Chapter2

Problems, Problem Spaces and search

(Cont…)

Problem Characteristics

- **In order to choose the most appropriate method for a particular problem, it is necessary to analyse the problem along several key dimensions:**
    - **Is the problem decomposable?**
    - **Can solution steps be ignored or undone?**
    - **Is the problem universe predictable?**
    - **Is a good solution absolute or relative?**
    - **Is the solution a state or a path?**
    - **What is the role of knowledge?**
    - **Does the task require interaction with person?**

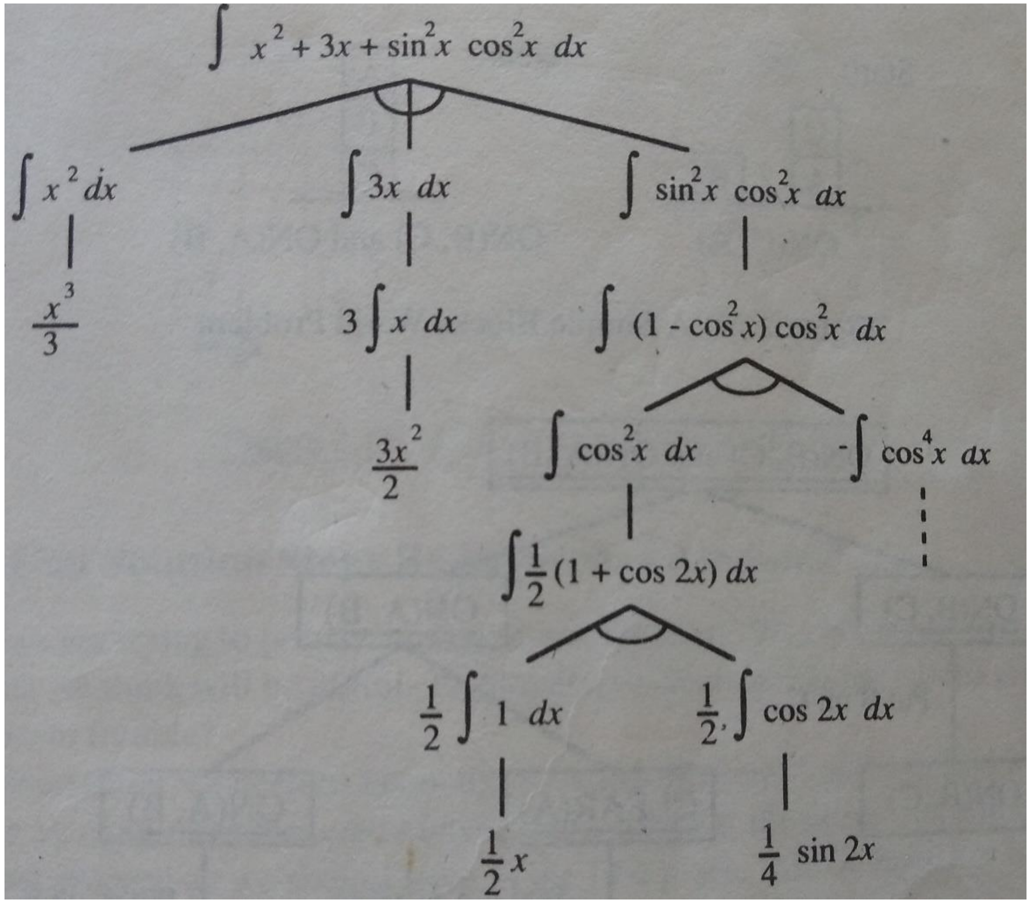
Is the problem decomposable?

- ❑ Whether the problem can be decomposed into smaller problems?
- ❑ Using the technique of problem decomposition , we can often solve very large problems easily

Examples:

Problem of computing

the expression



$\int x^2 + 3x + \sin^2 x \cos^2 x \, dx$
$\int x^2 dx$
$\int 3x \, dx$
$\int \sin^2 x \cos^2 x \, dx$
$\frac{x^3}{3}$
$3 \int x \, dx$
$\int (1 - \cos^2 x) \cos^2 x \, dx$
$\frac{3x^2}{2}$
$\int \cos^2 x \, dx$
$-\int \cos^4 x \, dx$
$\int \frac{1}{2}(1 + \cos 2x) \, dx$
$\frac{1}{2} \int 1 \, dx$
$\frac{1}{2} \int \cos 2x \, dx$
$\frac{1}{2} x$
$\frac{1}{4} \sin 2x$

Is the problem decomposable?

