
Mobile Ad-hoc Netze

Optimized Link State Routing (OLSR)

Elisa, Leon, Cliff, Binh, Jannis, Patrick - 25.04.22



Warum OLSR?

Wir haben uns für das OLSR-Routing Protokoll entschieden, da es eine verbesserte Version des in der Aufgabenstellung vorgeschlagenen Link State Routings ist. Wir haben es während unserer Recherche der anderen vorgeschlagenen Algorithmen entdeckt und hielten es für sehr geeignet, da es besonders bei dynamischen und dicht besiedelten Netzen empfohlen wird. Außerdem ist die Netzwerkauslastung durch die MPRs sehr gering.

Inhaltsverzeichnis

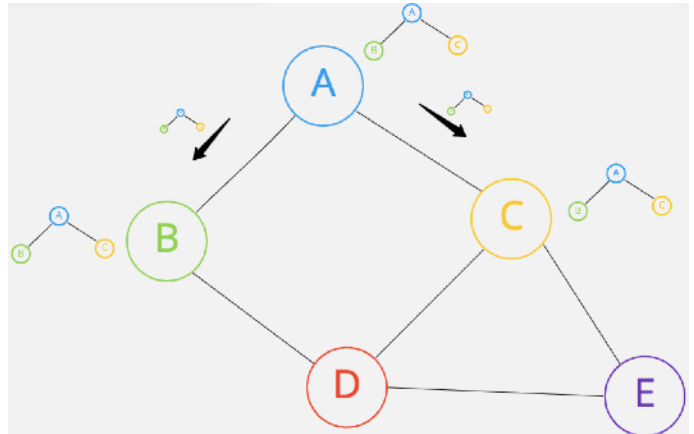
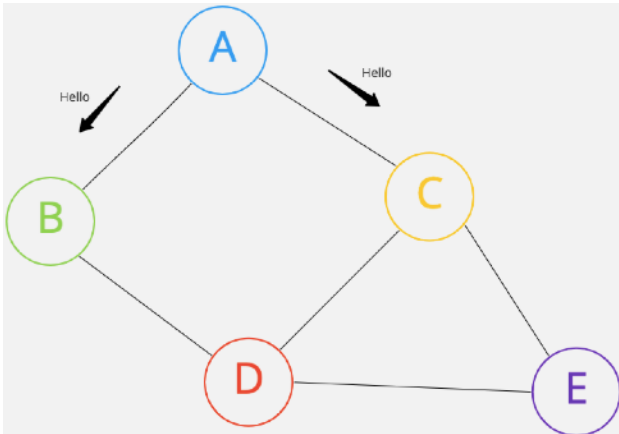
Warum OLSR?	1
Inhaltsverzeichnis	2
Einführung	3
Knoten	5
Multipoint-Relay und deren Auswahl	7
Informationsspeicherung	8
Testfälle	11
Vorteile des OLSR und der MANets	15
Nachteile des OLSR und der MANets	17
Lösungsansätze	18
Alternativen	19
Implementierung und Visualisierung	23
Literaturverzeichnis	24

Anmerkung

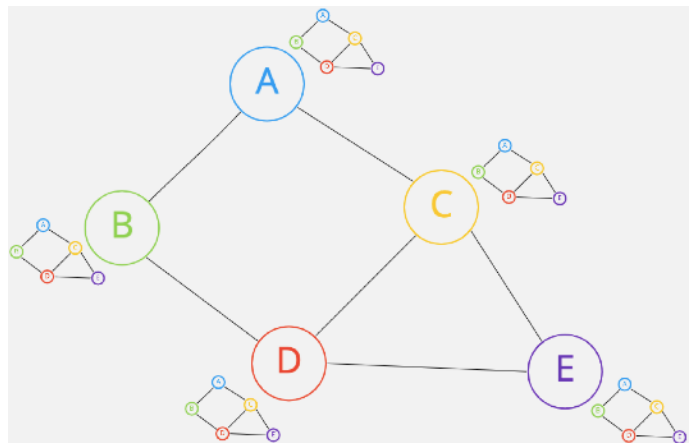
Der Link zum Github und die Aufgabenaufteilung sind dem Abschnitt „Implementierung und Visualisierung“ auf Seite 23 zu entnehmen.

Einführung

Zum Vergleich erst einmal die Arbeitsweise des Link State Routing (LSR):



Das Optimized Link State Routing (OLSR) ist eine verbesserte Version des Link State Routings. Diese Bilder dienen nur zur kurzen Erläuterung der Arbeitsweise des LSR und verdeutlichen den Vergleich zum OLSR.



Die Arbeitsweise des OLSR ist folgende:

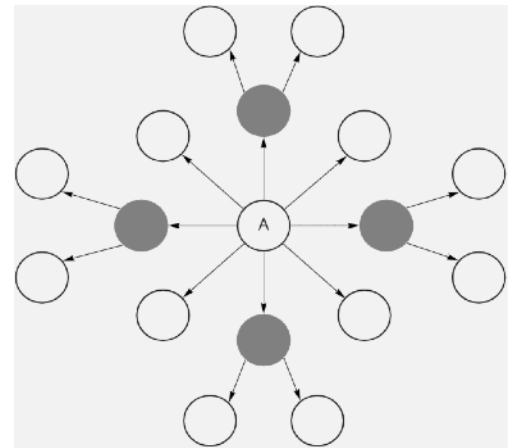
Zu Beginn kennen sich die Netzknoten einer Topologie gegenseitig nicht. Ihnen ist nicht bekannt, an welchen Stellen im Netzwerk sich die anderen Knoten befinden. Daher sendet jeder Knoten erst einmal sogenannte „Hello-Messages“, an alle anderen Knoten, die sich in Reichweite befinden, um seine direkten Nachbarn zu ermitteln. Hello-Messages werden zum Link Sensing, zur Nachbarentdeckung und zur Mitteilung der Wahl der Multipoint-Relays (s.u.) genutzt und werden periodisch an jeden Knoten verschickt (vgl. Campista & Rubinstein, 2014). Sie werden nicht weitergeleitet (vgl. Wodczak, 2018).

Jeder Knoten findet dadurch seine direkten Nachbarn (auch 1-hop-Nachbarn genannt, da sie einen Link entfernt sind) und da jede Hello-Message auch eine Liste der bereits bekannten 1-hop-Nachbarn enthält, auch seine 2-hop-Nachbarn (Nachbarn, die zwei Links entfernt sind) (vgl. HowTo, 2015).

Aus seinen 1-hop-Nachbarn wählt jeder Knoten einen Multipoint-Relay (MPR) aus, dafür muss eine bidirektionale Verbindung bestehen.

Die erwähnten MPRs sind besondere Knoten, nur sie dürfen Broadcast-Nachrichten weiterleiten. Dies verringert die Menge an gesendeten Nachrichten im Gegensatz zum Link State Routing signifikant, da dort jeder Knoten alle Nachrichten weiterleitet („Flooding“ von Nachrichten), dies führt zu hohen Redundanzen und somit zum unnötigen Verbrauch von Strom und Bandbreite. Mit den MPRs erhält kein Knoten eine Nachricht doppelt.

Die MPRs werden so gewählt, dass jeder Knoten durch sie seine 2-hop-Nachbarn erreichen kann. Insgesamt ist nun also jeder Knoten in der Lage, jeden Knoten zu erreichen (2-hop-Nachbarn werden über MPRs erreicht, im Bild grau hinterlegt).



Beispiel der Effizienz des MPR Mechanismus

Topology Control (TC) Messages werden dazu genutzt, gewonnene Informationen über die Topologie und mögliche Verbindungen im ganzen Netz zu verteilen.

