Algoritmo para la secuenciación en el ensamblaje de estructuras mediante robots

S. Vera
Universidad de Sevilla, Sevilla, España, svera@us.es
I. Maza
Universidad de Sevilla, Sevilla, España, imaza@cartuja.us.es
A. Ollero
Universidad de Sevilla, Sevilla, España, aollero@cartuja.us.es

Resumen

Los algoritmos de secuenciación son el punto crítico en la mayoría de los sistemas de planificación, teniendo todos ellos una fuerte componente combinatoria. En este artículo se propone un nuevo método, un enfoque alternativo del clásico método de búsqueda en espacio de estados, para su aplicación en el ensamblado mediante robots aéreos que se contempla en el proyecto FP7 ARCAS (Aerial Robotics Cooperative Assembly System). Se presenta un algoritmo de búsqueda iterativa acotada basada en heurística al que se ha denominado Iterated Heuristic Search (IHS) comparandolo con otros algoritmos de búsqueda heurística y poniendo de manifiesto su escalabilidad y bajo tiempo de cómputo.

1 Introducción

En este artículo se estudia el ensamblado de una estructura general compuesta por piezas independientes que forman un todo. A veces esta estructura puede componerse a su vez por otras subestructuras menores que se componen de piezas atómicas. La descomposición de la estructura se debe aplicar varias veces siguiendo un determinado orden para que sea posible, partiendo de la estructura completa, llegar a las piezas atómicas que la forman. Este orden, denominado secuencia de ensamblado, es necesario tanto para el ensamblado como para el desensamblado.

En el cómputo de esta secuenciación es necesario tener en cuenta tanto las restricciones propias del ensamblado de la estructura, como las restricciones geométricas del entorno y de los robots que se dediquen a su ensamblado. Esta secuenciación es una fase crítica del problema completo que es la planificación de ensamblado en robótica, que incluye otras tareas como la localización, el agarre de las piezas, navegación, resolución de errores, etc.

La secuenciación tradicionalmente se ha tratado como un problema combinatorio determinando todas las posibles combinaciones en la secuencia que hace que la estructura sea ensamblada satisfactoriamente, respetando principalmente las precedencias entre todas las piezas. Se han propuesto tambien algoritmos que, además de estudiar de todas las secuenciaciones satisfactorias, selecciona las mejores de acuerdo con un criterio determinado. Estos algoritmos se denominan en [1] *Three step approach* caracterizándose por tener una algoritmia de tres procesos que comprende la definición de precedencias entre piezas, la generación de todas las posibles combinaciones, y finalmente la elección de la mejor secuencia.

El problema de planificación de la construcción de una estructura mediante robots se puede abordar con una arquitectura en 2 niveles como en [2] que permite considerar en la planificación la accesibilidad y las restricciones de colisiones. El esquema que sigue consiste en un primer nivel en el que se resuelve el problema de navegación local, y un segundo nivel relacionado con la propia secuenciación del ensamblaje.

Los algoritmos de secuenciación más clásicos son los que se basan en la búsqueda dentro de un grafo estados, pero al ser este problema np-completo, estos algoritmos se caracterizan por ser muy poco escalables.

Para espacios de estados mayores se han aplicado técnicas como algoritmos genéticos o redes neuronales. En el siguiente apartado se hará un estudio de los distintos algoritmos que se pueden utilizar para este propósito. En el siguiente se hará un análisis del problema de ensamblado, introduciendo un tipo de representación acorde con el tipo de resolución algorítmica que se le da al problema en este artículo. A continuación se presenta el algoritmo Iterated Heuristic Search (IHS) desarrollado para resolver problemas de secuenciación, presentando sus ventajas e inconvenientes y comparando con otros algoritmos de búsqueda de estados, como el algoritmo Backtracking o el algoritmo A* [3] utilizado en [4].

2 Representación y complejidad de un ensamblaje

El método más común para la representación de un ensamblaje consiste en una definición de relación entre las propias piezas que forman la estructura (piezas atómicas) y los denominados subensambles (combinaciones de piezas). La representación de cada pieza atómica de la estructura se puede definir con una etiqueta, un símbolo o un nodo, mientras que los subensambles se definen mediante enlaces (relaciones) entre las piezas. Esta relación se suele modelar mediante listas de restricciones o mediante grafos dirigidos (grafo de precedencias) en el caso de una representación gráfica.

