Kapak Konusu: Geometrik Kombinatorik

Kurnaz Bir Algoritma Örneği:

En Kısa Kapsayıcı Ağaç

Selin Enüst Çalışkan* / *selinenust@hotmail.com*

Bu yazımızda kombinatoryal optimizasyon alanında birkaç problemden bahsetmek istiyoruz. Kombinatoryal optimizasyon, sonlu yapılar üzerinde optimizasyon problemlerini çözerken kombinatorik ve algoritmik tekniklerin kullanıldığı, konusu ve problemleri itibariyle hem matematik, hem bilgisayar mühendisliği hem de endüstri mühendisliğinin ilgi alanına giren oldukça hareketli bir alandır.

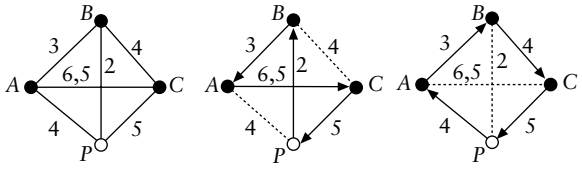
Bilinen en ünlü optimizasyon problemlerinden biri MD-2003-III, sayfa 37-40'te sözettiğimiz gezgin satıcı problemidir. Yazıda konu edeceğimiz problem gezgin satıcı probleminin bir başka vesiyonudur.

Varsayalım ki bir petrol şirketimiz var ve bu şirketin Nijerya kıyılarında 25 tane petrol kuyusu var. Her kuyudan tankerlere giden petrol miktarını ayarlamak mümkün ve bu ayarların periyodik olarak kontrol edilip değiştirilmesi gerekiyor. Bunun için belli bir noktadan kalkan bir helikopter kullanılıyor. Helikopter bütün kuyuları teker teker ziyaret edip yine aynı noktaya geri dönüyor. Tabii helikopterin uçuş maliyeti yüksek olduğu için bu işin zamandan ve yakıttan en çok tasarruf edilecek şekilde yapılması gerekiyor.

Bu problemi biraz matematikselleştirelim. Her kuyuyu düzlemde bir noktayla gösterelim. Helikopterin kalkış noktası da ayrı bir *P* noktasıyla gösterilmiş olsun. Amacımız bu *P* noktasından başlayıp bütün kuyuları (yani noktaları) dolaşan ve gene *P* noktasına geri dönen en kısa yolu bulmak.

Bütün yolların uzunluğunu hesaplayıp en kısa yolu seçmek elbette bir çözümdür. Ancak *n* tane kuyumuz varsa, bu, *n*!/2 tane yolun uzunluğunu hesaplamak demektir ve 25 gibi küçük sayılabilecek bir *n* için bile bu 10 milyon yıl gibi bir zaman alır.

Eğer zaman açısından bir kısıtlamamız varsa, *P*'den yola çıkıp, ziyaret edilen her kuyudan sonra daha önce ziyaret edilmemiş en yakın kuyuya gidelim. Her adımda en kolay görünen tercihi yapan algoritmalara bilgisayar biliminde ***kurnaz algoritmalar*** denir[1]. Bu problemde kurnaz algoritmanın en iyi çözümü garanti etmediğini Haldun Sural'ın MD-2003-III, sayfa 37-40'teki yazısında görmüştük. Hatırlamak için aşağıdaki örneği ele alalım.



Platformlar ve aralarındaki mesafe

Açgözlü yol: *PBACP*, toplam 2+3+6,5+5 = 16,5

En kısa yol: *PABCP*, toplam 4+3+4+5 = 16

En kısa yol, *ABC* ya da *CBA* şeklinde bir sıralamayla kuyuları dolaşıp *P*'ye geri döner. Ama yukarda açıkladığımız kurnaz yöntemi kullanırsak, ilk önce *P*'ye en yakın nokta olan *B*'yi ziyaret etmemiz, sonra *B*'ye en yakın olan *A*'yı ziyaret edip, oradan en son olarak ziyaret edilmemiş tek nokta olan *C*'ye gidip *P*'ye geri dönmemiz gerekir. Bu durumda yolumuzu uzatmış oluruz. Görüldüğü gibi kurnaz algoritma bu örnekte pek işe yaramadı.

