Dichotomic Selection on words : towards a probabilistic analysis

Brigitte Vallée

GREYC (CNRS et Université de Caen)

Joint work with Ali Akhavi, Julien Clément, Loïck Lhote

Work yet in progress !

AofA 2019

Marseille, June 2019

Plan of the talk

I. Dichotomic selection on words:

an alternative to the Trie approach?

Description of the algorithms.

Plan of the talk

I. Dichotomic selection on words:

an alternative to the Trie approach?

Description of the algorithms.

II. Towards a probabilistic analysis:

a new parameter, a new data structure.

Proof of the quasi -optimality of a DICHO-SELECT algorithm.

I. Dichotomic selection on words:

an alternative to the Trie approach?

Description of the algorithms.

General dichotomic principles

– A list $L = (L_1, \ldots, L_n)$ of n distinct values in the increasing order.

– Two guards – an infimum L_0 and a supremum L_{n+1}– are added to the list L.

– It involves the function $\mathrm{mid} : (b, e) \mapsto \lfloor (b + e)/2 \rfloor$

– It performs a binary search in the dichotomic tree

– On the input $x \notin L$, it determines the index

$$i \in [1\,..\,n+1] \text{ such that } L_{i-1} < x < L_i$$

General dichotomic principles

– A list $L = (L_1, \ldots, L_n)$ of n distinct values in the increasing order.

– Two guards – an infimum L_0 and a supremum L_{n+1}– are added to the list L.

– It involves the function $\text{mid} : (b, e) \mapsto \lfloor (b + e)/2 \rfloor$

– It performs a binary search in the dichotomic tree

– On the input $x \notin L$, it determines the index

$$i \in [1\,..\,n+1] \text{ such that } L_{i-1} < x < L_i$$

DICHO-SELECT (L, x);

$b := 0; e := n + 1$

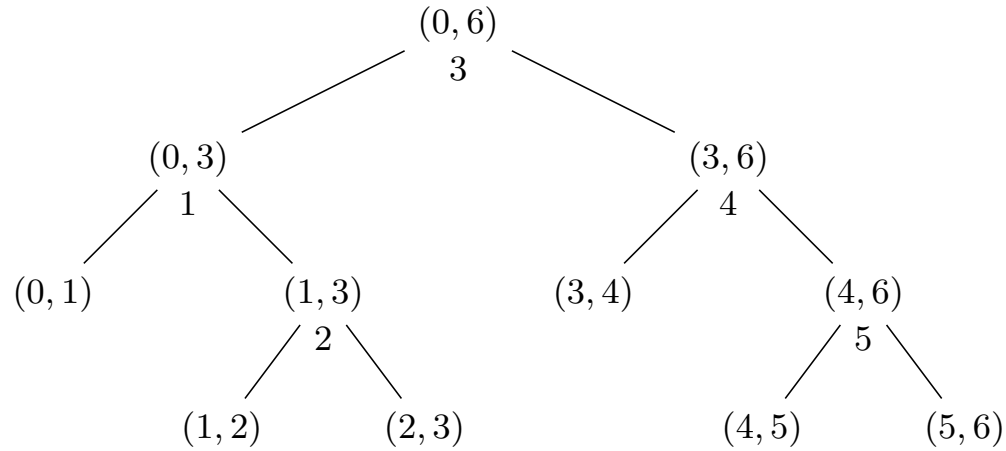
While $b + 1 < e$ **do**

$m := \text{mid}(b, e)$

If $L_m < x$ **then** $b := m$

else $e := m$;

Return (b, e).

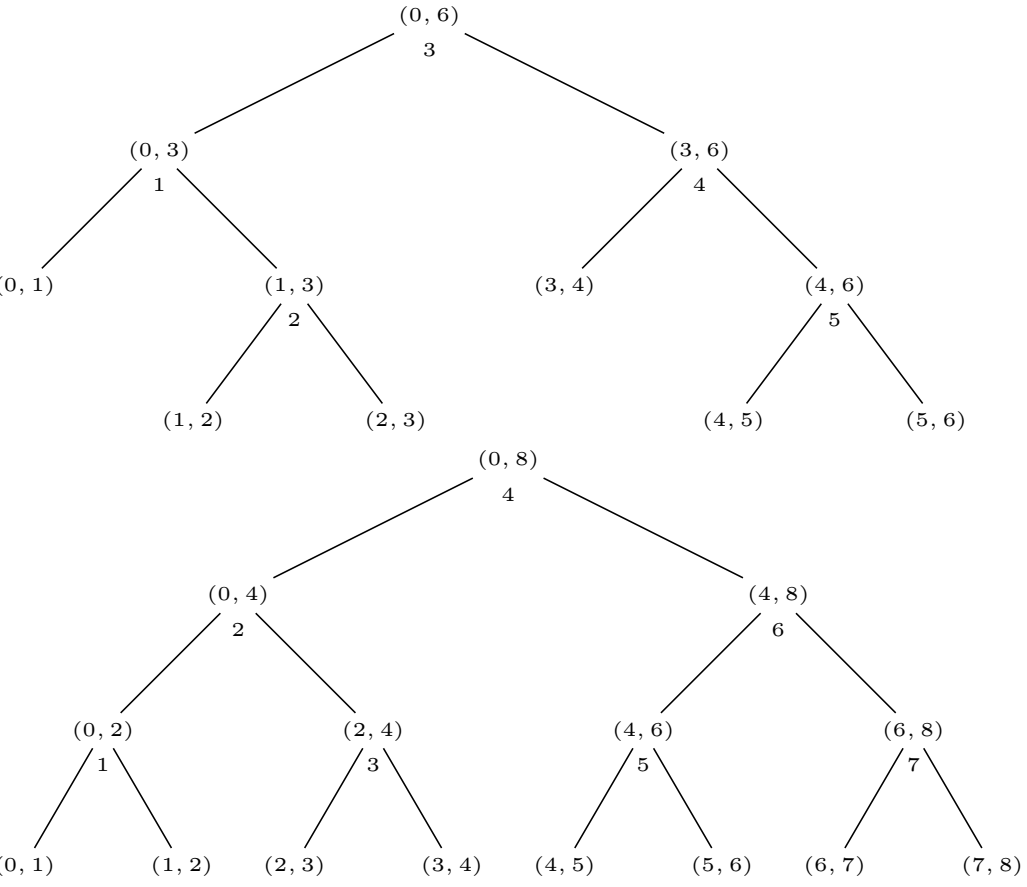


The node (b, e) contains the key $\text{mid}(b, e)$

The execution builds the path $\Pi(L, x)$ of the visited nodes

Its external node $(i - 1, i)$ is such that $L_{i-1} < x < L_i$

General dichotomic principles



Algorithm $\rightarrow$

A binary search in the dichotomic tree

Execution $\rightarrow$ a path $\Pi(L, x)$ in the tree

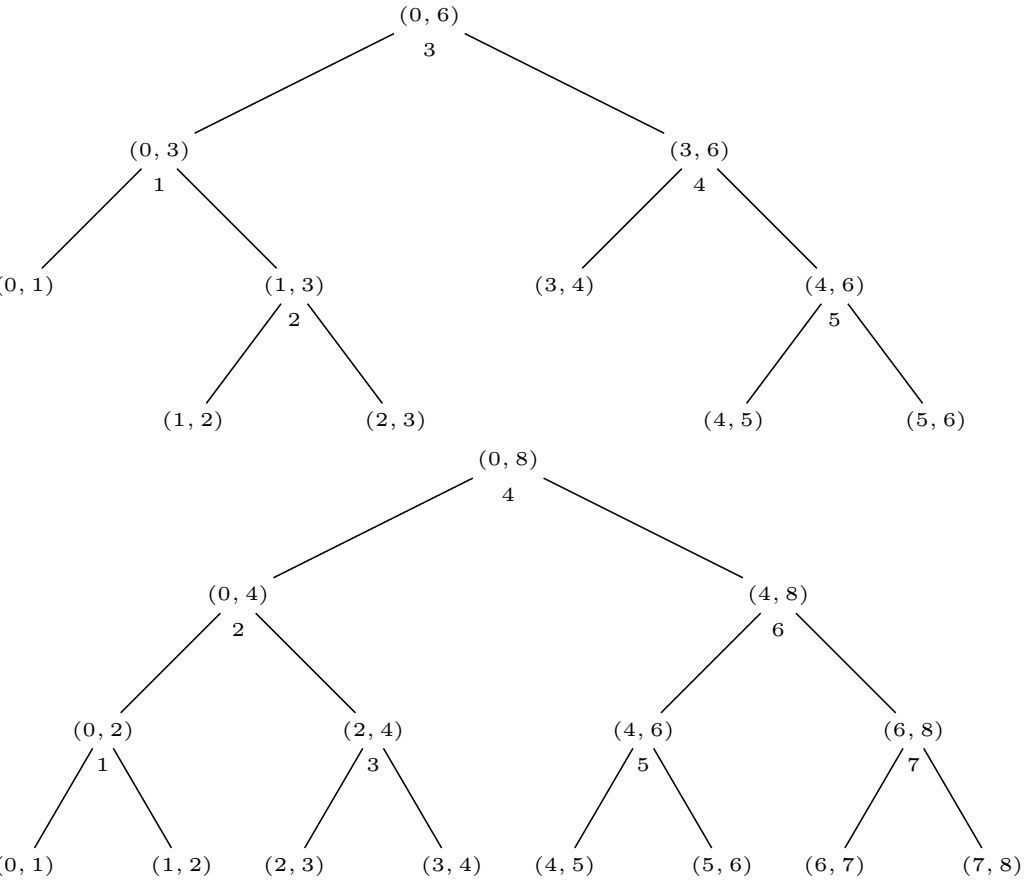
The internal part $\underline{\Pi}(L, x)$ of the path

