

# CS 730/730W/830: Intro AI

---

[Search](#)

[Basic Algorithms](#)

[A Clever Algorithm](#)

[EOLQs](#)

1 handout: slides

# EOLQs

---

[Search](#)

[Basic Algorithms](#)

[A Clever Algorithm](#)

[EOLQs](#)

## Search

- Cognitive Science
- Representation
- Problem Solving

Basic Algorithms

A Clever Algorithm

EOLQs

# Search

Search

Cognitive Science

Representation

Problem Solving

[Basic Algorithms](#)

[A Clever Algorithm](#)

[EOLQs](#)

The ability to think is perhaps the most distinctive of human capacities. Typically, thinking involves mentally representing some aspects of the world (including aspects of ourselves) and manipulating these representations or beliefs so as to yield new beliefs, where the latter may aid in accomplishing a goal.  
—Edward E. Smith (Psychology, U Michigan)

The ability to solve problems is one of the most important manifestations of human thinking. ... We might therefore suspect that problem solving depends on general cognitive abilities that can potentially be applied to an essentially unlimited range of domains.  
—Keith Holyoak (Psychology, UCLA)

# Representation

---

Search

■ Cognitive Science

■ Representation

■ Problem Solving

Basic Algorithms

A Clever Algorithm

EOLQs

VW search space

VW state space

MC representation

# Formalizing Problem Solving

---

Search

■ Cognitive Science

■ Representation

■ Problem Solving

Basic Algorithms

A Clever Algorithm

EOLQs

**State:** hypothetical world state

**Operators:** actions that modify world

**Goal:** desired state or test



(Herbert Simon and Allen Newell, “Computer simulation of human thinking and problem solving”, 1961)

Search

**Basic Algorithms**

- Alg 1
- Alg 2
- Uniform-cost
- Graphs
- Comparison
- Time vs space
- Both?
- Break

A Clever Algorithm

EOLQs

# Basic Algorithms

# -First Search

---

Search

Basic Algorithms

■ Alg 1

■ Alg 2

■ Uniform-cost

■ Graphs

■ Comparison

■ Time vs space

■ Both?

■ Break

A Clever Algorithm

EOLQs

$Q \leftarrow$  an ordered list containing just the initial state.

Loop

If  $Q$  is empty,  
then return failure.

$Node \leftarrow \text{Pop}(Q)$ .

If  $Node$  is a goal,  
then return  $Node$  (or path to it).

else

$Children \leftarrow \mathbf{Expand}(Node)$ .

Add  $Children$  to front of  $Q$ .

# Evaluating DFS

---

Assume branching factor  $b$  and solution at depth  $d$ .

Completeness:

Time:

Space:

Admissibility:

Search

Basic Algorithms

■ Alg 1

■ Alg 2

■ Uniform-cost

■ Graphs

■ Comparison

■ Time vs space

■ Both?

■ Break

A Clever Algorithm

EOLQs

# Breadth-First Search

---

Search

Basic Algorithms

■ Alg 1

■ Alg 2

■ Uniform-cost

■ Graphs

■ Comparison

■ Time vs space

■ Both?

■ Break

A Clever Algorithm

EOLQs

Let  $Q$  be an empty list.

Loop

If  $Q$  is empty,  
then return failure.

$Node \leftarrow Pop(Q)$ .

If  $Node$  is a goal,  
then return  $Node$  (or path to it).

else

$Children \leftarrow \mathbf{Expand} (Node)$ .

Add  $Children$  to end of  $Q$ .

←

# Evaluating BrFS

---

Search

Basic Algorithms

■ Alg 1

■ Alg 2

■ Uniform-cost

■ Graphs

■ Comparison

■ Time vs space

■ Both?

■ Break

A Clever Algorithm

EOLQs

Assume branching factor  $b$  and solution at depth  $d$ .

Completeness:

Time:

Space:

Admissibility:

# Uniform-Cost Search

---

Search

Basic Algorithms

■ Alg 1

■ Alg 2

■ Uniform-cost

■ Graphs

■ Comparison

■ Time vs space

■ Both?

■ Break

A Clever Algorithm

EOLQs

Let  $Q$  be an empty list.

Loop

If  $Q$  is empty,  
then return failure.

$Node \leftarrow Pop(Q)$ .

If  $Node$  is a goal,  
then return  $Node$  (or path to it).

else

$Children \leftarrow \mathbf{Expand} (Node)$ .

