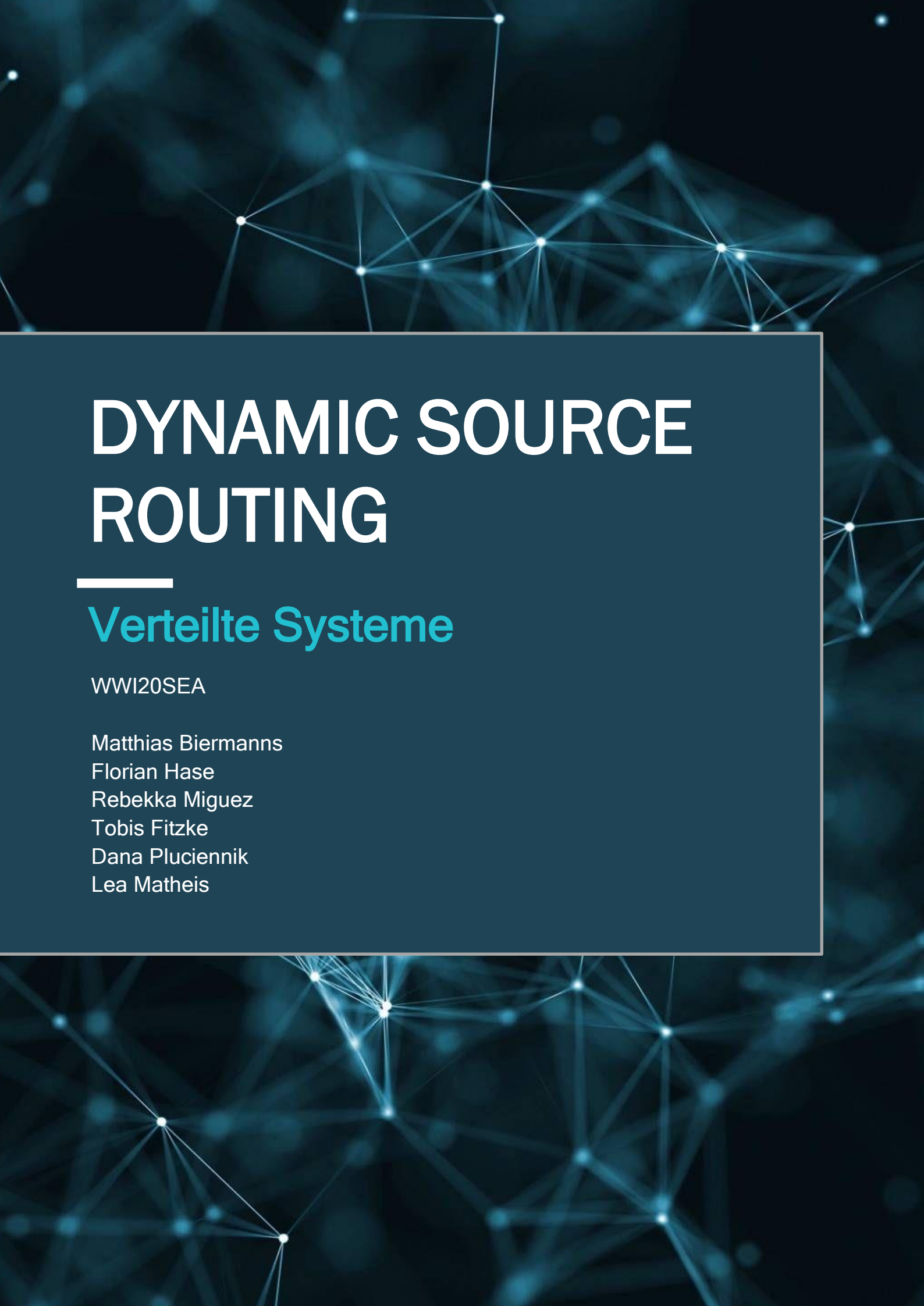


DYNAMIC SOURCE ROUTING

Verteilte Systeme

WWI20SEA

Matthias Biermanns
Florian Hase
Rebekka Miguez
Tobis Fitzke
Dana Pluciennik
Lea Matheis

Inhalt

Vorwort

Die vorliegende Ausarbeitung beinhaltet die Erarbeitungen der Aufgabenteile A-C der Portfolio-Aufgabe als Prüfungsleistung für das Modul „Verteilte Systeme“.

Einteilung der Aufgaben:

Matthias Biermanns:	Entwicklung des Codes
Florian Hase:	Entwicklung des Codes
Rebekka Miguez:	Aufgabenteil A
Tobias Fitzke:	Aufgabenteil B
Dana Pluciennik:	Aufgabenteil C
Lea Matheis:	Aufnahme des Videos für die Aufgabenteile A und B

Wir möchten bitte eine Gruppennote erhalten!

Portfolio-Aufgabenteil A

Abstract

Grundsätzlich lassen sich die für Mobile Ad-Hoc Netze (MANets) entwickelte Routingverfahren in drei Kategorien aufteilen: Tabellen basierte, auf Anfrage basierte und hybride Protokolle.[1] Bei tabellenbasierten Protokollen, solche wie das DSDV (Destination-Sequenced Distance Vector) Routingverfahren, hat jede Station eine Tabelle mit Information über die Topologie des gesamten Netzes. Wir haben uns dagegen für eine auf Anfrage basierten Protokoll entschieden, die nur nach Routing Pfaden sucht, wenn eine gebraucht wird und nicht bereits vorhanden ist.[2]

