Problem Seti 2

Okumalar: 5.1-5.3 kısımları ve Bölüm 6, 7, ve 9.

Hem egzersizler hem de problemler çözülecek, ama sadece problemler teslim edilecektir. Egzersizler ders materyalini hazmettirmek amacıyla hazırlanmıştır. Her ne kadar egzersiz çözümlerini teslim etmeyecek olsanız da egzersizdeki konulardan sorumlu olacaksınız.

Sık sık bir problem için " bir algoritma bulun " isteğiyle karşılaşacaksınız. Bu konudaki yanıtınız kısa bir makale şeklinde olmalıdır. Konu paragrafı, çözdüğünüz problem ve sonuçlarınızı özetleyecek şekilde düzenlenmelidir. Makalenizin ana yapısında aşağıdaki bilgiler verilmelidir:

1. Algoritmanın İngilizce açıklaması ve eğer faydalı olacaksa sözde kodu..
2. Algoritmanızın nasıl çalıştığını gösteren en az bir işlenmiş örnek veya şekil.
3. Algoritmanın doğruluğunun kanıtı (veya göstergesi).
4. Algoritmanın koşma zamanının çözümlemesi.

Amacınız iletişim kurmaktır. Tam not sadece iyi açıklanan doğru yanıtlara verilecektir. Ya da net olmayan açıklamalar daha düşük notlandırılacaktır.

Egzersiz 2-1. Kitaptaki 5.2-4 nolu egzersiz (98. Sayfa)

Egzersiz 2-2. Kitaptaki 6.5-7 nolu egzersiz (142. Sayfa)

Egzersiz 2-3. Kitaptaki 7.2-1 nolu egzersiz (153. Sayfa)

Egzersiz 2-4. Kitaptaki 9.3-8 nolu egzersiz (193. Sayfa)

Egzersiz 2-5. Kitaptaki 9-1 nolu egzersiz (194. Sayfa)

Problem 2-1. Bu (yaklaşık) sıralanmış mıdır?

Hogwarts ekolünün çocuk büyücüsü Harry Potter' ın başı bir kez daha derttedir. Profesör Snape Harry' i tutuklamış ve ona son 200 yılın eski ev ödevlerini sıralama işini vermiş. Harry bir büyücü olduğundan elini şöyle bir sallayıp "ordinatus sortitus" yani " düzenli sıra" demiş ve kağıtlar hızla kümelenmişler.

Ancak Profesör Snape Harry' nin büyüsünün kağıtları doğru olarak kümelediğini belirlemek istemektedir. Öte yandan n sayıda çok fazla kağıt olduğundan, doğru sırada olup olmadıklarını belirlemek, Ω(n) süresini alacaktır.

Bunun yerine Profesör Snape, kağıtların yaklaşık-sıralanmış olduğunu kontrol etmeye karar verir. Kağıtların %90' ının sıralandığını bilmek ister: kağıtların %10'u kümeden çıkarılırsa elde edilen liste sıralanmış olabilir mi?
Bu problemde Profesör Snape' e, n farklı elemanlı A girdi listesini işleyecek bir algoritma bularak yardım edeceğiz; algoritma aşağıdaki gibi davranacak:

- Eğer A listesi sıralıysa, algoritma her zaman **true/ doğru** çıktısını verecek.
- Eğer A listesi % 90 sıralı değilse, algoritma en az 2/3 olasılıkla **false/yanlış** çıktısını verecek.

(a) Profesör Snape önce şu algoritmayı düşünür:

k kere tekrarla:

1. Kağıtları bağımsız, tekdüze ve rastgele olarak seç ve (1,n) aralığını kullan. (Yani, 1 <i<n.)
2. A[i− 1] ve A[i] kağıtlarını karşılaştır. Eğer doğru sırada değillerse **yanlış** çıktısını ver ve dur.
3. A[i] ve A[i+ 1] kağıtlarını karşılaştır.. Eğer doğru sırada değillerse **yanlış** çıktısını ver ve dur.

