

# Suffix Vector: A Space-Efficient Suffix Tree Representation

Krisztián Monostori, Arkady Zaslavsky

School of Computer Science and  
Software Engineering,  
Monash University

István Vajk

Department of Automation and  
Applied Informatics,  
Budapest University of Technology and Economics

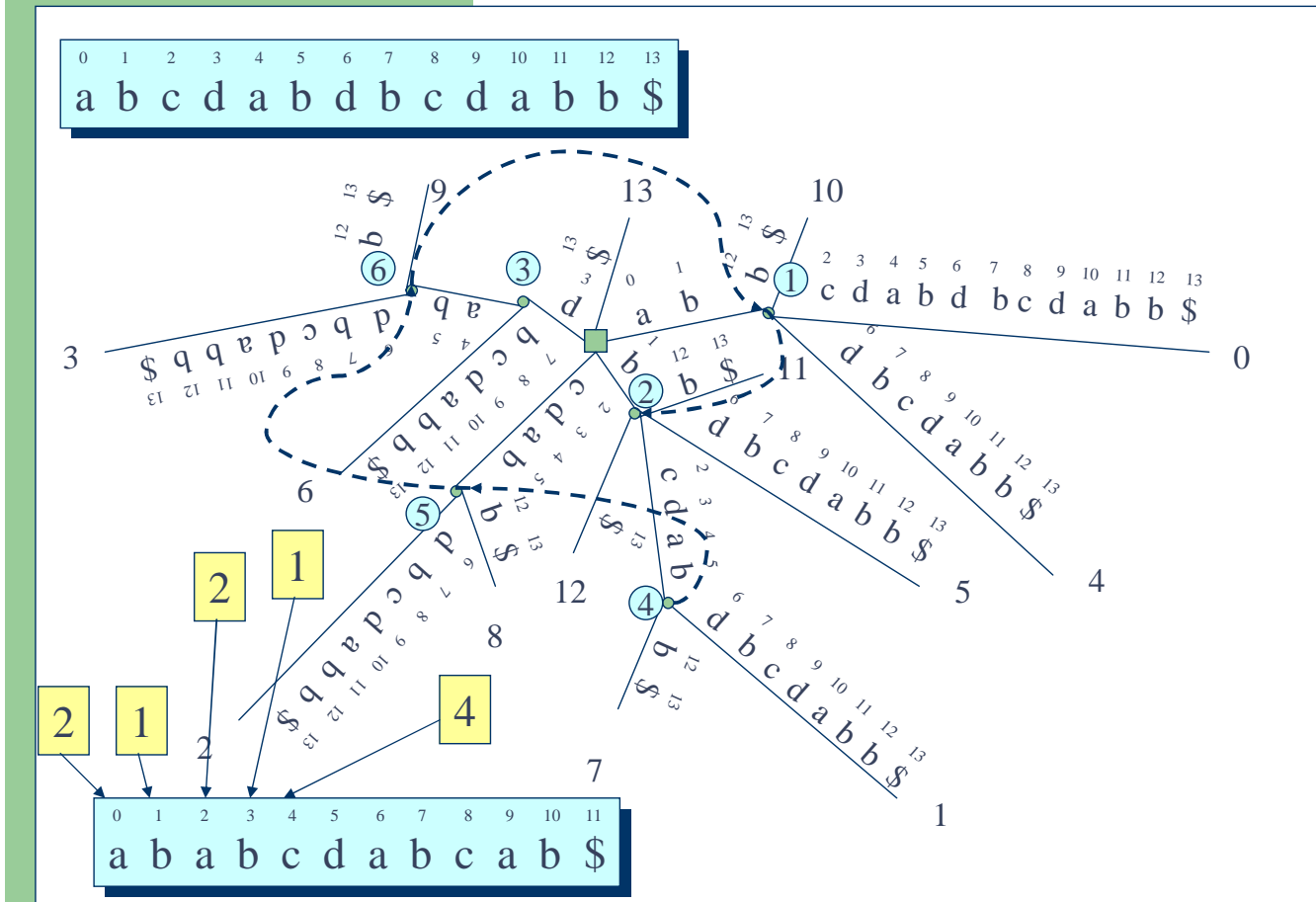


Budapest University of Technology and Economics

## Overview

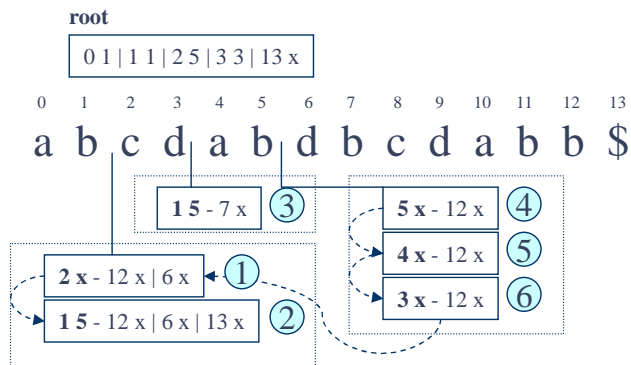
- Suffix tree representation
  - Matching statistics algorithm
