V12 from graph connectivity to network flow

Program for today

Menger‘s theorem

Insert: annotate and compare functional annotations of genes

Flow in networks

strict paths

Definition Let *W* be a set of vertices in a graph *G* and *x* another vertex not in *W.*
A **strict *x-W* path** is a path joining *x* to a vertex in *W* and
containing no other vertex of *W.*
A **strict *W-x* path** is the reverse of a strict *x-W* path (i.e. its sequence of vertices and edges is in reverse order).

Example: Let us consider the ***u-v* separating set** $W = \{y,s,z\}$ in the graph below.

There are four **strict *u-W* paths** $\langle u,x,y\rangle$, $\langle u,r,y\rangle$, $\langle u,r,s\rangle$, $\langle u,z\rangle$
And three **strict *W-v* paths** $\langle z,v\rangle$, $\langle y,t,v\rangle$, and $\langle s,v\rangle$.

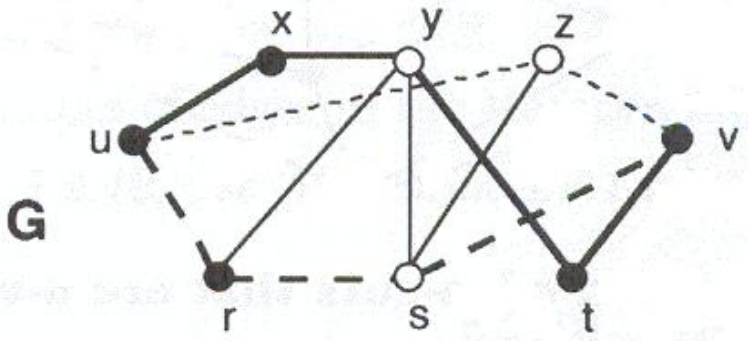


Figure 5.3.2

Menger's Theorem

Theorem 5.3.4 [Menger, 1927] Let *u* and *v* be distinct, non-adjacent vertices in a connected graph *G.*
Then the maximum number of internally disjoint *u*-*v* paths in *G*
equals the minimum number of vertices needed to separate *u* and *v.*

Proof: The proof uses induction on the number of edges.

The smallest graph that satisfies the premises of the theorem (non-adjacent *u* and *v*) is the path graph from *u* to *v* of length 2.

The theorem is trivially true for this graph : one cut-vertex, one *u*-*v* path.

Menger's Theorem

Assume now that the theorem is true for all connected graphs having fewer than m edges, e.g. for some $m \geq 3$.

Suppose that G is a connected graph with m edges, and let k be the minimum number of vertices needed to separate the vertices u and v.

By Corollary 5.3.2 (number of paths ≤ number of vertices), it suffices to show that there exist k internally disjoint u-v paths in G.

This is clearly true if $k = 1$ (since G is connected, there exists a u-v path).

Thus, we will assume $k \geq 2$.

Proof of Menger's Theorem

Assertion 5.3.4a If G contains a u-v path of length 2,
then G contains k internally disjoint u-v paths.

Proof: Suppose that $\mathcal{P} = \langle u, e_1, x, e_2, v \rangle$ is a path in G of length 2.
$G - x$ has fewer edges than $G \rightarrow$ by the induction hypothesis,
there are at least $k - 1$ internally disjoint $u - v$ paths in $G - x$.

Path $\mathcal{P}$ is internally disjoint from any of these, and, hence,
there are k internally disjoint u-v paths in G. □

If there is a u-v separating set that contains a vertex adjacent to *both* vertices u and v, then Assertion 5.3.4a guarantees the existence of k internally disjoint u-v paths in G.

The argument for *distance* $(u,v) \geq 3$ is now broken into two cases,
according to the kinds of u-v separating sets that exist in G.

Proof of Menger's Theorem

In **Case 1** (left picture), there exists a *u*-*v* separating set *W*,
where neither *u* nor *v* is adjacent to every vertex of *W* .

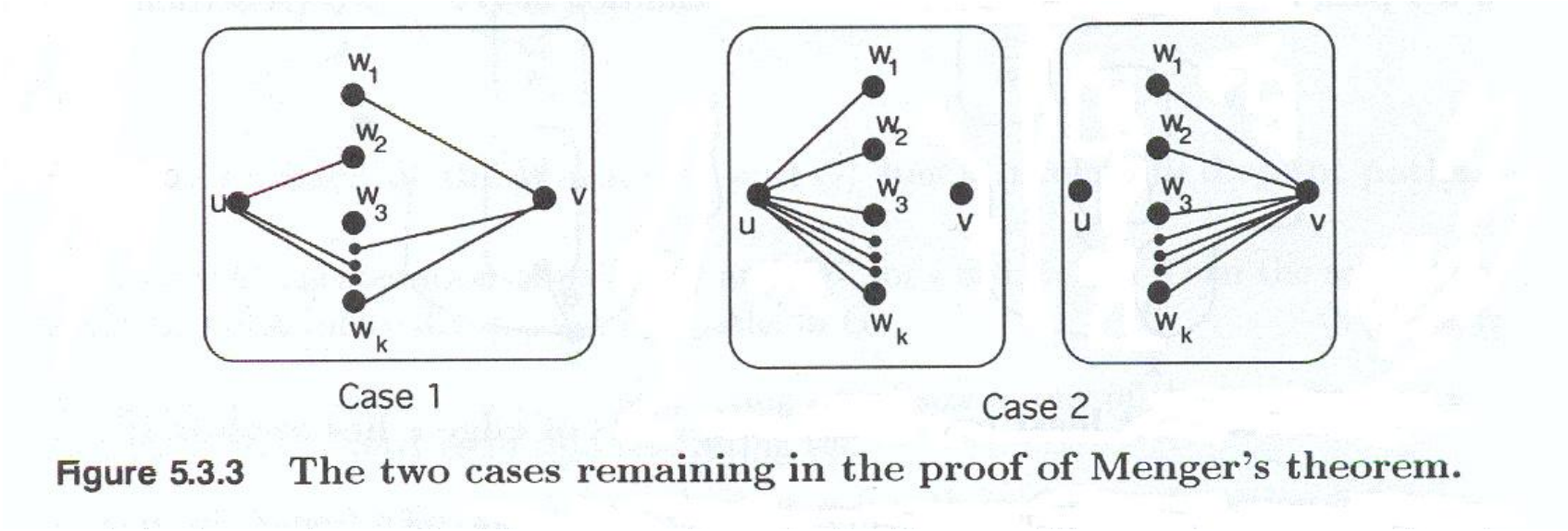


Figure 5.3.3 The two cases remaining in the proof of Menger's theorem.

In **Case 2** (right picture), no such separating set exists.
Thus, in every *u*-*v* separating set for Case 2,
either every vertex is adjacent to *u* or every vertex is adjacent to *v*.

Proof of Menger's Theorem

Case 1: There exists a *u-v* separating set $W = \{w_1, w_2, \ldots, w_k\}$ of vertices in *G* of minimum size *k,* such that neither *u* nor *v* is adjacent to every vertex in *W.*

Let G_u be the subgraph induced on the union of the edge-sets of all strict *u-W* paths in *G,*
and let G_v be the subgraph induced on the union of edge-sets of all strict *W-v* paths (see Fig. below).

