

Text Fingerprinting

Mathieu Raffinot
CNRS - LIAFA

- $S = s_1..s_N$ string of length n
- alphabet Σ of size $|\Sigma|$, not fixed (possibly $O(n)$)

A fingerprint f : set of character(s) of a substring $s_i.. s_j$

General problem:

Compute and represent the set of all fingerprints of S

Examples:

dccbcbabbbc

{a} {b} {c} {d} {c,d} {b,c} {a,b} {b,c,d} {a,b,c} {a,b,c,d}

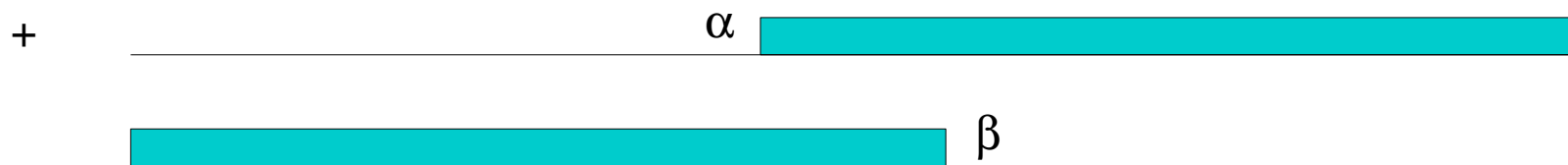
acbdcadad

{a} {b} {c} {d} {a,c} {a,d} {b,c} {b,d} {c,d} {a,b,c} {a,c,d} {b,c,d} {a,b,c,d}

Maximal location $\langle i, j \rangle$ of f



α not in f , β not in f



Number of maximal locations: $L \leq n|\Sigma|$ Complexity of the bound easily reached

But is usually much less

$$\Sigma_k = \{a_1, a_2, \dots, a_k\} \quad w_1 = a_1, w_k = w_{k-1} a_k w_{k-1}$$

$$w_2 = a_1(a_2)a_1, w_3 = (a_1 a_2 a_1) a_3 (a_1 a_2 a_1), \dots$$

$$|w_k| \cdot |L_k| = k \cdot (2^k - 1) \quad |L|_k = 2^{k+1} - (k+2) \quad \longrightarrow \quad |L|_k = o(|w_k| \cdot |L_k|)$$

First Part

Compute the set of all fingerprints

Naming technique

$\{a,c,e,f\}$ $\Sigma = \{a,b,c,d,e,f,g,h\}$

[7]							
[5]				[6]			
[2]		[2]		[3]		[4]	
[1]	[0]	[1]	[0]	[1]	[1]	[0]	[0]
a	b	c	d	e	f	g	h

$\log |\Sigma| + 1$

[9] ★							
[5]				[8]★			
[2]		[2]		[3]		[2]★	
[1]	[0]	[1]	[0]	[1]	[1]	[1]	[0]
						★	

$\{a,c,e,f,g\}$

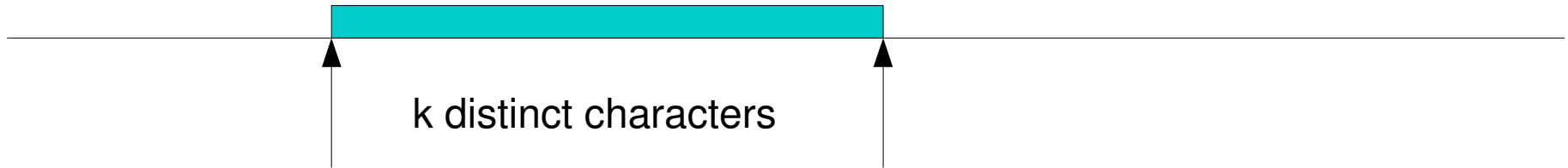
[10]★							
[5]				[5]★			
[2]		[2]		[2]★		[2]	
[1]	[0]	[1]	[0]	[1]	[0]	[1]	[0]
						★	

