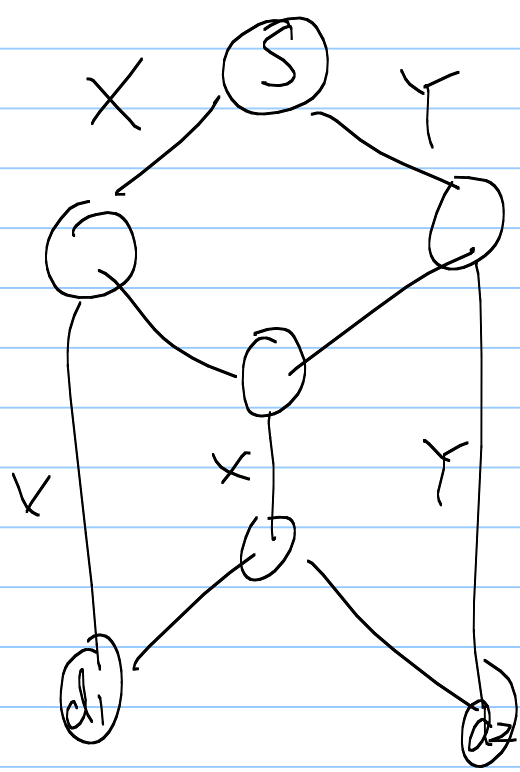
Lecture 27

Note Title 4/23/2012

* The throughput advantage of network coding & its main distinction from fountain codes.

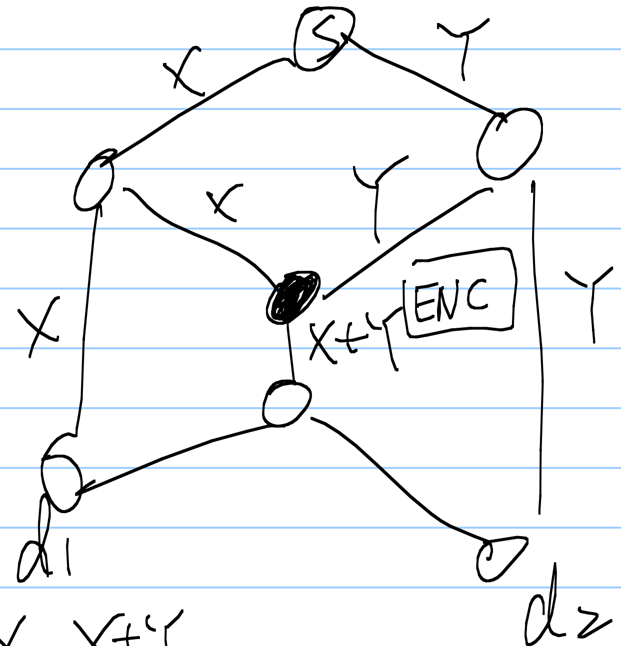
The butterfly example



We would like to send two packets X & Y to both d_1 & d_2. A multicast session $(s, \{d_1, d_2\})$

* Pure routing each time only 1 receiver can receive 2 pkts & the other receiver can only receive 1 pkt. $\Rightarrow$ The overall rate is 1.5 pkt/slot.

* Network coding



by performing XOR at the rate is enhanced to 2 pkt/slot.

X, X+Y

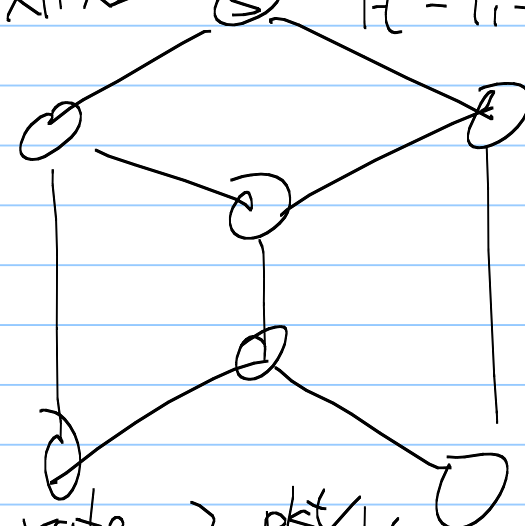
DEC X, Y

X+Y, Y

X, Y

* Can we use fountain codes at the sources to achieve the same throughput

$X_t' = X_1 + X_2 \cdots$ S $Y_t' = Y_1 + Y_2 + \ldots$



Ans: No, it does not increase the throughput

To achieve rate 2 pkt/slot coding needs to be performed at the intermediate node which coins the term "network coding"

