CSCI 1010
Models of Computation

Lecture 22
Space-Time Tradeoffs
Overview

• Pebble Game on directed acyclic graphs (DAGs)
• Models space-time tradeoffs
• We develop lower bounds on space for trees, FFT and pyramid graphs
• FFT exhibits genuine tradeoff of space for time
• We also exhibit extreme tradeoffs
• This lecture provides a taste of “applied” theory
  – See Chapter 10 of Models of Computation by Savage
Motivation

• A CPU has a finite number of registers in which to do a computation.

• The Pebble Game models assignment of data to registers so that we can reduce time for a given amount of space. This is the register allocation problem, a standard compiler problem.

• If primary memory is too small to hold a program and data, secondary storage is necessary. Then, minimization of data movement between the two memories, the I/O problem, an OS problem.
Pebble Game

- Game is played on a directed acyclic graph (DAG).
- A pebble on a vertex corresponds to a datum being placed in a memory location (register).
- The number of pebbles (registers) is fixed.
- The goal is to advance pebbles to output vertices.
  - Can remove a pebble on an output after placed there.
- What is the minimum number of pebbles?
- How does space trade for time?
- Can extreme space-time behavior arise?
The Pebbling Problem

• Full set of rules on next slide.
• The goal is to pebble (compute) outputs.
• Inputs can be pebbled at any time. To pebble non-inputs must have pebbles on their predecessors.
Rules of the Pebble Game

• (Initialization) A pebble can be placed on an **input vertex** at any time.

• (Computation Step) A pebble can be placed on (or moved to) any **non-input vertex** only if all its immediate predecessors carry pebbles.

• (Pebble Deletion) A pebble can be removed at any time.

• (Goal) Each output vertex must be pebbled at least once.
Pebbling Strategy

• A **pebbling strategy** is a series of applications of the rules that results in pebbling all output vertices with pebbles.

• The **space** used by a strategy is the maximum number of pebbles that are on the graph at any one time.

• The **time** used by a strategy is the total number of placement of pebbles on vertices.
The 8-Input FFT Graph

- What is the minimum number of pebbles needed to pebble it?
- Is there a tradeoff between space and time?
A Strategy to Pebble a First Output

• An FFT contains complete balanced binary trees.
• Five pebbles suffice to pebble the 16-input FFT.

Pebble vertices in numbered order. Here the most pebbles are on graph.
Pebbling Binary Tree

- **Theorem** The complete balanced binary tree on \(2^n\) leaves requires space \(\geq n+1\). It can be pebbled with this minimal space in \(2^{n+1}-1\) time (or placements).

- **Proof** There is a last path from an input to the output to have a pebble. (Solid line in figure.) Every edge into this path must be blocked by a pebble requiring \(\geq n\) pebbles. Add the last pebble on an input and the result follows. The time bound follows because each vertex is pebbled once.
Minimum Space on the Binary Tree

• As the theorem demonstrates, the minimum space, $S_{\text{min}}$, to pebble a complete balanced binary tree with $2^n$ leaves and $N = 2^{n+1} - 1$ vertices satisfies $S_{\text{min}} = n + 1$. Thus, $S_{\text{min}} \geq \log_2 (N+1)$.

• We ask if for some other graph $S_{\text{min}}$ is a larger function of $N$.

• This will be true for the pyramid graph.
Pebbling the Pyramid Graph

• The pyramid graph on $n$ inputs can be pebbled with $n$ pebbles. We show that $S_{\text{min}} = n$.
• As shown below, when a pebble is placed on the last open path to the output, there are at least $n-1$ other pebbles on the graph.
• This graph has $N = n(n+1)/2$.
• Thus, $S_{\text{min}} > \sqrt{2N} - 1$. 
A Genuine Tradeoff

• Balanced binary tree & pyramid can be pebbled with minimum space without repebbling. Not so with the FFT, as stated below. (See book.)

• Theorem To pebble the FFT with S pebbles in T time steps requires that \((S+1)T \geq n^2/16\).
Other Space-Time Lower Bounds

• n x n matrix multiplication: \((S+1)T \geq \frac{n^3}{4}\)
• Cyclic shifting of n-tuples: \((S+1)T \geq \frac{n^2}{16}\)
• Multiplication of n-bit integers: \((S+1)T \geq \frac{n^2}{64}\)
Extreme Tradeoffs

• Is there a family of graphs \( \{H_k\} \) such that if \( H_k \) is pebbled with \( S_{\text{min}} \) pebbles, time is very large, but if \( S_{\text{min}} + 1 \) pebbles used, time is minimal?
Motivation for This Topic

• Compilers generally try to minimize the number of registers used by code.
• Let \(|H_k|\) be the size of the graph \(H_k\).
• Theorem If a compiler succeeds for the graph family \(\{H_k\}\), the work (number of pebblings) will be exponential in \(|H_k|\) with \(S_{\text{min}}\) registers but only \(|H_k|\) with \(S_{\text{min}} + 1\) registers.
Extreme Tradeoffs

• By induction easy to see that $H_k$ has $S_{\text{min}} = k$.
• Similarly, can pebble $H_k$ in time $|H_k|$ with $S = k+1$.
• Also, $|H_k| = 4k+3 + |H_{k-1}|$, $|H_k| = O(k^2)$
Extreme Tradeoffs

- Let $T_k$ be the time to pebble $H_k$ with $S = S_{\text{min}} = k$.
- To pebble one output of $H_k$, one pebble must be advanced to the white vertex. Do this $k+1$ times to pebble the $k+1$ outputs.
Extreme Tradeoffs

• $T_k = [T_{k-1} + 2(k+1)](k+1) + (k+1)^2$ or
• $T_k = [T_{k-1} + 3(k+1)](k+1) > (k+1)T_{k-1}$
• Since $T_2 = 30$, by induction $T_k \geq 2((k+1)!)$
• By Stirling’s approx.*, $T_k$ is exponential in $|H_k|$.

* See https://en.wikipedia.org/wiki/Stirling%27s_approximation
Review

• Pebble Game on directed acyclic graphs (DAGs)
• Models space-time tradeoffs
• We develop lower bounds on space for trees, FFT and pyramid graphs
• FFT exhibits genuine tradeoff of space for time
• We also exhibit extreme tradeoffs
• This lecture provides a taste of “applied” theory