INFORMATIQUE THÉORIQUE ET APPLICATIONS

A. EHRENFEUCHT
H. J. HOOGEBOOM
G. ROZENBERG

Coordinated pair systems ; part I : Dyck works and classical pumping

Informatique théorique et applications, tome 20, nº 4 (1986), p. 405-424
<http://www.numdam.org/item?id=ITA_1986__20_4_405_0>

© AFCET, 1986, tous droits réservés.

L'accès aux archives de la revue « Informatique théorique et applications » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/conditions). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM
Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
http://www.numdam.org/

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 20, n° 4, 1986, p. 405 à 424)

COORDINATED PAIR SYSTEMS; PART I: DYCK WORKS AND CLASSICAL PUMPING (*)

by A. Ehrenfeucht [1], H. J. Hoogeboom [2] and G. Rozenberg [1, 2]

Communicated by J. Berstel

Abstract. – The notion of a *coordinated pair system*, *cp system* for short, is a special instance of an *ects system* which in turn provides a common framework for quite a number of grammar and machine models encountered in the literature. In particular, the notion of a cp system corresponds very closely to (is another formulation of) the notion of a push-down automaton.

In this paper we continue the investigation of cp systems and in particular we investigate the possibility of obtaining pumping properties of context-free languages via the analysis of computations in cp systems. In order to do this we analyze the combinatorial structure of Dyck words. The properties of Dyck words we investigate stem from the combinatorial analysis of computations in cp systems. We demonstrate how this correspondence can be used for proving the classical pumping lemma.

Résumé. – La notion de *système de paires coordonné*, abrégé en *cp système*, est un cas particulier de *ects système* qui, lui, fournit un cadre commun à un grand nombre de modèles de grammaires et de machines rencontrés dans la littérature. En particulier, la notion de cp système est très proche (est une autre formulation) de celle d'automate à pile.

Dans cet article, nous continuons l'étude des cp systèmes, et nous étudions en particulier la possibilité d'obtenir des propriétés d'itération de langages algébriques à travers l'analyse des calculs dans les cp systèmes. Pour ce faire, nous analysons la structure combinatoire des mots de Dyck. Les propriétés des mots de Dyck que nous étudions proviennent d'une analyse combinatoire des calculs dans les cp systèmes. Nous montrons comment cette correspondance peut seqskèù démontrer le lemme d'itération classique.

INTRODUCTION

The notion of an *ects system* provides a common framework for quite a number of grammar and machine models considered in the literature (*see* [R]). A considerably simplified model of an ects system consists of an n-tuple

(*) Received May 1985, revised February 1986.
[1] University of Colorado, Department of Computer Science, Boulder, CO, 80309, U.S.A.
[2] Institute of Applied Mathematics and Computer Science, University of Leiden, Leiden, The Netherlands.

of grammars which are working in a *coordinated fashion*— a direct rewriting step in any of the grammars is coordinated with certain rewriting steps in some of the other grammars.

In particular in the case of two grammars the first of which is right-linear and the second is right-boundary (a right-boundary grammar is like a right-linear grammar except that one does not distinguish between terminal and non-terminal symbols—still the rewriting is applied to the last symbol of a string only) we speak of a *coordinated pair system*, *cp system* for short. It turns out that cp systems correspond very closely to (are another formulation of) push-down automata.

The systematic research exploiting the cp system approach to the theory of push-down automata was initiated in [EHR2]. There, the basic formalism to deal with cp systems was settled as well as the basic technical tool—the *Exchange Theorem*—was proved. The proofs presented in [EHR2] indicate very clearly that in order to understand the structure of computations in cp systems one has to study the combinatorial structure of *Dyck words*.

The idea of a correspondence between the structure of Dyck words and computations in push-down-like models is one of the oldest ideas in formal language theory, however the correspondence we obtain for cp systems is very well suited for the analysis of computations in them.

The present paper takes up this idea. We formulate and prove a number of combinatorial properties of Dyck words (Section 3). Since Dyck words are used in the investigation of various types of data structures, these results seem to be of independent (combinatorial) interest.

Then in Section 4 we demonstrate the correspondence between these properties and the computations in cp systems (the correspondence is established using the Exchange Theorem). Using this we prove the classical pumping property for context-free languages (*see*, e. g., [B], [H] or [S]).

1. PRELIMINARIES

We assume the reader to be familiar with basic formal language theory, in particular with the theory of push-down automata and context-free grammars (*see*, e. g. [H] and [S]). Although the definitions of notions concerning cp systems that are used in this paper are provided in Section 2, it may be instructive for the reader to consult [EHR2] for some additional background and examples.

We fix now some more specific notation and terminology used in this paper.

$\mathbf{N}$ denotes the set of all nonnegative integers and $\mathbf{N}^+$ denotes the set of all positive integers.

For a set W, $\# W$ denotes its cardinality; for sets U, W, $U \subseteqq W$ denotes the inclusion of U in W and $U \subset W$ denotes the strict inclusion of U in W.

Whenever a subset $\mathbf{N}$ is specified in the form $\{i_1, \ldots, i_n\}$, it is assumed that $i_1 < \ldots < i_n$; to remind the reader of this convention we often use the notation $(i_1, \ldots, i_n)$ to denote $\{i_1, \ldots, i_n\}$. Given U, $W \subseteqq \mathbf{N}$ we write $U < W$ if all elements of U are smaller than all elements of W.

We assume that all the alphabets considered in this paper are finite and nonempty. Given an alphabet Σ, $\hat{\Sigma}$ will always mean the alphabet $\{\hat{a} \mid a \in \Sigma\}$ and it is assumed that $\Sigma \cap \hat{\Sigma} = \varnothing$.

For a word x, $|x|$ denotes its length and if $1 \leqq k \leqq |x|$, then $x(k)$ denotes the k-th letter of x. Λ denotes the empty word.

In this paper it is necessary to distinguish carefully between different occurrences of letters (or subwords) in a given word. Therefore we introduce the following notions that lead to a clear distinction between an object (letter, subword) and its occurrence in a word.

DEFINITION 1.1: Let w be a word.

(1) An element i of $\{1, \ldots, |w|\}$ is called an *occurrence in* w. We say that i is an *occurence of* (*the letter*) $w(i)$ *in* w.

(2.1) A subset U of $\{1, \ldots, |w|\}$ is called a *support* (*in* w); the set $\{1, \ldots, |w|\}$ is called the *full support* (*in* w) and is denoted by $fs(w)$. If $U = \{i, i+1, \ldots, j\}$ for some occurrences i and j in w, then U is called a segment (*in* w).

