Searching for Solutions

PoAI

School of Electronic and Computer Engineering
Peking University

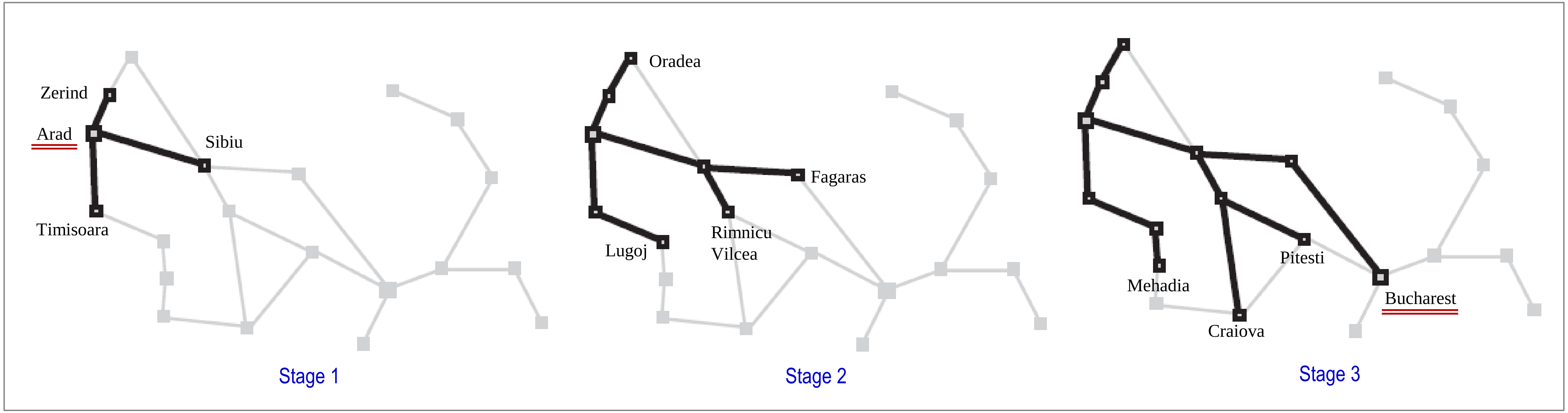
Wang Wenmin

Contents

- □ 3.3.1 Shortest Path Problem by Graph Search
- □ 3.3.2 Shortest Path Problem by Tree Search

Shortest Path Problem by Graph Search 采用图搜索的最短路径问题

□ A sequence of search paths generated by a graph search on the Romania map.
通过图搜索在该罗马尼亚地图上生成一系列搜索路径。



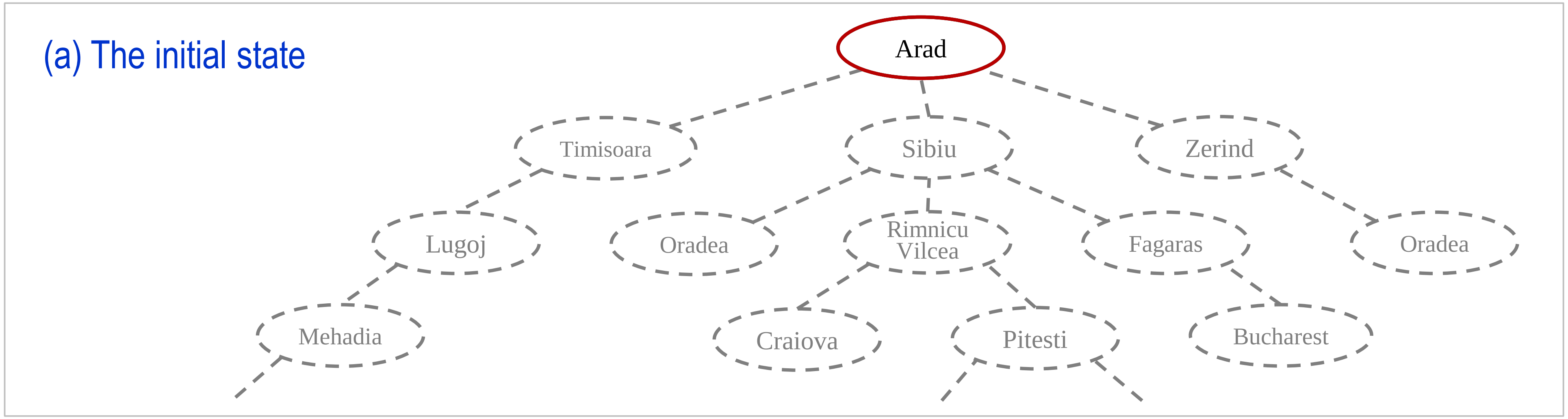
Each path has been extended at each stage by one step. Notice that at 3rd stage, the northernmost city (Oradea) has become a dead end.

