Lec 11: ILFP

data BinT a = ET | Node a (BinT a) (BinT a)

1. Property P: # of nodes in BinT $T \leq 2^{h(T)+1} - 1$

where h = height of T

Proof.

Base. For ET, $h(ET) = -1$

By defn. # of nodes (ET) $= 0$

$\leq 2^{h(T)+1} - 1$

≤ 0

I.S. Let $T_1, T_2 \in$ BinT

& $P(T_1), P(T_2)$ hold

If nodes $(T_1) \cap$ nodes $(T_2) = \phi$

& $r \notin$ nodes (T_1), nodes (T_2)

$\Longrightarrow P(T)$ holds when

$T =$ Node r T_1 T_2

· By def^n T_1 & T_2 are disjoint
if they donot contain r, then
by def^n

$$\text{nodes}(T) = |\text{nodes}(T_1) \cup \text{nodes}(T_2)_2 \cup \{r\}|$$
$$= |\text{nodes}(T_1)| + |\text{nodes}(T_2)| + 1$$
$$\leq (\text{By I.H.}) \left(2^{h(T_1)+1} - 1\right) + \left(2^{h(T_2)+1} - 1\right) + 1$$
$$\leq \left(2^{h(T)} - 1\right) \cdot 2 + 1 \quad (\text{By def}^n \text{ of ht.})$$
$$\leq \left(2^{h(T)+1} - 1\right)$$

Lists

data MyLst a = Nil | a : (MyLst a)

P : Length (L ++ M) = Length L + Length M

Defn of Length L =
case L of
[ ] $\longrightarrow$ 0
x : xs $\longrightarrow$ 1 + (Length xs)

Defn of ++
[ ] ++ L = L
(x : xs) ++ L = x : (xs ++ L)

Proof of P

Basis.

Let $L = []$

length ([] ++ M)

Basis [defn of ++]

= length (M)

= 0 + length (M)

Basis [defn of length]

= length ([]) + length (M)

= length L + Length M

I.S.

Assuming length (xs ++ M)

= length (xs) + length M

Let $L = x : xs$

then

$$
\begin{aligned}
\text{length}\,(x:xs) + \text{length}\,M &= \text{length}\,(x:xs) + \text{length}\,M \\
&= 1 + \text{length}\,(xs) + \text{length}\,M && [\text{def}^n \text{ of length}] \\
&= 1 + \text{length}\,(xs ++ M) && [\text{By I.H.}] \\
&= \text{length}\,(x:(xs ++ M)) && [\text{def}^n \text{ of length}] \\
&= \text{length}\,((x:xs) ++ M) && [\text{def}^n \text{ of } ++] \\
&= \text{length}\,(L ++ M)
\end{aligned}
$$

$\square$ QED

Universal Algebra

- Useful in understanding the essence of data types, pattern matching etc.

- The issues of language design & implementation are better exposed in their generality using concepts from universal algebra.

