

Written by: Gagan Aggarwal, Rajeev Motwani
Devavrat Shah, An Zhu

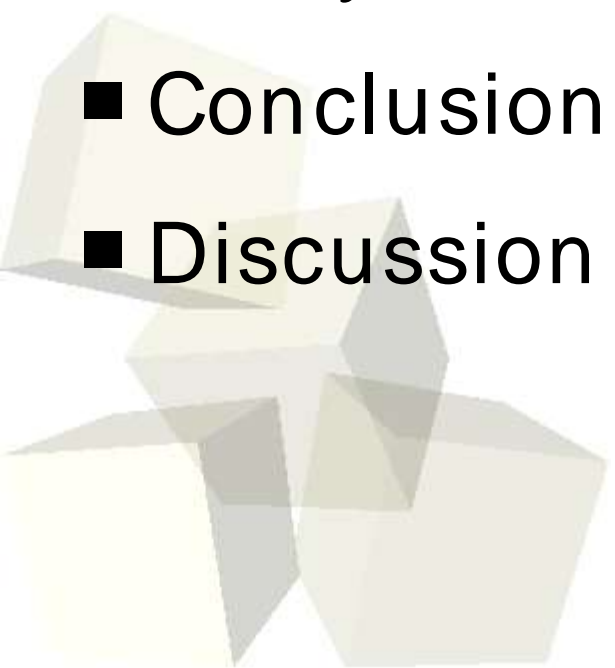
Presentation by: Charlie Wiseman

Discussion Leader: Jing Lu

Faculty Editor: Patrick Crowley

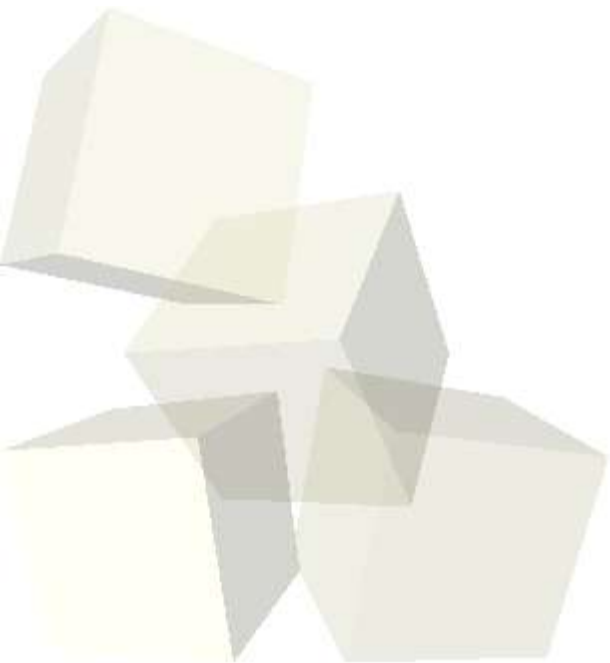


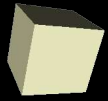
- Introduction
- Background
- Previous work
- New algorithm
- Analysis
- Conclusion
- Discussion



