Solution

Subtask 1

For every pair of two guests, check whether they wear the same costume. This is possible by inviting those two to a party. This way we need $\frac{N(N-1)}{2} \in O(N^2)$ parties.

Subtask 2

First solution

First, we construct a reference set of guests in which every costume is unique. This can be done by iteratively adding a single guest to this set and host a party. For this we need $N-1$ parties. Every remaining guest can be matched with a person with the same costume from the reference set by using a binary search, which takes at most $\lceil \log_2 N \rceil$ parties. So the total number of parties we need is bounded by $N - 1 + N\lceil \log_2 N \rceil \in O(N \log N)$.

Alternative, less efficient solution

We start by finding all persons wearing the same costume as the first person using a divide and conquer approach. For any group of people we can – by having exactly two parties – check whether there is any person in this wearing the same costume as the first person. Then a simple divide and conquer algorithm divides the sets of persons until it has only one member or does not contain any member wearing the same costume as the given person, thus we have found all people with the same costume as the first person.

Now we repeat this algorithm with the next person which does not wear the same costume as the first person and so on.

Analysis: The divide and conquer algorithm reaches every singleton set x at most once (afterwards the costume of x got recognized) and to reach this set it needs $\mathcal{O}(\log N)$ parties. Thus there are $\mathcal{O}(N \log N)$ parties where we get a positive result (the currently considered costume can be found among the guests). There are also $\mathcal{O}(N \log N)$ other parties, since and every such party the divide and conquer algorithm does no further splitting, thus the number of such parties is bounded from above by N plus the number parties with a positive result. Over all we need $\mathcal{O}(N \log N)$ parties.