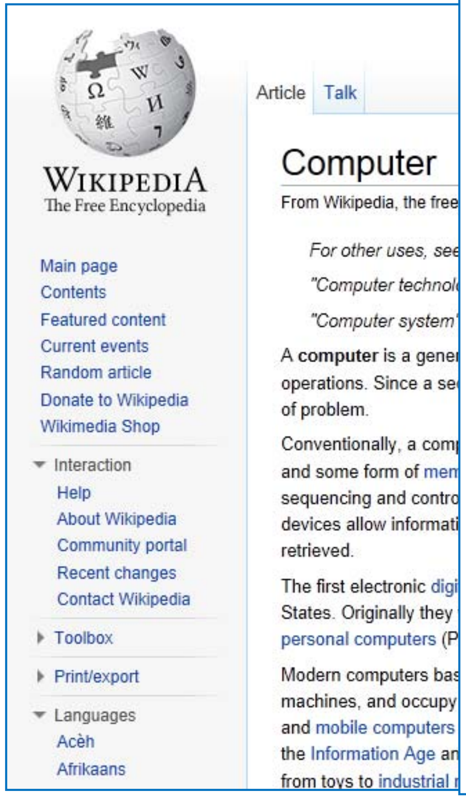
Computational Learning Theory
Frequent Substring Mining

Akihiro Yamamoto 山本 章博
http://www.iip.ist.i.kyoto-u.ac.jp/member/akihiro/
akihiro@i.kyoto-u.ac.jp

Why sequences in ML?(1)

- Sentences are strings(sequences) consisting of characters in an alphabet.



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikimedia Shop

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact Wikipedia

Toolbox

Print/export

Languages
Acèh
Afrikaans

Article Talk

Computer

From Wikipedia, the free

For other uses, see

"Computer technol

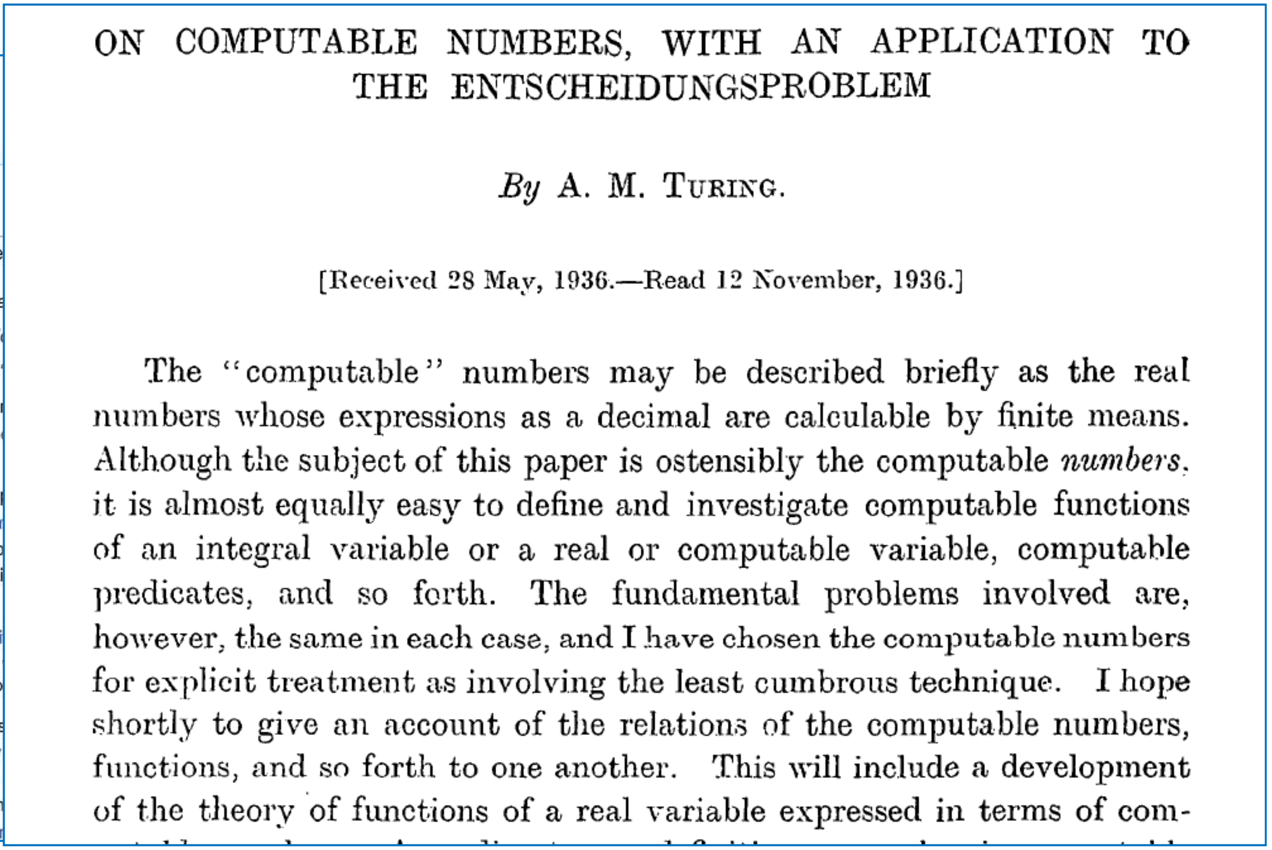
"Computer system"

