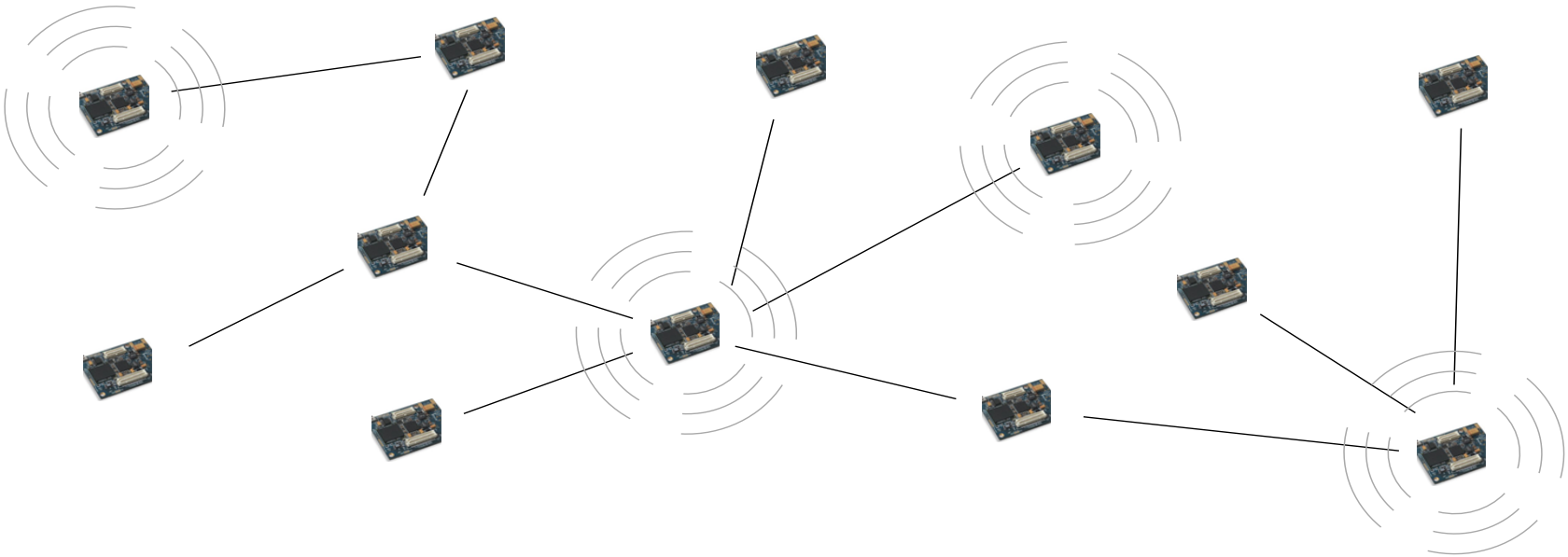
Karlsruher Institut für Technologie

Algorithmen für Ad-hoc- und Sensornetze

VL 08 – Data Gathering mit Network Coding

Fabian Fuchs | 26. Nov. 2015 (Version 2)

INSTITUT FÜR THEORETISCHE INFORMATIK - LEHRSTUHL FÜR ALGORITHMIK (PROF. WAGNER)



Überblick

- Organisatorisches
- Einleitung
  - Ursprung in drahtgebundenen Netzen
  - Anwendung in drahtlosen Netzen
- Data Gathering + Network Coding
  - Vergleich zur letzten Vorlesung
  - Zwei Ansätze zur Minimierung der Energie

Verbleibende Termine

Montag	Dienstag	Mittwoch	Donnerstag	Freitag
	(1)	2	3	4
7	8	9	10	11
14	15	16	17	18
21	(22)	23	24	25
28	29	30	31	

Montag	Dienstag	Mittwoch	Donnerstag	Freitag
				1
4	5	6	7	8
11	12	13	14	15
18	19	20	21	22

Alternativen: Dienstags-Termin

Stunde	Montag	Dienstag	Mittwoch	Donnerstag	Freitag
1 8:00-9:30					
2 9:45-11:15	außer 30.11				
3 11:30-13:00					
4 14:00-15:30		außer 1.12			
5 15:45-17:15				VL	
6 17:30-19:00	außer 30.11				

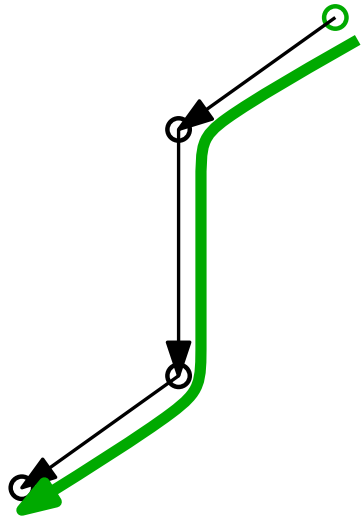
Zu verschiebende Termine: (01.12), 08.12, (22.12), 19.01, (26.01)

Überblick

- Organisatorisches
- Einleitung
  - Ursprung in drahtgebundenen Netzen
  - Anwendung in drahtlosen Netzen
- Data Gathering + Network Coding
  - Vergleich zur letzten Vorlesung
  - Zwei Ansätze zur Minimierung der Energie

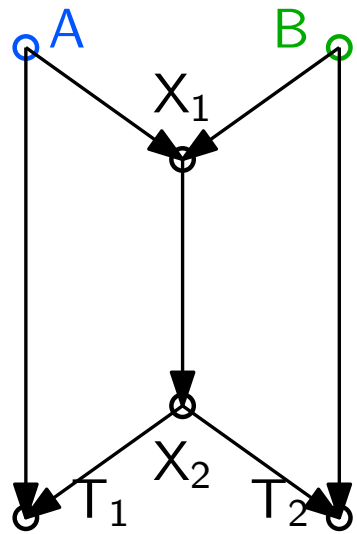
Informationsaustausch bisher..

- Daten werden von (einem/vielen) Knoten an (einen/viele) andere geschickt.
- Pakete werden
  - abgeschickt
  - weitergeleitet
  - empfangen



Informationsaustausch bisher..

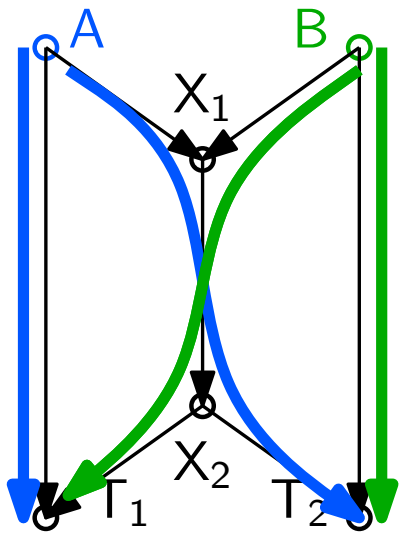
- Daten werden von (einem/vielen) Knoten an (einen/viele) andere geschickt.
- Pakete werden
  - abgeschickt
  - weitergeleitet
  - empfangen



Wenn A und B jeweils N Pakete für T_1 und T_2 haben, wie viele Zeitschritte dauert es, diese zuzustellen, wenn über jede gerichtete Kante in jeder Zeiteinheit ein Paket übertragen werden kann?

Informationsaustausch bisher..

- Daten werden von (einem/vielen) Knoten an (einen/viele) andere geschickt.
- Pakete werden
  - abgeschickt
  - weitergeleitet
  - empfangen



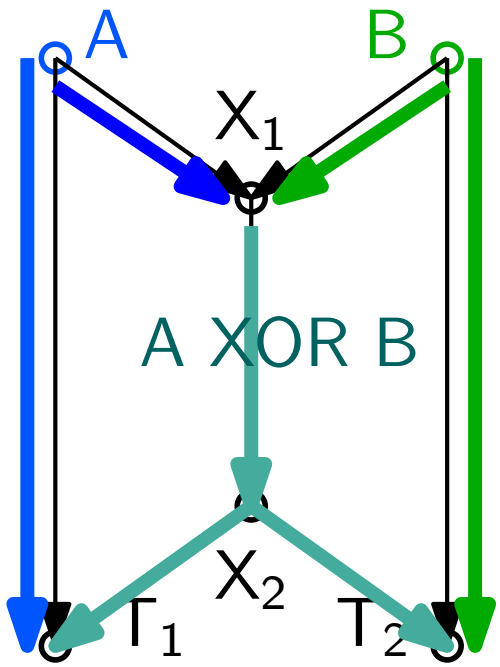
Wenn A und B jeweils N Pakete für T_1 und T_2 haben, wie viele Zeitschritte dauert es, diese zuzustellen, wenn über jede gerichtete Kante in jeder Zeiteinheit ein Paket übertragen werden kann?

- Klassische Antwort: etwas über 2N Zeitschritte!
  - Mittlere Kante muss jedes Paket einmal übertragen!

..und mit Network Coding

Das geht besser!

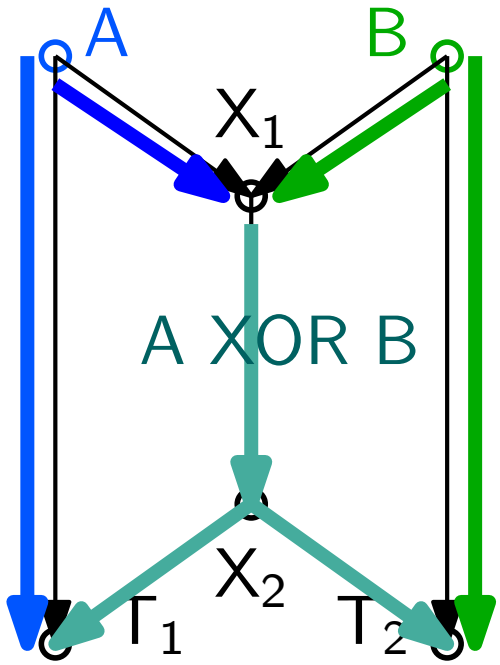
- A und B schicken nacheinander alle Pakete an X_1 und an T_1 bzw. T_2
- X_1 verknüpft ankommende Pakete von A und B per *XOR* und schickt sie an X_2
- X_2 schickt die XOR-Pakete an T_1 und T_2
- fertig in $N + 3$ Zeitschritten!



