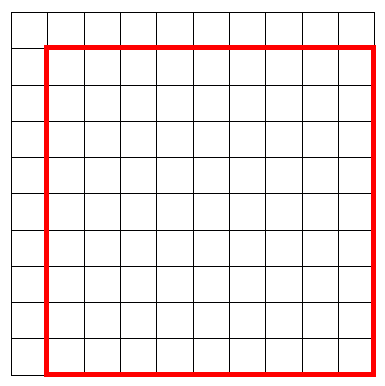
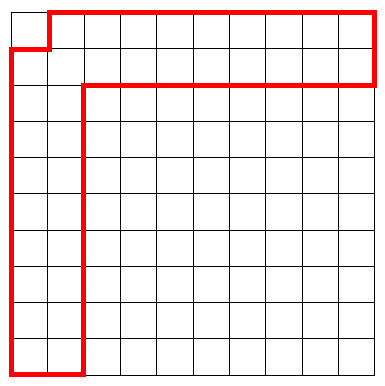
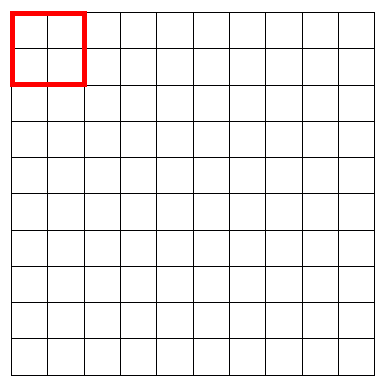
Hola, ya terminé los algoritmos e hice unas pruebas para compararlos. Le mando un reporte con la descripción de cómo funcionan y la comparación de los dos, junto con la búsqueda exhaustiva. Espero los comentarios, dudas, etc.

**Algoritmo Ventana**

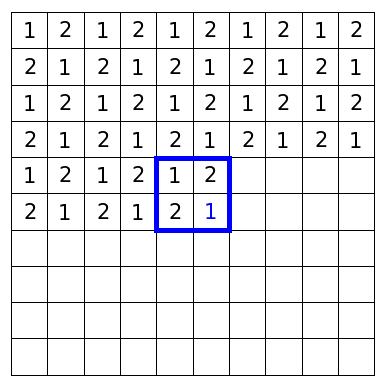
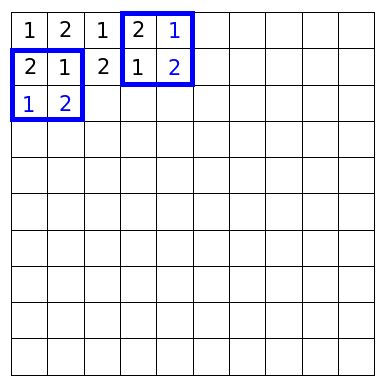
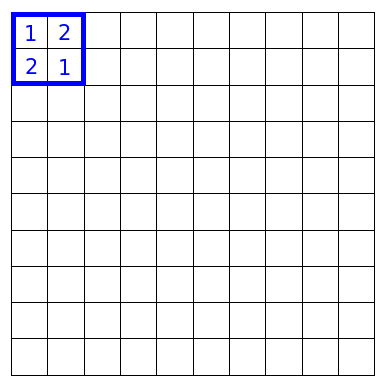
Este algoritmo funciona con una ventana de, la cual va recorriendo la malla de izquierda a derecha y de arriba a abajo una columna y un renglón a la vez. Cada vez que la ventana cambia de posición se determina cuantos espacios hay que llenar, ya sea con objetos o con espacios vacíos.

Es decir que al inicio se colocarán cuatro elementos en la malla; luego, cada vez que la ventana se desplace a lo largo del primer renglón se colocarán solo dos elementos; lo mismo sucede para la primera columna; mientras que para las posiciones restantes de la ventana, solo se colocará un elemento.

