(10) Relations

$A_1, A_2, \ldots, A_n$ - sets

def $\langle a_1, a_2, \ldots, a_n \rangle$ - n-tuple (or just tuple) on $A_1, \ldots, A_n$ - ordered sequence of elements $a_i \in A_i$, $i=1,\ldots,n$

Ex: $A_1 = \{1, 2\}$, $A_2 = \{Alice, Bob\}$ $(n=2)$

$\langle 2, Alice \rangle$ - 2-tuple ("pair") on A_1, A_2.

$\langle 2, Alice \rangle \neq \langle Alice, 2 \rangle$ - order matters.

$A_1 \times A_2 \times \ldots \times A_n = \{\langle a_1, a_2, \ldots, a_n \rangle \mid a_i \in A_i, i=1,\ldots,n\}$ - cross-product (Cartesian product) of $A_1, A_2, \ldots, A_n$ - the set of all the possible tuples on $A_1, \ldots, A_n$.

def A relation R on $A_1, A_2, \ldots, A_n$ is a (not necessarily proper) subset of $A_1 \times A_2 \times \ldots \times A_n$.

Ex: $A_1 = \{1\}$, $A_2 = \{Alice, Bob\}$, $A_3 = \{\bigcirc, \Delta, \Box\}$ $(n=3)$

$R_0 = \emptyset$ - a legal relation on A_1, A_2, A_3 (in fact, R_0 is a legal relation on any sets as long as R_0's arity (defined below) has not been pre-defined)

$R_1 = \{\langle 1, Alice \rangle\}$ - a relation on A_1, A_2.

$R_2 = \{\langle 1, \bigcirc, Alice \rangle, \langle 1, \Delta, Bob \rangle\}$ - a relation on A_1, A_3, A_2.

$R_3 = \{\langle 1, 1, 1 \rangle\}$ - a relation on A_1, A_1, A_1

$R_4 = \{\langle 1, \bigcirc \rangle, \langle \bigcirc, 1 \rangle\}$ - not a relation on A_1, A_3; not a relation on A_3, A_1

def: Arity of relation $R \subseteq A_1 \times A_2 \times \ldots \times A_n$ is n. When $n=2$, relation is "binary", and its tuples are called "pairs". When $n=1$, a relation is "unary", or we can just treat it as a subset ($R \subseteq A_1$). Arity can be 0; a relation of such arity can either be empty ($\emptyset$) or or contain an empty tuple $\langle \rangle$.

def: Cardinality (or size) of a relation is defined as the set cardinality (relations are sets)

Ex: $R \subseteq People \times \mathbb{N} \times \mathbb{N} \times \mathbb{N}$

$R = \{\langle p, d, m, y \rangle \mid$ person p's birth date is m-d-$y\}$

$\langle me, 29, 8, 1985 \rangle \in R$

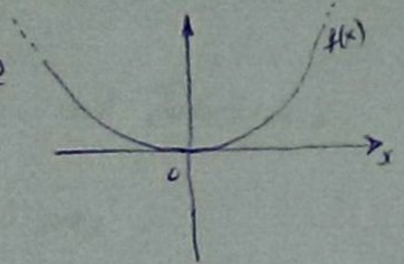
$arity(R) = 4$

$|R| = |People|$ (cardinality)

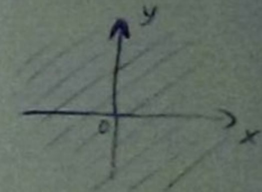
Functions as relations:

$f: \mathbb{R} \to \mathbb{R}$

$f(x) = x^2$



$\mathbb{R} \times \mathbb{R} = \{\langle x, y\rangle \mid x \in \mathbb{R}, y \in \mathbb{R}\}$ –



(2d - plane)

