

### Numerical Representation of Strings

First, we define two primitive recursive functions

$$R^+(x, y) = \begin{cases} R(x, y) & \text{if } \sim (y \mid x) \\ y & \text{otherwise} \end{cases}$$

$$Q^+(x, y) = \begin{cases} \lfloor x/y \rfloor & \text{if } \sim (y \mid x) \\ \lfloor x/y \rfloor - 1 & \text{otherwise} \end{cases}$$

where  $R(x, y)$  and  $\lfloor x/y \rfloor$  are defined as in Section 3.7.

Basically,  $R^+$  and  $Q^+$  are the "usual" remainder and quotient functions, except that remainders are now in the range between 1 and  $y$  instead of 0 and  $y-1$ .

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

1

### Numerical Representation of Strings

So whenever  $y$  divides  $x$ , we do not have a remainder of 0 but a remainder of  $y$ , and accordingly the quotient is one number below the "actual" quotient.

Therefore, like with the usual quotient and remainder, it is still true that:

$$x/y = Q^+(x, y) + R^+(x, y)/y,$$

only that now we have  $1 \leq R^+(x, y) \leq y$ .

We will use the functions  $Q^+$  and  $R^+$  to show how to obtain the subscripts  $i_0, i_1, \dots, i_k$  from any integer  $x > 0$ .

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

2

### Numerical Representation of Strings

Let us define:

$$u_0 = x$$

$$u_{m+1} = Q^+(u_m, n)$$

Then we have:

$$u_0 = i_k \cdot n^k + i_{k-1} \cdot n^{k-1} + \dots + i_1 \cdot n^1 + i_0$$

$$u_1 = i_k \cdot n^{k-1} + i_{k-1} \cdot n^{k-2} + \dots + i_1$$

:

$$u_k = i_k$$

The "remainders"  $R^+$  are exactly the values of the  $i_m$ :

$$i_m = R^+(u_m, n), m = 0, \dots, k.$$

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

3

### Numerical Representation of Strings

This is analogous to our usual base- $n$  notation:

$$u_0 = x$$

$$u_{m+1} = Q(u_m, n)$$

Then we have:

$$u_0 = i_k \cdot n^k + i_{k-1} \cdot n^{k-1} + \dots + i_1 \cdot n^1 + i_0$$

$$u_1 = i_k \cdot n^{k-1} + i_{k-1} \cdot n^{k-2} + \dots + i_1$$

:

$$u_k = i_k$$

The remainders  $R$  are exactly the values of the  $i_m$ :

$$i_m = R(u_m, n), m = 0, \dots, k.$$

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

4

### Numerical Representation of Strings

**Example:** Find binary representation of number 13:

Then  $u_0 = x = 13$ ;  $n = 2$

$$u_1 = Q(13, 2) = 6; i_0 = R(13, 2) = 1$$

$$u_2 = Q(6, 2) = 3; i_1 = R(6, 2) = 0$$

$$u_3 = Q(3, 2) = 1; i_2 = R(3, 2) = 1$$

$$u_4 = Q(1, 2) = 0; i_3 = R(1, 2) = 1$$

Then  $w = 1101$ .

Thus  $k = 3$  and we have

$$x = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$$

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

5

### Numerical Representation of Strings

You certainly noticed that in our string representation we used symbols  $s_1, \dots, s_n$ , while in the everyday number representation we use  $s_0, \dots, s_{n-1}$ .

So what exactly are the analogies and differences between the two systems?

To find out about this, let us look at the following modified set of digits  $D$ :

$$D = \{s_1, \dots, s_{10}\} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, X\}$$

Here, the  $X$  stands for a digit with the value 10, while there is no digit with the value 0.

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

6

### Numerical Representation of Strings

So what is the number  $x$  associated with the string  $w = 76$  ?  
 $x = 7 \cdot 10 + 6 = 76$

And what is the number for  $w = 3X6$  ?  
 $x = 3 \cdot 100 + 10 \cdot 10 + 6 = 406$

Finally, what is the number for  $w = XX$  ?  
 $x = 10 \cdot 10 + 10 = 110$

These examples already suggest that we can use this system in a way quite similar to our "usual" system.

November 12, 2009      Theory of Computation  
Lecture 17: Calculations on Strings II      7

### Numerical Representation of Strings

Now let us turn this around: What is the string  $w$  associated with the number  $x = 39$  ?  
 $w = 39$  (as long as  $x$  does not contain any 0s,  $w$  is the "usual" decimal string representing  $x$ )

And what is the string for  $x = 100$  ?  
 $w = 9X$

But what is the string for  $x = 504$  ?  
 $w = 4X4$

Finally, what is the string for  $x = 0$  ?  
 $w = 0$  (0 is the null string symbol)

November 12, 2009      Theory of Computation  
Lecture 17: Calculations on Strings II      8

### Numerical Representation of Strings

We can even transfer our elementary arithmetic to the new system:

$\begin{array}{r} X4 \\ + 596 \\ \hline 69X \end{array}$	corresponding to	$\begin{array}{r} 104 \\ + 596 \\ \hline 700 \end{array}$
$\begin{array}{r} X23 \\ - X1X \\ \hline 3 \end{array}$	corresponding to	$\begin{array}{r} 1023 \\ - 1020 \\ \hline 3 \end{array}$

November 12, 2009      Theory of Computation  
Lecture 17: Calculations on Strings II      9

### Numerical Representation of Strings

This applies even to multiplication:

$\begin{array}{r} 3X \\ \times X7 \\ \hline 27X \\ 39X \\ \hline 427X \end{array}$	corresponding to	$\begin{array}{r} 40 \\ \times 107 \\ \hline 4280 \end{array}$
--	------------------	--

November 12, 2009      Theory of Computation  
Lecture 17: Calculations on Strings II      10

### Numerical Representation of Strings

So you see that we can easily replace our usual digit range from 0 to  $n-1$  with the range 1 to  $n$ .