$\{a,c,e,g\}$

Names = $\{[1],[2],[3],[4],[5],[6],[7],[8],[9],[10]\}$

Fingerprints = $\{[7],[9],[10]\}$

Amir, Apostolico, Landau, Satta 2003

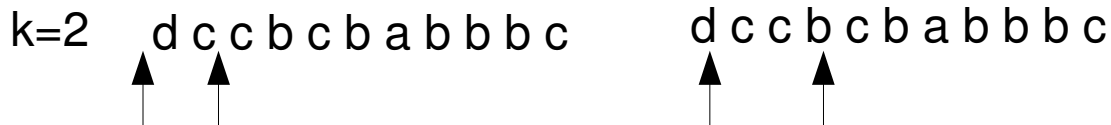
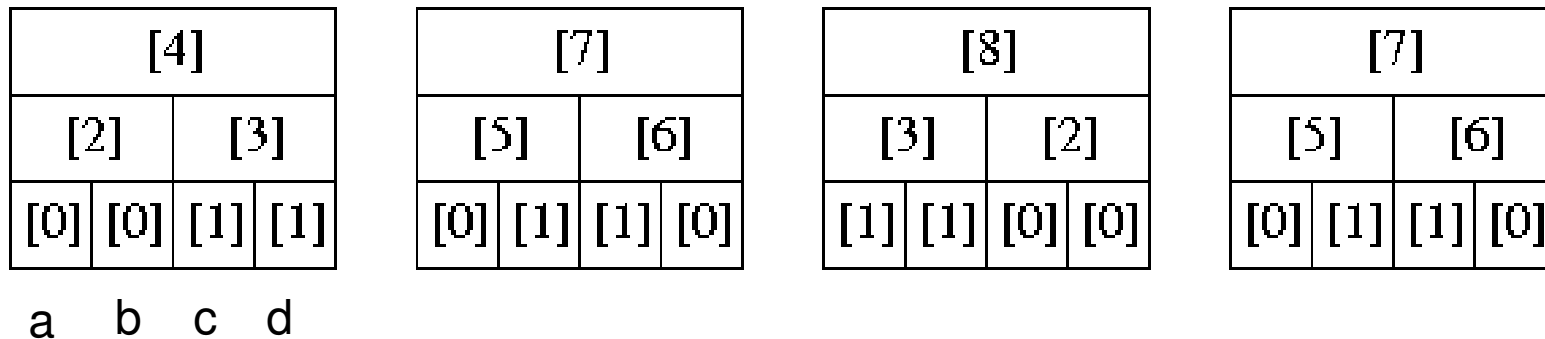


Changing a character: $O(\log |\Sigma| \log n)$ (n new names maximum by level)