$f \subseteq \mathbb{R} \times \mathbb{R}$

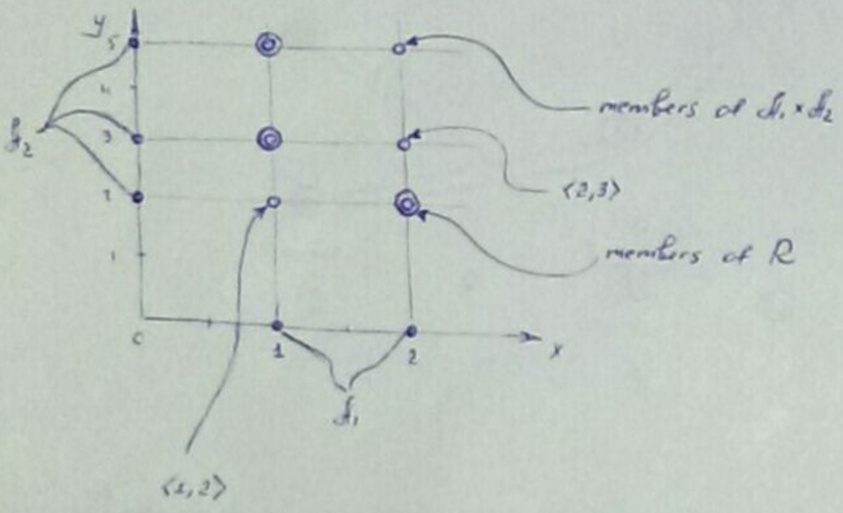
$f = \{\langle x, y\rangle \mid x \in \mathbb{R}, y \in \mathbb{R}, y = x^2\}$

Representing relations:

1) Graph ("grid" - a better name when relations are defined on discrete sets):

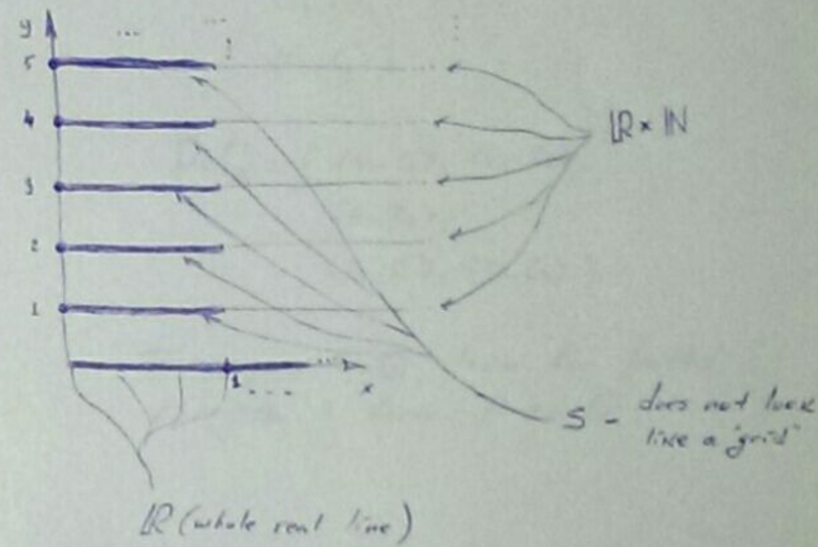
$R \subseteq \overset{\mathcal{A}_1}{\{1, 2\}} \times \overset{\mathcal{A}_2}{\{2, 3, 5\}}$

$R = \{\langle x, y\rangle \mid x \in \mathcal{A}_1,\ y \in \mathcal{A}_2,\ (x+y) - \text{even}\}$



$S \subseteq \mathbb{R} \times \mathbb{N}$

$S = \{\langle x, y\rangle \mid x \in \mathbb{R},\ y \in \mathbb{N},\ 0 \le x \le 1,\ y \le 5\}$



2) Boolean matrix:

$\mathcal{A}_1 \backslash \mathcal{A}_2$	2	3	5
1	0	1	1
2	1	0	0

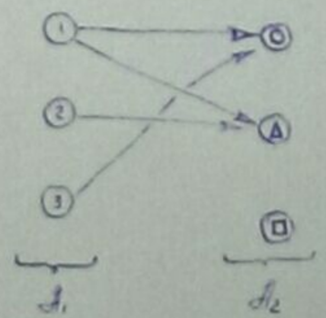
$\langle 1, 5\rangle \in R$

$\langle 2, 3\rangle \notin R$

3) Directed graph:

$R \subseteq \overset{\mathcal{A}_1}{\{1, 2, 3\}} \times \overset{\mathcal{A}_2}{\{\bigcirc, \triangle, \square\}}$

$R = \{\langle 1, \bigcirc\rangle, \langle 1, \triangle\rangle, \langle 2, \triangle\rangle, \langle 3, \bigcirc\rangle\}$



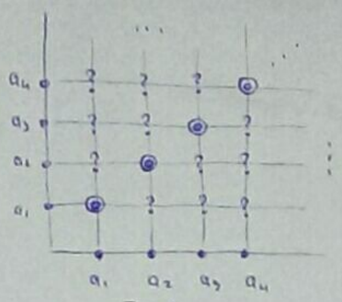
Properties of relations:

R – binary relation on A. ($R \subseteq A \times A$)

1) **R** – reflexive iff $\forall a \in A: \langle a, a \rangle \in R$

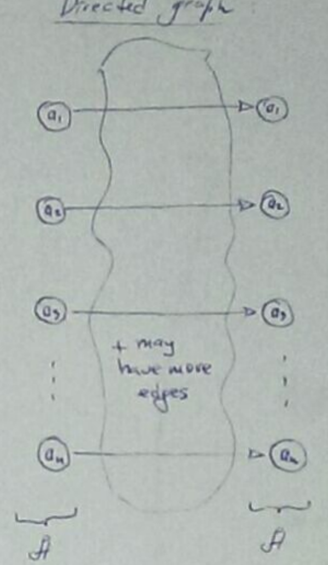
$A \backslash A$	a_1	a_2	a_3	...	a_n
a_1	1	?	?	...	?
a_2	?	1	?	...	?
a_3	?	?	1	...	?
⋮	⋮	⋮	⋮	⋱	?
a_n	?	?	?	...	1

Boolean matrix



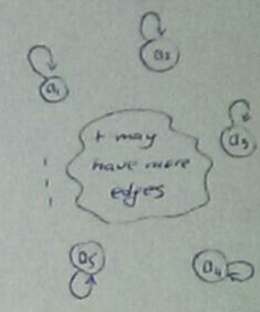
Grid

Directed graph:



OR

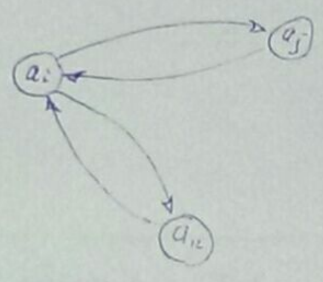
Compact notation (only one copy of A):



2) R – symmetric iff $\forall a, b \in A: \langle a, b \rangle \in R \rightarrow \langle b, a \rangle \in R$

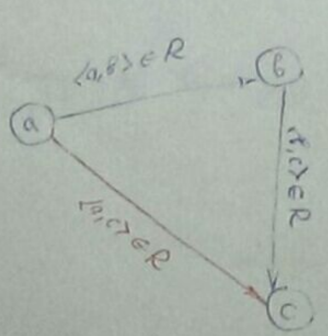
$A \backslash A$	a_1	a_2	a_3	...
a_1		x	y	⋮
a_2	x		z	⋮
a_3	y	z		
⋮		...		⋱

Boolean matrix



Directed graph

3) R – transitive iff $\forall a, b, c \in A: (\langle a, b \rangle \in R \wedge \langle b, c \rangle \in R) \rightarrow (\langle a, c \rangle \in R)$



Question: symmetric + transitive $\overset{?}{\Rightarrow}$ reflexive? (Try

	a_1	a_2	a_3
a_1	1	0	0
a_2	0	0	0
a_3	0	0	1

)

Self-study: closures, equivalence relations, partitioning into equivalence classes.

