Trie를 알아보자!

J029 김도연

https://programmers.co.kr/learn/courses/30/lessons/17685

어떻게 풀지?

코딩테스트 연습 > 2018 KAKAO BLIND RECRUITMENT > **[3차] 자동완성**

[3차] 자동완성

문제 설명

자동완성

포털 다음에서 검색어 자동완성 기능을 넣고 싶은 라이언은 한 번 입력된 문자열을 학습해서 다음 입력 때 활용하고 싶어 졌다. 예를 들어, `go` 가 한 번 입력되었다면, 다음 사용자는 `g` 만 입력해도 `go` 를 추천해주므로 `o` 를 입력할 필요가 없어진다! 단, 학습에 사용된 단어들 중 앞부분이 같은 경우에는 어쩔 수 없이 다른 문자가 나올 때까지 입력을 해야 한다.
효과가 얼마나 좋을지 알고 싶은 라이언은 학습된 단어들을 찾을 때 몇 글자를 입력해야 하는지 궁금해졌다.

예를 들어, 학습된 단어들이 아래와 같을 때

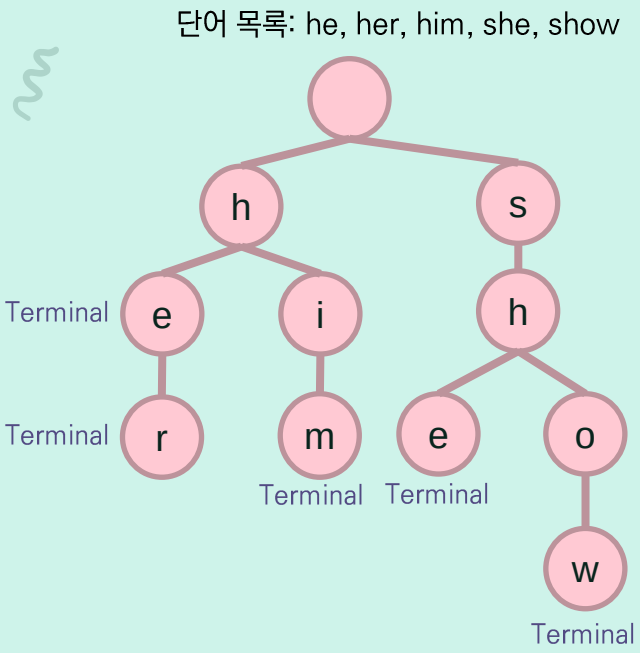
```
go
gone
guild
```

- `go` 를 찾을 때 `go` 를 모두 입력해야 한다.
- `gone` 을 찾을 때 `gon` 까지 입력해야 한다. (`gon` 이 입력되기 전까지는 `go` 인지 `gone` 인지 확신할 수 없다.)
- `guild` 를 찾을 때는 `gu` 까지만 입력하면 `guild` 가 완성된다.

이 경우 총 입력해야 할 문자의 수는 `7` 이다.

라이언을 도와 위와 같이 문자열이 입력으로 주어지면 학습을 시킨 후, 학습된 단어들을 순서대로 찾을 때 몇 개의 문자를 입력하면 되는지 계산하는 프로그램을 만들어보자.

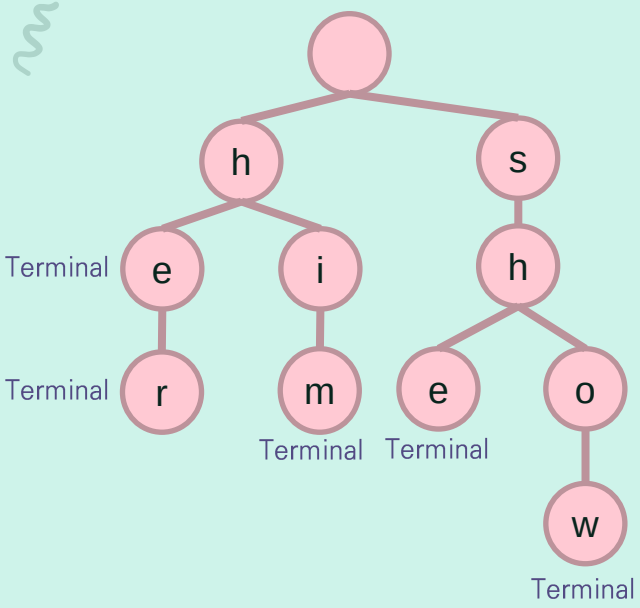
Trie가 뭔가요?



