Strings

- When all the sets in a Cartesian product are the same, we write A^n rather than $A \times A \times \ldots \times A$ (n times) and their elements are called *strings of length n.*

- We are often interested in the set of strings over A that are of arbitrary positive length, which we denote by A^+.

- A string of length 0 is an empty sequence, which we usually denote by ε; we define $A^0 = \{\varepsilon\}$.

- A^* is the set of all strings over A including the empty string.

Bit Strings

- A particularly interesting set is $\mathcal{B} = \{0, 1\}$ – of 'bits'.
  The elements of $\mathcal{B}^*$ are called *bit strings*.

- Bit strings are very useful for representing information, encoding it in a way that it can be processed by computers!

- An example is the way we can use bit strings to represent subsets of a given set.

Bit Strings

- Let S be a finite set of cardinality n (i.e., $|S|=n$) and let $s_1, s_2, \ldots, s_n$ be its elements in a particular order.

- Let A be a subset of S. We can define the bit string $b_A = (b_1, b_2, \ldots, b_n)$ by putting, for every $i=1,\ldots,n$

$$b_i = 1 \text{ iff } s_i \in A$$

- The bit string b_A is the characteristic vector of A.

- Likewise, every bit string b defines a subset of S – $\{s_i \in S \mid b_i = 1\}$.

Bit Strings

- For example, if $S = \{1, 3, 5, 7, 9, 11\}$ *in ascending order*
  - $b_{\{3,7\}} = (0, 1, 0, 1, 0, 0)$
  - $b_{\{1,7,9\}} = (1, 0, 0, 1, 1, 0)$
- Bit strings give us a convenient representation for defining algorithms that implement operations on sets.

Bit Strings

- To find $b_{A \cap B}$ we can use
  - ```
    FOR i=1 TO n DO
    bA∩B[i] = bA[i] * bB[i]
    ```
- For example, in the case where
  - $S = \{1, 3, 5, 7, 9, 11\}$
  - $b_{\{3, 7\}} = (0, 1, 0, 1, 0, 0)$
  - $b_{\{1, 7, 9\}} = (1, 0, 0, 1, 1, 0)$

we have $b_{\{3, 7\} \cap \{1, 7, 9\}} = (0, 0, 0, 1, 0, 0)$
    ```

Bit Strings

- To find $b_{A \cap B}$ we can use
  - `FOR i=1 TO n DO`
    `b`$_{A\cap B}$`[i] = b`$_A$`[i] * b`$_B$`[i]`

How do we know that this program is correct?

- For example, in the case where
  - $S = \{1, 3, 5, 7, 9, 11\}$
  - $b_{\{3, 7\}} = (0, 1, 0, 1, 0, 0)$
  - $b_{\{1, 7, 9\}} = (1, 0, 0, 1, 1, 0)$

we have $b_{\{3, 7\} \cap \{1, 7, 9\}} = (0, 0, 0, 1, 0, 0)$

Bit Strings

- Correctness of
  - `FOR i=1 TO n DO`
    $b_{A\cap B}$`[i] = `b_A`[i] * `b_B`[i]`

Requirement specification

- We need to prove that $b_{A\cap B}[i] = 1$ iff $s_i \in A \cap B$

- $b_{A\cap B}[i] = 1$ iff $b_A[i] * b_B[i] = 1$

  iff $b_A[i] = 1$ and $b_B[i] = 1$

  iff $s_i \in A$ and $s_i \in B$

  iff $s_i \in A \cap B$

José Fiadeiro – 2013/14

Bit Strings

How do we know that this program is correct?

- To find $b_{A \cup B}$ we can use
  - FOR i=1 TO n DO
    IF b_A[i] THEN $b_{A \cup B}$[i] = 1
    ELSE IF b_B[i] THEN $b_{A \cup B}$[i] = 1
    ELSE $b_{A \cup B}$[i] = 0

- For example, in the previous case

$$b_{\{3,7\} \cup \{1,7,9\}} = (1, 1, 0, 1, 1, 0)$$

Bit Strings

- Correctness of
  - ```
    FOR i=1 To n DO
       IF bA[i] THEN bA∪B[i] = 1
       ELSE IF bB[i] THEN bA∪B[i] = 1
            ELSE bA∪B[i] = 0
    ```
  - We need to prove that $b_{A\cup B}[i] = 1$ iff $s_i \in A \cup B$
  - $b_{A\cup B}[i] = 1$ iff $b_A[i] = 1$ or $(b_A[i] = 0$ and $b_B[i] = 1)$

    iff $(b_A[i] = 1$ or $b_A[i] = 0)$ and $(b_A[i] = 1$ or $b_B[i] = 1)$

    iff $(b_A[i] = 1$ or $b_B[i] = 1)$

    iff $s_i \in A$ or $s_i \in B$

    iff $s_i \in A \cup B$

Requirement specification
    ```

Infinite strings

- What if the set S is infinite?
  - We cannot work with an infinite product $\mathcal{B} \times \mathcal{B} \times \ldots$
- When S is finite, a bit string $(b_1, b_2, \ldots, b_n)$ maps every $s_i \in S$ to an element of $\mathcal{B}$.
- This can be generalised to the case where S is infinite by defining a mapping from S to $\mathcal{B}$.
  - Each mapping f defines the subset $\{s \in S \mid f(s) = 1\}$
  - And vice versa, each subset A of S defines the mapping $f_A(s) = 1$ *iff* $s \in A$

Infinite strings

- For example,
  - The bit mapping on $\mathcal{N}$ defined by *is_even(n) = 1 iff n is even* defines the subset of even numbers.
  - As a subset of $\mathcal{Z}$, $\mathcal{N}$ is defined by *is_nat(z) = 1 iff $z \geq 0$.*
- Therefore, there are as many subsets of a set *S* as there are mappings from *S* to $\mathcal{B}$.
  - There is a one-to-one correspondence between 2^S and the set of all mappings from *S* to $\mathcal{B}$.
- What is a *mapping* ? And a *one-to-one correspondence*?