APLICACION DE ALGORITMOS ESTOCASTICOS DE OPTIMIZACION AL PROBLEMA DE LA DISPOSICION DE OBJETOS NO-CONVEXOS

Israel Rebollo, Manuel Graña y Carmen Hernández, Departamento CCIA, UPC/EHU, España

RESUMEN

El problema de la disposición en el plano de objetos convexos, especialmente objetos rectangulares, ha sido tratado extensamente en la literatura y sus aplicaciones en el diseño VLSI, la industria del acero (*stock cutting*). La disposición de objetos no convexos ha sido menos estudiada y tiene aplicación en muchas industrias de diseño e impresión, especialmente la industria textil. Las aproximaciones matemáticas van desde la programación lineal hasta los más recientes algoritmos genéticos. En el presente trabajo consideramos un problema industrial en particular, el diseño de hojas de impresión de decorados para porcelanas. Estos objetos permiten inclusiones y otras situaciones geométricas que no aparecen en el caso de los objetos convexos. En el presente trabajo probamos los resultados sobre este problemas de varios algoritmos: un algoritmo heurístico, el algoritmo de enfriamiento simulado (*simulated annealing*) y un algoritmo genético.

MSC: 65U05.

ABSTRACT

The placement of convex objects in the plane, specially rectangular objects has been studied extensively in the literature and its applications to VLSI designs, the steel industry (*stock cutting*). The placement of non convex objects and its application in many design and textile industries, specially the textile industry has been less studied. Mathematical Approaches determining and adequate solution of such problems include linear programming and the newest genetic algorithms. In this paper we consider a particular industrial problem, the design of printing sheets for decorative porcelane. These objects permit inclusions and other geometric situations which do not appear in the case of convex objects. In this paper we proof results related with this problem for different algorithms: a heuristic algorithm, simulated annealing algorithm and a genetic algorithm.

1. INTRODUCCION

En el presente trabajo presentamos los resultados de la aplicación de algoritmos estocásticos de optimización combinatoria al problema de la disposición de objetos en el plano, en el marco de una aplicación industrial muy concreta. La aplicación consiste en el diseño de hojas de impresión de decorados (calcas) para piezas de vajilla de porcelana. La característica distintiva de esta aplicación es que los objetos son no convexos y se permiten inclusiones de objetos dentro de los "agujeros" de otros objetos. Respecto de las aplicaciones en la industria textil, del acero y del cristal (problemas de corte), nuestro problema presenta una superficie limitada de disposición y no existen restricciones de corte.

La función objetivo es la maximización del número de objetos que entran en la hoja. Esta función objetivo se formula aquí como la minimización del área utilizada para la disposición de los objetos, compactando la disposición lo más posible. Esta área se mide como la envolvente rectangular mínima del conjunto de los objetos una vez dispuestos en el plano.

Los algoritmos considerados son algoritmos estocásticos globales de optimización discreta que teóricamente producen óptimos globales de la función objetivo. Su convergencia al óptimo global está garantizada en el caso de tiempo de proceso infinito y bajo condiciones numéricas específicas. Sin embargo, en la práctica estas condiciones no son usualmente aplicables y es preciso probar la bondad de las aproximaciones obtenidas en condiciones realistas de cálculo. En concreto, hemos considerado el algoritmo de enfriamiento simulado [9] y un algoritmo genético [3]. Para el algoritmo de enfriamiento simulado, la convergencia al óptimo global está garantizada si la reducción de temperatura es suave y continua. Para el algoritmo genético la convergencia está garantizada si el operador de cruce y mutación permite explorar todo el espacio y la selección es elitista. Para constrastar los resultados prácticos aplicamos dos algoritmos de constrate: el cambio aleatorio y un algoritmo heurístico ajustado al problema. El camino aleatorio permite contrastar la mejora introducida por los algoritmos estocásticos respecto de la búsqueda no dirigida. El algoritmo heurístico proporciona soluciones subóptimas con un tiempo de cálculo más reducido, lo que permite valorar la mejora que proporcionan los algoritmos más sofisticados y generales.