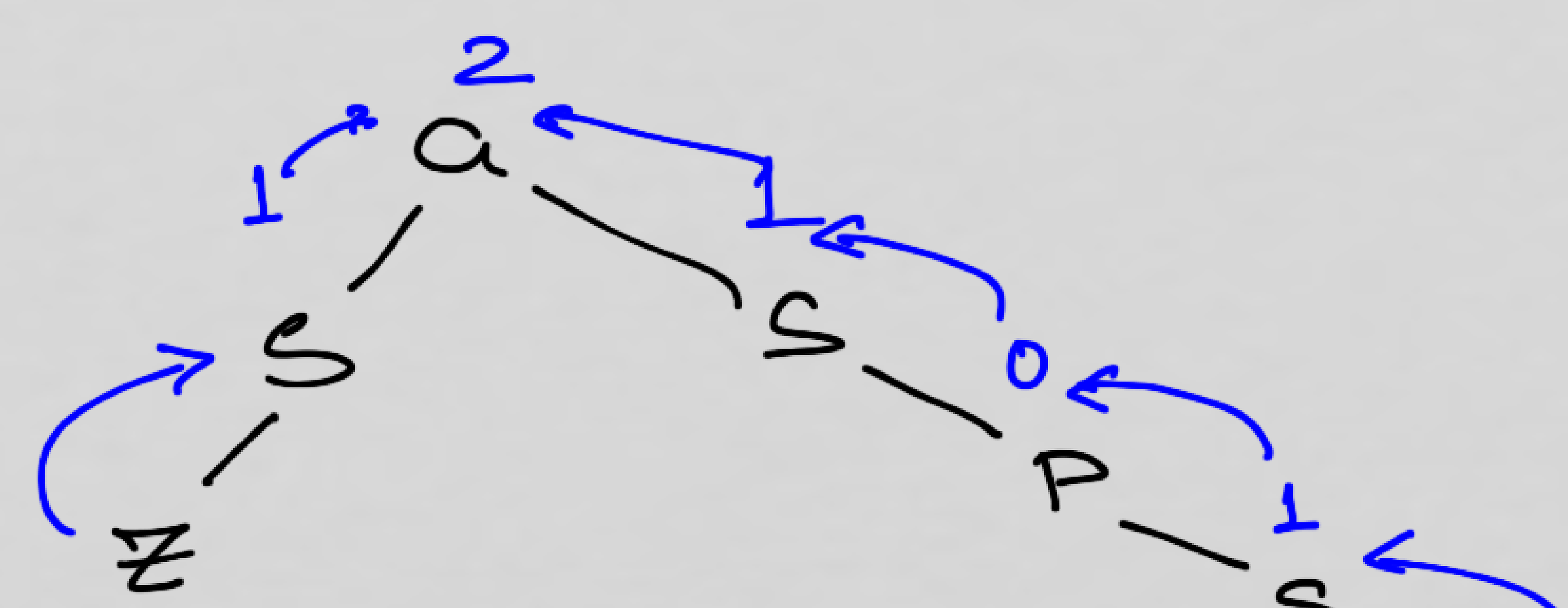
- Enables us to treat syntax & semantics without worrying about scanning & parsing of languages; instead treating them as algebraic systems

- Result of scanning & parsing a program is an AST.
- Program is a term generated by a data-type
- Each term is a linear representation of the syntax tree

Eg: Integer Datatype

data integer = z | S integer | p integer | a integer integer

terms: S z, P z, S(S z), S(P(S z)), a(S z)(S(P(S z)))



AST

Defn 1: Homogeneous or 1-Sorted Algebra

- An ordered pair $(A, \mathcal{F})$ where A is a $\boxed{\text{carrier set}}$, and f is set of operators / constructors / functions s.t. A is closed under $\mathcal{F}$. We refer to A as an algebra when f is understood.
- Operationally, $\forall f \in F$, $f : A^n \longrightarrow A$

1-sorted
because only one carrier set. & is closed over F.

Eg: Integer datatype defines a 1-sorted algebra over z, s, p, a !

Naming arbitrary patterns generated by a recursive data type

- They could be infinite; introduce variables for naming

Eg: general patterns of the form

$p(s(\ldots\ldots))$ may be represented as

$p(s(x))$, $x \in V$ → possibly infinite

Generalisation to many-sorted algebra

Signature : $\Omega = (S, F)$

where

- S is a set of carrier sets
- F is a collection of mappings of the form $(S^* \longrightarrow S)$

1. sorted algebra

structure $\langle A, \Omega \rangle$ is 1-sorted/homogeneous if A is closed under operations in Ω

i.e. Every constant in Ω is present in A

& Every op on Elements of A is a total function whose codomain is A.

Eg :

Homogeneous Algebra ?

