THE HACKBUSCH CONJECTURE ON TENSOR FORMATS — PART TWO

WERONIKA BUCZYŃSKA

Abstract. We prove a conjecture of W. Hackbusch in a bigger generality than in our previous article. Here we consider Tensor Train (TT) model with an arbitrary number of leaves and a corresponding "almost binary tree" for Hierarchical Tucker (HT) model, i.e. the deepest tree with the same number of leaves. Our main result is an algorithm that computes the flattening rank of a generic tensor in a Tensor Network State (TNS) model on a given tree with respect to any flattening coming from combinatorics of the space. The methods also imply that the tensor rank (which is also called CP-rank) of most tensors in a TNS model grows exponentially with the growth of the number of leaves for any shape of the tree.

1. Introduction.

In this article we study the *variety of tensor network states* $\mathrm{TNS}(\mathfrak{T}, f) \subset V_1 \otimes \cdots \otimes V_n$ for a given tree, a function on its edges, vector spaces V_i assigned to the leaves following [4].

Our goal is to build some tools that help to compare tensor network spaces with each other. We compute the maximum possible flattening rank of a tensor in a given TNS model with respect to a fixed subset of leaves which encodes the flattening. The central case of this paper is a version Hackbusch conjecture: two models are both defined by a constant function on trees with the same number of leaves one is Train Track model and the other is Hierarchical model on a *almost perfect* binary tree.

We also obtain a bound and an algorithm calculating the maximum flattening rank of a tensor with respect to a subset of leaves for a non-constant function defining the TNS space.

The main result of [2] is an exact bound for a flattening rank of a tensor. It is obtained from flattenings that are divisions of the initial flattening. Our upper bound is a special case of this result.

In numerical tensor analysis it is an important problem to know how fast can the dimension of flattenings of tensors grow with the size of the tree. The main result of [3] is a bound exponential in the number of leaves for HT model, that is on almost perfect binary tree. In the paper [5] Theorem 1 says that such a bound holds for Train Track model. With our technique we obtain exponential lower bound for tensors in a TNS defined by any binary tree and constatnt function.

2. Notation and the vertex definition.

For a set S, its size is $|S|$. A brief notation for the set $\{1, \ldots, j\}$ is $[j]$.

Given a tree $\mathfrak{T}$, we have the set of vertices $\mathcal{V}(\mathfrak{T})$, the set of edges $\mathcal{E}(\mathfrak{T})$, the set of leaves $\mathcal{L}(\mathfrak{T})$. When the tree is clear, we omit it. Let $e \in \mathcal{E}$ be an edge of the tree. Removing e from the tree yields two trees – we say one is to the left of e, the other to the right. By

Date: 1 February 2018.

W. Buczyńska is supported by Polish National Science Center (NCN), project 2013/11/D/ST1/02580.

$\overleftarrow{e}$ we denote the set of leaves of the tree to the left of the edge e. If $v \in \mathcal{V}$ is a vertex, then $\downarrow v$ is the set of leaves of which v is an ancestor.

Definition 2.1. Given a binary tree $\mathfrak{T}$ with n leaves, we pick a vector space V_i for each leaf. We also fix an integer-valued function $f : \mathcal{V}(\mathfrak{T}) \to \mathbb{N}$ on the vertices of the tree. We define the *variety of tensor network states* $\mathrm{TNS}(\mathfrak{T}, f) \subset V_1 \otimes \cdots \otimes V_n$ in the following way: $t \in \mathrm{TNS}(\mathfrak{T}, f)$ if and only if there exist linear subspaces U_v of dimension at most $f(v)$, such that:

- $U_i \subset V_i$, if $v = i$ is one of the leaves,
- $U_v \subset U_{v_1} \otimes U_{v_2}$ whenever v is not a leaf and v_1 and v_2 are its children,
- $t \in U_v$, if v is the root of the tree.