- Suffix vector representation at a high level
- Physical suffix vector representation
- Performance results
- Conclusion and future work

# Suffix Tree



# Suffix Vector at a High Level

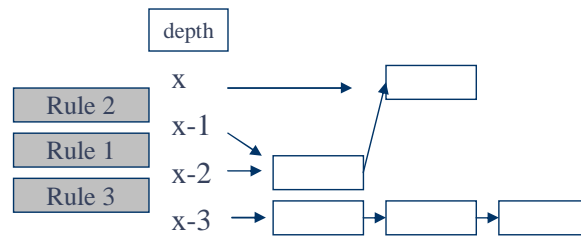
0 1 2 3 4 5 6 7 8 9 10 11 12 13  
a b c d a b d b c d a b b \$



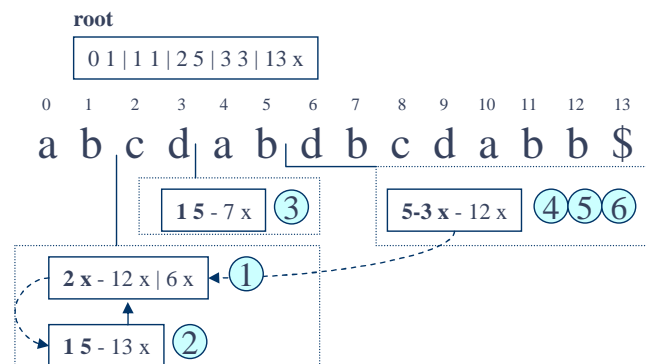
0 1 2 3 4  
c d a b b

# Redundancy

- **Rule 1:** same edges
- **Rule 2:** same edges plus extra edges
- **Rule 3:** not the same edges