$(N, \{+, *\},)$ — ✓

$(N, \{+, *, -\})$ — ✗

$(Z, \{+, *, -\})$ — ✓

Many - sorted Algebra (2-sorted Algebra)

$$\Omega_{BZ} = (\{B, Z\}, \{p, s, \vee, \wedge, \neg, <\})$$

where $<\ : Z \times Z \longrightarrow B$

identities :

$p\,x < x = true$

$x < s\,x = true$

$\vdots$

Terms & Ground terms

- Let Ω be a signature & V be an infinite set of variables s.t. $\Omega \cap V = \phi$
- The set $T_\Omega(V)$ of all Ω-terms over V is defined inductively as the smallest set of terms s.t.

  Basis. $V \subseteq T_\Omega(V)$

  I.S. If $f \in \Omega_n$ for $n \geq 0$ and $t_1, \ldots t_n \in T_\Omega(V)$, then $f(t_1, \ldots t_n) \in T_\Omega(V)$

- The set of ground terms (denoted by T_Ω) is the set of terms without any variables

In BNF

$$s, t \in T_\Omega(V) := x \in V \mid f(t_1, \dots, t_n)$$

Observations

- $T_\Omega(V)$ is never empty
- If Ω contains no constants then $T_\Omega = \emptyset$

set of ops with arity 0.

- Note ~~that~~ a constant $\boxed{c} \in s$ where $s \in S$ is a function symbol of type $\epsilon \to s$, where ϵ denotes an empty cartesian product of S.