Konkret fiel unsere Wahl auf das Dynamic Source Routing (DSR). Dieses Routingverfahren adaptiert sich schnell auf Veränderungen und ist entsprechend unabhängig von umgebungsbedingten Gegebenheiten.[3] Das DSR Verfahren setzt sich aus den Vorgängen Route Discovery and Route Maintenance zusammen. Innerhalb des Vorgangs der Route Discovery wird eine Route Request wird so lange multicasted bis die Zielstation erreicht wird. Dabei ist die Route Request durch eine eindeutige ID gekennzeichnet. Erst, wenn die Route Request bei der Zielstation ist, kann eine Route Reply an die Ausgangsstation zurück übertragen werden. Nach diesem Schritt erfolgt die Route Maintenance Phase. In dieser Phase wird das Nachrichten Packet an die Zielstation weiter versendet. Falls ein Fehler beim Versenden der Nachricht auftritt, wird ein Route Error an die Ausgangsstation versendet und der Vorgang der Route Discovery wird wiederholt.[4]

[1] A. R. Zarzoor, "Enhancing dynamic source routing (DSR) protocol performance based on link quality metrics," *2021 International Seminar on Application for Technology of Information and Communication (iSemantic)*, 2021, pp. 17-21, doi: 10.1109/iSemantic52711.2021.9573233

[2] G. R. Pathak, S. H. Patil, A. D. Rana and Y. N. Suralkar, "Mathematical model for routing protocol performance in NS2: Comparing DSR, AODV and DSDV as example," *2014 IEEE Global Conference on Wireless Computing & Networking (GCWCN)*, 2014, pp. 184-188, doi: 10.1109/GCWCN.2014.7030875

[3] Johnson, David B. and David A. Maltz,"Dynamic Source Routing in Ad Hoc Wirless Networks", Mobidata,1994, doi:10.1007/978-0-585-29603-6

[4] Istikmal, "Analysis and evaluation optimization dynamic source routing (DSR) protocol in Mobile Adhoc network based on ant algorithm," *2013 International Conference of Information and Communication Technology (ICoICT)*, 2013, pp. 400-404, doi: 10.1109/ICoICT.2013.6574609.

Anmerkung

Logfiles auf, die referiert werden, sind im Log Ordner aufzufinden und unterteilt nach den Testfunktionen, die im Laufe des Textes einzeln erläutert werden.

Versenden von einer Hallo-Nachricht

Basierend auf die Topologie eines Mobile Ad hoc Network (MANet), werden Endgeräte als Router dargestellt und miteinander vernetzt. In unserer Umsetzung wird jedes Endgerät durch zwei Objekte simuliert, dem Router und dem EndDevice. Der Router führt das DSR-Protokoll aus und übernimmt entsprechend die Routing-Aufgaben. Das EndDevice simuliert den Vorgang zum Verschicken von Nachrichten. Diese zwei Objekte lassen sich durch ihre aufeinanderfolgenden Ports unterscheiden. Der Port eines Routers ist immer gerade (z.B. 3000), während der Port eines EndDevice (z.B. 3001) immer ungerade ist. Zur Veranschaulichung unseres umgesetzten DSR-Routingverfahren, wird über die Testmethode sendMessage() eine “Hallo” Nachricht von EndDevice 3001 an das EndDevice 3011 versendet. Dabei erfolgt folgender Prozess:

1. Als Erstes erfolgt die Route Discovery Phase. Dabei sendet 3001 eine Route Request und es wird multicasted. Eine Route Request kommt bei 3002 an.

   Logfile 3002

ID:	*180db56b-17c9-449f-b015-7b8175f3e0aa*
Command:	*RouteRequest*
Source Port:	*3000*
Destination Port:	*3011*
Path:	*[3000]*
Content:	

   Die Route Request wird weiter multicasted und kommt bei 3004 an. Die Request hat immer eine eindeutige ID. Wenn die Route Request ID von einem Router zum ersten Mal gesehen wurde, wird die eigene Adresse an den Pfad angehängt.

Logfile 3004

ID:	*180db56b-17c9-449f-b015-7b8175f3e0aa*
Command:	*RouteRequest*
Source Port:	*3000*
Destination Port:	*3011*
Path:	*[3000, 3002]*
Content:	

Es wird weiter multicasted bis der Router 3010 erreicht wird. Dann sendet 3010 eine Route Reply zurück.

LogFile 3010

ID:	*180db56b-17c9-449f-b015-7b8175f3e0aa*
Command:	*RouteRequest*
Source Port:	*3000*
Destination Port:	*3011*
Path:	*[3000, 3002, 3004, 3006, 3008]*
Content:	

LogFile 3000

ID:	*180db56b-17c9-449f-b015-7b8175f3e0aa*
Command:	*RouteReply*
Source Port:	*3010*
Destination Port:	*3000*
Path:	*[3000, 3002, 3004, 3006, 3008, 3010]*
Content:	

Dadurch erhält der Router den Pfad, um die Nachricht an die Zielstation zu verschicken.

LogFile 3000

Path Cache: [3000, 3002, 3004, 3006, 3008, 3010]

Known Ids: 180db56b-17c9-449f-b015-7b8175f3e0aa

2. Als Zweites erfolgt die Route Maintenance Phase. Dabei sendet 3001 das Nachrichten Packet. Am Pfad orientiert wird es an 3002 versendet. 3002 sendet über einen Forward die Nachricht an 3004 weiter. Zur Überprüfung, ob das Packet weitergeleitet wurde, sendet 3002 nach dem erfolgreichen Senden eines Forwards ein Acknowledgment an 3000. Dieser Prozess wird so lange ausgeführt, bis die Nachricht bei Device 3011 ankommt.

