Willkommen bei Kommunikations- und Netztechnik!

- *Von Kupferkabel, Glasfaser und Mikrowelle*
  über Telefon, Ethernet und TCP
  zu E-Mail, Webserver und REST.
- 
- Heute: **Medium Access Control (MAC): Wer darf wann senden? Kollisionen.**

Wiederholung Bitübertragung

- Nyquist Formel
- Shannon Formel
- Kennen der Begriffe duplex, simplex etc.
- Unterteilung Übertragungsmedien, Beispiele für Kategorien
  - Kabelgebunden
    - Kupfer
    - LWL
  - Kabellos
- Modulation
- Passband: FSK, ASK, PSK
- Bandbreitenbedarf, Taktrückgewinnung, Gleichstromfreiheit
- Multiplexing: FDM (Frequenz), TDM (Zeit), CDMA (Code)
- CDMA
  - Walshcodes zur Encodierung mehrerer Signale in ein Signal

Weitere Fragen?

Wiederholung Sicherung I

- Dienst der Sicherungsschicht: Pakete aus Bits übertragen
- Protokoll der Sicherungsschicht: Pakete in Rahmen verpacken und übermitteln.
- Aufgaben der Sicherungsschicht: Rahmenbildung, Fehlerkorrektur und -erkennung, Flusskontrolle
- Fehlerkorrektur erhöht Latenz. Der richtige Grad an Fehlerkorrektur minimiert die durchschnittliche Gesamtzeit bis zur Korrekten Übertragung.
- Hamming-Codes haben Korrekturbits auf Zweierpotenzen.
- Hamming-Abstand: Abstand in Bitflips zwischen den zwei ähnlichsten Codewörtern. Ab dieser Zahl Bitfehler können Fehler unentdeckt bleiben.
- 1-Bit Schiebefensterprotokoll: Übertrage nur nach Freigabe via Bestätigung der Rahmennummer.

Lernziele I

- Sie kennen die Nachteile statischer Kanal Allokation
- Sie kennen die Protokolle
  - ALOHA, pure und slotted
  - verschiedene CSMA Ausprägungen
  - kollisionsfreie Protokolle
  - adaptive tree walk protocol
  - MACA
- Sie können Aloha, CSMA/CD, Token Ring und MACA anwenden.
- Sie kennen den Aufbau des Ethernet Frames
- Sie können die Switch Algorithmen anwenden:
  - backward learning
  - spanning tree
- Sie erkennen die VLAN Implementierung in Switches

Medium Access Control (MAC): Aufgabe

Mehrere Stationen verwenden das selbe Medium (Kanal)

- wer darf wann den Kanal verwenden?
- wie wird mit Kollisionen umgegangen?

Rückblick: Statische Verfahren

- FDM (Frequenz)
- TDM (Zeit)
- CDMA (Code)
- Vorraussetzungen
  - konstante Anzahl Stationen
  - konstante Last pro Station

Gut bei Radio.

Warum dynamische Verfahren?

- Probleme statischer Verfahren
  - **Variable Anzahl Stationen**
    - Verschwendung von Bandbreite (weniger Stationen als Slots)
    - Ausschluss von Stationen (mehr Stationen als Slots)
  - **Variable Last pro Station**
    - Verschwendung von Bandbreite (nicht alle haben immer etwas zu senden)

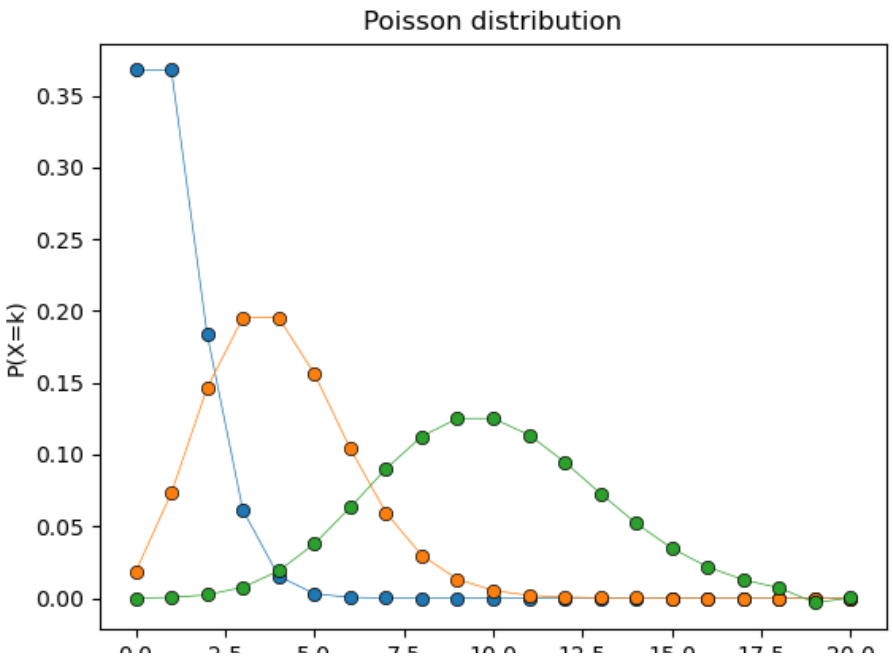
Annahmen für dynamische Verfahren

- Last in Computernetzwerken wird stoßweise erzeugt
  Beispiel: Neujahr, WM-Tor
- Die Anzahl der Stationen ist variabel
  „Bitte klappen Sie die Laptops zu“

Für statische Verfahren problematisch.

Annahme: Unabhängiger Traffic

- mehrere unabhängige Stationen
- Frames unvorhersagbar aber mit gemittelt konstanter Häufigkeit
  - konstant: Wahrscheinlichkeit, dass ein Frame generiert wird ist über alle Intervalle konstant
- Modellierung in Literatur: Poisson Verteilung



Annahme: Einzelner Kanal

- alle Stationen verwenden den selben Kanal
- zwischen den Stationen existiert kein anderer Kanal

Annahme: beobachtbare Kollisionen?

Können Stationen Kollisionen feststellen?

- Kabel: während dem Senden lesen
  - $S_{gesendet} \neq S_{gelesen} \rightarrow$ Kollision
- Kabellos: problematisch da
  - Signalstärke von anderen Stationen kann beträchtlich schwächer sein
  - Alternative: Kollisionserkennung durch Bestätigung des Empfängers

Annahme: kontinuierliche oder diskrete Zeit

- kontinuierlich: Übertragung kann zu jeder Zeit stattfinden
- diskret: Übertragung nur zu bestimmten Zeitpunkten
  - benötigt Synchronisation
  - kann Performance verbessern

Annahme: Carrier Sense oder nicht

- Carrier Sense: es kann festgestellt werden, ob der Kanal frei ist
- Kabellos: nicht alle Stationen sind in gegenseitiger Reichweite
- Kabel: einfach und üblich

Zusammenfassung

- MAC bisher statisch
- Jetzt dynamische Verfahren
  - Gemeinsamer Kanal
  - Kollisionserkennung

Drahtlos: ALOHA Protokolle

- auf Hawaii entwickelt
- Ziel: verbinden von Computern auf kleinen Inseln mit Universitätscomputer
- Unterseekabel legen war keine Option
- Verwendung von Radiowellen
- 2 Ausprägungen

Pure ALOHA

- jede Station sendet, sobald sie einen Frame generiert hat
- Kollisionen werden auftreten, Erkennung durch:
  - zentraler Computer bestätigt empfangene Frames
  - Sender liest Bestätigungen: wurde Frame empfangen?
  - falls nicht, gab es eine Kollision
- bei Kollision wird zufällige Zeit gewartet und erneut gesendet
- zufällige Zeit wichtig, da es sonst zu immer den selben Kollisionen kommt

Versuch: Pure ALOHA Namen

- Augen schließen
- Namen ohne Unterbrechungen in der Gruppe sagen
- Keine Reihenfolge!
- Ich bestätige alle gehörten Namen („zentraler Computer“)
- Wie viele Namen in 30 Sekunden?

Slotted ALOHA

- Zeit wird in diskrete Intervalle unterteilt (Slots)
- Intervalllänge entspricht Framelänge
- Frames dürfen nur zu Beginn eines Slots gesendet werden
- Benötigt Synchronisation

Slotted ALOHA

- Zeit wird in diskrete Intervalle unterteilt (Slots)
- Intervalllänge entspricht Framelänge
- Frames dürfen nur zu Beginn eines Slots gesendet werden
- Benötigt Synchronisation

sync

- z.B.: durch periodisches Signal des zentralen Computers
- Zeitansage bei Telefon
- einfache bundesweite Synchronisation

Pure ALOHA

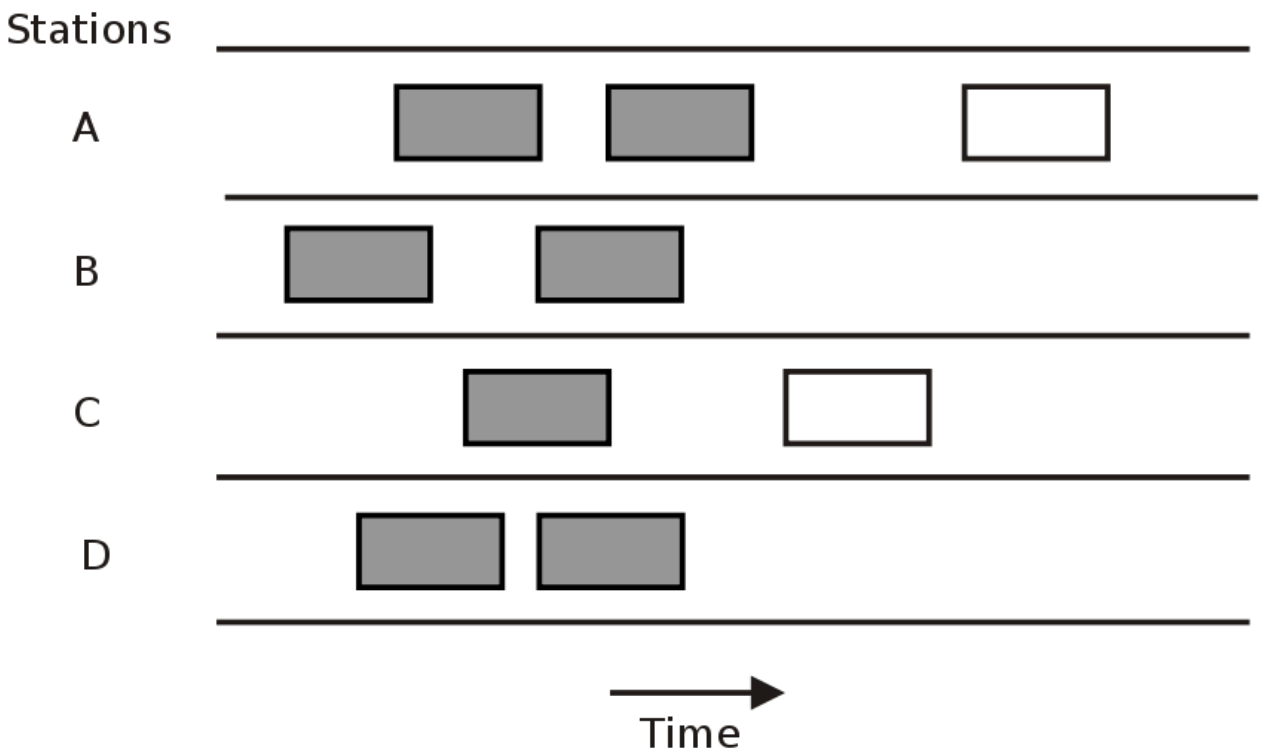
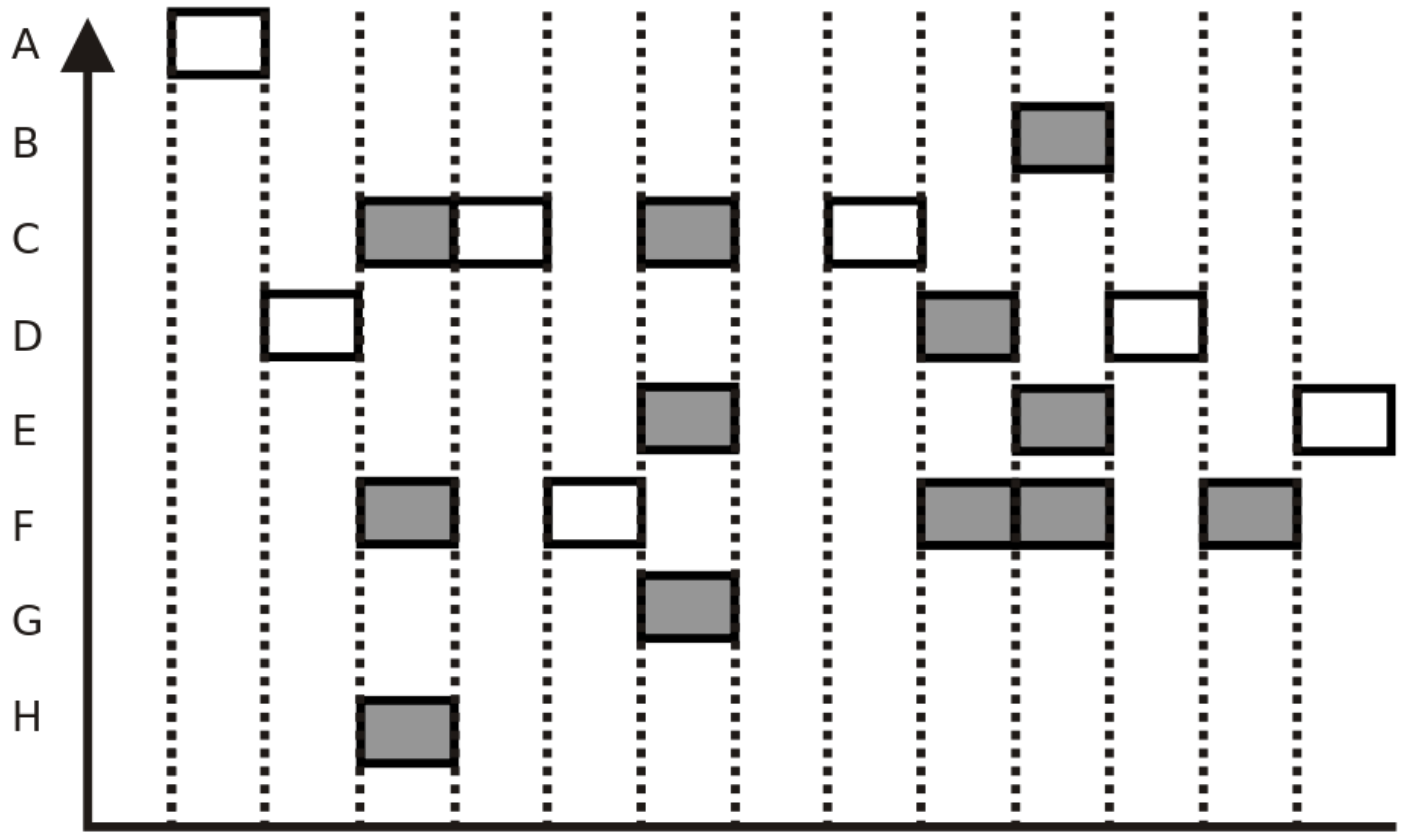


