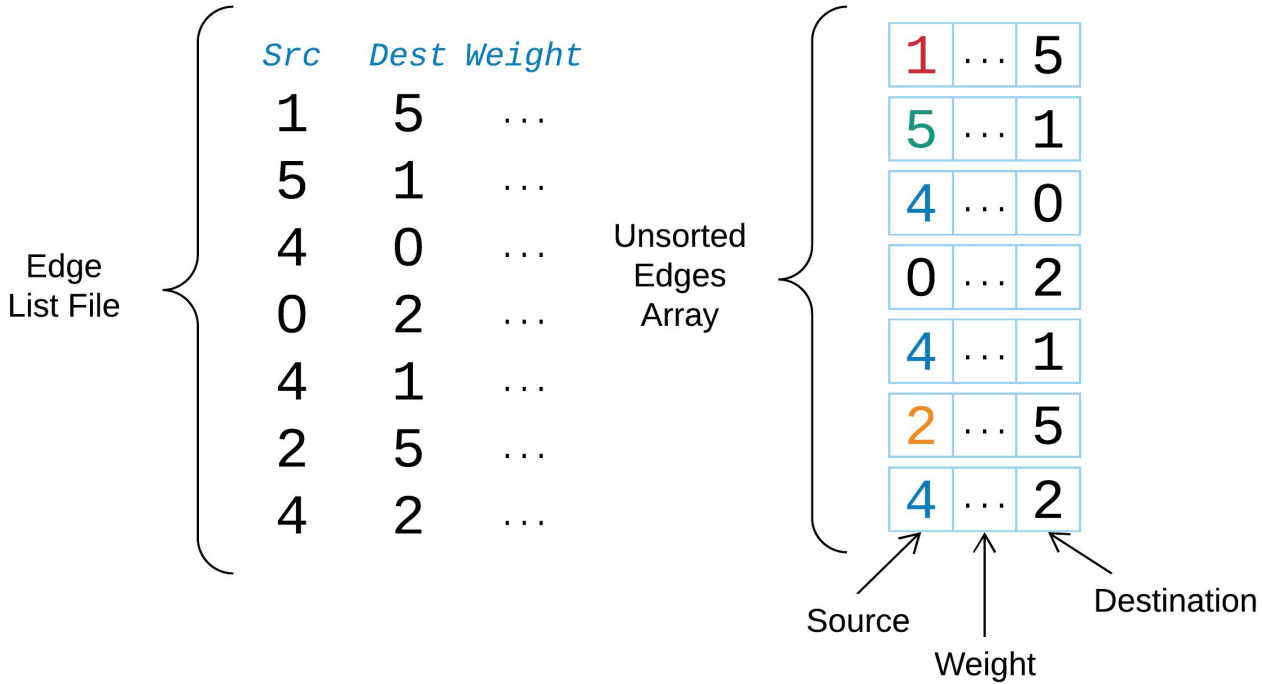
Sorting in linear time

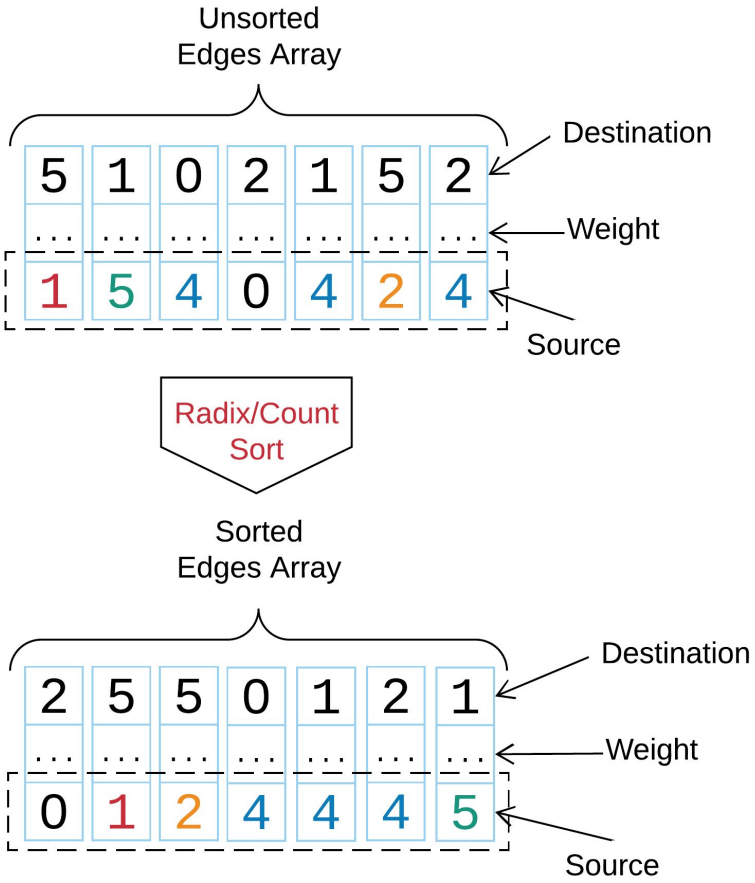
Counting sort serial and parallel.

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

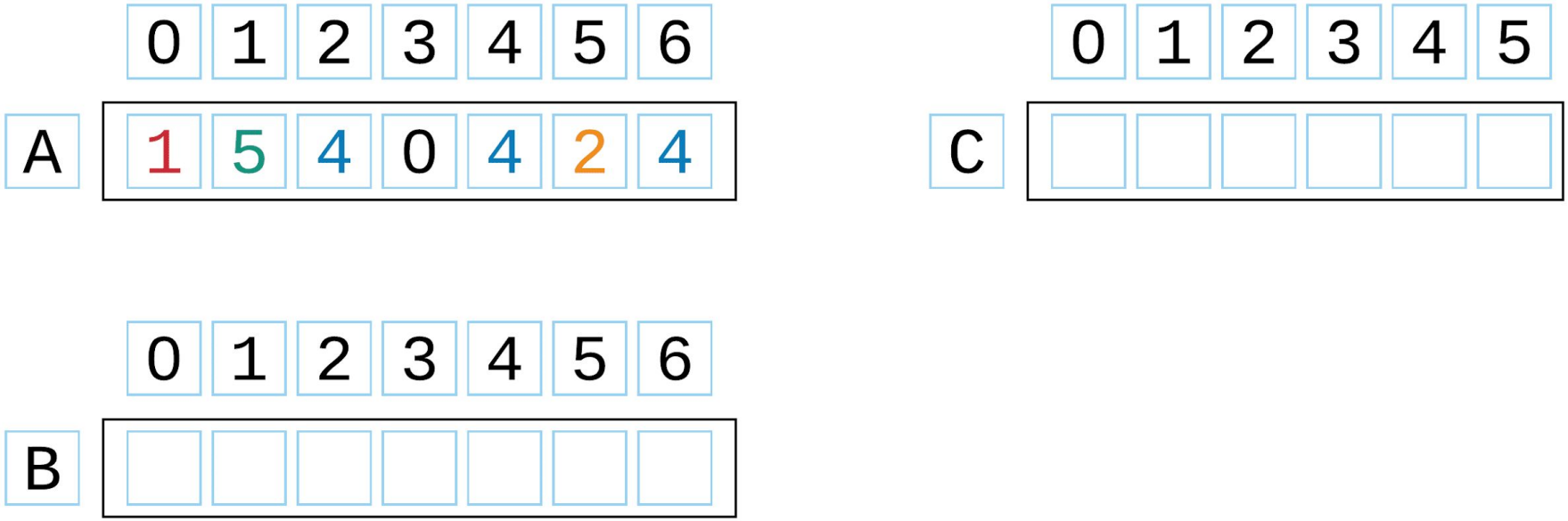
Counting-sort : No comparison between elements.

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

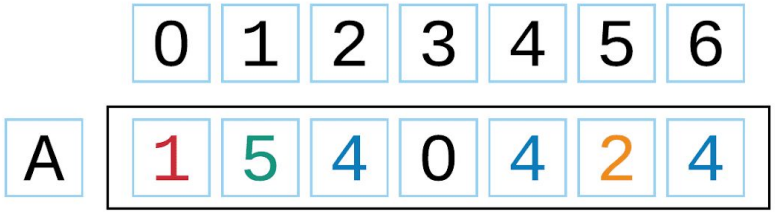
Counting-sort (Serial)

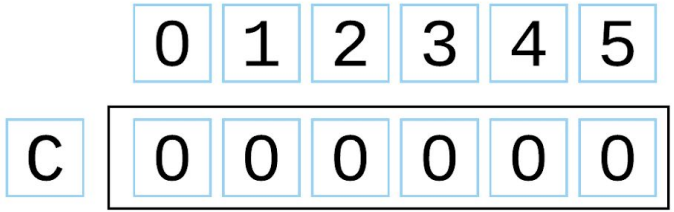
```
for i ←0 to (k-1)
   do C[i] ←0

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
```

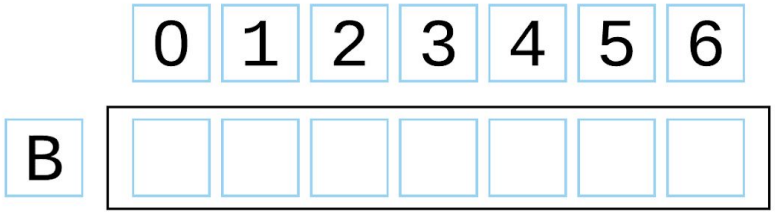
Counting-sort (Serial) - Example



Counting-sort (Serial) - Loop 1

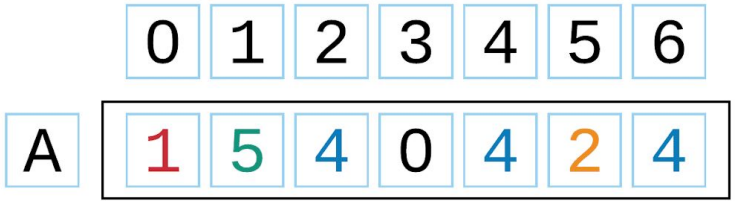


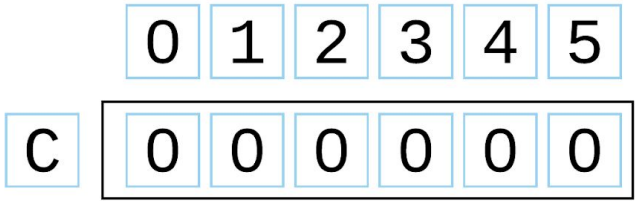


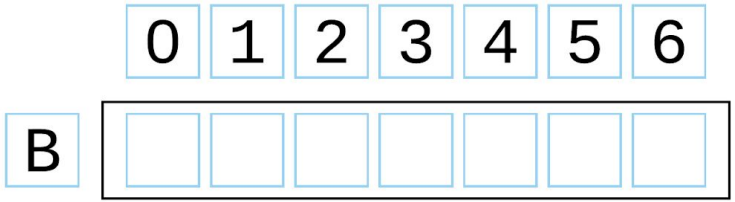



```
for i ←0 to (k-1)
  do C[i] ←0
```