One iteration: $n \log |\Sigma| \log n$

Important: different set of names for each iteration

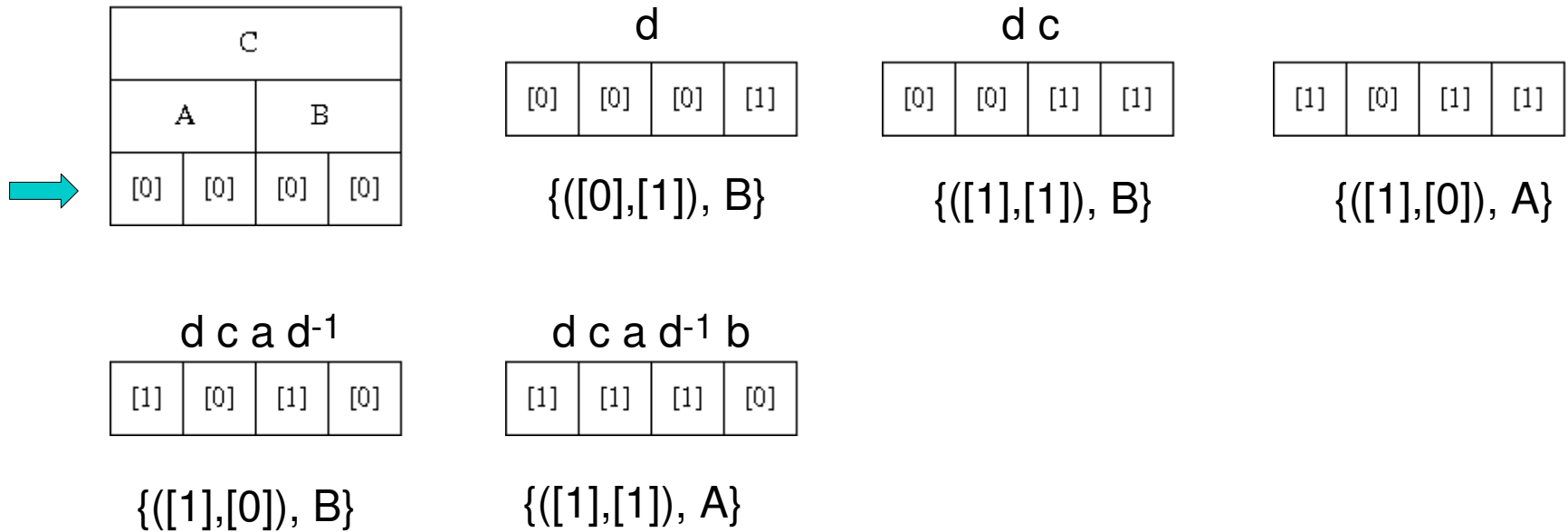
$|\Sigma|$ iterations: $|\Sigma| n \log |\Sigma| \log n$



Tsur 2005

List of fingerprints: $d c a d^{-1} b$

$\{d\}, \{c,d\}, \{a,c,d\}, \{a,c\}, \{a,b,c\}$



List of changes:

$\{([0],[0]), A\} \{([0],[0]), B\} \mid \{([0],[1]), B\} \{([1],[1]), B\} \{([1],[0]), A\} \{([1],[0]), B\} \{([1],[1]), A\}$

Radix sort on the pairs + unique \rightarrow new names

Tsur 2005

List of changes:

$\{([0],[0]), A\} \{([0],[0]), B\} \mid \{([0],[1]), B\} \{([1],[1]), B\} \{([1],[0]), A\} \{([1],[0]), B\} \{([1],[1]), A\}$

$[2] \rightarrow ([0],[0])$

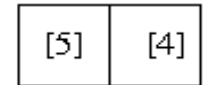
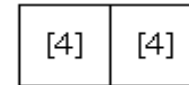
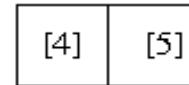
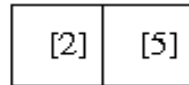
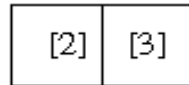
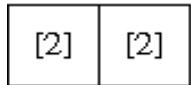
$[3] \rightarrow ([0],[1])$

$[4] \rightarrow ([1],[0])$

$[5] \rightarrow ([1],[1])$

New list:

$\{[2], A\} \{[2], B\} \mid \{[3], B\} \{[5], B\} \{[4], A\} \{[4], B\} \{[5], A\}$



$\{([2],[2]), C\} \quad \{([2],[3]), C\}$

New list: $\{([2],[2]), C\} \mid \{([2],[3]), C\} \{([2],[5]), C\} \{([4],[5]), C\} \{([4],[4]), C\} \{([5],[4]), C\}$

Radix sort, ...

Tsur 2005

Radix sort: $O(n)$ (bounded integers)

One iteration : $n \log |\Sigma|$ No more name search !