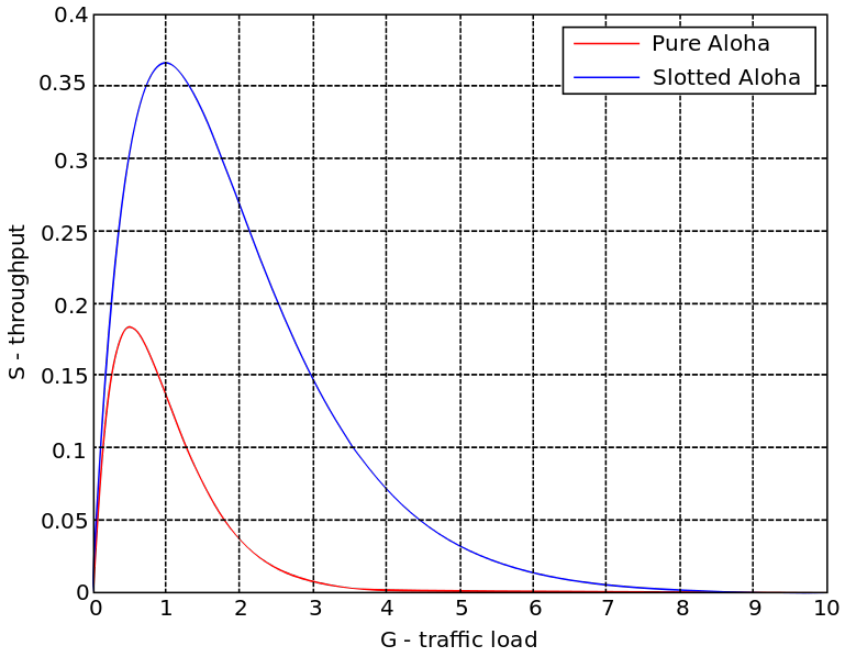
Abbildung: Pure Aloha, helix84, cc by-sa (Wikipedia:Pure_Aloha1.svg)

Slotted ALOHA



Slotted ALOHA protocol (shaded slots indicate collision)

Pure vs. Slotted



- G: Anzahl der zu übertragenen Frames pro Framezeit
- S: Anteil der Zeit in der der Channel genutzt wird
- Pure: S=max 18%
- Slotted: S=max 36%

Abbildung: Throughput Pure vs. Slotted, Kyurim und fl, CC0, commons.wikimedia.org/wiki/File:Aloha_PureVsSlotted.svg

Versuch: Slotted ALOHA Namen

- Namen ohne Unterbrechungen in der Gruppe sagen
- Ich gebe Takt: 2 Sekunden Zeitscheiben
- Keine Reihenfolge!
- Ich bestätige alle gehörten Namen („zentraler Computer“)
- Wie viele Namen in 30 Sekunden?

Zusammenfassung

- Aloha:
  - Drahtlos auf Hawai
  - Zufälliges Warten bei Kollisionen
- Slotted ist doppelt so effizient
- Slotted braucht Synchronisierung

CSMA (Carrier Sense Multiple Access)

In kabelgebundenen Netzwerken ist Carrier Sense möglich.

- z.B. in Ethernet
- mehrere unterschiedliche Protokolle
  - 1-persistent/p-persistent
  - nonpersistent
  - mit Kollisionserkennung

1-persistent

- vor dem Senden: überprüfe Kanal
  - wenn frei: sende
  - wenn belegt: warte bis frei und sende direkt
- Bezeichnung „1-persistent“:
  - sendet mit Wahrscheinlichkeit 1, wenn Kanal frei

Probleme bei 1-persistent

- A und B möchten senden
- Kanal ist belegt
- sobald Kanal frei ist, senden A und B –> Kollision

Nonpersistent: ALOHA auf Kabel

- vor dem Senden: überprüfe Kanal
  - wenn frei: sende
  - wenn belegt: warte zufällige Zeit und beginne von vorn
- Problem: höhere Latenz

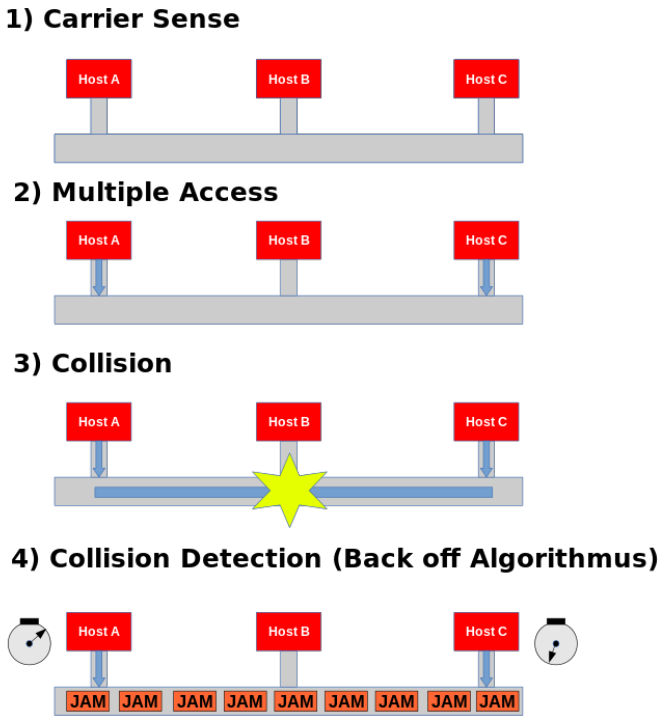
p-persistent

- Zeit wird in Slots unterteilt (diskret)
- vor dem Senden: überprüfe Kanal
  - wenn frei: sende mit Wahrscheinlichkeit p
    - bzw. warte mit Wahrscheinlichkeit $q = 1 - p$ auf den nächsten Slot
    - nächster Slot auch frei: wiederhole mit p und q
    - nächster Slot belegt: warte eine zufällige Zeit
  - wenn belegt: warte auf nächsten Slot

CSMA/CD — (early) Collision Detection

- CSMA erhöht Effizienz durch Carrier Sensing ggü. ALOHA
- Problem:
  - bei Kollision wird der gesamte Frame gesendet
- Verbesserung: erkenne eine Kollision frühzeitig und breche ab
  - spart Zeit und Bandbreite
  - Bestätigungsframes sind auch nicht mehr nötig
  - $S_{gesendet} \neq S_{gelesen} \rightarrow$ Kollision

Kollisionserkennung CSMA/CD, Beispiel



- zwei Stationen möchten senden
- es kommt zur Kollision→ JAM Signal
- beide Stationen warten zufällige Zeit

Abbildung: Wikimedia, Deadlyhappen, cc by-sa, 2013, CSMA-CD_Verfahren.svg

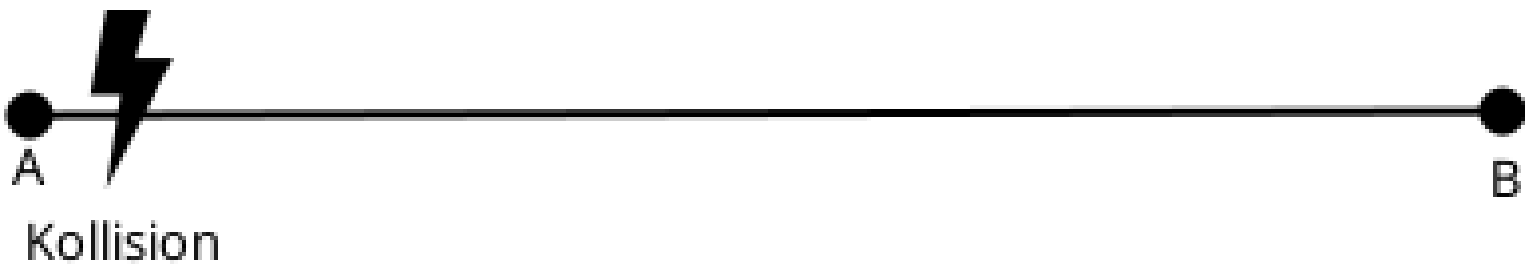
Länge der Contention Period I

- A sendet Frame zu B
- Signallaufzeit von A nach B: τ

- Frame ist zu $\tau - \epsilon$ kurz vor B

Länge der Contention Period II

- B sendet, es kommt zu Kollision



- Kollision kommt nach τ bei A an
- $->$ 2τ bis ein Sender sicher sein kann, dass Kanal ihm gehört

Kollisionserkennung, kurz

- ist ein analoger Prozess
- Hardware tastet Kanal beim Senden ab
- $S_{gesendet} \neq S_{gelesen} \rightarrow$ Kollision
- bei kabellos: Signalstärke kann sich um Faktor 1 Mio. unterscheiden
- Modulation muss passend gewählt werden
  - $0V + 0V = 0V$

CSMA/CD Verfahren

- vor dem Senden: überprüfe Kanal
  - wenn frei: sende und prüfe auf Kollision
    - bei Kollision, breche ab und warte zufällige Zeit
    - abbrechen: JAM-Signal
  - wenn belegt: warte bis frei

Versuch: CSMA/CD Namen

- Namen ohne Unterbrechungen in der Gruppe sagen
- Augen geschlossen
- Keine Reihenfolge!
- Kollisionen selbst hören –> Zufällige Zeit warten
- Wie viele Namen in 30 Sekunden?

Zusammenfassung

- CSMA: Prüfe, ob der Kanal frei ist
- Kollisionen durch Signallaufzeit
- 1-persistent: Sende sofort wenn frei
- nonpersistent: Warte zufällig wenn belegt.
- p-perrsistent: Sende wenn frei mit Wahrscheinlichkeit p
- CSMA/CD: Kollision erkennen: gelesen $\neq$ gesendet

Kollisionsfreie Protokolle

- Kollisionen möglich, vor allem bei großem τ
- Kollisionen steigern Latenz
- –> Kollisionen einfach verbieten
- Annahme: N Sender mit eindeutigen Adressen von 0 bis N - 1

Bitmap Protokoll

Setup:

- contention period aus N Intervallen (N Bit)
- wenn Station 0 senden möchte: sende 1 in Intervall 0
- nur Station 0 darf in Intervall 0 senden
- nach contention period wissen alle, wer senden möchte

Senden:

- jede Station sendet ihren Frame in Adressreihenfolge

Token Passing

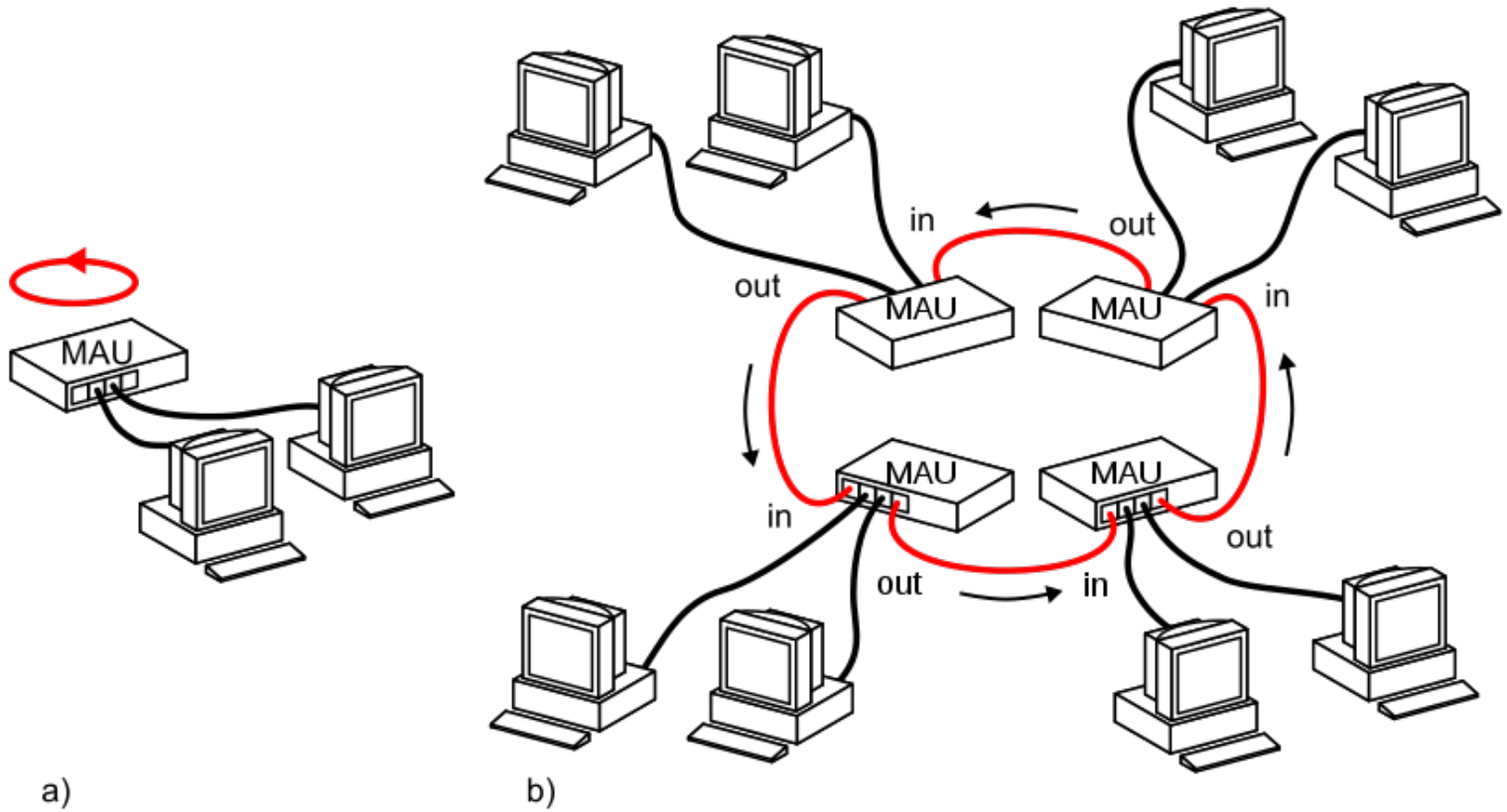
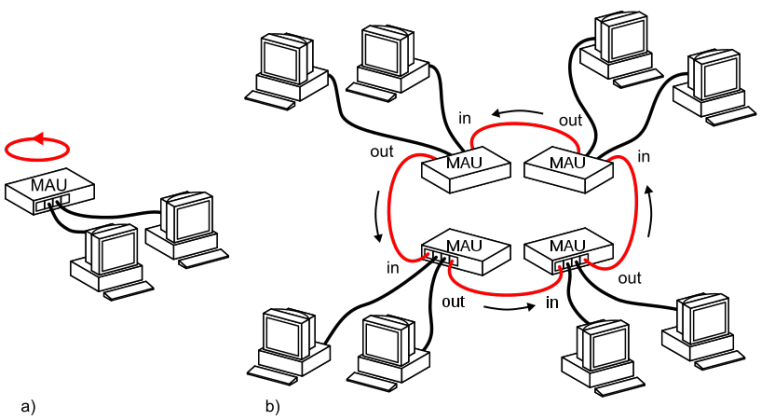


Abbildung: Token Ring, Andrew28913, cc by-sa, 2014, Wikimedia coomns, Token_ring.svg

Token Passing



- ein Token wird in Adressreihenfolge von Sender zu Sender weitergereicht
- wer senden möchte, sendet und gibt anschließend das Token weiter
- sonst wird das Token einfach weitergereicht
- MAU: Multistation Access Unit

- bei token ring wird die physische Verkabelung für die Reihenfolge verwendet
- Protokoll aber auch bei Bus

Versuch: Token Namen

- Token durch Blick in Reihenfolge weitergeben
- Namen ohne Unterbrechungen in der Gruppe sagen
- Wie viele Namen in 30 Sekunden?

Im Chat Token alphabetisch weitergeben (nächsten Namen nennen)

Probleme bei Bitmap und Token Passing

- fixer Overhead proportional zu N
- –> skaliert nicht gut

Versuch: Token Antworten

- Token durch Blick in Reihenfolge weitergeben
- Anfang: Zufälliger Name
- Schritt: Wenn Token bei Person mit Namen: zufälligen Namen nennen.
- Wie viele Namen in 30 Sekunden?

Im Chat Token alphabetisch weitergeben (nächsten Namen nennen)

Binary Countdown

- wer senden möchte sendet die eigene Adresse: A
- alle senden ihre Adresse gleichzeitig
- einzelnen Bits der Adresse werden `or` verknüpft: R
- Bits von A und R werden einzeln verglichen
  - ist $R_i = 1$ und $A_i = 0$ gibt die Station auf
- die letzte Station bekommt den Kanal
- Overhead proportional zu `len(A)`

Beispiel: Binary Countdown, Anfang

A	0	1	2	3
0010				
0100				
1001				
1010				

Beispiel: Binary Countdown, Gesendet

A	0	1	2	3
0010	0	-	-	-
0100	0	-	-	-
1001	1	0	0	-
1010	1	0	1	0
R	1	0	1	0

Zusammenfassung

- **Bitmap**: Jede Runde melden, wer senden will (Contention-Header)
- **Token-Ring**: Token im Kreis, wer es hat darf senden.
- **Binary Countdown**: Skaliert min log(N), binäre Hierarchie

Limited Contention Protocols

Bisher:

- ALOHA:
  - kaum Verzögerung wenn Kanal frei ist
  - bei steigender Last sinkt die Effizienz
- Kollisionsfreie Protokolle
  - viel Overhead bei wenig Last
  - bei steigender Last wird Overhead vernachlässigbar
- Idee: Kombination

Erfolgsrate

- weniger bereite Stationen –> höhere Erfolgrate
- Idee: Stationen in Gruppen unterteilen
  - Gruppen wechseln sich kollisionsfrei ab
  - innerhalb der Gruppe können Kollisionen auftreten

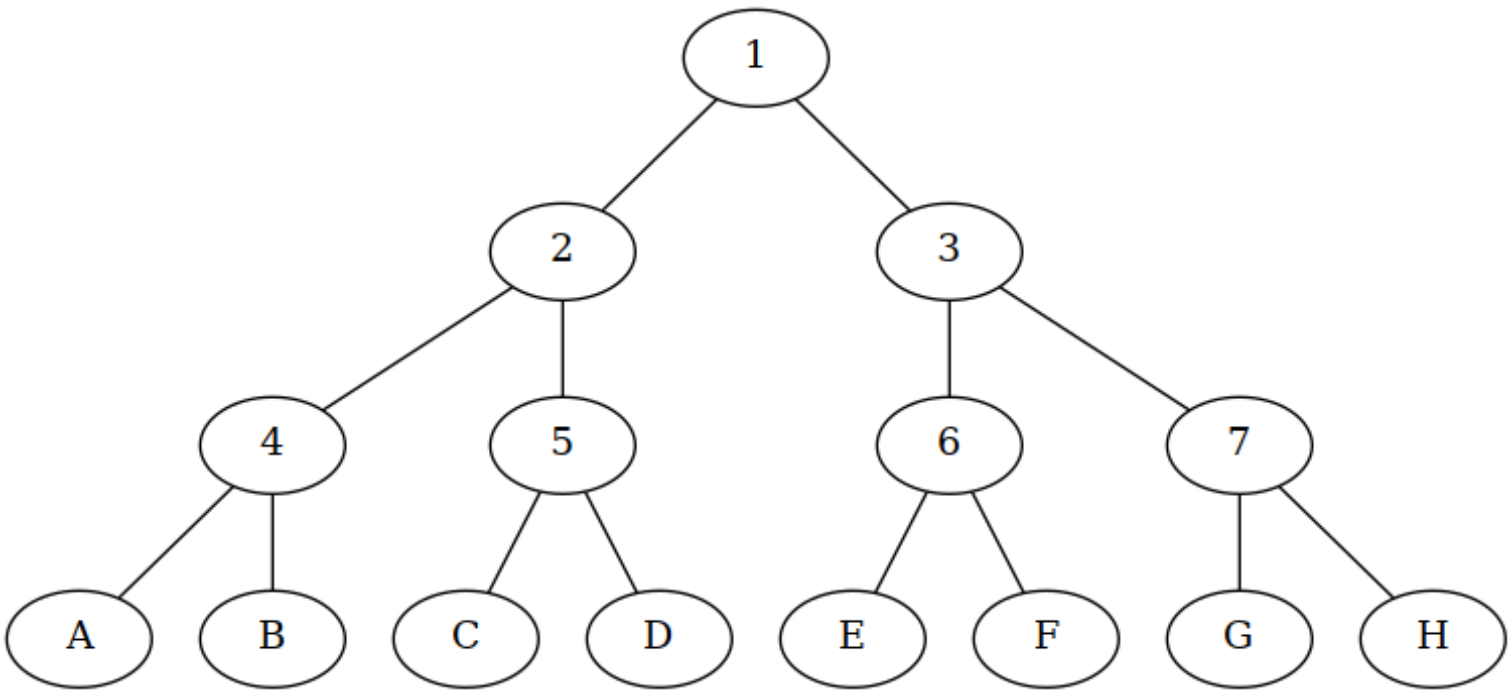
Wie Gruppen bilden?

2 Extreme:

- jede Gruppe hat eine Station –> kollisionsfreies Protokoll
- eine Gruppe mit allen Stationen –> z.B. slotted ALOHA

Tradeoff zwischen Kollisionschance und Overhead.