- **Blocks world problem**

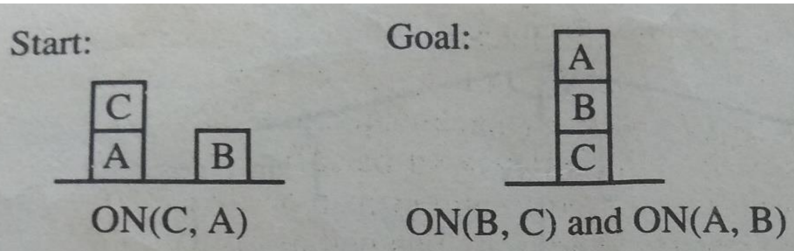
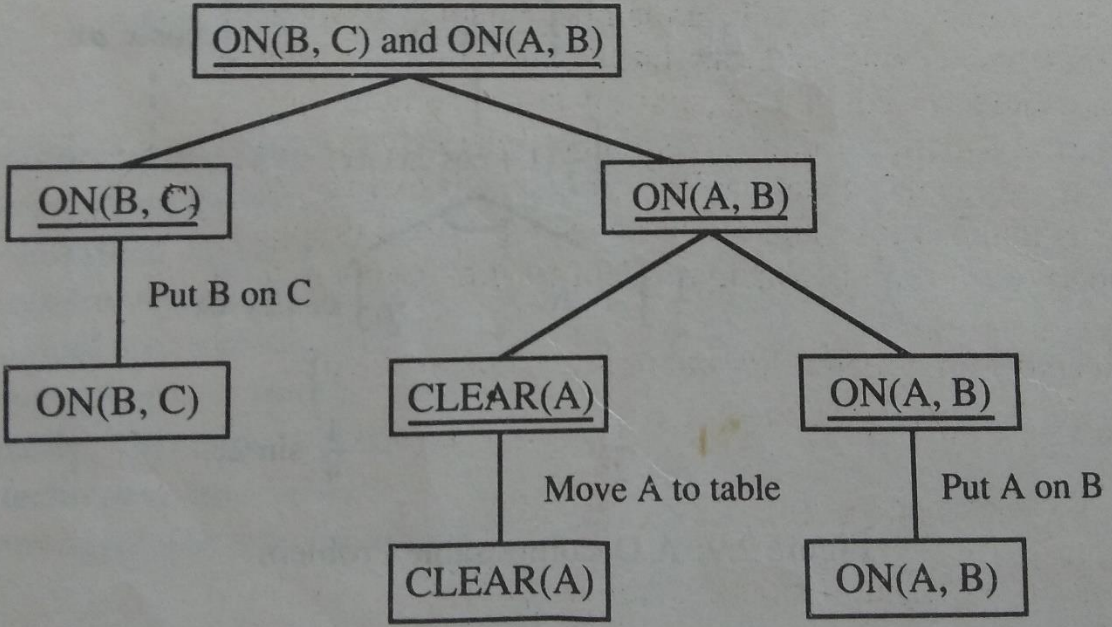


Figure 2.10: A Simple Blocks World Problem



Is the problem decomposable?

Assume that the following operators are available

1. CLEAR(x)[block x has nothing on it]$\rightarrow$ON(x, Table) [pick up x and put it on the table]
2. CLEAR(x) and CLEAR(y) $\rightarrow$ ON(x ,y) [put x on y]

Can solution steps be ignored or undone?

- Classes of problem
  - Ignorable: In which solution steps can be ignored

    e.g . Theorem proving
  - Recoverable: In which solution steps can be undone

    e.g . 8-puzzle
  - Irrecoverable: In which solution steps cannot be undone

    e.g . Chess

5		8
4	2	1
7	3	6

Start

1	2	3
4	5	6
7	8	

Goal

Can solution steps be ignored or undone?

Recoverability of a problem plays an important role in determining the complexity of control structure necessary for the problem solution

- **Ignorable: Simple control structure that never backtracks**
- **recoverable: Slightly more complicated control structure that allows backtracking in case of mistakes e.g. stack is useful**
- **Irrecoverable: Will need to have control structure that expends deal of effort making each decision since the decision must be final**

Is the universe predicate?

