AN ANALYSIS OF DIGITAL SEARCH TREES BUILT ON A GENERAL SOURCE

Kanal Hun, GREYC, University of Caen

Advisor: Brigitte Vallée, GREYC
CNRS and University of Caen

EJCIM, 8 April 2013

Aim: to analyze the mean path length of DST for a general source

We are interested in its asymptotic behavior

Aim: to analyze the mean path length of DST for a general source
We are interested in its asymptotic behavior

Plan

1. Main motivation (Lempel-Ziv parsing algorithm)
2. Digital Search Tree, compare to Trie structure
3. What is a source
4. Previous results and new results
5. Methods and main steps of analysis

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the “Déjà Vu” Principle

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the "Déjà Vu" Principle

The text $110001010110111 01$

is partitioned into

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the "Déjà Vu" Principle

The text 11000101011011101

is partitioned into

(ϵ)

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the “Déjà Vu” Principle

The text $110001010110111 01$

is partitioned into

(ϵ) (1)

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme
– partition a sequence into phrases (blocks) of variable sizes
– a new block is the shortest substring not seen in the past as a phrase
– or parsing according to the "Déjà Vu" Principle

The text 11000101011011101
is partitioned into

$(\epsilon)\,(1)\,(10)$

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the “Déjà Vu” Principle

The text 11000101011011101

is partitioned into

$(\epsilon)\ (1)\ (10)\ (0)$

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme
– partition a sequence into phrases (blocks) of variable sizes
– a new block is the shortest substring not seen in the past as a phrase
– or parsing according to the "Déjà Vu" Principle

The text $110001010110111 01$

is partitioned into

(ϵ) (1) (10) (0) (01)

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the "Déjà Vu" Principle

The text 110001010110111101

is partitioned into

(ϵ) (1) (10) (0) (01) (010)

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the "Déjà Vu" Principle

The text 11000101011011101

is partitioned into

$(\epsilon)\ (1)\ (10)\ (0)\ (01)\ (010)\ (11)\ (011)\ (101)$

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme

– partition a sequence into phrases (blocks) of variable sizes

– a new block is the shortest substring not seen in the past as a phrase

– or parsing according to the "Déjà Vu" Principle

The text 11000101011011101

is partitioned into

$(\epsilon)\ (1)\ (10)\ (0)\ (01)\ (010)\ (11)\ (011)\ (101)$

The phrases are inserted

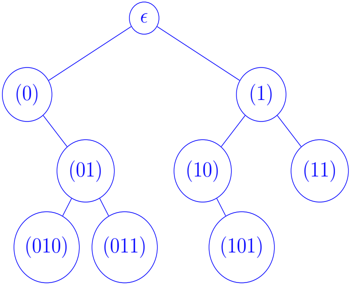
in a digital structure.

Main motivation: The Lempel et Ziv Algorithm.

The Lempel-Ziv algorithm is a **dictionary-based** scheme
– partition a sequence into phrases (blocks) of variable sizes
– a new block is the shortest substring not seen in the past as a phrase
– or parsing according to the "Déjà Vu" Principle

The text 110001010110111101
is partitioned into

$(\epsilon)\ (1)\ (10)\ (0)\ (01)\ (010)\ (11)\ (011)\ (101)$



The phrases are inserted
in a digital structure.

A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

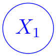
$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$

A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$

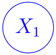


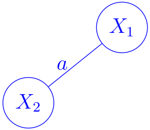
A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$



