Drahtlose Netzwerke
Grundlagen und Einsatzfelder

Ad-Hoc Netzwerke

Was ist ein Ad-Hoc Netzwerk?

- **Ein Netzwerk, welches bei Bedarf entsteht**
- **Nicht von existierender Infrastruktur unterstützt**

- **Alternative Bezeichnungen:**
  - Instant Infrastructure
  - (Mobile-) Mesh Networking
  - Packet Radio Network (PRNet)

- **Forschungen seit 1972 (DoD)**

Anwendungen I

- **Alles, wozu existierende Netzwerke benutzt werden:**
  - Telefonie (GSM, UMTS, POTS, PSTN)
  - Internetanbindung
  - ...

- **Weitere Anwendungen**
  - Spontane Treffen
  - „Ubiquitous Computing“
  - Katastrophenfall
  - Militär

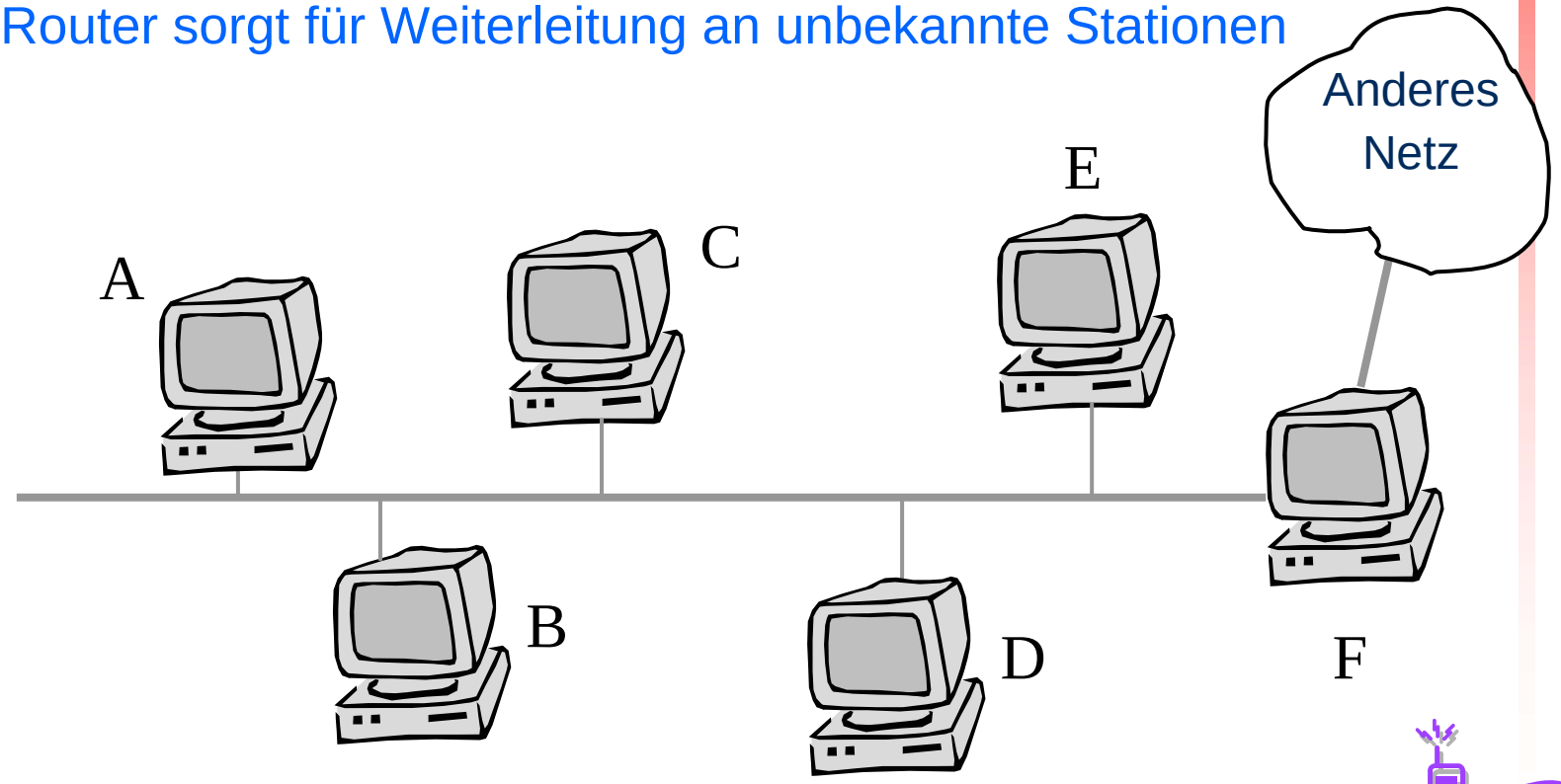
Anwendungen II

- **Ad-Hoc Netzwerke zur Versorgung unterversorgter Gebiete**
  - Telefon (VoIP), Internet
  - Private Initiativen oder kleine Firmen
- **Kostengünstige Alternative zur Infrastruktur**
- **Ortsabhängige Dienste (*Location-Based Services*)**
- ***Ubiquitous Computing***
  - Computer überall
- **Kommunikation bei Katastrophenfällen**

Drahtgebunden vs. Drahtlos (1)

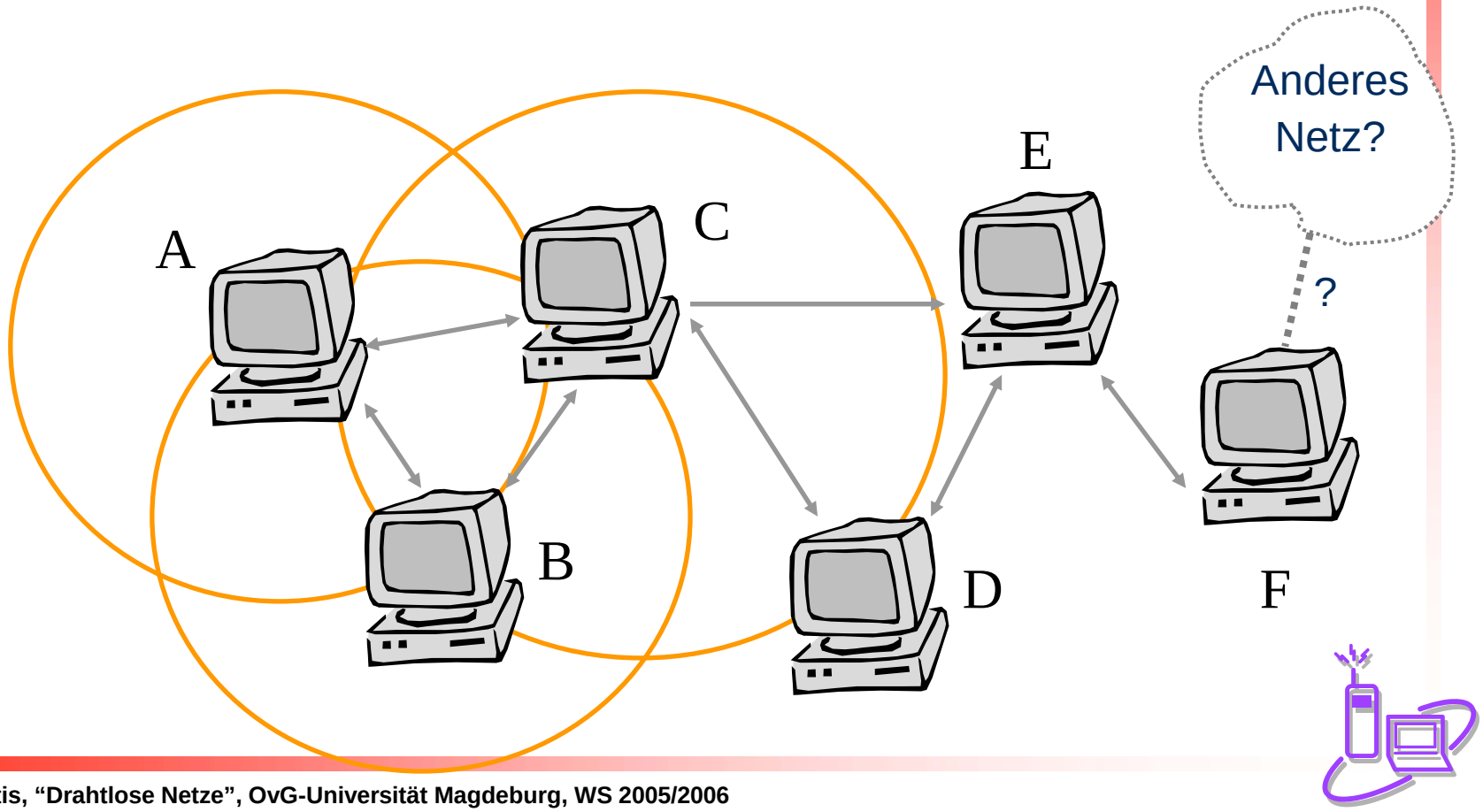
- **Drahtgebundenes Netzwerk:**
  - Jede Station kann jede andere direkt erreichen