(2.2) Let $U = (i_1, i_2, \ldots, i_n)$ be a nonempty support in w. Then $w(i_1) w(i_2) \ldots w(i_n)$ is called a *sparse subword of* w. If U is a segment in w, then we say that $w(i_1) w(i_2) \ldots (w_n)$ is a *subword of* w. U is referred to as an *occurrence of* $w(i_1) w(i_2) \ldots w(i_n)$; on the other hand, $w(i_1) w(i_2) \ldots w(i_n)$ is referred to as the *image of* U (*in* w) and it is denoted by $w(U)$.

Additionally, we define $w(\varnothing) = \Lambda$ and we say that the empty set is an occurrence of the empty word in w. ■

D_Σ denotes the *Dyck language* (*see* e.g., [S]) over the alphabet $\Sigma \cup \hat{\Sigma}$, where, for each letter a in Σ, its matching "right parenthesis" is $\hat{a}$ in $\hat{\Sigma}$. Formally, D_Σ is the minimal language L over $\Sigma \cup \hat{\Sigma}$ that satisfies:

(i) $\Lambda \in L$;

(ii) if $w \in L$, then $a w \hat{a} \in L$ for every $a \in \Sigma$, and

(iii) if $w_1, w_2 \in L$, then $w_1 w_2 \in L$.

Elements of D_Σ are called *balanced* words.

In this context symbols from Σ and $\hat{\Sigma}$ are referred to as *left* and *right letters* (or *parentheses*) respectively.

A letter to letter homomorphism is called a *coding* and a homomorphism that maps each letter either into a letter or into the empty word is called a *weak coding*.

Two languages are considered *equal* if they differ at most by the empty word, two language generating devices are said to be *equivalent* if their languages are equal.

A *context-free grammar*, abbreviated *cf grammar*, is specified in the form $G=(\Sigma, P, S, \Delta)$ where Σ is its alphabet, P its set of productions, $S \in \Sigma \setminus \Delta$ its axiom and $\Delta \subset \Sigma$ its terminal alphabet. For $x, y \in \Sigma^*$ and $\pi \in P$ we write $x \underset{G}{\overset{\pi}{\Rightarrow}} y$ if x directly derives y using π. We use $\mathbf{L}(CF)$ to denote the class of context-free languages.

A *right-linear grammar*, abbreviated *rl grammar*, is a context-free grammar $G=(\Sigma, P, S, \Delta)$, which has its productions in the set $(\Sigma \setminus \Delta) \times \Delta^* ((\Sigma \setminus \Delta) \cup \{\Lambda\})$.

A *right-boundary grammar*, abbreviated *rb grammar*, differs from the right-linear grammar essentially in the fact it does not distinguish between terminal and nonterminal symbols. A rb grammar is specified in the form of a 3-tuple $G=(\Sigma, P, S)$, where Σ is its alphabet, $P \subseteq \Sigma \times \Sigma^*$ is its set of productions and $S \in \Sigma$ its axiom. As in the case of a rl grammar productions are applied to the last occurrence in a word only. Thus, for $x, y \in \Sigma^*$ and $\pi = A \to w \in P$, x directly derives y (in G using π), written $x \underset{G}{\overset{\pi}{\Rightarrow}} y$, if $x = zA$ and $y = zw$ for some $z \in \Sigma^*$.

2. CP SYSTEMS; BASIC NOTIONS AND RESULTS

In this section we recall a number of notions and results presented in [EHR2]. We start by defining a *coordinated pair system*.

DEFINITION 2.1: A *coordinated pair system*, abbriated *cp system*, is a triple $G=(G_1, G_2, R)$ such that:

(1) $G_1 = (\Sigma_1, P_1, S_1, \Delta)$ is a rl-grammar,

(2) $G_2 = (\Sigma_2, P_2, S_2)$ is a rb-grammar, and

(3) $R \subseteq P_1 \times P_2$. ■

G_1 and G_2 are referred to as the *first* and *second component* of G respectively. Elements of R are called *rewrites* of G.

DEFINITION 2.2: Let $G=(G_1, G_2, R)$ be a cp system, where $G_1=(\Sigma_1, P_1, S_1, \Delta)$ and $G_2=(\Sigma_2, P_2, S_2)$.

(1) Let $x=(x_1, x_2)$, $y=(y_1, y_2) \in \Sigma_1^* \times \Sigma_2^*$. x *directly computes* y (*in* G), denoted $x \underset{G}{\Rightarrow} y$, if there exists a rewrite $\pi=(\pi_1, \pi_2) \in R$ such that $x_1 \overset{\pi_1}{\underset{G_1}{\Rightarrow}} y_1$ and $x_2 \overset{\pi_2}{\underset{G_2}{\Rightarrow}} y_2$; we write then $x \overset{\pi}{\underset{G}{\Rightarrow}} y$ and we say that x directly computes y (in G) *using* π.

$\overset{*}{\underset{G}{\Rightarrow}}$ denotes the reflexive and transitive closure of $\underset{G}{\Rightarrow}$. If $x \overset{*}{\underset{G}{\Rightarrow}} y$, then we say that x *computes* y (*in* G).

(2) A *computation* (*in* G) is a sequence $\rho=\rho(0), \ldots, \rho(n)$ of elements from $\Sigma_1^* \times \Sigma_2^*$ such that $n \geqq 0$, $\rho(0)=(S_1, S_2)$ and, for $1 \leqq i \leqq n$, $\rho(i-1) \underset{G}{\Rightarrow} \rho(i)$.

We say that ρ is *successful* if $\rho(n)=(u, \Lambda)$ for some $u \in \Delta^*$; then the *result of* ρ, denoted by *res* (ρ), is defined by *res* $(\rho)=u$.

(3) Let $\rho=\rho(0), \ldots, \rho(n)$, $n \geqq 1$, be a computation in G. The sequence $\pi_1, \pi_2, \ldots, \pi_n$ of rewrites from R such that, for $1 \leqq i \leqq n$, $\rho(i-1) \overset{\pi_i}{\underset{G}{\Rightarrow}} \rho(i)$ is called the *control sequence of* ρ and it is denoted by *cont* (ρ).

If $\rho=\rho(0)$, then we define cont (ρ) to be the empty sequence.

(4) The *language of* G, denoted $L(G)$, is defined by

$$L(G)=\{res(\rho) \mid \rho \text{ is a successful computation in } G\};$$

it is also referred to as a *coordinated pair language* or cp language for short. ■

If G is a cp system, then we say that G *computes* the language $L(G)$. The class of all cp languages is denoted by $\mathbf{L}(CP)$.

We realize that we somewhat abuse the notation by writing sequences in the form $\rho=\rho(0), \ldots, \rho(n)$ rather than $\rho=\langle \rho(0), \ldots, \rho(n) \rangle$ (this leads to somewhat ambiguous expressions like, e.g., $\rho=\rho(0)$). However, using brackets to delimit sequences would lead to an additional burden on the already involved notation. We hope that abuses of notation of this type will not lead to misunderstandings.

