Design of Search Experiments

Table of Contents

1 Introduction

This document describes the design of search experiments.

2 Experiments

2.1 Common Elements In Interface

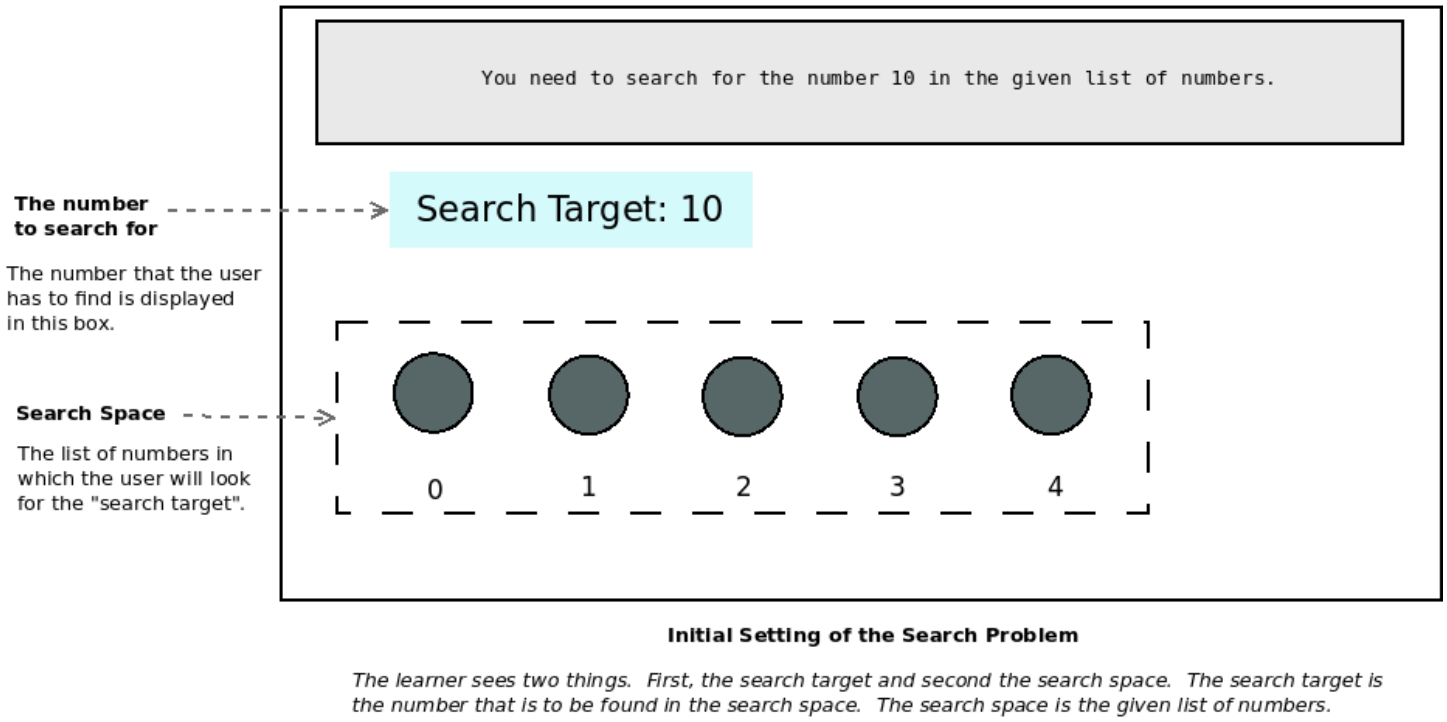


Figure 1: Search Initial Setup

Search Space
: List of items that acts as the search space.

Search Target
: Number that the user needs to search for.

Control buttons [optional]
: Buttons that allow the user to perform required operations.

Prompt
: The system should give a feedback message to the user after each interaction. This feedback indicates the result of the latest action and hints towards the next possible actions.