3. The edge definition of TNS.

The vertex definition of the Tensor Network Space encodes the space by a tree, a natural-valued function on vertices, and vector spaces on leaves. We rewrite this definition: the function has the same values as before assigned to the edges instead of vertices. Also, we remove the root and the value assigned to it. In our previous article Proposition 2.6 of [1], we explain that in this way we define the same variety:

Proposition 3.1. *Let f, $\mathfrak{T}$ and the order of leaves be as in Definition 2.1. The variety $\mathrm{TNS}(\mathfrak{T}, f)$ is the locus of tensors $t \in V_1 \otimes \cdots \otimes V_n$, such that for any vertex $v \in \mathfrak{V}$ we have:*

$$\dim \left(\left(\bigotimes_{l \in \{\downarrow v\}} V_l \right)^* \llcorner t \right) \leq f(v).$$

3.1. The edge definition versus vertex definition of TNS.

We rewrite the definition of the $\mathrm{TNS}(\mathfrak{T}, f)$. Given a tree $\mathfrak{T}$ and a function f on vertices, we construct function g from edges to the natural numbers. Let $v_s(e)$ and $v_f(e)$ be the two ends of the edge e so that $v_f(e)$ is the father of $v_s(e)$. We set $g(e) := f(v_s(e))$.

Moreover, we remove the root v_r of the tree. The two edges e_1 and e_2 adjacent to it become one edge e_r. The value $g(e_r) := \min(g(e_1), g(e_2)) = \min(f(v_s(e_1)), f(v_s(e_2))$ equal to the minimum of the values assigned to the two old edges or equivalently two sons of the root.

This does not change the $\mathrm{TNS}(\mathfrak{T}, f)$, since the value at the root was irrelevant anyway — see Proposition 3.1.

Definition 3.2. Let $\mathfrak{T}$ be a tree and $f : \mathcal{E}(\mathfrak{T}) \to \mathbb{N}$ a natural valued function on the set of edges of the tree $\mathfrak{T}$. Then $\mathrm{TNS}(\mathfrak{T}, f)$ is the set of tensors $t \in V_1 \otimes \cdots \otimes V_n$ such that

$$\dim \left(\left(\bigotimes_{l \in \{\overleftarrow{e}\}} V_l \right)^* \llcorner t \right) \leq f(v).$$

Lemma 3.3. *Given a tensor $t \in V_1 \otimes \ldots \otimes V_n$ and a subset $A \subset \mathfrak{L}$ of the leaves it is not important if we hook t in $\mathcal{A}$ or its complement*

$$\dim \left(\otimes_{l \in A} V_l\right)^* \llcorner t = \dim \left(\otimes_{l \notin A} V_l\right)^* \llcorner t$$

Proof. This follows from the properties of the rank – the rank of a matrix and its transpose is the same. □

Remark 3.4. In the edge definition of the TNS, as we said before, we do not have the root of the tree. But we can place the root on any edge we like and go back to the vertex definition.

Definition 3.5. Given a tree $\mathfrak{T}$ and a subset $\mathcal{A} \subset \mathfrak{L}$ of the leaves of the tree, we define a **minimal monochromatic cut** as a minimal set of edges, such that each tree in the forest obtained by removing those edges from the initial tree has all leaves either in the set $\mathcal{A}$ or in its complement. We denote by **MinMonoCuts** $(\mathfrak{T}, \mathcal{A})$ the set of all minimal monochromatic cuts. By

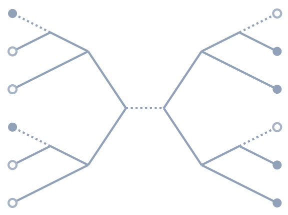
$$\text{MonoSize}\,|\mathfrak{T}, \mathcal{A}|$$

we denote the size of a minimal monochromatic cut.

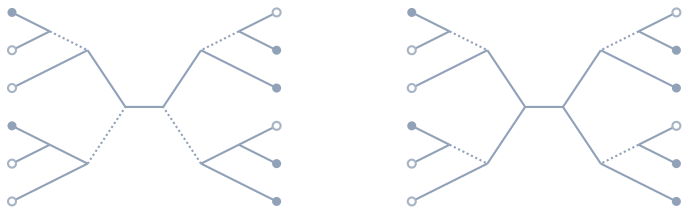
Definition 3.6. Given a tree $\mathfrak{T}$ and a subset of $\mathcal{A} \subset \mathfrak{L}$ of the leaves of the tree, we define **maximal colour cut** as a maximal set of edges, such that neither of the trees in the forest obtained by removing those edges from the initial tree has all the leaves in $\mathcal{A}$ or in its complement. We denote by **MaxColorCuts** $(\mathfrak{T}, \mathcal{A})$ the set of all maximal colour cuts.

