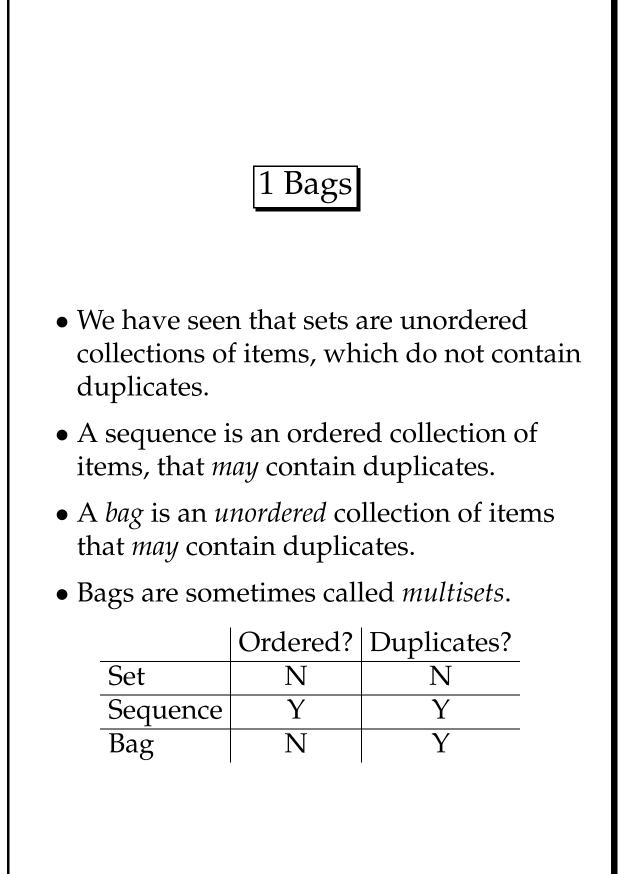
### LECTURE 14: BAGS (MULTISETS)

Software Engineering Mike Wooldridge





• **Definition:** The set of all bags over type *T* is given by the expression

 $\operatorname{bag} T.$ 

• Like sets and sequences, bags may be enumerated, by listing theeir contents between *Strachey brackets*: [].

```
• EXAMPLE. Suppose
```

```
B : bag \mathbb{N}
```

then

 $B == [\![1,1,2,3]\!]$ 

assigns to *B* the bag containing the value 1 twice, the value 2 once, and the value 3 once.

Note that this is *not* the same as the set

 $\{1, 2, 3\}.$ 

However,

 $\llbracket 1, 1, 2, 3 \rrbracket = \llbracket 1, 2, 3, 1 \rrbracket.$ 

## 2 Bag Membership and Sub-bags

 The 'equivalent' of the set membership predicate ∈ is 'in'.

(This is sometime written ' '.)

- **Definition:** If
  - B : bag T
  - x:T

then the predicate

x in B

is true iff *x* appears in *B* at least once.

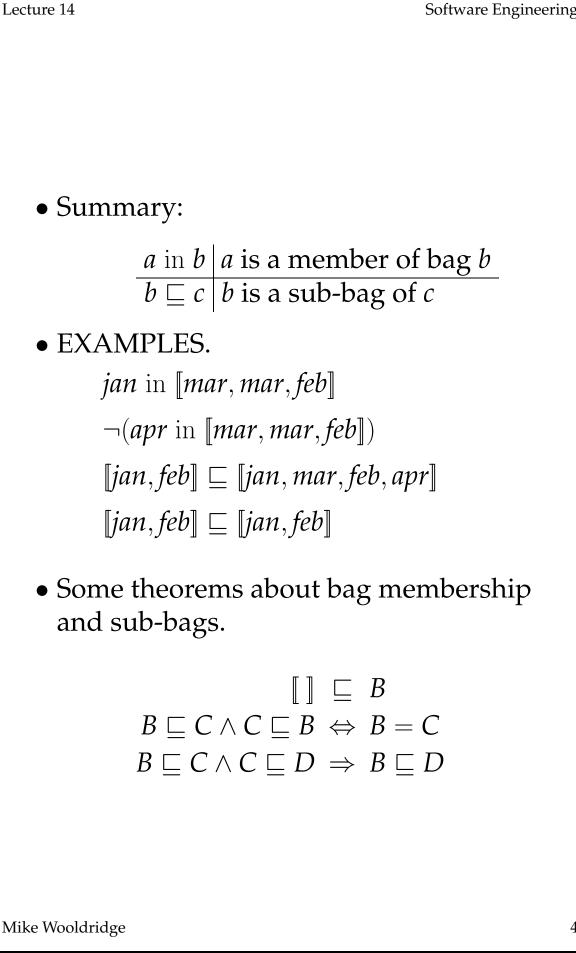
- The 'equivalent' of the subset predicate ⊂ is ⊑.
- **Definition:** If