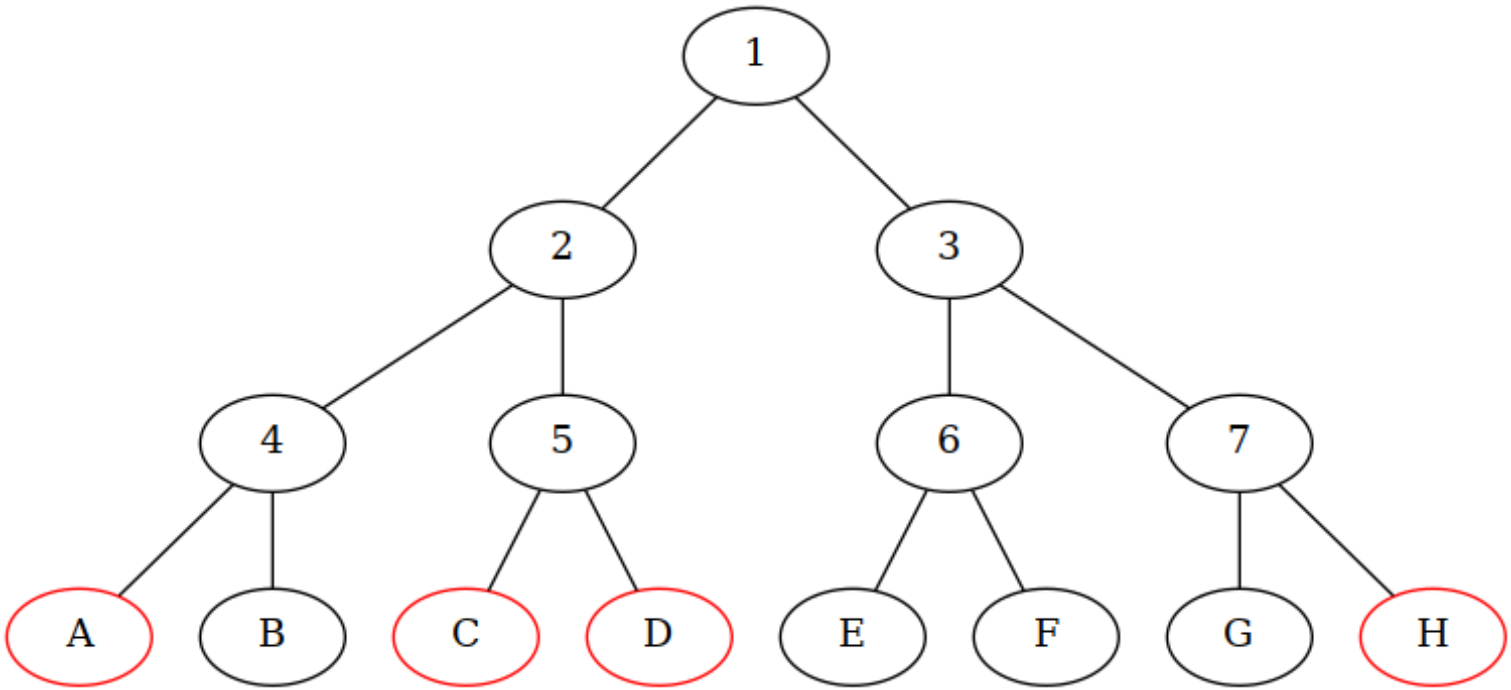
Adaptive Tree Walk Protocol I



Adaptive Tree Walk Protocol II

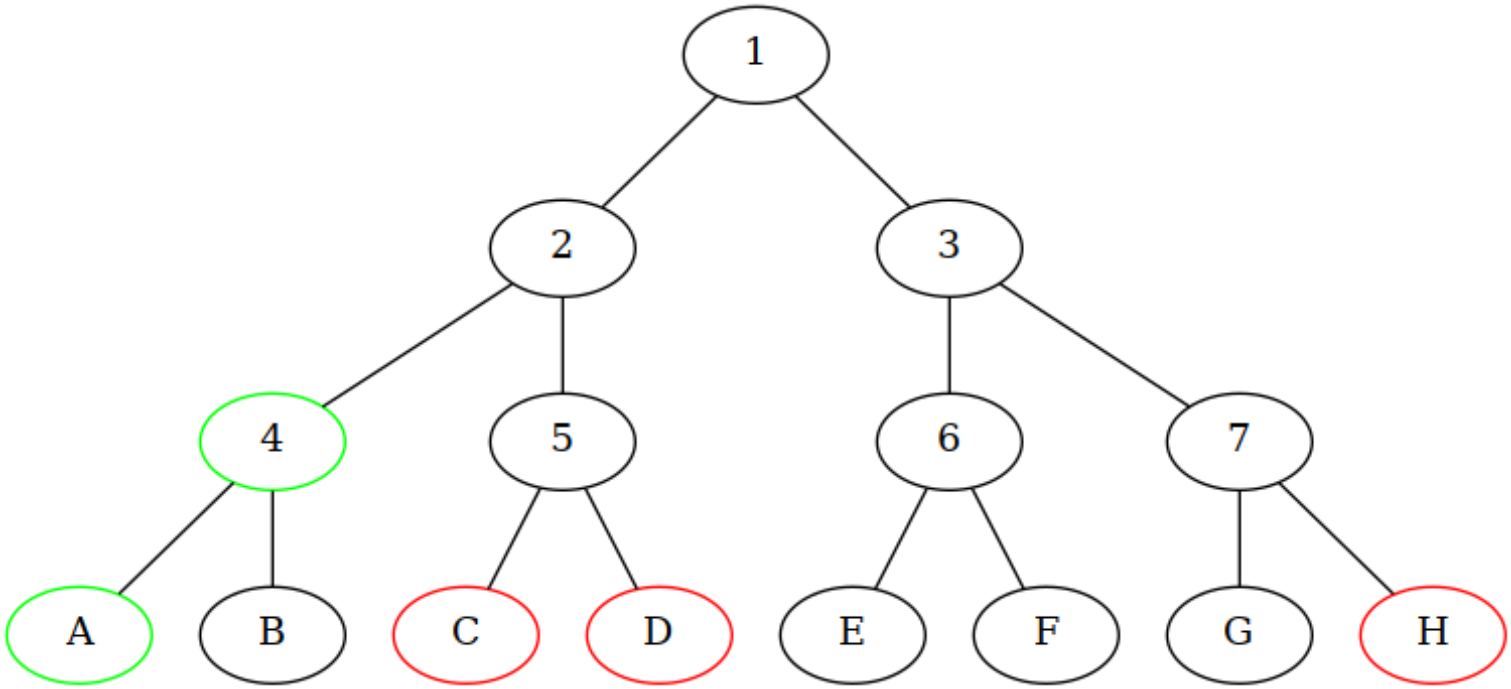
- bei hoher Last lohnt es sich nicht die Suche auf Level 0 zu beginnen
- Frage: ab welchem Level lohnt es sich?
- Annahme: jeder kann Anzahl bereiter Sender q abschätzen
- jeder Knoten auf Level i hat $2^{-i}N$ Stationen unter sich
- bei gleichmäßiger Verteilung der q Stationen:
  - befinden sich unter einem Knoten $2^{-i}q$ bereite Stationen
- optimales Level: durchschnittlich nur 1 Station bereit
  - $-> 2^{-i}q = 1 -> i = \log_2 q$

Adaptive Tree Walk Protocol III



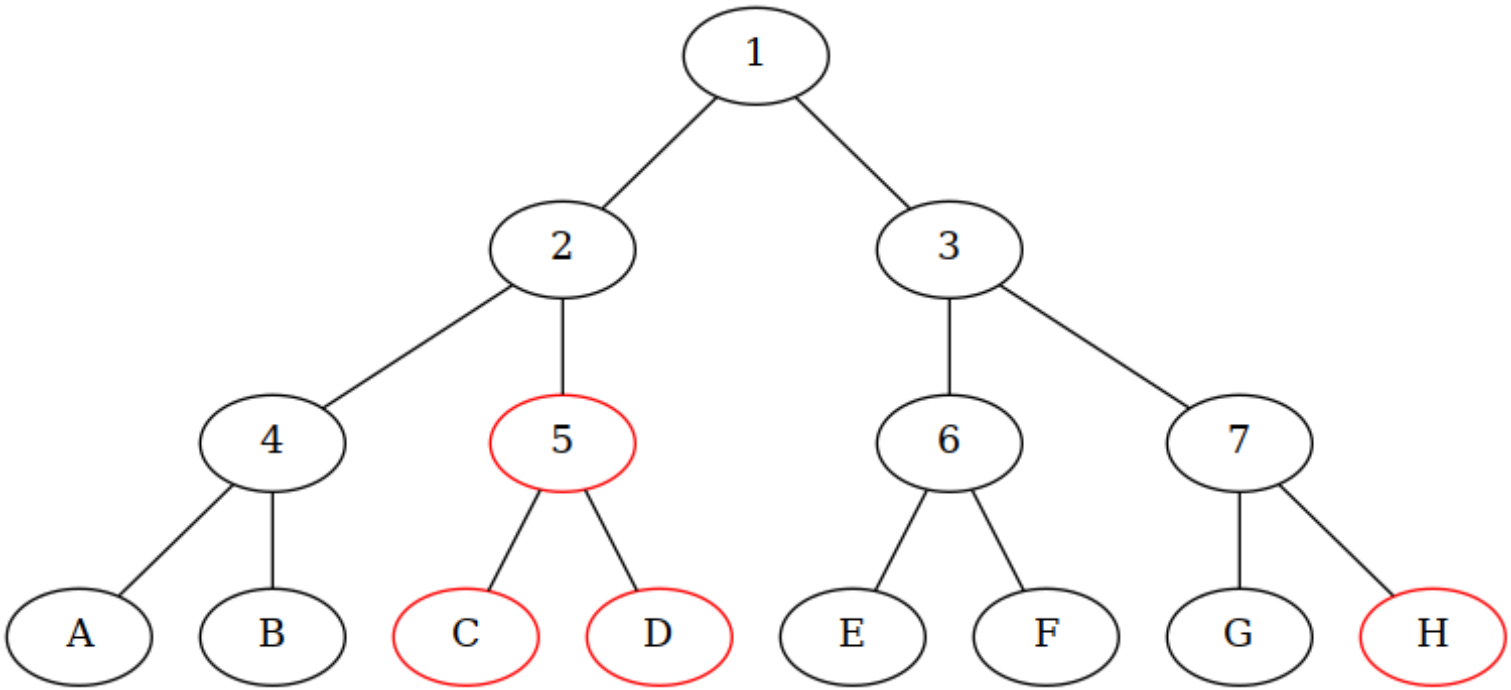
- A, C, D, H möchten senden
- $q = 4 \rightarrow i = \log_2(4) = 2$

Adaptive Tree Walk Protocol IV



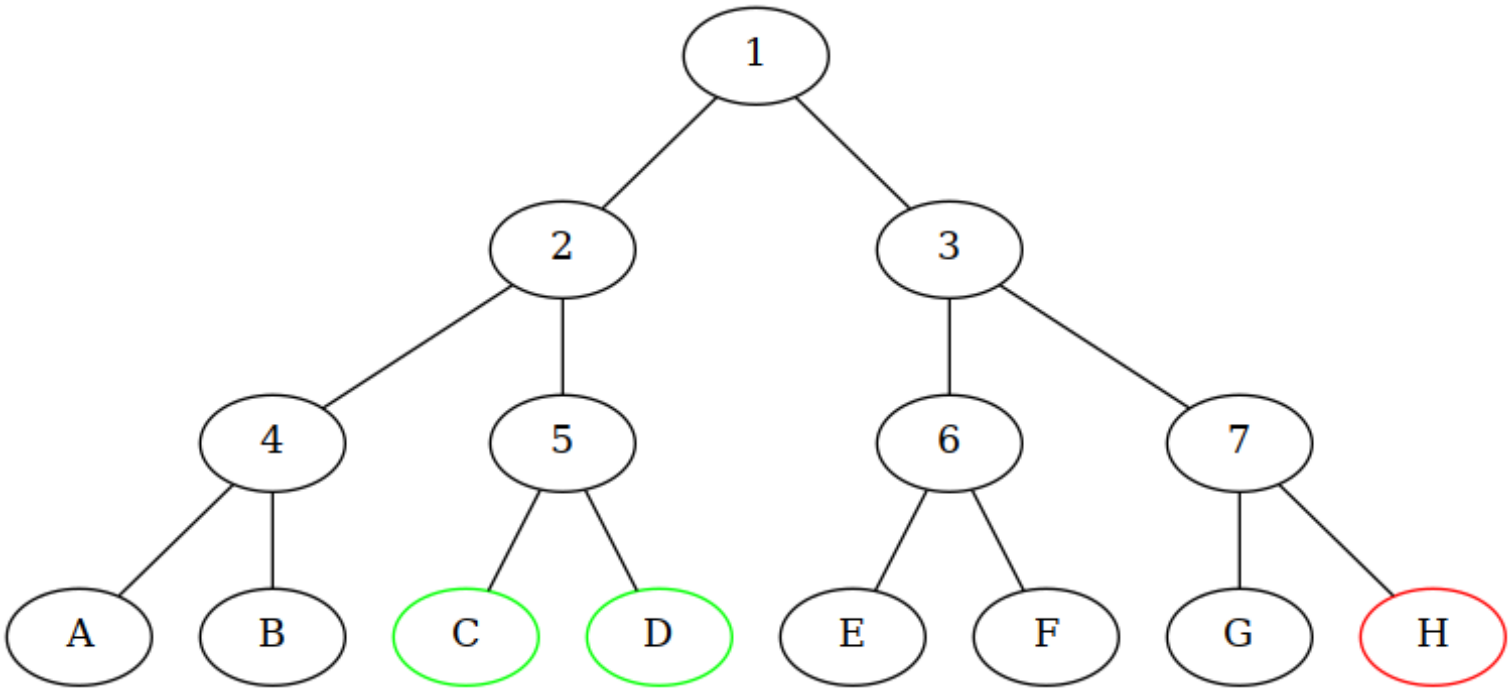
- keine Kollision in 4
- A sendet erfolgreich in Slot 0

Adaptive Tree Walk Protocol V



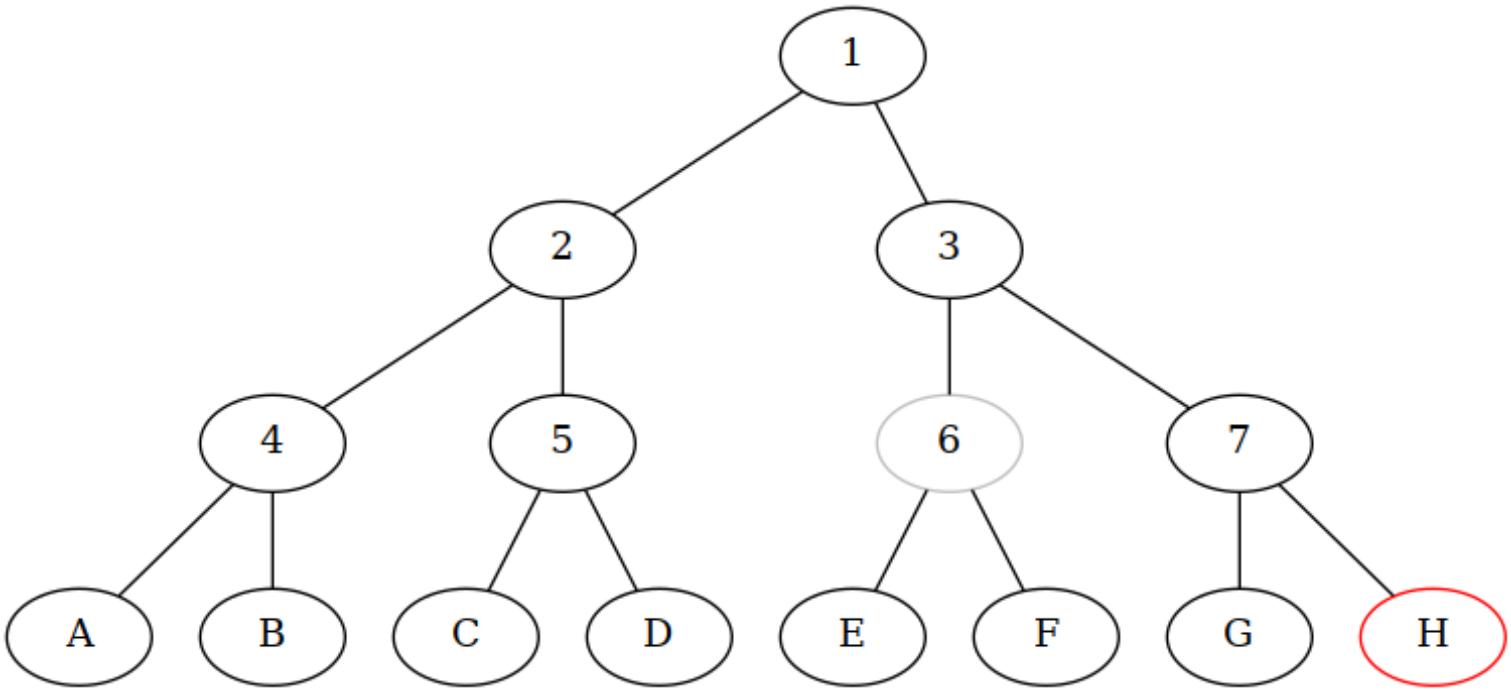
- Kollision in 5 in Slot 1
- –> die Kinder von 5 sind in den nächsten Slots dran