Lo que quiero decir con el párrafo anterior es que existen 3 zonas en la malla donde la ventana coloca un distinto número de elementos. Estas zonas están marcadas en rojo en las siguientes 3 imágenes:



En la primera, la ventana coloca 4 elementos, en la segunda 2 (porque ya hay 2 puestos) y en la tercera solo pone un elemento (porque ya hay 3 puestos), como se muestra en las siguientes imágenes (en azul están los elementos que pone la ventana y en negro los que ya están puestos):



La forma en que el algoritmo decide qué elementos colocar y en qué posición de la ventana, es mediante las mallas de “óptimas” (las que le mostré antes). Las cuales contienen combinaciones de objetos y vacíos que permiten tomar a los objetos con el menor número de movimientos, para mallas de . Primero se cuentan los objetos ya colocados en la ventana, se verifica cuantos objetos y vacíos quedan por colocar y dependiendo de cuántos de cada clase queden se hace la elección (esto lo explico después), de manera que al colocar los objetos que faltan en la ventana esta sea igual a una ventana óptima de .

Para elegir qué elementos colocar en la ventana, dependiendo de cuantos quedan por colocar, me basé en lo que llamé “elección por mayorías”, que consiste en elegir elementos de los que haya más, y si hay empates elegirlos al azar. Por ejemplo, si me quedan 6 cubos, 5 prismas y 4 vacíos por colocar y en la ventana actual tengo que colocar 2 elementos, entonces primero elijo un cubo, lo que me dejaría con un empate de 5 cubos y 5 prismas, por lo que ahora elijo al azar a uno de estos.

También intenté con otro método de elección que es “basado en proporciones”, que llena los elementos de la ventana, tratando de mantener la proporción de cada clase de objeto en esta, igual que la proporción de los objetos que quedan por colocar. Por ejemplo, si tengo 6 cubos, 2 prismas y 2 vacíos por colocar, entonces multiplico las proporciones (0.6, 0.2 y 0.2) por el número de espacios a llenar y redondeo; el número que resulte para cada clase de objeto es el número de elementos que se colocarán en la ventana. Este método da cosas raras, por ejemplo, en una malla de un conjunto de 33 cubos, 33 prismas y 34 vacíos los acomoda de esta manera:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 0 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 |
| 0 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 |
| 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 2 | 1 | 2 | 2 | 2 | 1 | 2 | 1 | 2 |
| 2 | 1 | 1 | 2 | 1 | 1 | 2 | 1 | 2 | 1 |
| 1 | 2 | 1 | 2 | 1 | 2 | 2 | 1 | 1 | 2 |
| 2 | 1 | 2 | 1 | 1 | 2 | 1 | 2 | 2 | 1 |

Mientras que el otro método hace un llenado más uniforme:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 0 | 1 | 0 | 2 | 2 | 1 | 0 | 1 | 0 |
| 0 | 1 | 2 | 1 | 0 | 1 | 0 | 2 | 2 | 1 |
| 2 | 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 |
| 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 | 0 |
| 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 |
| 0 | 2 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 |
| 1 | 0 | 2 | 0 | 1 | 2 | 0 | 1 | 2 | 0 |
| 2 | 1 | 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 |
| 2 | 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 |
| 0 | 1 | 2 | 0 | 1 | 2 | 0 | 1 | 2 | 0 |

Por lo que elegí mejor el “basado en mayorías”.

Aún así, el algoritmo se equivoca en algunos casos, por ejemplo cuando hay mucha diferencia entre el número de objetos de una clase y el número de vacíos. En estos casos el algoritmo tiende a crear casos imposibles (cuatro objetos de la misma clase en un bloque de ), debido a que tiene muchos objetos de una misma clase y aunque distribuya los objetos de forma uniforme muchas veces ocurren los casos imposibles. Por lo que identifiqué unos puntos clave en las mallas que “liberan” los casos imposibles:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

Si se tienen objetos de la misma clase en los espacios blancos, la mínima cantidad necesaria de espacios vacíos y/o objetos de diferente clase que se necesitan para que no haya ningún caso imposible es la cantidad de espacios verdes, en una configuración como la anterior.

Así que para no tener que hacer más reglas complicadas para evitar casos imposibles, simplemente se deja que el algoritmo acomode todos los objetos y después se revisa si hay casos imposibles; si los hay, se libera el punto clave del caso imposible al intercambiar un objeto de clase diferente o un espacio vacío por el objeto del punto clave. Y con esto concluye el algoritmo.

Entonces en resumen, lo pasos son:

1. Determinar cuantos elementos se requieren para llenar la ventana.
2. Elegir los elementos a colocar de acuerdo al criterio elegido y hacer que el acomodo del arreglo de en la malla concuerde con uno de los arreglos óptimos previamente establecidos.
3. Correr la ventana a la nueva posición y volver al paso 1 hasta que se recorra toda la malla.
4. Cuando se hayan colocado todos los elementos verificar si hay casos imposibles y corregirlos.