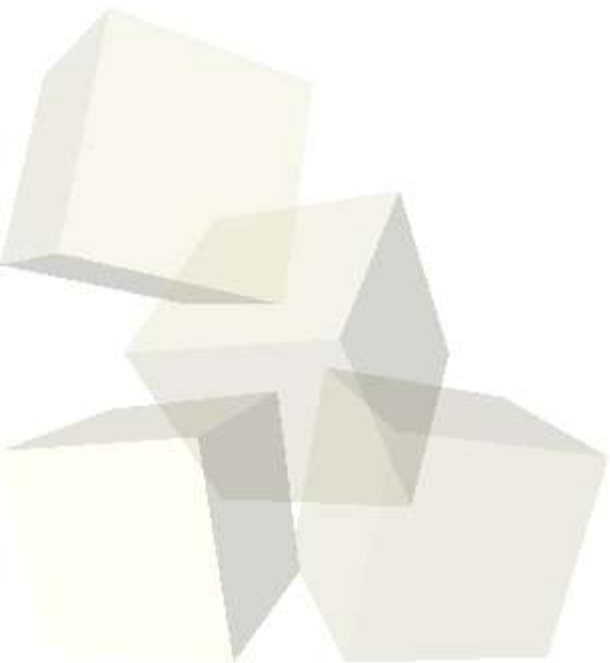


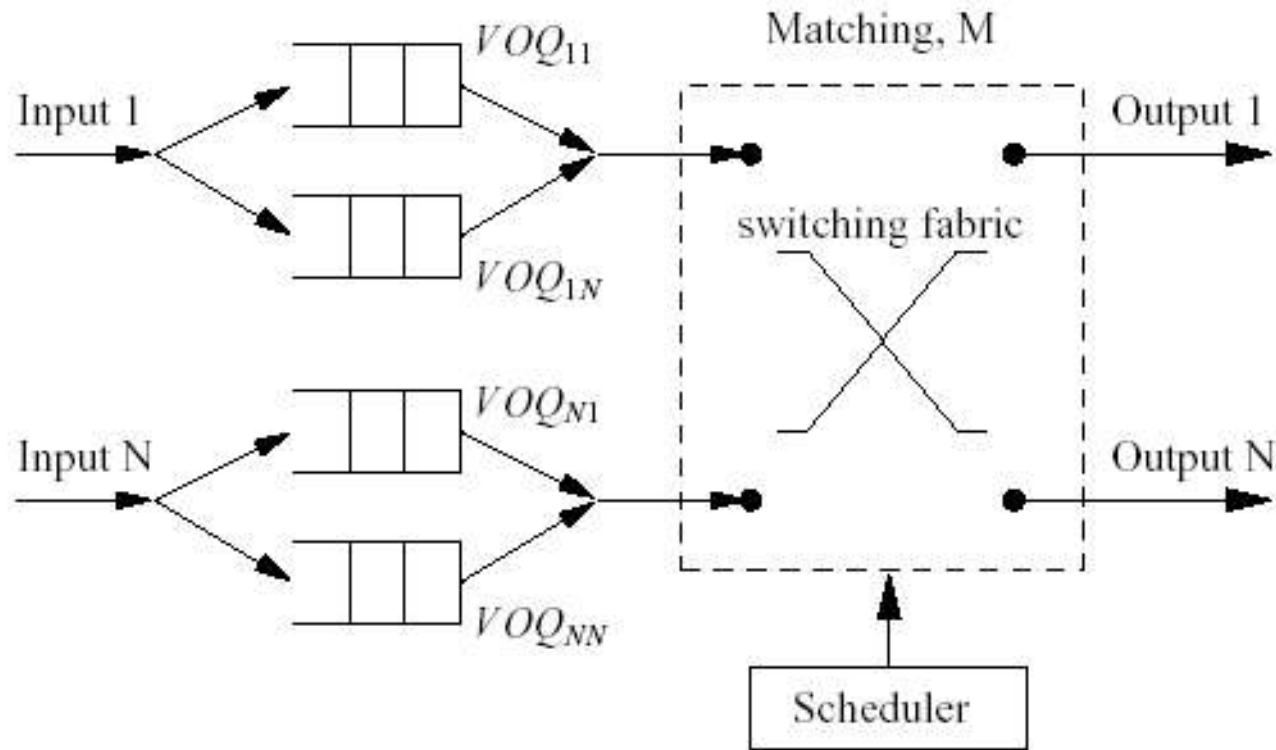
- Analytic paper
- Results apply to switching theory and graph theory
- Pencil and Paper proofs
- Stable, implementable algorithms





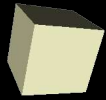
- Switching
- Probability
- Asymptotic notation



