PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 8

Course Review

華中科技大學
HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 1
INTRODUCTION

华中科技大学
HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

*

TOPIC and Reference OF THIS COURSE

http://www.cs.cmu.edu/~15210/

https://www.diderot.one/courses/121/books

- This course covers
  - defining precisely the problem you want to solve,
  - learning the different algorithm-design techniques for solving problems,
  - designing abstract data types and the data structures that implement them
  - analyzing and comparing the cost of algorithms and data structures

SYNOPSIS

- **Parallelism**
- **Specification, Problem, and Implementation**

Exercise

- **Prove： when using a greedy schedule on a computation with W work and S span, the runtime will be at most:**

$$T \leq \frac{W}{P} + S\ .$$

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 2 SPARC

华中科技大学

HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

*

SYNOPSIS

- Functional Algorithms
- The Lambda Calculus
- The SPARC Language
  - Syntax and Semantics
  - Type System of SPARC
- Threads, Concurrency, and Parallelism
- Critical Sections and Mutual Exclusion

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 3
ALGORITHMIC ANALYSIS

华中科技大学
HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

SYNOPSIS

- Asymptotics
- Cost Models
- Recurrences

Exercise

1. Prove that for all k, $f(n)=n$ asymptotically dominates $g(n)=\ln^k n$.
2. Prove that asymptotic dominance is transitive.
3. Prove or disprove the following statement:
   if $g(n) \in O(f(n))$ and $g(n)$ is a finite function ($g(n)$ is finite for all n), then it follows that there exist constants $k1$ and $k2$ such that for all $n \geq 1$,
   $$g(n) \leq k_1 \cdot f(n) + k_2.$$
4. Describe the conditions under which a parallel algorithm would obtain near perfect speedups.

Exercise

5. For each of the following recurrences state whether it is leaf dominated, root dominated or balanced, and then solve the recurrence

$$
\begin{aligned}
W(n) &= 3W(n/2) + n \\
W(n) &= 2W(n/3) + n \\
W(n) &= 3W(n/3) + n \\
W(n) &= W(n-1) + n \\
W(n) &= \sqrt{n}W(\sqrt{n}) + n^2 \\
W(n) &= W(\sqrt{n}) + W(n/2) + n
\end{aligned}
$$

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 4 SEQUENCE

SYNOPSIS

- Defining Sequences
- The Sequence Abstract Data Type
- Array Sequence
- Cost Specification
- Examples
- Ephemeral and Single-Threaded Sequences

Excercise

1. **Design an algorithm that, for each element in a sequence of integers, finds the rightmost positive number to its left. If there is no positive element to the left of an element, the algorithm returns** $-\infty$ **for that element. For example, given the sequence** $\langle 1,0,-1,2,3,0,-5,7\rangle$**the algorithm would return** $\langle -\infty,1,1,1,2,3,3,3\rangle$.
2. **Give an example function** f**, a left identity** x**, and an input sequence** a **such that** iterate $f\ x\ a$ **and** reduce $f\ x\ a$ **return different results.**
3. **Given that** reduce **and** iterate **are equivalent for assocative functions, why would we use** reduce**?**
4. **Analyze the cost of the algorithm for generating 2D points [given above](#).**
5. **Present an algorithm that generates all contiguous subsequences of a given sequence.**
6. **Analyze the cost of your algorithm for Exercise [All contiguous subsequences](#).**
7. **Given sequences** a **of natural numbers and** b **of letters of the alphabet, we wish to compute the sequence that pairs each even element of** a **with all elements of** b **that are vowels.**
8. **To generate all composite numbers between** 2 **and** n**, prove that it suffices to consider all** $i \in \mathbb{N}$ ¡ $1 \le i \le \sqrt{n}$ **and all of** i**' s multiples up to** n/i.

華中科技大學

HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 5

Algorithm-Design

SYNOPSIS

- Basic Techniques
- Divide and Conquer
- Contraction
- Maximum Contiguous Subsequence Sum

Exercise

- The analysis given in the example above is not “tight” in the sense that there is a bound that is asymptotically dominated by $O(|a|\cdot|b|)$. Can you improve on the bound?
- Describe the changes to the algorithms Algorithm MCS: Brute Force and Algorithm MCSS: Brute Force to implement the strengthening described above. How does strengthening impact the work and span costs?
- Prove the MCSSE Extension lemma.

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 6

Randomized Algorithms

SYNOPSIS

- Introduction
- Order Statistics

Exercise

- Consider a game in which we draw some number of tasks at random such that a task has length n with probability 1/n and has length 1 otherwise. The expected length of a task is therefore bounded by 2. Imagine now drawing n tasks and waiting for all them to complete, assuming that each task can proceed in parallel independently of other tasks. Prove that the expected completion time is not constant.
- Prove that the pivot tree has O(lgn) height, and is therefore balanced, with high probability.

PARALLEL AND SEQUENTIAL ALGORITHMS AND DATA STRUCTURES

LECTURE 8

Graphs and Graph Search

Synopsis

- Graph and their Representation
- Graph Search
- Breadth-First Search

Exercises

- Prove that for a graph with n vertices and m edges, $O(\lg m) = O(\lg n)$
- What is the cost of deleting a vertex with out-degree d?
- What is the cost of finding the out-neighbors of a vertex?
- Why does the cost of mapping over all edges require $\Omega(n)$ work?
- Give a constant-span algorithm for deleting an edge from a graph.
- Give a constant-span algorithm for computing the complement of a graph.
- Does the algorithm visit all the vertices in the graph?
- Present a multi-source version of the single-source graph search algorithm.
- The computation of the new F is not quite defined in the same way as in the generic graph search. Prove that the technique used here is consistent with that of the generic algorithm.
- Prove that the algorithm is correct, i.e., visits all reachable vertices from the source in the order of their distances to the source
- Prove that the queue based implementation of sequential BFS is correct.

