SAÜ Fen Bilimleri Enstitüsü Dergisi
2 (1998) 117-120

BULANIK MANTIK DESTEKLİ VERİ TABANININ SQL İLE SORGULANMASI

** Aşkın DEMİRKOL* *** Mithat UYSAL*

** İstanbul Üniversitesi, Teknik Bilimler M.Y.O, Avcılar - İSTANBUL*
*** İstanbul Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, Avcılar - İSTANBUL*

ÖZET

Bu çalışmada klasik veri tabanlarının, bulanık mantık ilkeleri ile desteklenerek, daha karmaşık sorgulamalara alternatif cevaplar üretmesi üzerinde durulmaktadır. Bulanık mantık tekniğinden yararlanarak geliştirilen yöntem ile, mevcut veri tabanının daha akıllı duruma getirilmesi ile, SQL'e olan uyumluluğu sağlanmaya çalışılmıştır. Oluşan bulanık veri tabanının, aranılan her türlü bulanık sorgulamaya karşı yeteneği, küçük bir örnek üzerinde denenmiştir.

1. GİRİŞ

Bulanık mantık, bilinen mantığın daha geniş bir aralıkta ifade edilmesi amacıyla ilk olarak 1965 yıllarında Azerbaycanlı matematikçi Ali Asker Lütfi ZADEH tarafından ortaya atılmıştır (1). Klasik mantık, iki değer(0,1), Bulanık mantık ise, çok değerli [0,1] mantık üzerine kurulmuştur(2). Zadeh, daha sonra bulanık mantık üzerine kurulmuş Bulanık Kümeler Teorisini geliştirmiştir. Fuzzy (bulanık) adı verilen küme teorisi, genellikle kesin olmayan belirsiz kavramları ifade etmek için kullanılmaktadır (3). Bulanık nesneler derecelendirme (üyelik işlevleri) yöntemiyle belirsiz miktarlarını gösterirler (4). Uzun, Genç, Eski, Soğuk gibi kavramlar söz konusu bulanık kümelerden bazılarıdır. Bu kümelerin elemanları, derecelendirme sistemine dayalı, üyelik işlevleri ile belirtilmektedir. Bulanık kümeler, karar veren ve yargılayan için oldukça güçlü bir çözüm yöntemi sağlamaktadır.

Bir matematik olarak ortaya çıkan bulanık mantığın yaklaşık on yıla değin, uygulamaya dönük kullanımı, istenilen ivmeyi yakalayamamıştır. Ancak son yıllarda kendisini özellikle proses kontrol uygulamalarında göstermiştir. Bulanık Kontrolörlerle elde edilen sonuçların, PID kontrolörlere göre çoğu kez daha olumlu oldukları görülmüştür (5).

Bulanık mantık, yalnız bulanık kontrolör tasarımında değil, daha bir çok farklı sahada kendisine pratik ve teorik uygulama alanı bulmuş durumdadır.

Bizim buradaki çalışmamızda ise bulanık mantığın, bilgisayar yazılımlarında oldukça önemli olan veri tabanlarının oluşturulmasına yapabileceği katkı araştırılmaya çalışılacaktır.

2. BULANIK VERİ TABANLARI

Veri tabanları, bilgisayar programlarının önemli yazılımlarındandır. Adından da görülebileceği gibi, verinin kendisiyle ilgilidirler. Diğer bir deyişle programın işleyişine katkı yapacak verilerin, bir disiplin içinde düzenlendiği ve daha sonra kullanılmak üzere saklandıkları kütüklerdir (6). Disiplin ile kastedilen; giriş, çıkış ve erişim gibi sorgulamaların, en verimli biçimde sağlanmasıdır.

Yukardaki tanıma uyan ve günümüzde yaygın olarak kullanılan bilinen veri tabanları ile bir çok karmaşık problemlere destek verilebilmektedir. Ancak söz konusu veri tabanları biraz daha akıllı olabilirse, destekleyeceği boyutlar da bir o kadar genişleyebilecektir. Konvansiyonel dediğimiz veri tabanlarını biraz daha akıllı hale getirecek tekniklerden biride Bulanık Mantık'dır. Bulanık Mantığın uygulanmasıyla, daha karmaşık program ve sorgulamaların altından kalkılabileceği veya en azından alternatif olabileceği düşünülmektedir (7).

