Algoritmos Evolutivos Multirecombinativos Híbridos Aplicados al Problema de Vehículos con Capacidad Limitada

Viviana Mercado[1], Andrea Villagra[1] , Daniel Pandolfi[1] , Guillermo Leguizamón [2]

[1] Laboratorio de Tecnologías Emergentes LabTEm, Departamento de Ciencias Exactas y Naturales, Universidad Nacional de la Patagonia Austral Unidad Académica Caleta Olivia,
[2] Laboratorio de Investigación y Desarrollo en Inteligencia Computacional, Departamento de Informática, Universidad Nacional de San Luis
[1]{vmercado, avillagra, dpandolfi }@uaco.unpa.edu.ar, [2]legui@unsl.edu.ar

Resumen. El objetivo perseguido en este campo es fundamentalmente el desarrollo de nuevos métodos capaces de resolver problemas complejos con el menor esfuerzo computacional posible, mejorando así a los algoritmos existentes. Las metaheurísticas son métodos que integran procedimientos de mejora local y estrategias de alto nivel para realizar una búsqueda robusta en el espacio-problema. El problema de ruteo de vehículos es un problema de optimización combinatoria de gran importancia en diferentes entornos logísticos debido a su dificultad (NP-duros). Se han propuesto varias soluciones a este problema haciendo uso de heurísticas y metaheurísticas. En este trabajo proponemos dos algoritmos para resolver el problema de ruteo de vehículos con capacidad limitada, utilizando como base un Algoritmo Evolutivo multirecombinativo conocido como MCMP-SRI (Stud and Random Inmigrants), combinado con operadores de mutación basados en conceptos computación cuántica. Detalles de los algoritmos y los resultados de los experimentos muestran un promisorio comportamiento para resolver el problema.

Keywords: Metaheurísticas, Algoritmos Evolutivos multirecombinativos, Hibridación, Problema de Ruteo de Vehículos.

1 Introducción y Trabajos Relacionados

El problema de ruteo de vehículos (Vehicle Routing Problem o las siglas en inglés VRP) 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 [4] y [12].
Algunas de las metaheurísticas más comúnmente utilizadas en el problema de ruteo de vehículo y sus variantes son por ejemplo los Algoritmos Genéticos (AGs) [8] que han tenido éxito en resolver problemas de ruteo de vehículos, corte de empaquetado (Strip Packing), entre muchos otros. Se han aplicado para el VRP original en [2] y [7].

Aplicando búsqueda en vecindarios variables (Variable Neighborhood Search) es referencia de su utilización el trabajo [6]. Para el caso de recocido simulado (Simulated Annealing), una metaheurística de trayectoria, es utilizada en [17], con resultados promisorios en un tiempo razonable para una variante de VRP con ventana de tiempo. Con búsqueda tabú (Taboo Search), otra metaheurística de trayectoria, en [13] se presenta una propuesta híbrida que combina la búsqueda tabú y recocido simulado obteniendo muy buenos resultados para las instancias analizadas en una variante de VRP. En una versión reciente en [3] para el VRP con una flota de vehículos heterogénea y distintas rutas aplica la búsqueda tabú y luego de realizados los experimentos con diversos benchmark del problema concluyen que su algoritmo produce soluciones de alta calidad en un tiempo computacionalmente aceptable. Utilizando colonias de hormigas (ACO) en [19] se presenta un algoritmo adaptativo con una búsqueda local Pareto aplicada a problemas de VRP y CVRP obteniendo resultados de mejor calidad comparados con metaheurísticas clásicas. Abordando enjambre de partículas (Particle Swarm) en [14]. Algunos ejemplos de algoritmos genéticos celulares se presentan en [1]. Una de las tendencias actuales es lograr obtener una formulación general para todos los problemas derivados del VRP, que los incluya como casos particulares. Un esfuerzo notable es el mostrado en [11] que se complementan el método heurístico y el algoritmo exacto unificado. Se observa al estudiar dichos trabajos, que el siguiente paso es lograr mejoras en el rendimiento de los algoritmos, ya sea mediante cambios en las estructuras de datos, o en cómo se acota el espacio de soluciones factibles o bien tratar de potenciarlos mediante la utilización de uno o más métodos, es decir, hibridándolos.

En este trabajo proponemos dos algoritmos evolutivos multirecombinativos que utilizan una mutación basada en conceptos de computación cuántica con el objetivo de mejorar la performance obtenida por el algoritmo sin hibridar.