Counting-sort (Serial) - Loop 2

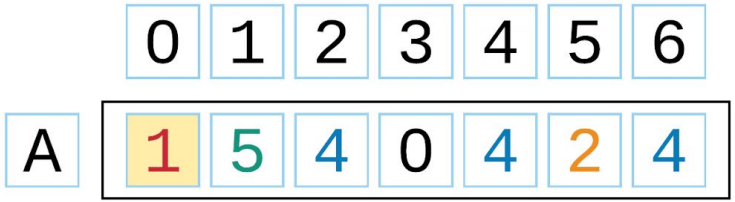


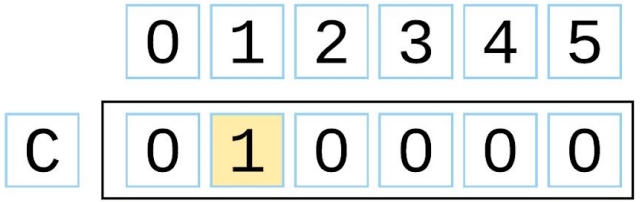


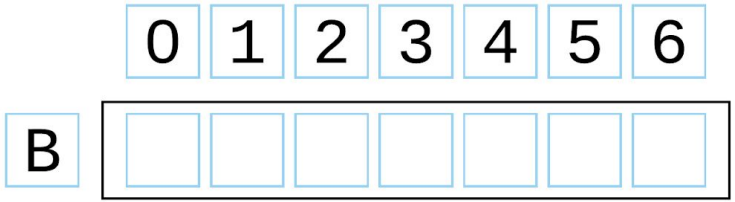



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 2

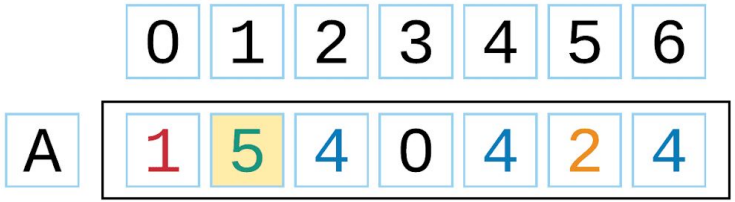


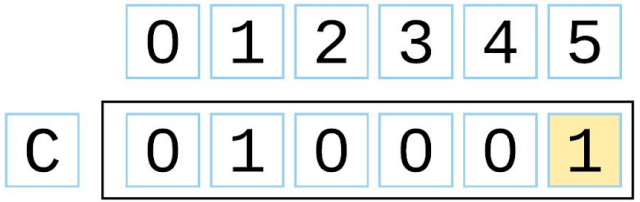


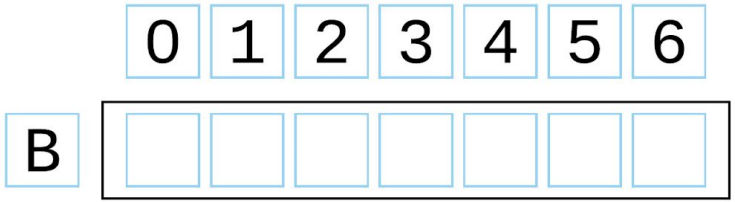



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 2

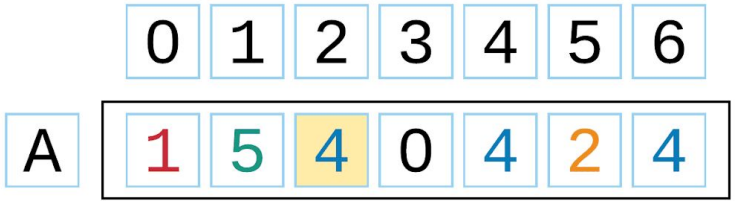


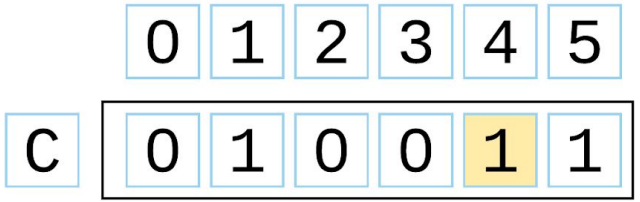


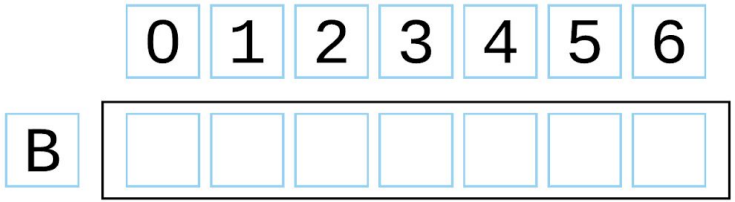



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 2

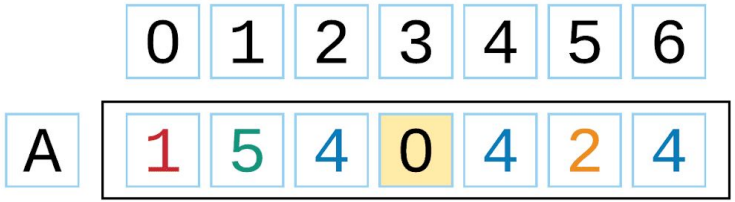


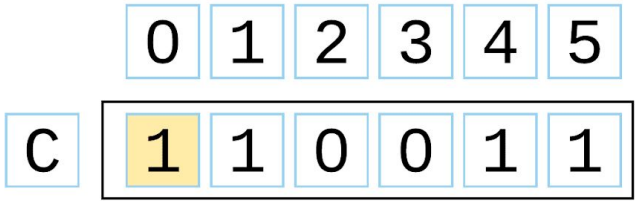


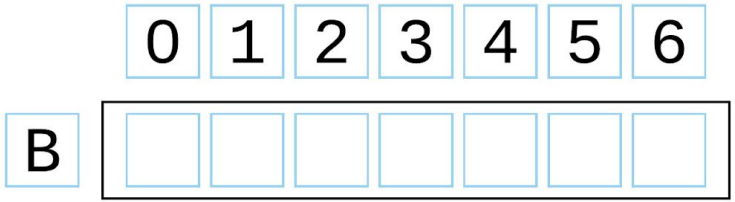



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 2

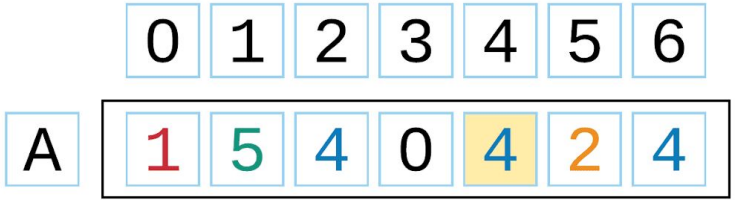


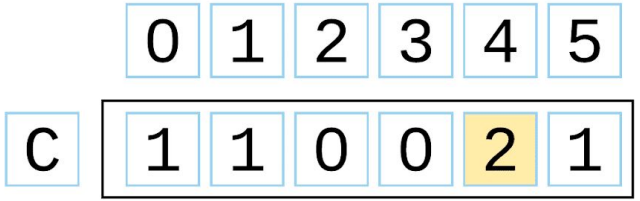


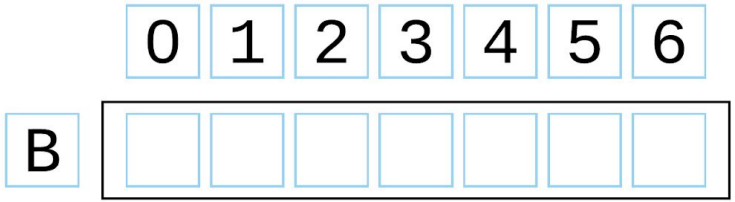



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 2

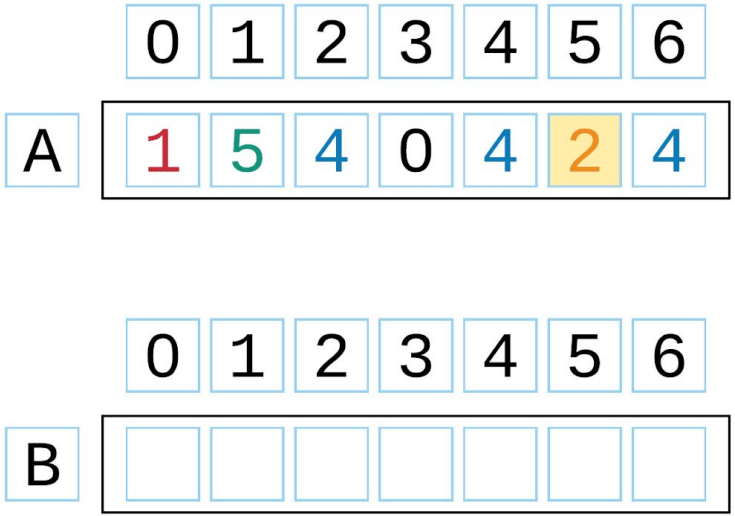


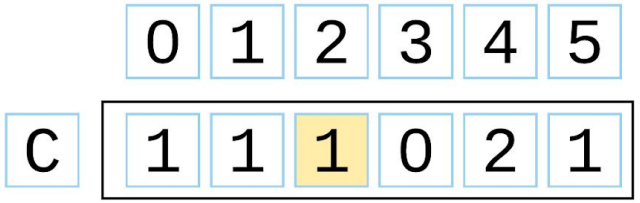





```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

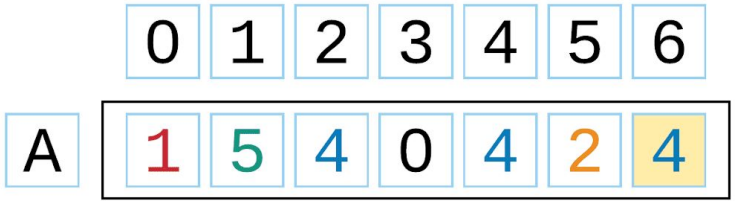
Counting-sort (Serial) - Loop 2

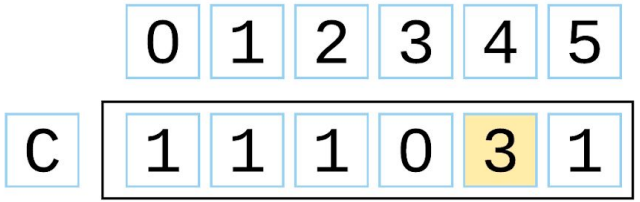


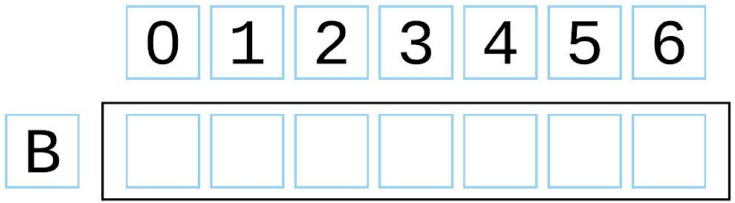



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 2

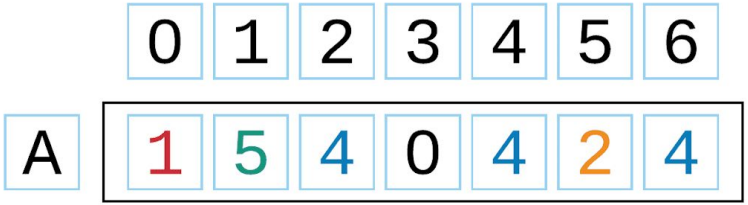


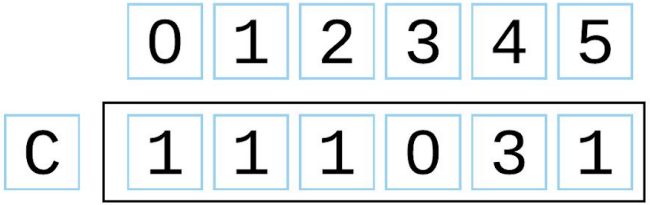


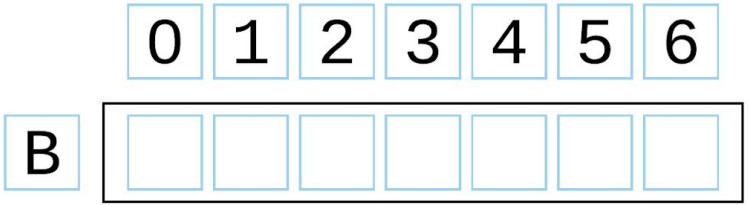