Exercises

- Prove that the queue based implementation of sequential BFS is correct.
- In general, from which frontiers could the vertices in $N_G(F_i)$ come when the graph is undirected? What if the graph is directed?

華中科技大學
HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Entry-level Problem 4: Fast Powering

```
fastPower(x, 16)
      ↓
fastPower(x, 8)
      ↓
fastPower(x, 4)
      ↓
fastPower(x, 2)
```

$$3^{1} * 3^{1} = 3^{2}$$

$$3^{2} * 3^{2} = 3^{4}$$

$$3^{4} * 3^{4} = 3^{8}$$

$$3^{8} * 3^{8} = 3^{16}$$

$$3^{20} \% 5 = (3^{10} * 3^{10}) \% 5$$

$$3^{20} \% 5 = ((3^{10} \% 5) * (3^{10} \% 5)) \% 5$$

设$\text{half} = 3^{10} \% 5$

$$3^{20} \% 5 = (\text{half} * \text{half}) \% 5$$

$$3^{21} \% 5 = (3^{20} * 3) \% 5$$

$$3^{21} \% 5 = ((3^{20} \% 5) * (3 \% 5)) \% 5$$

$$3^{21} \% 5 = (((\text{half} * \text{half}) \% 5) * (3 \% 5)) \% 5$$

Entry-level Problem 6: Fibonacci

1. 更大的斐波那契数列.

首先，这道题的数学基础是递推公式

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}F_n\\F_{n-1}\end{bmatrix}=\begin{bmatrix}F_n+F_{n-1}\\F_n\end{bmatrix}=\begin{bmatrix}F_{n+1}\\F_n\end{bmatrix}$$

于是我们有

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}F_2\\F_1\end{bmatrix}=\begin{bmatrix}F_3\\F_2\end{bmatrix}$$

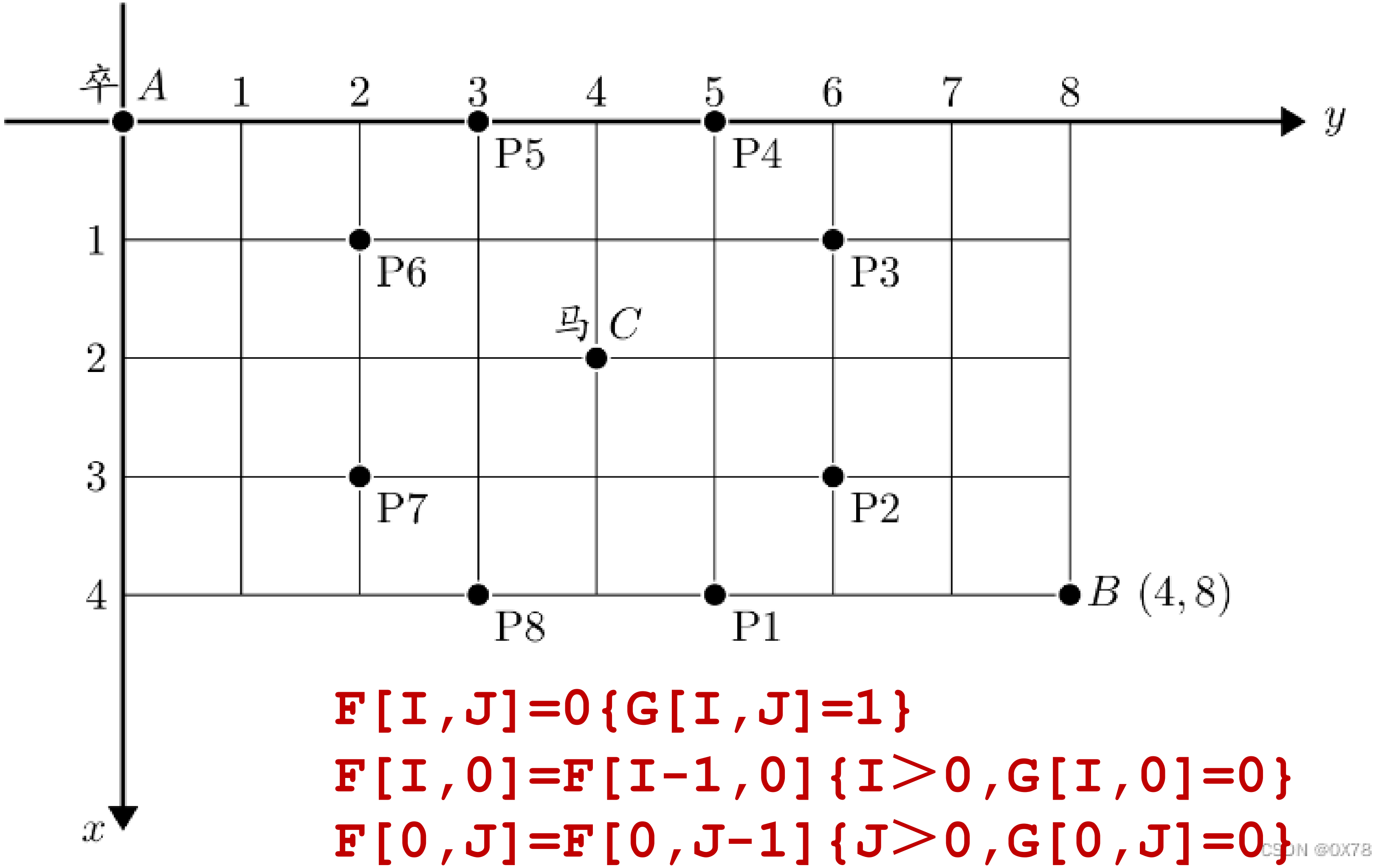
$$\begin{bmatrix}F_4\\F_3\end{bmatrix}=\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}F_3\\F_2\end{bmatrix}=\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\left(\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}F_2\\F_1\end{bmatrix}\right)=\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}^2\begin{bmatrix}F_2\\F_1\end{bmatrix}$$