El trabajo está organizado de la siguiente manera la Sección 2 describe el problema que se aborda en este trabajo. En la Sección 3 se introducen conceptos de hibridación y se muestran los algoritmos propuestos. La Sección 4 presenta el diseño de experimentos y los resultados obtenidos. Finalmente, la Sección 5 provee las conclusiones y futuras líneas de investigación.

2 Descripción del Problema

El VRP se puede definir como un problema de programación entera perteneciente a la categoría de problemas NP-duros. Entre las diferentes variedades de VRP trabajaremos con el VRP de Capacidad limitada (CVRP), en el que cada vehículo tiene una capacidad uniforme de un único artículo. Definimos el CVRP sobre un grafo no dirigido $G = (V, E)$ donde $V=\{v_0, v_1,..., v_n\}$ es un conjunto de vértices y $E = \{(v_i, v_j) / v_i, v_j \in V, i< j\}$ es un conjunto de ejes.

Los vértices v0 parten del depósito, y es desde donde m vehículos de capacidad Q deben abastecer a todas las ciudades o clientes, representados por un conjunto de n vértices $\{v_1, ..., v_n\}$.

Definimos E una matriz $C = (c_{ij})$ de costo, distancia o tiempo de viaje no negativos entre los clientes v_i y v_j. Cada cliente v_i tiene una demanda no negativa de artículos q_i

y tiempos de entrega δ_i (tiempo necesario para descargar todos los artículos). Siendo $v_1,..., v_m$ una partición de V, una ruta Ri es una permutación de los clientes en V_i especificando el orden en el que se visitan, comenzando y terminado en el deposito v_0. El costo de una ruta dada $R_i = \{ v_0, v_1, ..., v_{k+1} \}$, donde $v_j \in V$ y $v_0 = v_{k+1} = 0$ (*0* indica el depósito), viene dada por:

$$\mathrm{Cost}(\mathrm{R_i}) = \sum_{j=0}^{k} \mathrm{C_{i,\,j+1}} + \delta_{\mathrm{j}} \qquad \textbf{(1)}$$

y el costo de la solución al problema (S) es:

$$\mathrm{FCVRP}(\mathrm{S}) = \sum_{i=1}^{m} \mathrm{Cost}(\mathrm{R_i}) \qquad \textbf{(2)}$$

El CVRP consiste en determinar un conjunto de m rutas (i) de costo total mínimo - como especifica la ecuación (2); (ii) empezando y terminando en el deposito v0; de forma que (iii) cada cliente es visitado una sola vez por un sólo vehículo, sujeto a las restricciones (iv) de que la demanda total de cualquier ruta no exceda

$Q(\sum v_j \in R_i;\ \mathrm{q_j} \leq \mathrm{Q})$; y (v) la duración total de cualquier ruta no supera el límite preseleccionado $\mathrm{D}(\mathrm{Cost}(\mathrm{R_i}) \leq \mathrm{D})$. Todos los vehículos tienen la misma capacidad y transportan el mismo tipo de artículo. El número de vehículos puede ser un valor de entrada o una variable de decisión. En este estudio, la longitud de las rutas se minimiza independientemente del número de vehículos utilizados.

3 Hibridación de Metaheurísticas y Algoritmos Propuestos

En los últimos años, ha aumentado considerablemente el interés en las metaheurísticas híbridas en el campo de la optimización. Se han obtenido buenos resultados en muchos problemas de optimización clásicos y de la vida real utilizando metaheurísticas híbridas. Talbi en [15] y [16] propone una taxonomía para algoritmos híbridos y presenta dos clasificaciones para este tipo de algoritmos: jerarquizada y plana. Atendiendo a la taxonomía propuesta por Talbi, podemos decir que las hibridaciones propuestas en este trabajo se acercan a una hibridación de bajo nivel desde el punto de vista jerárquico y homogénea desde el punto de vista plano.

Para resolver el CVRP se utiliza el algoritmo MCMP-SRI [18] como algoritmo base, y dos hibridaciones que usan un operador de mutación basado en computación cuántica [5].

