COMP9414 Assignment 2

Q1：

a)

	start10	start12	start20	start30	start40
UCS	2565	Mem	Mem	Mem	Mem
IDS	2407	13812	5297410	Time	Time
A*	33	26	915	Mem	Mem
IDA*	29	21	952	17297	112571

Table 1 Test Table

b)

Firstly, the search strategies of UCS and IDS is Uninformed Search Strategies, which use only the information available in the problem definition, and can only distinguish a goal from a non-goal state. However, because of the A* is a method of the Informed Search Strategies and IDA is the updated way of A* and iterative deepening search, so A* and IDA* are all informed search strategies, which use task-specific knowledge and are more efficient than Uniformed search.

Secondly, when it comes to solving specific problem, in terms of time and memory usage, we have learned at class that the Time and Space complexity of the 4 different searches as follows:

	Time complexity	Space complexity
UCS	$O(b^{\lceil C*/\varepsilon \rceil})$	$O(b^{\lceil C*/\varepsilon \rceil})$
IDS	O(b^d)	O(bd)
A*	Exponential in [relative error in h× length of solution]	Keeps all nodes is memory
IDA*	The IDA* algorithm is applied on the basis of the A* algorithm by update IDS, so its space complexity is reduced to linear at all points (exponential growth) of the A* record, that is, only the current effective path is recorded. The time complexity does not change and it is still an exponential increase but because IDS can take advantage of both BFS and DFS which can filter out effective point sets in advance to reduce calculation time, so IDA* 's time complexity should be less than A*.	

Table 2 The Comparison of Time and Space

At last, on the one hand, we can get an apparent solution from the test table (Table 1) above, that uninformed searches account for the largest spaces and UCS's space is larger than IDS, because of IDS only remember all the effective nodes, so the only problem for the strategy is too long solving time. On the other hand, although the informed searches take over the space problems, but they still meet the difficulties of time. A* search occupies so much space to look for the optimal path that it is out of global stack, but IDA* work out to solve the problem by using more intelligent path selection, so the IDA* is the only method of these 4 methods which can both take the least time and spaces.

Q2:

a) & c):

	Start 50	Start60	Start60

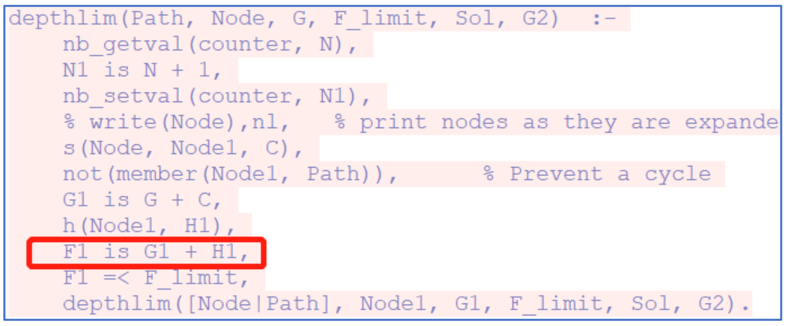
IDA*	50	1462512	60	321252368	64	1209086782
1.2	52	191438	62	230861	66	431033
1.4	66	116174	82	3673	94	188917
1.6	100	34647	148	55626	162	235852
Greedy	164	5447	166	1617	184	2174

b)

The heuristic path algorithm is a best-first search in which the objective function is:

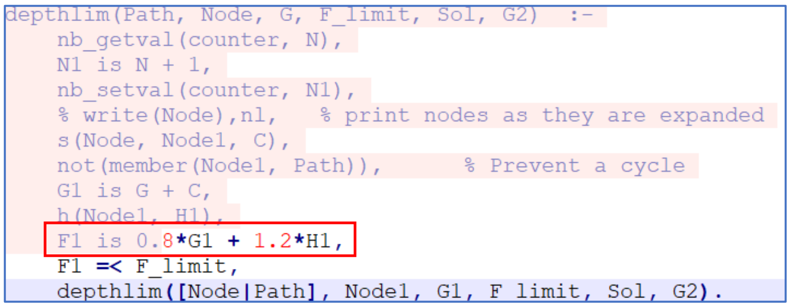
F(n) = (2-w) g(n)+w h(n), where 0<W<2

So I change these code:



```
depthlim(Path, Node, G, F_limit, Sol, G2)  :-
    nb_getval(counter, N),
    N1 is N + 1,
    nb_setval(counter, N1),
    % write(Node),nl,   % print nodes as they are expande
    s(Node, Node1, C),
    not(member(Node1, Path)),       % Prevent a cycle
    G1 is G + C,
    h(Node1, H1),
    F1 is G1 + H1,
    F1 =< F_limit,
    depthlim([Node|Path], Node1, G1, F_limit, Sol, G2).
```