```
for j←0 to (n-1)
   do C[A[j]]←C[A[j]]+1
```

Counting-sort (Serial) - Loop 3

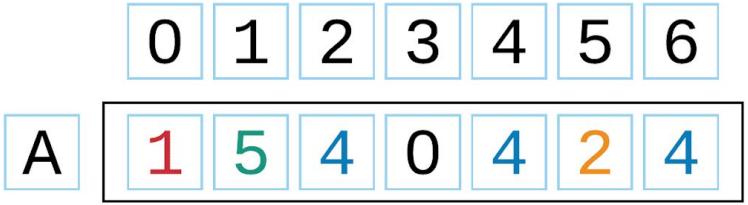


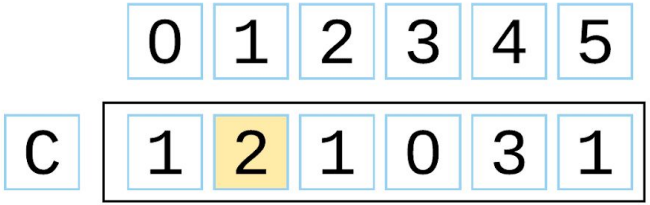


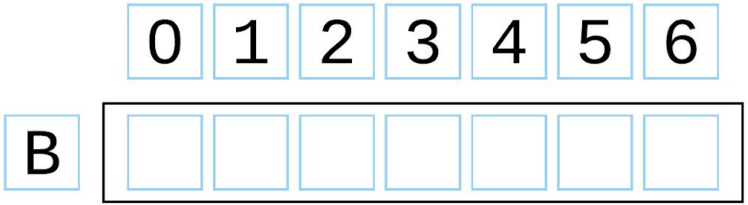



```
for i←1 to (k-1)
   do C[i]←C[i]+C[i-1]
```

Counting-sort (Serial) - Loop 3

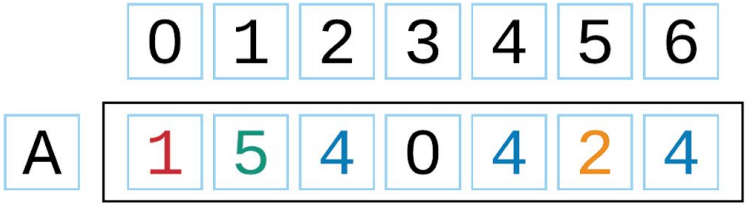


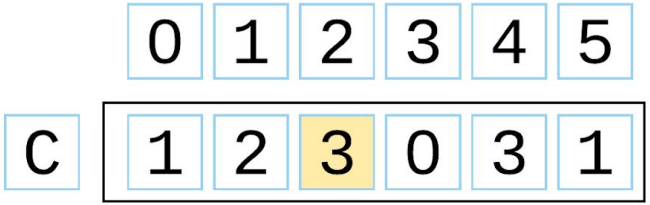


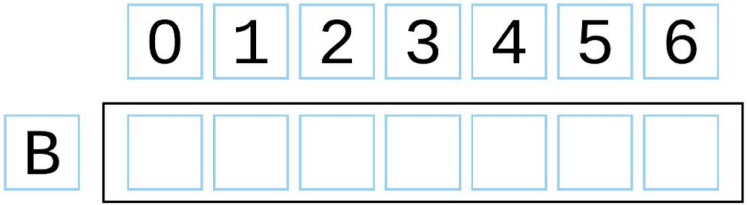



```
for i←1 to (k-1)
   do C[i]←C[i]+C[i-1]
```

Counting-sort (Serial) - Loop 3

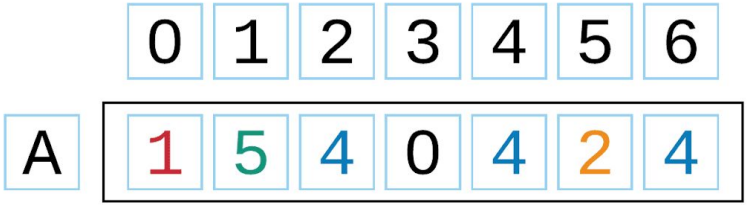


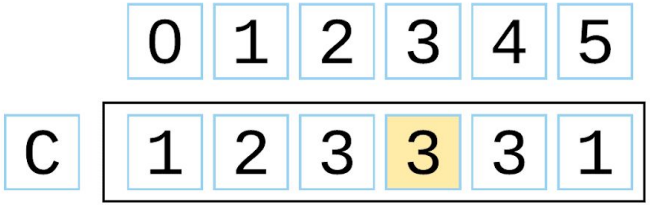


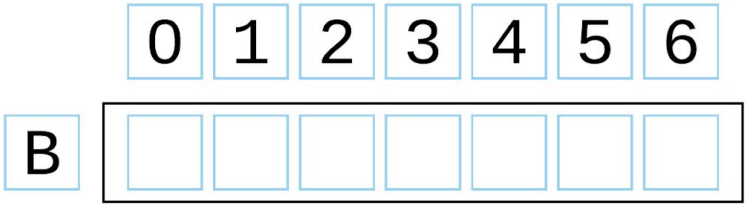



```
for i←1 to (k-1)
   do C[i]←C[i]+C[i-1]
```

Counting-sort (Serial) - Loop 3

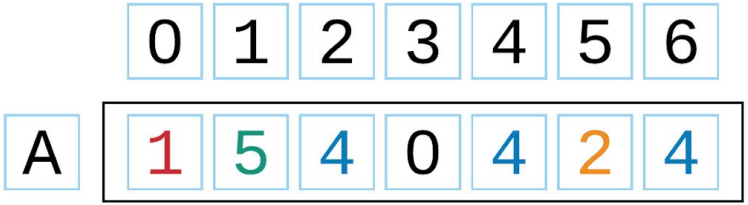


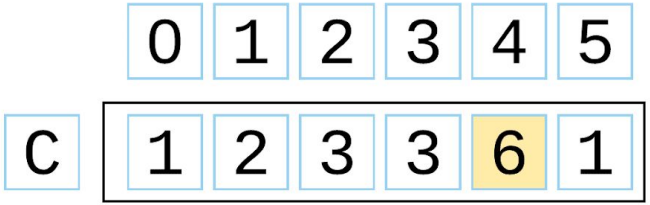


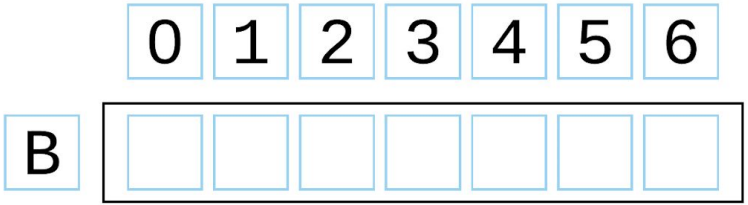



```
for i←1 to (k-1)
   do C[i]←C[i]+C[i-1]
```

Counting-sort (Serial) - Loop 3

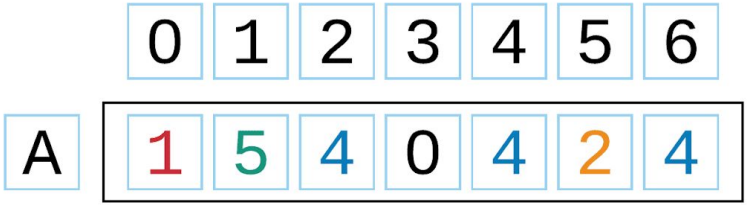


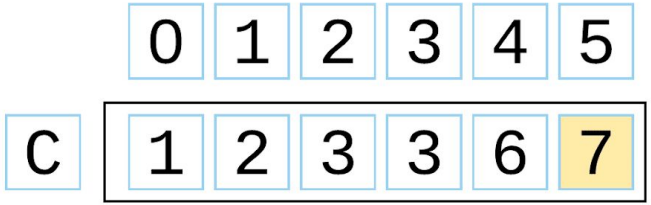


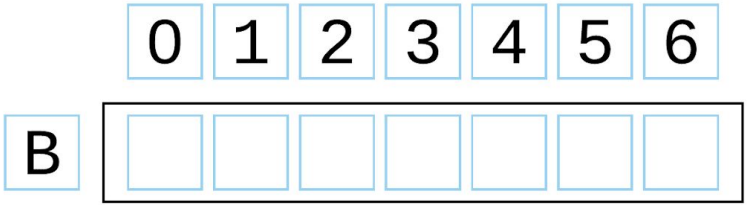



```
for i←1 to (k-1)
   do C[i]←C[i]+C[i-1]
```

Counting-sort (Serial) - Loop 3

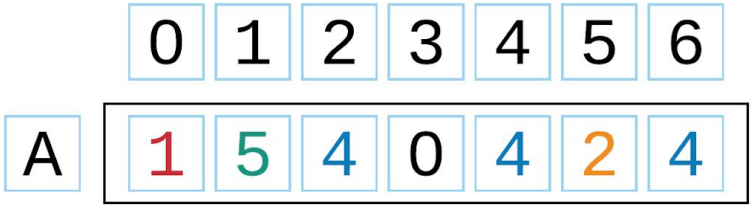


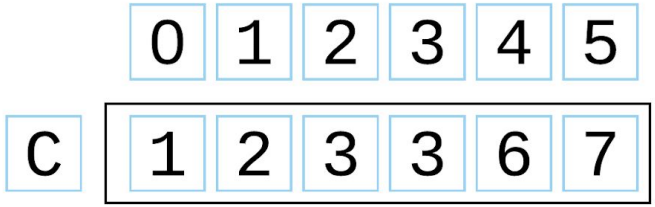


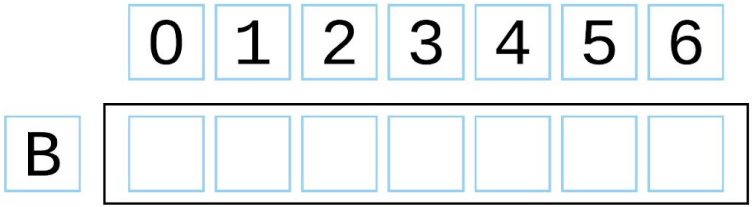



