ICT-UNPA-70-2013
ISSN: 1852 - 4516
Aprobado por Resolución Nro. 0862/13-R-UNPA

Hibridación de Metaheurísticas aplicadas al Problema de Ruteo de Vehículos

Mercado, V.[1] Pandolfi, D.[2] Villagra, A[2].
[1]{Becaria de Investigación Postgrado}
[2]{Docente Investigador UNPA}
{vmercado, dpandolfi, avillagra }@uaco.unpa.edu.ar

UNPA – UACO
Universidad Nacional de la Patagonia Austral – Unidad Académica Caleta Olivia
Departamento de Ciencias Exactas y Naturales
LabTEm- Laboratorio de Tecnologías Emergentes
Caleta Olivia, 2013

RESUMEN

El problema de ruteo de vehículos consiste en hallar un conjunto de rutas óptimas de reparto que permitan satisfacer la demanda de clientes. Desde el punto de vista algorítmico, son problemas de optimización combinatoria de alta complejidad Para esto, se revisa sistemáticamente la literatura generada al respecto en los últimos años respecto a la implementación de algoritmos metaheurísticos que permitan resolver el problema de ruteo de vehículos con capacidad uniforme *(Capacitated Vehicle Routing Problems).* Identificando los enfoques más exitosos, los esfuerzos de unificación de modelos, así como las proyecciones que existen en este campo y la combinación de métodos de resolución aproximados generales (metaheurísticas híbridas) permitirá la generación de métodos más exactos y una línea de investigación fructífera. En este trabajo proponemos un algoritmo para resolver el problema de ruteo de vehículos con capacidad limitada, utilizando como base un Algoritmo Evolutivo conocido como MCMP-SRI *(Stud and Random Inmigrants*) combinado con conceptos de computación cuántica aplicados a la mutación. Además los resultados son comparados con otros dos algoritmos híbridos que utilizan *Hill- Climbing*. Detalles de los algoritmos y los resultados de los experimentos muestran un promisorio comportamiento para resolver el problema.

Palabras Claves: Metaheurísticas, Mejoras en el proceso de búsqueda, Problemas de Optimización, Hibridación, Problema de Ruteo de Vehículos.

1. INTRODUCCIÓN

En la actualidad el problema de distribuir productos a partir de un depósito original (punto de origen) y una cantidad de clientes con una demanda por atender, juega un papel importante en empresas comercializadoras ya que planificar adecuadamente estos envíos puede significar considerables ahorros logísticos y sobretodo en costos como: el consumo de combustible, horas hombre, entre otros; que ayudarán a una mejor rentabilidad para los negocios hoy en día. Son por estas causas, que surge el problema de ruteo de vehículos *(Vehicle Routing Problem* o las siglas en inglés VRP).

Este problema consiste en generar rutas de reparto dado una cantidad de clientes por atender, un conjunto de vehículos de reparto y un punto de origen, permitiendo minimizar ciertos factores que ayuden a la empresa a obtener beneficios; estos pueden ser: minimizar el tiempo de reparto, maximizar el ahorro de combustible en los vehículos, minimizar la cantidad de vehículos de reparto, lo cual llevaría a obtener menores costos y por lo tanto obtener beneficios y una mejor calidad de servicio e imagen [Laporte 1992]. Asimismo, presenta una serie de variantes como es el caso de incluir: la capacidad de un vehículo, espacios de tiempo de entrega, incluir varios puntos de origen (que vendrían a ser los depósitos), entre otros.

Su objetivo principal será minimizar la cantidad de vehículos y el tiempo de viaje (reduciendo así el gasto de combustible, choferes, horas hombre), siempre y cuando se respete que cada reparto no pueda exceder la capacidad que tiene un vehículo [Christofides et al. 1979]. Este tipo de problemas no tienen una solución exacta porque encontrar la ruta mínima entre dos puntos en un mapa que contiene miles de conexiones llevaría a realizar cálculos computacionales muy elevados. Por ello este tipo de problemas son clasificados como problemas de clase NP (Nondeterministic Polynomial) una clase de problema en Teoría de la complejidad computacional. Si bien se pueden resolver, sus soluciones no son exactas sino aproximadas. [Laporte 1992].

Por lo tanto se propone utilizar algoritmos Metaheurísticos, ya que este tipo de algoritmos buscan una posible solución y luego la optimizan, dando así una muy buena solución. Para poder llevar a cabo lo anterior, es menester hacer una revisión sistemática de la literatura respecto a la implementación de algoritmos metaheurísticos que permitan resolver el problema de ruteo de vehículos con capacidad uniforme (*Capacitated Vehicle Routing Problems).*

Identificando los enfoques más exitosos, los esfuerzos de unificación de modelos, así como las proyecciones que existen en este campo y la combinación de métodos de resolución aproximados generales (metaheurísticas híbridas) permitirá la generación de métodos más exactos y una línea de investigación fructífera. Por otro lado, también las investigaciones y las revistas científicas han consolidado el interés en la optimización de metaheurísticas, como así también en las implementaciones paralelas de metaheurísticas. Además, existen diversas tendencias sobre las implementaciones multiobjetivo o problemas que cambian a lo largo del tiempo (problemas dinámicos). En este trabajo se hará hincapié en las técnicas hibridas que cobran cada vez más importancia dentro del campo de las metaheurísticas.

Un problema de optimización se formaliza como un par (S, f), donde S < > 0 representa el espacio de soluciones (o de búsqueda) del problema, mientras que f es un criterio de calidad conocido como función objetivo, definida como:

$$f: S \rightarrow R.$$

Así, resolver un problema de optimización consiste en encontrar un conjunto de valores adecuados de forma que la solución representada por estos valores $i^* \in S$ satisfaga la siguiente desigualdad: $f(i^*) \leq f(i), \ \forall i \in S$

Asumir el caso de maximización o minimización no restringe en ningún caso la generalidad de los resultados, puesto que se puede establecer una igualdad entre tipos de problemas de maximización y minimización [Bäck 1996; Goldberg. 1989].

A este tipo de problemas se los puede dividir en dos categorías [Blum y Roli 2003], [Papadimitriou y Steiglitz 1998]: aquellos en los que la solución está codificada mediante valores reales y aquellos cuya solución está codificada con valores enteros.
En los últimos encontramos los problemas denominados de optimización combinatoria. Algunos ejemplos muy conocidos son el problema del viajante de comercio (TSP - *Travelling Salesman Problem*), el problema de asignación cuadrática (QAP - *Quadratic Assignment Problems*) o los problemas de planificación *(Scheduling Problems)*, entre otros, [Cook et al. 1998; Papadimitriou y Steiglitz 1998].
En la literatura existen multitud de propuestas de técnicas algorítmicas, tanto exactas como aproximadas para resolver problemas de optimización. Los algoritmos exactos garantizan encontrar una solución óptima al problema para toda instancia de tamaño finito [Hochbaum 1996; Korf 1985; Russell y Norvig 1995; Vazirani 2003]. Sin embargo, los métodos exactos necesitan tiempos exponenciales de computación cuando se trata con instancias grandes de problemas complejos. Los problemas NP − completos no tienen un algoritmo en tiempo polinómico que los resuelva. También existe otro tipo de problemas al menos tan difíciles de resolver como los anteriores denominados NP-duros para los cuales tampoco existe un algoritmo polinómico que los resuelva, es decir que esta clase de problemas NP no puede abordarse de forma realista con técnicas exactas. En consecuencia, el uso de técnicas aproximadas está recibiendo en las últimas décadas cada vez más atención. En estos métodos aproximados se sacrifica la garantía de encontrar el óptimo global al problema (en muchos casos, aunque no siempre) con el fin de encontrar soluciones buenas en un tiempo significativamente reducido en comparación con los métodos exactos.
En las dos últimas décadas ha emergido un nuevo tipo de técnicas aproximadas que consiste básicamente en la combinación de métodos heurísticos (técnicas aproximadas con componentes aleatorios guiados) básicos en entornos de más alto nivel con el fin de explorar el espacio de búsqueda de una forma eficiente y efectiva. Estos métodos son comúnmente conocidos con el término metaheurísticas. En [Blum y Roli 2003] se pueden encontrar recopiladas varias definiciones de metaheurísticas dadas por diferentes autores, pero en general podemos decir que las metaheurísticas son estrategias de alto nivel que planifican de manera estructurada la aplicación de varias operaciones para explorar espacios de búsqueda de elevada dimensión y complejidad intrínseca.
Los algoritmos metaheurísticos son una familia de algoritmos cuya meta es precisamente dar soluciones aproximadas a problemas generales de tipo NP sin necesidad de recorrer todo el espacio de búsqueda.
El objetivo de este trabajo es proponer un algoritmo metaheurístico híbrido aplicado a resolver el problema de ruteo de vehículos.

