Clustering Superpeers in P2P Networks by Growing Neural Gas

Mihai Dumitrescu$^{1,3}$ and Răzvan Andonie$^{1,2}$

$^1$Electronics and Computers Department  
Transylvania University of Braşov, Romania  
$^2$Computer Science Department  
Central Washington University, Ellensburg, USA  
$^3$rosoftlab s.r.l., Brasov, Romania

15.02.2012
Outline

Introduction and Motivation
  Motivation
  Related Work
  Growing Neural Gas
  Challenges

Algorithm
  Model and Messages
  Peer stimulation and Superpeer Reaction

Experimental results
  Setup
  Results
  Comparison to SG-2

Conclusion and Outlook
Outline

Introduction and Motivation
- Motivation
- Related Work
- Growing Neural Gas
- Challenges

Algorithm
- Model and Messages
- Peer stimulation and Superpeer Reaction

Experimental results
- Setup
- Results
- Comparison to SG-2

Conclusion and Outlook
Motivation

- Why structure P2P networks?
  - Efficiency (time and space)
  - Resilience
  - Better use of network topology

- Why use superpeers?
  - Way of creating a structured overlay
  - General approach
  - Adjustable balance between structured / unstructured

- Why use GNG?
  - Growing Neural Gas (GNG) learns topologies incrementally
  - GNG input space and P2P networks have similarities
Related Work (excerpt)

- Building blocks & inspiration
  - Wojciech Galuba and Karl Aberer, “ProtoPeer: A P2P Toolkit Bridging the Gap Between Simulation and Live Deployment"

- Addressing a similar problem (SG-2)
Growing Neural Gas I

Original Version

- Used to estimate the topology of an input space consisting of \( n \)-dimensional vectors
- Works with a set of model vectors in \( w \in \mathbb{R}^n \)

**Figure:** [http://wwwold.ini.rub.de/VDM/research/gsn/JavaPaper/node19.html](http://wwwold.ini.rub.de/VDM/research/gsn/JavaPaper/node19.html)
Growing Neural Gas II

Algorithm loop (outline)

- For each new input vector find closest (s1) and second-closest (s2) model vectors
- Increment age of edges from s1
- Increment accumulated error of s1
- Move s1 and its neighbors towards input vector
- Create edge between s1 and s2 or reset its age
- Remove old edges and unconnected model vectors
- Insert new model vector between the model vector with the maximal global error and its neighbor with maximal error, decrease error of the model vectors.
- Decrease error variables of all nodes
Challenges

Make GNG work in a decentralized way, on a P2P network.

Some challenges include:

- Discrete location of "model vectors"
- Lack of global view
Outline

Introduction and Motivation
  Motivation
  Related Work
  Growing Neural Gas
  Challenges

Algorithm
  Model and Messages
  Peer stimulation and Superpeer Reaction

Experimental results
  Setup
  Results
  Comparison to SG-2

Conclusion and Outlook
Model

Graph $G(P, E)$

- set of peers $P$, any element of which can become a superpeer
- set of edges $E$

Superpeers $S \subseteq P$, each having

- *error*, the total accumulated error of the superpeer
- *fingerList* the set of overlay connections to other superpeers, each storing its *age*
Messages

Broadcasted messages

- **FindSuperpeer** locate one or more superpeers
- **CreateSuperpeerInTheMiddle** create a new superpeer half the way to the target
- **UpdateSuperpeerLocation** announce new location of superpeer

Directed messages

- **UpdateSuperpeer** announce changes related to a superpeer
- **MoveSuperpeer** move a superpeer and acknowledge its movement
- **CreateSuperpeer** make a peer become a superpeer
Peer stimulation

Once stimulated, a peer sends out

\textit{FindSuperpeer}

and waits for answers from \(S_1\), the closest, and \(S_2\), second-closest superpeers.

The peer ignores subsequent answers and sends

\textit{UpdateSuperpeer}(S_1, S_2, rt_{S_1}, rt_{S_2})

to \(S_1\).
Superpeer reaction I

\[
\begin{align*}
\text{newError} & \leftarrow rt_{S1} \ast \text{errorIncrementFactor} \\
S.\text{error} & \leftarrow S.\text{error} + \text{newError} \\
\text{for all } f & \in S.\text{fingerList} \text{ do} \\
& \quad f.\text{age} \leftarrow f.\text{age} + 1 \\
& \quad \text{if } f.\text{age} \geq \text{fingerDeleteThreshold} \text{ then} \\
& \quad \quad S.\text{fingerList}.\text{remove}(f) \\
& \quad \text{end if} \\
\text{end for} \\
\text{if } S2 & \in S.\text{fingerList} \text{ then} \\
& \quad S.\text{getFingerTo}(S2).\text{age} \leftarrow 0 \\
\text{else} \\
& \quad S.\text{fingerList} \leftarrow S.\text{fingerList} + S2 \\
\text{end if}
\end{align*}
\]
Superpeer reaction II