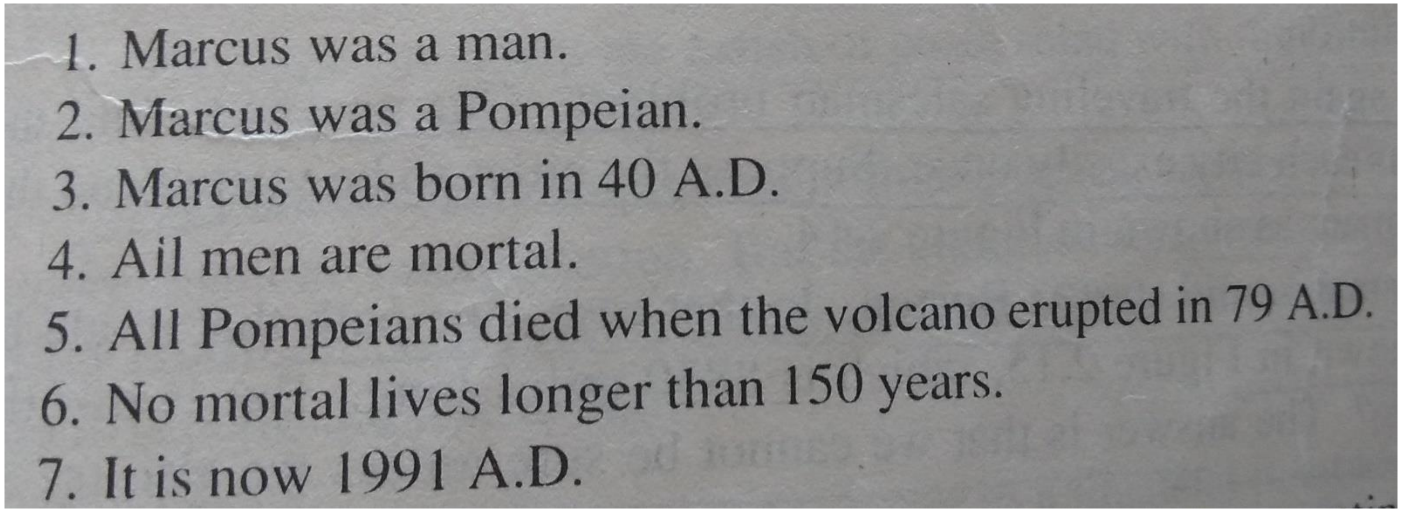
- Certain Outcome (ex: 8-puzzle)
  - One way of describing planning is that it is problem solving without feedback from the environment
  - For solving certain outcome problems, open loop approach will work fine since the result of action can ne predicted perfectly
  - Planning can be used to generate a sequence of operators that is guaranteed to lead to a solution

Is the universe predicate?

- **Uncertain Outcome (ex: Bridge)**
  - **Planning can at best generate a sequence of operators that has a good probability of leading to a solution**
  - **we need to allow for a process of plan revision to take place as the plan is carried out and the necessary feedback is provided**
  - **Providing no guarantee of an actual solution**
  - **Planning is very expensive since the number of solution paths that need to explored increases exponentially with the number of points at which the outcome cannot be predicted**

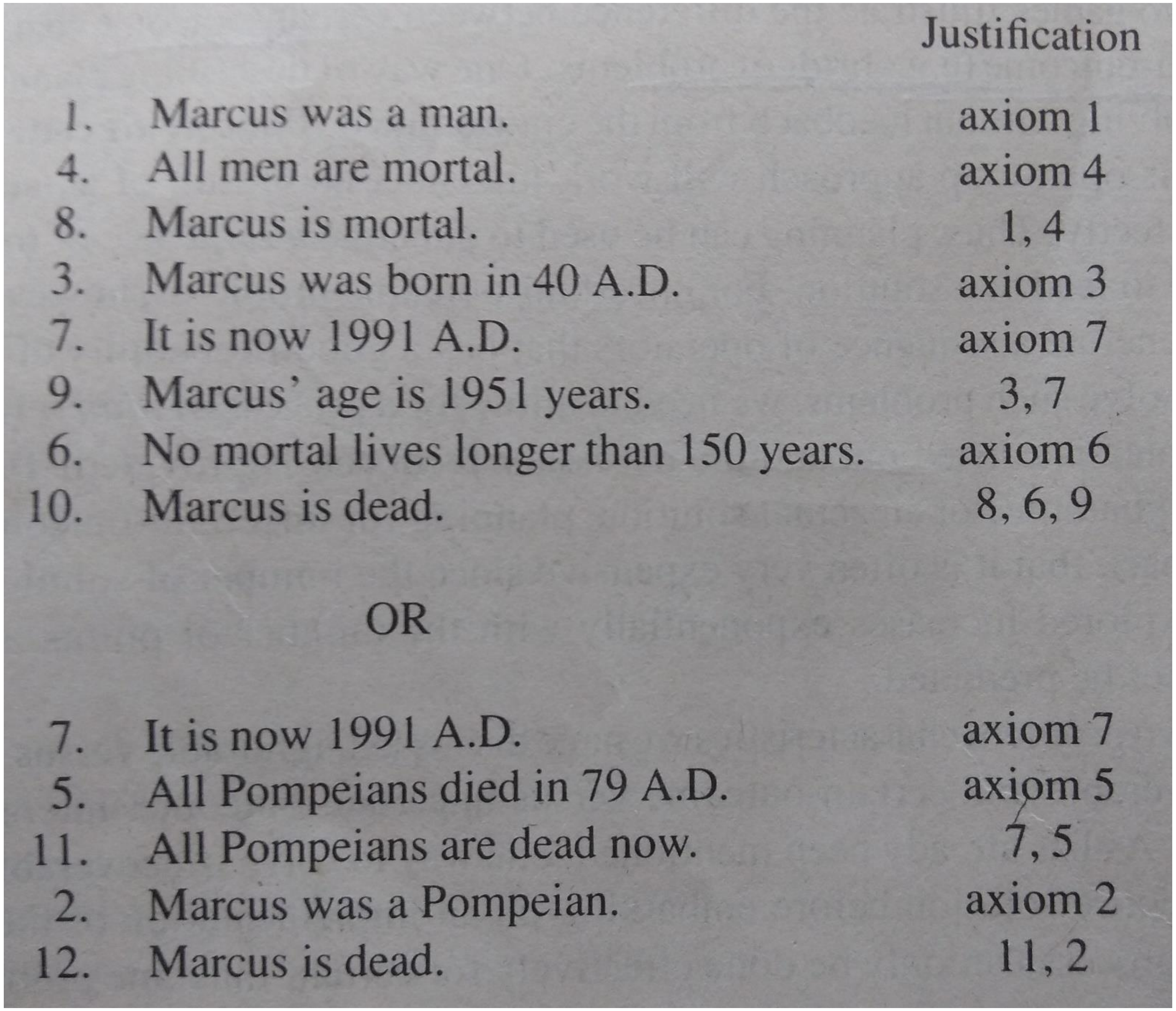
Is a good solution absolute or relative ?

