İnternet Servis Sağlayıcılarının Maliyet Optimizasyonu

Mehmet Emin KARAKAŞ
Bilgisayar Mühendisliği Bölümü
TOBB Ekonomi ve Teknoloji Üniversitesi
Ankara, TÜRKİYE
mekarakas@etu.edu.tr

***Özetçe* —Hızlı, güvenilir ve satın alınabilir internet çağımızın gereklilikrindendir. Bu kapsamda Türkiye'deki internet servis sağlayıcı(ISS) firmaların ağ kurma verimlilikleri incelenmiştir. ISS'lerin kullanıcılara daha ucuz ve hızlı hizmet sağlamaları için maliyetlerinin nasıl düşürüleceği konusu araştırılmış ve belirleyici noktanın kazı maliyetleri olduğu tespit edilmiştir. ISS' lerin hizmet götürmek istedikleri yerler için en az kazı yaparak, ideal ağı oluşturmaları hedeflenmiştir. Bu kapsamda problem Steiner Ağacı Problemi(SAP) [1] olarak ele alınmıştır ve problemin çözümü için yakınsama algoritmaları ile cografik data kullanılarak PostgreSQL [4] veritabanı yönetim sistemi ve Python [5] kullanılarak, gerçek hayatta kullanılabilir hale getirilmiştir. Çalışmamızda Türkiye'nin önde gelen internet servis sağlayıcılarından birinden alınan gerçek veriler kullanılmıştır. ISS tarafından fiziksel olarak oluşturulmuş, kullanımda olan ağ ile geliştirmiş olduğumuz programın çıktıları karşılaştırılmıştır. Yapılan kazıların uzunluğunun yaklaşık olarak yarı yarıya azaldığı görülmüştür. ISS tarafından oluşturulan ağda 24868 km kazı çalışması yapılmıştır, programın çıktısına göre ise 14776 km kazı çalışması yapılmasının yeterli olacağı orataya çıkmıştır.**

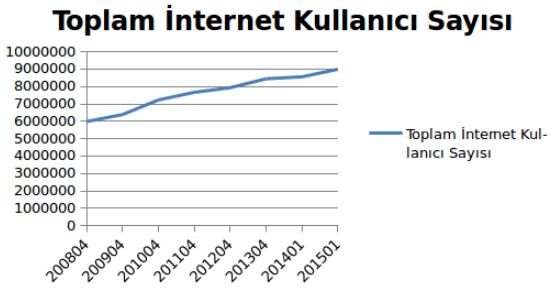
Anahtar Kelimeler—İnternet Servis Sağlayıcı, Ağ, Steiner Ağaç Problemi, Prim'in MST Sezgisel Algoritması, CBS, PostgreSQL, Postgis, Python

***Abstract*—Fast, reliable and afordable internet is the necessity of our age. In this context, the productivity of internet service provider (ISP) companies' in Turkey was examined. It was researced that how ISPs can lower the expences to provide cheaper and faster service for their customers and it was determined that the main point is the excavation expences. It was aimed that setting ideal network with minimizing the excavation work in places where ISPs wanted to serve. In this context, the problem was considered as The Stainer Tree Problem [1] and to solve the problem PostgreSQL [4] relational database management system and Python [5] based tool which is usable in real life was developed with using the convergence algorithms. In this study, real data was taken from one of the pioneer ISPs in Turkey is used. The outputs of the software that I developed was compared with the network that in use and physically created by the ISP. The excavations showed that the length of approximately halved. There was 24868 km long excavation for the network which was created by the ISP, according to the output of the software, it is revealed that 14776 km long excavation could be enough.**

Keywords—Internet Service Provider, Network, Steiner Tree Problem, Prim's MST Heuristic Algorithm, GIS, PostgreSQL, Postgis, Python

I. GİRİŞ

ISS şirketleri belirli bir ücret karşılığında veya ücretsiz olarak ineternet ağına bağlanmanızı sağlayan şirketlerdir. Bu şirketler hizmet verilecek olan tüm lokasyonları birbirine bağlayarak global ağlara bağlanmanızı sağlamaktadır.



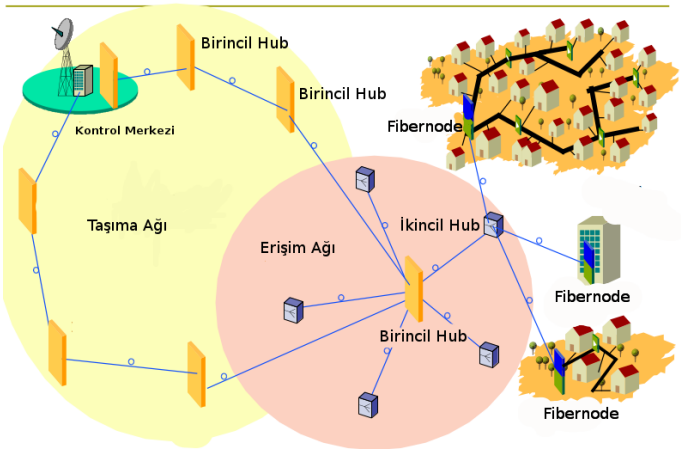
Şekil 1: BTK raporlarına dayanarak hazırlanmıştır. [6]

