Routing in a Delay-Tolerant Networks

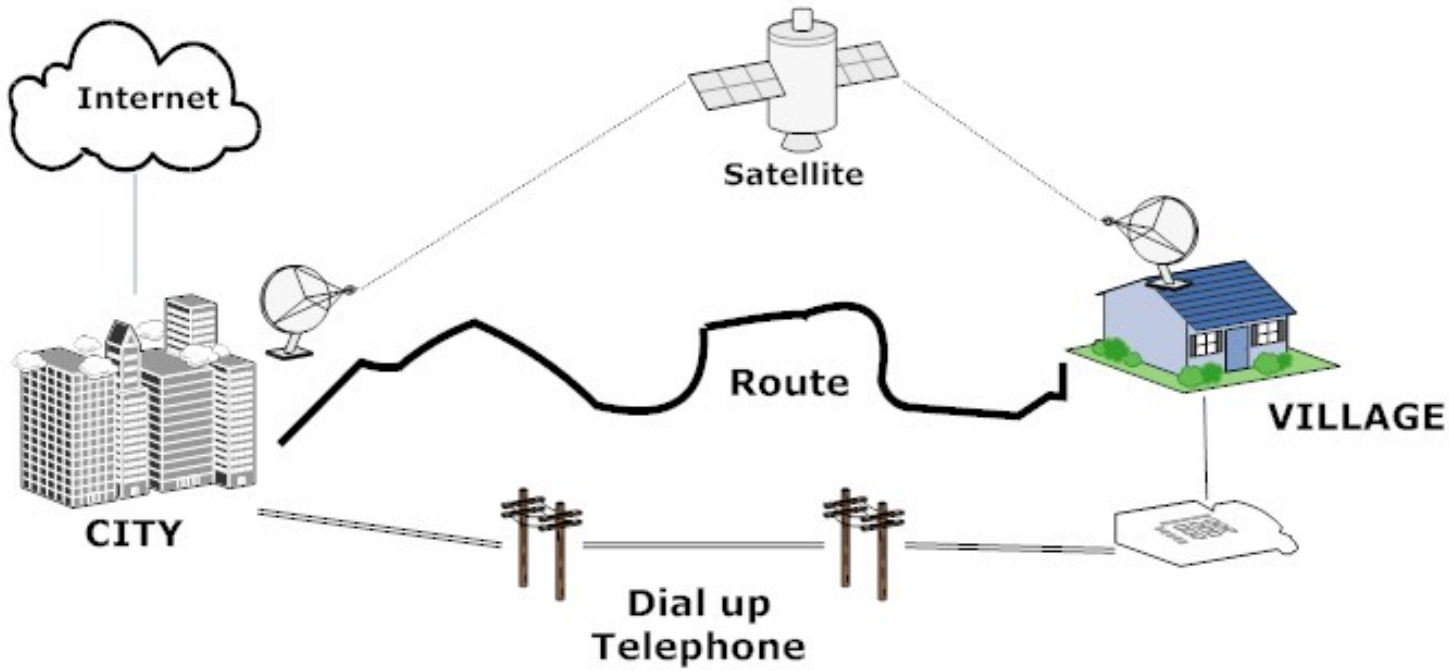
Referent: Achille Nana Tchayep

Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Seminar: Ad Hoc Networks

Motivation

Wizzy Digital Courier



Motivation

- Welche Route soll für den Transport der Daten ausgewählt werden?
- Und vor allem wann soll dies geschehen?
- Eine End-to-End Verbindung ist nicht immer vorhanden
- Die Wahl der Route hängt von mehreren Faktoren ab: die Quelle, das Ziel, die Größe der Daten, die Zeit des Datenversands und die Erreichbarkeit der Zwischenknoten.

Motivation

- Welche Route soll für den Transport der Daten ausgewählt werden?
- Und vor allem wann soll dies geschehen?
- Eine End-to-End Verbindung ist nicht immer vorhanden
- Die Wahl der Route hängt von mehreren Faktoren ab: die Quelle, das Ziel, die Größe der Daten, die Zeit des Datenversands und die Erreichbarkeit der Zwischenknoten.

