ISSN 1870-4069

Algoritmo genético para el problema logístico de asignación de la mochila (Knapsack Problem)

Gladys Bonilla Enríquez, Diana Sánchez Partida, Santiago Omar Caballer Morales

Universidad Popular Autónoma del Estado de Puebla A.C., Puebla, Puebla, México

{gladys.bonilla, diana.sanchez, santiagoomar.caballero}@upaep.mx

Resumen. El presente trabajo aborda el caso de estudio de una empresa comercializadora nacional mediante el Problema de la Mochila o Knapsack (KP). Este problema logístico consiste en determinar los productos a almacenar que tengan la mejor relación costo-beneficio para la empresa sin afectar su capacidad de almacenamiento, por lo cual es un problema Binario Multi-criterio del tipo NP-duro. Si bien existen herramientas exactas que pueden resolver algunas instancias de este problema en la práctica pueden tener un costo significativo para la empresa limitando su adquisición. Debido a esto se desarrolló una propuesta de acceso libre de un Algoritmo Genético (AG) para resolver el KP de una manera factible para la empresa. El AG utiliza operadores de cruzamiento aleatorio, cruzamiento uniforme y mutación. El desempeño del AG fue evaluado con un valor óptimo, mostrando un porcentaje de error menor al 4% en comparación de una herramienta comercial exacta.

Palabras clave: algoritmo genético, Knapsack binario multicriterio, optimización logística.

Genetic Algorithm for the Logistic Knapsack Problem

Abstract. The present work addresses the case study of a national trading company through the Knapsack Problem (KP). This logistic problem consists on determining the products to be stored that have the best cost-benefit ratio for the company without affecting its storage capacity. This is the reason why the KP a binary multi-criterion problem of NP-hard complexity. While there are exact tools that can solve some instances of this problem, in practice these can have a significant cost to the company which may limit their acquisition. Due to this, a free-access proposal which consists of a Genetic Algorithm (GA) was developed to solve the KP in a feasible way for the company. The GA uses random crossing, uniform crossing and mutation operators. The performance of the GA was evaluated with an optimal value, showing a percentage error smaller than 4% when compared to an exact commercial tool.

Keywords: Genetic Algorithms, Binary Multi-criterion Knapsack Problem, Logistic Optimization.

1. Introducción

En el presente trabajo, el Problema de la Mochila o Knapsack (KP) Binario Multi-criterio (Martello & Toth, 1990) es abordado como estrategia de solución para una empresa que presenta un problema de optimización de su inventario para maximizar su ganancia sin afectar sus restricciones de capacidad. El modelo cuenta con una variedad de aplicaciones tales como la selección de proyectos que den la mayor ganancia al inversionista, determinar los productos que puedan caber en un contenedor para optimizar áreas de almacenamiento con espacios reducidos o la transportación de productos con característica específicas. En conjunto con otras técnicas, KP puede ser de gran apoyo para resolver distintas instancias en la industria.

El KP está clasificado como un problema combinatorio NP-duro en donde la complejidad computacional que clasifica los problemas como NP-duro o NP-completo indica que la diferencia entre problemas "factibles" e "intratables" se basa en el tiempo que un algoritmo podría tardar en dar una solución a un problema con *n* objetos (MacGregor & Ormerod, 1996). Es debido a esto que los problemas de optimización combinatoria sirven como modelos para un gran número de problemas reales y se estudian para construir algoritmos que sean efectivos en términos de complejidad y de la calidad de las soluciones devueltas (Angel, Bampis, & Gourvès, 2014).

Si bien existen herramientas computacionales exactas para resolver problemas combinatorios, estas pierden eficiencia conforme el número de variables o factores incrementa, requiriendo mayor tiempo y recurso computacional. De igual manera, herramientas comerciales pueden tener un costo significativo para su uso por parte de pequeñas y medianas empresas. Debido a esto se han desarrollado diversos métodos meta-heurísticos para tener alternativas más rápidas y alcanzables para este tipo de problemas. Algunos trabajos reportados en la literatura que han abordado el KP con diferentes enfoques son los siguientes:

- En (Mencarelli, D'Ambrosio, Di Zio, & Martello, 2016) se presentó una propuesta de algoritmo meta-heurístico para el KP múltiple no lineal, con funciones de ganancia y peso separables no convexas.
- En (Gao, Lu, Yao, & Li, 2017) se describió el problema KP multidimensional de opción múltiple (MMKP) como un problema importante de optimización combinatoria NP-duro con muchas aplicaciones. Los resultados experimentales mostraron que un algoritmo meta-heurístico desarrollado para la resolución del mismo pudo ser muy competitivo "reduciendo y solucionando" el KP en 18 de 37 casos.
- En (Avci & Topaloglu, 2017) se presentó el KP Múltiple Cuadrático (QMKP, por sus siglas en inglés) como una variante del problema KP clásico en el que se consideran múltiples mochilas y cuyo objetivo es maximizar una función objetivo cuadrática sujeta a restricciones de capacidad.

