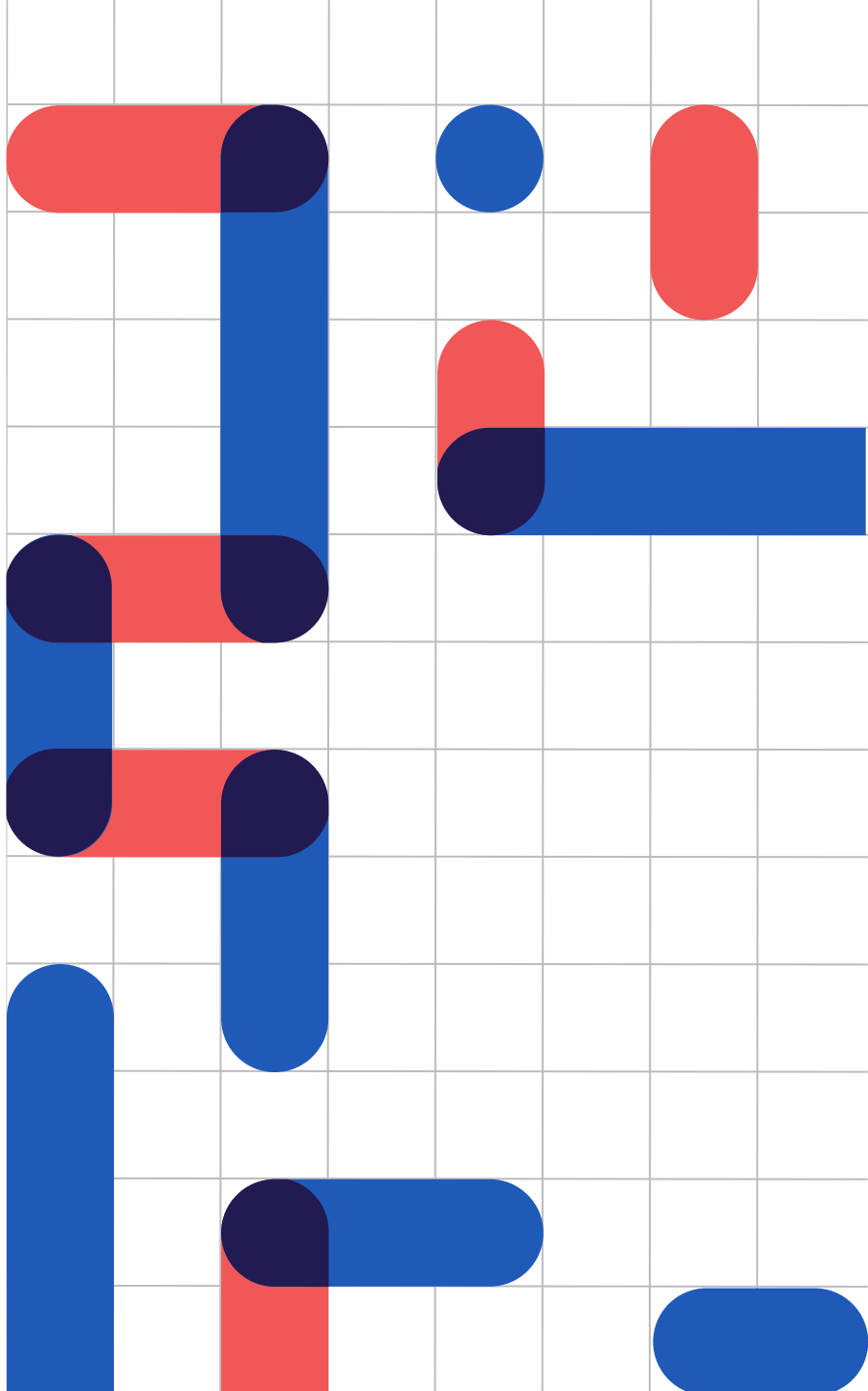
What is Broadcasting?

prepared by
Ramin Tavakoli



What is Broadcasting?

We don’t use iteration!

Broadcasting in python

As you have seen before, to perform a mathematical operation on each element of an array, a series of public functions are introduced in nampy, which speed it up by performing vector operations. A similar problem occurs when performing an operation between two arrays with different dimensions. Here, a feature called Broadcasting comes to our aid.

In [1]:
```
1 a = np.array([0, 1, 2, 3])
2 b = np.array([2, 2, 2, 2])
3 a * b
```

Out [1]:
```
array([0, 2, 4, 6])
```

Broadcasting in python

Suppose we want to multiply all the houses of an array by a fixed number. We learned earlier that by using the General multiplication function we can multiply two arrays of the same shape so that their corresponding elements are multiplied together. In this way, our request can be realized as follows.

