Parallel Complexity Classes

The Circuit Model of Computation

- The circuit model is studied for two reasons:
  - to derive lower bounds on time & space
  - to understand parallel computation
- To simulate Turing machine computation, an infinite family of circuits is needed.
  - what kind of circuit families suffice?
- What languages can infinite circuit families recognize?
  - every language $L$, as we show below
  - but not every lang. is TM computable!
- Why?
  - For each $n$, if $x$ is in $L$, let $f^{(n)}(x) = 1$. For each $x$ of length $n$ not in $L$, $f^{(n)}(x) = 0$.
  - This family recognizes $L$!

Circuit Families

**Definition** A circuit family $C = \{C_1, C_2, \ldots\}$ is a collection of logic circuits in which $C_n$ has $n$ inputs and $m(n)$ outputs for some function $m: \mathbb{N} \rightarrow \mathbb{N}$.

A Time-$r(n)$ (Space-$r(n)$) **uniform circuit family** is a circuit family for which there is a DTM that for $n$ in unary writes the description of $C_n$ on its output tape in time (space) $r(n)$.

The function $f : B^* \rightarrow B^*$ is computed by $C$ if on an input of length $n$, $C_n$ computes the mapping $f_n$ defined by restricting $f$ to inputs of length $n$.

(If $f : B^* \rightarrow B$, $f$ recognizes $L$ for some $L$. $f_n$ is the **characteristic function** of $L$ on inputs of length $n$)

A log-space uniform family of circuits is a Space-$O(\log n)$ uniform circuit family.

Uniform Circuits Equivalent to TMs

**Forward Direction**

**Theorem** Let $p(n)$ be a proper polynomial. Then, every total function $f : B^* \rightarrow B^*$ ($f$ is defined for all inputs) computed by a DTM $M$ in time $p(n)$ on inputs of length $n$ can be computed by a log-space uniform circuit family.

**Proof** Let $f_n$ be $f$ restricted to inputs of length $n$. We assume that the length of the value of $f_n$ is the same on all inputs of length $n$. If not, on page 374 of the book a replacement DTM $M^*$ for $M$ is described. (It pads results so that all values of $f_n$ have the same length.) $M^*$ runs in time at most $p^2(n)$ on input of length $n$. Now use the program on next slide (see p.129 of book) to show the desired result. Q.E.D.
Program to Write Circuit

- The following program writes the description of a circuit simulating a T-step Turing machine on input \( w = w_0w_1...w_{n-1} \).

- Why is this a log-space program?

```plaintext
for i := 0 to n-1
  READ_VALUE(w_i)
  WRITE_INPUT(i, w_i)
for j := n to m-1
  WRITE_INPUT(j, \beta)
for t := 0 to T
  WRITE_CONTROL_UNIT(t, c_t)
  WRITE_OR(t, m)
for j := 0 to m-1
  WRITE_CELL_CIRCUIT(j, t)
```

Equivalence of CREW PRAM & Circuits

- The CREW PRAM permits concurrent reads from common memory locations but requires exclusive writes.

- Why are infeasible serial problems also infeasible parallel ones?

- Thus, when studying parallel computation, we focus primarily on poly-time problems.

Our Goal is to Show:

**Theorem** The functions \( f : \mathbb{B}^* \rightarrow \mathbb{B}^* \) computed by circuits of polynomial size and poly-logarithmic depth are the same as those computed by the CREW PRAM with a polynomial number of processors and poly-logarithmic time.

Uniform Circuits Equivalent to TMs

**Reverse Direction**

**Theorem** Let \( C \) be a log-space uniform circuit family. There exists a polynomial time DTM \( M \) that computes the functions computed by \( C \).

**Proof** Let \( M_C \) be the DTM that computes the circuits in \( C \). \( M \) invokes \( M_C \) on input string \( w \) of length \( n \) to compute a representation of \( C_n \). It then uses this representation to compute the output of \( C_n \) in time quadratic in its length which is a polynomial in \( n \).

