

ÖZET

BİR VERİ TABANINDA BİR DİZİN YAPISININ SAĞLANMASI YÖNTEMİ

Bir veri tabanında bir dizin yapısının sağlanması yöntemi
5 birden çok türde nesne içermektedir, her türdeki nesne en az
bir nesne içermektedir. Söz konusu veri tabanındaki herbir
nesnenin bir kimlik dizini bit vektörünün bir pozisyonunun
bire ve diğer tüm pozisyonların sıfıra ayarlanması, söz konusu
pozisyon herbir nesne için farklıdır ve söz konusu
10 veritabanında herbir nesne için söz konusu herbir nesnenin
ilişkili olduğu tüm kimlik dizini bit vektörlerini
birleştirilerek bir ilişki dizini bit vektörü oluşturulması
adımlarını içermektedir, burada herbir bit vektörü nesnelerin
sayısına eşit bir uzunluğa sahiptir.

İSTEMLER

1. Nesnelerin her bir türünün en az bir nesne içerdiği, birden çok nesne türü içeren bir veri tabanında bir dizin yapısının sağlanması yöntemi olup, **özelliği**, aşağıdaki adımları içermesidir

Söz konusu veri tabanında her bir nesnenin bir kimlik dizini vektörünün bir pozisyonunun bire ve tüm diğer pozisyonlarının sıfıra ayarlanması, söz konusu pozisyon her bir nesne için farklıdır,

Söz konusu veri tabanında her bir nesne için söz konusu her bir nesnenin ilişkili olduğu nesnelerin tüm kimlik dizini bit vektörleri birleştirilerek bir ilişki dizini bit vektörünün oluşturulması, burada her bir bir vektörü nesnelerin sayısına eşit bir uzunluğa sahiptir, her türdeki nesnelerin tüm nesnelerinin kimlik dizini bit vektörlerini birleştirerek nesnelerin her bir türü için bir dışın bit vektörü içeren bir nesne türü kimlik dizini oluşturulması ve

Tam bir veri seti tutarak bir veri tabanından bir alt kümenin aşağıdaki gibi elde edilmesi

veri tabanındaki söz konusu tam veri tabanından seçilmiş nesneleri içeren verinin bir birinci alt kümesinin belirlenmesi,

verinin söz konusu birinci alt kümesine dahil edilen veya bununla ilgili ve seçilmiş bir kriteri yerine getiren söz konusu tam veri setindeki tüm nesnelerin tanımlanması, ve tüm tanımlanmış nesneleri ve söz konusu tanımlanmış nesneler ile ilişkili tüm nesneleri içeren verinin ikinci bir alt kümesinin oluşturulması.

2. İstem 1'e göre bir yöntem olup, **özelliği** ilaveten aşağıdakileri içermesidir

Tüm nesneleri temsil edebilen kelime uzunluğunun en düşük katına bitlerin sayısını ayarlayarak ve ekstra bitleri sıfıra ayarlayarak hedeflenmiş bir işlemci mimarisinin kelime uzunluğuna her bir dizin bit vektörünün hizalanması.

3. İstem 1 veya istem 2'ye göre bir yöntem olup, **özelliliği** ilaveten veri setinin birden çok bölüntüye ayrılmasını ve her bir bölüntü için aşağıdakileri tekrarlamasını içermektedir:

5

söz konusu veri tabanında her bir nesnenin bir kimlik dizini bit vektörünün bir pozisyonunun bire ve tüm diğer pozisyonların sıfıra ayarlanması, söz konusu pozisyon her bir nesne için farklıdır, ve

10

söz konusu veri tabanında her bir nesne için söz konusu her bir nesnenin ilgili olduğu nesnelerin tüm kimlik dizini bit vektörlerini birleştirerek bir ilişki dizini bit vektörünün oluşturulması, burada her bir bit vektörü nesnelerin sayısına eşit bir uzunluğa sahiptir.

15

TARİFNAME

BİR VERİ TABANINDA BİR DİZİN YAPISININ SAĞLANMASI YÖNTEMİ

TEKNİK ALAN

5

Bilgi üreten kuruluşlar tarafından oluşturulan bilginin çoğu değer zincirinin belirli bir kısmı için optimize edilmiş olan farklı bilgi sistemleri tarafından yakalanmaktadırlar. Çeşitli bütünleşme teknikleri kullanılarak, farklı bilgi sistemlerinden bilgiler kurum tarafından üretilen tüm bilginin kilidini açmak için ilaveten sorgulanabilen ve analiz edilebilen birleştirilmiş veri setlerine entegre edilebilmektedir. Veri setleri birçok kaynaktan birleştirildiğinden ve sıklıkla güncellenebildiğinden ötürü, birleştirilmiş veri setlerini kullanan bilgi sistem, hem sık sık yapılan güncellemeleri hem de veri setlerindeki yapısal değişiklikleri esnek ve kolay bir şekilde yapabilmelidir. Ayrıca, birleştirilmiş veri setlerindeki bilginin kilidinin açılmasını sağlamak için, bilgi sistem, birleştirilmiş veri setlerinin sorgulanmasını ve teknik kısıtlamalar dayatılmaksızın analiz edilmesini sağlamalıdır. Bu şekilde veri setlerine ulaşmadaki zorluk üretilen veriden ortaya çıkan bilgiyi en üst düzeye çıkarmayı hedefleyen birçok bilgi üreten organizasyonunda bir gerçekliktir.

25 ÖNCEKİ TEKNİK

Bu problemleri işaret eden mevcut çözümler bir kullanıcının perspektifinden üç ana kategoriye ayrılmaktadır. Çözümlerin birinci kategorisi kullanıcıya altta yatan veri modeline tam erişim vermektedir ve böylece veri setlerinin esnek bir biçimde sorgulanabilmesini ve analiz edilebilmesini sağlamaktadır. Ancak, kullanıcının altta yatan veri modeli hakkında iyi bir bilgiye sahip olması gerektiğinden ve sorgulama dilinin sıklıkla karmaşık olmasından ötürü, karmaşık sorguların sağlanması için, bu çözüm kategorisi genellikle ortalama bir kullanıcı için kullanımı zor bir kategori varsayılmaktadır. Ayrıca, veri setlerinde gerçekleştirilen

sorgular ve analizler bu kategorinin veri modeli çözümleri üzerinde doğrudan yapıldığından ötürü verinin yapısındaki değişikliklere duyarlı olmaktadır.

5 Çözümlere ilişkin ikinci kategori ise kullanıcıdan altta yatan veri modelini saklamaktadır ve bunun yerine belirli bir problem alanına değiştirilmiş bir bilgi modeli kullanmaktadır ve bilgi modeli ve veri modeli arasında çevirmek için önceden belirlenmiş kurallar kullanmaktadır. Bu çözümleri içeren
10 kategori belirli bir problem alanını sorgulamak ve analiz etmek için değiştirilmiş bir arayüz kullandığından ötürü, veri setleri kolay bir biçimde sorgulanabilmektedir ve analiz edilebilmektedir. Ancak, çözüm belirli bir problem alanına ilişkin olarak oluşturulduğundan ötürü, bu kategoride yer alan
15 birçok sistem veri setlerinin esnek bir biçimde sorgulanmasına ve analiz edilmesine imkân sağlamamaktadır, ancak bunun yerine mümkün olan sorgular ve analizlerin türü bilgi sisteminin tasarımı ile sınırlandırılmaktadır. Buna ilaveten, bu kategorinin çözümleri verinin yapısındaki değişikliklere de
20 son derece duyarlıdır çünkü sistem önceden tanımlanmış veri setlerinin üzerinde oluşturulmuştur.

Çözümlerin üçüncü kategorisi ise kullanıcıya altta yatan veri modeline erişme imkânı sağlamaktadır, ancak birçok karmaşık
25 görevi (örneğin veri setinin ilgili kısımlarına bağlanma gibi) otomatik olarak yapmaya çalışmaktadır. Bu da veri setlerinin sorgulanmasını ve analiz edilmesini esnek aynı zamanda kolay bir şekilde yapmayı olası hale getirmektedir çünkü kullanıcı veri modeline doğrudan ulaşabilmektedir, ancak veri setlerini
30 keşfetmek için veri modelinin tamamını veya karmaşık sorgu dilini bilmesi gerekmemektedir. Ancak, bu kategorinin çözümleri ile birlikte otomatik davranışın üstesinden gelinmesi ve otomatik davranıştan başka bir sorgu ve analiz modelinin oluşturulması sıklıkla karmaşık bir hal almaktadır.
35 Bundan dolayı da veri setlerinin sorgulanması ve analiz edilmesi sırasında bulunan esneklik çözümün tasarımı ile sınırlandırılmaktadır. Teknik uygulamaya dayalı olarak, bu

kategoride yer alan bazı çözümler ayrıca veri setlerinin yapısındaki değişikliklere duyarlıdır.

5 Yukarıda açıklanan çözümlerin tüm kategorileri ya esneklik üzerinde veya da veri setlerinin sorgulanması ve analiz edilmesinin kolaylığı üzerinde kısıtlamalar dayatmaktadır. Kullanılan çözümlerin birçoğu ayrıca veri setlerinin yapısal değişikliklerine duyarlı olmaktan mustarıptirler. Bundan dolayı da, kısıtlamalar dayatmaksızın esnek ve kolay bir 10 biçimde veri setlerinin sorgulanması ve analiz edilmesi için ve öte yandan da veri setlerinin yapısal olarak değişmesine imkân sağlanarak yeni bir yöntem ve teknoloji önererek durumun iyileştirilmesine ilişkin ihtiyaç vardır.

15 WO 00/72165 A1 veri görüntüleri oluşturmak için bit vektörlerini kullanan bir veri marketinden bahsetmektedir.

BULUŞUN KISA AÇIKLAMASI

20 Buluşun bir amacı yukarıda belirtilen eksikliklerin belirlenmesi ve azaltılmasıdır ve veri setlerindeki bilginin daha erişilebilir olması için destekleyici teknoloji ile birlikte yeni bir yöntemin sağlanmasıdır.

25 Çeşitli yapılandırmalarda, insan beyninin bilgiyi nasıl ilişkilendirdiği ve yapılandırdığını gösteren bir veri modeli kullanılabilmektedir. Bu türdeki bir veri modeli kullanılarak, buluş kullanıcıların doğal bir şekilde veri setlerinde ilgili bilgiyi hem bulmalarını hem de anlamalarını sağlamaktadır ve 30 böylece veri setlerinden bilginin elde edilmesini sağlamaktadır. Bu türdeki bir veri modeli aşağıdaki özellikler ile açıklanabilmektedir:

35 1. Her bir bilgi kavramı, literatürde açıklandığı ve o alandaki uzmanlar tarafından anlaşıldığı gibi, nesnelerin bir türü olarak modellenmektedir,

2. Her bir nesne nesnelerin bir türüne aittir,

3. Her bir nesne kendisini anlamlı ve anlaşılır hale getirmek için yeterli bilgiyi içermektedir,

4. Nesnelerin bir türü dâhilinde bulunan her bir nesnenin bilginin aynı türdeki özelliklerine veya sınıflandırılmasına sahip olması gerekmemektedir,

5. Her bir nesne kendisinin ilgili olduğu diğer nesneler hakkında ve her bir ilişkinin ne için olduğuna dair bilgiye sahiptir.

Yukarıda açıklanan veri modeli bilgi üzerinde herhangi bir katı şema dayatmadığından ötürü, veri setlerinin yapısındaki değişikliklerin inceliklerle ele alınması mümkün olabilmektedir. Ayrıca, veri modeli insan beyninin nasıl bilgiyi nasıl tesis ettiğini benzettiğinden ötürü, önceden tanımlanmış hiçbir şema mevcut olmasa dahi kullanıcılar için bilginin anlaşılması kolay olmaktadır.

Buluşun birinci unsuru, yukarıda açıklanan veri modelini kullanan veri setlerinin esnek, kolay ve karşılık veren bir şekilde sorgulanması ve analiz edilmesini sağlayan, uygun dizin yapıları ile birlikte, bir veri tabanı motorudur. Veri tabanı motorunun araştırma yöntemi aşağıdaki adımları içermektedir:

1. Ne türde bilginin geri alınacağı ve hangi filtreleme kriterlerinin uygulanması gerektiği veri modelinin belirlenmesi, kullanılması,

2. Veri tabanı motoruna yukarıdaki belirlemenin sunulması,

3. Veri tabanı motorundan bir sonucun alınması,

4. Kullanıcıya alınan sonucun sunulması,

Çeşitli yapılandırmalarda dizin yapıları bit eşlem esaslıdır.

Veri tabanı motorunun araştırılması için yukarıda açıklanan yöntemin kullanılması ile kullanıcı veri tabanı motorunda içsel olarak veri tabanlarının nasıl yapılandırıldığı bilgisine ihtiyaç duymamaktadır. Bunun yerine kullanıcı ilgi

alanındaki bilginin ne türde olduğu ve bilginin hangi kriterleri yerine getirmesi gerektiğini belirlemelidir. Bu kullanıcının bilgiye odaklanmasını sağlamaktadır ve veri setlerinden bilginin edinilmesini sağlamaktadır.

5

Buluşun ikinci unsuru, veri setlerine karşın sıklıkla sorulması gerekli olan belirli türdeki soruların, bununla birlikte bu türdeki soruların türünü belirlemek için yeni bir yöntem ve terminolojinin tanımlanmasıdır. Soruların bu belirlenmiş türü veri setlerinde dolaylı ilişkiler üzerinde işlemektedir ve kullanıldıkları yerde eğer yalnızca dolaylı ilişkiler var ise saklı olacak olan bilginin elde edilmesi için kullanılabilir. Bu türdeki bir sorunun örneği şu olabilmektedir: nesnelerin belirli bir türü için diğer türlerdeki nesnelerden özniteliklerin nasıl ilişkilendirildiği. Başka bir örnek şu olabilmektedir: nesnelerin belirli bir türü için, hangisi belirli bir kriteri karşılamaktadır, ortak sahip oldukları nesnenin diğer türünden hangi özniteliklerin bulunduğu. Bu türdeki sorular veri setlerinin sorgulanması ve analiz edilmesi sırasında yaygındır, burada belirli türdeki nesneler analizin merkezine konulmaktadır, örneğin klinik araştırma ve sağlık bakımında hastanın merkezi analizi. Buluşun bu unsuru buluşun diğer unsurları ile bağlantılı olarak kullanılabilir, ayrıca diğer veri tabanı motorları ve veri modelleri ile birlikte de kullanılabilir.

Buluşun üçüncü bir unsuru söz konusu veri tabanı motorunun esnek, kolay ve yanıt verici bir şekilde yukarıda açıklanan soru türünü kullanmasını sağlayan buluşun ikinci unsurunun bir açılımıdır. Buluşun bu unsuru bu türdeki soruların buluşun ikinci unsuru için açıklandığı gibi araştırma yöntemi kullanılarak araştırılmasını sağlamaktadır.

Yukarıda açıklanan buluşun unsurlarının bir sonucu olarak, buluş amacı elde etmektedir ve özellikle erişilecek olan veri setlerindeki bilgiyi sağlamaktadır.

ÇİZİMLERİN KISA AÇIKLAMASI

Yukarıda belirtilen ve diğer avantajlar ve buluşun amaçlarının elde edildiği biçimde hâlihazırda anlaşılması için, yukarıda
5 kısaca açıklanan daha özellikli açıklama ekli çizimlerde gösterilen belirli uygulamalara atfen yapılacaktır.

Bu çizimlerin yalnızca buluşun tipik yapılandırmalarını gösterdiğinin anlaşılması ve bundan dolayı da kapsamını
10 sınırlandırıyor olarak varsayılmaması ile, buluş ilave özellik ile birlikte anlatılacaktır ve açıklanacaktır ve ekli çizimlerin kullanımı ile de detaylandırılacaktır, burada:

Şekil 1 açıklanan veri modeli türünde temsil edilen verinin bir
15 örneğini gösteren bir diyagramdır,

Şekil 2 B türündeki nesnelerde verini alt kümesinin bir yansımasının bir örneğini gösteren bir diyagramdır,

Şekil 3 bir veri setindeki her bir nesne için veri tabanı motoru tarafından kullanılan kimlik ve ilişki
20 dizininin nasıl bakabildiğinin bir örneğini gösteren bir diyagramdır,

Şekil 4 veri tabanı motorunun veri setini filtrelemek için ve tüm ilgili bilginin bulunması için ilişki dizinini nasıl kullanabildiğinin bir örneğini gösteren bir
25 diyagramdır,

Şekil 5 veri tabanı motorunun B türündeki nesnelerde verinin alt kümesini nasıl yansıtabildiğinin bir örneğini gösteren bir diyagramdır,

Şekil 6 veri tabanı motorunun iki farklı yansıtma noktası
30 kullanılarak verinin iki alt kümesini nasıl yansıtabildiğinin bir örneğini gösteren bir diyagramdır,

Şekil 7 veri tabanı dizini ile kimlik dizinin nasıl oluşturulduğunu gösteren bir diyagramdır,

35 Şekil 8 veri tabanı dizini ile ilişki dizinin nasıl oluşturulduğunu gösteren bir diyagramdır,

Şekil 9 veri tabanı motorunun bir arama sorgusunu cevaplamak

için dizin yapılarını nasıl kullandığını gösteren bir diyagramdır,

Şekil 10 veri tabanı motorunun yansıtıcı mantık kullanılarak bir arama sorgusuna cevap vermek için dizin yapılarını nasıl kullandığını gösteren bir diyagramdır,

Şekil 11 açıklanan veri modeli türünde temsil edilen iki kök nesne ile birlikte verinin bir örneğini gösteren bir diyagramdır,

Şekil 12 veri tabanı motoru ile kullanılan kimlik dizininin bir bölüntülenmiş veri setinde her bir nesneye nasıl bakabildiğinin bir örneğinin gösteren bir diyagramdır,

Şekil 13 veri tabanı motorunun bölüntülenmiş veri ile birlikte çalışırken önceden tanımlanmış işlemlere nasıl yapılandırılabilirdiğini gösteren bir diyagramdır,

Şekil 14 veri tabanı motorunun veri setine yeni bir veri eklendiği zaman dizin yapılarının nasıl güncelleyebildiğinin bir örneğini gösteren bir diyagramdır,

Şekil 15 veri tabanı motorunun veri setinden veri kaldırıldığı zaman dizin yapılarının nasıl güncelleyebildiğinin bir örneğini gösteren bir diyagramdır,

Şekil 16 kimlik dizininin bir veri setine veri eklendiği zaman yeni nesneler için nasıl oluşturulduğunu gösteren bir diyagramdır,

Şekil 17 ilişki dizininin bir veri setine veri eklendiği zaman yeni nesneler için nasıl oluşturulduğunu gösteren bir diyagramdır, ve

Şekil 18 ilişki dizininin bir veri setinden veri kaldırıldığı zaman nasıl güncellendiğini gösteren bir diyagramdır.

DETAYLI AÇIKLAMA

Veri modelindeki en önemli birim nesnedir. Her bir nesne nesnenin dahil ettiği bilginin kavramını temsil eden nesnelerin bir türüne aittir. Nesnelerin türü literatür tarafından bilginin nasıl tanımlandığı ve o alandaki uzmanlar tarafından nasıl anlaşıldığına göre oluşturulmaktadır.

Bilginin sıklıkla nasıl davrandığını yansıtmak için, aynı türdeki farklı nesnelerin aynı türde bilgileri içermesi gerekmemektedir. Her bir nesne ayrıca kendisinin ilişkili olduğu diğer nesneleri ve bu ilişkinin ne demek olduğunu bilmektedir, yani ilişkinin varlığı ile temsil edilen bilgi.

