NC Algorithm

Step 1: Core neighbors generation

Initialize **CN**, **FN**, **indirect** set

```
Initialize top_k_distance = (0.25 % len($D$) > 50) ? (0.25% of len(D) : 50
for $i$ in len(D):
    $H_i$ = list of Euclidean distances between point i and the remaining points in D
    increasing_sort($H_i$)
    $H_i$ = $H_i$ [:top_k_distance]
    for $j$ in len($H_i$):
        if density($D_i$, $H_i[j]$) = 0:
            $CN_i$.add($H_i[j]$)
        else:
            $indirect_i$ = $j$
            break
    end for
Set FN = CN
end for
```

Step 2: Density connected neighbors

```
Initialize DN = FN
Initialize breakpoint set
for $i$ in len(D):
    $j$ = $indirect_i$
    do:
        $FN_i$.add($H_i$)
        $j$+=1
    while density($D_i$, $H_i[j]$ < density ($D_i$, $H_i$)[$j$-1])
    $breakpoint_i$ = $j$
Set FN = DN
end for
```

Step 3: Extended neighbors development

```
Initialize **EN** = **FN**
**for** $i$ in len(D):
    $j$ = breakpoint$_i$
    **do**:
        **if** density(D$_i$, H$_i$[j]) > density(D$_i$, H$_i$[$j$-1]) or \\
        is_intersect(DN$_i$, DN$_{\text{index\_of H}_i\text{[j] in D}}$):
            EN$_i$.add(H$_i$[j])
        **else:**
            break
    **while** ($j$ < top_k_distance)
**end for**
Set FN = EN
```

Step 4: Final neighbors construction

```
Initialize flag set
**do**:
    **for** $i$ in len(D):
        flag$_i$ = 0
        j = 1
        **do**:
            **if** D$_i$ $\in$ FN$_{\text{index of FN}_i\text{[j] in D:}}$
                continue
            **else if** FN$_i$[$j$] $\in$ CN$_i$:
                **if** !is_intersect(CN$_i$, FN$_{\text{index of FN}_i[j]\text{ in D}}$):
                    FN$_i$ = FN$_i$[0:$j$-1]
                    flag$_i$ = 1
        **while**(flag$_i$ $\neq$ 1 and $j$ < len(FN$_i$)
    **end for**
**while** $\sum flag \neq 0$
```