  oder
  - Router sorgt für Weiterleitung an unbekannte Stationen



Drahtgebunden vs. Drahtlos (2)

- **Drahtloses Netzwerk:**
  - Eingeschränkte Erreichbarkeit
  - Jede Station ist ein (potenzieller) Router



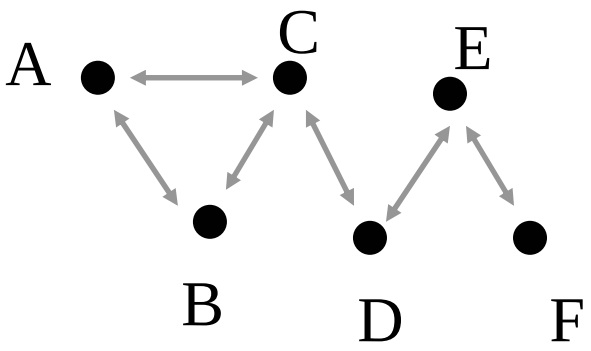
Klassifikationen von Ad-Hoc Netzwerken

- **1 Hop vs. Multi-Hop**

- **Sensornetzwerk vs. MANET**
  - Sensornetzwerk
    - Autonome Sensoren
    - Statisches Netzwerk
    - Netzwerk zum Auslesen der Sensoren
  - „Mobile Ad-hoc NETwork“ (MANET)
    - PDAs, Laptops, Mobiltelefone, ...
    - (zumeist) dynamisches Netzwerk
    - Jeder will mit jedem kommunizieren können

Routing im Internet (1)

- **Distance Vector Routing (*distributed Bellman-Ford* DBF, *Ford-Fulkerson*)**
  - Basierend auf lokalem Wissen
  - Entscheidungen nur aufgrund von Umgebungsinformationen
  - Geringer Aufwand
  - Schlechtes Verhalten bei Stationsausfall

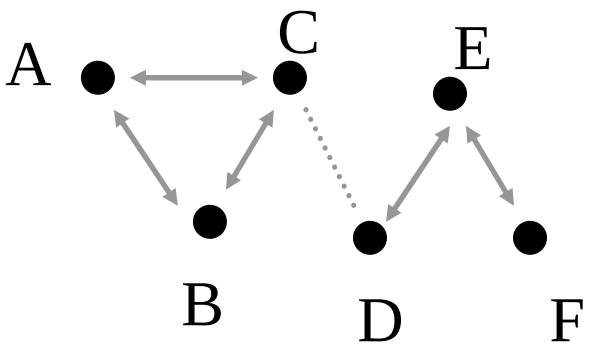


nach \ von	A	B	D
A	-	1	2
B	1	-	2
C	1	1	1
D	2	2	-
E	3	3	1
F	4	4	2

über	Kosten
A	1
B	1
-	0
D	1
D	2
D	3

Routing im Internet (1)

