ENSC 835: HIGH-PERFORMANCE NETWORKS

FINAL PROJECT PRESENTATIONS
Fall 2003

Simulating Search Strategies for Gnutella

Chun Wai Chan
http://www.sfu.ca/~cchany/ENSC835
cchany@sfu.ca

Roadmap

- Introduction: Motivation & Overview
- Overview of Related Work
- Technical Details
  - BFS
  - Randomized BFS
  - k-Walker Random Walk
  - Iterative Deepening (if time permits)
- Implementation & Discussion
  - Topology
  - Architecture
  - Results
- Future Work
- References

Introduction: Motivation & Overview

- This project is about simulating different search strategies in Gnutella.
- What is Gnutella?
  - An open source decentralized peer-to-peer (P2P) network
  - Allows users to search and download data from one another

Introduction: Motivation & Overview

- Why is it important?
  - Data-sharing P2P is popular
    - Lime Wire reported total number of users exceeds 500,000 in March 2002
    - Clip2 and Lime Wire have estimated with simultaneous users of about 40,000 (average) in 2001
  - Searches impose huge burden to the network

Overview of Related Work

- There are many researches for searches in P2P systems going on in major universities including:
  - Infrastructure for Resilient Internet Systems (IRIS) by MIT, and Stanford Peers in Stanford University
- Gnutella Protocol Development is active in renewing the Gnutella protocol
  - The newest proposed version is 0.7 while the current stable version is 0.4

Overview of Related Work

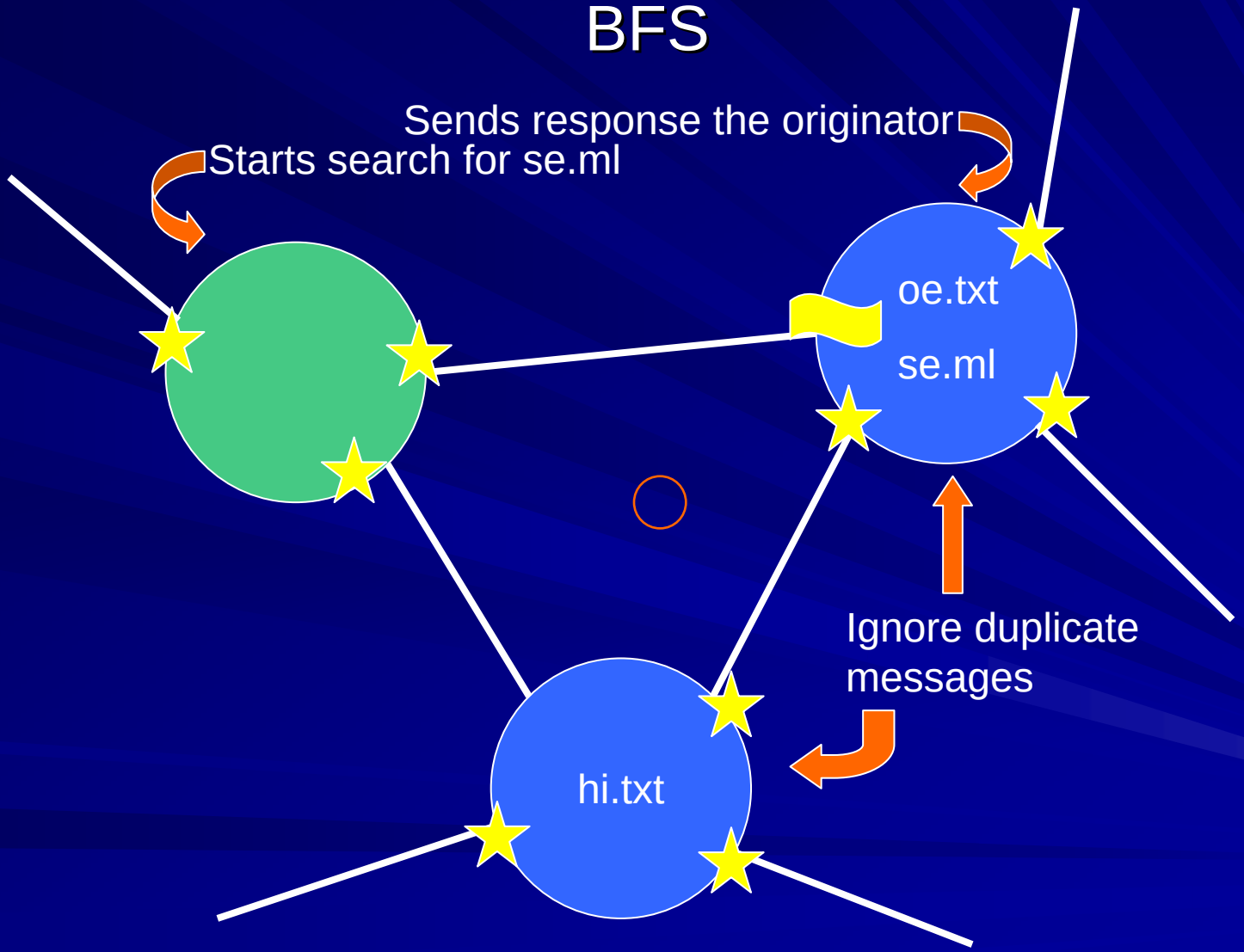
- Search techniques:
  - Gnutella protocol 0.4 uses breadth first search (BFS) as its search technique
  - Kalogeraki, et al. [2] has suggested the randomized BFS and the intelligent search
  - Lv, et al. [3] has proposed the k-walker random walk technique
  - Yang, et al. [5] has suggested iterative deepening search