Adaptive Tree Walk Protocol VI



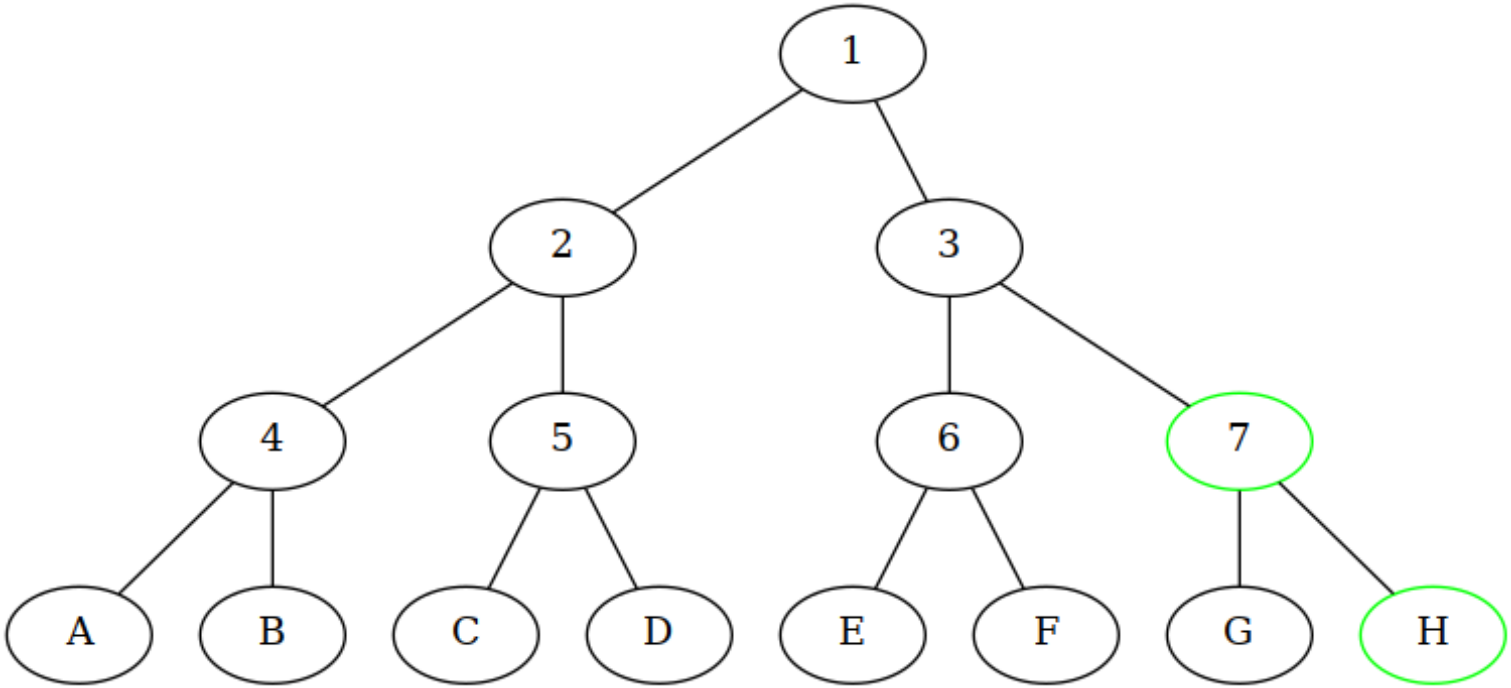
- C sendet in Slot 2
- D sendet in Slot 3

Adaptive Tree Walk Protocol VII



- Stationen unter 6 senden in Slot 4
  - keiner der Knoten sendet etwas, der Slot ist idle

Adaptive Tree Walk Protocol VIII



- Stationen unter 7 senden in Slot 5
- Station H sendet

Zusammenfassung

- Kombination zwischen ALOHA und Kollisionsfrei
- Baumstruktur zur Gruppenbildung
- Gruppierungstiefe nach Last
- Kollisionen innerhalb der Gruppe teilen Gruppe

WLAN Protokolle

- Carrier Sense nicht möglich
- Kollisionen werden durch fehlende Bestätigung erkannt
- nicht jede Station kann von jeder anderen Station empfangen
  - –> 2 Probleme

Hidden Terminal Problem

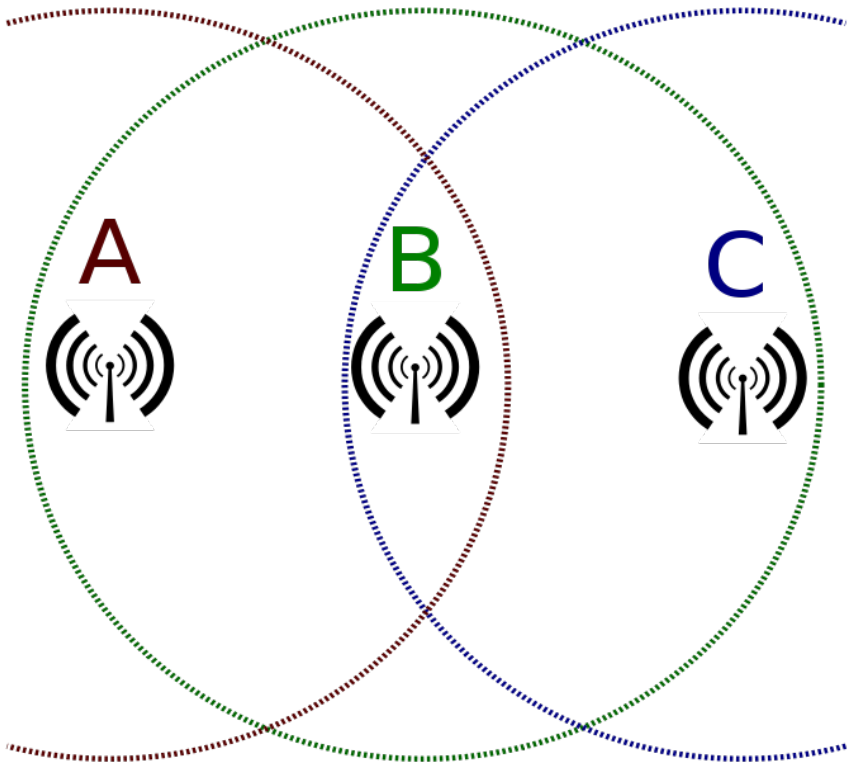


Abbildung: Hidden Terminal, Andrei Stoe, cc by-sa / art libre
Wifi_hidden_station_problem.svg

Ein Sender kann einen möglichen Mitbewerber nicht sehen.

Exposed Terminal Problem

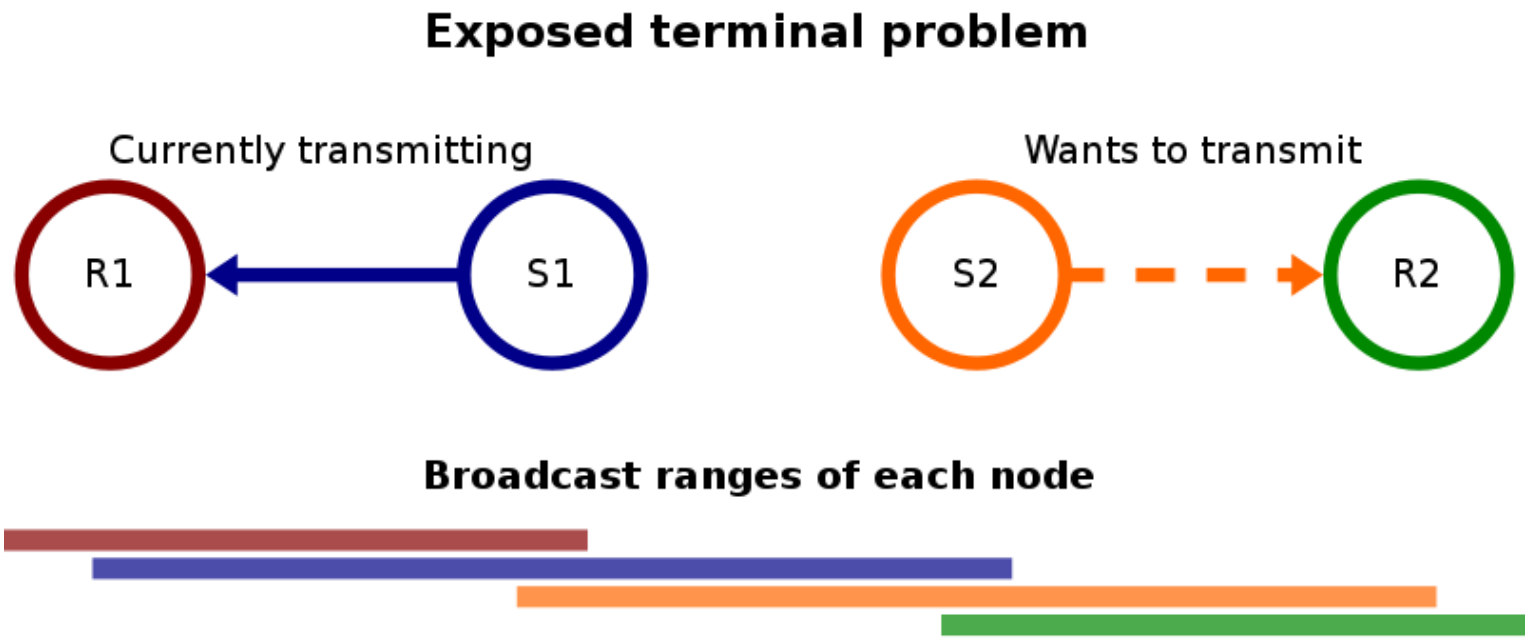


Abbildung: Exposed Terminal, Fibonacci and ALM scientist, public domain Exposed_terminal_problem.svg

Ein Sender sieht ein Signal, das für einen anderen Empfänger als den des Senders bestimmt ist.

MACA (Multiple Access with Collision Avoidance)

Allgemein: im WLAN ist nur das Medium in der Nähe des Empfängers wichtig.

- A möchte an B senden
- A sendet RTS (Request to Send)
- B antwortet mit CTS (Clear to Send)
- jede Station, die RTS hört, darf für die CTS Zeit nicht senden
- jede Station, die CTS hört, darf für die Frame Zeit nicht senden

Zusammenfassung

- Hidden Terminal: Anderen Sender nicht gesehen $\Rightarrow$ Unsichtbare Kollision
- Exposed Terminal: Nachricht an unsichtbaren Empfänger $\Rightarrow$ Unechte (irrelevante) Kollision
- MACA: Protokoll, das unter diesen Einschränkungen funktioniert

Ethernet

- durch Metcalfe und Boggs bei Xerox Parc entwickelt
  - –> classic Ethernet
- viele verschiedene Standards (Ethernet, fast Ethernet, Gigabit ...)
  - zuerst durch DEC, Intel und Xerox (DIX) standardisiert
  - standardisiert durch IEEE
- wir schauen uns nur classic Ethernet an

Ethernet Frames I

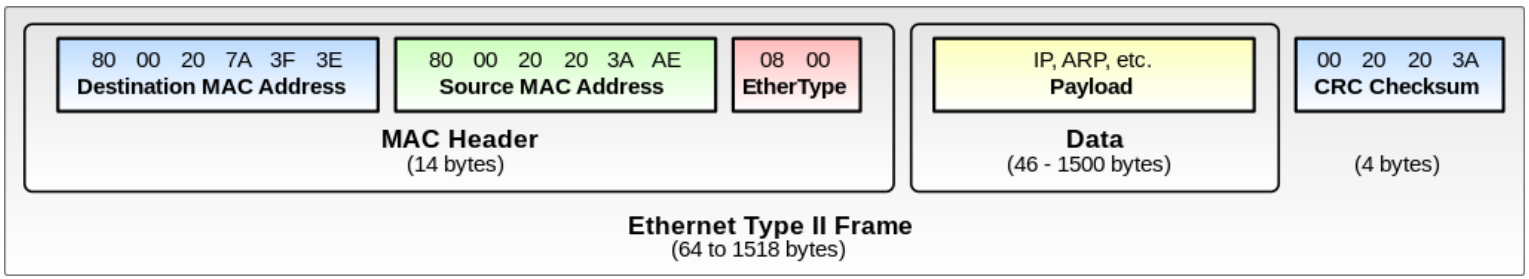


Abbildung: Ethernet Frames Type II (DIX), Mikm, public domain Ethernet_Type_II_Frame_format.svg

Ethernet Frames II

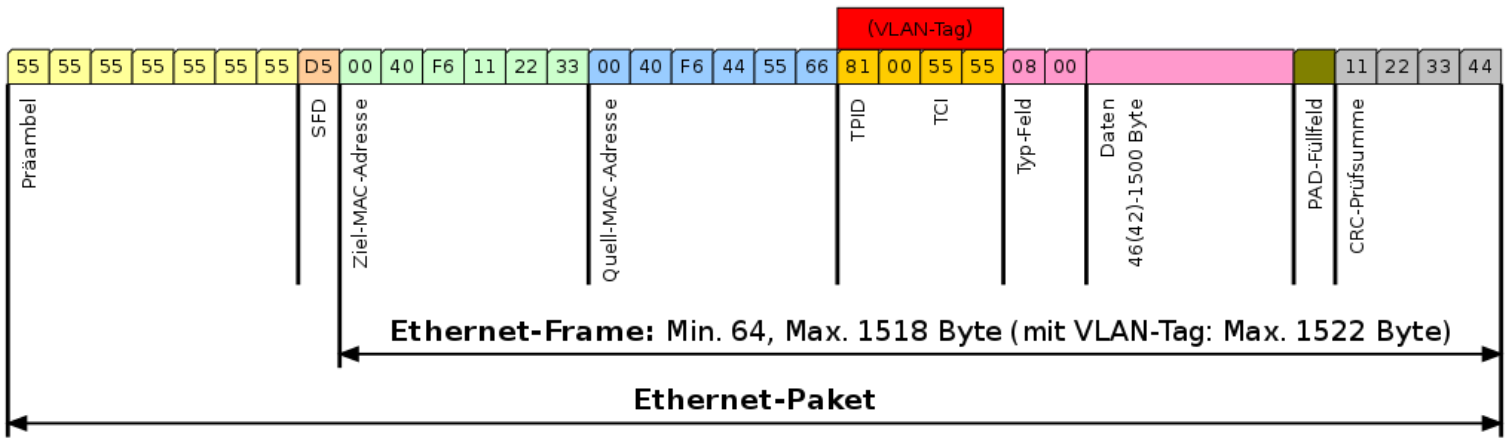


Abbildung: Ethernet-II-Frameformat aus IEEE 802.3, inklusive VLAN-Tag aus IEEE 802.1Q, Bluepoke, cc by-sa, Ethernetpaket.svg