La representación que se va a seguir en este artículo se denomina "Diagrama de enlaces" que consiste en un binomio formado por las piezas constituyen un ensamblaje y su grafo de relaciones, donde los nodos son las piezas del ensamblaje y las aristas son las relaciones de precedencia entre dichas piezas que conforman el ensamblaje. Partiendo de este grafo se puede obtener de forma directa una matriz de enlaces [5] donde cada fila y columna de la matriz corresponde a cada uno de los nodos, y los elementos propios de la matriz serán 1 si existe una relación de dependencia o 0 si no. En la figura 1se puede ver un ejemplo.

En este artículo se considera el ensamblado de estructuras teniendo en cuenta la estabilidad de cada pieza definida por su posición de reposo y condicionada por su centro de gravedad, así como la estabilidad de las subestructuras que se van construyendo en cada iteración del problema.

Por ejemplo, En la figura 1 se puede ver que la relación de dependencia A -¿ B, C que se observa en el grafo de restricciones del ensamblaje, equivale al modelado de la restricción física de la estructura que dice que, para colocar la pieza A y obtener un estado estable, es necesario partir de una estructura estable donde se encuentren colocadas las piezas B y C. Si no fuese así se llegaría a un estado inestable y se destruiría la estructura durante su construcción.

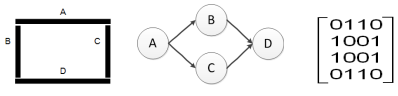


Figure 1: Representaciones de un ensamblaje

Otra característica de estas estructuras es que la posición de cada pieza en la estructura es única y la secuenciación propia de ensamblado es pieza a pieza, aunque también se tiene en cuenta más adelante la posibilidad de un escenario multi-robot y realizar una secuenciación tal que permita ensamblar la estructura con varias piezas a la vez trabajando varios robots en paralelo y reducir así el tiempo de ensamblado.

Además de las restricciones propias de la estructura a ensamblar, se puede modelar otro subconjunto de restricciones tales como las geométricas en [6], las restricciones propias de la asignación de recursos [7], o los agarres de piezas y fijaciones como en [7]. Todas estas restricciones se pueden introducir en el modelo y generar una secuenciación final más completa. En el caso multi-robot también se pueden modelar algunas restricciones propias del desplazamiento de las distintas piezas con distintos robots cooperativos. En todo caso el enfoque más utilizado en ensamblado robótico es el empleo de un sistema jerarquizado con dos niveles [2].

Complejidad.

El método general de resolución de la secuenciación consiste en construir una representación tal que determine la relación de precedencias entre las distintas piezas y que esta representación a su vez exprese la evolución del ensamblado de la estructura. De todas las posibles codificaciones de secuencias, se desestimarán aquellas que no son físicamente construibles siguiendo dicha secuencia. El método de representación más extendidoes mediante estados de ensamblaje [6] donde la relaciones de precedencia se denominan condiciones de establecimiento.

Los algoritmos de secuenciación de estados de ensamblaje generalmente están parametrizados por el numero n de elementos que componen la estructura a ensamblar. Sin embargo, esta medida no expresa literalmente cómo de difícil es obtener una secuencia de construcción válida. Existen otras características que parametrizan el problema del ensamblado como es la monotonía (si la estructura se monta pieza a pieza, o es necesario construir previamente varias subestructuras para ensamblar la estructura final) o la linealidad (si para ensamblar una determinada pieza sólo depende de sí misma, o de otro conjunto de piezas). Algunos autores como [8], [9] introducen también otras características en esta clasificación, como son los grados de libertad de un robot o las necesidades de fijaciones de las piezas.

En la figura 2 se muestra el tipo de estructuras monótonas y lineales que se pretende ensamblar. Su construcción es secuencial incremental; es decir, cada estado que caracteriza un instante concreto del ensamblaje siempre será incremental (dispondrá de una pieza más) que el estado predecesor.

