



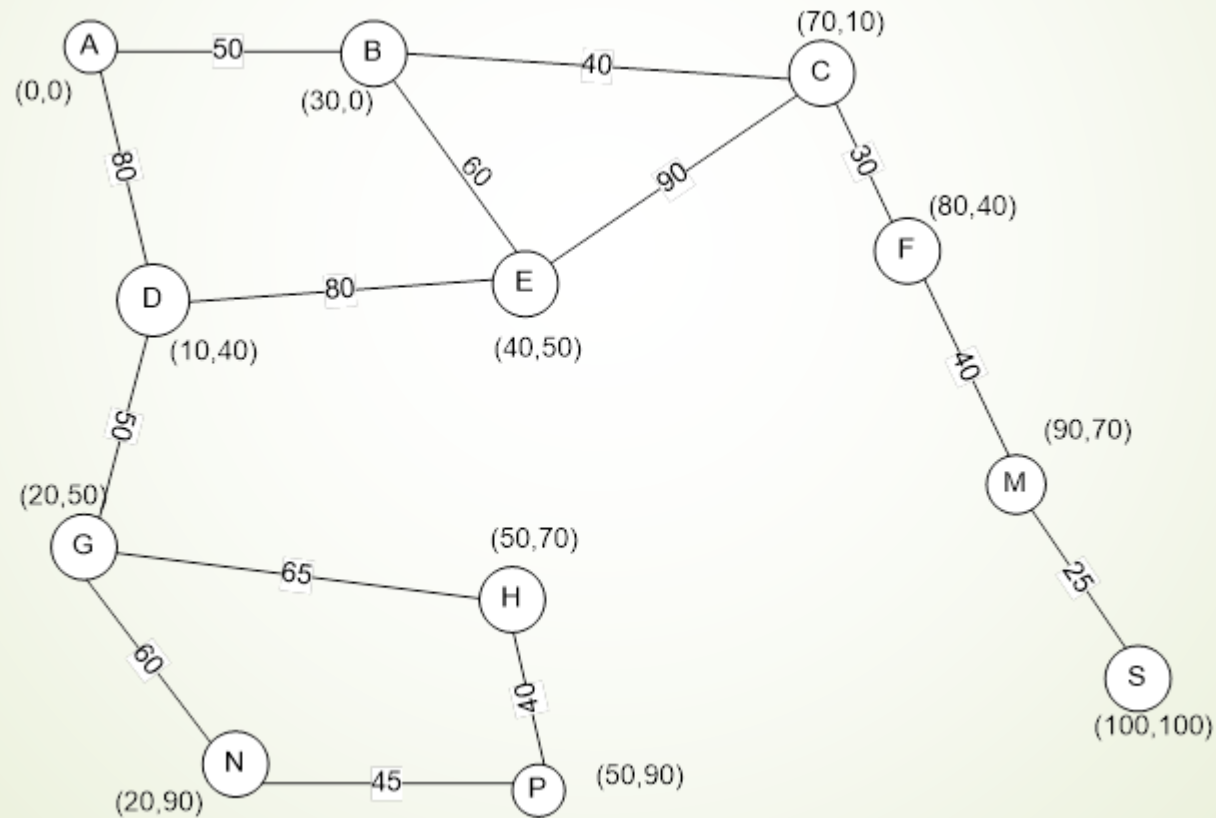
Artificial Intelligence

Examples

2

Example 1

- Do the search in the following graph:



3

Example 1

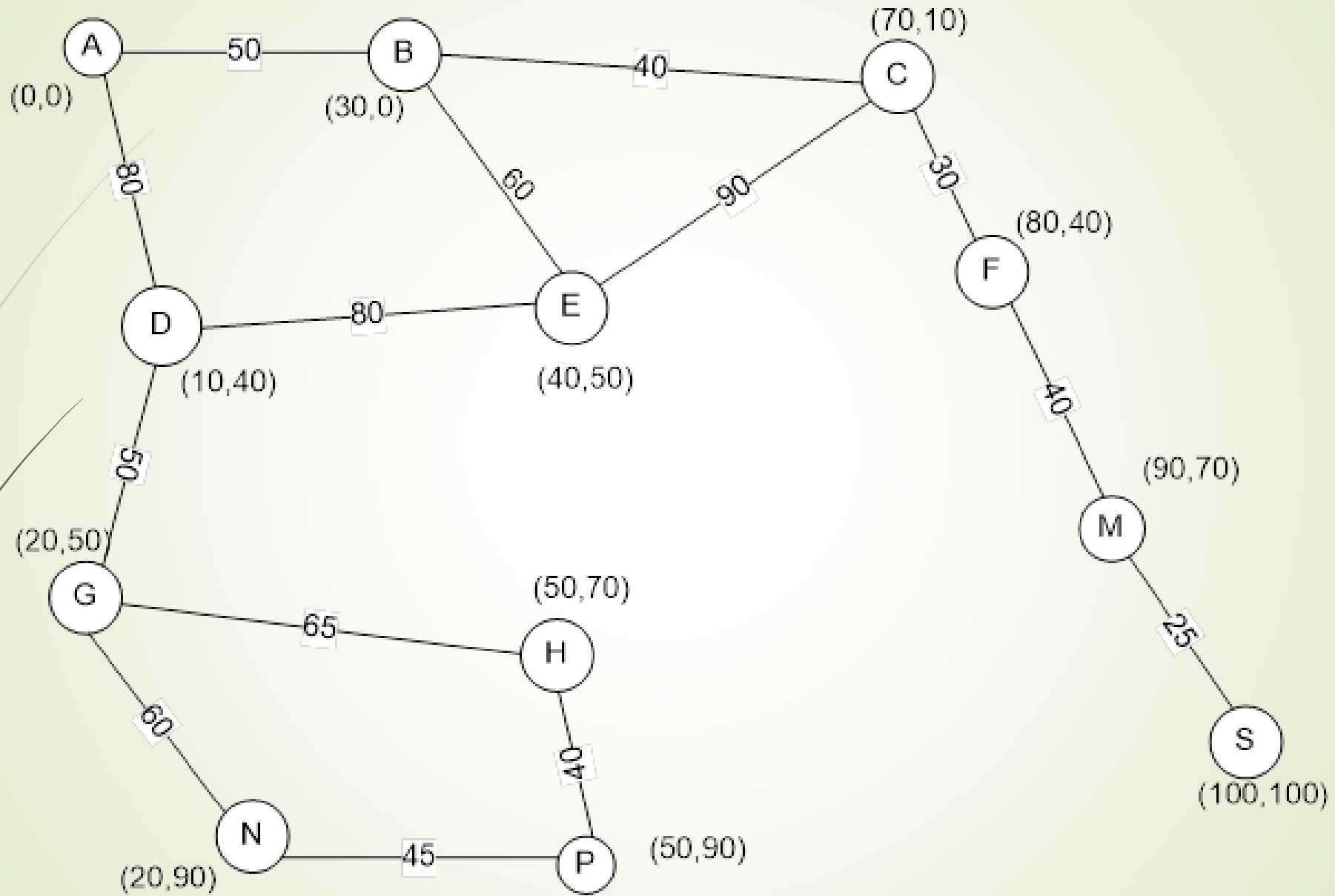
- ▶ Starting from city **A** we want to find the shortest path to city **S** (goal). The labels on the links show the distance between the cities. The coordinates of each city is available beside the nodes. In each step of the search we expand only one level so we can see the coordinates of the neighboring cities and the distance to each one of them.
- ▶ To estimate the distance from a city to the goal use city block distance (the distance between $X(a,b)$ and $Y(c,d)$ is $|a-c| + |b-d|$)
- ▶ Using the following algorithms determine the path to goal and discuss/compare the result of each method with the results of the other methods
- ▶ Depth First Search: Use one branch. Continue until there is no node to visit, or the goal is found. If the goal is not found, go back one step and try another link. The links are selected in the order of storing them (pointer list)
- ▶ Breadth First Search: Visit all child nodes at distance one from the initial state. Then visit all nodes at distance two, and so on, until either the goal is found or there is no node to continue.

4

Example 1

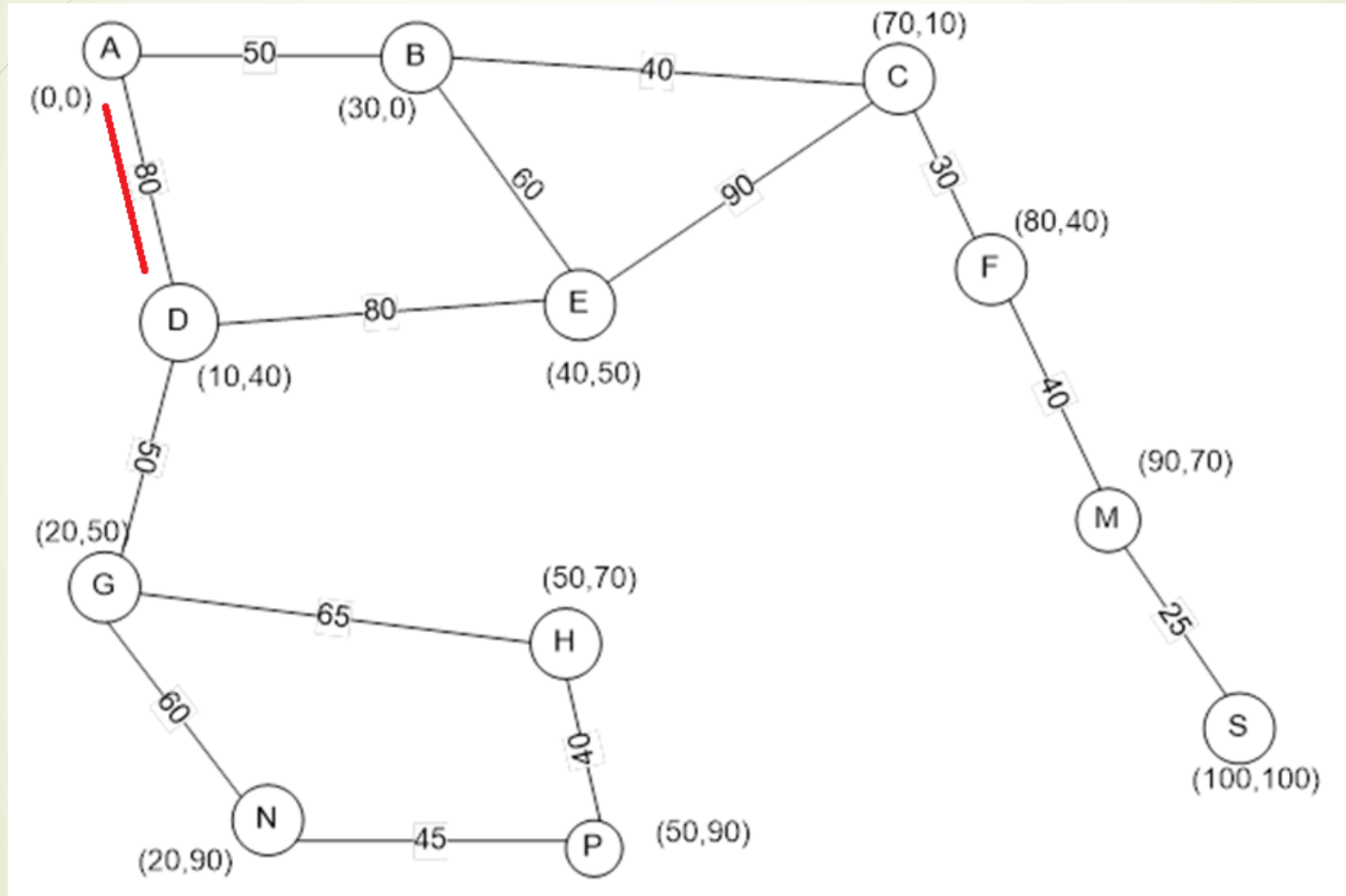
- Algorithm used by Uniform, greedy and A*
- Start with initial node. Put it in the list of visited nodes. Find the children of the nodes in the visited node list
- Choose a child. Add it to the list of the visited nodes. Again find the children of the nodes in the visited nodes list and choose one
-
- Uniform Search: In using the algorithm above, choose the child with shorter distance to initial node
- Greedy: In using the algorithm above, choose the child with shorter distance to the goal node
- A*: In using the algorithm above, choose the child with shorter distance to initial node + goal node
- Hill Climbing: Check the neighbors of the current node. Go the neighbor which is closer to the goal (should be also closer than the current node)
- Simulated Annealing (same as Hill Climbing but if the neighbors are all farther to the goal than the current node, still one or two neighbors will be visited)

