The Tiling Method

Max von Hippel

March 23, 2021

Consider a two-party communication problem, in which the participants

(a) Alice

and

(b) Bob

participate to compute a function:

$$f : \underbrace{\mathbb{B}^n}_{\text{Alice's input}} \times \underbrace{\mathbb{B}^n}_{\text{Bob's input}} \to \underbrace{\mathbb{B}}_{\text{global output}}$$

The players can come up with a *protocol* $\Pi = (p_1, \ldots, p_t)$, namely, for some natural $t \in \mathbb{N}$, a sequence of t-many functions $p_i : \mathbb{B}^* \to \mathbb{B}^*$ such that the communication between the players looks like this:

Alice is given input x.

Hi Bob. I'm not divulging x, but, $p_1(x) = \texttt{P1}$.

Bob is given input y.

Thanks Alice. I'm not divulging y, but, $p_2(y, \texttt{P1}) = \texttt{P2}$.

Thanks Bob. Don't tell anyone, but: $p_3(x, \texttt{P1}, \texttt{P2}) = \texttt{P3}$.

Is that so? Well, $p_4(y, \texttt{P1}, \texttt{P2}, \texttt{P3}) = \texttt{P4}$.

Wicked. In that case, $p_5(x, \texttt{P1}, \texttt{P2}, \texttt{P3}, \texttt{P4}) = \texttt{P5}$.

...yada yada yada...

In that case, the last thing you need to know is that $p_t(x, \texttt{P1}, \texttt{P2}, \texttt{P3}, \texttt{P4}, \ldots, \texttt{P(t-1)}) = \texttt{Pt}$.

Suppose that there is a protocol Π for f consisting of t messages, but, there does not exist any protocol Π' for f consisting of fewer than t messages. Then we say t is the *communication complexity* of f, and we write $C(f) = t$.

Given some such function f, it would be nice if we could automatically compute a reasonable lower bound on its communication complexity. One way to do this is with the *tiling method*. We will give the method immediately, and in tandem, we will illustrate the method using the function $f(x,y) = x < y$ where x, y are integers in $\{0,1,2,3\}$, encoded in $\mathbb{B}$oolean. First, let $M(f)$ be the *matrix of f*, namely, the $2^n \times 2^n$ matrix whose (x,y)th entry is the value $f(x,y)$.

	`000`	`001`	`010`	`011`	`100`
`000` = 0	0	1	1	1	1
`001` = 1	0	0	1	1	1
`010` = 2	0	0	0	1	1
`011` = 3	0	0	0	0	1
`100` = 4	0	0	0	0	0

Table 1: The matrix $M(<)$ for inputs $x, y \in \{0,1,2,3\}$. Values of x are given in the rows, while values of y are given in the columns. False (i.e. 0) values are marked red for clarity.

A *combinatorial rectangle* in $M(f)$ is any submatrix of M. We say a rectangle $A \times B$ in $M(f)$ is *monochromatic* if for all x, x' in A and y, y' in B, $M_{x,y} = M_{x',y'}$.

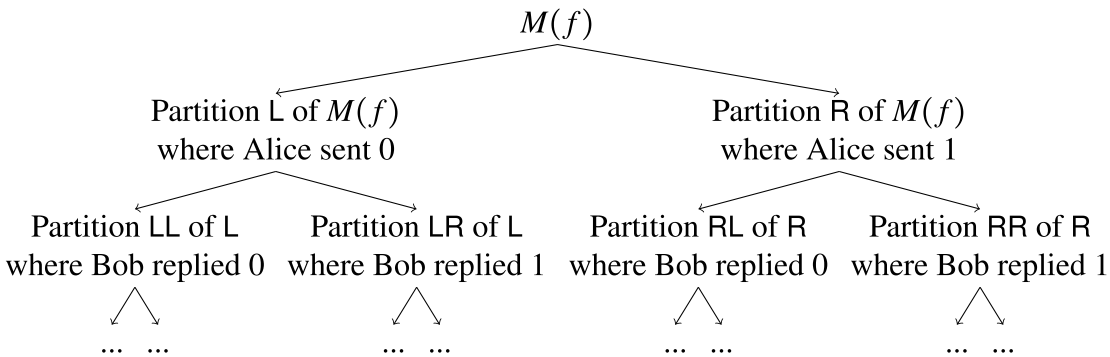
	0	1	2	3	4
0	0	1	1	1	1
1	0	0	1	1	1
2	0	0	0	1	1
3	0	0	0	0	1
4	0	0	0	0	0

	0	1	2	3	4
0	0	1	1	1	1
1	0	0	1	1	1
2	0	0	0	1	1
3	0	0	0	0	1
4	0	0	0	0	0

	0	1	2	3	4
0	0	1	1	1	1
1	0	0	1	1	1
2	0	0	0	1	1
3	0	0	0	0	1
4	0	0	0	0	0

Figure 2: Some example rectangles of $M(<)$. The first rectange, in purple, is monochromatically colored 0. The second rectangle, in orange, is illustrates the flexibility of our rectangle definition, namely, that the rectangle does not actually need to be connected in the original matrix (although, the entries cannot be permuted). Neither the second nor third rectangle is monochromatic.

Without loss of generality, suppose the protocol Π begins with Alice sending a bit b. Then *certainly* $M(f)$ partitions into two rectangles L and R, where L considers all the scenarios where the bit Alice sent was 0, and R considers all the scenarios where the bit Alice sent was 1. Notice that L and R are strictly smaller than $M(f)$; in fact, the number of cells in L plus the number of cells in R equals the number of cells in $M(f)$. This is what we mean by a *partition*.



Consider the L branch. The case of the original protocol where Alice starts by sending a 0 is precisely the smaller protocol in which Bob makes the first move, rooted in L. In other words, we have an inductive structure, where every step in a protocol Π yields a new, smaller protocol. Every step down the tree further disjointly partitions the space of possible values $f(x,y)$. Since there were finitely many such values $f(x,y) \in M(f)$ in the first place, clearly every such walk must eventually end in a *leaf* of the tree. In fact, each leaf is monochromatic.

Proof. For a contradiction assume some leaf J is not monochromatic. That is, J has entries i and j such that $i \neq j$. Since $i \neq j$ we know that they are in different cells. WLOG suppose i and j occur in different rows. Then the row of i or j refers to a bit of information about x which Alice did not yet share with Bob. (We know it's of x as it's a row; we know she did not share it yet as otherwise the subsequent partitioning would seperate i and j.) But this implies that we are not at a leaf yet, since Alice could send another bit of information and depending on that bit, partition the set of possible values $f(x,y)$ disjointly, into a set containing i and another containing j. So we have reached a contradiction since we assumed J was a leaf, and we are done. $\square$

We have just considered one protocol, Π, namely the protocol where Alice and Bob take turns reading aloud the bits of their input from least-significant to most-significant. And we've seen that this induces a natural upper bound on the communication complexity of f, namely the maximum depth of the tree we computed for Π where each branch ends as soon as it becomes monochromatic. But maybe there is some *better* protocol Π', which achieves a better communication complexity, i.e., that yields a less-obvious, tighter upper bound. Given f, let $\chi(f)$ denote the minimum number of rectangles in any monochromatic tiling of $M(f)$. Then we claim the following.

Theorem 1 (AUY83)**.**

$$\log_2 \chi(f) \leq C(f) \leq 16\left(\log_2 \chi(f)\right)^2$$

- TODO- prove it and explain and illustrate the proof
- TODO- give and prove Lemma 13.9 relating to fooling set