- Consider the problem of answering questions based on a database of simple facts, such as the following:


1. Marcus was a man.
2. Marcus was a Pompeian.
3. Marcus was born in 40 A.D.
4. All men are mortal.
5. All Pompeians died when the volcano erupted in 79 A.D.
6. No mortal lives longer than 150 years.
7. It is now 1991 A.D.

Question: Is Marcus Alive?

Is a good solution absolute or relative ?



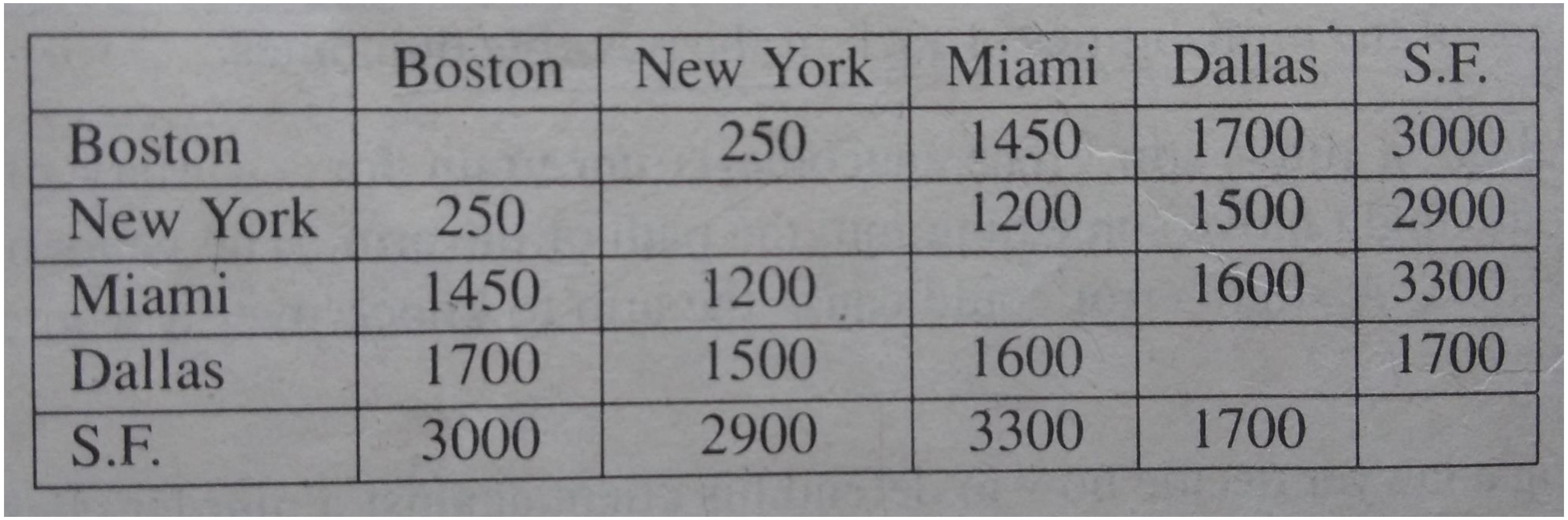
		Justification
1.	Marcus was a man.	axiom 1
4.	All men are mortal.	axiom 4
8.	Marcus is mortal.	1, 4
3.	Marcus was born in 40 A.D.	axiom 3
7.	It is now 1991 A.D.	axiom 7
9.	Marcus' age is 1951 years.	3, 7
6.	No mortal lives longer than 150 years.	axiom 6
10.	Marcus is dead.	8, 6, 9

