On is an n-MCFL
Sylvain Salvati

To cite this version:
Sylvain Salvati. On is an n-MCFL. [Research Report] Université de Lille, INRIA, CRIStAL CNRS. 2018. hal-01771670

HAL Id: hal-01771670
https://hal.archives-ouvertes.fr/hal-01771670
Submitted on 19 Apr 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
The language $O_n$ is the language built on the alphabet $\Sigma_n = \{a_i, b_i \mid i \in [1, n]\}$ and that contains exactly all those words $w$ which, for every $i$ in $[1, n]$, have the same number of occurrences of $a_i$ and of $b_i$. If we write $|w|_c$ the number of occurrences of the letter $c$ in $w$, then this condition becomes that for every $i$ in $[1, n]$, $|w|_{a_i} = |w|_{b_i}$. It has first been conjectured that the language $O_2$ was not a Multiple Context Free Language (MCFL), but it was subsequently shown that it actually was an MCFL \cite{Sal15} and more precisely an MCFL of dimension 2 (a 2-MCFL) while not being a well-nested MCFL of dimension 2 \cite{KS12} (a 2-MCFL\textsubscript{wn}). Giving a similar proof of the same result, Nederhof \cite{Ned16} conjectured that $O_n$ is an $n$-MCFL. As pointed in \cite{Ned16}, a simple pumping argument shows that for every $m < n$, $O_n$ cannot be an MCFL of dimension $m$. A recent breakthrough has been made about that conjecture by Meng-Che Ho \cite{Hoar} who proved that $O_n$ is an MCFL for every $n$. However the dimension of the MCFL induced by the proof is larger than $n$, it is $8 \left\lfloor \frac{n+1}{2} \right\rfloor - 2$. All the proofs related to these results are based on algebraic topology. While the result of \cite{Sal15} as well as the proof of \cite{Ned16} strongly use some properties of the plane (the existence of winding numbers of curves around points), the proof of \cite{Hoar} is based on a more general and well-known result of algebraic topology: Borsuk-Ulam Theorem. This theorem has topological proofs, but it can also be seen as a consequence of a combinatorial lemma of Tucker \cite{Tuc45}. In this paper, we present a rather elementary proof that $O_n$ is an $n$-MCFL using octahedral Tucker’s Lemma.

Interestingly several theorems that were proved using Borsuk-Ulam Theorem have been reproved using the octahedral Tucker’s Lemma. For example Matoušek \cite{Mat04} gave a combinatorial proof of Kneser Theorem proving and using that lemma. Similarly Pálvölgyi \cite{Pal09} used that lemma so as to give a combinatorial proof of the bound for the splitting necklace problem. Interestingly the first bounds for this problem were proved using Borsuk-Ulam theorem with the very same technique as the one used in \cite{Hoar} so as to prove that $O_n$ is an MCFL.

1 Preliminaries

We start out with some preliminaries. We introduce Multiple Context Free Grammars (MCFG) and their dimension.
For a given finite set $\Sigma$, we write $\Sigma^*$ for the monoid freely generated by $\Sigma$, and $\Sigma^+$ the free semigroup generated by $\Sigma$. The elements of $\Sigma$ are called letters while the elements of $\Sigma^*$ are called strings or words and we write $\epsilon$ for the empty word. Given a word $w$, we write $|w|$ for its length, and $|w|_c$ for the number of occurrences of the letter $c$ in $w$. Given two words $w_1$ and $w_2$ we write $w_1 \shuffle w_2$ for the shuffle of $w_1$ and $w_2$, that is the set of words $u_1v_1\ldots u_nv_n$ so that $w_1 = u_1\ldots u_n$ and $w_2 = v_1\ldots v_n$ (here $n$ is arbitrary and some of the $u_i$’s or $v_i$’s can be the empty word).

A ranked alphabet $\Omega$ is a pair $(\mathcal{A}, \rho)$ where $\mathcal{A}$ is a finite set and $\rho$ is a function from $\mathcal{A}$ to $\mathbb{N}$. Given $a \in \mathcal{A}$, $\rho(a)$ is the rank of $a$. We shall write $\Omega^{(n)}$ for the set $\{a \in \mathcal{A} \mid \rho(a) = n\}$.