Şekil 1 yıllara göre internet kullanıcı sayısındaki artışı göstermektedir. Mobil internet kullanan kullanıcılar şekil 1'e dahil edilmemiştir.ISS sektöründe bir çok oyuncu bulunmaktadır. Bu oyuncuların pazardaki payları şebeke genişlikleri ile doğru orantılıdır. Pazarın müşteri sayısında ki artış hızının azalmaya başladığı düşünülürse, şebekenizin genişlemesi pazar payınızı artıracağı bilinen bir gerçektir. Yani pazardaki müşteri sayısı artmıyorsa, müşteri sayınızın ve pazardaki yerinizi artırabilmenin yolu şebekenizi genişleterek diğer firmaların pazar paylarını küçültmektir.

Bu bildiri Türkiye'nin önde gelen ISS şirketlerinden birinden alınan gerçek dataların ve bilgilerin işlenmesi sonucu oluşturulmuştur. Verilerini incelemiş olduğumuz şirket ağ kurma işlemini kendi lokal ağlarını kurmak yoluyla oluşturmaktadır. Oluşturulmuş olan lokal ağların birbirine bağlanması başka bir özel şirket tarafından sağlanmaktadır. Böylelikle lokal ağlarınız global ağlara bağlanacak şekilde inşa edilir. Genel olarak mesafeye göre lokal ağlar kurulmuştur. Büyük şehirler veya

birbirine uzak ilçeleri bulunan iller haricinde tek bir lokal ağ, diğer şehirlerde ise birden fazla lokal ağ bulunmaktadır. (Örneğin; Yalova'da tek bir lokal ağ bulunurken, İstanbul altı adet lokal ağdan oluşmaktadır.) İncelemiş olduğumuz şirket Türkiye genelinde yirmi dört ilde hizmet vermektedir. Her bir il kendi başına tek bit lokal ağ oluşturacak şekilde tasarlanmıştır ve bu lokal ağ birden fazla lokal ağın birleşmesiyle oluşmuş olabilir. Toplamda 523629 adet binaya hizmet götürülmektedir. Herbir lokal ağ içerisinde kontrol merkezi denilen yapıdan yayın başlar ve yine kontrol merkezinde sona erer. Her bir lokal ağ içerisinde Metro Ethernet ile iletim sağlanmaktadır. Kontrol merkezinden çıkan yayın fiber optik kablolar aracılığı ile düğümlere iletilir. Düğümler yayının dağıtımının yapıldığı yerdir. Buradan sonra fiberoptik kablolar veya koaksiyel kablolar aracılığıyla binalara dağıtılmaktadır. Lokal ağların birbirine bağlanması başka bir şirket tarafından sağlanan dark fiber ağlarla sağlanmaktadır.

Şebekenizin genişletmeniz daha çok aboneye hizmetlerinizi ulaştırmanız anlamına gelmektedir. Şebekenizi genişletirken maliyetlerinizi ne kadar düşürebilirseniz, müşterilerinize bu hizmetlere ulaşma ücretlerini de düşürebilirsiniz.



Şekil 2: Lokal ağın genel görünüşü

Şekil 2'deki Kontrol Merkezi tüm yayının başlangıç ve bitiş noktasıdır. Kullanıcıların internete çıkış noktası burasıdır. Kontrol merkezi global ağlara bağlanma noktasıdır. Buradan çıkan yayın Taşıma ağı adı verilen katmanla birlikte birincil hublara taşınmaktadır. Birincil hublar ikincil hublara verileri taşımaktadır. İkincil hublardan veriler fibernode bölgelerine taşınmaktadır. Fiber node bölgeleri kendi içerisinde verilerin dağıtımını yapmaktadır. Bu bildiri kapsamında ilgilendiğimiz kısım fibernode bölgelerindeki bulunan hizmet verilecek tüm binaların en az kazı yapılacak şekilde bir ağ kurulmasıdır. En iyi ağı oluşturarak maliyetin en aza düşürülmesi hedeflenmiştir. Fibernode bölgesindeki ağ la ilgilenmemizin nedeni karmaşık olarak yapılacak kazı yoğunluğunun bu bölgede olmasıdır. Yapılan kazıların maliyet açısından en fazla oranı bu bölgelerdeki kazı çalışmalarına aittir.

II. ISS LERIN ŞEBEKE KURMA MALIYETLERI

A. Maliyet kalemleri

ISS ler genel olarak kuracakları ağlarda dört ana kalem üzerinden maliyet hesabı yapmaktadırlar. Maliyet kalemleri ve ortalama fiyatlar gerçek verilerdir.