Claim: $K_n = \sum_{i=0}^{n} \frac{1}{2^i} =$ [redacted] $2 - \frac{1}{2^n}$

Basis:

$K_0 \overset{?}{=} 2 - \frac{1}{2^0} = 1$ ✓

$\sum_{i=0}^{0} \frac{1}{2^i} = \frac{1}{2^0} = 1$

Hypothesis: $K_m = 2 - \frac{1}{2^m}$

Step: $K_{m+1} = \sum_{i=0}^{m+1} \frac{1}{2^i} = \sum_{i=0}^{m} \frac{1}{2^i} + \frac{1}{2^{m+1}} =$

$= (\text{from hypo}) = 2 - \frac{1}{2^m} + \frac{1}{2^{m+1}} =$

$= 2 - \frac{2}{2^{m+1}} + \frac{1}{2^{m+1}} = 2 - \frac{1}{2^{m+1}}$

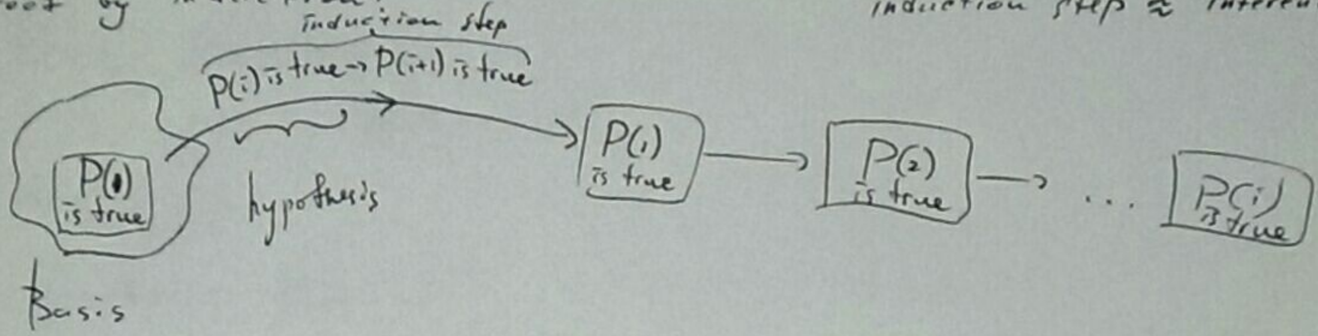
$\square$

Induction:

$P(n)$ – want to prove a (parameterised) proposition for all n ($\in \mathbb{N}$ - e.g.)

Proof by induction:

Induction step ≈ "Inference law"

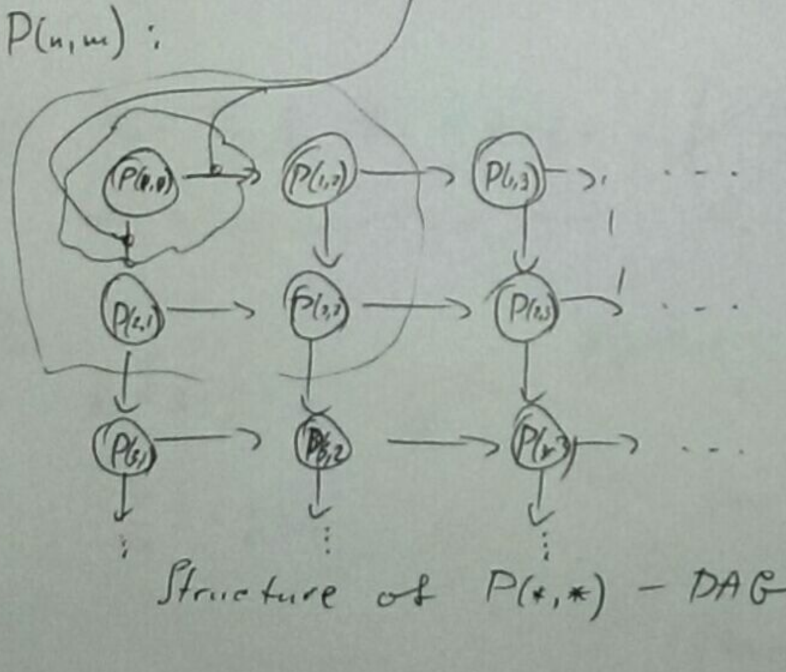



```
Produce-proof( P_i ) {
  emit(P_0 is true [premise])
  for k = 0 to i
     emit( P_k is true -> P_k+1 is true [inductive step])
}
```

Complex layouts:

2 steps (2 "inference laws")

$P(n,m)$:



Structure of $P(*,*)$ – DAG

Basis set may be larger!