2. MARCO TEÓRICO

El primer trabajo donde se plantea el VRP fue en la década del 50 [Dantzig y Ramser 1959], donde se aplicó a un problema distribución de combustible. En este trabajo abordaremos el CVRP (problema de ruteo de vehículos con capacidad uniforme).
Si al problema CVRP se tratara con programación lineal [Bodin, Golden et al. 1983] podría formularse de la siguiente manera:

$\mathbf{C_{lm}}$ Costo de moverse desde l hasta m **(1)**

$\mathbf{X_{lm}^{k}}$ $\begin{cases} \mathbf{1,} \text{ Si el arco (l,m) es transitado x por el vehículo k} \\ \mathbf{0}\text{, en otro caso} \end{cases}$ **(2)**

$\mathbf{q_l}$ Demanda existente en l **(3)**

$\mathbf{Q_k}$ Capacidad del vehículo k **(4)**

$\mathbf{st_l^k}$ Tiempo de servicio de la demanda l por el vehículo k **(5)**

$\mathbf{tt_{lm}^k}$ Tiempo de viaje desde l hasta m del vehículo **(6)**

$\mathbf{T_k}$ Máximo tiempo de ruta permitido para el vehículo k **(7)**

El Modelo resultante, minimizando la función es el siguiente:

$$\sum_{l=1}^{n}\sum_{m=1}^{n}\sum_{k=1}^{K} c_{lm} x_{lm}^{k} \quad \textbf{(8)}$$

Así la expresión (8), es la función objetivo de minimizar las distancias recorridas por los vehículos. [Laporte 1992]

Cada cliente tiene cierta demanda que deberá ser satisfecha por algún vehículo. En muchos casos, la demanda es un bien que ocupa lugar en los vehículos y es usual que un mismo vehículo no pueda satisfacer la demanda de todos los clientes en una misma ruta. Un caso equivalente al anterior ocurre cuando los clientes son proveedores y lo que se desea es recoger la mercadería y transportarla hacia el depósito. También podría ocurrir que la mercadería deba ser transportada a los clientes pero no esté inicialmente en el depósito, sino distribuida en ciertos sitios proveedores. En este caso, los proveedores deben ser visitados antes que los clientes. En otros casos la demanda no es un bien sino un servicio: el cliente simplemente debe ser visitado por el vehículo. Un mismo vehículo podría, potencialmente, visitar a todos los clientes. En otra variante del problema, cada cliente tiene una ubicación y desea ser transportado hacia otro sitio. Aquí la capacidad del vehículo impone una cota sobre la cantidad de clientes que puede alojar simultáneamente. Es usual que cada cliente deba ser visitado exactamente una vez. Sin embargo, en ciertos casos se acepta que la demanda de un cliente sea satisfecha en momentos diferentes y por vehículos diferentes.

Los clientes podrían tener restricciones relativas su horario de servicio. Usualmente estas restricciones se expresan en forma de intervalos de tiempo (llamados ventanas de tiempo) en los que se puede arribar al cliente. Tanto los vehículos como las mercaderías a distribuir (si las hubiera) suelen estar ubicadas en depósitos. Usualmente se exige que cada ruta comience y finalice en un mismo depósito, aunque este podría no ser el caso en algunas aplicaciones (por ejemplo, podría ser que el viaje debiera finalizar en el domicilio del conductor del vehículo).

En los problemas con múltiples depósitos cada uno de estos tiene diferentes características, por ejemplo, su ubicación y capacidad máxima de producción. Podría ocurrir que cada depósito tenga una flota de vehículos asignada a priori o que dicha asignación sea parte de lo que se desea determinar. Los depósitos, al igual que los clientes, podrían tener ventanas de tiempo asociadas. En algunos casos debe considerarse el tiempo necesario para cargar o preparar un vehículo antes de que comience su ruta, o el tiempo invertido en su limpieza al regresar. Incluso, por limitaciones de los propios depósitos, podría querer evitarse que demasiados vehículos estén operando en un mismo depósito a la vez (es decir, la congestión del depósito).

La capacidad de un vehículo podría tener varias dimensiones, como por ejemplo peso y volumen. Cuando en un mismo problema existen diferentes mercaderías, los vehículos podrían tener compartimentos, de modo que la capacidad del vehículo dependa de la mercadería de que se trate. En general, cada vehículo tiene asociado un costo fijo en el que se incurre al utilizarlo y un costo variable proporcional a la distancia que recorra.

Los problemas en que los atributos (capacidad, costo, etc.) son los mismos para todos los vehículos se denominan de flota homogénea, y, si hay diferencias, de flota heterogénea. La cantidad de vehículos disponibles podría ser un dato de entrada o una variable de decisión. El

objetivo más usual suele ser utilizar la menor cantidad de vehículos y minimizar la distancia recorrida ocupa un segundo lugar.
Regulaciones legales podrían imponer restricciones sobre el tiempo máximo que un vehículo puede estar en circulación e incluso prohibir el pasaje de ciertos vehículos por ciertas zonas. En algunos casos se desea que la cantidad de trabajo realizado por los vehículos (usualmente el tiempo de viaje) no sea muy dispar.
Como se ha mencionado anteriormente, se han propuesto diferentes variaciones del VRP canónico con la intención de acercarse a contextos reales del problema, estos problemas incluyen la adición de variables y restricciones. A continuación se nombran algunas de éstas variantes: VRPTW *(Vehicle Routing Problem with Time Windows)*, y variantes (VRPHTW) y (VRPSTW). CVRP *(Capacited Vehicle Routing Problem)*, SVRP *(Stochastic Vehicle Routing Problem)*, VRPSD *(Vehicle Routing Problem with Stochastic Demands)*, VRPSC *(Vehicle Routing Problem with Stochastic Customers)*, VRPST *(Vehicle Routing Problem with Stochastic Times)*, SDVRP *(Split Delivery Vehicle Routing Problem)*, VRPB *(Vehicle Routing Problem with Backhauls)*, VRPPD *(Vehicle Routing Problem with Pick-Up and Delivery)* y PVRP *(Periodic Vehicle Routing Problem)*, entre otras.

Metaheurísticas

Las Metaheurísticas (MHs) [Glover y Kochenberger 2002] son métodos que integran de diversas maneras, procedimientos de mejora local y estrategias de alto nivel para crear un proceso capaz de escapar de óptimos locales y realizar una búsqueda robusta en el espacio de búsqueda. En su evolución, estos métodos han incorporado diferentes estrategias para evitar la convergencia a óptimos locales, especialmente en espacios de búsqueda complejos.
Tienen un rol fundamental en la Investigación de Operaciones, pues pueden ser aplicadas a problemas de Optimización Combinatoria, con resultados muy cercanos al óptimo. Se basan en la observación de la naturaleza, la evolución biológica, procesos físicos asociados a la manufactura, etc. Dentro de las características deseables de una metaheurística, mencionadas en [Golden, Wasil et al. 1998], están:

- Ser algoritmos de optimización global, por ende implica la existencia de mecanismos que le permitan escapar de óptimos locales, ya sea perturbando la solución actual, o bien generándola basada en otras anteriores, aceptando con una cierta probabilidad alguna que no mejora la evaluación de la función objetivo, etc.
- Brindar suficiente libertad a quien la implemente, mediante la posibilidad de trabajar con distintos parámetros, estrategias de paralelización, adición de heurísticas complementarias, etc. Lograr un rendimiento consistente y estable en los problemas que intenta resolver.

Hay diferentes formas de clasificar y describir las técnicas metaheurísticas [Crainic y Toulouse 2003]. Dependiendo de las características que se seleccionen se pueden obtener diferentes taxonomías: basadas en la naturaleza o no basadas en la naturaleza, basadas en memoria o sin memoria, con función objetivo estática o dinámica, etc. En este trabajo se ha elegido clasificarlas de acuerdo a si en cada paso manipulan un único punto del espacio de búsqueda o trabajan sobre un conjunto (población) de ellos, es decir, esta clasificación divide a las metaheurísticas en basadas en trayectoria y basadas en población. Esta clasificación es ampliamente utilizada en la comunidad científica y se la muestra de forma gráfica en la Figura 1. Las siglas se corresponden de la siguiente manera: *Simulated Annealing (SA)* [Kirkpatrick et al. 1983], *Tabu Search (TS))* [Glover 1986; Glover y Laguna 1993], *Greedy Randomized Adaptive Search Procedure (GRASP)* [Feo y Resende 1995; Resende y Ribeiro 2003], *Variable Neighborhood Search (VNS)* [Hansen y Mladenovic 1997], *Iterated Local Search (ILS)* [Louren et al. 2001], *Evolutionary Algorithms (EA)* [Bäck et al. 1997], *Scatter Search (SS)* [Glover

y Kochenberger 2002], *Ant Colony Optimization (ACO)* [Dorigo 1992; Dorigo y Di Caro 1999], *Particle Swarm Optimization (PSO)* [Eberhart y Kennedy 1995].