2.1.1 Search Space

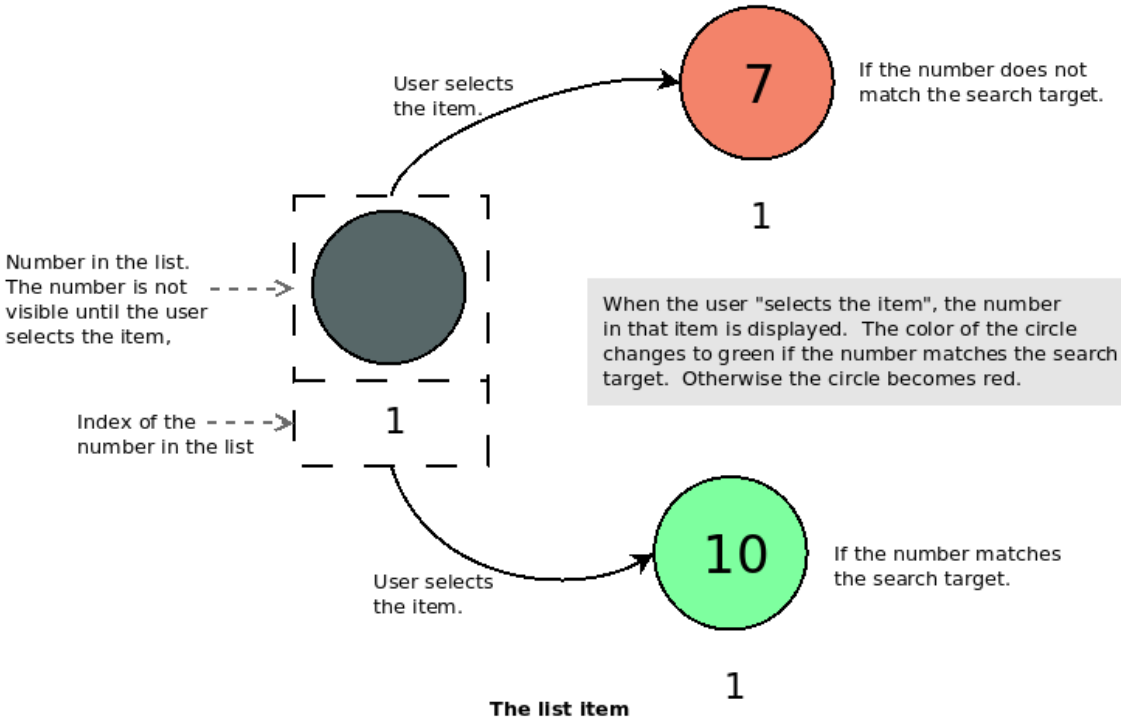
The search space for the search problem in our scope is a list of numbers. The search target may or may not be present in the search space. Each item in the list has two elements: First, the number that we compare the target with and second, the index of the item in the list.

The visual interface of the list item should display the index and the number. The visibility of the number depends on its system state which is defined by the experiment and is different for each experiment.

The number in the list item can be in two visibility states: **VISIBLE** or **HIDDEN**

When a number is **HIDDEN**, there should be a circle in place of the number.

When a number is **VISIBLE**, there should be circle in the background of the number. The color of the circle should indicate whether the number matches the search target.



The list item

The list item consists of two elements: The number which is enclosed in a circle and the index of the item in the list. The index is displayed below the circle containing the number.