6) Functions:

$f(n) = 6n \sim f(n+1) = f(n) + 6, \; f(0) = 0$

$f(n) = 10^n \sim f(n+1) = 10 f(n), \; f(0) = 1$

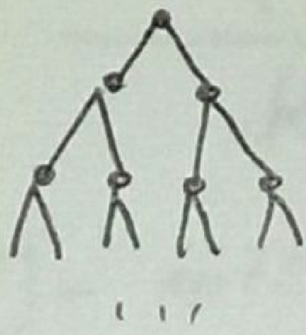
$f(n) = n! \sim f(n) = n\, f(n-1) \quad f(0) = 1$

$$\left. \begin{array}{l} f(n) = f(n-1) + d \\ f(0) = a_0 \end{array} \right\} \text{- arithmetic series}$$

$$\left. \begin{array}{l} f(n) = q \cdot f(n-1) \\ f(0) = a_0 \end{array} \right\} \text{- geometric series}$$

Recursively defined structures

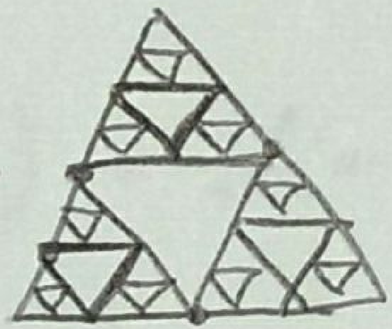
1) Trees:

T = 

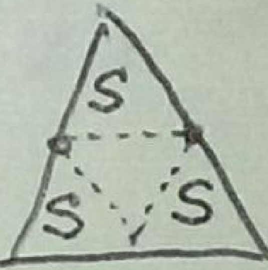
(rooted)
- infinite binary tree

T = 

2) Figures:

S = 

Serpinsky triangle

S = 

4) Numbers:

$$\varphi = 1 + \cfrac{1}{\underbrace{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\dots}}}}_{\sim\varphi}} = 1 + \frac{1}{\varphi}$$

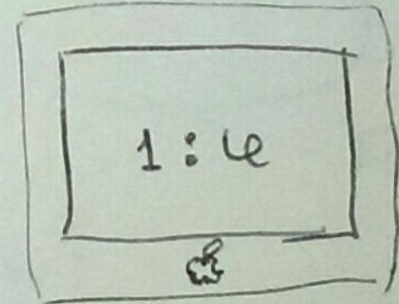
$$\varphi = 1 + \frac{1}{\varphi} \Rightarrow \varphi^2 - \varphi - 1 = 0$$

$$\varphi = \frac{1 \pm \sqrt{1+4}}{2} = \frac{1 \pm \sqrt{5}}{2}$$ – should be non-negative

$$\varphi = \frac{1+\sqrt{5}}{2}$$ – "golden ratio".

$$\approx (1.618)$$

$$\left(\psi = \frac{1-\sqrt{5}}{2}\right)$$



– aesthetically pleasing? :)

$(F_n =) F(n) = F(n-1) + F(n-2)$ – n'th Fibonacci number

$F(0) = 0$

$F(1) = 1$

$F(i)$	0	1	1	2	3	5	8	13	21	...
i	0	1	2	3	4	5	6	7	8	

$$F(n) = \frac{1}{\sqrt{5}} \left[\Big(\underbrace{\frac{1+\sqrt{5}}{2}}_{\varphi} \Big)^n - \Big(\underbrace{\frac{1-\sqrt{5}}{2}}_{\psi} \Big)^n \right]$$

$$= \frac{\varphi^n - \psi^n}{\sqrt{5}}$$

Claim: $F^2(1) + F^2(2) + \dots + F^2(n) = F(n) \cdot F(n+1)$, $n \in \mathbb{Z}^+ = \{1, 2, 3, \dots\}$