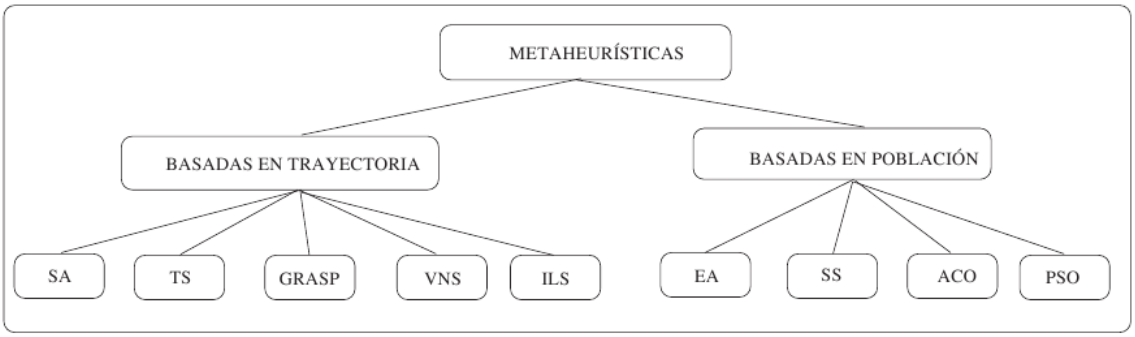

Figura 1: Clasificación de las Metaheurísticas

- Algunas de las metaheurísticas más comúnmente utilizadas en el problema de ruteo de vehículo y sus variantes son por ejemplo [Liu, Huang et al. 2009], en forma híbrida en [Bäker y Ayechew 2003], combinando características de otras metaheurísticas. En [Goel y Gruhn 2008] se complementan el método heurístico y el algoritmo exacto unificado, otro caso sería el presentado en [Alba y Dorronsoro 2004] donde conjugan metaheurísticas poblacional y de trayectoria. Por otra parte, se ha trabajado con algoritmos meméticos (algoritmos genéticos con algún procedimiento adicional de búsqueda local) en [Hart et al. 2005, Prins 2009], para la extensión del problema que considera una flota heterogénea de vehículos, al igual que en [Liu, Huang et al. 2009; Prins 2009], utilización de Tabu Search [Kelly y Xu 1996; Cordeau 1997; Brandão 2009], *Simulated Annealing* [Osman 1993; Czech y Czarnas 2002], *Ant Colony* [Bell y McMullen 2004], Algoritmos Evolutivos [Xu et al. 2005], para el problema OVRP se uso VNS en [Fleszar, Osman et al. 2009] y para otras extensiones de VRP trabajos en [Cordone y Calvo 2001; Mingozzi y Valletta 2003; Sigurjonsson 2008; Novoa y Storer 2008], entre muchas otros.

Metaheurísticas híbridas

Una vez presentadas las metaheurísticas en su generalidad, se presentan las metaheurísticas híbridas [Talbi 2009]. Las Metaheurísticas híbridas consisten en combinar dos o más algoritmos, diferentes metaheurísticas y de métodos de otros campos de la metaheurísticas. Para obtener sistemas que aprovechen más las ventajas de las estrategias individuales para conseguir un mayor beneficio, que por separado (sinergia). La combinación de estrategias que permitan la reducción de la complejidad del problema, y el mejoramiento de las soluciones son los enfoques más usados por los autores para hacer sus métodos competitivos.

Para algunos problemas prácticos y también académicos, los mejores resultados se han obtenido unos en otros, los mejores resultados se han obtenido utilizando algoritmos híbridos. [Talbi, 2009] presenta una taxonomía de metaheurísticas híbridas y propone dos clasificaciones para este tipo de métodos: jerarquizadas y plana. Las diferentes hibridaciones de metaheurísticas pueden clasificarse de un modo jerárquico en:

- Combinación de bajo nivel: los procedimientos heurísticos están embebidos unos en otros, de tal forma que para dos procedimientos dados una función del procedimiento continente se sustituye por el procedimiento embebido. Esta combinación se divide en:
  - Serie *(LRH- Low level Relay Hybrid)* un método se introduce dentro de otro como una función.

- Paralelo *(LTH- Low level Teamwork Hybrid)* se tiene una población de soluciones de tal forma que sobre cada solución actúa un método que caracteriza por contener a otro método.

- Combinación de alto nivel: los métodos heurísticos se combinan están autocontenidos, de forma que no existe interacción entre ellos. Se dividen en:
  - Serie *(LRH- Low level Relay Hybrid)* se tiene una única solución de tal forma que un método se aplica después del otro.
  - Paralelo *(LTH- Low level Teamwork Hybrid)* se tiene una población de soluciones de forma que cada método se aplica independientemente a cada solución.

Además, las metaheurísticas hibridas pueden organizarse en una clasificación plana de la siguiente manera:

- Homogéneas o Heterogéneas
  - Homogéneas: todos los algoritmos combinados utilizan la misma metaheurística.
  - Heterogéneas: los algoritmos combinados utilizan diferentes metaheurísticas.
- Globales o parciales
  - Globales: en las hibridaciones globales, todos los algoritmos buscan en todo el espacio de búsqueda. El objetivo es entonces explorar el espacio más minuciosamente.
  - Parciales: en las hibridaciones parciales el problema se descompone en subproblemas, cada uno definido en su propio espacio de búsqueda. Cada uno de los algoritmos se dedica a explorar uno de esos sub-espacios. Los subproblemas están relacionados entre sí a través de restricciones entre las soluciones encontradas en cada uno. Por lo tanto, los algoritmos se comunican entre ellos para respetar estas restricciones y construir una solución global factible.
- Especializados o generales
  - Especializados: se combinan algoritmos resuelven diferentes problemas de optimización.
  - Generales: todos los algoritmos resuelven el mismo problema de optimización.

Atendiendo a la clasificación de las Metaheurísticas propuestas por Talbi, en la siguiente sección presentamos los algoritmos híbridos utilizados en este trabajo.

3. ALGORITMOS UTILIZADOS PARA EL PROBLEMA DE CVRP

En esta sección se detalla los algoritmos utilizados para resolver el problema CVRP, cuyo objetivo es encontrar una distancia de recorrido total de un mínimo de un plan de ruta que satisfaga la limitar la capacidad de vehículo se presentan.

Para resolver éste problema se utiliza el algoritmo MCMP-SRI como algoritmo base, dos hibridaciones que utilizan Hill-Climbing [Miño y Villagra 2012] y se propone un algoritmo híbrido que atendiendo a la taxonomía propuesta por Talbi, podríamos decir que esta hibridación se acerca a una hibridación de bajo nivel desde el punto de vista jerárquico y homogénea desde el punto de vista plano. Además en nuestro caso se utiliza una mutación basada en conceptos de computación cuántica.

El algoritmo MCMP-SRI *(Multiple Crossover Multiples Parents – Stud and Random Inmigrates)* fue aplicado en diferentes problemas de planificación de máquina única para casos

estáticos y casos dinámicos y los resultados obtenidos fueron satisfactorios. En los problemas estáticos, MCMP-SRI se aplicó para resolver problemas de *Earliness* y *Tardiness* [Pandolfi et al. 2001], *Weighted Tardiness* [De San Pedro et al. 2001], *Average Tardiness* [Pandolfi et al. 2003] y *Weighted Number of Tardy Jobs* [De San Pedro et al. 2003a]. En los problemas dinámicos MCMP-SRI fue aplicado para resolver problemas de adaptabilidad para *Earliness* y *Tardiness* [Villagra et al. 2001], en problemas de dinámica parcial y total para Weighted *Tardiness* [Lasso et al. 2003] y *Average Tardiness* [De San Pedro et al. 2003b]. Además se lo utilizó el manejo de restricciones con metaheurísticas en [Villagra et al. 2012] con resultados promisorios.

Para codificar las visitas a los clientes, que representan una posible solución, se utilizó una permutación de números enteros donde cada permutación $p_i = (p_1, p_2 ..., p_n)$ es un cromosoma en el que pi representa el cliente i que debe ser visitado y n representa la cantidad de clientes a visitar. El cromosoma define el orden de la secuencia a seguir para visitar cada cliente, la función objetivo es minimizar la distancia total del plan de ruta general para satisfacer las demandas de todos los clientes, teniendo en cuenta la capacidad de limitar la vehículo, Q. El funcionamiento de MCMP-SRI para crear descendientes, en es el siguiente: de la vieja población de individuos, se selecciona un individuo, el semental (*Stud*), a través de selección proporcional. Se genera un pool de apareamiento con n2 padres generados aleatoriamente. El semental se aparea con cada padre del pool de apareamiento y las parejas se someten a operaciones de recombinación, y generan 2*n2 máximo de padres. El mejor de los 2*n2 descendientes, se almacena en un pool de hijos temporal. Esta operación de recombinación se repite n1 veces, para diferentes puntos de corte cada vez, hasta que el pool de hijos se complete. Finalmente, el mejor descendiente creado de n2 padres y n1 máximo de recombinaciónes, se inserta en la nueva población. El método de recombinación utilizado fue PMX *(Partial Mapped Crossover):* [Goldberg y Lingle 1987] que puede verse como una extensión del cruzamiento de dos puntos para representaciones basadas en permutaciones. La selección de individuos fue a través de selección proporcional. En la Figura 2 se muestra el código del algoritmo MCMP-SRI.

```
MCMP-SRI
t=0; {Generación Inicial}
inicializar (Stud(t));
evaluar (Stud(t));
while (not max_evaluaciones) do
        pool_apareamiento = Inmigrates_generados_aleatoriamente ∪ Select (Stud(t));
        while (not max_padres) do
                while (not max_recombinaciones) do
                        evalucionar (pool_apareamiento);
                end while
        end while
        evaluar (mating_pool);
        Stud(t+1) = seleccionar la nueva población del pool de apareamiento.
        t = t+1;
end while
```

Figura 2: Algoritmo MCMP-SRI

En cuanto al algoritmo de *Hill-Climbing* (HC) [Hoos y Stützle 2004] puede considerarse como una metaheurística de trayectoria ya que es una versión básica de un algoritmo de búsqueda local en donde se considera todo el vecindario. Se trata de un algoritmo de descenso. Es decir, que la elección de las dirección de búsqueda se hace de forma exhaustiva (se calculan todas las posibles direcciones), y se elige aquella que consigue un mayor descenso.

