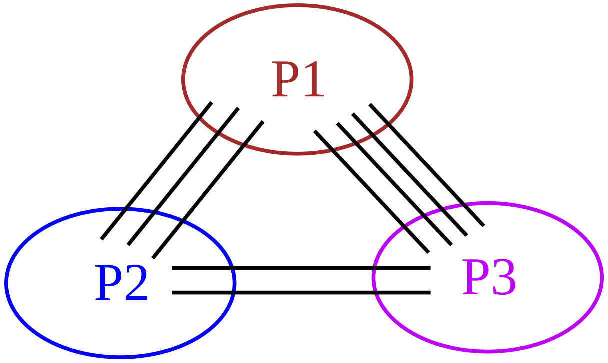
Network Flow Based Partitioning

- Min-cut balanced partitioning: Yang and Wong, ICCAD-94.
  - Based on max-flow min-cut theorem.

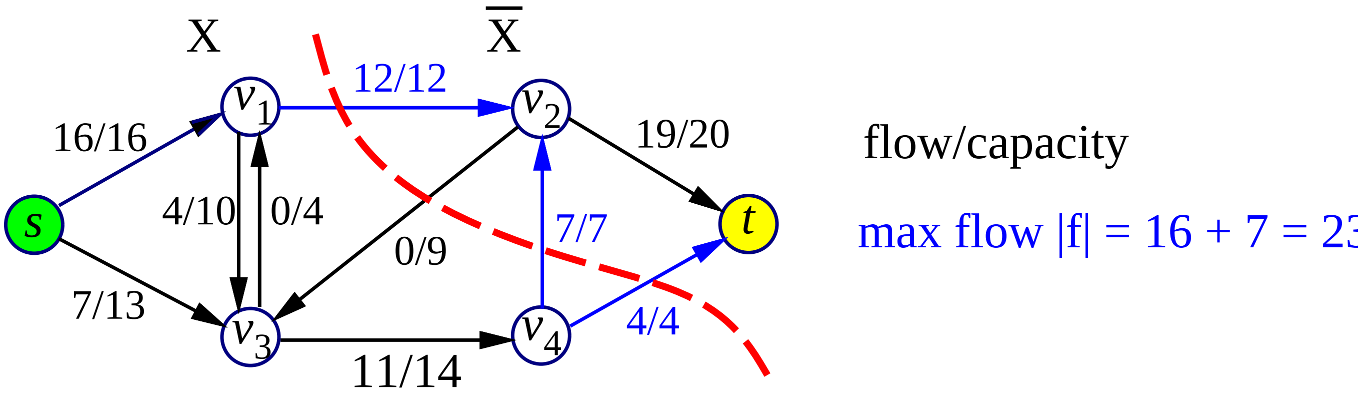


- Gate replication for partitioning: Yang and Wong, ICCAD-95.
- Performance-driven multiple-chip partitioning: Yang and Wong, FPGA'94, ED&TC-95.
- Multi-way partitioning with area and pin constraints: Liu and Wong, ISPD-97.
- Multi-resource partitioning: Liu, Zhu, and Wong, FPGA-98.
- Partitioning for time-multiplexed FPGAs: Liu and Wong, ICCAD-98.