1.1. Antecedentes y referencias

Los problemas de disposición en el plano son problemas de búsqueda sobre los que se han aplicado diferentes métodos a lo largo del tiempo. En concreto existen colecciones de implementaciones de los distintos algoritmos aplicados a este problema como el Timberwof 7, que consiste en un programa de optimización compuesto por diversos organismos y heurísticos destacando el Simulated Annealing; o el Genetic Workbench [4], que además de los algoritmos desarrollados en el Timberwolf 7 incluye algoritmos genéticos. En [5], se muestra la utilización del algoritmo de ramificación y acotación (*branck-and-bound*), con tiempo de cálculo exponencial en el número de objetos, lo que reduce su aplicabilidad práctica.

En [3] se plantea un problema de corte en la industria metalúrgica y se define un problema concreto en el que los objetos son ortogonales (OPP *orthogonal packing problem*). Esta aproximación utiliza un algoritmo genético y la estrategia BL *bottom left*, que es en la que se ha basado el heurístico construido para este problema. Macleod [2] crea un algoritmo específico para el problema de corte con guillotina. También podemos citar un artículo que muestra una solución más general utilizando programación lineal [1]. En [6] se afronta un problema de disposición de objetos no convexos, pero la formulación de la función objetivo y las restricciones consideradas no se ajustan al problema que tenemos entre manos.

1.2. Contenido del artículo

En primer lugar se plantea el problema en la siguiente sección. A continuación se presenta la formalización del problema haciendo hincapié en los aspectos relacionados con los algoritmos de búsqueda combinatoria a utilizar. Después se revisa la definición de los algoritmos utilizados: heurístico, enfriamiento simulado y algoritmos genéticos. Por último, antes de las conclusiones, se presentan los resultados de los experimentos realizados.

2. ENUNCIADO DEL PROBLEMA

Una empresa de porcelanas decoradas desea minimizar el espacio necesario para colocar decorados de la porcelana (calcas) en hojas de impresión a fin de reducir la cantidad de hojas necesarias para cubrir un pedido, lo cual reduce el coste asociado a los decorados.

Sea un pedido $\{o_i;\ i = 1,...,c\}$ donde o_i denota un decorado concreto. De cada decorado se necesitará una cantidad N_i. Existen varias empresas impresoras que ofrecen distintos tamaños de hoja. En adelante, T_k denota el k-ésimo tamaño de hoja de impresión. El problema que se nos plantea es minimizar la superficie ocupada por los decorados para de esta forma permitir la inclusión de nuevos decorados en la misma petición.

Las principales características y restricciones del problema son:

1. Las calcas son figuras no convexas, y en algunos casos tienen agujeros en su interior.

2. Las calcas se recortan para pegarlas en los objetos de porcelana, por lo que requieren una distancia mínima para el corte entre calca y calca. Esta distancia está implícita en la silueta de la calca; esto es, las siluetas de las calcas han sido dilatadas para tener en cuenta la distancia de corte mínima.

3. El área de la hoja de impresión es limitada y es un dato del problema. No se requiere la decisión de la empresa impresora.

El resultado del proceso de optimización es un valor de costo de impresión de las calcas, que en este artículo se corrresponde con el área ocupada, y la disposición gráfica de los decorados para su evaluación y posible mejora interactiva siguiendo criterios que son de difícil cuantificación (almacenamiento, estética, etc.).