La evolución de la secuenciación se puede representar mediante un grafo tipo árbol binario invertido, como se aprecia en la figura 2, donde en cada nivel se observa siempre un subensamblaje estable y una determinada pieza de la secuenciación. El último nodo del árbol representa a la estructura completa.

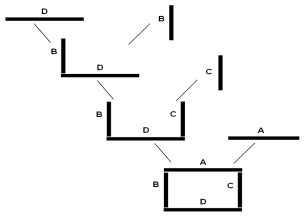


Figure 2: Ejemplo de secuencia lineal y monótona

El algoritmo de secuenciación una vez que trata todas las posibles secuencias construibles, analiza la mejor secuencia de acuerdo con algún criterio de optimización. Este criterio dependerá de lo que se considere más importante en cada caso. Por ejemplo, en el caso de varios robots trabajando en paralelo para ensamblar una misma estructura, un buen criterio a seleccionar sería la paralelización de la secuencia de ensamblado, teniendo en cuenta el número de robots que están trabajando en la construcción de la estructura. Otro criterio sería, por ejemplo, la minimización de consumo de energía de los robots. El coste computacional del cálculo de estos criterios es relevante en la complejidad final del algoritmo, ya que por lo general se ejecutan en el núcleo del algoritmo. En [10] se puede ver que la complejidad de la secuenciación optima se aproxima a la búsqueda exhaustiva, que en una estructura de n piezas equivaldría al número de permutaciones de n elementos (n! combinaciones) considerando estructuras cuyo ensamblaje es lineal y monótono, como las que se tratan en este artículo, ya que si el ensamblado no fuese lineal, la cota de complejidad ascendería a $^{(2n-2)!}/_{(n-1)!}$ según [10].

3 Algoritmos de secuenciación

Los algoritmos de búsqueda en un espacio de estados determinan una secuencia para ensamblar un objeto o encuentran la mejor secuencia de ensamblado de acuerdo con un determinado criterio. El método más usado es el de la búsqueda exhaustiva en grafos de estado. En la práctica, teniendo en cuenta la complejidad tratada en la sección anterior, este método solo puede aplicarse en ensamblajes de pequeñas estructuras. El empleo de heurísticas permite disminuir de forma importante la complejidad. El algoritmo A* [3] garantiza una solución de mínimo coste, pero esta solución está condicionada por la bondad de la heurística, teniendo en cuenta que el algoritmo devuelve la primera solución que encuentra.

Debido a la explosión computacional que se puede dar a partir de un numero n de piezas, se han presentado métodos que mejoran la escalabilidad de este problema tal como el algoritmo de recorrido simulado [11][12], que consiste en intercambiar dos piezas arbitrariamente de forma iterativa, y computar la función de energía correspondiente a la secuencia obtenida aceptando la nueva secuencia si la probabilidad de Boltzmann es mayor que una determinada cota. Se aplican también algoritmos genéticos [13][14] codificando el cromosoma como una secuencia de ensamblado, con lo que, tras varias generaciones del algoritmo, se obtiene un resultado cercano al óptimo, o las redes neuronales [15][16] definiendo una función de energía relacionada con los valores de entrada y salida de la neurona, iterando para depurar el comportamiento de la neurona.

El algoritmo IHS que se presenta a continuación es también un algoritmo de búsqueda en el espacio de estados comparable al Backtracking ya que realiza una búsqueda exhaustiva por todo el árbol devolviendo la mejor solución encontrada, con la condición que el IHS es un algoritmo acotado, y siempre da una solución en una cota de tiempo deseada. Este algoritmo se adapta muy bien al problema de secuenciación de ensamblajes, y en particular al montaje mediante robots de estructuras monótonas y lineales. Este algoritmo tiene entre sus propiedades que es escalable, lo que no se puede decir de ningún otro algoritmo basado en búsqueda de estados, ya que dispone de una metodología adaptativa que reajusta sus parámetros automáticamente para garantizar la cota de tiempo establecida.

