SISTEMAS INTELIGENTES I

Bloque 1: Algoritmos Genéticos

Guía de Prácticas

Javier J. Sánchez Medina
javier.sanchez@ulpgc.es

El propósito de esta guía de prácticas es dotar a los/as alumnos/as de Sistemas Inteligentes I, de los rudimentos prácticos en diseño y programación de Algoritmos Genéticos, Redes Neuronales y una introducción a la Minería de Datos

Esto complementa los contenidos teóricos que de esta asignatura serán impartidos.

La asignatura consta de 15 horas de prácticas de Aula y 15 horas de prácticas de Laboratorio. Se dividirán en los tres bloques arriba mencionados:

Bloque 1: Algoritmos Genéticos (6 semanas)
Bloque 2: Redes Neuronales Artificiales (6 semanas)
Bloque 3: Minería de Datos (2 semanas)

Las prácticas son individuales.

Práctica 2: Algoritmos Genéticos. Operadores “Selección”, “Cruce” y “Mutación”.

Introducción:

El propósito de la presenta práctica es profundizar un poco más en los distintos tipos de operadores de que se compone el algoritmo genético, visitando las versiones más conocidas de los mismos. En particular, se pretenden estudiar:

Operador Selección:

- Selección proporcional al fitness (Fitness-proportionate Selection): Selección de ruleta con reemplazamiento (RWSR).
- Selección estocástica universas (Stochastic Universal Selection, SUS)
- Escalado del fitness
- Truncado
- Selección por torneo
- Elitismo

Operador Cruce:

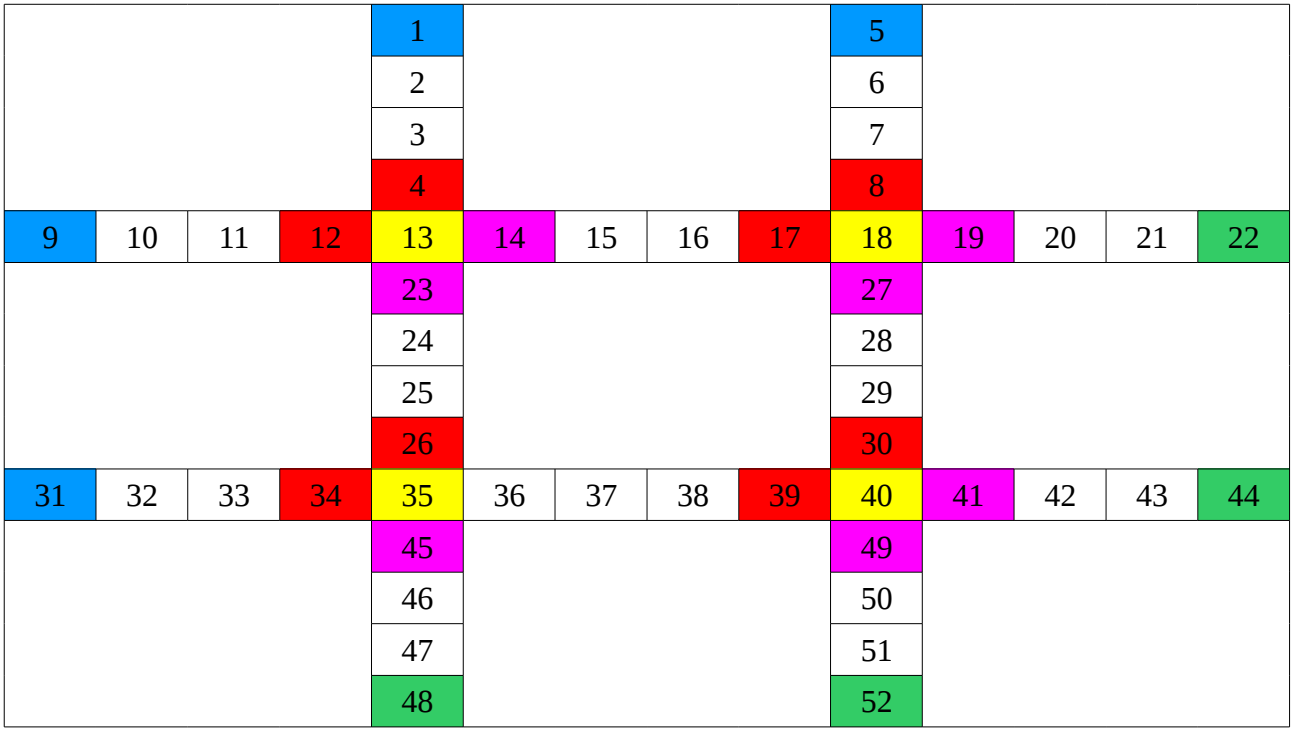
- One-point-crossover
- Two-points-crossover
- Cruce uniforme
- Cruce semi-uniforme
- Cut and splice
- Three parent Crossover

Operador Mutación:

- Bit string mutation
- Boundary
- Gausiana
- Hipermutación

Problema a resolver:

Sea el siguiente modelo de simulación de tráfico basado en autómatas celulares.