Families of PRAMs

**Definition** A family of PRAM’s is a collection \( \mathcal{P} = \{ P_1, P_2, ... \} \) of PRAM’s & a function \( m : \mathbb{N} \rightarrow \mathbb{N} \) such that \( P_n \) operates on \( n \) inputs and has \( m(n) \) processors.

A log-space uniform PRAM family is a PRAM family for which there is a DTM that for \( n \) in unary writes the description of \( P_n \) containing \( m(n) \) processors on its output tape in space \( \log \) in \( n \).

**Note:** The description of a PRAM is a description of one processor (they are all the same) and the programs for each processor.
CREW PRAMs Simulated by Circuits

Theorem Let \( f : B^n \mapsto B, B = \{0,1\} \), be computed by a CREW PRAM with \( p(n) \) (polynomial) processors in time \( t(n) \) in which the largest memory address is \( O(p(n)t(n)) \). Then \( f \) can be computed by a circuit of size \( O(p^2(n)t(n) + p(n)t^2(n)) \) & depth \( O(\log p(n)t(n)) \).

Proof Since the instr. set of a PRAM processor is limited to adds, compares, and one-bit shifts, the length of a word can increase from its initial value by at most one bit per cycle. At the end of computation, words contain \( \leq t(n) + n + K \) bits, \( K \geq 0 \).

From the construction of a simple CPU in Section 3.10 the circuit size and depth of next-state/output circuit for the RAM CPU are \( O(t(n) + \log p(n)t(n)) \) and \( O(t(n) + \log (p(n)t(n))) \), respectively.

Proof (cont.) In Section 3.4 we gave a construction of next-state/output circuits for a RAM memory unit. This circuit has circuits:

a) for each word to replace its current value, and

b) to select the output memory word.

Use the second circuit without change. Construct a next-state/output circuit for the PRAM common memory by modifying the first circuit as follows. Since the 3 gates closest to the output \( w^*_{i,j} \) select the new value for a memory word. Replace gates by a circuit connected to each processor that either makes the new value of the memory word be the old value if no processor is addressing it, or sets its value to the new word produced by the one processor that is addressing it. (How would you do that?)

Thus, the size and depth of the next-state/output circuit for the common memory are \( O(p^2(n)t(n)) \) and \( O(\log p(n)t(n)) \), respectively. Combining these results we have the desired bound. Q.E.D.

Clearly, given a logspace uniform PRAM family, we can construct a logspace uniform circuit family that simulates it.
**Logspace Circuits Simulated by PRAMs**

**Theorem** Let $C = \{C_1, C_2, \ldots\}$ be a logspace uniform family of circuits. There exists a CREW PRAM that computes in poly-logarithmic time and polynomial number of processors the function computed by $C$.

**Proof** Use configuration graph to represent $M_C$ generating $C$. Take transitive closure of adjacency matrix $Q$ of this graph. See book for details.

---

**Circuit Complexity Classes**

$NC^k$ contains languages recognized by uniform family of circuits of polynomial size and depth $O(log^k n)$. $NC$ is union of all $NC^k$. 

$$NC = \bigcup_k NC^k$$

$NC$ is considered the largest feasibly parallelizable class of languages, that is, can be computed by a parallel machine in poly-log depth (time).

Prefix circuits are in $NC^1$.

For $k \geq 2$,

$$NC^1 \subseteq L \subseteq NL \subseteq NC^2 \subseteq NC^k \subseteq NC \subseteq P$$

It is not believed that all problems in $P$ are feasibly parallelizable.

---

**The Parallel Computation Thesis**

“Sequential space and parallel time are polynomially related.” If there exists a sequential algorithm that uses space $S$, then there is a parallel algorithm using time $p(S)$ for some polynomial $p$ and vice versa.

**Lemma** Logspace transformations can be realized by a CREW PRAM with polynomially many processors in time $O(log^2 n)$.

**Proof** Use the CREW PRAM of previous theorem.

**Theorem** If a $P$-complete problem can be solved in poly-log time with polynomially many processors on a CREW PRAM, so can all problems in $P$. 