¿Era más o menos así como lo quería o estoy mal?

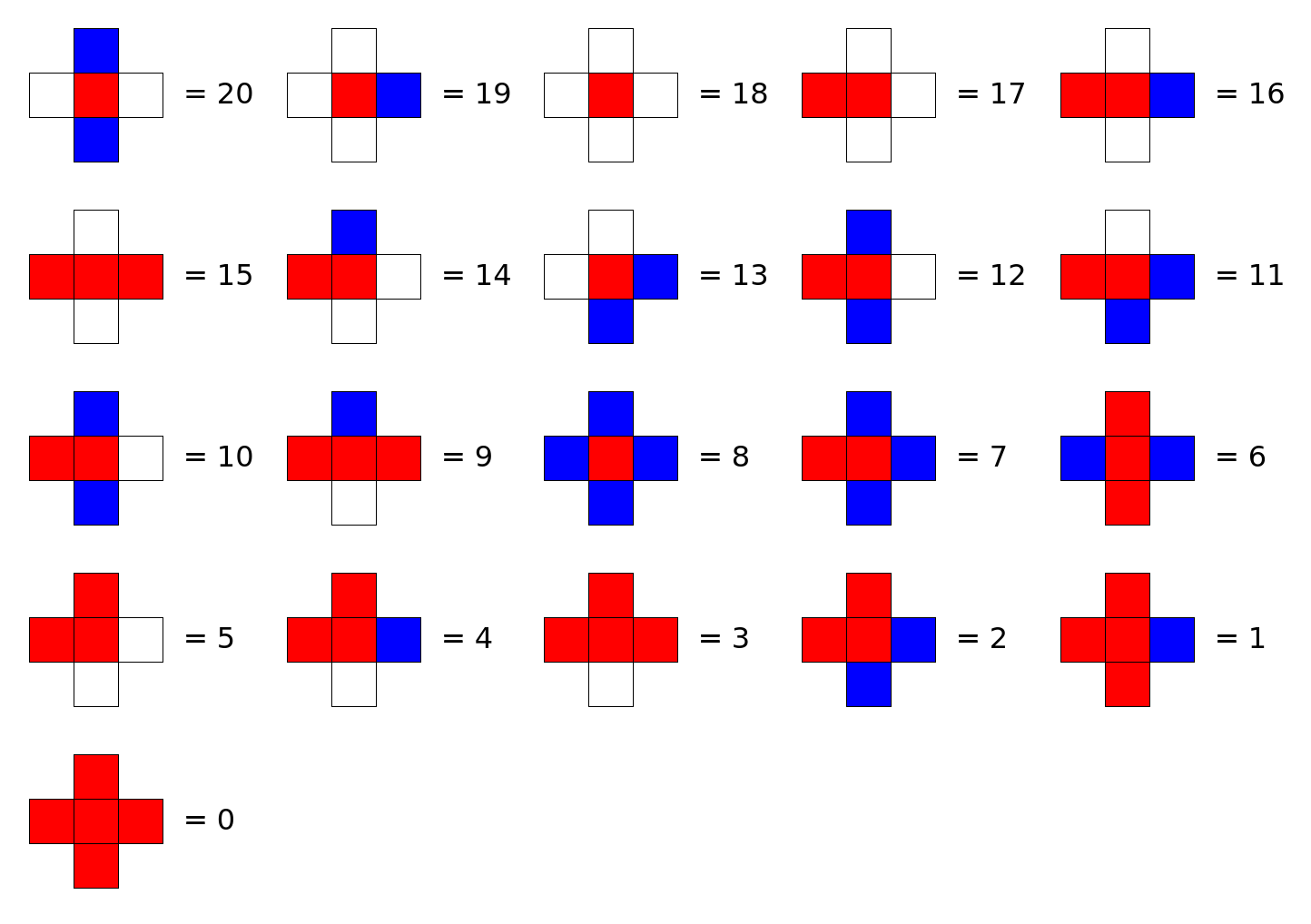
**Algoritmo Refuerzo**

Me había dicho que este algoritmo no era de aprendizaje por refuerzo, pero como no he encontrado otro nombre lo llamaré así por lo pronto.

