CS 860 Algebraic Complexity Theory May 06, 2024

Lecture 5 : Universal Circuits

Instructor: Rafael Oliveira *Scribe: Setareh Alagheh Band*

Remark These notes have not been subjected to the scrutiny of a journal publication, and it may contain errors. If you find one, please contact the instructor.

1 Introduction

In the same way that in Boolean complexity theory, we have the notion of universal Turing machines, in algebraic complexity theory, we have the notion of universal circuits. Universal circuits are circuits that can simulate any circuit of a given size. We now give a formal definition of universal circuits.

2 Universal Circuits

Definition 1 (Universal Circuits)**.** *A circuit Φ of deg ckt is said to be (n, s, d)-**universal** for circuits of input size n, output size n, degree d, and size s if for any n forms $f_1(x_1, ..., x_n), ..., f_n(x_1, ..., x_n)$ of degree at most d that can be simultaneously computed by a circuit of size at most s, then there is a circuit Ψ computing $f_1, ..., f_n$ such that the computation graph of Ψ is exactly that of Φ.*

*In other words, we say Ψ is a **projection** of Φ.*

We use circuits instead of formulas because in circuits we have the power of reusing the variables and the results of gates as mentioned before.

The definition mean for computing ckt Ψ there is no need to go very further from s and d, there is a universal circuit of size s that can compute it.

By the results above on efficient homogenization, it is enough to construct universal circuits for homogeneous circuits computing forms. In order to prove the existence of efficient universal circuits, it is good to first put a bit more structure on the circuits we are considering. This will be done by considering circuits in normal-homogeneous form.

Theorem 2 (Universal Circuits)**.** *For any integers $s \geq n \geq 1$ and $d \geq 1$, we can construct in poly(s, d) time a circuit Φ in normal-homogeneous form of size $O(d^3 s)$ nodes that is (n,s,d)-universal.*

We use the theorem below:

Theorem 3 (Homogenization)**.** *Let $f(x_1, \ldots, x_n)$ be a polynomial of degree d that can be computed by a circuit Φ of size s. Then, for any $r \leq d$ there is a circuit Ψ of size $O(r^2 s)$ that computes $H_0[f], H_1[f], \ldots, H_r[f]$.*

So if we have the circuit of size s degree d let's just convert it to a circuit size $O(d^2 s)$ that computes all the homogeneous components. So we can now have a sense, why we have the term homogeneous.

Definition 2 (Normal-Homogeneous Form): A homogeneous circuit Ψ is said to be in *normal-homogeneous form* if the following holds:

1. All inputs are labeled by a variable

2. All edges leaving an input gate are connected to sum gates

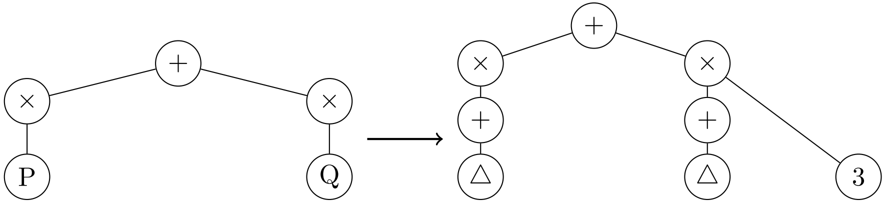
3. Output gates are sum gates

4. Non-input gates are alternating: that is, a product gate is connected to a sum gate, and a sum gate is connected to a product gate.

5. The fan-in of each product gate is exactly 2. (we do not restrict fan-in of sum gates)

6. The fan-out of each addition gate is at most 1.

Now we want to prove that the universal circuit exists at first place so we want to compute the first graph through universal circuit through second graph as shown to proof the universal circuit exists (each triangle is a universal circuit) :



As we can see we don't need any constant because we can replace any constant by a variable. But it seems that P and Q are coming from the same universal circuit model, but the construction of universal circuit is more complicated and that's where we have multiple inputs and multiple outputs to connect them to each other, for example P and Q can reuse more things.

For any homogeneous ckt Ψ of size s $\exists$ circuit Φ in normal homogeneous form of size $O(s)$ computing all gates of Ψ after projection.