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. Además se lo utilizó el manejo de restricciones con metaheurísticas en [18] 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 $pi = (p1, p2, \ldots, p_n)$ 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 Q del vehículo. Se crea una población inicial de soluciones generadas aleatoriamente *(Stud(0))*, y luego estas soluciones son evaluadas. A continuación, la población pasa por un proceso multirecombinativo, donde el algoritmo crea un pool de apareamiento que contiene, padres generados aleatoriamente y un individuo semental, seleccionado de la población a través de selección proporcional. El proceso de crear descendientes funciona de la siguientre manera: el individuuo semental se aparea con cada uno de los padres. Las parejas se somenten a operaciones de recombinación y se generan 2 * n_2 (n_2 <= max_padres) descendientes. El mejor de los 2 * n_2 descendientes se almacena en un pool temporal de hijos. La operación de recombinación se repite n_1 veces (max_recombinaciones) hasta que el pool de hijos se complete. Finalmente, el mejor descendiente creado de n_2 padres y n_1 recombinaciones se inserta en la nueva población. El método de recombinación utilizado fue PMX (Partial Mapped Crossover) [9]: 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 1 se muestra el código del algoritmo MCMP-SRI.

```
MCMP-SRI

t=0; {Generacion Incial}
inicializar (Stud(t));
evaluar (Stud(t));
   while (not max_evaluaciones) do
   pool_aparemiento= Inmigrantes_generados_aleatoriamente ∪ Seleccionar(Stud(t));
                     while (not max_recombinaciones) do
                     evolucionar (pool_apareamiento);  {recombinación y mutación}
                     end while
   end while
   evaluar(pool_apareamiento);
   Stud(t+1) = seleccionar la nueva poblacion del pool_apareamiento
     t=t+1
   end while
```

Figura 1 Algoritmo MCMP-SRI

El algoritmo MCMP-SRI está basado en un AG y por lo tanto luego de la operación de crossover, el individuo pasa por una operación de mutación con una probabilidad muy baja. Los algoritmos propuestos usan una mutación basada en conceptos de computación cuántica, en particular qbit [5] y [10] donde se perturban los alelos de la siguiente manera: en el momento de realizar la mutación se elige una ventana de tamaño tres (este valor se seleccionó debido a que es un valor pequeño y controlado para la manipulación en el experimento). 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). Al algoritmo resultante se lo denomina MCMP-SRI-MC y su funcionamiento se muestra en la Figura 2.

El siguiente algoritmo propuesto se denomina MCMP-SRI-MCV y en este caso para la mutación cuántica se utiliza una ventana de tamaño tres (igual que en el algoritmo anterior) pero los elementos que forman la ventana se eligen en forma aleatoria y en posiciones que no corresponden a una cadena adyacente en el cromosoma. Esto permite una mayor diversidad en la explotación. Luego de elegido los elementos que conforman la ventana se realizan todas las combinaciones posibles y se queda con la combinación que obtenga el mejor fitness. La Figura 3 muestra un ejemplo de este mecanismo.

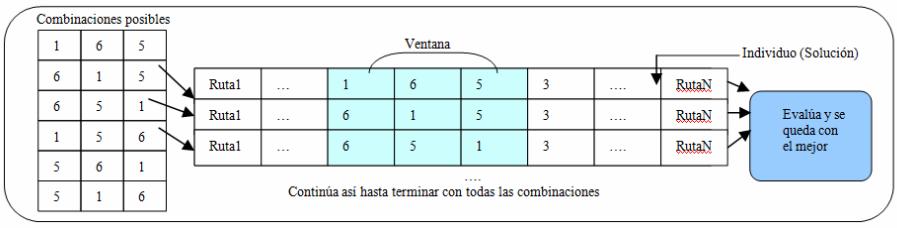

Figura 2 Mecanismo de MCMP-SRI-MC

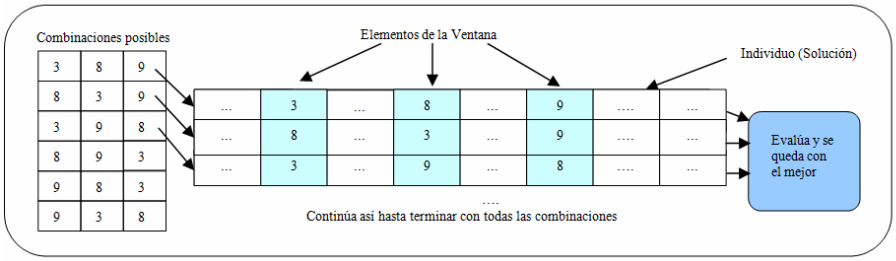

Figura 3 Mecanismo de MCMP-SRI-MCV

4 Diseño de experimentos y resultados

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

La batería de problemas para éste trabajo está constituida 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

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

clientes. La parametrización común utilizada en los algoritmos es la siguiente: la población se generó de forma aleatoria y se fijó en 15 individuos. El criterio de parada se estableció en 5000000 evaluaciones. En método de crossover utilizado es PMX y la mutación es de intercambio (en AG y MCMP-SRI). Se estableció la probabilidad de mutación en 0,05 y la probabilidad de recombinación en 0,65. El número n_1 (número de operaciones de recombinación) y n_2 (número de padres) se estableció en 16 y 18 respectivamente, estos parámetros se seleccionaron en base a la experimentación de los valores previamente usados exitosamente [18].