5



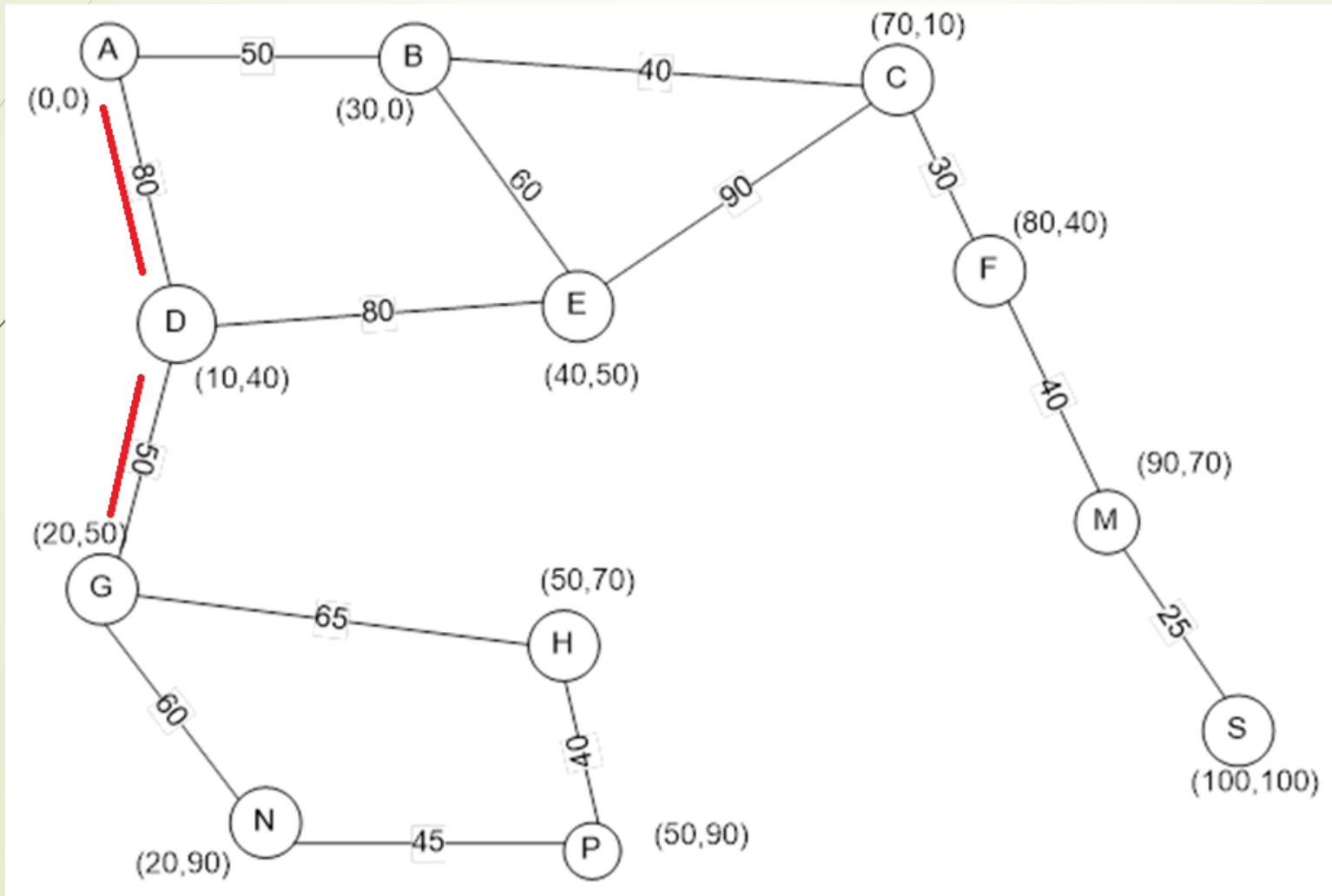
6

Depth First Search



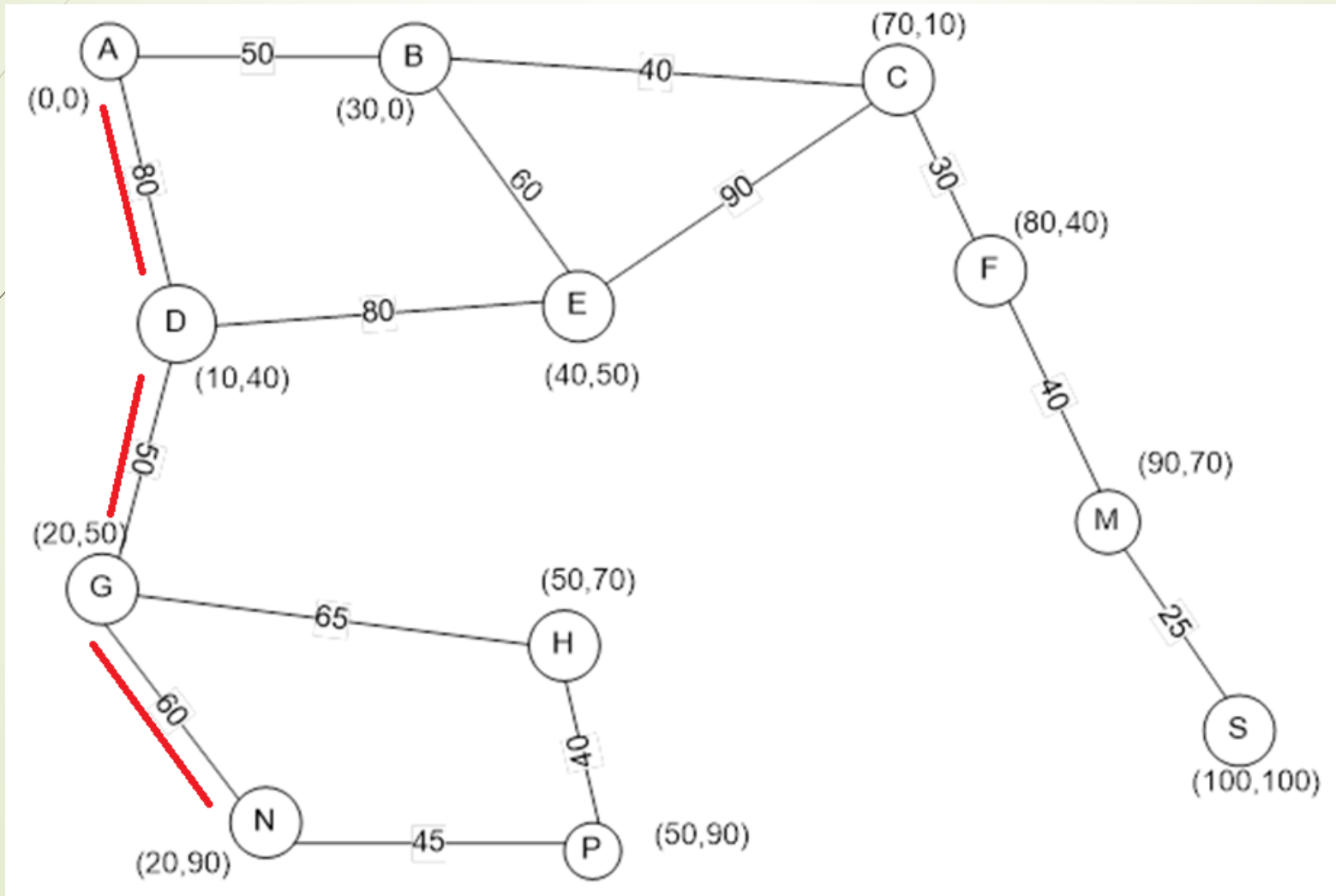
7

Depth First Search



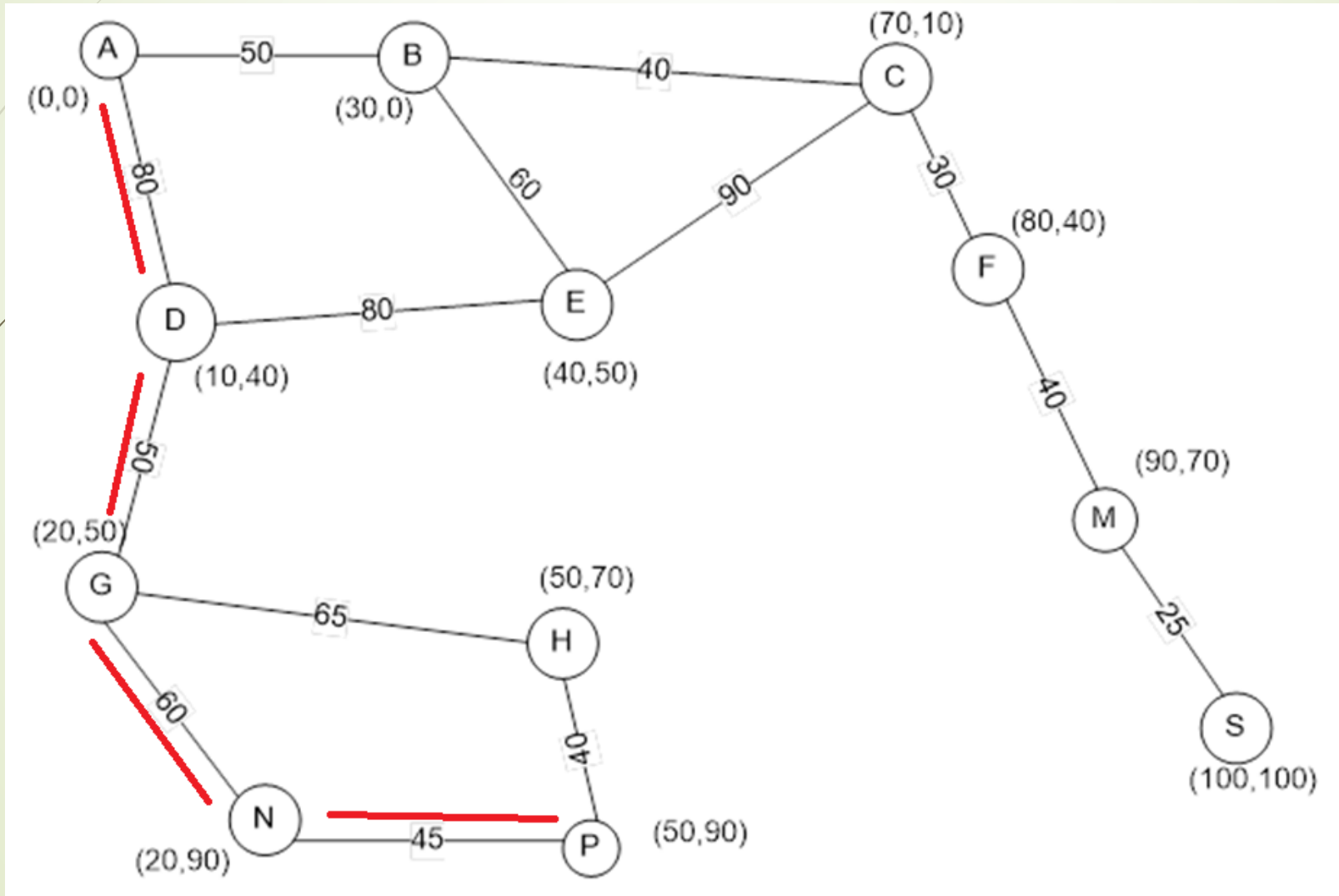
8

Depth First Search



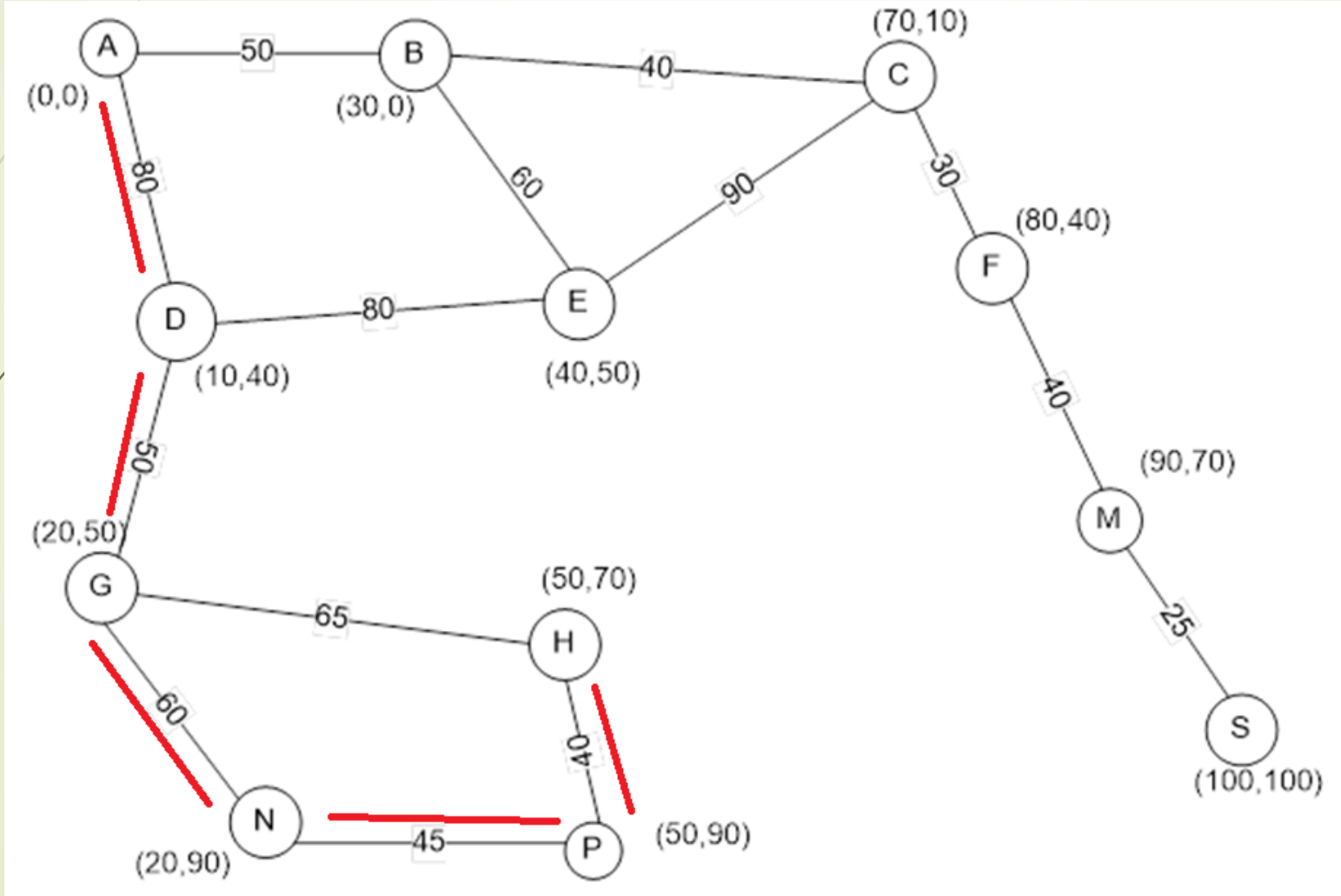
9

Depth First Search



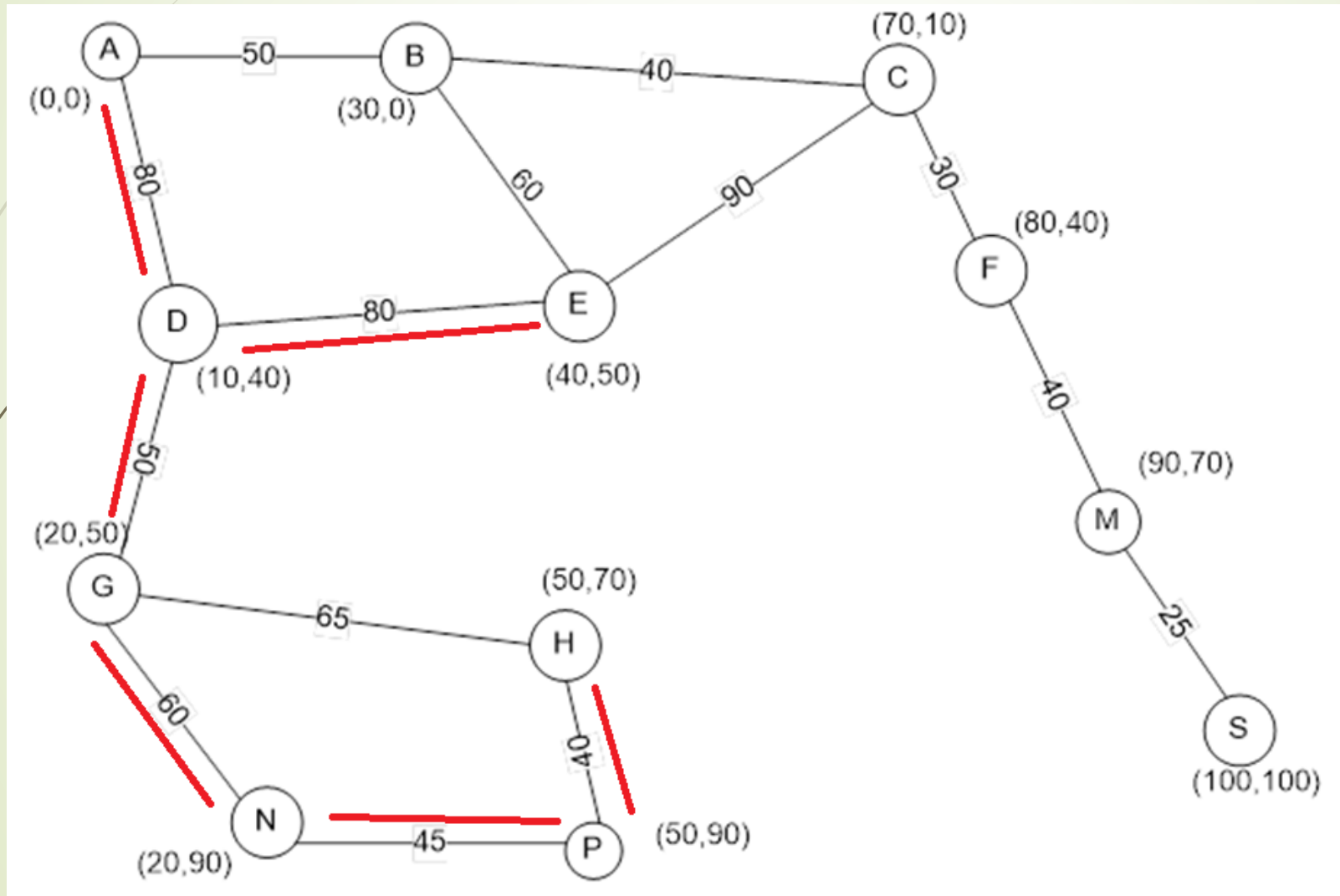
10

Depth First Search



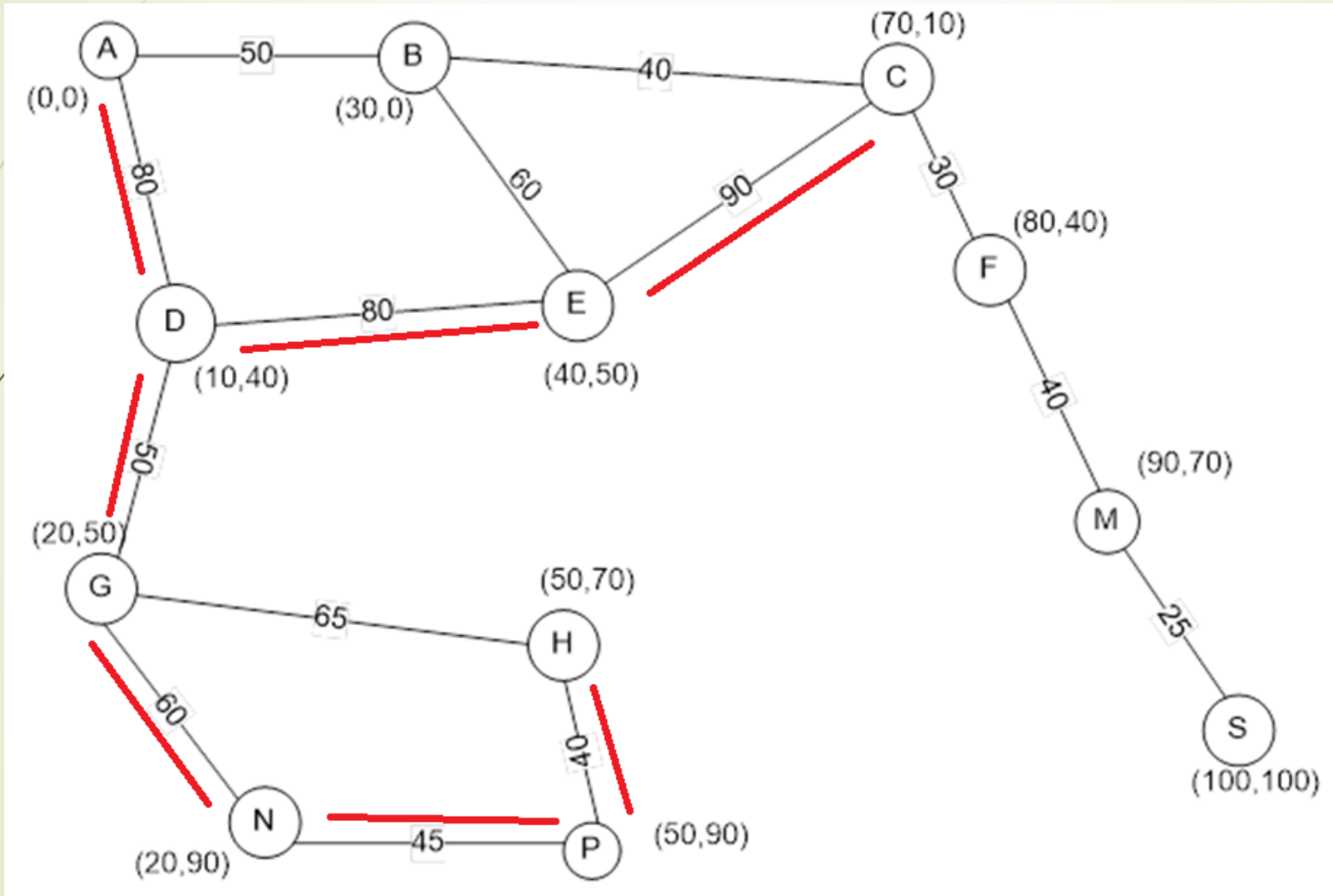
11

Depth First Search



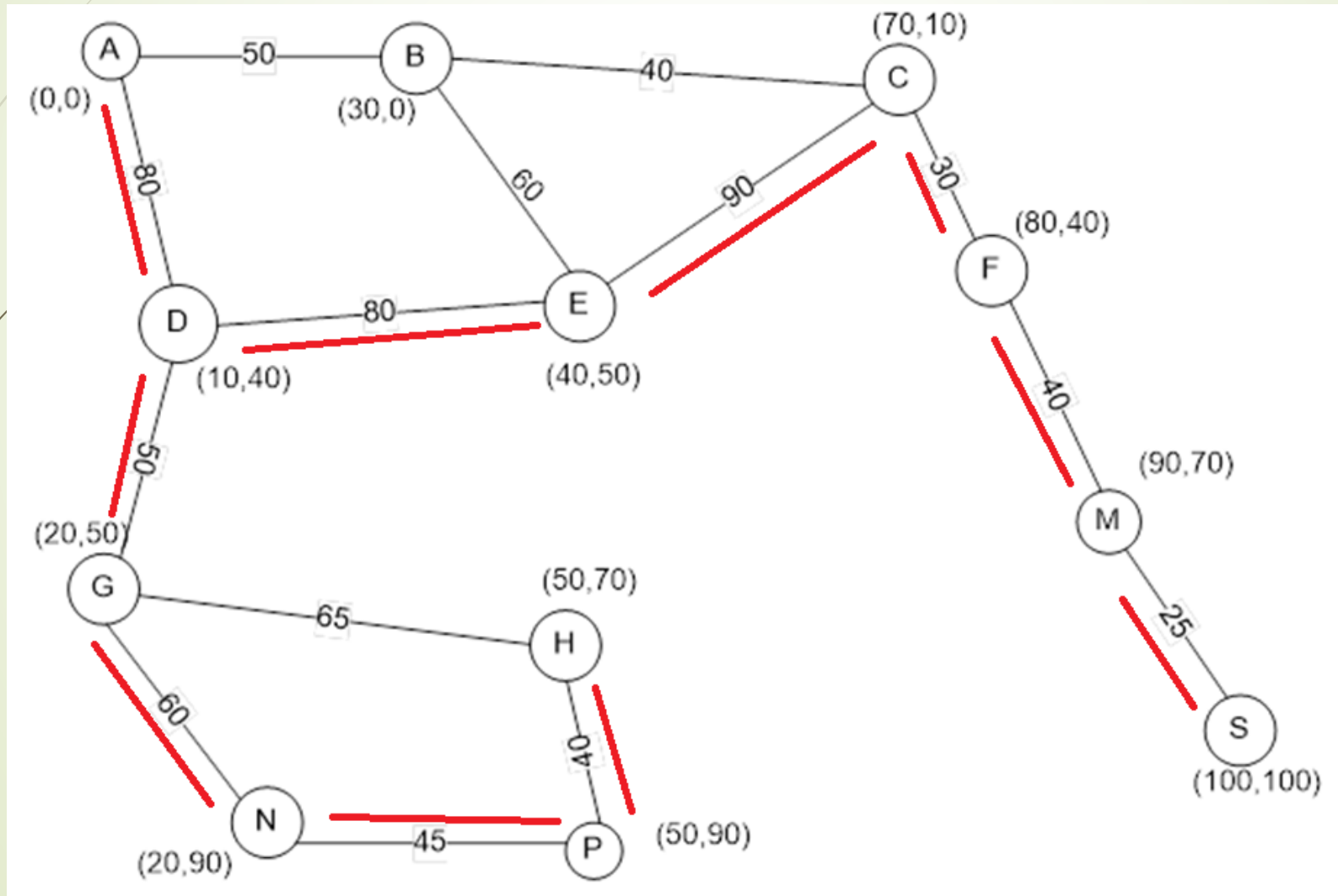
12

Depth First Search



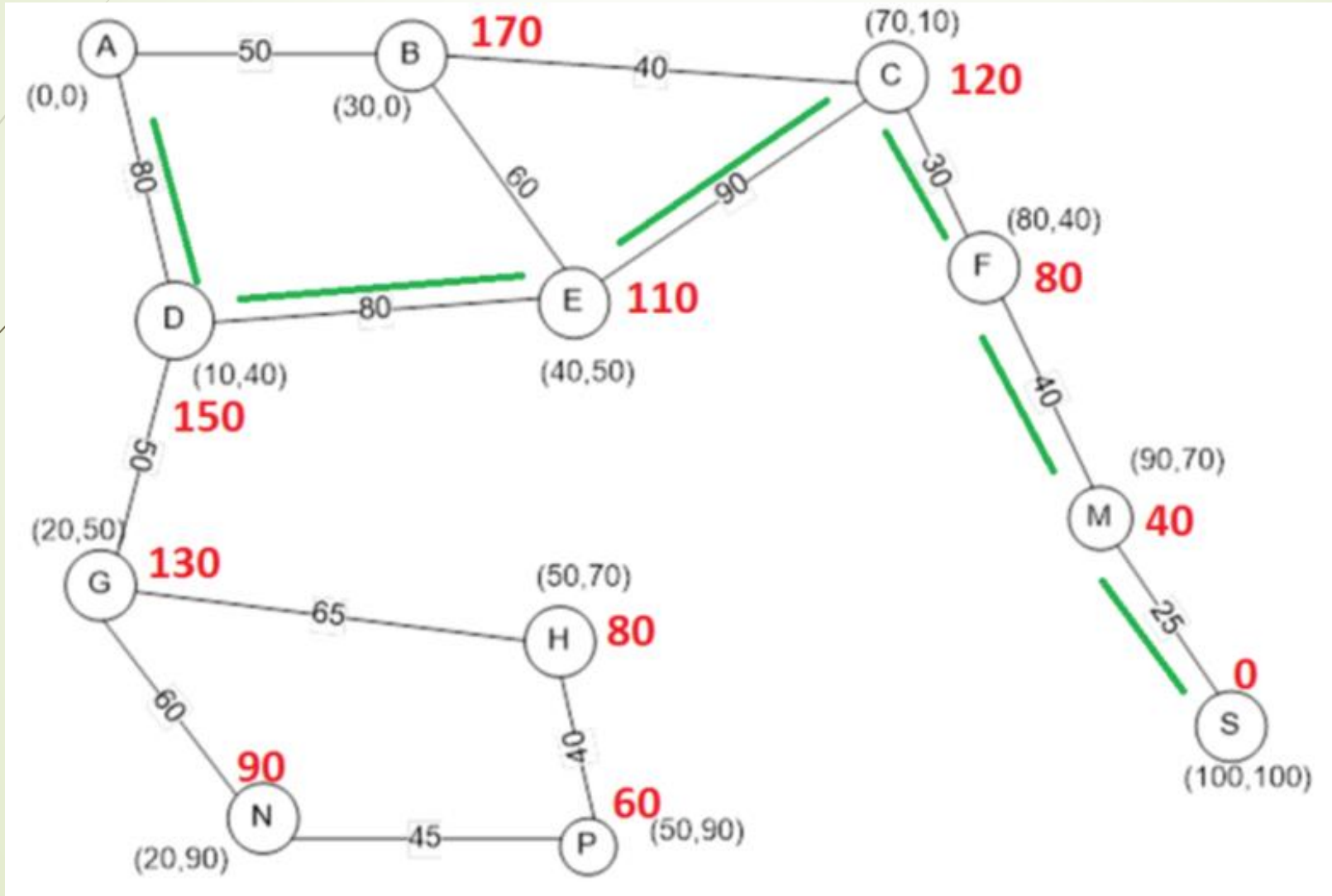
