Bilgi Dünyası *2007, 8(1): 49-75*

Kavram Tabanlı Bilgi Geri Getirim Yaklaşımı

Concept-Based Information Retrieval Approach

Hayri SEVER[*], Fuat AKAL[**] ve Güven KÖSE[***]

Öz

Arama motorları, hem belgelerin yazarları hem de bilgi ihtiyaçlarını açıklayan kullanıcılar tarafından kullanılan sözlükler arasındaki farklar yüzünden ortaya çıkan uçurumlarla başa çıkmada yetersiz hale gelmişlerdir. Bu problemi azaltmanın bir yolu kavram tabanlı bilgi erişimini ortaya koymaktır. Bu çalışmada, bazı perspektiflerde RUBRIC sisteminin bir uzantısı olarak değerlendirilen ancak betimsel düzeyde erişime uygun farklı özellikler içeren bir erişim modeli önerilmekte ve elde edilen sonuçlar değerlendirilmektedir.

Anahtar sözcükler: *Kavram tabanlı bilgi erişim, En küçük terim kümeleri*

Abstract

Search engines become incompetent in tackling with the gap caused by differences in vocabularies used by both the authors of documents and users in expressing their information needs. One way to alleviate this problem is to introduce concept based retrieval of information. In this study, a model that can be regarded as an extension of RUBRIC system in some perspectives, but it has some distinct features especially suited for enabling descriptive-level retrieval was proposed and obtained results were evaluated.

Keywords: *Concept-based information retrieval, Minimal term sets.*

[*] *Prof.Dr.; Çankaya Üniversitesi, Bilgisayar Mühendisliği Bölümü, 06530 Balgat /Ankara (sever@cankaya.edu.tr).*
[**] *Basel Üniversitesi, Bernoullistr 16, CH-4056 Basel, İsviçre (fuat.akal@unibas.ch).*
[***] *Öğr. Gör.; Başkent Üniversitesi, Bilgisayar Mühendisliği Bölümü, 06530 Bağlıca / Ankara (gkose@baskent.edu.tr).*

1. Giriş

Bilgi erişimi (BE), (*information retrieval, IR*) yaygın olarak kullanılan ancak, çoğunlukla doğru olarak tanımlanamayan bir kavramdır. *Bilgi* sözcüğü yanlış çağrışımlara neden olmaktadır. Bilgi erişimi bağlamındaki *bilgi* sözcüğü, Shannon'un iletişim teorisinde verilen teknik anlamdaki tanımda olduğu gibi kolayca ölçülemez (Shannon ve Weaver, 1964). Aslında birçok durumda *bilgi* sözcüğünün yerine *belge* sözcüğünün konulmasıyla, yapılan erişimin türü kolayca tanımlanabilmektedir.

Bilgi erişiminin kapsamı konusunda literatürde farklılıklar gösteren tanımların odaklandığı özelliklerin başında, kullanıcıların bilgi ihtiyacı ile ilgili bilgilerin seçilmesi (ya da derlem içinde bulunup geri getirilmesi) gelmektedir (Belkin, Cool, Croft ve Callan, 1993, ss. 339-346; Belkin, Kantor, Fox ve Shaw, 1995, ss. 431-448; Croft, 2000, ss. 1-36; Gauch, Wang ve Gomez, 1996, ss. 637-649; Lee, 1995, ss. 180-188; Saracevic ve Kantor, 1998, ss. 197-216). Bu özellik, bilgi ortamından (metin ya da çoklu ortam belgesi) ve erişim yönteminden (geri-getirme ya da süzgeçleme) farklı karakteristiklere sahiptir ve bu yönüyle bilgi erişimi veri erişiminden ayrılmaktadır (Rijsbergen, 1975; Klir, Clair ve Yuan, 1997, ss. 13-37; Silberschatz, Jorth ve Sudarshan, 1997, ss. 215-250).

İnternetin yaygınlaşmasıyla daha da büyüyen veri havuzundaki bilginin çıkarılması günümüzün en popüler konularından biri olmuş ve bu bağlamda birçok İnternet arama motoru geliştirilmiştir. Ancak, arama motorları yazar ve kullanıcı sözlüklerinin çakışmamasından doğan etkin erişim problemine çözüm getiremekte sınırlı kalmışlardır (Salton ve McGill, 1983; McCune, Tong, Dean ve Shapiro, 1985, ss. 939-944; Alsaffar, Deogun, Raghavan ve Sever, 1999, ss.114-122).

Yüksek hızlı bilgisayarların gelişimi ile birlikte birçok kişi bilgisayarların tüm belgeleri okuyup, ilgili olanları çıkarabileceğini düşünmeye başlamıştır. Bu düşünce tarzı, problemin sadece belgeleri depolamaktan ibaret olmadığı, aynı zamanda belgelerin içeriğini çözümlemek gibi entelektüel bir yanının da olduğunun anlaşılmasına yol açmıştır. Donanım teknolojilerinde kaydedilecek gelişmelerin, doğal dille veri girişi ve depolama işlemlerini daha olurlu hale getireceği akla yatkın bir düşüncedir. Ancak, insanın yaptığı *okuduğunu anlama* işleminin karşılığı olan, belgelerin *içerik analizlerinin* bilgisayarlar tarafından yapılması işlemi çok daha güç olan bir problemdir. Daha somut bir

şekilde söylenecek olursa, *okuduğunu anlama* işlemi sözdizimsel ve anlamsal olarak bilginin çıkarılması ve belgenin ilgili olup olmadığına karar verilmesi sürecidir. Buradaki zorluk, sadece bilginin nasıl çıkarılacağını bilmek değil aynı zamanda onun ilgililik derecesinin tespit edilmesi gerekliliğidir (Salton ve Buckley, 1988).

Bilgi erişim terminolojisinde mükemmel erişim (*perfect retrieval*), bilgi erişim çıktısındaki her bir belgenin kullanıcının bilgi ihtiyacı ile ilgili olması ve derlemde bilgi erişim çıktısında yer almayan bir ilgili belge olmaması durumu olarak tanımlanır. Fakat mükemmel erişimin pratikte uygulanması mümkün değildir. Bundan dolayı mükemmel erişim yerine kabul edilebilir erişim tanımlanmıştır. Buna göre, bilgi erişim çıktısındaki ilgili belgelerin ilgisizlere göre daha önde sıralanması gerekmektedir (Raghavan ve Sever, 1995, ss. 344-351). Bu bağlamda, geliştirilen bilgi erişim sistemindeki otomatik erişim stratejilerinin amacı, az miktarda ilgisiz belgeyi de kabullenerek ilgili olan tüm belgelere erişebilmektir. Erişim işlemi, bir şekilde belge içeriklerinin belirlenmesi ve bunlardan yararlanarak girilen sorguyla ilgili olanların bulunması yoluyla gerçekleştirilir. Kullanıcının sorgusuna cevap olarak döndürülen belge listesi (*erişim çıktısı*), erişim işlevi yardımıyla hesaplanan ve belgenin kullanıcının sorgusunu ne oranda karşıladığının bir göstergesi olan *erişim durum değeri*ne (*EDD*) göre sıralı olarak sunulur (Bookstein ve Cooper, 1976, ss. 153-167).

Var olan bilgi erişim sistemlerine bakıldığında, bunların büyük kısmının Boolean mantık çerçevesinde çalıştığı görülmektedir. Gerçekleştiriminin kolay olmasına karşın, Boolean tabanlı sistemlerde kullanıcılar istediklerini ifade etmekte güçlükler çekmektedirler. Kullanıcılarla belge yazarları arasında zaten bulunmakta olan terminolojik uçurum bu güçlüğü iyice artırmaktadır. Ayrıca, yapılan sorgulamaların en iyileştirildikten sonra saklanıp, daha sonra ihtiyaç duyulduğunda yeniden kullanılabilmesi de önemli bir ihtiyaçtır. Bu sorunların çözümü için yeni modeller aramak gerekmektedir. Bu çalışmanın temelini oluşturan *kavram tabanlı bilgi erişim modeli* bu arayışlar sonucu ortaya çıkmıştır (Alsaffar, Deogun, Raghavan ve Sever, 1999, ss. 114-122).

Kavram Tabanlı Bilgi Erişim (*Concept-Based Information Retrieval*) modeliyle ortaya konan bilgi tabanı ve kural ağacı yaklaşımları sorgular için hesaplamaların maliyetini oldukça yükseltmektedir. Bu nedenle yeni bir yaklaşıma ihtiyaç duyulmaktadır. Bu konudaki yaklaşımımızı *En Küçük Terim Kümeleri*, *ETK* (*Minimal Term Sets, MTS*)

olarak adlandırmaktayız. ETK modeli aracılığı ile kullanıcıya, var olan sorgu başlıkları kümesi içinden, kullanıcı ihtiyacını en iyi biçimde karşılayacak olanları, belirli bir sıra dahilinde seçme olanağı tanınmaktadır. Böylece, kural ağaçlarıyla gelen hesaplama maliyetinden kurtulmak mümkün olmaktadır. Bu çalışma kapsamında, ETK yaklaşımı ile *Isite/Isearch* sisteminin betimsel düzeyden kavramsal düzeye taşınması için gerekli model ortaya konulmuştur.

2. Kavram Tabanlı Bilgi Erişim

Literatürde, bilginin depolanması ve erişilmesi amacıyla kullanılan araçların yetenekleri hakkında çelişen görüşler vardır. Bir taraftan HTML belgelerinin insanlar tarafından kolay anlaşılır olduğu halde makinelerce kolay anlaşılır olmayışı nedeniyle İnternetin ortaya kaotik bir durum çıkardığı uzun zamandır iddia edilirken, diğer taraftan da bilginin özellikle elektronik kataloglar için Web madenleme açısından yeterince yapısal olduğu hipotezi ortaya atılmıştır.