Flow Networks

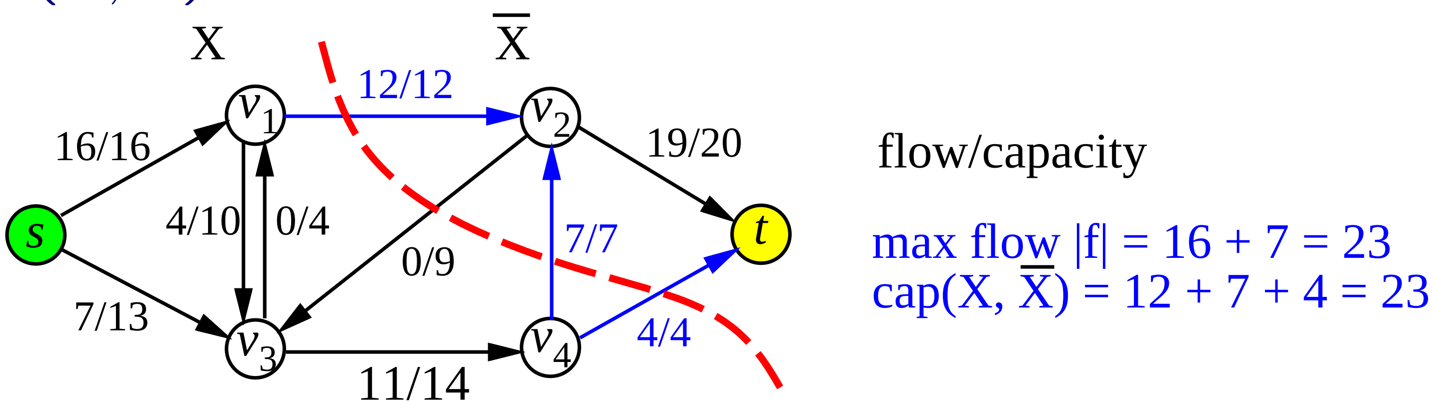
- A **flow network** $G = (V, E)$ is a **directed** graph in which each edge $(u, v) \in E$ has a **capacity** $c(u, v) > 0$.
- There is exactly one node with no incoming (outgoing) edges, called the **source** s (**sink** t).
- A **flow** $f : V \times V \to R$ satisfies
  - **Capacity constraint:** $f(u, v) \leq c(u, v), \forall u, v \in V$.
  - **Skew symmetry:** $f(u, v) = -f(v, u), \forall u, v \in V$.
  - **Flow conservation:** $\sum_{v \in V} f(u, v) = 0, \forall u \in V - \{s, t\}$.
- The **value** of a flow f: $|f| = \sum_{v \in V} f(s, v) = \sum_{v \in V} f(v, t)$

- **Maximum-flow problem:** Given a flow network G with source s and sink t, find a flow of maximum value from s to t.



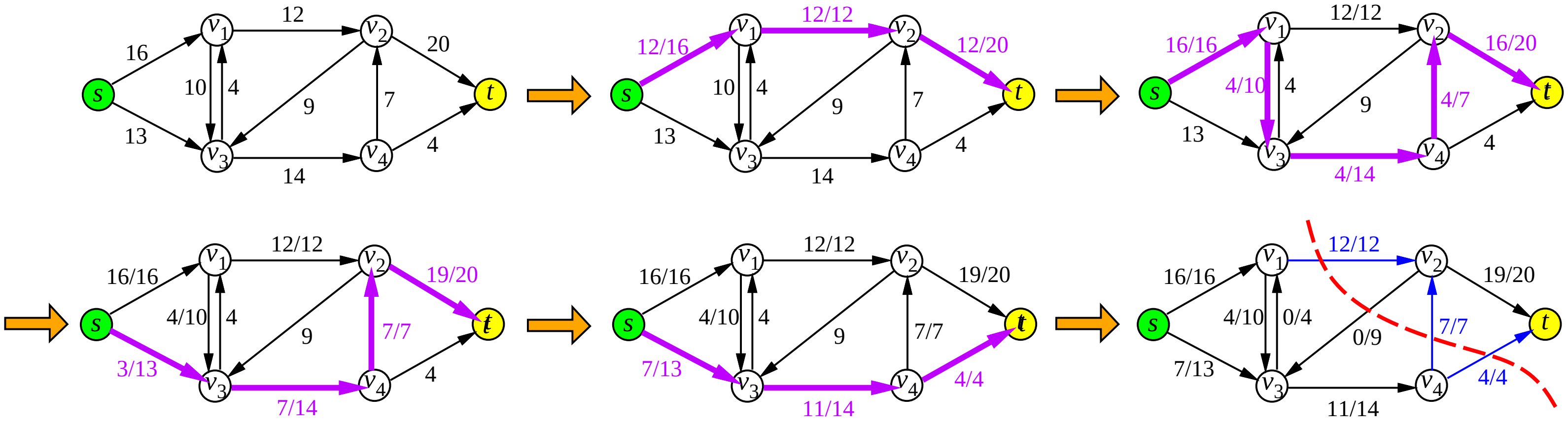
Max-Flow Min-Cut

- A **cut** $(X, \bar{X})$ of flow network $G = (V, E)$ is a partition of V into X and $\bar{X} = V - X$ such that $s \in X$ and $t \in \bar{X}$.
  - **Capacity of a cut:** $cap(X, \bar{X}) = \sum_{u \in X, v \in \bar{X}} c(u, v)$. (Count only **forward** edges!)
- **Max-flow min-cut theorem** Ford & Fulkerson, 1956.
  - f is a max-flow $\Longleftrightarrow$ $|f| = cap(X, \bar{X})$ for some min-cut $(X, \bar{X})$.