Las versiones utilizadas han sido seleccionadas luego de realizar diferentes propuestas variando la aplicación de HC, es decir aplicando HC a toda la solución o a cada uno de las rutas (que componen la solución) y además variando en cada caso la cantidad de vecinos utilizados por HC.
En la primera versión denominada MCMP-SRI-HC se aplica *Hill-Climbing* a cada ruta en cada generación. Es decir, que luego de finalizada una generación del MCMP-SRI se toma la mejor solución se le aplica HC a cada ruta y si el HC encuentra la mejor solución, ésta pasa a reemplazar la anterior; caso contrario se mantiene la mejor solución de esa generación.
Este proceso se repite para cada generación. La Figura 3 muestra la versión propuesta MCMP-SRI-HC.

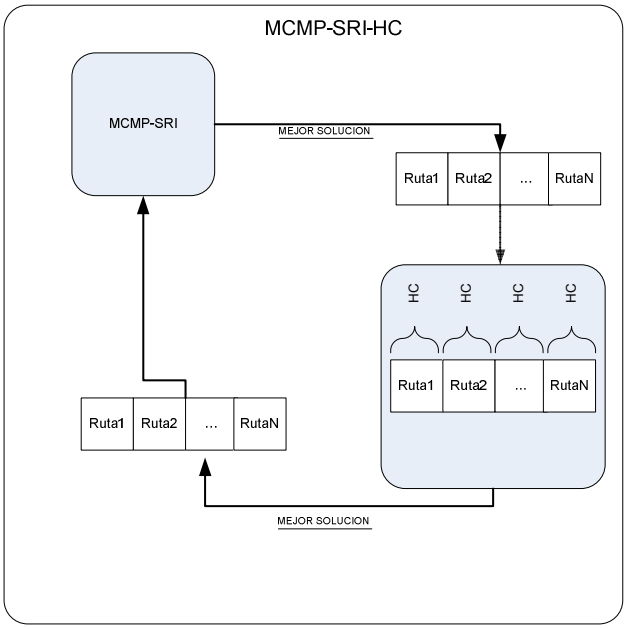

Figura 3: Mecanismo de MCMP-SRI-HC

En una segunda versión propuesta denominada MCMP-SRI-Comb. se decide si aplicar HC a cada una de las rutas que componen la solución (como en la versión anterior) o aplicar HC a toda la solución. El proceso es el siguiente: luego de finalizada una generación del MCMP-SRI se toma la mejor solución y se la compara con la mejor solución global, si esta solución es mejor que la mejor solución global entonces se le aplica HC a cada ruta y si el HC encuentra una mejor solución, ésta pasa a reemplazar la anterior; caso contrario se mantiene la mejor solución de esa generación. En caso de que esta solución no sea mejor que la mejor solución global entonces se le aplica HC global es decir, se le aplica HC a la solución como un todo. De igual forma que antes si el HC encuentra una mejor solución, ésta pasa a reemplazar la anterior; caso contrario se mantiene la mejor solución de esa generación. Este proceso se repite para cada generación. La Figura 4 muestra esta versión MCMP-SRI-Comb.

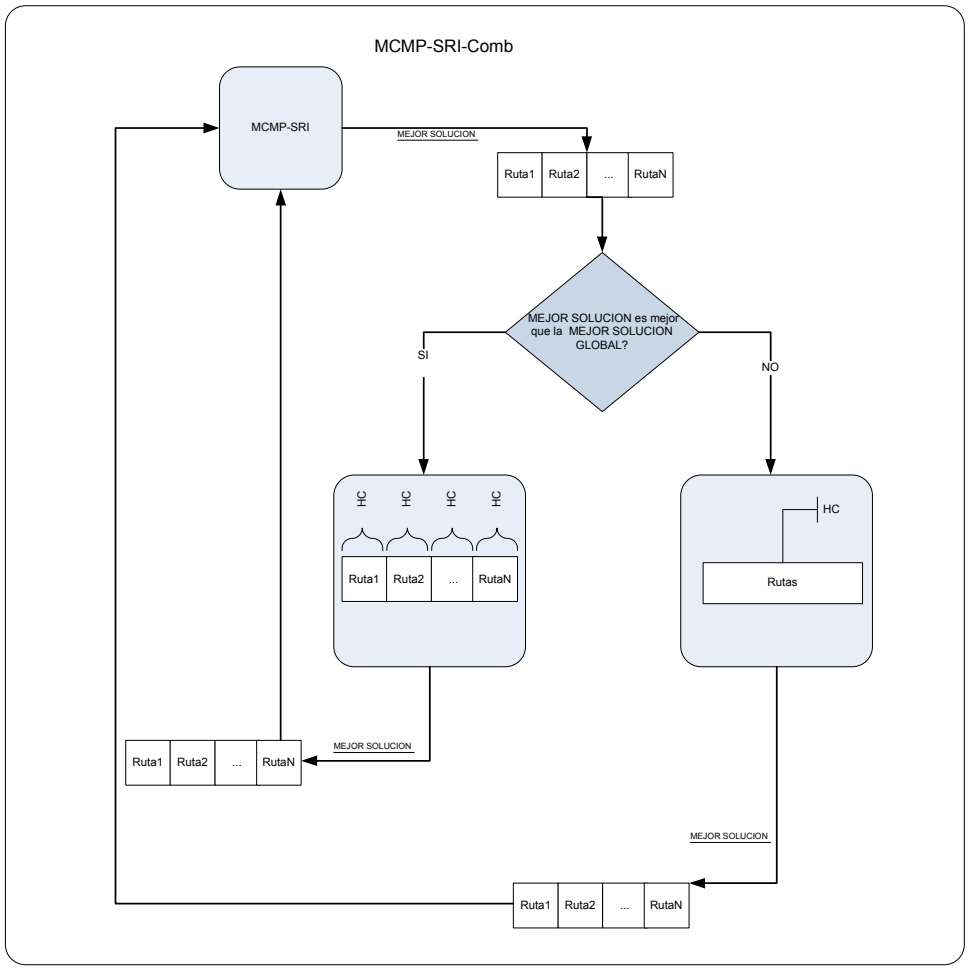

Figura 4: Mecanismo de MCMP-SRI-Comb

Algoritmo Propuesto

Antes de explicar el funcionamiento del algoritmo propuesto se presenta una breve introducción a Computación Cuántica.

A principios de los 80, Richard Feynman observó que algunos efectos de mecánica cuántica no pueden ser simulados de manera eficiente en una computadora clásica. Su observación llevó a la especulación de que la computación en general podría ser de forma más eficiente si se utilizan estos efectos cuánticos. Esta especulación resultó justificada en el año 1994 cuando Peter Shor describe un algoritmo cuántico de tiempo polinomial para factorizar números. En los sistemas cuánticos, el espacio computacional aumenta exponencialmente con el tamaño del sistema que permite paralelismo exponencial. Este paralelismo podría conducir a algoritmos cuánticos exponencialmente más rápido que los clásicamente usados [Draa et al. 2010].

En las computadoras clásicas la mínima cantidad de información almacenada es el bit. Una celda de memoria atómica puede almacenar uno de dos posibles estados discretos, el 0 o el 1. La aplicación de la mecánica cuántica al concepto de bit es lo que permite el nacimiento del bit cuántico o qubit (quantum bit): una celda de memoria que puede encontrarse en uno de los dos estados (0 ó 1), o en una determinada superposición de ambos.

Esto significa que en un registro de N qubits se pueden representar hasta 2N valores distintos. Y al hacer una operación sobre un registro de qubits se estará haciendo sobre todos los valores que estén superpuestos en el registro. A modo de ejemplo, si se tiene un registro con 10 qubits entonces ese mismo registro puede almacenar hasta 1024 valores distintos a la vez, esto

es la superposición de todos los posibles valores que podrían tomar los 10 bits clásicos. Al operar con ese registro, se estará aplicando dicha operación a todos los posibles valores del registro con lo que en este caso se estarán realizando 1024 operaciones con el costo de una sola. La potencia del sistema aumentará exponencialmente con respecto al número de qubits que se consigan agrupar en un registro. Para una profundización de estos conceptos ver [Rieffel y Polak 2000].

Teniendo en cuenta el concepto de computación cuántica y en particular el concepto de qubit se presenta una hibridación de MCMP-SRI con la implementación de una mutación basada en estos conceptos. El algoritmo propuesto se denomina MCMP-SRI-MC y el funcionamiento es el siguiente: en el momento de realizar la mutación se elige una ventana de tamaño tres (éste último valor se seleccionó debido a que es un valor pequeño y controlado para la manipulación en el experimento) y se realizan todas combinaciones posibles para ese tamaño de ventana, evaluando en cada caso el individuo resultante y quedándose con el mejor valor (mejor *fitness*). La Figura 5 brinda una gráfica explicativa de la mutación cuántica utilizado en MCMP-SRI-MC.

