Carnegie Mellon

15-826: Multimedia (Databases) and Data Mining

Lecture #12: Text – Part I

C. Faloutsos

Must-read Material

- MM Textbook, Chapter 6

Outline

Goal: 'Find similar / interesting things'

- Intro to DB
- Indexing - similarity search
- Data Mining

Indexing - Detailed outline

- primary key indexing
- secondary key / multi-key indexing
- spatial access methods
- fractals
- text
- multimedia
- ...

Text - Detailed outline

- text
  - problem
  - full text scanning
  - inversion
  - signature files
  - clustering
  - information filtering and LSI

Problem - Motivation

- Eg., find documents containing "*data*", "*retrieval*"
- Applications:

Problem - Motivation

- Eg., find documents containing "*data*", "*retrieval*"
- Applications:
  - Web
  - law + patent offices
  - digital libraries
  - information filtering

Problem - Motivation

- Types of queries:
  - boolean ('data' AND 'retrieval' AND NOT ...)

Problem - Motivation

- Types of queries:
  - boolean (‘data’ AND ‘retrieval’ AND NOT ...)
  - additional features (‘data’ ADJACENT ‘retrieval’)
  - keyword queries (‘data’, ‘retrieval’)

- How to search a large collection of documents?

Problem

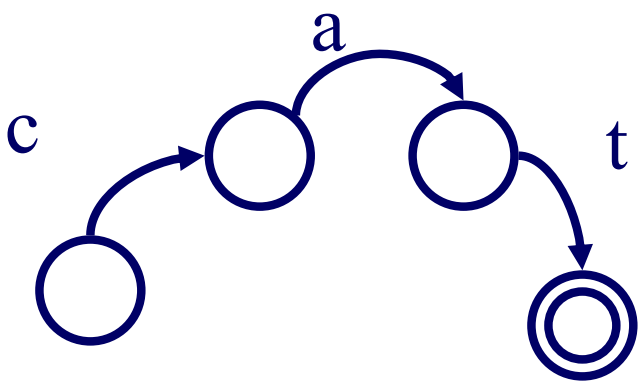
- How to find doc's with "data mining"?

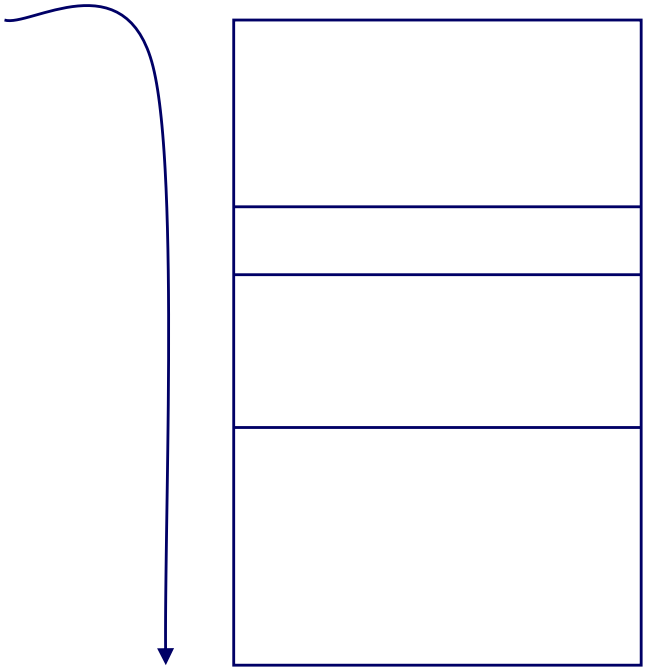
Conclusion

- How to find doc's with "data mining"?
- A1: full text scanning
  - A1.1: string editing distance

Full-text scanning

- Build a FSA; scan





Full-text scanning

- for single term:
  - (naive: O(N*M))

