Stochastic U-Curve Branch and Bound

Student: Gustavo Estrela de Matos

Advisor: Dr. Ulisses Braga-Neto

1 Input Generation

To generate the input for the problem we created two functions. The first creates a vector of floating points that simulates the values of a chain of a boolean lattice that respects the U-Curve assumption; and the second one adds a random noise to the values of the vector.

1.1 Chain Generation

The algorithm receives three parameters: n, $max\_distance$ and $center$; and returns as output a vector that has values from 0 to 1 and respects the U-Curve assumption. The first parameter defines the size of the chain; the second represents the greatest possible difference between the values of neighbour nodes, which is a random value uniformily distributed between 0 and $max\_distance$; and the last represents the index of the node with minimum value.

1.2 Noise

The noise is applied to the vector created by $GeneratePoints$

Algorithm 1 U-Curve Input Creator

procedure GeneratePoints($n, max\_distance, center$)
  $points \leftarrow \{0, ..., 0\}$
  $minimum \leftarrow \frac{random()}{2}$
  $points[center] \leftarrow minimum$

  for $i \in \{0, ..., center - 1\}$ **do**
    $points[i] \leftarrow points[i + 1] + (1 - points[i + 1]) * random()$
  end for
  for $i \in \{center + 1, ..., n - 1\}$ **do**
    $points[i] \leftarrow points[i - 1] + (1 - points[i - 1]) * random()$
  end for

  $j \leftarrow n * random()$
  $plain_size \leftarrow (n - j) * random()$
  for $k \in \{1, ..., plain_size\}$ **do** ▷ Creates a plain area in the chain
    $points[j + k] \leftarrow points[j]$
  end forreturn $points$
end procedure