

# OPTIMIZACIÓN DE REPARTO DE PRODUCTOS UTILIZANDO ALGORITMOS MEMÉTICOS

Cabrera Cabral, Priscila Beatriz

Tutor: Morel Peralta, Sergio Enrique

priscilabtz@gmail.com

Facultad Politécnica

Universidad Nacional del Este

## Resumen

El presente trabajo consiste en la utilización de algoritmos meméticos, para optimización de reparto de productos. Se tiene como objetivo principal maximizar las ganancias sin dejar de atender el estado de conservación de los productos entregados. Se trata siempre de cubrir la demanda de los clientes dentro de un tiempo razonable.

Como parte del problema, se contemplan escenarios que pueden contar con varios vehículos y localidades de reabastecimiento para la carga de productos. Los vehículos parten de alguna localidad para realizar los recorridos de entrega de productos y luego regresan a sus respectivas localidades de origen.

El algoritmo busca encontrar los mejores recorridos para cada vehículo de tal forma a conciliar los diferentes objetivos del problema. Se asegura que no se generen soluciones no representables en la realidad, por medio de optimizaciones locales. Dichas optimizaciones son aplicadas después de realizar las operaciones evolutivas de mutación y recombinación.

Se logró obtener recorridos viables en los cuales se optimizaron los diferentes objetivos del problema de reparto de productos.

**Palabras clave:** algoritmo memético, optimización, reparto de productos.

## 1. Introducción

Los algoritmos meméticos (Memetic Algorithms- MAs) son algoritmos evolutivos (Evolutionary Algorithms - EAs) que aplican un proceso de búsqueda local separada para mejorar a los individuos, es decir, mejorar su aptitud (*fitness*). Estos métodos están inspirados en modelos de adaptación de los sistemas naturales que combinan la adaptación evolutiva de las poblaciones de individuos con problemas de aprendizaje individual dentro de un periodo. Bajo diferentes contextos y situaciones, los MAs son también conocidos como EAs híbridos, buscadores locales genéticos, EA baldwiniana, EAs lamarckianos, etc. Este método se basa en una población de agentes y demostró éxito práctico en una variedad de dominios de problemas.

## 2. Objetivos

### Objetivo general

Desarrollar una aplicación capaz de optimizar de forma el reparto de productos utilizando algoritmos meméticos.

### Objetivos específicos

- 1) Identificar las restricciones que presenta el problema a tratar.
- 2) Identificar el espacio de soluciones del problema.

- 3) Definir representación de solución del algoritmo evolutivo.

- 4) Generar algoritmo para calcular calificación de dicha representación.

- 5) Generar algoritmos a ser aplicados a dicha representación para las operaciones evolutivas.

- 6) Generar algoritmos de optimización local a ser aplicados a las operaciones evolutivas.

- 7) Controlar generación de soluciones factibles mediante optimización local.

- 8) Codificar aplicación informática para automatizar los algoritmos generados.

## 3. Materiales y Métodos

Se han aplicado algoritmos meméticos para resolver el problema propuesto. El objetivo del algoritmo consiste en maximizar eficiencia por medio de métodos de optimización metaheurísticos.

Los MAs son conocidos por la optimización local que realizan en las diferentes fases del ciclo evolutivo. El método utilizado para aplicar MAs, al problema de reparto de productos consta de los procedimientos descritos a continuación. La aplicación se ha desarrollado en entorno de programación Java, utilizando la base de datos Postgres.

### 3.1 Representación del agente

Los agentes representan soluciones candidatas al problema propuesto. Una solución debe contener los recorridos de los vehículos y los productos que reparten en cada trayecto.

La figura 1 muestra un ejemplo de recorrido de un vehículo, indicando sus trayectos.