Entre los meta-heurísticos más utilizados para resolver problemas combinatorios se destacan a los Algoritmos Genéticos (AG) con las siguientes características:

- Son procesos adaptativos para la búsqueda de soluciones en espacios complejos inspirados en la evolución biológica (Herrera, Lozano, & Verdegay, 1995).
- Los AG permiten explorar una cantidad más amplia de posibles soluciones que los programas tradicionales (Holland J. H., 1992).
- Operan de forma simultánea con varias soluciones, en vez de trabajar de forma secuencial como las técnicas tradicionales (Vergara Canizales, 2005).
- Cuando se usan para problemas de optimización (por ejemplo, maximizar una función objetivo) resultan menos afectados por los máximos locales (falsas soluciones) que las técnicas tradicionales (Vergara Canizales, 2005).
- Usan operadores probabilísticos en vez de los típicos operadores determinísticos de otras técnicas (Vergara Canizales, 2005).

Dado que el KP es un problema con la característica NP-duro se decide abordar el problema con un AG. Estos algoritmos pueden dar buenos resultados con una selección de las mejores soluciones (individuos) a las cuales se aplican operadores genéticos para su diversificación. De esta manera se produce una nueva *población* de posibles soluciones, la cual reemplazará a la anterior verificando la propiedad de que la población siguiente posea mejores características en comparación con la población anterior. Así, a lo largo de diversas iteraciones o *generaciones* del AG, se espera que las buenas características se hereden (mantengan) a través de las poblaciones mejorando así la factibilidad de las soluciones (Scenna, Anaut, Passoni, & Meschino, 2013).

De esta manera el presente trabajo describe el desarrollo de un AG para resolver el KP de un caso de estudio. Uno de los objetivos del desarrollo de esta meta-heurística en una plataforma de libre acceso es que pueda proporcionar resultados cercanos a los obtenidos con sistemas restringidos por costos. El desarrollo del AG para la solución de la instancia considerada presenta más de una restricción y en su evaluación tuvo un desempeño significativo favorable al compararse con software especializado. Por lo tanto este desarrollo puede ser utilizado por pequeñas y medianas empresas con problemáticas que puedan modelarse mediante el KP y que tengan limitantes en cuanto a recursos o software especializado para resolverlo.

2. Marco teórico

El KP es un problema de tipo no-determinístico (NP) ya que existe una combinación exponencial de instancias que en su totalidad no pueden ser resueltas (NP-duro). Estos temas los aborda la teoría de la complejidad computacional que clasifica los problemas como problemas NP-duro y NP-completo, en donde la diferencia entre problemas “factibles” e “intratables” se basa en el tiempo que un algoritmo podría tardar en dar una solución al mismo con *n* objetos (MacGregor & Ormerod, 1996). El KP forma parte de una lista histórica de problemas NP-duros descrita en (Karp, 1972).

Dentro de los trabajos base que abordan el KP se encuentran aquellos reportados en (Martello & Toth, 1990), (Karp, 1972) y (Kellerer, Pferschy, & Pisinger, 2004). En (Iwama & Taketomi, 2002) se presenta el caso del KP en donde cada elemento a incorporar en la mochila tiene pesos y valores iguales. Otra variante del KP se presenta en (Disser, Klimm, Megow, & Stille, 2014) en donde la capacidad de la mochila es desconocida. En (Klamroth & Wiecek, 2000) se discuten diferentes modelos de KP como aquellos con múltiples criterios binarios con más de una restricción, con múltiples periodos y con dependencia del tiempo. El diseño de un algoritmo híbrido se presenta en (Carraway, Schmidt, & Weatherford, 1993) para el KP lineal estocástico en donde los costos se conocen con certeza, pero los rendimientos son variables aleatorias independientes normalmente distribuidas. En este caso, el objetivo es maximizar la probabilidad de que el rendimiento total sea igual o superior a un valor objetivo especificado. El trabajo reportado en (Angel, Bampis, & Gourvès, 2014) presenta de manera general los problemas multi-criterio en optimización combinatoria, identificando cuatro enfoques generales de la aproximación polinomial con garantías de rendimiento (enfoque de ponderación de criterios, enfoque simultáneo, enfoque presupuestario, y enfoque de la curva de Pareto). Finalmente, en (Kalaï & Vanderpooten, 2011) se presenta una versión robusta del KP utilizando un criterio de rechazo máximo-mínimo o mínimo-máximo. En este trabajo, se discuten los inconvenientes de dichos criterios y se propone un nuevo enfoque de robustez, denominado robustez-α lexicográfica. Se muestra que la complejidad del problema lexicográfico α-robusto no aumenta en comparación con la versión máximo-mínimo.