Die MPRs teilen den anderen Knoten durch TC-Messages mit, welcher der Knoten sie als MPR gewählt haben.

In den TC-Messages werden auch Link State Advertisements mitgesendet. Link State Advertisements sind Update-Pakete mit neuen Informationen, die von dem Empfänger-Knoten dann verarbeitet werden (vgl. Campista & Rubinstein, 2014). Nur MPRs dürfen TC-Messages senden, so werden weiterhin Redundanzen vermieden.

Mit den gewonnen Topologie-Informationen ermittelt jeder Knoten eine optimale Route zu jedem anderen bekannten Knoten (im Bezug auf die „hops“, also wie viele Sprünge von Knoten zu Knoten gemacht werden müssen). Diese Routen werden dann in Routing Tabellen gespeichert (vgl. Campista & Rubinstein, 2014).

Knoten

Das Optimized Link State Routing Protokoll (OLSR) ist auf Mobile Ad-Hoc Netzwerke ausgerichtet (vgl. Wodczak, 2018). Das bedeutet, dass das Protokoll sehr gut an die Gegebenheiten eines sehr dynamischen Netzwerkes angepasst ist, dass eine oft wechselnde Netzwerktopologie besitzt. Dabei ist es oft schwierig, die Menge an Mehraufwand für die Kontrolle des Protokolls auf einem vernünftigen Niveau zu halten (vgl. Wodczak, 2018). Sollen nämlich die Änderungen des Netzes zeitnah und korrekt in den Routing-Informationen vorliegen, sind regelmäßig sehr viele Nachrichten nötig.

Dürfte jeder Knoten erhaltene Nachrichten an seine Nachbarn weiterleiten, so wären die Verbindungen oft überlastet und die Nachrichten würden den einzelnen Knoten mehrfach vorliegen. Um die Auslastung mit Nachrichten zu verringern, gibt es beim OLSR sogenannte Multi-Point Relays.

Jeder Knoten besitzt Verbindungen zu Nachbarknoten. Ist dabei nur eine Verbindung zwischen den Knoten spricht man von 1-hop-Nachbarn. Für die Multipoint-Relay Auswahl sind neben diesen auch noch die sogenannten 2-hop-Nachbarn wichtig. Diese sind über zwei Verbindungen vom Quellknoten aus erreichbar und sind sozusagen die 1-hop-Nachbarn der 1-hop-Nachbarn des Quellknotens.

Damit die einzelnen Knoten darüber Bescheid wissen, wer die 1- und 2-hop-Nachbarn sind, sendet jeder in periodischen Abständen Hello-Messages an seine Nachbarn. Dabei ist das periodische Senden sehr wichtig, da das Netz und dessen Verbindungen ständig von Änderungen betroffen sein kann, es kann zum Beispiel eine Verbindung oder ein Knoten ausfallen.. Als vordefiniertes Intervall ist hierbei ein Wert von 2 Sekunden festgelegt, es kann jedoch im Bereich von 62,6 Millisekunden bis 2,28 Stunden liegen (vgl. Wodczak, 2018).

Diese Information ist sehr wichtig für das gesamte Netz, da sonst versendete Nachrichten im schlimmsten Fall nicht ankommen und einfach verloren gehen können. Je nach Anforderungen und Gegebenheiten kann man zwischen drei Hauptweiterleitungsklassen unterscheiden.

Die Erste ist die Proaktive Klasse, in der jeder Knoten in regelmäßigen Abständen eine Erkennung der Topologie durchführt, um seine Routing Tabellen auf dem aktuellen Stand zu halten. So ein Ansatz ist meist jedoch sehr kostspielig, wenn es um die Menge an Kontrollmehraufwand geht.

Als zweites gibt es die Reaktive Klasse. Hierbei wird eine Erkennung der Topologie immer nur dann vollzogen, wenn Routing-Tabellen auf den neusten Stand gebracht werden sollen.

Die letzte Klasse ist eine Mischung aus den ersten Beiden. Können hierbei die Topologie-Änderungen als unbedeutend betrachtet werden, findet die reaktive Klasse Anwendung. Sollte das Gegenteil der Fall sein, so wird die Methode gewechselt und es wird auf eine proaktive Klasse umgestellt (vgl. Wodczak, 2018).

Jede Nachricht beinhaltet auch ein Feld namens „VTime“ für Validity Time. Diese gibt die Zeit der Gültigkeit der Nachricht an. Diese Zeit kann mithilfe einer Formel berechnet werden. Ebenfalls ist eine sogenannte „Time To Live“ (TTL) enthalten, welche die maximale Anzahl an Weiterleitungen beinhaltet. Sie wird vor jeder Weiterleitung um den Wert 1 verringert. Sollte hierbei der Wert 1 oder 0 erreicht werden, wird die Nachricht nichtmehr weitergeleitet.

Ebenfalls ist in einer Nachricht ein sogenanntes „Willingness-Feld“ vorhanden. Hierdurch wird angegeben, ob ein bestimmter Knoten eine Nachricht weiterleiten wird oder nicht. Dabei gibt es verschiedene Stufen der Bereitwilligkeit. Namentlich heißen diese WILL_NEVER (0), WILL_LOW(1), WILL_DEFAULT (3), WILL_HIGH (6) und WILL_ALWAYS (7). Ist die Bereitwilligkeit bei 0, so wird dieser Knoten nie als Multipoint-Relay ausgewählt, ist sie bei 7 wird er immer ausgewählt (vgl. Wodczak, 2018).

Multipoint-Relay und deren Auswahl

Der große Vorteil an der Nutzung von Multipoint-Relays (MPRs) liegt darin, dass die Datenverteilung und Nachrichtenweiterleitung sehr gut kontrolliert werden kann. Beim OLSR-Protokoll kann deshalb nicht jeder Knoten eine erhaltene Nachricht weiterschicken, sondern nur die ausgewählten MPRs.

Möchte nun also ein Knoten eine Nachricht versenden, so analysiert er zuerst seine Nachbarknoten. Dabei schaut er nicht nur auf seine 1-hop-Nachbarn, sondern auch auf seine 2-Hop-Nachbarn. Auch wenn keine direkte Verbindung zu den zwei Verbindungen entfernten Knoten bestehen, so ist es doch wichtig, dass der Quellknoten der Nachricht darüber Bescheid weiß. Er wählt jetzt nämlich auf Basis dieser Informationen Multipoint-Relays aus, die für das Verbreiten seiner Nachricht zuständig sind. Diese sind immer direkte Nachbarn des Quellknotens und können zusammen alle 2-Hop-Nachbarn abdecken.

Wichtig dabei ist ebenfalls, dass die Verbindungen zwischen den Knoten symmetrisch sind. Das bedeutet, dass darüber Informationen sowohl empfangen als auch gesendet werden können.

Auch wenn alle direkt verbundenen Knoten die Nachricht empfangen und verarbeiten können, so dürfen nur die ausgewählten Multipoint-Relays die Nachricht weitersenden und damit das gesamte Netz abdecken. Dadurch soll die Menge an unnötig weitergeleiteten Nachrichten minimiert werden, da die Informationen von einem Knoten nicht mehrfach von dessen Nachbarn weitergeleitet werden.

Informationsspeicherung

Da das OLSR-Protokoll verschiedene Informationen mittels Link-Erkennung, Nachbarerkennung, MPR-Auswahl und Topologie-Erkennung erfasst, muss es bestimmte Informationsspeicher unterhalten, welche in diesem Abschnitt genauer charakterisiert werden (vgl. Wodczak, 2018).

