INSTITUTO POLITECNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERÍA CAMPUS ZACATECAS

Ing. Sistemas Computacionales

Análisis de Algoritmos

Jesús Abelardo Dávila Mauricio

Presenta:

PRUEBAS AL ALGORITMO DE N-REINAS PARA 8, 15, 30, 70, 90 REINAS

(Reporte)

A 29 de Noviembre del 2019

Introducción

El problema de las n-Reinas es muy antiguo, propuesto por primera vez en el año 1848, consiste en encontrar una asignación a *n* reinas en un tablero de ajedrez de *n* x *n* de modo tal, que estas no se ataquen. En el presente reporte se presentan las diferentes pruebas para el algoritmo proporcionado por el profesor; las pruebas se realizaron para 8, 15, 30, 70 y 90 reinas, para sus respectivos tableros.

Descripción

Las pruebas se realizaron modificando los datos de una función constructora de la clase *Configuración* tales como:

```
this.numGeneraciones = numGeneraciones;
this.tamPoblacion = tamPoblacion;
this.probMuta = probMuta;
this.pMuestra = pMuestra;
this.mask = Cruza.generarMascaraAleatoria(tamGenotipo);
this.tipoSeleccion = tipoSeleccion;
this.tamGenotipo = tamGenotipo;
```

Los datos que más se modificaban eran

- *numGeneraciones*
- *tamPoblacion*
- *probMuta*
- *pMuestra*
- *tamGenotipo*

estos datos se modificaron de tal manera que el algoritmo resolviera el problema en el menor tiempo posible, y que, en el mejor caso, se obtuviera la mejor solución para el problema.

A continuación, se muestran cada una de las configuraciones que se usaron para resolver el problema para cada una de las pruebas propuestas en clase.

Para 8 reinas en un tablero de 8x8:

Configuracion c1 = new Configuracion(100, 64, 0.1, 0.001, new Seleccion.TipoSeleccion[]{Seleccion.TipoSeleccion.RANDOM,Seleccion.TipoSeleccion.TORNEO}, 8);

Solución:

g: 10 [1, 5, 0, 6, 3, 7, 2, 4]

Para 15 reinas en un tablero de 15x15

Configuracion c1 = new Configuracion(10000, 225, 0.1, 0.001, new Seleccion.TipoSeleccion[]{Seleccion.TipoSeleccion.RANDOM,Seleccion.TipoSeleccion.TORNEO}, 15);

Solución:

g: 370 [5, 9, 11, 4, 10, 12, 0, 2, 7, 13, 1, 8, 14, 3, 6]

Para 30 reinas en un tablero de 30x30

Configuracion c1 = new Configuracion(10000, 900, 0.1, 0.01, new Seleccion.TipoSeleccion[]{Seleccion.TipoSeleccion.RANDOM,Seleccion.TipoSeleccion.TORNEO}, 30);

Solución:

g: 33 [6, 9, 19, 21, 29, 17, 22, 12, 28, 7, 13, 0, 14, 1, 4, 24, 26, 8, 2, 11, 20, 25, 15, 18, 23, 10, 27, 5, 3, 16]

Para 70 reinas en un tablero de 70x70

Configuracion c1 = new Configuracion(50000, 60, 0.3, 0.01, new Seleccion.TipoSeleccion[]{Seleccion.TipoSeleccion.RANDOM,Seleccion.TipoSeleccion.TORNEO}, 70);

Solución:

g: 2775 [53, 40, 35, 16, 13, 24, 36, 54, 18, 30, 60, 63, 4, 25, 51, 56, 17, 68, 49, 27, 55, 6, 10, 28, 19, 29, 37, 20, 62, 43, 0, 58, 61, 11, 14, 57, 33, 23, 21, 64, 66, 41, 31, 59, 47, 69, 48, 5, 2, 8, 34, 26, 3, 44, 22, 7, 52, 39, 65, 46, 50, 67, 38, 1, 45, 42, 15, 32, 12, 9]

Para 90 reinas en un tablero de 90x90

Configuracion c1 = new Configuracion(50000, 60, 0.3, 0.001, new Seleccion.TipoSeleccion[]{Seleccion.TipoSeleccion.RANDOM,Seleccion.TipoSeleccion.TORNEO}, 90);

Solución:

g: 3960 [68, 76, 10, 48, 30, 78, 37, 47, 6, 62, 22, 67, 88, 27, 8, 38, 31, 55, 52, 70, 9, 42, 39, 87, 2, 25, 61, 19, 58, 72, 14, 50, 64, 34, 73, 20, 83, 4, 84, 21, 26, 89, 16, 33, 13, 0, 56, 71, 75,

82, 29, 5, 51, 12, 74, 80, 18, 40, 46, 3, 35, 63, 7, 43, 69, 60, 77, 24, 45, 15, 57, 44, 65, 86, 1, 81, 85, 28, 36, 32, 41, 11, 17, 49, 54, 66, 23, 59, 79, 53]

Conclusión

A pesar de que las pruebas dieron un resultado en el menor tiempo, al probar otra vez la misma configuración, el algoritmo trabaja peor o en el mejor caso, mejor que la primera vez probado. Esto es a que su probabilidad de mutación influye durante el proceso de búsqueda de la mejor solución para el problema, para que este no se estanque; aunque en ocasiones esto lo puede empeorar.