Şekil 1 bu veri modeli kullanılarak bir veri setinin nasıl temsil edilebildiğinin bir grafiksel örneğini vermektedir. Daireler farklı nesneleri temsil etmektedir ve oklar farklı nesneler arasındaki yönlendirilmiş ilişkileri temsil etmektedir. Bir daire içerisindeki bir harf nesnenin türünü belirtmektedir ve daire içindeki sayı her bir nesnenin benzersiz tanımlayıcısıdır. Şekil 1 1 benzersiz tanımlayıcı sayısı ile birlikte A nesne türüne ait bir nesneyi, sırasıyla 2, 3 ve 4 sayısı ile tanımlanan B nesne türüne ait üç nesneyi, sırasıyla 5 ve 6 sayısı ile tanımlanan C nesne türüne ait iki nesneyi, ve nihai olarak 7 sayısı ile tanımlanan D nesne türüne ait bir nesneyi göstermektedir.

Şekilde gösterildiği gibi, aynı türdeki nesneler aynı ilişkilere sahip olmak durumunda değildirler. Ayrıca, şekil ile gösterildiği gibi, bu modelde temsil edilen veri yönlendirilmiş ilişkiler le birlikte bir grafik olarak kolaylıkla sağlanabilmektedir, bu insan beyninin bilgiyi nasıl ilişkilendirdiği ve yapılandırdığını temsil etmektedir. Bu türdeki veri modelinde, veri setinin sık sık güncellenmesi kolaylıkla yapılmaktadır, çünkü veri setine eklenen her bir nesne veya nesneler zinciri grafikteki uygun yere alt bir grafik olarak eklenmektedir. Veri setinin yapısal değişiklikleri ayrıca kolaylıkla ele alınmaktadır çünkü nesnenin her türü nesne içerisinde dâhil edilen bilginin herhangi bir katı yapısını dayatmamaktadır.

Buluş, sıklıkla veri setlerine karşı sorulması gerekmeyen özellikle belirli türde soruların tanımlanmasını, bununla birlikte bu türdeki soruların belirlenmesi için yeni bir yöntem ve terminolojiyi amaçlanmaktadır. Bu türdeki sorular aşağıdaki özellikler ile açıklanabilmektedir:

1.Verinin alt kümesinin verilmesi

2.Verinin alt kümesine dâhil edilen veya ilişkili olan belirli bir türdeki nesnelerin tamamının bulunması

5 3.Önceki adımda tanımlanan nesneleri ve bu nesneler ile ilişkili tüm nesneleri içeren tüm veri setinden verinin yeni alt kümesinin oluşturulması

10 Bu türdeki soruları açıklamak için kullanılan terminoloji aşağıdaki kavramlar ile tanımlanmaktadır:

- Bu şekilde bir veri setinin sorgulanması yöntemine yansıtıcı mantık denilmektedir
- Adım 2’de kullanılan nesnelerin türüne yansıtma noktası denilmektedir
- Yukarıdaki adım 1’den verinin alt kümesinin adım 3 sırasında yansıtma noktasından yansıtıldığı söylenmektedir

20 Yansıtıcı mantığın akademiden farmasötik araştırmaya ve sağlık bakımına değişen, çeşitli alanlarda bilgiye erişebilmek için farklı veri setlerinin nasıl sorgulanması ve analiz edilmesi gerektiğinin bir ayrılmaz parçası olarak bulunmuştur. Örneğin klinik veri gibi bir analizde tipik bir yansıtma noktası hasta olacaktır, ancak diğer yansıtma noktaları örneğin deney, de

25 ilgi konusu olabilmektedir.

Klinik verinin analizinin aşağıdaki örneği bir veri setinden bilgi elde etmek için yansıtıcı mantığın nasıl kullanılabildiğini gösterebilmektedir. Bir doz ilaç verilen belirli hastaların neden yüksek kan basıncına sahip olduğunun bir açıklamasının bulunması ile bir klinik araştırmacı ilgilenebilmektedir. Yukarıda tanımlanan terminolojinin kullanımı ile, bu problem şu şekilde belirtilebilmektedir:

- 35 1.Yüksek sayılan ve ilacın bir dozunun verilmiş olduğu hastalara bağlanmış tüm kan basıncı ölçümünün bulunması,
- 2.Hastaları temsil eden nesnelerin türündeki önceki durumu

karşılayan nesnelerin yansıtılması,

3.Yüksek kan basıncı için bir açıklama bulma amacı ile birlikte, ilgili hastalar için tüm bilgileri içeren ortaya çıkan veri setinin analiz edilmesi.

5

Yukarıdaki örnekte, hastaları temsil eden nesneler türü yansıtma noktası olarak kullanılmaktadır ve bu hastalarla ilgili tüm bilgileri elde etmek için ilgili ölçümler yansıtma noktasına yansıtılmaktadır.

10

Şekil 2 yansıtıcı mantığın nasıl bir veri setine dönüştürdüğünün bir örneğini göstermektedir. Bu örnekte verinin bir alt kümesi B etiketlenmiş türdeki nesnelere yansıtılmaktadır. Şeklin üst sağ kısmı kesintisiz bir çizgi dâhilinde verinin seçilmiş alt kümesini göstermektedir. Şeklin bir sol orta kısmı verinin alt kümesi ile ilişkili ve veri setini yansıtırken kullanılacak olan kesikli çizgi ile gösterilen B etiketlenmiş nesneleri göstermektedir. Şeklin alt sağ kısmı B etiketlenmiş türdeki nesnelere yansıtılan alt kümenin kesikli ve noktalı çizgisi ile gösterilen sonucu göstermektedir. Aşağıdakilere göre nesnelerin farklı türlerinin yorumlanması ile, Şekil 2 yukarıdaki örneği açıklamak için kullanılabilir:

15

20

25

30

35

- Etiketlenmiş A türündeki nesneler gerçekleştirilmiş klinik deneyi temsil etmektedir
- Etiketlenmiş B türündeki nesneler hastaları temsil etmektedir
- Etiketlenmiş C ve D türündeki nesneler kan basıncının ölçülmesi için farklı yöntemleri temsil etmektedir
- Etiketlenmiş E türündeki nesneler kan basıncı ölçümlerini temsil etmektedir,
- Kesintisiz çizgi ile işaretlenmiş nesneler, mesela 10 ve 12 ile tanımlanmış, yüksek kan basıncı ölçümlerini temsil etmektedir,
- Kesikli çizgi ile işaretlenmiş nesneler, mesela 2 ve 4 ile tanımlanmış, yüksek kan basıncı ölçümü yapılmış hastaları temsil etmektedir, ve
- Kesikli ve noktalı çizgiler ile işaretlenmiş nesneler yüksek

kan basıncı ölçümü yapılmış hastalar ile ilgili nesneleri temsile etmektedir.

Yansıtıcı mantığın açıklaması ve örnekleri yukarıda açıklanan veri modelini kullanmaktadır, ancak buluş ayrıca diğer veri modelleri üzerinde yansıtıcı mantığın kullanımını içermektedir.

Yukarıda belirtildiği gibi, yeniliğin bir unsuru ilişkilerin dizin oluşturulmasının yeni yöntemidir ve yukarıda tanımlanmış buluşun unsurlarını bir veri tabanı motorunun kullanmasının sağlamak için bu dizinlerin hem sıradan sorular gem de yansıtıcı mantık için kullanılmasıdır.

Kullanılan dizin yapıları bit eşlem esaslıdır, bu her bir bitin ya sıfır veya bir olabildiği ve her bir bitin belirli bir anlamla eşleştirildiği yerde bitlerin vektörlerinden oluştuğu anlamına gelmektedir. İlaveten bit vektörünün uzunluğu, yani vektördeki bitlerin sayısı, daima hedeflenmiş işlemci mimarisinin kelime uzunluğunun katlarıdır. Örnek olarak bir 64 bitlik işlemci mimarisi için, bit vektörünün uzunluğunun daima 64'ün katları olacağı anlamına gelmektedir. Bit vektörünün uzunluğunun bu sırlaması, bit vektörler üzerindeki mantıksal işlemlerin optimize edilmiş bitsel işlemler mesela VE veya VEYA kullanılarak yapılabilmesi ile sonuçlanmaktadır. Ancak, farklı dizin yapılarının kavramının anlaşılmasını kolaylaştırmak için, bit vektörleri aşağıda kullanılan örneklerde kelime uzunluğuna hizalanmamıştır.

Bir çift farklı bir eşlem dizin yapısı veri tabanı motorunun hem sıradan soruları hem de yansıtıcı mantığı kullanması için oluşturulmaktadır. Buluşun en belirgin iki dizin yapısı ilişki dizini ve kimlik dizinidir.

Kimlik dizini her birisinin bir veri setinde bir nesneyi tanımladığı dizin vektörlerinden oluşmaktadır. Her bir dizin vektörü içerisinde bir bitin bire ve diğer tüm bitlerin sıfıra

ayarlandığı bir bit vektörüdür. Bu şekilde bire ayarlanan bit veri setindeki nesneyi benzersiz olarak tanımlayacaktır. Çünkü kimlik dizini bir veri setinde her bir nesneyi tanımlamak için bir benzersiz bit kullanmaktadır, birçok nesne birçok bitin bire ayarlandığı bir tekil bit vektörü ile tanımlanabilmektedir. Bu türde bir bit vektörü bitsel VEYA işlemleri kullanan birçok kimlik dizini vektörü birleştirilerek oluşturulabilmektedir.

İlişki dizini her birisinin belirli bir nesnenin hangi nesnelere bağlı olduğunu tanımlayan dizin vektörlerinden oluşmaktadır. Her bir ilişki dizini vektörü mevcut nesnenin bağlı olduğu nesnelerin tüm kimlik dizini vektörlerinin kombinasyonunu içermektedir. Bir ilişki dizini vektörünün yukarıda tanımlanan şemayı kullanmada mevcut nesne ile ilişkili olan tüm nesneleri benzersiz tanımlayan bir bit vektörü olması anlamına gelmektedir.

Her türdeki nesneler için, o türdeki tüm nesnelerin kimlik dizini vektörlerinin kombinasyonundan oluşan bir dizin vektörü oluşturulmaktadır. Bu dizin yapılarına dayalı olarak, verinin bir alt kümesinde belirli türdeki hangi nesnelerin mevcut olduğunun belirlenmesi mümkün olmaktadır.

Şekil 3 bir veri setinde her bir nesne için bir kimlik ve bir ilişki dizininin bir örneğini göstermektedir. Sıfırlar ve birlerin birinci sırası kimlik dizinidir ve sıfırlar ve birlerin ikinci sırası da ilişki dizinidir. Bu şekilde görülebildiği gibi, her bir nesne kimlik vektöründe bire ayarlanan bir bit ile tam olarak tanımlanmaktadır. Örnek olarak, kimlik dizini vektörü 0001000 3 sayısı ile tanımlanan B türündeki nesneyi tanımlamaktadır. Şekil ayrıca veri setindeki her bir nesnenin benzersiz bir kimlik dizini vektörüne sahip olduğunu göstermektedir. Bu ilişki dizininin mevcut nesnenin ilişkili olduğu tüm nesnelerin kimliklerinin bir vektörü olarak etkili bir şekilde yorumlanmasını sağlamaktadır.

Şekil 3'te gösterilen yapılandırmada, A nesne türünün nesne 1'i birinci bit pozisyonunda bir olarak ve tüm diğer pozisyonlarda sıfır olarak yorumlanan kimlik dizini vektörü 1000000'i vermektedir. Dizin vektörü 7 bit uzunluğuna sahiptir. B türündeki nesnenin bir birinci nesnesi 2 olarak gösterilmektedir ve kimlik dizini vektörü 0100000'i vermektedir ve B türündeki nesnenin bir ikinci nesnesi 3 olarak gösterilmektedir ve kimlik dizini vektörü 0001000'i vermektedir. Kalan nesneler karşılık gelen kimlik dizini vektörlere sahiptirler.

10

Kimlik dizininin bir veri setinde her bir nesneyi tanımlamak için bir benzersiz bit kullanmasından ötürü, birçok nesne birçok bitin bire ayarlandığı yerde bir tekil bit vektörü ile tanımlanabilmektedir. Bu türde bir bit vektörü bitsel VEYA işlemleri kullanan birçok kimlik dizini vektörü birleştirilerek oluşturulabilmektedir. Örneğin, Şekil 3'te bit vektörü 0101100 sayılar 2, 3 ve 4 ile tanımlanmış nesneleri benzersiz olarak tanımlayacaktır.

15

20

Nesne 1'in (1000000) ilişki dizini vektörü nesne 1 ile ilgili tüm nesnelerin kimlik dizinlerinin bir kombinasyonu ile belirlenmektedir. Nesne 1 kendisini içeren tüm diğer nesneler ile bir ilişkiye sahiptir. Sonuç olarak nesne 1'in ilişki dizini vektörünün her bir pozisyonu bire ayarlanacaktır çünkü her bir ilgili nesne kendi kimlik dizini vektöründen bir ile katkıda bulunacaktır.

25

Nesne 4 (0000100) nesne 1 (1000000) ile bir ilişkiye sahiptir, nesne 6 (0000010) ile ilişki dizini vektörünün birinci pozisyonunda bir ile sonuçlanan, nesne 7 (0000001) ile altıncı pozisyonda bir ile sonuçlanan, son pozisyonda bir ile sonuçlanan, ve kendisi (0000100) ile birlikte beşinci pozisyonda bir ile sonuçlanan bir ilişkiye sahiptir. Nesne 4'ün nihai ilişki dizini vektörü bundan dolayı Şekil 3'te gösterildiği gibi 1000111 olacaktır. Tüm diğer ilişki dizini vektörleri aynı biçimde oluşturulmaktadır. İlave bir örnek olarak, 2 sayısı ile tanımlanan nesne 1, 2 ve 5 sayıları ile tanımlanan nesnelerin

30

35

kimlik dizini vektörüne karşılık gelen 1110000 olan bir ilişki dizini vektörüne sahiptir.

İlişki dizinini oluştururken, iki nesnenin ne zaman ilişkilendirildiği ve ne zaman ilişkilendirilmediğinin tanımlamak önemlidir. Örneğin, Şekil 3'te Etiketlenmiş B türündeki tüm nesneler 1 sayısı ile tanımlanmış nesne vasıtasıyla ilişkilendirilmektedir çünkü B türündeki nesneler için ilişki dizini vektörleri bu ilişkiyi içermemektedir. Bunun sebebi şudur, örneklerde kullanılan veri modelinde, bir nesnenin kendi ebeveynleri ve çocukları ile ilişkide olduğu söylenmektedir ve nesnenin kendisi gibi aynı türdeki kardeşler hariç tüm nesnelerin bunlarla ilişkili olduğu söylenmektedir. Şekil 3'teki farklı nesneler eğer aşağıda ortaya konulduğu gibi yorumlanır ise, ilişkilerin bu tanımlaması kolaylıkla anlaşılmaktadır:

- Etiketlenmiş A türündeki nesneler gerçekleştirilmiş klinik deneyi temsil etmektedir
- Etiketlenmiş B türündeki nesneler deneydeki hastaları temsil etmektedir
- Etiketlenmiş C türündeki nesneler bir hastanın boyunu temsil etmektedir
- Etiketlenmiş D türündeki nesneler bir hastanın kilosunu temsil etmektedir

Yukarıdaki yorumlamayı kullanarak, 2, 3 ve 4 sayıları ile tanımlanmış nesneler ile temsil edilen farklı hastaların doğrudan ilişkili olmayacağı ayrıca 2 sayısı ile tanımlanmış hastanın uzunluğunun 4 sayısı ile tanımlanmış hastanın uzunluğu ile ilişkili olmayacağı açıktır. Ancak 4 sayısı ile tanımlanmış olan hastanın boyu ve kilosu ilişkili olacaktır.

Nesne türü kimlik dizini olarak adlandırılan bir üçüncü dizin ayrıca veri tabanı motorunun diğer dizinleri daha verimli kullanması için oluşturulmaktadır. Bu dizin her türdeki nesneler için bir dizin vektöründen oluşmaktadır, burada her

bir dizin vektörü o türdeki tüm nesnelerin kimlik dizininin kombinasyonundan oluşturulmaktadır. Bu şu anlama gelmektedir bir nesne türü kimlik dizini vektörü yukarıda tanımlanan şemayı kullanan belirli bir türdeki tüm nesneleri benzersiz bir şekilde tanımlamaktadır. Bu dizin yapıları verinin bir alt kümesinde belirli bir türde hangi nesnelerin mevcut olduğunu bulabilmek için kullanılmaktadır. Örnek olarak, Şekil 3'te C türündeki nesneler için nesne türü kimlik dizini 0010010 olacaktır, bu C türündeki iki nesnenin kimlik dizini vektörlerinin bir kombinasyonudur.

Önceden belirtildiği gibi, farklı dizin vektörlerini temsil etmek için kullanılan bit vektörleri hedeflenmiş işlemci mimarisinin kelime uzunluğuna sıralanmaktadır. Bu aşağıdaki adımlar ile yapılmaktadır:

1.Eğer nesnelerin sayısı kelime uzunluğunun bir katı ise, o halde başka bir eylem gerekmemektedir

2.Eğer nesnelerin sayısı kelime sayısının bir katı değil ise, o halde tüm nesneleri temsil edebilen kelime uzunluğunun en düşük katına bitlerin sayısını ayarla ve ekstra bitleri sıfıra ayarla

Yukarıdaki sıralama yöntemi kullanılarak, eklenmiş ekstra bitlerin varlığı dizinleri etkilemeyecektir çünkü sıfıra ayarlanmış bir bit hiçbir nesne mevcut değilmiş gibi işlem görecektir.

Veri tabanı motoru aşağıdaki özelliklerin sorgularını cevaplamak için yukarıda açıklanan üç dizin yapısından faydalanmaktadır:

1.Bir veya daha fazla filtreleme kriterinin verilmesi yani veri setinin bir veya daha fazla alt kümesi ile ilişkilendirilen veri setindeki kısıtlamalar

2. Filtreleme kriterlerinin nasıl birleştirileceğinin verilmesi

3. Her bir filtreleme kriterini yerine getiren nesnelerin alt kümesinin bulunması

4. Her bir alt küme ile doğrudan veya dolaylı ilişkili veya dahil edilen tüm nesnelerin bulunması

5 5. Adım 2'den ortaya çıkan bilgiye göre mantıksa ayarlanmış işlemleri kullanarak adım 4'ten alt kümelerin birleştirilmesi

10 Yukarıda açıklanan türdeki sorular veri setinden bilgi elde etmek için veri setlerine karşı sıklıkla sorulmalıdır. Örnek olarak, hastaların boyunun ve kilosunun kaydedilmiş olduğu tıbbi kayıtların bir veri setini göz önünde bulundurun. Herhangi bir eylemin yapılıp yapılmayacağını bulmak için bir doktor düşük bir kilo ölçümü ile ilgili tüm bilgilerin bulunması ile
15 ilgilenebilmektedir. Şekil 4 bu türde bir sorguyu cevaplandırmak için veri tabanı motorunun kimlik dizini ve ilişki dizininden nasıl faydalanacağını bir örneğini göstermektedir. Eğer
20 şekildeki bilgi aşağıdakine göre yorumlanır ise, o halde yukarıdaki örnek şekli ilaveten açıklamak için kullanılabilmektedir:

- Etiketlenmiş A türündeki nesneler bilgileri toplayan hastaneleri temsil etmektedir
- Etiketlenmiş B türündeki nesneler hastaları temsil
25 etmektedir
- Etiketlenmiş C türündeki nesneler bir hastanın boyunu temsil etmektedir
- Etiketlenmiş D türündeki nesneler bir hastanın kilosunu temsil etmektedir
- 30 • 7 sayısı ile tanımlanan etiketlenmiş D türündeki nesne düşük bir kilo ölçümü olarak varsayılmaktadır
- İki filtreleme kriteri VE kullanılarak birleştirilecek olarak tanımlanmaktadır

35 • Bilgi belirli bir hastanede toplanmalıdır

• Ağırlık ölçümü düşük varsayılmalıdır

- Kesintisiz bir çizgi ile işaretlenmiş nesneler birinci filtreleme kriteri ile eşleşmektedir
- Kesikli bir çizgi ile işaretlenmiş nesneler ikinci filtreleme kriteri ile eşleşmektedir
- Kesikli ve noktalı çizgi ile birlikte olan nesneler sorgunun sonucudur

Şekil 4 doğrudan veya dolaylı olarak iki kriteri karşılayan tüm nesneleri bulmak için ilişki dizininin ve kimlik dizininin nasıl kullanılabildiğinin bir örneğini göstermektedir. Şekil 9'un açıklamasına bağlı olarak başka bir açıklama aşağıda ortaya konmaktadır. Şeklin üst kısmında kesintisiz çizgi elipsi ve kesikli çizgi elipsindeki nesneler iki kriteri sembolize etmektedir. Bir birinci nesne (kesintisiz çizgi) ilişki dizini vektörü 1111111'e sahiptir (nesne 1) ve ikinci bir nesne (kesikli çizgi) ilişki dizini vektörü 1000111'e sahiptir (nesne 7). Şeklin orta kısmında, nesnelerin iki setinin ilişki dizini vektörleri 1000111 ortaya çıkan kimlik dizini vektörü üretmek için bir bitsel VE işlemi kullanılarak birleştirilmektedir ve şeklin alt kısmında ortaya çıkan kimlik dizini vektörünü eşleştiren nesneler bir kesikli ve noktalı çizgi ile işaretlenmiştir. Bu nesneler kimlik dizini vektörleri 1000000 (nesne 1), 0000100 (nesne 4), 0000010 (nesne 6) ve 0000001 (nesne 7), sahiptir, karşılaştıran tablo 1.

