Counting sort (C)

From LiterateProograms

Other implementations: **C** | C# | Haskell | Java | Python, functional | Scala

Counting sort is a very simple sort that is efficient for lists that take on only a small number of values. A linear-time algorithm based on array indexing, it trades time for space.

For the purposes of this article, we'll assume the elements take on integer values, but the same ideas apply to any type. We begin by declaring an array that will hold a counter for each possible value that an element can take on:

```
<<counting_sort.c>>=
#include <stdio.h>   /* printf */
#include <stdlib.h>  /* exit */

/* Sorts a[0..length-1] where each element is between 0 and num_values-1. */
void counting_sort(int a[], unsigned int length, int num_values) {
    counting_sort variable declarations
    int* counts = malloc(sizeof(int) * num_values);
    if (counts == NULL) {
        printf("Out of memory\n");
        exit(-1);
    }
    perform sort
    free(counts);
}
```

Next, we perform the sort. There are two phases to counting sort. In the first phase, *count occurrences*, we iterate through the list, counting the number of times we observe each element by incrementing the array value corresponding to each element:

```
<<counting_sort variable declarations>>=
unsigned int i;
<<count occurrences>>=
for(i=0; i<length; i++) {
    counts[a[i]]++;
}
```

Note the use of a[i] as an array index. This is what makes counting sort not a comparison sort. In the second phase, *generate result*, we iterate through the counts array, outputting the counted number of elements for each value:

```
<<counting_sort variable declarations>>=
unsigned int j, output_pos;
<<generate result>>=
output_pos = 0;
for (i=0; i<num_values; i++) {
    for (j=0; j<counts[i]; j++) {
        a[output_pos] = i;
        output_pos++;
    }
}
```

Although this is a doubly-nested loop, the way in which we set the counters guarantees that the inner loop is executed only a linear ($O(n)$) number of times. Putting together these two phases, the sort is now complete:

<<perform sort>>=
count occurrences
generate result

Download code (http://en.literateprograms.org/index.php?title=Special:Downloadcode/Counting_sort_(C)&oldid=4214)

Retrieved from "http://en.literateprograms.org/index.php?title=Counting_sort_(C)&oldid=4214"
Categories: Programming language:C | Environment:Portable | Counting sort