Comparison: O — $\boxed{\text{BEC}}$ — O — $\boxed{\text{BEC}}$ — O

p p

Routing

End-to-end fountain codes
(End-to-end erasure control coding)

link by link error control coding

Network coding

Routing

$\boxed{\text{send}}$ ------> $\boxed{\text{relay}}$ ------> $\boxed{\text{receive}}$

With Fountain codes

$\boxed{\text{ENC}}$ ------> $\boxed{\text{Relay}}$ ------> $\boxed{\text{DEC}}$

Link-by-link ECC

$\boxed{\text{ENC}}$ ------> $\boxed{\text{DEC}}$ → $\boxed{\text{ReENC}}$ ------> $\boxed{\text{DEC}}$

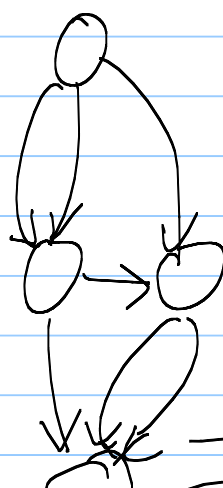
coding at intermediate nodes.

Network coding

$\boxed{\text{ENC}}$ ------> $\boxed{\text{ReENC even when decoding is impossible}}$ ------> $\boxed{\text{DEC}}$

* The min-cut/max-flow capacity of a Network coded multicast sessions.

* Consider a directed acyclic network such that each edge can transmit 1 symbol per unit. High-rate links are modeled as parallel edges



An direced acyclic graph is denoted by $G=(V,E)$, V: the set of nodes. E: the set of edges.

* Def: a (s,t) edge cut C is the collection of edges s.t. once removed, s & t are "disconnected".

  Example $\underline{E}$ is always a (s,t) cut.

Def:

* The value of a (s,t) cut is $|C|$, the number of edges in C.

Def: The minimum (s,t) cut (sometimes called the min (s,t)-cut). is the (s,t) cut that has the smallest value.

Def: min (s,t) cut value = the value of the min (s,t)-cut.

Def: A (s,t) flow is a collection of edges s.t. for each $v \neq s,t$.

In-flow $|\{(u,v) : (u,v) \in f\}|$

$=$ out-flow $|\{(v,w) : (v,w) \in f\}|$ the flow conservation law.

& $\exists$ no (v,s) edges or (t,v) edges in f.

Namely the source s has only outgoing edges, the destination t has only incoming edges.

Def: The value of a (s,t)-flow f

$$\left|\{(s,v) : (s,v) \in f\}\right| = \left|\{(v,t) : (v,t) \in f\}\right|$$

Def: the maximum (s,t)-flow is the flow that has the largest value.

Def: the max (s,t)-flow value

Lemma: Each (s,t)-flow with value g can be converted to g edge-disjoint paths from $s \to t$. & Vice Versa.

pf: ① Start from s, randomly pick 1 outgoing edge (s,v), & move to v. remove that edge.

② While $v \neq t$, randomly pick 1 outgoing edge (v,u) & move to u & remove that edge

③ The trace from $s \to t$ corresponds to 1 edge-disjoint path.

Repeat ① ② ③ for totally g times.

We sometimes need to consider the <u>residual graph</u>. (so that we can "back off" from the previous choices.)