4 Algoritmo Iterated Heuristic Search (IHS)

Este algoritmo se suele utilizar para dar solución al problema de satisfacción de restricciones (PSR). El método consiste en modelar primero el problema en estudio como un problema de espacios de estado. Partiendo de un estado inicial, que representa la configuración inicial del problema, se crea un grafo tipo árbol de decisiones donde cada nodo es un estado intermedio, y los arcos del árbol representan los operadores que se han aplicado para modificar el estado anterior. A continuación se hace una búsqueda sobre el árbol de los estados finales que representa una solución. Los operadores están relacionados con las acciones atemporales que se pueden realizar en un problema de satisfacción de restricciones. Estos op-

eradores también están directamente relacionados con la ramificación del grafo árbol que representa el problema. Este algoritmo implementa una búsqueda en profundidad acotada, de manera que a priori podría poner en juego la completitud del algoritmo, en el caso que el estado solución se encuentre a un nivel del árbol inferior al de la cota de búsqueda. El algoritmo se hace completo gracias a un proceso iterativo en el que se van obteniendo soluciones parciales del problema hasta que se llega a un estado final completo. Este comportamiento combinado es el núcleo del razonamiento que evita la explosión computacional.

4.1 Introducción

El algoritmo Iterated Heuristic Search (IHS) surge de la unificación de los algoritmos de búsqueda en profundidad acotada y de búsqueda informada con heurística. El primero introduce el núcleo propio del algoritmo limitando el nivel de la búsqueda en profundidad hasta un determinado nivel denominado cota. El segundo introduce un mecanismo de heurística para mejorar el proceso de búsqueda, e introduce la valoración de los estados intermedios que se convierten en finales cuando llegan al nivel de la cota. Por tanto, la ejecución de esta parte del algoritmo (núcleo) se repite el numero de veces que sea necesario hasta llegar a un estado solución del problema.

Las características del algoritmo de búsqueda ciega en profundidad iterada que establecen los órdenes máximos del IHS son los siguientes:

- Complejidad en tiempo:$\mathcal{O}r^c$

- Complejidad en espacio:$\mathcal{O}\left(r \times c\right)$

- Completitud: Es completa.

- Obtención de una solución con el mínimo número de operadores (óptimo).

En las expresiones anteriore r es el factor de ramificación y c es la cota de la profundidad

La búsqueda en profundidad iterativa es un algoritmo muy recomendado para problemas donde el espacio de búsqueda es grande y no se conoce la profundidad de la solución. La redundancia en el proceso de expansión del algoritmo se ve compensada con la completitud. Además, la redundancia no es significativa ya que supone sólo un 10 % más del coste propio del algoritmo, lo que tiene interés especial ya que en el ensamblado de estructuras los estados finales del proceso de secuenciación se encuentran todos en el último nivel del del grafo de búsqueda, donde se completan todas las secuencias. El proceso iterativo es el responsable de que en cada iteración del algoritmo se obtenga una solución parcial del problema lo que permite avanzar el proceso de construcción mediante un robot.

Algorithm 1 Núcleo del Algoritmo IHS

```
búsqueda-recursiva(nodo, estructura) {
if (prof == 0 OR esEstadoFinal(nodo)) {
return funEvaluacion(nodo); }
else {
sucesores(nodo, listaSuc);
return maximizador(listaSuc, prof); }
```

4.2 Descripción del problema de construcción.

Se pretende ensamblar mediante un robot estructuras como las que se pueden ver en la figura 3 compuestas por n piezas (barras) que se apoyan unas sobre otras. Las piezas están etiquetadas con un número. Las restricciones se dan mediante una lista de dependencias para simplificar la implementación. En principio no se tienen en cuenta la posición inicial y final de cada pieza.

La forma de representar un estado del problema de manera atemporal y genérica será mediante un vector de piezas (etiquetas de las piezas) que representa la misma secuenciación, un índice o traza que indica la evolución de los estados, el número total de piezas que aún quedan por ensamblar y, por último, una valoración del estado. Un ejemplo de estado inicial y final sería:

- Inicio = {0, {-10}, numPiezas, -10}

- Final = {9, {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}, 0, 65}

En el estado inicio se observa un indice de estado 0, que representa el primer nivel de decisión, una secuencia "vacía" por defecto {-10}, el número total de piezas de la estructura y una valoración inicial irrelevante: -10. En el estado final se observa un índice 9, que es el último nivel de la secuenciación, una secuencia, un número 0 de piezas por colocar y una valoración de la secuencia.