```
for i←1 to (k-1)
   do C[i]←C[i]+C[i-1]
```

Counting-sort (Serial) - Loop 4

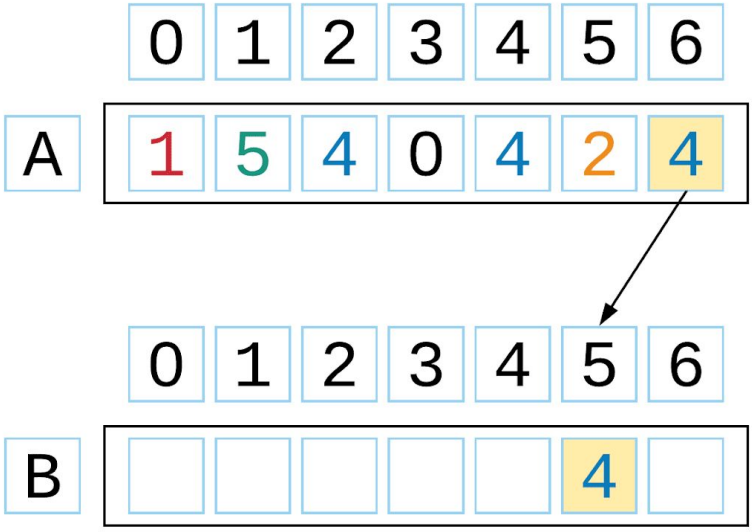


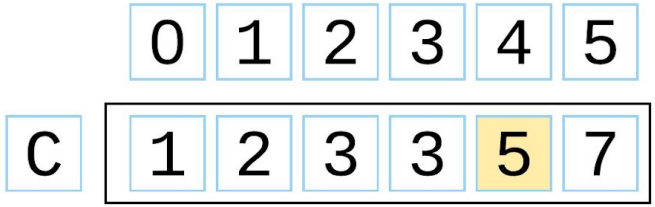





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

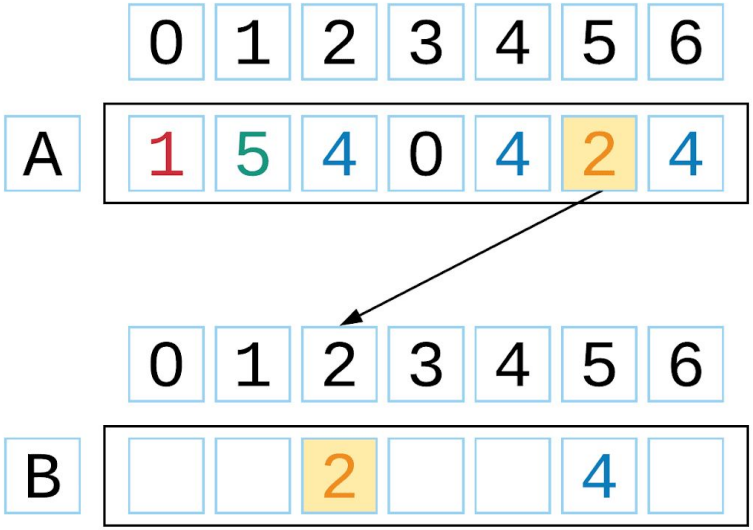
Counting-sort (Serial) - Loop 4

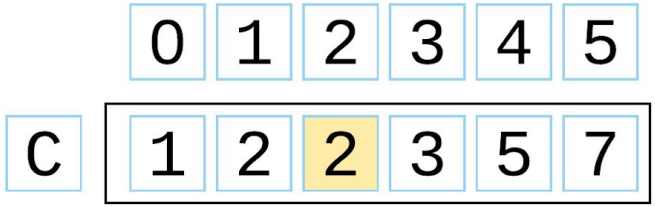





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

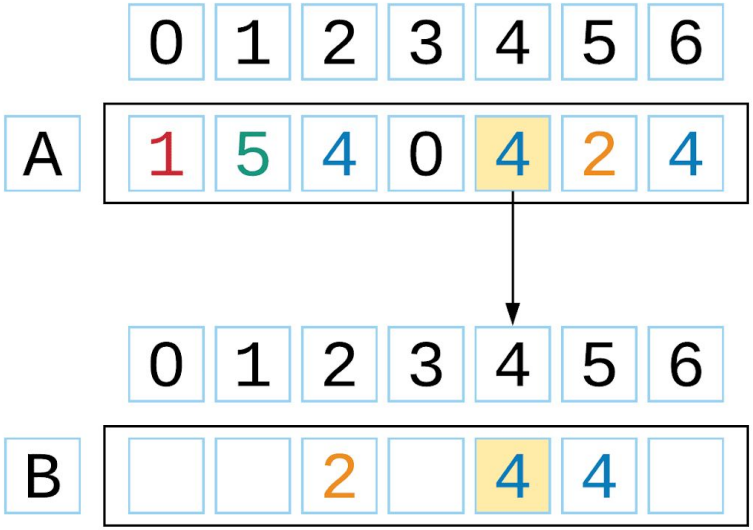
Counting-sort (Serial) - Loop 4

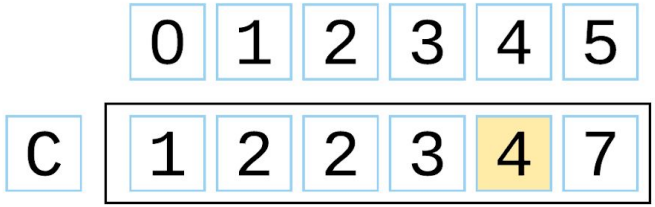





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

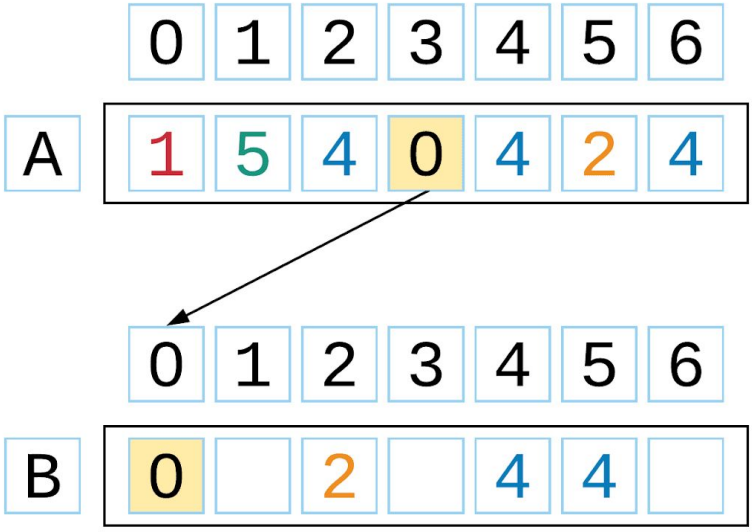
Counting-sort (Serial) - Loop 4

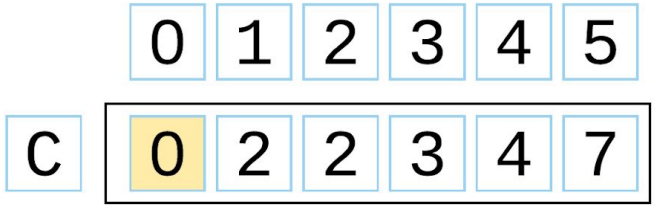





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

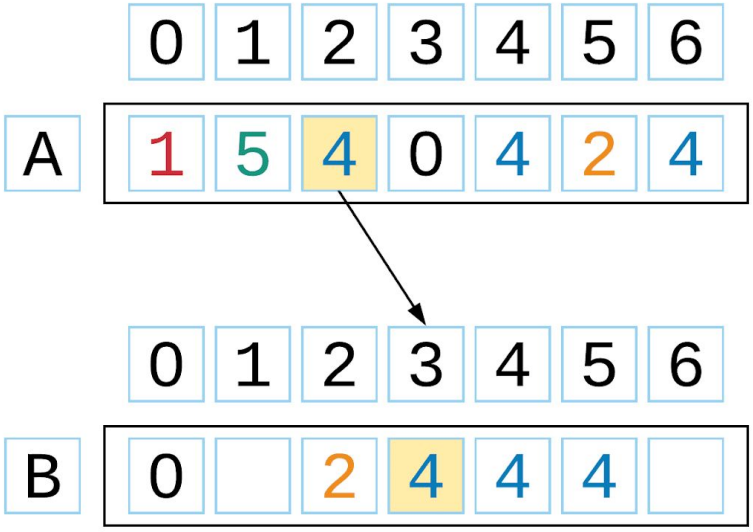
Counting-sort (Serial) - Loop 4

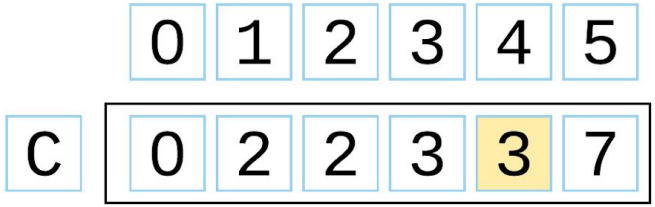





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

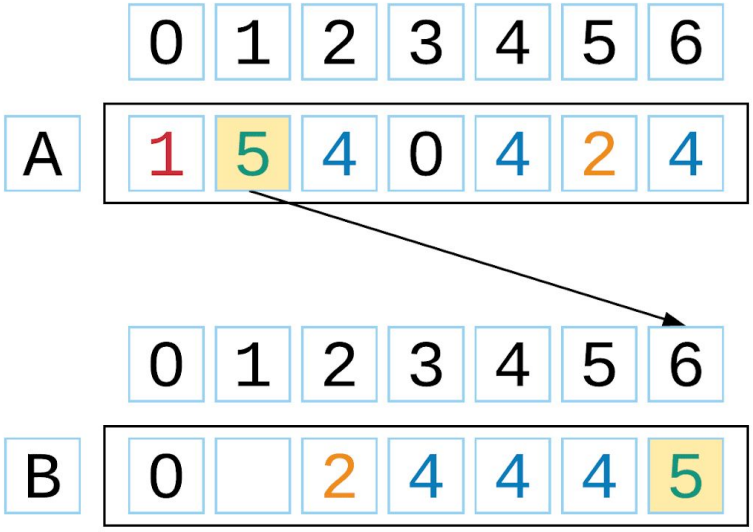
Counting-sort (Serial) - Loop 4

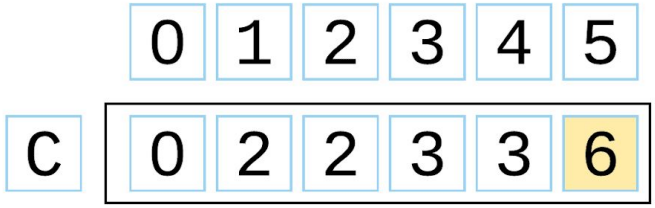





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

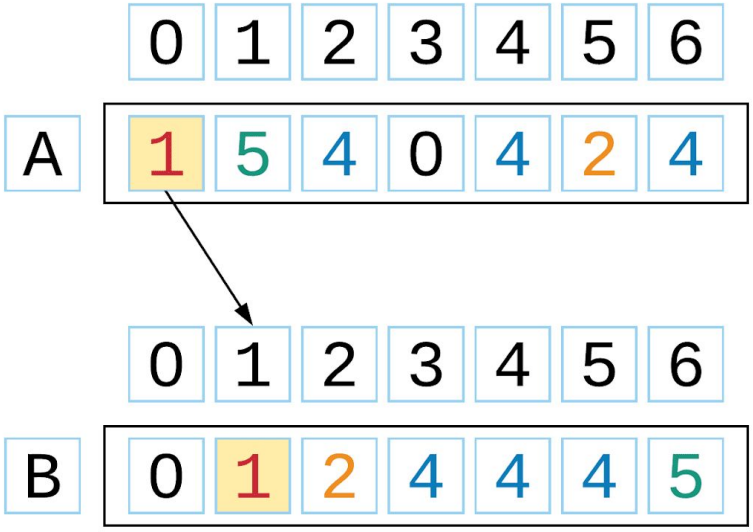
Counting-sort (Serial) - Loop 4

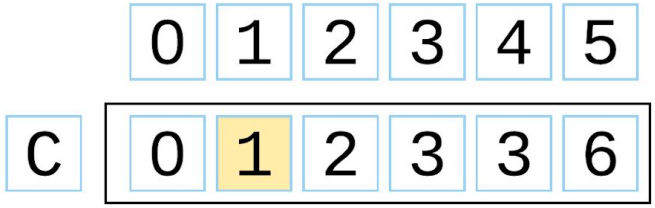





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

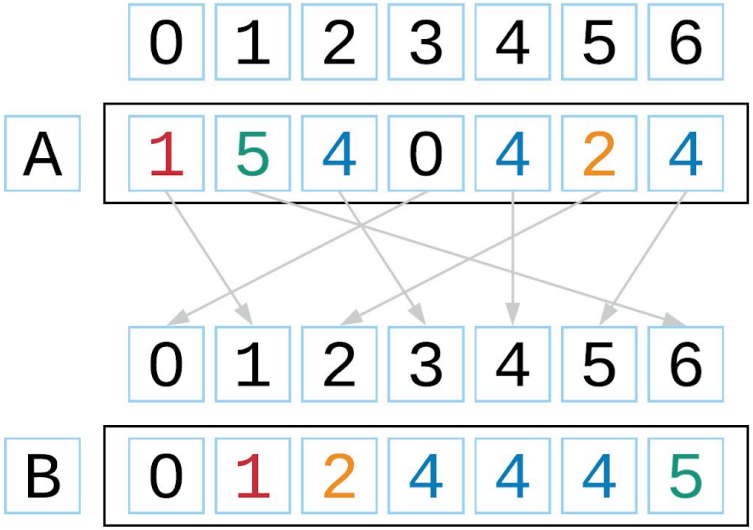
Counting-sort (Serial) - Loop 4

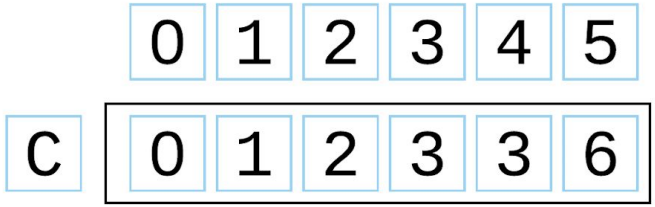