In [1]:

```
1 a = np.array([0, 1, 2, 3])
2 b = np.array([2, 2, 2, 2])
3 a * b
```

Out [1]:

```
array([0, 2, 4, 6])
```

Plain text Copy

Broadcasting in python

But fortunately, there is no need to do this and in Nampai we can achieve the same result by multiplying a number in the array.

```
In [2]:
1 | 2 * a
```

```
Out [2]:
array([0, 2, 4, 6])
```

Broadcasting in python

If we consider the constant number as a zero-dimensional array, what happens in the above calculation is called ”Broadcasting".

It's like the smaller array expands and maps itself to the larger array (as in the first example) and then the multiplication is done.

Of course, in reality, this enlargement does not happen and the extra memory is not used.

3	5	-4
1	4	2
10	0	3

×

2	2	2
2	2	2
2	2	2

=

6	10	-8
2	8	4
20	0	6

Broadcasting in python

This property is also valid for higher dimensions. Suppose we have a two-dimensional array and we want to add each of its rows, which is considered a one-dimensional subarray, with a one-dimensional array of the same size as the rows.

In [3]:

```
a = np.arange(12).reshape(3, 4)
print(a)
b = np.array([3, 2, 1, 0])
print(b)
a + b
```

Broadcasting in python

In [3]:

```
1 a = np.arange(12).reshape(3, 4)
2 print(a)
3 b = np.array([3, 2, 1, 0])
4 print(b)
5 a + b
```

Out [3]:

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
[3 2 1 0]
array([[ 3,  3,  3,  3],
       [ 7,  7,  7,  7],
       [11, 11, 11, 11]])
```

Plain text Copy

Broadcasting in python

Numpy first transforms the b array into (1, 4) so that the two arrays are aligned.
In the following, by spreading the array b in the direction of the first axis, this array is treated like an array (3, 4) obtained by repeating b in the direction of the first axis.

```
In [1]: import numpy as np

In [2]: a = np.arange(12).reshape(3, 4)

In [3]: b = np.array([3, 2, 1, 0])

In [4]: a.shape
Out[4]: (3, 4)

In [5]: b.shape
Out[5]: (4,)
```

Broadcasting in python

```
In [6]: b = np.array([[8],[4],[0]])

In [7]: print(b)
[[8]
 [4]
 [0]]

In [8]: a + b
Out[8]:
array([[ 8,  9, 10, 11],
       [ 8,  9, 10, 11],
       [ 8,  9, 10, 11]])

In [9]: a.shape
Out[9]: (3, 4)

In [10]: b.shape
Out[10]: (3, 1)
```

Broadcasting in python

Both may need to be played for the arrays to conform.

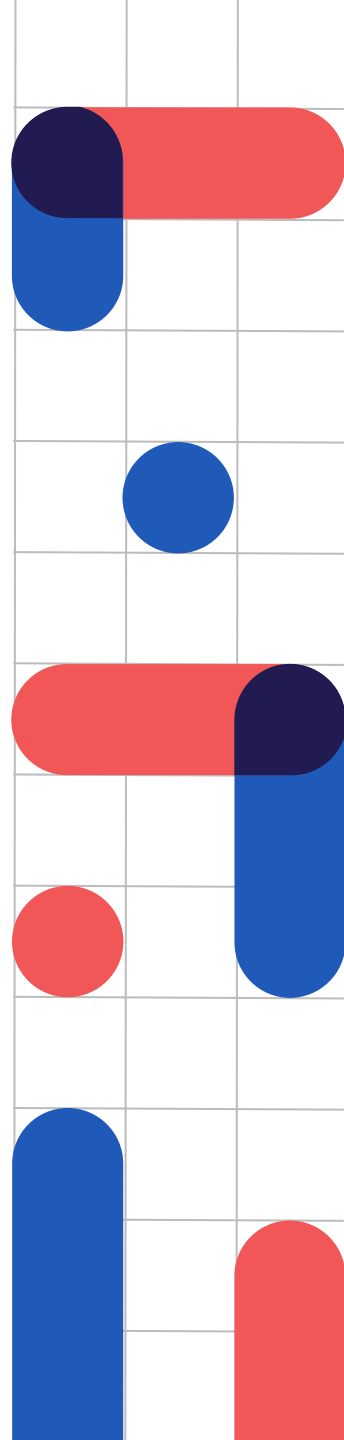
```
In [11]: a = np.array([1, 2, 3, 4, 5])

In [12]: b = np.array([1, 2, 3, 4]).reshape(4, 1)

In [13]: a.shape
Out[13]: (5,)

In [14]: b.shape
Out[14]: (4, 1)

In [15]: a * b
Out[15]:
array([[ 1,  2,  3,  4,  5],
       [ 2,  4,  6,  8, 10],
       [ 3,  6,  9, 12, 15],
       [ 4,  8, 12, 16, 20]])
```



Thank you!

Do you have any questions?

 Ramin6tavakoli6@gmail.com

 @Ramintavakkoli

 linkedin.com/in/ramin-tavakolii

 github.com/ramintavakolii