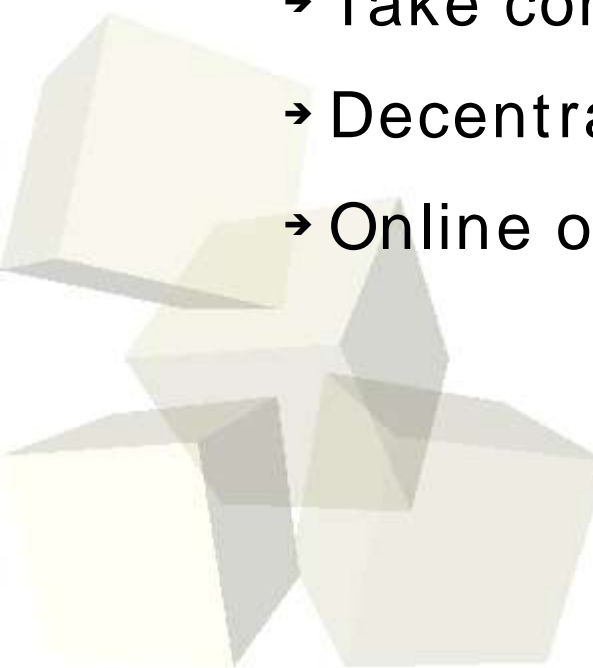


Input-queued NxN packet switch

- Virtual output queues
- Crossbar switching fabric

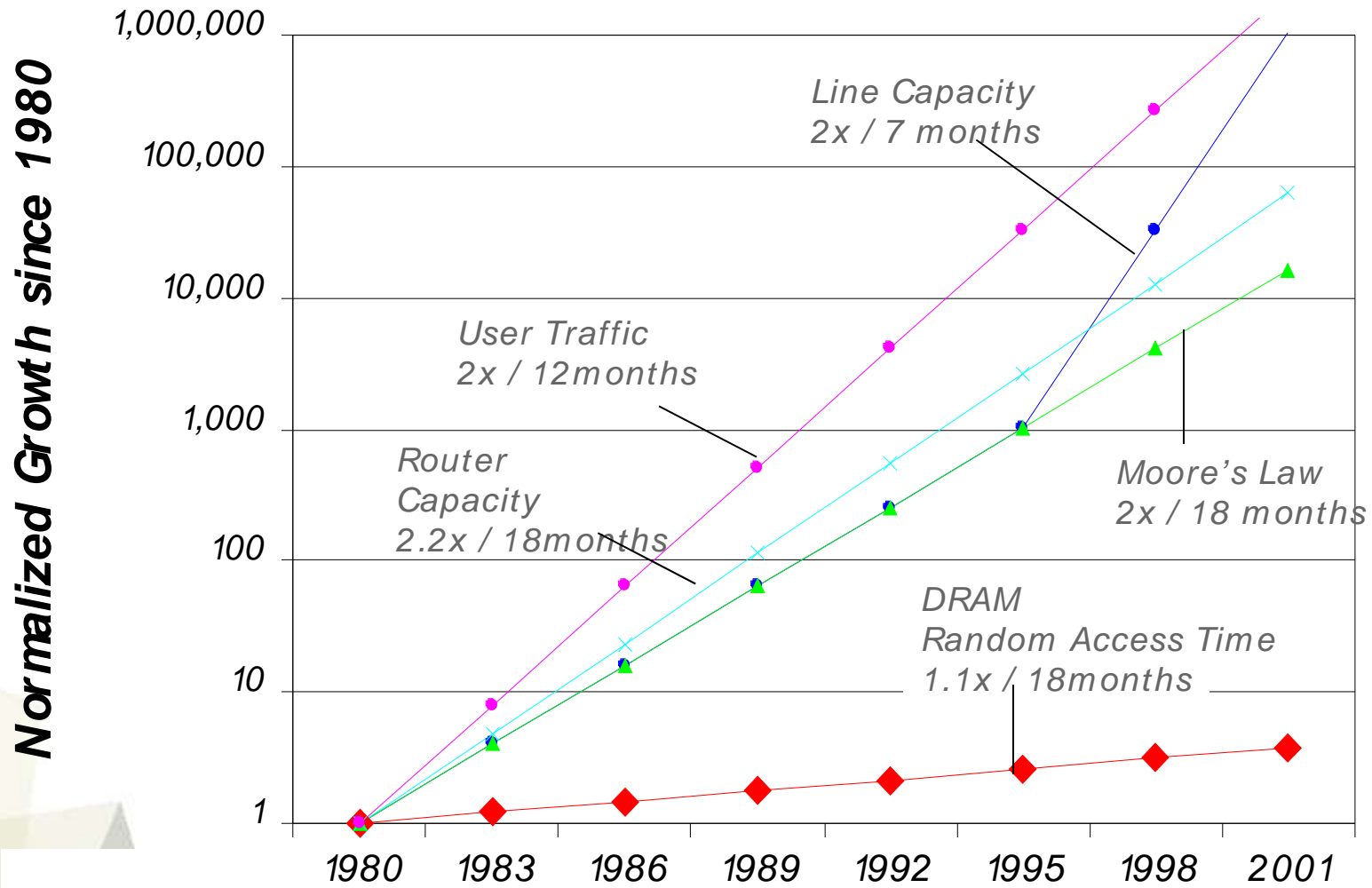


- Route packets from input port to output port
- Need a scheduling algorithm
 - ◆ Stable (reliable operation)
 - ◆ Implementable (hardware imposed constraints)
 - Must be relatively simple
 - Take constant time per switch time step
 - Decentralized operation
 - Online operation

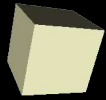




Switching



Graph source: Nick McKeown, Stanford University



■ Bernoulli distribution

- ◆ Random variable
- ◆ Two possible outcomes: 0, 1
- ◆ With probabilities: p , $1-p$
- ◆ Defined, then, by one parameter, p

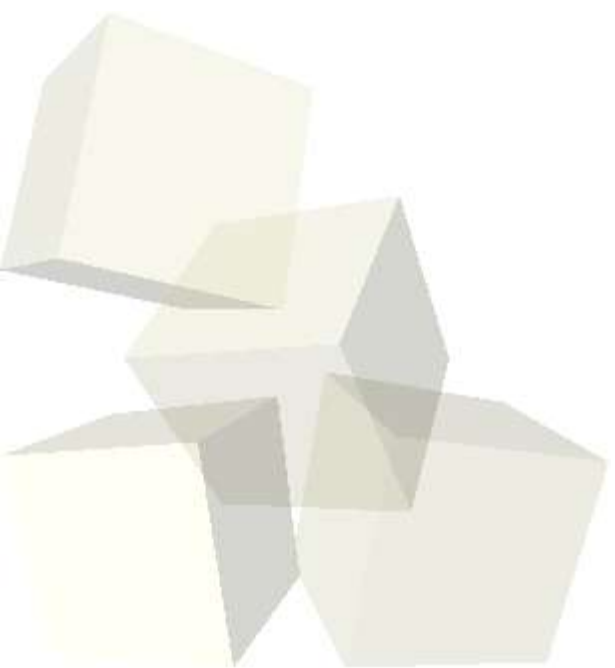
■ Often used in sets, like $A_{ij}(t)$

■ i.i.d.

- ◆ Independent: no one instance affects any other
- ◆ Identically distributed: each instance has same probability



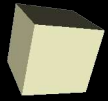
- Authors used iid Bernoulli extensively
- However, all results in the paper can be obtained with any regenerative traffic model





Asymptotic Analysis

| Notation | Definition | Mathematical definition |
|-----------------|--|--|
| $f = O(g)$ | asymptotic upper bound | $0 \leq \lim_{x \rightarrow \infty} \left \frac{f(x)}{g(x)} \right < \infty$ |
| $f = o(g)$ | asymptotically negligible | $\lim_{x \rightarrow \infty} \left \frac{f(x)}{g(x)} \right = 0$ |
| $f = \Omega(g)$ | asymptotic lower bound (iff $g = O(f)$) | $0 < \lim_{x \rightarrow \infty} \left \frac{f(x)}{g(x)} \right \leq \infty$ |
| $f = \omega(g)$ | asymptotically dominant (iff $g = o(f)$) | $\lim_{x \rightarrow \infty} \left \frac{f(x)}{g(x)} \right = \infty$ |
| $f = \Theta(g)$ | asymptotically tight bound (iff $f = O(g)$ and $f = \Omega(g)$) | $0 < \lim_{x \rightarrow \infty} \left \frac{f(x)}{g(x)} \right < \infty$ |