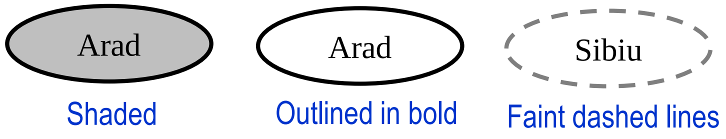
每个路径在每个阶段通过每一步加以扩展扩展。注意在第3阶段，最北部城市(Oradea)已成为死胡同。

Shortest Path Problem by Tree Search 采用树搜索的最短路径问题

- □ Use search trees to find a route Arad to Bucharest.

  用搜索树来寻找一条从Arad到Bucharest的路径。





Shaded: the nodes that have been expanded.

阴影：表示该节点已被扩展。

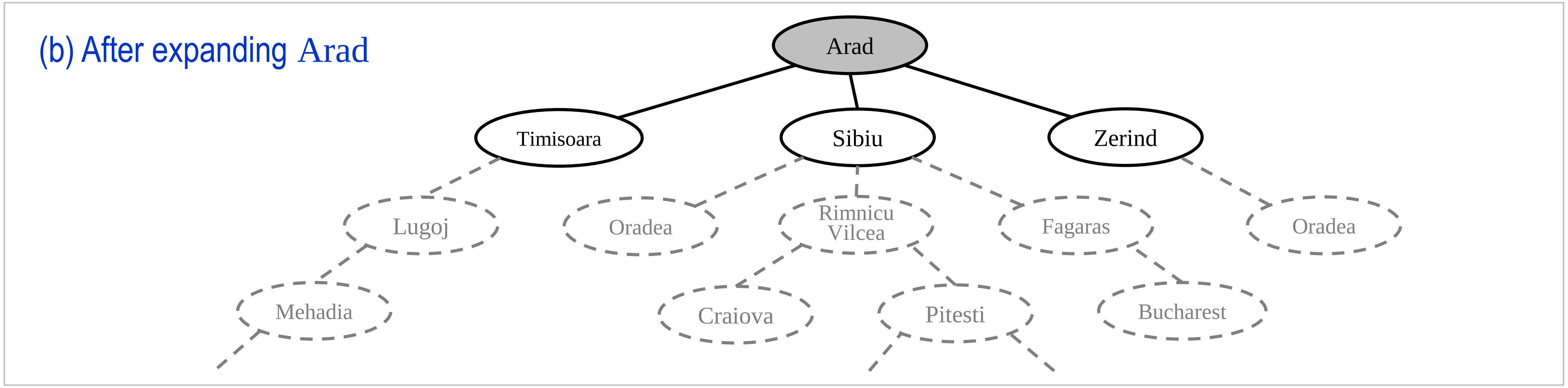
Outlined: the nodes that have been generated but not yet expanded.