Maliyet hesabı yapılan ana kalemlerin açıklamaları aşağıdadır;

1) Kazı Maliyeti: Fiziksel olarak yapılan kazıların maliyetidir. Bu maliyet kalemin içerisinde kullanılan kazı için gerekli olan araçların maliyetleri, asfaltlama ve sökme çalışmaları dahil maliyet hesabıdır.

2) Kurum Ödemeleri: Fiberoptik hat / kablo döşemek için kamu kurum ve kuruluşlarına ödenen ücretler toplamıdır. Bu maliyet kalemi her il için değişkenlik göstermektedir. Her belediye kendi ruhsat veya diğer gider işlemleri için farklı fiyat politikaları izlemektedir. Ankara iline ait ücret ortalamaları alınmıştır.

3) Pasif Elemanlar: Fiberoptik hat döşemek için kullanılan pasif elemanların maliyet hesabıdır. Bu maliyet kaleminin içerisine fiberoptik kablo maliyeti, kullanılan HDPE(fiberoptik kabloların içerisine döşendiği poliüretan borular) gibi cihaz kategorisine girmeyen malzemelerin maliyetidir.

4) Aktif Cihaz: Şebeke inşası sırasında kullanılan aktif cihazların maliyet hesabıdır. Erişim ve santral tarafında kullnılan anahtarlama ekipmanları, cmts gibi internet erişimini yönlendiren cihazlar dahil maliyetlerdir. Şebekede kullanılan tüm elektronik cihazlar bu kaleme girmektedir. Bu kalemin giderleri hizmet sayısı ile doğru orantılıdır. Yeni kazı çalışması yapılmadan hizmet verdiğiniz kişi sayısını arttırıldığı zaman bu maliyet kaleminde artış görülmektedir.

B. Maliyet Fonksiyonu

1) Kısaltmalar:

- M: Verilecek hizmet sayının toplamıdır. Burada kaç adet internet hizmeti veriyorsanız toplam sayısıdır.
- F: Toplam maliyeti ifade etmektedir. Maliyet fonksiyonun sonucudur.
- KM: Yapılacak olan kazı maliyetinin toplamını ifade etmektedir.
- KU: Yapılacak olan kazının toplam uzunluğudur. Metre cinsinden ifade edilmektedir.
- KO: Kurumlara yapılacak ödemeler toplamıdır.
- PE: Şebekede kullanılan pasif elemanların toplam maliyetidir.
- AC: Şebekede kullanılan aktif cihazların toplam maliyetidir.

2) Fonksiyon(Maliyet Fonksiyonu):

$$AC = (M/1000) * 35000 \quad (1)$$

Her bin hizmet için ortalama olarak 35000 TL aktif cihaz maliyeti bulunmaktadır. Toplam hizmet sayınızın bine

bölümünün 35000 ile çarpımı tüm müşterilerinize hizmet vermek için yapacağınız aktif cihaz yatırımıdır.

$$PE = KU * 10 \tag{2}$$

Şebekenizi inşa edebimeniz için gerekli olan pasif cihazların toplam maliyetidir. Toplamda yaptığınız kazı uzunluğunuzun 10 TL ile çarpımı pasif cihaz yatırım maliyetinizi vermketedir. Buradaki 10 TL bir metre kazı başına düşen ortalama pasif eleman maliyetidir.

$$KO = KU * 175 \tag{3}$$

Şebekenizin inşası sırasında belediyelere ve devlete ödemeniz gereken maliyeti ifade etmektedir. Kazı maliyetinizin 175 TL ile çarpımı tolam kurum ödemeleri maliyetinizi ifade etmektedir.Kurum ödemeleri kalemi Ankara ilinde yapılan her bir metre kazı için ortalama maliyettir.

$$KM = KU * 20 \tag{4}$$

Kazı maliyeti kazı uzunluğunuzun 20 TL ile çarpımı sonucunda hesaplanmaktadır. Buradaki 20 TL her bir metre kazı yapabilmeniz için ortalama maliyetinizdir.

$$F = AC + PE + KO + KM \tag{5}$$

Toplam maliyetiniz aktif cihazlar, pasif elemanlar, kurum ödemeleri ve kazı maliyetinizin toplanması sonucu ortaya çıkmaktadır.

Maliyet hesabından yola çıkarak maliyetlerinizi arttıran ana etmenin kazı uzunluğu olduğu görülmektedir. Maliyeti oluşturan üç ana hesabın kazı uzunluğu ile doğru orantılı olarak arttığı gözlemlenmektedir. Diğer kalemin ise şebekenizde bulunan hizmetlerinizin toplam sayısı ile alakalı olduğu görülmektedir. Bu yüzden kazı uzunluğunuzu ne kadar azaltabilirseniz o kadar daha az maliyetle aynı ağı oluşturmuş olursunuz.

