## Subject: Duality of quantifiers

So, you’ve got an XML document containing a bunch of integers:

<Test>

<x>2</x>

<x>4</x>

<x>5</x>

<x>7</x>

<x>10</x>  
 …

</Test>

You want to know: Is it true that not all the integers are even?

To find the answer, you create this XPath expression:

not (every $x in /Test/\* satisfies $x mod 2 eq 0)

“Is it not the case that every x in Test is even?”

Alternatively, you could have written the XPath expression this way:

some $x in /Test/\* satisfies not ($x mod 2 eq 0)

“Are there some x in Test that are not even?”

The two XPath expressions are equivalent – they produce the same results.

Let’s denote the test for evenness ($x mod 2 eq 0) with p(x), where p represents the evenness test on x.

Let’s denote the set of values being evaluated ( /Test/\* ) with seq.

Then the two XPath expressions can be written:

not (every $x in seq satisfies p(x))

some $x in seq satisfies not p(x)

Using a slightly more abstract (non-XPath) notation, we can write the expressions as:

not (for all x) p(x)

(for some x) [not p(x)]

The two expressions are equivalent.

“*for all*” and “*for some*” are called *quantifiers*. In particular, “for all” is called the *universal quantifier* and “for some” is called the *existential quantifier*.

The equivalence of these two expressions:

not (for all x) p(x)

(for some x) [not p(x)]

is called the “*duality of quantifiers*”.

Here are two more equivalent expressions, exhibiting the duality of quantifiers:

not (for some x) p(x)

(for all x) [not p(x)]

Converting them to XPath, we have:

not (some $x in /Test/\* satisfies $x mod 2 eq 0)

every $x in /Test/\* satisfies not ($x mod 2 eq 0)

Both express this: “Is it the case that none of the x values are even?”

## Alloy

The following signature represents an XML document containing a list of integers:

**one** **sig** Test {  
 contents: **set** **Int**  
}

The following predicate determines if an integer parameter x is even:

**pred** isEven [x: **Int**] {  
 rem [x, 2] = 0  
}

The following Alloy check command instructs the Alloy Analyzer to determine if the expression using the universal quantifier is equivalent to the expression using the existential quantifier:

**check** {  
 **not** (**all** x: Test.contents | isEven[x])

**iff**

**some** x: Test.contents | **not** (isEven[x])  
}

No counterexamples found. They are equivalent!

Here is a more abstract version:

**pred** p [x: **univ**] {}

**check** {  
 **not** (**all** x: **univ** | p[x])

**iff**

**some** x: **univ** | **not** (p[x])  
}

No counterexamples found. They are equivalent!