1000111	Ortaya çıkan kimlik dizini vektörü
1000000	Nesne 1
0000100	Nesne 4
0000010	Nesne 6
0000001	Nesne 7

Tablo 1

Şekil 5 yansıtıcı mantık içeren bir soruya cevap vermek için önceden açıklanmış dizin yapılarından veri tabanı motorunun

nasıl faydalandığının bir örneğini göstermektedir. Eğer şekildeki bilgi aşağıdakilere göre yorumlanır ise, yukarıdaki örnek şekli ilaveten açıklamak için kullanılabilir:

- 5 • Etiketlenmiş A türündeki nesneler gerçekleştirilmiş klinik deneyi temsil etmektedir
- Etiketlenmiş B türündeki nesneler hastaları temsil etmektedir
- Etiketlenmiş C türündeki nesneler kan basıncı ölçümlerini temsil etmektedir
- 10 • 5 ve 6 sayıları ile tanımlanmış olan etiketlenmiş C türündeki nesneler yüksek kan basıncı ölçümleri sayılmaktadır
- Kesintisiz çizgi ile işaretlenmiş nesneler filtreleme kriterlerini eşleştirmektedir, bu durumda yüksek sayılan kan basıncı ölçümleri
- 15 • Kesikli çizgi ile işaretlenmiş nesneler, filtreleme kriterleri ile eşleştirilen nesnelere bağlı sorgunun yansıtıldığı nesne türünün nesnelerini temsil etmektedir
- 20 • Kesikli ve noktalı çizgi ile işaretlenmiş nesneler sorgunun sonucudur

Şeklin üst kenarında filtreleme kriterlerine eşleşen nesneler tanımlanmaktadır. Veri tabanı motoru o halde bitset VEYA işlemleri kullanarak filtreleme kriterlerine eşleşen tüm nesnelerin ilişki dizini vektörlerini birleştirecektir. Bu durumda veri tabanı motoru 1110110 olacak olan yeni bir bit vektörünün içine bir bitset VEYA işlemi kullanarak 1110000 ve 1000110 iki ilişki dizini vektörünü birleştirecektir. Bu bit vektörü daha sonra bir bitset VE işlemi kullanarak yansıtma noktası için nesne türü kimlik dizini vektörü ile birleştirilecektir. Ortaya çıkan bit vektörü sorgunun yansıtıldığı, filtreleme kriterleri ile eşleşen nesnelerle ilişkili nesne türünün tüm nesnelerinin benzersiz kimliğini içerecektir.

Bu durumda nesne türü kimlik dizini vektörü 0101100 olmaktadır ve ortaya çıkan bit vektörü 0100100'ü verecek olan 1110110 ile birlikte, bir bitsel VE işlemi kullanarak, birleştirilecektir. Bu adım şeklin orta sol kısmında gösterilmektedir. Sonraki adım olarak sorgunun yansıtıldığı nesne türünün tüm nesnelerinin ilişki dizini vektörlerini bulmak için veri tabanı motoru kimlik dizini ile son adımda üretilen bit vektörünü kullanacaktır. Veri tabanı motoru daha sonra sorgu ile eşleşen tüm nesnelerin benzersiz kimliğini içeren bir bit vektörü elde etmek için bitsel VEYA işlemleri kullanarak tüm bu ilişki dizini vektörlerini birleştirecektir.

Bu durumda sorgunun yansıtıldığı, son bit vektöründe mevcut olan nesne türünün nesnelerinin ilişki dizini vektörleri 1110000 ve 1000111 olmaktadır. Bitsel bir VEYA işlemi kullanarak bu iki ilişki dizini vektörünün birleştirilmesi 1110111 bit vektörünü vermektedir, bu sorgu ile eşleşen tüm nesnelerin benzersiz kimliğini içermektedir. Bu adım şeklin sol tarafında gösterilmektedir. Son adım son bit vektöründe mevcut olan tüm nesneleri bulmak için kimlik dizinini kullanmaktadır. Son adımın sonucu şeklin alt kenarında gösterilmektedir.

Şeklin orta kısmında ortaya çıkan bit vektörü ile eşleşen etiketlenmiş B türündeki nesneler kesikli çizgiler ile işaretlenmiştir. Bu iki nesne, nesne 2 (0100000) ve nesne 4 (0000100) seçilmiş veri seti yansıtılırken kullanılacaktır. Son bit vektörü 1110111'i üretmek için 1110000 (nesne 2'nin) değerine sahip bir birinci ilişki dizini vektörü 1000111 (nesne 4'ün) değerine sahip bir ikinci ilişki dizini vektörü ile bir bitsel VEYA işleminde birleştirilmektedir. Şeklin alt sağ tarafında son bit vektöründe mevcut olan nesneler kesikli ve noktalı çizgi ile işaretlenmiştir, karşılaştıran tablo 2.

1110111	Ortaya çıkan kimlik dizini vektörü
1000000	Nesne 1
0100000	Nesne 2

0000100	Nesne 4
0010000	Nesne 5
0000010	Nesne 6
0000001	Nesne 7

Tablo 2

Birçok yansıtma noktası ile yansıtıcı mantık kullanan sorular, veri setinden bilginin daha yüksek düzeylerini elde etmek için veri setlerine karşın bazen sorulmalıdır. Örnek olarak, deneylerin bazılarının hastaların kan basıncını ölçtükleri klinik deneylerden bilgi içeren bir veri setini göz önünde bulundurun. Bir klinik araştırmacı yüksek kan basıncı olan hastalara bakmak ile ilgilenebilmektedir eğer bunlar belirli bir ilacın verildiği bir deneyin parçası olmuşlar ise. Bu soru yansıtıcı mantık için önceden tanımlanmış terminoloji kullanılarak, aşağıdaki gibi formüle edilebilmektedir:

1.Söz konusu ilacın verilmesi ile ilişkili tüm nesneleri bul

2.Yansıtma noktası olarak klinik deneyleri temsil eden nesne türünü kullan ve bu yansıtma noktasında adım 1'den nesneleri yansıt

3.Yüksek kan basıncı ölçümleri ile ilişkili tüm nesneleri bul

4.Yansıtma noktası olarak hastaları temsil eden nesnelerin türünü kullan ve bu yansıtma noktasında adım 3'ten nesneleri yansıt

5.Adım 2 ve adım 4'ten ortaya çıkanların her ikisinde mevcut olan tüm nesneleri bul

6.Adım 5'ten bilgileri analiz et

Şekil 6 yukarıda açıklanan gibi bir soruyu cevaplamak için veri tabanı motorunun önceden açıklanmış yapılardan nasıl faydalanabildiğinin bir örneğini göstermektedir. Şekildeki bilgi eğer aşağıdakine göre yorumlanır ise, o halde yukarıda örnek şekli ilaveten açıklamak için kullanılabilir:

- Etiketlenmiş A türündeki nesneler gerçekleştirilmiş klinik

deneyleri temsil etmektedir

- Etiketlenmiş B türündeki nesneler hastaları temsil etmektedir
 - Etiketlenmiş C türündeki nesneler kan basıncı ölçümlerini temsil etmektedir
 - Etiketlenmiş D türündeki nesneler söz konusu ilacın uygulanmasını temsil etmektedir
 - 6 sayısı ile tanımlanmış etiketlenmiş C türündeki nesne yüksek kan basıncı ölçümü var sayılmaktadır
 - Kesintisiz çizgi ile işaretlenmiş nesneler klinik deneylerde mevcut olan nesneler türünde yansıtılan sorgunun parçasıdır
 - Kesikli çizgi ile işaretlenmiş nesneler hastaları temsil eden nesnelerin türünde yansıtılmış sorgunun parçasıdır
 - Kesikli ve noktalı çizgi ile işaretlenmiş nesneler sorgunun sonucudur
 - Şeklin sol üst tarafı iki sorgu kısmı ile eşleşen nesneleri tanımlamaktadır
 - Şeklin sağ üst tarafı yansıtma noktaları olarak kullanılacak olan nesneleri tanımlamaktadır
 - Şeklin alt sağ tarafı yansıtmadan sonra, iki sorgu parçasını temsil eden, iki setin parçası olan nesneleri tanımlamaktadır
 - Şeklin alt sol tarafı sorunun cevabını tanımlamaktadır
- Şeklin üst sol tarafında sorgunun iki parçası ile eşleşen nesneler tanımlanmaktadır. Kesintisiz çizgi ile işaretlenmiş nesneler söz konusu ilacın verilmesini temsil etmektedir ve kesikli çizgi ile işaretlenmiş nesne yüksek kan basıncı ölçümünü temsil etmektedir. Veri tabanı motoru ilk olarak bitsel VEYA işlemleri kullanılarak sorgunun her bir parçası ile eşleşen tüm nesnelerin ilişki dizini vektörlerini birleştirecektir. Bu durumda veri tabanı motoru 1110111 olacak olan yeni bir bit vektörüne bir bitsel VEYA işlemi kullanılarak iki ilişki vektörü 1110000 ve 1000111'i birleştirecektir. Bu bit vektörü klinik deneyleri temsil eden nesnelerin türünde yansıtılacak sorgunun parçası için kullanılmaktadır.

Hastaları temsil eden nesneler türünde yansıtılacak sorgunun parçası için, veri tabanı motoru herhangi bir ilişki düzeni vektörünü birleştirmek durumunda değildir çünkü yalnızca tek
5 bir nesne filtreleme kriteriyle eşleşmektedir ve bu nesne 1000111 ilişki dizini vektörüne sahiptir. Sonraki adımdaki veri tabanı motoru, bitset VE işlemleri kullanılarak, ilişkili yansıtma noktası için nesne türü kimlik dizini vektörleri ile henüz oluşturulan bit vektörlerini birleştirecektir. Bu durumda
10 klinik deneylerini temsil eden nesnelerin türü için nesne türü kimlik dizini vektörü 1000000'dir ve hastaları temsil eden nesnelerin türü için nesne türü kimlik dizini vektörü 0101100'dür.

15 Bundan dolayı bit vektörü 1110111 bit vektörü 1000000 veren 1000000 ile birleştirilecektir ve bit vektörü 1000111 bit vektörü 0000100 veren 0101100 ile birleştirilecektir. Son adımdan bit vektörleri ile ilişkili nesneler şeklim üst sağ tarafında gösterilmektedir. Sonraki adım olarak veri tabanı motoru
20 her bir sorgu parçasının yansıtıldığı nesne türünün tüm nesnelerinin ilişki dizini vektörlerini bulmak için kimlik dizini ile birlikte son adımda üretilen bit vektörlerini kullanacaktır. Veri tabanı motoru daha sonra her bir sorgu parçası ile eşleşen tüm nesnelerin benzersiz kimliğini içeren bit
25 vektörlerini almak için bitset VEYA işlemleri kullanılarak, her bir sorgu parçası için, tüm bu ilişki dizini vektörlerini birleştirecektir.

Bu durumda klinik deneyleri temsil eden nesneler türünün
30 yalnızca bir nesnesi bit vektör ile eşleşmektedir böylece hiçbir kombinasyon gerekmemektedir. Bu nesne bir 1111111 ilişki dizini vektörüne sahiptir. İlaveten hastaları temsil eden nesneler türünün yalnızca bir nesnesi bit vektörü ile eşleşmektedir ve bundan dolayı hiçbir kombinasyon
35 gerekmemektedir. Bu nesne bir 1000111 ilişki dizini vektörüne sahiptir. Bu iki bit vektöründe kimlik dizini vektörlerinin mevcut olduğu nesneler şeklim alt sağ alt tarafında gösterilmektedir.

Son adım olarak veri tabanı motoru iki sorgu parçasının nasıl birleştirileceğine dayalı olarak bitset VE veya VEYA işlemleri kullanılarak son adımdan bit vektörlerini birleştirecektir. Bu
5 durumda sorgu parçaları bir bitset VE işlemi ile birleştirilmelidir çünkü soru iki sorgu parçasının kombinasyonu olarak formüle edilmiştir. Bu iki bit vektörünün 1111111 ve 1000111 bir bitset VE işlemi kullanılarak 1000111 sonucu vererek birleştirileceği anlamına gelmektedir. Bu yeni
10 bit vektörü sorgu ile eşleşen tüm nesnelerin benzersiz kimliğini içermektedir. Sorgunun sonucu şeklin sol alt tarafında gösterilmektedir.

Şekil 7 veri tabanı motorunun kimlik dizini vektörlerini nasıl
15 yaptığıнын şematik bir temsilidir. Blok 10'da veri tabanındaki nesnelerin toplam sayısı hesaplanmaktadır ve blok 12'de bir nesne sayacı 0'a ayarlanmaktadır. Birinci yineleme süreci daha sonra blok 14'te başlatılmıştır. Birden çok işlem adımı, içerisinde kimlik dizini vektörünün bir bit vektörü formunda oluşturulduğu blok
20 16'da başlayan veri tabanında her bir kök nesne için atılmaktadır. Kimlik dizini vektörünün uzunluğu, yani bitlerin sayısı, blok 10'da hesaplandığı gibi veri tabanındaki nesnelerin toplam sayısına eşit olarak ayarlanmaktadır.

25 Kimlik dizini vektörüne blok 18'de bir değer verilmektedir. Vektörün tüm bitleri nesne sayacının mevcut değerine karşılık gelen bir tekil bit hariç sıfıra ayarlanmaktadır. Bu tekil bir bire ayarlanmaktadır. Bundan dolayı, kimlik dizini vektöründe yalnızca bir bit pozisyonu bire ayarlanmaktadır öte yandan tüm
30 diğer bitler sıfıra ayarlanmaktadır. Daha sonra, blok 20'de nesne sayacı bir oranında arttırılmaktadır. Blok 22'de yineleme işleminin mevcut nesnesinin çocukları olup olmadığının sorulduğu bir soru bulunmaktadır. Eğer durum bu ise, bloklar 16 ila 22'nin tüm yineleme adımları daha sonra
35 blok 24'te ortaya konulduğu gibi her bir çocuk nesne için tekrarlanmaktadır. Veri tabanının tüm nesnelerinin kimlik dizini vektörüne kök nesneler ile aynı hizada bir değer

verilmektedir. Eğer hiçbir çocuk mevcut değil ise, yineleme işlemi devam etmektedir. Bunu takiben Şekil 1’de temsil edilen veri seti için şema aşağıdaki adımları verecektir:

- 5 • Veri dizisindeki nesnelerin toplam sayısını hesapla, bu durumda 7 nesne
- Bir nesne sayacı oluştur ve değerini 0’a ayarla
- 10 • Birinci kök nesne ile başla, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- Birinci kök nesne için bir kimlik dizin vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- 15 • Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7
- Nesne sayacı olarak aynı sayının biri hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 0’dır
- Ortaya çıkan bit vektörü 1000000’dır
- 20 • Nesne sayacını bir arttır, bu durumda yeni sayı 1’dir
- Birinci kök nesne çocukları içerdiğinden ötürü, herbir çocuk için işlemi devam ettir
- 25 • Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 2 sayısı ile tanımlanmaktadır
- 30 • Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7
- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 1
- Ortaya çıkan bit vektörü 0100000’dır
- 35 • Nesne sayacını bir arttır, bu durumda yeni değer 2’dir

- Mevcut nesne çocukları içerdiğinden ötürü, herbir çocuk için işlemi devam ettir
- 5 • Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 5 sayısı ile tanımlanmaktadır
 - »Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7
 - »Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 2
 - 10 »Ortaya çıkan bit vektörü 0010000'dir
- Nesne sayacını bir arttır, bu durumda yeni değer 3'tür
- 15 • Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 2 sayısı ile tanımlanmaktadır
- Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- 20 • İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 3 sayısı ile tanımlanmaktadır
- 25
 - »Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7
 - »Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 3
 - 30 »Ortaya çıkan bit vektörü 0001000'dir
- Nesne sayacını bir arttır, bu durumda yeni değer 4'tür
- Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- 35

- Üçüncü çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 4 sayısı ile tanımlanmaktadır

5 »Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7

 »Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 4

 »Ortaya çıkan bit vektörü 0000100'dür

10

- Nesne sayacını bir arttır, bu durumda yeni değer 5'tir
- Mevcut nesne çocukları içerdiğinden ötürü, her bir çocuk için işlemi devam ettir

15 • Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 6 sayısı ile tanımlanmaktadır

 »Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7

20 »Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 5

 »Ortaya çıkan bit vektörü 0000010'dur

- Nesne sayacını bir arttır, bu durumda yeni değer 6'dır

25

- Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 4 sayısı ile tanımlanmaktadır

30 • İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 7 sayısı ile tanımlanmaktadır

 »Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7

35 »Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 6

• Ortaya çıkan bit vektörü 0000001'dir

- Nesne sayacını bir arttır, bu durumda yeni değer 7'dir

5 • Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 4 sayısı ile tanımlanmaktadır

10 • Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır

- Birinci kök nesne daha fazla çocuk içermediğinden ötürü, sonraki kök nesneye devam et

15

- Daha fazla kök nesne olmadığından ötürü, işlem tamamlanmıştır.

20 Şekil 8 veri tabanı motorunun ilişki dizini vektörlerini nasıl yaptığını göstermektedir. İlişki dizini kök nesne(ler) ile başlatarak veri tabanının tüm nesneleri için oluşturulmaktadır. Blok 26'da ikinci bir yineleme işlemi başlatılmaktadır. İçerisinde bir ilişki dizini vektörünün oluşturulduğu blok 28'den başlatarak veri tabanındaki her bir kök nesne için 25 birden çok işlem adımı atılmaktadır. İlişki dizini vektörü yineleme işlemindeki mevcut nesnenin kimlik dizini vektörüne eşit bir bit vektörüdür.

30 Oluşturulan ilişki dizini vektörü yineleme işlemindeki mevcut nesnenin ebeveynlerinin ve tüm çocuklarının kimlik dizini vektörleri ile birlikte blok 30'da tamamlanmaktadır. Kimlik dizini vektörleri eklendiği zaman bitsel VEYA işlemleri kullanılabilir. Blok 32'de gösterildiği gibi bir nihai adımda ayrıca nesnelerin farklı türlerinde tüm kardeşlerin 35 kimlik dizini vektörleri.

