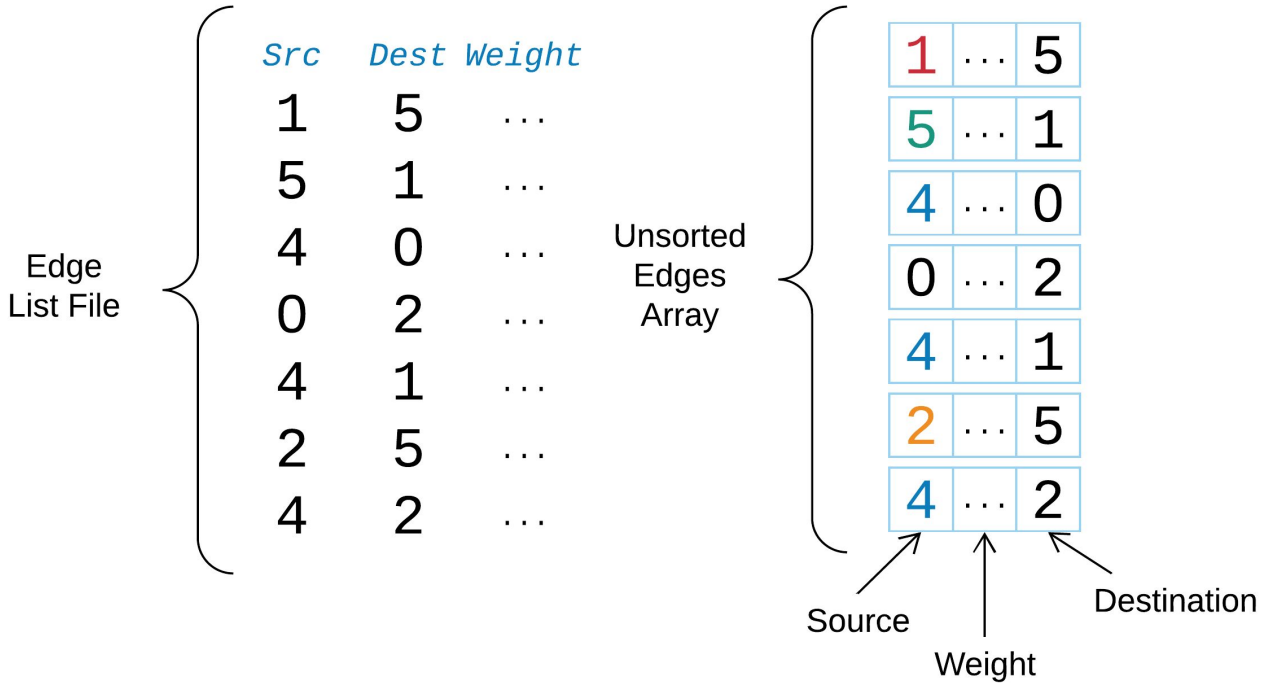
End-to-End Graph Processing

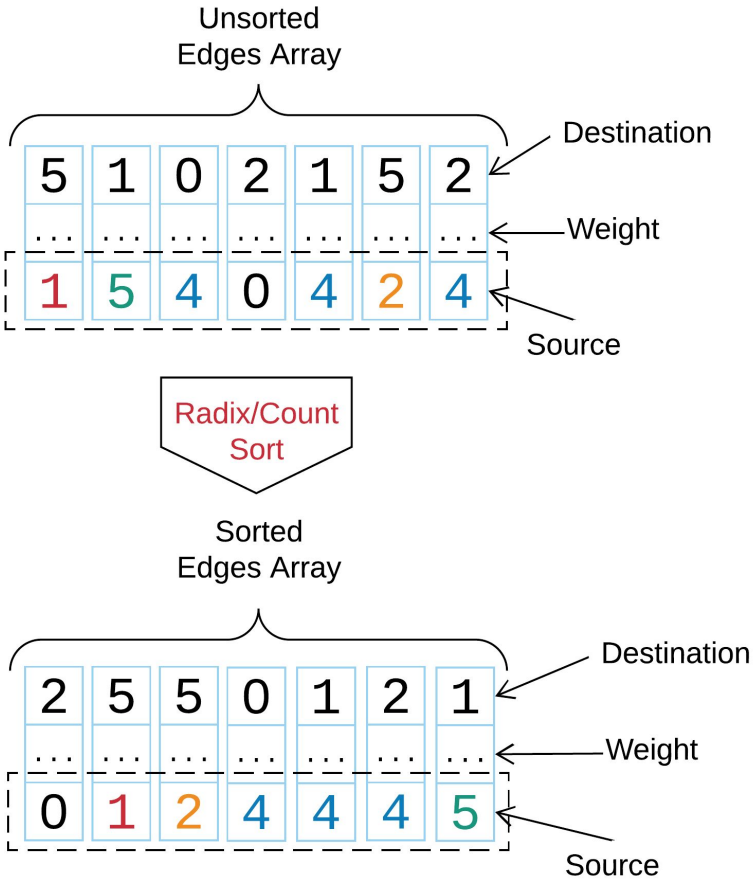
Algorithms

Presented by : Abdullah Mughrabi

Recap


Edge
List File
Src Dest Weight
1 5 ...
5 1 ...
4 0 ...
0 2 ...
4 1 ...
2 5 ...
4 2 ...
Unsorted
Edges
Array
1 ... 5
5 ... 1
4 ... 0
0 ... 2
4 ... 1
2 ... 5
4 ... 2
Source
Weight
Destination

Recap


Unsorted
Edges Array
Destination
5 1 0 2 1 5 2
Weight
1 5 4 0 4 2 4
Source
Radix/Count
Sort
Sorted
Edges Array
Destination
2 5 5 0 1 2 1
Weight
0 1 2 4 4 4 5
Source

Sorting in linear time

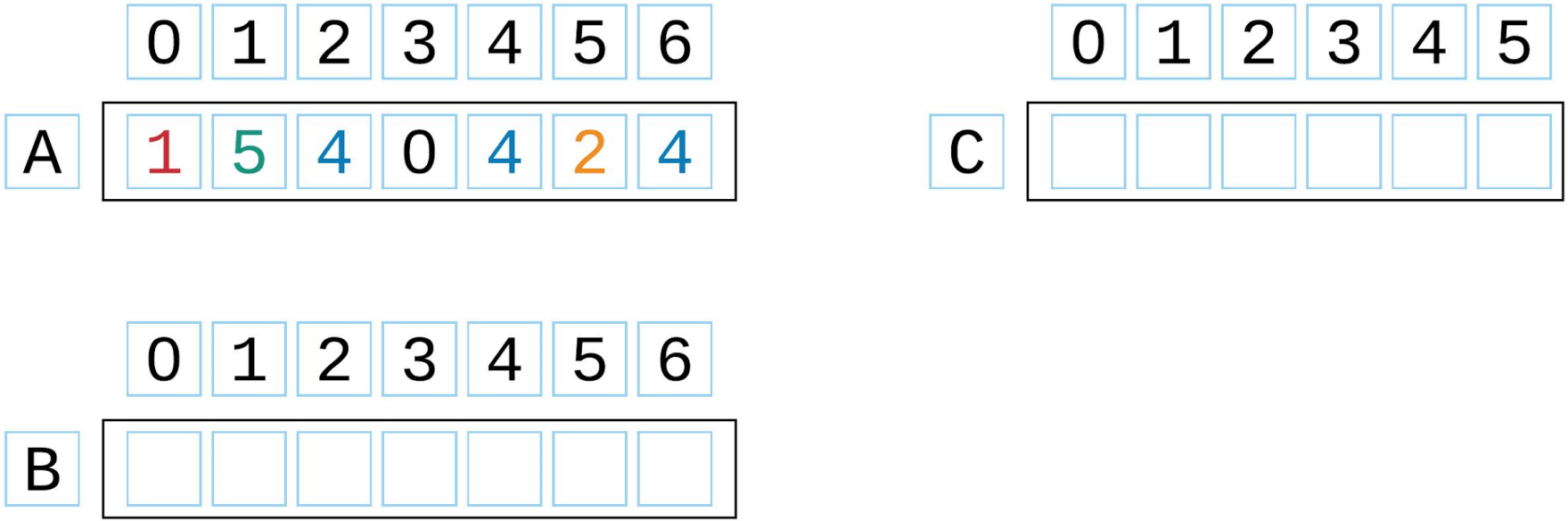
Counting-sort : No comparison between elements.

- ***Input***: A[1..n], where A[j] ∈ {1,2, ...,k}.
- ***Output***: B[1..n], sorted.
- ***Auxiliary* storage**: C[1..k], counting array.

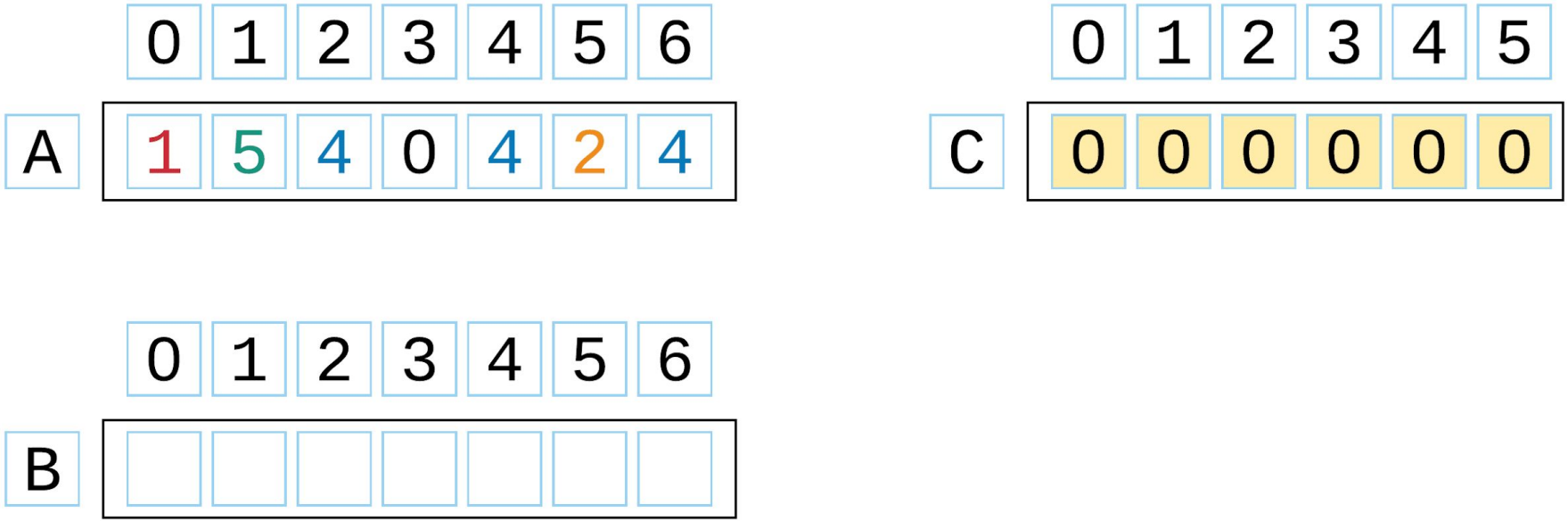
Counting-sort (Serial)

```
for i ←0 to (k-1)
   do C[i] ←0

for j ←0 to (n-1)
   do C[A[j]]←C[A[j]]+1

for i ←1 to (k-1)
   do C[i]←C[i]+C[i-1]

for j ←(n-1) downto 0
   do B[C[A[j]]-1] ←A[j]
      C[A[j]]←C[A[j]]-1
```