Ticari arama makinelerinin çoğu Boolean model üzerinde geliştirilmiştir. Bu tip bir sistem, Boolean *VE*, *YADA* ve *DEĞİL* işleçleriyle bağlanmış sorgu terimleriyle elde edilen sorgulama isteklerinin formülize edilmesini destekler. Sorgu deyiminde görülen Boolean bir işlecin değerlendirilmesi için, karşılık gelen küme işleci uygulanır. Örneğin, Boolean modelde çalışan bir bilgi erişim sistemine "*veri VE madenleme"* sorgusu gönderildiğinde; sistem, *A* ve *B*, sırasıyla *veri* ve *madenleme* dizin terimlerini içeren belgelerin kümelerini göstermek üzere, *A* ve *B* kümelerinin kesişimini döndürecektir.

Diğer bir deyişle, bir belge sorgu sonucunda sadece ve sadece *A* ve *B* kümelerinin kesişiminde yer alıyorsa döndürülecektir. Boolean modelin karakteristik bir özelliği olarak, getirilen ve sistem tarafından sorguyla ilgili olduğu varsayılan belgeler, belgenin sorguyu ne oranda karşıladığını gösteren bir değere göre sıralı değil, gelişigüzel bir düzendedir. Çünkü, *1* (*doğru*) ve *0* (*yanlış*) değerlerine göre çalışan bir sistemde belge sorguyu ya karşılıyordur ya da karşılamıyordur.

Gerçekleştiriminin kolay olmasına karşın Boolean bilgi erişim sistemlerinin bazı eksiklikleri vardır. Boolean bilgi erişim sistemlerinde kullanıcıya, sorgu terimi için terimin sorgu ya da belge içindeki önemini ifade eden ağırlık değeri atama izni verilmez ve sorgu sonucu getirilen belgeler, kullanıcıya ne kadar yararlı olduğuna göre sıralı bir liste halinde değil de rastgele bir sırada sunulur. Bu sorunları adreslemek üzere, literatürde *Genişletilmiş Boolean Erişim Modeli* (*Extended Boolean*

Retrieval Model) tanımlanmıştır. İçerik terimlerinin göreceli ağırlığını da öngörebilen bu model, sorgu cümleciklerinin ağırlıklandırılmasına göre Boolean ve vektör tabanlı erişim modelleri arasında bir kabiliyet sunmaktadır (Salton, 1988; Salton, 1984, ss. 277-285; Salton, Fox ve Wu, 1983). Özetle, kullanıcılar Boolean mantık tabanlı sorgu işleyici makinelerinde istediklerini ifade etmekte güçlük çekmektedirler ve bu güçlüğün kavram tabanlı sorgulama modeli ile aşılması mümkündür.

2.1. Bilgi Tabanı Yaklaşımı

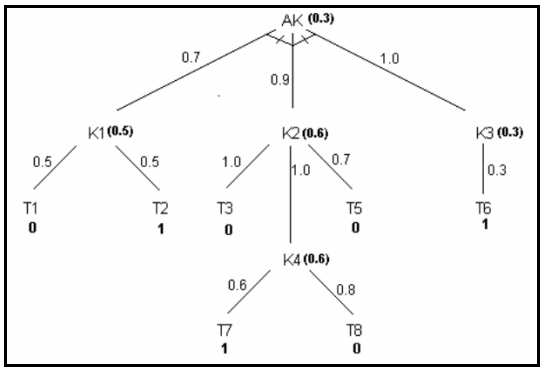
Kullanıcıların bilgi ihtiyaçlarını ifade etme biçimleriyle sorguların belgelerde geçen sözcükler ya da sözcük öbekleri ile ifade edilmesi arasında terminolojik bir uçurum bulunmaktadır. Bu handikapı gidermek için, uzman sistemlerin kullanımını içeren ve kural tabanlı bir bilgi erişim sistemi olan *Rule Based Information Retrieval By Computer (RUBRIC)* geliştirilmiştir (McCune, Tong, Dean ve Shapiro, 1985, ss. 939-944). *RUBRIC* sisteminde, kullanıcının sorgu başlıkları (kavramlar) bir kural tabanında tutulmakta ve bu kural tabanı bir VE/YADA ağacı ile temsil edilmektedir. Sorgu başlığıyla ilgili Ayrışık Normal Form-ANF; (*Disjunctive Normal Form-DNF*) ifadede bulunan birletimlerin (*conjunction*) sayısı *m* ise, bilgi erişim çıktısının VE/YADA ağacının hesaplanmasıyla belirlenmesinin maliyeti, *m*'ye göre üssel olarak verilir.

Kural tabanlı erişim sistemlerinde sorgular, kullanıcıya erişim kriterlerini tanımlama olanağı veren, mantıksal üretim kuralları kümesi ile temsil edilir. Kural, sorgu başlıkları (kavramları) ve sorgu alt başlıklarının bir sıra düzenini tanımlar. Kullanıcı, tek bir kök kavramın adlandırılmasıyla otomatik olarak ağacın sorgulanmasını ister. En alt seviyedeki alt başlıklar düz metin biçimindeki ifadelerden oluşur. Her kural, kullanıcı tarafından atanan sezgisel bir ağırlık taşıyabilir. Bu ağırlık, kullanıcının düşüncesine göre, bir bütün olarak kural örüntüsünün sonuç kavramını (ya da konu başlığını) ne derecede belirlediğini gösterir. Sistem, ilgili başlık altında verilen kurallar kümesi üzerindeki bağımlılık ilişkisini kısmi olarak sıralayarak bilgi erişimini gerçekleştirir. Kural tabanlı sorgu, Boolean bilgi erişim sistemlerinde kullanılan anahtar sözcük ifadelerinden daha karmaşık olabilir. Bu nedenle, kural tabanı yaklaşımının, aynı sorgunun belli periyotlarla tekrar tekrar çalıştırıldığı uygulamalarda kullanılması daha uygun görünmektedir. Bu gibi durumlarda sistemlerin, kullanıcıların sorgu başlığının daha detaylı bir kural tabanı tanımını yapabilmeleri için, çok yoğun bir çaba harcamaları gereklidir.

2.2. Kural Ağacının Hesaplanması

Kavram tabanlı erişimin anahtar özelliği, kural ağacı olarak oluşturulmuş kavramlar yardımıyla, sorguların formülize edilmesine verdiği destektir. İlgilenilen kavram, kural ağacı yardımıyla *yukarıdan aşağıya iyileştirme* (*top down refinement*) stratejisi kullanılarak ifade edilebilir. Yukarıdan aşağıya stratejisinde ilk adım, yapılan isteğin tek bir kavram ile açıklanmasıdır. Oluşturulan kavramın, asıl sorgulanmak isteneni çok soyut olarak temsil etmesi istenir. Bir sonraki adım, ilk verilen kavramın *VE* ya da *YADA* mantıksal işleçleriyle ilişkilendirilmiş kümeler halinde ayrıştırılmasıdır. Ayrıştırma sonucu elde edilen bileşenler farklı bir soyut düzeyde tanımlanmış yeni bir kavram olabileceği gibi, bir metin ifade ya da tek bir dizin terimi de olabilir. Her iki durumda, ayrıştırma sırasında oluşturulan tüm kavram-bileşen (*concept-component*) ikililerine birer ağırlık değeri atanır. Atanan ağırlık değeri, kullanıcının verilen bileşenin (*örüntü parçası*) ilgili kavramı ne oranda karakterize ettiğine olan inancının bir göstergesidir. Kavramın bileşenlere ayrıştırılması işlemi, oluşturulan kural ağacındaki her uç düğümün bir sorgu terimi ya da metin ifade gösterdiği soyutlama düzeyine ulaşılıncaya kadar tekrarlanır.

AK kullanıcının gerçek bilgi ihtiyacını temsil eden ana kavram (sorgu başlığı), K_n ana kavramın parçalandığı alt kavramlar, T_n dizin terimleri ve herhangi bir anda işlenen *D* belgesinde geçen dizin terimlerinin listesi $TD = \{T_2, T_6, T_7\}$ olmak üzere simgesel bir kural ağacı Şekil 1'deki gibi verilmiş olsun.



Şekil 1. İlk ve Hesaplanmış Ağırlık Değerleri ile Simgesel Bir Kural Ağacı

Kural ağacının hesaplanması iki adımda gerçekleştirilir:

Adım 1: TD listesinde yer alan terimleri içeren yaprak düğümlere *1*, kalanlarına ise *0* ağırlık değeri atanır. *Şekil 1*'deki örnekte T_2, T_6, T_7 terimlerine *1*, diğerlerine *0* ağırlık değeri atanmıştır.

