CSE 341T / 549T, Fall 2014
Parallel and Sequential Algorithms

Lecture 1: Overview

Kunal Agrawal and I-Ting Angelina Lee

August 25, 2014

What You Will Learn

- Algorithms
- Different techniques to parallelize these algorithms
- Cost analysis of these algorithms
- How to write parallel programs using Cilk Plus
- CSE 549: scheduling algorithms, cache-oblivious algorithms, concurrent data structures, nondeterministic programming.

Prerequisites

- Asymptotic Analysis
- How to solve recurrences
- Simple probability

We are going to have an in-class short quiz and admit students based on the performance in this quiz.

Review: The Master Method

The Master Method for solving divide-and-conquer recurrences applies to recurrences of the form*

$$T(n) = aT(n/b) + f(n),$$

where $a \geq 1$, $b > 1$, and f is asymptotically positive.

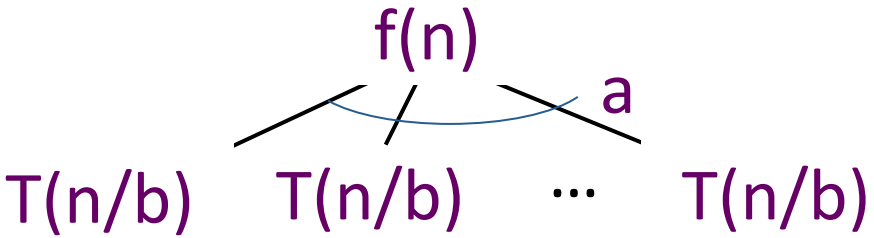
---

*The unstated base case is $T(n) = \Theta(1)$ for sufficiently small n.

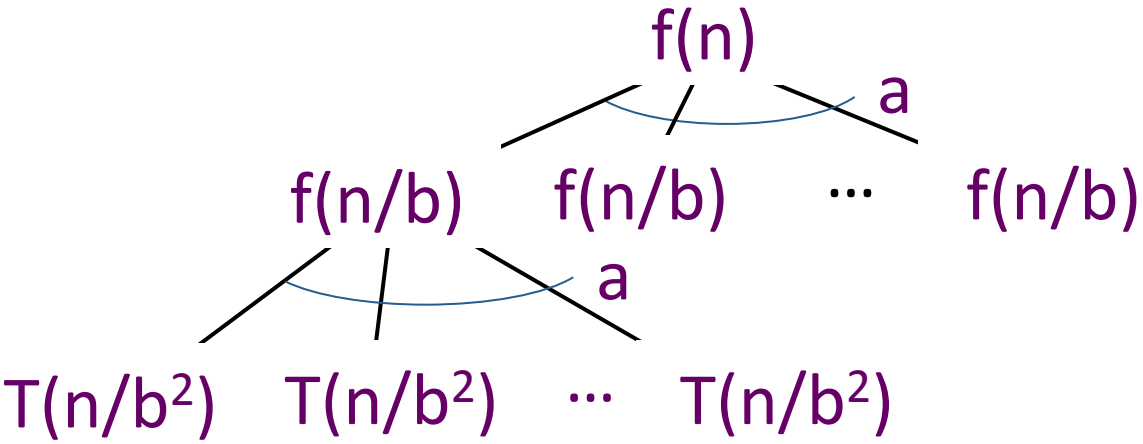
Recursion Tree: $T(n) = aT(n/b) + f(n)$

T(n)

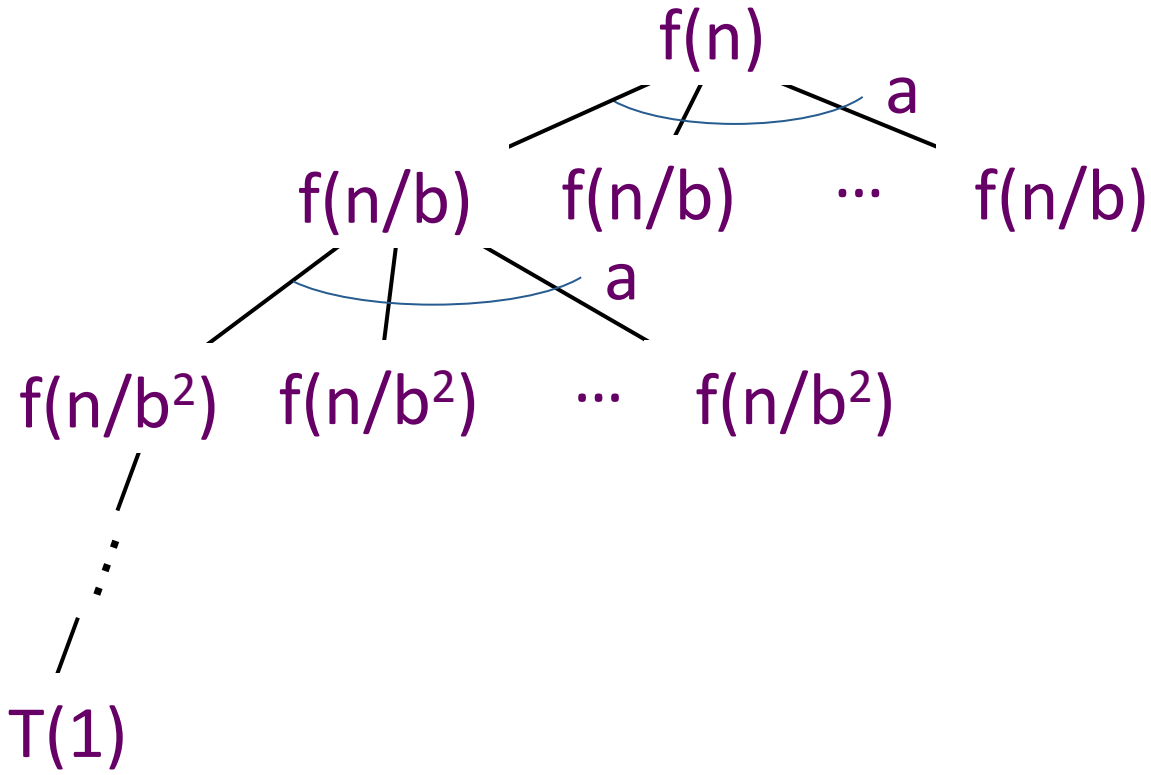
Recursion Tree: T(n) = aT(n/b) + f(n)



