알고리즘 스터디
분할정복
Divide & Conquer

NEXTERS

분할 정복

- 주어진 문제들을 둘 이상으로 나누어 각 문제에 대한 답을 재귀적으로 계산함

- 일반적인 재귀 호출과 다른 점은 문제를 한 조각과 나머지 전체로 나누는 대신에 거의 같은 크기의 부분 문제로 나눈다.

분할 정복의 과정

- 문제를 더 작은 문제로 분할 하는 과정 (Divide)
- 각 문제에 대해 구한 답을 원래 문제에 대한 답으로 병합 하는 과정 (Merge)
- 더 이상 분할 하지 않고 답을 풀 수 있는 기본 케이스(Base case)

분할 정복 문제의 특성

- 문제를 둘 이상으로 자연스럽게 나눌 수 있는 방법이 있어야 함
- 부분 문제의 답을 조합해 원래 문제의 답을 계산 하는 효율적인 방법이 필요
- 위의 두 가지 특성을 통해 더 빠르게 문제를 풀 수 있다.

수열의 빠른 합 (1)

- n 까지의 수열의 합을 아래와 같이 정의
  fastSum() = $1 + 2 + \cdots + n$

- 두 부분 문제로 나눌 수 있음

fastSum() = $\left(1 + 2 + \cdots + \frac{n}{2}\right) + \left(\left(\frac{n}{2} + 1\right) + \cdots + n\right)$

→ 첫 번째 부분 문제는 fastSum(n/2) 로 표현 될 수 있지만,

두 번째 부분 문제는 그렇지 않다.

수열의 빠른 합 (2)

- fastSum() 을 포함하는 표현으로 바꾸기 위해 다음과 같이 바꾼다.

- $\left(\frac{n}{2}+1\right)+\cdots+n=\left(\frac{n}{2}+1\right)+\left(\frac{n}{2}+2\right)+\cdots+\left(\frac{n}{2}+\frac{n}{2}\right)$

$=\frac{n}{2}\times\frac{\mathrm{n}}{2}+\left(1+2+3+\cdots+\frac{\mathrm{n}}{2}\right)$

$=\frac{n}{2}\times\frac{n}{2}+fastSum(\frac{n}{2})$

수열의 빠른 합 (3)

- 도출된 식을 재귀 함수로 표현
  시간 복잡도 : $O(lgn)$

```
int fastSum(int n) {
    if (n == 1) return 1;
    if (n % 2 == 1) return fastSum(n - 1) + n;
    return 2 * fastSum(n / 2) + ((n*n) / 4);
}
```

행렬의 거듭 제곱 (1)

- n x n 크기의 행렬 A가 주어질 때 A의 거듭 제곱 m 의 시간 복잡도 : $O(n^3 m)$
- 문제를 반으로 자르기 $A^m = A^{\frac{m}{2}} \times A^{\frac{m}{2}}$
- 위의 식으로 반복하여 재귀적으로 문제를 해결한다.

행렬의 거듭 제곱 (2)

- 행렬의 거듭 제곱을 구하는 분할 정복 알고리즘 코드

```
class SquareMatrix;
SquareMatrix identity(int n);
SquareMatrix pow(const SquareMatrix& A, int m) {
    if (m == 0) return identity(A.size());
    if (m % 2 > 0) return pow(A, m - 1)*A;
    SquareMatrix half = pow(A, m / 2);
    return half * half;
}
```

병합 정렬과 퀵 정렬

- 병합정렬: 주어진 수열을 가운데에서 쪼개 비슷한 크기의 수열 두 개로 만든 뒤 이들을 재귀 호출을 이용 하여 정렬

- 퀵 정렬: 병합 과정이 필요 없도록 한쪽의 배열에 포함된 수가 다른 쪽 배열의 수보다 항상 작도록 배열을 분할

카라츠바의 빠른 곱셈 알고리즘

- 카라츠바 알고리즘은 긴 정수 둘을 곱하는 분할 정복 알고리즘이다. B진수 n자리 긴 정수 a, b를 곱한다고 하자. (n은 2의 자승으로 가정한다.)
- 이 때 두 숫자를 모두 절반(n/2 자리씩)으로 다음과 같이 자르자.
  - $a=a1\times Bn/2+a0$
  - $b=b1\times Bn/2+b0$
- 이 때 다음과 같이 두자.
  - $z0=a0\times b0$
  - $z2=a1\times b1$
  - $z1=(a0+a1)\times(b0+b1)-z0-z1=a0\times b1+a1\times b0$

->

  - $a\cdot b=(a1\cdot Bn/2+a0)\cdot(b1\cdot Bn/2+b0)=(a1\cdot b1)\cdot Bn+(a0\cdot b1+a1\cdot b0)\cdot Bn/2+(a0\cdot b0)$
  - $=z2\cdot Bn+z1\cdot Bn/2+z0$

카라츠바의 빠른 곱셈 알고리즘

- 일반적인 곱셈의 시간 복잡도 : $O(n^2)$

- 카라츠바의 곱셈의 시간 복잡도 : $O(3^k) = (3^{lgn}) = O(n^{lg3})$

쿼드 트리 뒤집기 문제 (QUADTREE)

- 압축된 흑백 그림이 주어졌을 때 이를 상하로 뒤집은 그림을 쿼드 트리로 압축해서 표현하는 방법

- 압축을 재귀적으로 풀면서 작은 부분의 그림을 상하로 뒤집는 동작을 반복하여 결과적으로 모든 부분을 뒤집는다.

✓ 문제

https://algospot.com/judge/problem/read/QUADTREE

✓ 풀이코드

https://github.com/Nexters/algorithmStudy/blob/master/seokjoong/Chapter07/Quadtree.cpp

울타리 잘라내기 문제 (FENCE)

- 너비가 같은 사각형이 주어졌을 때 잘라 낼 수 있는 직사각형 중 가장 넓은 너비를 찾는 문제

- 가장 큰 직사각형을 찾을 수 있는 영역을 왼쪽, 오른쪽, 왼쪽과 오른쪽으로 걸쳤을 때 3 가지 경우로 나누어 재귀적으로 탐색한다.

✓ 문제

https://algospot.com/judge/problem/read/FENCE

✓ 풀이코드

https://github.com/Nexters/algorithmStudy/blob/master/seokjoong/Chapter07/Fence.cpp

팬미팅 문제 (FANMEETING)

- 멤버들과 팬들과 한 줄로 서있고, 남성끼리는 포옹하지 않는다고 하였을 때 모든 멤버가 포옹 하는 수를 구한다.

- 남성과 여성을 0과 1로 나누어 2진수의 곱셈으로 생각하여 문제를 해결한다. 빠른 곱셈을 위하여 카라츠바 알고리즘을 사용한다.

✓ 문제

https://algospot.com/judge/problem/read/FANMEETING

✓ 풀이코드

https://github.com/Nexters/algorithmStudy/blob/master/seokjoong/Chapter07/Fanmeeting_karatsuba.cpp

참고자료

- 알고리즘 문제 해결전략
- http://algospot.com
- https://code.msdn.microsoft.com/31e0a9b1-42b1-4454-af90-2bc8b3d10fe9