Many examples:

- Intrinsically combinatorial

A railway network, directed from (say) producer (S) to consumer (E), with capacities
- Cut edges to disconnect S and E
- Find the largest flow between S and E

As an optimization problem

\[
\max \quad x^T A x + b^T x \\
\text{s.t.} \quad x \in \{0, 1\}^n \quad \text{i.e. 0-1 vectors} \\
C x = d
\]
\[
\begin{align*}
\text{max} & \quad x^T A x + b^T x \\
\text{st} & \quad x \in \mathbb{Z} \quad \text{integers} \\
& \quad C x = d
\end{align*}
\]

Some tools:
Dynamic programming.

Easy case: (setup in AML notes)
\[
\text{Cost} (x_1, \ldots, x_K) = F(x_1, x_2) + F(x_2, x_3) + \ldots + F(x_{K-1}, x_K) + G(x_1) + G(x_2) + \ldots + G(x_K)
\]

Q: Find \( x_1, \ldots, x_K \in \mathbb{Z}^k \) to max cost.
A: Easier with a picture.
A trellis

\[
\begin{array}{cccc}
1 & 0 & 0 & \cdots \\
0 & 0 & 0 & \cdots \\
0 & 0 & 0 & \cdots \\
0 & 0 & 0 & \cdots \\
0 & 0 & 0 & \cdots \\
0 & 0 & 0 & \cdots \\
2 & 0 & 0 & \cdots \\
\vdots & \vdots & \vdots & \ddots \\
\end{array}
\]

- one col for each \( x_k \); one node for each value of \( x_k \)
- label nodes \( G(x_{*i} = m) \)
- directed edges \( F(x_i = m, x_{i+1} = n) \)

Then: the cost of directed path \( 1 \rightarrow k \)
\[
= \text{cost of } F(x_1 = a, x_2 = b, \text{ etc.})
\]
Pretty clearly, this won't work for

\[ F(x_1, x_2) + F(x_2, x_3) + F(x_1, x_3) \]

(try it!)

Criterion

- Each var gets a node
- If each pair of vars gets an edge if there is an \( F(x_i, x_j) \) term
- Resulting graph is a forest

\( \iff \) D.P. will work.
An abstracted picture of a railway network from $S$ to $T$

- each edge has a capacity, the most stuff you can move along that edge
- at each node, the amount of stuff arriving must be the same as the amount leaving - no storage! [Kirchhoff's laws]
- Q: how much stuff can go from $S$ to $T$?
Another case:

- Assume we have a directed graph $G$, with two distinguished vertices $S$ - source, only outgoing edges, $T$ - target, only incoming edges.
- Each edge has a non-negative capacity $c_e > 0$.
- A flow from $S$ to $T$ is a labelling of each edge $e$ with a weight $w_e$ such that:
  - Kirchhoff's laws are satisfied at all but $S$ and $T$.
  - $w_e < c_e$. 
We can set this up as a linear program.

. Each edge \((u \rightarrow v)\) gets \(w_{u \rightarrow v}\)

. Kirchhoff’s laws apply, except at \(S, T\)

\[
\sum_u \omega_{u \rightarrow v} - \sum_w \omega_{v \rightarrow w} = 0 \\
\text{for all } v \neq S, T
\]

. \(\frac{\omega}{\beta} \leq c_{u \rightarrow v}\)

. \(\omega_{u \rightarrow v} > 0\)

Total flow is:

\[
\sum_v \omega_{S \rightarrow v}
\]

[Recall Kirchhoff’s laws!]

Now imagine trying to cut this network.

- Construct two sets of edges

\[ S \cap T \]

where \( S \cup T = V \)
\[ S \cap C = \emptyset \]
\[ s \in S , \ t \in T \]

AND there By cutting edges from \( S \) to \( T \)

\[ S \rightarrow \]
\[ \times - \text{ cut edges} \]
Do this by cutting a set of edges w/ minimal capacity — min cut.

One formulation:

\[ x_v \]

one per vertex,

\[ x_s = 1, \quad x_t = 0 \]

\[ x_v = 1 \iff v \in S \]

\[ x_v = 0 \iff v \in T \]

Cost function:

\[ \sum c_{uv} \cdot (x_u)(1-x_v) \]

Constraints:

\[ x_u \in \{0, 1\} \]

\[ x_s = 1 \]

\[ x_t = 0 \]
Notice the cost is quadratic.

but if \( x_u \in \{0,1\} \) \( \text{and} \) \( x_v \in \{0,1\} \)

then \( x_v x_u \in \{0,1\} \).

Notice:

\[
\begin{align*}
x_u x_v & \leq x_u \\
x_u x_v & \leq x_v \\
x_u x_v & \leq x_u + x_v - 1
\end{align*}
\]

So make a new variable \( q_{uv} \in \{0,1\} \)

which represents \( x_{uv} \)

leading to
\[- \sum_{uv} c_{uv} q_{uv} + \left[ \sum_u \theta c_{uv} x_u \right] (u - 2) \]

s.t.

\[ q_{uv} \in \{0, 1\}, \; x_u \in \{0, 1\}, \; x_v \in \{0, 1\} \]

\[ q_{uv} - x_u \leq 0 \]
\[ q_{uv} - x_v \leq 0 \]
\[ q_{uv} - x_u - x_v + 1 \geq 0 \]

Now what?

Turns out, rather remarkably, this is easy.