Asymmetric Communication Complexity and Data Structure Lower Bounds

Yuanhao Wei

11 November 2014

1 Introduction

This lecture will be mostly based off of Miltersen's paper *Cell Probe Complexity - a Survey* that was publisehd in the year 2000.

1.1 The Cell Probe Model

You can think of the cell probe model as having a CPU and some random access memory (RAM). The state of your data structure is encoded in the memory cells of the RAM and the CPU must answer some query by accessing certain memory cells. For the rest of the lecture s will denote the number of memory cells and w will denote the number of bits in each memory cell. The only thing you are charged for is the number of memory cells accessed or modified. When answering a query, the CPU decides what cell to ask for next by looking at the query as well as all the previous cells that it has accessed.

The cell probe model was first introduced by Yao in 1981. This is the main model for proving data structure lower bounds because its nice and clean, and its the strongest possible model for lower bounds. This means that lower bounds in the cell probe model imply lower bounds in all other models of computation. Also upper bounds in other models imply upper bounds in the cell probe model.

Definition A static data structure problem is where you are given a map $f : Q \times D \to A$ where D is a finite set of possible data, Q is a finite set of possible queries, and $f(q, d)$ is the answer to query q about data d. A would be the set of answers.

Definition A solution to the static problem in the cell probe model with parameters s, w and t, is a method of encoding any data $d \in D$ in the memory of a random access machine, using s memory cells, each containing w bits, so that to answer any query in Q we need to probe at most t memory cells.

Example The *static predecessor problem*: Given an ordered universe $U = \{1, ..., m\}$ and a subset S of U of size n, we want to encode S in such a way that we can efficiently find the predecessor of x in S for any $x \in U$.

n typically denotes the number of elements in the universe U that is in each element of D. m typically denotes the size of U. In this lecture, I will prove lower bounds on t for the case where $s \in n^{O(1)}$. w is usually assumed to be $\Theta(\max(\log m, \log s))$ because that allows each cell in memory to store an element of the universe or a pointer to a location in memory.

Definition Dynamic data structure problems are similar to static data structure problems except that you must also be able to update the data in the data structure efficiently.

Example The *dynamic polynomial evaluation problem*: Given a finite field $\mathbb{F}$ of m elements, let $(a_0, ..., a_n) \in \mathbb{F}^{n+1}$ be the coefficients of a polynomial f. The goal is to efficiently maintain the coefficients $(a_0, ..., a_n) \in \mathbb{F}^{n+1}$ under $CHANGE(i, b)$ which sets $a_i = b$ and $EVALUATE(x)$ which returns $f(x)$.

For dynamic problems, the amount of space used by the data structure is usually ignored however adding the weak assumption that $s \in n^{O(1)}$ makes lower bounds easier to prove. Again w is assumed to be $\Theta(\max(\log m, \log s))$.

1.2 Asymmetric Communication Complexity vs. Cell Probe Complexity

For this lecture we will only need to deal with deterministic communication complexity. You can get a communication game from any static data structure problem in the intuitive way: Alice gets a query $q \in Q$ and Bob gets the data $d \in D$ and they want to compute the joint function $h(q, d)$. However there as a few difference between this communication game and what we have seen in class. First of all, for most natural data structure problems, the size of the data will be much greater than the size of the query so the communication game is asymmetric. Also only Alice needs to know the answer at the end. Furthermore the communication complexity of this game must be measured in a different way. Alice will choose her messages from $\{1, ..., s\}$ and Bob will choose his messages from $\{0, ..., 2^w - 1\}$. The complexity of a protocol is the maximum number of rounds of messages sent between Alice and Bob. This way Alice can request a memory cell in one message and Bob can tell Alice the contents of that cell in one message. Formally:

Definition We say that a protocol, P, is a $[t, a, w]^A$-protocol for a joint function h if each of Alice's messages in the protocol contains at most a bits and each of Bob's messages in the protocol contains at most b bits and at most t rounds of messages are sent, with Alice sending the first message. A $[t, a, w]^B$-protocol is defined similarly where Bob sends the first message.