Logfile 3002

ID:	*e28d48ee-33a8-4651-83c1-a43f6f0c9a86*
Command:	*Forward*
Source Port:	*3001*
Destination Port:	*3011*
Path:	*[3000, 3002, 3004, 3006, 3008, 3010]*
Content:	*Hallo*

Logfile 3000

ID:	*e28d48ee-33a8-4651-83c1-a43f6f0c9a86*
Command:	*Ack*
Source Port:	*3002*
Destination Port:	*3000*
Path:	*[3000, 3002, 3004, 3006, 3008, 3010]*
Content:	

Versenden von zwei Nachrichten an die gleiche Zielstation

Bei der Testmethode sendMessageParallel() wird von EndDevice 3001 eine Hallo1 Nachricht an EndDevice 3005 verschickt. Parallel sendet die EndDevice 3003 eine Hallo2 Nachricht ebenfalls an die EndDevice 3005. Beide Nachrichten kommen auch bei der Zielstation 3005 an.

Logfile 3000

ID:	*932e4c90-1421-4839-9d58-7a2277705fd7*
Command:	*Send*
Source Port:	*3001*
Destination Port:	*3005*
Path:	*[]*
Content:	*Hallo1*
ID:	*8c093178-07b2-4cde-9807-850816db21e2*
Command:	*RouteRequest*
Source Port:	*3002*
Destination Port:	*3005*
Path:	*[3002]*

Logfile 3002

ID:	*41bd738e-bf44-491c-95c5-e0042b26833d*
Command:	*Send*
Source Port:	*3003*
Destination Port:	*3005*
Path:	*[]*
Content:	*Hallo2*
ID:	*182c0f35-c426-4693-83cb-810552f73782*
Command:	*RouteRequest*
Source Port:	*3000*
Destination Port:	*3005*
Path:	*[3000]*
Content:	

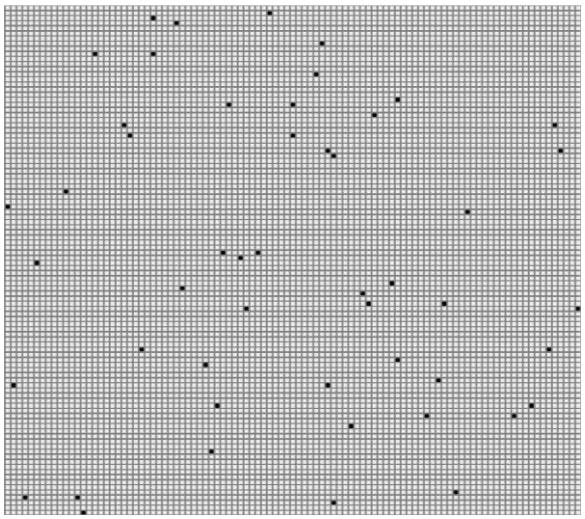
Logfile 3005

ID:	*932e4c90-1421-4839-9d58-7a2277705fd7*
Command:	*Forward*
Source Port:	*3001*
Destination Port:	*3005*
Path:	*[3000, 3004]*
Content:	*Hallo1*

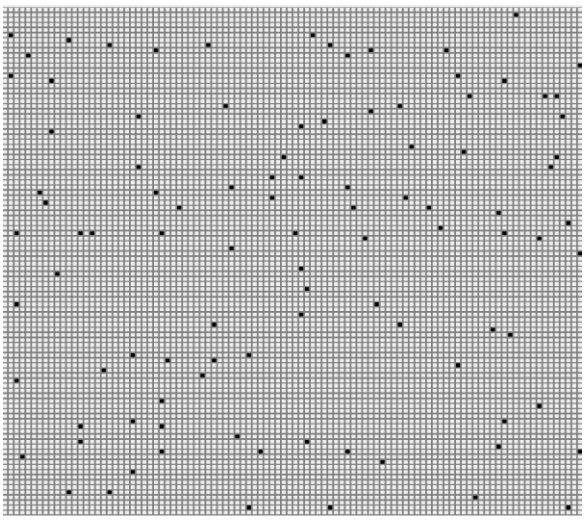
ID:	*41bd738e-bf44-491c-95c5-e0042b26833d*
Command:	*Forward*
Source Port:	*3003*
Destination Port:	*3005*
Path:	*[3002, 3004]*
Content:	*Hallo2*

Testläufe mit unterschiedlichen Knotenzahlen und variierenden räumlichen Verteilungen

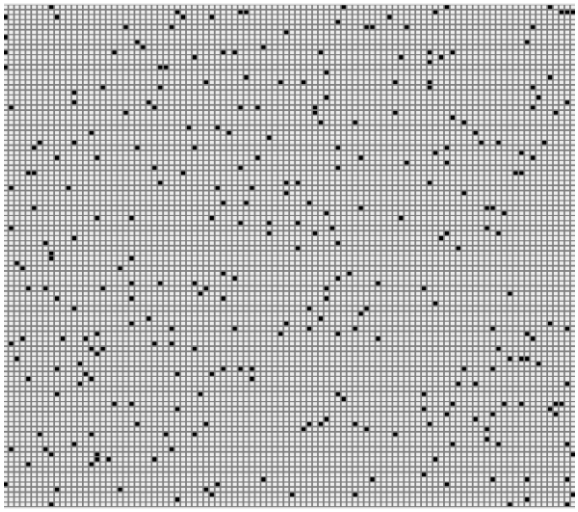
Mithilfe der Testmethode differentNumberOfRouters() wurde getestet, inwiefern die Anzahl an Router sich auf das Netz auswirken. Zum Test wurde eine Hallo Nachricht vom Device 3001 an 3003 gesendet. Dabei fiel auf, dass bei einer geringen Anzahl an Router, die im 100x100 Feld zufällig verteilt sind, die Nachricht nicht ankommt, weil die Distanz zu dem einzelnen Router zu groß ist.
Bei einem Durchlauf mit 50 oder 100 Router ist die Hallo Nachricht nicht an das Device 3003 angekommen. Das erkennt man unter anderem dadurch, dass im Router 3000 keine Route Reply zu sehen ist. Das bedeutet, dass innerhalb der Route Discovery Phase die Zielstation nicht erreicht, werden konnten.



