Multicommodity Flows Over Time

Martin Skutella

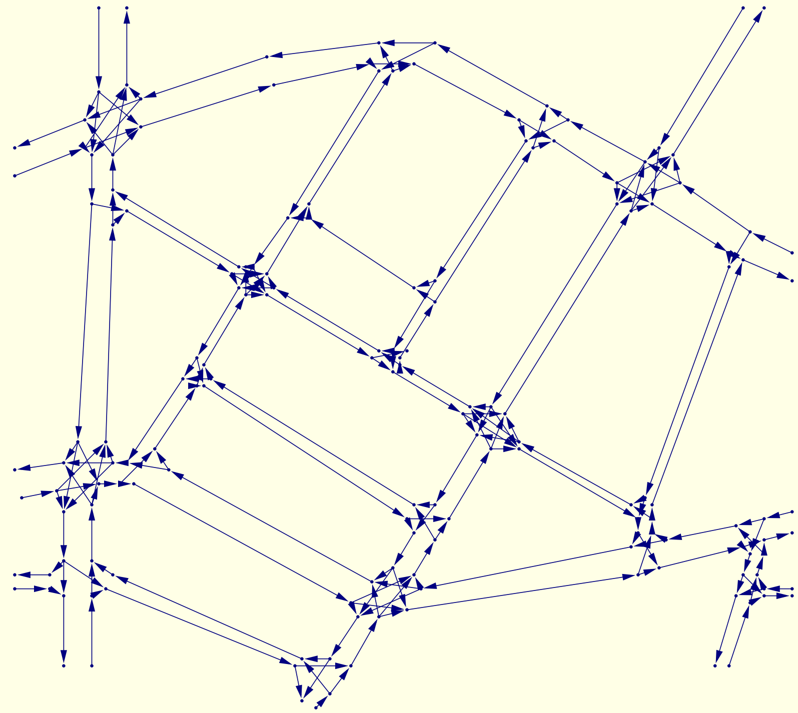
(TU Berlin $\longrightarrow$ MPI Saarbrücken)

joint work with:

- Lisa Fleischer
- Alexander Hall and Steffen Hippler

Traffic Management and Route Guidance

Network flow theory constitutes a promising approach to optimizing large real-life traffic systems.



Traffic can be modeled as flow in directed graph representing the road network.

Flows with Temporal Dimension

Classical network flow theory considers steady state flows. However, in many applications (e. g. road traffic), time plays a vital role!

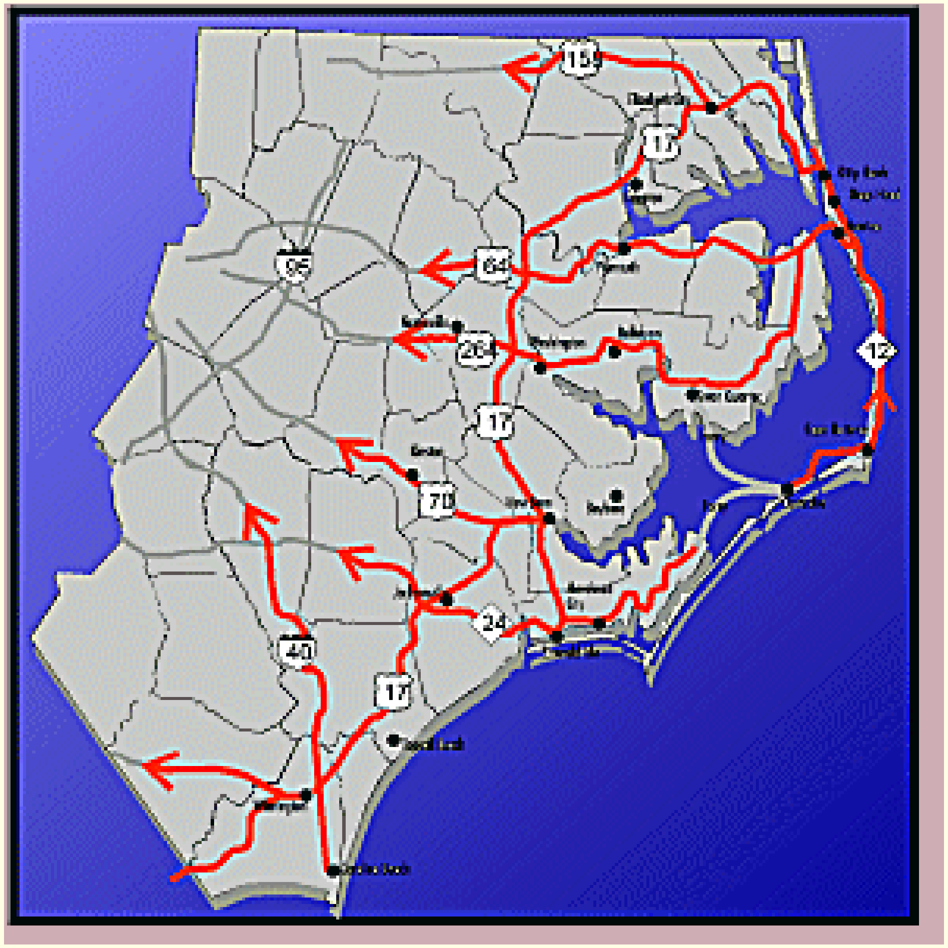
Flows with Temporal Dimension

Classical network flow theory considers steady state flows. However, in many applications (e. g. road traffic), time plays a vital role!

- Flow variation over time due to seasonal altering demands, supplies, and/or arc capacities.
- Flow travels only at a certain pace through the network, that is, there are transit times on the arcs.

Further Applications

- evacuation plans
- communication networks (e. g., Internet)
- production systems
- air traffic
- logistics
- financial flows



Literature: For surveys and more applications see, e.g.:

Aronson (1989); Powell, Jaillet & Odoni (1995).

Historical View

The notion of flows over time (or ‘dynamic flows’) was introduced by Ford & Fulkerson (1958):

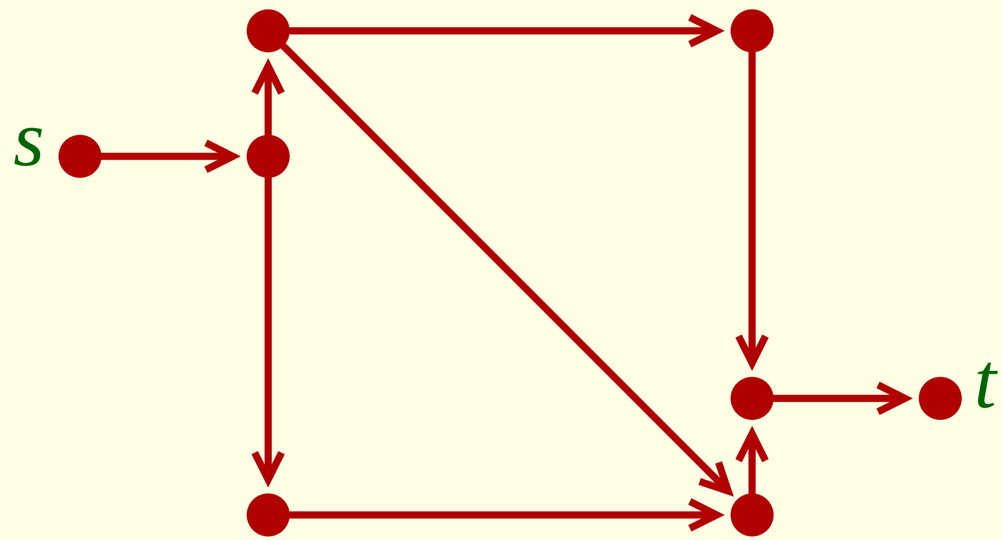
Given: Network $\mathcal{N} = (V, A)$ with capacities u_e and transit times τ_e on the arcs $e \in A$; time horizon $T \in \mathbb{Z}_{\geqslant 0}$.

Interpretation:

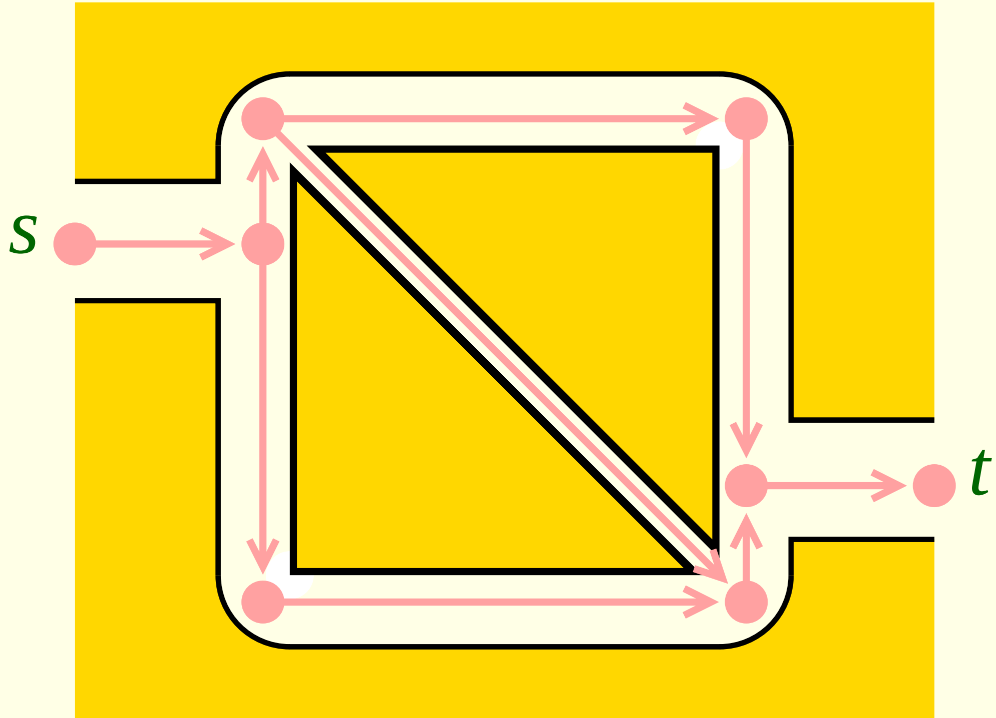
- The transit time τ_e of an arc $e = (v, w)$ specifies the time it takes for flow to travel from v to w on e.
- The capacity u_e bounds the rate of flow entering e.

Aim: Determine the maximal amount of flow that can be sent from source $s \in V$ to sink $t \in V$ within time T.

Intuition: Network of Pipelines



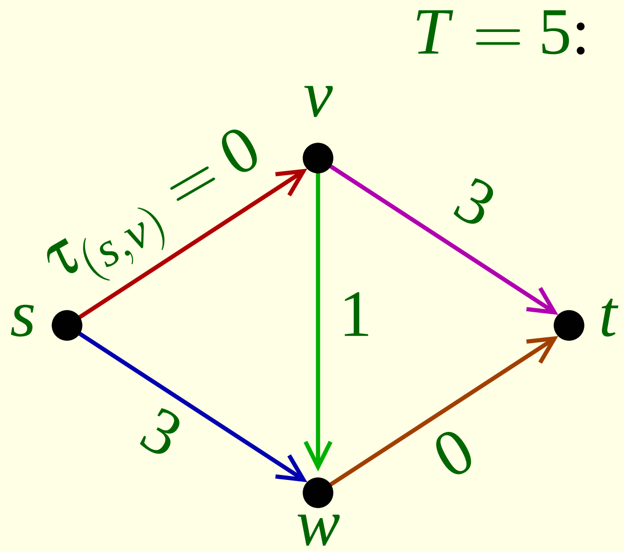
Intuition: Network of Pipelines

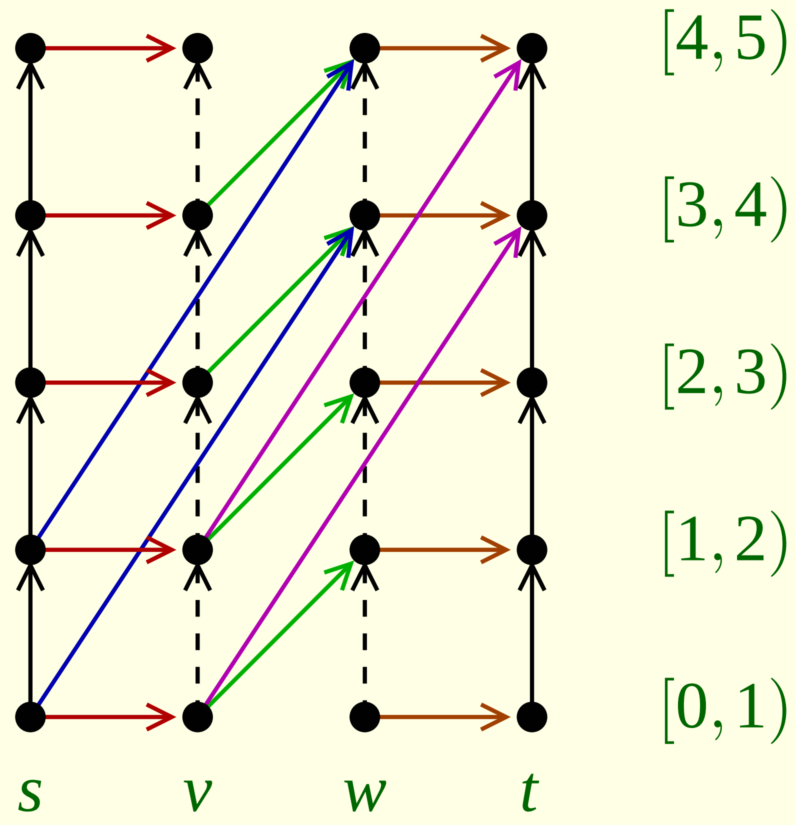


flow	$\longleftrightarrow$	fluid
arcs	$\longleftrightarrow$	pipes
transit time	$\longleftrightarrow$	length of pipe
capacity	$\longleftrightarrow$	width of pipe

