UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR TELEMATIK

Aus dem Institut für Telematik
der Universität zu Lübeck
Direktor: Prof. Dr. Stefan Fischer

DNA-BASIERTE NANONETZWERKE

Inauguraldissertation
zur
Erlangung der Doktorwürde
der Universität zu Lübeck
Aus der Sektion Informatik/Technik

vorgelegt von
Florian-Lennert Adrian Lau, M.Sc.
aus Lübeck

Lübeck, 2019

1. Berichterstatter: Prof. Dr. rer. nat. Stefan Fischer
2. Berichterstatter: Prof. Dr. rer. nat. Ralf Möller

Tag der mündlichen Prüfung: 12.06.2020

Zum Druck genehmigt. Lübeck, den 08.07.2020

ABSTRACT

Nanodevices are minuscule, autonomous devices that interact at the nanoscale. Due to their small size, nanodevices are subject to resource constraints. To circumvent these, nanodevices may form nanonetworks. In doing so, nanonetworks may solve problems that exceed the capabilities of single devices.

Nanodevices and nanonetworks may be used to solve problems for which no satisfactory solution exists. Popular examples are medical scenarios, e.g., combating various forms of cancer. Other medical examples are the suppression of diseases during emergence or the local treatment of infections. In medicine, both problems haven't been solved yet and mostly systemic treatment is necessary.

Despite the various use cases, there are several unsolved problems in nanoscience. While it is possible to create primitive nanodevices under laboratory conditions, they can only be regarded as proof of concepts.

One of the biggest unsolved problems is the actual construction of the components required in nanonetworks. While several suggestions and metalinguistical machine models exist, those are insufficiently specified or no means of construction are given. Most existing ideas are limited to partial solutions of construction, communication or computation. The compatibility of the modules is generally not clarified. No concepts that combine construction, communication and computation in a holistic model for nanonetworks have been established yet.

This work presents a combined solution for the enumerated problems. The basic component for the presented approach is DNA. The inherent properties of this molecule may be used to assemble complex nanostructures. DNA can be utilized to create both nanodevices and a communication mechanism. Properly designed DNA-molecules can even be utilized for computational purposes.

Based on these DNA-technologies, a concept is presented that solves all named problems. The resulting components are then combined into a reference architecture for DNA-based nanonetworks. This reference architecture revolutionizes the prevalent paradigm of communication and computation in nanonetworks. In the presented paradigm, computations are outsourced into the assembly process of molecules in the communication channel.

This work presents nanonetworks that solve computations of thresholds, basic arithmetics and distributed consensuses as a proof of concept.

ABSTRACT

Since wet-lab experiments with DNA tend to be very expensive, simulations are used to evaluate the presented models. The simulations show that the conceptualized components fulfill the hypothesized functions.

KURZFASSUNG

Nanogeräte sind winzige, autonome Geräte, welche auf der Nanoebene agieren. Durch ihre geringe Größe sind Nanogeräte starken Ressourcenbeschränkungen unterlegen. Um diese zu umgehen, können sich einzelne Nanogeräte zu Nanonetzwerken zusammenschließen. Kollaborativ können so Problemstellungen bewältigt werden, für die Kapazitäten einzelner Geräte nicht ausreichen.

Nanogeräte und Nanonetzwerke können eingesetzt werden, um Probleme in der Medizin zu lösen, für die zurzeit kein zufriedenstellender Lösungsansatz existiert. Dazu zählt zum Beispiel die Bekämpfung von Krebs. Weitere medizinische Beispiele sind die Bekämpfung von Krankheiten in der Entstehung oder eine lokale Behandlung eines Infektes. In der Medizin gelten beide Probleme als ungelöst, weswegen mit systemischen Behandlungen vorlieb genommen werden muss.

Trotz der vielversprechenden Anwendungsmöglichkeiten gibt es noch zahlreiche ungelöste Probleme in der Nanogeräte- und Nanonetzwerkforschung. Es ist zwar gelungen, primitive Nanogeräte unter Laborbedingungen herzustellen, allerdings stellen diese lediglich einen Machbarkeitsbeweis der Technologie dar.

Eines der größten Probleme ist die Konstruktion aller benötigter Komponenten von Nanonetzwerken. Es kursieren zwar zahlreiche Vorschläge und metasprachliche Maschinenmodelle für besagte Geräte, für diese werden jedoch kaum Spezifizierungen vorgenommen oder Konstruktionskonzepte vorgestellt. Oft beschränken sich Lösungsansätze auf Teilprobleme wie Konstruktion, Kommunikation oder Berechnungen. Die Kompatibilität der Teillösungen ist nicht geklärt. Bislang wurden keine Konzepte vorgestellt, welche sowohl Konstruktion als auch Kommunikation und Berechnungen in einem ganzheitlichen Modell vereinen.

In dieser Arbeit wird für die genannten Problemstellungen ein Lösungsansatz präsentiert. Den Grundbaustein hierfür stellt DNA dar. Die inhärenten Eigenschaften dieses Moleküls können genutzt werden, um komplexe Nanostrukturen zu erschaffen. Auf diese Weise lassen sich sowohl Nanogeräte als auch deren Kommunikationsmechanismen aus DNA erzeugen. Bei korrekter Verwendung kann DNA sogar für Berechnungen eingesetzt werden.

Es wird ein Modell konzipiert, welches alle genannten Probleme mittels DNA lösen kann. Dieses wird im Anschluss zu einer Referenzarchitektur für Nanonetzwerke zusammengefasst. Dadurch wird das vorherrschende Paradigma der Kommunikation und Berechnung in Nanonetzwerken revolutioniert. Berech-

nungen werden in den Zusammensetzungsprozess von Molekülen im Kommunikationskanal ausgelagert.

Im Anschluss werden Nanonetzwerke konzipiert, welche Berechnungen von Schwellwerten, Additionen und verteilten Einigungen in Nanonetzwerken durchführen. Die konzipierten Nanonetzwerke sind als Machbarkeitsbeweis für funktionsfähige Nanonetzwerke auf Basis von DNA zu verstehen.

Da Laborexperimente mit DNA sehr kostspielig sind, kommen Simulationswerkzeuge zum Einsatz. Dabei lässt sich feststellen, dass die konzipierten Komponenten mathematisch sinnvoll und effektiv umsetzbar sind.

VORWORT

Um einem möglichst großen Publikum die Lektüre einer wissenschaftlichen Arbeit zu ermöglichen sollte diese so einfach und voraussetzungslos wie möglich verfasst werden. Das Verständnis eines Textes kann zum Beispiel durch an passenden Stellen verwendete Wiederholungen beschleunigt werden. Einen Experten mag dies zuweilen irritieren, die zusätzlichen Erläuterungen kommen jedoch unbedarften, aber interessierten Lesern zugute.

Für das tiefgreifende Verständnis der folgenden Arbeit sei jedoch vorausgesetzt, dass der Leser Grundlagen der Komplexitätstheorie, sowie Logik und allgemeinen Informatik mitbringt oder willens ist, sich diese mittels der referenzierten Sekundärliteratur anzueignen. Die Wiederholung aller Grundlagen sprengt den Rahmen einer einzelnen Dissertation bei weitem.

Eine Besonderheit der Arbeit ist, dass viele Grundlagen vom Autor und Mitgliedern seiner Forschungsgruppe erarbeitet wurden und zum Eigenanteil der Dissertation maßgeblich beitragen. Außerdem werden in der englischen Sprache definierte Fachbegriffe nicht übersetzt, um Verwirrungen bei der Nachverfolgung von Referenzen zu vermeiden.

Mathematische Vorgehensweisen werden im Verlauf der Arbeit stets so gewählt, dass sie einen praktischen Nutzen liefern. Theoretische Machbarkeitsbeweise sind beispielsweise schön, jedoch oft von geringer praktischer Relevanz. Durch eine konstruktive Vorgehensweise oder Beweisführung ist es hingegen möglich, Ergebnisse effektiv zu nutzen oder Vorgehensweisen abzuleiten, die einen echten Mehrwert bieten.

INHALTSVERZEICHNIS

KAPITEL 1

EINLEITUNG

Dieses Kapitel dient der Schaffung einer Intuition der bearbeiteten Problemstellung. Zu Beginn wird die geschichtliche Entwicklung der Nanotechnologie im Allgemeinen erläutert. Im Anschluss daran wird der Fokus der Arbeit festgelegt. Danach wird die zukünftige Entwicklung abgeschätzt und das vorgestellte Thema aus verschiedenen Blickwinkeln motiviert. Zuletzt werden die eigenen Forschungsanteile beleuchtet und die übrigen Kapitel strukturiert dargestellt.

1.1 EINFÜHRUNG IN DIE NANOTECHNOLOGIE

1960 erklärte Feynman in seinem Vortrag "There is plenty of room at the bottom", dass die bereits fortgeschrittene Miniaturisierung von Technologien noch lange nicht beendet sei [65]. Der vorhergesagte Trend hält bis heute an und ist unter dem Namen "Moore's Law" bekannt geworden [103]. Damit legte Feynman den Grundstein für die Nanoforschung.

Die Nanoforschung ist stark interdisziplinär, und eine enorme Basis an Wissen hat sich über die letzten 50–100 Jahre angesammelt. Großes industrielles Interesse rührt vor allem von der Automobilindustrie, der Medizin und aus Bereichen der Computerchipherstellung her. Erstere ist an neuartigen Technologien, wie zum Beispiel Terahertzantennen für die Abstandsmessung bei Autos, interessiert. Chipdesignern geht es vor allem darum, IT-Produkte noch effizienter produzieren zu können. Die Medizin bietet eine Vielzahl von vorgeschlagenen Anwendungsmöglichkeiten, wie zum Beispiel die Bekämpfung von Krebs durch Nanopartikel, welche später genauer untersucht werden.

Das Interesse geht jedoch über rein industrielle Anwendungen hinaus. Weite Bereiche der Grundlagenforschung in der Physik, Chemie, Biologie und Informatik sind an Nanotechnologien interessiert. Hier geht es unter anderem um die Erforschung von Phänomenen auf Nanoebene und Technologien, welche deren Beobachtung und Manipulation ermöglichen oder erleichtern.

Eine Übersicht und Einordnung des Forschungsbereiches kann durch die in Abbildung 1.1 dargestellte Mindmap erfolgen. Diese stellt wichtige Wissenszweige und Kategorienbegriffe in der Nanoforschung hierarchisch dar. Orange hervorgehobene Bereiche der Mindmap werden in dieser Arbeit behandelt. Je intensiver die Umrandung, desto deutlicher der Fokus. Da eine Analyse des gesamten Forschungsbereiches den Umfang einer einzelnen Dissertation sprengt, fokussiert sich diese Arbeit auf die hervorgehobenen Aspekte.

Die generelle Idee der Arbeit ist es, Geräte und Netzwerke von Geräten auf Nanoebene wie Kristalle wachsen zu lassen. Kristalle sind dabei als eine Gitterstruktur aus einfachen Bausteinen zu verstehen. Auf der Nanoebene gibt es kaum andere Möglichkeiten, Materie im großen Maßstab zu manipulieren.

Kristallbildung basiert auf dem Prinzip der Selbstzusammensetzung. Winzige und oft einfach strukturierte Komponenten setzen sich gemäß lokaler Regeln von alleine zu größeren und oft komplexen Strukturen zusammen. Diese komplexen Strukturen könnten Nanogeräte, Nanosensoren, Nachrichtenpakete, vollständige Nanonetzwerke oder gar Rechenmaschinen sein.

Die meisten bekannten Kristalle folgen zwar dem Paradigma der Selbstzusammensetzung, allerdings kann kaum Einfluss auf den Prozess genommen werden. Eine Ausnahme stellen bestimmte Bausteine aus Desoxyribonukleinsäure – auch DNA genannt – dar. Diese folgen ebenfalls dem Paradigma der Selbstzusammensetzung und können wie Kristalle behandelt werden. Diese Bausteine haben jedoch die Eigenschaft, dass einige DNA-Stränge nur sinnvoll mit seinem Inversen eine Bindung eingehen können.

Es ist möglich, beliebige DNA-Stränge im Labor zu erzeugen. Folglich kann man DNA-Bausteine erschaffen, welche sich wie Puzzleteile verhalten. Diese Puzzleteile aus DNA werden auch Tiles genannt. Diese können sich nur auf vorher festgelegte Weise zusammensetzen und somit eingesetzt werden, um nahezu beliebige Strukturen auf Nanoebene zu erschaffen.

Neben DNA als möglichem Baumaterial existieren weitere Ideen, wie Nanogeräte erschaffen werden können. Beispielsweise gibt es verschiedene Ansätze, welche sich an natürlich vorkommenden Zellen oder Bakterien orientieren. Diese schlagen vor, vorhandene Mittel in der Natur zu nutzen und wenn nötig, auf die eigenen Bedürfnisse anzupassen. Beispiele dafür sind die in den Abbildungen 1.2 und 1.3 dargestellten biologischen Nanogeräte. Die charakteristischen Komponenten der Geräte wurden hier durch Zellorganellen umgesetzt und werden durch Buchstaben gekennzeichnet. Diese werden in Kapitel 2 im Detail erklärt.

Auch künstliche Ansätze sind möglich. Ein Beispiel für einen Nanoroboter auf Basis von klassischen, elektrotechnischen Komponenten kann in Abbildung 1.4 betrachtet werden. Der Ansatz orientiert sich an drahtlosen Sensornetzwerken, wie sie in der Forschung und Teilen der Wirtschaft eingesetzt werden. Es kommen neuartige Materialien, wie zum Beispiel Kohlenstoff-Nanoröhrchen, zum Einsatz, welche ein vielversprechender Baustein für Sensoren, Aktuatoren und Speichertechnologien auf Nanoebene sind [39].

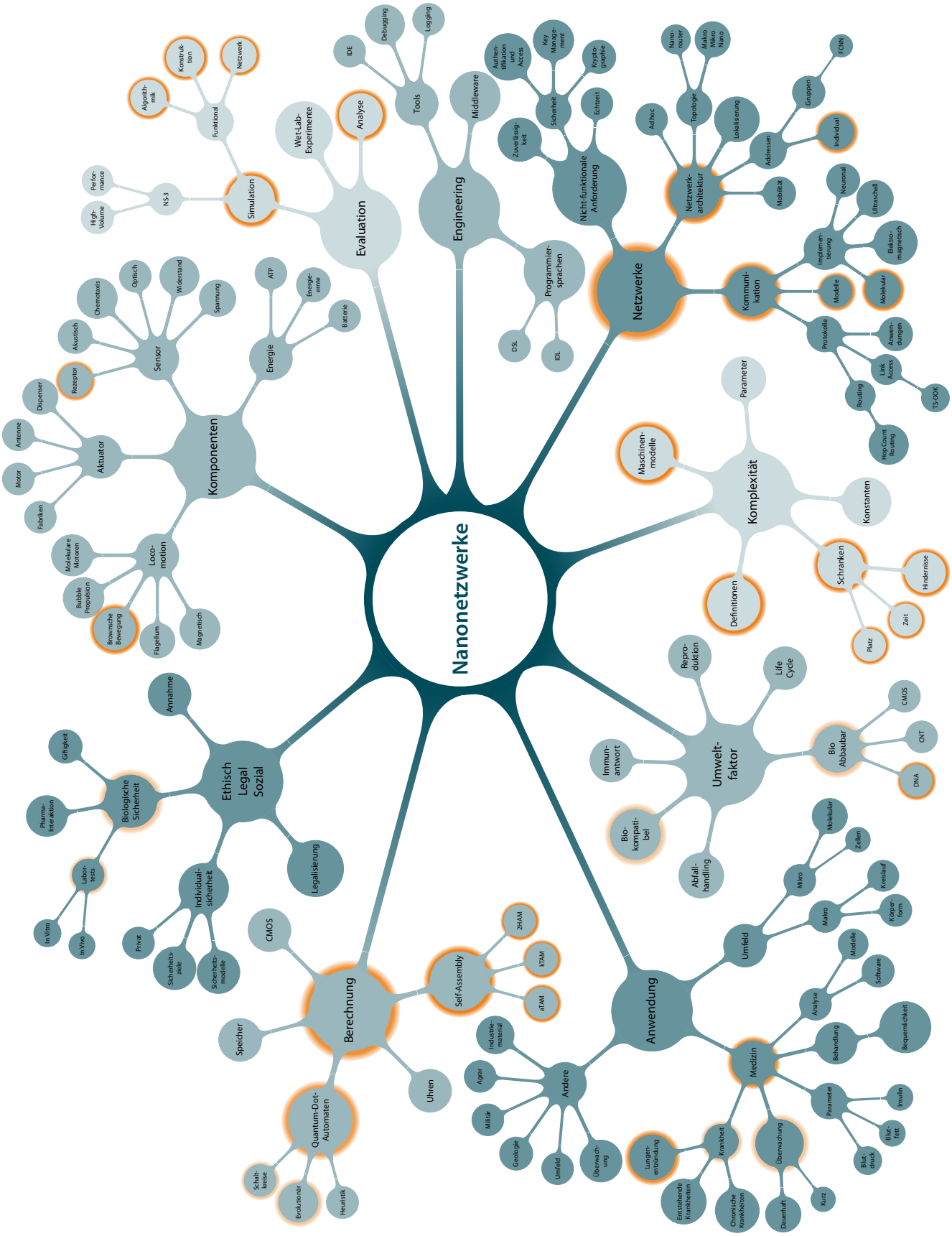


ABBILDUNG 1.1: Mindmap der Nanonetzwerktechnologien. Orange hervorgehobene Knoten deuten behandelte Themen an. Je intensiver die Farbe, desto deutlicher der Fokus.

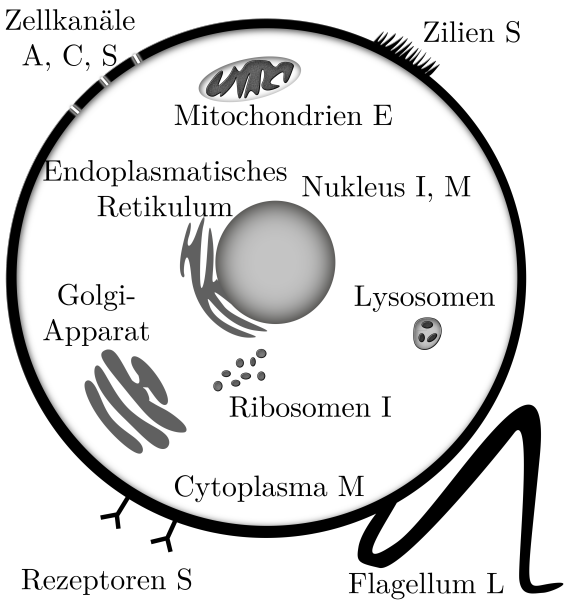


ABBILDUNG 1.2: Schematische Darstellung eines biologischen Nanoroboters. Die Organellen einer Zelle werden genutzt, um Funktionalitäten eines Nanoroboters umzusetzen. [7]

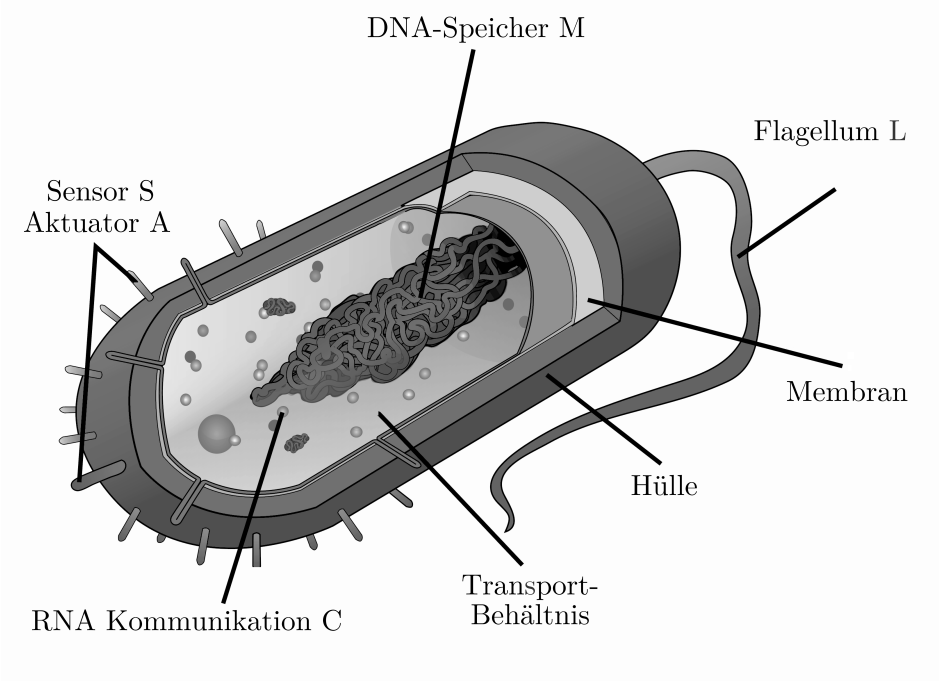


ABBILDUNG 1.3: Schematische Darstellung eines prokaryotischen Nanoroboters. Die einzelnen Organellen werden zu Funktionseinheiten von Nanorobotern umfunktioniert. [4]

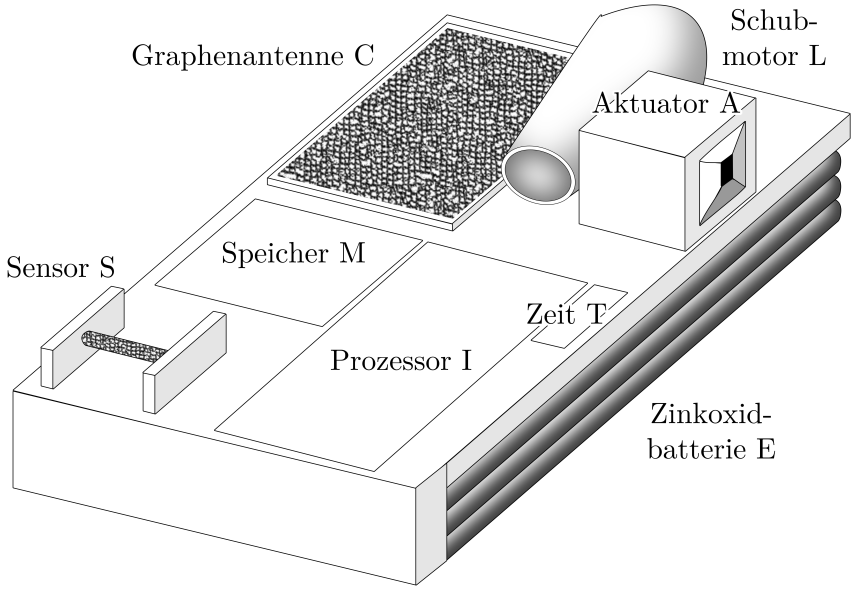


ABBILDUNG 1.4: Schematische Darstellung eines elektrischen Nanoroboters. Die einzelnen Bestandteile eines Nanoroboters werden durch elektrotechnische Komponenten umgesetzt. [4]

Da die übrigen Ansätze zu großen Teilen noch weit von einer physikalischen Implementierung entfernt sind, werden die im Verlauf der Arbeit vorgestellten Nanogeräte auf DNA und dem Paradigma der Selbstzusammensetzung basieren. Alle Bestandteile von Nanogeräten und ihre Kommunikationsmechanismen sowie Recheneinheiten können aus DNA gefertigt werden, ohne dass es weiterer menschlicher Eingriffe bedarf.

Der Vorteil dieser Vorgehensweise besteht darin, dass alle dafür benötigten Materialien bereits im Labor erzeugt werden können, und so erste, rudimentäre Nanonetzwerke im Labor getestet werden könnten.

1.2 ZUKUNFTSPROGNOSE UND MOTIVATION DER NANOTECHNOLOGIEN

Bislang ist es noch nicht gelungen Nanogeräte sinnvoll zu synthetisieren. Da DNA-Technologien jedoch mit großem Interesse weiterentwickelt werden, können Nanogeräteimplementierungen auf Basis von DNA vergleichsweise schnell verfügbar sein [123].

Der folgende Abschnitt prognostiziert die Zukunft der Nanotechnologien. Es werden zwei Möglichkeiten gewählt, sich der Fragestellung zu nähern:

- Extrapolation auf Basis bereits erforschter Technologien und
- die Nachverfolgung von industriellen Interessen und treibenden Kräften.

Der erste Punkt befasst sich mit der theoretischen Realisierbarkeit und der zweite mit dem gebündelten gesellschaftlichen Interesse an besagten Technolo-

Industrie	BIP in Mrd. $	% des BIP	Maslow
Wohnen	1898	13 %	2
Lokale Regierung	1336	9 %	2
Finanzen	1159	8 %	2
Gesundheit	1136	8 %	1
Haltbare Produkte	910	6 %	4
Einzelhandel	905	6 %	4
Großhandel	845	6 %	4
Staatsregierung	658	5 %	2
Daten	646	4 %	4
Unterhaltung	591	4 %	3–5
Bau	529	4 %	2
Entsorgung	448	3 %	2
Andere Dienste	447	3 %	4–5
Dienstleistungen	297	2 %	4–5
Bergbau	290	2 %	4
Management	284	2 %	4–5
Bildung	174	1 %	1–5
Agrarwirtschaft	173	1 %	1

TABELLE 1.1: Industrie der USA anteilig am BIP im Jahre 2011 mit zugeordneter Stufe der Maslowschen Bedürfnishierarchie.

gien. Kombiniert kann abgeleitet werden, warum anzunehmen ist, dass sich das Forschungsgebiet weiterentwickeln wird.