Adım 2: Ara düğümlerin ağırlıkları kök düğüme ulaşılıncaya kadar hesaplanarak kural ağacında üst düğümlere doğru yayılır. *Bileşen_*wt_{ik}, *k* kavramının *i.* bileşeniyle ilişkilendirilen hesaplanmış ağırlık değeri ve *Bileşen_Kavram_*wt_{ik} de (*Bileşen_i, Kavram_k*) ikilisiyle ilişkilendirilen kullanıcı tarafından atanmış ağırlık değeri olmak üzere, ağırlık değerlerinin uygulama boyunca yayılması genel olarak iki kural yardımıyla gerçekleştirilir:

i.) VE	*EnKüçük {* *Bileşen_*wt_{ik} * *Bileşen_Kavram_*wt_{ik}*}*	**(Eşitlik 1)**
ii.) YADA	*EnBüyük {* *Bileşen_*wt_{ik} * *Bileşen_Kavram_*wt_{ik}*}*	

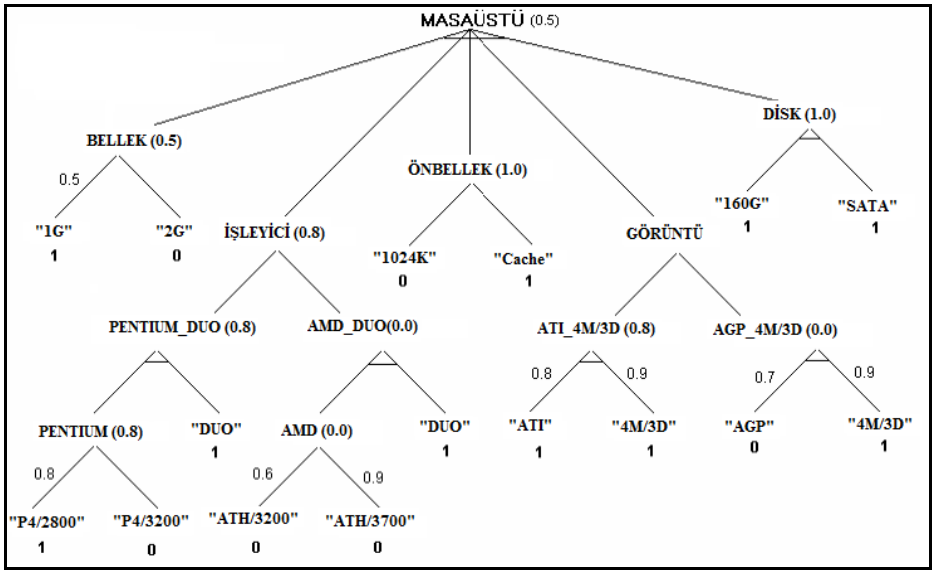
Km_Tn_wt, *Km* kavramıyla *Tn* terimi (bileşeni) arasındaki yayın değeri (kullanıcının atadığı ağırlık değeri) olmak üzere, *Eşitlik 1*'de verilen formüle göre *Km_wt* ağırlık değerleri şu şekilde hesaplanır:

K4_wt =EnBüyük { K4_T7_wt*T7_wt, K4_T8_wt*T8_wt }
=EnBüyük { 0.6*1, 0.8*0 } = 0.6
K1_wt =EnBüyük {K1_T1_wt*T1_wt, K1_T2_wt *T2_wt }
=EnBüyük {0.5*0, 0.5*1 }= 0.5
K2_wt =EnBüyük {K2_T3_wt *T3_wt, K2_K4_wt *K4_wt, K2_T5_wt*T5_wt }
=EnBüyük {1.0*0, 1.0*0.6, 0.7*0 }= 0.6
K3_wt =K3_T6_wt *T6_wt =0.3*1= 0.3
AK_wt =EnKüçük {AK_K1_wt*K1_wt, AK_K2_wt*K2_wt, AK_K3_wt*K3_wt }
=EnKüçük {0.7*0.5, 0.9*0.6, 1.0*0.3 }= 0.3

Kavramların hesaplanan ağırlık değerleri, Şekil 1'deki kural ağacı üzerinde parantez içinde gösterilmiştir.

Şekil 2, kullanıcı tanımlı bir kavram olan *Masaüstü* için oluşturulmuş örnek bir kural ağacını göstermektedir. Şekilde, yaprak

düğümlerde yer alan ve çift tırnak arasına yazılmış metinler dizin terimlerini, ara düğümler alt kavramları, kavram ve bileşenleri bağlayan kenarlar üzerine yazılmış sayılar da ağırlık değerlerini göstermektedir. Değeri verilmeyen kenarların ağırlığı *1*'dir. *Masaüstü* kavramı, önce *VE* işleciyle bağlanmış *BELLEK*, *İŞLEYİCİ*, *ÖNBELLEK*, *GÖRÜNTÜ* ve *DİSK* alt kavramları olarak beş bileşene ayrılmıştır. Bu özel durumda ayrıştırım sonucunda elde edilen tüm bileşenler birer kavramı ifade etmekle birlikte, *MASAÜSTÜ* kavramının altındaki bir soyutlama düzeyinde tanımlanmıştır. Bu beş kavram da tıpkı ilk kavram gibi bileşenleri cinsinden tanımlanmıştır. *Şekil 2*'de görüldüğü gibi, *BELLEK* kavramının bileşenleri *1G* ve *2G*, *ÖNBELLEK* kavramının bileşenleri *1024K* ve *Cache*, *DİSK* kavramının bileşenleri de *160G* ve *SATA* dizin terimleri olarak verilmiş ve bu üç kavram için ayrıştırma bitmiştir. *İŞLEYİCİ* ve *GÖRÜNTÜ* kavramlarıysa, *PENTIUM_DUO* ve *AMD_DUO* ile *ATI_4M/3D* ve *AGP_4M/3D* alt kavramlarına ayrıştırıldığından, ayrıştırma işlemi onlar için devam edecektir. Ayrıştırma işlemi, her uç düğüm bir dizin terimini gösterene dek sürer. Verilen örnekte bu duruma *PENTIUM* ve *AMD* kavramlarının ayrıştırılmasıyla ulaşılır.



Şekil 2: Masaüstü Kavramı için Kural Ağacı ve Zamanı Değerleri

Kural ağacının çalışma zamanındaki hesaplanma adımlarını göstermek için Şekil *2* 'deki kural ağacını ve "*p4/2800, duo, 1g, 160g, sata, ati, 4m/3d*" dizin terimlerini içeren *D* belgesini ele alalım. Kural ağacındaki herbir bileşene, terim ağırlıklarının yukarıya doğru yayılması sonucunda atanan ağırlık değerleri *Şekil 2*'de parantez içlerinde verilmiştir. İşlem, ağacın *D* belgesinde yer alan dizin terimlerini içeren yaprak düğümlerine *1*, kalanlarına *0* ağırlık değerinin atanmasıyla başlar. *D* belgesinin ara düğümler için olan önemini belirlemek için ağırlıkların yukarıya doğru yayılması *Eşitlik 1*'den yararlanılarak gerçekleştirilir.

Kök kavrama (kullanıcının ilgilendiği ana kavram) atanan ağırlık değeri, o an işlenen belgenin Erişim Durum Değerini --EDD, (*RSV, Retrieval Status Value*)-- verir. Örnek olarak verilen *D* belgesi için erişim durum değeri *0.5*'dir. Genel olarak söylenirse, EDD, belgenin kullanıcıya olan yararlılılığının erişim sistemi tarafından belirlenen tahmini bir göstergesidir.

Kullanıcı tanımlı kavramın hesaplanması, verilen kavramın kural ağacının analizini gerektirir. *RUBRIC* sisteminde hesaplama işlemi kural ağacının çalışma zamanında (*run time*) aşağıdan yukarıya olarak gerçekleştirilir. *Çalışma zamanı* terimi burada, kural ağacı analizinin sadece ağacın belge veri tabanına uygulanması sırasında olacağı gerçeğini göstermek için kullanılmıştır. Diğer bir deyişle, etkinlik sebepleri yüzünden kural ağacının durağan bir analizi söz konusu değildir (Lu, Johnsten, Raghavan ve Traylor, 1999).

2.3. Kavram Tabanlı Erişimde En Küçük Terim Kümelerinin (ETK) Kullanımı

Bilgi erişim sistemlerinin temel amaçlarından birisi kullanıcı tercihlerini düzgün bir biçimde ele almaktır. Bazı erişim makineleri kullanıcılara gelişmiş sorgu formları sunarak terimlerin ağırlıklandırılmasına ve erişim listesi çıktısının biçimlendirilmesine imkan tanır. Kullanıcının bilgi ihtiyacını ifadelendirmesinde kullandığı terminoloji ile, belge yazarlarının kullandığı terminolojiler arasındaki olası boşluğu adreslemek, bilgi erişim sistemleri üzerinde yapılan çalışmaların en önemli uğraş alanını teşkil eder. Kullanıcılar ve dizinlenen belgelerin yazarları arasındaki terminolojik boşluğu doldurmak için gömü (*thesaurus*) kullanımı ve çıkarsama (*inference*) modelleri ile bu iki yaklaşım üzerinde geribildirim

tekniklerinin uygulanması gibi birçok yaklaşım vardır (Salton, 1988; Raghavan ve Yu, 1979, ss. 240-260).

Kavramsal düzeyde erişim, betimsel düzeydeki erişimin üzerine inşa edilerek onu tamamlar. Sorgu başlığını belirtmek için tanımlayıcıların ya da metin ifadelerinin kombinasyonundan oluşan kurallar kullanılır. Kural, sorgu başlığının birbirine *VE* işleçiyle bağlı alt sorgular biçiminde tanımlanmasıdır. Mantıksal *YADA* işleci, aynı başlığın alternatif tanımlarına ulaşmak için kullanılır. *Şekil* 3'te Masaüstü kavramı için verilen kurallar kümesi görülmektedir. Türetilmiş terimler (alt başlıklar) büyük harflerle, belgelerde geçmesi düşünülen terimler (metin referansları) çift tırnak arasında gösterilmiştir. İfadenin değerlendirilmesi açısından *VE* (*&*) ve *YADA* (|) işleçleri aynı önceliğe sahiptir ve eşleştirme soldan sağa doğrudur.

İŞLEYİCİ & BELLEK & ÖNBELLEK & DİSK & GÖRÜNTÜ.....	MASAÜSTÜ
(PENTIUM | AMD) & "DUO".....	İŞLEYİCİ
["P4/2800", 0.8] | "P4/3200".....	PENTIUM
["ATH/3200", 0.6] | ["ATH/3700", 0.9].....	AMD
["1G", 0.5] | "2G".....	BELLEK
"1024K" | "Cache".....	ÖNBELLEK
"160G" & "SATA".....	DİSK
(["ATI", 0.8] | ["AGP", 0.7]) & ["4M/3D", 0.9].....	GÖRÜNTÜ

Şekil 3: Masaüstü Kavramı için Kural Tabanı Tanımı

Sorgu başlığı için verilen kurallar kümesi --VE/YADA yayları içeren-- bir amaç ağacı oluşturur. Yayların ağırlıklarını birleştirmek için *VE* yayında en küçük, *YADA* yayında ise en büyük yayılmış (*propagated*) ağırlık değeri kullanılır.