Remark 3.7. Neither minimal monochromatic cut or maximal colour cut are unique for a given tree and a subset of leaves.

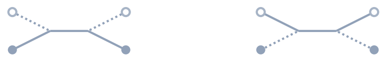
Example 3.8. The following tree with 12 leaves and the dark/white division of leaves has a unique monochromatic cut:



The colour cut in this case is not unique:



A simple example with a non-unique monochromatic cut:



Proposition 3.9. *Let $\mathfrak{T}$ be a tree with a subset of leaves $\mathcal{A}$. Let $\mathcal{M} \in$* **MinMonoCuts** $(\mathfrak{T}, \mathcal{A})$ *be a minimal monochromatic cut and $\mathcal{C} \in$* **MaxColorCuts** $(\mathfrak{T}, \mathcal{A})$ *a maximal colour cut, then*

$$|\mathcal{M}| = |\mathcal{C}| + 1.$$

For consistency, if **MaxColorCuts** $(\mathfrak{T}, A)$ *is empty, we replace $|\mathcal{C}|$ in the above formula by -1.*

Proof. We prove two inequalities. First we remove all the edges of $\mathcal{C}$ from the tree $\mathfrak{T}$ to obtain a forest of $|\mathcal{C}| + 1$ trees. Each tree has some leaves in the set $\mathcal{A}$ and some outside of it. Therefore, each tree must contain an element of $\mathcal{M}$ – our minimal mono cut. This proves that

$$|\mathcal{M}| \geqslant |\mathcal{C}| + 1.$$

For the other inequality, we use induction on the size of the tree $\mathfrak{T}$ and the size of the set $\mathcal{A}$. Let us choose a minimal monochromatic cut $\mathcal{M}$.

If the set $\mathcal{A}$ or its complement are empty, we stop here: there is no maximal colour cut, so the right side is 0. There is exactly one minimal monochromatic cut $\mathcal{M} = \emptyset$, so left side is also 0.

For the induction step, the set $\mathcal{A}$ and its complement are non-empty. We find a trivalent vertex v (not a leaf), such that the forest of three trees obtained by removing the vertex v consists of

- a tree $\mathfrak{T}_1$, attached to v by edge e_1, with all leaves in $\mathcal{A}$,
- a tree $\mathfrak{T}_2$, attached to v by edge e_2, with all leaves outside of $\mathcal{A}$,
- a tree $\mathfrak{T}_3$, attached to v by edge e_3.

Such a vertex exits: let e_3 be "an initial edge" in $\mathcal{C}$, that is removing e_3 from $\mathfrak{T}$, yields two trees, one with no edges in $\mathcal{C}$, the other is $\mathfrak{T}_3$.

The new smaller tree for the induction step is $\mathfrak{T}_3$ and v becomes its leaf. The new subset of leaves is $\mathcal{A}_3$ defined as

$$\mathcal{A}_3 = \begin{cases} \mathcal{A} \cap \mathcal{V}(\mathfrak{T}_3) & \text{if } e_1 \in \mathcal{M}, \\ \{v\} \cup (\mathcal{A} \cap \mathcal{V}(\mathfrak{T}_3)) & \text{if } e_2 \in \mathcal{M}. \end{cases}$$

This new leaf v is in $\mathcal{A}_3$ provided that the minimal cut contains edge e_2, and is outside of $\mathcal{A}_3$, if it contains e_1. We note that $\mathcal{M}_3 = \mathcal{M} \cap \mathcal{E}(\mathfrak{T}_3)$ is a minimal monochromatic cut for the tree $\mathfrak{T}_3$ and $\mathcal{C}_3 = \mathcal{C} \cap \mathcal{E}(\mathfrak{T}_3)$ is a maximal colour cut for $\mathfrak{T}_3$. Finally, $|\mathcal{C}_3| = |\mathcal{C}| - 1$ and $|\mathcal{M}_3| = |\mathcal{M}| - 1$, and by induction $|\mathcal{M}_3| \leqslant |\mathcal{C}_3| + 1$. This ends the proof.

