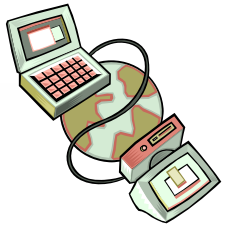
veritas
iustitia
libertas
Freie Universität Berlin

DHT-based Unicast for Mobile Ad Hoc Networks

Thomas Zahn, Jochen Schiller
Freie Universität Berlin



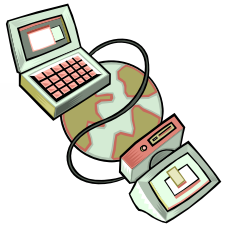
Background – MADPastry

- MADPastry combines ad hoc routing and P2P overlay routing at the network layer

  ➔provides **indirect – i.e. key-based** – routing in MANETs

- DHT-based distributed network applications from the Internet can be ported to MANETs

  – e.g. name services, messaging systems, event-notification, storage systems

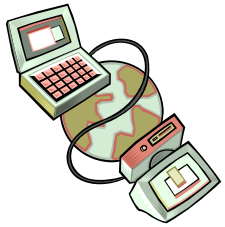


MADPastry – Overview

- Integrates physical (AODV) and overlay (PASTRY) routing at the network layer

→ provides indirect routing (i.e. key based) primitive in MANETs

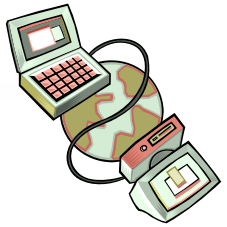
- MADPastry nodes maintain AODV RT and (degenerate) Pastry RT
- MADPastry explicitly considers locality in the construction of its overlay



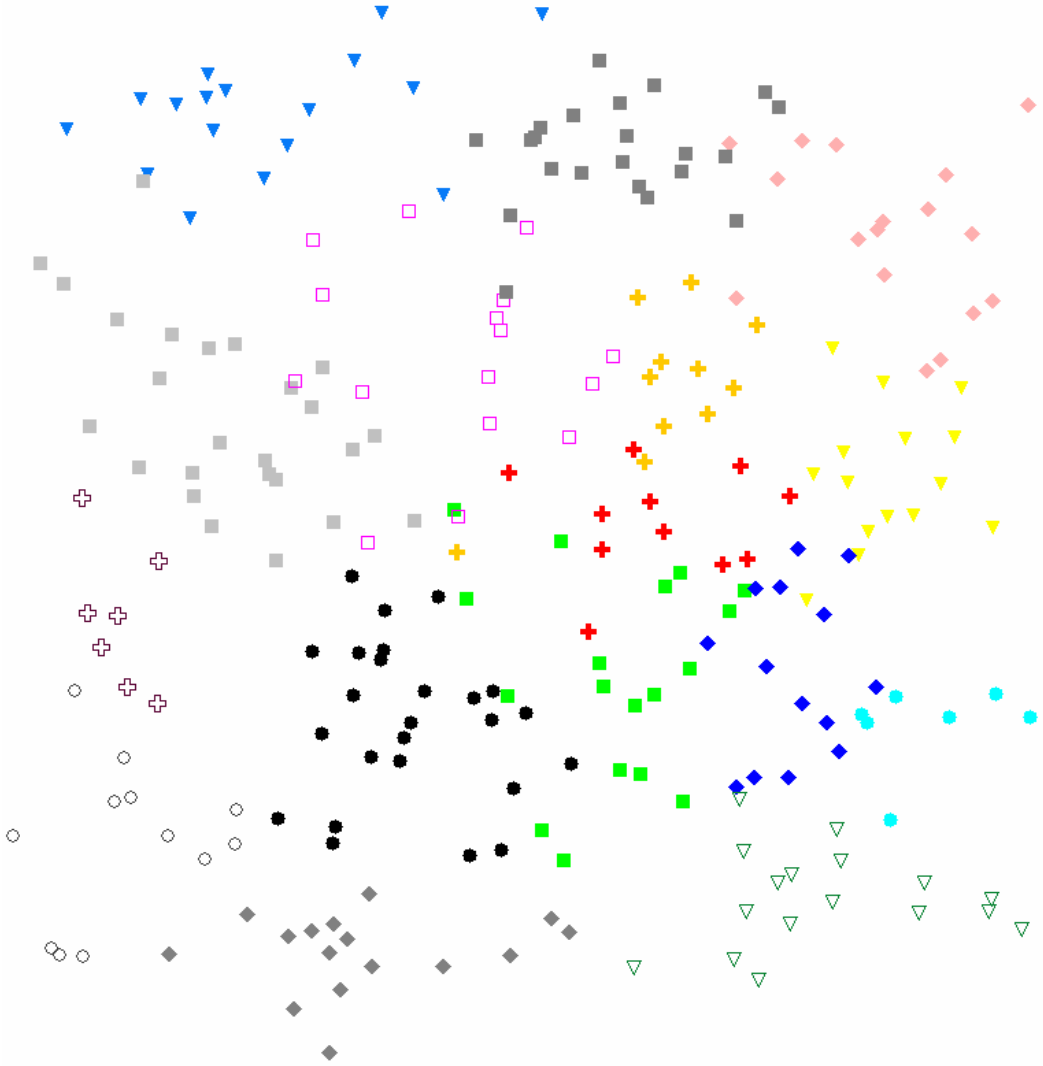
MADPastry – Random Landmarking

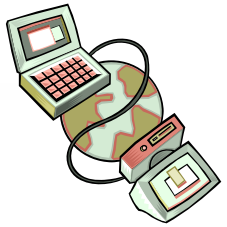
- No fixed landmark nodes, landmark keys instead: 0800..00, 1800..00, ..., F800..00
- Node currently closest to a landmark key ➔ temporary landmark node
- Periodic beacons to form physical clusters of common overlay ID prefixes
- Node associates itself with closest temporary landmark → assumes same overlay ID prefix

