ICPC Sinchon

2022 Winter Algorithm Camp

2. 문자열

서강대학교 임지환

2022 Winter Algorithm Camp

목차

강사 소개

임지환 (info)

- BOJ – raararaara
- 서강대학교 컴퓨터공학과 (2014.3 ~ 2022.2)
- Sogang ICPC Team 학회장 (2020.1 ~ 2020.12)
- ICPC Sinchon을 만든 사람

2022 Winter Algorithm Camp

Introduction – String

✓ 문자들로 구성된 배열

s[0]	s[1]	…	s[n-2]	s[n-1]	‘\0’

2022 Winter Algorithm Camp

Introduction – 기본 & 용어 정리

* Basic Notation

문자열 S의 길이 : $|S|$ (ex: $S = \text{“}raararaara\text{”}, |S| = 10$)

S의 $i(0 \le i < |S|)$번 글자: $S[i]$ (ex: $S = \text{“}Naver\text{”}, S[3] = e$)

* Substring(부분 문자열)

S의 i번 글자부터 j번 글자까지로 구성된 문자열($i < j$) : $S[i \ldots j]$

* Subsequence(부분 수열)

문자열의 일부 글자가 원래의 순서를 유지하며 나열된 형태

2022 Winter Algorithm Camp

Introduction – 기본 & 용어 정리

* prefix(접두사)

  S의 0번 글자부터 a번 글자까지로 구성된 부분 문자열 ($a < |S|$)

* Suffix(접미사)

  S의 a번 글자부터 끝까지로 구성된 부분 문자열

2022 Winter Algorithm Camp

Introduction – `std::string` (reference)

* 기본 입력 (공백 이전까지)

```
1 #include <string>
2 using namespace std;
3
4 string S;
5 cin >> S;
```

* 공백 포함

```
1 string S;
2 getline(cin, S);
```

* `std::basic_istream::ignore` (link)

구분 문자(delimeter)를 포함하여, 그 이전까지의 문자들을 stream에서 제거

2022 Winter Algorithm Camp

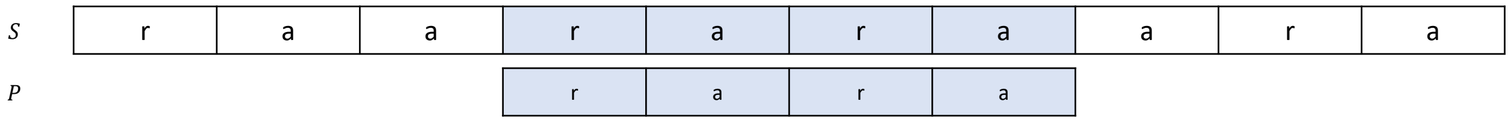
String Matching – Intro

String matching(= Pattern matching)

- string(or sentence) $S[1..n]$, pattern $P[1..m]$이 주어졌을 때 $(n >= m)$

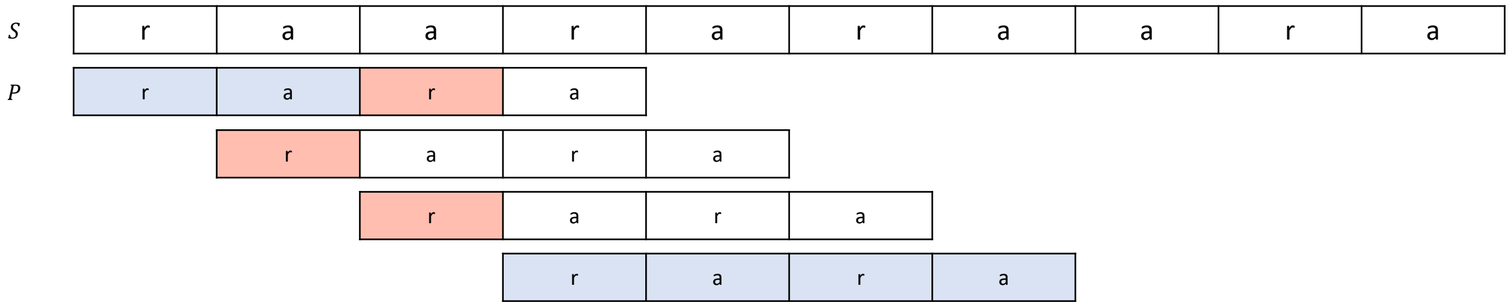
  S 내에서 P가 부분 문자열로 등장하는가?