Blok 34'te yineleme işleminin mevcut nesnesinin çocukları olup

olmadığının sorulduğu bir soru bulunmaktadır. Eğer durum bu ise, bloklar 28 ila 34'ün tüm yineleme adımları daha sonra blok 36'da ortaya konulduğu gibi her bir çocuk nesne için tekrarlanmaktadır. Bu adımlardan sonra veri tabanındaki tüm nesnelere benzersiz bir kimlik dizini vektörü değeri ve bir ilişki dizini vektörü değeri atanmıştır. İlişki dizini vektörleri daha sonra veri tabanını işlemek için kullanılabilmektedir. Bunu takiben Şekil 1'de temsil edilen veri seti için şema aşağıdaki adımları verecektir ve Şekil 3'te gösterildiği gibi aynı ilişki dizini vektörlerini oluşturmaktadır:

- Birinci kök nesne ile başla, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- Birinci kök nesne için bir ilişki dizini vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır

•İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 1000000

•Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitisel VEYA işlemleri kullan, bu durumda 0100000, 0010000, 0001000, 0000100, 0000010 ve 0000001

•Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitisel VEYA operasyonları kullan, bu durumda hiçbirisi

•Ortaya çıkan bit vektörü 1111111

- Birinci kök nesne çocukları içerdiğinden ötürü, her bir çocuk için işlemi devam ettir
- Birinci çocuk nesnesi için bir ilişki dizini vektörü oluştur, bu durumda nesne 2 sayısı ile tanımlanmaktadır

•İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 0100000

- Tüm ebeveynler ve çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu durumda 1000000 ve 0010000
- 5 •Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda hiçbirisi
- Ortaya çıkan bit vektörü 1110000
- 10 • Mevcut nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir
- Birinci çocuk nesnesi için bir ilişki dizini vektörü oluştur, bu durumda nesne 5 sayısı ile tanımlanmaktadır
- 15 •İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 0010000
- Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu durumda 1000000 ve 0100000
- 20 •Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda hiçbirisi
- 25 •Ortaya çıkan bit vektörü 1110000
- Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 2 sayısı ile tanımlanmaktadır
- 30 • Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- 35 • İkinci çocuk nesnesi için bir ilişki dizini vektörü oluştur, bu durumda nesne 3 sayısı ile tanımlanmaktadır

• İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 0001000

• Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu

5 durumda 1000000

• Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda hiçbirisi

10 • Ortaya çıkan bit vektörü 1001000

• Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır

15

• Üçüncü çocuk nesnesi için bir ilişki dizini vektörü oluştur, bu durumda nesne 4 sayısı ile tanımlanmaktadır

• İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 0000100

20 • Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu durumda 1000000, 0000010 ve 0000001

• Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda hiçbirisi

25 • Ortaya çıkan bit vektörü 1000111

• Mevcut nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir

30

• Birinci çocuk nesnesi için bir ilişki dizini vektörü oluştur, bu durumda nesne 6 sayısı ile tanımlanmaktadır

35

• İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 0000010

- Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu durumda 1000000 ve 0000100
- 5 •Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda 0000001
- Ortaya çıkan bit vektörü 1000111
- 10 • Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 4 sayısı ile tanımlanmaktadır
- 15 • Birinci çocuk nesnesi için bir ilişki dizini vektörü oluştur, bu durumda nesne 7 sayısı ile tanımlanmaktadır
- İlişki dizin vektörünü nesnenin dizin vektörüne eşit ayarla, bu durumda 0000001
- 20 •Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu durumda 1000000 ve 0000100
- Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda
- 25 0000010
- Ortaya çıkan bit vektörü 1000111
- Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 4 sayısı ile tanımlanmaktadır
- 30 • Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır
- 35 • Birinci kök nesne daha fazla çocuk içermediğinden ötürü, sonraki kök nesneye devam et

- Daha fazla kök nesne olmadığından ötürü, işlem tamamlanmıştır.

5 Şekil 9 veri tabanı motorunun yukarıda açıklanan gibi bir sorguyu cevaplandırmak için farklı dizin yapılarından nasıl faydalandığını göstermektedir. Yukarıdaki örnek için bu şemayı takiben ve Şekil 4'te temsil edilen veri seti aşağıdaki adımları verecektir:

10

- Kimlik dizini vektörleri ile aynı uzunluktaki bir bit vektörü olarak bir sonuç vektörü oluştur
- Birinci filtreleme kriteri için kriteri karşılayan tüm nesneleri bul, bu durumda nesne 1 sayısı ile tanımlanmaktadır

15

«Bitsel VEYA işlemleri kullanarak yukarıda bulunan tüm nesnelerin ilişki dizini vektörlerini birleştir, bu durumda ilişki dizini vektörü 1111111'dir ve hiçbir kombinasyon gerekmemektedir

20

«Kriterlerin nasıl birleştirileceğine dayalı olarak bitsel VE veya VEYA kullanarak sonuç vektörüne önceki adımdan birleştirilmiş ilişki dizini vektörünü ekle

▪ Mevcut ortaya çıkan vektör 0000000

25

▪ Birleştirilmiş ilişki dizini vektörü 1111111

▪ Kriter VEYA kullanılarak eklenmelidir, çünkü bu birinci kriterdir

▪ Sonuç 1111111

30

- İkinci filtreleme kriteri için kriteri karşılayan tüm nesneleri bul, bu durumda nesne 7 sayısı ile tanımlanmaktadır

«Bitsel VEYA işlemleri kullanarak yukarıda bulunan tüm nesnelerin ilişki dizini vektörlerini birleştir, bu durumda ilişki dizini vektörü 1000111'dir ve hiçbir

35

kombinasyon gerekmemektedir

• Kriterlerin nasıl birleştirileceğine dayalı olarak bitsel VE veya VEYA kullanarak sonuç vektörüne önceki adımdan birleştirilmiş ilişki dizini vektörünü ekle

5

- Mevcut ortaya çıkan vektör 1111111
- Birleştirilmiş ilişki dizini vektörü 1000111
- Kriter VE kullanılarak eklenmelidir, çünkü birinci ve ikinci kriterlerin VE kullanılarak birleştiriliyor olduğu tanımlanmaktadır
- Sonuç 1000111

10

- İlave hiçbir filtreleme kriteri mevcut olmadığından ötürü, sonuç vektöründe kimlik dizini vektörü mevcut olan tüm nesneleri döndür, bu durumda Şekil 4'ün alt kısmında gösterildiği gibi 1, 4, 6 ve 7 sayıları ile tanımlanmış nesneler

15

Blok 38'de bir sonuç vektörü oluşturulmaktadır. Sonuç vektörü bir sıfır değeri ile başlamaktadır. Sonuç vektörünün uzunluğu kimlik dizinin vektörlerinin uzunluğu ile aynıdır. Her bir filtreleme kriteri için bir yineleme işlemi daha sonra blok 40'ta başlamaktadır. Bu yineleme işlemindeki bir birinci adım blok 42'de gösterildiği gibi kriterleri karşılayan tüm nesneleri bulmaktır. Blok 44'te bulunmuş nesnelerin tüm ilişki dizini vektörleri mantıksal VEYA işlemleri kullanılarak birleştirilmiş bir ilişki dizini vektöründe birleştirilmektedir. Birleştirilmiş ilişki dizini vektörü sorgu veya filtreleme kriterlerime dayalı olarak uygun mantıksal VE veya VEYA işlemleri kullanılarak oluşturulan sonuç vektörüne eklenmiştir. Blok 48'de gösterildiği gibi tüm kriterler sonuç vektörüne eklendiği zaman yineleme işlemi tamamlanmaktadır. Nihai olarak, blok 50'de sonuç vektöründe mevcut olan kimlik dizini vektörleri ile birlikte tüm nesneler döndürülmektedir.

25

30

35

Yeniliğin en ayırt edilen kısmı veri tabanı motorunun verimli bir şekilde isteğe bağlı bir yansıtma noktası ile birlikte yansıtıcı mantığı içeren soruları cevaplamak için yukarıda

açıklanan dizin yapılarından nasıl faydalandığıdır. Bu türdeki sorular yukarıda tanımlanmış yansıtıcı mantığın terminolojisi kullanılarak aşağıdaki özellikle ile açıklanabilmektedir:

- 5 1.Bir veya daha fazla filtreleme kriteri verilerek, örneğin veri setinin bir veya daha fazla alt küme ile ilişkili olan veri setinde kısıtlamalar
- 2.Filtreleme kriterinin nasıl birleştirileceğinin belirlenmesi
- 3.Sorgunun yansıtılacağı nesneler türünün belirlenmesi
- 10 4.Her bir filtreleme kriterini karşılayan nesnelerin alt kümesini bul
- 5.Adım 4'ten her bir alt kümeye ya dahil edilmiş veya da doğrudan veya dolaylı ilişkilendirilmiş tüm nesneleri bul
- 6.Adım 2'den bilgilere göre mantıksa set işlemleri
- 15 7.Adım 5'ten alt kümeleri birleştir
- 8.Adım 6'dan alt kümede mevcut olan adım 3'te belirlenmiş nesne türünün tüm nesnelerini bul
- 9.Adım 7'den her bir nesne için ya doğrudan veya dolaylı olarak ilişkili olduğu tüm nesneleri bul
- 20 10.Adım 9'deki nesneler dahil, yeni bir alt küme adım 8'den tüm nesneleri birleştir

Veri setinden daha yüksek seviyede bilgi elde etmek için, yukarıda açıklanan türdeki sorular sıklıkla veri setlerine karşın sorulmalıdır. Örnek olarak, hastaların kan basıncını ölçen bir klinik deneyden bilgiler içeren bir veri setini ele alın. Bir klinik araştırmacı belirli hastaların neden yüksek kan basıncına sahip olduğunun bir açıklamasını bulmak ile ilgilenebilir. Bu soru yansıtıcı mantık için önceden tanımlanmış olan terminoloji kullanılarak formüle edilebilmektedir, aşağıdaki gibi:

- 1.Yüksek kan basıncı ölçümleri ile ilişkili tüm nesneleri bul
- 2.Yansıtma noktası olarak hastaları temsil eden nesne türünü
- 35 3.Yüksek kan basıncı için bir açıklama bulma amacı ile

birlikte adım 2'den bilgileri analiz et

Şekil 10 (Şekil 11 i underlag)

5 Şekil 10 yukarıda açıklanan gibi yansıtıcı bir mantık sorgusunu cevaplamak için how veri tabanı motorunun farklı dizin yapılarından nasıl faydalandığını göstermektedir. Blok 52'de ilk olarak bir boş sonuç vektörü oluşturulmaktadır ve ilave bir yineleme işlemi herbir filtreleme kriteri için blok 54'te başlatılmaktadır. Sonuç vektörü sıfır değeri ile 10 başlamaktadır. Sonuç vektörünün uzunluğu kimlik dizini vektörlerinin uzunluğu ile aynıdır. Kriterleri karşılayan tüm nesneler blok 56'da belirlenmektedir. Daha sonra, blok 58'de tüm ilişki dizini vektörleri birleştirilmiş bir ilişki dizini vektörü üretmek için mantıksal VEYA işlemleri kullanılarak 15 birleştirilmektedir. Blok 60'dai birleştirilmiş ilişki dizini vektörü sorgu veya filtreleme kriterlerinin birleştirilmesine dayalı olarak sonuç vektörüne eklenmiştir. Blok 62'de gösterildiği gibi tüm kriterler sonuç vektörüne eklendiği zaman yineleme işlemi tamamlanmaktadır. Blok 62'de gösterildiği gibi sonuç vektörüne tüm 20 kriterler eklendiği zaman yineleme işlemi tamamlanmaktadır. Daha sonra sonuç vektöründe bulunan sorgunun blok 64'e yansıtacağı nesne türünün tüm nesnelerini bul.

25 Yansıtma daha sonra kimlik dizini vektörleri ile aynı uzunluğa sahip yansıtma sonuç vektörünü oluşturan adım ile blok 66'da ilerlemektedir ve her bir nesne için bir ilerleme adımında, blok 68'de blok 70'deki bitsel bir VEYA işlemi kullanılarak yansıtma sonucu vektörüne ilişki dizini vektörünü ekleme adımı ile devam edilmektedir. Blok 72'de gösterildiği gibi yansıtma 30 sonucu vektörüne tüm dizin vektörleri eklendiği zaman, yineleme işlemi tamamlanmaktadır ve nihai olarak, blokta işlem yansıtma sonuç vektöründe mevcut olan kimlik dizini vektörleri ile tüm nesneleri geri döndüren adım ile birlikte blok 74'te tamamlanmaktadır.

35

Yukarıdaki örnek için bu şemayı takiben ve Şekil 5'te temsil edilen veri seti aşağıdaki adımları verecektir:

- Kimlik dizini vektörleri ile aynı uzunluktaki bir bit vektörü olarak bir sonuç vektörü oluştur
 - Birinci filtreleme kriteri için kriteri karşılayan tüm nesneleri bul, bu durumda 5 ve 6 sayıları ile tanımlanmış nesneler
- »Bitsel VEYA işlemleri kullanarak yukarıda bulunan tüm nesnelerin ilişki dizini vektörlerini birleştir, bu durumda ilişki dizini vektörleri 1110000 ve 1000110'dir ve birleştirilmiş sonuç 1110110'dur
- »Kriterlerin nasıl birleştirileceğine dayalı olarak bitsel VE veya VEYA kullanarak sonuç vektörüne önceki adımdan birleştirilmiş ilişki dizini vektörünü ekle
- Mevcut ortaya çıkan vektör 0000000
 - Birleştirilmiş ilişki dizini vektörü 1110110
 - Kriter VEYA kullanılarak eklenmelidir, çünkü bu birinci kriterdir
 - Sonuç 1110110
- İlave hiçbir filtreleme kriteri mevcut olmadığından ötürü sonraki adıma devam et
- Sorgunun yansıtıldığı, sonuç vektöründen mevcut olan nesneler türünün tüm nesnelerini bul
- »Sonuç vektörü 1110110'dur
- »Sorgunun yansıtıldığı nesnelerin türünün nesne türü kimlik dizini vektörü 0101100'dür
- »Bir bitsel VE işlemi kullanılarak yukarıdaki iki bit vektörünün birleşimi 0100100'dür
- »Yukarıdaki bit vektörü 2 ve 4 ile tanımlanmış nesnelerin benzersiz kimliğini içermektedir.
- Kimlik dizini vektörleri ile aynı uzunluktaki bir bit vektörü olarak bir yansıtma vektörünün oluşturulması

- Yukarıda tanımlanmış birinci nesne için bir bitsel VEYA işlemi kullanılarak yansıtma sonuç vektörüne ilişki dizini vektörünü ekle

5 ¢Tanımlanmış birinci nesne 2 sayısı ile tanımlanmış nesnedir
 ¢O nesnenin ilişki dizini vektörleri 1110000'dir
 ¢Mevcut yansıtma sonucu vektörü 0000000'dır
 ¢Sonuç 1110000'dir

10

- Yukarıda tanımlanmış ikinci nesne için, bir bitsel VEYA işlemi kullanılarak yansıtma sonucu vektörüne ilişki dizini vektörünü ekle

15 ¢Tanımlanmış ikinci nesne 4 sayısı ile tanımlanmış nesnedir
 ¢O nesnenin ilişki dizini vektörleri 1000111'dir
 ¢Mevcut yansıtma sonucu vektörü 1110000'dır
 ¢Sonuç 1110111'dir

20

- Yukarıda hiçbir ilave nesne tanımlanmamış olduğundan ötürü, yansıtma sonuç vektöründe kimlik dizini vektörünün mevcut olduğu tüm nesneleri geri döndür, bu durumda 1, 2, 4, 5, 6 ve 7 sayıları ile tanımlanmış nesneler

25

Yukarıda açıklanmış tüm dizin yapıları en temel formundadır, ancak buluş ayrıca sıkıştırma kullanımı ve daha karmaşık kodlama gibi optimizasyonları içermektedir. Dizin yapılarının hem boşluğu hem de bilgi sayımsal verimliliği bir seçirdim

30 kodlaması (RLE) esaslı sıkıştırma algoritması kullanılarak arttırılabilmektedir. Bir RLE esaslı sıkıştırma algoritması bir dizin vektöründe sıfırlar veya birlerin ardışık geçişlerini sıkıştırmaktadır. Bu nasıl oluşturulduklarına dayalı olarak yenilik ile açıklanan dizin vektörleri üzerinde
35 özellikle çok iyi çalışmaktadır. Tüm kimlik dizini vektörleri bire ayarlanmış bir bite sahiptir ve tüm diğer bitler sıfıra ayarlanmaktadır. Bir RLE esaslı sıkıştırma algoritması

sıfırların ardışık geçişlerini sıkıştırdığından ötürü, kimlik dizini vektörleri önemli ölçüde sıkıştırılmaktadır.

İlişki dizini vektörleri tüm nesnelerin kimliklerine karşılık gelen bitlere sahiptirler şöyle ki mevcut nesne bire ve diğer tüm nesneler sıfıra ayarlanmaktadır. Kimlik dizini vektörleri ilk olarak bir ağaç derinliği olarak veri modeli üzerinden geçilerek oluşturulduğundan ötürü, tüm ilişkili nesnelere karşılık gelen bitlerin birlerin bir veya daha fazla ardışık geçişi olacağı da yüksek ihtimal dahilindedir. Bir RLE esaslı sıkıştırma algoritması sıfırların ve birlerin ardışık geçişlerini sıkıştırdığından ötürü, ilişki dizini vektörleri önemli ölçüde sıkıştırılmaktadır.

Buluş verinin, yukarıda açıklanmış olan yöntem kullanılarak tüm veri dizin oluşturulduğu zamanki aynı işlemler kullanılarak dizinlenen ve ayrı ayrı kullanılabilen farklı bölüntülere bölünmesini sağlamaktadır. Bu her bir bölüntünün bağımsız bir şekilde güncellenmesini, dizinlenmesini ve işlenmesini sağlamaktadır. Her bir bölüntü bağımsız bir şekilde işlenebildiğinden ötürü, bu ayrıca her bir bölüntünün paralel olarak işlenmesini sağlamaktadır.

Veri modelinin kök nesnesi için kullanılan nesne türünün her bir nesnesi için bir bölüntü kullanılarak en temel bölüntüleme yapılmaktadır. Şekil 11 birçok kök nesnesi ile birlikte bir veri setinin nasıl temsil edildiğinin bir grafiksel örneğini vermektedir. Bu şekilde iki farklı kök nesnesi, 1 ve 8 sayıları ile tanımlanmış nesneler, farklı ağaç yapıları olarak gösterilmektedir. Eğer bu veri bu şema kullanılarak bölüntülenmiş ise, iki farklı bölüntü oluşturulacaktır. Birinci bölüntü 1 sayısı ile tanımlanmış bir kök nesne ile birlikte, şeklin sol tarafında ağacı içerecektir, ikinci bölüntü 8 sayısı ile tanımlanmış bir kök nesne ile birlikte, şeklin sağ tarafında ağacı içerecektir.

Herbir bölüntünün bağımsız bir şekilde dizinlerinin

oluşturulması ve işlenmesini sağlamak için, farklı bölüntülerin dizin vektörleri tamamen ayrılmaktadır.

5 Şekil 12, kesintisiz çizgide bir kutu ile işaretlenmiş olan, iki farklı bölüntü ile birlikte küçük bir veri seti için bir kimlik dizininin bir örneğini göstermektedir. Her bir bölüntü kutunun sol üst tarafında bir sayı ile tanımlanmaktadır ve kimlik dizini vektörü her bir nesneye bitişik basılmaktadır. Örneğin, 5 sayısı ile tanımlanmış olan nesne 1 sayısı ile 10 tanımlanmış olan bölüntüye sahiptir ve 0001000 kimlik dizini vektörüne sahiptir. 11 sayısı ile tanımlanmış nesne 2 sayısı ile tanımlanmış bölüntüye aittir ve 001000 bir kimlik dizini vektörüne sahiptir.