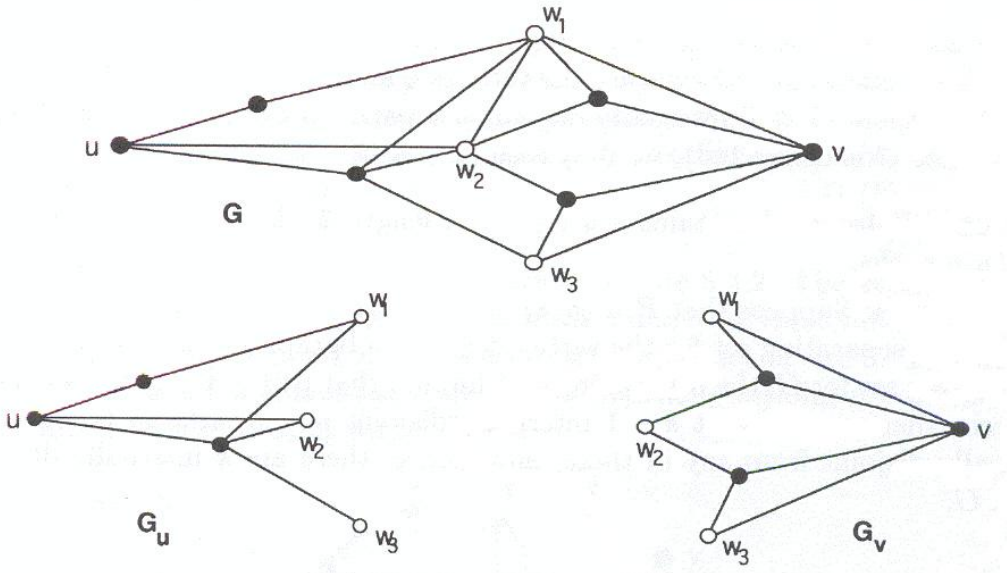


Figure 5.3.4 An example illustrating the subgraphs G_u and G_v.

Split up graph

Proof of Menger's Theorem

Assertion 5.3.4b: Both of the subgraphs G_u and G_v have more than k edges.

Proof : For each $w_i \in W$, there is a u-v path P_{wi} in G
on which w_i is the only vertex of W.
(Otherwise, $W - \{w_i\}$ would still be a u-v separating set, which would contradict the minimality of W).

The u-w_i subpath of P_{wi} is a strict u-W path that ends at w_i.
Thus, the final edge of this strict u-W path is different for each w_i.
Hence, G_u has at least k edges.

The only way G_u could have exactly k edges would be if each of these
Strict u-W paths consisted of a single edge joining u and w_i, $i = 1, \ldots, k$.
But this is ruled out by the condition for Case 1.
Therefore, G_u has more than k edges.

A similar argument shows that G_v also has more than k edges. □

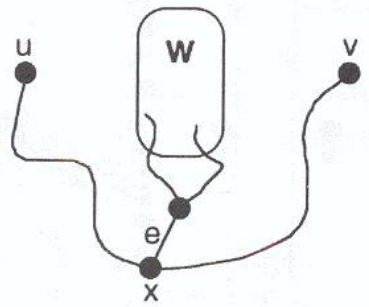
Proof of Menger's Theorem

Assertion 5.3.4c: The subgraphs G_u and G_v have no edges in common.

Proof of 5.3.4c: By way of contradiction, suppose that the subgraphs G_u and G_v have an edge e in common.

By the definitions of G_u and G_v, edge e would then be an edge of both a strict u-W path and a strict W-v path.

Hence, at least one of the endpoints of e, say x, is not a vertex in the u-v separating set W (see Fig. below).
This implies the existence of a u-v path in G-W,
which contradicts the definition of W. □



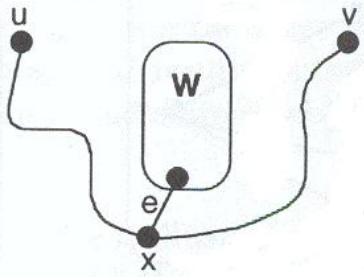


Figure 5.3.5 At least one of the endpoints of edge e lies outside W.

Proof of Menger's Theorem

We now define two auxiliary graphs G_u^* and G_v^*:

$\boldsymbol{G_u^*}$ is obtained from G by replacing the subgraph G_v with **a new vertex $\boldsymbol{v^*}$** and drawing an edge from each vertex in W to v^*, and

$\boldsymbol{G_v^*}$ is obtained by replacing G_u with **a new vertex $\boldsymbol{u^*}$** and drawing an edge from u^* to each vertex in W (see Fig. below).

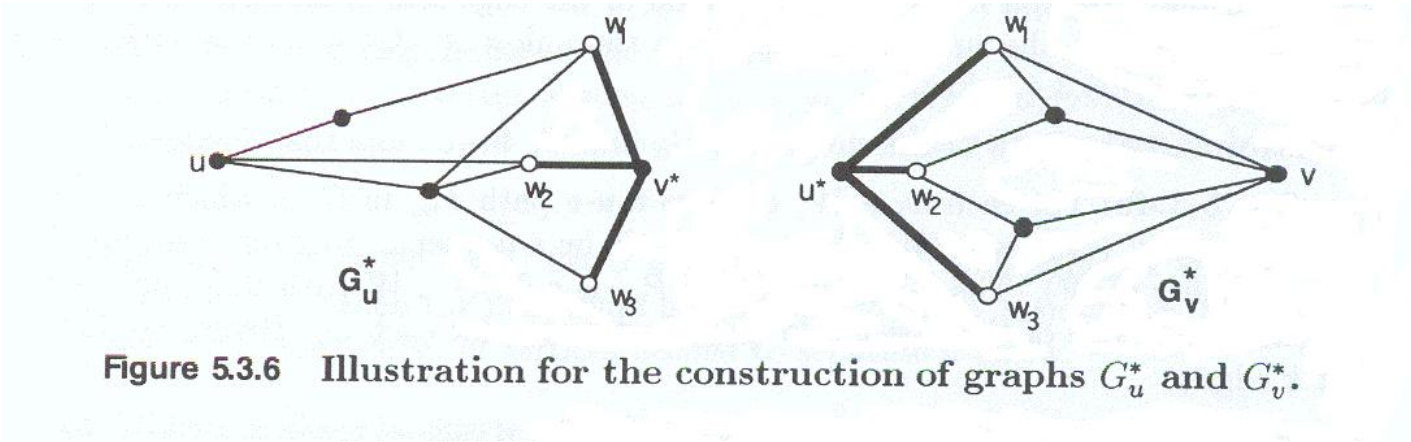


Figure 5.3.6 Illustration for the construction of graphs G_u^* and G_v^*.

Proof of Menger's Theorem

Assertion 5.3.4d: Both of the auxiliary graphs G_u^* and G_v^* have fewer edges than G.

Q: Why would this be useful?

Proof of 5.3.4d: The following chain of inequalities shows that graph G_u^* has fewer edges than G.

$$|E_G| \geq |E_{G_u \cup G_v}|$$ since $G_u \cup G_v$ is a subgraph of G

$$= |E_{G_u}| + |E_{G_v}|$$

$$\overset{5.3.4c}{>} |E_{G_u}| + k$$

$$\overset{5.3.4b}{=} |E_{G_u^*}|$$ by the construction of G_u^*

A similar argument shows that G_v^* also has fewer edges than G. □

By the construction of graphs G_u^* and G_v^*, every u-v^* separating set in graph G_u^* and every u^*-v separating set in graph G_v^* is a u-v separating set in graph G.