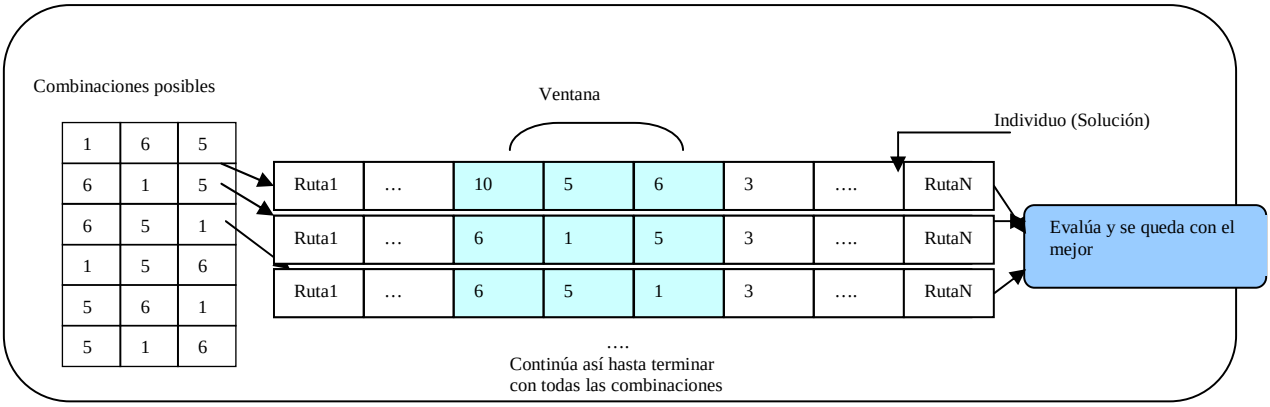

Figura 5: Mecanismo de MCMP-SRI-MC

4. DISEÑO DE EXPERIMENTOS Y RESULTADOS

Para analizar la performance de los algoritmos se utilizaron las instancias provistas por Augerat et al.[1]. Se realizaron 30 corridas independientes de todos los algoritmos de las cuales se seleccionaron las 10 instancias más representativas y se compararon los resultados de las versiones híbridas con los resultados obtenidos por un Algoritmo Genético Simple (AG) y MCMP-SRI.

A continuación se describen las instancias para el problema de Augerat et al.

Este batería de problemas, propuesto en 1995, se compone de tres conjuntos de problemas (conjuntos A, B y P). Todas las instancias de cada conjunto tienen distintas características, como la distribución de la posición de los clientes. Se ha comprobado que las mejores soluciones conocidas son los óptimos para cada instancia de este conjunto de problemas.

A continuación se describen los tres conjunto. (1) **Conjunto A**: Este conjunto está constituido por instancias donde tanto las posiciones de los clientes como las demandas se generan aleatoriamente mediante una distribución uniforme. El tamaño de las instancias está en el rango de 31 a 79 clientes. (2) **Conjunto B**: Las instancias de este conjunto se caracterizan principalmente por estar los clientes agrupados en zonas localizadas. (3) **Conjunto P**: Las instancias

[1] http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/index.htm.old

de la clase ‘P’ son versiones modificadas de otras instancias tomadas de la literatura. En este se trabajó se utilizan las instancias del conjunto A.
Seguidamente, se presentan los detalles y resultados de los métodos aplicados en este trabajo para resolver el CVRP, con el objetivo de encontrar una distancia mínima total de viaje de un plan de rutas que satisface la restricción de capacidad del vehículo. En cada uno de los algoritmos, para codificar las visitas a los clientes que representan una posible solución, se utilizó una permutación de números enteros. Donde cada permutación $p = (p1, p2, \dots , pn)$ es un cromosoma en el cual pi representa el cliente i que debe ser visitado y n representa la cantidad de clientes a visitar. El cromosoma establece el orden de la secuencia a seguir para visitar cada cliente.
La función objetivo es minimizar la distancia total del plan de rutas general de manera de satisfacer las demandas de todos los clientes, cumpliendo con la restricción de capacidad del vehículo, Q. (Detalle sobre la función a minimizar en la expresión (8))

Tamaño población	15
Tamaño cromosoma (cantidad de clientes)	75
Criterio parada (generación)	5000
Recombinación	PMX
Mutación	SW
Probabilidad Recombinación	0,65
Probabilidad Mutación	0,05
Nº de recombinación (n1)(*)	16
Nº de padres (n2)(*)	18
Nº de vecinos (**)	20

Tabla 1: Parámetros de los Algoritmos Utilizados

(*) El número recombinación y de padres utilizamos en los algoritmos multirecombinativos

(**) El número de vecinos sólo se aplica cuando utilizamos los algoritmos propuestos

Se utilizó un tamaño de población de 15 individuos. La población inicial se generó aleatoriamente. Se estableció la probabilidad de mutación en 0,05 y la probabilidad de recombinación en 0,65. El número n1 (número de operaciones de recombinación) y n2 (número de padres) se estableció en 16 y 18 respectivamente, estos parámetros seleccionaron en base a la experimentación de los valores previamente usados exitosamente [Lasso et al. 2003]. La Tabla 1 presenta los parámetros utilizados.
En la Tabla 2 se muestra el resumen de las 30 corridas independientes para las 10 instancias más representativas aplicándole los algoritmos mencionados anteriormente. Las dos primeras columnas corresponden a la instancia utilizada y a su valor óptimo. Luego para cada uno de los algoritmos se muestra la mediana y el error porcentual con respecto al valor óptimo. Los mejores valores para la mediana y para el error porcentual se colocan en negrita. Podemos observar que los algoritmos que han obtenido valores más cercanos al óptimo son MCMP-SRI y MCMP-SRI-MC.

		AG		MCMP-SRI		MCMP-SRI-HC		MCMP-SRI-Comb		MCMP-SRI-MC	
Instancia	Óptimo	Mediana	Error	Mediana	Error	Mediana	Error	Mediana	Error	Mediana	Error
A-n33-k5	661	863,72	0,31	706,20	0,07	705,21	0,07	692,99	**0,05**	695,88	0,06
A-n33-k6	742	895,87	0,21	770,61	**0,04**	778,72	0,05	781,14	0,05	765,43	**0,04**
A-n34-k5	778	970,02	0,24	833,41	0,07	836,64	0,07	841,48	0,08	823,03	**0,06**
A-n37-k6	949	1139,65	0,21	1004,87	0,07	1019,86	0,08	1016,67	0,07	999,46	**0,06**
A-n45-k7	1146	1396,02	0,22	1235,81	0,08	1253,75	0,09	1252,50	0,09	1233,50	**0,07**
A-n53-k7	1010	1467,17	0,46	1177,63	**0,16**	1189,02	0,18	1180,04	0,17	1174,63	0,17
A-n61-k9	1035	1449,55	0,41	1214,39	0,16	1208,56	0,17	1194,54	**0,15**	1201,36	**0,15**
A-n62-k8	1290	1840,78	0,42	1487,72	**0,16**	1511,99	0,18	1532,15	0,19	1485,29	**0,16**
A-n64-k9	1402	1881,99	0,35	1600,69	**0,14**	1625,44	0,16	1601,42	**0,14**	1596,97	**0,14**
A-n80-k10	1764	2553,64	0,45	2072,06	**0,18**	2116,96	0,21	2085,30	**0,18**	2092,03	0,20

Tala 2: Resultados obtenidos por los algoritmos para 10 instancias de problemas.

	AG	MCMP-SRI	MCMP-SRI-HC	MCMP-SRI-Comb.	MCMP-SRI-MC
Instancia	Error	Error	Error	Error	Error
A-n33-k5	0,31	0,07	0,07	**0,05**	0,06
A-n33-k6	0,21	**0,04**	0,05	0,05	**0,04**
A-n34-k5	0,24	0,07	0,07	0,08	**0,06**
A-n37-k6	0,21	0,07	0,08	0,07	**0,06**
A-n45-k7	0,22	0,08	0,09	0,09	**0,07**
A-n53-k7	0,46	**0,16**	0,18	0,17	0,17
A-n61-k9	0,41	0,16	0,17	**0,15**	**0,15**
A-n62-k8	0,42	**0,16**	0,18	0,19	**0,16**
A-n64-k9	0,35	**0,14**	0,16	**0,14**	**0,14**
A-n80-k10	0,45	**0,18**	0,21	**0,18**	0,20

Tabla3: Porcentaje de Error obtenidos por los algoritmos AG, MCMP-SRI, MCMP-SRI-HC, MCMP-SRI-Comb y MCMP-SRI-MC para 10 instancias de problemas

Para realizar un análisis más profundo de los resultados únicamente se analiza el error porcentual.
En la Tabla 3 se muestran los resultados utilizados para realizar los análisis estadísticos.
Podemos observar una clara diferencia en los resultados obtenidos entre el algoritmo AG y las versiones hibridas. Además es importante resaltar que nuestra propuesta (MCMP-SRI-MC) obtiene para 7 de las 10 instancias los mejores valores de mediana y por lo tanto menor error porcentual con respecto al valor óptimo. No obstante, para poder concluir sobre si existen diferencias estadísticamente significativas entre alguno de los algoritmos debemos realizar otros test estadísticos. Para ello comenzaremos verificando las condiciones necesarias para aplicar test estadísticos paramétricos o no paramétricos, éstas condiciones son: Independencia, Normalidad y Homocedasticidad.
Como los resultados a analizar provienen de corridas independientes sólo debemos verificar las otras dos condiciones.
En las Tablas 4, 5 y 6 mostramos los resultados obtenidos para los test de Normalidad y homocedasticidad. A continuación mostramos los test de Kolmogarov-Smirnov y Shapiro-Wilk, junto con un análisis de homocedasticidad utilizando el test de Levene. Para todos los test utilizamos una probabilidad de error p = 0, 05 y como herramienta SPSS. La Tabla 4 muestra los resultados donde el símbolo "*" indica que no cumple la normalidad y el valor a continuación representa el valor p en cada caso. Analizando los resultados de la Tabla 4 sobre la aplicación del test de Kolmogorov- Smirnov podemos decir que cuando los resultados de al

menos un algoritmo no cumple las condiciones de normalidad debemos entonces aplicar test no paramétricos. Es por eso, que por ejemplo en la instancia A-n33-k6 los resultados del algoritmo MCMP-SRI-Comb no son normales por lo tanto se deberán aplicar test no paramétricos. Analicemos ahora el caso de la instancia de problema A-n34-k6 en este caso ambos test de normalidad se cumplen pero al aplicar el test de Levene observamos que los resultados obtenidos no cumplen con la propiedad de homocedasticidad, por lo tanto también se deben aplicar test no paramétricos en este caso. Finalmente, observando cada uno de los resultados obtenidos se puede determinar que para todos los casos se utilizarán test no paramétricos.

