Nuevas Tecnologías de la Programación
Práctica 3

Javier Moreno

14 de junio de 2015

Resumen

Informe de la práctica 3 de la asignatura Nuevas tecnologías de la programación.

1. Introducción

La práctica pedía la implementación de un método de búsqueda de máximo valor para funciones complejas para las que las técnica analíticas de optimización no funcionan. Para ello se han implementado dos algoritmos de búsqueda: búsqueda aleatoria y recocido simulado.

2. Diseño

2.1. Explicación

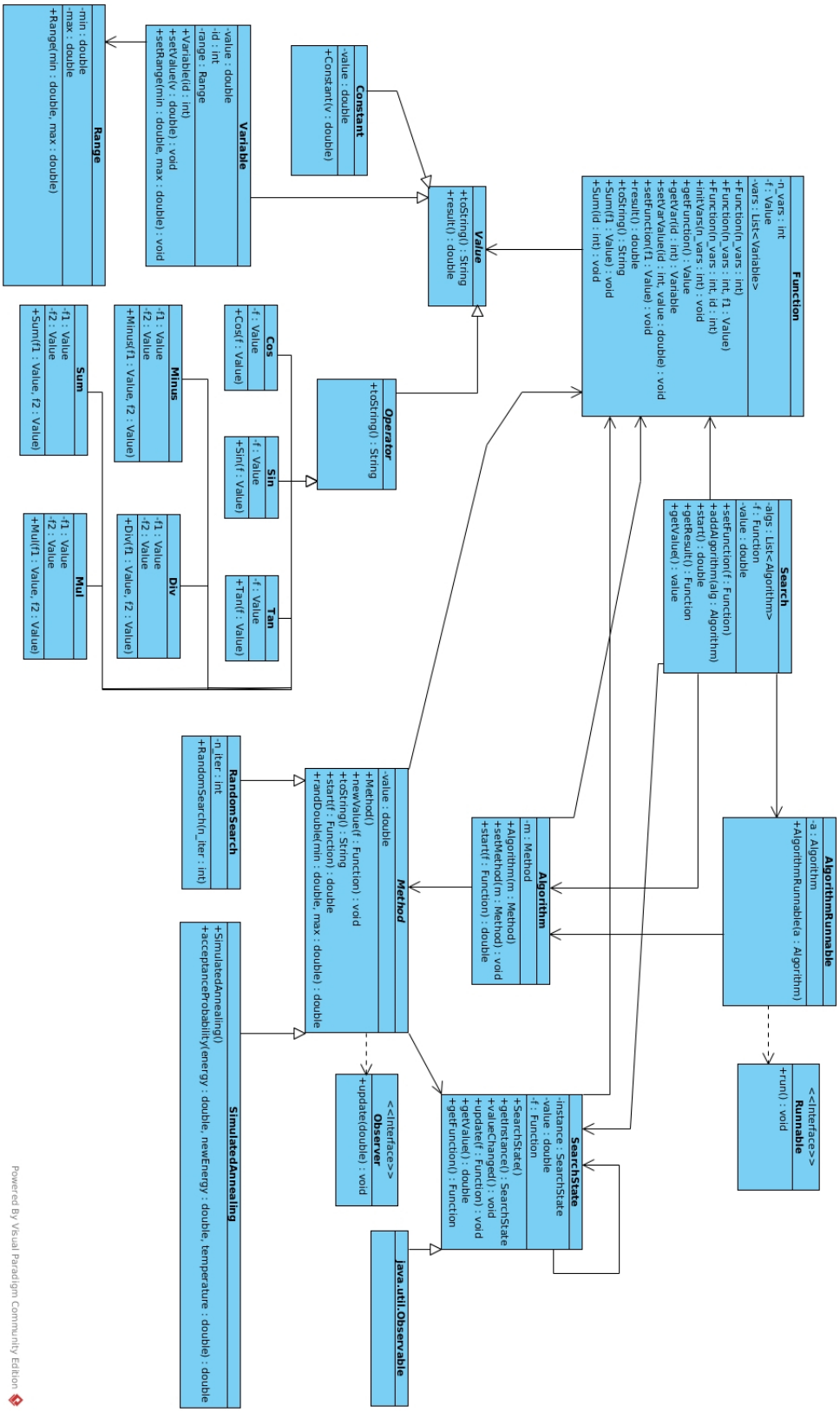
El diseño que se ha seguido incluye la utilización de 3 patrones de diseño: Singleton, Observer y Decorator. Las clases principales son: Function, Search y Algorithm.

Clase Function: Es la clase encargada de crear las funciones y controlar los valores de las variables, usa una clase interior de nombre Value para gestionar los operadores y los resultados de las operaciones, en esta clase uso el patrón Decorator para conseguir devolver el resultado de las operaciones directamente sin tener que modificar en el código las diferentes operaciones.

Clase Algorithm: Es la clase encargada de crear y gestionar los algoritmos de búsqueda, usa la clase Method que es de la que heredan los diferentes algoritmos de búsqueda (métodos), por ejemplo RandomSearch es una clase que hereda de Method. La clase Method además implementa el patrón Observer al usar la interfaz Observer para poder notificar y recibir las actualizaciones que los otros métodos en ejecición hacen en la clase SearchState, la cual hereda de la clase Observable.

Clase Search: Es la clase principal, la cual recibe la función a optimizar y los algoritmos de búsqueda que tiene que usar, devuelve cuando termina el mejor valor encontrado y la función que lo genera. Además incluye la clase AlgorithmRunnable que implementa la interfaz Runnable para poder ejecutar los diferentes algoritmos de búsqueda en forma paralela mediante el servicio ExecutorService.

2.2. Diagrama de clases



3. Ejemplo de uso

```
1  // Objeto f de la clase Function que contendra la funcion a optimizar.
2  Function f = new Function(2);
3
4  // Se definen las operaciones, constantes y variables de la funcion.
5  f.Sin(new Mul(new Mul(new Constant(20), new Constant(3.14)), f.getVar(1)));
6  f.Mul(1);
7  f.Sum(new Mul(f.getVar(0), new Sin(new Mul(new Mul(new Constant(4),
8          new Constant(3.14)), f.getVar(0)))));
9  f.Sum(new Constant(21.5));
10
11 // Rango de las variables.
12 f.getVar(0).setRange(-3, 12.1);
13 f.getVar(1).setRange(4.1, 5.8);
14
15 // Objeto e de la clase Search que controla la busqueda de valores maximos.
16 Search e = new Search();
17 // Se anade la funcion a optimizar.
18 e.setFunction(f);
19 // Se anaden los algoritmos de busqueda que se quieran usar.
20 e.addAlgorithm(new Algorithm(new RandomSearch(10000)));
21 e.addAlgorithm(new Algorithm(new SimulatedAnnealing()));
22 // Inicio de la busqueda.
23 e.start();
24
25 // Se escribe por pantalla la funcion a optimizar.
26 System.out.println("f()="+f.toString());
27 // Se escribe por pantalla el mejor valor encontrado.
28 System.out.println("Best Value: "+e.getValue());
```