Este algoritmo se basa en el intercambio de posiciones de pares de objetos o bien del traslado de un objeto a un lugar vacío, esto con el objetivo de ir explorando y encontrando posiciones que favorezcan la sujeción de los objetos. Está inspirado en cómo lo haría una persona (cómo acomodaría los objetos de acuerdo a nuestros intereses), es decir en el tipo de cosas en que alguien pensaría al realizar el trabajo ella misma. Por lo cual hice un experimento yo mismo con varios objetos en un espacio reducido.

Es como cuando cambias los muebles en una habitación, piensas cosas como “si muevo hacia allá el ropero, libero espacio para la mesa, pero ya no habrá espacio para la cama”, es decir, constantemente estas pensando en el costo beneficio, algo similar pasa cuando tratas de acomodar los objetos en la malla.

Entonces lo que hice fue darle una puntuación a cada espacio de la malla, en función de su vecindad y de la clase de objeto (o el vacío) que se encontrara en dicho espacio, por lo que, en nuestro caso, el número de puntuaciones posibles que se pueden asignar, es el número de combinaciones de 4 celdas (los 4 vecinos), sin tomar en cuenta las equivalentes (rotaciones y reflexiones) multiplicado por 3. Por ejemplo, las puntuaciones que les dí a las vecindades de un cubo son:



Al asignar las puntuaciones se le da prioridad a que el objeto sea sujetable. De esta forma el algoritmo trata de encontrar una configuración en la que los objetos se puedan tomar con el menor número de movimientos.

El procedimiento del algoritmo es el siguiente:

1. Acomodar los objetos en la malla de forma aleatoria, calcular la puntuación de cada celda de la malla y sumarlas para obtener la puntuación de la malla.
2. Tomar un par de elementos distintos en la malla, sumar su puntuación actual y su puntuación suponiendo que se intercambiaran. También hay que re-calcular la puntuación de sus vecinos cuando los objetos se intercambian, ya que esta se afecta con el intercambio.
3. Si la puntuación actual es menor que la del supuesto intercambio, hacer el intercambio y actualizar la puntuación de la malla. Si no, no hacer nada.
4. Elegir otro par de objetos y volver al paso 2, a menos que se cumpla la condición de paro (en mi caso elegí como condición de paro el que no mejore la puntuación de la malla en un número de intentos mayor al numero de celdas al cuadrado).

