

CS 491G Combinatorial Optimization

Lecture Notes

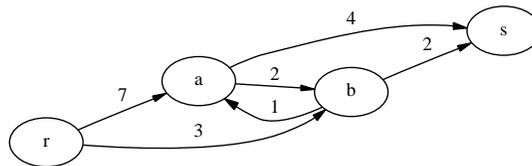
David Owen

June 11, 13

1 Maximum Flow Problems

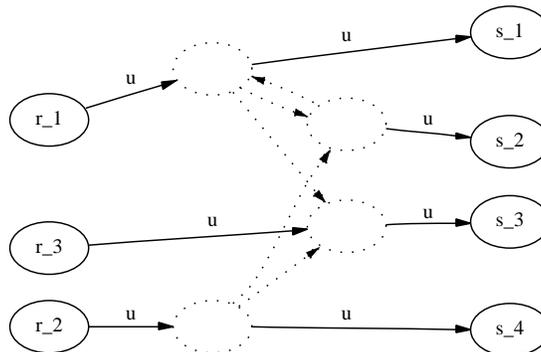
Given a directed graph with weighted edges, we would like to know the maximum flow possible between two special vertices in the graph, the source vertex r and the sink vertex s ; that is, we imagine some fluid being pumped from r through the graph to s , and we think of edge weights as capacities limiting the amount of fluid that may be pumped through that edge. We assume that an infinite supply is available at r , and that s can consume an infinite amount, so that only edge capacities (we will call the edge weights “capacities” from now on) limit flow through the graph. We also restrict all edge capacities and flows to integers, although in general the *Max-Flow* problem may be solved for any edge capacity and flow values.

For example, we can solve *Max-Flow* by inspection for the following simple graph:

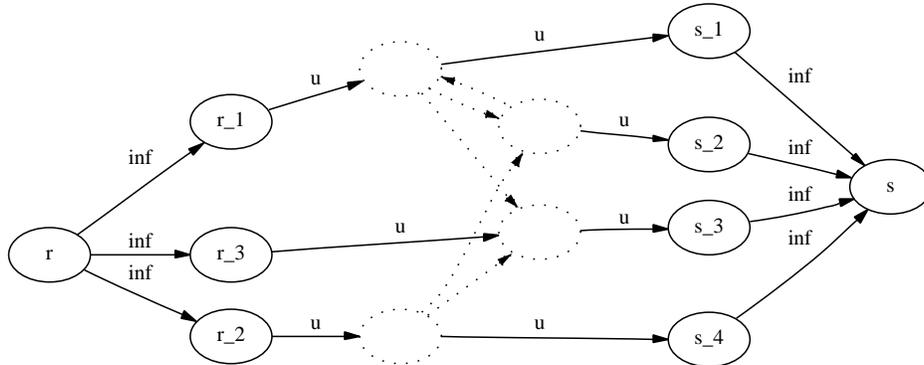


There are only two edges connected to s (e_{as}, e_{bs}), so the *Max-Flow* can not exceed the sum of these edges' capacities ($4 + 2 = 6$). We can pump 4 to a through e_{ra} without exceeding its capacity (7), and we can pump 2 to b without exceeding its capacity (3). With 4 available at a and 2 available at b , we can fill e_{as} and e_{bs} to capacity; therefore the *Max-Flow* = 6.

Edge capacities may be $= \infty$; if so that edge will not limit flow. This fact can be used to transform a graph with more than one source or sink (or more than one of both) into a single-source, single-sink problem. We start with the graph below, in which u indicates an arbitrary edge capacity, and the dotted vertices and edges indicate some arbitrary subgraph between multiple source vertices $r_1 \dots r_3$ and multiple sink vertices $s_1 \dots s_4$:

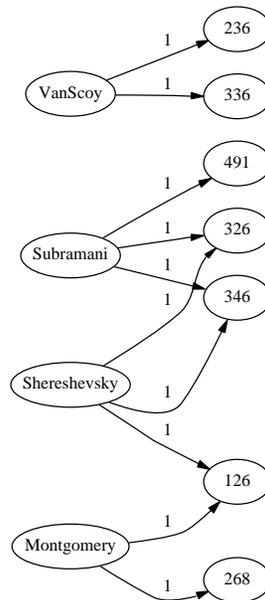


If we add a “super-source” r and a “super-sink” s to the graph, and if we connect r to the sources and s to the sinks already in in the graph via infinite-capacity edges, we have the following single-source, single-sink graph (since flow is only limited by the finite-capacity edges from the previous graph, our *Max-Flow* solution for this graph would be the same as for the previous):