It is easily seen that the notions push-down automaton and cp system are closely related (*see* [R]); the first component of a cp system acts as input, the second component as a push-down store. Hence we have the following result.

THEOREM 2.1: $\mathbf{L}(CP)=\mathbf{L}(CF)$. ■

We will now recall some notions used to describe computations in cp systems. Our first definition is that of a trail of a rewrite—it represents the detailed record of the way the rewrite is used after it has been split into "elementary actions". Once the notion of a trail is defined for rewrites, it carries over to computations through their control sequences. We use the symbol $[S_1;\ S_2]$ to indicate the beginning of this record.

DEFINITION 2.3: Let $G=(G_1,\ G_2,\ R)$ be a cp system, where $G_1=(\Sigma_1,\ P_1,\ S_1,\ \Delta)$ and $G_2=(\Sigma_2,\ P_2,\ S_2)$. Then let

$$\Gamma(G)=\{[S_1;\ S_2]\}\cup\{[\pi,\ i]\,|\,\pi=(\pi_1,\ A\rightarrow w)\in R,\ i\in\mathbf{N}\text{ and } i\leqq|w|\}.$$

(1) Let $\pi=(\pi_1,\ A\rightarrow w)\in R$.

The *trail of* π, denoted by $trl(\pi)$, is the word over $\Gamma(G)$ defined by $trl(\pi)=[\pi,\ 0][\pi,\ 1]\ldots[\pi,\ |w|]$.

(2) Let ρ be a computation in G with control sequence $\pi=\pi_1,\ \ldots,\ \pi_n$, for some $n\geqq 0$, $\pi_1,\ \ldots,\ \pi_n\in R$.

The *trail of* ρ, denoted $trl(\rho)$, is defined by $trl(\rho)=[S_1;\ S_2]\ trl(\pi_1)\ldots trl(\pi_n)$. ■

Given a trail τ of a computation ρ, its contribution, $ctb(\tau)$, gives the word that is generated on the first component during this computation. On the other hand, the weak description of τ, *wdes* (τ), yields the word that describes the sequence of actions taken during the computation ρ on the second component.

DEFINITION 2.4: Let $G=(G_1,\ G_2,\ R)$ be a cp system, where $G_1=(\Sigma_1,\ P_1,\ S_1,\ \Delta)$ and $G_2=(\Sigma_2,\ P_2,\ S_2)$.

(1) *ctb* is the homomorphism from $\Gamma(G)^*$ into Δ^* defined as follows.

For $\tau\in\Gamma(G)$, $ctb(\tau)$ equals

u, *if* $\tau=[\pi,\ 0]$ for some $\pi\in R$ where either $\pi=(X\rightarrow u\,Y,\ \pi_2)$ or $\pi=(X\rightarrow u,\ \pi_2)$, for some $X,\ Y\in\Sigma_1\setminus\Delta,\ u\in\Delta^*$,

Λ, otherwise.

For a word $\alpha \in \Gamma(G)^*$, $ctb(\alpha)$ is referred to as the *contribution of* α.

(2) *wdes* is the coding from $\Gamma(G)^*$ into $(\Sigma_2 \cup \hat{\Sigma}_2)^*$ defined as follows.

For $\tau \in \Gamma(G)$, $wdes(\tau)$ equals

$$wdes(\tau)=\begin{cases} S_2, & \text{if } \tau=[S_1;\ S_2], \\ \hat{A}, & \text{if } \tau=[\pi,0] \text{ where } \pi=(\pi_1, A \to w) \in R, \\ w(k), & \text{if } \tau=[\pi,k] \text{ where } \pi=(\pi_1, A \to w) \text{ and } 1 \leq k \leq |w|. \end{cases}$$

For a word $\alpha \in \Gamma(G)^*$, $wdes(\alpha)$ is referred to as the *weak description of* α. ■

If ρ is a computation in G, then $wdes(trl(\rho))$ is called the *weak description of* ρ. Note that if ρ is successful, then $ctb(trl(\rho))=res(\rho)$.

The following result concerning weak descriptions of successful computations is closely related to the fact that the second component acts like a pushdown store: the last symbol introduced is the first symbol to be rewritten.

LEMMA 2.1: *Let ρ be a successful computation in a cp system G. Then $wdes(trl(\rho)) \in D_{\Sigma_2}$, where Σ_2 is the alphabet of the second component of G.* ■

A basic property for cp systems is the *real-time* property. A real-time cp system is a cp system which generates exactly one terminal symbol on the first component in every computation step.

DEFINITION 2.5: Let $G=(G_1, G_2, R)$ be a cp system, where $G_1=(\Sigma_1, P_1, S_1, \Delta)$ and $G_2=(\Sigma_2, P_2, S_2)$. We say that G is *real-time* if every rewrite $\pi \in R$ is of the form $\pi=(X \to a\,Y, \pi_2)$, where $X \in \Sigma_1 \setminus \Delta$, $Y \in (\Sigma_1 \setminus \Delta) \cup \{\Lambda\}$, $a \in \Delta$ and $\pi_2 \in P_2$. ■

The following result establishes a normal form for cp systems. It is closely related to the Greibach normal form for cf grammars and it can be obtained by translating this grammatical normal form into the terminology of cp systems. In [EHR1] this result was proved directly within the theory of cp systems.

THEOREM 2.3: *For every cp system H there exists an equivalent real-time cp system G.* ■

The Exchange Theorem enables us to swap equivalent pieces of successful computations in a cp system to obtain new computations. (Using it we will show in Section 4 that given a suitable successful computation we can "pump it up" and obtain an infinite sequence of new computations.) In the formulation of the theorem we shall use the following notions.

Definition 2.6. Let $G=(G_1, G_2, R)$ be a cp system and let Σ_2 be the alphabet of G_2.

(1) A word $\alpha \in \Gamma(G)^*$ is *balanced*, if $wdes(\alpha) \in D_{\Sigma_2}$.

(2) Two nonempty words α and β in $\Gamma(G)^*$ are *equivalent*, denoted $\alpha \sim \beta$, if they are balanced and $\alpha(1)=\beta(1)$, $\alpha(|\alpha|)=\beta(|\beta|)$. ■

Theorem 2.4 (Exchange Theorem): *Let G be a cp system and let ρ_1, ρ_2 be two (not necessarily different) successful computations in G, where $trl(\rho_1)=\alpha_1 \beta_1 \gamma_1$ and $trl(\rho_2)=\alpha_2 \beta_2 \gamma_2$ with $\beta_1 \sim \beta_2$.*

Let $\omega_{12}=\alpha_1 \beta_2 \gamma_1$ and $\omega_{21}=\alpha_2 \beta_1 \gamma_2$.

