INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

ALGORITMO GENÉTICO DE SELECCIÓN POR TORNEO

Práctica 4

Dominguez de la Rosa Bryan

GRUPO 3CM5

Profesor: Morales Güitron Sandra Luz

22 de octubre de 2018

Introducción

La idea básica del método de selección por torneo es escoger individuos con base en comparaciones directas entre ellos.

Hay 2 versiones de la seleccion mediante torneo:

- Determinística.
- Probabilística.

En esta práctica implementamos la versión probabilística, la cual realiza los siguientes pasos:

1. Barajar los individuos de la población.
2. Escoger un número p de individuos (normalmente 2).
3. Compararlos con base en su aptitus.
4. Generar un *flip*.
5. Elegir al individuo más apto si *flip* resulta verdadero, en caso contrario se elige al individuo menos apto.

Contenido

Para la implementación del algoritmo de selección por torneo, implementé 4 arreglos de bits para controlar las distintas etapas que se realizan en el algoritmo:

- Población inicial.
- Población de individuos seleccionados mediante el algoritmo de selección por torneo.
- Población después de cruza.
- Población después de mutación.

En la primer etapa, se llena aleatoriamente el arreglo de población inicial con series de 5 bits. Después, para implementar la selección por torneo, se barajean los individuos de la población, con el fin de generar parejas para el enfrentamiento. Al darse el primer enfrentamiento, se seleccionan 16 individuos ganadores, por lo que es necesario barajear de nuevo la población y volver a realizar el algoritmo, para de este modo completar los 32 individuos que tenía nuestra población original.

Una vez teniendo la población de selección de padres, se realiza una cruza de individuos de la siguiente manera:

1. Se utilizan 2 individuos de la población de padres.
2. Se define un punto de cruza estático para todas las generaciones.
3. Se cruzan los individuos.
4. Se retorna el individuo resultante.

```
bitset<BITS_PER_INDIVIDUAL> crossAlgorithm(bitset<BITS_PER_INDIVIDUAL> &p1, bitset<BITS_PER_INDIVIDUAL> &p2, int cross_point) {

    bitset<BITS_PER_INDIVIDUAL> aux = p1;

    for (int i = 0; i <= cross_point; i++)
    {
        aux.set(cross_point - i, p2[cross_point - i]);
    }

    return aux;
}
```

Figura 1: Algoritmo de cruza de individuos

Al obtener la población de individuos después de la cruza se necesita realizar una mutación. En este caso se generó una mutación del 10 % de la población. Nuestra población total es de 32 elementos, entonces la cantidad de individuos a redondear es 3.2, redondeado como 3.

La mutación se realiza de la siguiente forma:

1. El algoritmo se realizará 3 veces.
2. La mutación buscará mejorar al individuo, por lo tanto, se buscará cambiar un bit 0 por un bit 1.
3. Debido a que se requiere buscar un 0 en el individuo a mutar, y es posible que el individuo no tenga bits 0, se define un número máximo de iteraciones para evitar que el programa se cicle.
4. Cuando se encuentre un bit 0, se cambia por un bit 1.

```
bitset<BITS_PER_INDIVIDUAL> mutationAlgorithm(bitset<BITS_PER_INDIVIDUAL> individual){
    bitset<BITS_PER_INDIVIDUAL> result = individual;

    int cont = 0;

    while(cont <= MAX_SEARCH_VALUE)
    {

        int mutation_point = rand() % BITS_PER_INDIVIDUAL;
        if(result[mutation_point] == 0){
            result.set(mutation_point, 1);
            break;
        }
        cont++;
    }

    return result;
}
```

Figura 2: Algoritmo de mutación de individuos

Una vez que se tenga la población mutada, se establece ésta como población inicial, para realizar el algoritmo de ruleta en la siguiente generación.

Al obtener la población final de una generación, se obtiene la aptitud del individuo de menor valor, la aptitud del individuo de mayor valor y el promedio de aptitud de cada generación.

A continuación se muestran 2 ejemplos con 10, 30, 50 y 100 generaciones, en los que la línea azul representa la aptitud del mejor individuo de cada generación, la línea roja representa la aptitud del peor individuo de cada generación y la línea blanca representa el promdedio de aptitud de cada generación.