- **Distance Vector Routing**
  - „Count-to-Infinity“ Problem
  - Ausfall von Links braucht sehr lange zum Entdecken
  - Kurz- und langfristige Schleifen möglich!



nach \ von	A	B	D
A	-	1	2
B	1	-	2
C	1	1	1
D	2	2	-
E	3	3	1
F	4	4	2

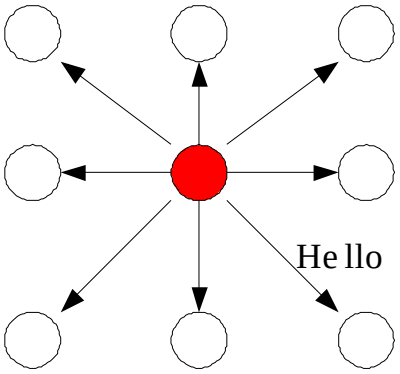
Kosten	über	
1	A	
1	B	
0	-	
3	B	1,D
4	B	2,D
5	A	3,D

Routing im Internet (2)

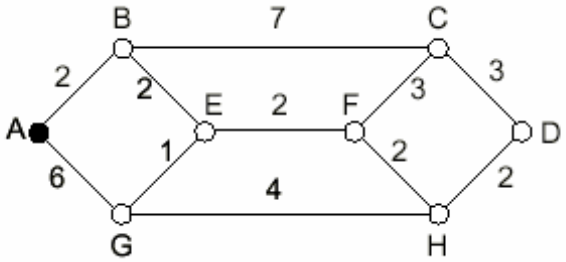
- **Link State Routing**
  - Basierung auf globalem Wissen
  - Basierend auf kürzeste-Wege-Algorithmus von Dijkstra
  - Schnelle Konvergenz
  - Schleifenfrei (nur kurzlebige Schleifen möglich)
  - großer Aufwand
    - Jeder Router braucht vollständiges Wissen
  - z.B. in:
    - IS-IS (Intermediate System to Intermediate System Protocol)
    - OSPF (Open Shortest Path First)

Routing im Internet (3)

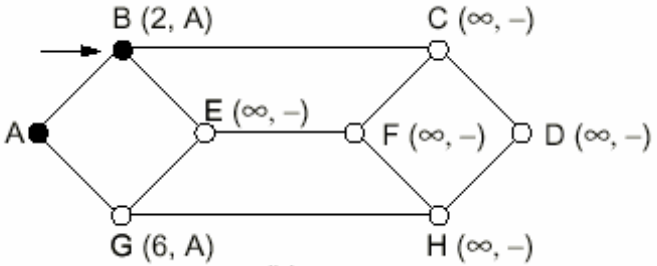
- **Link State Routing**
  - Arbeitsweise
    - Nachbarn erkennen
    - Latenz messen
    - link-state-packet (LSP) mit Nachbarn erzeugen
    - Fluten des LSP an alle (!) Router
    - Routing Table erzeugen aufgrund der LSPs



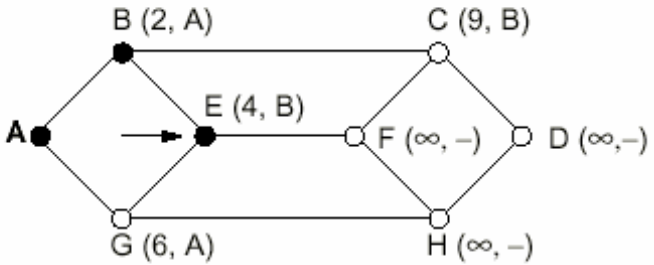
Kürzeste-Wege (Dijkstra)