In general the communication game might have a exponentially shorter protocol than the original static data structure problem. This is due to the fact that Alice can send Bob her entire input and Bob can compute the answer which is not possible in the cell probe model. Also in the cell probe model, the CPU is forced to query memory cells, whereas, Alice and Bob can sent whatever messages they want. However, the following lemma shows that a lower bound for the communication game implies a lower bound for the corresponding static data structure problem.

Lemma 1 If there is a cell probe solution using s memory cells and at most t probes for the static data structure problem, then there is a $[t, \log s, w]^A$-protocol for the communication problem corresponding to the static problem.

Proof The protocol basically simulates the solution to the static data structure problem. Bob encodes his input in a data structure using s space. Each time the solution queries a memory cell, Alice sends $\log s$ bits indicating the address of the cell she wants to read and Bob replies with the w bits stored at that location. Afterwards, Alice uses the cells she has read and the cell probe solution to compute the answer. Since the cell probe solution requires at most t queries, the protocol sends at most t rounds of messages.

Note that $\log s \in O(\log n)$ by our assumptions. So to prove a lower bound of $\Omega(f)$ on the static problem we just need to show that there is no $[o(f), O(\log n), w]^A$-protocol for the communication problem.

Now we will show that static data structure problems can be reduced to a corresponding dynamic data structure problem. In this reduction we have to be a little bit careful with the size bounds on the data structure.

Definition The static data structure problem f corresponding to a dynamic data structure problem g is defined as follows: for some parameter d, $f : D \times Q \to A$ where Q is the set of queries in the dynamic problem, and D is the set of states that are reachable from the initial state by performing at most d update operations in the dynamic problem.

Lemma 2 If there is a cell probe solution using $n^{O(1)}$ memory cells and t probes for the dynamic problem g then the corresponding static problem f has a solution using $O(dt)$ memory cells and $O(t)$ probes.

Proof The static solution basically simulates the dynamic solution. Assume that the data structure for the dynamic solution is encoded in memory in such a way that the initial state is encoded to all zeros. In the static solution, instead of storing the values of all the memory cells used by the dynamic solution, we map the cells that have changed to their final values. Since at most dt cells can be changed, using the FKS dictionary structure, this can be done in $O(dt)$ space. Each time the dynamic solution wants to query a cell, we check if that cell is in the dictionary. If it is then return the value that it is mapped to. Otherwise return 0. Each query takes $O(1)$ time and we make at most t queries so this solution to the static data structure problem takes $O(t)$ query time.

Therefore we have shown that lower bounds for a communication game implies a lower bound for a static data structure problem which implies a lower bound for a dynamic data structure problem.

Since Alice can always just send her entire input to Bob, the best possible lower bound we can prove using this kind of communication complexity reduction is t $= \Omega(\frac{\log |Q|}{\log s})$. If the number of queries is polynomial in n, this lower bounds is just a constant so for the rest of the lecture we will only consider problems where the number of queries are at least exponential in n. There are many natural problems that satisfy this criteria and this allows us to prove lower bounds up to $\Omega(\frac{n}{\log n})$.

2 Proving Lower Bounds

The main techniques for proving lower bounds in the cell probe model via communication complexity are round elimination, richness technique, greedy communication complexity technique and reduction to set disjointness. I will be presenting the greedy and round elimination techniques. You can read about the richness technique in Miltersen, Nisan, Safra and Wigderson's paper *On data structures and asymmetric communication complexity* and reductions to set disjointness in Patrascu's paper *Unifying the Landscape of Cell Probe Lower Bounds*.

2.1 Greedy Communication Complexity Technique

Monochromatic Row Condition Consider a function $h : A \times B \to C$. $A' \subseteq A$ and $B' \subseteq B$ satisfy the monochromatic row condition if:

$$\forall x \in A' \ \forall y, z \in B' : h(x, y) = h(x, z).$$

The idea behind the Greedy Communication Technique is to look at the communication matrix between Alice and Bob and show that all rectangles, A' and B' that satisfy the monochromatic row condition must be small then apply the Greedy Communication Lemma to get a lower bound on the number of rounds of the protocol. I will refer to rectangles that satisfy the monochromatic row condition as nice rectangles.