Motivation

- Welche Route soll für den Transport der Daten ausgewählt werden?
- Und vor allem wann soll dies geschehen?
- Eine End-to-End Verbindung ist nicht immer vorhanden

- Die Wahl der Route hängt von mehreren Faktoren ab: die Quelle, das Ziel, die Größe der Daten, die Zeit des Datenversands und die Erreichbarkeit der Zwischenknoten.

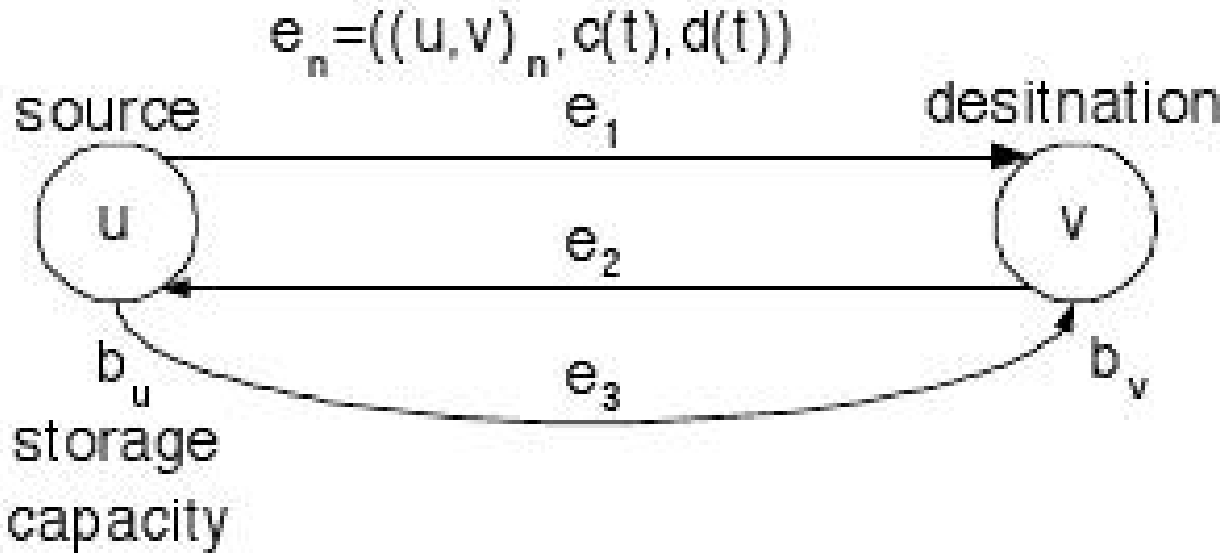
Motivation

- Welche Route soll für den Transport der Daten ausgewählt werden?
- Und vor allem wann soll dies geschehen?
- Eine End-to-End Verbindung ist nicht immer vorhanden

- Die Wahl der Route hängt von mehreren Faktoren ab: die Quelle, das Ziel, die Größe der Daten, die Zeit des Datenversands und die Erreichbarkeit der Zwischenknoten.

Gliederung

Knoten in DTN



Die Ziele

Eine Nachricht K in DTN ist ein geordneter Tupel (u,v,t,m)

- Minimierung der Anzahl der Datenübertragungen und dabei auch die Datenkopie. Also Buffer- und Kantenkapazität sollen nicht überschritten werden.
- Minimierung des Datenverlustes während der Datenübertragungen
- Minimierung der Verzögerungszeit.

- Es soll der beste Weg gefunden werden, um die Anzahl der erfolgreich ankommenden Datenpakete zu maximieren.

Die Ziele

Eine Nachricht K in DTN ist ein geordneter Tupel (u,v,t,m)

- Minimierung der Anzahl der Datenübertragungen und dabei auch die Datenkopie. Also Buffer- und Kantenkapazität sollen nicht überschritten werden.
- Minimierung des Datenverlustes während der Datenübertragungen
- Minimierung der Verzögerungszeit.
- Es soll der beste Weg gefunden werden, um die Anzahl der erfolgreich ankommenden Datenpakete zu maximieren.