Then there exist (unique) successful computations ρ_{12} and ρ_{21} in G such that $trl(\rho_{12})=\omega_{12}$ and $trl(\rho_{21})=\omega_{21}$. ■

3. COMBINATORIAL PROPERTIES OF DYCK WORDS

In this section we prove some basic properties of Dyck words. The results we prove are useful in the analysis of computations in cp systems. As a matter of fact in the next section we will demonstrate the use of these properties in providing an alternative proof for the classical pumping lemma for context-free languages.

We start by defining the following very basic notions.

Definition 3.1: Let $w \in D_\Sigma$.

(1) A pair $(i, j) \subseteq fs(w)$ is called a *(w-)balanced pair* if $w(i) w(i+1) \ldots w(j)$ forms a balanced subword of w.

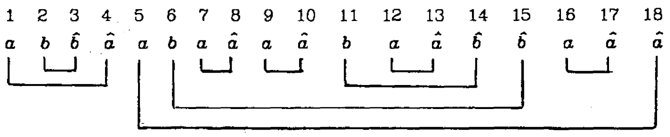
(2) A w-balanced pair (i, j) is called a *(w-)nested pair* if either $j=i+1$ or $(i+1, j-1)$ is a w-balanced pair. ■

Definition 3.2: Let $w \in D_\Sigma$ and let $U=(i, i+1, \ldots, j)$ be a nonempty segment in w. U is called *(w-)balanced* (*(w-)nested*), if (i, j) is a w-balanced (w-nested) pair in w. ■

We will say that a word $w \in D_\Sigma$ is *nested* if $fs(w)$ is a w-nested segment.

Remark: Note that a w-balanced pair (i, j) is not w-nested if and only if there exists an occurence k, $i<k<j-1$, such that (i, k) and $(k+1, j)$ are w-balanced pairs. In [B] nested words are called *(restricted) Dyck primes;* they are the balanced words that are not the product of two nonempty balanced words. ■

Example 3.1: Let $w = abb\hat{b}\hat{a}aba\hat{a}a\hat{a}ba\hat{a}\hat{b}\hat{b}a\hat{a}\hat{a} \in D_{\{a,\, b\}}$. w has the following nested structure:



(6, 15) and (9, 14) are w-balanced pairs.

(9, 14) is not a w-nested pair, since the subword $\hat{a}ba\hat{a}$ of w is not balanced.

(1, 4), (6, 15) and (11, 14) are w-nested pairs. ■

It turns out that Dyck words must contain balanced segements satisfying particular length constraints. This result will be used extensively in the sequel.

THEOREM 3.1: *Let $w \in D_\Sigma$ and let m be an integer such that $1 \leqq m \leqq (1/2)|w|$. Then there exists a w-balanced segment U with $m < \# U \leqq 2m$.*

Proof: If $m = (1/2)|w|$, then $U = fs(w)$ satisfies our lemma.

So we consider m with $m < (1/2)|w|$.

Let U_0 be a balanced segment in w such that $\# U_0 > 2m$ and moreover U_0 is a shortest w-balanced segment with that property; hence if U' is w-balanced and $\# U' > 2m$, then $\# U' \geqq \# U_0$. It is obvious that such a subword exists.

We consider separately two cases.

(i) $U_0 = (i, i+1, \ldots, j)$ is w-nested.

Then $U_1 = (i+1, \ldots, j-1)$ is w-balanced and clearly shorter than U_0. We have $2m \geqq \# U_1 = \# U_0 - 2 > 2m - 2$.

Consequently $\# U_1 = 2m$ (remember that U_1 is w-balanced and thus has an even number of elements) and so $U = U_1$ satisfies the lemma.

(ii) U_0 is not w-nested.

Then $U_0 = U_1 \cup U_2$ for some nonempty w-balanced segments $U_1 < U_2$. Both U_1 and U_2 are smaller than U_0. Hence $\# U_1 \leqq 2m$ and $\# U_2 \leqq 2m$. On the other hand either $\# U_1 > m$ or $\# U_2 > m$ (because $2m < \# U_0 = \# U_1 + \# U_2$). Thus one of the assignments $U = U_1$ or $U = U_2$ satisfies the statement of the lemma ■

The relative positions of w-nested words in a Dyck word fall into two basic categories: w-chain and w-cochain. This fact will be exploited in many considerations to follow.

DEFINITION 3.3: Let $w \in D_\Sigma$.

(1) A sequence $\kappa = (i_1, j_1), \ldots, (i_m, j_m)$, $m > 0$, of w-nested pairs is called a *w-chain* if $i_1 < i_2 < \ldots < i_m < j_m < \ldots < j_2 < j_1$.

m is the *length of* κ and is denoted by $|\kappa|$.

For each $1 \leqq l \leqq m$, (i_l, j_l) is referred to as a (*nested*) *pair of* κ.

(2) A sequence $\kappa = (i_1, j_1), \ldots, (i_m, j_m)$, $m \geqq 0$, of w-nested pairs is called a w-cochain if $i_1 < j_1 < i_2 < j_2 < \ldots < i_m < j_m$ and if, for every $1 \leqq k < l \leqq m$, (i_l, j_l) is w-balanced.

m is the *length of* κ and is denoted by $|\kappa|$.

For each $1 \leqq l \leqq m$, (i_l, j_l) is referred to as a (*nested*) *pair of* κ. ■

Example 3.1 (continued): $\kappa = (5, 18), (6, 15), (11, 14), (12, 13)$ is a w-chain, but also $\kappa' = (5, 18), (11, 14)$ is a w-chain. They have length 4 and 2 respectively.

$\mu = (7, 8), (9, 10), (11, 14)$ is a w-cochain of length 3.

$\mu' = (7, 8), (11, 14)$ is a w-cochain of length 2.

The sequence $(1, 4), (6, 15)$ consists of w-nested pairs, but it is not a w-cochain because it does not satisfy the second requirement: $(1, 15)$ is not w-balanced. ■

DEFINITION 3.4: Let $w \in D_\Sigma$.

(1) The *depth* of w, denoted $dp(w)$, is the length of the longest w-chain.

(2) The *width of* w, denoted $wd(w)$, is the length of the longest w-cochain. ■

Note that if U is a w-balanced segment and $u = w(U)$, then $dp(u) \leqq dp(w)$ and $wd(u) \leqq wd(w)$, because every u-(co)chain μ obviously corresponds to a w-(co)chain κ with $|\kappa| = |\mu|$.

Example 3.1 (continued): $dp(w) = 4$, $wd(w) = 3$. ■

The following relationship holds between the depth, width and length of a Dyck word. (This result was already obtained in [AB]; for the sake of completeness we give its proof here.)

THEOREM 3.2: *Let w be a word in* D_Σ. *Then* $|w| \leqq 2(q + q^2 + \ldots + q^p)$, *where* $p = dp(w)$ *and* $q = wd(w)$.

Proof: We keep q fixed and prove the lemma by induction on p.

