Definite d-limited multisets

For a given multiset of natural numbers A, we denote $\sum A = \sum_{a \in A} a$. For example, if $A = \{1, 2, 2, 2, 10, 10\}$, then $\sum A = 27$. For two multisets, we write $A \supseteq B$ if every element in B appears in A at least as many times as in B. For the purpose of this task, we adopt the following definitions.

Definition. A multiset A is called d-limited, for a natural number d, if it is finite and all its elements belong to $\{1, \ldots, d\}$ (with any repetitions).

Definition. A pair of d-limited multisets A, B is called uncontroversial if for all $A' \subseteq A$ and $B' \subseteq B$ it holds that $\sum A' = \sum B' \iff A' = B' = \emptyset \vee (A' = A \wedge B' = B)$. In other words, $\sum A = \sum B$, but the sums of any non-empty subsets of A and B must differ.

Problem. For a fixed $d \geq 3$ (we will not consider smaller d) and multisets A_0, B_0, we want to find uncontroversial d-limited multisets $A \supseteq A_0$ and $B \supseteq B_0$ that maximize the value $\sum A$ (equivalently $\sum B$). We denote this value by $\alpha(d, A_0, B_0)$. Assume $\alpha(d, A_0, B_0) = 0$ if A_0 and B_0 are not d-limited or do not have d-limited uncontroversial supersets.

Example. $\alpha(d, \emptyset, \emptyset) \geq d(d-1)$.

Proof sketch. The sets $A = \{d, \ldots, d\}$ (with $d-1$ repetitions) and $B = \{d-1, \ldots, d-1\}$ (with d repetitions) satisfy the conditions for $\sum A = d(d-1) = \sum B$.

Example. $\alpha(d, \emptyset, \{1\}) \geq (d-1)^2$.

Proof sketch. The sets $A = \{1, d, \ldots, d\}$ (with $d-2$ repetitions) and $B = \{d-1, \ldots, d-1\}$ (with $d-1$ repetitions) satisfy the conditions for $\sum A = 1 + d(d-2) = (d-1)^2 = \sum B$.

It can be proven that the above examples are optimal, i.e., $\alpha(d, \emptyset, \emptyset) = d(d-1)$ and $\alpha(d, \emptyset, \{1\}) = (d-1)^2$.

Nevertheless, in this task, we will want to verify this computationally for the largest possible d, as well as calculate the values of α for other forced multisets A_0, B_0.

Backtracking recursion

We can calculate the values $\alpha(d, A_0, B_0)$ recursively by incrementally building the multisets $A \supseteq A_0$ and $B \supseteq B_0$. Let $A^{\Sigma} = \{\sum A' : A' \subseteq A\}$, which is the set of all possible sums that can be obtained from the set A (not a multiset, i.e., we are not interested in how many ways a given sum can be obtained from the elements of one multiset). We use the following recursion.

Algorithm 1: Backtracking recursion

```
Function Solve(d, A, B):
    if ∑A > ∑B then
        swap(A, B);
    S ← A^Σ ∩ B^Σ;
    if ∑A = ∑B then
        if S = {0, ∑A} then
            return ∑A;
        else
            return 0;
    else if S = {0} then
        return max_{x ∈ {lastA,...,d} \ B_Σ} Solve(d, A ∪ {x}, B);
    else
        return 0;
```

where 'lastA' denotes the element last added to A; in the case of $A = A_0$, we assume 1 (i.e., the recursion adds elements to A_0 in non-decreasing order).

In practice, to avoid recalculating the sets of sums A^Σ and B^Σ each time, we pass A^Σ and B^Σ. When we add an element x to A, the new A^Σ is $A^\Sigma \cup (A^\Sigma + x)$, where $A^\Sigma + x$ is the set obtained from A^Σ by increasing each element by x. The sets of sums A^Σ and B^Σ are efficiently represented using so-called bitsets.