The external node $(i-1, i)$

is such that $L_{i-1} < x < L_i$.

A key comparison at each internal node

General dichotomic principles



Algorithm $\rightarrow$

A binary search in the dichotomic tree

Execution $\rightarrow$ a path $\Pi(L, x)$ in the tree

The internal part $\underline{\Pi}(L, x)$ of the path

The external node $(i-1, i)$

is such that $L_{i-1} < x < L_i$.

A key comparison at each internal node

Property. The total number $G(L, x)$ of key comparisons of DICHO-SELECT(L, x) is equal to the number of nodes $|\underline{\Pi}(L, x)|$ of the internal path $\underline{\Pi}(L, x)$.

For $|L| = n \in [2^{p-1}, 2^p - 1]$, one has $|\underline{\Pi}(L, x)| \in \{p-1, p\}$.

For $|L| = n = 2^p - 1$, one has $|\underline{\Pi}(L, x)| = p$ for any x.

Dichotomic selection for words

(Infinite) words produced by the same source on an alphabet Σ.

Comparison between words: based on the lexicographic order.

- $\Gamma(x, y)$ is the longest common prefix between two words x and y
- $\gamma(x, y)$ is its length, often called the coincidence;
- Comparing x and y needs $\gamma(x, y) + 1$ symbol comparisons.

DICHO-SELECT(L, x) easily adapted to the context.

- Shorthand notations $\gamma[b, e] := \gamma(L_b, L_e) \quad \gamma[x, j] := \gamma(x, L_j)$, etc....

Dichotomic selection for words

(Infinite) words produced by the same source on an alphabet Σ.

Comparison between words: based on the lexicographic order.

$\Gamma(x, y)$ is the longest common prefix between two words x and y

$\gamma(x, y)$ is its length, often called the coincidence;

Comparing x and y needs $\gamma(x, y) + 1$ symbol comparisons.

DICHO-SELECT(L, x) easily adapted to the context.

Shorthand notations $\gamma[b, e] := \gamma(L_b, L_e) \quad \gamma[x, j] := \gamma(x, L_j)$, etc....

WORD-DICHO-SELECT (L, x);

$b := 0; e := n + 1;$

While $b + 1 < e$

do $m := \text{mid}(b, e);$

$\ell := 0;$

While $x[\ell] = L_m[\ell]$

do $\ell := \ell + 1;$

If $L_m[\ell] < x[\ell]$ **then** $b := m$

else $e := m$;

Return (b, e).

A key-comparison between L_m and x

replaced by a red block of symbol comparisons

of length $\gamma[m, x] + 1$

Dichotomic selection for words

(Infinite) words produced by the same source on an alphabet Σ.

Comparison between words: based on the lexicographic order.

$\Gamma(x, y)$ is the longest common prefix between two words x and y

$\gamma(x, y)$ is its length, often called the coincidence;

Comparing x and y needs $\gamma(x, y) + 1$ symbol comparisons.

DICHO-SELECT(L, x) easily adapted to the context.

Shorthand notations $\gamma[b, e] := \gamma(L_b, L_e)$ $\gamma[x, j] := \gamma(x, L_j)$, etc....

WORD-DICHO-SELECT (L, x);

$b := 0; e := n + 1;$

While $b + 1 < e$

do $m := \mathrm{mid}(b, e);$

$\ell := 0;$

While $x[\ell] = L_m[\ell]$

do $\ell := \ell + 1;$

If $L_m[\ell] < x[\ell]$ **then** $b := m$

else $e := m$;

Return (b, e).

A key-comparison between L_m and x replaced by a red block of symbol comparisons of length $\gamma[m, x] + 1$

Proposition. The total number $W(L, x)$ of symbol comparisons performed by DICHO-SELECT (L, x) is

$$W(L, x) = \sum_{(b,e)\in\underline{\Pi}(L,x)} (\gamma[m, x] + 1)$$

with $m := \mathrm{mid}(b, e)$

$\underline{\Pi}(x, L) :=$ the internal part of $\Pi(L, x)$

A detour via the trie structure:
the main tree structure for comparing words

Definition. The trie $T(L)$ is defined for the alphabet $\Sigma := \{\mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_r\}$

– for $|L| \geq 2$, with the recursive rule $\quad T(L) = (T(L \setminus \mathtt{a}_1), \ldots, T(L \setminus \mathtt{a}_r))$
where $L \setminus \mathtt{u} :=$ the subset of L of words that start with the symbol $\mathtt{u}$ stripped of $\mathtt{u}$;

– for $|L| < 1$ the recursion is halted

$$T(L) = X \text{ for } L = \{X\} \text{ and } T(L) = \emptyset \text{ if } L = \emptyset.$$

A detour via the trie structure:
the main tree structure for comparing words

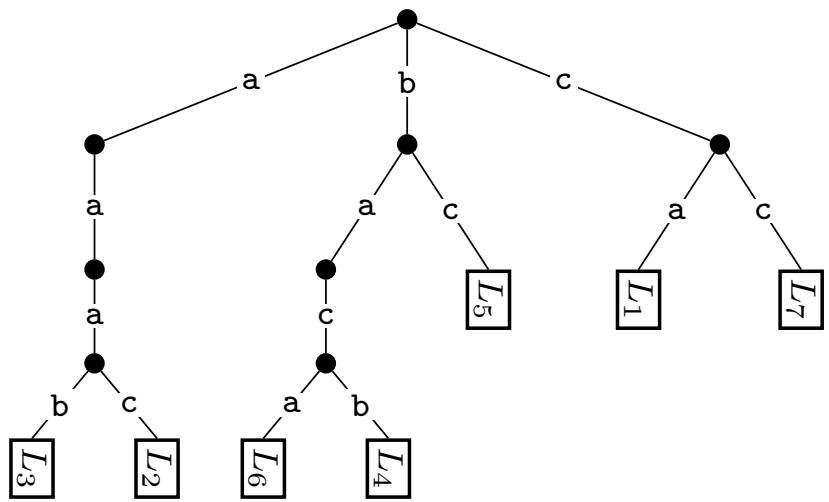
Definition. The trie $T(L)$ is defined for the alphabet $\Sigma := \{\mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_r\}$

– for $|L| \geq 2$, with the recursive rule $\quad T(L) = (T(L \setminus \mathtt{a}_1), \ldots, T(L \setminus \mathtt{a}_r))$
where $L \setminus \mathtt{u} :=$ the subset of L of words that start with the symbol $\mathtt{u}$ stripped of $\mathtt{u}$;

– for $|L| < 1$ the recursion is halted

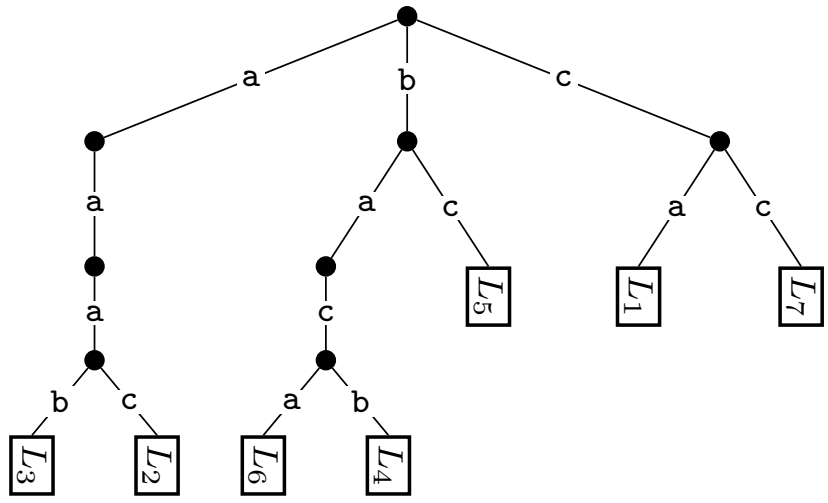
$$T(L) = X \text{ for } L = \{X\} \text{ and } T(L) = \emptyset \text{ if } L = \emptyset.$$

L_1	cacccaaac...
L_2	aaacacbac...
L_3	aaabcbbca...
L_4	bacbaabcc...
L_5	bcabbbbbc...
L_6	bacaabbba ...
L_7	ccbacbcbb...