A Multiple Context Free Grammar (MCFG) $G$ is a tuple $(\Omega, \Sigma, R, S)$ where $\Omega$ is a ranked alphabet, $\Sigma$ is a finite set of letters, $R$ is a set of rules and $S$ is an element of $\Omega^{(k)}$. The rules in $R$ are of the form

$$A(\alpha_1, \ldots, \alpha_n) \Rightarrow B_1(x_1, 1, \ldots, x_{1,l_1}), \ldots, B_p(x_p, 1, \ldots, x_{p,l_p})$$

where $A$ is in $\Omega^{(n)}$, $B_j$ is in $\Omega^{(l_j)}$, the $x_{k,j}$ are pairwise distinct variables and the $\alpha_j$ are elements of $(\Sigma \cup X)^*$ with $X = \{x_{k,j} \mid k \in [p] \wedge j \in [l_k]\}$ and the restriction that each $x_{k,j}$ may have at most one occurrence in the string $\alpha_1 \cdot \ldots \cdot \alpha_n$. Note that $p$ may be equal to 0 in which case the right part of the rule (the one on the right of the $\Rightarrow$ symbol) is empty, in such a case we will write the rule by omitting the symbol $\Rightarrow$.

An MCFG such as $G$ defines judgments of the form $\vdash_G A(s_1, \ldots, s_n)$ where $A$ is in $\Omega^{(n)}$ and $s_i$ belongs to $\Sigma^*$. Such a judgment is said to be derivable when there is a rule $A(\alpha_1, \ldots, \alpha_n) \Rightarrow B_1(x_1, 1, \ldots, x_{1,l_1}), \ldots, B_p(x_p, 1, \ldots, x_{p,l_p})$ and there are derivable judgments $\vdash_G B_k(w_{k,1}, \ldots, w_{k,l_k})$ for all $k$ in $[p]$ such that $s_j$ is equal to $\alpha_j$ where the possible occurrences of the $x_{k,j}$ are replaced by $w_{k,j}$. The language defined by $G$, $\mathcal{L}(G)$, is the set $\{w \in \Sigma^* \mid \text{$S(w)$ is derivable}$

Let $G_n$ be the $n$-MCFG that uses two non-terminals $S$ and $I$ that are respectively of arity 1 and $n$. The grammar uses the alphabet $\Sigma_n = \{a_i, b_i \mid i \in [1, n]\}$. The rules of the grammar are the following:

- $S(x_1 \ldots x_n) \Rightarrow I(x_1, \ldots, x_n)$,
- $I(s_1, \ldots, s_n) \Rightarrow I(x_1, \ldots, x_n), I(y_1, \ldots, y_n)$ where $s_1 \ldots s_n$ is in $x_1 \ldots x_n \shuffle y_1 \ldots y_n$,
- $I(a_i, b_i, \epsilon, \ldots, \epsilon)$ for every $i$ in $[1, n]$,
- $I(b_i, a_i, \epsilon, \ldots, \epsilon)$ for every $i$ in $[1, n]$,
- $I(\epsilon, \ldots, \epsilon)$.

Our main theorem is that the language of $G_n$ is $O_n$. The inclusion of $\mathcal{L}(G_n)$ into $O_n$ is obvious and our effort only bears on proving the converse inclusion. For this our main technical lemma is:

**Lemma 1** For every tuple $(s_1, \ldots, s_n)$ in $(\Sigma_n)^+$ so that $s_1 \ldots s_n$ is in $O_n$ and $|s_1 \ldots s_n| > 2$ there are tuples $(u_1, \ldots, u_n)$ and $(t_1, \ldots, t_n)$ so that
• \(|u_1 \ldots u_n| > 0, |t_1 \ldots t_n| > 0\)

• \(u_1 \ldots u_n\) and \(t_1 \ldots t_n\) are in \(O_n\) and,

• there is \(v_1 \ldots v_n\) in \(x_1 \ldots x_n \uplus y_1 \ldots y_n\) so that for every \(i\) in \([1, n]\), \(s_i\) is obtained by substituting in \(v_i\) the \(x_k\)'s by the corresponding \(u_k\)'s and the \(y_l\)'s by the corresponding \(t_l\)'s.