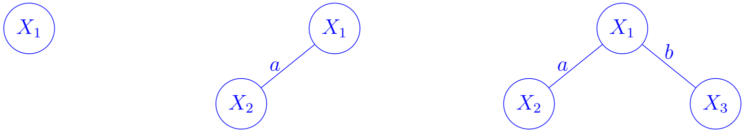


A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$

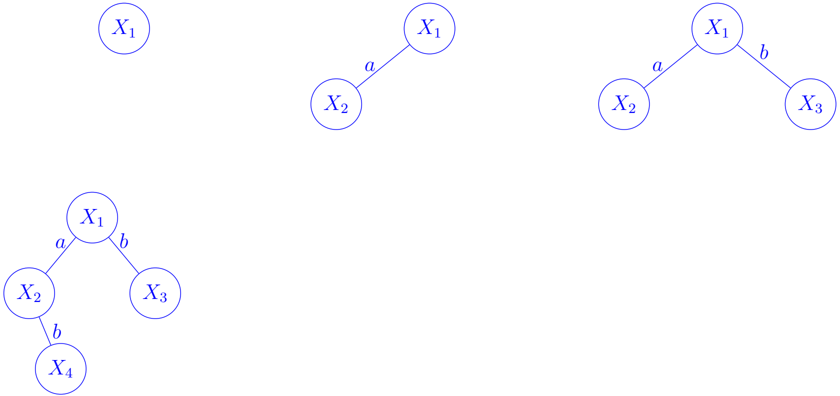


A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$

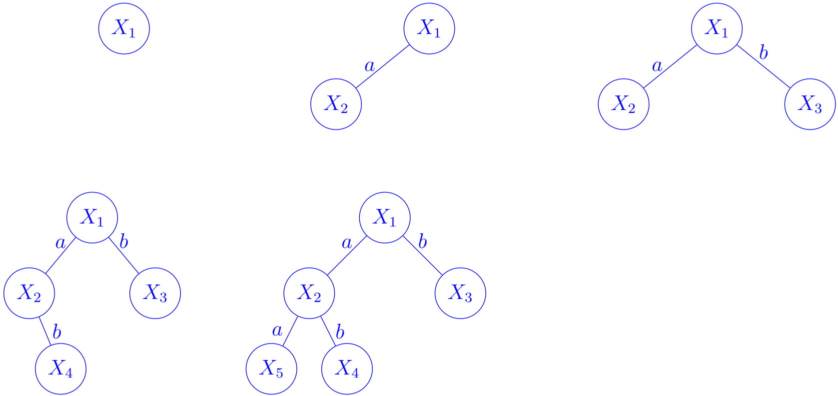


A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$

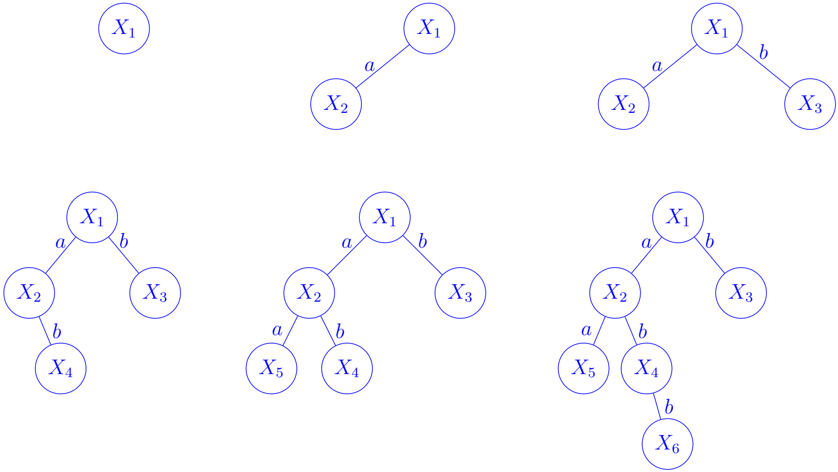


A description of DST structure (I)

A DST is a fundamental data structure in Computer Science.