El núcleo del algoritmo IHS que se dedica a realizar la propia búsqueda en el árbol es el siguiente:

El algoritmo realiza una búsqueda recursiva expandiendo el estado. La función "funEvaluación" es la encargada de la evaluación tanto de los estados finales como los estados intermedios que llegan al límite de su cota. El proceso iterativo consiste en la ejecución del algoritmo búsqueda-recursiva tantas veces como sea necesaria hasta encontrar una solución completa. El proceso iterativo tendrá la misma condición de parada "esEstadoFinal".

Definición y ajuste de la cota.

El algoritmo dispone de un único parámetro configurable, el cual juega un papel principal en la eficiencia del algoritmo, que es la cota del algoritmo. El ajuste de este parámetro debe hacerse teniendo en cuenta su significado, su impacto en la eficiencia, y sus consecuencias en los resultados. En la práctica se dispondrá de un mecanismo de cota adaptativa, de manera que la cota se ajustará a cada problema de forma transparente al usuario y se respetará el tiempo de cómputo máximo establecido.

En este problema de secuenciación se observa que a medida que se va bajando en el árbol de espacio de estados se va reduciendo la ramificación de cada nodo e incrementando el tamaño de la secuencia resultado hasta que llega al nodo hoja donde se encuentra la solución. Notese que todas las soluciones se encuentran en el último nivel del árbol.

Teniendo en cuenta que este problema es claramente np-completo es necesario poner una cota que si es demasiado pequeña puede generar soluciones parciales erróneas. Observe también que aunque se use una cota para reducir el cómputo, el índice de ramificación de los niveles más altos es mucho mayor que el de los niveles inferiores, por lo que las primeras iteraciones serán más críticas que en el resto. Esta peculiaridad también justifica el uso de cotas, ya que aunque se corta la búsqueda, el algoritmo se centra en la zona más importante del árbol, donde hay más decisiones mientras que las partes más bajas del árbol se dedican al ensamblaje del resto de piezas (pocas) que quedan por poner.

4.3 Heurística utilizada

Después del estudio y modelado del problema, para evaluar cada estado (final o no) se ha optado por el diseño de una heurística que concilia dos conceptos distintos. El primero consiste en establecer un criterio por el cual los elementos que componen una subestructura intermedia cumplan las restricciones de precedencia que involucren a las piezas que se han utilizado en dicha secuenciación parcial. Este mismo criterio se aplica, dentro de la función de evaluación, a toda la secuencia en los estados finales del algoritmo, estableciendo simplemente si una secuencia es buena o mala.

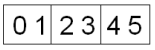
El segundo concepto está relacionado con la introducción de conocimiento humano al sistema. Se trata de comenzar el ensamblado de la estructura por su base. Por tanto, se establece un criterio que hace que la valoración de los estados que dispongan primero las piezas de la base sean mayores que los estados que no las establecen.

Los criterios anteriores son suficientes para obtener planes en tiempo reducido. No obstante, se han introducido criterios adicionales que se tratan en el siguiente apartado.

4.4 Criterios de optimización y heurísticas avanzadas

Se plantean dos líneas de optimización paralela, de manera que dependiendo de los requisitos del problema que se quiera abarcar, se pueda usar una u otra estrategia, e incluso una tercera estrategia mixta que sería una combinación de estas dos anteriores. Estas estrategias se deben implementar de forma independiente al propio algoritmo, de manera que es en la función de evaluación del algoritmo donde se hace una llamada a estas heurísticas de optimización y evalúa la secuencia propia del algoritmo de planificación.

La primera estrategia está relacionada con los sistemas multirobot y con los escenarios en los que se pretende minimizar el tiempo total de ensamblaje, aumentando el numero de ajentes necesarios para dicho propósito. Este criterio consiste en buscar una secuencia de ensamblaje tal que asignando secuencialmente cada pieza de la secuenciación a los distintos robots disponibles, el impacto de "estorbo" de un robot a otro debido a las propias restricciones de la estructura sea mínima. Este concepto se puede evaluar de la siguiente manera. Si, por ejemplo, se dispone de dos robots para ensamblar una determinada estructura de 6 piezas pueden establecerse tres ventanas con dos elementos como se muestra a continuación:



Cada ventana indica que existen n robots (2 robots en el ejemplo) moviendo n piezas (2 piezas), de manera que entre las piezas que están dentro de cada ventana no exista ninguna dependencia entre ellas. De esta manera cuanto más ventanas independientes tenga la secuencia, más paralelizable será.