..und mit Network Coding

Das geht besser!

- A und B schicken nacheinander alle Pakete an X_1 und an T_1 bzw. T_2
- X_1 verknüpft ankommende Pakete von A und B per *XOR* und schickt sie an X_2
- X_2 schickt die XOR-Pakete an T_1 und T_2
- fertig in $N + 3$ Zeitschritten!

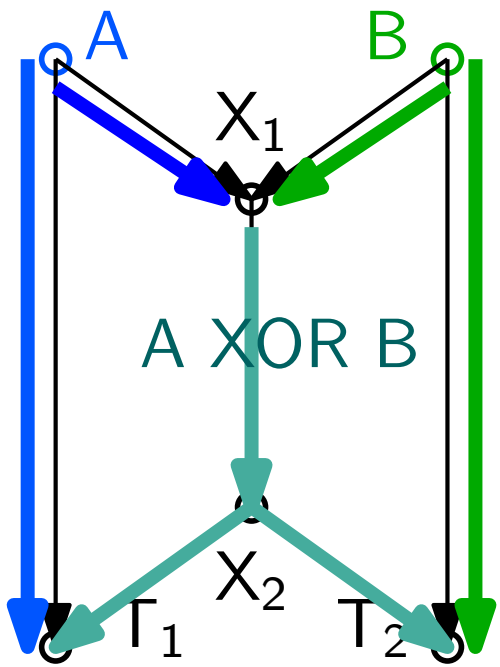


Klassisches Beispiel: Vermischen von Datenflüssen erhöht Durchsatz in drahtgebundenen Netzen!

Network Coding

Definition

Network Coding bezeichnet alle Techniken, bei denen die Pakete verschiedener Informationsströme miteinander vermischt werden.

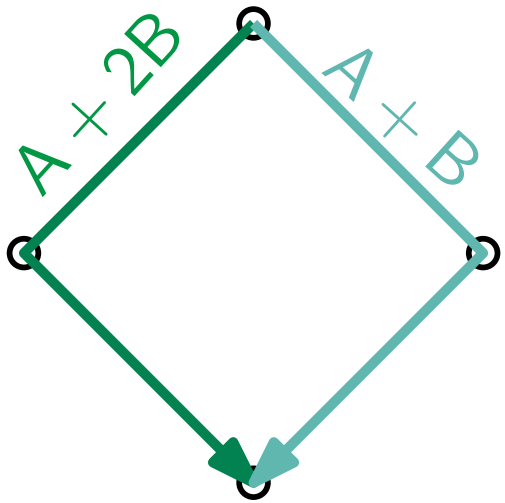


Network Coding

Definition

Network Coding bezeichnet alle Techniken, bei denen die Pakete verschiedener Informationsströme miteinander vermischt werden.

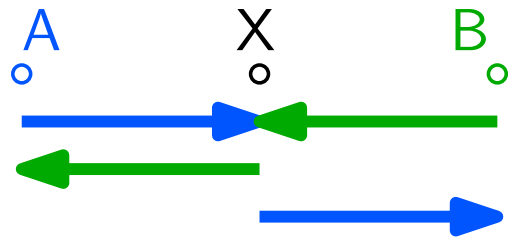
- Konsequenz: einzelne Pakete werden in der Regel wertlos!
  - das kann auch ein Feature sein!
  - Angreifer müssen mehrere Datenströme abgreifen



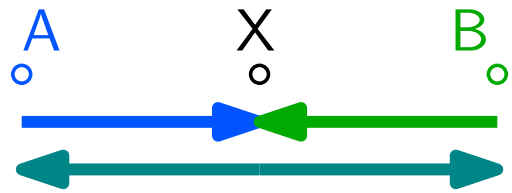
Network Coding

Definition

Network Coding bezeichnet alle Techniken, bei denen die Pakete verschiedener Informationsströme miteinander vermischt werden.



- Konsequenz: einzelne Pakete werden in der Regel wertlos!
  - das kann auch ein Feature sein!
  - Angreifer müssen mehrere Datenströme abgreifen
- Hilft das auch in drahtlosen Netzen?
  - Ja, das kann Übertragungen sparen!
  - Beispiel: Austausch zwischen zwei Knoten



Überblick

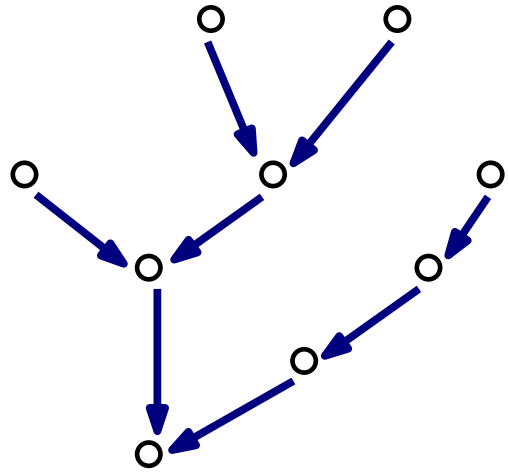
- Organisatorisches
- Einleitung
  - Ursprung in drahtgebundenen Netzen
  - Anwendung in drahtlosen Netzen
- Data Gathering + Network Coding
  - Vergleich zur letzten Vorlesung
  - Zwei Ansätze zur Minimierung der Energie

Data Gathering (DG) bisher...

I) (Fast) vollständige Aggregation

Jeder Knoten empfängt Pakete seiner Kinder und sendet *ein* Paket Richtung Senke

- Distributive, algebraische Funktionen



Data Gathering (DG) bisher...

I) (Fast) vollständige Aggregation

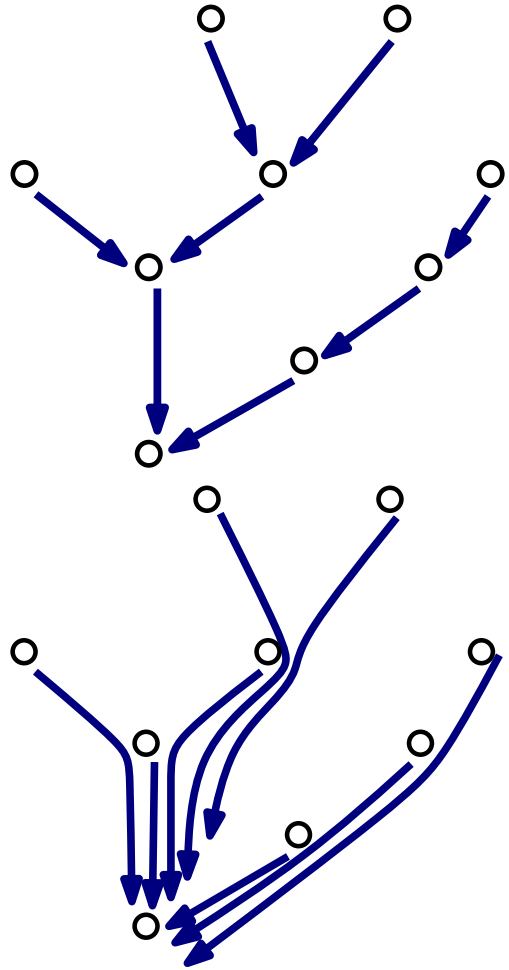
Jeder Knoten empfängt Pakete seiner Kinder und sendet *ein* Paket Richtung Senke

- Distributive, algebraische Funktionen

II) Data Gathering ohne Aggregation

Jeder Knoten schickt seine Pakete ab und leitet Pakete aus gesamtem Teilbaum weiter.

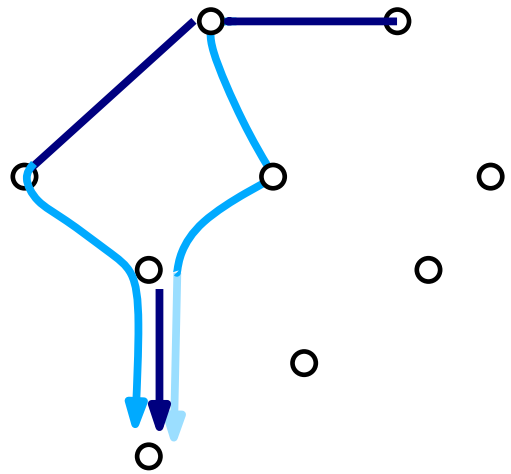
- typisch für wirklich komplexe Anfragen oder dann, wenn alle Daten zur Analyse benötigt werden.



...und jetzt viel allgemeiner

III) Network Coding

Jeder Knoten v_i hat Paket p_i mit $|p_i|$ Bits. Er darf es verschicken, Knoten können es umkodieren, am Ende muss die Senke das Paket rekonstruieren können.



- vollständige Aggregation fällt eigentl. nicht darunter
  - Senke erhält nicht alle Daten
- DG ohne Aggregation ist Sonderfall
  - Pakete werden nie umkodiert und auf Baum zur Senke geroutet
- wir betrachten das Problem noch allgemeiner, nämlich nicht nur beschränkt auf Bäume!

Optimale Topologien

Erinnerung

Wir sind bei der Aggregation und dem Einsammeln von Daten bisher davon ausgegangen, dass ein Baum gegeben ist, auf dem die Pakete zur Senke geroutet werden.