ABRACADABRA text

CAB pattern

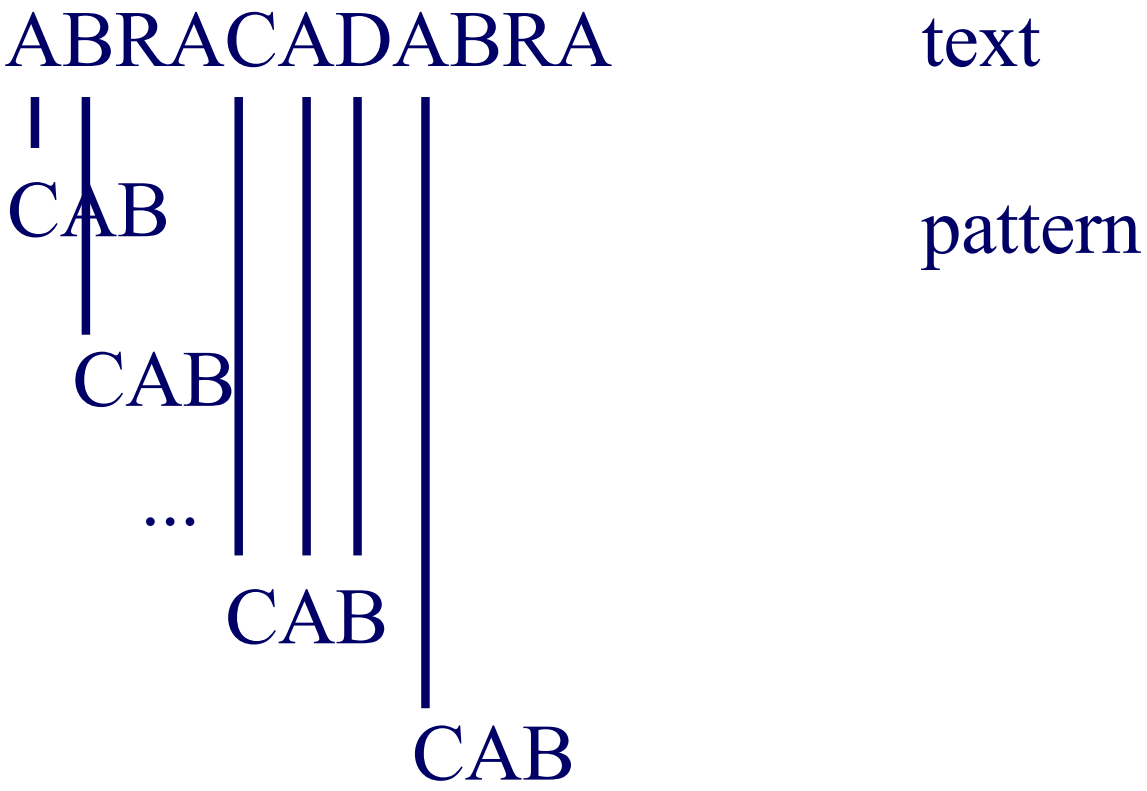
Full-text scanning

- for single term:
  - (naive: O(N*M))
  - Knuth Morris and Pratt (‘77)
    - build a small FSA; visit every text letter once only, by carefully shifting more than one step

