Complejidad ciclomática

Ciencias de la computación

Vicente Ferrari
Dpto. de Cs. de la computación e Informática
Universidad de la Frontera
Temuco, Chile
v.ferrari01@ufromail.cl

***Resumen*—En este texto se explorará la complejidad ciclomática de un programa de reducción de laberintos.**

I. Introducción

Los algoritmos se pueden visualizar como grafos de control de flujo. Como se observa en la Figura 1. Esto da paso a una métrica en informática conocida como complejidad ciclomática.

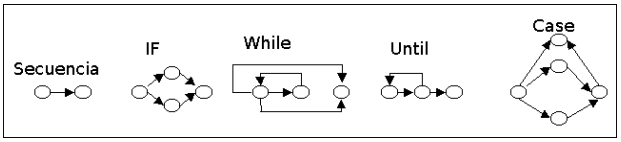

Figura 1. Grafos para las distintas estructuras de control.

II. Complejidad ciclomática

Una vez representado un algoritmo con grafos se puede hacer la siguiente medición:

$$M = E - N + 2P$$

Donde M es la complejidad. E el numero de lados del grafo, N el numero de nodos y P el numero de componentes conexas.

Luego, uno puede clasificar las distintas posibilidades de complejidad en las siguientes categorías:

1-5:	Facil de mantener
6-10:	Dificil de mantener
11-15:	Muy dificil
20+:	Casi imposible

III. Metodología

Usando Lizard[1] podremos medir la complejidad ciclomática de cada método dentro de nuestras clases de Java.

IV. Resultados

```
==============================================
  NLOC    CCN   token  PARAM  length  location
----------------------------------------------
16 5 180 0      20 Maze::generarNodos@35-54@.\Maze
39 10 329 0      59 Maze::siguientesPasadas@79-137
13 6 127 2      14 Maze::areNodesEquivalent@182-19
```

Es el resultado del análisis de Lizard. A partir de esto podemos determinar que hay 2 métodos en el proyecto que serían 'difícil de mantener'.

V. Subrutinas largas

Las rutinas 'largas' tienen una mala reputación en el ambiente empresarial y en la programación orientada a objetos. Existe la idea de compartimentar al máximo los métodos de una clase llegando en algunos casos a lo absurdo. Con el objetivo de hacer más fácil de entender estas rutinas se puede llegar a tal punto donde hay recorrer varios métodos para lograr entender un 'algoritmo'. Creo que esto no es siempre lo correcto, y se puede lograr el objetivo escribiendo la acción que se quiere realizar en un solo método, bien ordenado, e.g. Aprovechando los distintos scopes que se pueden hacer dentro de métodos en lenguajes como Java.

VI. Conclusión

La rutina 'siguientesPasadas' tiene un numero de complejidad ciclomática de 10, algunos lo considerarían cómo muy alto, para disminuir este numero se tratará de mejorar el grafo de flujo del método sin aumentar la cantidad de métodos necesarios para terminar la acción.

Referencias

[1] Lizard