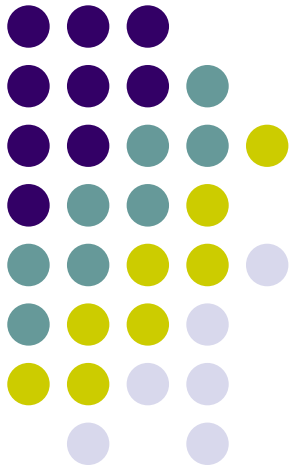
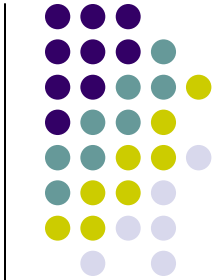
CS256
Applied Theory of Computation

Memory Hierarchy Tradeoffs I

John E Savage

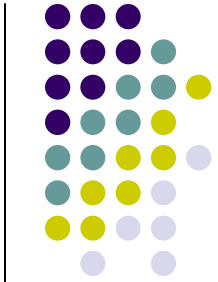


Overview

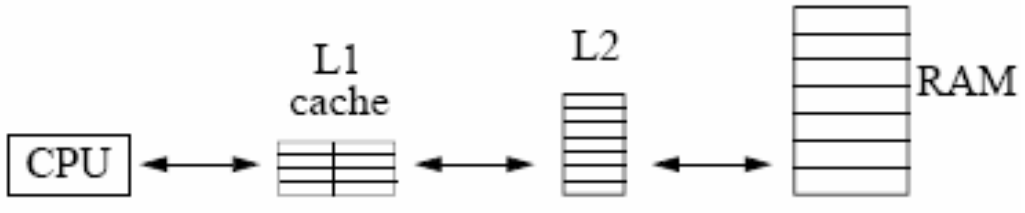


- Introduction to memory hierarchies
- The Red-Blue Pebbling Game
- Balancing I/O Time and Computation
- The Memory Hierarchy Game
- I/O Time Relationships
- The Hong-Kung Lower Bound

Introduction

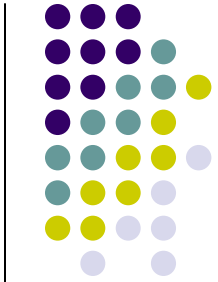


- Large RAM simulated by a hierarchy of memories; from small, very fast to slow, very large.



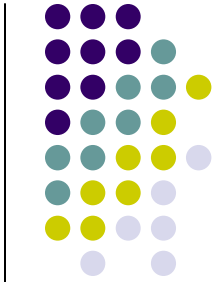
- I/O time is the time to move data between two levels in the hierarchy.

Introduction



- **Goal**: To understand how I/O time at each level trades with the size of the memories.
- This is an OS issue. Page replacement algorithms are used to decide which cache lines or RAM pages to move to the next higher level when data must be moved from the higher to the lower level.
- We start with two-level memory model.

The Red-Blue Pebble Game

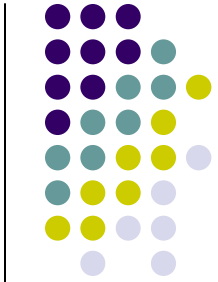


- The game is played on a DAG
- Hot red pebbles represent primary memory. Cool blue ones represent secondary memory.
- (initialization) A blue pebble can be placed on any input vertex at any time.
- (computation) A red pebble can be placed on (or moved to) any vertex all of whose predecessors carry red pebbles.
- (pebble deletion) A pebble can be deleted from a vertex at any time.

The Red-Blue Pebble Game

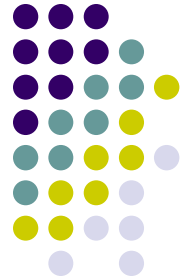
- (goal) A blue pebble must be on each output vertex at the end of the game.
- (input from blue level) A red pebble can be placed on any vertex carrying a blue one.
- (output from red level) A blue pebble can be placed on any vertex carrying a red one.

