GTNA
Routing Implementation

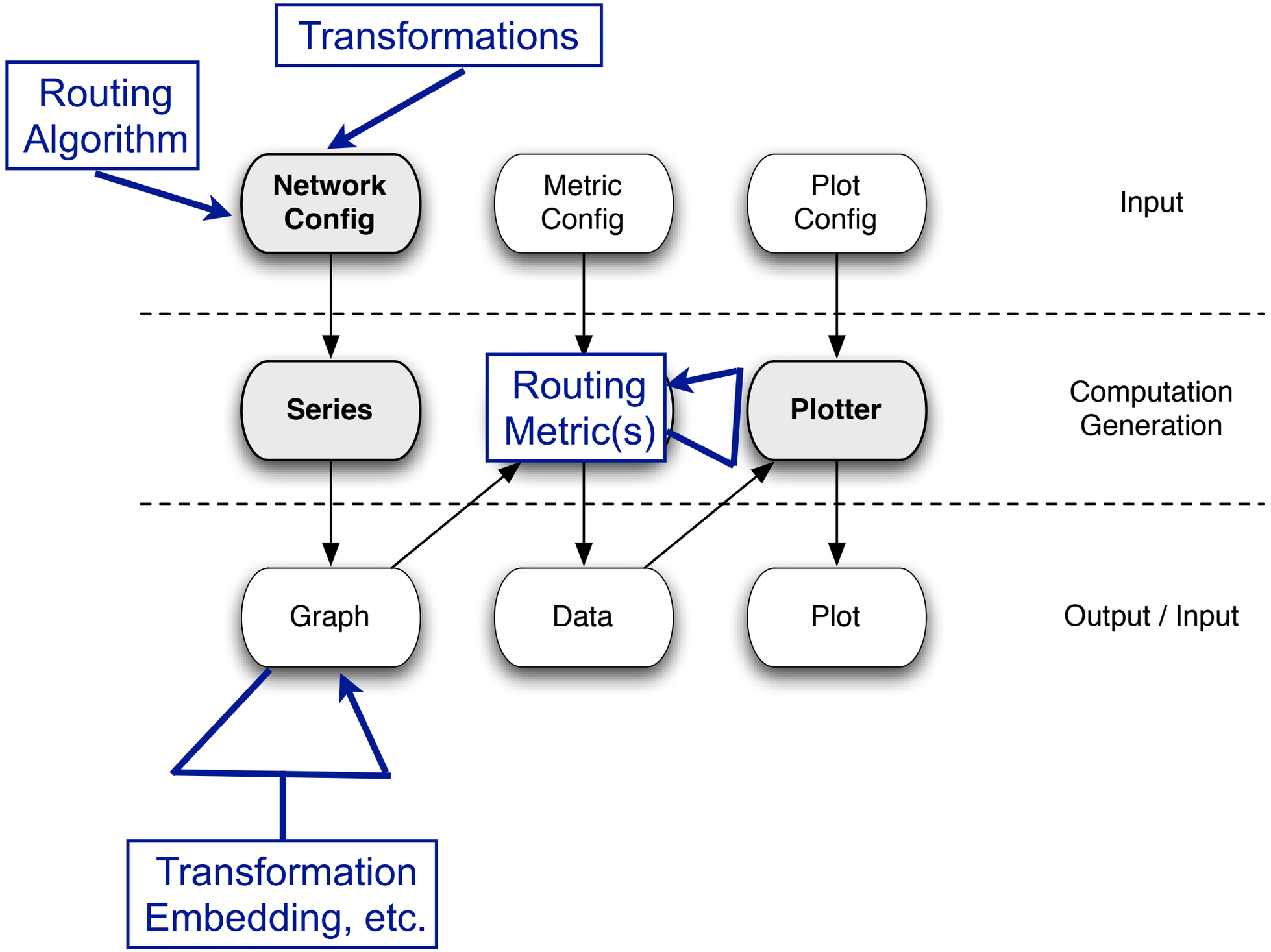
Benjamin Schiller, P2P Networks Group, TU Darmstadt

05.07.2011

Outline

- GTNA Workflow
- Metrics
- Performing routing
- Current components
- Current metric implementation
- Planned changes

GTNA workflow



Metrics

- Path length
  - Distribution, average, etc. of achieved routing paths
- Storage at nodes (after initialization)
  - Size / number of e.g. routing table, items, replicas, (key, value) pairs
- Initialization messages
  - Number of message per nodes during initialization
- Maintenance messages
  - Number of messages per node required for maintenance
- Resilience to errors / attacks
  - NOW: implementation in network generators

Performing the routing procedure on a graph

- For every node n in the graph
  - Select X nodes at random
  - Route towards them
  - record path taken

Current Components

- Graph, nodes, [edges]
  - In, out, index, storage, etc.
- IDNode
  - boolean contains(id)
  - double dist(id) / dist(node)
  - id randomID(nodes)
- Route
  - ArrayList<Node> path
  - boolean success()
  - int messages()
- Routing algorithm
  - Route randomRoute(nodes, src)
  - boolean applicable(nodes)
  - void init(nodes)

Current Metric Implementation

- void computeData(Graph g, Network nw, Hashtable metrics)
  - if(node instanceof StorageNode)
    - compute storage metrics
  - if(node instanceof messageCounter)
    - compute messages
  - ….

Planned changes

- Metrics: divide into multiple metrics
  - each one: applicable(node)
  - possible METRIC_DEPENDS_ON = ...
  - access to others by Hashtable<String, Metric> computedMetrics

- Unification of id / target selection

- Definition of node interfaces (storage counter, etc.)