Dado que se conoce la complejidad matemática y computacional del KP, para el presente trabajo se considera la herramienta de Algoritmos Genéticos (AG). Esto dado que proporcionan una alternativa a las técnicas tradicionales de optimización mediante el uso de búsquedas dirigidas aleatoriamente para localizar soluciones óptimas en escenarios complejos (Srinivas & Patnaik, 1994). Los AG tienen su origen a principios de los 60's por Holland quien desarrolló los fundamentos de los AG basándose en los conceptos evolutivos y reproductivos de la teoría de Darwin. Esto en un intento por imitar de manera artificial el proceso natural de la evolución de los seres vivos (Holland J. H., 1975). Bajo este contexto, los AG permiten evolucionar a una población de individuos al someterlas a combinaciones aleatorias que se asemejan a las presentes en la evolución biológica, decidiendo cuál de estas combinaciones es más adaptable al medio que se plantea (en este caso para obtener una ganancia máxima) (Srinivas & Patnaik, 1994). La evolución de una población se obtiene mediante operaciones de reproducción conocidos como mutación y cruzamiento (Pedemonte, 2003). En las siguientes secciones de presentan los detalles del modelo KP y el AG desarrollado para su resolución.

3. Descripción del modelo KP

El modelo del KP parte de la suposición de que un escalador tiene que llenar su mochila. Para ello debe seleccionar una cantidad de varios posibles objetos, los cuales le pueden brindar el mayor beneficio sin exceder la capacidad de la mochila. El KP puede ser matemáticamente formulado con un vector de variables binarias x_j en donde:

$$x_j = \begin{cases} 1 \; Si\ el\ elemento\ j\ es\ selecionado, \\ 0 \qquad en\ otro\ caso. \end{cases} \tag{1}$$

Entonces, si p_j es una medida de importancia (en este caso, ganancia) para un elemento j, w_j representa el tamaño de dicho elemento, y cv es el tamaño de la mochila, el problema refiere a la selección de la cantidad de todos elementos cuyos vectores binarios x_j satisfagan las siguientes restricciones:

$$\sum_{j=1}^{n} w_j\, x_j \leq cv, \tag{2}$$

$$x_j \in \{0,1\}, j = 1, \dots, n, \tag{3}$$

que deben contribuir a maximizar la siguiente función objetivo:

$$\sum_{j=1}^{n} p_j\, x_j. \tag{4}$$

Para este caso se extendió el modelo con una restricción adicional de tamaño: *cv* se considera la capacidad volumétrica de la mochila y *cz* se considera la capacidad en peso de la mochila. De esta manera se tienen dos variables de tamaño para cada elemento: w_j el tamaño en volumen (cm^3) de cada elemento, y z_j el tamaño en peso (kg) del elemento, lo cual da la siguiente restricción:

$$\sum_{j=1}^{n} z_j\, x_j \leq cz. \tag{5}$$

4. Desarrollo de la implementación

El modelo KP se consideró para resolver un problema de almacenamiento para una empresa en México. Esta empresa tiene 250 productos y las siguientes restricciones de volumen y peso: capacidad = 169,750 m^3, peso = 545 kilogramos. La determinación de estos 250 productos se realizó mediante una clasificación del tipo ABC en base a su costo y frecuencia de pedido. En la Tabla 1 se muestran los datos de cada producto respecto a su ganancia (en pesos mexicanos), volumen (en centímetros cúbicos) y peso (en kilogramos).

Es importante señalar que el modelo KP normalmente es encontrado con una restricción. Para este trabajo son dos las restricciones que se consideraron de acuerdo al requerimiento del problema real.

4.1. Programación del AG

La Figura 1 muestra la estructura del AG desarrollado para resolver el KP del caso de estudio. La programación del AG fue realizada mediante el software Octave. Como se muestra, el AG opera sobre una población con *P* soluciones que se generaron aleatoriamente. Estas soluciones se llaman individuos y para cada individuo $i \in P$ se define una función que evalúa su adaptabilidad al entorno, es decir, que determine qué tan factible u óptimo es para resolver el problema (Olivera, 2004).

Tabla 1. Datos del problema. Fuente propia.