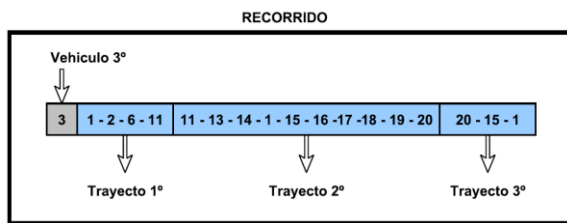


Figura 1: Ejemplificación de recorrido de un vehículo.

El cromosoma está estructurado por un conjunto de recorridos. La cantidad máxima de recorridos es proporcional a la cantidad de vehículos existentes.

La figura 2 muestra la estructura de un cromosoma.

1	5	20	11	20	1	20	5	
2	1	8	16	8	12	11	12	1
3	1	16	18	3	8	1		

Figura 2: Estructura de cromosoma implementado.

Para generar el agente hay dos algoritmos principales:

1) Algoritmo que genera trayecto entre dos locales de interés. Para generar

trayecto se aplica un método basado en árbol de búsqueda.

2) Algoritmo que genera recorrido de vehículo combinando con movimiento de producto, observando restricciones tales como stock de productos disponibles, capacidad de vehículo, demanda de cliente, etc.

### 3.2. Calificación del agente

El algoritmo empleado aborda la calificación de agente como si se tratase de un problema mono-objetivo. Los objetivos tenidos en cuenta para la calificación mono-objetivo son: el lucro, el tiempo de recorrido, el porcentaje de pedidos entregados y el porcentaje de conservación de productos entregados.

El lucro ( $L_c$ ) se calcula teniendo en cuenta la diferencia entre el ingreso por ventas ( $Ing_{vta}$ ) y la suma de los costos de recorrido ( $Costo_{rec}$ ) y de productos ( $Costo_{pro}$ ).

$$L_c = Ing_{vta} - (Costo_{rec} + Costo_{pro})$$

Donde:

$Ing_{vta}$ : ingreso por venta, se obtiene a través de la sumatoria del producto cantidad entregada ( $Cant_{ent}$ ) por precio de venta ( $Pre_{vta}$ ) del producto:

$$Ing_{vta} = \sum(Cant_{ent} \times Pre_{vta}).$$

**Costo<sub>rec</sub>**: costo de recorrido, se obtiene sumando gasto de combustible (**Gto<sub>comb</sub>**) más gasto de mantenimiento (**Gto<sub>mant</sub>**) más costos de reparación (**Costo<sub>rep</sub>**):

$$Costo_{rec} = Gto_{comb} + Gto_{mant} + Costo_{rep}$$

**Costo<sub>pro</sub>**: costo de producto, se calcula como la sumatoria del precio de productos (**Precio<sub>pro</sub>**) dividida entre cantidad (**Cant<sub>pro</sub>**) de los mismos.

$$Costo_{pro} = \frac{(\sum Precio_{pro})}{Cant_{pro}}$$

El tiempo de recorrido **Tiempo<sub>rec</sub>** se calcula como la sumatoria del tiempo de recorrido entre los enlaces (**TR<sub>eni</sub>**) de todos los trayectos que conforman los recorridos:

$$Tiempo_{rec} = \sum TR_{eni}$$

El porcentaje de pedidos entregados (**Porc<sub>ent</sub>**) se calcula como la división entre las sumatorias de cantidad entregada (**Cant<sub>ent</sub>**) y de cantidad pedida de productos (**Cant<sub>ped</sub>**).

Este resultado se multiplica por 100 para obtener el resultado porcentual.

$$Porc_{ent} = \left( \frac{\sum Cant_{ent}}{\sum Cant_{ped}} \right) \times 100$$

El valor utilizado como objetivo de la solución (**C<sub>final</sub>**), se calcula realizando el promedio entre el menor valor de conservación y el promedio del valor de conservación.

$$C_{final} = \left( \frac{PP_{cons} \text{ MIN}(PCon_{pro})}{2} \right) \times 100$$

Para hallar la proporción de conservación del producto (**PCon<sub>pro</sub>**), primeramente se debe calcular el tiempo de vida restante (**TV<sub>rest</sub>**) y el tiempo de vida original (**TV<sub>orig</sub>**) de los productos.

$$PCon_{pro} = \frac{TV_{rest}}{TV_{orig}}$$

El (**TV<sub>rest</sub>**), se refiere a cuanto falta para que el producto venza, en el momento de entrega al cliente. El (**TV<sub>orig</sub>**), se refiere a cuanto falta para que el producto venza, en el momento de envío al cliente.