Doğru çıktısını ver.

Bu algoritmanın, sıralama listesinin en az 2/3 olasılıkla yaklaşık doğru olduğunu keşfedebilmesi için k = Ω(n) gerektiğini gösterin. *İpucu*: Yaklaşık sıralı olmayan bir dizi bulun, ama algoritmanın yanlış çıktı vereceği az sayıda eleman olsun.

.

(b) Size içinde n sayıda top olan bir torba verilmiş olsun. İçinde en az %10 oranında mavi top olduğu ve kırmızı topların sayısının da % 90'ı geçmediği söyleniyor. Asimptotik olarak (yani n' nin büyük değerleri için) en az 2/3 olasılıkla mavi topu bulana dek kaç kez torbadan top çekmeniz gerekir? (topların torbadan çıkarıldıktan sonra yeniden torbaya konulduğunu varsayabilirsiniz.)

(c) Sıralanmamış bir listede bir “ikili arama” yaptığınızı varsayın:

```
BINARY-SEARCH( A, key, left, right ▷ Seach for key in A[left .. right]
if left = right
  then return left
  else mid ← $\lceil (left + right)/2 \rceil$
       if key < A[mid]
          then return BINARY-SEARCH(A,key,left,mid −1)
          else return BINARY-SEARCH(A,key,mid,right)
```

A' da (sıralı olmasa bile) key_1 için yapılan bir ikili aramanın i yuvasını geri dönüş olarak verdiğini farz edin. Benzer biçimde anahtar/ key_2 için ikili arama da j yuvasını versin. Aşağıdaki gerçeğin neden doğru olduğunu açıklayın:

Eğer i < j ise, $key_1 \le key_2$'dir. Bir şekil çizin. *İpucu*: Öncelikle bunun A listesinin sıralı olması durumunda neden açıkça doğru olduğunu düşünün.

(d) Profesör Snape listenin % 90 sıralı olduğunu belirlemek için rastgele bir algoritma öneriyor.. Algoritma bağımsız ve tekdüze bir tamsayıyı [1,n] kapalı aralığında rastgele seçmek için (Rastgele)/RANDOM (1,n) fonksiyonunu kullanıyor. Algoritma aşağıdaki gibidir:

```
IS-ALMOST-SORTED(A,n,k)    ▷   Determine if A[1 ..n] is almost sorted.
( Yaklaşık – Sıralı )          (A[1,n]'nin Yaklaşık-Sıralı olduğunu belirleyin)
for r ← 1 to k
    do i ← RANDOM(1,n)     ▷   Pick i uniformly and independently.
                               ( i' yi tekdüze ve bağımsız seçin)
       j ← BINARY-SEARCH(A,A[i],1,n)
       if i ≠ j
          then return false    (öyleyse cevabē"yanlış" olarak döndür)
return true                    ( doğru geri döndür )
```

Algoritmanın, k' nın yeterince büyük bir sabitse geçerli olacağını gösterin. Yani k uygun seçildiğinde algoritma liste doğru sıralanmışsa her zaman **doğru**‟yu çıktı olarak veriyor, ve liste %90 doğru sıralanmamışsa, 2/3 olasılıkla **yanlışı** çıktı olarak veriyor.

(e) Profesör Snape'in listenin $0 < \epsilon < 1$ arası bir epsilon değerinde $1 - \epsilon$ oranında sıralanmış olduğunu kontrol etmek istediğini düşünün. (Önceki örnekte $\epsilon = 0.10$ idi.

Büyük n değerleri için asimptotik olarak k değerini belirleyip algoritmanın doğru olduğunu gösterin. Toplam koşma süresi nedir?

Problem 2-2. Yaklaşık sıralanmış bir listeyi sıralamak.