15 Şekil 13 veri farklı bölüntülere bölündüğü zaman veri tabanı motorunun nasıl önceden açıklanmış işlemlerden faydalandığını göstermektedir. Bir işlemin genişletilmesine ilişkin bu genel yöntem hem dizinlerin oluşturulması hem de sorguların cevaplanması için kullanılmaktadır. Bu şemayı takiben alt işlem 20 olarak Şekil 7'den kimlik dizini oluşturma işleminin kullanılması ve Şekil 12'de temsil edilen veri setinin kullanılması, bakınız yukarıda, aşağıdaki adımları verecektir:

25 • Kimlik dizini oluşturma işlemi bir sonuç oluşturmadığından ötürü, hiçbir sonuç kapsayıcı oluşturulmamalıdır

• Birinci bölüntü için, bu durumda bölüntü 1 sayısı ile tanımlanmaktadır, kimlik dizini oluşturma işlemini yürüt

30 »Bölüntüdeki nesnelerin toplam sayısını hesapla, bu durumda 7 nesne

»Bir nesne sayacı oluştur ve değerini 0'a ayarla

»Birinci kök nesne ile başla, bu durumda nesne 1 sayısı ile tanımlanmaktadır

35 »Birinci kök nesne için bir kimlik dizin vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır

- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır
- 5
- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 0
 - Ortaya çıkan bit vektörü 1000000'dir
- »Nesne sayacını bir arttır, bu durumda yeni sayı 1'dir
- 10
- »Birinci kök nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir
- »Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 2 sayısı ile tanımlanmaktadır
- 15
- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır
 - Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 1
- 20
- Ortaya çıkan bit vektörü 0100000
- »Nesne sayacını bir arttır, bu durumda yeni değer 2'dir
- »Mevcut nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir
- 25
- »Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 4 sayısı ile tanımlanmaktadır
- 30
- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır
 - Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 2
- 35
- Ortaya çıkan bit vektörü 0010000

»Nesne sayacını bir arttır, bu durumda yeni değer 3'tür

»Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 2 sayısı ile tanımlanmaktadır

5 »İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 5 sayısı ile tanımlanmaktadır

10

- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır
- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 3
- Ortaya çıkan bit vektörü 0001000

15

»Nesne sayacını bir arttır, bu durumda yeni değer 4'tür

»Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 2 sayısı ile tanımlanmaktadır

20 »Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır

»İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 3 sayısı ile tanımlanmaktadır

25

- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır

30

- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 4
- Ortaya çıkan bit vektörü 0000100

»Nesne sayacını bir arttır, bu durumda yeni değer 5'tir

»Mevcut nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir

35 »Birinci çocuk için bir kimlik dizin vektörü oluştur, bu

durumda nesne 6 sayısı ile tanımlanmaktadır

- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır
- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 5
- Ortaya çıkan bit vektörü 0000010

10

«Nesne sayacını bir arttır, bu durumda yeni değer 6'dır

«Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 3 sayısı ile tanımlanmaktadır

15

«İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 7 sayısı ile tanımlanmaktadır

20

- Bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır, bu durumda 7bit vektörü uzunluk olarak nesnelerin toplam sayısına eşit olmalıdır
- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 6
- Ortaya çıkan bit vektörü 0000001

25

«Nesne sayacını bir arttır, bu durumda yeni değer 7'dir

«Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 3 sayısı ile tanımlanmaktadır

30

«Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır

«Birinci kök nesne daha fazla çocuk içermediğinden ötürü, sonraki kök nesneye devam et

35

«Daha fazla kök nesne mevcut olmadığından ötürü, alt işlem tamamlanmıştır

- Alt işlem bir sonuç vermediğinden ötürü sonuç kapsayıcıya hiçbir şey eklememektedir

- 5 • İkinci bölüntü için, bu durumda bölüntü 2 sayısı ile tanımlanmaktadır, kimlik dizini oluşturma işlemini yürüt

»Bölüntüdeki nesnelerin toplam sayısını hesapla, bu durumda 6 nesne

- 10 »Bir nesne sayacı oluştur ve değerini 0'a ayarla

»Birinci kök nesne ile başla, bu durumda nesne 8 sayısı ile tanımlanmaktadır

»Birinci kök nesne için bir kimlik dizin vektörü oluştur, bu durumda nesne 8 sayısı ile tanımlanmaktadır

15

- Bit vektörü nesnelerin toplam sayısının uzunluğuna eşit olmalıdır, bu durumda 6

- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 0

20

- Ortaya çıkan bit vektörü 100000

»Nesne sayacını bir arttır, bu durumda yeni sayı 1'dir

»Birinci kök nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir

25

»Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 9 sayısı ile tanımlanmaktadır

- Bit vektörü nesnelerin toplam sayısının uzunluğuna eşit olmalıdır, bu durumda 6

30

- Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 1

- Ortaya çıkan bit vektörü 010000

»Nesne sayacını bir arttır, bu durumda yeni değer 2'dir

35

»Mevcut nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir

- »Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır1
- 5
- Bit vektörü nesnelerin toplam sayısının uzunluğuna eşit olmalıdır, bu durumda 6
 - Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 2
 - Ortaya çıkan bit vektörü 001000
- 10
- »Nesne sayacını bir arttır, bu durumda yeni değer 3'tür
- »Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 9 sayısı ile tanımlanmaktadır
- 15
- »Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 8 sayısı ile tanımlanmaktadır
- »İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır0
- 20
- Bit vektörü nesnelerin toplam sayısının uzunluğuna eşit olmalıdır, bu durumda 6
 - Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 3
 - Ortaya çıkan bit vektörü 000100
- 25
- »Nesne sayacını bir arttır, bu durumda yeni değer 4'tür
- »Mevcut nesne çocukları içerdiğinden ötürü her bir çocuk için işlemi devam ettir
- 30
- »Birinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır2
- Bit vektörü nesnelerin toplam sayısının uzunluğuna eşit olmalıdır, bu durumda 6
 - Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 4
 - Ortaya çıkan bit vektörü 000010

- 5 Nesne sayacını bir arttır, bu durumda yeni değer 5'tir
- 5 Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır 0
- 5 İkinci çocuk için bir kimlik dizin vektörü oluştur, bu durumda nesne 1 sayısı ile tanımlanmaktadır 3
- 10
- Bit vektörü nesnelerin toplam sayısının uzunluğuna eşit olmalıdır, bu durumda 6
 - Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 5
 - Ortaya çıkan bit vektörü 000001
- 15 Nesne sayacını bir arttır, bu durumda yeni değer 6'dır
- 15 Mevcut nesne herhangi bir çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 1 sayısı ile tanımlanmaktadır 0
- 20 Mevcut nesne daha fazla çocuk içermediğinden ötürü, işlemi ebeveyn ile devam ettir, bu durumda nesne 8 sayısı ile tanımlanmaktadır
- 20 Birinci kök nesne daha fazla çocuk içermediğinden ötürü, sonraki kök nesneye devam et
- 25 Daha fazla kök nesne mevcut olmadığından ötürü, alt işlem tamamlanmıştır
- Alt işlem bir sonuç vermediğinden ötürü sonuç kapsayıcıya hiçbir şey eklememektedir
- 30 • Hiçbir bölüntü olmadığından ötürü işlem tamamlanmıştır

35 Şekil 13'te gösterilen işlemler aşağıdaki adımları içermektedir: bir sonuç kapsayıcı oluştur, blok 66, her bir bölüntü için yap, blok 68, alt işlemi yürüt, blok 70, eğer alt işlem bir sonuç getirir ise o halde sonucu sonuç kapsayıcıya ekle, blok 73, alt işlemlerin tümü yürütüldüğü zaman hepsini

bitir, blok 74, ve nihai olarak bir sonuç kapsayıcı getir, blok 76.

5 Bir veri seti üzerinde yapılabilen güncellemelerin türü dört farklı kategoriye bölünebilmektedir; yeni nesnelerin eklenmesi, mevcut nesnelerin çıkarılması, mevcut nesnelerdeki bilgilerin güncellemeleri ve nesneler arasındaki ilişkilerde değişiklikler. Mevcut nesnelerdeki bilgilerde güncellemeler ilişkileri veya nesnenin kimliğini modifiye etmediğinden 10 ötürü, dizin yapılarında hiçbir güncelleme gerekli olmamaktadır. Veri seti güncellemelerinin diğer kategorileri için, dizin yapıları modifiye edilmiş veri seti için geçerli olması için modifiye edilmelidir.

15 Şekil 14 veri setine yeni bir nesne eklendiği zaman veri tabanı motorunun nasıl dizin yapılarının güncelleyebildiğini göstermektedir. Şekildeki bilgi aşağıdakilere göre yorumlanmalıdır:

- 20
- Şekil üst sol tarafı orijinal veri setini göstermektedir
 - Şeklin sağ üst tarafı kesintisiz bir çizgi ile işaretlenmiş yeni nesne ile birlikte veri setini göstermektedir
 - Şeklin sağ alt tarafı güncellenmiş dizin yapıları ile birlikte veri setini göstermektedir
- 25
- Modifiye edilmiş tüm dizin vektörleri italiktir, ve diğer tüm vektörler modifiye edilmemiştir.

Bu örnekte 8 sayısı ile tanımlanmış bir nesne 3 sayısı ile tanımlanmış nesnenin çocuğu olarak veri setine eklenmektedir. 30 Dizini yapılarını geçerli tutmak için veri tabanı motoru kimlik dizini, ilişki dizini ve nesne türü kimlik dizinini güncellemelidir. Veri tabanı motoru 8 sayısı ile tanımlanan nesne için yeni bir kimlik dizini oluşturmakla başlamaktadır. Bu durumda yeni kimlik dizini vektörü 00000001 olacaktır. Veri 35 tabanı motoru için sonraki adım, yeni nesne için bir ilişki dizini vektörü oluşturmaktır. Bu durumda yeni ilişki dizini vektörü 10010001 olmaktadır. Sonraki adım olarak, yeni nesne

ile ilişkili, tüm nesnelerin ilişki dizini vektörleri güncellenmektedir. Bu durumda 1 sayısı ile tanımlanmış nesnenin ilişki dizini vektörü 11111111'e güncellenmektedir ve 3 sayısı ile tanımlanmış nesnenin ilişki dizini vektörü 10010001'e güncellenmektedir. Son adım veri tabanı motorunun nesne türü kimlik dizinini güncellemesidir. Bu durumda yeni nesnenin kimlik dizini vektörü 00000011 yeni vektörü ile sonuçlanan etiketlenmiş D türündeki nesneler için nesne türü kimlik dizinine eklenmektedir.

10

Şekil 14'te görülebileceği gibi, yeni ve güncellenmiş dizin vektörlerinin uzunluğu diğer dizin vektörlerinden bir bit daha uzundur. Bitsel işlemlerin çalışma yöntemine dayalı olarak farklı uzunluklardaki bit vektörleri üzerindeki işlemler halen istendiği biçimde çalışacaktır. Bunun sebebi şudur, bitsel işlem yapılmadan önce daha uzun bit vektörü ile aynı uzunluğa erişmek için daha kısa bir vektörü açık bir şekilde sıfırlar ile doldurulacaktır.

15

20

Şekil 15 bir nesne veri setinden kaldırıldığı zaman veri tabanı motorunun nasıl dizin yapılarını güncelleyebildiğini göstermektedir. Şekildeki bilgi aşağıdakilere göre yorumlanmalıdır:

25

- Şekil üst sol tarafı orijinal veri setini göstermektedir
- Şeklin sağ üst kısmı kesintisiz bir çizgi ile işaretlenmiş çıkarılacak olan nesne ile birlikte veri setini göstermektedir
- Şeklin sağ alt kısmı çıkarılan nesne ile birlikte veri setini ve güncellenmiş dizin yapılarını göstermektedir
- Modifiye edilmiş tüm dizin vektörleri italiktir, ve diğer tüm vektörler modifiye edilmemiştir.

30

35

Bu örnekte, 6 sayısı ile tanımlanmış bir nesne, veri setinden kaldırılmaktadır. Dizin yapılarını güncel tutmak için veri tabanı motoru ilişki dizinini, nesne tipi kimlik dizininin ve silinmiş kimlik bit vektörünü güncellemelidir. Veri tabanı

motoru nesne türü kimlik dizini vektöründen çıkarılmış nesnenin kimlik dizini vektörünün kaldırılmasıyla başlamaktadır. Bu durumda kimlik dizini vektörü, 0000010, yeni bir vektör, 0010000 ile sonuçlanan etiketlenmiş D türündeki nesneler için nesne türü kimlik dizini vektöründen kaldırılmaktadır. Veri tabanı motoru için sonraki adım, çıkarılan nesne ile ilişkili tüm nesnelerin ilişki dizini vektörlerini güncellemesidir. Bu durumda 1 sayısı ile tanımlanan nesnenin ilişki dizini vektörü 1111101'e güncellenmektedir, 4 sayısı ile tanımlanan nesnenin ilişki dizini vektörü 1000101'e güncellenmektedir ve 7 sayısı ile tanımlanan nesnenin ilişki dizini vektörü 1000101'e güncellenmektedir.

Şekil 16 veri tabanı motorunun veri setine eklenmiş nesneler için nasıl kimlik dizini vektörlerini oluşturduğunun şematik bir temsilini göstermektedir. Bu şemayı takiben, Şekil 14'te temsil edilen veri setinin kullanılması Şekil 14'te gösterildiği gibi aynı kimlik dizini vektörünü oluşturacak olan aşağıdaki adımları alacaktır:

- Veri setine yeni nesneler eklendikten sonra, nesnelerin toplam sayısını hesapla, bu durumda 8 nesne
- Bir nesne sayacı oluştur ve bunun değerini yeni nesneler eklenmeden önce veri setindeki nesnelerin toplam sayısına ayarla, bu durumda 7
- Eklenmiş birinci nesne ile başla, bu durumda nesne 8 sayısı ile tanımlanmaktadır

• Veri setindeki nesnelerin toplam sayısının ve en uzun kimlik dizini vektörünün en büyüğüne eşit uzunlukta bir kimlik dizini vektörü oluştur, bu durumda 8

• Bu veri setinde hiçbir nesne silinmediğinden ötürü, silinmiş kimlik bit vektörü herhangi birleri içermemektedir yeni kimlik dizini vektörü nesne sayacı kullanılarak oluşturulmalıdır

• Nesne sayacı ile aynı sayıda olan bir hariç tüm bitleri sıfıra ayarla, bu durumda bit sayısı 7

• Ortaya çıkan bit vektörü 00000001

»Nesne sayacını bir arttır, bu durumda yeni değer 8'dir

- Hiçbir nesne eklenmediğinden ötürü işlem tamamlanmıştır

5 Şekil 17 veri setine eklenen nesneler için veri tabanı motorunun nasıl ilişki dizini vektörü oluşturduğunun ve yeni nesneler ile ilişkili nesneler için ilişki dizini vektörlerini nasıl güncellediğinin şematik bir temsilini göstermektedir. Bu şemayı takiben Şekil 14'te temsil edilen veri seti kullanımı aşağıdaki adımları verecektir:

10

- Eklenmiş olan birinci nesne ile başla, bu durumda nesne 8 sayısı ile tanımlanmaktadır

15

»Nesneler kimlik dizini vektörüne eşit bir ilişki dizini vektörü oluştur, bu durumda 00000001

»Tüm ebeveynler ve tüm çocuklar için kimlik dizini vektörlerini eklemek için bitset VEYA işlemleri kullan, bu durumda 0001000 ve 1000000

20

»Farklı türdeki nesnelerin tüm kardeşlerinin ve ebeveynler için farklı türdeki nesnelerin tüm kardeşlerinin kimliğini eklemek için bitset VEYA operasyonları kullan, bu durumda hiçbir şey

25

»Ortaya çıkan bit vektörü 10010001

»Nesneler kimlik dizini vektörüne eşit bir ebeveyn kimlik bit vektörü oluştur, bu durumda 00000001

»Bu nesne ile ilişkili tüm nesneler için ilişki dizini vektörlerini güncelle, çocuklar hariç

30

»Yeni nesnenin ebeveyni ile başla, bu durumda nesne 3 sayısı ile tanımlanmaktadır

35

- Yeni nesnenin ilişki dizini vektörünü mevcut nesneye eklemek için bir bitset VEYA işlemi kullan, bu durumda 10010001 ile sonuçlanan 10010001'e 1001000'i eklemek için bir bitset VEYA işlemi kullan

5

- Mevcut nesnenin kimlik dizini vektörünü ebeveyn kimlik bit vektörüne ekle
- Ebeveyn kimliği bit vektörünün mevcut değeri 00000001

10

- Mevcut nesnenin kimlik dizini vektörü 0001000
- Sonuç 00010001

15

»Sonraki ebeveyn ile devam et, bu durumda nesne 1 sayısı ile tanımlanmaktadır

- Yeni nesnenin ilişki dizini vektörünü mevcut nesneye eklemek için bir bitset VEYA işlemi kullan, bu durumda 11111111 ile sonuçlanan 10010001'i 1111111'e eklemek için bir bitset VEYA işlemi kullan
- Mevcut nesnenin kimlik dizini vektörünü ebeveyn kimlik bit vektörüne ekle

20

- Ebeveyn kimliği bit vektörünün mevcut değeri 00010001
- Mevcut nesnenin kimlik dizini vektörü 1000000

25

- Sonuç 10010001

30

»Bu bölümde hiçbir ebeveyn olmadığından ötürü işlem tamamlanmıştır

»Bu nesnenin tüm çocuklar için ilişki dizini vektörlerini güncelle

»Bu bölümde hiçbir çocuk olmadığından ötürü işlem tamamlanmıştır

35

- Hiçbir nesne eklenmediğinden ötürü işlem tamamlanmıştır

Şekil 18 veri setinden kaldırılmış olan nesneler ile ilgili nesneler için veri tabanı motorunun ilişki dizini vektörlerini

nasıl güncellediğinin bir şematik temsilini göstermektedir. Bu şemayı takiben Şekil 15'te temsil edilen veri setinin kullanımı aşağıdaki adımları verecektir ve Şekil 15'te gösterildiği gibi aynı ilişki dizinini oluştur:

5

- Kaldırılmış olan birinci nesne ile başla, bu durumda nesne 6 sayısı ile tanımlanmaktadır

10

»Bu nesnenin kimlik vektörünü silinmiş kimlik bit vektörüne ekle

»Nesnenin kimlik dizini vektörü üzerinde bir bitset NOT işlemi yaparak yeni bir bit vektörü oluştur

15

- Kimlik dizini vektörü 0000010

- Sonuç 1111101

»Nesnenin kimlik dizini vektörüne eşit bir ebeveyn kimlik bit vektörü oluştur, bu durumda 0000010

20

»Bu nesne ile ilgili tüm nesneler için ilişki dizini vektörlerini güncelle, çocuklar hariç

»Silinmiş nesnenin ebeveyni ile başla, bu durumda nesne 4 sayısı ile tanımlanmaktadır

25

- Silinmiş nesnenin kimlik dizini vektörünü kaldırmak için silinmiş nesnenin olumsuzlanan kimlik dizini vektörü ve mevcut nesnenin ilişki dizini vektörü arasında bitset bir VE işlemi kullan, bu durumda 1000101 ile sonuçlanan 0000010'u 1000111'den kaldırmak için bir bitset VE işlemi kullan

30

- Mevcut nesnenin kimlik dizini vektörünü ebeveyn kimlik bit vektörüne ekle

35

- Ebeveyn kimliği bit vektörünün mevcut değeri 0000010

- Mevcut nesnenin kimlik dizini vektörü 0000100

- Sonuç 0000110

» Silinmiş nesnenin birinci kardeşi ile devam et, bu durumda nesne 7 sayısı ile tanımlanmaktadır

5

- Silinmiş nesnenin kimlik dizini vektörünü kaldırmak için silinmiş nesnenin olumsuzlanan kimlik dizini vektörü ve mevcut nesnenin ilişki dizini vektörü arasında bitisel bir VE işlemi kullan, bu durumda 1000101 ile sonuçlanan 1000111'den 0000010'u kaldırmak için bitisel bir VE işlemi kullan
- Mevcut nesnenin kimlik dizini vektörünü ebeveyn kimlik bit vektörüne ekle

15

- Ebeveyn kimliği bit vektörünün mevcut değeri 0000110
- Mevcut nesnenin kimlik dizini vektörü 0000001

20

- Sonuç 0000111

» Sonraki ebeveyn ile devam et, bu durumda nesne 1 sayısı ile tanımlanmaktadır

25

- Silinmiş nesnenin kimlik dizini vektörünü kaldırmak için silinmiş nesnenin olumsuzlanan kimlik dizini vektörü ve mevcut nesnenin ilişki dizini vektörü arasında bitisel bir VE işlemi kullan, bu durumda 1111101 ile sonuçlanan 1111111'den 0000010'u kaldırmak için bitisel bir VE işlemi kullan
- Mevcut nesnenin kimlik dizini vektörünü ebeveyn kimlik bit vektörüne ekle

30

35

- Ebeveyn kimliği bit vektörünün mevcut değeri 0000111
- Mevcut nesnenin kimlik dizini vektörü 1000000

- Sonuç 1000111

5 ºBu bölümde hiçbir ebeveyn olmadığından ötürü işlem tamamlanmıştır

ºBu nesnenin tüm çocuklar için ilişki dizini vektörlerini güncelle

ºBu bölümde hiçbir çocuk olmadığından ötürü işlem tamamlanmıştır

- Hiçbir nesne kaldırılmadığından ötürü işlem tamamlanmıştır

15 Şekil 16, Şekil 17 ve Şekil 18 ile gösterilen yukarıda belirtilen ve şemalarda kullanılan silinmiş kimlik bit vektörü daha önceden silinmiş nesneler tarafından kullanılmış olan farklı dizin yapılarında bitleri yeniden kullanabilmek için kullanılmaktadır. Bu veri tabanı motorunun veri setinde mevcut olan nesnelerin maksimum miktarından daha uzun bir uzunlukta dizin vektörlerini asla kullanmamasını sağlamaktadır.