- S의 substring 중 길이가 m인 것의 개수 : $n - m + 1$



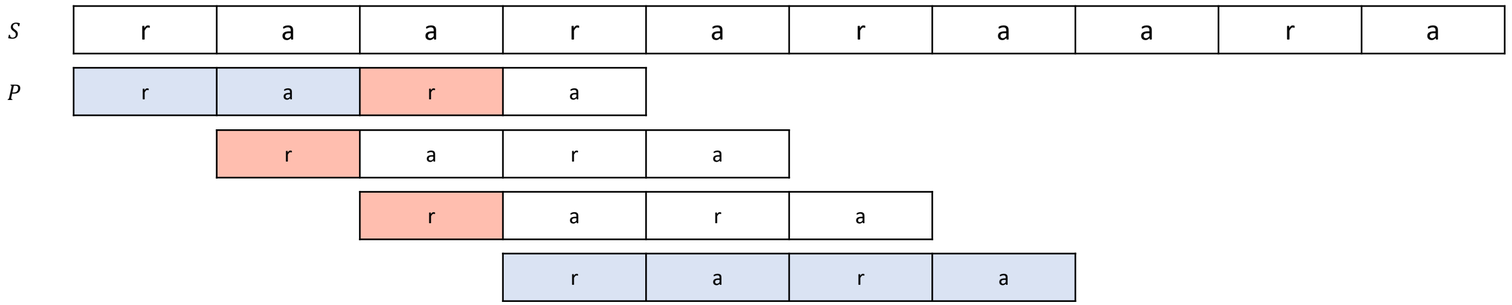
2022 Winter Algorithm Camp

String Matching – Naïve approach




```
PatternMatchingBF(S[1..n], P[1..m]):
    for s <- 1 to n - m + 1
        equal = True
        i <- 1
        while equal and i <= m
            if S[s+i-1] != P[i]
                equal <- False
            else
              i <- i + 1
        if equal
            return s
    return None
```

String Matching – Naïve approach

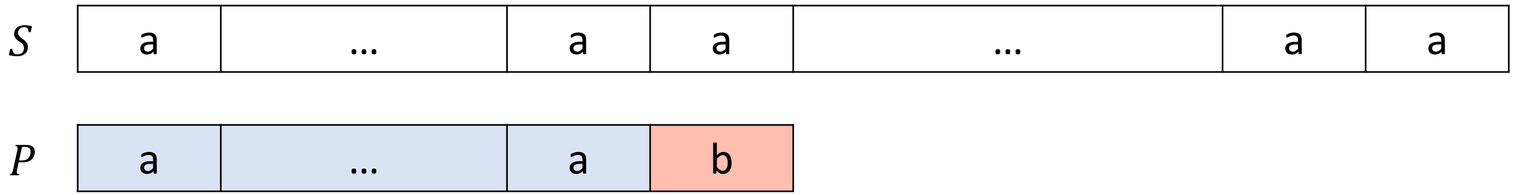



```
PatternMatchingBF(S[1..n], P[1..m]):
    for s <- 1 to n - m + 1
        equal = True
        i <- 1
        while equal and i <= m
            if S[s + i - 1] != P[i]
                equal = False
            i <- i + 1
        if equal
            return s
    return None
```

$$O(nm)$$

String Matching – Naïve approach

* What if?



String Matching – strings to numbers

* Strings -> numbers

- 문자도 값을 가진다. (ex: a = 97, b = 98, ⋯)

* 문자열을 정수로 치환할 수 있다면

- 각 케이스 별 매치 여부 : $O(m) \rightarrow O(1)$
- Total Time Complexity : $O(nm) \rightarrow O(n)$
  - $O(m)$ for preprocessing (pattern string to int)
  - $O(n-m+1+m)$ for comparing

String Matching – Polynomial Rolling hash

given string(or sentence) $S[1..n]$, pattern $P[1..m]$ $(n > m)$

let ‘a’ = 1, ‘b’ = 2, ‘c’ = 3, ⋯, ‘y’ = 25, ‘z’ = 26,

$p(x)$: pattern $P[1..m]$을 정수화시킨 값

$s_k(x)$: string $S[1..n]$의 부분문자열 $S[k..k+m-1]$을 정수화시킨 값

$$p(x) = \left(\sum_{i=0}^{m-1} x^i \cdot P[m-i-1] \right) mod\ M, \qquad s_k(x) = \left(\sum_{i=0}^{m-1} x^i \cdot S[k+m-i-1] \right) mod\ M$$

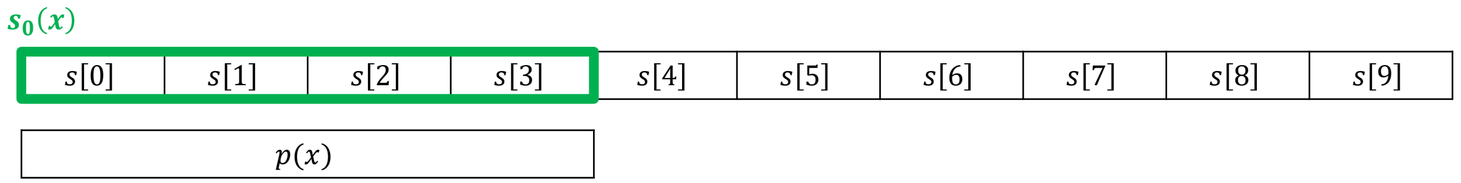
ex: icpcsinchon

i	c	p	c	s	i	n	c	h	o	n
$9x^{10}$	$3x^9$	$16x^8$	$3x^7$	$19x^6$	$9x^5$	$14x^4$	$3x^3$	$8x^2$	$15x^1$	$14x^0$

