Q1.

2. Modify the algorithm for *Depth First Search* (Figure 2.18) to incorporate the heuristic function and do a *Best First Search.*

```
DepthFirstSearch()
  1 open ← ((start NIL))
  2 closed ← ()
  3 while not Null(open)
  4   do nodePair ← Head(open)
  5     node ← Head(nodePair)
  6     if GoalTest(node) = TRUE
  7       then return ReconstructPath(nodePair, closed)
  8       else closed ← Cons(nodePair, closed)
  9            children ← MoveGen(node)
 10            noLoops ← RemoveSeen(children, open, closed)
 11            new ← MakePairs(noLoops, node)
 12            open ← Append(new, Tail(open))
 13 return "No solution found"

RemoveSeen(nodeList, openList, closedList)
  1 if Null(nodeList)
  2   then return ()
  3     else n ← Head(nodeList)
  4       if (OccursIn(n, openList) OR OccursIn(n, closedList))
  5         then return RemoveSeen(Tail(nodeList), openList, closedList)
  6         else return Cons(n,RemoveSeen(Tail(nodeList), openList, closedList)

OccursIn(node, listOfPairs)
  1 if Null(listOfPairs)
  2   then return FALSE
  3   else if n = Head(Head(listOfPairs))
  4          then return TRUE
  5          else return OccursIn(node, Tail(listOfPairs))

MakePairs(list, parent)
  1 if Null(list)
  2   then return ()
  3   else return Cons(MakeList(Head(list), parent),
                       MakePairs(Tail(list), parent))
```

h (list of pairs)

h (pairs):

when · hmax For pair in pairs

return Pair hmax

Q2.

2. Describe the *Simulated Annealing* algorithm. When does one prefer to use the algorithm? What is the role of the parameter “temperature” in the algorithm?

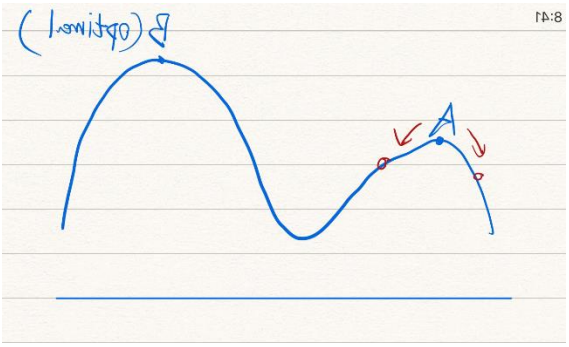
SIMULATED ANNEALING: If J(Y(i+1))>= J(Y(i)) (that is, a better solution is obtained after the move), the move is always accepted. If J(Y(i+1))< J(Y(i)) (that is, the solution after the movement is worse than the current solution), the movement is accepted with a certain probability, and this probability gradually decreases over time (gradually Only by lowering can it tend to be stable). The calculation of "a certain probability" here refers to the annealing process of metal smelting, which is also the origin of the name of the simulated annealing algorithm.

WHEN: When the hill climbing method finds the local optimal solution. To find the maximum value of f(x), the mountain climbing algorithm will stop searching when it reaches point A, because the values around point A are less than the value of point A. The solution adopted by the simulated annealing algorithm is to select the points on both sides of A with a certain probability, although the points on both sides of A are not locally optimal solutions, so there is a certain probability to search for point D, and then search for point B, and finally obtain Global optimal solution.
TEMPERATURE: When the temperature is T, the probability of a temperature drop with an energy difference of dE is P(dE), expressed as: P(dE) = exp(dE/(kT)) where k is a constant, exp represents the natural exponent, and dE <0. The higher the temperature, the greater the probability of a cooling with an energy difference of dE; the lower the temperature, the smaller the probability of cooling. Also, since dE is always less than 0, dE/kT <0, so the value range of P(dE) is (0,1). As the temperature T decreases, P(dE) will gradually decrease. By slowly lowering the temperature, the algorithm may eventually converge to the global optimal solution

Q3

3. When does *Simulated Annealing* perform better than *Hill Climbing*? How is this better performance achieved? Would you ever prefer *HC* to *SA*? If yes, when?

When the hill climbing method finds the local optimal solution.