Ethernet Frames III

Preamble:

- 8 Bytes mit Muster 10101010
  - mit Ausnahmer der letzten 2 Bit (11) (SDF)
- Manchester Code: erzeugt Welle zur Synchronisation
- 1-persistent CSMA/CD wird verwendet

Ethernet Frames IV

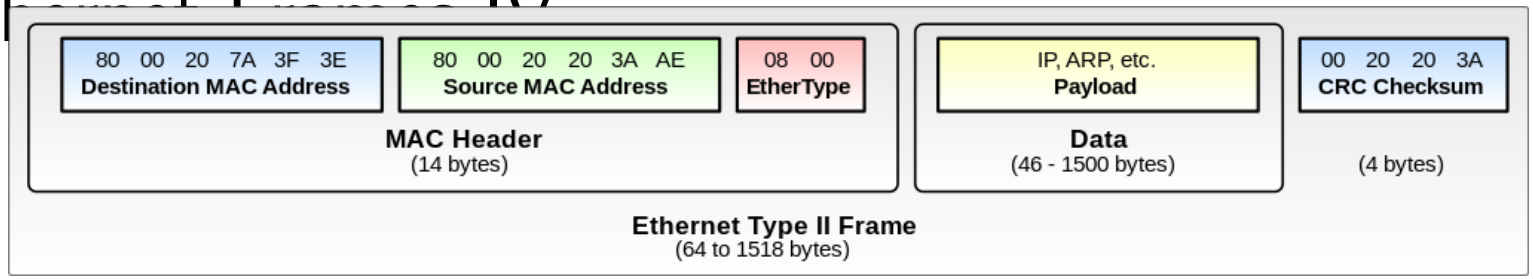


Abbildung: Ethernet Frames Type II (DIX), Mikm, public domain Ethernet_Type_II_Frame_format.svg

Destination Address, Source Adress:

- 6 Byte = 48 Bit lang
- `DE:AD:BE:EF:CA:FE`
- weltweit eindeutig
- ersten 3 Byte werden von IEEE den Herstellern zugewiesen
- letzten 3 Byte werden vom Hersteller gesetzt
  - **Update**: Seit Android 12: randomisierte MAC-Adressen!
- Destination:
  - erstes Bit 1: multicast, sonst unicast
  - alle Bits auf 1: broadcast

Ethernet Frames V

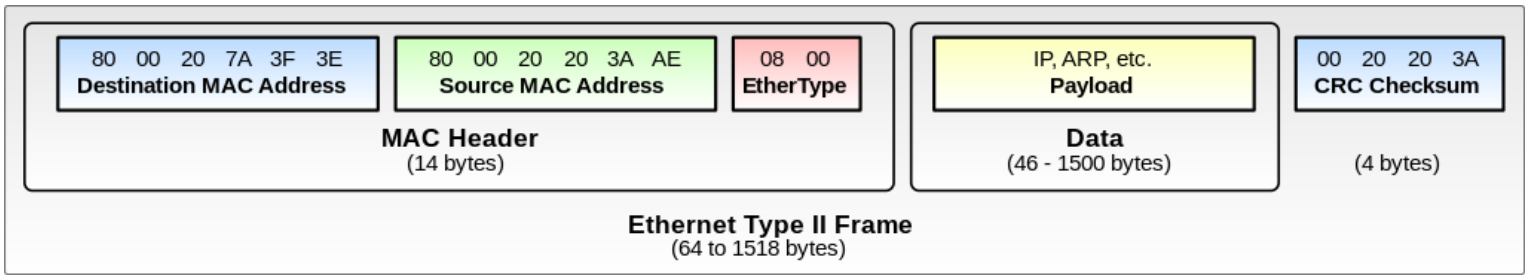


Abbildung: Ethernet Frames Type II (DIX), Mikm, public domain Ethernet_Type_II_Frame_format.svg

Type oder Length:

- Type gibt Pakettyp an (0x0800 IPV4)
- Length gibt Länge der Daten an
- Werte größer 0x0600 (1536) sind Types
- kleinere Werte sind Lengths
- ermöglicht Kompatibilität zwischen DIX und IEEE

Ethernet Frames VI

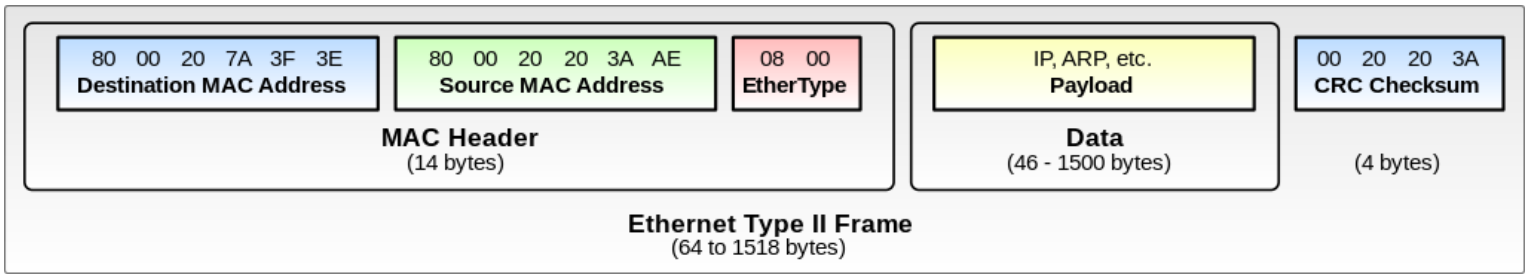


Abbildung: Ethernet Frames Type II (DIX), Mikm, public domain Ethernet_Type_II_Frame_format.svg

Data und Pad:

- maximale Länge von 1500 Bytes ist willkürlich gewählt
- Mindestlänge sind 64 Bytes
  - wird durch Pad Feld garantiert
- Mindestlänge resultiert aus maximaler Kabellänge und τ

Ethernet Frames VII

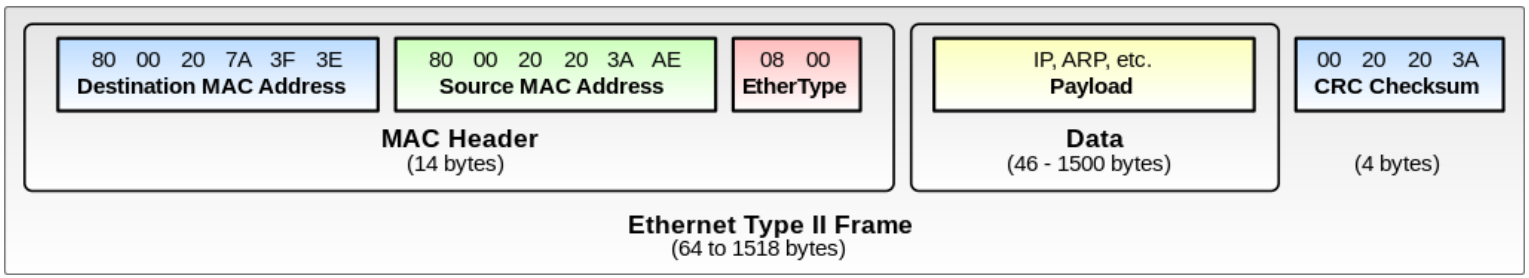


Abbildung: Ethernet Frames Type II (DIX), Mikm, public domain Ethernet_Type_II_Frame_format.svg

Checksum:

- 32 Bit CRC Prüfsumme
- wird über Destination Adress bis Pad berechnet

CSMA/CD mit Binary Exponential Backoff

- 1-persistent CSMA/CD
- sobald Medium frei: senden
- bei Kollision wird zufällig lang gewartet
- Binary Exponential Backoff
  - nach Kollision wird Zeit in Slots mit der Länge 2τ unterteilt
  - nach der ersten Kollision wird entweder in Slot 0 oder 1 erneut versucht
  - bei erneuter Kollision: Slots 0, 1, 2 oder 3
  - Allgemein: nach i Kollisionen stehen 2^i Slots zur Verfügung
  - maximal 1024 Slots nach 10 Kollisionen
  - nach 16 Kollisionen gilt die Übertragung als gescheitert
- dynamische Anpassung an momentane Last

Kollisionsdomäne

- verschiedene Verkabelungen nach verschiedenen Standards erlaubt
- Bus Topologie mit Coax Kabel
- TP Kabel mit Hubs
- TP Kabel mit Switches
- LWL auch erlaubt, aber für Kollisionsdomäne uninteressant

Zusammenfassung

- Präambel: Bitmuster + x
- MAC-Adresse: 6 Byte, Hersteller + Gerät
- Quell- und Zieladresse
- Typ + Länge
- Mindest-Länge aus maximaler Kabellänge (Kollisionen erkennen)
- Prüfsumme (CRC)

Data Link Layer Switching

- Bridges (ab hier Switches genannt) verbinden unterschiedliche LANs miteinander
- arbeiten im Data Link Layer
  - inspizieren Destination und Source Adress
- Verwendung soll transparent sein
  - einfach ein- und ausstecken
  - ohne Änderungen an Hard- oder Software
- ermöglicht durch 2 Algorithmen

Store Forward vs Cut Through

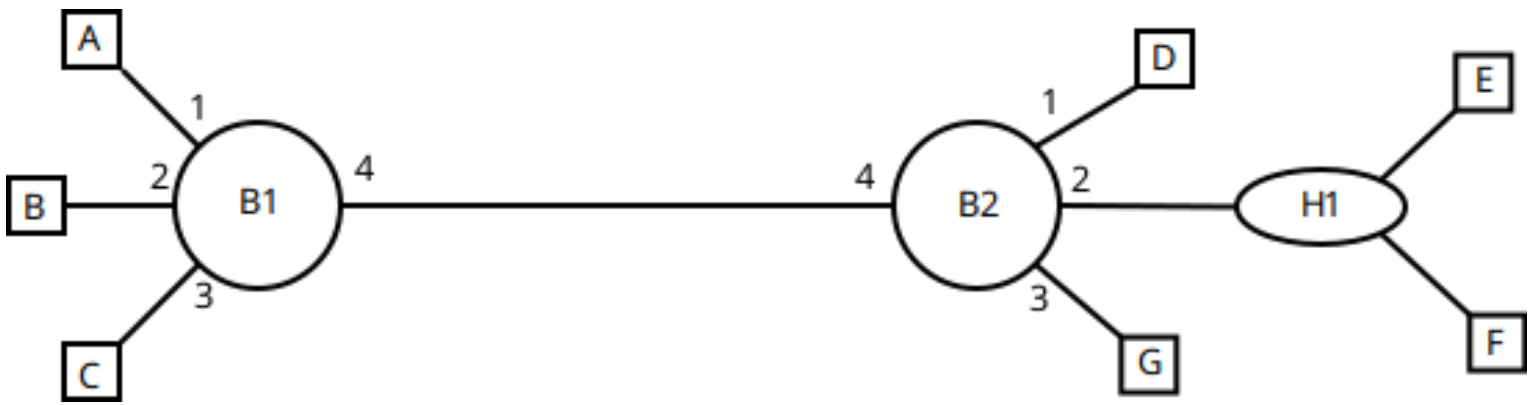
SwitchModi:

- Store Forward: Frame wird empfangen und intern gebuffert, erst wenn Frame komplett empfangen wurde, wird er ausgegeben
- Cut Through: sobald Zieladresse des Frames gelesen wurde, wird mit der Ausgabe begonnen
  - –> kleiner Buffer notwendig
  - –> weniger Latenz

Backward Learning I

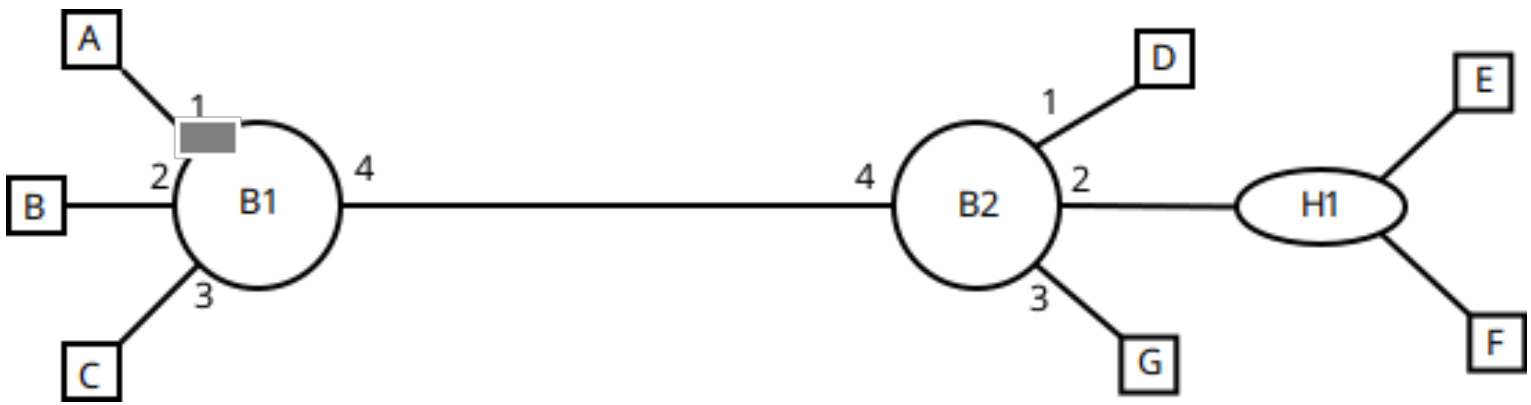
- Switches akzeptieren alle Frames an ihren Ports
- Frames werden an entsprechendem Port weitergeleitet
- Tabelle von Adresse zu Port
- zuerst Tabelle leer
- wenn ein Frame kommt wird er auf allen Ports bis auf den Eingangsport ausgegeben
  - Source Adress des Frames wird in Tabelle eingetragen
- Eintrag wird noch mit Zeit des letzten Frames versehen
- periodisch werden alte Einträge gelöscht

