COMP108
Data Structures and Algorithms

Dynamic Programming (Part II Assembly Line Scheduling)

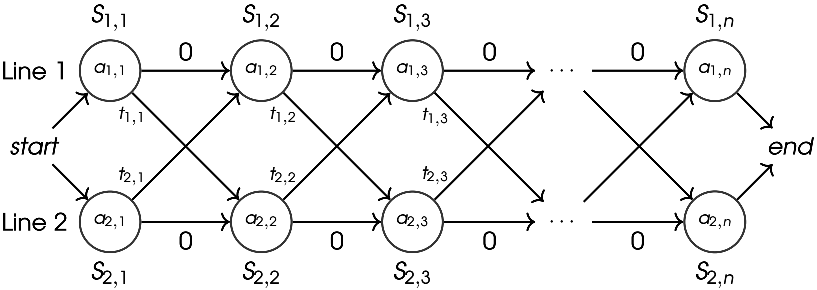
Professor Prudence Wong

pwong@liverpool.ac.uk

2020-21

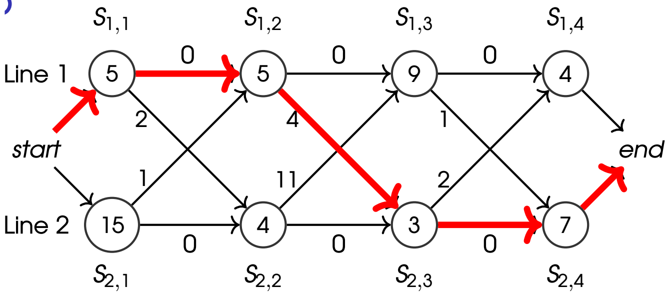
Assembly Line Scheduling

- 2 assembly lines, each with n stations ($S_{i,j}$: line i station j)
- $S_{1,j}$ and $S_{2,j}$ perform same task but time taken is different
- $a_{i,j}$: assembly time at $S_{i,j}$
- $t_{i,j}$: transfer time from $S_{i,j}$



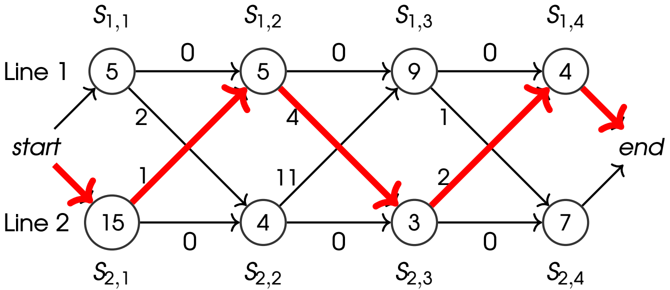
Problem: To determine which stations to go in order to **minimize** the total time through the n stations

Example (1)



station chosen	$S_{1,1}$	$S_{1,2}$		$S_{2,3}$	$S_{2,4}$	
time required	5	5	4	3	7	= 24

Example (2)



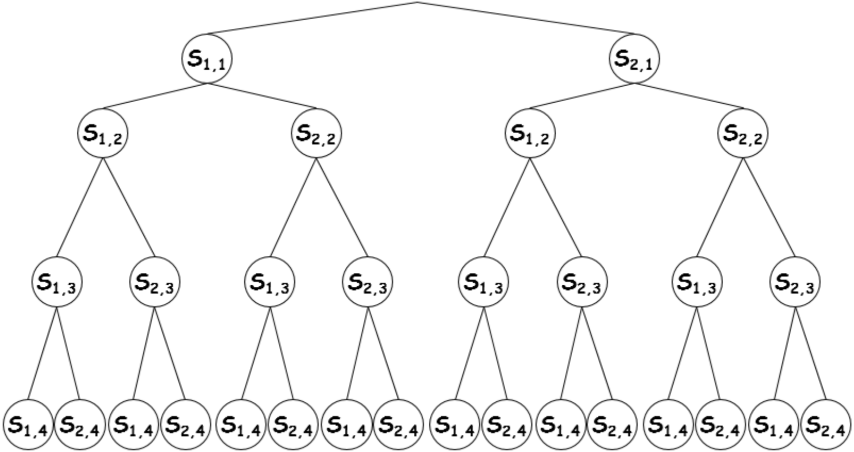
station chosen	$S_{2,1}$		$S_{1,2}$		$S_{2,3}$		$S_{1,4}$	
time required	15	1	5	4	3	2	4	= 34

How to determine the best stations to go?

