A linear time and space algorithm for detecting path intersection in $\mathbb{Z}^d$

S. Brlek, M. Koskas, **X. Provençal**

Université Savoie Mont Blanc

June 21, 2016

LAMA
Laboratoire de Mathématiques
Université de Savoie

Freeman chain code

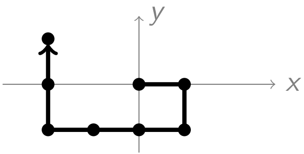
Let $\Sigma = \{a_1, \bar{a}_1, a_2, \bar{a}_2, \ldots, a_d, \bar{a}_d\}$ be a $2d$ letter alphabet and consider the mapping

$$\overrightarrow{a_i} \mapsto e_i, \qquad \overrightarrow{\bar{a}_i} \mapsto -e_i.$$

A word $w \in \Sigma^*$ defines the path p in $\mathbb{Z}^d$ such that starting from a point $x \in \mathbb{Z}$, is $p_0 = x$ and

$$p_k = p_{k-1} + \overrightarrow{w_k}, \text{ for } 1 \leq k \leq |p|.$$

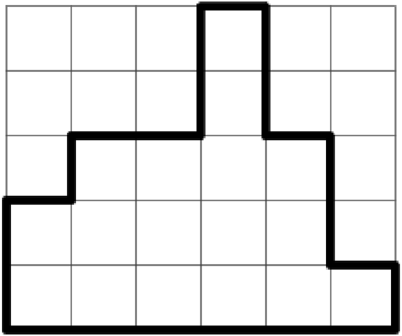
Example : $x = (0,0)$, $w = a_1\ \bar{a}_2\ \bar{a}_1\ \bar{a}_1\ \bar{a}_1\ a_2\ a_2$.



Path intersection

Problem

Given a word $w \in \Sigma^*$ *of length* n *is the path coded by* w *self-intersecting ?*



Path intersection

Problem

Given a word $w \in \Sigma^*$ of length n is the path coded by w self-intersecting ?

Given a *boundary word* of length n, we can compute in time $O(n)$:

- The sense of rotation.
- The area, the center of gravity and moment of inertia [Brlek, Labelle, Lacasse, 2003].
- Digital convexity [Debled-Rennesson, Rémy, Rouyer-Degli, 2003], [Brlek, Lachaud, P. Reutenauer, 2009].
- Tangent, length and curvature estimation [Feschet, Tougne 1999], [Lachaud, de Vieilleville 2007], [Lachaud, Kerautret, Naegel 2008].
- Does the shape tiles the plane by translation [Winslow, 2015].
- ...

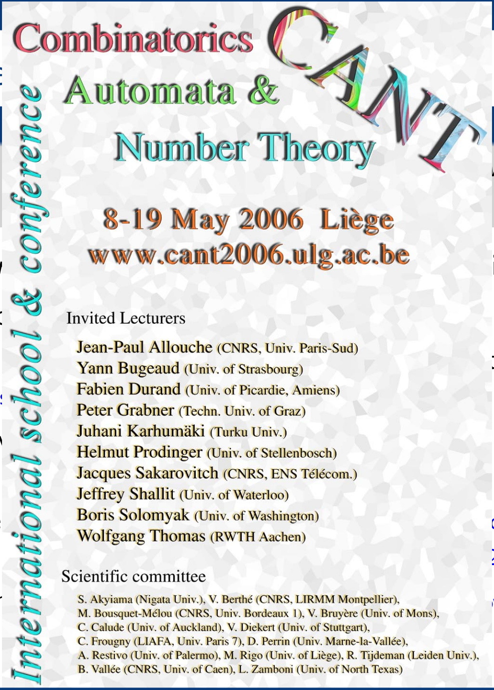
Path interse

Problem

Given a word w ... *v* *self-intersecting*

Given a *boundar* ... ime $O(n)$:

- The sense c
- The area, t ... cia [Brlek, Labelle, Lacas
- Digital conv ... 2003], [Brlek, Lacha
- Tangent, le ... ougne 1999], [Lachaud, de ... 2008].
- Does the sh ... w, 2015].
- ...



Obvious solutions

Solution (1)

Use a $n \times n$ matrix and draw the path. $O(n^2)$

Obvious solutions

Solution (1)

Use a $n \times n$ matrix and draw the path. $O(n^2)$

Solution (2)

Compute the list of visited points, sort it and check for a repetition. $O(n \log n)$

Obvious solutions

Solution (1)

Use a $n \times n$ matrix and draw the path. $O(n^2)$

Solution (2)

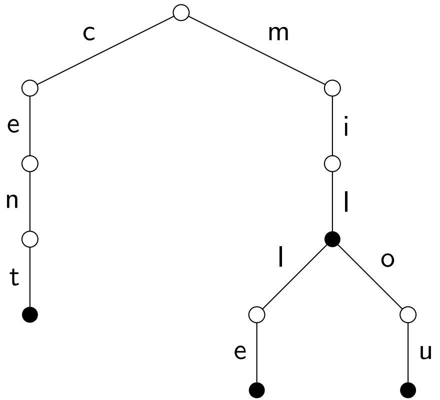
Compute the list of visited points, sort it and check for a repetition. $O(n \log n)$

Solution (3)

Build the set of visited points using a self-balancing search tree and test for existence before insertion. $O(n \log n)$