ABRACADABRA text

CAB pattern

Full-text scanning



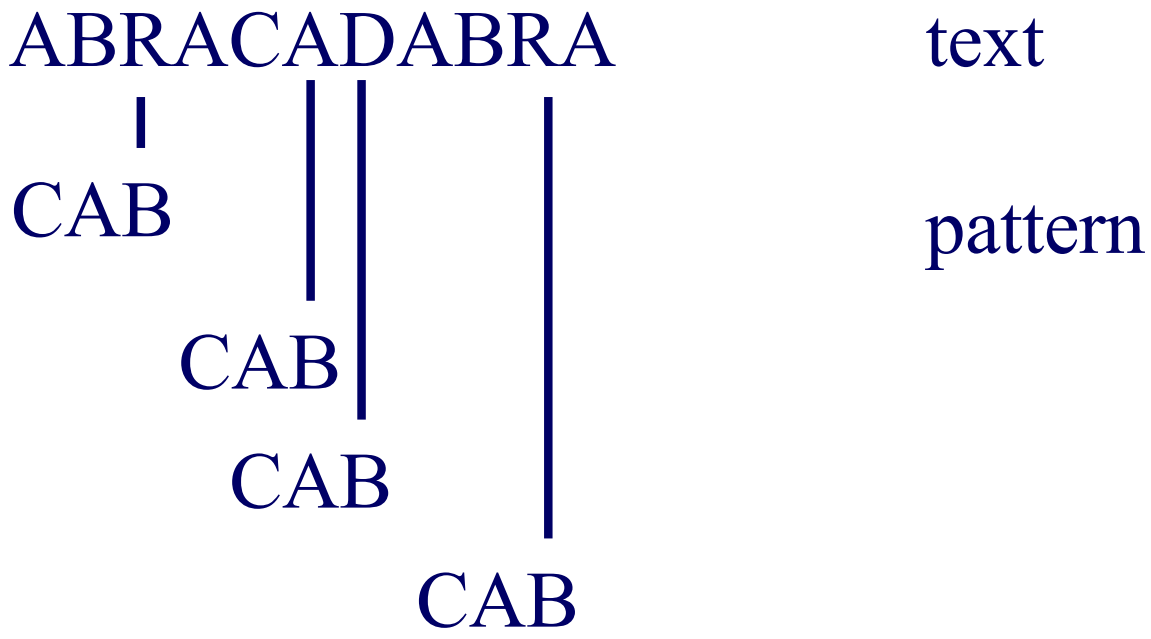
Full-text scanning

- for single term:
  - (naive: O(N*M))
  - Knuth Morris and Pratt (‘77)
  - Boyer and Moore (‘77)
    - preprocess pattern; start from **right to left** & **skip**!

ABRACADABRA text

CAB pattern

Full-text scanning



Full-text scanning

ABRACADABRA text

OMINOUS pattern

OMINOUS

Boyer+Moore: fastest, in practice
Sunday (‘90): some improvements

Full-text scanning

- For multiple terms (w/o "don't care" characters): Aho+Corasic ('75)
  - again, build a simplified FSA in O(M) time
- Probabilistic algorithms: 'fingerprints' (Karp + Rabin '87)
- approximate match: 'agrep' [Wu+Manber, Baeza-Yates+, '92]; available on unix/linux/win/mac

Full-text scanning

- Approximate matching - **string editing** distance:

  d('survey' , 'surgery') = 2

  = min # of insertions, deletions, substitutions to transform the first string into the second

  SURVEY
  SURGERY

Full-text scanning

- **string editing** distance - how to compute?
- A:

Full-text scanning

- **string editing** distance - how to compute?
- A: dynamic programming

  cost(i, j) = cost to match prefix of length *i* of first string *s* with prefix of length *j* of second string *t*

Full-text scanning

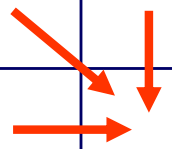
```
if s[i] = t[j] then
      cost( i, j ) = cost(i-1, j-1)
else
      cost(i, j ) = min (
                  1 + cost(i, j-1) // deletion
                  1 + cost(i-1, j-1) // substitution
                  1 + cost(i-1, j) // insertion
                  )
```