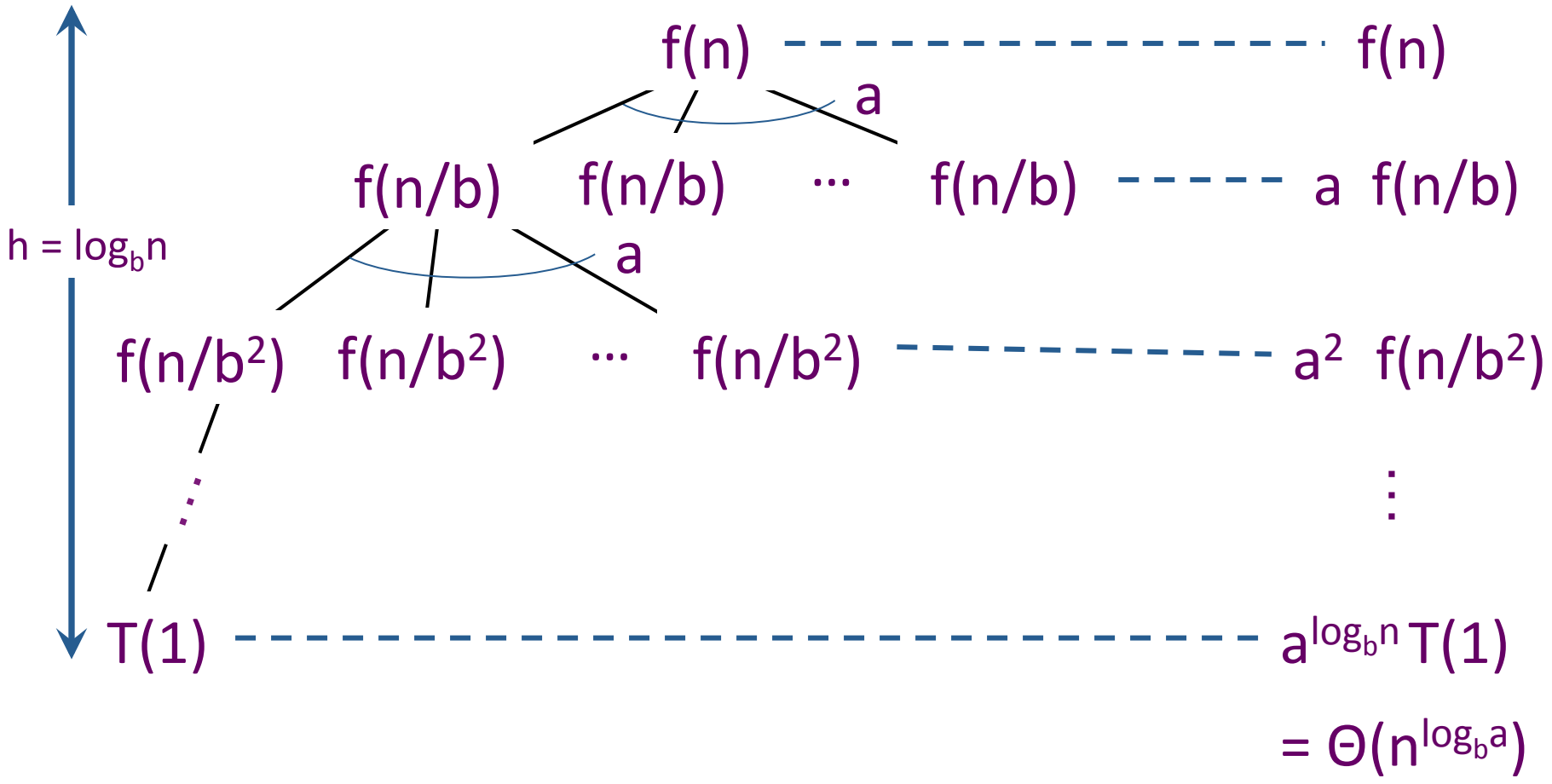
Recursion Tree: $T(n) = aT(n/b) + f(n)$



Recursion Tree: $T(n) = aT(n/b) + f(n)$

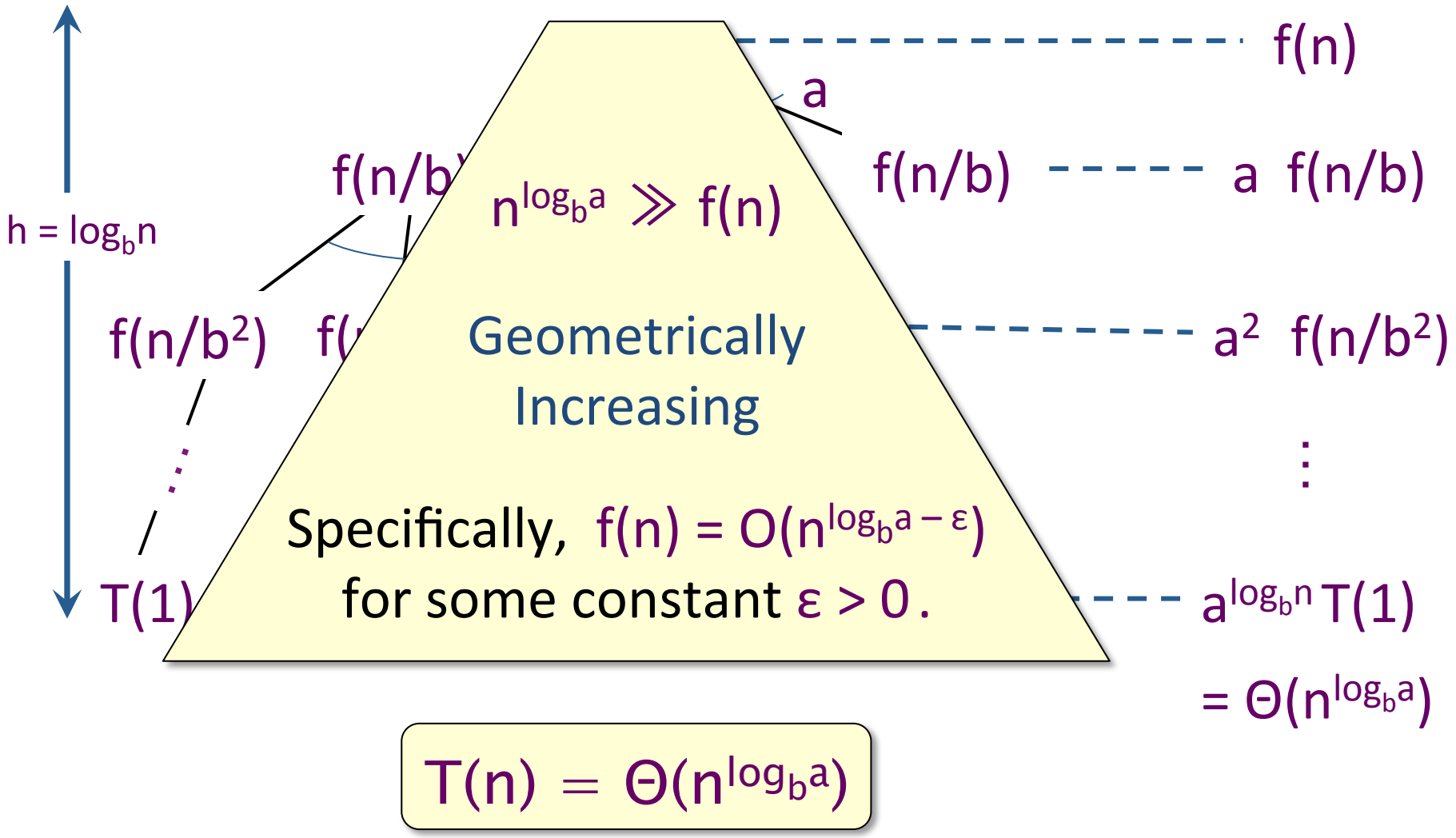