From this the inclusion follows easily.

**Proposition 1** We have \(O_n \subseteq \mathcal{L}(G_n)\).

**Proof**
To show this inclusion, we prove that for every \(w\) in \(O_n\) if \(w = s_1 \ldots s_n\), then \(I(s_1, \ldots, s_n)\) is derivable.

When for every \(i\), \(|s_i| \leq 1\), then it is obvious that \(I(s_1, \ldots, s_n)\) is derivable by induction on \(|s_1 \ldots s_n|\). Indeed, when all the \(s_i\)'s are empty then by definition of \(G_n\), \(I(s_1, \ldots, s_n)\) is derivable. Now suppose that \(s_i = a_k\), there must \(j\) so that \(s_j = b_k\). Without loss of generality let’s assume that \(i < j\). By induction we obtain that \(\vdash_{G_n} I(s_1, \ldots, s_i - 1, \epsilon, \ldots, s_k - 1, \epsilon, \ldots, s_n)\) is derivable. Then using the rule

\[
I(x_1, \ldots, x_{i-1}, x_i y_1, x_{i+1}, \ldots x_j y_2, \ldots, y_n, x_j, \ldots, x_n) \Rightarrow I(x_1, \ldots, x_n), I(y_1, \ldots, y_n)
\]

with the derivable judgments \(\vdash_{G_n} I(s_1, \ldots, s_i - 1, \epsilon, \ldots, s_k - 1, \epsilon, \ldots, s_n)\) and \(\vdash_{G_n} I(a_k, b_k, \epsilon, \ldots, \epsilon)\) we obtain a derivation of \(\vdash_{G_n} I(s_1, \ldots, s_n)\).

Now we assume that for some \(i\), \(|s_i| > 1\) and let \(p\) be the number of components of \((s_1, \ldots, s_n)\) which are empty. We then show by induction on the pair of numbers \((|s_1 \ldots s_n|, p)\) ordered lexicographically that \(I(s_1, \ldots, s_n)\) is derivable. In case \(p\) is equal to 0, it suffices to use Lemma 1. In case \(p > 0\), then there is \(i\) so that \(|s_i| > 1\) and \(j\) so that \(s_j = \epsilon\). Let us assume that \(i < j\), the other case is treated similarly. Since \(|s_i| > 1\) there are \(u\) and \(v\) so that \(s_i = uv\) and \(|u| > 0\) and \(|v| > 0\). Let \(J\) be the judgment \(\vdash_{G_n} I(s_1, \ldots, s_i - 1, u, s_i + 1, \ldots, s_j - 1, v, s_j + 1, \ldots, s_n)\). In case for all \(i\), the element of that tuple have length smaller than 1, we can use the result above to obtain that \(J\) is derivable. Otherwise, the induction hypothesis implies that \(J\) is derivable and using an appropriate rule, as above, we obtain that \(\vdash_{G_n} I(s_1, \ldots, s_n)\) is derivable.

From this we can derive our main theorem.

**Theorem 2** The language \(O_n\) is an \(n\)-MCFL.

### 2 The main technical Lemma

We now turn to the proof of Lemma 1. We start by proving it for relatively small tuples.
Lemma 3 For every tuple \((s_1, \ldots, s_n)\) in \((\Sigma_n)^+\) so that \(s_1 \ldots s_n\) is in \(O_n\) and 
\(4n > |s_1 \ldots s_n| > 2\), there are tuples \((u_1, \ldots, u_n)\) and \((t_1, \ldots, t_n)\) so that

- \(|u_1 \ldots u_n| > 0, |t_1 \ldots t_n| > 0\)
- \(u_1 \ldots u_n\) and \(t_1 \ldots t_n\) are in \(O_n\) and,
- there is \(v_1 \ldots v_n\) in \(x_1 \ldots x_n \cup \{y_1 \ldots y_n\}\) so that for every \(i\) in \([1, n]\), \(s_i\) is obtained by substituting in \(v_i\) the \(x_k\)’s by the corresponding \(u_k\)’s and the \(y_i\)’s by the corresponding \(t_i\)’s.