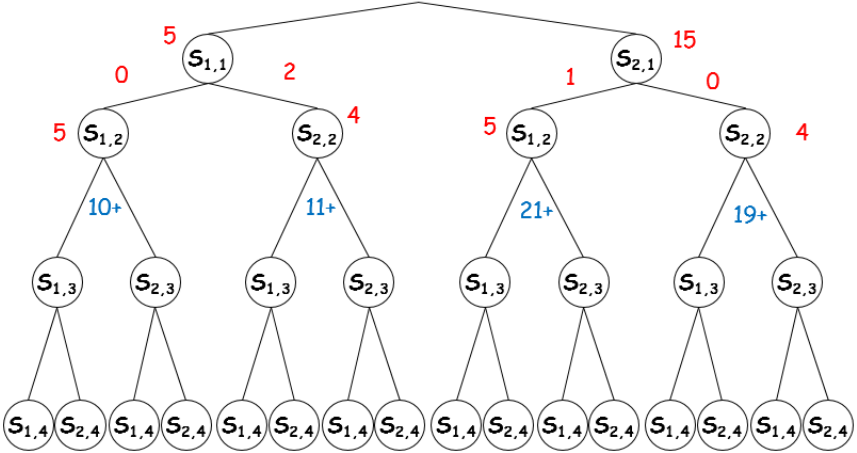
There are altogether 2^n choices of stations.

Should we try them all?