Consider a general communication problem $h : A \times B \to C$, where Alice is given $a \in A$ and Bob is given $b \in B$ and Alice needs to compute $h(a, b)$. Alice will choose her messages from $\{1, ..., s\}$ and Bob will choose his from $\{1, ..., k\}$.

Greedy Communication Lemma If h has a t round protocol, then there is $A' \subseteq A$ and $B' \subseteq B$ such that $|A'| \geq |A|/s^t$ and $|B'| \geq |B|/k^t$ and A', B' satisfy the monochromatic row condition.

Proof Proceed by induction on t. For $t = 0$, the lemma holds because we can take $A' = A$ and $B' = B$. This satisfies the monochromatic row condition because this problem requires 0 rounds of communication so the answer only depends on Alice's input. For the inductive step, suppose this lemma holds for t. Given any communication problem h with a $t+1$ round protocol P. For $a \in \{1, ..., s\}$, define A_a to be the inputs $x \in A$ for which Alice would send a as her first messages. By the pidgeonhole principle, we can fix a so that $|A_a| \geq |A|/s$. Similarly for $b \in \{1, ..., k\}$, define B_b be the inputs $y \in B$ for which Bob would send b as his first messages if he recieved the message a from Alice. Again fix b so that $|B_b| \geq |B|/k$. There is a t round protocol P' for the communication problem restricted to $A_a \times B_b$. P' just simulates P from the second round onwards assuming that Alice sent a and Bob sent b in the first round. By the inductive hypothesis, we can find A'_a and B'_b such that they statify the monochromatic row condition and $|A'_a| \geq |A_a|/s^t$ and $|B'_b| \geq |B_b|/k^t$. So we can take $A' = A'_a$ and $B' = B'_b$ then they statisfy the monochromatic row condition and $|A'| \geq |A|/s^{t+1}$ and $|B'| \geq |B|/k^{t+1}$ as required.

Example $\Omega(n)$ lower bound for the dynamic polynomial evaluation problem.

Recall the dynamic polynomial evaluation problem defined in the introduction. Clearly there is a trivial $t \in O(n)$ solution for the problem where we store each of the coefficients in a memory cell and for each call to $EVALUATE(x)$ we read all $n+1$ memory cells and evaluate it at x. For each call to $CHANGE(i, b)$ we just update the value at the corresponding memory cell. In this example we will show that this trivial solution is in fact optimal.

Definition Define the static version of this problem as: Store $(a_0, ..., a_{n-1}) \in \mathbb{F}^n$, the coefficients of f, in a data structure using $s = O(n^2)$ memory cells, so that $f(x)$ can be computed efficiently.

By Lemma 2, a lower bound of $\Omega(n)$ for this problem implies a lower bound of $\Omega(n)$ for the dynamic polynomial evaluation problem where $s \in n^{O(1)}$. We can pick $s = O(n^2)$ in the static problem because any polynomial can be achieved from the initial state by at most $n+1$ $CHANGE$ operations and each operation can modify at most $O(n)$ memory cells.

From the static polynomial evaluation problem we can get the corresponding communication game where Bob gets a n degree polynomial, f, which is represented by some $y \in F^{n+1}$ and Alice gets $x \in F$. The goal is to compute $f(x)$. Also Alice sends messages from $\{1, ..., s\}$ and Bob sends messages from $\{1, .., |F|\}$. Consider the case where $F \geq 2^{n \log n}$.

Now we use a nice property of polynomials to show that all nice rectangles in the communication matrix of this problem must be small. Suppose $A' \times B'$ is a nice rectangle. Two different n degree polynomials can agree on at most n different points. Therefore $|A'| \leq n$ or $|B'| \leq 1$. Next we apply the greedy communication lemma.

If $|A'| \leq n$, then by the greedy communication lemma, $n \geq |F|/s^t$ so $t \geq \frac{\log |F| - \log n}{\log s} \in \Omega(n)$. If $|B'| \leq 1$, then by the greedy communication lemma, $1 \geq \frac{|F|^{n+1}}{|F|^t}$ so $t \geq n+1$. Therefore $t \in \Omega(n)$ as required.