Proof
There are two simple cases where it is easy to find tuples \((u_1, \ldots, u_n)\) and
\((t_1, \ldots, t_n)\) that satisfy the conclusions of the lemma.

The first one is when there is \(i\) and \(j\) in \([1, n]\) (here \(i\) can be equal to \(j\)) so that \(a_k\) and \(b_k\) occur as the first or last letter of \(s_i\) or \(s_j\). For example when \(i < j\) and \(s_i = a_k s_i'\) and \(s_j = b_k s_j'\). In that case, it suffices to choose \(u_l = s_i\) for every \(l \neq i\) and \(l \neq j\), \(u_i = s_i'\), \(u_j = s_j'\) and \(t_1 = a_k, t_2 = b_k\), and \(t_j = \epsilon\) for \(j \in [3, n]\). All the other cases are handled similarly.

The second one is when some of the \(s_i\)’s is equal either to \(a_k\) or to \(b_k\). Assume that \(s_i = a_k\) (the other case is similar). Because \(s_1 \ldots s_n\) is in \(O_n\), there is \(j\) so that \(s_j = s_j' b_k s_j''\). Without loss of generality, we suppose that \(i < j\). In that case it suffices to take \(u_k = s_k\) for \(k < i\), \(u_k = s_{k+1}\) when \(i \leq k < j - 1\), \(s_{j-1} = s_j'\), \(s_j = s_j''\) and \(u_k = s_k\) when \(j < k\) and \(t_1 = a_k, t_2 = b_k\) and \(t_j = \epsilon\) for \(j \in [3, n]\).

The proof of this lemma consists in remarking that when \(|s_1 \ldots s_n| < 4n\), we must be in one of the two previous cases. We prove that if we are not in the second case, then we necessarily are in the first one. So suppose that for every \(i\), \(|s_i| > 1\), and that \(s_i = c_{i, 1} s_i' c_{i, 2}\) with \(c_{i, 1}, c_{i, 2} \in \Sigma_n\). For \(c \in \Sigma_n\), we introduce the notation \(c_i\) to be \(b_l\) when \(c = a_i\) and \(a_i\) when \(c = b_i\). If for every \((i, j), (k, l)\) in \([1, n] \times [1, 2]\), \(c_{i, j} \neq c_{l, j}\), then, as \(s_1 \ldots s_n\) is in \(O_n\), it must be the case that \(|s_1' \ldots s_n'| \geq 2n\) and thus that \(|s_1 \ldots s_n| \geq 4n\) which is in contradiction with the hypotheses. Therefore, there must be \((i, j)\) and \((k, l)\) in \([1, n] \times [1, 2]\), \(c_{i, j} = c_{l, j}\).

To complete the proof of Lemma 3, it remains to treat the case of large tuples. This is where we use the octahedral Tucker lemma. Tucker’s lemma is the combinatorial counter-part of Borsuk-Ulam theorem. This lemma is generally stated using an antipodally symmetric triangulation of the \(m\)-sphere. However an octahedral version has been given a combinatorial proof by Baker [Bak70] and later on it has been given an even more combinatorial statement together with a proof that does not require topological notions by Matoušek [Mat04]. This statement has been later on used to give a combinatorial proof of the necklace splitting problem [Păl09]. We take a more convenient and equivalent formulation of the octahedral Tucker lemma by Alishahi and Meunier [AM17]. This formulation uses strings built on the alphabet \(\mathbb{O} = [-1, 1]\). These string are ordered by the least congruence of monoid \(\preceq\) so that \(0 \preceq -1\) and \(0 \preceq 1\). In other words, \(x \preceq y\) when they have the same length and the respective letters of \(x\) and \(y\) at a position \(i\) may be different only when the \(i\)th letter of \(x\) is 0. We
write \(-\cdot\) the (unique) homomorphism that maps \(-1\) to 1, \(-1\) to 1 and 0 to 0. Then \(-x\) is the string that is obtained from \(x\) by replacing 1’s by \(-1\)’s and \(-1\)’s by 1’s. We may also write \(\pi\) for \(-x\).