■ Example: $3x^2$

- ♦ $o(x^3)$

- ♦ $O(x^3)$

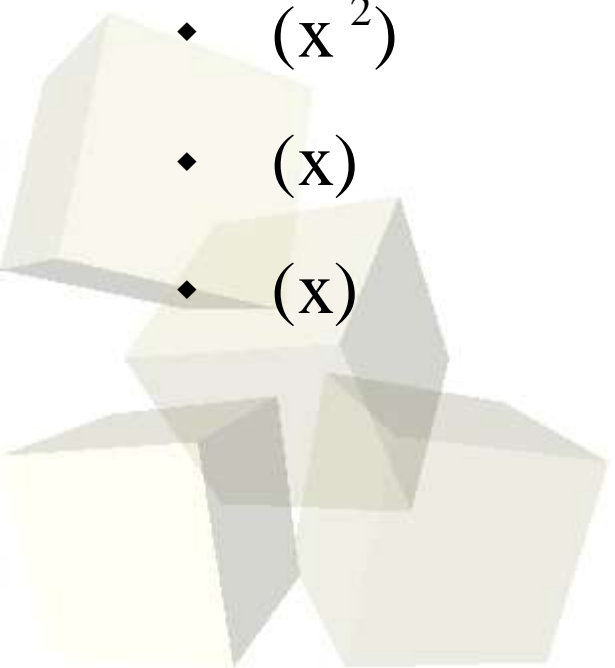
- ♦ $O(x^2)$

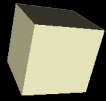
- ♦ (x^2)

- ♦ (x^2)

- ♦ (x)

- ♦ (x)





■ Maximum weight matching

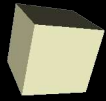
- ◆ McKeown, Anantharan, Walrand
- ◆ Stable; not implementable ($(n^3) / \text{time step}$)
- ◆ Complex algorithm

■ Other similar algorithms

- ◆ Tassiulas; Giaccone, Prabhakar, Shah
- ◆ Stable; still require centralized control

■ iSLIP (maximal matching)

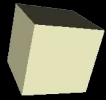
- ◆ Decentralized, constant number of cycles / time step
- ◆ Unstable



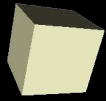
- Switch scheduling and Edge coloring are related
- Bipartite multigraph between input ports and output ports, where each edge represents one packet
- A correct edge coloring of this graph corresponds to a possible switch schedule
- Need a good edge coloring algorithm for bipartite multigraphs



- Algorithms with good number of colors
 - ◆ Centralized, (1) time / packet
- Distributed algorithms
 - ◆ Panconesi, Srinivasan
 - ◆ Offline, too many colors for stable operation
- Others
 - ◆ Still take (1) time / packet
- No prior algorithms can use few colors and operate in $O(1)$ time / packet



- Online randomized edge coloring
- Batch scheduling
- t_k is the time at which the k^{th} batch ends
 - ◆ During time $(t_{k-1}, t_k]$, incoming packets are colored.
 - ◆ Each color is then assigned a time slot in $(t_k, t_{k+1}]$, and all packets of a color are sent on that color's time slot.