Time-Expanded Networks

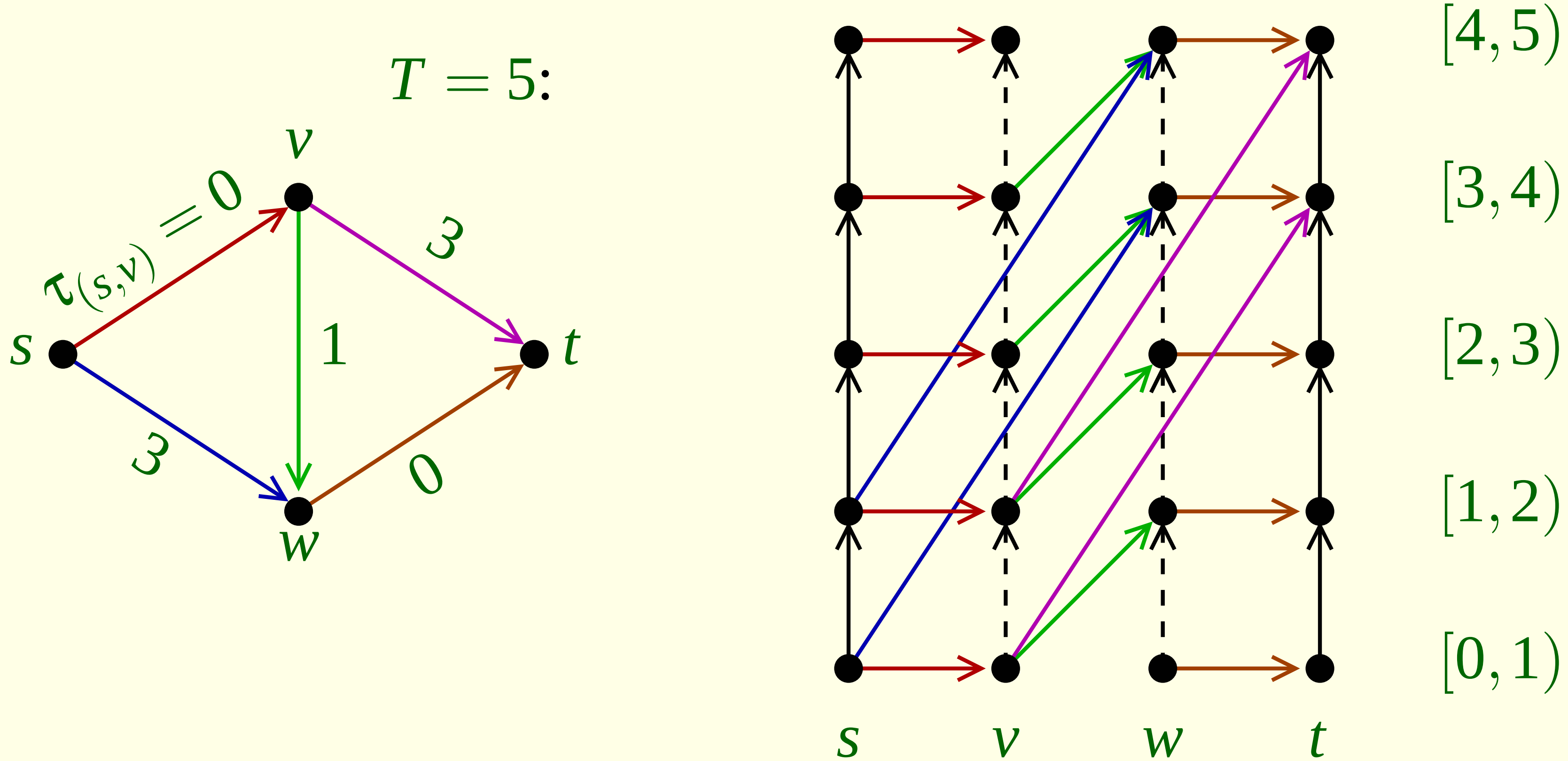
Observation. Flows over time can be solved as static flow problems in time-expanded networks:





Time-Expanded Networks

Observation. Flows over time can be solved as static flow problems in time-expanded networks:



Drawback: Time-expanded network consists of T time layers — only pseudo-polynomial in input size!

The Complexity Landscape of Flows Over Time

	s-t-flow	trans-shipment	min-cost	multi-commodity
static	poly		poly	poly (LP)

The Complexity Landscape of Flows Over Time

	s-t-flow	trans-shipment	min-cost	multi-commodity
static	poly		poly	poly (LP)
dyn.				

The Complexity Landscape of Flows Over Time

	s-t-flow	trans-shipment	min-cost	multi-commodity
static	poly		poly	poly (LP)
dyn.	poly (static min-cost flow)			

Ford & Fulkerson (1958):

Maximum s-t-flow over time can be solved by one static min-cost flow computation in the given network.

The Complexity Landscape of Flows Over Time

	s-t-flow	trans-shipment	min-cost	multi-commodity
static	poly		poly	poly (LP)
dyn.	poly (static min-cost flow)	poly (minimize submodular functions)		

Hoppe & Tardos (1994/95):

Transshipment over time can be solved in polynomial time (but relies on submodular function minimization).

The Complexity Landscape of Flows Over Time

	s-t-flow	trans-shipment	min-cost	multi-commodity
static	poly		poly	poly (LP)
dyn.	poly (static min-cost flow)	poly (minimize submodular functions)	pseudo-poly NP-hard	

Klinz & Woeginger (1995):

Minimum cost s-t-flow over time is NP-hard (already on series-parallel graphs).

The Complexity Landscape of Flows Over Time

	s-t-flow	trans-shipment	min-cost	multi-commodity
static	poly		poly	poly (LP)
dyn.	poly (static min-cost flow)	poly (minimize submodular functions)	pseudo-poly NP-hard	pseudo-poly (LP) NP-hard

Hall, Hippler & Sk. (2003):

Fractional multicommodity flow over time is NP-hard (already on series-parallel graphs). Without storage of flow at intermediate nodes, it is even strongly NP-hard.

Further Results and References

- Gale (1959) observes that earliest arrival flows exist.
- Wilkinson (1971) and Minieka (1973) give equivalent pseudo-polynomial time algorithms to find them.
- Hoppe & Tardos (1994) approximate them with a fully polynomial time approximation scheme (FPTAS).
- Orlin (1983, 1984) considers infinite horizon (minimum cost) flows over time that maximize throughput.
- Fleischer (2001a,2001b) and Fleischer & Orlin (2000) study flows over time with zero transit times.
- Fleischer & Tardos (1998) discuss continuous versus discrete time model.

Problem Definition

Multi-Commodity Flow Over Time.

Given: Network $\mathcal{N}$, time horizon T, set of commodities $i = 1, \ldots, k$ with sources s_i, sinks t_i, and demand values D_i.

Task: Satisfy all demands within time T.

Problem Definition

Multi-Commodity Flow Over Time.

Given: Network $\mathcal{N}$, time horizon T, set of commodities $i = 1, \ldots, k$ with sources s_i, sinks t_i, and demand values D_i.

Task: Satisfy all demands within time T.

Multi-Commodity Transshipment Over Time.

Every commodity can have several sources and sinks with given supplies and demands.

Problem Definition

Multi-Commodity Flow Over Time.

Given: Network $\mathcal{N}$, time horizon T, set of commodities $i = 1, \ldots, k$ with sources s_i, sinks t_i, and demand values D_i.

Task: Satisfy all demands within time T.

Multi-Commodity Transshipment Over Time.

Every commodity can have several sources and sinks with given supplies and demands.

Min-Cost Multi-Commodity Transshipment Over Time.

Minimize total flow cost in network with costs on arcs.

Polynomially Solvable Cases

Joint work with Alex Hall & Steffen Hippler:

- Multicommodity flow over time with intermediate storage is polynomially solvable if every node has at most one outgoing arc:

  (Route all flow greedily, i.e., as early as possible; whenever a conflict occurs on an arc, give priority to the commodity which is further away from its sink.)

Polynomially Solvable Cases

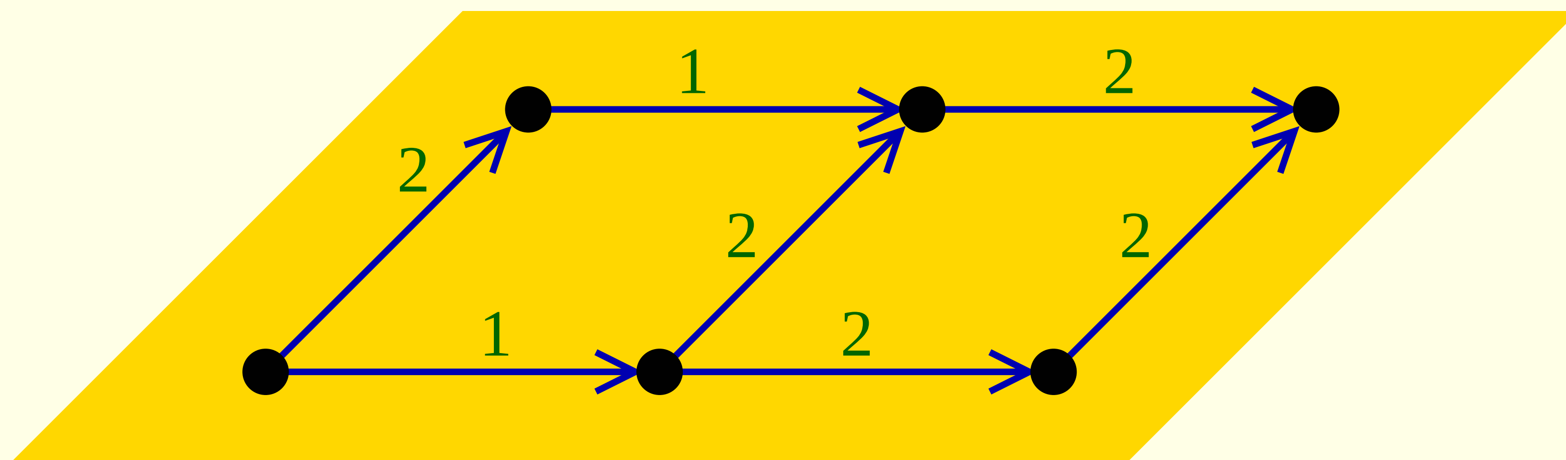
Joint work with Alex Hall & Steffen Hippler:

- Multicommodity flow over time with intermediate storage is polynomially solvable if every node has at most one outgoing arc:

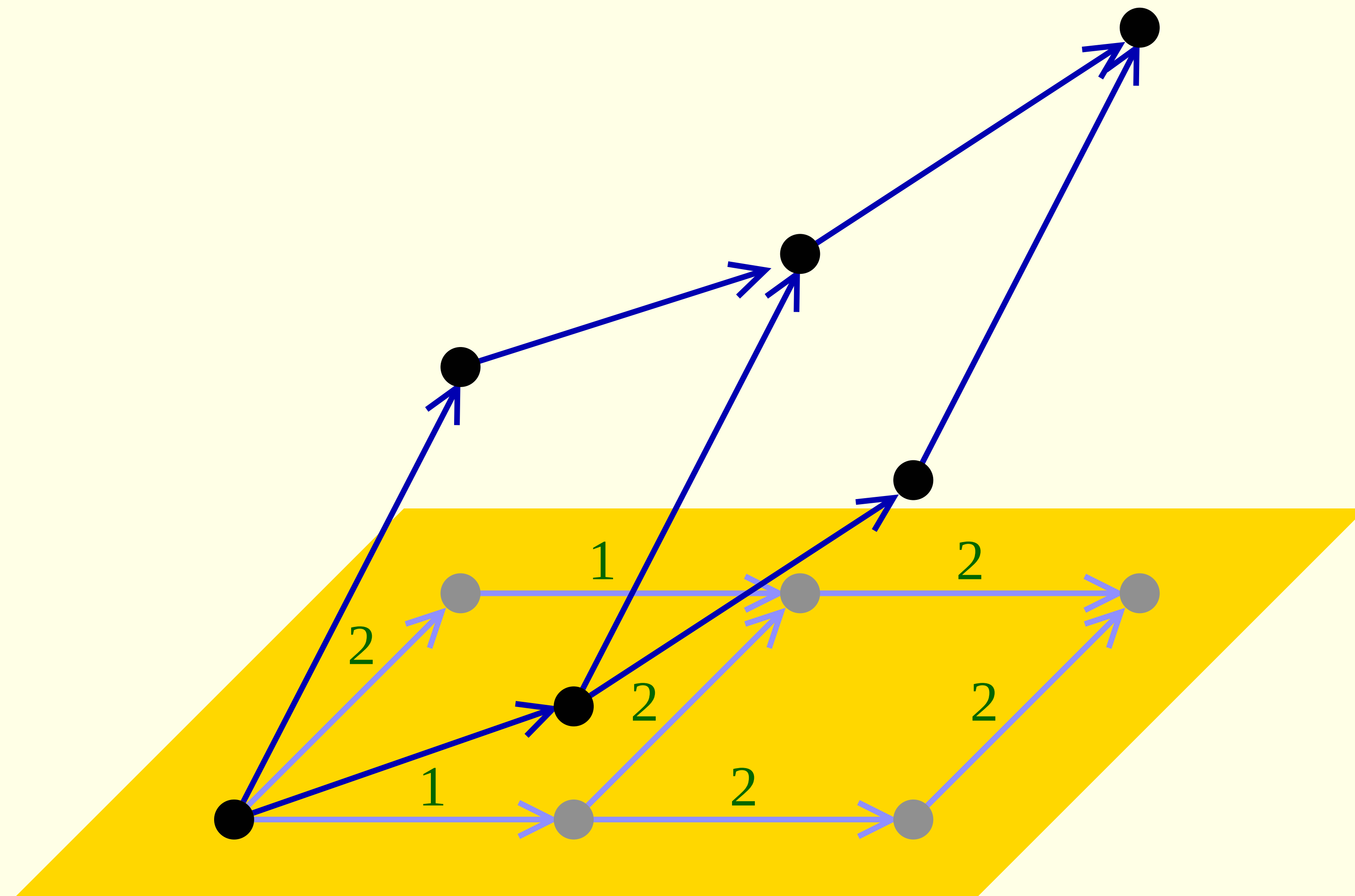
  (Route all flow greedily, i.e., as early as possible; whenever a conflict occurs on an arc, give priority to the commodity which is further away from its sink.)

- If between every fixed pair of nodes all paths have the same transit time, a min-cost multicommodity transshipment over time (with or without intermediate storage) can be computed in polynomial time.