A **computer** is a gener
operations. Since a se
of problem.

Conventionally, a comp
and some form of mem
sequencing and contro
devices allow informati
retrieved.

The first electronic digi
States. Originally they
personal computers (P

Modern computers bas
machines, and occupy
and mobile computers
the Information Age an
from toys to industrial r



ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

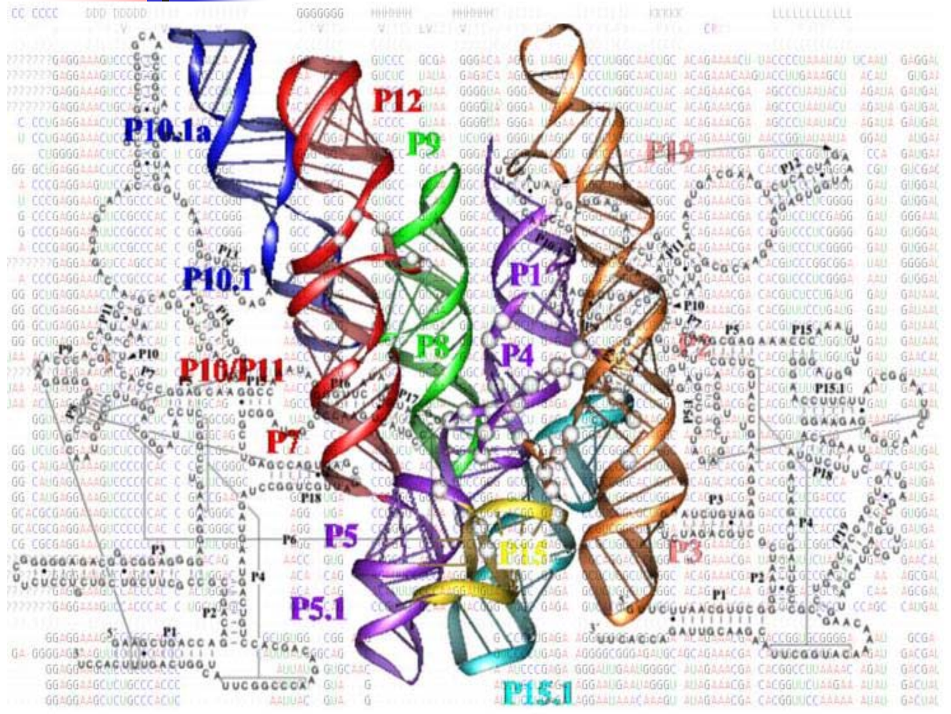
By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable *numbers*, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of com-

[Wikipedia] [Davis, M. : *The Unsolvable,* Raven Press]

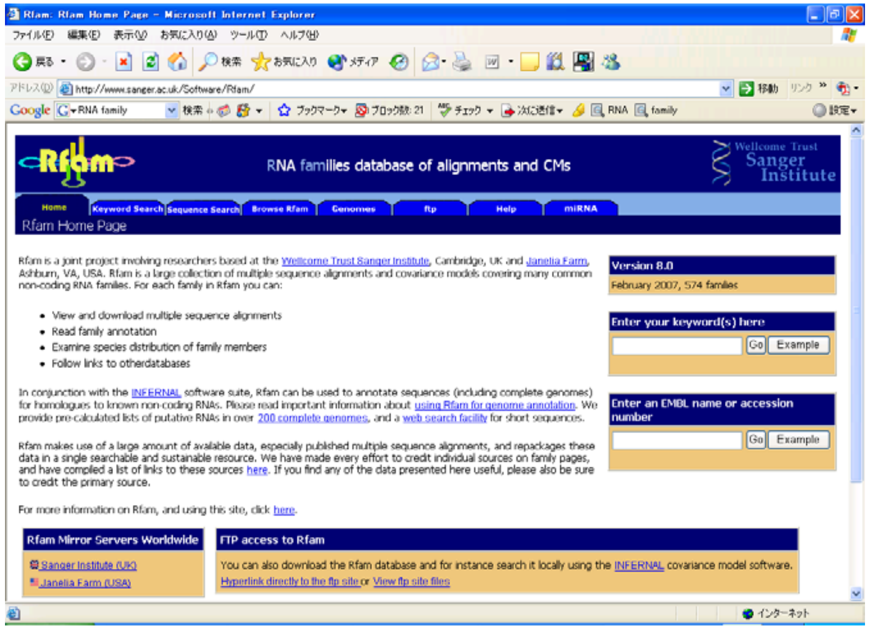
Why sequences in ML?(2)



CACAUGUACAAGACUU

5′ 3′

- Many data for academic research is now open. In particular, many string data are provided in the area of bio-informatics.