j	p_j	w_j	z_j	j	p_j	w_j	z_j	j	p_j	w_j	z_j	j	p_j	w_j	z_j	j	p_j	w_j	z_j
1	6985	400	3	51	6419	853	4	101	4746	581	3	151	7645	495	5	201	5947	802	5
2	5716	1272	4	52	4956	655	3	102	4245	556	5	152	5802	343	4	202	4051	361	3
3	5907	128	4	53	8256	1095	3	103	5275	1010	5	153	4171	1344	4	203	3662	1451	5
4	5177	751	4	54	5871	646	3	104	5843	482	5	154	5669	850	3	204	8479	955	5
5	8195	794	5	55	4724	547	4	105	4485	829	4	155	5544	878	3	205	8296	459	5
6	3818	1040	5	56	5123	608	3	106	4630	228	3	156	6195	137	5	206	9257	828	3
7	8681	405	3	57	5945	795	3	107	8093	1467	4	157	9357	1293	5	207	7623	603	3
8	5368	1414	4	58	4517	1427	4	108	6078	942	4	158	5747	434	4	208	3693	1283	5
9	7653	655	5	59	9383	433	4	109	3946	1298	3	159	9037	929	3	209	4039	465	5
10	5482	367	3	60	4109	1158	4	110	4592	568	4	160	5882	1110	5	210	6446	999	5
11	8004	381	5	61	6419	905	5	111	3887	433	5	161	6741	1514	5	211	8330	918	3
12	9112	924	3	62	5514	592	5	112	4598	983	5	162	8558	659	4	212	4086	1281	3
13	5139	233	3	63	4703	1380	4	113	7030	927	5	163	4990	301	4	213	4859	309	5
14	5725	1511	5	64	3860	1315	4	114	6486	821	5	164	6466	1295	3	214	3639	994	5
15	7826	1048	5	65	5849	1184	4	115	4781	1352	3	165	7762	394	3	215	8633	380	5
16	7816	1064	4	66	5472	182	4	116	3510	1485	4	166	9278	1419	5	216	9821	1043	4
17	4077	1078	5	67	3591	1426	3	117	3676	1081	5	167	6031	1197	4	217	9500	473	3
18	3757	220	4	68	8085	1240	3	118	5259	867	4	168	7314	1019	4	218	5692	377	5
19	5954	405	5	69	4069	1340	4	119	8452	326	5	169	8186	385	4	219	6582	471	3
20	3196	819	5	70	9395	1070	5	120	6836	479	4	170	7042	1277	5	220	7506	1231	3
21	3325	772	4	71	6675	1158	5	121	6952	1311	4	171	5084	918	3	221	8216	1485	5
22	4586	824	5	72	4398	1515	5	122	3152	866	4	172	9443	870	5	222	9808	328	5
23	4199	1216	4	73	9150	865	3	123	7021	543	3	173	3853	1270	4	223	6115	1497	4
24	5574	512	5	74	6864	606	5	124	7889	434	5	174	6946	1238	4	224	6003	1143	5
25	8184	122	4	75	8550	847	4	125	4897	1358	5	175	9542	860	5	225	9725	352	5
26	4157	465	3	76	9683	483	3	126	7558	1103	5	176	6595	1408	5	226	8048	367	3
27	6638	569	4	77	9768	1500	4	127	3448	384	4	177	7636	187	3	227	7906	1231	3
28	6873	680	4	78	9186	360	3	128	3227	329	4	178	3411	720	5	228	4915	795	3
29	5122	1491	5	79	8843	1287	4	129	6201	1535	3	179	9834	361	5	229	6784	1148	4
30	4681	965	5	80	4065	493	3	130	6030	451	4	180	6201	398	5	230	7021	349	3
31	5013	340	5	81	7292	1263	5	131	7667	603	4	181	3539	1025	4	231	4900	929	3
32	4586	892	5	82	3589	801	5	132	8840	1094	3	182	4753	359	4	232	6116	1420	5
33	3749	1081	5	83	9630	1277	4	133	8968	1109	4	183	5948	380	4	233	6999	386	3
34	6377	451	5	84	8381	486	3	134	7692	620	5	184	6402	631	5	234	4773	949	4
35	5081	1152	3	85	5997	917	4	135	6285	707	5	185	7857	1309	4	235	6674	1049	3
36	7413	286	4	86	4950	255	4	136	7344	856	5	186	8444	1447	4	236	6647	333	3
37	8087	451	5	87	5635	1114	4	137	9099	981	4	187	6921	642	5	237	4578	1159	5
38	4866	599	3	88	8505	516	3	138	3508	220	5	188	8901	1516	5	238	8897	839	5
39	7641	1105	5	89	6593	1409	4	139	3145	396	3	189	6428	712	5	239	5081	1121	4
40	8131	1077	5	90	8478	327	5	140	3330	716	5	190	9870	974	5	240	9644	650	4
41	9928	925	3	91	7032	1185	3	141	8565	1085	3	191	7180	405	4	241	3941	704	3
42	7961	745	3	92	3077	634	4	142	8017	1414	3	192	7132	1368	5	242	5099	607	3
43	9477	751	4	93	3134	452	4	143	5467	699	5	193	5557	880	5	243	8854	193	5
44	3635	214	4	94	8525	345	3	144	9030	1256	3	194	3106	572	3	244	7636	1368	4
45	3780	860	3	95	6369	152	4	145	3435	305	4	195	6303	1284	5	245	3431	683	4
46	4803	1083	4	96	7574	1313	5	146	9267	861	4	196	8695	735	3	246	6859	1092	5
47	3798	310	5	97	3346	1385	5	147	9195	437	5	197	6163	607	5	247	8933	1190	5
48	8216	1244	3	98	7667	253	3	148	7626	214	3	198	9333	380	5	248	7412	1511	3
49	3178	781	3	99	7013	1503	3	149	5160	1174	5	199	7857	813	3	249	8730	839	4
50	9705	388	4	100	7367	580	3	150	7619	1420	3	200	3922	1395	5	250	7354	754	5