OR

7.	It is now 1991 A.D.	axiom 7
5.	All Pompeians died in 79 A.D.	axiom 5
11.	All Pompeians are dead now.	7, 5
2.	Marcus was a Pompeian.	axiom 2
12.	Marcus is dead.	11, 2

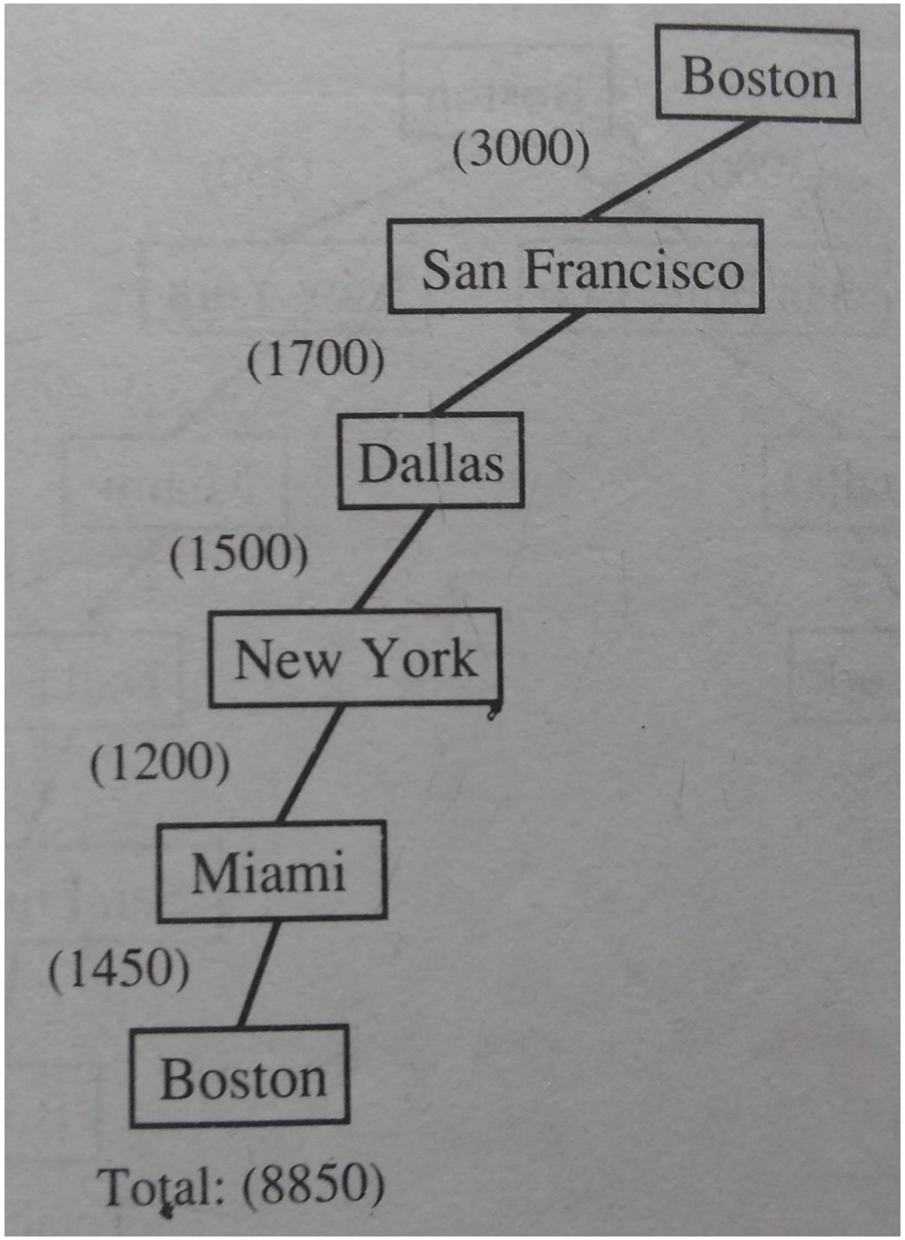
Is a good solution absolute or relative ?