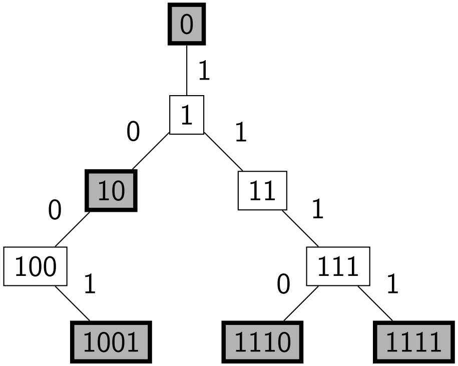
Radix tree for words

A set of words $D = \{cent, mil, mille, milou\}$ is represented by the radix tree :



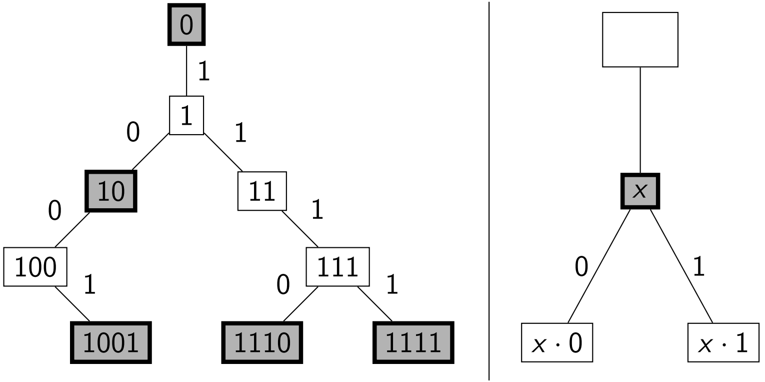
Radix tree for binary numbers

A set of words $D = \{0, 2, 9, 14, 15\}$ is represented by the binary radix tree :



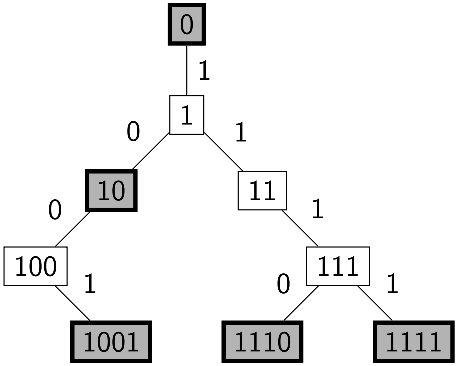
Radix tree for binary numbers

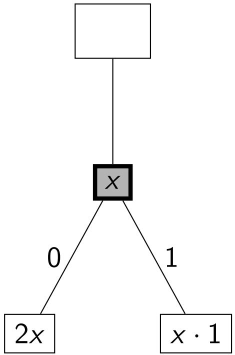
A set of words $D = \{0, 2, 9, 14, 15\}$ is represented by the binary radix tree :



Radix tree for binary numbers

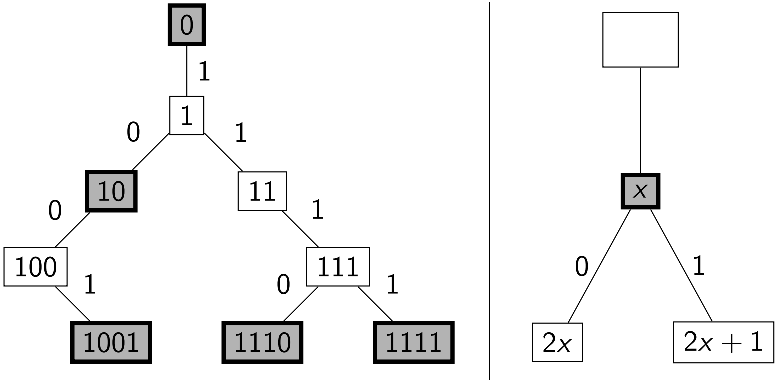
A set of words $D = \{0, 2, 9, 14, 15\}$ is represented by the binary radix tree :





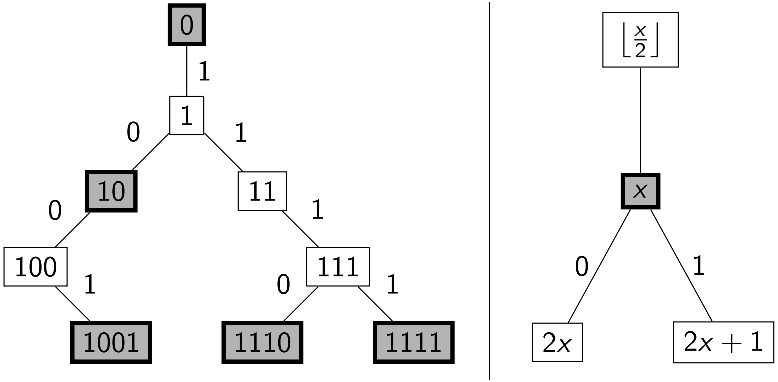
Radix tree for binary numbers

A set of words $D = \{0, 2, 9, 14, 15\}$ is represented by the binary radix tree :