Technical Details
BFS

- Node sends a search request to all neighbouring nodes
- Upon receiving search request,
  - It sends a response to the querying node if it contains the data
  - Then, it forwards the request to all its neighbours
- For duplicate requests, the node will drop that request and will not forward it
- The process continues until the preset number of hops (call it TTL) becomes 0

Technical Details

BFS



Technical Details

Randomized BFS

- Almost the same as BFS
- Except that instead of sending to all neighbours, send to a preset number of neighbours randomly
- Reduces network traffic while decreasing the number of data found

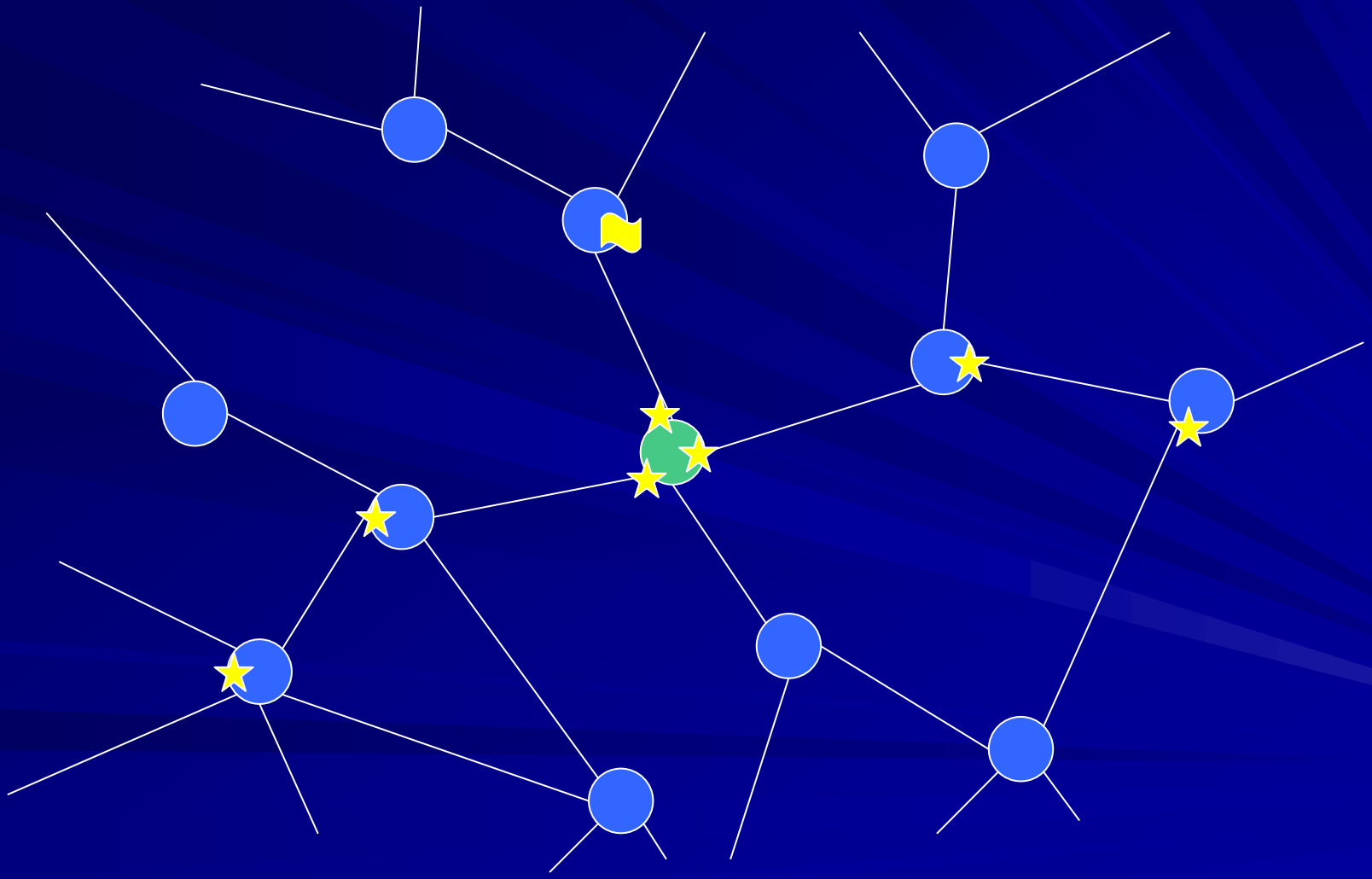
Technical Details

k-Walker Random Walk

- Search originator sends request to k randomly selected neighbours
- If a neighbour has the data, it response to the originator and stop there
- If not, it forwards to one randomly chosen neighbour only
- Idea is like k walkers searching on the network

Technical Details

k-Walker Random Walk (k = 3)



Technical Details
Iterative Deepening

- Perform BFS with TTL 1
- If successful, respond and finish the search
- If not, continue with TTL 2
- Repeat and increase the TTL again until a preset limit is reached

Implementation & Discussion

- Ns 2.26 will be used as the simulation tool
- Detailed instructions on setting up ns under cygwin by Nicolas Christin can be found in