ISS lerin maliyet en iyilemesini yaparken oluşturacağınız ağın en kısa olacak şekilde inşa edilmesi ile ilgili olarak çalışma yapmış bulunmaktayız. Bu kapsamda problemi Steiner Ağacı problemi olarak ele aldık ve problemin çözümüne yönelik Prim'in MST sezgisel algoritmasını [2] iyileştirerek bir algoritma geliştiridik. Aynı zamanda geliştirmiş olduğumuz algoritmayı gerçek datalar üzerinde çalıştırarak gerçek hayatta kullanılabilir bir hale getirdik.

III. Problemin Tanimi ve Modelleme

A. Steiner Ağacı Problemi

Bilgi: Steiner ağacı problemi(SAP) [1] genellikle ağ dizaynı ve kablolama problemleri için kullanılmaktadır. Bize verilmiş olan bir grup düğümün(Bizim problemimiz için lokasyon) olabildiğince ucuz bir şekilde birbirine bağlanmasıdır. Gerçek hayatta bir çok problem için SAP kullanılmaktadır. Örnek olarak; kablolama, su borularının döşenmesi, ısıtma veya doğalgaz sistemlerinin döşenmesi verilebilir.

B. Problem

SAP problemi klasik kombinatoryal opimizayon problemidir. Bu nedenle polinom zamanda problemim çözümü mümkün değildir. NP-hard bir problemdir. Bize köşe(V) ve kenar(E) oluşan bir G çizgesi verilmiş olsun. T köşe kümesi V köşelerinin bir alt kümesi olsun. T köşeleri terminal olarak adlandırılır. Bütün terminalleri içerecek şekilde tüm köşelerin birleştirilmesi ve bu birleştirme işleminin en ucuz şekilde yapılması gerekmektedir.

IV. Kullanilan Algoritmalar

A. Prim'in Minumun Yayılan Ağaç Problemi

Minumum yayılan ağaç problemi şebeke optimizasyonunda en çok kullanılan algortimalardan biridir. Porblemin formulasyonu:

$$G = (V, E) \tag{6}$$

şeklindedir. Bizim problemimizdeki terimleri algoritmaya uyarladığımız zaman:

$$E = \{e_1, e_2, ..., e_m\} \tag{7}$$

Hizmet verilecek noktaları ifade etmektedir.

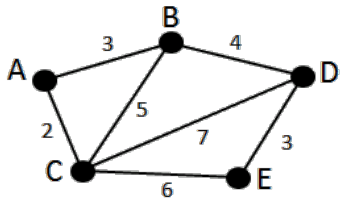
$$V = \{v_1, v_2, ..., v_m\} \tag{8}$$

Şebeke genişleme çalışması sırasında düğümleri birleştirmek amacıyla yapmış olduğunuz kazılar kenarları oluşturmaktadır.

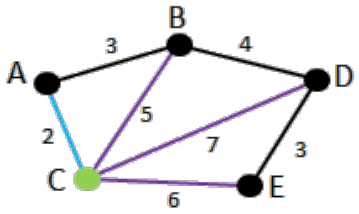
$$W = \{w_1, w_2, ..., w_m\} \tag{9}$$

İki düğüm arasında yapılan kazının uzunluğu algoritmadaki kenar uzunluğuna denk gelmektedir. Prim'in MST algortiması polinom zaman da çözülebilen bir problemdir. Ayrıca MST Steiner ağaç probleminin özelleşmiş halidir.

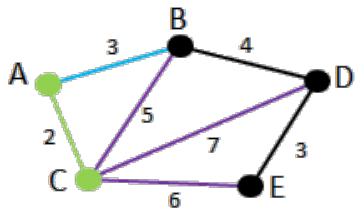
1) Örnek:



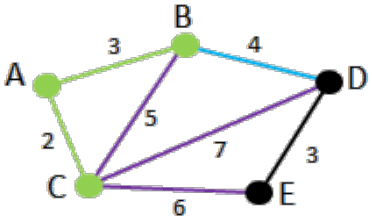
Şekil 3: Bu çizgenin orijinal hali. Ayrıtların üzerindeki sayılar ağırlıkları. Ayrıtlardan hiçbiri henüz seçili değil ve C düğümü başlangıç düğümü olarak rastgele seçildi.



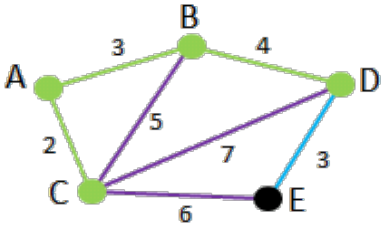
Şekil 4: İkinci olarak seçilecek düğüm C düğümüne en yakın olan A(2), B(5), D(7) ve E(6) düğümlerinin en yakın olan A(2) düğümüdür.



