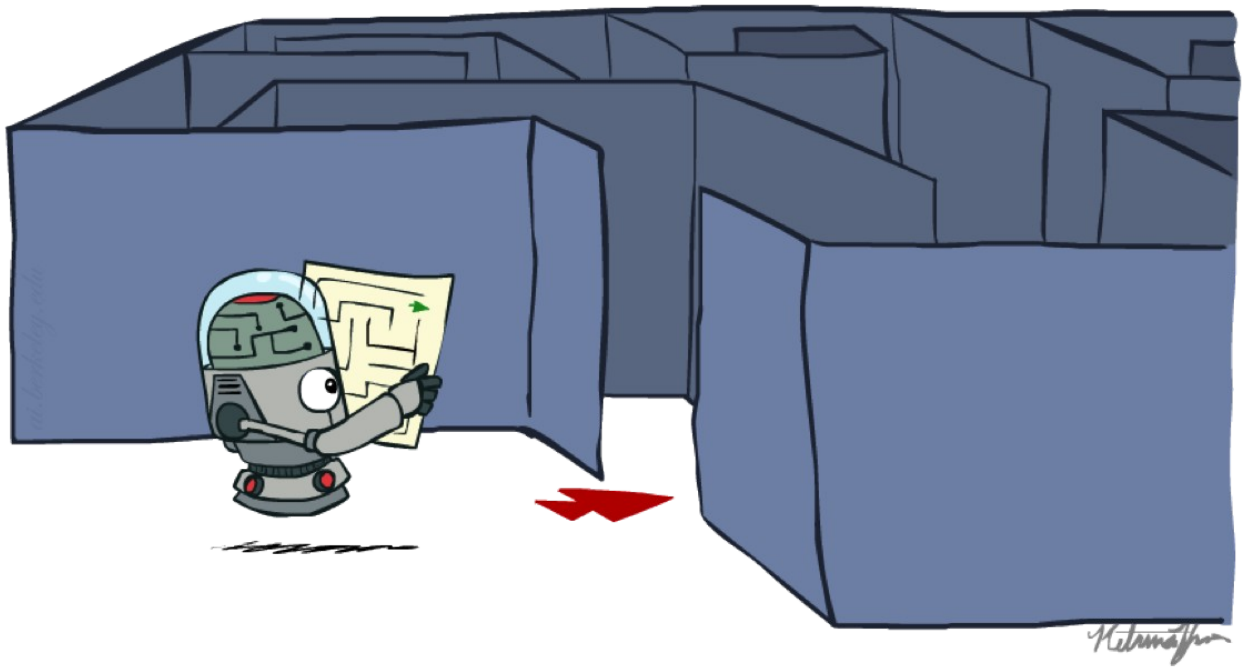
Artificial Intelligence

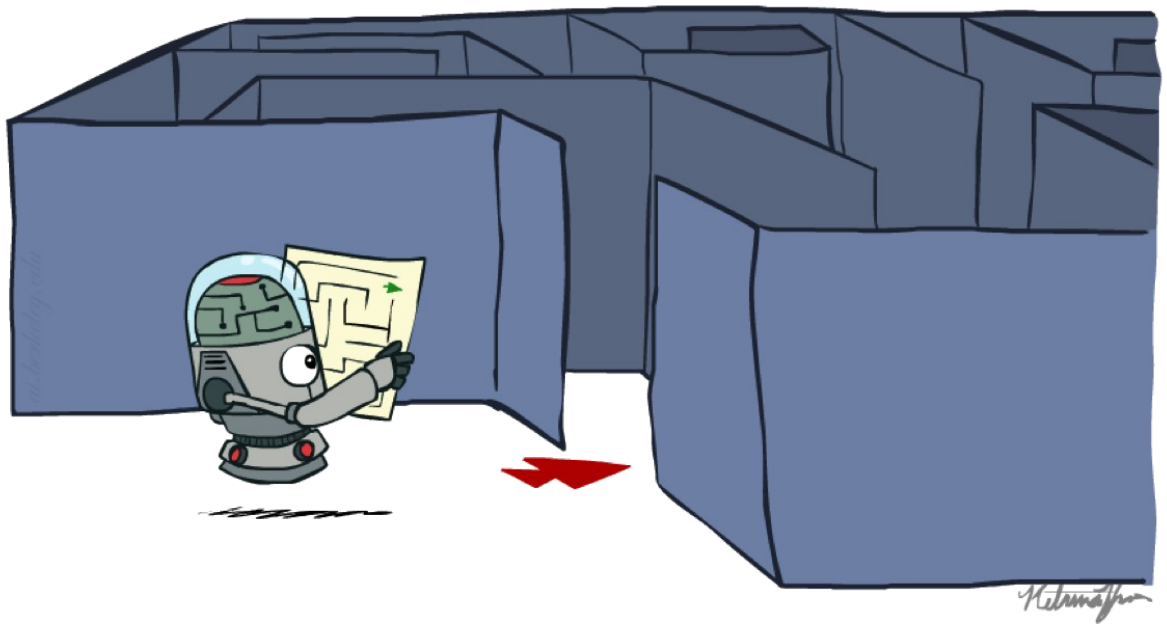
Search



[slides adapted from Dan Klein, Pieter Abbeel]