13

Depth First Search



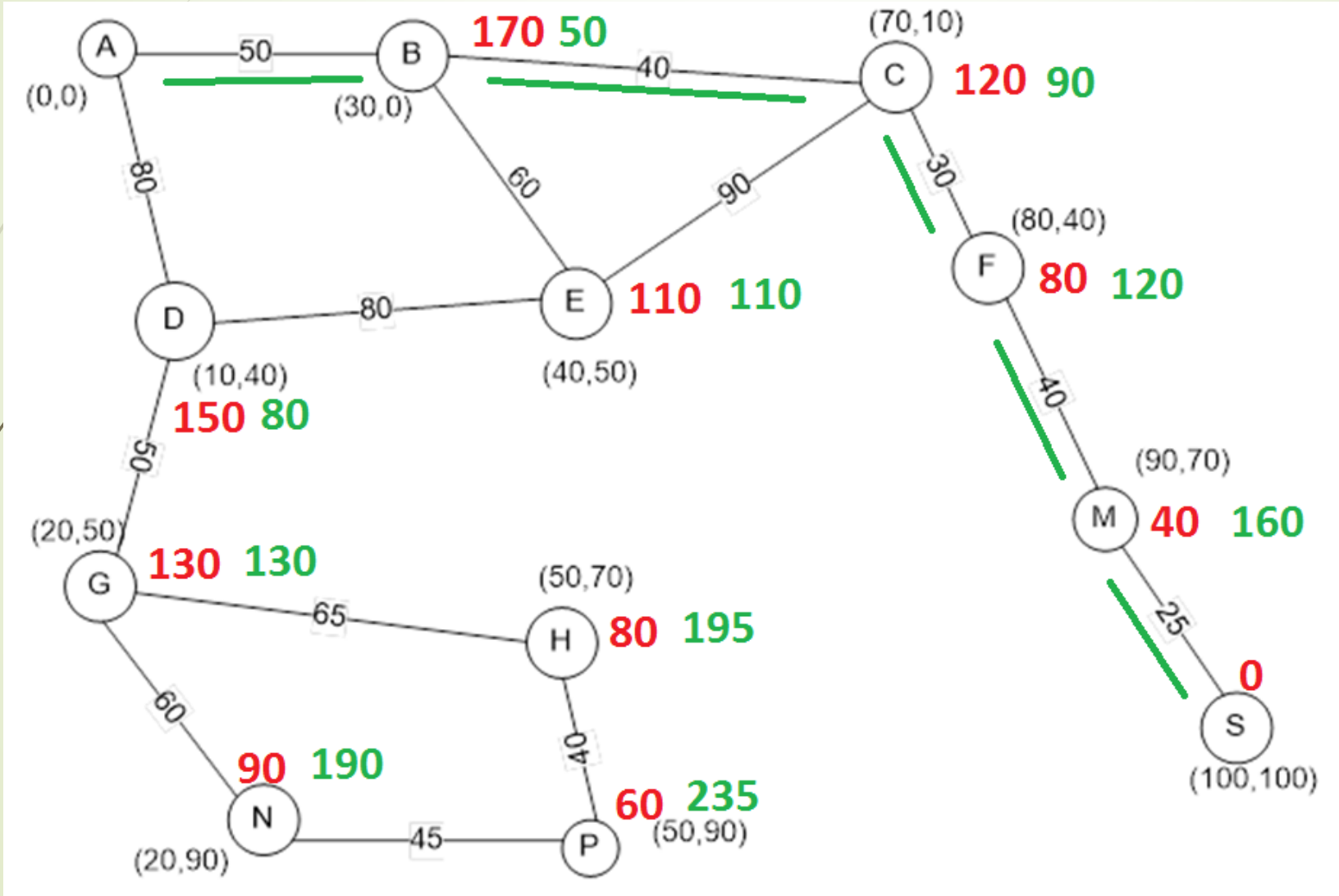
14

Greedy



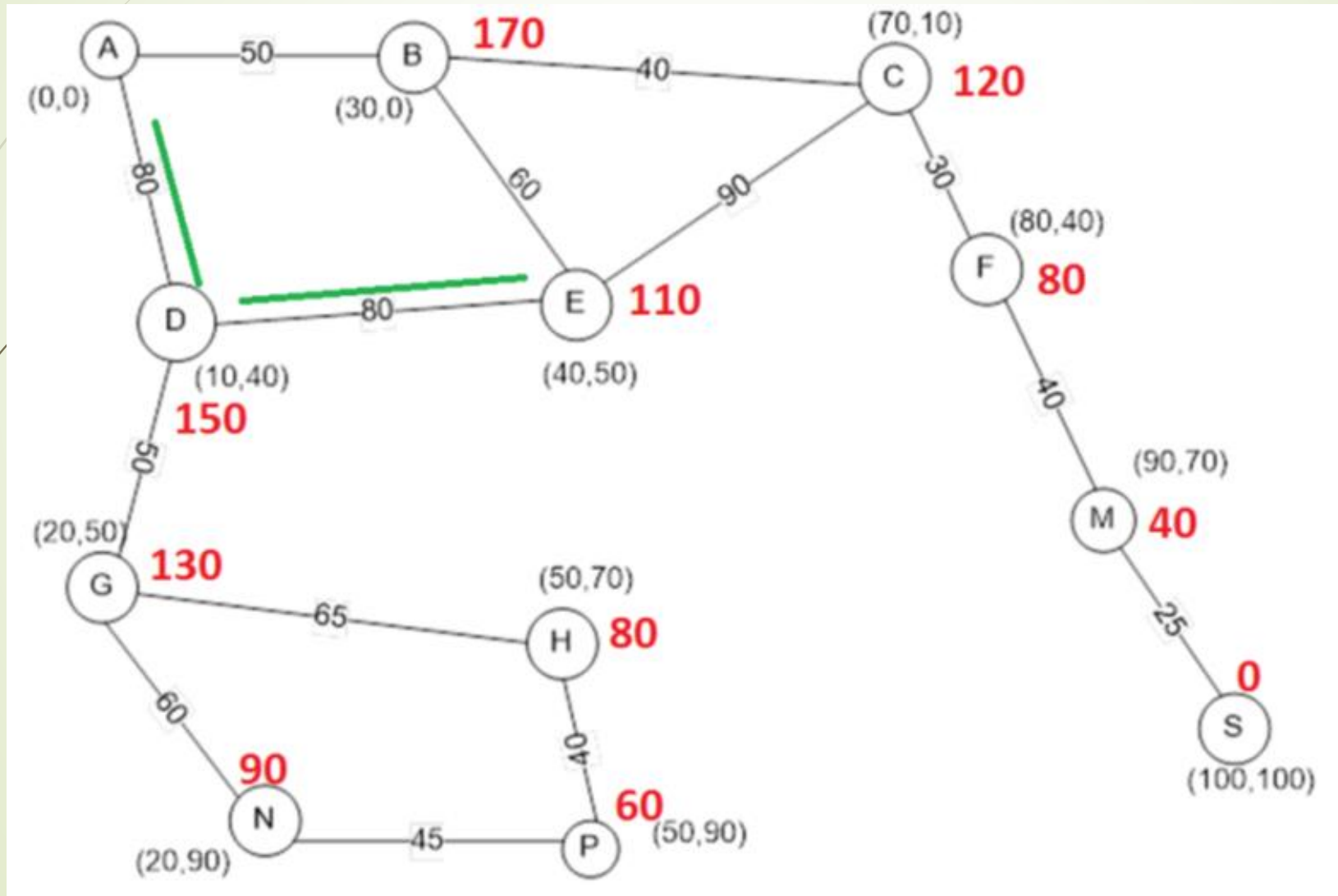
15

A*



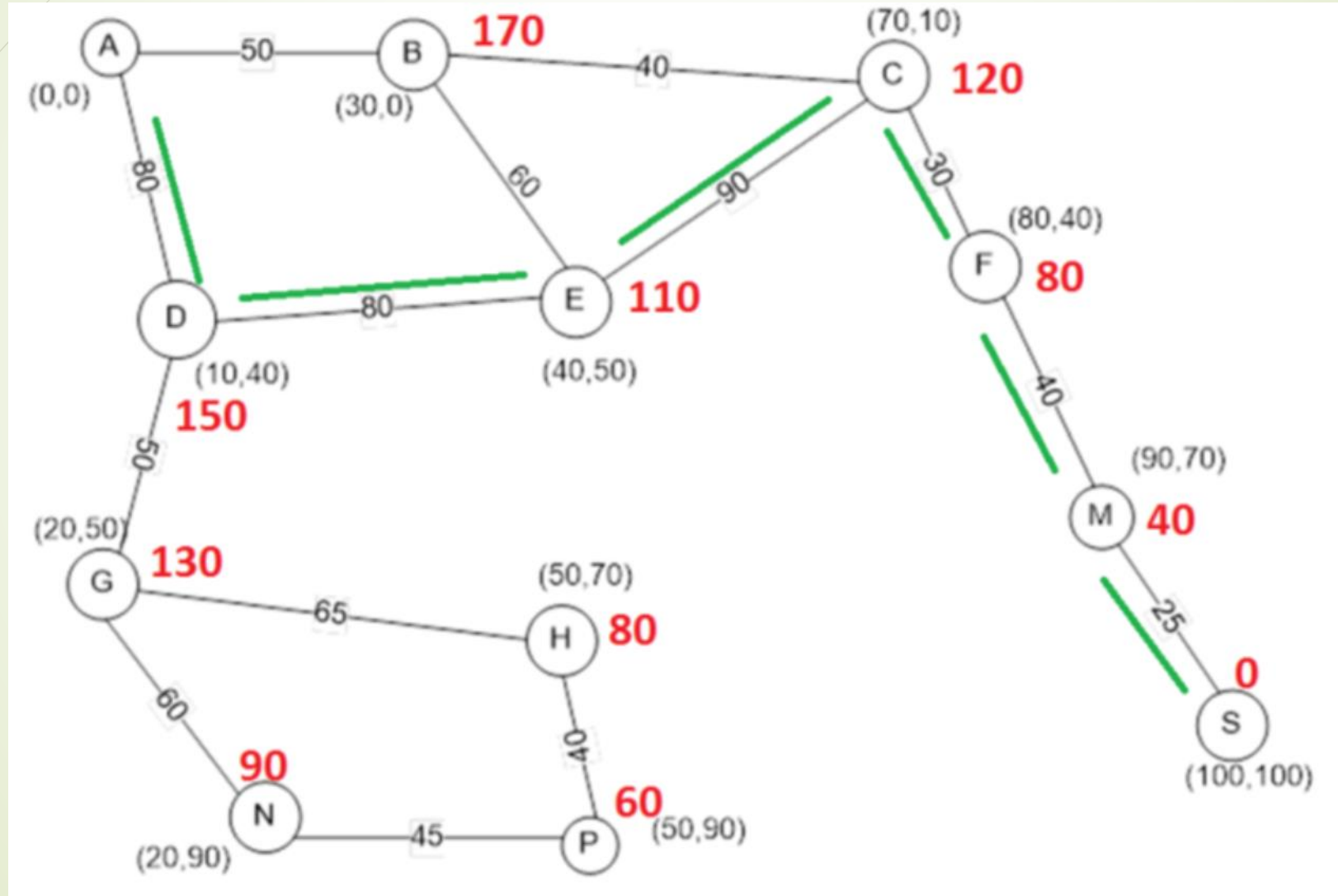
16

Hill Climbing



17

Simulated Annealing



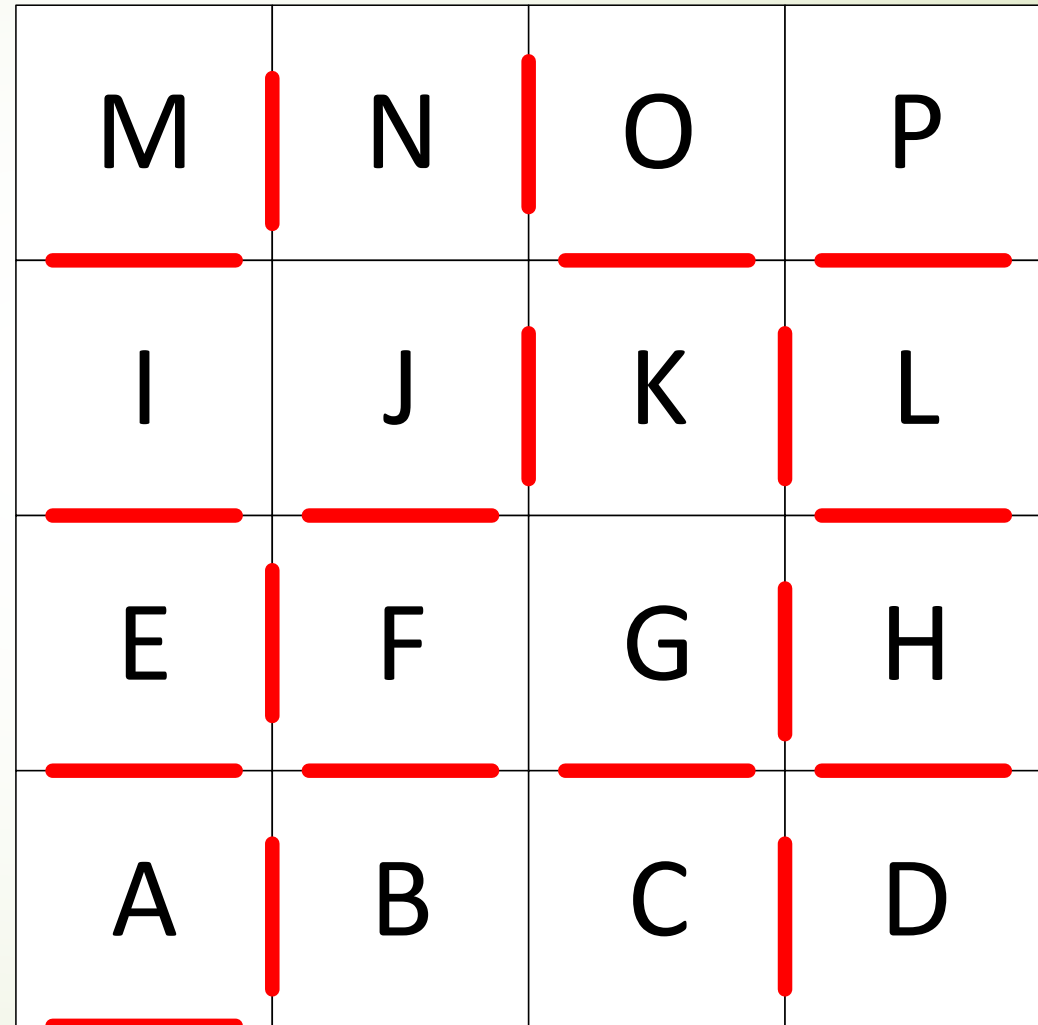
Example 2

- Assume a building has 16 rooms located in a rectangular grid as shown in the image (next slide)
- Between some rooms there is a door letting people pass from one room to the next.