Her durumda problemin makul bir zamanda çözümünü veren bir algoritma bilinmiyor. Burada "makul zaman"la, çözümü bulmak için gereken sürenin problemdeki nokta sayısı olan *n*'ye oranla çok hızlı büyümediği bir algoritmayı kastediyoruz. Konumuz kapsamında olmadığından burada açıklamayacağımız nedenlerden ötürü, yaygın inanç, bu şekilde bir algoritma olmadığı yönündedir.

Uygun Fiyata Kablo Döşenir! Diyelim bir şirketimiz var ve bu şirketin belli sayıda ofisi var. Bütün ofisleri birbirine bağlayan bir iletişim ağı kurmak, bunun için de ofisler arasına kablo çekmek istiyoruz. Tabii her bağlantının bize belli bir maliyeti olacak. Amacımız bütün ofislerin birbirine dolaylı da olsa bağlantılı olduğu bir iletişim ağını olabilecek en düşük maliyetle kurmak.

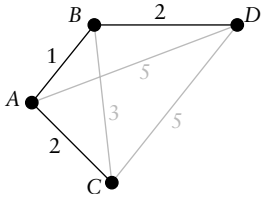
* Georgia Institute of Technology Matematik Bölümü doktora öğrencisi.

1 İngilizcesi "greedy algorithms".

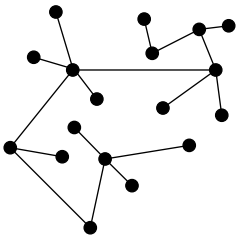
Bunu bir örnekle açıklayalım. Şirketimizde dört tane ofis olsun. Bu ofisleri A, B, C ve D harfleriyle gösterelim. A ofisiylele B ofisi arasında bir bağlantı kurmanın bedelini de M_{AB} ile gösterelim. Maliyetler aşağıdaki gibi olsun.

$$M_{AB} = 1,\ M_{AC} = 2,\ M_{AD} = 5,$$
$$M_{BC} = 3,\ M_{BD} = 2,\ M_{CD} = 5.$$

Bu durumda aşağıdaki diyagram bize maliyeti 5 birim olan bir iletişim ağı verir ve bu, bütün ofislerin birbiriyle bağlantılı olduğu en düşük maliyetli ağdır.



Eğer ofisleri birer nokta, herhangi iki ofis arasındaki bağlantıyı da bir kenar olarak düşünürsek, o zaman kurduğumuz iletişim ağı aslında bir çizgeye karşılık gelir. Maliyetimizi düşük tutmak istediğimiz için çözümümüz olan çizge bir döngü içermemelidir. Yani biz herhangi iki noktası arasında bir yol bulunan ve döngü içermeyen bir çizge kurmak istiyoruz. Bu tür çizgelere ***ağaç*** adı verilir. Yanda 18 noktadan oluşan bir ağaç örneği görüyoruz.



Aşağıdaki gri karede n noktası ve $n-1$ kenarı olan ve döngü içermeyen her çizgenin bir ağaç olduğunu kanıtladık. Demek ki n tane ofisimiz varsa, $n-1$ tane bağlantısı olan ve döngü içermeyen bir iletişim ağı kurmak istiyoruz.

Hangi Çizgeler Bir Ağaçtır?

Eğer bir ağaçta n tane nokta varsa n–1 tane de kenar vardır. Ağaçların bu özelliğini MD-2003-III, sayfa 19'da kanıtlamıştık. Ayrıca,

Teorem. *n noktası ve $n-1$ kenarı olan ve döngü içermeyen her çizge bir ağaçtır.*

Kanıt: n üzerine tümevarımla kanıtlayacağız. Çizgede tek bir kenara bağlı olan bir uç noktası vardır mutlaka. Bu uç noktayı ve o kenarı atalım. Geriye $n-1$ noktası ve $n-2$ kenarı olan ve döngü içermeyen bir çizge kalır, tümevarım varsayımına göre bu çizge bir ağaçtır. Şimdi attığımız noktayı ve kenarı tekrar yerine koyarsak, gene bir ağaç elde ederiz. □

Bir çizgedeki her noktadan geçen ve kenarlarının toplam maliyeti en ucuz olan ağaçlara ***en kısa kapsayıcı ağaç*** denir. Bu n–1 bağlantıyı nasıl seçmeliyiz ki elde ettiğimiz ağın maliyeti en düşük olsun?