Optimale Topologien

Erinnerung

Wir sind bei der Aggregation und dem Einsammeln von Daten bisher davon ausgegangen, dass ein Baum gegeben ist, auf dem die Pakete zur Senke geroutet werden.

Was, wenn wir nur einen Graphen vorgeben und ganz individuell für Pakete angeben können, wie sie weitergereicht werden und wo sie umkodiert werden sollen?

- Kosten c_e für jede Kante e pro übertragenem Bit gegeben
- Ziel: Minimale Gesamtkosten

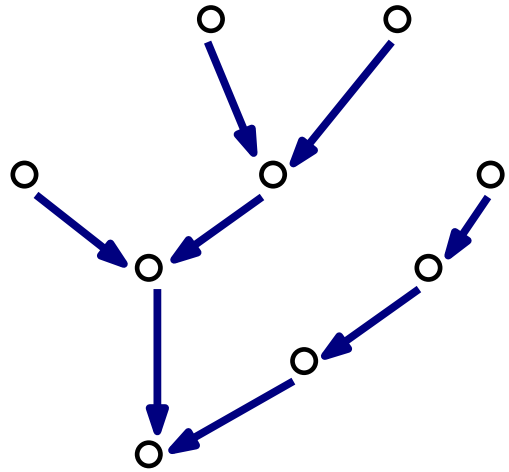
$$\sum_{e} c_e \sum_{p \text{ nutzt } e} |p|$$

Optimale Topologien (Warm-up I)

Beobachtung I

Bei vollständiger Aggregation und uniformen Paketgrößen ist ein zur Senke gerichteter MST optimal.

- jedes Paket kostet nur auf erster Kante
- erste Kanten bilden gerichteten Spannbaum
- Routingkosten entsprechen genau den Kosten des Baumes



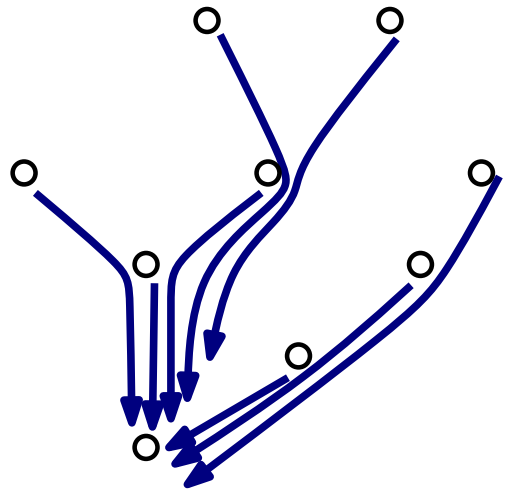
Optimale Topologien (Warm-up II)

Beobachtung II

Beim DG ohne Aggregation ist unabhängig von den Paketgrößen ein zur Senke gerichteter Kürzeste-Wege-Baum[a] optimal.

- jedes Paket „zahlt“ für Pfadlänge
- KW-Baum offensichtlich optimal
- zumindest zentral leicht zu berechnen

[a] bzgl. der Kosten



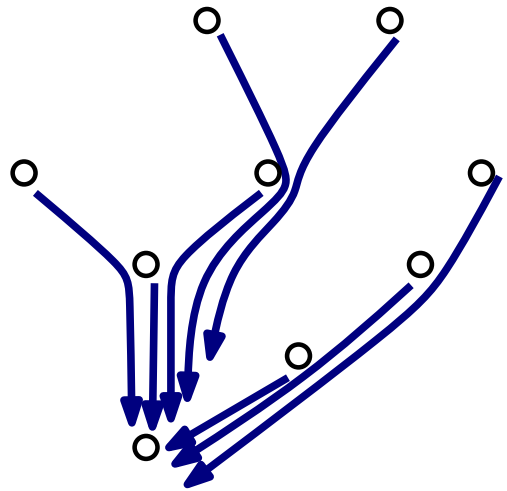
Optimale Topologien (Warm-up II)

Beobachtung II

Beim DG ohne Aggregation ist unabhängig von den Paketgrößen ein zur Senke gerichteter Kürzeste-Wege-Baum[a] optimal.

- jedes Paket „zahlt“ für Pfadlänge
- KW-Baum offensichtlich optimal
- zumindest zentral leicht zu berechnen

[a] bzgl. der Kosten



Schon wieder ist ein Baum optimal, das setzen wir aber nicht voraus! Pakete dürfen sich beliebig bewegen!

Data Gathering + Network Coding

Idee

Man kann auch alle Daten einsammeln und die Korrelationen zwischen den Daten ausnutzen!

- Zwei ähnliche Datenpakete nehmen weniger Platz ein, wenn man das erste schickt und statt des zweiten nur die Differenz/Quotienten usw.

Data Gathering + Network Coding

Idee

Man kann auch alle Daten einsammeln und die Korrelationen zwischen den Daten ausnutzen!

- Zwei ähnliche Datenpakete nehmen weniger Platz ein, wenn man das erste schickt und statt des zweiten nur die Differenz/Quotienten usw.

Wir gehen davon aus, dass jeder Knoten v_i in jeder Runde ein Datenpaket mit $|p_i|$ Bits „produziert“. Wie kann man geschickt erreichen, dass eine Senke alle Pakete rekonstruieren kann, wenn wir Network-Coding-Techniken nutzen?

Time Coding

Nicht sehr einfallsreich: Time Coding

Knoten nutzen vorangegangene *eigene* Messungen, um Pakete zu kodieren

- Messwerte verändern sich nicht drastisch
- verschicke ab und zu komplette Messwerte und sonst nur Unterschiede
- (wie bei Videokompression)

Das ist *kein* echtes Network Coding! Network Coding nutzt Redundanzen über Knotengrenzen hinweg!

Multi-Input- vs. Single-Input-Coding

Grundprinzip

Ein Paket p_i eines Knotens v_i kann zu einem (kleineren) Paket $p_i^{j_1,j_2,\ldots}$ kodiert werden, wenn Kodierer und Senke Kenntnis über Pakete $p_{j_1}, p_{j_2}, \ldots$ haben.

Multi-Input-Coding

Daten eines Knoten können in Abhängigkeit der Daten *vieler* anderer Knoten kodiert werden

- \+ nutzt Redundanzen vieler Knoten aus
- \- setzt geordneten Informationsfluss voraus

Single-Input-Coding

Daten eines Knotens werden nur abhängig von Information *eines* anderen Knotens kodiert.

- \- nutzt weniger Redundanz
- \+ leichter zu koordinieren
- \+ jedes Paket wird nur einmal (um)kodiert

Single-Input-Coding: Annahmen

Beim Single-Input-Coding darf jedes Paket p_i nur einmal durch Ausnutzung eines Paketes p_j umkodiert werden. Wir

- bezeichnen dann p'_i als p_i^j
- gehen davon aus, dass wir für alle i, j wissen, wie stark sich p_i unter Kenntnis von p_j komprimieren lässt, also $|p_i^j|/|p_i|$

Single-Input-Coding: Annahmen

Beim Single-Input-Coding darf jedes Paket p_i nur einmal durch Ausnutzung eines Paketes p_j umkodiert werden. Wir

- bezeichnen dann p_i' als p_i^j
- gehen davon aus, dass wir für alle i, j wissen, wie stark sich p_i unter Kenntnis von p_j komprimieren lässt, also $|p_i^j|/|p_i|$

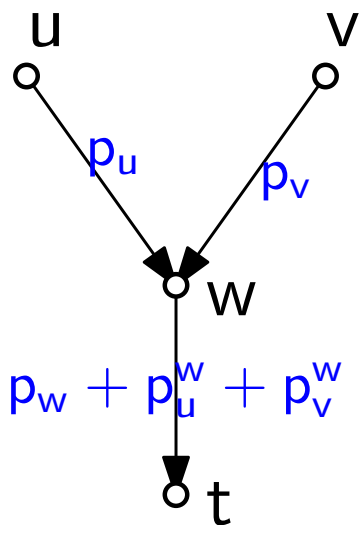
Achtung! Ein Knoten der p_i zu p_i^j umkodiert, muss dazu p_j zumindest rekonstruieren können und sollte p_i^j weiterleiten.

Foreign-Coding vs. Self-Coding

Foreign-Coding

Jeder Knoten darf bisher unkomprimierte Daten, die er weiterleitet kodieren und dabei eigene Daten verwenden.

- Kodierung von p_i zu p_i^j in Knoten v_j!



Foreign-Coding vs. Self-Coding

Foreign-Coding

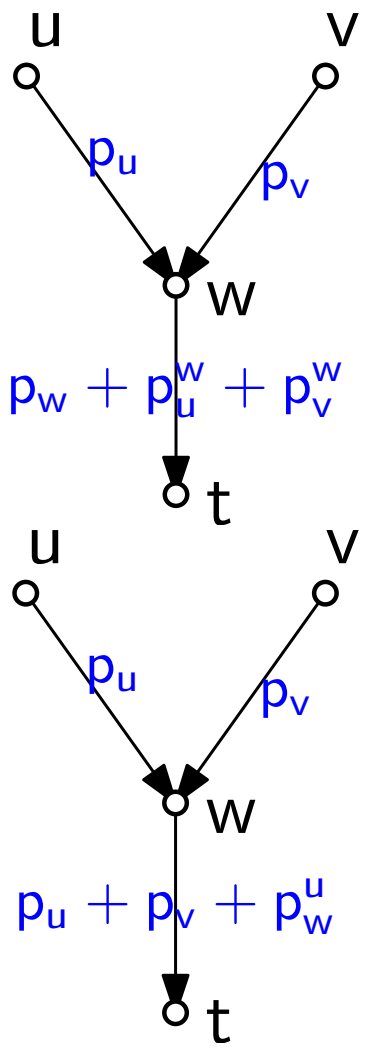
Jeder Knoten darf bisher unkomprimierte Daten, die er weiterleitet kodieren und dabei eigene Daten verwenden.

