Biased Random Key Genetic Algorithm con Búsqueda Local para el Team Orienteering Problem

Alejandro Lix Klett

Directora: Prof. Dra. Irene Loiseau

Departamento de Computación

June 1, 2018

Contenido

Orienteering Problem

- Orientación es un deporte originario de Escandinavia

Orienteering Problem

- Orientación es un deporte originario de Escandinavia
- Cada jugador comienza en un punto de control y debe visitar tantos otros puntos de control como le sea posible dentro de un tiempo limite preespecificado.

Orienteering Problem

- Orientación es un deporte originario de Escandinavia
- Cada jugador comienza en un punto de control y debe visitar tantos otros puntos de control como le sea posible dentro de un tiempo limite preespecificado.
- Cada punto de control tiene un puntaje.

Orienteering Problem

- Orientación es un deporte originario de Escandinavia
- Cada jugador comienza en un punto de control y debe visitar tantos otros puntos de control como le sea posible dentro de un tiempo limite preespecificado.
- Cada punto de control tiene un puntaje.
- Cada punto de control puede ser visitado una sola vez a lo sumo.

Orienteering Problem

- Orientación es un deporte originario de Escandinavia
- Cada jugador comienza en un punto de control y debe visitar tantos otros puntos de control como le sea posible dentro de un tiempo limite preespecificado.
- Cada punto de control tiene un puntaje.
- Cada punto de control puede ser visitado una sola vez a lo sumo.
- El objetivo es maximizar el puntaje total.

Orienteering Problem

- Orientación es un deporte originario de Escandinavia
- Cada jugador comienza en un punto de control y debe visitar tantos otros puntos de control como le sea posible dentro de un tiempo limite preespecificado.
- Cada punto de control tiene un puntaje.
- Cada punto de control puede ser visitado una sola vez a lo sumo.
- El objetivo es maximizar el puntaje total.
- Este problema se conoce como Orienteering Problem (OP). El OP es NP-Hard como demostraron Golden, Levy y Vohra.

Team Orienteering Problem

- Hay M clientes, cada uno tiene un beneficio b_i y una coordenada en el plano.

Team Orienteering Problem

- Hay M clientes, cada uno tiene un beneficio b_i y una coordenada en el plano.
- Los puntos de salida y llegada tienen beneficio cero

Team Orienteering Problem

- Hay M clientes, cada uno tiene un beneficio b_i y una coordenada en el plano.
- Los puntos de salida y llegada tienen beneficio cero
- Hay N vehículos

Team Orienteering Problem

- Hay M clientes, cada uno tiene un beneficio b_i y una coordenada en el plano.
- Los puntos de salida y llegada tienen beneficio cero
- Hay N vehículos
- El beneficio de los clientes solo puede ser recolectado una vez.

Team Orienteering Problem

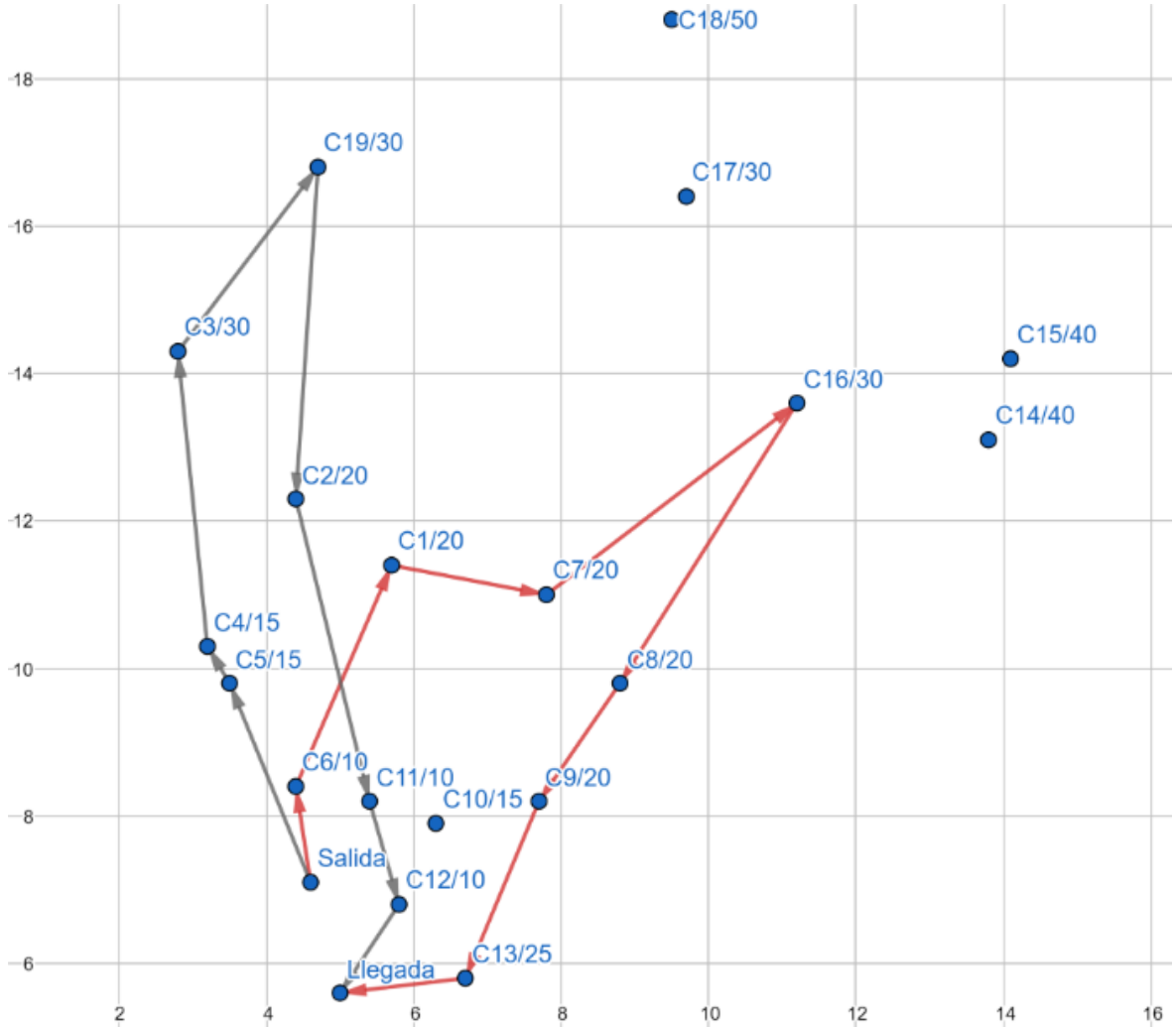
- Hay M clientes, cada uno tiene un beneficio b_i y una coordenada en el plano.
- Los puntos de salida y llegada tienen beneficio cero
- Hay N vehículos
- El beneficio de los clientes solo puede ser recolectado una vez.
- El objetivo es maximizar la sumatoria de los beneficios recolectados de todos los vehículos.