To the code below:



```
depthlim(Path, Node, G, F_limit, Sol, G2)  :-
    nb_getval(counter, N),
    N1 is N + 1,
    nb_setval(counter, N1),
    % write(Node),nl,   % print nodes as they are expanded
    s(Node, Node1, C),
    not(member(Node1, Path)),       % Prevent a cycle
    G1 is G + C,
    h(Node1, H1),
    F1 is 0.8*G1 + 1.2*H1,
    F1 =< F_limit,
    depthlim([Node|Path], Node1, G1, F_limit, Sol, G2).
```

d):

Greedy search function is $f_g(\mathrm{n}) = \mathrm{h(n)}$

Heuristics IDA* search function is $f_{ida*}(\mathrm{n}) = (2 - \mathrm{w})\mathrm{g(n)} + \mathrm{wh(n)}$ when $0 < \mathrm{w} < 2$.

From the functions above, it is apparent that Greedy is just twice the special case of Heuristics IDA* search when w=2, $f_{ida*}(\mathrm{n}) = (2 - 2)\mathrm{g(n)} + 2\mathrm{h(n)} \Rightarrow \mathrm{f(n)} = 2\mathrm{h(n)} \equiv 2f_g(\mathrm{n}) \equiv f_g(\mathrm{n})$ The search efficiency isn't influenced by the parameter 2.

And when w=1, it is the traditional A* search and when w=0, it comes to the basic UCS strategy. So as the w increasing from 1 to 2, the length of the path is returned as G is increasing, and the total number of states expanded during the search is returned as N is decreasing. Although there may be some special cases, the general trend is like what it is said before.

Q3:

a):

Because the agent only can move can't move diagonally in the maze, I think the Manhattan distance may be the other way to define the formula:

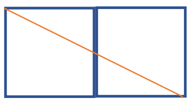
$$\mathrm{h}(\mathrm{x}, \mathrm{y}, x_G, y_G) = |\mathrm{x} - x_G| + |\mathrm{y} - y_G|$$

b):

i): No, Straight-Line-Distance heuristic cannot be admissible, because in the h_{SLD} the least diagonal move can be recognized as $\sqrt{2}$, but it should be 1(which is less than $\sqrt{2}$). So it is not admissible.

ii): No, my formula (Manhattan Distance) cannot be admissible either, because in the formula $h(x, y, x_G, y_G)$ the least diagonal move can be recognized as 2, but it should be 1. So it is not admissible.

iii):



As the graph shown, x=2, y=1, h(…) = 2, h(…) should be the larger side of the path.

So in my opinion, the formula should be:

$$h(x, y, x_G, y_G) = \max[|x_G - x|, |y_G - y|]$$

Q4:

a):

n	Optimal sequence	n	Optimal sequence
1	[+ -]	12	[+ + + 0 - - -]
2	[+ 0 -]	13	[+ + + 0 - - 0 -]
3	[+ 0 0 -]	14	[+ + + 0 - 0 - -]
4	[+ + - -]	15	[+ + + 0 0 - - -]
5	[+ + - 0 -]	16	[+ + + + - - - -]
6	[+ + 0 - -]	17	[+ + + + - - - 0 -]
7	[+ + 0 - 0 -]	18	[+ + + + - - 0 - -]
8	[+ + 0 0 - -]	19	[+ + + + - 0 - - -]
9	[+ + + - - -]	20	[+ + + + 0 - - - -]
10	[+ + + - - 0 -]	21	[+ + + + 0 - - - 0 -]
11	[+ + + - 0 - -]		

b):

Condition 1:

when the sequence is [+++… - - - …] without 0

when M(n,0)=2k, $n = \frac{k(k+1)}{2} + \frac{(k-1)k}{2} = k^2$, so under this condition, n should be an exponential of a integer. $M(n, 0) = 2k = 2\sqrt{n}$ Because $k \in \mathbf{Z}$, so $\lceil k \rceil = k$.

It is worthy to mention that this condition is the farthest node to reach, for there's no 0s in the sequence.