The trie structure

L_1	cacccaaac...
L_2	aaacacbac...
L_3	aaabcbbca...
L_4	bacbaabcc...
L_5	bcabbbbbc...
L_6	bacaabbba ...
L_7	ccbacbcbb...



– It maintains the minimal prefix set (written in blue) needed to distinguish the elements of L.

– It sorts the list L.

Without any prior knowledge about the set of the words, this is the most efficient tool for comparing words.

Proposition. The minimal number of symbol comparisons needed to solve the selection problem $\textsc{Select}\ (L, x)$ (without prior knowledge on any list L) equals the branch length $|B(L, x)|$ of the trie $T(L \cup \{x\})$ leading to the external node associated with x.

Trie and Dichotomy (I)

Proposition. The minimal number of symbol comparisons needed to solve the selection problem $\textsc{Select}\ (L, x)$ (without prior knowledge on any list L) equals the branch length $|B(L, x)|$ of the trie $T(L \cup \{x\})$ leading to the external node associated with x.

Trie and Dichotomy (I)

Proposition. The minimal number of symbol comparisons needed to solve the selection problem $\textsc{Select}\ (L, x)$ (without prior knowledge on any list L) equals the branch length $|B(L, x)|$ of the trie $T(L \cup \{x\})$ leading to the external node associated with x.

For $|L| = n \to \infty$, in a quite general probabilistic framework, in terms of number of symbol comparisons, and on average, (well known for AofA people)

– the asymptotic cost of building the trie $T(L)$ is $[1/h(\mathcal{S})]\, n \log n$,

– the asymptotic cost $|B(L, x)|$ of inserting x in $T(L)$ is $[1/h(\mathcal{S})] \log n$.

here : $h(\mathcal{S})$ is the entropy of the source.

Trie and Dichotomy (I)

Proposition. The minimal number of symbol comparisons needed to solve the selection problem $\textsc{Select}\ (L, x)$ (without prior knowledge on any list L) equals the branch length $|B(L, x)|$ of the trie $T(L \cup \{x\})$ leading to the external node associated with x.

For $|L| = n \to \infty$, in a quite general probabilistic framework, in terms of number of symbol comparisons, and on average, (well known for AofA people)

- the asymptotic cost of building the trie $T(L)$ is $[1/h(\mathcal{S})]\, n \log n$,
- the asymptotic cost $|B(L, x)|$ of inserting x in $T(L)$ is $[1/h(\mathcal{S})] \log n$.

here : $h(\mathcal{S})$ is the entropy of the source.

Here, the list L is ordered. There are now main questions:

- Is it possible to take advantage of the ordering of the list, with a clever use of the dichotomic strategy?
- What can be done without explicitly building or storing the trie $T(L)$?
- How far are algorithms from the lower bound $|B(L, x)|$?

Main facts.

There is an algorithm proposed by Crochemore Hancart and Lecroq, called here DICHO-SELECT-W-MAX, that answers these main questions:

Main facts.

There is an algorithm proposed by Crochemore Hancart and Lecroq, called here DICHO-SELECT-W-MAX, that answers these main questions:

– It takes advantage of the ordering of the list,
with a clever use of the dichotomic strategy.

Main facts.

There is an algorithm proposed by Crochemore Hancart and Lecroq, called here DICHO-SELECT-W-MAX, that answers these main questions:

– It takes advantage of the ordering of the list,
with a clever use of the dichotomic strategy.

– It does not explicitely build or store the trie $T(L)$.

Main facts.

There is an algorithm proposed by Crochemore Hancart and Lecroq, called here DICHO-SELECT-W-MAX, that answers these main questions:

– It takes advantage of the ordering of the list,
with a clever use of the dichotomic strategy.

– It does not explicitely build or store the trie $T(L)$.
Instead, it precomputes (and stores) an integer array of length n.
The mean complexity cost for building the array is $\Theta(n)$.

Main facts.

There is an algorithm proposed by Crochemore Hancart and Lecroq, called here DICHO-SELECT-W-MAX, that answers these main questions:

- It takes advantage of the ordering of the list,
  with a clever use of the dichotomic strategy.
- It does not explicitely build or store the trie $T(L)$.
  Instead, it precomputes (and stores) an integer array of length n.
  The mean complexity cost for building the array is $\Theta(n)$.
- The mean complexity cost of the algorithm is $\Theta(\log n)$.

Main facts.

There is an algorithm proposed by Crochemore Hancart and Lecroq, called here DICHO-SELECT-W-MAX, that answers these main questions:

– It takes advantage of the ordering of the list,
with a clever use of the dichotomic strategy.

– It does not explicitely build or store the trie $T(L)$.
Instead, it precomputes (and stores) an integer array of length n.
The mean complexity cost for building the array is $\Theta(n)$.

– The mean complexity cost of the algorithm is $\Theta(\log n)$.

Main result of the talk :

We make precise the constants hidden in the Θ's .

Trie and Dichotomy (II)

Even though the trie $T(L)$ is not explicitely built,

our analysis uses simultaneously the dichotomic tree and the trie $T(L)$.

and, as we will see, an hybrid structure that mixes the two structures

Trie and Dichotomy (II)

Even though the trie $T(L)$ is not explicitely built,
our analysis uses simultaneously the dichotomic tree and the trie $T(L)$.
and, as we will see, an hybrid structure that mixes the two structures

We associate with each dichotomic node (b, e)
the longest common prefix $\Gamma[b, e]$ between L_b and L_e;
it appears as a portion of a branch in the trie $T(L)$.

The execution path $\Pi(L, x)$ in $D(L)$ leading to an external node $(i-1, i)$
gives rise in the trie $T(L)$,

– to the branch labelled with $\Gamma[i-1, i]$
– completed by a suffix $B(L, x) \setminus \Gamma[i-1, i]$ needed to insert x in $T(L)$.

Trie and Dichotomy (II)

Even though the trie $T(L)$ is not explicitely built,
our analysis uses simultaneously the dichotomic tree and the trie $T(L)$.
and, as we will see, an hybrid structure that mixes the two structures

We associate with each dichotomic node (b, e)
the longest common prefix $\Gamma[b, e]$ between L_b and L_e;
it appears as a portion of a branch in the trie $T(L)$.

The execution path $\Pi(L, x)$ in $D(L)$ leading to an external node $(i-1, i)$
gives rise in the trie $T(L)$,

– to the branch labelled with $\Gamma[i-1, i]$
– completed by a suffix $B(L, x) \setminus \Gamma[i-1, i]$ needed to insert x in $T(L)$.

Two facts:

– The sequence $(b, e) \mapsto \Gamma[b, e]$ is increasing along the path $\Pi(L, x)$.
– The strategy of the algorithm defines how each $\Gamma[b, e]$ is computed.

A first improvement : the algorithm "With Min"

DICHO-SELECT(L, x) restarts from the beginning of $B(L, x)$ at each step.

A first improvement uses the properties of coincidences

$$\gamma[m, x] \geq \min(\gamma[b, x], \gamma[e, x]) = \gamma[b, e]\,.$$

A first improvement : the algorithm "With Min"

DICHO-SELECT(L, x) restarts from the beginning of $B(L, x)$ at each step.