En este caso, cada individuo representa un vector de asignación de elementos en la mochila x_j (vector de 1's y 0's) y la factibilidad del mismo se evalúa en base al valor de la función objetivo (Ecuación 4) con los respectivos valores de las variables x_j. Si la asignación cumple con las restricciones de capacidad (Ecuación 2 y Ecuación 5) la solución recibe el valor de la función objetivo correspondiente. Si no se cumplen dichas restricciones entonces la factibilidad recibe un valor de 0 dado que es un problema de maximización.

El AG se ejecuta durante 100 generaciones o iteraciones. En cada iteración se aplican los operadores evolutivos sobre un conjunto de i individuos denominados *padres* para recombinarlos y modificarlos con el propósito de generar nuevos individuos denominados *hijos* y constituir una nueva población. Los individuos *padres* se seleccionan mediante el operador probabilístico conocido como *Ruleta* el cual da privilegios a los individuos con mejor factibilidad en la población para su reproducción.

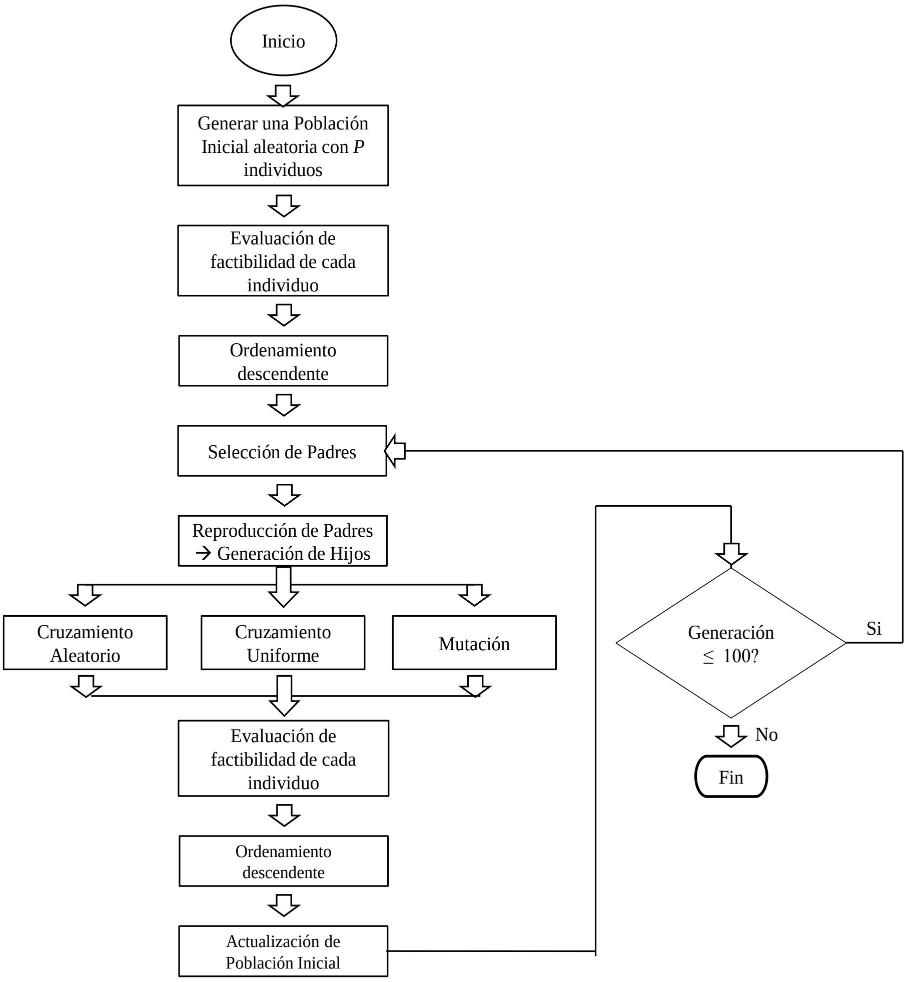

Fig. 1. Diagrama del AG. Fuente propia.

La manera en que los individuos *padres* se reproducen (recombinan) para generar nuevos individuos *hijos* es por medio de los operadores conocidos como *cruzamiento* y *mutación*. Usualmente los operadores de cruzamiento toman dos individuos *p1* y *p2* *padres* y generan dos individuos *h1* y *h2* *hijos* mediante la aplicación de la regla probabilística. Los operadores de mutación solo necesitan un individuo *padre* para generar un individuo *hijo*.