Hence, the set W is a **smallest u-v^* separating set** in G_u^* and a smallest u^*-v separating set in G_v^*.

Proof of Menger's Theorem

Since G_u^* and G_v^* have **fewer edges** than G, the **induction hypothesis** implies the existence of two collections, $\mathcal{P}_u^*$ and $\mathcal{P}_v^*$ of k internally disjoint u-v^* paths in G_u^* and k internally disjoint u^*-v paths in G_v^*, respectively (see Fig.).

For each w_i, one of the paths in $\mathcal{P}_u^*$ consists of a u-w_i path P_i' in G plus the new edge from w_i to v^*, and one of the paths in $\mathcal{P}_v^*$ consists of the new edge from u^* to w_i followed by a w_i-v path P_i'' in G.

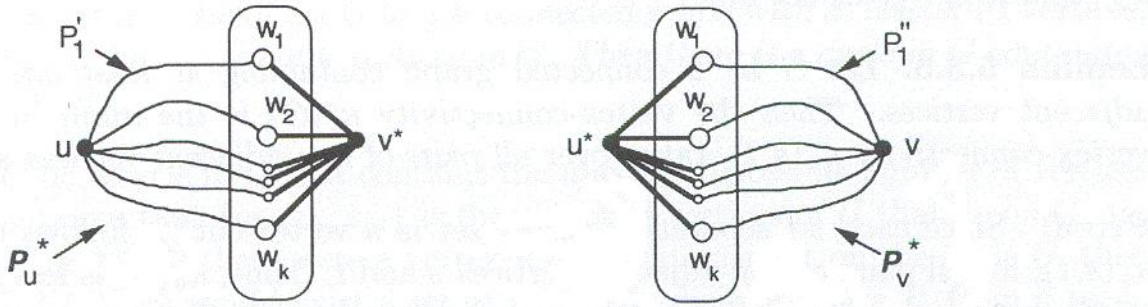


Figure 5.3.7 Each of the graphs G_u^* and G_v^* has k internally disjoint paths.

Let P_i be the **concatenation** of paths P_i' and P_i'', for $i = 1, ..., k$.
Then the set $\{P_i\}$ is a collection of k internally disjoint u-v paths in G. □ (Case 1)

Proof of Menger's Theorem

Case 2: Suppose that for each *u*-*v* separating set of size *k,* one of the vertices *u* or *v* is adjacent to all the vertices in that separating set.

will not be proven in lecture

Let $P = \langle u, e_1, x_1, e_2, x_2, \ldots, v \rangle$ be a shortest *u*-*v* path in *G.*

By Assertion 5.3.4a, we can assume that *P* has length at least 3 and that vertex x_1 is not adjacent to vertex *v.*

By Proposition 5.1.3, the edge-deletion subgraph $G - e_2$ is connected.

Let *S* be a smallest *u*-*v* separating set in subgraph $G - e_2$ *(*see Fig.).

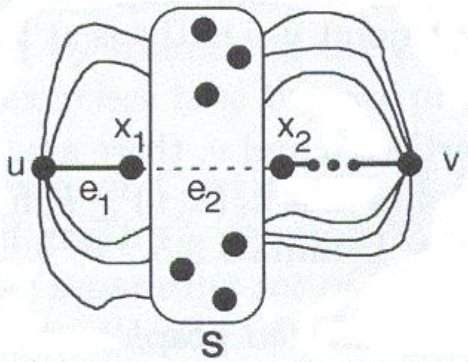


Figure 5.3.8 Completing Case 2 of Menger's theorem.

Proof of Menger's Theorem

Then S is a u-v separating set in the vertex-deletion subgraph $G - x_1$.
Thus, $S \cup \{x_1\}$ is a u-v separating set in G, which implies that $|S| \geq k - 1$, by the minimality of k. On the other hand, the minimality of
$|S|$ in $G - e_2$ implies that $|S| \leq k$, since every u-v separating set in G is also a u-v separating set in $G - e_2$.

If $|S| = k$, then, by the induction hypothesis, there are k internally disjoint u-v paths in $G - e_2$ and, hence, in G.
If $|S| = k - 1$, then $x_i \notin S$, $i = 1,2$ (otherwise $S - \{x_i\}$ would be a u-v separating set in $G - e_2$, contradicting the minimality of k).
Thus, the sets $S \cup \{x_1\}$ and $S \cup \{x_2\}$ are both of size k and both u-v separating sets of G. The condition for Case 2 and the fact that vertex x_1 is not adjacent to v imply that every vertex in S is adjacent to vertex u.
Hence, no vertex in S is adjacent to v (lest there be a u-v path of length 2).
But then the condition of Case applied to $S \cup \{x_2\}$ implies that vertex x_2 is adjacent to vertex u, which contradicts the minimality of path P and completes the proof. □

Insert: functional annotation of gene function

- Functional annotation of genes/gene products: Gene Ontology (GO)
- significance of annotation: hypergeometric test
- (mathematical) similarity of GO-terms

See lecture V3 page 20. There we stated

3) co-functionality

it is realistic to assume that members of a protein complex should have **closely related biological functions** -> check whether interaction proteins have overlapping annotations with terms from Genome Ontology (GO)

Equivalently, we can expect that members of a protein complex should have **higher functional similarity** to eachother than random proteins.

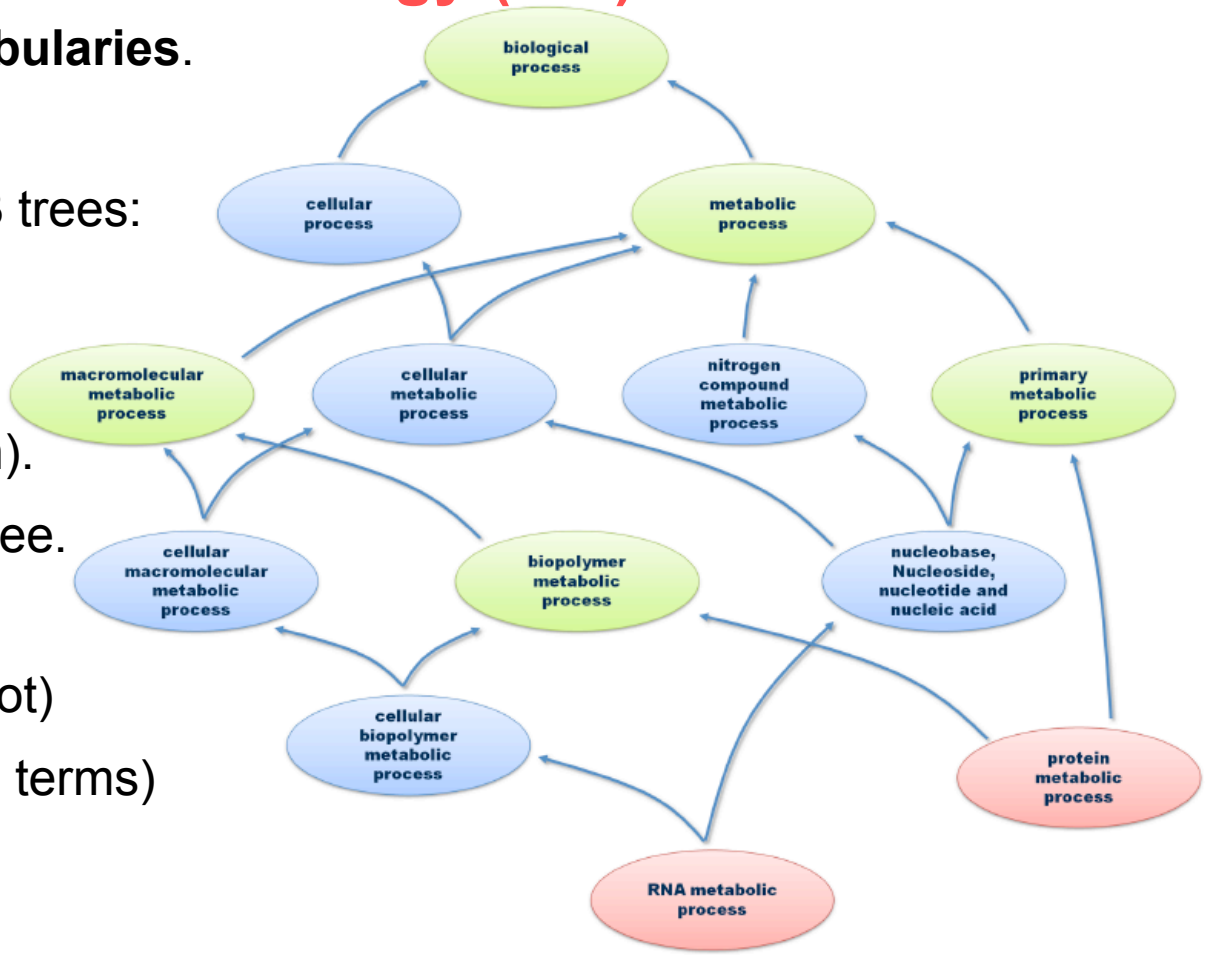
The Gene Ontology (GO)