Todos los algoritmos se implementaron en Java y las ejecuciones se realizaron en un procesador 2,53 GHz Intel i5 bajo Windows 7.

En la Tabla 1 se muestra el resumen de las 30 corridas independientes para las 10 instancias seleccionadas. 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 una clara diferencia en los resultados obtenidos entre el algoritmo AG y las versiones multirecombinativas. Además los algoritmos propuestos obtienen para todas las instancias mejores resultados con respecto a la mediana y el error porcentual comparado con MCMP-SRI. La propuesta MCMP-SRI-MCV obtiene en 6 de las 10 instancias los mejores valores para las variables de performance analizadas (mediana y error porcentual).

Tabla 1 Resultados obtenidos por AG, MCMP-SRI, MCMP-SRI-MC y MCMP-SRI-MCV para las instancias analizadas.

		AG		**MCMP-SRI**		**MCMP-SRI-MC**		**MCMP-SRI-MCV**	
Instancia	Optimo	Mediana	Error	Mediana	Error	Mediana	Error	Mediana	Error
A-n33-k5	661	863,72	0,31	706,20	0,07	695,88	0,06	**695,37**	**0,05**
A-n33-k6	742	895,87	0,21	770,61	**0,04**	**765,43**	**0,04**	771,39	**0,04**
A-n34-k5	778	970,02	0,24	833,41	0,07	823,03	0,06	**813,21**	**0,05**
A-n37-k6	949	1139,65	0,21	1004,87	0,07	999,46	0,06	1005,42	**0,06**
A-n45-k7	1146	1396,02	0,22	1235,81	0,08	1233,50	0,07	1233,67	0,08
A-n53-k7	1010	1467,17	0,46	1177,63	0,16	1174,63	0,17	**1166,48**	**0,15**
A-n61-k9	1035	1449,55	0,41	1214,39	0,16	**1201,36**	**0,15**	1204,55	0,16
A-n62-k8	1290	1840,78	0,42	1487,72	0,16	1485,29	0,16	**1474,55**	**0,14**
A-n64-k9	1402	1881,99	0,35	1600,69	0,14	1596,97	0,14	**1572,27**	**0,12**
A-n80-k10	1764	2553,64	0,45	2072,06	0,18	2092,03	0,20	**2049,61**	**0,16**

Para realizar un análisis más profundo de los resultados únicamente se analiza el error porcentual. Comenzaremos verificando las condiciones necesarias para aplicar test estadísticos paramétricos o no paramétricos, estas condiciones son: Independencia, Normalidad y Homocedasticidad. Como los resultados a analizar provienen de corridas independientes sólo debemos verificar las otras dos condiciones. Para todos los test utilizados se obtiene el p-valor asociado, que representa la disimilitud de los resultados de la muestra con respecto a la forma normal. Por lo tanto, un p-valor bajo, señala una distribución no-normal. En este estudio consideramos un nivel de significancia $\alpha = 0,05$ por lo tanto un p-valor mayor que α indica que se cumple la condición de normalidad. Se utilizó como herramienta estadística SPSS.

En la Tabla 2 se muestran los resultados obtenidos por los test aplicados. El resultado del primer test aplicado, test de Kolmogarov-Smirnov, está compuesto por las cuatro columnas que representan los algoritmos analizados.
El símbolo "*" indica que los resultados no cumplen con la condición de normalidad y el valor a continuación representa el p-valor en cada caso.
En la ante-última columna se muestra el resultado de la aplicación del test de Levene para analizar la homocedasticidad. Para este caso el símbolo "*" indica que los resultados no cumplen la condición de homocedasticidad. La última columna representa el resultado de la aplicación del test de Kruskal-Wallis.

Tabla 2 Test de normalidad de Kolmogorov-Smirnov, Test de Levene y Test de Kruskal-Wallis.