Recursion Tree: $T(n) = aT(n/b) + f(n)$



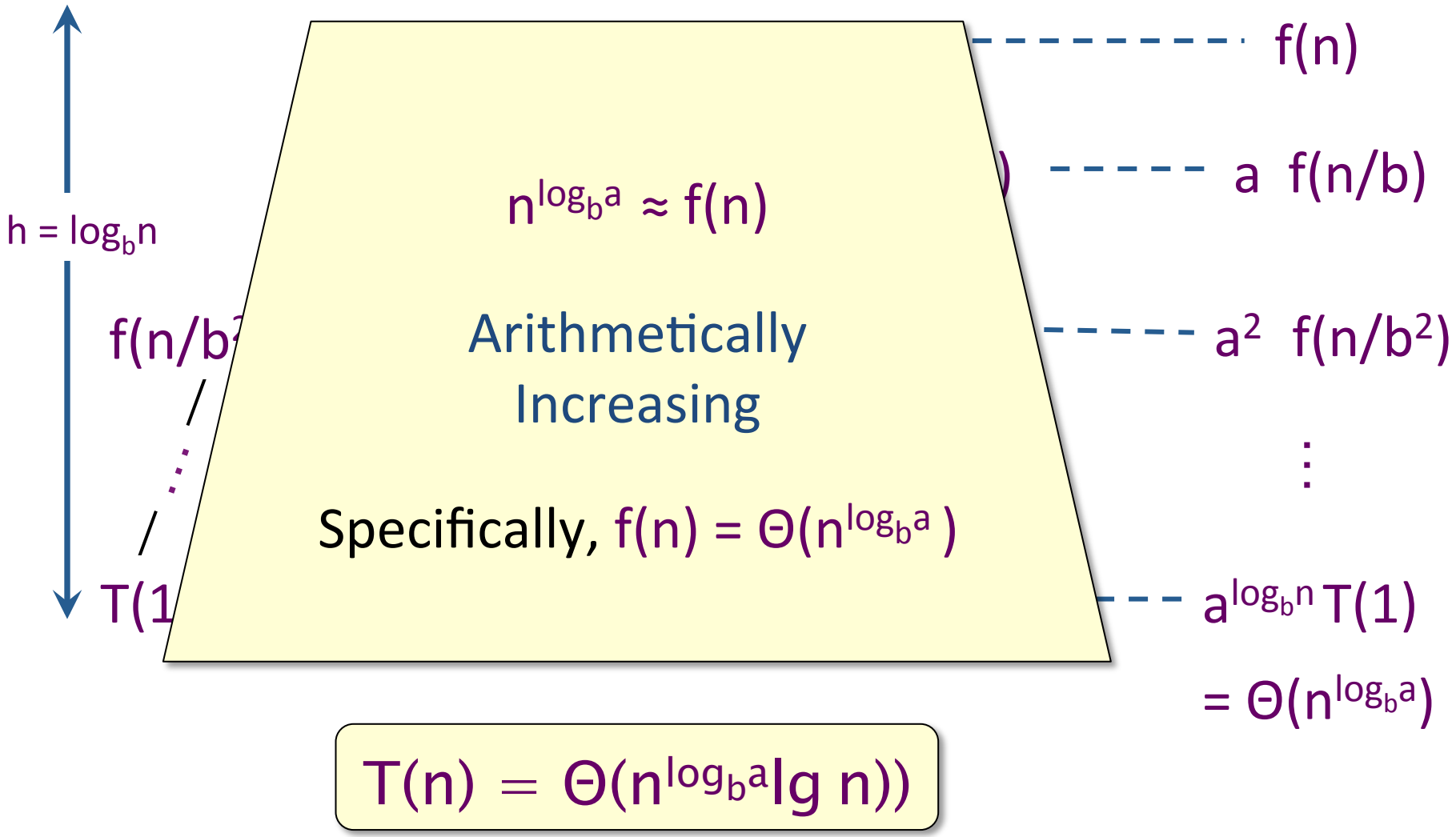
Idea: Compare $n^{\log_b a}$ with $f(n)$.