  → physically close nodes are also likely to be close in the overlay



MADPastry – Spatial Topology





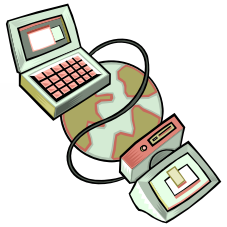
MADPastry - Routing

- At the beginning of an overlay hop, node inspects Pastry RT → destination of current overlay hop
- (Intermediate) nodes on physical path of an overlay hop consult AODV RT for the next physical hop
- When physical route for an overlay hop is unknown:

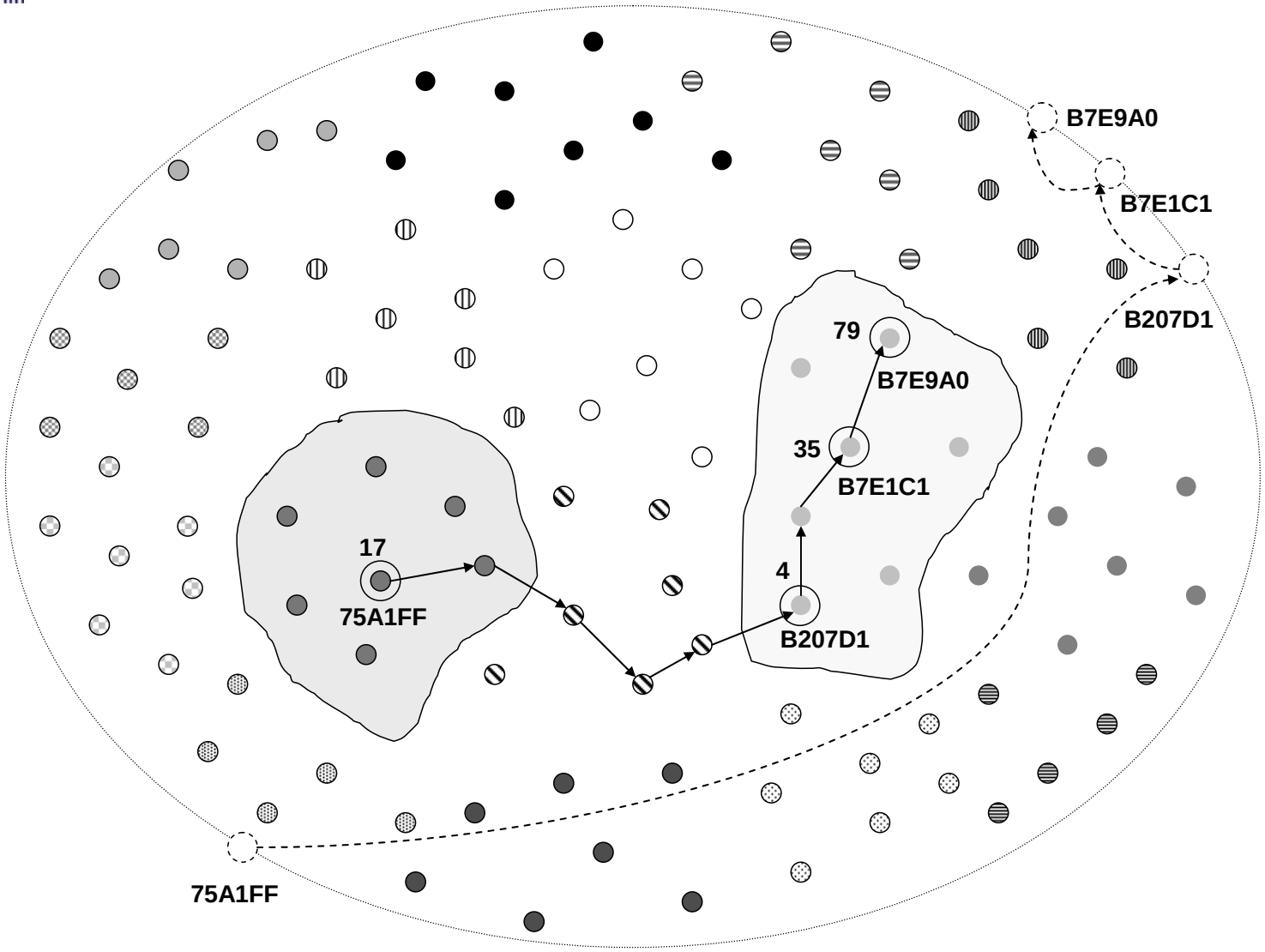
→inspect pkt's key → possibly redirect

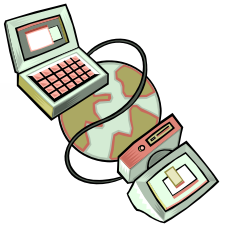
→if already in my cluster, restrict route discovery to cluster broadcast

→otherwise, do full AODV route discovery



MADPastry – Key-Based Routing



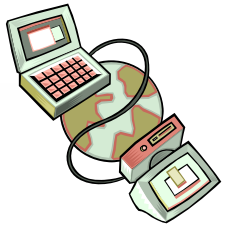


Motivation

- MADPastry solves key-based routing in MANETs
- **BUT**: in MANETs **unicast** (given src → given dst) routing is also needed
- Need to maintain conventional ad hoc protocol for network-layer unicasting

➔Maintenance of 2 routing protocols

- **MADPastry can also do unicasts**

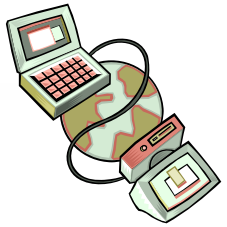


Concept

- **<u>Key Assumption</u>: In MANETs it can be advantageous to travel numerous short and up-to-date routes instead of one long direct route**