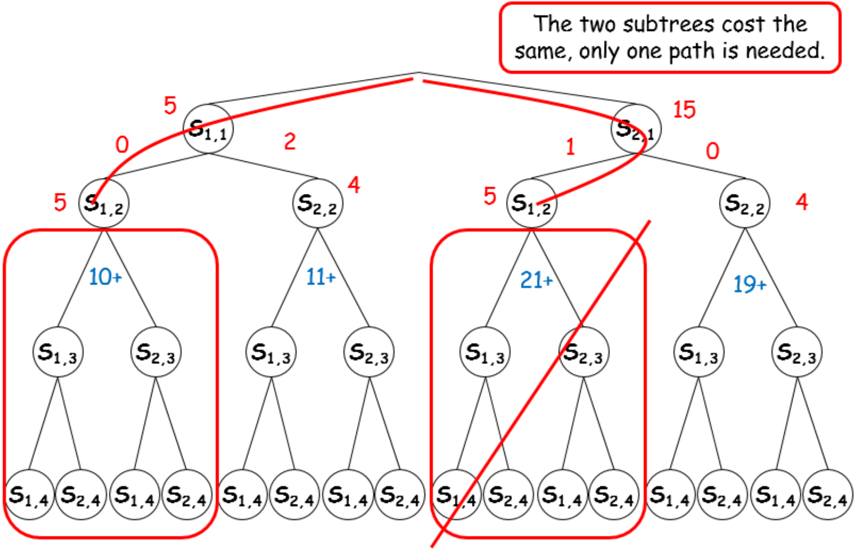
All possible choices



All possible choices



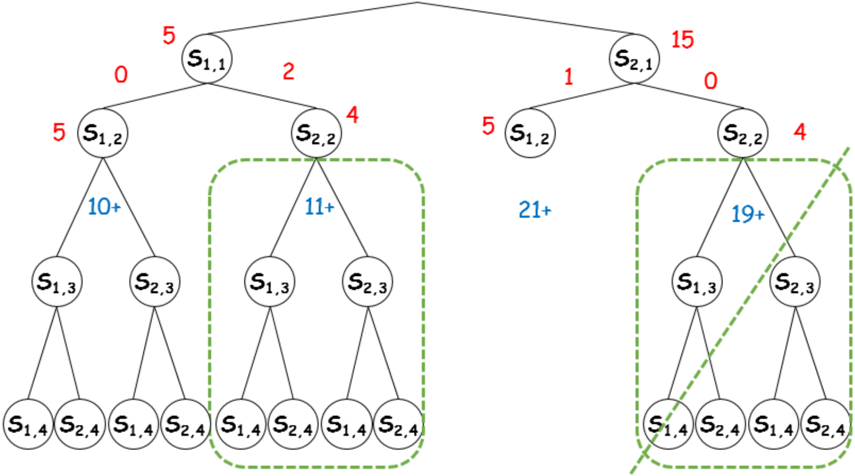
All possible choices



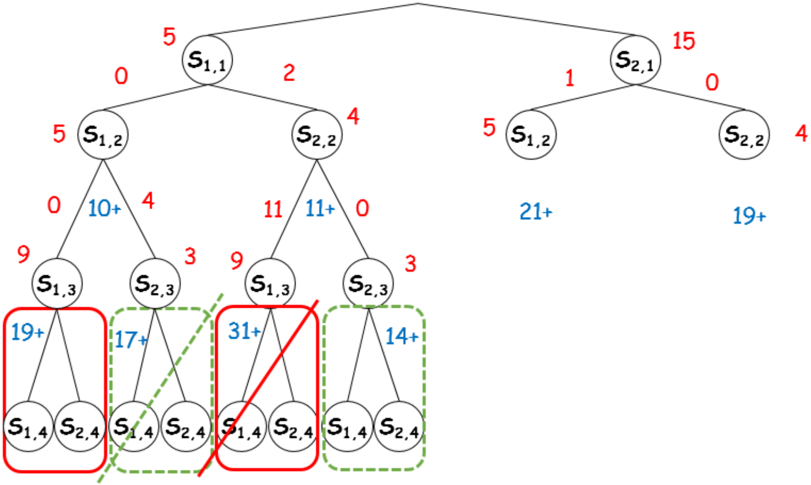
The two subtrees cost the same, only one path is needed.

All possible choices

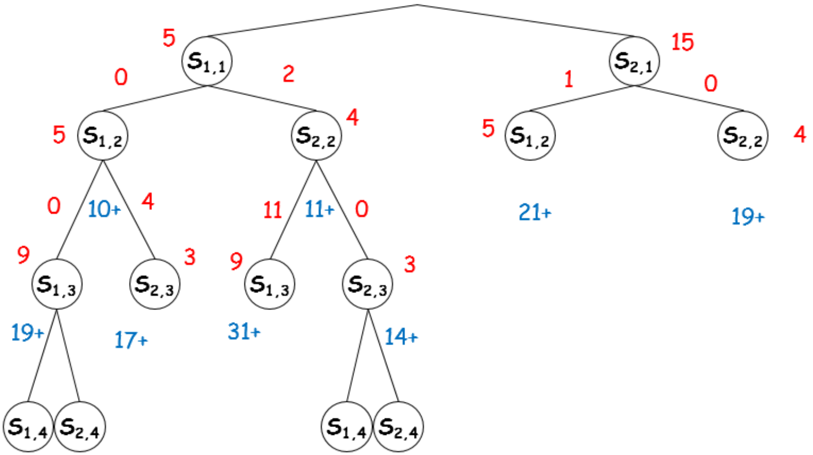
Similarly, ...



All possible choices



All possible choices



Good news: Dynamic Programming

- We don′t need to try all possible choices.
- We can make use of **dynamic programming**:
  1. If we know the fastest ways to get thro′ station $S_{1,n}$ and $S_{2,n}$
     $\Longrightarrow$ the faster of these two is overall fastest
  2. Fastest ways to get thro′ $S_{1,n}$?
     need to know fastest way to get thro′ $S_{1,n-1}$ and $S_{2,n-1}$
  3. Similarly for $S_{2,n}$
  4. Generalsing, we want fastest way to get thro′ $S_{1,j}$ and $S_{2,j}$, for all j.

A dynamic programming solution - formalisation

What are the sub-problems?