It means every polynomial computed by Ψ is going to be computed by Φ. The proof is going to be just constructing a circuit by layers. So now we are constructing Φ out of Ψ

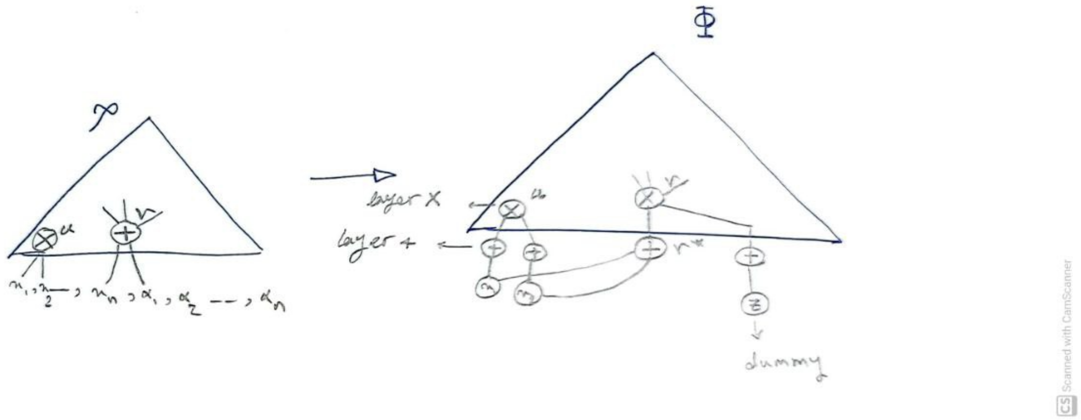

Ψ
u
v
$x_1, x_2, \ldots, x_n, \alpha_1, \alpha_2, \ldots, \alpha_n$
Φ
layer ×
layer +
u
v
v*
dummy

Figure 1: constructing Φ out of Ψ

We have input variables and field elements as inputs, every gate is computing its homogeneous. Now we build first layer, for example we have a time gate between x_1 and x_2. As shown we can have first layer for sum and second layer for products in Φ

We can construct any gates that we have in Ψ. From the first condition, we replace the field elements with variables to satisfy the condition 1. We can solve condition 5 by taking the children of the multiplication that we want as an inputs. As shown we know the trick to overcome condition 4. So the only bottleneck here is the condition 6, let's see we can fix that.

If we have v which is an addition and we use it bunch of times in Ψ, as we can see in Φ we can send it to a multiplication gate to reuse it. As fan-in of each times gate is 2 we can assign a dummy variable to second fan-in of multiplication. We can always substitute the dummy variable z with 1. So for every gate we can connect it to a multiplication gate which is multi-fan-out

So now we are ready to construct our universal circuit. So we now see how we can prove this theorem:

$\Psi^{(\text{in})} \to \Psi(\text{hom.}) \to \hat{\Psi}$ (normalhom.)

size: s $\to$ O(d^2s) $\to$ O(d^2s)

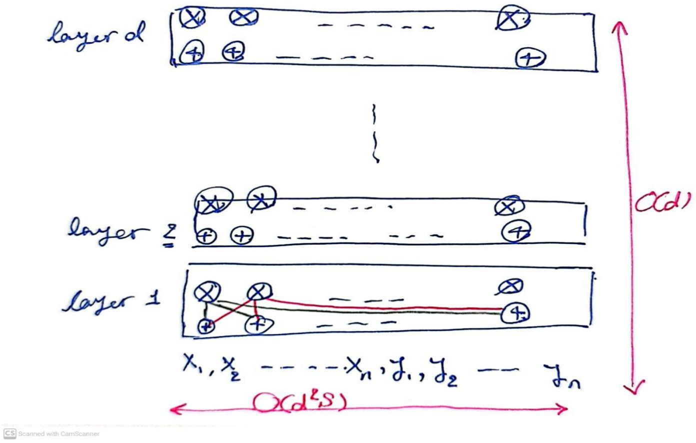


Figure 2: shape of the layers

We have layer 1 up to layer d, and on the output we have sum gates at the end. We have the layers alternating, so it means we have sums at first and multiplications then and after that sums again. Any circuit of each layer is a normal homogeneous circuit of size d^2s. As we have d layers of width $O(d^2s)$, the universal circuit of normal homogeneous circuit is size of $O(d^2s)$. Also we have one more layer of inputs and we have $O(d^2s)$ variables. Additions don't have any limitation so every addition is connected to everybody through edges that have a gadget. For each of these gadgets we are adding another helper variable. each edge is going to have a helper boolean variable in its gadget. If it is one we are activating the edge, if it's zero it's deactivated. We can see the gadget as shown below.