Die Versionen des OLSR stützen sich auf unterschiedliche Konzepte der Informationsspeicherung. Hierbei werden unterschiedliche Repositories genutzt. Anbei werden die vier Hauptkategorien der in einem Node enthaltenen Speicherbereiche nach Clausen und Jacquet (2003) dargestellt:

- Multiple Interface Association Information Base
- Local Link Information Base
- Neighborhood Information Base
- Topology Information Base

Es gilt festzuhalten, dass dieses das für dieses Projekt relevante Modell darstellt und die beiden Autoren im Laufe der Zeit auch weitere Speichermodelle entwickelt haben, welche im Aufgabenteil c) kritisch bewertet werden.

Die vier bereits vorgestellten Kategorien werden wiederum durch die Darstellung des Schnittstellenzuordnungstupels, des Verbindungstupels, des Nachbartupels, des 2-hop-Nachbartupels und des MPR-Selektorsatzes angesprochen, um mit dem Topologietupel abzuschließen.

Die Multiple Interface Association Information Base enthält den sogenannten Schnittstellenassoziationssatz. Dieser Satz besteht aus den sogenannten Schnittstellenzuordnungstupeln, die für jeder Zielknoten im Netzwerk gespeichert wird. Ein solches Tupel besteht aus der `I_iface_addr`, welche die Adresse der Schnittstelle eines bestimmten Zielknotens angibt, der `I_main_addr`, welche die Hauptadresse eines bestimmten Zielknotens angibt und der Ablaufzeit, zu der dieses Tupel verworfen wird mit der `I_time` (Wodczak, 2018).

Als nächstes kommt die Local Link Information Base, die sich auf den Betrieb der Link-Erkennung bezieht und den zugehörigen Link-Satz enthält, ins Spiel. Diese Menge besteht aus den sogenannten Link-Tupeln, welche auch wieder Zeit und Adress-Informationen

beinhalten. Hierbei gibt `L_local_iface_addr`, die Adresse der Schnittstelle eines bestimmten lokalen Netzwerkknotens an und die `L_neighbor_iface_addr`, die Adresse der Schnittstelle eines bestimmten Nachbarknotens an.

Weiterhin beinhaltet die Local Link Information Base auch die `L_SYM_time`, welche die Zeit angibt, bis zu der ein bestimmter Link als symmetrisch betrachtet werden soll und die `L_ASYM_time`, welche die Zeit angibt, bis zu der eine bestimmte Schnittstelle eines bestimmten Nachbarknotens als gehört gelten soll. Die Ablaufzeit, bis zu welcher das Tupel verworfen werden soll, wird hierbei wieder durch die `I_time` angegeben.

Basierend auf diesen aufgezeigten Informationen können die Nachbarschnittstellen unter Verwendung von Hello-Messages deklariert werden. Sollten sowohl die `L_SYM_time` als auch die `L_ASYM_time` ablaufen, wird die Verbindung als verloren betrachtet. Hier kommt dann die Neighbor Information Base ins Spiel. Diese bezieht sich auf die Operation der Nachbarerkennung und enthält die ursprüngliche Nachbarmenge, die 2-Hop-Nachbarmenge sowie die zugehörige MPR-Menge und die MPR-Selektormenge. Diese Mengen werden insbesondere durch die Hauptadressen der als MPRs ausgewählten Nachbarknoten zusammen mit den sogenannten Nachbartupeln, Zweisprungtupeln und MPR-Auswahltupeln gebildet.

Die Nachbarmenge wird in dem sogenannten Neighbour Tupel abgebildet. Dieses beinhaltet die `N_neighbor_main_addr`, welche die Hauptadresse eines bestimmten Nachbarknotens angibt und den `N_status`, welcher angibt ob ein bestimmter Nachbarknoten über eine symmetrische oder nicht symmetrische Verbindung verbunden angesehen werden soll. Zusätzlich dazu gibt `N_willingness` einen ganzzahligen Wert im Bereich von 0 bis 7 an, der beschreibt, ob ein bestimmter Nachbarknoten bereit sein soll, Verkehr von anderen Netzwerkknoten zu tragen und weiterzuleiten (s.o.).


Um die 2-hop-Nachbarn zu speichern, beinhaltet die Neighbour Information Base auch ein 2-hop-Nachbartupel, welcher die Hauptadresse eines bestimmten Nachbarknotens in der `N_neighbor_main_addr` beinhaltet und die `N_2hop_addr` gibt die Hauptadresse eines gegebenen 2-Hop-Nachbarknotens an, der mit diesem durch `N_neighbor_main_addr` bezeichneten über eine symmetrische Verbindung verbunden ist. Weiterhin beinhaltet auch dieses Tupel mit `N_Zeit` die Zeit, in der das Tupel verworfen werden soll.

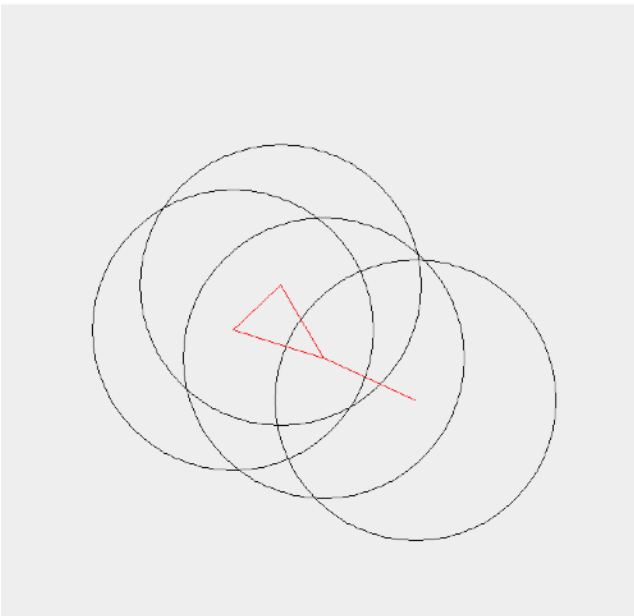
Ein weiteres wichtiges Tupel stellt das MPR-Selector-Tupel dar. Dieses beinhaltet die Hauptadresse des Nachbarknotens, der diesen bestimmten Netzwerkknoten als MPR ausgewählt hat in `MS_main_addr` und die Ablaufzeit, zu der dieses Tupel verworfen werden soll in `MS_time`.

Schließlich gibt es die Topology Information Base, die sich auf die Operation der Topologieerkennung bezieht und den Topologiesatz enthält. Diese Menge besteht aus den sogenannten Topologie-Tupeln, die sich aus `T_dest_addr` und `T_last_addr` zusammensetzt. Die `T_dest_addr` ist die Hauptadresse des Nachbarknotens, die in einem Hop von dem mit `T_last_addr` bezeichneten Netzwerkknoten erreicht werden kann. Die `T_last_addr` spezifiziert hierbei den Netzwerkknoten, der typischerweise der MPR des durch `T_dest_addr` bezeichneten Zielknotens ist. Zusätzlich gibt hier `T_seq` die Sequenznummer und `T_Zeit` die Ablaufzeit, zu der dieses Tupel verworfen werden soll, an.