Luego, se calcula la suma ponderada de proporciones (**SP<sub>prop</sub>**), que equivale a la sumatoria del producto de la proporción de

conservación de los productos ( $PCon_{pro}$ ) por la cantidad entregada ( $Cant_{ent}$ ) de los mismos.

$$SP_{prop} = \sum (PCon_{pro} \times Cant_{ent})$$

Seguidamente, se calcula el promedio de proporción de conservación ( $PP_{cons}$ ) que es igual al resultado de la suma ponderada de las proporciones ( $SP_{prop}$ ) dividido la cantidad total de ítems entregados ( $CantT_{ent}$ ).

$$PP_{cons} = \frac{SP_{prop}}{CantT_{ent}}$$

Se calcula el valor de la calificación mono-objetivo ( $Calif_{mono}$ ) que se asocia a cada agente con la siguiente definición:

$$Calif_{mono} = \sum_{i=0}^n (Obj_i \times Peso_i)$$

Este cálculo realiza la sumatoria de todos los objetivos del problema ( $Obj_i$ ), mencionados más arriba, multiplicados por sus ( $peso_i$ ) respectivos.

### Normalización de objetivos

La normalización de los objetivos tiene como función evitar que los mismos afecten a la calificación final en proporciones diferentes.

Realizando la normalización de objetivos, se pueden obtener valores en un mismo rango, los cuáles pueden ser comparados equitativamente.

Para el problema de reparto de productos se decidió normalizar todos los objetivos, con valores de 0 a 100. Dentro de la normalización también se tiene en cuenta cuáles son los objetivos que se desean maximizar o minimizar, de forma que la calificación final del agente se pueda calcular con una suma directa de todos los objetivos normalizados.

### 3.3. Recombinación

En los MAs se utiliza la recombinación como un operador que combina el cromosoma de dos agentes y obtiene un nuevo agente con el cromosoma combinado.

En este trabajo la recombinación se realizó con puntos de corte aleatorios diferentes para cada recorrido de cada vehículo (figura 3).



Figura 3: Ejemplo de recombinación y sus puntos de corte.

En la recombinación solo se realiza combinaciones entre recorridos del mismo

vehículo, para evitar problemas de caminos no existentes entre dos enlaces.

### 3.4. Mutación

La mutación consiste en aplicar operaciones de Eliminación de localidad y Regeneración de trayectos a cada agente.

El operador de Eliminación de localidad selecciona aleatoriamente una localidad del recorrido y la elimina. En la figura 4 se muestra un ejemplo en donde se obtiene un agente mutado a partir de la eliminación de una localidad del agente original.



Figura 4: Eliminación de localidad.

En el operador de Regeneración de trayectos, la localidad inicial y final de los trayectos se mantienen constantes, pero el orden de los enlaces de dicho trayecto se vuelve a generar en forma aleatoria.

La figura 5 muestra un ejemplo de regeneración de trayecto.

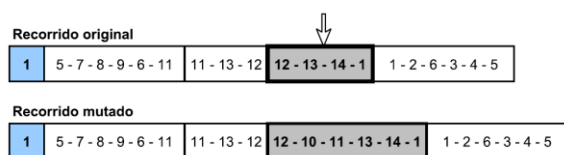


Figura 5: Regeneración de trayecto.

### 3.5. Optimización local

Las operaciones evolutivas que se utilizan pueden dar como resultado agentes que no representen solución posible. Para solucionar este inconveniente se realiza correcciones al agente resultante. Además hay situaciones en las que el agente resultante es mejorado, evitando movimientos innecesarios.

