Goals

- **Recap:** $s - t$-flows, $s - t$-cuts, weak duality
- Ford-Fulkerson algorithm
- Strong duality: Max-Flow-Min-Cut-Theorem
- Edmonds-Karp algorithm
- Application: Scheduling on Uniform Parallel Machines
Recap: $s - t$-flows

- **Network:** Digraph $D = (V, A)$ with capacity function $u : A \rightarrow \mathbb{R}_{\geq 0}$.
- For $s, t \in V$, an $s - t$-flow in $D$ is a function $f : A \rightarrow \mathbb{R}_{\geq 0}$ such that
  \[
  \sum_{a \in \delta^{\text{out}}(v)} f(a) = \sum_{a \in \delta^{\text{in}}(v)} f(a), \quad \text{for all } v \in V - \{s, t\}.
  \]
- $f$ is feasible, if $f(a) \leq u(e)$ for all $a \in A$.
- The value of $f$ is
  \[
  \text{value}(f) := \sum_{a \in \delta^{\text{out}}(s)} f(a) - \sum_{a \in \delta^{\text{in}}(s)} f(a).
  \]
Recap: $s - t$-flows

Example:

Maximum $s - t$-flow problem
Find a feasible $s - t$-flow of maximum value.
Recap: \( s - t \)-cuts

\[ D = (V, A), \quad u : A \rightarrow \mathbb{R}_{\geq 0} \]

- \( U \subseteq V \): \( \delta^\text{out}(U) := \{(u, v) \in A : u \in U, \ v \notin U\} \) is a cut.
- If \( s \in U, \ t \notin U \): \( \delta^\text{out}(U) \) is an \( s - t \)-cut.
- Capacity: \( u(\delta^\text{out}(U)) := \sum_{a \in \delta^\text{out}(U)} u(a) \).

Minimum \( s - t \)-cut problem

Find an \( s - t \) cut \( \delta^\text{out}(U) \) such that \( u(\delta^\text{out}(U)) \) is minimal.

Theorem (Weak duality)

Let \( f \) be a feasible \( s - t \)-flow and let \( \delta^\text{out}(U) \) be an \( s - t \)-cut, then

\[ \text{value}(f) \leq u(\delta^\text{out}(U)) \].
Informal idea: While there is an $s - t$-path with “positive capacity“ in the graph, send as much flow as possible along the path.
Computing max flows: Ford-Fulkerson algorithm

Informal idea: While there is an $s - t$-path with "positive capacity" in the graph, send as much flow as possible along the path.
Informal idea: While there is an $s - t$-path with “positive capacity“ in the graph, send as much flow as possible along the path.
Computing max flows: Ford-Fulkerson algorithm

**Informal idea:** While there is an $s-t$-path with “positive capacity“ in the graph, send as much flow as possible along the path.
Residual graphs

- For an arc \( a = (u, v) \in A \), let \( a^{-1} \) denote the arc \((v, u)\).
- **Wlog.** \( a^{-1} \not\in A \).
- Let \( f : A \to \mathbb{R} \) and \( u : A \to \mathbb{R}_{\geq 0} \) with \( 0 \leq f \leq u \).
- Define
  \[
  A_f := \{ a : a \in A, f(a) < u(a) \} \cup \{ a^{-1} : a \in A, f(a) > 0 \}.
  \]
- \( D_f = (V, A_f) \) is the **residual graph** of \( f \).
- Define **residual capacities**
  \[
  u_f : A_f \to \mathbb{R}_{\geq 0}, \quad u_f(a) = \begin{cases} u(a) - f(a), & \text{if } a \in A \\ f(a), & \text{if } a^{-1} \in A. \end{cases}
  \]
Ford-Fulkerson algorithm

- An **undirected path** is a sequence $P = (v_0, a_1, v_1, \ldots, v_{m-1}, a_m, v_m)$ such that $a_i \in A$ and $a_i = (v_{i-1}, v_i)$ or $a_i = (v_i, v_{i-1})$ for each $i = 1, \ldots, m$.
- Every directed path $P$ in $D_f$ yields an undirected path in $D$.
- Define $\chi^P \in \{0, \pm 1\}^A$ as
  
  $$
  \chi^P(a) = \begin{cases} 
  1, & \text{if } P \text{ traverses } a, \\
  -1, & \text{if } P \text{ traverses } a^{-1}, \\
  0, & \text{if } P \text{ traverses neither } a \text{ nor } a^{-1}.
  \end{cases}
  $$