	Test Kolmogorov-Smirnov				Test de Levene	Test de Kruskal-Wallis
Instancias	AG	MCMP-SRI	MCMP-SRI-MC	MCMP-SRI-MCV		
A-n33-k5	0,20	0,20	0,20	0,54	0,00*	(+)
A-n33-k6	0,20	0,03*	0,06	0,16*	0,00*	(+)
A-n34-k5	0,20	0,20	0,20	0,12	0,00*	(+)
A-n37-k6	0,20	0,06	0,19	0,19*	0,00*	(+)
A-n45-k7	0,13	0,20	0,19	0,20	0,00*	(+)
A-n53-k7	0,20	0,20	0,20	0,20	0,00*	(+)
A-n61-k9	0,20	0,20	0,20	0,20	0,00*	(+)
A-n62-k8	0,20	0,13	0,20	0,20	0,00*	(+)
A-n64-k9	0,20	0,20	0,20	0,20	0,00*	(+)
A-n80-k10	0,20	0,20	0,12*	0,20	0,00*	(+)

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. Utilizamos el signo (+) para especificar que existen diferencias significativas entre los resultados obtenidos por los algoritmos y (-) en caso contrario. 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. Luego se la aplicación de este último test únicamente se encontraron diferencias estadísticamente significativas entre el algoritmo genético (AG) y los versiones multirecombinativas.

En las Figuras 5 y Figura 6 se muestra cómo 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. En la Figura 5, para la instancia A-n64-k9, vemos que MCMP-SRI-MCV obtiene en mediana el menor error porcentual pero MCMP-SRI-MC es más robusto. En la Figura 6, para la instancia A-n53-k7, podemos observar que los valores obtenidos por MCMP-SRI-MCV son levemente menores a los valores obtenidos por los otros dos algoritmos multirecombinativos. Si bien el algoritmo es más robusto, no obstante, estas diferencias no son estadísticamente significativas.

Con la idea de analizar ahora el desempeño de los algoritmos sobre el conjunto de problemas aplicaremos un test no paramétrico (para muestras relacionadas). El siguiente test se ha 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.

Aplicamos 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. 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, que mantienen su identidad con respecto al conjunto de datos y la combinación de algoritmos a los que pertenecen. Seguidamente, se clasifican de 1 a k_n los algoritmos.

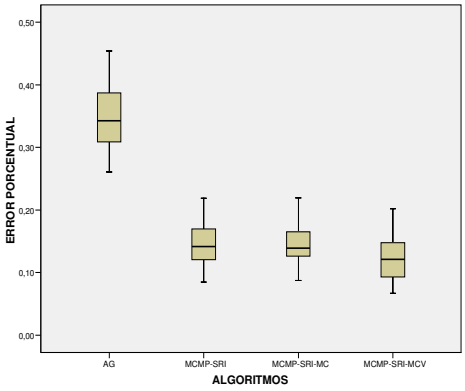

Figura 5 Ejemplo de Diagrama de Cajas (Boxplot) para la instancia A-n64-k9

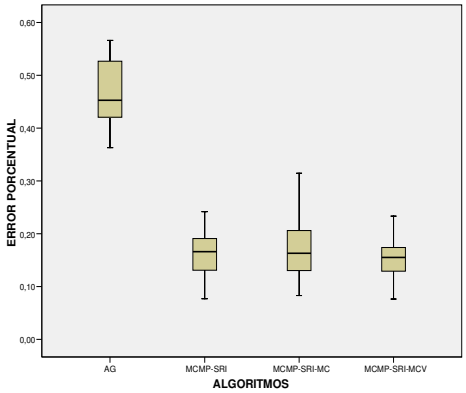

Figura 6 Ejemplo de Diagrama de Cajas (Boxplot) para la instancia A-n53-k7

Los resultados del estadístico de Friedman Alineado (distribuido de acuerdo a χ^2 con 4 grados de libertad: 7,76) son: AG ranking 3,89; MCMP-SRI ranking 2,28; MCMP-SRI-MC ranking 2,12; MCMP-SRI-MCV ranking 1,72. 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 nuestra propuesta algoritmo MCMP-SRI-MCV es el que obtiene el mejor ranking luego le sigue MCMP-SRI-MC, continua MCMP-SRI 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 buenas soluciones en una amplia clase de los problemas que de otra manera serían computacionalmente intratables. Para mejorar el funcionamiento de los AEs, se pueden usar enfoques multirecombinativos los cuales 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 que utilizan una mutación basada en conceptos de computación cuántica (MCMP-SRI-MC) y (MCMP-SRI-MCV). Hemos aplicado una familia de test no paramétricos para comparar los resultados (error porcentual) de los algoritmos y se realizaron comparaciones múltiples.

