강의 1: 강의 소개 및 극댓값 찾기

강의 개요

- 운영 방침
- 과목 개요
- 극댓값 찾기 문제 - 1차원과 2차원의 경우

과목 개요

이 과목에서 다루는 내용:

- 대규모의 입력이 필요한 문제들을 해결하는 효율적인 방법 (예: 미국 고속도로망, 인간 게놈)
- 확장성
- 전형적인 자료 구조와 기초 알고리즘 (CLRS 교과서)
- 파이썬에서의 실제 구현
- 재미있는 연습문제

이 과목은 8개의 단원으로 나누어져 있고, 마지막 단원을 제외한 각 단원에는 동기 부여 문제와 연습문제들이 있습니다. 잠정적인 단원 주제와 동기 부여 문제는 아래와 같습니다:

1. 알고리즘적 사고: 극댓값 찾기
2. 정렬 & 트리: 이벤트 시뮬레이션
3. 해싱: 게놈 비교
4. 수: RSA 암호
5. 그래프: 루빅 큐브
6. 최단 경로: Caltech → MIT
7. 동적 프로그래밍: 이미지 압축
8. 심화 주제

극댓값 찾기

1차원의 경우

b ≥ a이고 b ≥ c 이면 2번 위치가 극댓값이다. i ≥ h 이면 9번 위치가 극댓값이다.

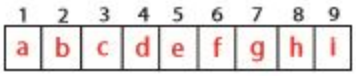
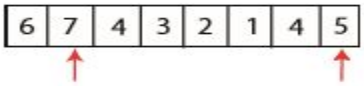


figure 1: a-i는 숫자

문제: 극댓값이 존재할 경우 그 값을 찾아라. (항상 존재하는가?)

간단한 알고리즘



왼쪽부터 시작하는 경우

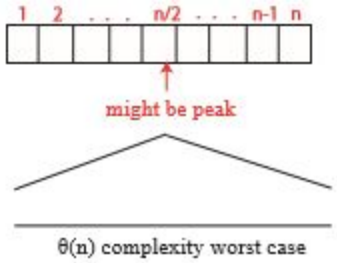
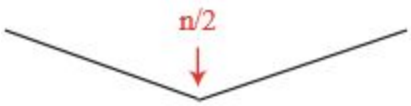


Figure 2: 평균적으로 n/2개의 원소 확인. 최악의 경우에는 n개의 원소 확인

중앙부터 시작하면 어떻게 될까? 아래 그림과 같은 상황에서는 n/2개의 원소를 확인해야 한다. 중앙부터 시작하여 인접한 원소 중 중앙 원소보다 값이 큰 쪽으로 방향을 선택한다면, n/2개보다 더 많이 확인해야 할 경우가 있을까?



더 효율적으로 찾을 수 있을까?

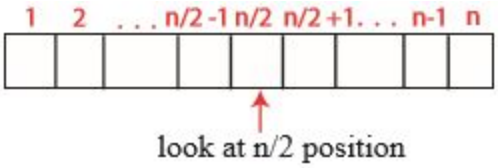


Figure 3: 분할 정복

- a[n/2] < a[n/2 - 1] 이면 왼쪽 절반인 1부터 n/2-1까지 보고 극댓값을 찾는다.
- 그게 아니고 a[n/2] < a[n/2 + 1] 이면 오른쪽 절반인 n/2 + 1부터 n까지 보고 극댓값을 찾는다.
- 그것도 아니면 n/2 위치가 극댓값이다: 왜일까? -> 양 옆의 값보다 크기 때문.

$$a[n/2] \geq a[n/2 - 1]$$

$$a[n/2] \geq a[n/2 + 1]$$

이 경우 복잡도는?

$$T(n) = T(n/2) + \underbrace{\Theta(1)}_{\text{to compare a[n/2] to neighbors}} = \Theta(1) + \ldots + \Theta(1) \ (\log_2(n) \ times) = \Theta(\log_2(n))$$

위와 같이 Θ(i)들을 합하려면, 모든 경우에 적용되는 상수를 찾아야 한다. n = 1000000인 경우, 파이썬에서 Θ(n) 알고리즘을 실행하는 데 13초가 걸린다. Θ(log n) 알고리즘은 0.001초밖에 걸리지 않는다.

알고리즘이 맞는지 증명하라. 1과 상수n의 차이는 커야하다. n과 2^n의 차이.

2차원의 경우

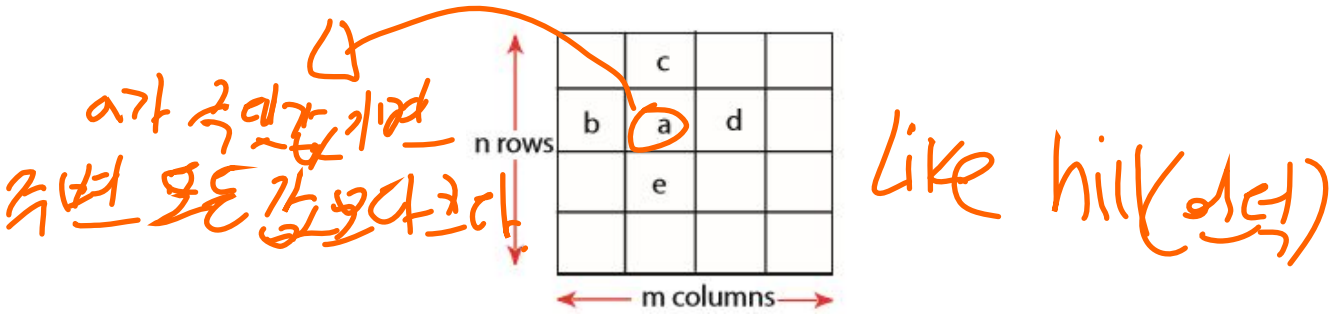


Figure 4: 탐욕 상승 알고리즘: Θ(nm) 복잡도, m = n이면 Θ(n 2) 알고리즘

$a \geq b$, $a \geq d$, $a \geq c$, $a \geq e$이면 a는 2차원 극댓값이다.