Rfam: Rfam Home Page - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
戻る 検索 お気に入り メディア
アドレス(D) http://www.sanger.ac.uk/Software/Rfam/
移動 リンク
Google RNA family 検索 ブックマーク ブロック数: 21 チェック 次に送信 RNA family 設定
Rfam
Wellcome Trust Sanger Institute
Home
Keyword Search
Sequence Search
Browse Rfam
Genomes
ftp
Help
miRNA
Go
Example
インターネット

RNA families database of alignments and CMs

Rfam Home Page

Rfam is a joint project involving researchers based at the Wellcome Trust Sanger Institute, Cambridge, UK and Janelia Farm, Ashburn, VA, USA. Rfam is a large collection of multiple sequence alignments and covariance models covering many common non-coding RNA families. For each family in Rfam you can:

- View and download multiple sequence alignments
- Read family annotation
- Examine species distribution of family members
- Follow links to otherdatabases

In conjunction with the INFERNAL software suite, Rfam can be used to annotate sequences (including complete genomes) for homologues to known non-coding RNAs. Please read important information about using Rfam for genome annotation. We provide pre-calculated lists of putative RNAs in over 200 complete genomes, and a web search facility for short sequences.

Rfam makes use of a large amount of available data, especially published multiple sequence alignments, and repackages these data in a single searchable and sustainable resource. We have made every effort to credit individual sources on family pages, and have compiled a list of links to these sources here. If you find any of the data presented here useful, please also be sure to credit the primary source.

For more information on Rfam, and using this site, click here.

Version 8.0

February 2007, 574 families

Enter your keyword(s) here

Enter an EMBL name or accession number

Rfam Mirror Servers Worldwide

Sanger Institute (UK)

Janelia Farm (USA)

FTP access to Rfam

You can also download the Rfam database and for instance search it locally using the INFERNAL covariance model software.

Hyperlink directly to the ftp site or View ftp site files

FORMALIZATION OF THE PROBLEM

Formalization of the Task

Inputs : a (long) string S

a minimum number of

appearance(threshold) $\sigma \in \mathbf{N}$

Enumerate all patterns P satisfying supp(P) $\geq \sigma$

- Here a pattern P is defined as a substring.
- The support of P means how many times P occurs in S.

Substring and Subsequence

For a string $S = a_0 a_1 \ldots a_n$

- A substring of S is $S[i,\ j] = a_i\, a_{i+1} a_{i+2} \ldots a_j$
  - The terminology "pattern matching" means to find all substrings which is identical to a given sting T.
- A subsequence of S is $a_{i_1} a_{i_2} a_{i_3} \ldots a_{i_k}$ $0 \leq i_1 < i_2 < \ldots < i_k \leq n$
  - just in the terminology "longest common subsequece"

例 For $S =$ abcabcab and $T =$ cccaacaaba,

cabc is a substring and also a subseqence of S, while

cacb is not a substring of S but a subsequence, and

a common subsequence of S and T.

Suffix of a String

- A suffix of $S = a_0 a_1 \ldots a_n$ is a subsequence $S[i, n]$ ($i = 1, 2, \ldots, n-1$) ending with a_n.
  - We assume a_n e is a special symbol $