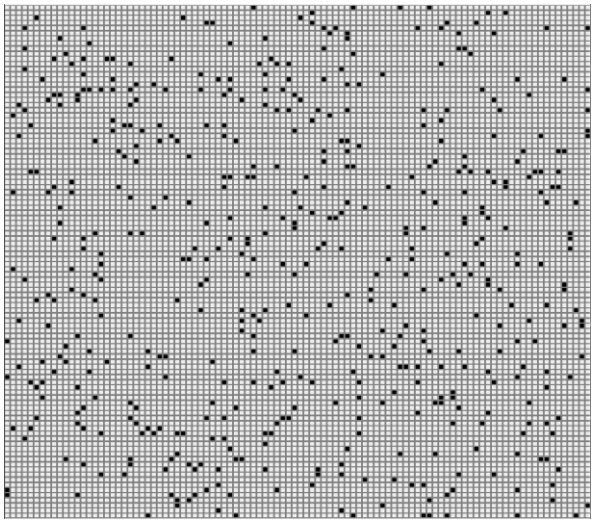
50 Router



100 Router



300 Router



500 Router

Bei einem Durchlauf mit 300 oder 500 Router kommt dagegen die Hallo Nachricht auch nach mehreren Durchläufen mit neuen Positionen der Router an.

Knotenzahl mit der man mit ca.90% Wahrscheinlichkeit ein vollständig verbundenes Netz in einem 100x100 Feld hat

Mittels der Funktion isNetzVermascht() aus der Field-Klasse wird überprüft, ob das Netz vollständig vermascht ist. Um zu untersuchen ab welcher Knotenzahl bei einer 90% Wahrscheinlichkeit ein solches vermaschtes Netz auftritt, wird mithilfe der Testmethode testNintyPercentVermascht() die Anzahl an Router zum Testen festgelegt und so lange hoch addiert bis zu der Anzahl ein vermaschtes Netz auftritt. In den ersten Durchläufen traten vermaschte Netze ab einer Knotenzahl von ca. 168 bis 250 auf. Manchmal können vermaschte Netze bei einer geringen Knotenzahlen auftauchen, aber diese treten nur vereinzelt in verschiedenen Durchgängen auf und haben somit keine 90% Wahrscheinlichkeit. Bei genauer Betrachtung diesen Zahlenspektrums wurde deutlich, dass die 90% Wahrscheinlichkeit in den häufigsten Fällen bei einer Knotenanzahl über 200 erreicht wird. Beim Testen der Router Startwerte 205, 210, 215 und 216 kam heraus, dass bei allen bei einer 90% Wahrscheinlichkeit ein vollständig verbundenes Netz bei einer Knotenanzahl von 216 auftreten. Auch beim mehrmaligen Austesten zur Evaluierung ihrer Aussagekraft liegt die Knotenzahl eines voll vermaschten Netz bei 216.

Anmerkung

Bei der Umsetzung des Testvorgang, um zu bestimmen ab welcher Knotenzahl zu 90% ein voll vermaschtes Netz auftritt, sind im Laufe des Projektes im Code Veränderungen aufgetreten. Somit sind die im Video genannten 148 Knotenzahl nicht mehr aktuell. Das Ergebnis liegt mittlerweile bei einer Knotenzahl von 216.

Portfolio-Aufgabenteil B

1 Verschiedene Stationen bewegen:

Die Testmethode moveRandomRandomRouter() bewegt alle 1-5 Sekunden eine zufällig gewählte Station an eine neue Koordinate auf dem Feld. Um dies anhand der Logs nachvollziehen zu können wurde die Methode dreimal auf einem 100x100 Feld mit 30 Stationen ausgeführt und jeweils 30 Sekunden laufen gelassen. Die genauen Logs sind im Anhand in dem Zip-Ordner, welcher den Namen der Methode trägt und die drei genannten Testdurchläufe beinhaltet.

In den einzelnen Durchgängen befinden sich nun die Logs der einzelnen Stationen. Wurde ein Station während dieses Vorgangs bewegt sieht das Log wie folgt aus (Station 3036 aus Durchlauf 1):

Apr. 19, 2022 4:19:58 PM Router setUpLogger
INFO: Router startet logging
Position: (x: 35; y: 12)
Port: 3036

Apr. 19, 2022 4:20:04 PM Router logNewPosition
INFO:
New Position: (x: 49; y: 1)

Jeder Durchlauf beinhaltet zusätzlich ein Logfile des gesamten Feldes. Dieses bietet eine sehr übersichtliche Darstellung aller getätigten Movements (Ausschnitt aus Feld-Logfile aus Durchlauf 2):

Apr. 19, 2022 4:21:57 PM Field moveDevice
INFO: Devices 3016, 3017[5] *moved from x: 22, y: 64 to x: 71, y: 31*

[5] Router-Port, EndDevice-Port

Apr. 19, 2022 4:21:59 PM Field moveDevice
INFO: Devices 3000, 3001 moved from x: 86, y: 38 to x: 32, y: 91

Apr. 19, 2022 4:22:03 PM Field moveDevice
INFO: Devices 3018, 3019 moved from x: 14, y: 26 to x: 75, y: 45

Apr. 19, 2022 4:22:07 PM Field moveDevice
INFO: Devices 3000, 3001 moved from x: 32, y: 91 to x: 87, y: 53

Apr. 19, 2022 4:22:10 PM Field moveDevice
INFO: Devices 3032, 3033 moved from x: 80, y: 43 to x: 70, y: 6

Apr. 19, 2022 4:22:10 PM Field moveDevice
INFO: Devices 3024, 3025 moved from x: 10, y: 29 to x: 1, y: 52