**Lemma 4 (Octahedral Tucker lemma)** Given the natural numbers \(m\) and \(p\), if there is a map \(\lambda : \mathbb{O}^m - 0^* \rightarrow [-p, p] - \{0\}\) so that:

- \(\lambda(x) = -\lambda(-x)\),
- when \(x \leq y\), \(\lambda(x) + \lambda(y) \neq 0\),

then \(p \geq m\).

We use this lemma in a way that is similar to [Pal09]. Let \(\mathbb{B} = \{-1, 1\}\). In this setting given a string two strings \(w\) and \(x\) that have the same length and which are respectively built on \(\Sigma_n^+\) and \(\mathbb{B}^+\), the string \(w\) serves as a mask that describes a factorization of \(w\). For example, take \(\kappa \in \mathbb{B}\) and let \(x = k_1 \pi_1 \ldots k_r \pi_r\) with the \(l_i\)’s and \(k_i\)’s strictly positive, then \(w = u_1 v_1 \ldots u_r v_r\) with \(|u_i| = k_i\) and \(|v_i| = l_i\). Here, \(x\) describes uniquely decomposition of \(w\) into two \(r\)-tuples \((u_1, \ldots, u_r), (v_1, \ldots, v_r)\). When \(x\) is in \(\mathbb{B}^+\), \(x\) describes an unspecified factorization, i.e. a factorization where it remains to determine where the letters of \(w\) that are aligned with 0 have to go.

We generalize in an obvious way this notion of factorization induced by strings in \(\mathbb{O}^+\) to tuples \((s_1, \ldots, s_n)\) so that every \(s_i\) is in \(\Sigma_n^+\). Let \(m\) be \(|s_1 \ldots s_n|\), we are going to consider a strings \(x\) in \(-1, 1)^m\). Such a string \(x\) can be factored in \(x_1 \ldots x_n\) so that for every \(i\) in \([1, n]\), \(|x_i| = |s_i|\). Now each \(x_i\) determines a factorization of \(s_i\), say \(u_i = (u_{i1}, \ldots, u_{ik_i})\), \(v_i = (v_{i1}, \ldots, v_{il_i})\) where \(u_i\) corresponds to the parts of \(s_i\) that are marked with 1 and \(v_i\) corresponds to the parts of \(s_i\) that are marked with \(-1\). Now we let \(u\) be the tuple \((u_{i1}, \ldots, u_{ik_1}, \ldots, u_{ik_n}, \ldots, u_{ik_n})\) and \(v\) be the tuple \((v_{i1}, \ldots, v_{il_1}, \ldots, v_{il_n}, \ldots, v_{il_n})\); these tuples are the factorisations that \(x\) induces on the tuple \((s_1, \ldots, s_n)\). Again, when we consider \(x\) in \(\mathbb{O}^m\), it represents a partially defined factorization where the positions corresponding to 0 need to be assigned either to \(-1\) or to 1. The octahedral Tucker lemma will help us to prove that a factorization such as the one required in Lemma 4 always exists in the case \(m \geq 4n\).

From now on, we fix a tuple \(s = (s_1, \ldots, s_n)\), so that for every \(i\) in \([1, n]\), \(s_i \in \Sigma_n^+\), \(|s_1 \ldots s_n| = m\) and \(m \geq 4n\). We are going to define \(\lambda\) from \(\mathbb{O}^m\) to \([-m + 1, m - 1]\) and show using Lemma 4 that it has a 0. From this 0, we define \(\lambda\) requires a bit of work.