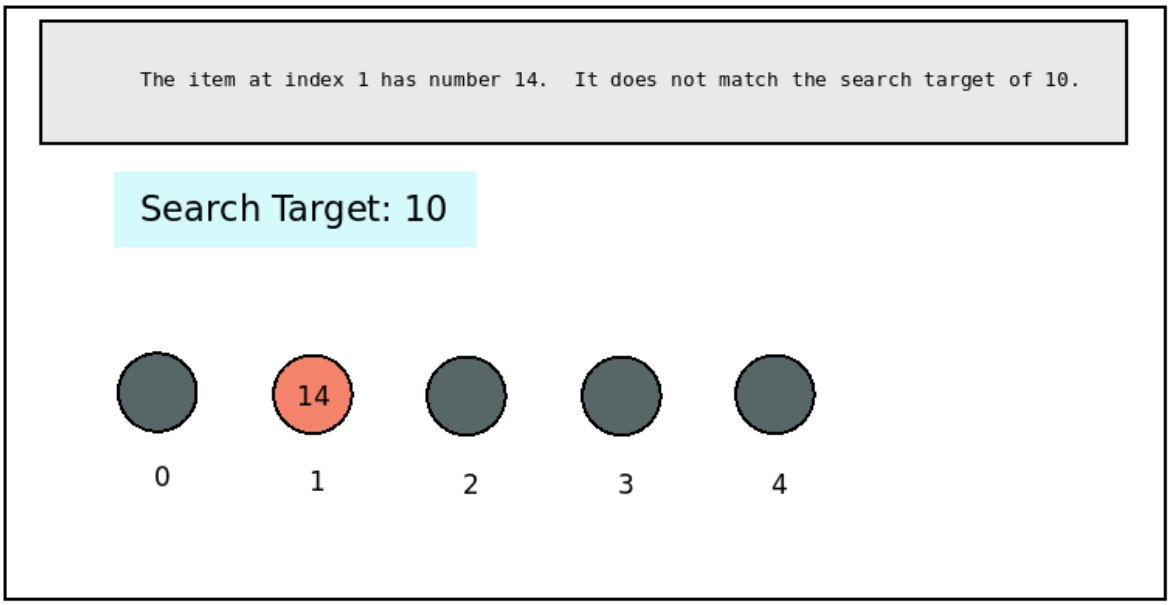
Search target matches the number

The circle should be "green".

Search target does not match the number

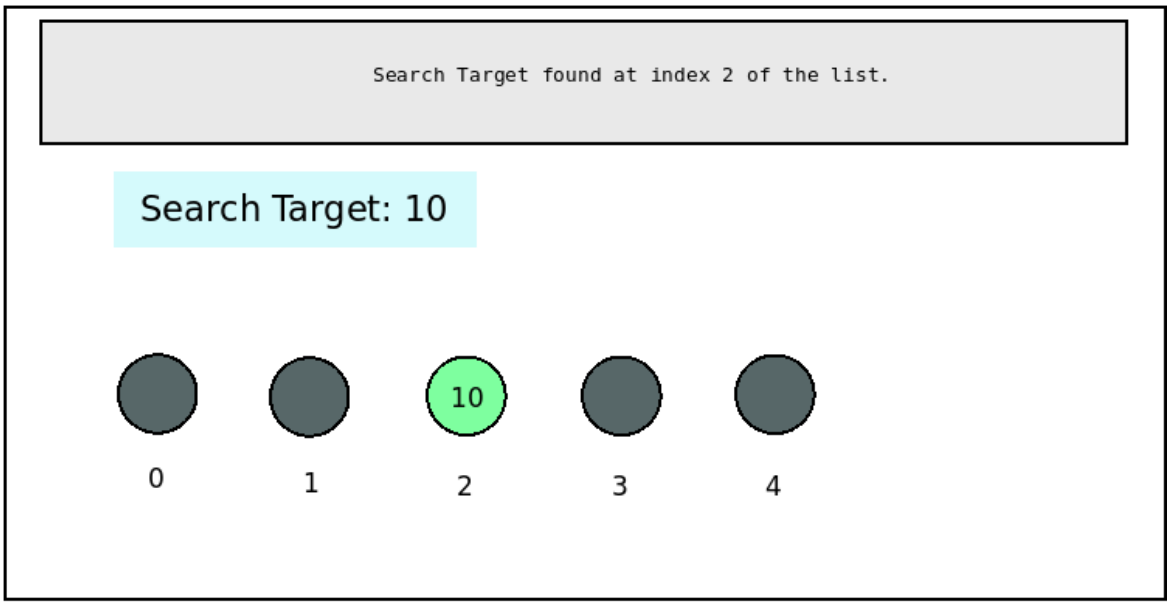
The circle should be

"red".