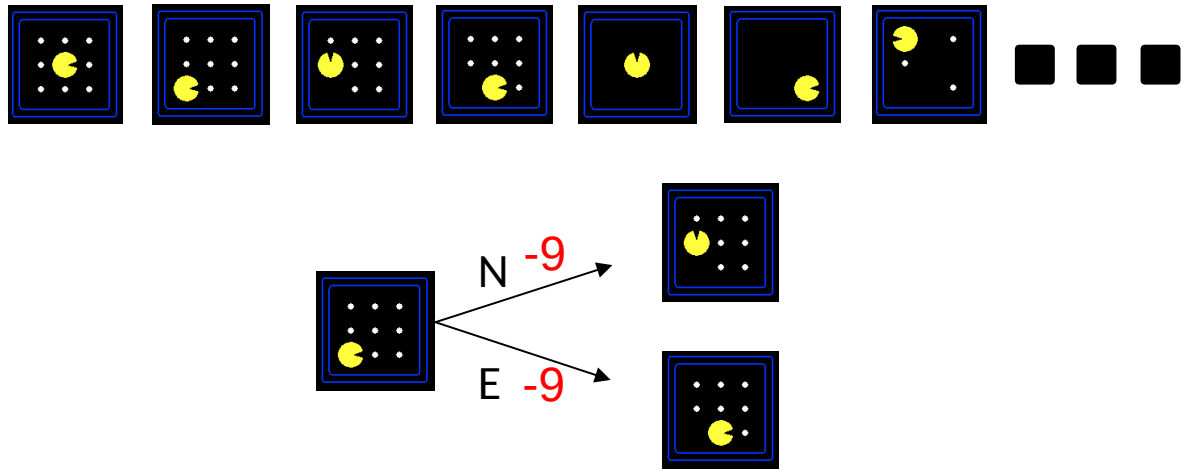
目录

- 3.1 问题求解 Agent
- 3.2 问题形式化
- 3.3 搜索算法
- 3.4 无信息搜索策略



Search Problems

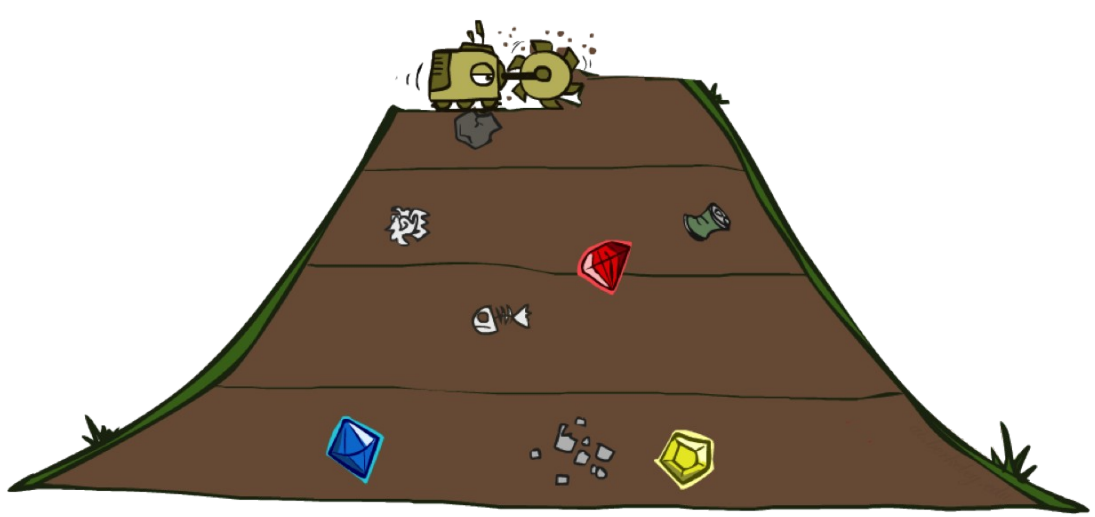
- A search problem consists of:
  - A state space S
  - An initial state s_0
  - Actions $A(s)$ in each state
  - Transition model $Result(s,a)$
  - A goal test $G(s)$
    - s has no dots left
  - Action cost $c(s,a,s')$
    - +1 per step; -10 food; -500 win; +500 die; -200 eat ghost

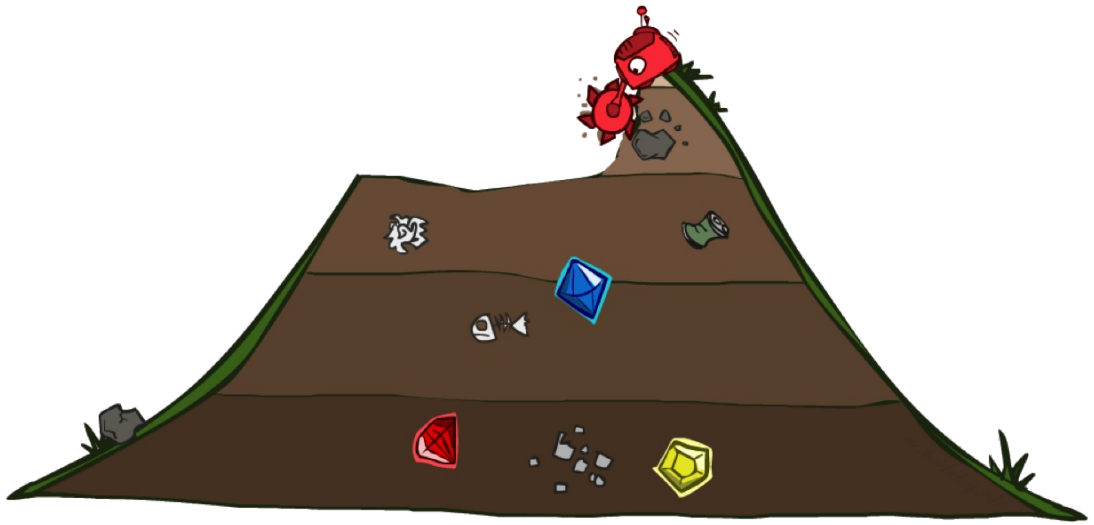


- A solution is an action sequence that reaches a goal state
- An optimal solution has least cost among all solutions

宽度优先搜索 vs. 一致代价搜索

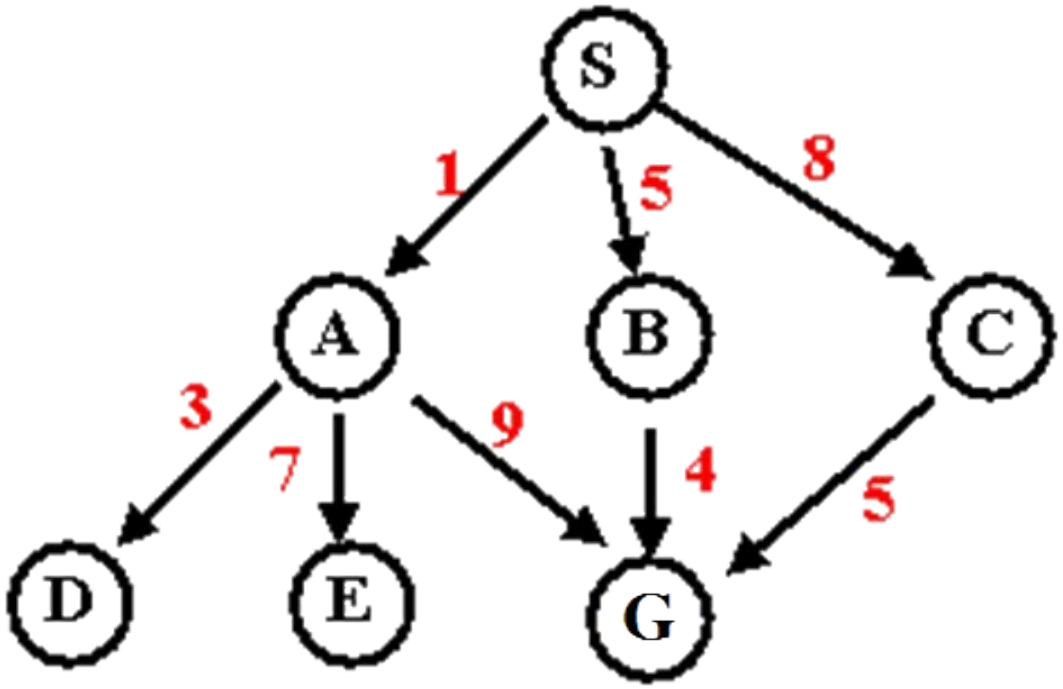
	宽度优先搜索	一致代价搜索
边缘队列	FIFO	优先队列（路径耗散）
最优性	所有边耗散相同时	最优解





