Wasp Swarm Generative Algorithm

Alberto Casado Garfia

Universidad Politécnica de Madrid

Abstract. Este paper está basado en el paper Modelling the Collective Building of Complex Architectures in Social Insects with Lattice Swarms[1], en el que se trata un modelo que intenta explicar cómo las avispas sociales generan sus colmenas usando bloques. Cada avispa observa los bloques que tiene su alrededor y si son parecidos a ciertas reglas que tienen memorizadas, añade un bloque que falte para seguir construyendo la colmena. En este paper se pretende realizar lo contrario, encontrar estas reglas para que las avispas construyan colmenas con formas que nosotros le indiquemos. De esta forma podemos convertirlo en un modelo generativo.

Keywords: Generative, Stigmery, Swarm

1 Introducción

Las avispas tienen la habilidad de construir colmenas de múltiples formas. Para ello deben de tener un "algoritmo" para poder seguir construyendo la colmena con solo ver su estado, usando la estigmergia[2]. En el artículo Modelling the Collective Building of Complex Architectures in Social Insects with Lattice Swarms[1] se intenta explicar este fenómeno con un algoritmo estigmergico de construcción, el cual utiliza una cuadricula tridimensional por donde se mueven las avispas aleatoriamente y ponen bloques cúbicos para construir la colmena. Estos bloques tienen un número asociado para distinguir diferentes tipos, en este trabajo solo se contempla utilizar 2 tipos. Cuando una avispa detecta que los bloques a su alrededor tienen la misma configuración que una de las reglas añade un nuevo bloque. Estas reglas se deben especificar de antemano. Reglas diferentes producirán construcciones diferentes, algunas de ellas muy parecidas a las realizadas por las avispas en la vida real.

En este paper se propone un método para generar estas reglas, de forma que las avispas creen colmenas con las formas que nosotros les enseñemos. Por ello el algoritmo consta de una fase de entrenamiento donde se generan las reglas y una fase de generación donde las avispas crean las colmenas, similar al algoritmo antes mencionado. Además, se han realizado ciertos cambios al algoritmo original para que sea más rápido a la hora de entrenar y generar y por tanto no son equivalentes.

2 Algoritmo

2.1 Entrenamiento

Para entrenar al algoritmo es necesario un conjunto de formas iniciales como conjunto de entrada. El algoritmo iterará en bucle estas formas intentando generar colmenas con la misma forma. En cada iteración las avispas irán paso a paso dejando bloques que formarán la colmena, comenzando por un bloque con el valor 1 colocado en la posición del primer bloque de la forma.

Cada avispa memoriza una única regla y tendrá un atributo vida. En cada paso las avispas visitarán uno de los últimos bloques que no ha sido visitado e intentarán encajar su regla, es decir, que los bloques que rodean al bloque visitado sean los mismos que en su regla. Si lo consiguen, adquieren vida y se añaden a una lista de avispas candidatas a añadir bloques. Si la regla encaja, pero la forma resultante no es igual a la forma con la que se entrena en la iteración, pierde vida. Finalmente, si ninguna de las avispas encaja, se crea una nueva avispa con la forma necesaria, donde los valores de los nuevos bloques serán n+1, donde n es el mayor número asociado a los bloques de la iteración.

Tras encontrar todas las avispas candidatas a añadir bloques, se selecciona una de ellas aleatoriamente con una probabilidad proporcional a su vida y se añaden todos los bloques de su regla.

Finalmente se reduce la vida de cada avispa y se continua con el siguiente paso hasta que la forma se completa y por tanto no hay nuevos bloques que visitar. Al final de cada iteración, se reduce de nuevo la vida de las avispas y se eliminan las avispas que tengan menos vida que un umbral o en caso de que haya más avispas que el máximo permitido, se eliminan las peores.