递推下去，我们得到

$$\begin{bmatrix}1 & 1\\1 & 0\end{bmatrix}^{n-1}\begin{bmatrix}F_2\\F_1\end{bmatrix}=\begin{bmatrix}F_{n+1}\\F_n\end{bmatrix}$$

那么如何计算矩阵的幂呢？我们应该已经写过计算整数的高次幂的算法，只需要把传入的参数类型从无穷精度整数换成矩阵，并且把对应的整数乘法换成 (2×2) 矩阵乘法即可完成。

Entry-level Problem 20: Promoted Pawn



F[I,J]=0{G[I,J]=1}

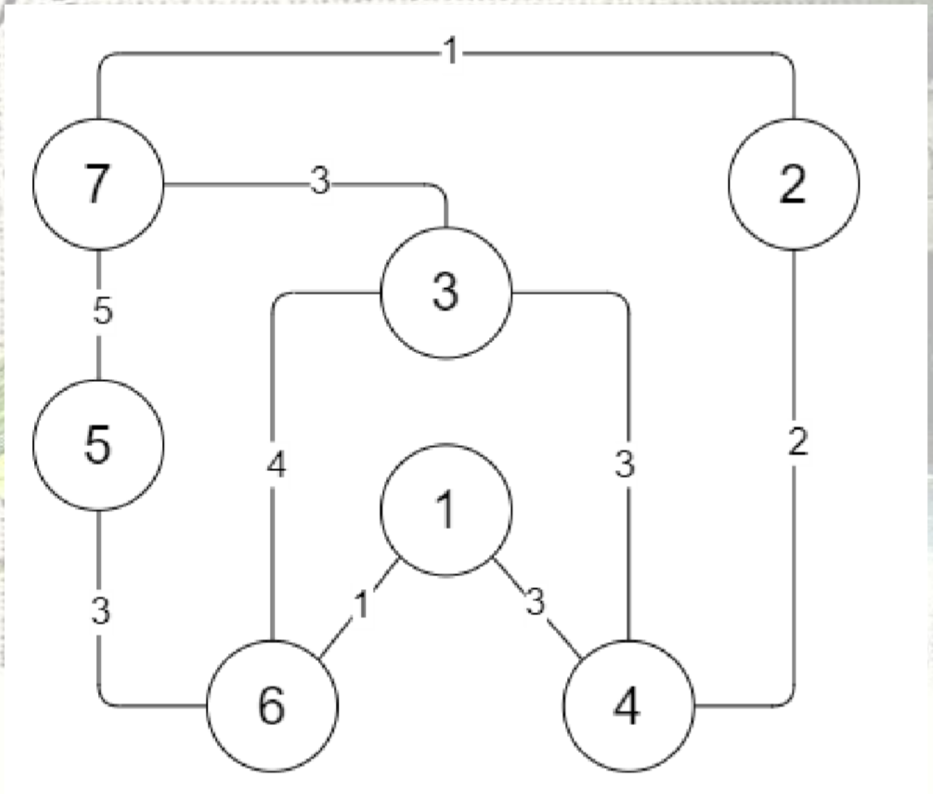
F[I,0]=F[I-1,0]{I>0,G[I,0]=0}

F[0,J]=F[0,J-1]{J>0,G[0,J]=0}

F[I,J]=F[I-1,J]+F[I,J-1]{I>0,J>0,G[I,J]=0}

Med-level Problem 2: Shortest-Path

- **Adjacency Matrix**



	0	1	2	3	4	5	6	7
0	∞	∞	∞	∞	∞	∞	∞	∞
1	∞	0	∞	∞	3	∞	1	∞
2	∞	∞	0	∞	2	∞	∞	1
3	∞	∞	∞	0	3	∞	4	3
4	∞	3	2	3	0	∞	∞	∞
5	∞	∞	∞	∞	∞	0	3	5
6	∞	1	∞	4	∞	3	0	∞
7	∞	∞	1	3	∞	5	∞	0

Med-level Problem 2: Shortest-Path

- **Dijkstra Algorithm**

因为题目明确给出各个边的权值非负，因此我们可以放心的使用 Dijkstra 最短路径算法，从源点不断向外扩张，直至能够以最短路径到达并标记到所有的节点，对于无法到达和标记到的节点，用默认的 -1 值来初始化即可。在给出伪代码之前，我们不妨证明一下该算法的正确性。