- Kodierung von p_i zu p_i^j in Knoten v_j!

Self-Coding

Jeder Knoten darf nur seine eigenen Daten kodieren und dabei Daten von Knoten verwenden, die er weiterleitet.

- Kodierung von p_i zu p_i^j in Knoten v_i

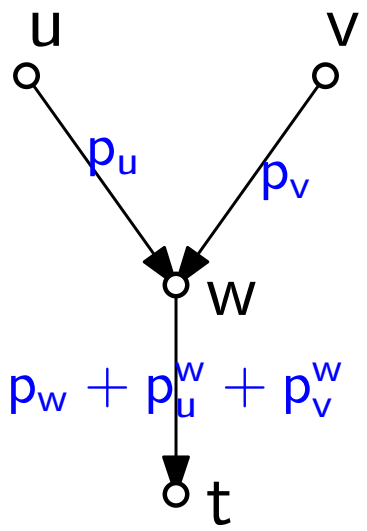


Problem: Single-Input Foreign-Coding

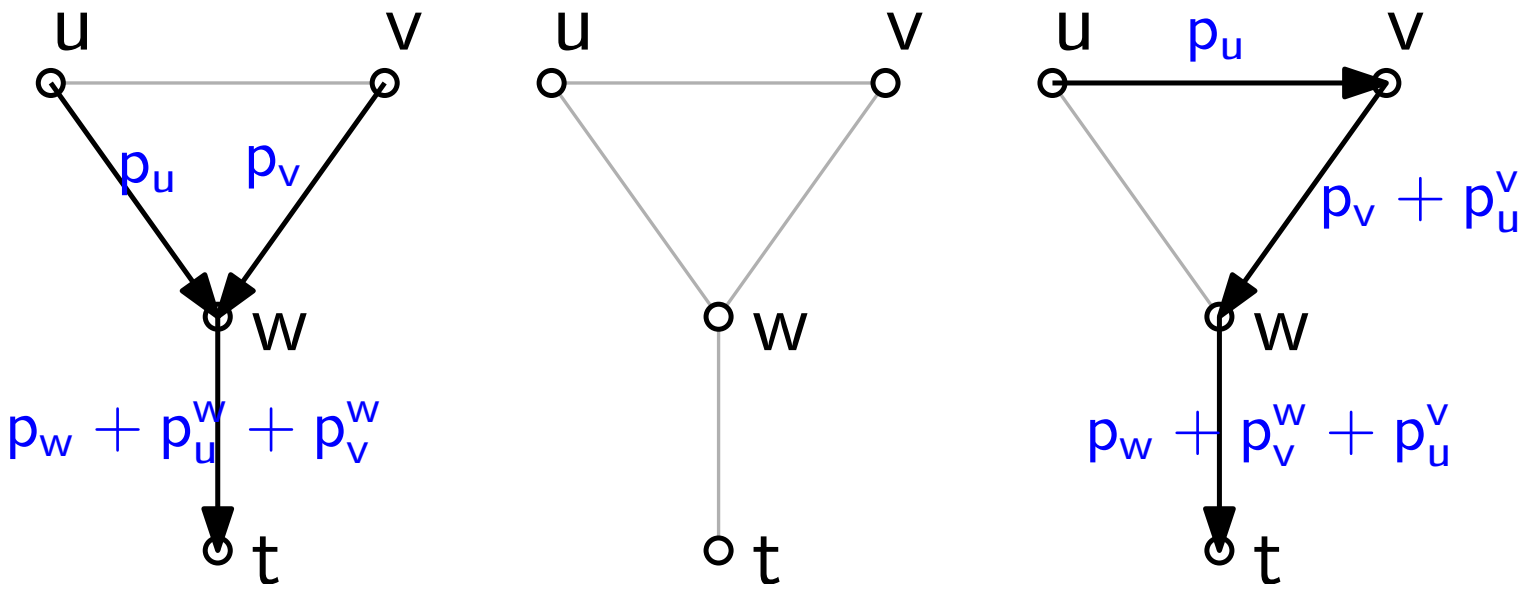
Gegeben: Graph $G = (V, E)$, Kantenkosten c_e, Paketgrößen $|p_i|$ für alle $v_i \in V$, abhängige Paketgrößen $|p_i^j|$ für alle $v_i, v_j \in V$

Gesucht: Routing und Kodierungsknoten so, dass die Gesamtkosten minimiert werden

- *Wohin* wird ein Paket geschickt?
- *Wo* wird es umkodiert?
- *Wohin* wird das umkodierte Paket geschickt?



Ein kleines Beispiel



Unterschiedliches Routing kann zu unterschiedlichen Kosten führen!

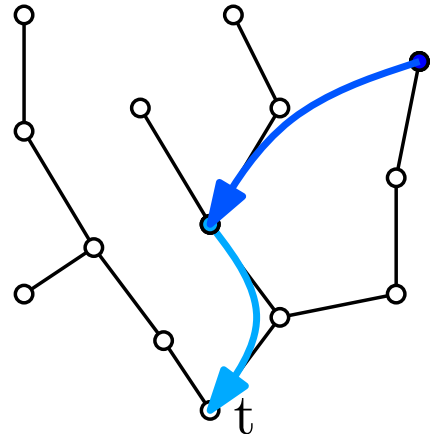
- Im Beispiel für einheitliche Kantenkosten
- links: $|p_u| + |p_v| + |p_w| + |p_u^w| + |p_v^w|$ (besser für $|p_u^w| < 2|p_u^v|$)
- rechts: $|p_u| + |p_v| + |p_w| + 2|p_u^v| + |p_v^w|$ (besser sonst)

Beobachtungen

Beobachtung I

Sobald ein Paket umkodiert wurde, sollte es auf dem billigsten Weg zur Senke geroutet werden

- braucht Kürzeste-Wege-Baum zu t

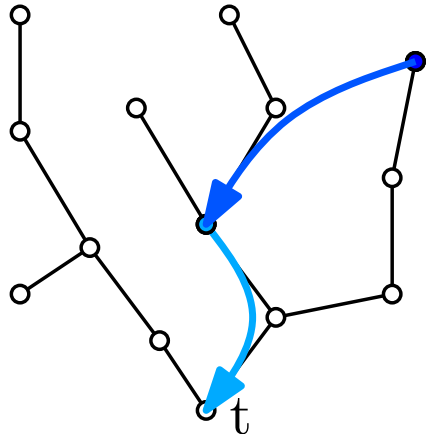


Beobachtungen

Beobachtung I

Sobald ein Paket umkodiert wurde, sollte es auf dem billigsten Weg zur Senke geroutet werden

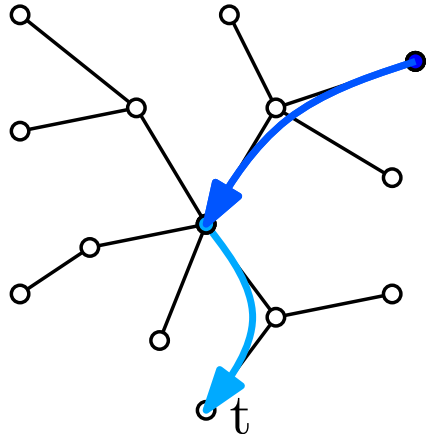
- braucht Kürzeste-Wege-Baum zu t



Beobachtung II

Um zu dem Knoten zu kommen, an dem es umkodiert wird, sollte ein Paket idealerweise auf dem billigsten Weg geroutet werden.

- braucht alle Kürzeste-Wege-Bäume

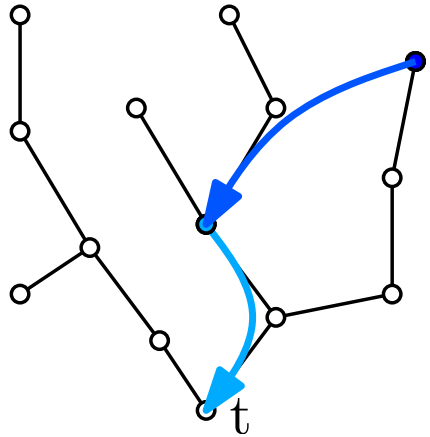


Beobachtungen

Beobachtung I

Sobald ein Paket umkodiert wurde, sollte es auf dem billigsten Weg zur Senke geroutet werden

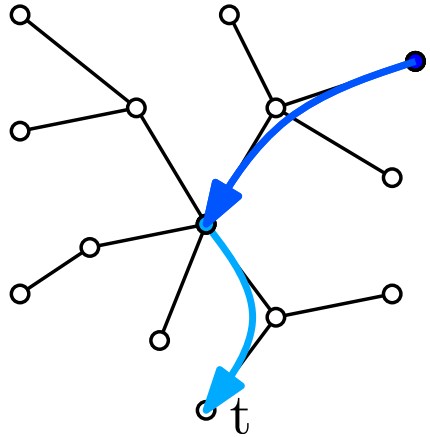
- braucht Kürzeste-Wege-Baum zu t



Beobachtung II

Um zu dem Knoten zu kommen, an dem es umkodiert wird, sollte ein Paket idealerweise auf dem billigsten Weg geroutet werden.

- braucht alle Kürzeste-Wege-Bäume



- Wie wählen wir Kodierungsknoten?