1: **function** $\text{FORD-FULKERSON}(D = (V, A), u)$
2: $f \leftarrow 0.$
3: while $\exists s-t$-path $P$ in $D_f$ do
4: $\varepsilon \leftarrow \min\{u_f(a) : a \in P\}$.
5: $f \leftarrow f + \varepsilon \cdot \chi^P$.
6: end while
7: return $f$.
8: end function
Ford-Fulkerson: Correctness

1: function **Ford-Fulkerson**(*D = (V, A), u*)
2: \( f \leftarrow 0. \)
3: while \( \exists s - t\)-path \( P \) in \( D_f \) do
4: \( \varepsilon \leftarrow \min \{ u_f(a) : a \in P \} \).
5: \( f \leftarrow f + \varepsilon \cdot \chi^P. \)
6: end while
7: return \( f. \)
8: end function

**Theorem**

The output \( f \) of FF is a flow.

**Theorem**

If \( u \) is integer, then the output \( f \) of FF is integer.
Strong duality

Recall: $D = (V, A)$, $u : A \rightarrow \mathbb{R}_{\geq 0}$, $f$ flow, $U \subseteq V$

- $\text{excess}_f(U) := \sum_{a \in \delta^{\text{in}}(U)} f(a) - \sum_{a \in \delta^{\text{out}}(U)} f(a)$.
- $\text{excess}_f(U) = \sum_{v \in U} \text{excess}_f(v)$.
- $A_f := \{a : a \in A, f(a) < u(a)\} \cup \{a^{-1} : a \in A, f(a) > 0\}$.

**Theorem (Max-Flow-Min-Cut-Theorem)**

The maximum value of a feasible $s - t$-flow is equal to the minimum capacity of an $s - t$-cut.

**Corollary**

$FF$ computes a maximum $s - t$-flow. If $u$ is integer, there is an integer max flow.
Running time

1: function **Ford-Fulkerson**\((D = (V, A), u)\)
2: \( f \leftarrow 0. \)
3: \( \text{while } \exists s-t\text{-path } P \text{ in } D_f \text{ do} \)
4: \( \varepsilon \leftarrow \min\{u_f(a) : a \in P\}. \)
5: \( f \leftarrow f + \varepsilon \cdot \chi^P. \)
6: \( \text{end while} \)
7: \( \text{return } f. \)
8: \( \text{end function} \)

- Each iteration takes time \( O(|A|) \) (using e.g. Breadth-First search).
- How many iterations?
Running time

1: function **Ford-Fulkerson**\((D = (V, A), u)\)
2: \(f \leftarrow 0.\)
3: \(\textbf{while } \exists s - t\)-path \(P\) in \(D_f\) \(\textbf{do}\)
4: \(\varepsilon \leftarrow \min\{\mu_f(a) : a \in P\}\).
5: \(f \leftarrow f + \varepsilon \cdot \chi_P\).
6: \(\textbf{end while}\)
7: \(\textbf{return } f\).
8: \(\textbf{end function}\)

- Each iteration takes time \(O(|A|)\) (using e.g. Breadth-First search).
- How many iterations?
Running time

1: function **FORD-FULKERSON**\((D = (V, A), u)\)
2: \( f \leftarrow 0. \)
3: \( \textbf{while} \ \exists \ s - t\text{-path } P \text{ in } D_f \ \textbf{do} \)
4: \( \varepsilon \leftarrow \min\{u_f(a) : a \in P\}. \)
5: \( f \leftarrow f + \varepsilon \cdot \chi^P. \)
6: \( \textbf{end while} \)
7: return \( f. \)
8: end function

- Each iteration takes time \( O(|A|) \) (using e.g. Breadth-First search).
- How many iterations?
Running time

1: function \textsc{Ford-Fulkerson}(D = (V, A), u)
2: \hspace{1em} f \leftarrow 0.
3: \hspace{1em} while \ \exists s - t\text{-path } P \text{ in } D_f \ \text{do}
4: \hspace{2em} \varepsilon \leftarrow \min\{u_f(a) : a \in P\}.
5: \hspace{2em} f \leftarrow f + \varepsilon \cdot \chi^P.
6: \hspace{1em} \text{end while}
7: \hspace{1em} \text{return } f.
8: \text{end function}