- Consider the Travelling salesman problem. Our goal is to find the shortest route that visits each city exactly once

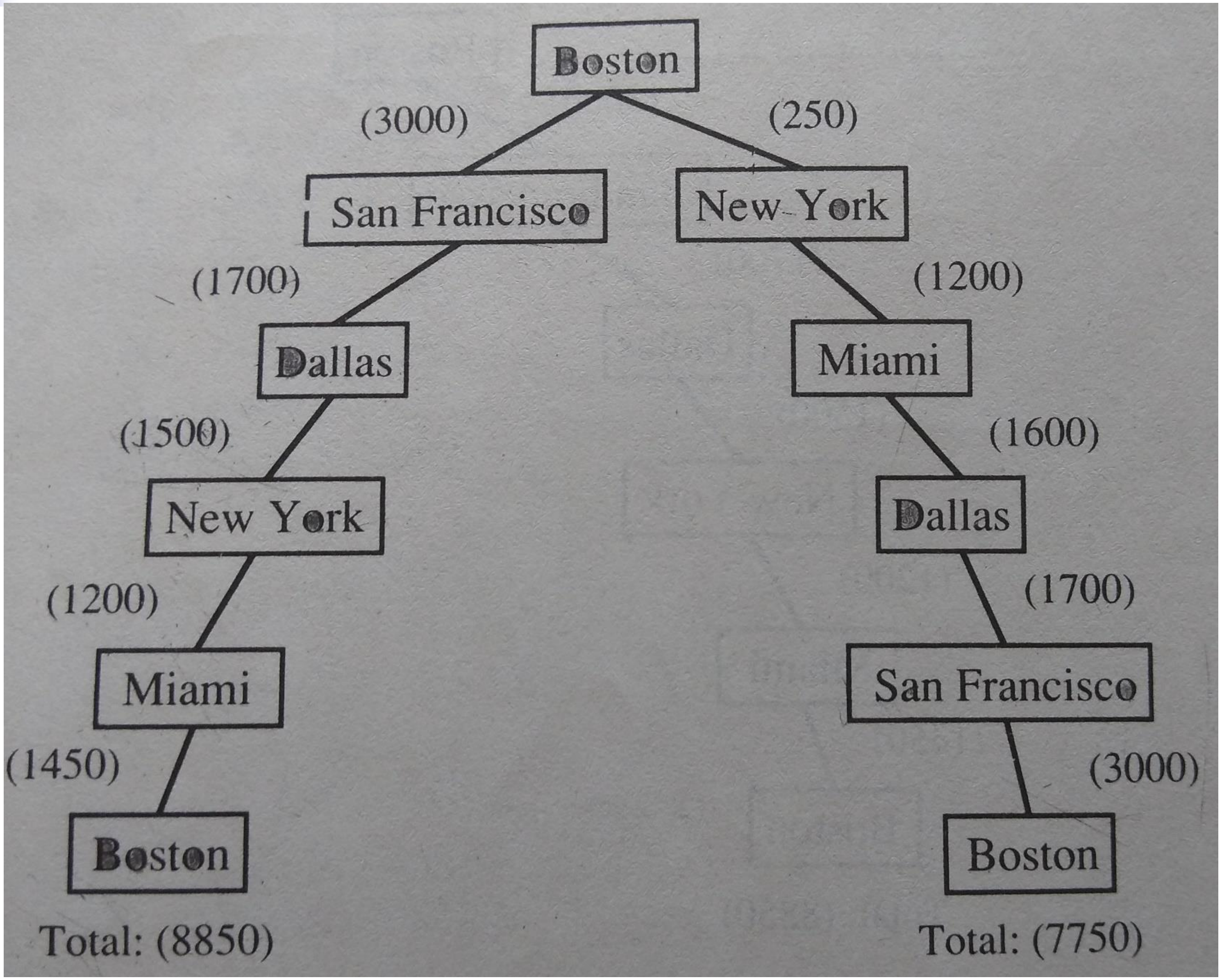


	Boston	New York	Miami	Dallas	S.F.
Boston		250	1450	1700	3000
New York	250		1200	1500	2900
Miami	1450	1200		1600	3300
Dallas	1700	1500	1600		1700
S.F.	3000	2900	3300	1700	

Is a good solution absolute or relative ?



Is a good solution absolute or relative ?



Is a good solution absolute or relative ?

- Any –path problem
  - Any-path problems can often be solved in a reasonable amount of time by using heuristics that suggest good paths to explore
- Best-path problem
  - In Best-path problem, no heuristic that could possibly miss the best solution can be used.so a much more exhaustive search will be performed

Is the solution a state or a path?

- Consider a problem of finding a consistent interpretation for the sentence

  “The bank president ate a dish of pasta salad with fork”

  - We need only final interpretation, not the record of processing by which the interpretation was found

- Consider the Water jug problem

  - A statement of a solution to this problem must be a sequence of operations that produces the final state
  - Here, it is not sufficient to report that we have solved the problem and that final state is (2,0).for this kind of problem, What we really must report is not the final state but the path that we found to that state

What is the role of knowledge?