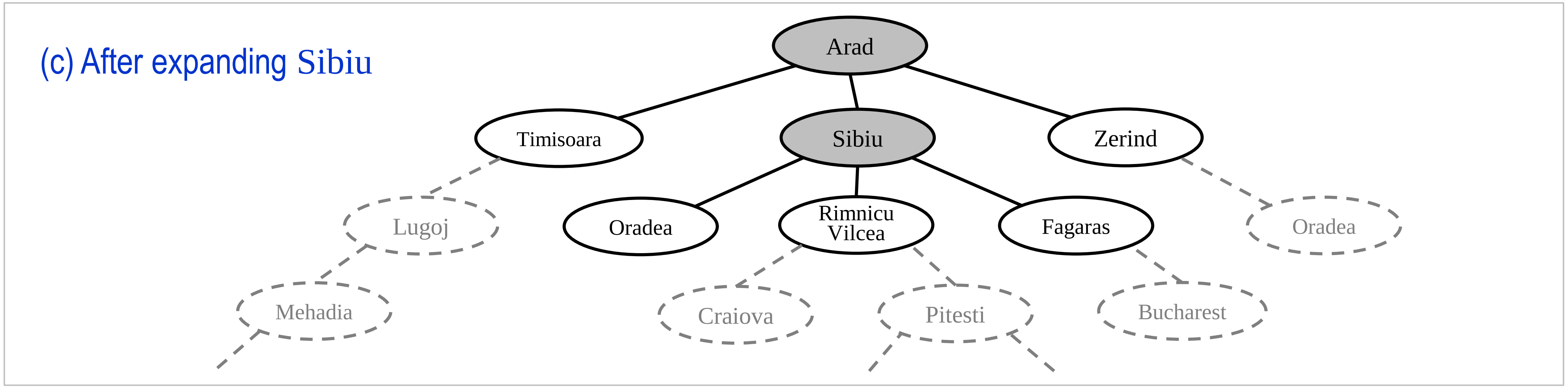
粗实线：表示该节点已被生成，但尚未扩展。

Faint dashed lines: the nodes that have not been generated.

浅虚线：表示该节点尚未生成。

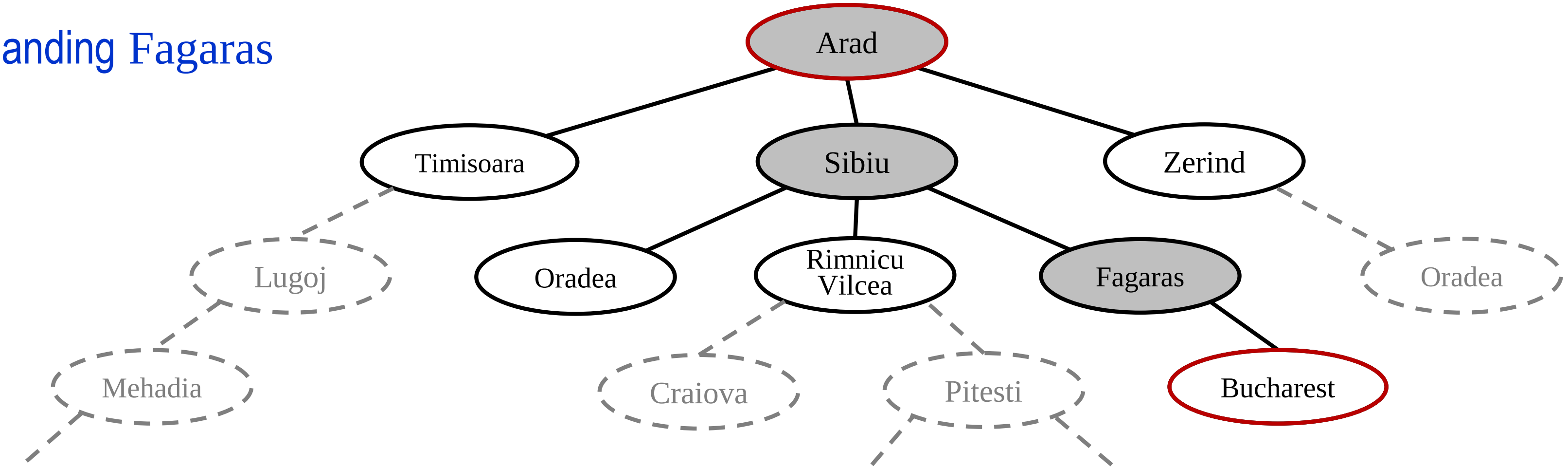
Shortest Path Problem by Tree Search 采用树搜索的最短路径问题