2.2 Round Elimination Technique

Definition Given a communication problem $f : X \times Y \to \{0, 1\}$. We define the augmented problem, $f^{(r)}$, by: Alice gets r elements of X, labeled, $x_1, ..., x_r$. Bob gets $y \in Y$, an integer $i \in \{1, ..., r\}$ and $x_1, ..., x_{i-1}$ from Alice's input. The goal is to compute $f(x_i, y)$. The reverse augmented problem, ${}^{(r)}f$, is defined symmetrically switching the roles of Bob and Alice.

Round Elimination Lemma There exists c, a universal constant, so that for any communication problem f, if there is a $[t, a, w]^A$-protocol for $f^{(ca)}$ then there is a $[t-1, ca, cw]^B$-protocol for f. Symmetrically, if there is a $[t, a, w]^B$-protocol for ${}^{(ca)}f$ then there is a $[t-1, ca, cw]^A$-protocol for f.

Proof The proof for this lemma is quite long and it can be found in Miltersen, Nisan, Safra and Wigderson's paper *On data structures and asymmetric communication complexity.* The intuition is that since Alice doesn't know which of her inputs is important, she cannot give much information to Bob about the important x_i in her first message.

The lemma says the you can convert a solution to the augmented version into a solution to the original problem which requires one less round of communication.

The general round elimination technique works as follows. Given a protocol for the original problem, try to convert it into a protocol for the augmented version of the problem with smaller problem size and the same number of rounds. Once you have accomplished this, you will be able to go back and forth between the the augmented version and the non-augmented version and each time you will shave off one round of communication. The second step is to find a general formula for the size of the problem after T rounds of communication have been elminated. Finally substitute $T(n) \in \Theta(f)$ into the general formula and arrive at a contradiction. This gives a lower bound of $\Omega(f)$.

Example $\Omega(\log^{1/3} n)$ lower bound for the static predecessor search problem.

The upper bound for this problem is $t \in O(\sqrt{\frac{\log n}{\log \log n}})$ for $s \in O(n)$ and the tight lower bound is $t \in \Omega(\sqrt{\frac{\log n}{\log \log n}})$ for $s \in n^{O(1)}$. Both these results were proven by Beame and Fich in their 1999 paper, *Optimal bounds for the predecessor problem.* Although the proof of the tight lower bound involves round elimination, there is much more going on in that proof so I will only give the proof for the weaker $\Omega(\log^{1/3} n)$ lower bound.

First, we will consider a different problem.

Definition The predecessor color problem is where you are given a universe $U = \{0, ..., m-1\}$ and you want to store a subset S of U of size at most n where each element of S is assigned a color from $\{red, blue\}$ so that you can efficiently find the color of the predecessor of x for any $x \in U$.

This clearly reduces to the predecessor search problem because to solve the new problem you can just find the predecessor and read its color from a lookup table. Therefore we just need to prove a lower bound on this new problem. Define col$[b, n]$ to be the communication problem where Alice gets $a \in U = \{0, ..., 2^b - 1\}$ and Bob gets a subset S of U where each element of S is colored either red or blue. Alice has to solve the predecessor color problem on this input.

The following lemma shows that a solution to the predecessor color problem can be converted into a solution to the augmented version with smaller problem size.

Lemma 3 If there is a $[t, a, w]^A$-protocol for col$[b, n]$ then there is a $[t, a, w]^A$-protocol for col$[\frac{b}{ca}, n]^{(ca)}$.