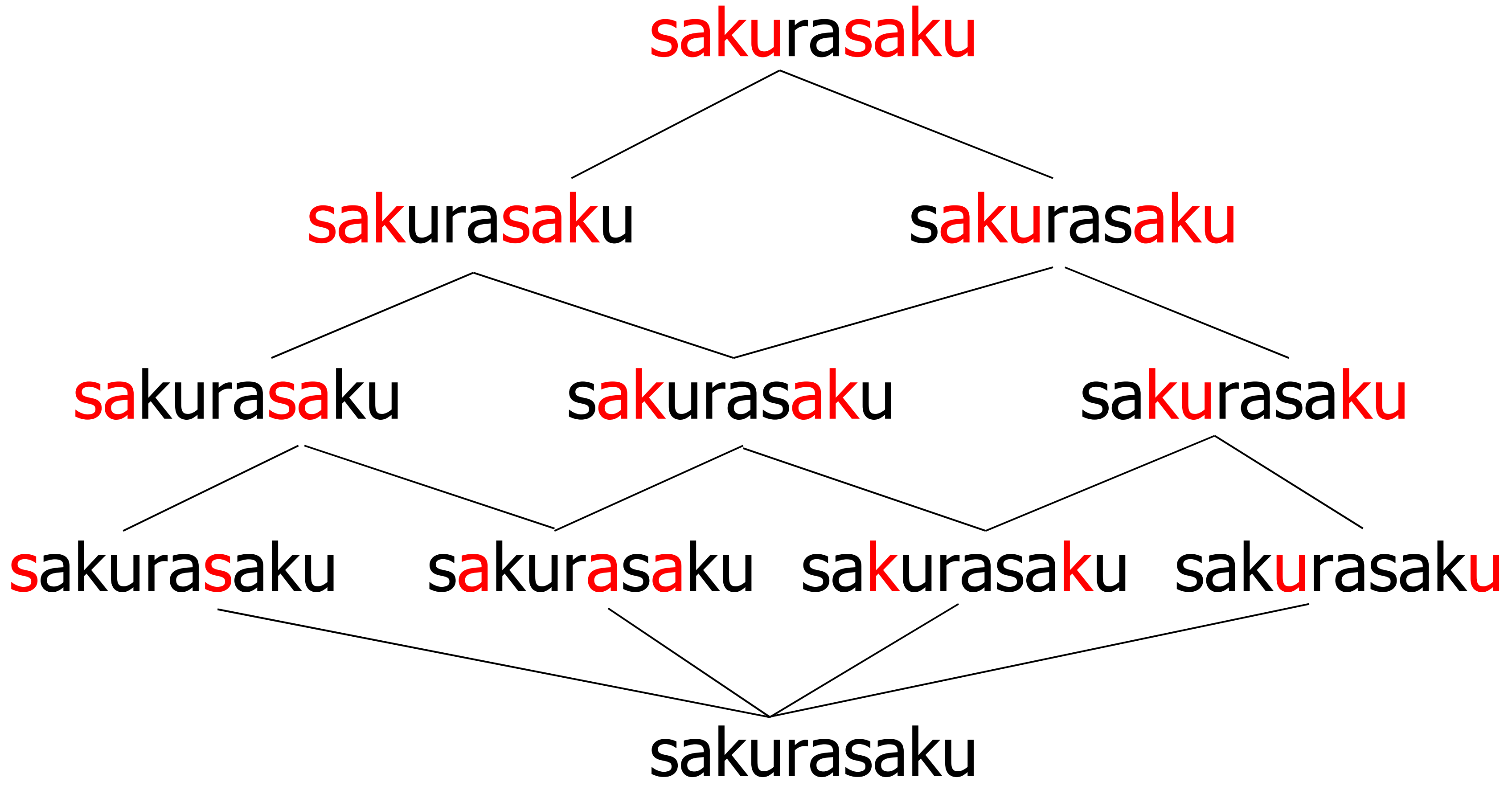
Example S = sakurasaku$

$S[0,10]$= sakurasaku$	$S[5,10]$= asaku$
$S[1,10]$= akurasaku$	$S[6,10]$= asaku$
$S[2,10]$= kurasaku$	$S[7,10]$= aku$
$S[3,10]$= urasaku$	$S[8,10]$= ku$
$S[4,10]$= rasaku$	$S[9,10]$= u$

Example (Tsuboi[2003])

$S =$ sakurasaku

minimum support $\sigma = 2$



Tsuboi’s Algorithm [Tsuboi 03]

Inputs :

a (long) string S ending with a special symbol \$

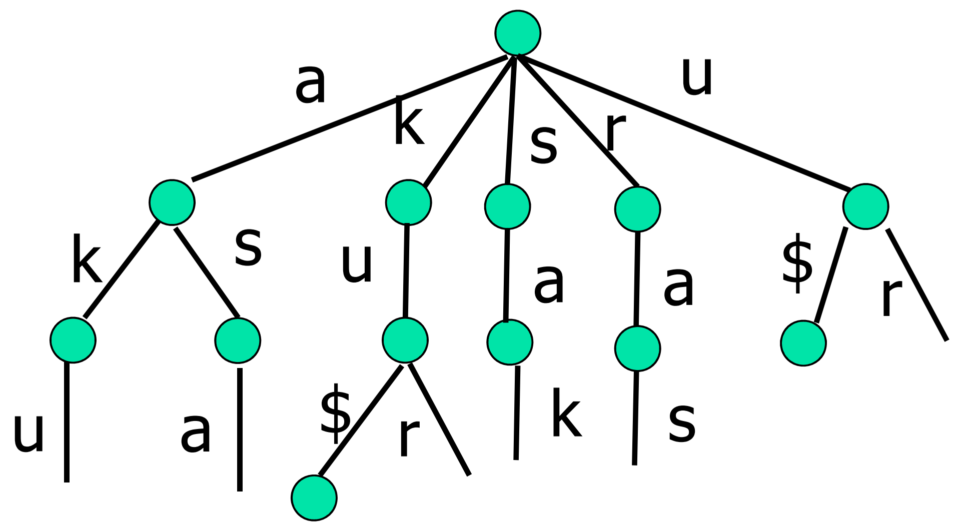
a minimum number $\sigma \in \mathbf{N}$ of appearance (threshold)