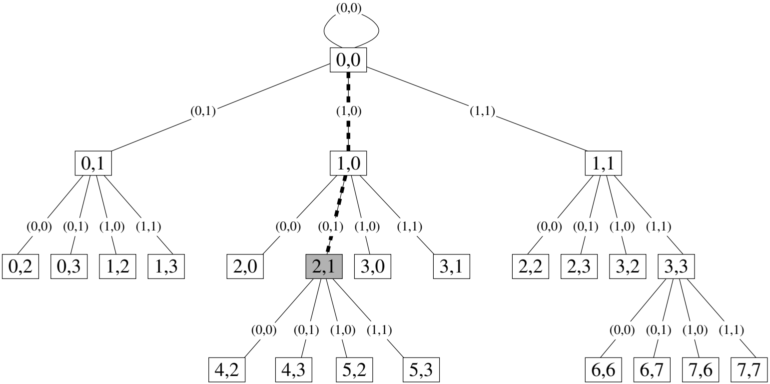
Radix tree for binary numbers

A set of words $D = \{0, 2, 9, 14, 15\}$ is represented by the binary radix tree :



Radix tree for points in $\mathbb{N}^2$

- Edge labels are in $\{(0,0),(1,0),(0,1),(1,1)\}$.
- The root $(0,0)$ is its own son for edge $(0,0)$.
- Note (x,y) has 4 child : $(x0,y0),(x0,y1),(x1,y0),(x1,y1)$.



Simplifications

- Dimension is 2.
- The path starts at $(0, 0)$.
- All coordinates are positive (path stays in $\mathbb{N}^2$).
- We use $\Sigma = \{a, \bar{a}, b, \bar{b}\}$ instead of $\{a_1, \bar{a}_1, a_2, \bar{a}_2\}$.

l-neighbors

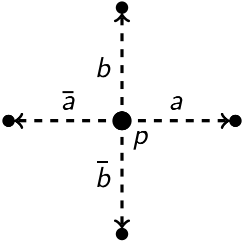
Definition

Given two points $p, q \in \mathbb{Z}^2$ and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, q is the l-neighbor of p if $q = p + \overrightarrow{l}$.

l-neighbors

Definition

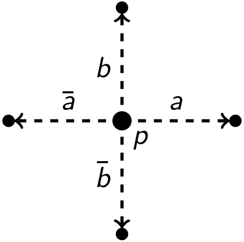
Given two points $p, q \in \mathbb{Z}^2$ and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, q is the l-neighbor of p if $q = p + \overrightarrow{l}$.



l-neighbors

Definition

Given two points $p, q \in \mathbb{Z}^2$ and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, q is the l-neighbor of p if $q = p + \overrightarrow{l}$.

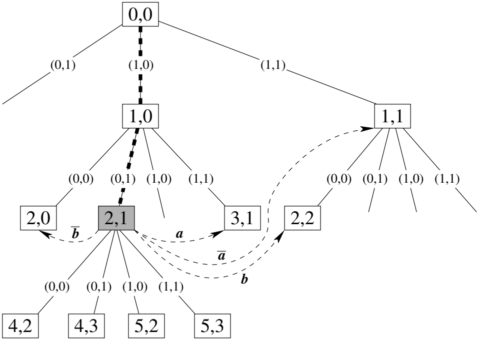


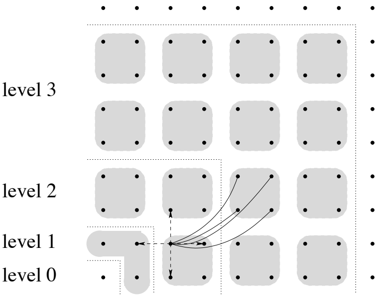
Two points p and q are neighbors iff $|p - q| = 1$.

The radix tree with the neighborhood relation

Let $G = (P, R, N)$ the graph where $P \subset \mathbb{N}^2$ is the set of **nodes** and $R \cup N$ are the **edges**.

- Edges from R (⟋) provide the radix-tree structure.
- Edges from N (⇢) links neighbors to each other.





Neighborhood and fatherhood

Notation

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$ *:*

- $F(p)$ *is the father of node* p.
- $n_l(p)$ *is the* l*-neighbor of* p.

Neighborhood and fatherhood

Notation

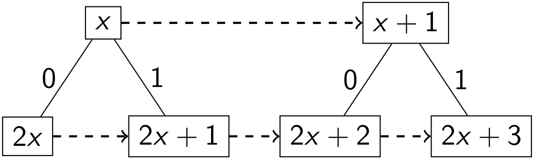
Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$:

- *$F(p)$ is the father of node p.*
- *$n_l(p)$ is the l-neighbor of p.*

Lemma

Two nodes p, q such that $n_l(p) = q$ then

$$F(p) = F(q) \quad \text{or} \quad n_l(F(p)) = F(q).$$



Propagating the carry

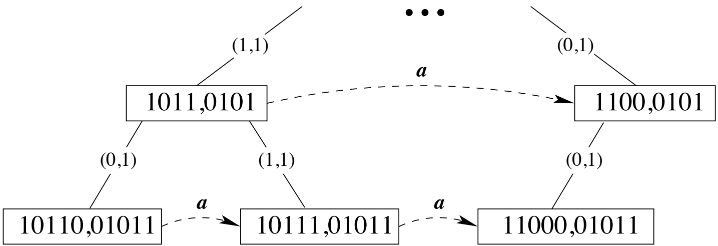
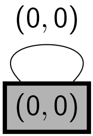


FIGURE : Adding 1 to the first coordinate in the radix tree representation.

The linear time algorithm

Each node of G is marked as *visited* or *non-visited*.