The Red-Blue Pebble Game

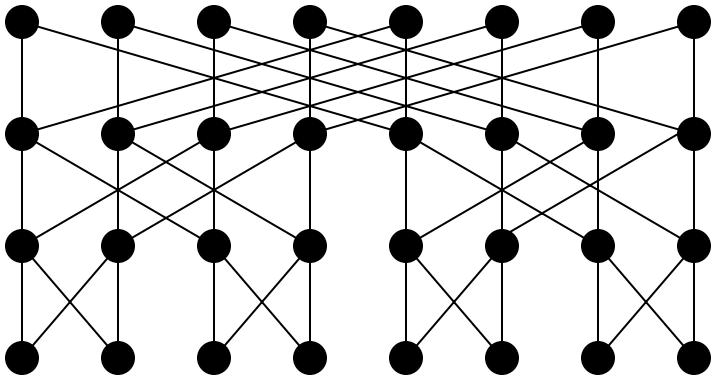


- **Definition** A **pebbling strategy** for a DAG is the sequential execution of the rules that places a blue pebble on each output. **Space** S is the number of red pebbles used. **I/O time** is number of I/O operations used by the strategy. A **minimal pebbling strategy** has minimal I/O time for a given amount of space.

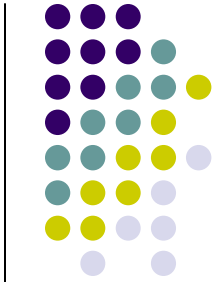
The Red-Blue Pebble Game on the FFT Graph



- The FFT graph on 2^k inputs can be pebbled with S = 2 but requires S = 2^k + 1 to pebble the graph with minimum number of I/Os.



Balancing I/O Time and Computation



- Later we show that number of I/O ops to do matrix multiplication using the standard algorithm on two-level machine is $O(n^3/\sqrt{S})$. The number of computations performed on the CPU is $O(n^3)$. If system is balanced and each I/O operation takes $t_{I/O}$ CPU steps, then $t_{I/O}\, n^3/\sqrt{S} \approx n^3$ or $t_{I/O} \approx \sqrt{S}$.
- As $t_{I/O}$ grows **S must grow quadratically** to keep a computer balanced for matrix multiplication.
- For FFT, we need $t_{I/O} \approx \log(S)$. Thus, balance requires **S grow exponentially with $t_{I/O}$ for FFT.**

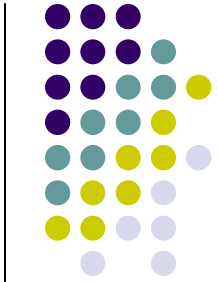
The Memory Hierarchy Game

- The game is played on a DAG. The number of pebbles at level $1 \le l \le$ L-1is p_l. p_L unlimited
- (initialization) A level-L pebble can be placed on any input vertex at any time.
- (computation) Level-1 pebble can be placed on (or moved to) vertex whose predecessors have level-1 pebbles.
- (pebble deletion) A pebble can be deleted from a vertex at any time.

The Memory Hierarchy Game

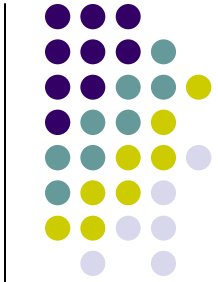
- (goal) A level-L pebble must be on each output vertex at the end of the game.
- (input from level-k) Level-(k-1) pebble can be placed on vertex carrying level-k pebble, $2 \leq k \leq L$
- (output from level-k) Level-(k+1) pebble can be placed on vertex carrying level-k pebble, $1 \leq k \leq L-1$.

The Memory Hierarchy Game



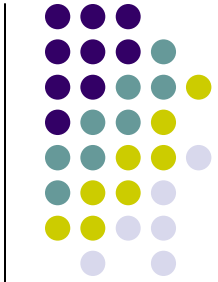
- In the **I/O-limited version** game level-L pebbles are allowed only on inputs and outputs. When L=2, this is the red pebble game; blue pebbles are allowed only on inputs and outputs.
- **Resource vector (RV) $\boldsymbol{p}$** = $(p_1, p_2, \ldots, p_{L-1})$ specifies the amount of *space* used at each level. $T_k^{(L)}(\boldsymbol{p},G,P)$ is the I/O time at level k with RV $\boldsymbol{p}$ on the DAG G using pebbling strategy P for $2 \le k \le L$. The computation time $T_1^{(L)}(\boldsymbol{p},G,P)$ is the number of times vertices are pebbled with level-1 pebbles.