A first improvement uses the properties of coincidences

$$\gamma[m, x] \geq \min(\gamma[b, x], \gamma[e, x]) = \gamma[b, e]\,.$$

DICHO-SELECT-W-MIN (L, x);

$b := 0$; $e := n + 1$;

$\ell_b := 0$; $\ell_e := 0$;

While $b + 1 < e$

  do $m := \mathrm{mid}(b, e)$;

$\ell := \min(\ell_b, \ell_e)$;

While $x[\ell] = L_m[\ell]$

  do $\ell := \ell + 1$;

If $L_m[\ell] < x[\ell]$

  then $b := m$; $\ell_b := \ell$;

  else $e := m$; $\ell_e := \ell$;

Return (b, e).

The new algorithm maintains along the path $\Pi(L, x)$ the current coincidences

$$\ell_b := \gamma[b, x] \quad \text{and} \quad \ell_e := \gamma[b, x]$$

... and their minimum

A first improvement : the algorithm "With Min"

DICHO-SELECT(L, x) restarts from the beginning of $B(L, x)$ at each step.

A first improvement uses the properties of coincidences

$$\gamma[m, x] \geq \min(\gamma[b, x], \gamma[e, x]) = \gamma[b, e]\,.$$

DICHO-SELECT-W-MIN (L, x);

$b := 0$; $e := n + 1$;

$\ell_b := 0$; $\ell_e := 0$;

While $b + 1 < e$

  do $m := \mathrm{mid}(b, e)$;

$\ell := \min(\ell_b, \ell_e)$;

While $x[\ell] = L_m[\ell]$

  do $\ell := \ell + 1$;

If $L_m[\ell] < x[\ell]$

  then $b := m$; $\ell_b := \ell$;

  else $e := m$; $\ell_e := \ell$;

Return (b, e).

The new algorithm maintains along the path $\Pi(L, x)$ the current coincidences

$$\ell_b := \gamma[b, x] \quad \text{and} \quad \ell_e := \gamma[b, x]$$

... and their minimum

Proposition. The total number $M(L, x)$ of symbol comparisons performed by DICHO-SELECT-W-MIN is

$$M(L, x) = \sum_{(b,e) \in \underline{\Pi}(L,x)} (\gamma[m, x] - \gamma[b, e] + 1)$$

where $m := \mathrm{mid}(b, e)$

$\underline{\Pi}(x, L) :=$ the internal part of $\Pi(L, x)$

Further improvements on DICHO-SELECT: Min Versus Max ?

DICHO-SELECT-W-MIN(L, x) does not seem yet to be optimal:

After computation at node $(b, e) \in D(L)$,
– the prefixes $\Gamma(b, x)$ and $\Gamma(e, x)$ are already computed,
– the branch $B(L, x)$ is known until depth $\max(\gamma[b, x], \gamma[e, x])$.

Yet, the previous algorithm
– begins the next computation at depth $\min(\gamma[b, x], \gamma[e, x])$ along $B(L, x)$
– thus may repeat the same symbol comparisons along the branch $B(L, x)$.

Further improvements on DICHO-SELECT: Min Versus Max ?

DICHO-SELECT-W-MIN(L, x) does not seem yet to be optimal:

After computation at node $(b, e) \in D(L)$,
– the prefixes $\Gamma(b, x)$ and $\Gamma(e, x)$ are already computed,
– the branch $B(L, x)$ is known until depth $\max(\gamma[b, x], \gamma[e, x])$.

Yet, the previous algorithm
– begins the next computation at depth $\min(\gamma[b, x], \gamma[e, x])$ along $B(L, x)$
– thus may repeat the same symbol comparisons along the branch $B(L, x)$.

Is it possible to re-start at depth $\max(\gamma[b, x], \gamma[e, x])$ *in* $B(L, x)$ *?*

Further improvements on DICHO-SELECT: Min Versus Max ?

DICHO-SELECT-W-MIN(L, x) does not seem yet to be optimal:

After computation at node $(b, e) \in D(L)$,
– the prefixes $\Gamma(b, x)$ and $\Gamma(e, x)$ are already computed,
– the branch $B(L, x)$ is known until depth $\max(\gamma[b, x], \gamma[e, x])$.

Yet, the previous algorithm
– begins the next computation at depth $\min(\gamma[b, x], \gamma[e, x])$ along $B(L, x)$
– thus may repeat the same symbol comparisons along the branch $B(L, x)$.

Is it possible to re-start at depth $\max(\gamma[b, x], \gamma[e, x])$ *in* $B(L, x)$ *?*

Crochemore, Hancart and Lecroq show that
– the auxilliary set LCP$(L) := \{\gamma[b, e] \mid (b, e) \in D(L)\}$ makes this possible.
– this set that only depends on L is precomputed and stored in an array

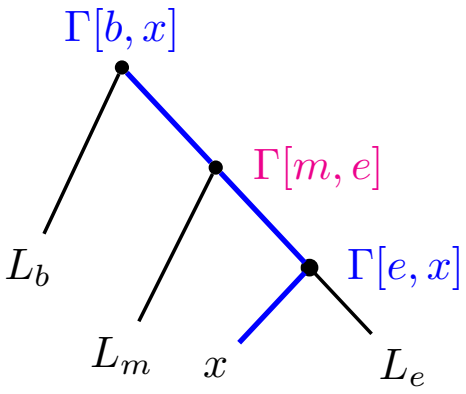
Now, at each step (b, e) in the path $\Pi(L, x)$, comparing the sets

$$\{\gamma[b, x], \gamma[e, x]\}, \quad \{\gamma[b, e], \gamma[b, m], \gamma[m, e]\},$$

avoids repetition of the same symbol comparison. (we will see it now)

Main principles of DICHO-SELECT-W-MAX viewed on the trie $T(L)$

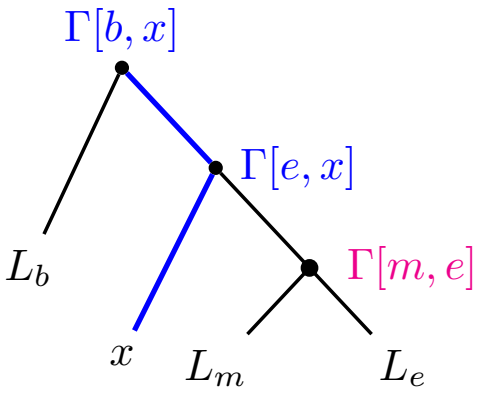
Color coding: Already computed – Precomputed – New computation

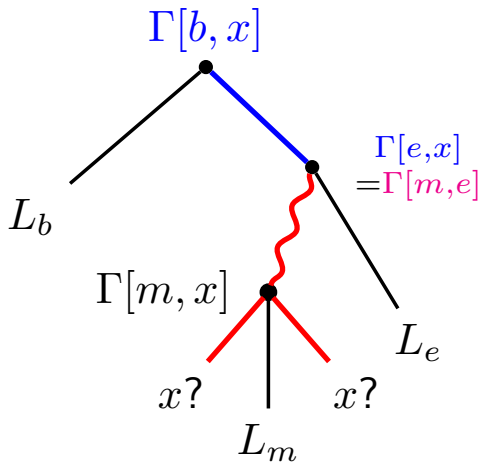


When $\gamma[e,x] \geq \gamma[b,x]$, comparison between $\gamma[e,x]$ and $\gamma[m,e]$.

Three cases:

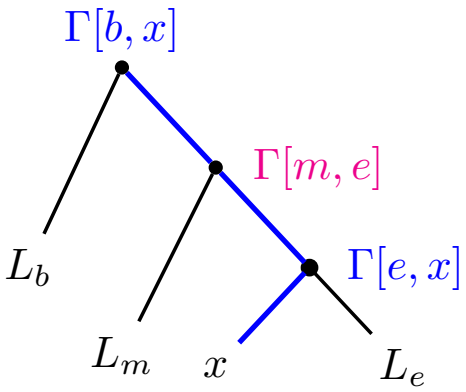
$\gamma[e,x] > \gamma[m,e]$, $\gamma[e,x] < \gamma[m,e]$, $\gamma[e,x] = \gamma[m,e]$





Main principles of DICHO-SELECT-W-MAX viewed on the trie $T(L)$

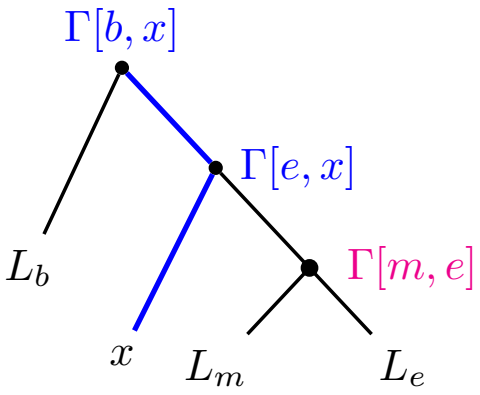
Color coding: Already computed – Precomputed – New computation



When $\gamma[e,x] \geq \gamma[b,x]$, comparison between $\gamma[e,x]$ and $\gamma[m,e]$.

