How to use spanning trees to navigate in Graphs

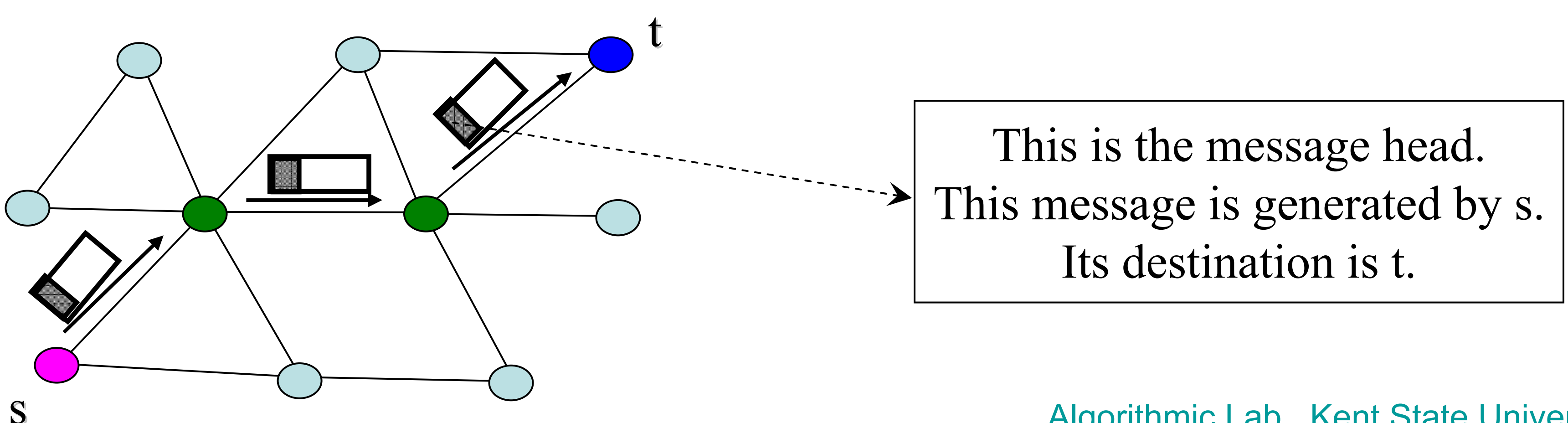
Feodor F. Dragan and Yang Xiang

(MFCS 2009)

Kent State University, USA

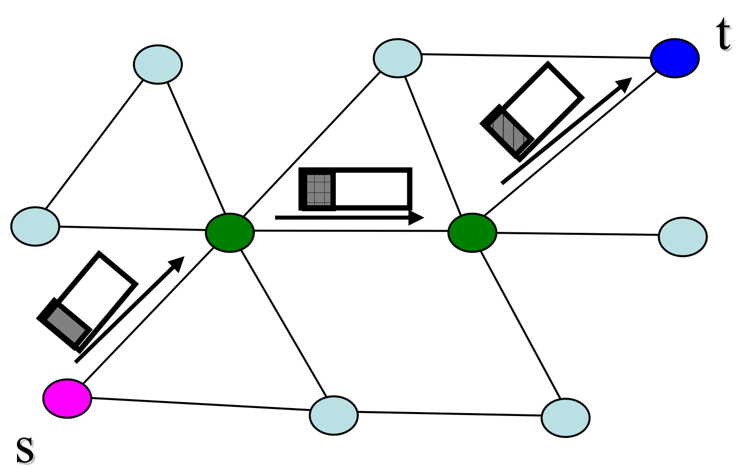
Routing in networks

- A routing scheme is a mechanism to deliver a package/message from one node to another node in the network
- Each node knows some information about the network, stored locally.
- Package has a head with information about the destination (its address, some other useful information)
- A node having a package needs to decide if the package reached the destination and, if not, to which neighbor to forward the package.
- To make this decision the current node uses own information stored locally, information from head of the package, and perhaps local information stored at neighbors.
- The purpose of routing is to let message generated by the source node reach the destination node.



Parameters of a Routing Scheme

- What information and how much is stored locally at each node.
- What information and how much is stored in the head of a message.
- How current node decides to choose a neighbor to forward the message towards the destination



- Each routing scheme generates a routing path for a given source and a given destination.

Stretch factors of a Routing Scheme

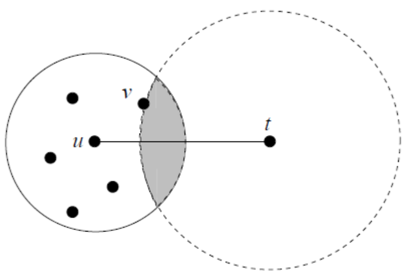
- Each routing scheme generates a routing path *R(x,y)* for a given source *x* and a given destination *y*.
- The goodness of a routing scheme is measured by how much routing paths differ from shortest paths.

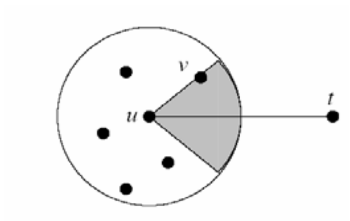
1. Additive max stretch factor: $$\max_{x,y \in V} \{R(x,y) - d(x,y)\}$$
2. Additive average stretch factor: $$\frac{1}{n^2} \sum_{x,y \in V} (R(x,y) - d(x,y))$$
3. Multiplicative max stretch factor: $$\max_{x,y \in V, x \neq y} \{R(x,y) / d(x,y)\}$$
4. Multiplicative average stretch factor: $$\frac{1}{n(n-1)} \sum_{x,y \in V, x \neq y} (R(x,y) / d(x,y))$$

Know in wireless networks Greedy Routing and Routing with Guaranteed Delivery

- Nodes know their Euclidean coordinates

- Two typical greedy routing schemes:
  - geographic routing
    - a neighbor which is geographically closest to destination is chosen.
  - Compass routing
    - a neighbor *v* such that angle *vut* is smallest is chosen.
  - Both strategies *do not* guarantee delivery



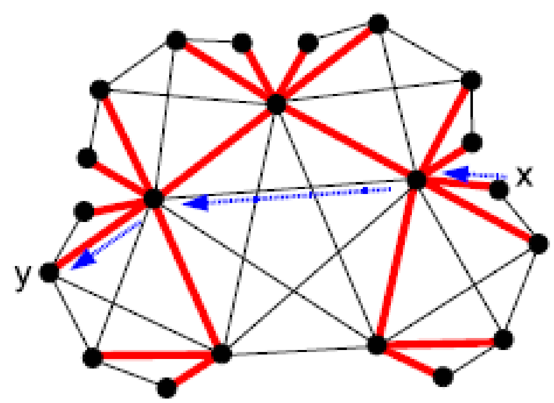


- Routing that guarantees delivery
  - Planar connection graph is constructed
  - Face routing is by traversing faces towards destination

