Determining Smallest Path Size of Multiplication Transducers Without a Restricted Digit Set

Aditya Mittal[1], **Karthik Mittal**[1], **Simone Sisneros-Thiry**[2]

[1]James Logan High School, [2]California State University, East Bay

Abstract

Directed multiplication transducers are a tool for performing base multiplication without converting to base ten, allowing for faster computation and visualization. We traversed these transducers through a recursive approach and conjectured the smallest closed loop around a multiplication transducer starting and ending at zero using these results. A general pattern for this loop, the length of the loop for a transducer base b, and the range of multipliers having a particular length n for multiplier m were identified. This research can be expanded by testing reductions of the digit set to allow for additional constraints.

Multiplication Transducers

Directed multiplication transducers are tools to multiply a number by base b without the need for conversion into an intermediary base such as base ten. Finding patterns in these transducers (e.g. recursive formulas and minimum path lengths for specific base and multiplier transducers) can optimize methods for finding products and create new insights in the field of quotient sets.

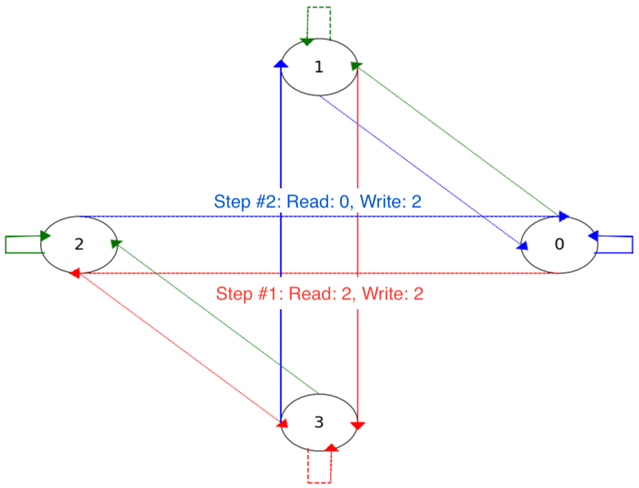

Fig. 1: Representation of the multiplication transducer $T_{4,3}$ using Matplotlib with steps from the section "Example".

In regular base ten multiplication, you use an algorithm, splitting up the mutlplicand into manageable terms (e.g. 132 into 100, 30, and 2) to be multiplied by the multiplier. This uses a tweaked algorithm for any non-base-10 multiplication.

There are five components in step i: the carry-in value c_i, the read value r_i, the total value t_i, the write value w_i and the carry-over value c_{i+1}.

These values are governed by these equations:

$$t_i = r_i m + c_i \tag{1}$$

$$c_{i+1} = \left\lfloor \frac{t_i}{b} \right\rfloor \tag{2}$$

Example

Let's take an example of $m = 4$ and $b = 3$. Let $r = [20]_{10} = [202]_3$. We can use multiplication transducers to determine $[202]_3 * 4$ without converting to base 10.

Step 1 ($i = 0$): $c_0 = 0$ (our initial state) and $r_0 = 2$. The following is determined:

- $t_0 = r_0 m + c_0 = 2 * 4 + 0 = 8$
- $c_1 = \lfloor \frac{t_0}{b} \rfloor = \lfloor \frac{8}{3} \rfloor = 2$
- $w_0 = t_0 \text{ (mod b)} = 8 \text{ (mod 3)} = 2$

Step 2 ($i = 1$): $c_1 = 2$ and $r_1 = 0$. The following is determined:

- $t_1 = r_1 m + c_1 = 0 * 4 + 2 = 2$
- $c_2 = \lfloor \frac{t_1}{b} \rfloor = \lfloor \frac{2}{3} \rfloor = 0$
- $w_1 = t_1 \text{ (mod b)} = 2 \text{ (mod 3)} = 2$

The iteration is terminated when there are no more read values and $c_{i+1} = 0$. Adding up the write values gives our final value. We know w in base 3 is $[w_3 w_2 w_1 w_0]_3 = [2222]_3$.