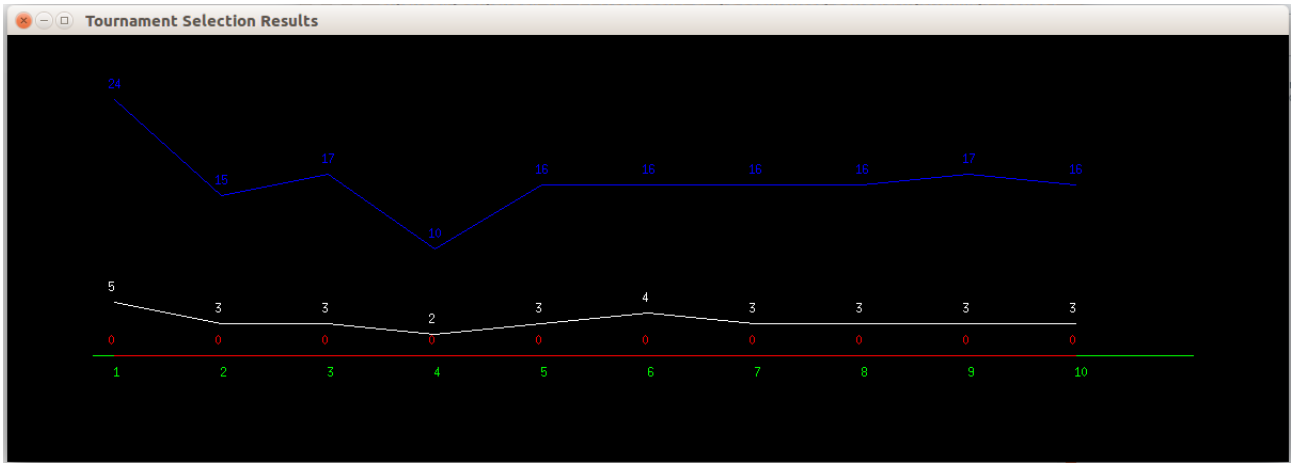

Figura 3: Resultado 1 con 10 generaciones

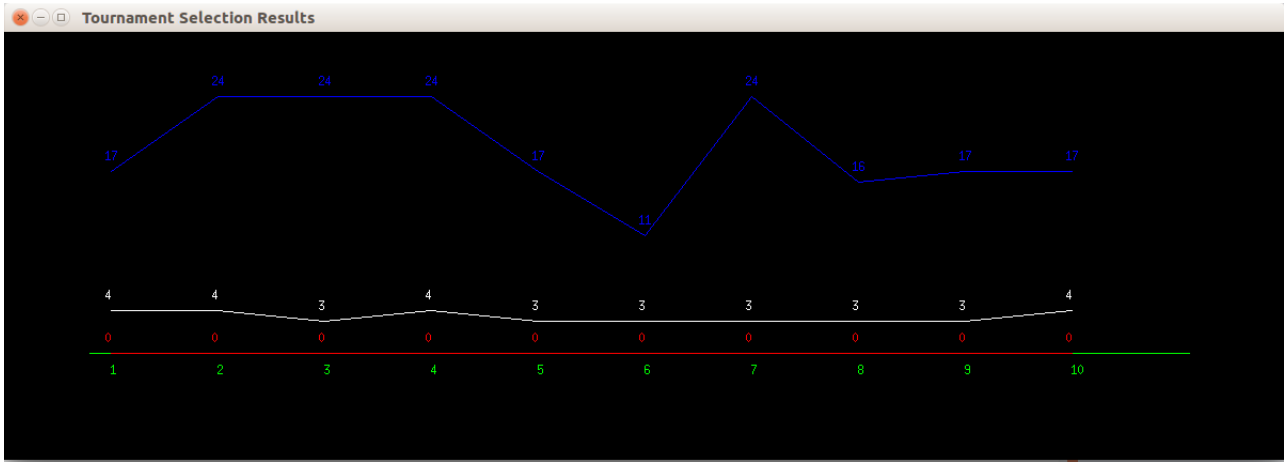

Figura 4: Resultado 2 con 10 generaciones

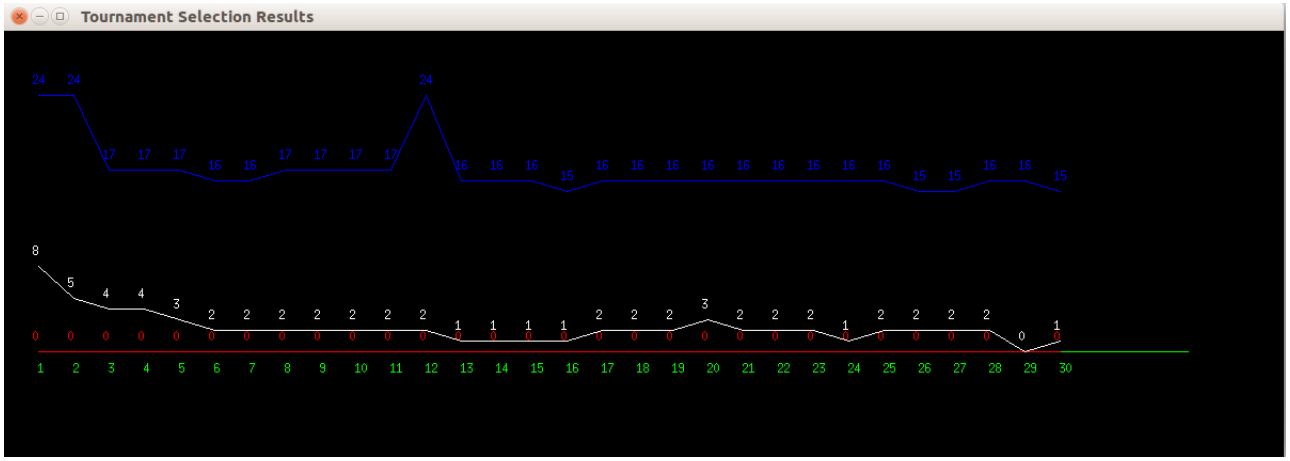

Figura 5: Resultado 1 con 30 generaciones

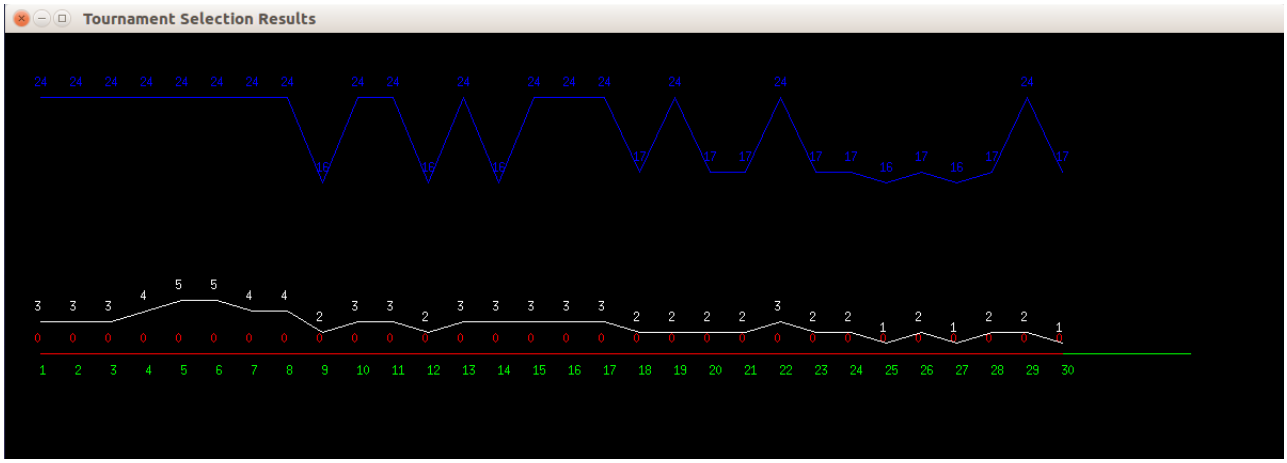