→ $|\Sigma|$ iterations: $|\Sigma| n \log |\Sigma|$

Problems

- does not depend of L
- distinct names at each iteration

Our approach (2006)

Simple sequence: no repeated character

lfo(i) a b a c e a b a c d

lfo(4)=ceab

a b a c e a b a c d

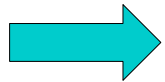
lfo(2) = bace

Concatenate # to the sequence

Bijection L / proper prefixes of lfo(i)

cea a b a c e a b a c d #

bac a b a c e a b a c d #



Compute all lfo(i) of S#

Our approach (2006)

Naming all proper prefixes of $lfo(i)$

a	b	c	b	a	d	c	a
b		b	a	d	c	a	
		a	d		a		
			c				

n lists:

- Tsur algorithm
- Common names

Simple sequence: $O(|L| \log |\Sigma|)$

General sequence: $O(n + |L| \log |\Sigma|)$

$|L| \leq n |\Sigma|$



Faster or as fast as that of Tsur

Our approach (2006)

Properties and operations on our names

- a unique set of names

→ Compute the LCP of two fingerprints in $\log |\Sigma|$

[9] ★							
[5] ★				[8] ★			
[2]		[2]		[3]		[2]	
[1]	[0]	[1]	[0]	[1]	[1]	[1]	[0]

★

[10] ★							
[5] ★				[5] ★			
[2]		[2]		[2] ★		[2]	
[1]	[0]	[1]	[0]	[1]	[0]	[1]	[0]

★

- names sorted by lexicographic order of fingerprints

Second Part

Represent the set of all fingerprints

Table of names

A fingerprint f  Fingerprint table of size $|\Sigma|$

Bottom up name

Complexity: $O(|\Sigma| \log n)$

Perfect hashing

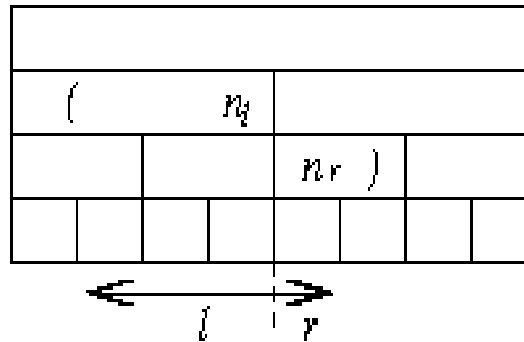
	Search	Preprocessing
Expected time:	$O(\Sigma)$	$O(F \log \Sigma)$