La segunda estrategia está relacionada con la optimización de distancia de navegación o consumo de energía. Se plantea un escenario alternativo en el que uno o más robots robots están trabajando en el ensamblaje de una estructura, pero las piezas a ensamblar están distribuidas por el espacio o colocadas en puntos estratégicos de almacenaje. En este caso se propone un criterio de optimización de la distancia total recorrida para el ensamblaje de la estructura.

Aunque ambos criterios son aplicables en lo que sigue sólo se considera el primero de ellos de

manera que en el siguiente punto se realizará un análisis comparativo entre 2 algoritmos de secuenciación, utilizando un criterio de optimización que es el paralelismo en el ensamblado de la estructura.

5 Análisis y resultados

En este apartado se pretende hacer un estudio comparativo del algoritmo IHS frente a un algoritmo de búsqueda informada. En particular se consideran la escalabilidad y el tiempo de ejecución en obtener una solución completa. También se pretende confirmar que los algoritmos clásicos de búsqueda en árbol de estados encuentran su explosión computacional sobre los problemas de 20 piezas [1], mientras que el algoritmo IHS funciona correctamente y devuelve una solución en una cota de tiempo asumible en un problema real de ensamblado de estructuras.

Para realizar el estudio primero se considera un algoritmo guiado por heurística que sea similar al núcleo del IHS. Lo ideal sería hacer una comparativa con el algoritmo A*, que es el algoritmo de búsqueda informada más extendido, pero se tiene la peculiaridad de que este algoritmo devuelve la primera solución que encuentra (condicionado por su heurística) y el algoritmo IHS analiza todas la posibles soluciones, para devolver la mejor, por lo que la comparación con el A* no sería totalmente representativa. Un algoritmo representativo por su similitud sería la búsqueda Backtracking, pero en general este algoritmo es tipo búsqueda ciega; es decir, sin información. No obstante, se puede establecer que el método de expansión del algoritmo se base en una heurística, y hacer que la heurística sea exactamente la misma que se usa en IHS, de manera que se obtendría un algoritmo completo, con búsqueda informada, y similar a IHS.

Para este análisis se utilizarán 5 estructuras de distinto tamaño, una básica de 4 piezas figura 1, una mediana de 12 piezas (el cubo de la figura 3) y 3 estructuras más grandes de 18, 19 y 25 piezas respectivamente. La implementación de estas pruebas se han realizado en el lenguaje C ejecutándose sobre una sobre una máquina AMD Phenon x3 con un sistema operativo Linux Ubuntu 10.10.

5.1 Análisis de los resultados.

Algoritmo Backtracking +: Los resultados de secuenciación obtenidos por este algoritmo son los siguientes:

- La secuenciación obtenida para la estructura 1 (Cuadrado) es: 0, 1, 2, 3.

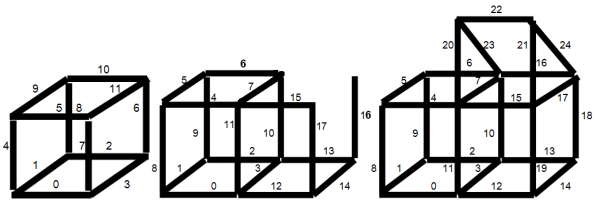


Figure 3: Estructuras para ensamblado de 12, 18, y 25 piezas respectivamente.

- La secuenciación obtenida para la estructura 2 (Cubo) es: 0, 1, 2, 3, 4, 5, 6, 7, 9, 8, 10, 11.

- La secuenciación obtenida para la estructura 3 (18 piezas): 0, 1, 2, 3, 12, 13, 14, 8, 9, 10, 5, 6, 11, 16, 4, 17, 7, 15.

- La secuenciación obtenida para la estructura 4 (19 piezas): 0, 1, 2, 3, 12, 13, 14, 8, 9, 10, 5, 6, 11, 17, 4, 7, 18, 16, 15.