Given the alphabet $\Sigma := \{a, b\}$, consider the sequence of six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba$$



Description of the DST structure(II)

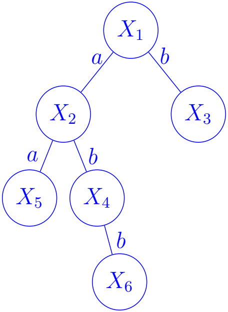
$\mathcal{X}$ = an ordered sequence of infinite words on the alphabet $\Sigma := \{a, b\}$

$$\mathcal{X} := \{X_1, X_2, \ldots, X_n\}$$

Description of the DST structure(II)

$\mathcal{X}$ = an ordered sequence of infinite words on the alphabet $\Sigma := \{a, b\}$

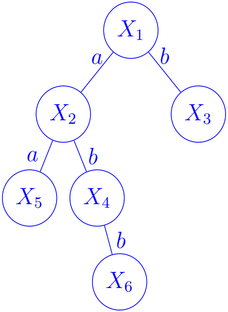
$$\mathcal{X} := \{X_1, X_2, \ldots, X_n\}$$



Description of the DST structure(II)

$\mathcal{X}$ = an ordered sequence of infinite words on the alphabet $\Sigma := \{a, b\}$

$$\mathcal{X} := \{X_1, X_2, \ldots, X_n\}$$



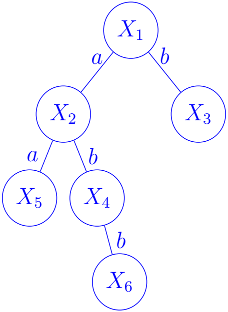
The first word of the sequence $\mathcal{X}$ is placed at the root.

Root [DST ($\mathcal{X}$)] := First ($\mathcal{X}$).

Description of the DST structure(II)

$\mathcal{X}$ = an ordered sequence of infinite words on the alphabet $\Sigma := \{a, b\}$

$$\mathcal{X} := \{X_1, X_2, \ldots, X_n\}$$



The first word of the sequence $\mathcal{X}$ is placed at the root.

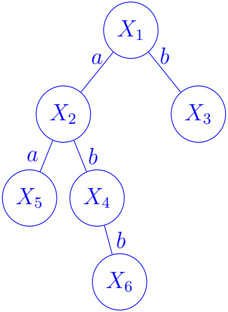
Root [DST ($\mathcal{X}$)] := First ($\mathcal{X}$).

Two subtrees built with the sequence $\mathcal{Y} := \mathcal{X} \setminus \{\mathrm{First}(\mathcal{X})\}$

Description of the DST structure(II)

$\mathcal{X}$ = an ordered sequence of infinite words on the alphabet $\Sigma := \{a, b\}$

$$\mathcal{X} := \{X_1, X_2, \ldots, X_n\}$$



The first word of the sequence $\mathcal{X}$ is placed at the root.

Root [DST ($\mathcal{X}$)] := First ($\mathcal{X}$).

Two subtrees built with the sequence $\mathcal{Y} := \mathcal{X} \setminus \{\mathrm{First}(\mathcal{X})\}$

– The left subtree contains the words of $\mathcal{Y}$ beginning with a

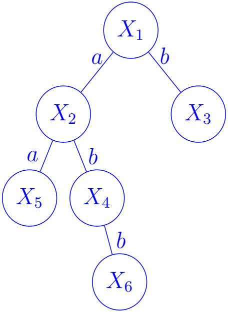
$\mathcal{Y}_{(a)}$: subsequence of $\mathcal{Y}$ formed with words beginning with a

Left [DST ($\mathcal{X}$)] := DST ($\mathcal{Y}_{(a)}$).

Description of the DST structure(II)

$\mathcal{X}$ = an ordered sequence of infinite words on the alphabet $\Sigma := \{a, b\}$

$$\mathcal{X} := \{X_1, X_2, \ldots, X_n\}$$



The first word of the sequence $\mathcal{X}$ is placed at the root.

Root [DST ($\mathcal{X}$)] := First ($\mathcal{X}$).