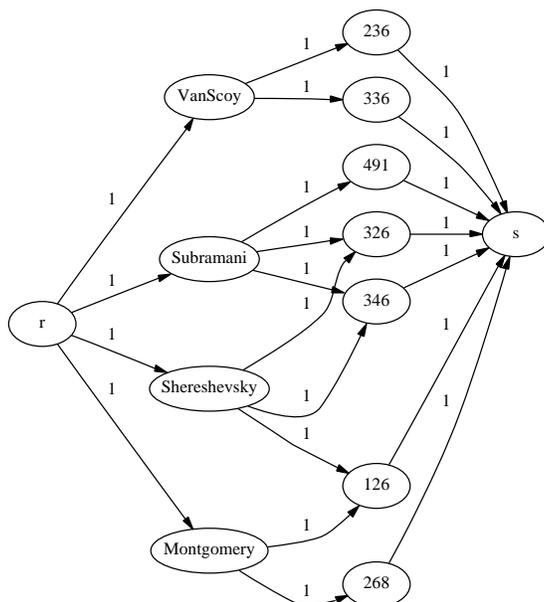


1.1 Example: Professors’ Teaching Assignments

Suppose we have a list of professors and a list of courses in need of a professor for the coming semesters. The professors have indicated which of the courses they would be willing to teach. We would like to assign one course to each professor such that *a)* at most one professor is assigned to any course, and *b)* professors are only assigned to courses they would be willing to teach. We can model the problem as a directed graph so that the *Max-Flow* solution for that graph is the assignment solution we are looking for. First of all, we put professors on the left and courses on the right, with a (unit-capacity) directed edge from a professor to a course indicating the professor would be willing to teach the course:



We now add a source vertex r and a sink vertex s connected in the following way via unit-capacity edges:



Since a maximum of 1 may flow into each professor vertex (limited by the capacity of edges coming from r), a maximum of 1 may flow out of each professor vertex. And we have restricted flows to integers, so that the flow out of each professor vertex to any particular course must be 0 or 1. This means there will be at most one edge through which there is a flow from a particular professor vertex to any course vertices (i.e. a professor will not be assigned more than one course). Flows out of course vertices to s must also be 0 or 1, since the capacity of edges from courses to s is 1. And because there is a single edge coming out of each course vertex, a maximum flow of 1 may go through that vertex (i.e. no course may be assigned more than one professor). In this case our *Max-Flow* solution = 4, which means that an assignment exists in which all four professors will be assigned courses they have indicated they are willing to teach.

1.2 Background and Definitions

It turns out that we do not need to stipulate both integral capacities and integral flows. We have the following lemma (we will not prove it here):

Lemma

If all edge capacities are integers, the *Max-Flow* must also be integral.

For an integral flow = k , there is a “family” of k (r - s) dipaths (not necessarily distinct dipaths beginning at r and ending at s) $\{P_1 \dots P_k\}$ such that:

$$|\{i : P_i \text{ uses edge } e\}| \leq u_e$$

That is, the number of P_i 's using edge e is \leq the capacity of e (u_e). We can think of these paths as each carrying a unit flow from r to s . There can not be more than u_e unit flows going through e , so at most u_e of these paths may use e . And if the total flow is k , there must be k unit flows (and therefore k of these paths) going from r to s .

We let x_e be = to the number of P_i 's using edge e :

$$x_e = |\{i : P_i \text{ uses } e\}|$$

So we have an x_e for every edge; that is, \vec{x} assigns a number to every edge. We have stated above that for a flow k there must be k (r - s) dipaths, that x_e is equal to the number of those dipaths using edge e . But what if we are given \vec{x} by itself and want to know whether it represents a legitimate flow vector? The following condition must hold:

\vec{x} is a flow if:

$$\sum(x_{wv} : w \in V, e_{wv} \in E) - \sum(x_{vw} : w \in V, e_{vw} \in E) = 0, \forall v \in V \setminus \{r, s\}.$$