Three cases:

$\gamma[e,x] > \gamma[m,e]$, $\gamma[e,x] < \gamma[m,e]$, $\gamma[e,x] = \gamma[m,e]$



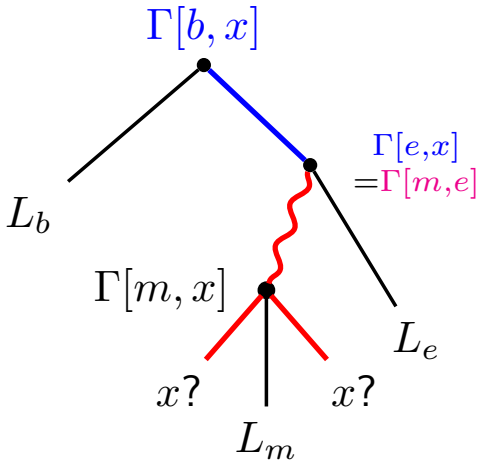
The last case: the only case

– where there is a computation,

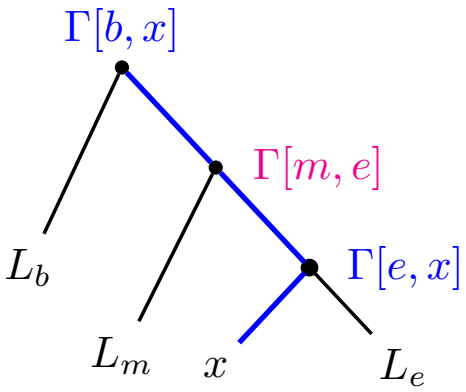
– which thus makes a (possible) progress and ends with a failure.

Such a node is called a forward node.

The forward set of $\underline{\Pi}(L,x)$ is denoted as $F(L,x)$.



Main principles of DICHO-SELECT-W-MAX viewed on the trie $T(L)$



Color coding: Already computed – Precomputed – New computation

When $\gamma[e,x] \geq \gamma[b,x]$, comparison between $\gamma[e,x]$ and $\gamma[m,e]$.

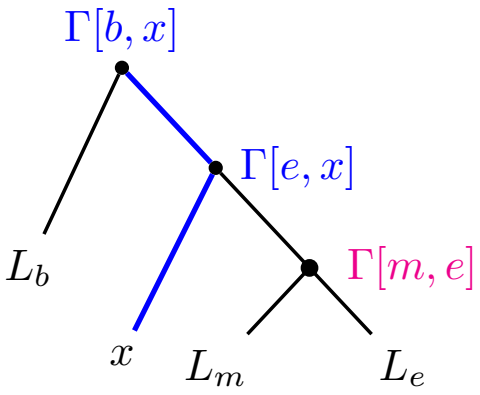
Three cases:

$\gamma[e,x] > \gamma[m,e]$, $\gamma[e,x] < \gamma[m,e]$, $\gamma[e,x] = \gamma[m,e]$

The last case: the only case

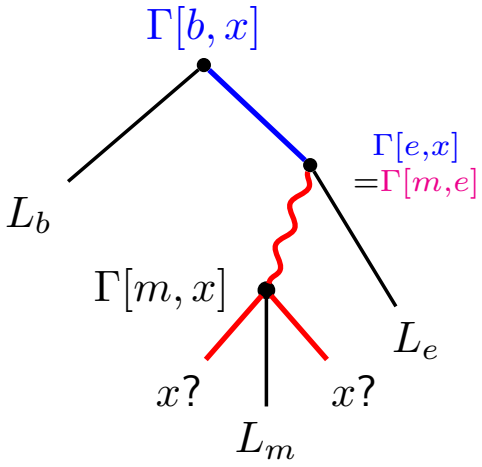
– where there is a computation,

– which thus makes a (possible) progress and ends with a failure.



Such a node is called a forward node.

The forward set of $\underline{\Pi}(L,x)$ is denoted as $F(L,x)$.



Proposition. The number of symbol comparisons performed by DICHO-SELECT-W-MAX(L,x) equals

$$D(L,x) = |B(L,x)| + |F(L,x)|$$

$B(L,x)$ is the branch of the trie $T(L \cup \{x\})$ that leads to x

$F(L,x)$ is the set of forward nodes of $\underline{\Pi}(L,x)$

Pre-computation of the LCP array

The LCP array contains all the coincidences $\gamma[b,e]$ for any $(b,e) \in D(L)$.

We propose a top-down computation

while the authors deal with a bottom-up principle.

BUILD-LCP(b,e,ℓ)
if $b=0$ **or** $e=n+1$
  then $\ell'=0$
  else
    $\ell' := \ell$
    while $L_b[\ell'] = L_e[\ell']$ **do**
      $\ell' := \ell' + 1$
$\gamma[b,e] := \ell'$
if $b+1<e$ **then** $m := \text{mid}(b,e)$
  BUILD-LCP(b,m,ℓ')
  BUILD-LCP(m,e,ℓ')

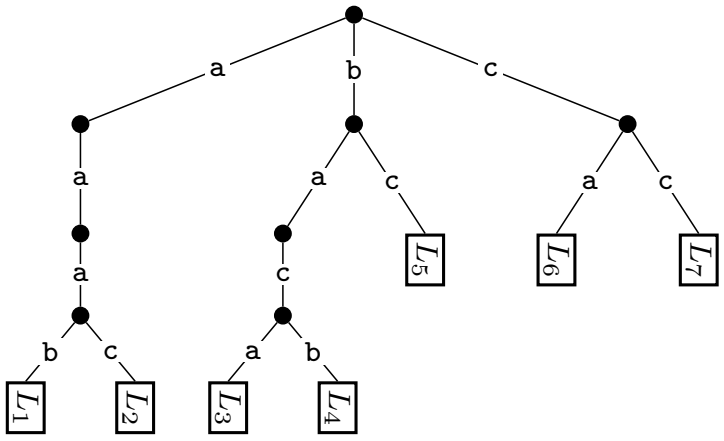
On the node (b,e), the algorithm uses $\ell := \gamma[b',e']$ with the parent (b',e') of node (b,e).

Total computation with BUILD-LCP$(0,n+1,0)$.

Proposition. The number of symbol comparisons performed to compute the LCP array for list L equals

$$C(L) = \sum_{(b,e)\in \underline{D}(L)} (\gamma[b,m]-\gamma[b,e]) + (\gamma[m,e]-\gamma[b,e])$$

The cost $C(L)$ for computing the LCP-array viewed on the trie $T(L)$



An internal node of $T(L)$ of label w

$\longrightarrow$ $\texttt{Int}(w)$:= the largest interval $[a, b]$ for which $w = \Gamma[a, b]$

$\texttt{Int(a)} = \texttt{Int(aa)} = \texttt{Int(aaa)} = [1, 2]$

$\texttt{Int(b)} = [3, 5], \quad \texttt{Int(c)} = [6, 7]$

$\texttt{Dic}(w)$:= the decomposition of $\texttt{Int}(w)$ into largest possible dichotomic intervals

$\texttt{dic}(w)$:= the cardinality of the decomposition $\texttt{Dic}(w)$

$\texttt{Dic(aaa)} = \{[1, 2]\}, \quad \texttt{dic(aaa)} = 1,$

$\texttt{Dic(b)} = \{[3, 4], [4, 5]\}, \quad \texttt{dic(b)} = 2$

Proposition. The number $C(L)$ of symbol comparisons needed to compute the LCP array on list L is closely related to

$$\widehat{C}(L) = \sum_{\substack{w \in T(L) \\ w \neq \epsilon}} \texttt{dic}(w)$$

More on the parameter dic – Case of a pure dichotomy

In this case, the dichotomic intervals are defined with binary codings.