Preprocessing worst case time and space: $O(|F|^2 \log |\Sigma|)$

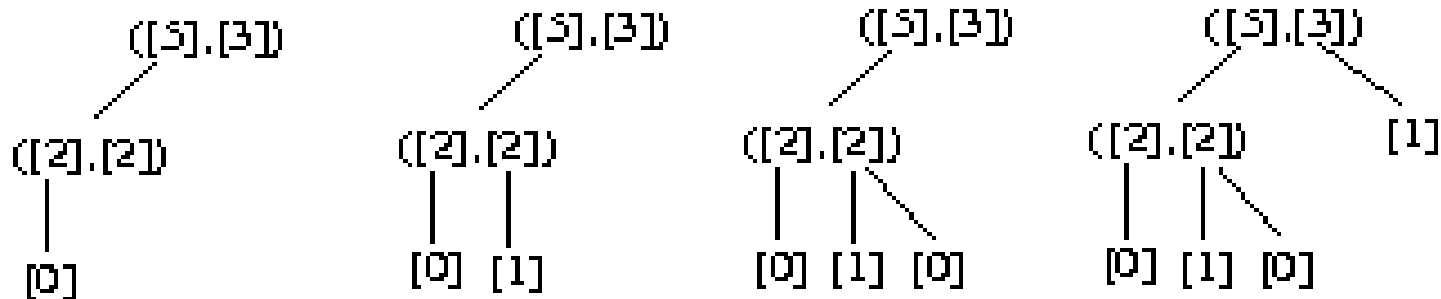
Fingerprint tree

- names in lexicographic order
- LCP

Missing: Edge coding



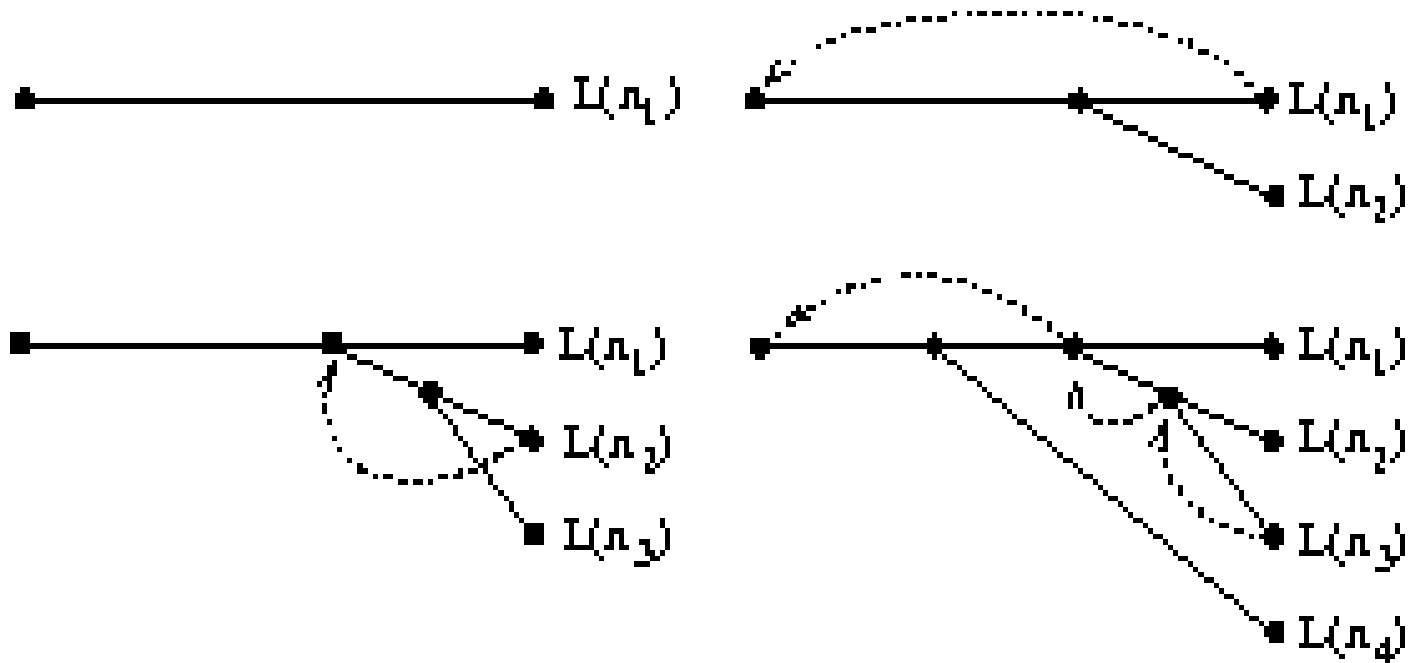
Decode an edge



Decoding is linear in the size of the edge !