String editing distance

	ϕ	S	U	R	V	E	Y
ϕ	0	1	2	3	4	5	6
S	1						
U	2						
R	3						
G	4						
E	5						
R	6						
Y	7						

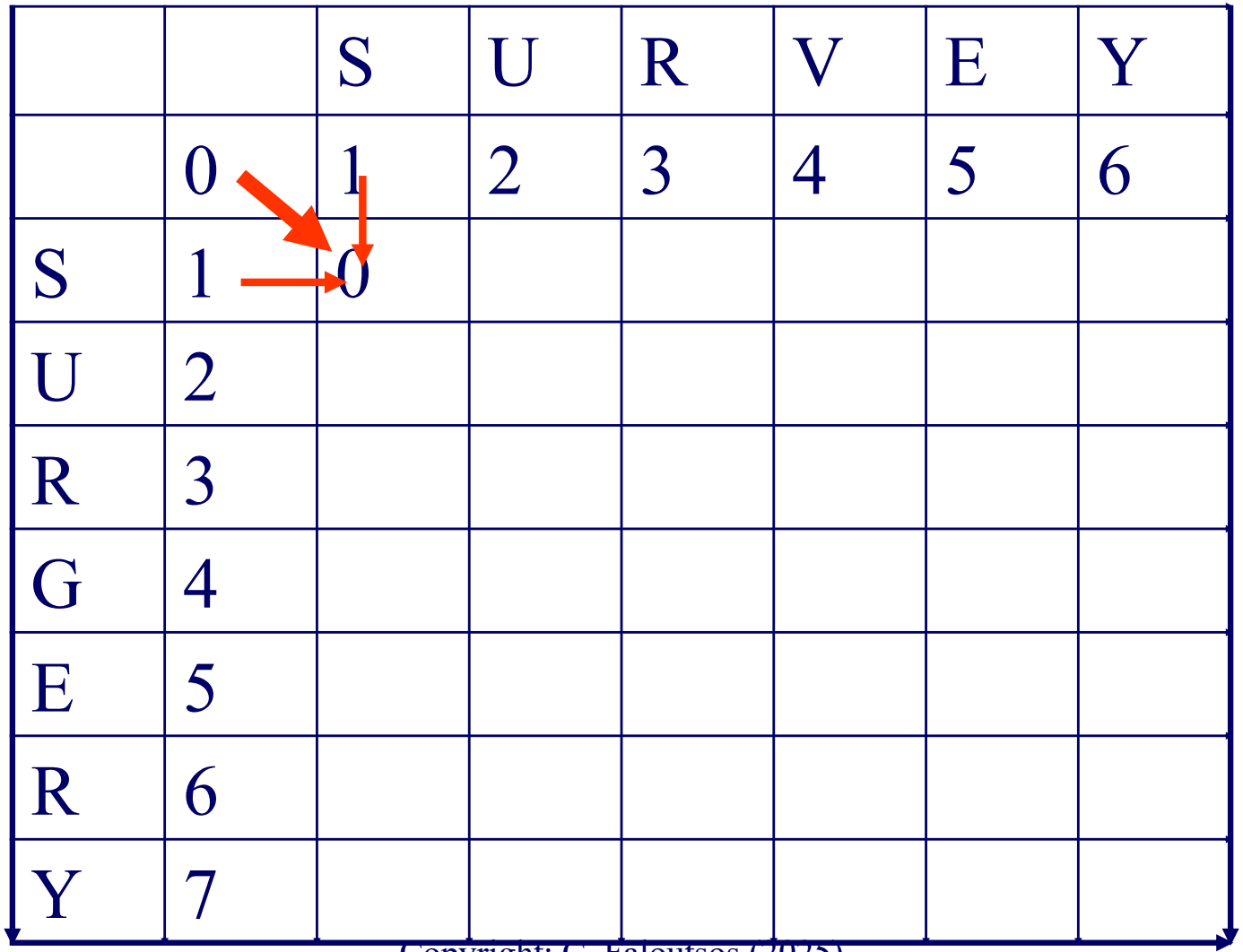
String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1						
U	2						
R	3						
G	4						
E	5						
R	6						
Y	7						



Carnegie Mellon

String editing distance



		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0					
U	2						
R	3						
G	4						
E	5						
R	6						
Y	7						

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0				
R	3	2					
G	4	3					
E	5	4					
R	6	5					
Y	7	6					

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1				
G	4	3	2				
E	5	4	3				
R	6	5	4				
Y	7	6	5				

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1			
E	5	4	3	2			
R	6	5	4	3			
Y	7	6	5	4			

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2		
R	6	5	4	3	3		
Y	7	6	5	4	4		

