

# Routing

A Routing algorithm tells each intermediate node on the route which outgoing edge the message should be sent, depending on the destination.

Cost of routing (messages) Vs. Size of routing table (memory space).

Cost of routing is measured by *stretch factor* :

$$\max_{u,v} \frac{\text{cost}(v,u)}{\text{dist}(u,v)}.$$

# Compact Routing

Two extreme approaches:

1. Shortest path routing: stretch factor is 1. Memory:  $O(n \log n)$  bits per node.
2. Flooding: stretch is  $O(m)$ . No memory.

Name-independent vs. name-based routing.

Tradeoff between stretch and memory requirements.

**Theorem 1.** *There exists a routing algorithm with stretch factor  $O(k)$  which uses an average of  $O(n^{1/k} \log^2 n)$  bits per node, for any value of  $k \geq 1$ .*

## Cover: Definition

Consider a network  $G = (V, E)$ .

Let  $S \subseteq V$ .  $S$  is a **cluster** in  $G$  if the subgraph induced by  $S$  is connected.

The size of a cluster  $S$  — number of nodes in  $S$ .

The diameter of a cluster —  $diam(S)$  is the diameter of this induced subgraph (**strong diameter**).

A **cover** is a collection of clusters  $C = \{S_1, S_2, \dots, S_k\}$  such that the  $\cup_{i=1}^k S_i = V$ .

$$vol(C) = \sum_{i=1}^k |S_i| \geq n.$$

Cover parameters:

**Sparsity:**  $vol(C)/n$  is the average number of occurrences of any node in all clusters of  $C$  (*average degree of cover*).

Determines the cost of the algorithm using the cover.

2. **Locality:**  $diam(C)$  — maximum diameter of a cluster in  $C$ .

Low diameter — faster running time.

Goal: Construct covers with low volume and diameter.

Tradeoff between diameter and volume.

# Neighborhood Cover

Given a subset of vertices  $W \subseteq V$ , the  $\rho$ -neighborhood cover of  $W$  is the collection of  $\rho$ -neighborhoods of the vertices of  $W$ , denoted by

$$\Gamma_\rho(W) = \{\Gamma_\rho(v) \mid v \in W\}.$$

Typically  $W = V$  in many applications and  $\rho$  will be chosen appropriately.

Goal: To find Neighborhood covers with low volume and diameter.

# Coarsening

A Cover is *coarser* than another cover if its clusters contain the clusters of the other cover.

Given two collections of clusters,  $C = \{S_1, \dots, S_h\}$  and  $C' = \{S'_1, \dots, S'_h\}$ ,  $C'$  is a coarsening of  $C$  if for every  $S_i \in C$ , there exists  $S'_j \in C'$  such that  $S_i \subseteq S'_j$ .

# A Cover Construction Algorithm

**Input:** Cover  $S$ ; **Output:** Cover  $T$ .

$T = \phi$ .

While  $S \neq \phi$  do

    select an arbitrary cluster  $s_0 \in S$ .

$z = s_0$ .

    Repeat (merge cluster layers around  $z$ )

$y = z$  ( $y$  is kernel of resulting cluster  $z$ )

$Z = \{s \mid s \in S, s \cap y \neq \phi\}$

$z = \cup_{s \in Z} S$

    until  $|z| \leq n^{1/k}|y|$

$S = S - Z$

$T = T \cup \{Z\}$ .

Endwhile

Output( $T$ ).

## Analysis

**Theorem 2.** *Given a weighted graph  $G = (V, E, w)$ ,  $|V| = n$ , a cover  $S$  and an integer  $k \geq 1$ , the algorithm constructs a coarsening cover  $T$  of  $S$  that satisfies the following properties:*

1.  $Rad(T) \leq (2K + 1)Rad(S)$ . ( $Diam(T) \leq (k + 1/2)Diam(S)$ ).

2.  $vol(T) \leq n^{1+1/k}$  (i.e., the ave. degree  $\leq n^{1/k}$ )

**Proof:** Part 2: Consider the collection  $C$  of kernels  $y$  corresponding to the clusters  $Z$  generated by the algo.

The kernels in  $C$  are mutually disjoint.

From the termination condition,  $|z| \leq n^{1/k}|y|$ .