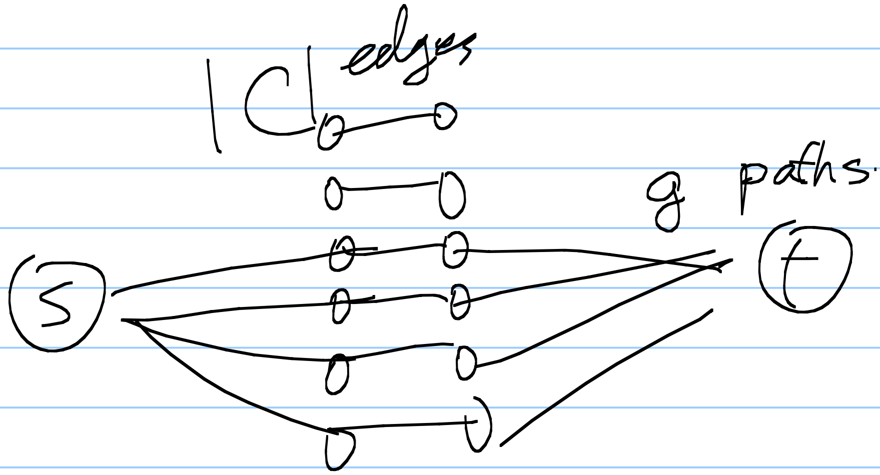
Pf of correctness: By the flow conservation law.

(The converse is straightforward.

Lemma: For any (s,t)-flow f & (s,t)-cut C, we have

min value of $C \geq$ max value of f.

pf



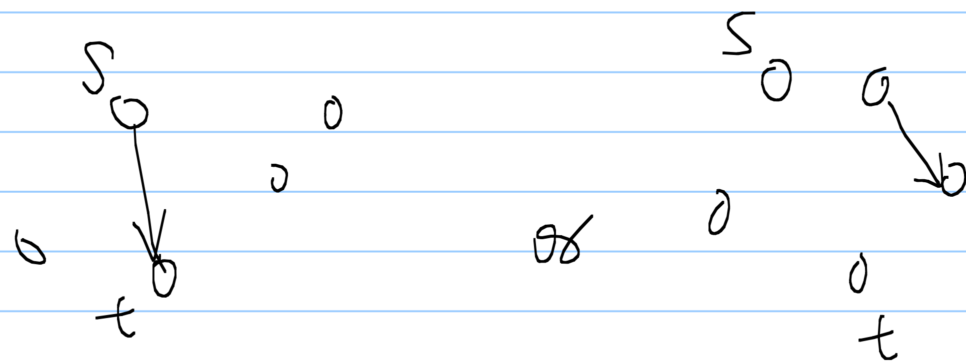
$\boxed{\text{The min-cut/max-flow theorem.}}$ also known as Menger's theorem 1927

Theorem: For any graphs with source s & dest t.

$$\text{min cut-value}^{(s,t)} = \text{max flow-value}^{(s,t)}$$

A stronger version: $(S,T) = (S,T)$

Pf: By induction: When $|E| = 1$, either



The min-cut-value = max-flow value.

Suppose it is true for $|E| \le N$.