Kavramsal düzey modülü, standart betimsel düzey erişim makinesinin üstünde gerçekleştirilebilir. Kavramsal düzey modülüyle erişim makinesinin etkileşiminde kullanılan standart yaklaşım, sorgunun gönderilip, her belge için, ağacın yaprak düğümlerinde bulunan terimlerin işlenen belgede var olanlarından oluşan bir terim listesinin döndürülmesidir. Bu bilgi daha sonra belgenin, kural tabanı için EDD'sinin hesaplanmasında kullanılır. *VE/YADA* amaç ağacının üretimi ve hesaplanması yavaş gerçekleştiğinden, kavramsal düzey ile erişim makinesi arasındaki etkileşim için yeni bir model sunulmaktadır. Bu yaklaşımda, sorgu başlığı için verilen kural tabanı ayrışık (*disjunctive*)

normal biçime çevirilir ve buna *En Küçük Terim Kümesi* (*ETK, --Minimal Term Set, MTS*) adı verilir. Burada *En Küçük* bilinen anlama sahiptir yani, ETK'deki bazı sorgu terimleri belge içinde görünmezse, o belgenin EDD'si -- bu terimlerin hepsini içeren belgeye göre-- azalacaktır. Ayrıca, eğer sorgu terimleri belge terimlerinin bir alt kümesini teşkil ederse, ilgili belgenin verilen sorgu için hesaplanacak EDD'sinde artık terimler gözönünde bulundurulmayacaktır. Daha önce verilen *Masaüstü* kavramı için ETK dönüşümü *Şekil 4'* te verilmiştir. ETK içindeki her birletim (*conjunct*) bağımsız bir sorgu olarak düşünülür ve sisteme gönderilir. ETK modeli, kural tabanlı erişim modellerinde amaçlandığı gibi, kullanıcıya var olan sorgu başlıkları kümesi içinden bilgi ihtiyacını en iyi karşılayacak olanları belirli bir sıradan seçme olanağı verir.

```
[{"P4/2800", "DUO", "2G", "1024K", "160G", "SATA", "ATI", "4M/3D"}, 0.8]
[{"P4/2800", "DUO", "2G", "Cache", "160G", "SATA", "ATI", "4M/3D"}, 0.8]
[{"P4/2800", "DUO", "2G", "1024K", "160G", "SATA", "AGP", "4M/3D"}, 0.7]
[{"P4/2800", "DUO", "2G", "Cache", "160G", "SATA", "AGP", "4M/3D"}, 0.7]
[{"P4/2800", "DUO", "1G", "1024K", "160G", "SATA", "ATI", "4M/3D"}, 0.5]
[{"P4/2800", "DUO", "1G", "1024K", "160G", "SATA", "AGP", "4M/3D"}, 0.5]
[{"P4/2800", "DUO", "1G", "Cache", "160G", "SATA", "ATI", "4M/3D"}, 0.5]
[{"P4/2800", "DUO", "1G", "Cache", "160G", "SATA", "AGP", "4M/3D"}, 0.5]
```

Şekil 4: P2/233 Masaüstü Alt Kavramı için ETK

2.3.1. VE/YA DA Amaç Ağacı

D_i derlemdeki *i.*belge, T_j belgeleri dizinlemek için kullanılan terimler kümesinin *j.*elemanı ve w_{ij} de *j.*terimin *i.*belgedeki göreceli önemi olmak üzere; $W(D_i, T_j) = w_{ij}$ olan bir ağırlık atama işlevi olsun. Eğer ağırlıklara terim adıyla erişmek mümkünse, ağırlık atama işlemini gerçekleştirmek için dizin kütüğü ya da devrik dizin kütüğü adlı yardımcı yapıdan yararlanılır.

Prolog veri tabanında devrik dizin kütüğü bir *gerçekler* (*facts*) kümesi olarak gösterilir. Eğer D_j belgesinin içinde T_i terimi varsa, $T_i(D_j, w_{ji})$ gerçeği Prolog veri tabanında yer alır.

Sıralı *(ID#, EDD)* ikililerden oluşan bilgi erişim çıktısını üreten bir algoritma, VE/YADA ağacının işlenmesi sonucu elde edilebilir. Algoritmaya girdi olarak verilen iki kütük vardır. Ters dizin kütüğüne karşılık gelen gerçekler (*facts*) kümesi ve sorgu başlığı için verilen bir kurallar kümesi. Algoritmanın çıktısı, verilen sorgu başlığı için VE/YADA

amaç ağacını hesaplamakta kullanılacak Prolog benzeri bir koddur. Algoritmanın temel adımları şöyle verilebilir:

Adım 1: Aynı başlığa sahip olan kuralları *YADA* bağlacıyla birleştir ve sonra her bir kural için --sorgu başlığının bir ya da daha fazla terimden oluşan birletimler kümesinden oluştuğu ve her bir birletimin bir sonrakine *YADA* bağlacıyla bağlandığı-- ayrışık normal formu (ANF) yarat.

CF_i, *yapı* (*structure*) ya da *izleç* (*functor*) olabilen bir birletim olmak üzere sorgu başlığı $QT = \{ CF_1, CF_2, ..., CF_n\}$ olarak verilir. Bir alt sorgu ya da metin sözcük ona kullanıcının sağladığı bir ağırlık atanırsa *yapı*, aksi takdirde *izleç* olarak adlandırılır. Bir izleç için öngörülen ağırlık *1* olduğundan, sorgu başlığını temsil eden birletimlerin ayırtlamı oluşturulurken ağırlık değeri olarak *1* atanır. *Şekil 5*'de, Şekil *3*'te verilen kural kümesi için bir arabiçim verilmiştir.

```
Katman 0:
        AMD       = { {["ATH/3200", 0.6]}, {["ATH/3700", 0.9]} }
        BELLEK    = { {["1G", 0.5]}, {["2G", 1]} }
        DİSK      = { {["160G", 1], ["SATA", 1]} }
        GÖRÜNTÜ   = { {["ATI", 0.8], ["4M/3D", 0.9]},
                      {["AGP", 0.7], ["4M/3D", 0.9]} }
        ÖNBELLEK  = { {["1024K", 1]}, {["Cache", 1]} }
        PENTIUM   = { {["P4/2800", 0.8]}, {["P4/3200", 1]} }
Katman 1:
        İŞLEYİCİ  = { {[PENTIUM, 1], ["DUO", 1]},
                      {[AMD, 1], ["DUO", 1]} }
Katman 2:
        MASAÜSTÜ  = { {[İŞLEYİCİ, 1], [BELLEK, 1], [ÖNBELLEK, 1],
                      [DİSK, 1], [GÖRÜNTÜ, 1] }
```

Şekil 5: *Şekil 3*'te Tanımlanan Kurallar için ANF Gösterimi

Adım 2: Sorgu başlığı *QT* ile gösterilsin. Eğer QT_i'nin gövdesinden QT_j'ye bir referans varsa, QT_i kuralı QT_j kuralına bağımlıdır denir. Bu sınıflama, kural kümesi üzerinde kısmi bir sıra belirler. Anlatım kolaylığı açısından, kural kümesi üzerinde tam bir sıra tanımlanacak olursa: Eğer QT_j, QT_i den daha düşük bir katmandaysa ya da QT_i ve QT_j aynı katmandalarsa ve QT_j alfabetik olarak QT_i'den daha küçükse

$QT_j \leq QT_i$'dir. Kural kümesi, ileriye doğru zincir şeklinde, yani önce düşük seviyedeki kurallar işlenerek değerlendirilir.

Adım 3: Kuralların, ikinci adımda tanımlanan biçimde sıralandığı varsayılsın. Üçüncü adım ilk kuraldan başlanarak kural kümesindeki her bir kural için yinelenecektir. $[T_i, w_i]$ ikilisi w_i ağırlığına sahip *i.*terim olmak olmak üzere, *k.*kuraldaki *m.*birletim $CF_{km} = \{[T_1, w_1], [T_2, w_2], ..., [T_n, w_n]\}$ olarak verilir. Bir cümle ile söylenirse; *ID#* ve *EDD* ile gösterilen belge eğer CF_{km}*(ID#, EDD)* önermesini sağlarsa *CFkm(ID#, EDD)* önerme kümesinin içindedir. Benzer bir yorum T_i*(ID#, EDD)* için de geçerlidir. Burada T_i, bir alt sorgu ya da metin referansdır.

Prolog benzeri bir kuralın üretilmesi şunu gösterir: bir belge tüm T_i terimlerini ($1 \leq i \leq n$) içeriyorsa CF_{km} içerisindedir. Bu belge için *EDD* aşağıdaki gibi hesaplanır. CF_{km} içindeki belgenin *EDD*'sinin hesaplanmasında T_i'nin iki rolü vardır;

- EDD_i ile gösterilen, belge terimi olarak belgedeki önemi.
- w_i ile gösterilen, sorgu terimi olarak kullanıcı sorgusundaki önemi.

Bu iki önem faktörü, terimin tüm etkisini bulabilmek için çarpılır ve sonuç N_i ile gösterilir. Eğer CF_{km} içinde tek bir terim varsa bu, belgenin *EDD*'sini hesaplamak için yeterli olacaktır. CF_{km} tanımında birden çok terimin olduğu durumda tüm *N*'lerin en küçüğü alınır. Başka bir deyişle, belgenin CF_{km} önerme kümesi içindeki *EDD*'sinin hesaplanması, tüm terimlerin birleşik etkisinin tüm terimler içindeki en küçük *N* değerine özdeşleştirilmesine bağlıdır.