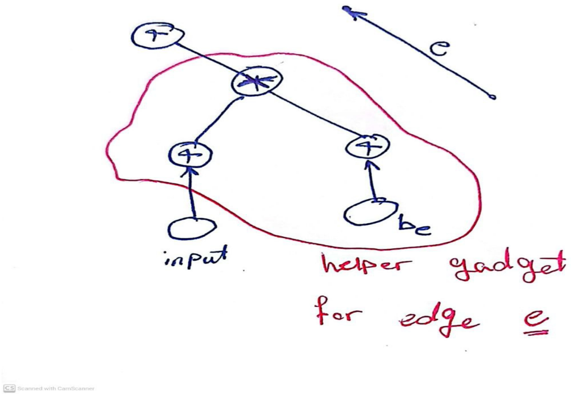


Figure 3: helper gadget

As shown, for every possible edge between a sum gate and a product gate, we put a little gadget, involving a new "help-variable", such that by setting the help-variable to 0 we essentially "erase" the edge and by setting it to 1 we keep the edge; therefore, we basically cover all possible circuits of normal-homogeneous-form (of a certain size and degree) in this way. It is clear that the circuit that was obtained in the first step can be computed by making the appropriate assignment to the variables of this circuit.

We construct the linking edges using the gadget for each plus.Then we connect each time gate to all the pluses in the bottom. So we construct the universal circuit, with the size of $O(d^3 s)$. And here our circuit only is homogeneous, and if every edge has a label we can go from the homogeneous circuit to normal homogeneous circuit through label gadget to set right edges to one with the helper variables.[AS10]

3 Computing First-Order Partial Derivatives

We will now prove the following seminal result in algebraic complexity theory, due to Baur and Strassen, which is also known as backpropagation in Machine Learning.

Theorem 4 (Baur-Strassen)**.** *Let $f(x_1, \ldots, x_n)$ be a polynomial that can be computed by an algebraic circuit of size s and depth Δ. Then, f and the first-order partial derivatives of f can be computed by an algebraic circuit of size $5s$ and depth $O(\Delta)$.*

This result is very interesting as a prior one might expect that the circuit complexity of $\partial_{x_1}(f), \ldots, \partial_{x_n}(f)$ will be of order $n \cdot s$ rather than $O(s)$, as these are n polynomials that are, intuitively, "as complex" as f is.

We are now going to prove this theorem. We start by defining the notion of partial derivatives. The partial derivative with respect to a variable x is defined as follows: $\partial_x(x) = 1$, and for a polynomial f that does not contain x, $\partial_x(f) = 0$. To define $\partial_x(f)$ for a general polynomial, we use the two identities $\partial_x(f + g) = \partial_x(f) + \partial_x(g)$ and $\partial_x(fg) = \partial_x(f)g + f\partial_x(g)$. It is not difficult to prove that this is a well-defined notion that makes sense over any field. Note, however, that while the definition is the same for all fields, a partial derivative can behave differently over different fields. For example, over the real numbers $\partial_x(x^2) = 2x$, while over the field with two elements $\partial_x(x^2) = 0$. A useful property of this definition is that it satisfies the chain rule for partial derivatives.

$$\frac{\partial}{\partial x_i} f(g_1, g_2, \ldots, g_n) = \sum_{j=1}^{n} \frac{\partial f}{\partial g_j} \cdot \frac{\partial g_j}{\partial x_i}$$

Here in below we have the proof of the theorem by induction.