Let $w \in D_\Sigma$ and let $p = dp(w)$, $q = wd(w)$.

(i) If $p = 0$, then obviously $w = \Lambda$.

Hence $|w| = 0$ (note that for $p = 0$ the sum $q + q^2 + \ldots + q^p$ becomes 0).

(ii) Induction step. Let $p \geqq 1$. We assume the lemma holds for every $w' \in D_\Sigma$ with $dp(w') < p$.

We can decompose w into nested subwords: for some $m \leqq q$ there exist m balanced words $w_1, \ldots, w_m$ such that $w = \sigma_1 w_1 \hat{\sigma}_1 \ldots \sigma_m w_m \hat{\sigma}_m$, where $\sigma_1, \ldots, \sigma_m \in \Sigma$.

Obviously, for every $1 \leqq i \leqq m$, $p_i = dp(w_i) \leqq p-1$ and $q_i = wd(w_i) \leqq q$.

Consequently, by our assumption,

$$|w_i| \leqq 2(q_i + q_i^2 + \ldots + q_i^{p_i - 1}) \leqq 2(q + q^2 + \ldots + q^{p-1}).$$

Thus

$$\begin{aligned}|w| &= 2m + |w_1| + \ldots + |w_m| \\ &\leqq 2m + m\,2(q + q^2 + \ldots + q^p + \ldots q^{p-1}) \leqq 2(q + q^2 + \ldots + q^p).\end{aligned}$$ ■

The following corollary of the above result will be especially useful in the sequel.

COROLLARY 3.3: *Let $p \geqq 1$, $q \geqq 2$ and let $w \in D_\Sigma$ be such that $|w| \geqq 4q^p$. Then either $dp(w) > p$ or $wd(w) > q$.*

Proof: Assume to the contrary that, for a word $w \in D_\Sigma$ with $|w| \geqq 4q^p$, both $dp(w) \leqq p$ and $wd(w) \leqq q$.

Then, combining Theorem 3.2 with the assumption of the corollary, we get $4q^p \leqq |w| \leqq 2(q + q^2 + \ldots + q^p)$.

However

$$2(q + q^2 + \ldots + q^p) = 2q\frac{q^p - 1}{q-1} \leqq 4(q^p - 1),$$

and so we get $4q^p \leqq 4(q^p - 1)$; a contradiction.

Hence the result holds. ■

Once a w-chain or a w-cochain is fixed in a Dyck word w it leads to a natural partition (splitting) of w.

DEFINITION 3.5. Let $w \in D_\Sigma$.

(1) Let $\kappa = (i_1, j_1), \ldots, (i_m, j_m)$, $m \geqq 1$, be a w-chain and let:

$U_0 = \{1, 2, \ldots, i_1 - 1\}$,

$U_l = \{i_l, i_l + 1, \ldots, i_{l+1} - 1\}$ for all $1 \leqq l < m$,

$U_m = \{i_m, i_m + 1, \ldots, j_m\}$,

$U_{2m-l} = \{j_{l+1} + 1, \ldots, j_l - 1, j_l\}$ for all $1 \leqq l < m$, and

$U_{2m}=\{j_m+1, j_m+2, \ldots, |w|\}$.

The sequence $U_0, U_1, \ldots, U_{2m}$ is called the *k-splitting of w.*

(2) Let $k=(i_1, j_1), \ldots, (i_m, j_m), m \geqq 1$ be a w-cochain and let

$U_0=\{1, 2, \ldots, \mathrm{i}_1-1\}$,

$U_{2l-1}=\{i_l, i_l+1, \ldots, j_l\}$ for all $l \leqq l \leqq m$,

$U_{2l}=\{j_l+1, j_l+2, \ldots, i_{l+1}-1\}$ for all $1 \leqq l < m$,

and

$U_{2m}=\{j_m+1, j_m+2, \ldots, |w|\}$.

The sequence $U_0, U_1, \ldots, U_{2m}$ is called the *κ-splitting of w.* ■

We say that a segment V of w *contains* the *w-(co)chain* $\kappa=(i_1, j_1), \ldots, (i_m, j_m), m \geqq 1$, if $(i_1, j_1) \subseteq V ((i_1, j_m) \subseteq V$ respectively), or equivalently if $\bigcup_{k=1}^{2m-1} U_k \subseteq V$, where $U_0, U_1, \ldots, U_{2m}$ is the κ-splitting of w.

Example 3.1 (continued): The sequence

$$\{1,2,3,4\}, \{5\}, \{6,7, \ldots, 10\}, \{11\}, \{12,13\}, \{14\}, \{15\}, \{16,17,18\}, \varnothing$$

is the κ-splitting of w.

The κ′-splitting of w is the sequence

$$\{1, 2, 3, 4\}, \{5, 6, \ldots, 10\}, \{11, 12, 13, 14\}, \{15, 16, 17, 18\}, \varnothing.$$

The μ-splitting of w is the sequence

$$\{1,2, \ldots, 6\}, \{7,8\}, \varnothing, \{9,10\}, \varnothing, \{11,12,13,14\}, \{15,16,17,18\}.$$

The μ′-splitting of w equals $\{1, 2, \ldots, 6\}, \{7, 8\}, \{9, 10\}, \{11, 12, 13, 14\}, \{15, 16, 17, 18\}$. ■

In analyzing the structure of Dyck words (especially in the context of computations in cp systems) it is often useful to group together various occurences in a Dyck word. This leads us to the following notion.

DEFINITION 3.6: Let $w \in D_\Sigma$ and let Γ be an alphabet.

(1) A *Γ-coloring of w* is a mapping $\delta : fs(w) \rightarrow \Gamma$.

The cardinality of Γ is called the *index of* δ and is denoted as *ind*(δ).

Let δ be a Γ-coloring of w.

(2) Two occurrences i and j in w are *δ-equivalent* if $\delta(i)=\delta(j)$.

Two pairs (i_1, j_1) and (i_2, j_2) of occurrences in w are δ-equivalent if $\delta(i_1)=\delta(i_2)$ and $\delta(j_1)=\delta(j_2)$.

(3) A w-(co) chain is called *δ-uniform* if all its nested pairs are δ-equivament. ■

Example 3.1 (continued): We define the $\{0, 1\}$-coloring δ of w as follows.

For $k\in\{1, 2, \ldots, 18\}$, $\delta(k)=\begin{cases}1, & \text{if } k \text{ is prime,}\\ 0, & \text{otherwise.}\end{cases}$

Then κ' is a δ-uniform w-chain. μ' is a δ-uniform w-cochain. Both κ and μ are not δ-uniform. ■

Our next result formulates the basic property of colorings of Dyck words. We start with the following lemma.