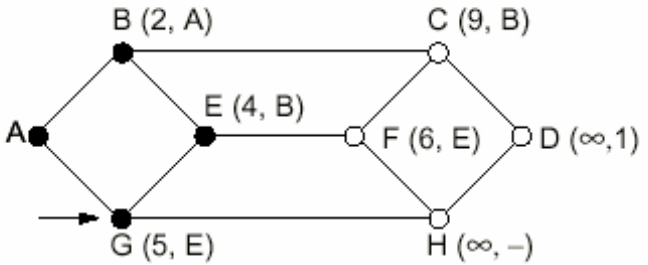
(a)



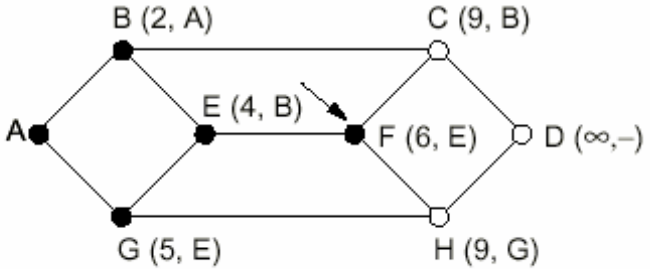
(b)



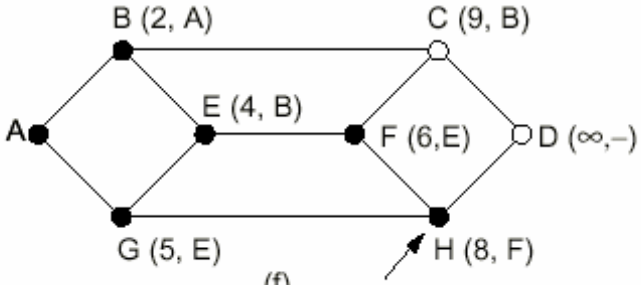
(c)



(d)



(e)



(f)

Proaktives vs. reaktives Routen

- **Proaktive Protokolle**
  - besitzen jederzeit vollständiges Wissen

  \+ tabellengesteuertes Routen

  \- viele und periodische Nachrichtenübermittlungen notwendig

- **Reaktive Protokolle**
  - Routen erst bei Bedarf gesucht (*on demand*)

  \+ Wenige Nachrichtenübermittlungen notwendig

  \- hohe Latenz

- **Auch: Hybride Ansätze**

Struktur reaktiver Protokolle

- **Reaktive Protokolle zumeist zweigeteilt:**
  - Finden von Routen (*Route Discovery*)
    - *S* will ein Paket zu *D* senden, kennt aber keinen Weg
    - Vorgehen: Zumeist Fluten
  - Pflege von Routen (*Route Maintenance*)
    - *S* kennt schon eine Route zu *D*, stellt aber fest, dass sich die Topologie signifikant geändert hat

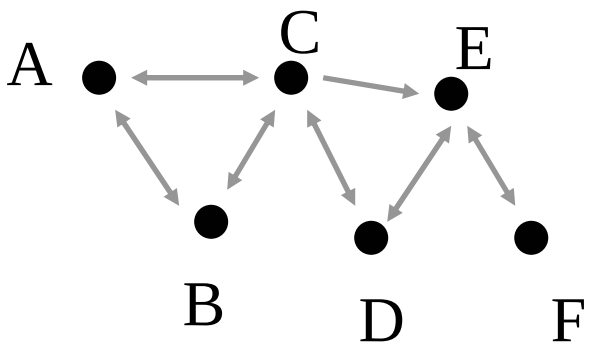
Beispiel: Dynamic Source Routing (DSR)

- **DSR ist vollständig reaktiv (*on demand*)**
- **Keinerlei periodische Nachrichten irgendwo im Netzwerk**
- **Finden von Routen (*Route Discovery*)**
  - Netzwerk geflutet
  - Jeder Hop schreibt sich in den Header
    - Kompletter Pfad entsteht im Header
- **Pflege von Routen (*Route Maintenance*)**
  - Im Fehlerfall *Route Error* (RERR) Paket an Sender zurück
  - Sender sucht neue Route
- **Senden von Paketen**
  - Quelle schreibt Pfad vor