4. FORMALIZACION DEL PROBLEMA

Parámetros:

1. Las calcas pertenecientes a un decorado: $[o_1,...,o_n]$.

2. El número de calcas que se necesitan. $N_i \triangleq$ número de calcas de tipo o_i.

3. El tamaño T de hojas disponibles, que dependerá del impresor seleccionado para realizar el pedido.

Espacio de búsqueda:

El espacio de búsqueda será toda combinación de calcas en hojas de forma que no se superpongan y que repeten las distancias mínimas de corte entre las mismas. Una solución del problema viene dada por la configuración de una hoja:

$$H = \{o_i, i = 1...n\}, \tag{3.1}$$

donde $o_i = ((x,y)_i, \theta_i, O_i)$ siendo (x,y) la posición del centro del objeto dentro de la calca, θ el ángulo de rotación respecto de la posición nominal y O la descripción del objeto dada por su centro geométrico y la polilínea que describe su silueta.

Una configuración es factible si no existen solapamientos entre los objetos. Dichos solapamientos se calculan como las intersecciones entre las siluetas de los objetos colocados en la disposición descrita por la configuración.

Sistema de vecinos:

Los vecindarios considerados en el camino aleatorio, el enfriamiento simulado y el algoritmo genético vienen dados por las posibles traslaciones y rotaciones de los objetos. En general, se calcula un vecino aplicando una traslación/rotación a uno de los objetos.

Dada una solución $H_j = \{o_i, i = 1...n\}$ del problema, la solución $H_{j+1} = \{o_i', i = 1...n\}$ es vecina de H_j si solo se diferencian en un único objeto trasladado y/o rotado: $\dot{\exists} u, o_u' = ((x,y)_u + \zeta, \theta_u, O_u)$ u $o_u' = ((x,y)_u, \theta_u + \Theta, O_u)$, donde ζ es el desplazamiento y Θ es la rotación. (Por razones prácticas, las orientaciones de los objetos están restringidas a múltiplos de 45°). Esta definición de vecindario es válida para todos los algoiritmos de búsqueda aleatoria que consideramos, puesto que en todos los casos, las soluciones alternativas a evaluar se generan por rotación/traslación de un objeto.

Función de coste

El coste considerado es el área ocupada por los objetos, definida como el área de la envolvente rectangular mínima de los objetos. La evaluación de la función de coste consiste en generar la disposición de los objetos a partir de la representación dada en (3.1) y calcular el área de la envolvente rectangular de los objetos en la imagen. En los algoritmos de búsqueda aleatoria se incluye un término de penalización para los solapamientos entre objetos. Estos solapamientos se calculan como intersecciones de las imágenes de los objetos. El óptimo buscado, obviamente, es una solución sin penalización que corresponde a una solución factible sin solapamientos.

Salida del algoritmo

El sistema devolverá el área ocupada por la solución encontrada y la disposición óptima de las caldas. $H = \{H = \{o_i, i = 1...n\}$.

4. ALGORITMOS UTILIZADOS:

En esta sección vamos a describir sumariamente los algoritmos aplicados a este problema: un heurístico específico para el problema, el enfriamiento simulado (*simulated annealing*) y un algoritmo genético.

4.1. Algoritmo Heurístico

Se ha utilizado un heurístico basado en la estrategia *bottom-left* (BL) que consiste en colocar los objetos lo más abajo y a la izquierda posible. El funcionamiento del heurístico se divide en dos partes: la preparación y la colocación propiamente dicha.

La preparación es un paso muy importante ya que la estrategia es determinista y crea las condiciones iniciales. Los objetos se ordenan según el tamaño del rectángulo mínimo en el que se puede inscribir el objeto.

La colocación, como ya se ha dicho, se basa en la estrategia BL. Se coloce el primer objeto lo más cerca posible del vértice inferior izquierdo de la hoja. Para colocar el siguiente objeto, se desplaza ligeramente a la derecha del objeto colocado justo en el paso anterior, pues a la izquierda y debajo de dicho objeto ya no existe la posibilidad de colocar más objetos. Se comprueba si hay solapamiento, si no lo hay el objeto se coloca y se pasa a colocar el siguiente. Si se produce un solapamiento entre el objeto y algún otro previamente colocado, se desplaza a la derecha y se comprueba nuevamente si se ha producido un solapamiento o no. Este proceso se repite hasta poder colocar el objeto o hasta alcanzar el borde derecho de la hoja. Si se ha llegado al borde derecho de la hoja, se desplaza el objeto lo más a la izquierda posible pero hacia arriba.

Este heurístico recibe unos objetos y debe colocarlos en la hoja. Si no es posible colocarlos, es decir, si se ha llegado con un objeto al extremo superior derecho de la hoja y no se ha podido colocar el objeto porque ese superpone con los que ya están colocados, entonces se rota noventa grados dicho objeto y se repite todo el proceso desde el principio con el objeto rotado. Si se vuelve a llegar al extremo superior derecho, entonces se considera que es imposible colocar dicho objeto en esa hoja y lo coloca en una segunda hoja. Cuando este hecho se produce, los siguientes objetos (si es que queda alguno para ser colocado) se intentan colocar desde el principio, en el extremo inferior izquierdo de la hoja.

Su mayor ventaja es que solo realiza una iteración y es determinista. La solución que se obtiene es subóptima pero bastante buena como se comprueba en los experimentos. Además, por construcción la solución es compacta y ordenada.

4.2. Enfriamiento Simulado (Simulated Annealing)

Se ha utilizado una implementación de este algoritmo siguiendo su descripción general. El algoritmo parte de una disposición inicial aleatoria de los objetos; esto es, de un cierto H(0) generado aleatoriamente. El algoritmo consiste en una sucesión de transiciones aleatorias $H(t) \rightarrow H(t+1)$ que constituyen una cadena de Markov no homogénea. [9] Cada transición aleatoria se descompone en dos pasos: la generación de la solución alternativa y la aceptacion de la misma. Formalmente:

$$P[H(t) \rightarrow H(t+1)] = P_G[H(t+1)|H(t)]P_A[H(t+1)|H(t)]$$

donde P_G denota la probabilidad de generación de la solución alternativa y P_A denota la probabilidad de aceptación de la misma. La probabilidad de generación sigue una distribución uniforme en los vecindarios de las soluciones. Esto es, para generar una solución alternativa, se selecciona aleatoriamente uno de los objetos y se desplaza y rota de forma aleatoria. La probabilidad de aceptación viene dada por:

$$P_H[H_1|H_2] = \exp\left(-\frac{C(H_1) - C(H_2)}{c}\right)$$

Si $C(H_1) > C(H_2)$ (estamos asumiendo un proceso de minimización de la función de costo). Si el costo de la nueva solución decrece, la probabilidad de aceptación es 1. El parámetro c es la temperatura que controla la posibilidad de escapar de subóptimos locales mediante “empeoramientos temporales” de la solución. Para valores grandes de este parámetro se aceptan todas las transiciones y para valores muy pequeños sólo las transiciones que mejoran la solución. Este parámetro se inicializa en un valor alto y se decrementa progresivamente mediante un factor α. Nosotros hemos aplicado una relación $c(t+1) = 0.97c(t)$ en los experimentos. Los algoritmos de ascenso de colinas(hill climbing) y paseo aleatorio pueden considerarse casos extremos del algoritmo de enfriamiento simulado para temperaturas cero e infinita, respectivamente.

Los parámetros básicos del SA son, en primer lugar, la solución inicial $H(0) \in \mathcal{H}$, donde $\mathcal{H}$ es el espacio de soluciones, c_0 la temperatura inicial, α el factor de modificación de la temperatura, (0.97 en nuestro caso), el tamaño de cadena *tam_max*, durante las cuales se mantiene constante la temperatura, y un número de iteraciones. La determinación de estos parámetros es empírica. En este caso, se ha utilizado una temperatura inicial de 1000 grados ya que dependiendo del valor de α se puede jugar más o menos con la temperatura. El valor α que se ha probado oscila entre 0.95 y 0.98. Pero finalmente se ha utilizado un valor de 0.97, ya que se ha llegado a un compromiso entre los extremos del intervalo utilizado. El tamaño de cadena máximo es uno, porque cada vez que se prueba una nueva disposición se modifica la temperatura. El número de iteraciones que se han intentado han ido desde 500 a 2000. Se ha podido observar que un número de 1500 iteraciones obtiene mejores resultados en relación con la calidad y tiempo de ejecución.