Our basic idea

- Greedy routing with the aid of a spanning tree information.
- Assign pseudo-coordinates to nodes using a spanning tree.
- Possible information obtained from a spanning tree and stored locally at a vertex:
  - Distance Labels (*O(log n)* digits) or
  - Ancestry information, i.e., DFS-intervals (2 digits)

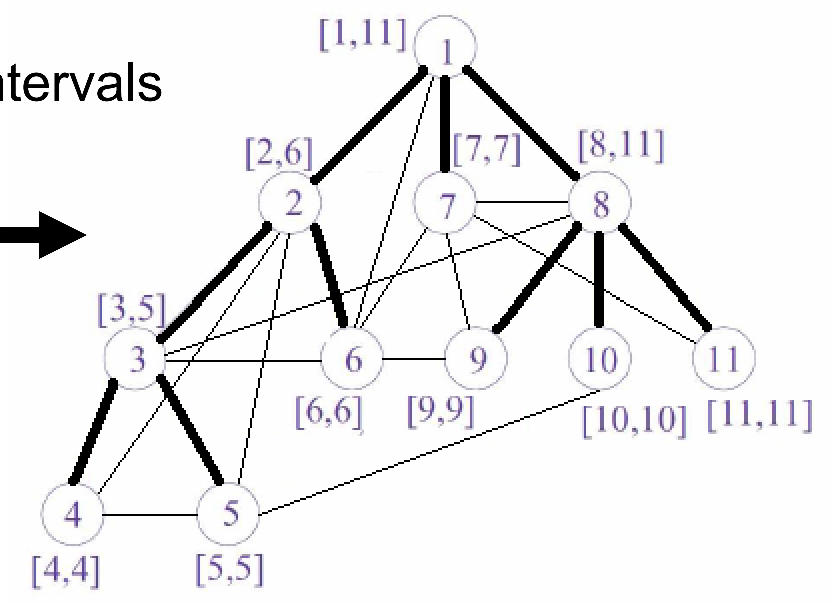
According to [Peleg'99], to each vertex of a n-vertex tree can be assigned O(log n) digits such that the distance in tree between any two vertices x and y can be computed in constant time by merely inspecting the digits assigned to x and y. Each digit has at most log n bits. These digits will become local information stored at each vertex as well as the address of each vertex put in the head of a message.



Why we use a spanning tree?

- Easy to construct and maintain.
- There exist distributed, self-stabilized algorithms for constructing spanning trees.
- Not hard to calculate the tree distance between two vertices
  - in constant time using only O(log n) digits per vertex [Peleg'99]
- Easy to tell the ancestry relationship between any two vertices.
  - In constant time using only DFS-intervals

Given a graph,
- build a spanning tree,
- root the tree at a node,
- find DFS-intervals in that tree
- assign to each node of graph its DFS-interval obtained



TDGR strategy (previous results)

Feodor F. Dragan, Martín Matamala: Navigating in a Graph by Aid of Its Spanning Tree. ISAAC 2008: 788-799

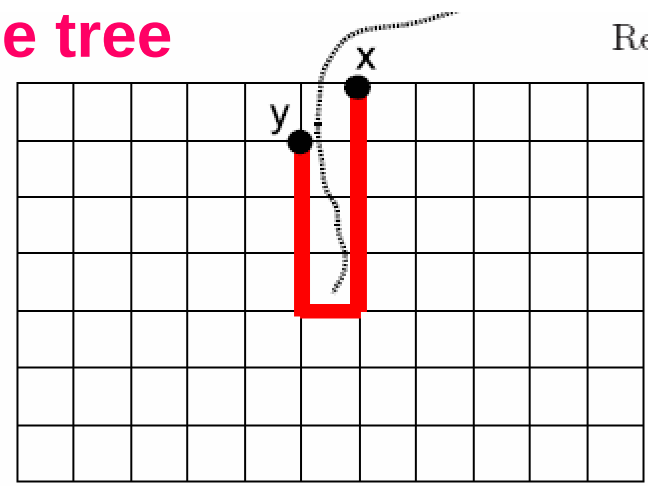
TDGR (**T**ree **D**istance **G**reedy **R**outing) strategy: *from a current vertex z (initially $z = x$), unless $z = y$, go to a neighbor of z in G that is closest to y in T.*

Additive r-carcass:

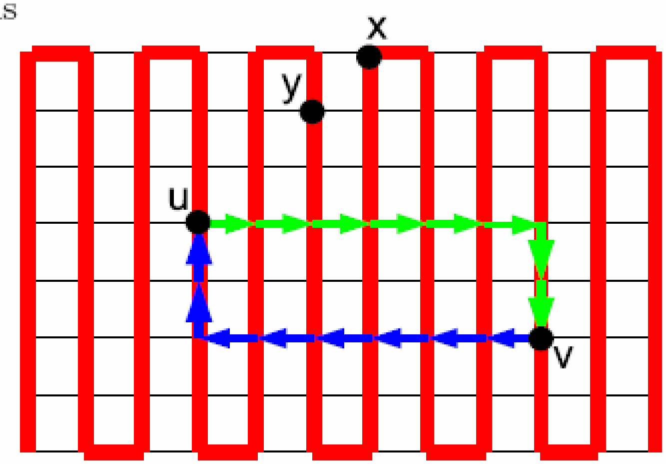
a spanning tree T of a graph G is an *additive r-carcass* for G if $g_{G,T}(x,y) \leq d_G(x,y) + r$ for each ordered pair $x, y \in V$

Each vertex knows its neighbors and O(logn) digits from the tree

Rectilinear grids



do not admit any (additive or multiplicative) tree r-spanners with a constant r



have additive 0-carcasses

IGR strategy (new)

IGR (**I**nterval **G**reedy **R**outing) strategy.

To advance in G from a vertex x towards a target vertex y ($y \neq x$), do:

if there is a neighbor w of x in G such that $y \in I_w$ (i.e., $w \in sTy$),

then go to such a neighbor with smallest (by inclusion) interval;