Team Orienteering Problem

- Hay M clientes, cada uno tiene un beneficio b_i y una coordenada en el plano.
- Los puntos de salida y llegada tienen beneficio cero
- Hay N vehículos
- El beneficio de los clientes solo puede ser recolectado una vez.
- El objetivo es maximizar la sumatoria de los beneficios recolectados de todos los vehículos.
- Como TOP contiene a OP, es al menos tan difícil.

Instancia p2.2.k del benchmark de Tsiligirides

La instancia tiene dos vehículos con un $d_{max} = 22{,}50$. Hay 19 clientes además de los puntos de salida y llegada.



Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Características:

- Son estrategias que guían procesos de búsqueda.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Características:

- Son estrategias que guían procesos de búsqueda.
- Sus conceptos se pueden describir con un gran nivel de abstracción. No son para un problema específico.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Características:

- Son estrategias que guían procesos de búsqueda.
- Sus conceptos se pueden describir con un gran nivel de abstracción. No son para un problema específico.
- En muchos casos son algoritmos no-determinísticos.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Características:

- Son estrategias que guían procesos de búsqueda.
- Sus conceptos se pueden describir con un gran nivel de abstracción. No son para un problema específico.
- En muchos casos son algoritmos no-determinísticos.
- Sus desarrollos y diseños suelen estar motivados por comportamientos naturales.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Características:

- Son estrategias que guían procesos de búsqueda.
- Sus conceptos se pueden describir con un gran nivel de abstracción. No son para un problema específico.
- En muchos casos son algoritmos no-determinísticos.
- Sus desarrollos y diseños suelen estar motivados por comportamientos naturales.
- No garantizan que una solución óptima sea encontrada.

Metaheurísticas

- Son métodos diseñados para encontrar buenas soluciones, en un tiempo razonable, a problemas de optimización combinatoria en general.
- Las metaheurísticas son estrategias de alto nivel que guían una heurística específica del problema a resolver para mejorar su perfomance.

Características:

- Son estrategias que guían procesos de búsqueda.
- Sus conceptos se pueden describir con un gran nivel de abstracción. No son para un problema específico.
- En muchos casos son algoritmos no-determinísticos.
- Sus desarrollos y diseños suelen estar motivados por comportamientos naturales.
- No garantizan que una solución óptima sea encontrada.
- Las técnicas metaheurísticas van desde algoritmos simples de búsqueda local a complejos procesos de aprendizaje.

Metaheurísticas

Algunas técnicas:

- Simulated Annealing

Metaheurísticas

Algunas técnicas:

- Simulated Annealing
- Tabu Search

Metaheurísticas

Algunas técnicas:

- Simulated Annealing
- Tabu Search
- Algoritmos evolutivos

Metaheurísticas

Algunas técnicas:

- Simulated Annealing
- Tabu Search
- Algoritmos evolutivos
- Colonia de hormigas

Metaheurísticas

Algunas técnicas:

- Simulated Annealing
- Tabu Search
- Algoritmos evolutivos
- Colonia de hormigas
- Variable Neighborhood Search

Metaheurísticas

Algunas técnicas:

- Simulated Annealing
- Tabu Search
- Algoritmos evolutivos
- Colonia de hormigas
- Variable Neighborhood Search
- Iterated Local Search

Metaheurísticas

Algunas técnicas:

- Simulated Annealing
- Tabu Search
- Algoritmos evolutivos
- Colonia de hormigas
- Variable Neighborhood Search
- Iterated Local Search
- Etc

Algoritmos Genéticos (GA)

- Motivados en el concepto de supervivencia del más apto.

Algoritmos Genéticos (GA)

- Motivados en el concepto de supervivencia del más apto.
- Los algoritmos genéticos manejan un conjunto de individuos.

Algoritmos Genéticos (GA)

- Motivados en el concepto de supervivencia del más apto.
- Los algoritmos genéticos manejan un conjunto de individuos.
- Cada individuo es un cromosoma que codifica una solución.

Algoritmos Genéticos (GA)

- Motivados en el concepto de supervivencia del más apto.
- Los algoritmos genéticos manejan un conjunto de individuos.
- Cada individuo es un cromosoma que codifica una solución.
- Cada cromosoma tienen asociado un nivel de condición física que está correlacionado con el correspondiente valor de la función objetivo de la solución que codifica.

Algoritmos Genéticos (GA)

- Motivados en el concepto de supervivencia del más apto.
- Los algoritmos genéticos manejan un conjunto de individuos.
- Cada individuo es un cromosoma que codifica una solución.
- Cada cromosoma tienen asociado un nivel de condición física que está correlacionado con el correspondiente valor de la función objetivo de la solución que codifica.
- En cada generación se crea una nueva población con individuos provenientes de tres fuentes distintas: crossover, elites y mutantes.

Random Key Genetic Algorithm (RKGA)

- Los individuos son representados por un vector de números reales en el intervalo [0, 1].

Random Key Genetic Algorithm (RKGA)

- Los individuos son representados por un vector de números reales en el intervalo [0, 1].
- La población inicial es generada al azar.