The amount coming into vertex w (the first $\sum(\dots)$) minus the amount going out (the second $\sum(\dots)$) must be zero—nothing can be stored in w . We call this “conservation of flow.” We call flows “feasible” if they satisfy the condition above, \vec{x} values are positive, and no edge capacities are exceeded (the simplest feasible flow $\vec{x} = \vec{0}$):

\vec{x} is a feasible flow if:

$$\vec{x} \text{ is a flow, and } 0 \leq x_{vw} \leq u_{vw}.$$

When given a flow \vec{x} through some graph G , we represent the edges of G as (u_e, x_e) , and we define $f_{\vec{x}}(v)$ as the net flow into vertex v ; the net flow into s , $f_{\vec{x}}(s)$, is the flow we would like to maximize. Using this notation we formulate *Max-Flow* as a linear program:

$$\begin{array}{ll} \max & f_{\vec{x}}(s) \\ \text{s.t.} & f_{\vec{x}}(v) = 0, \quad \forall v \in V \setminus \{r, s\} \\ & 0 \leq x_e \leq u_e \end{array}$$

We can formulate *Max-Flow* as a linear program, but we would like to find a simpler way to solve it. With that in mind, we propose the following connection between the P_i dipaths and flow value k discussed above:

(Proposition 3.1 in the text [1, p.38])

(a) \exists a family of (r - s) dipaths $(P_1 \dots P_k)$ such that $|\{i : P_i \text{ uses edge } e\}| \leq u_e, \forall e \in E$
if and only if (b) \exists an integral feasible (r - s) flow of value k .

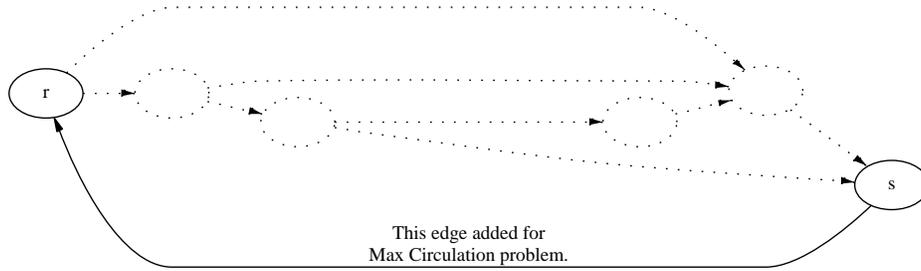
The proof that (a) \Rightarrow (b) should be obvious: we can use the left-hand side of the capacity limitation for each path P_i ($|\{\dots\}|$ above) as the value for each x_i in \vec{x} , so that no edge’s capacity is exceeded and the sum of all x_i ’s = k .

Now we must prove that (b) \Rightarrow (a). We are given an integral feasible flow value k . If $k = 0$, we have zero P_i ’s and the proposition is trivial. If $k \geq 1$, there must be at least one edge of at least unit-capacity coming into s . If that edge exists (we will call it e_{vs}), there must be at least one edge of unit capacity coming into its beginning vertex v , and there must be some edge coming into that edge’s beginning vertex, etc., all the way back to r . So if $k \geq 0$ there must be at least one dipath P_i . We can eliminate that path from the graph and let $k = k - 1$. If k is still ≥ 0 , we repeat the process again and again until $k = 0$. At that point we have eliminated k dipaths $P_1 \dots P_k$. We note the following related lemma:

Lemma (which we will not prove here)

Every feasible flow is the sum of at most m flows.

We also note that a “maximum circulation” problem related to *Max-Flow* may be modeled by adding an edge from s back to r and extending the flow conservation condition to these vertices:



1.3 Maximum Flows and Minimum Cuts

Cut (for digraphs):

Given a set $R \subseteq V$, the set $\delta(R) = \{e_{vw} : v \in R, w \notin R\}$ is called a “cut.”

So a cut is a set of edges beginning in R but ending outside of R (in \bar{R}).