Para el problema tratado se realizan optimizaciones locales luego de las operaciones de mutación y de recombinación.

La optimización local realizada después del operador mutación tiene en cuenta dos casos. El primer caso se encarga de realizar las modificaciones cuando la localidad eliminada corresponde a una localidad de entrega de pedidos.

El segundo caso se encarga de realizar la optimización cuando la localidad eliminada corresponde a una localidad de reabastecimiento de productos.

#### Eliminación de localidad de entrega.

En la figura 6 se puede observar que la localidad a ser eliminada es la 20, que a su vez elimina la localidad 1.

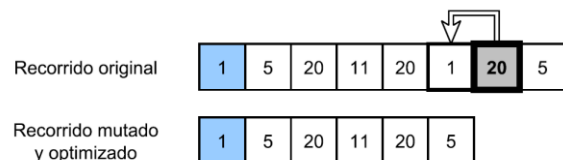


Figura 6: Optimización de eliminación de localidad de entrega.

Esto ocurre debido a que la localidad 1 provee productos únicamente para la localidad eliminada, por lo tanto, dicha localidad ya no será necesaria.

### Eliminación de localidad de reabastecimiento.

En la figura 7 la localidad 1 es eliminada, corresponde a una localidad de reabastecimiento de productos.

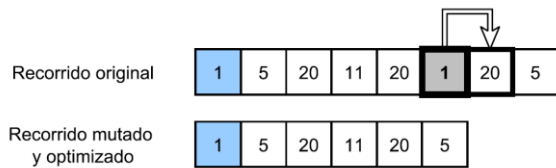


Figura 7: Optimización por eliminación de localidad de reabastecimiento.

Por tal motivo, al ser eliminada la localidad de reabastecimiento 1, la localidad de entrega 20 ya no cuenta con productos que puedan satisfacer su pedido, entonces se elimina y ya no será incluida en el recorrido.

### Recombinación

La figura 8 muestra un ejemplo de optimización luego de la recombinación de dos agentes. Se puede observar que luego de combinar los cromosomas queda una localidad que debe ser eliminada.

Luego de la eliminación de dicha localidad se tienen dos localidades iguales adyacentes, lo cual implica un viaje en círculo sin beneficios. Por este motivo el algoritmo combina el movimiento de ambas localidades y mantiene una sola localidad en el recorrido.

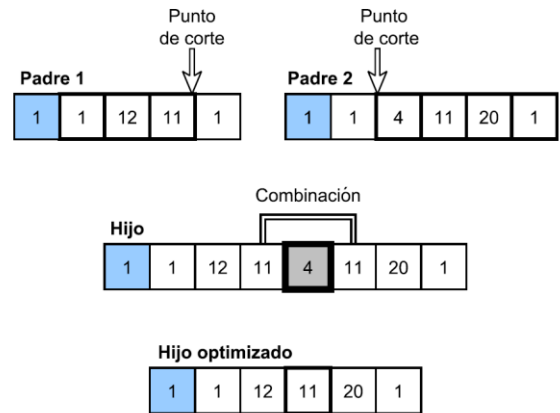


Figura 8: Combinación de dos localidades iguales.

## 4. Resultados y Discusión

El algoritmo genera recorridos que realizan la entrega de los pedidos teniendo en cuenta las restricciones fuertes de coherencia de recorridos y los diferentes objetivos establecidos en la operación de calificación.

Se realizaron ejecuciones del algoritmo para obtener recorridos óptimos de los vehículos. Como entrada se utilizaron datos de prueba almacenados en una base de datos.

Se cuenta con una tabla de localidades, donde cada localidad corresponde a una ciudad del departamento Alto Paraná.

Los datos de localidades se pueden visualizar en la tabla 1.