  http://www.sims.berkeley.edu/~christin/ns-cygwin.shtml

Implementation & Discussion

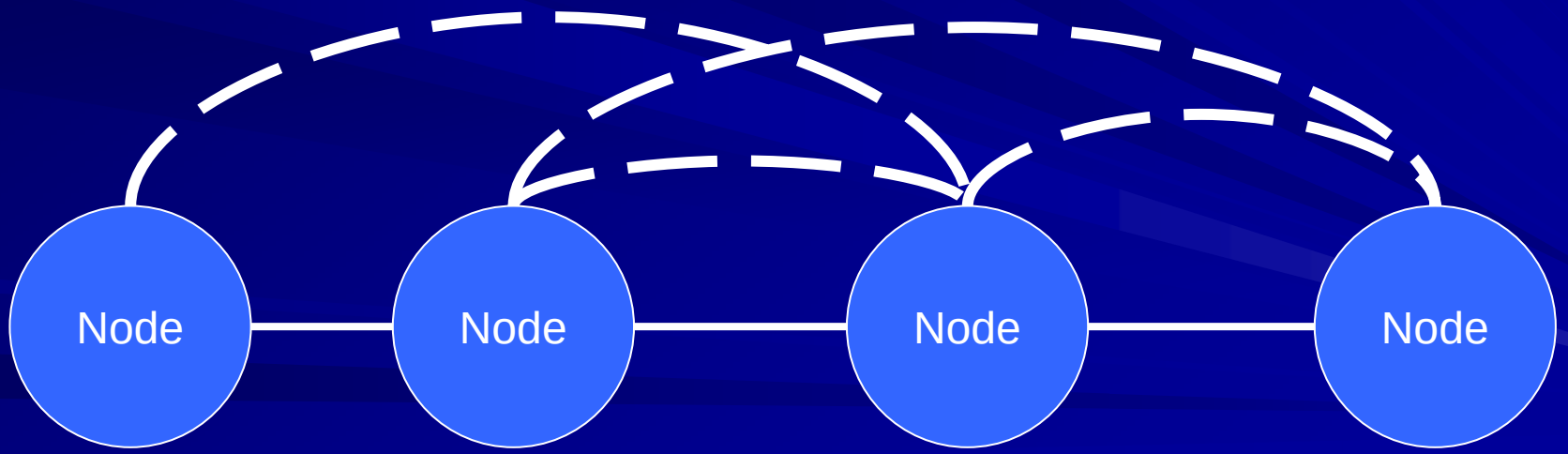
Simulation Scenario

1. Randomly generate the network topology, the data each node contains, and the search requests into a file
2. OTcl script reads the file and runs the simulation on a specific search strategy
3. Repeat step 2 for other search strategies
4. Collect and analyse the result