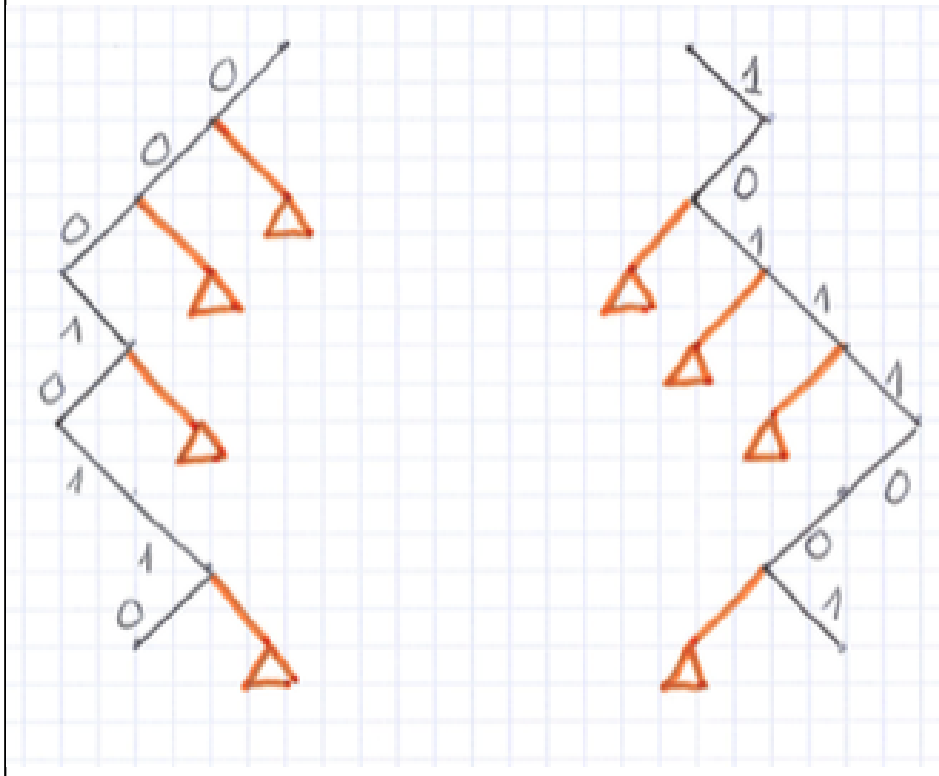
The dic parameter related to numbers of zeroes and ones in binary codings

Assume $\texttt{Int}(w) = [A_w, B_w]$
Consider the two binary words A_w and B_w
and their longest common prefix C_w

The two paths in $D(L)$
– from A_w to the root – from B_w to the root
meet at the node with binary coding C_w:

$\texttt{dic}(w)$ is the number of trees
that are hung inside the "triangle" A_w, C_w, B_w

Their total number
= (number of $'0'$s of the left branch C_w, A_w) -1
+ (number of $'1'$s of the right branch C_w, B_w) -1



More on the parameter dic – Case of a pure dichotomy

In this case, the dichotomic intervals are defined with binary codings.

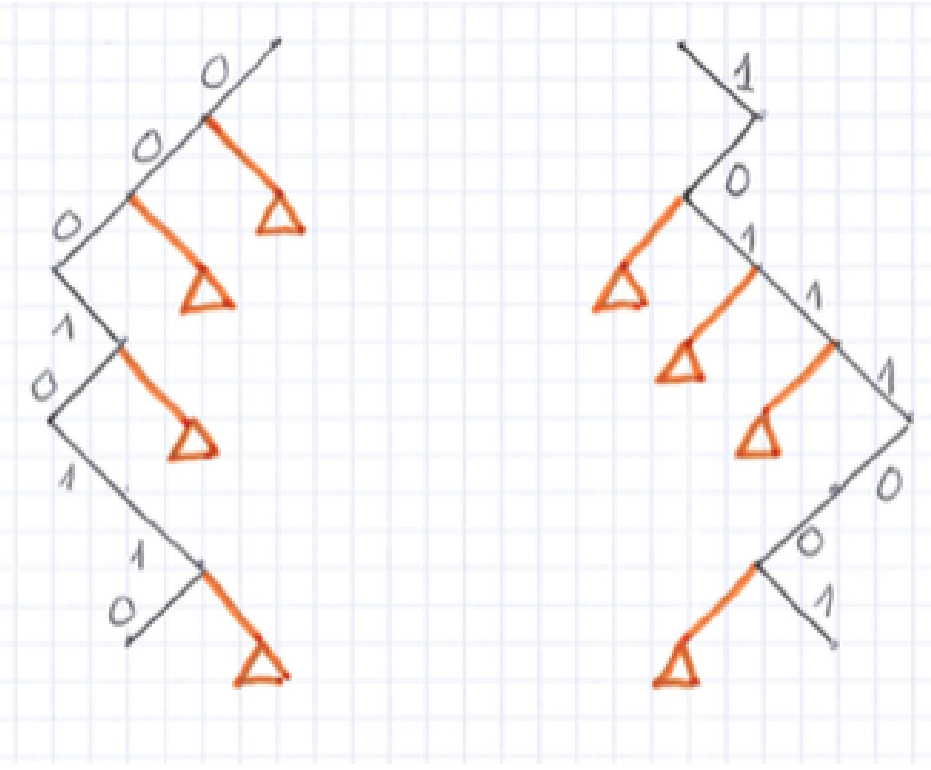
The dic parameter related to numbers of zeroes and ones in binary codings

Assume $\texttt{Int}(w) = [A_w, B_w]$
Consider the two binary words A_w and B_w
and their longest common prefix C_w

The two paths in $D(L)$
– from A_w to the root – from B_w to the root
meet at the node with binary coding C_w:

$\texttt{dic}(w)$ is the number of trees
that are hung inside the "triangle" A_w, C_w, B_w

Their total number
= (number of $'0'$s of the left branch C_w, A_w) -1
+ (number of $'1'$s of the right branch C_w, B_w) -1



One has $\texttt{dic}(w) \leq \log_2(C_w - A_w) + \log_2(B_w - C_w) \leq 2\log_2 N_w$.

Probabilistic analysis of the DICHO-SELECT-W-MAX Algorithm

Three costs of interest for DICHO-SELECT-W-MAX

– The length $|B(L, x)|$ of the branch which leads to x in the trie $T(L \cup \{x\})$

– The cost $C(L)$ of building the LCP array.

– The number $|F(L, x)|$ of forward nodes in the dichotomic path $\underline{\Pi}(L, x)$

Three costs of interest for DICHO-SELECT-W-MAX

– The length $|B(L,x)|$ of the branch which leads to x in the trie $T(L \cup \{x\})$

– The cost $C(L)$ of building the LCP array.

– The number $|F(L,x)|$ of forward nodes in the dichotomic path $\underline{\Pi}(L,x)$

The first parameter $|B(L,x)|$ is a classical parameter of the trie $T(L \cup \{x\})$

– it is only based on probabilistic properties of the (initial) source $\mathcal{S}$

Three costs of interest for DICHO-SELECT-W-MAX

– The length $|B(L,x)|$ of the branch which leads to x in the trie $T(L \cup \{x\})$

– The cost $C(L)$ of building the LCP array.

– The number $|F(L,x)|$ of forward nodes in the dichotomic path $\underline{\Pi}(L,x)$

The first parameter $|B(L,x)|$ is a classical parameter of the trie $T(L \cup \{x\})$

– it is only based on probabilistic properties of the (initial) source $\mathcal{S}$

The other two parameters are much more difficult to analyze.

They involve both the trie $T(L)$ and the dichotomic tree $D(L)$,

Three costs of interest for DICHO-SELECT-W-MAX

– The length $|B(L,x)|$ of the branch which leads to x in the trie $T(L \cup \{x\})$

– The cost $C(L)$ of building the LCP array.

– The number $|F(L,x)|$ of forward nodes in the dichotomic path $\underline{\Pi}(L,x)$

The first parameter $|B(L,x)|$ is a classical parameter of the trie $T(L \cup \{x\})$

– it is only based on probabilistic properties of the (initial) source $\mathcal{S}$

The other two parameters are much more difficult to analyze.
They involve both the trie $T(L)$ and the dichotomic tree $D(L)$,

– However, the cost $C(L)$ may be read on the trie $T(L)$

– The parameter $F(L,x)$ seems difficult to deal with ... : no idea !

Only trivial bounds $1 \leq |F(L,x)| \leq \underline{\Pi}(L,x)$

A detour : Probabilistic analysis of "costs with toll" in a random trie $T(L)$

With a finite prefix w, we associate

$$N_w = N_w(L) \text{ the number of elements of } L \text{ which begin with } w.$$

The variable N_w plays a central role in the analysis of the trie parameters.

Definition. A cost X is a cost with toll in the trie $T(L)$ if there is a cost $x : k \mapsto x(k)$ defined on $\mathbb{N}$ for which

$$X(L) := \sum_{w \mid N_w \geq 2} x(N_w)$$

Classical examples of costs with toll: size, path-length, sorting cost

Even if the (new) parameter `dic` is NOT a cost with toll, it satisfies

$$1 \leq \texttt{dic}(w) \leq 2 \log_2 N_w$$

A detour : Probabilistic analysis of "costs with toll" in a random trie $T(L)$

With a finite prefix w, we associate

$N_w = N_w(L)$ the number of elements of L which begin with w.

The variable N_w plays a central role in the analysis of the trie parameters.

Definition. A cost X is a cost with toll in the trie $T(L)$ if there is a cost $x : k \mapsto x(k)$ defined on $\mathbb{N}$ for which

$$X(L) := \sum_{w \mid N_w \geq 2} x(N_w)$$

Classical examples of costs with toll: size, path-length, sorting cost

Even if the (new) parameter `dic` is NOT a cost with toll, it satisfies

$$1 \leq \texttt{dic}(w) \leq 2 \log_2 N_w$$

Proposition. For a "good" source with entropy $h(\mathcal{S})$, and for a toll that satisfies $x(k) = O(k^a)$ with $0 \leq a < 1$, the mean value X_n of $X(L)$ for $|L| = n$ satisfies

$$X_n \sim \frac{n}{h(\mathcal{S})} \sum_{k \geq 2} \frac{x(k)}{k(k-1)}$$

Applied with costs x

$x(k) = 1$

$x(k) = (2/\log 2) \log k$

First conclusion: Probabilistic analysis of DICHO-SELECT-W-MAX

Proposition. Consider a list L with n words uniformly drawn from the same "good" source.

– The mean value B_n of the length $B(L)$ of a random branch of $T(L)$ satisfies

$$B_n \sim \frac{1}{h(\mathcal{S})} \log n$$

– The mean value C_n of the cost $C(L)$ for building the LCP array satisfies

$$\frac{n}{h(\mathcal{S})} \leq C_n \leq \frac{n}{h(\mathcal{S})} \frac{1}{\log 2} \sum_{k \geq 2} \frac{\log k}{k(k-1)}$$

– The mean value F_n of the number of forward nodes satisfies

$$1 \leq F_n \leq \frac{1}{\log 2} \log n$$

The mean cost $B_n + F_n$ of DICHO-SELECT-W-MAX is $\Theta(\log n)$.

The mean cost C_n of the precomputation is $\Theta(n)$.

with a precise evaluation of the constants in the Θ's.

We have compared DICHO-SELECT-W-MAX with the Trie strategy. We prove

– DICHO-SELECT-W-MAX is quasi-optimal,

– with a better precomputation (time and space)

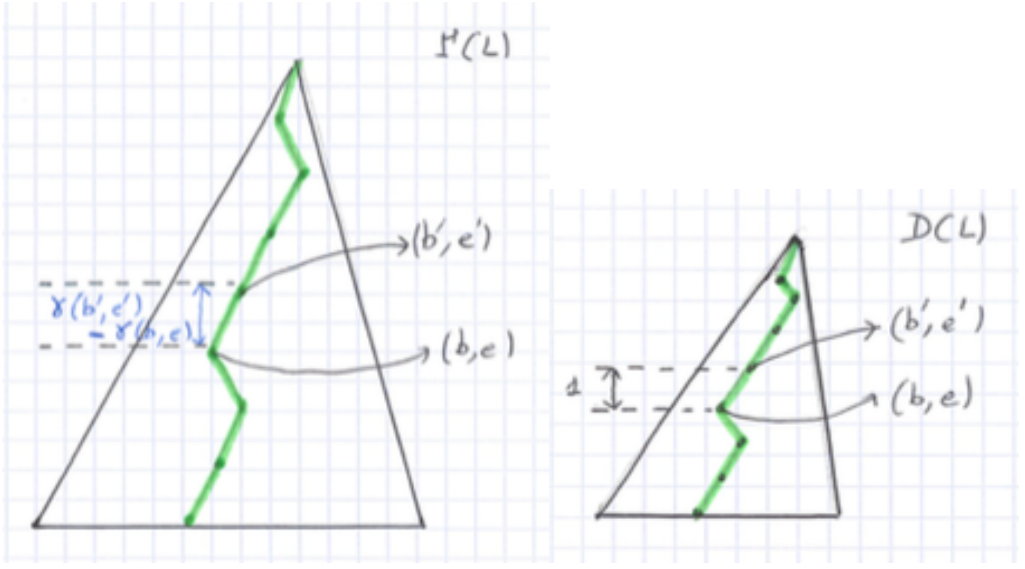
An hybrid structure that underlies the dichotomic strategy on words

The weighted dichotomic tree $\Gamma(L)$

[Case of a pure dichotomy $n = 2^p - 1$. Different roles for the guards]

$\Gamma(L)$ is the weighted dichotomic tree for which the edge $(b', e') \to (b, e)$ has the weight $\gamma[b, e] - \gamma[b', e']$

A scale change $D(L) \mapsto \Gamma(L)$



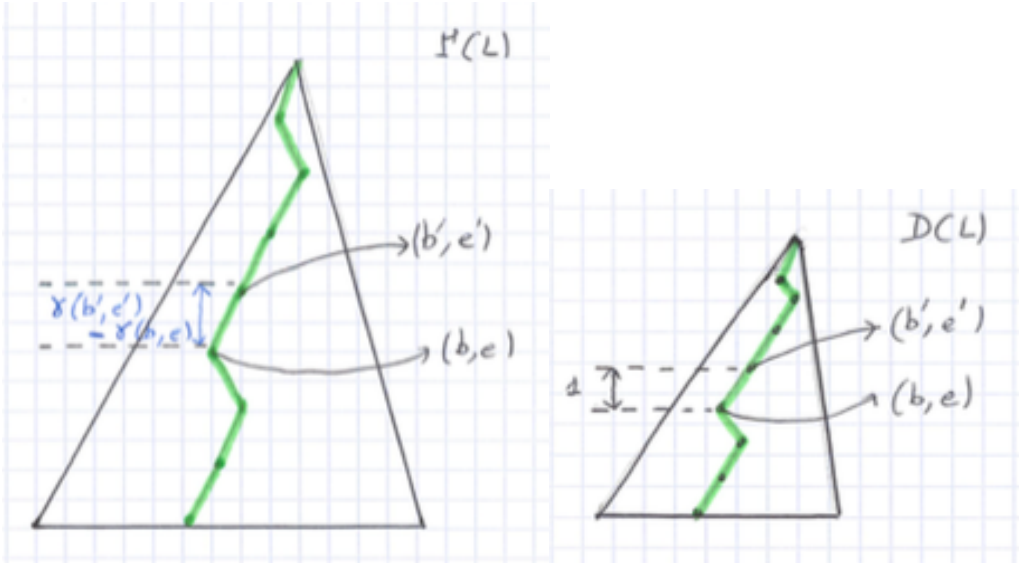
An hybrid structure that underlies the dichotomic strategy on words

The weighted dichotomic tree $\Gamma(L)$

[Case of a pure dichotomy $n = 2^p - 1$. Different roles for the guards]

$\Gamma(L)$ is the weighted dichotomic tree for which the edge $(b', e') \to (b, e)$ has the weight $\gamma[b, e] - \gamma[b', e']$

A scale change $D(L) \mapsto \Gamma(L)$



The main parameters of the various DICHO-SELECT algorithms may be read on $\Gamma(L)$.

Main question: Is the $\Gamma(L)$ tree is (globally) homothetic to the dichotomic tree?