□

Fact 3.10. *Let $t_1 \in V_1 = W_1 \otimes W_1'$ and $t_2 \in V_2 = W_2 \otimes W_2'$. Then*

$$(W_1 \otimes W_2)^* \llcorner (t_1 \otimes t_2) = W_1^* \llcorner t_1 \otimes W_2^* \llcorner t_2$$

Lemma 3.11 (Lemma 4.1, [1])**.** *Fix any subset $\mathcal{A} \subset \mathcal{L}(\mathfrak{T})$ of leaves of a tree $\mathfrak{T}$, and choose two disjoint subtrees $\mathfrak{T}'$ and $\mathfrak{T}''$ and set $\mathcal{A}' = \mathcal{A} \cap \mathcal{L}(\mathfrak{T}')$ and $\mathcal{A}'' = \mathcal{A} \cap \mathcal{L}(\mathfrak{T}'')$. Define $q' := \dim\left(\left(\bigotimes_{l \in A'} V_l\right)^* \llcorner t'\right)$ and $q'' := \dim\left(\left(\bigotimes_{l \in A''} V_l\right)^* \llcorner t''\right)$. Then there exists a tensor $t = t' \otimes t'' \in \mathrm{TNS}(\mathfrak{T}, r)$ such that*

$$\dim\left(\left(\bigotimes_{l \in A} V_l\right)^* \llcorner t\right) = q'q''.$$

3.2. **Optimal function.**

Definition 3.12. The function $f : \mathcal{E} \to \mathbb{N}$ on edges of the tree is *optimal* if for every edge $e \in \mathcal{E}(\mathfrak{T})$ the flattening rank of a generic tensor $t \in \mathrm{TNS}(\mathfrak{T}, f)$ at e is equal to $f(e)$:

$$\dim\left(\bigotimes_{l \in \overleftarrow{e}} V_l^* \llcorner t\right) = f(e).$$

Remark 3.13. The other way of saying the function f is optimal is that it is the smallest function that gives the variety in question. In particular, for every edge e the bound $f(e)$ is attained for a general tensor in the TNS, for the flattening associated to the edge in question.

The algorithm that transforms a function into an optimal one is described in the proof of Proposition 2.7 of [1]. From a given function we construct a function f', which is the optimal function and defines the same TNS as the function f.

Fact 3.14. *The constant function is optimal if its value is not bigger then the dimension of the vector spaces at the leaves of the tree.*

3.3. **An upper bound on the rank.**

Theorem 3.15. *Let $\mathfrak{T}$ be a tree, $\mathcal{L}$ its set of leaves, $f : \mathcal{E} \to \mathbb{N}$ a function defining a* TNS. *Let $\mathcal{A}$ be a subset of leaves of the tree $\mathfrak{T}$ and let $\mathcal{M}$ be a monochromatic cut, i.e. a subset of edges such that after removing them from the tree we get a forest of trees, each with all leaves either in $\mathcal{A}$ or $\mathcal{L}(\mathfrak{T}) \setminus \mathcal{A}$. Then the flattening rank of any tensor in* $\mathrm{TNS}(\mathfrak{T}, f)$ *with respect to $\mathcal{A}$ is not bigger then $\prod_{e\in\mathcal{M}} f(e)$.*

Proof. To prove the inequality, let $t \in \mathrm{TNS}(\mathfrak{T}, f)$ be a tensor and let $e \in \mathcal{M}$ be an initial edge of $\mathcal{M}$. By this we mean that removing e from $\mathfrak{T}$ yields two trees:

- a tree $\mathfrak{T}_1$ with all leaves either in or outside of $\mathcal{A}$ and
- a tree $\mathfrak{T}_2$ which is the rest of the tree.

Let us place the root of the tree $\mathfrak{T}$ on the edge e. We denote by U_1 and U_2 the vector spaces at the two ends of e — roots of respectively $\mathfrak{T}_1$ and $\mathfrak{T}_2$. Thus, by definition $t \in U_1 \otimes U_2$. So we write $t = \sum_{i=1}^{f(e)} \alpha_i \otimes \beta_i$ where $\alpha_i \in U_1$ is a basis of U_1 and $\beta_i \in U_2$ a basis of U_2.

Let us write $\mathcal{A}_1 = \mathcal{L}(\mathfrak{T}_1) \cap \mathcal{A}$ and $\mathcal{A}_2 = \mathcal{L}(\mathfrak{T}_2) \cap \mathcal{A}$, $\mathcal{A}^*$ for $\bigotimes_{l\in\mathcal{A}} V_l^*$, similarly $\mathcal{A}_1^*$ and $\mathcal{A}_2^*$.