Ontologies are **structured vocabularies**.

The Gene Ontology consists of 3 trees:

- Biological process (BP)
- molecular function (MF)
- cellular component (localisation).

Shown here is a part of the BP tree.



At the top: most general term (root)

Red: tree leafs (very specific GO terms)

Green: common ancestor

Blue: other nodes.

Arcs: „Y is contained in X“-relations

PhD Dissertation Andreas Schlicker (UdS, 2010)

Where do the Gene Ontology annotations come from?

Evidence code	Evidence code description	Source of evidence	Manually checked	Current number of annotations*
IDA	Inferred from direct assay	Experimental	Yes	71,050
IEP	Inferred from expression pattern	Experimental	Yes	4,598
IGI	Inferred from genetic interaction	Experimental	Yes	8,311
IMP	Inferred from mutant phenotype	Experimental	Yes	61,549
IPI	Inferred from physical interaction	Experimental	Yes	17,043
ISS	Inferred from sequence or structural similarity	Computational	Yes	196,643
RCA	Inferred from reviewed computational analysis	Computational	Yes	103,792
IGC	Inferred from genomic context	Computational	Yes	4
IEA	Inferred from electronic annotation	Computational	No	15,687,382
IC	Inferred by curator	Indirectly derived from experimental or computational evidence made by a curator	Yes	5,167
TAS	Traceable author statement	Indirectly derived from experimental or computational evidence made by the author of the published article	Yes	44,564
NAS	Non-traceable author statement	No 'source of evidence' statement given	Yes	25,656
ND	No biological data available	No information available	Yes	132,192
NR	Not recorded	Unknown	Yes	1,185

*October 2007 release

Rhee et al. Nature Reviews Genetics 9, 509-515 (2008)

Signifkance of GO annotations

Very **general GO terms** such as “cellular metabolic process“
are annotated to many genes in the genome.

Very **specific terms** belong to a few genes only.

→ One needs to compare how **significant** the occurrence of a
GO term is in a given set of genes
compared to a randomly selected set of genes of the same size.

This is often done with the **hypergeometric test**.

PhD Dissertation Andreas Schlicker (UdS, 2010)

Hypergeometric test

$$\text{p-value} = \sum_{i=k_\pi}^{min(n,K_\pi)} \frac{\binom{K_\pi}{i}\binom{N-K_\pi}{n-i}}{\binom{N}{n}}$$

The hypergeometric test is a statistical test.

It can be used to check e.g. whether a biological annotation π is **statistically significant enriched** in a given test set of genes compared to the full genome.

- N : number of genes in the genome
- n : number of genes in the test set
- K_π : number of genes in the genome with annotation π.
- k_π : number of genes in test set with annotation π.

The hypergeometric test provides the **likelihood** that k_π or more genes that were **randomly selected** from the genome also have annotation π.

Hypergeometric test

Select $i \geq k_\pi$ genes with annotation π from the genome.

There are K_π such genes.

The other $n - i$ genes in the test set do NOT have annotation π.

There are $N - K_\pi$ such genes in the genome.

$$\text{p-value} = \sum_{i=k_\pi}^{min(n,K_\pi)} \frac{\binom{K_\pi}{i}\binom{N-K_\pi}{n-i}}{\binom{N}{n}}$$

The sum runs from k_π elements to the maximal possible number of elements.

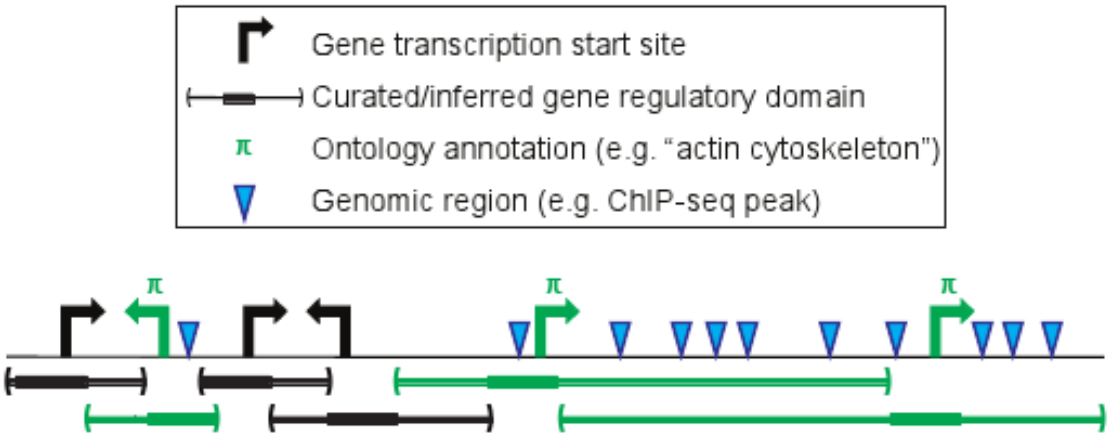
This is either the number of genes with annotation π in the genome (K_π) or the number of genes in the test set *(n).*

corrects for the number of possibilities for selecting *n* elements from a set of *N* elements.

This correction is applied if the sequence of drawing the elements is not important.

Example

p-Wert = $$\sum_{i=k_\pi}^{min(n,K_\pi)} \frac{\binom{K_\pi}{i}\binom{N-K_\pi}{n-i}}{\binom{N}{n}}$$



Is annotation π significantly enriched in the test set of 3 genes?

Hypergeometric test over genes
N = 6 total genes
K_π = 3 genes annotated with π
n = 3 genes with an associated genomic region
k_π = 3 genes annotated and with a genomic region
P-value = 0.05

Yes! p = 0.05 is (just) significant.

