FROM

MODERN ALGORITHMS WORKSHOP

Parallel Algorithms

Prof. Charles E. Leiserson

Dr. Tao B. Schardl

September 19, 2018

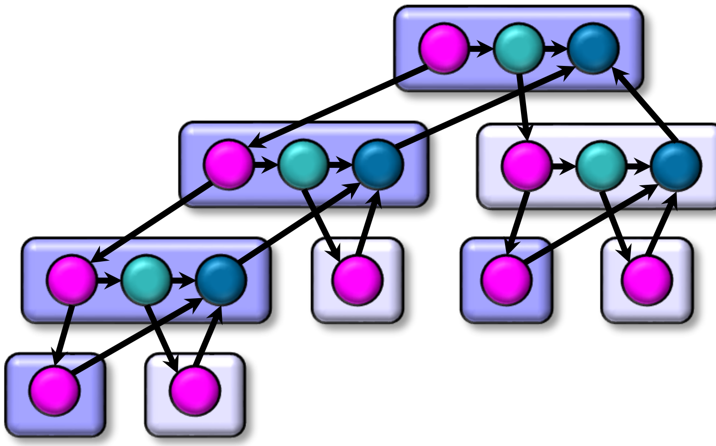
Outline

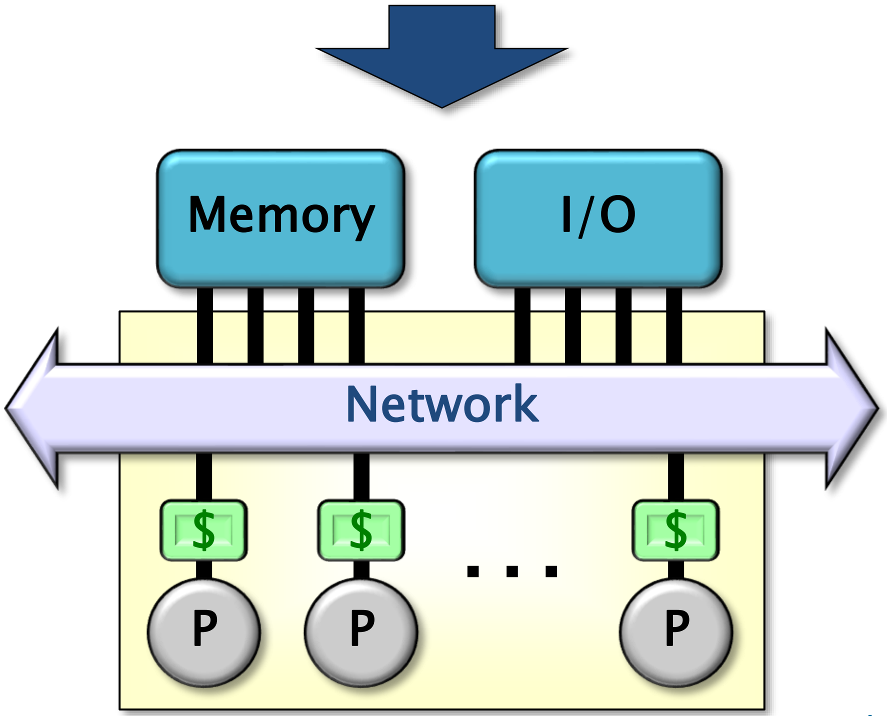
- Introduction
- Cilk Model
- Detecting Nondeterminism
- What Is Parallelism?
- Scheduling Theory Primer
- *Lunch Break*
- Analysis of Parallel Loops
- Case Study: Matrix Multiplication
- Case Study: Jaccard Similarity
- Post-Moore Software

Scheduling Theory Primer

Scheduling

- Cilk allows the programmer to express potential parallelism in an application.
- The Cilk scheduler maps strands onto processors dynamically at runtime.
- Since the theory of distributed schedulers is complicated, we'll explore the ideas with a centralized scheduler.