Implementation & Discussion

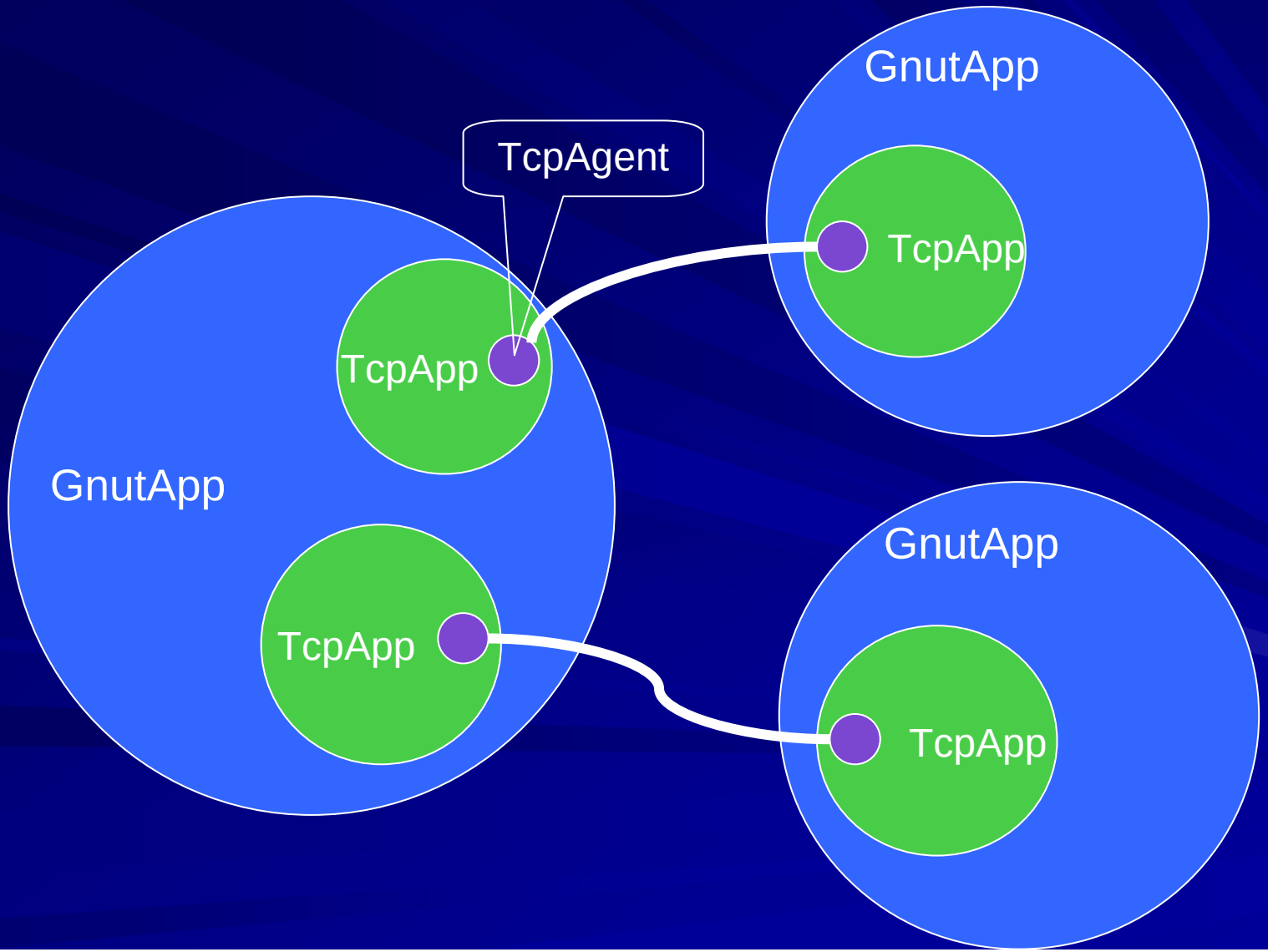
Topology generation

- A C++ program will be written to do step 1
- A perl script will be used to translate the result to OTcl script



Implementation & Discussion

Architecture



Implementation & Discussion

Results

- We specify:
  - The number of nodes, the number of links, the max number of data a node contains, the number of search queries to be initiated, and the size of a search message
- We are interested in:
  - Average number of success per search request
  - Average number of duplicate messages received per search
  - Average number of search messages generated per search
  - A graph of number of search messages vs. time

Future Work

- Run the simulation on different topologies
  - Modify Inet Topology Generator which approximate Internet AS topology for smaller number of nodes (currently only > 3037)

    http://topology.eecs.umich.edu/inet/
  - Port Georgia Tech Internetwork Topology Models topology generator which generates different types of random graphs to Linux

    http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html
- Run the simulation on more powerful machines
- Simulation Gnutella 0.6's search on its hierarchical topology (with UltraPeers)

References

- [1] Protocol Specification: Clip2 (n.d.). The Gnutella Protocol Specification v0.4. Retrieved January 1, 2003, from http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf
- [2] Series Proceeding Section Article: Kalogeraki, V., Gunopulos, D., & Zeinalipour-Yazti, D. (2002, November). A Local Search Mechanism for Peer-to-Peer Networks [Electronic version]. Series-Proceeding-Section-Article, 300-307.
- [3] Conference Paper: Lv, Q., et al. (2002, June). Search and Replication in Unstructured Peer-to-Peer Networks. Paper presented at the 16th International Conference on Supercomputing. New York City, NY.
- [4] Online Article: Ritter, J. (2001, February). Why Gnutella Can't Scale. No, Really. Retrieved January 1, 2003, from http://www.darkridge.com/~jpr5/doc/gnutella.html

References

- [5] Conference Paper: Yang, B., & Garcia-Molina H. (2002, July). Improving Search in Peer-to-Peer Networks. Paper presented at the 22nd International Conference on Distributed Computing Systems. Vienna, Austria.
- [6] Web Page: Gnutella Protocol Development (n.d). Retrieved October 30, 2003, from http://rfc-gnutella.sourceforge.net/index.html
- [7] Web Page: IRIS: Infrastructure for Resilient Internet Systems. (n.d.). Retrieved January 2, 2003, from http://iris.lcs.mit.edu
- [8] Web Page: Lime Wire (n.d). Retrieved November 11, 2003, from http://www.limewire.com
- [9] Web Page: Stanford Peers. (n.d.). Retrieved January 2, 2003, from http://www-db.stanford.edu/peers
- [10] Web Page: Clip2 (n.d). Retrieved January 1, 2003, from http://www.clip2.com