TIPOS DE ALGORITMOS PARA LA BUSQUEDA Y RECOGIDA DE BASURA (PROBLEMA DE LA ASPIRADORA):

Fernando Candelario Herrero

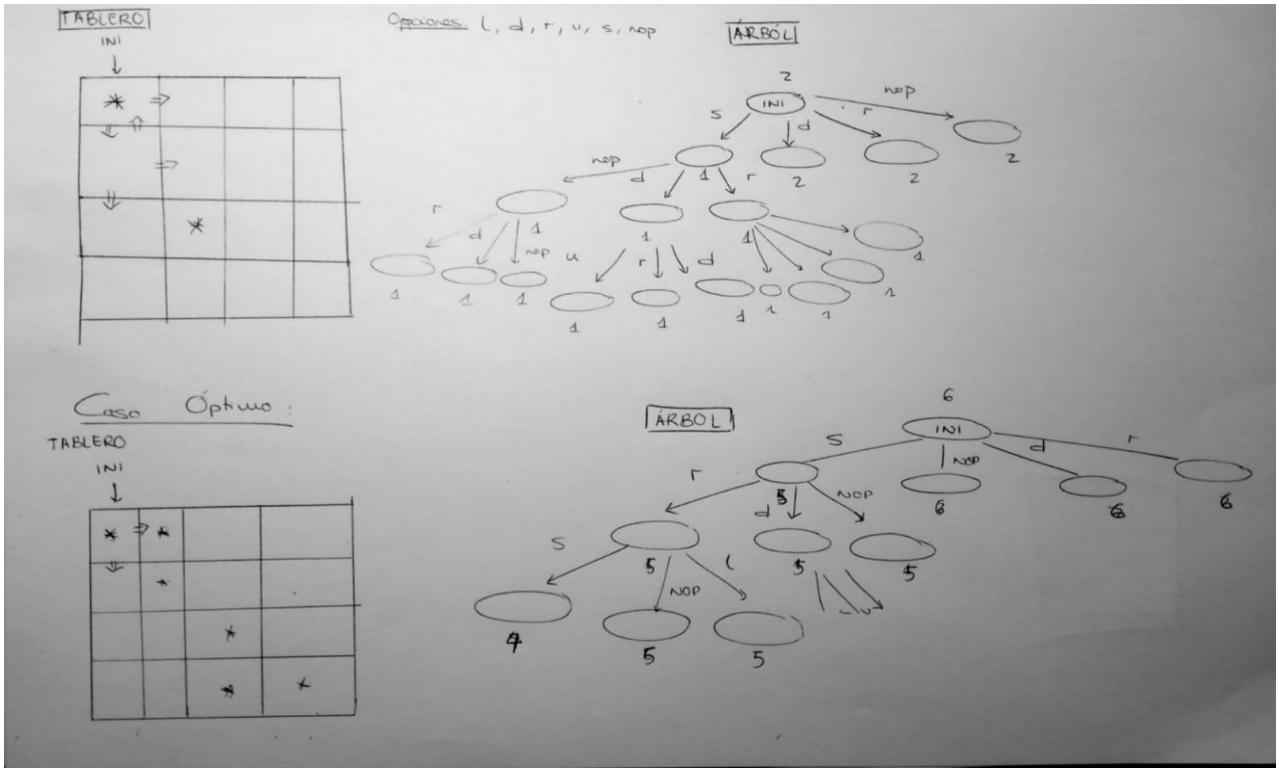
Gonzalo Sanz Rodríguez

Algoritmo AStarSearchHGlobalDirt:

El algoritmo de búsqueda de **AStarSearchHGlobalDirt** usa una heurística basada en el número de basuras que se encuentran en el tablero, algo inusual en los algoritmos de búsqueda ya que suelen tener en cuenta más atributos del tablero (anchura, longitud, distancia entre basuras, etc).

El principal inconveniente de esta heurística es si el número de basuras es ínfimo y la distancia entre las basuras en considerable el número de estados en cada nodo aumenta exponencialmente, ya que en todos los estados disponibles la prioridad es la misma y a medida que avanzamos en cada una de las ramas provocando en numerosas ocasiones que tenga que expandirse en cada uno de los estados posibles, hasta llegar a un nodo no destino y de esta manera a una indeterminación.

Así mismo, sabiendo que el algoritmo no es óptimo para un numero de basuras pequeño, es significativamente eficaz para un numero de basuras mayor (y a poder ser contiguas), ya que el algoritmo sabe muy fácilmente cual es la rama destino y de esa manera no se pierde expandiéndose en varios estados.



Algoritmo de A* pero usando heurística de basura global y distancia (AStarGDirAndDistance):

Para explicar el funcionamiento de la **heurística basuraGlobal + Distancia** usaremos un tablero ejemplo:

Dada una localización, se calcula la distancia total con el resto de las basuras:

"distance=distance+

(x-vs.getLocation().getX())*(x-vs.getLocation().getX())+

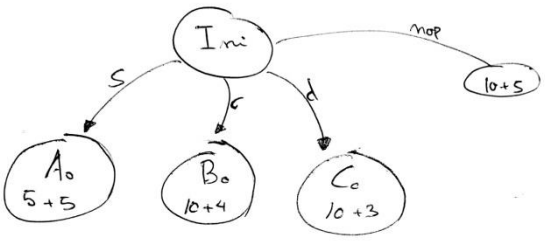
(y-vs.getLocation().getY())*(y-vs.getLocation().getY());"

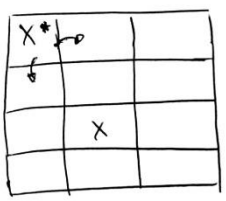
A esa distancia se le suma el número de basuras totales multiplicado por 5 para dar así más valor a está heurística, así obtenemos el coste de cada nodo.

