<u>**EX**</u>: Estimating the size & shape of a search tree. Consider sequence of r.v.s, $X_0, X_1, \ldots$, where $X_l$ estimates # nodes @ level *l*. We want $E\left[X_l\right] = $ E[# nodes @ level *l*]. $\textit{SIZE} = \sum_{0 \le i} X_i$, so *size* is a r.v.

<u>Experiment</u>: $X_0$:=1

$l$ :=0
*current_node* ← root of tree
*neighbors* ← *children* (*current_node* )
**while** |*neighbors* | > 0 **do**
  $l \leftarrow l+1$
  $X_l \leftarrow |neighbors| * X_{l-1}$
  *current_node* ← random_select(*neighbors* )
  *neighbors* ← *children* (*current_node* )
*shape* ← $(X_0, X_1, \ldots)$ $\textit{SIZE} \leftarrow \sum_{0 \le l} X_l$

<u>CLAIM</u>: $X_l$ is an unbiased estimator. (Proof by induction on *l*).

*******************************************************************

**EX:** STABLE MARRIAGE *n* men, *n* women, *preferences*, *marriage*, *unstable marriage*, *stable marriage*

<u>**THEOREM**</u>: There always exists a stable marriage.

<u>**PROOF**</u>: *Male Proposal Strategy* (*MPS*), & no man runs off his list.

<u>Analysis</u>: worst-case $n^2$ proposals

average-case Consider *Amnesiac Proposal Strategy* (*APS*) where, upon rejection, man proposes to a random woman. Let rv $X_m$ - # proposals of *MPS*
rv $X_a$ - # proposals of *APS*

Clearly $\forall q \Pr\left[X_m > q\right] \le \Pr\left[X_a > q\right]$ Trial *k* is *success* if $k^{\underline{th}}$ coupon not previously drawn.

$X$ - # trials until $n^{\underline{th}}$ success.
rv $X_k$ – # trials from $k^{\underline{th}}$ success → $(k+1)^{\underline{st}}$ success.

$$X_a = \sum_{n-1 \ge k \ge 0} X_k \;, \quad E\left[X_a\right] = n \sum_{n \ge k \ge 1} \frac{1}{k} = nH_n .$$

*******************************************************************

<u>**DEF**</u>: A *cut* in graph is minimal set of edges whose removal increases # of components. *min-cut* is cut of minimum cardinality.

Karger & Stein, http://www.acm.org/pubs/contents/journals/jacm/1996-43/#4

<u>**FIND MIN-CUT**</u>: (Edge-)*contraction* from (multi)graph ⇒ multigraph identifies endpoints of an edge (keeping multiple edges, but removing loops).

*Contract*(*G*,2) {Contract *G* down to **output** of two vertices)
**while** $\left|V\left(G\right)\right| > 2$ *edge-contract*(*random*( $e \in E\left(G\right)$ ))
**return** {edges between two vertices of *G*}

<u>Analysis</u>: There are *n*-2 edge-contractions.

$\boxed{\text{Every surviving cut of contracted graph corresponds to a cut of the original graph of same value.}}$

Let *C* be a min-cut. Every vertex has degree $\ge |C|$. *G* has at least $n|C|/2$ edges.

$\Pr(C \text{ survives contraction}) = \left(1 - \frac{|C|}{|E|}\right) \geq \left(1 - \frac{2}{n}\right)$. If *i* previous edge-contractions, $n-2 \geq i \geq 1$, avoided *C*, then *C* is still min-cut, *G* has *n*-*i*+1 vertices and $\geq |C|(n-i+1)/2$ edges, probability of edge-contraction *i* avoiding *C* is $\geq \left(1 - \frac{2}{n-i+1}\right)$.

Pr[*C* surviving]$\geq \prod_{n-2 \geq i \geq 1} \left(1 - \frac{2}{n-i+1}\right) = \frac{2}{n(n-1)} \geq \frac{2}{n^2}$. Can be executed in time $O(n^2)$.

After *m* executions, Pr[error]$\leq \left(1 - \frac{2}{n^2}\right)^m$. After $(n^2 \ln n)/2$ executions, Pr[error]$\leq 1/n$.