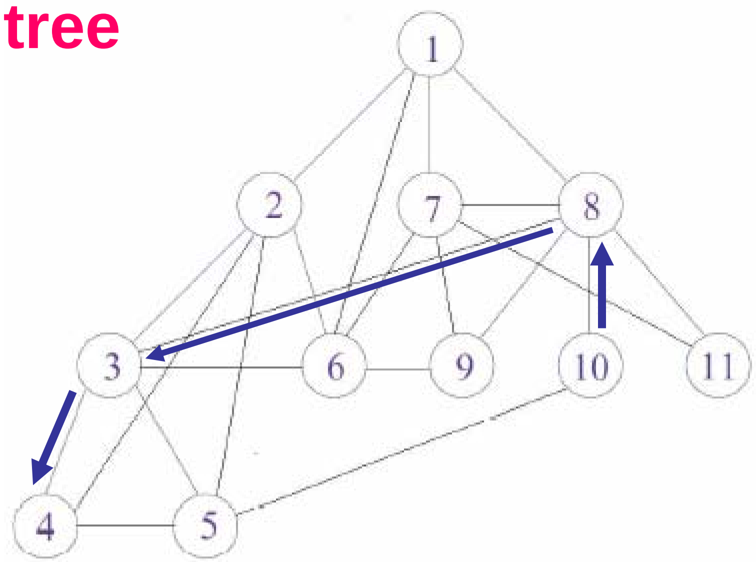
else (which means $x \notin N_G[sTy]$),

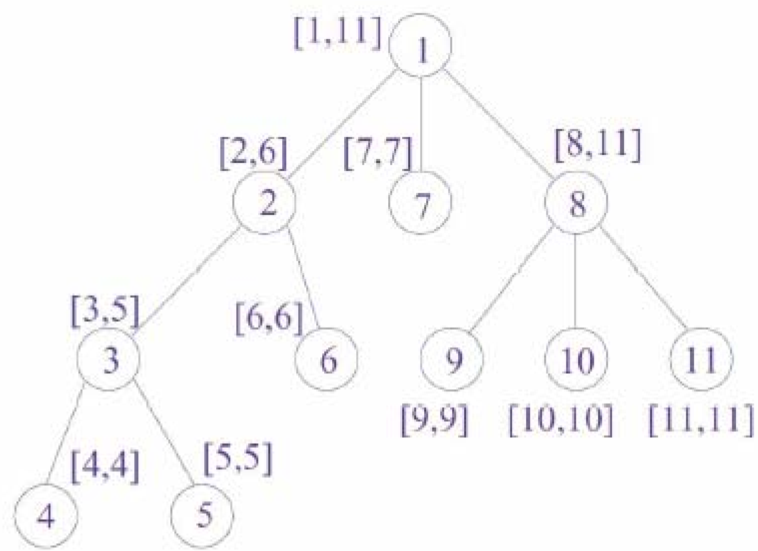
go to a neighbor w of x in G such that $x \in I_w$ and I_w is largest such interval.

Additive r-frame:

a spanning tree T of a graph G is an *additive r-frame* for G if $g_{G,T}(x,y) \leq d_G(x,y) + r$ for each ordered pair $x, y \in V$

Each vertex knows its neighbors and 2 digits from the tree



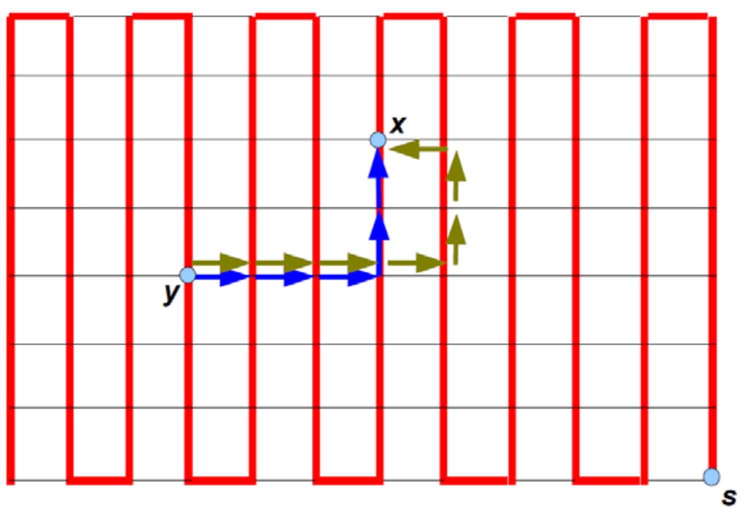


Carcass vs Frame

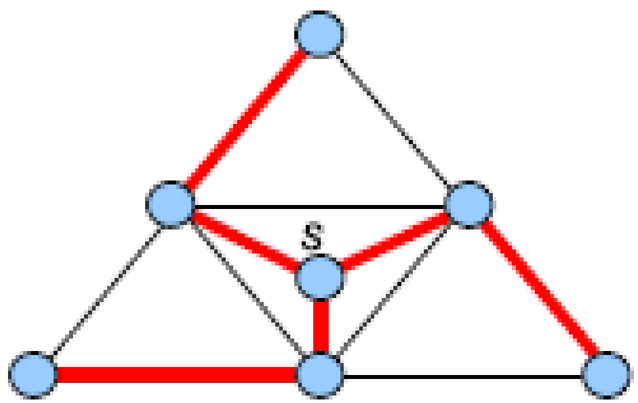
O(log n) digits stored per vertex

2 digits stored per vertex

Different criteria to move



The column-wise Hamiltonian path for the rectilinear grid is a 0-carcass but not a 0-frame.



A chordal graph with an additive 0-frame having no 0-carcass.

k-Localized strategies

- Now, we assume that each vertex can gather information from vertices at distance at most k in G.
- k=1 is the previous case.

k-**localized TDGR** strategy.

To advance in G from a vertex x towards a target vertex y, do:

go, using a shortest path in G, to a vertex $w \in D_k(x, G)$ that is closest to y in T.

k-**localized IGR** strategy.

To advance in G from a vertex x towards a target vertex y, do:

if there is a vertex $w \in D_k(x, G)$ such that $y \in I_w$ (i.e., $w \in sTy$),

then go, using a shortest path in G, to such a vertex w

with smallest (by inclusion) interval;

else (which means $d_G(x, sTy) > k$),

go, using a shortest path in G, to a vertex $w \in D_k(x, G)$ such

that $x \in I_w$ and I_w is largest such interval.

Our Results

- We proved that greedy routing with the aid of a spanning tree can produce near optimal paths in some special graph families, such as:
  - Rectilinear Grids
  - Graphs admitting locally-connected spanning trees (i.e., dually chordal graphs, strongly chordal graphs, interval graphs)
  - k-chordal graphs
  - chordal bipartite graphs
  - AT-free graphs
  - Tree-length λ graphs
  - δ-hyperbolic graphs
- In the following we highlight some interesting results.