We know $e \in \mathcal{M}$ and there are two cases:

First case is when $\mathcal{L}(\mathfrak{T}_1) \cap \mathcal{A} = \emptyset$. Then the flattening space of the tensor t with respect to $\mathcal{A}^*$ is contained in the algebraic sum of vector spaces:

$$\mathcal{A}^* \llcorner t \subset \sum_{i=1}^{r} \alpha_i \otimes (\mathcal{A}_2^* \llcorner \beta_i) \simeq \bigoplus_{i=1}^{r} (\mathcal{A}_2^* \llcorner \beta_i).$$

Denote by $\mathcal{M}_2 = \mathcal{M} \setminus \{e\}$. As $\beta_i \in \mathrm{TNS}(\mathfrak{T}_2)$, by induction we have

$$\dim(\mathcal{A}_2^* \llcorner \beta_i) \le \prod_{e\in\mathcal{M}_2} f(e).$$

Combining the above we get the required inequality, namely

$$\dim \mathcal{A}^* \llcorner t \le \prod_{e\in\mathcal{M}} f(e).$$

The second case is when $\mathcal{L}(\mathfrak{T}_1) \subset \mathcal{A}$. Since all leaves of $\mathfrak{T}_1$ are in $\mathcal{A}$, we have

$$\mathcal{A}^* \llcorner t \subset \sum_{i=1}^{r} (\mathcal{A}_1^* \llcorner \alpha_i) \otimes (\mathcal{A}_2^* \llcorner \beta_i) \subset \sum_{i=1}^{r} \mathbb{C} \otimes (\mathcal{A}_2^* \llcorner \beta_i) = \sum_{i=1}^{r} (\mathcal{A}_2^* \llcorner \beta_i).$$

Thus, as before

$$\dim \mathcal{A}^* \llcorner t \le \sum_{i=1}^{f(\varepsilon)} \prod_{e\in\mathcal{M}_2} f(e) = f(\varepsilon) \cdot \prod_{e\in\mathcal{M}_2} f(e) = \prod_{e\in\mathcal{M}} f(e).$$

□

3.4. **The rank for constant function.**

Theorem 3.16. *Let $\mathfrak{T}$ be a tree, $\mathcal{L}$ its set of leaves, $f : \mathcal{E} \to \mathbb{N}$ a constant function equal to r. Let $\mathcal{A}$ be a subset of leaves of the tree $\mathfrak{T}$. Then the flattening rank of a generic tensor in* $\mathrm{TNS}(\mathfrak{T}, r)$ *with respect to $\mathcal{A}$ equals* $r^{\mathrm{MonoSize}\,|\mathfrak{T},\mathcal{A}|}$.

Proof. We prove two inequalities. The upper bound for the rank is a special case of Theorem 3.15. For the lower bound we argue by induction on the size of the tree to construct a tensor with the required flattening rank.

Let $\mathcal{C}$ be a maximal colour cut of the tree $\mathfrak{T}$ with the set $\mathcal{A}$ and $\mathcal{M}$ be a minimal monochromatic cut for the same tree and set. Let v, e_1, e_2, e_3, $\mathfrak{T}_3$, $\mathcal{C}_3$, $\mathcal{M}_3$, $\mathcal{A}_3$, $\mathfrak{T}_1$, $\mathfrak{T}_2$ be as in the proof of Proposition 3.9. Let also $\mathcal{A}_1 = \mathcal{A} \cap \mathcal{L}(\mathfrak{T}_1)$ and $\mathcal{A}_2 = \mathcal{A} \cap \mathcal{L}(\mathfrak{T}_2)$.

To start the induction let $\mathfrak{T}$ be a tree with at most tree leaves, then any $\mathcal{M} \in$ **MinMonoCuts** $(\mathfrak{T}, \mathcal{A})$ has at most one element and the statement is straightforward.

Now let $\mathfrak{T}$ be a tree. By induction there exists a tensor $t_3 \in \mathrm{TNS}(\mathfrak{T}_3, r)$ with flattening rank $r^{|\mathcal{M}_3|}$ with respect to $\mathcal{A}_3$ as $\mathcal{M}_3$ is a minimal monochromatic cut for $\mathfrak{T}_3$ and subset of its leaves $\mathcal{A}_3$. Also, there exists a tensor in $t_{12} \in \mathrm{TNS}(\mathfrak{T}_1 \cup_{e_1-e_2} \mathfrak{T}_2, r)$ flattening rank r with respect to the set $\mathcal{A}_1 \cup \mathcal{A}_2$.