Einschub: Fluten (am Beispiel DSR)

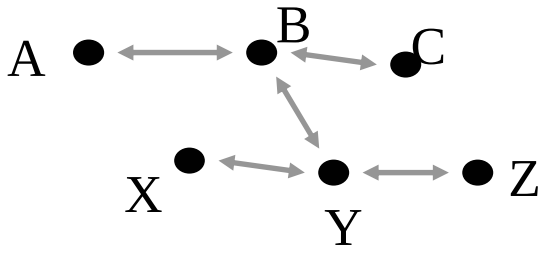
- **Fluten (*flodding*) zur Pfadsuche geeignet**
- **Sehr sinnvoll, wenn gar kein Wissen vorab existiert**
  - Sender startet Suchvorgang mit *Route Request* (RREQ)
  - Alle Empfänger leiten RREQ weiter
  - Ziel antwortet mit *Route Reply* (RREP)

- **A sucht Route nach E:**
  - A → C (AC), A → B (AB)
  - C → D (ACD), C → E (ACE), B → C (ABC)
  - D → E (ACDE), E antwortet (ACE) C verwirft (ABC)

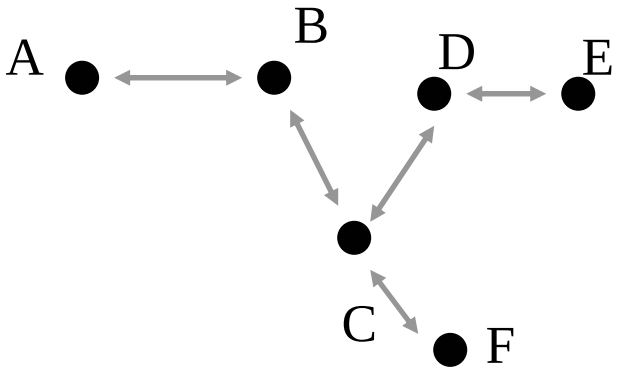


Optimierungen bei DSR (1)

- **Benutzung von Routen-Caches**
- **Cachen von „mitgehörten“ Routing-Informationen**



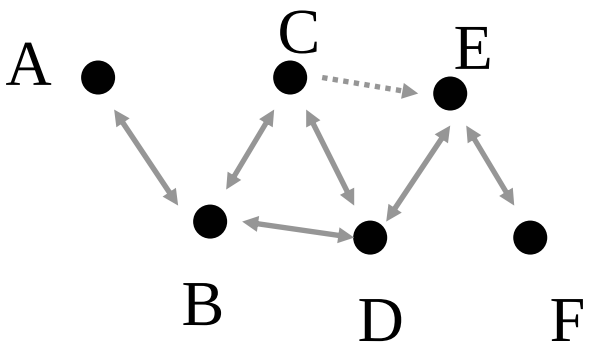
- **Beantworten von *Route Requests* nur „ohne Duplikate“**
  - Sonst „anfällige“ Routen



- **Suche in einem sich erweiternden Ring**

Optimierungen bei DSR (2)

- **„Retten“ von Routen**
  - Bei RERR testen ob „eigener“ Pfad gecached

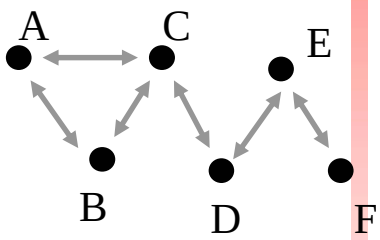


- **Automatisches Verkürzen von Routen**

- **Caching von negativen Informationen**
  - Beispiel: CE ist sehr instabil: Nach Möglichkeit vermeiden

