Proyecto 2:
Implementación de la Optimización por enjambre de particulas para el Problema de la galería de arte

Carlos Gerardo Acosta Hernández

Heurísticas de optimización combinatoria 2017-2
Facultad de Ciencias UNAM

Como segundo proyecto del seminario, decidí implementar el problema de la galería de arte[1] en su versión con guardias para los vértices de un polígono simple, junto con la heurística de optimización por enjambre de partículas[2] en su versión binaria para la elección de dichos guardias. En las siguientes secciones abordaré algunos conceptos importantes para la implementación que realicé, tanto de la heurística, como del problema de mi elección; asimismo, presentaré algunos detalles sobre los componentes de mi sistema, las entradas y la experimentación a la que lo he sometido hasta el momento. Por último, al final del documento proporciono las instrucciones y herramientas necesarias para construir el sistema y manejarlo con éxito.

1. Preliminares

1.1. Problema de la galería de arte

El *Problema de la galería de arte* -también conocido como problema del museo-, se puede fácilmente explicar como un problema común en la vida real: Para la disposición de guardias de seguridad en una galería de arte, encontrar una configuración en la que toda el área de la galería esté resguardada por un número mínimo de guardias. Este problema vio la luz en el año 1973, cuando el matemático estadounidense Victor Klee, inquirió a la comunidad científica sobre cuántos guardias han de ser necesarios para vigilar un polígono de n vértices. Por el momento diremos que un polígono está vigilado si para todo $p \in P$, el segmento de línea formado por $g \in G$ y p se mantiene *dentro* del polígono, donde P es el conjunto de los puntos en el plano dentro del polígono y G un conjunto de puntos en el plano. En respuesta a la interrogante, Václav Chvátal, propuso propuso y demostró un teorema para este problema en el que se establecía una cota superior de $floor\frac{n}{3}$ del mínimo número de guardias **necesarios** para vigilar un polígono de n vértices. Es decir que no siempre es necesario ese número de guardias pero siempre es suficiente. Sin embargo, esta cota

[1]
[2]

sugiere que las soluciones factibles que permanecen cercanas a la suficiencia, siguen estando muy alejadas de una configuración óptima para los guardias. El problema de hallar tal configuración pertenece a la clase de complejidad *NP-difícil*, incluso para polígonos simples.

Con el fin de definir el problema considerado lo más generalmente posible, es conveniente revisar lo siguiente: Sea P un polígono con n vertices. Sea h el número de "hoyos" en P, para este proyecto he decidido trabajar para $h = 0$, es decir que se trabaja con polígonos simples, ya sean convexos o cóncavos. Para cualesquiera dos puntos $p, q \in V(P)$, el conjunto $V(P)$ de los vértices del póligono, se dice que son "visibles" entre ellos si el segmento de recta pq se encuentra "dentro" del polígono. Para definir esta contención en el polígono he considerado estas dos condiciones:

- Que pq no intersecta ninguna de las aristas del polígono y,
- el punto medio de pq se encuentra dentro del polígono.

Ahora bien, denotemos el conjunto de visibilidad de un punto $p \in V(P)$ como $M(p) = \{q \in V(P) \mid q \text{ es visible a } p\}$, de manera que para un conjunto de puntos $G \subseteq P$, $M(G)$ estará definido como $M(G) = \cup_{r \in G} M(r)$. Determinaremos que un conjunto $G \subset V(P)$, vigila a todo el polígono si $M(G) = V(P)$. El objetivo entonces es encontrar un conjunto G vigilante de P, tal que su cardinalidad sea mínima, aquí nos centraremos cuando menos en hallar una cardinalidad que esté por debajo de la cota establecida, de ser posible.