1. Enumerate all suffixes of S
2. Make three clusters of the suffixes.
3. For each cluster obtained by Step 2, apply Step 2,
   so we get a tree whose nodes has at most 3 children.

/* This is hierarchical clustering. */

Prefix Tree (Trie) of all Suffixes

- Construct the prefix tree (trie) of all suffixes $S[i, n]$ $(i = 1, 2, \ldots, n-1)$.



A Step in Hierarchal Clustering (1)

- Just like the Quick Sort Algorithm, choose a pivot v and make three clusters with the head symbol $S[i]$ of each suffix.
    - If all suffixes have the same head symbol, try to use the second character, third character,…

Example v = k

$S[i] < v$	$S[i] = v$	$S[i] > v$
akurasaku$	kurasaku$	sakurasaku$
asaku$	ku$	urasaku$
aku$		rasaku$
		sa$
		u$

A Step in Hierarchal Clustering (2)

v = k

akurasaku\$
asaku\$
aku\$

kurasaku\$
ku\$

sakurasaku\$
urasaku\$
rasaku\$
saku\$
u\$

v = s

akurasaku\$
aku\$

asaku\$

v = r

kurasaku\$
ku\$

v = s

rasaku\$

sakurasaku\$
saku\$

urasaku\$
u\$

Pruning Clusters (1)

- $\mathrm{supp}(S[i,\ j]) \geq \mathrm{supp}(S[i-k\ ,\ j])\ \ (k > 0)$
- We can delete all clusters which has suffixes less than σ.

Pruning Clusters (2)