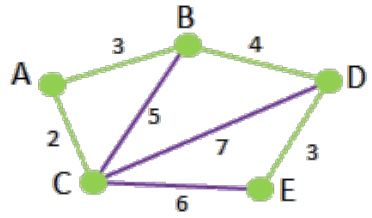
Şekil 5: Sonraki seçilecek olan düğüm C veya A düğümlerine en yakın olan AB(3), CB(5), CD(7) ve CE(6) düğümlerinin en yakını olan B düğümüdür.



Şekil 6: Algoritma devam edince A, B veya C düğümlerine en yakın olan D düğümü seçilir.



Şekil 7: A, B, C veya D düğümlerine en yakın olan E düğümü seçilir.



Şekil 8: Yeşil çizgilerle belirtilen ağaç tüm düğümleri kapsayan en hafif ağaçtır.

2) Sözdekod:

Algorithm 1 Prim

procedure PRIM(GRAPH, SOURCE)
  $Q \leftarrow PQinit()$
  for each*vertex v in graph G* **do**
    $key(v) \leftarrow \infty$
    $pred(v) \leftarrow nil$
    $PQinsert(v, Q)$
  $key(s) \leftarrow 0$
  while $!PQisempty(Q)$ **do**
    $v = PQdelmin(Q)$
    for each*edge v-w s.t. w is in Q* **do**
      if $key(w) > c(v, w)$ **then**
        $PQdeckey(w, c(v, w), Q)$
        $pred(w) \leftarrow v$

B. Dijkstra' nın en kısa yol algoritması

Coğrafi veri üzerinde rota bulma işlemi için kullanılmıştır. Vermiş olduğumuz iki nokta arasında en kısa yolu bulmak amacıyla çalışmamız içinde kullanılmıştır. [3]

C. SAP Probleminin Yakınsama Algoritması ile Çözümü(Prim'in Minumum Yayılan Ağaç Sezgisel Algoritması)

1) Problemin Formülasyonu:

$$G = (V, E, w) \tag{10}$$

şeklindedir. Yönsüz ve negatif kenarı bulunmayan bir çizge kullanılmaktadır.

$$L \subset V \tag{11}$$

L tüm köşelerin alt kümesidir. Bu köşeler terminal olarak adlandırılır.

$$T \subset G \tag{12}$$

T çizgesi ise tüm köşelerin kenar ağırlıkları toplamı en az olacak şekilde birleştirilmiş halidir.

2) Sözdekod:

Algorithm 2 SAP Probleminin MST Yakınsama Algoritması

procedure STPWITHMST(GRAPH, SOURCE)
  $ConstructthemetricclosureG_Lontheterminalset L$
  $FindanMSTT_LonG_L$
  $T \leftarrow \emptyset$
  for each*edge* $= (u, v)E(TL)$ **do**
    $FindashortestpathPfromutovonG.$
    if $PcontainslessthantwoverticesinT$ **then**
      $AddPtoT$
    else
      $Letpiandpjbethefirstandthelastverticesalreadyin T$
      $Addsubpathsfromutopiandfrompjtovto T$
  return $OutputT$

D. Geliştirmiş olduğumuz algoritma

1) Tanımlar:

Nokta: CBS sistemlerinde bulunan bir veri tipidir. İki boyutlu düzlem üzerinde x ve y kordinatları bulunan harita üzerinde tanımlanabilen lokasyon bilgisidir.

Çizgi: CBS sistemlerinde bulunan bir veri tipidir. Birbirleri ile sıralı şekilde birleştirilmiş olan noktalar grubudur. En az iki noktanın birleşimi ile oluşmaktadır.

Alan: CBS sistemlerinde bulunan bir veri tipidir. Harita üzerinde belirli bir alanı ifade etmektedir. Örnek olarak; mahalle sınırları,ilçe sınırları veya il sınırları verilebilir.

Kuş bakışı uzunluk: Harita üzerinde nokta veya çizginin diğer nokta veya çizgiye olan uzaklığıdır. Buradaki uzunluk yeryüzündeki engellerden etkilenmeyen havadan uzunluğudur.

Yönlendirme: Harita verisi üzerinde iki nokta arasındaki yolun cadde, sokak, bulvar veya meydan(CSBM) düzeyinde bulunması işlemidir. Navigasyon cihazlarının yaptığı işlemdir. İki nokta arasındaki ulaşabileceğiniz yolun tanımlanması işlemidir.

2) Algoritmanın Tanımı: MST problemi Steiner Ağaç Probleminin özelleşmiş halidir. SAP probleminin çözümü polinom zamanda mümkün değildir. Bu yüzden Prim'in MST sezgisel algoritması yakınsama algoritması olarak kullanılmıştır. Problemin çözümü için geliştirmiş olduğumuz algoritmanın temelini de Prim'in MST sezgisel algortiması oluşturmaktadır.

Hesaplama yapmak istediğimiz çizge üzerinde yaklaşık olarak bir buçuk milyon köşe bulunmaktadır. Hizmet vermek istediğimiz nokta sayısı ise 523629 binadan oluşmaktadır. Hizmet verilecek binaları yönlendirme işlemine tabi tutabilmek için noktaların bağlı bulunduğu cadde, sokak, bulvar veya meydanların(CSBM) bulunması gerekmektedir. Bu indirgeme işlemi yapıldıktan sonra 32000 terminal köşesi elde etmiş bulunuyoruz.