课堂练习

宽度优先搜索算法 (BFS) 的边缘队列、搜索序列、解序列，解序列的路径耗散值



宽度优先搜索：

边缘队列：

S

A B C

B C D E G

C D E G

D E G

E G

G

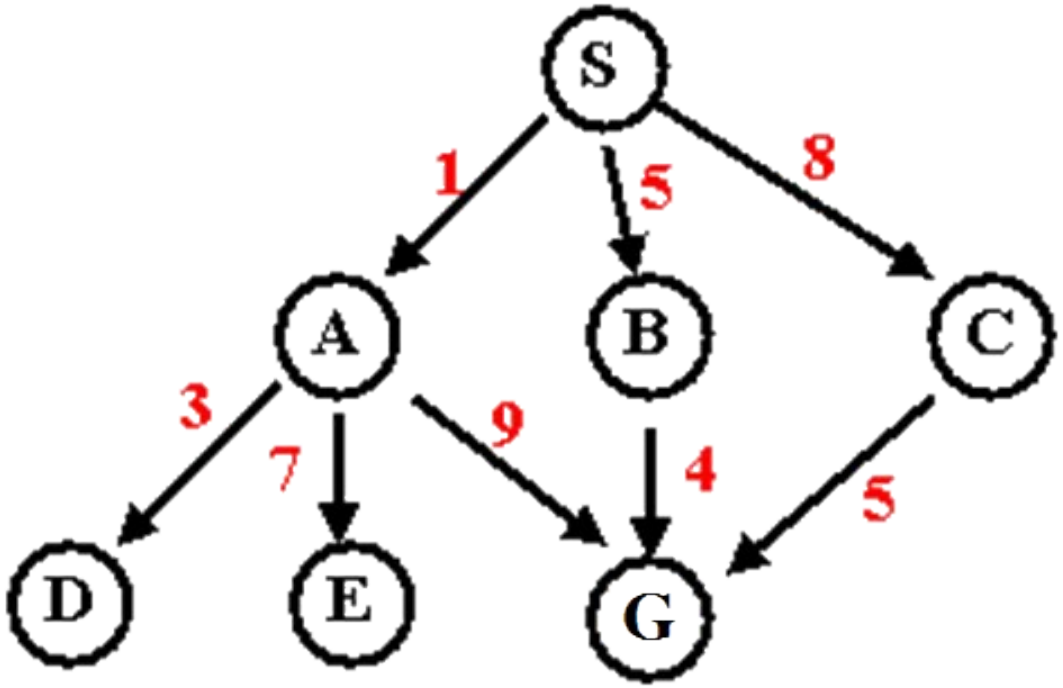
搜索序列：S A B C D E G

解序列： S A G

解序列的路径耗散： 10

课堂练习

一致代价搜索算法 (UCS) 的边缘队列、搜索序列、解序列，解序列的路径耗散值



边缘队列：

S(0)

A(1) B(5) C(8)

D(4) B(5) C(8) E(8) G(10)

B(5) C(8) E(8) G(10)

C(8) E(8) G‘(9) G(10)

E(8) G‘(9) G(10)

G‘(9) G(10)