Verilen sorgu başlığındaki her birletim, ana sorgu başlığına varmak için kullanılan alternatif bir yol sunan bağımsız birer alt sorgu olarak düşünülür. Birçok birletimin olduğu durumlarda bir belgenin, erişim çıktısında birden fazla kez yer almasını önlemek için örtüşen durumların (belge birden fazla önerme kümesinde yer alabilir) dikkate alınması gereklidir.

C(m, i), *m* birletim için belgenin sağladığı *i*'li birletimlerin kombinasyon sayısını verir. *i* genişliğindeki örtüşen durumların sayısı *C(m, i)*'dir. Bu nedenle, tekrarlamayı önlemek için $\Sigma_{i=1} C(m,i=2^m-1$ durumunun dikkate alınması gerekmektedir.

$\{0,1\}^m - \{0^m\}$ düzenli ifadesiyle üretilen kümenin içinde yer alan *m* genişliğindeki ikili dizgiler kullanılarak herbir bölge kodlanır ve her *r* bölgesi için karşılık gelen bir QT_k^r üretilir. QT_k^r*(ID#, EDD)* önerme

kümesinde yer alan belgenin *EDD*'si, QT_k için *r* bölgesinde yer alan alternatif tanımları sağlayan belgenin en büyük *EDD* değerine eşittir. Örneğin *m=3* için *7* bölge olsun, $(101)_2=5_{10}$ kodlama değeri, birinci ve üçüncü birletimleri sağlayan belgeleri içeren QT_k^5 önermesini gösterir.

Yukarıda verilen algoritmada üçüncü adım en fazla zaman gerektiren adımdır. *k* sorgu ana başlığı ve alt sorgu başlıklarının sayısının toplamı; *m* bir sorgu başlığında olabilecek en fazla birletim sayısı; ve *n* de bir birletim de yer alabilecek en fazla terim sayısı olsun. *k* tane ANF'ye sahip olunduğu, her ANF için en fazla *m* tane birletim olabileceği, her birletim için en fazla *n* terimli önermeler belirtilmesinin gerekeceği ve tekrarlı olmayan ANF'lerin değerlendirilmesi için herbiri *m* birletimli 2^m durumun ele alınması gerektiği için, üçüncü adım en çok $O(km(n+2^m))$ kez yer alacaktır.

Kolayca görülebileceği gibi, VE/YADA hesaplanmasıyla ilgili zaman karmaşıklığı, *J*, önerme kümesini hesaplamakta harcanan çaba olmak üzere $O(km(n+2^m)J)$ ile sınırlıdır.

2.3.2. En Küçük Terim Kümesi (ETK)

VE/YADA amaç ağacının hesaplanmasında adreslenmesi gereken iki nokta vardır. Birincisi, *VE/YADA* ağaç üretme algoritması vektör tabanlı modelleri ele alabilir (Vektör Uzayı Modeli, VUM). Aslında, belge terimlerinin ağırlıklandırılmasını sağlamak için algoritmada bir değişiklik yapmak gerekmemektedir. Bununla beraber, yöntemin VUM terimleriyle uyuşmayan bir durumu da vardır. VE/YADA hesaplanmasındaki sorgu işleme, belgenin, ANF biçimindeki kural kümesinin hiçbir birletimini sağlamazsa, ilgili erişim çıktısında yer alması mümkün olmamaktadır. Bu durum VUM yerine Boolean sorgu modelini çağrıştırmaktadır. İkinci nokta ise VE/YADA amaç ağacının hesaplanma maliyetinin üssel olarak verilmesidir.

2.3.2.1. İkili Ağırlıklar ve Basit Erişim İşlevi

En Küçük Terim Kümesi (ETK), sorgu başlığının *Şekil 4*'te görüldüğü gibi dizin terimleriyle ifade edilmesi sürecinin sonucudur. Burada gösterilmek istenen, VE/YADA amaç ağacı ya da ETK hesaplanması yoluyla aynı erişim çıktısına ulaşıldığıdır.

ETK üretme algoritmasının girdisi sadece sözdizimsel olarak doğru ve döngüsel olmayan bir başlık tanımıdır. *VE/YADA* üretim algoritmasının ilk iki adımı, kuralların ANF biçimine çevirilmesi ve

bağımlılık ilişkileri gözetilerek kurallar üzerinde bir sıralamanın tanımlanması işlemleri, *ETK* üretme algoritmasında da tekrarlanır. ETK üretiminin üçüncü adımı Şekil *6*'da gösterilmiştir.

Kuralların (ANF'lerin) sayısı *k*, birletimlerin sayısının en büyük değeri *m*, birletim içindeki terim sayılarının en büyüğü *n* olmak üzere, algoritmanın en kötü durumdaki zaman karmaşıklığı $O(k^2m^n)$ olarak verilir.

Teorem 1: *QT*, sözdizimsel olarak doğru olup döngüsel olmayan bir sorgu başlığı olsun. $QT_{VE/YADA}(ID, EDD)$ ve $QT_{ETK}(ID, EDD)$, sırasıyla VE/YADA amaç ağacı ve ETK kullanılarak hesaplanmış belgeler ve onların erişim durum değerlerinden oluşan ikili kümelerini göstersin. Terim ağırlıklarının ikili olduğu varsayılsın. Bu durumda ETK'nin değerlendirilmesi aşağıdaki iki aşamada gerçekleştiriliyorsa

$$QT_{VE/YADA}(ID, EDD) = QT_{ETK}(ID, EDD)$$

eşitliği vardır.

1. Metin referanslarının her bir birletimi için, Boolean sorgu işlenişini kullan ve sonra birletim ağırlığını erişim çıktısındaki her belgeyle ilişkilendir.

2. Tüm birletimler bittiğinde, tekrarlı belgeleri, en büyük ağırlıklı olanları tutarak ele.

İspat: *RUBRIC* sistemi tarafından üretilen prolog benzeri kodun işlenişi, bulanık erişim modelindeki bir sorgunun işlenişine benzemektedir. Öteki deyişle, $T_i(ID\#, EDD_i)$ önerme terimi, EDD_i değerini *ID#* belgesindeki T_i teriminin doğruluk derecesi olarak döndüren bir üye işlevi olarak düşünülür. Dahası, örneğin bir kural T_i ve T_j adlı iki önerme içeriyorsa, bu kuralı sağlayan belge, ağırlıklandırılmış iki EDD'den en küçük olanına sahip olacaktır. Aşağıdaki kural bir birletim için ele alınsın.

$$P(ID\#, EDD) :- T_1(ID\#, EDD_1), T_2(ID\#, EDD_2), \ldots, T_n(ID\#, EDD_n),$$
$$N_1 = EDD_1 * w_1,$$
$$N_2 = EDD_2 * w_2,$$
$$\ldots$$
$$N_n = EDD_n * w_n,$$
$$EDD = EnKüçük(N_1, N_2, \ldots, N_n)$$

Teoremin hipotezine göre belge terimi ağırlıkları ikilidir. $T_i(ID\#, EDD_i)$ önerme kümesindeki belgenin EDD_i'si *1* olarak hesaplanır.

Böylece, yukarıdaki kuralı sağlayan bir belge için, $N_i=w_i$ ve $EDD = EnKüçük_i \{w_i\}, 1 \le i \le n$ eşitlikleri yazılabilir. Bu nedenle, kural hesaplanması sadece sorgu terimlerinin ağırlıklarına bağlıdır ve *Şekil 6*'da verildiği gibi, *ETK_üret* algoritmasının üçüncü adımında, birletim ağırlığı, birletimdeki sorgu terimlerinin ağırlıklarının en küçüğü alınarak üretilir.

```
/* ANF lerin toplam sırası {QT1, QT2, ..., QTk} olsun */
Sayarak Yinele (t:=1, t=k-1, t:=t+1)
[
   QTt := {CFt1, CFt2, ..., CFtm} ;
   Sayarak Yinele (u:=1, u=m, u:=u+1)
   [
      CFtu = { [T1, W1], [T2, W2],..., [Ti, Wi], ..., [Tn, Wn] } ;
      ea := EnKüçükAğırlıkBul(CFtu)
      QTt'ye Bağlı Her QTd İçin Yinele, (d>t)
      [
         QTd := {CFd1, CFd2, ..., CFdy} ;
         CFdv = { [T1, W1], [T2, W2],..., [Tj, Wj], ..., [Tx, Wx] } ;
         /* v≤y, Tj=QTt */
         Her CFdv İçin Yinele
         [
            Eğer u < m
            İse [
               CFyeni := CFdv ;
               QTd := QTd ∪ {CFyeni} ;
            ]
            CFdv := CFdv – { [Tj, Wj] } ;
            CFtu İçindeki Her [Ti, Wi] İçin Yinele
               CFdv := CFdv ∪ { [Ti, ea*Wj] } ;
         ]
      ]
   ]
]
QTk'deki Her CFki Birletimini CF`ki Biçimine Dönüştür ;
/* CFki = {[T1, W1], [T2, W2],..., [Tn, Wn] }, CF`ki = [{T1, T2, ..., Tn}, EnKüçük(w1, w2, .., wn)] */
```

Şekil 6: ETK Üretiminin Üçüncü Adımı

ETK üretim algoritması hakkında verilen aşağıdaki notları gözden geçirmekte fayda vardır.