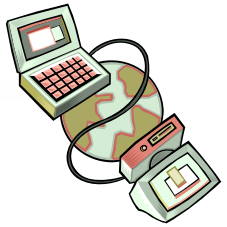
- Challenge: MADPastry routing is based on overlay keys

→ Node x needs to find out node y's current overlay key before sending a msg

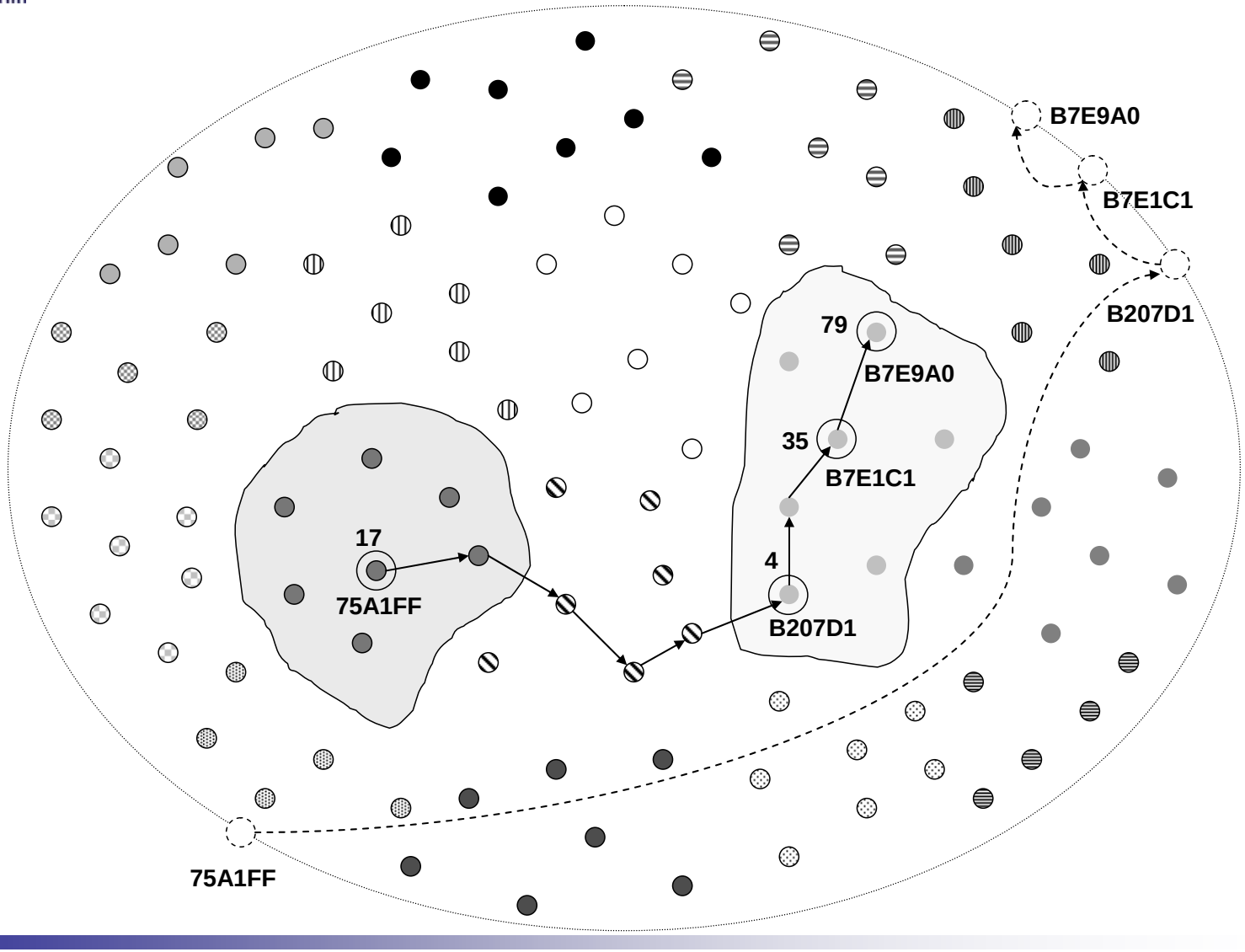


MADPastry Unicast – Address Server

- Each nodes has exactly one temporary address server
- Address server stores its client's current overlay ID
- Node A hashes its node ID → hash key
- Node A publishes its current overlay ID towards hash key using MADPastry
- Node currently responsible for node A's hash key becomes node A's address server