Merge  $Children$  into  $Q$ , keeping sorted by path cost. ←

# Dealing with Graphs

---

Search

Basic Algorithms

- Alg 1
- Alg 2
- Uniform-cost

■ **Graphs**

- Comparison
- Time vs space
- Both?
- Break

A Clever Algorithm

EOLQs

1. Check for cycles with ancestors
2. Maintain closed list (hash table) to detect duplicates

# Comparison

---

## Search

---

## Basic Algorithms

---

- Alg 1
- Alg 2
- Uniform-cost
- Graphs

## ■ Comparison

- Time vs space
- Both?
- Break

## A Clever Algorithm

---

## EOLQs

---

Algorithm	Time	Space	Complete	Admissible
Depth-first	$b^m$	$bm$	If $m \geq d$	No
Breadth-first	$b^d$	$b^d$	Yes	If ops cost 1
Uniform-cost	$b^d$	$b^d$	Yes	Yes

  

branching factor	b
maximum depth	m
solution depth	d

# Time and Space for BrFS/UCS

Assume  $b = 10$ , 1,000 nodes/sec, 100 bytes/node.

Sol. depth	Nodes	Time	Space
1	11	11 ms.	1.1 Kb
2	111	.1 sec	11 Kb
4	11,111	11 sec	1 Mb
6	$10^6$	18 min	111 Mb
8	$10^8$	31 hours	11 Gb
10	$10^{10}$	128 days	1 Tb
12	$10^{12}$	35 yrs	111 Tb
14	$10^{14}$	3,500 yrs	11 Pb

## Search

### Basic Algorithms

- Alg 1
- Alg 2
- Uniform-cost
- Graphs
- Comparison
- Time vs space
- Both?
- Break

### A Clever Algorithm

### EOLQs

# Search Conundrum

---

Search

Basic Algorithms

- Alg 1
- Alg 2
- Uniform-cost
- Graphs
- Comparison
- Time vs space

■ Both?

■ Break

A Clever Algorithm

EOLQs

Breadth-first uses  $b^d$  space

*but complete and admissible*

Depth-first complete only if *limit*  $> d$ , not admissible

*but  $bd$  space*

How can we get only the best of both?

# Break

---

Search

Basic Algorithms

- Alg 1
- Alg 2
- Uniform-cost
- Graphs
- Comparison
- Time vs space
- Both?

■ Break

A Clever Algorithm

EOLQs

- website
- Asst 1: EC
- recitation
- blog entries
- office hours: Thu 10:30-11:30

Search

Basic Algorithms

**A Clever Algorithm**

- IDS
- Evaluating IDS
- IDS time

EOLQs

# A Clever Algorithm

# Iterative Deepening Search

---

Search

Basic Algorithms

A Clever Algorithm

■ **IDS**

■ Evaluating IDS

■ IDS time

EOLQs

```
for  $d = 1$  to  $\infty$  do
  depth-first search to level  $d$ 
  if it succeeds
    then return solution
```

Could this possibly be efficient?

# Evaluating IDS

---

[Search](#)

[Basic Algorithms](#)

[A Clever Algorithm](#)

■ IDS

■ Evaluating IDS

■ IDS time

[EOLQs](#)

Assume branching factor  $b$  and solution at depth  $d$ .

Completeness:

Time:

Space:

Admissibility:

# Nodes Generated by IDS

Search

Basic Algorithms

A Clever Algorithm

■ IDS

■ Evaluating IDS

■ **IDS time**

EOLQs

$b = 2$

$d$	at $d$	in prev.	total	IDS	% of opt.
0	1	0	1	1	100.0
1	2	1	3	4	133.3
2	4	3	7	11	157.1
4	16	15	31	57	183.9

$b = 10$

$d$	at $d$	in prev.	total	IDS	% of opt.
0	1	0	1	1	100.0
1	10	1	11	12	109.1
2	100	11	111	123	110.8
4	10000	1111	11,111	12,345	111.1

# Nodes Generated by IDS

---

[Search](#)

[Basic Algorithms](#)

[A Clever Algorithm](#)

■ IDS

■ Evaluating IDS

■ **IDS time**

[EOLQs](#)

$$b^d + 2b^{d-1} + 3b^{d-2} + \dots + (d-1)b^2 + db$$

$$\approx b^d \left( \frac{b}{b-1} \right)^2$$

Search

Basic Algorithms

A Clever Algorithm

**EOLQs**

■ EOLQs

# EOLQs

[Search](#)

[Basic Algorithms](#)

[A Clever Algorithm](#)

[EOLQs](#)

[EOLQs](#)

Please write down the most pressing question you have about the course material covered so far and put it in the box on your way out.

*Thanks!*