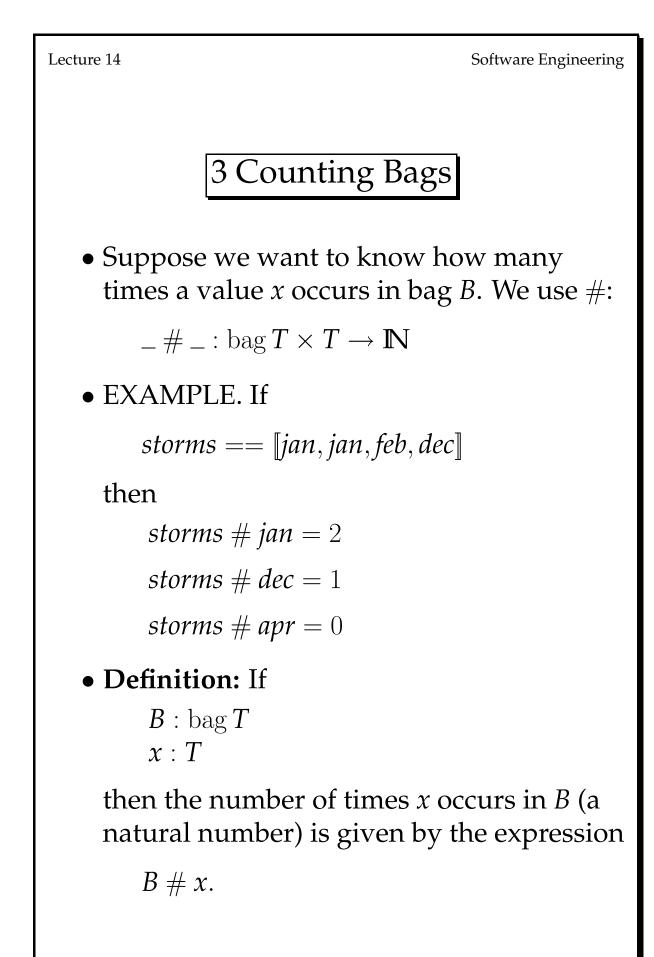
 $B_1, B_2$ : bag T

then the predicate

 $B_1 \sqsubseteq B_2$ 

is true iff each element that occurs in  $B_1$  occurs in  $B_1$  no more often than it occurs in  $B_2$ .





# 4 Scaling Bags

- Another common operation we want to do is *scale* bags; that is, we want to 'multiply' their contents. We do this using the bag scaling operator: ⊗.
- EXAMPLE. Let

*storms* == [*jan*, *jan*, *feb*]