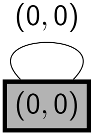
1. Initialize $G = (P, R, N)$ with $P = \{(0,0)\}$,
   R has only one edge from $(0,0)$ to $(0,0)$ with label $(0,0)$ and N is empty.
2. Let p be the root $(0,0)$
3. Mark p as visited.
4. For each letter l in w
   Let $p \leftarrow n_l(p)$.
   If p is *visited* then
   The path is self intersecting.
   otherwise
   Mark p as visited.
5. If the loop ends, then the path is not self-intersecting.



The linear time algorithm

Each node of G is marked as *visited* or *non-visited*.

1. Initialize $G = (P, R, N)$ with $P = \{(0,0)\}$, R has only one edge from $(0,0)$ to $(0,0)$ with label $(0,0)$ and N is empty.
2. Let p be the root $(0,0)$
3. Mark p as visited.
4. For each letter l in w
   Let $p \leftarrow n_l(p)$.
   If p is *visited* then
      The path is self intersecting.
   otherwise
      Mark p as visited.
5. If the loop ends, then the path is not self-intersecting.



Seeking for a neighbor

Question

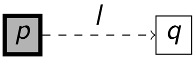
Given a node p *and a letter* $l \in \{a, \bar{a}, b, \bar{b}\}$*, how to access* $q = n_l(p)$ *?*

Seeking for a neighbor

Question

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, how to access $q = n_l(p)$?

① If there is a neighboring link with label l starting from p, use it to access q.

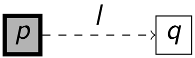


Seeking for a neighbor

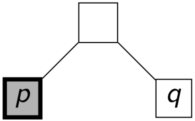
Question

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$*, how to access* $q = n_l(p)$ *?*

① If there is a neighboring link with label l starting from p, use it to access q.



② If $F(p) = F(q)$, go to $F(p)$ to access (and maybe create) the node q, then add a neighboring link from p to q.

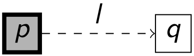


Seeking for a neighbor

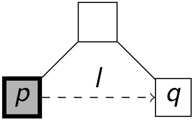
Question

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$*, how to access* $q = n_l(p)$ *?*

① If there is a neighboring link with label l starting from p, use it to access q.



② If $F(p) = F(q)$, go to $F(p)$ to access (and maybe create) the node q, then add a neighboring link from p to q.

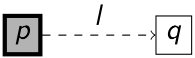


Seeking for a neighbor

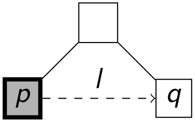
Question

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, how to access $q = n_l(p)$?

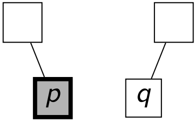
1. If there is a neighboring link with label l starting from p, use it to access q.



2. If $F(p) = F(q)$, go to $F(p)$ to access (and maybe create) the node q, then add a neighboring link from p to q.



3. If $F(p) \neq F(q)$, go to $F(p)$, use recursion to access/create $n_l(F(p)) = F(q)$, access/create q and add a neighboring link from p to q.



Seeking for a neighbor

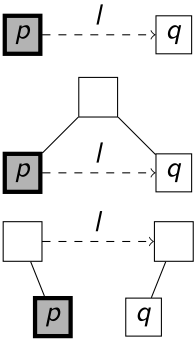
Question

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, *how to access* $q = n_l(p)$?

① If there is a neighboring link with label l starting from p, use it to access q.

② If $F(p) = F(q)$, go to $F(p)$ to access (and maybe create) the node q, then add a neighboring link from p to q.

③ If $F(p) \neq F(q)$, go to $F(p)$, use recursion to access/create $n_l(F(p)) = F(q)$, access/create q and add a neighboring link from p to q.

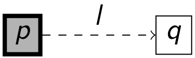


Seeking for a neighbor

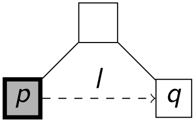
Question

Given a node p and a letter $l \in \{a, \bar{a}, b, \bar{b}\}$, how to access $q = n_l(p)$?

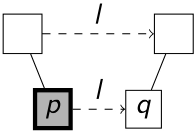
① If there is a neighboring link with label l starting from p, use it to access q.



② If $F(p) = F(q)$, go to $F(p)$ to access (and maybe create) the node q, then add a neighboring link from p to q.

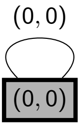


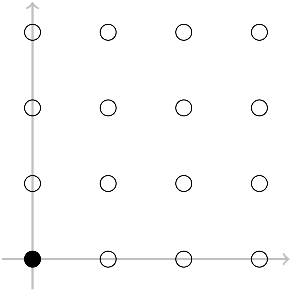
③ If $F(p) \neq F(q)$, go to $F(p)$, use recursion to access/create $n_l(F(p)) = F(q)$, access/create q and add a neighboring link from p to q.