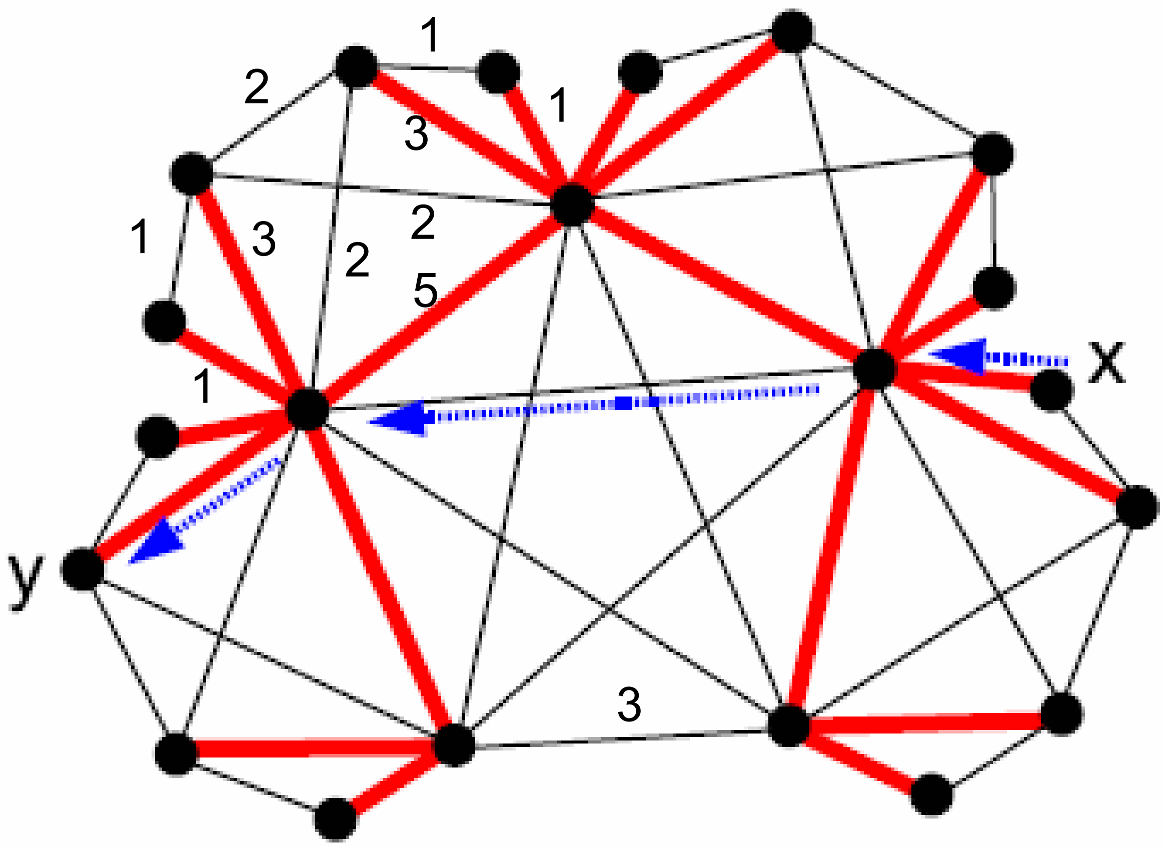
Graphs admitting locally connected spanning trees

Definition: We say that a spanning tree T of G is locally connected if the closed neighborhood $N_G[v]$ of any vertex v of G induces a subtree in T (i.e., $T \cap N_G[v]$ is a connected subgraph of T.)

Theorem: If T is a locally connected spanning tree of a graph G, then T is an additive 0-carcass and an additive 0-frame of G.

Known fact: Dually chordal graphs are exactly the graphs having locally connected spanning trees

How to find a locally connected spanning tree in a Dually Chordal graph



locally connected spanning tree

Definition:

A dually chordal graph is the intersection graph of the maximal cliques of a chordal graph.

- w(e) := # of triangles edge e belongs to
- Find maximum weight spanning tree

Navigating in a Dually Chordal Graph

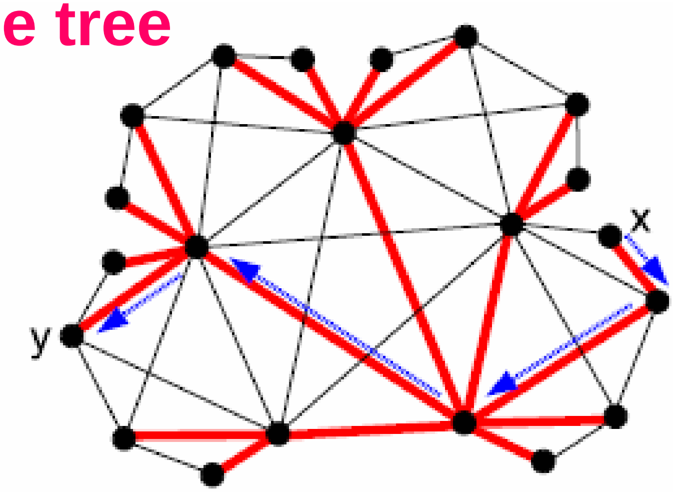
Dually chordal graphs are exactly the graphs having locally connected spanning trees

TDGR (**T**ree **D**istance **G**reedy **R**outing) strategy: *from a current vertex z (initially $z = x$), unless $z = y$, go to a neighbor of z in G that is closest to y in T.*

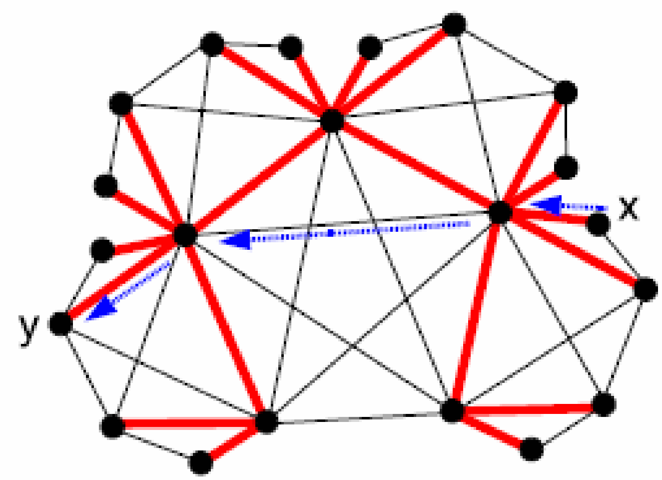
Additive r-carcass:

a spanning tree T of a graph G is an *additive r-carcass* for G if $g_{G,T}(x,y) \leq d_G(x,y) + r$ for each ordered pair $x, y \in V$

Each vertex knows its neighbors and O(logn) digits from the tree



additive tree 3-spanner



locally connected spanning tree

Navigating in a Dually Chordal Graph

Dually chordal graphs are exactly the graphs having locally connected spanning trees

IGR (**I**nterval **G**reedy **R**outing) strategy.

To advance in G from a vertex x towards a target vertex y $(y \neq x)$, do:

if there is a neighbor w of x in G such that $y \in I_w$ (i.e., $w \in sTy$),

then go to such a neighbor with smallest (by inclusion) interval;

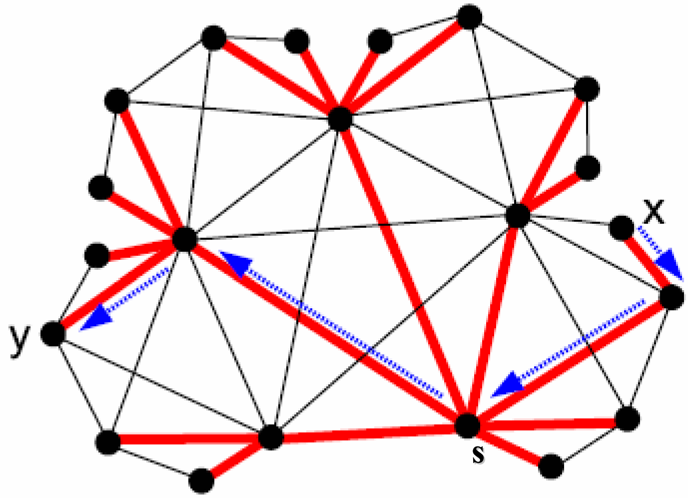
else (which means $x \notin N_G[sTy]$),

go to a neighbor w of x in G such that $x \in I_w$ and I_w is largest such interval.

