Max flow. Given a flow network $G = (V, V \times V)$ with edge capacities $c : V \times V \to \mathbb{R}$, source s, and sink t, here is a linear program for the max-flow problem:

$$\text{maximize } f(t,s)$$
$$\text{subject to} \begin{cases} \sum_{v \in V} f(w,v) - \sum_{u \in V} f(u,w) & = & 0 & \forall w \in V \\ f(u,v) & \leq & c(u,v) & \forall u,v \in V, (u,v) \neq (t,s) \\ f(u,v) & \geq & 0 & \forall u,v \in V. \end{cases}$$

There is one variable $f(u,v)$ for each edge (u,v) representing the flow on the edge. The "artificial" edge (t,s) is there to simplify things a bit — it allows a simple expression for the value of the flow $(f(t,s))$ and allows conservation to hold at all vertices.

Recall that if the capacities are integers, then there is always a maximum flow that is integer-valued.

Shortest path. Given a graph $G = (V, V \times V)$ with edge weights $w : V \times V \to \mathbb{R}$, source s, and sink t, here is a linear program for the shortest $s - t$ path problem:

$$\text{minimize } \sum_{u,v} w(u,v) f(u,v)$$
$$\text{subject to} \begin{cases} \sum_{v \in V} f(w,v) - \sum_{u \in V} f(u,w) & = & 0 & \forall w \in V \\ f(t,s) & = & 1 & \\ f(u,v) & \geq & 0 & \forall u,v \in V. \end{cases}$$

This represents a flow of one unit of substance from s to t. The way to understand the relation to shortest paths is that any $s - t$ flow can be decomposed into a collection of $s - t$ paths, each with some flow along them (this takes some proof). This means that the vertices of the polytope defined by the linear program correspond to flows that represent paths. Thus, there is always an optimal solution corresponding to a single path (i.e. a vertex).

Dual of max-flow. The dual of the max-flow linear program is the following:

$$\text{minimize } \sum_{(u,v) \neq (t,s)} c(u,v) \delta(u,v)$$
$$\text{subject to} \begin{cases} \delta(u,v) & \geq & \pi(u) - \pi(v) & \forall (u,v) \neq (t,s) \\ \pi(s) - \pi(t) & \geq & 1 & \\ \delta(u,v) & \geq & 0. & \end{cases}$$

There is one variable $\pi(u)$ for each vertex u and another variable $\delta(u,v)$ for each edge (u,v) other than the "artificial" edge (t,s). To see the relation to max-cut, let $(S, T = V - S)$ be any $s - t$ cut, then set

$$\pi(u) = \begin{cases} 1 & \text{for } u \in S \\ 0 & \text{for } u \in T \end{cases}$$

and let

$$\delta(u,v) = \begin{cases} 1 & \text{for } (u,v) \in S \times T \\ 0 & \text{otherwise.} \end{cases}$$

Then the cost of the solution is the capacity of the cut (S,T). Again, in this case, these solutions correspond to vertices of the polytope, so that there is always an optimal solution of this form.

Dual of shortest path. The dual of the shortest-path linear program is equivalent to the following:

$$\begin{aligned} &\text{maximize } \pi(t) - \pi(s) \\ &\text{subject to } \left\{ \begin{array}{rcl} \pi(v) - \pi(u) & \leq & w(u,v) \end{array} \right. \end{aligned}$$

There is one variable $\pi(u)$ for each vertex u. An interpretation of this linear program is that the graph is being embedded on a line, with $\pi(u)$ representing the position of the vertex u on the line, subject to the constraints that for every edge (u, v), there is a "string" preventing vertex v from being put more than $w(u, v)$ units further along the line than vertex u.

Dijskstra's shortest path algorithm has a nice interpretation this way: replace each edge in the graph by a string of the appropriate length, then, starting with all vertices in a pile on the table, gradually lift up the vertex s, letting the other vertices be pulled down towards the table by gravity. As s lifts, the strings will cause the other vertices to lift too. A vertex u will start lifting when the strings along the shortest path from s to u are all "tight".

Linear programming duality is often useful in designing algorithms for combinatorial optimization problems.