Die Ziele

Eine Nachricht K in DTN ist ein geordneter Tupel (u,v,t,m)

- Minimierung der Anzahl der Datenübertragungen und dabei auch die Datenkopie. Also Buffer- und Kantenkapazität sollen nicht überschritten werden.
- Minimierung des Datenverlustes während der Datenübertragungen
- Minimierung der Verzögerungszeit.
- Es soll der beste Weg gefunden werden, um die Anzahl der erfolgreich ankommenden Datenpakete zu maximieren.

Die Ziele

Eine Nachricht K in DTN ist ein geordneter Tupel (u,v,t,m)

- Minimierung der Anzahl der Datenübertragungen und dabei auch die Datenkopie. Also Buffer- und Kantenkapazität sollen nicht überschritten werden.
- Minimierung des Datenverlustes während der Datenübertragungen
- Minimierung der Verzögerungszeit.

- Es soll der beste Weg gefunden werden, um die Anzahl der erfolgreich ankommenden Datenpakete zu maximieren.

Die Ziele

Eine Nachricht K in DTN ist ein geordneter Tupel (u,v,t,m)

- Minimierung der Anzahl der Datenübertragungen und dabei auch die Datenkopie. Also Buffer- und Kantenkapazität sollen nicht überschritten werden.
- Minimierung des Datenverlustes während der Datenübertragungen
- Minimierung der Verzögerungszeit.

- Es soll der beste Weg gefunden werden, um die Anzahl der erfolgreich ankommenden Datenpakete zu maximieren.

Knowledge Oracles: Definition

Knowledge Oracle: Abstrakte Elemente, die Informationen über das gesamte Netz verfügen und sie sind somit für die Routing-Entscheidung wichtig.

1. *Contacts Summary Oracle*: Er gibt Informationen über die durchschnittliche Wartezeit bis zum nächsten Nachbarknoten.
2. *Contacts Oracle*: Zwischen 2 Knoten gibt er Informationen über die dazwischen liegenden Knoten.
3. *Queuing Oracle*: Er verfügt jeder Zeit über Informationen der Bufferbelegung jedes Knotens.
4. *Traffic Demand Oracle*: Er ermittelt Informationen über den aktuellen und den zukünftigen Datenverkehrszustand im Netz.

Knowledge Oracles: Definition

Knowledge Oracle: Abstrakte Elemente, die Informationen über das gesamte Netz verfügen und sie sind somit für die Routing-Entscheidung wichtig.

1. *Contacts Summary Oracle*: Er gibt Informationen über die durchschnittliche Wartezeit bis zum nächsten Nachbarknoten.
2. *Contacts Oracle*: Zwischen 2 Knoten gibt er Informationen über die dazwischen liegenden Knoten.
3. *Queuing Oracle*: Er verfügt jeder Zeit über Informationen der Bufferbelegung jedes Knotens.
4. *Traffic Demand Oracle*: Er ermittelt Informationen über den aktuellen und den zukünftigen Datenverkehrszustand im Netz.

Knowledge Oracles: Definition

Knowledge Oracle: Abstrakte Elemente, die Informationen über das gesamte Netz verfügen und sie sind somit für die Routing-Entscheidung wichtig.

1. *Contacts Summary Oracle*: Er gibt Informationen über die durchschnittliche Wartezeit bis zum nächsten Nachbarknoten.
2. *Contacts Oracle*: Zwischen 2 Knoten gibt er Informationen über die dazwischen liegenden Knoten.
3. *Queuing Oracle*: Er verfügt jeder Zeit über Informationen der Bufferbelegung jedes Knotens.
4. *Traffic Demand Oracle*: Er ermittelt Informationen über den aktuellen und den zukünftigen Datenverkehrszustand im Netz.

Knowledge Oracles: Definition

Knowledge Oracle: Abstrakte Elemente, die Informationen über das gesamte Netz verfügen und sie sind somit für die Routing-Entscheidung wichtig.