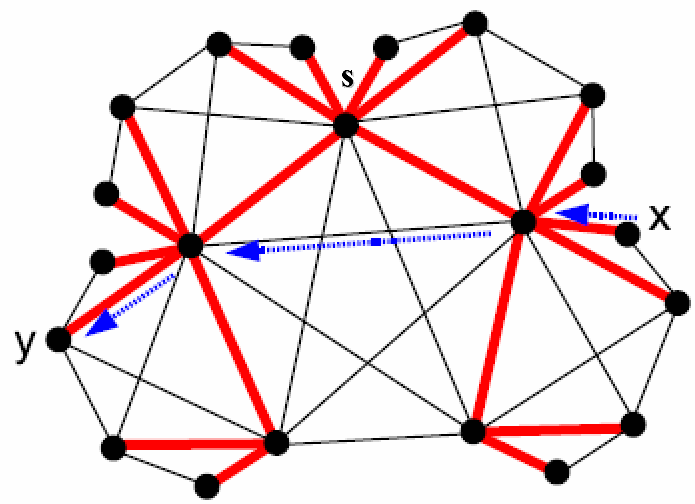
Additive r-frame:

a spanning tree T of a graph G is an *additive r-frame* for G if $g_{G,T}(x,y) \leq d_G(x,y) + r$ for each ordered pair $x, y \in V$

Each vertex knows its neighbors and 2 digits from the tree



additive tree 3-spanner



locally connected spanning tree

Frames for *k*-chordal graphs

Definition: A graph *G* is called *k*-chordal if it has no induced cycles of size greater than *k*. When *k*=3, *G* is a chordal graph.

Theorem: Any BFS-tree *T* of a *k*-chordal graph *G* is an additive (*k*-1)-frame.

Theorem: Any LexBFS-tree *T* of a chordal graph *G* is an additive *1*-frame.

Frames for chordal bipartite graphs

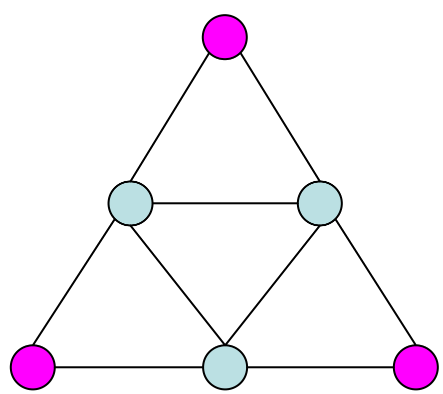
Definition: A graph *G* is called chordal bipartite if it is bipartite and has no induced cycles of size greater than 4.

Theorem: Every chordal bipartite graph *G* admits an additive *0*-frame

Frames for AT-free graphs

Definition: A graph is called AT-free if it does not have an asteroidal triple (i.e., a set of three vertices such that there is a path between any pair of them avoiding the closed neighborhood of the third).

Theorem: Any BFS-tree *T* of an AT-free graph G is an additive *2*-frame.



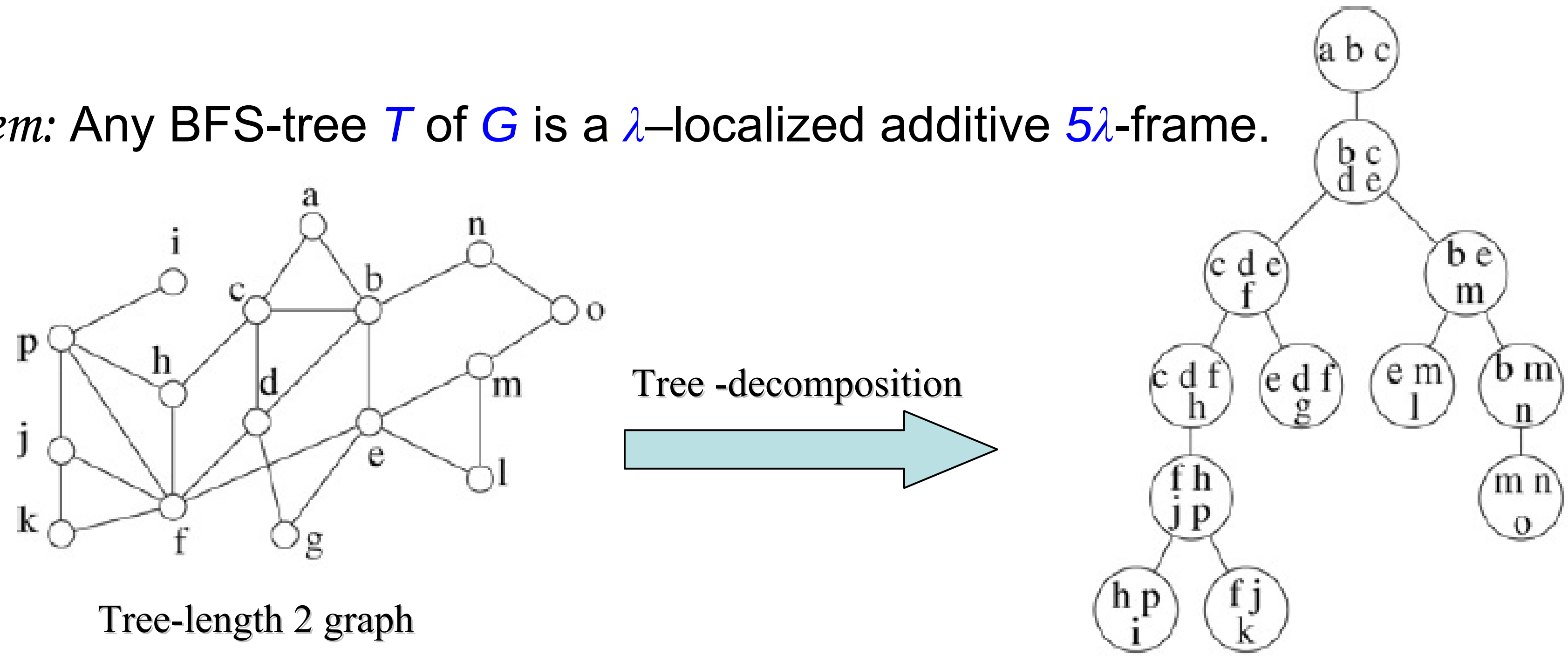
An asteroid triple

Frames for tree-length λ graphs

Definition: A graph G is a tree-length λ graph if and only if G admits a tree-decomposition into bags of diameter at most λ. The tree-decomposition is a tree T whose vertices, called bags, are subsets of $V(G)$ such that:

- $\bigcup_{X \in V(T)} X = V(G)$;
- For all $uv \in E(G)$, there exists $X \in V(T)$ such that $u,v \in X$; and
- For all $X, Y, Z \in V(T)$, if Y is on the path from X to Z in T then $X \cap Z \subseteq Y$.