```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

Counting-sort (Serial) - Loop 4



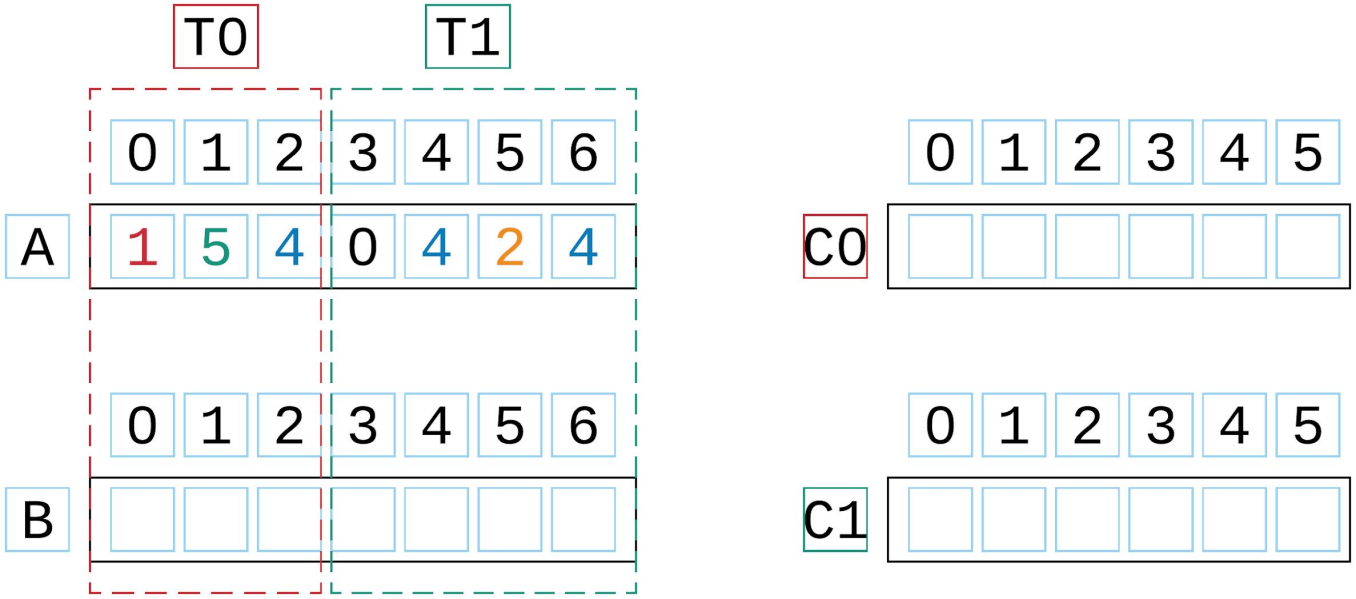



```
for j ← (n-1) downto 0
    do B[C[A[j]]-1] ← A[j]
       C[A[j]] ← C[A[j]]-1
```

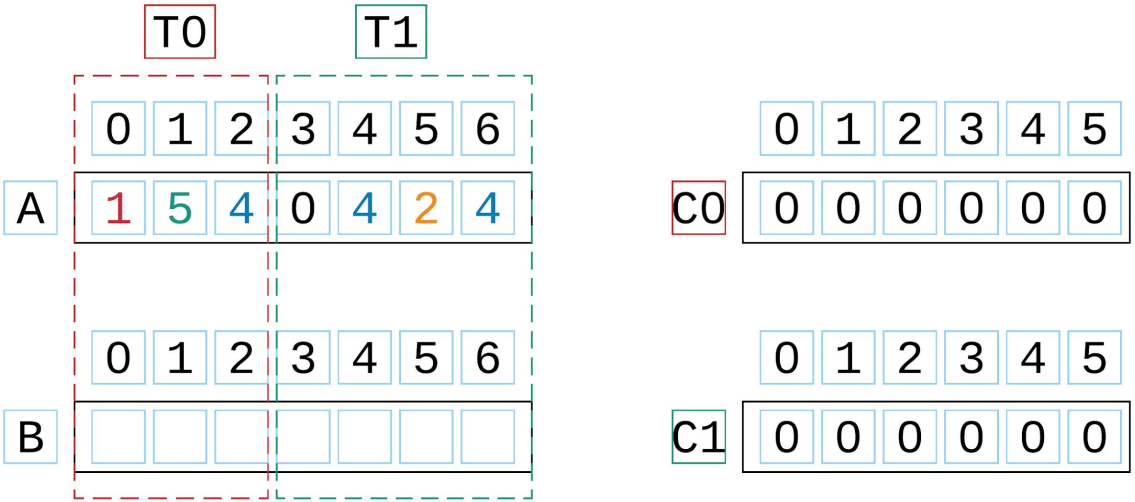
Counting-sort (Parallel)

```
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 downto tid_start
   do B[C[tid][A[j]]-1] ←A[j]
      C[tid][A[j]]←C[tid][A[j]]-1
```

Counting-sort (Parallel)

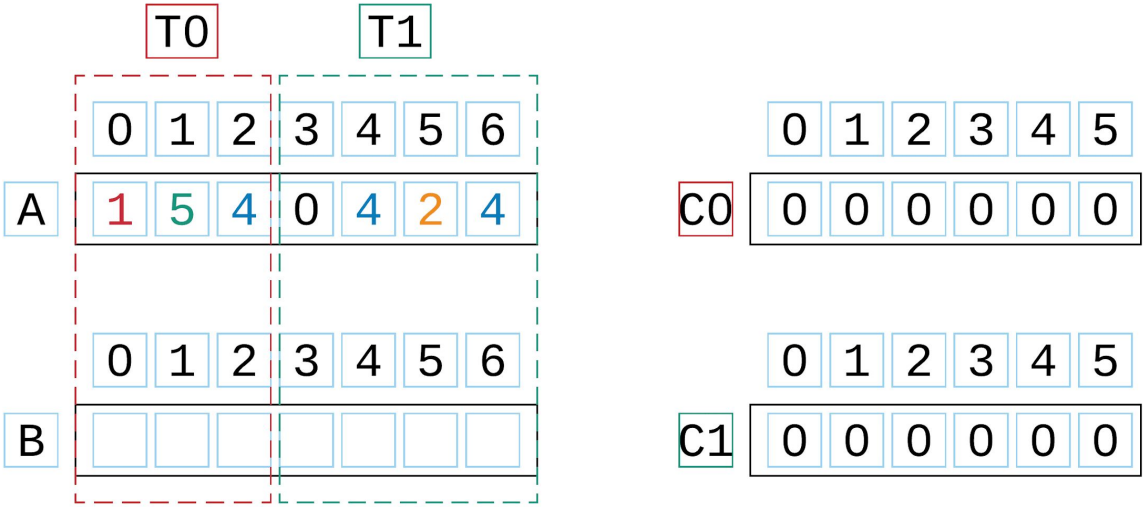


Counting-sort (Parallel) - Loop 1



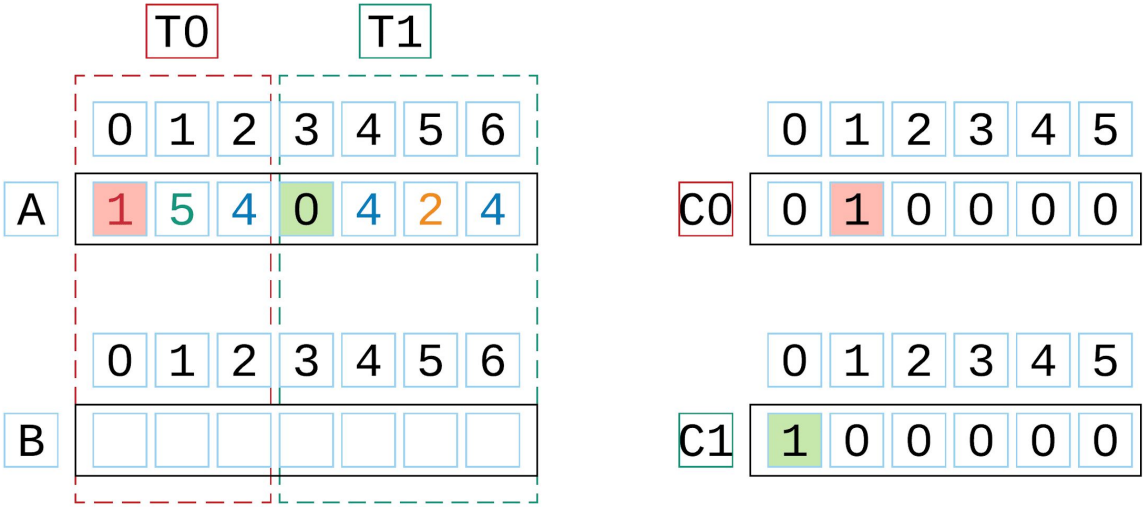

```
Parallel for i←0 to (k-1)
  do C[tid][i]←0
```

Counting-sort (Parallel) - Loop 2



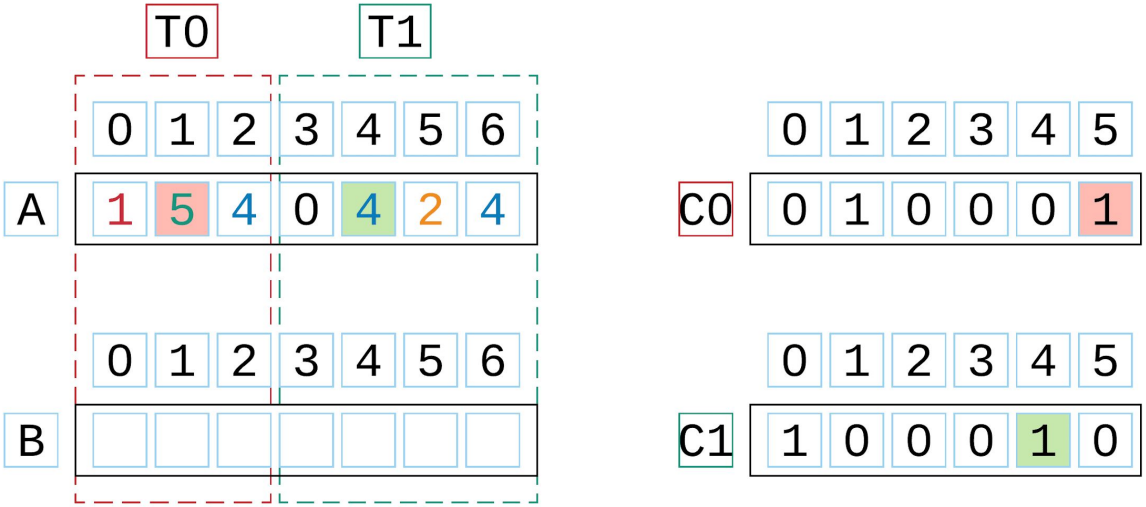

```
Parallel for j ←tid_start to tid_end
   do C[tid][A[j]]←C[tid][A[j]]+1
```

Counting-sort (Parallel) - Loop 2



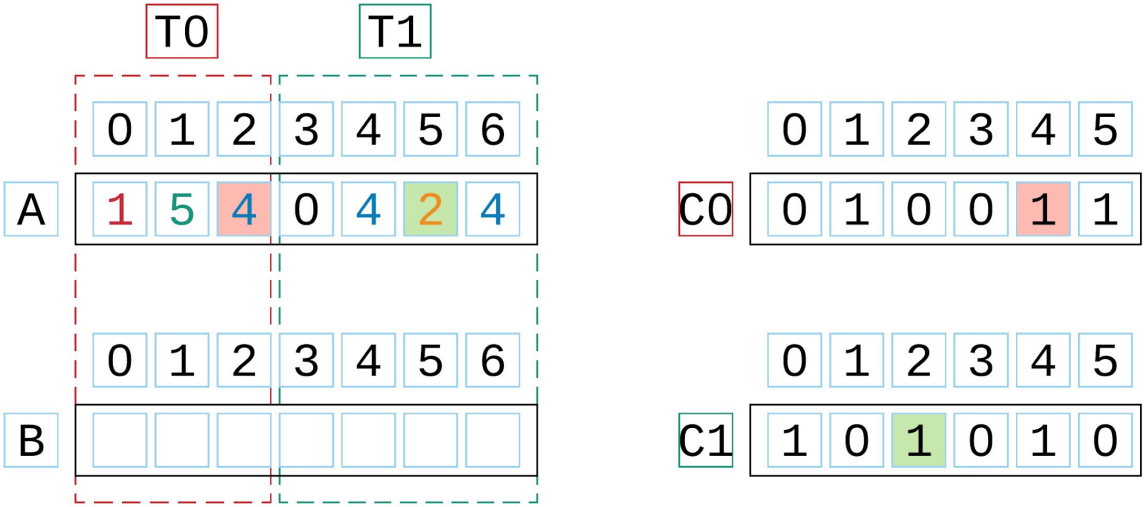

```
Parallel for j ←tid_start to tid_end
   do C[tid][A[j]]←C[tid][A[j]]+1
```

