Sets

Simon Robinson

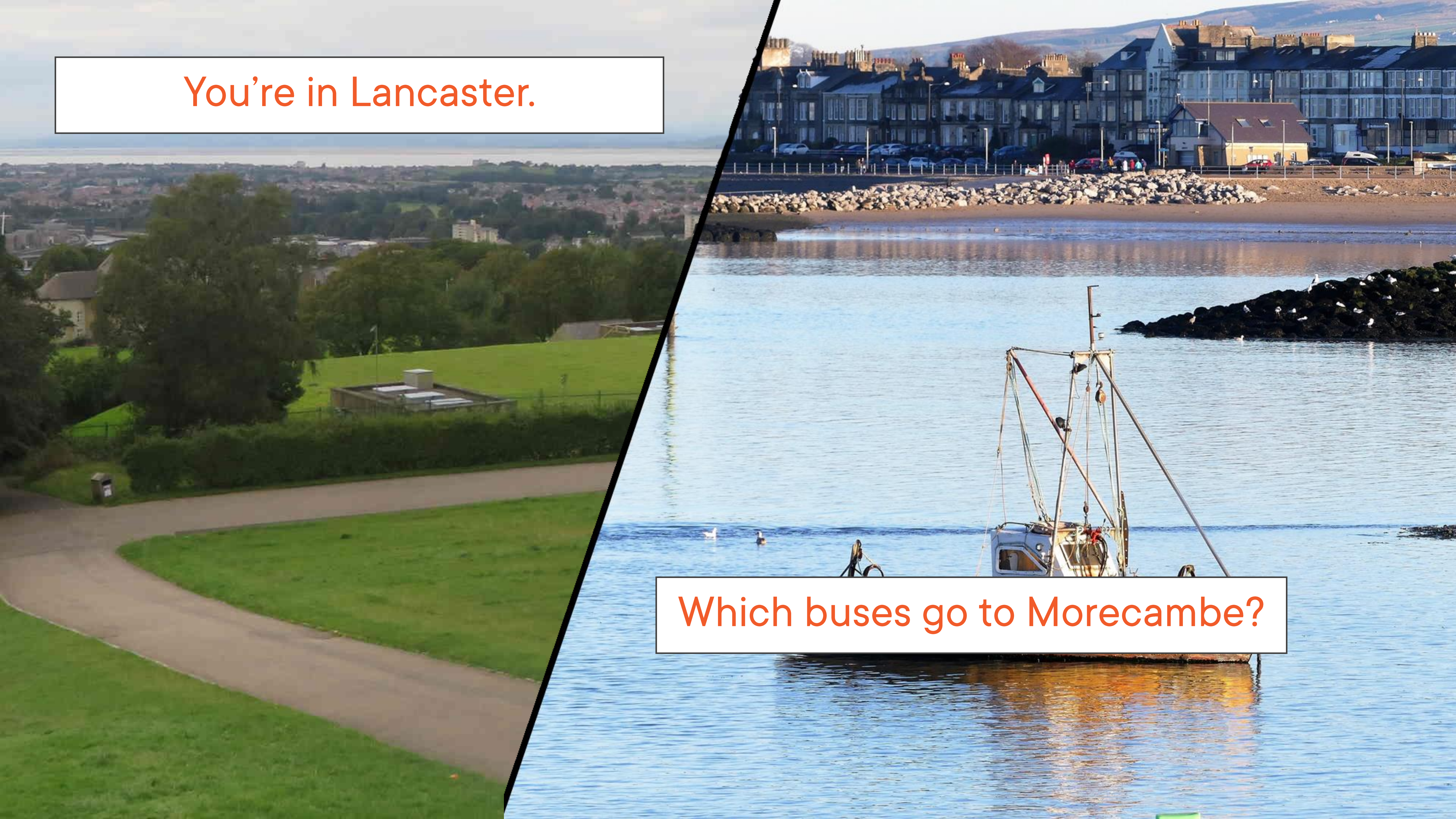
Software Developer

@TechieSimon www.SimonRobinson.com

Overview

`HashSet<T>`

- Other collections deal with individual elements
- `HashSet<T>` treats collections as a whole
    - Combines collections



You’re in Lancaster.

Which buses go to Morecambe?

Two Similar Problems

You're in Lancaster.

Which buses go to Morecambe?

This specifies the starting location too

Same as asking, which buses serve Lancaster **and** Morecambe

Which bus goes to Blackpool?