20 Veri tabanı motorunun dizin yapılarını güncelleyebilmesi gereken yerde, veri seti güncellemelerinin son kategorisi, nesneler arasındaki ilişkilerde değişikliklerdir. Eğer bir ilişki bir nesne için değiştirilir ise, örneğin bir ilişkinin hedefi değiştirilir ise veya bir ilişki kaldırılır ise, o halde dizin yapıları nesne ilk kez silinmiş gibi güncellenmektedir ve daha sonra yeni ilişki bilgisine eklenmektedir. Bu şekilde veri tabanı motoru, bir ilişki değiştikten sonra dahi tüm dizin yapılarını tutabilir.

30 Yukarıda açıklanan biçimde dizin yapılarının güncellenmesi zaman içinde parçalanmış dizin yapılarını yapacağından ötürü, yeterli güncellemeler gerçekleştirildikten sonra dizin yapılarının yeniden yapılmasına ihtiyaç bulunmaktadır. Dizin yapıları parçalanmış araçlar olduğundan ötürü, bu bağlamda, silinmiş bir nesnenin kullanıldığı kullanılmamış bitler içerdiğinden ötürü veya bit vektörlerindeki nesnelerin

sırasının artık optimal olmadığından ötürü bundan dolayı daha düşük bir sıkıştırma oranı ve işleme hızı elde edilebilmektedir.

- 5 Buluşun belirli örnek yapılandırmaları özellikle açıklanırken, çeşitli diğer modifikasyonlar buluşun kapsamından ayrılmaksızın teknikte kalifiye kişiler tarafından hâlihazırda anlaşılacaktır. Dolayısıyla, buraya ekli olan istemlerin kapsamını burada ortaya konulan açıklama ile sınırlı olmadığı
- 10 ancak bunun yerine istemlerin buluşun ilgili olduğu teknik alanda kalifiye kişilerin kolaylıkla anlayabileceği mevcut buluşun tüm eşdeğerlerini kapsıyor olarak anlaşılması gerekmektedir.

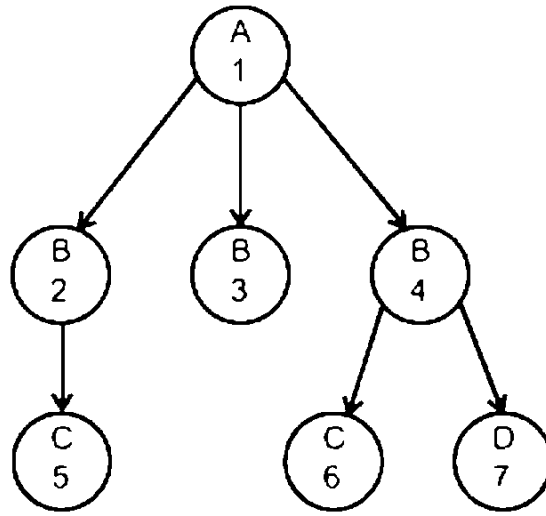
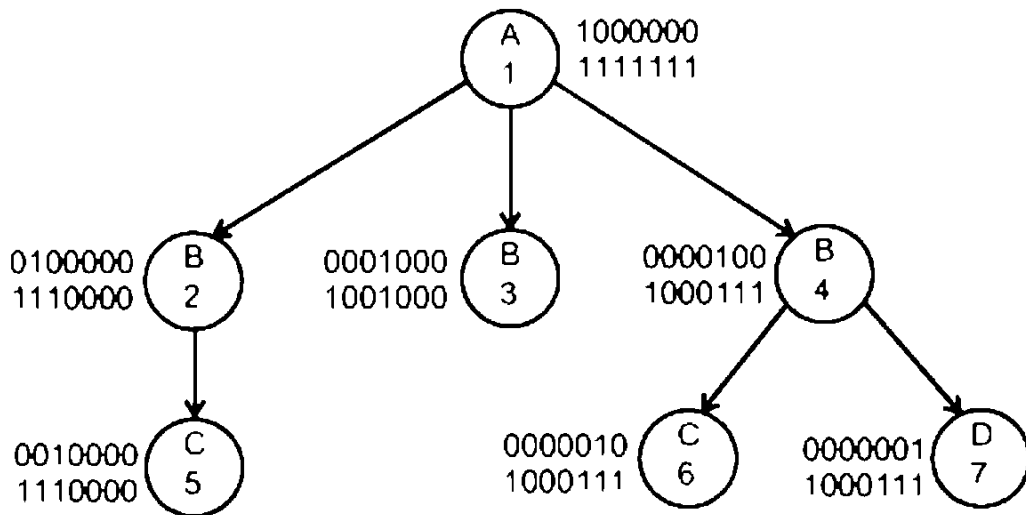
15

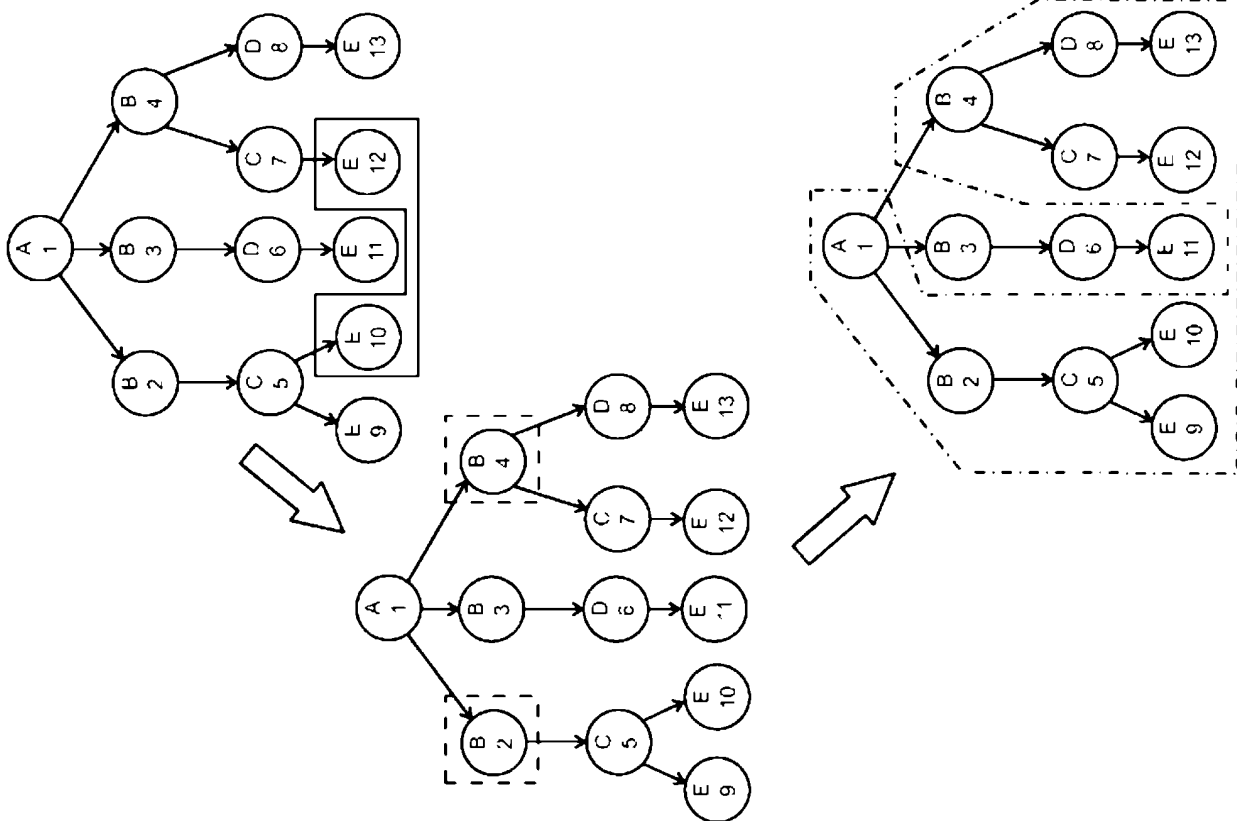
20

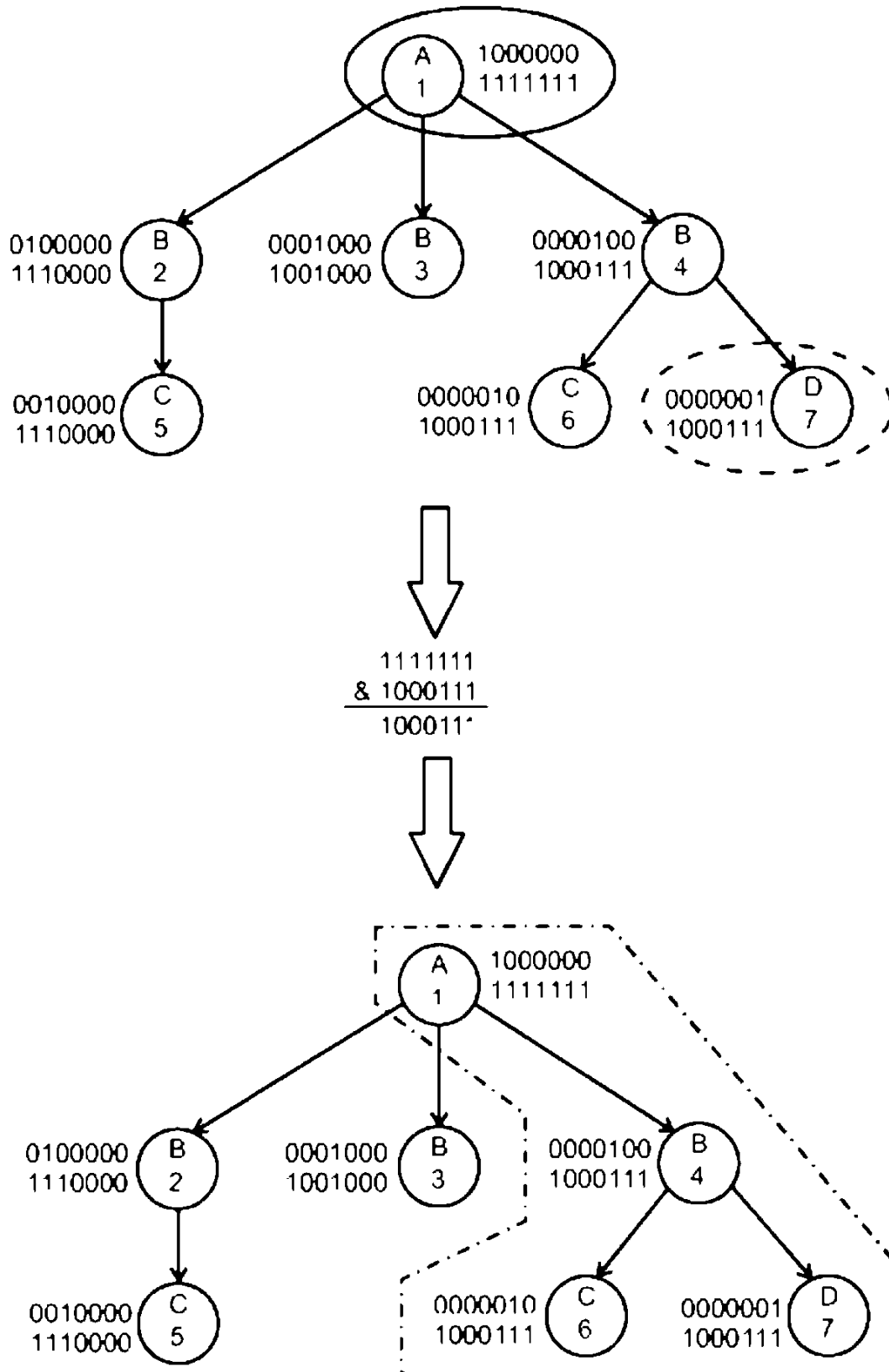
25

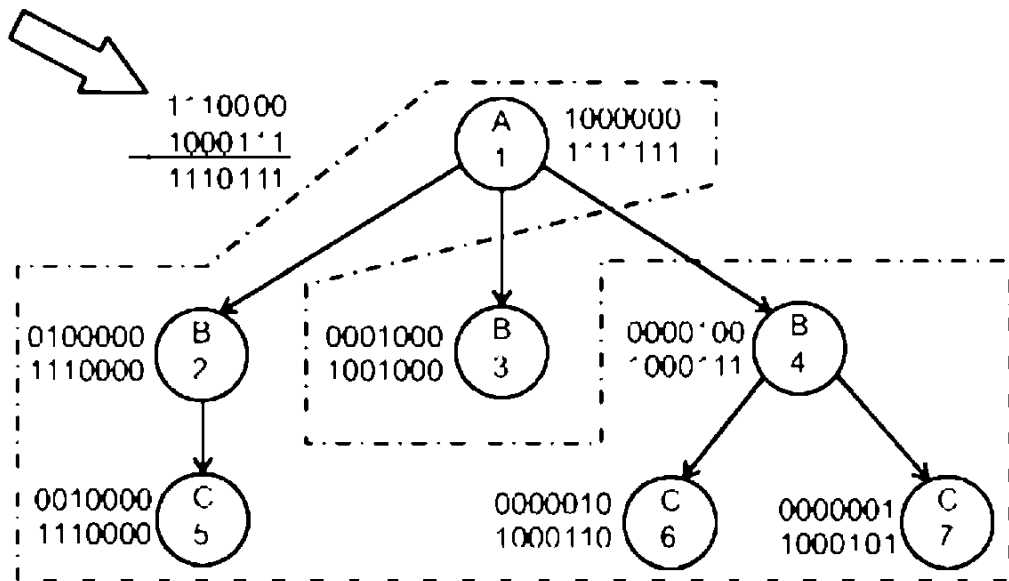
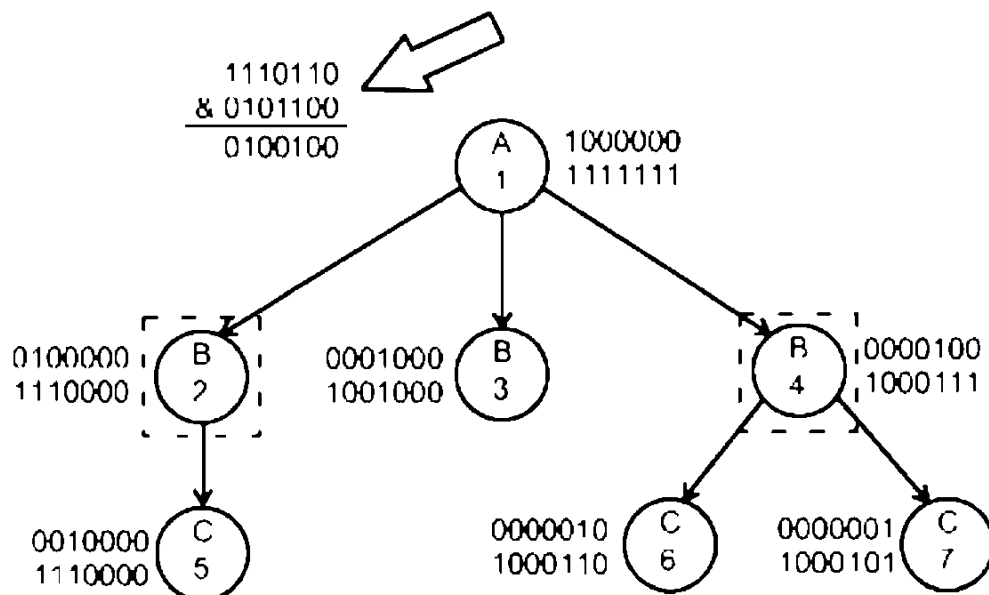
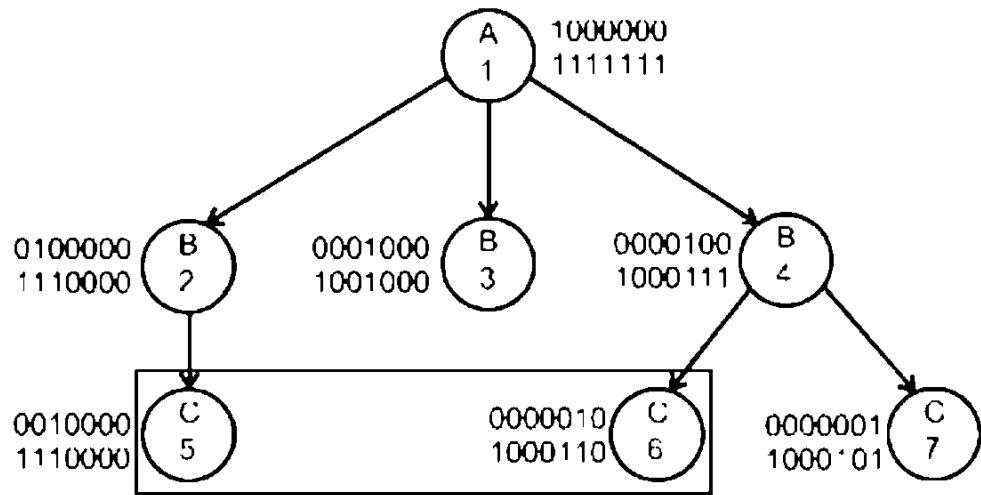
30