Selected Item that does not match the Search Target

When the user selects a list item, and the number in that list item does not match the search target, the circle turns red.



Selected Item matches the Search Target

When the user selects a list item, and the number in that list item matches the search target, the circle turns green.

2.1.2 Interactions

Interaction	Strategy	Requires Control Button
Select/unselect an item from the list.	Random	No

Interaction	Strategy	Requires Control Button
Compare the search item with the selected item from the list.	Random, Linear, Binary	No
Select the next item in the iteration	Linear	Yes
Show All elements	Binary	Yes

3 Random Search With Replacement

3.1 Interface

The user interface for the Random Search With Replacement experiment consists of the following:

Common Elements
All elements from the common elements list except for the buttons.

3.2 Interactions

The shuffle experiment consists of the following interactions:

- Select an element from the list of elements randomly.
- Deselect the selected element.

The next section describes this interaction in detail.

3.2.1 Pick an element from the list of elements

Let us say there is an list of some 'n' elements, from which we need to search for a target element 't'. All the elements will be hidden initially.

When we click on that element it will become visible, and the background color of the element will change depending on the fact whether it matches t or not.

For example, consider the following setting:

```
SEARCH_TARGET = 't'

LIST = [_, _, _, _,_]
```

If we click on the `3rd element(0 based index)` , the system results in the following states:

1. State 1 : Element selected is found to be 't'

   ```
   SEARCH_TARGET = t

   LIST = [_, _, _, t,_]
   ```

   Observe that only the `3rd element` became visible, since this time it matches the search target 't', the background color of the element will be `GREEN`.

2. State 2 : Element selected is not found to be 't'

```
SEARCH_TARGET = t

LIST = [_, _, _, k,_]
```

Observe that only the `3rd element` became visible, since this time it does not matches the search target 't', the background color of the element will be `RED`.

3.2.2 Deselect the selected element

When an element is selected (i.e. it is visible), the used would not be able to select another element. To select another element, the user needs to deselect the element first, by clicking the selected element again. This will result in making all the elements invisible again.

For example, consider the following setting:

```
SEARCH_TARGET = 't'

LIST = [_, _, _, k,_]
```

If we click on the `3rd element(0 based index)` , the system results in the following state:

```
SEARCH_TARGET = t

LIST = [_, _, _, _,_]
```

Observe that only the `3rd element` became invisible again.

If we click any other element than the selected one, then system will remain in same state.

```
SEARCH_TARGET = t

LIST = [_, _, _, k,_]
```

Observe that only the `3rd element` is still visible.

4 Random Search Without Replacement

4.1 Interface

The user interface for the Random Search Without Replacement experiment consists of the following:

Common Elements
All elements from the common elements list except for the buttons.

4.2 Interactions

The shuffle experiment consists of the following interactions:

- Select an element from the list of elements randomly.

It does not have any Deselect interaction, as no elements can be added back to the search space.

To read in detail how the Select element works, refer to Interactions section of Random Search With Replacement.

5 Linear Search

Linear search is a more constrained version of Random search without Replacement. The user here needs to select the elements linearly, this improves the search strategy.

5.1 Interface

The user interface for the Random Search Without Replacement experiment consists of the following:

Common Elements
: All elements from the common elements list.

Buttons
: There will be a single button `Increment`, which will increament the search index.

5.2 Interactions

The shuffle experiment consists of the following interactions:

- Select an element from the list of elements linearly from left to right.

It does not have any Deselect interaction, as no elements can be added back to the search space.

In the next section we describe how select works in this experiment.