Testfälle

Jede Node wird durch einen kleinen Punkt visualisiert, dessen Sendereichweite als Kreis um diesen Punkt herum gezeigt wird. Die Größe des Kreises (der Radius) ist bei allen Nodes gleich. Zwei Nodes können eine Verbindung zueinander aufbauen – also sich gegenseitig Nachrichten senden – wenn der Punkt im Kreismittelpunkt (also die Node) im Kreis der anderen Node liegt. Sind zwei Knoten in Reichweite voneinander, so werden diese durch eine rote Linie miteinander verbunden. Im Folgenden werden einige Testläufe durchgeführt und beschrieben, von nicht beweglichen Knoten mit variierenden räumlichen Verteilungen, Sendereichweiten und Knotenanzahlen.


 Mobiles Ad-Hoc Netz

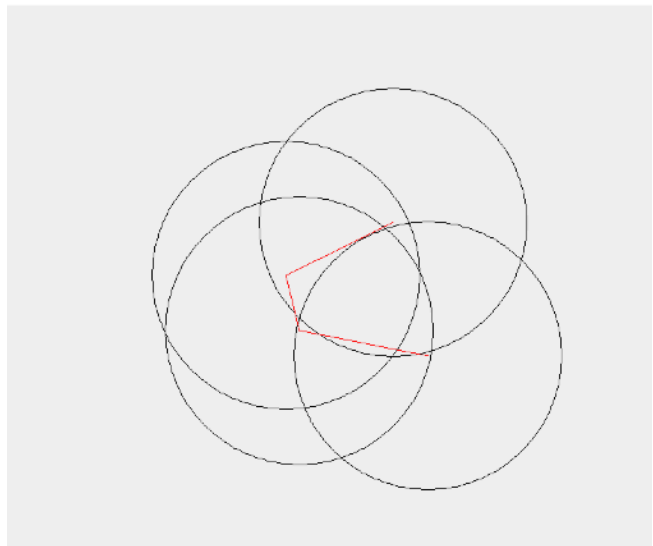


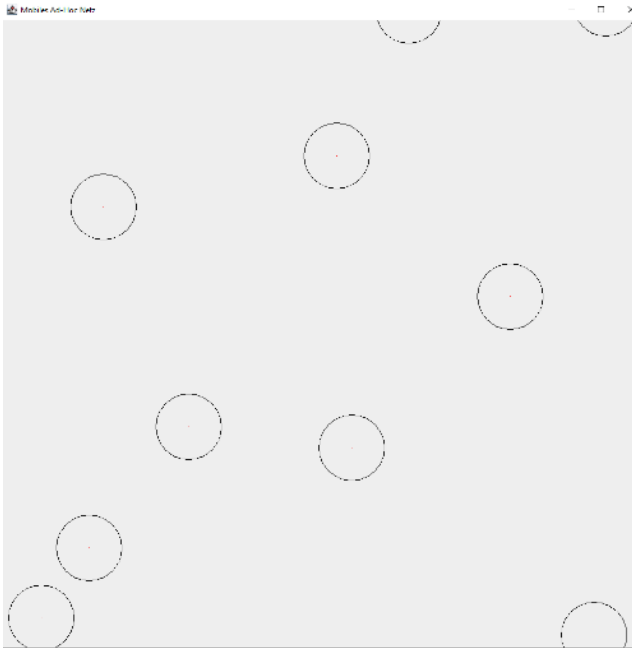
Wählt man eine geringe Anzahl an Nodes (hier etwa 4) und generiert diese relativ nah beieinander, so hat man eine geringe Chance, dass ein kleines, aber vollständig verbundenes Netz entsteht. Die Chance auf ein vollständig verbundenes Netz wird außerdem dadurch gestigert, dass in der Abbildung hier die Sendereichweite der einzelnen Knoten verdoppelt wurde. Die entstandenen möglichen Routen werden mittels roter Verbindungslinien dargestellt.

So entsteht ein einfaches, verbundenes kleines Netz, in dem jeder Knoten von jedem anderen Knoten (über Weiterleitungen) erreicht werden kann.

Die nächste Abbildung zeigt ein weiteres Beispiel für ein solches Netz:

 Mobiles Ad-Hoc Netz





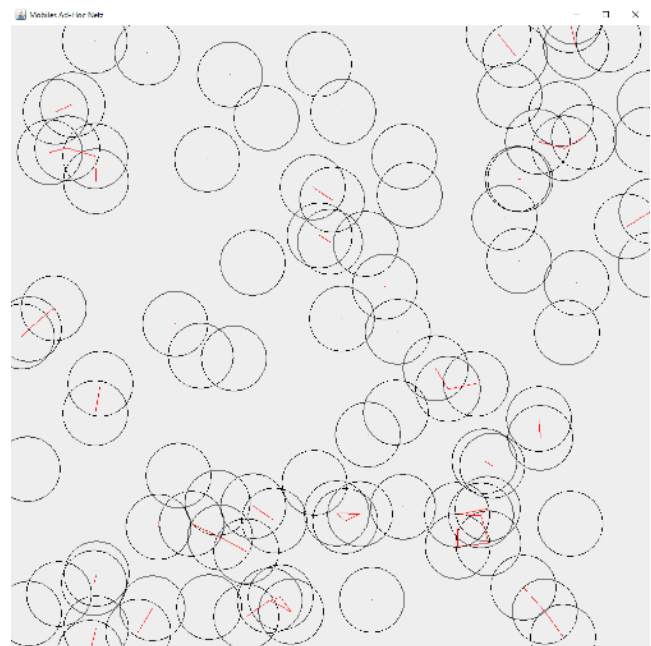
Lässt man sich ein Netz mit normaler Sendereichweite, zufälliger räumlicher Entfernung und geringer Knotenzahl generieren, so ist die Wahrscheinlichkeit für ein vollkommen verbundenes Netz sehr gering. In dieser Abbildung wurde ein Netz aus 10 Knoten mit 10m Sendereichweite auf einem 100m x 100m großen Spielfeld mit zufälliger Räumlicher Verteilung generiert. Da dieses Spielfeld für die geringe Anzahl einzelner Knoten jedoch sehr groß ist, kommt ein Netz mit maximaler Anzahl an Partitionierungen zustande: Keine Station kann eine andere Station erreichen!

Damit ist keine Kommunikation möglich.

Erhöht man die Knotenzahl deutlich auf 100 Nodes, lässt die übrigen Bedingungen aber gleich, so entsteht die Verteilung rechts. Hier gibt es immer noch eine große Zahl an Knoten, die keinen anderen Knoten innerhalb ihrer eigenen Sendereichweite haben, es entstehen aber auch einige kleinere Teilnetze. Häufig sind 2-4 Knoten miteinander verbunden und bilden damit kleinere Netzpartitionen.

Auch dieses Netz ist jedoch noch sehr weit von einem guten oder gar optimalen Netz entfernt. Hier kann eine einzelne Node maximal 4 andere erreichen, viele haben auch hier noch keinen direkten Nachbarn und können damit überhaupt nicht kommunizieren.

Dieses Problem löst sich erst, wenn man die Knotenzahl weiter drastisch erhöht, oder die räumliche Verteilung eingrenzt.