	AG	MCMP-SRI	MCMP-SRI-HC	MCMP-SRI-Comb	MCMP-SRI MC
A-n33-k5	0,20	0,20	0,07	*0,03	0,20
A-n33-k6	0,20	*0,03	0,20	0,20	0,06*
A-n34-k5	0,20	0,20	0,20	0,20	0,20
A-n37-k6	0,20	0,06	0,20	0,20	0,04*
A-n45-k7	0,13	0,20	0,16	0,20	0,19
A-n53-k7	0,20	0,20	0,20	*0,00	0,20
A-n61-k9	0,20	0,20	0,20	0,20	0,20
A-n62-k8	0,80	0,14	0,11	0,07	0,20
A-n64-k9	0,20	0,20	0,20	0,20	0,20
A-n80-k10	0,20	0,20	*0,03	0,11	0,12

Tabla 4: Test de normalidad de Kolmogorov-Smirnov

La Tabla 5 muestra los resultados aplicando el test de normalidad de Shapiro-Wilk con el mismo nivel de confianza y los mismos valores que la Tabla anterior. Donde podemos observar casos similares a los encontrados aplicando el test de Kolmogorov-Smirnov. Por ejemplo si vemos los resultados obtenidos por los algoritmos en la instancia A-n34-k5 observamos que se cumple también en este caso la condición de normalidad pero si observamos la Tabla 6 vemos que no se cumple la homocedasticidad. Analizamos cada una de las instancias y vemos que se deben aplicar test no paramétricos en todos los casos.

	AG	MCMP-SRI	MCMP-SRI-HC	MCMP-SRI- Comb	MCMP-SRI-MC
A-n33-k5	0,33	0,21	0,17	*0,02	0,02*
A-n33-k6	0,08	*0,01	*0,05	*0,03	0,03*
A-n34-k5	0,82	0,67	0,08	0,61	0,52
A-n37-k6	0,32	*0,02	0,25	0,38	0,28
A-n45-k7	0,07	0,48	0,18	0,25	0,63
A-n53-k7	0,09	0,86	0,29	*0,01	0,22
A-n61-k9	0,19	0,77	0,51	*0,01	0,80
A-n62-k8	0,29	0,38	0,30	0,81	0,06*
A-n64-k9	0,20	0,57	0,81	0,42	0,56
A-n80-k10	0,34	0,49	0,51	0,45	0,45

Tabla 5: Test de normalidad de Shapiro-Wilk

Instancia	Levene	Instancia	Levene
A-n33-k5	*0,000	A-n53-k7	*0,000
A-n33-k6	*0,000	A-n61-k9	*0,000
A-n34-k5	*0,000	A-n62-k8	*0,000
A-n37-k6	*0,000	A-n64-k9	*0,000
A-n45-k7	*0,000	A-n80-k10	*0,000

Tabla 6: Test de Homocedasticidad de Levene (basado en medias)

Seguidamente determinaremos si las diferencias de los resultados obtenidos por los algoritmos para cada una de las instancias son estadísticamente significativas. Teniendo en cuenta que no se cumplen las condiciones para realizar los test paramétricos se aplica entonces el test de Kruskal-Wallis para determinar si existen diferencias significativas entre los algoritmos. Utilizaremos el signo (+) para especificar que existen diferencias significativos entre los resultados obtenidos por los algoritmos y (-) en caso contrario.
La Tabla 7 muestra los resultados obtenidos para las instancias utilizadas. Podemos observar que en todos los casos las diferencias entre los resultados obtenidos por los algoritmos son estadísticamente significativas.
Ahora para saber entre los resultados de qué algoritmos existen diferencias estadísticamente significativas aplicamos el test de Tukey.

Instancia	Kruskal-Wallis	Instancia	Kruskal-Wallis
A-n33-k5	(+)	A-n53-k7	(+)
A-n33-k6	(+)	A-n61-k9	(+)
A-n34-k5	(+)	A-n62-k8	(+)
A-n37-k6	(+)	A-n64-k9	(+)
A-n45-k7	(+)	A-n80-k10	(+)

Tabla 7: Test de Kruskal-Wallis

Únicamente se encontraron diferencias estadísticamente significativas entre el algoritmo genético (AG) y los cuatro restantes, esto significa que podemos afirmar con un nivel de confianza del 95% que los algoritmos MCMP-SRI, MCMP-SRI-HC, MCMP-SRI- Comb. y MCMP-SRI- MC tienen un mejor desempeño (en cuanto al porcentaje de error) con respecto al algoritmo genético.
Podemos observar este comportamiento en la gráfica de Box-Plot (Figura 5) que muestra como se distribuyen los resultados con respecto a la mediana y podemos notar claramente la diferencia de resultados obtenidos por el algoritmo AG y los restantes algoritmos para la instancia A-n45-k7. Además, se puede observar que los resultados de nuestra propuesta MCMP-SRI-MC nuevamente mejoran a los resultados obtenidos por MCMP-SRI. No obstante, estas diferencias no son estadísticamente significativas.
En la Figura 6 para la instancia A-n37-k6 podemos observar nuevamente que los valores obtenidos por MCMP-SRI-MC son menores a los obtenidos por MCMP-SRI. Sin embargo, éstas diferencias no son estadísticamente significativas.

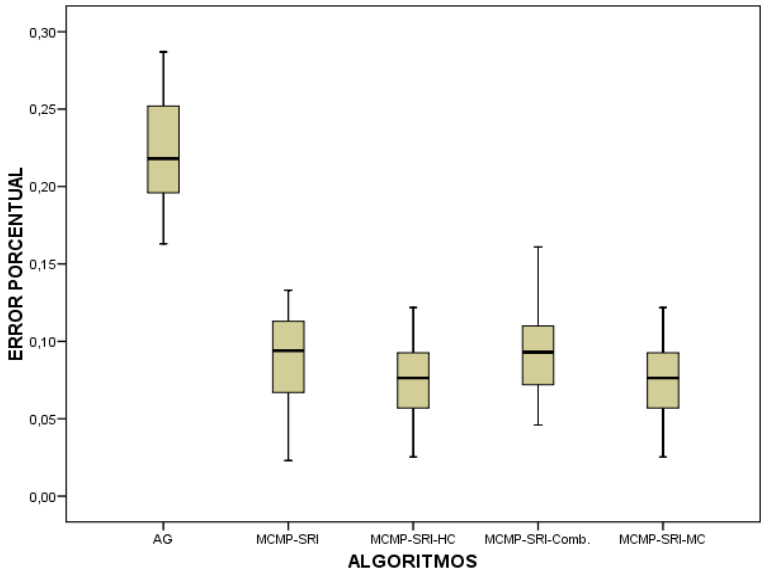

Figura 6: Diagrama de Cajas (Boxplot) para la instancia A-n45k7

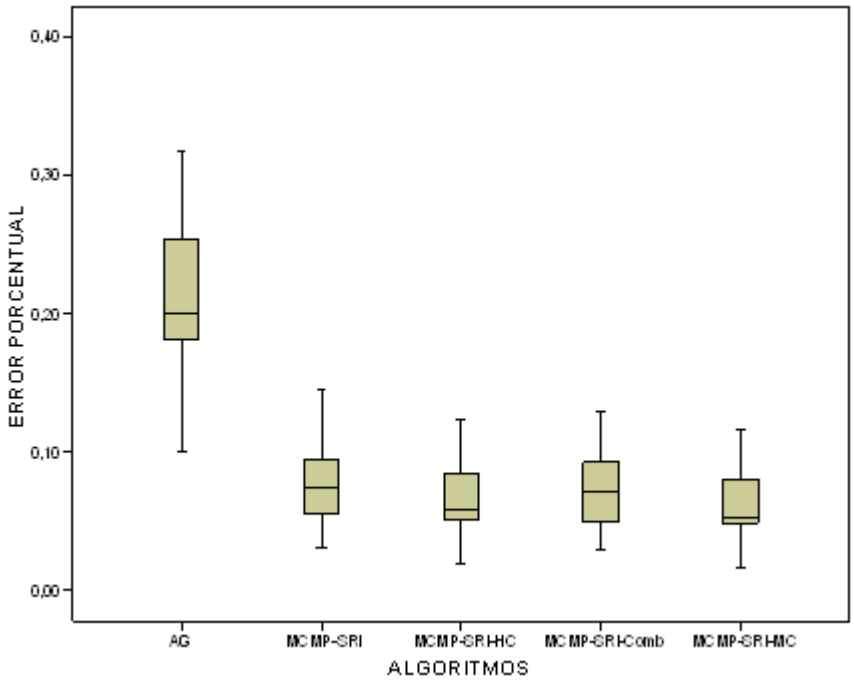

Figura 7: Diagrama de Cajas (Boxplot) para la instancia A-n37k6

Con la idea de analizar ahora el desempeño de los algoritmos sobre el conjunto de problemas aplicaremos un conjunto de test no paramétricos (para muestras relacionadas). Los siguientes test se han aplicado utilizando el paquete (en Java) CONTROLTEST que se puede obtener en el sitio web público temático SCI2S - Statistical Inference in Computational Intelligence and Data Mining. Para una profundización de los test aplicados a continuación ver [Derrac et al. 2011].