Master Method — CASE I

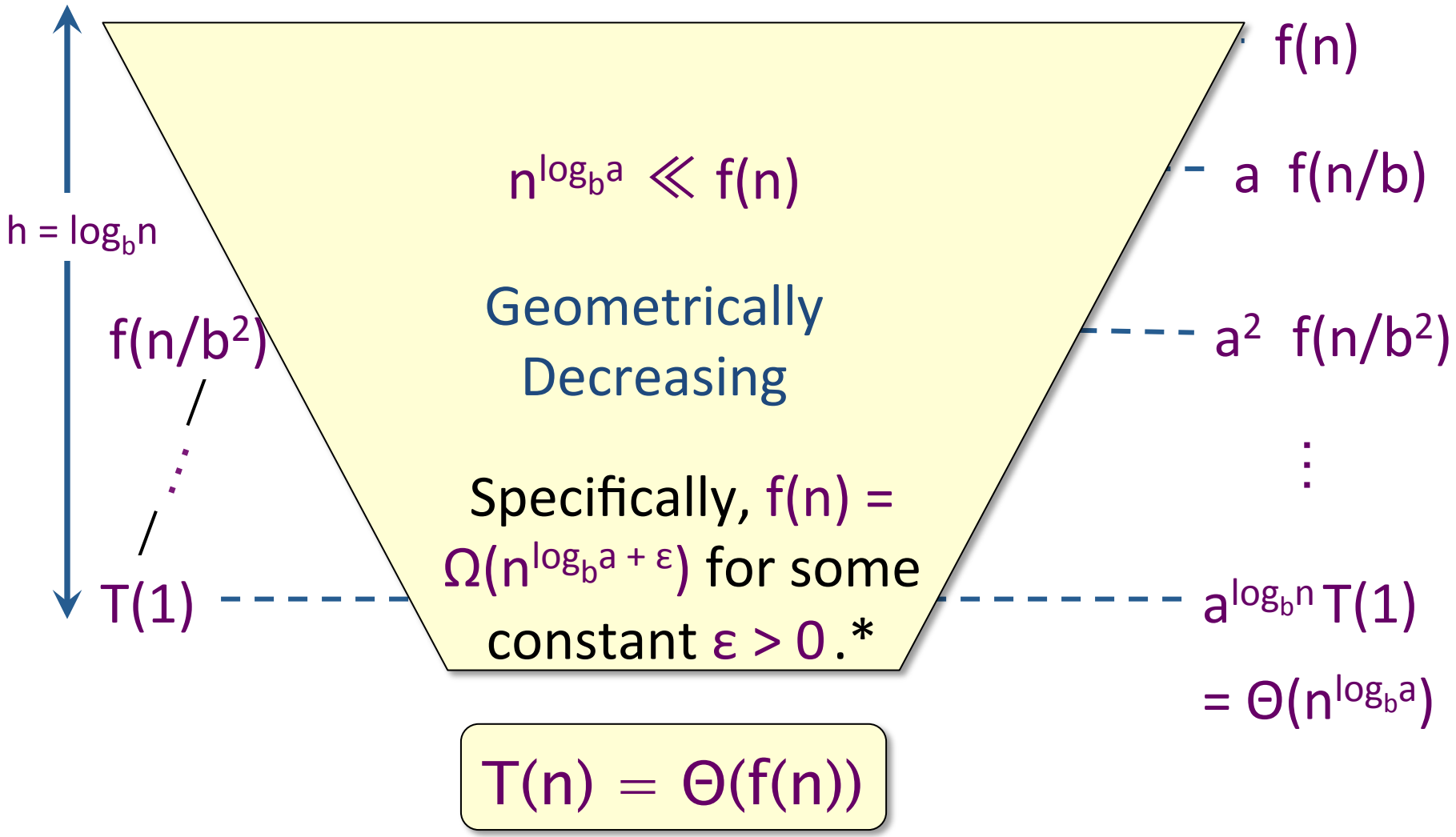


$$T(n) = \Theta(n^{\log_b a})$$

Master Method — CASE 2



Master Method — CASE 3



* and f(n) satisfies the **regularity condition** that $af(n/b) \le cf(n)$ for some constant $c < 1$.

In-Class Short Quiz

Minimum Edit Distance

Definition. Given a character set $\sum$, and two strings $S_1 = \sum^*$ and $S_2 = \sum^*$, determine the minimum number of insertions and deletions of single characters required to transform S_1 to S_2.

Applications. Spell correction, version control system, approximate marching of genome sequences … etc.

Minimum Edit Distance: An Example

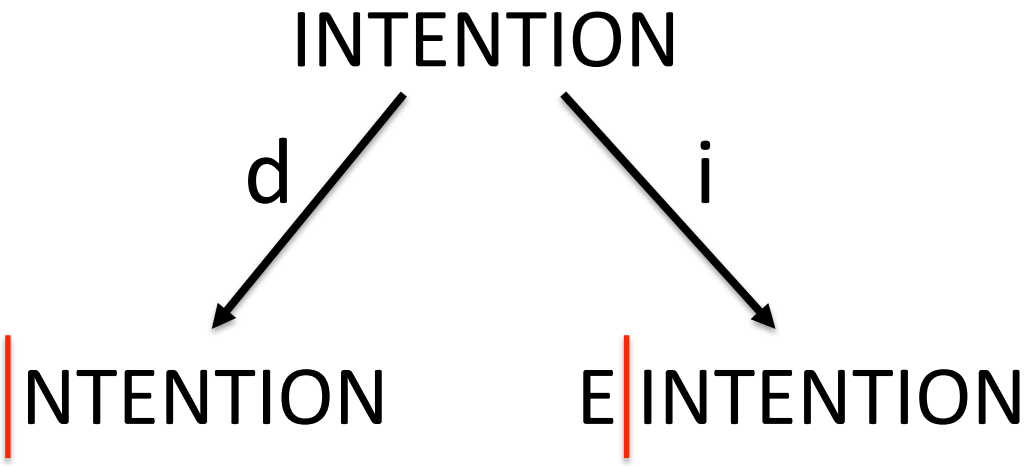
Input: S_1 = INTENTION, S_2 = EXECUTION

I	N	T	E	*	N	T	I	O	N
|	|	|	|	|	|	|	|	|	|
*	E	X	E	C	U	T	I	O	N
d	d	d		i	d				
	i	i			i				

Edit Cost: 8, assuming each operation costs 1

Computing Minimum Edit Distance

Strategy: Search for path (sequence of edits) from S_1 to S_2



Observation:

- Many distinct paths end up in the same state; and
- Combining optimal solutions to subproblems (matching the prefix) lead to an optimal solution to the problem.