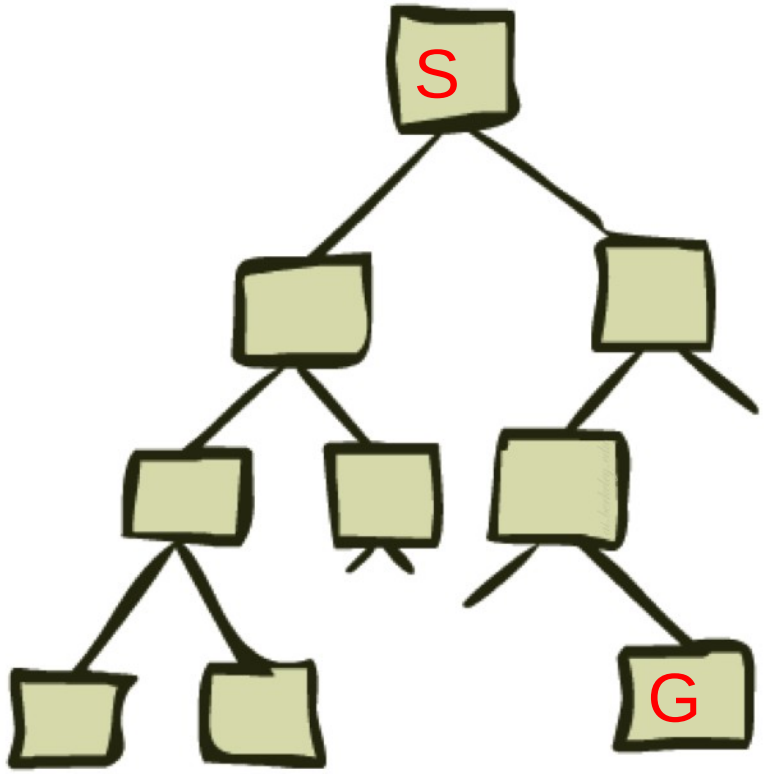
搜索序列：S A D B C E G

解序列： S B G ，路径耗散：

9

3.4 无信息搜索策略

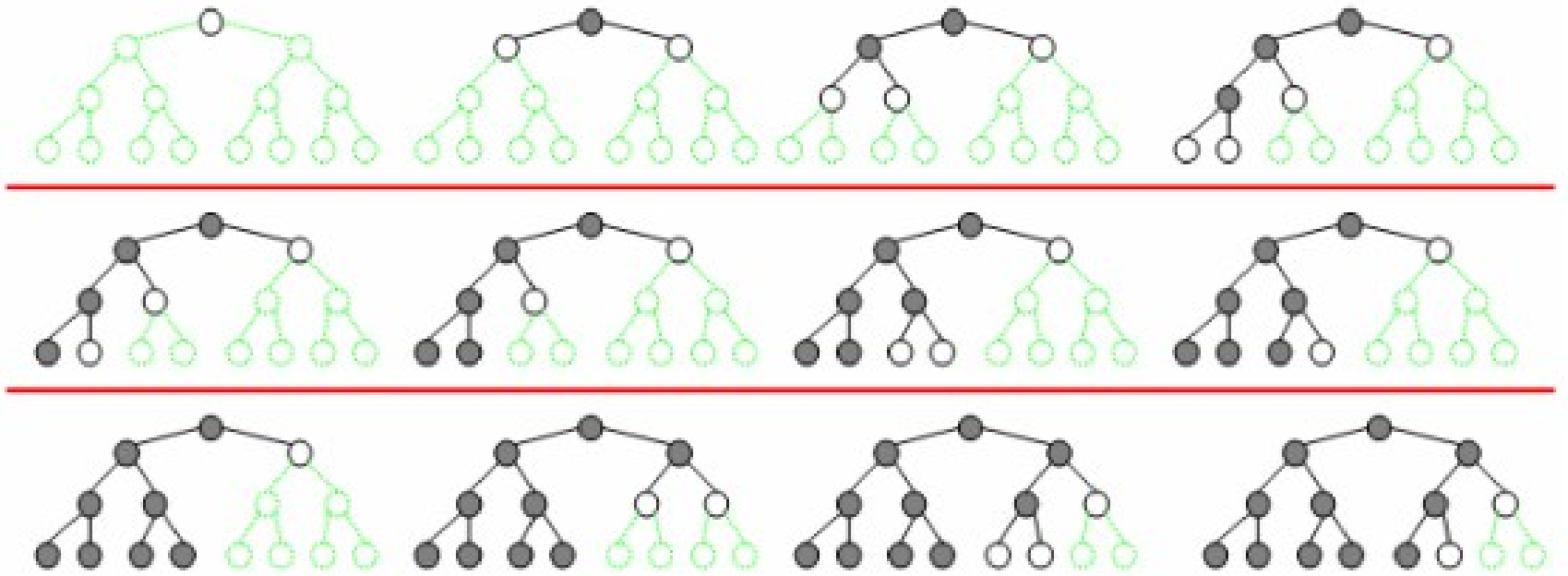
- 5 种无信息搜索（盲目搜索）：
  - 宽度优先搜索 Breadth-first
  - 一致代价搜索 Uniform-cost
  - 深度优先搜索 Depth-first
  - 深度受限搜索 Depth-limited
  - 迭代加深的深度优先搜索 Iterative-deepening



3.4.3 深度优先搜索（DFS）

深度优先搜索（ DFS ）

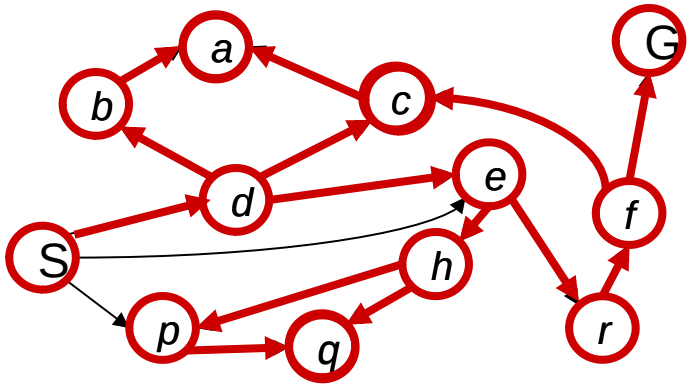
DFS 总是优先扩展当前搜索树中最深的结点。当到达搜索树中边缘无后继的结点时，该算法回溯到次深未扩展的结点继续搜索，直到搜索到目标状态为止。



深度优先搜索（DFS）

搜索策略：先扩展深度最深的结点

实现：边缘是 LIFO 栈



LIFO 栈：

后进先出队列；

最新生成的结点最早被选择被扩展

Fringe:

S

d e p

b c e e p

a c e e p

c e e p

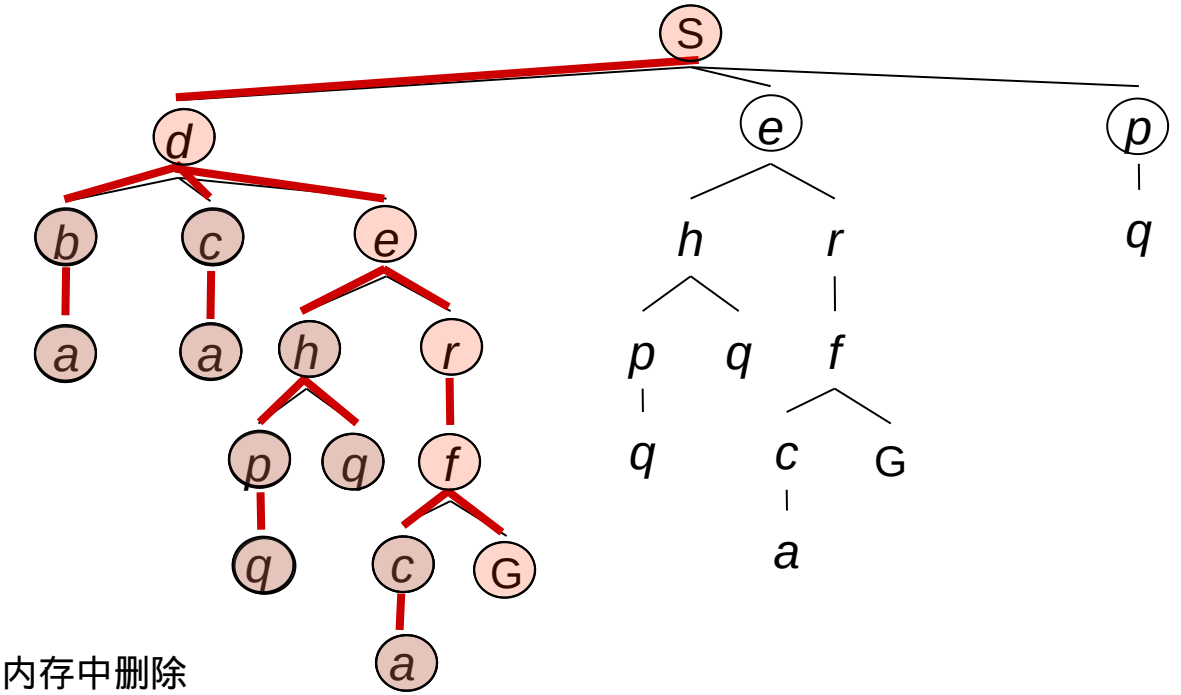
a e e p

e e p

h r e p

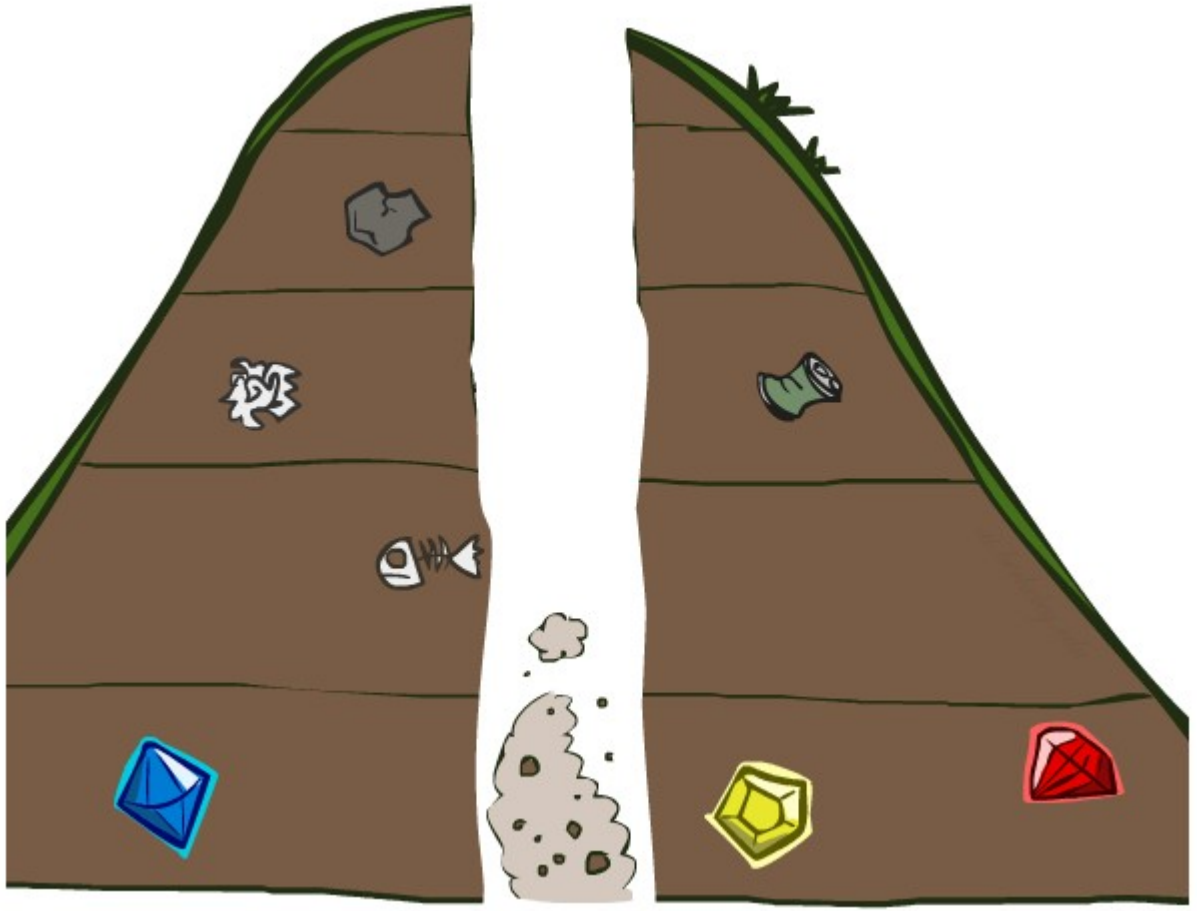
p q r e p

……



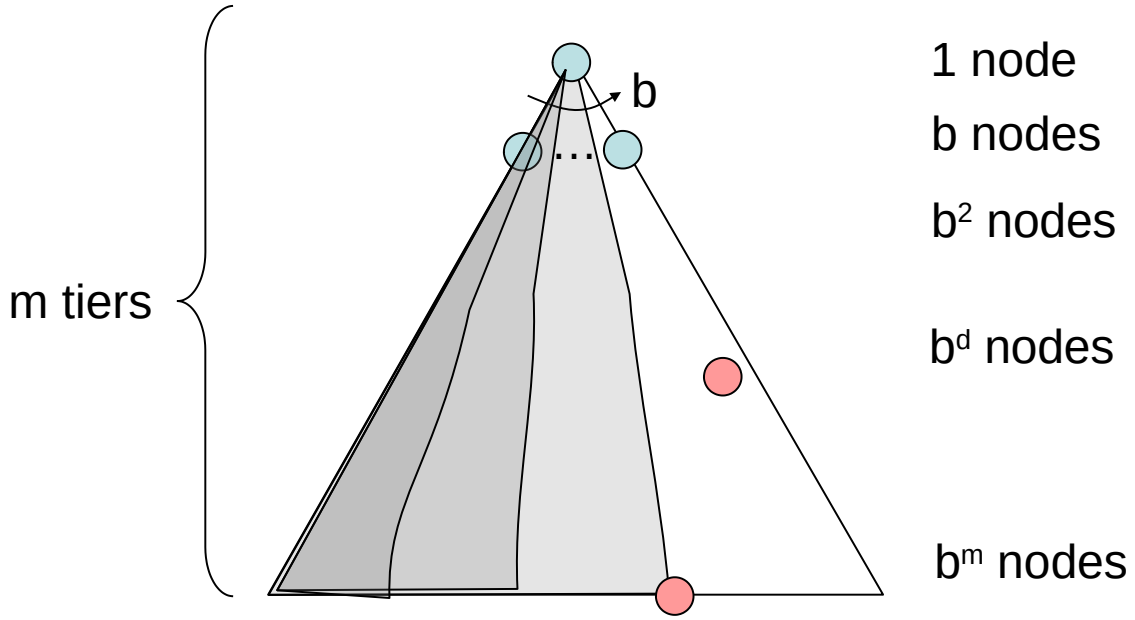
已扩展且在边缘中没有后代的结点可以从内存中删除

Search Algorithm Properties



深度优先搜索（DFS）的性能

- 完备性？No complete in finite spaces
- 最优性？No
- 时间复杂度？$O(b^m)$
- 空间复杂度？$O(bm)$ linear space!