*****************************************************************

**<u>Ex</u>**: 2-SAT

SAT A *variable* is $x_1, x_2, ...$

A *literal* is a variable or its negation - $x_i$, $\overline{x_i}$

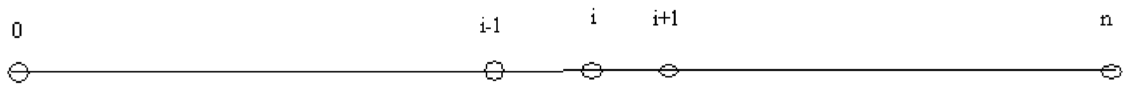
A *k-clause* is a disjunction of *k* literals - $\left(x_2 \vee \overline{x_3} \vee x_3\right)$ is a 3-clause

A *k-CNF* is a conjunction of *k-clauses* - $\left(x_1 \vee \overline{x_2}\right) \wedge \left(x_2 \vee x_3\right) \wedge \left(\overline{x_1} \vee \overline{x_3}\right)$ is a 2-CNF

An *interpretation* is a function $\iota : \{\text{variables}\} \rightarrow \{\text{true, false}\}$. A *k-CNF F* is *satisfiable* if there is an interpretation $\iota$ which makes it true.

**<u>THEOREM</u>** (Papadimitriou): If *F* is a satisfiable 2-CNF with *n* variables, then there is an algorithm to find a satisfying interpretation in at most $2n^2$ steps with probability $\geq 1/2$.

**<u>PROOF</u>**: Let $\iota_0$ be a satisfying assignment of *F*, and for any interpretation $\iota$, mark its position by the number of variables for which it disagrees with $\iota_0$.




ALGORITHM: **while** $\iota$ doesn't satisfy *F*
randomly select an unsatisfied clause *c* of *F*
randomly select a literal $x_i$ of *c*
flip the value of $x_i$ in $\iota$

<u>ANALYSIS</u>: If the position of $\iota$ is *i*, then the position of the new interpretation is *i*-1 or *i*+1.
Let $t(i)$ be expected # steps for interpretation at *i* to reach 0.

$$t(0) = 0$$

$$t(n) = 1 + t(n-1)$$

Let $p_{i,j}$ - proability particle moves from position *i* to position $j \in \{i-1, i+1\}$

$$t(i) = p_{i,i-1}\left(1 + t(i-1)\right) + p_{i,i+1}\left(1 + t(i+1)\right)$$

Because $\geq 1$ literal in an unsatisfied clause disagrees with $\iota_0$, $p_{i,i-1} \geq 1/2$.

Thus $t(i) \leq \frac{t(i-1) + t(i+1)}{2} + 1$ which admits the solution $t(n) \leq n^2$.

MARKOV'S INEQUALITY: For any rv $X$ and $\lambda \in \Re$, $\Pr(X \geq \lambda) \leq \frac{E[X]}{\lambda}$.

Letting $\lambda = 2n^2 \leq 2t(n)$, $\Pr(X \geq 2n^2) \leq \frac{n^2}{2n^2} = \frac{1}{2}$

## Ex: Polynomial = 0?

```
> a := (sin(x)^2 - cos(x)*tan(x)) * (sin(x)^2 +
cos(x)*tan(x))^2;
```

$$a := (\sin(x)^2 - \cos(x)\tan(x))(\sin(x)^2 + \cos(x)\tan(x))^2$$

```
b := 1/4*sin(2*x)^2 - 1/2*sin(2*x)*cos(x) - 2*cos(x)^2
       + 1/2*sin(2*x)*cos(x)^3 + 3*cos(x)^4 - cos(x)^6;
```

$$b := \frac{1}{4}\sin(2x)^2 - \frac{1}{2}\sin(2x)\cos(x) - 2\cos(x)^2 + \frac{1}{2}\sin(2x)\cos(x)^3 + 3\cos(x)^4 - \cos(x)^6$$

```
testeq( a = b );
```

*true*

Maple:

- The function **testeq** tests for equivalence probabilistically. It returns **false** if the expressions are not equal (or not equal to 0) and **true** otherwise for the class of expressions that **testeq** recognizes. The result **false** is always correct; the result **true** may be incorrect with very low probability.
- This function will succeed over expressions formed with rational constants, independent variables, and **I**, combined by arithmetic operations, exponentials, trigonometrics and a few others. It may also succeed with some expressions involving algebraic constants and functions and involving **Pi** as an argument of trigonometrics. If the expressions do not fall in this class, **testeq** returns **FAIL**. **testeq** may also return **FAIL** if it cannot find an appropriate modulus that works after seven trials.