Now, by Lemma 3.11, there exists a tensor t in $\mathrm{TNS}(\mathfrak{T}, r)$, namely $t_{12} \otimes t_3$, such that $\dim(\bigotimes_{l \in \mathcal{A}} V_l^* \llcorner t) = r^{|\mathcal{M}|}$, since $|\mathcal{M}| = |\mathcal{M}_3| + 1$. As the flattening rank is semicontinuous, a generic tensor in $\mathrm{TNS}(\mathfrak{T}, r)$ has flattening rank at least $r^{|\mathcal{M}|}$. $\square$

Remark 3.17. To compute the rank of a general tensor in $\mathrm{TNS}(\mathfrak{T}, r)$ with the constant function equal to r, we can equally well compute it for $r = 2$. This is because the exponent is independent of r.

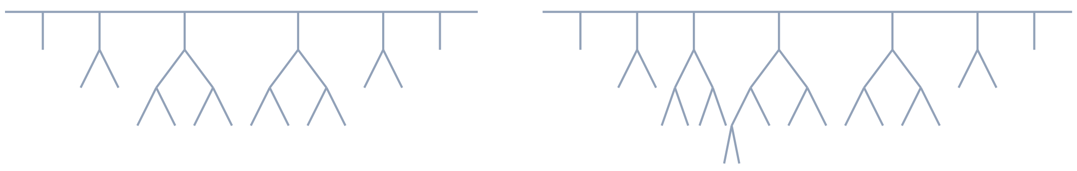
4. The train track and almost binary models compared.

Definition 4.1. We say that a binary tree is an **almost perfect binary tree** if it differs from a perfect binary tree only by removing the last leaves from the last row.

Let $\mathfrak{T}_n^{train}$ denote a train track tree with n leaves. Let $\mathfrak{T}_n^{bin}$ denote an almost binary tree with n leaves.

In [1] we proved a simple version of the Hackbusch conjecture, namely we compared a $\mathrm{TNS}(\mathfrak{T}_{2^q}^{train}, r_1)$ of a train track tree and $\mathrm{TNS}(\mathfrak{T}_{2^q}^{bin}, r_2)$ of a perfect binary tree with 2^q leaves. In this paper we extend this result a bit by allowing arbitrary number of leaves for both tree types.

In order to compare the tensor network spaces coming from a train track tree and an almost binary tree, both with a natural permutation of leaves (from left to right), we will draw the almost binary tree in a specific way. Namely,

(4.2) 

On the above picture both trees are almost perfect binary trees, one is perfect with 16 leaves, the other has nine new leaves in the new row and a total of 21 leaves. For a binary tree drawn as on the Figure (4.2), each subtree below a vertical edge is a *hanging subtree*.

Remark 4.3. Let $a_k = \sum_{i=0}^{i=k} 4^i$ for $k > 0$, and set $a_0 = 0$. This number can be interpreted combinatorially as the biggest number of leaves that the almost binary tree

$\mathfrak{T}$ has, if there exists a subset $\mathcal{A}$ of its leaves and $\mathrm{MonoSize}(\mathfrak{T}, \mathcal{A}) \leqslant k$. Other way of defining these numbers is to define a sequence of almost binary trees for each a_k. The first one is empty. The next has $a_1 = 5$ leaves. Having defined those trees up to k-th, the $k+1$ tree is the almost binary tree with the smallest number of leaves, such that it has the kth one as a hanging subtree.

Lemma 4.4. *Suppose n is in the set $\{a_{k-1}+1, \dots, a_k\}$ and the leaves of the almost binary tree with n leaves are labelled from left to right, when the tree is drawn as above. Then there exists $j \in \{1, \dots, n\}$ such that* $\mathrm{MonoSize}\,|\mathfrak{T}_n^{bin}, [j]| \geqslant k$.

