A Branching System: Given ***R*** = {R(i): for each i = 1,...k,
R(i) is a subset of the nodes of a directed graph, G},
an R(i)-branching (i.e., a branching rooted at node-set R(i))
is subset, B(i), of the edges of the edges of G, such that
(a) B(i) forms a forest in G (i.e., does not contain the edges of a cycle in G);
(b) no edge of B(i) enters any node in R(i);
(c)for each node, v, not in R(i), exactly one edge of B(i) enters node v.
An "(edge-disjoint) branching system", ***B*** = {B(i): i = 1,...,k},
is where the edge-sets, B(i), are disjoint.

"**The optimum branching-system problem"** is:
Given a directed graph, G, and an R as above, and a cost c(e) ≥ 0 for each edge e of G,find a branching system, B, as above, such that the sum of its edge-costs is minimum.

There is a polytime algorithm for it. The simplest known statement of the algorithm uses matroid partitioning and matroid intersection.

[The "optimum branching system problem" is simpler to talk about and essentially the same in its methods of solution in the case where each R(i) is the same single node r.]

"**The optimum r-rooted branching system problem**"
asks: given a directed graph G, and a node r, an integer k, and a cost for each edge, find, if such exists, a cheapest set of k edge-disjoint branchings each rooted at r.

A r-branching B in G is a spanning tree B of G such that every path in B which starts at node r is a directed path.
i.e., Where r is a single node of G, an r-branching B in G is a spanning tree of G which has no edge entering node r, and exactly one edge entering every other node of G.

For a subset $S \subseteq V$ of the nodes of a digraph G = (V, E), a **cut,** d(S,G), means the set of edges of G which enter S, i.e., the edges j such that $h(j) \in S$ and $t(j) \notin S$.

(2.2.0) Edge-disjoint branching theorem:
Either there are k edge-disjoint r-branchings in G or there is a cut of size k-1 separating some non-empty subset S of nodes from r. (Not both.) In other words:
the maximum number of edge-disjoint r-rooted branchings in G = the minimum size of a cut separating some non-empty subset S of nodes from r.

Unlike finding k edge-disjoint paths in G from r to t,
in order to find k edge-disjoint r-branchings in G,
we grow a partial r-branching T (i.e., , for some $r \in R \subset V$,
T is an r-branching of the subgraph of G induced by R) such that

(2.2.1) **G' ≡ (V, [E(G) – E(T)]) has $|d(S, G')| \geq (k - 1)$,
∀ non-empty S not containing r.**

As soon as the node-set of T is V, we have the 1st of our r-branchings of G.
Because G' satisfies (2.2.1), we can repeat using G' instead of G to find a 2nd r-branching T' of G such that G'' ≡ (V, [E(G') – E(T')])
has $|d(S, G'')| \geq (k - 2)$, ∀ non-empty S not containing r.
And so on until we have k edge-disjoint r-branchings of G.

For this to work we must have
(2.2.2) Lemma. For any graph G ≡ (V, E(G)) such that
$|d(S, G)| \geq k$, ∀ non-empty S not containing r;
and any partial r-branching T in G, with node-set R, $r \in R \subset V$,
such that (2.2.1) holds,
there is an edge j = (u,v) with $u \in R$ and $v \in V - R$ such that
property (2.2.1) still holds when T is extended by edge j.

To prove Lemma (2.2.2) we use the following easy but important "submodularity lemma" which is left to the reader:

(2.2.3)For any digraph G = (V, E), and subsets A and B of V,

$$|d(A\cup B,G)| + |d(A\cap B,G)| \leq |d(A,G)| + |d(B,G)|.$$

To get the edge j of Lemma (2.2.2) is no problem if the inequality of (2.2.1) is "$\geq$ **k**" for every r∉S such that (S – R) is not empty. Otherwise let **A** be a minimal set, r∉A, and (A – R) not empty, such that $|d(A, G')| = (k - 1)$. There is an edge j = (u,v) where u ∈ A∩R and v ∈ (A – R) since $|d(A - R, G')| \geq k$.
For any other set B ⊆ V – r, we trivially still have, after we delete edge j from G′, at least (k – 1) edges entering B, unless u ∈ (V – B) and v ∈ B.
We have that also **when u ∈ (V – B) and v ∈ B**, since it then follows from submodularity that

$$|d(B, G')| \geq |d(A\cup B, G')| + |d(A\cap B, G')| - |d(A, G')| \geq k,$$

since $|d(A\cup B, G')| \geq (k - 1)$ by the construction of T, and
$|d(A\cap B, G')| \geq k$ and $|d(A, G')| = k - 1$
by the minimality choice of **A**. That proves ((2.2.2).

(2.2.4) "**The optimum (single) r-branching theorem":**
For a given directed graph G, and specified root node r,
and an integer edge-cost, c(e) ≥ 0, for each edge e of G,
the minimum cost of an r-branching in digraph G
= the maximum size of a multi-set H of cuts
("multi-set" means that H might contain multiple copies of a cut),
each separating a non-empty subset of nodes from r,
such that, for each edge e, H uses at most c(e) copies of cuts containing e.

The theorem is proved by an elegant polytime algorithm for finding a minimum cost r-branching in digraph G
(not nearly as simple as an algorithm
for finding a minimum cost spanning tree in a graph):

(2.2.5) Algorithm for finding an optimum r-branching in G.
Keep doing the following "Phase 1" until, in the graph resulting from the shrinkings, the chosen edges form an r'-branching where r' is either r or the pseudo-node containing r:

Choose a cheapest edge j directed toward a node or pseudo-node, v, toward which no unshrunken edge has yet been chosen.
If this creates a directed cycle C of chosen edges
then shrink C to be a pseudo-node, v(C).
For each unshrunken edge j directed toward v(C), let h be the edge of C which has the same head in C as j.
Change the effective cost of j from c(j) to c(j) – c(h).

Then do "Phase 2": Expand pseudo-nodes v(C) in the order they were created, and each time delete from the set of chosen edges the unique edge of C which will preserve that the remaining chosen edges form an r'-branching.

(2.2.6) Theorem. The final set of chosen edges form a least cost r-branching in G.

There are two interesting LP relaxations of finding an optimum r-branching which are obviously equivalent:

(2.2.7) minimize cx, where x ≥ 0 satisfies
(2.2.7.1) ∀ node v ≠ r, ∑[x(j): h(j) = v] = 1; and
(2.2.7.2) ∀`subset S ⊆ V, ∑[x(j): j has both ends in S] ≤ |S|- 1.

(2.2.8) minimize cx, where x ≥ 0 satisfies
(2.2.8.1) node v ≠ r, ∑[x(j): h(j) = v] = 1; and
(2.2.8.2) ∀`subset S ⊆ V, ∑[x(j): j ∈ d(S,G)] ≥ 1.

The incidence vector x of the r-branching given by algorithm (2.2.5) in fact solves (2.2.7) or (2.2.8).
The proof is to derive from the algorithm an (integer-valued) optimum for the dual linear program.
For LP (2.2.8) that dual optimum gives Theorem (2.2.4).

From LP (2.2.7) we see **optimum r-branchings** are an instance of optimum matroid intersections.

[∀ ⊂ ⊆ ∪ ∈ ∈ ∉ ∈, ≡ , | , ≥ , ∩, ≤, ∑, ≠

]