**THEOREM** (Zippel, Schwartz): If $P(x_1,\ldots,x_n)$ is a nonzero polynomial of degree $d$ over field $F$ and $S \subseteq F$ and $(s_1,\ldots,s_n)$ is a random element of $S^n$, then

$$\Pr\{P(s_1,\ldots,s_n) = 0\} \leq \frac{d}{|S|}.$$

ISOMORPHISM OF UNORDERED ROOTED TREES

Associate a polynomial with an unordered rooted tree by

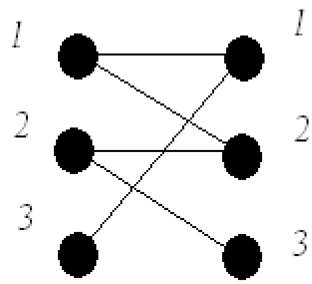
- If $v$ is a leaf (has height 0), then $f_v = x_0$,
- If $v$ has height $k$ and children $v_1,\ldots,v_m$, then $f_v = (x_k - f_{v_1})(x_k - f_{v_2})\ldots(x_k - f_{v_m})$.

If tree has root $r$, then its polynomial is $f_r$. Trees with roots $r_1$ and $r_2$ are isomorphic iff

$$f_{r_1} - f_{r_2} = 0$$

PERFECT MATCHINGS IN BIPARTITE GRAPH $G = (\{1,\ldots,n\},\{1,\ldots,n\},E)$

Form array $X_G$ with $(i,j)$ entry $= x_{ij}$ if $ij \in E$ and 0 otherwise.




$$X_G = \begin{bmatrix} x_{11} & x_{12} & 0 \\ 0 & x_{22} & x_{23} \\ x_{31} & 0 & 0 \end{bmatrix} \qquad \det X_G = x_{12}x_{23}x_{31} \neq 0$$

To test if $\det X_G = 0$ we choose some prime $p>2n$ and randomly set $x_{ij} \leftarrow U\left(\mathbb{Z}_p\right)$. There are algorithms to test efficiently if det$X$=0 if entries of $X$ are drawn form $\mathbb{Z}$

$$\Pr\left\{\det X_G\,|_{\mathbb{Z}^n} = 0\right\} = \begin{cases} 1, \text{if } \det X_G = 0 \\ < \frac{1}{2}, \text{if } \det X_G \neq 0 \end{cases}.$$

*****************************************************************

**Chapt. 26** Maximum (steady-state, single source, single sink) Flow in Networks

A *network* is:

-a digraph $(V,E)$,

-$s,t\in V$, called *source* and *sink*

-*capacity* $c{:}E\to\mathbf{R}^+$

A *flow*, $f{:}E\to\mathbf{R}^+$ satisfies:

**CAPACITY CONSTRAINT**-$(\forall e \in E) 0 \leq f(e) \leq c(e)$

**FLOW CONSERVATION**- $\sum_u f(u,v) = \sum_u f(v,u)$ for all $v\in V\text{-}\{s,t\}$

The *total flow* of $f$ is $|f| = \sum_{v\in V} f(s,v) = \sum_{v\in V} f(v,t)$. $\quad f: E \to 0$ is a flow.

Given a network, find a maximum total flow.

**GENERALIZATION**: *Multiple sources/Multiple sinks* Given sources $\{s_1,\ldots,s_j\}$ and sinks $\{t_1,\ldots,t_k\}$, reduce to single source/single sink by adding a super source $s^*$ and a super sink $t^*$ and edges $\{s^*\}\times\{s_1,\ldots,s_j\}\cup\{t_1,\ldots,t_k\}\times\{t^*\}$ with $\infty$ capacities.

**DEF**: For $S\subset V$ such that $s\in S$ and $t\in T=V\text{-}S$, the *cut* $(S,T)=\{uv\in E \mid (u\in S\wedge v\in T)\vee(v\in S\wedge u\in T)\}$.

**THEOREM**: For every cut $(S,T)$, the total flow is $|f| = \sum_{e\in(S,T)} f(e) - \sum_{e\in(T,S)} f(e)$.

**PROOF**: (by induction on $|S|$). Basis: trivial. Assume for $|S|-1$. Pick $v\in S-\{s\}$. In moving $v$ to $S$ in cut $(S-\{v\}, T\cup\{v\})$, decrease flow across cut by $\sum_{u\in S-\{v\}} f(uv) + \sum_{u\in T} f(uv)$ and increase flow across cut by $\sum_{u\in S-\{v\}} f(vu) + \sum_{u\in T} f(vu)$, and by flow conservation this change is 0.

**DEF**: For any $S \subset V$ such that $s \in S$ and $t \in T$, the *capacity of the cut* determined by $S$ is $c(S) = \sum_{e \in (S,T)} c(e)$.

**THEOREM**: For every flow $f$ and every $S \subset V$, $|f| \le c(S)$.

**PROOF**: $|f| = \sum_{e \in (S,T)} f(e) - \sum_{e \in (T,S)} f(e) \le \sum_{e \in (S,T)} f(e) \le \sum_{e \in (S,T)} c(e) = c(S)$.