This specifies the starting location too

Same as asking, which buses serve Lancaster **and** Morecambe

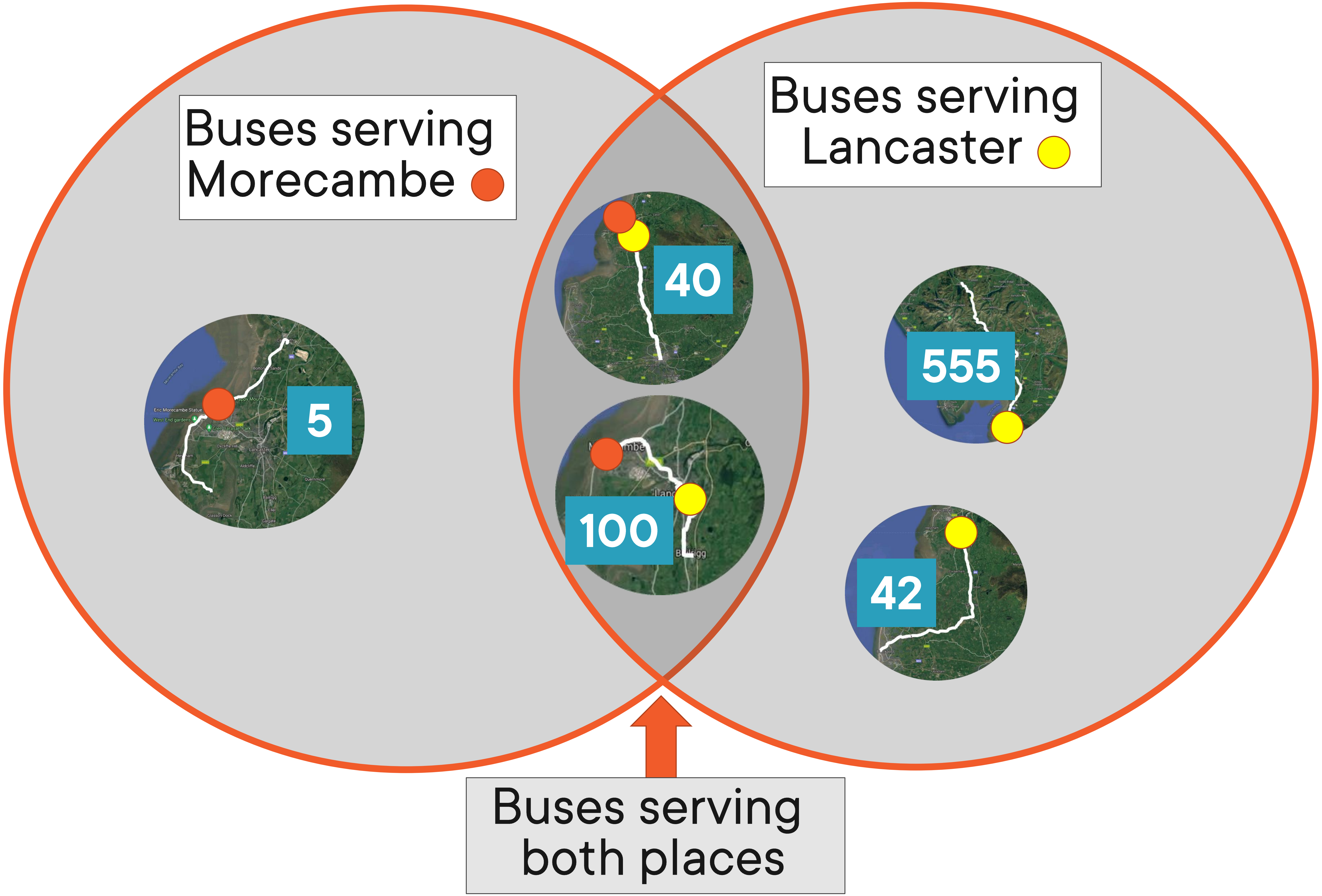
There are two solutions:

