Stephen Belden

Chris Ruiz

Meghan Haukaas

2015-Sept-16

**Self-Adjusting List Tests**

**Description of Test Cases**

There are two cases for valid non-trivial inputs:

* Test Case 1 is the perfect case, in which all of the elements searched for are in the list. We accomplish this by setting the List Length and Max Value to the same number.
* Test Case 2 is the case in which at least 1 number searched for is not in the list. When a number is not in the list, every element of the list must be visited, regardless of whether the self-adjusting algorithm is used. Because of this, searching for an element that is not in the list always takes N steps, were N is the length of the list. In these tests we exaggerate this problem by setting the Max Value to ten times the List Length.

Several cases are not tested:

* There is a trivial case when the List Length is 0. In this case no elements are ever compared, regardless of what values are searched for. The Max Value and Cluster Size can be any value, and this case will still be trivial, with a total of 0 steps.
* Setting the Max Value to a number less than the List Length is an invalid set of inputs, as every element of the list must be unique.
* No negative inputs are considered valid.

**Test Case Construction - CR**

* Cluster size of 1, 7, 30
* List Length of 10, 100, 1000

**Expected Outcomes - MH**

Test Case 1 – Perfect Case

|  |  |  |  |
| --- | --- | --- | --- |
|  | **List Size 10** | **List Size 100** | **List Size 1000** |
| **Cluster size 1** | self-adjusting larger | **larger** | **larger** |
| **Cluster size 7** | **smaller** | **smaller** | **smaller** |
| **Cluster size 30** | **larger** | **smaller** | **smaller** |

Test Case 2 – More Misses than Hits

|  |  |  |  |
| --- | --- | --- | --- |
|  | **List Size 10** | **List Size 100** | **List Size 1000** |
| **Cluster size 1** | Self-adjusting larger | **larger** | **Larger** |
| **Cluster size 7** | **smaller** | **smaller** | **smaller** |
| **Cluster size 30** | **smaller** | **smaller** | **smaller** |