Dijkstra 算法主要由以下几步构成

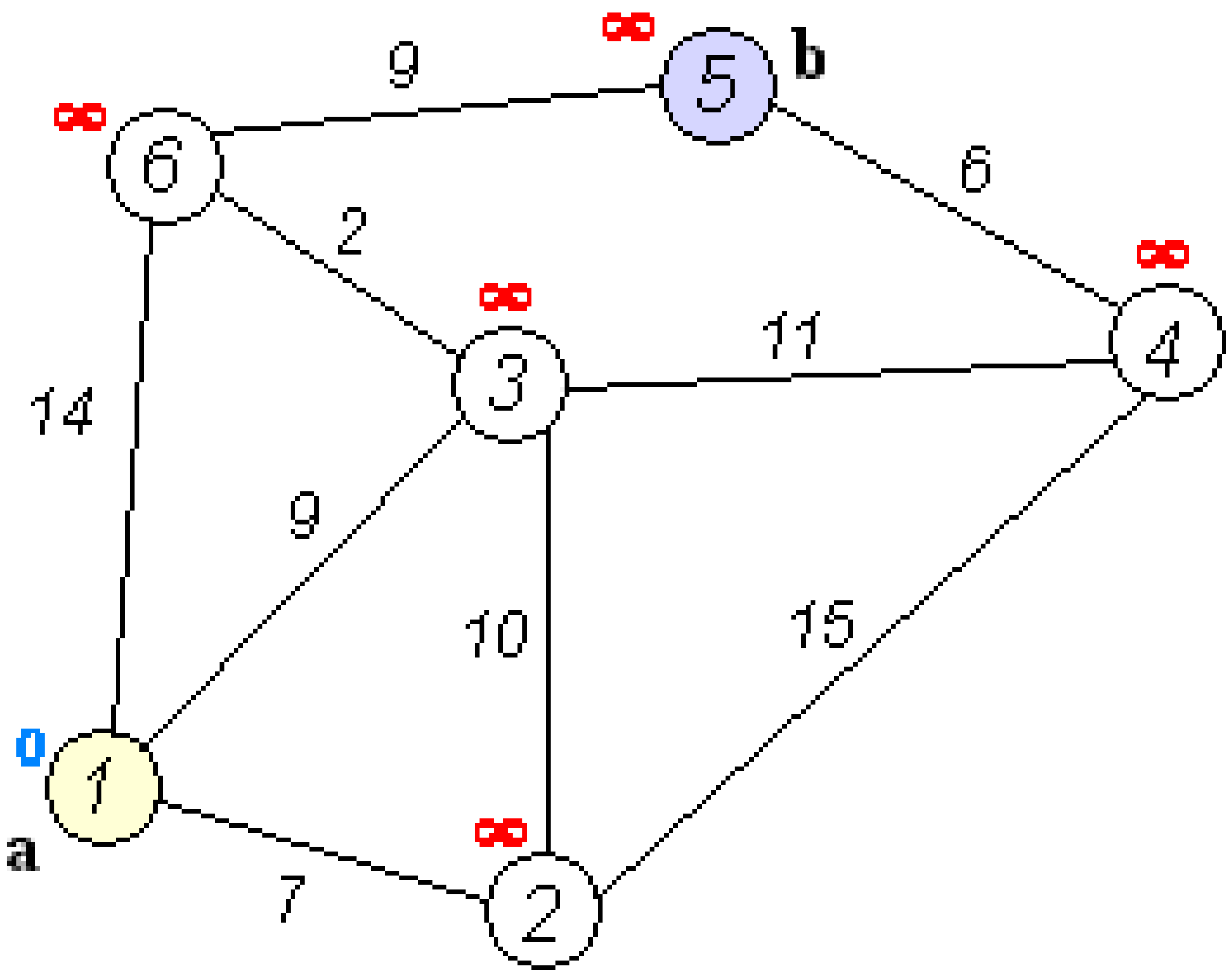
- 找出未标记过的离源点最近的点 (如果没有的话，便回退)
- 以该点为中心点更新该点周围的点 (更新方法为取最小值，一开始将所有点到源点的距离都初始化为无穷大)
- 一直执行，直到找到所有的点，将未更新过的点改为 -1，表示无法到达。

该算法主要基于一个性质：一个最短路径的任意子路都是最短路径。证明如下

证明. 假设最短路径 p 有一条子路径 q 不是自己起始点之间的最短路径，那么必然存在一条 $q < q'$，将 q 替换成 q'，得到一条新路径 $p' > p$，与 p 是最短路径矛盾，故该性质成立 $\square$

Med-level Problem 2: Shortest-Path

- **Dijkstra Algorithm**



Med-level Problem 3: Longest parenthesis distance

Algorithm 3: The Longest Distance

Data: S: 被处理过的序列 (即每一个原始数据前加上了下标形成 (i, x_i) 的形式 ts: 存储的左括弧值, m: 当前的符合要求的最大值)

Result: m: 最大括号距离

TheLongestDistance $S(\ \langle (0, x_1), (1, x_2), \cdots \rangle\ )=$

let

    match $(i, x), (ts, m)=$

    if $x = 0$ **then**

        $(i$ **++** $ts\ ,\ m)$

    else if $|ts| = 0$ **then**

        $([\,],\ m)$

    else

        let

            $h \leftarrow$ `hd` ts

            $ms \leftarrow \max(m, i - h + 1)$

            $rem \leftarrow$ `tl` ts

        in

            (rem, ms)

        end

    end

in

    `second` `reduce` match $([\,], 0)\ S$

end

Med-level Problem 4: Skyline

Algorithm 4: The skyline solving algorithm

Data: $B_s = \langle (l_1, r_1, h_1), (l_2, r_2, h_2), \cdots, (l_n, r_n, h_n) \rangle$

Result: $[(x_1, y_1), (x_2, y_2), \cdots]$

Skyline B_s =

if $|B_s| = 1$ **then**

    $[(l_1, h_1), (r_1, 0)]$

else

    let

        $\text{len} \leftarrow \frac{|Bs|}{2}$

        $B_l \leftarrow \text{Skyline } \langle (l_1, r_1, h_1), (l_2, r_2, h_2), \ldots, (l_{len}, r_{len}, h_{len}) \rangle$