- Search an array
- Use a HashSet
- I'll cover both techniques

Demo

Adapt FindBusTo demo

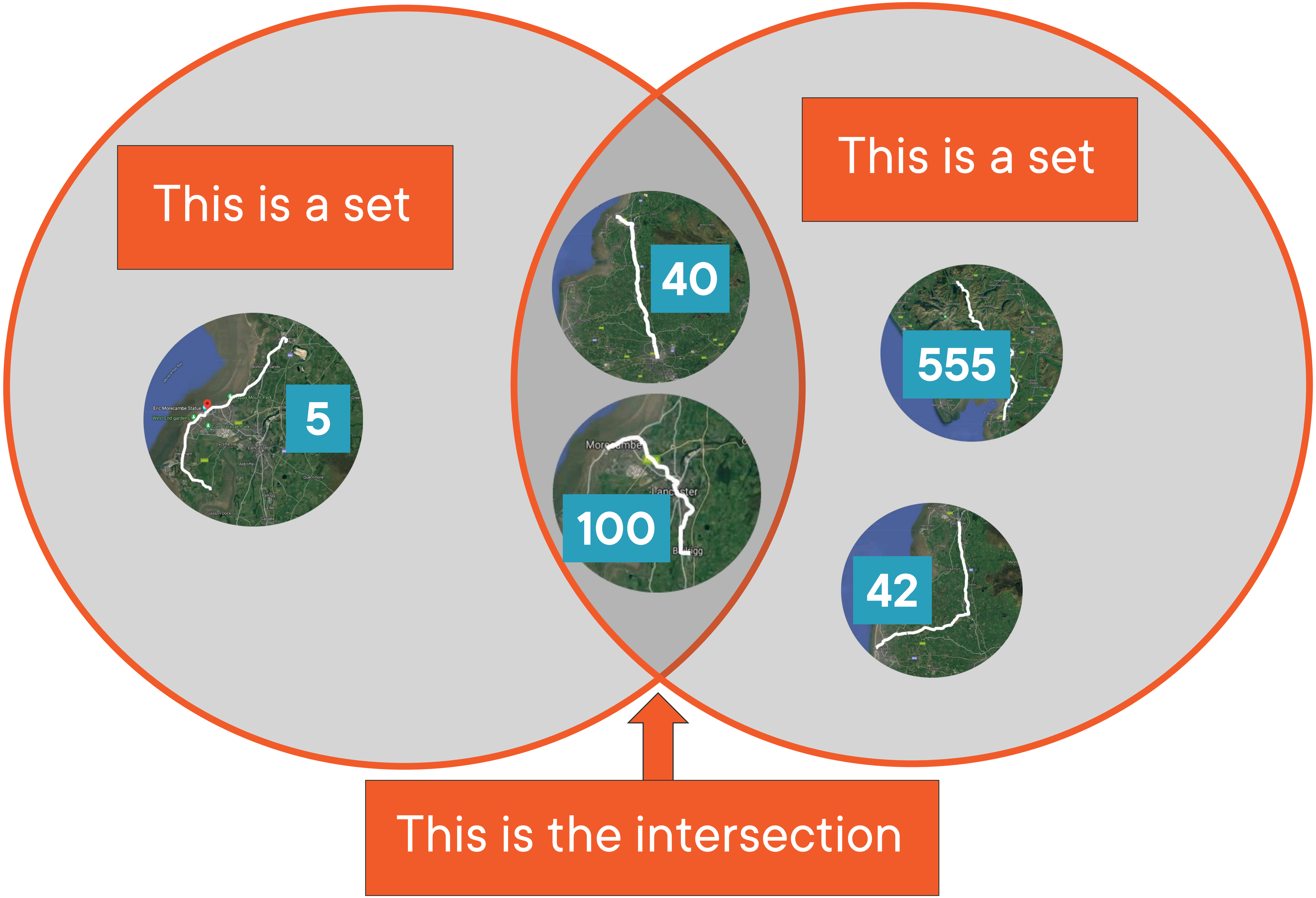
`HashSet<T>` gives a different way of looking at collections