Proof. We argue by induction on the size of the binary tree. First we check case by case $n \in \{1, \dots, 6\}$. Up to $n = a_1 = 5$ leaves of the subsets of leaves of type $[j]$ for some $j \in \{1, \dots, n\}$ are also of type $\overleftarrow{e}$, so there is nothing to prove and the number of cuts is 1. When we get to $n = 6 = a_1 + 1$, then we need one more cut — for the subset $\{1, 2, 3\}$ of the leaves. Now we want to prove the claim for the pair of trees with n leaves. Suppose we proved our claim for all $m < n$. Let us observe that the induced permutations on the hanging subtrees are natural. We distinguish two cases.

The first case is when all hanging subtrees of our binary tree with n leaves require at most $k-1$ cuts. Then for each subset of leaves coming from the train track model, that is for the sets of type $\{1, \dots, j\}$, the whole tree requires at most k cuts. Indeed, any minimal monochromatic cut induces a minimal monochromatic cut for the hanging subtree, which has at least $k-1$ elements. One more cut is needed in order to separate the hanging tree from the leaves with indices that are either greater or less than j.

The second case is when at least one of the *hanging* trees needs k cuts. We know that for $m < n$ the increase in the number of cuts needed occurs at each $m = a_l + 1$ for some $l \in \mathbb{N}$.

The smallest n in question for which this happens is $n = a_k + 1$. Then all the hanging trees are perfect binary trees except one called $\mathfrak{T}_\xi$, which has $a_{k-1}+1$ leaves and hangs from a vertical edge ξ — keep in mind our tree is almost perfect binary tree. If we look at the edge ξ, we see that it has two horizontal incident edges one to the left and one to the right, call them ξ_l and ξ_r respectively. By construction, the tree with the root equal to the left (respectively right) vertex of ξ_l (respectively ξ_r) is perfect binary with $a_{k-1} < 4^k < a_k$ leaves. Thus, both also need k cuts.

For all bigger trees, that is for $n > a_k + 1$, by the induction assumption there exists j such that to cut out the set $\{1, \dots, j\} \cap \mathcal{L}(\mathfrak{T}_\xi) = \{j', \dots, j\}$ or its complement in $\mathfrak{T}_\xi$, we need at least k cuts inside the tree $\mathfrak{T}_\xi$. As ξ is neither first or last vertical edge, one more cut outside the tree $\mathfrak{T}_\xi$ is needed.

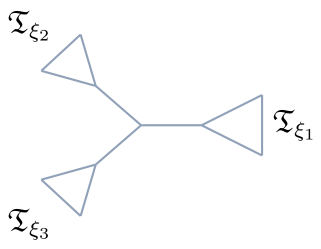
□

Lemma 4.5. *Suppose n is in the set $\{a_{k-1}+1, \dots, a_k\}$ and the leaves of the almost binary tree with n leaves are labelled by any permutation. Then there exists $j \in \{1, \dots, n\}$ such that* $\mathrm{MonoSize}\,|\mathfrak{T}_n^{bin}, [j]| \geqslant k$. *In other words, natural permutation always gives the smallest minimal monochromatic cut for any subset from the definition of $\mathfrak{T}_n^{train}$.*

Proof. Again we proceed by induction on the number of leaves. As the first induction step, for the number of leaves from 1 to 6 we check case by case that switching from natural permutation to any other permutation, the number of cuts can only increase.

For the induction step we will construct $\mathcal{M} \in \mathrm{MinMonoCuts}(\mathfrak{T}, \mathcal{A})$. We consider two situations. The first case is when at any vertex, at most two of the three trees that have a root at this vertex require k cuts, the other(s) at most $k-1$. Then, as in the proof of Lemma 4.4, the number of cuts required for the whole tree is at least k.

In the second case there exists a vertex, such that all three subtrees require k cuts. The smallest n for which this situation occurs for all permutations, is $n = a_k + 1$. To see this, use induction combined with Lemma 4.4.



We call those trees $\mathfrak{T}_{\xi_1}$, $\mathfrak{T}_{\xi_2}$, $\mathfrak{T}_{\xi_3}$. Since we work with almost perfect binary tree, this is true for all bigger n as well. We claim at least $k+1$ cuts are needed for the whole tree for a set $[j]$ for some j.