**COROLLARY**: If $f$ and $S$ satisfy the previous equation with equality, then $f$ is maximum (*max-flow*) and the cut defined by $S$ is minimum (*min-cut*).

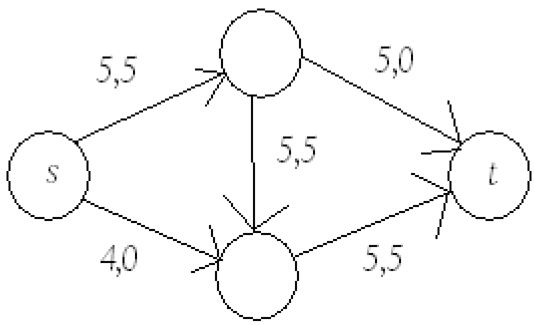
NOTE: Can't enumerate cuts to find max flow, because there are $2^{|V|-2}$ of them.

**DEF**: An *augmenting path* relative to flow $f$ is a sequence of edges from $s$ to $t$ such that for each edge $e$ on the path:

-if $e$ points from $s$ to $t$, then $f(e) < c(e)$, (*forward edge*)

-if $e$ points from $t$ to $s$, then $f(e) > 0$. (*back edge*)

Greed fails –




**ALG**: -Start with a legal flow $f$, such as $f(e)=0$ for all $e \in E$.

-**while** there exists an augmenting path relative to $f$

push as much extra flow as possible through the path

How do we find an augmenting path? How much extra flow can be pushed through an augmenting path?

**LABELING ALGORITHM**:

-label $s$ {every $v \in V$ for which we can find augmenting path $s \to v$ is labeled; if $t$ labeled, then done.}

-*forward labeling* of $v$ by edge $u \xrightarrow{e} v$ applicable if

-$u$ labeled,

-$v$ not labeled,

-$c(e) > f(e)$

-$v$ gets label $e$, $\Delta(e) \leftarrow c(e) - f(e)$

-*backward labeling* of $v$ by edge $u \xleftarrow{e} v$ applicable if

-$u$ labeled,

-$v$ not labeled,

-$f(e) > 0$

-$v$ gets label $e$, $\Delta(e) \leftarrow f(e)$

**FORD & FULKERSON ALG**:

-Start with a legal flow $f$, such as $f(e)=0$ for all $e \in E$.

(**) -mark $s$ labeled, all $v \in V-\{s\}$ unlabeled.

**-while** an unlabeled vertex $v$ can be labeled

-label $v$

**-if** $v=t$

let the augmenting path be $s = v_0 \overset{e_1}{-} v_1 \overset{e_2}{-} v_2 \overset{e_3}{-} \ldots - v_{l-1} \overset{e_l}{-} v_l = t$

let $\Delta = \min_{1 \le k \le l} \Delta(e_k)$

**if** $e_k$ forward (i.e., $v_{k-1} \xrightarrow{e_k} v_k$) **then** $f(e_k) \leftarrow f(e_k)+\Delta$

**if** $e_k$ backward (i.e., $v_{k-1} \xleftarrow{e_k} v_k$) **then** $f(e_k) \leftarrow f(e_k)-\Delta$

**goto** (**)

**-quit** {$f$ is a maximum flow}

**THEOREM**: Algorithm identifies a min-cut (hence a max flow). That is, the following are equivalent for flow $f$ in $G$:

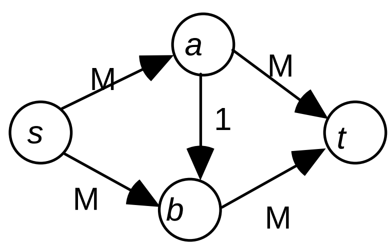
- $f$ is a max flow,
- $G$ admits cut $(S,T)$ with $c(S,T)=|f|$,
- $G$ does not admit an augmenting path relative to $f$.

**PROOF**: Let $S$ be set of labeled vertices when algorithm quits. $s \in S$ and $t \in V-S=T$. Consider the cut defined by $S$. $(\forall e \in (S,T))\ f(e)=c(e)$. $(\forall e \in (T,S))\ f(e)=0$.

♦ If initial flow integral & all capacities integral, algorithm never introduces fractions.

-Analysis

Even if capacities are integral, may be real slow: initially, $f(e)=0$ for all $e \in E$




Augmenting paths: *s-a-b-t, s-b-a-t, s-a-b-t,...* +1 every time
need $2M$ augmentations before flow of $M$ reached.

There is a case with irrational capacities for which the algorithm neither terminates nor converges.