Information content of GO terms

The **likelihood** of a node *t* can be defined in 2 ways:

How many genes have annotation *t* relative to the root node?

$$p_{anno}(t) = \frac{occur(t)}{occur(root)}$$

Number of GO terms in subtree below *t* relative to number of GO terms in tree

$$p_{graph}(t) = \frac{D(t)}{D(root)}$$

The likelihood takes values between 0 and 1 and increases monotonic from the leaf nodes to the root.

Define **information content** of a node from its likelihood:

$$IC(t) = -\log p(t)$$

A rare node has high information content.

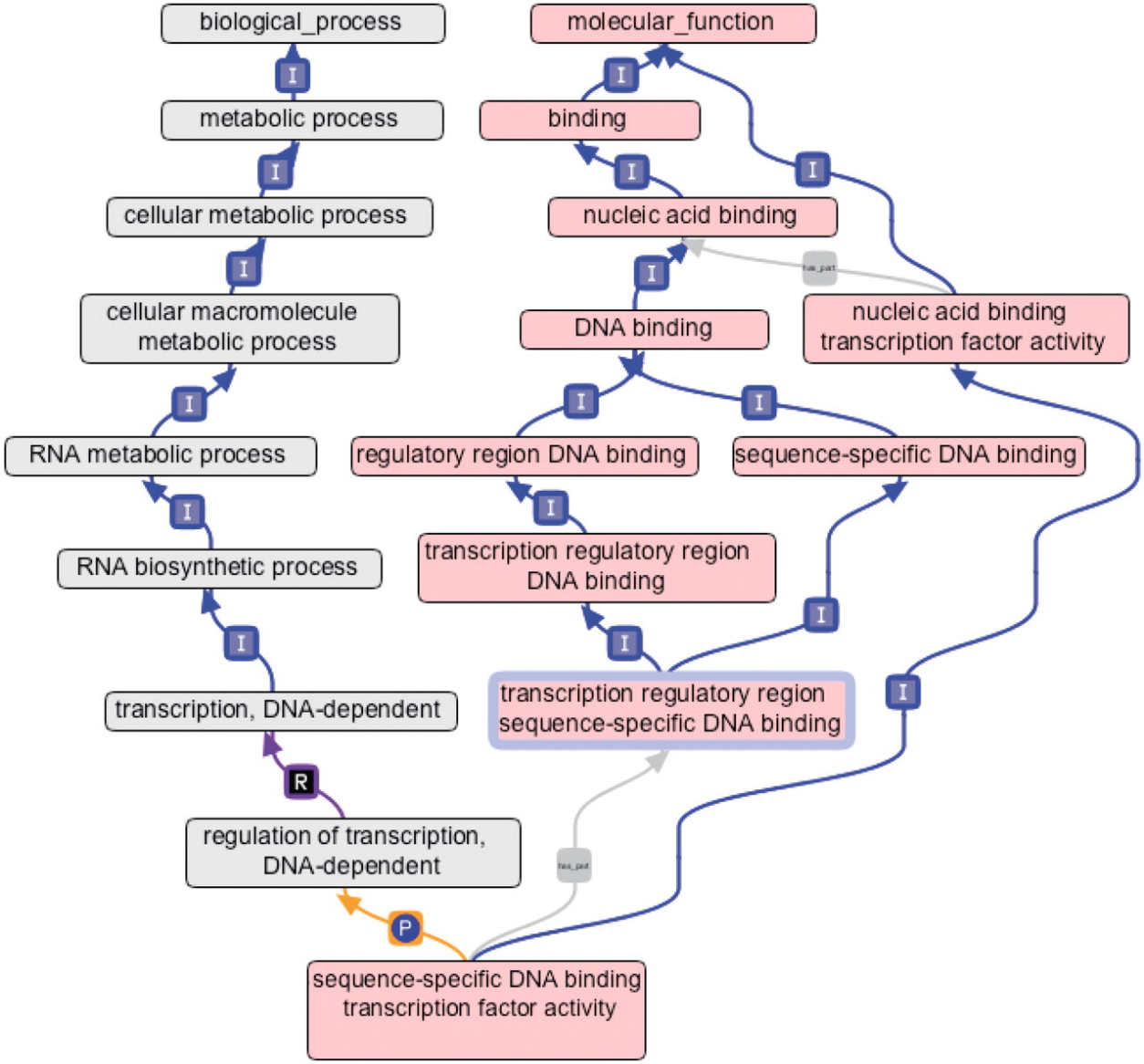
PhD Dissertation Andreas Schlicker (UdS, 2010)

Common ancestors of GO terms

Common ancestors of two nodes t_1 and t_2 : all nodes that are located on a path from t_1 to root AND on a path from t_2 to root.

The **most informative common ancestor** (MICA) of terms t_1 und t_2 is their common ancestor with highest information content.

Typically, this is the closest common ancestor.



Nucl. Acids Res. (2012) 40 (D1): D559-D564

Measure functional similarity of GO terms

Schlicker *et al.* defined the **similarity** of two GO terms t_1 und t_2
based on the information content of the most informative common ancestor (MICA)

$$sim_{Rel}(t_1,t_2) = \frac{2 \cdot IC(MICA)}{IC(t_1)+IC(t_2)}$$

The following variant worked slightly better in practice:

$$sim_{Rel}(t_1,t_2) = \frac{2 \cdot IC(MICA)}{IC(t_1)+IC(t_2)} \cdot (1 - p(MICA))$$

PhD Dissertation Andreas Schlicker (UdS, 2010)

Measure functional similarity of two genes

Two genes or two sets of genes *A* und *B* typically have more than 1 GO annotation each. → Consider similarity of all terms *i* and *j*:

$$s_{ij} = sim(GO_i^A, GO_j^B), \forall i \in 1,...,N, \forall j \in 1,...,M.$$

and select the maxima in all rows and columns:

$$rowScore(A,B) = \frac{1}{N}\sum_{i=1}^{N} \max_{1 \le j \le M} s_{ij}, \qquad GOscore_{avg}^{BMA}(A,B) = \frac{1}{2} \cdot (rowScore(A,B) + columnScore(A,B))$$

$$columnScore(A,B) = \frac{1}{M}\sum_{j=1}^{M} \max_{1 \le i \le N} s_{ij}. \qquad GOscore_{max}^{BMA}(A,B) = max(rowScore(A,B), columnScore(A,B))$$

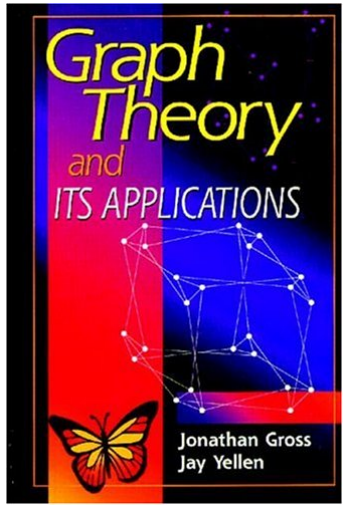
Compute *funsim*-Score from scores for BP tree and MF tree:

$$funsim(A,B) = \frac{1}{2} \cdot \left[\left(\frac{BPscore}{\max(BPscore)} \right)^2 + \left(\frac{MFscore}{\max(MFscore)} \right)^2 \right]$$

PhD Dissertation Andreas Schlicker (UdS, 2010)

Flows and Cuts in Networks

The second part of this lecture follows closely chapter 12.1 in the book on the right on „Flows and Cuts in Networks and Chapter 12.2 on “Solving the Maximum-Flow Problem“



Flow in Networks can mean
- flow of oil or water in pipelines, electricity
- phone calls, emails, traffic networks ...