- A person is in one of the rooms of this building. He has a device which broadcasts a signal continuously.
- An agent should locate and help this person.
- Find the settings to guide the agent

19

The rooms

➔ Red line segments show doors

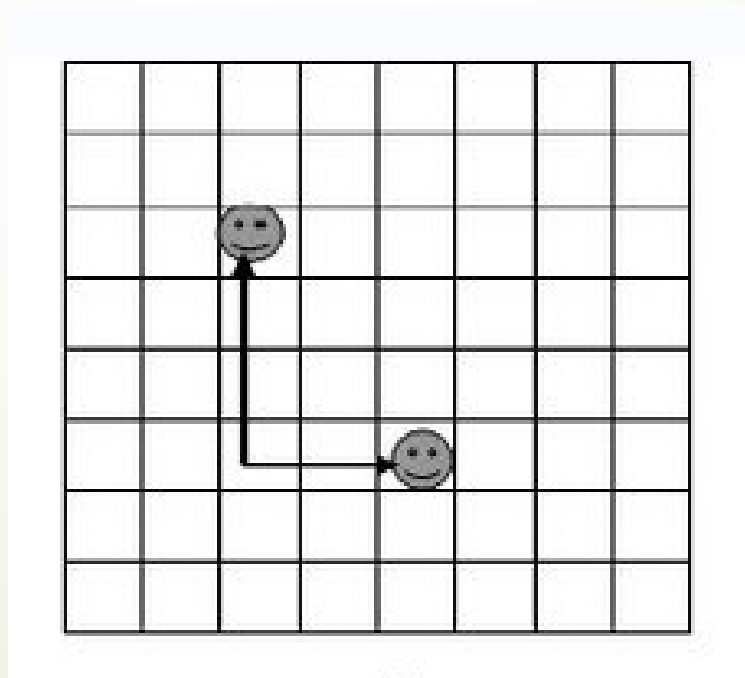


State Space

- Assume each room is a node in the state space graph.
- The adjacent rooms are connected in the state space if there is a door between them.
- The agent can estimate the distance to the person (**count the number of rooms using city block distance method**)