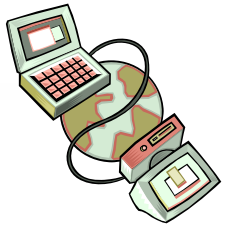


Address Publication

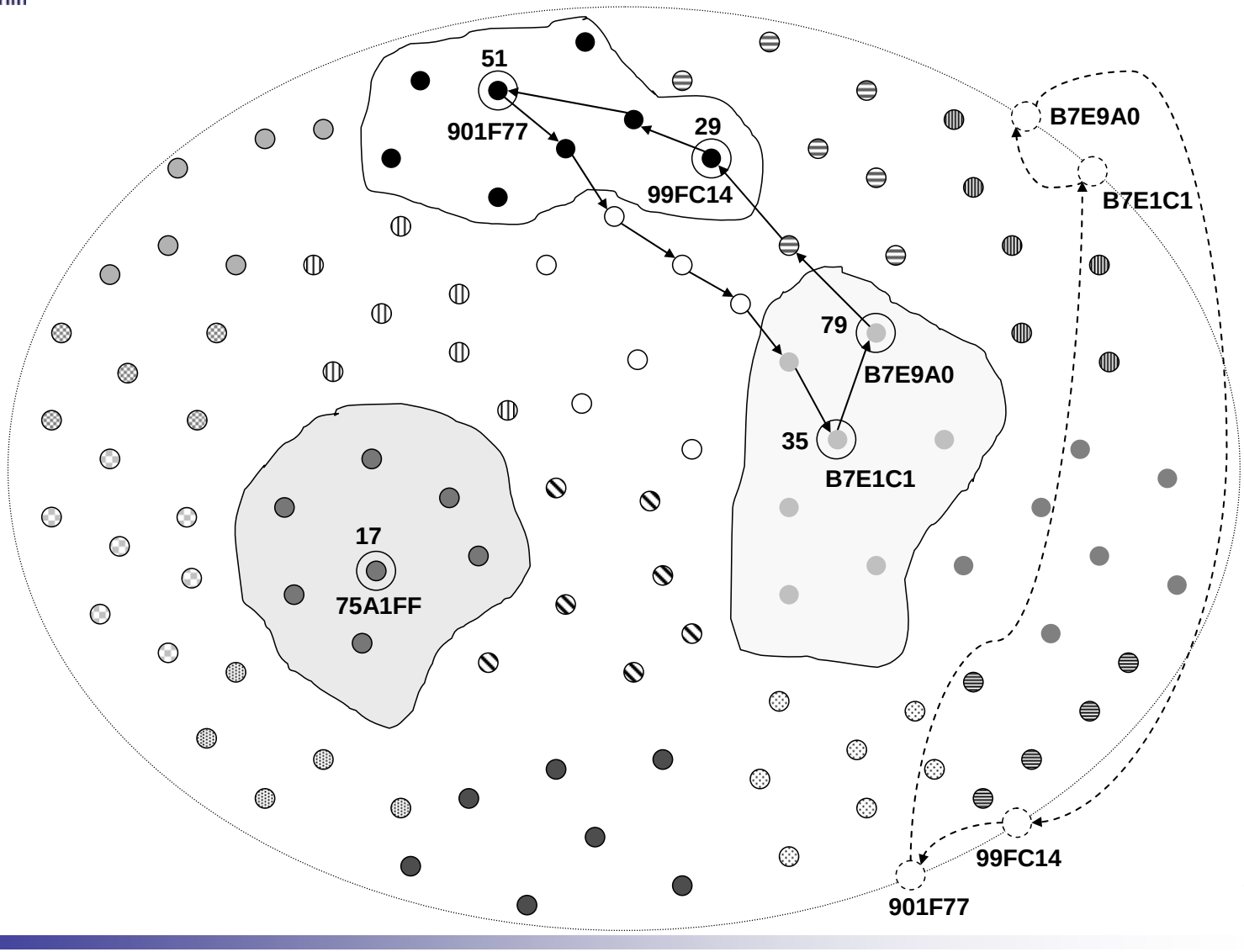


MADPastry Unicast – Address Resolution

- Node A wants to communicate with node B
- Node A does not know node B's current overlay ID
- Node A hashes node B's net ID to get hash key
- Node A sends request towards hash key
- Node B's address server replies with node B's current overlay ID