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	
Y	7	6	5	4	4	3	

String editing distance

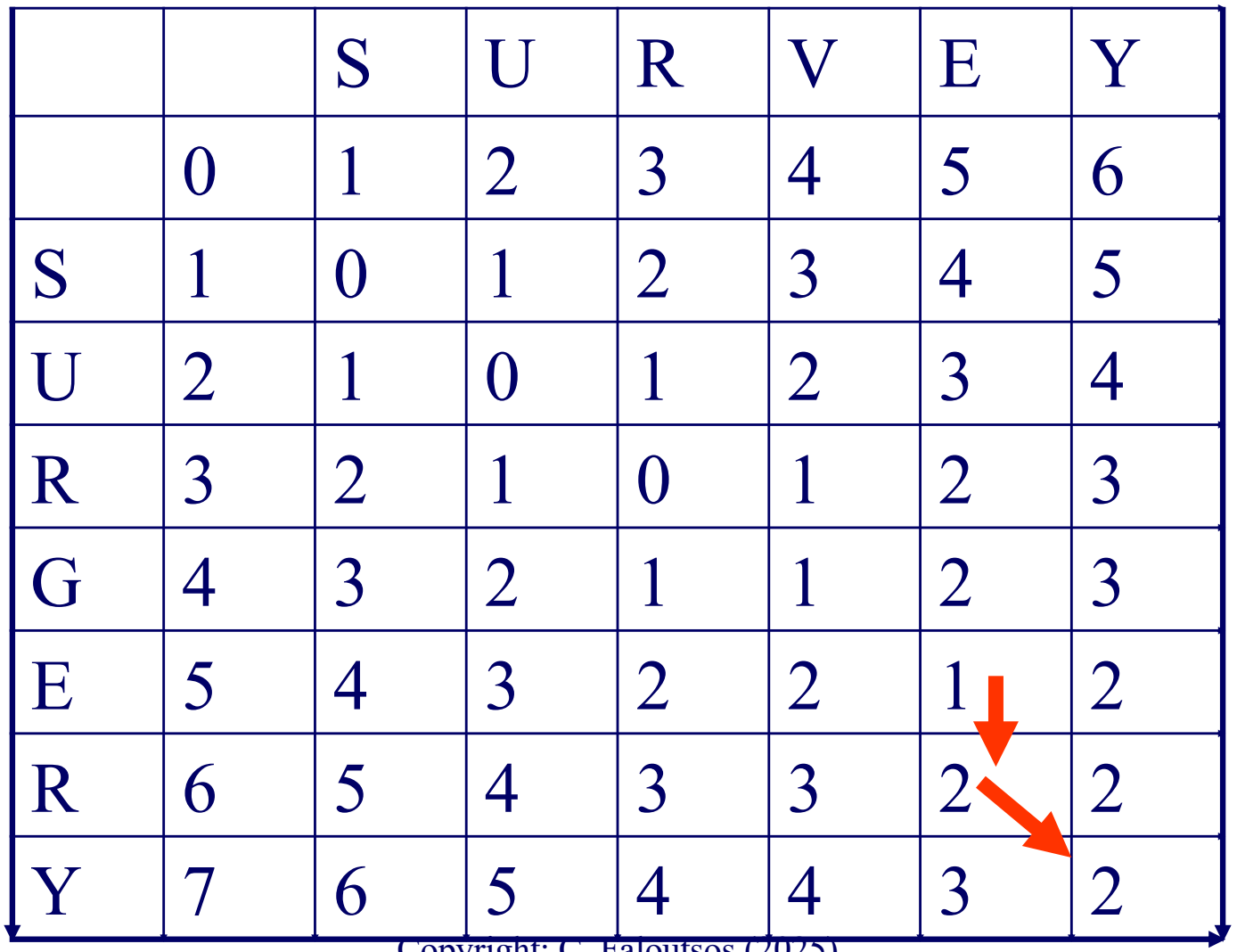
		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	2