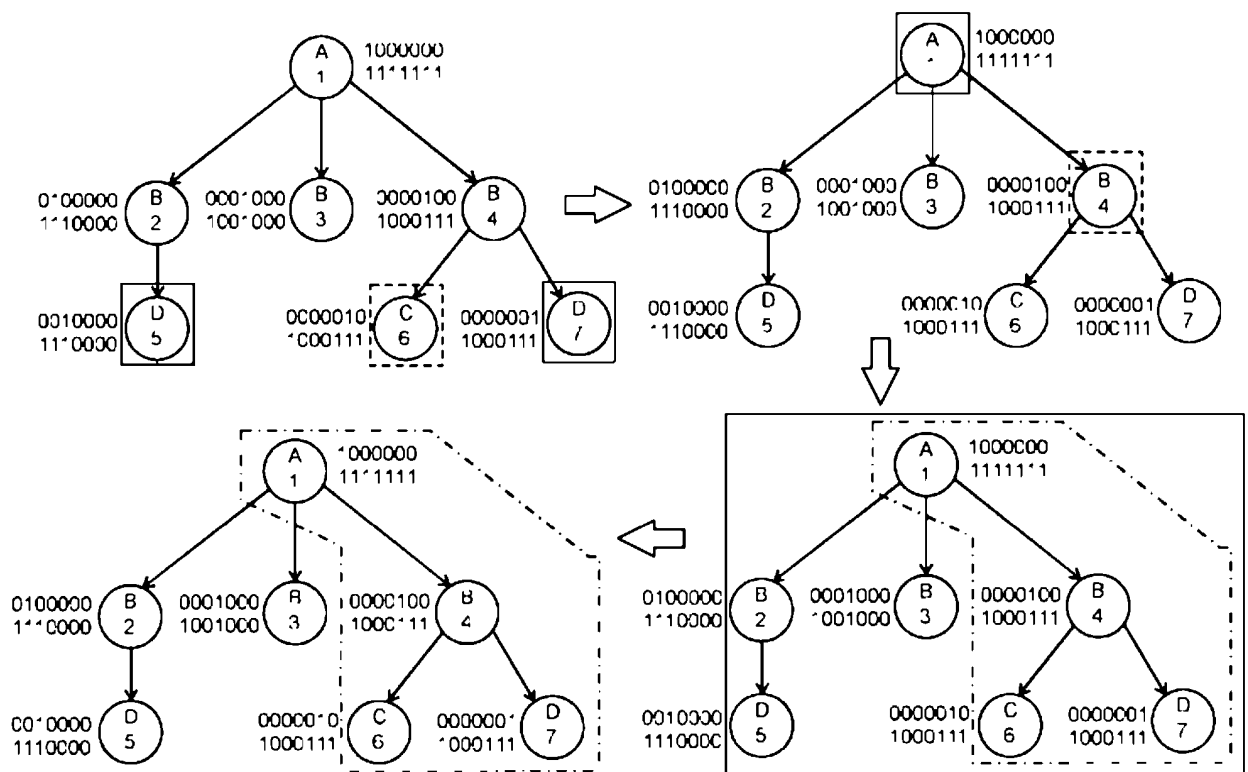
35

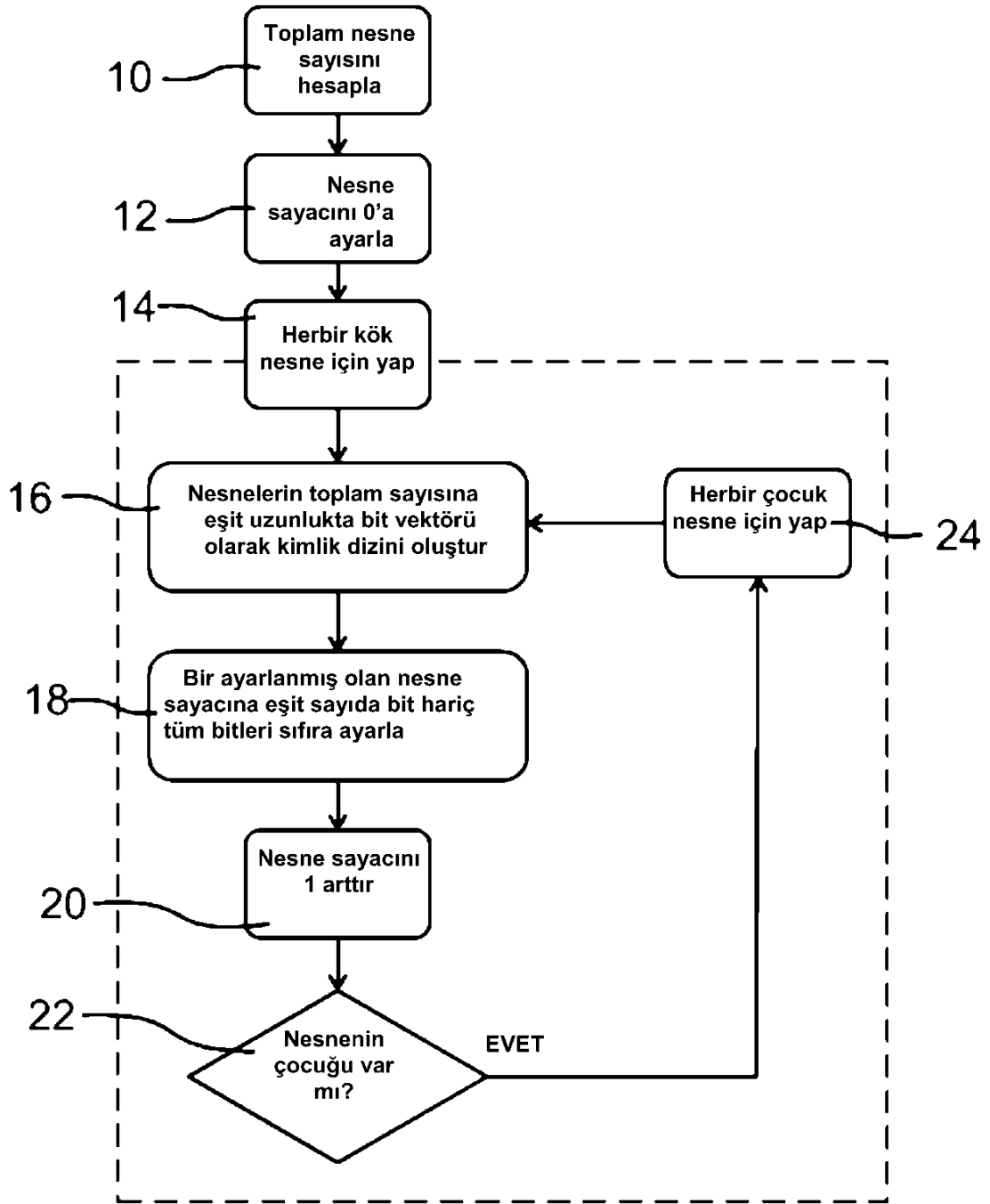
**Şekil 1****Şekil 3**



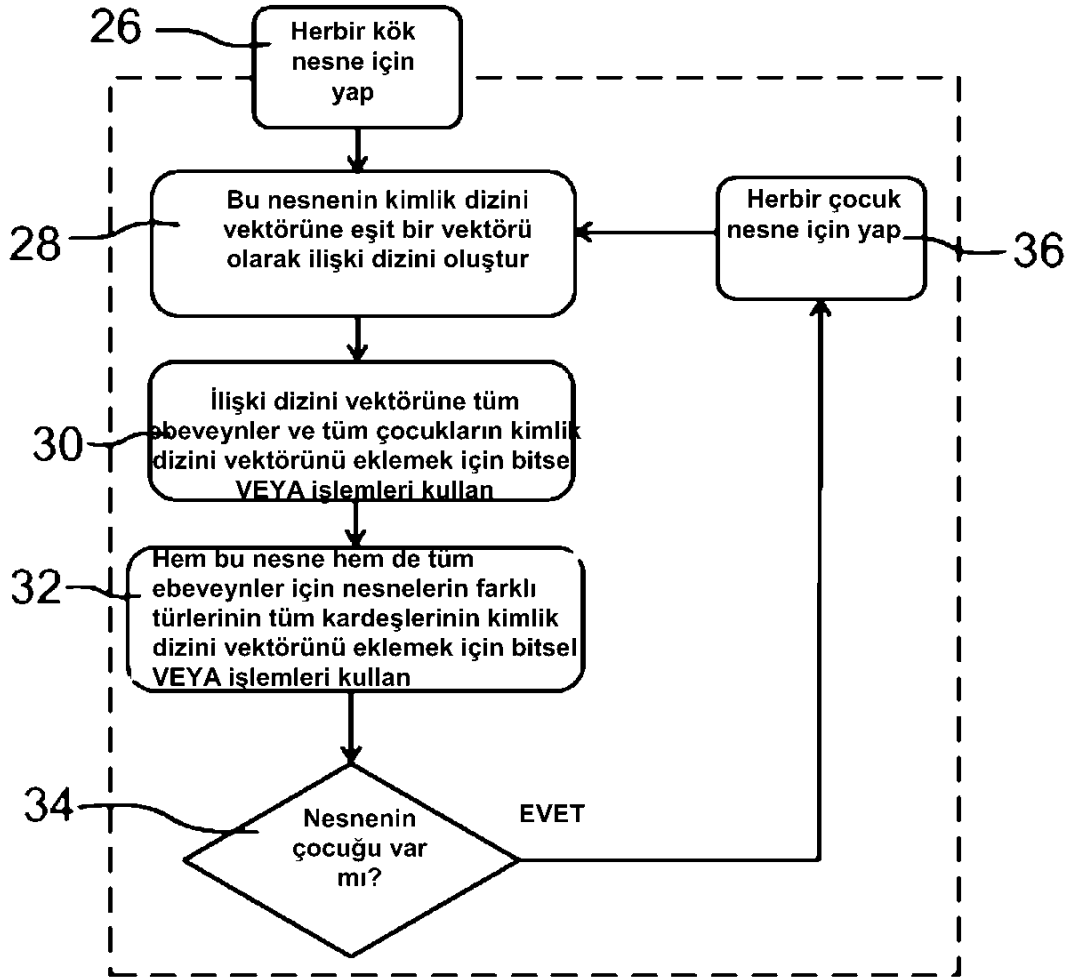
**Şekil 4**



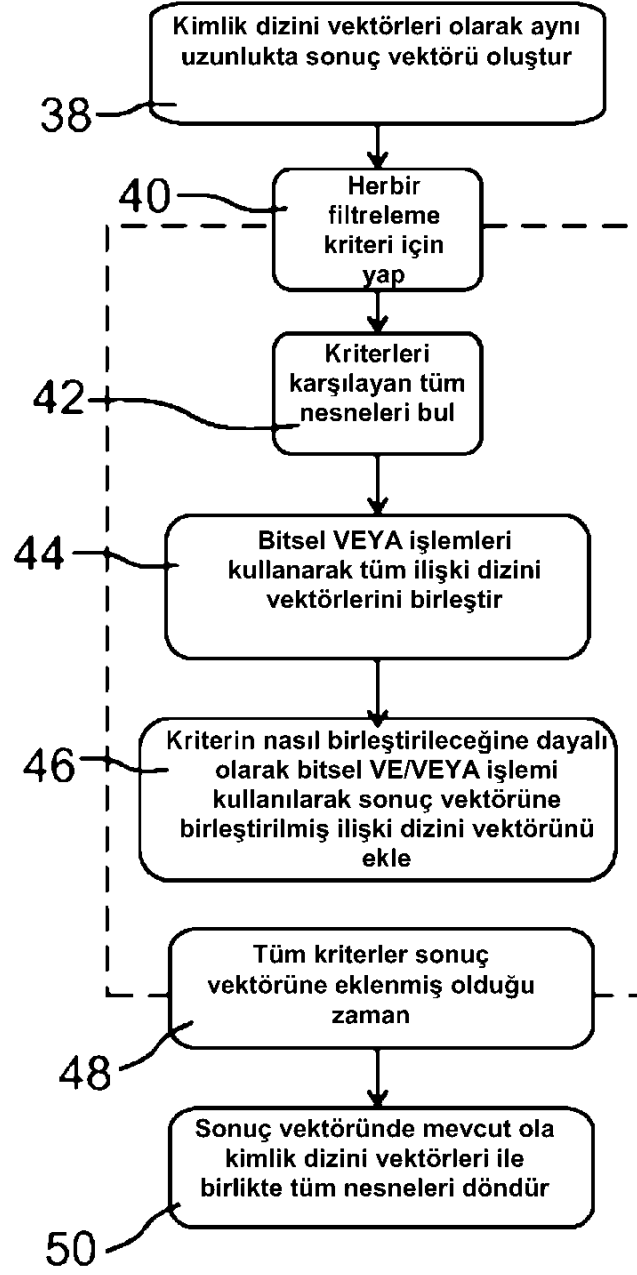




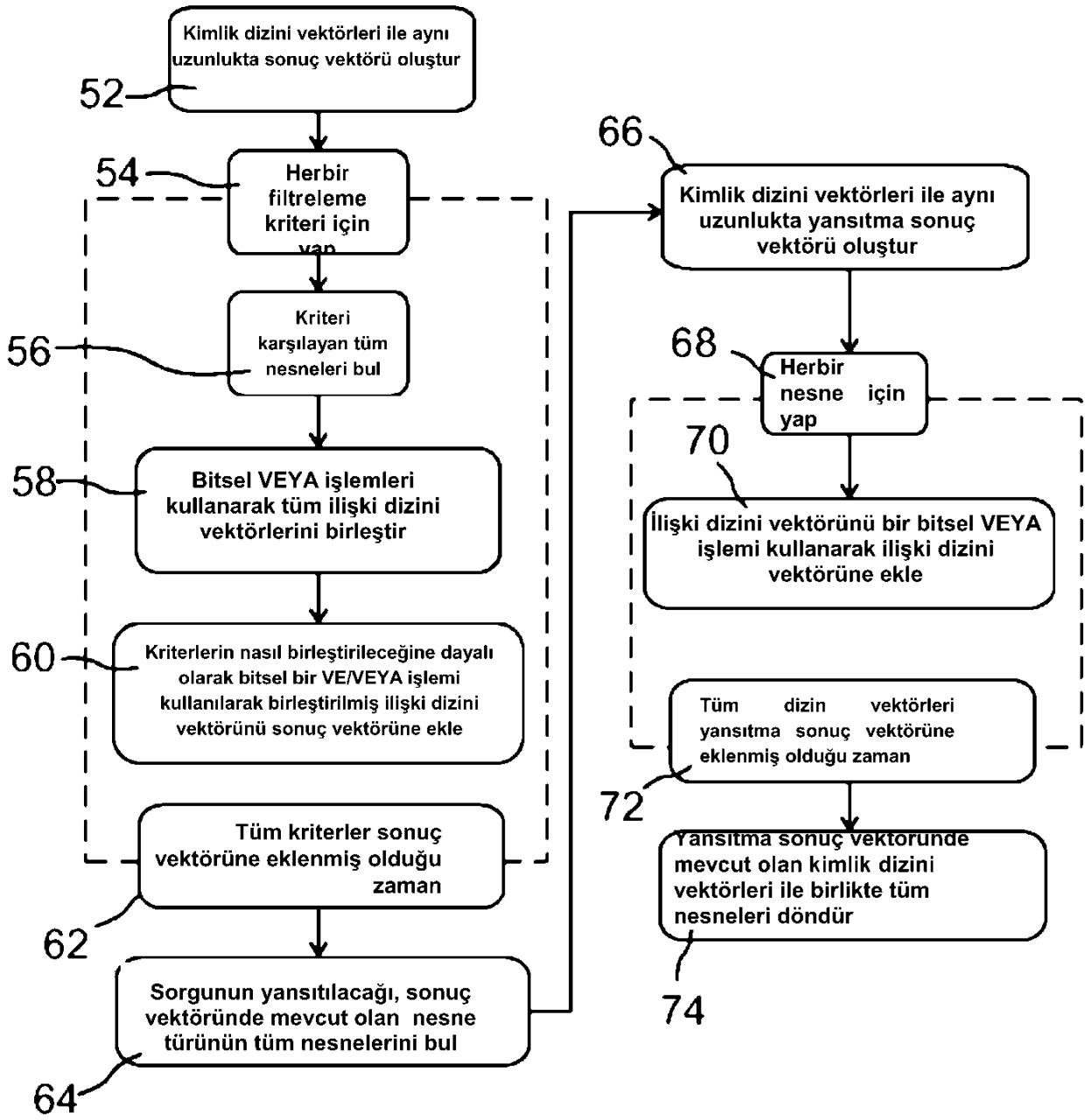
Şekil 7



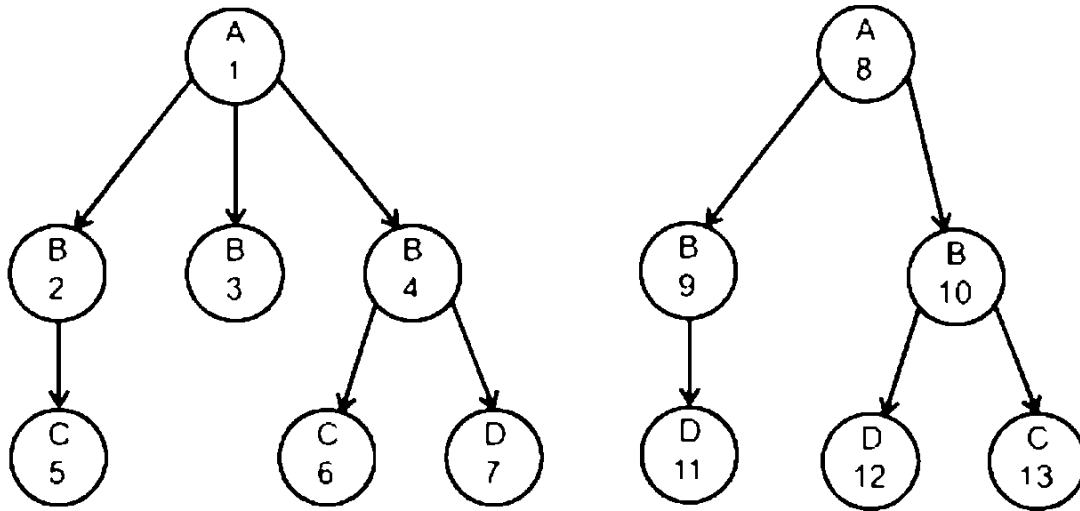
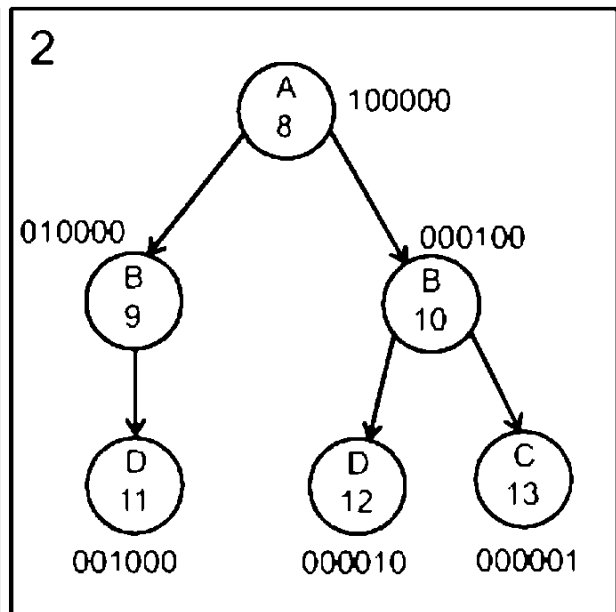
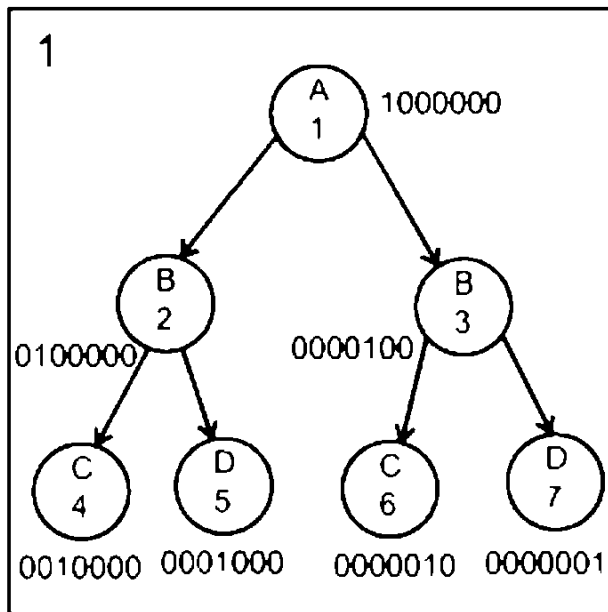
Şekil 8

