MODULE *RadixIteratorValidation*

EXTENDS *FiniteSets*, *Integers*, *RadixTrees*, *Sequences*, *TLC*

Set of characters to use for the alphabet of generated strings.

CONSTANT *Alphabet*

CmpOp is the comparison operator for ordered iteration. This should be TRUE if the first value is less than the second value.

CONSTANT *CmpOp*(_, _)

Length of input strings generated

CONSTANT *MinLength*, *MaxLength*

ASSUME

$\land \{MinLength, MaxLength\} \subseteq Nat$

$\land MinLength \leq MaxLength$

Number of unique elements to construct the radix tree with. This is a set of numbers so you can test with inputs of multiple sizes.

CONSTANT *ElementCounts*

ASSUME $ElementCounts \subseteq Nat$

Inputs is the set of input strings valid for the tree.

$Inputs \triangleq \text{UNION } \{[1 \,..\, n \to Alphabet] : n \in MinLength \,..\, MaxLength\}$

InputSets is the full set of possible inputs we can send to the radix tree.

$InputSets \triangleq \{T \in \text{SUBSET } Inputs : Cardinality(T) \in ElementCounts\}$

InputTrees is a set of two trees for all inputs used to test iteration of multiple trees.

$InputTrees \triangleq \{\langle RadixTree(input1), RadixTree(input2)\rangle : input1, input2 \in InputSets\}$

TRUE iff the sequence s contains no duplicates. Copied from *CommunityModules*.

LOCAL $isInjective(s) \triangleq \forall i, j \in \text{DOMAIN } s : (s[i] = s[j]) \Rightarrow (i = j)$

Converts a set to a sequence that contains all the elements of S exactly once. Copied from *CommunityModules*.

LOCAL $setToSeq(S) \triangleq \text{CHOOSE } f \in [1 \,..\, Cardinality(S) \to S] : isInjective(f)$

INSTANCE *RadixIterator*

Expected result given an input set is the sorted input set.

$Expected(input) \triangleq$

$SortSeq(setToSeq(input),$

LAMBDA x, y :

$\lor Len(x) < Len(y)$

$\lor \land Len(x) = Len(y)$
$\land \forall\, i \in \text{DOMAIN } x : CmpOp(x[i],\, y[i]) \lor x[i] = y[i])$

The iteration of a tree should be just its sorted inputs.

$IterateIsSortedInput \triangleq$
$\forall\, input \in InputSets :$
LET
$actual \triangleq Iterate(\langle RadixTree(input)\rangle)$
$expected \triangleq Expected(input)$
IN
IF $actual \neq expected$
THEN $Print(\langle$"actual: ", $actual$, "expected: ", $expected$, "input: ", $input\rangle$, FALSE)
ELSE TRUE

The iteration of two things in a stack should have the results of the second element followed by the first (*FIFO*).

$IterateMultiple \triangleq$
$\forall\, stack \in InputTrees :$
LET
$actual \triangleq Iterate(stack)$
$expected \triangleq Expected(Range(stack[2])) \circ Expected(Range(stack[1]))$
IN
IF $actual \neq expected$
THEN $Print(\langle$"actual: ", $actual$, "expected: ", $expected$, "stack: ", $stack\rangle$, FALSE)
ELSE TRUE

The expression that should be checked for validity in the model.

$Valid \triangleq$
$\land IterateIsSortedInput$
$\land IterateMultiple$

* Modification History
* Last modified *Fri Jul* 02 08:13:53 *PDT* 2021 by *mitchellh*
* Created *Thu Jul* 01 09:57:41 *PDT* 2021 by *mitchellh*