Taking \(x\) in \(\mathbb{O}^m - 0^*\), we let \(x = x_1 \ldots x_n\) so that \(|x_i| = |s_i|\) for every \(i\) in \([1, n]\). We define \(\text{ind}(x)\) to be the \(\min\{i \mid x_i \notin 0^*\}\). Notice that, since \(x\) cannot be in \(0^*\), \(\text{ind}(x)\) is well-defined. Now given \(x\) in \(\mathbb{B}^m\) we define \(\text{size}_\kappa(x)\) with \(\kappa \in \mathbb{B}\) to be the number of components of the tuple defined from \(s\) by the positions marked with \(\kappa\). We then let \(\text{size}(x) = \max(\text{size}_+(x), \text{size}_-(x))\). We can bound \(\text{size}(x)\) with \(\frac{m + n}{2}\). Indeed so as to maximize the size of a tuple, it suffices to take for every \(x_i\) a string which alternates between \(-1\) and 1 at each position.
In that case, if the length of $x_i$ is even, the number of factors that contribute the factorization of $s$ is the same for the $-1$ and 1. Otherwise when the length of $x_i$ is odd, the number of factors that contribute to the factorization of $s$ is one more for the symbol that starts $x_i$. Thus if $o$ is the number of $x_i$’s that have an odd length we can construct an $x$ that maximizes size($x$) and in that case size($x$) = $\frac{m+n}{2}$. As $o$ is smaller than $n$, we obtain the bound we have given.

Lemma 5 For every $x$ in $\mathbb{B}^m$, we have size($x$) $\leq \frac{m+n}{2}$.

We now extend the function size($\cdot$) to any $x$ in $\mathbb{O}^m$ as follows:

$$\text{size}(x) = \max\{\text{size}(y) \mid y \in \mathbb{B}^m \land x \preceq y\}.$$

By definition and Lemma 5 we also have that size($x$) $\leq \frac{m+n}{2}$. There are several simple properties that we can observe about the functions size($\cdot$) and ind($\cdot$).

Lemma 6 The functions size($\cdot$) and ind($\cdot$) are even and antitone, i.e. for every $x, y$ in $\mathbb{O}^m$ so that $x \preceq y$ we have:

- size($x$) = size($-x$), ind($x$) = ind($-x$),
- size($y$) $\leq$ size($x$) and ind($y$) $\leq$ ind($x$).

Proof
These properties are immediate consequences of the definitions. □

We now associate a sign to every $x$ in $\mathbb{O}^m$. Suppose that ind($x$) = $i$ and $x_i = 0^k\kappa u$ for some $\kappa \in \mathbb{B}$, $k \in \mathbb{N}$ and $u \in \mathbb{O}^*$, we let sign($x$) $\in \mathbb{B}$ be $\kappa \times (-1)^k$.

Notice that sign($\cdot$) is odd, i.e. that sign($-x$) = $-\text{sign}(-x)$.

Lemma 7 Given $x, y$ in $\mathbb{O}^m$, if $x \preceq y$, size($x$) = size($y$) and ind($x$) = ind($y$), then sign($x$) = sign($y$).

Proof
Let $i = \text{ind}(x)$, and suppose that $x_i = 0^k\kappa u$. So as to obtain $z$ in $\mathbb{B}^m$ so that $x \preceq z$ and size($z$) = size($x$). Then it must be the case that $z_i = u_1\kappa u_2$ so that $u_1\kappa$ alternates between $-1$ and $1$ at each letter. Notice then that sign($x$) is actually the first letter of $u_1$. It must then be the case that $y_i = v_1\kappa v_2$ with $v_1\kappa \preceq u_1\kappa$ because otherwise we would not have size($x$) = size($y$). But then as ind($x$) = $i = \text{ind}(y)$, we must also have that sign($y$) is the first letter of $u_1$. As a consequence we have sign($x$) = sign($y$). □