The simulated annealing algorithm has asymptotic convergence, and it has been proved in theory to be a global optimization algorithm that converges to the global optimal solution with probability l;The first step is to generate a new solution in the solution space from the current solution by a generating function, and the second step is to calculate the difference of the objective function corresponding to the new solution. The third step is to judge whether the new solution is accepted. The judgment is based on an acceptance criterion. The most commonly used acceptance criterion is the Metropolis criterion. The fourth step is to replace the current solution with the new solution when the new solution is confirmed to be accepted.
However, because the simulated annealing algorithm is a random algorithm, when there is no local optimal solution but only a global optimal solution, the efficiency of the hill climbing algorithm is higher than that of the simulated annealing algorithm

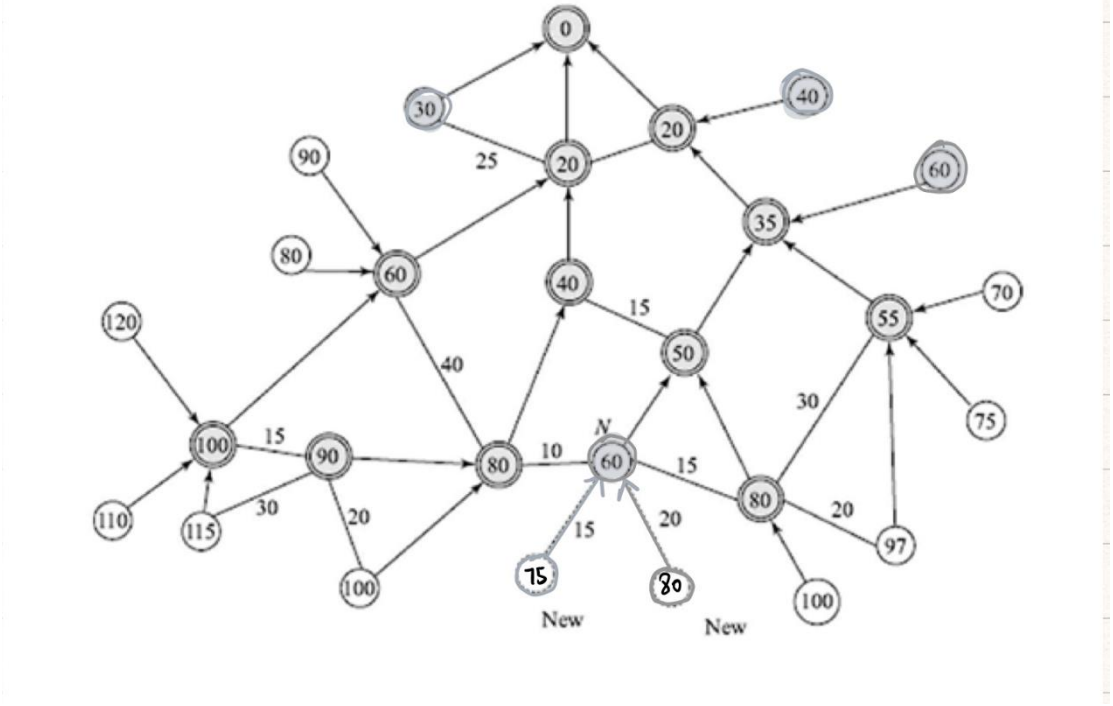
Q4

5. What is the motivation and strategy behind Genetic Algorithms? Under what conditions are GAs likely to perform better than other optimization algorithms?

In the genetic algorithm, we first need to map the problem to be solved into a mathematical problem, which is the so-called "mathematical modeling", then a feasible solution to this problem is called a "chromosome". A feasible solution is generally composed of multiple elements, then each element is called a "gene" on the chromosome.
The strategy of Genetic Algorithms is:
1. Randomly generate population
2. Judging the fitness of the individual according to the strategy and whether it meets the optimization criteria. If so, output the best individual and its optimal solution, and end. Otherwise, proceed to the next step
3. Choose parents based on fitness. Individuals with high fitness have a high probability of being selected, and individuals with low fitness are eliminated.
4. Use parental chromosomes to cross over according to a certain method to generate offspring
5. Mutation of offspring chromosomes
Generate a new generation of population by crossover and mutation, return to step 2 until the optimal solution is generated
When it is necessary to solve global optimization problems in large-scale, multimodal and polymorphic functions, including discrete variables, etc., genetic algorithm can handle the constraint conditions well and obtain the optimal solution faster