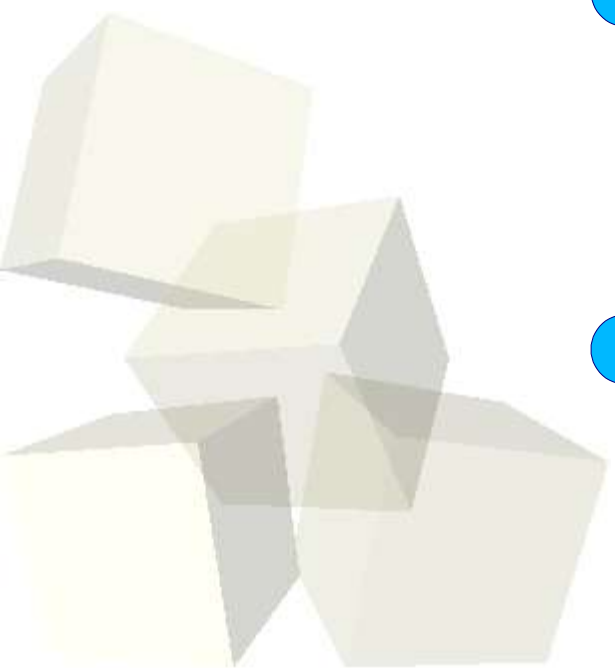
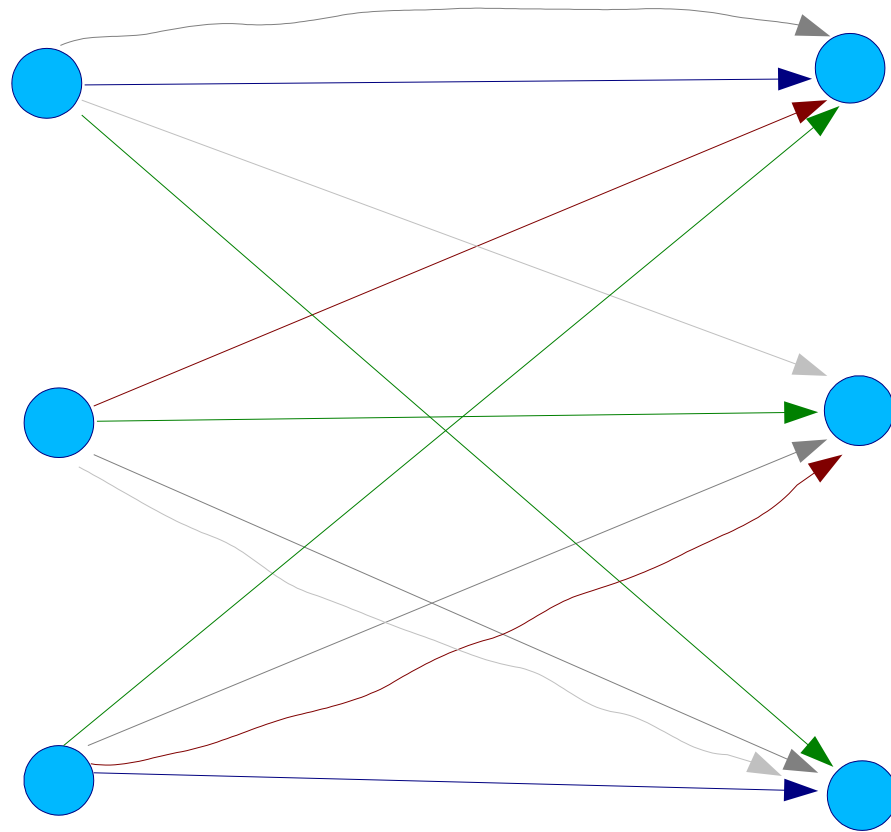
■ Edge coloring algorithm

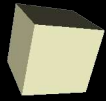
- ◆ Have initial pool of available colors
- ◆ As a new packet comes in (randomized process, if traffic is not already random enough), select a color at random from set of free colors for that pair of nodes
- ◆ If no free color is available, then assign that edge the minimum color from a fresh set of colors



New Algorithm

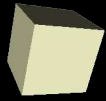
- Example: 4 initial colors, 4 time steps in batch





■ Theorem 1

- ♦ If we can color a graph with maximum degree Δ using at most $\Delta + o(\Delta)$ colors, then the switch scheduling algorithm is stable, i.e., the queue lengths are bounded
- ♦ This is true with high probability
 - Probability at least $1 - o(1 / (\Delta^2 - 2))$
- ♦ Proof involves applying a Markov chain result



■ Theorem 2

- ◆ Gives the result needed for Theorem 1
- ◆ Relies on having a dense bipartite multigraph where maximum degree = (n^2)
 - Reasonable for heavily loaded Internet routers
- ◆ Proof (involved) divides the stages into epochs to aid in the analysis
 - Bound the number of common colors between nodes i, j when adding edge (i, j)
 - Limit the number of edges at each node that have to use extra colors



■ Theorem 3

- ◆ Extends Theorem 2 to hold for general multigraphs (not just bipartite)
- ◆ Proof very similar

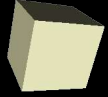
■ Theorem 4

- ◆ Relax the $= (n^2)$ condition
- ◆ Again, the proof is similar to Theorems 2 and 3





- Different kind of paper
- In theory, this paper presents a stable and implementable algorithm
- Good to have pencil and paper proof of ideas and concepts
- Not an actual implementation (or simulation)
- Many needed details left out



- Questions
- Thoughts
- Complaints

- Discussion