Vértice	Localidad
1	Ciudad del Este
2	Minga Guazú
3	Yguazú
4	Dr. Juan León Mallorquín
5	Juan Emilio O'Leary
6	Santa Rita
7	San Cristobal
8	Naranjal
9	Iruña
10	Nacunday
11	Santa Rosa del Monday
12	Domingo Martínez de Irala
13	Cedrales
14	Presidente Franco
15	Hernandarias
16	Santa Fe del Paraná
17	Mbaracayú
18	San Alberto
19	Minga Porá
20	Ytakry

Tabla 1: Referencia de localidades.

Entre los datos incluidos se encuentran los productos disponibles en stock y los productos pedidos en las distintas localidades.

Otra información importante es la disponible de los vehículos de reparto. El algoritmo tiene en cuenta la posición inicial de los vehículos y la capacidad de carga de los mismos.

En la figura 9 se puede visualizar el agente con la mejor calificación que se ha obtenido luego de una ejecución. Dicho agente cuenta con 280,07 como valor de calificación.

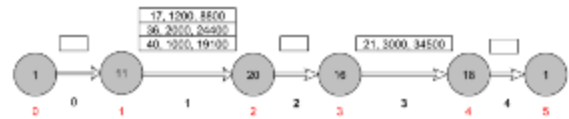


Figura 9: Recorrido del mejor agente.

El agente resultante cuenta 6 vértices que conforman los 5 trayectos del recorrido, cada trayecto inicia y termina en una localidad de interés. El número identificador de cada localidad se indica dentro del círculo. Por ejemplo, la localidad 1 corresponde a Ciudad del Este, de acuerdo a la tabla 1. Los movimientos o los productos pedidos se muestran en los cuadros encima de cada trayecto.

La calificación se obtuvo a partir de los datos en la figura 10.

Objetivos	Valor normal	Valor normalizado
Lucro	18.973.000	88,11
Tiempo de recorrido	742	68,71
Porcentaje de pedidos entregados	12,88	12,88
Porcentaje de conservación de productos entregados	88,37	88,37

Figura 10: Calificación de los objetivos del mejor agente.

## 5. Conclusiones

Se desarrolló un algoritmo memético para la optimización del problema de reparto de productos. Básicamente, el algoritmo se encarga de realizar las operaciones evolutivas, tales como recombinación y



mutación, y luego refinar el resultado a través de la optimización local aplicada al agente.

Entre los logros más relevantes del del algoritmo memético desarrollado para optimizar reparto de productos, se encuentran los siguientes:

- Se han logrado identificar las restricciones fuertes que presenta el problema y se las trató en los algoritmos de generación de agentes y optimizaciones locales.
- Se identificó el espacio de soluciones del problema de reparto, como un conjunto de recorridos compuestos por una cantidad finita de trayectos de acuerdo a las restricciones de caminos disponibles entre localidades.
- Se ha logrado definir la forma de representar las soluciones del problema, según la sección 3.1.
- Se ha logrado definir al algoritmo mono-objetivo para el cálculo de la calificación del agente, según la sección 3.2.
- Se han logrado crear algoritmos individuales para calcular calificación de cada objetivo del problema.
- Se han logrado definir los algoritmos que fueron aplicados al agente para las operaciones evolutivas, según las secciones 3.3 y 3.4.
- Se definió la optimización local a ser realizada después de la operación de recombinación y mutación para asegurar

la integridad de las soluciones obtenidas y evitar trabajar con soluciones no factibles.

## **Bibliografía**

- Wu, F. (2001). A framework for Memetic algorithms (Doctoral dissertation, University of Auckland).
- Moscato, P., &Cotta, C. (2003). Una introducción a los algoritmos meméticos. *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, 7(19), 0.
- Von Lücken, C., Hermosilla, A., &Barán, B. (2004). Algoritmos Evolutivos para Optimización Multiobjetivo: un estudio comparativo en un ambiente paralelo asíncrono. In X Congreso Argentino de Ciencias de la Computación.
- Russell, S., &Norvig, P. (2004). *Inteligencia Artificial Un Enfoque Moderno*.