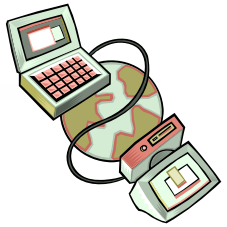
Address Resolution



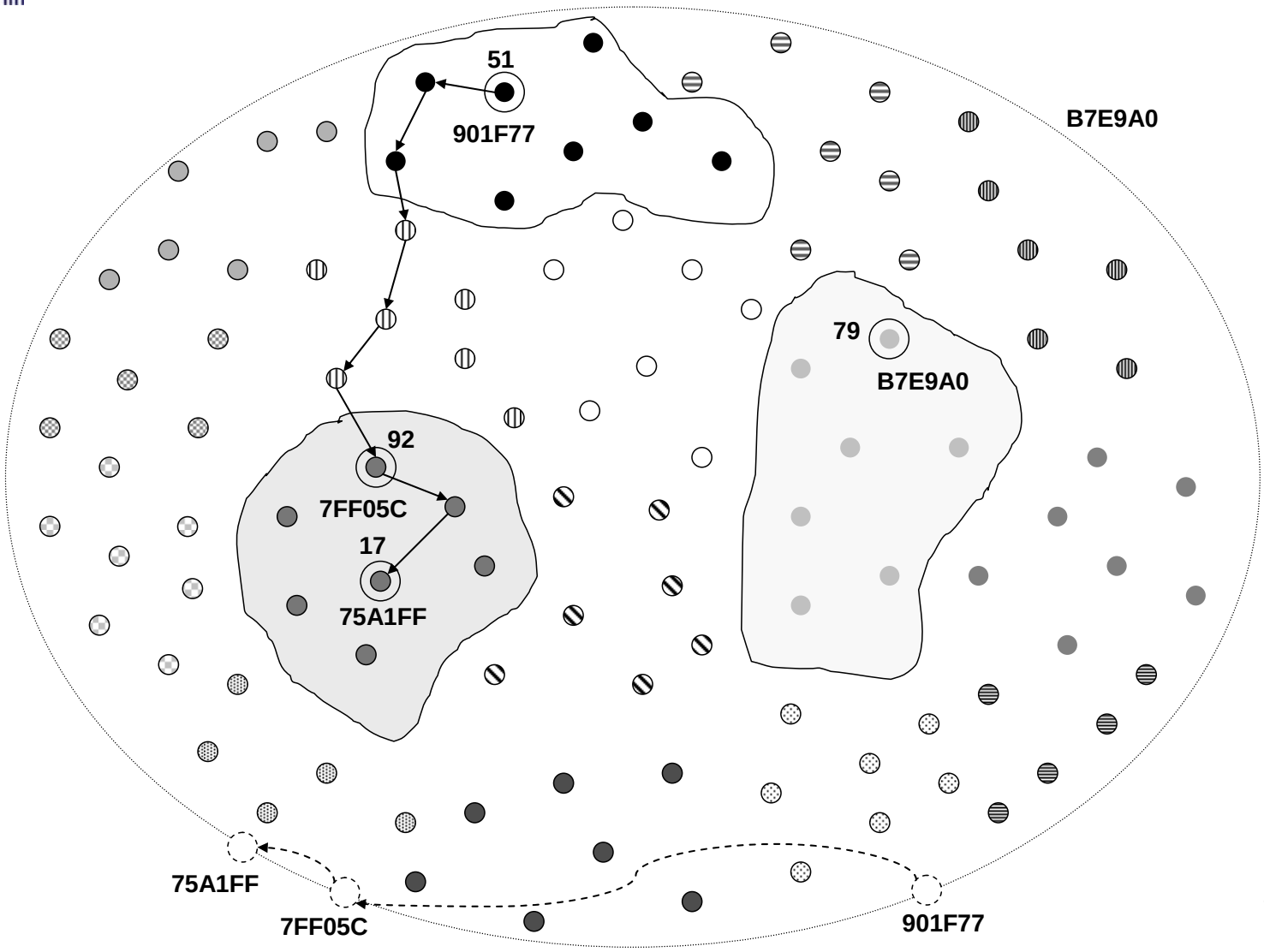
MADPastry Unicast

- Node A uses overlay ID from reply to send message to node B

- MADPastry delivers message using indirect routing



Unicast

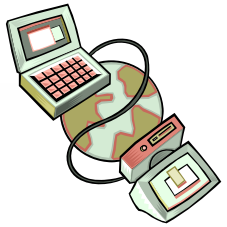

51
901F77
B7E9A0
79
B7E9A0
92
7FF05C
17
75A1FF
75A1FF
7FF05C
901F77

Simulation Results

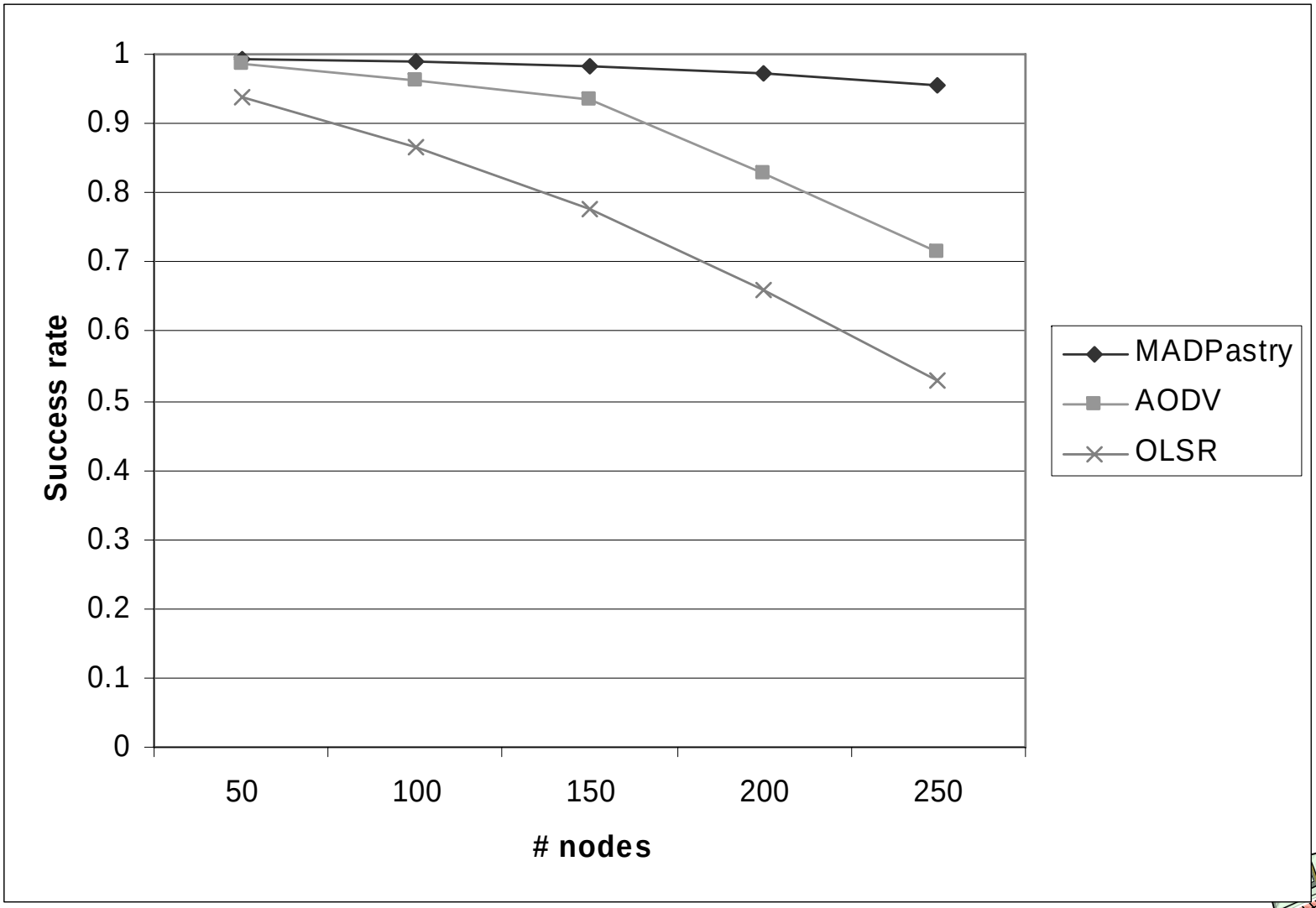
- Compare MADPastry's unicast against a popular reactive and proactive ad hoc routing protocol

  →AODV (reactive), OLSR (proactive)