- **Examples:**
  - **Playing chess: Knowledge required to constrain the search for a solution**
  - **Newspaper story understanding: Lot of knowledge is required even to be able to recognize a solution**

What is the role of knowledge?

- Consider the problem of scanning daily newspapers

" To decide which are supporting the Democrats and which are supporting Republicans in some upcoming election"

We need lot of knowledge to answer such question as:

- The names of the candidates in each party
- The fact that if the major thing you want to see done is have taxes lowered, you are probably supporting the Republicans
- The fact that if the major thing you want to see done is improved education for minority students, you are probably supporting the Democrats
- And so on…

Does the task require interaction with a person?

- Sometimes it is useful to program computers to solve problems in ways that the majority of people would not be able to understand. This is fine if the level of the interaction between the computer and its human users is problem-in solution out
- But increasingly we are building programs that require intermediate interaction with people, both to provide additional input to the program and to provide additional reassurance to the user

Does the task require interaction with a person?

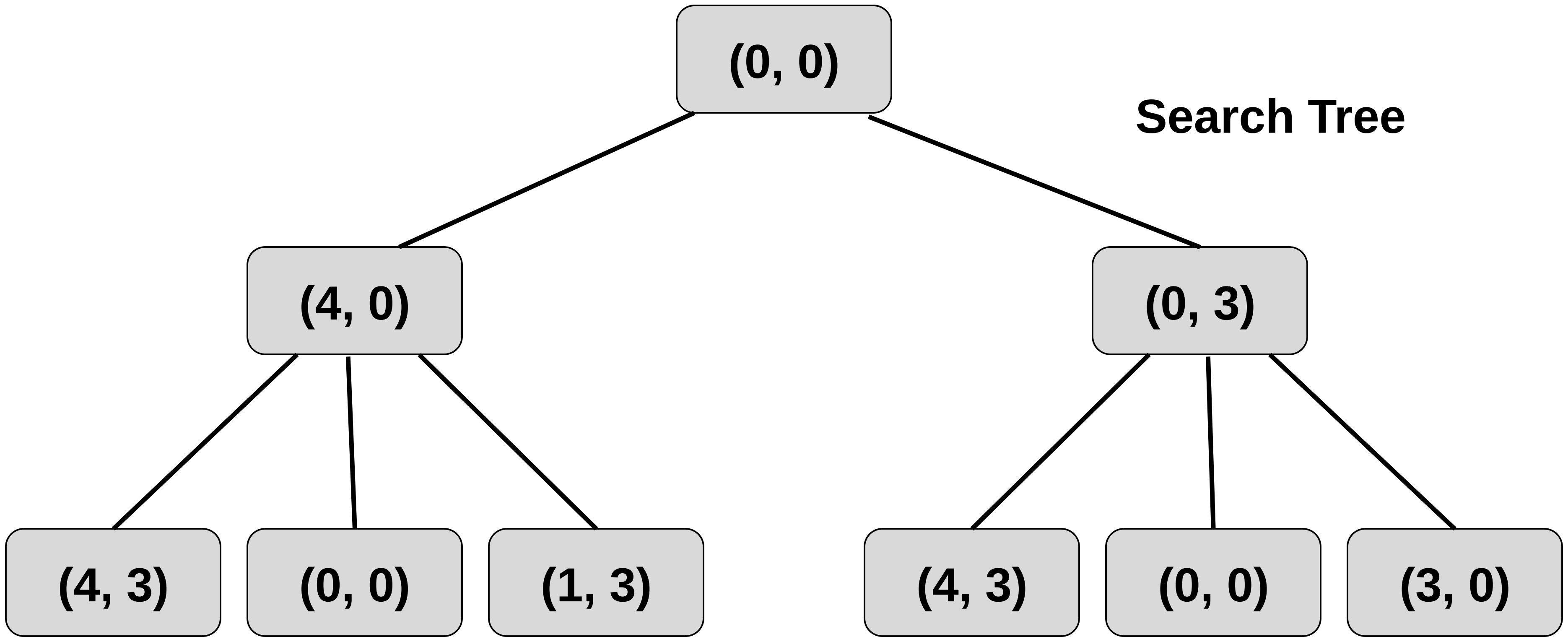
- Types of problems:
  - Solitary: In which the computer is given a problem description and produces an answer with no intermediate communication and with no demand for an explanation of the reasoning process
  - Conversational: In which there is intermediate communication between a person and the computer, either to provide additional assistance to the computer or to provide additional information to the user or both

Issues in the design of search programs

- Every search process can be viewed as a traversal of tree structure

  node: problem state

  edge: rule



Issues in the design of search programs

- Explicit search tree: First create search tree explicitly and then exploration
- Implicit search tree: Generate only part that needs to explore