Bu amaçla, Bulanık Mantığı uygularken basit bir örnekle yöntemi somutlaştırmaya çalışacağız. Örnek olarak İstanbul'un değişik dört semtini farklı sosyolojik kriterlere göre analiz etmeye çalışacağız.

Bu dört semt, öncelikle SQL'e uygun olarak bulanık verilerle donatıldıkdan sonra, SQL desteği ile bulanık sorgulamalar yapılarak, gerekli cevaplar yine bulanık çıkarım yöntemi ile verilmeye çalışılacaktır.

Bulanık sorgulamaların, klasik tekniklerle cevaplandırılması mümkün olabilir. Ancak bizim araştırmamız, yanıtların bulanık yöntemle verilmesi üzerine kurulacaktır. Amacımız, klasik yönteme bulanık teknikle en azından bir alternatif sunabilmektir.

SQL (Structure Query Language), bir bilgisayar veri tabanı sorgulama dilinin yanısıra, veri tabanı uygulamalarında, veri tanımlama, veri tabanı bütünlüğünün ve veri tabanlarına erişimin kontrolü ve güncellenmesi amaçları için gerekli komutlara sahip olan bir alt dildir (8).

Bulanık veri tabanı için gerekli verileri bir örnek çalışmayı göz önüne alarak oluşturmayı, pratik açıdan uygun bulduk. İstanbul'un 4 değişik yerleşim alanını, 4 kritere göre değerlendirmeye çalışacağız. Dört kriter, bulanık kümeler olarak düşünülmüştür. Her bir semt'in sahip olduğu veriler, resmi veya gerçek bilgiler değildir. Ancak bulanık mantık'da uzman veya operatörlerin deneyimleri önemli olduğundan, sözkonusu verilere yaklaşımımız bu açıdan da değerlendirilebilir. Bu çerçeve ile ilgili tablo aşağıda verilmiştir.

Tablo-1: İlçelerin Dört Kritere Göre Genel Görünümü

İlçeler	Gelir Durumu (mil.T.L)	Yeşil Alan ($K.B.m^2$)	Nüfus (bin)	Öğrenim
Etiler	>500	2	400	Lisans Üstü
B.Evler	500	3	700	Üniversite
Z.Burnu	300	4	900	Orta
Esenyurt	150	10	1100	ilk

mil.T.L = milyon T.L
$K.B.m^2$ = Kişi Başına m^2
B.Evler = bahçelievler
Z.Burnu = Zeytinburnu

Böyle bir veri tabanına sahip tablo, SQL ile aşağıdaki şekilde oluşturulur.

```
CREATE TABLE Semtler
(ilçeler CHAR (12), gelir INTEGER, yeşil INTEGER,
nüfus INTEGER, öğrenim CHAR (10)) ;
```

Böyle bir tabloya verileri girmek için, INSERT INTO Semtler ve VALUES (" veriler ") ; , komutlarının tekrarlanmasının yeterli olacağı bilinmektedir.

Oluşturulan veri tabanı ile ilgili bir çok sorgulama yapılması mümkündür. Buradan bulanık yapıya geçmek için, aşağıdaki sorunun sorulduğunu düşünelim.

SORU - 1 : gelir durumu YÜKSEK, yeşil alanı YETERLİ, nüfusu YOĞUN ve öğrenim durumu YÜKSEK olan ilçe hangisidir ? ,

Aynı soruyu SQL de ;

```
SELECT ilçe
FROM Semtler
WITH Gelir = " yüksek " and Yeşil = " yeterli " and
     Nüfus = " yoğun " and Öğrenim = " yüksek "
```

şeklinde sorabiliriz.

Bu soruya, klasik veri tabanında bir dizi araştırma, düzenleme ve hesaplama yaparak cevap bulmak mümkündür. Ancak bulanık mantık yöntemi ile bu işlem biraz daha basit görünmektedir.Bu sorunun cevabını verebilmek için, klasik tabloyu bulanık yapıya aşağıdaki şekilde dönüştürmemiz mümkündür.