Network Flow Algorithms

- An **augmenting path** p is a simple path from s to t with the following properties:
  - For every edge $(u, v) \in E$ on p in the **forward** direction (a **forward edge**), we have $f(u, v) < c(u, v)$.
  - For every edge $(u, v) \in E$ on p in the **reverse** direction (a **backward edge**), we have $f(u, v) > 0$.
- f is a max-flow $\iff$ no more augmenting path.



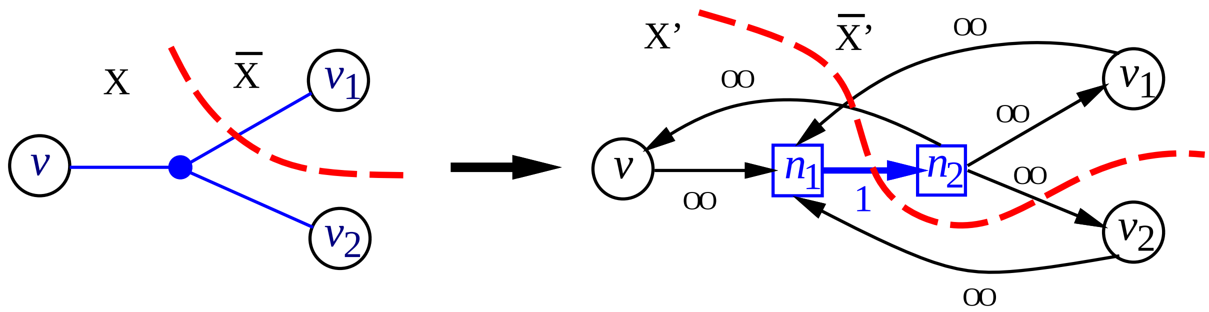
- First algorithm by Ford & Fulkerson in 1959: $O(|E||f|)$; First **polynomial-time** algorithm by Edmonds & Karp in 1969: $O(|E|^2|V|)$; Goldberg & Tarjan in 1985: $O(|E||V|\lg(|V|^2/|E|))$, etc.

Network Flow Based Partitioning

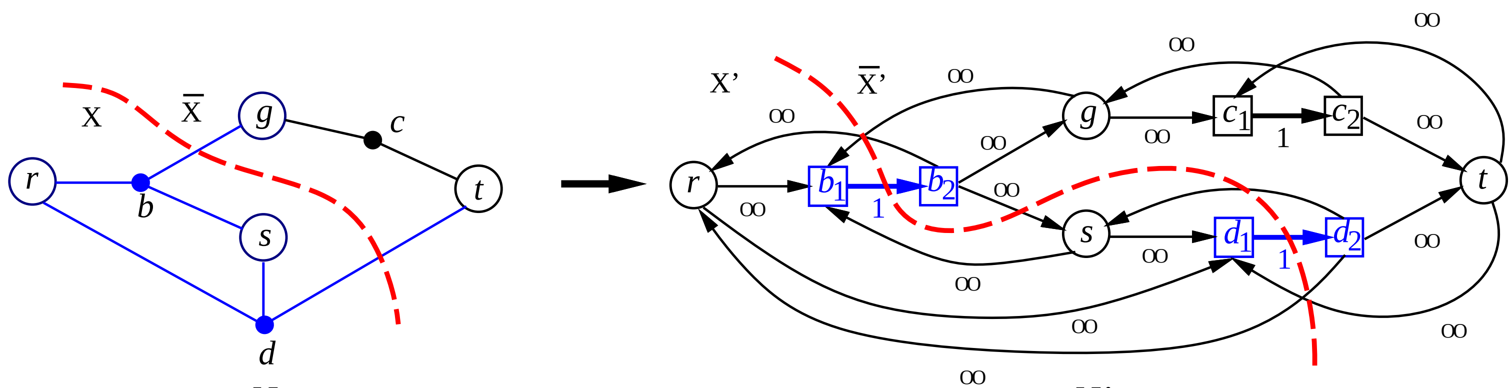
- Why was the technique not wisely used in partitioning?
  - Works on graphs, not hypergraphs.
  - Results in unbalanced partitions; repeated min-cut for balance: $|V|$ max-flows, time-consuming!
- Yang & Wong, ICCAD-94.
  - Exact **net** modeling by flow network.
  - Optimal algorithm for min-net-cut bipartition (unbalanced).
  - Efficient implementation for repeated min-net-cut: same asymptotic time as **one** max-flow computation.