Ayrıca sıralı her iki nokta arasındaki kenarın da belirli bir uzunluğu bulunmaktadır. Yani üzerinde çalışmış olduğumuz çizge yönsüz ağırlıklı bir çizgedir.

Algortimamız en fazla terminal düğüm sayısı kadar çalışması beklenmektedir. Fakat herhangi bir nokta çözüm çizgemize eklendiği zaman üzerinde bulunduğu çizgideki diğer noktalar da çözüm çizgemize otomatik olarak eklenecek şekilde geliştirilmiştir. Bunun bizim algortimamızın çalışma zamanını en az yarıya düşürecektir. Çünkü bir çizgi iki veya daha fazla noktanın birleşiminden oluşur. Böylelikle algortimamızı en fazla terminal sayısının yarısı kadar çalışacak şekle getirmiş bulunmaktayız.

Prim'in MST sezgisel algortimasının gereği olarak her terminal köşeden diğer köşelere olan uzaklığın hesaplanarak mesafesi en kısa olan köşenin çözüm çizgemize eklenmesi gerekmektedir. Fakat köşe ve terminal köşe sayımızın çok fazla olmasından dolayı burada iyileştirme yapılmıştır.

Herhangi bir terminal köşenin çözüm çizgemize olan uzaklığını hesaplamak için CBS datası üzerinde rota bulma işlemi yaparak harita üzerinde fiziksel olarak takip edebileceğiniz bir yol bulmanız gerekmektedir. Burada Dijsktra'nın en kısa yol algoritması işletilmektedir.

Normal Prim'in MST sezgisel algortimasınının işletilmesi sırasın da çözüm çizgemizde bulunan her bir düğümün çözüm çizgemize eklenmemiş olan düğümlere olan uzaklığının yönlendirme yapılarak hesaplanıp mesafesi en kısa olan düğümün çözüm kümesine eklenmesi gerekmektedir. Bunun yerine CBS datasının bize sağlamış olduğu bazı fonksiyonlardan yararlanıyoruz.

İlk adım olarak çözüm çizgemizi birleşik bir çizgi olarak ele alıyoruz ve bu çizgiye en yakın kuş bakışı uzunluğu en az olan on adet çözüm kümesine eklenmemiş terminal köşeyi buluyoruz. Normalde beklentimiz yönlendirme yapıldığı zaman eklenecek olan köşenin bu on noktanın içerisinde olması gerekir.

İkinci adım olarak seçmiş olduğumuz on köşeye kuş bakışı uzaklığı en yakın olan çözüm çizgemize eklenmiş beş adet köşeyi buluyoruz.

Üçüncü adımda seçmiş olduğumuz on adet eklenmemiş köşe ve beş adet eklenmiş köşe için teker teker Djisktra'nın en kısa yol algoritmasını kullanarak rota hesaplaması yapıyoruz. Elli itarasyondan sonra en kısa mesafeye sahip sahip çözüm çizgesine en yakın eklenmemiş terminal köşe bulunarak çözüm çizgemize ekleniyor.

Rota çizilmesi sırasında bir noktadan diğer noktaya ulaşabilmek için çok fazla sayıda yol olabilir. Rotanın hesaplanması çok fazla zaman almaktadır. Sonuç olarak algoritmanızın çalışma zamanı ne kadar kısa ise gerçek hayatta kullanılma olasılığı o kadar artmaktadır. Burada rota hesaplaması yaparken hesaplama yapmak istediğimiz her iki noktayı kapsıyacak şekilde mahalle alanlarını birleştirerek yeni bir rota hesaplama kümesi oluşturuyoruz. Rota hesaplaması yeni oluşturulan harita verisi üzerinde yapılarak çalışma zamanın da ciddi bir azalma sağlanmıştır. Eğer yeni oluşturmuş olduğumuz küme içerisinde herhangi bir rota bulunamaz ise tüm Türkiye verisi üzerinde rota hesaplaması yapılır. Böylelikle rota hesaplaması garanti altına alınır.

Tüm terminal düğümlerinizin çözüm çizgesine eklendiği zaman çalışma sona erer. Bu anda hizmet vermek istediğiniz tüm köşeler çözüm çizgenize eklenmiş olur.