- given j, what is the fastest way to get thro' $S_{1,j}$
- given j, what is the fastest way to get thro' $S_{2,j}$

Definitions:

- $\mathbf{f_1[j]}$: the fastest time to get thro' $S_{1,j}$
- $\mathbf{f_2[j]}$: the fastest time to get thro' $S_{2,j}$

The final solution equals to $\mathbf{min\{f_1[n], f_2[n]\}}$

Task:

- Starting from $f_1[1]$ and $f_2[1]$, compute $f_1[j]$ and $f_2[j]$ incrementally
- i.e., $f_1[2] \& f_2[2]$, $f_1[3] \& f_2[3]$, $\cdots$, $f_1[n] \& f_2[n]$

Solving the sub-problems (1)

Q1: What is the fastest time to get thro' $S_{1,j}$

A: either

- the fastest way thro' $S_{1,j-1}$, then directly to $S_{1,j}$, or
- the fastest way thro' $S_{2,j-1}$, a transfer from line 2 to line 1, and then thro' $S_{1,j}$

$$\text{(i) } f_1[j-1] + a_{1,j} \qquad \text{(ii) } f_2[j-1] + t_{2,j-1} + a_{1,j}$$

$$\therefore f_1[j] = \min\{f_1[j-1] + a_{1,j}, \quad f_2[j-1] + t_{2,j-1} + a_{1,j}\}$$

Boundary case: $f_1[1] = a_{1,1}$

Solving the sub-problems (2)

Q1: What is the fastest time to get thro' $S_{2,j}$

A: either

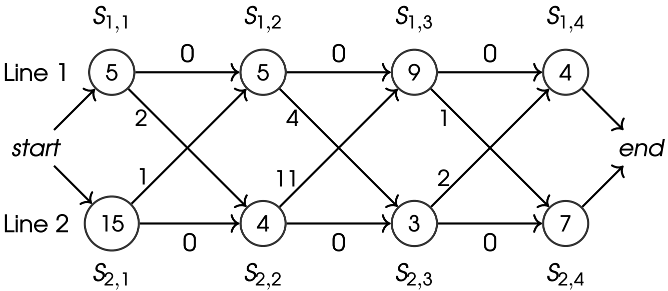
- the fastest way thro' $S_{1,j-1}$, a transfer from line 2 to line 1, and then thro' $S_{2,j}$, or
- the fastest way thro' $S_{2,j-1}$, then directly to $S_{2,j}$

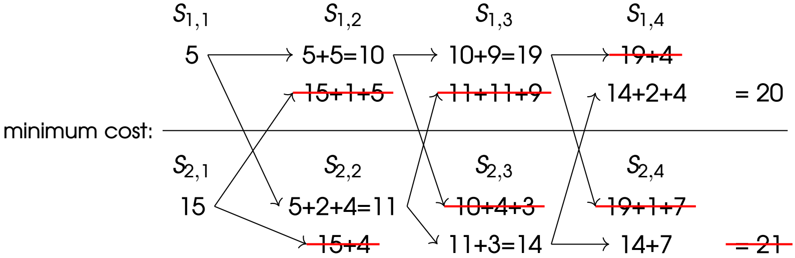
$$\text{(i) } f_1[j-1] + t_{1,j-1} + a_{2,j} \qquad \text{(ii) } f_2[j-1] + a_{2,j}$$

$$\therefore f_2[j] = \min\{f_1[j-1] + t_{1,j-1} + a_{2,j}, \quad f_2[j-1] + a_{2,j}\}$$

Boundary case: $f_2[1] = a_{2,1}$

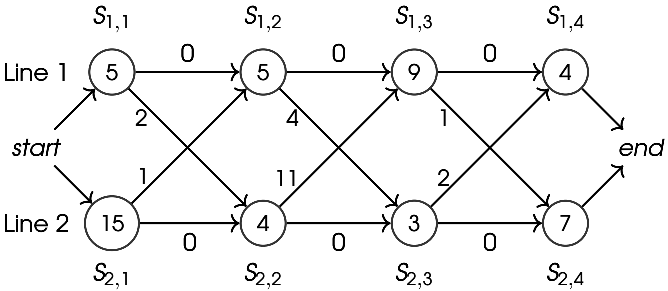
Example again

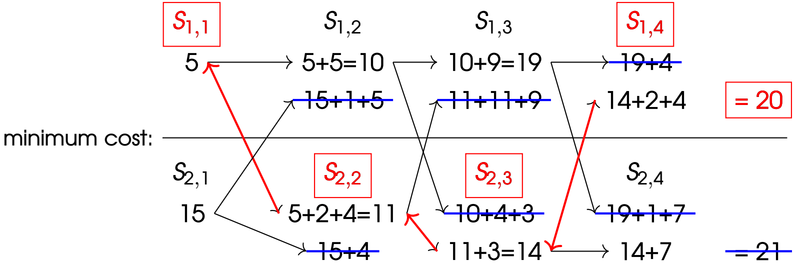




	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$	$S_{1,4}$	
	5	5+5=10	10+9=19	~~19+4~~	
		~~15+1+5~~	~~11+11+9~~	14+2+4	= 20
minimum cost:					
	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$	$S_{2,4}$	
	15	5+2+4=11	~~10+4+3~~	~~19+1+7~~	
		~~15+4~~	11+3=14	14+7	~~= 21~~