Apr. 19, 2022 4:22:13 PM Field moveDevice
INFO: Devices 3004, 3005 moved from x: 50, y: 12 to x: 86, y: 87

2 Versenden von Nachrichten während sich Stationen

Dieser Testfall wird durch die Methode sendMessageWhileMovingRouter() realisiert. Hier wurden 3 Durchläufe auf einem 100x100 Feld mit 250 Routern durchgeführt, wobei sich im 1. Durchlauf alle 2 Sekunden ein zufälliger Router bewegt und eine Nachricht von einem zufälligen Router an einen anderen zufälligen Router versendet wird. Im 2. Durchlauf wird dies jede halbe und im 3. Durchlauf alle 99 Millisekunden wiederholt. Das Bewegen von Routern und senden von Nachrichten wird in jedem Durchlauf jeweils 20-mal wiederholt, sodass am Ende von jedem Durchlauf gleich viele Move und Send Befehle ausgeführt wurden und die Daten im Anschluss verglichen werden können. Der erste Durchlauf dauert dementsprechend 40 Sekunden, der zweite 10 Sekunden und der dritte knapp 2 Sekunden. Alle Logs der Durchläufe sind im bereits beschrieben Ordner zu finden.

Um zu erklären was in diesen Durchläufen passiert betrachten wir zuerst den ersten Durchlauf. Hier ist wie im oben beschrieben Test im Field Logfile zu sehen, welche der 250 Router sich in den 20 Durchgängen mit jeweils 2 Sekunden Abstand bewegt haben.

Bspw.:

Apr. 20, 2022 5:54:27 PM Field moveDevice
INFO: Devices 3390, 3391 moved from x: 82, y: 30 to x: 17, y: 11

Zudem wurden, wie bereits beschreiben auch alle 2 Sekunden neue Nachrichten versendet. Bspw. Wurde die Nachricht „Hallo from 3119 to 3105“ von Device 3119 zu Device 3105 versendet.
Im Logfile des Routers 3119 ist dementsprechend zu sehen, wann diese Nachricht versendet, wurde:

Apr. 20, 2022 5:54:57 PM Router evaluateMessage
INFO:

ID:	*cbe26fed-4da4-498a-8067-bacfdf708b96*
Command:	*Send*
Source Port:	*3119*
Destination Port:	*3105*
Path:	*[]*
Content:	*Hallo from 3119 to 3105*

Im Logfile des Empfängers (hier 3105) ist entsprechend zu sehen, wann, über welchen Pfad diese Nachricht angekommen ist:

Apr. 20, 2022 5:54:58 PM Router evaluateMessage
INFO:

ID:	*cbe26fed-4da4-498a-8067-bacfdf708b96*
Command:	*Forward*
Source Port:	*311*
Destination Port:	*3105*
Path:	*[3118, 3026, 3394, 3198, 3014, 3304, 3078, 3406, 3008, 3338, 3146, 3362, 3104]*
Content:	*Hallo from 3119 to 3105*

Betrachtet man nun den kompletten ersten Durchlauf so lässt sich festhalten, dass von zufälligen gewählten Routern von 20 versendeten Nachrichten 20 erfolgreich ankamen.

Im zweiten Durchlauf konnten ebenfalls alle 20 Nachrichten übermittelt werden.

Im dritten Durchlauf konnten von den 20 Nachrichten lediglich 17 erfolgreich gesendeten werden. Folgende 3 Nachrichten konnten nicht übermittelt werden:

Hallo from 3035 to 3181
Hallo from 3253 to 3219
Hallo from 3427 to 3123

Da, wie bereits beschreiben, das Feld bereits mit 216 Routern zu 90% vollvermascht ist, ist davon auszugehen, dass bei den hier verwendeten 250 Routern weder Start noch Empfangsrouter außerhalb der Reichweite eines anderen Routers sind. Dementsprechend lassen sich die Auftretenden Fehler auf die Dauer der RouteDiscovery und/oder der RouteMaintenance zurückführen. Da das Netz auf Grund der vielen und schnellen Bewegungen recht instabil ist (mit zunehmender Geschwindigkeit instabiler wird),

ist es entweder nicht möglich einen Pfad zwischen den beiden Devices zu finden oder es ist möglich einen Pfad zu finden, dieser ist jedoch bereits wieder ungültig, sobald die Nachricht versendet wird.

3 Versenden von Nachrichten während sich Stationen bewegen (mit weniger Routern):

Das in 2. Erreichte Ergebnis ist auf Grund der hohen Routeranzahl nicht so schlecht, wie man es evtl. bei den im Test verwendeten Zeiten (Abständen) vermute würde. Um zu zeigen, welchen Einfluss die Routeranzahl hat, wird im Folgenden die gleiche Methode erneut mit nur 175 Routern durchgeführt. Dabei wird es einmal einen Abstand von 2 Sekunden (4. Durchlauf) und einmal einen Abstand von 99 Millisekunden (5. Durchlauf) geben.

Im 4. Durchlauf schaffen es die 2 folgenden Nachrichten nicht ihr Ziel zu erreichen:

Hallo from 3041 to 3073
Hallo from 3197 to 3257

Im 5. Durchlauf schaffen es die 9 folgenden Nachrichten nicht ihr Ziel zu erreichen:

Hallo from 3043 to 3193
Hallo from 3051 to 3281
Hallo from 3077 to 3051
Hallo from 3109 to 3327
Hallo from 3219 to 3157
Hallo from 3227 to 3019
Hallo from 3307 to 3331
Hallo from 3317 to 3233
Hallo from 3321 to 3267

Somit lässt sich schlussfolgern, dass wenn die Anzahl der Router abnimmt, der Zeitabstand zwischen dem Verschieben und Senden jedoch gleichbleibt, die Anzahl an nicht zustellbaren Nachrichten steigt (vgl. Durchlauf 1 & 5). Verringert man zusätzlich auch noch den Abstand zwischen Bewegen und Senden so steigen ebenfalls die nicht zustellbaren Nachrichten (vgl. Durchlauf 1 und 3 im Vergleich zu 4 und 5). Dabei ist jedoch nicht zu vernachlässigen, dass bei dem 4. & 5. Durchgang nun 41 Router weniger verwendet wurden, als für ein 90%ig vermaschtes Netz notwenig wären. Dementsprechend ist hier nun neben den in 2. Genanten Gründen für das scheiten der Nachrichtenübermittlung noch zu ergänzen, dass es nun auch möglich ist, dass Start oder Empfangsdevice zu weit von anderen Routern entfernt sind.