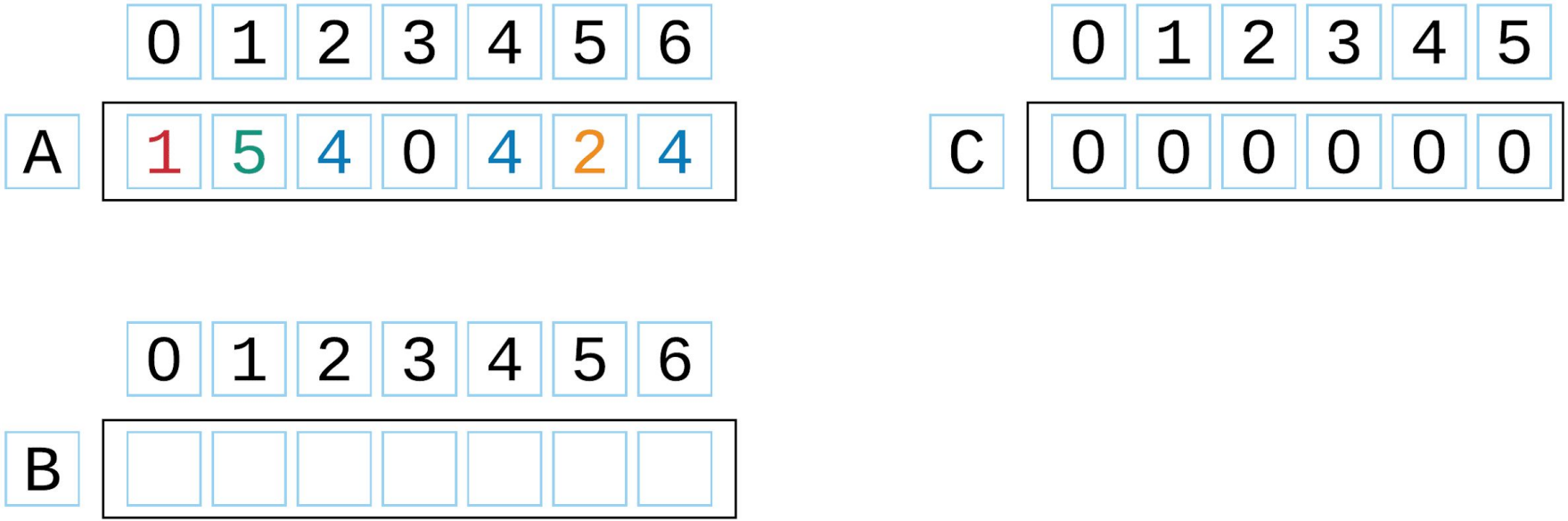
Counting-sort (Serial) - Example



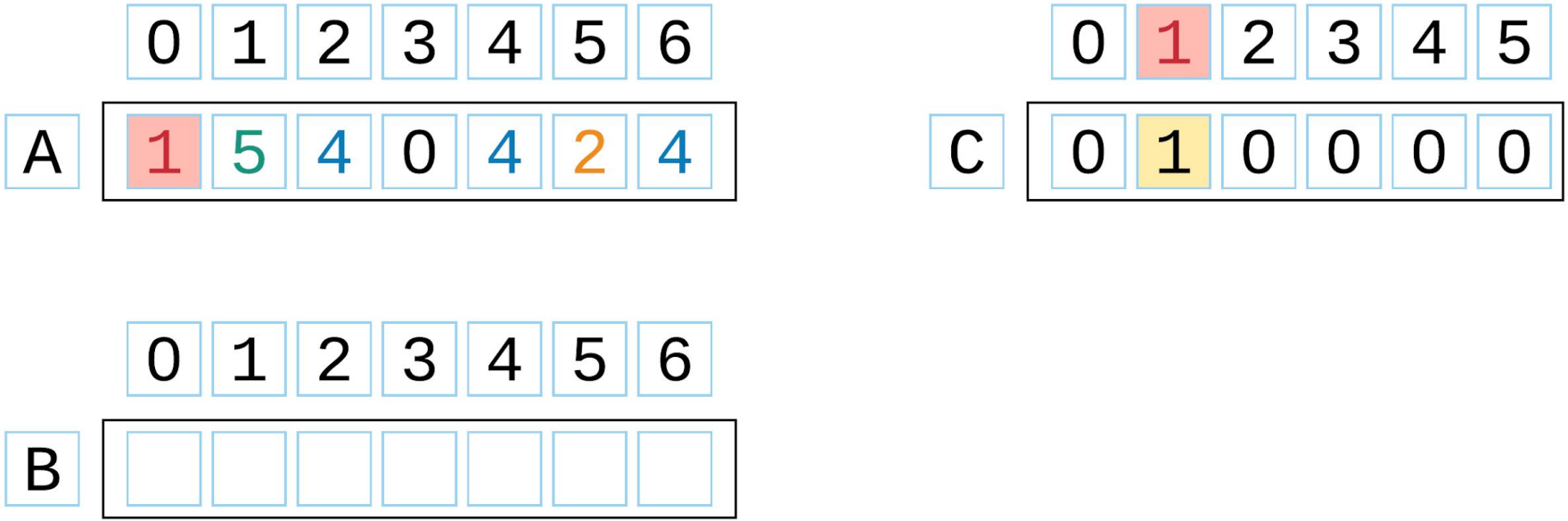
Counting-sort (Serial) - Loop 1



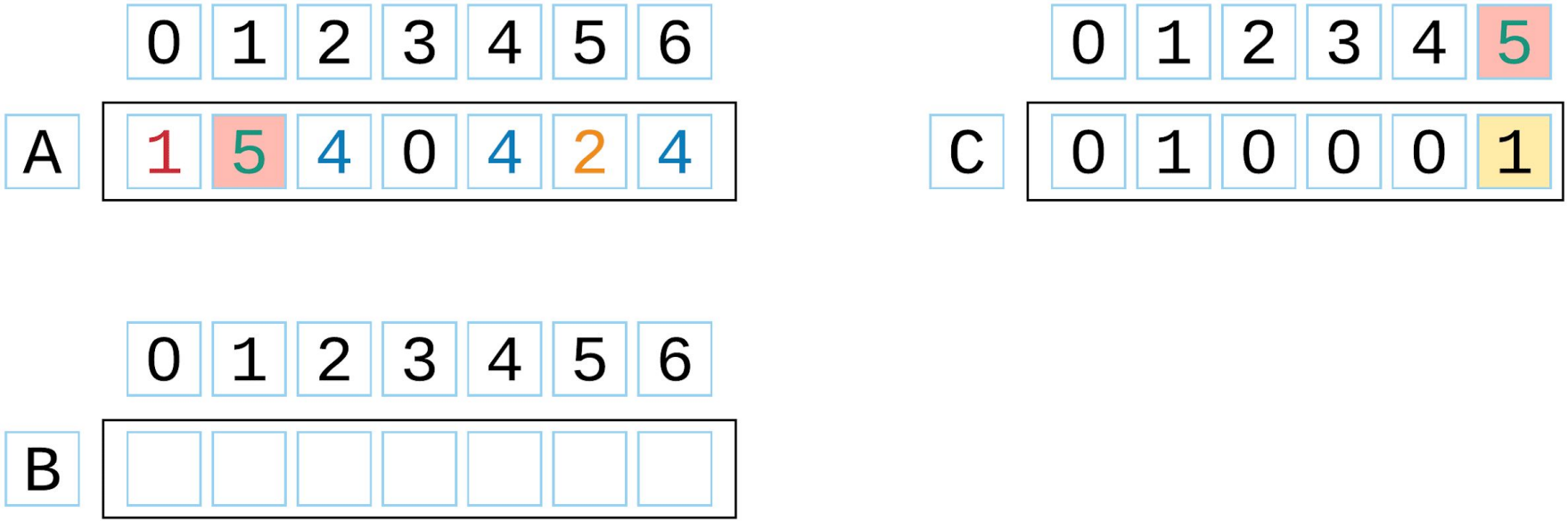
Counting-sort (Serial) - Loop 2



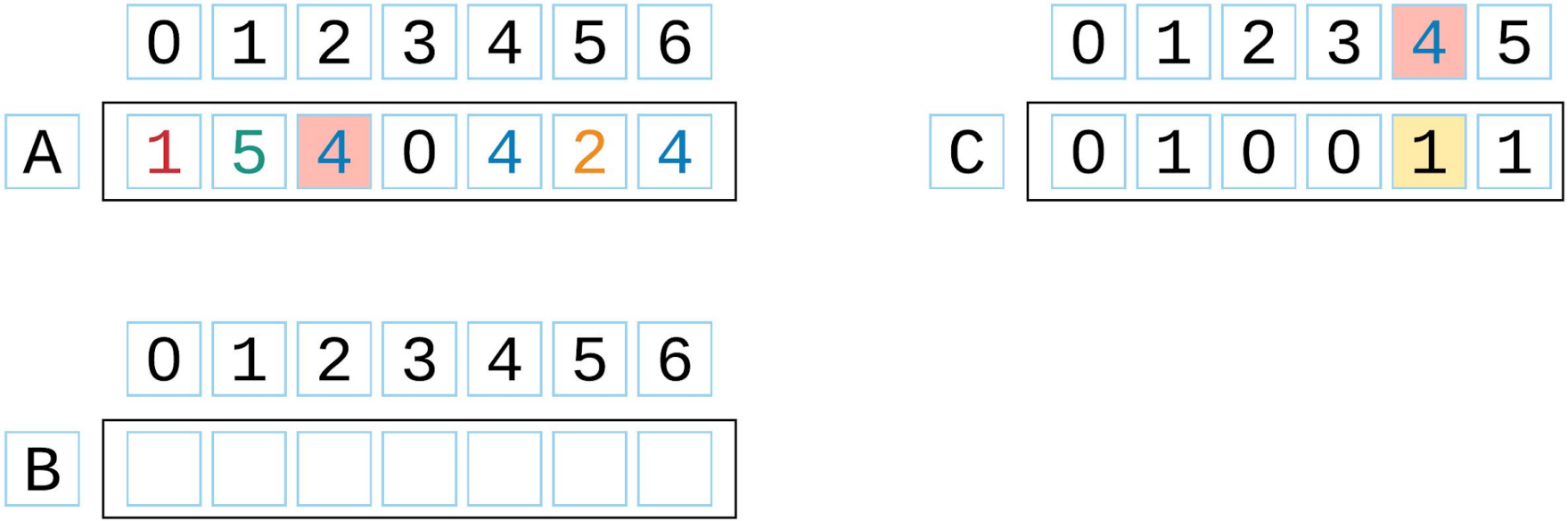
Counting-sort (Serial) - Loop 2



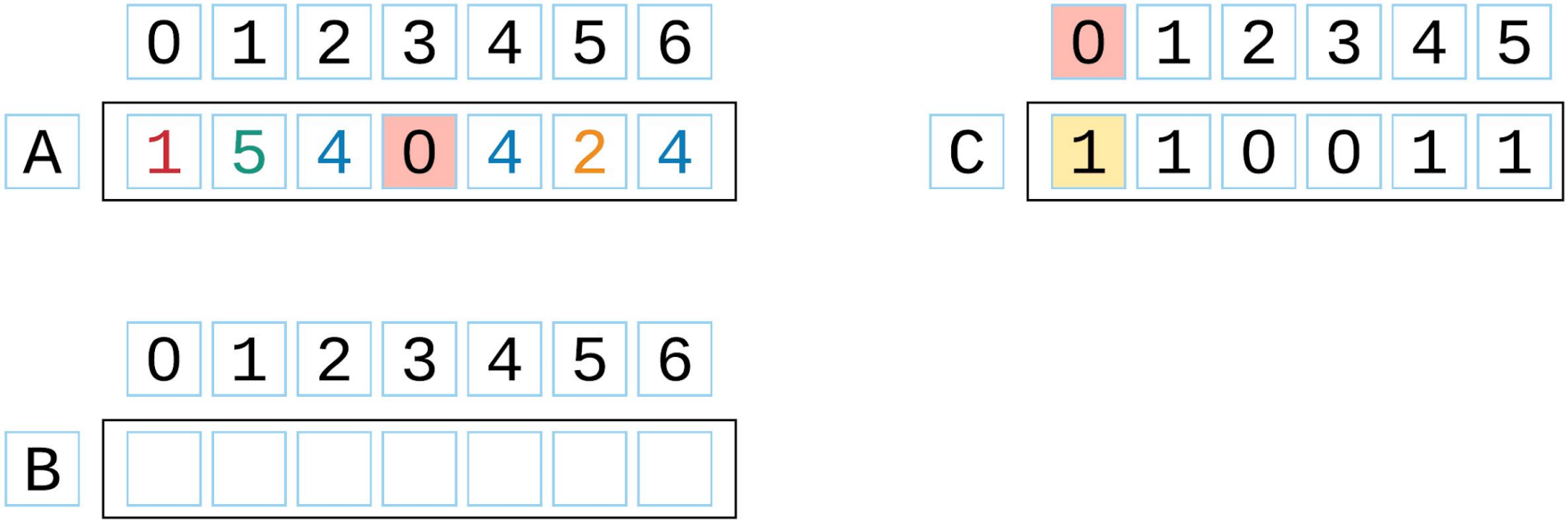
Counting-sort (Serial) - Loop 2



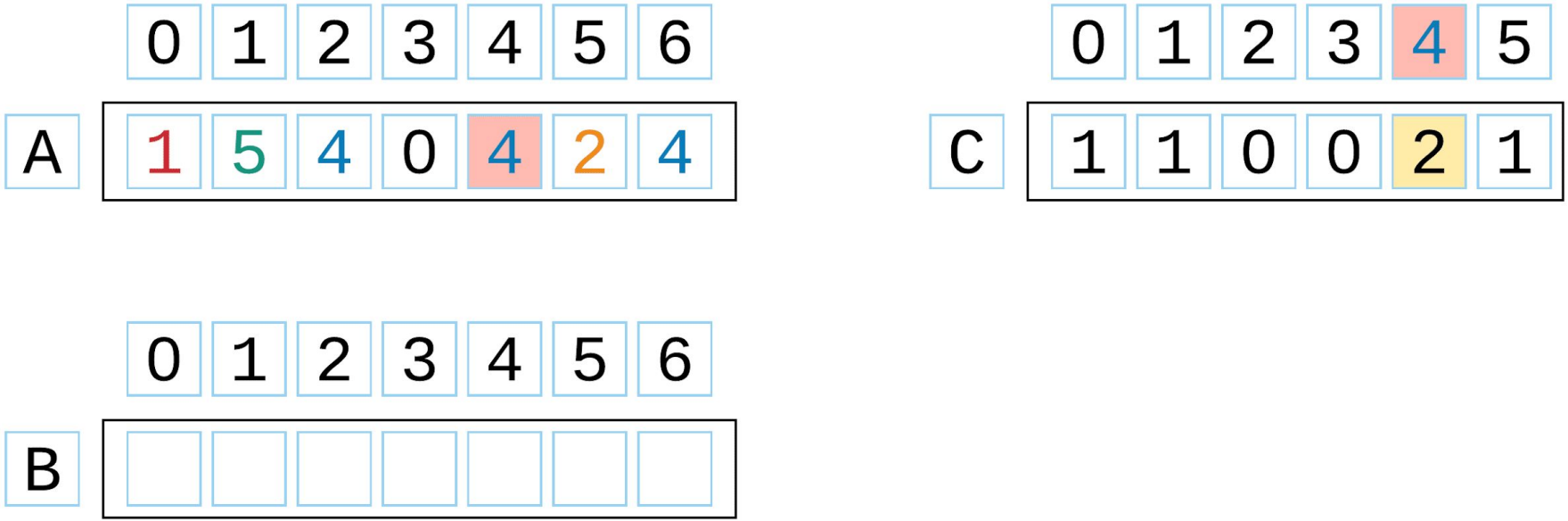
Counting-sort (Serial) - Loop 2



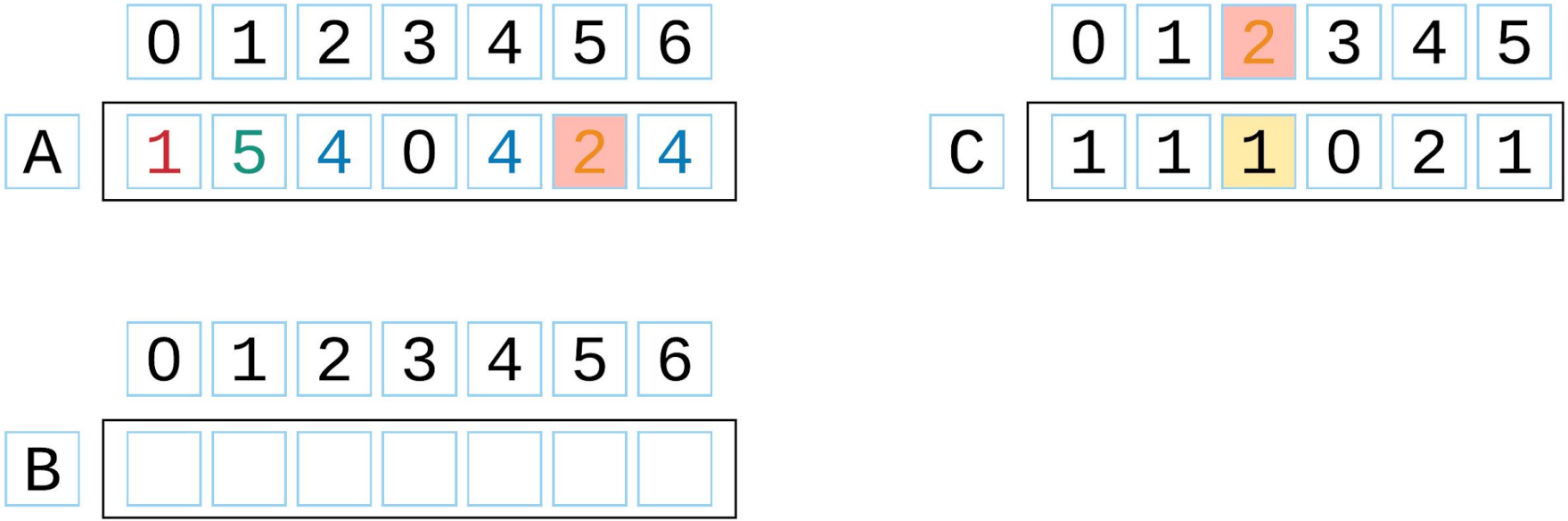
Counting-sort (Serial) - Loop 2



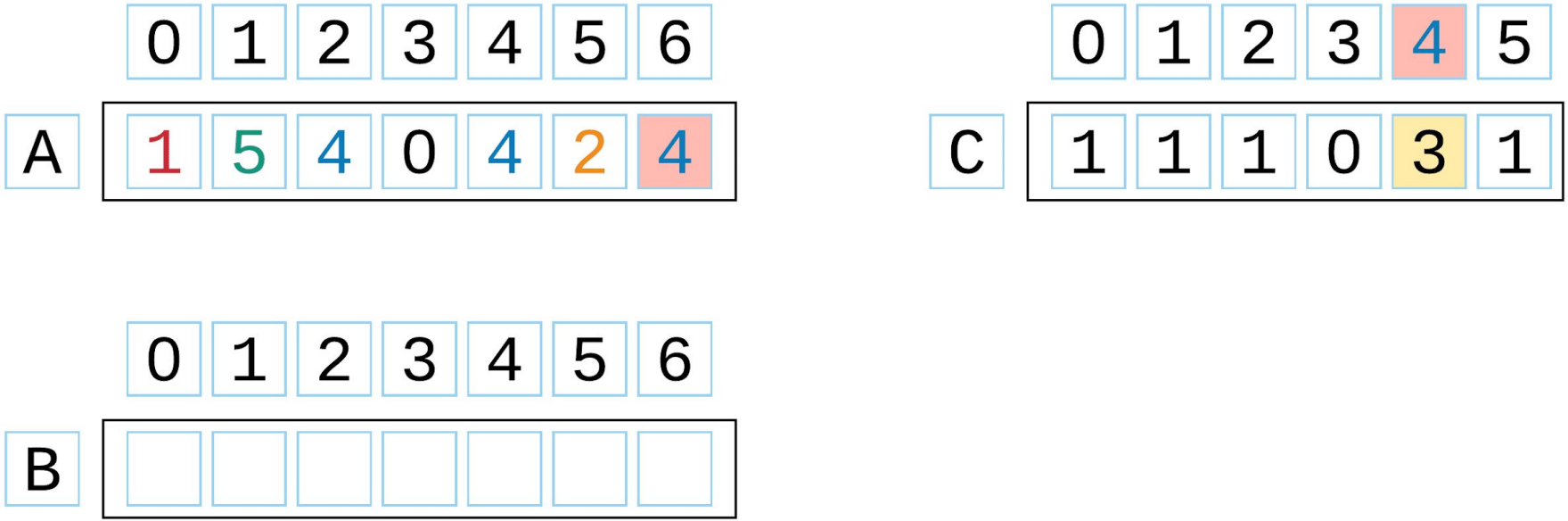
Counting-sort (Serial) - Loop 2



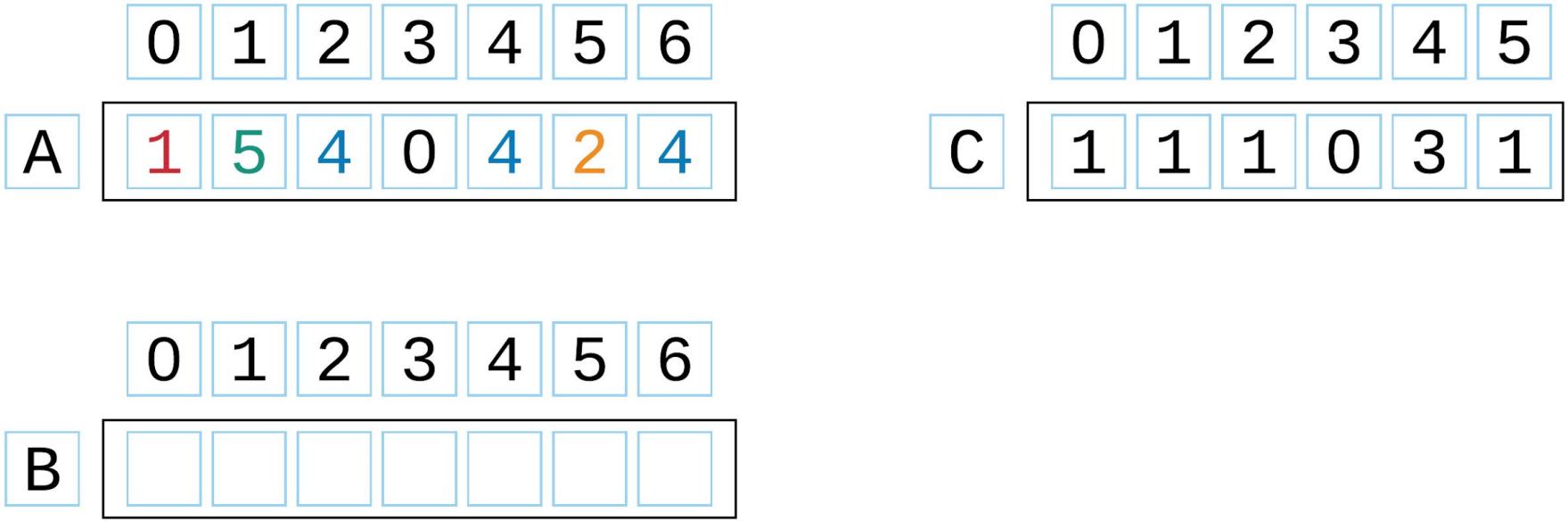
Counting-sort (Serial) - Loop 2



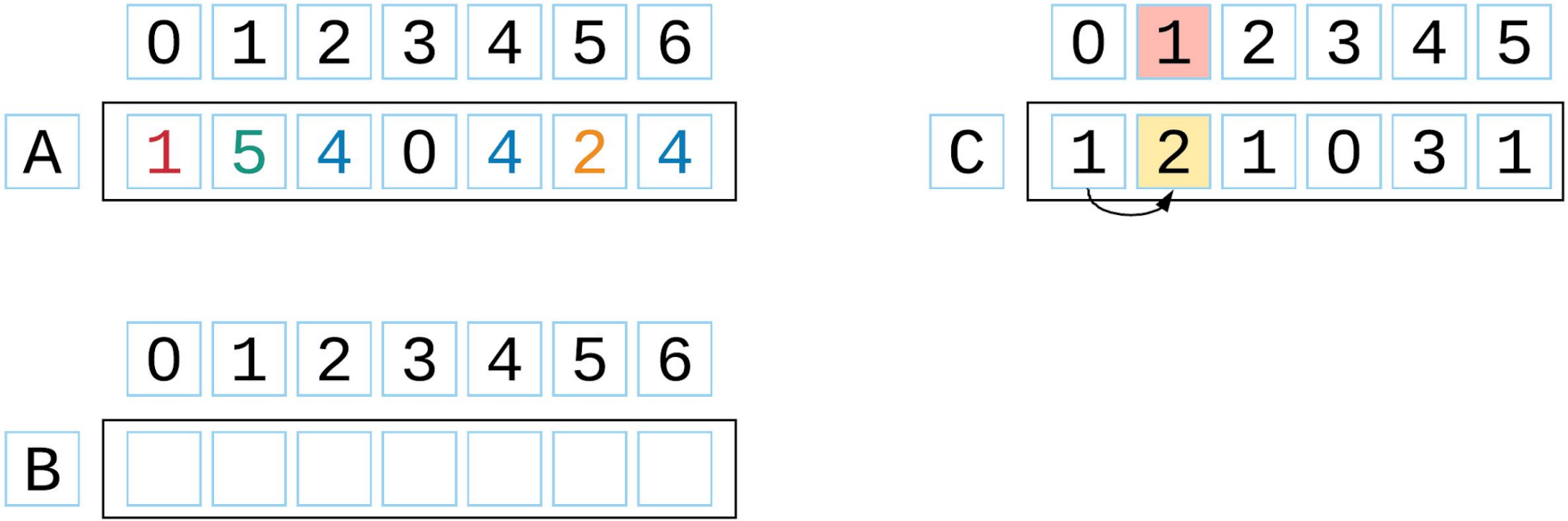
Counting-sort (Serial) - Loop 2



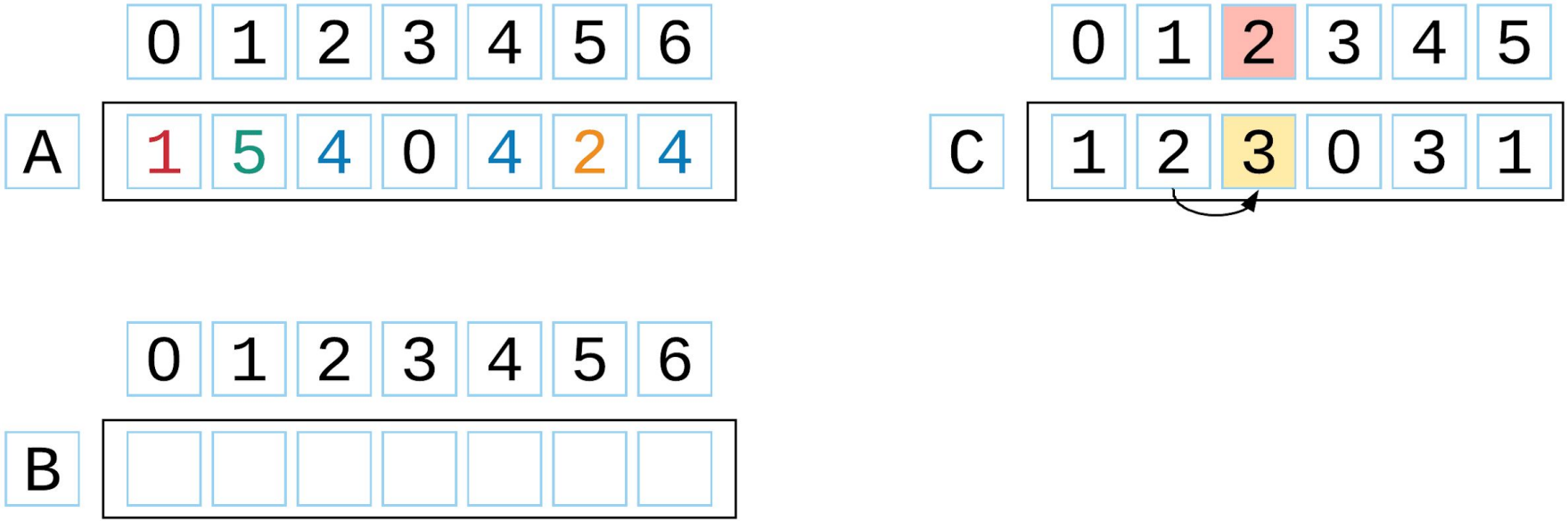
Counting-sort (Serial) - Loop 3



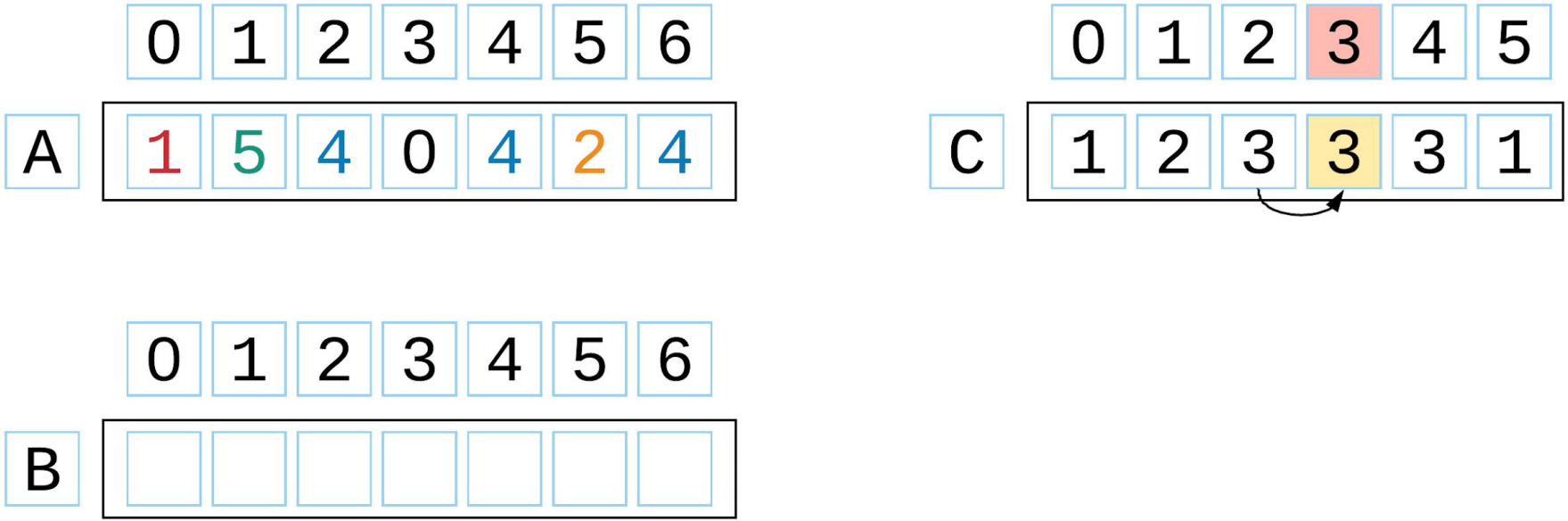
Counting-sort (Serial) - Loop 3



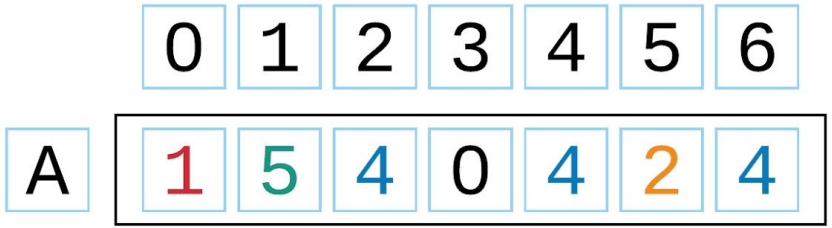
Counting-sort (Serial) - Loop 3

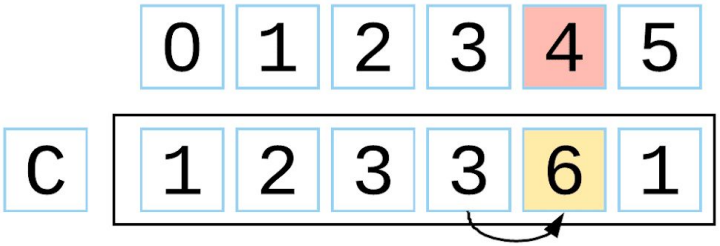


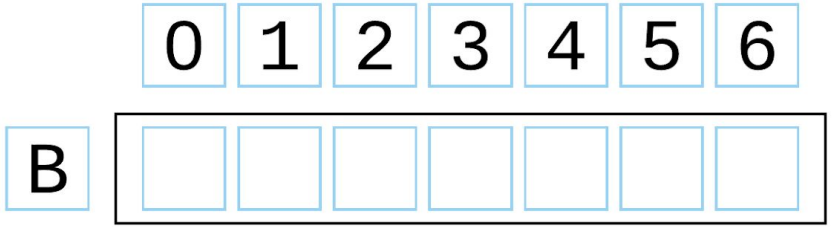
Counting-sort (Serial) - Loop 3



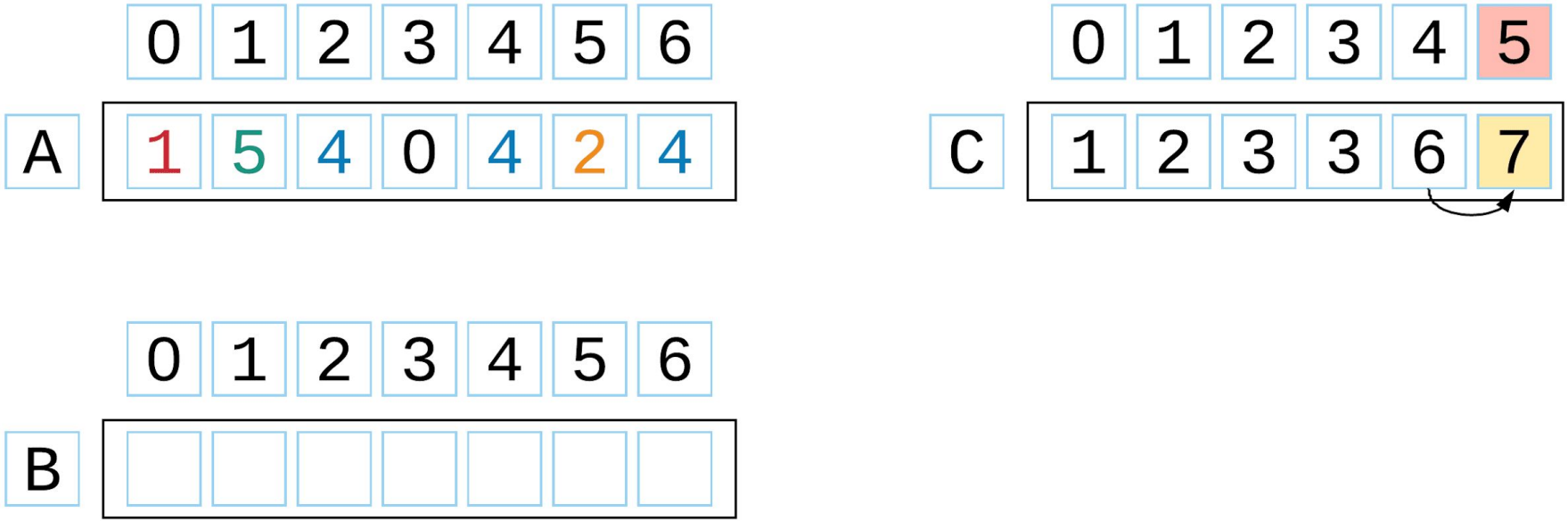
Counting-sort (Serial) - Loop 3



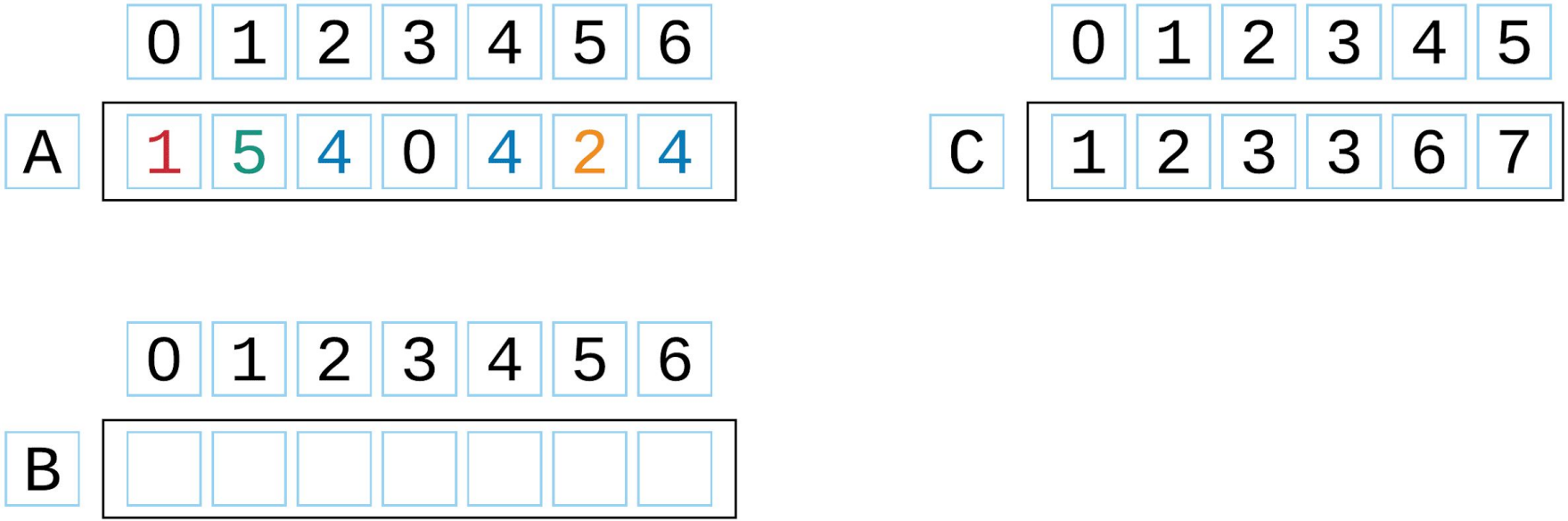