Two subtrees built with the sequence $\mathcal{Y} := \mathcal{X} \setminus \{\mathrm{First}(\mathcal{X})\}$

– The left subtree contains the words of $\mathcal{Y}$ beginning with a

$\mathcal{Y}_{(a)}$: subsequence of $\mathcal{Y}$ formed with words beginning with a

Left [DST ($\mathcal{X}$)] := DST ($\mathcal{Y}_{(a)}$).

– The right subtree contains the words of $\mathcal{Y}$, begin with b.

$\mathcal{Y}_{(b)}$: subsequence of $\mathcal{Y}$ formed with words beginning with b

Right [DST ($\mathcal{X}$)] := DST ($\mathcal{Y}_{(b)}$).

Comparing to the Trie Structure.

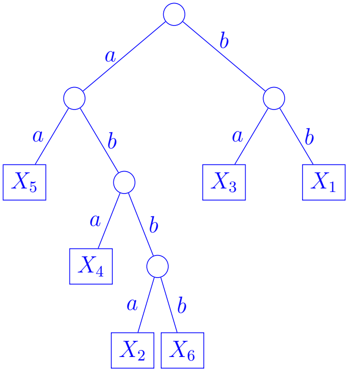
A Trie built from the six words

$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba.$$

Comparing to the Trie Structure.

A Trie built from the six words

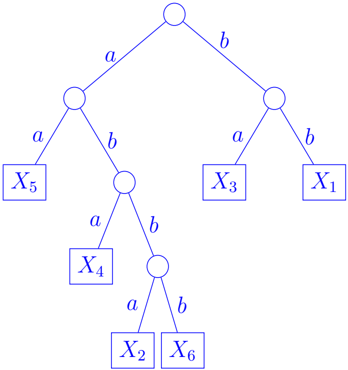
$$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba.$$



Comparing to the Trie Structure.

A Trie built from the six words

$X_1 = bbabb, X_2 = abbaa, X_3 = babba, X_4 = ababb, X_5 = aaaab, X_6 = abbba.$



The internal nodes are empty: they are only used to guide the search

The trie does not depend on the order of the words, where as DST does.

Parameters of interest: depth, profile, path length, etc.

Parameters of interest: depth, profile, path length, etc.

A full node := a node which contains a word

The depth of a node: the distance between the node and the root.

Parameters of interest: depth, profile, path length, etc.

A full node := a node which contains a word

The depth of a node: the distance between the node and the root.

Here, i am interested in:

– the path length L_n := the sum of the depths of the full nodes

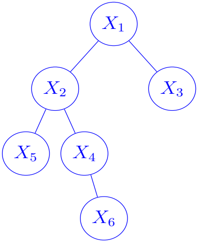
Parameters of interest: depth, profile, path length, etc.

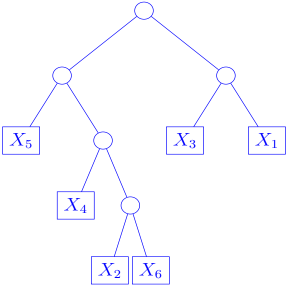
A full node := a node which contains a word

The depth of a node: the distance between the node and the root.

Here, i am interested in:

– the path length L_n := the sum of the depths of the full nodes





For DST, (internal) path length

$$2 + 2 \times 2 + 1 \times 3 = 9$$

For trie, (external) path length

$$3 \times 2 + 1 \times 3 + 2 \times 4 = 17$$

What is a source

In information theory, a source:=

a mechanism which produces symbols from alphabet Σ,

one for each time unit.

When (discrete) time evolves, a source produces (infinite) words of $\Sigma^{\mathbb{N}}$.

$$X_n \text{ the symbol emitted at time } t = n$$

What is a source

In information theory, a source:=

a mechanism which produces symbols from alphabet Σ,

one for each time unit.

When (discrete) time evolves, a source produces (infinite) words of $\Sigma^{\mathbb{N}}$.

X_n the symbol emitted at time $t = n$

A probabilistic source is defined by

sequence $(X_0, X_1, \ldots, X_n, \ldots)$ of random variables

What is a source

In information theory, a source:=

a mechanism which produces symbols from alphabet Σ,

one for each time unit.