Aplicaremos el Test de Friedman Alineado, que se trata de un equivalente no paramétrico del test de ANOVA. Se basa en n conjuntos de filas, donde el rendimiento de los algoritmos analizados se clasifica por separado para cada conjunto de datos. Este esquema permite la clasificación “intra-conjunto” estableciendo comparaciones, ya que comparaciones “inter-conjunto”

no tienen sentido. Cuando el número de algoritmos de comparación es pequeña, esto puede suponer una desventaja. En tales casos, la comparabilidad entre los conjuntos de datos es deseable y se puede emplear el método de filas alineadas [Hodges y Lehmann 1962]. En esta técnica, un valor de posición se calcula como el promedio de rendimiento conseguido por todos los algoritmos en cada conjunto de datos.
Luego, se calcula la diferencia entre el rendimiento obtenido por un algoritmo y el valor de la ubicación. Este paso se repite para algoritmos y conjuntos de datos. Las diferencias resultantes, llamadas observaciones alineadas, mantienen su identidad con respecto al conjunto de datos y la combinación de algoritmos a los que pertenecen. Seguidamente, se clasifican de 1 a kn relativa a cada otro. Entonces, el esquema de clasificación es la misma que la empleada por un procedimiento de comparación múltiple que emplea muestras independientes; tales como el test de Kruskal-Wallis. Los rangos asignados a las observaciones alineados se llaman filas alineadas.

Algoritmo	Ranking
AG	40,00
MCMP-SRI	17,80
MCMP-SRI-HC	24,15
MCMP-SRI-Comb	24,20
MCMP-SRI-MC	21,35

Tabla 8: Ranking promedio de los algoritmos (Friedman Alineado)

La Tabla 8 muestra los resultados del estadístico de Friedman Alineado (distribuido de acuerdo a X^2 con 4 grados de libertad: 7,76). El valor de *p* calculado por el test de Friedman Alineado es: 0,10 (valor > 0,05) esto significa que existen diferencias estadísticamente significativas entre los resultados obtenidos por algoritmos. Podemos observar que el algoritmo MCMP-SRI es el que obtiene el mejor ranking luego le sigue nuestra propuesta MCMP-SRI-MC que es el algoritmo que se ha hibridado y el peor ranking es el obtenido por el AG.

5. CONCLUSIONES

Los algoritmos evolutivos son algoritmos de búsqueda robustos en el sentido que proporcionan soluciones buenas en una amplia clase de problemas que de otra manera serían computacionalmente intratables. Para mejorar el funcionamiento AEs, los enfoques multirecombinativos permiten múltiples intercambios de material genético entre múltiples padres y con ello mejorar la velocidad de convergencia. Para enriquecer el proceso de búsqueda, mediante un mejor equilibrio entre la exploración y la explotación, el concepto de *stud* e inmigrantes aleatorios fue insertado en MCMP-SRI. La presencia del *stud* asegura la retención de los rasgos buenos de soluciones anteriores y los inmigrantes aleatorios, como una fuente continua de diversidad genética, evita la convergencia prematura.
El Problema de Ruteo de Vehículos con Capacidad limitada (CVRP), es una de las variantes del VRP que es considerado emblemático en el campo de la distribución, logística y tráfico.
En este trabajo hemos presentado dos versiones de MCMP-SRI combinadas con Hill-Climbing (MCMP-SRI-HC, MCMP-SRI-Comb) y proponemos una versión (MCMP-SRI-MC) que utiliza una mutación basada en conceptos de computación cuántica. Hemos aplicado una familia de test no paramétricos para comparar los resultados (error porcentual) de los algoritmos. Una vez determinada la necesidad de aplicar test no paramétricos se analizaron los resultados realizando comparaciones múltiples aplicando test apareados y no-apareados.

En cuanto a los resultados obtenidos, luego de aplicar test no-apareado (Kruskal-Wallis) podemos afirmar que existen diferencias estadísticamente significativas entre los algoritmos en cada uno de los problemas tratados. Al aplicar el test post-hoc (Tukey) confirmamos que las diferencias en cada uno de los problemas estaban entre los resultados obtenidos por el algoritmo AG con respecto a resultados obtenidos por los otros cuatro algoritmos (MCMP-SRI, MCMP-SRI-HC, MCMP-SRI-Comb y MCMP-SRI-MC). Seguidamente aplicamos un test apareado (Friedman Alineado). En Friedman Alineado, obtuvimos diferencias estadísticamente significativas en los resultados de los algoritmos. Si bien el algoritmo propuesto (MCMP-SRI-MC) obtiene resultados mejores y similares a MCMP-SRI. Las diferencias no son estadísticamente significativas, es importante destacar el fuerte estudio estadístico realizado sobre los resultados obtenidos. Trabajos futuros incluirán otras formas de hibridación y en particular la mutación cuántica se modificará seleccionando una ventana de tamaño determinado y los elementos serán elegidos aleatoriamente, para otras variantes de VRP.

6. AGRADECIMIENTOS

Se agradece la cooperación del equipo de proyecto del LabTEm y a la Universidad Nacional de la Patagonia Austral, de los cuales se recibe apoyo continuo.

7. REFERENCIAS

[Alba, y Dorronsoro 2004]. Alba, E. y Dorronsoro B. "Solving the Vehicle Routing Problem by Using Cellular Genetic Algorithms." Evolutionary Computation in Combinatorial: pp.1-10. 2004

[Bäck 1996] Bäck, T. "Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, genetic Algorithms". Oxford University Press, 1996.

[Bäck et al. 1997] T. Bäck, D. Fogel B., and Michalewicz Z., editors. Handbook of Evolutionary Computation. Oxford University Press, 1997.

[Bäker y Ayechew 2003] Bäker, B. M. y Ayechew M. A: "A genetic algorithm for the vehicle routing problem". Computers & Operations Research, pages 787-800, 2003.

[Bell y McMullen 2004] Bell, J. y McMullen, P. "Ant colony optimization techniques for the vehicle routing problem". Advanced Engineering Informatics. pages 41-48. 2004.

[Blum y Roli 2003] Blum, C. y Roli, A.. "Metaheuristics in combinatorial optimization: Overview and conceptual comparison". ACM Computing Surveys, 35(3):268–308, 2003.

[Brandão, J. 2009]. Brandão, J. "A deterministic tabu search algorithm for the fleet size and mix vehicle routing problem." European Journal of Operational Research vol.195: pp.716-728. 2009.

[Christofides et al. 1979]. Christofides, N., Mingozzi A. Toth P., et al "The vehicle routing problem." Reveu francaise d´automatique d´informatique et de recherche opérationnelle. vol.1.tome10 (Combinatorial Optimization): pp.315–338.1979

[Cook et al. 1998] Cook, W.J. Cunningham, W.H. Pulleyblank, W.R. y Schrijver, A.. Combinatorial Optimization. Wiley Interscience, 1998.

[Cordeau 1997] Cordeau, J-F, Gendreau, M. y Laporte, G."A tabu search heuristic for periodic and multi-depot vehicle routing problems". Networks, 30(2):105–119. 1997.

[Cordone y Calvo 2001]. Cordone, R. y Calvo, R. W. "A Heuristic for the Vehicle Routing Problem with Time Windows." Journal of Heuristics vol.7: pp.107-129. 2001