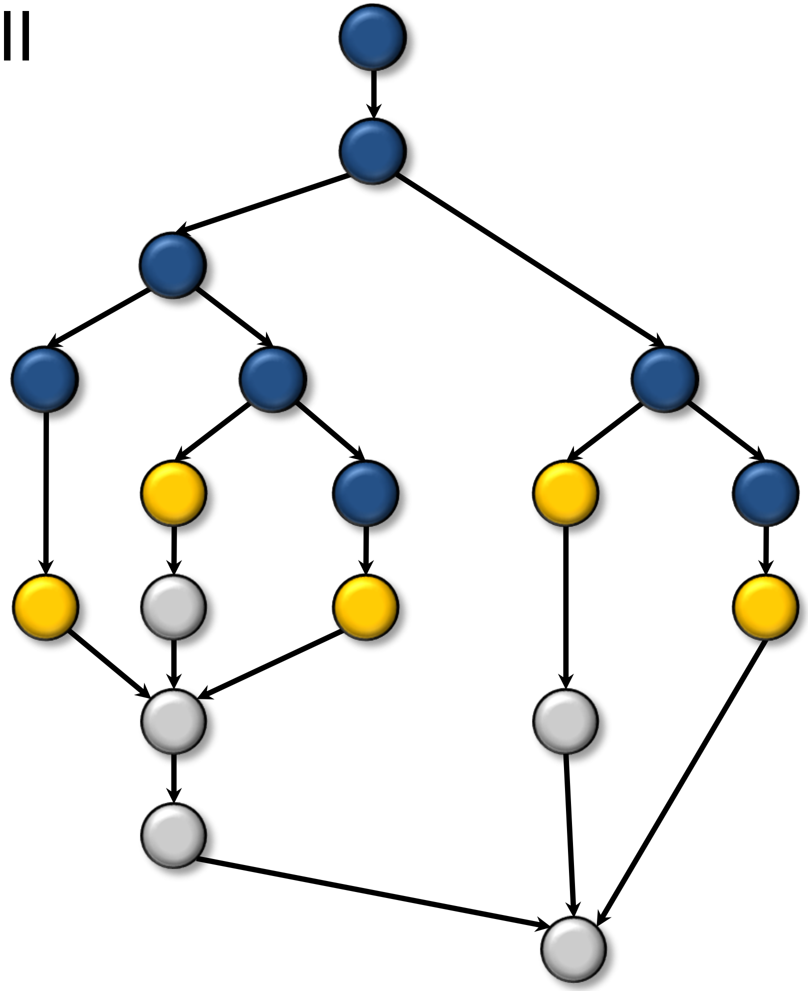




Greedy Scheduling

IDEA: Do as much as possible on every step.

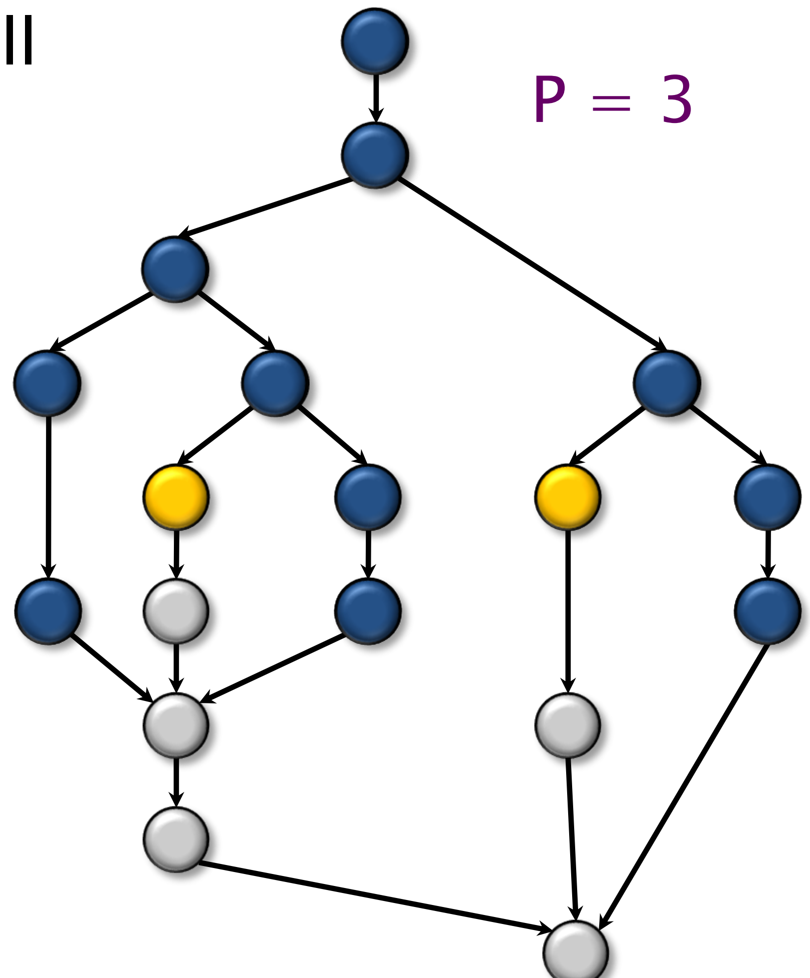
Definition. A strand is ready if all its predecessors have executed.



Greedy Scheduling

IDEA: Do as much as possible on every step.