Given $x$ in $\mathbb{O}^m$, $c \in \Sigma_n$ and $\kappa \in \mathbb{O}$, we write $\delta_{c,\kappa}(x)$ for the number of $c$ in $s_1 \ldots s_n$ that are aligned with $\kappa$ in $x$. We then write $E_{\kappa,i}(x)$ for $\delta_{a_i,\kappa}(x) - \delta_{b_i,\kappa}(x)$. Notice that because $s_1 \ldots s_n$ is in $\mathbb{O}_n$, we have that $E_{1,i}(x) + E_{0,i}(x) + E_{-1,i}(x) =$
0 in particular, when \( x \) is in \( \mathbb{B}^m \) we have \( E_{1,i}(x) = -E_{-1,i}(x) \). We define \( \text{unb}_i(\cdot) \) that measures how unbalanced \( x \) is as follows for the letter \( i \),

\[
\text{unb}_i(x) = \begin{cases} 
1 & \text{when for every } y \in \mathbb{B}^m \text{ so that } x \leq y, \ E_{1,i}(x) > 0 \\
-1 & \text{when for every } y \in \mathbb{B}^m \text{ so that } x \leq y, \ E_{-1,i}(x) > 0 \\
0 & \text{otherwise}
\end{cases}
\]

notice that \( \text{unb}_i(\cdot) \) is odd, i.e. \( \text{unb}_i(x) = -\text{unb}_i(-x) \). Let \( U_x \) be the set \( \{ i \mid \text{unb}_i(x) \neq 0 \} \), we then let

\[
\text{unb}(x) = \begin{cases} 
\text{unb}_i(x)i & \text{when } U_x \neq \emptyset \land i = \min(U_x) \\
0 & \text{otherwise}
\end{cases}
\]

Again \( \text{unb}(x) = -\text{unb}(-x) \).

We can now define \( \lambda(\cdot) \) as follows:

\[
\lambda(x) = \begin{cases} 
\text{sign}(x)(\text{size}(x) + \text{ind}(x)) & \text{when } |x|_1 > 0 \land |x|_{-1} > 0 \land \text{size}(x) > n \\
n + 1 & \text{when } |x|_1 = 0 \\
-n - 1 & \text{when } |x|_{-1} = 0 \\
\text{unb}(x) & \text{when } |x|_1 > 0 \land |x|_{-1} > 0 \land \text{size}(x) \leq n
\end{cases}
\]

**Lemma 8** There is \( x \) so that \( \lambda(x) = 0 \).

**Proof**

We first remark that \( \lambda \) is odd, i.e. \( \lambda(x) = -\lambda(-x) \). Now supposing that for every \( x \), \( \lambda(x) \neq 0 \), we can prove that for every \( x, y \) in \( \mathbb{Q}^m \), if \( x \leq y \), then \( \lambda(x) + \lambda(y) \neq 0 \). But, \( |\lambda(x)| \) is bounded by \( \max(n + 1, \text{size}(x) + \text{ind}(x)) \), as \( \text{size}(x) \leq \frac{m+n}{2} \) and \( \text{ind}(x) \leq n \), we have that \( |\lambda(x)| \leq \frac{m+3n}{2} \). But we can prove \( \frac{m+3n}{2} \leq m - 1 \), which is equivalent to \( 3n + 2 \leq m \). Indeed, since, \( n \geq 2 \) and \( m \geq 4n \) we have \( 3n + 2 \leq 4n \leq m \). And as a conclusion \( |\lambda(x)| \leq m - 1 \). Then the use of the octahedral Tucker lemma leads to a contradiction and therefore there is \( x \) so that \( \lambda(x) = 0 \).

So as to get our conclusion, it remains to show that when \( x \leq y \), \( \lambda(x) + \lambda(y) \neq 0 \) under the assumption that for every \( x \), \( \lambda(x) \neq 0 \). We proceed by case analysis on the properties of \( x \).

Suppose that either \( |x|_1 = 0 \) (or similarly that \( |x|_{-1} = 0 \)). As \( x \) is not in \( 0^* \), we have that \( |x|_{-1} > 0 \) and thus for every \( y \) so that \( x \preceq y \), we also have \( |y|_{-1} > 0 \). So it cannot be the case that \( \lambda(y) = -n - 1 \) and therefore \( \lambda(x) + \lambda(y) \neq 0 \).