- 검색을 뜻하는 retrieval에서 온 단어
- 문자열을 저장하고 효율적으로 탐색하기 위한 트리 형태의 자료구조
- 탐색 트리의 일종으로 동적 집합이나 연관 배열을 저장하는데 사용된다
- 각 Node는 글자를 갖는 **key**(ex. s, h, e)와 단어의 마지막 글자인지를 알려주는 **flag**(terminal)로 이루어져 있습니다.

Trie는 언제 왜 쓰나요?

단어 목록: he, her, him, she, show



사용 목적 및 용도

- 문자열 탐색을 할 때 효율적이다. 단순하게 하나씩 탐색하는 것보다 빠르게 탐색이 가능하다는 장점이 있다. (그러나, 저장 공간의 크기가 크다는 단점이 있다.)
- 검색어 자동완성, 사전 찾기, 문자열 검사 등에서 사용될 수 있다.

시간 복잡도

- 제일 긴 문자열의 길이가 L, 총 문자열들의 수가 M 일 때
- 트리 생성 시간 복잡도: O(L*M)
  모든 문자열을 넣어야 하기 때문에
- 트리 탐색 시간 복잡도: O(L)
  가장 긴 문자열의 길이만큼만 탐색하기 때문에

Python으로 Trie 구현해보기

```python
class Node(object):
    def __init__(self, key, data=None):
        self.key = key
        self.data = data
        self.children = {}
```

1. Node 구현

- key 필드는 단어의 글자 하나를 담는데 이용한다. (ex, h, e, r)
- data 필드는 글자의 마지막을 나타내는 flag로 사용한다. (ex. h, e, r 에서 e 에는 he가 r에는 her이 들어가며 h에는 None이 들어간다.)

```python
class Trie(object):
    def __init__(self):
        self.head = Node(None)
```

2. Trie 구현

Python으로 Trie 구현해보기

3. Trie의 insert 함수 구현

- Trie에 문자열 삽입

```
def insert(self, string):
    current_node = self.head
    for char in string:
        if char not in current_node.children: # 자식이 없다면 자식으로 등록해준다.
            current_node.children[char] = Node(char)
        current_node = current_node.children[char] # 다음 자식으로 이동
    current_node.data = string # 마지막까지 도달하면 문자열 전체를 저장해준다.
```

Python으로 Trie 구현해보기

4. Trie의 search 함수 구현

- 주어진 단어 string이 트라이에 존재하는지 여부 반환

```
def search(self, string):
    current_node = self.head
    for char in string: # 입력받은 문자열을 한글자씩 탐색
        if char in current_node.children:
            current_node = current_node.children[char]
        else:
            return False
    if (current_node.data != None): # 마지막 노드(글자)의 data가 있다면
        return True # string이 trie에 존재함
    return False # 그렇지 않다면 존재하지 않음
```

Python으로 Trie 구현해보기

5. Trie의 starts_with 함수 구현

- 특정 prefix로 시작하는 단어들을 트라이에서 찾아 리스트로 반환

```
def starts_with(self, prefix):
    current_node = self.head
    result = []
    subtrie = None
    for char in prefix: # trie 에서 prefix를 찾음
        if char in current_node.children:
            current_node = current_node.children[char]
            subtrie = current_node # prefix의 마지막 노드를 subtrie로 설정
        else:
            return None
    queue = list(subtrie.children.values())
    while queue: # subtrie의 자식들을 하나씩 순회함
        current_node = queue.pop()
        if current_node.data != None: # data가 있는 노드(=완전한 단어)일 경우
            result.append(current_node.data) # 반환 값에 더해줌
        queue += list(current_node.children.values())
    return result
```

THANKS

출처

https://ko.wikipedia.org/wiki/트라이_(컴퓨팅)
https://twpower.github.io/187-trie-concept-and-basic-problem
https://blog.ilkyu.kr/entry/파이썬에서-Trie-트라이-구현하기 [컴언]

관련 알고리즘 문제

https://programmers.co.kr/learn/courses/30/lessons/17685

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**