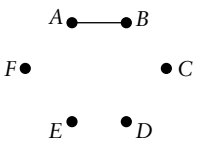
Bu sefer kurnaz algoritma işe yarıyor: Fazla düşünmeden, her aşamada o zamana dek kurulmamış bağlantılar arasından maliyeti en düşük olan bağlantıyı kuralım, tek koşulumuz döngünün olmaması... Bunu böyle devam ettiğimizde en kısa kapsayıcı ağacı buluruz.

Kurnaz algoritmayı bir örnek üzerinde gösterelim. Ofislere A, B, C, D, E, F diyelim ve ofisler arasına kablo döşemenin maliyetleri şöyle olsun.

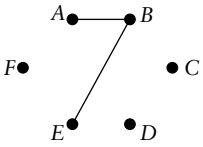
$M_{AB} = 1$,	$M_{AC} = 2$,	$M_{AD} = 2$,
$M_{AE} = 1$,	$M_{AF} = 4$,	$M_{BC} = 5$,
$M_{BD} = 2$,	$M_{BE} = 1$,	$M_{BF} = 4$,
$M_{CD} = 4$,	$M_{CE} = 3$,	$M_{CF} = 2$,
$M_{DE} = 5$,	$M_{DF} = 1$,	$M_{EF} = 2$.

Ofis sayısı 6. Demek ki bağlantı sayısı 5 olmalı.

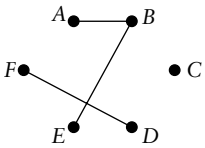
Önce en ucuz maliyetli bağlantıyı kuracağız. Burada en ucuz bağlantının maliyeti 1 birim ve bu maliyete sahip birden fazla bağlantı var, biz A ile B arasındaki bağlantıyı seçelim.



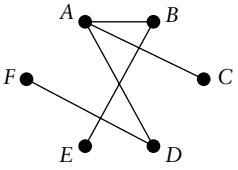
Şimdi bağlantı sayımız 5 olana kadar döngü oluşturmayacak şekilde en ucuz bağlantıları seçmeye devam edelim. Hâlâ maliyeti 1 birim olan üç bağlantımız var: B ile E arasınbaki bağlantı, A ile E arasındaki bağlantı ve D ile F arasındaki bağlantı. Rastgele BE kenarını seçelim.

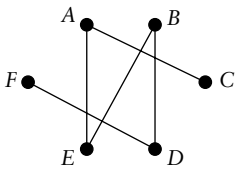


Dikkat edersek artık bir sonraki adımda AE kenarını seçemeyiz çünkü AB, BE ve AE kenarları bir döngü oluştururlar. O yüzden üçüncü adımda DF kenarını eklemeliyiz.



1 birim maliyetli bağlantılarımız bittigi için 2 birim maliyetli bağlantıları ekleyerek devam ediyoruz. Dördüncü adımda AC ve beşinci adımda AD kenarlarını seçersek bir döngü oluşturmamış oluruz. 5 tane bağlantı seçmiş olduğumuza göre artık istediğimiz iletişim ağını kurmuş olmamız gerekir. Yandaki şekilde kurmuş olduğumuz iletişim ağını görüyorsunuz.





En kısa kapsayıcı ağaç tek bir tane olmayabilir. Örneğin yandaki ağaç, yukarıdaki örnek için bir başka en kısa kapsayıcı ağaçtır.

Matematik! Şimdi, kurnaz algoritmanın gerçekten işe yaradığını gösterelim.

Kurnaz algoritmanın bize bütün ofislerin birbiriyle bağlantılı olduğu ve olabilecek en düşük maliyette bir ağ verdiğini kanıtlamalıyız.

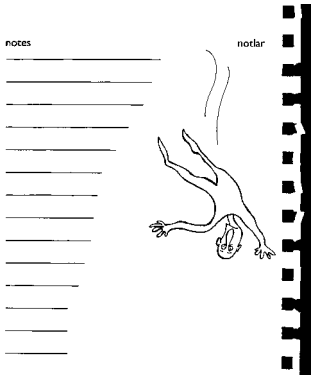
Daha önce de belirtmiş olduğumuz gibi n tane ofisin birbiriyle bağlantılı olduğu ve döngü içermeyen bir iletişim ağı, çizge olarak düşündüğümüzde bir ağaca karşılık geliyor ve $n - 1$ kenarı olup hiç döngü içermeyen her çizge de aslında bir ağaç. Bizim algoritmamız da kenar sayısı $n - 1$ oluncaya kadar her adımda döngü oluşturmayacak şekilde bir kenar ekleyerek ilerlediği için, algoritmanın sonunda elde ettiğimiz çizge elbette bir ağaç olacaktır.