3) Sözdekod:

Algorithm 3 SAP(Her bir iterasyon için)

- Çözüm çizgesine eklenmemiş en yakın 10 köşeyi bul ve teker teker dön
- Herbir eklenmemiş köşe için en yakın çözüm çizgesine eklenmiş 5 köşeyi bul ve dön
- Eklenmemiş ve eklenmiş köşeler arasındaki Djisktra' nın en kısa yol algoritmasını kullanarak rota hesabını yap
- Eğer en az ağırlığa sahipse sakla
- 50 iterasyon sonucunda saklanan köşelerden en az ağırlığa sahip olan noktayı çözüm çizgesine ekle

V. Geliştirmiş Olduğumuz Yazilim

A. Kullanılan Teknolojiler

1) PostGIS: PostGIS [7] coğrafik nesneleri destekleyeyen PostgreSQL ilişkisel veritabanı yazılımının özelleşmiş hali olan open source bir eklentidir. Lokasyon ve coğrafik nesneler üzerinde SQL dilini kullanarak işlemler yapmanız için tasarlanmıştır.

2) Open Street Map: OpenStreetMap (OSM) [8] dünya üzerinde serbest düzenlenebilir bir harita oluşturmak için kurulan bir işbirliği projesidir.

1.6 milyon kayıtlı kullanıcı tarafından anket, GPS verileri, hava fotğrafçılığı ve diğer serbest kaynaklar kullanılarak datalar

toplanmaktadır.Daha sonra bu kalabalık data kümesi Open Database Lisası altında kullanılabilir hale getirilir.

3) Open Source Routing Machine(pgRouting): pgRouting [9], PostGIS/PostgreSQL coğrafi veritabanı üzerinde coğrafi yönlendirme yapmayı sağlamaktadır. Üzerinde tanımlı olan birçok algoritma bulunmaktadır. Bu algoritmalardan herhangi birini kullanarak iki nokta arasındaki rotayı hesaplayabilirsiniz.

4) Open Layers: OpenLayers [10] harita datalarının web tarayıcısında görüntülenmesine yarayan bir open source javascript kütüphanesidir.

B. Program Çalışma Ayrıntıları

Geliştirmiş olduğumuz yazılım kullanıcılara hesaplama sonuçlarını gösteren haritaların bulunduğu kullanıcı arayüzü ve algoritmayı işleten hesaplama motoru olmak üzere iki ana kısımdan oluşmaktadır.

Kullanıcılar için hazırlanan arayüzler django framework(python programlama dili) kullanılarak yazılmıştır. Haritalar içinse openlayer javascript kütüphanesi kullanılmıştır.

Kullanıcı arayüzlerinin temelde iki adet görevi bulunmaktadır. Bunlardan ilki kullanıcıların hizmet götürülecek binaları harita üzerinde kaydebilmelerini sağlamaktadır. Harita üzerinden veri girişi imkanı kullanıcıların verilere kolay bir şekilde erişmesini ve yeni lokasyonlar ekleyebilmelerini sağlamaktadır. İkinci olarak ise kullanıcıların oluşturulan ağları görsel olarak görebilmeleri sağlamaktır.

Hesaplama Motoru ise girişi yapılan veriler üzerinde iki temel işlevi yerine getirmektedir. Bunlardan ilki bir ağın sıfırdan oluşturulması işlemidir. Diğeri ise oluşturulmuş bir ağ üzerine yeni bir düğümün eklenmesi işlemidir.

Geliştirmiş olduğumuz yazılım programlama dili olarak python kullanmaktadır. Data saklama yöntemi olarak ise CBS datalarını işlemek için özelleştirilmiş PostgreSQL veritabanı yönetim sisteminin bir eklentisi olan PostGIS kullanılmaktadır.

Ayrıca harita üzerinde yönlendirme yapabilmek için yine PostgreSQL veritabanı yönetim sisteminin bir eklentisi olan pgRouting kullanılmaktadır.

Hesaplama motoru hesaplama işlemini iki adımda gerçekleştirmektedir.

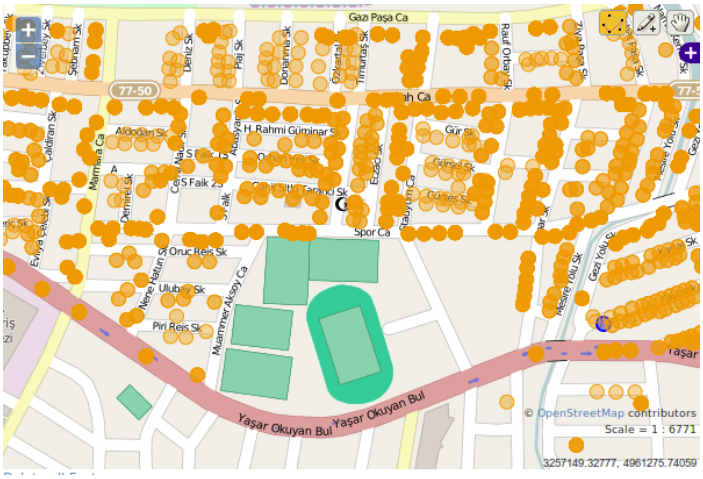
Verilerin yönlendirme yapabilecek şekilde dönüştürülmesi

Harita dataları üzerinde yönlendirme yapılmak isteniyorsa bu işlem cadde, sokak, meydan veya bulvarlar(CSBM) üzerinden yapılmak zorundadır. Hizmet verilecek noktalar kordinatlar halinde daha önce girişi yapılmış verilerdir. Hizmet verilecek noktaların bağlı bulundukları CSBM bulunur. CSBM'yi oluşturan noktalar yazılım tarafından kaydedilir. Böylelikle aynı csbm üzerinde bulunan onlarca hizmet noktasına hizmet götürebilmek için hesap yapacağımız nokta sayısını birkaç noktaya indirgemiş oluyoruz. Bu bize hesaplama sırasında ciddi manada zaman kazancı sağlamaktadır.