1. *Contacts Summary Oracle*: Er gibt Informationen über die durchschnittliche Wartezeit bis zum nächsten Nachbarknoten.
2. *Contacts Oracle*: Zwischen 2 Knoten gibt er Informationen über die dazwischen liegenden Knoten.
3. *Queuing Oracle*: Er verfügt jeder Zeit über Informationen der Bufferbelegung jedes Knotens.
4. *Traffic Demand Oracle*: Er ermittelt Informationen über den aktuellen und den zukünftigen Datenverkehrszustand im Netz.

Knowledge Oracles: Definition

Knowledge Oracle: Abstrakte Elemente, die Informationen über das gesamte Netz verfügen und sie sind somit für die Routing-Entscheidung wichtig.

1. *Contacts Summary Oracle*: Er gibt Informationen über die durchschnittliche Wartezeit bis zum nächsten Nachbarknoten.
2. *Contacts Oracle*: Zwischen 2 Knoten gibt er Informationen über die dazwischen liegenden Knoten.
3. *Queuing Oracle*: Er verfügt jeder Zeit über Informationen der Bufferbelegung jedes Knotens.
4. *Traffic Demand Oracle*: Er ermittelt Informationen über den aktuellen und den zukünftigen Datenverkehrszustand im Netz.

Routing-Klasse

- *Zero Knowledge*:Die Algorithmen in dieser Klasse benutzen keine *Oracles*.
- *Totale knowledge*:Hier werden alle *Oracles* (*contacts, queuing, traffic demand*) verwendet.
- *Partiel knowledge*:Bei Algorithmen dieser Klasse fällt der *traffic demand oracle* aus.

Routing-Klasse

- *Zero Knowledge*:Die Algorithmen in dieser Klasse benutzen keine *Oracles*.
- *Totale knowledge*:Hier werden alle *Oracles* (*contacts, queuing, traffic demand*) verwendet.
- *Partiel knowledge*:Bei Algorithmen dieser Klasse fällt der *traffic demand oracle* aus.

Routing-Klasse

- *Zero Knowledge*:Die Algorithmen in dieser Klasse benutzen keine *Oracles*.
- *Totale knowledge*:Hier werden alle *Oracles* (*contacts, queuing, traffic demand*) verwendet.
- *Partiel knowledge*:Bei Algorithmen dieser Klasse fällt der *traffic demand oracle* aus.

Routing-Klasse

- *Zero Knowledge*:Die Algorithmen in dieser Klasse benutzen keine *Oracles*.
- *Totale knowledge*:Hier werden alle *Oracles* (*contacts, queuing, traffic demand*) verwendet.
- *Partiel knowledge*:Bei Algorithmen dieser Klasse fällt der *traffic demand oracle* aus.

Routing mit keinen Kenntnissen

- *First Contact* (**FC**)
  - Die Nachricht wird auf die nächst zufällig ausgewählten Kanten weitergeleitet.
  - Falls keine Kanten erreichbar sind, wartet er bis eine erreichbar wird und leitet dann die Nachricht weiter.
  - Einfach zu implementieren.

Routing mit keinen Kenntnissen

- *First Contact* (**FC**)
  - Die Nachricht wird auf die nächst zufällig ausgewählten Kanten weitergeleitet.
  - Falls keine Kanten erreichbar sind, wartet er bis eine erreichbar wird und leitet dann die Nachricht weiter.
  - Einfach zu implementieren.

Routing mit keinen Kenntnissen

- *First Contact* (**FC**)
  - Die Nachricht wird auf die nächst zufällig ausgewählten Kanten weitergeleitet.
  - Falls keine Kanten erreichbar sind, wartet er bis eine erreichbar wird und leitet dann die Nachricht weiter.
  - Einfach zu implementieren.

Routing mit keinen Kenntnissen

- *First Contact* (**FC**)
  - Die Nachricht wird auf die nächst zufällig ausgewählten Kanten weitergeleitet.
  - Falls keine Kanten erreichbar sind, wartet er bis eine erreichbar wird und leitet dann die Nachricht weiter.
  - Einfach zu implementieren.