深度优先搜索只要存储一条从根结点到叶子结点的路径，以及该路径上每个结点所有未被扩展的兄弟结点即可。

已扩展并且在边缘中没有后代的结点可以从内存中删除

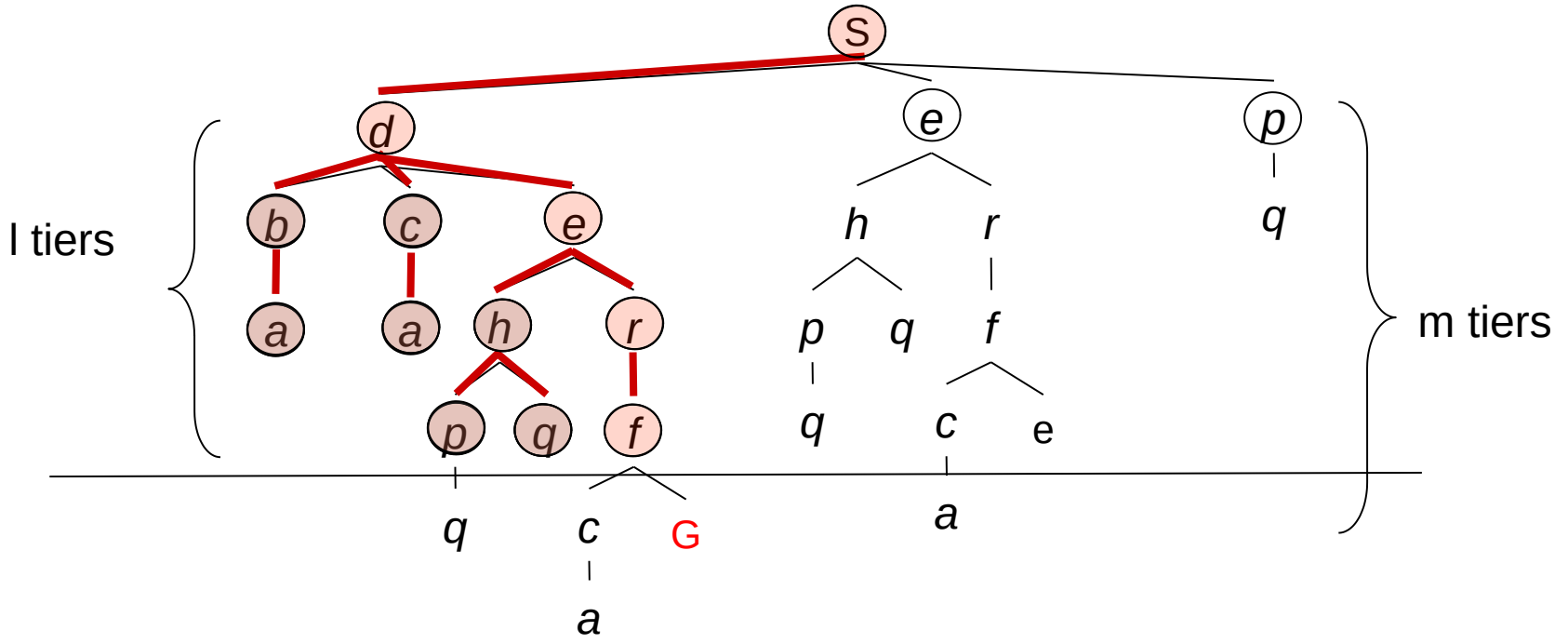
3.4.4 深度受限搜索

= DFS with depth limit *l*

- 深度为 l 的结点被当做最深层结点（没有后继结点）来对待。

- 避免 DFS 在无限状态空间下搜索失败，解决了无穷路径问题

- 算法的性能：
  - 时间复杂度 $O(b^l)$
  - 空间复杂度 O(bl)