Backward Learning II



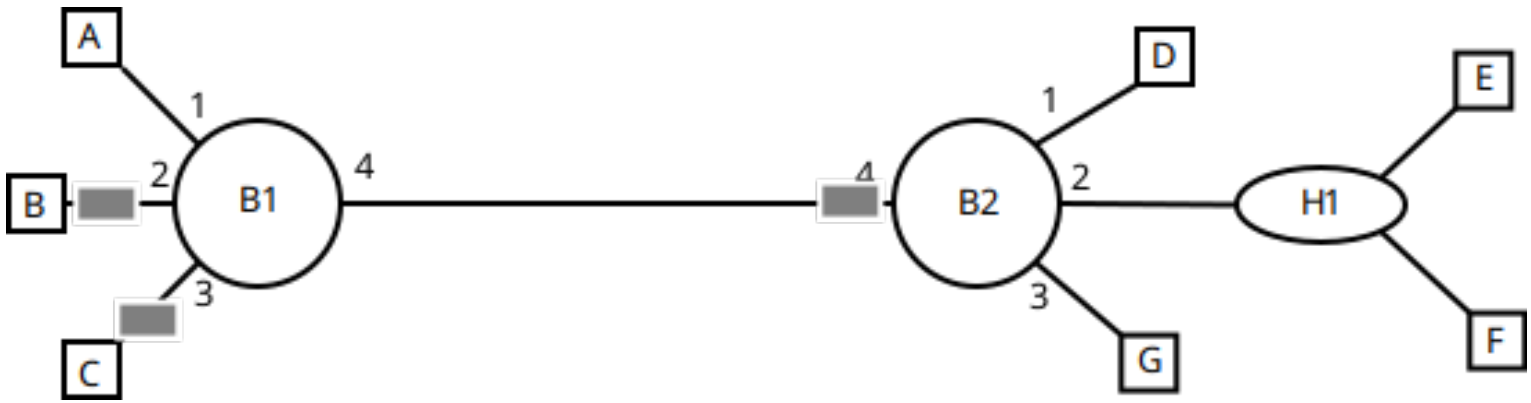
- Frame von A zu C

Backward Learning III



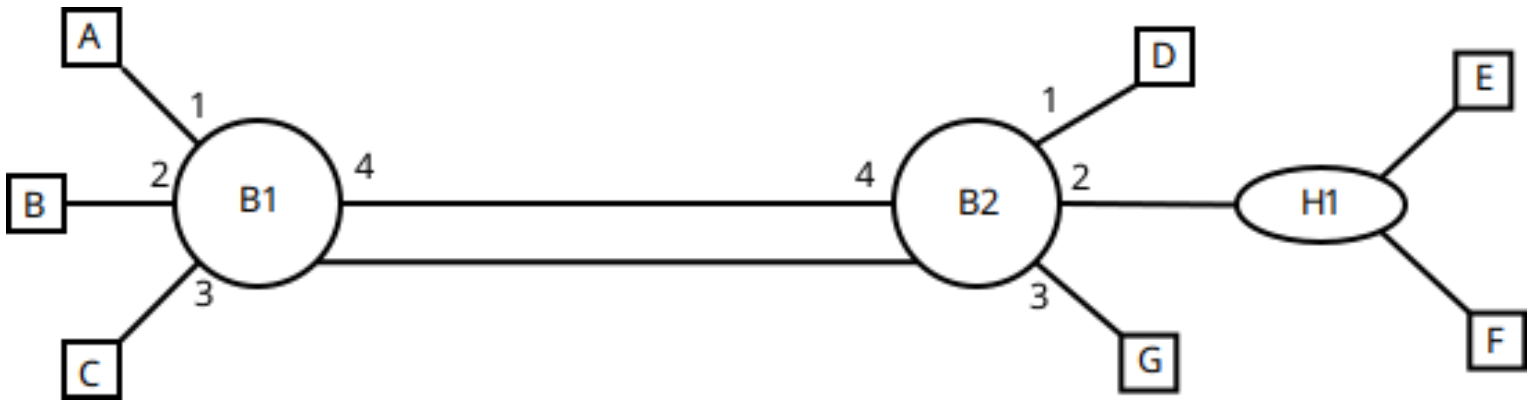
- B1 ordnet Port 1 zu A

Backward Learning IV



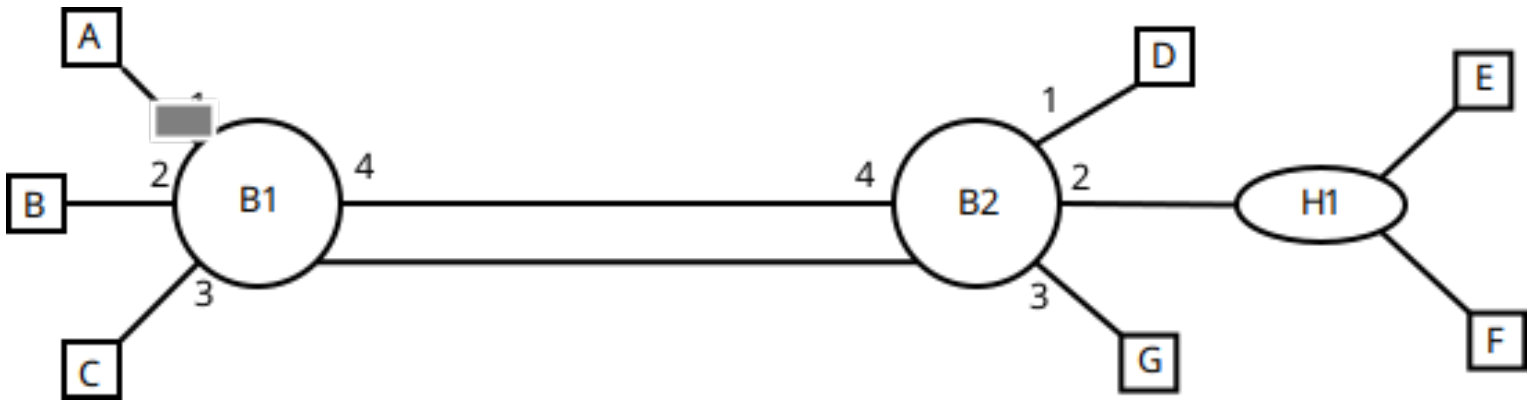
- B1 sendet Frame an alle Ports außer 1
- C empfängt Frame
- B2 ordnet Port 4 zu A

Spanning Tree Motivation 0



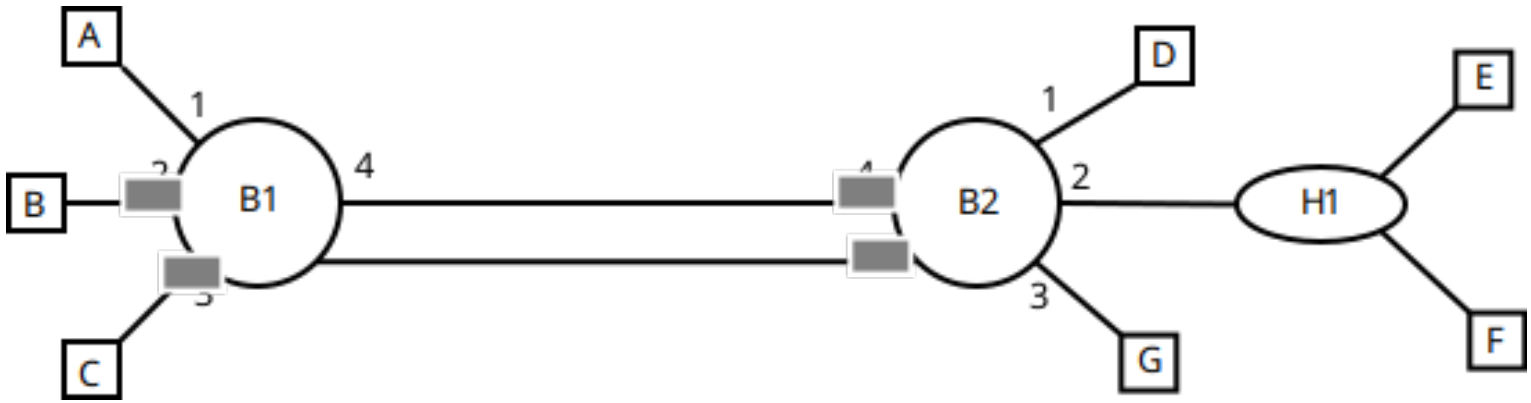
- 2 Switches sind redundant verbunden
- Tabelle von B1 und B2 jeweils leer

Spanning Tree Motivation 1



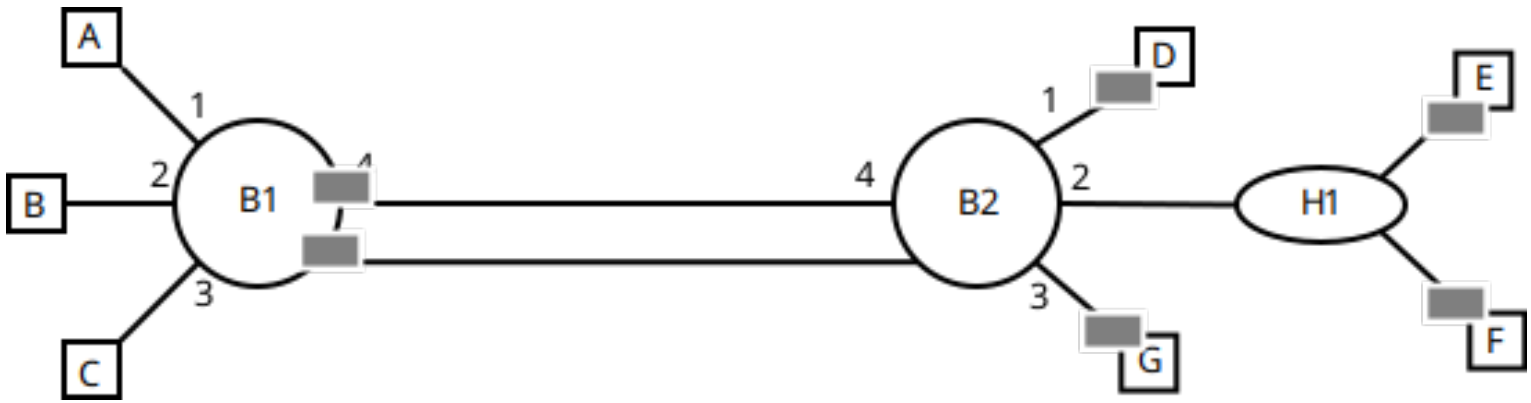
- A sendet einen Frame

Spanning Tree Motivation 2



- B1 gibt Frame an Ports aus
- B2 empfängt 2 (identische) Frames
  - kann aber nicht sehen, dass die Frames identisch sind