(r - s) Cut:

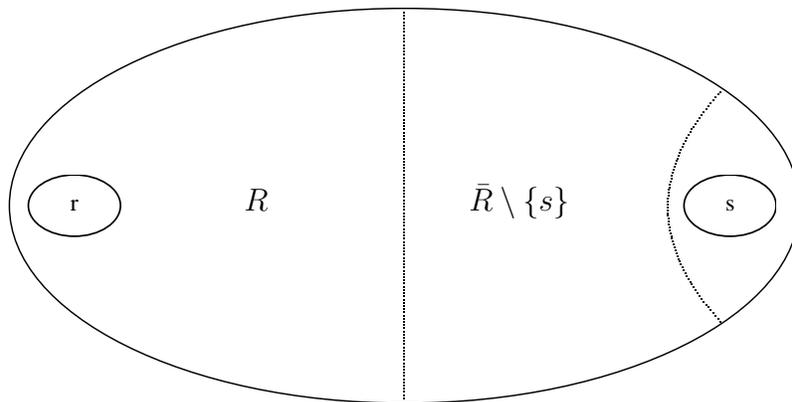
A cut $\delta(R)$ such that $r \in R, s \notin R$.

And an (r - s) cut is a set of edges as defined above, with the extra stipulation that the graph is divided by the cut, so that r is in R and s is not.

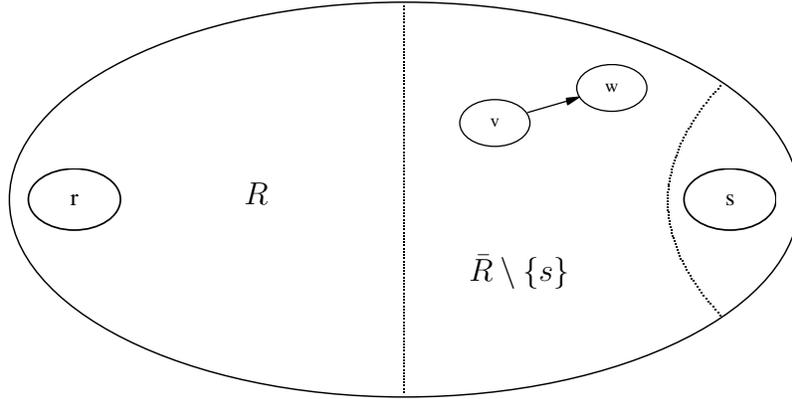
Lemma (Proposition 3.3 in the text [1, p.40]):

Given an (r - s) cut $\delta(R)$ and any (r - s) flow \vec{x} , we have $x(\delta(R)) - x(\delta(\bar{R})) = f_{\vec{x}}(s)$
 (x (a set) = the sum of \vec{x} values on edges in that set).

This lemma says that the total flow out of R minus the total flow into $R = f_{\vec{x}}(s)$. To prove the lemma, we first of all consider a graph divided into subgraphs $R, \bar{R} \setminus \{s\}$, and s :



There will likely be some edges that begin and end inside the set $\bar{R} \setminus \{s\}$ (any edge e_{vw} such that $v \in \bar{R} \setminus \{s\}, w \in \bar{R} \setminus \{s\}$):



Since vertices inside the graph (all vertices except r and s) cannot store any flow, the net flow of all these edges inside $\bar{R} \setminus \{s\}$ must be zero:

$$\sum f_{\bar{x}}(v) = 0, \quad \forall v \in \bar{R} \setminus \{s\}$$

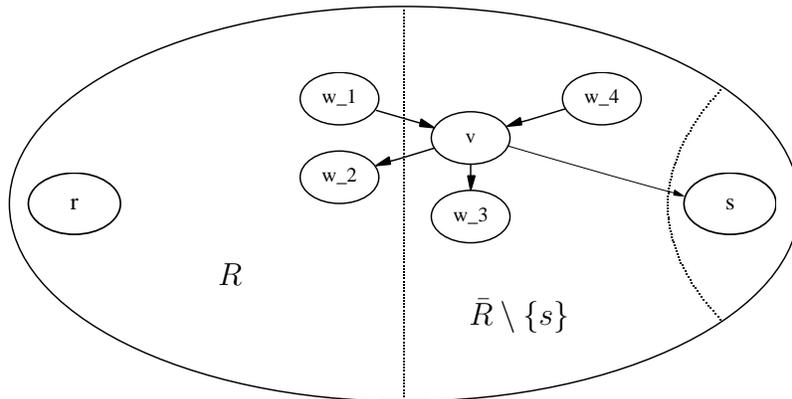
Since $f_{\bar{x}}(s) = f_{\bar{x}}(s)$, we can add the equalities to get the following relationship:

$$\begin{array}{rcl} \sum f_{\bar{x}}(v) & = & 0, \quad \forall v \in \bar{R} \setminus \{s\} \\ f_{\bar{x}}(s) & = & f_{\bar{x}}(s) \\ \hline \sum f_{\bar{x}}(v) + f_{\bar{x}}(s) & = & f_{\bar{x}}(s), \quad \forall v \in \bar{R} \setminus \{s\} \end{array}$$