Dynamic Programming (DP) for Minimum Edit Distance

- Two input strings
  S_1 of length n
  S_2 of length m

- Define D(i, j) be the minimum edit distance between S_1[1..i] and S_2[1..j]

- The minimum edit distance between S_1 and S_2 is thus D(n, m).

Dynamic Programming (DP) for Minimum Edit Distance

- **Base case:**

  $D(0, 0) = 0$

  $D(i, 0) = D(i-1, 0)$ + cost to delete $S_1[i]$ for $i = 1 .. n$

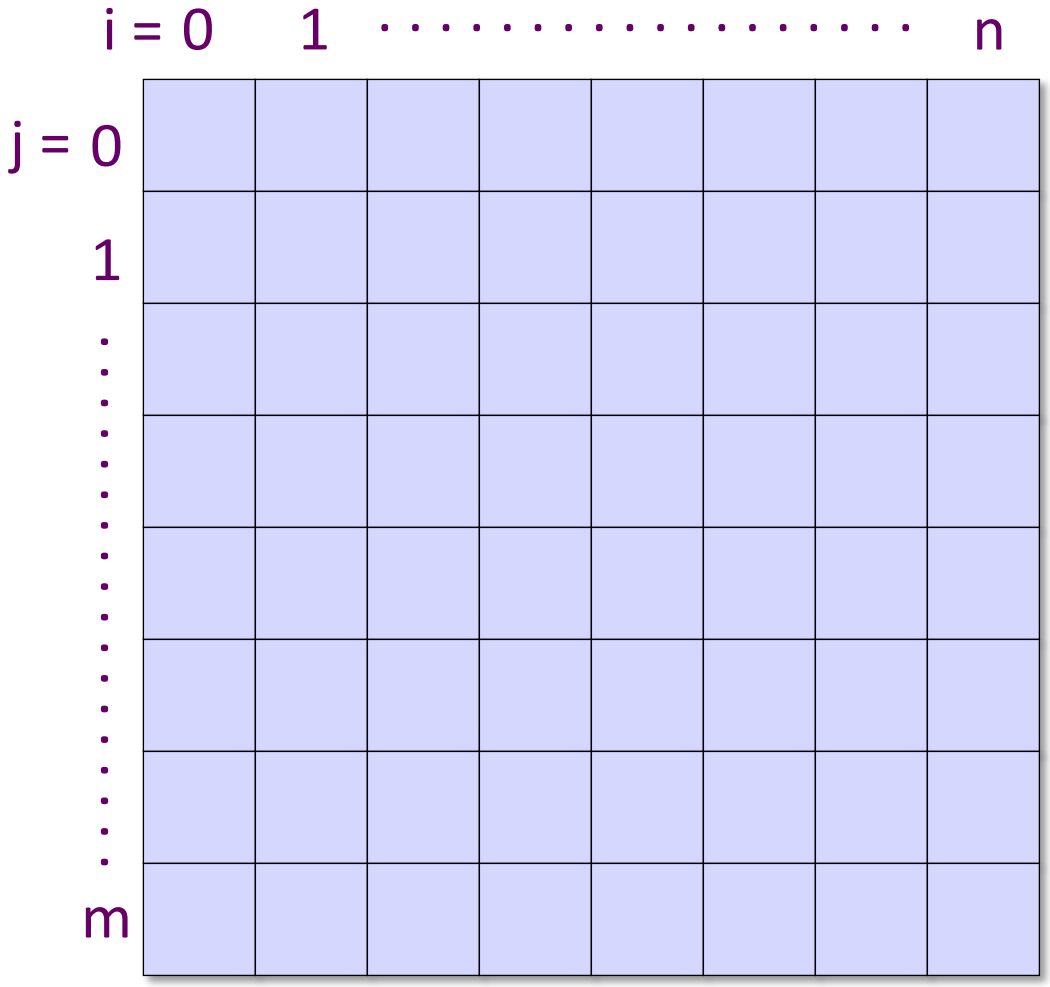
  $D(0, j) = D(0, j-1)$ + cost to insert $S_2[j]$ for $j = 1 .. m$

- **Recurrence Relation:**

$$D(i, j) = \begin{cases} \min \begin{cases} D(i-1, j-1) \\ D(i-1, j) + \text{cost to delete } S_1[i] \\ D(i, j-1) + \text{cost to insert } S_2[j] \end{cases} & S_1[i] = S_2[j] \\ \min \begin{cases} D(i-1, j) + \text{cost to delete } S_1[i] \\ D(i, j-1) + \text{cost to insert } S_2[j] \end{cases} & \text{otherwise.} \end{cases}$$