then

```
2 \otimes storms = 
[jan, jan, jan, jan, feb, feb]].
```

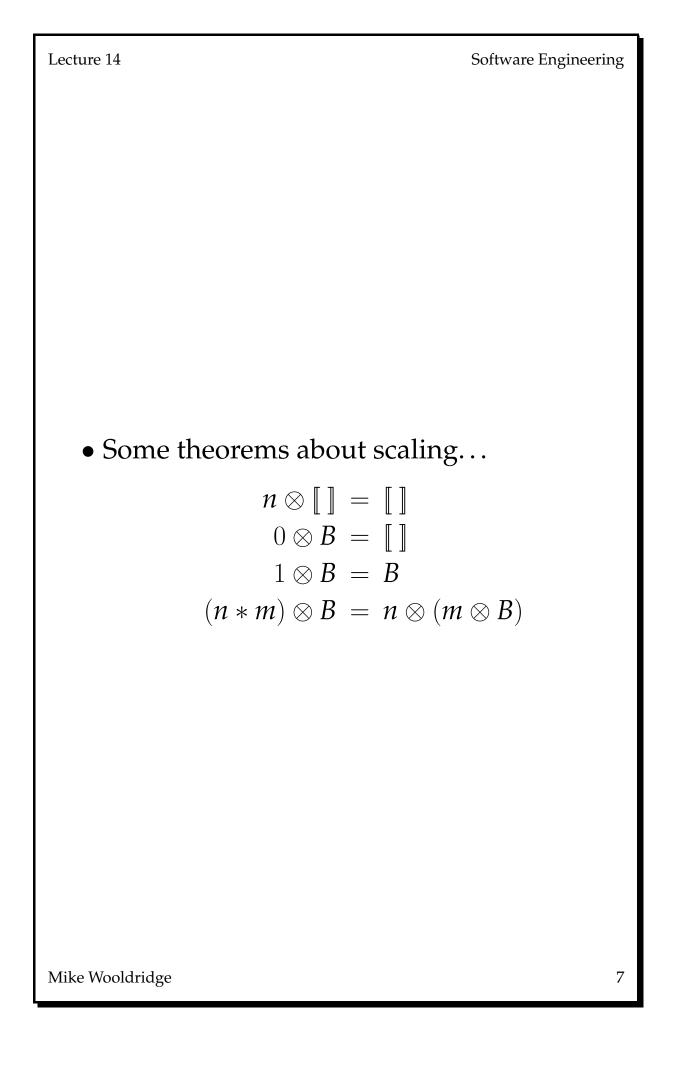
#### • **Definition:** If

B: bag Tn:  $\mathbb{N}$ 

then

 $n \otimes B$ 

is a bag which contains the same elements as *B*, except that every element that occurs *m* times in *B* occurs n \* m times in  $n \otimes B$ .





Software Engineering

### 5 Bag Union

- Just as there is a set union operator, so there is a bag union operator.
- EXAMPLE. Let

$$storms == [jan, jan, feb]$$

then

 $storms \uplus \llbracket mar \rrbracket = \llbracket jan, jan, feb, mar \rrbracket$  $storms \uplus \llbracket jan \rrbracket = \llbracket jan, jan, jan, feb \rrbracket$ 

#### • **Definition:** If

 $B_1, B_2$ : bag T

then

 $B_1 \uplus B_2$ 

is bag that contains just those values that occur in either  $B_1$  or  $B_2$ , except that the number of times a value x occurs in  $B_1 \uplus B_2$ is equal to  $(B_1 \# x) + (B_2 \# x)$ .

## 6 Making Bags out of Sequences

• One last thing we often want to do is to make a bag out of a sequence, by counting up all number of times in a sequence. We do this using *items*.

#### EXAMPLE.

 $items \langle a, b, a, b, c \rangle = \llbracket a, a, b, b, c \rrbracket$  $items \langle a, c, d, a, a \rangle = \llbracket a, a, a, c, d \rrbracket$ 

• **Definition:** If

 $\sigma: \operatorname{seq} T$ 

then  $items(\sigma)$  is a bag over T such that a value x occurs in  $items(\sigma)$  exactly as many times as it appears in  $\sigma$ .

Lecture 14

# 7 A Model for Bags

• In the previous lecture, we saw that sequences are defined in terms of functions.

$$\langle a, b, c \rangle = \{1 \mapsto a, 2 \mapsto b, 3 \mapsto c\}.$$

Bags are defined in a similar way:

 $\operatorname{bag} T == T \twoheadrightarrow \mathbb{N}_1$ 

• So the bag

[*jan*, *feb*, *jan*]

is really the function

 $\{jan \mapsto 2, feb \mapsto 1\}.$ 

- So we can use all the function manipuating operations to manipulate bags.
- In particular:

 $\operatorname{dom}\llbracket a_1,\ldots,a_n\rrbracket = \{a_1,\ldots,a_n\}$ 

and so

dom $[jan, jan, feb] = \{jan, feb\}.$ 

• Taking the range of a bag is not generally as useful.

Mike Wooldridge

10

Software Engineering

- QUESTION: If bags are defined in this way, then how do we define all the operations on them?
- The difficult one is *#*; given this, the others are all more or less easy...
- First,  $\otimes$ :

$$a \mapsto (n * m) \in (B \otimes m) \Leftrightarrow a \mapsto n \in B$$

• Now, 'in':

 $x \text{ in } B \Leftrightarrow (B \# x) > 0$ 

• The  $\sqsubseteq$  predicate is a bit more complicated.

$$\begin{array}{l} \forall B_1, B_2 : \text{bag } T \bullet \\ B_1 \sqsubseteq B_2 \Leftrightarrow \\ \forall x : T \bullet B_1 \# x \leq B_2 \# x \end{array}$$