Random Key Genetic Algorithm (RKGA)

- Los individuos son representados por un vector de números reales en el intervalo [0, 1].
- La población inicial es generada al azar.
- El decodificador es el responsable de convertir un cromosoma en una solución válida del problema.

Random Key Genetic Algorithm (RKGA)

- Los individuos son representados por un vector de números reales en el intervalo [0, 1].
- La población inicial es generada al azar.
- El decodificador es el responsable de convertir un cromosoma en una solución válida del problema.
- En cada iteración se toman los mejores individuos y pasan directamente a la siguiente generación (elites).

Random Key Genetic Algorithm (RKGA)

- Los individuos son representados por un vector de números reales en el intervalo [0, 1].
- La población inicial es generada al azar.
- El decodificador es el responsable de convertir un cromosoma en una solución válida del problema.
- En cada iteración se toman los mejores individuos y pasan directamente a la siguiente generación (elites).
- La mayoría de los individuos de la nueva generación se genera cruzando dos individuos de la generación actual (crossover).

Random Key Genetic Algorithm (RKGA)

- Los individuos son representados por un vector de números reales en el intervalo [0, 1].
- La población inicial es generada al azar.
- El decodificador es el responsable de convertir un cromosoma en una solución válida del problema.
- En cada iteración se toman los mejores individuos y pasan directamente a la siguiente generación (elites).
- La mayoría de los individuos de la nueva generación se genera cruzando dos individuos de la generación actual (crossover).
- Un porcentaje muy bajo de los nuevos individuos es generado al azar, para escapar de mínimos locales (mutantes).

Biased Random Key Genetic Algorithm (BRKGA)

- Cada individuo se genera combinando un elemento seleccionado al azar del conjunto de elite y el otro de la conjunto no-elite.

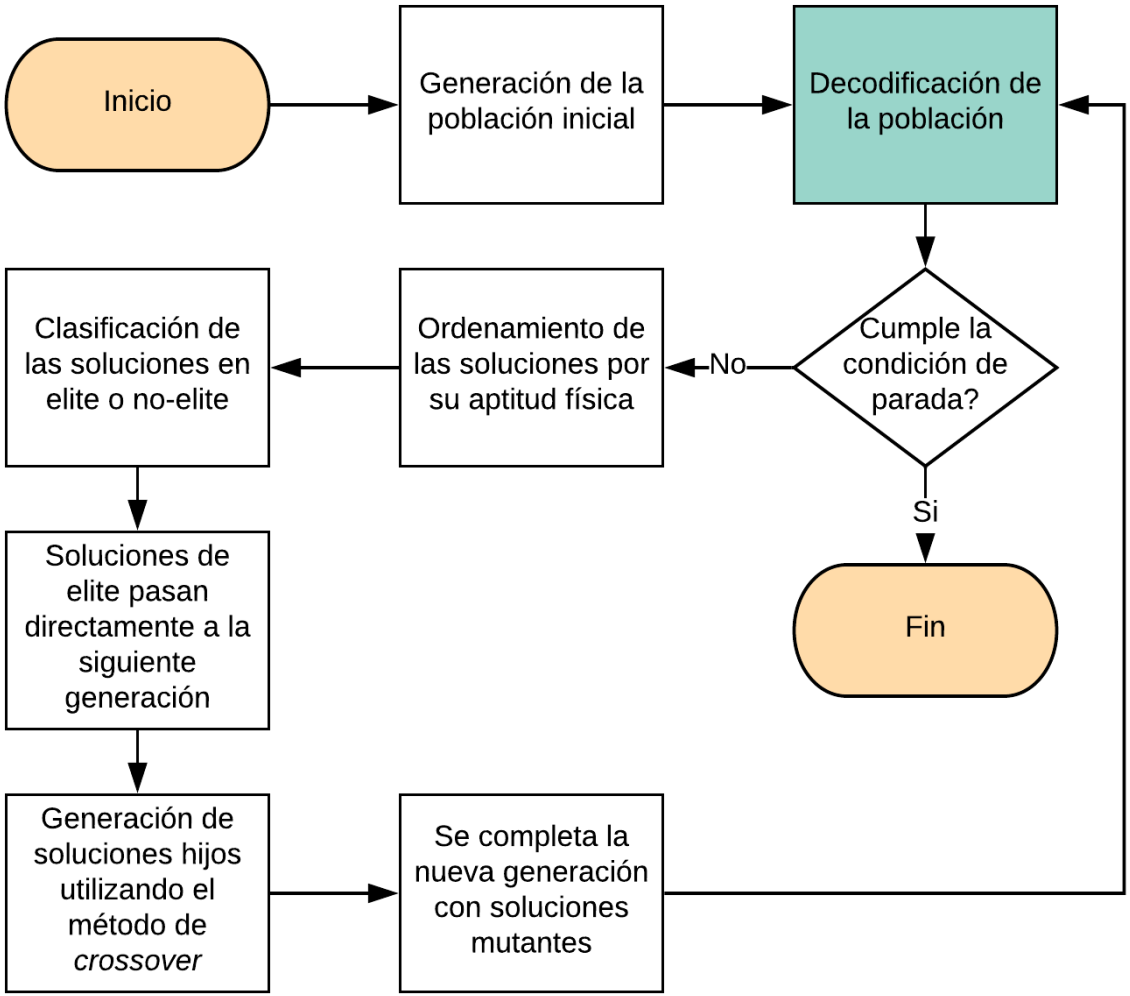
Biased Random Key Genetic Algorithm (BRKGA)

- Cada individuo se genera combinando un elemento seleccionado al azar del conjunto de elite y el otro de la conjunto no-elite.
- Parameterized Uniform Crossover. La probabilidad de que se trasmita el alelo del padre de elite es mayor que la del padre de no-elte.

Biased Random Key Genetic Algorithm (BRKGA)

- Cada individuo se genera combinando un elemento seleccionado al azar del conjunto de elite y el otro de la conjunto no-elite.
- Parameterized Uniform Crossover. La probabilidad de que se trasmita el alelo del padre de elite es mayor que la del padre de no-elte.

