UNIVERSIDAD
DE SANTIAGO
DE CHILE

Departamento de Matemática y Ciencia de la Computación

Laboratorio 1

Algoritmo para Determinar si un Alfil Ataca a una Pieza Contraria

Segundo Semestre 2016

Programación Avanzada 26106
Licenciatura en Ciencia de la Computación

Rubén Carvajal Schiaffino
ruben.carvajal@usach.cl

1 Introducción

El objetivo de este trabajo es construir un algoritmo que dadas las coordenadas de dos piezas de ajedrez, de las cuales una es un alfil, determine si el alfil puede atacar a la otra pieza.

2 Procedimiento

Un alfil ataca a la pieza contraria si esta se encuentra en alguna de las diagonales Dado que cada pieza ocupa una posición en el tablero es suficiente con calcular la pendiente de la recta que pasa por las coordenadas en las cuales se encuentran ambas piezas.

Si el valor de dicha pendiente es 1 o -1, entonces el alfil ataca a la pieza contraria.

2.1 Restricciones

Se asume que las coordenadas de ambas piezas son válidas y que cada ambas piezas no ocupan la misma posición.

3 Estructuras de Datos Utilizadas

Para este problema no se ocupan estructuras de datos.

4 Algoritmo

El algoritmo para ...

Algorithm Bootstrap

Input:	X, Y set of n observations
	nbi Number of bootstrap iterations
	p percent of the confidence interval
Output:	x_l, x_r Confidence interval

1 $\rho \leftarrow$ **CorrelationCoef**(X, Y)
2 **for** $i \leftarrow 1$ **to** nbi **do**
3 $\quad X', Y' \leftarrow$ **SampleFrom**(X, Y, n)
4 $\quad cint_i \leftarrow$ **CorrelationCoef**(X', Y')
5 **Sort**$(cint)$
6 $x_l, x_r \leftarrow$ **ConfidenceInterval**$(cint, p)$
7 **return** x_l, x_r

El algoritmo primero verifica si ambas piezas se encuentran en la misma fila

(línea 1) o en la misma columna (línea 3); en ambos casos el resultado es *Fail*. Si ambas piezas no se encuentran en la misma fila o columna se calcula la pendiente de la recta que pasa por las coordenadas en las cuales se encuentran ambas piezas (línea 5) y luego se verifica si el valor de la pendiente es ± 1 (línea 6).

4.1 Análisis de Complejidad

El algoritmo propuesto tiene costo constante.

5 Implementación

El algoritmo está implementado en lenguaje C como se muestra en la figura a continuación:

5.1 Plataforma Computacional

El programa fue ejecutado eun un computador con las siguientes características:

- **Procesador:** Intel(R) Pentium(R) 4 CPU 3.00GHz
- **Memoria RAM:** 994312 kB
- **Sistema Operativo:** Linux - Ubuntu 9.04

6 Resultados Experimentales

En este caso es trivial ...

Nov 11, 10 15:00 **alfil.c** Page 1/1

```c
/*
 * alfil.c :
 *
 * Author: Ruben Carvajal Schiaffino
 *
 * Date: 11/11/2010
 *
 */

#include <stdio.h>
#include <stdlib.h>

#define POSITIVE  1.
#define NEGATIVE -1.

int InSamePos(int af, int ac, int pf, int pc) {

   if (af == pf && ac == pc)
      return 1;
   return 0;
}

float Slope(int x1, int y1, int x2, int y2) {

   if (x1 == x2)
      return 0.;
   return (float)(y2 - y1) / (float)(x2 - x1);
}

main(int argc, char **argv) {

   int af,ac,pf,pc;
   float m;

   if (argc == 5) {
      af = atoi(argv[1]);
      ac = atoi(argv[2]);
      pf = atoi(argv[3]);
      pc = atoi(argv[4]);
      if (!InSamePos(af,ac,pf,pc)) {
         m = Slope(af,ac,pf,pc);
         if (m == POSITIVE || m == NEGATIVE)
            printf("Success\n");
         else
            printf("Fail\n");
      }
   }
   else
      printf("Error\n");
}
```

Thursday November 11, 2010 1/1