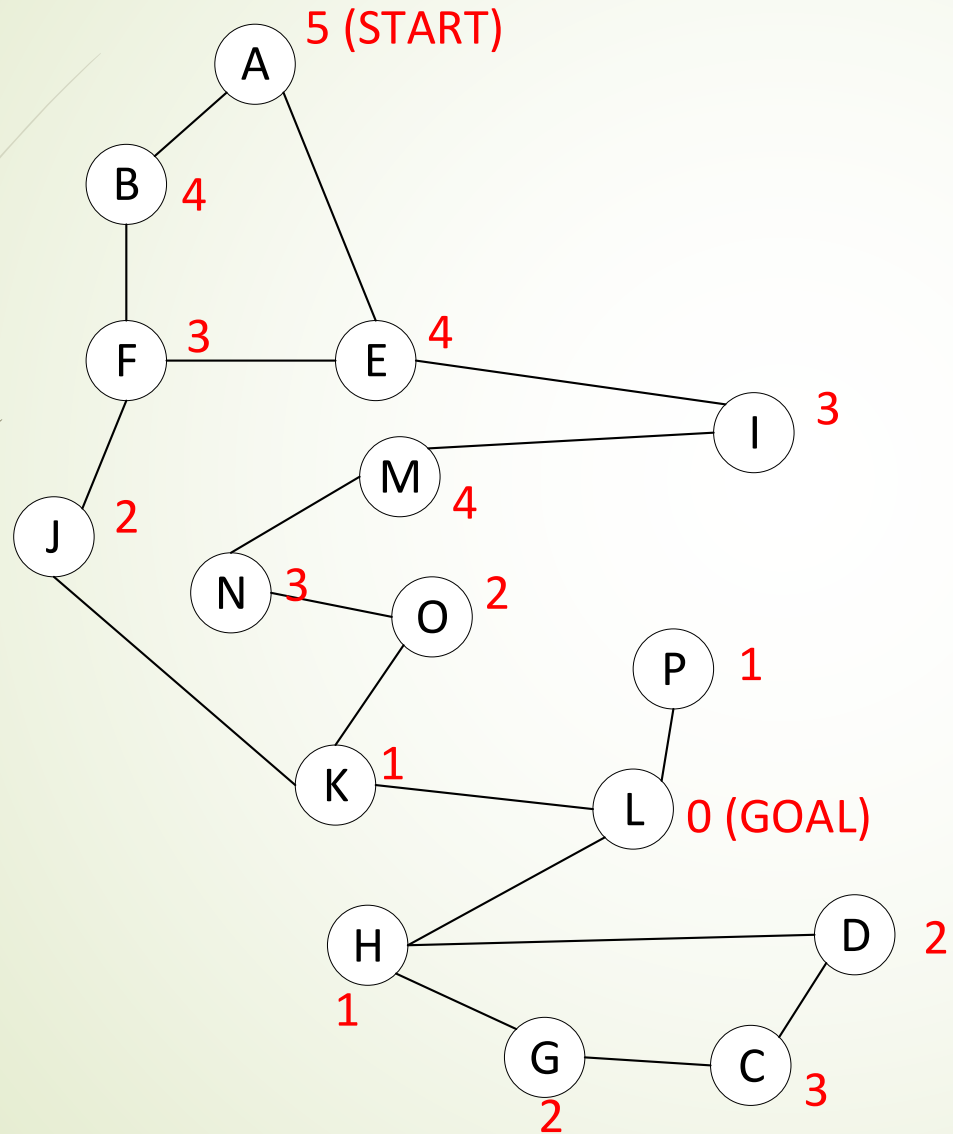
City Block Distance

- Add the number of horizontal and vertical blocks to the destination



The State Space Graph

22



M	N	O	P
I	J	K	L
E	F	G	H
A	B	C	D

Search Using Greedy Algorithm

- The goal is finding the person as soon as possible
- The agent can estimate the distance to the person
- Assume the person is at room L, and agent enters the building from A.

Searching with Greedy Algorithm

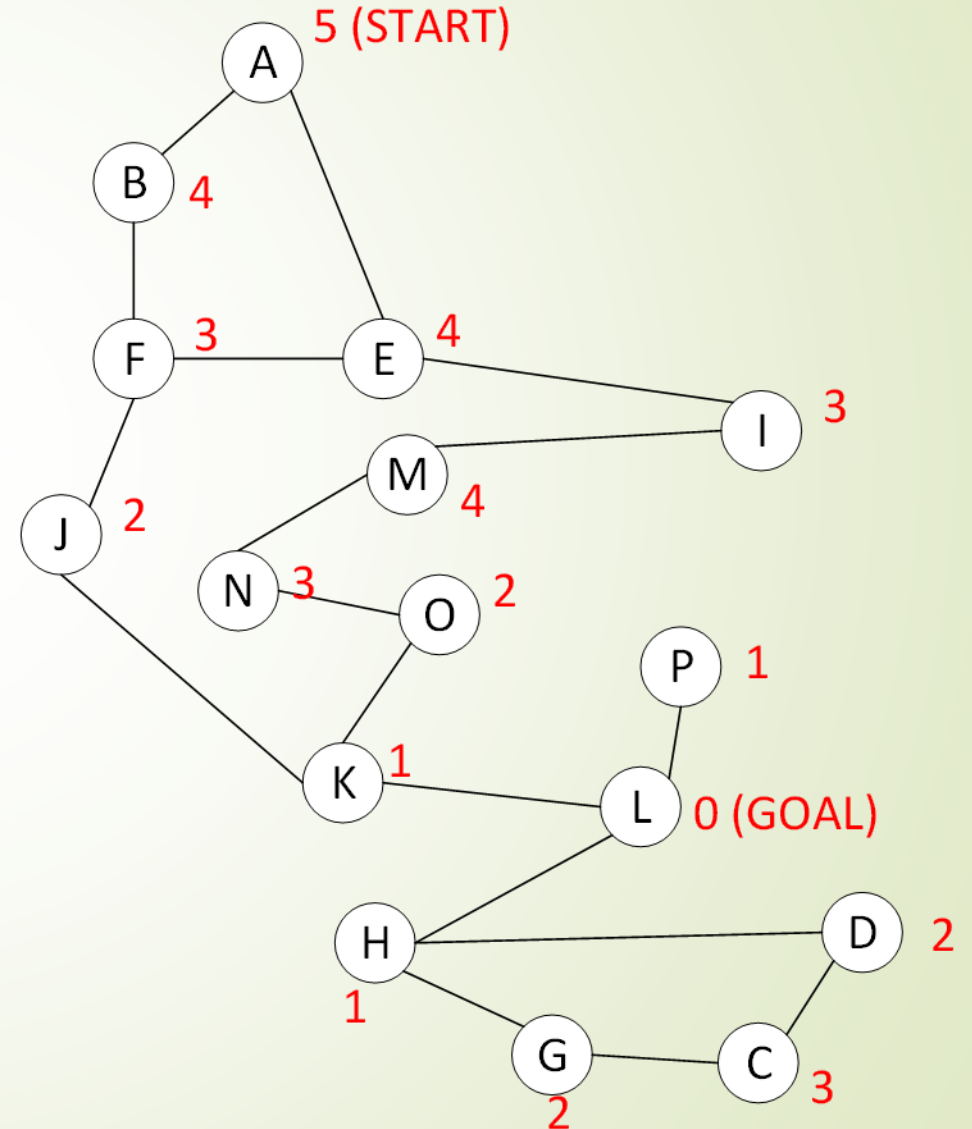
► Case 1:

- If two neighboring nodes have the same distance to the goal, choose according to the alphabetical order of their names

25

Searching with Greedy Algorithm

- Between the neighbors of A, select B
- Between neighbors of A and B select F
- Between neighbors of A, B, F select J
- Between neighbors of A,B,F,J select K
- Between neighbors of A,B,F,J,K select L (Goal found)



Searching with Greedy Algorithm

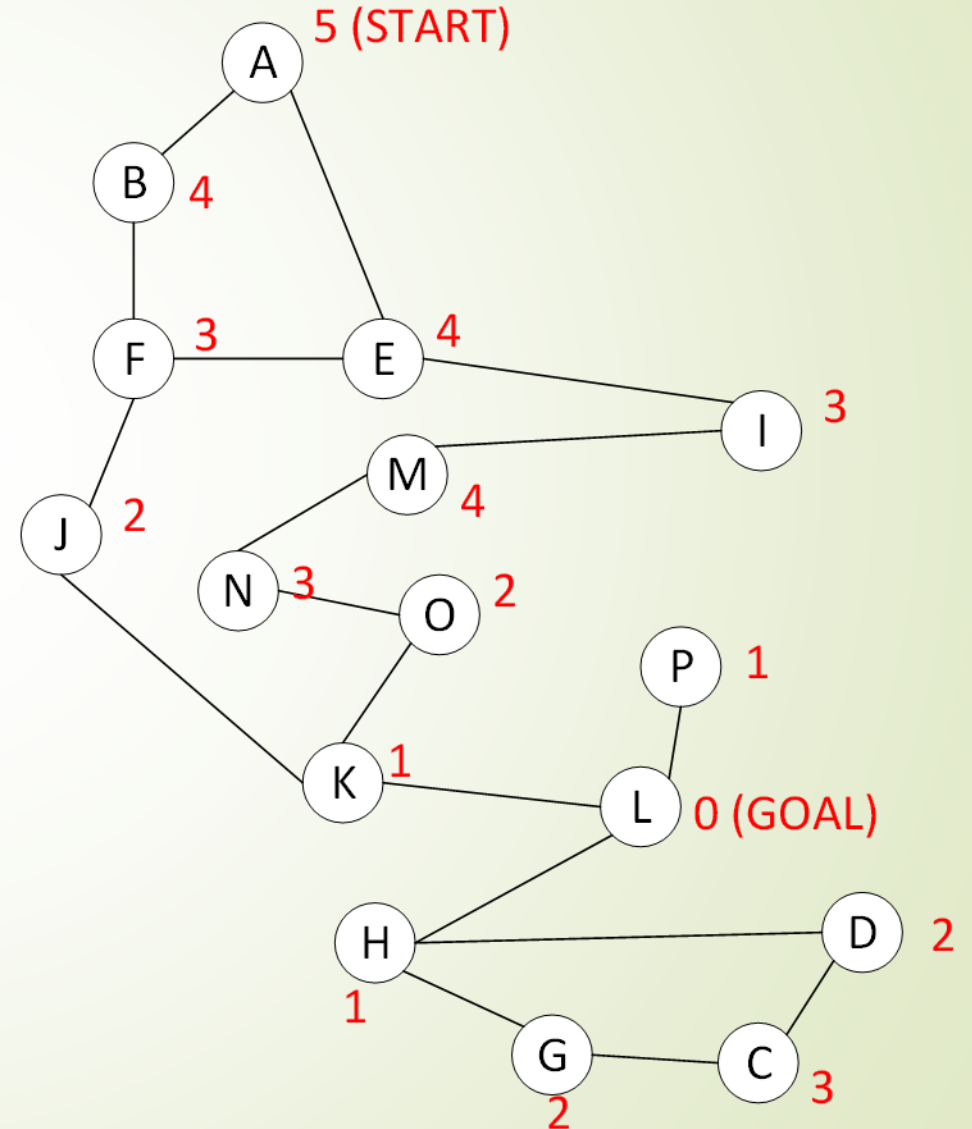
► Case 2:

- If two neighboring nodes have the same distance to the goal, choose according to the reverse alphabetical order of their names

27

Searching with Greedy Algorithm

- Between the neighbors of A, select E
- Between neighbors of A and E select I
- Between neighbors of A, E, I select F
- Between neighbors of A,E,I,F select J
- Between neighbors of A,E,I,F,J select K
- Between neighbors of A,E,I,F,J,K select L (Goal found)



Example 3: Knowledge Base Creation

- ▶ Assume a path_finder agent is trying to reach location B starting from location A. The agent has access to a map of the neighborhood and the connecting roads. The goal is finding the fastest path to reach B.
- ▶ The roads however, can be under construction from time to time. In these cases either the traffic goes more slowly (the agent can find out the new speed of the road), or the road is blocked altogether. In case that the road is blocked, the agent has to find a different path.
- ▶ The agent has its own copy of the graph of the locations, and the roads connecting them including the distances. The agent can update the graph using the data from its percepts.
- ▶ The agent can move forward, turn left or right (to side roads), U turn and go backward. In rainy days the agent is not allowed to make a U-turn or go back.
- ▶ The speed limits of the roads depends on the weather (sunny, rainy; in rainy weather the speed limit is reduced by 30%) and the time (at night the speed limits are reduced by 20%)

Modeling the Environment

- The environment is made of a set of locations and roads connecting them to each other
- The roads have attributes such as distance, speed limit and state
- The roads have conditions such as rainy, blocked, under construction

- The environment is a dynamic environment. The agent should percept the conditions and update the environment.

Knowledge-base using Propositional Logic

- ▶ List the locations:
 - ▶ P1: A is a location
 - ▶ P2: B is a location

- ▶ Define the roads connecting the locations:
 - ▶ Q1: C is connected to F
 - ▶ Q2: G is connected to P

Knowledge-base using Propositional Logic

- Define the attributes of the locations and roads:
 - R1: The road connecting A to H is 15 km
 - R2: The speed limit on road connecting A to H is 80 km

- Define the possible percepts:
 - Read(Speed limit)
 - Read(Time)
 - Read(Weather)
 - Read(current location)
 - Read(status)


32

Knowledge-base using Propositional Logic

► Define rules:

- **Read(current location)** = Road connecting A to H \wedge **Read(status)** = blocked \rightarrow R1: The road connecting A to H is **infinity** km.
- **Read(current location)** = Road connecting A to H \wedge **Read(status)** = under construction \rightarrow R2: The speed limit on road connecting A to H is **Read(speed limit)** km
- **Read(current location)** = Road connecting A to H \wedge **Read(Weather)** = raining \rightarrow The speed limit on road connecting A to H is **Read(speed limit) /1.3** km
- **Read(current location)** = Road connecting A to H \wedge **Read(Time)** = night \rightarrow The speed limit on road connecting A to H is **Read(speed limit) /1.2** km
- **Read(current location)** = Road connecting A to H \wedge **Read(Time)** = night \wedge **Read(Weather)** = raining \rightarrow The speed limit on road connecting A to H is **Read(speed limit) /1.5** km

Actions

- ▶ The actions that the agent can take are:
 - ▶ move forward,
 - ▶ turn left
 - ▶ turn right
 - ▶ U turn
 - ▶ go backward
- ▶ Conditional moves are decided about during inference:
 - ▶ Example: **Read(*Weather*)** = raining →  Make U turn

Create the Knowledge-base

- Add all propositions as sentences to the knowledge-base
- Assume the knowledge-base is called KB
- Using TELL:
 - TELL(KB, R1: The road connecting A to H is 15 km)
 - TELL(KB, **Read(current location)** = Road connecting A to H **^** **Read(status)** = blocked → R1: The road connecting A to H is **infinity** km)
- Ask the agent to find the shortest path from A to B
 - ASK(KB, what is the shortest path from A to B)

What Agent Does?