Proof: Basis: $n = 1 \Rightarrow LHS = F^2(1) = 1$

$RHS = F(1) \cdot F(2) = 1 \cdot 1 = 1$ ✓

Hypothesis: $F^2(1) + \dots + F^2(n-1) = F(n-1) \cdot F(n)$

OR

Step: $F^2(1) + \dots + F^2(n-1) + F^2(n) \overset{\text{ind hypo}}{=} F(n-1)\,F(n) + F^2(n) =$

$= F(n)\underbrace{\left(F(n-1) + F(n)\right)}_{F(n+1)} = F(n)\,F(n+1)$ $\square$

$\boxed{13}$ **Claim:** $F(1) + F(3) + \dots + F(2n-1) = F(2n)$, $n \geq 1$

Proof: Basis $(n=1)$: $LHS = F(1) = 1$

$RHS = F(2 \cdot 1) = F(2) = 1$ ✓

Hypothesis: $F(1) + F(3) + \dots + F(2n-1) = F(2n)$

Step: $F(1) + F(3) + \dots + F(2n-1) + F(2n+1) = F(2n) + F(2n+1) = F(2n+2)$

$= F(2(n+1))$

Languages:

Σ - alphabet - a finite set of symbols (characters) $\quad (\Sigma = \{0, 1\})$

$w = a_0 a_1 \dots a_{n-1}$ - word over Σ; $\quad |w| = n$

$a_i \in \Sigma$

"word" = "string"

$w_1 w_2$ - words over Σ

$w_1 \cdot w_2 = w_1 w_2$ - concatenation of w_1 and w_2.

$w_1 = a_0 a_1 \dots a_{n-1}$

$w_2 = b_0 b_1 \dots b_{m-1}$ $\quad | \quad w_1 w_2 = a_0 \dots a_{n-1} b_0 \dots b_{m-1}$

$|w_1 w_2| = |w_1| + |w_2|$

$\underbrace{w w \dots w}_{n \text{ times}} = w^n$ $\qquad w = abc$ $\quad \Sigma = \{a, b, c\}$ $\quad w^3 = abc\,abc\,abc$

ε - empty word; $|\varepsilon| = 0$ $\qquad w\varepsilon = w$

$w^0 = \varepsilon$ $\qquad \varepsilon w = w$ $\qquad$ (ε or λ - diff notations)

Language - a set of words over an alphabet.

Ex: $\Sigma = \{0, 1\}$

L - language of all strings (words) over Σ.

Ex: lang of balanced bin. words TODO

[redacted] $\varepsilon \in L$

$w \in L \to w0 \in L$

$w \in L \to w1 \in L$

$L = \{ \varepsilon,$

$\varepsilon 0 = 0, \quad \varepsilon 1 = 1,$

$00, 01, 10, 11,$

$\dots$

$\}$

[redacted]

Reversal:

$r: L \to L$

$r(w)$ = reversed w $\qquad r(01011) = 11010$

[redacted] $r(\varepsilon) = \varepsilon$

$|w| > 0 \quad w =$ [redacted] $m\overset{1 \text{ symb.}}{a}$, $\quad m \in L$

$|w| = n$ $\qquad a (\in \Sigma) \in L$

$w = \underbrace{[\quad m \quad]}_{n-1}\underbrace{[a]}_{1}$

$r(w) = a\, r(m)$

$\boxed{38}$ Recursive definition of [redacted] the set of bit strings that are polyndromes

Basis:

$\varepsilon \in P$

$0 \in P$

$1 \in P$

Step:

$x \in P \rightarrow 0x0 \in P$

$x \in P \rightarrow 1x1 \in P$

Ex: 1011001101

Use language notation

def: $r(\varepsilon) = \varepsilon$, $r(wx) = x\,r(w)$, w - string, x - symbol.