En cuanto a los resultados obtenidos, luego de aplicar el test no-apareado (Kruskal-Wallis) podemos afirmar que existen diferencias estadísticamente significativas entre los algoritmos en cada una las instancias analizadas. Al aplicar el test post-hoc (Tukey) confirmamos que las diferencias correspondían a los enfoques multirecombinativos con respecto al algoritmo AG. Seguidamente aplicamos un test apareado (Friedman Alineado). En Friedman Alineado, obtuvimos diferencias estadísticamente significativas en los resultados de los algoritmos. Con este test el algoritmo MCMP-SRI-MCV obtiene los mejores resultados y las diferencias son estadísticamente significativas. Trabajos futuros incluirán otras formas de hibridación y su aplicación a otras variantes de VRP.

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. El último autor agradece el constante apoyo de la Universidad Nacional de San Luis y a ANPCYT que financia su investigación.

Referencias

1. Alba, E. y Dorronsoro B.; Solving the Vehicle Routing Problem by Using Cellular Genetic Algorithms. Evolutionary Computation in Combinatorial: pp.1-10. (2004).
2. Bäker, B.M. y Ayechew M.A. A genetic algorithm for the vehicle routing problem. Computers & Operations Research, pp. 787-800. (2003).
3. Brandão, J. A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. Computers & Operations Research, 38(1), 140-151.(2011).
4. Christofides, N., Mingozzi A. y Toth P. The vehicle routing problem. Reveu francaise d´automatique d´informatique et de recherche opérationnelle. vol.1.tome10 (Combinatorial Optimization).pp.315–338.(1979).
5. Deutsch, D. Quantum theory, the church-turing principle and the universal quantum computer. Proceedings of the Royal Society of London A.400:97–117. (1985).

6. Fleszar, K. y Osman, I. H. A variable neighborhood search algorithm for the open vehicle routing problem. European Journal of Operational Research vol.195: pp.803-809. (2009).
7. Goel, A. y Gruhn, V. A General Vehicle Routing Problem. European Journal of Operational Research vol.191: 650–660. (2008).
8. Goldberg, D. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Co. (1989).
9. 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).
10. Han, K y Kim, J. Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. IEEE Trans. Evolutionary Computation, 6:580–593. (2002).
11. Huang S., Fu X., Chen P., Ge CC y Teng S. An Application Study on Vehicle Routing Problem Based on Improved Genetic Algorithm. Journal of Pervasive Computing and the Networked World. Lecture Notes in Computer Science vol. 7719. pp. 246-258. (2013).
12. Laporte, G. The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research vol.59 pp.345-358. (1992).
13. Lin S.W., Ying K. C., Lee C. y Lee Z.J.A Hybrid Approach for Vehicle Routing Problem with Time Windows. Journal of Computational Information Systems 7: 13 pp. 4939-4946. (2011).
14. Liu J., y Kachitvichyanukul, V. A New Solution Representation for Solving Location Routing Problem via Particle Swarm Optimization. Proceedings of the Institute of Industrial Engineers Asian Conference 2013. pp 103-110. (2013).
15. Talbi, E.-G. A taxonomy of hybrid metaheuristics. Heuristics, pp.541–564. (2002).
16. Talbi, E.-G.. Metaheuristics: From design to Implementation. Wiley. (2009).
17. Tavakkoli-Moghaddama R., Gazanfarib M., Alinaghianb M., Salamatbakhshc A. y Norouzic N. A new mathematical model for a competitive vehicle routing problem with time windows solved by simulated annealing. Journal of Manufacturing Systems. vol 30(2),pp. 83–92. Technical paper. (2013).
18. Villagra A. , Pandolfi D. y Leguizamón G. Handling constraints with an evolutionary tool for scheduling oil wells maintenance visits. Engineering Optimization. vol. 45(8), pp. 963-981. (2013).
19. Wang Y.P. Adaptive Ant Colony Algorithm for the VRP Solution of Logistics Distribution. Research Journal of Applied Sciences, Engineering and Technology 6(5):807-811, 2013ISSN:2040-7459. Maxwell Scientific. (2013).