4.3. Algoritmos Genéticos

Los algoritmos genéticos (GA) son algoritmos de optimización que intentan emular la evolución natural. Partiendo de una población dada por un conjunto de individuos, se generan los posibles miembros de la siguiente generación mediante la aplicación de los operadores genéticos: el cruce y la mutación. El cruce consiste en: intercambiar partes de la estructura o genes de dos individuos para generar un nuevo individuo llamado hijo. La mutación consiste en modificar de forma aleatoria y con una cierta probabilidad los genes de un hijo en el momento de su creación. Ambos son operadores de exploración del espacio de soluciones. Para las soluciones generadas en cada momento, se evalúa su bondad mediante una función de ajuste (*fitting*). El operador de selección se aplica sobre los valores de estas funciones de ajuste para seleccionar de forma aleatoria los individuos que formarán la generación futura. Si se construye el operador de selección de forma que se asegure que el individuo de mejor función de ajuste sea siempre seleccionado, el algoritmo sera 'elitista' y puede demostrarse que converge en probabilidad al óptimo global.

En el artículo [7] se plantea una combinación entre algoritmos genéticos y enfriamiento simulado para un problema de disposición en el dominio del diseño VLSI.

En los algoritmos genéticos es fundamental representar el problema de forma que los operadores de cruce y mutación sean cerrados (el resultado siempre será una solución factible). En el problema concreto bajo estudio un individuo se representa como una tupla compuesta por las posiciones y orientaciones de las calcas

$$\textit{individuo} = \left((\textit{posi,orient})_i,\ i = 1...n\right).$$

Por lo tanto, la población es un conjunto de soluciones que pueden ser factibles o no, dependiendo de la existencia de solapamientos.

La función de ajuste es la función de coste descrita anteriormente con un término de penalización para los solapamientos y los desbordamientos de los objetos.

En este problema, los operadores genéticos son los más convencionales [3]. Esto es, la selección de los padres se realiza de forma aleatoria. Tras una generación se ordenan todos los individuos según el valor de su función objetivo y se seleccionan los mejores individuos que pasan a formar parte de la nueva población.

Para calcular un número mínimo de generaciones necesarias, se tendrá en cuenta la forma de selección de los individuos de una generación a otra. Si tras un número de generaciones ningún hijo ha conseguido formar parte de la población, quiere decir que los padres serán las mejores soluciones y por lo tanto el sistema para. Otra posibilidad es definir un número de iteraciones fija y parar cuando se llegue a ellas. Este último caso es el utilizado porque permite comparar los resultados con otros algoritmos en lo referente a los recursos de cálculo dedicados para obtener y una solución.

5. EXPERIMENTOS

5.1. Descripción, diseño experimental

Se han seleccionado un número de casos reales para los experimentos ocmputacionales dados por conjuntos reducidos de objetos. Se aplican los algoritmos de la siguiente manera:

- El algoritmo heurístico se aplica una sola vez.
- Los algoritmos estocásticos se replican 10 veces en cada caso para estimar el comportamiento promedio.
- Para los algoritmos aleatorios se fija un límite computacional que viene dado por el número de soluciones probadas, lo que determina el número de transiciones del enfriamiento simulado y del paseo aleatorio, y el número de generaciones y tamaño de población en el algoritmo genético.
- Los casos considerados son incrementables en el número de objetos considerado. Los conjuntos de objetos de los casos más sencillos están incluidos en los más complejos.

5.2. Resultados

Las Tablas que se muestran a continuación contienen los resultados de los experimentos computacionales. Se muestra, para cada algoritmo, el tiempo promedio de cálculo y el mejor valor obtenido de la función de coste, el valor y el peor. Cabe destacar que los experimentos se han realizado en IDL.