Den Ausgangspunkt bilden dabei die bereits realisierten Technologien. Diese werden im Verlauf der folgenden Kapitel detailliert analysiert. Zusammenfassend lässt sich sagen, dass primitive Annäherungen an Nanogeräte herstellbar sind. Es handelt sich lediglich um Machbarkeitsbeweise ohne Anwendungsfälle in der Realität.

Treibende Kräfte hingegen bilden die Grundlage für die Geschwindigkeit, mit der sich Neuerungen weiterentwickeln. Ausgangspunkt für eine Analyse auf deren Basis ist das kapitalistische Gesellschaftssystem, in dem für eine Nachfrage zeitnah eine Antwort in Form eines Produktes geliefert wird, welches sich schrittweise im Zuge von Konkurrenzprozessen verbessert.

Analysiert man nun den finanziellen Wert verschiedener Industrien und vergleicht ihn, kann man ableiten, an welchen Produkten ein großes gesellschaftliches Interesse besteht. Tabelle 1.1 zeigt ein Beispiel der US-amerikanischen Wirtschaft im Jahr 2011 [174]. Es ist zu erkennen, dass Wirtschaftszweige, die mit einer niedrigen Schicht der Maslowschen Bedürfnishierarchie korrelieren [100], einen großen finanziellen Anteil einnehmen. Abbildung 1.5 zeigt die Hierarchie. Besonders grundlegende Bedürfnisse sind weit unten dargestellt. Treten dort Probleme auf, gibt es ein großes gesellschaftliches Interesse Lösungen zu finden.

Wendet man die Maslowsche Bedürfnishierarchie auf die Wirtschaftsstatistik an,

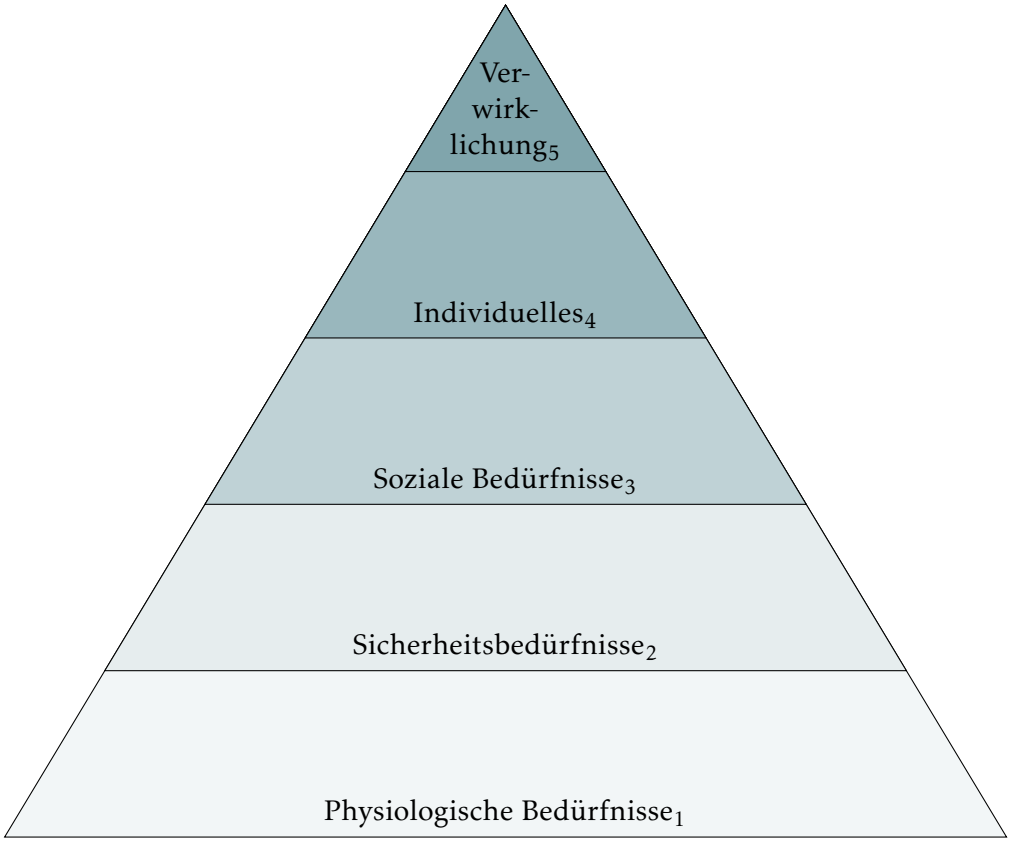


ABBILDUNG 1.5: Maslowsche Bedürfnishierarchie. Weiter unten dargestellte Bedürfnisse müssen gestillt sein, bevor übergeordnete Bedürfnisse auftreten.

so ergeben sich grobe Kategorien, welche ein besonders starkes gesellschaftliches Interesse erwecken. Ein besonders großes Interesse bietet Raum und Mittel für intensive Forschung an neuartigen Technologien, die zurzeit nicht lösbare Probleme untersuchen.

Es ist zu erkennen, dass vor allem das Gesundheitswesen von Interesse ist, da es Stufe 1 der Maslowschen Bedürfnishierarchie anspricht und ein großer Teil des BIP investiert wird. Zudem wird die Medizin oft als potentielles Anwendungsgebiet für Nanotechnologien vorgeschlagen.

Im Folgenden liegt der Fokus folglich auf medizinischen Anwendungsfällen.

1.2.1 MEDIZINISCHE ANWENDUNGSFÄLLE

Die Science-Fiction-Literatur hat einen nicht zu vernachlässigenden Einfluss auf die Ideenbildung im Bereich der Nanotechnologien und hat diese maßgeblich mit motiviert. So schrieb Arthur C. Clarke in "The Next Tenants" bereits 1956 von winzigen Robotern, welche den heutigen Darstellungen von medizinischen Nanorobotern nicht unähnlich sind [14]. Damit war er Feynman um vier Jahre voraus. Der polnische Autor Stanislaw Lem griff das Konzept ebenfalls in seiner Geschichte "The Invincible" 1964 auf [19]. 1984 schrieb er von bakteriellen

Nanorobotern in "Peace on Earth", welche von einer künstlichen Intelligenz entwickelt wurden [18].

Auch wenn die Wissenschaft noch nicht den Stand der Science-Fiction-Literatur erreicht hat, sind doch beachtliche Fortschritte gemacht worden. Heutzutage werden bereits regelmäßig Nanopartikel eingesetzt [137, 129, 95] – zum Beispiel in der Krebsforschung [124]. Einige Nanopartikel sind so strukturiert, dass sie nur an Krebsgewebe binden und dort durch äußere Einwirkung erhitzt werden. Dies kann zum Beispiel durch Laser geschehen. Dabei wird das betroffene Gewebe zerstört und die umliegenden, gesunden Zellen können sich erholen. Visionäre Vorgehensweisen planen den Gebrauch von Nanogeräten im menschlichen Körper, um Krebszellen besser markieren oder direkt zerstören zu können.

Diese Vorgehensweise kann leicht generalisiert werden. Nanogeräte können allgemein bei der Erkennung und Behandlung von Krankheiten eine sinnvolle, unterstützende Maßnahme darstellen. Dies schließt unter anderem die Behandlung von Diabetes [71], die Erkennung und örtliche Behandlung von entzündlichen Erkrankungen [125, 165, 7] und viele weitere Diagnose- und Therapiemöglichkeiten ein [41]. Oft wird vorgeschlagen, dass Nanogeräte im Verbund arbeiten können, da sie aufgrund ihrer Größe erheblichen Einschränkungen unterliegen [26].

Komplexe Szenarien schildern den Gebrauch von molekularen Automaten, welche im Labor bereits in der Lage sind, mehrere Marker für eine Krankheit zu erkennen und mit der Produktion spezieller RNA-Sequenzen zu beginnen, sobald alle Marker anwesend sind [35]. Die produzierten Sequenzen erfüllen momentan noch keinen Zweck, zeigen aber prinzipiell, dass auf Umgebungsparameter regiert werden kann – auch konstruktiv.

In [70] wird die Erkennung von Krankheiten weiter abstrahiert. Unter anderem werden Anwendungen wie intra-zelluläre Operationen vorgeschlagen. Solche Technologien erfordern ein hohes Maß an Präzision [66].

Ein weiterer, beliebter Anwendungsfall für Nanogeräte ist die Unterstützung des menschlichen Immunsystems. Oft kommt es vor, dass die Selbstheilungskräfte des Menschen für gewisse Krankheitsbilder nicht mehr ausreichen. Hat ein Patient ein hohes Alter erreicht, so dauern Regenerationsprozesse oft deutlich länger als bei jungen Menschen. Genauso ist eine generelle Verbesserung des Immunsystems, unabhängig von der Ausgangslage denkbar [66, 26, 71].

Drei Szenarien sind aus medizinischer Sicht von gesondertem Interesse. Dabei handelt es sich um Problemstellungen, welche sich mit den momentan verfügbaren Technologien nicht oder nur eingeschränkt lösen lassen, weswegen Nanotechnologien zum Einsatz kommen könnten:

1. Die erschwingliche Dauerüberwachung von Gesundheitsparametern bei Hochrisikogruppen unter Beibehaltung der Lebensqualität: Eine gesundheitlich angeschlagene Person könnte so beispielsweise medizinische Nanoroboter verabreicht bekommen, welche relevante Gesundheitsparameter überwachen. Auf diese Weise könnte die Person weiterhin ihrem geregelten

ABBILDUNG 1.6: Storyboard-Darstellung einer medizinischen Dauerüberwachung unter Zuhilfenahme von Nanogeräten [1].

Alltag nachgehen, ohne auf eine stationäre Überwachung angewiesen zu sein.

2. Die unmittelbare Bekämpfung von Krankheiten bei der Entstehung, ohne dass es zu einem Ausbruch von Symptomen kommt: Viele Krankheiten könnten auf diese Weise deutlich effizienter bekämpft oder verhindert werden. Kann beispielsweise die exponentielle Vermehrung von Bakterien bereits bei wenigen Exemplaren entdeckt und gestoppt werden, sind kaum schwerwiegende Eingriffe notwendig.
3. Die lokale Behandlung von Krankheiten ohne systemische Zuführung von Medikamenten durch gezielte, örtliche Verabreichung. Eines der größten ungelösten Probleme der modernen Medizin ist die zunehmende Resistenz von Erregern beispielsweise gegen Antibiotika. Diese können reduziert werden, indem Medikamente nur dort eingesetzt wird, wo sie wirklich nötig sind. Bei einer bakteriellen Lungenentzündung entspräche das zum Beispiel einer lokalen Behandlung der Lunge durch Nanoroboter.

Der in der Fachliteratur am häufigsten erwähnte Anwendungsfall ist die konstante Überwachung der menschlichen Gesundheit [27]. Abbildung 1.6 verdeutlicht den Prozess anhand eines Beispiels. Bei einer Behandlung werden Nanogeräte verabreicht, welche fortan Gesundheitsparameter messen und kontinuierlich an eine externe Software übermitteln.

Momentan ist eine solche Überwachung kaum möglich oder mit enormer Anstrengung und Entbehrungen verbunden. Bluttests werden beispielsweise nur stichprobenartig unternommen – meist erst, sobald ein begründeter Verdacht besteht. Viele Krankheiten lassen sich auf diese Weise jedoch kaum oder zu spät erkennen [66].

Im Folgenden werden weitere mögliche Anwendungen für Nanotechnologien vorgeschlagen, welche nicht direkt mit der Medizin in Verbindung stehen.

1.2.2 WEITERE ANWENDUNGSFÄLLE

Medizinische Problemstellungen stehen im Mittelpunkt der Forschung, da ein begründetes öffentliches Interesse an ihnen besteht. Aber auch andere Industriezweige und Interessengruppen zeigen ein nicht zu vernachlässigendes Interesse an Nanotechnologien.

- Die Agrarindustrie hat Interesse an weiterer Effizienzsteigerung. Der Thematik wurde jüngst ein ganzes Buch gewidmet [13]. Im Hinblick auf das ungebremste Bevölkerungswachstum in Afrika und Teilen Asiens wird es immer wichtiger, dass die vorhandenen Ackerflächen so effizient wie möglich genutzt werden, damit alle Menschen ernährt werden können. Aber auch Luxusprobleme treten auf. Je reicher ein Land wird, desto höher wird der Fleischkonsum. Die Produktion von tierischer Biomasse ist um einen Faktor ineffizienter als die Produktion von pflanzlicher Biomasse. Zurzeit wird hauptsächlich an genveränderten Lebensmitteln geforscht – in der Hoffnung, dass diese Technologie eine ausreichende Effizienzsteigerung bewirken kann.

  Nanogeräte könnten darüber hinaus unterstützend eingesetzt werden – zum Beispiel bei der Schädlingsbekämpfung oder als Katalysator für die Freisetzung von im Mutterboden gebundenen Nährstoffen. Zusätzlich ist es denkbar, dass Nanogeräte eingesetzt werden können, um benötigte Nährstoffe an die Versorgungsorgane der Nutzpflanzen zu befördern. Dadurch kann eine Mitnutzung durch andere Organismen möglicherweise verhindert werden. Zuletzt ist es denkbar, dass Nanotechnologien genutzt werden können, um einen positiven Einfluss auf die Sensor- und Kommunikationskanäle, wie zum Beispiel Rhizom- und Fungusnetzwerke, zu nehmen [141].
- Geologische Anwendungsfälle sind ebenfalls von Interesse. Dabei handelt es sich in erster Linie um Forschungsfragen und Frühwarnsysteme zum Beispiel für Naturkatastrophen. Nanotechnologien könnten an Orte gelangen, welche für makroskopische Technologien kaum zugänglich sind. Dieses Prinzip lässt sich ebenfalls auf die Erkundung und Vermessung von unterirdischen Ölfeldern übertragen. Der Einsatz von Nanogeräten könnte hier eine Aussage über die Wirtschaftlichkeit einer Förderung tätigen, welche durch herkömmliche Methoden nicht eingeschätzt werden kann [82].
- Das Militär hat ein begründetes Interesse an Überwachungstechnologien, welches die Entwicklung im Bereich der Nanotechnologien beschleunigen könnte. Der Rüstungsindustrie steht durch staatliche Maßnahmen ein enormes Forschungsbudget zur Verfügung und sie ist immer daran interessiert Kontrahenten einen Schritt voraus zu sein.
- Materialwissenschaftler sind an der Erforschung neuartiger Baustoffe interessiert, welche sich z. B. auf Self-Assembly-Prozesse oder Kristallbildung stützen könnten. Einige Veröffentlichungen schlagen vor, dass die Zusammensetzung von Materialien auf Nanoebene von DNA geleitet werden könnte [152]. Besonders bei sogenannten Metamaterialien könnte DNA als Platzierungshilfe für nanogroße Komponenten verwendet werden, damit diese die Oberflächeneigenschaften des Materials präzise beeinflussen können [159].

1.3 PROBLEMBESCHREIBUNG

In der Nanogeräteforschung gibt es drei übergeordnete Problemstellungen, welche ein Forschungsinteresse begründen. Dabei handelt es sich um:

- Konstruktion von nanogroßen Strukturen und Geräten,

- Berechnungen unter Berücksichtigung der Beschränkungen auf Nanoebene und
- Kommunikation zwischen einzelnen Nanogeräten.

Für die Teilprobleme existieren Lösungsansätze auf Basis verschiedenster Prinzipien. Allerdings werden diese weitestgehend unabhängig voneinander untersucht und selten auf Kompatibilität geprüft.

Bislang ist es beispielsweise gelungen, primitive Nanogeräte unter Laborbedingungen herzustellen. Diese können jedoch bestenfalls als Machbarkeitsbeweis der Technologie im Allgemeinen verstanden werden. Es ist gänzlich unklar, wie diese miteinander kommunizieren oder Berechnungen durchführen können [79].

Im Umkehrschluss gibt es Ansätze für die Kommunikation unter Geräten, welche allerdings mit den momentan verfügbaren Nanostrukturen nicht umsetzbar erscheinen, da die nötigen Komponenten zur Informationsverarbeitung noch nicht kompatibel sind [26].

Berechnungen sind auf Nanoebene zwar umsetzbar, allerdings existieren keine realistischen Konzepte, wie diese innerhalb von Nanogeräten, autark stattfinden können [179].

Bislang wurde, nach ausgiebiger Suche, kein Konzept gefunden, welches sowohl Konstruktion als auch Kommunikation und Berechnungen in einem ganzheitlichen Modell vereint.

1.4 FOKUS DER ARBEIT

Diese Dissertation nimmt sich der genannten Problemstellung an. Es wird ein ganzheitliches Modell für Nanonetzwerke auf Basis von DNA vorgestellt. Dieses schließt sowohl die Konstruktion von Nanogeräten, die Kommunikation untereinander als auch ein Konzept für Berechnungen mit ein. Alle Komponenten im Fokus der Arbeit basieren auf DNA als Baustoff. Da die Interaktion der Komponenten auf DNA-Bindungsreaktionen basiert, sind diese nahezu beliebig miteinander kombinierbar. Die folgenden wissenschaftlichen Beiträge stellen die jeweiligen Schwerpunkte der Arbeit aufgelistet dar.

1.4.1 WISSENSCHAFTLICHE BEITRÄGE DER ARBEIT

Folgende Probleme wurden in Publikationen, welche mit der Dissertation in Verbindung stehenden vorgestellt oder werden im Verlauf der Dissertation bearbeitet:

- Das mathematische Modell der Tile-basierten Self-Assembly-Systeme wurde vom Autor für die Nanonetzwerkgemeinschaft nutzbar gemacht [6, 1].
- Es wird eine formale Definition von Nanostrukturen, Nanogeräten, Nanosensoren, Nanomaschinen, Nanoknoten, Nanorobotern und Nanonetzwerken vorgestellt [4].

- Es werden medizinische Szenarien auf die Einsetzbarkeit von Nanorobotern und Nanonetzwerken untersucht. Dabei wird überprüft, was die Minimalanforderungen an Maschinenmodelle sind, damit sinnvoll Probleme gelöst werden können. Auf Basis dieser Untersuchung werden verschiedene Komplexitätsklassen vorgestellt und verwendet, um besonders leistungsschwache Geräte zu beschreiben und zu vergleichen [7, 5].
- Es werden DNA-basierte Self-Assembly Systeme für den Kontext von Nanonetzwerken als Konstruktionsmechanismus und Berechnungsmodell etabliert [1, 2, 6, 4]. Diese dienen als Grundlage für Simulationen und Evaluationen.
- Es wird das vorherrschende Paradigma für Kommunikation und Berechnungen in Nanonetzwerken grundlegend erweitert, indem Berechnungen in den Übertragungskanal / Nachrichtenmoleküle ausgelagert werden, anstatt "in" Nanogeräten stattzufinden. Dies verringert den Einfluss von oft festgestellten Platzbeschränkungen [1]. Aufbauend darauf werden für die Lösung zahlreicher mathematischer Operationen Nachrichtenmoleküle konzipiert. Es wird zudem ein allgemeines Verfahren für die Erzeugung von Nachrichtenmolekülen aus aussagenlogischen Formeln bewiesen.
- Es werden Algorithmen zur Zusammensetzung von dreidimensionalen Strukturen vorgestellt, welche als Ausgangspunkt für die Konstruktion von Nanogeräten verwendet werden können [2, 11, 12].
- Es werden Techniken vorgestellt, mit denen die fehlerhafte Natur von Tile-Assembly-Prozessen in drei Dimensionen kompensiert werden kann und eine fehlertolerante Skalierung der Größe von Nanostrukturen möglich wird [3, 8].
- Es wird eine Referenzarchitektur für medizinische Nanonetzwerke aus DNA-Tiles vorgestellt [1]. Diese vereint Nanogeräte auf aus DNA, Nachrichtenmoleküle auf Basis von DNA und Berechnungen, welche in den Zusammensetzungsprozess von besagten Molekülen integriert werden.
- Diese Arbeit stellt erstmals ein Konzept vor, welches vollwertige Nanonetzwerke unter Verwendung von verfügbaren Technologien beschreibt. Seit Beginn des Forschungszweiges im Jahre 2008 ist dies der erste Lösungsansatz [1].

1.5 STRUKTUR DER DISSERTATION

Kapitel 2 stellt den aktuellen Stand der Technik der verwendeten Technologien dar. Im Anschluss daran werden mathematische und biologische Grundlagen sowie geeignete Maschinenmodelle definiert, welche später umgesetzt werden. Es erfolgt eine Analyse der Mächtigkeit von Berechnungsmodellen. Es werden passende Komplexitätsklassen erläutert, welche ressourcenbeschränkte Nanogeräte beschreiben. Es werden die Grundlagen für Self-Assembly-Systeme beschrieben und begründet, warum diese im Verlauf der Dissertation Verwendung finden.

Kapitel 3 beschäftigt sich mit den verschiedenen Arten von möglichem Fehlverhalten von Self-Assembly Systemen. Zuerst werden Growth-Errors, Facet-Errors und Nucleation-Errors formal definiert. Im Anschluss daran werden verschiede-

ne Mechanismen zur Fehlervermeidung vorgestellt. Zuletzt wird ein Algorithmus präsentiert und verifiziert, welcher sowohl Facet- als auch Growth-Errors in dreidimensionalen Self-Assembly-Systemen stark reduziert.

Kapitel 4 analysiert die Fachliteratur im Bereich Nanomedizin auf Anwendungsszenarien für Nanonetzwerke. Es werden mathematische Probleme extrahiert und entsprechend ihrer Schwierigkeit sortiert. Des Weiteren werden bestehende Simulationswerkzeuge gesichtet und analysiert sowie auf ihre Tauglichkeit zur Evaluation der präsentierten Modelle geprüft.

Kapitel 5 definiert Nanonetzwerke sowie Referenzarchitekturen für Nanonetzwerke. Es zeigt Algorithmen für Tile-basierte Self-Assembly-Systeme, welche Nanostrukturen und Nanonetzwerke entstehen lassen. Weiterhin werden Tilesets für Nachrichtenmoleküle, welche Berechnungen durchführen, vorgestellt. Es wird ein Vorgehen bewiesen, mit dem sich beliebige mathematische Berechnungsprobleme durch DNA-Tile-basierte Nanonetzwerke darstellen und lösen lassen. Zum Schluss werden exemplarisch Nanonetzwerke zur Lösung prominenter Problemstellungen entworfen.

Kapitel 6 überprüft die vorgestellten Ansätze und Algorithmen durch Simulationen auf Korrektheit und Effizienz. Dabei wird die Software ISU TAS verwendet. Es werden Simulationen in den drei Modellen abstract Tile Assembly Modell , kinetic Tile Assembly Modell und twohanded Tile Assembly Modell durchgeführt, um möglichst viele Aspekte der Realität durch Simulationen abzubilden. Die Modelle werden mit den Mitteln der Komplexitätstheorie analysiert und kategorisiert. Dabei werden sowohl Laufzeit als auch Tile-Komplexität dargestellt. Zuletzt werden statistische Erhebungen in Histogrammen präsentiert und die Evaluationsergebnisse bewertet.

Kapitel 7 fasst die Ergebnisse der Arbeit zusammen und vergleicht diese mit der Zielsetzung. Es werden die wichtigsten offenen Fragestellungen präsentiert und weiterführende Forschungsthemen motiviert.

KAPITEL 2

GRUNDLAGEN

DIESES Kapitel[1] befasst sich mit den Grundlagen und dem Stand der Technik von Berechnungsmodellen sowie einer Einführung in die wichtigsten mathematischen und biologischen Grundlagen der Arbeit.

In Abschnitt 2.1 erfolgt zunächst eine geschichtliche Einordnung der Entdeckung und Erforschung der DNA. Im Anschluss werden die biologischen Grundlagen in Abschnitt 2.2 vermittelt. Danach wird in Abschnitt 2.3 eine formale Definition für verschiedene Geräte und Strukturen auf Nanoebene vorgenommen. Auf Grundlage derer werden Maschinenmodelle spezifiziert und ihre individuellen Rechenleistungen untersucht. Dies geschieht basierend auf der in Abschnitt 2.4 vorgestellten, komplexitätstheoretischen Grundlagen. Dort wird das nötige Wissen vermittelt, um die mathematischen Problemstellungen formal zu verstehen und kategorisieren zu können. In Abschnitt 2.5 werden verschiedene Berechnungsmodelle vorgestellt und auf ihre Anwendbarkeit auf Nanoebene untersucht. Es werden Schaltkreise mit Quantum-Dot zellulären Automaten , chemischen Reaktionsnetzwerken und Self-Assembly-Systemen verglichen. Auf Basis der Gegenüberstellung wird das am passendsten erscheinende Berechnungsmodell ermittelt und begründet.

2.1 HISTORISCHE ENTWICKLUNG UND ENTDECKUNG VON DNA-BAUSTEINEN

Dieser Abschnitt verdeutlicht die historische Entwicklung von DNA-basierten Technologien. Abbildung 2.1 zeigt einen Überblick über die wichtigsten Ereignisse, welche einen Einfluss auf Self-Assembly-Systeme hatten mit Referenzen.

Bereits einige Jahre nach der Entdeckung der DNA (siehe Abschnitt 2.2.1) durch Watson und Crick im Jahre 1953 [132] wurde sie als potentielles Baumaterial auf Nanoebene vorgeschlagen.