Tablo-2: Veri Tabanının Bulanık Görüntüsü

İlçeler	Gelir Durumu (mil.T.L)	Yeşil Alan ($K.B.m^2$)	Nüfus (bin)	Öğrenim
Etiler	yüksek	çok az	az	yüksek
B.Evler	orta	yetersiz	yoğun	yüksek
Z.Burnu	düşük	yeterli	ol.yoğun	orta
Esenyurt	çok düşük	çok	çok yoğun	düşük

Görüldüğü gibi her bir küme (sütun), belirsiz miktarlarla ifade edilmiştir. Böyle bir veri tabanında sorgulama yapmak hemen mümkün değildir. Diğer yandan bulanık terimlerin, insan düşünüş şekline çok yakın oldukları görülmektedir. Bulanık mantık daha çok bu yönüyle ön plana çıkmaktadır. Yukarıdaki şekilde kurulan bulanık tablo yardımıyla verilen soruyu yanıtlamak için bir dizi bulanık işlem yapmamız gerekecektir. Bunun için öncelikle bulanık kümeleri üyelik dereceleriyle ifade etmemiz gerekecek.

GELİR DURUMU

Yüksek = {0.1/300, 0.3/400, 0.5/500, 0.7/600, 0.9/700, 1/800, 1/900, 1/1000}
Orta = {0/100, 0.3/150, 0.5/200, 0.6/250, 0.7/300, 0.8/350, 0.9/400, 1/450, 1/500, 0.6/600}
Düşük = {0/25, 0.3/50, 0.7/100, 1/150, 0.6/200, 0.2/250, 0/300}
Çok Düşük = {0.2/25, 0.5/50, 0.8/75, 1/100, 0.6/125, 0.2/150, 0/175}

YEŞİL ALAN

Çok Az = {0.4/0.5, 0.7/1,1/1.5, 1/2, 0.3/2.5, 0/3}
Yetersiz = {0.3/1, 0.5/1.5, 0.7/2, 1/2.5, 0.6/3, 0.2/3.5, 0/4}
Yeterli = {0/1, 0.3/2, 0.6/3, 0.8/4, 1/5, 1/6, 0.5/7,0.2/8, 0/9}
Çok = {0/2, 0.3/4, 0.5/6, 0.8/8, 1/10}

NÜFUS

Az = {0/50, 0.3/100, 0.5/150, 0.8/200, 1/250, 0.7/300, 0.4/350, 0/400}

Yoğun = {0.1/200, 0.4/300, 0.7/400, 1/500, 0.8/600, 0.5/700, 0.4/800, 0.3/900, 0/1000}

Oldukça Yoğun = {0.2/400, 0.4/500, 0.6/600, 1/700, 1/800, 0.6/900, 0.4/1000, 0.2/1100, 0/1200}

Çok Yoğun = {0.3/700, 0.5/800, 0.7/900, 1/1000, 1/1100, 0.6/1200, 0.3/1300, 0/1400}

ÖĞRENİM

Yüksek = {1/lisans üstü, 0.8/üniversite, 0.4/lise, 0/orta}

Orta = {0/lisans üstü, 0.5/üniversite, 1/lise, 0.7/orta, 0.2/ilk}

Düşük = {0/üniversite, 0.4/lise, 0.6/orta, 1ilk, 0.3/yok}

Oluşturulan bulanık kümelerin üyelik işlevlerinin tesbiti, bu mantığa uygun olarak kabul edilen pratik değerlerdir. Bu derecelendirmelere göre soruyu cevaplamaya geçebiliriz. Bunun için her bir semt için aynı soruyu karşılayacak genel bir derece saptamamız gerekecektir. Bunu hazırlamak için ise, bulanık hesaplamada çok popüler olan " min , max " işlemlerinden istifade edeceğiz. " min ve max " işlemlerinin sorgulamadaki karşılıkları " VE (and) ve VEYA (or) " operatörlerine karşılık gelmektedir (4). Soru AND bağlacı ile bağlanmışsa " min " , OR ile bağlanmışsa " max " operatörü kullanılacaktır. Soruyu her bir semt için aşağıdaki tablo halinde düşünebiliriz.