5.2.1 Using Increment to Select an element from the list

The list is of length 'n', and we need to find the target element 't'. All the elements will be hidden initially.Initially the search index will be `-1`, as we are starting our search.

When we click on `Increment` button, the search index will increment and the corresponding element will become visible, and the background color of the element will change depending on the fact whether it matches `'t'` or not.

For example, consider the following setting:

```
SEARCH_TARGET = 't'

SEARCH_INDEX = 0

LIST = [k, _, _, _,_]
```

If we click on the `Increment` button, the system results in the following state:

```
 SEARCH_TARGET = t

 SEARCH_INDEX = 1

 LIST = [k, l, _, _,_]
```

Observe that all the element till `SEARCH_INDEX` become visible, the background of the visible elements will indicate whether they are the search target or not.

5.2.2 Final State of the experiment

The `Increment` button will be disabled when either the element at current `SEARCH_INDEX` is 't' or it has reached the end of the list.

For example, in the following setup

```
SEARCH_TARGET = t

SEARCH_INDEX = 1

LIST = [k, l, _, _,_]
```

Clicking increment, we find the element `t` at `SEARC_INDEX = 2`. The machine reaches the following final state.

```
SEARCH_TARGET = t

SEARCH_INDEX = 2

LIST = [k, l, t, _,_]
```

Now it is not possible to click `Increment` again.

6 Binary Search

Linear search is a new type of search strategy, where the target element `t` needs to be searched in **Sorted** List.

6.1 Interface

The user interface for the Random Search Without Replacement experiment consists of the following:

Common Elements
: All elements from the common elements list.

Buttons
: There will be a single button `Show All`, which will only be clickable when the experiment reaches the final state.

6.2 Interactions

The shuffle experiment consists of the following interactions:

- Select an element from the list of elements randomly.
- Click `Show All` to reveal all elements at once.

It does not have any Deselect interaction, as no elements can be added back to the search space.

To read about how the select action is implemented, please read Interactions Subsection of Random Search with Replacement experiment.

In the next section we describe how selection reduces the search space, which finally reduces the selectable elements.

6.2.1 Effect of selecting an element

The list is of length 'n', and we need to find the target element 't'. All the elements will be hidden initially.Initially the search index will be `-1`, as we are starting our search.

When we click on any element, the search index will be set to that element, if the element is the target element 't',then the experiment reaches the final state. In this case No more elements can be selected and `Show All` button will become active.

In case the selected element is not the target element, then depending whether, the selected element is less than the target element or greater than that, the element to the left and the elements to the right of selected element will become **Unselectable** (Still invisible, but background changed to `RED`).

For example

```
SEARCH_TARGET = t

SEARCH_INDEX = -1

LIST = [_, _, _, _,_]
```

if the user clicks `2nd element`, and the element is `k`. Then if k==t, we will set all other elements as unselectable, and system will reach final state. Else the system can be in following states

1. State 1 : `k` < `t`

   ```
   SEARCH_TARGET = t

   SEARCH_INDEX = 2

   LIST = [-, -, k, _,_]
   ```

   Here `-` denotes not selectable, only the elements right of k, can be selected.

2. State 1 : `k` > `t`

   ```
   SEARCH_TARGET = t

   SEARCH_INDEX = 2

   LIST = [_, _, k, -,-]
   ```

   Here `-` denotes not selectable, only the elements left of k, can be selected.

6.2.2 Clicking `Show All`

Let the machine be in final state (target element found/ does not exist), in such scenario, user can verify this using clicking the `Show All` button. This will make all the elements visible.

For example in the following configuration

```
SEARCH_TARGET = t

SEARCH_INDEX = 2

LIST = [-, -, t, _,_]
```

After clicking `Show All` system will be in the following state

```
SEARCH_TARGET = t

SEARCH_INDEX = 2

LIST = [a, b, t, c, d]
```

Date: 2021-04-19 Mon 00:00
Author: Venkatesh Choppella and Archit Goyal
Created: 2021-04-19 Mon 16:49
Validate