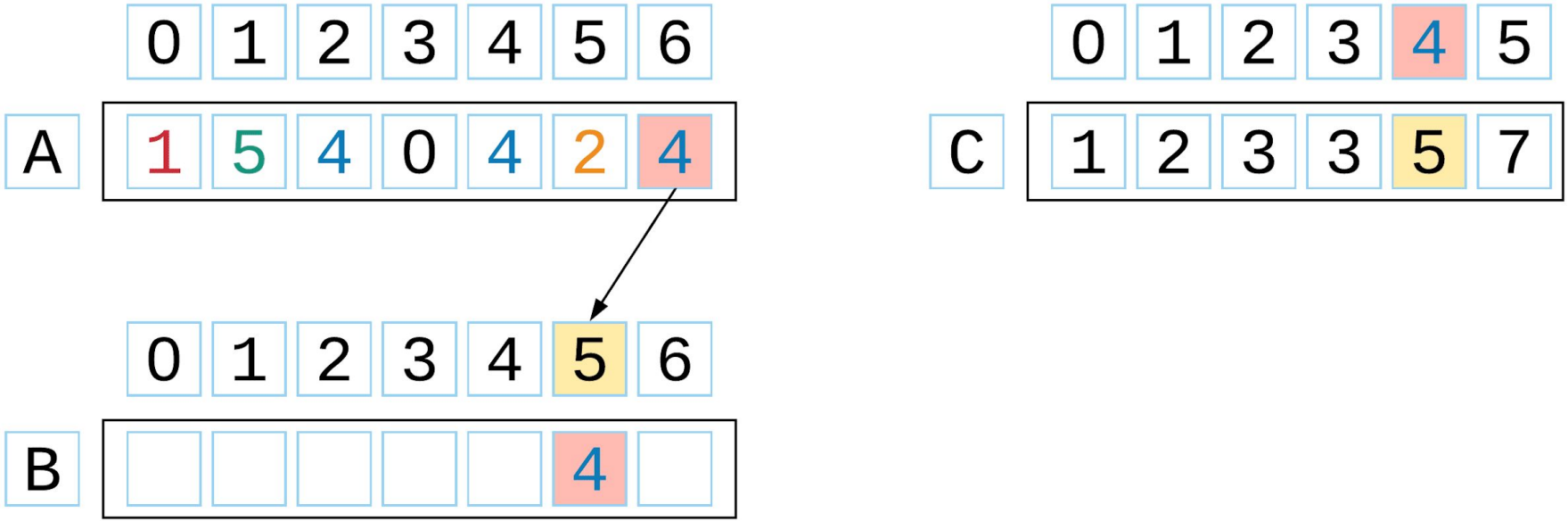
Counting-sort (Serial) - Loop 3



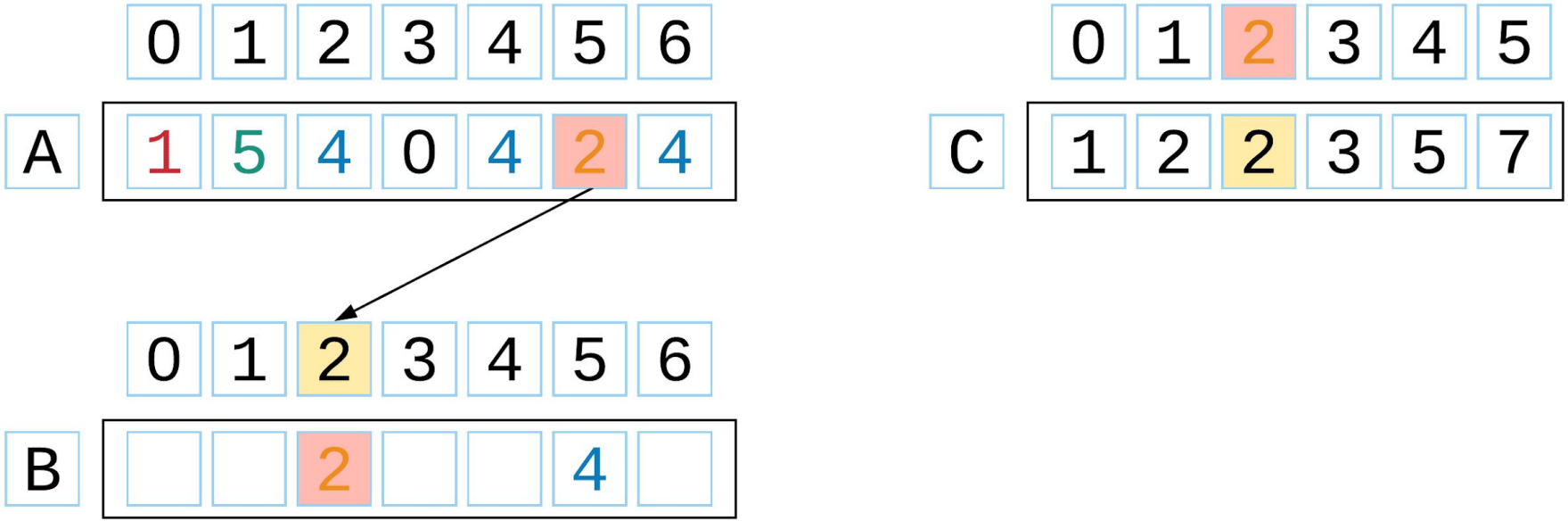
Counting-sort (Serial) - Loop 4



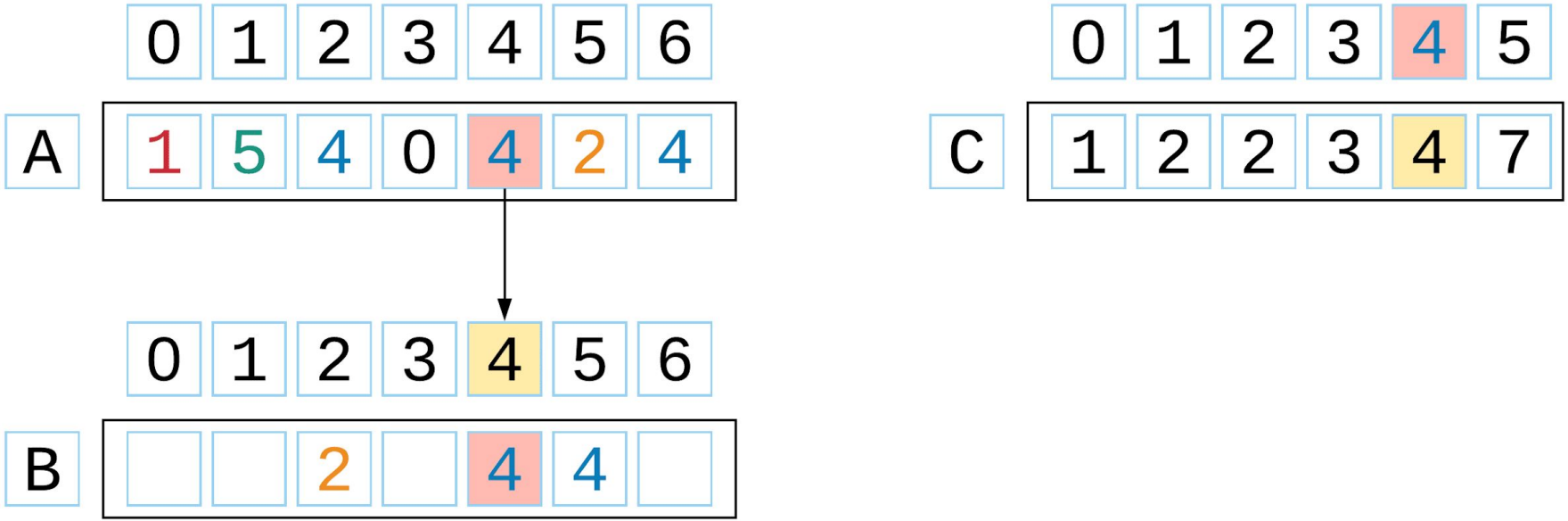
Counting-sort (Serial) - Loop 4



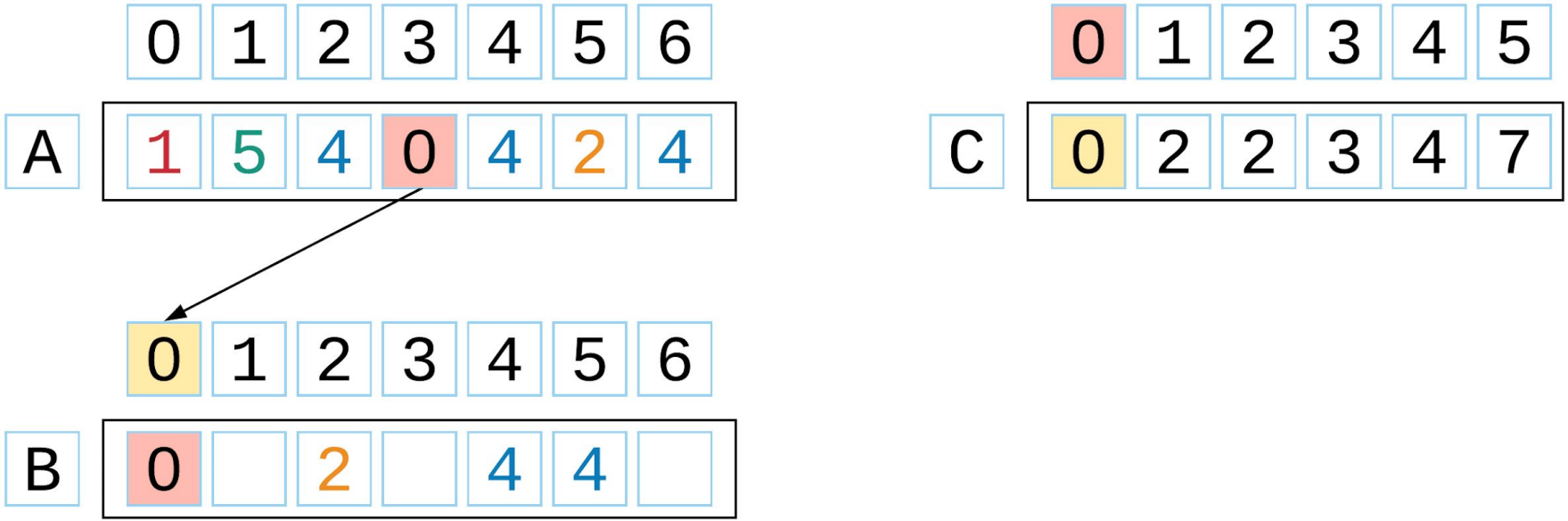
Counting-sort (Serial) - Loop 4



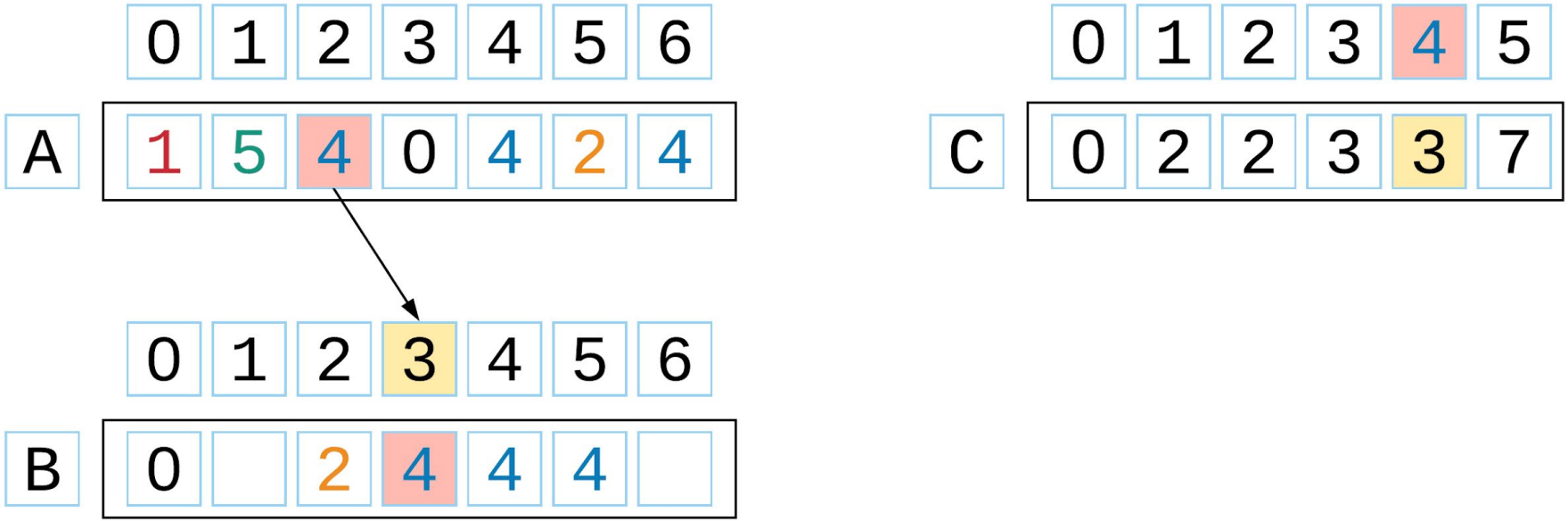
Counting-sort (Serial) - Loop 4



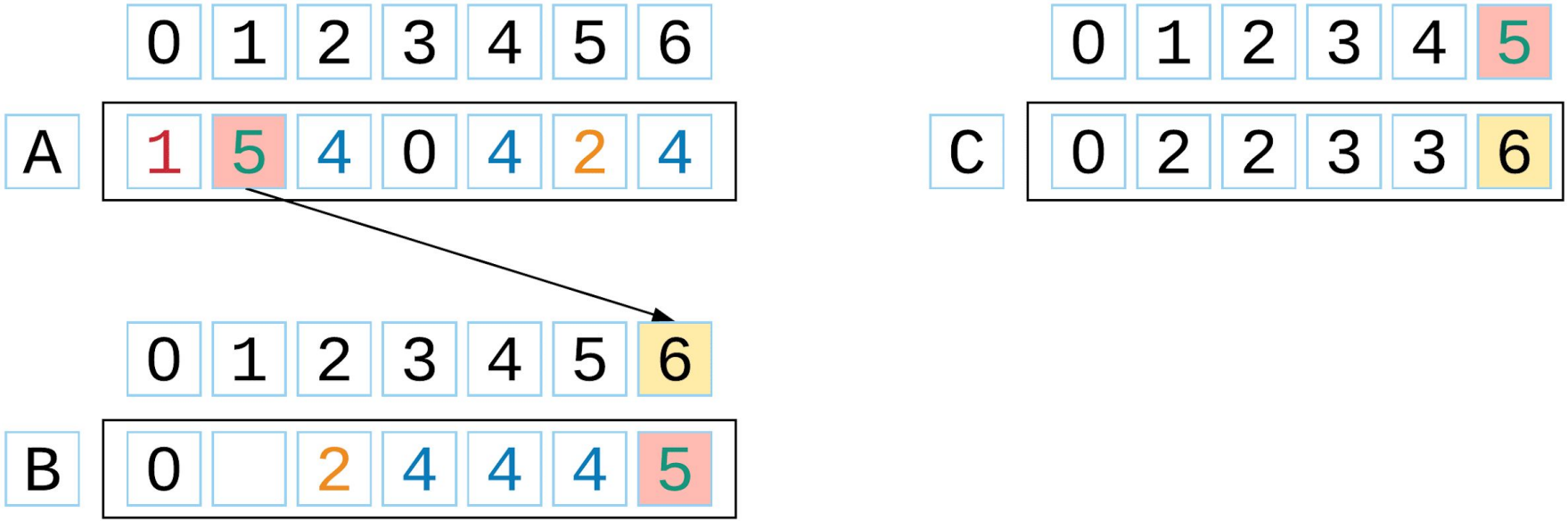
Counting-sort (Serial) - Loop 4



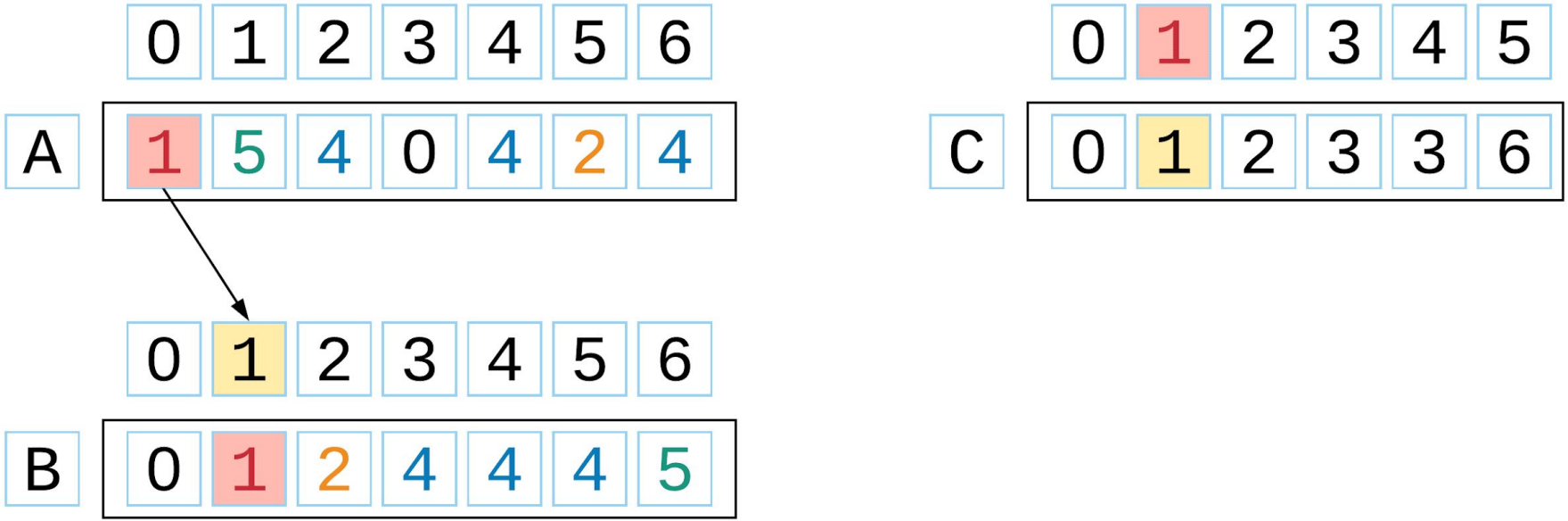
Counting-sort (Serial) - Loop 4



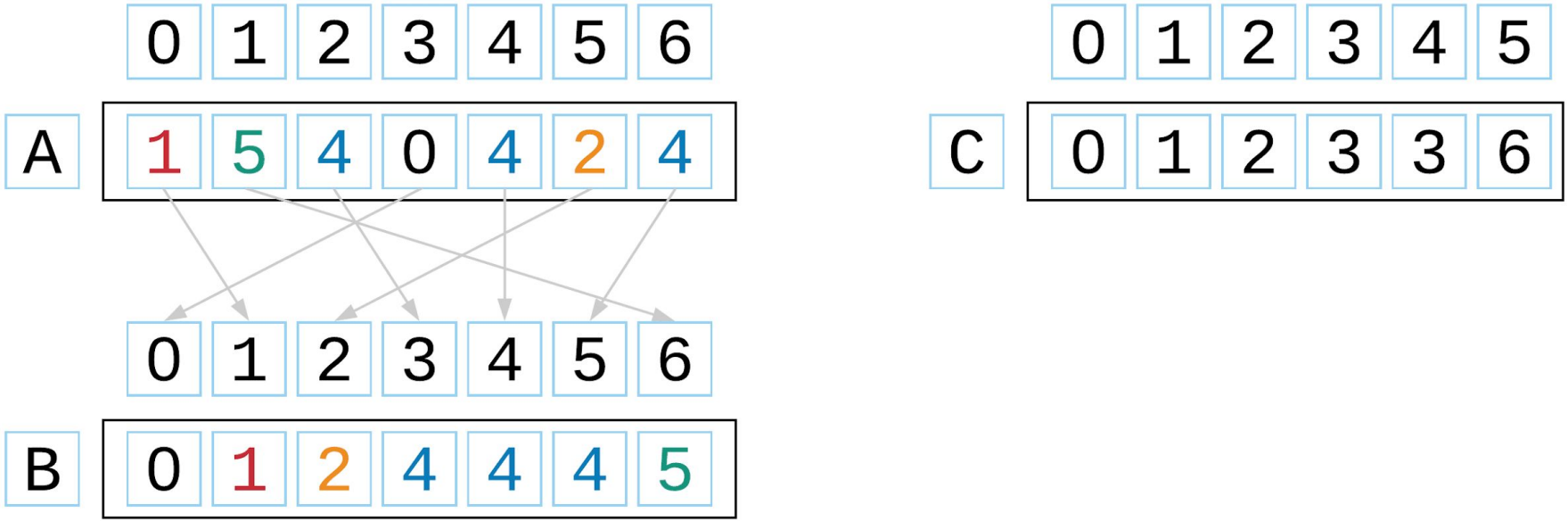
Counting-sort (Serial) - Loop 4



Counting-sort (Serial) - Loop 4



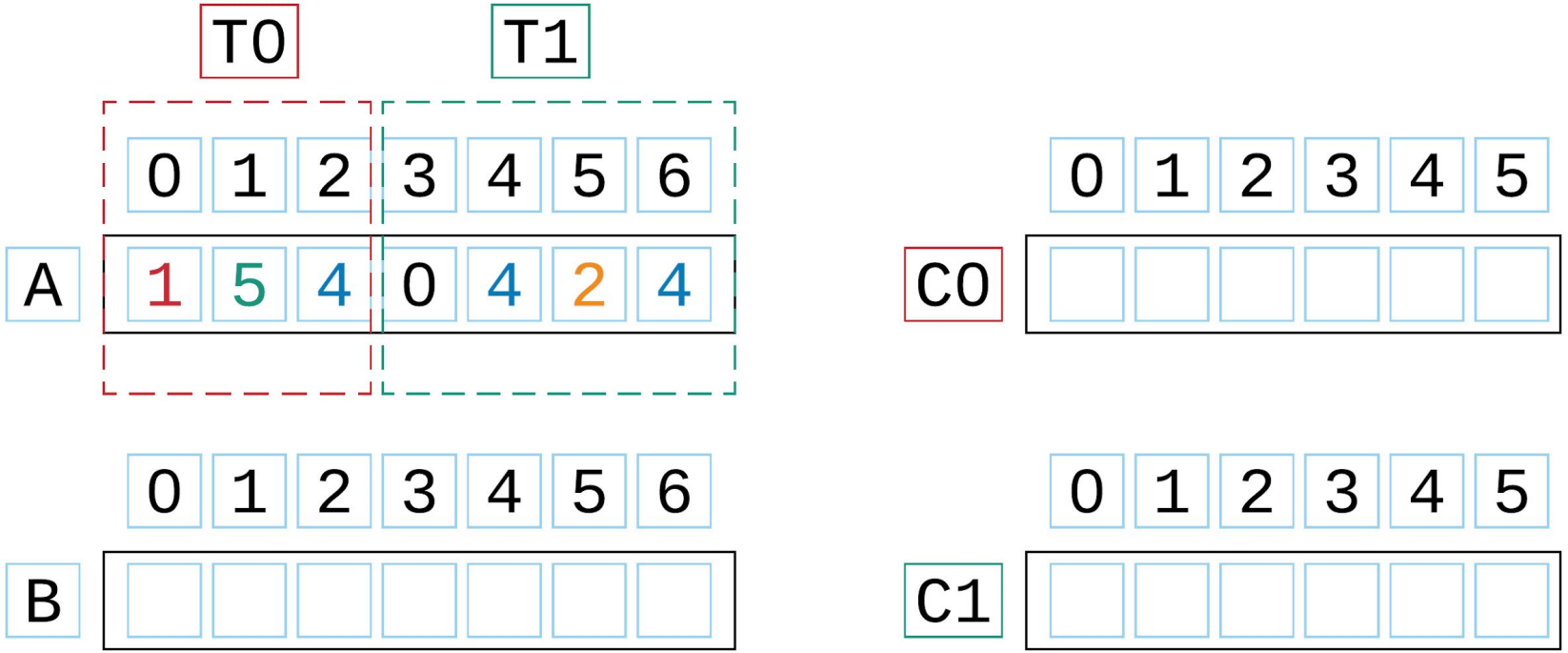
Counting-sort (Serial) - Loop 4



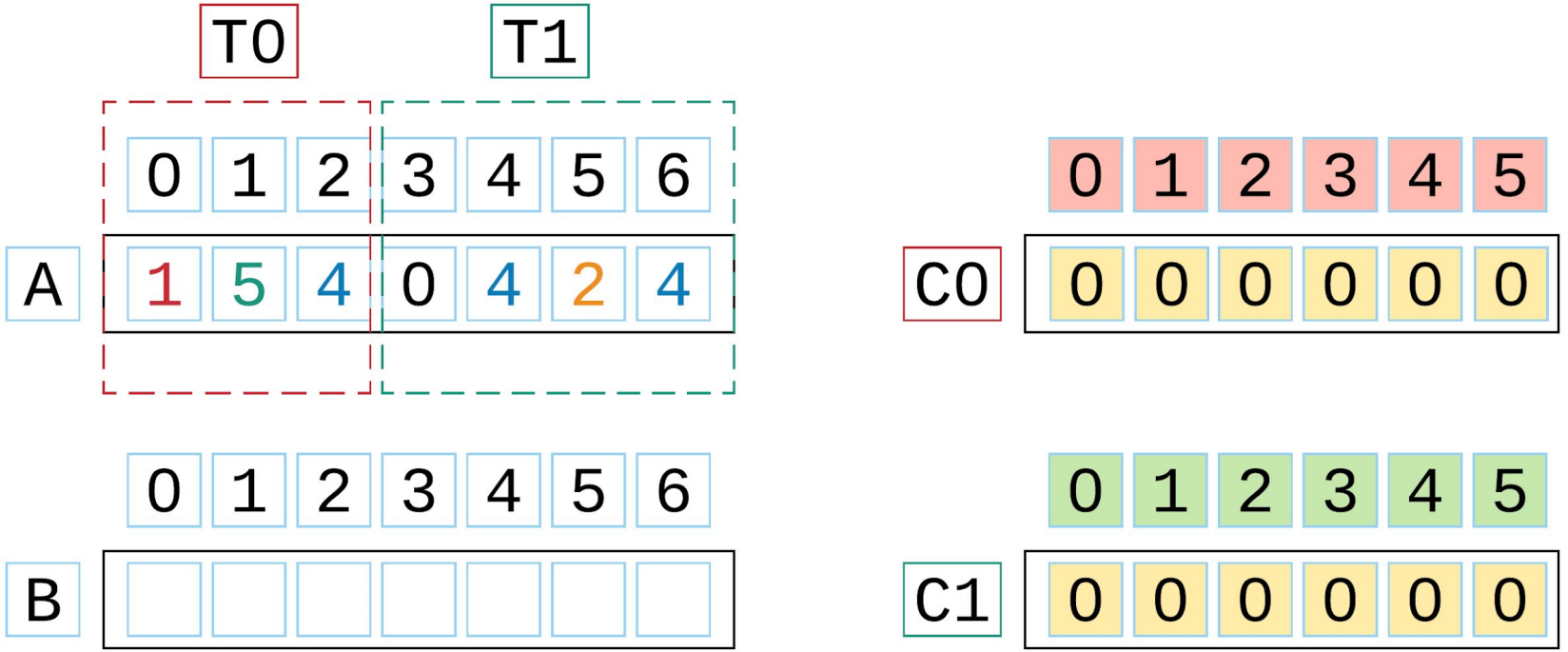
Counting-sort (Parallel)

```
base←0
Parallel for i←0 to (k-1)
   do C[tid][i]←0
Parallel for j ←tid_start to tid_end
   do C[tid][A[j]]←C[tid][A[j]]+1
Parallel barrier
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
Parallel barrier
Parallel for j ←(tid_end-1) downto tid_start
   do B[C[tid][A[j]]-1] ←A[j]
      C[tid][A[j]]←C[tid][A[j]]-1
```

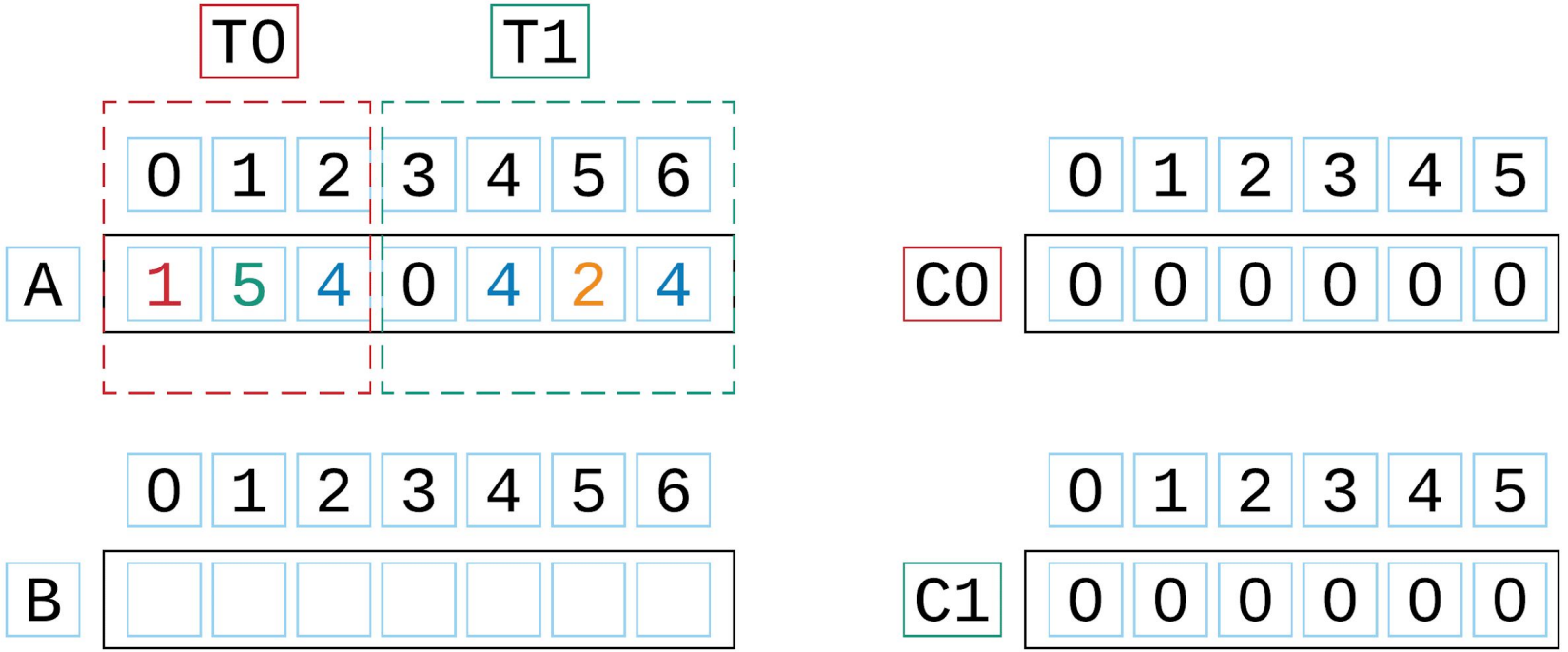
Counting-sort (Parallel)