When (discrete) time evolves, a source produces (infinite) words of $\Sigma^{\mathbb{N}}$.

X_n the symbol emitted at time $t = n$

A probabilistic source is defined by

sequence $(X_0, X_1, \ldots, X_n, \ldots)$ of random variables

Simple sources: sources with weak correlations between successive symbols

What is a source

In information theory, a source:=

a mechanism which produces symbols from alphabet Σ,

one for each time unit.

When (discrete) time evolves, a source produces (infinite) words of $\Sigma^{\mathbb{N}}$.

$$X_n \text{ the symbol emitted at time } t = n$$

A probabilistic source is defined by

sequence $(X_0, X_1, \ldots, X_n, \ldots)$ of random variables

Simple sources: sources with weak correlations between successive symbols

Memoryless source : the variables X_i are independent,

with the same distribution defined by $p_i = \Pr[X_n = i] (i \in \Sigma)$

What is a source

In information theory, a source:=

a mechanism which produces symbols from alphabet Σ,

one for each time unit.

When (discrete) time evolves, a source produces (infinite) words of $\Sigma^{\mathbb{N}}$.

X_n the symbol emitted at time $t = n$

A probabilistic source is defined by

sequence $(X_0, X_1, \ldots, X_n, \ldots)$ of random variables

Simple sources: sources with weak correlations between successive symbols

Memoryless source : the variables X_i are independent,

with the same distribution defined by $p_i = \Pr[X_n = i] (i \in \Sigma)$

Markov chain: the only dependence is between consecutive X_n's

defined by the transition matrix $p_{j|i} = \Pr[X_{n+1} = j | X_n = i]$

A general source

A general source may have many, strong correlations between its symbols.

For $w \in \Sigma^\star$, $p_w :=$ probability a word begins with the prefix w.

Then: the set $\{p_w, \quad w \in \Sigma^\star\}$ defines the source $\mathcal{S}$.

A general source

A general source may have many, strong correlations between its symbols.
For $w \in \Sigma^\star$, $p_w :=$ probability a word begins with the prefix w.

Then: the set $\{p_w, \quad w \in \Sigma^\star\}$ defines the source $\mathcal{S}$.

The entropy $h(\mathcal{S})$ relative to a probabilistic source $\mathcal{S}$ is defined as, if exists,

$$h(\mathcal{S}) := \lim_{k \to \infty} \frac{-1}{k} \sum_{w \in \Sigma^k} p_w \log p_w$$

A general source

A general source may have many, strong correlations between its symbols.
For $w \in \Sigma^\star$, $p_w :=$ probability a word begins with the prefix w.

Then: the set $\{p_w, \quad w \in \Sigma^\star\}$ defines the source $\mathcal{S}$.

The entropy $h(\mathcal{S})$ relative to a probabilistic source $\mathcal{S}$ is defined as, if exists,

$$h(\mathcal{S}) := \lim_{k\to\infty} \frac{-1}{k} \sum_{w\in\Sigma^k} p_w \log p_w$$

The analyses involve the Dirichlet series of the source, defined as

$$\Lambda(s) = \sum_{w\in\Sigma^\star} p_w^s \qquad \text{Remark:} \quad \Lambda(1) = \infty$$

Tameness of the source

Recall: the Dirichlet series of the source $\Lambda(s) = \sum_{w \in \Sigma^\star} p_w^s$

For example, memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1 - \lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^s$$

Tameness of the source

Recall: the Dirichlet series of the source $\Lambda(s) = \sum_{w \in \Sigma^\star} p_w^s$

For example, memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1 - \lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^s$$

For a general source $\mathcal{S}$, $\Lambda(s)$ can be expressed with $(I - \mathbf{P}_s)^{-1}$,
where $\mathbf{P}_s$ is an operator.