  - l<m: 不是完备的
  - 不是最优的

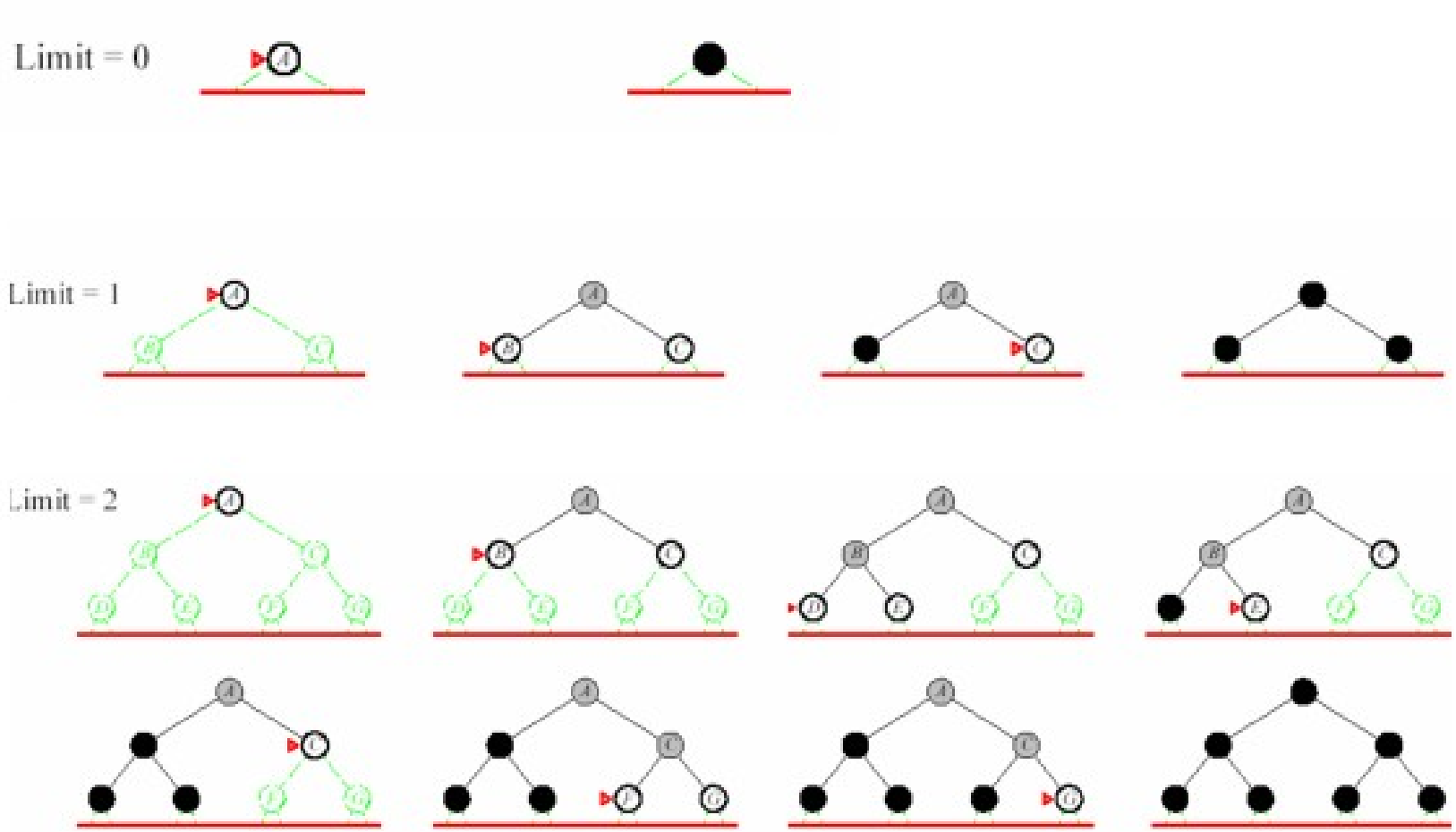


3.4.5 迭代加深的深度优先搜索（ IDS ）

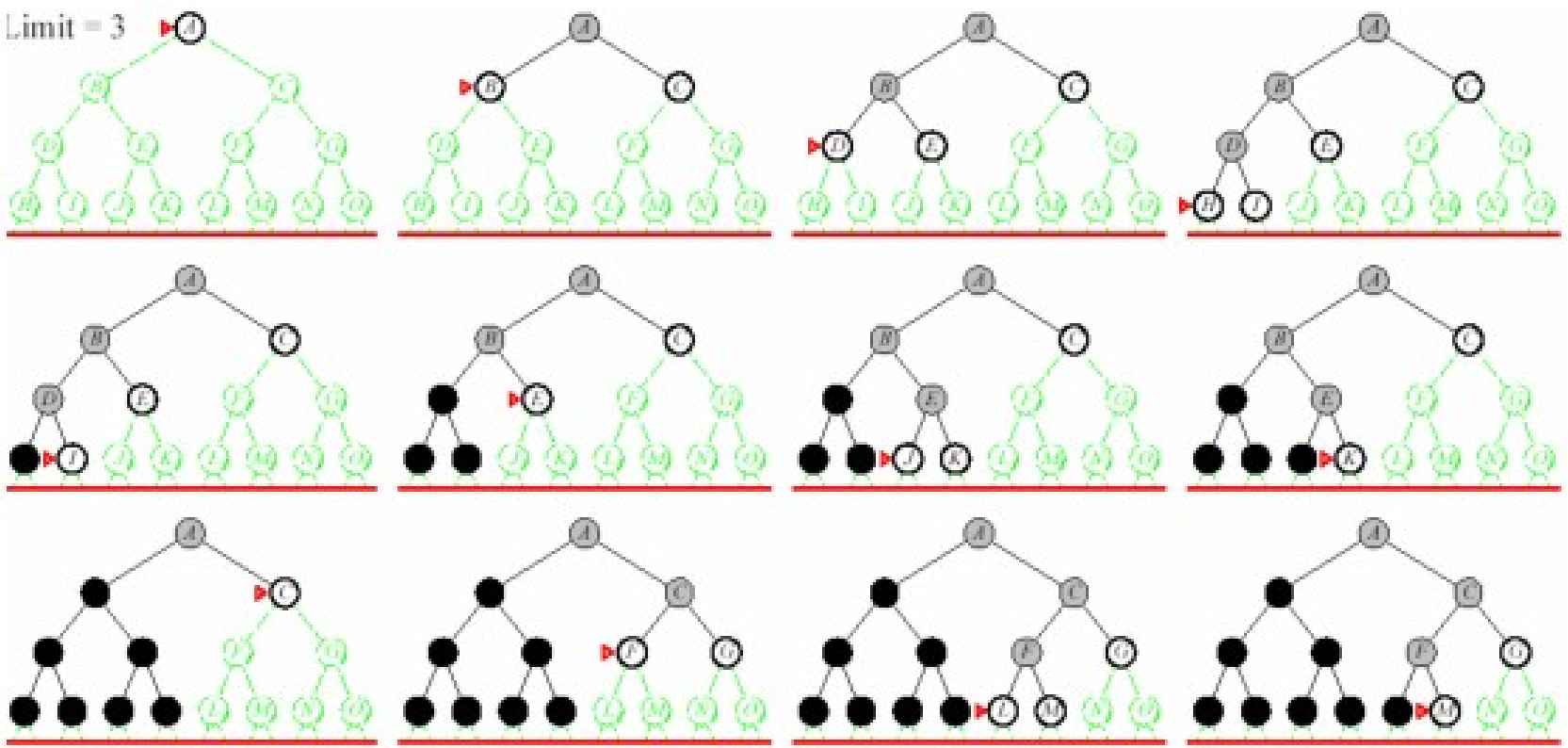
- 在深度受限搜索的基础上，逐步增加深度限制。该算法结合了深度优先和广度优先的优点。

- 算法原理：设最大深度 limit ，开始 limit 设为 1 ，深度优先搜索，如果没有找到目标，则 limit 加一，再次深度优先搜索，以此类推直到找到目标。