[1] Teile dieses Kapitels wurden bereits in [4, 1, 3] vorgestellt.

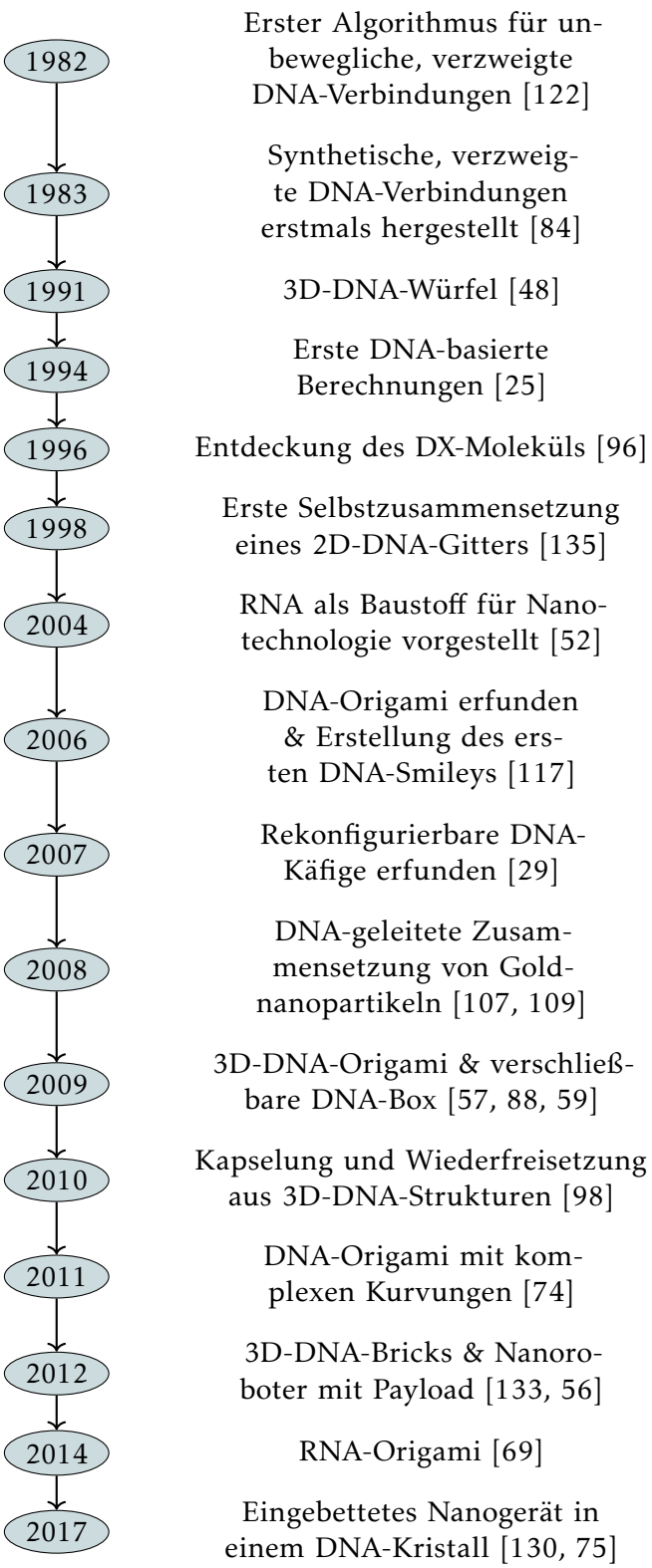


ABBILDUNG 2.1: Zeitstrahl der wichtigsten DNA-basierten Nanotechnologien und Entdeckungen [123].

1982–1983 kam zum ersten Mal die Idee auf, (künstliche) verzweigte DNA-Verbindungen zu verwenden. Acht Jahre später wurde ein Würfel aus DNA synthetisiert. Diese Schritte stellen die wichtigste Grundlage für dreidimensionale Self-Assembly-Systeme und Transportmechanismen dar. Im Jahre 1996 wurde das DX-Molekül vorgestellt, welches die Basis für moderne Self-Assembly-Systeme ist. Tiles sind die Grundbausteine für Self-Assembly-Systeme und werden auf Basis dieses Moleküls oder Variationen davon erstellt.

1998 gelang es Winfree erstmals ein zweidimensionales Gitter aus DNA wachsen zu lassen. Sechs Jahre später wurde erstmals *Ribonukleinsäure* (RNA) als Baustoff für Nanotechnologien vorgeschlagen. Anders als DNA, besteht RNA aus einem einzelnen Strang mit der Base Uracil anstelle von Thymin. 2006 wurde die DNA-Origami-Technologie vorgestellt und Rothemund nutze diese, um das erste nano-Smiley Face zusammenzusetzen. Ein Jahr später gelang die DNA-geleitete Zusammensetzung von Goldnanopartikeln. Diese spielen eine wichtige Rolle in der Bekämpfung von Krebs und anderen Krankheiten. Ein weiteres Jahr später wurde der DNA-Origami Prozess auf drei Dimensionen verallgemeinert. Außerdem wurde eine DNA-Box erschaffen, welche durch ein Signal geöffnet werden konnte. Diese beiden Meilensteine motivieren den dreidimensionalen Tile-Ansatz in dieser Arbeit weiter und stellen einen Proof-of-Concept dar. Im Jahre 2010 wurde die DNA-Box weiter verfeinert. Es gelang erstmals Nanopartikel in eine 3D-DNA-Struktur zu laden und später wieder freizusetzen. Diese Idee bildet die Grundlage für die lokale Anwendung von Medikamenten mittels Nanogeräten.

2011 gelang die Erstellung von DNA-Origami mit komplexen Kurvungen. 2012 wurden DNA "Backsteine" vorgestellt, welche als Baustoff für komplexere Strukturen dienen können. Außerdem wurde die Medikamentenapplikation durch Nanogeräte weiter verbessert. Zwei Jahre später wurde der Origami-Ansatz auf RNA erweitert und 2017 gelang erstmals die Entwicklung komplizierter Nanogeräte. Es wurden Halbleiter und Nanogeräte in dreidimensionale DNA-Kristalle eingebettet. Letzteres Ergebnis zeigt, dass es möglich ist DNA zu verwenden, um andere Strukturen darin zu transportieren.

Die relevantesten biologischen Grundlagen werden im folgenden Abschnitt im Detail erläutert.

2.2 BIOLOGISCHE GRUNDLAGEN

Dieser Abschnitt erläutert die biologischen Bestandteile von Self-Assembly Systemen. Dabei wird zuerst auf die Wirkungsweisen des elementaren Baustoffes DNA eingegangen. Im Anschluss daran werden Techniken vorgestellt, mit denen spezielle Moleküle, auch *Tiles* genannt, erschaffen werden können. Diese bilden die Basis für viele der später vorgestellten Modellierungen.

2.2.1 DESOXYRIBONUKLEINSÄURE

1953 entdeckten Watson und Crick den genetischen Code, welcher die Erbinformation aller Lebewesen der Erde enthält: *Desoxyribonukleinsäure* (DNA)

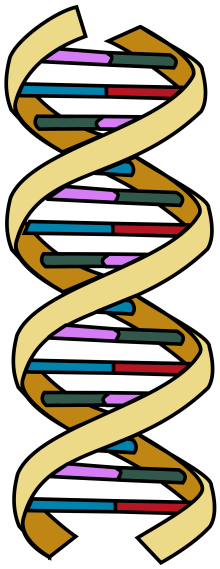


ABBILDUNG 2.2: Darstellung der DNA-Doppelhelix.

[132]. Auf nur zwei Seiten wurden die revolutionären Ergebnisse der beiden Wissenschaftler vorgestellt. Sie erkannten die Doppelhelixstruktur der DNA und bemerkten sofort den potentiellen Vervielfältigungsmechanismus, welcher der Redundanz des Doppelstrangs innewohnte. Anders als bei DNA ist diese Eigenschaft bei RNA nicht gegeben. Es ist somit unklar, ob Tiles aus RNA angefertigt werden können.

Abbildung 2.2 zeigt den Aufbau von DNA schematisch. Die dargestellte Doppelhelix besteht aus einem Phosphat-Desoxyribose-Rückgrat (gelb) und verschiedenen Basen, welche die beiden Stränge miteinander verbinden. Die Verbindungsglieder werden Thymin (violett), Adenin (grün), Guanin (blau) und Cytosin (rot) genannt. Die Besonderheit der vier Basen ist, dass sie ausschließlich in Paaren in der DNA-Struktur auftreten. Thymin bindet nur mit Adenin und Guanin bindet nur mit Cytosin. Durch diese Bedingung wird sichergestellt, dass die jeweiligen Stränge jeweils das Komplement ihres Nachbarn sind – ein Mechanismus, welcher zur Vervielfältigung genutzt werden kann.

2.2.1.1 POLYMERASEKETTENREAKTION

Die *Polymerasekettenreaktion* (PCR) ist eines der wichtigsten Werkzeuge bei der künstlichen Vervielfältigung von DNA-Molekülen. Lange Zeit war bekannt, dass DNA in lebenden Organismen vervielfältigt wird, allerdings gelang es erst mit der Entdeckung der Polymerasekettenreaktion durch Kary Mullis, den Prozess effizient außerhalb eines Organismus stattfinden zu lassen [104]. Die Polymerasekettenreaktion erfolgt in drei Schritten, welche beliebig oft wiederholt werden können:

1. Denaturierung,
2. Primerhybridisierung und
3. Elongation.

Alle Komponenten, die für die Vervielfältigung von DNA benötigt werden, befinden sich in einem gemeinsamen Medium. Dies schließt freie Nukleotide (die vier Basen), Primer und Polymerasen mit ein. *Primer* sind kurze DNA-Stränge, welche an Teile des zu replizierenden DNA-Stranges binden. Sie ermöglichen der Polymerase, ihre Arbeit aufzunehmen. *Polymerase* ist ein Enzym, das die Entstehung eines Komplementärstranges zu einem Einzelstrang katalysiert.

In der Denaturierungsphase wird das zu vervielfältigende DNA-Segment auf eine hohe Temperatur erhitzt. 94 °C bis 96 °C sind typische Werte. Bei hohen Temperaturen werden die Bindungen der Basenpaare instabil und die Stränge lösen sich voneinander. Das Phosphat-Desoxyribose-Rückgrat hingegen bindet stärker und ist selbst bei 96 °C noch stabil.

Es folgt die Primerhybridisierung. Hier wird die Temperatur des Mediums, in dem sich alle Komponenten befinden, soweit gesenkt, dass sich die Primer an der zu vervielfältigenden DNA festsetzen können. Die Primer können aufgrund ihres Aufbaus aus Basen nur an bestimmten Orten der zu replizierenden DNA haften.

Sobald die Primer fest gebunden sind, beginnt die Elongationsphase. Hier setzt die Polymerase an den Primern an und beginnt mit dem Bau eines Komplementärstranges. Damit die Polymerase bei der Denaturierung nicht ebenfalls zerstört wird, werden Polymerasen aus dem Bakterium Thermus aquaticus verwendet. Dieses kommt natürlich in Geysiren vor und ist besonders hitzebeständig [112]. Entsprechend der Länge des zu replizierenden Stranges werden ca. 30 Sekunden für 500 Basenpaare benötigt.

Diese drei Phasen werden so lange wiederholt, bis die gewünschte Menge DNA-Stränge erzeugt wurde. Nach jedem Zyklus verdoppelt sich die Anzahl der DNA-Segmente. Durch das exponentielle Wachstum kann in kurzer Zeit eine enorme Menge an DNA synthetisiert werden. Die PCR wird für die Herstellung der Grundbausteine (Tiles) von Self-Assembly-Systemen verwendet.

2.2.2 DNA-TILES

DNA-Tiles beschreiben eine Klasse von Molekülen, welche besondere Eigenschaften erfüllen. Sie bestehen aus speziellen DNA-Molekülen, deren Doppelstränge miteinander verwoben wurden, um so einen stabilen "Kristall" zu erschaffen. Diese Kristalle verfügen im Gegensatz zu herkömmlicher DNA über mehrere offene Enden in unterschiedlichen Richtungen. Offene Enden ermöglichen eine Bindungsinteraktion mit anderen Tiles, welche über entsprechende offene Enden mit Komplementärsträngen verfügen.

Das Bindungsverhalten von DNA-Tiles lässt sich mit Puzzleteilen vergleichen. Zwei Puzzleteile passen nur, wenn sie entsprechende, genau ineinander passende Aussparungen haben. Der einzige Unterschied besteht darin, dass DNA-Tiles sich selbst zusammensetzen, wenn sie zueinander passen.

ABBILDUNG 2.3: Schematische Repräsentation eines DX-Tiles. [170]

Es gibt zahlreiche Variationen von DNA-Tiles. Diese sind im zweidimensionalen Fall ca. 2 nm hoch. Länge und Breite können nach Belieben gewählt werden [97].

Im Folgenden werden relevante Exemplare genauer untersucht.

2.2.2.1 DX-TILE

Das *Double-Crossover-Tile* (DX-Tile) wurde erstmals von Seeman und Tsu Ju Fu im Jahre 1993 vorgestellt [67]. Abbildung 2.3 zeigt eine schematische Darstellung. Zu sehen sind zwei Doppelstränge, welche miteinander verwoben wurden. Die äußeren Teile der Doppelstränge sind jeweils ozeangrün oder rot dargestellt. Die inneren Hälften sind schwarz und blau. Teile der Doppelstränge gehen in den jeweils anderen über, sodass ein stabiler Zusammenschluss entsteht, welcher auch bei Temperaturen von ca. 70 °C nicht denaturiert.

Dadurch, dass mehrere offene Enden an der linken und rechten Seite eines jeden Tiles existieren, können sich eine Vielzahl von Tiles zu einem großen Kristall zusammenschließen, um so beispielsweise die zweidimensionale Ebene teilweise zu füllen [135]. Für komplexere Interaktionen oder Berechnungen sind jedoch Tiles mit offenen Enden in mehr als zwei Richtungen nötig.

2.2.2.2 TX-TILE

Das sogenannte *Triple-Crossover-Tile* (TX-Tile) wurde von Winfree et al. entdeckt [91] (siehe Abbildung 2.4). Es stellt eine Erweiterung des vorgestellten DX-Tiles dar. Das TX-Tile entsteht durch Verwebung dreier Doppelstränge, sodass ein sehr stabiles DNA-Molekül entsteht. Die offenen Enden (*Kleber*) eines TX-Tiles haben einen größeren Abstand voneinander und erlauben somit die Konstruktion von anderen Formen.

2.2.2.3 HOLLIDAY-JUNCTION

Robin Holliday hat die Existenz der nach ihm benannten *Four-Arm-Junctions* oder *Holliday-Junctions* bereits 1963 vorhergesagt [76]. Obwohl diese Art Molekül natürlich im menschlichen Körper vorkommt, gelang es Nadrian C. Seeman erst zwei Jahrzehnte später diese unter Laborbedingungen herzustellen [84].

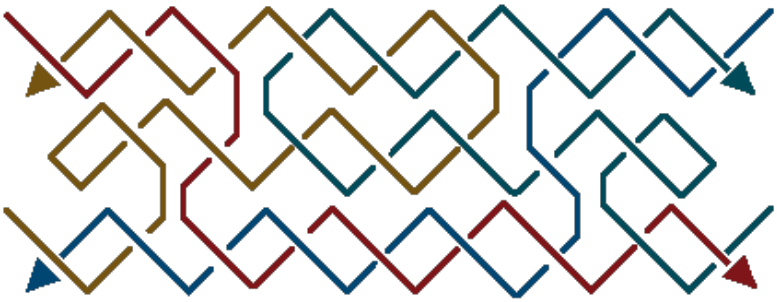


ABBILDUNG 2.4: Eine schematische Repräsentation eines TX-Tiles. [12]

Holliday-Junctions sind von besonderem Interesse für Nanogeräte und Nanonetzwerke auf Basis von DNA, da sie über offene Enden in vier verschiedene Richtungen verfügen. Das genügt, um komplexes Bindungsverhalten umsetzen zu können. Zudem lässt sich die Länge der offenen Enden und damit die Stärke einer potentiellen Bindung variieren.

Abbildung 2.5 zeigt eine schematische 3D-Darstellung der Molekularstruktur einer Holiday-Junction. Abbildung 2.6 zeigt selbiges Tile inklusive eingetragener, beispielhafter Basenpaare. Es ist zu erkennen, dass vier unterschiedliche Einzelstränge am Prozess beteiligt sind. Jeder Strang ist mit zwei weiteren, benachbarten Strängen verbunden. Die offenen Enden der Tiles können beliebige Sequenzen von Basenpaaren enthalten. Auf die Weise kann sichergestellt werden, dass nur bestimmte Tiles miteinander Bindungen eingehen können.

2.2.2.4 3D-TILES

Der nächste logische Schritt nach stabilen Tiles mit offenen Enden in vier verschiedenen Richtungen einer Ebene sind dreidimensionale Tiles [138]. Bislang wurden keine dreidimensionalen Tiles in einer Laborumgebung synthetisiert. Dies schließt die generelle Machbarkeit jedoch nicht aus. Es wurde bereits eine Vielzahl von dreidimensionalen Strukturen im Labor hergestellt [57, 86, 87, 139, 77, 42, 85, 121]. Einige Techniken basieren auf zweidimensionalen Strukturen, welche gebogen oder geknickt werden [74, 55]. Ein Beispiel dafür ist die DNA-Box von Andersen et al. [34]. All diese Beispiele zeigen, dass dreidimensionale Tiles synthetisierbar sein sollten, wenn ausreichend Interesse in der Forschungsgemeinschaft besteht.

Abbildung 2.7 zeigt ein Skizze eins dreidimensionalen Tiles. Das dargestellte Objekt gleicht einem dreidimensionalen kartesischen Koordinatensystem. Es sind insgesamt sechs Doppelstränge abgebildet, jeder Teil von zwei offenen Enden der dreidimensionalen Tile. Die offenen Enden mit den Basensequenzen C-A-T und G-T-A verzweigen sich in die dritte Dimension. Die übrigen offenen Enden verlaufen in der zweidimensionalen Ebene.

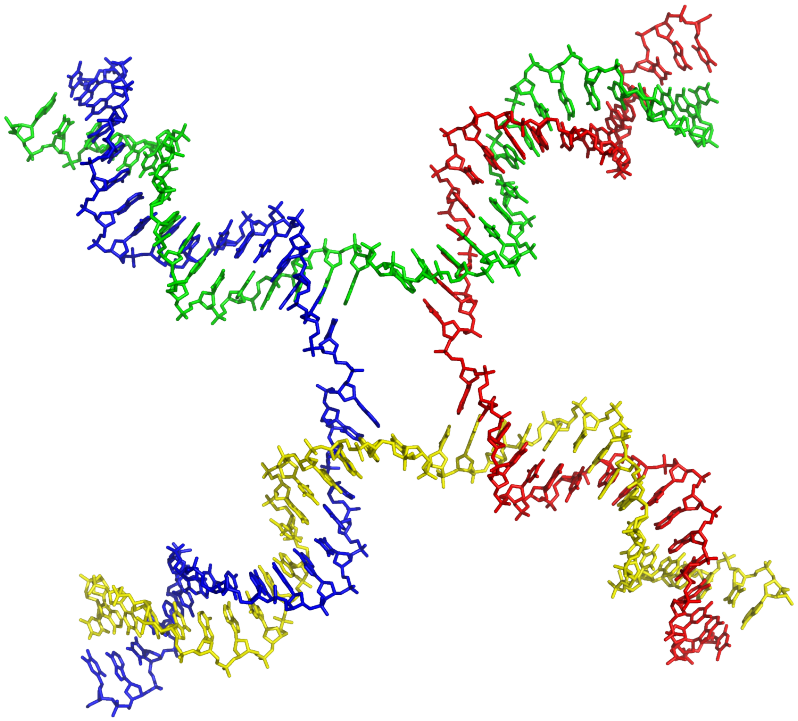


ABBILDUNG 2.5: Schematische Repräsentation eines DNA-Tiles mit Klebern in vier Richtungen. [169]

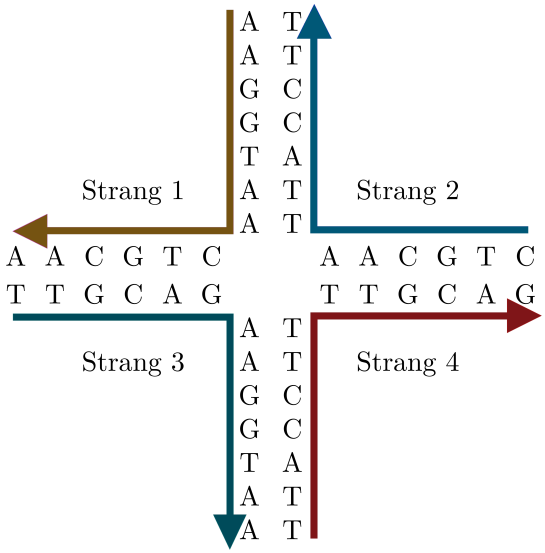


ABBILDUNG 2.6: Schematische Repräsentation eines DNA-Tiles mit Klebern in vier Richtungen und eingetragenen Basenpaaren. [168]

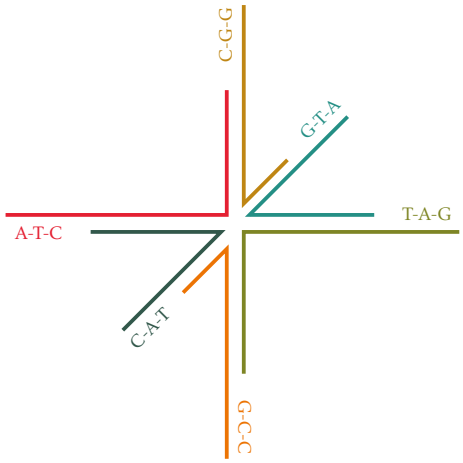


ABBILDUNG 2.7: Schematische Darstellung eines 3D-Tiles.

2.2.3 DNA-ORIGAMI

DNA-Origami als Technik zur Erzeugung von nanogroßen Strukturen wurde erstmals von Paul Rothemund im Jahre 2006 vorgestellt [117]. Die Technik basiert auf einem langen DNA-Strang, der an diversen Stellen durch DNA-Klammern mit sich selbst verbunden wird, wie in Abbildung 2.8 (a) verdeutlicht wird. *DNA-Klammern* sind kurze DNA-Stränge, welche an zwei vorgesehenen Orten des langen DNA-Stranges binden und diesen so falten. Durch die Basensequenzen der Klammern und des langen Stranges kann präzise bestimmt werden, welche Form das entstehende Objekt haben soll. Abbildung 2.8 (b) zeigt, wie der Einzelstrang reihenweise gefaltet wird. Mögliche Ergebnisse sind in Abbildung 2.8 (d) dargestellt. Als Machbarkeitsbeweis produzierte Rothemund nanogroße Strukturen in der Form von Smileys, Sternen oder Rechtecken. Abbildung 2.8 (c) und (d) zeigen dies im Detail.

DNA-Origami wurde auf verschiedene Weisen auf die dritte Dimension erweitert. Eine der einfachsten Techniken besteht darin, mehrere Schichten von DNA-Geflechten zu stapeln und mittels DNA-Klammern zu einer stabilen, dreidimensionalen Struktur zusammenzufassen [85]. Die Robust- und Einfachheit des DNA-Origami-Verfahrens machen es zu einem der beliebtesten Werkzeuge zur Erschaffung von Strukturen auf Nanoebene. Sogar die steuerbare DNA-Box von Andersen et al. konnte mittels DNA-Origami hergestellt werden [34].

Im folgenden Abschnitt werden verschiedene Strukturen und Geräte definiert, welche sich unter anderem aus DNA erzeugen lassen könnten.

2.3 DEFINITION VON MASCHINENMODELLEN UND NANONETZWERKEN

In den folgenden Abschnitten werden Nanogeräte und Nanonetzwerke mathematisch formalisiert. Dafür werden verschiedene Definitionen von makroskopische Geräten und Komponenten zurate gezogen und für die Nanoebene adaptiert.

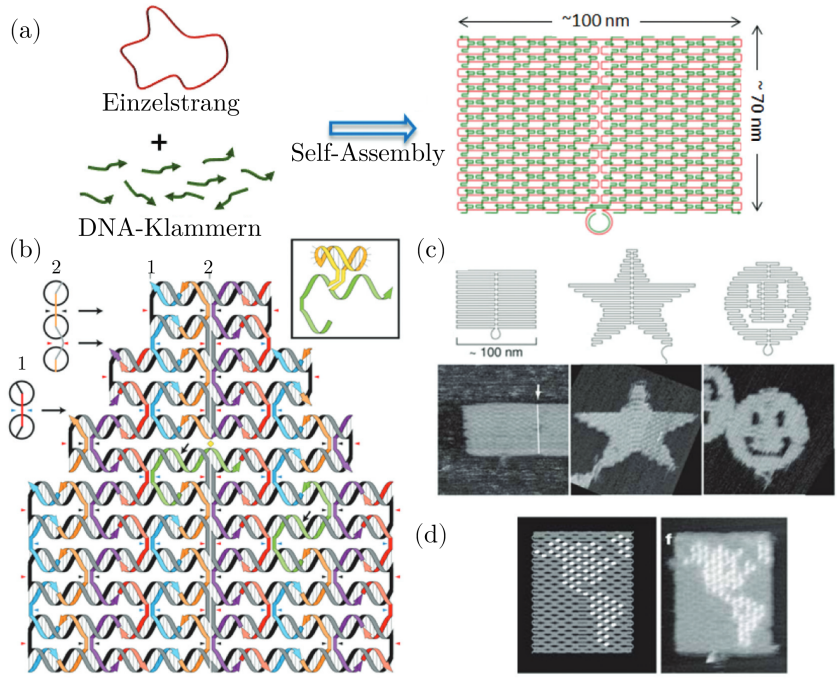


ABBILDUNG 2.8: Ein langer DNA-Strang wird mittels kurzer DNA-Klammern an sich selbst befestigt, um beliebige zweidimensionale Strukturen zu erzeugen. [60]

Des Weiteren werden die bestehenden Definitionsansätze analysiert und bei der Generalisierung für den Nanobereich berücksichtigt.