Diagrama de Flujo del BRKGA



Generación de la Población Inicial

- Se crea una cantidad de vectores de enteros aleatorios igual a la cantidad de soluciones por generación que se desea.

Generación de la Población Inicial

- Se crea una cantidad de vectores de enteros aleatorios igual a la cantidad de soluciones por generación que se desea.
- Los vectores tienen un tamaño igual a la cantidad de clientes de la instancia.

Generación de la Población Inicial

- Se crea una cantidad de vectores de enteros aleatorios igual a la cantidad de soluciones por generación que se desea.
- Los vectores tienen un tamaño igual a la cantidad de clientes de la instancia.
- Cada entero aleatorio del vector esta asociado a un identificador de cliente.

Generación de la Población Inicial

- Se crea una cantidad de vectores de enteros aleatorios igual a la cantidad de soluciones por generación que se desea.
- Los vectores tienen un tamaño igual a la cantidad de clientes de la instancia.
- Cada entero aleatorio del vector esta asociado a un identificador de cliente.

Ejemplo de un nuevo vector de enteros aleatorios.

Key	27	13	79	45	21	7	98	54
ClientId	1	2	3	4	5	6	7	8

Decodificación de los vectores en soluciones validas del problema

- Se ordena el vector de enteros aleatorios por el valor de la clave aleatoria de forma ascendente.

Decodificación de los vectores en soluciones validas del problema

- Se ordena el vector de enteros aleatorios por el valor de la clave aleatoria de forma ascendente.

Ejemplo del vector de enteros aleatorios ordenado.

Key	7	13	21	27	45	54	79	98
ClientId	6	2	5	1	4	8	3	7

Decodificación de los vectores en soluciones validas del problema

- Se ordena el vector de enteros aleatorios por el valor de la clave aleatoria de forma ascendente.

Ejemplo del vector de enteros aleatorios ordenado.

Key	7	13	21	27	45	54	79	98
ClientId	6	2	5	1	4	8	3	7

- Implementé dos decodificadores cada uno con su estrategia para generar soluciones.

Decodificación de los vectores en soluciones validas del problema

- Se ordena el vector de enteros aleatorios por el valor de la clave aleatoria de forma ascendente.

Ejemplo del vector de enteros aleatorios ordenado.

Key	7	13	21	27	45	54	79	98
ClientId	6	2	5	1	4	8	3	7

- Implementé dos decodificadores cada uno con su estrategia para generar soluciones.
- Ambos decodificadores generan una solución válidas del problema a partir de un vector de enteros aleatorios ordenado.

Decodificador Simple

- Los vehículos están ordenados de forma ascendente según su identificador.

Decodificador Simple

- Los vehículos están ordenados de forma ascendente según su identificador.
- Toma el primer cliente e intenta agregarlo en la ruta del primer vehículo disponible.

Decodificador Simple

- Los vehículos están ordenados de forma ascendente según su identificador.
- Toma el primer cliente e intenta agregarlo en la ruta del primer vehículo disponible.
- Si logra insertarlo repite el proceso con el siguiente cliente para el mismo vehículo.

Decodificador Simple

- Los vehículos están ordenados de forma ascendente según su identificador.
- Toma el primer cliente e intenta agregarlo en la ruta del primer vehículo disponible.
- Si logra insertarlo repite el proceso con el siguiente cliente para el mismo vehículo.
- Si no lo logra, considera que la ruta del vehículo actual esta completa e intenta agregar el mismo cliente en el siguiente vehículo disponible.

Decodificador Simple

- Los vehículos están ordenados de forma ascendente según su identificador.
- Toma el primer cliente e intenta agregarlo en la ruta del primer vehículo disponible.
- Si logra insertarlo repite el proceso con el siguiente cliente para el mismo vehículo.
- Si no lo logra, considera que la ruta del vehículo actual esta completa e intenta agregar el mismo cliente en el siguiente vehículo disponible.
- Repite hasta completar la ruta de todos los vehículos disponibles.

Decodificador Simple

- Los vehículos están ordenados de forma ascendente según su identificador.
- Toma el primer cliente e intenta agregarlo en la ruta del primer vehículo disponible.
- Si logra insertarlo repite el proceso con el siguiente cliente para el mismo vehículo.
- Si no lo logra, considera que la ruta del vehículo actual esta completa e intenta agregar el mismo cliente en el siguiente vehículo disponible.
- Repite hasta completar la ruta de todos los vehículos disponibles.

Ejemplo de la solución generada por el decodificador simple

Key	7	13	21	27	45	54	79	89
ClientId	6	2	5	1	4	8	3	7

Vehículo 1: 6 -> 2
Vehículo 2: 5 -> 1

Decodificador Goloso

- Se diferencia del decodificador simple en el momento en que encuentra un cliente que no entra en la ruta del vehículo actual.

Decodificador Goloso

- Se diferencia del decodificador simple en el momento en que encuentra un cliente que no entra en la ruta del vehículo actual.
- En vez de pasar al siguiente vehículo, prueba con el siguiente cliente.

Decodificador Goloso

- Se diferencia del decodificador simple en el momento en que encuentra un cliente que no entra en la ruta del vehículo actual.
- En vez de pasar al siguiente vehículo, prueba con el siguiente cliente.
- Por lo tanto por cada vehículo prueba todos los clientes en el orden dado.

Decodificador Goloso

- Se diferencia del decodificador simple en el momento en que encuentra un cliente que no entra en la ruta del vehículo actual.
- En vez de pasar al siguiente vehículo, prueba con el siguiente cliente.
- Por lo tanto por cada vehículo prueba todos los clientes en el orden dado.
- No prueba con los clientes que ya fueron asignados a otro vehículo

Decodificador Goloso

- Se diferencia del decodificador simple en el momento en que encuentra un cliente que no entra en la ruta del vehículo actual.
- En vez de pasar al siguiente vehículo, prueba con el siguiente cliente.
- Por lo tanto por cada vehículo prueba todos los clientes en el orden dado.
- No prueba con los clientes que ya fueron asignados a otro vehículo

Ejemplo de la solución generada por el decodificador goloso

Key	7	13	21	27	45	54	79	89
ClientId	6	2	5	1	4	8	3	7

Vehículo 1: 6 -> 2 -> 8
Vehículo 2: 5 -> 1 -> 3

Descripción de las columnas 1

- *Instancia*: Nombre de la instancia utilizada.
- $N/V/D$: Cantidad de **N**odos / Cantidad de **V**ehículos / **D**istancia máxima de la ruta del vehículo
- *Config*: La configuración utilizada de mi BRKGA al ejecutar la prueba. Es un código que sintetiza la configuración global del algoritmo.
- T_{avg}: El **T**iempo promedio en milisegundos de la ejecución del algoritmo para la instancia mencionada sobre n ejecuciones.
- D: El **D**ecodificador utilizado.

