2022

Verslag – Bucket Sort

IN DIT DOCUMENT WORD ER BESCHREVEN WAT DE TIJD EN RUIMTE COMPLEXITEIT IS VAN DE BUCKET SORT.

CEYHUN ÇAKIR - 1784480

Inhoud

Inleiding

In deze document ga ik uitleggen wat de tijd en ruimte complexiteit is van de bucket sort algoritme. Ook ga ik uitleggen hoe ik dit heb gedaan en wat de uiteindelijke uitkomsten zijn.

Hoe heb ik het gemaakt?.

Ik ben begonnen met het analyseren van het opdracht. Na dat ik de opdracht had geanalyseerd ben ik begonnen met het opzetten van het environment. Toen ik klaar was met het opzetten van het opdracht ben begonnen met het aanmaken van het bucket sort functie. Voor dit functie hebben we een template aangewezen om vectoren binnen te krijgen met meerdere datatypes. In dit functie gaf ik twee parameters mee, de array die gesorteerd moest worden en de max element binnen die array.

```
template <typename T>
std::vector<T> bucket_sort(std::vector<T> array, int max_elem) {
```

Binnen de functie ben ik begonnen met het schrijven van een for loop die verlaagt vanaf de lengte van de max_elem tot aan 0. Aangezien we vanaf de 100 tallen naar de een tallen willen sorteren was dit noodzakelijk in het functie. Na dat ik de for loop had geschreven had ik ook een vector aangemaakt met buckets. Dit was voornamelijk noodzakelijk voor het sorteren van de getalen.

```
std::vector<std::vector<T>> buckets = {{}, {}, {}, {}, {}, {}, {}, {}, {}, {}};
```

Verder ben ik door gegaan met de onderdeel van distribution pass. In dit onderdeel werd er verwacht dat de meest rechtse cijfer (de een waarde) geplaats werd in een van de correspondente bucket. Ik deed dit door een for loop te schrijven die door de gegeven unsorted array heen loopt. Binnen de loop vereenvoudig ik de absolute waarde van de element tot de macht van 10. Dit doe ik zodat wanneer de index waarde 3 is en de waarde wat gepakt word een index van 2 heeft, dat het nog wel mogelijk is dat die ook in een bucket word gestopt. Uiteindelijk stop ik de element in een bucket door de meest rechtse waarde van het element als index te gebruiken.

```
for(int elements = 0; elements < array.size(); elements++) {
  // creert nullen voor de cijfers zodat wanneer er 3 getallen er zij
  int big_int = std::abs(array[elements]) * std::pow(10, index - 2);

  // we pushbacken de laatse integer als bucket en de element
  buckets[big_int % 10].push_back(array[elements]); //
}
```

Na de onderdeel van het distribution pass verwijder ik alle elementen binnen de array met unsorted elementen. dit doe ik zodat de gesorteerde nummers voor een ronde in een lege array komen. Toen alle elementen verwijdert waren ben ik begonnen met het maken van het gathering pass onderdeel. Hierin werd gevraagt dat je alle elementen binnen de buckets naar de originele array terug zet. Dit heb ik kunnen realiseren door twee for loops te schrijven die nested waren. Een for loop om door de buckets heen te gaan en een for loop om door de elementen binnen de bucket te gaan.

```
for(int bucket = 0; bucket < buckets.size(); bucket++) {
  // we itereren voor elke bucket in buckets om de getallen te krijgen.
  for(int integers = 0; integers < buckets[bucket].size(); integers++) {
    // we pushen uiteindelijk alle nummers die we steeds tegen komen in
    array.push_back(buckets[bucket][integers]);
  }
}
```

Uiteindelijk na de gathering pass onderdeel is de algoritme compleet. Als de waarde 0 is van de eerste for loop dan returnen we de array met de gesorteerde nummers.