Acciones: l(left),r(right),s(suck),d(down),u(up),nop

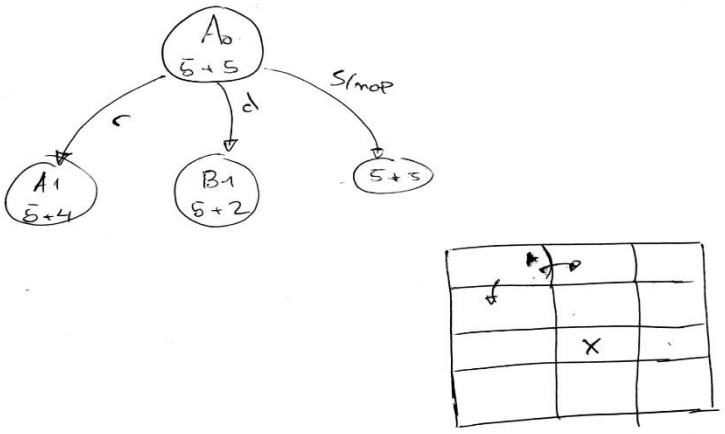
El tablero Inicial sería,pos inicial (0,0), las “X” indican las basuras y “*” indica donde está situado la aspiradora

Cada vez que realizamos una expansión, recalculamos el "coste" de cada posible nodo y elegimos el nodo que tenga menor "coste".

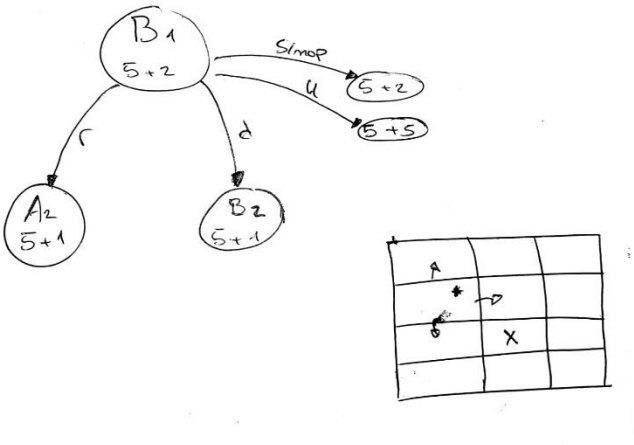


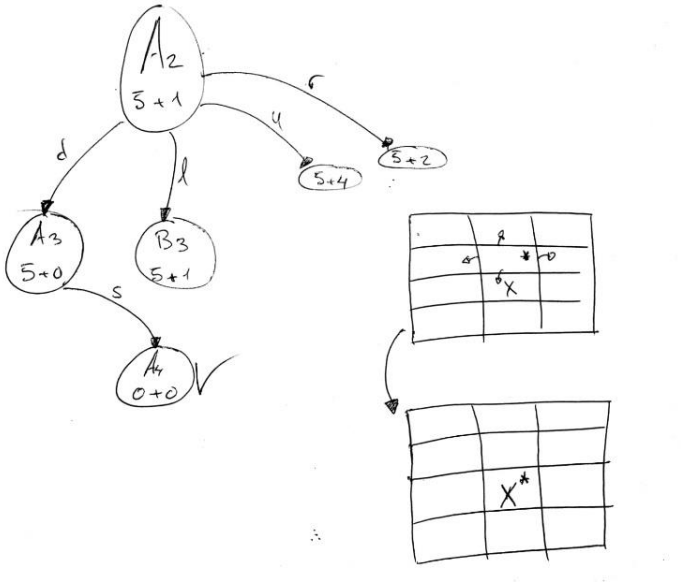


Así por cada expansión elegimos la rama que tenga el nodo más prometedor para llegar a un nodo objetivo (coste 0)



Una vez terminada la búsqueda del nodo objetivo, devolvemos la lista de acciones (ramas del árbol) que hemos elegido.





Esta heurística es útil para escenarios que no contengan un gran número de basuras ya que, en caso contrario, la heurística no es capaz de determinar un camino en el árbol ya que hay una gran cantidad de nodos con valores similares y para continuar con la búsqueda tiene que seguir expandiendo.

Conclusión:

Tras haber realizado estos ejemplos, nos hemos dado cuenta que ninguna de las dos heurísticas es mejor o peor que la otra, cada una es buena en los suyo, en su escenario ideal. La de basura global es buena para escenarios con muchas basuras y la de basura global+distancia es buena para escenarios con un número de basuras no tan elevado. En un primer momento opinábamos que la heurística de basura global era mejor que la de global+distancia por que suponíamos que el escenario más común al que se iba a enfrentar el "Aspirador" era un escenario con muchas basuras, pero esto está claro que no tiene por qué ser así.