Allgemein darf bei allen Tests nicht vergessen werden, dass alle durchgeführten Tests nur Momentaufnahmen sind und durch die vielen zufälligen Werte immer verschieden ausfallen. Dementsprechend lassen sich mit diesen Tests keine 100%igen Verhaltensmuster erklären, die Tendenz ist jedoch trotzdem erkennbar.

Portfolio-Aufgabenteil C

In der folgenden Ausarbeitung des Aufgabenteils C werden zunächst die Vor- und Nachteile des, von unserer Gruppe ausgewähltem, Dynamis Source Routings (DSR) dargestellt. Des Weiteren werden identifizierte Problemfälle in unserem System aufgezeigt und mögliche Lösungsansätze vorgeschlagen.
Der letzte Abschnitt dieses Aufgabenteils befasst sich mit möglichen Alternativen zu Ad-Hoc-Netzen in Krisengebieten mit einer zerstörten Infrastruktur.

Vorteile

- Das DSR bietet die Möglichkeit zum spontanen und schnellen Datenaustausch.
- Das DSR garantiert ein schleifenfreies Routing.[6]
- Der Overhead der Routenwartung wird reduziert, da die Routen nur zwischen den Knoten gepflegt werden, die auch miteinander kommunizieren müssen.[7]
- Die Anzahl der Routinganfragen wird minimiert, indem die Knoten nur die Route-Request verarbeiten, die bisher noch nicht gesehen wurden.[8]
- Zwischenknoten müssen keine aktuellen Routinginformationen pflegen, um die Pakete weiterzuleiten, sondern nutzen die Routencacheinformationen. Durch die Nutzung der Routencacheinformationen wird der Kontrollaufand verringert.[9]

---

[6] Vgl. International Journal of Wired and Wireless Communications Vol.1, Issue 2, April, 2013, online: We have discussed the related issues and advantages and disadvantages of Reactive and Proactive routing protocol in WMNs (psu.edu) zuletzt aufgerufen am 19.04.2022

[7] Vgl. DYNAMIC SOURCE ROUTING - EIN ONDEMAND-ROUTINGPROTOKOLL IN MOBILEN AD-HOC-NETZWERKEN | Open-Access-Zeitschriften (rroij.com) zuletzt aufgerufen am 19.04.2022

[8] Vgl. Dynamic Source Routing protocol (DSR): Algorithm, Example, Advantages, Disadvantages (brainkart.com) zuletzt aufgerufen am 19.04.2022

[9] Vgl. Dynamic Source Routing protocol (DSR): Algorithm, Example, Advantages, Disadvantages (brainkart.com) zuletzt aufgerufen am 19.04.2022

- Das Netzwerk muss nicht regelmäßig mit Aktualisierungsnachrichten überflutet werden, da das DSR einen reaktiven Ansatz verwendet und kein tabellengesteuerten Ansatz. Das vermeidet Bandbreitenverschwendung.[10]
- Der Routing-Prozess stellt sicher, dass vorkommende Änderungen an alle verbundenen Router gesendet werden.[11]
- Es besteht keine Notwendigkeit eine Routing-Tabelle in jedem Knoten zu hinterlegen.[12]

Nachteile

- Die Verzögerung beim Verbindungsaufbau ist höher als bei einem tabellengesteurten Ansatz.[13]
- Die Paketheadergröße nimmt aufgrund des Quellroutings mit der Länge der Route zu.[14]
- Die Leistung nimmt mit zunehmender Mobilität rapide ab.[15]
- Das DSR ist für große Netzwerke nicht skalierbar und erfordert deutlich mehr Verarbeitungsressourcen als die meisten anderen Protokolle.[16]

---

[10] Vgl. The International Journal Of Engineering And Science (IJES), Volume 1, Issue 2, Pages 54-60, 2012, ISSN: 2319 – 1813 ISBN: 2319 – 1805 zuletzt aufgerufen am 19.04.2022

[11] Vgl. 7 Vor- und Nachteile von Dynamic Routing | Nachteile & Vorteile von Dynamic Routing (hitechwhizz.com) zuletzt aufgerufen am 19.04.2022

[12] Vgl. DSR Vor- und Nachteile | Tabelle herunterladen (researchgate.net) zuletzt aufgerufen am 19.04.2022

[13] Vgl. The International Journal Of Engineering And Science (IJES), Volume 1, Issue 2, Pages 54-60, 2012, ISSN: 2319 – 1813 ISBN: 2319 – 1805 zuletzt aufgerufen am 19.04.2022

[14] Vgl. DYNAMIC SOURCE ROUTING – EIN ONDEMAND-ROUTINGPROTOKOLL IN MOBILEN AD-HOC-NETZWERKEN | Open-Access-Zeitschriften (rroij.com)
zuletzt aufgerufen am 19.04.2022

[15] Vgl. The International Journal Of Engineering And Science (IJES), Volume 1, Issue 2, Pages 54-60, 2012, ISSN: 2319 – 1813 ISBN: 2319 – 1805 zuletzt aufgerufen am 19.04.2022