When $|E| = N+1$. Suppose the

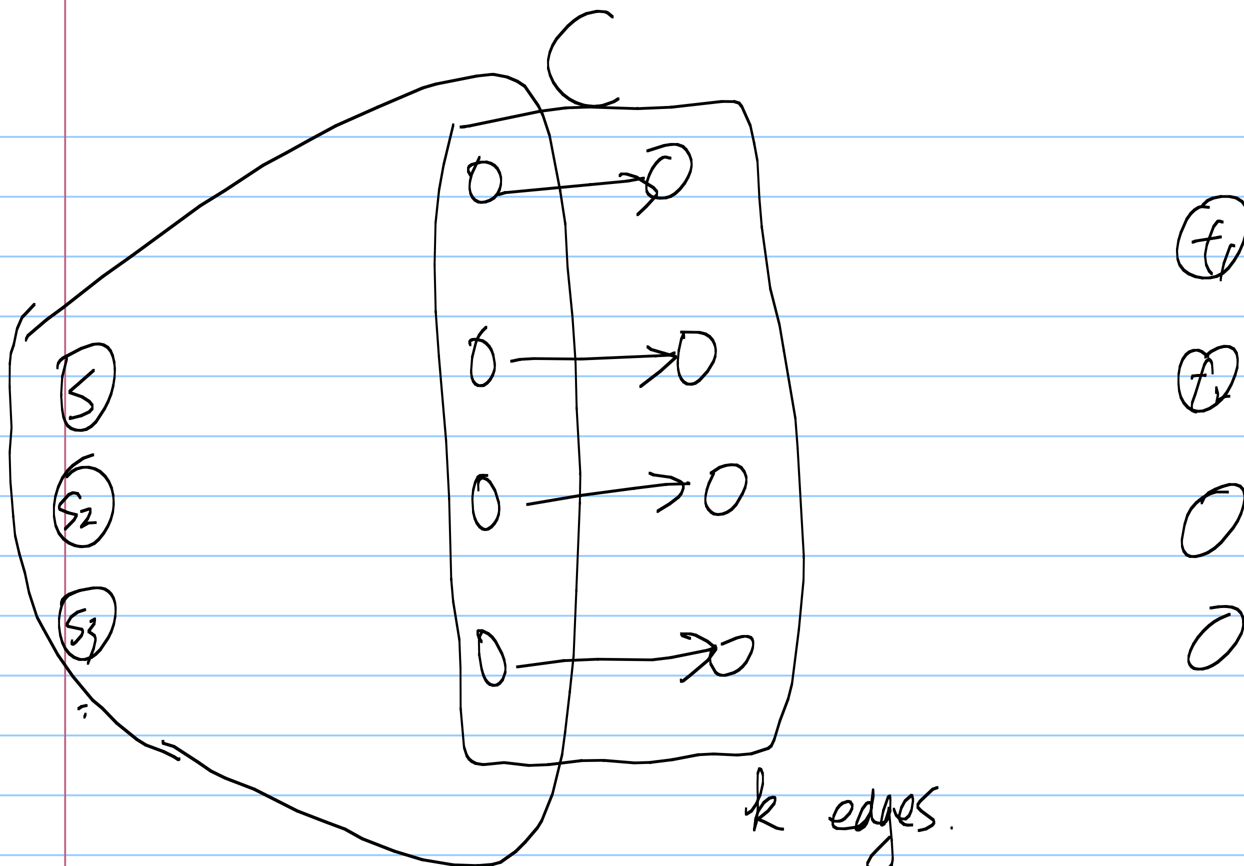
$k \triangleq$ min-cut value $\geq$ max-flow value

We will construct k-edge-disjoint paths from s to t.

Case 1: When $k=0$, since S T are disconnected, max-flow-value $= 0$ ✓

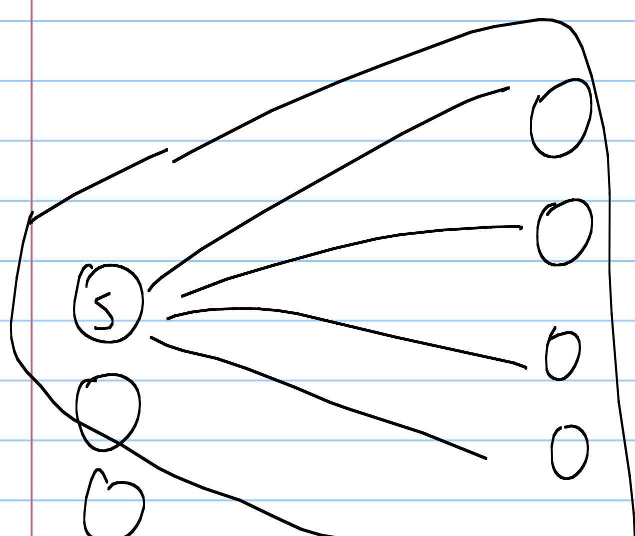
Case 2: When $k>0$,

Step 1: Choose any min-cut C.



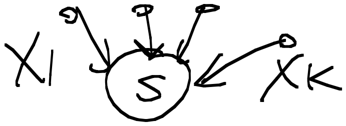
k edges.

