Mathematics for Computer Science
MIT 6.042J/18.062J

Recursive Definitions
Structural Induction

Recursive Definitions

Define something in terms of a simpler version of the same thing:

- Base case(s) that don't depend on anything else.
- Constructor case(s) that depend on simpler cases.

Example Definition: set E

Define set $E \subseteq \mathbb{Z}$, recursively:

- Base case: $0 \in E$
- Constructor cases:

If $n \in E$, then

1. $n + 2 \in E$, if $n \geq 0$;
2. $-n \in E$, if $n > 0$.

Example Definition: set E

1. $n \in$ E and $n \geq 0$, then $n + 2 \in E$:

0, 0+2, (0+2)+2, ((0+2)+2) +2

0, 2, 4, 6, …

2. $n \in E$ and $n > 0$, then $-n \in E$

–2, –4, –6, …

all even numbers

Recursive Definition: Extremal Clause

So, E *contains* the even integers

Anything Else? No!

- $0 \in E$
- If $n \in E$ and $n \geq 0$, then $n + 2 \in E$
- If $n \in E$ and $n > 0$, then $-n \in E$
- That's All!

Extremal Clause

(Implicit part of definition)

Example Definition: set E

So E is exactly:
The Even Integers

Example: Matched Paren Strings, M

Set of strings, $M \subseteq \{\,),(\,\}^*$

- Base: $\lambda \in M$,
  (the *empty string*)
- Constructor:
  If $s, t \in M$, then
  $$(s)t \in M$$

Example: Matched Paren Strings, M

Lemma: Every s in M has an equal number of $)$'s and $($'s.

Proof by Structural Induction on the definition of M

Example: Matched Paren Strings, M

Lemma: Every s in M has an equal number of $)$'s and $($'s.

Let EQ ::=
{strings with = number of $)$ and $($}

Lemma (restated): $M \subseteq EQ$

Structural Induction on M

Proof:

Hypothesis $P(s) ::= s \in EQ$

Base case: $s = \lambda$. $P(\lambda)$?

0 $)$'s and 0 $($'s. OK

Structural Induction on M
Constructor step

$r = (s)t$. **Assume:** $P(s)$ and $P(t)$

$\#)$ in r = $\#)$ in s + $\#)$ in t + 1

$\#($ in r = $\#($ in s + $\#($ in t + 1

$\therefore$ are =   = by $P(s)$ = by $P(t)$

Structural Induction on M

by structural induction,

$$\forall s \in M.\ s \in EQ$$

QED

The 18.01 Functions, F18

The set F18 of functions on $\mathbb{R}$:

- $\mathrm{Id}_{\mathbb{R}}$, constant functions, and $\sin x$ are in F18.
- if $f, g \in$ F18, then
  - $f+g$, $f \cdot g$, e^f, (the constant e)
  - the inverse, $f^{(-1)}$, of f, and
  - $f \circ g$ (the composition of f and g)

  are in F18.

 February 26, 2007

The 18.01 Functions, F18

Some functions in F18:

$-x = (-1)\cdot x$

$\sqrt{x} = (x^2)^{(-1)}$ ---*inverse*

$\cos x = (1-(\sin x \cdot \sin x))^{1/2}$

$\ln x = (e^x)^{(-1)}$

 February 26, 2007

The 18.01 Functions, F18

Lemma. F18 is *closed under derivative*:

if $f \in$ F18, then $f' \in$ F18.

(Team problem 2)

 February 26, 2007

Recursive Data Types

Arithmetic Expressions

 February 26, 2007

Arithmetic Expressions

Defined recursively as follows:

Base:

- if $n \in \mathbb{N}$, then <`int`, n> $\in$ Aexp
- if $n \in \mathbb{N}$, then <`var`, n> $\in$ Aexp

"tagged" data

 February 26, 2007

Arithmetic Expressions

Constructors:

if $e,f \in$ Aexp, then

1. <`sum`, e,f> $\in$ Aexp
2. <`prod`, e,f> $\in$ Aexp

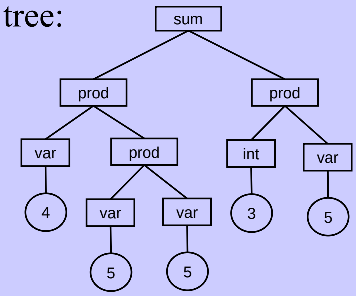
 February 26, 2007

$x_4(x_5)^2 + 3x_5$

<sum, <prod,
<var, 4>,
<prod, <var, 5>, <var, 5>>
>,
<prod, <int, 3>, <var, 5>>
>

$x_4(x_5)^2 + 3x_5$

Parse tree:



Recursive Functions on Aexp

Recursive def. of *size*, $|e|$, of e

$|$<`int`, n>$|$ ::= 1
$|$<`var`, n>$|$ ::= 1
$|$<`sum`, e,f>$|$ ::= $|e| + |f| + 1$
$|$<`prod`, e,f>$|$ ::= $|e| + |f| + 1$

Recursive Functions on Aexp

Recursive def. of *depth*, $d(e)$

d(<`int`, n>) ::= 0
d(<`var`, n>) ::= 0
d(<**sum**, e,f>) ::= $1 + \max\{d(e), d(f)\}$
d(<**prod**,e,f>) ::= $1 + \max\{d(e), d(f)\}$

Size versus Depth

Lemma: $|e| + 1 \leq 2^{d(e)+1}$

Proof by Structural Induction

Base case : e = <int, n> (or <var, n>)

$|e| + 1 = 1+1 = 2 = 2^{0+1} = 2^{d(e)+1}$

OK!

Size versus Depth

Constructor case: e = <**sum**, e_1, e_2>

by ind. hypothesis:

$|e_i| + 1 \leq 2^{d(e_i)+1}$ i=1,2

Size versus Depth

$|e| + 1 = |\langle \texttt{sum}, e_1, e_2 \rangle| + 1$ def. of e

$= (|e_1| + |e_2| + 1) + 1$ def. of size

$= (|e_1| + 1) + (|e_2| + 1)$

$\leq 2^{d(e_1)+1} + 2^{d(e_2)+1}$ induction hyp.

$\leq 2^{\max(d(e_1), d(e_2))+1} + 2^{\max(d(e_1), d(e_2))+1}$

$= 2^{(\max(d(e_1), d(e_2))+1)+1} = 2^{d(e)+1}$ def. of depth

QED

 February 26, 2007 lec 4M-28

Team Problems

Problems 1--3

 February 26, 2007 lec 4M-29