SPIHT Algorithm

\( \mathcal{O}(i,j) \): set of coordinates of all offspring of node \((i,j)\); children only
\( \mathcal{D}(i,j) \): set of coordinates of all descendants of node \((i,j)\); children, grandchildren, great-grand, etc.
\( \mathcal{H}(i,j) \): set of all tree roots (nodes in the highest pyramid level); parents
\( \mathcal{L}(i,j) \): \( \mathcal{D}(i,j) - \mathcal{O}(i,j) \) (all descendents except the offspring); grandchildren, great-grand, etc.

**Initialization**

\[ n = \left\lfloor \log_2 (\max |\text{coeff}|) \right\rfloor \]
\( LIP = \) All elements in \( \mathcal{H} \)
\( LSP = \) Empty
\( LIS = \mathcal{D}'s \) of Roots

**Significance Map Encoding (“Sorting Pass”)**

**Process LIP**

for each coeff \((i,j)\) in LIP

Output \( S_n(i,j) \)

If \( S_n(i,j) = 1 \)

Output sign of coeff\((i,j)\): 0/1 = +/-

Move \((i,j)\) to the LSP

Endif

End loop over LIP

\[ S_n(\Gamma) = \begin{cases} 
1, & \max_{(i,j) \in \Gamma} \left\{ c_{ij} \right\} \geq 2^n \\
0, & \text{otherwise}
\end{cases} \]
**SPIHT Algorithm (cont.)**

### Process LIS

for each set (i,j) in LIS
  - if type \( D \)
    - Send \( S_n(D(i,j)) \)
    - If \( S_n(D(i,j)) = 1 \)
      - for each \((k,l) \in O(i,j)\)
        - output \( S_n(k,l) \)
        - if \( S_n(k,l) = 1 \), then add \((k,l)\) to the LSP and output sign of coeff: 0/1 = -/+ 
        - if \( S_n(k,l) = 0 \), then add \((k,l)\) to the end of the LIP 
      - endif 
    - endif 
  - endif 
else (type \( L \))
  - Send \( S_n(L(i,j)) \)
  - If \( S_n(L(i,j)) = 1 \)
    - add each \((k,l) \in O(i,j)\) to the end of the LIS as an entry of type \( D \)
    - remove \((i,j)\) from the LIS 
  - endif on type 
End loop over LIS

### Refinement Pass

#### Process LSP

for each element (i,j) in LSP – except those just added above 
  - Output the nth most significant bit of coeff 
End loop over LSP

### Update

- Decrement n by 1
- Go to Significance Map Encoding Step

### Adaptive Arithmetic Code (Optional)
SPIHT Sorting Pass

Initialize: All Roots

\[ \text{LIP} \ (i,j) \]

If Significant

Scan List

Insignificant

Significant

Single-Element Subsets

\[ O \text{'s stripped from } D \text{'s} \]

Initialize: Empty

\[ \text{LSP} \ (l,k) \]

Initialize: \( D \)'s of Roots

\[ \text{LIS} \ D(m,n) \]

or

\[ \text{L}(m,n) \]

Set Partitioning Operation

Multiple-Element Subsets

Remainder from \( D \)'s, as \( L \)'s

or

\( O \)'s (as \( D \)'s)stripped from \( L \)'s
SPIHT Refinement Pass

Initialize: Empty

LSP

(1,k)

Scan List

Additions From Last Sorting Pass (NOT scanned)

Output nth MSB of each $|c_{i,k}|$
### Example of SPIHT

From Kahlid Sayood’s Book

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>6</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>-7</td>
<td>7</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>-4</td>
<td>4</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>-2</td>
<td>-2</td>
<td>0</td>
</tr>
</tbody>
</table>

### Initialization

**LIP**

- (0,0) $\rightarrow$ 26
- (0,1) $\rightarrow$ 6
- (1,0) $\rightarrow$ -7
- (1,1) $\rightarrow$ 7

**LSP**

Empty

**LIS**

- (0,1) $D$ $\rightarrow$ $\{13, 10, 6, 4\}$
- (1,0) $D$ $\rightarrow$ $\{4, -4, 2, -2\}$
- (1,1) $D$ $\rightarrow$ $\{4, -3, -2, 0\}$

\[ n = \lfloor \log_2 (26) \rfloor = 4 \]
Threshold = 16

After First Sorting Pass

LIP
(0,1) → 6
(1,0) → -7
(1,1) → 7

LSP
Empty

LIS
(0,1)D → {13, 10, 6, 4}
(1,0)D → {4, -4, 2, -2}
(1,1)D → {4, -3, -2, 0}

(0,0) → 26

No Refinement Needed

After First Refinement Pass

LIP
(0,1) → 6
(1,0) → -7
(1,1) → 7

LSP
(0,0) → 26

LIS

(0,1)D → {13, 10, 6, 4}
(1,0)D → {4, -4, 2, -2}
(1,1)D → {4, -3, -2, 0}

1 1 0 0 0
Sig./+ Insig.

0 0 0
All D sets Insig.

Significant

{13, 10, 6, 4}
During Second Sorting Pass

LIP
(0,1) → 6
(1,0) → -7
(1,1) → 7
(1,2) → 6
(1,3) → 4

LSP
(0,0) → 26
(0,2) → 13
(0,3) → 10

LIS
(0,1)D → {13, 10, 6, 4}
(1,0)D → {4, -4, 2, -2}
(1,1)D → {4, -3, -2, 0}

Significant Pixels

Significant Set
{13, 10, 6, 4}

Insignificant Pixels

After Second Sorting Pass

LIP
(0,1) → 6
(1,0) → -7
(1,1) → 7
(1,2) → 6
(1,3) → 4

LSP
(0,0) → 26
(0,2) → 13
(0,3) → 10

LIS
(1,0)D → {4, -4, 2, -2}
(1,1)D → {4, -3, -2, 0}

Refine This

nth MSB of 26 = 1

n = 3; Threshold = 8

0 0 0 0
Insig.

1 1 1 1 0 0

nth MSB of 26 = 1

+26_{10} = 111010_{2}
↑ sign bit
During Third Sorting Pass

**LIP**
- (0,1) → 6
- (1,0) → -7
- (1,1) → 7
- (1,2) → 6
- (1,3) → 4
- (3,0) → 2
- (3,1) → -2
- (2,3) → -3
- (3,2) → -2
- (3,3) → 0

**LSP**
- (0,0) → 26
- (0,2) → 13
- (0,3) → 10
- (0,1) → 6
- (1,0) → -7
- (1,1) → 7
- (1,2) → 6
- (1,3) → 4
- (2,0) → 4
- (2,1) → -4
- (2,2) → 4

**LIS**
- (1,0) → {4, -4, 2, -2}
- (1,1) → {4, -3, -2, 0}

Threshold = 4

Both Sets Significant

Significant
### After Third Sorting Pass

<table>
<thead>
<tr>
<th>LIP</th>
<th>LSP</th>
<th>LIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>(3,0) → 2</td>
<td>(0,0) → 26</td>
<td>Empty</td>
</tr>
<tr>
<td>(3,1) → -2</td>
<td>(0,2) → 13</td>
<td></td>
</tr>
<tr>
<td>(2,3) → -3</td>
<td>(0,3) → 10</td>
<td></td>
</tr>
<tr>
<td>(3,2) → -2</td>
<td>(0,1) → 6</td>
<td></td>
</tr>
<tr>
<td>(3,3) → 0</td>
<td>(1,0) → -7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1,1) → 7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1,2) → 6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1,3) → 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2,0) → 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2,1) → -4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2,2) → 4</td>
<td></td>
</tr>
</tbody>
</table>