- Agent has a graph connecting locations.
- The graph is in the form of a set of propositions (statements)
- Agent makes a copy of the original graph
- Agent runs a search algorithm to find the shortest path to its destination
- While agent is on the way, it senses the environment.
- Agent updates the graph based on its percepts
- Agent repeats the search algorithm after updating the graph

Using Predicate Logic

- ▶ Consider locations as constants
 - ▶ A,B,C,...
- ▶ Define roads as objects. Use predicates to assign attributes and connect the locations
 - ▶ Road(x)
 - ▶ RoadFrom(x, L)
 - ▶ RoadTo(x, L)
 - ▶ RoadSpeed(x, v)
 - ▶ RoadLength(x, d)

Define Rules

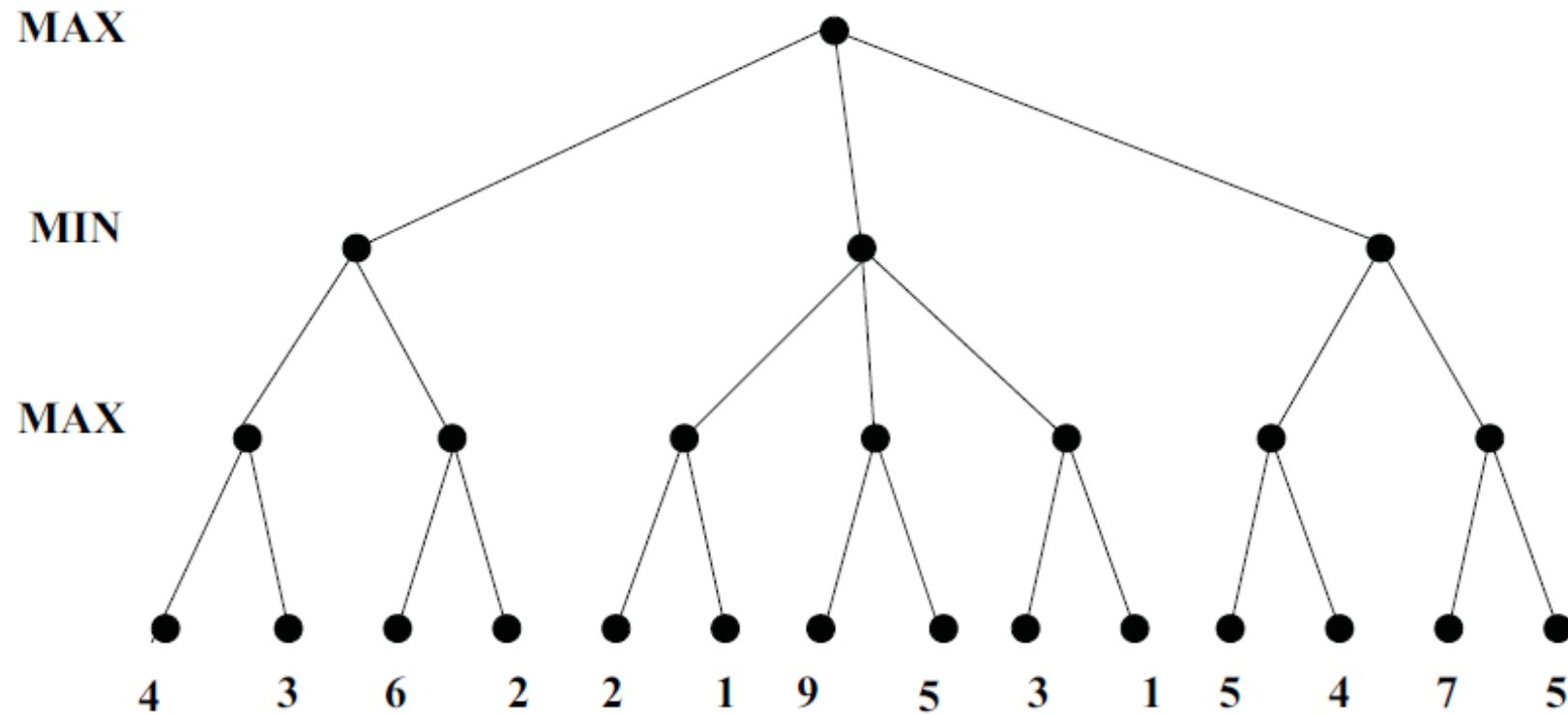
- $\forall X, \text{Road}(X), \text{Weather}(X) = \text{Raining}, \text{Speed}(X) = V \rightarrow \text{RoadSpeed}(X, V/1.3)$
- $\forall X, \text{Road}(X), \text{Status}(X) = \text{Blocked} \rightarrow \text{RoadLength}(X, \infty)$