Optimale Route

Zusammenfassung

Wird ein Paket p_i in einem Knoten v_j zu p_i^j kodiert, sind die Kosten für p_i minimal, wenn

- p_i auf kürzestem Weg von v_i zu v_j geroutet wird
- p_i^j auf kürzestem Weg von v_j zur Senke t geroutet wird
- Kosten für Paket p_i sind dann

$$C(i,j) := |p_i| KW(v_i, v_j) + |p_i^j| KW(v_j, t)$$

(unkodierte Pakete haben Kosten $C(i,t)$)

Optimale Route

Zusammenfassung

Wird ein Paket p_i in einem Knoten v_j zu p_i^j kodiert, sind die Kosten für p_i minimal, wenn

- p_i auf kürzestem Weg von v_i zu v_j geroutet wird
- p_i^j auf kürzestem Weg von v_j zur Senke t geroutet wird
- Kosten für Paket p_i sind dann

$$C(i,j) := |p_i| KW(v_i, v_j) + |p_i^j| KW(v_j, t)$$

(unkodierte Pakete haben Kosten $C(i,t)$)

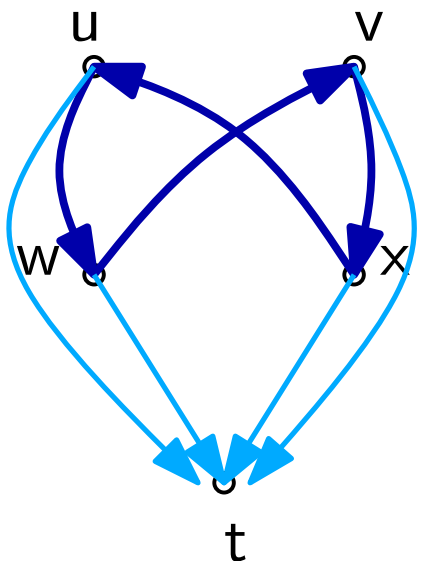
Trugschluss

Wenn man für jedes p_i das v_j zum Umkodieren so wählt, dass $C(i,j)$ minimiert wird, ist das Problem optimal gelöst.

Uhh-ohh

Definition

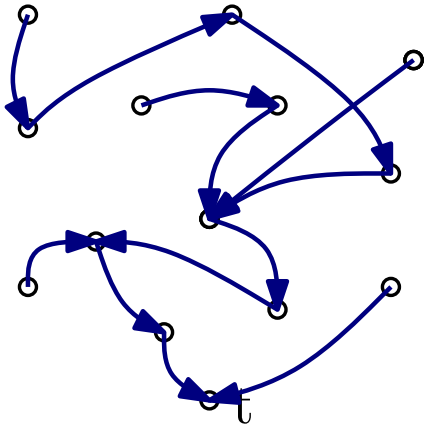
Der *Abhängigkeitsgraph* eines Single-Input-Codierungsschemas bezeichnet den gerichteten Graphen auf der Knotenmenge, der eine Kante (v_i, v_j) enthält, wenn p_i mit Hilfe von p_j kodiert wird.



Uhh-ohh

Definition

Der *Abhängigkeitsgraph* eines Single-Input-Codierungsschemas bezeichnet den gerichteten Graphen auf der Knotenmenge, der eine Kante (v_i, v_j) enthält, wenn p_i mit Hilfe von p_j kodiert wird.



Lemma

Die Senke kann die Pakete p_i nur dann rekonstruieren, wenn der Abhängigkeitsgraph zyklenfrei ist.

- wird p_i mit Hilfe von p_j kodiert, muss die Senke p_j zuerst rekonstruiert haben!

Optimale Lösung

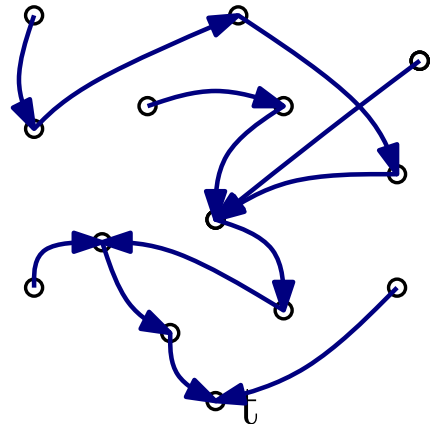
Satz

Ist $T = (V, E)$ ein zur Senke gerichteter Baum, der

$$C_T := \sum_{(v_i, v_j) \in E} C(i, j)$$

minimiert, dann ist es optimal, wenn für jedes $(v_i, v_j) \in E$ das Paket p_i von v_j kodiert wird.

- (Kante in Baum $\neq$ Kante in Netzwerk)



Optimale Lösung

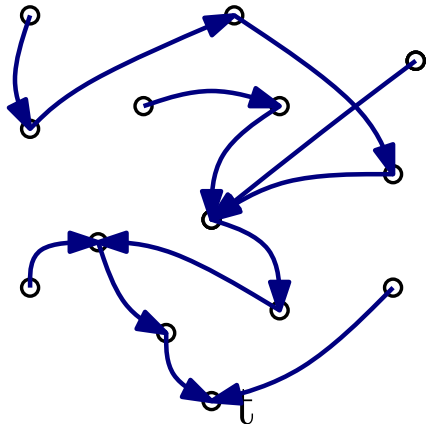
Satz

Ist $T = (V, E)$ ein zur Senke gerichteter Baum, der

$$C_T := \sum_{(v_i, v_j) \in E} C(i, j)$$

minimiert, dann ist es optimal, wenn für jedes $(v_i, v_j) \in E$ das Paket p_i von v_j kodiert wird.

- (Kante in Baum $\neq$ Kante in Netzwerk)
- $C(i, j)$ sind die Kosten, die Paket p_i erzeugt, wenn es in v_j kodiert wird
  - wieder: $C(i, t)$ für unkodierte Pakete!



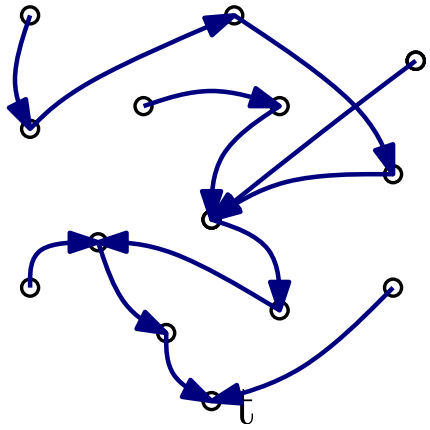
Optimale Lösung

Satz

Ist $T = (V, E)$ ein zur Senke gerichteter Baum, der

$$C_T := \sum_{(v_i, v_j) \in E} C(i, j)$$

minimiert, dann ist es optimal, wenn für jedes $(v_i, v_j) \in E$ das Paket p_i von v_j kodiert wird.



- (Kante in Baum $\neq$ Kante in Netzwerk)
- $C(i, j)$ sind die Kosten, die Paket p_i erzeugt, wenn es in v_j kodiert wird
  - wieder: $C(i, t)$ für unkodierte Pakete!
- die Kosten dieser Lösung sind genau C_T

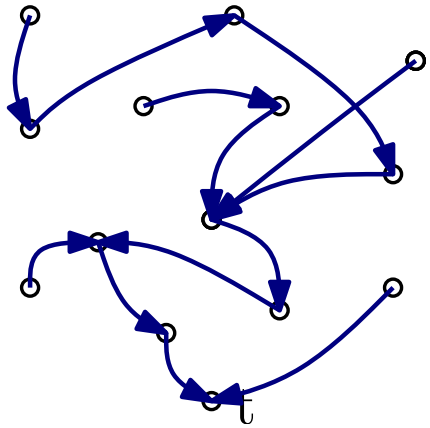
Optimale Lösung

Satz

Ist $T = (V, E)$ ein zur Senke gerichteter Baum, der

$$C_T := \sum_{(v_i, v_j) \in E} C(i, j)$$

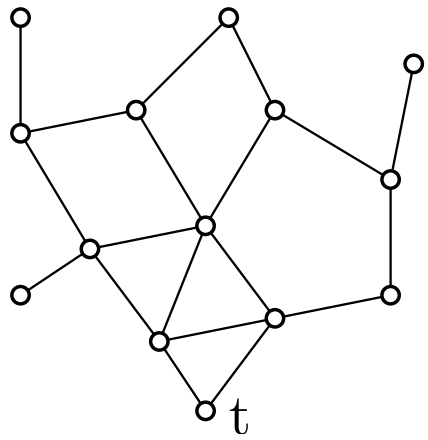
minimiert, dann ist es optimal, wenn für jedes $(v_i, v_j) \in E$ das Paket p_i von v_j kodiert wird.



- (Kante in Baum $\neq$ Kante in Netzwerk)
- $C(i, j)$ sind die Kosten, die Paket p_i erzeugt, wenn es in v_j kodiert wird
  - wieder: $C(i, t)$ für unkodierte Pakete!
- die Kosten dieser Lösung sind genau C_T
- keine Lösung kann besser sein:
  - jede Lösung muss azyklisch sein
  - jedes Paket wird entweder irgendwo kodiert oder kommt unkodiert in t an!