Not-1: ETK'nin hesaplanması basit bir tablodan bakma işlemidir. ETK hesaplanması *Teorem 1*'de tanımlandığı gibi iki aşamada gerçekleştirilirse, ilk aşamanın zaman karmaşıklığı, devrik dizin kütüğündeki terim listesini hesaplamak ve iki terim listesinin kesişimini almak için harcanan çaba *J* olmak üzere,
$O(m*[(n-1)*J])$ ile verilir. İkinci aşama için zaman karmaşıklığı, erişim çıktısındaki belgelerin sıralanması için kullanılan algoritmanın karmaşıklığıyla sınırlıdır.

Not-2: $i \neq j$, $CF_i = \{[T_1, \ldots, T_m], c_i\}$ ve $CF_j = \{[T_1, \ldots, T_n], c_j\}$ olmak üzere, CF_i ve CF_j, verilen ETK biçiminde yer alan birletimler olsun. Ve üç yararlı işlev tanıtılsın;

1. $t(CF) =_{def.} \{ T_1, \ldots, T_m \}$, ETK biçiminde yer alan *CF* birletiminin içerdiği terimlerin kümesini döndüren işlevdir.

2. $tt(D) =_{def.} \{ T_1, \ldots, T_n \}$, verilen *D* belgesini temsil eden terimlerin kümesini döndüren işlevdir.

3. $\Upsilon(CF) =_{def.} c$, ETK biçiminde yer alan *CF* birletimiyle ilişkili sabit değeri döndüren işlevdir.

Verilen bir birletim ve belge için, terim eşlemeye dayalı Boolean erişim işlevi, izleyen karakteristik işlev yardımıyla tanımlanabilir:

$\beta(CF, D) = \begin{Bmatrix} 1 : \mathrm{t(CF)} \subseteq \mathrm{tt(D)} \\ 0 : \text{Aksi takdirde} \end{Bmatrix}$	**(Eşitlik 2)**

$\alpha(CF, D) =_{def.} \beta(CF, D) * \Upsilon(CF)$, ETK değerlendirmesinin erişim işlevi olsun. Kolayca görüldüğü gibi $t(CF_i)$, $t(CF_j) \subseteq tt(D)$ ve $\Upsilon(CF_i) \geq \Upsilon(CF_j)$ eşitlikleri varsa,

$\alpha(CF_i, D) \geq \alpha(CF_j, D)$	**(Eşitlik 3)**

eşitliği de vardır.

Not 2, erişim çıktısındaki belge sayısını sınırlayabilme olanağı tanımaktadır. Aynı zamanda, izleyen birletimlerin hesaplanmasının, erişim çıktısında yer alan belgelerden daha büyük erişim değerine sahip bir belgenin getirilmesinin imkansız olduğunu garanti etmektedir. ETK birletimlerinin kısmi değerlendirilmesi, toplam zaman karmaşıklığını azaltmaktadır. Örneğin Şekil *3'*de verilen masaüstü kavramı için birletim sayısı 32'dir. ETK'nin tam bir hesaplanması, erişim işlevinin her birletim için ters dizin kütüğü üzerinde çalıştırılmasını gerektirir. Bunun yerine, *Not 2*'nin sonucu olarak, en üstteki *n* farklı belge getirildiğinde diğer belgelerin getirilmesi durdurulacaktır.

2.3.2.2. İkili Ağırlıklar ve Genelleştirilmiş Erişim İşlevi

ETK biçiminde yer alan bir birletim, belirtik terimlerin birletiminden oluşur ve onun verilen belge için yorumlanması, belge terimlerinin, birletimin belirtik terimlerini kapsamasını gerektirir. Diğer bir deyişle, belirtik terimlerinin hepsi yalnızca biri haricinde belge terimleri içinde bulunsa bile, ilgili belgenin eldeki belirtim için EDD'si *0*'dır ve belirtimin önerme kümesi içinde yer almaz. Her birletim, sorgu başlığının alternatif bir belirtimini sunsa bile bu, oldukça kuvvetli bir sınırlamadır. Bu yüzden, birletimleri bir terimler kümesi olarak gören başka erişim işlevlerini de düşünmekte fayda vardır. *Eşitlik 4*'te, Boolean terimlerin kosinüs işlevi verilmiştir. Bu erişim işlevi, *Not 2*'nin hipotezinin sağlanması gereklilik koşulunu ortadan kaldırmaktadır. Bununla beraber, hala CF_i'nin değerlendirme kümesinde yer alan belgelerin, EDD'leri en azından CF_{i+1}'in hesaplanmasıyla elde edilecek en büyük EDD'ye eşit olan ($\Upsilon(CF_i) \geq \Upsilon(CF_{i+1})$) bir alt kümesi tanımlanabilir. Bu durum, kosinüs işlevinden elde edilecek en büyük değerin *1* olması gerçeğinin sonucudur.

$\beta'(CF, D) = \frac{\lvert t(CF) \cap tt(D) \rvert}{\lvert t(CF) \cup tt(D) \rvert}$	**(Eşitlik 4)**

Şekil 7'de görülen ETK değerlendiriminde *SorguBaşlığı*, sorgu ağırlıklarına göre azalan biçimde sıralanmış birletimlerin kümesi, *n* ise erişim çıktısı boyunun alabileceği en büyük değerdir.

```
ETK_DEĞERLENDİRME(SorguBaşlığı, n) YORDAMI ;
DIŞYAPILAR ;
   SorguBaşlığı Yapı ;
   n Tamsayı ;
KOMUTLAR ;
   m := |SorguBaşlığı| ;                          /* Terim Listesi Sayısı */
   ErişimÇıktısı := ϕ ;        GeçiciListe := ϕ ;
   ebGetirimDeğeri := 1 ;      ebGeçiciEDD := 0 ;   /* eb : en büyük */

   Sayarak Yinele (i:=1, i=m, i:=i+1)
   [
      ebEDD := ebGetirimDeğeri * γ(CF_{i+1}) ;
      Derlemdeki Her D Belgesi İçin Yinele
      [
         EDD := β`(CF_i, D) * γ(CF_i) ;
         Eğer (EDD > ebEDD) VE (EDD > ebGeçiciEDD)
         İse ErişimÇıktısı := ErişimÇıktısı ∪ {(D, EDD)} ;
            /* D ∉ π_D ErişimÇıktısı(D, EDD) */
         Değilse [
            GeçiciListe := GeçiciListe ∪ { (D, EDD) } ;
            Eğer (EDD > ebGeçiciEDD)
            İse    ebGeçiciEDD := EDD ;
         ]
      ]
      GeçiciListe'nin Her (geçiciD, geçiciEDD) Elemanı İçin Yinele
                                    /* geçiciEDD > ebEDD */
      [
         GeçiciListe := GeçiciListe − { (geçiciD, geçiciEDD) } ;
         ErişimÇıktısı := ErişimÇıktısı ∪ { (geçiciD, geçiciEDD) } ;
            /* geçiciD ∉ π_D ErişimÇıktısı(D, EDD) */
      ]
      ebGeçiciEDD := EnBüyük(EDDListesi(GeçiciListe)) ;
      Eğer [|ErişimÇıktısı| ≥ n]
      İse Döngüden Çık ;
   ]
   GeçiciListe'deki Her (D, EDD) İçin Yinele
   [
      Eğer [|ErişimÇıktısı| ≥ n]
      İseDöngüden Çık ;
      ErişimÇıktısı  := ErişimÇıktısı ∪ { (D, EDD) } ;
   ]
   ETK_değerlendirme := ErişimÇıktısı ; /* D ∉ π_D ErişimÇıktısı(D, EDD) */
ETK_DEĞERLENDİRME BİTTİ ;
```

Şekil 7: Genişletilmiş ETK Değerlendirimi

$\Upsilon(null) =_{def.} 0$ olduğu varsayılsın. *GeçiciListe*, belgeler ve onların EDD'lerinin ikililerinden oluşan sıralı bir kümedir. EDD'lerin azalan sırada tutulması durumu, birleşim işleminin uygulanmasından sonra da korunmaktadır. $R(A_1, A_2, \ldots, A_n)$ ilişkisinin bazı $X \subseteq \{ A_1, A_2, \ldots, A_n\}$ niteliklerine olan izdüşümü $\pi_X R(A_1, A_2, \ldots, A_n)$ ile gösterilir.

Şekil 7'de verilen algoritma, üst limit değerine sahip olan her erişim işleviyle çalışabilecek kadar geneldir. Bu algoritmanın önemi, belge ve EDD ikilisini sadece, belgenin EDD'sinin en az, erişim çıktısına eklenecek kalan tüm belgelerin EDD'leri kadar olduğunu garanti etmesidir.

2.3.3. Kural Tabanının Paylaşılması

Kuralın bir kavrama karşı geldiği varsayılsın. Kavramlar, kullanıcının görüş açısını yansıtan bir kavram sıradüzeni (VE/YADA amaç ağacı) içinde düzenlenir. Çünkü kullanıcı, bir kavram tanımlar ve onun kalıcı olmasını isterse, o kavram kullanıcının kendi belgisinde (*profile*) tutulur. Diğer bir deyişle, kullanıcının tercihlerine saygı göstermek için, kullanıcılara kavramları kendi bakış açılarıyla tanımlamalarına ve başka kullanıcıların tanımlarıyla karışmadan saklamalarına olanak verilmelidir.

Bununla beraber, kullanıcılara yerel bilgi tabanlarını paylaşma imkanı vermek de olurlu bir davranıştır. Bunu yapmak için, kullanıcıların yerel bakışlarını birleştirmeye gerek yoktur. Kullanıcının, paylaşım işlemine bazı kavramlarına dışarıdan erişime izin vererek katıldığı düşünülsün. Her kavramın kesin bir tanımı tutuluyor olsun. Eğer yeni bir kullanıcı, kavramların listesini görmek isterse, kendisinden, istediği kavramı doğrudan dizin terimlerini kullanarak tanımlaması istenir. Kullanıcının tanımı, veri tabanında kalıcı olarak saklanmış ETK'lerle eşleştirilir ve seçilen ETK'lerle ilişkilendirilen kurallardan oluşan sonuç kümesi döndürülür.