LEMMA 3.4: *Let $r, p\geqq 1$, $q\geqq 2$. Let $w\in D_\Sigma$ and let δ be a coloring of w with $ind(\delta)\leqq r$.*

If $|w|\geqq 4(qr^2)^{pr^2}$, then there exists:

either a δ-uniform w-chain κ with $|\kappa|>p$

or a δ-uniform w-cochain κ with $|\kappa|>q$.

Proof: According to Corollary 3.3 either $dp(w)>pr^2$ or $wd(w)>qr^2$.
Assume that $dp(w)>pr^2$.

Hence there exists a w-chain μ with $|\mu|>pr^2$. δ may have at most r^2 values on the set of all balanced pairs in w. Consequently more than p pairs of μ must be δ-equivalent. These pairs form a w-chain κ, with $|\kappa|>p$, which is δ-uniform.

If, on the other hand, $wd(w)>qr^2$, then by the same arguments there exists a δ-uniform w-cochain κ with $|\kappa|>q$. ■

THEOREM 3.5: *Let $r, p\geqq 1$, $q\geqq 2$. Let $w\in D_\Sigma$ and let δ be a coloring of w with $ind(\delta)<r$.*

If U is a w-balanced segment with $\# U\geqq 4(qr^2)^{pr^2}$, then

either U contains a δ-uniform w-chain κ, with $|\kappa|>p$,

or U contains a δ-uniform w-cochain κ, with $|\kappa|>q$.

Proof: Apply Lemma 3.4 to $w'=w(U)$. Obviously a w'-(*co*)chain κ' corresponds to a w-(co)chain κ with $|\kappa|=|\kappa'|$. ■

4. THE CLASSICAL PUMPING LEMMA FOR CONTEXT-FREE LANGUAGES

In this section we demonstrate how using results from the last section (combined with the Exchange Theorem) one proves (a somewhat strengthened version of) the classical pumping lemma for context-free languages.

We start by oberving the relationship between the length of a contribution from a segment of (the trail of) a successful computation and the number of occurrences of (letters corresponding to) right parentheses in the Dyck words corresponding to the weak description of the computation.

LEMMA 4.1: Let ρ *be a successful computation in a real-time cp system G. Let* $\alpha = trl(\rho)$, $\xi = wdes(\alpha)$ *and let U be a segment in* α.

Then $|ctb(\alpha(U))| = \#\{k \in U \mid \xi(k) \in \hat{\Sigma}_2\}$, *where* Σ_2 *is the alphabet of the second component of G. Furthermore, if* $\alpha(U)$ *is balanced, then* $|ctb(\alpha(U))| = (1/2)\#U$.

Proof: Let $G = (G_1, G_2, R)$ be a real-time cp system, where

$G_1 = (\Sigma_1, P_2, S_2, \Delta)$ and $G_2 = (\Sigma_2, P_2, S_2)$.

Then (directly from the real-time property) it follows that, for any $\tau \in \Gamma(G)$, $ctb(\tau) \in \Delta$ if $\tau = [\pi, 0]$ with $\pi \in R$ and $ctb(\tau) = \Lambda$, otherwise.

Moreover, for $\tau \in \Gamma(G)$, $wdes(\tau) \in \hat{\Sigma}_2$, if $\tau = [\pi, 0]$ with $\pi \in R$ and $wdes(\tau) \in \Sigma_2$, otherwise.

Now consider $\tau = \alpha(k)$ for some $k \in fs(\alpha)$.

Since *wdes* is a coding, $\xi(k) = wdes(\alpha(k))$. Consequently,

$ctb(\alpha(k)) = \Lambda$ if and only if $\xi(k) \in \Sigma_2$, and

$ctb(\alpha(k)) \in \Delta$ if and only if $\xi(k) \in \hat{\Sigma}_2$.

From the above the lemma easily follows. ■

Given a word w over an alphabet Σ, we can interpret w as a function from $fs(w)$ into Σ, which maps an occurence k in w to the letter $w(k)$ in Σ.

So, let α be the trail of a successful computation in a cp system G and let ξ be the weak description of α. Then $fs(\alpha) = fs(\xi)$, hence, using the above interpretation of α, we can regard α as a $\Gamma(G)$-coloring of ξ. ξ itself is a balanced word (see Lemma 2.1) thus it is now possible to talk about α-equivalent balanced pairs of ξ. As our next lemma shows such pairs are closely related to equivalent (in the sense of Definition 2.6. (2)) subwords in the trail α of ρ.

LEMMA 4.2: Let ρ *be a successful computation in a cp system G and let* $\alpha = trl(\rho)$ *and* $\xi = wdes(\alpha)$. *Let* (i_1, j_1), (i_2, j_2) *be two* α*-equivalent balanced* *pairs in ξ and let $U_1=(i_1, i_1+1, \ldots, j_1)$, $U_2=(i_2, i_2+1, \ldots, j_2)$. Then $\alpha(U_1)\sim\alpha(U_2)$.*

Proof: (1) First we will show that $\alpha(U_1)$ and $\alpha(U_2)$ are balanced. This is seen as follows.

According to our assumption (i_1, j_1) is a balanced pair in ξ. This means that $\xi(U_1)\in D_{\Sigma_2}$. But *wdes* is a coding, hence $\xi(U_1)=wdes(\alpha(U_1))$. Now, by Definition 2.6.(1), $\alpha(U_1)$ is balanced.

The same argument used for (i_2, j_2) leads to the conclusion that $\alpha(U_2)$ is balanced.

(2) We now prove the equivalence of $\alpha(U_1)$ and $\alpha(U_2)$.

(i_1, j_1) and (i_2, j_2) are α-equivalent pairs in ξ, so we have $\alpha(i_1)=\alpha(i_2)$ and $\alpha(j_1)=\alpha(j_2)$. But this implies that the first (and last) letters of $\alpha(U_1)$ and $\alpha(U_2)$ are equal to each other, because $\alpha(U_1)=\alpha(i_1)\alpha(i_1+1)\ldots\alpha(j_1)$ and $\alpha(U_2)=\alpha(i_2)\alpha(i_2+1)\ldots\alpha(j_2)$.

This completes the proof of the lemma. ■

The next lemma is about the "regularity" of occurrences of right letters between the occurrences of the letters in Dyck words (corresponding to weak descriptions of successful computations).

LEMMA 4.3: *Let ρ be a successful computation in a cp system G and let $\alpha=trl(\rho)$ and $\xi=wdes(\alpha)$. Let i and j be two α-equivalent occurrences in ξ of left letters, where $i<j$. Then there exists an occurrence k with $i<k<j$ such that $\xi(k)$ is a right letter.*

Proof: Let $i<j$ be two occurrences in ξ as in the statement of the lemma. $\alpha(i)\neq[S_1;\ S_2]$, because otherwise the symbol $[S_1;\ S_2]$ would occur twice in α.

