Using Alltoallv in Bucket Sort

Here is one way to use `Alltoallv()` in bucket sort. Here I have stored all my small buckets in a single array `small_bucket` with the buckets spaced at equal intervals along the array indices. The number items that were placed in the small buckets is stored in the `numpb` array. Each process has its own `small_bucket` and `numpb` with their own data. To empty the small buckets into large buckets, each processor needs to know how much data to receive. This is done by transposing the data in `numpb` into the receive counts `recvcnt` and thus requires an `Alltoall()` before doing the main `Alltoallv()` operation. It is also necessary to set up the send displacements, `sendoff` and the receive displacements `recvoff`.

```
int empty_small(std::vector<float>& small_bucket,
                std::vector<int>& numpb,
                std::vector<float>& big_bucket, int num_data_pp)
{
  int num_buckets = numpb.size();

  std::vector<int> recvcnt(num_buckets);
  MPI::COMM_WORLD.Alltoall(&numpb[0], 1, MPI_INT, &recvcnt[0], 1, MPI_INT);

  std::vector<int> recvoff(num_buckets);
  recvoff[0] = 0;
  int num_recv = recvcnt[0];
  for (int n = 1; n < num_buckets; ++n) {
    recvoff[n] = recvoff[n-1] + recvcnt[n-1];
    num_recv += recvcnt[n];
  }

  std::vector<int> sendoff(num_buckets);
  for (int n = 0; n < num_buckets; ++n) sendoff[n] = n * num_data_pp;

  MPI::COMM_WORLD.Alltoallv(
         &small_bucket[0], &numpb[0], &sendoff[0], MPI_FLOAT,
         &big_bucket[0], &recvcnt[0], &recvoff[0], MPI_FLOAT);

  return num_recv;
}
```

Note here that I am making use of the `vector` data structure of the C++ standard library. These are much more convenient than standard C-style arrays. However, note how `vector`s are passed as arguments in the MPI functions.