Buses serving Morecambe ●

Buses serving Lancaster ●

Buses serving both places

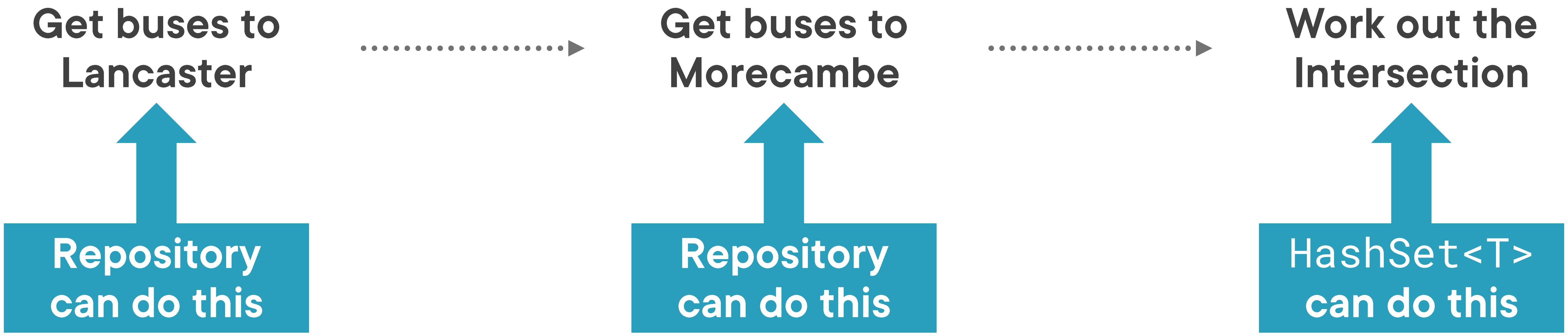


This is a set

This is a set

This is the intersection

To Get Buses from Lancaster to Morecambe



Get buses to Lancaster

Repository can do this

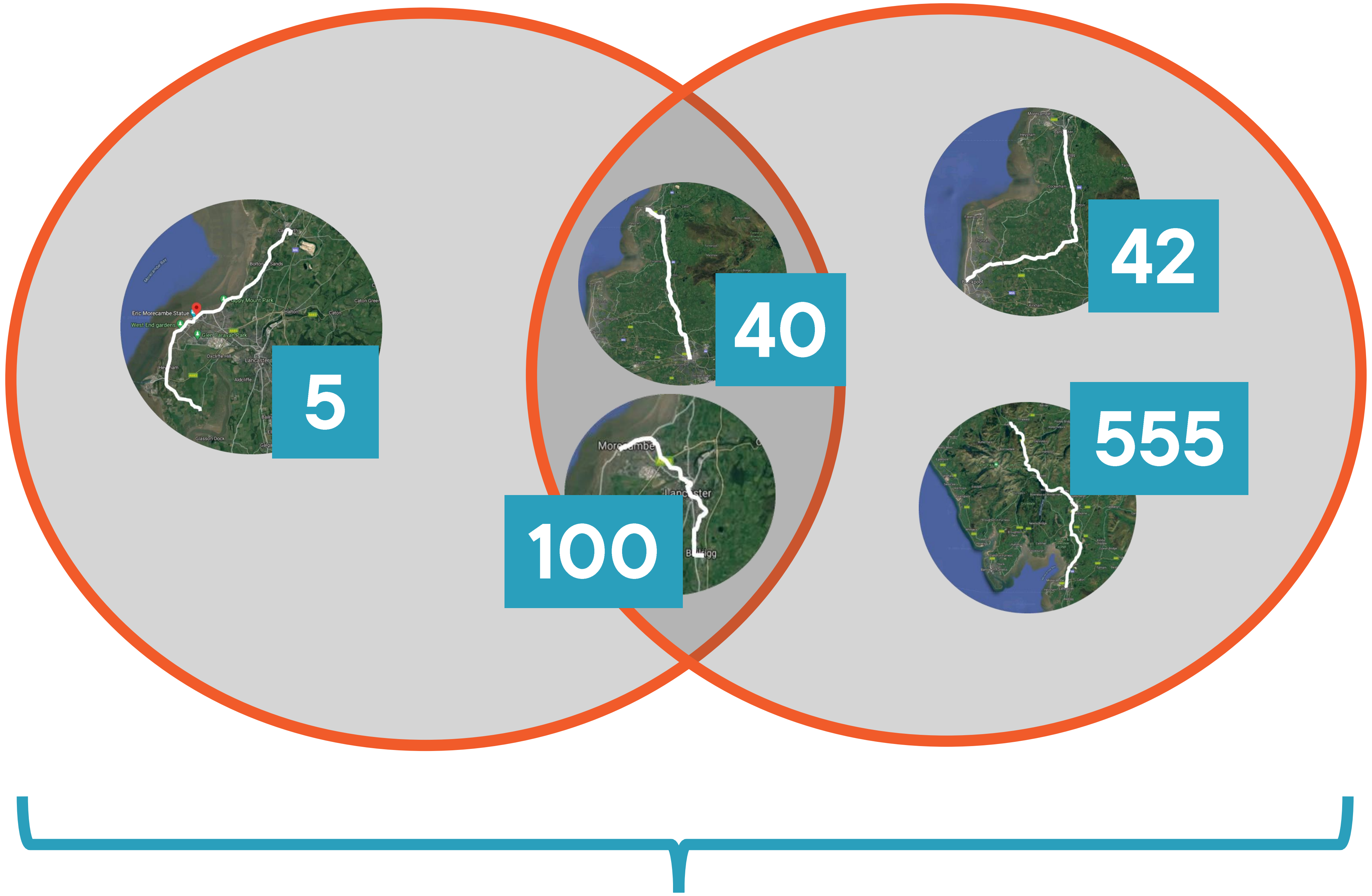
Get buses to Morecambe

Repository can do this

Work out the Intersection

HashSet<T> can do this

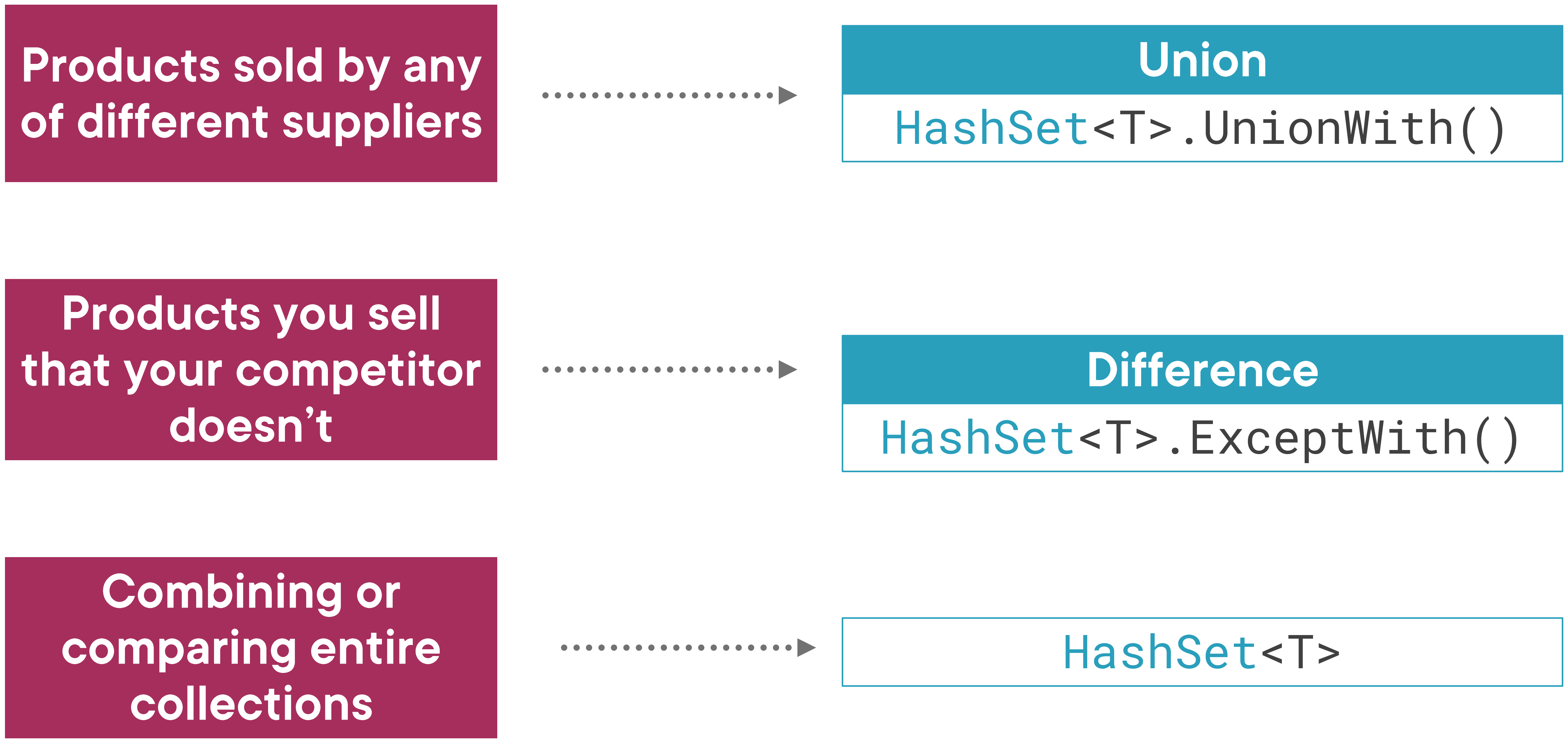
Union



(Buses that serve Lancaster *or* Morecambe)

Everything together is the union of the sets

HashSet Operations



Products sold by any of different suppliers → **Union**: `HashSet<T>.UnionWith()`

Products you sell that your competitor doesn’t → **Difference**: `HashSet<T>.ExceptWith()`

Combining or comparing entire collections → `HashSet<T>`

Summary

Combine collections together

- For example, Intersection
- HashSet<T> can do those kinds of operations

Up Next:
Multidimensional and Jagged Arrays