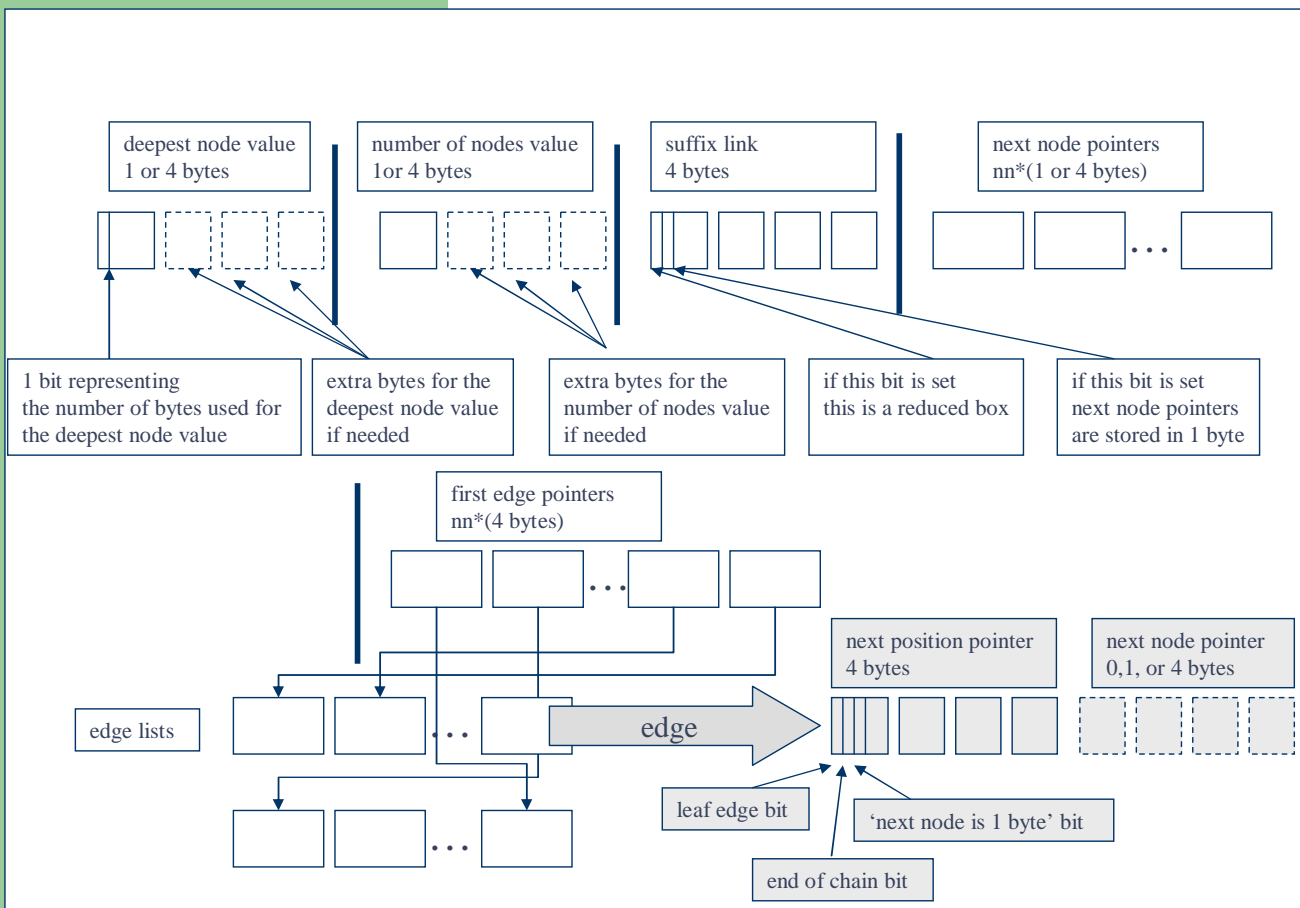
# Eliminating Redundancies



# Observations

1. The depth of the deepest node can usually fit into one byte
2. The number of nodes in a box is usually small
3. The length of an edge is usually small
4. Only one suffix link needs to be stored

## Physical Representation of a Box



## Performance Results

File name:	File size	Vector size	Bytes/symbol Suffix Vector	Bytes/symbol Kurtz
book2	610856	5561505	8.61	9.67
progl	71646	603927	8.06	10.22
K02402	38095	484229	12.56	12.59

## Internal Structure of Suffix Vectors

Type	Large nodes/ Total nodes	Boxes/ Total Nodes	Reduced nodes saved/ Total nodes
English text	23.85%	24.92%	31.05%
Formal text	17.78%	18.65%	34.88%
DNA	65.95%	50.76%	8.92%

# Conclusion and Future Work

- Most space-efficient representation
- Direct construction: extra space
- Matching statistics algorithm runs faster
- Future work
  - Other algorithms
  - Other physical representations