Suppose that both \( |x|_1 > 0 \) and \( |x|_{-1} > 0 \) and \( \text{size}(x) > n \). In that case, we have that \( |\lambda(x)| > n + 1 \). Take \( y \) verifying \( x \preceq y \). Suppose that \( \text{size}(y) \leq n \), we have that \( |\lambda(y)| \leq n \) and therefore \( \lambda(x) + \lambda(y) \neq 0 \). Now in case \( \text{size}(y) > n \) the conclusion follows from the use of Lemma [6] and Lemma [7]. Indeed, as from Lemma [6], \( \text{size}(\cdot) \) and \( \text{ind}(\cdot) \) are antitone, the only means for \( \lambda(x) + \lambda(y) \) to be equal to 0 is that \( \text{size}(x) = \text{size}(y) \) and \( \text{ind}(x) = \text{ind}(y) \). But in that case Lemma [7] tells us that \( \text{sign}(x) = \text{sign}(y) \) and so \( \lambda(x) = \lambda(y) \). Therefore \( \lambda(x) + \lambda(y) \neq 0 \).
The last case is when \(|x|_1 > 0, |x|_{-1} > 0\) and \(\text{size}(x) \leq n\). Take \(y\) so that \(x \preceq y\), now it must be the case that \(\text{size}(y) \leq \text{size}(x)\) by Lemma 8. So, \(\lambda(y) = \text{unb}(y)\), and as, from hypothesis, \(\lambda(y) \neq 0\), we have that \(\text{unb}(y) = \text{unb}(y)\) where \(j = \min(U_y)\). Similarly we have that \(\lambda(x) = \text{unb}(x) = \text{unb}_i(x)i\) where \(i = \min(U_x)\). As \(x \preceq y\), it must be the case that \(U_x \subseteq U_y\). Indeed, by definition, for every \(l\) in \([1,n]\), \(\text{unb}_l(x) \neq 0\) implies that \(\text{unb}_l(y) \neq 0\). Moreover, the definition also implies that \(\text{unb}_i(x) = \text{unb}_i(y)\). In particular \(\text{unb}_i(x) = \text{unb}_i(y)\). So, if \(\min(U_y) = \min(U_x) = i\), then \(\lambda(y) = \lambda(x)\), so \(\lambda(x) + \lambda(y) \neq 0\). If \(\min(U_y) \neq \min(U_x)\), as \(U_x \subseteq U_y\), it must be the case that \(\min(U_y) < \min(U_x)\) and in that case \(|\lambda(x)| > |\lambda(y)|\) so that \(\lambda(x) + \lambda(y) \neq 0\).

**Lemma 9** There is \(x\) in \(\mathbb{B}^m\) so that \(\lambda(x) = 0\).

**Proof**
From the previous lemma, we know that there is \(x\) in \(\mathbb{O}^m\) so that \(\lambda(x) = 0\). We construct \(y\) so that \(x \preceq y\), \(y \in \mathbb{B}^m\) and \(\lambda(y) = 0\). As \(\lambda(x) = 0\), it must be the case that, \(|x|_1 > 0, |x|_{-1} > 0\) and \(\text{size}(x) \leq n\). So for every \(y\) in \(\mathbb{B}^m\) so that \(x \preceq y\) it must be the case that we also have \(|y|_1 > 0, |y|_{-1} > 0\) and \(\text{size}(y) \leq n\). From the fact that \(\lambda(x) = 0\), we know that \(U_x = \emptyset\) and that for every \(i\), \(\text{unb}_i(x) = 0\). In particular, this means that there is \(z_i\) in \(\mathbb{B}^m\) so that \(x \preceq z_i\) and \(E_{1,i}(z_i) = 0 = E_{-1,i}(z_i)\). Now if we turn every occurrence of 0 in \(x\) that is in front of an \(a_i\) or a \(b_i\) in \(s\) to the same value as in \(z\), we obtain \(x_i\) so that \(\lambda(x_i) = 0\). By iteratively doing this construction for every \(i\) we obtain \(y\) in \(\mathbb{B}^m\) so that \(\lambda(y) = 0\).

This last lemma actually finishes the proof Lemma 8.

**References**


