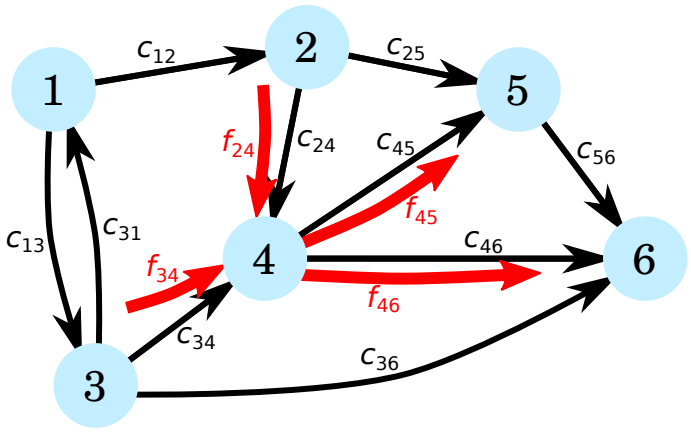
Tutorial 4

Make sure you prepare these BEFORE the class.

Solutions will be handed out at the tutorial. They will not be put on MyUni.

1. **Translation:** A *flow network* is a directed graph $G = (V, E)$ where each edge (u, v) has a capacity $c_{u,v}$, and each edge could receive a flow $f_{u,v}$ of some commodity (*e.g.,* electricity or traffic). See the figure below for an example.



A flow network is constrained in various ways:

- Flows are non-negative.
- The flow on an edge may not exceed the capacity. For example, in the figure, we must choose $f_{24} \leq c_{24}$. In particular, if no edge exists (*e.g.,* between nodes 1 and 4), this flow must be zero.
- Flows are conserved, *i.e.,* everything that flows into a node must also flow out, except at a source or sink, For instance, in the figure

$$f_{34} + f_{24} = f_{45} + f_{46}.$$

- The total flow in at the source must be balanced by the flow out at the sink.

The *maximum flow* problem is the problem: given source node s and destination sink t, what is the maximum flow that can pass across the network between these two (any flow that enters at the source must exit at the sink).

(a) Formulate this problem as a LP in general terms, *i.e.,* write the objective and constraints as a set of general equations or inequalities (you don't need to translate into standard form yet).

(b) Take the particular problem above (assuming all possible flows, not just those illustrated), and write it in a form suitable for input to MATLAB, *i.e.,* work out the matrices A and A_{eq} and vectors $\mathbf{b}$, $\mathbf{b}_{eq}$ and $\mathbf{c}$, assuming that node 1 is a source (*i.e.,* it can supply an arbitrary amount of flow) and 6 is a sink (it can absorb an arbitrary amount). You should have 1 variable for each edge, and 1 constraint for each node (conservation) and for each edge (capacity).

(NB: don't solve – there are better approaches than just solving directly as a LP, *e.g.,* the Ford-Fulkerson algorithm).

2. **Interpretation:** Describe in words the optimisation problem specified by the AMPL files:

Model file.

```
set MEDIA;

param total_budget;
param cost{MEDIA};
param audience{MEDIA};
param conversion{MEDIA};

var number{MEDIA} >=0 integer;

maximize value: sum{i in MEDIA} audience[i]*conversion[i]*number[i];

subject to total_cost: sum{i in MEDIA} cost[i]*number[i] <= total_budget;
```

Data file.

```
set MEDIA := radio tv magazines newspapers;

param total_budget := 100; # thousands of $
param:        cost audience conversion :=
  radio          3     100     0.03
  tv            50     350     0.09
  magazines      2      40     0.02
  newspapers     1      60     0.01;
```

Explain what output you might expect if we displayed `number`.

3. **Calculations:** Consider the primal LP

$$\begin{array}{lrrl} \text{(P)} & \max & z & = 3x_1 - x_2 + 7x_3 \\ & \text{subject to} & -x_1 + x_2 + 2x_3 & \leq 8 \\ & & x_1, x_2, x_3 & \geq 0 \end{array}$$

   (a) Write down the dual.

   (b) Show by inspection that the dual is infeasible.

   (c) What can you conclude about the solution to (P)?

   (d) Now describe the relationship between primal and dual again if we restricted the variables to be integers.

4. **Derivation of the week:**

   The following describes a greedy algorithm for finding an approximate solution to the Travelling Salesperson Problem: start from an arbitrary city (say city 1), and then go to the nearest city not yet visited. When you have visited all cities return to the start.

   Derive the computational complexity of this algorithm, assuming the distances are all stored in a large array.