1.2. Optimización por enjambre de partículas

La optimización por enjambre de partículas es una heurística inspirada en el comportamiento social colectivo de agrupaciones o aglomeraciones de seres vivos -manadas, colonias, enjambres, entre otros. Utilizando una población de individuos, el algoritmo tiene por objetivo que cada individuo represente una solución al problema mediante su posición, codificada en un vector multidimensional del espacio de búsqueda. Además, el individuo debe tener cierta velocidad que le permita actualizar su posición, codificada similarmente en otro vector. La idea general de la búsqueda es que la actualización de la posición de una partícula no sea un mero factor aleatorio, sino que incluya cierta influencia de la mejor solución de toda la población, digamos un individuo destacado que "lidere" al colectivo, y la mejor posición que haya encontrado en iteraciones anteriores que no necesariamente es la misma que mantiene en la más reciente. Por tanto, la *exploración* es regida por el factor de aleatoriedad y la *explotación* mediante las posiciones que influyen en la actualización mencionada. De tal suerte que cada partícula tenga posibilidad de explorar con cierta libertad el espacio de búsqueda pero se mantenga cerca del enjambre, que avanza en conjunto hacia valores óptimos.

La heurística, propuesta por Russel Eberhart y James Kennedy en 1995, fue originalmente concebida para espacios de búsqueda contínuos, sin embargo, se han propuesto versiones discretas del algoritmo desde entonces, entre ellas, su versión binaria que implementé para este proyecto. En esta adecuación, cada entrada en el vector de posición de una partícula es un *bit* y una entrada en el vector de velocidad representa la probabilidad de que la entrada correspondiente en el vector de posición se establezca como 1 o 0.

El algoritmo siguiente describe el funcionamiento general de la heurística, por lo que sólo será necesario hacer un par de anotaciones posteriores.

ALGORITHM 1: Particle Swarm Optimization (PSO)

Input: n-vertices polygon
Output: Best particle in swarm
$t \leftarrow 0$; $E^t \leftarrow randomValidPopulation(s)$;
repeat
  $t \leftarrow t + 1$;
  for *each* p_i^t *in* E^t **do**
    for *each* $d = 0, 1, ..., n-1$ **do**
      $r_p \leftarrow U(0,1)$, $r_g \leftarrow U(0,1)$
      $v_{i,d}^{t+1} \leftarrow \omega v_{i,d}^t + \varphi_p r_p (b_{i,d}^t - p_{i,d}^t) + \varphi_g r_g (E_{g_d}^t - p_{i,d}^t)$
    end
    update particle's position: $p_i^{t+1} \leftarrow p_i^t + v_i^t$
    if $f(p_i^t) < f(b_i^t)$ **then**
      $b_i^{t+1} \leftarrow p_i^t$
    if $f(p_i^t) < f(E_g^t)$ **then**
      $E_g^{t+1} \leftarrow p_i^t$
  end
until $t = maxGen$;
return E_g

Aquí vale la pena señalar que U es un generador de números pseudoaleatorios entre [0,1], b_i es la mejor posición de la i-ésima partícula del enjambre, ω es el peso o inercia que se ejerce sobre la partícula que determina en buena medida su "movimiento", φ_p y φ_g son dos constantes positivas. Estas tres últimas variables, junto con el tamaño s del enjambre o población y el máximo de iteraciones $maxGen$ establecido -que en este caso funge como condición de terminación, pero puede ser sustituido por otro que se considere más conveniente-, son parte de los parámetros que afectan el comportamiento de la heurística y que deben ser explorados durante la experimentaación para mejores (y reproducibles) resultados.

Como ya había mencionado, deben hacerse un par de cambios a la versión original, considerando que por ejemplo, no podemos sumar el vector de posición $p_i \in \{0,1\}^n$ con el vector de velocidad $v_i \in \mathbb{R}^n$ directamente. Es por ello, que tanto la actualización de la velocidad como de la posición de la partícula se ven definidos por estas nuevas ecuaciones:

$$v_{i,d}^{t+1} = \frac{1}{1 + e^{-v_{i,d}^t}} \tag{1}$$

$$p_{i,d}^{t+1} = \begin{cases} 0 & \text{if } r_{i,d} < v_{i,d}^t \\ 1 & \text{en otro caso} \end{cases} \tag{2}$$

2. Especificación

2.1. Diseño

2.2. Implementación

2.3. Experimentacion

3. Manejo del sistema