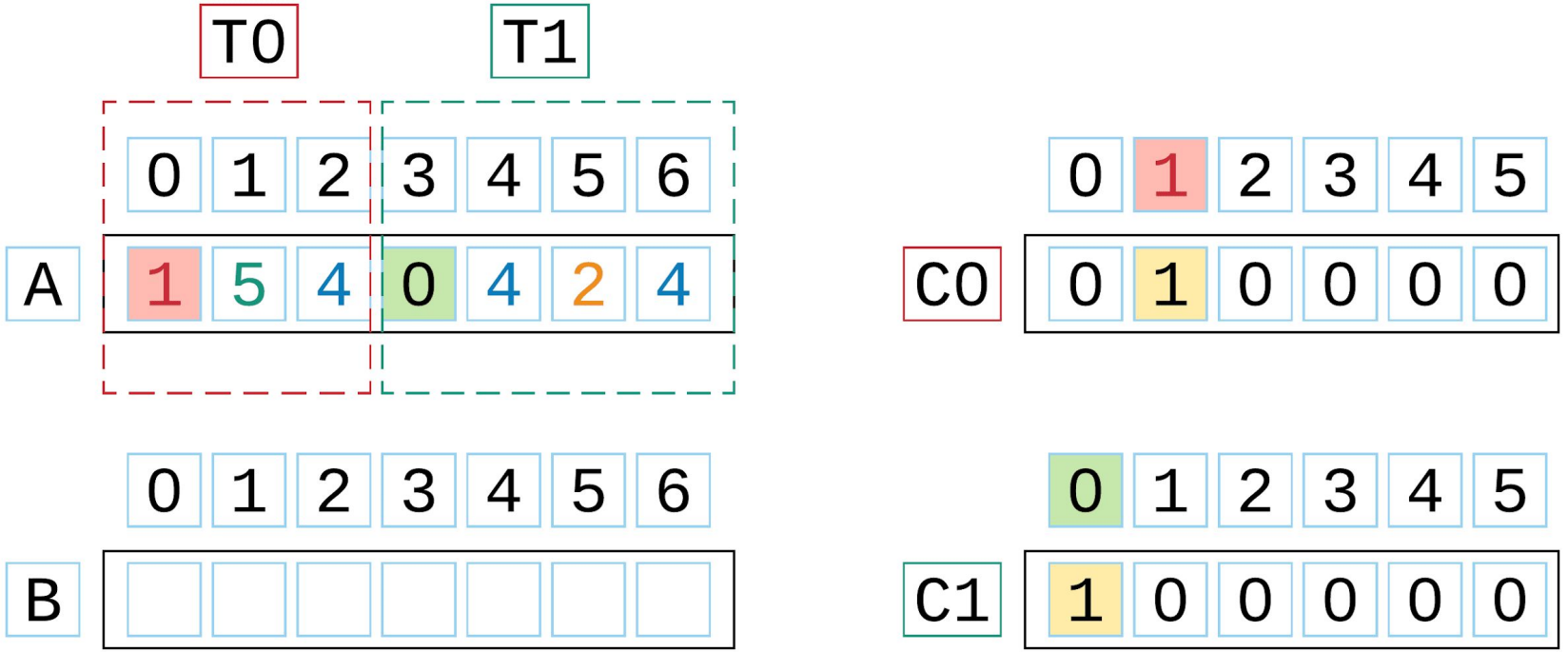
Counting-sort (Parallel) - Loop 1



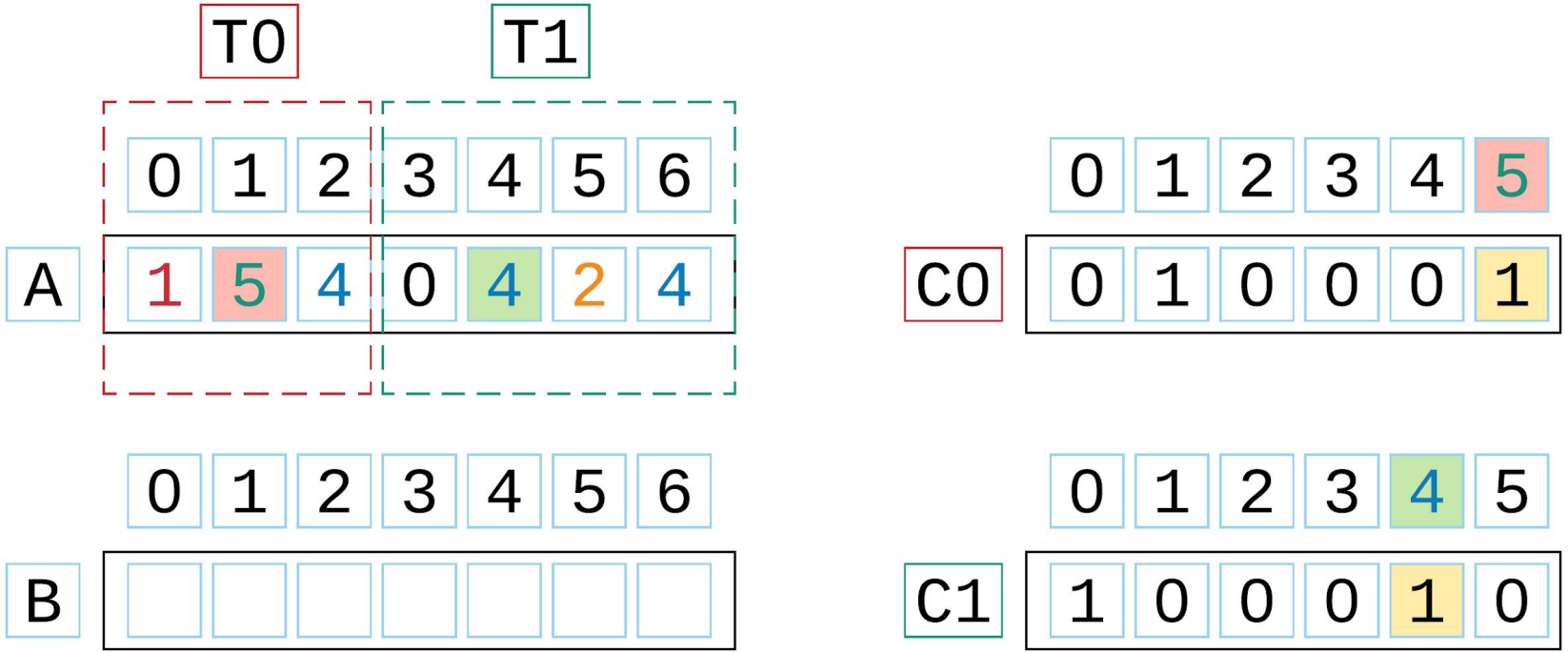
Counting-sort (Parallel) - Loop 2



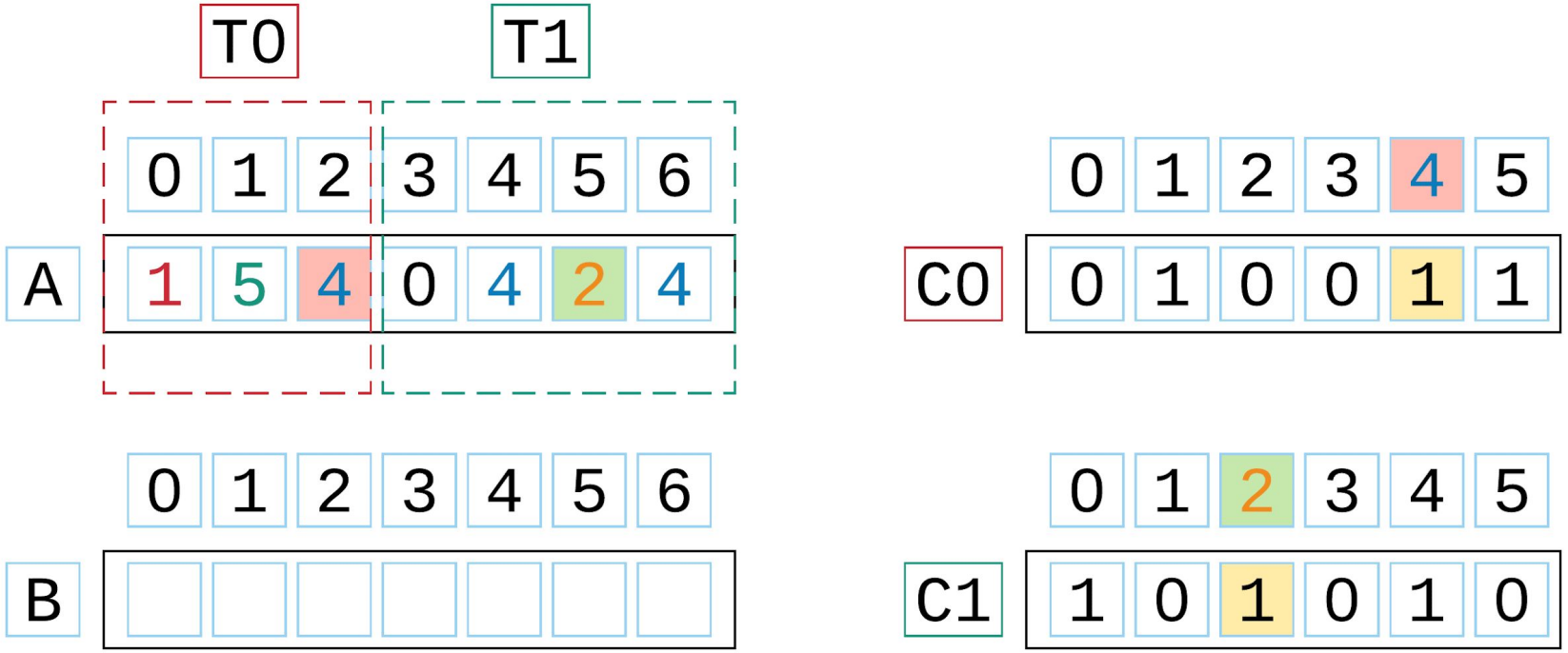
Counting-sort (Parallel) - Loop 2



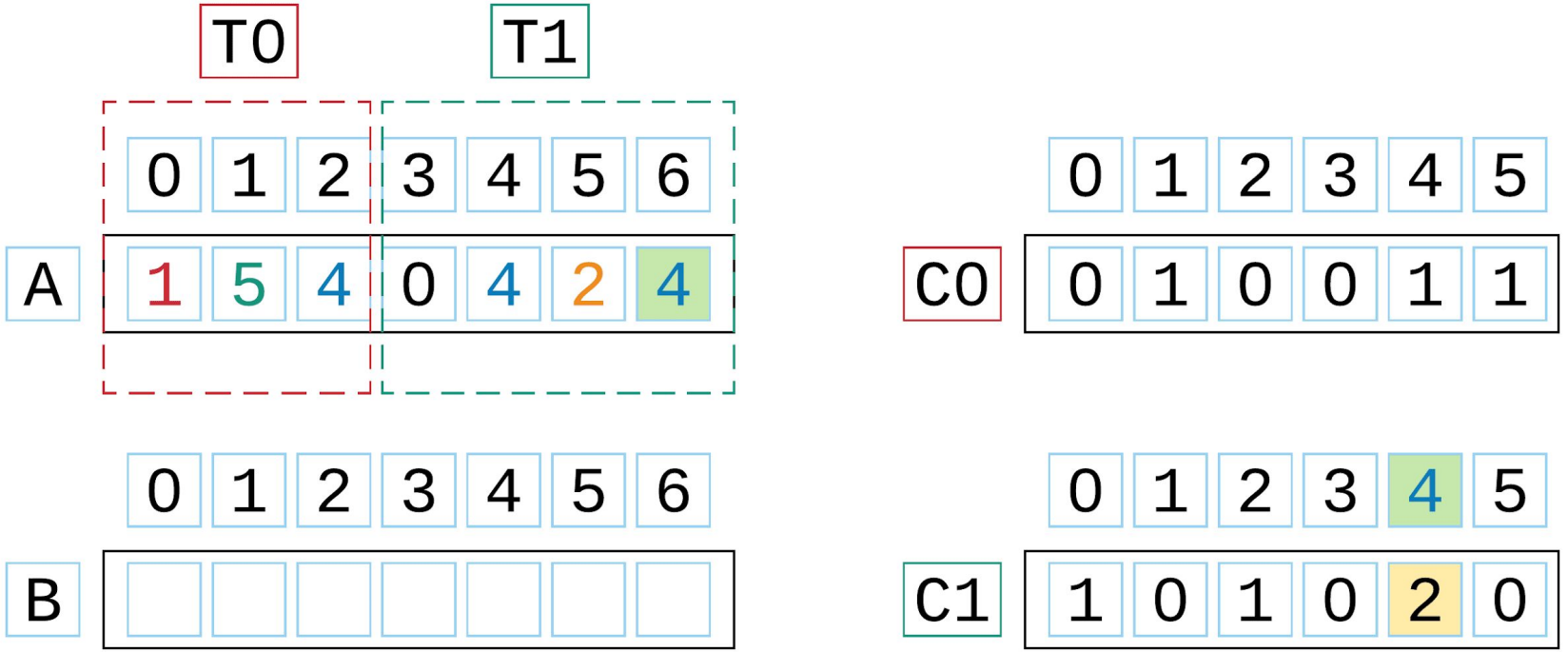
Counting-sort (Parallel) - Loop 2



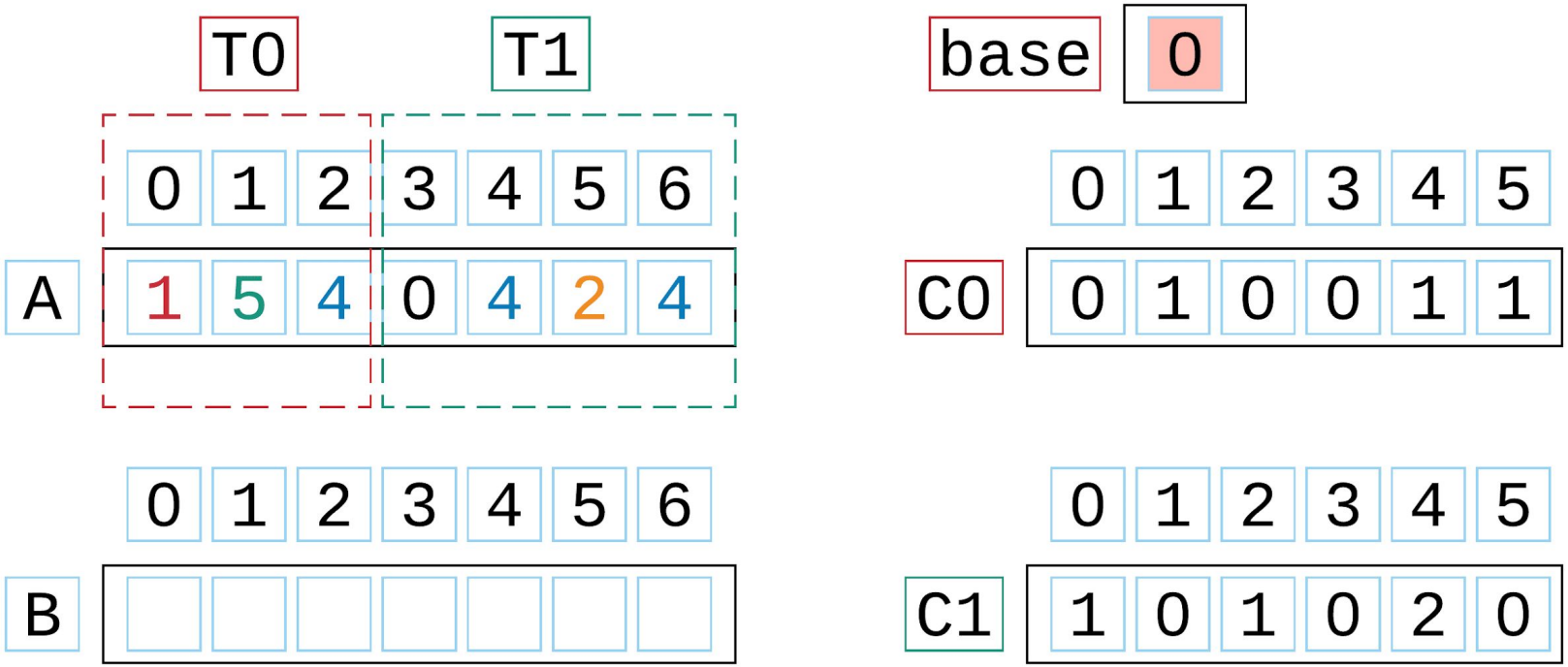
Counting-sort (Parallel) - Loop 2



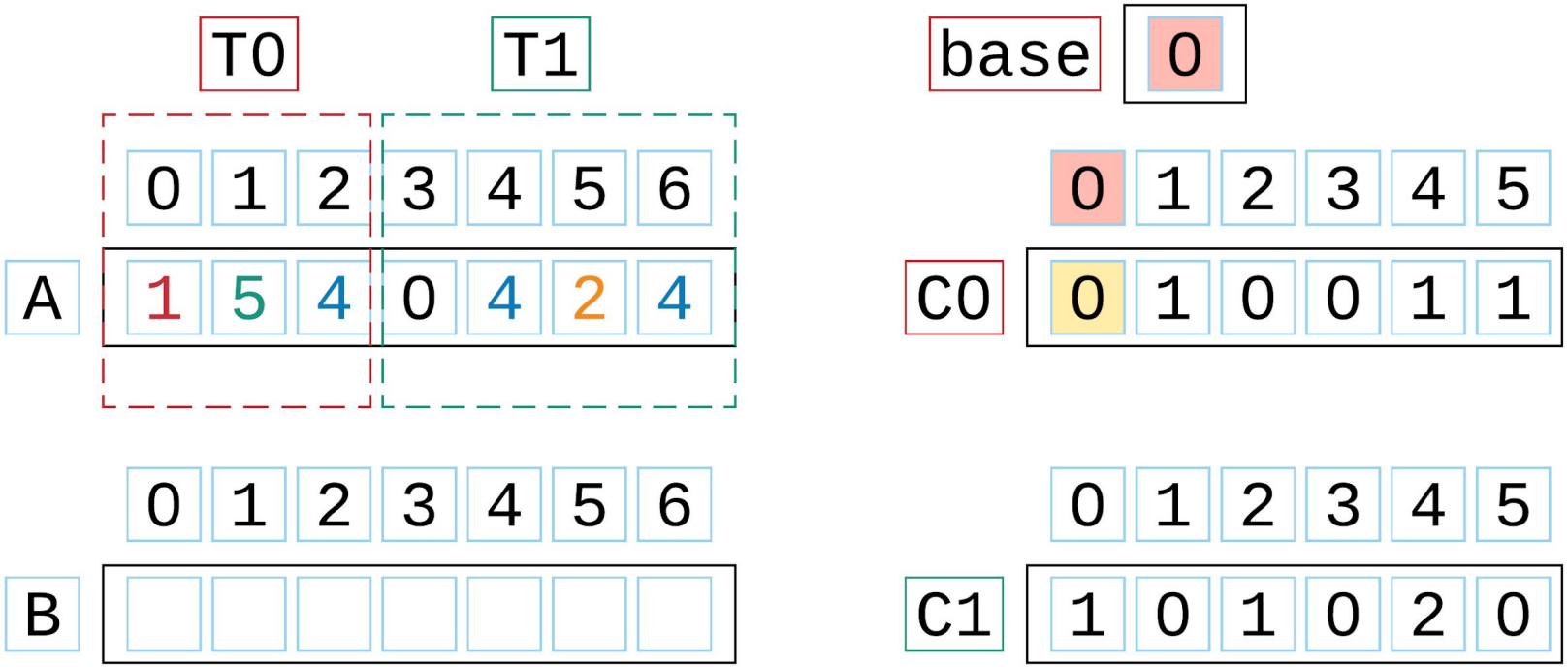
Counting-sort (Parallel) - Loop 2



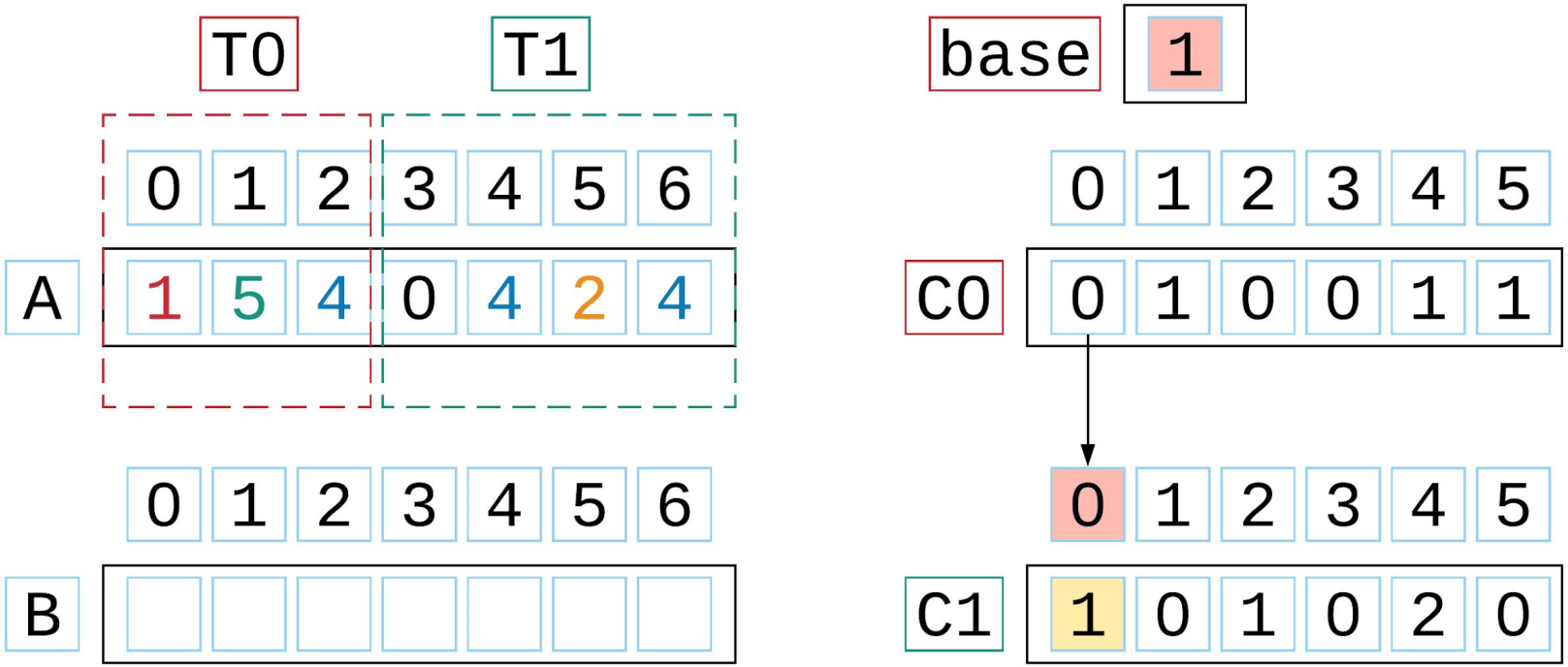
Counting-sort (Parallel) - Loop 3



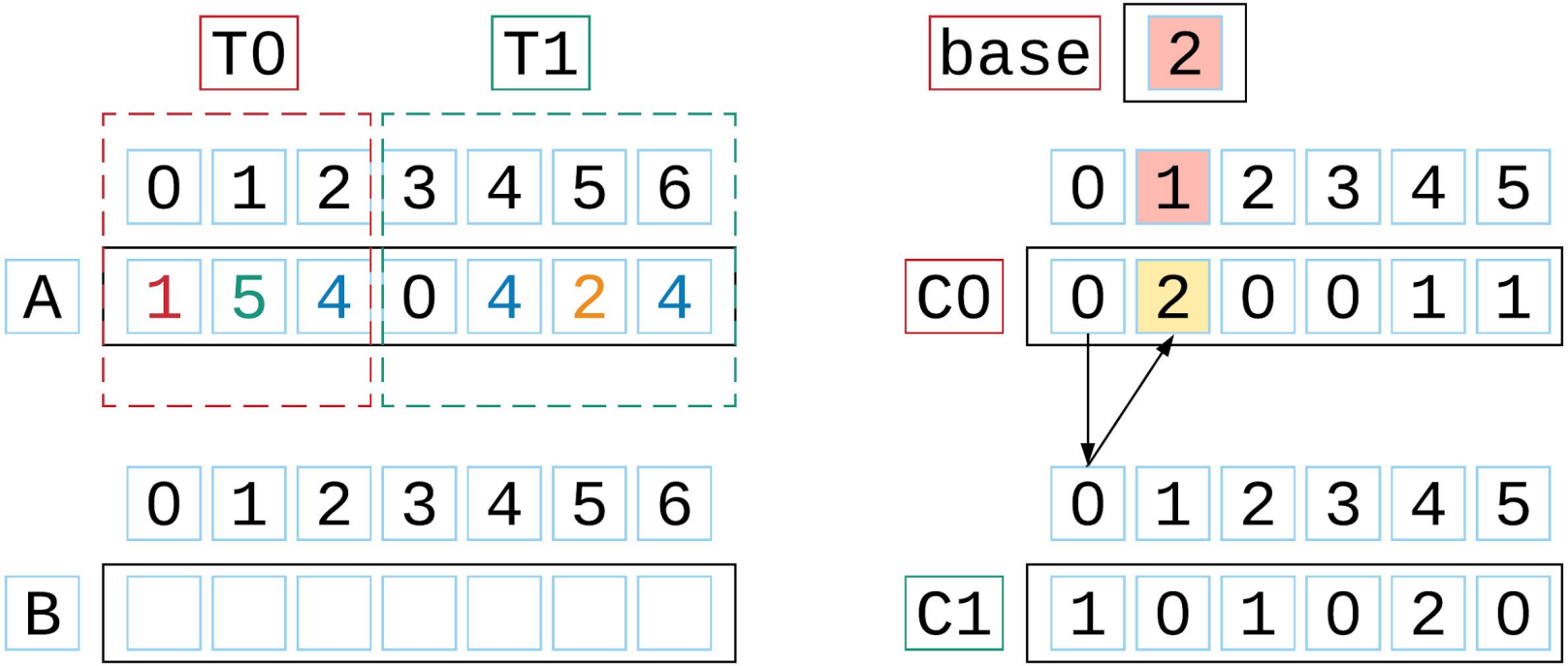
Counting-sort (Parallel) - Loop 3



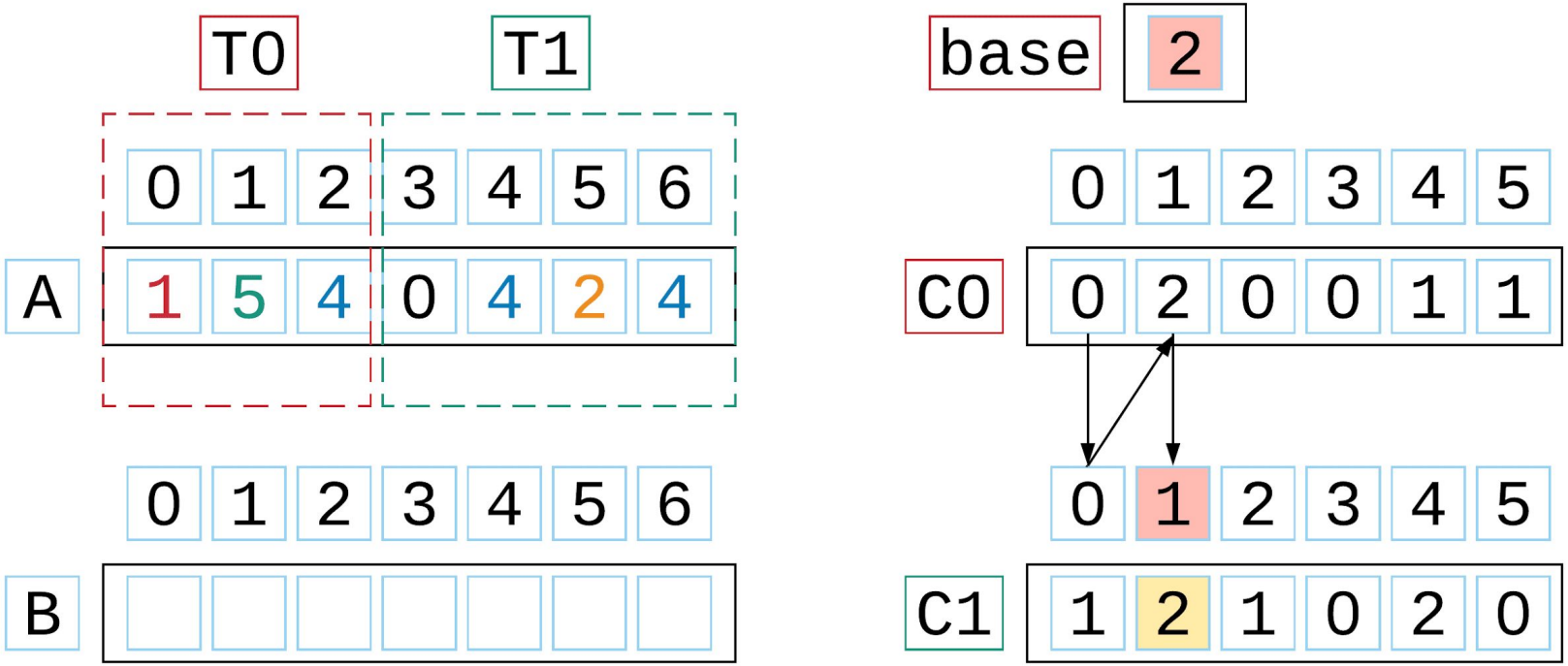
Counting-sort (Parallel) - Loop 3



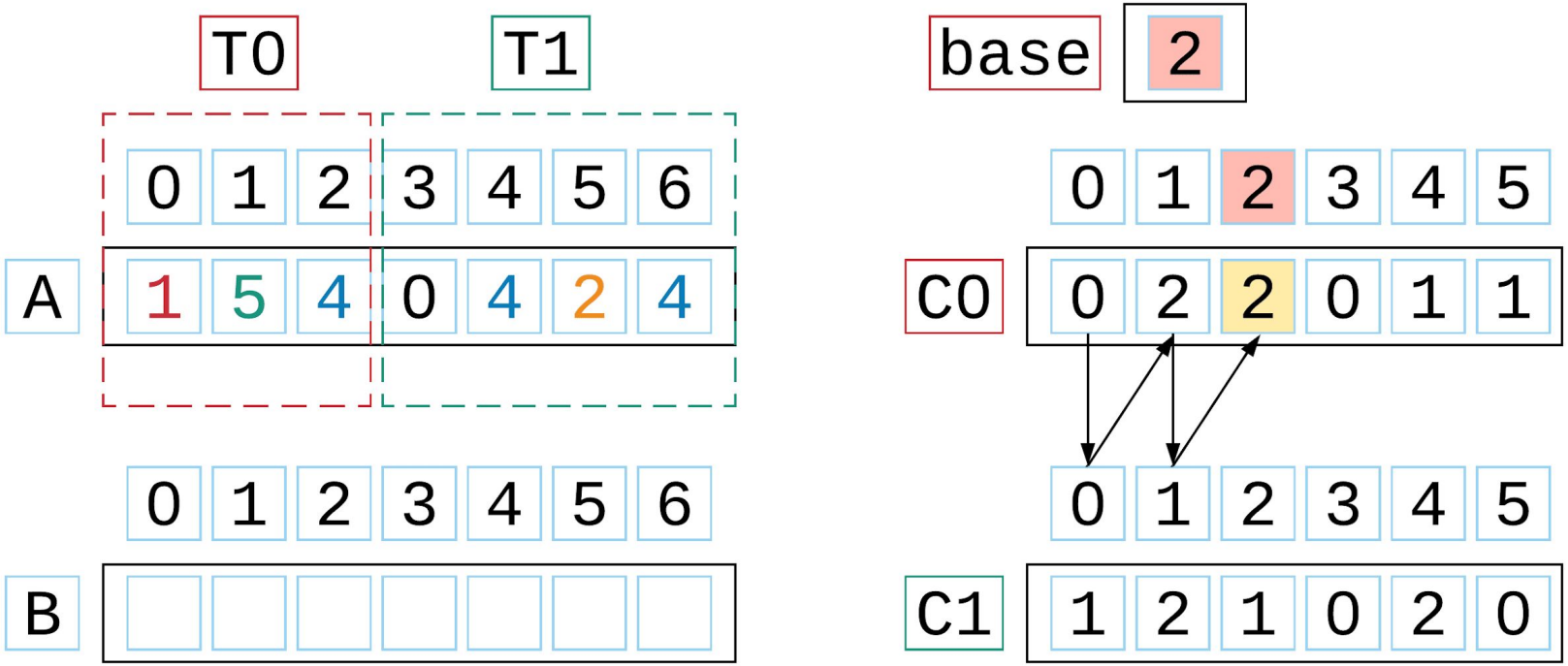
Counting-sort (Parallel) - Loop 3



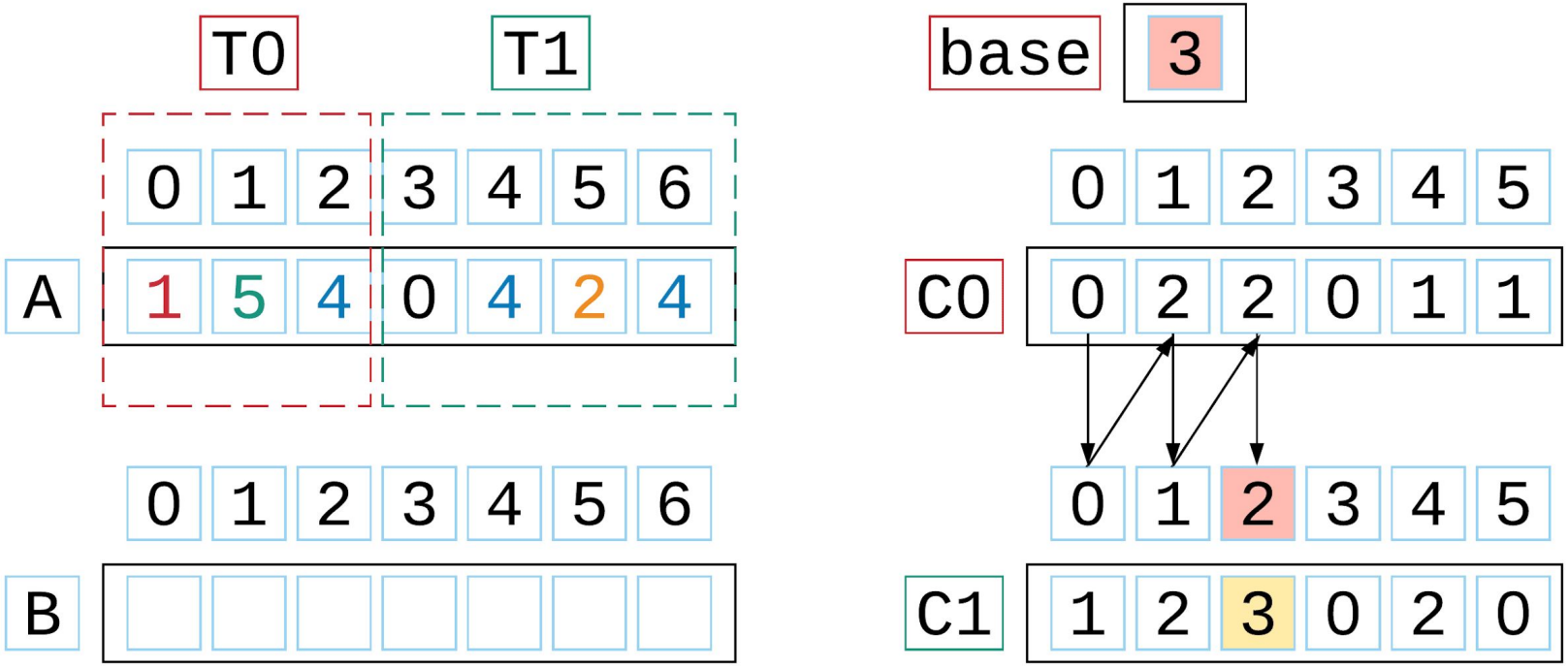
Counting-sort (Parallel) - Loop 3



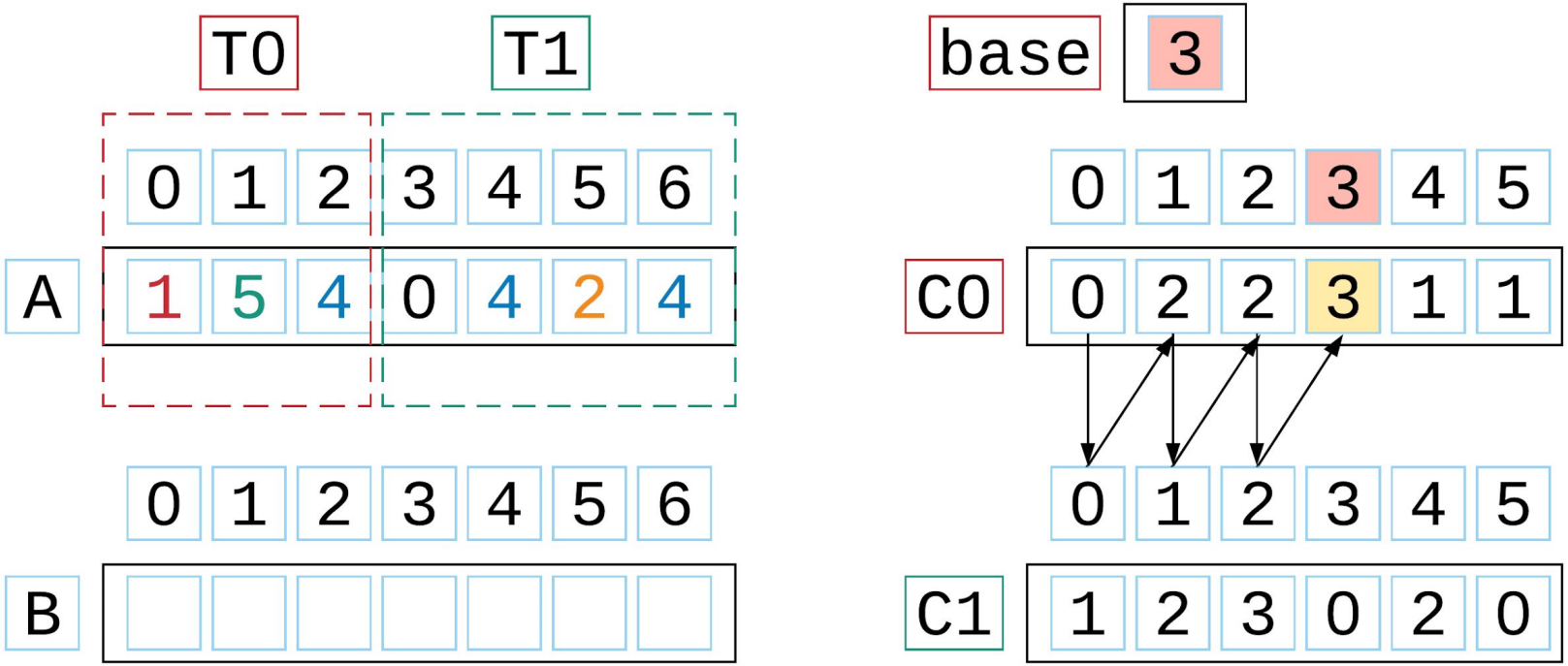
Counting-sort (Parallel) - Loop 3



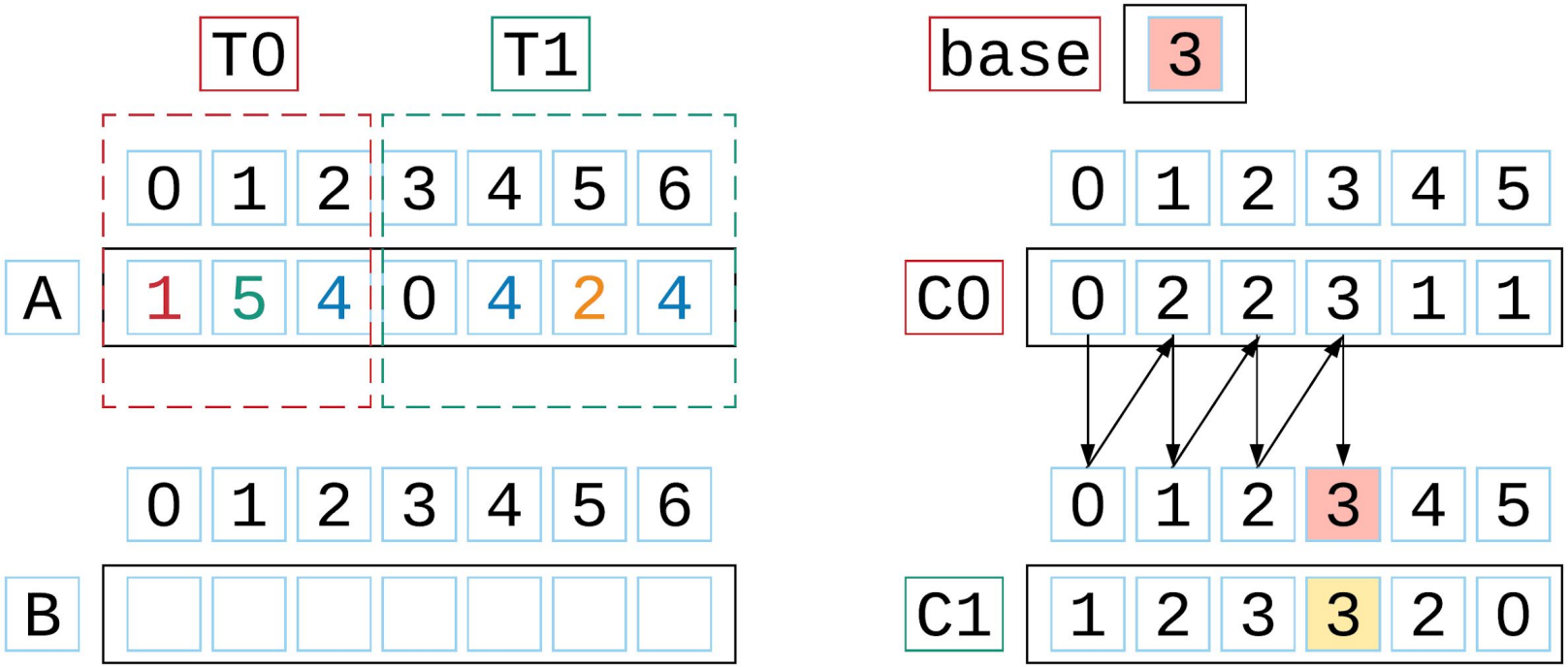
Counting-sort (Parallel) - Loop 3



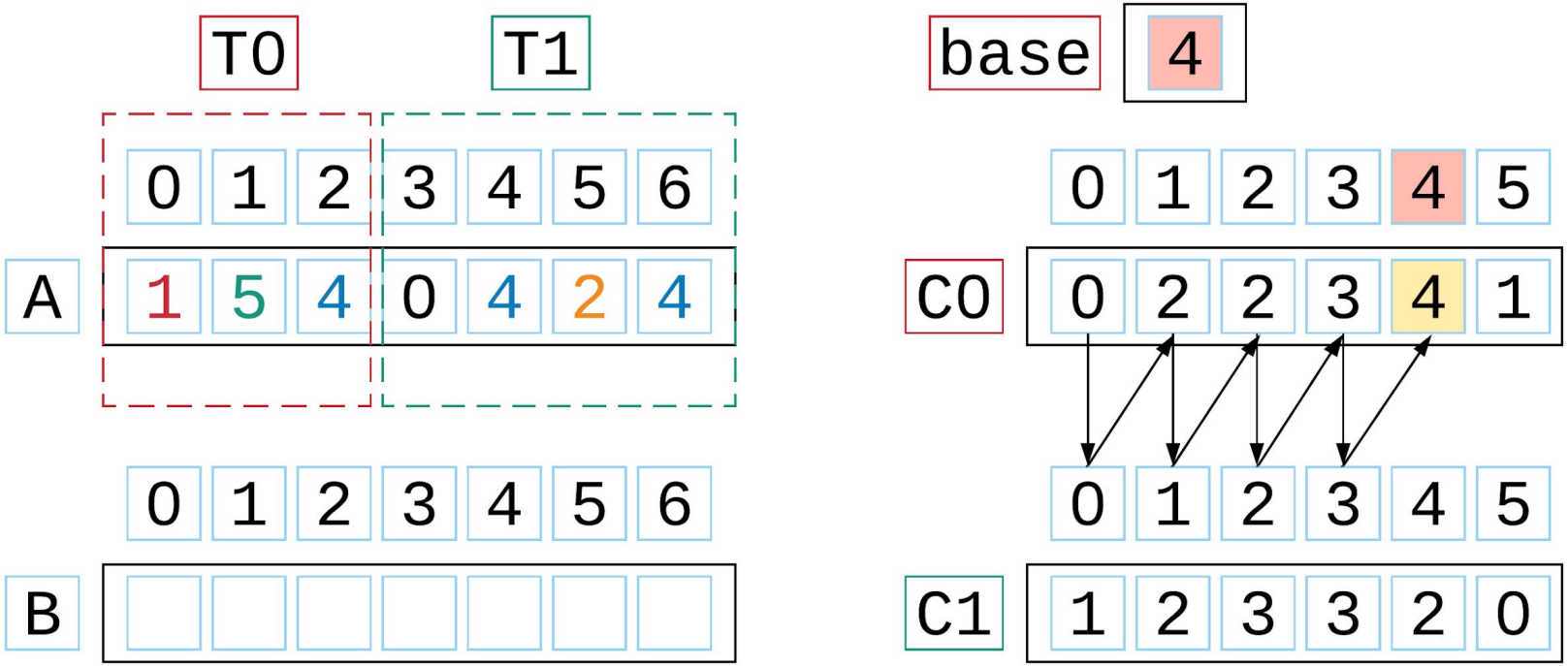
Counting-sort (Parallel) - Loop 3



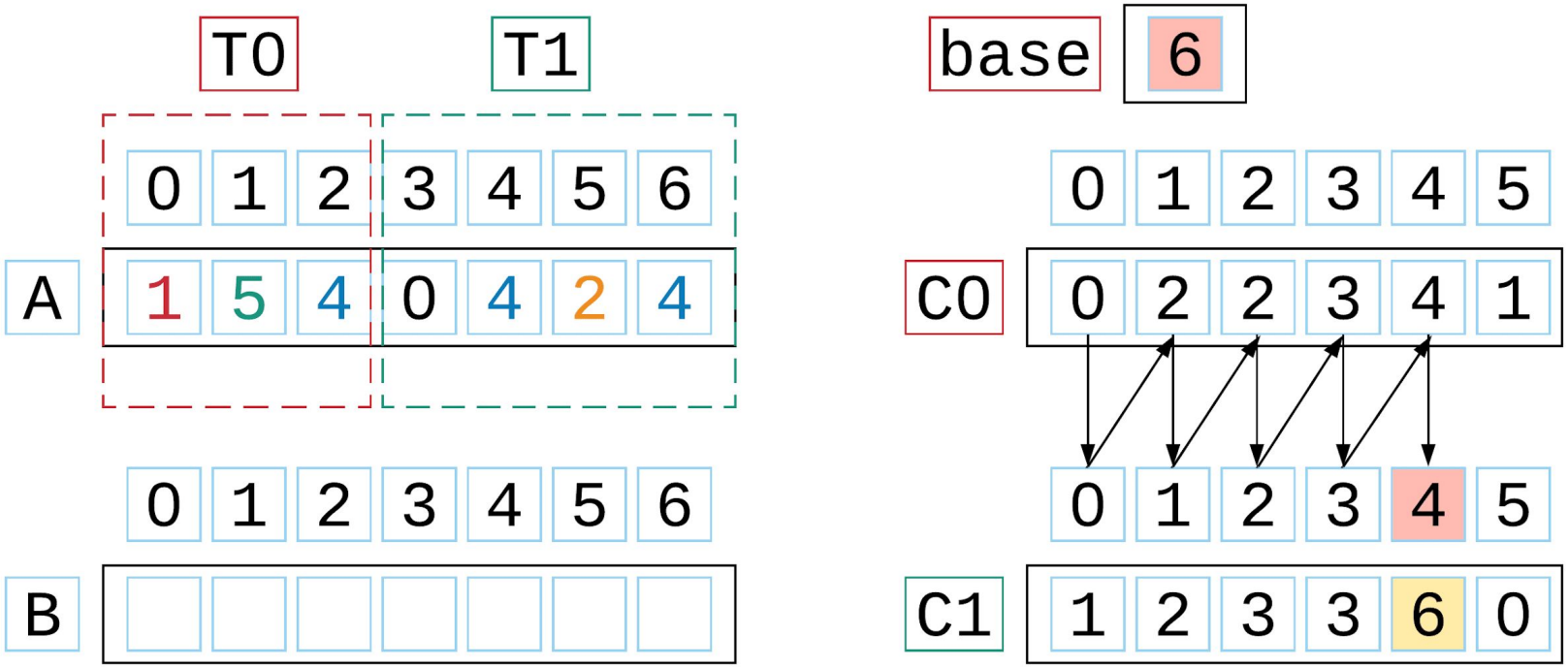
Counting-sort (Parallel) - Loop 3



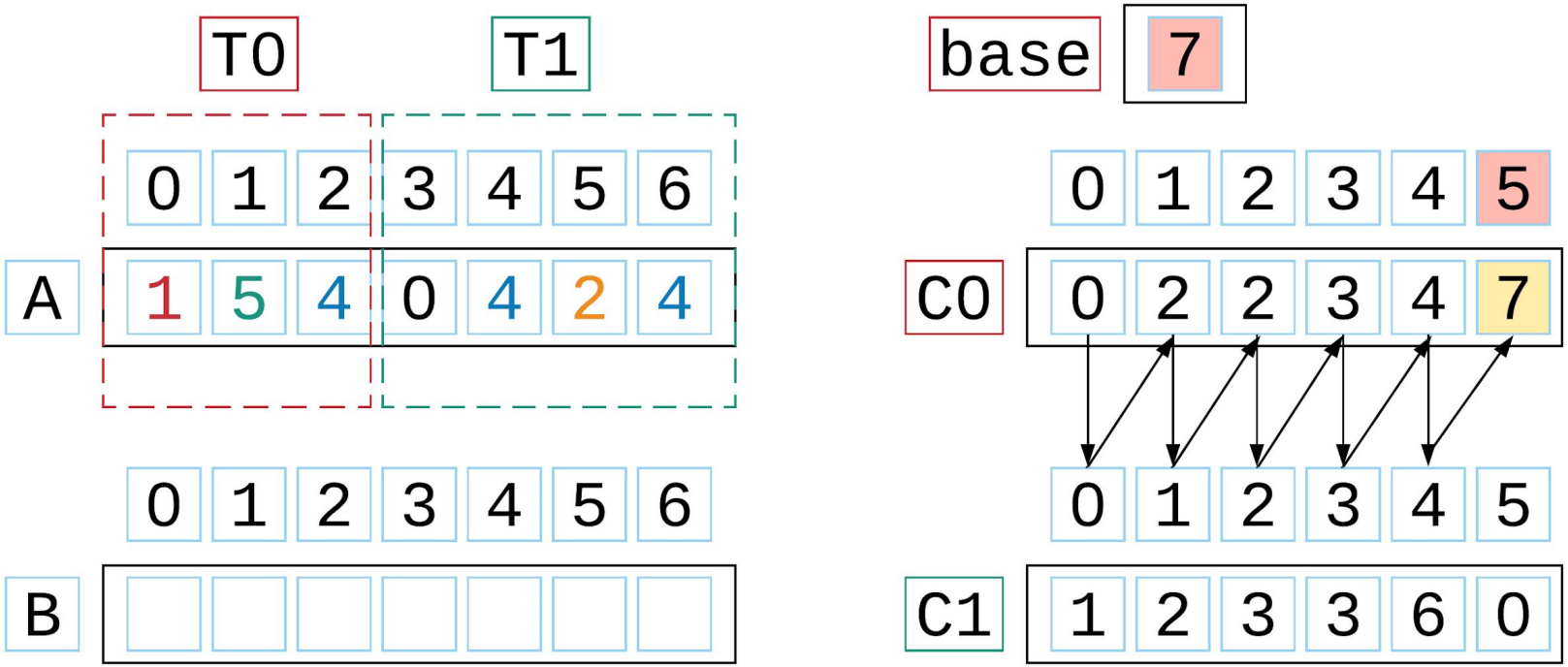
Counting-sort (Parallel) - Loop 3



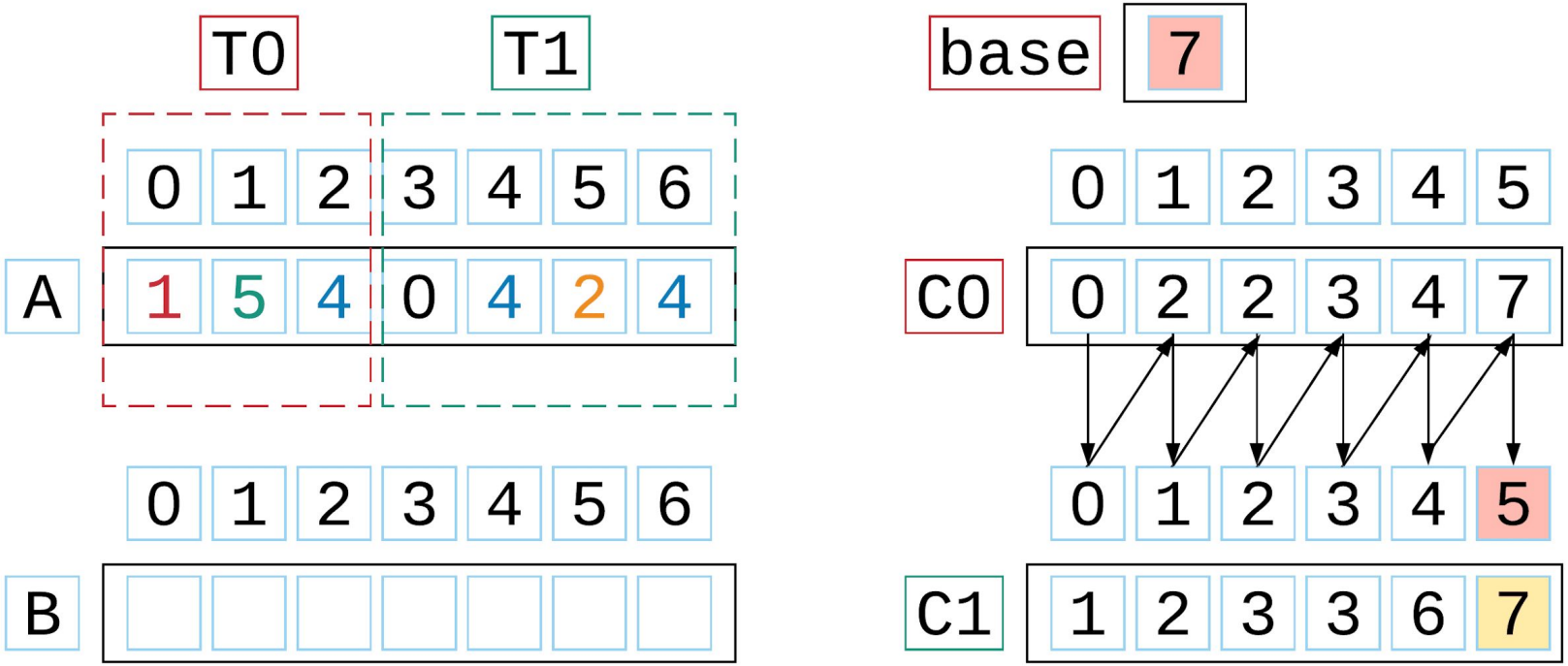
Counting-sort (Parallel) - Loop 3



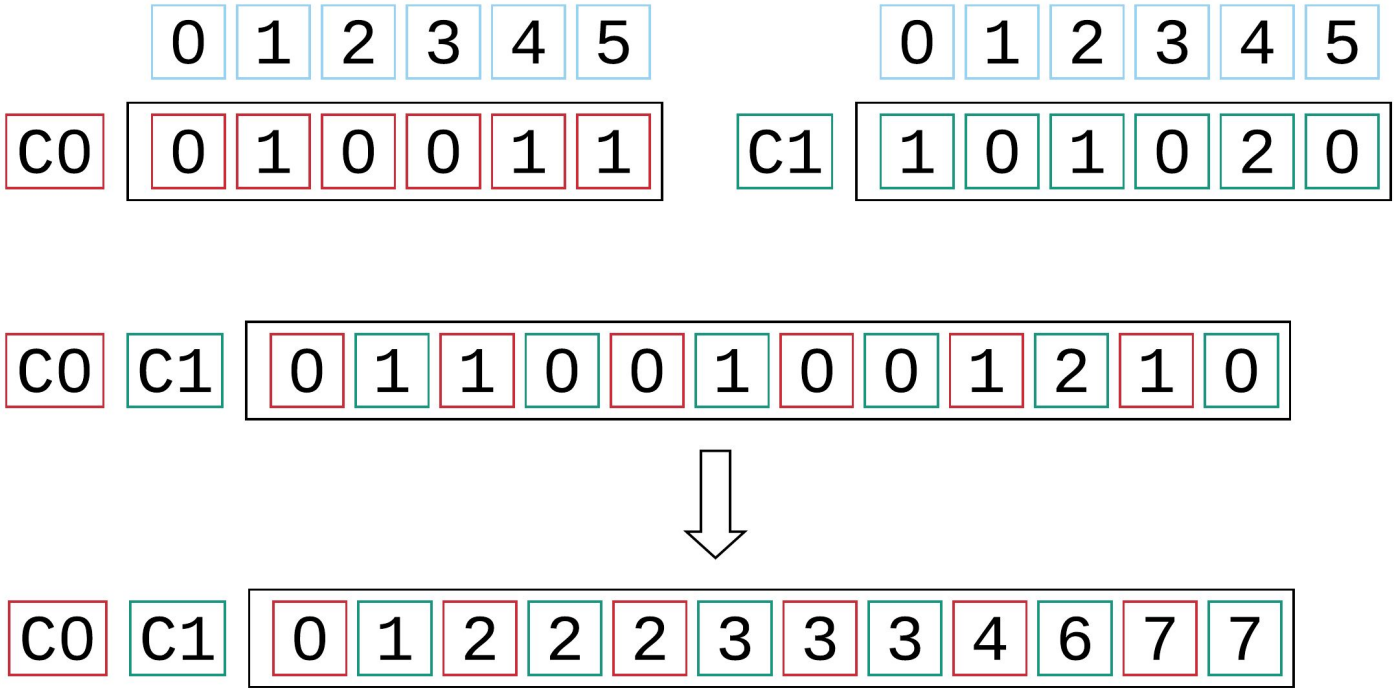
Counting-sort (Parallel) - Loop 3



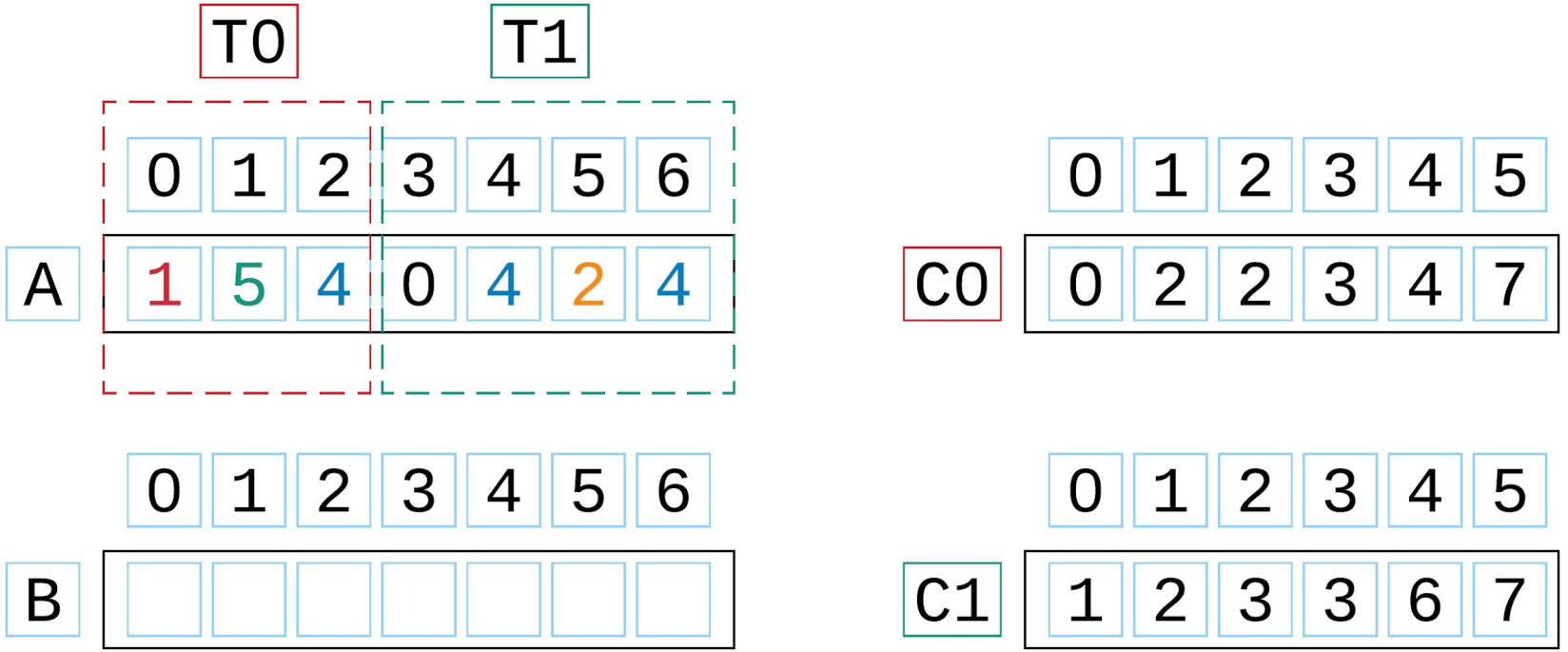
Counting-sort (Parallel) - Loop 3



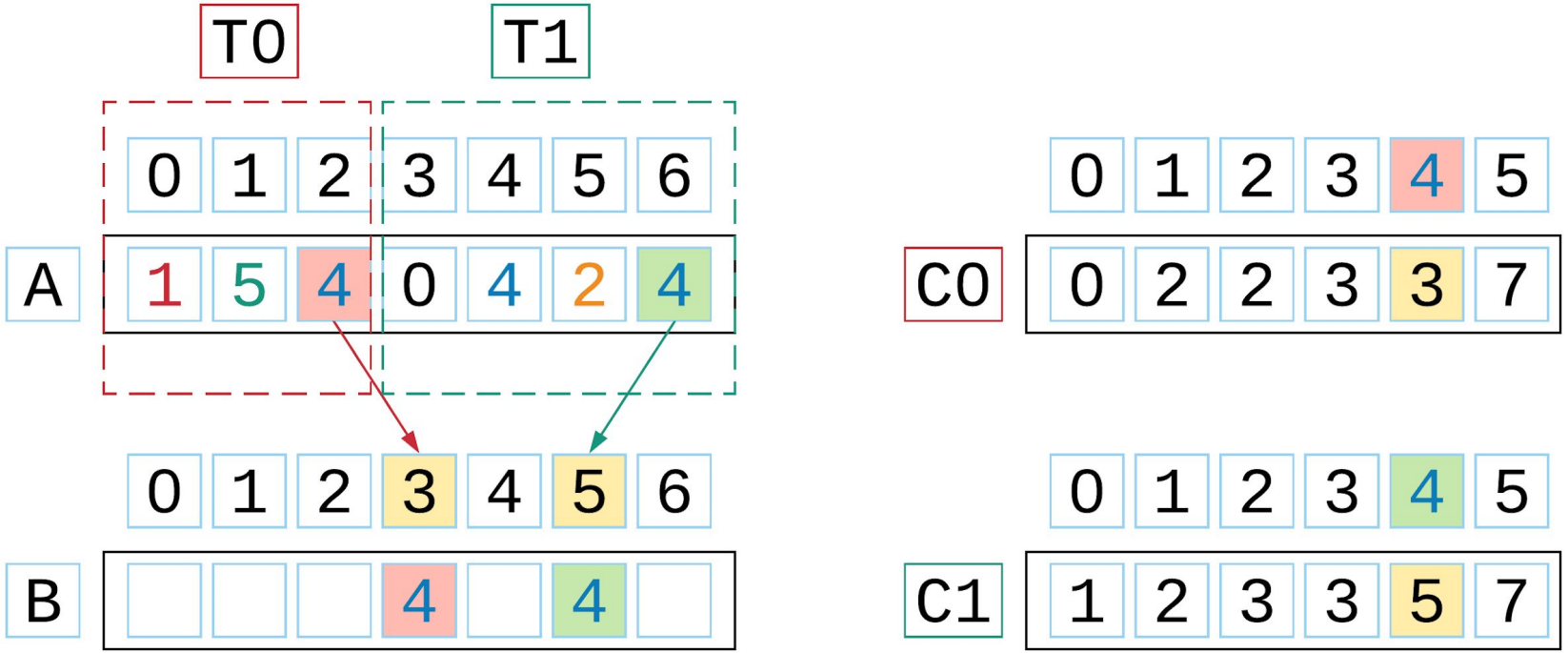
