$$\text{polytype P1} = \{\vec{x} | A_0 \vec{x} \preccurlyeq \overrightarrow{b_0}\}$$
$$\text{polytype P2} = \{\vec{x} | A_1 \vec{x} \preccurlyeq \overrightarrow{b_1}\}$$

Let's say $A_0$ has $n$ rows. The question is basically asking how to determine whether polytype P1 is inside polytype P2. For polytype P1 to be inside polytype P2, each of polytype P1's faces have to be inside polytype P2. This can be phrased as a series of linear programming problems:

$$\forall i \in [n],$$
$$\text{minimize } A_0^i \vec{x} - \overrightarrow{b_0^i}$$
$$\text{subject to } A_1 \vec{x} \preccurlyeq \overrightarrow{b_1}$$

Where $\overrightarrow{b^i}$ is the $i$-th element of $\vec{b}$ and $A^i$ is the $i$-th row of matrix $A$.

If there is any negative objective, then that means a polytype P1 face is outside of polytope P2 and output NO; otherwise output YES.

In the other direction, if P1 is inside P2, then our algorithm will output YES because then that means either $P2 \setminus P1$ is non-empty and is thus union of polytopes ($A_0^i \vec{x} - \overrightarrow{b_0^i} > 0$), or is empty ($A_0^i \vec{x} - \overrightarrow{b_0^i} = 0$), which means $P1 = P2$.

The runtime of this is at most $O\big(n * t(N)\big)$.