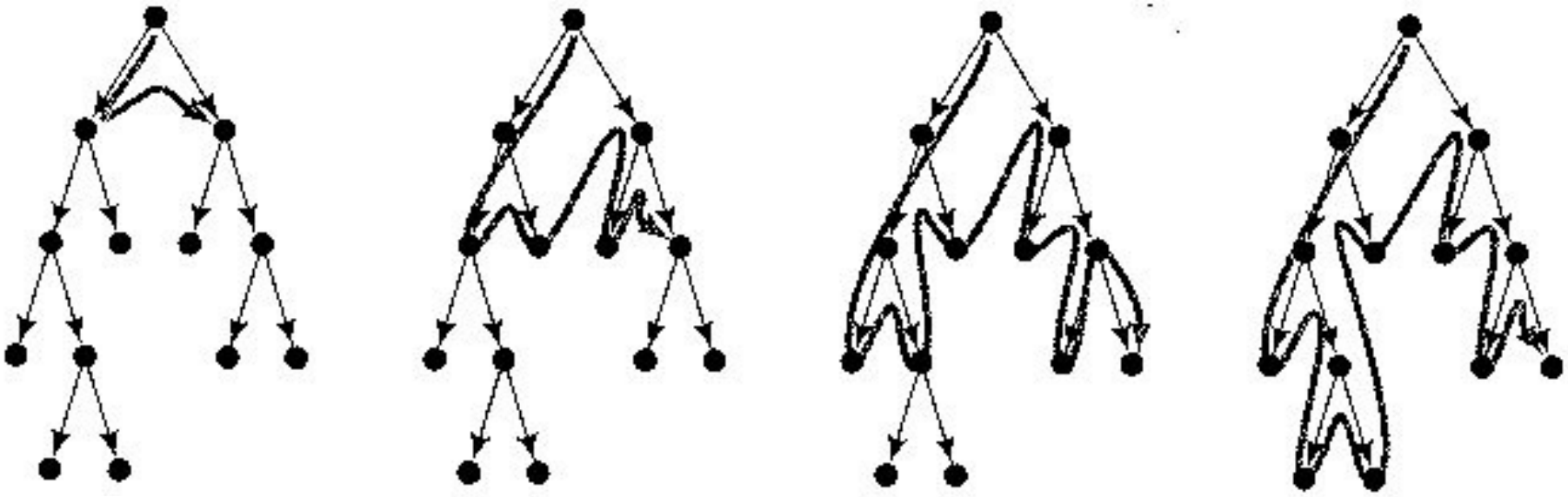
迭代加深的深度优先搜索（ IDS ）



迭代加深的深度优先搜索（ IDS ）



3.4.5 迭代加深的深度优先搜索（ IDS ）

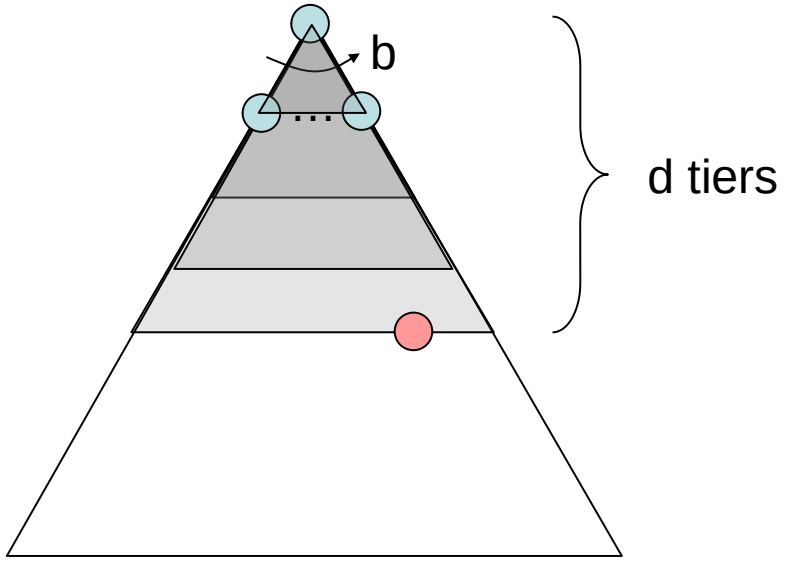


limit=1　　limit=2　　limit=3　　limit=4

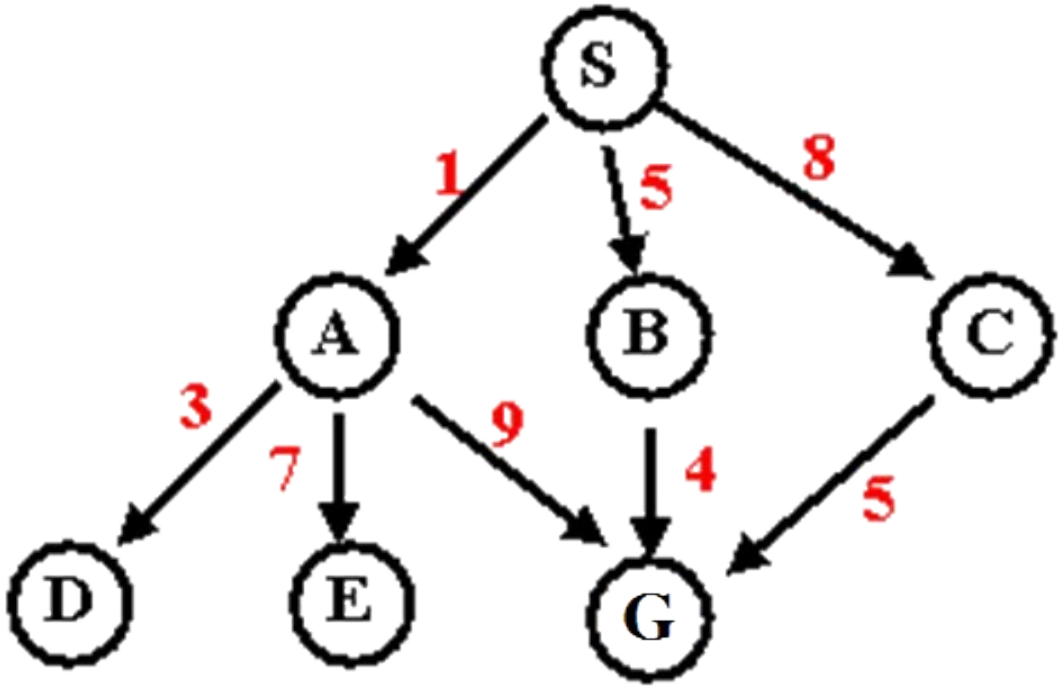
优势：既可以避免陷入深度无限的分支，同时还可以找到深度最浅的目标解，从而在每一步代价一致的时候找到最优解，再加上其优越的空间复杂度，常常作为首选的无信息搜索策略。

迭代加深的深度优先搜索（ IDS ）

- IDS = DFS + BFS
- 完备性？Yes（分支因子 b 有限时）
- 最优性？Yes, if step cost = 1
- 时间复杂度？$O(b^d)$
- 空间复杂度？$O(bd)$



课堂练习



迭代加深的深度优先搜索

边缘队列：

Limit=1

S

A B C

B C

C

Limit=2

S

A B C

D E G B C

E G B C

G B C

搜索序列：S A B C S A D E G

解序列： S A G ，路径耗散：

Summary of algorithms

Criterion	Breadth-First	Uniform-Cost	Depth-First	Depth-Limited	Iterative Deepening
Complete?	Yes[a]	Yes[a,b]	No	No	Yes[a]
Time	$O(b^d)$	$O(b^{1+\lceil C^*/\epsilon \rceil})$	$O(b^m)$	$O(b^l)$	$O(b^d)$
Space	$O(b^d)$	$O(b^{1+\lceil C^*/\epsilon \rceil})$	$O(bm)$	$O(bl)$	$O(bd)$
Optimal?	Yes[c]	Yes	No	No	Yes[c]

搜索策略比较：

b 指分支因子， *d* 指最浅解的深度， *m* 指搜索树的最大深度， *l* 是深度界限。

右上角标的含义： a 指当 *b* 有限时算法是完备的， b 指若对正数□有单步代价□□ □，则是完备的。 *c* 单步代价相同时算法最优。

Search Gone Wrong?