Alpha Beta Pruning

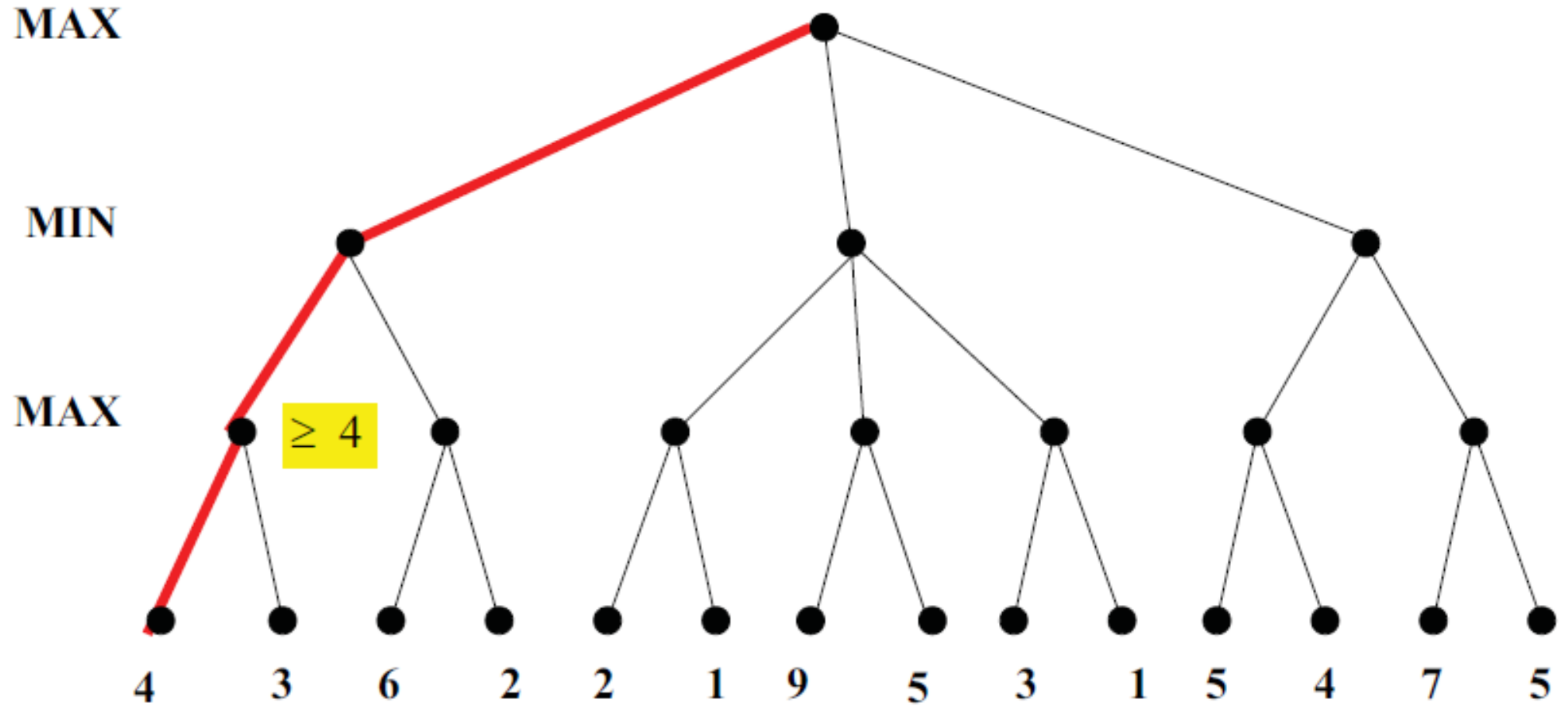
Alpha Beta Pruning

39

Alpha beta pruning. Example

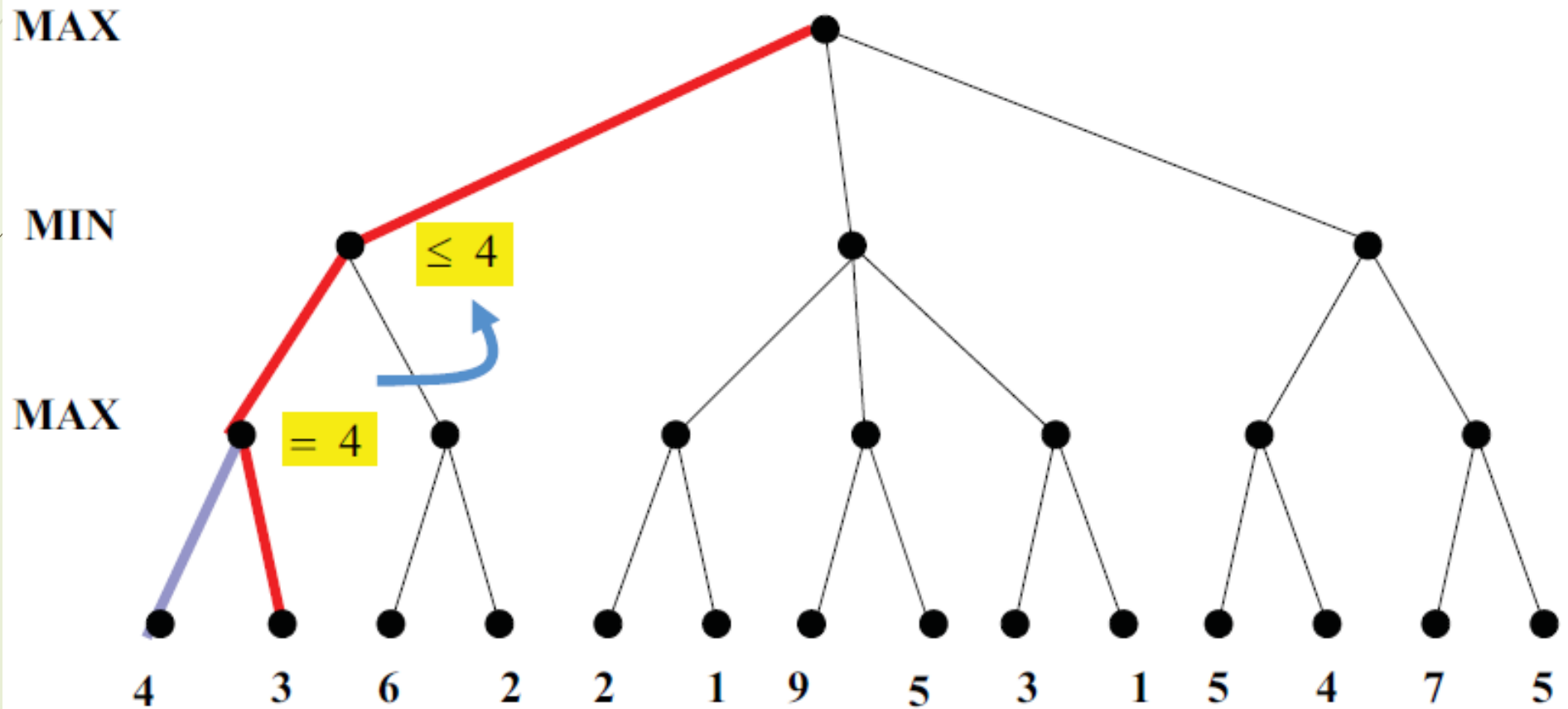


Alpha beta pruning. Example

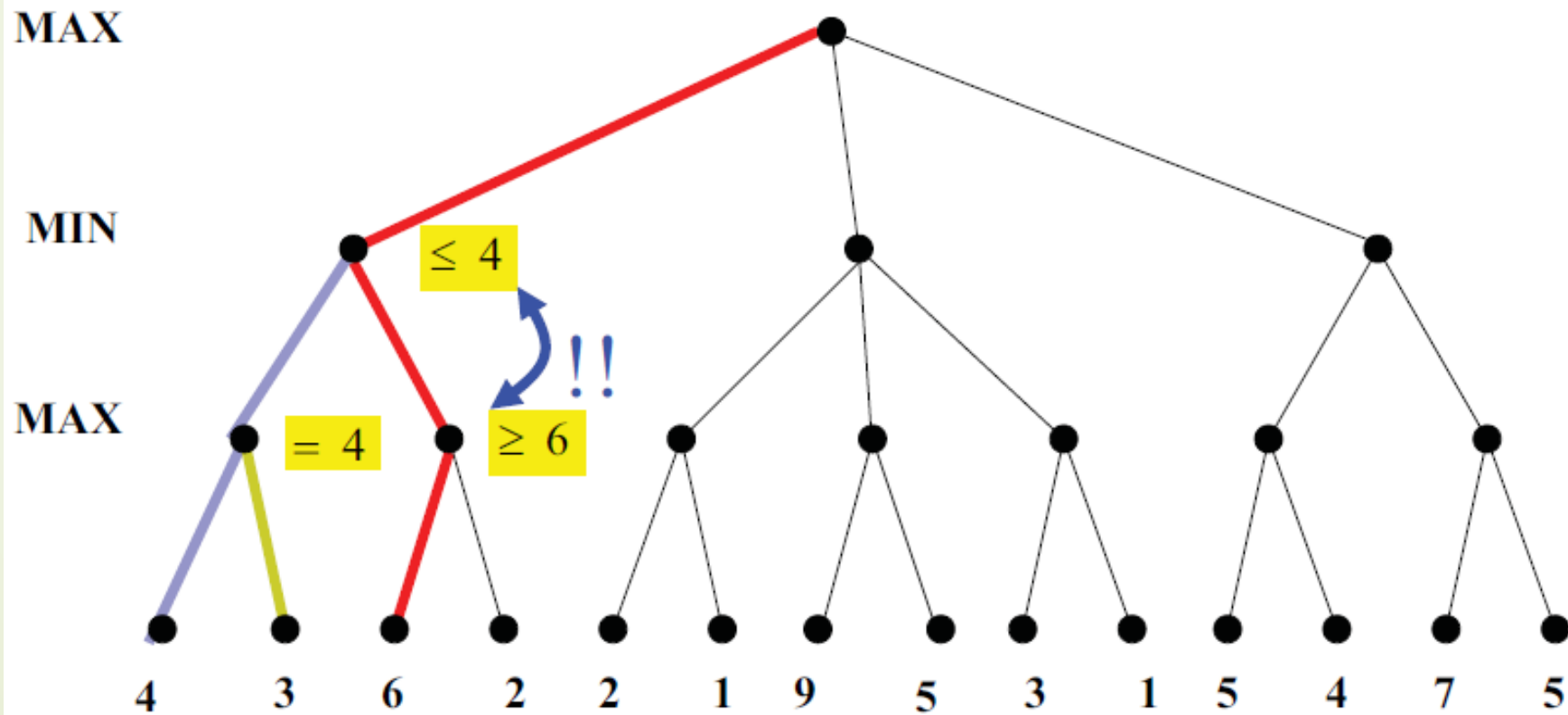


41

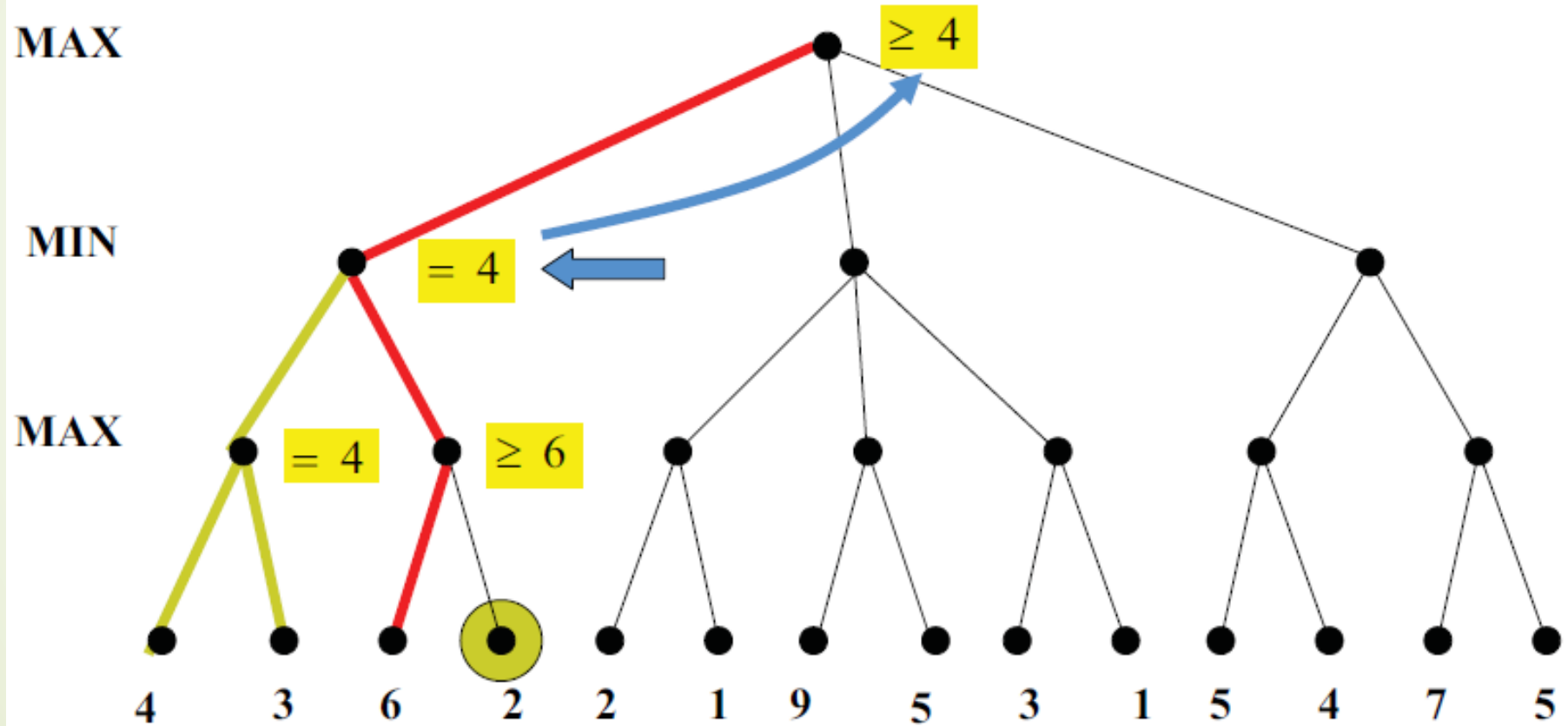
Alpha beta pruning. Example



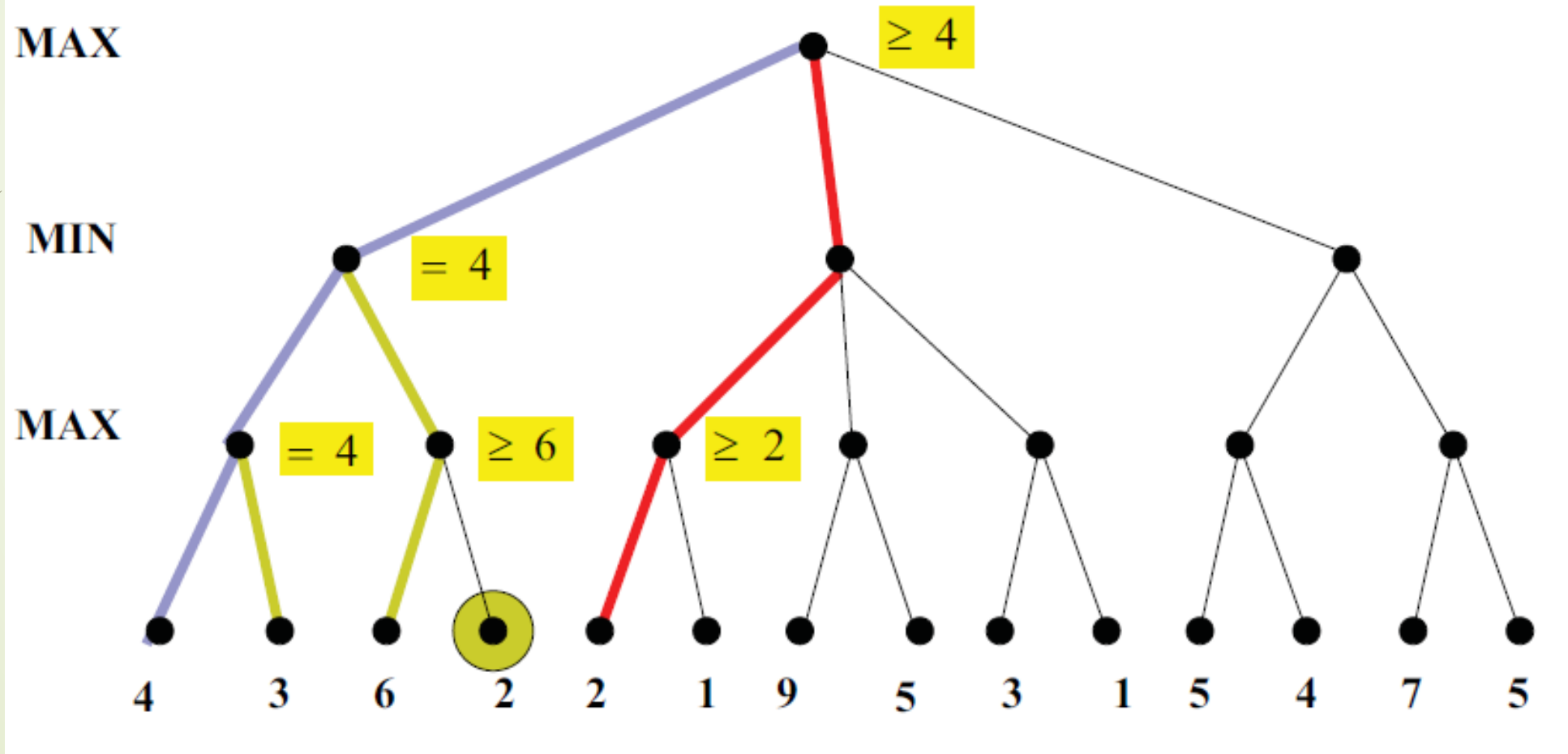
Alpha beta pruning. Example



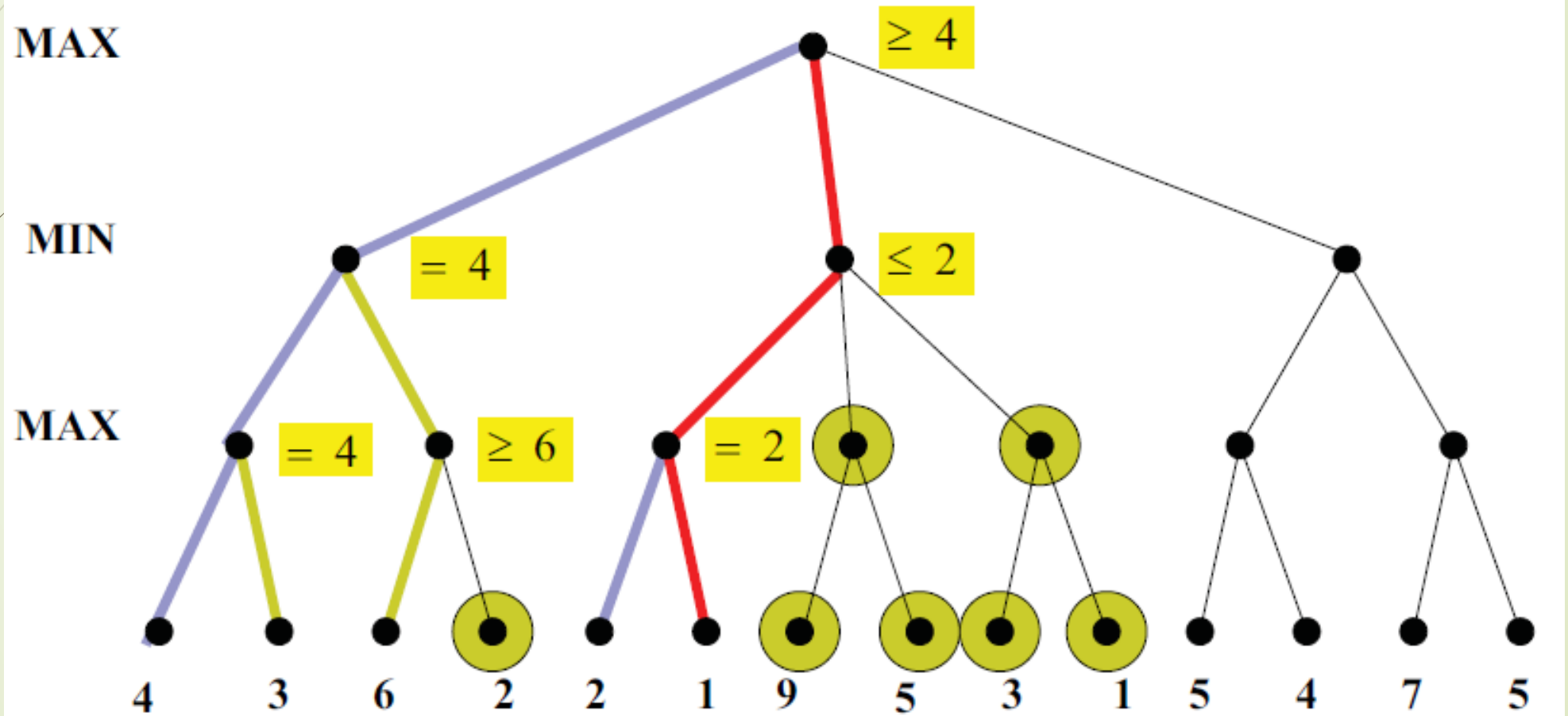
Alpha beta pruning. Example