Consider a new graph G_1



Since G_1 has a smaller # of edges. In any cut for G_1 is a cut for G. By induction we have k paths connecting S & new T'. Similarly, we have k paths ~~too~~ from S' to T $\Rightarrow$ We can construct new k-paths

* Def: $In(v) = \{(u,v) : (u,v) \in E\}$ the incoming edges
* Def: A feasible network code that sends K symbols $Y_1, \cdots, Y_K \in GF(2^b)$ from a single source s to a single desti. d is defined by M_e: the coded message M on edge e s.t.



an alternative perspective to unify the ENC functions.

$$M_{(s,u)} = f_{(s,u)}(Y_1, \cdots, Y_k) \in GF(2^b)$$

A function indexed by (s,u)

For $v \neq s$

$$M_{(v,w)} = f_{(v,w)}\left(M_{(u,v)} : (u,v) \in In(v)\right)$$

$\boxed{f_e\text{: the encoding functions}}$

& there exists a vector-input / vector-output $\boxed{\text{decoding function}}$

$$(\hat{Y}_1, \cdots, \hat{Y}_k) = f_{DEC}\left(M_{(v,t)} : (v,t) \in In(t)\right)$$

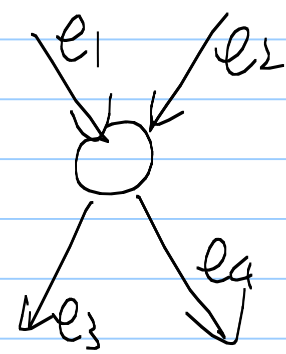
s.t. $\forall k$, $\hat{Y}_k = Y_k$ for all $Y_1, \cdots, Y_k$ values.

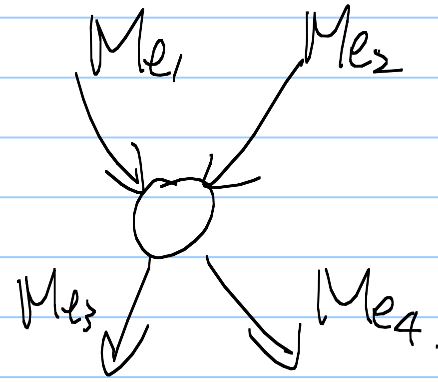
f_e

* The definition allows arbitrary encoding & decoding functions at the intermediate nodes

f_{DEC}

Ex:





We have the freedom of designing two functions

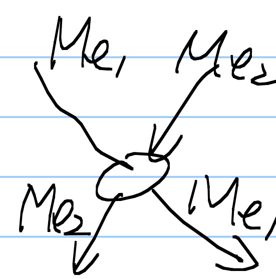
$M_{e_3} = f_{e_3}(M_{e_1}, M_{e_2})$

$M_{e_4} = f_{e_4}(M_{e_1}, M_{e_2})$

By choosing different f_{e_3}, f_{e_4} functions, network coding can perform

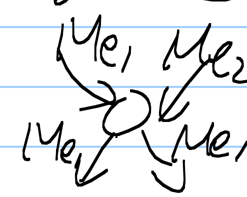
① relay : $f_{e_3}(M_{e_1}, M_{e_2}) = M_{e_2}$

$f_{e_4}(M_{e_1}, M_{e_2}) = M_{e_1}$



② duplication : $f_{e_3}(M_{e_1}, M_{e_2}) = M_{e_1}$

$f_{e_4}(M_{e_1}, M_{e_2}) = M_{e_1}$



③ Blocking: $f_{e_3}(\cdot,\cdot) = 0 = f_{e_4}(\cdot,\cdot)$

④ Other encoding: $f_{e_3}(M_{e_1}, M_{e_2}) = M_{e_1} \cdot M_{e_2}$

* The optimization question is: Given the underlying network topology, what is the largest K^* info symbols of a feasible network code by arbitrarily design the f_e & f_{DEC} functions.

* It can be generalized to a multicast session that sends $Y_1, \cdots, Y_K$ from S to multiple destination t_i.

(all of t_i requested the same # of symbols.)

The only changes are

$$(\hat{Y}_{i,1}, \cdots, \hat{Y}_{i,K}) = f_{i,DEC}\left(M_{(v,t_i)} : (v,t_i) \in In(t_i)\right)$$

$\hat{Y}_{i,k} = Y_k \ \forall k$, & $Y_1 \cdots Y_K$ values

* The goal is again to maximize K.