Práctica 10: algoritmo genético

1445183

9 de abril de 2019

1. Objetivo

Estudiar los efectos del tiempo de ejecución del código paralelizado proporcionado por la práctica [2] variando el número de objetos en las tres instancias siguientes:

1.- El peso y el valor de cada objeto se generan independientemente con una distribución normal

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

2. Descripción

El código se paraleliza desde el principio, usando tres núcleos de los cuatro posibles

```
library(testit)
suppressMessages(library(doParallel))
cls <- makeCluster(detectCores() - 1)
registerDoParallel(cls)
```

Se paralelizan las funciones de mutación, reproducción, factibilidad y objetivo basado en el código de *Reyna Fernández* [1]

```
mutacion2<- function() {
  if (runif(1) < pm) {
    return(mutacion(p[i,], n))
  }
}

reproduccion2<- function() {
  padres <- sample(1:tam, 2, replace=FALSE)
  hijos_t <- reproduccion(p[padres[1],], p[padres[2],], n)
  return(hijos_t)
}

objetivo2<- function() {
  obj_t <- double()
  obj_t <- c(obj_t, objetivo(p[i,], valores))
  return(obj_t)
}
```

```
18
19 factible2 <- function() {
20   fact_f <- integer()
21   fact_f <- c(fact_f, factible(p[i,], pesos, capacidad))
22   return(fact_f)
23
24 }
```

después se dan las indicaciones que se piden en el *objetivo* en relación *peso* y *valor* y se varía el número de objetos por instancia con valores de `30, 50` y `80`, se toma el tiempo de ejecución usando `system.time`

```
1 n <- seq(30,50,80)
2 for(ins in 1:3) {
3
4   if(ins == 1) {
5     valores <- generador.valores(pesos, 10, 500)
6   } else if(ins == 2) {
7     valores <- generador.valores.correlacionados(pesos,10,500)
8   } else if(ins == 3) {
9     valores <- generador.valores.correlacionados(rev(pesos),10,500)
10  }
```

3. Resultados

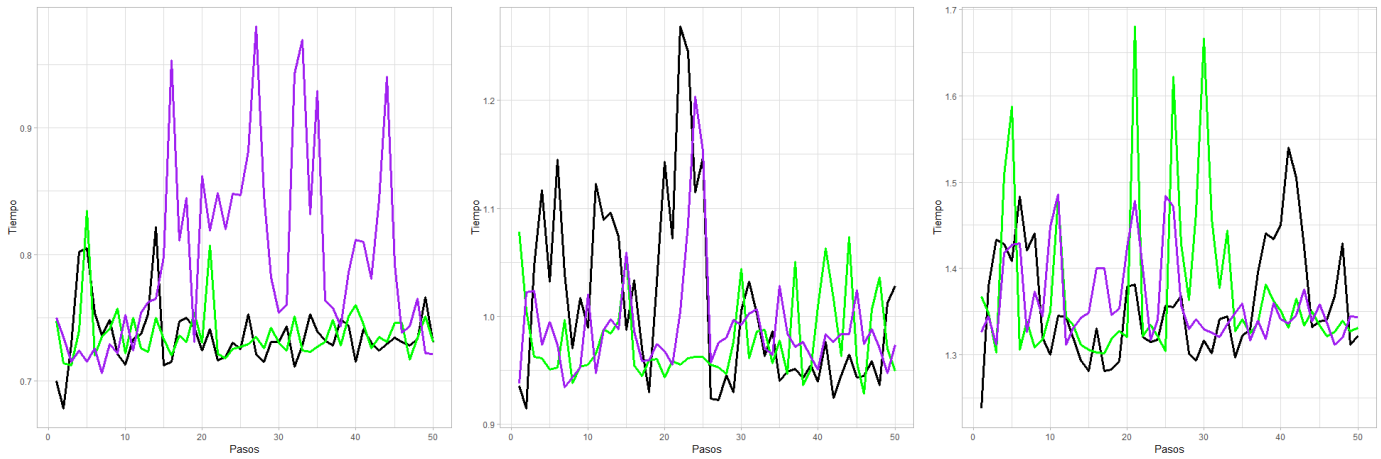
En la figura 1 se puede observar el tiempo que se tarda en dar 50 pasos, los colores corresponden a las instancias indicadas anteriormente.

El tiempo es aumenta cuando se tiene correlación y la cantidad de onjetos aumenta, cuando la correlación es inversa se tarda menos tiempo conforme se tiene más cantidad de objetos y cuando los valores son independientes, es decir, no están correlacionados no se tiene una secuencia fija en el tiempo al cambiar la cantidad de objetos.

Línea negra - sin correlación

Línea verde - correlación

Línea morada - correlación inversa



(a) 30 objetos (b) 50 objetos (c) 80 objetos

Figura 1: Interacción de partículas

4. Conclusión

Es más fácil (rápido) decidir por objetos que valen mucho y pesan poco como pasa en la correlación inversa, donde el tiempo es menor.

Que los objetos pesen más pero tengan poco valor, hace que tarde en decidir si llevarlos o no, como pasa en la correlación.

Referencias

[1] Yessica Reyna. Práctica 10: Algoritmo genético, 2018. URL https://sourceforge.net/projects/simulacion-de-sistemas/.

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