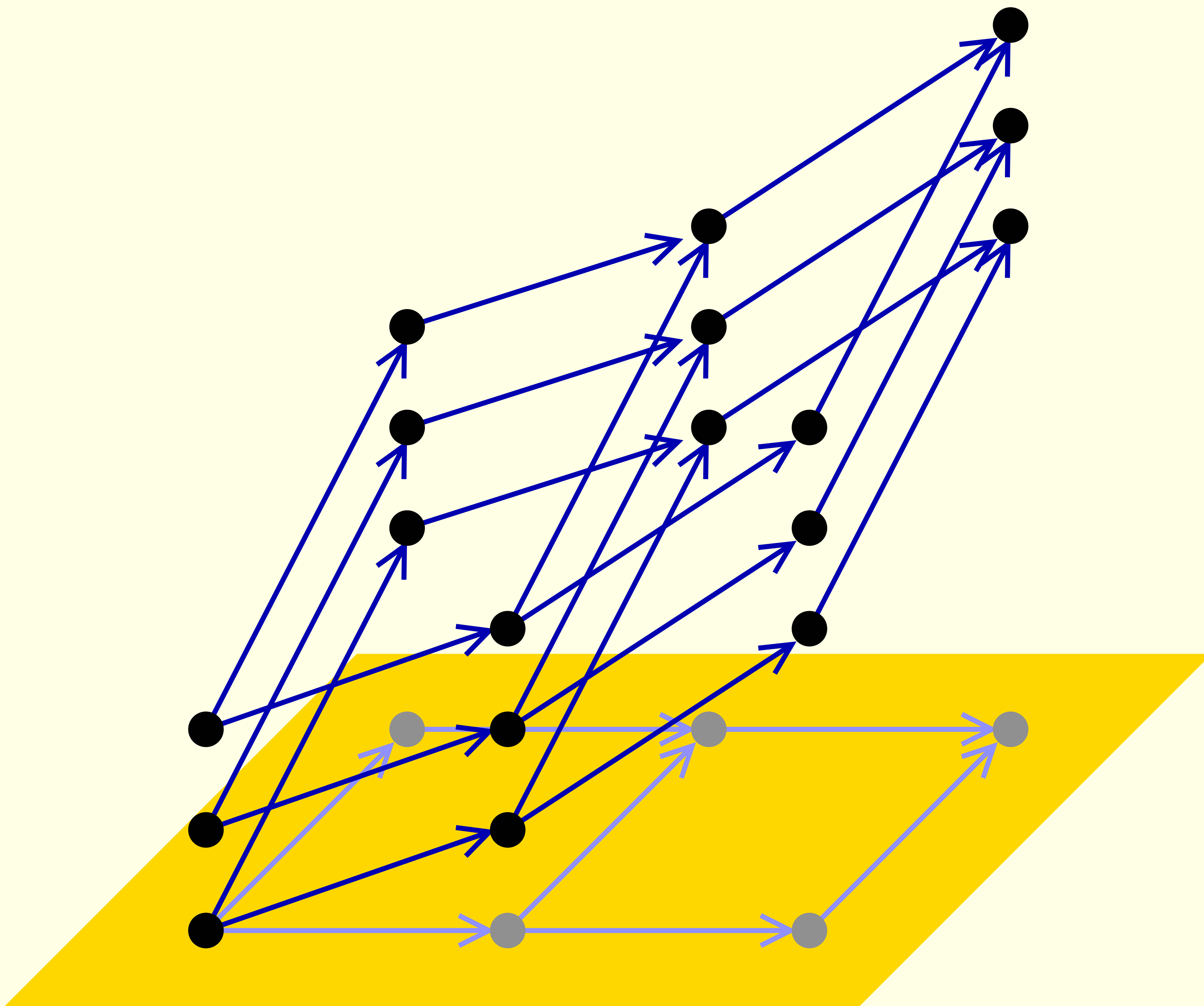
Example



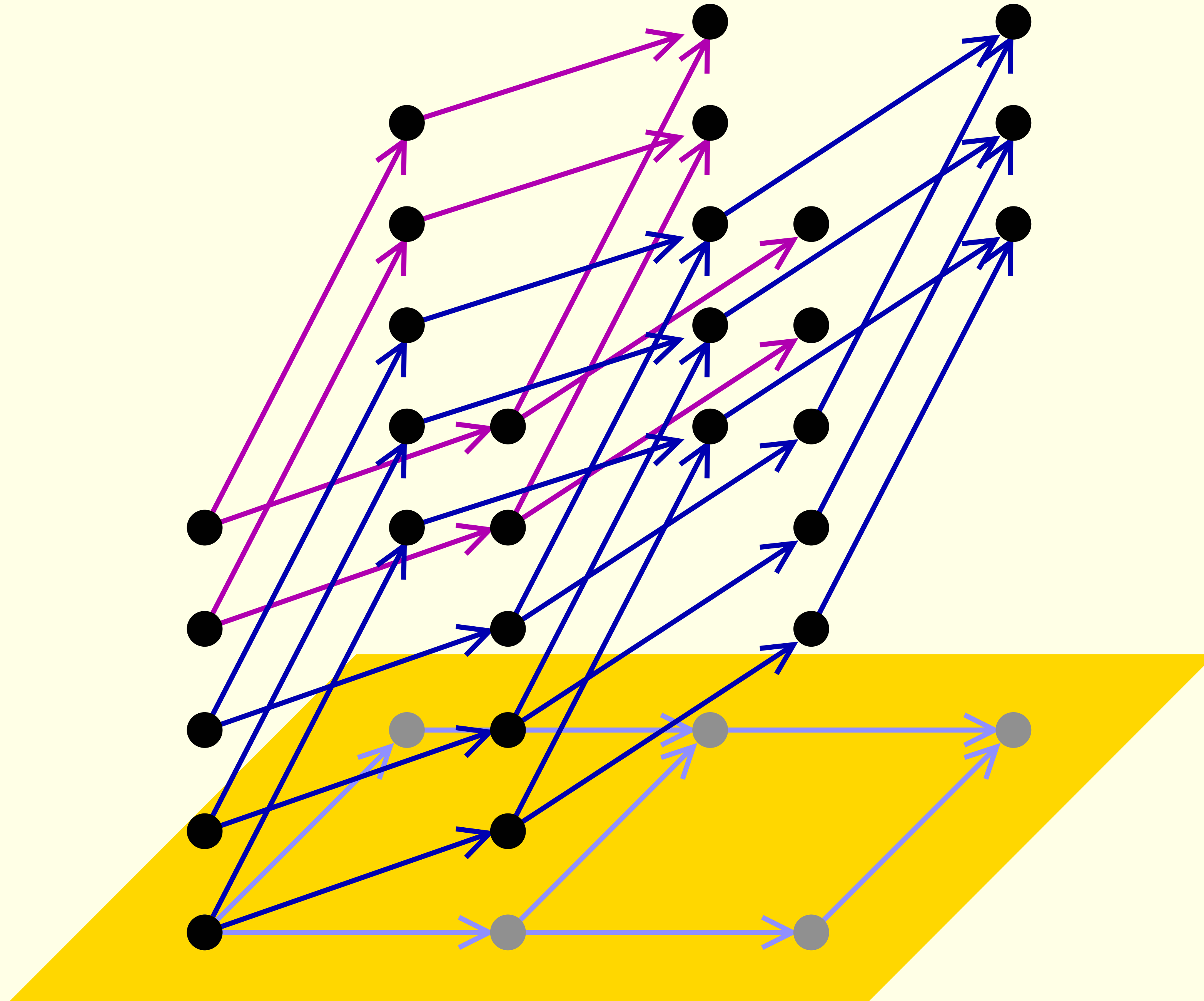
Proof Sketch



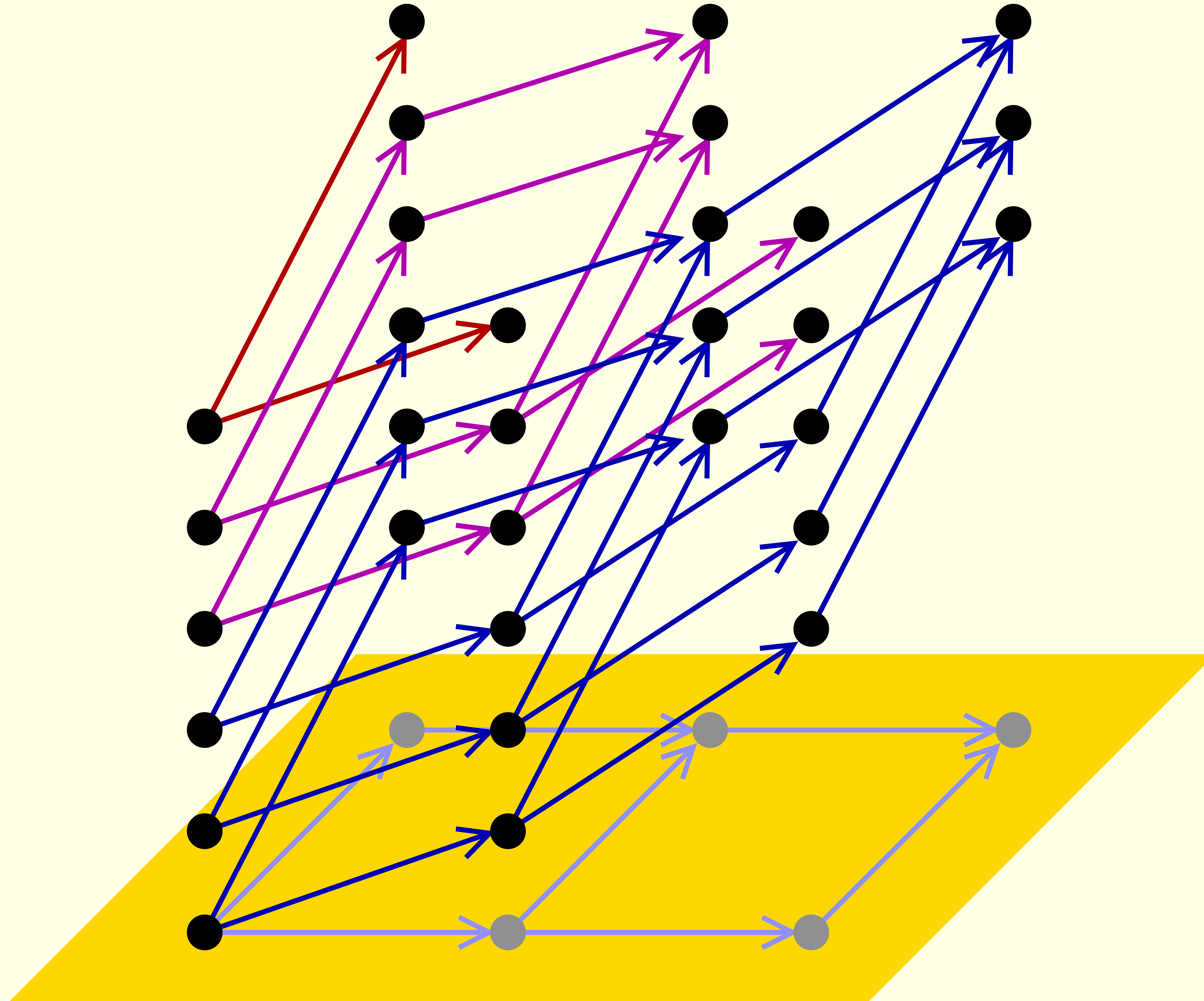
Proof Sketch



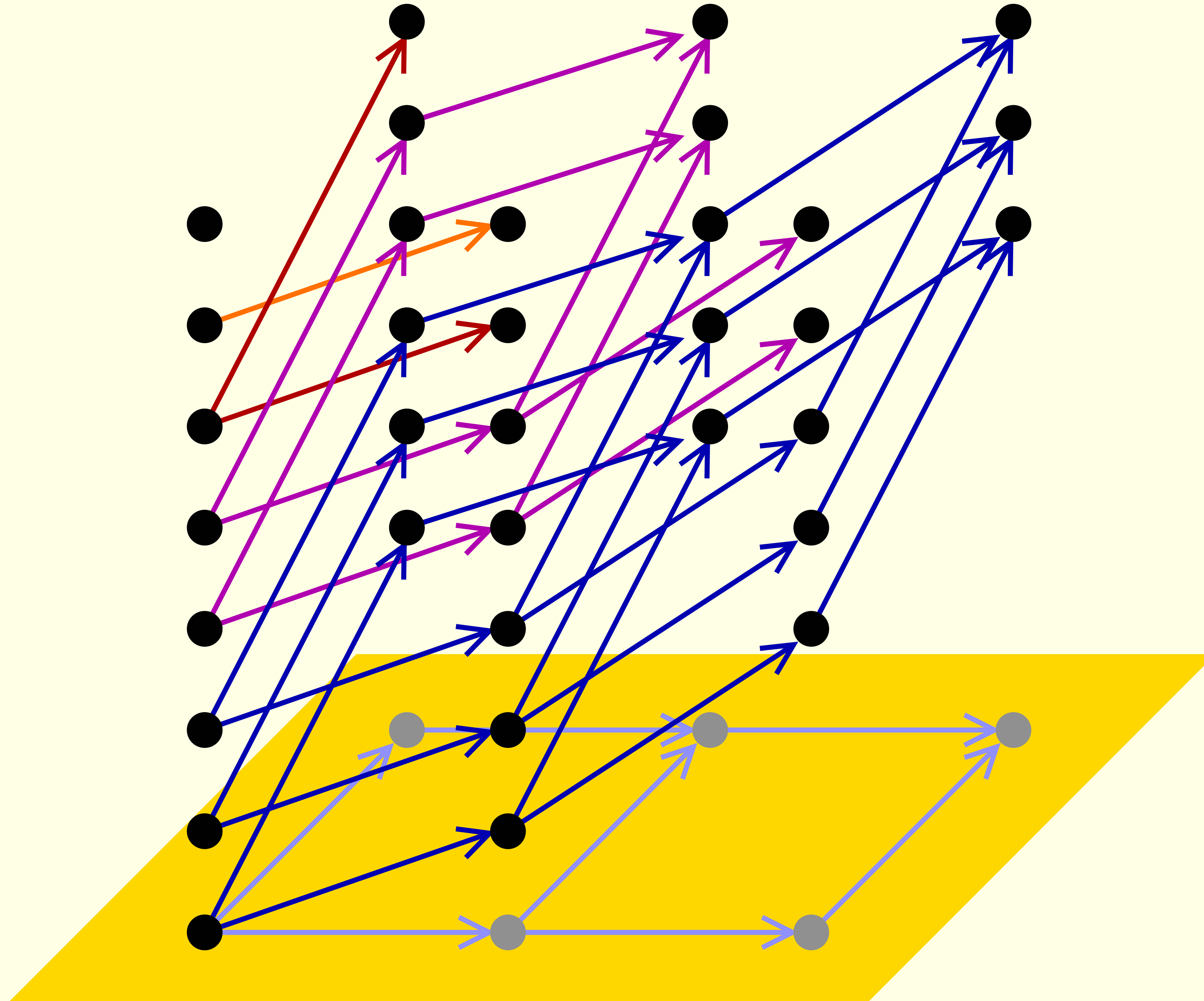
Proof Sketch



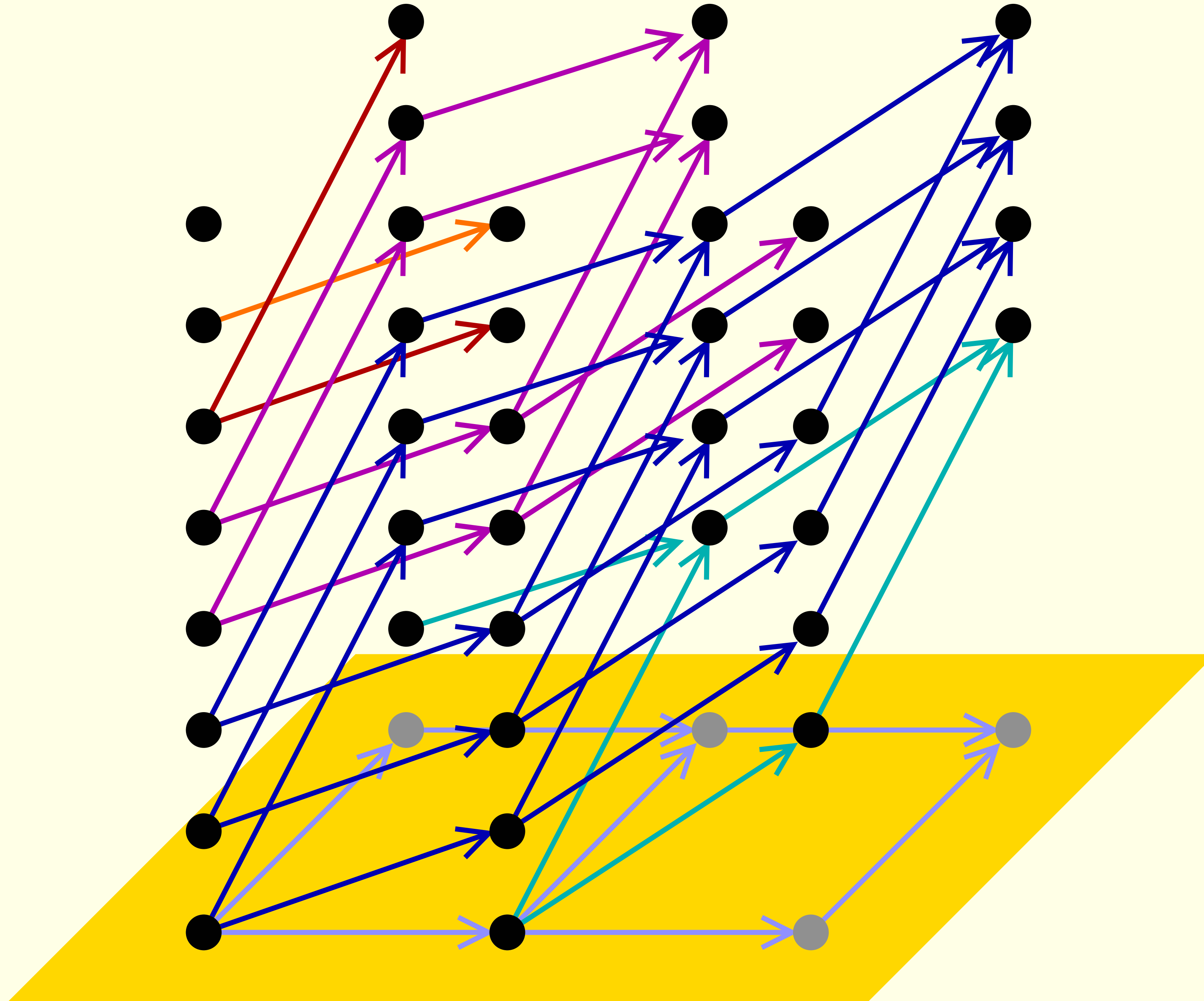
Proof Sketch



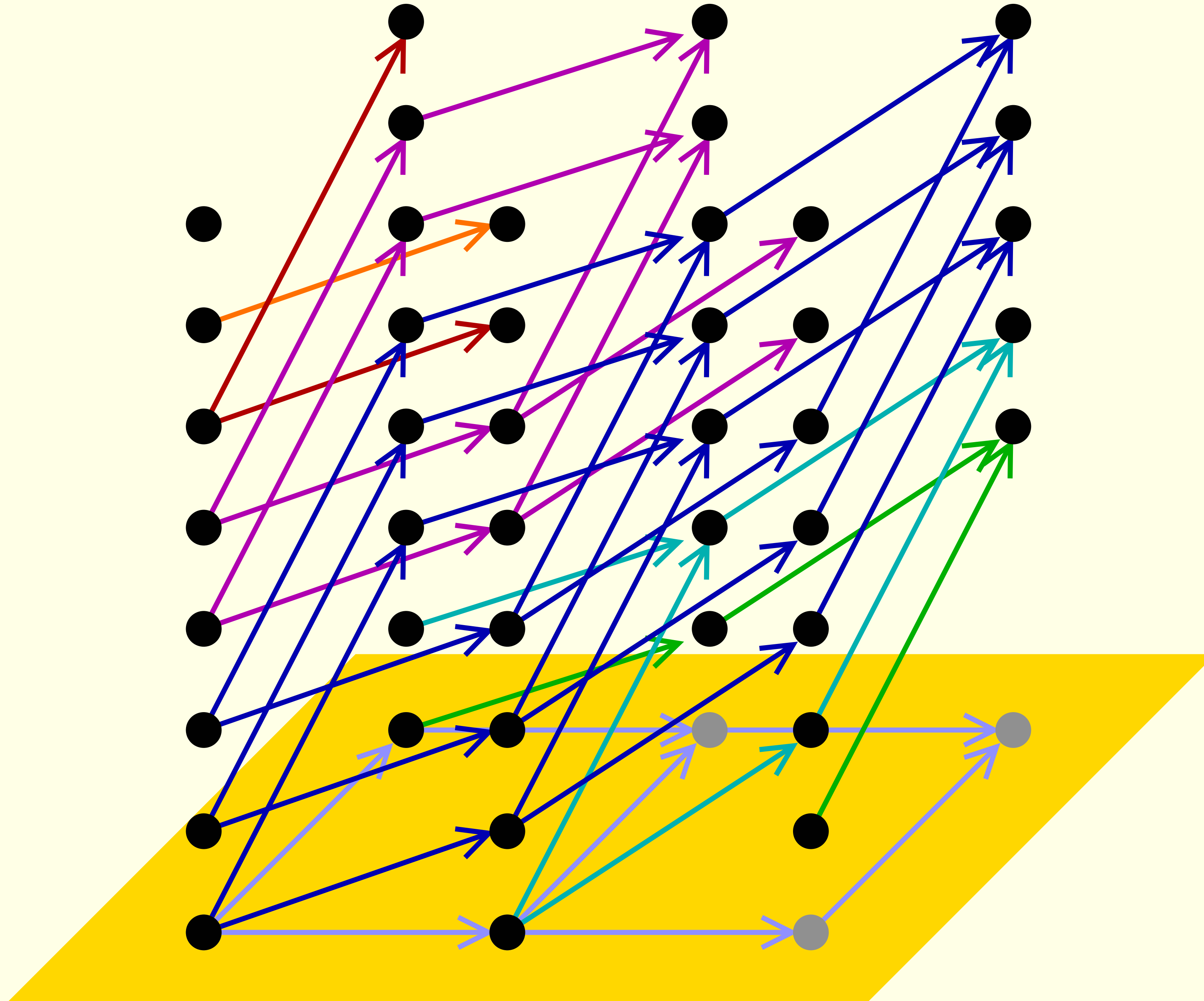
Proof Sketch



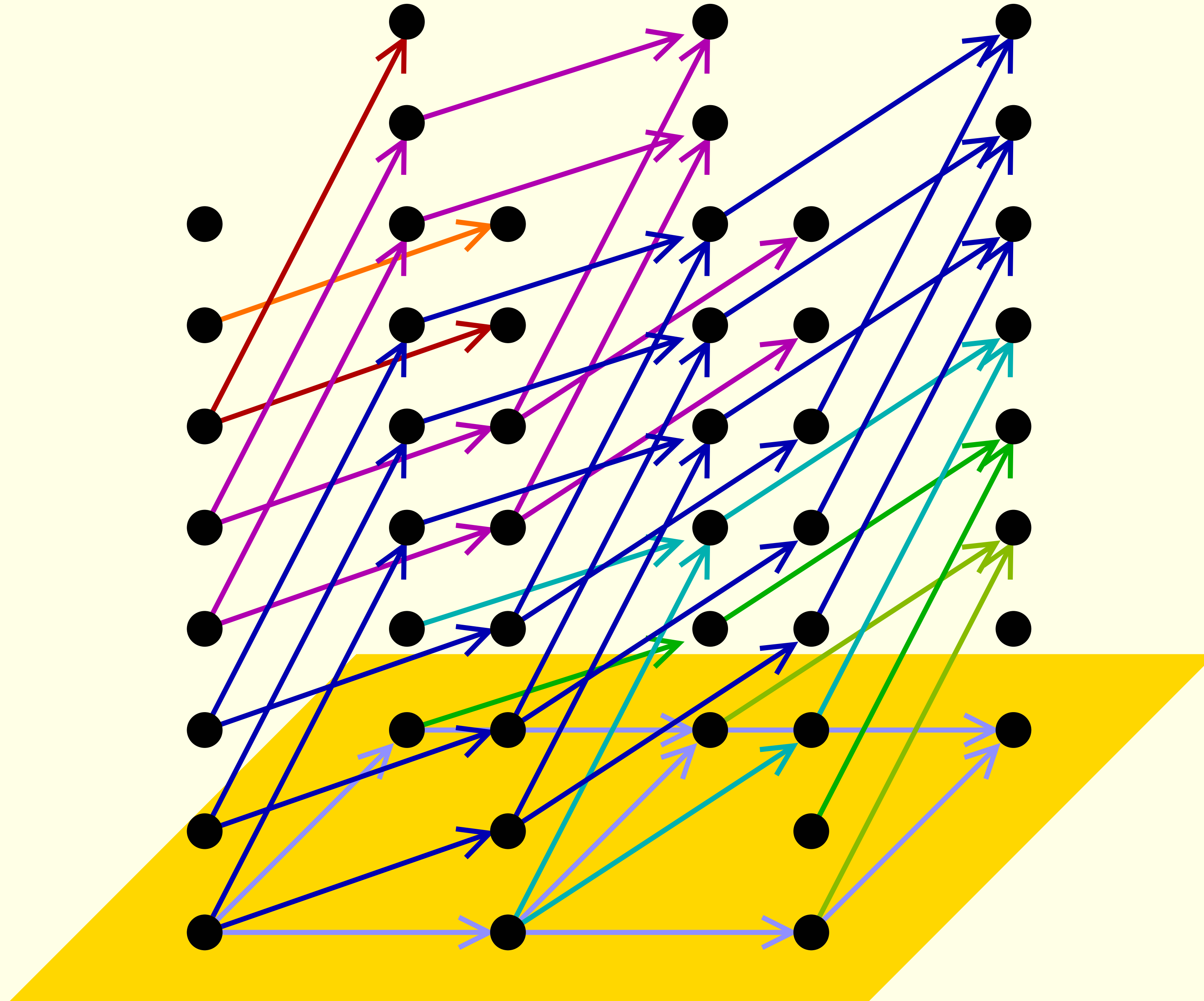
Proof Sketch



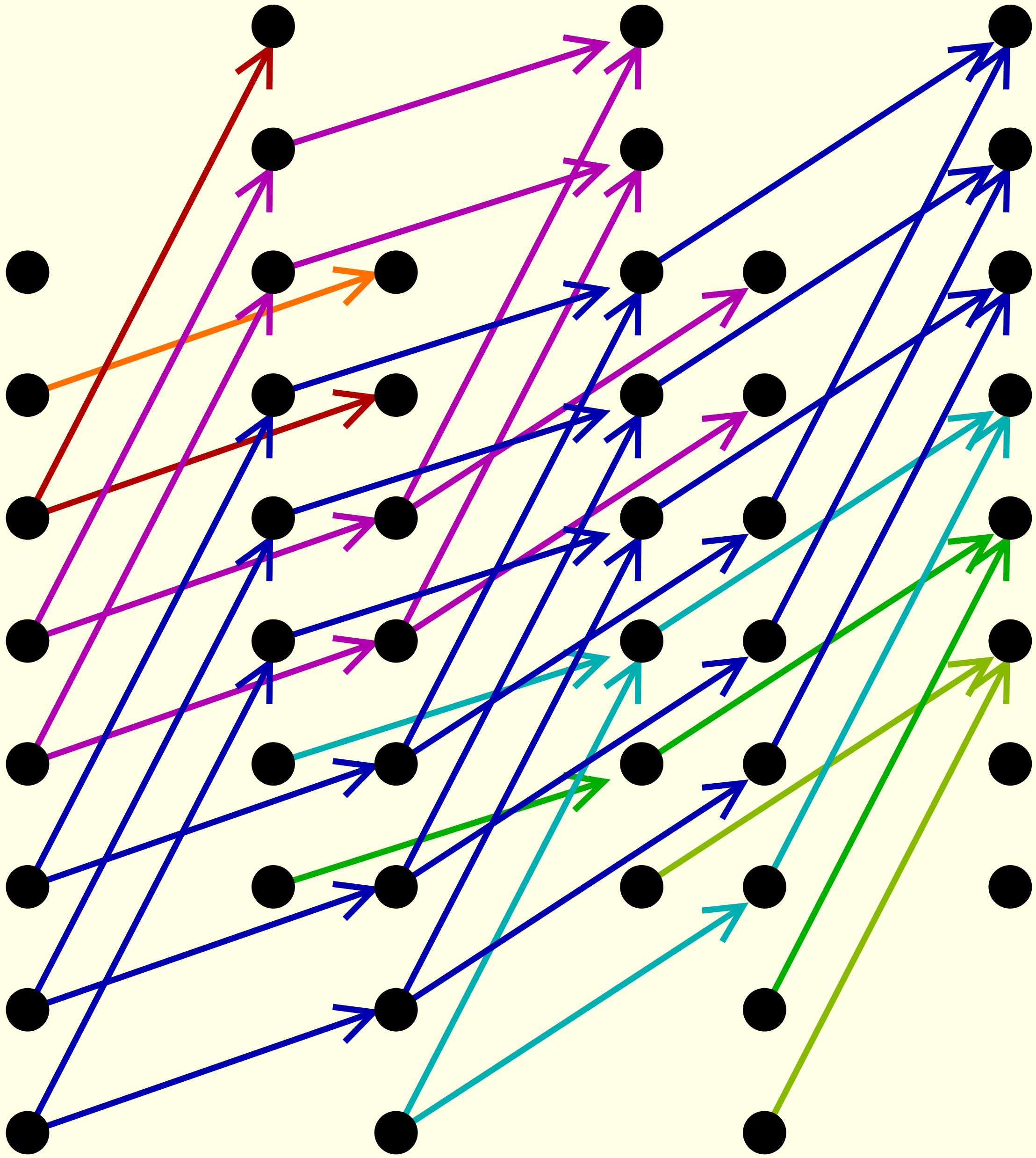
Proof Sketch