\[
\text{if } \text{newError} \geq \text{moveThreshold} \text{ then}
\]
\[
\quad S.\text{send}(\text{MoveSuperpeer}, P')
\]
\[
\quad S.\text{error} \leftarrow S.\text{error} + \text{movePunishment}
\]
\[
\text{end if}
\]
\[
S_\text{max} \leftarrow \text{fingerList.findMaxErrorNeighbor()}
\]
\[
\text{if } S.\text{error} \geq \text{createNewThreshold} \text{ then}
\]
\[
\quad S.\text{send}(\text{CreateSuperpeerInTheMiddle}, S_\text{max})
\]
\[
\quad S.\text{error} \leftarrow 0
\]
\[
\text{end if}
\]
Superpeer movement and creation

Superpeer movement based on threshold

- Peers receiving `MoveSuperpeer` become superpeers and send acknowledge.
- Superpeers receiving acknowledge of `MoveSuperpeer` become normal peers.
- Fresh superpeers send out `UpdateSuperpeerLocation`.

Superpeer creation based on threshold

- Peers receiving `CreateSuperpeerInTheMiddle` locate the peer at half the way and send `CreateSuperpeer` to it.
Superpeer removal

Overlay connections exceeding the age $\textit{fingerDeleteThreshold}$ are unused and thus removed.

When $\textit{fingerList}$ is empty, a superpeer becomes a peer.

$\textit{fingerDeleteThreshold}$ directly influences size and granularity of overlay network.
Algorithm parameters

- **moveThreshold**: If the `newError` of a superpeer exceeds this, then the superpeer is moved towards the requesting peer.
- **createNewThreshold**: If the total `error` of a superpeer exceeds this, then a new superpeer is created.
- **fingerDeleteThreshold**: Fingers older than this are deleted.
- **minHopsToInsertSuperpeer**: Minimum distance between superpeers, so a new superpeer can be inserted.
- **attractionProbability**: To avoid agglomeration of peers, a newly created peer sends out attraction messages with this probability.
Complexity

- Number of \textit{FindSuperpeer} messages grows polynomially with network size and does not depend on diameter.
- Number of maintenance messages independent of network diameter in a stable system state.
- \textit{FindSuperpeer} messages can be piggybacked to useful messages. Their amount needs to be proportional to the load originating from that peer.
Outline

Introduction and Motivation
  Motivation
  Related Work
Growing Neural Gas
Challenges

Algorithm
  Model and Messages
Peer stimulation and Superpeer Reaction

Experimental results
  Setup
  Results
  Comparison to SG-2

Conclusion and Outlook
Experimental setup

Protopeer

- Simulation and live deployment in one framework
- Many delay and loss models for communication channels
- Customizable network topology

Our customizations

- Power-law topology generator
- Multi-threaded event scheduler with idle-capability
- Interactive network viewer using JGraph
Network Viewer
Network Load & Measurements

- Continuous load on each peer
- Measurements
  - Number of peers in the network $n$
  - Number of superpeers $n_{sp}$
  - Number of overlay edges
  - Average age of overlay edges
  - Average distance of a peer to the closest superpeer $d_{sp}$
Evolution - power law network

Without / with churn
Evolution - random network

Average degree 2, 10
Problems

- Discrete superpeer locations: We use a local threshold to decide when to move superpeers. Discrete locations actually help avoiding overcrowding a network region.
- Lack of global view: Even if superpeers are created locally, they move towards the right positions in subsequent steps.
- Too many messages in the beginning: Use $ttl_F$ on the $FindSuperpeer$ messages.
- Superpeers tend to crowd up: Use superpeer attraction messages.
- Superpeers move very often: Adjust $moveThreshold$. 
Comparison to SG-2

- Quality metrics:
  - SG-2: Number of peers satisfying peer latency condition
  - Our algorithm: Average distance to superpeer
- Comparison: Time to stabilization (first order derivative)
- Conclusion: Comparable convergence speed
Outline

Introduction and Motivation
  Motivation
  Related Work
  Growing Neural Gas
  Challenges

Algorithm
  Model and Messages
  Peer stimulation and Superpeer Reaction

Experimental results
  Setup
  Results
  Comparison to SG-2

Conclusion and Outlook
Conclusion and Outlook

Adaptive and completely decentralized superpeer overlay management algorithm.

Outlook
- Analytical study
- PlanetLab implementation
- Self-optimize algorithm parameters
- Gossiping scheme for global error maximum
- Consider node capacities