Şimdi bunun en kısa kapsayıcı ağaç olduğunu kanıtlayalım. Algoritmanın her adımında elde ettiğimiz çizgenin en kısa kapsayıcı ağacın bir altkümesi olduğunu göstereceğiz. En son adımda elde ettiğimiz ağacın kendisi bir kapsayıcı ağaç olduğu için, böylece en son adımda bir en kısa kapsayıcı ağaç elde etmiş olduğumuzu göstermiş oluruz.

Algoritmanın herhangi bir adımında elde ettiğimiz çizgenin bir en kısa kapsayıcı ağacın altkümesi olduğunu göstermek için tümevarım yöntemini kullanacağız.

Algoritmanın sıfırıncı adımındaysak, yani daha işin en başındaysak ve "boşağacımız" varsa herhangi bir en kısa kapsayıcı ağaç işimizi görür.

Şimdi $k \geq 1$ olsun ve algoritmanın k-inci adımında elde ettiğimiz çizgenin bir en kısa kapsayıcı ağacın altkümesi olduğunu kanıtlayalım. Algoritmanın k-inci adımında elde ettiğimiz çizgeye S diyelim. Tümevarım varsayımımızdan dolayı, bir önceki $(k-1)$-inci adımda elde etmiş olduğumuz R çizgesi bir T en kısa kapsayıcı ağacının altkümesidir. Bu arada R'nin boşçizge olabileceğini de belirtelim. S çizgesi, algoritmaya göre, R'ye döngü oluşturmayacak şekilde kalan kenarlar içinde maliyeti en düşük kenarın eklenmesiyle elde edilir. Bu kenara e diyelim. Eğer e kenarı T'deyse, S zaten T'nin bir altkümesidir ve kanıt biter. Eger e, T'de değilse, T'de biraz değişiklik yaparak S'yi içeren bir başka en kısa kapsayıcı ağacı şu şekilde inşa edebiliriz: Önce e'yi T ağacına ekleyelim. Bu yeni kenar T'nin kenar sayısını bir artırır ve böylece T'de bir döngü oluşmasına neden olur. Bu döngüdeki kenarların hepsi R'de olamaz, çünkü öyle olsa bu döngü aynı zamanda S'de de olurdu ama S döngü içermeyecek şekilde inşa edilmişti. Dikkat edilirse, T'ye e'yi ekleyerek elde edilen döngüde R'de olmayan kenarların maliyeti e'nin maliyetinden büyükeşittir, çünkü e kenarı, R'de olmayan ve R'ye eklendiğinde döngü oluşturmayan kenarlar arasında maliyeti en düşük olan kenarlardan biridir. (Aslında biraz düşünürsek görürüz ki bu döngüde R'de olmayan bütün kenarların maliyeti birbirine eşit olmalıdır, yoksa daha yüksek maliyetli bir kenarı atıp T'den daha düşük maliyetli bir kapsayıcı ağaç bulabilirdik.) Bu durumda o kenarlardan birini T'den atıp yerine e'yi eklersek yeni bir en kısa kapsayıcı ağaç elde ederiz. Bu yeni ağaç S'yi içerir. Demek ki S'yi içeren bir en kısa kapsayıcı ağaç vardır. Kanıtımız bitmiştir. ♣



Selin Enüst Çalışkan

1978 Trabzon doğumluyum. Babamın mesleği gereği bütün çocukluğum Türkiye'nin değişik yerlerinde geçti. İlkokulu dört ayrı okulda okudum. Ortaokulu Kars'ın Akyaka Lisesi'nde, liseyi Kıbrıs Girne Anafartalar Lisesi'nde bitirdim. Üniversite hayatıma İzmir Dokuz Eylül Matematik bölümünde başladım ve ikinci senenin sonunda Boğaziçi Üniversitesi'nin Matematik bölümüne transfer oldum. 2000'de Boğaziçi Matematik bölümünden mezun oldum ve ardından aynı bölümde yüksek lisansımı tamamladım. Şu anda Georgia Institute of Technology'de doktora çalışmalarımı sürdürmekteyim.

Boş vakitlerimde film seyretmek, kitap okumak, ve jogging yapmak en büyük hobilerim. Özellikle psikoloji ve felsefe okumaktan hoşlanırım. ♣