Bu yordam, kural ve onun kural tabanındaki ETK'si arasında bire-bir karşılıkların tutulmasını gerektirir.

2.4. Genişletilmiş Boolean Modellerle Erişim ve Belge Terimi Ağırlıkları

Bu bölümde, *p*-Norm (Salton, Fox ve Wu; 1983) modeline dayanarak erişim çıktısının, VE/YADA amaç ağacı ve ETK'nin ikisi için belge terimi ağırlıklarının, belge-sorgu benzerlik hesaplama ölçütü içinde nasıl değerlendirilebileceği üzerinde durulacaktır. Sorgu başlıkları, soyut

sorgu terimleri (*abstract query terms*) ve metin referanslar da somut sorgu terimleri (*concrete query terms*) olarak adlandırılacaktır. Verilen sorgu başlığı için VE/YADA amaç ağacının hesaplanması, izleyen hesaplamanın özyineli olarak uygulanmasına dayanan kural tabanlı bir erişim modelidir.

- Birletimdeki sorgu terimi ağırlıklarından en küçüğünü al.
- Ayırtlamdaki sorgu terimi ağırlıklarından en büyüğünü al.

Buna ek olarak, soyut sorgu teriminin değerlendirilmesinde, aşağıdaki hesaplama da gereklidir:

- Kullanıcı tanımlı ağırlık ve adım 1 ve 2'den elde edilmiş değerin çarpımını al.

Bu kural tabanlı erişim modelinde, belge terimlerinin ağırlıklarının ikili olduğu varsayılmıştır. Bölüm 2.3.2'de ETK gösterimi verilip, VE/YADA amaç ağacı ve ETK değerlendirmesinin aynı belge sıralamasıyla geldiği gösterilmişti. Belgelerin, sorgu terimlerinin birletimler ve ayırtlamlar için en küçük ya da en büyük ağırlık değerlerine göre derecelendirilmesinde kullanılan hesaplama, klasik erişim sistemlerinde olduğu gibi, belgeler arasındaki ayırdedilme eksikliğinden olumsuz yönde etkilenir. Açıkça söylenmese bile (McCune ve diğerleri, 1985), VE/YADA amaç ağacı, belge terimi ağırlıklarının kolayca işlenebileceği bir biçimde tasarlanmıştır. Çünkü, yeterlilik (*implication*) faktörü yerine sorgu terimi ağırlıkları kullanılmıştır. Bu bölümde, belge terimi ağırlıklarının, sorgu belge benzerliğinin hesaplanmasına eklenmesi durumu incelenmektedir. Bu durumda, erişim durum değerlerini kural tabanlı erişimdekilerden daha iyi hesaplayacak bir modele ihtiyaç duyulmaktadır. *Salton* ve doktora öğrencisi E. Fox tarafından geliştirilmiş *genişletilmiş Boolean erişim modeli* (*Extended Boolean Retrieval*) kullanılarak VE/YADA amaç ağacı ve ETK için, sorgu-belge benzerliğinin nasıl hesaplandığı gösterilecektir.

p-Norm Modeli

Boolean model, sorgunun belgeye olan benzerliğinin *n* boyutlu vektör uzayındaki uzantısını yansıtan L_p uzaklık işlevi kullanılarak genişletilir. Genişletilmiş modelde sorgular ve belgeler şu şekilde gösterilir: D, T_i ($1 \leq i \leq n$) belge terimlerine karşılık gelen $W(D, T_i)=d_i$ ağırlıklarından oluşturulan *p*-Norm vektörü olsun. Derlemdeki bazı D belgeleri $(d_1, d_2, \ldots, d_n)$ ile gösterilsin ve genelleştirilmiş *VE* ve *YADA* sorgularının aşağıdaki gibi ifade edildiği varsayılsın.

$Q_{VEp} = (T_1, w_1)\ VE_p\ (T_2, w_2)\ VE_p \ldots VE_p\ (T_n, w_n)$	**(Eşitlik 5)**
$Q_{YADAp} = (T_1, w_1)\ YADA_p\ (T_2, w_2)\ YADA_p \ldots YADA_p\ (T_n, w_n)$	**(Eşitlik 6)**

p katsayısı *1* ile ∞ arasında değişmekte ve kesinlik derecesini göstermektedir (*1*:en az, ∞:en çok).

$SIM(Q_{YADAp}, D) = \sqrt[p]{\frac{w_1^p d_1^p + \ldots + w_n^p d_n^p}{w_1^p + \ldots + w_n^p}}$	**(Eşitlik 7)**
$SIM(Q_{VEp}, D) = 1 - \sqrt[p]{\frac{w_1^p (1-d_1)^p + \ldots + w_n^p (1-d_n)^p}{w_1^p + \ldots + w_n^p}}$	**(Eşitlik 8)**

p'nin değerinin ∞ olduğu durumda eşitliklerin,

- sorgu ve belge terimlerinin ağırlıkları, terimlerin varlığını ya da yokluğunu gösteren ikili değerlerse, Boolean erişim modelindeki sorgu hesaplamaya
- belge terimleri ağırlıklandırılmışsa ve sorgu terimleri hala ikili ise, bulanık küme erişim modelindeki sorgu hesaplamaya indirildiği gösterilmiştir (Salton, Fox ve Wu;, 1983).

Dahası, *p*'nin değeri *1* yapılarak, vektör uzayı modelinde sorgunun hesaplanması için gereken normalleştirilmiş koordinat düzeyi eşleme (kosinüs erişim işlevi olarak da bilinir) elde edilir. *p* için en iyi bir ara değer deneysel olarak belirlenmiştir (Salton, Fox ve Wu;, 1983). Bu değer tipik olarak $2 \leq p \leq 5$ aralığındadır. *p*'nin ara değerleri için, daha iyi EDD'ler, sorgu yapısında verilen küçük kayıplarla elde edilir. Yani, küçük *p* değerleri bazı terim öbeklerini daha az ayırtedebilir yapmaktadır.

2.4.1. Belge Terimi Ağırlıklarının VE/YADA Amaç Ağacına Eklenmesi

VE/YADA_üret algoritmasının iskeleti, terim ağırlıkları eklendiğinde de uygulanabilirliğini korumaktadır. Dahası, algoritmanın 1. ve 2. adımları, hiçbir değişikliğe uğratılmadan kullanılabilir (sorgu başlığının ANF biçimine çevirilmesi, katmanlardaki birletimler için bir sıralamanın kabul edilmesi). Bununla beraber, her kural için karşılık gelen Prolog kodunu

üreten 3.adım *p*-Norm modeli için değiştirilmelidir. Genelliğin kaybolmaması için, *Eşitlik 7* ve *Eşitlik 8* VE_p ve $VEYA_p$ formülleri olarak gösterilsin. VE/YADA amaç ağacı biçiminde verilen bir sorgu olsun. Bu gibi bir sorgu için hesaplama şeması şöyle verilebilir:

- Birletimler için, *EnKüçük* işlecinin yerine VE_p formülünü uygula.
- Ayırtlamlar için, *EnBüyük* işlecinin yerine $YADA_p$ formülünü uygula.

Sorgu başlığının ANF'ye çevirildiği düşünülsün. Bu, sadece kurallar arasındaki bağımlılık ilişkisinin gösterimini değil, aynı zamanda VE_p ve $YADA_p$ formüllerinin gerçekleştirimini de basitleştirir. Bununla beraber, Boolean işleçlerin birleşme ve dağılma özellikleri genişletilmiş Boolean erişim modelinde yer almayacaktır (Salton, Fox ve Wu;, 1983). En fazla, VE_p ve $YADA_p$ işleçlerinin sözde-birleşme özelliklerinden yararlanılabilir. Fakat bu özellik, verilen sorgu başlığını gösteren VE/YADA amaç ağacının eşleniği olan ANF'nin üretilmesi için tek başına yeterli değildir. Bu yüzden, VE/YADA amaç ağacının ANF'si üretilirken p, ∞ değerine kurulur ve sonra, ara bir değere kurulduğu varsayılır. İlk varsayım, klasik Boolean modeldeki karşılığı olan sorgu yapısını verir. İkinci varsayımsa bize vektör uzayı erişim modelindeki sorgu hesaplanmasını taklit etme imkanı sağlar.

2.4.2. Belge Terimi Ağırlıklarının ETK'ye Eklenmesi

Terim ağırlıklı belgeler söz konusu olduğunda, bölüm 2.3.2'de verilen ETK üretimi geçerliliğini kaybedecektir. Çünkü, birletimlerin ağırlıkları, karşı gelen sorgu terimlerinin belgelerde yer aldığı varsayımı altında hesaplanmaktadır. Burada, kural tabanlı erişimde ETK yaklaşımını genişletilmiş Boolean erişime uyarlamak için, bölüm 2.3.2'de öne sürülen iki önermeye bağlı kalınmaktadır. Yani, ETK üretiminde, p değerinin başlangıçta Boolean işleçlerin birleşme ve dağılma özelliklerinden yararlanabilmek için ∞ değerine kurulduğu varsayılmaktadır. Dahası, VE/YADA amaç ağacı durumunun tersine, verilen sorgu başlığı için ETK üretilirken, belge terimlerinin ağırlıkları dikkate alınmamakta ve ETK hesaplanması sırasında p'nin, sorgu yapısını küçük bir kayba uğratacak bir ara değeri kullanılmaktadır. Bu aynı zamanda, her bir birletimin ayrı birer alt sorgu başlığı olarak ele alınmasını önlemektedir. Çünkü, sorgu başlığının ANF'si içindeki birletimin erişim durum değeri, o ANF'nin sonuç EDD'sini belirlemek zorunda değildir. Bu yüzden, belgeleri derecelendirmek için, tüm birletimlerin ayırtlamı hesaplanmalıdır.