Hay 6 tipos de celdas:

1 – **Celdas de entrada**. Pueden tener dos estados: ocupado o libre. Tienen asociada una variable entera (natural) "cola" donde acumulamos los vehículos esperando ocupar esa celda. Sólo tiene una celda vecina, que es la convencional posterior.

2 – **Celdas de salida**. Pueden tener dos estados: ocupado o libre. Asumimos que más allá de las celdas de salida hay un sumidero de infinita capacidad. Es decir, que siempre hay una celda libre que no vemos. Sólo tiene una celda vecina, la celda convencional anterior.

3 – **Celdas semáforo**. Pueden tener 2 estados de ocupación (ocupado o libre) y dos estados de semáforo (verde o rojo). Están asociadas por parejas (12,4), (8,17), (26,34) y (30,39). El estado de semáforo de los dos miembros de una pareja siempre es complementario, pueden estar en (verde, rojo) o en (rojo y verde). Tiene dos celdas vecinas: La celda convencional anterior y la celda intersección posterior.

4 – **Celdas intersección**. Pueden tener tres estados: libre, ocupado_1 (por un vehículo que va hacia la derecha) y ocupado_2 (por un vehículo que va hacia abajo). Tiene 4 celdas vecinas: Las dos celdas semáforo anteriores y las dos celdas convencionales siguientes, para las dos ramas que parten de ella.

5 – **Celdas post-intersección**. Pueden tener dos estados: libre y ocupado. Tiene dos celdas vecinas: la celda intersección anterior y la celda convencional posterior.

6 – Celda convencional. Pueden tener dos estados: libre y ocupado. Tiene dos celdas vecinas, la celda convencional o post-intersección anterior, y la celda convencional o semáforo o salida posterior.

Para cada tipo de celda definimos un conjunto de reglas para el cálculo del siguiente estado de cada una de ellas, en función de sus vecinas. Se calcula la actualización de todas las celdas en una copia

de la red antes de actualizarlas todas a la vez en la red que estamos simulando.

Reglas de actualización de estado:

1 – **Celdas de entrada**.

Si toca nuevo vehículo (según el perfil de demanda) → incrementamos el valor de la cola en una unidad.

Si el estado de la celda es ocupada

Si la celda convencional vecina tiene estado=LIBRE

Si el valor de la cola es mayor que 0, el estado futuro = OCUPADO, y cola = cola-1

Si no (Si el valor de la cola es 0), estado futuro = LIBRE

Fin si

Si no, no se modifica el estado de la celda.

Fin si

Si no (si el estado de la celda es LIBRE)

Si el valor de la cola es mayor que 0, el estado futuro = OCUPADO, y cola = cola-1

Si no, no se modifica el estado de la celda.

Fin si

Fin si

2 – **Celdas de salida**.

Si el estado de la celda es ocupada

Incrementamos el contador de vehículos que sale por esta salida en uno.

Fin si

Si la celda convencional vecina tiene estado=LIBRE, estado futuro = LIBRE.

Si no (si la celda convencional vecina tiene estado = OCUPADO), estado futuro = OCUPADO.

Fin si

3 – **Celdas semáforo**.

Si el estado de semáforo es ROJO OR (estado semaforo = VERDE AND celda interseccion <> LIBRE)

Si el estado de la celda convencional vecina tiene estado=LIBRE, no se modifica el estado

Si no (si la celda convencional vecina tiene estado = OCUPADO), estado futuro = OCUPADO.

Fin si

Si no (estado semaforo = VERDE AND celda interseccion = LIBRE)

Si celda convencional vecina tiene estado=OCUPADO, estado futuro=OCUPADO

Si no, estado futuro = LIBRE

Fin si

Fin si

4 – **Celdas intersección**.

Miramos sólo la celda semáforo vecina con el semáforo verde.

Si estado semáforo verde vecino es OCUPADO

Si estado celda intersección es LIBRE, estado futuro es Si su estado es OCUPADO_1 si está abierto el semáforo de la izquierda o OCUPADO_2 si está abierto el semáforo que viene de arriba.

