majd a rangsoroló modul ezen értékek alapján csökkenő sorrendbe rendezi a dokumentumokat, és ezt találati listaként megjeleníti a felhasználónak. A találati lista vizuális megjelenítése is különböző lehet. A találati lista tartalmazhatja a dokumentumokhoz tartozó hasonlósági értékeket is, ami segítheti a felhasználót abban, hogy meddig menjen el a találatok megtekintésében.

A weben található webkeresők gyakran súlyszámmal látják el az egyes keresések eredményeiként kapott weboldalakat annak alapján, hogy a felhasználó első ránézésre relevánsnak találta-e az eredményt, vagyis rákattintott-e az eredményre. Ezáltal a visszajelzett válaszok között rangsort tud felállítani. Gyakran a rangsort csak az indexkifejezések előfordulási számával arányosan állítja össze. Vannak olyan webkeresők is, ahol pénz kérdése egy-egy weboldal súlyszámának megállapítása. Ennek etikussága természetesen megkérdőjelezhető, főleg ha erről nem tájékoztatják kellőképpen a felhasználót, de tény, hogy ilyen létezik. A webkeresők részletes ismertetésére későbbi fejezetben kerül sor.

PÉLDA:

Legyen az adatbázis az O, azaz az eredeti (azaz valós) objektumok halmaza az alábbi:

$O = \{O_1, O_2, O_3\}$

- O_1= Az információ-visszakeresés egy nagyon gyorsan és dinamikusan fejlődő tudományág. Az információ-visszakeresés modelljeit három nagy csoportba lehet sorolni. A leggyakrabban használt a Boole-féle információ-visszakeresés.
- O_2 = Napjainkban a kereskedelmi keresők a Boole-féle információ-visszakeresés modelljét használják leggyakrabban. Ez a megoldás implementálható a legkönnyebben az információ-visszakeresés modelljei közül.
- O_3 = Az implementálás során fontos a memória és a lemez megfelelő használata. Az információ-visszakeresés implementálása nem egyszerű feladat.

A különböző szövegfeldolgozási műveletek elvégzése (lexikai egységek azonosítása, stoplista alkalmazása, szótövesítés) után az alábbi indexkifejezések nyerhetők ki: A $T = \{t_1, t_2, t_3, t_4\}$

T = {t_1 = információ-visszakeresés, t_2 = tudományág, t_3 = Boole-féle, t_4 = implementálás}

Az O objektumhalmazból nyert dokumentumokhalmaz a következő:

D = {D_1 , D_2 , D_3 }

- D_1= {információ-visszakeresés, tudományág} = $\{t_1, t_2\}$
- D_2= {információ-visszakeresés, Boole-féle, implementálás} =$\{t_1, t_3, t_4\}$
- D_3= {információ-visszakeresés, implementálás} = $\{t_1, t_4\}$

A kérdés:

Q = {információ-visszakeresés, Boole-féle} = $\{t_1, t_3\}$

A dokumentumokat és a kérdést kifejezések halmazának tekintjük.

A kifejezés-dokumentum mátrix az alábbi formában adható meg hossznormalizált súlyszámítási sémát alkalmazva: TD:=

$$\begin{pmatrix} 0.707 & 0.577 & 0.707 \\ 0.707 & 0 & 0 \\ 0 & 0.577 & 0 \\ 0 & 0.577 & 0.707 \end{pmatrix}$$

,ahol $w_{22} = 0$ azt jelenti, hogy a t_2 indexkifejezés egyszer sem fordul elő a D_2 dokumentumban.

A Q keresőkérdés legyen az alábbi: $Q = \{t_1, t_3\}$

Q:=

$$\begin{pmatrix} 0.707 \\ 0 \\ 0.707 \\ 0 \end{pmatrix}$$

A számítás során $K = 0.7$ küszöbértéket definiáltunk.

Hasonlósági értékek meghatározása három féle (Pont szorzat, Koszinusz, Dice) hasonlósági mérték alkalmazásával.

A számítás eredményét az alábbi táblázat tartamazza.

Számítás típusa	s_{1k}	s_{2k}	s_{3k}	**Válaszdokumentumok**
Pont szorzat	0.5	0.82	0.5	D_2
Koszinusz	0.5	0.82	0.5	D_2
Dice-együttható	0.5	0.8	0.5	D_2

A számítás során $K = 0.7$ küszöbérték definiálásával:

A Q keresőkérdésre a válasz az O_2 objektum lesz mind a három hasonlósági mérték alkalmazása esetén.

Irodalomjegyzék

[1] Baeza-Yates, R. and Ribeiro-Neto, B., *Modern Information Retrieval*, ACM Press New York, Addison-Wesley, 1999.

[2] Baeza-Yates, R., *Information retrieval in the Web: Beyond current search engines*, International Journal of Approximate Reasoning, vol. 34, pp: 97-104., 2003.

[3] Belew, R.K., *Finding Out About*, Cambridge University Press, 2000.

[4] Meadow, C.T., Boyce, B.R. and Kraft, D.H., *Text Information Retrieval Systems*, Second edition, Academic Press, San Diego, CA., 1999.

[5] Berry, W.M., Browne, M. *Understanding Search Engines*, SIAM, Philadelphia, 1999.

[6] Dominich, S., *Mathematical Foundations of Information Retrieval*, Kluwer Academic Publishers, Dordrecht, Boston, London, 2001.

[7] Dominich, S., *Paradox-free Formal Foundation of Vector Space Model*, Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information retrieval, Tampere, Finlad, August 11-15., pp. 43-60, 2002.

[8] Dominich, S., *Connectionist Interaction Information Retrieval*, Information Processing and Management. Elsevier, vol 39, no 2, pp: 167-194, 2003.

[9] Salton, G., *Automatic Phrase Matching*, In: Hays. D. G. (ed.) Readings in Automatic Language Processing, pp:169-188., American Elsevier, New York, 1966.

[10] Salton, G. and McGill, M., *Introduction to Modern Information Retrieval*, McGraw Hill, New York, 1983.

[11] Salton, G., *Another look at automatic text-retrieval systems*, Communications of the ACM, vol. 29, no. 7, pp: 648-656., 1986.

[12] M. F. Porter, *An algorithm for suffix stripping*, Program, 14(3):130-137., 1980.

[13] Van Rijsbergen, C.J., *Information Retrieval*, Butterworth, London, 1979.

[14] Van Rijsbergen, C.J., *The Geometry of IR*, Cambridge University Press, Cambridge, U.K., 2004.

[15] Luhn H.P., *Keyword-in-Context Index for Technical Literature (KWIC Index)*, In: Readings in Automatic Language Processing, ed by Hays, D. D., American Elsevier Publishing Company, Inc., 1966.

[16] Bernard J. Jansen, Spink, A., and Saracevic. T., *Real life, real users, and real needs: a study and analysis of user queries on the web*, Information Processing and Management, 36(2) pp: 207-227., 2000.

[17] Turtle, H. and Croft, W. B., *Inference Networks for Document Retrieval*, Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp .1-24, Brussels, Belgium, 1990.

[18] Zipf, G., *Human behavior and the principle of least effort*, Addison-Wesley, Cambridge, MA.1996.

[19] Zimmerman, H.J., *Fuzzy set theory - and its applications*, Kluwer Academic Publishers, Norwell-Dordrecht, 1996.