**Pruebas**

Calculé el número de pasos para tomar los objetos de las mallas de y que me arrojaron los algoritmos y los compare junto con el mejor resultado de la búsqueda exhaustiva para cada caso. Los resultados están en la siguiente tabla, en verde están los mejores resultados. Para las mallas de faltan muchos datos de búsqueda exhaustiva porque tengo esa información en otra compu y aún no la tengo a la mano jeje.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Objetos** | | | **Algoritmo Ventana** | | | | **Algoritmo Refuerzo** | | | | **Búsqueda Exhaustiva** | | |
| **N1** | **N2** | **Min. mov.** | | **Max. mov.** | **Suma mov.** | **Min. mov.** | | **Max. mov.** | **Suma mov.** | **Min. mov.** | | **Max. mov.** | **Suma mov.** |
| **Malla 3×3, 9 objetos** | | | | | | | | | | | | | |
| 8 | 1 | 1 | | 3 | 21 | 1 | | 3 | 21 | 1 | | 3 | 21 |
| 7 | 2 | 1 | | 3 | 19 | 1 | | 4 | 21 | 1 | | 2 | 18 |
| 6 | 3 | 1 | | 3 | 17 | 1 | | 3 | 16 | 1 | | 3 | 16 |
| 5 | 4 | 1 | | 7 | 23 | 1 | | 2 | 14 | 1 | | 2 | 14 |
| 4 | 5 | 1 | | 2 | 13 | 1 | | 2 | 13 | 1 | | 2 | 13 |
| 3 | 6 | 1 | | 3 | 15 | 1 | | 2 | 13 | 1 | | 2 | 13 |
| 2 | 7 | 1 | | 3 | 15 | 1 | | 2 | 13 | 1 | | 2 | 13 |
| 1 | 8 | 1 | | 3 | 15 | 1 | | 3 | 15 | 1 | | 3 | 15 |
| **Malla 3×3, 8 objetos** | | | | | | | | | | | | | |
| 8 | 0 | 1 | | 2 | 12 | 1 | | 2 | 12 | 1 | | 2 | 12 |
| 7 | 1 | 1 | | 3 | 18 | 1 | | 2 | 11 | 1 | | 2 | 11 |
| 6 | 2 | 1 | | 3 | 16 | 1 | | 3 | 13 | 1 | | 2 | 12 |
| 5 | 3 | 1 | | 6 | 19 | 1 | | 2 | 9 | 1 | | 2 | 9 |
| 4 | 4 | 1 | | 2 | 12 | 1 | | 1 | 8 | 1 | | 1 | 8 |
| 3 | 5 | 1 | | 3 | 13 | 1 | | 1 | 8 | 1 | | 1 | 8 |
| 2 | 6 | 1 | | 3 | 13 | 1 | | 1 | 8 | 1 | | 1 | 8 |
| 1 | 7 | 1 | | 3 | 13 | 1 | | 2 | 10 | 1 | | 2 | 10 |
| 0 | 8 | 1 | | 2 | 12 | 1 | | 2 | 12 | 1 | | 2 | 12 |
| **Malla 3×3, 7 objetos** | | | | | | | | | | | | | |
| 7 | 0 | 1 | | 2 | 10 | 1 | | 2 | 9 | 1 | | 2 | 9 |
| 6 | 1 | 1 | | 2 | 9 | 1 | | 2 | 8 | 1 | | 2 | 8 |
| 5 | 2 | 1 | | 3 | 12 | 1 | | 1 | 7 | 1 | | 1 | 7 |
| 4 | 3 | 1 | | 4 | 15 | 1 | | 1 | 7 | 1 | | 1 | 7 |
| 3 | 4 | 1 | | 2 | 10 | 1 | | 1 | 7 | 1 | | 1 | 7 |
| 2 | 5 | 1 | | 2 | 10 | 1 | | 1 | 7 | 1 | | 1 | 7 |
| 1 | 6 | 1 | | 2 | 9 | 1 | | 1 | 7 | 1 | | 1 | 7 |
| 0 | 7 | 1 | | 2 | 10 | 1 | | 2 | 9 | 1 | | 2 | 9 |
| **Malla 3×3, 6 objetos** | | | | | | | | | | | | | |
| 6 | 0 | 1 | | 2 | 8 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| 5 | 1 | 1 | | 2 | 7 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| 4 | 2 | 1 | | 2 | 7 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| 3 | 3 | 1 | | 1 | 6 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| 2 | 4 | 1 | | 1 | 6 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| 1 | 5 | 1 | | 2 | 7 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| 0 | 6 | 1 | | 2 | 8 | 1 | | 1 | 6 | 1 | | 1 | 6 |