In der Nanoforschung wurde lange auf formale Definitionen verzichtet. Bis [4] sich der Probleme annahm, wurden die Begriffe Nanomaschine, Nanogerät, Nanoroboter und Nanobot mehr oder weniger synonym verwendet, wobei sie oft in nicht genannte Details voneinander abweichen. Häufig wurde sich auf intuitive Beschreibungen verlassen. Maschinenmodelle wurden praktisch nie verwendet.

Obwohl kaum ganzheitliche, formale Ansätze vorliegen, schränken einige Publikationen den Möglichkeitenraum von Nanogeräten ein. Dies erfolgt entweder durch globale Beschränkungen oder durch geforderte Eigenschaften. Es wird beispielsweise davon ausgegangen, dass einige Komponenten vorhanden sein müssen, damit man von Nanogeräten sprechen kann. Unter anderem handelt es sich dabei um Sensoren, Aktuatoren, Antennen, Prozessoren, Speicher und eine Energieversorgung [26, 158, 172]. Als *Aktuatoren* werden mechanisch agierende Komponenten von Nanogeräten bezeichnet.

Einige Herangehensweisen beziehen sich vornehmlich auf biologische Komponenten [28, 155]. Andere fokussieren sich auf die Beschreibung von Nanogeräten durch die von ihnen zu absolvierenden Aufgaben. Dabei liegt der Schwerpunkt auf Aktuatorik, Sensorik und Berechnung [114, 58].

Bezüglich der Größe herrscht Unklarheit. Einige Veröffentlichungen nehmen eine generelle Größenbeschränkung von 1–100 Nanometern an [114]. Andere verwenden eine Maximalgröße von wenigen Mikrometern [172]. Eine schlüssige Argumentation für die Parameterwahl wird selten präsentiert.

Obwohl Geräte in der Größenordnung von wenigen Mikrometern eigentlich nicht mehr Nanogeräte genannt werden können, sind sie immer noch von Effekten betroffen, welche einen negativen Einfluss auf Nanogeräte haben können. Dabei kann es sich beispielsweise um Elektronen handeln, welche zwischen elektrischen Leitern springen und so das Ergebnis von Berechnungen verfälschen. Der Einfluss solcher Effekte wird als Schlüsselklassifikator für die Benennungsfrage herangezogen.

Neben der Größe als Alleinstellungsmerkmal für Nanogeräte werden diese oft als autonome Maschinen oder Roboter bezeichnet. Makroskopische Maschinen und Roboter sind Teil eines etablierten Forschungsgebietes und sinnvolle Definitionen können als Inspiration herangezogen werden [23, 154].

Der vorgestellte Definitionsansatz kombiniert beide Bereiche miteinander, wodurch eine präzise und formale Definition für verschiedene nanogroße Geräte abgeleitet wird. Die folgenden Definitionen berücksichtigen zudem das Umfeld, in dem Nanoroboter eingesetzt werden sollen. Ein beliebter Anwendungsbereich ist Kontext der Arbeit die Medizin.

2.3.1 MAKROSKOPISCHE GERÄTEKLASSEN

In diesem Abschnitt werden zuerst Definitionen von makroskopischen Geräten vorgestellt, welche als Basis für ähnliche Definitionen auf Nanoebene herangezogen werden.

Definitionen für Roboter wurden unter anderem von Xie und der ISO-Norm 8373 vorgestellt:

„Ein Roboter ist ein Zusammenschluss von manipulativen Fertigkeiten, Fortbewegungsmitteln, sensorischen Fertigkeiten, Kommunikationsfertigkeiten und Verarbeitungslogik in einem künstlichen Körper." [23]

Ein Roboter nach Xie ist folglich ein Zusammenschluss von bestimmten Komponenten und Fertigkeiten.

„Ein Roboter ist ein in zwei oder mehr Dimensionen programmierbarer Aktuator, welcher über einen Grad an Autonomie verfügt und sich in seinem Umfeld bewegen kann, um Aufgaben zu erfüllen." [154]

Der ISO-Norm 8373 entsprechend ist ein Roboter weniger spezifisch definiert. Sensorische Fertigkeiten werden nur implizit gefordert und Kommunikation wird nicht erwähnt. Allerdings wird zusätzlich das Umfeld eines Roboters berücksichtigt.

Ein Roboter wird demnach durch zwei verschiedene Kategorien von Klassifikatoren definiert:

1. Die beteiligten physischen Komponenten. Dazu zählen Aktuatoren, Fortbewegungsmittel, Sensorik, Kommunikationsfertigkeit, Verarbeitungslogik…

2. Qualitäten, welche Design und Verhalten spezifizieren. Unter anderem die Künstlichkeit des Roboters, die Programmierbarkeit, ein gewisser Grad an Autonomie und die Tatsache, dass das Gerät die vorgesehene Tätigkeit ausführt.

Für die folgenden Definitionen wurden beide Kategorien berücksichtigt. Die erste Kategorie umfasst Komponenten, welche notwendig sind, um das beschriebene Verhalten physikalisch zu ermöglichen. Dazu zählen *Sensoren* S, *Aktuatoren* A, eine Komponente zur *Fortbewegung* L und eine Komponente für die *Kommunikation* C mit anderen Geräten. Des Weiteren muss es Komponenten geben, welche komplexes Verhalten ermöglichen. Dies umfasst eine Komponente zur *Informationsverarbeitung* I, optional mit *Speicher* M für Daten oder Programme, und einem *Zeitgeber* T. Zuletzt benötigen Nanogeräte eine *Energieversorgung* P. Die Komponenten werden in einem späteren Abschnitt detaillierter beschrieben. Im Gegensatz zu [4] werden die verschiedenen, folgenden Geräteklassen als eine Menge von Komponenten definiert.

Auf Basis der genannten Komponenten kann nun eine Definition für makroskopische Maschinen und Roboter hergeleitet werden. Dabei werden Teile der Notationen von Xie [23] und dem ISO-Norm [154] verwendet.

DEFINITION 2.1 – Eine *Maschine* $\mathcal{M} = K_{\text{mand}} \cup K_{\text{opt}}$ ist ein künstliches Konstrukt, welches konzipiert ist, um eine definierte Aktion auszuführen. Sie besteht aus einer Menge notwendiger Komponenten $K_{\text{mand}} = \{A, P\}$ und einer Menge aus null oder mehr optionalen Komponenten $K_{\text{opt}} \subseteq \{C, I, L, M, S, T\}$.

Diese Definition repräsentiert eine Maschine als eine energieautarke Entität, welche ihre Umwelt auf definierte Weise beeinflussen kann. Beispielsweise verfügt ein Bagger über Kraftstofftank und Motor P sowie einen schwenkbaren Arm A. Damit sind alle Komponenten vorhanden, sodass ein Bagger nach der vorgestellten Definition eine Maschine ist. Darüber hinaus verfügen viele Exemplare über Ketten oder Räder L, was verdeutlicht, dass oft zusätzliche Fertigkeiten vorhanden sein können. Davon abgeleitete, oft komplexere Konstrukte können demnach auch Maschinen sein, obwohl sie einen größeren Funktionsumfang haben.

Die zusätzlichen Einschränkungen grenzen Maschinen von natürlichen oder zufälligen Phänomenen ab. Die Künstlichkeit fordert, dass nur von Menschen gemachte Konstrukte Maschinen genannt werden. Die Forderung nach einer vordefinierten Aufgabe bedingt eine intentionale Erzeugung. Weitere Einschränkungen sind nicht nötig, um makroskopische Maschinen korrekt zu klassifizieren.

DEFINITION 2.2 – Ein *Roboter* $\mathcal{R} = K_{\text{mand}} \cup K_{\text{opt}}$ ist eine Maschine, die programmierbar ist und über eine Menge an notwendigen Komponenten $K_{\text{mand}} = \{A, I, M, P, S\}$ verfügt. Zudem kann ein Roboter über eine Menge an optionalen Komponenten $K_{\text{opt}} \subseteq \{C, L, T\}$ verfügen.

Roboter verfügen über zusätzliche notwendige Komponenten sowie weitere Einschränkungen. Die wichtigste neue Komponente I dient der Informationsverarbeitung. Diese kann einem Roboter die Durchführung komplexer Aufgaben durch verschiedene Programmierungen ermöglichen. Programmierung darf dabei nicht mit Konfigurierbarkeit verwechselt werden, da der Grad an Komplexität deutlich höher ist. Da der Übergang jedoch kontinuierlich ist, könnten manche Maschinen fälschlicherweise als Roboter klassifiziert werden. Maschinen sind eine Obermenge von Robotern.

Damit eine Programmierung vorgenommen werden kann, muss eine Informationsverarbeitungseinheit in der Lage sein, einen Zustand zu speichern. Dies erfordert eine Form der persistenten Sicherung von Informationen, zum Beispiel in Form von Bits. Damit die Definition der obigen gerecht wird, verfügt ein Roboter ebenfalls über eine Sensorkomponente.

2.3.2 NANOGERÄTEKLASSEN

Auf Basis der makroskopischen Definitionen für Maschinen und Roboter können nun die Derivate auf Nanoebene abgeleitet werden. Um weniger spezifische Geräte zuzulassen, werden zuerst abstrakte Klassen für Nanostrukturen und Nanogeräte definiert.

Die Definitionen der Nanogeräte stammen von Florian Büther, Florian Lau, Marc Stelzner und Sebastian Ebers aus [4]. Florian Lau hat dabei die Formalisierung der Nanogeräte durchgeführt.

DEFINITION 2.3 – Eine *Nanostruktur* $\mathcal{N}_S = K_{\text{opt}}$ ist ein nanogroßes, künstliches Konstrukt, welches konzipiert wurde, um eine bestimmte Funktion in einer Umgebung Γ zu erfüllen. Eine Nanostruktur besteht aus null oder mehr optionalen Komponenten $K_{\text{opt}} \subseteq \{A, C, I, L, M, P, S, T\}$.

Nanostrukturen sind nach Partikeln und natürlich vorkommenden Strukturen die allgemeinsten, potenziell komplexen Strukturen, welche für den Einsatz auf Nanoebene geschaffen werden können. Diese sind für die vorliegende Arbeit von besonderem Interesse. Diesen wurde in der Nanonetzwerkgemeinschaft bislang wenig Beachtung geschenkt.

Nanogröße entspricht dabei einer Größe von einem Nanometer bis zu weniger Mikrometern. Auch wenn mehr als 1000 Nanometer technisch nicht mehr "nanogroß" sind, wird der Begriff dennoch in der Literatur so verwendet, da beispielsweise das menschliche Kapillarsystem Blutkörperchen von vier Mikrometern Größe zulässt [172, 114].

DEFINITION 2.4 – Ein *Nanogerät* $\mathcal{N}_D = K_{\text{mand}} \cup K_{\text{opt}}$ ist eine Nanostruktur. Es besteht aus einer Menge notwendiger Komponenten $K_{\text{mand}} = \{P\}$ und einer Menge aus null oder mehr optionalen Komponenten $K_{\text{opt}} \subseteq \{A, C, I, L, M, S, T\}$.

Nanogeräte benötigen eine autarke Energieversorgung P, damit sie von vollständig passiven Nanostrukturen oder Partikeln unterschieden werden können.

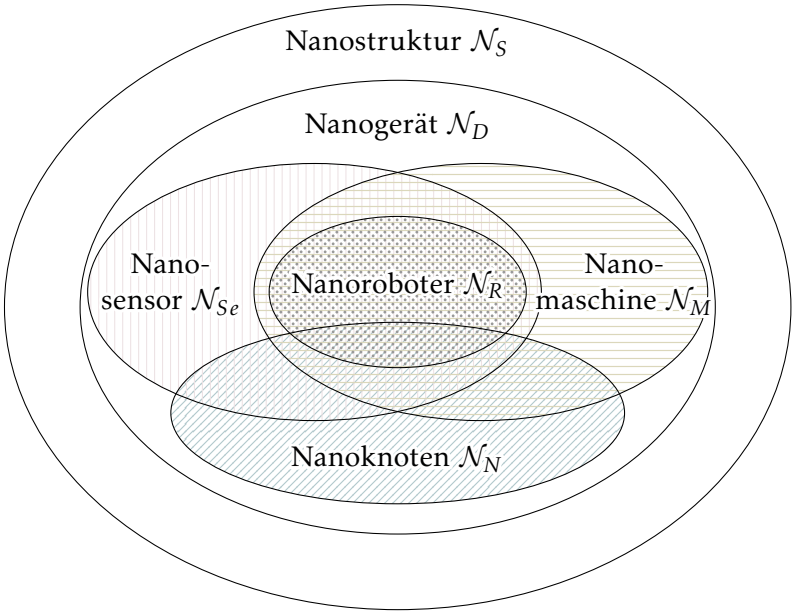


ABBILDUNG 2.9: Übersicht verschiedener Nanostrukturen.

Genau wie bei makroskopischen Maschinen dürfen zusätzliche Komponenten vorhanden sein. Ein Nanogerät ist eine Nanostruktur und alle weiteren Geräte werden auf deren Basis definiert. Abbildung 2.9 illustriert die hierarchische Vorgehensweise bei der Definition der Geräte.

Nanogeräte werden voraussichtlich in ungewöhnlichen Umgebungen, zum Beispiel im menschlichen Körper, eingesetzt. Die Größe der Nanogeräte birgt neue Herausforderungen wie zum Beispiel molekulare Absorption bei der Kommunikation oder Quanteneffekte [46]. Die Existenz dieser Anforderungen wird durch die Umgebung Γ verdeutlicht. Die Umgebung Γ kann als eine Liste von Einschränkungen oder Bedingungen verstanden werden. Eine mögliche Bedingung ist die Größe eines Gerätes. Wie in Abschnitt 2.3.5 gezeigt wird, beschränkt zum Beispiel das Kapillarsystem des Menschen einige Nanogeräte auf eine Größe von weniger als vier Mikrometern.

DEFINITION 2.5 – Eine *Nanomaschine* $\mathcal{N}_M = K_{\text{mand}} \cup K_{\text{opt}}$ ist ein Nanogerät mit notwendigen Komponenten $K_{\text{mand}} = \{A, P\}$ und einer Menge aus null oder mehr optionalen Komponenten $K_{\text{opt}} \subseteq \{C, I, L, M, S, T\}$, in einer Umgebung Γ.

Diese Definition interpretiert Nanomaschinen als sehr kleine Maschinen. Ein künstliches Enzym, welches Proteine faltet, kann laut dieser Definition, unter Nichtbeachtung einer direkten Energieversorgung, eine Nanomaschine sein.

Ähnlich wie bei Maschinen kann das Konzept eines Sensorknotens für die Nanoebene umformuliert werden.

DEFINITION 2.6 – Ein *Nanosensor* $\mathcal{N}_{Se} = K_{\text{mand}} \cup K_{\text{opt}}$ ist ein Nanogerät mit den notwendigen Komponenten $K_{\text{mand}} = \{P, S\}$ und einer Menge von null oder mehr optionalen Komponenten $K_{\text{opt}} \subseteq \{A, C, I, L, M, T\}$ in einer Umgebung Γ.

Eine einfache Nanomaschine oder ein einfacher Nanosensor kann eine simple Aufgabe endlos wiederholen, ohne jemals Informationen aus der Umgebung für die Interaktion auszuwerten. Um komplexere Aufgaben lösen zu können ist es jedoch erforderlich autonom eine Abstraktion des Umfeldes Γ innerhalb des Nanogerätes zu erzeugen.

DEFINITION 2.7 – Ein *Nanoroboter* oder *Nanobot* $\mathcal{N}_R = K_{\text{mand}} \cup K_{\text{opt}}$ ist ein programmierbares Nanogerät mit einem hohen Grad an Autonomie in einer Umgebung Γ. Er besteht aus einer Menge notwendiger Komponenten $K_{\text{mand}} = \{A, I, M, P, S\}$ und einer Menge von null oder mehr optionalen Komponenten $K_{\text{opt}} \subseteq \{C, L, T\}$.

Nanoroboter haben dieselben notwendigen und optionalen Komponenten sowie die zusätzlichen Einschränkungen der makroskopischen Variante. Die Nanoroboter werden um Autonomie erweitert. Die Umgebung ist in diesem Fall von besonderer Bedeutung, da Roboter in ständiger Wechselwirkung mit ihrem Umfeld Γ stehen.

Als Beispiele für Nanogeräte auf biologischer, prokaryotischer und auf Basis von elektrotechnischen Komponenten dienen die Abbildungen 1.2, 1.3 und 1.4. Eine Klassifikation auf Basis der vorhandenen Komponenten wird dem interessierten Leser als Übung überlassen.

2.3.3 KOMPONENTEN VON NANOGERÄTEN UND NANOSTRUKTUREN

Im Abschnitt 2.3.1 wurde eine Reihe von Komponenten genannt, welche Teil der verschiedenen, vorgestellten Geräte sein können. Um die genannten Definitionen zu präzisieren, werden die einzelnen Komponenten hier im Detail erläutert.

Alle hier aufgezeigten Komponenten fallen in die Kategorie der Hardware. Sie sind physikalische Konstrukte oder Teile von Maschinen und Robotern. Jegliches Verhalten ist ein Softwareteil der Komponente Informationsverarbeitung I.

Die einzelnen Komponenten werden in den folgenden Abschnitten in alphabetischer Reihenfolge erläutert.

Aktuatoren A Manipulation oder Aktuation beschreiben die Fertigkeit eines Nanogerätes, seine Umgebung zu beeinflussen. Dies kann über verschiedene Umweltparameter geschehen, beispielsweise physikalische, chemische oder biologische.

Aktuatoren lassen sich durch Nutzung von Komplementarität konzipieren. Dabei werden molekulare Anziehungs- und Abstoßungsreaktionen genutzt. So lässt sich zum Beispiel ein Öffnungsmechanismus erzeugen, indem ein Behältnis mit einer schwachen Bindung geschlossen wird. Dieses kann geöffnet werden, indem ein stärker bindendes Molekül in die Nähe des Verschlussmechanismus gebracht wird und diesen so öffnet [34]. Abbildung 2.10 zeigt ein Beispiel dafür.

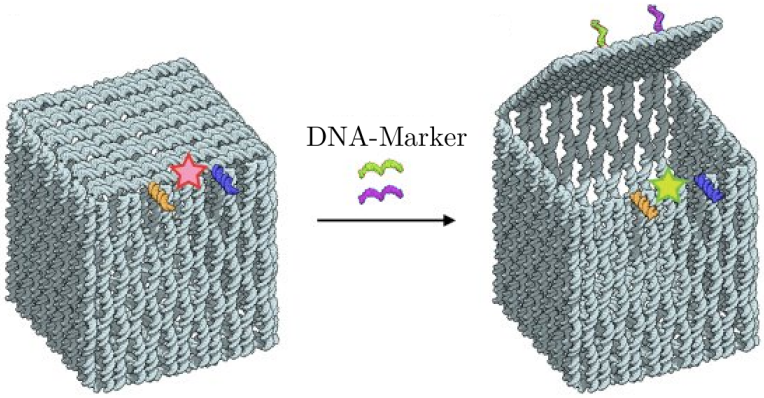


ABBILDUNG 2.10: Schematische Darstellung einer Box, die durch DNA-Origami hergestellt wurde und sich durch spezielle DNA-Moleküle öffnen lässt. Der rote Stern repräsentiert einen geschlossenen Zustand der Box, ein Grüner einem offenen Zustand. Die DNA-Marker in der Mitte passen besser auf einen Teil des Verschlussmechanismus und können diesen öffnen. [34]

Kommunikation *C* Kommunikation ist ein Informationsaustausch zwischen zwei oder mehr Entitäten, welche über die nötige Befähigung verfügen. Dabei wird eine passende, möglichst verlässliche Umgebungsvariable manipuliert und anschließend von einem anderen Gerät gemessen. Dafür werden sowohl Aktuatoren *A* zur Manipulation als auch Sensoren *S* für die Messung benötigt.

Eine Kommunikation muss unter dem Vorsatz eines Informationsaustausches geschehen, da anderweitig natürliche Prozesse als Kommunikation verstanden werden könnten. Kommunikation beschreibt lediglich die "Befähigung" zum Informationsaustausch. Ein Nanogerät kann durch Effekte wie Rauschen permanent an einer erfolgreichen Kommunikation gehindert werden.

Die für die Kommunikation eingesetzte Komponente unterscheidet sich dabei technisch nicht von herkömmlichen Sensoren und/oder Aktuatoren. Gerade bei biologischen Nanogeräten, welche über den Austausch von Molekülen kommunizieren, sind diese oft identisch. Um Messung und Aktuation von Kommunikation unterscheiden zu können, wird eine zuvor festgelegte Intention vorausgesetzt. Ein Nanogerät kommuniziert demnach, wenn Manipulation und Messung mit dem Vorsatz der Kommunikation geschehen, andernfalls wird nur ein Umgebungsparameter manipuliert oder gemessen.

In einigen Fällen wird es einem Nanogerät nicht möglich sein, zwischen Kommunikationsversuch und einer zufälligen Veränderung des beobachteten Parameters zu unterscheiden. Hier können Kommunikationsprotokolle Abhilfe schaffen, indem Fehlerkorrekturmechanismen Anwendung finden oder Prüfsummen zur Detektion von Fehlern verwendet werden.

Es werden wenigstens drei wichtige Verfahren zur Kommunikation auf Nanoebene unterschieden:

- Akustische Kommunikation,
- Elektromagnetische Kommunikation und
- Molekulare Kommunikation.

Lediglich molekulare Kommunikation ist im Kontext der Arbeit von Interesse, da diese durch DNA-Tiles realisiert werden kann.

Im folgenden Abschnitt werden die verschiedenen molekularen Kommunikationskonzepte auf Nanoebene gegenübergestellt.

Tabelle 2.1 zeigt eine Liste von molekularen Kommunikationskonzepten und den ihnen zugeordneten Eigenschaften. Die Kommunikationsarten werden anhand ihrer zugrundeliegenden Konzepte kategorisiert.

Die Tabelle verdeutlicht, dass die meisten Kommunikationsmechanismen auf der Basis von Diffusion Informationen verbreiten. Lediglich Bakterien-basierte Techniken weichen davon ab, diese verwenden Chemotaxis [38]. *Chemotaxis* beschreibt dabei die von Bakterien genutzten Anziehungs- und Abstoßungskräfte.

Partikelbasierte Verfahren nutzen meistens Moleküle, welche im Umfeld vorhanden sind. Sollte das Umfeld der menschliche Organismus sein, so kann dies die körpereigenen Regelkreise beeinträchtigen. Tiles und Bakterien haben vermutlich wenig Konfliktpotential.

Nichtsdestotrotz ist es schwierig, die verschiedenen Kommunikationsarten miteinander zu vergleichen.

Vor allem Tile-basierte Kommunikation stellt ein Problem dar, da aus Tiles zuerst Nachrichtenmoleküle zusammengesetzt werden, welche wiederum wie herkömmliche Moleküle bei der Kommunikation verwendet werden können. Der hierarchische Prozess der Nachrichtenzusammensetzung ist im Hinblick auf Netzwerke kaum untersucht worden.

Aus Tabelle 2.1 geht weiterhin hervor, dass Tile-basierte Verfahren im Vergleich als positiv zu bewerten sind, da sie wenig bioinvasiv sind und viel Information pro Partikel versendet werden kann. Die geringe Bioinvasivität ist darauf zurückzuführen, dass DNA im menschlichen Körper an praktisch allen Orten vorkommen kann und demnach kaum Antikörper existieren. Es ist demnach davon auszugehen, dass DNA-Tiles bereits existierende Kommunikationsmechanismen im menschlichen Körper weniger beeinflusst als andere Verfahren.

Konzentrationsbasierte Verfahren vermitteln vergleichsweise wenig Bits pro Partikel, da nur die Gesamtkonzentration von Partikeln in einem Medium gemessen werden kann, welche oft sehr hoch sein muss um sinnvoll zwischen einer logischen „1" und einer logischen „0" unterscheiden zu können.