Figura 6: Resultado 2 con 30 generaciones

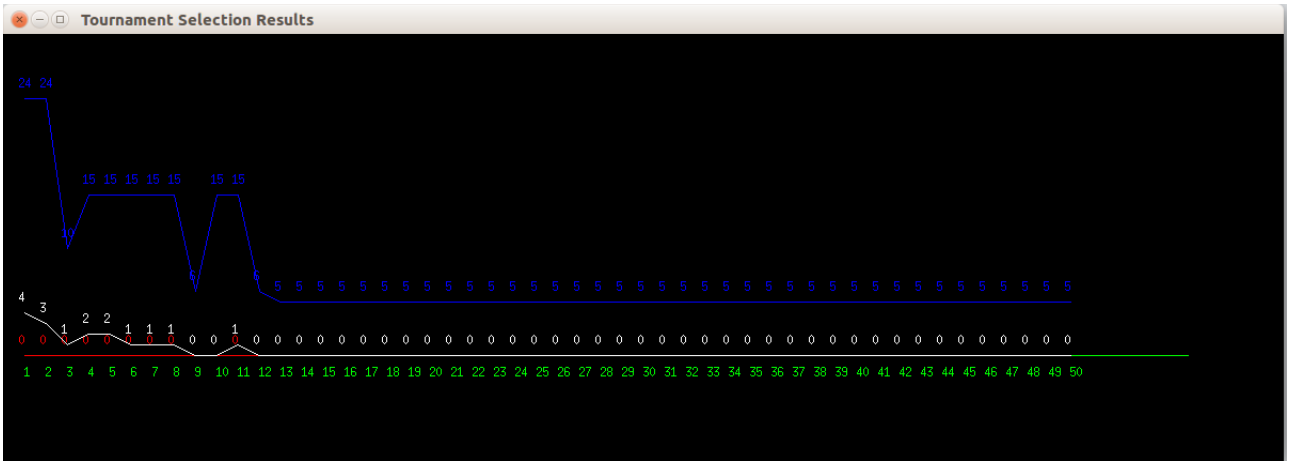

Figura 7: Resultado 1 con 50 generaciones

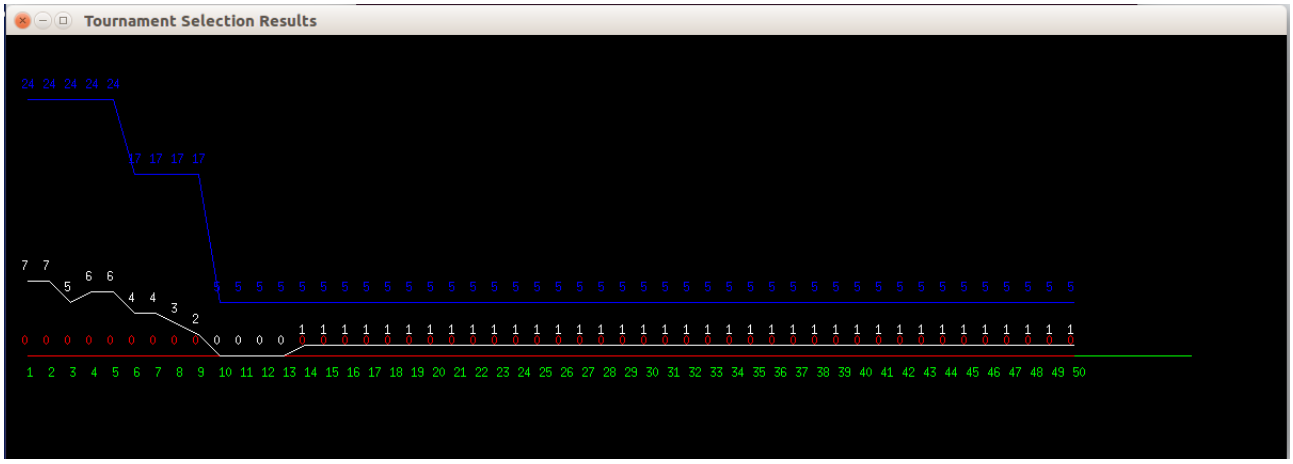

Figura 8: Resultado 2 con 50 generaciones

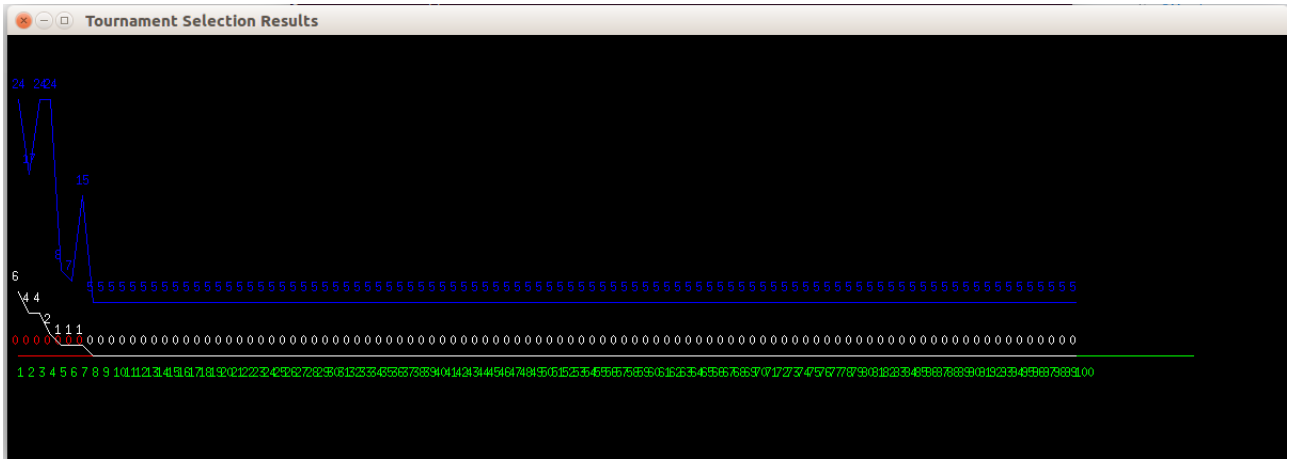

Figura 9: Resultado 1 con 100 generaciones