We also claim the following equality to be true for all $v \in \bar{R} \setminus \{s\}$:

$$f_{\bar{x}}(v) = \sum_{w \in R} x_{wv} - \left(\sum_{w \in R} x_{vw} + \sum_{w \in \bar{R} \setminus \{s\}} x_{vw} + x_{vs} \right) + \sum_{w \in \bar{R} \setminus \{s\}} x_{wv}$$

The picture below illustrates the terms of the equation above: $f_{\bar{x}}(v)$ is the net flow into v ; $\sum_{w_1 \in R} x_{w_1 v}$ is the sum of everything flowing into v from the set R (the set of $x_{w_1 v}$'s is $\delta(R)$); $\sum_{w_2 \in R} x_{vw_2}$ is the sum of everything flowing out of v into R (the set of x_{vw_2} 's is $\delta(\bar{R})$); $\sum_{w_3 \in \bar{R} \setminus \{s\}} x_{vw_3}$ is the sum of everything flowing from v to vertices in $\bar{R} \setminus \{s\}$ (the same set v is in); x_{vs} is the flow from v to s ; and $\sum_{w_4 \in \bar{R} \setminus \{s\}} x_{w_4 v}$ is the sum of everything flowing into v from $\bar{R} \setminus \{s\}$ (again, the same set v is in) into v . So the right hand side of this equation covers all the possible edges to and from v , adding flows coming into v and subtracting flows going out. And the left side (again) is therefore the net flow into v .



The equation and illustration above apply to each vertex v from the set $\bar{R} \setminus \{s\}$. So we can sum up all of those equations (one for each vertex $v \in \bar{R} \setminus \{s\}$) and get $\sum_{v \in \bar{R} \setminus \{s\}} f_{\vec{x}}(v)$, which is the first term from the previous equation ($\sum f_{\vec{x}}(v) + f_{\vec{x}}(s) = f_{\vec{x}}(s)$). Summing all these vertices, we have:

$$\sum_{v \in \bar{R} \setminus \{s\}} (f_{\vec{x}}(v)) = \sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in R} x_{wv} - (\sum_{w \in R} x_{vw} + \sum_{w \in \bar{R} \setminus \{s\}} x_{vw} + x_{vs}) + \sum_{w \in \bar{R} \setminus \{s\}} x_{wv})$$

We stated earlier that for all v in $\bar{R} \setminus \{s\}$, $\sum f_{\vec{x}}(v) = 0$, since none of these vertices can store any flow. So this means that the two terms above pertaining to edges completely within $\bar{R} \setminus \{s\}$ will cancel each other out:

$$\sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in \bar{R} \setminus \{s\}} x_{wv}) - \sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in \bar{R} \setminus \{s\}} x_{vw}) = 0$$

After cancelling, we have:

$$\sum_{v \in \bar{R} \setminus \{s\}} (f_{\vec{x}}(v)) = \sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in R} x_{wv} - (\sum_{w \in R} x_{vw} + x_{vs}))$$

$f_{\vec{x}}(s)$ can be broken up in the following way:

$$f_{\vec{x}}(s) = \sum_{v \in R} x_{vs} + \sum_{v \in \bar{R} \setminus \{s\}} x_{vs}$$

We now have equalities for $\sum f_{\vec{x}}(v)$ and $f_{\vec{x}}(s)$, which we can substitute into $\sum f_{\vec{x}}(v) + f_{\vec{x}}(s) = f_{\vec{x}}(s)$:

$$\sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in R} x_{wv} - (\sum_{w \in R} x_{vw} + x_{vs})) + \sum_{v \in R} x_{vs} + \sum_{v \in \bar{R} \setminus \{s\}} x_{vs} = f_{\vec{x}}(s)$$