Routing mit keinen Kenntnissen

- *First Contact* (**FC**)
  - Die Nachricht wird auf die nächst zufällig ausgewählten Kanten weitergeleitet.
  - Falls keine Kanten erreichbar sind, wartet er bis eine erreichbar wird und leitet dann die Nachricht weiter.
  - Einfach zu implementieren.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).

- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).
- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).

- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).

- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).

- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).

- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (1)

- **Minimum Expected Delay** (**MED**): *Contacts Summary*
  - Die Routenberechnung ist zeitunanhängig (*proactive routing*).
  - Kantenkosten = avg(propagation+transmission delay).

- **Earliest Delivery** (**ED**):*Contacts*
  - Die Route wird an der Quelle berechnet und festgelegt (*source routing*).
  - Die Routeberechnung wird ohne Rücksicht auf die Bufferkapazität der Zwischenknoten durchgeführt.
  - kann zum *Buffer oberflow* führen.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit partiellen Kenntnissen (2)

- *Earliest Delivery with Local Queuing* (**EDLQ**): *Contacts*
  - Etwa identsich wie beim **ED**
  - Die Speicherkapazität der dazwischen liegenden Knoten werden in Betracht gezogen.
  - Die Route wird bei jedem erreichbaren Knoten neu berechnet (*per-hop routing*).

- **Earliest Delivry with All Queues**(**EDAQ**): *Contacts,Queuing*
  - Funktioniert wie beim *ED*.
  - die Speicherbelegung der Zwischenknoten wird bei der Routenberechnung in Betracht gezogen.
  - Nach der Routenberechnung wird der Speicher auf jedem Knoten reserviert.

Routing mit totalen Kenntnissen

- *Linear Programming*(**LP**): *Contacts, Queuing, traffic*
  - Der perfekte Algorithm.
  - Er verfügt über alle Kenntnisse über das Netz, denn er nutzt alle *knowledge Oracle* aus.

Routing mit totalen Kenntnissen

- *Linear Programming*(**LP**): *Contacts, Queuing, traffic*
  - Der perfekte Algorithm.
  - Er verfügt über alle Kenntnisse über das Netz, denn er nutzt alle *knowledge Oracle* aus.