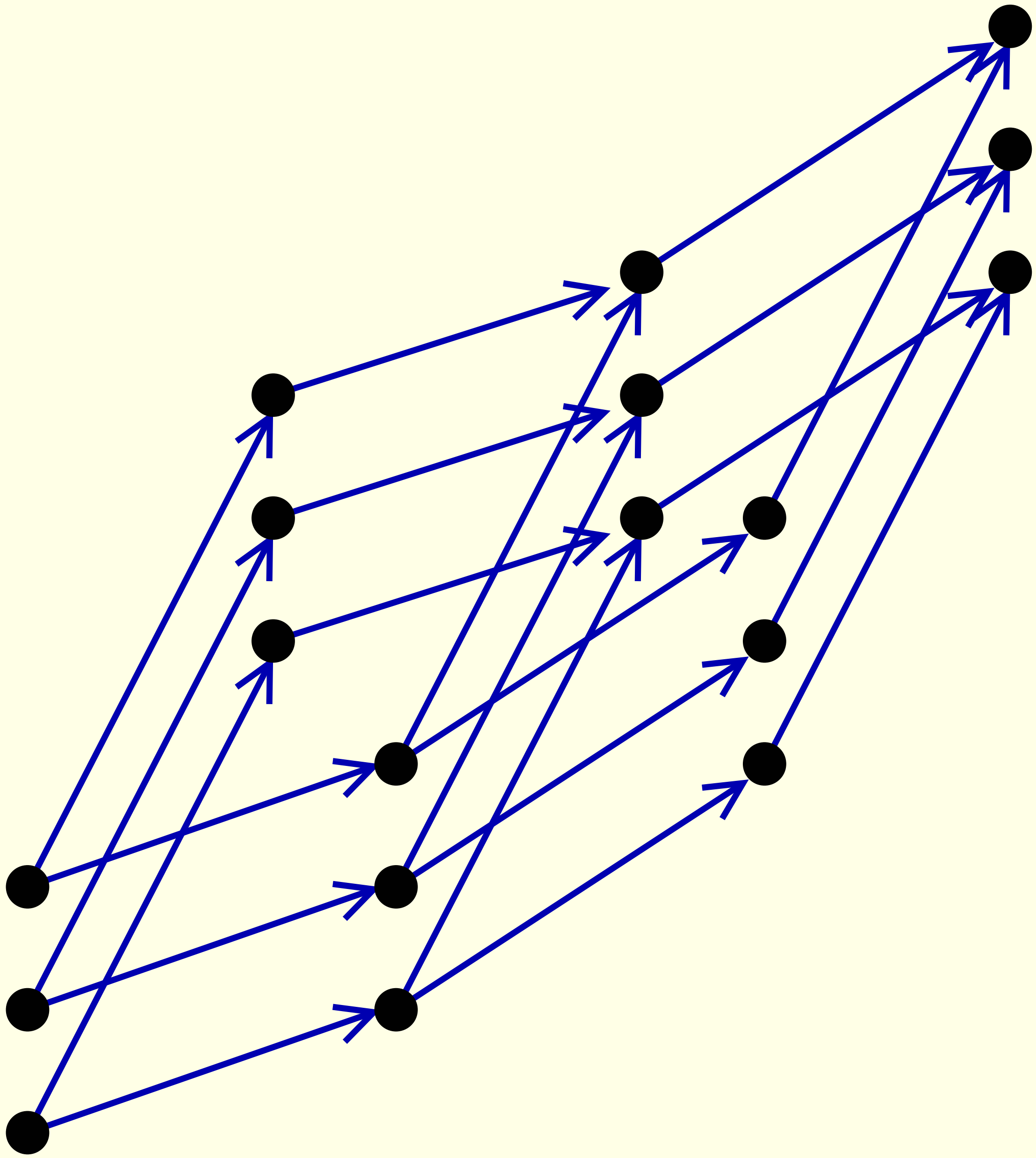
Proof Sketch



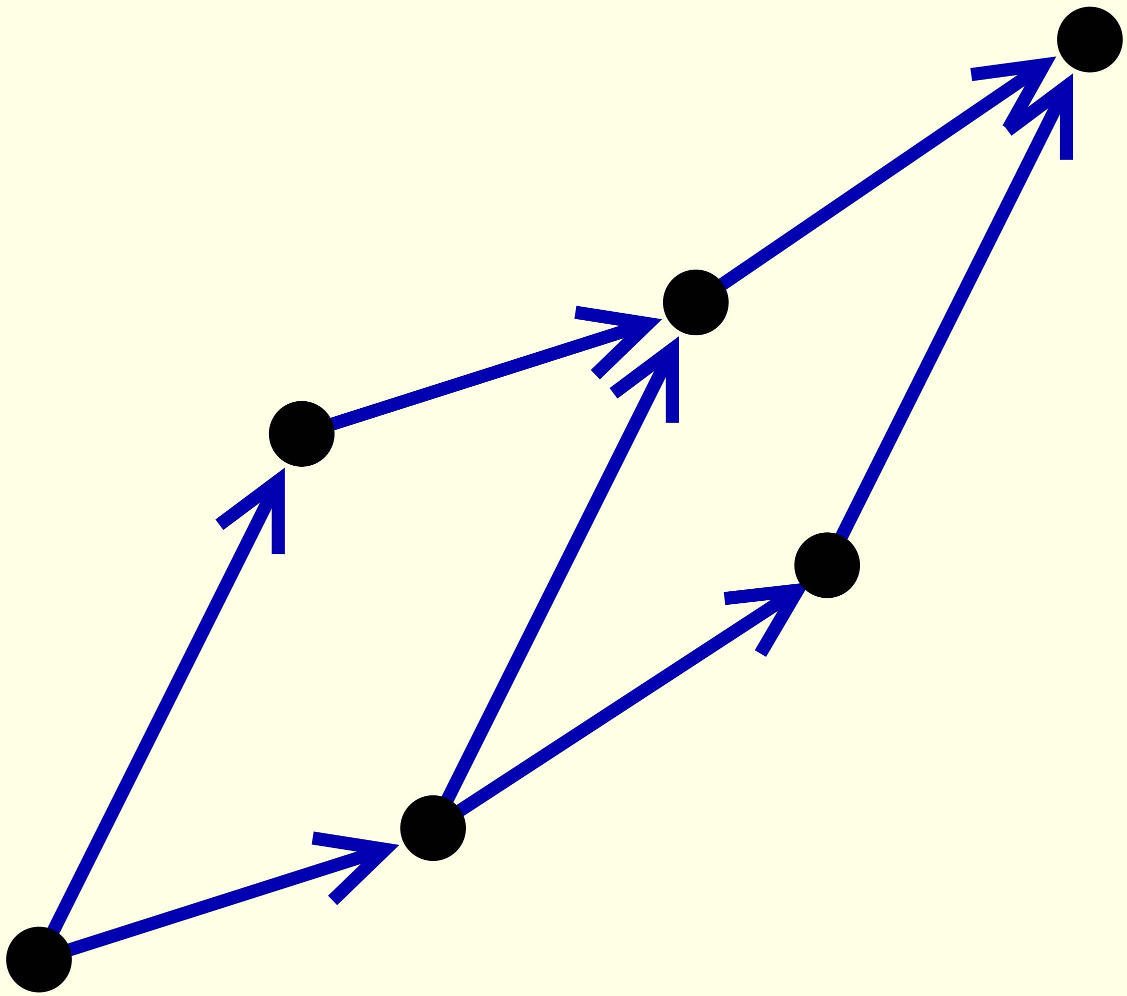
Proof Sketch



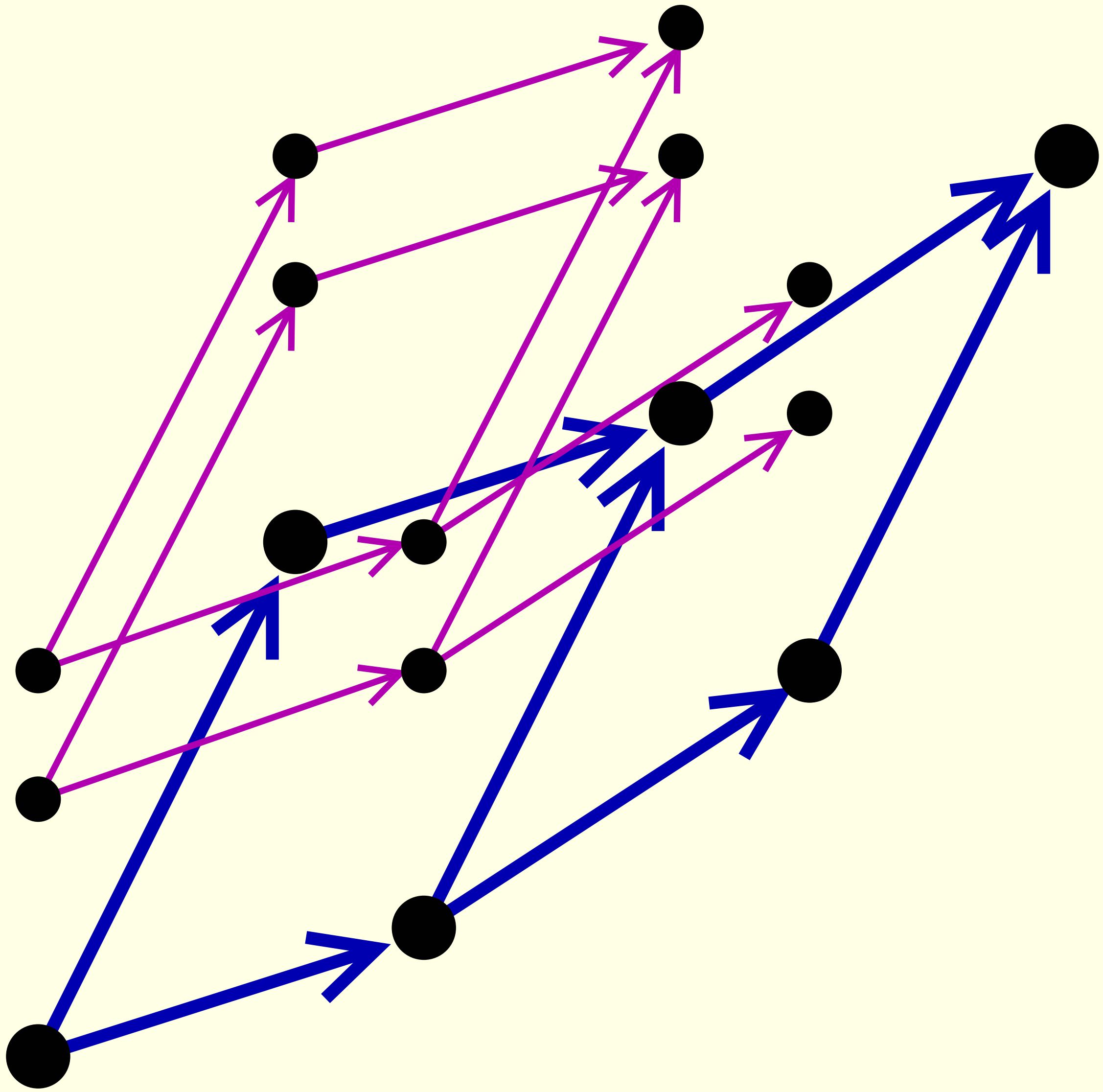
Proof Sketch



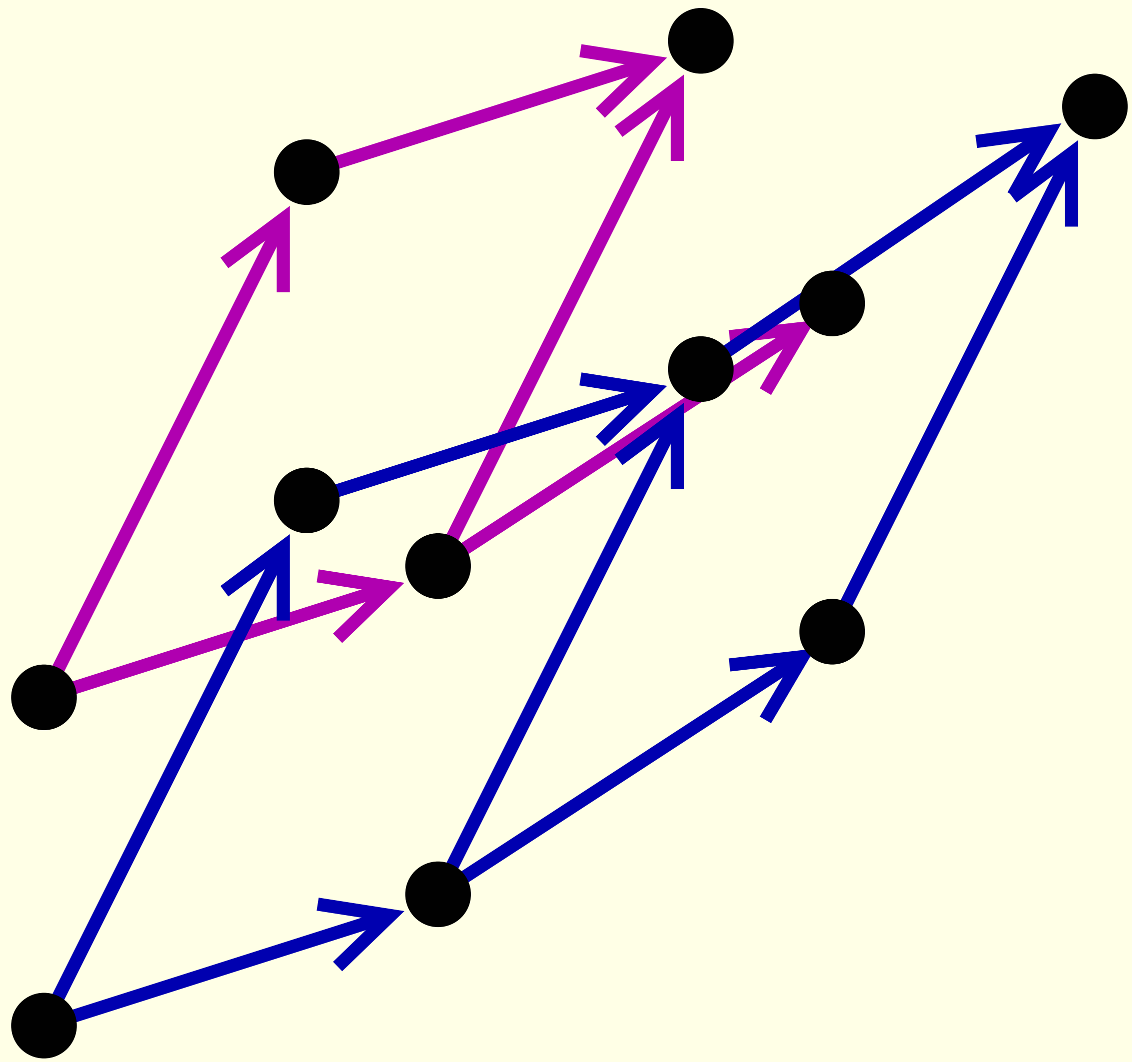
Proof Sketch



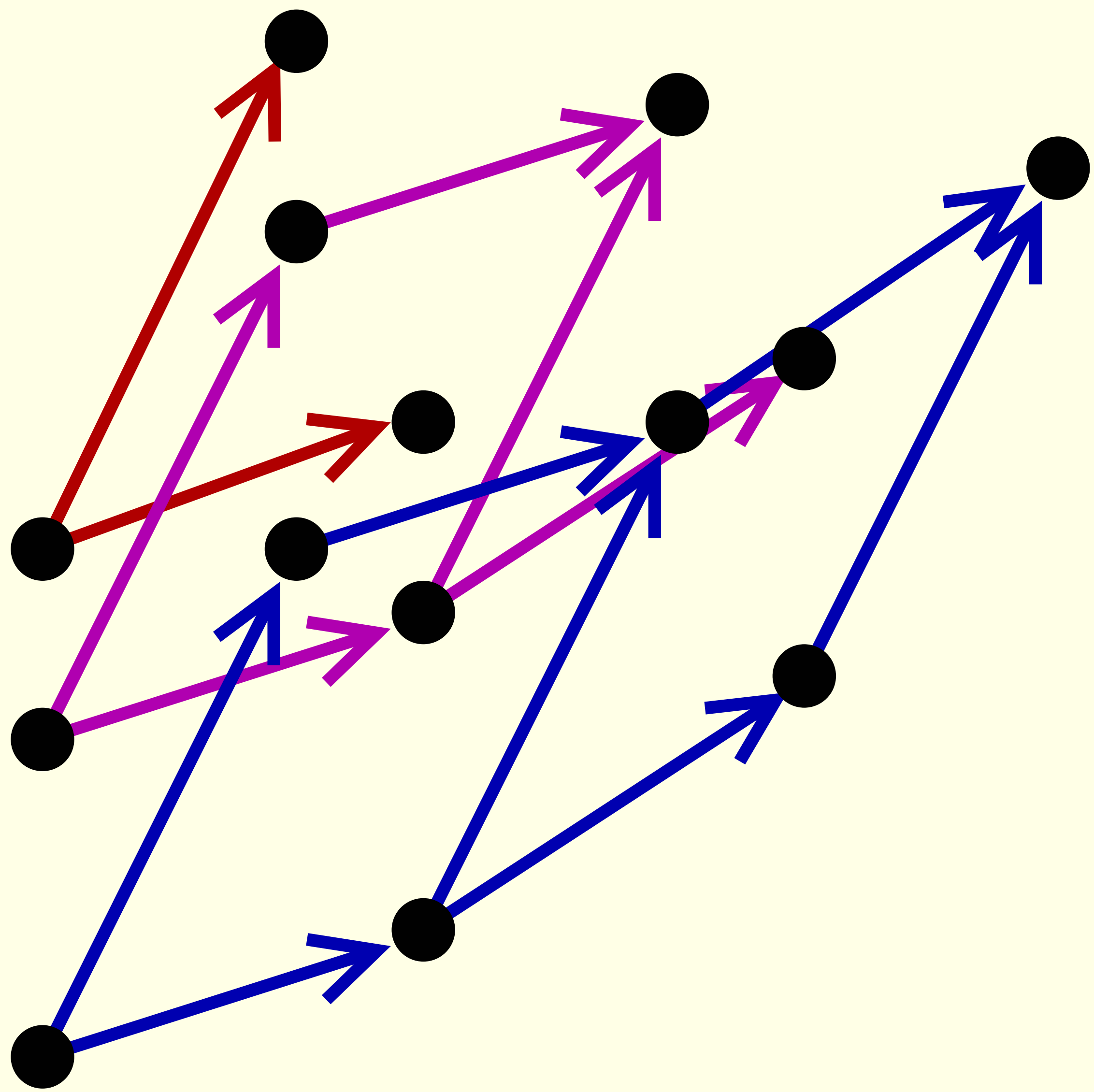
Proof Sketch



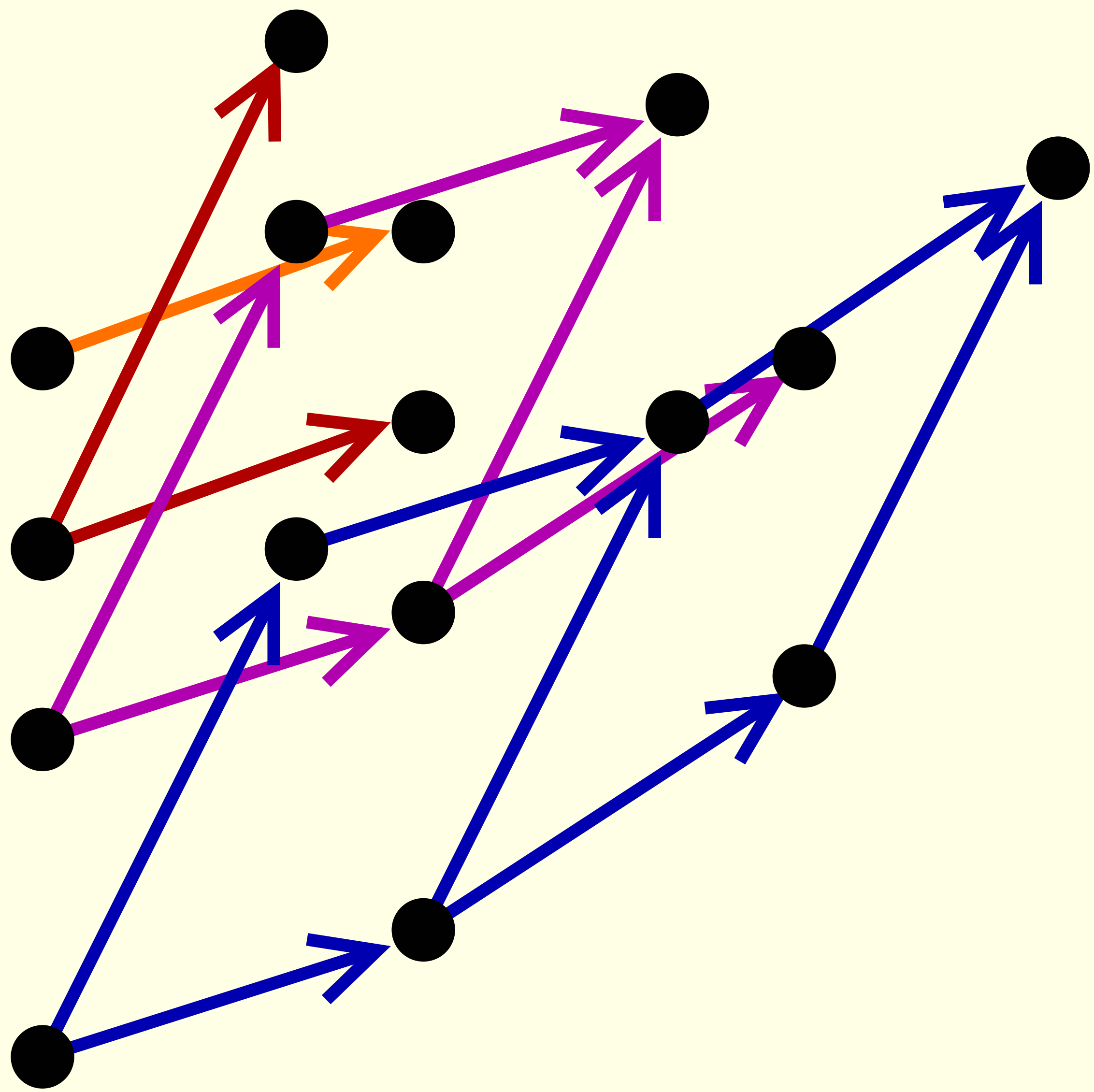
Proof Sketch



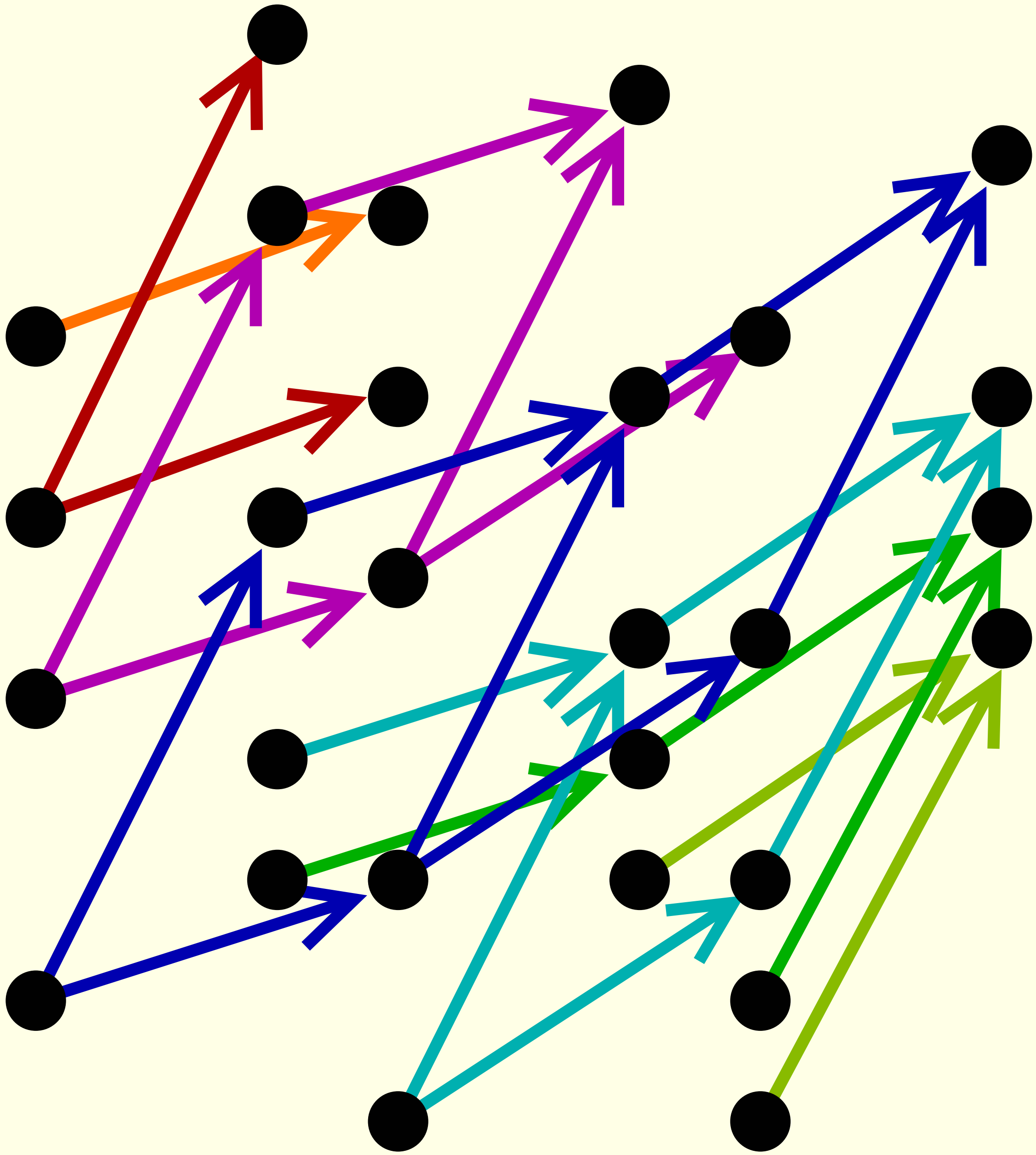
Proof Sketch



Proof Sketch



Proof Sketch



Bottom Line

Theorem.

If between every fixed pair of nodes all paths have the same transit time, an optimal flow over time can be obtained from a static flow computation in a time-expanded network with $O(n^2)$ nodes and $O(nm)$ arcs.

Interesting Special Case: Tree Networks!

The Quickest Flow Problem

Definition. (Quickest Multi-Commodity Flow Problem) Construct a multi-commodity flow over time satisfying given demands D within minimal time T (and cost bounded by C).

Burkard, Dlaska & Klinz (1993) use Megiddo's method of parametric search to give a strongly polynomial algorithm for quickest s-t-flows.

The Quickest Flow Problem

Definition. (Quickest Multi-Commodity Flow Problem) Construct a multi-commodity flow over time satisfying given demands D within minimal time T (and cost bounded by C).

Burkard, Dlaska & Klinz (1993) use Megiddo's method of parametric search to give a strongly polynomial algorithm for quickest s-t-flows.