Theorem: Any BFS-tree T of G is a λ–localized additive 5λ-frame.



Tree-length 2 graph

Frames for δ-hyperbolic graphs

Definition: G is a δ-hyperbolic graph if and only if for any four vertices $u,v,w,x,$ the two larger of the distance sums $d_G(u,v)+d_G(w,x)$, $d_G(u,w)+d_G(v,x)$, $d_G(u,x)+d_G(v,w)$ differ by at most 2δ.

Theorem: Any BFS-tree T of G is a 4δ -localized additive 8δ-frame.

Note that hyperbolic graphs are used to model some topological properties of the Internet.

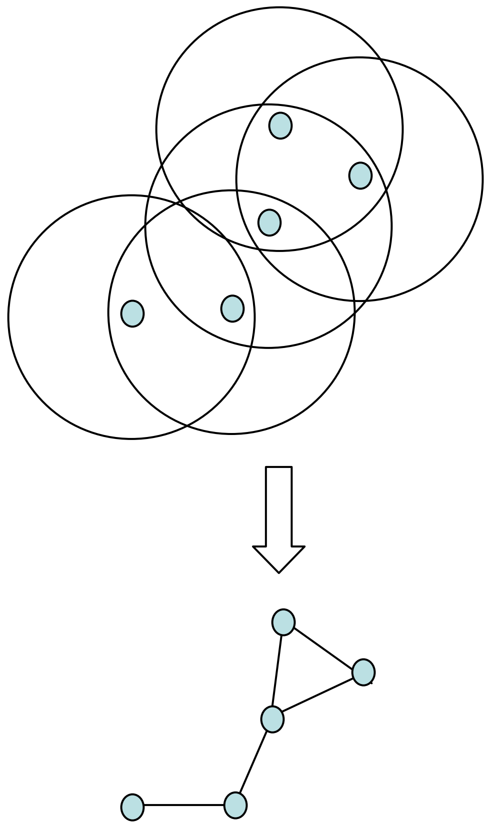
Some Lower Bounds for Frames and Carcasses of Tree-length λ Graphs

- For any $\lambda \geq 4$, there exists a tree-length λ graph G with n vertices for which no $\lfloor 2(\lambda-2)/3 \rfloor$ –localized additive $\frac{2}{3}\sqrt{\log \frac{3(n-1)}{4\lambda}}$ –frame exists.

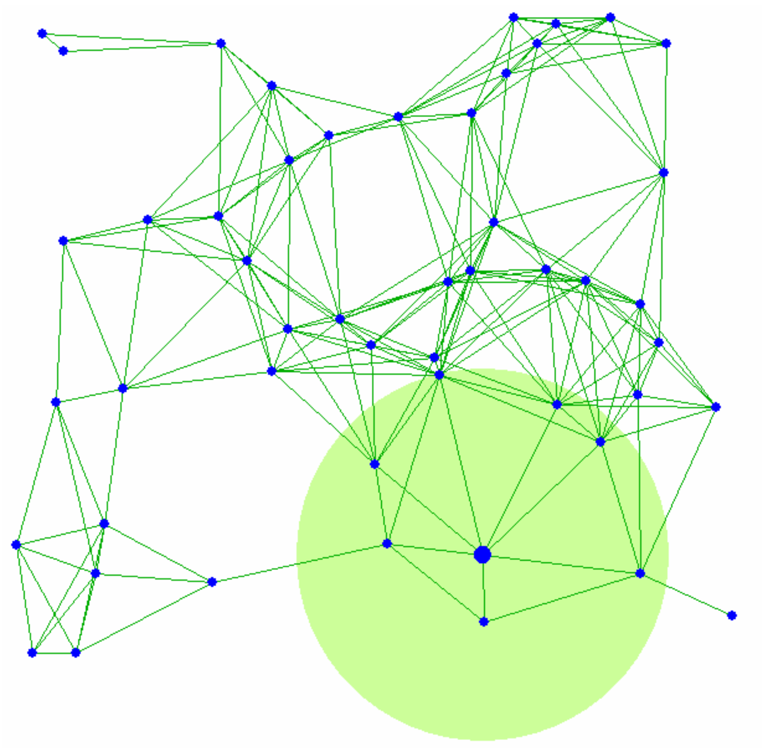
- For any $\lambda \geq 6$, there exists a tree-length λ graph G with n vertices for which no $\lfloor (\lambda-2)/4 \rfloor$ –localized additive $\frac{3}{4}\sqrt{\log \frac{n-1}{\lambda}}$ –carcass exists

Unit Disk graphs

- Unit Disk Graphs (UDGs) are the intersection graphs of equal sized circles in the plane.