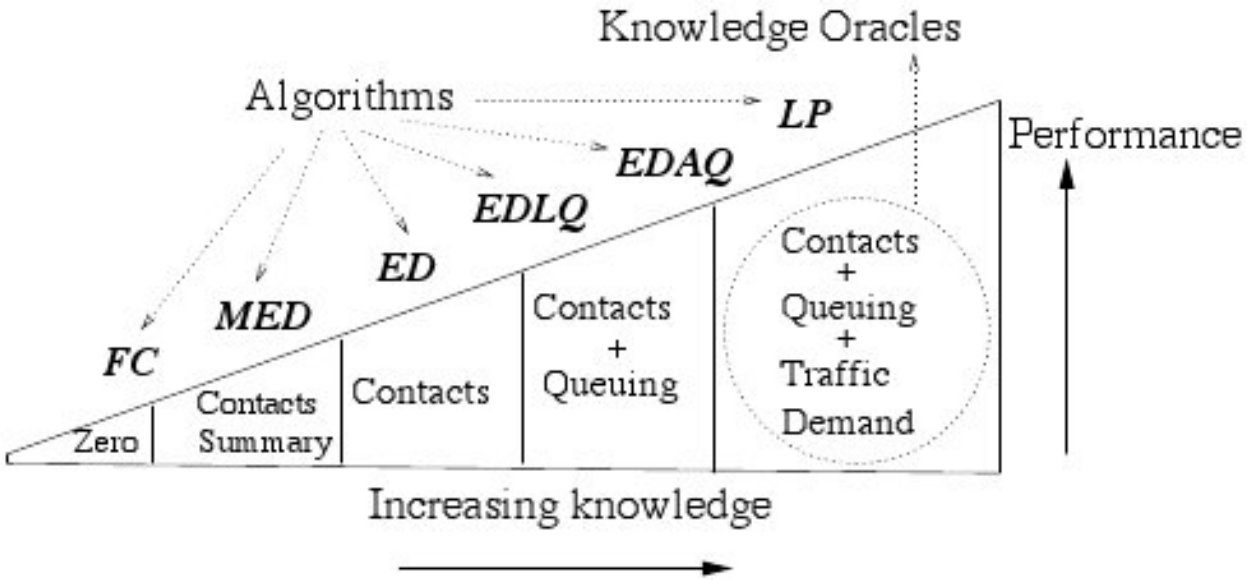
Routing mit totalen Kenntnissen

- *Linear Programming*(**LP**): *Contacts, Queuing, traffic*
  - Der perfekte Algorithm.
  - Er verfügt über alle Kenntnisse über das Netz, denn er nutzt alle *knowledge Oracle* aus.

Routing mit totalen Kenntnissen

- *Linear Programming*(**LP**): *Contacts, Queuing, traffic*
  - Der perfekte Algorithm.
  - Er verfügt über alle Kenntnisse über das Netz, denn er nutzt alle *knowledge Oracle* aus.

Performance



Kostenberechnung

1. Jeder Verbindung wird ein Kosten zugewiesen
2. Berechnung der minimalen Kosten.

Kostenberechnung

1. Jeder Verbindung wird ein Kosten zugewiesen
2. Berechnung der minimalen Kosten.

Kostenberechnung

1. Jeder Verbindung wird ein Kosten zugewiesen
2. Berechnung der minimalen Kosten.

Modifizierter Dijkstras Algorithmus

Input: $G = (V, E)$, s, T, $w(e, t)$
Output: L

1: $Q \leftarrow \{V\}$
2: $\mathrm{L}[s] \leftarrow 0$, $\mathrm{L}[v] \leftarrow \infty \ \forall\ v \in V \ s.t\ v \neq s.$
3: **while** $Q \neq \{\}$ **do**
4: Let $u \in Q$ be the node s.t $\mathrm{L}[u] = \min_{x \in Q} \mathrm{L}[x]$
5: $Q = Q \leftarrow \{u\}$
6: **for** each edge $e \in E$, s.t. $e = (u, v)$ **do**
7: **if** $\mathrm{L}[v] > (\mathrm{L}[u] + w(e, \mathrm{L}[u] + T))$ **then**
8: $\mathrm{L}[v] \leftarrow \mathrm{L}[u] + w(e, \mathrm{L}[u] + T)$
9: **end if**
10: **end for**
11: **end while**