- Each iteration takes time \( O(|A|) \) (using e.g. Breadth-First search).
- How many iterations?
Running time

1: function \textsc{Ford-Fulkerson}(D = (V, A), u)
2: \hspace{1em} f \leftarrow 0.
3: \hspace{1em} \textbf{while} \ \exists \ s - t\text{-path} \ P \ \text{in} \ D_f \ \textbf{do}
4: \hspace{2em} \varepsilon \leftarrow \min\{u_f(a) : a \in P\}.
5: \hspace{2em} f \leftarrow f + \varepsilon \cdot \chi^P.
6: \hspace{2em} \textbf{end while}
7: \hspace{1em} \textbf{return} \ f.
8: \textbf{end function}

- Each iteration takes time $O(|A|)$ (using e.g. Breadth-First search).
- How many iterations?
Theorem

If we choose in each iteration a shortest $s - t$-path in $D_f$ as a flow augmenting path, the number of iterations is at most $|V| \cdot |A|$.

- Digraph $D = (V, A)$, $s, t \in V$
- $\mu(D)$ is length of a shortest path from $s$ to $t$.
- $\alpha(D)$ is the set of arcs contained in at least one shortest $s - t$-path.

Lemma

Let $D = (V, A)$ be a digraph and $s, t \in V$. Define $D' := (V, A \cup \alpha(D)^{-1})$. Then $\mu(D) = \mu(D')$ and $\alpha(D) = \alpha(D')$.

Corollary

A maximum $s - t$-flow can be found in time $O(|V| \cdot |A|^2)$.
Scheduling on Uniform Parallel Machines

The setting

- \( n \) jobs, job \( j \) characterized by
  - processing time \( p_j \),
  - release time \( r_j \),
  - deadline \( d_j \).

- \( M \) identical machines.

- Each machine can process only one job at a time.

- Each job requires a total processing time \( p_j \) between release time \( r_j \) and deadline \( d_j \).

- **Preemption** allowed, i.e. processing of a job can be interrupted and continued later.

Scheduling problem

Find a feasible schedule that meets all constraints or assert that no such schedule exists.
Scheduling on Uniform Parallel Machines

Example:

<table>
<thead>
<tr>
<th>Job ( j )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_j )</td>
<td>1.5</td>
<td>1.25</td>
<td>2.1</td>
<td>3.6</td>
</tr>
<tr>
<td>( r_j )</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>( d_j )</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

Feasible schedule on 2 processors:

What does this have to do with Max-Flows?
Scheduling on Uniform Parallel Machines

- Sort the release times and deadlines ascending. Let $t_1, \ldots, t_k$ be the sorted sequence. (In our example: 1 3 4 5 7 9)
- Split the time horizon into disjoint intervals, i.e. define intervals $T_i := [t_i, t_{i+1})$ for $i = 1, \ldots, k$.
  (In our example we have $T_1 = [1, 3), T_2 = [3, 4), T_3 = [4, 5), T_4 = [5, 7), T_6 = [7, 9]$).
- Construct a network as follows:
  - Insert a node $j$ for each job $j = 1, \ldots, n$.
  - Insert a node $T_i$ for each interval $T_i$, $i = 1, \ldots, k - 1$.
  - Add an arc $(j, T_i)$ for each job $j$ such that $T_i \subseteq [r_i, d_i]$ of capacity $|T_i|$.
  - Add a node $s$ and arcs $(s, j)$ for each job $j$, of capacity $p_j$.
  - Add a node $t$ and arcs $(T_i, t)$ for each interval $i$, of capacity $M \cdot |T_i|$.

**Theorem**

There is a feasible schedule for the scheduling problem if and only if there is a flow of value $\sum_{j=1}^{n} p_j$. 
Goals

- Recap: $s - t$-flows, $s - t$-cuts, weak duality ✓
- Ford-Fulkerson algorithm ✓
- Strong duality: Max-Flow-Min-Cut-Theorem ✓
- Edmonds-Karp algorithm ✓
- Application: Scheduling on Uniform Parallel Machines ✓