Práctica 10: Algoritmo genético

Simulación de sistemas

Marco Antonio Guajardo Vigil 2095

09 de abril, 2019

1. Introducción

El problema de la mochila (en inglés: knapsack) es un problema clásico de optimización, particularmente de programación entera, en donde la tarea consiste en seleccionar un subconjunto de objetos de tal forma que (i) no exceda la capacidad de la mochila en términos de la suma de los pesos de los objetos incluidos, y que (ii) el valor total de los objetos incluidos sea el máximo posible [3].

Se hace uso de un algoritmo de tabulación que determina la combinación óptima, aunque, solo sirve para pesos enteros, es de utilidad para esta práctica, donde se prueba la implementación de un algoritmo genético en `R` de manera paralela. Los algoritmos genéticos representan posibles soluciones de un problema en términos de un genoma. Se busca imitar la evolución permitiendo que las soluciones aleatorias de la población inicial puedan crear nuevas soluciones mediante dos mecanismos: la **reproducción** donde dos soluciones x y y intercambian pedazos uno con otro, creando dos nuevas soluciones (hijos) y la **mutación** donde una solución crea una réplica de sí misma, cambiando algunos de sus valores.

2. Objetivo

Se paraleliza el algoritmo genético y se estudia los efectos en su tiempo de ejecución con pruebas estadísticas y visualizaciones, variando el número de objetos en la instancia. Se generan tres instancias con diferentes reglas:

1. El peso y el valor de cada objeto se generan *independientemente* con uns distribución normal.
2. El pero de cada objeto se genera independientemente con una distribución normal y su valor es *correlacionado* con el peso, con un ruido normalmente distribuido de baja magnitud.
3. El peso de cada objeto se genera independientemente con una distribución normal y su valor es *inversamente correlacionado* con el peso, con un ruido normalmente distribuido de baja magnitud.

Se determina para cada uno de los tres casos a partir de qué tamaño de instancia el algoritmo genético es mejor que el algoritmo exacto en términos de valor total obtenido por segundo de ejecución.

2.1. Implementación de R

Para la elaboración de este experimento, se hace uso de un software libre para computación estadística y gráficos llamado R [1], el cual nos permite realizar los cálculos necesarios para dicho experimento. Con él, se pueden controlar los datos estadísticos que se ocupan para dar seguimiento con la práctica, se necesita graficarlos para así poder compararlos mejor, ya que se maneja una cantidad de datos considerable y trabajaremos con ellos en forma estadística.

2.2. Experimentación

Se establecen 50 partículas para el modelo de experimentación dadas en la variable `n`, se agrega masa a las partículas con una distribución normal para poder crear una fuerza gravitacional entre ellas y se normaliza para generar masas con rango $[0, 1]$, para simplificar las normalizaciones requeridas para las otras variables se crea una función para ello, lo cual se observa en el código:

```
n <- 50
p <- data.frame(x = rnorm(n), y = rnorm(n), carga = rnorm(n), masa = rnorm(n))
carga <- FALSE
fg <- FALSE
normalizar <- function(v){
  max <- max(v)
  min <- min(v)
  V <- (v - min) / (max - min)
  return(V)
}
p$x <- normalizar(p$x) # ahora son de 0 a 1
p$y <- normalizar(p$y) # las y tambien
p$masa <- normalizar(p$masa) # la masa es de 0 a 1
cargamax <- max(p$c)
cargamin <- min(p$c)
p$carga <- 2 * (p$carga - cargamin) / (cargamax - cargamin) - 1 # cargas son entre -1 y 1
```

Se crea la fuerza de atracción gravitacional F siguiendo la fórmula (1):

$$F = G \times \frac{m_1 \times m_2}{d^2}, \tag{1}$$

omitiendo su constante universal de gravitación ya que se trabaja con partículas muy pequeñas, de tal modo que esta causa un efecto en las fuerzas f_x y f_y; donde m_1 es la masa de la partícula que se esta comparando con la de otra partícula m_2 y d^2 es la distancia que existe entre ellas al cuadrado. Si la masa m_1 es menor a la masa m_2 esta se vera atraída por la de mayor masa (m_2).Se suman los dos factores que ahora influyen en las fuerzas (atracción de cargas y gravitacional por sus masas) como se muestra en el código:

```r
fuerza <- function(i) {
  xi <- p[i,]$x
  yi <- p[i,]$y
  cargai <- p[i,]$carga
  masai <- p[i,]$masa
  fx <- 0
  fy <- 0
  for (j in 1:n) {
    cargaj <- p[j,]$carga
    masaj <- p[j,]$masa
    dirc <- (-1)^(1 + 1 * (cargai * cargaj < 0))
    dirm <- (-1)^(1 + 1 * (masai < masaj))
    dx <- xi - p[j,]$x
    dy <- yi - p[j,]$y
    factorc <- dirc * abs(cargai - cargaj) / (sqrt(dx^2 + dy^2) + eps)
    factorm <- dirm * (masai * masaj) / (dx^2 + dy^2 + eps)
    fx <- fx - dx * (factorc + factorm)
    fy <- fy - dy * (factorc + factorm)
  }
  return(c(fx, fy))
}
```

Obtenemos la velocidad sumando los Δx y Δy que se proporcionan para cada partícula en el paso del tiempo:

```r
p$v <- foreach(i = 1:n, .combine=c) %dopar% (p[i,]$x + p[i,]$y)
```

2.3. Resultados y conclusiones

Se creo un `gif` donde se muestra el comportamiento de las partículas afectadas por la atracción de cargas y la fuerza gravitacional de atracción que hay entre ellas, el cual se encuentra en el repositorio de esta misma práctica [2].

Como se muestra en la figura **??**, tomando en cuenta que las partículas rojas son las que mayor velocidad tienen y las amarillas la menor, la velocidad de la partícula es mayor conforme su magnitud de carga sea más neutra, entre mayor masa la velocidad se vuelve más lenta ya que la partícula difícilmente se ve atraída por otras, en cambio cuando su masa es menor, estas suelen tener una velocidad más alta.

Referencias

[1] The R Project for Statistical Computing. 2019. URL https://www.r-project.org/.

[2] Guajardo Vigil Marco Antonio. Simulación de sistemas. 2019. URL https://sourceforge.net/projects/simulaciondesistemas/.

[3] Satu Elisa Schaeffer. Práctica 10: Algoritmo genético. 2019. URL https://elisa.dyndns-web.com/teaching/comp/par/p10.html.