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

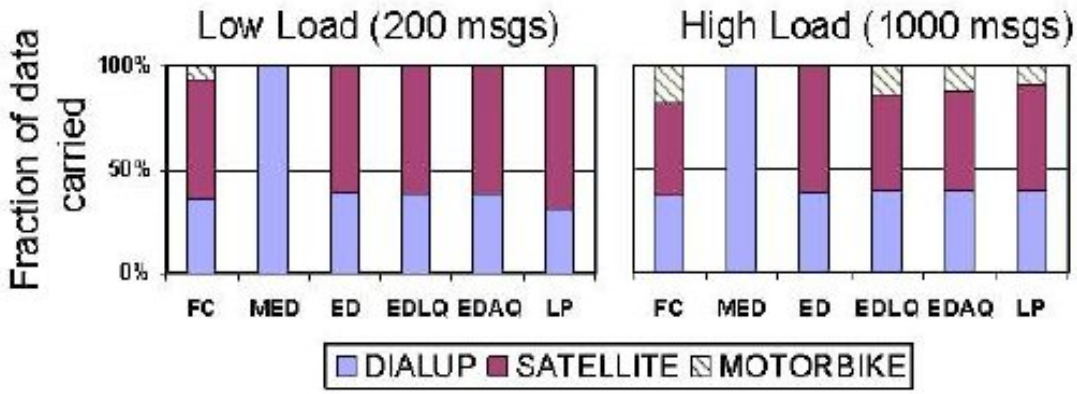
Simulation

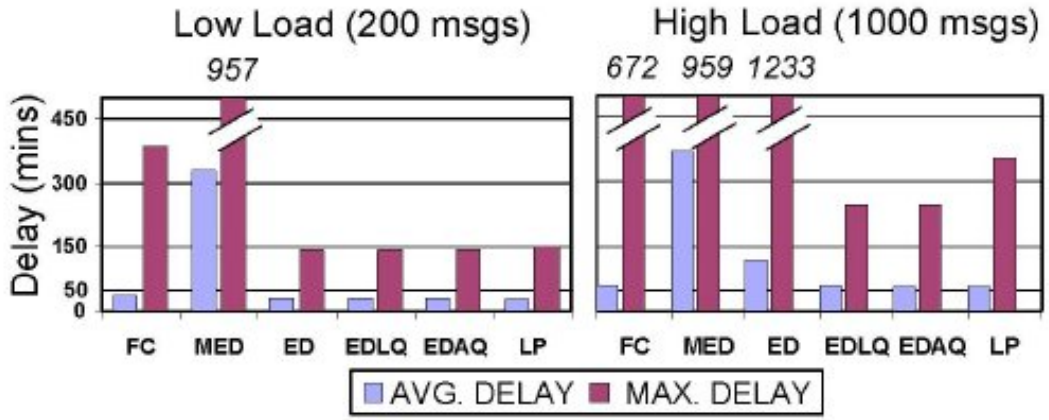
- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Simulation

- Ein JAVA-basierter Simulator
- Dialup: 4kbps(Abends 23:00 - 06:00)
- Satellites(x2): 4-5, Mal/Tag sichtbar während 10Min., 10kbps
- Motoradfahrer(x3): 2Std/Tag, 1Mbps, 128MB Speicherkapazität, Kontaktzeit 5Min
- Nachricht Village to City 1KB
- Nachricht City to Village 10KB
- 2 Datenverkehrsraten 200(*low load*) und 1000(*high load*) Nachrichten Village
- Die Nachtichten werden zu unterschiedlichen Zeitpunkten ins Netz injektiert. (Begin 00:00Uhr)

Ergebnisse


Low Load (200 msgs)
High Load (1000 msgs)
Fraction of data carried
100%
50%
0%
FC MED ED EDLQ EDAQ LP
FC MED ED EDLQ EDAQ LP
DIALUP SATELLITE MOTORBIKE


Low Load (200 msgs)
High Load (1000 msgs)
957
672 959 1233
Delay (mins)
450
300
150
50
0
FC MED ED EDLQ EDAQ LP
FC MED ED EDLQ EDAQ LP
AVG. DELAY MAX. DELAY

Beobachtungen

- **MED**: Dialup wird immer benutzt sowohl beim *high load* als auch beim *low load*. Also er hat die beste durchschnittlische Verzögerungszeit.
- **ED**: 60% der Daten werden per Satellites geroutet und der Rest per Díalup.
- **FC**: Daten werden manchmal per Motoradfahrer geroutet. Dies erklärt die größere Verzögerungszeit als beim **EDAQ**.
- **EDAQ**/**EDLQ** sind identisch in *low load*.

Beobachtungen

- **MED**: Dialup wird immer benutzt sowohl beim *high load* als auch beim *low load*. Also er hat die beste durchschnittlische Verzögerungszeit.
- **ED**: 60% der Daten werden per Satellites geroutet und der Rest per Díalup.
- **FC**: Daten werden manchmal per Motoradfahrer geroutet. Dies erklärt die größere Verzögerungszeit als beim **EDAQ**.
- **EDAQ**/**EDLQ** sind identisch in *low load*.

Beobachtungen

- **MED**: Dialup wird immer benutzt sowohl beim *high load* als auch beim *low load*. Also er hat die beste durchschnittlische Verzögerungszeit.
- **ED**: 60% der Daten werden per Satellites geroutet und der Rest per Díalup.
- **FC**: Daten werden manchmal per Motoradfahrer geroutet. Dies erklärt die größere Verzögerungszeit als beim **EDAQ**.
- **EDAQ**/**EDLQ** sind identisch in *low load*.