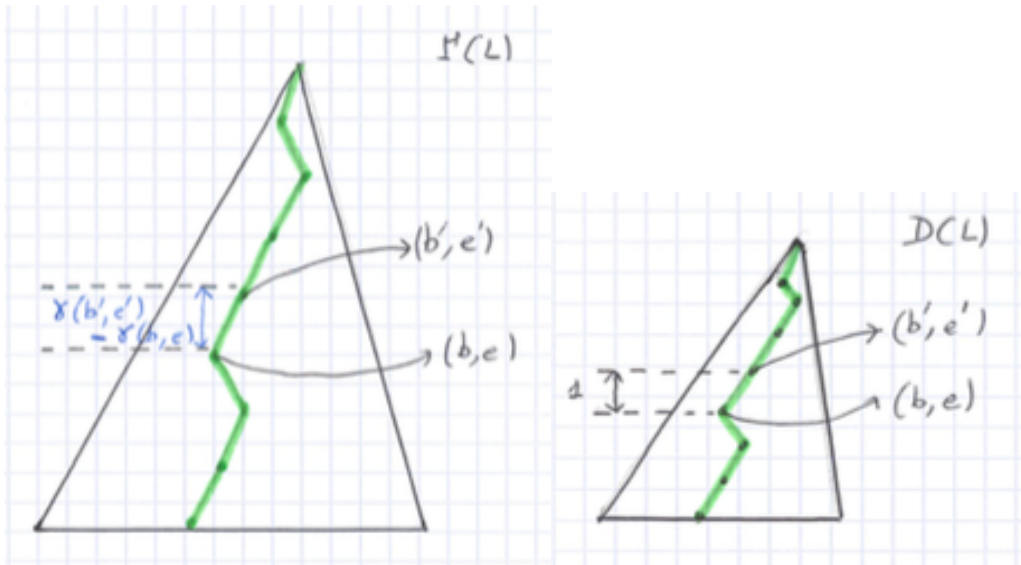
An hybrid structure that underlies the dichotomic strategy on words

The weighted dichotomic tree $\Gamma(L)$

[Case of a pure dichotomy $n = 2^p - 1$. Different roles for the guards]

$\Gamma(L)$ is the weighted dichotomic tree for which the edge $(b', e') \to (b, e)$ has the weight $\gamma[b, e] - \gamma[b', e']$

A scale change $D(L) \mapsto \Gamma(L)$



The main parameters of the various DICHO-SELECT algorithms may be read on $\Gamma(L)$.

Main question: Is the $\Gamma(L)$ tree is (globally) homothetic to the dichotomic tree?

Conjecture. Let $\pi(b, e) :=$ depth of $(b, e) \in D(L)$. The estimate holds for $p \to \infty$,

$$\gamma(b, e) \approx \frac{\log 2}{h(\mathcal{S})} \pi(b, e), \qquad \text{for any } (b, e) \in D(L) \text{ with } \pi(b, e) \to \infty$$

This is also asympt. true for any node (c, d) with $\pi(c, d) \to \infty$

$\pi(c, d) :=$ the coincidence between the binary writing of c and d.

Regular shape of the $\Gamma(L)$ tree ?

Conjecture. Let $\pi(b,e) :=$ depth of $(b,e) \in D(L)$. The estimate holds for $p \to \infty$,

$$\gamma(b,e) \approx \frac{\log 2}{h(\mathcal{S})}\pi(b,e), \qquad \text{for any } (b,e) \in D(L) \text{ with } \pi(b,e) \to \infty$$

This is also asympt. true for any node (c,d) with $\pi(c,d) \to \infty$

$\pi(c,d) :=$ the coincidence between the binary writing of c and d.

Regular shape of the $\Gamma(L)$ tree ?

Conjecture. Let $\pi(b,e) :=$ depth of $(b,e) \in D(L)$. The estimate holds for $p \to \infty$,

$$\gamma(b,e) \approx \frac{\log 2}{h(\mathcal{S})}\pi(b,e), \qquad \text{for any } (b,e) \in D(L) \text{ with } \pi(b,e) \to \infty$$

This is also asympt. true for any node (c,d) with $\pi(c,d) \to \infty$

$\pi(c,d) :=$ the coincidence between the binary writing of c and d.

Under the validity of the Conjecture,
we can deal in the analysis of DICHO-SELECT with $\pi(c,d)$ instead of $\gamma(c,d)$.

Regular shape of the $\Gamma(L)$ tree ?

Conjecture. Let $\pi(b,e) :=$ depth of $(b,e) \in D(L)$. The estimate holds for $p \to \infty$,

$$\gamma(b,e) \approx \frac{\log 2}{h(\mathcal{S})}\pi(b,e), \qquad \text{for any } (b,e) \in D(L) \text{ with } \pi(b,e) \to \infty$$

This is also asympt. true for any node (c,d) with $\pi(c,d) \to \infty$

$\pi(c,d) :=$ the coincidence between the binary writing of c and d.

Under the validity of the Conjecture,
we can deal in the analysis of DICHO-SELECT with $\pi(c,d)$ instead of $\gamma(c,d)$.

As parameter $\pi(c,d)$ is easy to deal with,
It would be then easy to study and compare all the DICHO-SELECT algorithms.

Regular shape of the $\Gamma(L)$ tree ?

Conjecture. Let $\pi(b,e) :=$ depth of $(b,e) \in D(L)$. The estimate holds for $p \to \infty$,

$$\gamma(b,e) \approx \frac{\log 2}{h(\mathcal{S})}\pi(b,e), \qquad \text{for any } (b,e) \in D(L) \text{ with } \pi(b,e) \to \infty$$

This is also asympt. true for any node (c,d) with $\pi(c,d) \to \infty$

$\pi(c,d) :=$ the coincidence between the binary writing of c and d.

Under the validity of the Conjecture,
we can deal in the analysis of DICHO-SELECT with $\pi(c,d)$ instead of $\gamma(c,d)$.

As parameter $\pi(c,d)$ is easy to deal with,
It would be then easy to study and compare all the DICHO-SELECT algorithms.

Complete proof of the conjecture :

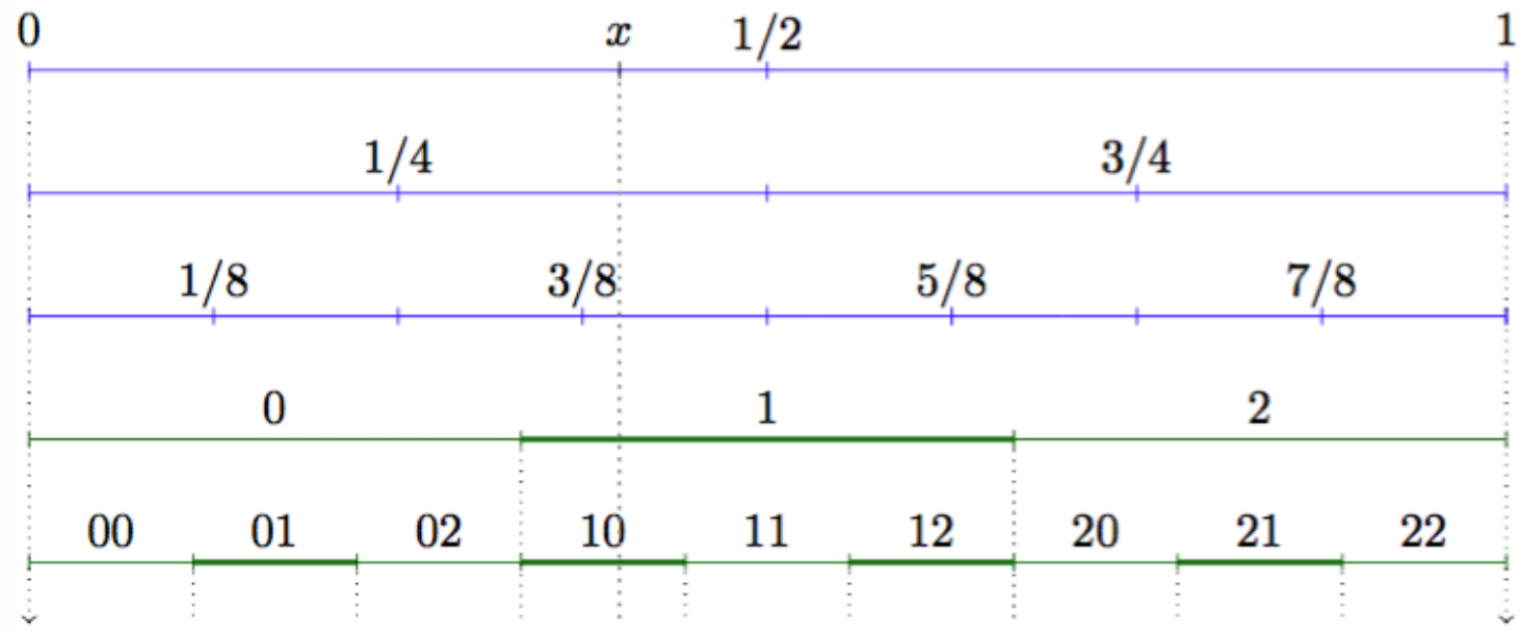
work yet in progress, with some steps already proven.

Main steps in the proof of the conjecture

– Results on order statistics

– Introduction of the (implicit) binary source $\mathcal{B}$
associated with the dichotomic process.

– Comparison of the coincidences associated with each of the two sources
  – the binary source (in blue)
  – the initial source (in green) : here the ternary source



THANK YOU !