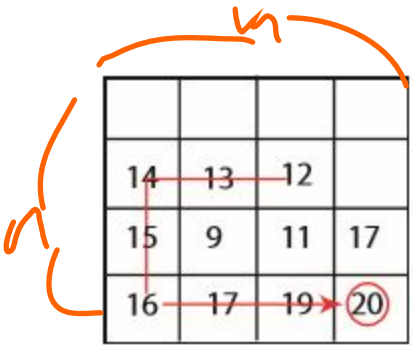
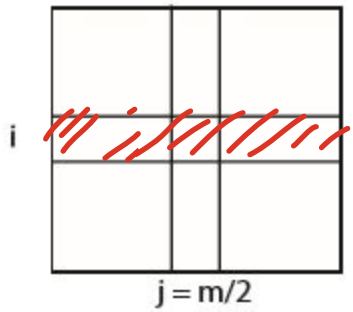


Figure 5: 동그라미 친 값이 극댓값이다

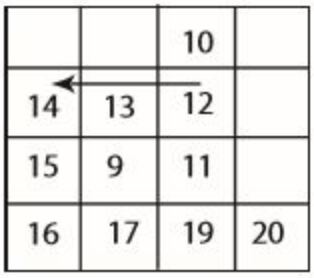
시도 # 1: 1차원 분할 정복을 2차원으로 확장



- 중앙 열 j = m/2을 선택한다.
- (i, j)에서 1차원 극댓값을 발견한다.
- (i, j)를 시작 지점으로 하여 i행에서 1차원 극댓값을 찾는다.

시도 #1 실패

문제: 2차원 극댓값이 i행에 존재하지 않을 수도 있다



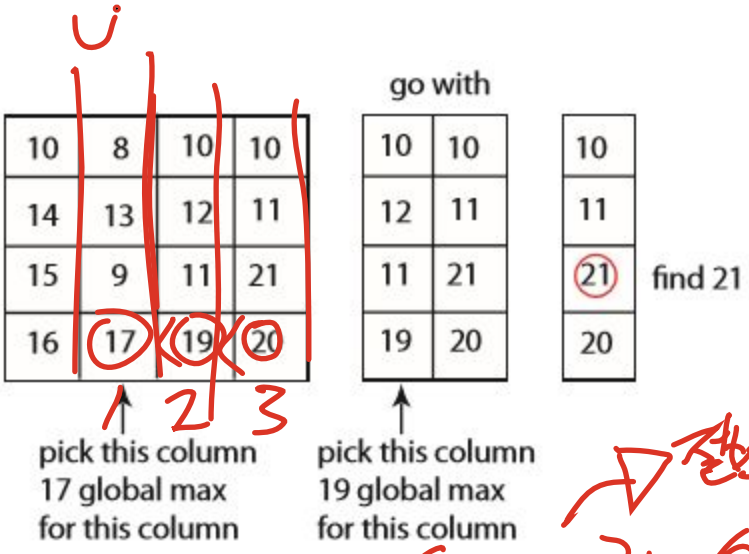
2차원 극댓값이 아닌 14를 찾는 걸로 끝남.

시도 # 2

- 중앙 열 j = m/2을 선택한다.
- (i, j)에서 j열의 최댓값을 찾는다.
- (i, j – 1), (i, j), (i, j + 1)를 비교한다.
- (i, j – 1) > (i, j)이면 왼쪽 열을 선택한다.
- 오른쪽도 똑같이 진행한다.
- 두 조건 모두 만족하지 않으면, (i, j)가 2차원 극댓값이다 ← 왜일까?
- 열 개수가 절반으로 줄어든 새로운 문제를 푼다.
- 열이 1개 남으면, 최댓값을 찾고 끝난다.

i=최댓값이 위치한 행

시도 #2 예시



절반의 열에서 해결하는 것과 같다

$T(n,m) = T(n,m/2) + \Theta(n)$

시도 #2 복잡도

n개의 행과 m개의 열이 있는 문제를 해결하기 위해 요구되는 일의 양을 T (n, m)라고 하면,

$$T(n,m) = T(n,m/2) + \Theta(n) \text{ (to find global maximum on a column — (n rows))}$$

$$T(n,m) = \underbrace{\Theta(n) + \ldots + \Theta(n)}_{\log m}$$

$$= \Theta(n \log m) = \Theta(n \log n) \text{ if } m = n$$

최댓값을 찾기 위해 n번만큼 해야함.

질문: 시도 #2에서 최댓값 대신에 1차원 극댓값을 사용하면 어떻게 될까? 제대로 작동할까?

결과 $T(n,m) = T(n,m/2) + \Theta(n) = T(n \log m)$

MIT OpenCourseWare

http://ocw.mit.edu

6.006 알고리즘 개론 Fall 2011