Equivalences exist between
max-flow min-cut theorem of Ford and Fulkerson
& the **connectivity theorems** of Menger

→ this led to the development of efficient algorithms for a number of practical problems to solve scheduling and assignment problems.

Single Source – Single Sink Capacitated Networks

Definition: A **single source – single sink network** is a connected digraph that has a distinguished vertex called the **source** with nonzero outdegree and a distinguished vertex called the **sink** with nonzero indegree.

Such a network with source *s* and sink *t* is often referred to as a ***s-t* network**.

Definition: A **capacitated network** is a connected digraph such that each arc *e* is assigned a nonnegative weight ***cap(e),*** called the **capacity** of arc *e.*

Notation: Let *v* be a vertex in a digraph *N*. Then ***Out(v)*** denotes the set of all arcs that are directed **away from** vertex *v*. That is,

$$Out(v) = \{e \in E_N \mid tail(e) = v\}$$

Correspondingly, ***In(v)*** denotes the set of arcs that are directed **to** vertex *v:*

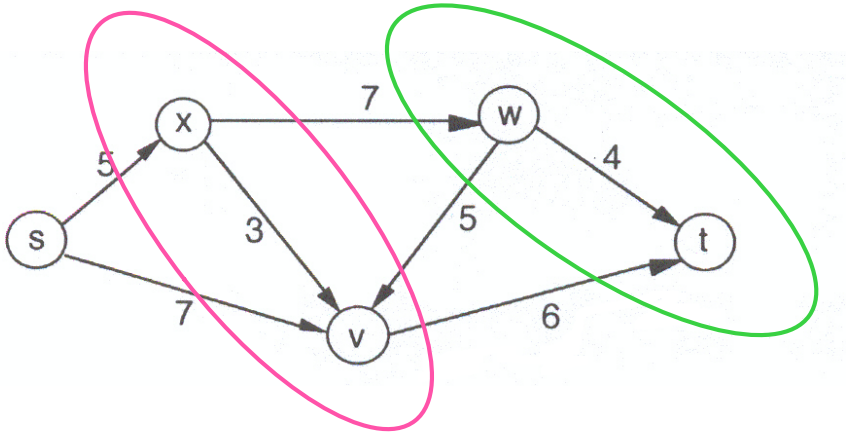
$$In(v) = \{e \in E_N \mid head(e) = v\}$$

Single Source – Single Sink Capacitated Networks

Notation: For any two vertex subsets *X* and *Y* of a digraph *N*, let $\langle X,Y \rangle$ denote the set of arcs in *N* that are directed **from** a vertex in *X* **to** a vertex in *Y*.

$$\langle X,Y \rangle = \{ e \in E_N \mid tail(e) \in X \text{ and } head(e) \in Y \}$$

Example: The figure shows a 5-vertex capacitated *s*-*t*-network.
If ***X*** = *{x,v}* and ***Y*** = *{w,t}*, then the elements of arc set $\langle X,Y \rangle$ are the arc directed from vertex *x* to vertex *w* and the arc directed from vertex *v* to sink *t*.



A 5-vertex capacitated network with source *s* and sink *t*.

The only element in arc set $\langle Y,X \rangle$ is the arc directed from vertex *w* to vertex *v*.

Feasible Flows

Definition: Let *N* be a capacitated *s*-*t*-network.

A **feasible flow** *f* in *N* is a function $f:E_N \rightarrow R^+$ that assigns a nonnegative real number to every vertex *v* in network *N*, other than source *s* and sink *t,* and that fulfills the following two conditions

1. (**capacity constraints**) $f(e) \leq cap(e)$, for every arc *e* in network *N.*
2. (**conservation constraints**) $\sum_{e \in In(v)} f(e) = \sum_{e \in Out(v)} f(e)$

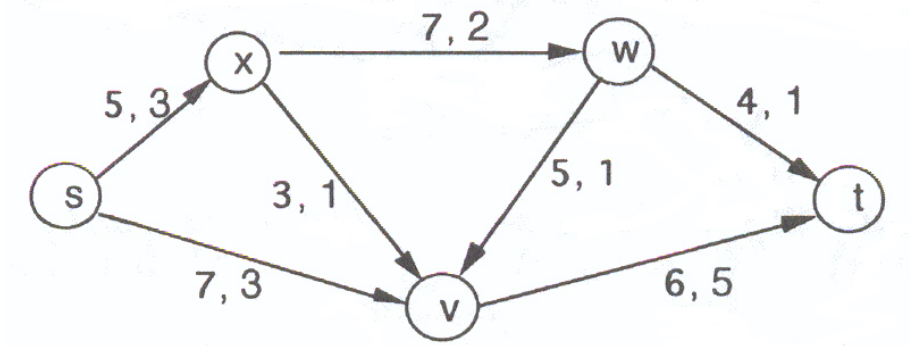
Property 2 above is called the **conservation-of-flow** condition.

E.g. for an oil pipeline, the total flow of oil going into any juncture (vertex) in the pipeline must equal the total flow leaving that juncture.

Notation: to distinguish visually between the flow and the capacity of an arc, we adopt the **convention** in drawings that when both numbers appear, the **capacity** will always be in **bold** and to the left of the flow.

Feasible Flows

Example: The figure shows a feasible flow for the previous network.
Notice that the total amount of flow leaving source *s* equals 6, which is also the net flow entering sink *t.*



Definition: The **value of flow** *f* in a capacitated network *N*, denoted with ***val(f)***, is the net flow leaving the source *s*, that is

$$val(f) = \sum_{e \in Out(s)} f(e) - \sum_{e \in In(s)} f(e)$$

Definition: The **maximum flow *f**** in a capacitated network *N* is a flow in *N* having the maximum value, i.e. $val(f) \leq val(f^*)$, for every flow *f* in *N.*

Cuts in *s-t* Networks

By definition, any nonzero flow must use at least one of the arcs in *Out(s).*
In other words, if all of the arcs in *Out(s)* were deleted from network *N*,
then no flow could get from source *s* to sink *t.*

This is a special case of the following definition, which combines the concepts of **partition-cut** and **s-t separating set.**

From V11
Definition: Let *G* be a graph, and let X_1 and X_2 form a **partition** of V_G.
The set of all edges of *G* having one endpoint in X_1 and the other endpoint in X_2 is called a **partition-cut** of G and is denoted $\langle X_1, X_2 \rangle$.