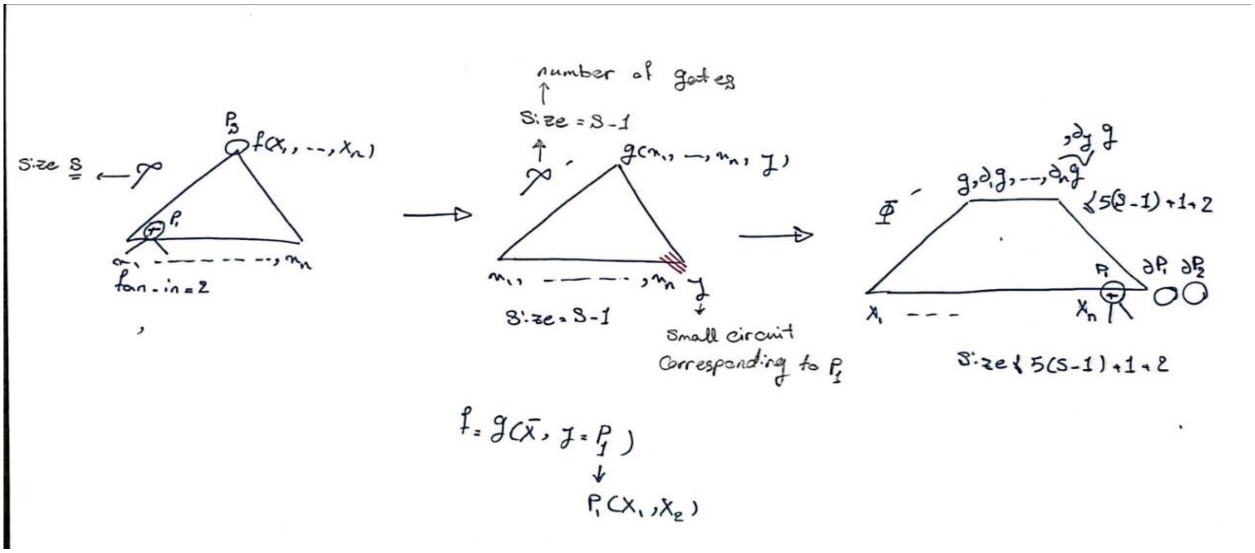


Figure 4: shape of constructing Φ'

Proof. We prove the existence of Ψ by induction on s, computing derivatives from the root down (in fact, we describe how to construct Φ given Ψ). If Ψ is an input gate, constructing Ψ is straightforward. Otherwise, let v be the deepest gate in Ψ (i.e., the gate that is the farthest from the output gate) and denote its children, which are input gates, by u, w. Consider the circuit $\Psi_{v=y}$, and denote its output polynomial by $f_{v=y}$. As we deleted two edges, $\Psi_{v=y}$ is smaller than Ψ. By induction, there exists a circuit Ψ' computing $\partial_{x_1}(f_{v=y}), \ldots, \partial_{x_n}(f_{v=y}), \partial_y(f_{v=y})$ of size $O(s-1)$. Denote by X' the set of variables that label either u or w in Φ (X' could be empty). Note that $f = f_{v=y}\big|_{y=f_v}$, where f_v is the polynomial that v computes in Ψ. The chain rule for partial derivatives implies that for every x_i,

$$\partial_{x_i}(f) = \partial_{x_i}(f_{v=y})\big|_{y=f_v} + \partial_y(f_{v=y})\big|_{y=f_v} \cdot \partial_{x_i}(f_v).$$

Therefore, for every $x_i \notin X'$, $\partial_{x_i}(f) = \partial_{x_i}(f_{v=y})\big|_{y=f_v}$. Since $\partial_{x_i}(f_v)$ is either a variable or a field element, and since the size of X' is at most two, we can compute $\{\partial_{x_i}(f)\}_{x_i \in X'}$ by adding at most a constant number of gates and using the gates in Ψ'. The size of Ψ is thus at most $O(s-1) + O(1) = O(s)$.

So as described we have circuit Ψ' which there, the gate P_1 is replaced by another input y so there is a reduction in the size of Ψ' and its size is $O(s-1)$ because that one gate has been removed. So from this circuit Ψ' we can build the circuit Φ' which has the size at most $5(s-1)$ and it computes all the partial derivatives of the polynomial f and also it computes $\partial_y(g)$.

So if we can compute from circuit size s, g() by another circuit of size $s-1$, then we replace y with p_1 in Ψ' which increases its size by one which will be 5(s-1)+1 and P_1 is made of 2 variables so we have 2 other gates and the size will be 5(s-1)+1+2. As seen P_1 depends only on 2 variables so this whole thing below is non-zero only for 2 variables.

$$\begin{aligned}\partial_i f = \quad & \partial_i g(\bar{X}, P_1) \\ & \implies \partial_j \left(\sum_{i=1}^{n} g_i(\bar{X}, P_1) \partial_i x_j \right) + (\partial_j g)\,(\bar{X}, P_1) \partial_i P_1 \\ & = \partial_i g(\bar{X}, P_1) + (\partial_i) P_1 (\partial_j) g(\bar{X}, P_1)\end{aligned}$$

References

[AS10] Amir Ye. Amir Sh. *Arithmetic Circuits: a survey of recent results and open questions*. Technion-Israel Institute of Technology, 2010.