String Matching – Polynomial Rolling hash

* String Matching - Comparison

전처리한 $p(x)$와 $s_i(x)$ 비교 $\rightarrow O(1)$

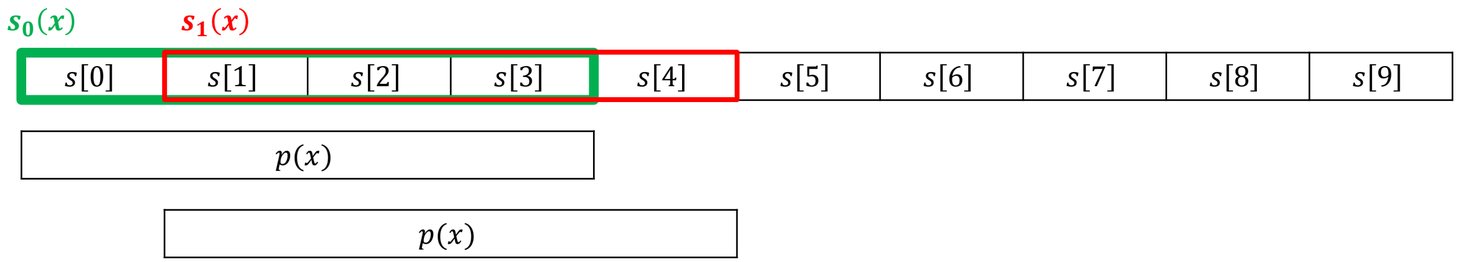


String Matching – Polynomial Rolling hash

* Shifting Process - $s_i(x)$로부터 $s_{i+1}(x)$ 도출 -> $O(1)$ per operation

$$s_0(x) = s[0]x^3 + s[1]x^2 + s[2]x^1 + s[3]x^0$$
$$s_1(x) = s[1]x^3 + s[2]x^2 + s[3]x^1 + s[4]x^0$$

$$s[1]x^3 + s[2]x^2 + s[3]x^1 = (s[1]x^2 + s[2]x^1 + s[3]x^0)x$$
$$s_1(x) = (s_0(x) - s[0]x^3)x + s[4]x^0$$



String Matching – Hash collisions

* 해시 충돌 (Hash collision)

두 문자열의 내용이 다르지만 해시값이 같은 경우

* 해결법

- Chaining
- Open Addressing

...

다중 해시: 다른 상수를 이용하여 해시값을 여러 번 계산

2022 Winter Algorithm Camp

2866. 문자열 잘라내기

R개의 행과 C개의 열로 이루어진 테이블 $(2 \leq R, C \leq 1{,}000)$

테이블의 각 열을 위에서 아래로 읽어서 열 간 문자열이 중복되지 않는다면 가장 위의 행 삭제

최초 테이블은 동일한 문자열이 존재하지 않음(count변화없이 지운다고 가정)

몇 행까지 삭제할 수 있을까?

2022 Winter Algorithm Camp

2866. 문자열 잘라내기

1. 크기가 최대 1000×1000

2. 각 행으로 시작하는 접미사(suffix)문자열의 hash값을 배열에 저장하자.

S	O	G
A	N	G
B	A	B
A	B	A

$Sx^3 + Ax^2 + Bx + A$	$Ox^3 + Nx^2 + Ax + B$	$Gx^3 + Gx^2 + Bx + A$
$Ax^2 + Bx + A$	$Nx^2 + Ax + B$	$Gx^2 + Bx + A$
$Bx + A$	$Ax + B$	$Bx + A$
A	B	A

2022 Winter Algorithm Camp

2866. 문자열 잘라내기

1. 크기가 최대 1000×1000

2. 각 행으로 시작하는 접미사(suffix)문자열의 hash값을 배열에 저장하자.

S	O	G
A	N	G
B	A	B
A	B	A

$Sx^3 + Ax^2 + Bx + A$	$Ox^3 + Nx^2 + Ax + B$	$Gx^3 + Gx^2 + Bx + A$
$Ax^2 + Bx + A$	$Nx^2 + Ax + B$	$Gx^2 + Bx + A$
$Bx + A$	$Ax + B$	$Bx + A$
A	B	A

2022 Winter Algorithm Camp

10840. 구간 성분

길이가 N, M인 두개의 문자열 $(1 \leq N, M \leq 1{,}500)$

어떤 구간에 포함된 문자의 종류와 개수가 순서에 상관없이 동일하면 '구간의 성분'이 같다.