Counting-sort (Parallel) - Loop 3



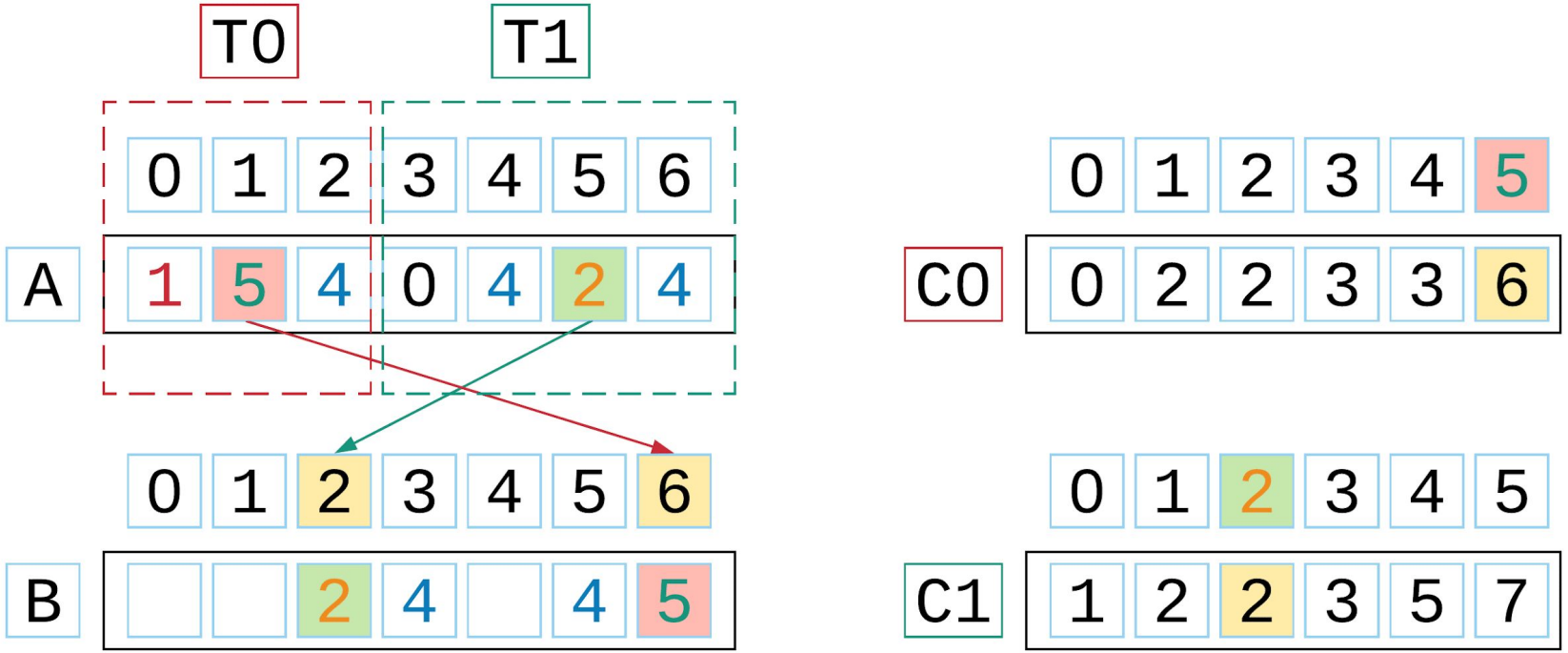
Counting-sort (Parallel) - Loop 4



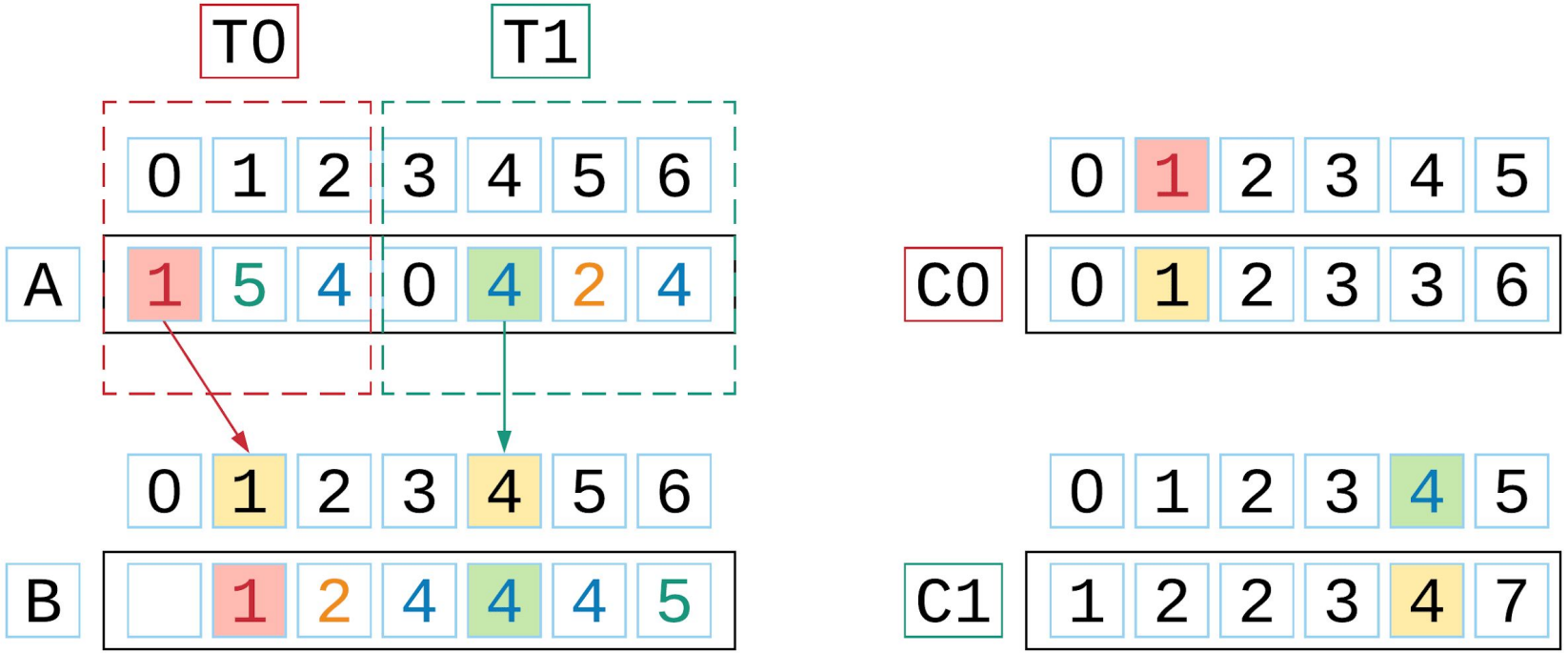
Counting-sort (Parallel) - Loop 4



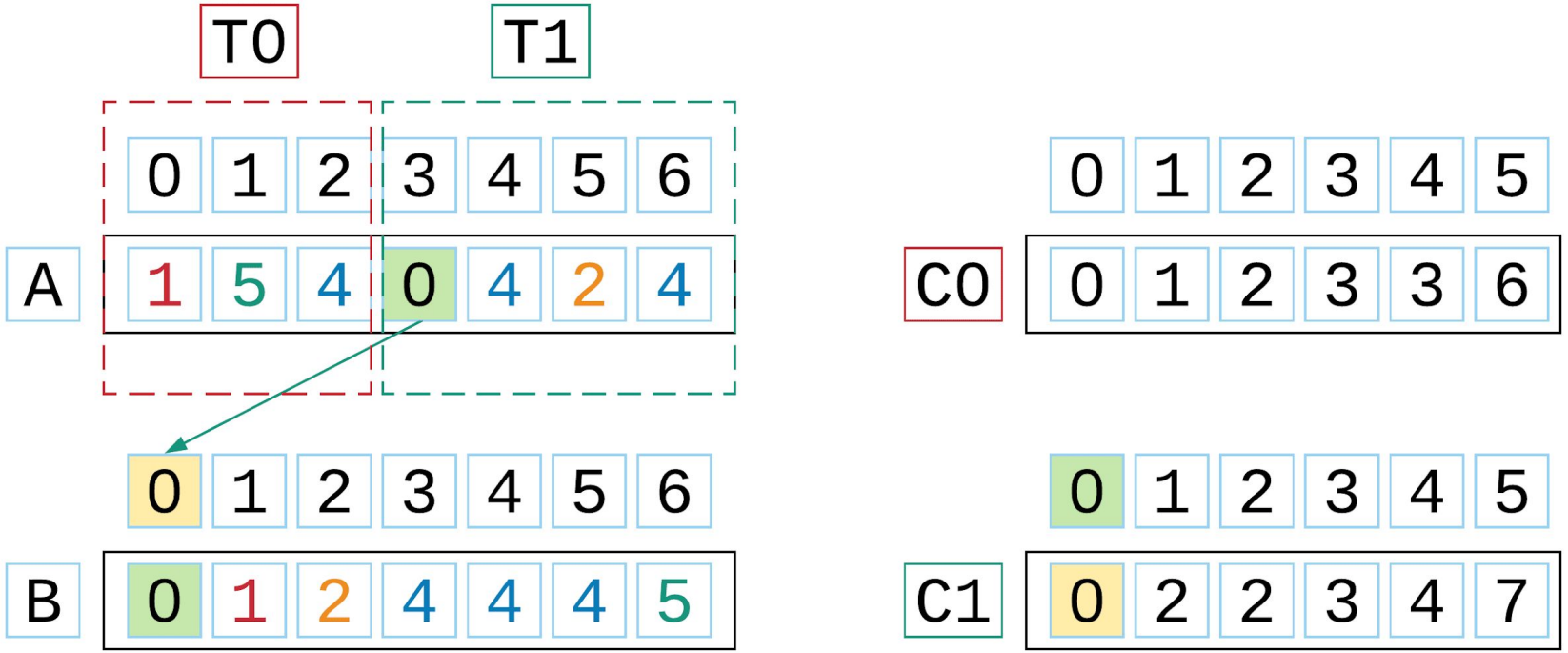
Counting-sort (Parallel) - Loop 4



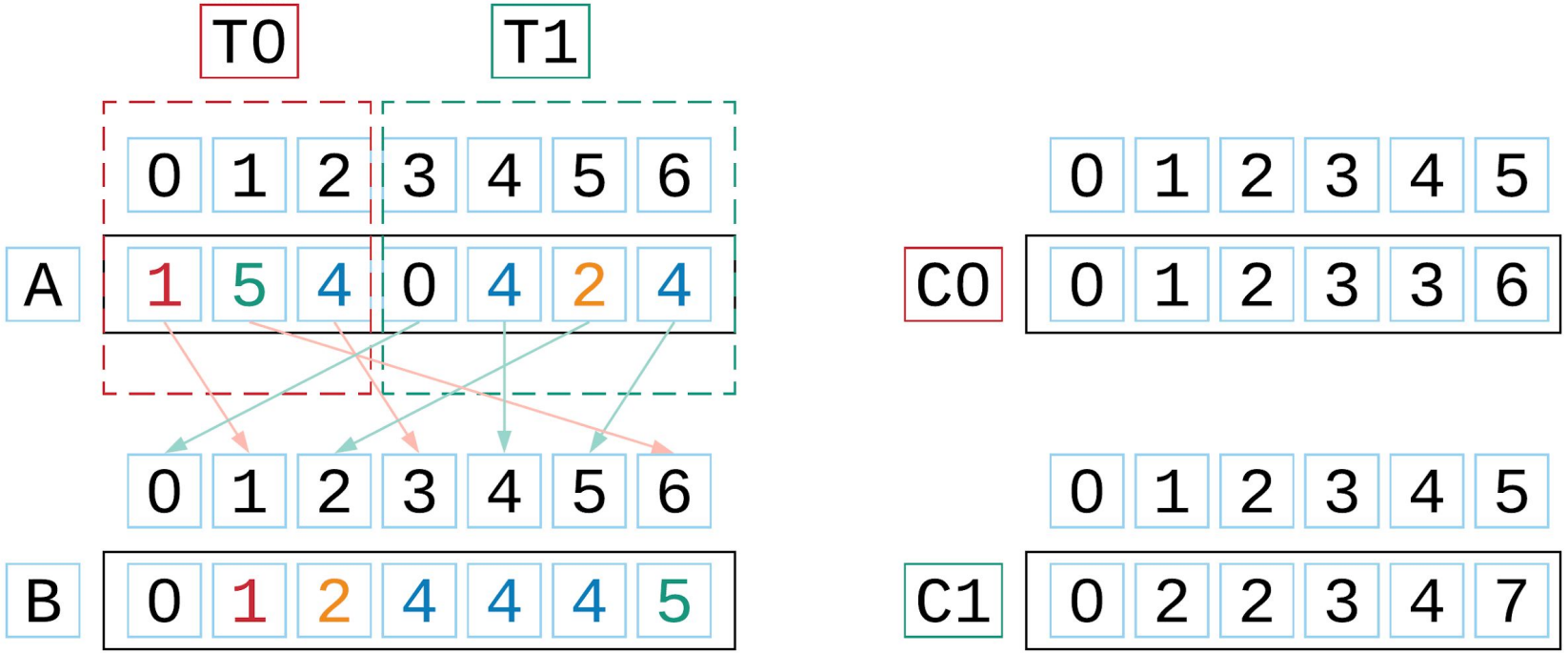
Counting-sort (Parallel) - Loop 4



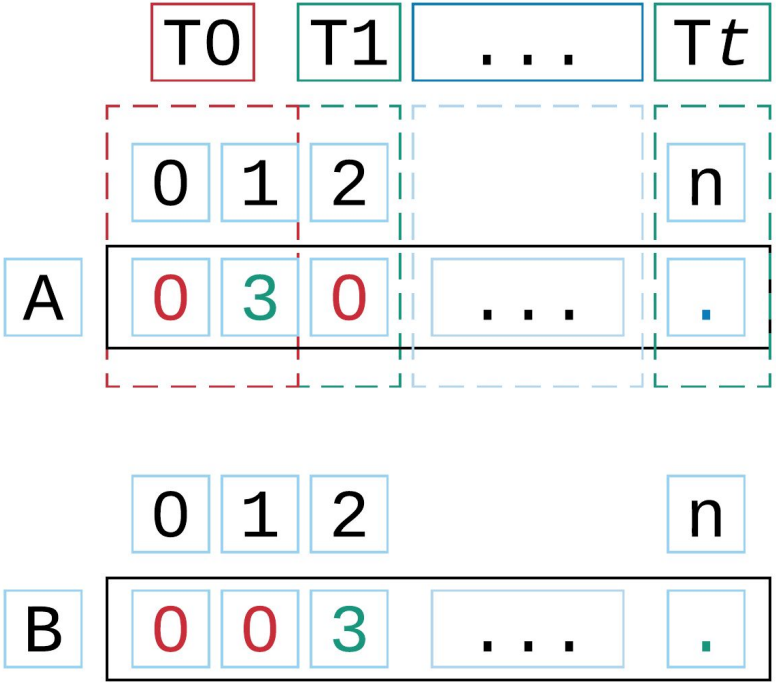
Counting-sort (Parallel) - Loop 4

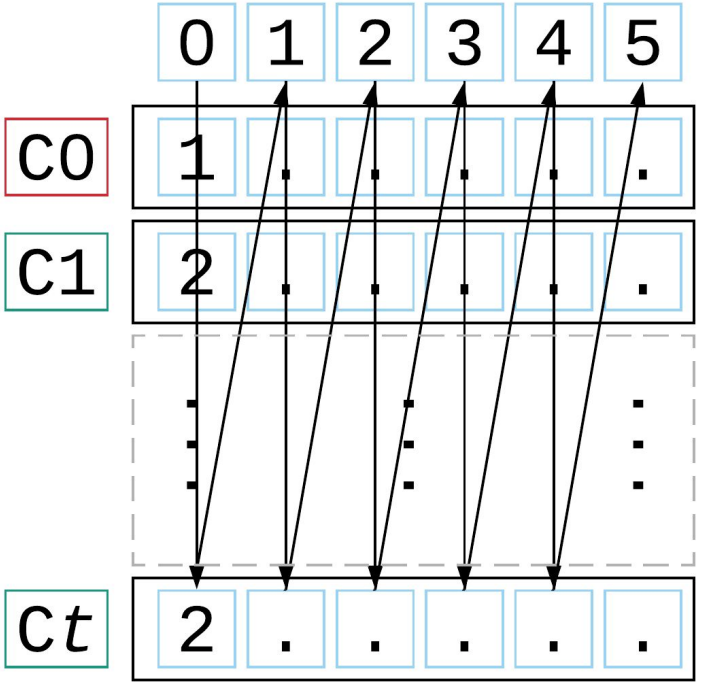


Counting-sort (Parallel) - 2 Threads



Counting-sort (Parallel) - (t) Threads





Breadth First Search (BFS)

Breadth First Search (BFS) : Traversing or searching tree or graph data structures.

- ***Input***: vertices[1..n], Source.
- ***Output***: parent[1..v].
- ***Auxiliary* storage**: frontier, next.

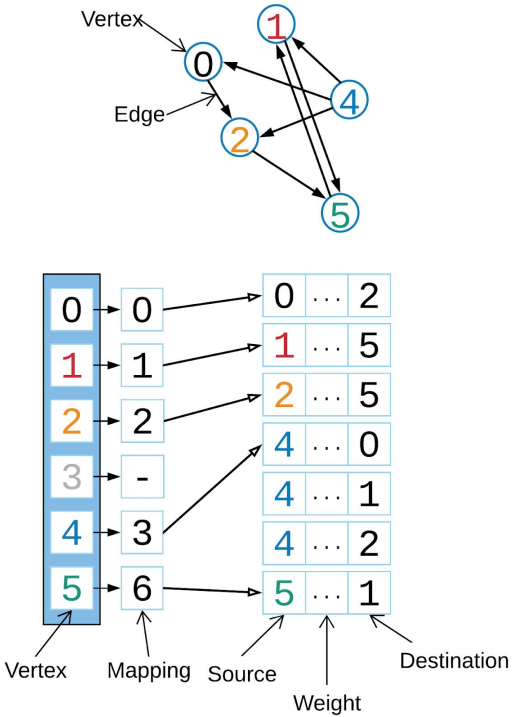
Breadth First Search (BFS)

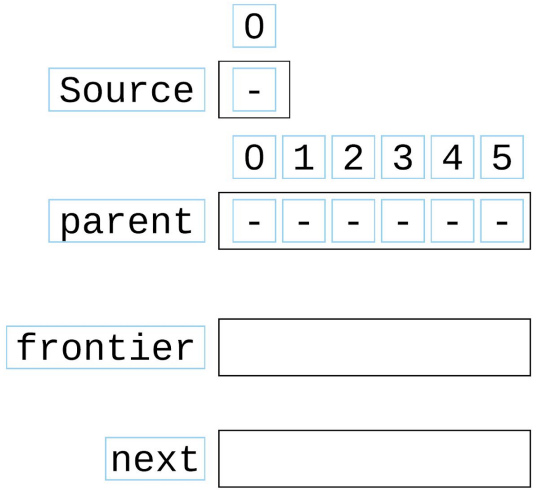
```
function breadth-first-search(vertices,source)
    frontier←{source}
    next←{}
    parents←[-1,-1,...-1]
    while frontier≠{} do
        top-down-step(vertices,frontier,next,parents)
        frontier←next
        next←{}
    end while
    return tree
```

Top Down Step (Push)

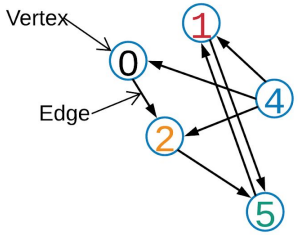
```
function top-down-step(vertices,frontier,next,parents)
   for v ∈ frontier do
     for n ∈ neighbors[v] do
       if parents[n] = -1 then
          parents[n]←v
          next←next⋃{n}
       end if
     end for
   end for
```

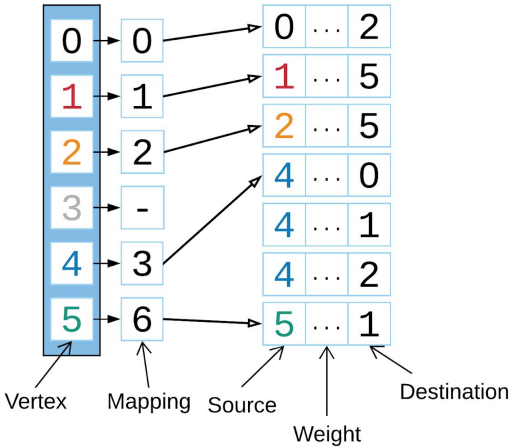
Breadth First Search (BFS)