Counting-sort (Parallel) - Loop 2



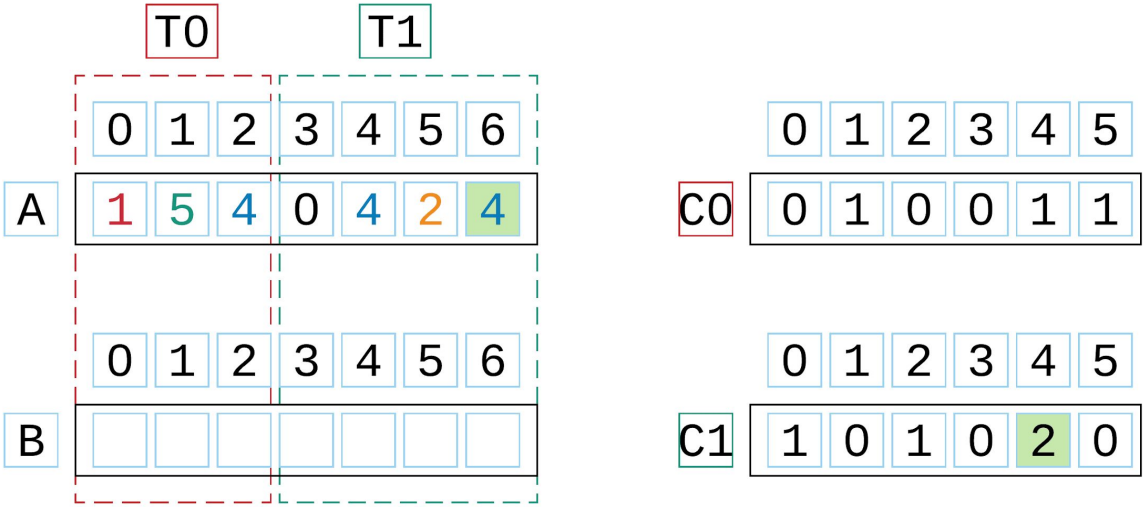

```
Parallel for j ←tid_start to tid_end
   do C[tid][A[j]]←C[tid][A[j]]+1
```

Counting-sort (Parallel) - Loop 2



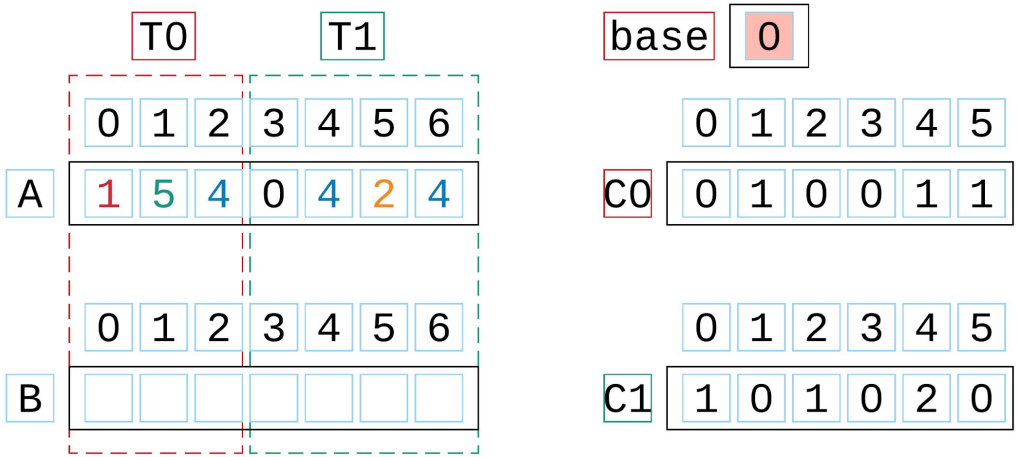

```
Parallel for j ←tid_start to tid_end
   do C[tid][A[j]]←C[tid][A[j]]+1
```

Counting-sort (Parallel) - Loop 2



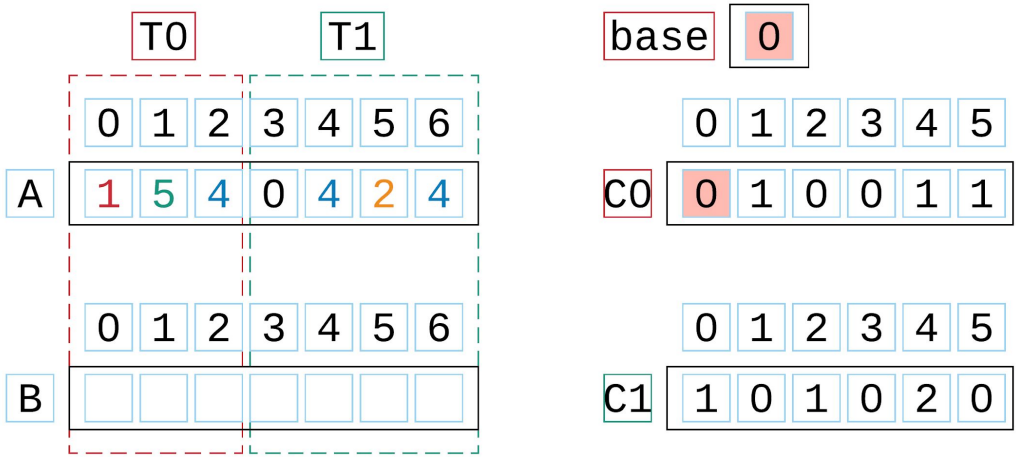

```
Parallel for j ←tid_start to tid_end
   do C[tid][A[j]]←C[tid][A[j]]+1
```