Syntactic identity denoted by $\equiv$

<u>Term Algebra</u>

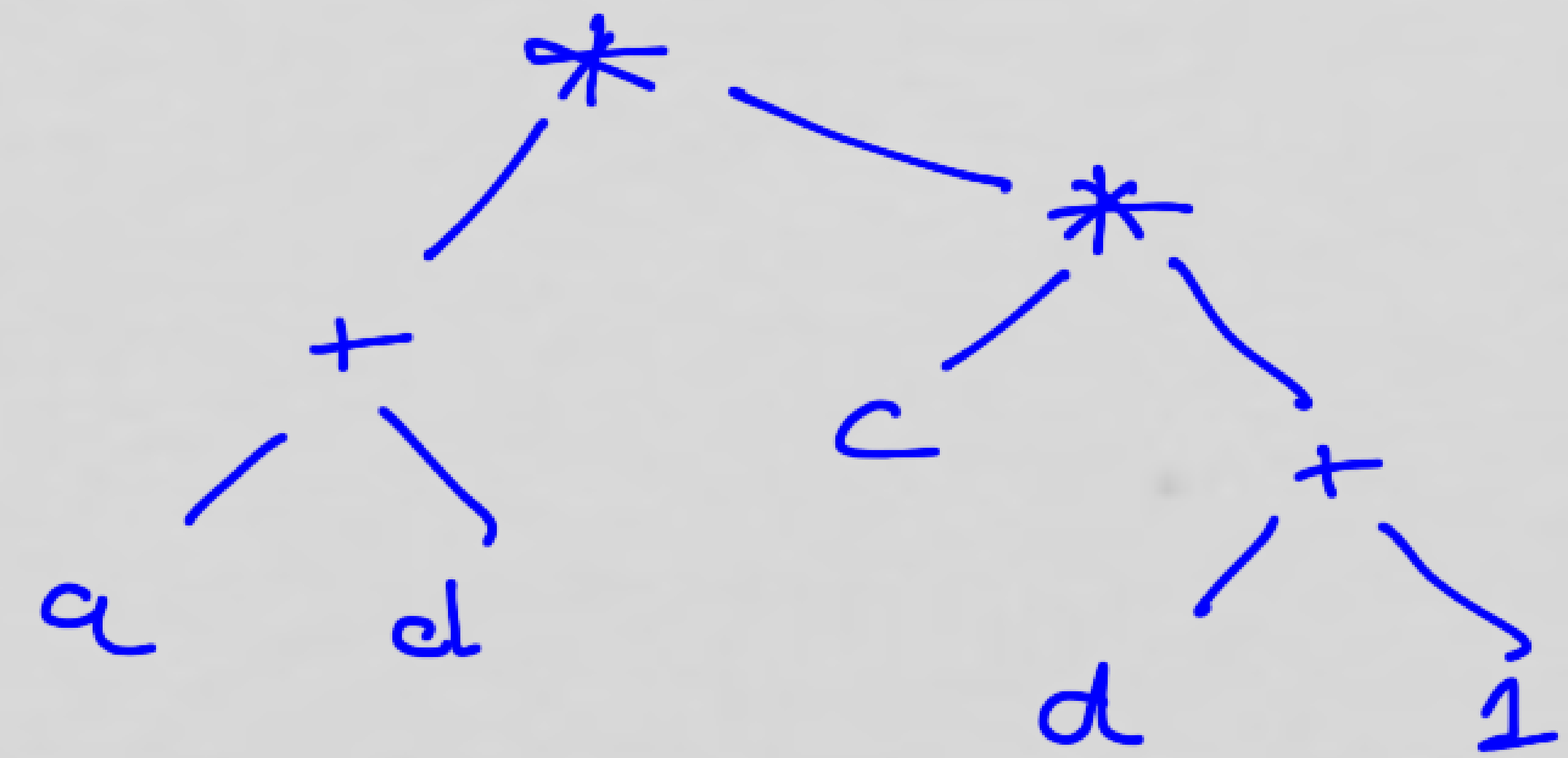
$$(T_\Omega(V), \Omega, \{\equiv\})$$

<u>Eg</u>: $(T_\Omega(V), (\{Z\}, \{+, *, -\}), \{\equiv\})$

↳ Term algebra generated by a set of integers

↳ $(a+d) * (c * (d+1))$

<u>AST</u>



Term Algebra with Op. Schemas

- Let Ω consist of operators (ranged over by o)
- Let Ω also contain operator schemas (ranged over by O)
- Then set $T_\Omega(V)$ can be defined by BNF

$$s, t, u \in T_\Omega(V) ::= x \in V \mid o(t_1, \dots, t_n) \mid O x [t]$$

each t_i is a $\boxed{\text{proper}}$ subterm of $O(t_1, \dots t_n)$

infact every term is a subterm of itself.

- $O x$ is called a binding occurrence of variable x.
- the brackets $[\,,]$ are used to delimit the scope of bound var x, all occurrences of x in the subterm t of the term $O x [t]$ are said to be bound.

Eg of operator schema

$$\sum_{i} (a_i \times b_i)$$

free variables

parameter of operator schema Σ

- linear notation of BNF $\Sigma\, i\, [a_i * b_i]$

Free & Bound Variables

- for any term $s \in T_\Omega(V)$, $FV(s)$ denotes the set of free variables of s & $Var(s)$ denotes the set of all variables (free & bound) that occur in s.

Inductive def^n of functions

s	$FV(s)$	$Vars(s)$
c	$\emptyset$	$\emptyset$
x	$\{x\}$	$\{x\}$
$o(t_1, \ldots t_n)$	$\bigcup_{1 \le i \le n} FV(t_i)$	$\bigcup_{1 \le i \le n} Vars(s)$
$Ox[t]$	$FV(t) - \{x\}$	$Var(t) \cup \{x\}$

Eg: Consider the formula

$$\forall x\,[P(x) \wedge \exists x\,[q(x)]] \vee r(x)$$

- Has 2 distinct bound vars x
- 1 free var. x

referred to as $(x,0)$, $(x,1)$

all three are distinct

Subterm ordering

$s \sqsubseteq t$ iff $s \in ST(t)$.

$\sqsubseteq$ is a partial order

- $\sqsubseteq$ subterm relation
- $\sqsubset$ proper subterm, iff $s \sqsubseteq t \wedge s \not\equiv t$