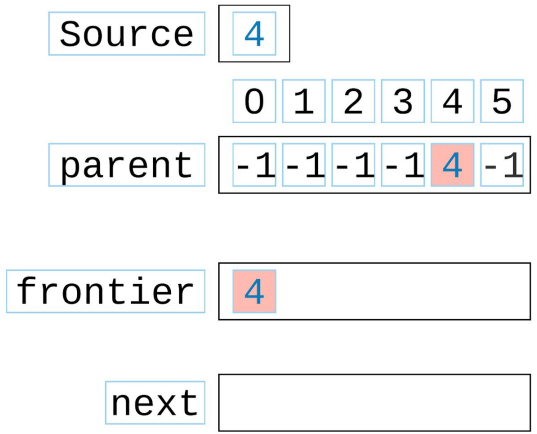




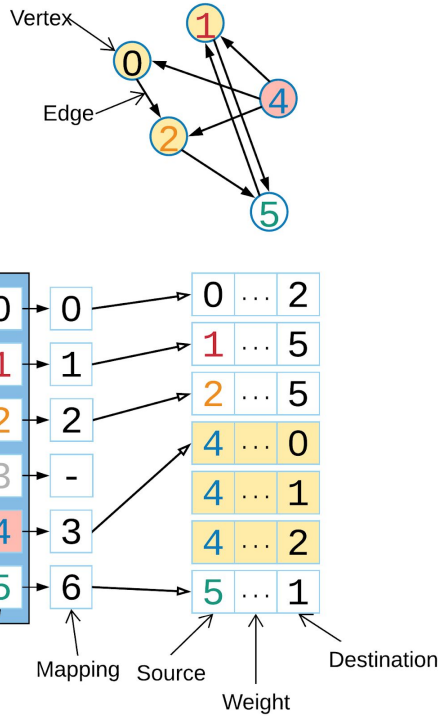
Breadth First Search (BFS) Initialize

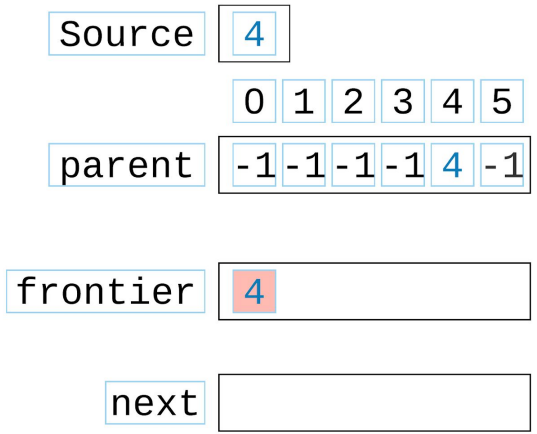




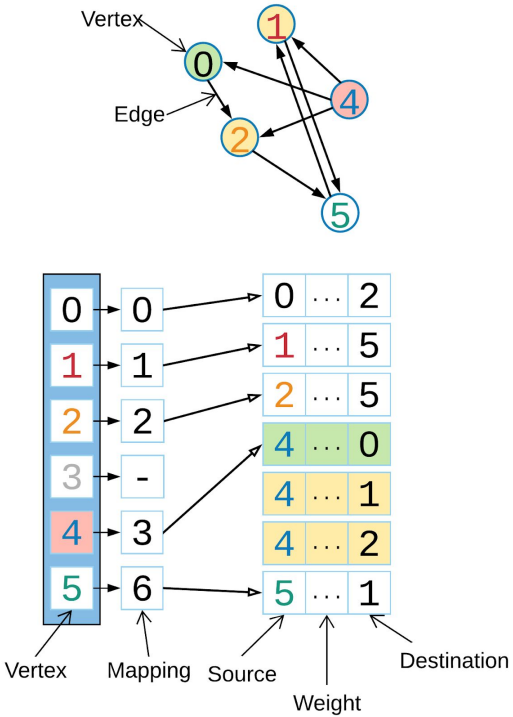


Breadth First Search (BFS) Iteration-1





Breadth First Search (BFS) Iteration-1



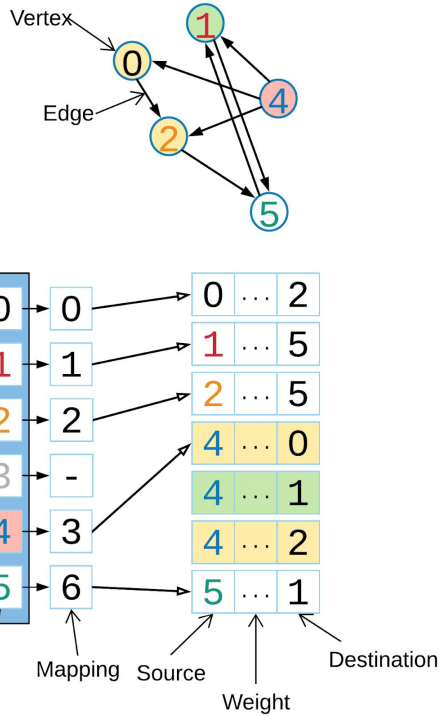
Source	4

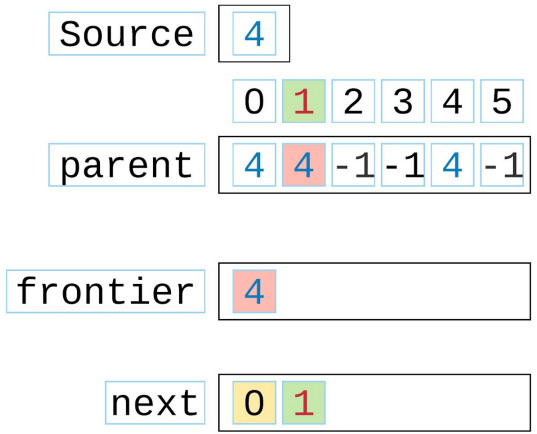
	0	1	2	3	4	5
parent	4	-1	-1	-1	4	-1

frontier	4

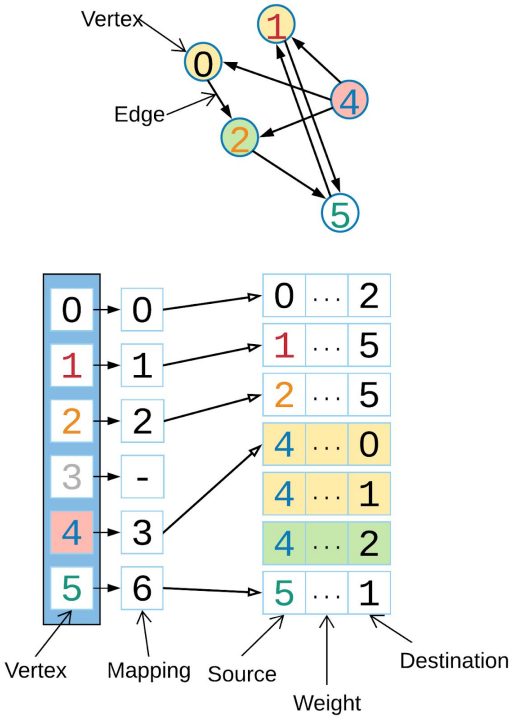
next	0

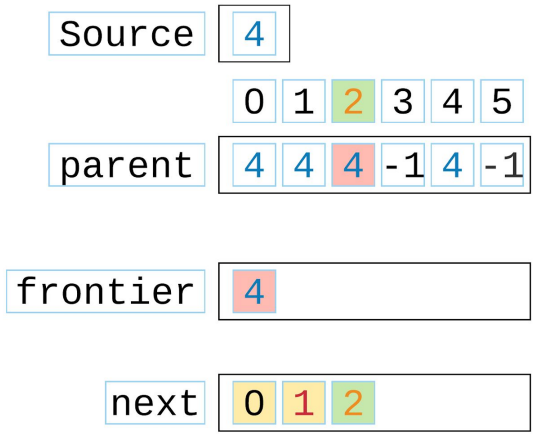
Breadth First Search (BFS) Iteration-1



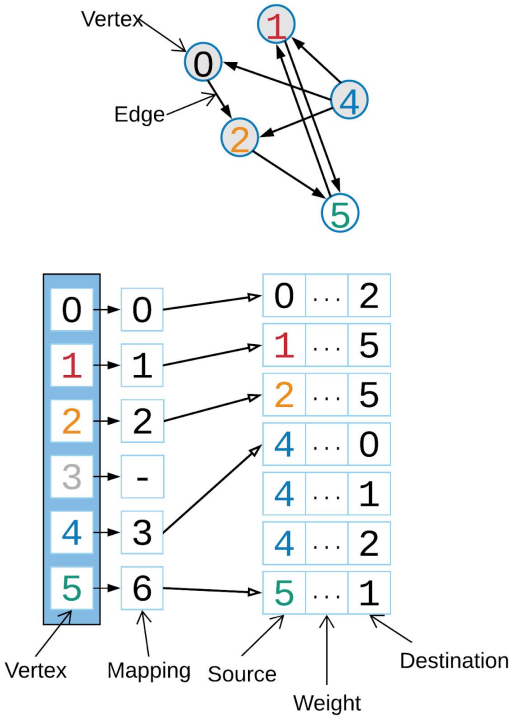


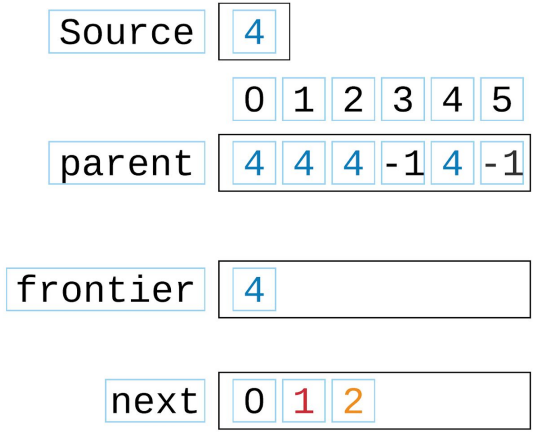
Breadth First Search (BFS) Iteration-1



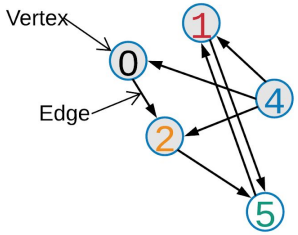


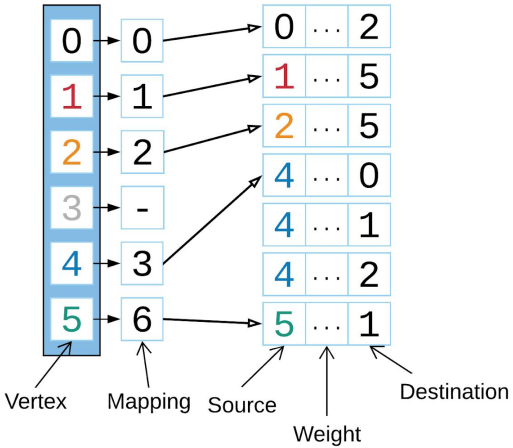
Breadth First Search (BFS) Iteration-1-end





Breadth First Search (BFS) Iteration-2





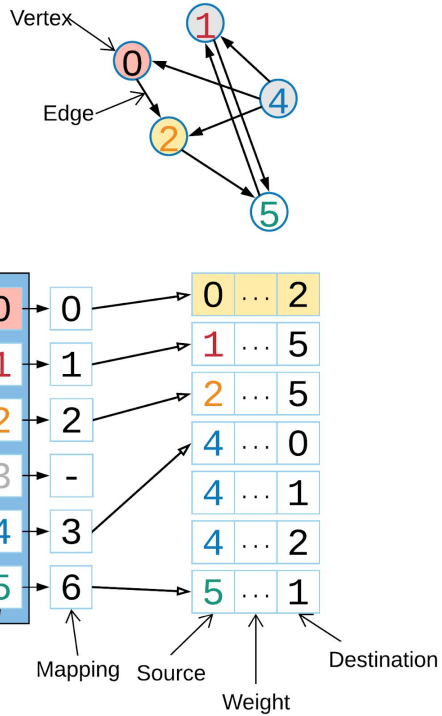
Source	4

	0	1	2	3	4	5
parent	4	4	4	-1	4	-1

frontier	0	1	2

next	

Breadth First Search (BFS) Iteration-2



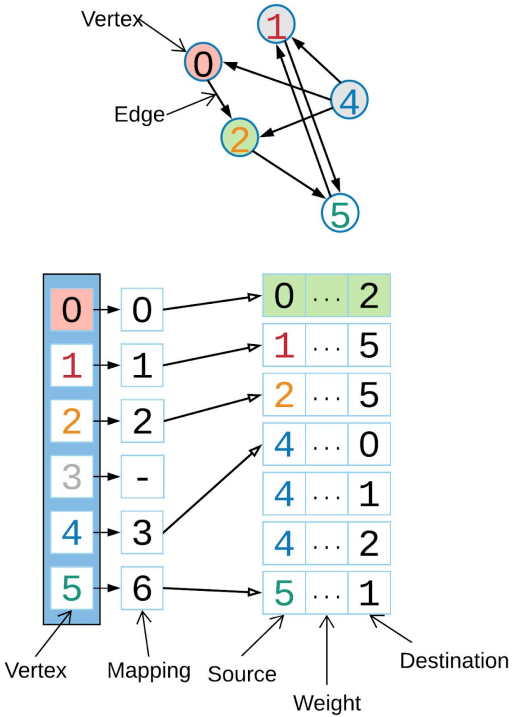
Source	4

	0	1	2	3	4	5
parent	4	4	4	-1	4	-1

frontier	0	1	2

next	

Breadth First Search (BFS) Iteration-2



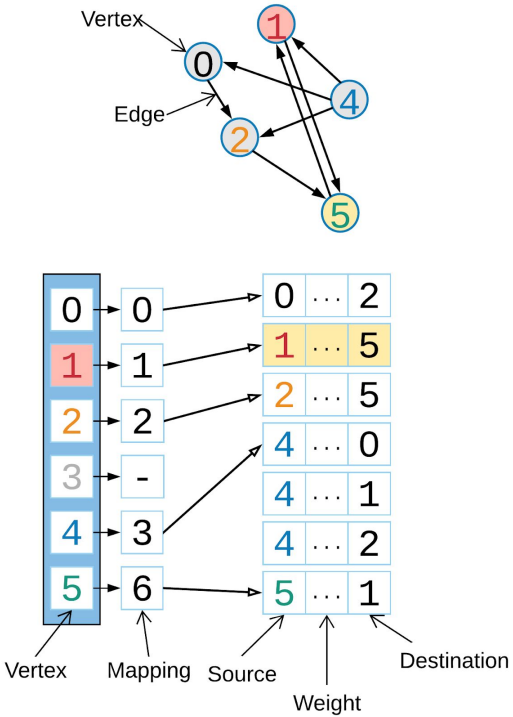
Source	4

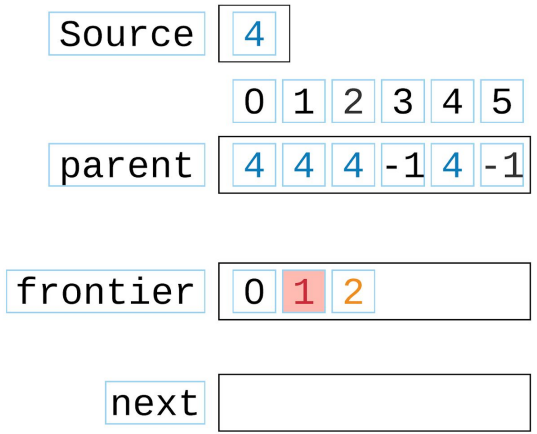
	0	1	2	3	4	5
parent	4	4	4	-1	4	-1

frontier	0	1	2

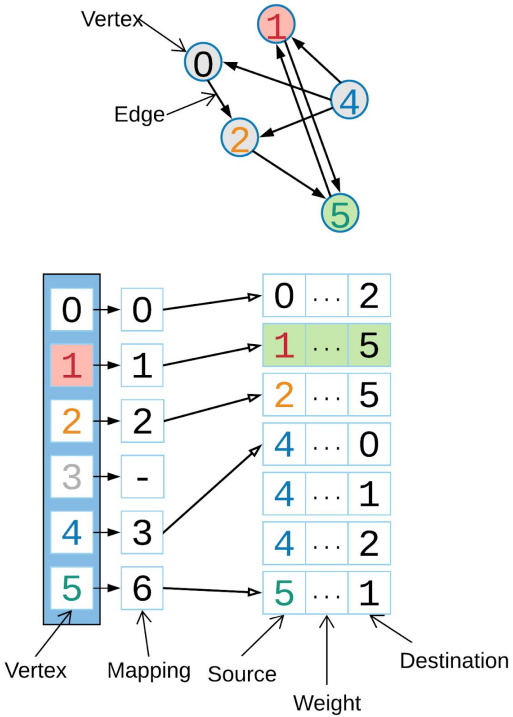
next	

Breadth First Search (BFS) Iteration-2





Breadth First Search (BFS) Iteration-2



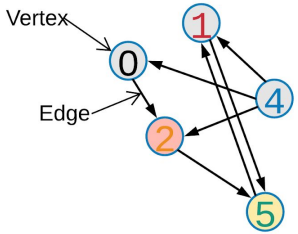
Source 4

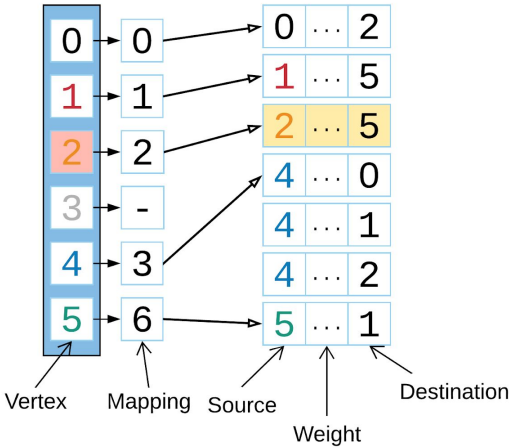
	0	1	2	3	4	5
parent	4	4	4	-1	4	1

frontier 0 1 2

next 5

Breadth First Search (BFS) Iteration-2





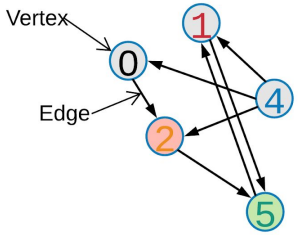
Source	4

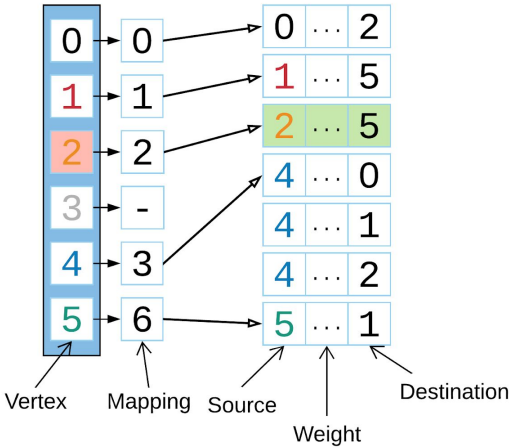
	0	1	2	3	4	5
parent	4	4	4	-1	4	1

frontier	0	1	2

next	5

Breadth First Search (BFS) Iteration-2





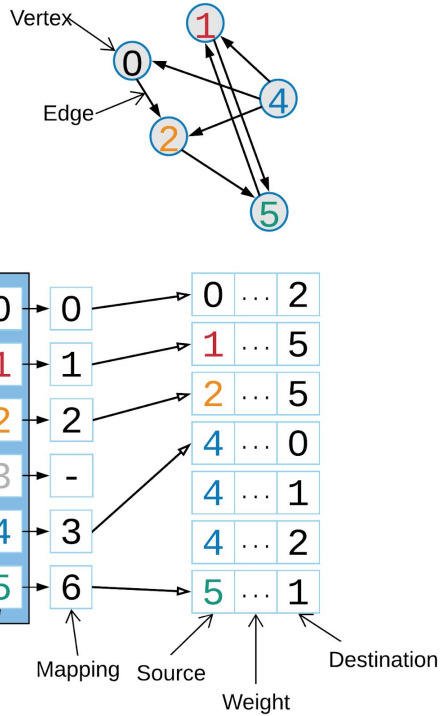
Source	4

	0	1	2	3	4	5
parent	4	4	4	-1	4	1

frontier	0	1	2

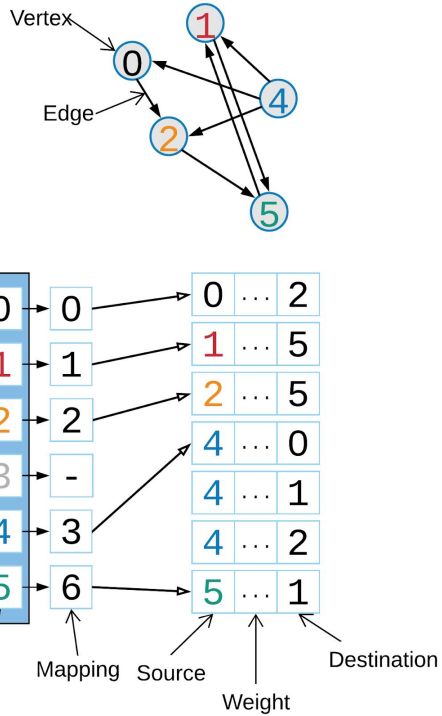
next	5

Breadth First Search (BFS) Iteration-2-end



Source	4					
	0	1	2	3	4	5
parent	4	4	4	-1	4	1
frontier	0	1	2			
next	5					

Breadth First Search (BFS) Iteration-3



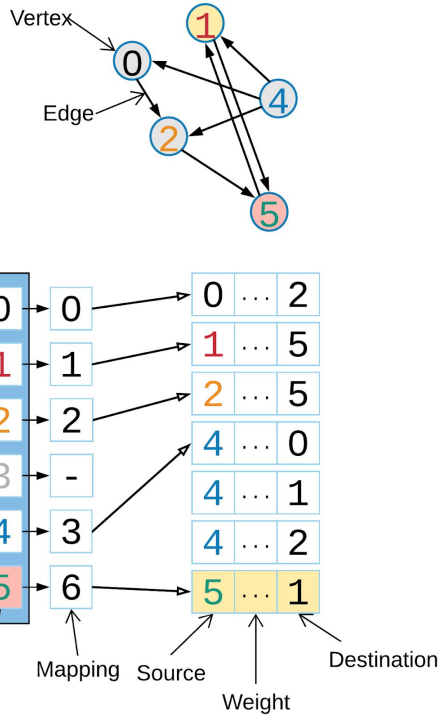
Source 4

	0	1	2	3	4	5
parent	4	4	4	-1	4	1

frontier 5

next

Breadth First Search (BFS) Iteration-3



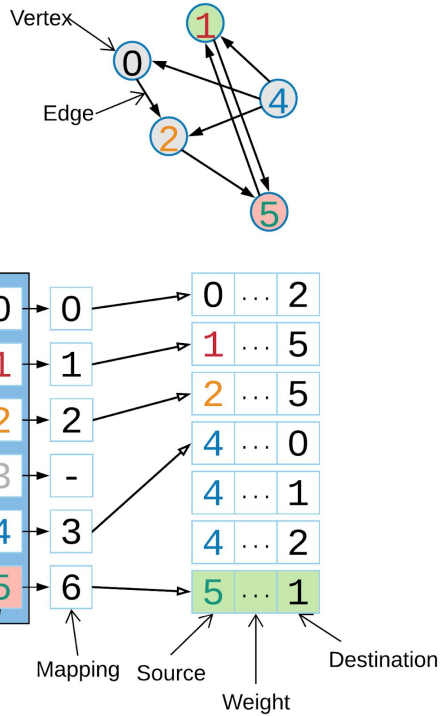
Source	4

	0	1	2	3	4	5
parent	4	4	4	-1	4	1

frontier	5

next	

Breadth First Search (BFS) Iteration-3



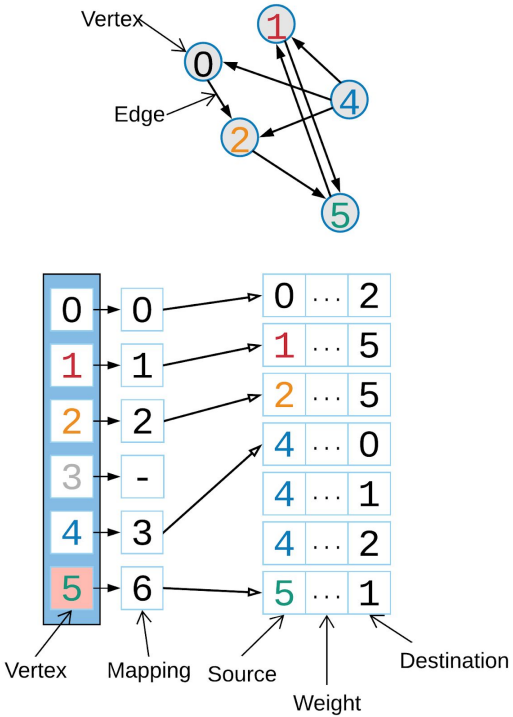
Source 4

	0	1	2	3	4	5
parent	4	4	4	-1	4	1

frontier 5

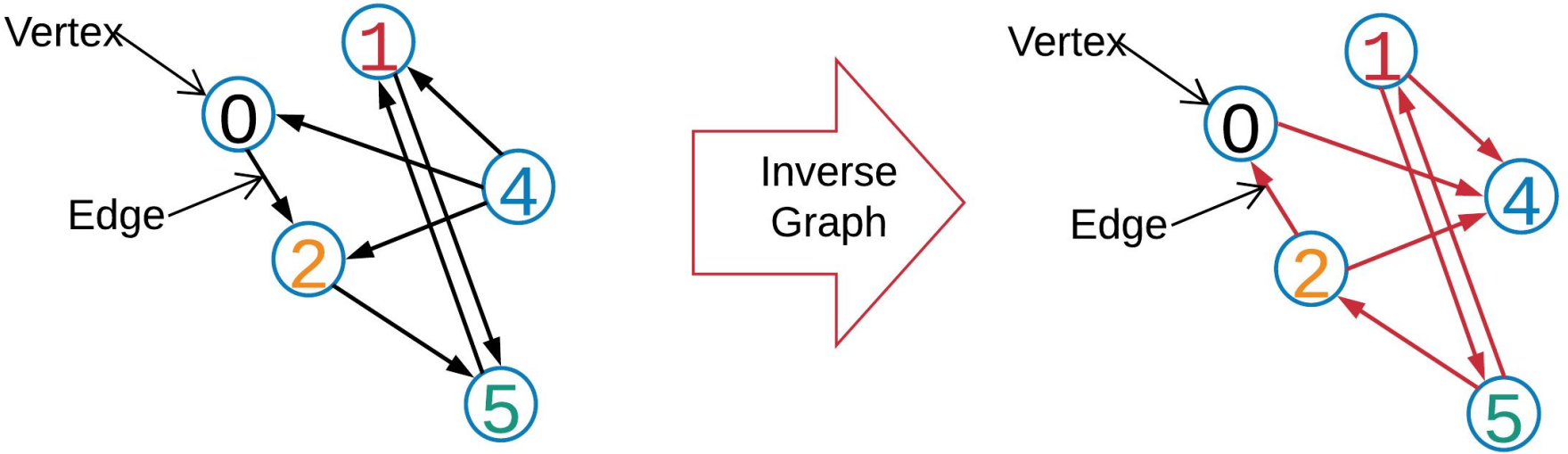
next

Breadth First Search (BFS) Iteration-3

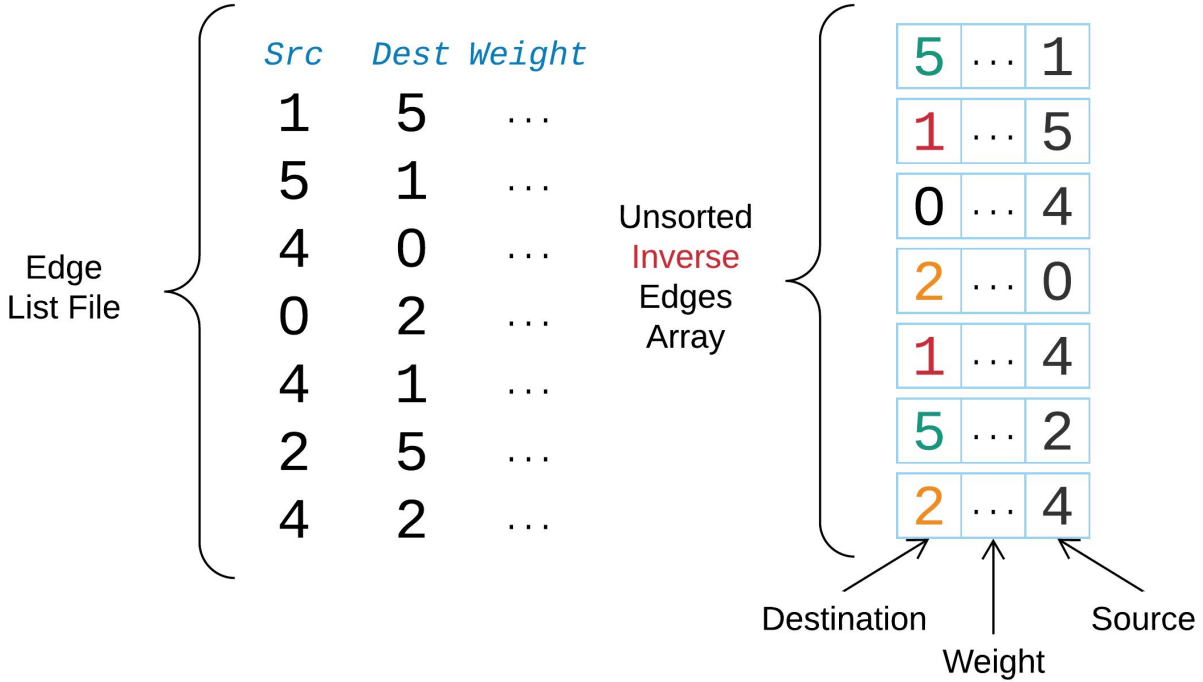


Source	4					
	0	1	2	3	4	5
parent	4	4	4	-1	4	1
frontier						
next						

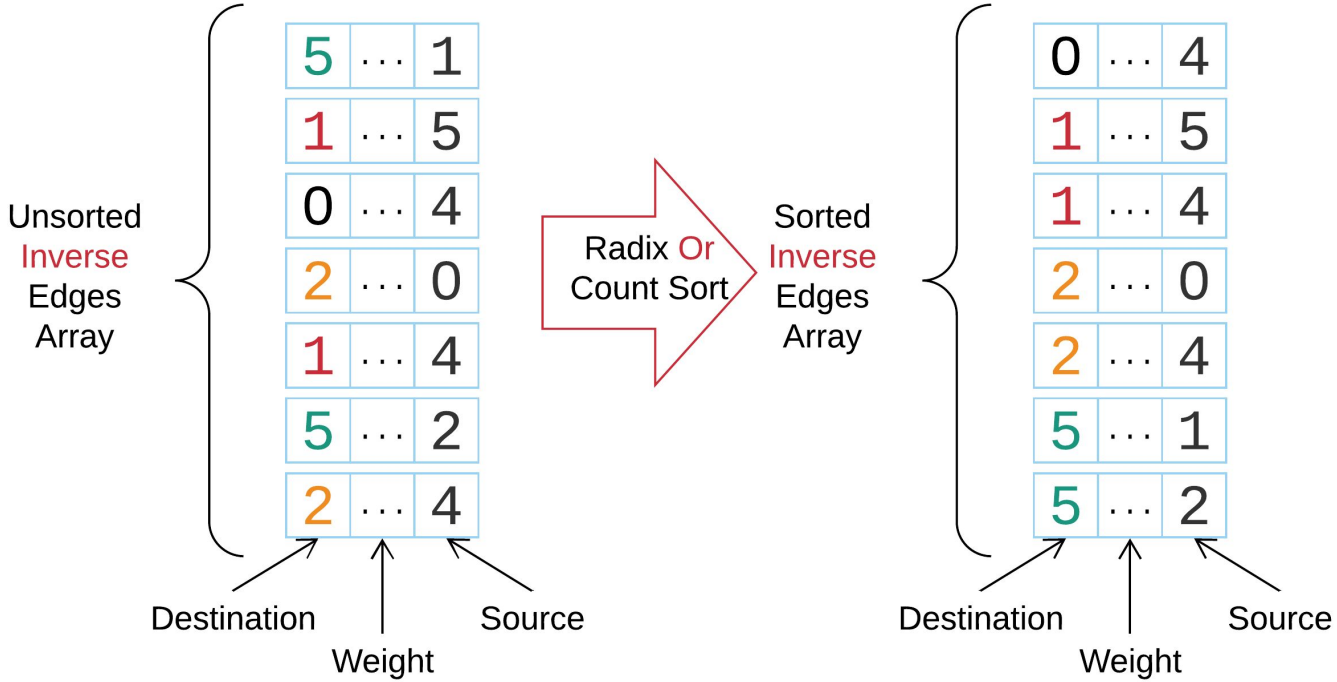
Inverse Graph



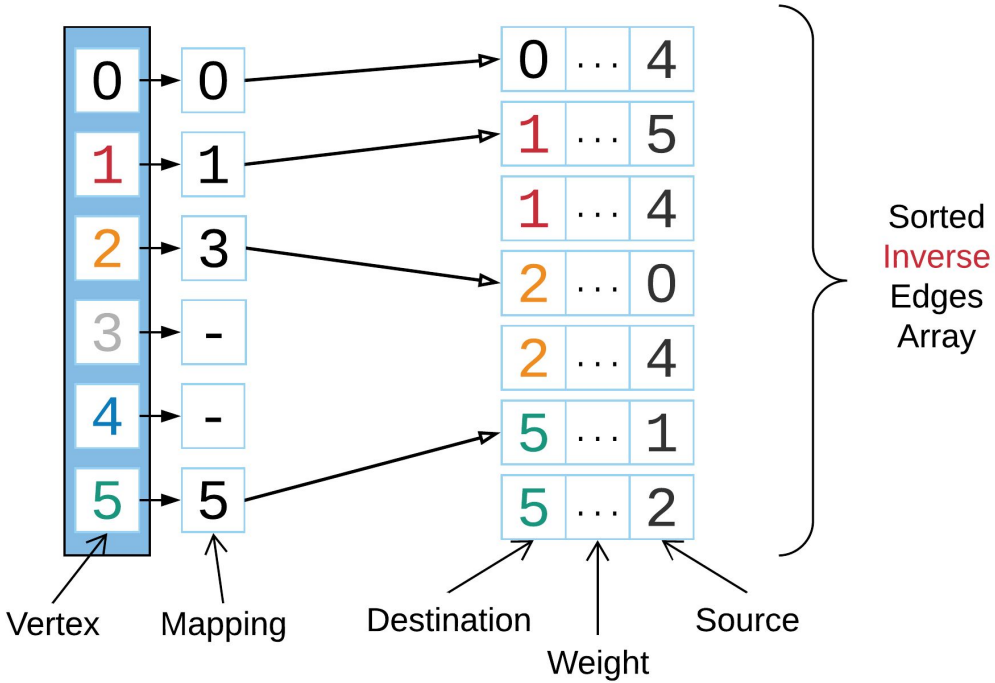
Inverse Graph (load)



Inverse Graph (sort)



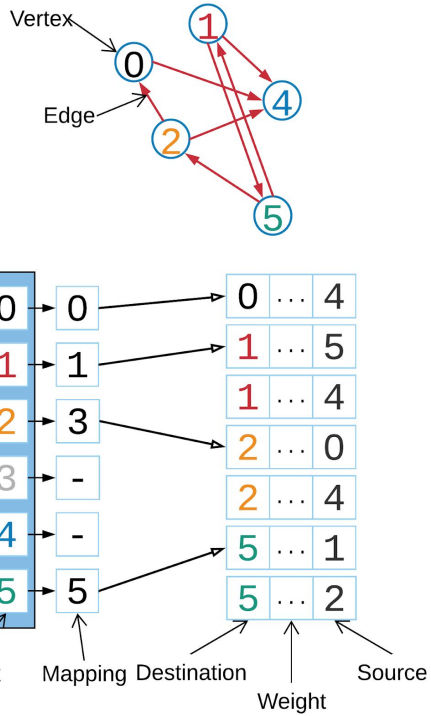
Inverse Graph (map)

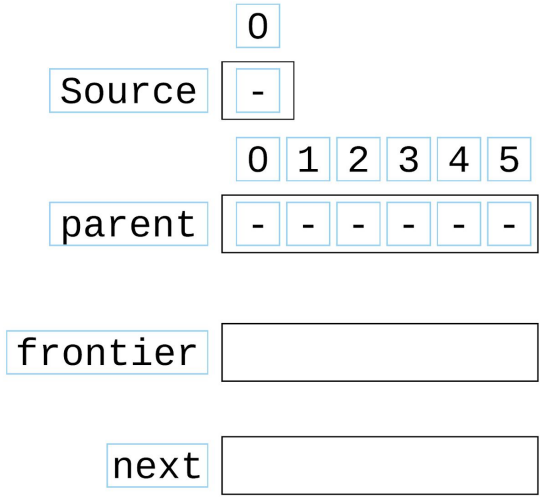


Bottom Up Step (Pull)

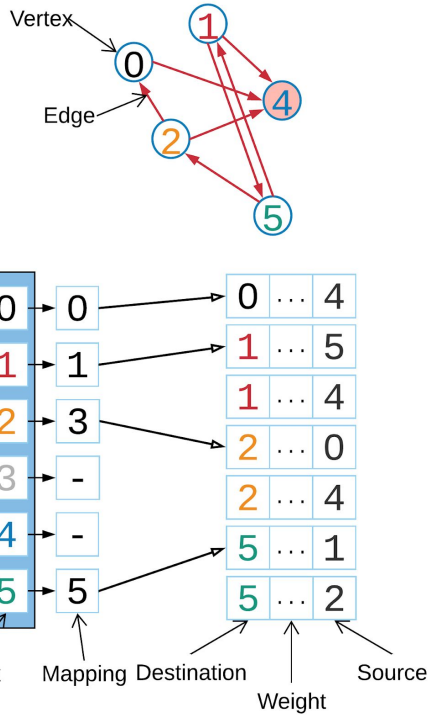
```
function bottom-up-step(vertices,frontier,next,parents)
    for v ∈ vertices do
      if parents[v] = -1 then
        for n ∈ neighbors[v] do
          if n ∈ frontier do
             parents[v]←n
             next←next⋃{v}
             break
          end if
        end for
      end if
    end for
```

