# Domain Sorting

Damien Beecroft

June 28, 2023

Let $\mathbb{I} \equiv \{i : i \in \mathbb{N}, 1 \leq i \leq N\}$. Suppose that we have a domain $\Omega$. We are given a set of potentially overlapping sub-domains, $\{\Omega_i\}_{i=1}^N$, whose union is $\Omega$. In other words, $\cup_{i=1}^N \Omega_i = \Omega$. We are also given a set of elements of $\Omega$: $X$. We desire an efficient algorithm that tells us for each $x \in X$ which of the sets $\{\Omega_i\}_{i=1}^N$ have $x$ as an element. This is a sorting problem and can be solved efficiently in $\mathcal{O}(N \log(N))$ using a divide and conquer algorithm. In order to implement this algorithm, we must define a hierarchy of subsets of $\Omega$. Suppose this hierarchy consists of $L$ layers. Define the set of subsets on each layer, $l$, as $\{\Omega_i^{(l)}\}_{i=1}^{N^{(l)}}$. We have

$$\Omega_k^{(l)} = \bigcup_{i \in P_k^{(l)}} \Omega_i$$

where $\{P_i^{(l)}\}_{i=1}^{N^{(l)}}$ are disjoint, non-empty subsets of $\mathbb{I}$ whose union is $\mathbb{I}$. We define $\Omega_k^{(L)} \equiv \Omega_k$. This sequence of sets is related in the following manner.