두 문자열에서 같은 성분을 가진 구간 중 가장 긴 구간의 길이를 구해보자.

2022 Winter Algorithm Camp

10840. 구간 성분

1. 구간의 성분은 부분문자열 상에서의 위치에 관계가 없다.

$[L, R]$에서 나온 문자 $c(0 \leq c \leq 25)$의 개수를 $cnt[c]$라 할 때,

$$h(x) = cnt[0]x^0 + cnt[1]x^1 + cnt[2]x^2 + \cdots + cnt[25]x^{25}$$

2022 Winter Algorithm Camp

10840. 구간 성분

2. 특정 구간 길이 L에 대한 가능성 여부 판단

$|S_1| = N,\ |S_2| = M$이라 할 때
각 부분 문자열의 개수 = $N - L + 1,\ M - L + 1$

즉, $O(N - L)$개의 부분문자열 집합에서 각 $O(M - L)$개의 부분문자열이 등장하는지를 확인 -> 이분탐색, std::set, std::map, dictionary..

$$O(N \log N)$$

10840. 구간 성분

3. 가장 긴 구간의 길이를 구해야 하니, 후보 길이에 대해 역순 탐색을 해보자.

```
string s[2];

for (int i = 0; i < 2; i++)
    cin >> s[i];

N = (int)s[0].size(), M = (int)s[1].size();

for (int len = min(N, M); len; len--)
    if (found(len))
      return cout << len, 0;

cout << 0;
```

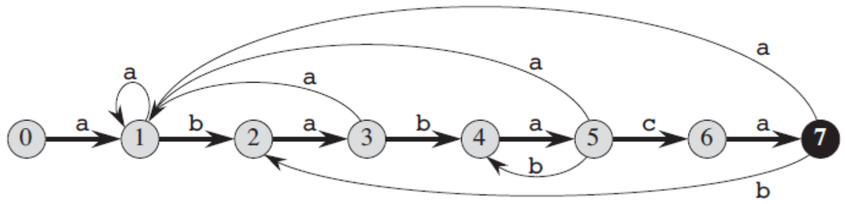
Appendix – Additional Topics

* Finite State Machine & Pattern matching

S의 부분문자열을 입력으로 하고, 상태를 패턴의 매칭 수로, 상태 전이는 패턴마다의 알파벳으로 설정하면 유한 상태 기계를 통해 패턴매칭을 시도할 수 있습니다.

FSM을 이용하여 구현할 수 있는 것들은 다음과 같습니다.

- 정규표현식(Regular Expression) : 특정한 규칙을 가진 문자열의 집합을 표현하는데 사용하는 식입니다.
- Knuth-Morris-Pratt(KMP) 알고리즘 : 해싱과 달리 확률론적이지 않은 패턴매칭 알고리즘입니다.

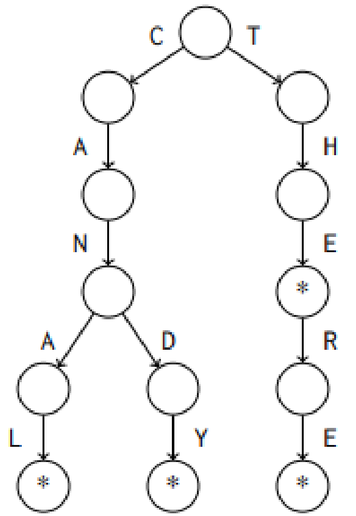


Appendix – Additional Topics

* Trie

루트 트리 구조 형태로 문자열의 집합을 관리할 수 있습니다. 각 문자열은 트리의 루트 노드에서 시작하는 글자 경로의 형태로 저장됩니다.
특정 문자열(S)이 집합에 존재하는지를 $O(|S|)$에 찾을 수 있습니다.

트라이 구조를 이용한 Aho-Corasick(아호-코라식) 알고리즘을 통해 단일 패턴이 아닌 다중 패턴을 찾을 수 있습니다.
아호코라식 알고리즘의 동작원리는 KMP의 failure function을 Automata형태로 나타낸 형태로 간선을 정의합니다. KMP 알고리즘에 대한 이해를 선수지식으로 요구합니다.



Appendix – Problems

필수문제

11091	알파벳 전부 쓰기
8892	팰린드롬
5525	IOIOI
2866	문자열 잘라내기
16916	부분 문자열

연습문제

4583	거울상	23304	아카라카
6996	애너그램	5052	전화번호 목록
15904	UCPC는 무엇의 약자일까?	1701	Cubeditor
1764	듣보잡	10840	구간 성분
14425	문자열 집합		