Definition. A strand is ready if all its predecessors have executed.



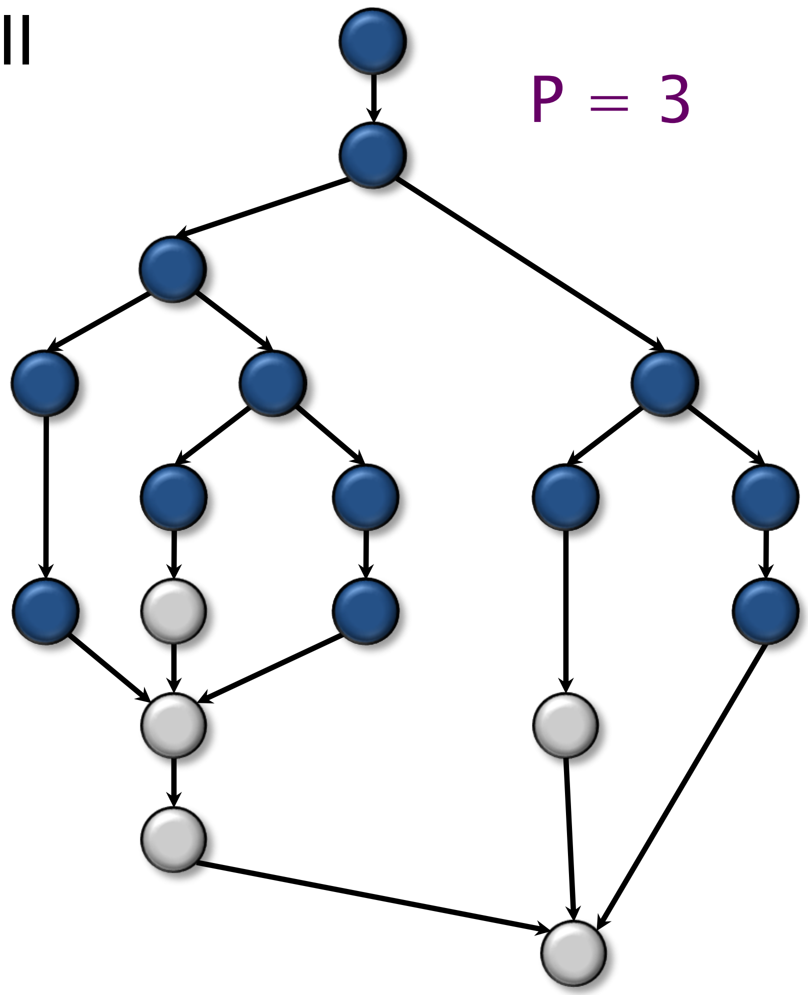
Complete step

- $\geq$ P strands ready.
- Run any P.

Greedy Scheduling

IDEA: Do as much as possible on every step.

Definition. A strand is *ready* if all its predecessors have executed.



Complete step

- $\geq P$ strands ready.
- Run any P.