        $B_r \leftarrow \text{Skyline } \langle (l_{len+1}, r_{len+1}, h_{len+1}), \ldots, (l_n, r_n, h_n) \rangle$

    in

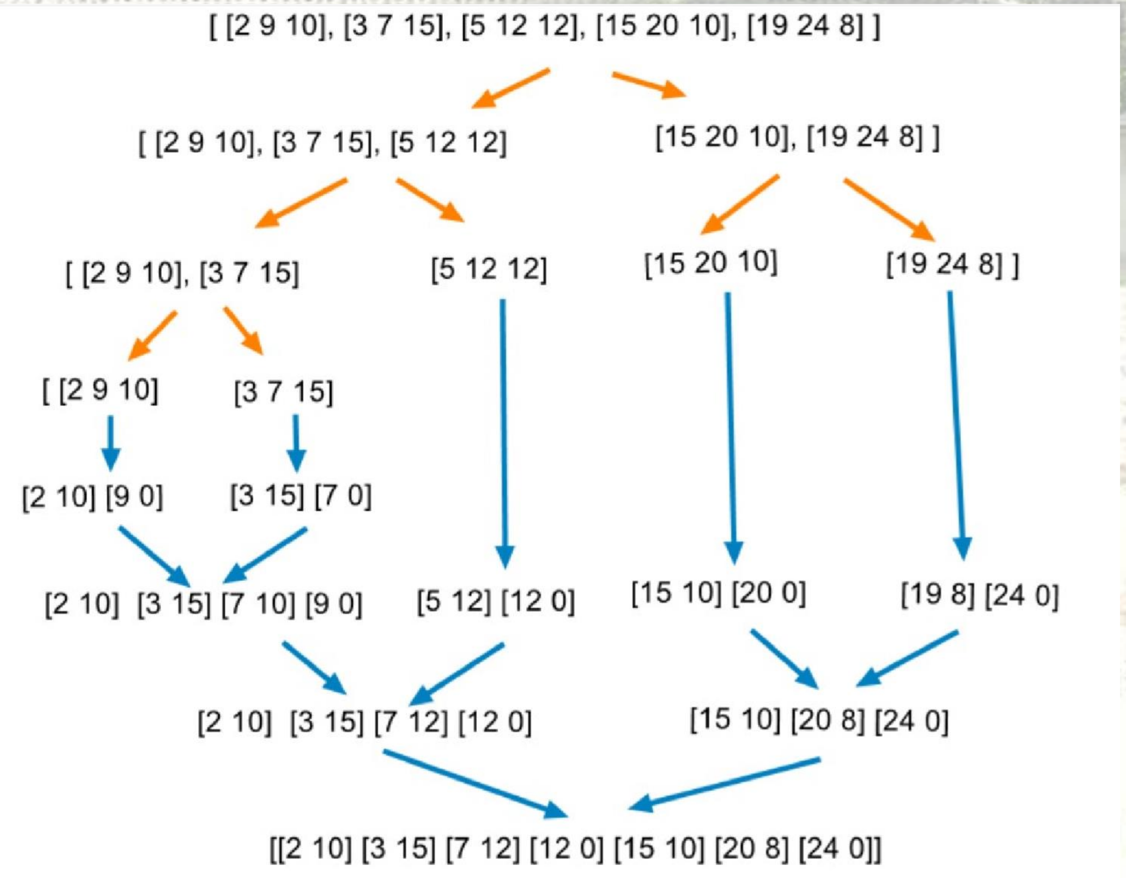
        `merge` B_l B_r

    end

end

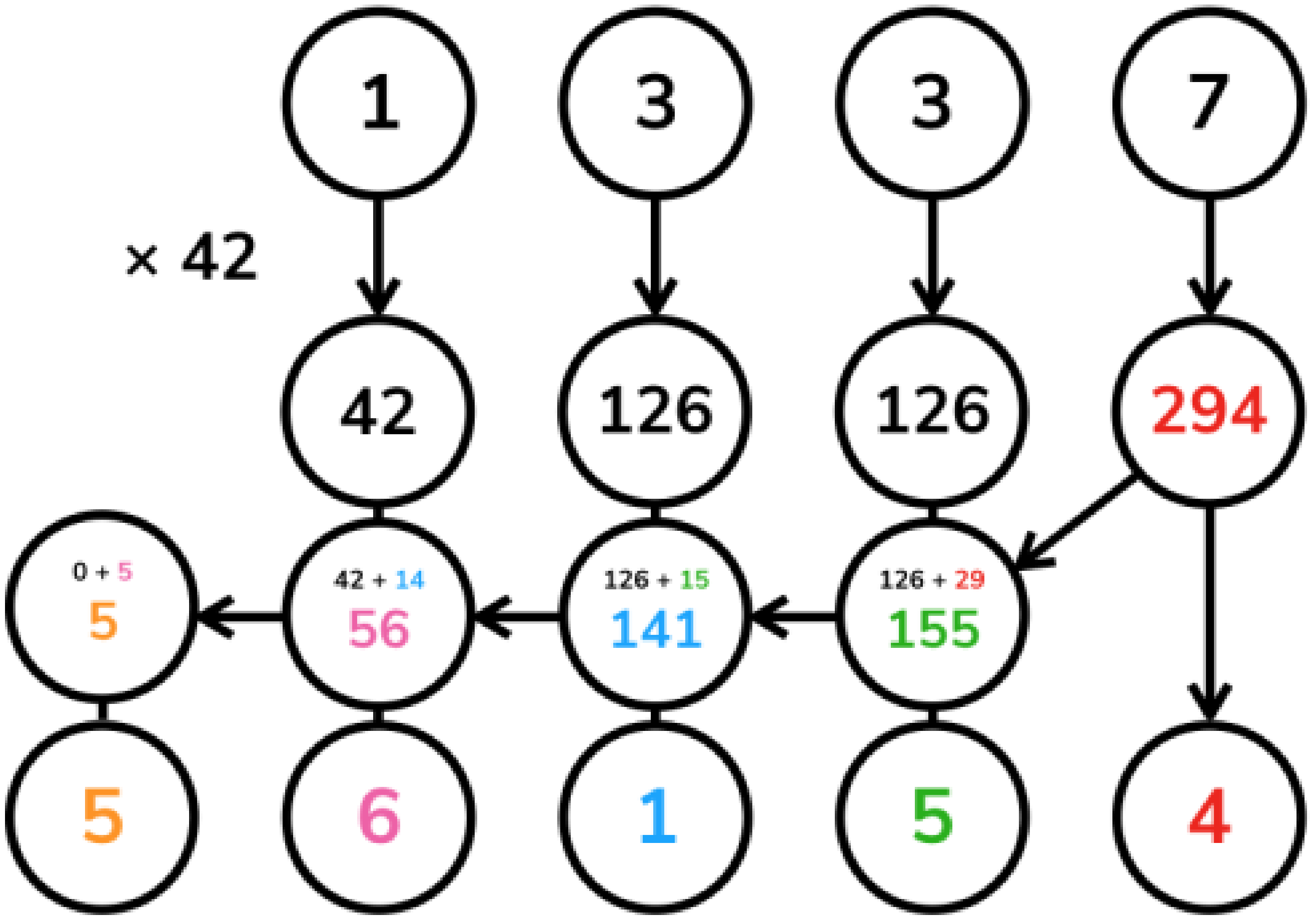
Med-level Problem 4: Skyline

- **How to merge?**



Med-level Problem 6: Arbitrary-Precision Arithmetic

- BigNum x Integer
  - (1337 x 42)

