Dependency Grammars

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

$$\text{Head} \xrightarrow{\text{Dependency Type}} \text{Dependant}$$

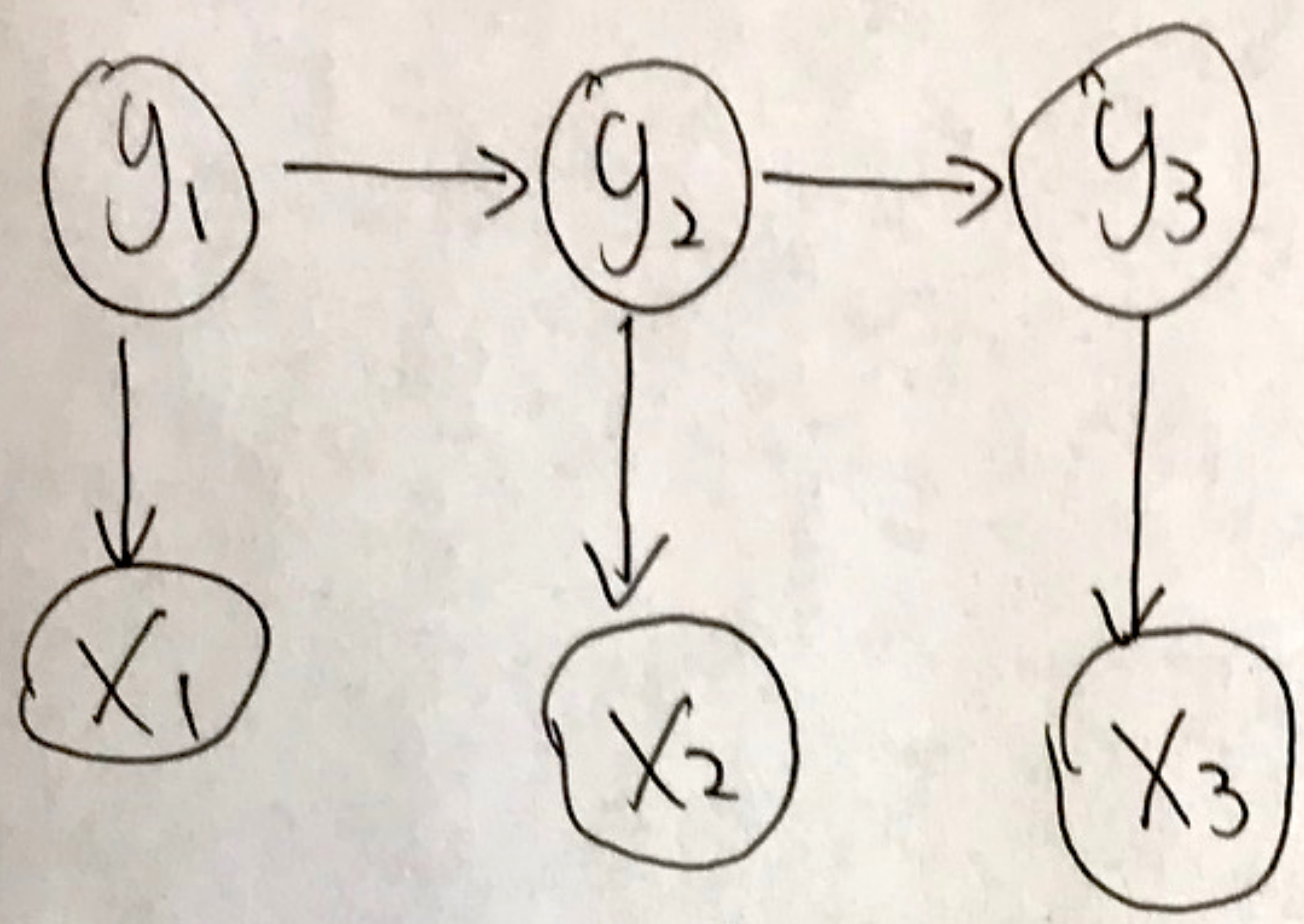
(modifier / object / compliment)

$$\underbrace{P(Y|X)}_{\text{posterior}} = \frac{P(Y,X)}{P(X)} = \frac{\overbrace{P(X|Y)}^{\text{likelihood}}\ \overbrace{P(Y)}^{\text{prior}}}{P(X)}$$