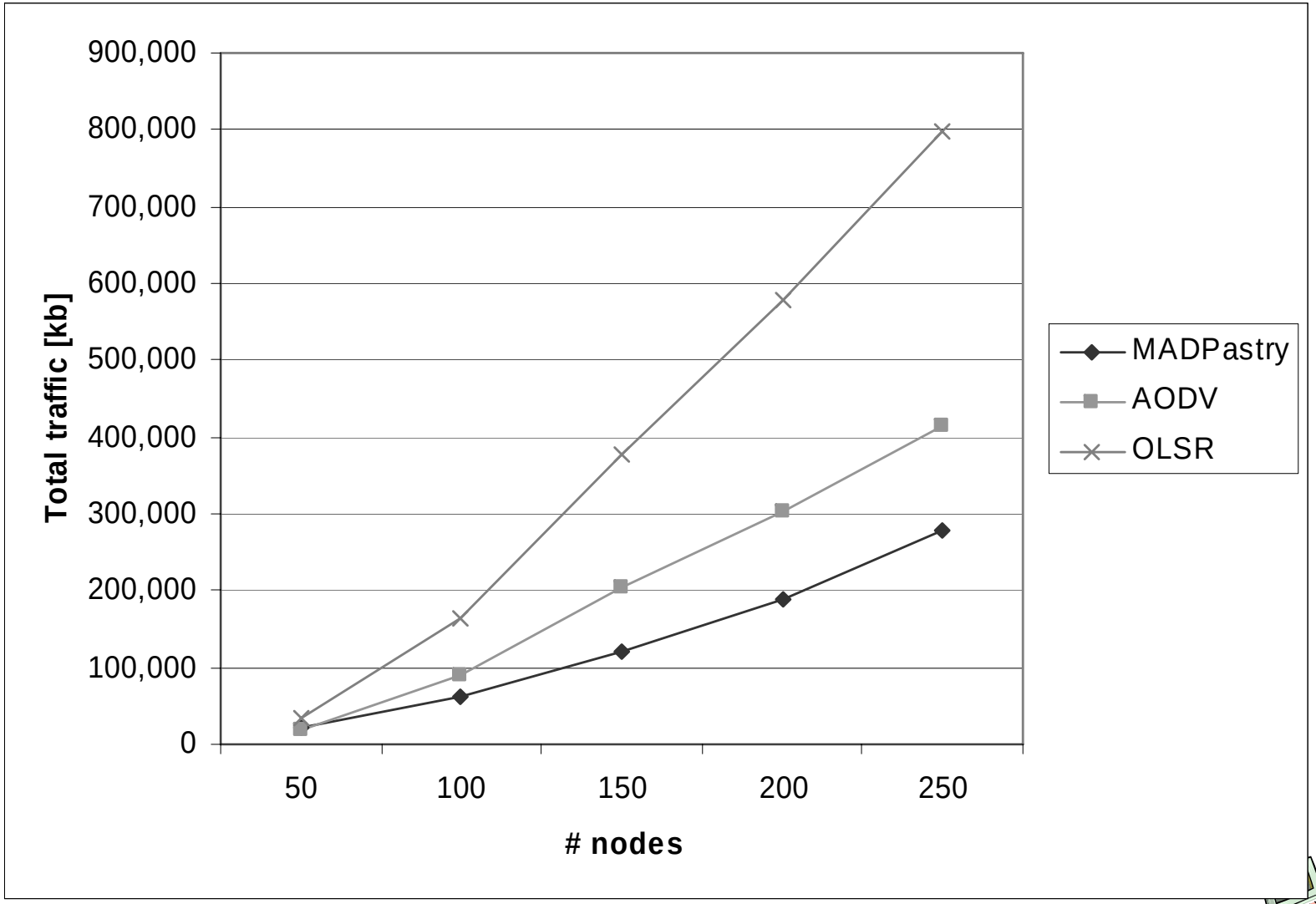
- Simulations in ns2
- varying network sizes (50,100,150,200,250)
- varying node velocities (0.1,1.4, 2.5,5.0 m/s)
- 1 random request every 10s per node