- La secuenciación obtenida para la estructura 5 (25 piezas): No se encuentra.

Todas las secuencias obtenidas son correctas y cumplen las restricciones de precedencias. Analizando el nivel de paralelismo se observa que todas las secuenciaciones permiten el máximo de paralelismo posible. Prueba de ello es que en las estructuras 2, 3 y 4, si se segmenta la secuencia de dos en dos y se asignan esas tareas a dos robots, no se encontraría ningún conflicto por dependencia de piezas. Respecto a los tiempos de cómputo (tabla1) se constata que con estructuras pequeñas el tiempo de computación necesario es muy pequeño (menos de una décima de segundo en el caso de 12 piezas). Otro aspecto que se extrae de los resultados es que el requerimiento de tiempo no es proporcional al tamaño de la estructura y se puede confirmar que a partir 18 - 19 piezas, se produce la "explosión computacional" que se expone en [10], por lo cual se constata también que la escalabilidad es pequeña.

Table 1: Tiempos de computo Backtracking+.

	T total	**T media x pieza**
Estr 1	0.005 s	0.001 s
Estr 2	0.079 s	0.006 s
Estr 3	92.807 s	5.322 s
Estr 4	1126.953 s	59.313 s

Algoritmo IHS: Los resultados de secuenciación obtenidos por este algoritmo son los siguientes:

- La secuenciación obtenida para la estructura 1 (Cuadrado) es: 0, 1, 2, 3.

- La secuenciación obtenida para la estructura 2

(Cubo) es: 0, 1, 2, 3, 4, 7, 6, 8, 5, 11, 10, 9.

- La secuenciación obtenida para la estructura 3 (18 piezas): 0, 1, 2, 3, 12, 13, 14, 11, 8, 17, 4, 16, 9, 10, 5, 6, 7, 15.

- La secuenciación obtenida para la estructura 4 (19 piezas): 0, 1, 2, 3, 12, 13, 14, 11, 8, 18, 4, 17, 9, 15, 10, 5, 16, 7, 6.

- La secuenciación obtenida para la estructura 5 (25 piezas): 0, 1, 2, 3, 12, 13, 14, 11, 8, 19, 4, 18, 9, 17, 5, 15, 10, 21, 6, 16, 7, 20, 22, 23, 24.

Al igual que en el algoritmo Backtracking+ se obtienen secuencias correctas y paralelizables. Lo que cambia drásticamente es el tiempo de cómputo de estas estructuras. En la Tabla 2 se observa que e con las estructura pequeñas, el algoritmo se comporta de forma similar al caso anterior, lo que se debe a que en estos casos el impacto de la cota es muy pequeño o nulo. En el caso de las estructuras más grandes, se observa que gracias al proceso iterativo, se consigue un tiempo de cómputo muy bajo. En el caso concreto de la estructura de 25 piezas, ha encontrado solución, y sólo ha sido necesario expandir el árbol de búsqueda hasta una cota de 5 confirmándoseque el algoritmo es más escalable que el Backtracking +.

Table 2: Tiempos de cómputo alg. IHS.

	T total	**media x pieza**	**Nº iterac**
Estr 1	0.004 s	0.001 s	–
Estr 2	0.085 s	0.007 s	5 Itr
Estr 3	1.086 s	0.06 s	12 Itr
Estr 4	2.380 s	0.125 s	13 Itr
Estr 5	7.730 s	0.309 s	20 Itr

6 Conclusiones

La planificación del ensamblado de estructuras mediante uno o varios robots trabajando en paralelo es un problema relevante que requiere el desarrollo de algoritmos que incorporen heurísticas apropiadas. En este artículo se ha presentado un algoritmo de secuenciación del ensamblado al que se ha denominado Iterated Heuristic Search (IHS). Este algoritmo es el resultado de combinar la búsqueda en profundidad iterada y la búsqueda informada mediante heurística. En general se ha obtenido un algoritmo más rápido y escalable gracias a un parámetro de cota que se adapta a cada problema. Como trabajo futuro se pretende aplicar el algoritmo para resolver los problemas planteados por el montaje de estructuras en el marco del proyecto ARCAS

Se ha realizado una comparación entre el algoritmo IHS y una versión de búsqueda backtrcking con heurística comparando la escalabilidad, tiempo de cómputo y bondad de la solución de ambos algoritmos. De tal estudio se ha podido concluir que el algoritmo IHS es mucho más escalable que el comparado (que encontró su límite computacional con 19 piezas). En general, los tiempos de cómputo del algoritmo IHS son siempre menores y acotados.