[16] Vgl. DYNAMIC SOURCE ROUTING – EIN ONDEMAND-ROUTINGPROTOKOLL IN MOBILEN AD-HOC-NETZWERKEN | Open-Access-Zeitschriften (rroij.com) zuletzt aufgerufen am 19.04.2022

- Der Mechanismus der Routenwartung repariert ausgefallene Verbindungen nicht lokal.[17]
- Eine Verbindung mit geringer Geschwindigkeit kann drastisch mehr Bandbreite benötigen und Bandbreitenengpässe verursachen.[18]
- Wenn es sehr viele Knoten im Netz gibt, dann kann das Senden der Round-Request-Nachricht von der Quelle an alle Nachbarsknoten einen Antwortansturm verursachen, der wiederum Kollisionen von Pakten zur Folge haben kann.[19]

[17] Vgl. The International Journal Of Engineering And Science (IJES), Volume 1, Issue 2, Pages 54-60, 2012, ISSN: 2319 – 1813 ISBN: 2319 – 1805 zuletzt aufgerufen am 19.04.2022

[18] Vgl. 7 Vor- und Nachteile von Dynamic Routing | Nachteile & Vorteile von Dynamic Routing (hitechwhizz.com) zuletzt aufgerufen am 19.04.2022

[19] Vgl. DYNAMIC SOURCE ROUTING - EIN ONDEMAND-ROUTINGPROTOKOLL IN MOBILEN AD-HOC-NETZWERKEN | Open-Access-Zeitschriften (rroij.com)
zuletzt aufgerufen am 19.04.2022

Welche Probleme bestehen noch in unserem System?

Wir konnten in unserem System ein Problemfall identifizieren, zudem wir im Folgenden versucht haben einen passenden Lösungsansatz zu definieren.

Problemsituation:

Ein Router sendet einen Route-Request an seine Nachbarsknoten los. Das Netzwerk ist allerdings nicht mehr vollständig miteinander verbunden und es kommt nie eine Antwort auf den losgeschickten Route-Request und der Router selbst erkennt das unvollständige Netzwerk nicht und der Route-Request geht somit verloren.

Lösungsvorschlag:

Die Problematik mit dem verlorengehenden Route-Requests bei Vorliegen eines nicht mehr verbundenen Netzwerkes könnte eventuell über den EndDevice gelöst werden, indem der Route-Request erneut gesendet wird.
Durch die Implementierung eines Timers, der nach einer gewissen Zeit abläuft, wird signalisiert, dass auf den gesendeten Route-Request keine Antwort kommt. Der EndDevice könnte dann entscheiden, den Route-Request nochmals zu schicken.

Alternative zu Ad-Hoc Netzen in Krisengebieten mit zerstörter Infrastruktur

Allgemein lassen sich Netzwerke in zwei verschiedene Netztypen unterscheiden: Statische Netze (auch Infrastrukturnetze genannt) und Dynamische Netze (auch Ad-Hoc-Netz genannt).

Bei statischen Netzen handelt es sich um eine bestehende Infrastruktur, in welcher sich beliebig viele Endgeräte über Access Points verbinden können. Die verschiedenen Geräte bilden in Verbindung mit den Access Points ein sogenanntes Basic Service Set (BSS). Mithilfe eines Distribution System (DS) ist es möglich mehrerer BSS miteinander zu verbinden (beliebig via Kabel oder Funk). Auf Basis dieser zusammenfassenden Vorstellung ist festzuhalten, dass ein Infrastrukturnetzwerk dann verwendet werden sollte, wenn ein dauerhaftes Netzwerk eingerichtet werden soll, dessen Reichweite sich ganz einfach über das Hinzufügen weiterer Access Points erweitern lässt.[20]

Dynamische Netze organisieren sich selbst und bauen sich nur für die Dauer der Kommunikation auf. Das Routing wird nicht wie bei den statischen Netzen von Access Points übernommen, sondern über die Knoten des dynamischen Netzwerkes ausgeführt. Durch den schnellen, temporären und dynamischen Verbindungsaufbau ist diese Art von Netzwerk an beliebigen Orten und unter beliebigen Bedingungen aufspannbar. Dazu gehören auch Gebiete mit einer fehlenden Netzwerkinfrastruktur.[21]

In Krisengebieten mit einer zerstörten und somit nicht mehr funktionstüchtigen Infrastruktur können keine Infrastrukturnetze aufgespannt werden, um einen Datenaustausch zu realisieren. Lediglich Ad-Hoc-Netze können jederzeit und unabhängig jeglicher Bedingungen aufgestellt werden.

[20] Vgl. Konnektivität des Internets der Dinge - learn.sparkfun.com zuletzt aufgerufen am 19.04.2022
[21] Vgl. Ad-hoc-Netz :: ad-hoc network (AHN) :: ITWissen.info zuletzt aufgerufen am 19.04.2022

In den folgenden Absätzen geht es nun darum Optionen aufzuzeigen, einen Kommunikationsweg in oben beschriebenen Situationen herzustellen, ohne dabei auf bestehende Ad-Hoc Systeme zurückzugreifen. Um dies umsetzen zu können, muss also ein Weg gefunden werden ein statisches Netz flexibel an den benötigten Ort zu bringen.