Descripción de las columnas 2

- B_{max}: El **B**eneficio máximo que obtuve para la instancia mencionada sobre n ejecuciones.
- B_{min}: El **B**eneficio mínimo que obtuve para la instancia mencionada sobre n ejecuciones.
- B_{avg}: El **B**eneficio promedio que obtuve para la instancia mencionada sobre n ejecuciones.
- i_{eMax}: Indice de efectividad máximo. $i_{eMax} = B_{max}/Best$
- i_{eAvg}: Indice de efectividad promedio. $i_{eAvg} = B_{avg}/Best$
- $Best$: Máximo beneficio obtenido por algún trabajo previo sobre la misma instancia mencionada.

Comparación entre Decodificadores

- Se realizó una prueba para comparar los tiempos de ejecución y aptitud de las soluciones generadas por ambos decodificadores.

Comparación entre Decodificadores

- Se realizó una prueba para comparar los tiempos de ejecución y aptitud de las soluciones generadas por ambos decodificadores.
- Se crearon 200 vectores de enteros aleatorios y se decodificaron utilizando ambos decodificadores.

Comparación entre Decodificadores

- Se realizó una prueba para comparar los tiempos de ejecución y aptitud de las soluciones generadas por ambos decodificadores.
- Se crearon 200 vectores de enteros aleatorios y se decodificaron utilizando ambos decodificadores.

Instancia	N/V/D	D	B_{min}	B_{avg}	B_{max}	i_{eAvg}	*Best*
p2.2.k	21/2/22.50	S	50	101	170	0.37	275
p2.2.k	21/2/22.50	G	105	166	230	0.60	275
p2.3.g	21/3/10.70	S	45	84	140	0.58	145
p2.3.g	21/3/10.70	G	90	122	145	0.84	145
p5.3.x	66/3/40.00	S	195	301	425	0.19	1555
p5.3.x	66/3/40.00	G	305	412	560	0.26	1555
p7.2.e	102/2/50.00	S	8	39	113	0.13	290
p7.2.e	102/2/50.00	G	37	95	171	0.33	290
p7.4.t	102/4/100.00	S	38	114	238	0.11	1077
p7.4.t	102/4/100.00	G	165	273	439	0.25	1077

Condición de parada

- Cantidad de iteraciones.

Condición de parada

- Cantidad de iteraciones.
- Últimas X generaciónes sin que haya mejorado el beneficio de la mejor solución.

Condición de parada

- Cantidad de iteraciones.
- Últimas X generaciónes sin que haya mejorado el beneficio de la mejor solución.
- Ambas condiciones deben ser válidas a la vez para que termine el algoritmo.

Evolución de la Población

- Se ordenan los soluciones por su aptitud física. (vectores decodificados)

Evolución de la Población

- Se ordenan los soluciones por su aptitud física. (vectores decodificados)
- Las mejores X soluciones se agregan al conjunto elite y pasan directamente a la siguiente generación.

Evolución de la Población

- Se ordenan los soluciones por su aptitud física. (vectores decodificados)
- Las mejores X soluciones se agregan al conjunto elite y pasan directamente a la siguiente generación.
- El resto de las $\#Poblacion - X$ soluciones, se agregan al conjunto no elite.