MEGA: *Minimum Energy Gathering Algo*

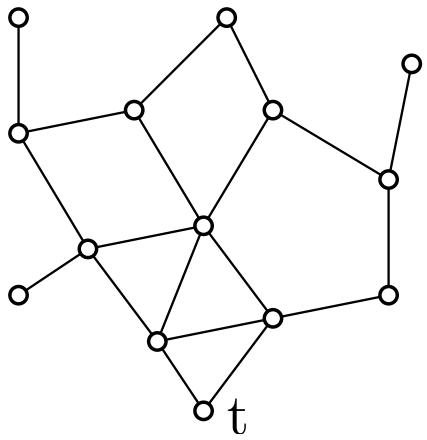
1. Berechne Kürzeste-Wege-Bäume zu allen Knoten
   - (das ist einfach, aber viel Arbeit)
2. Berechne alle $C(i, j)$
3. Berechne DMST (gerichteten MST) $T = (V, E)$ mit Wurzel t und Gewichten $C(i, j)$
   - das geht (nicht in dieser Vorlesung)
4. route Paket p_i über v_j für $(v_i, v_j) \in E$
   - route p_i auf kürzestem Weg zu v_j
   - kodiere dort p_i zu p_i^j
   - route p_i^j auf kürzestem Weg zu t



(Kommunikationsgraph, zur Erläuterung)

MEGA: *Minimum Energy Gathering Algo*

1. Berechne Kürzeste-Wege-Bäume zu allen Knoten
   - (das ist einfach, aber viel Arbeit)
2. Berechne alle $C(i,j)$
3. Berechne DMST (gerichteten MST) $T = (V, E)$ mit Wurzel t und Gewichten $C(i,j)$
   - das geht (nicht in dieser Vorlesung)
4. route Paket p_i über v_j für $(v_i, v_j) \in E$
   - route p_i auf kürzestem Weg zu v_j
   - kodiere dort p_i zu p_i^j
   - route p_i^j auf kürzestem Weg zu t



(Kommunikationsgraph, zur Erläuterung)

Satz (Beweis ist schon vorbei)

MEGA löst das Single-Input-Foreign-Coding-Problem optimal!

Diskussion: MEGA verteilt?

- woher kommen $|p_i^j|/|p_i|$ für entfernte Knoten?
- benötigt Kürzeste-Wege-Bäume *aller Knoten*
- berechnet DMST auf vollständigem Graphen

Diskussion: MEGA verteilt?

- woher kommen $|p_i^j|/|p_i|$ für entfernte Knoten?
- benötigt Kürzeste-Wege-Bäume *aller Knoten*
- berechnet DMST auf vollständigem Graphen

Beobachtung

Wenn wir die Auswahl der Kodierungsknoten auf die Nachbarschaft einschränken, haben wir diese Probleme nicht!

+ benachbarte Knoten können Korrelation ermitteln
+ *ein* Kürzeste-Wege-Baum von der Senke reicht aus
+ Nur tatsächlich vorhandene Kanten sind relevant für DMST-Berechnung (das geht verteilt, ohne Beweis)

Und wir verlieren nichts! (fast)

Satz

Sind die Pakete aller Knoten gleich groß und nimmt $|p_i^j|/|p_i| = |p_j^i|/|p_j|$ mit der Entfernung zwischen v_i und v_j zu, dann findet MEGA in UDGs auch eine optimale Lösung, wenn nur benachbarte Knoten als Kodierungsknoten ausgewählt werden dürfen.

Und wir verlieren nichts! (fast)

Satz

Sind die Pakete aller Knoten gleich groß und nimmt $|p_i^j|/|p_i| = |p_j^i|/|p_j|$ mit der Entfernung zwischen v_i und v_j zu, dann findet MEGA in UDGs auch eine optimale Lösung, wenn nur benachbarte Knoten als Kodierungsknoten ausgewählt werden dürfen.

- Es reicht zu zeigen: Es gibt immer eine optimale Lösung, in der jedes Paket von einem benachbarten Knoten kodiert wird

Und wir verlieren nichts! (fast)

Annahme

Kodierungsrelay v_j eines Knotens v_i in einer kostenoptimalen Data Gathering Topologie G_{opt} ist mehr als ein Hop von v_i entfernt.

1. Schritt

- Paket p_i wird entlang eines Pfades $p(v_i, v_j) = (v_i = u_0, u_1 \dots u_k = v_j)$ geroutet
- es gilt $d(v_i, v_j) > d(v_i, u_1)$

$\Rightarrow$ entsprechend unserer Annahme kann p_i in u_1 effizienter kodiert werden als in v_j

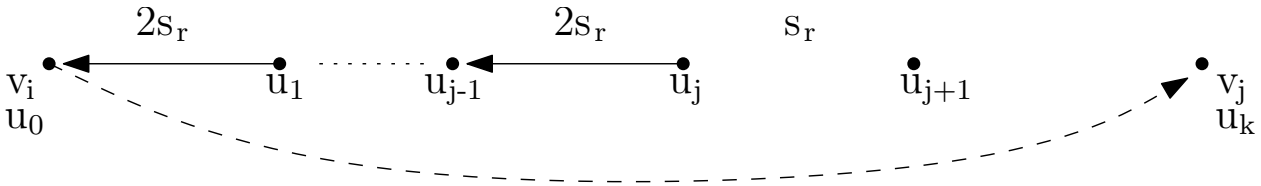
$\Rightarrow$ falls v_i nicht das Kodierungsrelay von u_1 ist, könnte u_1 als Kodierungsrelay für v_i verwendet werden

$\Rightarrow$ Topologie mit weniger Kosten! Widerspruch!

Und wir verlieren nichts! (fast)

2. Schritt

- Paket p_i wird entlang eines Pfades $(v_i = u_0, u_1 \ldots u_{j-1}, u_j, u_{j+1}, \ldots u_k = v_j)$ geroutet, wobei u_j letzter Knoten, so dass Vorgänger u_{j-1} Kodierungsrelay von u_j
  - solcher Knoten u_j muss existieren, da Abhängigkeitsgraph zyklenfrei

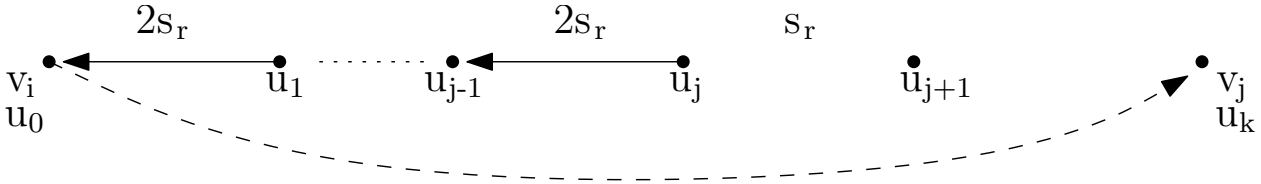


(*Die Abbildung zeigt die unkodierten Datenmengen (s_r pro Paket) auf dem Pfad.*)

Und wir verlieren nichts! (fast)

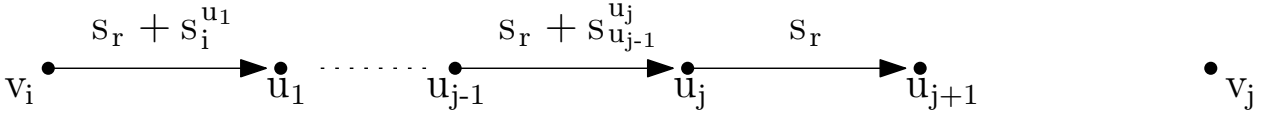
2. Schritt

- Paket p_i wird entlang eines Pfades $(v_i = u_0, u_1 \ldots u_{j-1}, u_j, u_{j+1}, \ldots u_k = v_j)$ geroutet, wobei u_j letzter Knoten, so dass Vorgänger u_{j-1} Kodierungsrelay von u_j
  - solcher Knoten u_j muss existieren, da Abhängigkeitsgraph zyklenfrei



(*Die Abbildung zeigt die unkodierten Datenmengen (s_r pro Paket) auf dem Pfad.*)

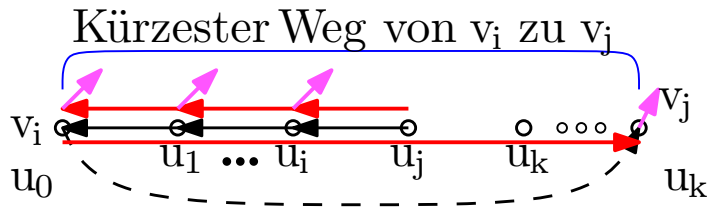
- durch Änderung der Kodierungsrelays lassen sich die Kosten auf dem Pfad senken



(*Ein Paket p_{u_i} wird zu einem Knoten u_{i+1} geschickt und das kodierte Paket $p_{u_i}^{u_{i+1}}$ auf Weg zur Senke möglicherweise wieder über u_i geleitet - Details auf nächster Folie*)

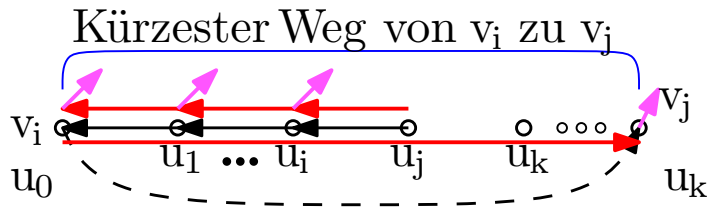
Und wir verlieren nichts! (fast)

- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



Und wir verlieren nichts! (fast)

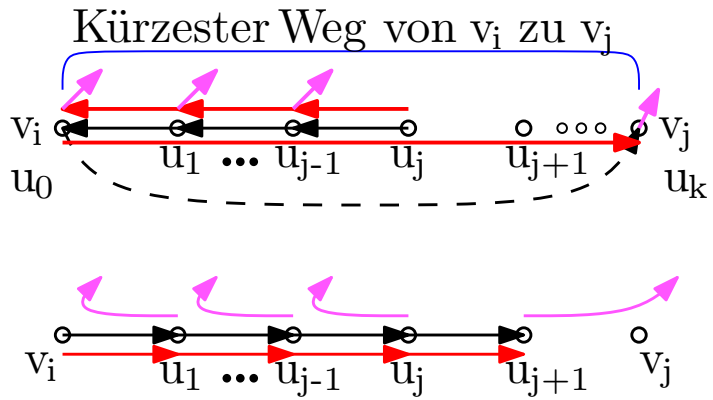
- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



- Kosten der Pakete von $v_i = u_0$ bis u_j bis zur Kodierung $\left(2\sum_{i=1}^{j} c_{(u_{i-1},u_i)} + \sum_{i=j}^{k} c_{(u_{i-1},u_i)}\right)|p|$