Example

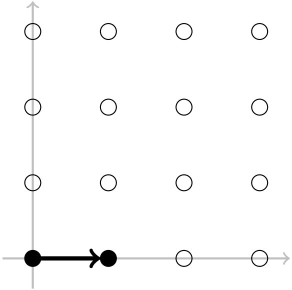
$$w = a\ a\ b\ b\ \bar{a}.$$

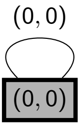




Example

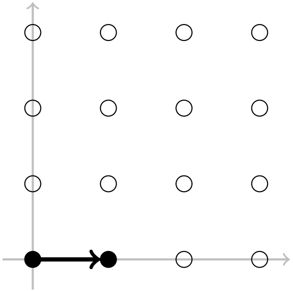
$w = \textcolor{red}{a}\ a\ b\ b\ \bar{a}.$

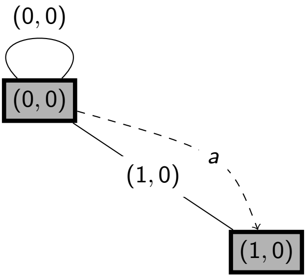




Example

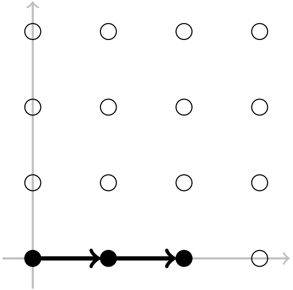
$w = \color{red}{a}\ a\ b\ b\ \bar{a}.$

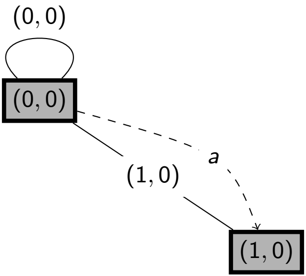




Example

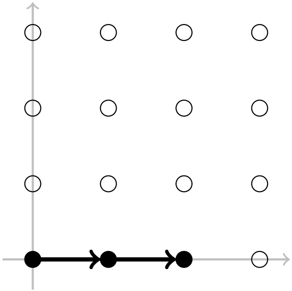
$w = a\ \color{red}{a}\ b\ b\ \bar{a}.$

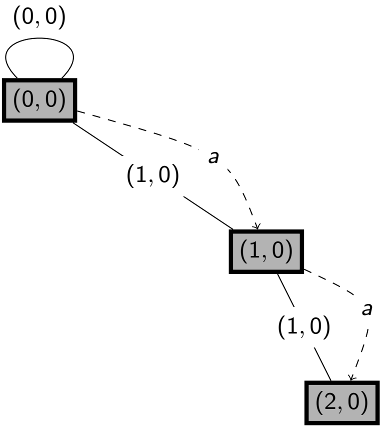




Example

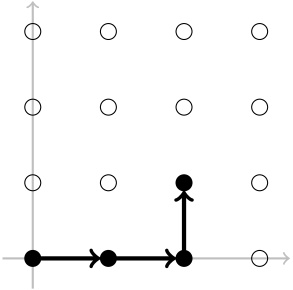
$w = a\ \color{red}{a}\ b\ b\ \bar{a}.$

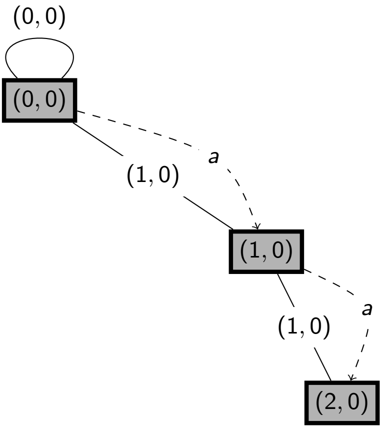




Example

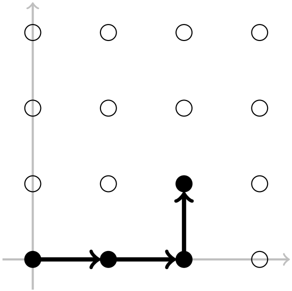
$w = a\ a\ {\color{red}b}\ b\ \bar{a}.$

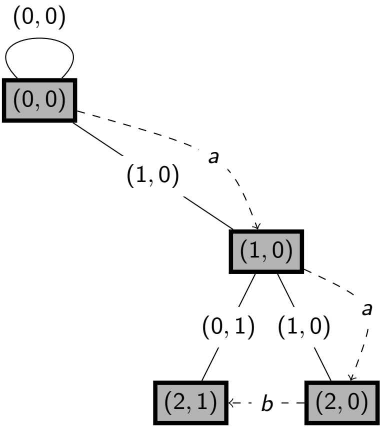




Example

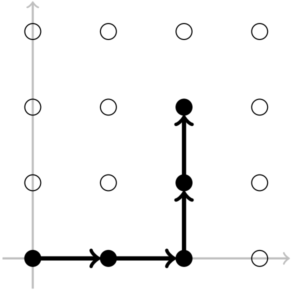
$w = a\ a\ {\color{red}b}\ b\ \bar{a}.$

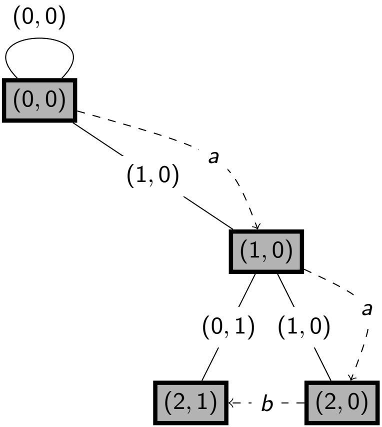




Example

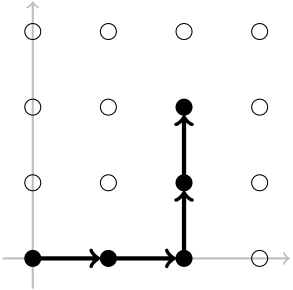
$w = a\ a\ b\ \color{red}{b}\ \bar{a}.$

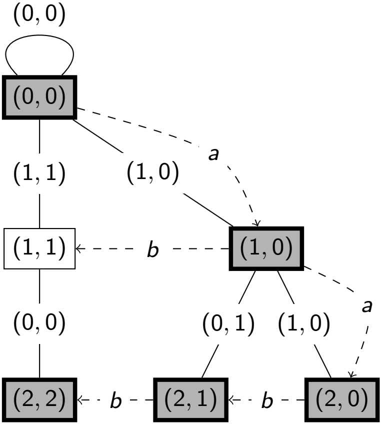




Example

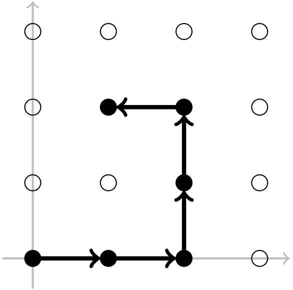
$w = a\ a\ b\ {\color{red}b}\ \bar{a}.$

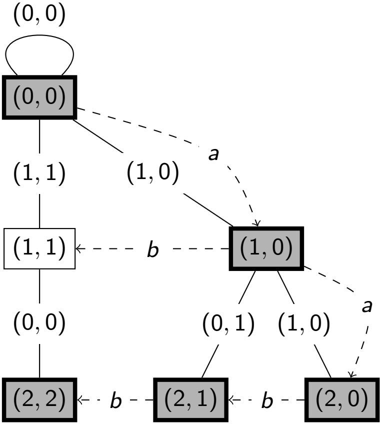