En la Tabla 1 se presentan los resultados obtenidos tras 1250 iteraciones (50 generaciones de 25 individuos) con 5 objetos y un tamaño de hoja de 70 x 50 centímetros. En la Tabla 2 se muestran los resultados para 10 objetos. En la Tabla 3 se presentan resultados con un juego de calcas distinto, compuesto por 14 objetos y un tamaño de hoja de 100 x 70 centímetros. En la Tabla 4 se presentan resultados con 24 objetos del primer juego elegido en una hoja de tamaño 100 x 70 centímetros.

Los resultados confirman la teoría del "*No-Free-Lanch*", esto es, ningún algoritmo es generalmente superior en todos los problemas. Se observa que el heurístico está bien ajustado al problema y proporciona resultados subóptimos de calidad en un tiempo de cálculo aceptable. Para tamaños de instancias del problema no grandes (5, 10 y 14 decorados) el tiempo de cálculo es reducido y explota para 24 decorados. Cabe destacar que gran parte del tiempo se dedica a la evaluación de las colisiones. Los algoritmos genéticos dan el mejor resultado para instancias no grandes, con el mayor tiempo de cálculo pese a que la complejidad en número de individuos probados es la misma para todos los algoritmos estocásticos. Para la instancia más grande, los algoritmos estocásticos son incapaces de competir con el algoritmo heurístico, dentro de las restricciones de tiempo de cálculo impuestas.

En cuanto a la calidad de las soluciones obtenidas, las Figuras 5.1, 5.2 y 5.3 muestran los resultados obtenidos respectivamente por el enfriamiento simulado, el algoritmo genético y el heurístico para una instancia de 14 decorados. El heurístico proporciona la solución más ordenada. En general todos mejoran a la búsqueda aleatoria.

Tabla 1. Resultado para el caso con 5 decorados.

Algoritmo	Tiempo promedio	Mejor valor	Valor promedio	Peor valor
Simulated Annealing	1:03:20	260	289	336
Algoritmo Genético	2:05:50	206	238	2656
Búsqueda aleatoria	0:27:26	506	634	699
Heurístico	0:00:39	330	330	330

Tabla 2. Resultado para el caso con 10 decorados.

Algoritmo	Tiempo promedio	Mejor valor	Valor promedio	Peor valor
Simulated Annealing	2:39:22	639	5223	16391
Algoritmo Genético	3:45:20	563	674	699
Búsqueda aleatoria	1:58:10	232239	377659	445368
Heurístico	0:12:15	690	690	690

Tabla 3. Resultado para el caso con 14 decorados.

Algoritmo	Tiempo promedio	Mejor valor	Valor promedio	Peor valor
Simulated Annealing	3:53:30	660	30430	97641
Algoritmo Genético	5:16:47	699	699	699
Búsqueda aleatoria	3:05:40	492792	801307	986472
Heurístico	0:15:58	695	695	695

Tabla 4. Resultado para el caso con 24 decorados.

Algoritmo	Tiempo promedio	Mejor valor	Valor promedio	Peor valor
Simulated Annealing	6:56:42	17013	108524	230499
Algoritmo Genético	7:46:40	215500	326835	392500
Búsqueda aleatoria	4:40:38	1033700	1201551	1292470
Heurístico	2:12:20	690	690	690