So let $\alpha(i)=\alpha(j)=[\pi,\ t]$ for some $\pi\in R$, $t\in\mathbf{N}$.

Then clearly $t>0$, because $\xi(i)$ and $\xi(j)$ have to be left letters.

Since ξ is the weak description of a computation, there exists an occurrence k with $i<k<j$ such that $\alpha(k)=[\pi,\ 0]$. Obviously k is an occurrence of a right letter in ξ. ■

We are now ready to provide an alternative proof of the pumping property.

THEOREM 4.4: *Let K be a context-free language over an alphabet Δ.*

Then there exists a constant $d\in\mathbf{N}^+$ such that, for every $w\in K$ with $|w|\geq d$, there exist words $w_1, w_2, w_3, w_4, w_5\in\Delta^*$ satisfying:

(i) $w=w_1 w_2 w_3 w_4 w_5$,

(ii) $w_i\neq\Lambda$ *for all* $1\leq i\leq 5$,

(iii) $|w_2 w_3 w_4|\leq d$ *and*

(iv) $w_1 w_2^n w_3 w_4^n w_5 \in K$ *for every* $n \in \mathbf{N}$.

Proof: Let $G=(G_1, G_2, R)$ be a real-time cp system computing the language $K=L(G)$, where $G_1=(\Sigma_1, P_1, S_1, \Delta)$ and $G_2=(\Sigma_2, P_2, S_2)$.

We choose $d=4(4r^2)^{(2r^2)}$, where $r=\#\Gamma(G)$ and we will prove that the theorem holds for this choice of d.

Now consider $w \in K$ with $|w| \geqq d$ and let ρ be a successful computation in G with $res(\rho)=w$. Met $\alpha=trl(\rho)$ and $\xi=wdes(\alpha)$. Thus $w=ctb(\alpha)$.

As before we can regard α as a $\Gamma(G)$-coloring of ξ. Then α, seen as a coloring, has index r.

By Lemma 2.1, ξ is a word in D_{Σ_2} and so it is possible to apply to ξ our results on balanced words from Section 3.

The relationship between occurrences in w and occurrences of right letters in ξ as well as the relationship between occurrences in α and ξ has already been discussed to some extent in the proof of Lemma 4.1. From this discussion it easily follows that $|\alpha|=|\xi|=2|w|>2d$.

According to Theorem 3.1 there is a ξ-balanced segment U such that $d<\#U \leqq 2d$.

Theorem 3.5 implies that U contains

either an α-uniform ξ-chain κ with $|\kappa|=3$

or an α-uniform ξ-cochain κ with $|\kappa|=5$.

We consider separately each of these cases.

(*a*) $\kappa=(i_1, j_1), (i_2, j_2), (i_3, j_3)$ is an α-uniform ξ-chain contained in U.

Let $U_0, U_1, \ldots, U_6$ be the κ-splitting of ξ.

Then let

$$W_1=U_0 \cup U_1, \qquad W_2=U_2, \qquad W_3=U_3,\ W_4=U_4 \quad \text{and} \quad W_5=U_5 \cup U_6.$$

We will consider the image of the sequence W_1, W_2, W_3, W_4, W_5 both in ξ and in α; let

$$\xi_i=\xi(W_i), \qquad \alpha_i=\alpha(W_i) \quad \text{for } i=1, \ldots, 5.$$

Clearly $\xi_i=wdes(\alpha_i)$ for $1 \leqq i \leqq 5$.

Note that (i_2, j_2) and (i_3, j_3) are α-equivalent balanced pair in ξ. Thus, according to Lemma 4.2,

$$\alpha_3=\alpha(W_3) \sim \alpha(W_2 \cup W_3 \cup W_4)=\alpha_2 \alpha_3 \alpha_4.$$

Now it is possible "to pump" the pieces α_2 and α_4 in the computation ρ as follows.

Let $\rho_1 = \rho$ and apply the Exchange Theorem to ρ_1 and ρ; there exist (unique) successful computations ρ_0 and ρ_2 in G such that $trl(\rho_0) = \alpha_1 \alpha_3 \alpha_5$ and $trl(\rho_2) = \alpha_1 \alpha_2 \alpha_2 \alpha_3 \alpha_4 \alpha_4 \alpha_5$.

Once more we consider the equivalent pieces $\alpha_2 \alpha_3 \alpha_4$ and α_3, this time in ρ and ρ_2 respectively. If we apply the Exchange Theorem to these pieces we obtain a successful computation ρ_3 in G such that $trl(\rho_3) = \alpha_1 \alpha_2^2 \alpha_2 \alpha_3 \alpha_4 \alpha_4^2 \alpha_5$.

Continuing this process inductively yields an infinite number of successful computations ρ_0, ρ_1, ρ_2, ρ_3, ... in G with $trl(\rho_n) = \alpha_1 \alpha_2^n \alpha_3 \alpha_4^n \alpha_5$ for all $n \in \mathbf{N}$.

Let, for $1 \leqq i \leqq 5$, $w_i = ctb(\alpha_i)$.

Then $w = res(\rho) = ctb(\alpha) = ctb(\alpha_1 \alpha_2 \alpha_3 \alpha_4 \alpha_5) = w_1 w_2 w_3 w_4 w_5$ and, for all $n \in \mathbf{N}$, $res(\rho_n) = ctb(\alpha_1 \alpha_2^n \alpha_3 \alpha_4^n \alpha_5) = w_1 w_2^n w_3 w_4^n w_5$.

This proves the existence of words w_i satisfying requirements (i) and (iv) from the statement of the theorem.

We proceed now by proving that $|w_2 w_3 w_4| \leqq d$. First we observe that $W_2 \cup W_3 \cup W_4 \subseteqq U$, because κ is contained in U.

Consequently

$$|w_2 w_3 w_4| = |ctb\,\alpha(W_2 \cup W_3 \cup W_4)| \leqq |ctb\,\alpha(U)|.$$

On the other hand, using Lemma 4.1 we see that

$$|ctb\,\alpha(U)| = (1/2)\,\#U$$

– remember that U is a ξ-balanced segment.

Hence $|w_2 w_3 w_4| \leqq 1/2\,\#U \leqq d$.

This proves (iii).

Finally we will show that each of the words w_i is nonempty.

This is clear for w_3, w_4 and w_5, because each of ξ_3, ξ_4 and ξ_5 explicitly contains a right letter ($\xi(j_3)$, $\xi(j_2)$ and $\xi(j_1)$ respectively) and the corresponding symbols in α contibute letters to $ctb(\alpha_3)$, $ctb(\alpha_4)$ and $ctb(\alpha_5)$.

So we are left with w_1 and w_2.

i_1, i_2 and i_3 are α-equivalent occurences of left letters in ξ. By Lemma 4.3 there exist occurences k_1 and k_2 of right letters in ξ such that $i_1 < k_1 < i_2 < k_2 < i_3$. Hence ξ_1 and ξ_2 contain occurrences of right letters.