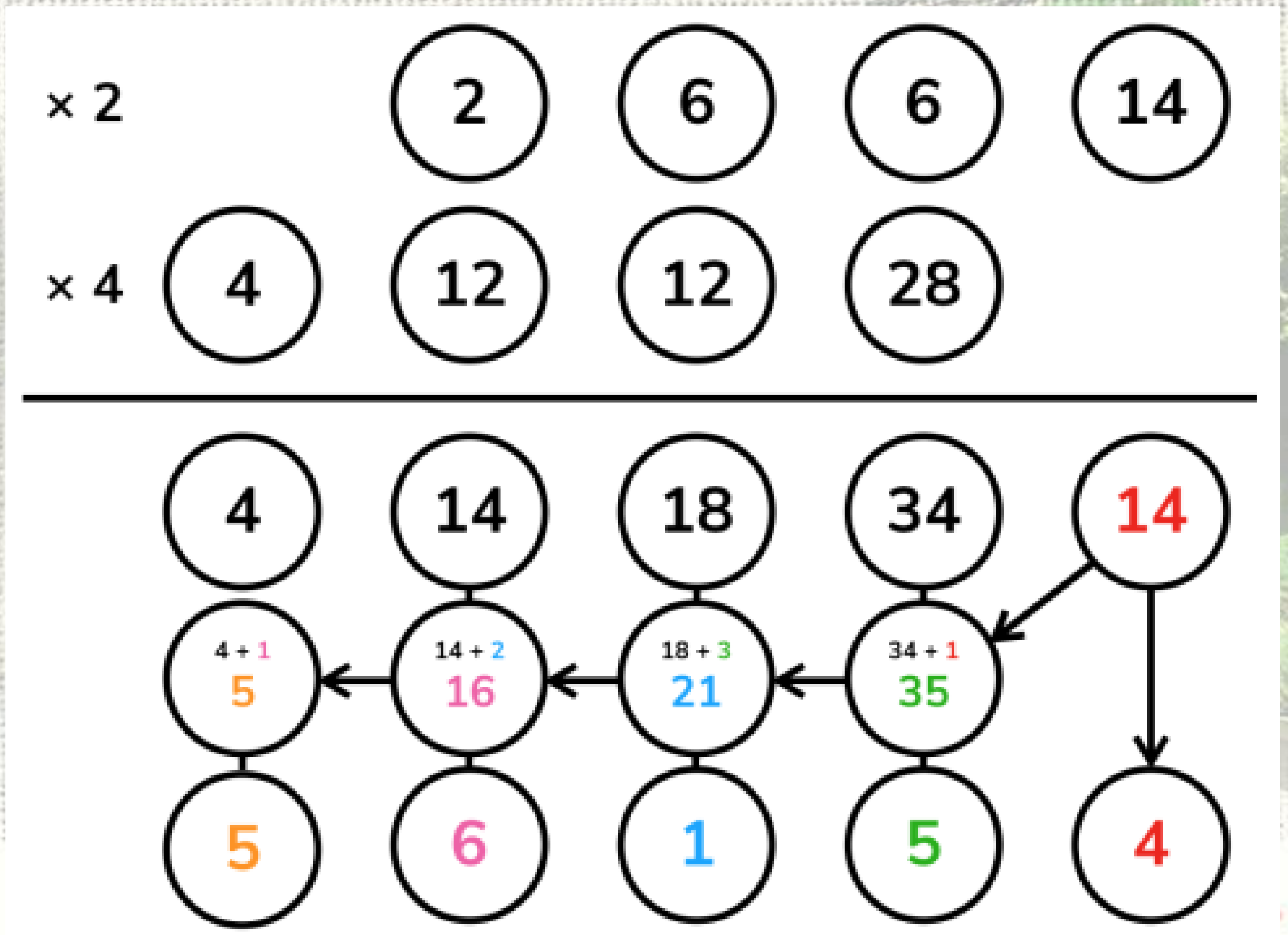


Med-level Problem 6: Arbitrary-Precision Arithmetic

- **BigNum x BigNum**

  1337 x 42

  = 10 x (1337 x 4) + 1337 x 2



Med-level Problem 7: Cut-Vertex & Cut-Edge

- **Tarjan Algorithm**

Algorithm 9: Tarjan 算法

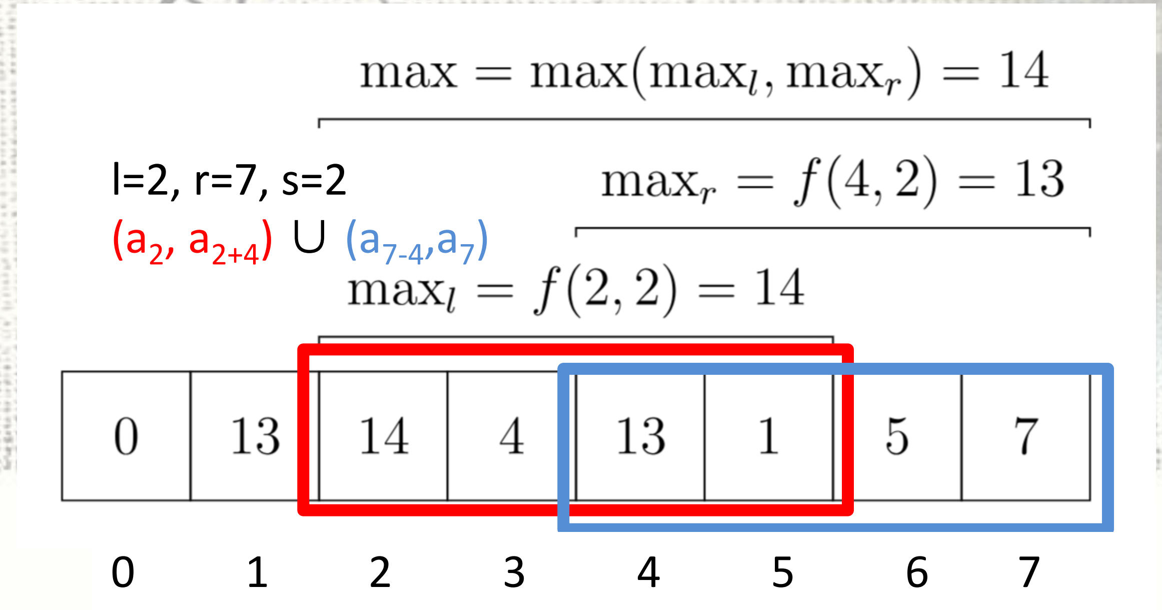
Data: u: 节点，v：u 的子节点 (即存在一条边使得 u 能到达 v)

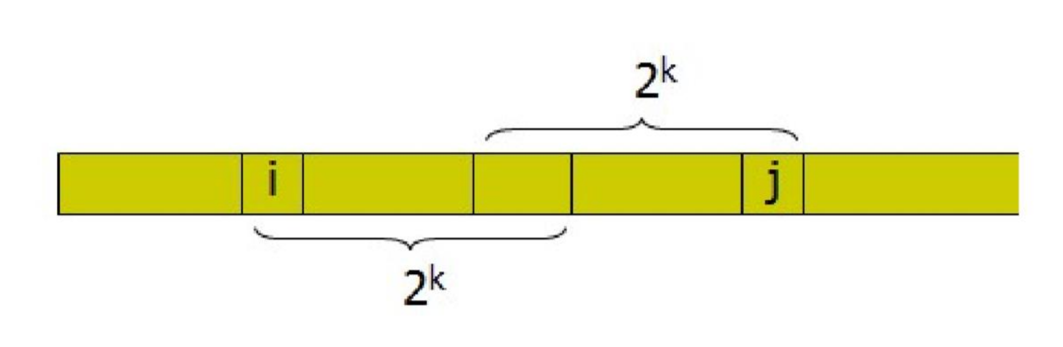
Result: ce: 割边, cn: 割点

```
Tarjan u=
Initialization :   dfn[u] = low[u] = ++Index
Stack.push(u)
foreach (u, v) ∈ E do
    if v is not visited then
        Tarjan(v)
        low[u]=min(low[u],low[v])
    else
        low[u]=min(low[u],low[v])
    end
    /* 接下来开始判断割边和割点                    */
    if low[v] = dfn[u] then
        cn ← cn + 1 if u is not counted
    end
    if low[v] > dfn[u] then
        cn ← cn + 1 if u is not counted
        ce ← ce + 1
    end
end
```

Med-level Problem 8: Non-update Range Min/Max Querying

- **Sparse Table**
- **Exists an integer *s*, such that arbitrary interval (l, r) can be written as (l, l + 2^s-1) ∪ (r - 2^s+1, r)**