v = k

akurasaku$
asaku$
aku$

kurasaku$
ku$

sakurasaku$
urasaku$
rasaku$
saku$
u$

v = s

akurasaku$
aku$

asaku$

v = r

kurasaku$
ku$

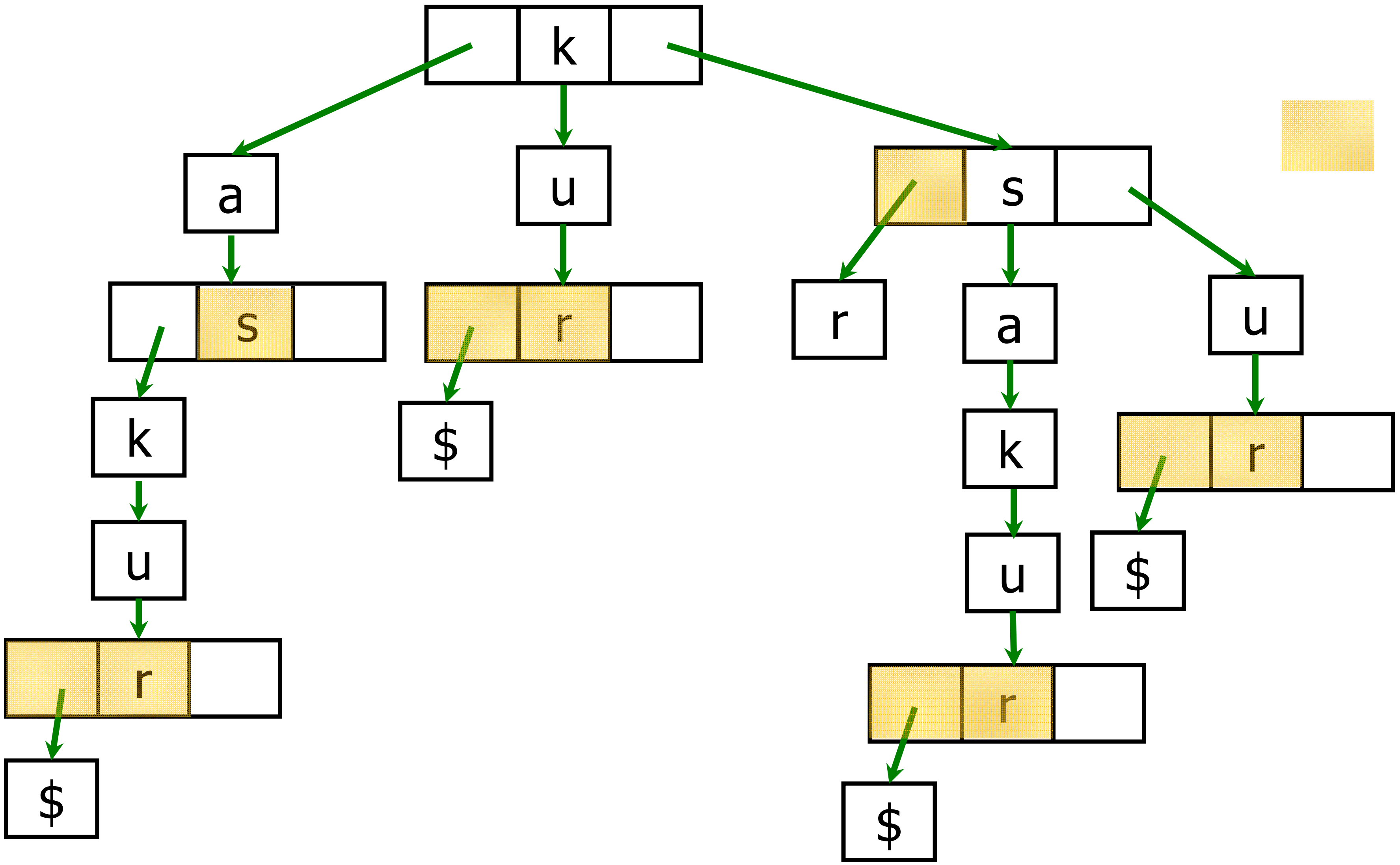
v = s