Thus, by an argument as above, we conclude that w_1 and w_2 are nonempty.

This proves (ii) and concludes the proof of the theorem in the chain-case.

(*b*) $u=(i_1, j_1), \ldots, (i_5, j_5)$ is an α-uniform ξ-cochain contained in U.

Let $U_0, U_1, \ldots, U_{10}$ be the κ-splitting of ξ and let

$$W_1=U_0 \cup U_1 \cup U_2, \qquad W_2=U_3 \cup U_4,\ W_3=U_5,$$
$$W_4=U_6 \cup U_7 \qquad \text{and} \qquad W_5=U_8 \cup U_9 \cup U_{10}.$$

As in the first case we consider $\xi_i=\xi(W_i)$ and $\alpha_i=\alpha(W_i)$, for $1 \leqq i \leqq 5$, and we continue the proof by showing that for each $n \in \mathbf{N}$, $\alpha_1 \alpha_2^n \alpha_3 {}_4^n \alpha_5$ is the trail of a successful computation in G. We then prove (analogously to (a) above) that conditions (i) through (iv) hold. Actually, the situation is somewhat simpler now—to see that the words $w_i=ctb(\alpha_i)$, $1 \leqq i \leqq 5$, are nonempty we observe immediately that each of the words ξ_i contains an occurrence of a right letter.

Thus the theorem holds also in the cochain case.

From (a) and (b) the theorem follows. ■

We conclude this section by the following remarks.

In the proof of Theorem 4.4 we have analyzed separately two cases: the "chain" and the "cochain" case. The analysis of these two cases lead us to the classical context-free pumping property.

Let us consider now the "cochain" case in more detail—in this way we will obtain a "regular-like" pumping property.

LEMMA 4.5: *Let* ρ *be a successful computation in a real-time cp system* G *and* let $\alpha=trl(\rho)$ *and* $\xi=wdes(\alpha)$. *If there exists an* α*-uniform* ξ*-cochain of length* 3, *then there exist nonempty words* w_1, w_2 *and* w_3 *such that:*

(i) $res(\rho)=w_1 w_2 w_3$ *and*

(ii) $w_1 w_2^* w_3 \subseteq L(G)$.

Proof: Let k be an α-uniform ξ-cochain with $|\kappa|=3$ and let $U_0, U_1, \ldots, U_6$ be the κ-splitting of ξ.

For $i=0, 1, \ldots, 6$ let $\alpha_i=\alpha(U_i)$.

Then $\alpha_1 \sim \alpha_1 \alpha_2 \alpha_3$. Within the trail of ρ we apply repeatedly the Exchange Theorem to α_1 and $\alpha_1 \alpha_2 \alpha_3$ to obtain successful computations ρ_0, $\rho_1=\rho$, ρ_2, ρ_2, $\rho_3, \ldots$ such that, for each $n \in \mathbf{N}$,

$$trl(\rho_n)=\alpha_0 \alpha_1 (\alpha_2 \alpha_3)^n \alpha_4 \alpha_5 \alpha_6.$$

Hence if we write

$$w_1=ctb(\alpha_0 \alpha_1), \qquad w_2=ctb(\alpha_2 \alpha_3) \quad \text{and} \quad w_3=ctb(\alpha_4 \alpha_5 \alpha_6),$$

then

$$res(\rho_n) = w_1 w_2^n w_3 \qquad \text{for each} \quad n \in \mathbf{N}.$$

But w_1, w_2 and w_3 are nonempty, because U_1, U_3 and U_5 contain occurrences of right letters in ξ and consequently $ctb(\alpha_1)$, $ctb(\alpha_3)$ and $ctb(\alpha_5)$ are nonempty—*see* Lemma 4.1.

From these considerations the lemma follows. ■

Discussion

In this paper we have exploited the "cp system point of view" in the analysis of the structure of computations on a push-down store.

In particular we have depicted a number of properties of Dyck words that seem to be very basic in such an analysis. We believe that these results are of independent interest in the general theory of Dyck words.

We have demonstrated the use of these results (combined with the Exchange Theorem) in providing an alternative proof of the classical pumping property of context-free languages. We have tried to illustrate that the Exchange Theorem and the combinatorial properties of Dyck words we have given in this paper form a very basic and useful set of tools in the investigation of context-free languages (through push-down computations and not through derivation trees in context-free grammars!).

This paper points to (at least) two areas of research which seem to be worthwhile to continue.

(1) Analyze in more detail the combinatorial structure of Dyck words, so that it can be connected to other known (and hopefully new) pumping properties of context-free languages.

(2) Can a comparative study of the chain and cochain cases lead to a combinatorial characterization of those context-free languages that are regular?

We believe that further research in this direction will increase our understanding of the nature of push-down compulations (and context-free languages). As a matter of fact in the second part of this paper we present results pertinent to (1) above. We will investigate the structure of sparse subwords of Dyck words and use our results about this sparse structure to derive Ogden's pumping lemma for context-free languages.

ACKNOWLEDGMENTS

The authors are very obliged to the Leiden TIC group and in particular to J. Engelfriet, H.C.M. Kleijn and E. Welzl for very useful comments concerning the first draft of this paper. We are indepted to the referee of this paper for pointing out a number of improvements. The first and the third author gratefully acknowledge the support of NSF grant MCS 83-05245.

REFERENCES

[AB] J. M. Autebert and J. Beauquier, *Une caracterisation des generateurs standard*, R.A.I.R.O., R-1, 1974, pp. 63-83.

[BPS] Y. Bar-Hillel, M. Perles and E. Shamir *On Formal Properties of Simple Phrase-structure Grammars*, Zeitschrift für Phonetik, Sprachwissenschaft, und Kommunikationsforschung, Vol. 14, 1961, pp. 143-177.

[B] J. Berstel *Transductions and Context-free Languages*, Teubner, Stuttgart, 1979.

[EHR1] A. Ehrenfeucht, H. J. Hoogeboom and G. Rozenberg, *Real-time coordinated pair systems*, Dept. of Computer Science, University of Colorado at Boulder, Techn. Rep. No. CU-CS-259-83, 1983.

[EHR2] A. Ehrenfeucht, H. J. Hoogeboom and G. Rozenberg, *Computations in Coordinated pair Systems*, Fundamenta Informaticae (to appear).

[H] M. Harrison, *Introduction to Formal Language Theory*, Addison-Wesley Publ. Co., Reading, Massachusetts, 1978.

[R] G. Rozenberg, *On Coordinated Selective Substitutions: Towards a Unified Theory of Grammars and Machines*, Theoretical Computer Sciences, Vol. 37, 1985, pp. 31-50.

[S] A. Salomaa, *Formal languages*, Academic Press, London-New York, 1973.