Beobachtungen

- **MED**: Dialup wird immer benutzt sowohl beim *high load* als auch beim *low load*. Also er hat die beste durchschnittlische Verzögerungszeit.
- **ED**: 60% der Daten werden per Satellites geroutet und der Rest per Díalup.
- **FC**: Daten werden manchmal per Motoradfahrer geroutet. Dies erklärt die größere Verzögerungszeit als beim **EDAQ**.
- **EDAQ**/**EDLQ** sind identisch in *low load*.

Zusammenfassung

- **DTN** sind Netzwerke, die große Verzögerungszeit ertragen können, da die Verbindung zwischen den Knoten zeitweilig ist.
- Die Algorithmen, die wenig Kenntnisse über den Dynamismus des Netzes verfügen, haben eine schwache Leistung.
- Generell ist es nicht trivial Algorithmen im DTN-Netz zu implementieren, denn die Topologie des Netzes sehr dynamisch ist.

- Der nächste Challenge wäre die Implementierung der *Oracles*.
- Wie wäre es, wenn die *Oracles* nicht die Wahrheit sagen würde?

Zusammenfassung

- **DTN** sind Netzwerke, die große Verzögerungszeit ertragen können, da die Verbindung zwischen den Knoten zeitweilig ist.
- Die Algorithmen, die wenig Kenntnisse über den Dynamismus des Netzes verfügen, haben eine schwache Leistung.
- Generell ist es nicht trivial Algorithmen im DTN-Netz zu implementieren, denn die Topologie des Netzes sehr dynamisch ist.

- Der nächste Challenge wäre die Implementierung der *Oracles*.
- Wie wäre es, wenn die *Oracles* nicht die Wahrheit sagen würde?

Zusammenfassung

- **DTN** sind Netzwerke, die große Verzögerungszeit ertragen können, da die Verbindung zwischen den Knoten zeitweilig ist.
- Die Algorithmen, die wenig Kenntnisse über den Dynamismus des Netzes verfügen, haben eine schwache Leistung.
- Generell ist es nicht trivial Algorithmen im DTN-Netz zu implementieren, denn die Topologie des Netzes sehr dynamisch ist.

- Der nächste Challenge wäre die Implementierung der *Oracles*.
- Wie wäre es, wenn die *Oracles* nicht die Wahrheit sagen würde?

Zusammenfassung

- **DTN** sind Netzwerke, die große Verzögerungszeit ertragen können, da die Verbindung zwischen den Knoten zeitweilig ist.
- Die Algorithmen, die wenig Kenntnisse über den Dynamismus des Netzes verfügen, haben eine schwache Leistung.
- Generell ist es nicht trivial Algorithmen im DTN-Netz zu implementieren, denn die Topologie des Netzes sehr dynamisch ist.

- Der nächste Challenge wäre die Implementierung der *Oracles*.
- Wie wäre es, wenn die *Oracles* nicht die Wahrheit sagen würde?

Zusammenfassung

- **DTN** sind Netzwerke, die große Verzögerungszeit ertragen können, da die Verbindung zwischen den Knoten zeitweilig ist.
- Die Algorithmen, die wenig Kenntnisse über den Dynamismus des Netzes verfügen, haben eine schwache Leistung.
- Generell ist es nicht trivial Algorithmen im DTN-Netz zu implementieren, denn die Topologie des Netzes sehr dynamisch ist.

- Der nächste Challenge wäre die Implementierung der *Oracles*.
- Wie wäre es, wenn die *Oracles* nicht die Wahrheit sagen würde?

Literatur

1. Jean Patrick Gelas. Reseaux Interplanetaires et Reseaux tolérants au delais, 2003.
2. http://www.dtnrg.org/. Delay Tolerant Networking Research Group.
3. http://www.wizzy.org.za. Wizzy Digital Courier, 2003.
4. Tayeb LEMLOUMA. Routage dans les reseaux mobile Ad Hoc.
5. R. Patra S. Jain, K. Fall. Routing in a Delay Tolerant Network, 2004.
6. Forrest Warthman. Delay Tolerant Notworks (DTNs), 2003.
7. Wikipedia. http://de.wikipedia.org.

Ich bedanke mich für eure Aufmerksamkeit!