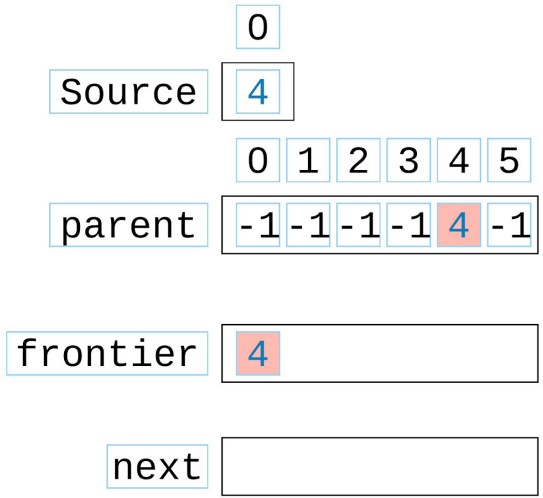
Bottom Up Step (Pull)



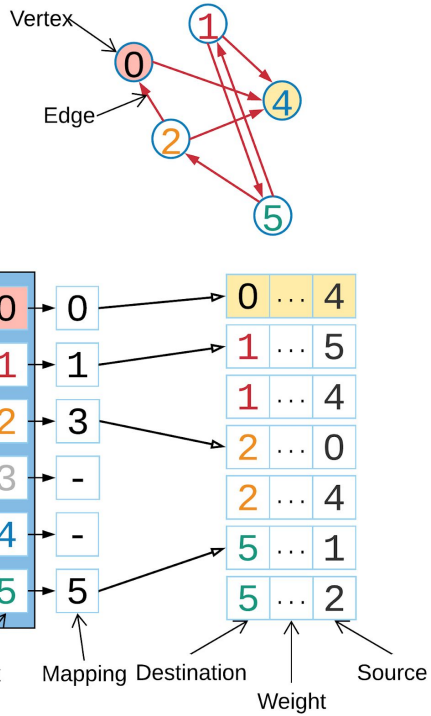


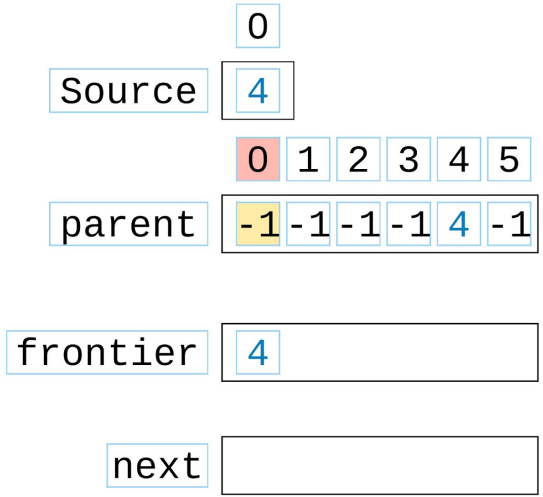
Breadth First Search (BFS) Initialize



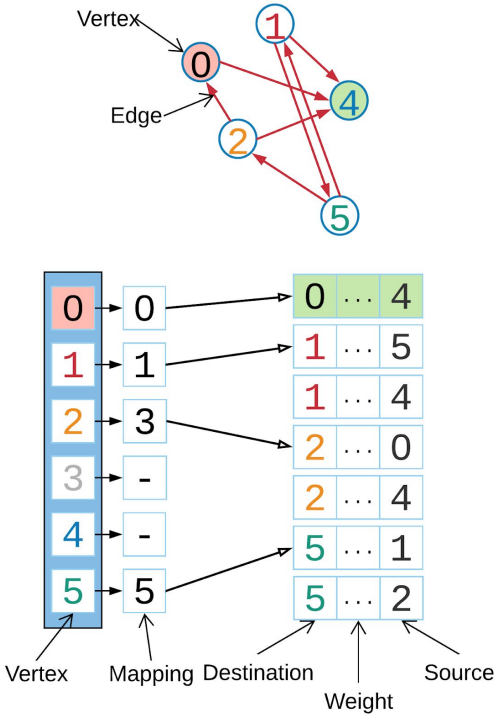


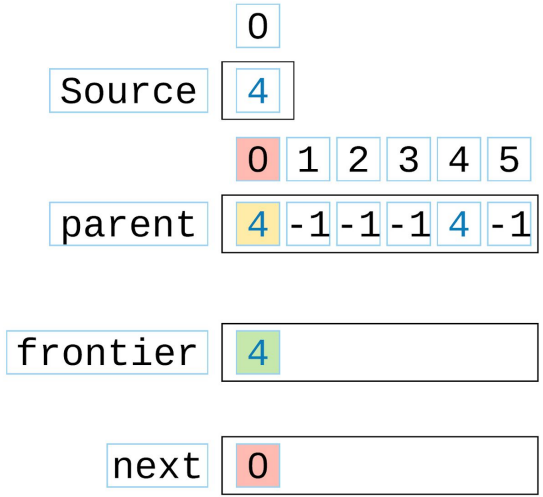
Breadth First Search (BFS) Iteration-1



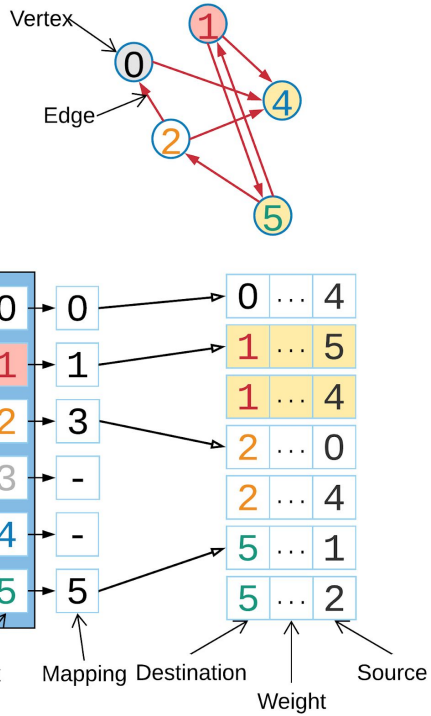


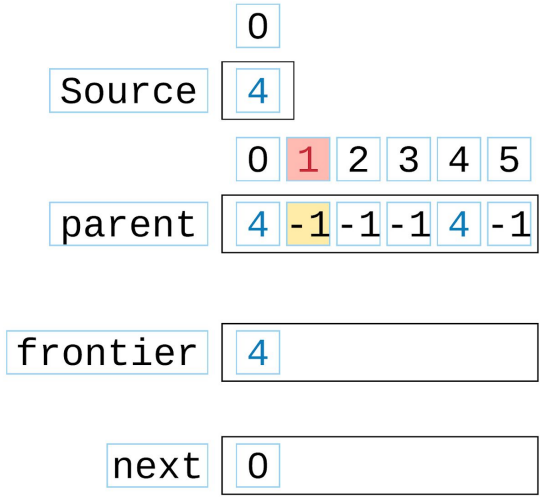
Breadth First Search (BFS) Iteration-1



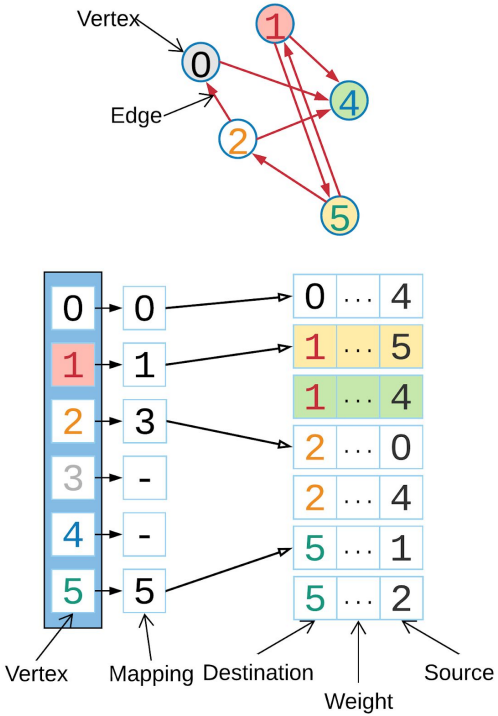


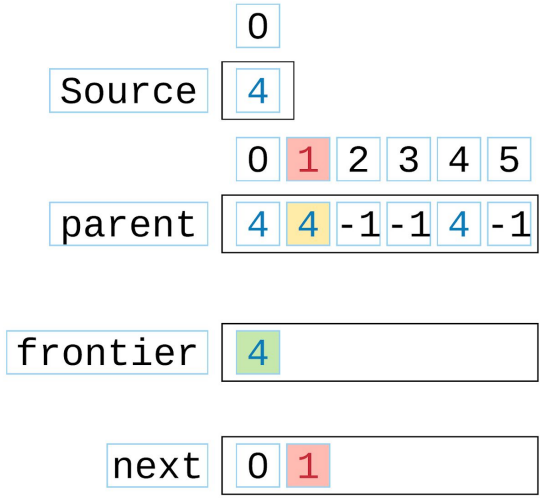
Breadth First Search (BFS) Iteration-1



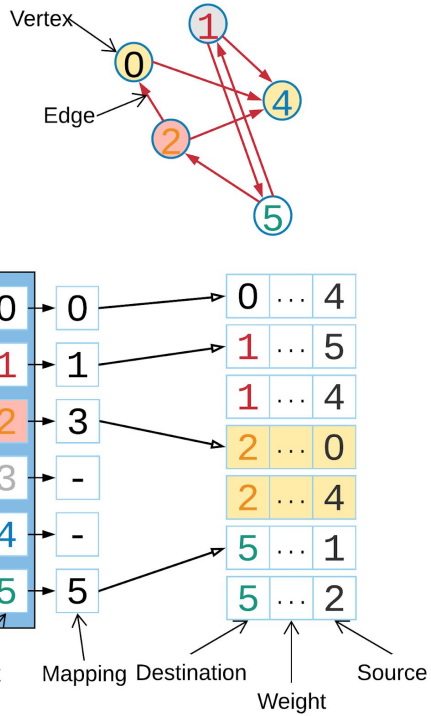


Breadth First Search (BFS) Iteration-1



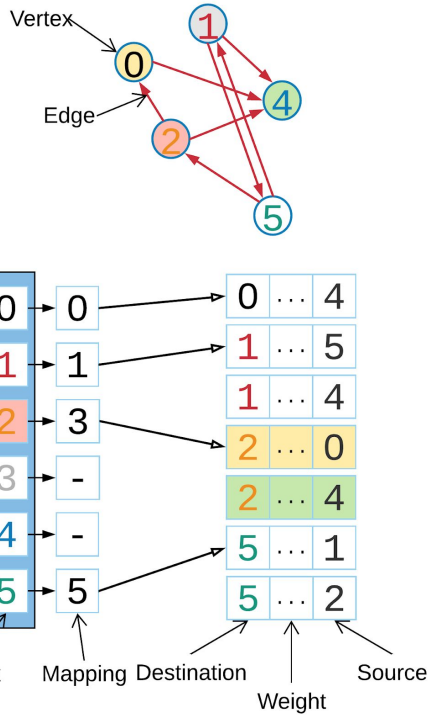


Breadth First Search (BFS) Iteration-1



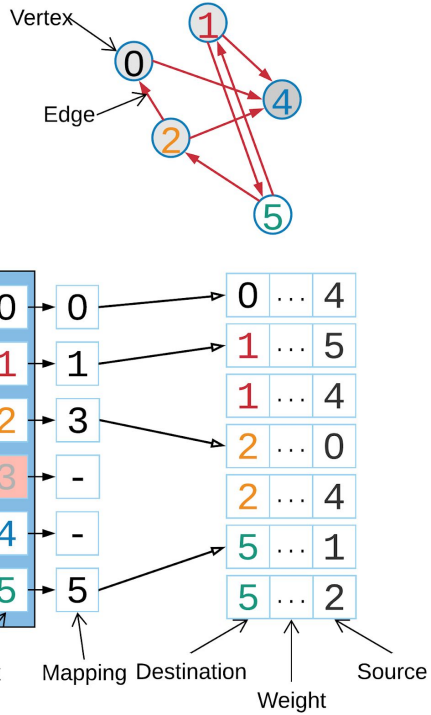
	0	1	2	3	4	5
Source	4					
parent	4	4	-1	-1	4	-1
frontier	4					
next	0	1				

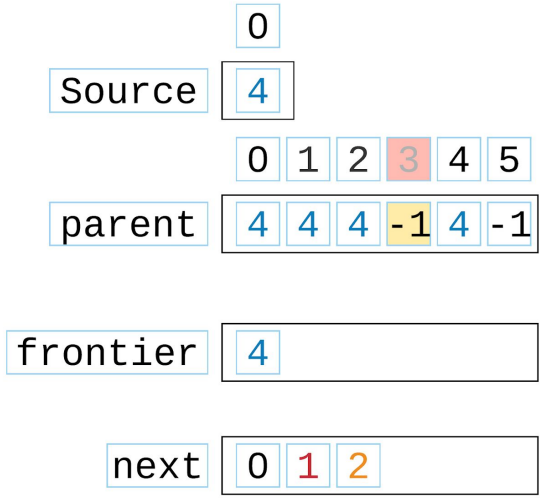
Breadth First Search (BFS) Iteration-1



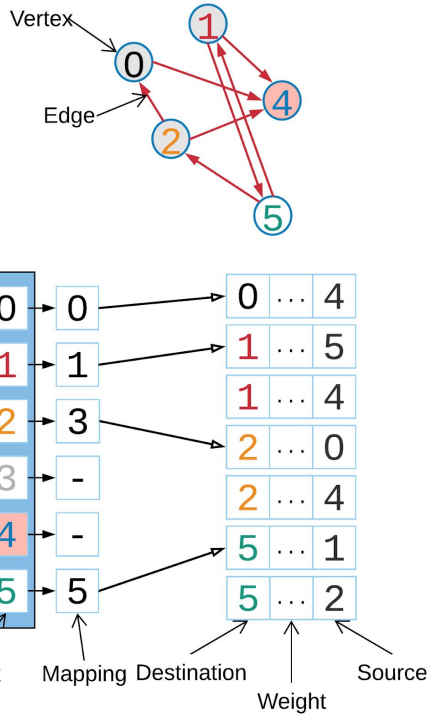
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	-1
frontier	4					
next	0	1	2			

Breadth First Search (BFS) Iteration-1



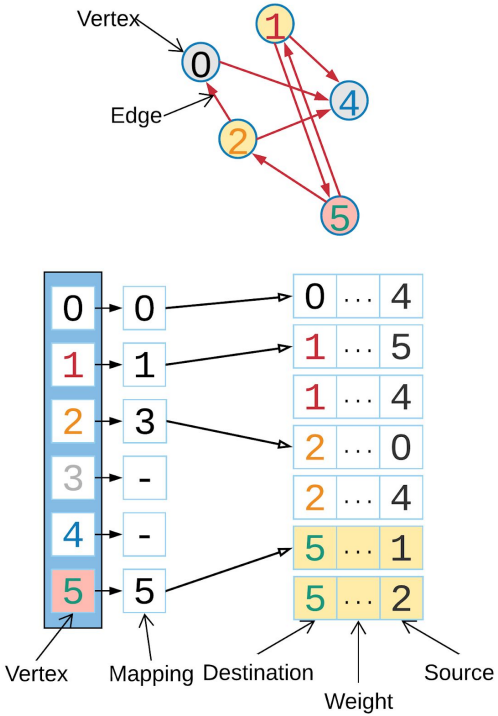


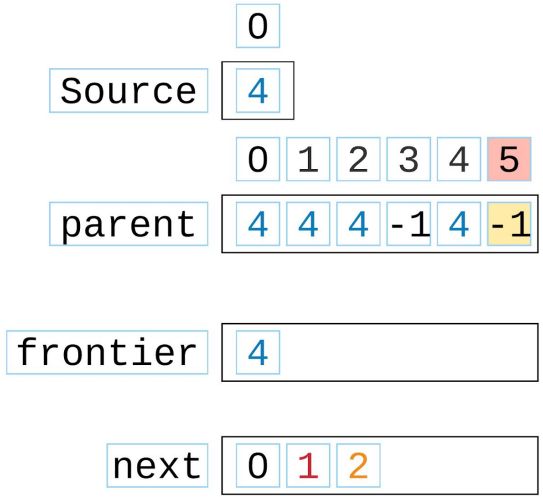
Breadth First Search (BFS) Iteration-1



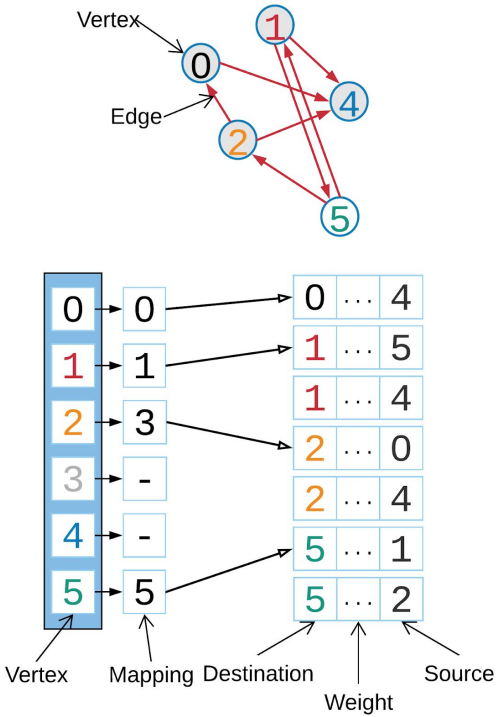
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	-1
frontier	4					
next	0	1	2			

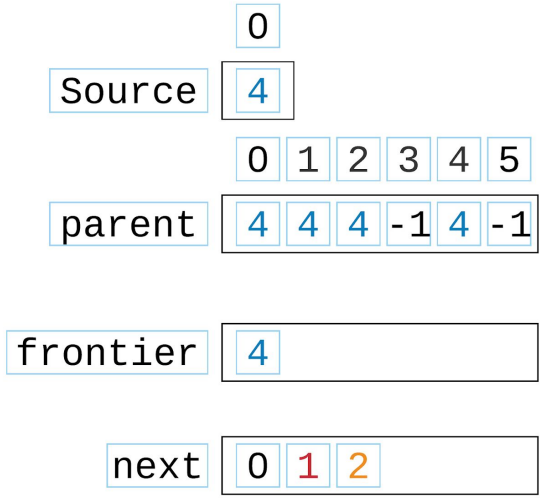
Breadth First Search (BFS) Iteration-1



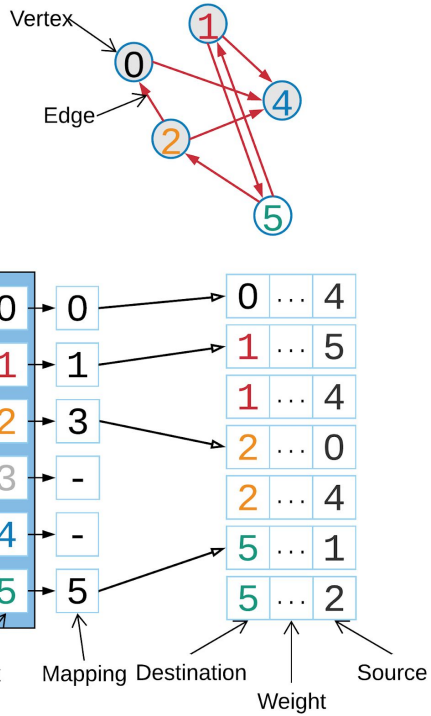


Breadth First Search (BFS) Iteration-1-end



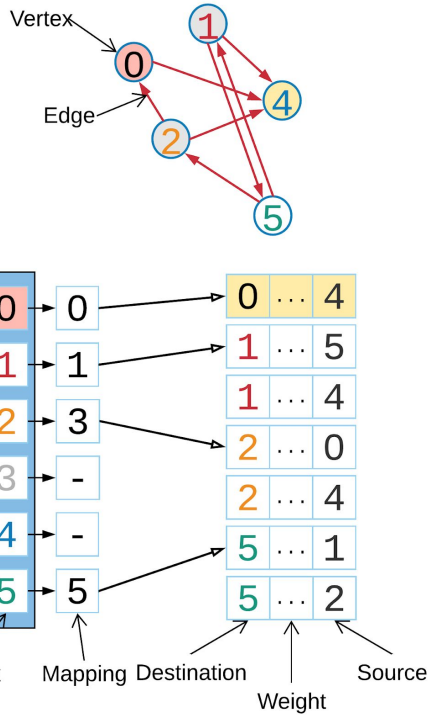


Breadth First Search (BFS) Iteration-2



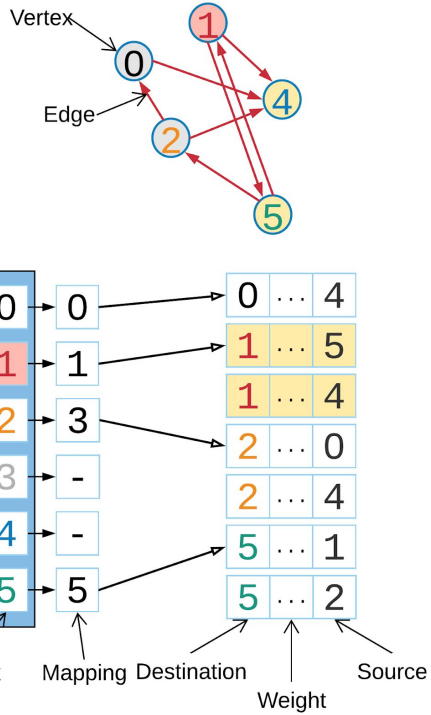
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	-1
frontier	0	1	2			
next						

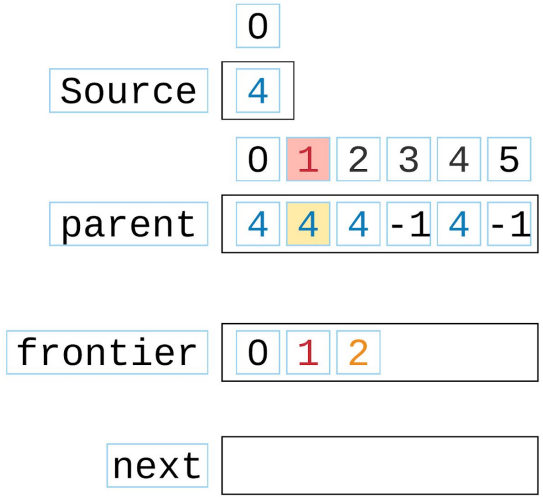
Breadth First Search (BFS) Iteration-2



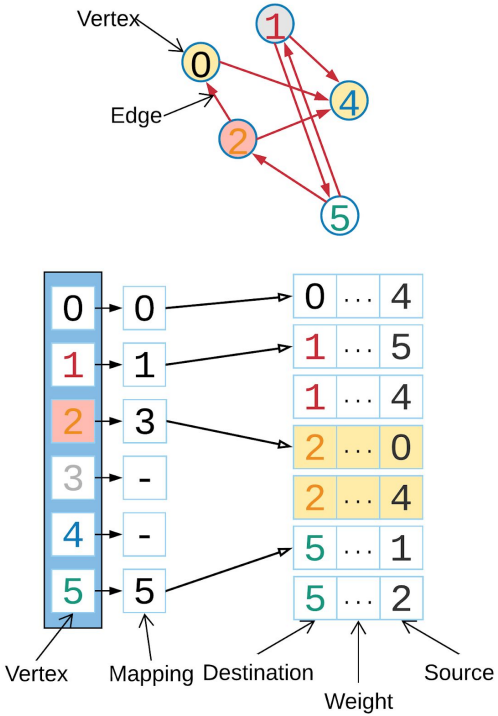
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	-1
frontier	0	1	2			
next						

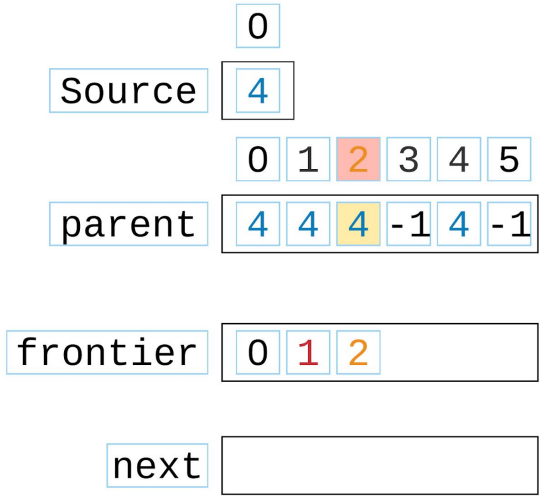
Breadth First Search (BFS) Iteration-2



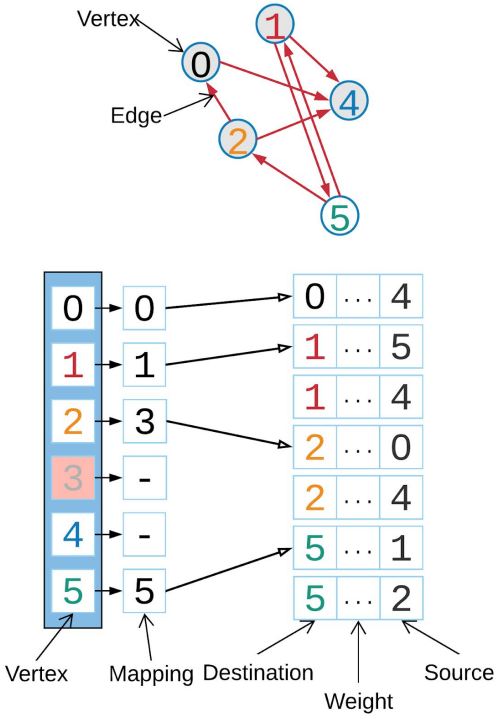


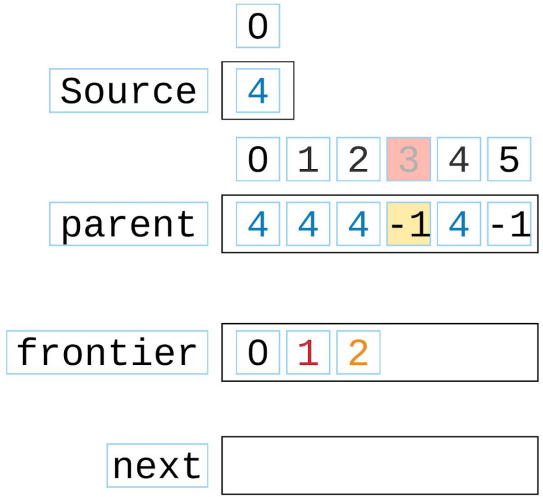
Breadth First Search (BFS) Iteration-2



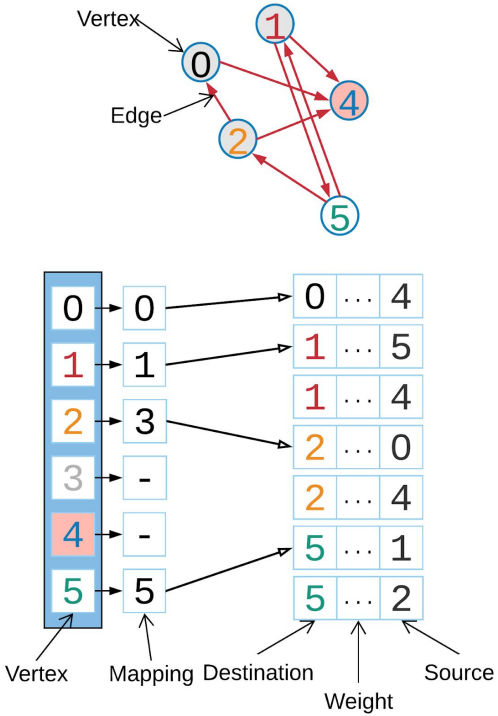


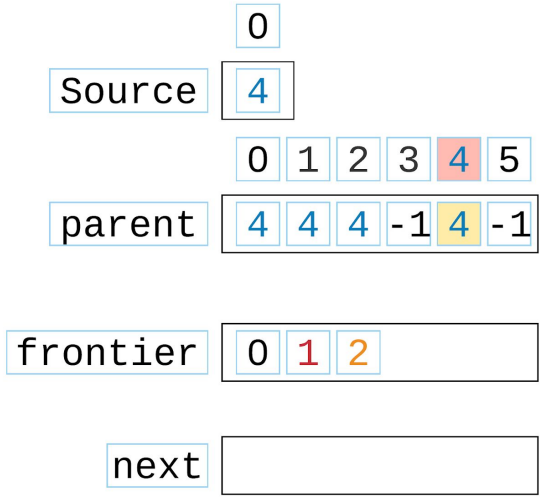
Breadth First Search (BFS) Iteration-2



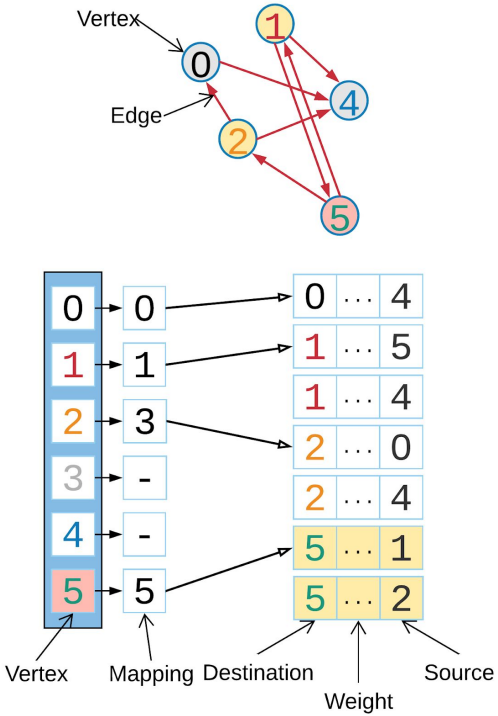


Breadth First Search (BFS) Iteration-2



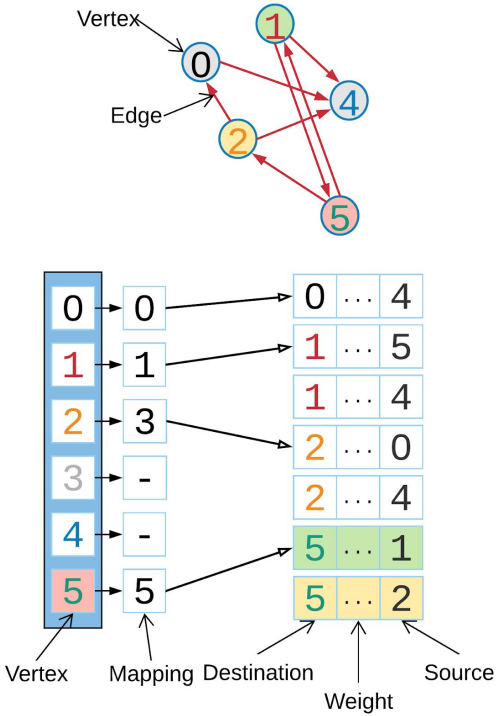


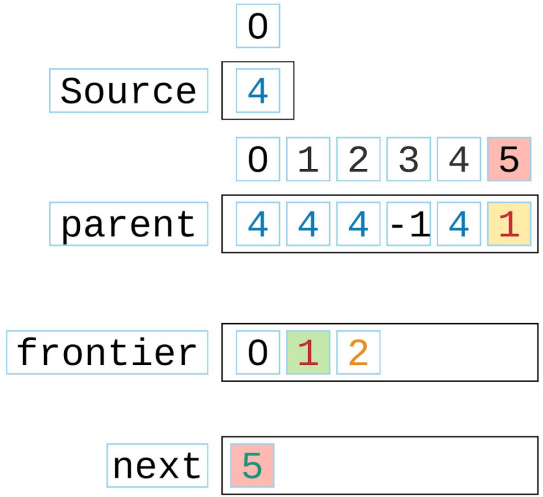
Breadth First Search (BFS) Iteration-2



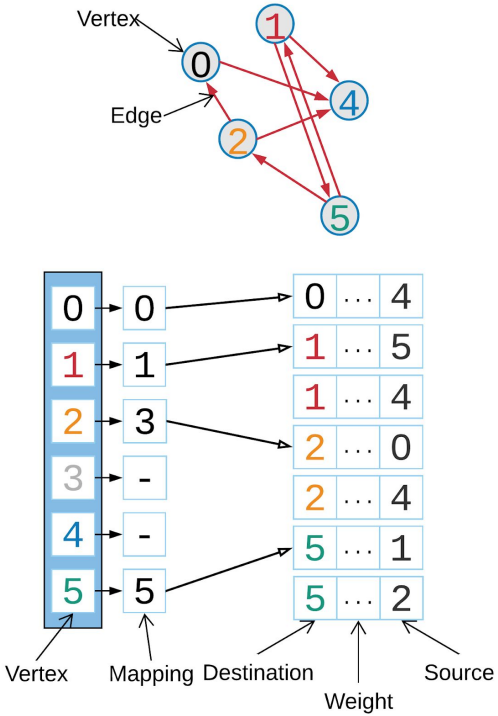
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	-1
frontier	0	1	2			
next						