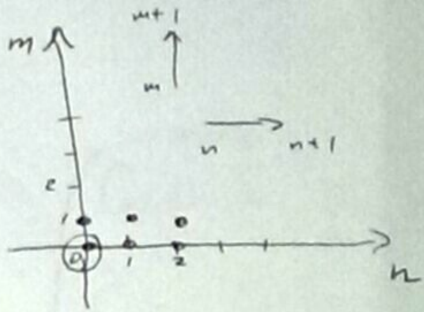
Claim: $r(w_1 w_2) = r(w_2)\,r(w_1)$, $w_1, w_2 \in L$ - some language over some alph Σ

Proof: [redacted]

Basis: [redacted] $|w_1| = 0$, $|w_2| = 0$ $(w_1 = w_2 = \varepsilon)$

$$r(w_1 w_2) = r(\varepsilon) = \varepsilon = r(\varepsilon)\,r(\varepsilon) = r(w_1)\,r(w_2)$$

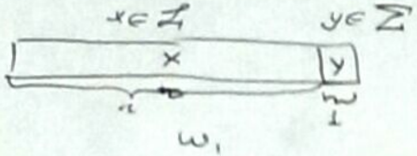
Hypothesis: $\forall w_1, w_2 : (|w_1| \le n \wedge |w_2| \le m) \rightarrow r(w_1 w_2) = r(w_2)\,r(w_1)$



Step: 2) $|w_1| = n+1$, $|w_2| = m$

$n \ge 0$, $m \ge 0$

$x \in L$ $y \in \Sigma$

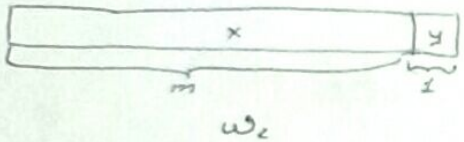


$$r(w_1 w_2) = r(\underbrace{x}_{n}\underbrace{y}_{1}\underbrace{w_2}_{m}) = (1^{st}\text{ step}) =$$

$$= r(y w_2)\,r(x) = r(w_2)\,y\,r(x) = r(w_2)\big(r(y)\,r(x)\big) = (\text{hypo}) = r(w_2)\,r(xy)$$

$$= r(w_2)\,r(w_1)$$

1) $|w_1| = n$, $|w_2| = m+1$



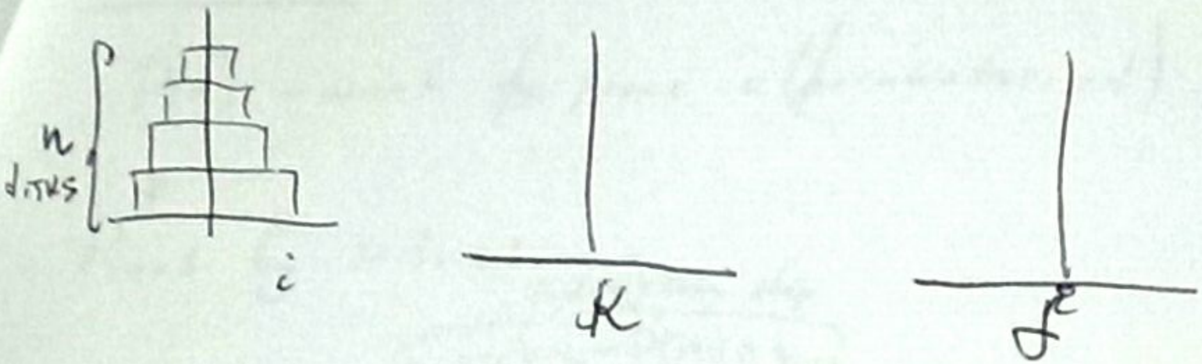
$$r(w_1 w_2) = r(w_1 x \underbrace{y}_{1}) = y\,r(w_1 x) =$$

$$= (\text{hypo}) = y\,r(x)\,r(w_1) = (\text{def of } r) =$$

$$= r(y)\,r(x)\,r(w_1) = (\text{hypo}) =$$

$$= r(xy)\,r(w_1) = r(w_2)\,r(w_1)$$

5) Algorithms:



Hanoi Towers
(simple version:
- no disk repetitions
- only 3 pegs
- $n \geq 3$)

$A(n, i, j)$ - optimally move n top disks from peg i to peg j

$$A(n, i, j) = A(n-1, i, \text{[redacted]}\ k), \ A(1, i, j), \ A(n-1, k, j)$$

(Want $A(n, 1, 3)$) $\qquad A(1, i, j)$ - easy to do.

[redacted]

Costs: $|A(n, i, j)| = |A(n-1)| + |A(n-1)| + |A(1)|$

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

$$\overset{?}{\Rightarrow} T(n) = 2^n - 1.$$ (unroll). and prove by int.

Functions:

Algorithm to reverse the word order in a string:

$f(\text{"one\_two\_three"}) = \text{"three\_two\_one"}$

```
function reverse_word_order (str) {
    str = reverse (str)          | n/2 + c
    for each w in str            | x m
        w = reverse (w)          | (n/m)/2 + c
    return str.
}
```

```
reverse (str) {
    n = length(str)
    for i = 1 to n
        str[i] <-> str[n-i]
}
```

$$T_{rev}(w) = \frac{|w|}{2} + c$$

$$T(S) = T(n, m) = \left(\frac{n}{2} + c\right) + m \times \left(\frac{n/m}{2} + c\right)$$

$|S| = n$

$\#$ of words in $S = m$

$$= \frac{n}{2} + c + \frac{n}{2} + c \cdot m =$$

$$= n + c \cdot m + c \leq n + c \cdot n + c = O(n).$$

Recursive algorithms; algorithm analysis

$F(n)$ - n-th Fibonacci number

```
function fib1(n) {
    // F(n) = (φ^n - ψ^n)/√5 ,  φ,ψ = (1±√5)/2
    [redacted]
    x = pow(φ, n)                    | O(log n) + c1
    y = pow(ψ, n)                    | O(log n) + c1
    result = (x - y)/ [redacted] sqrt(5)   | c2
}
```

$T_{fib1}(n) =$ [redacted]

$O(\log n) + c_1 + O(\log n) + c_1 + c_2$

$= O(\log n).$

```
function pow(x, n) - fast power
    if (n=0) return 1
    if (n-even) return pow(x*x, n/2)
    if (n-odd) return x*pow(x*x, n/2)
}
```

[redacted]

$T_{pow}(n) = c + T_{pow}\left(\frac{n}{2}\right)$

$\Rightarrow T_{pow}(n) =$ [redacted]

$c \cdot \log n = O(\log n).$

```
function fib2(n) {
    // F(n) = F(n-1) + F(n-2)
    if n=0 return 0
    if n=1 return 1                   } c1
    else return fib2(n-1) + fib2(n-2)
}
```

Claim: $T(n) = O(2^n)$

Proof: $T(0) \overset{?}{=} O(1)$

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

$= O(2^{n+1})$

$T_{\cancel{fib2}}(n) = \underbrace{c_1 + c_2}_{c} + \cancel{2}\, T(n-1) + T(n-2)$

$T(0) = T(1) = c_1 = O(1).$

$T(n) = c + T(n-1) + T(n-2) = c + [\cancel{T(n-2)}\ c + T(n-2) + T(n-3)] + [c + T(n-3) + T(n-4)] =$

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

$= (1+2+4)c + (T(n-3) + T(n-4)) + (2T(n-4) + 2T(n-5)) + (T(n-5) + T(n-6)) =$

$= (1+2+4)c + T(n-3) + 3T(n-4) + 3T(n-5) + T(n-6)$

$= \ldots$

$= (1+2+4+\ldots+2^{n-1})c + \ldots = c\cdot(2^n - 1) + \ldots = O(2^n) + \ldots$

~~6464~~ ~10~