Una vez que se han generado los individuos *hijos* se hace la actualización de la población con los mejores *P* individuos existentes considerando a todos los individuos. Esta población actualizada será la nueva población para la siguiente generación o iteración del AG. Este proceso de depuración se presenta en la Figura 2.

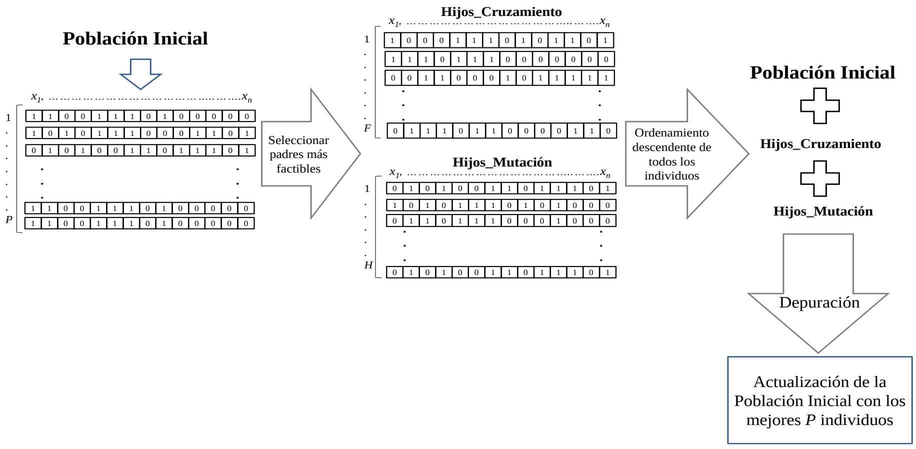

Fig. 2. Proceso depurativo de la población del AG.

A continuación se describen los operadores de cruzamiento y mutación del AG propuesto. Estos operadores fueron seleccionados dada la naturaleza binaria de la variable de decisión $\underline{x_j}$.

a) *Cruzamiento Aleatorio*: El cruzamiento se lleva a cabo entre Padre1 y Padre2 generando los hijos h1_cruzamiento y h2_cruzamiento. Los puntos de cruzamiento para la recombinación de información se seleccionan de manera aleatoria. La Figura 3 muestra el funcionamiento de este operador.

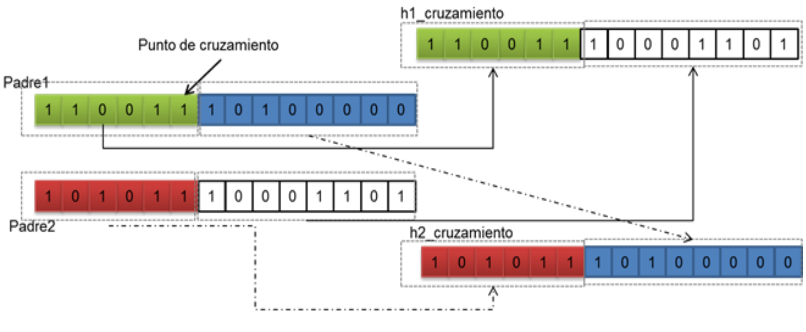

Fig. 3. Cruzamiento aleatorio. Fuente propia.

b) *Cruzamiento Uniforme*: Se genera aleatoriamente un vector plantilla del mismo tamaño que los padres (en este caso, de tamaño 250) con 1´s y 0´s. Partiendo de dos soluciones (Padre1 y Padre2) se generan dos hijos h1_uniforme y h2_uniforme de la siguiente manera con la plantilla generada (véase Figura 4):

- Para h1_uniforme: Si plantilla = 0 tomar el valor de Padre1, de lo contrario tomar el valor de Padre2
- Para h2_uniforme: Si plantilla = 0 tomar el valor de Padre2, de lo contrario tomar el valor de Padre1

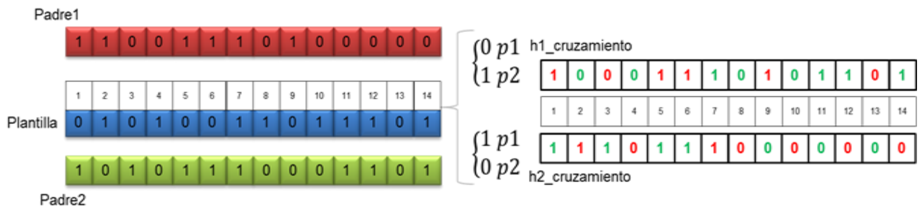

Fig. 4. Cruzamiento uniforme. Fuente propia.

c) *Mutación*: Para obtener un hijo h_mutación se toma un individuo Padre y se cambia de valor un elemento de x_j (de 0 a 1 y viceversa). Este proceso es ejemplificado en la Figura 5.