Und wir verlieren nichts! (fast)

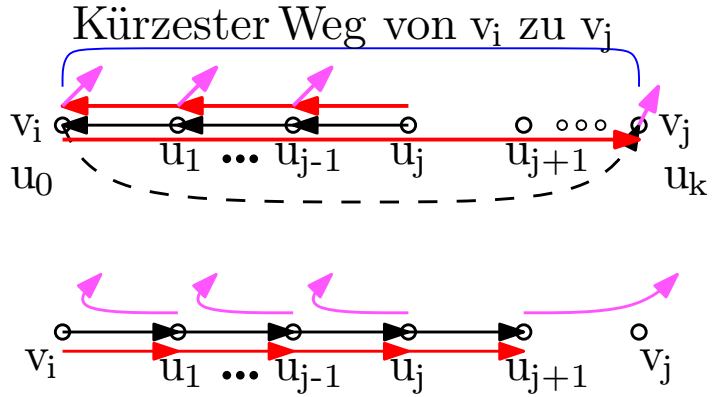
- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



- Kosten der Pakete von $v_i = u_0$ bis u_j bis zur Kodierung $\left(2\sum_{i=1}^{j} c_{(u_{i-1},u_i)} + \sum_{i=j}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Ändere Kanten wie in Abbildung

Und wir verlieren nichts! (fast)

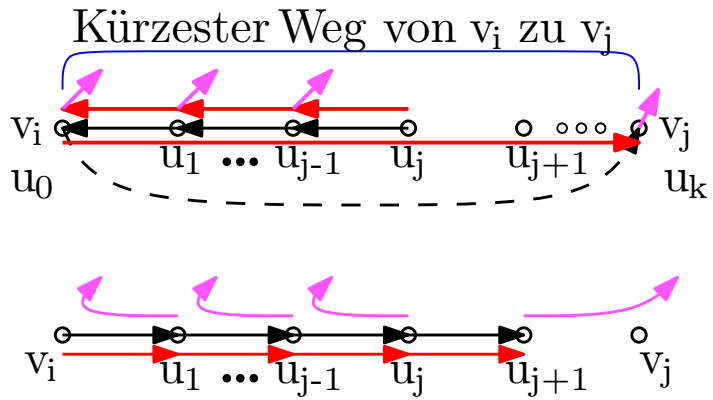
- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



- Kosten der Pakete von $v_i = u_0$ bis u_j bis zur Kodierung $\left(2\sum_{i=1}^{j} c_{(u_{i-1},u_i)} + \sum_{i=j}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Ändere Kanten wie in Abbildung
- Kosten der Pakete von u_0 bis u_j bis zur Kodierung jetzt noch $\left(\sum_{i=1}^{j+1} c_{(u_{i-1},u_i)}\right)|p|$

Und wir verlieren nichts! (fast)

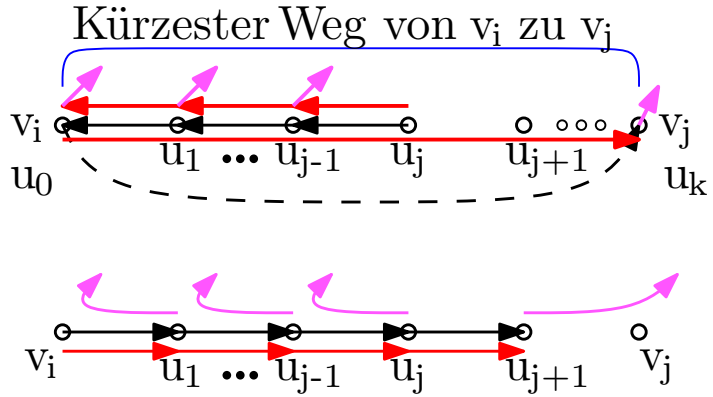
- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



- Kosten der Pakete von $v_i = u_0$ bis u_j bis zur Kodierung $\left(2\sum_{i=1}^{j} c_{(u_{i-1},u_i)} + \sum_{i=j}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Ändere Kanten wie in Abbildung
- Kosten der Pakete von u_0 bis u_j bis zur Kodierung jetzt noch $\left(\sum_{i=1}^{j+1} c_{(u_{i-1},u_i)}\right)|p|$
- Dazu im schlimmsten Fall noch Zusatzkosten um wieder genausoviele kodierte Pakete in allen Knoten zu haben wie vorher (falls das besser war) $\left(\sum_{i=1}^{j} c_{(u_{i-1},u_i)}\right)|p| + \left(\sum_{i=j+2}^{k} c_{(u_{i-1},u_i)}\right)|p|$

Und wir verlieren nichts! (fast)

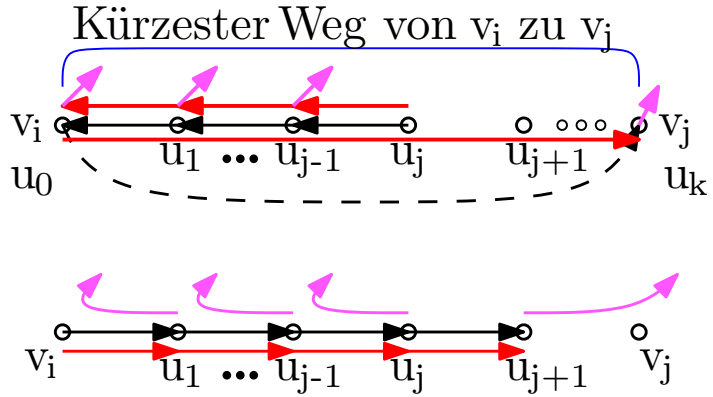
- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



- Kosten der Pakete von $v_i = u_0$ bis u_j bis zur Kodierung $\left(2\sum_{i=1}^{j} c_{(u_{i-1},u_i)} + \sum_{i=j}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Ändere Kanten wie in Abbildung
- Kosten der Pakete von u_0 bis u_j bis zur Kodierung jetzt noch $\left(\sum_{i=1}^{j+1} c_{(u_{i-1},u_i)}\right)|p|$
- Dazu im schlimmsten Fall noch Zusatzkosten um wieder genausoviele kodierte Pakete in allen Knoten zu haben wie vorher (falls das besser war) $\left(\sum_{i=1}^{j} c_{(u_{i-1},u_i)}\right)|p| + \left(\sum_{i=j+2}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Nachrechnen, das ist dasselbe!

Und wir verlieren nichts! (fast)

- betrachte Knoten $v_i = u_0$ mit Kodierungsknoten v_j außerhalb der Nachbarschaft und kürzesten Weg dorthin
  - es gibt ein letztes u_j, das einen Pfad im DMST zu v_i hat



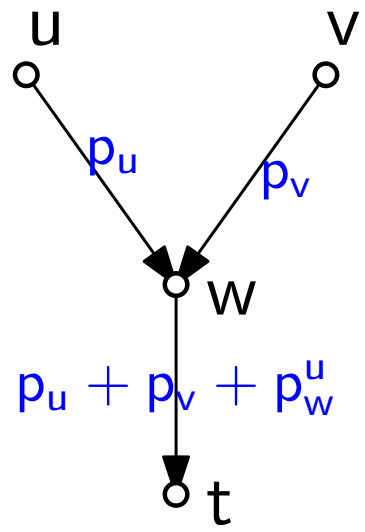
- Kosten der Pakete von $v_i = u_0$ bis u_j bis zur Kodierung $\left(2\sum_{i=1}^{j} c_{(u_{i-1},u_i)} + \sum_{i=j}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Ändere Kanten wie in Abbildung[1]
- Kosten der Pakete von u_0 bis u_j bis zur Kodierung jetzt noch $\left(\sum_{i=1}^{j+1} c_{(u_{i-1},u_i)}\right)|p|$
- Dazu im schlimmsten Fall noch Zusatzkosten um wieder genausoviele kodierte Pakete in allen Knoten zu haben wie vorher (falls das besser war) $\left(\sum_{i=1}^{j} c_{(u_{i-1},u_i)}\right)|p| + \left(\sum_{i=j+2}^{k} c_{(u_{i-1},u_i)}\right)|p|$
- Nachrechnen, das ist dasselbe!

[1] Abbildung: Schwarz: Kodierungsabhängigkeit, Rot: Unkodierte Pakete, Violet: Kodierte Pakete

Problem: Single-Input Self-Coding

Erinnerung: Self-Coding

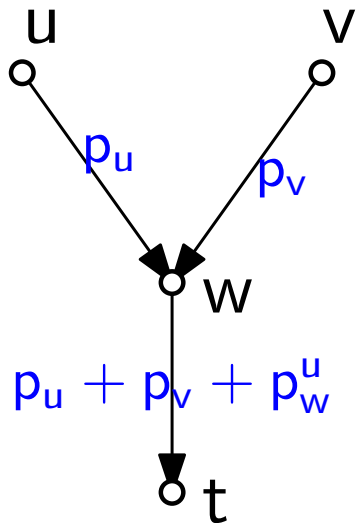
- Paket p_i wird in v_i kodiert mit Hilfe eines Paketes p_j, das v_i rekonstruieren kann.



Problem: Single-Input Self-Coding

Erinnerung: Self-Coding

- Paket p_i wird in v_i kodiert mit Hilfe eines Paketes p_j, das v_i rekonstruieren kann.



Gegeben: Graph $G = (V, E)$, Kantenkosten c_e, Paketgrößen $|p_i|$ für alle $v_i \in V$, abhängige Paketgrößen $|p_i^j|$ für alle $v_i, v_j \in V$

Gesucht: Routing und Kodierungszuordnung, die die Gesamtkosten minimieren

- *Wohin* werden Pakete verschickt?
- *Welches* Paket wird zur Kodierung verwendet?