2.2 Generación

Para la generación simplemente se pone un bloque con el valor 1 y se buscan las avispas que puedan encajar, añadiéndolas a una lista de avispas candidatas. Se selecciona una de ellas de la misma forma que en el entrenamiento y se añaden los bloques especificados en su regla. Se continúa realizando este proceso con los nuevos bloques añadidos hasta que las reglas dejan de generar bloques o se supera un número máximo de bloques añadidos.

2.3 Diferencias con el algoritmo original

Tanto las avispas, que se comportan diferente, como los bloques, que pueden tener un mayor número de tipos, son diferentes al algoritmo original. Las avispas en el algoritmo original se mueven aleatoriamente en el aire, lo que, aunque es más similar con la realidad es muy lento a la hora de entrenar y generar, por lo que se optó por la estrategia de visitar los últimos bloques añadidos directamente para añadir nuevos

bloques. Además, en vez de añadir un solo bloque como en el modelo original, se añaden varios. El cambio más notorio, inspirado por los algoritmos genéticos, es el funcionamiento de las avispas, que pasan a ser individuos que codifican una regla con un grado de adaptación, representada como la vida. Este cambio es muy similar al realizado por un trabajo posterior de los mismos autores realizado al algoritmo [3]

En el modelo original se pueden especificar grupos de reglas, de forma de que, si las avispas no pueden añadir nuevos bloques usando un grupo de reglas, se comienza a usar el siguiente. En el algoritmo propuesto no se generan estos grupos porque se espera que al usar bloques con valores asociados mayores que el resto generen este mismo comportamiento, a pesar de ser más costoso computacionalmente. También las reglas pueden aplicarse con diferentes rotaciones y en este algoritmo no se utiliza.

La elección de la avispa con más vida propicia que en el entrenamiento no se creen tan frecuentemente avispas que dependan de otra avispa que posteriormente muera, provocando que el resto de las avispas generadas tras ella no puedan encajar bien y terminen muriendo. En la generación la selección de mejores avispas consigue resultados más fieles a las formas de entrada y la aleatoriedad permite generar formas diferentes en cada ejecución.

3 Resultados

A pesar de que el modelo original funciona en un espacio de tres dimensiones, para reducir la complejidad de este algoritmo en esta primera implementación se ha optado por usar únicamente 2 dimensiones, aunque el algoritmo podría ser usado en cualquier número de dimensiones. Por tanto, mientras las reglas en el modelo original tenían dimensiones 3x3x3, en estas pruebas serán de 3x3.

Para probar el algoritmo se crearon diferentes generadores de formas. La primera de ellas es una única forma la cual consigue replicar.

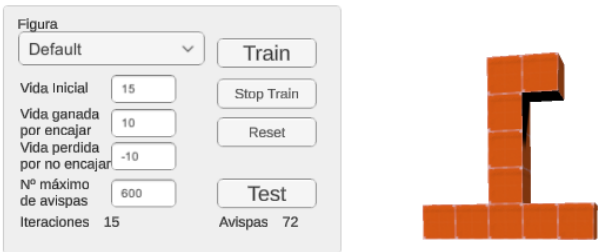

Fig. 1 Primera figura de prueba generada por el algoritmo tras su entrenamiento junto con el panel con los hiperparámetros del algoritmo.

Se puede observar en la figura 1 que el algoritmo genera 72 avispas, un número mucho mayor que los 10 bloques que componen la figura, por lo que ha creado diferentes maneras de construirla.

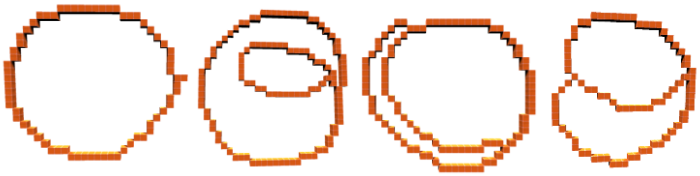

Fig 2. Diferentes figuras generadas por el algoritmo al ser entrenado con círculos con diferentes radios.