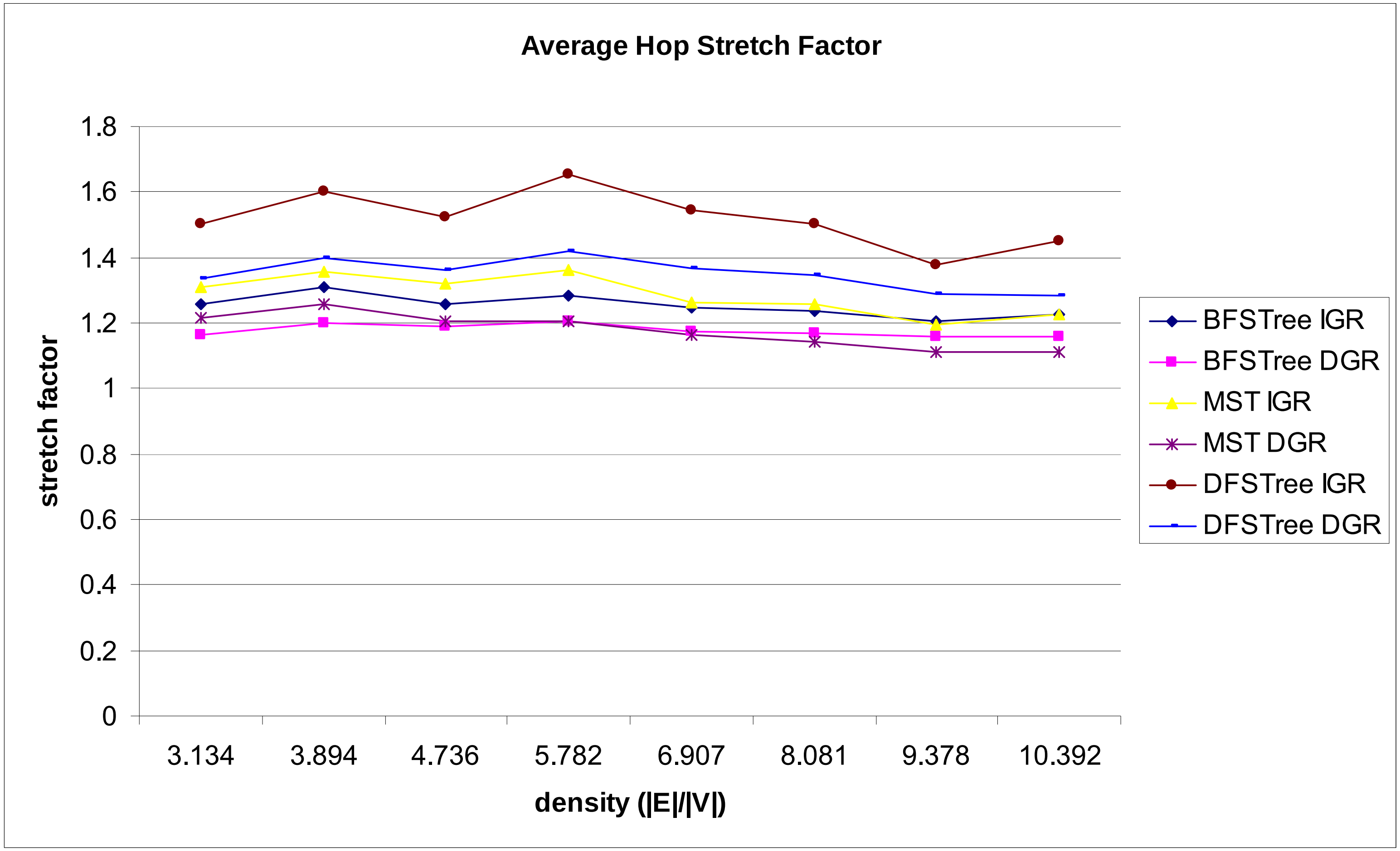


Model wireless networks



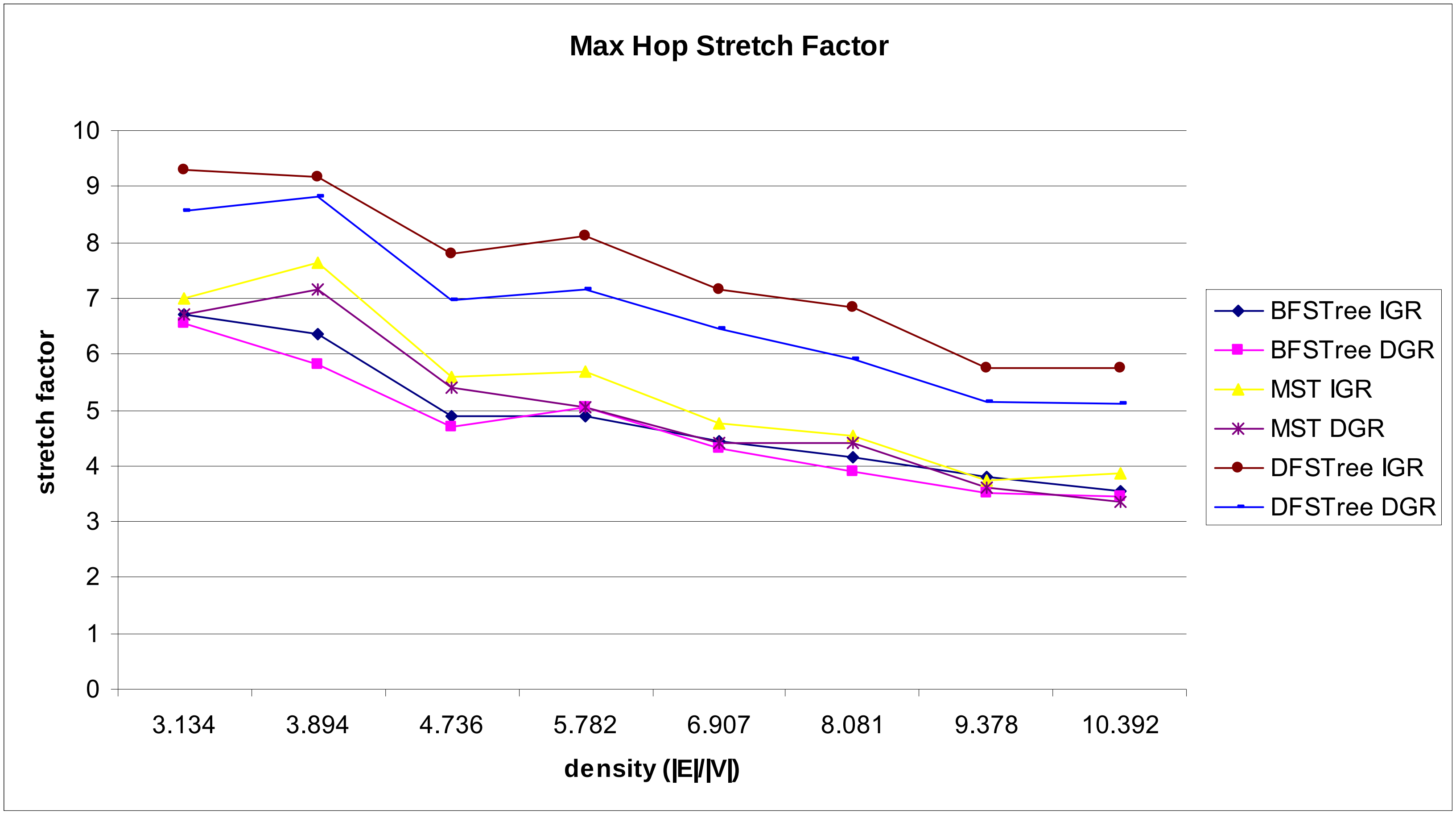
Experimental results in UDGs

Fix the number of vertices, change density



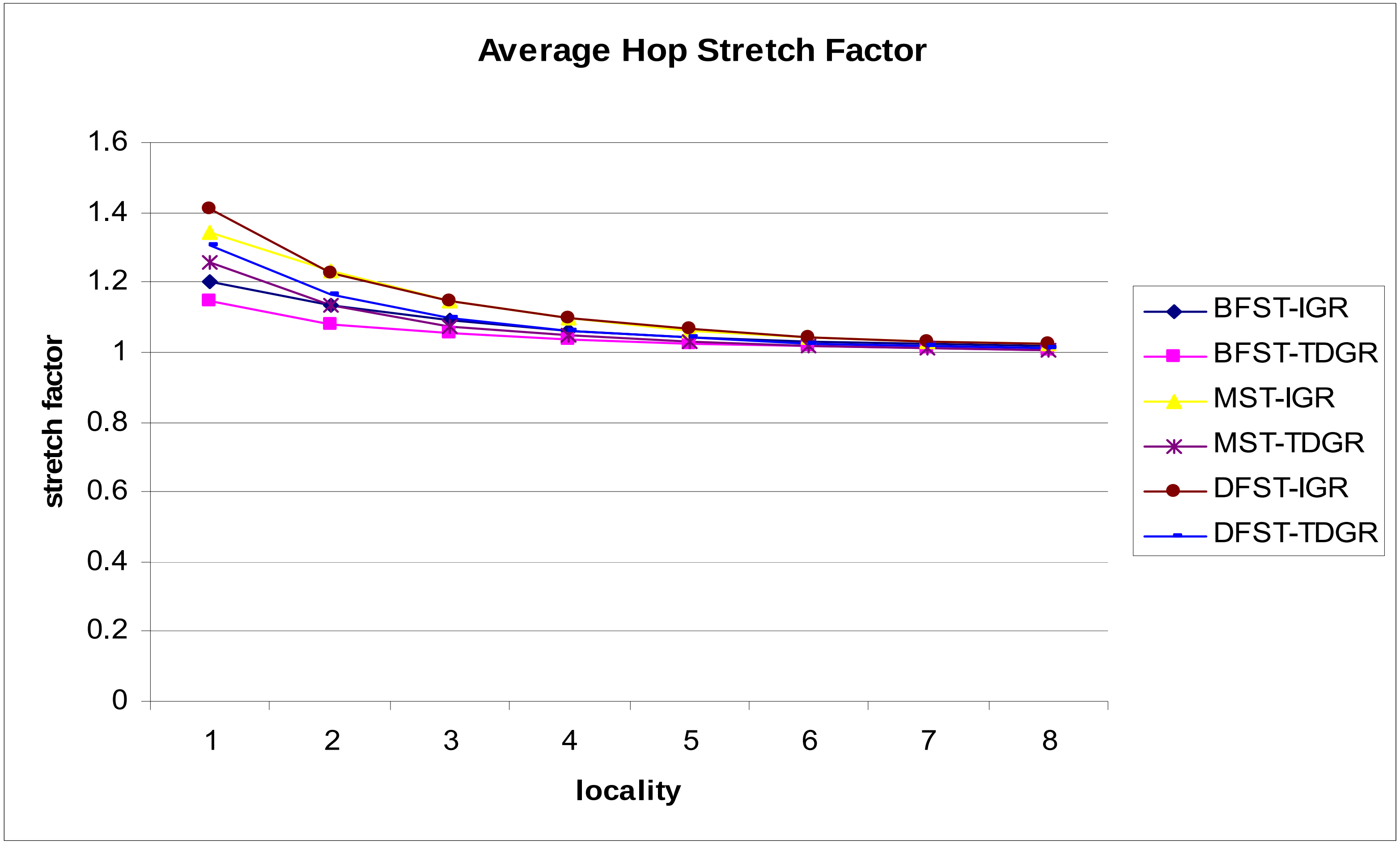
Experimental results in UDGs