Informationsverarbeitung *I* Die Informationsverarbeitungskomponente *I* erlaubt einem Nanogerät die Manipulation und Transformation von Daten. Im

Parameter	Fortbewegung	Bioinvasivität	Bits/Partikel
Tiles	Diffusion	sehr niedrig	$1/k$
Partikelanzahl	Diffusion	hoch	$1/k$
Konzentration	Diffusion	hoch	$1/k$
Partikeltyp	Diffusion	hoch	$1/\log_2 n$
Partikelanordnung	Diffusion	hoch	$2/n$
Bakterien [38]	Chemotaxis	niedrig–hoch	einige
Viren [127]	Diffusion	niedrig–hoch	einige

TABELLE 2.1: Vergleich verschiedener molekularer Kommunikationsmethoden aus [61]. k ist die Anzahl der Partikel in einer Nachricht. n ist die Anzahl verschiedener Partikeltypen.

einfachsten Fall kann die Informationsverarbeitung durch eine boolesche Operation repräsentiert werden, welche durch einen einzelnen Transistor umgesetzt wird. Informationsverarbeitung setzt voraus, dass Nanogeräte programmiert werden können, um so das Verhalten einer Informationstransformation steuern zu können. Berechnungsmodelle wie Schaltkreise oder Self-Assembly-Systeme lassen sich nicht im klassischen Sinne programmieren. In diesem Fall muss es möglich sein, einen neuen Schaltkreis oder ein neues Self-Assembly-System zu erzeugen, welches bei gleichen Eingaben unterschiedliche Ausgaben erzeugt.

Prinzipiell kann die Programmierbarkeit als eine Form der Gerätekonfiguration interpretiert werden. Programmierbarkeit ist folglich ein Spektrum, beginnend mit einigen Bit-Schaltern, welche das grobe Maschinenverhalten steuern, bis hin zu Turing-vollständigen Interpretern für eine Maschinensprache.

Mögliche Modelle für die Informationsverarbeitung sind klassische Computer auf Basis von Transistortechnologien wie CMOS. Diese können zu Schaltkreisen organisiert werden, um eine Turing-vollständige Logik abzubilden.

Eine Logik von ähnlicher Mächtigkeit kann auf Basis von Quantum-Dot zellulären Automaten generiert werden, welche im Abschnitt 2.5.3 vorgestellt werden.

Ein weiteres Berechnungsmodell, welches zur Informationsverarbeitung herangezogen werden kann, sind Self-Assembly-Systeme, welche ebenfalls Turing-vollständig sind. Diese werden in Abschnitt 2.5.5 im Detail erklärt.

Fortbewegung L Die Fortbewegungskomponente L beschreibt die Möglichkeit eines Nanogerätes seinen Aufenthaltsort zu verändern. Fortbewegung wird in zwei Kategorien unterteilt: aktive und passive Bewegung. Passiv mobile Nanogeräte werden durch Strömungsvorgänge in einem Medium befördert und

benötigen keine weiteren externen Einflüsse. Ein Beispiel sind Nanoroboter, welche sich zusammen mit dem Blutfluss durch einen Organismus bewegen.

Aktiv mobile Nanogeräte verwenden, ähnlich wie bei der Kommunikation, eine spezielle aktuatorische Komponente. Die Beeinflussung der Umgebung geschieht dabei unter der Intention der Fortbewegung. Die folgende Aufzählung zeigt verschiedene aktive und passive Möglichkeiten zur Fortbewegung in flüssigen Medien. Einige sind dabei biologisch motiviert, während andere klassische, mechanische Vorbilder haben:

1. Externe Applizierung von Magnetfeldern [131],
2. Brownsche Molekularbewegung,
3. Bubble-Propulsion,
4. DNA-Walker,
5. Chemotaxis,
6. Flagella und
7. Molekulare Motoren.

Eine Zusammenfassung zum Thema Fortbewegung auf Nanoebene kann in [128] eingesehen werden.

Von den aufgezählten Technologien ist vor allem die passive Fortbewegung (Punkt 2) für diese Arbeit von Interesse, da die Kompatibilität der übrigen Techniken mit DNA-Technologien ungeklärt ist. DNA ist, wie alle anderen Moleküle auch, der Brownschen Molekularbewegung unterworfen. Dieser Effekt wird in der Arbeit ausgenutzt und als Verteilungs- und Fortbewegungsmechanismus für alle DNA-basierten Technologien angenommen.

Dieser passive Fortbewegungsmechanismus steht in jeder Flüssigkeit zur Verfügung und stellt dabei keinen unbekannten Einfluss für biologische Systeme dar.

Es kann schwierig sein, einen Nanoroboter manuell und gezielt im menschlichen Körper zu positionieren, ohne dabei unerwünschte Nebenwirkungen zu erzeugen. Das Zielgebiet könnte zum Beispiel schwer erreichbar oder durch Vorerkrankungen nur eingeschränkt belastbar sein. Bei der passiven Fortbewegung kann wenigstens damit gerechnet werden, dass der Organismus in gewissem Maße an den Vorgang angepasst ist.

Speicher M Der Speicher M beschreibt die Fähigkeit eines Nanogerätes Informationen persistent vorzuhalten. Speicher ist vor allem in Hinblick auf die Informationsverarbeitung I von Interesse, da erst durch Speicher Programme oder Konfigurationen verwendet werden können, was wiederum steuerbares oder komplexes Verhalten ermöglicht.

Trotz des offensichtlichen Nutzens für viele Anwendungen ist Speicher keineswegs notwendig oder selbstverständlich. Schaltkreise benötigen beispielsweise keinen Speicher und sind dennoch in der Lage komplexe Berechnungen durchzuführen. Für die Datenerfassung und Übermittlung ist meistens jedoch Speicher notwendig.

Eine Möglichkeit, Speicher zu realisieren, basiert auf Molekülen wie der DNA. DNA stellt, anders als das Binäralphabet, ein Alphabet zur Basis vier zur Verfügung. Informationen können kodiert – in Form von DNA – über lange Zeit gespeichert werden. Die Lebensdauer von einigen durch DNA-basierte Speichervarianten hinterlegten Informationen überschreitet die von Magnetbändern, CDs oder Festplatten bei weitem [99].

Energieversorgung *P* Die Energieversorgungskomponente *P* beschreibt auf welche Weise die Sensorik und Aktuatorik eines Nanogerätes mit Energie versorgt werden. Die Energieversorgung *P* ist eine der wenigen, für Nanogeräte notwendigen Komponenten. Es kann nicht vorausgesetzt werden, dass ein Nanogerät zuverlässig und dauerhaft an eine Energiequelle angeschlossen ist. Eine interne Energiequelle wie zum Beispiel eine Batterie oder ein Miniaturkraftwerk können notwendig sein.

DNA-basierte Systeme nutzen die im Kommunikationskanal oder Medium vorhandene Wärmeenergie, um stabile Bindungen einzugehen oder diese aufzubrechen. DNA-Tiles ernten somit Energie aus der Umgebung und versorgen dadurch den Self-Assembly-Prozess mit Energie. Der Vorteil besteht darin, dass keine dedizierte Energieversorgungskomponente angefertigt werden muss. Teile des Umfeldes in lebenden Organismen sind darauf angepasst, Bindungsreaktionen ein synergetisches Umfeld zur Verfügung zu stellen. Ähnliche Bedingungen lassen sich leicht in einem künstlichen Umfeld wie einer Petrischale reproduzieren.

Es gibt zahlreiche weitere Verfahren, mit denen Energie aus dem Umfeld nutzbar gemacht werden kann. Ein verbreitetes Beispiel sind Zuckerbatterien, welche z. B. Maltodextrin nutzen, um Energie zu gewinnen [24].

Sensoren *S* Eine Messung ist ein Vorgang, bei dem ein meist kontinuierlicher Umweltparameter erfasst und in eine verarbeitbare Form umgewandelt wird. Ein typisches Beispiel für besagte Parameter ist eine elektrische Spannung oder ein elektrischer Widerstand.

Da Sensoren *S* oft kontinuierliche Werte wie Spannungen oder Widerstandswerte bereitstellen, kann ein A/D-Wandler notwendig sein, um die erfassten Werte später mittels einer digitalen Informationsverarbeitungseinheit *I* zu transformieren oder manipulieren zu können. Obwohl A/D-Wandler sehr komplex sein können, werden sie als Teil eines Sensors betrachtet – nicht als Teil einer Informationsverarbeitungseinheit *I*.

Im medizinischen Bereich unterteilt sich Sensorik *S* in Labor-analytische und apparative Verfahren [22]. Da apparative Verfahren meist komplizierte Geräte voraussetzen erscheinen diese ungeeignet für den Einsatz auf Nanoebene.

Labor-analytische Verfahren sind Mess- oder Nachweisverfahren bei denen durch den Einsatz von Sensorik die Ausprägung und Signifikanz von ausgewählten Parametern bestimmt wird. Ein Beispiel ist eine Blutgasanalyse.

Labor-analytische Verfahren sind von besonderem Interesse für Nanogeräte, da diese sich unter anderem mit der Messung von Körperparametern beschäfti-

gen. Sie unterteilen sich in quantitative – analoge Messwerte – und qualitative Verfahren – Wahrheitswerte.

Immunologische Verfahren sind ein besonders vielversprechender Spezialfall Labor-analytischer Verfahren. Diese können im Nanobereich dazu eingesetzt werden, um Antikörper der Größenordnung von ca. 15 nm nachzuweisen [102]. Dieses und weitere Verfahren z. B. auf Basis von Kohlenstoff-Nanoröhrchen nutzen das Konzept der Komplementarität [173]. Dieses beschreibt, dass sich bestimmte Moleküle anziehen oder abstoßen. Moleküle, die sich anziehen, binden oft aneinander und verändern so ihre elektrischen Eigenschaften. Auf diese Weise lassen sich auf Nanoebene Sensoren realisieren. Ist im Verlauf der Arbeit von "Markern" die Rede, so wird ein auf Komplementarität basierender, messbarer Antikörper angenommen.

Zeitgeber T Ein Zeitgeber T ermöglicht es einem Nanogerät, kausale Aussagen zu tätigen und zeitlich bedingtes Verhalten umzusetzen. Interne Uhren sind ein integraler Bestandteil von makroskopischen Rechenmaschinen. Ihre Verfügbarkeit auf der Nanoebene kann jedoch nicht vorausgesetzt werden. Demnach erscheint auch die Untersuchung der Mächtigkeit von Geräten ohne Zeitverständnis sinnvoll. Es werden drei verschiedene Arten von Zeitverständnis mit unterschiedlicher Mächtigkeit in Nanogeräten unterschieden:

1. Relatives Zeitverständnis durch die Happened-before Relation [92],
2. relatives Zeitverständnis, welches einem Nanogerät ermöglicht, eine zeitliche Differenz zwischen zwei Ereignissen festzustellen,
3. und absolutes Zeitverständnis durch beispielsweise Zeitstempel oder Uhren, welche in definierten Zeitschritten zählen und diese Information bereitstellen.

Generell ist bekannt, dass absolutes Zeitverständnis in verteilten Systemen nicht umsetzbar ist [92]. Da es sich bei Nanonetzwerken um verteilte Systeme auf Nanoebene handelt, folgt dass auch hier kein absolutes Zeitverständnis möglich ist. Es ist jedoch möglich, dass Nanonetzwerke sinnvoll interagieren, ohne den globalen Systemzustand abrufen zu können.

Die im Verlauf der Arbeit vorgestellten Self-Assembly-Systeme verfügen lediglich über ein Zeitverständnis auf Basis der Happened-before Relation. Dieses basiert auf der Tatsache, dass sich aus Tiles bestehende Strukturen inkrementell aufbauen und die Anwesenheit bestimmter Vorgänger vorausgesetzt wird.

2.3.4 NANONETZWERKE

Da sowohl der Platz für Berechnungen als auch der Speicher auf Nanoebene eingeschränkt sind, kann davon ausgegangen werden, dass viele Nanogeräte im Verbund an der Lösung einer Problemstellung beteiligt sein müssen [26]. Im Folgenden werden Definitionen für Geräteverbünde auf Nanoebene vorgestellt.

Eine Voraussetzung, um an einem Nanonetzwerk teilzunehmen, ist die Fertigkeit der potentiellen Teilnehmer zur Kommunikation. Diese gestattet Nanogeräten,

Informationen auszutauschen, sodass auf ein gemeinsames Ziel hingearbeitet werden kann.

DEFINITION 2.8 – Ein *Nanonetzwerk* ist ein gerichteter ad-hoc-Graph $G = (V, E)$, wobei V eine Menge von *Nanoknoten* $\mathcal{N}_N$ ist, welche in einer Umgebung Γ operieren. Nanoknoten sind Nanogeräte, welche zusätzlich über die notwendige Kommunikationskomponente $C \in K_{\text{mand}}$ verfügen. E ist eine Menge von Kanten $e \in V \times V$.

Analog zu makroskopischen Netzwerken können Nanonetzwerke als gerichteter Graph modelliert werden. Dabei beschreiben Knoten Nanoknoten und Kanten entsprechen der Möglichkeit einer Kommunikation mit anderen Geräten, welche sich in Reichweite befinden und nicht anderweitig an einer Kommunikation gehindert werden.

Das Umfeld Γ beeinflusst den Graphen maßgeblich. Kommunikationsreichweite und Rauschen werden durch das Umfeld bedingt. In einem instabilen oder mobilen Umfeld kann es leicht vorkommen, dass sowohl die Knotenpositionen als auch die Kanten variabel sind.

Klassische Netzwerktechnologien wie Routingalgorithmen oder Protokolle können in Nanonetzwerken vielleicht nicht zum Einsatz kommen, da Nanonetzwerke voraussichtlich aus enorm vielen Teilnehmern bestehen und das Speichern eines einzelnen Identifikators schon eine logarithmische Menge Speicher im Verhältnis zur Anzahl der Teilnehmer in Anspruch nimmt. Neue Methoden, um ein Netzwerk spontan zu formieren, obwohl kaum Speicher zur Verfügung steht, sind voraussichtlich notwendig.

2.3.5 MEDIZINISCHE NANOROBOTER UND NETZWERKE

Medizinische Nanoroboter existieren in einem spezifischen Umfeld, welches die Größe der Geräte einschränkt. In der menschlichen Blutbahn kann eine Größe von vier Mikrometern beispielsweise nicht überschritten werden, da dies den minimalen Durchmesser des Gefäßsystems beschreibt [17]. Um Nanogeräte in Zellen einzusetzen, müssen sie deutlich kleiner als vier Mikrometer sein. Im Verdauungstrakt hingegen sind weit größere Geräte denkbar.

DEFINITION 2.9 – Ein *medizinischer Nanoroboter* $\mathcal{R}_{\mathcal{M}}$ ist ein Nanoroboter, dessen Umgebung Γ ein Teil des menschlichen Körpers ist.

Der medizinische Anwendungsbereich stellt zusätzlich strenge Anforderungen an die Biokompatibilität. Es sollte nicht vorkommen, dass ein medizinischer Nanoroboter ungewollt eine Reaktion des Immunsystems provoziert. Nachdem ein medizinischer Nanoroboter seine Aufgabe erfüllt hat oder ein Defekt eingetreten ist, muss es dem menschlichen Körper möglich sein, die entstehenden Abfallprodukte abzubauen und auszuscheiden.

DEFINITION 2.10 – Ein *medizinisches Nanonetzwerk* ist ein Nanonetzwerk, in dem die teilnehmenden Nanoknoten die Einschränkungen des Umfelds Γ berücksichtigen, welche im menschlichen Körper auftreten können.

Ein medizinisches Nanonetzwerk ist damit ein Nanonetzwerk, welches explizit für den Einsatz im menschlichen Körper konzipiert ist. Ein Beispiel ist ein medizinisches Nanonetzwerk zur Erkennung von Lungenentzündungen [7]. Das Szenario wurde von Marc Stelzner, Florian Lau, Katja Freundt, Florian Büther, Mai Linh Nguyen, Cordula Stamme und Sebastian Ebers entwickelt. Florian Lau hat federführend mitgewirkt, Grafiken angefertigt und das Szenario mit entwickelt.

Um die Software von Nanogeräten und Nanonetzwerken und damit deren Verhaltensweisen genauer zu analysieren, ist es nötig, die Grundlagen der Komplexitätstheorie zu verstehen, welche im folgenden Abschnitt vorgestellt werden.

2.4 KOMPLEXITÄTSTHEORETISCHE GRUNDLAGEN

Die folgenden Abschnitte führen die zum Verständnis notwendigen Grundlagen der Komplexitätstheorie und der theoretischen Informatik ein. Zuerst wird formalisiert, was mathematisch unter "Problemen" und Berechnungen verstanden wird. Im Anschluss werden Werkzeuge präsentiert, mit denen die Güte von algorithmischen Problemlösungsverfahren bestimmt werden kann. Zuletzt wird das Konzept der Problemreduktion vorgestellt, welches genutzt werden kann, um Probleme anhand ihrer Schwierigkeit zu vergleichen und zu kategorisieren.

2.4.1 PROBLEME FORMALISIEREN

In der theoretischen Informatik sind zwei verschiedene Arten von formalen Problemen für Nanogeräte von besonderem Interesse. Es handelt sich dabei um Entscheidungsprobleme und Funktionsprobleme. Diesen Problemarten liegt das Berechnungsproblem zugrunde:

DEFINITION 2.11 – Ein *Berechnungsproblem* ist ein Tupel aus Eingaben, auch *Instanzbereich* genannt, und Ausgaben, auch *Antworten* genannt. Des Weiteren existiert eine Abbildung von Instanzen auf Antworten.

Instanzen können dabei beliebige Datenstrukturen sein. Bei klassischen Rechnern sind Binärzahlen beliebte Instanzen. Viele Entscheidungsprobleme werden hingegen als Sprache modelliert. Dabei ist ein Entscheidungsproblem Teil einer Sprache, wenn es zu „1" auswertet. Ein Entscheidungsproblem kann somit als eine Menge von „Ja"-Instanzen verstanden werden. Wichtig ist, dass Probleme in einer Datenstruktur kodiert werden müssen.

Gewisse Problemarten lassen sich effizienter von Self-Assembly-Systemen berechnen als andere. Die folgenden Problemdefinitionen beschränken sich auf die für Self-Assembly-Systeme wichtigen Aspekte.

DEFINITION 2.12 – Sei $\mathcal{D} = \Sigma^*$ eine *Instanzmenge*, auch Sprache genannt. Ein *Entscheidungsproblem* ist eine Teilmenge von Σ^*, welche zu „1" auswertet.

Entscheidungsprobleme sind für DNA-basierte Systeme von besonderem Interesse, da die Information der Lösung eines solchen Problems direkt durch eine Bindungsreaktion kommuniziert werden kann. Es sind folglich keine zusätzlichen Interpretationen notwendig.

Ein Beispiel für ein Entscheidungsproblem ist, ob für eine Krankheit signifikant viele Marker vorhanden sind oder im einfachen, mathematischen Fall, ob eine Zahl gerade oder ungerade ist.

DEFINITION 2.13 – Sei $\mathcal{D}$ eine Instanzmenge. Ein *Funktionsproblem* ist eine Abbildung von $\mathcal{D}$ nach $\mathcal{D}$.

Das Berechnungsergebnis eines Funktionsproblems kann nicht durch ein einziges Bit kommuniziert werden. Die Antwort auf ein Problem kann beliebig viele Bits umfassen. Ein Beispiel für ein Funktionsproblem ist die Addition zweier Zahlen. Dabei ist $\mathcal{D} = \mathbb{N}$.

Funktionsprobleme können leicht von Self-Assembly-Systemen berechnet werden, wie in den folgenden Abschnitten demonstriert wird. Allerdings ist unklar, wie eine errechnete Information auf Nanoebene kommuniziert werden kann.

2.4.2 LANDAU-NOTATION

Die Landau-Notation wird in der Theoretischen Informatik unter anderem verwendet, um das Grenzwertverhalten von Funktionen zu beschreiben [149]. Im Kontext der Arbeit wird die Landau-Notation vor allem genutzt, um das asymptotische Verhalten von Self-Assembly-Prozessen sowie deren Größe und Komplexität zu analysieren und zu beschreiben. Die Landau-Notation wird auch eingesetzt, um die Laufzeit von Algorithmen zu beschreiben.

D.E. Knuth prägte im Jahre 1976 die O-Notation, wie sie in der modernen Informatik verwendet wird [89].

DEFINITION 2.14 – Sei f eine Funktion über den reellen Zahlen $x \in \mathbb{R}$ und $a \in \mathbb{R} \cup \{-\infty, +\infty\}$. Für $f \in \mathcal{O}(g)$ gilt:

$$\limsup_{x \to a} \left| \frac{f(x)}{g(x)} \right| < \infty$$

Die Limesschreibweise besagt, dass das Wachstum der Funktion f durch das Wachstum der Funktion g beschränkt ist.

Die O-Notation ermöglicht es, eine beliebige Funktion einer Menge ähnlich schnell wachsender Funktionen zuzuordnen. Eine Funktion $f \in \mathcal{O}(1)$ überschreitet beispielsweise einen konstanten Wert nicht. $f \in \mathcal{O}(\log n)$ steigt um einen

Konstanten Wert c, wenn sich n verdoppelt. $f \in \mathcal{O}(n)$ wächst in etwa so schnell wie n. $f \in \mathcal{O}(n^2)$ vervierfacht sich bei einer Verdoppelung von n.

2.4.3 KOMPLEXITÄTSTHEORIE FÜR NANONETZWERKE

2008 haben Akyldiz et al. erstmals Nanonetzwerke als Kommunikations- und Rechenparadigma auf Nanoebene vorgeschlagen [26]. Die Fertigkeiten einzelner Nanoknoten sind dabei wenig spezifiziert worden. Es kann vorausgesetzt werden, dass wenigstens eine Möglichkeit zur Kommunikation bestehen muss, da andernfalls kein Netzwerk aufgebaut werden kann. Daraus geht hervor, dass koordinierte Aktuatorik und Sensorik und damit rudimentäre Möglichkeiten zur Informationsverarbeitung vorhanden sein müssen, um überhaupt sinnvolle Anwendungen zu ermöglichen.

Es bleibt jedoch unklar, wie die genannten Technologien realisiert werden können. Die geringe Größe der Nanoknoten impliziert starke Ressourcenbeschränkungen. Dies bezieht sich sowohl auf Rechenleistung als auch auf Speichergrößen.

Leider ist für zahlreiche Szenarien kaum formalisiert worden, wie komplex einzelne Nanoknoten sein müssen, um gegebene Problemstellungen bewältigen zu können [125, 165]. Die vorausgesetzten Fertigkeiten von Nanorobotern variieren stark. Teilweise ist von Nanopartikeln mit verschwindend geringer Komplexität die Rede. In anderen Publikationen wird von Nanorobotern geschrieben, welche eine ähnliche Mächtigkeit wie heutige Prozessoren haben [53].

Viele Szenarien erläutern nur sehr abstrakt, was zur Lösung eines Problems getan werden muss. Manchmal sind Formeln, abstrakte Operationen oder Algorithmen angegeben, jedoch ist die Mächtigkeit eines Nanoknotens oft nicht spezifiziert. Manchmal wird sie gar überschätzt. Genauso sind die Konsequenzen der benötigten Mächtigkeit auf die Realisierbarkeit der Maschinen nicht Teil der Ausarbeitungen.

Wenig komplexe Nanoknoten sind vermutlich einfacher zu realisieren als komplexe. Einfache, potenzielle Bestandteile von Nanogeräten wurden bereits realisiert [32, 41]. Die Kombination jener Bausteine zu komplexen Strukturen, wie bei modernen CPUs, ist jedoch noch weitestgehend unerforscht und diverse Fehlerquellen, wie zum Beispiel die präzise Platzierung von Komponenten, gestalten Forschungsfortschritte schwierig. Denn obwohl CPUs immer kleiner werden, füllt die Kombination aller Elemente einer CPU mehrere Quadratzentimeter aus.

Dies führt zu der Frage, was die geringstmöglichen Anforderungen an einzelne Nanoknoten sind, damit sie für ein gegebenes Szenario von Nutzen sind. Anders formuliert: Was ist die untere Schranke der benötigten Komplexität eines Nanoknotens, damit alle in einem Szenario gestellten Anforderungen erfüllt werden können? Ein sehr einfacher Nanoknoten mag zwar leicht realisierbar sein, könnte jedoch wichtige Fertigkeiten missen.

In [5] wurde diese Problemstellung bearbeitet, um eine Aussage über die Realisierbarkeit von vorgeschlagenen Szenarien zuzulassen. Außerdem ermöglicht

eine genaue Festlegung des Maschinenmodells eine einfachere und präzisere Simulierbarkeit. Dies wird in Kapitel 4 im Detail erklärt.

2.4.3.1 REDUKTIONEN

Damit Anforderungen an Nanogeräte ermittelt werden können, müssen Probleme aus vorgeschlagenen Szenarien entsprechend ihrer Komplexität sortiert werden. Dafür werden *Komplexitätsklassen* verwendet. Diese sind ein Konzept aus der Theoretischen Informatik – genauer gesagt aus der Komplexitätstheorie. Der Begriff Komplexität beschreibt dabei Problematiken, welche über schlichte Kompliziertheit hinausgehen. *Kompliziertheit* kann dabei als "schwierig" oder "aufwendig" zu beschreiben interpretiert werden, wohingegen *Komplexität* Sachverhalte beschreibt, denen zwar einfache Regeln zugrunde liegen können, deren Verhalten jedoch durch wiederholte Anwendung oder große Eingaben schwer oder unvorhersehbar wird.

Die hier gegebene Kurzeinführung bezieht sich auf [5] von Florian Lau, Florian Büther und Bennet Gerlach. Florian Lau hat alle inhaltlichen Beiträge geleistet, Literatur analysiert und die Kategorisierung in Komplexitätsklassen vorgenommen. Für ein tiefergreifendes Verständnis wird dem interessierten Leser die Lektüre von [15] nahegelegt.