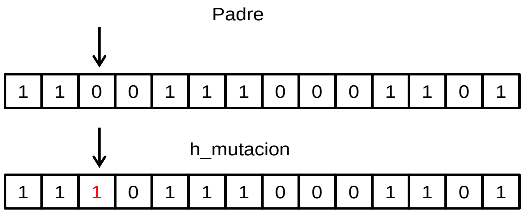

Fig. 5. Mutación. Fuente propia.

4.2. Comparación con Lingo

Para comparar el desempeño del AG se utilizó el software comercial Lingo. Esta es una herramienta de programación lineal que puede aplicar métodos como Branch & Bound para la resolución óptima de algunos tipos de problemas combinatorios (Martello & Toth, 1990).

Tabla 2. Comparativa del modelo exacto vs. AG. Fuente propia.

Comparativa		
	Lingo	AG Octave
	Óptimo	Factible
Ganancia	1093549	1058253
Productos	143	142
% Error		3.23%

La programación del modelo consiste en la maximización de la ganancia, considerando las restricciones de volumen y peso. La Figura 6 muestra los resultados obtenidos por Lingo que da una solución óptima del problema abordado.

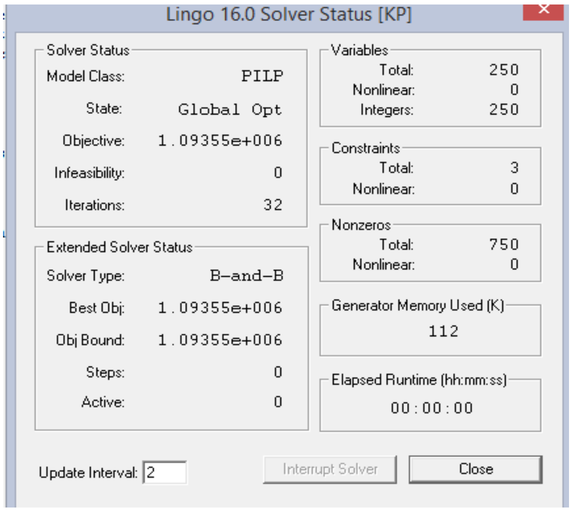

Fig. 6. Resultados con Lingo del KP del caso de estudio. Fuente Lingo.

De acuerdo al modelo de maximización de ganancia el software comercial Lingo da un resultado de 1,093,550.

5. Resultados

La Figura 7 muestra la convergencia del AG hacia un valor máximo de ganancia. En el eje horizontal se muestra la iteración o generación del AG en tanto que el eje vertical muestra el rango de la mejores ganancias totales considerando los productos asignados en cada iteración. Aproximadamente en la iteración 50 se alcanza el valor de convergencia que es de 1,058,253.

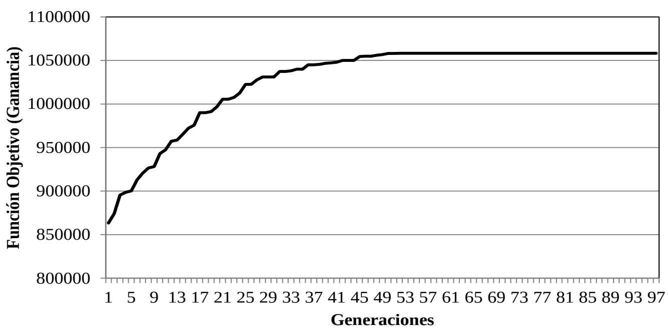

Fig. 7. Convergencia del AG. Fuente propia.

En la Tabla 2 se presenta la comparativa de los resultados de ambas herramientas. De acuerdo a la maximización de la ganancia con Lingo se obtiene un resultado de 1093590. En cambio, con el AG se obtiene una ganancia de 1058253. Esto es debido a que Lingo incluye 143 productos en tanto que el AG incluye 142 productos. Esta diferencia radica en un error de aproximación del 3.23%.

En la Tabla 3 se enlistan los productos a considerar de acuerdo a la solución proporcionada por el AG.

Tabla 3. Asignación generada por el AG. Fuente propia.