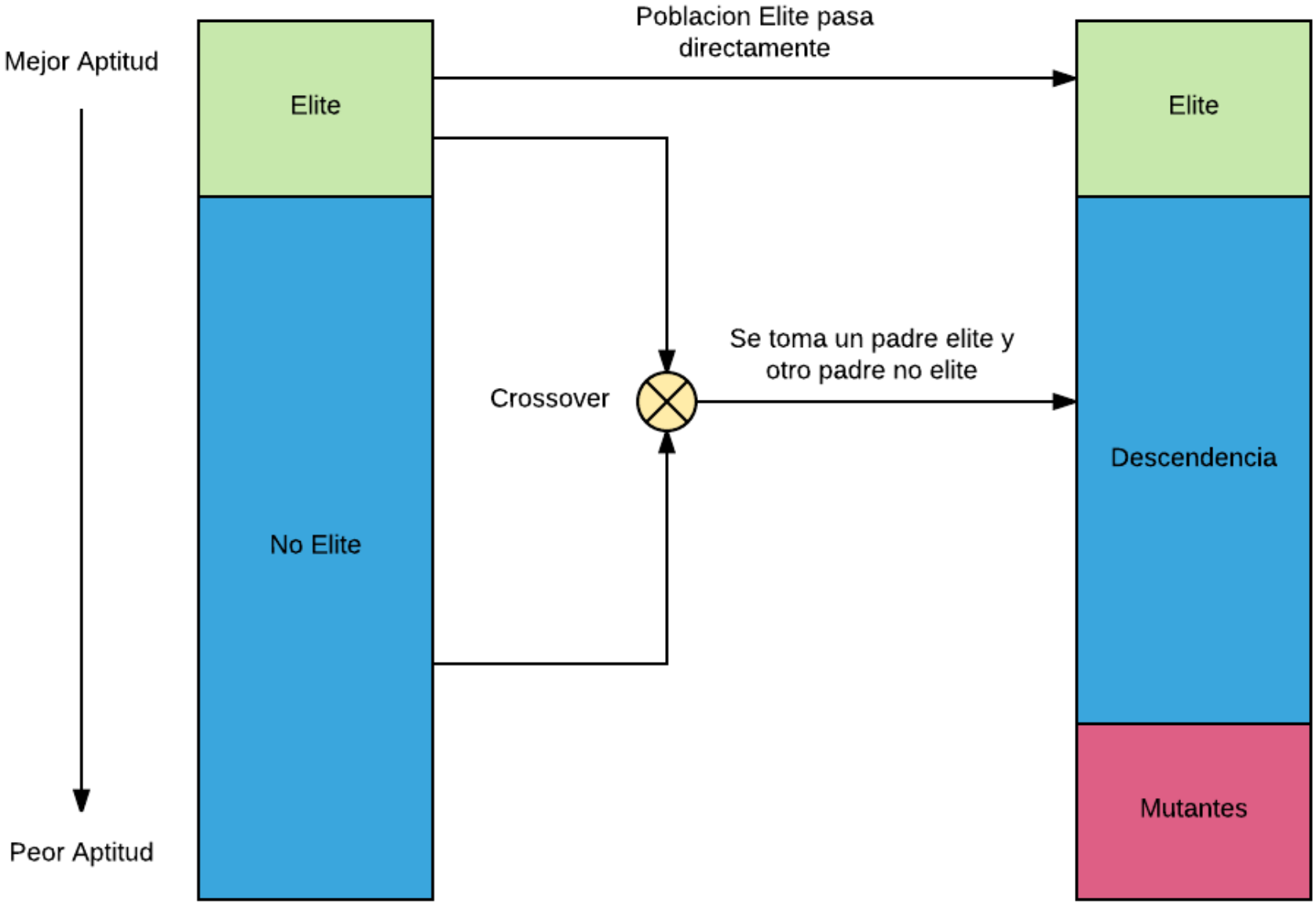
Evolución de la Población

- Se ordenan los soluciones por su aptitud física. (vectores decodificados)
- Las mejores X soluciones se agregan al conjunto elite y pasan directamente a la siguiente generación.
- El resto de las $\#Poblacion - X$ soluciones, se agregan al conjunto no elite.
- Se generan Y soluciones mutantes del mismo modo que se generaron las soluciones iniciales que se agregan a la nueva generación.

Evolución de la Población

- Se ordenan los soluciones por su aptitud física. (vectores decodificados)
- Las mejores X soluciones se agregan al conjunto elite y pasan directamente a la siguiente generación.
- El resto de las $\#Poblacion - X$ soluciones, se agregan al conjunto no elite.
- Se generan Y soluciones mutantes del mismo modo que se generaron las soluciones iniciales que se agregan a la nueva generación.
- Se completa la nueva generación mediante el proceso de *crossover*. Proceso por el cual a partir de dos individuos se genera un tercer individuo.

Evolución de la Población



Crossover

- Se toma un cromosoma del conjunto de elite y otro del conjunto de no elite.

Crossover

- Se toma un cromosoma del conjunto de elite y otro del conjunto de no elite.
- Se crea un vector de números reales al azar en el intervalo [0, 1] del mismo tamaño que los cromosomas.

Crossover

- Se toma un cromosoma del conjunto de elite y otro del conjunto de no elite.
- Se crea un vector de números reales al azar en el intervalo [0, 1] del mismo tamaño que los cromosomas.
- Si el valor en la posición i del vector de números reales es menor a ρ_e, el cromosoma hijo hereda el alelo de la posición i del padre elite. En caso contrario el cromosoma hijo hereda el alelo de la posición i del padre no elite.

Crossover

- Se toma un cromosoma del conjunto de elite y otro del conjunto de no elite.
- Se crea un vector de números reales al azar en el intervalo [0, 1] del mismo tamaño que los cromosomas.
- Si el valor en la posición i del vector de números reales es menor a ρ_e, el cromosoma hijo hereda el alelo de la posición i del padre elite. En caso contrario el cromosoma hijo hereda el alelo de la posición i del padre no elite.
- El valor de ρ_e suele ser 0.70, favoreciendo los alelos del cromosoma elite.

Crossover

- Se toma un cromosoma del conjunto de elite y otro del conjunto de no elite.
- Se crea un vector de números reales al azar en el intervalo [0, 1] del mismo tamaño que los cromosomas.
- Si el valor en la posición i del vector de números reales es menor a ρ_e, el cromosoma hijo hereda el alelo de la posición i del padre elite. En caso contrario el cromosoma hijo hereda el alelo de la posición i del padre no elite.
- El valor de ρ_e suele ser 0.70, favoreciendo los alelos del cromosoma elite.

Crossover


Cromosoma Elite
0.17
0.48
0.25
0.89
0.68
Cromosoma No Elite
0.28
0.72
0.53
0.61
0.11
Números al Azar
0.73
0.41
0.57
0.11
0.81
Relación (0.70)
>
<
<
<
>
Cromosoma Resultante
0.28
0.48
0.25
0.89
0.11

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.
- Tener individuos repetidos en la población reduce el dominio de soluciones exploradas.

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.
- Tener individuos repetidos en la población reduce el dominio de soluciones exploradas.
- Tener individuos repetidos en la población genera cálculos repetidos.

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.
- Tener individuos repetidos en la población reduce el dominio de soluciones exploradas.
- Tener individuos repetidos en la población genera cálculos repetidos.
- Un individuo se inserta en la población si no existe algún individuo en la población con el mismo valor de hash.

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.
- Tener individuos repetidos en la población reduce el dominio de soluciones exploradas.
- Tener individuos repetidos en la población genera cálculos repetidos.
- Un individuo se inserta en la población si no existe algún individuo en la población con el mismo valor de hash.
- Se implementaron dos formas de calcular el hash de un individuo.

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.
- Tener individuos repetidos en la población reduce el dominio de soluciones exploradas.
- Tener individuos repetidos en la población genera cálculos repetidos.
- Un individuo se inserta en la población si no existe algún individuo en la población con el mismo valor de hash.
- Se implementaron dos formas de calcular el hash de un individuo.
- La primer forma que implemente calcula el hash sin conocer el problema que se esta resolviendo, luego mantiene la evolución de la población independiente del problema que se esta resolviendo.