Die Komplexität von Systemen wird in Abhängigkeit eines Systemparameters wie zum Beispiel dem zur Verfügung stehenden Speicher oder der für eine Operation benötigten Laufzeit bestimmt. Für diese Arbeit sind vornehmlich Komplexitätsklassen von Interesse, die sich auf den zur Verfügung stehenden Arbeitsspeicher beziehen, auch *Platzklassen* genannt. Im klassischen Rechenmodell, der einfachen Turingmaschine, ist davon unbegrenzt viel vorhanden. Das ist keine realistische Annahme für Nanogeräte, weswegen an dieser Stelle nur jene Komplexitätsklassen vorgestellt werden, welche extrem platzbeschränkt sind.

Einfach gesagt enthalten Komplexitätsklassen Probleme, welche durch ein bestimmtes Maschinenmodell gelöst werden können. Beispielsweise enthält die Komplexitätsklasse L all jene Probleme, die durch eine Turing-Maschine mit logarithmisch platzbeschränktem Arbeitsband gelöst werden können. Eingabe und Ausgabe werden nicht berücksichtigt. Dabei bezieht sich das logarithmische Verhältnis auf die Länge der kodierten Eingabe unter Berücksichtigung von Faktoren, welche in der Landau-Betrachtung wegfallen. Ein Nanogerät, welches wenigstens so mächtig ist wie ein bestimmtes Maschinenmodell, kann also wahrscheinlich auch alle Probleme der jeweiligen Klasse lösen. Im Falle von Schaltkreisen muss es wenigstens unter ähnlichem Aufwand möglich sein ein Nanogerät zu erzeugen, welches die verwandten Probleme löst.

Neben der Klasse L werden außerdem zwei weitere Schaltkreisklassen untersucht, welche für Nanogeräte interessant sein können. Schaltkreise sind von besonderem Interesse, da sie ohne Speicher Berechnungen durchführen können. Manche Veröffentlichungen befassen sich bereits mit schaltkreisähnlichen Systemen [106]. Diese Schaltkreise bestehen im Wesentlichen aus INV-, OR- und AND-Gattern. Diese drei Gattertypen bilden zusammen mit den Konstanten

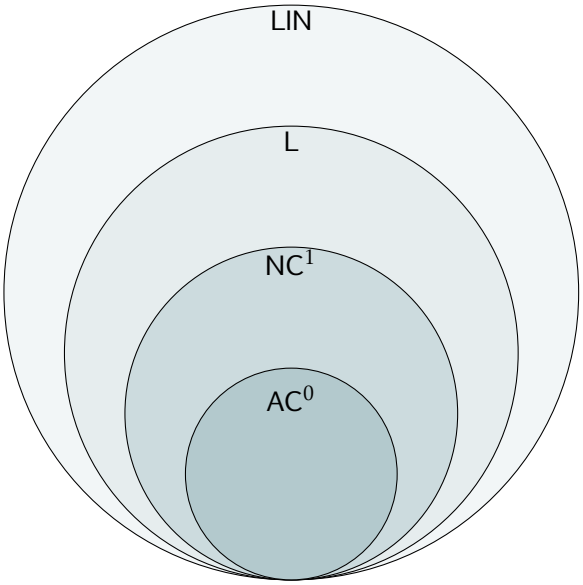


ABBILDUNG 2.11: Verhältnisse zwischen verschiedenen Platzklassen: LIN, L, NC^1 und AC^0.

die atomaren Komponenten der Klassen AC^0 und NC^1 [15]. Die Klasse AC^0 ist zusätzlich dadurch gekennzeichnet, dass boolesche Schaltkreise, welche durch sie beschrieben werden, polynomielle Größe und konstante Tiefe haben. Die Metriken gelten im Verhältnis zur Anzahl der Eingaben. NC^1-Schaltkreise haben zusätzlich die Einschränkung, dass jedes Gatter nur maximal zwei Eingänge haben darf. Sie dürfen dafür jedoch logarithmisch tief sein.

Komplexitätsklassen können einander enthalten. Beispielsweise enthält die Klasse NC^1 die gesamte Klasse AC^0, welche wiederum beide in L enthalten sind. Das Inklusionsverhältnis wird in Abbildung 2.11 verdeutlicht. Die Inklusion ist einfach zu sehen, da AC^0-Gatter mit beliebig vielen Eingängen durch eine logarithmisch tiefe Baumstruktur ersetzt werden können, welche das gleiche Ergebnis liefert.

Dadurch ist klar, dass Probleme in vielen Klassen liegen können. Das Problem LOG_2 ist z. B. durch AC^0-Schaltkreise und durch L-Turingmaschinen lösbar.

Wenn ein Nanogerät wenigstens so mächtig wie eine L-Turingmaschine ist, so kann es folglich ebenfalls das Problem LOG_2 lösen.

Um ein Problem oder ein Nanogerät in Komplexitätsklassen zu sortieren bzw. sie ihnen zuzuweisen, verwendet man *Reduktionen*. Abbildung 2.12 verdeutlicht das Vorgehen anhand einer schematischen Abbildung.

Eine Reduktion beweist, dass ein Problem A höchstens so schwer zu lösen ist wie ein anderes Problem B.

A reduziert sich auf B, wenn folgendes gilt:

Eine Instanz i_A von einem Problem A kann unter geringem Aufwand in eine Instanz i_B von einem Problem B umgewandelt werden. Die Instanz i_B wird dann

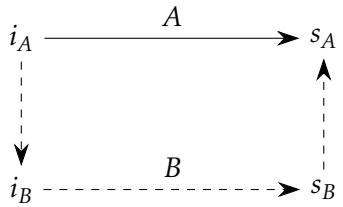


ABBILDUNG 2.12: Darstellung eines Reduktionsschemas. Eine Probleminstanz i_A wird in eine andere i_B umgewandelt, welche von einem Maschinenmodell B, welches wir bereits kennen, gelöst werden kann. Das Ergebnis der Berechnung s_B wird dann in ein für Maschine A verwertbares Ergebnis s_A transformiert.

durch einen "bekannten" Algorithmus B gelöst, sodass sich die Lösung s_B ergibt. Diese Lösung kann dann unter geringem Aufwand in eine Lösung s_A für das Problem A umgewandelt werden.

Beispielsweise kann man zeigen, dass das Subtraktionsproblem SUB höchstens so schwer ist wie das Additionsproblem ADD – also in der Klasse AC^0 liegt – indem man SUB auf ADD reduziert. Um das zu tun, verändern wir die Eingabe für SUB so, dass sie zu einer Eingabe für ADD wird. Dies kann durch ein Negationsgatter beim zweiten Summanden geschehen. Der veränderte Schaltkreis ist immer noch von konstanter Tiefe und verwendet die erlaubten Gatter der Klasse AC^0. Die Rücktransformation wäre in diesem Fall die Identitätsfunktion. Es ist wichtig, dass der kombinierte Aufwand für Hin- und Rücktransformation nur unter der Verwendung erlaubter Mittel (oder weniger) für die Klasse von Problem A geschieht. Andernfalls könnte man Probleme nur durch die Transformation alleine lösen.

Im folgenden Abschnitt werden aufbauend auf den komplexitätstheoretischen Grundlagen verschiedene Berechnungsmodelle vorgestellt.

2.5 ABGRENZUNG VON BERECHNUNGSMODELLEN AUF NANOEBENE

Die folgenden Abschnitte präsentieren die Berechnungsmodelle Schaltkreise, chemische Reaktionsnetzwerke, Quantum-Dot zelluläre Automaten und Self-Assembly-Systeme. Diese werden in ausreichender Tiefe erklärt, um begründen zu können, warum sie in der vorliegenden Arbeit genauer untersucht oder ausgelassen werden. Das Schlüsselkriterium für die Verwendbarkeit eines Berechnungsmodells im Rahmen der Arbeit ist die physikalische Realisierbarkeit.

Die Suche nach neuen Berechnungsmodellen auf der Nanoebene ist von Interesse, da angenommen wird, dass eine weitere Miniaturisierung auf Basis der CMOS-Technologie nur begrenzt funktioniert. Herkömmliche Transistortechnologien sind bereits ca. 7 nm klein [179].

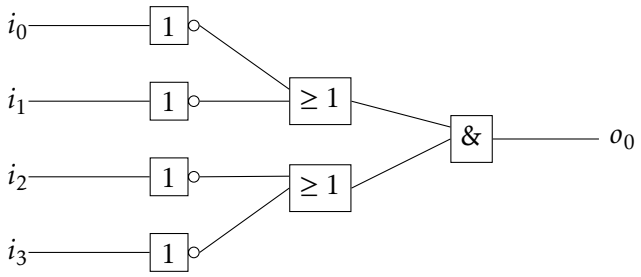


ABBILDUNG 2.13: Ein Beispiel für einen booleschen Schaltkreis mit vier Eingaben und einer Ausgabe.

2.5.1 SCHALTKREISE

Schaltkreise sind der de-facto-Standard bei physikalisch implementierbaren Berechnungsmodellen. Diese werden üblicherweise auf Basis von *komplementärern Metall-Oxid-Halbleitern* (CMOS) gefertigt. Viele Berechnungsmodelle in der Forschung teilen Eigenschaften mit Schaltkreisen oder emulieren diese vollständig.

Um zu verstehen, wie Nanogeräte Berechnungen anstellen, kann es hilfreich sein, mit dem Schaltkreismodell vertraut zu sein. Ein Beispiel ist in Abbildung 2.13 dargestellt. Der Schaltkreis bekommt gewisse Informationen in Form von Bits (hohe oder niedrige Spannung) geliefert, transformiert diese auf geeignete Art und Weise und gibt am Ende ein Ergebnis zurück, welches für eine einfache Ja/Nein-Entscheidung stehen kann. Dieses Verhalten könnte für Nanogeräte genutzt werden. Ein Nanogerät könnte Medikamente mit sich führen und diese freisetzen, sobald der korrekte Ort und die geeigneten Marker dafür gefunden wurden.

Schaltkreise werden im Allgemeinen nicht genauer betrachtet, da diese bereits dem Ende ihrer Miniaturisierungsmöglichkeiten entgegen zu gehen scheinen [147].

2.5.2 CHEMISCHE REAKTIONSNETZWERKE

Chemische Reaktionsnetzwerke (CRN) modellieren das Verhalten gut gemischter Lösungen [44]. Ein CRN besteht dabei aus einer endlichen Menge chemischer Reaktionen. CRN können eingesetzt werden, um Berechnungen durchzuführen. In [151] wurde gezeigt, dass einige CRN Turing-vollständig sind. Für Berechnungen verwendete Reaktionsnetzwerke sind abstrakt. Die verwendeten Chemikalien haben kein reales Pendant. Für eine detaillierte Einleitung in das Thema empfiehlt sich [36]. Die folgende Terminologie basiert auf den dort vorgestellten Inhalten.

Mathematisch besteht ein CRN aus einer endlichen Menge *Reaktionen* $\mathcal{R}_i$ mit $i \in \mathbb{N}^+$ und einer Menge von chemischen Bausteinen S_j mit $j \in \mathbb{N}^+$, auch *Spezies* genannt. Eine Reaktion ist dabei folgendermaßen spezifiziert:

$$\mathcal{R}_i : \sum_{j \in S} \alpha_{ij} S_j \mapsto \sum_{j \in S} \beta_{ij} S_j$$

Dabei sind $\alpha_{ij}, \beta_{ij} \in \mathbb{N}$. Die Komponenten auf der linken Seite der Reaktion werden *Reaktanten* genannt. Die Komponenten der rechten Seite der Reaktion werden *Produkte* genannt. Eine Reaktion kann nur in Pfeilrichtung stattfinden. Sowohl Reaktanten als auch Produkte dürfen $\emptyset$ sein.

Das folgende Beispiel zeigt ein mögliches CRN. Gegeben seien die sechs Spezies E, F, P, S, ES, EP und die Reaktionen:

$$E + S \leftrightarrow ES \rightarrow E + P$$

$$F + P \leftrightarrow FP \rightarrow F + S$$

Die zwei Gleichungen kodieren insgesamt sechs Reaktionen. „$\leftrightarrow$" bedeutet, dass für eine Reaktion ebenfalls die inverse Reaktion existiert. Existieren mehrere Pfeile in einer Gleichung, so werden mehrere Reaktionen auf einmal durch eine transitive Beziehung dargestellt.

Nun kann aus den genannten Komponenten und Reaktionen ein korrespondierendes Petrinetz erzeugt werden. Dabei wird für jede Spezies ein Knoten erzeugt. Entsprechend der Reaktionen können nun Kanten erzeugt und mit Transaktionen versehen werden, welche den Koeffizienten α_{ij}, β_{ij} entsprechen.

Durch die Transaktionsregeln des Petrinetzes kann bedingtes Verhalten implementiert werden, was wiederum beliebige Berechnungen erlaubt.

Berechnungen können so z. B. in einem Reagenzglas durchgeführt werden. Der Prozess ist jedoch langwierig und das Auslesen von Ergebnissen ist mit erheblichem makroskopischen Aufwand verbunden. Es ist zudem schwer vorstellbar, dass CRN in einem Umfeld wie dem menschlichen Körper eingesetzt werden können, da eine große Zahl reagierender Substanzen mit hoher Wahrscheinlichkeit entweder natürliche Prozesse im Körper stört oder von Ihnen gestört wird.

Außerdem stellt die Einführung abstrakter Spezies bei realen Implementierungen ein Problem dar. Diese verfügen über kein reelles Pendant und es ist nicht klar, dass es immer einen real existierenden Ersatz gibt.

Da der Fokus der Arbeit auf einer zeitnahen Umsetzbarkeit der vorgestellten Modelle liegt, werden CRN nicht genauer untersucht.

2.5.3 QUANTUM-DOT ZELLULÄRE AUTOMATEN

In diesem Abschnitt werden die Grundlagen von Quantum-Dot zellulären Automaten vorgestellt. Da es sich bei Quantum-Dot zellulären Automaten um eine Form von zellulären Automaten handelt, werden diese zuerst definiert und Quantum-Dot zelluläre Automaten daraus abgeleitet.

Quantum-Dot zelluläre Automaten (QCA) verwenden Quantum-Dot-Zellen als elementare Komponenten. Eine Quantum-Dot-Zelle kann exakt einen von zwei möglichen Zuständen annehmen. Die Zustände werden dabei über die Positionen von Partikeln in einem abgegrenzten Bereich bestimmt. Die erste Quantum-Dot-Zelle wurde 1997 physikalisch realisiert [32].

Das Modell erscheint vorteilhaft für die Anwendung auf Nanoebene, da die Größe einer Quantum-Dot-Zelle nur wenige Nanometer beträgt. Außerdem nutzen Quantum-Dot-Zellen genau jene Effekte, welche CMOS an einer weiterer Miniaturisierung hindern könnten. Liegen Drähte beispielsweise zu nah beieinander könnten Elektronen zwischen ihnen tunneln.

QCA haben folglich ein größeres Potential für die zukünftige Entwicklungen [16]. QCA werden detaillierter betrachtet, da sie auf den ersten Blick eine sinnvolle Alternative zu bewährten Technologien darstellen.

2.5.3.1 ZELLULÄRE AUTOMATEN

Dieser Abschnitt präsentiert mathematische Definitionen für zelluläre Automaten. Diese basieren auf der von Wolfram eingeführten Terminologie [136]. Die verwendete Notation geht auf Buss zurück [10].

DEFINITION 2.15 – Der Zustand eines zellulären Automaten zum Zeitpunkt $t \in \mathbb{N}$ ist ein unendlich großes, d-dimensionales *Gitter* L^t mit $d \in \mathbb{N}^+$. Eine Zelle L_v im Gitter ist durch einen Vektor $v \in \mathbb{Z}^d$ indiziert. Jede Zelle L_v kann einen beliebigen Zustand L_v^t einer endlichen Menge von Zuständen zum Zeitpunkt t annehmen. Das Zustandsalphabet wird S genannt. Ein Gitter ist ein Element aus $L^t \in S^{(\mathbb{Z}^d)}$.

Wolfram hat maßgeblich ein- und zweidimensionale zelluläre Automaten untersucht [136, 108]. Damit man Drähte in allen Situationen übereinander platzieren kann, ist eine dritte Dimension erforderlich.

DEFINITION 2.16 – Seien $v \in \mathbb{Z}^d$. Sei N eine Funktion $N : \mathbb{Z}^d \to (\mathbb{Z}^d)^n$, sodass $N(v)$ eine Sequenz von Zellindizes der Länge n liefert. $N(v)$ wird die *Nachbarschaft* einer Zelle L_v genannt.

Dies ist eine sehr abstrakte Definition. Für eindimensionale Gitter mit $v \in \mathbb{Z}$ hat Wolfram die Nachbarschaft einer Zelle L_v durch $N(v) = [v - r, v + r]$ definiert, wobei $r \in \mathbb{N}$ ein beliebiger Radius ist.

Für zweidimensionale Gitter sind die verbreitetsten Nachbarschaftsfunktionen jene, die die Nachbarn in den Himmelsrichtungen berücksichtigen und jene, die ebenfalls alle Nachbarn auf den Diagonalen einbeziehen. Abbildung 2.14 (a) und (b) zeigen die jeweiligen Beispiele.

Man kann leicht erkennen, dass sich die Definition ebenfalls auf drei Dimensionen erweitern lässt. Die Erweiterungen berücksichtigen im ersten Fall zusätzlich

(a) (b)

ABBILDUNG 2.14: Die zwei verbreitetsten Nachbarschaftsfunktionen. Der Zustand der Zelle in der Mitte wird unter Berücksichtigung aller angrenzenden Zellen berechnet.

die Zellen in der dritten Dimension darüber und darunter. Im zweiten Fall die weiteren 18 Nachbarn.

DEFINITION 2.17 – Seien $N(v)$ die Indizes der Nachbarn von L_v. Zu jedem diskreten Zeitpunkt t wird der Zustand einer Zelle L_v aktualisiert. Der neue Zustand hängt ausschließlich von den Nachbarn von L_v zum Zeitpunkt t ab:

$$L_v^{t+1} = \psi(L^t_{N(v)_1}, L^t_{N(v)_2}, \ldots, L^t_{N(v)_n})$$

für eine Funktion $\psi : S^n \to S$.

Da $N(v)$ eine Sequenz von Indizes beschreibt, ist der nächste Zustand der Zelle nicht nur von den Werten der Nachbarzellen abhängig, sondern auch von ihren Indizes. Jede Nachbarzelle kann also unterschiedlich gewichtet werden.

DEFINITION 2.18 – Eine *lokale Transitionsfunktion* ψ ermöglicht eine globale Transitionsfunktion Ψ. Sei L^t der Zustand des Automaten zum Zeitpunkt t. Der Gitterzustand ist wie folgt definiert:

$$L^{t+1} = \Psi(L^t),$$

wobei der Zustand L_v^{t+1} jeder Zelle L_v in L^{t+1} durch Applizieren der Funktion ψ bestimmt wird.

Der Zustand L^t eines zellulären Automaten L zum Zeitpunkt $t \in \mathbb{N}^+$ wird vollständig durch den Initialzustand L^0 festgelegt.

DEFINITION 2.19 – Seien $d, n \in \mathbb{N}^+$. Sei S eine endliche Zustandsmenge. Sei $L^0 \in S^{(\mathbb{Z}^d)}$ ein Initialzustand. Sei N eine Funktion $N : \mathbb{Z}^d \to (\mathbb{Z}^d)^n$ und sei ψ eine Funktion $\psi : S^n \to S$. Ein *zellulärer Automat* ist ein Tupel $L = (d, n, S, L^0, N, \psi)$.

Nun können Modifikationen vorgenommen werden, um QCA als Spezialfall von zellulären Automaten zu beschreiben.

Intuitiv kann eine Quantum-Dot-Zelle als Zelle eines zellulären Automaten verstanden werden. Jedoch muss das Modell um einen Taktgeber erweitert

werden, da der Signalfluss sonst kaum zu kontrollieren ist. Zu jedem Zeitpunkt soll jeweils nur eine Untermenge aller Zellen betrachtet werden. Abbildung 2.15 erklärt die Funktionsweise einer Quantum-Dot-Zelle.

DEFINITION 2.20 – Eine *clock* koordiniert die Zustandswechsel einer Quantum-Dot-Zelle und wird durch folgende Phasen beschrieben, welche sich in angegebener Reihenfolge unendlich wiederholen:

- In der relax-Phase ist das Signal stark und die Tunnelübergänge sind geöffnet. Elektronen können sich völlig frei bewegen.
- In der switch-Phase wird das Signal gesenkt und die Tunnelübergänge werden geschlossen. In dieser Phase finden die eigentlichen Berechnungen statt.
- In der hold-Phase ist das Signal schwach und die Tunnelübergänge geschlossen – Elektronen sind folglich an ihre Positionen gebunden. In dieser Phase nimmt eine Zelle einen der möglichen Zustände an.
- In der release-Phase wird das Signal verstärkt und die Tunnelübergänge werden geöffnet. Elektronen können anfangen sich frei zu bewegen.

In der hold-Phase ist eine Zelle fest in einem Zustand verankert. Einer der Zustände „0" oder „1" wird angenommen. Zusätzlich wird aus technischen Gründen ein neutraler Zustand benötigt. Dieser wird ϵ genannt.

2.5.3.2 INSTANZIIERUNG VON QCA

Im folgenden Abschnitt werden die allgemeinen Definitionen von zellulären Automaten verwendet, um die notwendigen Konkretisierungen für QCA abzuleiten.

- *Gitter*: Die Dimension des Gitters ist $d = 3$, da die dritte Ebene für Drahtübergänge benötigt wird. Des Weiteren ist die Anzahl der Quantum-Dot-Zellen endlich. Maximal drei Schichten sind zulässig. Alle L_v^t mit $v \notin \mathbb{Z} \times \mathbb{Z} \times \{0, 1, 2\}$ sind leer. Ein Gitter mit Quantum-Dot-Zellen, welches eine boolesche Funktion berechnet, wird auch *Chip* genannt.
- *Alphabet*: Das Alphabet eines QCA besteht aus $(p, t) \in S = \{-1, 0, 1\} \times C$ mit $C = \{\text{relax}, \text{switch}, \text{hold}, \text{release}, \epsilon\}$. p beschreibt die Polarisierung einer Zelle. „0" entspricht einer unpolarisierten oder keiner Zelle. Der zweite Teil des kartesischen Produktes beschreibt die Clock C. Eine Clock-Zone ϵ beutetet, dass niemals Änderungen eintreten werden. Die Clock-Phase einer Zelle beschreibt gleichzeitig die Zone.
- *Nachbarn*: Die Nachbarschaft $N(v)$ einer Zelle mit Index v kann frei gewählt werden. Je größer sie ist, desto realitätsgetreuer ist der QCA. Weiter entfernte Nachbarn haben einen deutlich geringeren Einfluss.

  $N(v)$ sollte wenigstens aus folgenden Elementen bestehen: $\mathcal{N}(v)$ ist die Menge aller Elemente $N(v)$:

$$\begin{aligned}\mathcal{N}((v_1, v_2, v_3)) \supseteq \{&(v_1 + 1, v_2, v_3), (v_1 - 1, v_2, v_3), (v_1, v_2 + 1, v_3), (v_1, v_2 - 1, v_3),\\ &(v_1 + 1, v_2 + 1, v_3), (v_1 + 1, v_2 - 1, v_3), (v_1 - 1, v_2 + 1, v_3),\\ &(v_1 - 1, v_2 - 1, v_3), (v_1, v_2, v_3 + 1), (v_1, v_2, v_3 - 1)\}\end{aligned}$$

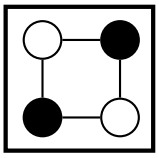


ABBILDUNG 2.15: Eine Quantum-Dot-Zelle mit als Linien dargestellten Tunnelübergängen.

Die ersten vier sind die Nachbarn in Himmelsrichtung. Danach folgen die Diagonalen. Die letzten beiden sind Zellen ober- und unterhalb von v.

- *Zustandstransitionsfunktion*: Die Transitionsfunktion muss die Clockzonen berücksichtigen. Dafür wird folgende Funktion verwendet:

$$\text{next} : C \to C$$

$$\text{next}(t) = \begin{cases} \text{switch} & \text{wenn } t = \text{relax} \\ \text{hold} & \text{wenn } t = \text{switch} \\ \text{release} & \text{wenn } t = \text{hold} \\ \text{relax} & \text{wenn } t = \text{release} \\ \epsilon & \text{sonst} \end{cases}$$

Eine Zelle in der switch-Phase wird gerade vollständig polarisiert. Da die Nachbarschaft unklar ist, kann ψ nur in Abhängigkeit der Nachbarschaftswahl angegeben werden. L_v^{t+1} ist wie folgt definiert, wobei $L_v^t = (p, c)$:

$$L_v^{t+1} = \begin{cases} (p, c) & \text{wenn } c = \epsilon \\ (\psi(N(v), t), \text{hold}) & \text{wenn } c = \text{switch} \\ (0, \text{next}(c)) & \text{sonst} \end{cases}$$

Das Verhalten dieses Clockings ist identisch zu dem von Braun [9].