Min-Net-Cut Bipartition

- Net modeling by flow network:



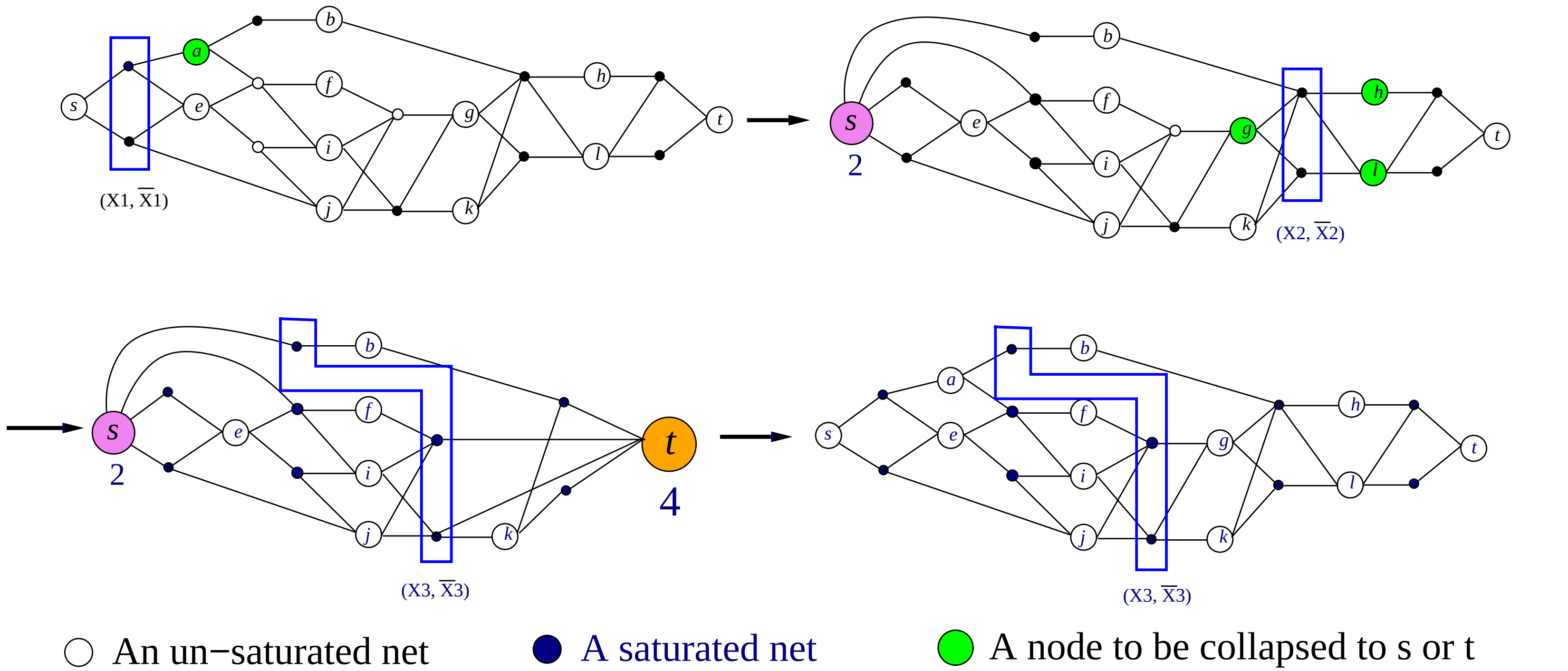
- A min-net-cut $(X, \bar{X})$ in $N \iff$ A min-capacity-cut $(X', \bar{X}')$ in N'.
- Size of flow network: $|V'| \leq 3|V|$, $|E'| \leq 2|E| + 3|V|$.
- Time complexity: O(min-net-cut-size) $\times |E| =$ O$(|V||E|)$.



N   *N'*

Repeated Min-Cut for Balanced Bipartition (FBB)

- Allow component weights to deviate from $(1-\epsilon)W/2$ to $(1+\epsilon)W/2$.



Incremental Flow

- Repeatedly compute max-flow: very time-consuming.
- No need to compute max-flow from scratch in each iteration.
- Retain the flow function computed in the previous iteration.
- Find additional flow in each iteration. Still correct.
- FBB time complexity: $O(|V||E|)$, same as **one** max-flow.
  - At most $2|V|$ augmenting path computations.
    - At each augmenting path computation, either an augmenting path is found, or a new cut is found, and at least 1 node is collapsed to s or t.
    - At most $|f| \leq |V|$ augmenting paths found, since bridging edges have unit capacity.

– An augmenting path computation: $O(|E|)$ time.