We increase j until we need k cuts inside one of the trees $\mathfrak{T}_{\xi_i}$ for $i_1 \in \{1,\dots,3\}$ for the first time. This guarantees the other two have some leaves outside the set $\{1,\dots,j\}$. If at least one has a leaf in this set, we are done. If not, we continue increasing j until a second of the trees, say $\mathfrak{T}_{\xi_{i_2}}$ needs k cuts. In this situation $\mathfrak{T}_{\xi_{i_1}}$ has some leaves in $\{1,\dots,j\}$ and $\mathfrak{T}_{\xi_{i_3}}$ has some leaves outside of it. This implies we need k cuts inside $\mathfrak{T}_{\xi_{i_2}}$ and at least one more outside of it, which concludes the proof. □

Theorem 4.6 (Hackbush conjecture)**.** *Let $n \in \{a_{k-1}+1,\dots,a_k\}$. Then*

$$HH(n,r) \subset TT(n,r^k)$$

when both underlying trees have the same order of leaves. On the other hand for any permutation of leaves

$$HH(n,r) \not\subseteq TT(n,r^k-1).$$

5. Models with non-constant function

Suppose now we have two trees $\mathfrak{T}_1$ and $\mathfrak{T}_2$. Let us fix a function on edges of the first tree. Then, using our methods, we can give bounds for the function on the edges of the second tree so that there is an inclusion of the TNS models.

Theorem 5.1. *Let* $\mathrm{TNS}(\mathfrak{T}_1, f)$ *and* $\mathrm{TNS}(\mathfrak{T}_2, g)$ *be two tensor network spaces with the same number of leaves and the same vector spaces associated to them. If*

$$\mathrm{TNS}(\mathfrak{T}_1, f) \subset \mathrm{TNS}(\mathfrak{T}_2, g)$$

then for any edge $\varepsilon \in \mathcal{E}(\mathfrak{T}_2)$

$$g(\varepsilon) \geqslant \prod_{e \in \mathcal{M}} f(e)$$

*where $\mathcal{M} \in$ **MinMonoCuts** $\left(\mathfrak{T}_2, \overleftarrow{\varepsilon}\right)$.*

Proof. The statement follows from Theorem 3.15 applied once for each edge of the tree $\mathfrak{T}_2$, with the tree $\mathfrak{T}_1$ and subset given by the edge. □

6. Exponential growth of the rank

Theorem 6.1 (Exponential growth of tensor rank). *Let* $\mathrm{TNS}(\mathfrak{T}, r)$ *be a tensor network space on a binary tree with* n *leaves and a constant function. Then the rank of a generic tensor in* $\mathrm{TNS}(\mathfrak{T}, r)$ *is at least* $r^{\lfloor \frac{n}{2} \rfloor}$*. In particular, the growth of the rank is at least exponential.*

Proof. We construct a subset $\mathcal{A} \subset \mathcal{L}$ of the leaves such than $\mathrm{ColorSize}(\mathfrak{T}, \mathcal{A}) \geqslant \lfloor \frac{n}{2} \rfloor$. Initially $\mathcal{A} = \emptyset$.

We say that an inner (edge or) vertex of the tree is *initial*, if it is (adjacent to) a leaf in a tree obtained from $\mathfrak{T}$ by removing all leaves and then removing vertices that have exactly two adjacent edges. Every initial vertex has two sons, which are leaves. We pick an initial edge (there will be always at least one) and we say that one son of the corresponding initial vertex is in $\mathcal{A}$ and the other is not in $\mathcal{A}$. At each step we cut an edge removing two leaves from the initial tree.

Now it is enough to use Theorem 3.16 with the constructed set.

□

References

[1] Weronika Buczyńska, Jarosław Buczyński, and Michałek Mateusz. The Hackbusch conjecture on tensor formats. *J. Math. Pures Appl. (9)*, 104(4):749–761, 2015.

[2] Enrico Carlini and Johannes Kleppe. Ranks derived from multilinear maps. *J. Pure Appl. Algebra*, 215(8):1999–2004, 2011.

[3] Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. *JMLR: Workshop and Conference Proceedings*, 49, 2016.

[4] Wolfgang Hackbusch. *Tensor spaces and numerical tensor calculus*, volume 42 of *Springer Series in Computational Mathematics*. Springer, Heidelberg, 2012.

[5] Ivan Oseledets Valentin Khrulkov, Alexander Novikov. Expressive power of recurrent neural networks. arXiv:1711.00811 [cs.LG], 2017.

Weronika Buczyńska, Departement of Mathematics, Mechanics and Computere Science, ul. Banacha 2, 02-097 Warszawa, Poland

E-mail address: `wkrych@mimuw.edu.pl`