Eine Zelle L_v kann man sich, wie in Abbildung 2.15 dargestellt, vorstellen. Zwei Elektronen sind in einem Gitter gefangen. In jeder der vier Ecken befindet sich ein Quantum-Dot. Diese sind über Tunnelübergänge verbunden. Aufgrund von elektrischen Abstoßungsreaktionen befinden sich Elektronen immer in gegenüberliegenden Ecken, was zwei mögliche logische Zustände zulässt. Diese sind in Abbildung 2.16 dargestellt. Elektronen stoßen sich durch ihre negative Ladung gegenseitig ab. Dieser Effekt tritt auch über Zellen hinweg auf, weswegen man mit Quantum-Dot-Zellen rechnen kann. Man berücksichtige, dass die Zellen in den Abbildungen in der Realität deutlich andere Abstände aufweisen können.

Geschickt platziert lassen sich aus den Zellen logische Gatter formen. Die elementaren Komponenten sind hierbei Majoritätsgatter und Inverter. Zusammen mit konstanten Zellen lässt sich eine Turing-vollständige Logik bilden. Abbildung 2.17 zeigt ein Majoritätsgatter. Die Elektronen versuchen in den Zustand zu wechseln, der die geringste Energie benötigt. Das führt zum Verhalten einer Majoritätsfunktion auf drei Bits.

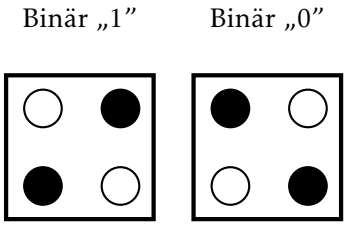


ABBILDUNG 2.16: Zwei Quantum-Dot-Zellen in unterschiedlichen Zuständen. Diese werden als logische „1” und „0” interpretiert.

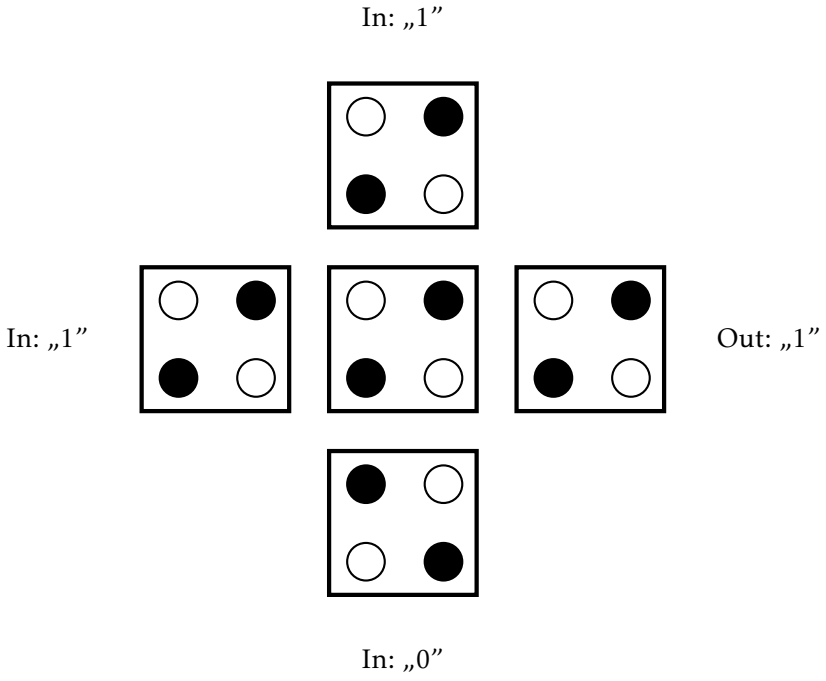


ABBILDUNG 2.17: Dargestellt ist ein einfaches Majoritätengatter aus Quantenzellen. Durch die Abstoßungsreaktion zwischen den Zellen versucht die Zelle in der Mitte das Maximum der angrenzenden Zellen anzunehmen.

Abbildung 2.18 zeigt einen Inverter. Es gilt derselbe Grundsatz wie beim Majoritätengatter. Der energetisch günstigste Zustand ist das Inverse der Eingabe.

2.5.3.3 TRANSITIONSFUNKTION

Eine Transitionsfunktion koordiniert die Weise auf die Zellen ihren Zustand wechseln. Eine mögliche Transitionsfunktion ψ auf Basis der vorherigen Definitionen kann wie folgt aussehen: Zwei Quantum-Dots einer Zelle sind 18 nm voneinander entfernt, der Abstand zwischen zwei Zellen beträgt 2 nm und Zellen in der dritten Dimension sind 11,5 nm voneinander entfernt. Die QCA-Designer Software verwendet dieselben Werte [177].

Der Einfluss von Diagonalzellen ist geringer als der Einfluss einer Zelle in den Himmelsrichtungen. Eine Zelle in der dritten Dimension hat einen höheren Ein-

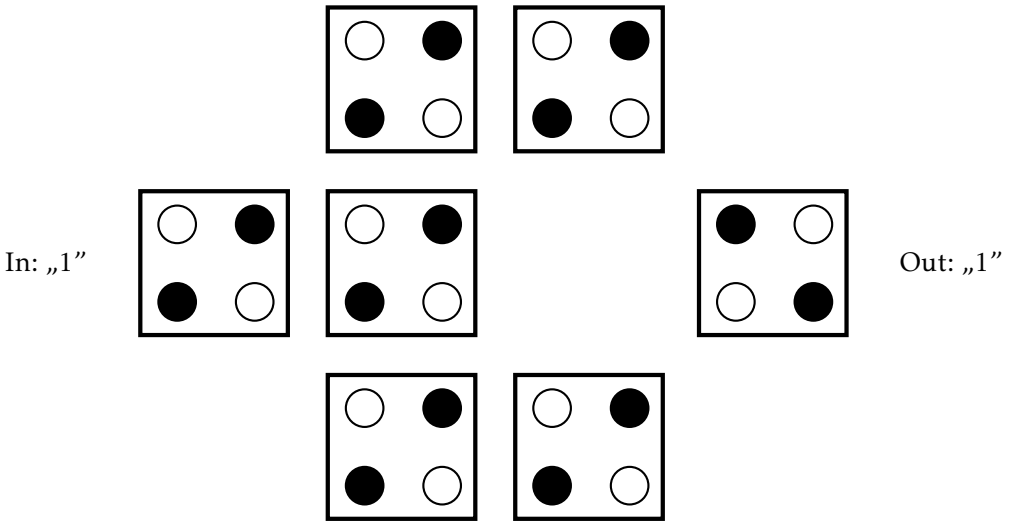


ABBILDUNG 2.18: Dargestellt ist ein Invertergatter. Durch Abstoßungsreaktionen zwischen den Elektronen nimmt die Ausgangszelle den invertierten Wert des Eingangs an.

fluss als eine Zelle in den Himmelsrichtungen. Die unterschiedlichen Einflüsse basieren auf den Abständen und der Anzahl der Elektronen zueinander. Die abstoßenden Kräfte nehmen dabei quadratisch mit dem Abstand ab.

DEFINITION 2.21 – Sei $\mathcal{N}(v)_d$ eine Menge an Indizes der diagonalen Nachbarn, sei $\mathcal{N}(v)_c$ die Menge der Indizes der Nachbarn in den Himmelsrichtungen und $\mathcal{N}(v)_z$ die Menge der Indizes der vertikalen Nachbarn einer Zelle L_v. Eine mögliche Transitionsfunktion ψ kann wie folgt aussehen:

$$\Psi(N(v),t) = 1 \cdot (\sum_{i \in \mathcal{N}(v)_c} L_i^t) - 0.2 \cdot (\sum_{i \in \mathcal{N}(v)_d} L_i^t) - 1.35 \cdot (\sum_{i \in \mathcal{N}(v)_z} L_i^t)$$

Die Funktion garantiert, dass nur „0" und „1" zulässige Zustände in der switch-Phase sind:

$$\psi(N(v),t) = \begin{cases} 0 & \text{wenn } \Psi(N(v),t) < 0 \\ 1 & \text{sonst} \end{cases}$$

Obwohl QCA ein vielversprechendes Berechnungsmodell auf der Nanoebene darstellen, existieren noch viele ungelöste Problemstellungen. Es wurden zwar einzelne Quantum-Dot-Zellen im Labor hergestellt, allerdings stellt z. B. deren präzise Platzierung eine Hürde dar. Große Mengen von Quantum-Dot-Zellen können zurzeit nicht industriell gefertigt werden.

Des Weiteren ist es notwendig, einen geeigneten Synchronisationsmechanismus wie Clocking zu verwenden, welcher in der Praxis ein ungelöstes Problem darstellt. Andernfalls lassen sich die Ergebnisse von Berechnungen nicht sinnvoll

verwerten. Ein Gitter von Quantum-Dot-Zellen kann ohne Clocking chaotisch wirken.

Zusätzlich besteht die Notwendigkeit, Signale nach einer gewissen Zeit zu verstärken, was *Repeater* oder eine ähnliche Technologie erfordert. Signale werden mit zunehmender Distanz zu schwach, um eine Polarisierung von Nachbarzellen zu bewirken.

Ein weiteres Problem stellt die Realisierung von Speicher dar. Es existieren zwar Lösungsansätze, diese sind jedoch wenig ausgereift [161].

Zuletzt ist die Platzierung von Quantum-Dot Zellen in der dritten Dimension eine Herausforderung. Es ist nötig, Zellen über die Ebene hinaus zu platzieren, um Drahtübergänge realisieren zu können. Ohne besagte Übergänge können beispielsweise Gatter mit zahlreichen Eingängen nicht oder nur eingeschränkt umgesetzt werden.

Aus diesen Gründen werden QCA im folgenden Text nicht als Ausgangspunkt für aufbauende Forschung verwendet. Der Fokus der Arbeit liegt explizit auf momentan verfügbaren, effektiv umsetzbaren Technologien.

2.5.4 DNA-BASIERTE SELF-ASSEMBLY-SYSTEME

Neben der Intelligenz von Nanogeräten ist deren Konstruktion ein weiteres ungelöstes Problem. DNA-basierte Self-Assembly-Systeme können sowohl zum Rechnen als auch zum Konstruieren von Nanostrukturen eingesetzt werden.

Die Idee hinter Herstellungsprozessen auf Basis von DNA ist, dass man Nanogeräte wie Kristalle "wachsen" lässt. In dem Fall müsste man nur die Bausteine zur Verfügung stellen und warten. Da DNA ein im menschlichen Körper vorkommendes Molekül ist, kann man davon ausgehen, dass die Komponenten eher biologisch abbaubar sind als beispielsweise Strukturen auf Basis von Kohlenstoff-Nanoröhrchen oder Schwermetallen. [93] sagt voraus, dass DNA-basierte Technologien eine entscheidende Rolle in der nahen Zukunft spielen werden. Beispielsweise gibt es Experimente, welche die Tauglichkeit von DNA als Rahmenstruktur für nanogroße elektronische Komponenten überprüfen [143]. Diese Experimente nutzen oft eine gitterartige DNA-Struktur, in die andere Komponenten eingebettet werden.

Um Self-Assembly-Systeme in ihrer Gänze verstehen zu können, ist es notwendig, das Konzept von Kristallbildung oder Selbstzusammensetzung zu begreifen. Der Prozess findet praktisch auf allen Größenordnungen der Wirklichkeit statt. Atome setzen sich zu Molekülen und Moleküle zu noch komplexeren Strukturen oder sogar Lebewesen zusammen.

In lebenden Zellen besteht die DNA aus einer Doppelhelix, welche wiederum aus den Basen Adenin, Cytosin, Guanin und Thymin besteht. Bestimmten Regeln folgend formen diese Basen Bindungen miteinander. Tatsächlich setzen sich alle Lebewesen mehr oder weniger selbst zusammen und könnten einer recht allgemeinen Definition zufolge als "Kristalle" bezeichnet werden.

Ein anschauliches Beispiel für Selbstzusammensetzung stellen Schneeflocken dar (siehe Abbildung 2.19). Schneeflocken bilden sich bei niedrigen Tempera-

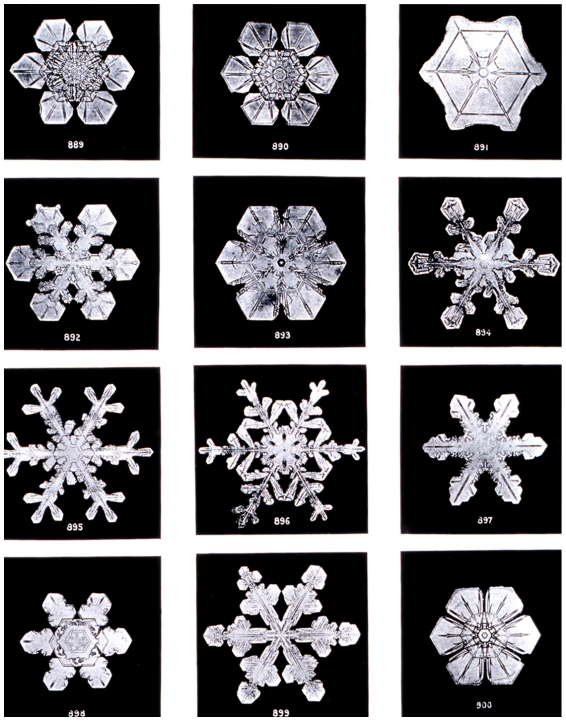


ABBILDUNG 2.19: Vergrößerte Ansicht verschiedener Schneeflocken (Kristalle), welche sich aus Wassermolekülen und Schmutz bei niedrigen Temperaturen in der Luft formen.

turen im Wesentlichen aus zwei Grundbausteinen. Dabei handelt es sich um Wassermoleküle und ein Staubkorn, welches den Kristallbildungsprozess startet. Im Laufe der Zeit bildet sich durch Hinzufügen weiterer Wassermoleküle eine kristalline Struktur.

Bei Nanogeräten ist der Prozess im Wesentlichen identisch – nur die Bausteine unterscheiden sich. Ein möglicher Baustein ist DNA. DNA-Bausteine sind im menschlichen Körper fast überall verfügbar und DNA hat die Tendenz sich auf eine bestimmte Weise zu strukturieren. Dieses Verhalten kann man ausnutzen, um gezielt Strukturen zu erzeugen. Abbildung 2.20 zeigt Beispiele für Nanostrukturen, welche bereits im Labor aus DNA erzeugt wurden. Es liegt nicht fern anzunehmen, dass der Prozess ausgenutzt werden kann, um Nanogeräte oder sogar Nanonetzwerke zu konstruieren [140].

Aufbauend auf der Idee, DNA für Berechnungen zu verwenden, wurden theoretische Modelle entworfen, mit denen am Computer getestet werden kann, wie DNA-Bausteine beschaffen sein müssen, damit die gewünschte Struktur aus DNA erwächst. Diese Systeme, auch *Self-Assembly-Systeme* genannt, sind jedoch zu weitaus mehr fähig, wie im Verlauf der Arbeit gezeigt wird. Sie können zur Lösung von drei Arten von Problemen eingesetzt werden:

1. Nanokommunikation,
2. Nanoberechnung und
3. Nanokonstruktion.

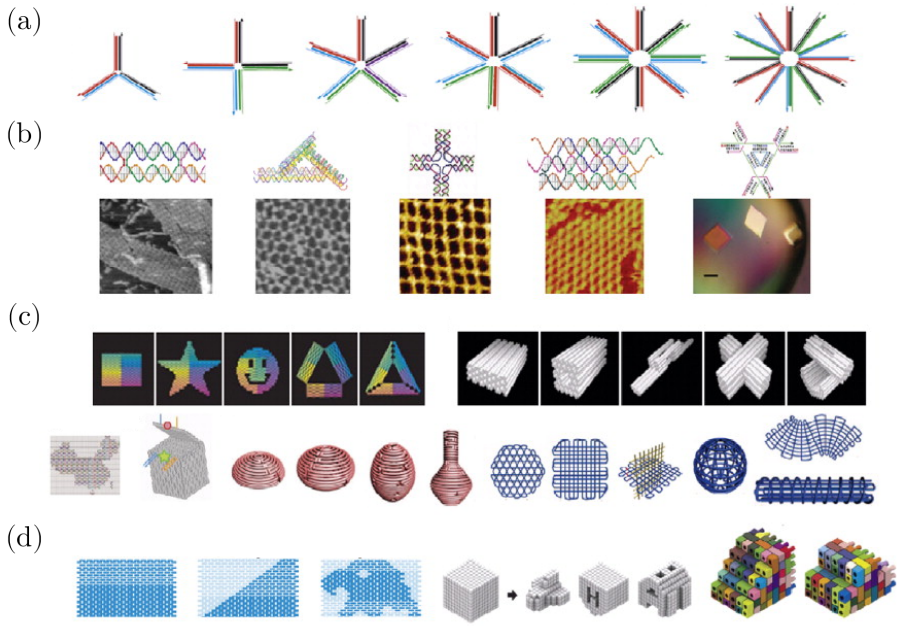


ABBILDUNG 2.20: Bereits erzeugte Nanostrukturen. [47]

Im Folgenden werden die notwendigen Definitionen vorgestellt, um Self-Assembly-Systeme für die beschriebenen Probleme einzusetzen. Dabei werden Self-Assembly-Systeme im Kontext dieser Arbeit sowohl als Berechnungsmodell als auch als Konstruktionsmechanismus verstanden. Die physikalische Implementierung geschieht dabei auf Basis der DNA-Strukturen aus Abschnitt 2.2. Self-Assembly-System-Implementierungen auf Basis anderer Baustoffe sind aber ebenfalls denkbar.

2.5.5 SELF-ASSEMBLY-SYSTEME FORMALISIEREN

Die folgenden Abschnitte definieren die für diese Arbeit wichtigsten mathematischen Modelle zum Thema Self-Assembly-Systeme. Zuerst wird das Wang-Tile-Modell erklärt. Dieses stellt den historischen Vorreiter für die folgenden Modelle dar. Danach erfolgt die Definition des kinetic Tile Assembly Modell. Dieses wurde erstmals von Eric Winfree in seiner Dissertation vorgestellt [135]. Es ist das am häufigsten verwendete Modell für die Simulation von realitätsnahen Self-Assembly-Prozessen. Es basiert auf dem ebenfalls von Winfree erdachten abstract Tile Assembly Modell, welches eine deutliche Vereinfachung darstellt und hauptsächlich zur Untersuchung theoretischer Fragestellungen eingesetzt wird.

Direkt im Anschluss wird das twohanded Tile Assembly Modell vorgestellt, welches eingesetzt werden kann, um parallel stattfindende Self-Assembly-Prozesse zu untersuchen.

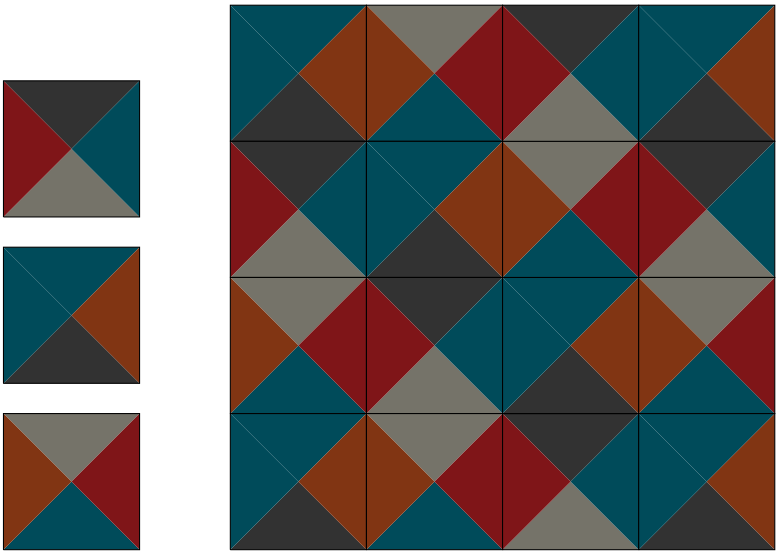


ABBILDUNG 2.21: Wang-Tiling einer zweidimensionalen Ebene.

2.5.5.1 WANG-TILES

Wang-Tiles sind die zugrundeliegende Datenstruktur, aus der Self-Assembly-Systeme hervorgegangen sind [178]. Abbildung 2.21 zeigt ein Beispiel. Links sind drei Wang-Tiles zu sehen, welche genutzt werden können, um eine Ebene komplett zu füllen, ohne dabei Lücken zu lassen.

Wang-Tiles können nur dann benachbarte Plätze im diskreten $\mathbb{Z}^2$-Raum einnehmen, wenn ihre Farben übereinstimmen. Die Farben der Wang-Tiles verhalten sich analog zu den im folgenden Abschnitt vorgestellten Klebern von Self-Assembly-Systemen – nur die Stärke eines Klebers ist nicht repräsentiert. Aufbauend auf Wang-Tiles werden nun die generellen Tiles definiert.

2.5.5.2 DEFINITIONEN FÜR SELF-ASSEMBLY-SYSTEME

Dieser Abschnitt präsentiert Definitionen und Beispiele für Self-Assembly-Systeme auf Basis von Eric Winfree [135]. Tiles sind die elementaren Komponenten von Self-Assembly-Systemen. Die Bindungsinteraktion von Tiles führt zur Bildung von Assemblies. Assemblies werden im Verlauf der Arbeit sowohl für die Modellierung von Nanosensoren und Nanorobotern als auch für die Modellierung von Nanonetzwerken, Nachrichtenmolekülen, Rezeptoren und Liganden eingesetzt.

Die im Folgenden verwendete Notation basiert auf [150]. Für eine detaillierte Definition von Self-Assembly-Systemen und ein Überblick der wichtigsten Ergebnisse zu Self-Assembly-Systemen kann [110] konsultiert werden.

DEFINITION 2.22 – Ein *n-dimensionales Tile* t_n ist ein Objekt im $\mathbb{Z}^n$ mit Einheitslänge und 90 Grad-Winkeln. Eine *Seite* eines Tiles t_n ist durch einen Vektor $u_t \in U_t \subseteq \mathbb{Z}^n$ definiert. U_t ist eine Menge eindeutiger Richtungsvektoren. Der

Vektor $-u_t$ beschreibt die gegenüberliegende Seite u_t in derselben Dimension. Der Vektor u_t hat genau einen Eintrag der ungleich 0 ist und den Wert 1 hat. Die Seiten eines Tiles sind durch folgende Relation definiert:

$$\text{Seite} : t_n \mapsto \underbrace{U_t \times U_t \cdots \times U_t}_{n}$$

Im Folgenden wird die Dimension größtenteils weggelassen. Wenn nicht anders angegeben, handelt es sich um dreidimensionale oder zweidimensionale Tiles.

Ein 2D-Tile t_2 ist ein Quadrat im $\mathbb{Z}^2$ und ein 3D-Tile t_3 ist ein Würfel im $\mathbb{Z}^3$. Abbildungen 2.22 (a)–(d) zeigen Beispiele für zweidimensionale und dreidimensionale Tiles. Es werden sowohl grafische Modellierungen mit Fokus auf die mathematischen Bestandteile als auch mit Fokus auf die biologischen Komponenten dargestellt.

Jede Seite eines Tiles kann eine beliebige Anzahl Kleber besitzen, welche durch eine Bindungsstärke modelliert werden:

DEFINITION 2.23 – Ein *Kleber* oder *Glue* $g \in G$, wobei G die Menge der möglichen Kleber beschreibt, ist durch ein *Label* $\mathcal{L}_g \in \Sigma^*$ definiert, wobei Σ ein Alphabet ist und $s \in \mathbb{N}$ eine Kleber-*Stärke* oder Kleber-*Strength* mit den Funktionen:

$$\begin{aligned} \text{label} &: G \mapsto \Sigma^* \\ \text{strength} &: G \mapsto \mathbb{N} \end{aligned}$$

Das Kleber-Label wird häufig *Farbe* genannt. Kleber werden durch offene DNA-Enden mit selbstbestimmten Basensequenzen umgesetzt. Dadurch lassen sich sowohl Farbe, bedingt durch den Einsatz bestimmter Basen, und Stärke (Länge/Anzahl der Sequenzen) umsetzen.

In der DNA-Computing-Forschungsgemeinschaft werden Tiles aus DNA hergestellt [110]. Ein Beispiel ist in Abbildung 2.23 (a) dargestellt. Die offenen Enden mit den eingetragenen Basen A, T, C, G können nur mit einer passenden Komplementärsequenz binden. Der Bindungsprozess ist in Abbildung 2.23 (b) dargestellt.

Jedes Tile kann nur genau einen Ort im diskreten, zweidimensionalen Raum $\mathbb{Z}^n$ belegen. Der Vektor $v_t \in \mathbb{Z}^n$ beschreibt die *Ort* genannten, Positionen des $\mathbb{Z}^n$.

DEFINITION 2.24 – Zwei Tiles t und t', die durch v_t und $v_{t'} \in \mathbb{Z}^n$ definierte Orte belegen, sind genau dann *benachbart*, wenn $|v_t - v_{t'}| = 1$ und der resultierende Vektor $e = v_t - v_{t'}$ genau ein Element ungleich 0 hat.

Ein Tile kann keinen oder genau einen Nachbar an jeder Seite haben. Tiles t und t' interagieren nur miteinander, wenn sie benachbart sind. Die Interaktionsregeln werden durch die Kleber bestimmt.