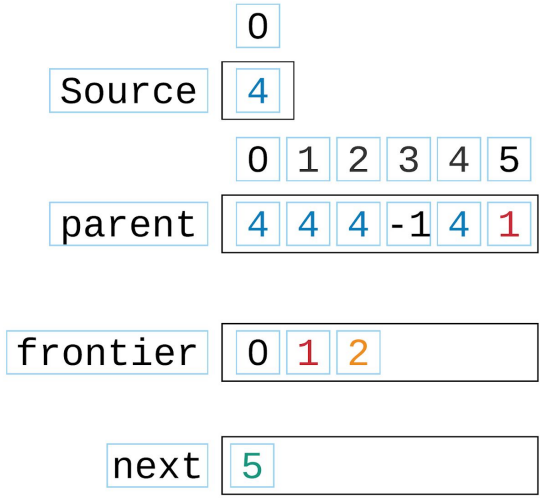
Breadth First Search (BFS) Iteration-2



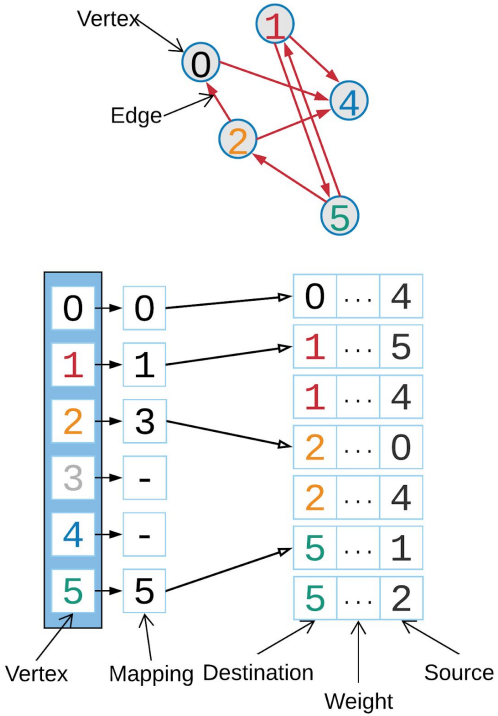


Breadth First Search (BFS) Iteration-2-end



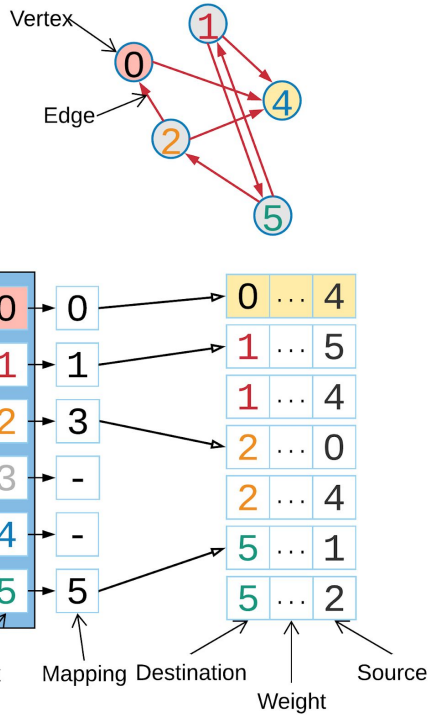


Breadth First Search (BFS) Iteration-3



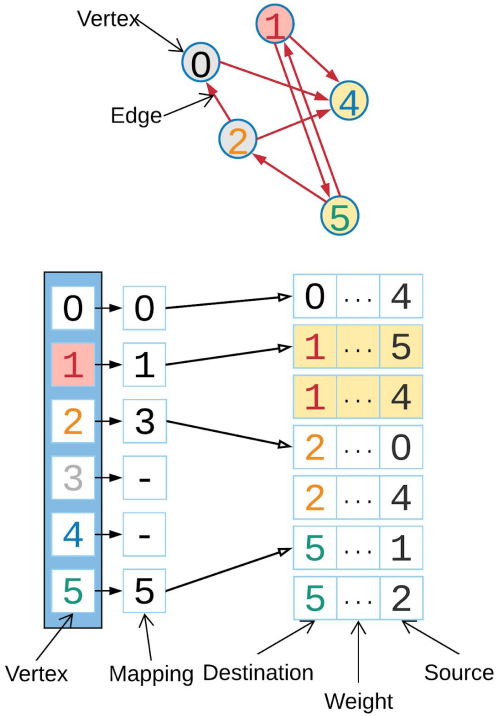
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	1
frontier	5					
next						

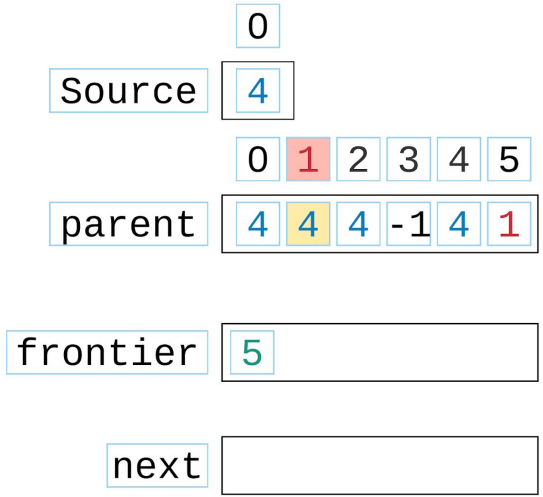
Breadth First Search (BFS) Iteration-3



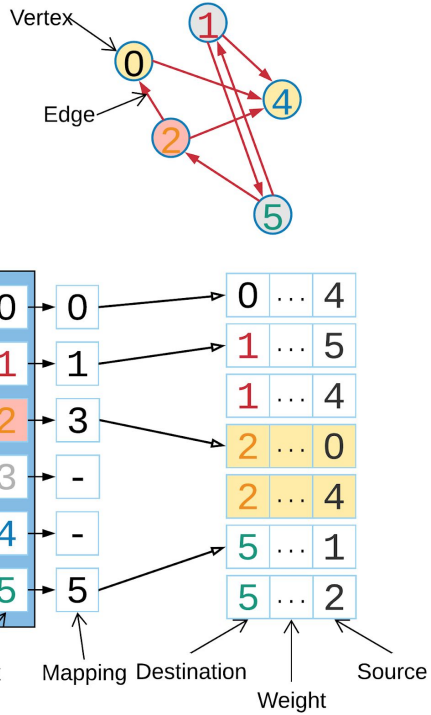
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	1
frontier	5					
next						

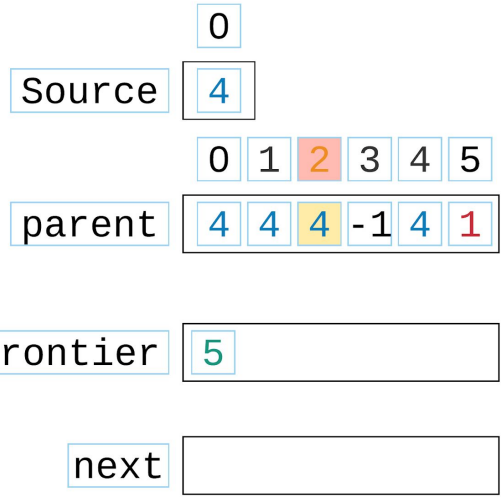
Breadth First Search (BFS) Iteration-3



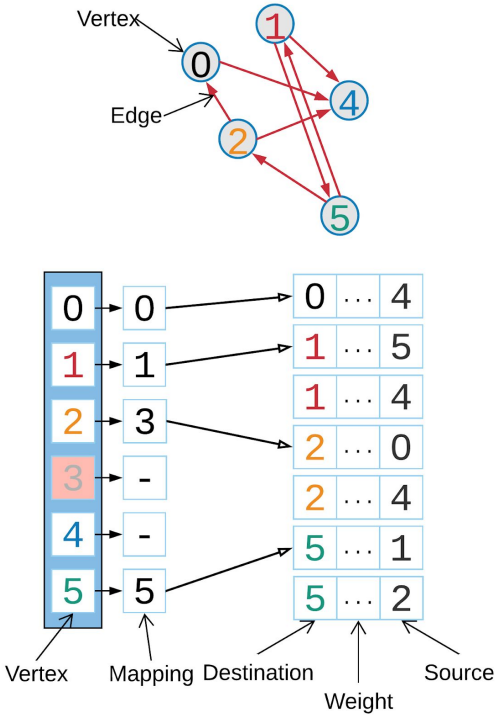


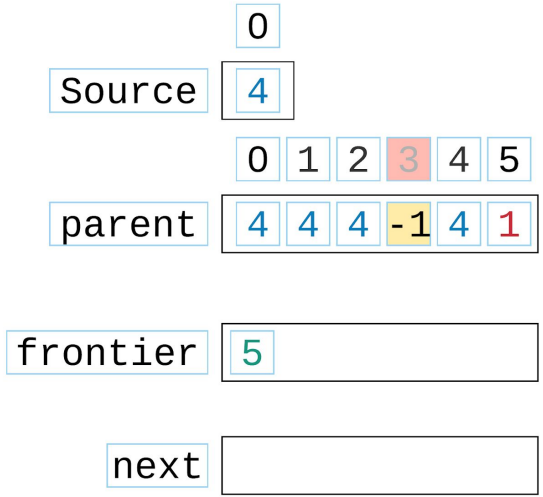
Breadth First Search (BFS) Iteration-3



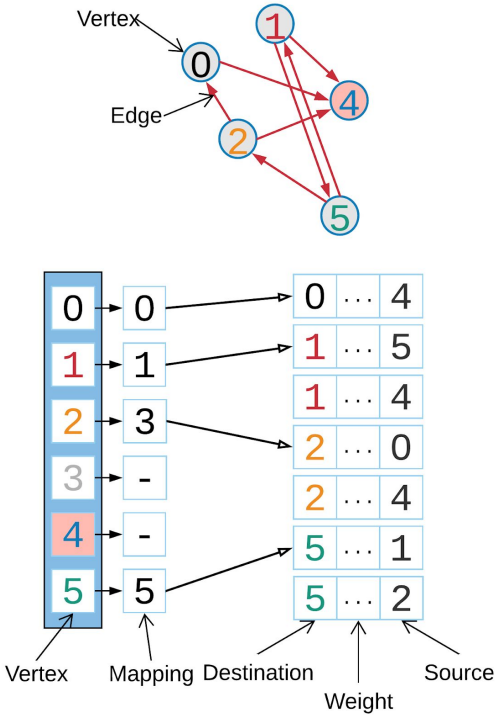


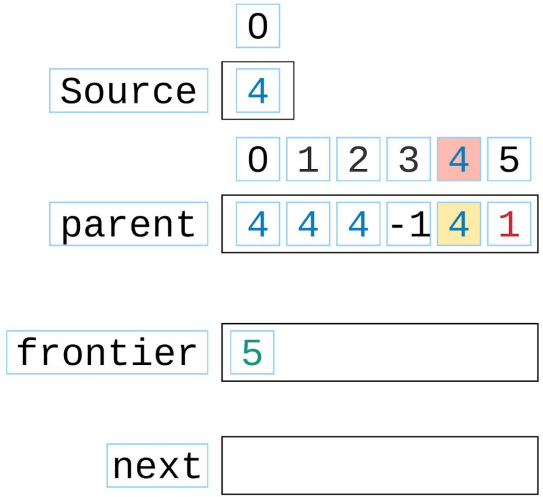
Breadth First Search (BFS) Iteration-3



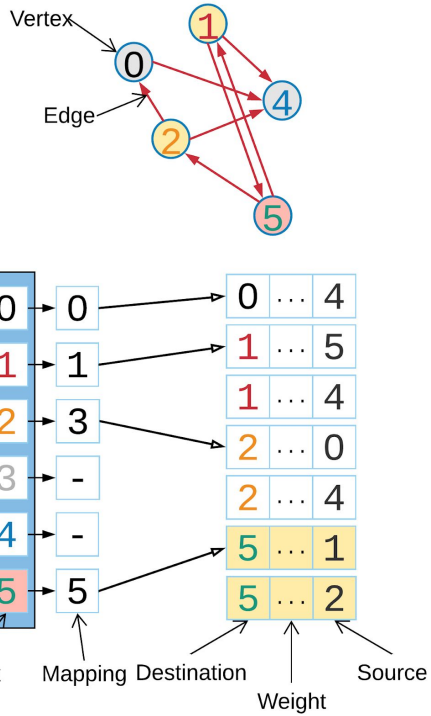


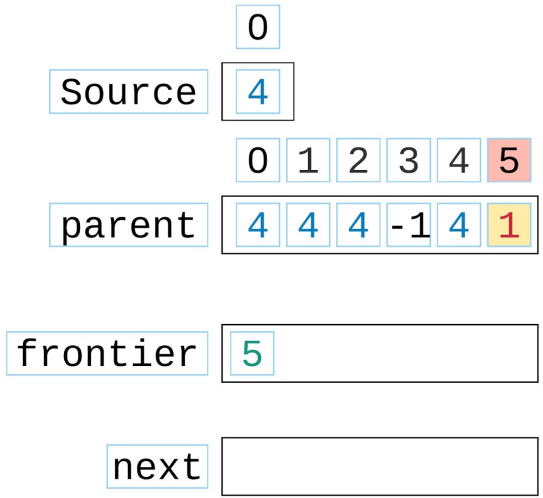
Breadth First Search (BFS) Iteration-3



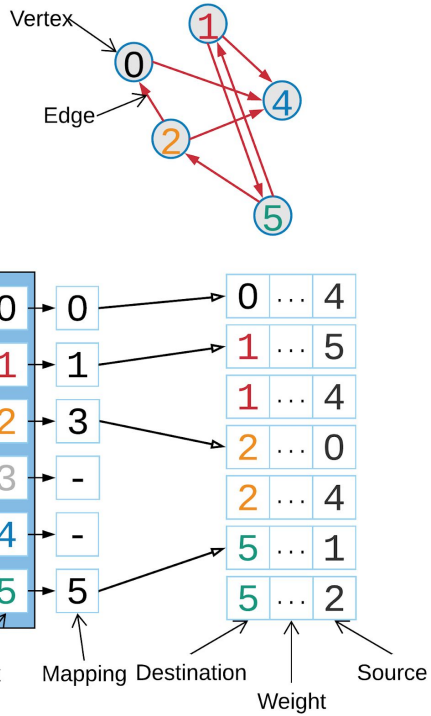


Breadth First Search (BFS) Iteration-3



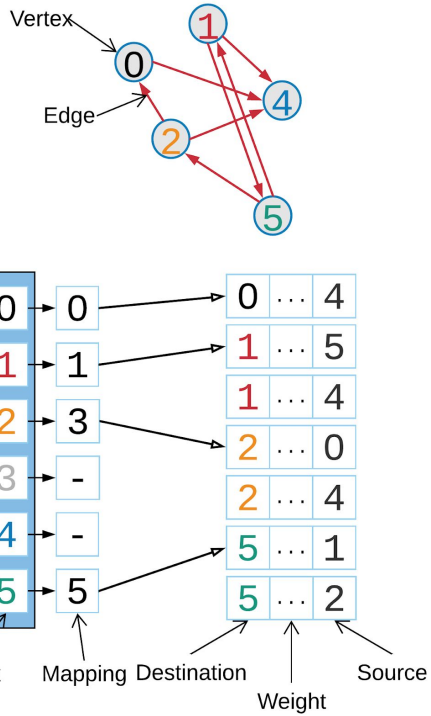


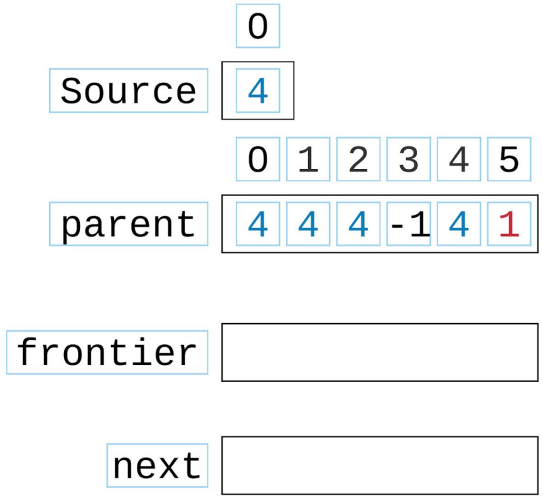
Breadth First Search (BFS) Iteration-end



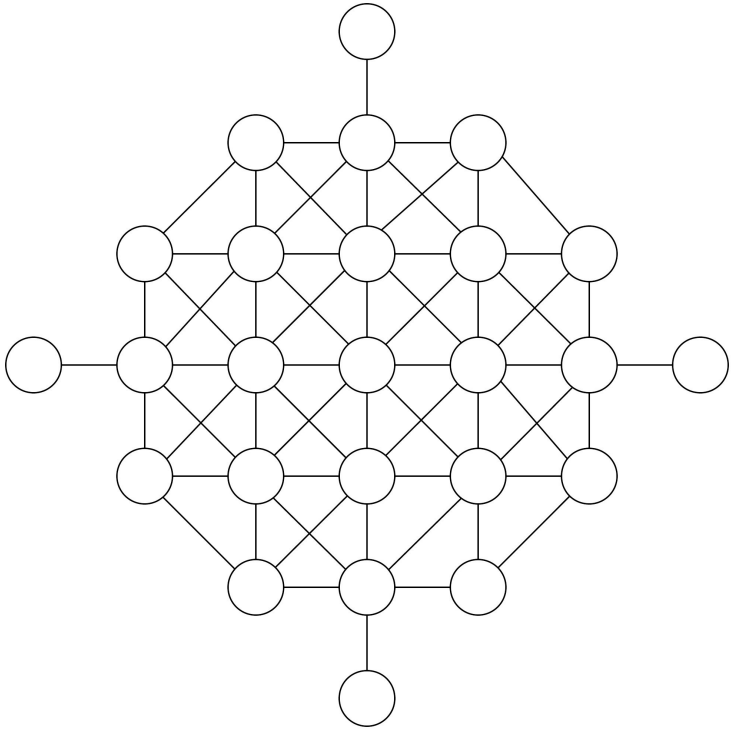
	0	1	2	3	4	5
Source	4					
parent	4	4	4	-1	4	1
frontier	5					
next						

Breadth First Search (BFS) Done

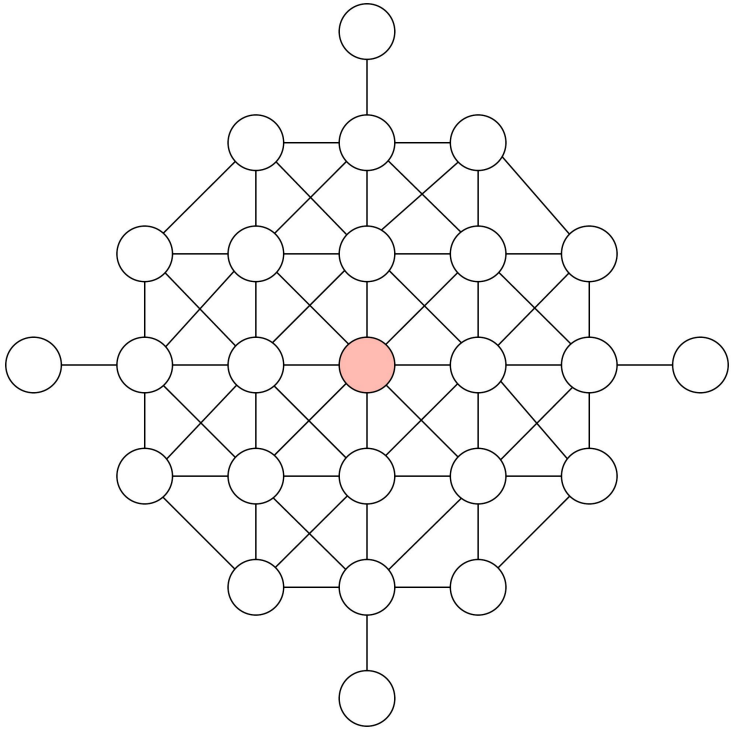




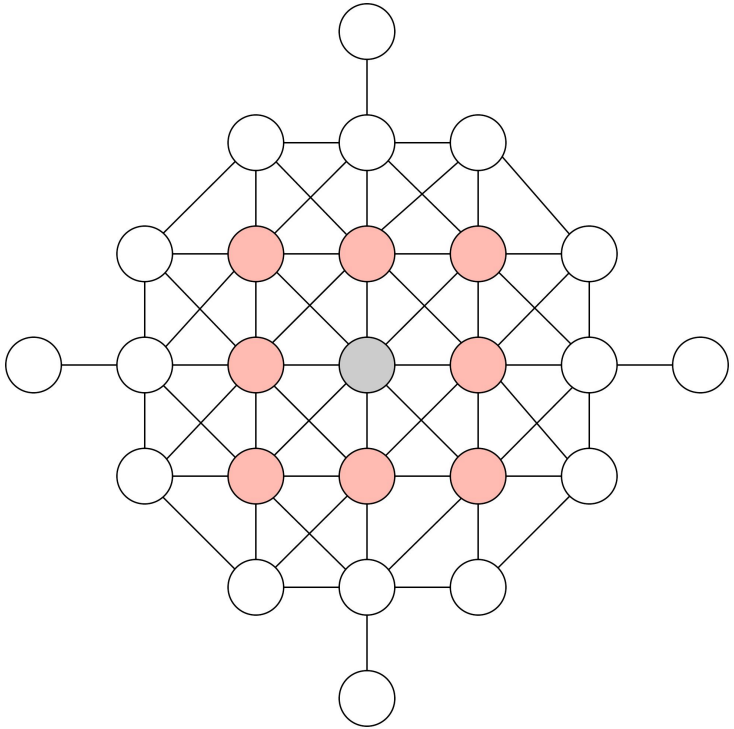
Breadth First Search (BFS)



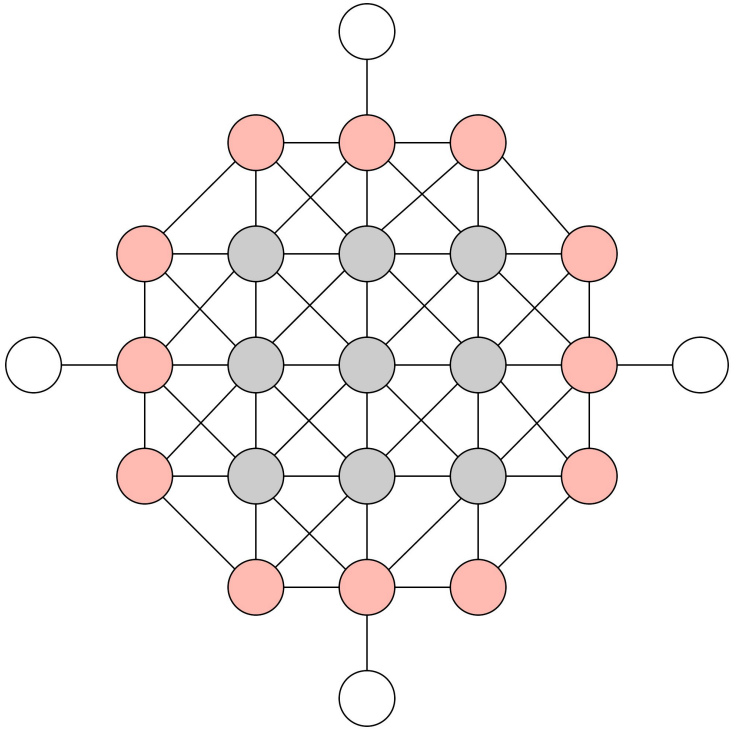
Breadth First Search (BFS)



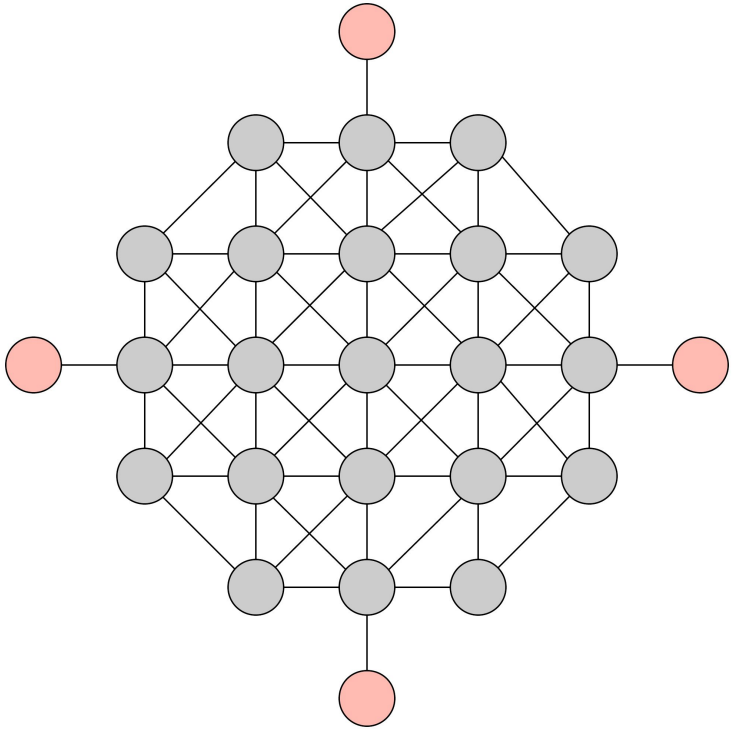
Breadth First Search (BFS)



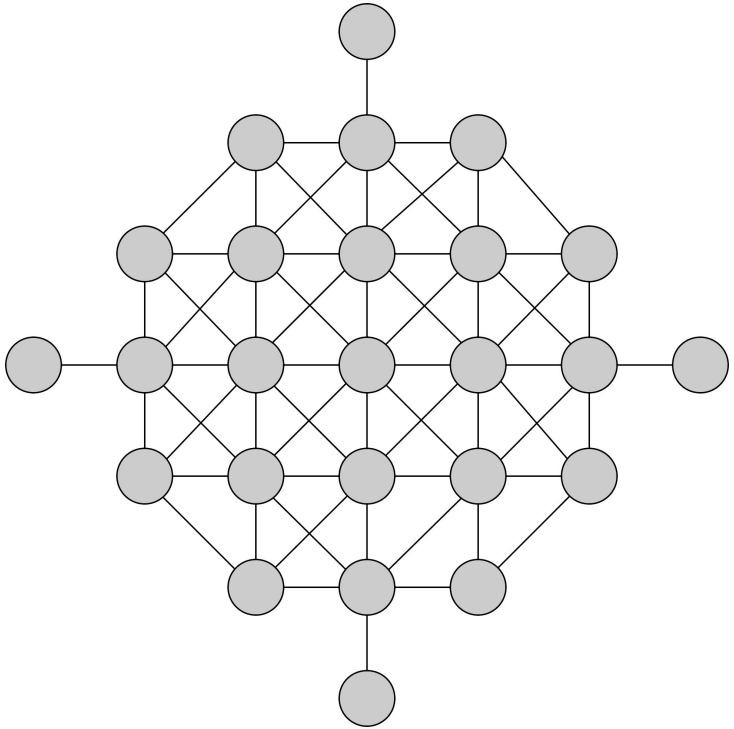
Breadth First Search (BFS)



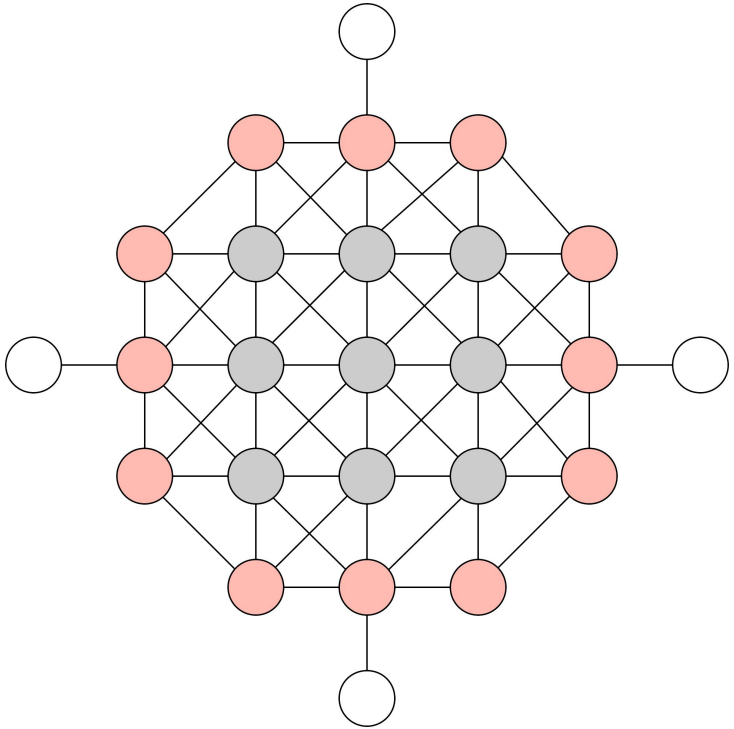
Breadth First Search (BFS)



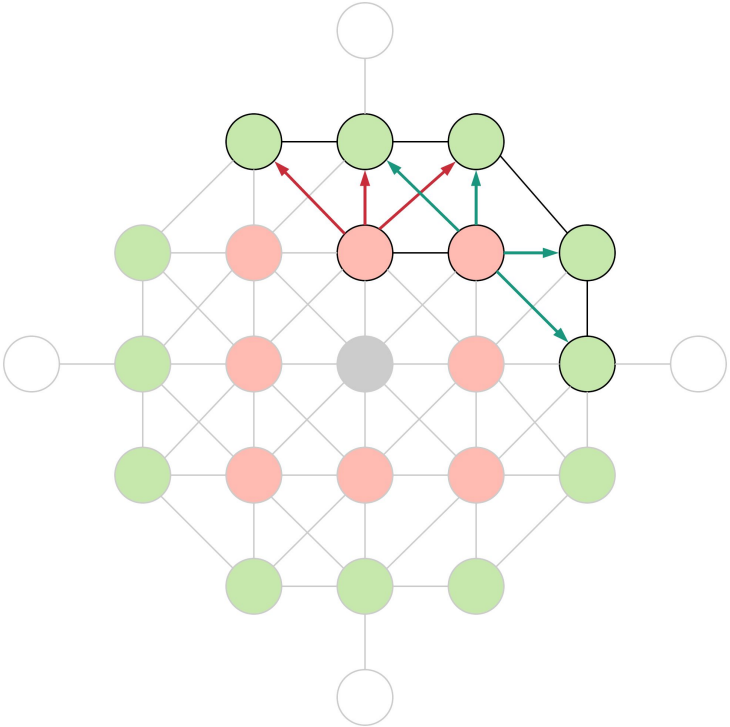
Breadth First Search (BFS)



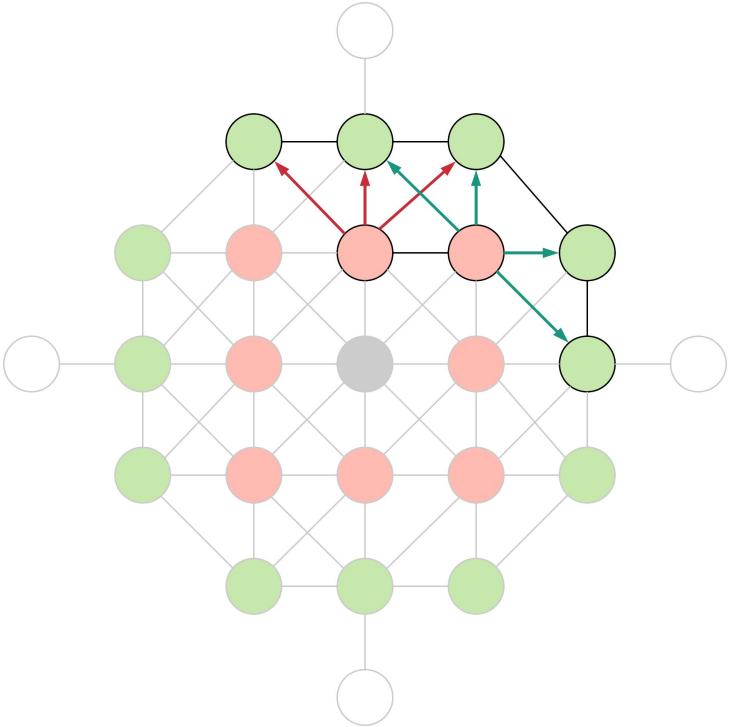
Breadth First Search (BFS) (push)



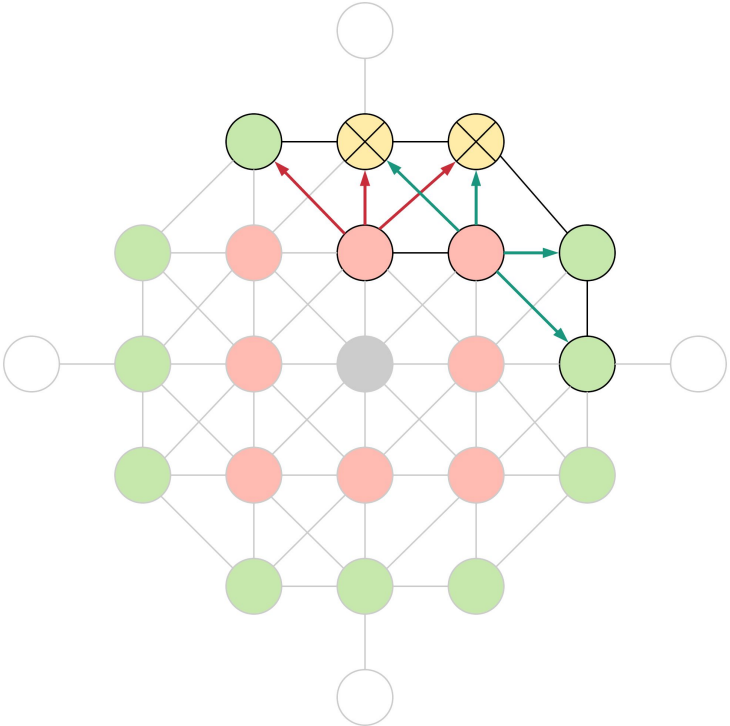
Breadth First Search (BFS) (push)



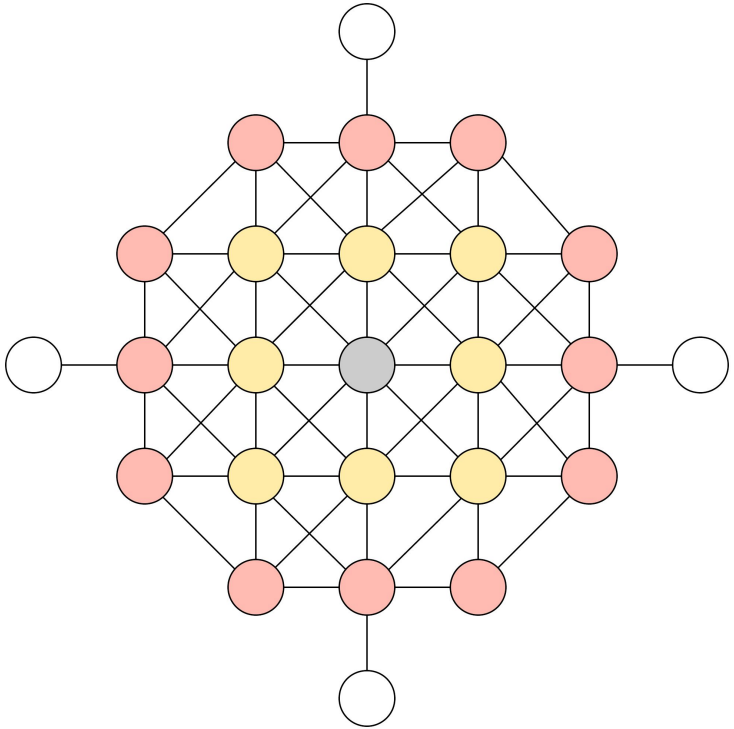
Breadth First Search (BFS) (push)



Breadth First Search (BFS) (push)



Breadth First Search (BFS) (pull)



Breadth First Search (BFS) (pull)