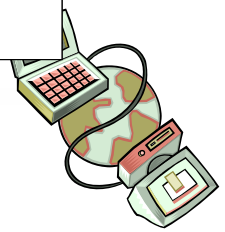


Success Rates – 1.4 m/s

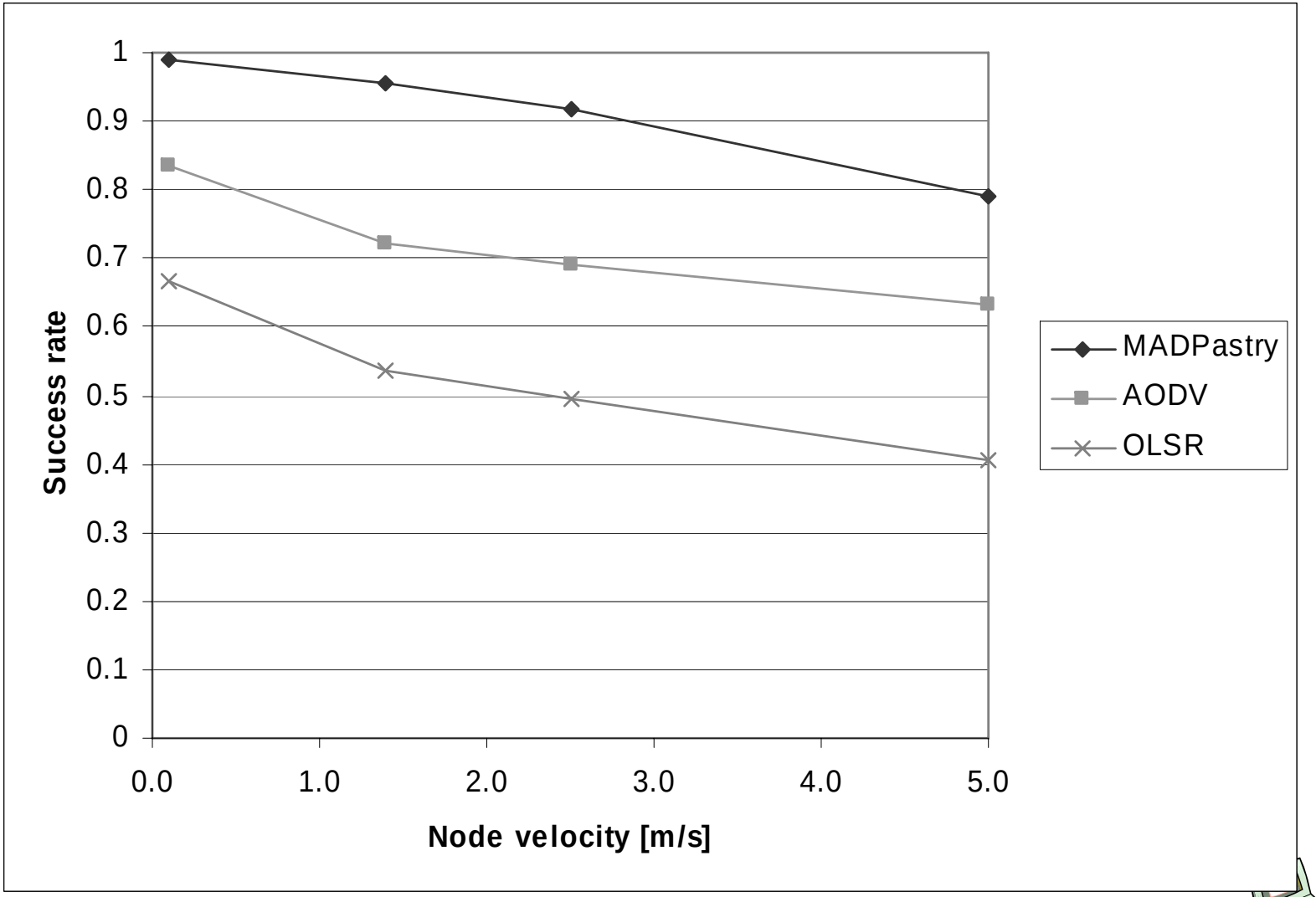


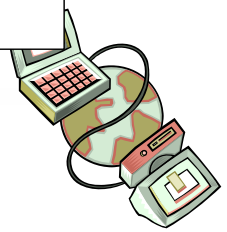
Total Traffic – 1.4 m/s





Success Rates vs Node Velocity



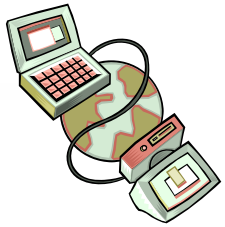


Conclusion - I

- MADPastry's unicast can outperform popular reactive and proactive ad hoc routing protocols

- Key assumption appears confirmed

→In MANETs it can be advantageous to travel numerous short up-to-date routes instead of one long direct route

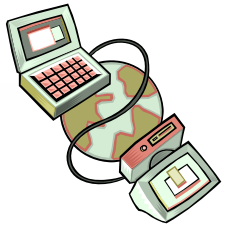


Conclusion - II

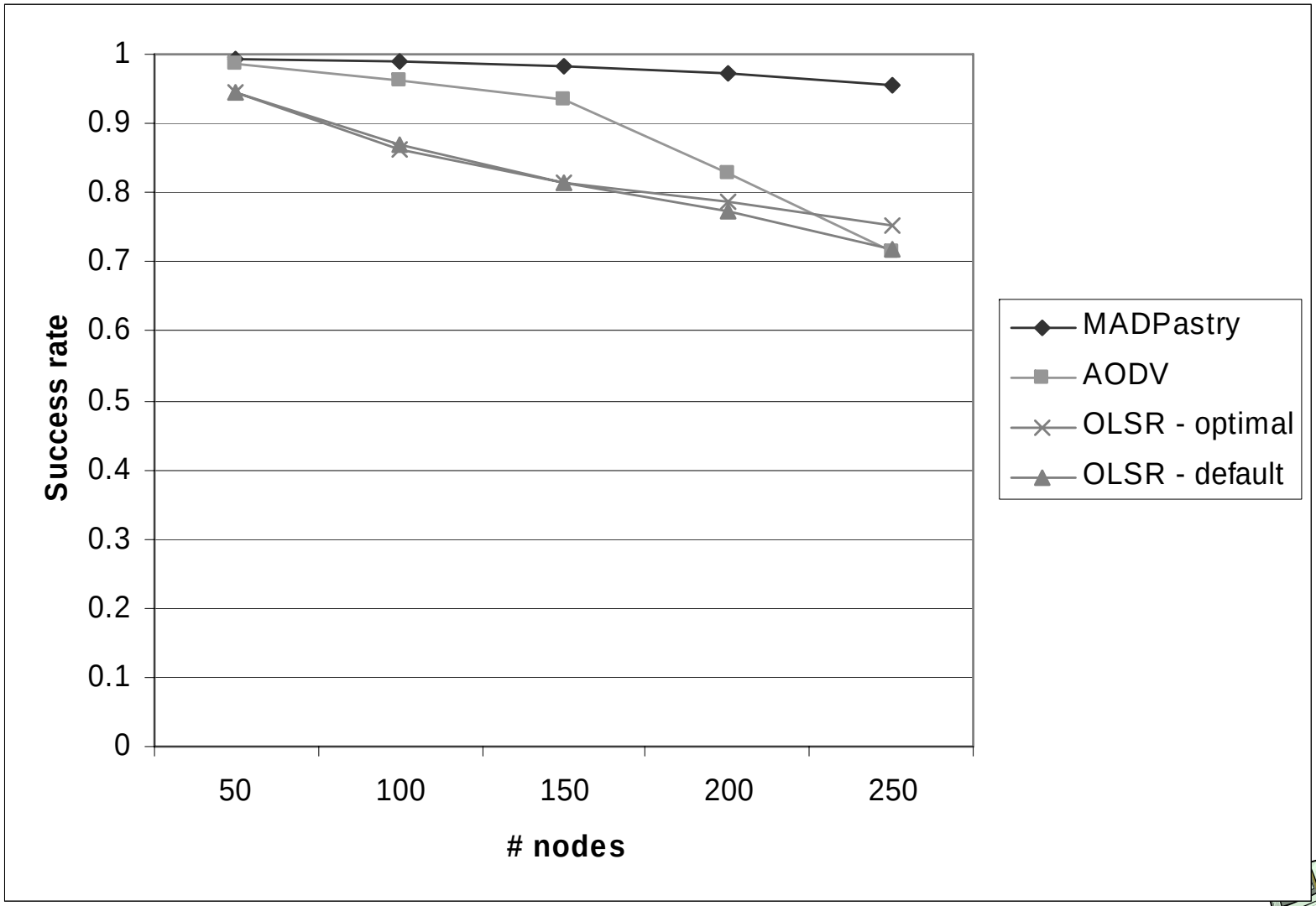
- MADPastry's unicast **NOT** one-fit-all-scenarios solution for unicast in MANETs
- BUT: MADPastry already present in the MANET to provide key-based, indirect routing