—

The quickest flow problem with bounded cost and/or multiple commodities is NP-hard!

Approximation Algorithms

Joint work with Lisa Fleischer (IPCO'02 & SODA'03):

- Generalization of Ford & Fulkerson's approach: $(2+\varepsilon)$-approximation for quickest multicommodity flow based on length-bounded static flow computation.
- Introduce condensed time-expanded network with scaled transit times: General framework to obtain FPTASes for various quickest flow problems.
- Simple capacity scaling FPTAS for quickest min-cost s-t-flows with cost proportional to transit time.
- Important insight: Minimum convex cost transshipment over time never requires intermediate storage.

Static Average Flows

Given an optimal flow over time f^* with time horizon T^*, consider the corresponding static average flow x^* given by

$$x^* \;:=\; \frac{1}{T^*}\int_0^{T^*} f^*(\theta)\; d\theta \;\; .$$

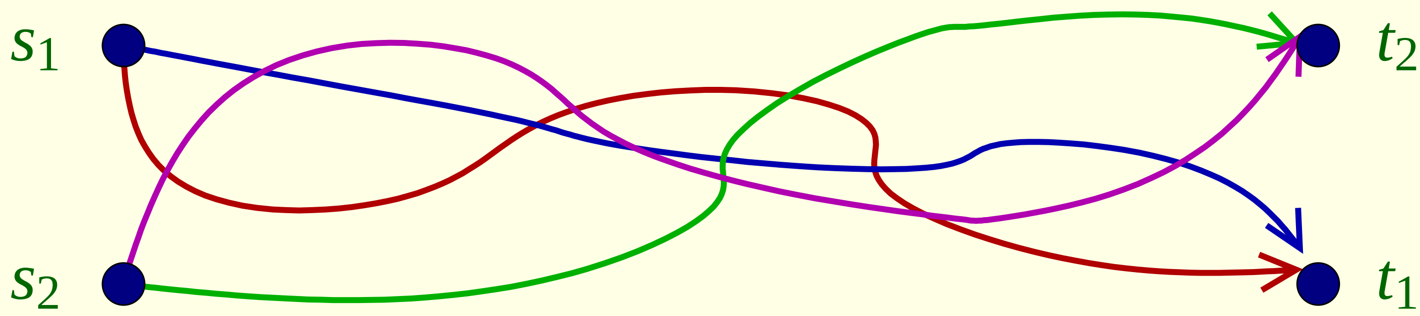
Then, x^* fulfills capacity and flow conservation constraints since f^* does.

Moreover,

$$|x^*| \;=\; \frac{|f^*|}{T^*} \;=\; \frac{D}{T^*} \quad \text{and} \quad c(x^*) \;=\; \frac{c(f^*)}{T^*} \;\leqslant\; \frac{C}{T^*} \;\; .$$

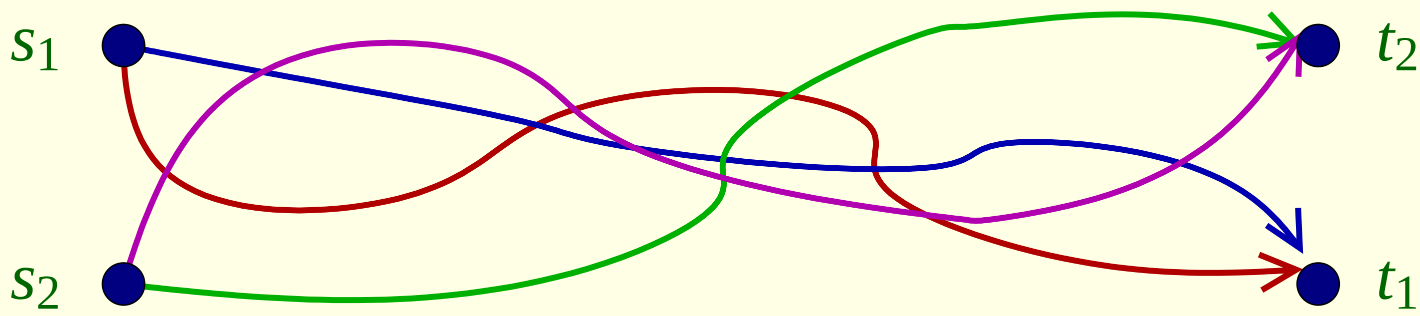
Length-Bounded Flows

Since f^* has time horizon T^*, any path P taken by an arbitrary flow unit has length $\tau_P \leqslant T^*$.



Length-Bounded Flows

Since f^* has time horizon T^*, any path P taken by an arbitrary flow unit has length $\tau_P \leqslant T^*$.



Observation.

Static average flow x^* is T^*-length-bounded, i. e., there is a path decomposition $(x_P^*)_{P \in \mathcal{P}}$ with $\tau_P \leqslant T^*$ for all $P \in \mathcal{P}$.

A Simple Algorithm

Problem: Find a quickest flow (i. e., minimize T) satisfying all demands D and with cost bounded by C.

Algorithm.

A Simple Algorithm

Problem: Find a quickest flow (i. e., minimize T) satisfying all demands D and with cost bounded by C.

Algorithm.

- Guess the optimal time horizon T^* (binary search).

A Simple Algorithm

Problem: Find a quickest flow (i. e., minimize T) satisfying all demands D and with cost bounded by C.

Algorithm.

- Guess the optimal time horizon T^* (binary search).
- Compute a T^*-length-bounded static flow $(x_P)_{P\in\mathcal{P}}$ with

$$|x| = \frac{D}{T^*} \qquad \text{and} \qquad c(x) \leqslant \frac{C}{T^*} \ .$$

A Simple Algorithm

Problem: Find a quickest flow (i. e., minimize T) satisfying all demands D and with cost bounded by C.

Algorithm.

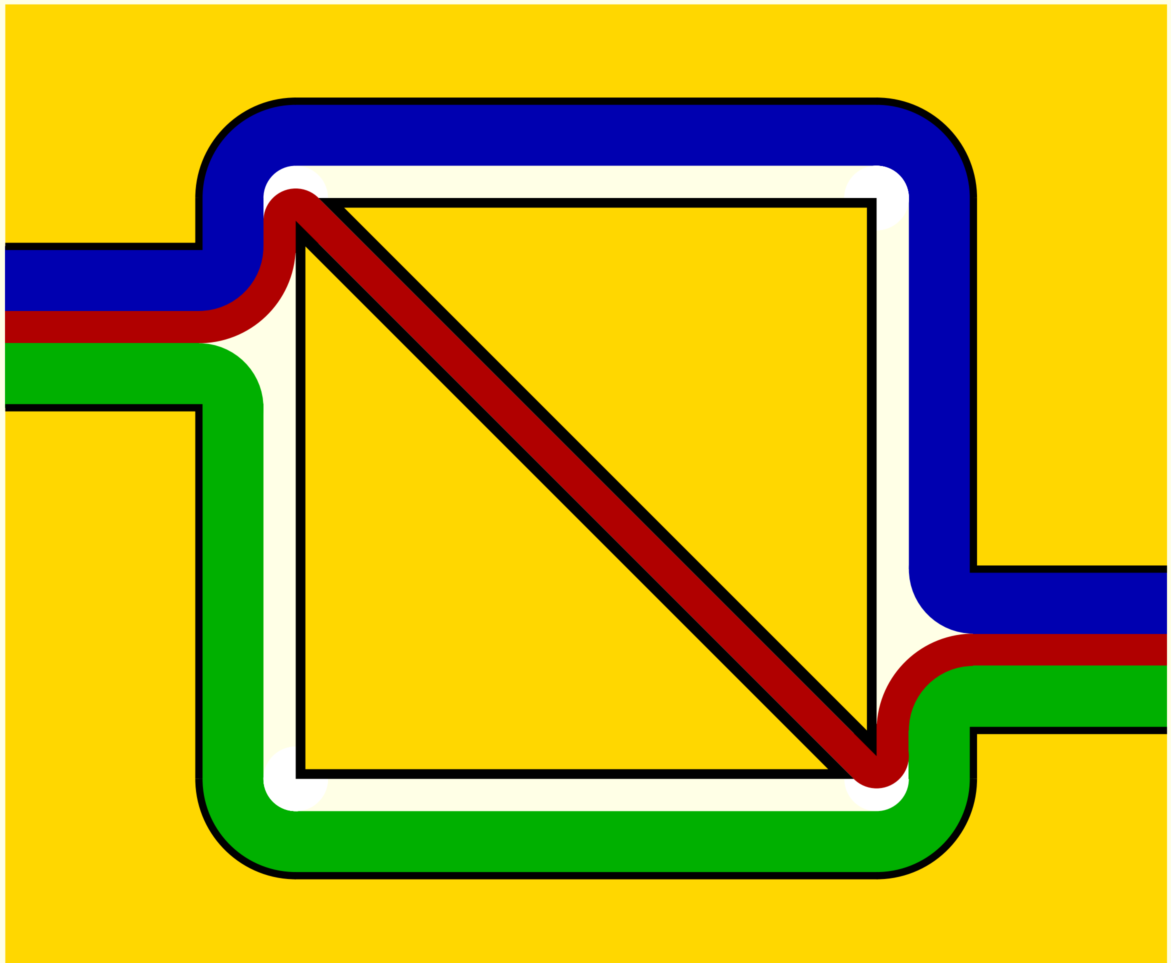
- Guess the optimal time horizon T^* (binary search).
- Compute a T^*-length-bounded static flow $(x_P)_{P \in \mathcal{P}}$ with

$$|x| = \frac{D}{T^*} \qquad \text{and} \qquad c(x) \leqslant \frac{C}{T^*} .$$

- Construct flow over time f by sending flow at constant rate x_P into paths $P \in \mathcal{P}$ during the time interval $[0, T^*)$. Then wait until all flow has arrived at the sink.

Example

The T^*-length-bounded static flow x:



Analysis

Flow value: The solution f sends flow according to a path decomposition of x into the network for T^* time units. Thus, $|f| = T^*|x| = D$.

Analysis

Flow value: The solution f sends flow according to a path decomposition of x into the network for T^* time units. Thus, $|f| = T^*|x| = D$.

Cost:

$$c(f) = \sum_{P \in \mathcal{P}} c_P T^* x_P = \sum_{P \in \mathcal{P}} \sum_{e \in P} c_e T^* x_P$$

$$= T^* \sum_{e \in A} c_e \sum_{\substack{P \in \mathcal{P} \\ e \in P}} x_P = T^* \sum_{e \in A} c_e x_e = T^* c(x) \leqslant C$$

Analysis

Flow value: The solution f sends flow according to a path decomposition of x into the network for T^* time units. Thus, $|f| = T^*|x| = D$.

Cost:

$$c(f) = \sum_{P \in \mathcal{P}} c_P T^* x_P = \sum_{P \in \mathcal{P}} \sum_{e \in P} c_e T^* x_P$$
$$= T^* \sum_{e \in A} c_e \sum_{\substack{P \in \mathcal{P} \\ e \in P}} x_P = T^* \sum_{e \in A} c_e x_e = T^* c(x) \leqslant C$$

Time horizon: The flow over time f sends flow into paths $P \in \mathcal{P}$ until time T^*. Since $\tau_P \leqslant T^*$ for all $P \in \mathcal{P}$, the last unit of flow arrives at the sink before time $2T^*$.

Approximation Result

Theorem:

The algorithm achieves performance ratio 2.

Approximation Result

Theorem:

The algorithm achieves performance ratio 2.

Problem:

Computing a T^*-length-bounded static flow is NP-hard.

Approximation Result

Theorem:

The algorithm achieves performance ratio 2.

Problem:

Computing a T^*-length-bounded static flow is NP-hard.

Solution:

For any $\varepsilon > 0$, a $(1+\varepsilon)T^*$-length-bounded static flow can be computed in polynomial time. (Dual separation is length-bounded shortest path problem $\longrightarrow$ FPTAS.)

$\Longrightarrow$ Polynomial time algorithm with performance $2+\varepsilon$.

Concluding Remarks

- Flows over time are of great practical importance.
- Our theoretical and practical understanding of the subject is not satisfactory yet.

Concluding Remarks

- Flows over time are of great practical importance.
- Our theoretical and practical understanding of the subject is not satisfactory yet.
- In real-world situations, transit times vary with the amount of flow on an arc.

  Problem: Find a realistic and computationally tractable mathematical model!

  $\longrightarrow$ Next talk!