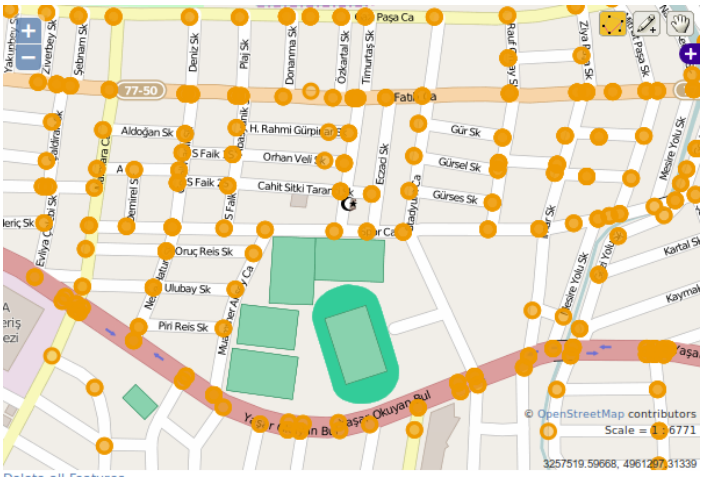
Algoritmanın işletilmesi

Kendi oluşturmuş olduğumuz algoritma dönüştürülmüş veri üzerinde işletilmektedir. Üretilen sonuçlar harita üzerinde gösterilebilecek bir formata getirilmektedir.

C. Ekran Görüntüleri



Şekil 9: Hizmet götürülmek istenen her bir düğümün lokasyonunu gösterimi



Şekil 10: Hizmet götürülecek noktalar üzerinde yönlendirme yapabilmemiz için gerekli olan formata getirilmiş halini gösterir harita. Bu harita verimizin CSBM düzeyine indirgenmiş halidir..

Şekil 11: Yazılımın çıktısını gösteren harita. Burada sadeleştirilmiş olan CSBM verisini oluşturan tüm noktaların algoritmamız tarafından birleştirilmiş halini göstermektedir.

VI. Sonuç

Sonuç olarak yapmış olduğumuz çalışma kapsamında matematikte çok sık kullanılan problemlerden biri olan Steiner ağaç problemi gerçek hayatta kullanılabilir hale getirilmiştir. Yapmış olduğumuz çalışma sayesinde firmaların maliyetlerini optimize etme şansı doğmuştur. Ayrıca toplumsal açıdan bakıldığı zaman: kazı çalışması yapılması çevre ve ses kirliliğine neden olmaktadır. Bu çalışma sayesinde dolaylı yoldan daha az kazı çalışması yaparak kısmi olarak çevre ve ses kirliliğinin önüne geçmiş oluruz.

Kaynakça

[1] Hans Jürgen Prömel and Angelika Steger.“The Steiner Tree Problem: A Tour through Graphs, Algorithms, and Complexity (Advanced Lectures in Mathematics)“,2002

[2] Hwang, F. “On Steiner Minimal Trees with Rectilinear Distance“ SIAM J. Appl. Math. SIAM Journal on Applied Mathematics, 104-114.

[3] F. Benjamin Zhan. “Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures“. Journal of Geographic Information and Decision Analysis, vol.1, no.1, pp. 70-82, 1997

[4] Postgresql Dökümantasyon. Erişim Ekim 15, 2015, “https://wiki.postgresql.org/wiki“

[5] Python 3.5.0 documentation. Erişim Ekim 15, 2015, “https://docs.python.org/3/ “

[6] Bilgi Teknolojileri ve İletişim Kurumu - Pazar Verileri. Erişim Haziran 14, 2014, “http://www.btk.gov.tr/tr-TR/Sayfalar/Pazar-Verileri “

[7] PostGIS 2.2.1dev Dökümantasyon. Erişim Ekim 15, 2015, “http://postgis.net/docs/manual-2.2/“

[8] OpenStreetMap Hakkında. Erişim Ekim 15, 2015, “https://www.openstreetmap.org/about“

[9] OpenStreetMap Dökümantasyon. Erişim Ekim 15, 2015, “http://pgrouting.org/documentation.html“

[10] Openlayers Dökümantasyon. Erişim Ekim 15, 2015, “ http://openlayers.org/en/v3.10.1/doc/ “