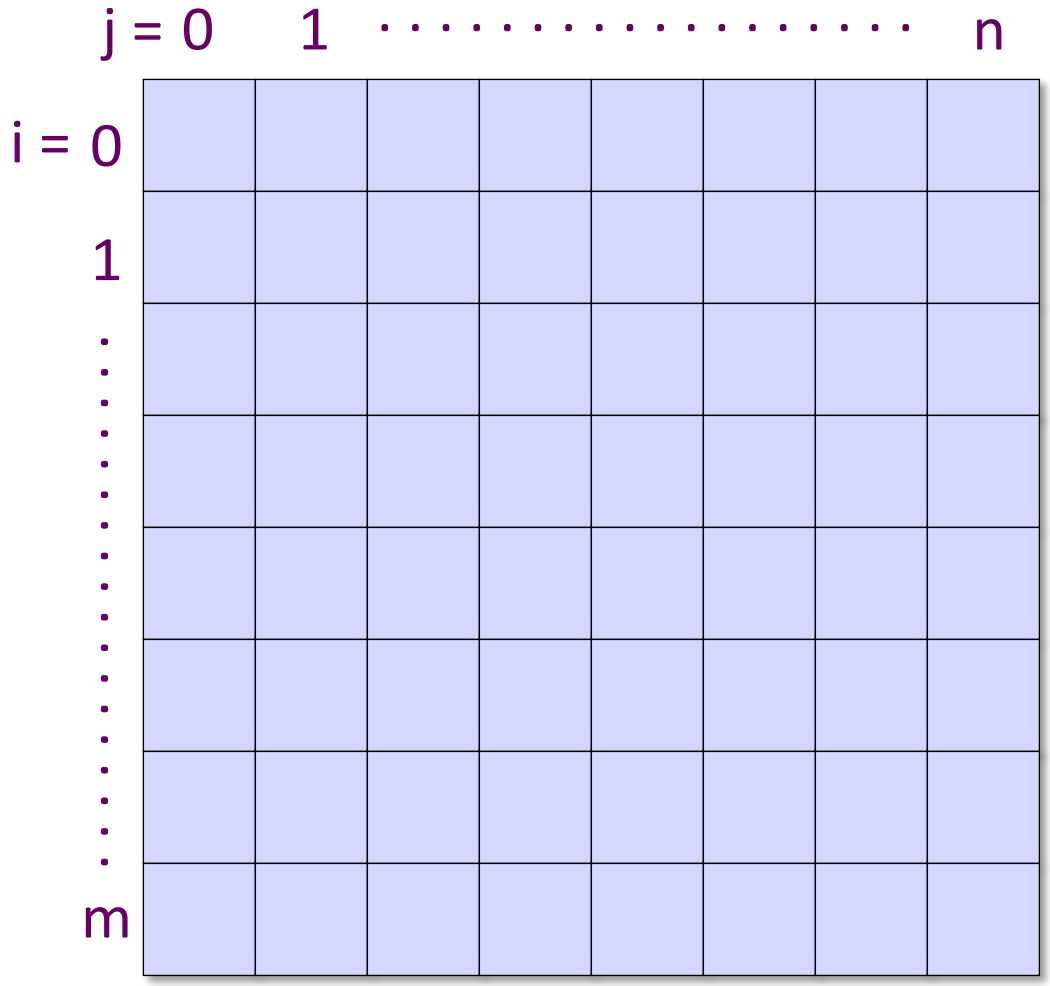
- **Termination:** $D(n, m)$

Dynamic Programming (DP) the Bottom-Up Approach

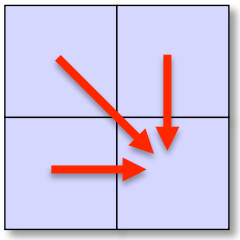


It takes $\Theta(nm)$ time to execute serially.

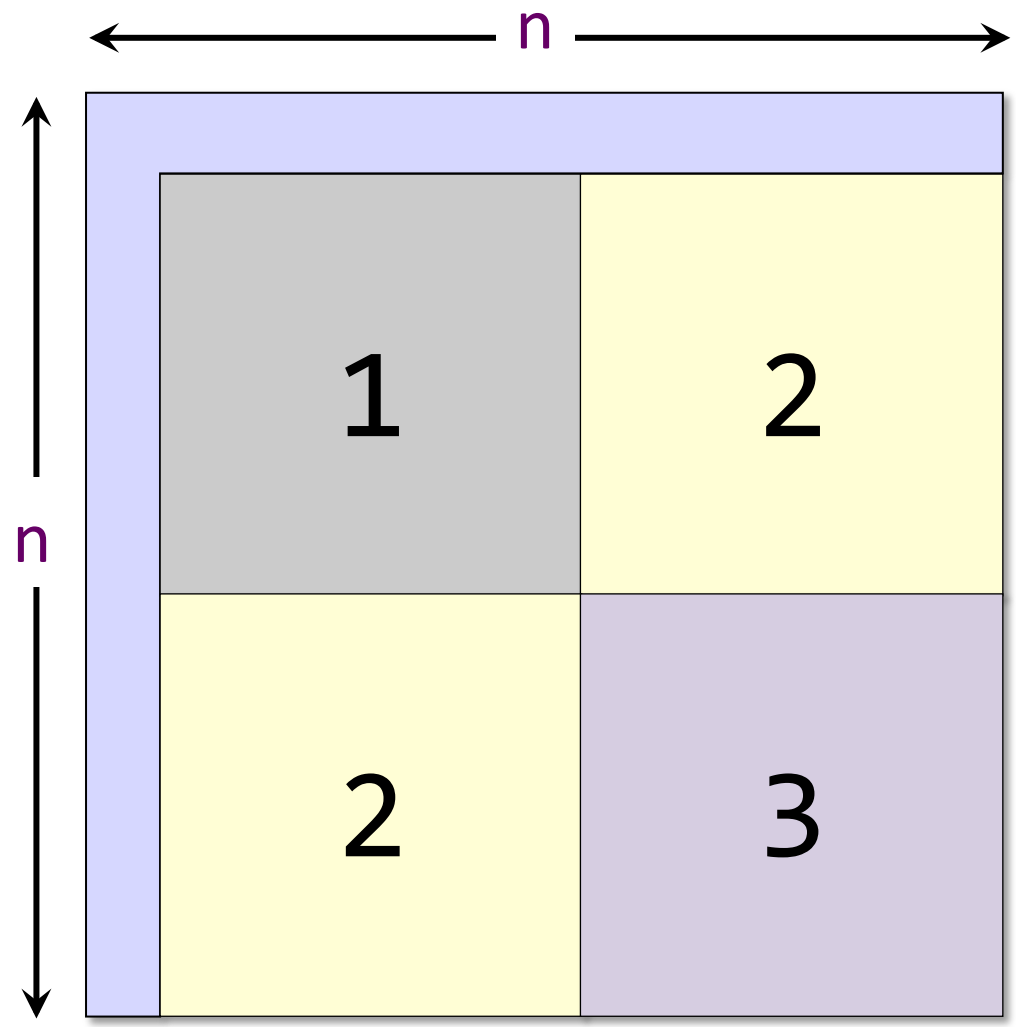
Parallelizing DP: Recursive Construction



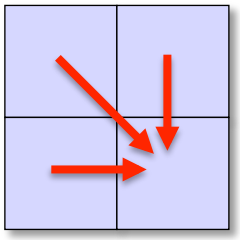
Dependency:



Parallelizing DP: Recursive Construction



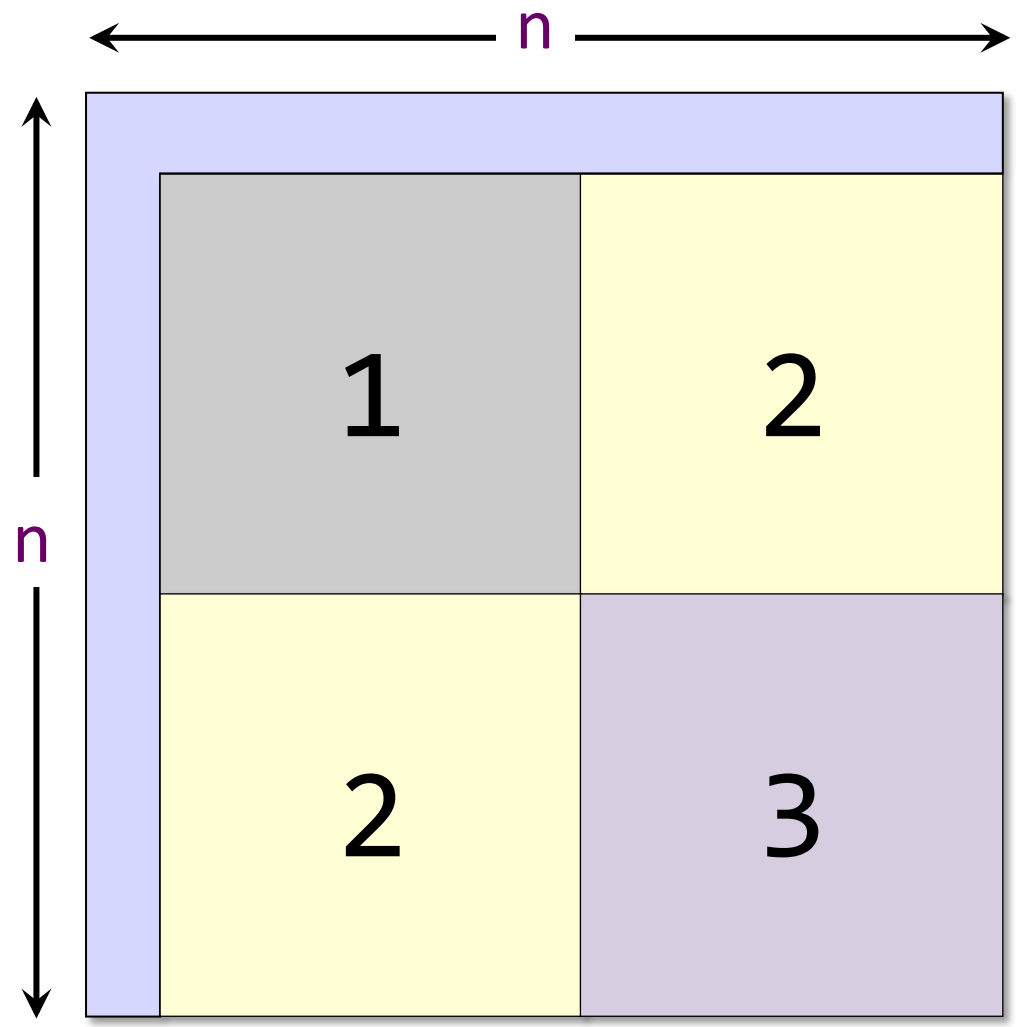
Dependency:



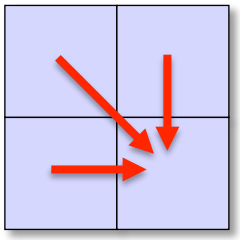
Example:

- Minimum edit distance
- Longest common subsequence
- Time warping

Parallelizing DP: Recursive Construction



Dependency:

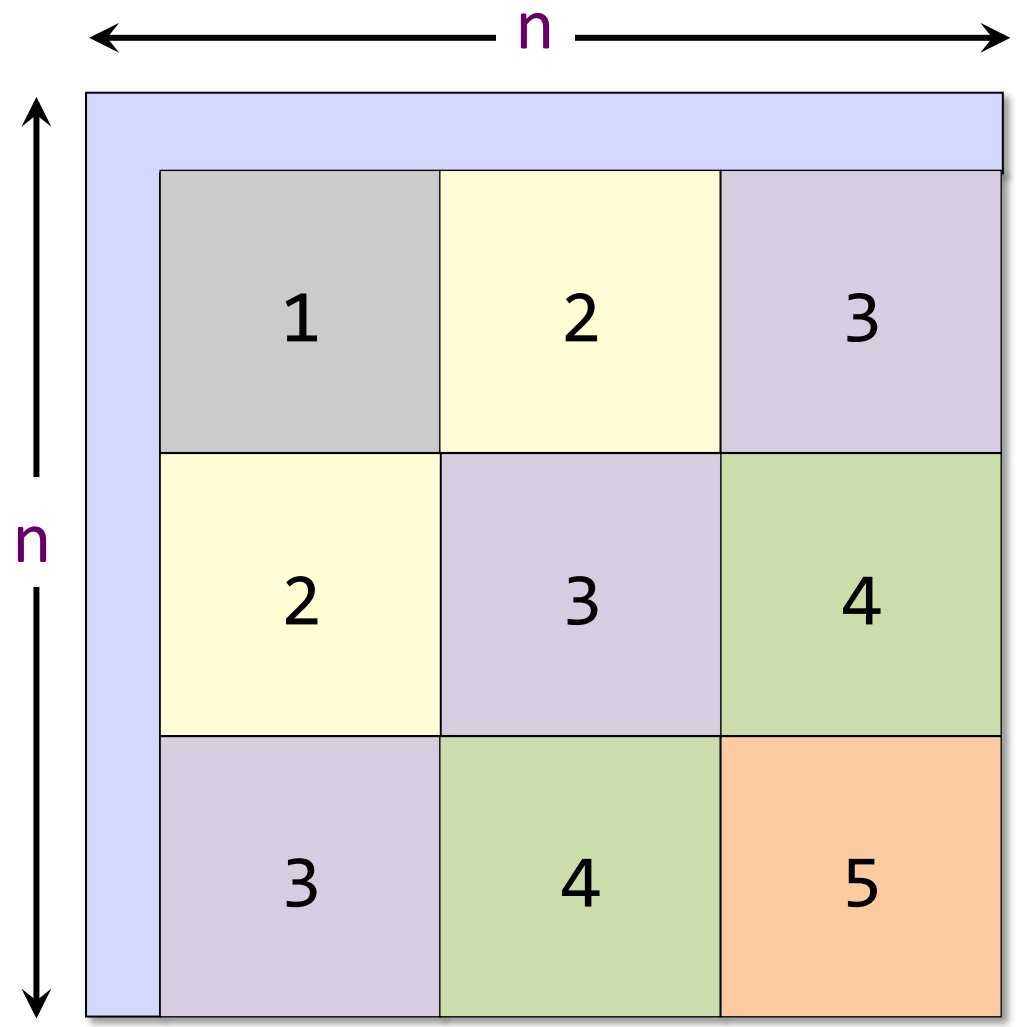


Parallelism

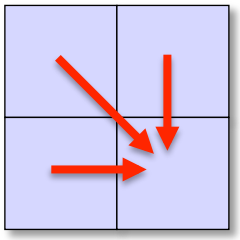
$= \Theta(n^{2-\lg 3})$

$= \Omega(n^{0.41})$

Parallelizing DP: Recursive Construction



Dependency:

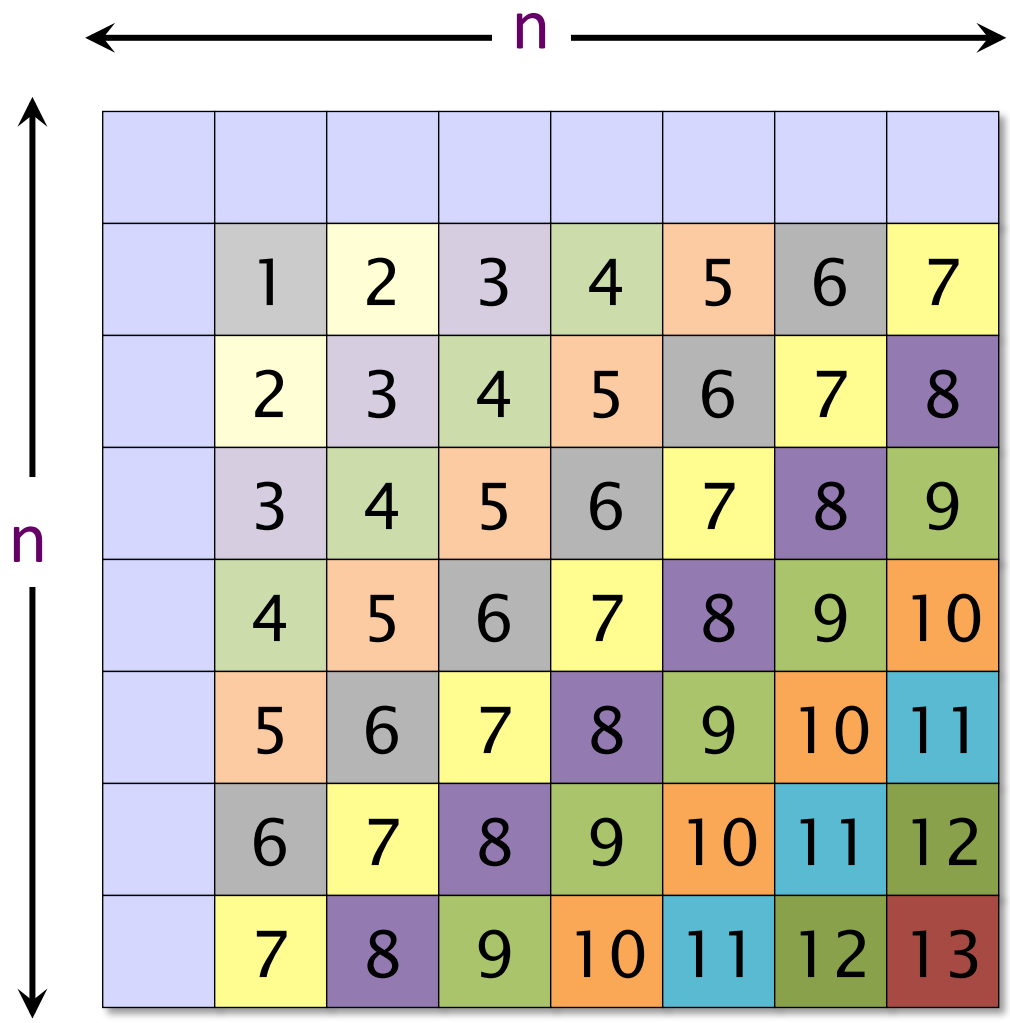


Parallelism

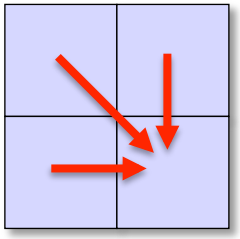
$= \Theta(n^{2-\log_3 5})$

$= \Omega(n^{0.53})$

Parallelizing DP: Wave Front



Dependency:



Parallelism

$= \Theta(n / \lg n)$

But we can do better still …

Course Logistics

- Course webpage:
  http://classes.cec.wustl.edu/~cse341/web/
- Piazza page:
  https://piazza.com/wustl/fall2014/cse341cse549/home
  - **Please accept the piazza invitation ASAP.**

Homeworks and Exams

- Homework 0 will go out on Wednesday and is due on Sept 08 (Monday) before class
- All other homeworks go out on Mondays and are due on Mondays --- the schedule is on the course webpage.
- There is one midterm and one final exam. The final exam is NOT during finals week, but during the last week of classes.

Grading Policy

- CSE 341T:
  - 40 points for homeworks (10 each)
  - 60 points for exams (30 each)
- CSE549T:
  - 20 points for homeworks (5 each)
  - 40 points for exams (20 each)
  - 30 points for final project
  - 10 points for in-class paper presentation.

Collaboration Policy

- Please read the policy posted on the webpage.