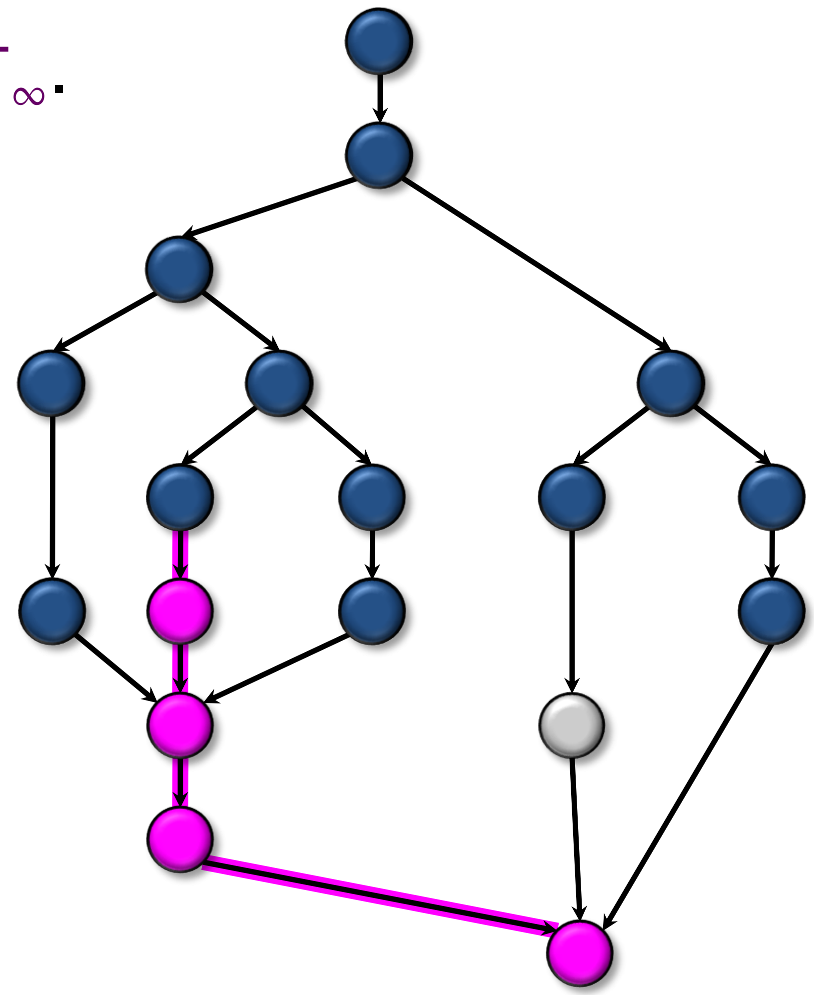
Incomplete step

- $< P$ strands ready.
- Run all of them.

Analysis of Greedy

Theorem [G68, B75, EZL89]. Any greedy scheduler achieves

$$T_P \le T_1/P + T_\infty.$$



Proof.

- # complete steps $\le T_1/P$, since each complete step performs P work.
- # incomplete steps $\le T_\infty$, since each incomplete step reduces the span of the unexecuted dag by 1. ■

Optimality of Greedy

Corollary. Any greedy scheduler achieves within a factor of 2 of optimal.

Proof. Let T_P^* be the execution time produced by the optimal scheduler. Since $T_P^* \geq \max\{T_1/P, T_\infty\}$ by the WORK and SPAN LAWS, we have

$$\begin{aligned} T_P &\leq T_1/P + T_\infty \\ &\leq 2\cdot\max\{T_1/P, T_\infty\} \\ &\leq 2T_P^* . \quad \blacksquare \end{aligned}$$

Linear Speedup

Corollary. Any greedy scheduler achieves near-perfect linear speedup whenever $T_1/T_\infty \gg P$.

Proof. Since $T_1/T_\infty \gg P$ is equivalent to $T_\infty \ll T_1/P$, the Greedy Scheduling Theorem gives us

$$\begin{aligned} T_P &\le T_1/P + T_\infty \\ &\approx T_1/P \, . \end{aligned}$$

Thus, the speedup is $T_1/T_P \approx P$. ■

Definition. The quantity T_1/PT_∞ is called the parallel slackness.

Cilk Performance

- Cilk's work-stealing scheduler achieves
  - $T_P = T_1/P + O(T_\infty)$ expected time (provably);
  - $T_P \approx T_1/P + T_\infty$ time (empirically).
- Near-perfect linear speedup as long as $P \ll T_1/T_\infty$.
- Instrumentation in Cilkscale allows you to measure T_1 and T_∞ .

Quiz on Scheduling Theory

Your team is performance-engineering a parallel program.* Although the program will be run on a 512-processor ideal machine, you are testing the program on a similar but smaller 32-processor machine. One of the engineers proposes a change that improves the 1- and 32-processor performance of the code as follows:

	T_1	T_{32}
Before	2048	65
After	1024	40

Assume that the running times obey the equation $T_P = T_1/P + T_\infty$. Is the optimization wise? Justify.

---

*This scenario really happened during the development of the ★Socrates chess-playing program, although the timing numbers have been simplified.

Quiz Solution

	T_1	T_{32}
Before	2048	65
After	1024	40

Before:

$$65 = 2048/32 + T_\infty$$

$$\Rightarrow T_\infty = 1$$

$$\Rightarrow T_{512} = 2048/512 + 1 = 5$$

After:

$$40 = 1024/32 + T_\infty$$

$$\Rightarrow T_\infty = 8$$

$$\Rightarrow T_{512} = 1024/512 + 8 = 10$$

Reject the optimization!

Quiz 2 on Scheduling Theory

Ben Bitdiddle measures the running time of his deterministic parallel program scheduled using a greedy scheduler on an ideal parallel computer with 4, 10, and 64 processors. Ben obtains the following running times:

Processors	Time
T_4	80
T_{10}	42
T_{64}	9

Argue that Ben messed up at least one of his measurements.

Quiz 2 Solution

Processors	Time
T_4	80
T_{10}	42
T_{64}	9

Work Law: $T_1 \le 4 * T_4$

$= 320$

Span Law: $T_\infty \le T_{64}$

$= 9$

Greedy scheduling on 10 processors: $T_{10} \le 320/10 + 9$

$= 41$

But Ben measured $T_{10} = 42$! ■