rasaku$

sakurasaku$
saku$

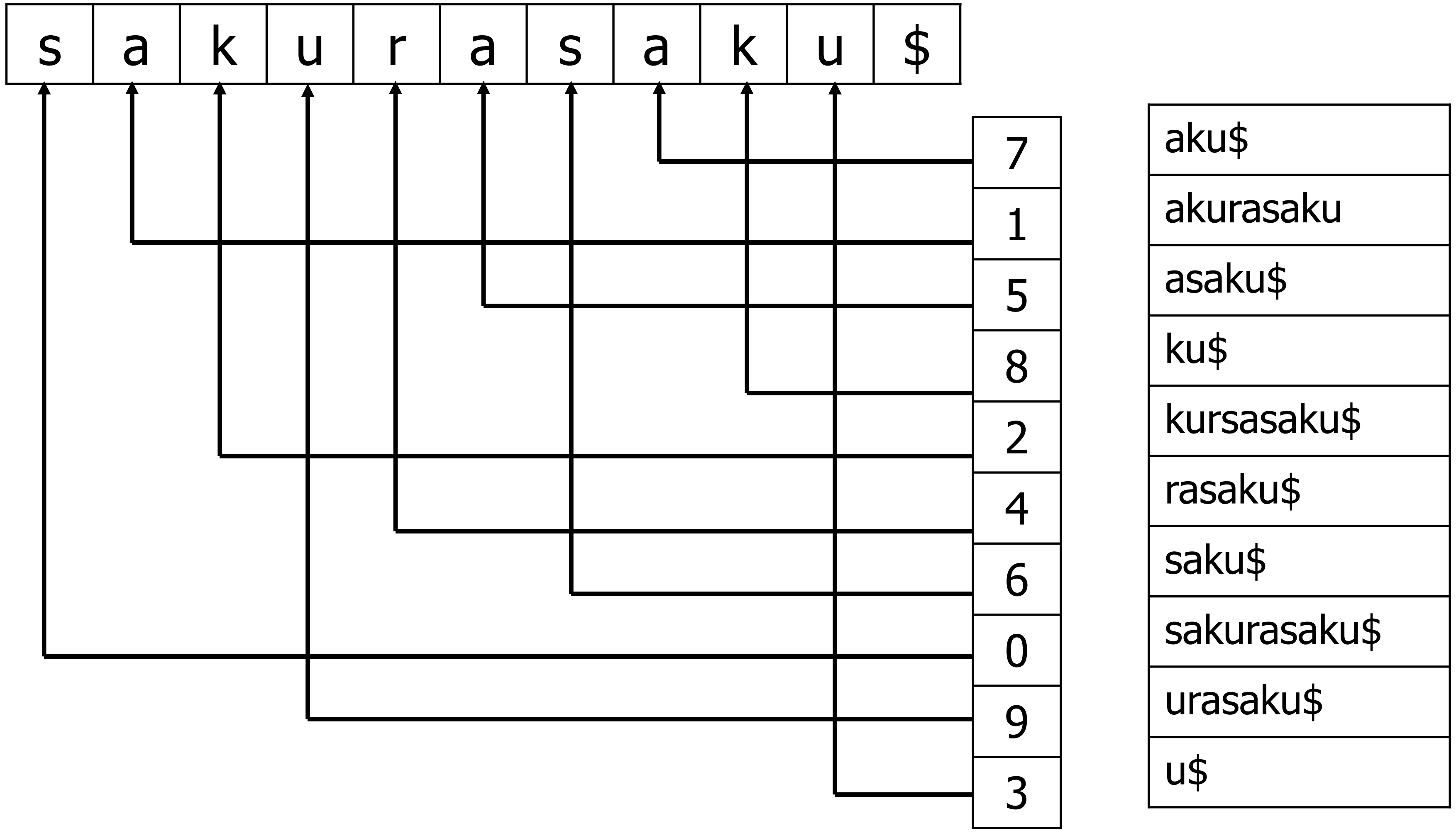
urasaku$
u$

Implementation with a Tree.



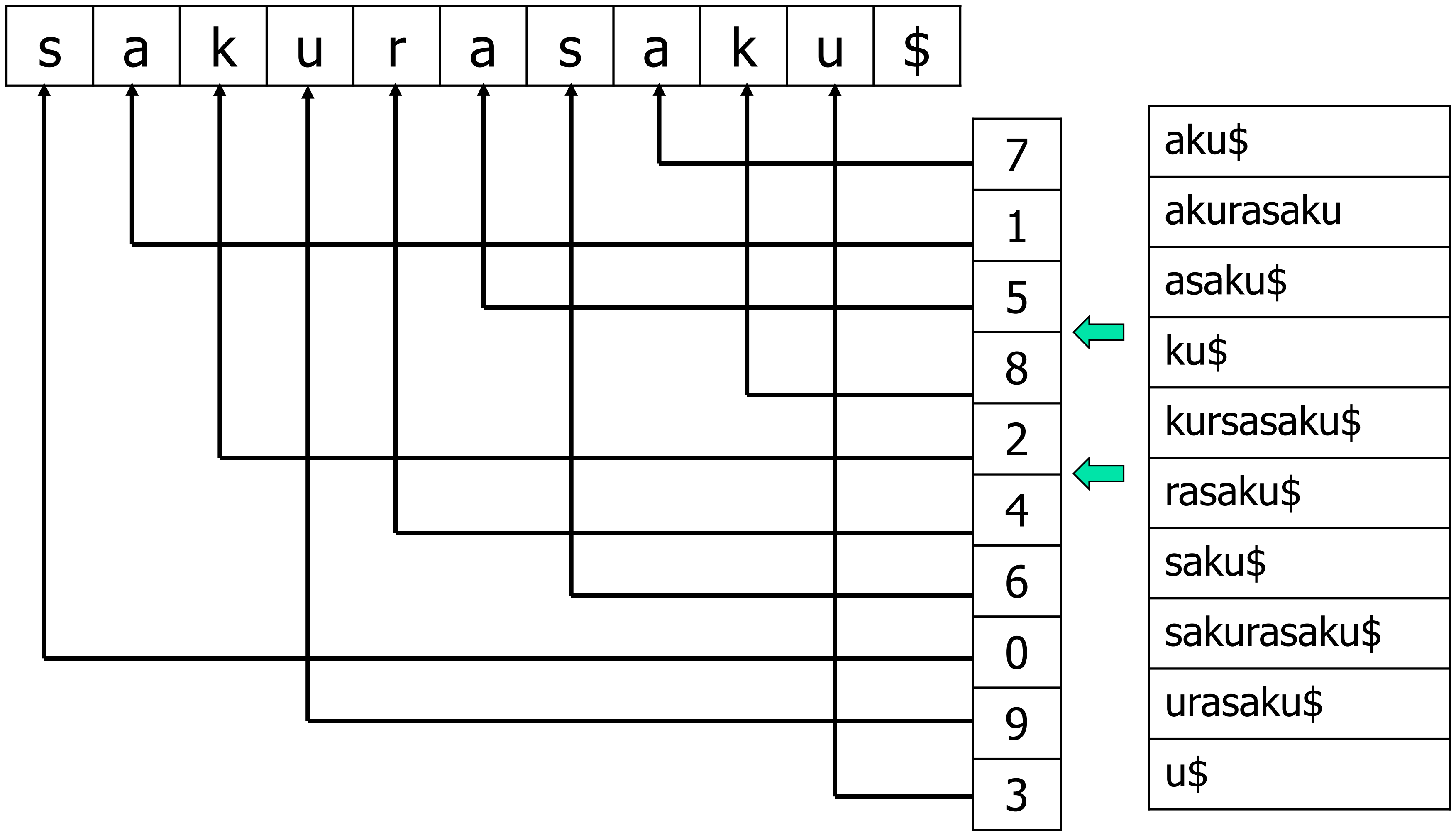
Implementation with an Array(1)