Tameness of the source

Recall: the Dirichlet series of the source $\Lambda(s) = \sum_{w \in \Sigma^\star} p_w^s$

For example, memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1 - \lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^s$$

For a general source $\mathcal{S}$, $\Lambda(s)$ can be expressed with $(I - \mathbf{P}_s)^{-1}$,
where $\mathbf{P}_s$ is an operator.

For the asymptotics of the mean path length, we need that:

– the source $\mathcal{S}$ to be "tame"

Tameness of the source

Recall: the Dirichlet series of the source $\Lambda(s) = \sum_{w \in \Sigma^\star} p_w^s$

For example, memoryless sources, with probabilities (p_i)

$$\Lambda(s) = \frac{1}{1 - \lambda(s)} \qquad \text{with} \quad \lambda(s) = \sum_{i=1}^{r} p_i^s$$

For a general source $\mathcal{S}$, $\Lambda(s)$ can be expressed with $(I - \mathbf{P}_s)^{-1}$,
where $\mathbf{P}_s$ is an operator.

For the asymptotics of the mean path length, we need that:

- the source $\mathcal{S}$ to be "tame"
- i.e, $\Lambda(s)$ has nice analytic properties
  on the left of the vertical line $\Re s = 1$.

Previous results of Trie for a general source

[Clément, Flajolet, Vallée (2001)]

Consider n words independently drawn from a general tame source. Then the mean path-length $\mathbb{E}[T_n]$ of the Trie satisfies

$$\mathbb{E}[T_n] = \frac{1}{h_{\mathcal{S}}}\, n \log n + nB_{\mathcal{S}} + n\delta_T(n) + R_n$$

The remainder term R_n depends on the "tameness" region of the source.

The "fluctuating" (small) term $\delta_T(n)$ exists when the source is periodic.

The constant term $B_{\mathcal{S}}$ is expressed with characteristics of the source.

Previous results for a DST built on a simple source

[Flajolet, Sedgewick (1986); Jacquet, Szpankowski, Tang (2001)]

Consider n words independently drawn from a simple source. Then:

Previous results for a DST built on a simple source

[Flajolet, Sedgewick (1986); Jacquet, Szpankowski, Tang (2001)]

Consider n words independently drawn from a simple source. Then:

The mean internal path-length of the Digital Search Tree satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}} n \log n + n A_{\mathcal{S}} + R_n^{[D]}$$

Previous results for a DST built on a simple source

[Flajolet, Sedgewick (1986); Jacquet, Szpankowski, Tang (2001)]

Consider n words independently drawn from a simple source. Then:
The mean internal path-length of the Digital Search Tree satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}}\, n \log n + n A_{\mathcal{S}} + R_n^{[D]}$$

To be compared to known results for a Trie

Previous results for a DST built on a simple source

[Flajolet, Sedgewick (1986); Jacquet, Szpankowski, Tang (2001)]

Consider n words independently drawn from a simple source. Then:

The mean internal path-length of the Digital Search Tree satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}} n \log n + n A_{\mathcal{S}} + R_n^{[D]}$$

To be compared to known results for a Trie

The mean external path length of the Trie satisfies

$$\mathbb{E}[T_n] = \frac{1}{h_{\mathcal{S}}} n \log n + n B_{\mathcal{S}} + R_n^{[T]}$$

Previous results for a DST built on a simple source

[Flajolet, Sedgewick (1986); Jacquet, Szpankowski, Tang (2001)]

Consider n words independently drawn from a simple source. Then:

The mean internal path-length of the Digital Search Tree satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}} n \log n + nA_{\mathcal{S}} + R_n^{[D]}$$

To be compared to known results for a Trie

The mean external path length of the Trie satisfies

$$\mathbb{E}[T_n] = \frac{1}{h_{\mathcal{S}}} n \log n + nB_{\mathcal{S}} + R_n^{[T]}$$

The difference $A_{\mathcal{S}} - B_{\mathcal{S}}$ is always negative for any simple source.