Thus,  $\sum_z |z| \leq \sum_y n^{1/k}|y| \leq n^{1/k}n = n^{1+1/k}$ .

Part 1: Consider some iteration of the while loop, starting with some cluster  $s \in S$ .

Let  $J$  denote the number of times the repeat loop is executed.

Denote the initial sets  $Z$  and  $z$  by  $Z_0$  and  $z_0$  respectively.

In the  $i$ th iteration of the repeat loop, the sets constructed are  $Z_{i-1}$ ,  $z_i$  and  $y_i$ .

For  $1 \leq i \leq J$ ,  $y_i = z_{i-1}$ .

Claim 1:  $|z_i| \geq n^{i/k}$ , for every  $0 \leq i \leq J - 1$ , and strict inequality holds for  $i \geq 1$ .

Proof by induction on  $i$ . Trivial for  $i = 0$ .

Now,  $|z_i| \geq n^{1/k} |z_{i-1}| \geq n^{1/k} n^{(i-1)/k}$ .

The first inequality follows from the fact that termination condition of the repeat loop was not met.

Claim 2: For every  $1 \leq i \leq J$ ,  $Rad(y_i) \leq (2i - 1)Rad(S)$ .

Proof by induction on  $i$ .

Base case trivial (since  $Y_1 = s \in S$ ).

Induction step:  $Rad(y_i) = Rad(z_{i-1}) \leq Rad(y_{i-1}) + 2Rad(S)$ .

Since  $J \leq k$ ,  $Rad(y_j) \leq (2k - 1)Rad(S)$ .

Hence  $Rad(z_j) \leq (2k + 1)Rad(S)$ .

# Application to Compact Routing

Name-independent routing scheme.

Hierarchy of routing algorithms, each restricted to a region of certain diameter.

A **regional**  $(s, d)$ -**routing algorithm** provides  $O(s)$  stretch inside regions of diameter  $d$ :

For any  $v, u$ , any message sent from  $v$  to  $u$  is delivered if  $\text{dist}(v, u) \leq d$ ; otherwise  $v$  is notified.

In both cases, only  $O(s.d)$  messages sent.

Let  $R_i$  be the regional  $(s, 2^i)$  routing algorithm.

We use  $R_1, \dots, R_{\log D}$  algorithms concurrently ( $D = \text{Diam}(G)$ ).

Try  $R_1, \dots$  till successful.

Number of messages is  $\sum_{i=0}^{\log d} s2^i = O(sd)$ .

# Regional Routing Algorithm

A Regional  $(s, d)$  routing algorithm with  $O(s)$  stretch factor and  $O(n^{1/s} \log n)$  average memory per node.

Idea: Use sparse covers.

Start with a  $d$ -neighborhood cover  $D$  and coarsen it to get a cover  $D'$  such that:

1.  $diam(D') \leq (s + 1/2)diam(D) \leq (s + 1/2)2d$ .
2.  $vol(D') \leq n^{1+1/s}$ .

For each node  $v$ , designate one cluster containing the  $d$ -neighborhood of  $v$  to be the home cluster of  $v$ . (Ignore non-home clusters in  $D'$ .)

For each home cluster in  $D'$  choose some node as root and construct the shortest paths tree from the root to all other nodes in the cluster — **cluster tree**.

## DFS routing

Inside the cluster, routing is done based on DFS numbering of the cluster tree.

Each node records its own dfs number —  $dfs(v)$ , and the dfs numbers of its children in the cluster tree.

The root maintains a table with the dfs number of each node  $v$  in the cluster.

To send a message to  $u$ ,  $v$  sends a message to the cluster root.

root either notifies failure, or routes by sending message down the tree.

At most  $2\text{diam}(D')$  messages are sent. Hence stretch is  $O(s.d)$ .

Total memory bits in a cluster is number of nodes in the cluster times  $\log n$ .

Hence total memory in all nodes is  $O(\text{vol}(D') \log n) = O(n^{1+1/s} \log n)$  bits.

Total memory overhead of all routing schemes is  $O(n^{1+1/s} \log^2 n)$  bits.

**Theorem 3.** *There exists a routing algorithm with stretch factor  $O(k)$  which uses an average of  $O(n^{1/k} \log^2 n)$  bits per node, for any value of  $k \geq 1$ .*