- Use the suffix array



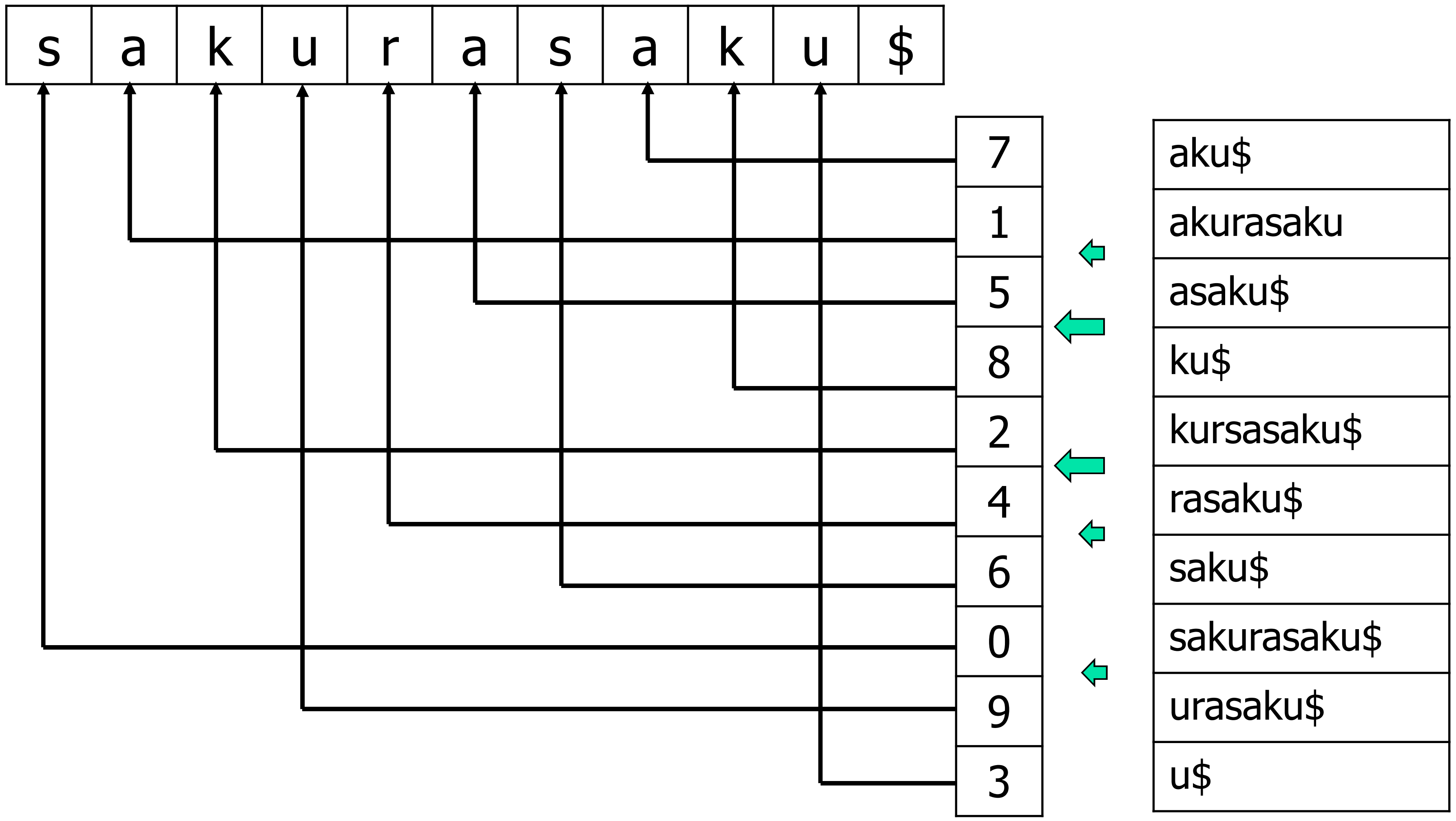
Implementation with an Array(2)

- Divide the suffix array into three recursively.



Implementation with an Array(3)

- Divide the suffix array into three recursively.



s a k u r a s a k u $
7
1
5
8
2
4
6
0
9
3
aku$
akurasaku
asaku$
ku$
kursasaku$
rasaku$
saku$
sakurasaku$
urasaku$
u$