The x_{vs} terms cancel each other out, so that we have:

$$\sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in R} x_{wv} - (\sum_{w \in R} x_{vw})) + \sum_{v \in R} x_{vs} = f_{\vec{x}}(s)$$

We can simply rename v in the third term above w :

$$\sum_{v \in \bar{R} \setminus \{s\}} (\sum_{w \in R} x_{wv} - (\sum_{w \in R} x_{vw})) + \sum_{w \in R} x_{ws} = f_{\vec{x}}(s)$$

And simplify:

$$\sum_{v \in \bar{R} \setminus \{s\}, w \in R} (x_{wv} - x_{vw} + x_{ws}) = f_{\vec{x}}(s)$$

These are the edges referred to in the lemma! The x_{wv} 's and x_{ws} 's together constitute the flow through the cut $\delta(R)$, and the x_{vw} 's the flow through the cut $\delta(\bar{R})$. So we have proved the lemma.

Corollary (3.4 in the text [1, p.40]):

For any feasible flow \vec{x} and any $(r-s)$ cut, $f_{\vec{x}}(s) \leq u(\delta(R))$.

The corollary states that the net flow into s cannot exceed the sum of the capacities of the edges in the $(r-s)$ cut. This makes sense, since we we have just shown that the net flow into s is at most equal to the flow across the cut (it would be equal if there is no flow across the cut in the opposite direction—if $\delta\bar{R} = 0$), which obviously can not exceed the sum of the capacities of edges in the cut.

Max-Flow Min-Cut Theorem

If $G = \langle V, E, \vec{u} \rangle$ is a flow graph,

$$\max\{f_{\vec{x}}(s) : \vec{x} \text{ is a feasible flow}\} = \min\{u(\delta(R)) : R \text{ is an } (r-s) \text{ cut}\}$$

We note that in terms of linear programming, *Max-Flow* and *Min-Cut* are duals of each other. Before proving the theorem we need two definitions:

\vec{x} -Incrementing Path:

Given a flow \vec{x} , a path p in a graph G is called “ \vec{x} -incrementing” if

1. every forward arc e has flow $x_e < \text{capacity } u_e$;
2. every reverse arc e has flow $x_e > 0$.

\vec{x} -Augmenting Path:

An $(r-s)$ \vec{x} -incrementing path is called \vec{x} -augmenting path.

\vec{x} -augmenting paths are those paths from r to s through which the flow may be increased, through forward arcs whose flow is less than their capacity and through backward arcs whose flow is greater than zero (and may therefore be decreased).

To prove the *Max-Flow Min-Cut* theorem, we let \vec{x} be a *Max-Flow*. This means there is no \vec{x} -augmenting path. If there were an \vec{x} -augmenting path, the flow from r to s through that path could be increased, implying that the former flow \vec{x} was not in fact the *Max-Flow*. There may be \vec{x} -incrementing paths, however. We let $R = \{v : \text{there is an } \vec{x}\text{-incrementing path from } r \text{ to } v\}$. We note that s may not be part of R , but r must be part of R . It follows that $\{e_{vw} : v \in R, w \notin R\}$ is an $(r-s)$ cut. And we know that the flow x_{vw} (through edge $e_{vw} \in \delta(R)$) must be equal to the capacity u_{vw} , otherwise e_{vw} would make up part of an \vec{x} -incrementing path, and we would put w in R . By the same reasoning all edges in $\delta(\bar{R})$ (those edges going from $v \in \bar{R}$ to $w \in R$) must have flow $x_{vw} = 0$. So we know that, for all edges in the cut $\delta(R)$, $x_{vw} = u_{vw}$; and for all edges in the cut $\delta(\bar{R})$, $x_{vw} = 0$. From the previous theorem, we know:

$$\vec{x}(\delta(R)) - \vec{x}(\delta(\bar{R})) = f_{\vec{x}}(s)$$

If we substitute $u(\delta(R))$ for $\vec{x}(\delta(R))$ and 0 for $\vec{x}(\delta(\bar{R}))$, we have proved the theorem:

$$u(\delta(R)) = f_{\vec{x}}(s)$$

References

- [1] William Cook, William H. Cunningham, William Pulleyblank, and Alexander Schrijver. *Combinatorial Optimization*. John Wiley & Sons, 1998.