Algorithm 1 The algorithm layout

```
$data \leftarrow$ vector of $N$ data points
$C_k(i) \leftarrow$ define connectivity function for data
        with $k = 1, 2..., p$ where $p$ is the amount of connections.
$[W_G, P_G] =$ CONSTRUCT GRAPH($data, C_k(i)$)
$FeatureTable =$ FEATURE TABLE($W_G$)
$P_G =$ DISCRETE PULSE TRANSFORM($W_G, P_G, FeatureTable$)
```

Algorithm 2 Construct Graph

```
procedure CONSTRUCT GRAPH($data, C_k(i)$)
    Create the work graph $W_G = (V_W, E_W)$
    Create the pulse graph $P_G = (V_P, E_P)$
    Create node $V_{W,0}$ in work graph $W_G$        ▷ This is the Zero node for boundary conditions.
            $size \leftarrow \infty$
            $height \leftarrow$value of data point
    for every $i$ in $data$ do
        Create node $V_{W,i}$ in work graph $W_G$
            $size \leftarrow 1$                      ▷ Nodes have multiple properties
            $height \leftarrow$value of data point
        Create node $V_{P,i}$ in pulse graph $P_G$
            $size \leftarrow 0$                      ▷ Show it present position data
            $height \leftarrow i$                    ▷ $i$ denotes index in data
        Create VirtualEdge $(V_{W,i}, V_{P,i})$
    end for
    SET CONNECTIVITY($W_G, C_k(i)$)
end procedure
```

Algorithm 3 Set Connectivity

```
procedure SET CONNECTIVITY($W_G, C_k(i)$)
    for every node $V_{W,i}$ in work graph $W_G$ do
        for every $k$ in $C_k(i)$ do
            $j_k \leftarrow C_k(i)$          ▷ Calculate relative index for node connection
            if $j$ out of bounds then
                $j_k \leftarrow 0$                              ▷ Connect to *Zero* node
            end if
            Create edge $(V_{W,i}, V_{W,j_k})$ in $W_G$
        end for
    end for
end procedure
```

Algorithm 4 Feature Table

```
procedure FEATURE TABLE($W_G$)
    for every node $V_{W,i}$ in work graph $W_G$ do
        if *height* of $V_{W,i}$ = any *height* of neighbor $V_{W,j}$ then
            *size* of $V_{W,j} \leftarrow$ *size* of $V_{W,j}$ + *size* of $V_{W,i}$
            Neighbor $V_{W,j}$ inherit all neighbors of $V_{W,i}$
            Neighbor $V_{W,j}$ inherit all *VirtualEdges* connected to $V_{W,i}$
            Delete $V_{W,i}$ from work graph $W_G$
        else if (*height* of $V_{W,i}$ > *height* of all neighbors)
            OR (*height* of $V_{W,i}$ < *height* of all neighbors) then
                Add node $V_{W,i}$ to *FeatureTable*
        end if
    end for
end procedure
```

Algorithm 5 Discrete Pulse Transform

```
procedure Discrete Pulse Transform($W_G, P_G$, FeatureTable)
    scale ← 1
    CNode ← first node $V_{W,i}$ in FeatureTable
    while FeatureTable ≠ empty do
        if CNode size = scale then
            if Check Feature($W_G$, FeatureTable, CNode) then
                NodeIsPulse ← false
                if CNode is a min feature then                ▷ Apply $L_n$ first
                    if max feature with size = scale ∄ in FeatureTable then
                        NodeIsPulse ← true
                    end if
                else if CNode is a max feature then
                    NodeIsPulse ← true
                end if
            end if
        end if
        if NodeIsPulse = true then
            Add Pulse($W_G, P_G$, FeatureTable, CNode)
        end if
        if FeatureTable contains no features which size = scale then
            Increase scale
        end if
        CNode ← next node in FeatureTable
    end while
end procedure
```

Algorithm 6 Add Pulse

```
procedure Add Pulse($W_G, P_G$, FeatureTable, CNode)
    $i, j, k$ ← arbitrary node indexes
    $V_{W,j}$ ← neighbor of CNode with nearest height
    Create node $V_{P,k}$ in work graph $P_G$
        size ← size of $V_{W,j}$
        height ← Cnode height minus $V_{W,j}$ height
    for every VirtualEdge (CNode, $V_{P,p}$) connected to CNode do
        Create directional edge ($V_{P,p}, V_{P,k}$)
        Delete VirtualEdge (CNode, $V_{P,p}$)
    end for
    Add VirtualEdge ($V_{W,j}, V_{P,k}$)
    Delete CNode from FeatureTable
    Delete CNode from work graph $W_G$
    Add $V_{W,j}$ to FeatureTable
end procedure
```

Algorithm 7 Check Feature

```
function CHECK FEATURE($W_G$,$FeatureTable, CNode$)
    if $height$ of $CNode$ = any $height$ of neighbor $V_{W,j}$ then
        $size$ of $V_{W,j} \leftarrow size$ of $V_{W,j}$ + $size$ of $CNode$
        Neighbor $V_{W,j}$ inherit all neighbors of $CNode$
        Neighbor $V_{W,j}$ inherit all $VirtualEdges$ connected to $CNode$
        Delete $CNode$ from $FeatureTable$
        Delete $CNode$ from work graph $W_G$
        Add $V_{W,j}$ to $FeatureTable$
        return $false$
    else if $height$ of $CNode$ > $height$ of all neighbors then
        $CNode$ is max feature
        return $true$
    else if $height$ of $CNode$ < $height$ of all neighbors then
        $CNode$ is min feature
        return $true$
    else
        Delete $CNode$ from $FeatureTable$
        return $false$
    end if
end function
```