Algoritmalara Giriş Eylül 7, 2005
Massachusetts Institute of Technology
6.046J/18.41OJ

Profesörler Erik D. Demaine ve Charles E. Leiserson
Çalışma notu 5

Problem Seti 1

Okumalar: Bölüm 1-4 (4.4 kısmı hariç)

Hem egzersizler hem de problemler çözülecek, ama sadece problemler teslim edilecekti. 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 ingilizce 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. Net olmayan açıklamalar daha düşük notlandırılacaktır.

Egzersiz 1-1. Kitaptaki 2.3-6 nolu egzersiz. (37. Sayfa)

Egzersiz 1-2. Kitaptaki 3.1-6 nolu egzersiz .(50. Sayfa)

Egzersiz 1-3. Kitaptaki 3.2-4 nolu egzersiz .(57. Sayfa)

Egzersiz 1-4. Kitaptaki 4.3-4 nolu egzersiz .(75. Sayfa)

Problem 1-1. Asemptotik Simgelem

Aşağıdaki her şık için asemptotik olarak negatif olmayan fonksiyonlar durumunda çözümün her zaman doğru (always true), hiçbir zaman doğru değil (never true) veya bazen doğru (some times true) olduğuna karar verin ve;

her zaman doğru ya da hiçbir zaman doğru değil durumlarında neden böyle olduğunu açıklayın.

Bazen doğru durumunda doğru olduğu durum için ve yanlış olduğu durum için birer örnek verin.

(a) $f(n) = O(f(n)^2)$

(b) $f(n)+g(n) = \Theta(\max(f(n),g(n)))$

(c) $f(n) + 0(f(n)) = \Theta(f(n))$

(d) $f(n) = \Omega(g(n))$ ve $f(n) = o(g(n))$ (küçük-o' ya dikkat edin)

(e) $f(n) \neq O(g(n))$ ve $g(n) \neq O(f(n))$

Problem 1-2. Yinelemeler

Aşağıdaki yinelemelerde *T(n)'* nin asemptotik üst ve alt sınırlarını verin. *T(n)'* nin $n \le 10$ için sabit olduğunu varsayın. Sınırlarınızı mümkün olduğunca sıkı tutun ve yanıtlarınızın gerekçesini verin.

(a) $T(n) = 2T(n/3) + n \lg n$

(b) $T(n) = 3T(n/5) + \lg^2 n$

(c) $T(n) = T(n/2) + 2^n$

(d) $T(n) = T(\sqrt{n}) + \Theta(\lg \lg n)$

(e) $T(n) = 10T(n/3) + 17n^{1.2}$

(f) $T(n) = 7T(n/2) + n^3$

(g) $T(n) = T(n/2 + \sqrt{n}) + \sqrt{6046}$

(h) $T(n) = T(n-2) + \lg n$

(i) $T(n) = T(n/5) + T(4n/5) + \Theta(n)$

(j) $T(n) = \sqrt{n}\, T(\sqrt{n}) + 100n$

Problem 1-3. Unimodal Search (Tek doruklu arama)

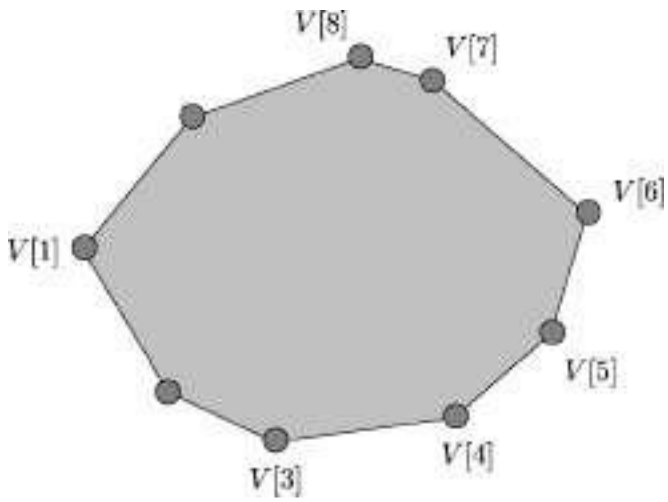
Bir *A[l.. n]* dizilimi, eğer artan bir diziyi takip eden azalan bir dizi varsa unimodal ya da **tek doruklu**dur; daha kesin bir anlatımla m Є {1, 2,..., *n*} dizininde

- $A[i] < A[i+1]$ tüm $1 \leq i < m$, ve
- $A[i] > A[i+1]$ tüm $m \leq i < n$.

Özellikle *A[m]* maksimum elemandır ve ($A[m-1]$ ve $A[m+1]$) elemanları tarafından çevrili tek yerel maksimumdur.

(a) Tek doruklu giriş dizilimi *A[l.. n] için O* (lg n) süresinde maksimum elemanı hesaplayacak algoritmayı bulun. Algoritmanızın doğruluğunu kanıtlayın ve koşma süresindeki sınırı kanıtlayın.

Bir poligonun tüm iç açıları 180°' den küçükse, ve hiçbir kenar birbirini kesmiyorsa o polygon conveks yani iç bükeydir. Şekil 1' de bir örnek verilmiştir. İçbükey bir poligonu biz *V[l.. n]* dizilimiyle gösteririz ve bu dizilimdeki her eleman poligonun bir köşesini (x,y) koordinat çiftleri olarak tanımlar. *V[1]* köşesi minimum x değeri olan köşe olarak verilmiştir ve *V[l.. n*] köşeleri şekilde olduğu gibi satin aksi istikametinde sıralanmıştır. Köşelerin tüm x koordinatlarının ve y koordinatlarının farklı olduklarını varsayabilirsiniz.



Şekil 1: V[l.. 9] diziliminin bir iç bükey poligon şeklinde gösterilmesine örnek. *V[l]*en küçük x koordinatı değeri olan köşedir ve V[l.. 9] saatin ters yönünde sıralanmıştır.

(b) Maksimum *x* koordinatlı köşeyi 0(lg *n)* süresinde bulacak algoritmayı belirleyin.

(c) Maksimum *y* koordinatlı köşeyi 0(lg *n)* süresinde bulacak algoritmayı belirleyin.