[Crainic y Toulouse 2003] Crainic, T. y Toulouse, M.. Handbook of Metaheuristics, chapter Parallel Strategies, pages 475–513. Kluwer Academic Publishers. 2003.
[Czech y Czarnas 2002]. Czech Z. J. y Czarnas P.. "Parallel simulated annealing for the vehicle routing problem with time windows". ISBN: 0-7695-1444-8pp. 376 – 383. 2002
[Dantzig y Ramser 1959] Dantzig, G. B., Ramser, R.H.: "The Truck Dispatching Problem". Management Science 6, 80–91. 1959.
[Derrac et al. 2011] Derrac J., García S., Molina D., y Herrera F.. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation. 2011.
[De San Pedro et al. 2001] De San Pedro M.E., Pandolfi D., Villagra A., Vilanova G., Gallard R. 2001; "Stud and immigrants in multirecombined evolutionary algorithm to face weighted tardiness scheduling problems". CACIC'01 VII Congreso Argentino de Ciencias de la Computación, El Calafate, Argentina, Octubre, pp. 1251-1258. 2001
[De San Pedro et al. 2003a] De San Pedro M.E., Villagra A., Lasso M., Pandolfi D., Diaz Vivar M., Gallard R. 2003. "Solutions for the Weighted Number of Tardy Jobs in Single Machine Environments via Evolutionary Algorithms", CSITeA03 International Conference on Computer Science, Software Engineering Information Technology, E-Business and Applications, Rio de Janeiro, Brazil, pp. 438-443. 2003
[De San Pedro et al. 2003b] De San Pedro M.E., Lasso M., Villagra A., Pandolfi D., Gallard R., 2003; "Solutions to the Dynamic Average Tardiness Problem in the Single machine Environments". CACIC'03 IX Congreso Argentino de Ciencias de la Computación, La Plata, Argentina, pp. 1251-1258. 2003
[Dorigo 1992] Dorigo, M. "Optimization, Learning and Natural Algorithms". PhD thesis, Dipartimento di Elettronica, Politecnica di Milano. 1992.
[Dorigo y Di Caro 1999] Dorigo, M. y Di Caro, G. "The ant colony optimization metaheuristic". In Corne D., Dorigo M., y Glover F., editors, New Ideas in Optimization, pages 11–32. McGraw Hill. 1999.
[Draa at el 2010]. Draa A., Talbi H., Batouche M. "A Quantum Genetic Hybrid Algorithm for Solving the Traveling Salesman Problem". Vision & Computer Graphics Team, LIRE Laboratory, Mentouri University, Constantine, Algeria. 2010
[Eberhart y Kennedy 1995] Eberhart, R. y Kennedy, J.. "A new optimizer using particles swarm theory". In Sixth International Symposium on Micro Machine and Human Science (Nagoya, Japan), IEEE Service Center, Piscataway, pages 39–43, 1995.
[Feo y Resende 1995] Feo, T.A. y Resende, M.G.C.. "Greedy randomized adaptive search procedures". Journal of Global Optimization, 6:109–133, 1995.
[Fleszar, Osman, et al. 2009]. Fleszar, K., Osman, I. H. et al."A variable neighborhood search algorithm for the open vehicle routing problem." European Journal of Operational Research vol.195: pp.803-809. 2009
[Glover 1986] Glover, F. "Future paths for integer programming and links to artificial intelligence". Computers & Operations Research. vol. 13, Issue 5, 1986, pp. 533–549.1986
[Glover y Kochenberger 2002] Glover F. y Kochenberger G.. Handbook of Metaheuristics. Kluwer Academic Publishers, Norwell, MA, 2002. Computer and Operations Research, 13:533–549. 1986.
[Glover y Laguna 1993] Glover F. y Laguna M.. "Tabu search in Modern Heuristic Techniques for Combinatorial Problems". John Wiley & Sons, 1993.
[Goel y Gruhn 2008]. Goel, A. y Gruhn, V. "A General Vehicle Routing Problem." European Journal of Operational Research v.191: 650–660. 2008
[Golden y Wasil, et al. 1998]. Golden, B., Wasil, E. "The Impact of Metaheuristics on Solving the Vehicle Routing Problem: algorithms, problem sets, and computational results." Fleet Management and Logistics.Boston: Kluwer: pp.33–56. 1998

[Goldberg. 1989] Goldberg, D. "Genetic Algorithms in Search, Optimization and Machine Learning". Addison-Wesley Publishing Co., 1989.
[Goldberg y Lingle 1987] Goldberg, D y Lingle, R. "Alleles, loci and the traveling salesman problem". Proc.of the First International Conference on Genetic Algorithms, Lawrence Eribaum Associates, pp.154-159.Hilldale, NJ. 1987.
[Hansen y Mladenovic 1997] Hansen, P. y Mladenovic, N.. "Variable neighborhood search for the p-median". Location Science, 5:207–226. 1997.
[Hart et al. 2005] Hart, W., Krasnogor, N., Smith J.E.."Recent Advances in Memetic Algorithms." Sringer: pp.418. 2005
[Hochbaum 1996] Hochbaum, D. S.. "Approximation Algorithms for NP-Hard Problems". International Thomson Publishing, 1996.
[Hodges y Lehmann. 1962] Hodges,J.L. y Lehmann,E.L."Ranks methods for combination of independent experiments in analysis of variance". Annals of Mathematical Statistics 33 pp. 482–497.1962
[Hoos y Stützle 2004] Hoos, H. H y Stützle, T. "Stochastic local search: Foundations & applications". Morgan Kaufmann. 2004.
[Kelly y Xu 1996]. Kelly y Xu ."A Network Flow-Based Tabu Search Heuristic for the Vehicle Routing Problem". vol. 30 no. 4 379-393. 1996.
[Kirkpatrick et al. 1983] Kirkpatrick S., Gelatt, J. y Vecchi M.. "Optimization by simulated annealing". Science, 220:671–680. 1983.
[Korf 1985] Korf, R.. "Depth-first iterative-deepening: An optimal admissible tree search". Artificial Intelligence, 27(1):97–109. 1985.
[Laporte 1992]. Laporte, G. "The vehicle routing problem: An overview of exact and approximate algorithms." European Journal of Operational Research vol.59 pp.345-358. 25 June 1992
[Lasso et al. 2003] Lasso M., Pandolfi D., De San Pedro M. E., Villagra A., Gallard R.; “Solving Dynamic Tardiness Problems in Single Machine Environments”; Congress on Evolutionary Computation - CEC ’04; Portland, U.S.A, Vol 1, 1143- 1149. 2003
[Louren et al. 2001] Louren¸ H. R., Martin, O., y Stützle, T.. "A beginner´s introduction to iterated local search". In MIC, pages 1–6. 2001.
[Liu, Huang, et al. 2009]. Liu, S., Huang, W., Ma, H. "An effective genetic algorithm for the fleet size and mix vehicle routing problems." Transportation Research Part E vol.V45: pp.434-445. 2009
[Mingozzi y Valletta 2003]. Mingozzi, A. y Valletta A. "An Exact Algorithm for period and multi-depot Vehicle Routing Problems."2003.
[Miño y Villagra 2012] Miño, R. y Villagra, A.. "Algoritmos Multirecombinativos aplicados al Problema de Ruteo de Vehículos". Informe científico Técnico UNPA 2012 (en proceso de evaluación). 2012.
[Novoa y Storer 2008]. Novoa, C. y Storer, R. "An approximate dynamic programming approach for the vehicle routing problem with stochastic demands." European Journal of Operational Research vol.196: pp.509-515. 2008
[Osman 1993] Osman I. H.. "Metastrategy Simulated Annealing and Tabu Search Algorithms for the vehicle routing problem". Ann. Operations Research 40(1), pages 421-451. 1993.
[Papadimitriou y Steiglitz 1998] Papadimitriou, C. y Steiglitz, K.. "Combinatorial Optimization". Dover Publications.1998.
[Pandolfi et al. 2001] Pandolfi D., Vilanova G., De San Pedro M.E, Villagra A.; Gallard R. 2001; "Solving the single-machine common due date problem via studs and immigrants in evolutionary algorithms". Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics, Vol. III Emergent Computing and Virtual Engineering, pp. 409-413, Orlando, Florida.2001

[Pandolfi et al. 2003] Pandolfi D., Lasso M., De San Pedro M.E., Villagra A., Gallard R. 2003; "Evolutionary Algorithms to solve average tardiness problems in the single machine environments". CSITeA03 International Conference on Computer Science, Software Engineering Information Technology, E-Business and Applications, Rio de Janeiro, Brazil, pp. 444-449.2003

[Prins 2009] Prins, C.: "Efficient Heuristics for the Heterogeneous Fleet Multitrip VRP with Application to a Large-Scale Real Case". Journal of Mathematical Modelling and Algorithms vol n 1 pp.135-150. 2002.

[Resende y Ribeiro 2003] Resende, M.G.C. y Ribeiro, C.C.. Handbook of Metaheuristics, chapter Greedy randomized adaptive search procedures, pages 219–249. Kluwer Academic Publishers. 2003.

[Rieffel y Polak 2000] Rieffel E. y Polak W., "An introduction to quantum computing for non-physicists", arxive.org, quant-ph/9809016 v2, January 2000.

[Russell y Norvig 1995] Russell S. y Norvig P.. "Artificial Intelligence: A Modern Approch". Prentice-Hall.1995.

[Sigurjonsson 2008]. Sigurjonsson, K "Taboo Search Based Metaheuristic for Solving Multiple Depot VRPPD with Intermediary Depots." Informatics and Mathematical Modelling: pp.2-115.

[Talbi 2009] Talbi, E. G. "Metaheuristics from design to implementation." University of Lille - CNRS - INRIA: 593. 2009

[Vazirani 2003] Vazirani V.. Approximation Algorithms. Springer. 2003.

[Villagra et al. 2001] Villagra A., Pandolfi D., Vilanova G., De San Pedro M.E; Gallard R. 2001. "Adaptability of multirecombined evolutionary algorithms in the single-machine common due date problem". Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics, Vol. III Emergent Computing and Virtual Engineering, pp. 401-404, Orlando, Florida. 2001

[Villagra et al. 2012] Villagra A. , Pandolfi D., Leguizamón G. "Handling constraints with an evolutionary tool for scheduling oil wells maintenance visits". September 2012. Pages 1-19. doi 10.1080/0305215X.2012.713354. 2012

[Xu et al. 2005] Xu, Y. L, Lim M. H. y Er M. J. "Investigation on Genetic Representations for Vehicle Routing Problem", IEEE International Conference on System, Man and Cybernetics, pp. 3083- 3088. 2005.