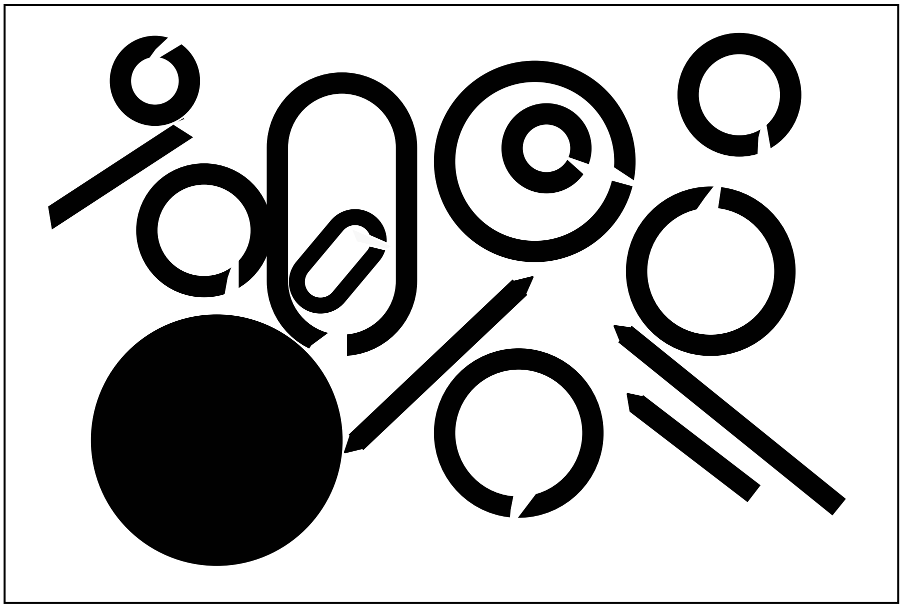

Figura 5.1.
Resultado del enfriamiento simulado con 14 objetos.

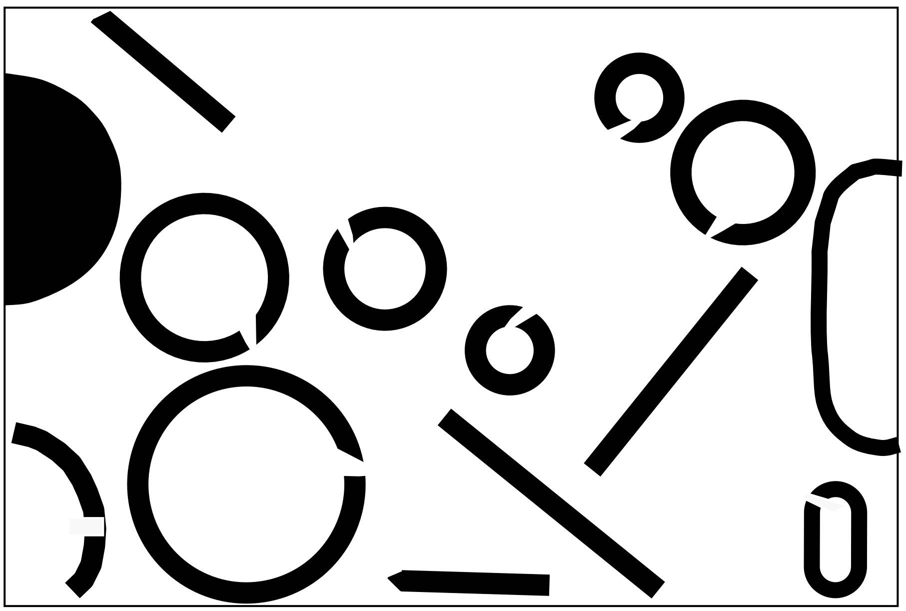

Figura 5.2.
Resultado del algoritmo genético con 14 objetos.