Counting-sort (Parallel) - Loop 3



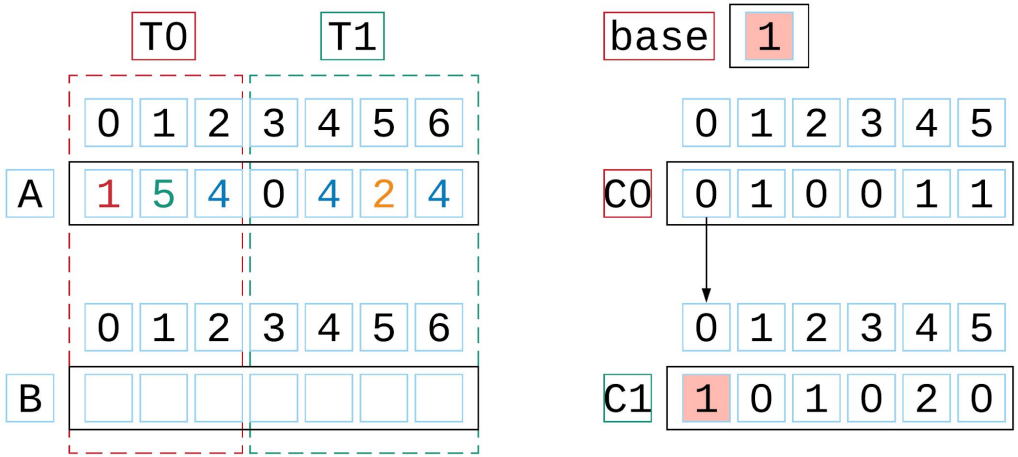

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



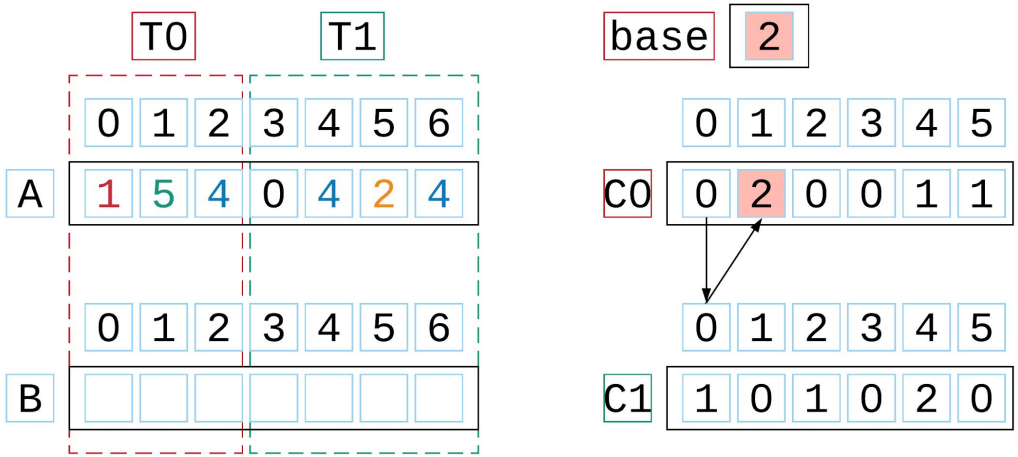

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



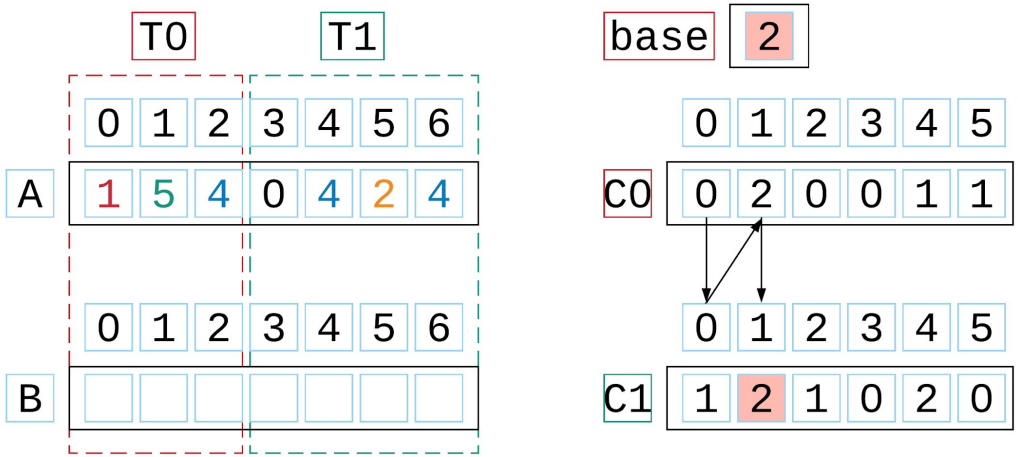

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



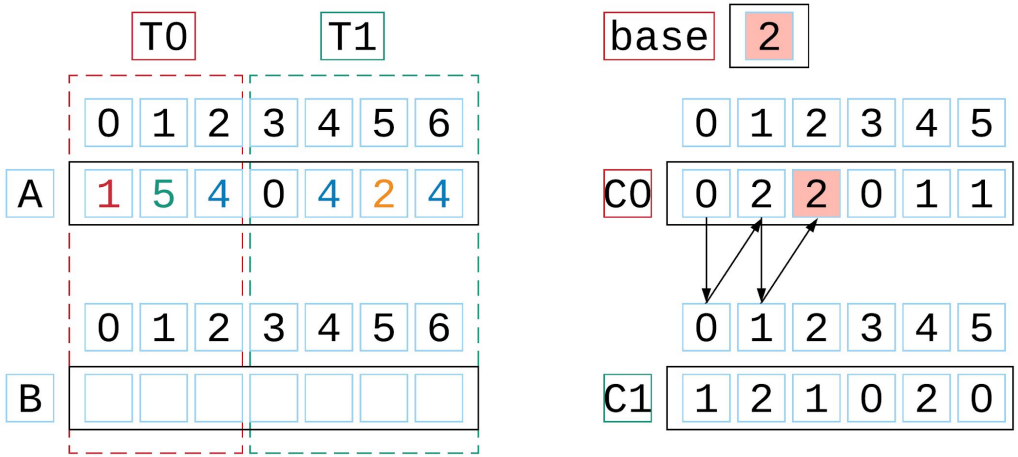

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



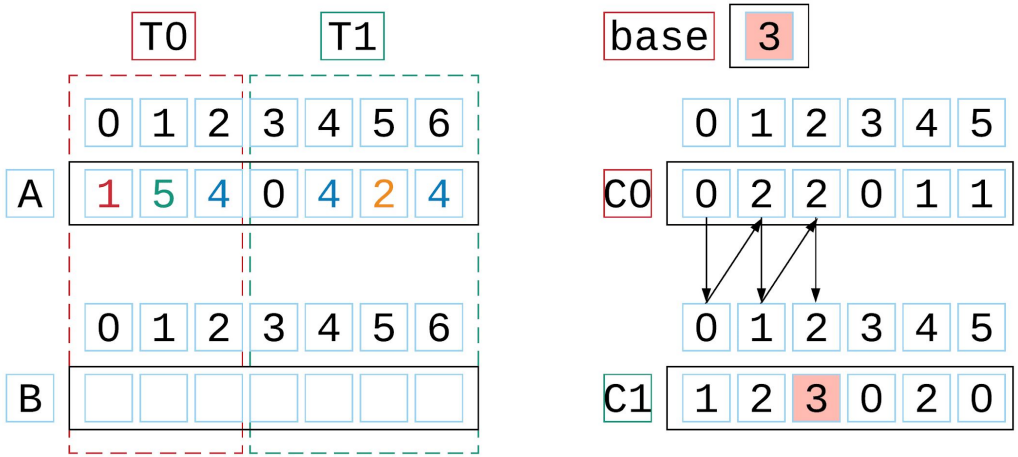

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



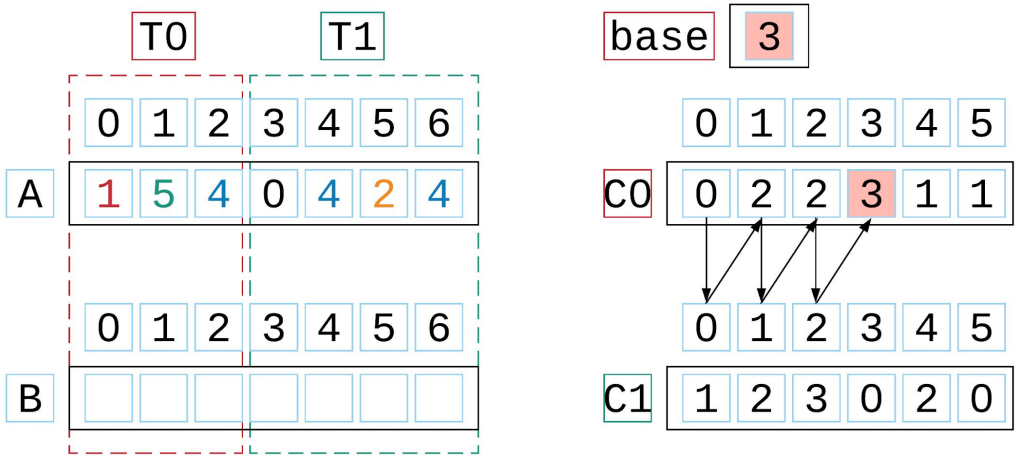

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



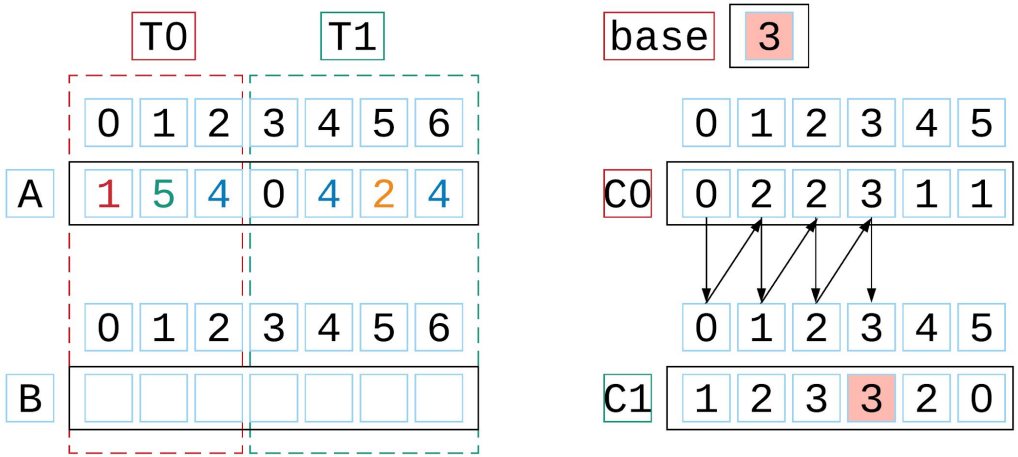

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



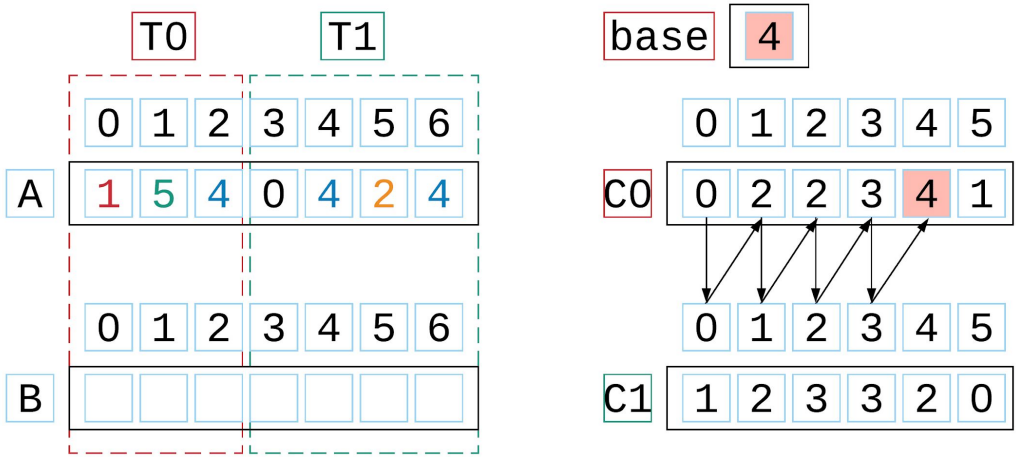

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



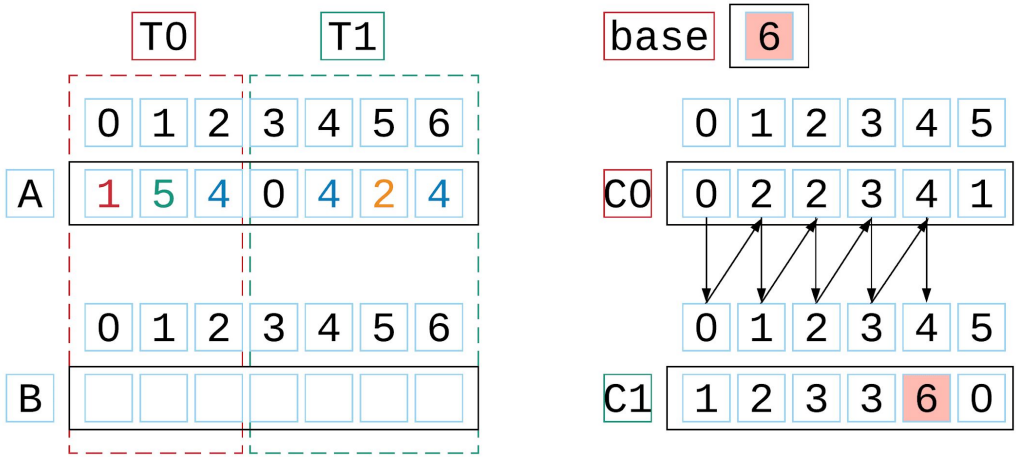

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



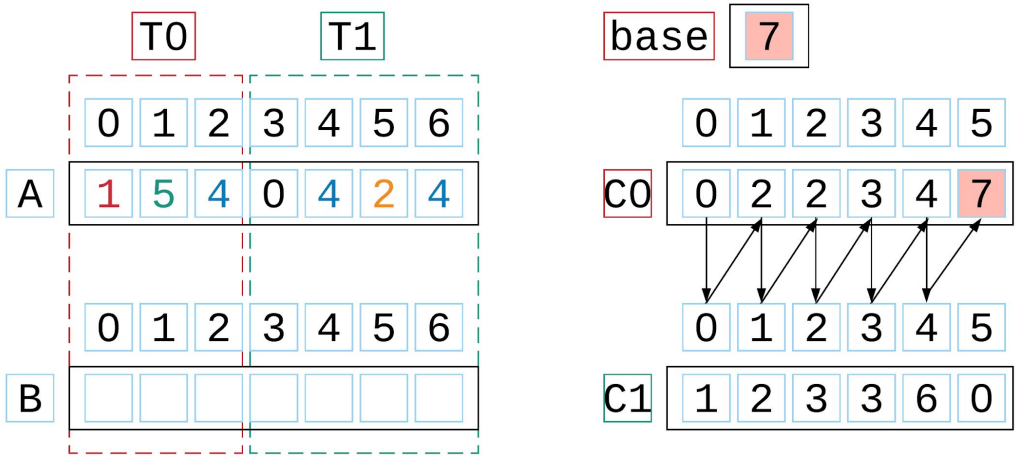

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



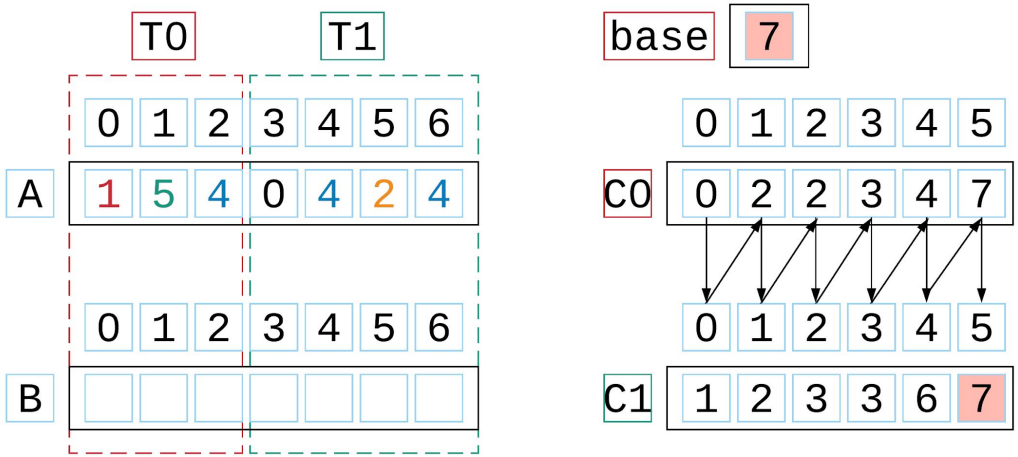

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



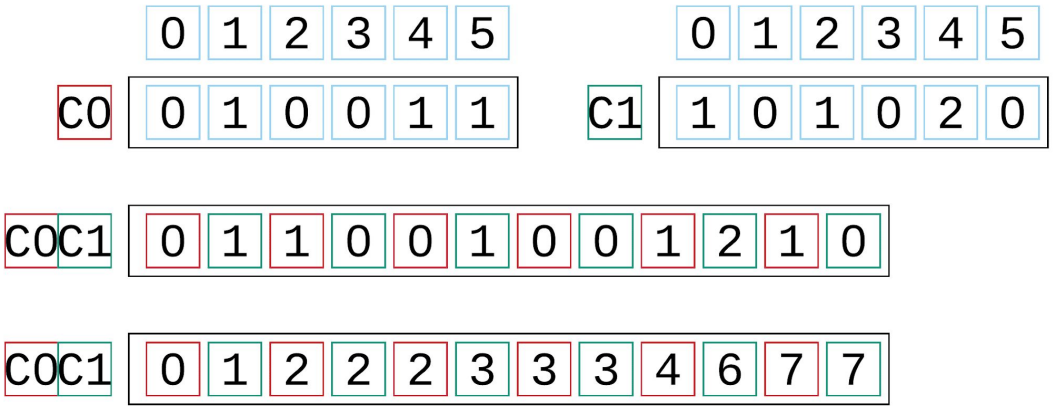

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



