



String Theory

(Computer Science Version)

Robert M. Keller
Harvey Mudd College
March 2010

Why are we doing this?

- Strings are **fundamental** to most formal systems (e.g. grammars).
- Strings **generalize natural numbers** (which are themselves very important).
- Strings are **easy** to work with compared to set-theoretic definitions of natural numbers.

Informal Definition of Strings

- A **string** is just a sequence $x_1x_2\dots x_n$ where $n \geq 0$, with each x_i being a member of some common **alphabet** Σ .
- The elements of a string are called "letters".
- While infinite strings are sometimes used, for now we will be working only with **finite** strings.

Formal Definition of Strings

- A string over the set of letters Σ is any member of the following inductively defined set, named Σ^* :
 - The empty string, Λ , is in Σ^* .
 - If a string x is in Σ^* , and σ is in Σ , then σx is in Σ^* .
 - The only elements of Σ^* are those obtained by applying the above rules.
- By σx we mean a combination of σ with x in such a way that both can be recovered from the result. This can be informally thought of as the "followed by" operator (similar to $[\sigma \mid x]$ in the rex language).

Notes on the Empty String

- The empty string symbol Λ (upper-case lambda) is a "meta symbol" and as such is never an ordinary letter in Σ .
- Other symbols are often seen in place of Λ :
 - λ (lower-case lambda) is used in CS 60.
 - ϵ (lower-case epsilon) used in some texts.

Strings vs. Natural Numbers

- Natural numbers N can be viewed as a special case of strings over a 1-letter alphabet.
- Let say the letter is just '1'.
- Then the connection is suggested by:

N	$\{1\}^*$
0	Λ
1	1
2	11
3	111
4	1111
n	1^n (n 1's in a row)

String Concatenation

- Concatenation means "chaining together", i.e. following one string by another.
- Concatenation will be shown by juxtaposition: xy is y concatenated to x .
- Addition in \mathbb{N} can be viewed as a special case of concatenation in $\{1\}^*$.
- Some texts take concatenation as a primitive operation.

String Concatenation

- Concatenation is a function or binary operator on Σ^* and is defined inductively:
 - $\Lambda y = y$ basis
 - $(\alpha x)y = \alpha(xy)$ induction step
- On the left-hand sides above is the thing we are defining, and on the right how we are defining it.
- This can be seen as analogous to defining *append* in *rex* or *Scheme*.

Why define by induction?

- Definition by induction is **precise**, compared to other alternatives.
- Definition by induction enables **proof** by induction.

String Axioms

- These axioms characterize strings, analogously to the way in which the Peano axioms characterize the natural numbers.
- In the following x, y, \dots are implicitly quantified over strings
- σ, σ', \dots are implicitly quantified over letters.

String Axioms (similar to Peano axioms)

- SA1: $(\forall x) (\forall \sigma) \sigma x \neq \Lambda$
- SA2: $(\forall x, x') (\forall \sigma, \sigma')$
 $\sigma x = \sigma' x'$ implies $\sigma = \sigma'$ and $x = x'$
- SA3(Induction): Let $P(x)$ be any formula with free variable x .

$$(P(\Lambda) \wedge \forall x (P(x) \rightarrow \forall \sigma P(\sigma x))) \rightarrow \forall x P(x)$$

Example of Inductive Proof

- ST1: $(\forall x) x\Lambda = x$, in other words, Λ concatenated to any string is just that string.
- For proof, we identify $P(x)$ in SA3 with $x\Lambda = x$.
- SA3 says that ST1 follows if we can show two things:
 - $P(\Lambda)$, i.e. $\Lambda\Lambda = \Lambda$. basis
 - $(\forall x)(P(x) \Rightarrow (\forall \sigma)P(\sigma x))$,
 i.e. $(\forall x)(x\Lambda = x \Rightarrow (\forall \sigma)(\sigma x)\Lambda = \sigma x)$ induction step
- How are these two statements shown?