Para terminar, se dejará para desarrollos futuros la implementación de una arquitectura de planificación global en dos niveles para la ejecución de las tareas de ensamblado utilizando robots manipuladores con base móvil, resolviendo en un segundo nivel de abstracción los problemas relacionados con la geometría del entorno.

Agradecimientos

Este trabajo ha sido financiado por el proyecto de excelencia de la Junta de Andalucía “Robótica ubicua en entornos urbanos” (RURBAN P09-TIC 5121), así como por el Proyecto Integrado del Séptimo Programa Marco de la Comisión Europea “Aerial Robotics Cooperative Assembly System” (ARCAS), Grant agreement 287617.

References

[1] Jimenez P. (2012). Survey on assembly sequencing: a combinatorial and geometrical perspective. Journal of Intelligent Manufacturing, DOI: 10.1007/s10845-011-0578-5

[2] Heger, F. W. (2008). Generating robust assembly plans in constrained environments. In: Proceedings of the IEEE international conference on robotics and automation, pp. 4068–4073.

[3] Nilsson, N. J. (1980). Principles of Artificial Intelligence. Palo Alto: Tioga.

[4] Martelli,A.,&Montanari, U. (1978).Optimizing decision trees through heuristically guided search. Communications of the ACM, 21(12), 1025–1039.

[5] Lai, H. Y., & Huang, C. T. (2004). A systematic approach for automatic assembly sequence plan generation. The International Journal of Advanced Manufacturing Technology, 24(9–10), 752– 763. doi:10.1007/s.

[6] de Mello, L. S. H., & Sanderson, A. C. (1991). Representations of mechanical assembly sequences. IEEE Transactions on Robotic and Automation, 7(2), 211–227. doi:10.1109/70.75904.

[7] Huang, Y., & Lee, C. (1991) A framework of knowledge-based assembly planning. In: IEEE international conference on robotics and automation, (vol. 1, pp. 599–604). doi:10.1109/ROBOT.1991. 131647.

[8] Goldwasser, M. H., & Motwani, R. (1999). Complexity measures for assembly sequences. International Journal of Computational Geometry and Application, 9(4/5), 371–417.

[9] Chakrabarty, S., & Wolter, J. (1997). A structure-oriented approach to assembly sequence planning. IEEE Transactions on Robotics and Automation, 13(1), 14–29. doi:10.1109/70.554344.

[10] Marian, R.M. (2003) Optimisation of assembly sequences using genetic algorithms. PhD thesis, University of South Australia.

[11] Motavalli, S., & Islam, A. U. (1997). Multicriteria assembly sequencing. Computer and Industrial Engineering, 32(4), 743– 751. doi:10.1016/S0360-8352(97)00014-4.

[12] Hong, D., & Cho, H. (1999). Generation of robotic assembly sequences using a simulated annealing. In: Proceedings of the 1999 IEEE/RSJ international conference on intelligent robots and systems (pp. 1247–1252).

[13] Bonneville, F., Henrioud, J., & Bourjault, A. (1995). Generation of assembly sequences with ternary operations. IEEE international symposium on assembly and aask planning (pp. 245–249). doi:10. 1109/ISATP.1995.518778.

[14] Sebaaly, M., Fujimoto, H., & Mrad, F. (1996). Linear and non-linear assembly planning: fuzzy graph representation and GA search. In: Proceedings of the 1996 IEEE international conference on robotics and automation (pp. 1533–1538).

[15] Chen, C. L. P. (1992). Design of a real-time and/or assembly scheduler on an optimization neural network. Journal of Intelligent Manufacturing, 3(4), 251–261. doi:10.1007/BF01473902.

[16] Hong, D., & Cho, H. (1995). A neural-network-based computational scheme for generating optimized robotic assembly sequences. Engineering Applications of Artificial Intelligence, 129– 145.