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	2



String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	2

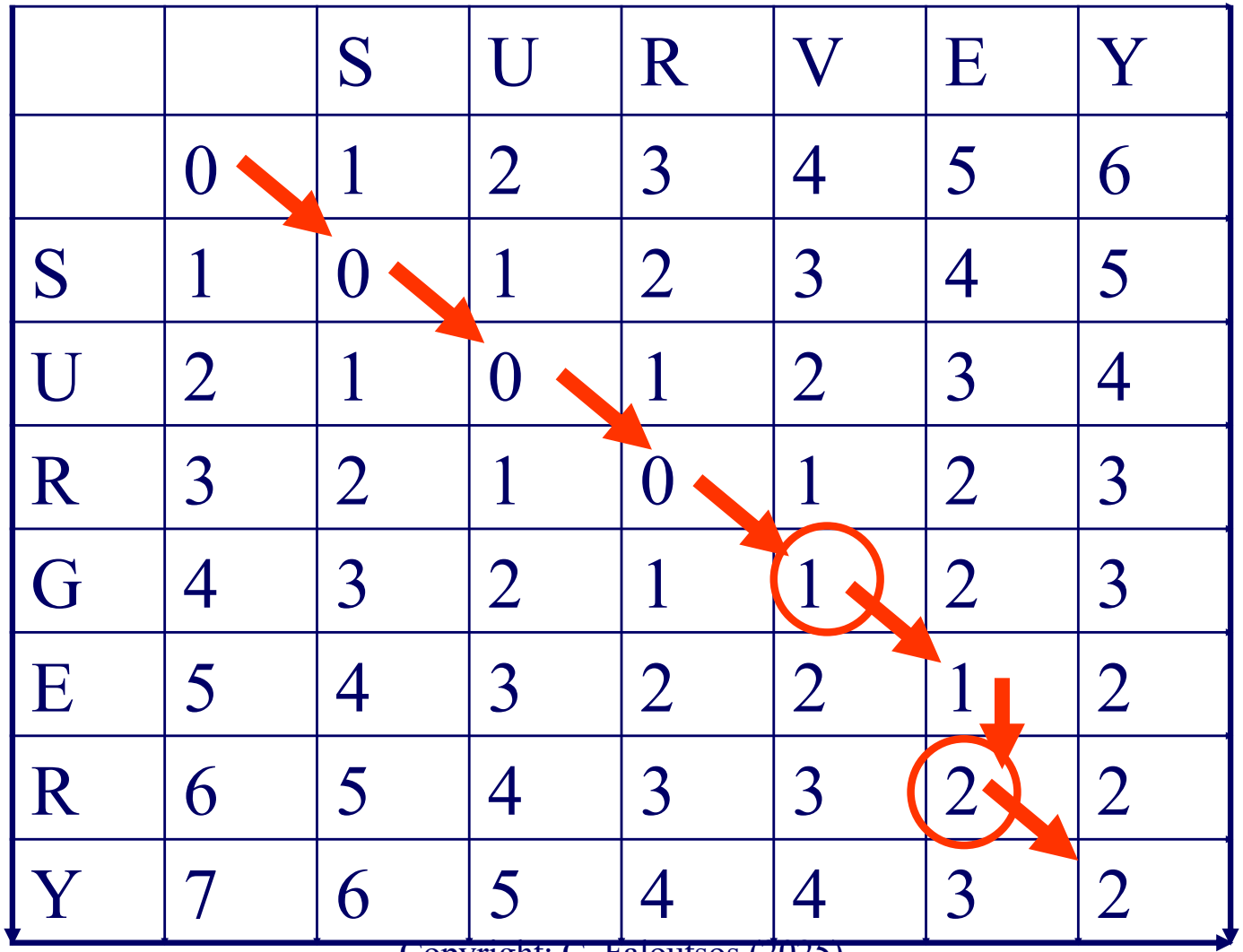
String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	2

String editing distance

		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	2

String editing distance



		S	U	R	V	E	Y
	0	1	2	3	4	5	6
S	1	0	1	2	3	4	5
U	2	1	0	1	2	3	4
R	3	2	1	0	1	2	3
G	4	3	2	1	1	2	3
E	5	4	3	2	2	1	2
R	6	5	4	3	3	2	2
Y	7	6	5	4	4	3	2

subst.

del.

Full-text scanning

Complexity: O(M*N) (when using a matrix to 'memoize' partial results)

Full-text scanning

Conclusions:

- Full text scanning needs no space overhead, but is slow for large datasets

Conclusion

- How to find doc's with "data mining"?
- A1: full text scanning
  - A1.1: string editing distance