$$p(x_i|y_i) \propto \exp \theta^T \boxed{f} (x_i, y_i)$$

suffix
hyphen
capital letters
numbers
⋮

First-Order Markov Assumption



Assume that

$y_2 \perp\!\!\!\perp x_1 \mid y_1$

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

$y_3 \perp\!\!\!\perp \cancel{x}\, y_1, x_1, x_2 \perp y_2$

$x_3 \perp\!\!\!\perp y_1, x_1, y_2, x_2 \perp y_3$

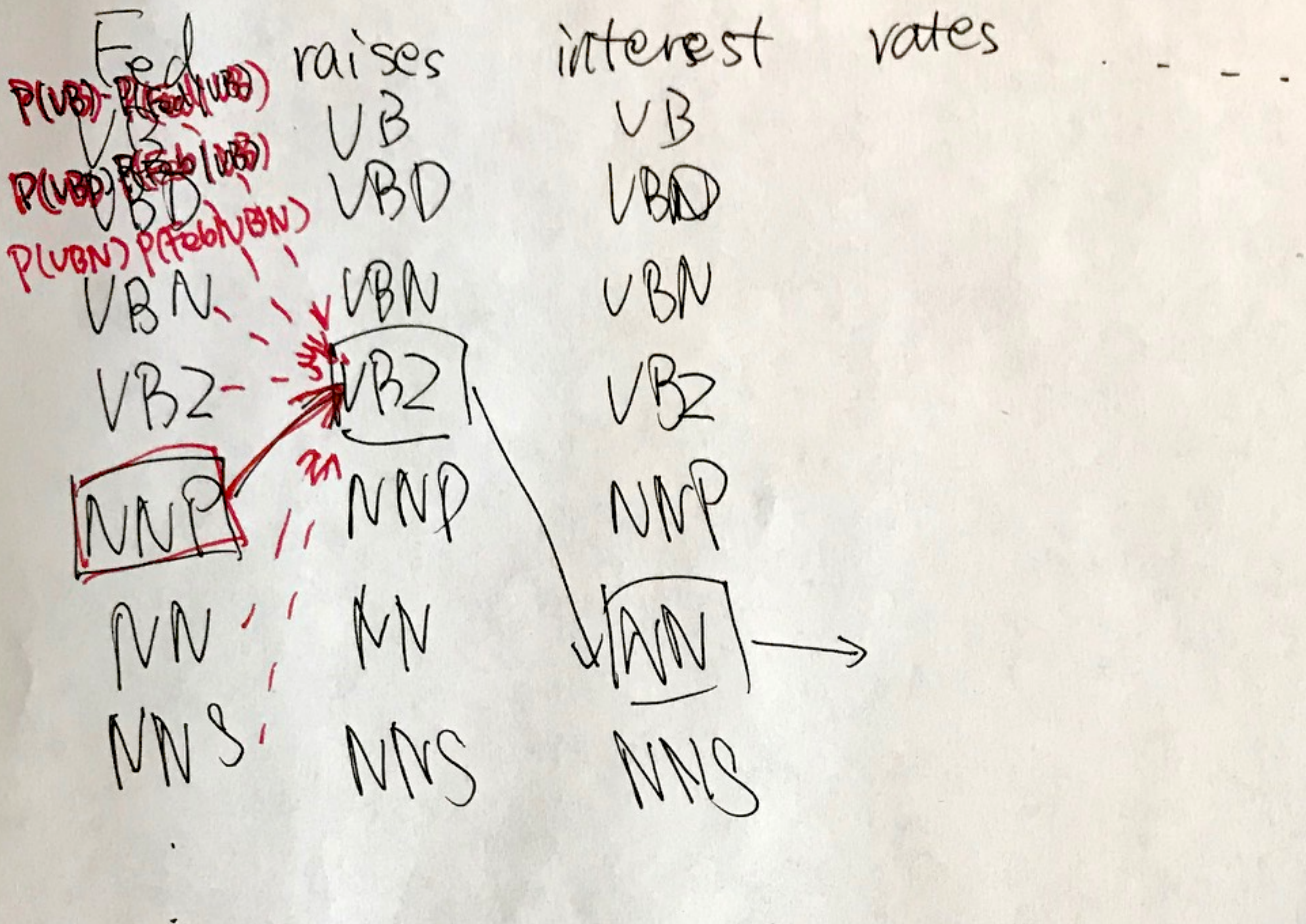
$P(\mathbf{x}, \mathbf{y}) = P(x_1, x_2, x_3, y_1, y_2, y_3)$

$\cancel{= P(x_1)\, P(y_1 \mid x_1)}$

Chain Rule: $= P(y_1)\, P(x_1 \mid y_1)\, P(y_2 \mid y_1, x_1)\, P(x_2 \mid y_1, x_1, y_2)$
$P(y_3 \mid y_1, x_1, y_2, x_2)\, P(x_3 \mid y_1, x_1, y_2, x_2, y_3)$

Markov Assumption

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



$$\mathrm{score}_1(y_1) = P(y_1)\,P(x_1|y_1)$$

$$\mathrm{score}_2(y_2) = \max_{y_1} P(y_2|y_1)\,P(x_2|y_2)\cdot \mathrm{score}_1(y_1)$$

$$\vdots$$

$$\mathrm{score}_i(y_i) = \max_{y_{i-1}} P(y_i|y_{i-1})\,P(x_i|y_i)\cdot \underline{\underline{\mathrm{score}_{i-1}(y_{i-1})}}$$

$1, 1, 2, 3, 5, 8, 13, \ldots$

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)
```

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

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

$$\geq 2T(n-2) + \underbrace{O(1)}_{c}$$

$$\cancel{\geq 4T(n-4)+}$$

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

$$\vdots$$

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

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

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

Fibonacci

Dynamic Programming

memo = {}

fib (n) :

if n in memo : return memo[n]

else : if $n \le 2$: f = 1

else : $f = fib(n-1) + \underbrace{fib(n-2)}_{\text{free}}$

memo[n] = f

return f .

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