Hash de un individuo

- Dos individuos son iguales si representan la misma solución.
- Tener individuos repetidos en la población reduce el dominio de soluciones exploradas.
- Tener individuos repetidos en la población genera cálculos repetidos.
- Un individuo se inserta en la población si no existe algún individuo en la población con el mismo valor de hash.
- Se implementaron dos formas de calcular el hash de un individuo.
- La primer forma que implemente calcula el hash sin conocer el problema que se esta resolviendo, luego mantiene la evolución de la población independiente del problema que se esta resolviendo.
- La segunda forma de calcular es optimo detectando repetidos y requiere conocer el problema que se esta resolviendo.

Método independiente del problema para calcular el hash

- Se toma el vector de claves aleatorias del individuo y se lo ordena por su clave aleatoria. El valor de su hash es la concatenación de los identificadores de clientes separados con un símbolo.

Método independiente del problema para calcular el hash

- Se toma el vector de claves aleatorias del individuo y se lo ordena por su clave aleatoria. El valor de su hash es la concatenación de los identificadores de clientes separados con un símbolo.

Key	27	13	79	45	21	7	98	54
ClientId	1	2	3	4	5	6	7	8

Key	7	13	21	27	45	54	79	98
ClientId	6	2	5	1	4	8	3	7

- Hash resultante del vector de enteros de la imagen: 6@2@5@1@4@8@3@7

Método óptimo para calcular el hash

- Se toman los recorridos de los vehículos y se los ordena por el identificador del primer cliente que visitan.

Método óptimo para calcular el hash

- Se toman los recorridos de los vehículos y se los ordena por el identificador del primer cliente que visitan.
- Se calcula el hash de cada vehículo y se concatenan todos los hash utilizando otro símbolo separador.

Vector de enteros aleatorios ordenado y la solución en la que decodifica.

Key	7	13	21	27	45	54	79	89
ClientId	6	2	5	1	4	8	3	7

Vehículo 1: 6 -> 2
Vehículo 2: 5 -> 1

- Hash resultante de la solución de la imagen: 5@1#6@2

Método óptimo para calcular el hash

- Se toman los recorridos de los vehículos y se los ordena por el identificador del primer cliente que visitan.
- Se calcula el hash de cada vehículo y se concatenan todos los hash utilizando otro símbolo separador.

Vector de enteros aleatorios ordenado y la solución en la que decodifica.

Key	7	13	21	27	45	54	79	89
ClientId	6	2	5	1	4	8	3	7

Vehículo 1: 6 -> 2
Vehículo 2: 5 -> 1

- Hash resultante de la solución de la imagen: 5@1#6@2
- Si cambiamos la posición de los ClientId 8, 3 y 7 obtenemos la misma solución al decodificarla. El hash con este método no cambia y el hash con el método anterior cambia.

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).
- Cantidad de iteraciones sin cambios en la mejor solución. (Entre 50 y 100).

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).
- Cantidad de iteraciones sin cambios en la mejor solución. (Entre 50 y 100).
- Tamaño de la población. (Entre 100 y 200).

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).
- Cantidad de iteraciones sin cambios en la mejor solución. (Entre 50 y 100).
- Tamaño de la población. (Entre 100 y 200).
- Porcentaje de la poblacion elite. (Entre el 20% y el 30%)

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).
- Cantidad de iteraciones sin cambios en la mejor solución. (Entre 50 y 100).
- Tamaño de la población. (Entre 100 y 200).
- Porcentaje de la poblacion elite. (Entre el 20% y el 30%)
- Porcentaje de mutantes. (Entre el 5% y el 10%)

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).
- Cantidad de iteraciones sin cambios en la mejor solución. (Entre 50 y 100).
- Tamaño de la población. (Entre 100 y 200).
- Porcentaje de la poblacion elite. (Entre el 20% y el 30%)
- Porcentaje de mutantes. (Entre el 5% y el 10%)
- Probabilidad de heredar alelo del padre elite ($\rho_e \in [0.5, 0.7]$)

Pruebas sobre el BRKGA sin Búsqueda Local

- Se realizaron varias pruebas sobre el BRKGA modificando las variables de su configuración. Tales variables:
- Iteraciones. (Entre 200 y 300).
- Cantidad de iteraciones sin cambios en la mejor solución. (Entre 50 y 100).
- Tamaño de la población. (Entre 100 y 200).
- Porcentaje de la poblacion elite. (Entre el 20% y el 30%)
- Porcentaje de mutantes. (Entre el 5% y el 10%)
- Probabilidad de heredar alelo del padre elite ($\rho_e \in [0.5, 0.7]$)
- Tipo de decodificador. (El simple o el goloso).

La configuración que mejor resulto para el BRKGA sin Búsqueda Local

- Luego de varias pruebas, evaluando los resultados obtenidos y el tiempo de ejecución, la configuración que mejores resultados obtenía era:

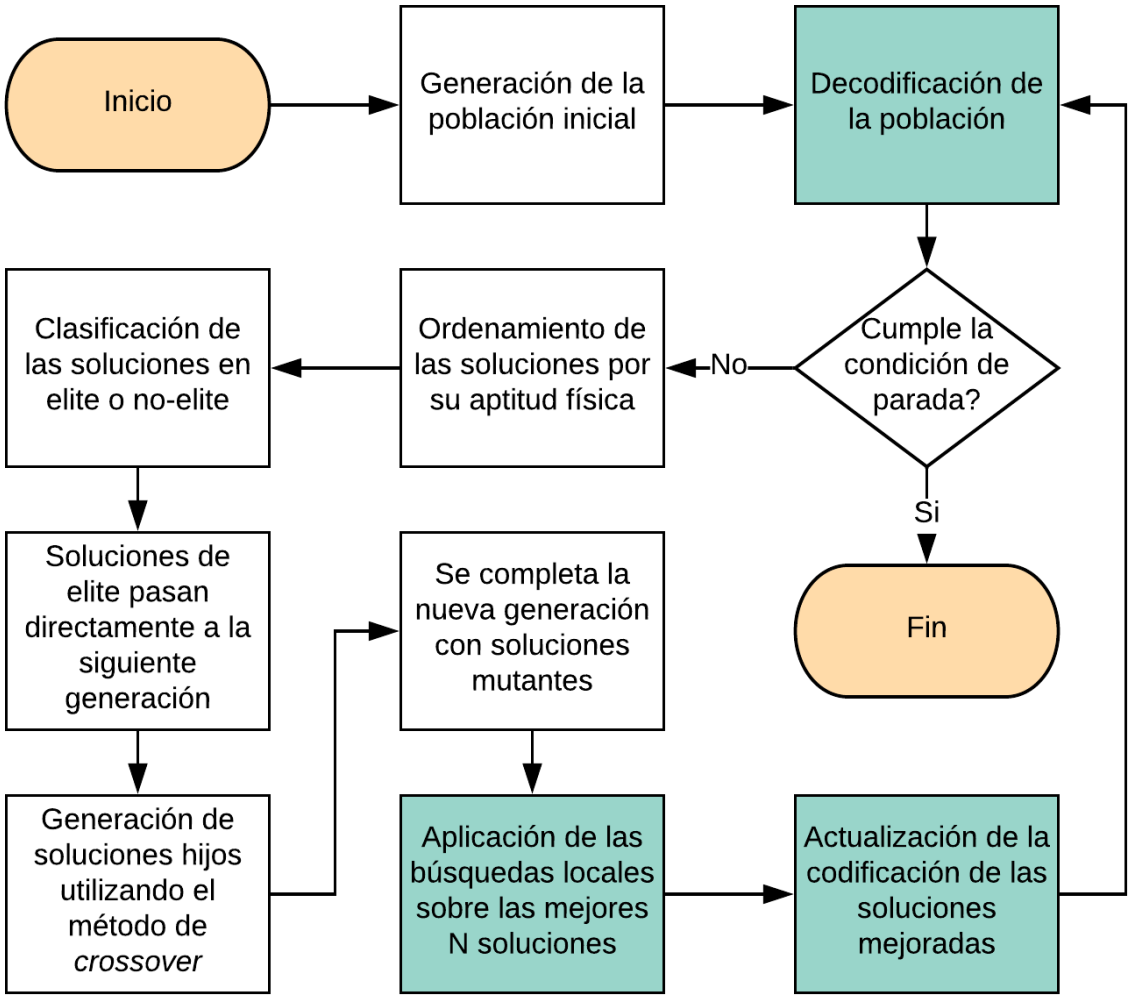
La configuración que mejor resulto para el BRKGA sin Búsqueda Local

- Luego de varias pruebas, evaluando los resultados obtenidos y el tiempo de ejecución, la configuración que mejores resultados obtenía era:
- Iteraciones: 250; Sin cambios: 50; Población.100; Porcentaje elite: 30%; Porcentaje mutante: 10%; ρ_e: 0.70; Deco: Goloso.

La configuración que mejor resulto para el BRKGA sin Búsqueda Local

- Luego de varias pruebas, evaluando los resultados obtenidos y el tiempo de ejecución, la configuración que mejores resultados obtenía era:
- Iteraciones: 250; Sin cambios: 50; Población.100; Porcentaje elite: 30%; Porcentaje mutante: 10%; ρ_e: 0.70; Deco: Goloso.

Flow Chart del BRKGA con Búsqueda Local



Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Flow Chart del BRKGA con Búsqueda Local

Paragraphs of Text

Sed iaculis dapibus gravida. Morbi sed tortor erat, nec interdum arcu. Sed id lorem lectus. Quisque viverra augue id sem ornare non aliquam nibh tristique. Aenean in ligula nisl. Nulla sed tellus ipsum. Donec vestibulum ligula non lorem vulputate fermentum accumsan neque mollis.

Sed diam enim, sagittis nec condimentum sit amet, ullamcorper sit amet libero. Aliquam vel dui orci, a porta odio. Nullam id suscipit ipsum. Aenean lobortis commodo sem, ut commodo leo gravida vitae. Pellentesque vehicula ante iaculis arcu pretium rutrum eget sit amet purus. Integer ornare nulla quis neque ultrices lobortis. Vestibulum ultrices tincidunt libero, quis commodo erat ullamcorper id.

Bullet Points

- Lorem ipsum dolor sit amet, consectetur adipiscing elit
- Aliquam blandit faucibus nisi, sit amet dapibus enim tempus eu
- Nulla commodo, erat quis gravida posuere, elit lacus lobortis est, quis porttitor odio mauris at libero
- Nam cursus est eget velit posuere pellentesque
- Vestibulum faucibus velit a augue condimentum quis convallis nulla gravida

Blocks of Highlighted Text

Block 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Block 2

Pellentesque sed tellus purus. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Vestibulum quis magna at risus dictum tempor eu vitae velit.

Block 3

Suspendisse tincidunt sagittis gravida. Curabitur condimentum, enim sed venenatis rutrum, ipsum neque consectetur orci, sed blandit justo nisi ac lacus.

Multiple Columns

Heading

1. Statement
2. Explanation
3. Example

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer lectus nisl, ultricies in feugiat rutrum, porttitor sit amet augue. Aliquam ut tortor mauris. Sed volutpat ante purus, quis accumsan dolor.

Table

Treatments	**Response 1**	**Response 2**
Treatment 1	0.0003262	0.562
Treatment 2	0.0015681	0.910
Treatment 3	0.0009271	0.296

Table caption

Theorem

Theorem (Mass–energy equivalence)

$E = mc^2$

Verbatim

Example (Theorem Slide Code)

```
\begin{frame}
\frametitle{Theorem}
\begin{theorem}[Mass--energy equivalence]
$E = mc^2$
\end{theorem}
\end{frame}
```

Figure

Uncomment the code on this slide to include your own image from the same directory as the template .TeX file.

Citation

An example of the `\cite` command to cite within the presentation:

This statement requires citation [Smith, 2012].

References

John Smith (2012)

Title of the publication

Journal Name 12(3), 45 – 678.

The End