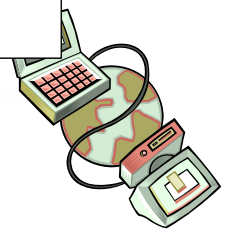
→MADPastry can also provide point-to-point unicasting

→No need to maintain ad hoc routing protocol in parallel

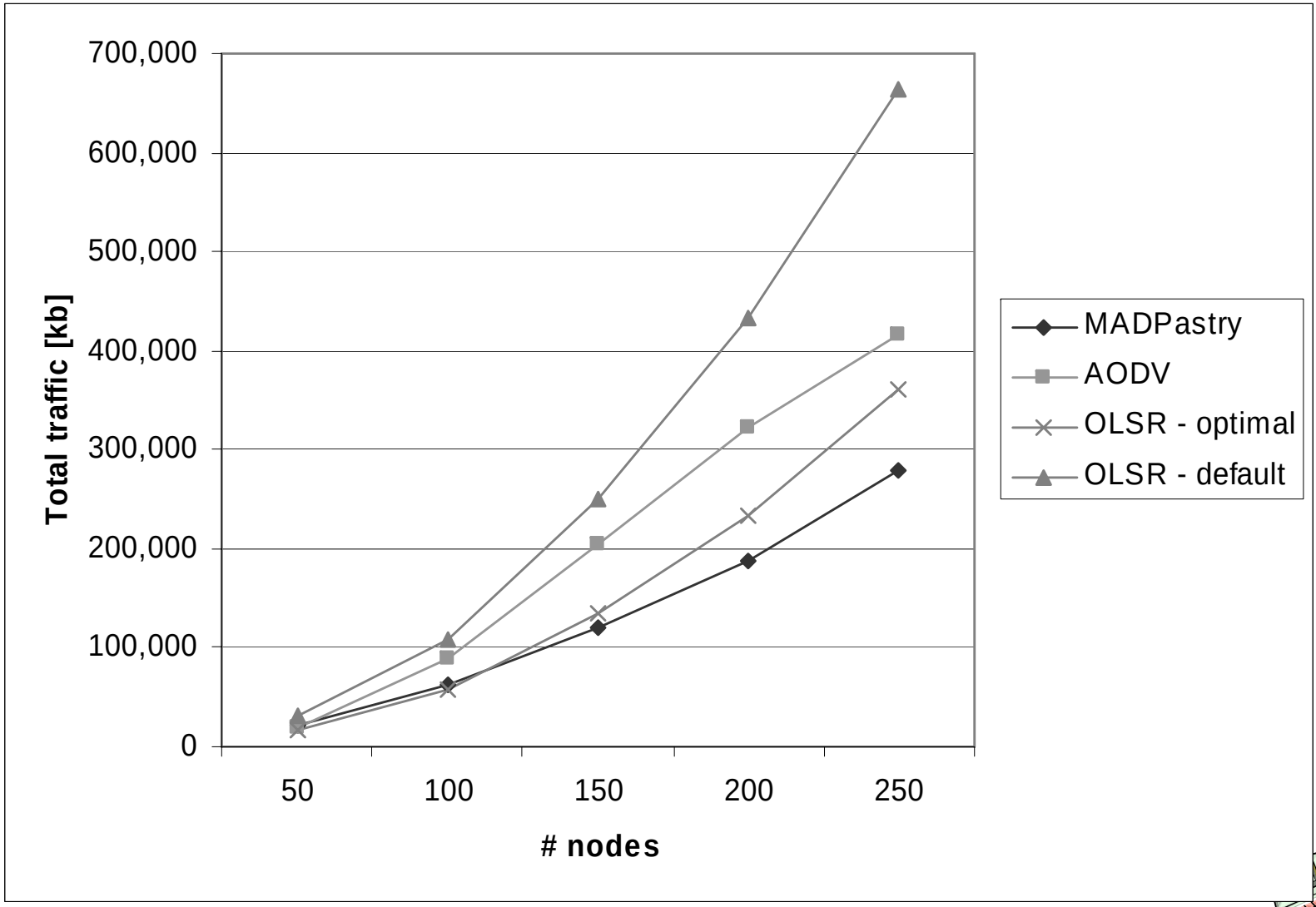


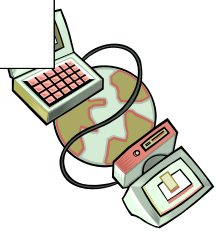
Success Rates – 1.4 m/s / New OLSR Implementation





Overall Traffic – 1.4 m/s – New OLSR Implementation





Success Rates vs. Node Velocity – 250 nodes – New OLSR Implementation