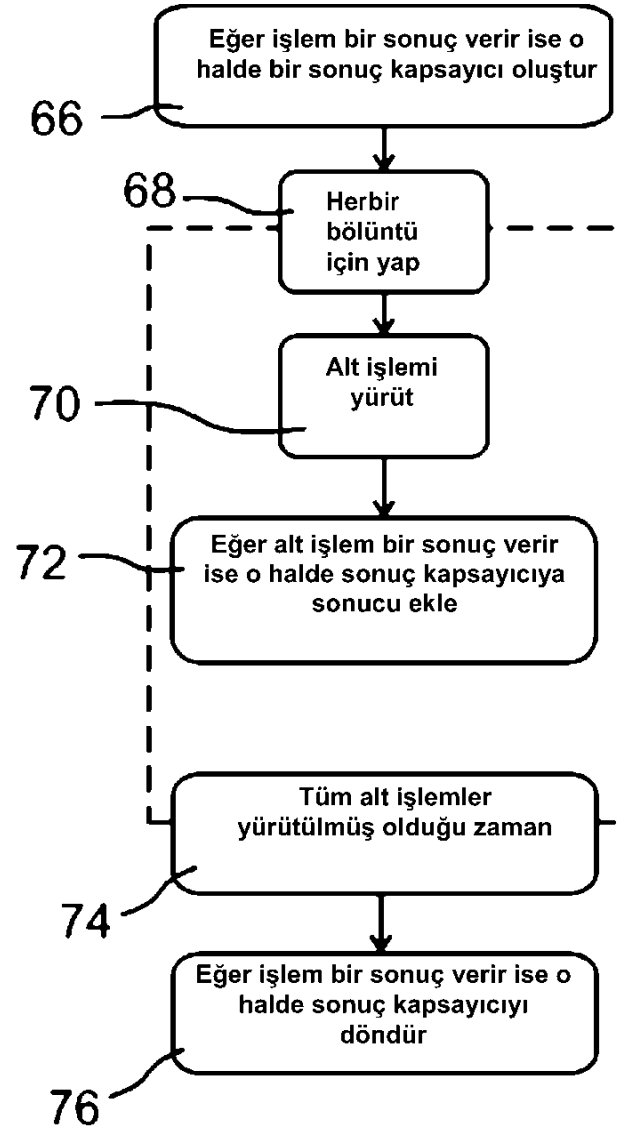


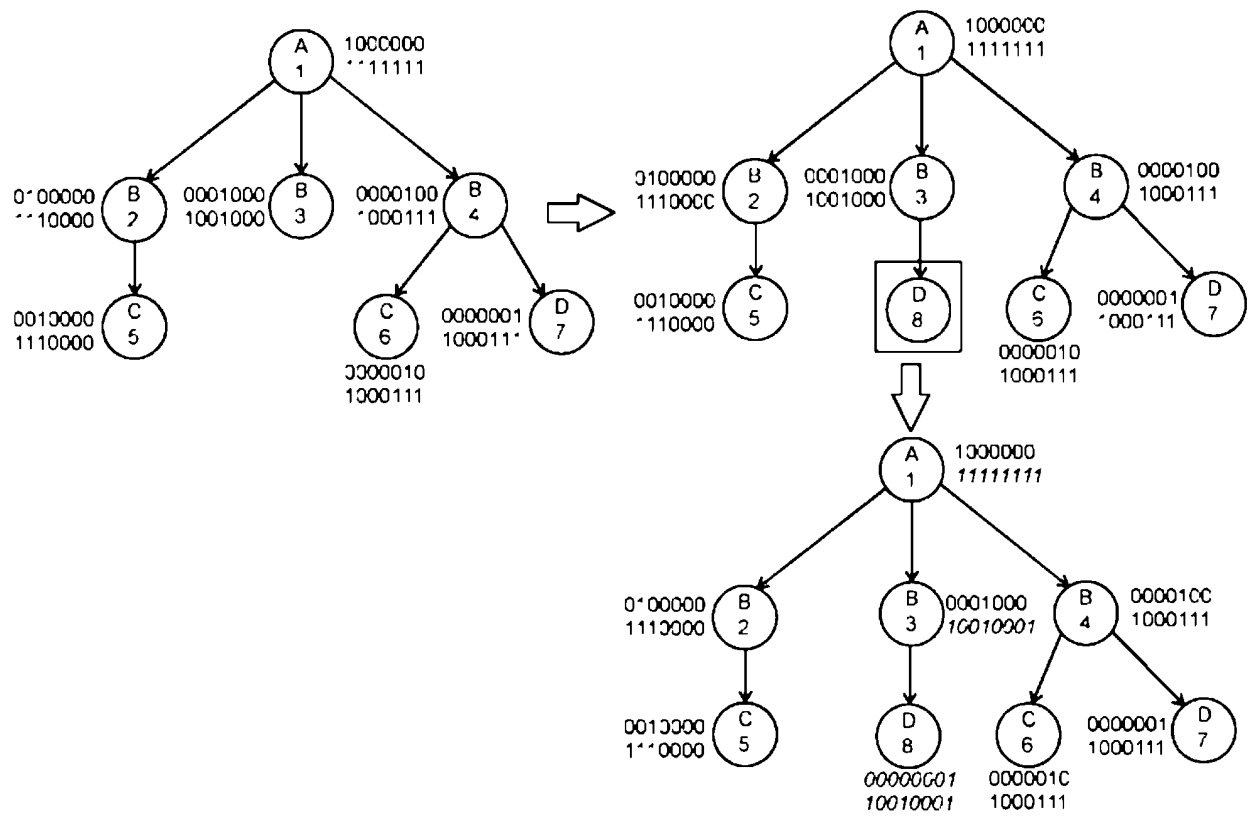
Şekil 9

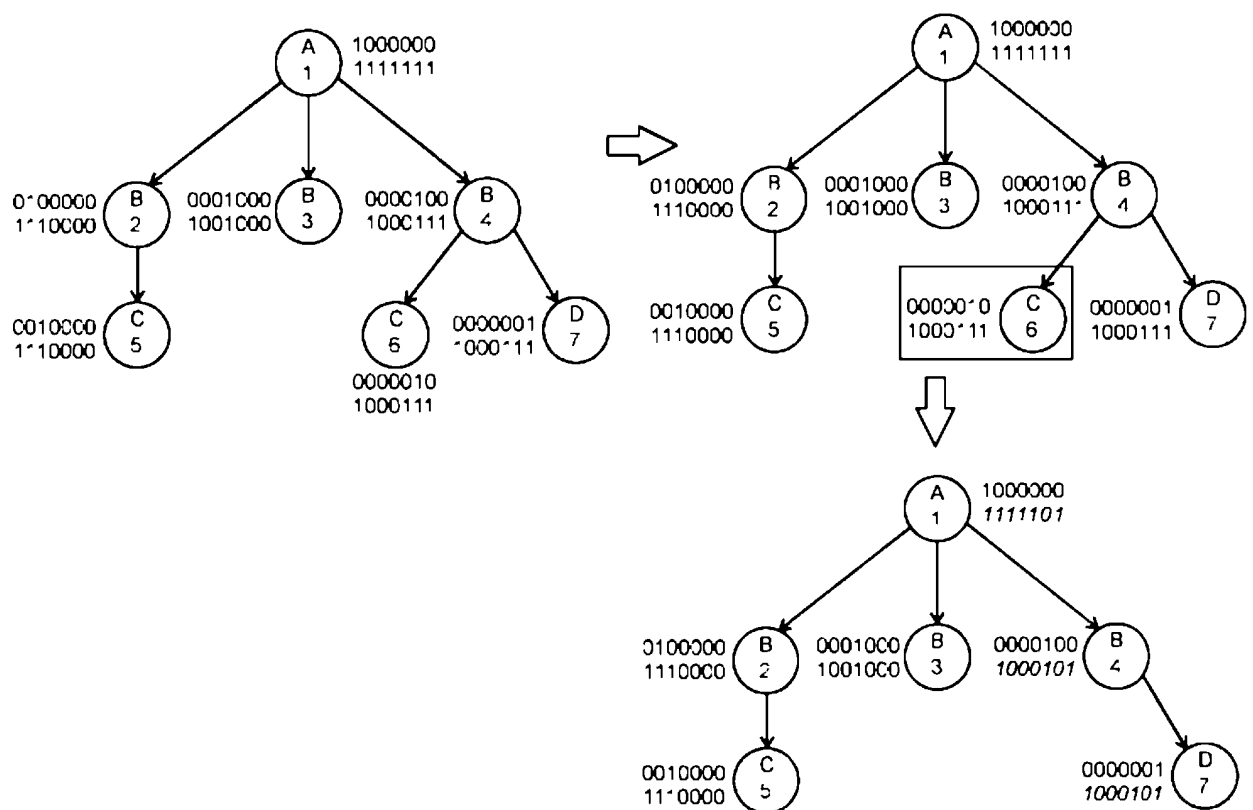


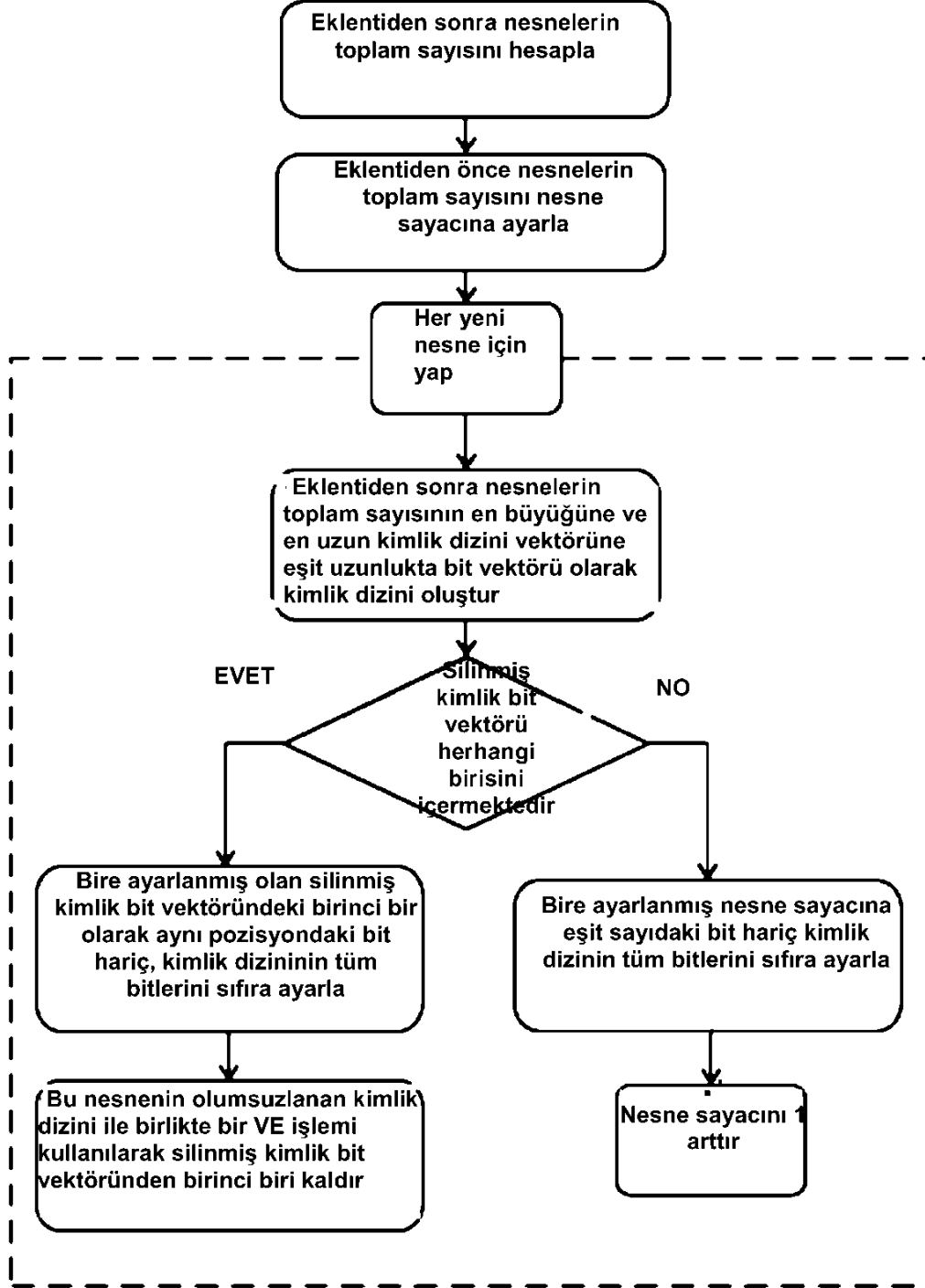
Şekil 10

**Şekil 11****Şekil 12**

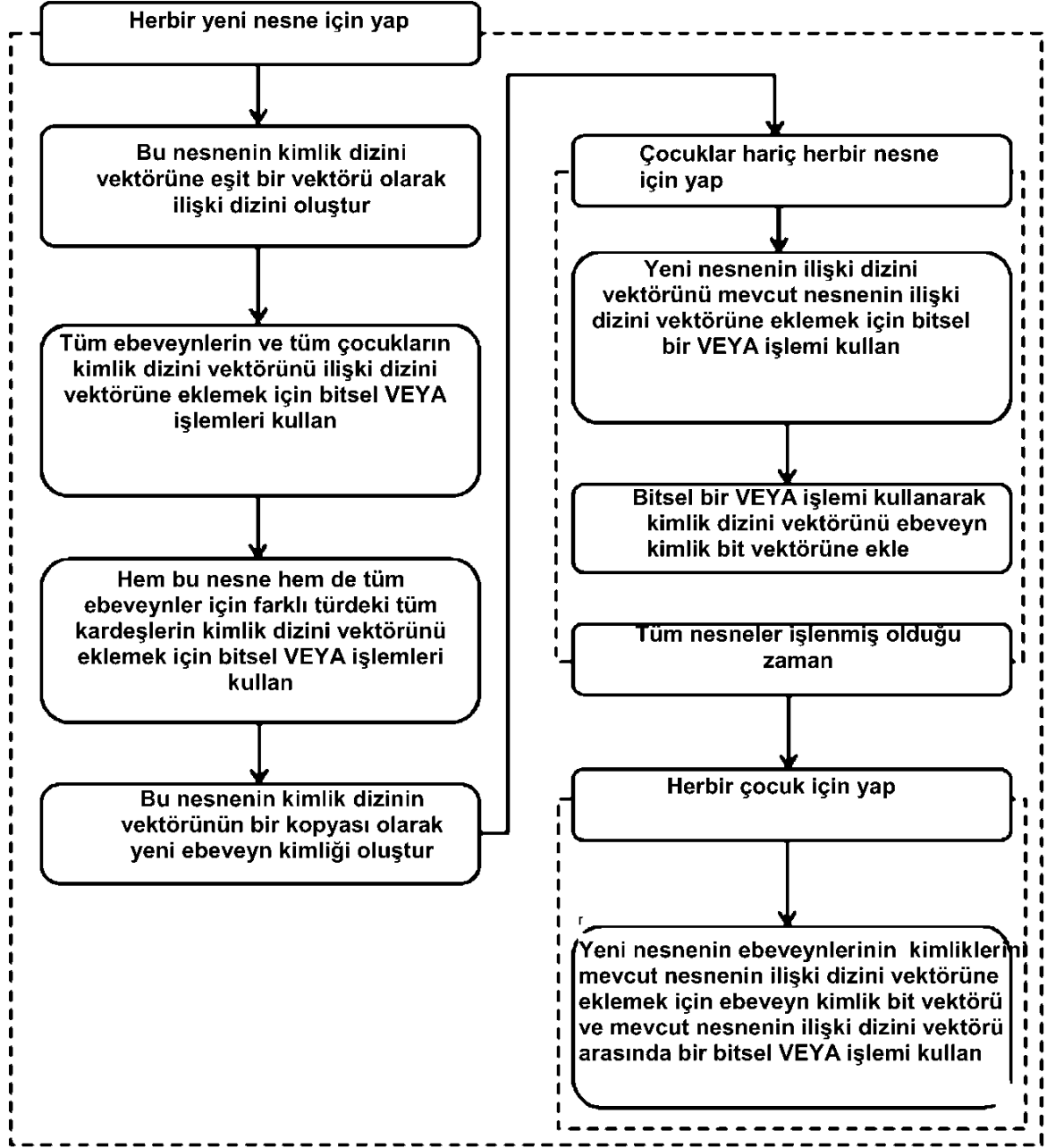
**Şekil 13**



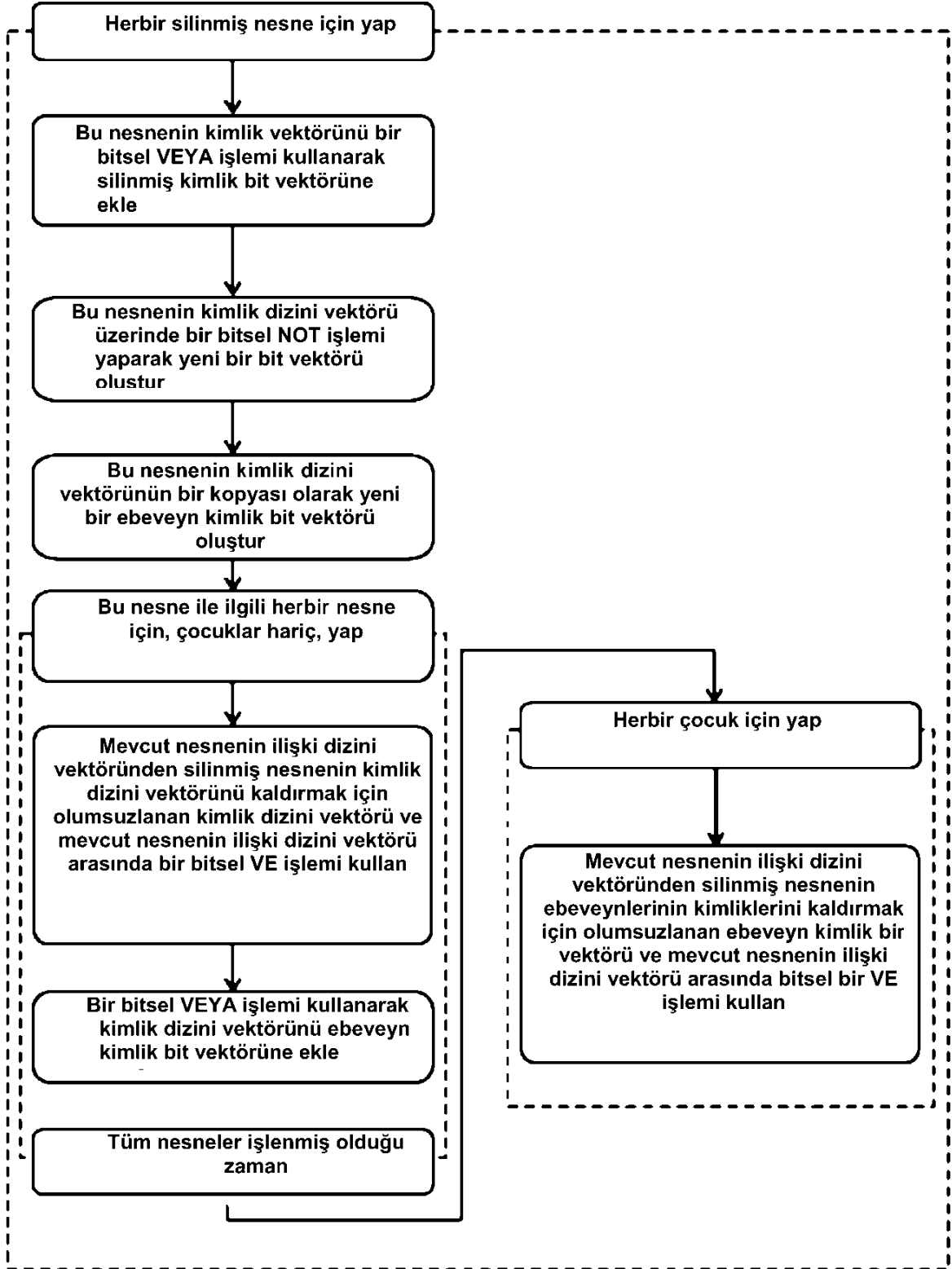




Şekil 16



Şekil 17



Şekil 18