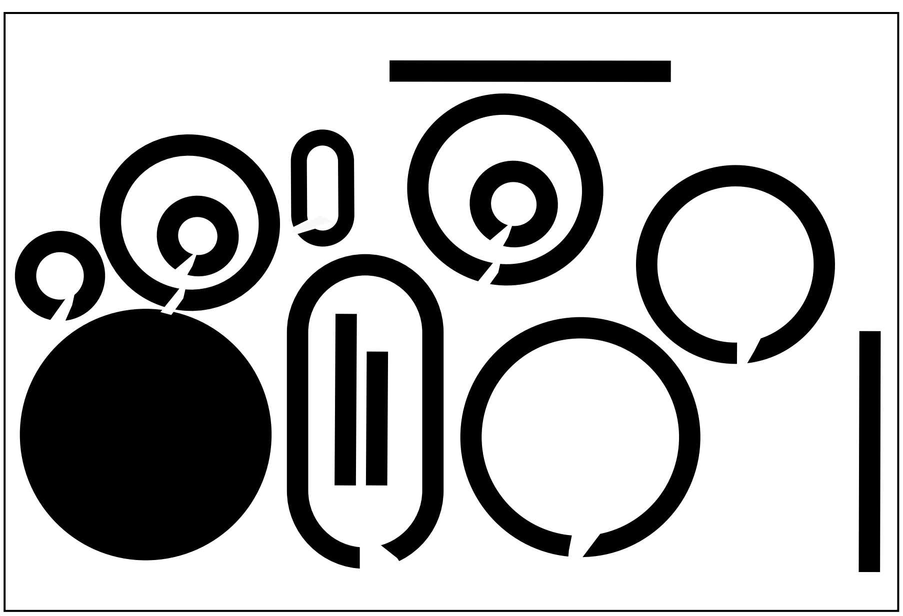

Figura 5.3.
Resultado del heurístico con 14 objetos.

6. CONCLUSIONES Y TRABAJO FUTURO

Hemos probado distintos algoritmos de optimización combinatoria sobre un problema industrial de disposición de objetos en el plano. El trabajo realizado confirma el interés del algoritmo heurístico que aporta con requerimientos computacionales reducidos soluciones subóptimas muy cercanas a las obtenidas por los algoritmos de optimización global.

En el futuro, se buscará una función de costo más realista y ajustada a las necesidades de la industria. También nos proponemos ampliar los resultados experimentales y el conjunto de lagoritmos utilizados.

AGRADECIMIENTOS

Este trabbajo se ha realizado en el marco del proyecto UE97/16 financiado por el Gobierno Vasco. Israel Rebollo disfruta de una beca predoctoral del Departamento de Industria del Gobierno Vasco. Estudio Atlas SL ha cedido las licencias para el software de desarrollo IDL.

REFERENCIAS

DEGRAEVE, Z. and L. SCHRAGE (1999): "Optimal Integer Solutions to Industrial Cutting Stock Problems", **INFORMS Journal on Computing**, 11(4), Fall1999.

MACLEOD, B.; R. MOLL; M. GIRKAR and N. HANIFI (1993): "An Algorithm for the 2D Guillotine Cutting Problem", **European Journal of Operational Research**, 68, 400-412, North Holland.

JAKOBS, STEFAN (1996): "On Genetic Algorithms for the Packing of Polygons", **European Journal on Operational Research**, 88, 165-181.

TURRINI, Silvio (1996): "Optimization and Placement with the Genetic Workbench", **DIGITAL Western Research Laboratory**, december..

ONODERA, H.; Y. TANIGUCHI and K. TAMARU (1991): "Branch and Bound Placement for Building Block Layout", 28^{th} ACM/IEEE Design Automation Conference, paper 25.3, 433-439.

WESTERLUND, T.; J. ISAKSSON and I. HARJUNKOSKI (1998): "Solving a Two-dimensional Trim-loss Problem with MILP", **European Journal on Operational Research** 104, 572-581.

KOAKUTSU, S.; M. KANG and W. WEI-MING DEI (1995): Genetic Simulated Annealing and Application to Non-slicing Floorplan Design", **UCSC-**CRL-95-52 november 18.

TAZAWA, I.; S. KOAKUTSU and H. HIRATA (1996): An Inmunity Based Genetic Algorithm and its Application to the VLSI Floorplan Design Problem", **IEEE** 3/96, 417-421.

AARTS, E. and J. KORST (1989): "Simulated Annealing and Boltzman Machines", John Wiley and Sons.