Concurrency
- access shared data
  - execute simultaneously
  - 2 or more threads
  - conflicts

Fall 2015
CSE 430: Operating Systems
\[ T_1 \]

CPU₁: \[ R₁ = 3 \]

STO \( R₁ \to i \)

CPU₂

\[ R₁ = 5 \]

STO \( R₁ \to i \)

\[ i = 0 \]

\[ \begin{cases} 3 \text{ or } 5 \end{cases} \]

CPU₂ \( \to \) Load \( R₁ \leftarrow i \)

\[ i \to \text{shrank} \]
The circled section problem
If $I_1$ wants to cause $E$ and

$$I_2$$ is not in $C_1$, then $I_2$ should not have been in $C_1$.

Progress:

1. Aligned with $C_1$ (notes)
2. Process
Cannot be defined indefinitely

Cannot depend on other things

Which things ensure that every decision can be seen or made to work
If I go to be green envy

[ Insert thing here ]

can become a measure

if I see CS, then T2

if I found a sure

Bounded nothing
\[ \text{graph} [i] = 0 \]

\[ \text{graph} [0] = 0 \]

CS

\[ \text{volt.} \left( \text{graph} [i] \right) \]

\[ \text{volt.} [0] = 1 \]

\[ \text{vote} [0] = 1 \]

WS

\[ \text{vote} [2] \rightarrow \text{vote} [0] \]
while (player_i < shifts)
    F = [07]
    for (i = 0; i < 5; i++)
        S
1 & 3 fails properly

but 1 has deadlock

2 has livelock (starvation)
1st known correct algo
1964 Dekker's Alg

1981 Peterson's Alg