Die vermutlich am naheliegendste Lösung ist das Satelliteninternet. Das System dahinter ist von der Idee her gar nicht all zu komplex:

Rund um die Erde kreisen mehrere tausend Satelliten, welche als Vermittler zwischen den einzelnen Empfängern und Bodenstationen dienen. So ist es möglich seinen Empfänger an einem beliebigen, jedoch von Satelliten abgedeckten, Bereich aufzustellen, die Antenne auszurichten und mithilfe der Satelliten Zugang zum Internet zu erlangen. Der aktuelle Marktführer in diesem Bereich ist Starlink, welcher aktuell den flächendeckendsten und zuverlässigsten Zugang zum Internet über das Weltall zur Verfügung stellt.[22]

Wie sich dieses System in Krisengebieten bewährt, wird sich anhand des aktuell andauernden Krieges in der Ukraine zeigen, wo Starlink mehrere Satelliten so aufgestellt hat, dass die Ukraine möglichst flächendeckend abgedeckt wird, um von überall aus mit entsprechenden Endgeräten eine Verbindung zum Internet herstellen und somit weltweit kommunizieren zu können. Offenbar sichert das Satelliteninternet der Ukraine selbst und den einheimischen Truppen eine stabile Kommunikation durch die Verfügbarkeit einer hohen Datentransferrate von Starlink.[23]

Folgender Artikel wirft eine andere Möglichkeit auf, ein Netzwerkaufbau in Krisengebieten zu realisieren:

Online, Focus (2010): Fliegende Roboterschwärme: Kommunikationsnetze für Krisen- und Einsatzgebiete - Videos - FOCUS Online, zuletzt aufgerufen am 19.04.2022.

[22] Vgl. Was ist Starlink? Das Satelliten-Internet von Elon Musk - Netzpiloten.de zuletzt aufgerufen am 19.04.2022

[23] Vgl. Ukraine nutzt Elon Musks Starlink für Drohnenangriffe | Wissen & Umwelt | DW | 25.03.2022 zuletzt aufgerufen am 22.04.2022

Dabei ist die Überlegung, Drohnen zu nutzen, um ein solches Statisches Netz aufspannen zu können. Dies hat den Vorteil, dass das Netz beliebig bewegt und erweitert werden kann. Allerdings gibt es bei dieser Idee auch einiges zu beachten. Es muss immer mindestens eine erreichbare Basis vorhanden sein, zu welcher das Drohnennetz Kontakt hat. Dies sorgt dafür, dass die Basis nicht allzu weit entfernt sein darf.[24]
Auch die Wahl des Routingverfahrens ist bei dieser Idee nicht uninteressant, da sich die Frage gestellt werden muss, was passiert, wenn eine Drohne ausfällt oder eine Drohne eventuell zurück zur Basis kommen muss, um zu laden. Nach Klärung all dieser Fragen klingt diese Idee jedoch sehr interessant, um schnell ein Netzwerk an einem sonst vom Internet abgeschnitten Gebiet aufzuspannen. Dieses könnte dann von jedem Endgerät genutzt werden und benötigt, im Vergleich zum Satelliteninternet, keinen zusätzlichen Empfänger.
In einem umkämpften Kriegsgebiet wäre der ledigliche Einsatz von Drohnen jedoch ungeeignet, da es zu vielen ausfallenden Drohnen kommen kann.

Im aktuellen Ukraine-Russland-Krieg werden Satelliten für die Bereitstellung eines stabilen Netzwerkes genutzt und Drohnen dann damit verbunden, um weitere wesentliche Informationen sammeln zu können.[25]

[24] Vgl. Fliegende Roboterschwärme: Kommunikationsnetze für Krisen- und Einsatzgebiete - Videos - FOCUS Online zuletzt aufgerufen am 22.04.2022
[25] Vgl. Ukraine nutzt Elon Musks Starlink für Drohnenangriffe | Wissen & Umwelt | DW | 25.03.2022 zuletzt aufgerufen am 22.04.2022

Eine weitere mögliche Lösung zur Kommunikation im Katastrophenfall soll folgendes Projekt bieten: „smarter: Smartphone-based Communication Networks for Emergency Response“.

Das Bundesministerium für Bildung und Forschung (BMBF) fördert dieses Projekt, um eine Lösung für infrastrukturunabhängige Notfallkommunikation über Smartphones zu realisieren. Dieses Projekt hat in Zusammenarbeit mit einigen Partnern, darunter auch der Technischen Universität Darmstadt, einen technischen Lösungsentwurf entwickelt, um im Katastrophenfall auch ohne Mobilfunknetz über Smartphones kommunizieren zu können. Die App smarter soll dabei Abhilfe schaffen. Dabei soll sich das Smartphone über den WLAN-Chip sofort mit einem anderen Smartphone verbinden, dass ebenfalls die App besitzt und nicht weiter als 250 Meter entfernt ist. So bildet sich ein großes Netzwerk, das den Hilferuf so lange über verschiedene Smartphones weiterleitet, bis es am Zielgerät angekommen ist.[26]

Dieser Lösungsansatz wurde beispielsweise für den Fall eines Stromausfalls entwickelt. So sollen die Smartphones Daten verschicken und empfangen können, ohne ein Funknetz zu benötigen.

Diese Idee weist jedoch noch einige Komplikationen auf:
Die Leistungsfähigkeit unserer Smartphones leidet rapide darunter, da die Netzwerkbereitstellung und -aufrechterhaltung von der Handy-Akku-Technik Höchstleistung verlangt. Die Offline-Chatfunktion muss ebenfalls noch optimiert werden. [27]

Schlussendlich lässt sich also sagen, dass diese Option in der Zukunft eine helfende Komponente im Prozess der Versorgung der Krisengebieten mit einer stabilen Verbindung darstellen könnte, dieses Problem jedoch nicht alleine lösen kann. Wie sich dieses Projekt in der Zukunft noch ausbaut, ist noch nicht bekannt, aber es bildet eine mögliche Alternative.

[26] Vgl. Kritische Infrastruktur schützen: Roboter und Drohnen als Krisenhelfer (handelsblatt.com) zuletzt aufgerufen am 22.04.2022
[27] Vgl. „Smarter“: App kann im Katastrophenfall ohne Mobilfunknetz kommunizieren | rettungsdienst.de zuletzt aufgerufen am 22.04.2022