Q5

1. Figure 5.33 below shows the graph being explored by algorithm *A** at the point when node *N* is about to be expanded. The shaded double circles represent nodes on CLOSED, and the unshaded circles represent nodes on OPEN. Values inside the circles represent *g*-values of nodes. The two nodes marked *New* in dashed circles are about to be put on *OPEN*. The arrows depict back pointers and labels on edges denote costs.
   Redraw the graph after *A** has finished expanding the node *N*.


0
30
40
20
25
20
90
60
35
80
60
40
15
70
120
55
50
40
30
75
N
100
15
90
80
10
60
15
80
20
110
115
30
20
20
15
97
75
80
100
100
New
New

Q6

2. Given the two graphs from Chapter 3 reproduced below, the task is to find an optimal path from the start node *S* to the goal node *G*. The length of each edge is marked on the graph. Use the Manhattan distance as the heuristic function. Assume that each unit on the grid is 10 kilometres.
   (a) List the nodes that are on *OPEN* or *CLOSED* when *A** terminates, and indicate their *f*-values.
   (b) Show the order in which *A** adds nodes to *CLOSED*.
   (c) Mark the parent pointers on the graph as and when they are assigned. Show clearly if they are reassigned.

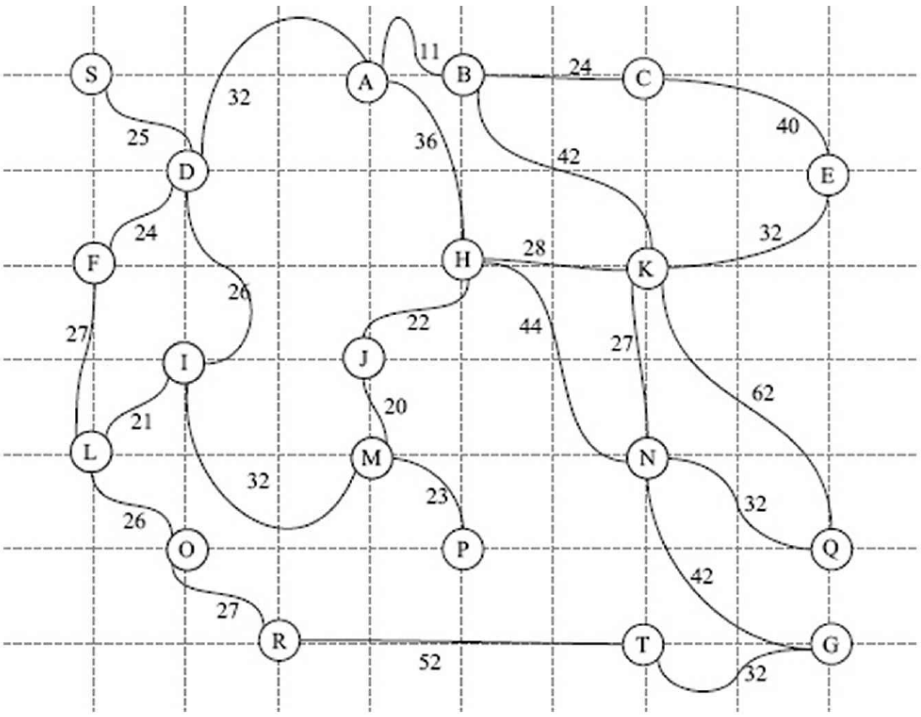
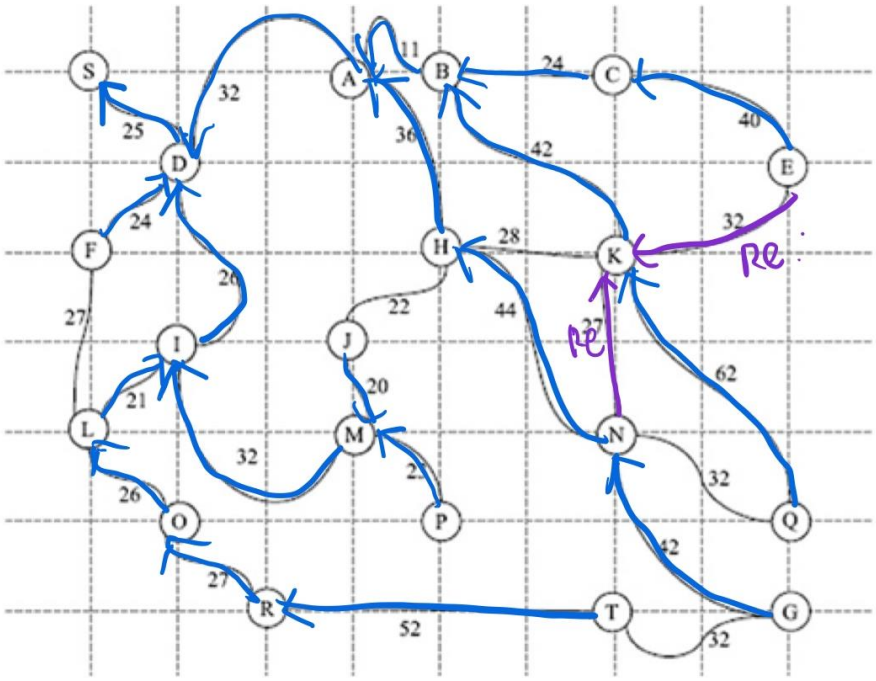