Fingerprint tree

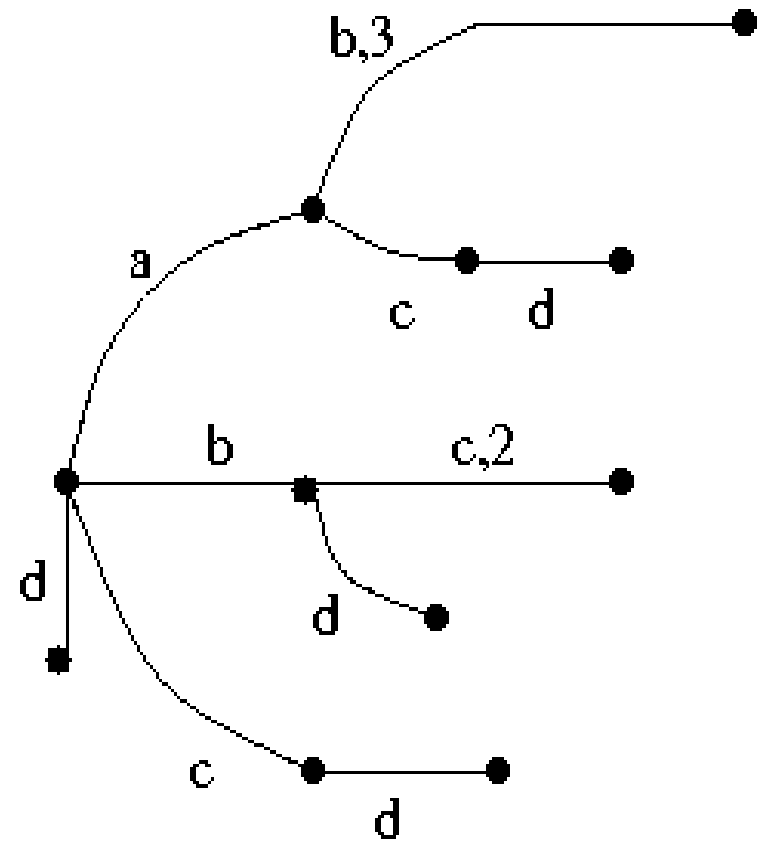
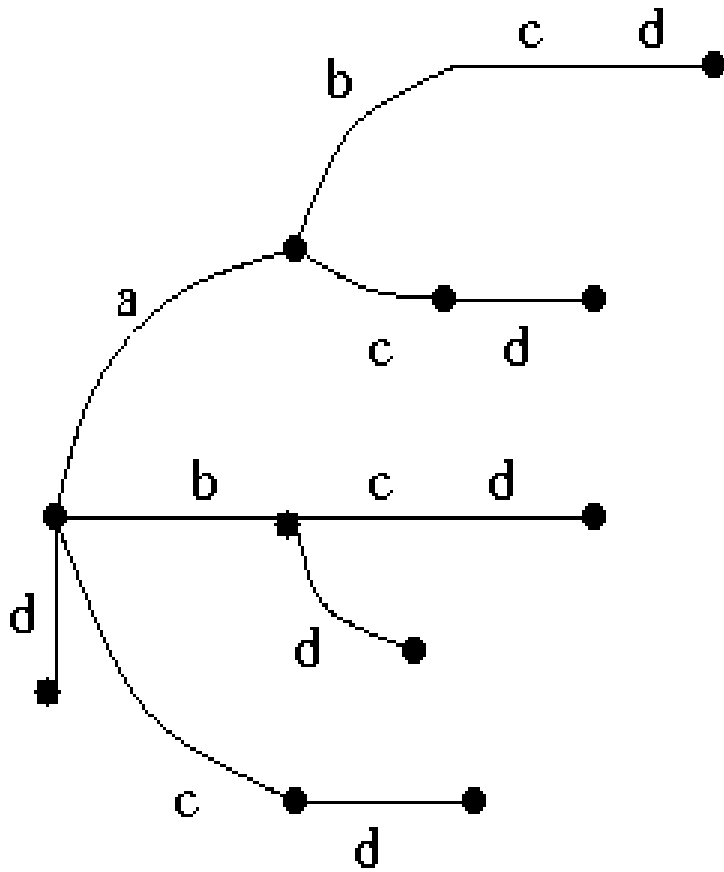
- 3 steps:
- $LCP[n_i, n_{i+1}]$ $O(|F| \log|\Sigma|)$
 - Structure of the tree $O(|F|)$
 - Compute each edge code $O(|F| \log|\Sigma|)$