Problem: Single-Input Self-Coding

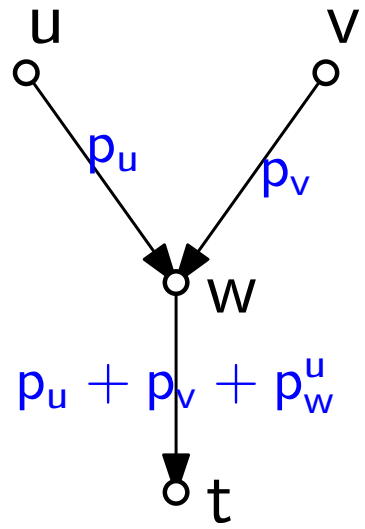
Erinnerung: Self-Coding

- Paket p_i wird in v_i kodiert mit Hilfe eines Paketes p_j, das v_i rekonstruieren kann.

Gegeben: Graph $G = (V, E)$, Kantenkosten c_e, Paketgrößen $|p_i|$ für alle $v_i \in V$, abhängige Paketgrößen $|p_i^j|$ für alle $v_i, v_j \in V$

Gesucht: Routing und Kodierungszuordnung, die die Gesamtkosten minimieren

- *Wohin* werden Pakete verschickt?
- *Welches* Paket wird zur Kodierung verwendet?



Lösungen müssen nicht so einfach aussehen!

Problem: Single-Input Self-Coding

Satz (ohne Beweis)

Single-Input Self-Coding ist schon NP-schwer für einheitliche Paketgrößen $|p_i| \equiv s$ und von den Quellen unabhängige Kompression $p_i^j \equiv s_e \ (\forall i, j)$.

Problem: Single-Input Self-Coding

Satz (ohne Beweis)

Single-Input Self-Coding ist schon NP-schwer für einheitliche Paketgrößen $|p_i| \equiv s$ und von den Quellen unabhängige Kompression $p_i^j \equiv s_e \ (\forall i, j)$.

- nur für diesen Fall gibt es einen Faktor-2-Approximationsalgorithmus
  - LEGA: Low-Energy Gathering Algorithm
  - (eigentlich: Faktor $2(1 + \sqrt{2})$ und Kommunikation nur auf *einem* Baum, im Folgenden vereinfacht)

Beobachtungen

DG mit Self-Coding kostet mindestens so viel wie es kostet, alle kodierten Pakete auf dem billigsten Weg zur Senke zu bringen:

$$C_{\mathrm{OPT}} \geq s_e \sum_{u \in V} KW(u,t)$$

Beobachtungen

DG mit Self-Coding kostet mindestens so viel wie es kostet, alle kodierten Pakete auf dem billigsten Weg zur Senke zu bringen:

$$C_{\text{OPT}} \geq s_e \sum_{u \in V} KW(u,t)$$

Knoten schicken entweder ihre Rohdaten unkodiert zur Senke oder benötigen Rohdaten eines anderen Knotens zum Kodieren ihrer Daten:

$$C_{\text{OPT}} \geq s \cdot c(MST)$$

Beobachtungen

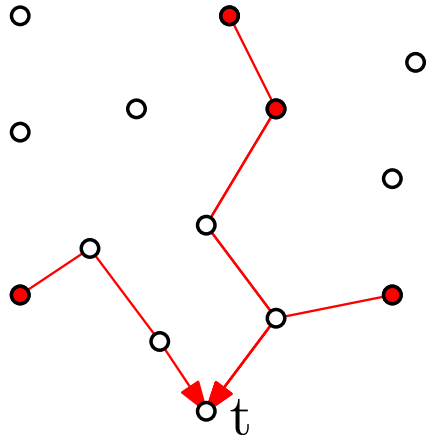
DG mit Self-Coding kostet mindestens so viel wie es kostet, alle kodierten Pakete auf dem billigsten Weg zur Senke zu bringen:

$$C_{\text{OPT}} \geq s_e \sum_{u \in V} KW(u, t)$$

Knoten schicken entweder ihre Rohdaten unkodiert zur Senke oder benötigen Rohdaten eines anderen Knotens zum Kodieren ihrer Daten:

$$C_{\text{OPT}} \geq s \cdot c(MST)$$

- einige Knoten senden Rohdaten an die Senke
  - das ergibt Menge von Pfaden zwischen diesen Knoten und der Senke



Beobachtungen

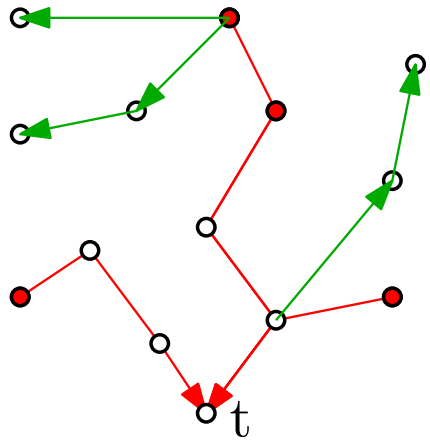
DG mit Self-Coding kostet mindestens so viel wie es kostet, alle kodierten Pakete auf dem billigsten Weg zur Senke zu bringen:

$$C_{OPT} \geq s_e \sum_{u \in V} KW(u,t)$$

Knoten schicken entweder ihre Rohdaten unkodiert zur Senke oder benötigen Rohdaten eines anderen Knotens zum Kodieren ihrer Daten:

$$C_{OPT} \geq s \cdot c(MST)$$

- einige Knoten senden Rohdaten an die Senke
  - das ergibt Menge von Pfaden zwischen diesen Knoten und der Senke
- auch ein Knoten, der nicht auf einem dieser Pfade liegt, empfängt trotzdem mind. einmal Rohdaten!
  - das „verbindet“ sie mit den Pfaden



Beobachtungen

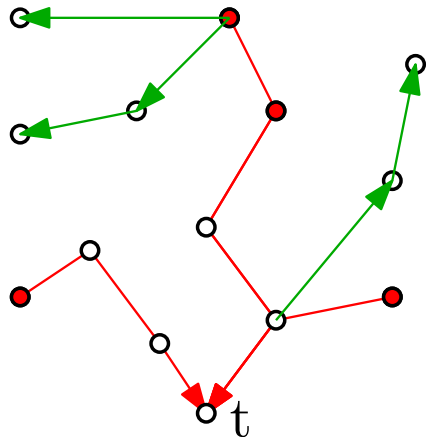
DG mit Self-Coding kostet mindestens so viel wie es kostet, alle kodierten Pakete auf dem billigsten Weg zur Senke zu bringen:

$$C_{\text{OPT}} \geq s_e \sum_{u \in V} KW(u,t)$$

Knoten schicken entweder ihre Rohdaten unkodiert zur Senke oder benötigen Rohdaten eines anderen Knotens zum Kodieren ihrer Daten:

$$C_{\text{OPT}} \geq s \cdot c(MST)$$

- einige Knoten senden Rohdaten an die Senke
  - das ergibt Menge von Pfaden zwischen diesen Knoten und der Senke
- auch ein Knoten, der nicht auf einem dieser Pfade liegt, empfängt trotzdem mind. einmal Rohdaten!
  - das „verbindet“ sie mit den Pfaden



Kombinierte Schranke: $2C_{\text{OPT}} \geq s \cdot c(MST) + s_e \sum_{u \in V} KW(u,t)$

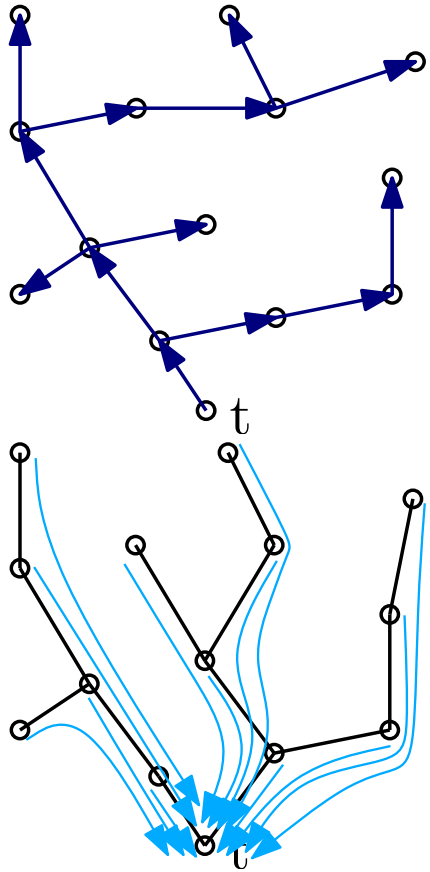
LEGA, vereinfacht

1. Berechne MST *und* Kürzeste-Wege-Baum
2. jedes v_i schickt sein p_i an seine Kinder im MST
3. jedes v_i kodiert sein p_i mit dem Paket seines Vorgängers im MST und schickt es im Kürzeste-Wege-Baum Richtung Senke

- Abhängigkeiten sind wieder azyklisch!

$$C_{\text{LEGA}} \leq s \cdot MST + s_e \cdot \sum_{u \in V} KW(u,t) \leq 2C_{\text{OPT}}$$

- aber: kein „klassisches“ Routing
- dafür: nur Pakete von Nachbarn zum Kodieren



Zusammenfassung

- Network Coding
  - Vermischen von Datenströmen kann Schranken brechen!
- Data Gathering + Network Coding
  - *eine* Möglichkeit, Redundanz in Messdaten auszunutzen
  - bei Korrelation naher Knoten enorme Einsparung mit verhältnismäßig wenig Aufwand
  - viele Varianten denkbar!

Literatur

1. P. von Rickenbach, R. Wattenhofer. *Gathering Correlated Data in Sensor Networks*. In: ACM Joint Workshop on Foundations of Mobile Computing (DIALM-POMC), 2004