Proof Suppose we have a $[t, a, w]^A$-protocol for col$[b, n]$. Given a col$[\frac{b}{ca}, n]^{(ca)}$ problem, we know that Alice gets $x_a, ..., x_{ca}$, each containing $\frac{b}{ca}$ bits, and Bob gets $S, i, x_1, ..., x_{i-1}$. Alice can compute $x' = x_1 \circ x_2 \circ ... \circ x_{ca}$ where $\circ$ indicates concatenation. Bob can compute $S' = \{x_1 \circ x_2 \circ ... \circ x_{i-1} \circ y \circ 0^{b - \frac{b}{ca} i} | y \in S\}$, where each element in S' inherits the color of y. Then they execute the $[t, a, w]^A$-protocol for col$[n, b]$ using x' and S' as inputs. The anwser to col$[\frac{b}{ca}, n]^{(ca)}$ is the same as the answer to col$[n, b]$ on x' and S'.

The following lemma serves a similar purpose but in the case where Bob communicates first.

Lemma 4 If there is a $[t, a, w]^B$-protocol for col$[b, n]$ then there is a $[t, a, w]^B$-protocol for ${}^{(cw)}$col$[b - \log(cw), \frac{n}{cw}]$.

Proof Suppose we have a $[t, a, w]^B$-protocol for col$[b, n]$. Given the problem ${}^{(cw)}$col$[b - \log(cw), \frac{n}{cw}]$. Alice gets x and i and Bob gets $S_0, ..., S_{cw-1}$ as input. Alice can compute $x' = i \circ x$ and Bob can compute $S' = \bigcup_{j=0}^{cw-1} \{j \circ y | y \in S\}$ which inherits its color from y. Then they execute the $[t, a, w]^B$-protocol for col$[b, n]$ using x' and S' as inputs. The anwser to ${}^{(cw)}$col$[b - \log(cw), \frac{n}{cw}]$ is the same as the answer to col$[n, b]$ on x' and S'.

Now we use these lemmas to create a chain of conversions. A $[t, a, w]^A$-protocol for $col[b, n]$ can be converted into a $[t, a, w]^A$-protocol for $col[\frac{b}{ca}, n]^{(ca)}$ by lemma 3. This can be converted into a $[t-1, ca, cw]^B$-protocol for $col[\frac{b}{ca}, n]$ by the round elimination lemma. Again, this can be converted into a $[t-1, ca, cw]^B$-protocol for ${}^{(c^2w)}col[\frac{b}{ca} - \log(c^2w), \frac{n}{c^2w}]$ by lemma 4. This in turn can be converted into a $[t-2, c^2a, c^2w]^A$-protocol for $col[\frac{b}{ca} - \log(c^2w), \frac{n}{c^2w}]$ by the round elimination lemma with the roles of Alice and Bob switched. Let C be a constant so that $C \geq 2c^2$ and we only consider the setting where $log(c^2w) \leq \frac{b}{2ca}$. Then the previous protocol is a $[t-2, Ca, Cw]^A$-protocol for col$[\frac{b}{Ca}, \frac{n}{Cw}]$.

After repeating these steps $T = t/2$ times, we can convert a $[t, a, w]^A$-protocol for col$[b, n]$ into a $[0, C^T a, C^T w]$-protocol for col$[bc^{-T(T+1)/2}a^{-T}, nc^{-T(T+1)/2}w^{-T}]$ which also works for $col[bc^{-T^2}a^{-T}, nc^{-T^2}w^{-T}]$.

The final step is to plug in the lower bound that we want to prove to get a contradiction. Consider a $[T, O(\log n), w]^A$ protocol for col$[w, n]$ with $b = 2^{(\log w)^{3/2}}$ and $T = \frac{\sqrt{\log w}}{10\sqrt{\log c}}$. We have picked $T \in \Theta(\log^{1/3} n)$. Since n and w are both size parameters of col$[w, n]$, we can fix n in terms of w. After eliminating all the rounds of communication we get a zero round protocol for col$[w^{99/100-o(1)}, n^{9/10-o(1)}]$. However this problem clearly requires some communication to solve so we get a contradiction. Since the resulting problem size increases as T decreases, if we picked any $T \in o(\log^{1/3} n)$ then we would arrive at the same contradiction. Therefore $T \in \Omega(\log^{1/3} n)$ which is the lower bound we wanted. Also, if we picked T to grow faster than $\log^{1/3} n$ then the predecessor color problem will become trivial after T rounds of elimination and we would be unable to arrive at a contradiction.