Yukarıda verilen kısıtlamalar altında, ETK için VE_p ve $YADA_p$ formüllerine dayanan hesaplama yordamı şöyle tanımlanabilir:

1. Sorgu başlığının ETK gösterimine (somut sorgu terimlerinin birletimlerin ayırtlamı) dönüştürüldüğü varsayılsın.
2. ETK üretiminin sonucu olarak elde edilen birletimin ağırlığı, onun sorgu başlığındaki göreli öneminin uzantısı olarak düşünülür.
3. Her somut sorgu teriminin ağırlığı korunur ve karşılık gelen birletime VE_p formülü uygulanır.
4. Belgenin, sorgu başlığı için EDD'sini bulmak için birletimler üzerinde $YADA_p$ formülü uygulanır.

D belgesi ve *Q* ETK sorgusu arasındaki sorgu-belge benzerliğini VE_p ve $YADA_p$ formüllerini kullanarak hesaplamak için **Şekil** *8*'de bir algoritma verilmiştir. Burada ilk önce, her birletimle *D* belgesi arasındaki uzaklık hesaplanmaktadır. Bu uzaklık, *D* belgesinin EDD'si olarak yorumlanmaktadır. Her birletim alternatif bir yol sunduğundan, birletimin VE_p değeri, bu birletimin tüm terimlerinin *D* belgesinde içerilme derecesini belirler. Yani, karşılık gelen belgenin hesaplanmış belge ağırlığını gösterir. Bu nedenle, tüm bu *hesaplanmış belge ağırlıkları* belirlendikten sonra, belgenin sorgu başlığına olan göreli ağırlığı $YADA_p$ formülü kullanılarak hesaplanır. $YADA_p$ formülünün uygulanmasından sonra elde edilen sonuç, *D* belgesinin erişim durum değerini verir.

Genişletilmiş ETK değerlendirmesinin avantajı, kullanıcı tercihlerine, kullanıcı profili ya da yerel bilginin tek ve genel bir bilgi ağacının yönetiminden daha kolay ve etkin bir biçimde yönetimine olanak verecek biçimde saygı gösteriliyor olmasıdır. Bu yolla, yeni bir kullanıcı, ihtiyaç duyduğu kavramı, Boolean işleçlerle bağladığı kendi sözcükleriyle ifade ederek var olan ilgili sorgu başlıklarını görmek istediğinde, kalıcı olarak saklanmış kurallar kümesi biçiminde gösterilen sorgu başlıklarının ETK'lerinde sorgu-sorgu benzerliğinden yararlanılabilir (Raghavan ve Sever, 1995). Bu sorgu başlıkları, kullanıcının bilgi ihtiyacını karşılama derecelerine göre sıralanmış olarak

/* $D = \{ (T_1, d_{T1}), (T_2, d_{T2}), ..., (T_k, d_{Tk}) \}$ */
/* $Q = \{C_1, C_2, ..., C_n\}$ */
/* $C_j = \{ [(T_1, q_{T1}), (T_2, q_{T2}), ..., (T_m, q_{Tm})], w_{Cj}\}$ */

ETK-Sorgusunu-Değerlendir(D, Q) YORDAMI ;

$V = \{v_1, v_2, ..., v_n\}$; /* $v_j = w_{Jc}$ */

/* *Her* C_j *birletimi için* AND_p *değerini hesapla* */

Sayarak Yinele (j=1, j=n, j:=j+1)

$$AND_p^{C_j} = 1 - \sqrt[p]{\frac{q_{T1}^p (1-d_{T1})^p + ... + q_{Tm}^p (1-d_{Tm})^p}{q_{T1}^p + ... + q_{Tm}^p}} \;;$$

/* *Q için* $YADA_p$*'yi hesapla* */

$$EDD = \sqrt[p]{\frac{v_1^p (VE_p^{C1})^p + ... + v_n^p (VE_p^{Cn})^p}{v_1^p + ... + v_n^p}} \;;$$

ETK-Sorgusunu-Değerlendir := EDD ;

ETK-Sorgusunu-Değerlendir BİTTİ ;

Şekil 8: Genişletilmiş ETK Değerlendirimi

sunulacaktır. Dahası, ETK modülü hala, tanımlayıcılarının ağırlıklarını yardımcı kütüklerde tutan bir betimsel düzey bilgi erişim sisteminin üzerinde gerçekleştirilebilir. Denetimli derecelendirme için orijinal ETK sorgusu içine gömülen kısmi hesaplamanın avantajı, genişletilmiş ETK sorgu hesaplamasını azaltmasıdır. Bu durum, belgenin, izleyen birletimlerle hesaplandığında daha düşük bir EDD'ye sahip olacağının garanti edilememesinin bir sonucudur.

3. Sonuç

Belgelerin (ya da daha genel adlandırmayla biçimsiz verilerin) elektronik ortamlarda saklanması ve bunlar üzerinde erişimi sağlayacak sistemlerin (ya da araçların) bilgi toplumlarındaki önemi yadsınamaz. Verilen bir sorguya göre, İnternet kaynaklarını arama sonuçlarının kalitesinin (duyarlılık ve anma değerlerinin) artırılması için önerilen çözümlerden birisi bilindiği gibi kontrol edilmiş sözlük temeline dayalı RDF/DC türü standartlaşmadır; fakat bu tek çözüm yolu değildir. Diğer bir yol ise, bu çalışma kapsamında modellenen, kavram tabanlı arama makineleridir. Kullanıcının bilgi ihtiyacını karşılamada terimler arasındaki ilişkileri kullanarak alternatif yollar sunan model; betimsel bilgi erişim sistemleri üzerinde, hiç bir kural veri tabanı gerektirmeksizin çalışabilme ve verilen bir eşik değer üzerinde kısmi hesaplama yapabilme gibi yetenekler sunmaktadır.

Sonuç olarak, bu çalışma kapsamında doğruluğu analitik olarak gösterilen kavram tabanlı arama modelinin gerçekleştirilmesi ve kullanıcı profillerinin tanımlanması ile arama sonuçlarının kalitesinin artacağını söylemek yanlış olmayacaktır.

Kaynakça

Alsaffar, A., Deogun, J., Raghavan, V.V., ve Sever, H. (1999). Concept based retrieval by minimal term sets. *International Symposium on Methodologies for Intelligent Systems* içinde (ss. 114–122). London: Springer-Verlag.

Belkin, N., Cool, C., Croft, W. ve Callan, J. (1993). The effect of multiple query representation on information retrieval system performance. *Proceedings of the 16th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Pittsburg 27 June–1 July 1993* içinde (ss. 339–346). Pennsylvania: ACM Press.

Belkin, N., Kantor, P., Fox, E. ve Shaw, J. (1995). Combining the evidence of multiple query representations for information retrieval. *Information Processing & Management, 31*(3), 431–448.

Bookstein, A. ve W. S. Cooper (1976). General mathematical model for information retrieval systems. *Library Quarterly*, *46*(2),153–167.

Croft, W. (2000). Combining approaches to information retrieval. *Advances in Information Retrieval. Kluwer Academic Publishers* içinde (ss.1–36). Boston: Kluwer Academic Publishers.

Gauch, S., Wang, G., Gomez ve Profusion, M. (1996). Intelligent fusion from multiple, distributed search engines. *Journal of Universal Computer Science, 2*(9), 637–649.

Klir, G. J., Clair, U.H. ve Yuan, B. (1997). *Fuzzy Set Theory: Foundations and Applications*. New Jersey: Prentice Hall.

Lee, J. H. (1995). Combining multiple evidence from different properties of weighting schemes. *Proceedings of the 18th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval* içinde (ss. 180–188). Seattle: ACM Press.

Lu, F., Johnsten, T. D., Raghavan, V. V., ve Traylor, D. (1999). Enhancing Internet search engines to achieve concept-based retrieval. *InForum'99- Improving the Visibility of R & D Information*, *8*(1), 30–36.

McCune, B. P., Tong, R. M., Dean, J. S., ve Shapiro, D. G. (1985), RUBRIC: A system for rule-based information retrieval. *IEEE Transactions on Software Engineering*, *11*(9), 939–944.

Raghavan, V. V. ve H. Sever. (1995). On the reuse of past optimal queries. *Proceedings of 18th ACM International Conference on Research and Development in Information Retrieval (SIGIR'95)* içinde (ss. 344–351). Seattle: ACM Press.

Raghavan, V. V. ve Yu, C. T. (1979). Experiments on the determination of the relationships between terms. *ACM Transactions on Database Systems, 4*(2), 240–260.

Salton, G. ve McGill, M. J. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.

Salton, G. (1984). The use of extended Boolean logic in information retrieval. *SIGMOD Conference* içinde (ss. 277–285). Boston: ACM Press.

Salton, G. (1988). *Automatic text processing*. Reading: Addison-Wesley Publishing Company.

Salton, G. ve Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, *24*(5), 513–523.

Salton, G., Fox, E.A. ve Wu, H. (1983). Extended Boolean information retrieval. *Communications of the ACM, 26*(11), 1022–1036.

Saracevic, T. ve Kantor, P. A. (1998). Study of information seeking and retrieving. III. searchers, searchs, and overlap. *Journal of American Society for Information Science, 39*(3), 197–216.

Shannon, C.E. ve Weaver, W. (1964). *The mathematical theory of communication*. Urbana: University of Illinois Press.

Silberschatz, A., Jorth, H.F. ve Sudarshan, S. (1997). *Database system concepts*. New York: McGraw Hill.

Van Rijsbergen, C. J. (1975). *Information retrieval.* London: Butterworths.