| **Malla 4×4, 16 objetos** | | | | | | | | | | | | | |
| 12 | 4 | 1 | | 7 | 36 | 1 | | 5 | 38 | 1 | | 4 | 36 |
| 11 | 5 | 1 | | 5 | 34 | 1 | | 4 | 34 |  | |  |  |
| 10 | 6 | 1 | | 5 | 31 | 1 | | 4 | 32 |  | |  |  |
| 9 | 7 | 1 | | 5 | 32 | 1 | | 5 | 32 |  | |  |  |
| 8 | 8 | 1 | | 5 | 30 | 1 | | 3 | 28 | 1 | | 3 | 26 |
| 7 | 9 | 1 | | 4 | 28 | 1 | | 3 | 26 |  | |  |  |
| 6 | 10 | 1 | | 7 | 42 | 1 | | 3 | 26 |  | |  |  |
| 5 | 11 | 1 | | 4 | 29 | 1 | | 3 | 26 |  | |  |  |
| 4 | 12 | 1 | | 3 | 28 | 1 | | 3 | 26 |  | |  |  |
| **Malla 4×4, 15 objetos** | | | | | | | | | | | | | |
| 12 | 3 | 1 | | 6 | 32 | 1 | | 3 | 28 |  | |  |  |
| 11 | 4 | 1 | | 5 | 36 | 1 | | 4 | 31 |  | |  |  |
| 10 | 5 | 1 | | 6 | 38 | 1 | | 4 | 29 |  | |  |  |
| 9 | 6 | 1 | | 4 | 30 | 1 | | 4 | 25 |  | |  |  |
| 8 | 7 | 1 | | 7 | 33 | 1 | | 2 | 21 |  | |  |  |
| 7 | 8 | 1 | | 3 | 25 | 1 | | 2 | 21 |  | |  |  |
| 6 | 9 | 1 | | 3 | 25 | 1 | | 2 | 21 |  | |  |  |
| 5 | 10 | 1 | | 4 | 27 | 1 | | 2 | 21 |  | |  |  |
| 4 | 11 | 1 | | 3 | 25 | 1 | | 2 | 21 |  | |  |  |
| 3 | 12 | 1 | | 3 | 24 | 1 | | 2 | 22 |  | |  |  |
| **Malla 4×4, 14 objetos** | | | | | | | | | | | | | |
| 12 | 2 | 1 | | 5 | 32 | 1 | | 3 | 24 |  | |  |  |
| 11 | 3 | 1 | | 5 | 32 | 1 | | 3 | 22 |  | |  |  |
| 10 | 4 | 1 | | 5 | 33 | 1 | | 4 | 24 |  | |  |  |
| 9 | 5 | 1 | | 4 | 28 | 1 | | 3 | 20 |  | |  |  |
| 8 | 6 | 1 | | 4 | 25 | 1 | | 2 | 18 |  | |  |  |
| 7 | 7 | 1 | | 4 | 23 | 1 | | 2 | 18 |  | |  |  |
| 6 | 8 | 1 | | 3 | 23 | 1 | | 2 | 18 |  | |  |  |
| 5 | 9 | 1 | | 6 | 34 | 1 | | 2 | 18 |  | |  |  |
| 4 | 10 | 1 | | 3 | 22 | 1 | | 2 | 18 |  | |  |  |
| 3 | 11 | 1 | | 3 | 22 | 1 | | 2 | 19 |  | |  |  |
| 2 | 12 | 1 | | 3 | 22 | 1 | | 3 | 22 |  | |  |  |
| **Malla 4×4, 13 objetos** | | | | | | | | | | | | | |
| 12 | 1 | 1 | | 4 | 23 | 1 | | 3 | 20 |  | |  |  |
| 11 | 2 | 1 | | 4 | 26 | 1 | | 3 | 19 |  | |  |  |
| 10 | 3 | 1 | | 4 | 23 | 1 | | 3 | 19 |  | |  |  |
| 9 | 4 | 1 | | 4 | 24 | 1 | | 3 | 17 |  | |  |  |
| 8 | 5 | 1 | | 4 | 25 | 1 | | 2 | 15 |  | |  |  |
| 7 | 6 | 1 | | 5 | 27 | 1 | | 2 | 15 |  | |  |  |
| 6 | 7 | 1 | | 3 | 20 | 1 | | 2 | 15 |  | |  |  |
| 5 | 8 | 1 | | 2 | 18 | 1 | | 2 | 15 |  | |  |  |
| 4 | 9 | 1 | | 3 | 22 | 1 | | 2 | 15 |  | |  |  |
| 3 | 10 | 1 | | 3 | 20 | 1 | | 2 | 16 |  | |  |  |
| 2 | 11 | 1 | | 3 | 20 | 1 | | 2 | 17 |  | |  |  |
| 1 | 12 | 1 | | 3 | 19 | 1 | | 3 | 19 |  | |  |  |
| **Malla 4×4, 12 objetos** | | | | | | | | | | | | | |
| 12 | 0 | 1 | | 3 | 17 | 1 | | 2 | 16 |  | |  |  |
| 11 | 1 | 1 | | 3 | 20 | 1 | | 2 | 15 |  | |  |  |
| 10 | 2 | 1 | | 3 | 20 | 1 | | 2 | 14 |  | |  |  |
| 9 | 3 | 1 | | 3 | 17 | 1 | | 2 | 13 |  | |  |  |
| 8 | 4 | 1 | | 3 | 21 | 1 | | 2 | 14 |  | |  |  |