The Memory Hierarchy Game



- A **minimal pebbling strategy** minimizes the number of I/O ops at level L, then at level L-1, all the way down to level 2. Finally it minimizes $T_1^{(L)}(\boldsymbol{p},G,P)$.

I/O Time Relationships



- Let P be strategy to pebble DAG G with RV $\boldsymbol{p}$. Let In(G) and Out(G) be G's inputs & outputs. Then,

$$T_k^{(L)}(\boldsymbol{p},G,P) \geq |In(G)| + |Out(G)| \text{ for } 2 \leq k \leq L$$

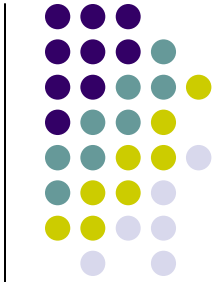
$$T_1^{(L)}(\boldsymbol{p},G,P) \geq |V| - |In(G)|$$

Theorem Let $s_k = p_1+p_2+\ldots+p_{k-1}$. Let $T_1^{(2)}(S,G,P)$ & $T_2^{(2)}(S,G,P)$ be the computation and I/O times for a minimal red-blue pebbling of G with S red pebbles.

$$T_k^{(L)}(\boldsymbol{p},G,P)) \geq T_2^{(2)}(s_{k-1},G,P) \text{ for } 2 \leq k \leq L$$

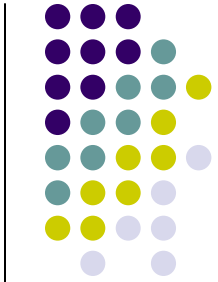
$$T_1^{(L)}(\boldsymbol{p},G,P) \geq T_1^{(2)}(s_{k-1},G,P) \text{ for } 2 \leq k \leq L$$

I/O Time Relationships



Proof Play red-blue game with S = s_{k-1} red pebbles and unlimited no. of blue pebbles. Classify S red and s_L -S blue pebbles into L-1 groups with p_j in the jth group. Now play the red-blue game by emulating moves in the minimal level-L game. This doesn't use fewer level-(k-1) I/O moves than does the minimal red-blue game, which implies the first inequality. The second is similar.

Hong-Kung Lower Bound

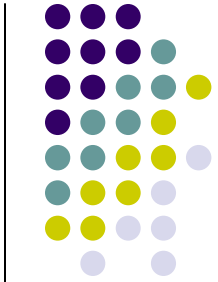


Definition The **S-span** of DAG G, ρ(S,G), is the maximum number of vertices of G that can be pebbled with S red pebbles in red pebble game maximized over all initial placements of S red pebbles. (Initialization rule is disallowed.)

Theorem For every pebbling P of G = (V,E) in the red-blue pebble game with S red pebbles, the I/O time used, $T_2^{(2)}(S,G,P)$ satisfies

$$\left\lceil T_2^{(2)}(S,G,P)/S \right\rceil \rho(2S,G) \geq |V| - |In(G)|$$

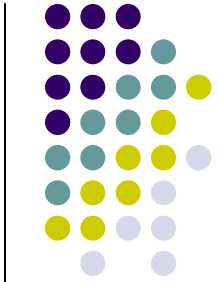
Hong-Kung Lower Bound



Proof Divide the pebbling steps P into subsets P_1, ..., P_h where each P_j consists of steps containing S I/O ops except the last which may have fewer. Then $h = \left\lceil T_2^{(2)}(S,G,P)/S \right\rceil$

We derive upper bound Q to number of vertices pebbled by red pebblings in P_j. Thus, Q*h* is at least |V| - |In(G)|.

Hong-Kung Lower Bound



Proof (cont.) We show that with S additional red pebbles we can move all I/O ops in an interval P_j to beginning or end of interval. Thus, we now have 2S pebbles and can pebble at most ρ(2S,G) vertices in this interval and at most *h*ρ(2S,G) in all *h* intervals.

Consider two types of vertex a) one carrying a red pebble at beginning of interval that is pebbled blue later, and b) vice versa. In a) use a new red pebble in lieu of first blue pebble. Keep it on the vertex until the end of interval or until last blue pebbling. All blue pebblings now at end of interval. In b) use a new red pebble for the first red pebbling & keep it on for rest of interval, moving blue pebblings to start of interval.