Fix the number of vertices, change density



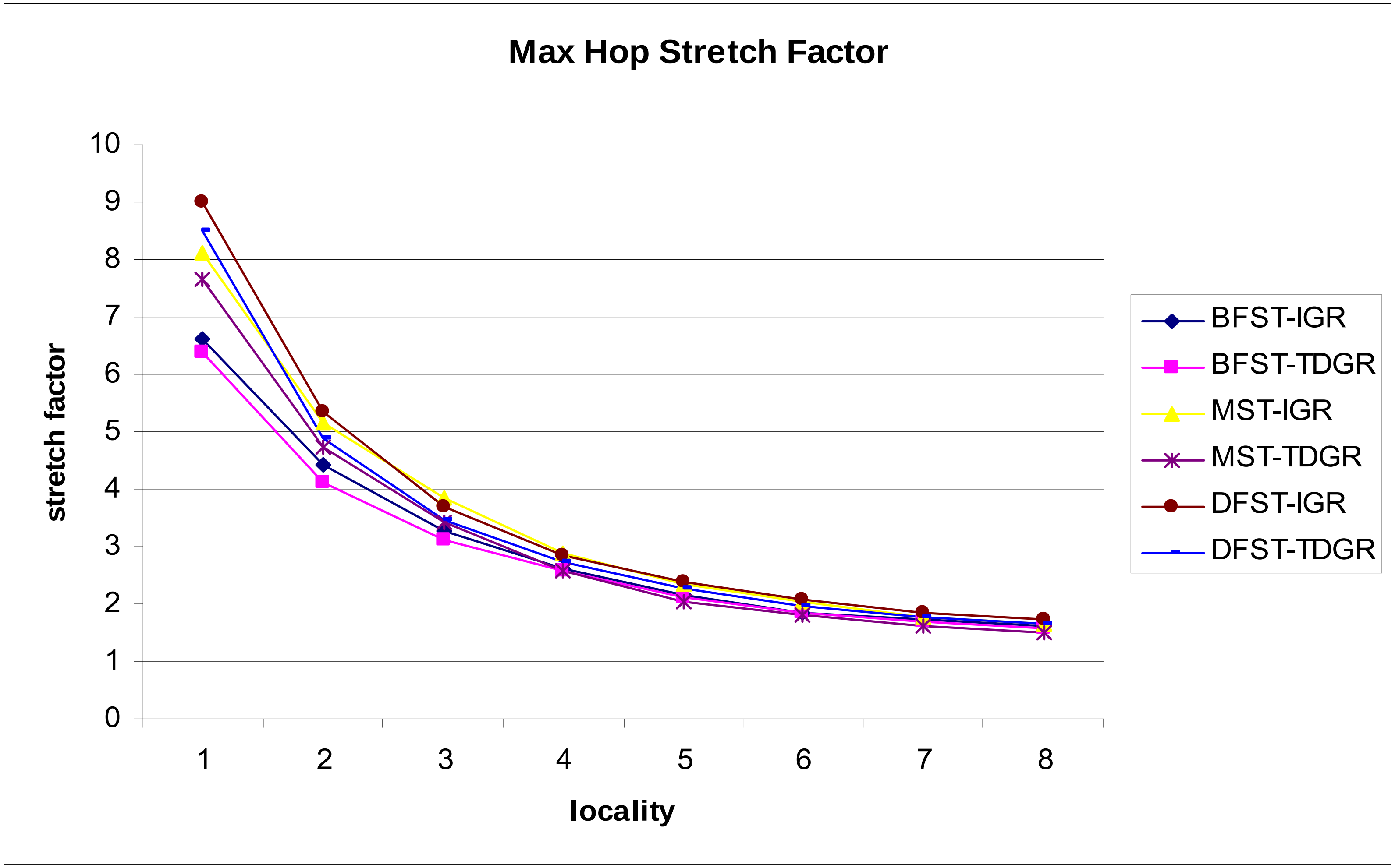
Experimental results in UDGs

Fix the number of vertices, change locality



Experimental results in UDGs

Fix the number of vertices, change locality



Thank You