Med-level Problem 8:
Non-update Range Min/Max Querying

Algorithm 10: RMQ 的 ST 解决算法

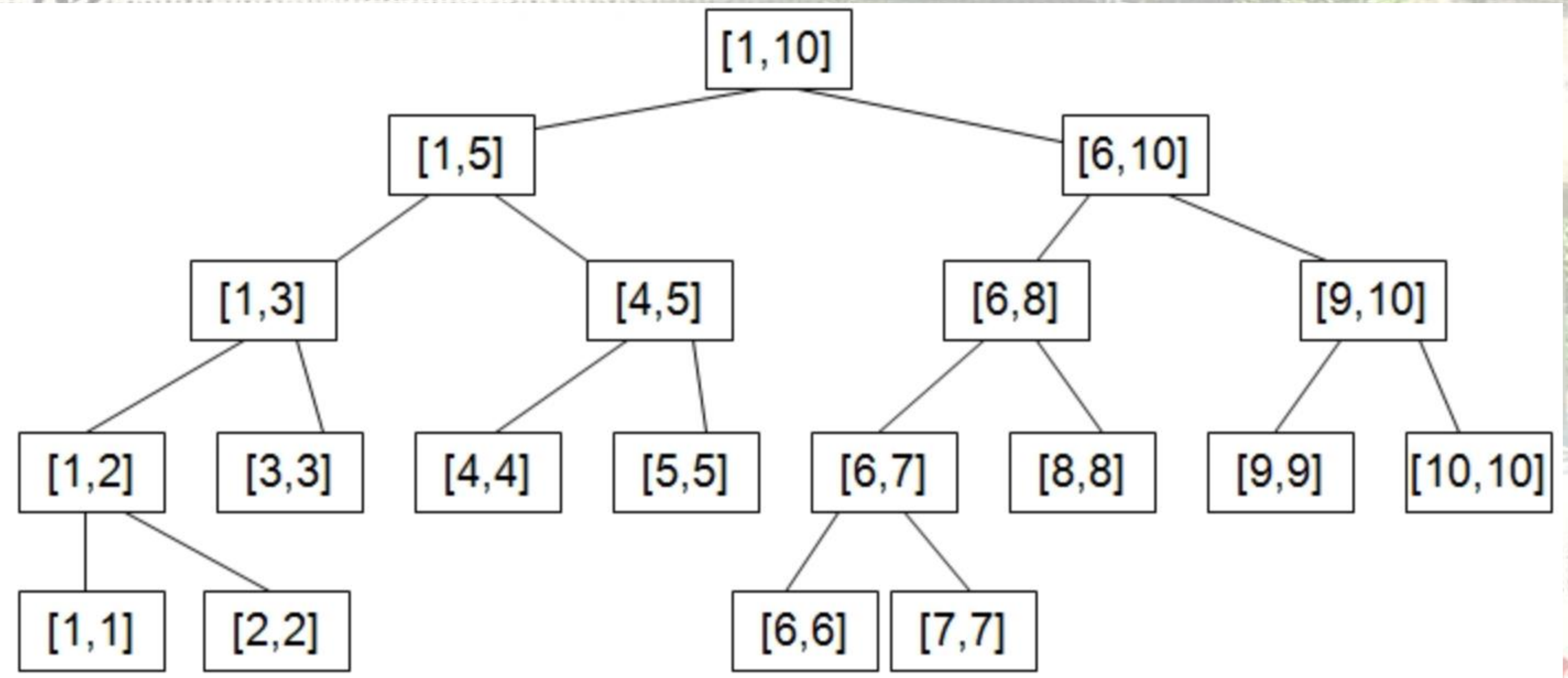
Data: $F[i][j]$ 表示在 sequence$\langle a_0, a_1 \cdots, a_n\rangle$ 中从 i 开始长度为 2^j 的最大值，其中 $F[i][0]$ 就是 a_i 本身,N 表示输入的串 S 的长度,S' 表示查询的串，其元素为 (x_i, y_i) 表示查询从 x 到 y 的最大值

Result: 一系列查询到的值

preprocess S=
while $i < N$ **do**
    for j **to** $i + 2^j - 1 \leq N$ **do**
        $F[i][j] \leftarrow \max\{F[i][j-1], F[i+2^{j-1}][j-1]\}$
    end
end
querySequence S'=
let
    query (x,y)=
    let
        $k \leftarrow \lceil \log_2(y - x + 1) \rceil$
    in
        $\max\{F[x][k], F[y - 2^k + 1][k]\}$
    end
in
    map query S'
end

Med-level Problem 8: Non-update RMQ

- **Segment Tree**



Med-level Problem 9: Prime Query

- Fermat's Little Theorem (Th.1)

$$a^p \equiv a(\bmod\ p),\quad p\ \text{是一个素数}$$

- Select several base numbers ***a*** **<** ***p*****, if** a^p **mod** ***p*** **=** ***a*****, then** ***p*** **might be a prime number**
- **Quadratic Probing (Th. 2)**

如果 p 是一个素数，$0 < x < p$，则方程 $x^2 \equiv 1(\bmod\ p)$ 的解为 $x = 1$ 或 $x = p - 1$

Med-level Problem 9: Prime Query

- **Miller-Rabin Algorithm**

Algorithm 11: Miller-Rabin 算法

Data: n: 待检测数 $n \geq 2$,(s, t): $n - 1 = t \cdot 2^s$

Result: bool 值表明是否为素数

Miller-Rabin a n=

if $n = 2$ **then**

    True

else if $(x < 2) \vee (x\ is\ even)$ **then**

    False

else

    let

        $at \leftarrow a^t \bmod n$

    in

        twicejudge(s, at)

    end

end

试题类型

- 是非题（约6*3）
- 简答题（约6*5）
- 复杂度计算（约2*6）
- 计算证明题（约1*12）
- 算法题（约1*12）
- 实验题（约1*16）