Tablo-3: Birinci Sorunun Bulanık Çıkarımı

İlçeler	Gelir yüksek (a)	Yeşil Alan yeterli (b)	Nüfus yoğun (c)	Öğrenim yüksek (d)	ÇIKARIM min(a,b,c,d)
Etiler	1	0.3	0.7	1	0.3
B.Evler	0.5	0.6	0.5	0.8	0.5
Z.Burnu	0.1	0.8	0.3	0	0
Esenyurt	0	1	0	0	0

Tablonun sonuç işleminin yapıldığı son sütundaki değerlendirmeye göre ; geliri yüksek, yeşil alanı yeterli, nüfusu yoğun ve öğrenimi yüksek olan, ilçe ; Bahçelievler' dir. Onu Etiler takip ederken, son ikisinin soruya ilişkin bir iddiası bulunmamaktadır. Bahçelievler birinci olmasına rağmen, üyelik derecesi (0.5), göz önüne alınırsa, bunun tam bir birincilik olmadığı görülmektedir. Eğer tam bir birincilik olsa idi, bu değerin " 1 " olması gerekir idi. Buradan bahçelievler'in tam birinciliğini engelleyen eksikliklerin olduğu söylenebilir. Bu tüm kriterlerde tam puan (1) alamadığından da görülebilmektedir. Bulanık mantığa göre düşünürsek söz konusu birinciliğin, " kötünün iyisi " şeklinde yorumlanması akla en yatkın olanıdır. Yukarda bulunan değerler, SQL ile yeni bir veri tabanı gibi oluşturulabilir.

Soru bu kez,

SELECT ilçe
FROM Semtler
WITH Gelir = " orta " and Yeşil = " yeterli or Nüfus = " oldukça yoğun " and Öğrenim = " orta "

yani,

SORU - 2 : gelir durumu ORTA, yeşil alanı YETERLİ veya nüfusu OLDUKÇA YOĞUN ve öğrenim durumu ORTA olan ilçe hangisidir ? ,

şeklinde olsa idi bulanık çıkarımımız nasıl olurdu. Bunun cevabını da aynı yöntemle bulabiliriz.

Tablo-4: İkinci Sorunun Bulanık Çıkarımı

İlçeler	Gelir orta (a)	Yeşil Alan yeterli (b)	Nüfus oldukça yoğun (c)	Öğrenim orta (d)	ÇIKARIM min (a,max(b,c),d)
Etiler	0.6	0.3	0.2	0	0
B.Evler	1	0.6	1	0.5	0.5
Z.Burnu	0.7	0.8	0.6	0.7	0.7
Esenyurt	0.3	1	0.2	0.2	0.2

İkinci sorunun karşılığı olan bu tablo incelendiği vakit, en uygun cevabın " Zeytinburnu " ilçesi olduğu görülmektedir.

Verilen bulanık çözüm yöntemi ile, yukarda istenen daha on'larca bulanık sorunun karşılığını bulmak mümkündür. Çünkü bu metotla, benzer bir çok bulanık veriyi kapsayan tablonun SQL ile, aynı veya ayrı veri tabanları olarak gerçeklenmesi oldukça kolaylaşmıştır. Sonuçta mevcut verilerden yararlanarak hem bulanık veri tabanı, hemde sorgulamalar SQL 'e uyumlu hale gelmiştir.

KAYNAKLAR

(1) - KANDEL,A,Fuzzy Mathematical Techniques with Applications, s.1, 1986

(2) - MİRANDA,V,SARAİVA,J,T, Fuzzy Modelling of Power System Optimal Load Flow, IEEE transactions on power systems, Vol.7, No.2, s.843,May 1992

(3) - KHALID,M,OMATU,S, Adaptive Fuzzy Control of Water Bath Process with Neural Networks, Engineering Application, Artificial İntelligence, Vol.7, No.1, s.39, 1994

(4) - LEE,C,C, Fuzzy Logic in Control Systems : Fuzzy Logic Controller, Part-1, Part-2, EEE Transaction on Systems,Man, and Sybernetic, V.20,No.2, s.404-406, s. 420-421,april 1990
(5) - CHAO,C,T, TENG,C,C, A PD - Like Self - Tuning Fuzzy Controller without Steady - State Error, FUZZY sets and systems,V.87, No.2, s.141, april 1997
(6) - MITTRA,S,S, Principles of Relational Database Systems, s.1-4, 1991
(7) - NAKAJİMA,H,SENOH,Y, A Spreadsheet-Based Fuzzy Retrieval Systems,International Journal of Intelligent Systems, V.11, s.661, 1996
(8) - UYSAL,M, SQL Veri Tabanı Sorgulama Dili, s.XI-12,1994