The remainder terms R_n depend on the tameness properties of the source

Previous results for a DST built on a simple source

[Flajolet, Sedgewick (1986); Jacquet, Szpankowski, Tang (2001)]

Consider n words independently drawn from a simple source. Then:

The mean internal path-length of the Digital Search Tree satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}}\, n\, \log n + n A_{\mathcal{S}} + R_n^{[D]}$$

To be compared to known results for a Trie

The mean external path length of the Trie satisfies

$$\mathbb{E}[T_n] = \frac{1}{h_{\mathcal{S}}}\, n\, \log n + n B_{\mathcal{S}} + R_n^{[T]}$$

The difference $A_{\mathcal{S}} - B_{\mathcal{S}}$ is always negative for any simple source.

The remainder terms R_n depend on the tameness properties of the source

For the binary unbiased source: $A_{\mathcal{S}} - B_{\mathcal{S}} = -\left[\frac{1}{\log 2} + \sum_{k\geq 1} \frac{1}{2^k - 1}\right]$

Main new result on DST

Consider a general source $\mathcal{S}$ which is super-tame. Then, the mean internal path-length of a digital search tree built on n words independently emitted by $\mathcal{S}$ satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}} n \log n + A_{\mathcal{S}}\, n + n\delta_D(n) + R_n^{[D]}.$$

Main new result on DST

Consider a general source $\mathcal{S}$ which is super-tame. Then, the mean internal path-length of a digital search tree built on n words independently emitted by $\mathcal{S}$ satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}} n \log n + A_{\mathcal{S}}\, n + n\delta_D(n) + R_n^{[D]}.$$

To be compared with the analog result previously obtained for tries

Main new result on DST

Consider a general source $\mathcal{S}$ which is super-tame. Then, the mean internal path-length of a digital search tree built on n words independently emitted by $\mathcal{S}$ satisfies

$$\mathbb{E}[L_n] = \frac{1}{h_{\mathcal{S}}} n \log n + A_{\mathcal{S}}\, n + n\delta_D(n) + R_n^{[D]}.$$

To be compared with the analog result previously obtained for tries

Consider a general tame source. Then, the mean path-length of a trie built on n words independently emitted by $\mathcal{S}$ satisfies

$$\mathbb{E}[T_n] = \frac{1}{h_{\mathcal{S}}} n \log n + B_{\mathcal{S}}\, n + n\delta_T(n) + R_n^{[T]}$$

We have obtained an expression for $A_{\mathcal{S}} - B_{\mathcal{S}}$ which proves that $A_{\mathcal{S}} < B_{\mathcal{S}}$

Overview of the method
Two dictionaries –algebraic and analytic–

Overview of the method
Two dictionaries –algebraic and analytic–

The source $\mathcal{S}$, its characteristics and the data structure, its recursive definition

(A)

The mixed Dirichlet series $\varpi(s)$ depends both on the source and the data structure

(B)

The mean value $\mathbb{E}[L_n]$ of the path length of the data structure when built on n words of $\mathcal{S}$

Derivation of $\varpi(s)$

exact formula from which asymptotics can be derived

Overview of the method
Two dictionaries –algebraic and analytic–

The source $\mathcal{S}$, its characteristics and the data structure, its recursive definition

(A) →

The mixed Dirichlet series $\varpi(s)$ depends both on the source and the data structure

(B) →

The mean value $\mathbb{E}[L_n]$ of the path length of the data structure when built on n words of $\mathcal{S}$

Derivation of $\varpi(s)$

exact formula from which asymptotics can be derived

(A) Combinatorial - algebraic step: true for all the sources. $\varpi(s)$, an important tool which links the properties of the source and the strategy of the data structure:

Overview of the method
Two dictionaries –algebraic and analytic–

The source $\mathcal{S}$, its characteristics and the data structure, its recursive definition