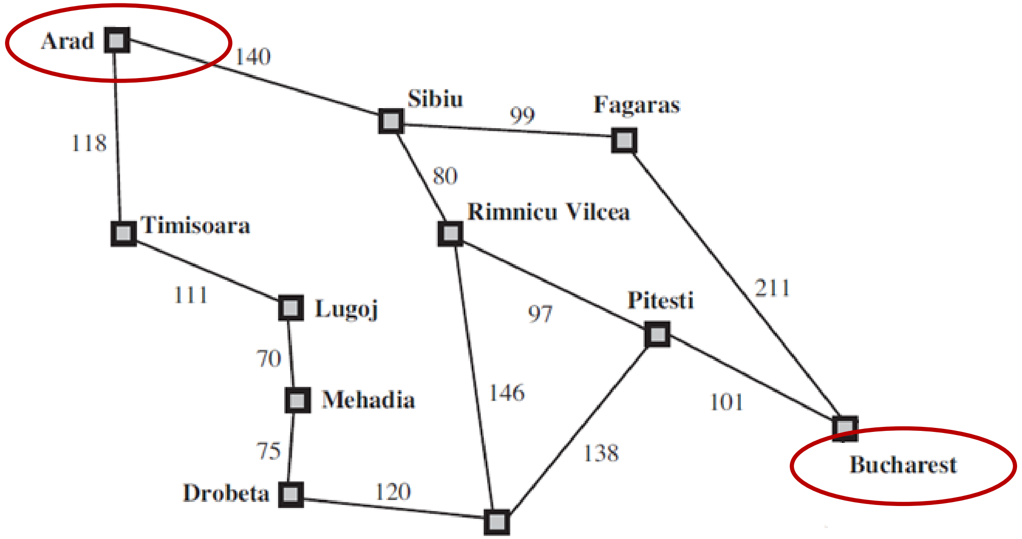




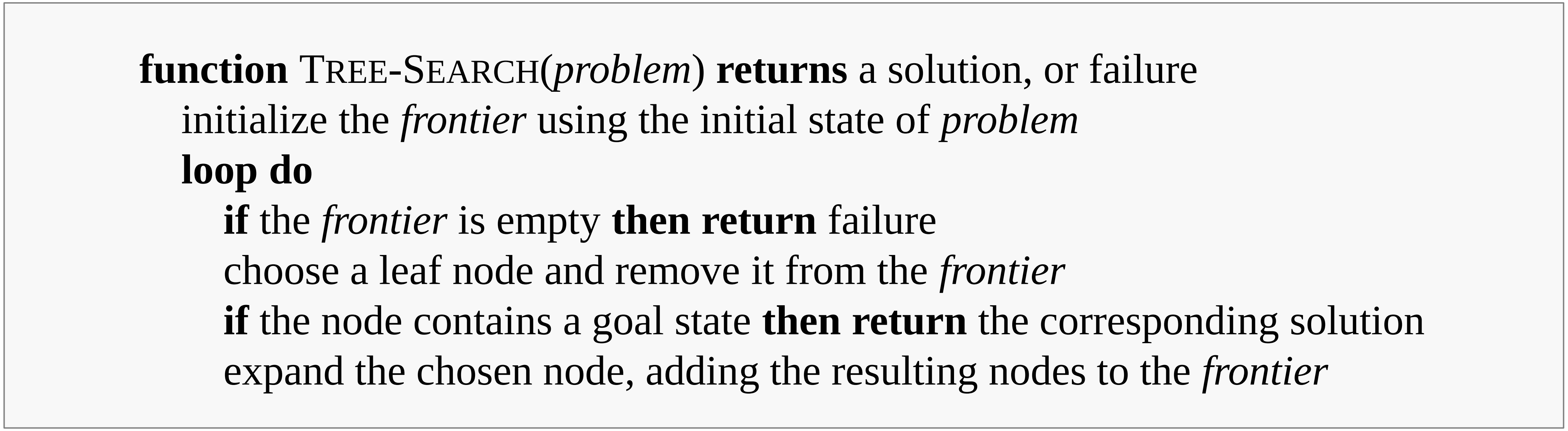
Shortest Path Problem by Tree Search 采用树搜索的最短路径问题

(d) After expanding Fagaras





A General Tree-search Algorithm 一种通用的树搜索算法



```
function TREE-SEARCH(problem) returns a solution, or failure
    initialize the frontier using the initial state of problem
    loop do
        if the frontier is empty then return failure
        choose a leaf node and remove it from the frontier
        if the node contains a goal state then return the corresponding solution
        expand the chosen node, adding the resulting nodes to the frontier
```

The *frontier* (also known as *open list*): an data structure, to store the set of all leaf nodes.

该*frontier*（亦称*open list*）：一种数据结构，用于存储所有的叶节点。

The process of expanding nodes on the *frontier* continues until either a solution is found or there are no more states to expand.

在*frontier*上扩展节点的过程持续进行，直到找到一个解、或没有其它状态可扩展。

A General Graph-search Algorithm 一种通用的图搜索算法

```
function GRAPH-SEARCH (problem) returns a solution, or failure
    initialize the frontier using the initial state of problem
    initialize the explored to be empty
    loop do
        if the frontier is empty then return failure
        choose a leaf node and remove it from the frontier
        if the node contains a goal state then return the corresponding solution
        add the node to the explored
        expand the chosen node, adding the resulting nodes to the frontier
            only if not in the frontier or explored
```

The *explored* (aka *closed list*) is an data structure to remember every expanded node.

该*explored*（亦称*closed list*）：一种数据结构，用于记忆每个扩展节点。

The nodes in the *explored* or the *frontier* can be discarded.

*explored*或*frontier*中的节点可以被丢弃。

Thank you for your attention!

PoAI