2018년도 가을학기 SCSC 알고리즘 실습 12주차

문현수, munhyunsu@cs-cnu.org

Thursday November 29, 2018

1 개요

컴퓨터공학은 기계, 항공, 우주, 전기, 전자 공학뿐 아니라 생명 공학과도 많은 영향을 주고 받으며 성장하고 있다. 특히, 생명 공학의 염기 서열 분석은 오늘날 급속도로 향상된 컴퓨팅 성능으로 그 성장이 가속화되었다. 염기 서열 분석에서 중요한 것 중 하나는 두 염기 서열이 얼마나 비슷한지 찾는 것이다. 이럴 때 주로 사용하는 알고리즘이 '최장 공통 부분 수열'찾기 알고리즘이다. 이번 실습과 과제에서는 'Longest common subsequence'를 찾는 알고리즘을 학습한다.

2 최장 공통 부분 수열(Common Subsequence)

공통 부분 수열은 두개의 수열(혹은 문자열)에서 공통으로 들어가있는 부분 수열을 의미한다. 이 때 각 수 혹은 문자가 연속해서 나타날 필요는 없고 순서만 일치하면 된다. 예를 들어 `ABCD`와 `ACBD`의 공통 부분 수열은 `AB, AC, AD, BD, CD, ABD, ACD`다.

최장 공통 부분 수열은 공통 부분 수열 중에서 가장 긴 수열들을 말한다. 위의 예에서는 `ABD, ACD`가 LCS, 즉 최장 공통 부분 수열이다.

2.1 구현 - 수도코드

알고리즘은 모든 언어에서 구현하기 쉽게 수도코드로 자주 표현된다. LCS를 구할 때에는 그 복잡도를 줄이기 위하여 다이나믹프로그래밍이 주로 사용된다. 그림 1은 `AGCAT`와 `GAC`의 공통 부분 수열을 찾는 알고리즘 결과로 생성되는 테이블이다. 이 테이블이 LCS를 구하기 위하여 꼭 채워져야하는 데이터이므로 테이블을 채워넣는 수도코드먼저 확인하도록 하자.

Listing 1: LCS 테이블 제작 수도 코드

Completed LCS Table

	Ø	A	G	C	A	T
Ø	Ø	Ø	Ø	Ø	Ø	Ø
G	Ø	↑←Ø	↖(G)	←(G)	←(G)	←(G)
A	Ø	↖(A)	↑←(A) & (G)	↑←(A) & (G)	↖(GA)	←(GA)
C	Ø	↑(A)	↑←(A) & (G)	↖(AC) & (GC)	↑←(AC) & (GC) & (GA)	↑←(AC) & (GC) & (GA)

Figure 1: AGCAT, GAC 최장 공통 부분 수열 테이블(결과)

```
1  function LCSLength(X[1..m], Y[1..n])
2      C = array(0..m, 0..n)
3      for i := 0..m
4          C[i,0] = 0
5      for j := 0..n
6          C[0,j] = 0
7      for i := 1..m
8          for j := 1..n
9              if X[i] = Y[j]
10                 C[i,j] := C[i-1,j-1] + 1
11             else
12                 C[i,j] := max(C[i,j-1], C[i-1,j])
13     return C[m,n]
```

2.2 LCS 테이블 backtrack

LCS 테이블을 읽어 최장 공통 부분 수열을 찾아내는 작업을 backtrack이라고 부른다. 모든 최장 공통 부분 수열을 찾아내기 전에 하나를 출력하는 수도코드를 살펴보자. 그림 **??**은 그림 **??**로 생각하여 계산한다.

Storing length, rather than sequences

	Ø	A	G	C	A	T
Ø	0	0	0	0	0	0
G	0	↑←0	↖1	←1	←1	←1
A	0	↖1	↑←1	↑←1	↖2	←2
C	0	↑1	↑←1	↖2	↑←2	↑←2

Figure 2: AGCAT, GAC 최장 공통 부분 수열 테이블(결과)의 backtrack 버전

Listing 2: LCS 테이블 제작 수도 코드

```
function backtrack(C[0..m,0..n], X[1..m], Y[1..n], i, j)
    if i = 0 or j = 0
        return ""
    if  X[i] = Y[j]
        return backtrack(C, X, Y, i-1, j-1) + X[i]
    if C[i,j-1] > C[i-1,j]
        return backtrack(C, X, Y, i, j-1)
    return backtrack(C, X, Y, i-1, j)
```

2.3 backtrack-all

최장 공통 부분 수열 중 하나를 출력해보았으니 모든 최장 공통 부분 수열을 출력해보자.

Listing 3: LCS 테이블 제작 수도 코드

```
function backtrackAll(C[0..m,0..n], X[1..m], Y[1..n], i,
    j)
    if i = 0 or j = 0
        return {""}
    if X[i] = Y[j]
        return {Z + X[i] for all Z in backtrackAll(C, X,
            Y, i-1, j-1)}
    R := {}
    if C[i,j-1] >= C[i-1,j]
        R := R U backtrackAll(C, X, Y, i, j-1)
    if C[i-1,j] >= C[i,j-1]
        R := R U backtrackAll(C, X, Y, i-1, j)
    return R
```

3 (과제)비속어 탐지 시스템

최장 공통 부분 수열 알고리즘을 이용하면 비속어를 탐지하는 시스템을 제작할 수 있다. 비속어 리스트를 `set`이나 `list`형태로 사전에 저장해둔 상태로 입력된 값이 비속어인지 판별해보자. 아래는 그 결과이다.

Listing 4: 비속어 탐지 시스템 결과

```
비속어인지 확인할 단어를 입력하세요 : 개
비속어가 아닙니다 .
비속어인지 확인할 단어를 입력하세요 : 개복치
비속어가 아닙니다 .
비속어인지 확인할 단어를 입력하세요 : 개아이
비속어가 있습니다 . {' 개아이 '}
비속어인지 확인할 단어를 입력하세요 : 시쓰고 싶다 .
비속어가 아닙니다 .
비속어인지 확인할 단어를 입력하세요 : 시123바 !!!
비속어가 있습니다 . {' 시바 '}
비속어인지 확인할 단어를 입력하세요 : 시바견은 귀엽다 .
비속어가 있습니다 . {' 시바 '}
비속어인지 확인할 단어를 입력하세요 : 종료합니다 .
```

3.1 과제 목표

- Longest Common Subsequence 알고리즘 설명: 예제를 들어 테이블을 채워가며 설명
- 비속어 탐지 시스템 제작

3.2 제출 관련

- 마감 날짜: 2018. 12. 05. 23:59:59
- 딜레이: 1일당 10% 감점(처음 2일까지는 -2)
- 제출 방법: 과목 사이버캠퍼스
- 제출 형식: 과제 리포트 PDF(HWP, DOC 받지 않음!), 소스코드(구현한 .py 만 추가할 것)를 압축한 .zip 파일
- 리포트에 포함해야하는 내용: 목표, 목표를 위해 알아야하는 것, 해결 방법, 결과, (선택)느낀점 or전달할 말
- 제출 파일 제목: AL_201550320_문현수_12.zip(파일명 준수!)

3.3 조교 연락처

- 문현수
- munhyunsu@cs-cnu.org
- 공학5호관 633호 데이터네트워크연구실
- 이메일, 연구실 방문