Example again - $S_{1,1}$, $S_{2,2}$, $S_{2,3}$, $S_{1,4}$





$\boxed{S_{1,1}}$ $S_{1,2}$ $S_{1,3}$ $\boxed{S_{1,4}}$

5 → 5+5=10 → 10+9=19 → ~~19+4~~

~~15+1+5~~ ~~11+11+9~~ 14+2+4 $\boxed{= 20}$

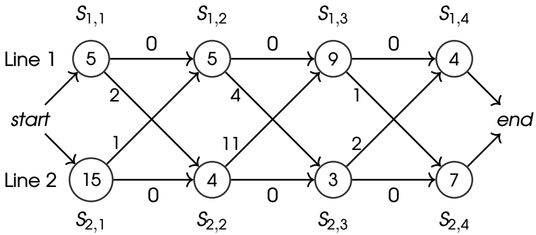
minimum cost:

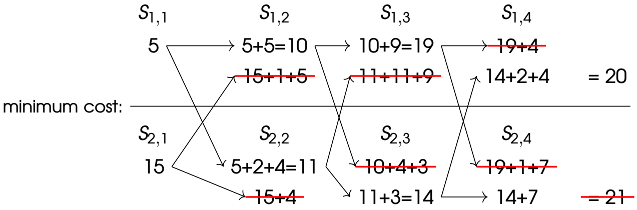
$S_{2,1}$ $\boxed{S_{2,2}}$ $\boxed{S_{2,3}}$ $S_{2,4}$

15 5+2+4=11 ~~10+4+3~~ ~~19+1+7~~

~~15+4~~ 11+3=14 → 14+7 ~~= 21~~

Summary on the Example





$S_{1,1}$ $S_{1,2}$ $S_{1,3}$ $S_{1,4}$

5 → 5+5=10 → 10+9=19 → ~~19+4~~

~~15+1+5~~ ~~11+11+9~~ 14+2+4 = 20

minimum cost:

$S_{2,1}$ $S_{2,2}$ $S_{2,3}$ $S_{2,4}$

15 5+2+4=11 ~~10+4+3~~ ~~19+1+7~~

~~15+4~~ 11+3=14 → 14+7 ~~= 21~~

j	$f_1[j]$	$f_2[j]$
1	5	15
2	10	11
3	19	14
4	20	21

$$f_1[j] = \begin{cases} a_{1,1} & \text{if } j = 1 \\ \min\{f_1[j-1] + a_{1,j}, f_2[j-1] + t_{2,j-1} + a_{1,j}\} & \text{if } j > 1 \end{cases}$$

$$f_2[j] = \begin{cases} a_{2,1} & \text{if } j = 1 \\ \min\{\min\{f_1[j-1] + t_{1,j-1} + a_{2,j}, f_2[j-1] + a_{2,j}\} & \text{if } j > 1 \end{cases}$$

$$f^* = \min\{f_1[n], f_2[n]\}$$

Pseudo code

```
$f_1[1] \leftarrow a_{1,1}$
$f_2[1] \leftarrow a_{2,1}$
for $j \leftarrow 2$ to $n$ do
begin
    $f_1[j] \leftarrow \min\{f_1[j-1] + a_{1,j}, f_2[j-1] + t_{2,j-1} + a_{1,j}\}$
    $f_2[j] \leftarrow \min\{f_2[j-1] + a_{2,j}, f_1[j-1] + t_{1,j-1} + a_{2,j}\}$
end
$f^* \leftarrow \min\{f_1[n], f_2[n]\}$
```

What about 3 assembly lines?

Summary

Summary: Dynamic Programming for Assembly Line Scheduling

Next: Revision Lecture

For note taking