Tarea 6
Ramificaciones y búsquedas para optimización

Pamela Jocelyn Palomo Martínez

29 de mayo de 2017

Este trabajo consiste en resolver el siguiente problema: Dado un punto q y un conjunto de puntos P en un plano bi-dimensional, encontrar el punto $p^* \in P$ tal que la distancia euclidiana de q a p^* es menor que la distancia de q a cualquier otro punto de P. El algoritmo propuesto para resolver dicho problema es el siguiente:

Algorithm 1 PuntoMasCercano(q, P)

Require:
$q = (x_q, y_q)$ ▷ Un punto en un plano bidimensional
P ▷ Un conjunto de puntos en un plano bidimensional

if $|P| = 1$ **then**
  return $p^* : p^* \in P$
end if
Seleccionar un punto $p_r \in P$ de manera aleatoria
Calcular la distancia euclidiana $d(p_R, q)$ de p_r a q
$x_{min} \leftarrow x_q - d(p_R, q)$
$x_{max} \leftarrow x_q + d(p_R, q)$
$y_{min} \leftarrow x_q - d(p_R, q)$
$y_{max} \leftarrow x_q + d(p_R, q)$
Generar un conjunto P' de la siguiente manera: $P' = \{p \in P : x_{min} \leq x_p \leq x_{max} \wedge y_{min} \leq y_p \leq y_{max}\}$
if $|P| = |P'|$ **then**
  $min \leftarrow +\infty$
  for $p \in P$ **do**
    if $d(p, q) < min$ **then**
      $min \leftarrow d(p, q)$
      $p^* \leftarrow p$
    end if
  end for
  return p^*
end if
return PuntoMasCercano(q, P')

El algoritmo consiste en ir tomando un punto aleatorio, calcular su distancia a q y ejecutar el algoritmo recursivamente tomando solamente a los puntos que se encuentren en el cuadrado con centro en q y lados iguales a dos veces la distancia calculada. En el mejor escenario, se llegará a un punto en el cual solo se tenga un punto dentro del cuadrado, entonces ese será el punto más cercano. Sin embargo, hay algunos casos en los cuales existen varios puntos dentro del cuadrado que no equidistan a q, tales que no hay manera de seleccionar alguno de manera que el nuevo cuadrado deje fuera a al menos uno de los otros puntos. Por ejemplo, considere el caso en que $q = (0, 0)$ y restan por explorar los puntos $p_1 = (1, 0)$ y $p_2 = (1, 9/10)$, la elección de cualquiera de ellos no elimina al otro en el nuevo cuadrado y, claramente, p_1 es más cercano a q.

Esto ocurre debido a que la estrategia óptima para determinar cuales son los puntos más cercanos a q que un punto p, es verificar cuáles puntos quedan dentro de la circunferencia con centro en q y radio $d(p, q)$; sin embargo, en el algoritmo no se hace esto porque para determinar qué puntos quedan dentro de dicha circunferencia, es necesario calcular la distancia de los puntos a q, hecho que se quiere evitar. Por ello, se eligió aproximar la circunferencia con un cuadrado en el cual esta se encuentra circunscrita, ya que es fácil determinar qué puntos quedan fuera del cuadrado sin calcular distancias. No obstante, hay áreas del cuadrado que no pertenecen al círculo, causando que el algoritmo ya no pueda eliminar más puntos. Esta es la explicación de por qué se tomaron en cuenta las líneas 11 a 20 del algoritmo.

La Figura **??** muestra el número promedio de veces que se calculó la distancia entre dos puntos ejecutando el algoritmo propuesto 10 veces por cada diferente tamaño de P y el número de veces que se tendría que calcular la distancia entre dos puntoos si se hace por fuerza bruta. El tamaño de P se varió entre 100, 200, 400, 800, 1600, 3200 y 6400.

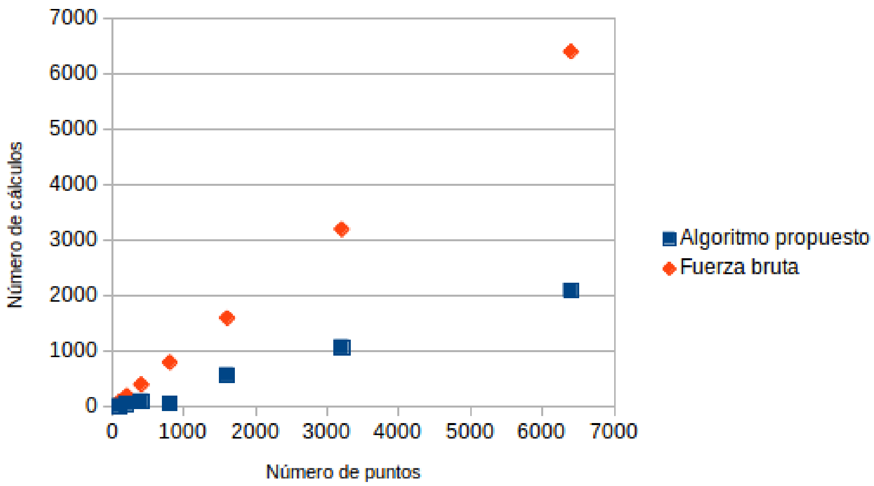

Figura 1: Número de cálculos de distancia entre puntos ejecutados por el algoritmo propuesto contra procedimiento por fuerza bruta