Spanning Tree Motivation 3

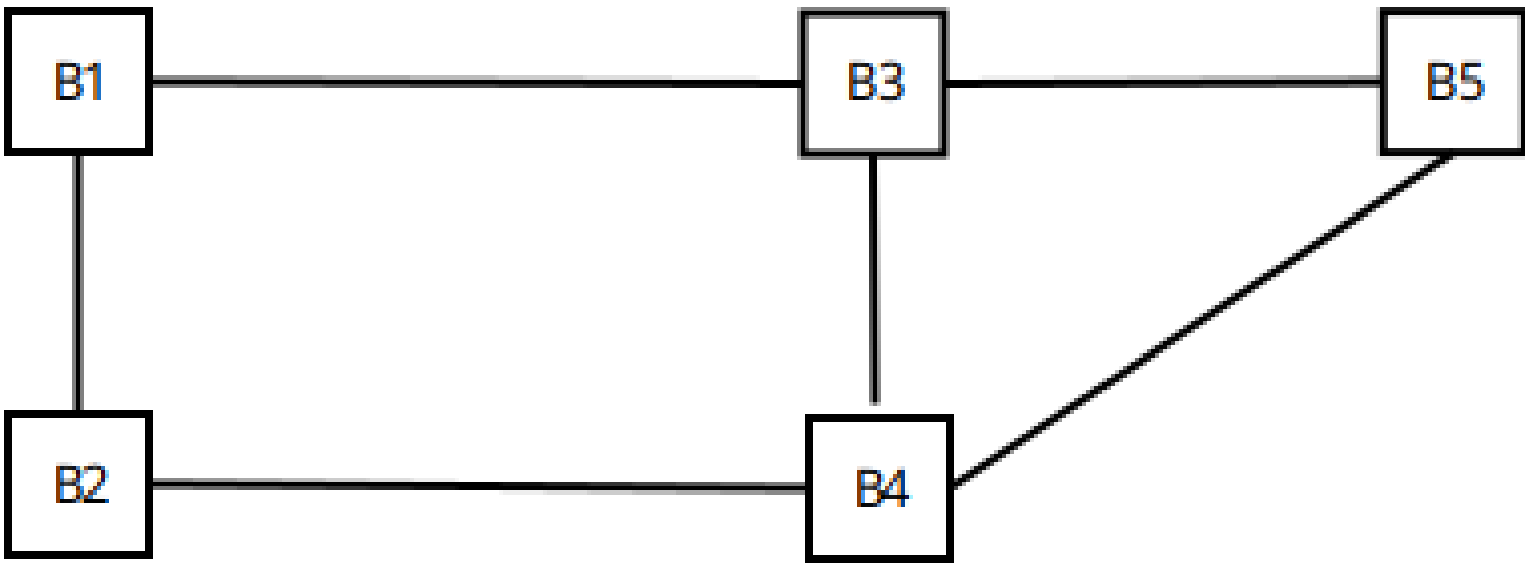


- B2 gibt Frame an Ports aus
- B1 empfängt 2 Frames

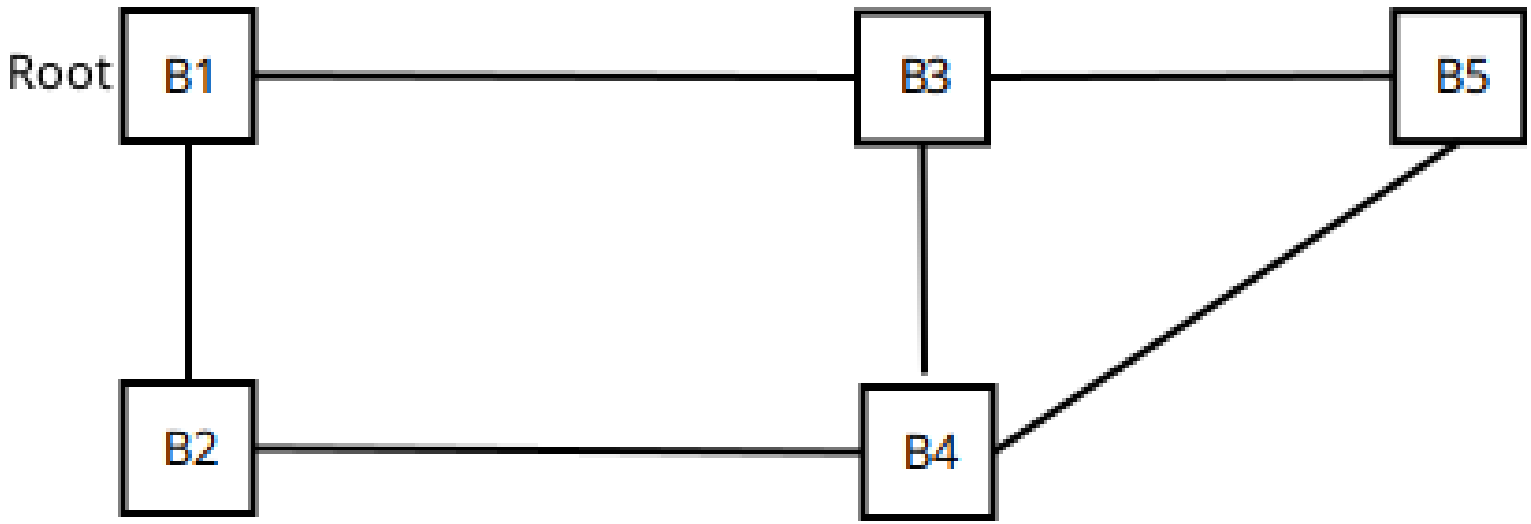
Spanning Tree Algorithmus

- von einem zyklischen Graphen wird der Spanning Tree ermittelt
- jeder Switch broadcastet regelmäßig Konfigurationsnachrichten, für den Algorithmus
- Bestimmung der Wurzel
  - jeder Switch erzeugt eine ID aus seiner MAC
  - diese ID wird mit der ID des Switches, der für die Wurzel gehalten wird versendet
  - dies wird wiederholt bis Konsens eintritt
  - die niedrigste ID wird genommen
- jetzt wird für jeden Switch der Shortest Path zur Wurzel bestimmt
  - kann ein Switch über mehrere Switches erreicht werden, wird der Switch mit der niedrigsten ID verwendet
- Algorithmus wird periodisch wiederholt, um auf Netzwerkveränderungen zu reagieren

Beispiel: Spanning Tree I

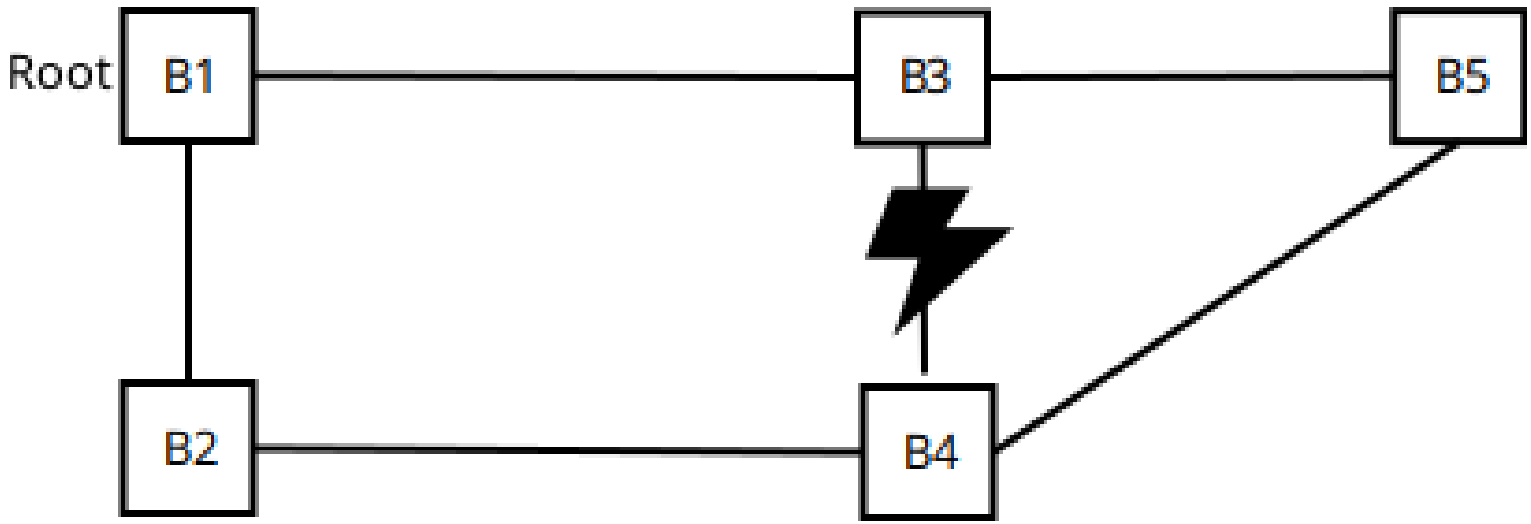


Beispiel: Spanning Tree II



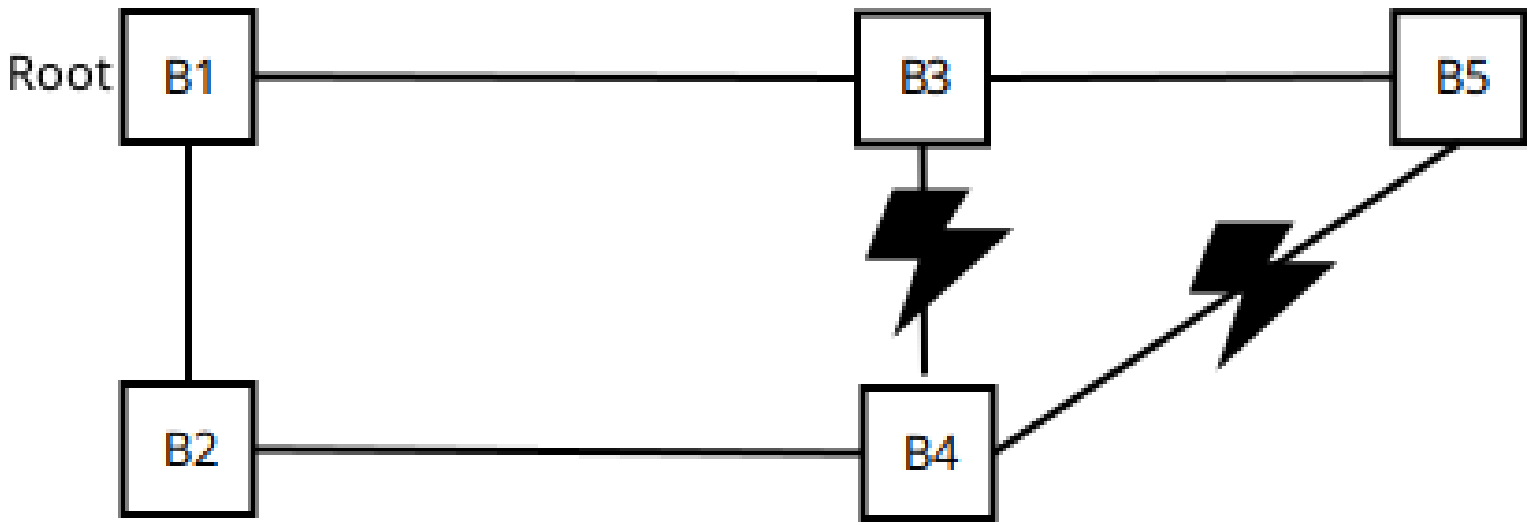
- B1 wird als Wurzel gewählt
- Shortest Path zu B2 und B3 jeweils Distanz 1

Beispiel: Spanning Tree III



- Shortest Path zu B4 über B2 oder B3 möglich
- B2 < B3 –> verwende Kante(B3, B4) nicht

Beispiel: Spanning Tree IV



- Shortest Path zu B5 über B3 möglich
- Kanten, die zu keinem Shortest Path gehören werden entfernt

VLANs

- Aufteilung der LANs richtet sich nach physischen Gegebenheiten
- meist jedoch Aufteilung noch Organigramm gewünscht
  - Sicherheit
  - Lastverteilung
- VLANs lösen dies
  - Konfiguration welcher Port zu welchem VLAN gehört
  - nur Switches müssen VLANs unterstützen (Erweiterung des Frames mit VLAN Tag)
  - Tag wird an Übergang Switch zu Computer entfernt
  - Frames werden nur an entsprechenden Ports weitergeleitet

Zusammenfassung

- Backward Learning:
  - Speichere von wo wessen Pakete ankommen
  - Anfangs kein Wissen $\Rightarrow$ Broadcast
- Spannbaum
  - Gegen Doppelpakete
  - Baumstruktur mit niedrigster ID als Wurzel

Zusammenfassung

- Dynamische Verfahren für bessere Auslastung
- ALOHA: Kollision erkannt $\Rightarrow$ zufällig warten
- Slotted ALOHA 2x so effizient, aber sync
- CSMA: Prüfen: Ist Kanal frei? 1-, non-, p-persistent
- Kollisionsfrei:
  - O(N): Bitmap, Token-Ring,
  - O(log(N)): Binary-Countdown (hierarchisch)
- Adaptive Tree Walk: Kombination nach Last
- WLAN Sichtbarkeit $\Rightarrow$ MACA
- Ethernet: Präambel, Adressen, Länge, CRC, Mindestlänge
- Backward-Learning, Spannbaum gegen Redundanz

Fragen für die Prüfung?

Ideensammlung:

-
-
-
-
-

Selbststudium diese Woche I

- Übungsblatt
- Nicht in Gruppe

Viel Erfolg bei Übungen oder Projekt!

Bearbeitung: Einzeln. Dateinamen bitte:
`Uebungsblatt_MAC_Name.pdf`.

Bei Multiple Choice Aufgaben reicht eine Lösung nach folgendem Muster:

Beispiel Aufgabe Multiple Choice

Kreuze die korrekten Aussagen an:

1. ☐ Die letzte Vorlesung war viel zu schnell
2. ☐ Sriracha passt zu allem
3. ☐ Tabs sind besser geeignet für die Einrückung von Quellcode

Beispiel Lösung Multiple Choice

1, 2

Aufgabe 1

Welche der folgenden Aussagen sind korrekt für ALOHA?

- ☐ Nach einer Kollision wird eine zufällige Zeit gewartet, um eine erneute Kollision zu vermeiden.
- ☐ Bei pure ALOHA kann es zu Kollisionen kommen, wenn die einzelnen Stationen fast zur selben Zeit senden.
- ☐ Beim Senden beobachtet eine Station das Medium, um Kollisionen zu erkennen.

Aufgabe 2

Welche der folgenden Aussagen sind korrekt für CSMA?

- ☐ Beim Senden beobachtet eine Station das Medium, um Kollisionen zu erkennen.
- ☐ Bei der Verwendung von persistent CSMA wird gesendet sobald das Medium wieder frei wird.

Aufgabe 3

Welche der folgenden Aussagen sind korrekt für CSMA/CD?

- ☐ CSMA/CD verwendet Bestätigungsframes zur Feststellung von Kollisionen.
- ☐ Eine Station muss vor dem Senden prüfen, ob das Medium frei ist.

Aufgabe 4

Vergleiche pure und slotted ALOHA hinsichtlich de Latenz bei sehr geringer Last. Bei welchem Protokoll ispt die Latenz geringer?

Aufgabe 5

Wie lange muss eine Station bei Verwendung des Bitmap Protokolls im schlimmsten Fall warten bis sie senden darf?

Aufgabe 6

Wie lange muss eine Station bei Verwendung des Binary Countdown Protokolls im schlimmsten Fall warten bis sie senden darf?

Aufgabe 7

Die 6 Stationen A-F kommunizieren mit dem MACA Protokoll. Ist es möglich, dass 2 Übertragungen gleichzeitig stattfinden?

Aufgabe 8

Erkläre Store and Forward und Cut Through Switching.

Aufgabe 9

Nehme die Netzwerkkonfiguration aus der Folie "Beispiel: Backward Learningäls Ausgangspunkt. Die Tabellen der beiden Switches B1 und B2 sind anfangs leer. Die nachfolgenden Übertragungen finden nacheinander statt. Notiere jeweils die Ports von B1 und B2 auf denen Frames ausgegeben werden.

1. A -> C
2. E -> F
3. F -> E
4. G -> E
5. D -> A
6. B -> F

Aufgabe 10

Zeichne den Spannig Tree für das Netzwerk aus der Folie "Beispiel: Spanning Tree". Allerdings wurden die Switches B1 und B5 vertauscht.