From V11
Definition: Let *u* and *v* be distinct vertices in a connected graph G.
A vertex subset (or edge subset) *S* is *u-v* **separating** (or **separates** *u* and *v*), if the vertices *u* and *v* lie in different components of the deletion subgraph *G – S.*

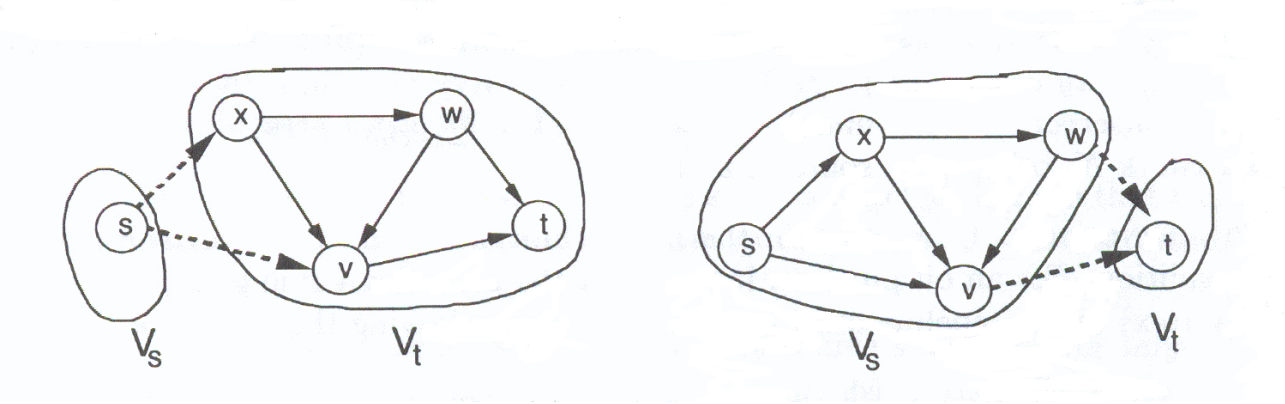
Cuts in *s-t* Networks

Definition: Let N be an *s-t* network, and let V_s and V_t form a **partition** of V_G such that source $s \in V_s$ and sink $t \in V_t$.
Then the set of all arcs that are directed from a vertex in set V_s to a vertex in set V_t is called an ***s-t* cut** of network N and is denoted $\langle V_s, V_t \rangle$.

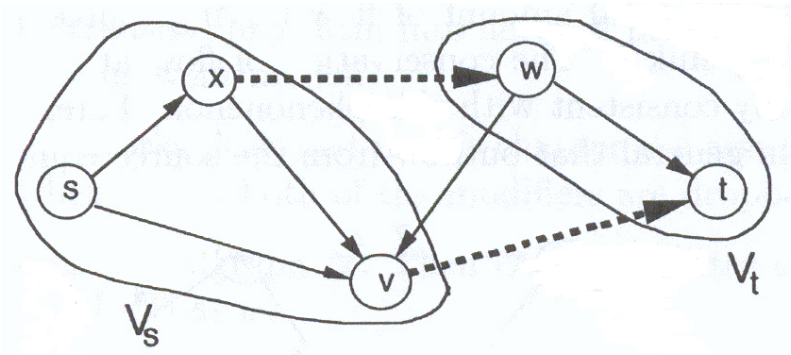
Remark: The arc set ***Out(s)*** for an *s-t* network N is the *s-t* cut $\langle \{s\}, V_N - \{s\} \rangle$, and ***In(t)*** is the *s-t* cut $\langle V_N - \{t\}, \{t\} \rangle$.

Cuts in *s-t* Networks

Example. The figure portrays the arc sets *Out(s)* and *In(t)* as *s-t* cuts, where *Out(s)* = ⟨ *{s}, {x,v,w,t}* ⟩ and *In(t)* = ⟨ {*s,x,v,w*},{*t*} ⟩.



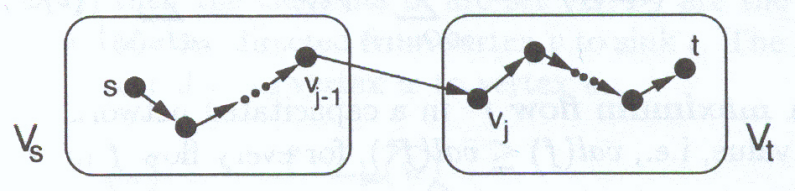
Example: a more general *s-t* cut ⟨ V_s,V_t ⟩ is shown below, where V_s = {*s,x,v*} and V_t = {*w,t*}.



Cuts in *s-t* Networks

Proposition 12.1.1 Let $\langle V_s, V_t \rangle$ be an *s-t* cut of a network *N*.
Then every directed *s-t* path in *N* contains at least one arc in $\langle V_s, V_t \rangle$.

Proof. Let $P = \langle s = v_0, v_1, v_2, \ldots, v_l = t \rangle$ be the vertex sequence of a directed *s-t* path in network *N*.



Since $s \in V_s$ and $t \in V_t$, there must be a first vertex v_j on this path that is in set V_t (see figure below).

Then the arc from vertex v_{j-1} to v_j is in $\langle V_s, V_t \rangle$. □

Relationship between Flows and Cuts

Similar to viewing the set *Out(s)* of arcs directed from source *s* as the *s*-*t* cut $\langle \{s\}, V_N - \{s\} \rangle$, the set *In(s)* may be regarded as the set of „backward“ arcs relative to this cut, namely, the arc set $\langle V_N - \{s\}, \{s\}, \rangle$.

From this perspective, the definition of *val(f)* may be rewritten as

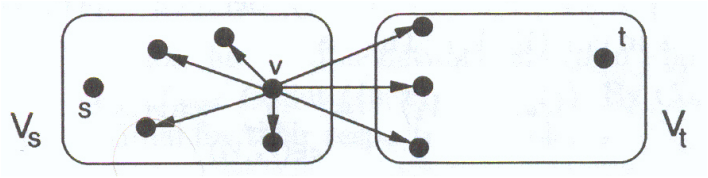
$$val(f) = \sum_{e \in \langle \{s\}, V_N - \{s\} \rangle} f(e) - \sum_{e \in \langle V_N - \{s\}, \{s\} \rangle} f(e)$$

Relationship between Flows and Cuts

Lemma 12.1.2. Let $\langle V_s, V_t \rangle$ be any *s-t* cut of an *s-t* network *N*. Then

$$\bigcup_{v \in V_s} Out(v) = \langle V_s, V_s \rangle \cup \langle V_s, V_t \rangle \quad \text{and} \quad \bigcup_{v \in V_s} In(v) = \langle V_s, V_s \rangle \cup \langle V_t, V_s \rangle$$

Proof: For any vertex $v \in V_s$, each arc directed from v is either in $\langle V_s, V_s \rangle$ or in $\langle V_s, V_t \rangle$. The figure illustrates for a vertex v the partition of $Out(v)$ into a 4-element subset of $\langle V_s, V_s \rangle$ and a 3-element subset of $\langle V_s, V_t \rangle$.



$$\bigcup_{v \in V_s} Out(v) = \langle V_s, V_s \rangle \cup \langle V_s, V_t \rangle$$

Similarly, each arc directed to vertex v is either in $\langle V_s, V_s \rangle$ or in $\langle V_t, V_s \rangle$. □

Relationship between Flows and Cuts

Proposition 12.1.3. Let *f* be a flow in an *s-t* network *N,* and let $\langle V_s, V_t \rangle$ be any *s-t* cut of *N.* Then $val(f) = \sum_{e \in \langle V_s, V_t \rangle} f(e) - \sum_{e \in \langle V_t, V_s \rangle} f(e)$

Proof: By definition,

$$val(f) = \sum_{e \in Out(s)} f(e) - \sum_{e \in In(s)} f(e)$$

And by the conservation of flow

$\sum_{e \in Out(v)} f(e) - \sum_{e \in In(v)} f(e) = 0$ for every $v \in V_s$ other than s. Thus one can expand

$$val(f) = \sum_{v \in V_s} \left(\sum_{e \in Out(v)} f(e) - \sum_{e \in In(v)} f(e) \right) = \sum_{v \in V_s} \sum_{e \in Out(v)} f(e) - \sum_{v \in V_s} \sum_{e \in In(v)} f(e) \quad (1)$$

By Lemma 12.1.2.