Condition 2:

When there are some 0 in the sequence, so the 0s length scope should be less than 2k(two 0s), k(one 0s).

Firstly, we need to prove that the largest number of 0s is 2.

Generally, we have known that if there's no 0s ,the distance should be k^2. So if we add one 0 into the sequence the longest distance should be like this:

$$n1 = k^2 + k \text{ at this time , } M = 2k + 1$$

and we try to add more 0s, if there are two 0s:

$$n2 = k^2 + 2k \text{ at this time , } M = 2k + 2$$

Because n、k ∈ Z， n2 +1 is just $(k+1)^2$ which means the longest distance should be no 0s.

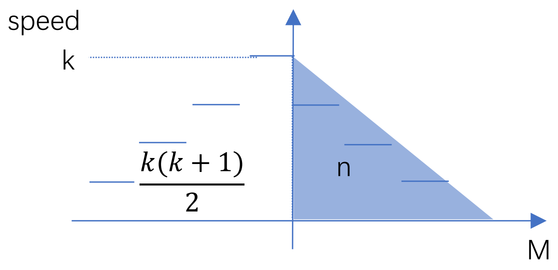
So there shouldn't be more than two 0s in the sequence.
Then following the previous conclusion, we can divide all the positive integers into :

$$k^2 < n \le k(k+1)\ ,\ k(k+1) < n \le (k+1)^2.$$

And we can know that the former part means to add one 0, and the latter part means to add two 0s.
From above deduction, we can apparently know that it is equivalent to the functions below:

$$\lceil 2\sqrt{n}\rceil = \begin{cases} 2s+1, & s^2 < n \le s(s+1) \\ 2s+2, & s(s+1) < n \le (s+1)^2 \end{cases}$$

c):
When we calculate M(n,k), we can regard the M(n,k) as a process during the process from (0,0) to (k,0). And we can assume that the former conditions before M(n,k) as below:



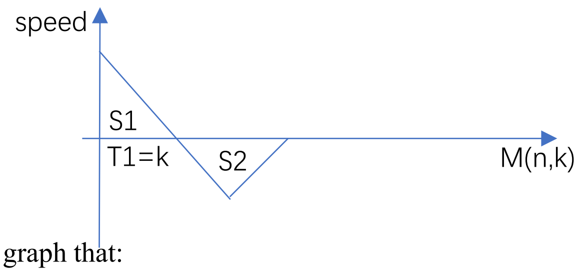
From the deduction from question b and $n \ge \frac{k(k-1)}{2}$, we can know that there's no negative part in the process, and the least M(n,0)=2k. so the distance before the M(n,k) should be the sum of 1+2+3+…+k = $\frac{k(k+1)}{2}$. And the n means the shade square in the graph above. The whole distance is the whole square of all times which means:

L = $\frac{k(k+1)}{2} + n$, from the b's conclusion we can know that $\mathrm{M}(\mathrm{L},0) = \left\lceil 2\sqrt{\frac{k(k+1)}{2} + n}\right\rceil$, so M(n,k) should minus the passed speeding up process (t=k), so:

$$\mathrm{M}(\mathrm{n},\mathrm{k}) = \left\lceil 2\sqrt{\frac{k(k+1)}{2} + n}\right\rceil - k$$

d):

Because without negative part in process, the n should more than $\frac{k(k-1)}{2}$, so when as the question's condition, there should be some negative part, as the graph below.



We can learn from the graph that:

$$\mathrm{n} = \mathrm{S1} - \mathrm{S2}$$

$$\mathrm{S1} = \frac{k(k-1)}{2} \text{ and S1part time :T1=k}$$

$$\mathrm{S2} = \frac{k(k-1)}{2} - n$$

From the 2nd question's conclusion, T2=$\left\lceil 2\sqrt{\frac{k(k-1)}{2} - n} \right\rceil$,so the whole time should be

T1+T2=$\left\lceil 2\sqrt{\frac{k(k-1)}{2} - n} \right\rceil + k.$

So the conclusion formula should be like this:

$$\mathrm{M(n,k)} = \left\lceil 2\sqrt{\frac{k(k-1)}{2} - n} \right\rceil + k$$

e):

Because it is the 2-dimensional move, so the function should meet all the needs of both dimensions, so the time should be the maximum value between them.

$$\mathrm{h(r,c,u,v,}r_G, c_G) = \max(\mathrm{M}(r - r_G, u), \mathrm{M}(c - c_G, v))$$