Fingerprint trie

bdcad

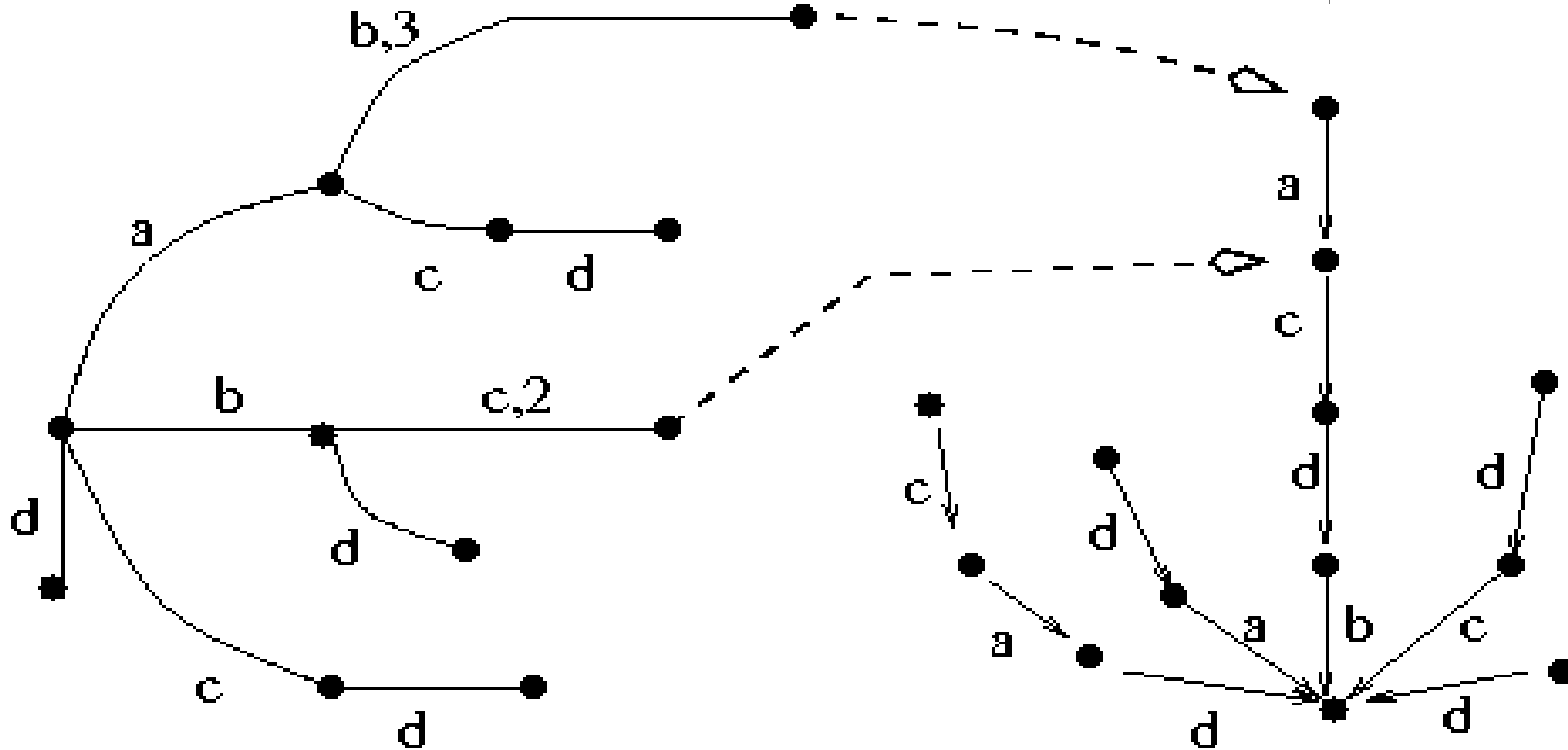
Chan *et al*, ESA 2007



Fingerprint trie

bdcad

b	d	c	a	d
d	c	a	d	
c		d		
a				



$O(|F|)$ space

$O(|F| \log |\Sigma|)$ time

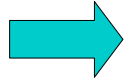
Search in $O(|f| \log(|f|/|\Sigma|))$

Open problems

 Memory space reduction

 Order ?

 Approximate fingerprint

 Distance by fingerprints

 2D fingerprints