Práctica 11: frentes de Pareto

Gabriela Sánchez Y.

Introducción

La optimización multiobjetivo busca un conjunto de variables que optimizan dos o más funciones objetivo, lo que da lugar a ciertos conflictos ya que puede que una mejora en una corresponda a una empeora en otra, además de que deben respetarse las restricciones del problema. En la presente práctica se estudia este tema.

Se crea un experimento que genera polinomios aleatorios que se utilizan como funciones objetivo. Sólo se permite una variable por término y un término por grado por variable. Se generan soluciones al azar y se calculan los valores de los objetivos para cada solución, suponiendo, por simplicidad, que no hay restricciones. Para cada objetivo se determina al azar si se maximiza o minimiza.

Se usa la dominancia de Pareto: una solución domina a otra si no empeora ninguno de los objetivos y mejora en por lo menos uno, para definir cuales son buenas soluciones. A las soluciones no dominadas se les conoce como un frente de Pareto.

En el archivo `p11or.R` se encuentra el experimento en la versión secuencial.

Tarea

- Paralelizar el cálculo donde convenga y graficar el porcentaje de soluciones de Pareto como función del número de funciones objetivo como diagramas de violín combinados con diagramas de caja-bigote. ¿A qué se debe el comportamiento observado?

Los posibles ciclos candidatos a paralelizar, por el número de operaciones que realizan, son los siguientes:

- Evaluación de las soluciones

```
for (i in 1:n) {
    for (j in 1:k) {
        val[i, j] <- eval(obj[[j]], sol[i,], tc)
    }
}
```

- Dominancia de las soluciones

```
for (i in 1:n) {
    d <- logical()
    for (j in 1:n) {
        d <- c(d, domin.by(sign * val[i,], sign * val[j,], k))
    }
    cuantos <- sum(d)
    dominadores <- c(dominadores, cuantos)
    no.dom <- c(no.dom, cuantos == 0)
}
```

La paralelización se realiza utilizando la funcion *foreach* de la librería *doParallel*.

En el caso de la evaluación de las soluciones se invierten los ciclos y se paraleliza el for que corre sobre n:

```
for (j in 1:k) {
  obj[[j]] <- poli(vc, md, tc)
  val[,j] <- foreach(i = 1:n, .combine=rbind) %dopar%
                                      eval(obj[[j]], runif(vc), tc)
}
```

En el caso de la dominancia de las soluciones se crea la función auxiliar `dom`:

```
dom <- function(i) {
    d <- logical()
    for (j in 1:n) {
        d <- c(d, domin.by(sign * val[i,], sign * val[j,], k))
    }
    cuantos <- sum(d)
    return(cuantos == 0)
}
```

Para verificar si en realidad conviene paralelizar, se miden los tiempos de ejecución en la versión paralela y en la versión secuencial del experimento, considerando variaciones en el número de soluciones, realizando 30 réplicas en cada caso. La figura 1, muestra los resultados obtenidos.

Puede observarse que la versión en paralelo es ligeramente mejor cuando se tienen más soluciones al azar iniciales.

Ahora nos interesa el efecto que tiene el número de funciones objetivo en la variación de las soluciones de Pareto. Se realiza la simulación para distintos valores de k, donde k denota el número de funciones objetivo que se utilizan en el experimento.

Para cada uno de éstos valores se realizaro 30 réplicas, en las que se guardó el porcentaje de soluciones no dominadas. Los resultados son graficados en la figura 2.

El comportamiento que se observa se debe a que mientras más funciones objetivo se tengan, es más complicado de existan soluciones que pudiesen cumplir con todas, es por eso que el porcentaje de soluciones no dominadas aumenta conforme el número de funciones objetivo crece.