Example

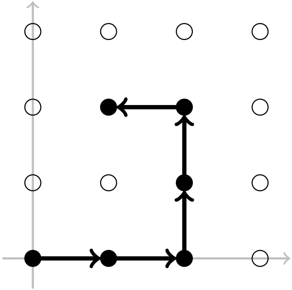
$w = a\ a\ b\ b\ \textcolor{red}{\bar{a}}.$

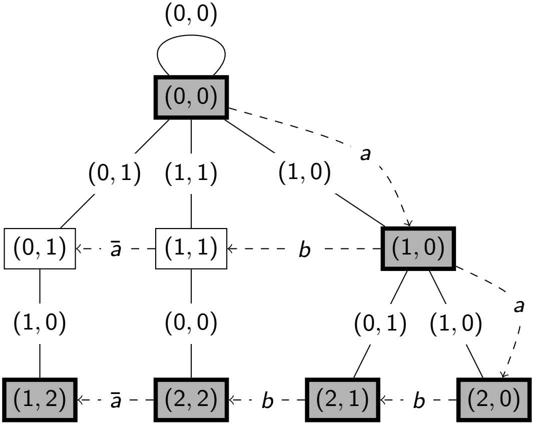




Example

$w = a\ a\ b\ b\ \bar{a}.$



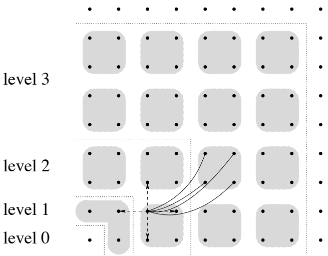


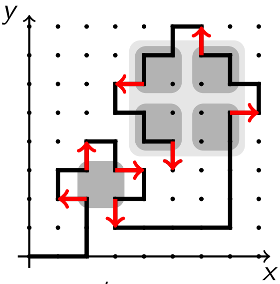
Time complexity

Lemma

The time complexity is $\theta(m)$ where m is the number of nodes in G.

Proof. The lower bound $\Omega(m)$ is trivial.
The upper bound $O(m)$ comes from the fact that after each recursive call, a new neighboring link is added. □





Max number of recursive call on a node : $d2^d$.

Space complexity

Lemma

Given a word w of length n, the graph $G_w = (N, R, T)$ obtained by our algorithm is such that $|N| \in O(n)$.

Space complexity

Lemma

Given a word w of length n, the graph $G_w = (N, R, T)$ obtained by our algorithm is such that $|N| \in O(n)$.

Proof. Consider N_v the nodes of N that are marked as visited and h be the height of the tree (N, R).

$$|F^{i+1}(N_v)| \leq \frac{4}{5}|F^i(N_v)|$$

Space complexity

Lemma

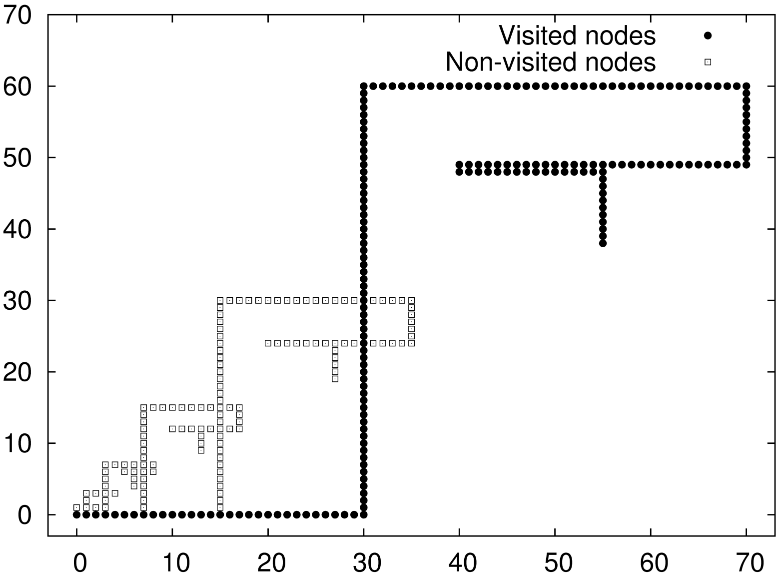
Given a word w of length n, the graph $G_w = (N, R, T)$ obtained by our algorithm is such that $|N| \in O(n)$.

Proof. Consider N_v the nodes of N that are marked as visited and h be the height of the tree (N, R).

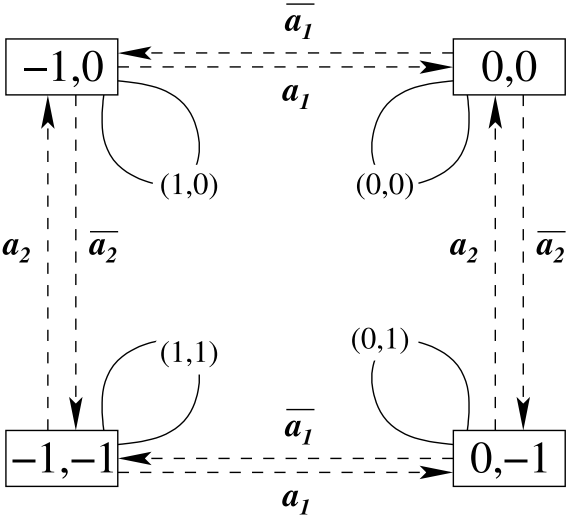
$$|F^{i+1}(N_v)| \leq \frac{4}{5}|F^i(N_v)|$$

$$|N| \leq \sum_{0 \leq i \leq h} |F^i(N_v)| \leq 5|N_v| + 20h$$

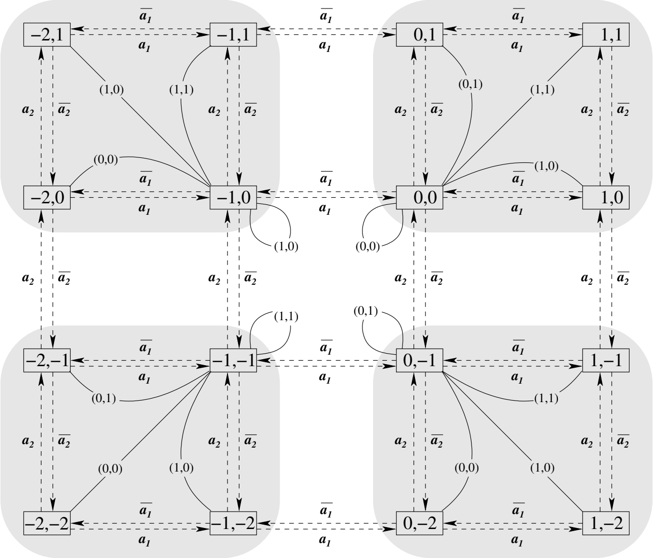
□


Visited nodes
Non-visited nodes
0
10
20
30
40
50
60
70
0
10
20
30
40
50
60
70

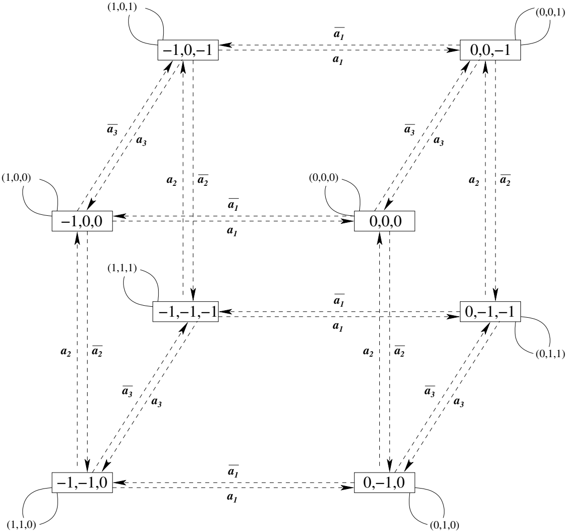
Coordonnées négatives



Coordonnées négatives

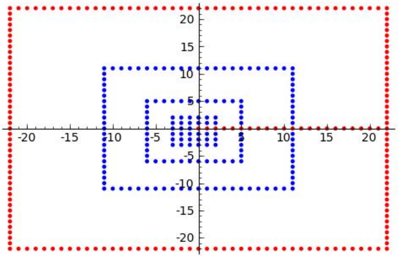


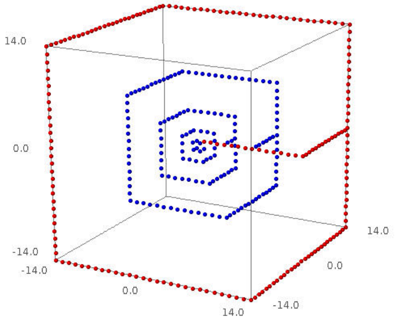
Coordonnées négatives



Experimental resultats

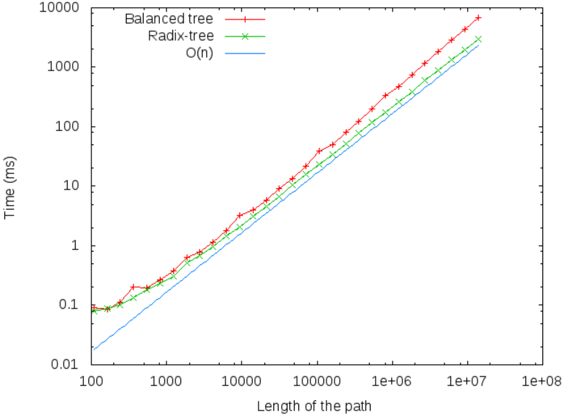
Let $w_{(n,d)} = a_1^k \cdot a_2^k \cdots a_d^k \cdot \bar{a}_1^{2k} \cdot \bar{a}_2^{2k} \cdots \bar{a}_d^{2k} \cdot a_1^{2k} a_2^{2k} \cdots a_{d-1}^{2k} \cdot a_d^k$ where $k = \left\lfloor \frac{n}{5d-1} \right\rfloor$.





Experimental resultats

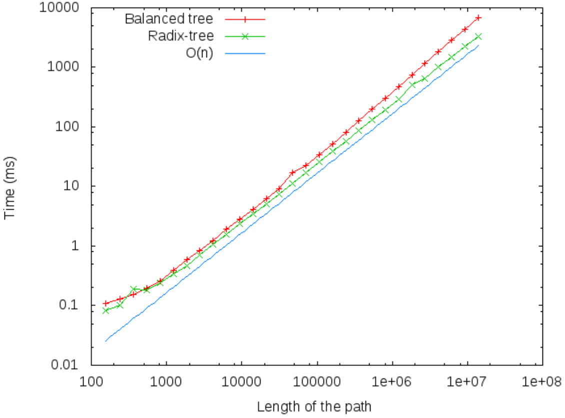
Let $w_{(n,d)} = a_1^k \cdot a_2^k \cdots a_d^k \cdot \bar{a}_1^{2k} \cdot \bar{a}_2^{2k} \cdots \bar{a}_d^{2k} \cdot a_1^{2k} a_2^{2k} \cdots a_{d-1}^{2k} \cdot a_d^k$ where $k = \left\lfloor \frac{n}{5d-1} \right\rfloor$.



$d = 2$

Experimental resultats

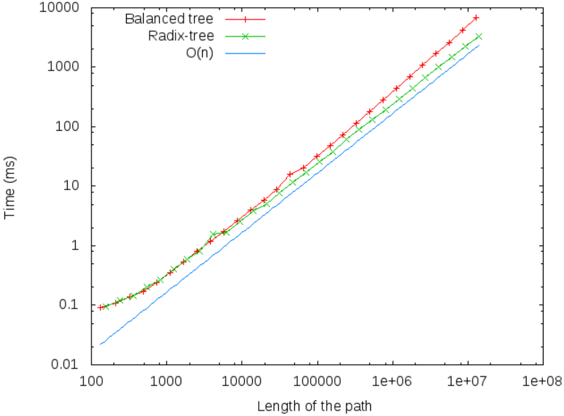
Let $w_{(n,d)} = a_1^k \cdot a_2^k \cdots a_d^k \cdot \bar{a}_1^{2k} \cdot \bar{a}_2^{2k} \cdots \bar{a}_d^{2k} \cdot a_1^{2k} a_2^{2k} \cdots a_{d-1}^{2k} \cdot a_d^k$ where $k = \left\lfloor \frac{n}{5d-1} \right\rfloor$.



$d = 3$

Experimental resultats

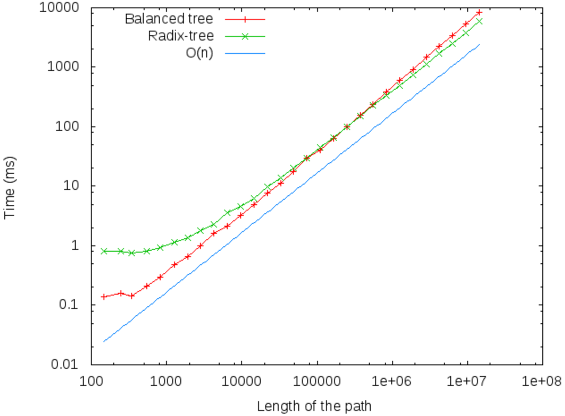
Let $w_{(n,d)} = a_1^k \cdot a_2^k \cdots a_d^k \cdot \bar{a}_1^{2k} \cdot \bar{a}_2^{2k} \cdots \bar{a}_d^{2k} \cdot a_1^{2k} a_2^{2k} \cdots a_{d-1}^{2k} \cdot a_d^k$
where $k = \left\lfloor \frac{n}{5d-1} \right\rfloor$.



$d = 4$

Experimental resultats

Let $w_{(n,d)} = a_1^k \cdot a_2^k \cdots a_d^k \cdot \bar{a}_1^{2k} \cdot \bar{a}_2^{2k} \cdots \bar{a}_d^{2k} \cdot a_1^{2k} a_2^{2k} \cdots a_{d-1}^{2k} \cdot a_d^k$ where $k = \left\lfloor \frac{n}{5d-1} \right\rfloor$.



$d = 10$

Thank you for your attention.