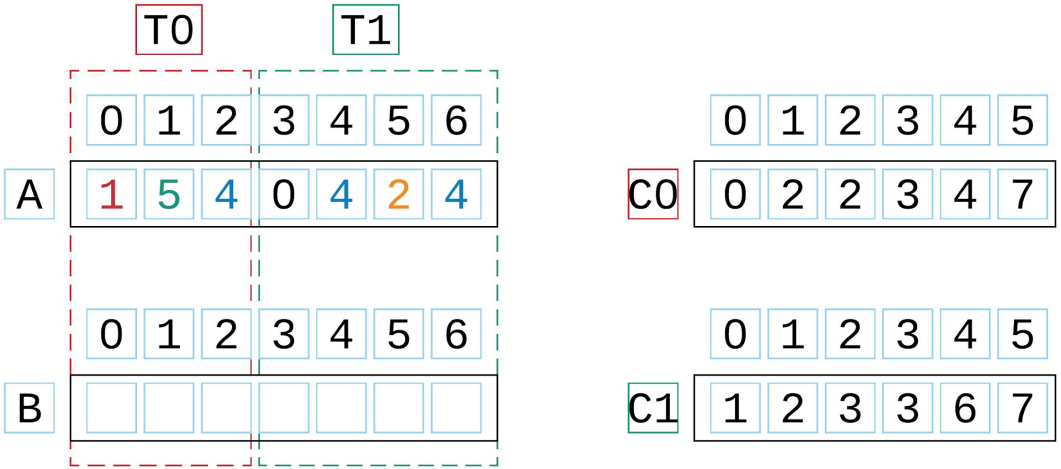

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 3



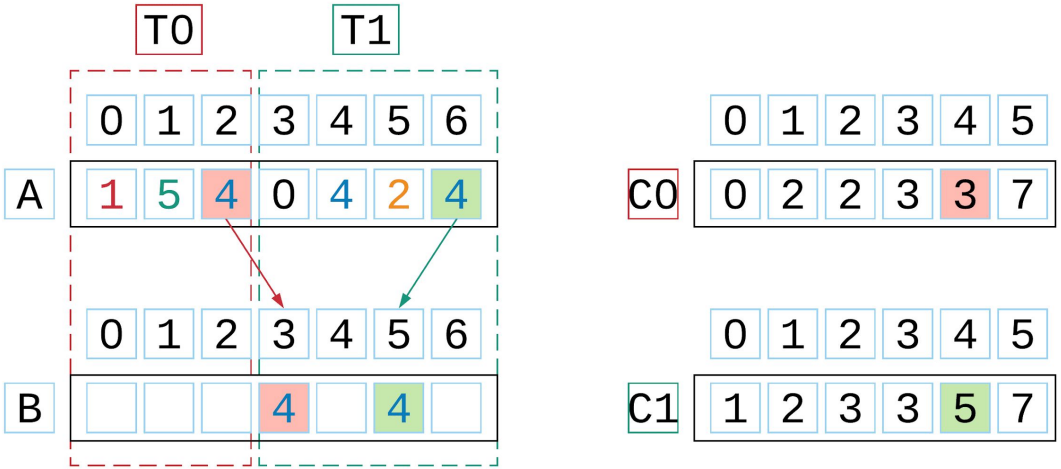

```
for i←0 to (k-1)
   for tid←0 to (t-1)
      do base←base + C[tid][i]
         C[tid][i]←base
```

Counting-sort (Parallel) - Loop 4



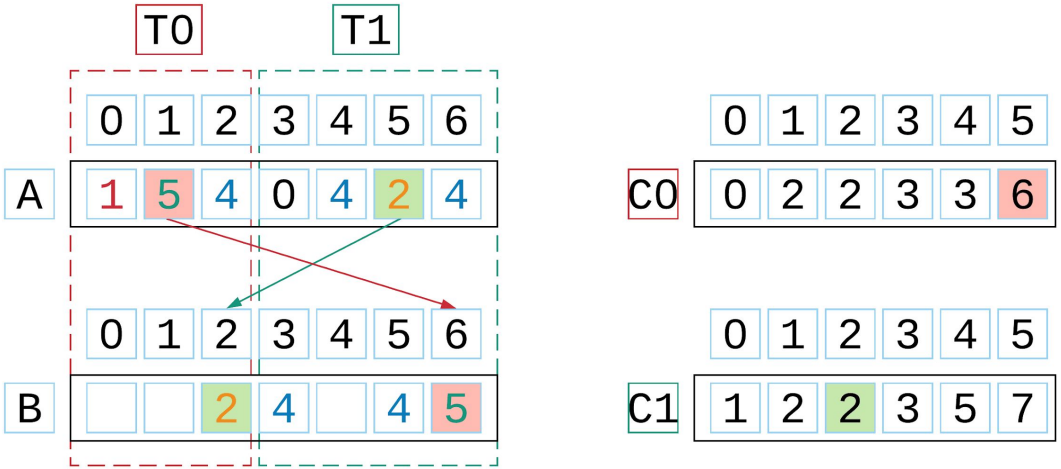

```
Parallel for j ←tid_end downto tid_start
    do B[C[tid][A[j]]-1] ←A[j]
       C[tid][A[j]]←C[tid][A[j]]-1
```

Counting-sort (Parallel) - Loop 4



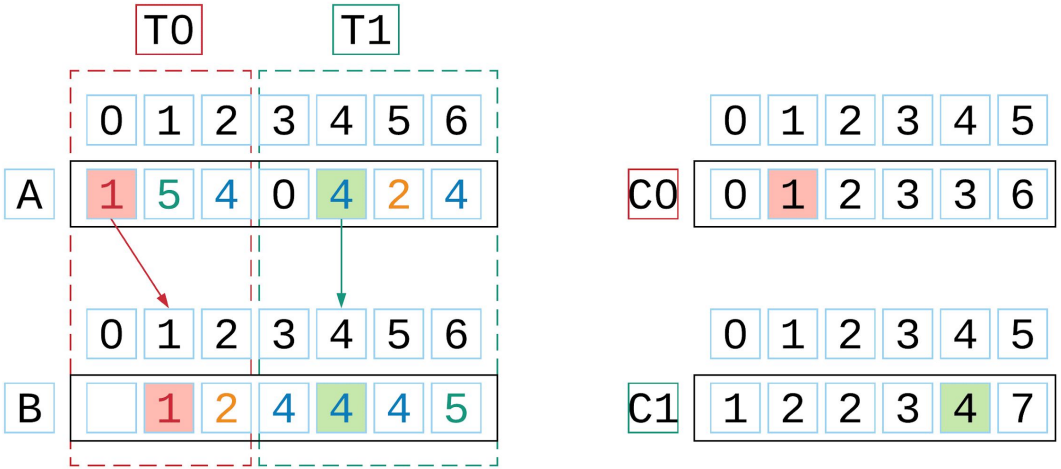

```
Parallel for j ←tid_end downto tid_start
    do B[C[tid][A[j]]-1] ←A[j]
       C[tid][A[j]]←C[tid][A[j]]-1
```

Counting-sort (Parallel) - Loop 4



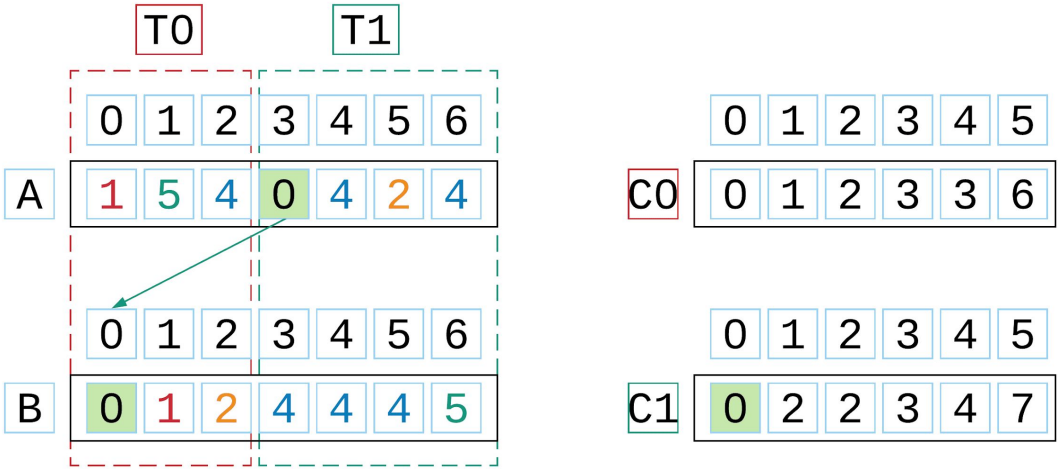

```
Parallel for j ←tid_end downto tid_start
    do B[C[tid][A[j]]-1] ←A[j]
       C[tid][A[j]]←C[tid][A[j]]-1
```

Counting-sort (Parallel) - Loop 4



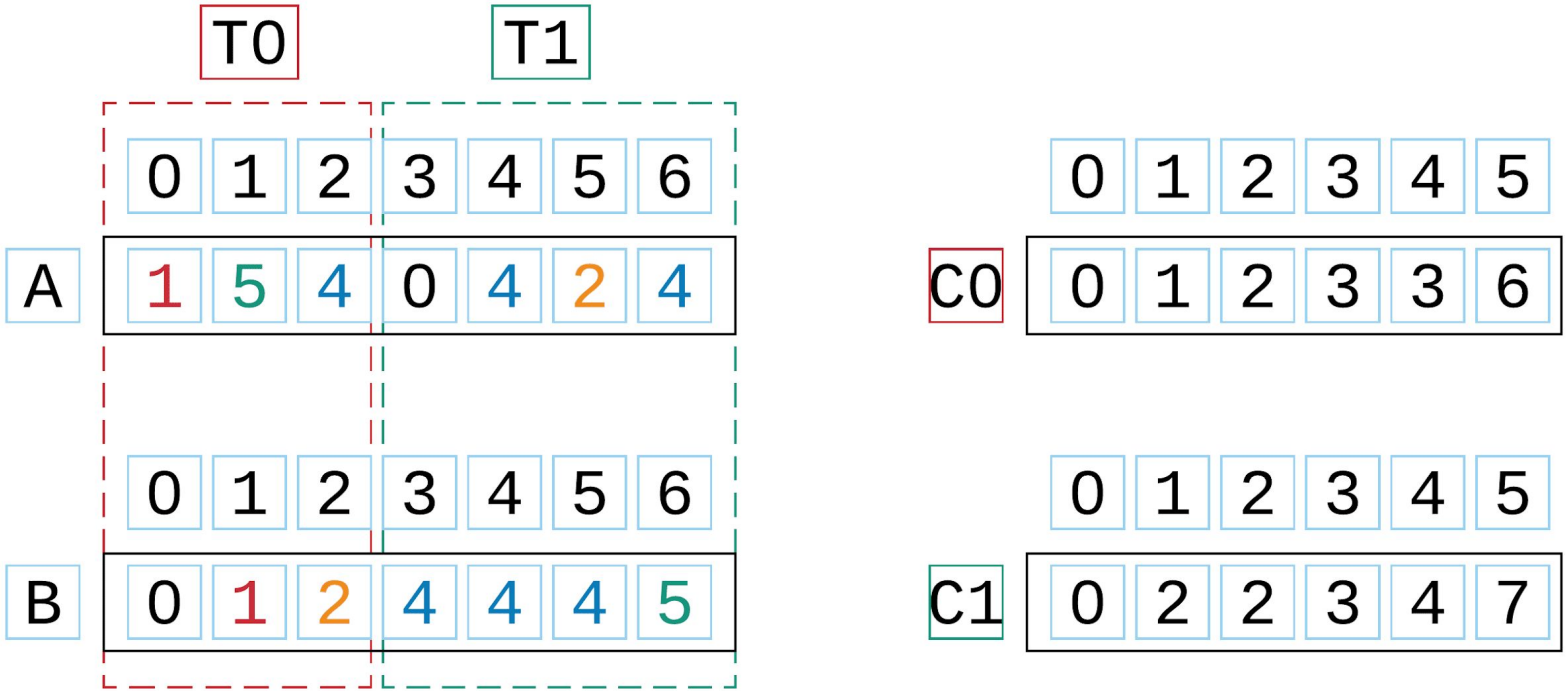

```
Parallel for j ←tid_end downto tid_start
    do B[C[tid][A[j]]-1] ←A[j]
       C[tid][A[j]]←C[tid][A[j]]-1
```

Counting-sort (Parallel) - Loop 4




```
Parallel for j ←tid_end downto tid_start
    do B[C[tid][A[j]]-1] ←A[j]
       C[tid][A[j]]←C[tid][A[j]]-1
```

Counting-sort (Parallel) - 2 Threads



Counting-sort (Parallel) - (t) Threads