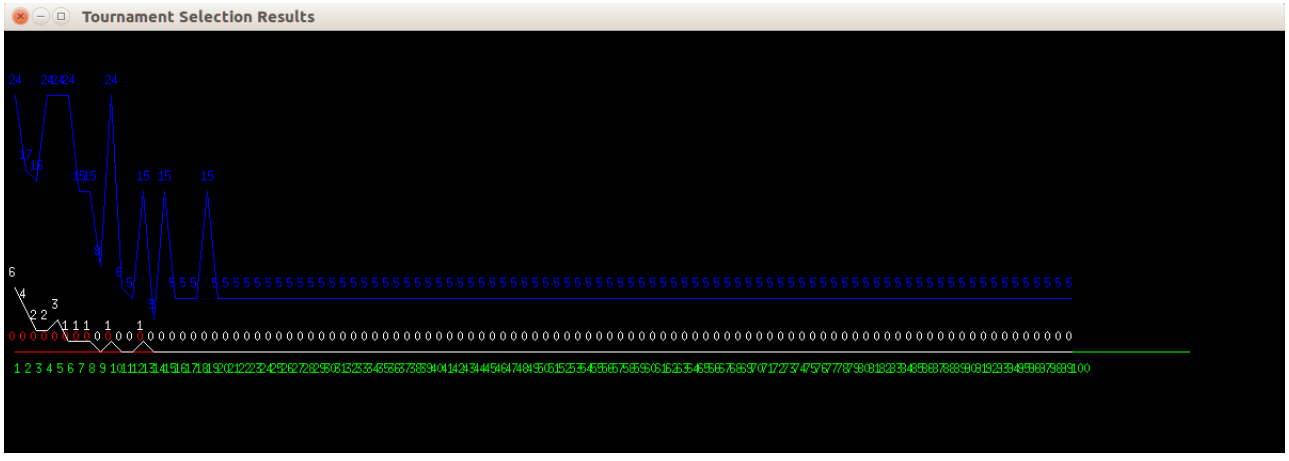

Figura 10: Resultado 2 con 100 generaciones

Conclusión

La selección probabilística implica que los individuos que resultan ser seleccionados pueden no ser los más aptos, a diferencia de la selección determinista, que siempre nos entregará como resultado un individuo más apto que su rival.

En la naturaleza, existen muchos factores que determinan que un individuo pueda resultar ganador en un enfrentamiento, y no siempre el más apto resulta ganador, por lo que el algoritmo probabilístico resulta ser una simulación más cercana a la realidad.