Tutulduğu yerden dönerken Harry arkadaşı Hermione ile karşılaşır. Harry, sıralama büyüsünün tutmadığını Profesör keşfettiği için mutsuzdur. Kağıtları doğru sıralamak yerine, her kağıt doğru konumda *k* sayıda yuvaya konulmuştu. Hermione hemen araya yerleştirme sıralamasının bu sorunu kolayca çözeceğini söyler. Bu problemde Hermione'nun (çoğunlukla olduğu gibi) haklı olduğunu göstereceğiz. Önceki gibi, n farklı elemandan oluşan A[1..n] diziliminde..

(a) Önce bir evirme (inversion) tanımlayacağız. Eğer i< j ve A[i] > A[j] ise, (i,j) ikilisine A' nın evirmesi denir. { 1,2,…n} diziliminin hangi devşiriminde (permutation) en çok evirme vardır? Bunlardan kaç tane vardır?

(b) Eğer her kağıt başlangıçta doğru konumdaki k yuvada ise, araya yerleştirme sıralamasının koşma süresi O (nk) dır. *İpucu*: Önce ARAYA-YERLEŞTİRME (A)'nın koşma süresinin O (n+I) olduğunu gösterin; burada I, A' daki evirmelerin sayısıdır.

(c) Her kağıdın doğru konumdaki k yuvada olduğu bir listede sıralama için Ω(n lg k) sayıda karşılaştırma gerektiğini gösterin. *İpucu*: Karar ağacı tekniğini kullanın..

(d) Alt sınırı karşılayacak bir algoritma tasarlayın; yani her kağıdın doğru konumdaki k yuvada olduğu bir listede sıralamayı Θ (n lg k) sürede yapsın. *İpucu*: Kitabın 142. sayfasındaki 6.5-8 no.lu probleme bakın.

Problem 2-3. Ağırlıklı Ortanca

n sayıda farklı x_1, x_2,..., x_n elemanı var ve positif ağırlıkları w_1, w_2,..., w_n ; öyleyse $\sum_{i=1}^{n} w_i = 1$ teriminde ağırlıklı (alt) ortanca aşağıdaki durumları karşılayan x_k ' dır.

$$\sum_{x_i<x_k} w_i < \frac{1}{2}$$

ve

$$\sum_{x_i>x_k} w_i \leq \frac{1}{2} .$$

(a) x_1, x_2,..., x_n 'nin ortancasının, w_i =1/n (i =1,2,...,n için) ağırlıkları olan x_1, x_2,..., x_n'nin ağırlıklı ortancası olduğunu savunun..

(b) En kötü durum O (n lg n) süresinde ağırlıklı ortancanın sıralama kullanarak nasıl hesaplanacağını gösterin.

(c) Kitabın Bölüm 9.3'ündeki SELECT (SEÇ) gibi bir komutu bir doğrusal-zaman ortanca algoritması kullanarak ağırlıklı ortancayı Θ (n) en kötü sürede nasıl hesaplayacağınızı gösterin.

(d) Postane yeri problemi şöyle tanımlanır. Bize ilintili ağırlıkları w_1, w_2,..., w_n olan n sayıda p_1, p_2,..., p_n noktası veriliyor. Biz a ile b noktalarının arasındaki uzaklığın d (a,b) olarak gösterildiği durumda $\sum_{i=1}^{n}$ w_i d(p, p_i) toplamını en aza indirecek bir p noktası bulmak istiyoruz.(bu noktanın girişlerden bir olması şart değil).

Ağırlıklı ortancanın tek-boyutlu postane yeri problemi için en iyi çözüm olduğunu savunun; burada noktalar basit gerçek sayılar ve a ile b noktası arasındaki uzaklık d(a,b)=|a− b|.

(e) İki boyutlu postane yeri problemi için en iyi çözümü bulun.; burada noktalar (x,y) koordinat çiftleri ve a =(x_1, y_1) ve b =(x_2, y_2) noktaları arasındaki uzaklık d(a,b)=| x_1 − x_2| +|y_1− y_2| ile verilen Manhattan distance (uzaklığı).