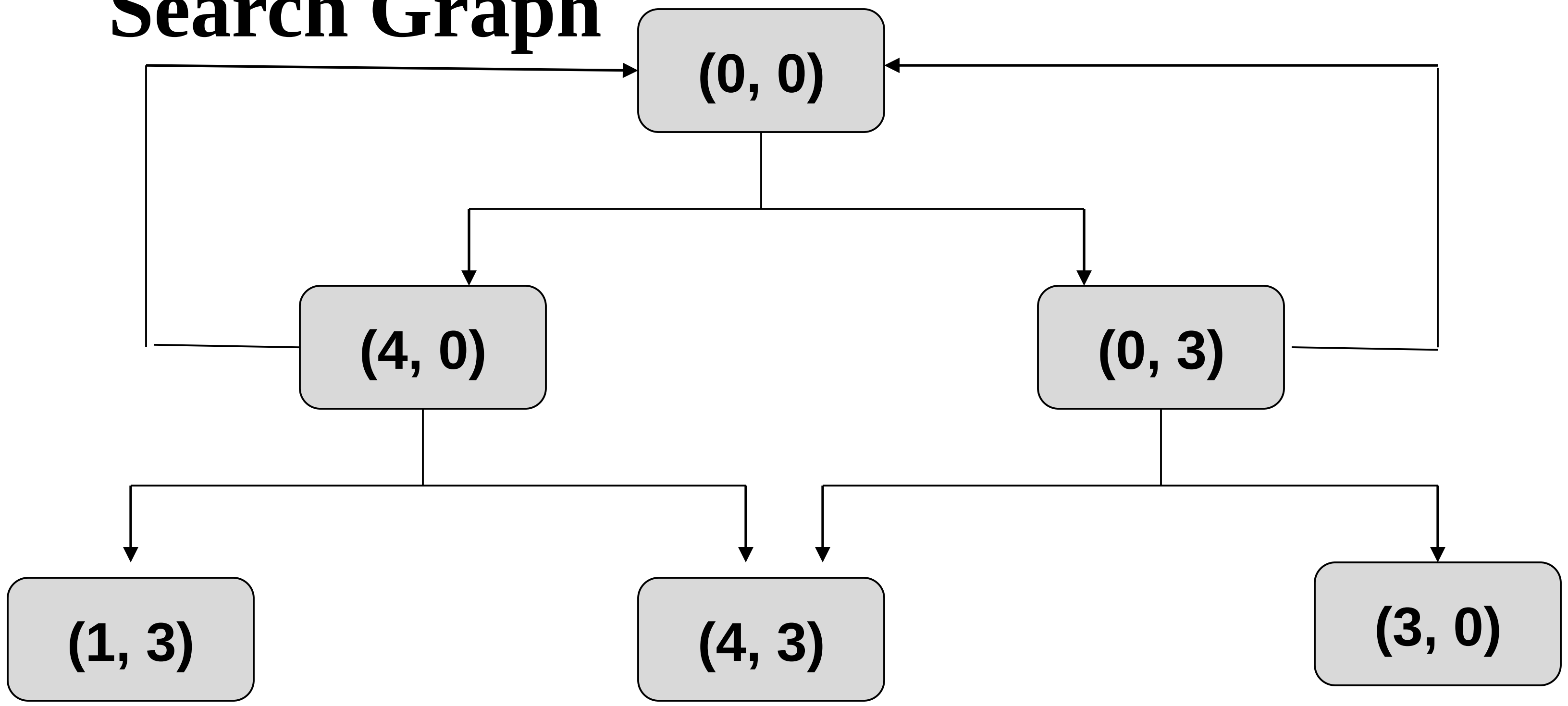
❑ **Issues in search program**

1. The direction in which to conduct search (Forward VS Backward reasoning)
2. How to select applicable rules(matching): Need an efficient procedure for matching rules against states
3. How to represent each node of the search process

Search tree

- Same node generates twice (i.e. (4,3))
- Start state is obtained again that means exploration of path is not leading us to a solution (i.e. (0,0))
- So, eliminate them and try other paths

Search Graph



Algorithm: Check Duplicate Nodes

1. Examine the set of nodes that have been created so far to see if the new node already exists
2. If it does not, simply add it to the graph just as for a tree
3. If it does already exist, then do the following:
   a. Set the node that is being expanded to point to the already existing node corresponding to its successor rather than to the new one. The new one can simply thrown away
   b. If you are keeping track of the best(shortest or otherwise least-cost) path to each node, then check to see if the new path is better or worse than the old one. If worse, do nothing. If better, record the new path as the correct path to use to get to the node and propagate the corresponding change in cost down through successor nodes as necessary

Problems of Search graph

- Additional bookkeeping is necessary to test duplicate nodes
- Cycle may be introduced into the graph search