




Word Search



EECE 3326 Optimization Methods
Instructor: Ningfang Mi

Word Search Problem

- ▶ Given a file containing
 - ▶ an $n \times n$ grid of letters
 - ▶ a dictionary containing a list of K possible words

s	e	z	y	w
k	c	o	d	e
u	z	a	e	a
a	b	z	c	d
t	j	m	k	p

5×5

code
bad
cake
dab
deck
...

- ▶ Print out all words found in the grid to the screen "correctness"
- ▶ Find an algorithm to solve this problem which runs quickly for large n and K "efficiency"

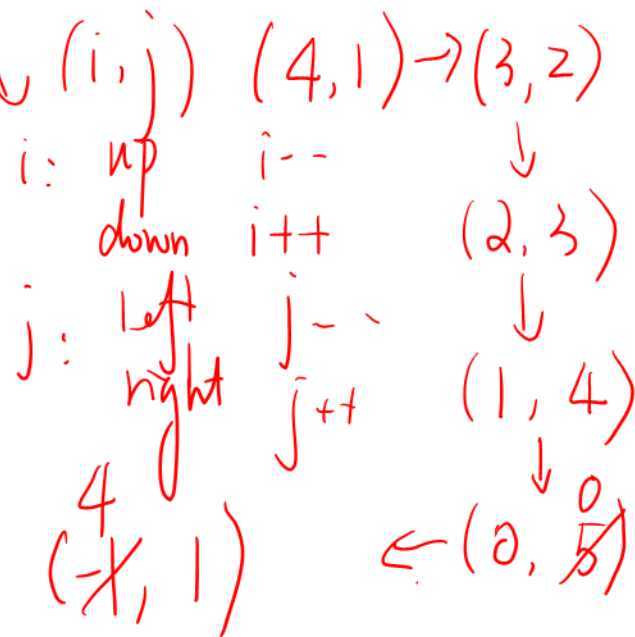
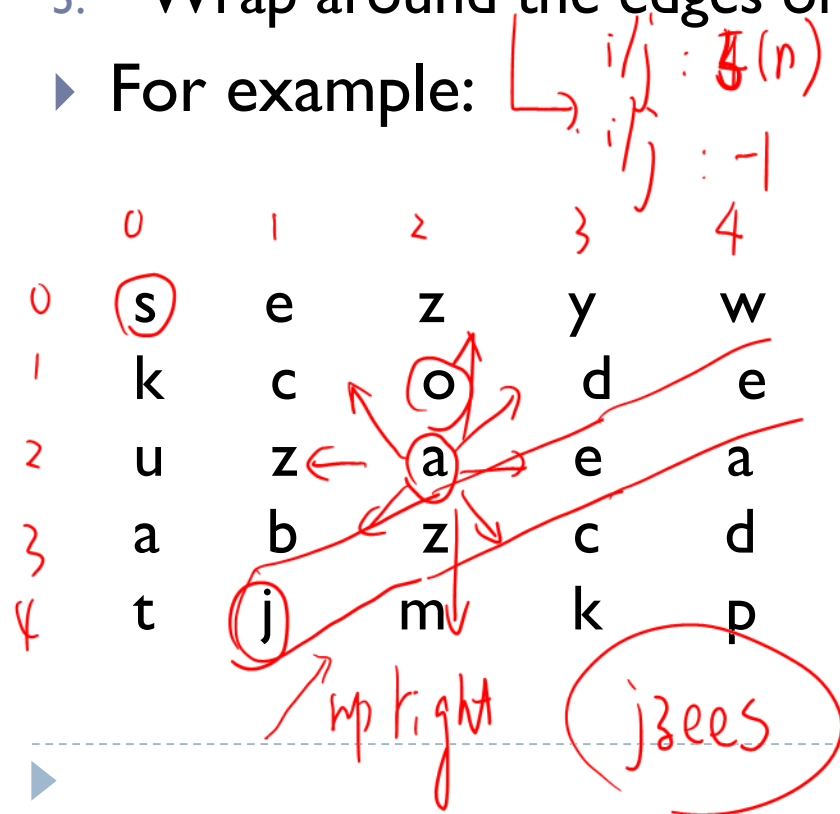
4. Give a starting letter, stop search when we've back to this letter.

Word Searching Rules

subprob. get a candidate string from the grid

1. Start from any letter in the grid ($n \times n$ starting points)
2. Search in eight directions
 - ▶ up, down, left, or right, or in any of the 4 diagonals
3. Wrap around the edges of the grid

▶ For example:



Brute-force Solution

- ▶ Scan the grid looking for all candidate words at each possible start position ⁽¹⁾ (2) all directions (8) (3) all possible lengths =
- ▶ For example:

5x5

	R	D	L	U	UR	DR	UL	DL
(1)	a ✓							
2	ae	az	az	ao	.	-	-	(8)
3	aea	..	azu	u	.	.	-	(8)
4
5	aeanz		azu	ae				(8)

(h-1) rows

Running Time for Brute Force

- ▶ How many starting locations are there?

$$n \times n = 25$$

- ▶ How many candidate strings start at each location?

$$(n-1) \times 8 + 1 = 4 \times 8 + 1 = 33$$

- ▶ Total number of candidate strings:

$$25 \times 33 = 825$$



Subprob2: Search a key string in the dict...

Running Time for Brute Force

- ▶ Running time for searching each candidate word in the dictionary

$$K = 90,000$$

- ▶ Worst case: not found.
K comparisons

- ▶ Avg case: $K/2 = 45,000$

- ▶ Avg time to compare two strings: 100ns

V

- ▶ Avg time to look up one candidate string in dictionary:

$$45,000 \times 100\text{ns} = 4.5 \times 10^{-3} \text{ s}$$

- ▶ Time to look up all candidate strings in dictionary:

$$825 \times \leftarrow = 2.7 \text{ s}$$

Analysis of Brute Force Algorithm

- ▶ **Input size**

- ▶ n x n letters in grid
- ▶ K words in **unsorted** dictionary

- ▶ **Basic operation:**

- ▶ *String comparison*
Total candidate words:

$$C(n) = n^2(8(n-1) + 1)$$

- ▶ **Total comparisons:**

$$(8n^3 - 7n^2) * \frac{k}{2} = O(n^3 k)$$



Analysis of Brute Force Algorithm

- ▶ Input size

- ▶ $n \times n$ letters in grid
- ▶ K words in sorted dictionary

- ▶ Sorting Algorithms

- ✓ ▶ Selection Sorting: $O(K \log K)$
- ▶ Binary Search: $O(\log K)$ ← $O(K)$ (seq. search)

- ▶ Total candidate words:

$$8n^3 - 7n^2$$

- ▶ Total comparisons:

$$(8n^3 - 7n^2) \cdot \log K = O(n^3 \log K)$$

- ▶ Total run time:

$$O(n^3 \log K + K \log K)$$

Analysis of Brute Force Algorithm

- ▶ Input size
 - ▶ $n \times n$ letters in grid
 - ▶ K words in **sorted** dictionary
- ▶ Sorting Algorithms
 - ▶ **QuickSort** :
 - ▶ Binary Search:
- ▶ Total candidate words:
- ▶ Total comparisons:
- ▶ Total run time:

