UNIVERSITÉ NAZI BONI

Année universitaire : 2014 – 2015

UNITÉ DE FORMATION ET DE RECHERCHE
EN SCIENCES ET TECHNIQUES

École Doctorale Sciences et Techniques

Laboratoire d'Algèbre, de Mathématiques Discrètes
et d'Informatique (L. A. M. D.I)

MÉMOIRE DU DIPLÔME D'ÉTUDES APPROFONDIES (D.E.A) DE MATHÉMATIQUES APPLIQUÉES

Option : Mathématiques discrètes

Spécialité : Combinatoire des mots

Thème :
Opérations de complétions carrées préfixes-suffixes

Présenté par Thomas Bèbyada OUEDRAOGO

Soutenu publiquement le 13/06/2017 devant le jury composé de :

Président : M. Théodore M. Y. TAPSOBA, Professeur Titulaire - U. N. B

Examinateurs : M. Idrissa KABORE, Maître de Conférences - U. N. B

M. Joseph BAYARA, Maître de Conférences - U. N. B

Directeur de mémoire : M. Idrissa KABORE, Maître de Conférences - U. N. B

OPERATIONS DE COMPLETIONS CARREES PREFIXES-SUFFIXES

OUEDRAOGO Bèbyada Thomas

20 juillet 2017

Dédicaces

Je dédie ce document à :

- mes parents Dominique OUEDRAOGO et Margueritte TAPSOBA,
- mes frères à Bobo-Dioulasso et à Ouagadougou,
- l'ensemble du personnel du Lycée Privé Espoir et du G.A.S.MO II.

Remerciements

Je tiens à remercier mon Directeur de Mémoire, le Professeur Idrissa KABORE, qui m'a permis d'apprendre la combinatoire des mots ainsi que les opérations qui en découlent. Je lui réitère mes sincères remerciements pour sa disponibilité, son sens de l'écoute, son encadrement de qualité qu'il m'a offert, de même que pour ses conseils multiformes.

Mes remerciements vont à l'endroit du Professeur Théodore M. Y. TAPSOBA, Directeur du Laboratoire d'Algèbre, de Mathématiques Discrètes et d'Informatique (L.A.M.D.I), pour ses divers cours, son engagement à promouvoir la discipline et pour son cadre d'études adéquat mis à notre disposition afin de mener à bien nos différents travaux.

Mes honneurs s'en vont à l'endroit du Professeur Joseph BAYARA pour la richesse de son enseignement et pour son engagement à promouvoir l'émergence des algèbres non associatives.

Je remercie le Professeur Sado TRAORE, Directeur de l'Unité de Formation et de Recherche en Sciences et Technique (UFR/ST) pour les efforts colossaux dans la quête d'une bonne formation et d'enseignement de qualité, malgré les ressources limitées.

J'adresse un vibrant hommage aux vaillants Professeurs et Enseignants qui m'ont permis d'avoir une bonne formation et de braver des difficultés.

Au Professeur Nicolas BEDARIDE, j'adresse mes remerciements particuliers pour ces différents cours sur le système dynamique qu'il nous a dispensé en D.E.A.

Particulièrement, je remercie tous mes aînés Doctorants, ainsi que mes camarades du D.E.A pour leurs soutiens et encouragements.

Je remercie notamment KIENTEGA Boucaré, BARRO Moussa et BOGNINI K. Ernest qui m'ont apporté un appui considérable dans les travaux.

Je ne saurai terminer sans adresser un merci franc et sincère à tous ceux et celles qui n'ont ménagés aucun effort dans l'aboutissement de ce travail de recherche.

Résumé

Dans ce mémoire nous considérons une famille d'opérations formelles sur les mots telles que la duplication et la complétion carrée de préfixe-suffixe. Nous montrons que certains mots infinis classiques tels que le mot de Fibonacci, de Thue-Morse, de Stewart's choral et de period-doubling peuvent être générés par des opérations issues de cette famille. Grâce à ces opérations nous élaborons trois algorithmes :

- la première prend en compte un mot fini puis retrouve le préfixe minimal générateur du mot par complétion carrée de suffixe,
- la deuxième prend en compte un mot fini puis, détermine les facteurs minimaux générateurs du mot par complétion carrée de préfixe-suffixe,
- en fin, la troisième prend en compte deux mots de tailles différentes puis, vérifie si le mot le plus court peut générer le plus long par complétion carrée de préfixe-suffixe.

Mots clés : mot fini, mot infini, période, morphisme, duplication, complétion carrée.

Table des matières

Résumé iii

1 Généralités 2

1.1 Généralités sur les mots 2

1.2 Opérations de duplications 5

1.3 Opérations de complétions 8

2 Mots infinis générés par des opérations de complétions carrées 12

2.1 Le mot infini de *Fibonacci* 12

2.2 Le mot infini *Period-doubling* 13

2.3 Le mot infini de *Thue-Morse* 15

2.4 Le mot infini de *Stewart's choral* 21

3 Complétions carrées de préfixes-suffixes des mots finis et algorithmes 23

3.1 Algorithme de recherche 23

3.1.1 Recherche de préfixe minimal pour *SSC* 26

3.1.2 Recherche des facteurs minimaux pour *PSSC* 28

3.2 Algorithme de test 32

Bibliographie 35

INTRODUCTION

La complétion carrée de préfixe, de suffixe et de préfixe-suffixe est une opération formelle sur les mots introduite par Dumitran et Manea dans [9]. Ces opérations sont des extensions d'opérations de duplication de préfixe-suffixe introduites dans [6]. Garcia-Lopez, Manea et Mitrana s'inspirent de l'étude d'un processus biologique qui consiste à créer des répétitions à la fin des séquences génétiques pour introduire la notion de duplication de préfixe-suffixe. Dans leurs travaux, ils montrent premièrement que l'ensemble des mots qui peuvent être générés de façon arbitraire en plusieurs étapes de duplication à partir d'un mot initial donné n'est pas prévisible en général. Ensuite, ils développent un algorithme qui teste en un temps $\mathcal{O}(n^2log(n))$ pour deux mots donnés de tailles différentes, la possibilité de générer le plus long à partir du plus court par duplication de préfixe-suffixe.

D'autres travaux ont été menés sur les mêmes opérations dans [6] en vue de borner la duplication de préfixe-suffixe. Pour ces opérations la longueur du préfixe ou du suffixe dupliqué a été bornée par une constante prédéfinie. Ils ont obtenu des propriétés de certains mots générés par itération bornée de duplications de préfixe-suffixe, à partir d'un ensemble donné de mots. Ensuite, des algorithmes effectifs permettant d'évaluer en un temps $\mathcal{O}(nklog(k))$, si un mot fini donné peut être généré à partir d'un mot plus court par duplication bornée de préfixe-suffixe où la longueur du préfixe ou du suffixe à dupliquer est majorée par k.

Dans notre travail, nous étudions d'autres types d'opérations en relation avec la duplication de préfixe-suffixe. Il s'agit de créer des carrés à l'une des extrémités d'un mot en complétant en un carré, un préfixe ou un suffixe du mot considéré par un facteur. On désigne cette opération par complétion carrée préfixes-suffixes.

Dans ce mémoire nous présenterons d'abord, les différentes opérations de complétions carrées de préfixe-suffixe. Ensuite, nous montrerons comment ces opérations peuvent être utilisées pour générer certains mots infinis classiques tels que le mot de Fibonacci, de Thue-Morse, de Stewart's choral et de period-doubling. Enfin, nous étudierons la possibilité qu'un mot fini soit généré par complétions carrées de préfixes-suffixes à partir de l'un de ses facteurs.

Chapitre 1

Généralités

Dans ce chapitre nous rappelons des notions de base en combinatoire de mots et donnons quelques notations et définitions utiles pour la suite.

1.1 Généralités sur les mots

Un alphabet est un ensemble fini non vide de symboles appelés lettres. Soit $\sum_k = \{0, \cdots, k-1\}$ un alphabet à k lettres où k est un entier naturel non nul. Pour la suite, nous nous restreindrons à l'alphabet binaire noté $\sum = \{0,\ 1\}$. Un mot sur $\sum$ est une suite finie ou infinie de lettres de $\sum$. Le mot vide se définit comme la suite vide, il est noté ε.

La concaténation notée "." est une opération binaire associative admettant pour élément neutre ε. Ainsi, $(\sum^*, ., \varepsilon)$ a une structure de monoïde libre engendrée par $\sum$. On désigne respectivement par $\sum^*$, $\sum^+$ et $\sum^\omega$, l'ensemble des mots finis, des mots finis non vide et des mots infinis sur $\sum$.

Pour un mot fini w sur $\sum$ tel que $w = w_1 w_2 \cdots w_n$, on désigne par $|w|$ la longueur de w et on a $|w| = n$. Pour tout $i \in \{1, \cdots, |w|\}$, le mot w_i, indique la i-ème lettre dans w. On muni $\sum$ de la topologie discrète. On peut munir l'ensemble $\sum^\omega$ d'une distance d définie comme suit : pour deux mots infinis $x = x_1 x_2 \cdots$ et $y = y_1 y_2 \cdots$, $d((x, y) = 2^{-min\{j \in \mathbb{N} / x_j \neq y_j\}}$. Muni de cette distance, $\sum^\omega$ est un espace métrique compact.

Le nombre d'occurrences d'une lettre a dans w est donné par $|w|_a$. Nous notons par $\overline{w}$ le mot obtenu par échange de lettres de w.

Définition 1.1.1. *Soit w et r deux mots sur $\sum$. On dit que r est un facteur de w s'il existe deux mots u, v tels que $w = urv$. Si $u = \varepsilon$ (resp. $v = \varepsilon$) alors r est un préfixe (resp. un suffixe) de w. On dit que r est un facteur propre de w, lorsque $w \neq r$.*

Pour un mot fini $w = w_1 \cdots w_n$ et deux entiers naturels i et j tel que $1 \leq i \leq j \leq n$, le mot $w[i..j]$ désigne le facteur $w_i \cdots w_j$ de w. On désigne par $Fact(w)$ l'ensemble des facteurs de w. L'ensemble des préfixes (resp. des suffixes) de w est noté $Pref(w)$ (resp. $Suff(w)$).

On appelle occurrence d'un facteur r dans w, l'indice $i \leq |w|$-$|r|$ tel que $r = w_i w_{i+1} \cdots w_{i+|r|-1}$.
La puissance d'un mot w se définit comme suite :

$$\begin{cases} w^0 = \varepsilon \\ w^n = ww^{n-1}, \forall n \geq 1. \end{cases}$$

Définition 1.1.2. *On dit qu'un mot w est primitif s'il n'existe pas de mot x et d'entier $k \geq 2$ tel que $w = x^k$.*

Définition 1.1.3. *Soit $w = w_1 w_2 \cdots w_n$ un mot sur $\sum$. Une période de w est un entier p compris entre 1 et $|w|$ tel que $w_i = w_{i+p}$ pour tout $1 \leq i \leq |w| - p$.*

Par convention, la longueur de w est une période de w. On note $Per(w)$ l'ensemble des périodes de w.
Observons que une période ne divise pas nécessairement la longueur du mot.
De plus, tout mot w de période minimale p se factorise comme suit : $w = (xy)^k x$ tel que $k \geq 1$, $p = |xy|$ avec $x \in \sum^+$ et $y \in \sum^*$.
Désignons par $p(w)$ la période minimale de w. On appelle exposant de w, le nombre rationnel noté $e(w) = \dfrac{|w|}{p(w)}$. Si $e(w) \geq 2$, alors w est appelé une répétition.
En général, tout mot w de période minimale p et d'exposant e s'écrit $u^k v$ avec u un mot primitif de longueur p et v un préfixe propre de u et $e(w) = k + \dfrac{|v|}{|u|}$.

Définition 1.1.4. *Un mot w est une répétition d'ordre α ($\alpha \geq 1$ rationnel) si $w = x^n x'$ tel que $\alpha = n + \dfrac{|x'|}{|x|}$, où $n \in \mathbb{N}$ et x' un préfixe de x.*

Exemple 1.1.5. *Le mot $u = entente = (ent)(ent)e$ est une répétition d'ordre $\dfrac{7}{3}$.*
Le mot $v = blablabla = (bla)(bla)(bla)$ est un cube.

Définition 1.1.6. *Soit w un mot fini non vide de longueur n.*
On appelle répétition maximale dans w, toute répétition $w[i..j]$ telle que :
(i) si $i > 1$, alors $p(w[i-1..j]) > p(w[i..j])$,
(ii) si $j < n$, alors $p(w[i..j+1]) > p(w[i..j])$.

Autrement dit, une répétition maximale dans w est une répétition $w[i..j]$ telle qu'aucun facteur de w contenant $w[i..j]$ comme un facteur propre ne possède la même période minimale que $w[i..j]$.

Exemple 1.1.7. *Considérons le mot $w[1..13] = 1011010110110$.*
Le facteur $w[4..8] = 10101 = (10)^2 1$ est une répétition maximale de période 2.
Le préfixe $w[1..11] = 10110101101 = (10110)^2 1$ est une répétition maximale de période 5.
Le suffixe $w[6..13] = 10110110 = (101)^2 10$ est une répétition maximale de période 3.

Remarque 1.1.8. *Toute répétition dans un mot peut être prolongée en une unique répétition maximale.*

Exemple 1.1.9. *Soit* $w = 1011010110110$*. La répétition* 10101 *est une répétition maximale. Elle est obtenue en prolongeant à droite la répétition* 1010 *par la lettre* 1*.*

On appelle chevauchement toute répétition d'ordre α strictement supérieur à 2.

Définition 1.1.10. *Soient* u, r, s, w *quatre mots tels que* $w = rus$ *et* $s \in \sum^{\infty}$*. Notons* g *la dernière lettre de* r *si* $r \neq \varepsilon$ *et* h *la première lettre de* s *si* $s \neq \varepsilon$*. On dit que le mot* u *est un facteur répétitif maximal ou* ***run*** *si :*

(i) u *est périodique,*

(ii) pour $r \neq \varepsilon$, *et* $s \neq \varepsilon$ *les mots* gu *et* uh *ne sont pas périodique de période* p *où* p *est la période minimale de* u*.*

Exemple 1.1.11. *Considérons le mot* $w = 10110101$*. Le facteur* $w[3..4] = (1)^2$ *est un* run *de période* 1*.*
Les facteurs $w[4..7] = (10)^2$ *et* $w[4..8] = (10)^2 1$ *sont des* runs *de période* 2*.*
Le facteur $w[1..6] = (101)^2$ *est un* run *de période* 3*.*

Proposition 1.1.12. *[15] Un run* $w[i..j]$ *est dit maximal si, et seulement si, il ne peut être prolongé ni à gauche, ni à droite pour obtenir un mot de même période i.e :*

(i) $i = 1$ *ou* $w[i-1] \neq w[i+p-1]$*,*

(ii) $j = n$ *ou* $w[j+1] \neq w[j-p+1]$*.*

L'exposant d'un *run* maximal $w[i..j]$ apparaissant dans w est défini par $\dfrac{j-i+1}{p(w[i..j])}$.
La somme des exposants d'un *run maximal* dans un mot de longueur n est bornée par n [2].

Exemple 1.1.13. *Considérons le mot* $w = 10110101$*. Les facteurs* $w[3..4] = (1)^2$, $w[4..8] = (10)^2 1$ *et* $w[1..6] = (101)^2$ *sont des facteurs* ***runs*** *maximaux.*

Définition 1.1.14. *Un mot infini* w *est la limite d'une suite de mots* $(u_n)_{n\geq 0}$ *si pour tout préfixe* v *de* w*, il existe un entier* n_v *tel que pour tout* $n \geq n_v$*,* v *est un préfixe de* u_n*.*

Nous notons $w = \lim\limits_{n\to+\infty} u_n$.
Autrement dit, une suite $(u_n)_{n\geq 0}$ de mots finis converge vers un mot infini w si tout préfixe de w est le préfixe de tous les u_n à l'exception d'un nombre fini.

Définition 1.1.15. *Soient* A *et* B *deux alphabets,* $(A^*, ., \varepsilon_1)$ *et* $(B^*, ., \varepsilon_2)$ *les monoïdes libres associés.*
On appelle morphisme de monoïdes, toute application $\varphi : A^* \longrightarrow B^*$ *vérifiant :*

(i) $\varphi(uv) = \varphi(u)\varphi(v)$*, pour tout* $u, v \in A^*$*,*

(ii) $\varphi(\varepsilon_1) = \varepsilon_2$.

Remarque 1.1.16. *On dit que φ est un endomorphisme lorsque $A = B$.*
La forme itérative du morphisme φ est notée φ^i pour tout $i \in \mathbb{N}$, et définie par :

$$\begin{cases} \varphi^0(a) = a \\ \varphi^{i+1}(a) = \varphi \circ \varphi^i(a), \forall a \in A. \end{cases}$$

Proposition 1.1.17. *Soit φ un morphisme sur $\sum^*$, tels qu'il existe $a \in \sum$ et $x \in \sum^*$ vérifiant les conditions suivantes :*
(i) $\varphi(a) = ax$,
(ii) pour tout $i \geq 1$, $\varphi^i(x) \neq \varepsilon$.
Alors, la suite de mots a, $\varphi(a)$, $\varphi^2(a), \cdots, \varphi^n(a), \cdots$ converge vers un mot infini noté $\varphi^\omega(a)$: $\varphi^\omega(a) = \lim_{i \to +\infty} \varphi^i(a)$. Ce mot infini est un point fixe de φ.

Preuve *: Supposons que les conditions (i) et (ii) sont vérifiés. Alors $\varphi^{i+1}(a) = ax\varphi(x)\varphi^2(x)\cdots\varphi^i(x)$ pour tout $i \geq 0$. On a $\varphi^{i+1}(a) = \varphi^i(a)\varphi^i(x)$. Ainsi, $\varphi^i(a)$ est un préfixe de de $\varphi^{i+1}(a)$. La suite de mots $\varphi^i(a)$ converge alors vers le mot infini noté $\varphi^\omega(a)$.* ■

Définition 1.1.18. *Soit φ un morphisme défini sur $\sum^*$ et u un mot sur $\sum$. On dit que u est un point fixe de φ si $\varphi(u) = u$.*
Un mot w est dit morphique pur s'il existe un morphisme non effaçant φ sur $\sum^*$ et $a \in \sum$ tels que $w = \varphi^\omega(a) = \lim_{n \to +\infty} \varphi^n(a)$.

Définition 1.1.19. *La complexité d'un algorithme calcule le nombre de ressources nécessaire à l'exécution de cet algorithme. La complexité temporelle d'ordre linéaire est noté $\mathcal{O}(n)$.*

1.2 Opérations de duplications

Les opérations de duplication de préfixe, de suffixe, de préfixe-suffixe sont des opérations formelles qui permettent de générer des mots contenant des carrés mais, avec des répétitions d'exposant entier moins élevés.

Définition 1.2.1. *Soit w un mot fini non vide sur $\sum$. La duplication d'un préfixe de w est l'opération qui consiste à générer tout mot de la forme xw tel que x est un préfixe non vide de w.*

L'ensemble de tous les mots générés par une duplication d'un préfixe non vide de w est noté $PD(w)$. Nous avons

$$PD(w) = \left\{xw / w = xw' ; x \in \sum^+\right\}.$$

L'ensemble de tous les mots générés par n itérations de l'opération de duplication de préfixes non vides de w, est noté $PD^n(w)$.
Pour tout $n \geq 1$,

$$PD^n(w) = \bigcup_{0 \leq k \leq n} PD^k(w).$$

Par convention $PD^0(w) = \{w\}$.
On désigne par $PD^*(w)$, l'ensemble des mots générés par itérations finies de l'opération de duplication de préfixes non vides de w. Nous avons

$$PD^*(w) = \bigcup_{n \geq 0} PD^n(w).$$

Exemple 1.2.2. *Considérons les mots $w = 010$ et $s = 01001010$.*
Nous avons $s \in PD(w)$. En effet, en dupliquant d'abord le préfixe $x = 01$ de w on obtient $w_1 = 01010$ comme suit :

$$w = \underline{01}0 \xmapsto{PD} \underline{01}010 = (01)^2 0 = w_1 \in PD(w).$$

En dupliquant ensuite, le préfixe $x_1 = 010$ de w_1 on obtient alors le mot $01001010 = s$ comme suit :

$$w_1 = \underline{010}10 \xmapsto{PD} \underline{010}01010 = (010)^2 10 = s \in PD(w_1) \subset PD^2(w).$$

Ainsi, $s \in PD^2(w)$.

Définition 1.2.3. *Soit w un mot fini non vide sur $\sum$. La duplication d'un suffixe de w est l'opération qui consiste à générer tout mot de la forme wx tel que x est un suffixe non vide de w.*

L'ensemble de tous les mots générés par une duplication d'un suffixe non vide de w, est noté $SD(w)$. Nous avons

$$SD(w) = \left\{ wx / w = w' x ;\ x \in \sum^+ \right\}.$$

L'ensemble de tous les mots générés par n itérations de l'opération de duplication de suffixes non vides de w, est noté $SD^n(w)$.
Pour tout $n \geq 1$,

$$SD^n(w) = \bigcup_{0 \leq k \leq n} SD^k(w).$$

Par convention $SD^0 = \{w\}$.
On désigne par $SD^*(w)$, l'ensemble de tous les mots générés par itérations finies de l'opération de duplication de suffixes non vides de w. Nous avons

$$SD^*(w) = \bigcup_{n \geq 0} SD^n(w).$$

Exemple 1.2.4. *Considérons les mots* $u = 01010010$ *et* $w = 010$. *Le mot* u *est dans* $SD^*(w)$. *En effet, en dupliquant premièrement le suffixe* $x = 10$ *de* w, *on obtient* $w_1 = 01010$ *comme suit :*

$$w = 0\underline{10} \xmapsto{SD} 010\underline{10} = 0(10)^2 = w_1 \in SD(w).$$

En dupliquant ensuite, le suffixe $x_1 = 010$ *de* $w_1 = 0(10)^2 = 01010$ *on obtient* u *comme suit :*

$$w_1 = 01\underline{010} \xmapsto{SD} 01010\underline{010} = 01(010)^2 = u \in SD(w_1).$$

Ainsi, $u \in SD^2(w)$.

Définition 1.2.5. *Soit* w *un mot fini non vide sur* $\sum$. *La duplication de préfixe-suffixe est la combinaison des deux opérations.*

L'ensemble des mots générés par une duplication de préfixe-suffixe de w, est noté $PSD(w)$. Nous avons

$$PSD(w) = PD(w) \cup SD(w).$$

L'ensemble des mot générés par n itérations de l'opération de duplication de préfixe-suffixe de w est noté $PSD^n(w)$.
Pour tout $n \geq 1$,

$$PSD^n(w) = \bigcup_{k+l+m=n} PD^k\left(SD^l\left(PD^m(w)\right)\right).$$

On désigne par $PSD^*(w)$, l'ensemble des mots générés par itérations finies de l'opération de duplication de préfixe-suffixe de w. Nous avons

$$PSD^*(w) = \bigcup_{n\geq 0} PSD^n(w).$$

Exemple 1.2.6. *Les mots* $s = 01001010$, $u = 01010010$ *et* $z = 010100$ *sont dans* $PSD^*(w)$ *avec* $w = 010$. *En effet, comme* $u \in SD(w)$ *et* $s \in PD(w)$ *alors* $u, s \in PSD^*(w)$. *Pour le mot* z, *en dupliquant premièrement le préfixe* $x = 01$ *de* w *on obtient le mot* $w_1 = 01010$ *comme suit :*

$$w = \underline{01}0 \xmapsto{PD} 10\underline{01}0 = (01)^2 0 = w_1.$$

En dupliquant ensuite, le suffixe $x_1 = 0$ *de* w_1 *on obtient le mot* z *comme suit :*

$$w_1 = 0101\underline{0} \xmapsto{SD} 0101\underline{0}0 = 0101(0)^2 = z.$$

Ainsi, $z \in PD(w) \bigcup SD(w) \subset PSD^*(w)$.

Remarque 1.2.7. *La longueur du préfixe ou du suffixe à dupliquer n'est pas bornée par la longueur du générateur initial.*
En chacune des étapes de la duplication, le choix du préfixe ou du suffixe à dupliquer n'est pas unique à moins que le mot w *se réduit à une lettre de l'alphabet.*

1.3 Opérations de complétions

La complétion carrée est une extension de l'opération de duplication. A partir de cette opération, nous considérons la possibilité de créer des carrés à l'une des extrémités d'un mot en complétant juste, un préfixe ou un suffixe considéré en un carré. Cette opération s'effectue en utilisant un facteur du préfixe ou du suffixe à compléter en un carré.

Définition 1.3.1. *Soit w un mot fini non vide sur $\sum$. On appelle complétion carrée d'un préfixe de w, l'opération qui consiste à générer tout mot de la forme xw où yxy est un préfixe de w.*

L'ensemble de tous les mots générés par une complétion carrée d'un préfixe non vide de w est noté $PSC(w)$. Nous avons

$$PSC(w) = \left\{xw/w = yxyw'; x \in \sum^{+}, y \in \sum^{*}\right\}.$$

On désigne par $PSC^n(w)$, l'ensemble de tous les mots générés par n itérations de l'opération de complétion carrée de préfixes de w.
Pour tout $n \geq 1$,

$$PSC^n(w) = \bigcup_{0 \leq k \leq n} PSC^k(w).$$

Par convention $PSC^0(w) = \{w\}$.
L'ensemble de tous les mots générés par itérations finies de l'opération de complétion carrée de préfixes de w est noté $PSC^*(w)$. Nous avons

$$PSC^*(w) = \bigcup_{n \geq 0} PSC^n(w).$$

Exemple 1.3.2. *Le mot $v = 01010 \in PSC(w)$ où $w = 010$. En effet, en prenant $x = 1$, $y = 0$ et en considérant le préfixe $yxy = 010$ de w on obtient par une première complétion le mot w_1 comme suit :*

$$w = 0\underline{1}0 \stackrel{PSC}{\longmapsto} \underline{1}010 = (10)^2 = w_1.$$

Ainsi, le mot $w_1 = (10)^2$ est dans $PSC(w)$.
Prenons ensuite $x = 0$ et $y = 1$ et considérons le préfixe $yxy = 101$, en complétant yxy en carré nous obtenons v comme suit :

$$w_1 = 1\underline{0}10 \stackrel{PSC}{\longmapsto} \underline{0}1010 = (01)^2 0 = v.$$

Ainsi, $(01)^2 0 = v \in PSC^2(w)$.

Définition 1.3.3. *Soit w un mot fini non vide sur $\sum$. La complétion carrée d'un suffixe est l'opération qui consiste à générer tout mot de la forme wx où yxy est un suffixe de w.*

L'ensemble des mots générés par une complétion carrée d'un suffixe de w est noté $SSC(w)$. Nous avons

$$SSC(w) = \left\{wx/w = w'yxy;\ x \in \sum^{+}, y \in \sum^{*}\right\}.$$

L'ensemble de tous les mots générés par n itérations de l'opération de complétion carrée de suffixes de w est noté $SSC^n(w)$.
Pour tout $n \geq 1$,

$$SSC^n(w) = \bigcup_{0\leq k\leq n} SSC^k(w).$$

Par convention $SSC^0(w) = \{w\}$.
On désigne par $SSC^*(w)$, l'ensemble de tous les mots générés par itérations finies de l'opération de complétion carrée de suffixes de w. Nous avons

$$SSC^*(w) = \bigcup_{n\geq 0} SSC^n(w).$$

Exemple 1.3.4. *Le mot $m = 010101 \in SSC^*(w)$ avec $w = 010$.*
Prenons $y = 0$, $x = 1$ et considérons le suffixe $yxy = 010$ à compléter en un carré. Une première complétion de yxy permet d'obtenir w_1 comme suit :

$$w = 0\underline{1}0 \overset{SSC}{\longmapsto} 0\underline{1}0\underline{1} = (01)^2 = w_1$$

Ainsi, le mot $w_1 = (01)^2$ est dans $SSC(w)$.
Prenons ensuite $x = 01$, $y = \varepsilon$ et considérons le suffixe $yxy = 01$ de w_1 à compléter en un carré. On obtient m en complétant yxy comme suit :

$$w_1 = 01\underline{01} \overset{SSC}{\longmapsto} 0101\underline{01} = 01(01)^2 = m.$$

Ainsi, le mot $(010)^2 = m \in SSC^2(w) \subset SSC^*(w)$.

Définition 1.3.5. *Soit w un mot fini non vide sur $\sum$. La complétion carrée de préfixe-suffixe est la combinaison des opérations de complétions carrées d'un préfixe ou d'un suffixe de w.*

L'ensemble des mots générés par complétion carrée d'un préfixe ou d'un suffixe de w est noté $PSSC(w)$. Nous avons

$$PSSC(w) = PSC(w) \cup SSC(w).$$

L'ensemble des mots générés par n itérations de l'opération de complétion carrée préfixes-suffixes est noté $PSSC^n(w)$. Pour tout $n \geq 1$,

$$PSSC^n(w) = \bigcup_{k+l+m=n} PSC^k\left(SSC^l\left(PSC^m(w)\right)\right).$$

On désigne par $PSSC^*$ l'ensemble des mots générés par itérations finies de complétion carrée de préfixes-suffixes de w.
Nous avons :

$$PSSC^*(w) = \bigcup_{n\geq 0} PSSC^n(w).$$

Exemple 1.3.6. *Les mots $m = 010101$, $v = 01010$ et $M = 10101$ appartiennent à $PSSC^*(w)$ avec $w = 010$. En effet, comme $m = (10)^3$ et $v = 01010$ appartient à $PSC^*(w)$ et à $SSC^*(w)$ alors m et v appartient à $PSSC^*(w)$.*
Considérons le mot $M = 10101$.
Prenons $x = 1$ et $y = 0$, on a $yxy = 010 \in Pref(w)$. Une première complétion carrée de préfixe donne :

$$w = 0\underline{1}0 \overset{PSC}{\longmapsto} \underline{1}010 = (10)^2 = w_1.$$

Prenons ensuite, $x = 1$ et $y = 0$, on a $yxy = 010 \in Suff(w_1)$. Une seconde complétion donne :

$$w_1 = 10\underline{1}0 \overset{SSC}{\longmapsto} 1010\underline{1} = 1(01)^2 = M.$$

Ainsi, le mot $M \in PSSC^*(w)$.

Remarque 1.3.7. *En particulier, dans le choix du préfixe ou du suffixe de la forme yxy à compléter en un carrée, si $y = \varepsilon$ alors l'opération de complétion carrée se réduit à une opération de duplication.*

Définition 1.3.8. *On dit qu'un mot infini w est généré par l'opération*

$$\theta \in \{PD,\, SD,\, PSD,\, PSC,\, SSC,\, PSSC\}$$

.i.e, $w \in \theta^\omega$ s'il existe une suite finie de mots $(w_n)_{n\in\mathbf{N}}$ tel que w_i est un préfixe de w et $w_{i+1} \in \theta(w_i)$ pour tout $i \in \mathbf{N}$. On note

$$w = \lim_{n\to+\infty} w_n,$$

le mot infini généré par itération de ces différentes opérations.

Lemme 1.3.9. *Soient x, y et z trois mots finis tels que $x \in Pref(y)$ et $y \in Pref(z)$. Si $z \in SSC^*(x)$ alors $z \in SSC^*(y)$.*

<u>**Preuve**</u> : Nous distinguons deux cas.
Supposons $x = y$. Alors $SSC^*(x) = SSC^*(y)$. Ainsi, $z \in SSC^*(x)$ implique $z \in SSC^*(y)$.
Supposons $x \neq y$ et $z \in SSC^*(x)$. Alors il existe une suite de mots $(x_i)_{0\leq i\leq n}$ tels que :

$$\begin{cases} x_0 = x \\ x_i \in SSC^*(x_{i-1}) \quad \text{pour tout } 1 \leq i \leq n \\ x_n = z. \end{cases}$$

Par hypothèse $x = x_0$ est un préfixe de y, lui même préfixe de $z = x_n$. Comme $x \in Pref(y)$ et $z \in SSC^*(x)$, il existe un entier k avec $1 \leq k \leq n$ tel que x_{k-1} soit un préfixe de y et y un préfixe de x_k. Nous écrivons $y = x_{k-1}u$, avec $u \in \sum^*$.

Puisque $x_k \in SSC^*(x_{k-1})$, il existe r et s tels que $x_{k-1} = wrsr$ et $x_k = wrsrs$. De l'expression de x_{k-1} dans y, nous avons $y = x_{k-1}u = wrsru$. Comme y est un préfixe de x_k alors nous pouvons écrire

$$x_k = yt = wrsrut = wrsrs$$

avec $t \in \sum^*$. Ainsi $s = ut$. Par suite, en remplaçant s par son expression dans x_k, on obtient :

$$x_k = wr(ut)r(ut) = w(ru)t(ru)t = yt.$$

Comme $y = w(ru)t(ru) \overset{SSC}{\longmapsto} w(ru)t(ru)t = yt = x_k$ alors $x_k \in SSC^*(y)$. Par suite, pour tout $j \geq k$, $x_j \in SSC^*(y)$. En particulier $x_n = z \in SSC^*(y)$. ■

Chapitre 2

Mots infinis générés par des opérations de complétions carrées

Dans ce chapitre, nous utiliserons les opérations de complétions carrées de préfixes-suffixes pour générer certains mots infinis classiques tels que le mot de Fibonacci, Period-doubling, Thue-morse et Stewarst's choral. Nous montrerons d'une part, comment générer les trois premiers mots par complétions carrées de suffixes et d'autre part, générer le mot de Stwarst's choral par l'opération de duplication de suffixes.

2.1 Le mot infini de *Fibonacci*

Le mot *f de Fibonacci* est le mot infini engendré par le morphisme φ_f en 0 et défini par : $\varphi_f(0) = 01$ et $\varphi_f(1) = 0$.

$$f = \lim_{n \to +\infty} \varphi_f^n(0).$$

Le mot de *Fibonacci* est également défini comme limite de la suite de mots f_n définie par : $f_0 = 0$, $f_1 = 01$ et $f_k = f_{k-1}f_{k-2}$ pour $k \geq 2$.

Proposition 2.1.1. *Pour tout* $n \geq 4$ *il existe* $f_n^{'} \in SSC^*(f_n)$ *tel que* $f_{n+1} \in Pref(f_n^{'})$ *et* $f_n^{'} \in Pref(f_{n+2})$.

<u>**Preuve**</u>: Nous procédons par récurrence sur $n \geq 4$.
Pour $n = 4$, considérons $f_4 = f_3f_2$, le mot initial de la dérivation et f_2 le suffixe de f_4 à compléter en un carré. Par SSC nous avons

$$f_4 = f_3f_2 \overset{SSC}{\longmapsto} f_3f_2f_2 \overset{SSC}{\longmapsto} f_3f_2f_2f_2 = f_4f_2f_1f_0 = f_4f_3f_0.$$

Posons $f_4^{'} = f_4f_3f_0 = f_5f_0$, on a $f_4^{'} \in SSC^2(f_4) \subset SSC^*(f_4)$. Par ailleurs $f_4^{'} = f_5f_0$ donc $f_5 \in Pref(f_4^{'})$ et $f_4^{'} \in Pref(f_6)$. La propriété est donc vraie pour $n = 4$.
Supposons qu'il existe $f_n^{'} \in SSC^*(f_n)$ tel que $f_{n+1} \in Pref(f_n^{'})$ et $f_n^{'} \in Pref(f_{n+2})$

Puisque $f_{n+1} = f_n f_{n-1}$, on applique alors l'hypothèse de récurrence à f_{n+1} en dupliquant le suffixe f_{n-1}. Nous avons

$$f_{n+1} = f_n f_{n-1} \xmapsto{SSC} f_n f_{n-1} f_{n-1} \xmapsto{SSC} f_n f_{n-1} f_{n-1} f_{n-1} = f_{n+1} f_{n-1} f_{n-2} f_{n-3} = f_{n+1} f_n f_{n-3}.$$

Par ailleurs nous avons $f_{n+1} f_n f_{n-3} = f_{n+2} f_{n-3}$. Ainsi, en posant $f'_{n+1} = f_{n+2} f_{n-3}$, il vient que $f'_{n+1} \in SSC^2(f_{n+1}) \subset SSC^*(f_{n+1})$. De plus $f_{n+2} \in pref(f'_{n+1})$ et $f'_{n+1} \in Pref(f_{n+3})$. ■

Proposition 2.1.2. *Le mot f de Fibonacci est dans* SSC^ω.

<u>**Preuve**</u> : On génère le mot infini f de *Fibonacci* par complétions carrées de suffixe à partir du préfixe initial f_4. On génère le mot f'_4 grâce à la Proposition 2.1.1. Puisque $f_5 \in Pref(f'_4)$ et $f'_4 \in SSC^*(f_4)$, on déduit par le Lemme 1.3.9 que $f'_4 \in SSC^*(f_5)$. De la même manière on dérive le mot f'_5 par SSC du mot f_5, et on vérifie ensuite par le Lemme 1.3.9 que $f'_5 \in SSC^*(f_6)$.
En itérant le processus, on obtient la suite $(f'_n)_n$ pour $n \geq 4$, qui est une suite qui converge vers le mot infini f de Fibonacci. ■

2.2 Le mot infini *Period-doubling*

Le mot *Period-doubling* d est le mot infini engendré par le morphisme φ_d défini par $\varphi_d(0) = 01$ et $\varphi_d(1) = 00$:

$$d = \lim_{n \to +\infty} \varphi_d^n(0).$$

Proposition 2.2.1. *Pour tout* $n \geq 3$, $d_{n+1} \in SSC^*(d_n)$ *implique* $d_{n+2} \in SSC^*(d_{n+1})$.

<u>**Preuve**</u> : D'abord vérifions que pour $n = 3$, $d_4 \in SSC^*(d_3)$ implique $d_5 \in SSC^*(d_4)$. Pour ce faire, nous choisissons respectivement $\underline{01}$, $1\underline{0}1$, $\underline{0}$, $\underline{0}$, $0\underline{100}0$ les suffixes de la forme yxy à compléter en carré selon les étapes de la complétion. Par souci de clarté, nous avons souligné le facteur x dans chaque suffixe de la forme yxy. On a :

$$d_3 = 010001\underline{01} \xmapsto{SSC} 01000101\underline{01} = x_1 = 010001(01)^2$$
$$x_1 = 01000101\underline{0}1 \xmapsto{SSC} 0100010101\underline{0} = x_2 = 0100010(10)^2$$
$$x_2 = 0100010101\underline{0} \xmapsto{SSC} 01000101010\underline{0} = x_3 = 0100010101(0)^2$$
$$x_3 = 01000101010\underline{0} \xmapsto{SSC} 010001010100\underline{0} = x_4 = 01000101010(0)^2$$
$$x_4 = 0100010101\underline{100}0 \xmapsto{SSC} 0100010101000\underline{100} = x_4 = 01000101(0100)^2 = d_4.$$

Ainsi, nous avons $d_4 \in SSC^5(d_3) \subset SSC^*(d_3)$. On montre maintenant que
$d_5 \in SSC^*(d_4)$. Comme suffixes à compléter en un carrés, nous prenons respectivement,
$\underline{\varphi_d(01)}$, $\varphi_d(1)\underline{\varphi_d(0)}\varphi_d(1)$, $\underline{\varphi_d(0)}$, $\underline{\varphi_d(0)}$, $\varphi_d(0)\underline{\varphi_d(100)}\varphi_d(0)$, les images respectives de

01, 101, 0, 0 et 100 par le morphisme φ_d. La dérivation donne :

$d_4 = \varphi_d^4(0) = \varphi_d(d_3) = \varphi_d(010001)\underline{\varphi_d(01)} \overset{SSC}{\longmapsto} \varphi_d(010001)\varphi_d(01)\underline{\varphi_d(01)} = x_1$

$x_1 = \varphi_d(0100010)\varphi_d(1)\underline{\varphi_d(0)}\varphi_d(1) \overset{SSC}{\longmapsto} \varphi_d(0100010)\varphi_d(1)\varphi_d(0)\varphi_d(1)\underline{\varphi_d(0)} = x_2$

$x_2 = \varphi_d(0100010101)\underline{\varphi_d(0)} \overset{SSC}{\longmapsto} \varphi_d(0100010101)\varphi_d(0)\underline{\varphi_d(0)} = x_3$

$x_3 = \varphi_d(01000101010)\underline{\varphi_d(0)} \overset{SSC}{\longmapsto} \varphi_d(01000101010)\varphi_d(0)\underline{\varphi_d(0)} = x_4$

$x_4 = \varphi_d(01000101)\varphi_d(0)\underline{\varphi_d(100)}\varphi_d(0) \overset{SSC}{\longmapsto} \varphi_d(01000101)\varphi_d(0)\varphi_d(100)\varphi_d(0)\underline{\varphi_d(100)}$

$x_5 = \varphi_d(01000101)\varphi_d(0)\varphi_d(100)\varphi_d(0)\underline{\varphi_d(100)} = \varphi_d(0100010101000100) = \varphi_d(d_4) = \varphi_d^5(0) = d_5$.
Ainsi, $d_5 \in SSC^5(d_4) \subset SSC^*(d_4)$.
Supposons que la propriété reste vraie jusqu'à l'ordre n i.e., $d_{n+1} \in SSC^*(d_n)$ implique $d_{n+2} \in SSC^*(d_{n+1})$. D'après l'hypothèse de récurrence, il nous suffit de montrer que $d_{n+3} \in SSC^*(d_{n+2})$.
Pour ce faire, en fonction des étapes de la dérivation nous prenons respectivement

$$\underline{\varphi_d^{n-1}(01)},\ \varphi_d^{n-1}(1)\underline{\varphi_d^{n-1}(0)}\varphi_d^{n-1}(1),\ \underline{\varphi_d^{n-1}(0)},\ \underline{\varphi_d^{n-1}(0)},\ \varphi_d^{n-1}(0)\underline{\varphi_d^{n-1}(100)}\varphi_d^{n-1}(0),$$

les suffixes de la forme yxy à compléter en un carrés. Par dérivations successives on a :
$d_{n+2} = \varphi_d^{n+2}(0) = \varphi_d^{n-1}(d_3) = \varphi_d^{n-1}(01000101) = \varphi_d^{n-1}(01001)\underline{\varphi_d^{n-1}(01)} \overset{SSC}{\longmapsto}$

$\varphi_d^{n-1}(01001)\varphi_d^{n-1}(01)\underline{\varphi_d^{n-1}(01)} = \varphi_d^{n-1}(010010101) = x_1$

$x_1 = \varphi_d^{n-1}(010010)\varphi_d^{n-1}(1)\underline{\varphi_d^{n-1}(0)}\varphi_d^{n-1}(1) \overset{SSC}{\longmapsto}$

$\varphi_d^{n-1}(010010)\varphi_d^{n-1}(1)\varphi_d^{n-1}(0)\varphi_d^{n-1}(1)\underline{\varphi_d^{n-1}(0)} = x_2$

$x_2 = \varphi_d^{n-1}(010010101)\underline{\varphi_d^{n-1}(0)} \overset{SSC}{\longmapsto} \varphi_d^{n-1}(010010101)\varphi_d^{n-1}(0)\underline{\varphi_d^{n-1}(0)} = x_3$

$x_3 = \varphi_d^{n-1}(01000101010)\underline{\varphi_d^{n-1}(0)} \overset{SSC}{\longmapsto} \varphi_d^{n-1}(01000101010)\varphi_d^{n-1}(0)\underline{\varphi_d^{n-1}(0)} = x_4$

$x_4 = \varphi_d^{n-1}(01000101)\varphi_d^{n-1}(0)\underline{\varphi_d^{n-1}(100)}\varphi_d^{n-1}(0) \overset{SSC}{\longmapsto}$

$\varphi_d^{n-1}(01000101)\varphi_d^{n-1}(0)\varphi_d^{n-1}(100)\varphi_d^{n-1}(0)\underline{\varphi_d^{n-1}(100)} = \varphi_d^{n-1}(0100010101000100) =$

$x_5 = \varphi_d^{n-1}(d_4) = \varphi_d^{n+3}(0) = d_3$

.

Ainsi, $d_{n+3} \in SSC^5(d_{n+2}) \subset SSC^*(d_{n+2})$.
Par conséquent, $d_{n+1} \in SSC^*(d_n)$ implique $d_{n+2} \in SSC^*(d_{n+1})$. ■

Proposition 2.2.2. *Le mot infini Period-doubling* d *est dans* SSC^ω.

<u>**Preuve**</u>: Grâce à la Proposition 2.2.1, nous générons par SSC une suite finie de mots $(d_n)_n$ qui converge vers le mot infini d nommé *Period-doubling*. Ainsi, il suffit de commencer la dérivation avec le préfixe d_2. ■

2.3 Le mot infini de *Thue-Morse*

Le mot t de *Thue-Morse* est le mot infini engendré par le morphisme φ_t défini par : $\varphi_t(0) = 01$ et $\varphi_t(1) = 10$. On a :

$$t = \lim_{n \to +\infty} \varphi_t^n(0).$$

Soit t_n un préfixe de t. Nous désignons par $t_n^{'}$ le mot obtenu de t_n en supprimant la dernière lettre et $\bar{t}_n$ le mot obtenu de t_n par échange de lettres.

Lemme 2.3.1. *Soit n un entier naturel. Alors, $t_n = \varphi^n(0)$ se termine par* 0 *si n est pair et par* 1 *sinon.*

<u>**Preuve**</u>: Par récurrence sur n, montrons que t_{2n} se termine par 0.
La propriété est vraie pour $n = 0$ et $n = 1$ puisque $t_0 = 0$ et $t_2 = 0110$.
Supposons la propriété vraie pour tout $n \geq 1$ et vérifions la pour $n + 1$. Nous avons

$$t_{2(n+1)} = t_{2n+2} = \varphi_t^{2n+2}(0) = \varphi_t^2\left(\varphi_t^{2n}(0)\right) = \varphi_t^2(t_{2n}).$$

Comme par hypothèse de récurrence t_{2n} se termine 0, on peut écrire $t_{2n} = u0$ avec $u \in \sum^*$. Par suite on a :

$$t_{2(n+1)} = \varphi_t^2(u0) = \varphi_t^2(u)\varphi_t^2(0) = \varphi_t^2(u)t_2 = \varphi_t^2(u)0110.$$

Ainsi, $t_{2(n+1)}$ se termine par 0.
De façon similaire, on montre que t_{2n+1} se termine par 1. ■

Proposition 2.3.2. *Soit t_n, un préfixe du mot infini t de Thue-Morse. Pour tout $n \geq 2$, $t_n^{'} \in SSC^*(t_{n-1})$.*

<u>**Preuve**</u>: Du Lemme 2.3.1, nous distinguons deux cas suivant la parité de n.
Pour $n = 1$ et $n = 2$, on a :

$$t_1 = 0\underline{1} \overset{SSC}{\longmapsto} 01\underline{1} = 0(1)^2 = t_2^{'} \in SSC(t_1) \subset SSC^*(t_1)$$

et

$$t_2 = 01\underline{10} \overset{SSC}{\longmapsto} 0110\underline{10} = 01101\underline{0} \overset{SSC}{\longmapsto} 011010\underline{0} = 01101(0)^2 = t_3' \in SSC^2(t_2) \subset SSC^*(t_2).$$

La propriété est donc vérifiée pour $n \in \{1, 2\}$.
Supposons que la propriété est vérifiée jusqu'à l'ordre n i.e, $t_{2n}' \in SSC^*(t_{2n-1})$ et $t_{2n+1}' \in SSC^*(t_{2n})$.
Montrons que $t_{2n+2}' \in SSC^*(t_{2n+1})$ et $t_{2n+3}' \in SSC^*(t_{2n+2})$.
Par ailleurs, une décomposition de t_{2n+2}' et t_{2n+1} donne :

$$t_{2n+2}' = t_{2n+1}\bar{t}_{2n+1}' = t_{2n}\bar{t}_{2n}\bar{t}_{2n}t_{2n}' \quad \text{et} \quad t_{2n+1} = t_{2n}\bar{t}_{2n} = t_{2n}\bar{t}_{2n-1}t_{2n-1}.$$

Par complétions carrées de suffixes, on a :

$$t_{2n+1} = t_{2n}\underline{\bar{t}_{2n}} \overset{SSC}{\longmapsto} t_{2n}\bar{t}_{2n}\underline{\bar{t}_{2n}} = t_{2n+1}\bar{t}_{2n-1}\underline{t_{2n-1}} \overset{SSC}{\longmapsto} t_{2n+1}\bar{t}_{2n-1}t_{2n-1}\underline{t_{2n-1}}.$$

D'après l'hypothèse de récurrence, $t_{2n-1} \overset{SSC^*}{\longmapsto} t_{2n}'$. Par suite :

$$t_{2n+1}\bar{t}_{2n-1}t_{2n-1}t_{2n-1} = t_{2n+1}\bar{t}_{2n}[t_{2n-1}] \overset{SSC^*}{\longmapsto} t_{2n+1}\bar{t}_{2n}[t_{2n}'] = t_{2n+1}\bar{t}_{2n+1}' = t_{2n+2}'.$$

Ainsi, $t_{2n+2}' \in SSC^*(t_{2n+1})$.
De même, montrons que $t_{2n+3}' \in SSC^*(t_{2n+2})$.
Une décomposition de t_{2n+3}' et t_{2n+2} donne :

$$t_{2n+3}' = t_{2n+2}\bar{t}_{2n+2}' = t_{2n+1}\bar{t}_{2n+1}\bar{t}_{2n+1}t_{2n+1}'$$

et

$$t_{2n+2} = t_{2n+1}\bar{t}_{2n+1}.$$

Par complétions carrées de suffixes, on a :

$$t_{2n+2} = t_{2n+1}\underline{\bar{t}_{2n+1}} \overset{SSC^*}{\longmapsto} t_{2n+1}\bar{t}_{2n+1}\underline{\bar{t}_{2n+1}} = t_{2n+2}\bar{t}_{2n}\underline{t_{2n}} \overset{SSC}{\longmapsto} t_{2n+2}\bar{t}_{2n}t_{2n}\underline{t_{2n}}.$$

D'après l'hypothèse de récurrence $t_{2n} \overset{SSC^*}{\longmapsto} t_{2n+1}'$. Par conséquent

$$t_{2n+2}\bar{t}_{2n}t_{2n}[t_{2n}] \overset{SSC^*}{\longmapsto} t_{2n+2}\bar{t}_{2n}t_{2n}[t_{2n+1}'] = t_{2n+2}t_{2n+1}t_{2n+1}' = t_{2n+2}t_{2n+2}' = t_{2n+3}'.$$

Ainsi, $t_{2n+3}' \in SSC^*(t_{2n+2})$. ■

Proposition 2.3.3. *Pour tout* $n \geq 6$, $t_n' \overset{SSC}{\longmapsto} t_n\varphi_t^{n-6}(1)$.

<u>Preuve</u> : Pour $n = 6$, il suffit de vérifier que $t_6\varphi_t^0(1) \in SSC(t_6')$. En effet, considérons le mot $t_6 = \varphi_t^6(0) = t_5\bar{t}_5 = t_5\bar{t}_4t_3\bar{t}_1010110$. Comme 10110 est un suffixe de t_6, il existe alors $z \in \sum^+$ tels que $t_6 = z10110$ et $t_6' = z1011$. Comme $t_n = \varphi_t^n(0)$ alors t_6' peut s'écrire sous la forme

$$t_6' = z\varphi_t^0(1)\varphi_t^0(0)\varphi_t^0(1)\varphi_t^0(1).$$

Par une complétion carrée de suffixe, on a :

$$t_6' = z\varphi_t^0(1)\underline{\varphi_t^0(0)\varphi_t^0(1)}\varphi_t^0(1) \overset{SSC}{\longmapsto} z\varphi_t^0(1)\varphi_t^0(0)\varphi_t^0(1)\varphi_t^0(1)\underline{\varphi_t^0(0)\varphi_t^0(1)} = t_6\varphi_t^0(1).$$

Par suite, $t_6\varphi_t^0(1) \in SSC(t_6')$.
Supposons maintenant que $t_n' \overset{SSC}{\longmapsto} t_n\varphi^{n-6}(1)$. Une décomposition de t_n donne :

$$t_n = \varphi_t^{n-6}(\varphi_t^6(0)) = \varphi_t^{n-6}(t_6) = \varphi_t^{n-6}(z10110) = \varphi_t^{n-6}(z)\varphi_t^{n-6}(10)\varphi_t^{n-6}(1)\varphi_t^{n-6}(10).$$

• Si n est pair alors, t_n se termine par 0 d'après le Lemme 2.3.1 et on a :

$$t_n = \varphi^{n-6}(z)\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(0)1[\underline{0\varphi^{n-6}(1)}]\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(0)10.$$

et

$$t_n' = \varphi^{n-6}(z)\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(0)1[\underline{0\varphi^{n-6}(1)}]\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(0)1$$

• Si n est impair alors, t_n se termine par 1 d'après le Lemme 2.3.1, et on a :

$$t_n = \varphi^{n-6}(z)\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(1)0[\underline{1\varphi^{n-6}(1)}]\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(1)01.$$

et

$$t_n' = \varphi^{n-6}(z)\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(1)0[\underline{1\varphi^{n-6}(1)}]\varphi^{n-6}(1)\varphi^{n-7}(0)\cdots\varphi(1)0$$

A l'ordre $n+1$, nous distinguons deux cas.
• Supposons n pair.
La valeur $n+1$ est impair. Alors, t_{n+1} se termine par un 1. On a

$$t_{n+1} = \varphi^{n-5}(z)\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(1)0[1\varphi^{n-5}(1)]\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(1)01$$

et

$$t_{n+1}' = \varphi^{n-5}(z)\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(1)0[\underline{1\varphi^{n-5}(1)}]\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(1)0.$$

En prenant $x = 1\varphi^{n-5}(1)$ et $y = \varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(1)0$, on obtient le suffixe yxy à compléter en un carré. Une étape de dérivation par SSC permet d'obtenir $t_{n+1}' \overset{SSC}{\longmapsto} t_{n+1}\varphi^{n-5}(1)$.
• Supposons n impair.
La valeur $n+1$ est pair, il vient que t_{n+1} se termine par 0 d'après le Lemme 2.3.1. On a :

$$t_{n+1} = \varphi^{n-5}(z)\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(0)1[0\varphi^{n-5}(1)]\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(0)10$$

et

$$t_{n+1}' = \varphi^{n-5}(z)\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(0)1[\underline{0\varphi^{n-5}(1)}]\varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(0)1.$$

En prenant $x = 0\varphi^{n-5}(1)$ et $y = \varphi^{n-5}(1)\varphi^{n-6}(0)\cdots\varphi(0)1$, il vient que yxy est un suffixe de t_{n+1}' à compléter en un carré. En une dérivation par SSC on obtient $t_{n+1}' \overset{SSC}{\longmapsto} t_{n+1}\varphi^{n-5}(1)$.
On en déduit que pour tout $n \geq 6$, $t_n' \overset{SSC}{\longmapsto} t_n\varphi^{n-6}(1)$. ∎

Proposition 2.3.4. *Le mot infini t de Thue-Morse est dans SSC^{ω}.*

Preuve : Soit t_5, le préfixe initial générateur de t par SSC. Par la Proposition 2.3.2 on dérive par SSC le mot $t_6^{'}$ de t_5. Grâce à la Proposition 2.3.3, nous dérivons le mot t_61 de $t_6^{'}$ en une étape de SSC. Une fois le mot t_61 obtenu, nous dérivons $t_7^{'}$ de t_61 par le Lemme 2.3.1. En poursuivant la dérivation avec la même procédure, on finit par dériver en une étape donnée le mot $t_n^{'}$. De $t_n^{'}$ on dérive $t_n\varphi_t^{n-6}(1)$. A l'aide du Lemme 1.3.9 on dérive $t_{n+1}^{'}$. En itérant le processus on obtient la suite de mots $(t_n^{'})_n$, qui converge vers le mot infini t de *Thue-Morse*. ■

Dans le mot de *Thue-morse*, l'observation des carrés à conduit au Lemmes suivants :

Lemme 2.3.5. *Pour tout $n \geq 1$, t_n et $\bar{t}_n$ ne possèdent pas de préfixes carrés.*

Preuve : Pour $n \in \{1, 2\}$, on a $t_1 = 01$ et $t_2 = 0110$. Ces deux mots ne possèdent pas de préfixes carrés. La propriété est alors vérifiée pour $n \in \{1, 2\}$.
Supposons que jusqu'à l'ordre n, t_n ne possède toujours pas de préfixes carrés. Vérifions à l'ordre $n+1$ que t_{n+1} ne possède pas de préfixes carrés. Supposons que t_{n+1} possède un préfixe carré. Soit xx ce préfixe carré. Comme $|xx|$ est paire, nous distinguons alors deux cas de figures :

• Supposons que $|xx|$ est une puissance de 2. i.e, $|xx| = 2^k$ avec $k \in \{1, ..., n+1\}$. Alors, il existe l avec $1 \leq l \leq n+1$ tels que $t_l = t_{l-1}\bar{t}_{l-1} = xx$. Ainsi, il vient que $t_{l-1} = \bar{t}_{l-1}$. Ce qui est absurde car pour tout $l \geq 1$ on a $t_{l-1} \neq \bar{t}_{l-1}$.

• Supposons que $|xx|$ n'est pas une puissance de 2 .i.e, $|xx| = 2^k + 2p$ avec $1 \leq k \leq n$ et $1 \leq p \leq 2^{n-2}$. Alors, $xx = t_kt_p = t_{k-1}\bar{t}_{k-1}\bar{t}_{p-1}t_{p-1}$. Le centre du carré se trouve dans $\bar{t}_{k-1}$. Il existe $w_1 \in Fact(t_{k-1})$ et $w_2 \in Fact(\bar{t}_{k-1})$ avec $|w_1| = |w_2|$ tels que $t_{k-1} = t_p\bar{t}_pw_1$ et $\bar{t}_{k-1} = \bar{t}_pt_pw_2$. On a $xx = (t_p\bar{t}_pw_1)(\bar{t}_pt_pw_2)\bar{t}_{p-1}t_{p-1} = (t_{p-1}\bar{t}_{p-1}\bar{t}_pw_1)(\bar{t}_{p-1}t_{p-1}t_pw_2)\bar{t}_{p-1}t_{p-1}$.
Par suite, $xx = (t_{p-1}\bar{t}_{p-1}\bar{t}_pw_1\bar{t}_{p-1})(t_{p-1}t_{p-1}\bar{t}_{p-1}w_2\bar{t}_{p-1}t_{p-1})$. Ainsi, $t_{p-1} = \bar{t}_{p-1}$. Ce qui est absurde, car $t_{p-1} \neq \bar{t}_{p-1}$ pour tout $p \geq 1$. Par un raisonnement analogue, on montre que $\bar{t}_n$ ne possède pas de préfixes carrés. ■

Lemme 2.3.6. *Pour tout $n \geq 0$, t_n et $\bar{t}_n$ ne possèdent pas de suffixes carrés.*

Lemme 2.3.7. *Pour tout $n \geq 2$, si $t_n[i..j]$ est un facteur carré de t_n tel que $i \leq 2^{n-1}$ et $j > 2^{n-1}$ alors, ce carré est un facteur de $t_n[2^{n-2}+1..2^{n-1}+2^{n-2}] = \bar{t}_{n-2}\bar{t}_{n-2}$. Pour tout $n \geq 2$, si $\bar{t}_n[i..j]$ est un facteur carré de $\bar{t}_n$ tel que $i \leq 2^{n-1}$ et $j > 2^{n-1}$ alors, ce facteur carré est un facteur de $\bar{t}_n[2^{n-2}+1..2^{n-1}+2^{n-2}] = t_{n-2}t_{n-2}$.*

Autrement dit :

• s'il existe un facteur carré de t_n qui contient le centre de t_n, alors ce carré est un facteur de $\bar{t}_{n-2}\bar{t}_{n-2}$.

• s'il existe un facteur carré de t_n qui contient le centre de $\bar{t}_n$, alors ce carré est facteur de $t_{n-2}t_{n-2}$.

Lemme 2.3.8. *Pour tout $n \geq 0$, si $t_n[i..j]$ est un facteur carré tel que $i \leq 2^{n-1}$ et $j > 2^{n-1}$ de plus si $i \neq 2^{n-2}+1$ ou $j \neq 2^{n-1}+2^{n-2}$ alors, ce carré est un facteur de $t_n[2^{n-2}+2^{n-3}+1..2^{n-1}+2^{n-3}]$.*
Pour tout $n \geq 0$, si $\bar{t}_n[i..j]$ est un facteur carré tel que $i \leq 2^{n-1}$ et $j > 2^{n-1}$ de plus si $i \neq 2^{n-2}+1$ ou $j \neq 2^{n-1}+2^{n-2}$ alors, ce carré est un facteur de $\bar{t}_n[2^{n-2}+2^{n-3}+1..2^{n-1}+2^{n-3}]$.

Autrement dit :
• s'il existe un facteur carré de t_n qui contient le centre de t_n tel que ce facteur ne soit pas identique à $\bar{t}_{n-2}\bar{t}_{n-2}$ alors, ce facteur carré est un facteur de $t_{n-3}\bar{t}_{n-3}$.
• s'il existe un facteur carré de $\bar{t}_n$ qui contient le centre de $\bar{t}_n$ tel que ce facteur soit différent de $t_{n-2}t_{n-2}$, alors ce facteur carré est un facteur de $\bar{t}_{n-3}t_{n-3}$.

Proposition 2.3.9. *Pour tout $n \geq 1$, on considère deux mots finis x et y préfixes de t tels que $x \in Pref(t_n)$ et $t_n \in Pref(y)$. Si $y \in SD^*(x)$ alors $|y| \leq 2^{n+1}+2^{n-1}$.*

<u>**Preuve**</u>: Procédons par récurrence sur $n \geq 1$.
Pour $n = 1$, $t_1 = 01$ et $\bar{t}_1 = 10$. Par suite $\{0, 01\} \in Pref(t_1)$ et $\{1, 10\} \in Pref(\bar{t}_1)$. On considère les préfixes de t_1 et $\bar{t}_1$. Par une première duplication de suffixes on a :

$$\left\{\begin{array}{r} 0 \overset{SD}{\longmapsto} 0^2 \notin Pref(t) \\ 01 \overset{SD}{\longmapsto} 011 \in Pref(t) \\ 01 \overset{SD}{\longmapsto} 0101 \notin Pref(t) \\ 1 \overset{SD}{\longmapsto} 1^2 \notin Pref(\bar{t}) \\ 10 \overset{SD}{\longmapsto} 100 \in Pref(\bar{t}) \\ 10 \overset{SD}{\longmapsto} 0110 \notin Pref(\bar{t}). \end{array}\right.$$

Considérons les préfixes de t et $\bar{t}$ obtenus à l'issue de la première duplication de suffixes. Par une seconde duplication de suffixes nous avons :

$$\left\{\begin{array}{l} 011 \overset{SD}{\longmapsto} \{011\underline{1},\ 01\underline{1}\underline{1}1,\ 011\underline{011}\} \notin Pref(t) \\ 100 \overset{SD}{\longmapsto} \{100\underline{0},\ 100\underline{00},\ 100\underline{100}\} \notin Pref(\bar{t}). \end{array}\right.$$

Nous avons soulignés les suffixes dupliqués. Le plus long préfixe de t obtenu par duplication de suffixes à partir d'un préfixe de t_1 est 011 et celui de $\bar{t}$ obtenu à partir d'un préfixe de $\bar{t}_1$ est 100. Ainsi, nous avons $|011| = |100| < 2^{1+1} + 2^0 = 5$. La propriété est donc vérifiée pour $n = 1$.
Supposons que pour tout entier naturel $l \leq n-1$, pour $y \in SD^*(x)$ avec $x \in \{Pref(t_l), Pref(\bar{t}_l)\}$, on a $|y| \leq 2^{l+1} + 2^{l-1}$.
Supposons qu'a partir d'un préfixe initial x de t_n ou $\bar{t}_n$ on arrive à dériver en k

étapes de SD, un mot u tel que $|u| < 2^{n+1} + 2^{n-1}$ et qu'à l'étape $k+1$ de SD l'on obtient le mot uz avec $|uz| > 2^{n+1} + 2^{n-1}$.
Une décomposition t_{n+2} donne :

$$t_{n+2} = t_{n-2}\bar{t}_{n-2}\bar{t}_{n-2}t_{n-2}\bar{t}_{n-2}t_{n-2}t_{n-2}\bar{t}_{n-2}\bar{t}_{n-2}t_{n-2}t_{n-2}\bar{t}_{n-2}t_{n-2}\bar{t}_{n-2}\bar{t}_{n-2}t_{n+2}$$

ou

$$t_{n+2} = \tau_1\tau_2\tau_3\tau_4\tau_5\tau_6\tau_7\tau_8\tau_9\tau_{10}\tau_{11}\tau_{12}\tau_{13}\tau_{14}\tau_{15}\tau_{16}.$$

Pour $i \in \{1, \cdots, 16\}$, les τ_i représentent les facteurs t_{n-2} ou $\bar{t}_{n-2}$ de t_{n+2}.
Comme $|\tau_1..\tau_{10}| = 10 \times 2^{n-2} = 8 \times 2^{n-2} + 2 \times 2^{n-2} = 2^{n+1} + 2^{n-1}$ alors, $u \in Pref(\tau_1..\tau_{10})$ et uz est un préfixe de t_{n+3}. Si $u \in SD^k(x)$ alors, il existe $u_1 \in Pref(u)$ tel que $u \in SD(u_1)$. Ainsi, $u = u_1 z$ avec $z \in Suff(u_1)$.
Déterminons deux préfixes intermédiaires obtenus dans la dérivation de uz à partir du préfixe initial x de t_n. Soit u_1, le préfixe qui se termine dans τ_7 et u le préfixe de uz.
Nous procédons par une réduction d'étapes de duplications en commençant par l'étape $k+1$.
D'abord, il est nécessaire de trouver la position finale de u qui correspond au centre du carré zz. Ainsi, nous analysons les positions suivantes :
• Le préfixe u ne se termine pas dans $\tau_1..\tau_4$ ($u \notin Pref(\tau_1..\tau_4 = t_n)$). En appliquant le Lemme 2.3.7 à t_{n+2}, il vient que le carré qui contient le centre de t_{n+2} commence nécessairement après la position $2^n + 1 = |\tau_1..\tau_4\bar{t}_0|$. Comme uz contient le centre de t_{n+2} qui est dans $\tau_8\tau_9$ alors, le premier z de zz ne peut commencer dans $t_n = \tau_1..\tau_4$. Comme il est possible que uz termine dans t_{n+3}, alors en appliquant le Lemme 2.3.7 t_{n+3}, il vient que le carré centré dans t_{n+3} commence nécessairement après la position $2^{n+1}+1$. Ainsi, le premier z du carré zz ne peut se trouver dans $t_n = \tau_1..\tau_4$ de même que la position finale de u.
• Le préfixe u de uz ne se termine pas dans $\tau_5\tau_6$.
Si le carré zz commence dans τ_5, il serait le seul carré $\tau_5..\tau_{12} = \bar{t}_n\bar{t}_n$.
Dans ce cas $uz = t_{n+1}$. Alors, t_{n+1} possède un suffixe carré, ce qui est absurde par Lemme 2.3.6.
Si le carré zz commence dans τ_6 alors la position finale de uz se trouve dans t_{n+2}. En appliquant le Lemme 2.3.8 à t_{n+2}, il vient que le carré zz est situé dans

$$t_{n+2}[2^n + 2^{n-1} + 1..2^{n+1} + 2^{n-1}] = \tau_7..\tau_{10}.$$

Ce qui est absurde. Puisque par hypothèse uz se termine strictement après la position finale de τ_{10}.
• Selon le Lemme 2.3.8 appliqué à t_{n+2}, il n'existe pas de carré qui commence dans τ_8 et se termine strictement après la position finale de τ_{10}.
• Puisque le facteur carré zz ne commence pas dans $\tau_1..\tau_8$ et par l'hypothèse selon laquelle uz se termine strictement après la position finale de τ_{10}, il vient que le centre

de zz ne peut se trouver que dans la seconde moitié de τ_9. Ainsi, u se termine dans la seconde moitié de τ_9.
En admettant que cette position finale de uz se trouve dans la seconde moitié de τ_9, alors On procède à une seconde réduction d'étape de duplication suffixes quitte à obtenir le préfixe u de uz à l'étape k.
Comme $u \in SD^k(x)$ alors, il existe $u_1 \in Pref(u)$ et $z_1 \in Suff(u_1)$ tels que $u = u_1z_1$. Par un raisonnement similaire à l'étape $k+1$, on montre que u_1 ne se termine pas dans $\tau_1..\tau_6$ mais se termine dans τ_7. Prenons w_1 le suffixe de u_1 contenue dans τ_7 et w_2 le suffixe de u contenu dans $\tau_7..\tau_{10}$. Alors, il existe r_1 tel que : $u_1 = r_1w_1$ et $u = r_1w_2$. Comme $u_1 \in SD^{k_1}(x)$ avec $k_1 < k$, et $u \in SD^k(x)$ alors $u \in SD^{k-k_1}(u_1)$. Par suit $w_1 \overset{SD}{\longmapsto} w_2$. Nous avons $w_1 \in Pref(\bar{t}_{n-2})$ et $w_1 \in Pref(w_2)$. Ainsi, $w_2 \in Pref(\bar{t})$ et comme la position finale de u se situe dans la seconde moitié de τ_9, on a $|\tau_7\tau_8t_{n-3}| = 2^{n-1} + 2^{n-3} < |w_2| < |\tau_7..\tau_{10}|$ ce qui est absurde d'après l'hypothèse de récurrence. ■

Proposition 2.3.10. *Le mot infini t de Thue-morse n'est pas dans PSD^ω*

__Preuve__: Puisque, le mot de *Thue – morse* t est un mot infini à droite alors, il ne peut être généré par duplication de préfixes. De plus, par la Proposition 2.3.9, t ne peut être généré par duplication de suffixes. Ainsi, t n'est pas dérivable par duplications de préfixes-suffixes. ■

2.4 Le mot infini de *Stewart's choral*

Le mot s *de Stewart's choral* est le mot infini obtenu comme limite de la suite $(s_n)_n$ définie par : $s_0 = 0$ et $s_{n+1} = s_ns_ns_n^*$ pour tout $n \geq 0$. Le suffixe s_n^* est obtenu par échange de la lettre du milieu du mot s_n.
Ainsi,

$$s = \lim_{n\to+\infty} s_n.$$

Le mot infini s ne contient pas de cube. De plus $|s_n| = 3^n$.

Proposition 2.4.1. *Pour tout $n \geq 1$, $s_{n+1} \in SD^*(s_n)$ implique $s_{n+2} \in SD^*(s_{n+1})$.*

__Preuve__: D'abord, vérifions pour $n = 1$, $s_2 \in SD^*(s_1)$ implique $s_3 \in SD^*(s_2)$. Pour ce faire, nous choisissons respectivement s_1, $s_0s_0^*$ et s_0^* comme suffixes à dupliquer selon les étapes. On a :

$$\underline{s_1} \overset{SD}{\longmapsto} s_1\underline{s_1} = s_1s_0\underline{s_0s_0^*} \overset{SD}{\longmapsto} s_2s_1s_1s_1^*\underline{s_1s_1^*} = s_1s_1s_0\underline{s_0^*} \overset{SD}{\longmapsto} s_1s_1s_0s_0^*\underline{s_0^*} = s_1s_1s_1^* = s_2.$$

Ainsi, $s_2 \in SD^3(s_1) \subset SD^*(s_1)$.
Montrons maintenant que $s_3 \in SD^*(s_2)$. Comme suffixes à dupliquer, nous prenons respectivement s_2, $s_1s_1^*$, et s_1^*. La dérivation donne :

$$\underline{s_2} \overset{SD}{\longmapsto} s_2\underline{s_2} = s_2s_1\underline{s_1s_1^*} \overset{SD}{\longmapsto} s_2s_1s_1s_1^*\underline{s_1s_1^*} = s_2s_2s_1\underline{s_1^*} \overset{SD}{\longmapsto} s_2s_2s_1s_1^*\underline{s_1^*} = s_2s_2s_2^* = s_3.$$

Ainsi, $s_3 \in SD^3(s_2) \subset SD^*(s_2)$.
Supposons que la propriété reste vraie jusqu'à l'ordre n i.e., $s_{n+1} \in SD^*(s_n)$ implique $s_{n+2} \in SD^*(s_{n+1})$. Alors, vérifions que $s_{n+3} \in SD^*(s_{n+2})$. Nous avons

$$\underline{s_{n+2}} \xmapsto{SD} s_{n+2}\underline{s_{n+2}} = s_{n+2}s_{n+1}\underline{s_{n+1}s^*_{n+1}} \xmapsto{SD} s_{n+2}s_{n+1}s_{n+1}s^*_{n+1}\underline{s_{n+1}s^*_{n+1}} =$$

$$s_{n+2}s_{n+2}s_{n+1}\underline{s^*_{n+1}} \xmapsto{SD} s_{n+2}s_{n+2}s_{n+1}s^*_{n+1}\underline{s^*_{n+1}} = s_{n+2}s_{n+2}s^*_{n+2} = s_{n+3}.$$

Ainsi, $s_{n+3} \in SD^3(s_{n+2}) \subset SD^*(s_{n+2})$. Par conséquent, $s_{n+1} \in SD^*(s_n)$ implique $s_{n+2} \in SD^*(s_{n+1})$. ■

Proposition 2.4.2. *Le mot infini* s *de* Stewart's choral *est dans* SD^ω.

<u>**Preuve**</u>: A l'aide de la Proposition 2.4.1 nous générons par SD une suite finie de mots $(s_n)_n$ qui converge vers le mot infini s de Stewart's choral. ■

Chapitre 3

Complétions carrées de préfixes-suffixes des mots finis et algorithmes

Dans ce chapitre nous nous intéressons aux mots finis. D'une part, nous définirons une procédure qui permet de déterminer les facteurs minimaux générateurs d'un mot par $PSSC$. D'autre part, nous établirons un test, qui prend en compte deux mots finis de tailles différentes puis vérifie la possibilité à générer par $PSSC$ le mot le plus long à partir du plus court.

3.1 Algorithme de recherche

Dans cette partie nous élaborons deux procédures. La première détermine le préfixe minimal permettant de générer w par SSC et la seconde permet de déterminer les facteurs minimaux pouvant générer w par $PSSC$.

Définition 3.1.1. *Soit w un mot fini non vide de longueur n. Pour tout $i \in \{1, \cdots, n\}$:*

• On désigne par $MRE[i]$ l'indicatrice de la position minimale $j > i$ tel qu'il existe un facteur carré de position finale j dont la première moitié contient la position i. Nous avons

$$MRE[i] = min\Big\{j/\exists\, w[l..j]\, un\, carr\acute{e}\ avec\ l \leq j < l + \frac{j-l+1}{2}\Big\}$$

. *• On désigne par $MLE[i]$ l'indicatrice de la position maximale $l < i$ tel qu'il existe un facteur carré de position initiale l dont la seconde moitié contient la position i. Nous avons*

$$MLE[i] = max\Big\{l/\exists\, w[l..j]\, un\, carr\acute{e}\ avec\ l + \frac{j-l+1}{2} \leq i \leq j\Big\}$$

Exemple 3.1.2. *Considérons le mot w et i l'indice de ses lettres représentés dans le tableau ci-dessous.*

w	0	1	0	0	1	0	1
i	1	2	3	4	5	6	7

Les différents facteurs carrés de w *sont :* $w[1..6] = (010)^2$, $w[3..4] = (0)^2$, $w[4..7] = (01)^2$. *Déterminons maintenant* $MRE[i]$ *et* $MLE[i]$ *pour tout* $i \in \{1...7\}$.

- *Valeur de* $MRE[i]$ *pour tout* $i \in \{1...7\}$

On détermine premièrement les positions finale de tous les facteurs carrés qui contiennent la position i *dans la première moitié. Ensuite, on retient la valeur minimale de ces positions finales obtenues. Ainsi, on obtient* $MRE[i]$.

Pour $i = 1$, *on a le seul facteur carré* $w[1..6] = (010)^2 = 010010$. *Alors,* $MRE[1] = min\{6\} = 6$.

Pour $i = 2$, *on a le seul facteur carré* $w[1..6] = 010010 = (010)^2$. *Ainsi,* $MRE[2] = min\{6\} = 6$.

Les autres valeurs se déterminent de façon similaire.

On a : $MRE[3] = min\{4, 6\} = 4$, $MRE[4] = min\{7\} = 7$, $MRE[5] = min\{7\} = 7$, $MRE[6] = MRE[7] = min\{\varnothing\} = +\infty$.

- *Valeur de* $MLE[i]$ *pour tout* $i \in \{1...7\}$.

On détermine la position initiale de tous les facteurs carrés contenant la position i *dans la seconde moitié. Une fois ces facteurs retrouvés, on retient la valeur maximale de ses positions initiales. Ainsi, on obtient* $MLE[i]$.

Pour $i = 1$, *il n'existe pas de facteurs carrés. Alors,* $MLE[1] = Max\{\varnothing\} = -\infty$.

Les autres valeurs se déterminent de façon similaire.

On a : $MLE[2] = Max\{\varnothing\} = -\infty$, $MLE[3] = Max\{\varnothing\} = -\infty$,
$MLE[4] = Max\{3, 1\} = 3$, $MLE[5] = Max\{1\} = 1$,
$MLE[6] = Max\{4, 1\} = 4$, $MLE[7] = Max\{4\} = 4$.

Lemme 3.1.3. *Soit* w *un mot fini non vide sur* $\sum$. *On considère deux facteurs* $w[i_1..j_1]$ *et* $w[i_2..j_2]$ *de* w *tels que* $i_1 \leq i_2 \leq j_2 \leq j_1$. *Si* $w \in SSC^*(w[i_2..j_2])$ *alors* $w \in SSC^*(w[i_1..j_1])$.

Preuve: Supposons que $w \in SSC^*(w[i_2..j_2])$ et $w[i_2..j_2] \in Fact(w[i_1..j_1])$. Alors $w \in SSC^*(w[i_2..j_2])$ implique $w[i_2..j_2] \in Pref(w)$. Comme $w[i_2..j_2] \in Fact(w[i_1..j_1])$ et que les deux mots $w[i_1..j_1]$, $w[i_2..j_2]$ sont des facteurs de w, alors $w[i_1..j_1] \in Pref(w)$. Ainsi, $w[i_2..j_2] \in Pref(w[i_1..j_1])$ et $w[i_1..j_1] \in Pref(w)$.
Il en résulte Lemme 1.3.9 que $w \in SSC^*(w[i_1..j_1])$. ∎

Lemme 3.1.4. *Soit* w *un mot fini non vide de longueur* n. *Les valeurs* $MRE[i]$ *et* $MLE[i]$ *peuvent être déterminées en un temps linéaire* $\mathcal{O}(n)$.

Preuve: Pour déterminer chaque valeur $MRE[i]$ et $MLE[i]$, on a besoin de parcourir toutes les positions de w pour retrouver les facteurs carrés et vérifier les hypothèses. Comme w est un mot fini, alors dans le pire des cas, on aurait au plus n tests à faire. Ainsi, la complexité algorithmique est donc d'ordre linéaire. ∎

compteurs i_{min} et l à la valeur $7 = |w|$. On désigne par i, la position initiale du facteur run considéré et par p la période de ce même facteur. Nous parcourons ensuite, les positions de w dans l'ordre décroissante de la longueur de w comme suit :

- *Pour $l = 7$ on a $i_{min} = 7$ car $w \in SSC^0(w)$.*
- *Pour $l = 6$ on a $i_{min} = 3$ car $L[6] = 1$ et $p + i - 1 = 3 + 1 - 1 = 3 < l$.*
  Le facteur run considéré est $w[1..6] = (010)^2$ qui pour période $p = 3$.
- *Pour $l = 5$ on a $i_{min} = 3$ car $L[5] = 1$ mais $5 > 3$.*
  Pour $l = 4$ on a $i_{min} = 3$ car $L[4] = 1$ et $4 > 3$.
- *Pour $l = 3$ on a $i_{min} = 3$.*
  Pour $l = 2$ on a $i_{min} = 3$ car $L[2] = 0$
  Par complétions carrés de suffixes de $w[1..i_{min}] = w[1..3] = 010$ on a :

$$w[1..3] = 01\underline{0} \overset{SSC}{\longmapsto} 010\underline{0} = 0\underline{100} \overset{SSC}{\longmapsto} 01001\underline{0} = 0100\underline{1}0 \overset{SSC}{\longmapsto} 010010\underline{1} = w$$

  Ainsi, $w \in SSC^3(w[1..3])$. D'où, $i_{min} = i = 3$.

Remarque 3.1.10. *Certain mots finis possèdent des propriétés particulières rendant possible toute complétion carrée de préfixes-suffixes à partir d'un facteur quelconque.*

3.1.2 Recherche des facteurs minimaux pour $PSSC$

Dans cette partie nous construisons une procédure qui permet de déterminer les facteurs minimaux d'un mot fini capable de le générer par $PSSC$.

Définition 3.1.11. *Soit w un mot fini non vide de longueur n. Pour tous entiers naturel i, k tels que $1 \leq i \leq k \leq n$, $RMQ(i,k)$ est la valeur minimale comprise dans $\Big\{MLE[i],\ MLE[i+1], \cdots, MLE[k]\Big\}$ et on note :*

$$RMQ(i,k) = min\Big\{MLE[i],\ MLE[i+1], \cdots\ MLE[k]\Big\}$$

$posRMQ(i,k)$ est l'indicatrice de la position où $RMQ(i,k)$ est atteinte. On note

$$posRMQ(i,k) = \Big\{j / RMQ(i,k) = MLE[j]\Big\}$$

Exemple 3.1.12. *On considère le mot $w = 0100101$.*
$RMQ(4,7) = min\{MLE[4],\ MLE[5],\ MLE[6],\ MLE[7]\} = min\{3,\ 1,\ 4,\ 4\} = 1$ et $posRMQ(1,5) = 5$.

Théorème 3.1.13. *Soit w un mot fini non vide de longueur n. Pour tout $i \leq n$, il existe un facteur minimal $w[i..j_i]$ tel que $w \in PSSC^*(w[i..j_i])$.*

<u>**Preuve**</u> : Nous déterminons les valeurs minimales j_i pour chaque position i considérée en procédons à une série d'analyses.

D'abord, il convient de déterminer pour chaque position i, la valeur $MRE[i]$. L'analyse de cette valeur permettra d'identifier la complétion carrée nécessaire à appliquer en premier. Aux cas où une complétion carrée de préfixe n'est pas possible, on détermine alors les valeurs $MLE[i]$ et $RMQ(i,k)$ pour tout $k \geq i$. L'analyse de ces valeurs permet d'envisager d'éventuelles complétions carrées de suffixes suivit de complétions carrées de préfixes. Les valeurs j_i à déterminer vérifient la relation $j_1 \leq j_2 \leq \cdots \leq j_n$, d'après le Lemme 3.1.3.

Suivant les valeurs de i, nous déterminons les valeurs des j_i en procédant comme suit :

• Pour $i = 1$, il suffit d'utiliser l'algorithme 1 du Théorème 3.1.8 qui permet de trouver la valeur j_1 qui correspond à la valeur minimale telle que $w \in SSC^*(w[1..j_1])$.

• Pour $i = 2$ on procède comme suit :

- Si $MRE[1] \leq j_1$, il suffit de poser $j_2 = j_1$.
  En effet, par la Proposition 3.1.5, $j_1 \geq MRE[1]$ implique $w[1..j_1] \in PSC^*(w[2..j_1])$. Il suffit de procéder par une complétion carrée de préfixe pour retrouver le préfixe initial $w[1..j_1]$. Dans ce cas, on utilise l'algorithme 1 pour générer w par SSC à partir de $w[1..j_1]$. Ainsi, pour $j_2 = j_1$ on a $w \in PSSC^*(w[2..j_1])$.
- Si $MRE[i] > j_1$ alors, d'après la Proposition 3.1.5, il n'est pas possible d'appliquer une complétion carrée de préfixe au facteur considéré. On envisage dans ce cas, la possibilité d'appliquer une éventuelle complétion carrée de suffixes. On pourrait premièrement générer par complétion carrée de suffixes, un facteur de position finale m contenant la position $MRE[1]$
  (i.e, $w[2..MRE[1]] \in Pref(w[2..m])$). A partir de ce facteur, on pourrait générer le préfixe $w[1..m]$ de w par complétions carrées de préfixes en utilisant la Proposition 3.1.4. Ainsi, il convient donc de déterminer la valeur $RMQ(j_1, MRE[1])$ afin d'envisager cette possibilité.
  - Si $RMQ(j_1, MRE[1]) \geq 2$ alors, on pose dans ce cas $j_2 = j_1$.
    En effet, si $RMQ(j_1, MRE[1]) \geq 2$ alors, il existe un facteur carré qui commence au moins à la position 2 et qui contient toute position j telle que $j_1 \leq j \leq MRE[1]$. D'après la Proposition 3.1.10, le facteur $w[2..j_1]$ peut générer par SSC un facteur $w[2..p]$ tel que $p \geq MRE[1]$. Ainsi, on obtient un facteur qui contient $MRE[1]$. On procède maintenant par complétions carrées de suffixes pour générer le mot $w[1..p]$ à l'aide de la Proposition 3.1.5. Une fois le mot $w[1..p]$ obtenu, on génère tout le mot w par complétions carrées de suffixes à l'aide de la Proposition 3.1.10.
  - Si $RMQ(i,k) < 2$ alors, on pose $j_2 = posRMQ(j_1, MRE[1])$.
    En effet, d'une part, on ne peut dériver un facteur qui contient la position 1 à partir d'un autre facteur qui ne contient pas la position $MRE[1]$. D'autre part, on ne peut pas générer un facteur qui contient la position $MRE[1]$ à partir d'un facteur qui commence à la position 2. La seule possibilité

est de générer des facteurs contenant toutes les positions j telles que $j_1 \leq j \leq MRE[1]$ et $MRE[j] = 1$. Ainsi, aucune complétion carrée n'est alors possible.

Supposons que les valeurs $j_1, j_2, \cdots, j_{n-1}$sont connues et que nous voulons déterminer j_n. Alors, nous procédons de la manière suivante :
Par un raisonnement similaire à l'étape $i = 2$ on a :
Si $j_{i-1} \geq MRE[i-1]$ alors, $j_i = j_{i-1}$.
Si ce n'est pas le cas ($j_{i-1} < MRE[i-1]$), alors, on calcule la valeur $RMQ(j_{i-1}, MRE[i-1])$. Par cette valeur, on vérifie :
Si $RMQ(j_{i-1}, MRE[i-1]) \geq i$, alors on pose $j_i = j_{i-1}$. Dans le cas contraire (i.e, $RMQ(j_{i-1}, MRE[i-1]) < i$) on procède comme suit :
Cette partie diffère de celle de l'étape 2.
Nous posons $l = posRMQ(j_{i-1}, MRE[i-1])$. Par cette valeur, on calcule $l = posRMQ(l, MRE[i-1])$. Tant que $RMQ(l, MRE[i-1]) < i$, nous gardons la valeur de $l = posRMQ(l, MRE[i-1])$.
A la fin du cycle i.e lorsque $RMQ(l, MRE[i-1]) \geq i$, on pose $j_i = l$. Ainsi, la valeur $l = posRMQ(l, MRE[i-1])$ donne la position finale du facteur qui commence à la position n.
On en déduit un algorithme permettant de déterminer pour tout $i \leq n$, les valeurs j_i telles que $w \in PSSC^*(w[i..j_i])$. Voir l'algorithme 2. ∎

Exemple 3.1.14. *Considérons le mot $w = 0100101$ représenté dans le tableau ci-dessous :*

w	*0*	*1*	*0*	*0*	*1*	*0*	*1*
i	*1*	*2*	*3*	*4*	*5*	*6*	*7*

Déterminons les valeurs des j_i pour $i \in \{1, .., 7\}$.
Pour $i = 1$, en utilisant l'algorithme 1 du Théorème 3.1.8. On a $j_1 = 3$.
Pour $i = 2$, puisque $MRE[1] = 6 > j_1 = 3$, on a $RMQ(j_1, MRE[1]) = RMQ(3, 6) = 1 < 2$. Ainsi, on pose $j_2 = posRMQ(3, 6) = 5$.
Pour $i = 3$, puisque $MRE[2] = 6 > j_2 = 5$, on a $RMQ(j_2, MRE[2]) = RMQ(5, 6) = 1 < 2$. Ainsi, $j_3 = posRMQ(5, 6) = 5$.
En procédant de façons similaire tout en suivant les étapes de l'algorithme, on retrouve les valeurs j_i consignées dans le tableau suivant :

i	*1*	*2*	*3*	*4*	*5*	*6*	*7*
j_i	*3*	*5*	*5*	*5*	*7*	*7*	.

Algorithme 2 : facteurs générateurs par $PSSC$

Entrée : $w[1..n]$
Sortie : j_i
Début :

Pour i allant de 1 à n **faire**
- **si** $i = 1$ **faire**
  - Utiliser l'algorithme 1
- **fin de si**
- **si** $i = 2$ **faire**
  - calculer $MRE[1]$
  - **si** $(j_1 \geq MRE[1])$ **faire**
    - $j_2 \leftarrow j_1$
  - **sinon**
  - Calculer $RMQ(j_1, MRE[1])$
    - **si** $RMQ(j_1, MRE[1]) \geq 2$ **faire**
    - $j_2 \leftarrow j_1$
    - **sinon**
    - $j_2 \leftarrow posRMQ(j_1, MRE[1])$
    - **finsi**
  - **finsi**
- **finsi**
- **si** $i > 2$ **faire**
  - **si** $(j_{i-1} \geq MRE[i-1])$ **faire**
    - $j_i \leftarrow j_{i-1}$
  - **sinon**
    - **si** $(RMQ(j_{i-1}, MRE[i]) \geq i)$
    - $j_i \leftarrow j_{i-1}$
    - **sinon**
    - $l \leftarrow posRMQ(j_{i-1}, MRE[i-1])$
    - **tant que** $RMQ(l, MRE[i]) < i$ **faire**
    - $l \leftarrow posRMQ(l, MRE[i])$
    - **fintantque**
    - retourne l
    - $j_i \leftarrow l$
    - **finsi**
  - **finsi**
- **finsi**
- **finsi**

Fin

Pour calculer la complexité de cet algorithme, on utilise toujours la technique incrémenter compter. Soit $C(n)$ et $C(n+1)$ le nombre d'instructions à l'étape n et à l'étape $n+1$. On a $C(n+1) = C(n)+k$ où k est une constante entière. La complexité est alors d'ordre linéaire $\mathcal{O}(n)$.

3.2 Algorithme de test

Dans cette partie, nous établirons un algorithme test qui prend en compte deux mots de tailles différentes puis teste la possibilité à générer le mot le plus long à partir du plus court par $PSSC$.

Théorème 3.2.1. *Soient w et x deux mots finis non vides tels que $|x| < |w| = n$. On peut tester en un temps linéaire si $w \in PSSC^*(x)$.*

Preuve: Déterminons toutes les occurrences de x dans w s'il y en a. Soit m la longueur de x et n la longueur de w telles que $n > m$. Nous définissons un algorithme qui permet de retrouver les occurrences d'un facteur dans un mot.

Algorithme 3 : occurrences de x dans w

Entrée : $w[1..n]$, $x[1..m]$
Sortie : $T[1..n-m+1]$
Début :

- $i \longleftarrow 1$
- $T[1] \longleftarrow 0$
- **Tant que** $i \leq (n-m+1)$ faire
  - **Si** $w[i..m+i-1] = x[1...m]$ alors
    - $T[i] \longleftarrow 1$
    - $i \longleftarrow i+1$
  - **Sinon**
    - $T[i] \longleftarrow 0$
    - $i \longleftarrow i+1$
  - **fin de si**
- **fin tant que**

  Retourner $T[i]$

Fin.

Cet algorithme prend en compte deux mots finis $x[1..m]$ et $w[1..n]$ de tailles différentes tels que $n > m$. A la fin, l'algorithme retourne un tableau d'entier $T[i] \in \{0, 1\}$.

Si $T[i] = 1$ alors, il existe une occurrence de x dans w et i représente cette occurrence. En revanche, si $T[i] = 0$ alors, l'occurrence de x n'est pas égale à i.

En générale,

- Si pour tout $i \in \{1, .., n-m+1\}$, $T[i] = 0$ alors, il n'existe pas d'occurrences de

x dans w. Dans ce cas, x n'est pas un facteur de w, par conséquent x ne peut être un facteur générateur de w par $PSSC$.

• S'il existe $i \in \{1,..,n-m+1\}$ tel que $T[i] = 1$ alors, x est un facteur de w. Pour savoir s'il génère w, il suffit d'utiliser l'algorithme 2 afin de retrouver tous les facteurs minimaux générateurs de w par $PSSC$. Une fois ces facteurs générateurs trouvés, on les comparent avec x. Ainsi, s'il y' à équivalence, on en déduit que w peut être généré à partir de x ou que x est un générateur de w par $PSSC$.

La complexité de cet algorithme est d'ordre linéaire. En utilisant toujours la méthode incrémenter compter, On a $C(n+1) = C(n)$. La complexité est constante, donc d'ordre linéaire $\mathcal{O}(n)$.

Construisons maintenant un algorithme qui permet de vérifier pour deux mots finis de tailles différentes, si le plus long peut être généré du plus court par $PSSC$.

Algorithme 4 : test de dérivation par $PSSC$

Entrée : $w[1..n]$, $x[1..m]$

Sortie : k

Début :

- $k \leftarrow 0$
- **Pour** i allant de 1 à $(n-m+1)$ **faire**
  - **si** $T[i] = 1$ et $w[i..j_i] = x[1..m]$ **alors**
    - $k \leftarrow k+1$

    finsi

  finpour
- retourner k

Fin

Cet algorithme retourne un entier naturel k. Si $k = 0$, alors $w \notin PSSC^*(x)$.

En revanche, si $k > 0$ alors, $w \in PSSC^*(x)$.

La complexité de cet algorithme est d'ordre linéaire. En considérant le pire des cas possible et en appliquant la technique incrémenter compter on $C(n+1) = C(n) + k$ où k est une constante. Ainsi, la complexité est d'ordre linéaire. ■

CONCLUSION

Ce travail a mis en évidence les opérations de complétions carrées sur des mots finis et infinis. Il a été établi que les mots de *Fibonacci*, de *Thue-morse* et de *Period-doubling* se génèrent par complétions carrées de suffixes tandis que le mot de *Stewart's choral* se génère par duplication de suffixes. En s'intéressant aux mots finis, on obtient un algorithme de complexité d'ordre linéaire qui permet d'identifier certains mots finis qui se génèrent par complétions carrées de préfixes-suffixes à partir d'un facteur donné. Malgré ces résultats obtenus, une caractérisation propre des mots finis générés à partir d'un facteur en utilisant les opérations de complétions carrées reste indéterminée ainsi que la possibilité de générer des mots infinis à partir d'un facteur.

Bibliographie

[1] Allouche, J.,Shallit, J.O. : *Automatic Sequences-Theory, Applications, Generalizations.* Cambridge University press, Cambridge, 2003.

[2] Bannai, H., I, T., Inenaga, S., Nakashima, Y., Takeda, M., Tsuruta, K. :*A new characterization of maximal repetitions by Lyndon trees.* In : *Proceedings of SODA*, pp. 562-571, 2015.

[3] Bender, M.A., Farach-Colton, M. : *The LCA problem revisited.* . In : Gonnet, G.H., Viola, A. (eds.) LATIN 2000. LNCS, vol. 1776, pp. 88-94. Springer, Heidelberg, 2000.

[4] Currie, J.D., Rampersad, N., Saari, K., Zamboni, L.Q. : *Extremal words in morphic Subshifts.* Discret Math. **322**, 53-60, 2014.

[5] Damanik, D. : *Local symmetries in the period-doubling sequence.* Discrete Appl. Math. **100**(1-2), 115-121, 2000.

[6] Dumitran, M., Gil, J., Manea, F., Mitrana, V : *Bounded prefix-suffix duplication.* In :*Holzer*, M.(eds) CIAA 2014. LNCS, vol. 8587, pp. 176-187. Springer, Heidelberg, 2014.

[7] Dumitran M., Manea, F.(2015) *Prefix-Suffix Square Completion* . In : Manea F., Nowokta D.(eds) Combinatorics on words. WORDS 2015. LNCS, vol 9304. pp. 147-159, 2015.

[8] Endrullis, J., Hendriks, D., Klop, J.W. : *Degrees of streams.* Integers Electron. J. Comb. Number Theor. **11B**(A6), 1-40, 2011.

[9] Garcia-Lopez, J. Menea, F., Mitrana, V. : *Prefix-suffix duplication.* J. Comput. Syst. Sci. **80**(7), 1254-1265, 2014.

[10] Gusfield, D. :*Algorithms on Strings, Trees,* and Sequences. : *Computer Science and Computational Biology.* Cambridge University Press, New york , 1997.

[11] Hall, M. : *Generators and Relation in Groups- The Burnside Problem.* Lectures on Modern Mathematics, Vol. 2. Wiley, New York, 1964. 42-92.

[12] Kärkkäinen, J., Sander, P., BurKhardt, S. : *Linear work suffix array construction.* J. ACM 53, 918-936, 2006.

[13] Knuth Jr., D.E., Morris, J.H, Pratt, V.R. : *Fast pattern matching in strings.* SIAM J. Comput. **6**(2), 323-350, 1977.

[14] Kolpakov, R., Kucherov, G. : *Finding maximal repetitions in a word in linear time.* In :*Proceedings of FOCS*, PP. 596-604, 1999.

[15] Lothaire, M. : *Comb.* words. Cambridge University Press, Cambridge, 1997.

[16] Thue, A. : *Über unendliche Zeichenreihen.* Norske Vid. Skrifter I. Mat-Nat. Kl., Christiania **7**, 1-22, 1906.

[17] Thue, A. : *Über die gegenseitige Lage gleicher Teile gewisser Zeichenreihen.* Norske Vid. Skrifter I. Mat.-Nat. Kl., Christiania **1** , 1-67, 1912.