Lec2_Models of Computation

Goal

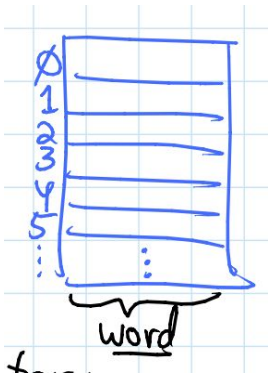
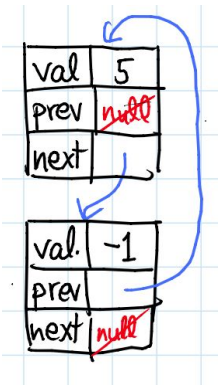
- What's an algorithm? What is time?
- Random Access Machine
- Pointer Machine
- Python Model
- Document Distance : Problem & Algorithms

What's an algorithm?

- Mathematical abstraction of computer program
- Computational procedure to solve a problem

Model of Computation

-

Random Access Machine	Pointer Machine
 Modeled by big array	 Dynamically allocated objects

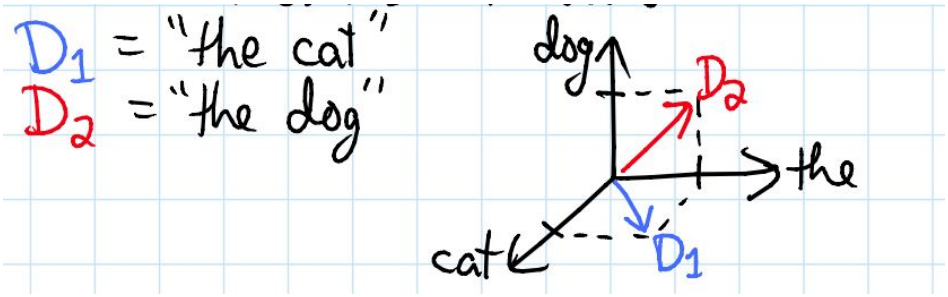
Python Model

1. List → RAM
    a. L[i] = l[j] + 5 → Θ(1) time
    b. L.append(x) → Θ(1) time
    c. L = L1 + L2 → Θ(1 + L1 + L2) time

    d. L1. extend(L2) → Θ(1+L2) time
    e. len(L) → Θ(1) time
    f. L.sort() → Θ(|L|*lg|L|) time
2. Object with O(1) attributes → Pointer machine
    a. x = x.next → Θ(1) time
3. dict
    a. D[key] = val (key in D) → Θ(1) time
4. heap q
    a. heappush & heappop → Θ(lg n) time
5. long
    a. x+y → O(|x|+|y|) time
    b. x*y → O((|x|+|y|)^lg 3) time

Document Distance Problem

- applications : find similar documents
- How to approach?
    - word = sequence of alphanumeric chars
    - document = sequence of words
    - Think of document D as **VECTOR**



    - Distance : d''(D1,D1) = (D1*D2) / (|D1|*|D2|)
    - Geometric re-scaling d(D1,D2) = arccos(d''(D1,D2))

- Distance Algorithm
    - Split each document into words
        - Θ(Doc)
    - Count word frequencies document
        - Θ(k * lg(k) *|word|) → k : # words
    - Compute dot product
        - O(|doc|)
            -

Code : If words equal

```
        Total += count1 * count2
If word1 <= word2
        Advance list 1
else
        Advance list 2
Repeat until either list done
```