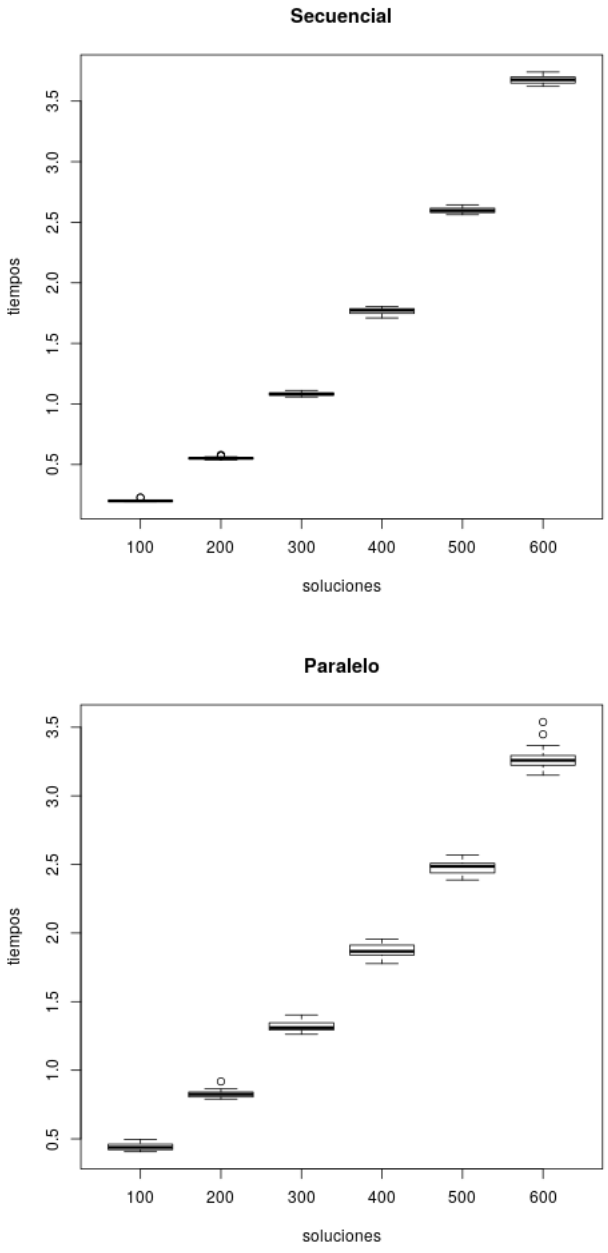

Figura 1: Tiempos de ejecución.

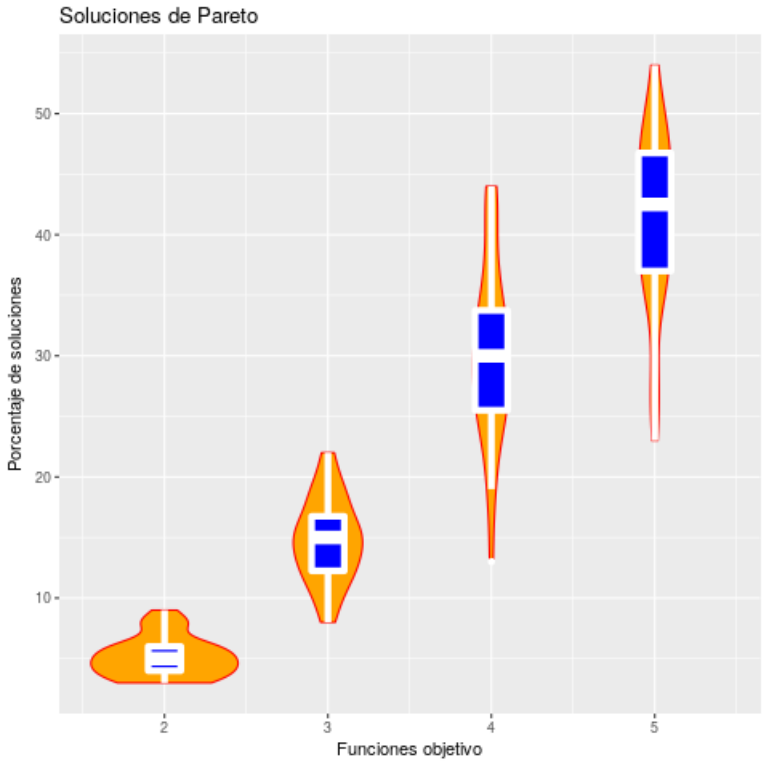

Figura 2: Porcentaje de soluciones no dominadas en función de k.