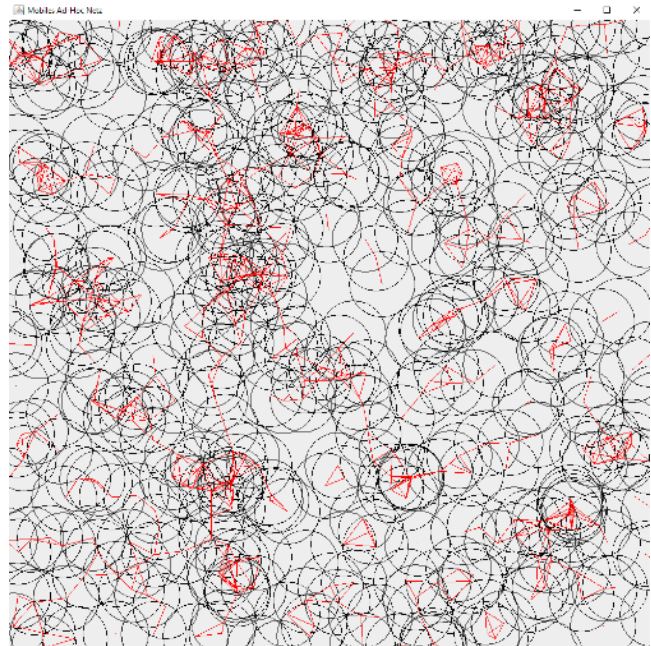




Auf dieser Abbildung ist ein Netz mit 20 Knoten und normaler Sendereichweite zu sehen, bei dem die räumliche Verteilung der einzelnen Knoten stark eingeschränkt ist. Dadurch entsteht nun ein relativ gutes Netz, es ist nicht vollständig verbunden, aber es gibt keine Station mehr ohne direkten Nachbarn, jeder kann also mit mindestens einem anderen kommunizieren. In diesem Szenario speziell haben sich jetzt 3 Subnetze gebildet, das kleinste mit 3 Nodes.

Belässt man die räumliche Verteilung zufällig, so muss man die Anzahl der einzelnen Knoten weiter deutlich erhöhen, um ein ähnliches Ergebnis erzielen zu können.

Hier wurde nun ein Netz aus 500 gleichen Nodes mit normaler Sendereichweite erstellt. Man erkennt die deutlich dichtere Vermaschung in einzelnen Bereichen des Netzes, es haben sich einige große Teilnetze, aber auch vereinzelt sehr kleine Teilnetze gebildet, die untereinander nicht verbunden sind. Gleichzeitig gibt es (sehr vereinzelt) immernoch einzelne Knoten, die außer Reichweite jedes einzelnen andern Knotens sind und für die daher keine Kommunikation möglich ist.



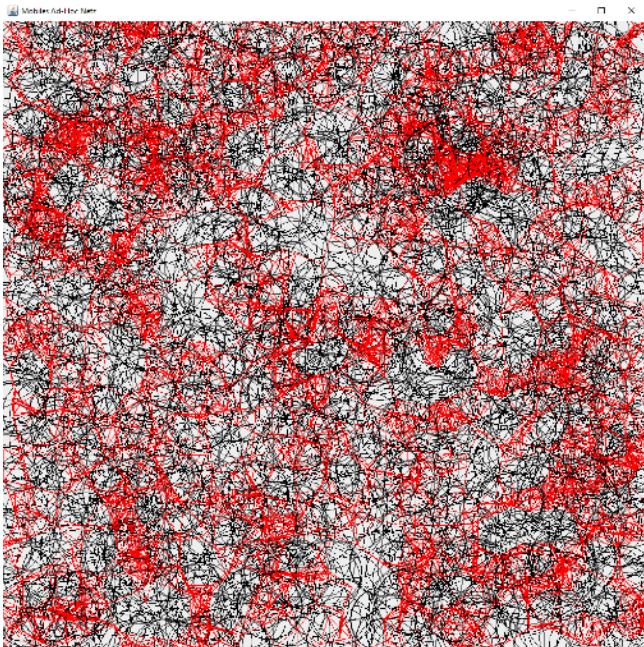
Das hier gezeigte Netz ist also immer noch nicht vollständig verbunden, es ähnelt allerdings bereits sehr dem vorherigen Ergebnis, mit weit weniger Nodes auf sehr viel engerem Raum.

Um die Frage zu klären, ab welcher Knotenzahl ein Netz mit ca. 90% Wahrscheinlichkeit vollständig verbunden ist, haben wir in einer neuen Klasse (TestCases.java) ein Programm geschrieben, das diese Berechnung für uns durchführt. Da die Knotenzahl für diese

Rechnung sehr hoch sein wird (wir haben mit 500 Nodes noch immer kein vollständig verbundenes Netz erhalten) werden Nodes in dieser Berechnung „gemockt“. Anstatt also jede Node durch einen Thread zu simulieren, der einen eigenen Sender und einen eigenen Receiver besitzt, ist ein Knoten in diesem Kontext nur ein Objekt, das seine eigene ID, seine eigenen Koordinaten und seine direkten Nachbarn kennt. Dadurch wird weit weniger Rechenleistung benötigt, ein normaler Prozessor mit 4 oder 8 Kernen käme sonst beim Simulieren von 1000 oder mehr Threads schnell an seine Grenzen.

Um nun auf die benötigte Anzahl „Fake-Knoten“ zu kommen werden bei Ablauf des Programms mehrere Knotenzahlen jeweils 100 mal durchprobiert, entsteht bei einem dieser Durchläufe ein vollständig vermaschtes Netz, so wird ein Zähler um 1 erhöht. Am Ende hat man pro Knotenzahl eine Anzahl an Durchläufen mit vollständig verbundenen Netzen – Diese entspricht der jeweiligen Wahrscheinlichkeit für diese Knotenzahl.

In jedem Durchlauf werden nun entsprechend viele „MockNodes“ mit zufälligen Koordinaten generiert und gespeichert. Anschließend erhält jede dieser „MockNodes“ die vollständige Liste aller Knoten, um sich seine direkten Nachbarn herauszusuchen und zu speichern. Ist dies für alle Nodes geschehen, sucht man sich eine heraus, fragt sie nach ihren direkten Nachbarn und speichert sich diese in eine Map. Nun geht man alle dieser Nachbarn durch und fragt diese wiederum nach ihren Nachbarn, bis keine weiteren mehr hinzukommen. Hat man am Ende genauso viele Einträge in seiner Map wie es Knoten insgesamt gibt, ist das Netz vollständig verbunden. Ansonsten ist es das nicht.



Für eine Wahrscheinlichkeit von ca. 90 Prozent für ein vollständig verbundenes Netz kommt man auf ungefähr 1630 einzelne Knoten. Ein Netz dieser Größe sähe wie folgt aus:

Diese Berechnungen gelten für die Größen- und Verhältnismäßigkeiten der TestCases.java.

Vorteile des OLSR und der MANets

Mobile Ad-hoc Netze (MANETs) sind generell sehr leistungsfähig und besitzen eine gute Lastverteilung (vgl. Ad-hoc-Netz, 2022). Diese Vorteile kann man sehr gut beim OLSR-Protokoll sehen, bei dem nur eine begrenzte Anzahl an Knoten, die TC (Topology Control) Nachricht verbreiten dürfen. Die anderen dürfen die Nachrichten nur empfangen und verarbeiten, jedoch nicht an die jeweiligen Nachbarn weiterschicken.

Dadurch wird das Überschwemmen des Netzes mit Nachrichten reduziert (vgl. Campista & Rubinstein, 2014). Das Verwenden von MPRs verringert dabei die Anzahl und Größe der TC Nachrichten und ermöglicht es nur mithilfe deren jeweiligen MPR-Selektoren andere Knoten zu erreichen (vgl. Campista & Rubinstein, 2014). Deshalb ist das OLSR-Protokoll besonders gut für dicht besiedelte Netzwerke geeignet (vgl. Campista & Rubinstein, 2014).

Ebenfalls ist es sehr gut für Mobile Ad-hoc Netze geeignet, bei denen sehr dynamische Schwankungen in Bezug auf ständig wechselnde Netzwerk Topologien auftreten können. Mobile Ad-hoc Netze können beispielsweise bei Ausfall eines Knoten die Datenkommunikation umleiten, sodass diese weiterhin möglich ist (vgl. Ad-hoc-Netz, 2022). Mithilfe des OLSR-Protokolls soll hierbei der Kontrollaufwand auf einem vernünftigen Niveau gehalten werden, wobei jedoch immer noch jede Änderung bemerkt und akkurate Routing Informationen bereitgestellt werden (Wodczak, 2018).

Je nach Beschaffenheit des Netzwerkes können drei verschiedenen Routing-Klassen implementiert werden, was zu einer höheren Flexibilität bei Anwendungsfällen führt. Dies kann beispielsweise dadurch entschieden werden, ob sich die Netzwerktopologie oft oder selten ändert. Die erste Klasse ist dabei die proaktive Routing Klasse, bei der jeder Knoten in regulären Abständen eine Topologie Erkennung durchführt und ist somit gut dafür geeignet, wenn alle Routing Tabellen der Knoten auf dem neusten Stand bleiben sollen.

Die zweite Routing-Klasse ist die reaktive Klasse, die nur eine Netzwerk Erkennung durchführt, wenn die Routing Tabellen aktualisiert werden müssen. Dies führt zu einem niedrigen Kontrollaufwand. Die dritte Klasse ist eine Mischung aus der proaktiven und reaktiven Klasse, wobei bei einzelnen Netzwerkregionen je nach Aktivität das eine oder das andere implementiert werden kann (vgl. Wodczak, 2018).

Ein weiterer Vorteil von Mobilen Ad-hoc Netzen ist, dass keine zentrale Verwaltung nötig ist (vgl. Ad- hoc-Netz, 2022). Die Infrastruktur des Netzwerkes wird durch das OLSR-Protokoll selbst erstellt und benötigt keine traditionellen fest aufgestellten Router.

Dadurch benötigt es also keine zentrale Einheit und ist sehr flexibel (Azzuhri, Suhaimi und K. Wong, 2022). Somit wäre es also auch beispielsweise in Krisengebieten sehr gut einsetzbar, wo entweder keine Infrastruktur mehr vorhanden ist oder noch keine vorhanden war.

Mithilfe der Angabe eines Willingness-Wertes bei jedem einzelnen Knoten, kann der Energieverbrauch optimiert und auf das wesentliche verringert werden (Azzuhri, Suhaimi und K. Wong, 2022). Je nachdem, wie abgelegen oder schwer erreichbar ein Knoten ist, kann dieser seine Willingness bis zum Wert 0 angeben, um so wenig Strom zu verbrauchen, wie möglich. Das bedeutet also, sollte wie oben angesprochen beispielsweise keine oder kaum Infrastruktur vorhanden sein, ist es sehr von Vorteil, die Wartung der einzelnen Knoten so gering wie nötig halten zu können. Je niedriger also dieser Willingness Wert, desto weniger bis gar nicht wird er als Multi Point Relay ausgewählt, was dazu führt, dass kein zusätzlicher Strom für das Weiterleiten von Nachrichten verbraucht wird.

Nachteile des OLSR und der MANets

Da keine zentrale Verwaltung oder Router notwendig sind, muss jeder einzelne Knoten seine Routing Tabelle abspeichern. Ebenfalls arbeiten Knoten oft als Router, was dazu führt, dass sie oft aktiv sein müssen und somit möglichst immer eingeschaltet bleiben müssen. Dies ist also zum Beispiel der Fall, wenn eine Nachricht verschickt wird und die gespeicherten MPRs des Quellknoten der Nachricht diese weiterschicken müssen. Eine hohe Erreichbarkeit führt jedoch zu einem höheren Stromverbrauch der einzelnen Knoten (vgl. Ad-hoc-Netz, 2022).

Wenn das Netzwerk spärlich mit Knoten besiedelt ist, werden die meisten Nachbarknoten zum jeweiligen MPR des anderen (Campista & Rubinstein, 2014). Damit wäre der eigentliche Vorteil des OLSR-Protokolls nicht vorhanden, da die einzelnen Knoten zum Großteil oder schlimmstenfalls alle als MPR ihres jeweiligen Nachbarn eingetragen werden müssen, um das gesamte Netz mit einer Nachricht erreichen zu können.

Eine geringe Anzahl an TC-Nachrichten kann zu Problemen führen, besonders wenn die Verbindungsqualität schlecht ist. Da beim OLSR-Protokoll wenig redundante Nachrichten vorhanden sind, kann deren möglicher Verlust zu falsch aktualisierten Routing Tabellen führen. Dies kann wiederum dazu führen, dass Topologie Karten der einzelnen Knoten nicht synchronisiert sind, was zu Schleifen und Staus und damit zu immer wiederholenden Nachrichtenverlusten führen kann (Campista & Rubinstein, 2014).

Es kann bei nicht richtig oder gar nicht optimierten proaktiven Routing-Klassen zu hohen Kontrollaufwendungen kommen (vgl. Wodczak, 2018). Bei proaktiven Routing-Klassen werden in regulären Abständen Topologie-Erkennungen durch jeden Knoten des Netzwerkes durchgeführt, was sehr aufwandsintensiv sein kann (vgl. Wodczak, 2018). Wird hierbei beispielsweise ein sehr kurzes Intervall gewählt, so können die Routing-tabellen zwar immer auf dem neusten Stand sein, aber die Netzwerkauslastung ist aufgrund der vielen Nachrichten sehr hoch. Dies kann dazu führen, dass Nachrichten verloren gehen und ein hoher Energieverbrauch vorliegt. Sollten Knoten mit Batterien oder Akkus betrieben werden, müssen diese dadurch häufiger geladen oder ausgetauscht werden.

Lösungsansätze

Das OLSR-Protokoll leider die eben genannte Nachteile hat, befinden sich viele Protokolle in der Entwicklung oder wurden entwickelt, um auf die Mängel des Protokolls reagieren.

Zu diesen Lösungsansätzen gehören:

- BATMAN

Das OLSR-Protokoll hat starke Probleme mit der Synchronisation der TC-Nachrichten und damit auch den Informationen der Routen, die in den Knoten enthalten sind.

Dies ist der Fall, wenn sich der aktuelle Zustand der Topologie von den den Knoten bekannten Routen unterscheidet (z.B. durch Änderungen wie ein neu hinzugekommener Knoten). Dann muss gewartet werden, bis sich die Topologie wieder aktualisiert hat und Routen wieder korrekt gespeichert sind (vgl. frwiki, 2022).

Dafür wurde die Entwicklung des BATMAN-Protokolls begonnen. Dessen zentrale Idee „besteht darin, Informationen über die beste Verbindungen zwischen allen BATMAN-Knoten im gesamten Netzwerk auszutauschen“ (BATMAN, 2022). Dadurch ist es nicht mehr nötig, alle Knoten bei Änderungen des Systems zu informieren.

- OLSRv2

Seit 2005 wird OLSRv2 von IETF entwickelt, um bekannte Mängel des OLSR zu lösen. Zum aktuellen Zeitpunkt befindet sich eine Version (RFC) in der Standardisierungsphase und wird womöglich bald veröffentlicht (vgl. frwiki, 2022). Der Hauptunterschied zum OLSR ist die aussagekräftigere Link-Metrik statt dem reinen Hop Count.

- CE-OLSR

Bei wird die Position von Knoten integriert. Das funktioniert, indem in jeder Hello-Message ein Feld hinzugefügt wird, das die eigene Position und die der Nachbarn in GPS-Koordinaten enthält. Bei den TC-Messages wird ein Feld hinzugefügt, dass die Koordinaten der anderen MPR-Selektoren enthält (vgl. frwiki, 2022). Dies bringt folgende Vorteile:

- Die Kontrollnachrichten sind in Bezug auf Paketverlust deutlich robuster.
- Änderungen in der Topologie werden schneller erkannt, auch können Nachbarn präziser verfolgt werden.
- Es ist die Möglichkeit gegeben, alte und aktuelle Topologieinformationen voneinander zu unterscheiden.
- Dies ermöglicht eine bessere Leistung in Bezug auf den Durchsatz als auch die Antwortzeit, außerdem erhalten bekannte Routen eine längere Gültigkeit.

Alternativen

Abgesehen von Ad-hoc Netzwerken gibt es noch weitere Alternativen, welche in Katastrophengebieten eingesetzt werden können, um Funknetzwerke aufzubauen um den Datenverkehr zu ermöglichen. In diesem Abschnitt werden einige dieser Alternativen aufgezeigt.

- Das Serval Mesh Projekt

Ein Team von Expertenforschern an der Flinders University hat eng mit der Notfalleinsatzeinheit für Informationstechnologie und Telekommunikation des Neuseeländischen Roten Kreuzes eine alternative Kommunikationslösung zu Ad-hoc Netzwerken entwickelt, die speziell auf die Pazifikregion zugeschnitten ist.

Die Lösung mit dem Namen Serval Mesh ist eine Software-Suite, mit der handelsübliche Android-Telefone infrastrukturfreie Peer-to-Peer-Sprach-, Text- und Datendienste ausführen können. Im Wesentlichen ermöglicht diese Lösung, Menschen über drahtlose Mesh-Netzwerke zu kommunizieren zu lassen, anstatt über herkömmliche Telekommunikationsnetzwerke. Anders als Ad-hoc Netzwerke ermöglicht diese Lösung mobilfunkähnliche Kommunikation ohne Mobilfunksignal oder Internet.

Die Technologie ermöglicht Live-Sprachanrufe, wann immer das Mesh in der Lage ist, eine Route zwischen den Teilnehmern zu finden. Textnachrichten und andere Daten können mithilfe eines Speicher- und Weiterleitungssystems namens „Rhizome“ kommuniziert werden, welches eine Kommunikation über unbegrenzte Entfernungen und ohne eine stabile Live-Mesh-Verbindung zwischen allen Teilnehmern ermöglicht (Paul Gardner-Stephen, 2013).

Technisch umfasst das Serval-Projekt eine kollaborative Mapping-Anwendung, die Katastrophenhilfe und Wiederherstellungsbemühungen unterstützen soll. Zusätzlich wurde ein „Mesh-Extender“ entwickelt, der ein Serval-Mesh mit kurzer Reichweite über WiFi aufbaut und es mit anderen weiter entfernten Meshs verbindet, indem es sich mit anderen Mesh-Extendern über Packet Radio verbindet, das im ISM-915-MHz-Band operiert (AARON MAK, 2021).

Serval Mesh basiert wie bereits beschrieben auf eine Android-Anwendung, welche derzeit über verschiedene Plattformen und Repositories vertrieben wird und auch direkt von der

Website des Projekts heruntergeladen werden kann. Die Anwendung kann über WLAN oder Bluetooth direkt von einem Gerät an andere in der Nähe weitergegeben werden.

Die Serval Mesh-Anwendung selbst besteht aus zwei Komponenten: einer Benutzeroberfläche namens Batphone und einer zentralen Netzwerk-, Verschlüsselungs- und Dateifreigabekomponente namens Serval DANN (Paul Gardner-Stephen, 2013).

Um die Effektivität zu verbessern, lässt sich die Software auch in optionale, kostengünstige Funkhardwareeinheiten im Taschenformat, sogenannte Serval Mesh Extender, integrieren, die es Menschen ermöglichen, über Funksignale zu kommunizieren (Flinders University, 2020).

Kritisch zu bewerten ist hierbei, dass die Gewährleistung des Serval Root-Zugriffs, dazu führt, dass Serval Mesh die Kontrolle über das WLAN des Telefons übernimmt. Nutzer müssen sich entsprechend von der Serval App abmelden, um zu Ihrer normalen WLAN-Verbindungen zurückzukehren. Serval Mesh kann das Telefon dazu auch in den Hotspot-Modus versetzen, wodurch Telefone in Nähe von Serval Mesh auf den Datentarif des Telefons zugreifen können. Dies kann entsprechend hohe Kosten mit sich bringen.

- Das Loon Projekt

Ein weitaus radikalerer Ansatz zur Erweiterung der Internetkonnektivität stellt das Loon Projekt dar. Loon entstand als Forschungsprojekt von Google mit dem Ziel, abgelegene Gegenden mit einem Internetzugang über Heliumballons in der Stratosphäre zu versorgen und die Widerstandsfähigkeit des Netzwerks im Katastrophenfall zu verbessern.

Anstatt zu versuchen, das Internet mit traditioneller bodengestützter Infrastruktur wie Glasfaserkabeln oder Mobilfunkmasten zu erweitern, stieg Loon mit einem Netzwerk aus Ballons in die Lüfte. Projekt-Loon-Ballons wurden mit einer Höhe von etwa 20 km doppelt so hoch wie Flugzeugflüge positioniert. Gelenkt wurden die Loon-Ballons, indem sie unter Verwendung von Winddaten von NOAA in eine Windschicht aufstiegen oder abstiegen, die in die gewünschte Fahrtrichtung wehten. Mit der Bewegung des Windes, konnten die Ballons dann so angeordnet werden, dass sie ein großes Kommunikationsnetzwerk bildeten (Katikala, 2014).

Ein von Google entwickelter Algorithmus sollte hierbei garantieren, dass der Ballon vollkommen autonom agieren kann und abhängig von der gewünschten Flugrichtung selbstständig auf- und absteigt. Gleichzeitig sollten die Ballons fähig sein, untereinander zu kommunizieren, um somit automatisch ein komplettes Kommunikationsnetzwerk aufbauen zu können (Gordon Ecco, 2015).

Wie bereits am sprachlichen Tempus dieses Abschnittes zu erkennen ist, konnte das Projekt Loon sich nicht durchsetzen.

Der jährliche Finanzierungsbedarf wurde zuletzt auf USD 100 Mio. geschätzt, was die Suche nach Investoren zum Scheitern brachte, sodass Loon im Januar 2021 bekannt gab, dass die Dienste beendet werden. Die Gründe hierfür waren rein ökonomisch, denn das System konnte sich in Notsituationen nach Katastrophen bewähren. Indien war 2017 das erste Land, das Loon landesweit einsetzte und auch in Kenia wurde ein Mobilfunknetz mit den solarbetriebenen Heliumballons eingerichtet.

Ein weiterer Punkt, der gegen Loon sprach, war das aufkommende Konkurrenzunternehmen Starlink.

- Satellitenkommunikation

Die Aktualität dieser Thematik ist wie in der Einführung dieser Arbeit bereits beschrieben, näher und aktueller denn je (Der Spiegel, 2022).

Das bereits beschriebene Konkurrenzunternehmen zum Loon Projekt namens Starlink ist ein vom US-Raumfahrtunternehmen SpaceX betriebenes Satellitennetzwerk, welches sich zum Ziel genommen hat, den Menschen in Zukunft weltweiten Internetzugang zu bieten und dies mithilfe von Satellitenkommunikation.

Unter Satellitenkommunikation wird die bidirektionale Telekommunikation über einen Satelliten zwischen zwei Bodenstationen verstanden. Diese mobilen Satellitenanlagen können von nahezu jedem Punkt der Erde einen schnellen Internetzugang ermöglichen (Evakuierungen & Hochschule Wildau, 2014).

Die Satelliten kommunizieren hierbei miteinander per Laser, welche sich entsprechend mit Lichtgeschwindigkeit fortbewegen, wobei das Licht im Vakuum pro Sekunde ca. 300.000 Kilometer zurücklegt. Die Daten werden per Funk zur Erde gesandt, wo sie mit sogenannten Phased-Array-Geräten empfangen werden (AC Boley, 2022).

Die Übertragungsbandbreite kann mehrere MBits/s betragen. Die Möglichkeiten und Grenzen der Kommunikation hängen wesentlich vom verwendeten Satellitenkommunikationsnetzwerk ab, wobei die Unterschiede vor allem die Kosten, die Datenraten für Upload und Download bis hin zur Begrenzung von Traffic, die Netzabdeckung und den benötigten Höhenwinkel, um eine unterbrechungsfreie Kommunikation sicherzustellen, betreffen (Evakuierungen & Hochschule Wildau, 2015). Die Übertragungsrate von Starlink soll zukünftig beispielsweise mit 10GBit pro Sekunde erfolgen. Das ist zum jetzigen Stand zweimal so schnell wie eine Glasfaserübertragung (Uwe. R, 2022).

Die bereits beschriebenen Empfangsgeräte stellen eine erste Problematik der Satellitenkommunikation dar, denn wer die Signale aus dem Weltall empfangen möchte, der braucht leistungsstarke Empfangsgeräte. Phased-Array-Geräte sehen wie kleine Bretter aus, haben keine beweglichen Teile und können in Millisekunden von einem Satelliten zum nächsten schalten, sobald dies notwendig wird. Diese Technik existiert bereits, ist aber noch relativ teuer. Derzeit kosten diese Empfänger noch ca. 200,- Dollar, doch die Weiterentwicklungen laufen in hohem Tempo und mit der Massenproduktion kann der Preis entsprechend skaliert werden (Uwe. R, 2022).

Weiterhin steht die Satellitenkommunikation auch wegen der Vermüllung des Weltraumes in der Kritik. Sobald ein Satellit kaputt geht, fliegt er als Metallgeschoss sinnlos im Erdbereich herum und gefährdet andere Satelliten und Raumfahrtobjekte. Für diese Problematik suchen Unternehmen, jedoch bereits nach entsprechenden Lösungen (Marton, 2022).

Zurzeit gibt es noch einige kritische Fragen rund um diese Form der Funknetzwerke, jedoch bewerten wir im Rahmen dieser Arbeit diese Technologie als aktuell beste Alternative um in Katastrophen Fällen eine schnelle Ad-hoc-Infrastruktur zu ermöglichen.

Implementierung und Visualisierung

Der Source-Code zu unserer Implementierung und Visualisierung des OLSR-Protokolls, ebenso wie das Video ist in folgendem Github-Repository zu finden:

<https://github.com/juengeja/MANet-Portfolio.git>

Die Implementierung wurde von Binh und Jannis durchgeführt, während Patrick sich um die Umsetzung der Visualisierung gekümmert hat. Wer welchen Teil der Ausarbeitung bearbeitet und geschrieben hat, ist jeweils der Fußzeile des jeweiligen Abschnitts zu entnehmen. Um die Aufnahme und den Schnitt des Videos hat Binh übernommen.

Wir möchten eine Gruppennote erhalten.

Matrikelnummern:

Elisa Popp - 7765345

Binh Dich - 2007710

Jannis Jüngert - 2079264

Leon Engelhardt - 5527999

Cliffordrichard Okoro - 8158562

Patrick Vollstedt - 2571435

Literaturverzeichnis

- (Campista & Rubinstein, 2014): Campista, M. E. M., & Rubinstein, R. M. G. (2014). *Advanced routing protocols for wireless networks*. John Wiley & Sons, Incorporated. Pages 35f.
- (Wodczak, 2018): Wodczak, M. (2018). *Autonomic intelligence evolved cooperative networking*. John Wiley & Sons, Incorporated. Pages 138f.
- (HowTo, 2015): HowTo. (2015). *Optimized Link State Routing (OLSR) Mobile Adhoc Network Proactive Routing Protocol*. Link: <https://www.youtube.com/watch?v=3V19nPxpMp8&t=181s> (Stand: 09.04.22, 21:49 Uhr)
- (Azzuhri, Suhaimi & Wong, 2022): Saaidal Razalli Azzuhri, Suhazlan Suhaimi und K. Daniel Wong (Stand: 11.04.2022). <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.458.5677&rep=rep1&type=pdf>
- (Ad-hoc-Netz, 2022): Wikipedia (2022, 9. Januar). *Ad-hoc-Netz*. <https://de.wikipedia.org/wiki/Ad-hoc-Netz>
- (frwiki, 2022): frwiki (Stand: 08.04.2022). *Optimiertes Verbindungsstatus-Routing-Protokoll*. https://de.frwiki.wiki/wiki/Optimized_link_state_routing_protocol
- (BATMAN, 2022): frwiki (Stand: 08.04.2022). *BATMAN (Protokoll)*. [https://de.frwiki.wiki/wiki/BATMAN_\(protocole\)](https://de.frwiki.wiki/wiki/BATMAN_(protocole))
- (Clausen und Jacquet, 2003): Clausen, T. H., Polytechnique, É., & Jacquet, P. (2003). *Optimized link state routing protocol (OLSRP) Routing Protocols for the IoT View project tracking topics and trends on social networks via analytic pattern matching*. View project. <https://www.researchgate.net/publication/270394473>
- (Paul Gardner-Stephen, 2013): Paul Gardner-Stephen. (24. Januar. 2013). *Serval Technology Stack (Part 2) - Rhizome*. <http://servalpaul.blogspot.com/2013/01/serval-technology-stack-part-2-rhizome.html>

-
- (AARON MAK, 2021): AARON MAK. (2021). *Why Google's Internet-Beaming Balloons Ran Out of Air*. *Slate*. https://slate.com/technology/2021/01/loon-google-alphabet-shuttered.html?via=rss_socialflow_twitter
 - (Flinders University, 2020): Flinders University. (2020). *Serval project*. <https://www.flinders.edu.au/about/making-a-difference/serval-project>
 - (Katikala, 2014): Soujanya Katikala. (2014). *GOOGLE™ PROJECT LOON*. https://www2.rivier.edu/journal/ROAJ-Fall-2014/J855-Katikala_Project-Loon.pdf
 - (Gordon Ecco, 2015): Gordon Ecco. (2015). *Google Loon: Internet für die Welt*.
 - (Der Spiegel, 2022): Der Spiegel. (2022). *Elon Musk hilft Ukraine über Satelliten mit Internet aus*. *Der Spiegel*.
 - (Evakuierungen & Hochschule Wildau, 2014): Evakuierungen, und, & Hochschule Wildau, T. (2014). *Entwurf eines Systemkonzeptes zur Unterstützung der gezielten mobilen Informationsweitergabe bei Großereignissen, Katastrophen*.
 - (AC Boley, 2022): AC Boley, E. W. S. L. P. H. (2022). *Observations of Starlink Satellites*. *Iopscience*.
 - (Uwe. R, 2022): Uwe. R. (2022). *Starlink – Satelliten Internet kommt und verändert die Welt*. *RockyourGoal*.
 - (Marton, 2022): Marton, M. (2022). *Stars, Satellites, and SpaceX*. *The E&S Magazine*.