En la figura 2 podemos ver el resultado de entrenar el algoritmo con círculos. Estos resultados suelen tener 2 círculos ya que se generan desde el bloque con valor 1 inicial hacia ambos lados y terminan de generarse cuando una de las líneas choca con otra y ninguna de las reglas de las avispas le permite continuar.

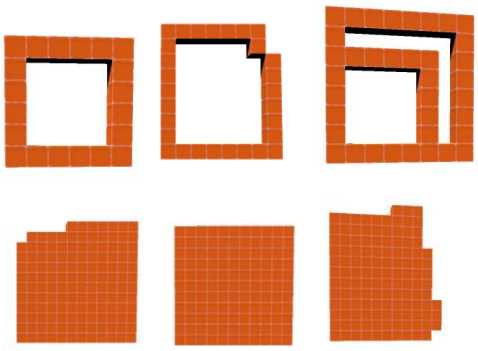

Fig 3. Diferentes figuras generadas por el algoritmo al ser entrenado con cuadrados rellenos y sin rellenar de diferentes tamaños.

Algo similar sucede al probar con cuadrados. Los cuadrados rellenos usados para entrenar el algoritmo tienen el primer bloque en la esquina inferior izquierda, por lo que la esquina superior derecha cambia de posición con el tamaño, es por ello que estos lados se ven distorsionados al generarse usando diferentes avispas que se han ajustado a diferentes tamaños.

Usando formas más complejas como la de un árbol el cual varía su altura y anchura de tronco, el tamaño de la copa del árbol y su rama, podemos observar en la figura 4 que en las formas cerradas sigue ocurriendo el mismo fenómeno.

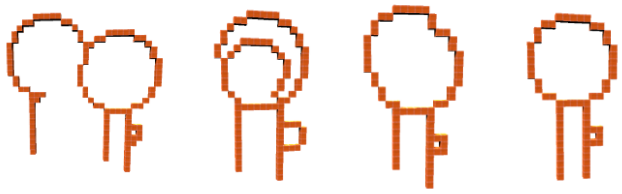

Fig 4. Árboles generados por el algoritmo
En ocasiones se generan cadenas del mismo objeto como vemos en la figura 5

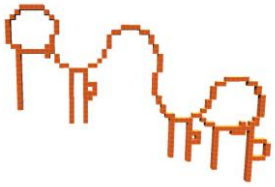

Fig 5. Cadena de árboles generada por el algoritmo

4 Conclusión y trabajo futuro

Este algoritmo permite generar nuevas formas usando otras como entrada, pero produce distorsiones con facilidad. Además, utiliza más avispas de las necesarias ya que tiende a sobreajustarse a las formas. Por ello también es muy dependiente de como se encuentra los bloques. Pero futuros cambios pueden resolver estos problemas, como en la creación de nuevas avispas evitando usar nuevos números, modificar la vida de las avispas en base a más aspectos o usar otros aspectos de la computación evolutiva como el cruce para cruzar avispas que puedan tener alguna relación. Si la forma es muy compleja se puede simplificar pixelandola y creando nuevas formas más simples. Luego se puede aplicar a cada "píxel" el mismo algoritmo para conseguir la complejidad original.

Este algoritmo es similar al propuesto por los mismos autores del trabajo original[1], en el que se intentaba generar nuevos patrones[3], a pesar de en un primer momento desconocer este trabajo, pues es un avance lógico del trabajo original.

References

1. Theraulaza, G., Bonabeau, E.: Modelling the Collective Building of Complex Architectures in Social Insects with Lattice Swarms. Journal of Theoretical Biology 381-400 (1995)
2. Grassé, P.-P. La reconstruction du nid et le coordinations inter-individuelles chez Bellicositermes natalensis et Cubitermes sp. La theorie de la stigmergie: Essai d'interpretation du comportement des termites constructeurs. Insectes Sociaux, 6, 41–81 (1959)
3. Theraulaza, G., Bonabeau, E.: Three-dimensional architectures grown by simple 'stigmergic' agents. Biosystems 13-32 (2000)