| 7 | 5 | 1 | | 5 | 25 | 1 | | 1 | 12 | 1 | | 1 | 12 |
| 6 | 6 | 1 | | 3 | 17 | 1 | | 1 | 12 | 1 | | 1 | 12 |
| 5 | 7 | 1 | | 2 | 17 | 1 | | 1 | 12 | 1 | | 1 | 12 |
| 4 | 8 | 1 | | 2 | 16 | 1 | | 2 | 14 |  | |  |  |
| 3 | 9 | 1 | | 3 | 18 | 1 | | 2 | 13 |  | |  |  |
| 2 | 10 | 1 | | 3 | 18 | 1 | | 2 | 14 |  | |  |  |
| 1 | 11 | 1 | | 3 | 18 | 1 | | 2 | 15 |  | |  |  |
| 0 | 12 | 1 | | 3 | 17 | 1 | | 2 | 17 |  | |  |  |
| **Malla 4×4, 11 objetos** | | | | | | | | | | | | | |
| 11 | 0 | 1 | | 2 | 15 | 1 | | 2 | 14 |  | |  |  |
| 10 | 1 | 1 | | 3 | 17 | 1 | | 2 | 13 |  | |  |  |
| 9 | 2 | 1 | | 3 | 17 | 1 | | 3 | 15 |  | |  |  |
| 8 | 3 | 1 | | 3 | 18 | 1 | | 2 | 12 |  | |  |  |
| 7 | 4 | 1 | | 3 | 14 | 1 | | 1 | 11 | 1 | | 1 | 11 |
| 6 | 5 | 1 | | 3 | 13 | 1 | | 1 | 11 | 1 | | 1 | 11 |
| 5 | 6 | 1 | | 3 | 13 | 1 | | 1 | 11 | 1 | | 1 | 11 |
| 4 | 7 | 1 | | 3 | 14 | 1 | | 1 | 11 | 1 | | 1 | 11 |
| 3 | 8 | 1 | | 2 | 15 | 1 | | 1 | 11 | 1 | | 1 | 11 |
| 2 | 9 | 1 | | 3 | 17 | 1 | | 2 | 12 |  | |  |  |
| 1 | 10 | 1 | | 2 | 15 | 1 | | 2 | 13 |  | |  |  |
| 0 | 11 | 1 | | 2 | 15 | 1 | | 2 | 15 |  | |  |  |
| **Malla 4×4, 10 objetos** | | | | | | | | | | | | | |
| 10 | 0 | 1 | | 2 | 14 | 1 | | 2 | 12 |  | |  |  |
| 9 | 1 | 1 | | 3 | 15 | 1 | | 2 | 12 |  | |  |  |
| 8 | 2 | 1 | | 3 | 13 | 1 | | 2 | 11 |  | |  |  |
| 7 | 3 | 1 | | 3 | 13 | 1 | | 2 | 11 |  | |  |  |
| 6 | 4 | 1 | | 2 | 12 | 1 | | 1 | 10 | 1 | | 1 | 10 |
| 5 | 5 | 1 | | 2 | 12 | 1 | | 1 | 10 | 1 | | 1 | 10 |
| 4 | 6 | 1 | | 2 | 12 | 1 | | 1 | 10 | 1 | | 1 | 10 |
| 3 | 7 | 1 | | 3 | 13 | 1 | | 1 | 10 | 1 | | 1 | 10 |
| 2 | 8 | 1 | | 3 | 13 | 1 | | 1 | 10 | 1 | | 1 | 10 |
| 1 | 9 | 1 | | 3 | 15 | 1 | | 2 | 11 |  | |  |  |
| 0 | 10 | 1 | | 2 | 14 | 1 | | 2 | 12 |  | |  |  |
| **Malla 4×4, 9 objetos** | | | | | | | | | | | | | |
| 9 | 0 | 1 | | 2 | 11 | 1 | | 2 | 10 | 1 | | 2 | 10 |
| 8 | 1 | 1 | | 2 | 10 | 1 | | 2 | 10 |  | |  |  |
| 7 | 2 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 6 | 3 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 5 | 4 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 4 | 5 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 3 | 6 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 2 | 7 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 1 | 8 | 1 | | 2 | 10 | 1 | | 1 | 9 | 1 | | 1 | 9 |
| 0 | 9 | 1 | | 2 | 11 | 1 | | 2 | 10 |  | |  |  |

La complejidad del algoritmo ventana depende del número de celdas |E|, y la del algoritmo de aprendizaje por refuerzo depende del número de objetos de cada clase (N1 y N2), del número de vacíos |Ø| y del número de veces que se analizan todos los pares diferentes posibles de la malla X.

Complejidad algoritmo ventana: O(|E|).

Complejidad algoritmo refuerzo: O(X(N1(N2+|Ø|) + N2(N1+|Ø|) + |Ø|(N1+N2))) ≤ O(X|E|2)