In our representation of strings on the alphabet  $A$ , we use the range from 1 to  $n$  to have a bijective mapping between strings and numbers.

When using a range from 0 to  $n-1$ , there is no such mapping, because different strings correspond to the same number, for example:

$87 = 087 = 0087 = 00087 = \dots$

November 12, 2009      Theory of Computation  
Lecture 17: Calculations on Strings II      11

### Numerical Representation of Strings

Now let us return to our general case of associating numbers with strings.

For an alphabet  $A = \{s_1, \dots, s_n\}$ , the string  $w = s_{i_k} s_{i_{k-1}} \dots s_{i_1} s_{i_0}$  is called the **base  $n$  notation** for the number  $x$  as defined by

$$x = i_k \cdot n^k + i_{k-1} \cdot n^{k-1} + \dots + i_1 \cdot n^1 + i_0 .$$

When  $n$  is fixed, we can consider a function of one or more variables on  $A^*$  as a function of the corresponding numbers.

November 12, 2009      Theory of Computation  
Lecture 17: Calculations on Strings II      12

### Numerical Representation of Strings

So we can speak of an  $m$ -ary partial function on  $A^*$  with values in  $A^*$  as being **partially computable**, or when it is total, we can speak of it as being **computable**.

Similarly, we can say that an  $m$ -ary function on  $A^*$  is **primitive recursive**.

But what about predicates?

Notice that for an alphabet  $A = \{s_1, \dots, s_n\}$  the value  $s_1$  denotes 1 in base  $n$  notation.

Thus an  $m$ -ary predicate on  $A^*$  is simply a total  $m$ -ary function on  $A^*$  whose values are either  $s_1$  ("true") or 0 ("false").

Therefore, it makes sense to speak of an  $m$ -ary predicate on  $A^*$  as being computable.

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

13

### Numerical Representation of Strings

For a given alphabet  $A$ , any subset of  $A^*$  (any set of strings on  $A$ ) is called a **language** on  $A$ .

By associating numbers with the elements of  $A^*$ , we can speak of a language on  $A$  as being **r.e.**, **recursive**, or **primitive recursive**.

Notice that our base  $n$  notation even works for  $n = 1$ , that is, an alphabet containing only one symbol.

For example, if  $A = \{s_1\}$  then

$x = 1$  is represented by the string  $w = s_1$

$x = 7$  is represented by the string  $w = s_1s_1s_1s_1s_1s_1s_1$

:

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

14

### Numerical Representation of Strings

To avoid confusion, we will not use decimal digits as symbols in our alphabets.

Accordingly, a string of decimal digits will always be meant to refer to a number.

We will now define some useful functions for string operations.

The first function will be CONCAT.

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

15

### Numerical Representation of Strings

Let  $A$  be some fixed alphabet containing  $n$  symbols,  $A = \{s_1, \dots, s_n\}$ .

For each  $m \geq 1$ , we define  $\text{CONCAT}_n^{(m)}$  as follows:

$\text{CONCAT}_n^{(1)}(u) = u$

$\text{CONCAT}_n^{(m+1)}(u_1, \dots, u_m, u_{m+1}) = zu_{m+1}$ ,

where  $z = \text{CONCAT}_n^{(m)}(u_1, \dots, u_m)$ .

This means that for given strings  $u_1, \dots, u_m \in A^*$ ,  $\text{CONCAT}_n^{(m)}(u_1, \dots, u_m)$  is obtained by concatenating the strings  $u_1, \dots, u_m$ .

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

16

### Numerical Representation of Strings

We will usually omit the superscript, i.e., the number of arguments. For example:

$\text{CONCAT}_3(s_3s_1s_2, s_1s_3) = s_3s_1s_2s_1s_3$

Translating this equation from strings to numbers:

$\text{CONCAT}_3(32, 6) = 294$

It is also true that

$\text{CONCAT}_5(s_3s_1s_2, s_1s_3) = s_3s_1s_2s_1s_3$

Translating this equation from strings to numbers:

$\text{CONCAT}_5(82, 8) = 2058$

Obviously, the definition of CONCAT depends on the base  $n$ .

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

17

### Numerical Representation of Strings

We will now look at several **primitive recursive functions on strings**.

These functions will be useful for our further discussion of calculation on strings.

It will be helpful to remember the functions  $g$  and  $h$ :

$g(0, n, x) = x$

$g(m+1, n, x) = Q^+(g(m, n, x), n)$ ,

then  $g(m, n, x) = u_m$ .

$h(m, n, x) = R^+(g(m, n, x), n)$ ,

then  $i_m = h(m, n, x)$ ,  $m = 0, \dots, k$ .

November 12, 2009

Theory of Computation  
Lecture 17: Calculations on Strings II

18