$\xrightarrow{(A)}$

The mixed Dirichlet series $\varpi(s)$ depends both on the source and the data structure

$\xrightarrow{(B)}$

The mean value $\mathbb{E}[L_n]$ of the path length of the data structure when built on n words of $\mathcal{S}$

(A): Derivation of $\varpi(s)$

(B): exact formula from which asymptotics can be derived

(A) Combinatorial - algebraic step: true for all the sources. $\varpi(s)$, an important tool which links the properties of the source and the strategy of the data structure:

$$\mathbb{E}[L_n] = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \varpi(k)$$

(B) Analytic asymptotic step: depends on the tameness of the source or the analytic properties of $\varpi(s)$.

The exact value of the mean path length of DST

$$\mathbb{E}[L_n] = \sum_{\ell=2}^{n} (-1)^{\ell} \binom{n}{\ell} \varpi(\ell) \qquad \text{with} \qquad \varpi(s) := \sum_{v \in \Sigma^{\star}} \delta(v) p_v^s$$

The exact value of the mean path length of DST

$$\mathbb{E}[L_n] = \sum_{\ell=2}^{n} (-1)^\ell \binom{n}{\ell} \varpi(\ell) \qquad \text{with} \qquad \varpi(s) := \sum_{v \in \Sigma^\star} \delta(v) p_v^s$$

$$\delta(v) = \frac{1}{p_v} \sum_{w \geq v} p_w \prod_{\substack{\alpha \leq w, \\ \alpha \neq v}} \frac{1}{1 - p_v p_\alpha^{-1}},$$

Main steps of analysis of the mean path length

Main steps of analysis of the mean path length

+ Basic recurrence of the mean path length

Main steps of analysis of the mean path length

+ Basic recurrence of the mean path length

+ Poisson generating function transforms the recurrence into functional differential equation

Main steps of analysis of the mean path length

+ Basic recurrence of the mean path length
+ Poisson generating function transforms the recurrence into functional differential equation
+ Solve the functional differential equation

$$\frac{d}{dz}B^{(w)}(z) + B^{(w)}(z) = z + \sum_{i\in\Sigma} B^{(w.i)}(q_{i|w}z).$$

Main steps of analysis of the mean path length

+ Basic recurrence of the mean path length
+ Poisson generating function transforms the recurrence into functional differential equation
+ Solve the functional differential equation

$$\frac{d}{dz}B^{(w)}(z) + B^{(w)}(z) = z + \sum_{i\in\Sigma} B^{(w.i)}(q_{i|w}z).$$

  – Laplace transform to obtain the exact expression

Main steps of analysis of the mean path length

+ Basic recurrence of the mean path length
+ Poisson generating function transforms the recurrence into functional differential equation
+ Solve the functional differential equation

$$\frac{d}{dz}B^{(w)}(z) + B^{(w)}(z) = z + \sum_{i\in\Sigma} B^{(w.i)}(q_{i|w}z).$$

  – Laplace transform to obtain the exact expression
  – Mellin to provide an alternative expression and information about the singularities

Main steps of analysis of the mean path length

+ Basic recurrence of the mean path length

+ Poisson generating function transforms the recurrence into functional differential equation

+ Solve the functional differential equation

$$\frac{d}{dz}B^{(w)}(z) + B^{(w)}(z) = z + \sum_{i \in \Sigma} B^{(w.i)}(q_{i|w}z).$$

– Laplace transform to obtain the exact expression

– Mellin to provide an alternative expression and information about the singularities

+ Under some hypotheses on the source, apply the Rice formula to obtain the asymptotic value

(:–) Merci de votre attention

Various extensions

Main result. For a general “hyper-tame” source, the typical depth of a DST follows an asymptotic gaussian law.

– To be done :

  – Return to the analysis of the Lempel Ziv algorithm.

  – Make precise all the tameness properties,

  (tame, super-tame, hyper-tame)

  even in the case of simple sources