Uitkomsten

Hieronder in de afbeelding zien we dat er 10 getallen gegenereerd zijn met een min en max waarde. We zien de ongesorteerde array en de uiteindelijke gesorteerde array.

```
desired amount of integers: 10
desired range of random generated numbers (min/max) -9999/9999
given array - -9958 | 8468 | -3665 | -3497 | 9170 | 5725 | 1479 | -639 | -3035 | -5533 |
sorted array - -9958 | -5533 | -3665 | -3497 | -3035 | -639 | 1479 | 5725 | 8468 | 9170 |
```

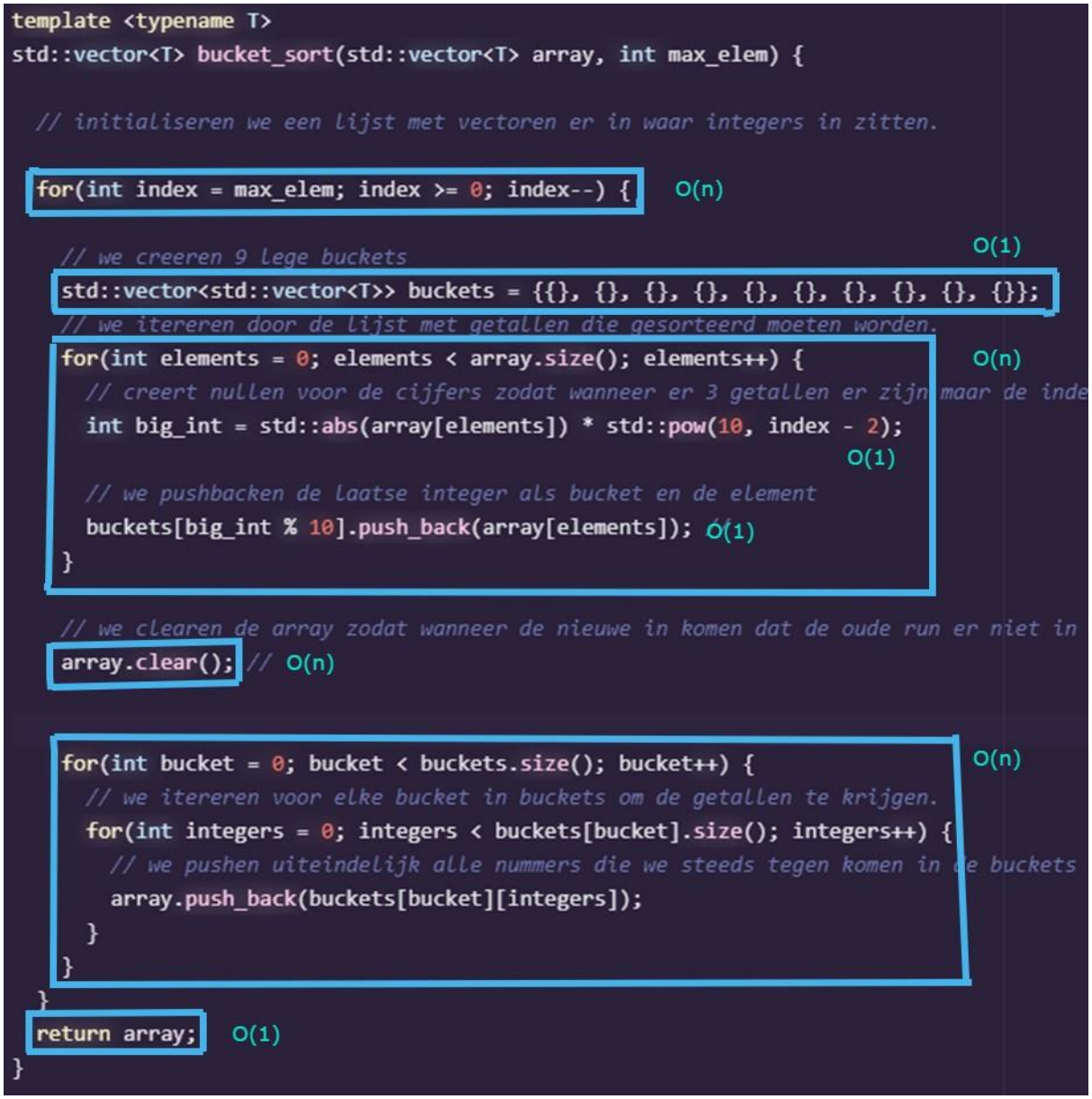
Tijd complexiteit

n + index + index(1 + n(1 + 1) + n) + 1

n + index + index(n + n)

index(n + n)

O(index + n^2)



Ruimte complexiteit