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
x_j	1	0	0	1	0	0	1	1	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	1	1
j	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
x_j	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	1	0	1	0	1
j	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
x_j	0	1	1	1	0	1	1	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	1
j	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
x_j	1	1	1	1	0	0	0	1	1	1	1	0	1	1	1	1	0	0	1	1	1	0	1	1	1
j	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125
x_j	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0
j	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
x_j	0	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	1	0	1	0	1	1	1	0	1
j	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
x_j	1	0	1	1	1	1	1	0	1	0	0	1	0	1	1	1	0	0	1	1	1	1	0	1	1
j	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200
x_j	1	1	0	1	1	0	0	0	0	1	1	0	1	0	1	0	0	0	0	0	1	0	1	1	0
j	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225
x_j	0	1	0	1	1	1	1	0	0	0	1	1	0	0	1	1	1	0	1	1	0	1	0	0	1
j	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250
x_j	1	1	1	0	1	1	1	1	0	1	1	0	1	1	1	0	1	1	1	1	0	1	1	0	1

6. Conclusiones

El AG brinda una solución factible de manera libre con un error de 3.23% a un problema con amplio número de productos (250) y dos restricciones de capacidad. Debido a que el problema de la mochila o KP es del tipo NP-duro, con instancias mayores un software de optimización comercial puede ser más costo y requerir más tiempo computacional que un AG.

Dado el error mínimo obtenido por el AG se puede llegar a la conclusión de que es una herramienta factible para que empresas con recursos limitados lo puedan utilizar para la optimización de sus inventarios. Como trabajo a futuro aún existe área de oportunidad para continuar mejorando el desempeño del AG con distintos operadores y extenderlo para considerar diferentes tipos de productos con dimensiones de volumen.

Referencias

1. Angel, E., Bampis, E., Gourvès, L.: Polynomial Approximation for Multicriteria Combinatorial Optimization Problems. 2nd Edition (ed V. Th. Paschos), Hoboken, NJ, USA: Wiley & Sons, Inc. (2014)

2. Avci, M., Topaloglu, S.: A Multi-Start Iterated Local Search Algorithm for the Generalized Quadratic Multiple Knapsack Problem. Computers and Operations Research 83, pp. 54–65 (2017)
3. Carraway, R. L., Schmidt, R. L., Weatherford, L. R.: An algorithm for maximizing target achievement in the stochastic knapsack problem with normal returns. Naval Research Logistics 40(2), pp. 161–173 (1993)
4. Disser, Y., Klimm, M., Megow, N., Stille, S.: Packing a knapsack of unknown capacity. In: Proc. of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS), pp. 276–287 (2014)
5. Gao, C., Lu, G., Yao, X., Li, J.: An iterative pseudo-gap enumeration approach for the Multidimensional Multiple-choice Knapsack Problem. European Journal of Operational Research 260(1), pp. 1–11 (2017)
6. Herrera, F., Lozano, M., Verdegay, J.: Algoritmos Genéticos: Fundamentos, Extensiones y Aplicaciones. Madrid: Arbor (1995)
7. Holland, J. H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Second Edition (1975)
8. Holland, J. H.: Algorimos Genéticos. Investigación y Ciencia, pp. 38–45 (1992).
9. Iwama, K., Taketomi, S.: Removable online knapsack problems. In: Proc. of the 29th International Colloquium on Automata, Languages and Programming (ICALP), pp. 293–305 (2002)
10. Kalaï, R., Vanderpooten, D.: The lexicographic α-robust knapsack problem. International Transactions in Operational Research 18(1), pp. 103–113 (2011)
11. Karp, R.: Reducibility among combinatorial problems. Proc. of Symposium on the Complexity of Computer Computations, pp. 85–103 (1972)
12. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack problems. New York: Springer (2004)
13. Klamroth, K., Wiecek, M. M.: Dynamic programming approaches to the multiple criteria knapsack problem. Naval Research logistics 47(1), pp. 57–76 (2000)
14. MacGregor, J. N., Ormerod, T.: Human performance on the traveling salesman problem. Perception & Psychophysics 58(4), pp. 527–539 (1996)
15. Martello, S., Toth, P.: Knapsack Problems: Algorithms and Computer Implementations. England: John Wiley & Sons Ltd (1990)
16. Mencarelli, L., D'Ambrosio, C., Di Zio, A., Martello, S.: Heuristics for the General Multiple Non-linear Knapsack Problem. Electronic Notes in Discrete Mathematics 55, pp. 69–72 (2016)
17. Olivera, A.: Heurísticas para Problemas de Ruteo de Vehículos. Montevideo, Uruguay: Instituto de Computación, Facultad de Ingeniería (2004)
18. Pedemonte, M.: Estudio Empírico de Operadores de Cruzamiento en un Algoritmo Genético Aplicado al Problema de Steiner Generalizado. RedUNCI-CACIC, pp. 216–624 (2003)
19. Scenna, F., Anaut, D., Passoni, L. I., Meschino, G. J.: Recofiguration of electrical networks by an Ant Colony optimization algorithm. IEEE Latin America Transactions, pp. 538–544 (2013)
20. Srinivas, M., Patnaik, L. M.: Genetic algorithms: a survey. IEEE-Computer 27(6), pp. 17–26, (1994)
21. Vergara-Canizales, V. G.: Aplicación de Algoritmos Genéticos en el balanceo de líneas de producción. Tesis, Reynosa, Tamaulipas, México: Universidad Autónoma de Nuevo León (2005)