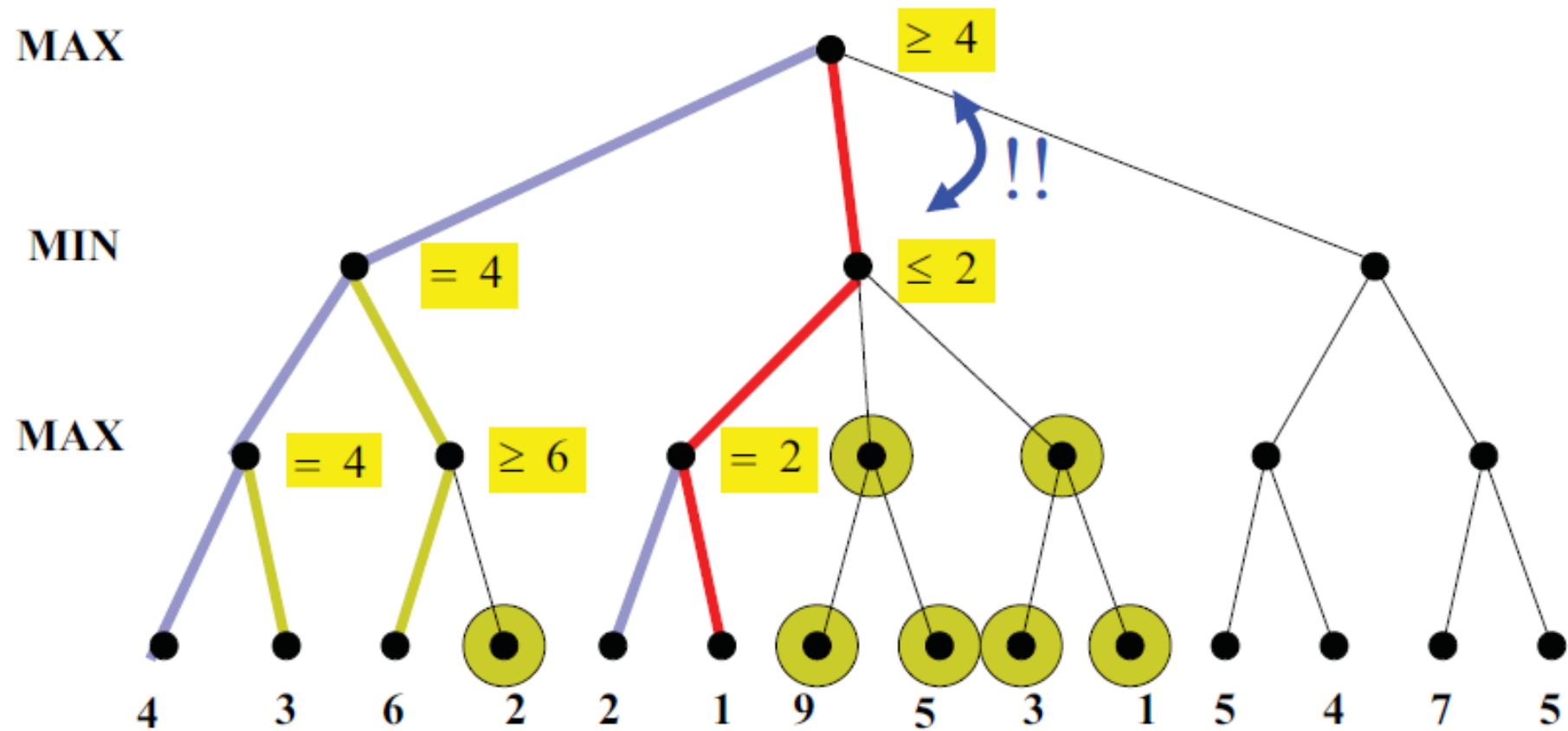
Alpha beta pruning. Example



Alpha beta pruning. Example

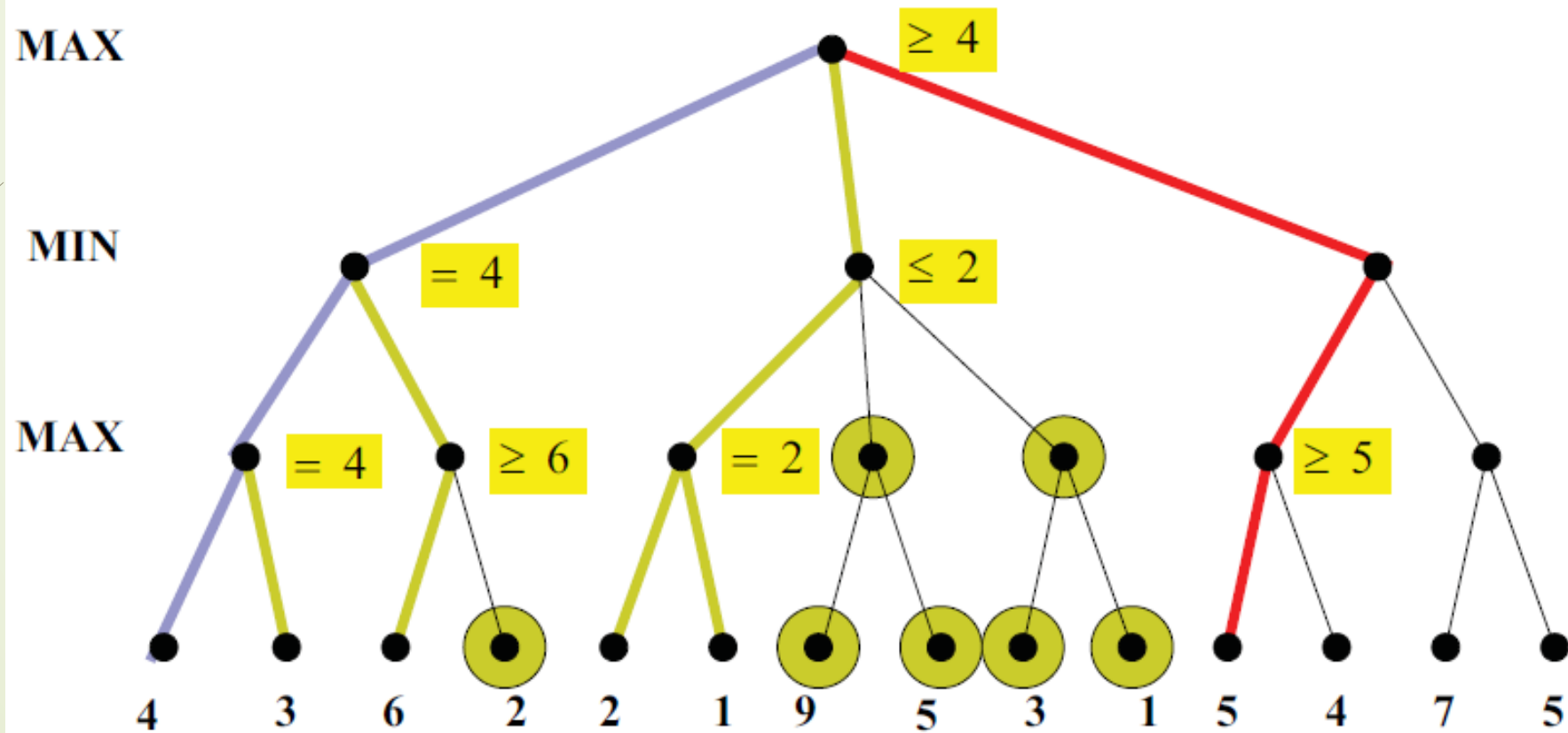


Alpha beta pruning. Example

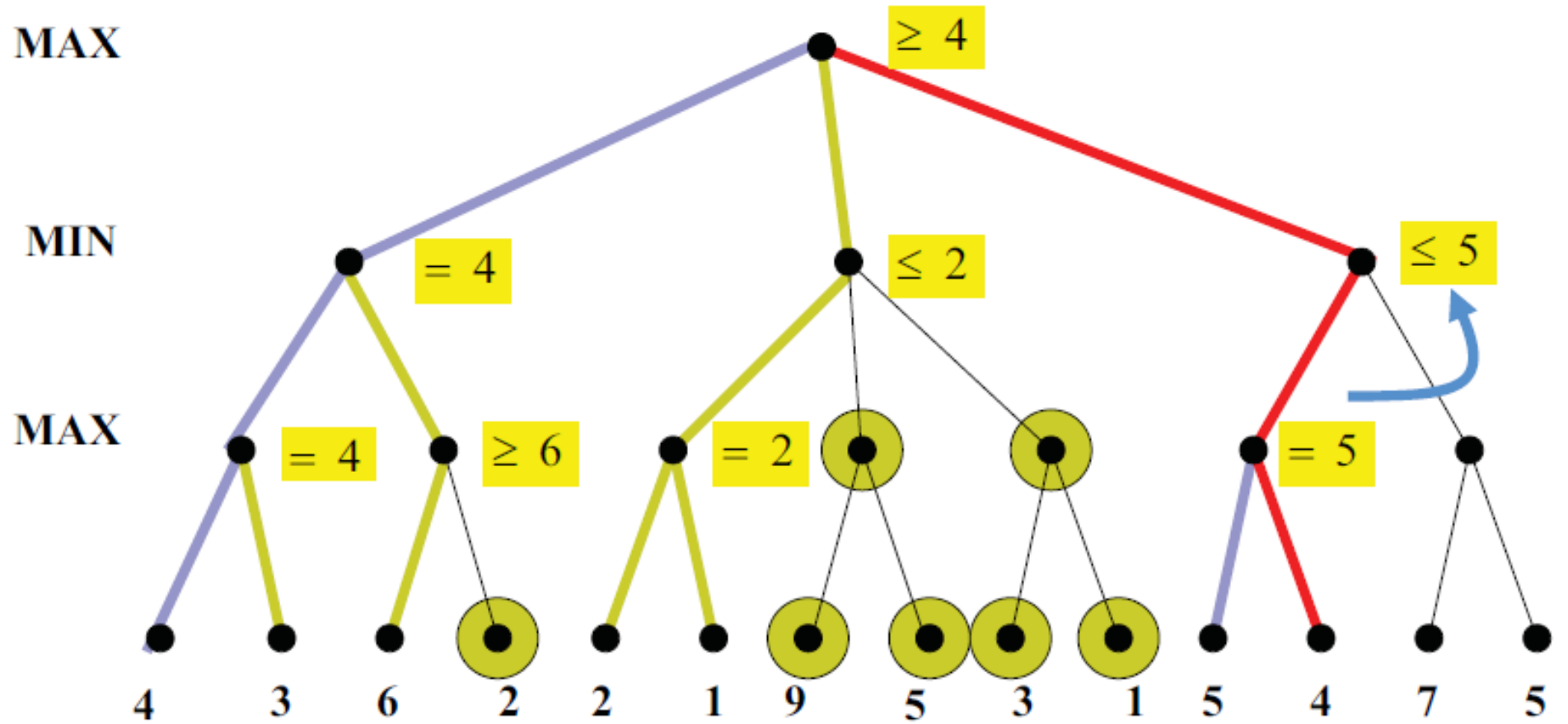


47

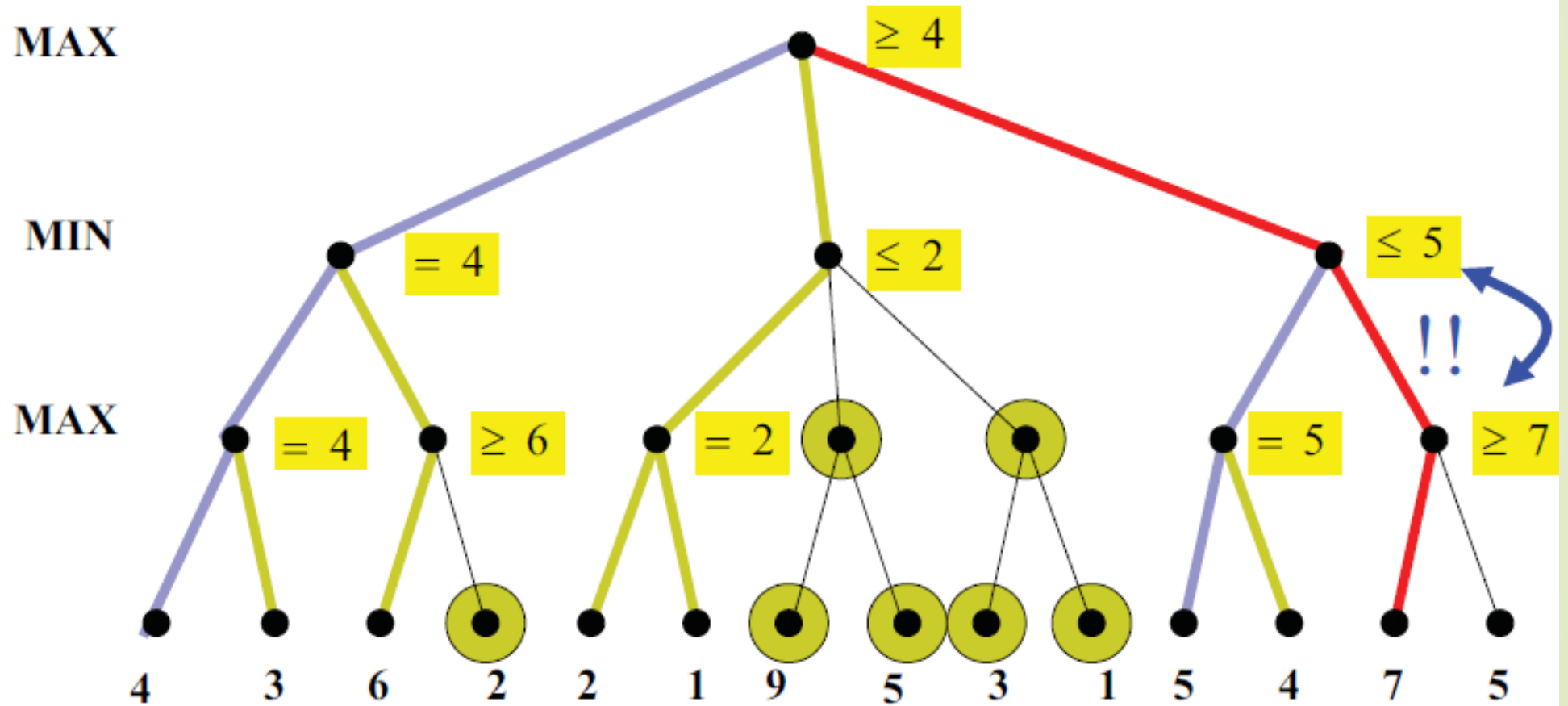
Alpha beta pruning. Example



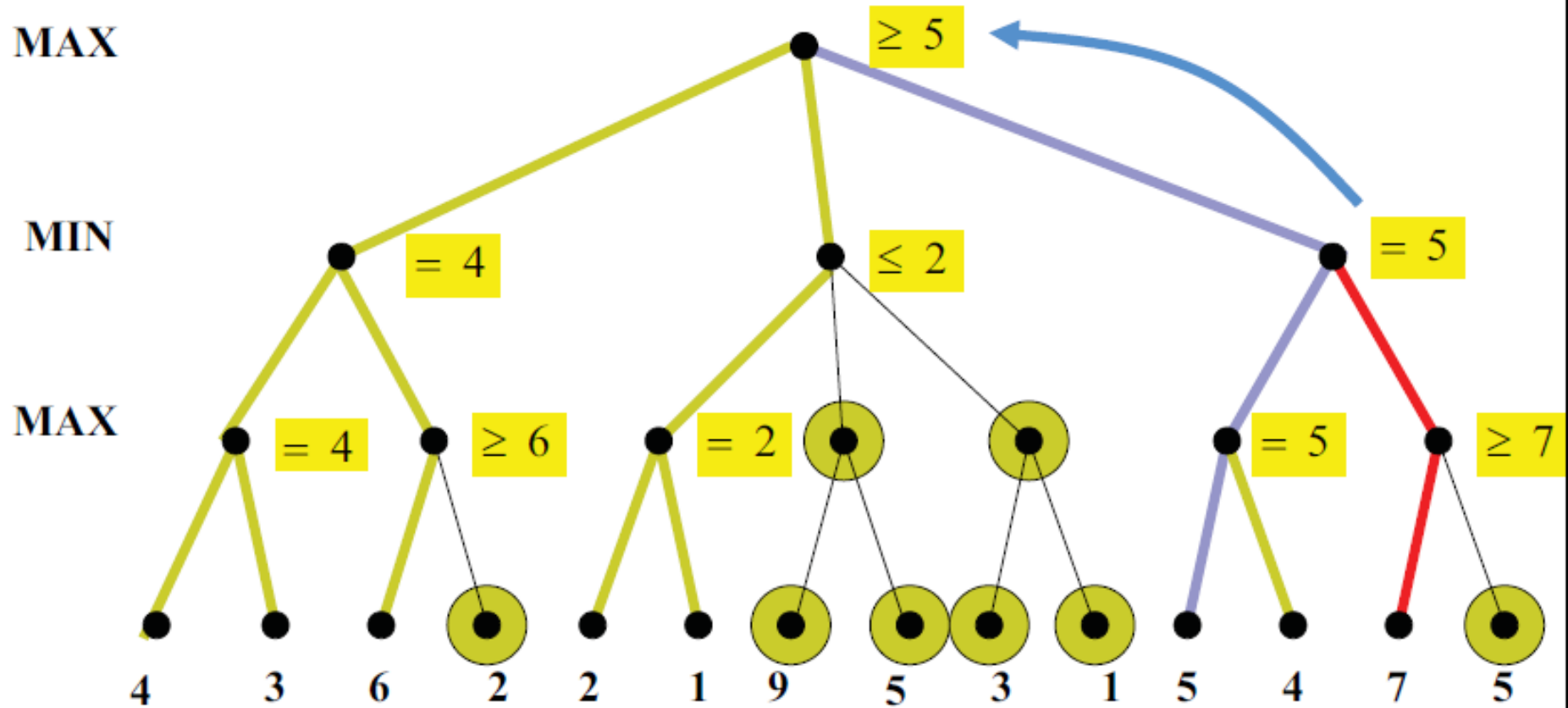
Alpha beta pruning. Example



Alpha beta pruning. Example



Alpha beta pruning. Example



Alpha beta pruning. Example