Si no (estado celda intersección <> LIBRE)

Si (estado celda intersección = OCUPADO_1 AND estado vecina derecha = LIBRE) OR (estado celda intersección = OCUPADO_2 AND estado vecina inferior = LIBRE)

estado futuro celda interseccion = OCUPADO_semaforo_VERDE

Si no (no hay hueco para que el vehículo en la intersección salga de la misma por el carril que debe) se mantiene el estado de la celda intersección

Fin si

Fin si

Si no (estado semáforo verde vecino es LIBRE)

Si (estado celda intersección es OCUPADO_1 AND estado vecina derecha = LIBRE) OR (estado celda intersección es OCUPADO_2 AND estado vecina inferior = LIBRE)

estado celda interseccion es LIBRE

Si no, se mantiene el estado de la celda intersección

Fin si

Fin si

5 – **Celdas post-intersección**. La celda post-intersección tiene interés por el estado OCUPADO_1, si es la derecha, o el estado OCUPADO_2 si es la post-intersección inferior. Ese estado de interés lo vamos a llamar OCUPADO_Interés.

Si (estado de la celda convencional = LIBRE AND estado celda anterior = OCUPADO_Interés) OR (estado de la celda convencional = OCUPADO AND estado celda posterior = OCUPADO) OR (estado de la celda convencional = OCUPADO AND estado celda anterior = OCUPADO_Interés AND estado celda posterior = LIBRE)

estado futuro de celda convencional = OCUPADO

Si no, estado futuro de celda convencional = LIBRE

Fin si

6 – Celda convencional. Cada celda convencional tiene una celda vecina anterior y una celda vecina posterior, y las trata de manera independiente al tipo de celdas que son.

Si (estado de la celda convencional = LIBRE AND estado celda anterior = OCUPADO) OR (estado de la celda convencional = OCUPADO AND estado celda posterior = OCUPADO) OR (estado de la celda convencional = OCUPADO AND estado celda anterior = OCUPADO AND estado celda posterior = LIBRE)

estado futuro de celda convencional = OCUPADO

Si no, estado futuro de celda convencional = LIBRE

Fin si.

La forma de realizar la simulación es:

1 – actualizar los valores de las colas

2 – actualizar los estados de los semáforos según el cromosoma que se esté evaluando

3 – actualizar los valores de estado de todas las celdas, de los 6 tipos, en una copia de los estados

4 – sobre-escribir los estados de todas las celdas con los nuevos estados (y almacenar el número de vehículos que van saliendo)

Se itera hasta alcanzar el número máximo de iteraciones de la simulación.

El objetivo del ejercicio:

Obtener la secuencia de estados de semáforos óptima, desde el punto de vista del ancho de banda de salida (vehículos por minuto), medido durante 2 horas virtuales de simulación (7200 iteraciones). Para cada par de semáforos, se optimizará un ciclo de 2 minutos. La duración mínima de un estado es de 10 segundos (una iteración del simulador será equivalente a un segundo). Por ello, la longitud de genes del cromosoma será de 4x12 (4 pares de semáforos, secuencias de 12 estados en cada uno de ellos).

El perfil de demanda será muy exigente: un nuevo vehículo en cada entrada cada 5 segundos/iteraciones.

Objetivos de la práctica:

Los objetivos de esta práctica se pueden resumir como sigue:

0: Montar la estructura de datos de la red de tráfico y el simulador basado en autómatas celulares.
1: Diseñar la codificación apropiada para el problema a resolver.
2: Diseñar la función de *fitness* basada en la maximización del número de vehículos por minuto que salen de la red durante la simulación.
3: Diseñar los operadores genéticos a utilizar y los parámetros Tamaño de la Población, Probabilidad de Mutación, número de generaciones (iteraciones)
4: Programar el algoritmo genético
5: Ejecutarlo e iterar al punto 3 hasta que se obtengan los mejores resultados, que nos de confianza en encontrar el óptimo en la mayoría de las ejecuciones del algoritmo, evitando la convergencia prematura.
6: Representar la evolución del fitness máximo y medio de la población.
7: Opcionalmente, representar visualmente las simulaciones, en particular, las que se produzcan con los mejores resultados del ciclo de los semáforos.

Entregables:

Memoria de menos de 10 páginas con un anexo con el código Matlab (u otros) con el algoritmo genético y el simulador de tráfico basado en autómatas celulares..

La memoria debe incluir:

- Codificación utilizada para el cromosoma
- Función de evaluación diseñada
- Operadores utilizado de selección, cruce, mutación y condición de finalización
- Valores de los parámetros ajustados:
  - Tamaño de la población
  - Número de generaciones
  - Probabilidad de Mutación
- Representación de la evolución del *fitness* máximo y medio para la mejor combinación