Basis

- ST1': $(\forall x) \Lambda x = x$
- This in itself requires a secondary induction proof. Let $Q(x)$ be $\Lambda x = x$.
- Basis: $Q(\Lambda)$: $\Lambda \Lambda = \Lambda$

Another Example of Inductive Proof

- ST2: $(\forall x, y, z) x(yz) = (xy)z$. In other words, concatenation is associative.
- Here we identify $P(x)$ in SA4 with $x(yz) = (xy)z$.
- SA4 says that ST2 follows if we can show two things:
 - $P(\Lambda)$, i.e. $\Lambda(yz) = (\Lambda y)z$. basis
 - $(\forall x)(P(x) \Rightarrow (\forall \alpha)P(\alpha x))$, induction step
i.e. $(\forall x)(x(yz) = (xy)z \Rightarrow (\forall \alpha)(\alpha x)(yz) = ((\alpha x)y)z$
- How are these two statements shown?

Defining String Reversal

- We want to define inductively reversal (R operator)
- Here's an intuitively-correct idea:
 - $\Lambda^R = \Lambda$ basis
 - $(\alpha x)^R = x^R(\alpha \Lambda)$ induction step
- Note: We have to use $\alpha \Lambda$ rather than just α because $x^R \alpha$ is not defined. But since $\alpha \Lambda$ is a string, we can concatenate.
- Note that we are using concatenation to define this operator. This could be considered somewhat heavy-handed.

Defining String Reversal

- Another option would be to use an auxiliary function r :
 - $x^R = r(x, \Lambda)$, where
 - $r(\Lambda, y) = y$ basis
 - $r(\alpha x, y) = r(x, \alpha y)$ induction step
- Here we haven't used any overt concatenation, and this makes us feel better.
- Note that this is the familiar rex definition in disguise.
- We could now try to prove that the two versions of reversal are equivalent. We'll table this.

Proving stuff about reversal (R operator)

- ST3: $(xy)^R = y^R x^R$
 - Try induction with $P(x)$: $(\forall y)(xy)^R = y^R x^R$
 - Basis: $P(\Lambda)$: $(\forall y)(\Lambda y)^R = y^R \Lambda^R$
 - Induction step: $(\forall y)(xy)^R = y^R x^R$
implies $(\forall \alpha)(\forall y)((\alpha x)y)^R = y^R (\alpha x)^R$
 - LHS of = is: $((\alpha x)y)^R$
 - $= (\alpha(xy))^R$
 - $= (xy)^R (\alpha \Lambda)$
 - $= (y^R x^R) (\alpha \Lambda)$
 - RHS of = is: $y^R (\alpha x)^R$
 - $= y^R (x^R (\alpha \Lambda))$
 - $= (y^R x^R) (\alpha \Lambda)$
- } Justify each step

Free the Monoids!

- Algebraically, Σ^* is known as the "free monoid generated by Σ ".
(usually serves to deter the casual reader from going any further)
- Recall that a monoid is a set together with:
 - An associative operator
 - An identity for the operator
- Identify the components that justify calling Σ^* a monoid.

Natural Numbers

- Pick Σ to be any 1-letter alphabet. (e.g. use \emptyset as a letter).
- Then Σ^* is essentially the set of natural numbers:
 - Λ is like 0
 - αx is like $x+1$
- The string axioms are then equivalent to the Peano axioms.
- The induction rule is the usual mathematical induction principle.

Natural Numbers

- Addition is just concatenation over Σ^* .
- What are subtraction, multiplication, etc.?
- You'd need to define them inductively: more on this later.

Natural Numbers from Zip

- 0 is \emptyset basis
- $n+1$ is $n \cup \{n\}$ induction step

for example

- 1 is $\emptyset \cup \{\emptyset\}$ which is $\{\emptyset\}$ which has 1 element
- 2 is $\{\emptyset\} \cup \{\{\emptyset\}\}$ which is $\{\emptyset, \{\emptyset\}\}$ which has 2 elements
- 3 is $\{\emptyset, \{\emptyset\}\} \cup \{\{\emptyset, \{\emptyset\}\}\}$ which is $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$ which has 3 elements, etc.
- ...
- Let ω designate the set of natural numbers.

The Length $| \cdot |$ of a String

- $|\Lambda| = 0$ basis
- $|\alpha x| = |x| + 1$ induction step

Some Properties of Length

- L1: $|xy| = |x| + |y|$
- L2: $|x^R| = |x|$