FIGURE 5.34 The graph from Figure 3.22 with weights assigned to edges.

(a) OPEN:
T-179,
CLOSED:
S-14, D-37,,I-61, F-61,A-68, L-72, B-78, M-90, C- 100, H-101, O-106, J-111, P-111, K-116, R-131, E-137, N-141,Q-173, G-179

(b) S D I F A L B M C H O J P K R E N Q G

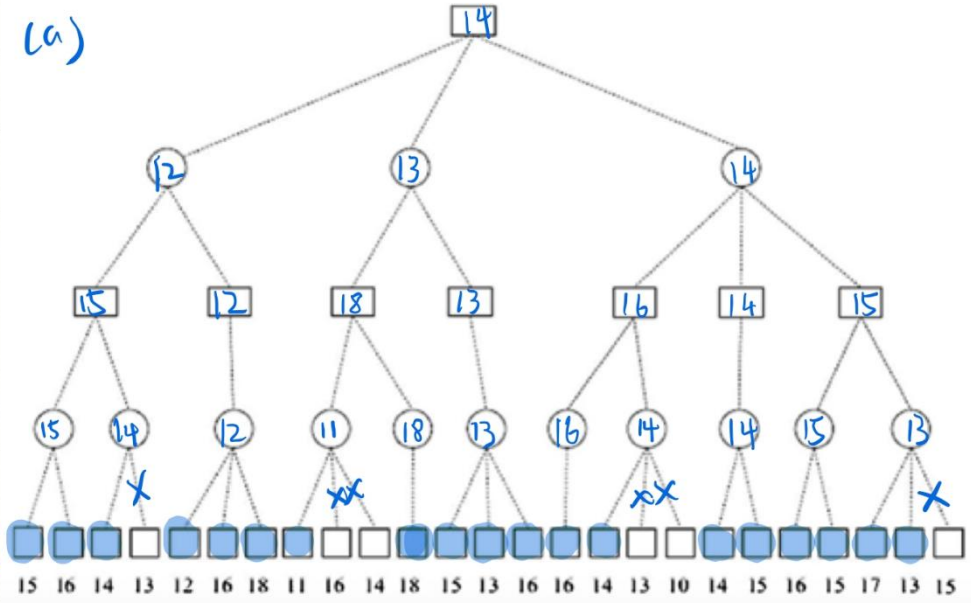


(c)

Q7

14. Show how the algorithm *AlphaBeta* explores the game tree, searching from left to right.
    (a) Fill in the leaves that are inspected by *AlphaBeta*.
    (b) Show the cutoffs and label them with their type.
    (c) Mark the move that *AlphaBeta* will choose for *MAX* at the root.

(a)



(b)

all the cutoffs are optimal cutoffs.

(up and lower bounds pruning)

(c)