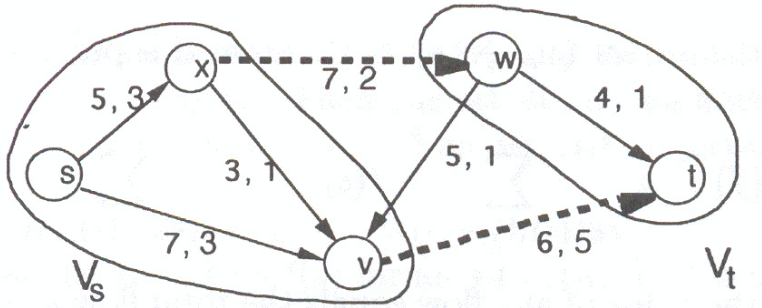
$$\sum_{v \in V_s} \sum_{e \in Out(v)} f(e) = \sum_{e \in \langle V_s, V_s \rangle} f(e) + \sum_{e \in \langle V_s, V_t \rangle} f(e) \text{ and}$$

$$\sum_{v \in V_s} \sum_{e \in In(v)} f(e) = \sum_{e \in \langle V_s, V_s \rangle} f(e) + \sum_{e \in \langle V_t, V_s \rangle} f(e) \quad (2)$$

Now enter the right hand sides of (2) into (1) and obtain the desired equality. □

Example

The flow *f* and cut $\langle \{s,x,v\},\{w,t\} \rangle$ shown in the figure illustrate Proposition 12.1.3.



$$6 = val(f) = \sum_{e \in \langle \{s,x,v\},\{w,t\} \rangle} f(e) - \sum_{e \in \langle \{w,t\},\{s,x,v\} \rangle} f(e) = 7 - 1$$

The next corollary confirms something that was apparent from intuition: the net flow out of the source *s* equals the net flow into the sink *t.*

Corollary 12.1.4 Let *f* be a flow in an *s*-*t* network. Then

$$val(f) = \sum_{e \in In(t)} f(e) - \sum_{e \in Out(t)} f(e)$$

Proof: Apply proposition 12.1.3 to the *s*-*t* cut $In(t) = \langle V_N - \{t\}, \{t\} \rangle$. □

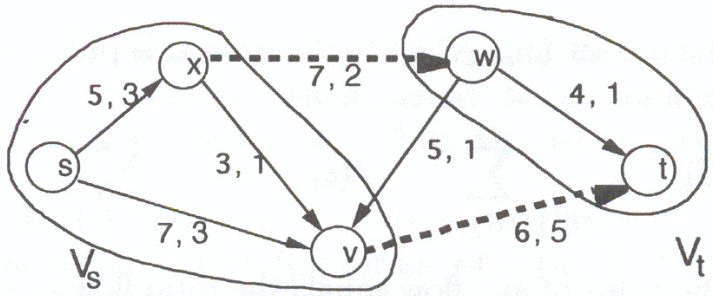
Example

Definition. The **capacity of a cut** $\langle V_s, V_t \rangle$ denoted **cap**$\langle V_s, V_t \rangle$, is the sum of the capacities of the arcs in cut $\langle V_s, V_t \rangle$. That is

$$cap\langle V_s, V_t \rangle = \sum_{e \in \langle V_s, V_t \rangle} cap(e)$$

Definition. The **minimum cut** of a network *N* is a cut with the minimum capacity.

Example. The capacity of the cut shown in the previous figure is 13,
And the cut ⟨{s,x,v,w},{t}⟩ with capacity 10, is the only minimum cut.



Maximum-Flow and Minimum-Cut Problems

The problems of finding the maximum flow in a capacitated network *N* and finding a minimum cut in *N* are closely related.

These two optimization problems form a *max-min* pair.

The following proposition provides an upper bound for the maximum-flow problem.

Maximum-Flow and Minimum-Cut Problems

Proposition 12.1.5 Let *f* be any flow in an *s*-*t* network, and let $\langle V_s, V_t \rangle$ be any *s*-*t* cut. Then $val(f) \leq cap\langle V_s, V_t \rangle$

Proof:

$$
\begin{aligned}
val(f) &= \sum_{e \in \langle V_s, V_t \rangle} f(e) - \sum_{e \in \langle V_t, V_s \rangle} f(e) && \text{(by proposition 12.1.3)} \\
&\leq \sum_{e \in \langle V_s, V_t \rangle} cap(e) - \sum_{e \in \langle V_t, V_s \rangle} f(e) && \text{(by capacity constraints)} \\
&= cap\langle V_s, V_t \rangle - \sum_{e \in \langle V_t, V_s \rangle} f(e) && \text{(by definition of } cap\langle V_s, \mathrm{V_t} \rangle) \\
&\leq cap\langle V_s, V_t \rangle && \text{(since each } f(e) \text{ is nonnegative)}
\end{aligned}
$$

□

Maximum-Flow and Minimum-Cut Problems

Corollary 12.1.6 (Weak Duality) Let *f** be a maximum flow in an *s-t* network *N*, and let *K** be a minimum *s-t* cut in *N.* Then

$$val(f*) \leq cap(K*)$$

Proof: This follows immediately from proposition 12.1.5.

Corollary 12.1.7 (Certificate of Optimality) Let *f* be a flow in an *s-t* network *N* and *K* an *s-t* cut, and suppose that *val(f) = cap(K).*
Then flow *f* is a **maximum flow** in network *N*, and cut *K* is a **minimum cut**.

Proof: Let *f'* be any feasible flow in network *N.*
Proposition 12.1.5 and the premise give

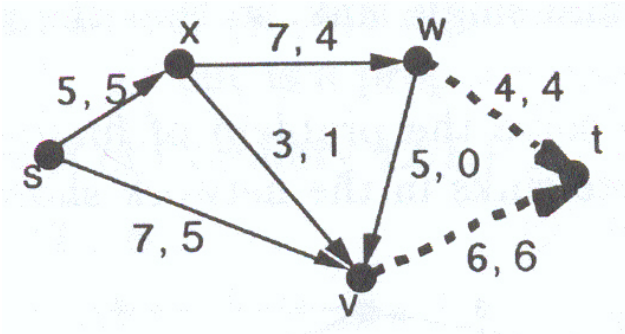
$$val(f') \leq cap(K) = val(f) \qquad \rightarrow f \text{ is a maximum flow}$$

On the other hand, let $\langle V_s, V_t \rangle$ be any *s-t* cut. Proposition 12.1.5:

$$cap(K) = val(f) \leq cap\langle V_s, V_t \rangle \qquad \rightarrow K \text{ is a minimum cut. } \square$$

Example

Example The flow for the example network shown in the figure has value 10, which is also the capacity of the *s-t* cut $\langle\{s,x,v,w\},\{t\}\rangle$.
By corollary 12.1.7, both the flow and the cut are optimal for their respective problem.



A maximum flow and minimum cut.

Corollary 12.1.8 Let $\langle V_s, V_t\rangle$ be an *s-t* cut in a network *N*, and suppose that *f* is a flow such that

$$f(e)=\begin{cases} cap(e) & \text{if } e\in\langle V_s,V_t\rangle \\ 0 & \text{if } e\in\langle V_t,V_s\rangle \end{cases}$$

Then *f* is a maximum flow in *N,* and $\langle V_s,V_t\rangle$ is a minimum cut.