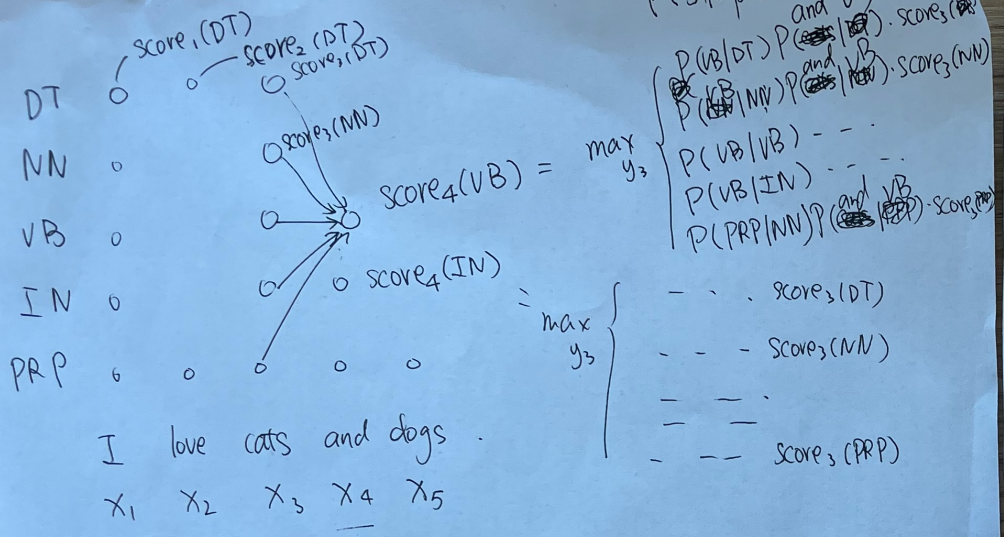
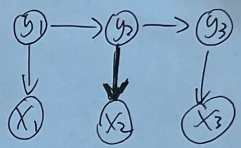



## HMM




n=3

| $x_1$ | $x_2$ | $x_3$ |
|---|---|---|
| I | love | cats |
| PRP | VBP | NNS |
| $y_1$ | $y_2$ | $y_3$ |

. edges convey conditional independence

i.e. $x_2$ is conditionally independent of everything else give $y_2$

$x_2 \perp\!\!\!\perp y_1, x_1 \mid y_2$.

$P(\mathbf{x}, \mathbf{y})$

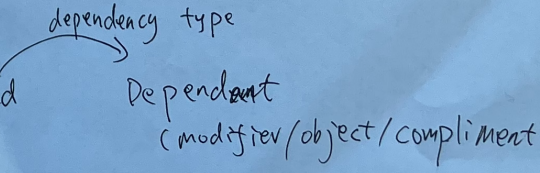
$= P(x_1, x_2, x_3, y_1, y_2, y_3)$

$= P(y_1) \cdot P(x_1|y_1) \cdot P(y_2|y_1, x_1) \cdot P(x_2|y_1, x_1, y_2) \cdot P(y_3|y_1, x_1, y_2, x_2) \cdot P(x_3|y_1, x_1, y_2, x_2, y_3)$

$P(y_2|y_1)$ ; $P(x_2|y_2)$ ; $P(y_3|y_2)$ ; $P(x_3|y_3)$

## Dependency Grammars

syntactic structure = lexical items linked by binary asymmetrical relations called dependency



# Dynamic Programming

1, 1, 2, 3, 5, 8, 13, ...

Fibonacci Numbers: $F_1 = F_2 = 1$; $F_n = F_{n-1} + F_{n-2}$

## ◦ naive algorithm

```
fib(n):
    if n <= 2, return 1
    else return fib(n-1) + fib(n-2).
```

$\Rightarrow T(0) = T(1) = 1$

$T(n) = T(n-1) + T(n-2) + O(1)$ (constant) $= c$

$\geq 2T(n-2) + O(1) = c$

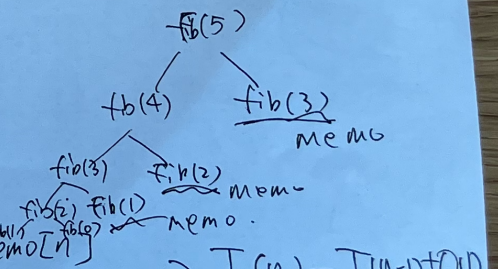
$\geq 2(2T(n-4)+c)+c = 4T(n-1) + 3c$

$\vdots$

$\geq 2^k T(n-2k) + (2^k - 1)c$

when $n - 2k = 0 \Rightarrow k <= n/2$

$T(n) \sim 2^{n/2} \cdot T(0)$




$\Rightarrow T(n) = T(n-1) + O(1)$

$= O(n)$.

## ◦ dynamic programming

```
memo = {}
fib(n):
    if n in memo, return memo[n]
    else: if n <= 2: f = 1
          else: f = fib(n-1) + fib(n-2)
          memo[n] = f
          return f.
```

free lookup.