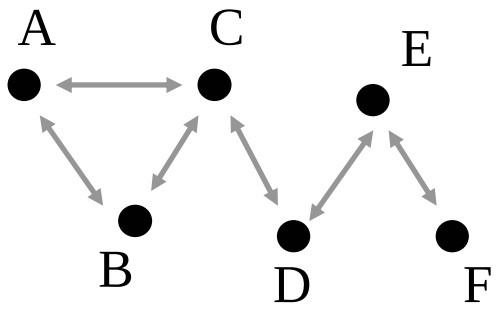
Bsp.: Ad-Hoc On-Demand Distance Vector

- **AODV ist reaktiv (*on demand*)**
  - Tabellengesteuert
  - Sequenzgesteuert („Alter“ von Routen)
  - Nur symmetrische Verbindungen berücksichtigt
- **Finden von Routen (*Route Discovery*)**
  - Netzwerk geflutet
  - Merken von „Rückrouten“ bei Empfang von RREQ
  - Antwort RREP von jedem, der Weg zum Ziel kennt
  - Falls Antwort (RREP), Vorwärtsroute etablieren
  - Falls mehrere Antworten:
    - Erste senden
    - Neuere nur, falls größere Sequenznummer, oder kleinere Hopzahl



Bsp.: AODV (2)

- **Pflege von Routen (*Route Maintenance*)**
  - Im Fehlerfall *Route Error* (RERR) Paket an Sender zurück
  - DV Eintrag entlang Route als ungültig markiert
  - Sender sucht neue Route
  - Optimierung: Lokale Reparatur

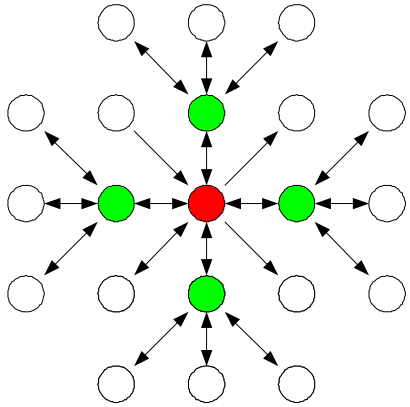


- **Senden von Paketen**
  - Anhand Tabelle (*distance vector*)

- **Weitere Erweiterungen**
  - MAODV, AOMDV, QoS

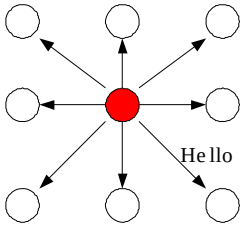
Bsp: Optimized Link-State Routing

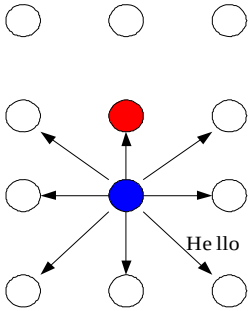
- **OLSR ist proaktiv**
  - Optimierung von LSR für wenig Bandbreite
  - Tabellengesteuert
  - Nur symmetrische Verbindungen berücksichtigt
- **Optimierung durch**
  - Auswahl von „Multipoint-Relays“ (MPR)
  - MPR so gewählt, dass alle 2-Hop-Nachbarn erreichbar



Bsp: OLSR (2)

- **Arbeitsweise Routenfindung (Route Discovery):**
  - Nachbarn erkennen
  - Multipoint-Relays (MPR) auswählen
  - Topology-Informationen versenden (fluten *nur* über MPR)
  - Routingtabellen aufbauen
- **Nachbarerkennung:**
  - Per Hello-Paket
  - Unterscheidung bidirektional und unidirektional





Weitere Forschung

- **Skalierbarkeit**
  - Wie groß kann ein Ad-Hoc Netzwerk werden?
- **Quality of Service (QoS)**
  - Audio-, Videoübertragungen?
- **Ist das Client-Server-Modell angebracht?**
  - Was ist ein Server in einem Ad-Hoc Netzwerk?
- **Sicherheit (*security*)**
  - Sicherheits-Infrastruktur passt nicht ins Modell
- **Interoperabilität mit dem Internet**
  - Wie kann ein Ad-Hoc Netzwerk sich mit dem Internet verbinden?
- **Stromsparen**
  - Wie kann die Laufzeit maximiert werden?