Therefore, we can say $w = (2 * 3^0) + (2 * 3^1) + (2 * 3^2) + (2 * 3^3) = 80$. This multiplication is confirmed in base-10: $r * m = 20 * 4 = 80$.

Methods - Transducer Traversal and Visualization

These methods find the **minimum length for a path of c's starting and ending at zero for $T_{m,b}$. Transducers were formed with the *networkx* library** using the .add_node(), and .add_edge() commands. The .DiGraph() command was used since the transducer is a directed graph, with arrows going from one state to another. A transducer was formed, iterating through read and carry value combinations for certain *b*'s and *m*'s.

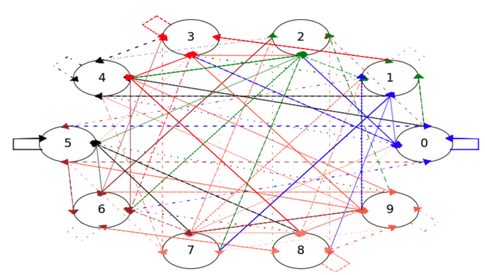

Fig. 2: Representation of the more complex multiplication transducer $T_{7,10}$ using Matplotlib.

Using *networkx* commands, this directed graph was traversed to find the shortest paths in the graph that start and end at zero; we primarily implemented depth first search as our graph traversal algorithm on the transducer to identify the minimum path, running a different graph traversal than the one offered in *networkx*.

A C++ implementation was taken due to the language's computational efficiency in reducing time and space complexity; further, other graph-based libraries were tested alongside *networkx* to determine the most efficient approach to take.

Results

These results discuss derivations for the smallest closed loop across a multiplication transducer $T_{m,b}$ starting and ending at 0.

Conjecture 1 (computationally proven for $1 < m, b < 2000$)**:** For all natural numbers b, $m > 1$, the path of carry values c_i's is $c_0 = 0$, $c_1 = \lfloor \frac{m}{b} \rfloor$, and $c_i = \lfloor \frac{c_{i-1}}{b} \rfloor$ for $i \geq 2$.

Theorem 1: $r = 1$ and $w = m$.

- *Proof.* The smallest closed loop across a multiplication transducer $T_{m,b}$ must contain a $c_i \neq 0$ to be non-trivial.
- We must set the read value to 1 to minimize t_i and future carry values to reach a carry value of 0 as quickly as possible.

Corollaries:

- **Corollary 1:** The length for the path of carry values is $\lfloor m^{1/b} \rfloor + 2$ for $b \geq 2$.
  - *What is the relationship with the length of carry values to m, b?* From Conjecture 1, we divide by b in every step, so to make the last integer division produce a zero, $\frac{m}{b^{w-1}} < 1$.
- **Corollary 2:** The multipliers that have a length of $n + 1$ has a range of $m \in [b^{n-1}, b^n - 1]$ for all $n \geq 3$ and $b \geq 2$.
  - *How can we prove these bounds are sharp?* We prove that the multiplier length as a function of m is monotonically increasing, and then find the multiplier lengths of the two given bounds.

Quotient Sets

We now add a further constraint to limit the r values that can be multiplied by m.This constraint involves **reducing the original digit set $\{d_1, d_2, ..., d_k\}$, the set of digits that can be used to represent r in base b, where $k = b$.**

Let $S(b; \{d_1, ..., d_k\})$ be the set of r that exist in base b with digit set $\{d_1, ..., d_k\}$. We are interested in studying a quotient set, the positive numbers that come from dividing numbers in $S(b; \{d_1, ..., d_k\})$ or the set of all q for base b.

References

F. Blanchard, J. M. Dumont, and A. Thomas. "Generic sequences, transducers and multiplication of normal numbers". Israel Journal of Mathematics, 1992.

T. Everitt and M. Hutter. "Analytical Results on the BFS vs. DFS Algorithm Selection Problem. Part I: Tree Search". AI 2015: Advances in Artificial Intelligence, 2015.

S. Sisneros-Thiry. "Combinatorial Number Theory Through Diagramming And Gesture". University of Illinois at Urbana-Champaign, 2020.

Github: `https://github.com/karthikm15/Multiplication-Transducer-Research`