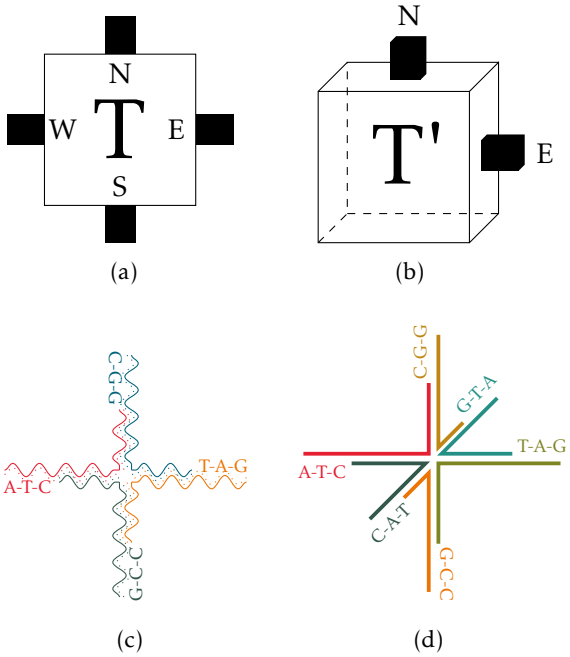


ABBILDUNG 2.22: Beispiele eines Tiletypen in 2D und 3D – mathematisch und biologisch modelliert. Die mathematischen Kleber sind durch schwarze Quader dargestellt, ihr Typ durch ein Label. Die biologischen Kleber werden mittels Basensequenzen dargestellt.

DEFINITION 2.25 – Ein *n-dimensionaler Tiletyp* T_n ist eine Schablone für ein Tile t_n. Ein n-dimensionaler Tiletyp wird durch ein Label $\mathcal{L}_T \in \Sigma^*$ und eine Menge von Klebern $g_i \in G$ definiert. Ein Kleber befindet sich an jeder Seite $u_{t,i} \in U_t$ von t_n. Folgende Funktionen sind jedem Tiletyp zugeordnet:

$$\text{glue} : T_n \times U_t \mapsto G$$
$$\text{strength} : g \in G \mapsto \mathbb{N}$$

Tiles, die identische Kleber und Kleberstärken sowie dieselben Labels haben, sind vom selben Tiletyp.

Werden Kleber nicht explizit gezeigt oder erwähnt, so wird vom leeren Kleber $\mathcal{L}_g = \emptyset$ mit dem Label VOID mit der Stärke 0 ausgegangen.

Ein Tile t, das einen Kleber mit Stärke 0 oder keinen Kleber an der Seite u_t hat kann dort nicht mit benachbarten Tiles t' interagierten.

DEFINITION 2.26 – Die *Temperatur* τ eines Self-Assembly-Systems beschreibt die für eine stabile Bindung minimal benötigte Kleberstärke s.

Zwei Tiles *binden* aneinander, wenn sie benachbart sind, Kleber mit passender Farbe haben und die Bindungsstärke mit allen Nachbarn wenigstens der

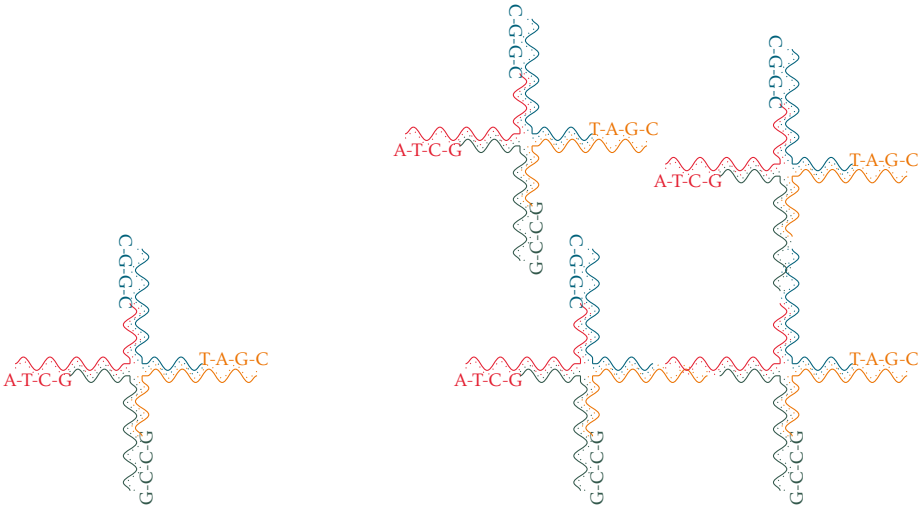


ABBILDUNG 2.23: (a) Schematische Repräsentation eines DNA-Tile. (b) Drei DNA-Tiles, die einen Zusammenschluss gebildet haben und ein viertes während der Bindung.

Temperatur τ entspricht. Ein passendes Label eines DNA-Tiles ist durch einen passenden Komplementärstrang ausreichender Länge realisierbar.

Die Temperatur τ modelliert die physikalische Temperatur. Erhört sich die Temperatur, so bewegen sich Moleküle schneller. Durch die zusätzliche Energie wird es wahrscheinlicher, dass Bindungen zerstört werden oder sich nicht stabil bilden können.

DEFINITION 2.27 – Zwei Tiles t und $t^{'}$ vom Typ T und $T^{'}$ *binden korrekt* bei Temperatur τ, wenn folgende Bedingungen gelten:

$$
\begin{aligned}
&(1)\ t \text{ und } t' \text{ sind benachbart} \\
&(2)\ \exists\, u_i \in U_t : \text{label}(\text{glue}(t, u_i)) = \text{label}(\text{glue}(t', -u_i)) \\
&\quad \wedge\, \text{strength}(\text{glue}(t, u_i)) \geq \tau \\
&\quad \wedge\, \text{strength}(\text{glue}(t^{'}, -u_i)) \geq \tau
\end{aligned}
$$

Bindungen, welche Bedingung (1) oder (2) verletzen, werden *falsch* oder *Fehler* genannt.

Damit Tiles miteinander binden können, müssen sie passend zueinander rotiert sein. Bei biologischen Systemen auf Basis von DNA passiert dies automatisch. Bei der mathematischen Modellierung wird gefordert, dass Tiles nicht rotierbar sind.

Durch die intrinsischen Eigenschaften von DNA sind Fehler unumgänglich [135]. Die Anzahl von Fehlern kann jedoch verringert werden, wenn spezielle

Tiletypen eingesetzt werden. In [110] werden Vorgehensweisen geschildert, die Fehlerwahrscheinlichkeiten in zwei Dimensionen stark reduzieren.

Die *Bindungsstärke* s zwischen den Tiles t und t' entspricht der Stärke der beteiligten korrekten Kleber. Die Stärke s eines Klebers wird durch die Anzahl der schwarzen Boxen pro Seite bestimmt. In Abbildung 2.22 haben alle Kleber die Stärke $s = 1$.

Die *absolute Bindungsstärke* eines Tiles t ist die Summe der Bindungsstärken zwischen t und seinen korrekten Nachbarn. Inkorrekte Kleber von Nachbarn tragen nicht zur absoluten Bindungsstärke bei. Tiles mit zu geringer Bindungsstärke oder falschem Label binden, werden Fehler genannt.

DEFINITION 2.28 – Ein n-dimensionales *Tile Assembly*, oder *Assembly*, ist eine partielle Funktion $\alpha : \mathbb{Z}^n \mapsto T$, wobei T ein Tileset von n-dimensionalen Tiletypen T_n ist. Ein Assembly α ist τ-stabil, wenn kein Tile $t_n \in T$ aus dem Assembly entfernt werden kann, ohne dass Kleber der Stärke $\tau \in \mathbb{N}$ entfernt werden müssen.

DEFINITION 2.29 – Der *Rand* eines Assembly α ist eine Untermenge von α. Der Rand beinhaltet alle Tiles mit wenigstens einem freien Nachbarort.

Nur Rand-Tiles eines Assemblies können mit freien Tiles interagieren und somit das Assembly wachsen lassen.

DEFINITION 2.30 – Die *Wachstumsfront* eines n-dimensionalen Assemblies ist eine Untermenge von Orten des $\mathbb{Z}^n$. Ein Ort ist genau dann Teil der Wachstumsfront, wenn er nicht belegt ist und ein benachbarter Ort von einem Rand-Tile ist. Außerdem wird ein Kleber der Minimalstärke 1 zum benachbarten Ort vorausgesetzt.

Die Orte der Wachstumsfront ändern sich, wenn Tiles zum Assembly hinzugefügt oder davon entfernt werden. Es wird angenommen, dass zu jedem diskreten Zeitpunkt genau ein Tile zum Assembly hinzugefügt oder abgelöst wird.

Das initiale Assembly α_0 zum Zeitpunkt 0 wird *Seed-Assembly* oder *Seed-Tile* σ genannt.

Beginnend bei σ werden dem Assembly nichtdeterministisch Tiles an der Wachstumsfront hinzugefügt. Wegen des inhärenten Nichtdeterminismus von Self-Assembly-Systemen kann es schwierig sein Tilesets zu erschaffen, die zu gewünschten Formen heranwachsen.

DEFINITION 2.31 – Ein *Tile Assembly Modell* (*TAM*) ist ein Tupel $\mathcal{T}_\tau = (T, \sigma, \tau)$, wobei T eine endliche Menge von Tiletypen beschreibt – auch *Tileset* genannt, σ ein Seed-Assembly ist und $\tau \in \mathbb{N}$ die Temperatur des TAM $\mathcal{T}_\tau$ beschreibt.

Im Verlauf der Arbeit wird τ als 2 oder 3 angenommen und ausgelassen. Temperaturen 2 und 3 genügen, um alle im Verlauf der Arbeit relevanten Funktionali-

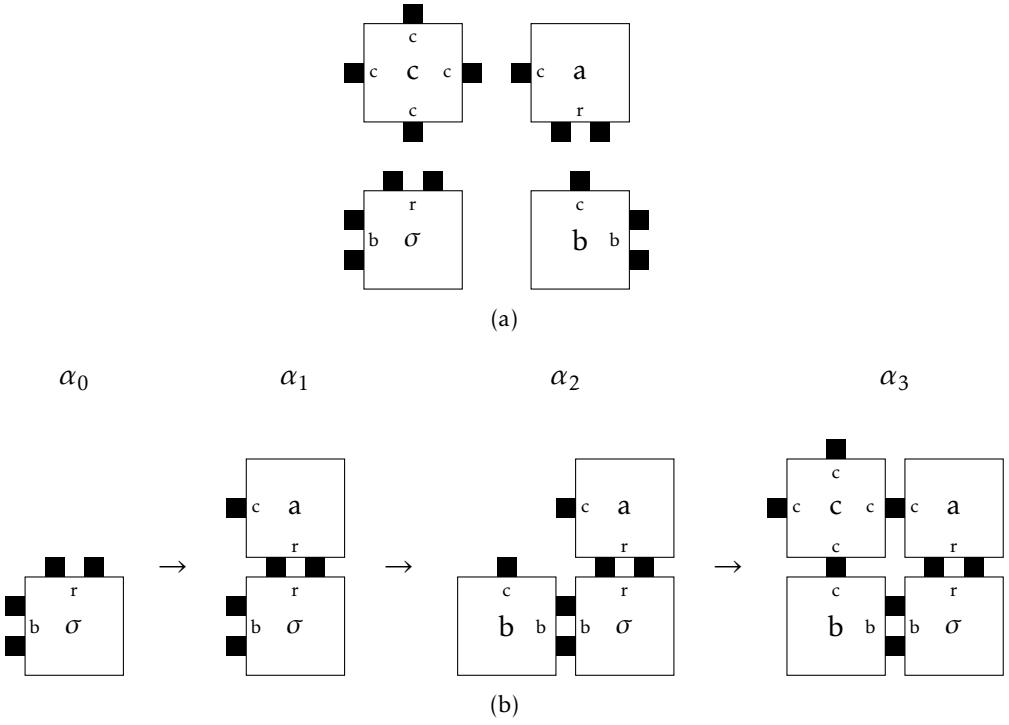


ABBILDUNG 2.24: (a) Beispiel für ein Tileset eines 2D-Self-Assembly-Systems. (b) Die Assemblysequenz zum Tileset (a). Das Seed-Tile wird σ genannt und die Temperatur hat den Wert 2. $\langle \alpha_0, \ldots, \alpha_3 \rangle$ zeigt die drei nötigen Schritte bis zum terminalen Assembly.

täten umzusetzen.

DEFINITION 2.32 – $\mathcal{A}[\mathcal{T}]$ beschreibt die Menge aller terminalen Assemblies, welche aus einem TAM $\mathcal{T}_\tau$ in endlicher Zeit hervorgehen können. Ein Assembly $\alpha \in \mathcal{A}[\mathcal{T}]$ ist *terminal* ($\alpha_\square$), wenn kein Tile τ-stabil zum Assembly hinzugefügt werden kann.

DEFINITION 2.33 – Sei α_i ein durch ein TAM $\mathcal{T}_\tau$ erzeugtes Assembly. Eine *Assemblysequenz* eines TAM $\mathcal{T}_\tau$ ist eine Sequenz $S = \langle \alpha_0, \alpha_1, \ldots \rangle$, wobei α_{i+1} aus α_i durch Hinzufügen eines Tiles zu α_i erzeugt wurde. Ist die Sequenz endlich, so wird das letzte Element das *Ergebnis* $\alpha_\square$ oder *terminal* genannt.

Abbildung 2.24 (a) zeigt ein beispielhaftes Tileset eines 2D-TAS mit Seed-Tile σ. Die Assemblysequenz des TAM $\mathcal{T}_\tau$ bei der Temperatur $\tau = 2$ ist in Abbildung 2.24 (b) dargestellt. α_0 beschreibt das Assembly zum Zeitpunkt 0. α_1, α_2 und α_3 zeigen das Assembly nach Hinzufügen von jeweils einem weiteren Tile.

Des Weiteren ist es wichtig, dass 3D-Self-Assembly-Systeme wenigstens so mächtig sind wie 2D-Self-Assembly-Systeme. Andernfalls kann keine Turing-Vollständigkeit gewährleistet werden. Lemma 2.1 zeigt dies.

LEMMA 2.1 – Ein n-dimensionales Self-Assembly-System ist wenigstens so mächtig wie ein $n-1$-dimensionales Self-Assembly-System.

Beweis. Ein n-dimensionales Self-Assembly-System kann ein $n-1$-dimensionales Self-Assembly-System simulieren, indem Kleber in der zusätzlichen Dimension weggelassen werden. Eine Projektion des entstehenden Assemblies auf den $n-1$-dimensionalen Raum kann ein identisches semantisches Verhalten erzeugen. □

Ein Self-Assembly-System ist eine Erweiterung eines TAM $\mathcal{T}_\tau$. Ein Self-Assembly-System koordiniert das zeitliche Verhalten eines TAM $\mathcal{T}_\tau$. Es gibt wenigstens drei häufig verwendete Self-Assembly-Systeme. Sie sind auf Eric Winfrees grundlegende Arbeit zurückzuführen [135]. Sie werden *abstract Tile Assembly Modell* (aTAM), *kinetic Tile Assembly Modell* (kTAM) und *twohanded Tile Assembly Modell* (2HAM) genannt. Die Besonderheiten der Modelle werden in den folgenden Abschnitten genauer erklärt.

2.5.5.3 ABSTRACT TILE ASSEMBLY MODEL

Das aTAM ist das einfachste hier vorgestellte Self-Assembly-System. Es wird verwendet, um theoretische Fragestellungen zum Thema Selbstzusammensetzung und Kristallbildung zu beantworten. Im aTAM werden keine Fehler modelliert und Parallelität ist nicht vorgesehen.

DEFINITION 2.34 – Sei T eine Menge von Tiletypen, die nicht rotiert werden können. Sei σ ein endlich großes Seed-Assembly. Ein *abstract Tile Assembly Modell* (aTAM) ist ein Tupel $\mathcal{A} = (T, \sigma, \tau)$ mit der Temperatur τ. Als zusätzliche Einschränkung darf zu jedem Zeitpunkt nur ein einzelnes Tile zum Assembly hinzugefügt werden. Es können keine Tiles aus dem Assembly entfernt werden.

2.5.5.4 KINETIC TILE ASSEMBLY MODEL

Das kTAM ist das für diese Arbeit wichtigste Self-Assembly-System. Das kTAM ähnelt der grundlegenden Funktionsweise des aTAM. In jedem diskreten Simulationsschritt wird genau ein Tile behandelt. Im Gegensatz zum aTAM können sich einmal gebundene Tiles wieder vom Assembly lösen. Zudem ist es möglich, dass fehlerhafte Bindungen auftreten – was im aTAM nicht modelliert wird. Das Fehlverhalten von Self-Assembly-Systemen wird in Kapitel 3 im Detail untersucht. Durch diese zwei Kernunterschiede ist das Simulationsergebnis eines kTAM deutlich näher an der Realität als es beim aTAM der Fall ist.

Das Hinzufügen und Ablösen von Tiles wird durch eine Vorwärtsrate und eine Rückwertsrate bestimmt.

DEFINITION 2.35 – Die *Vorwärtsrate* r_f, mit der Tiles t zum Assembly hinzugefügt werden, ist durch $r_f(t) = k_f e^{-G_{mc}}$ definiert. k_f beschreibt das Zeitverhalten des Systems. G_{mc} beschreibt die benötigte Energie, um ein Tile zu fixieren, wobei $G_{mc} > 0$.

Für die Vorwärtsrate wird angenommen, dass zu jedem Zeitpunkt ein endloser Vorrat an Tiles von jedem Typ vorliegt und die Konzentration der Tiles zu jedem Zeitpunkt überall gleich ist. G_{mc} hängt dabei ausschließlich von der Tilekonzentration ab und kann nach Belieben verändert werden. Diese Vereinfachung ermöglicht eine effizientere Analyse. Neben der Möglichkeit eine Bindung an falschen Orten einzugehen, ist es überdies auch möglich, dass sich Tiles wieder aus dem Assembly lösen.

DEFINITION 2.36 – Die *Rückwärtsrate* $r_{r,b}(t) = k_f e^{-bG_{se}}$ beschreibt die Rate, mit der Bindungen in einem Assembly aufbrechen können. G_{se} beschreibt die dafür benötigten Energiekosten. b beschreibt die Anzahl der Bindungen, welche ein Tile bereits geformt hat.

Da die Rate exponentiell mit der Anzahl der geformten Bindungen fällt, wird es immer unwahrscheinlicher, dass ein Tile entfernt wird, je mehr Bindungen es eingegangen ist. Der Parameter G_{se} kann in Wet-Lab Experimenten durch Manipulation der Umgebungstemperatur angepasst werden.

DEFINITION 2.37 – Sei σ ein nicht leeres Seed-Assembly und T ein Tileset. Weiterhin sei $r_{r,b}$ eine Rückwärtsrate und r_f eine Vorwärtsrate. Ein *kinetic Tile Assembly Modell* (kTAM) ist ein Tupel $\mathcal{K} = (T, \sigma, r_{r,b}, r_f)$.

Die Temperatur τ eines kTAM korreliert mit dem Verhältnis $\frac{G_{mc}}{G_{se}}$ und wird im Kontext von kTAM fortan ebenfalls als Temperatur bezeichnet. Die korrekte Position eines Tiles wird im kTAM um den Faktor $e^{G_{se}}$ bevorzugt. Es ist folglich wahrscheinlicher, dass ein Tile an einem korrekten Ort bindet.

2.5.5.5 TWO-HANDED TILE ASSEMBLY MODEL

Das 2HAM ist, im Gegensatz zu dem aTAM und dem kTAM, ohne ein Seed-Tile oder Seed-Assembly formuliert. Zu jedem Zeitpunkt können alle Tiles und Assemblies miteinander interagieren, sofern die Temperaturbeschränkung τ und die Form der Assemblies es zulassen. Die Abwesenheit eines Seed-Assemblies spiegelt die Realität in DNA-Experimenten deutlich realistischer wider.

Im Folgenden werden die wichtigsten Definitionen für 2HAM-Systeme aufgezeigt. Diese gehen auf Patitz zurück [110].

DEFINITION 2.38 – Zwei Assemblies α und β sind *disjunkt*, wenn für alle Orte von α und β gilt, dass $\alpha \cap \beta = \emptyset$ ist. Ein Assembly besteht aus wenigstens einem Tile.

Ein Assembly im 2HAM ist entweder ein Tile oder ein Assembly, für das gilt: Zwei Assemblies α und β lassen sich *korrekt vereinigen*, wenn das resultierende Assembly τ-stabil ist und α und β disjunkt sind.

DEFINITION 2.39 – Der Zustand S eines Tilesets T ist eine Multimenge von Assemblies, für die gilt:

- Alle Assemblies $\in S$ sind τ-stabil. Es müssen wenigstens τ Kleber entfernt werden, damit das Assembly zerfällt.
- Alle Assemblies $\in S$ lassen sich durch korrekte Vereinigung aus T bilden.

Der Zustand S von T wird durch eine Multimenge beschrieben, da es möglich sein muss Assemblies mit sich selbst zu vereinigen. Es werden maximal zwei Exemplare jedes Assemblies benötigt.

DEFINITION 2.40 – Ein *twohanded Tile Assembly Modell* (2HAM) ist ein Tupel $\mathcal{H} = (T, S_0, \tau)$, wobei T eine Menge an Tiletypen beschreibt, S_0 ein Startzustand ist und τ eine Temperaturbeschränkung.

DEFINITION 2.41 – Sei $\mathcal{H} = (T, S, \tau)$ ein 2HAM. Die Assemblysequenz eines 2HAM ist eine Sequenz von Zuständen $S = \langle S_0 \dots S_k \rangle$ mit $k \in \{1 \dots \infty\}$. S_{i+1} wird aus S_i durch korrekte Bildung von allen möglichen Vereinigungen von Assemblies $s, s' \in S$ erzeugt.

DEFINITION 2.42 – Sei $\mathcal{H} = (T, S_0, \tau)$ ein 2HAM. Ein Zustand S_i ist *terminal*, wenn S_{i+1} nach Anwendung aller möglichen Vereinigungen äquivalent zu S_i ist.

2.5.5.6 WAHRHEITSWERTE IN SELF-ASSEMBLY-SYSTEMEN

Zahlen werden in Berechnungsmodellen als eine Verkettung von zustandsbehafteten Grundkomponenten repräsentiert. Zwei unterscheidbare Zustände genügen, um platzeffizient Zahlen darstellen zu können. Die meisten Berechnungsmodelle verwenden Wahrheitswerte, um die Zahlen „0" und „1" darzustellen. Im Fall von klassischen Rechnern wird eine „0" oft durch „keine Spannung" und eine „1" durch "hohe Spannung" repräsentiert.

In Self-Assembly-Systemen gibt es zwei Möglichkeiten Wahrheitswerte auszudrücken. Zum einen kann die Anwesenheit eines Tiles als „0" oder „1" interpretiert werden. Dies hat den Vorteil, dass keine weitere Interpretation nötig ist, um Ergebnisse auf Nanoebene weiter verarbeiten zu können. Allerdings kann nur einer der beiden Wahrheitswerte zeitgleich dargestellt werden. Es ist nicht möglich, die Abwesenheit eines Tiles als den inversen Wahrheitswert zu interpretieren. Es kann nicht sinnvoll unterschieden werden, ob ein Tile nicht anwesend oder "noch" nicht anwesend ist.

Die zweite Möglichkeit besteht in der Darstellung von Wahrheitswerten durch Kleber von Tiles. Dabei wird einem eindeutig bestimmbaren Tile eine Semantik zugeordnet und entsprechend der Kleber können Berechnungen durchgeführt werden, welchen durch entsprechende Tile-Labels eine Semantik zugeordnet werden kann. Es können sowohl „0" als auch „1" zeitgleich dargestellt und verarbeitet werden.

2.5.6 GEGENÜBERSTELLUNG DER BERECHNUNGSMODELLE

Klassische Schaltkreise erscheinen für den Einsatz in Nanonetzwerken ungeeignet, da diese von Quanteneffekten gestört werden können und eine weitere Miniaturisierung unwahrscheinlich erscheint.

Chemische Reaktionsnetzwerke erscheinen ungeeignet, da diese komplizierte, makroskopische Messtechnik benötigen, um Ergebnisse ein- und auszulesen. Es erscheint unwahrscheinlich, dass Nanogeräte die benötigten Messungen und Manipulationen autark durchführen können.

Von den vorgestellten Berechnungsmodellen erscheinen QCA und Self-Assembly-Systeme am besten geeignet für den Einsatz in Nanonetzwerken.

Bei QCA sind nicht dieselben Limitationen wie bei klassischen Schaltkreisen zu erwarten, da diese auf denselben Effekten basieren, welche Schaltkreise ausschließen. Allerdings gibt es zahlreiche ungelöste Probleme bei QCA. Dazu zählt vor allem die präzise Platzierung von Quantum-Dot-Zellen.

Self-Assembly-Systeme erscheinen am besten geeignet für den Einsatz als Berechnungsmodell und Konstruktionsmechanismus auf Nanoebene. Alle dafür benötigten Bausteine sind bereits verfügbar – lediglich die Funktionsweisen und Paradigmen müssen umdefiniert werden.

Aus diesen Gründen beschäftigt sich diese Arbeit vornehmlich mit Self-Assembly-Systemen.

KAPITEL 3

FEHLER IN SELF-ASSEMBLY-SYSTEMEN

DIESES Kapitel[1] befasst sich mit den verschiedenen Arten von Fehlern, wie sie in DNA-basierten System auftreten können. Zuerst werden drei Arten von Fehlern definiert, welche im kTAM und 2HAM auftreten können. Im Anschluss daran werden Verfahren vorgestellt und bewiesen, welche zur Fehlervermeidung eingesetzt werden können.

3.1 FEHLERTYPEN

Das kTAM wird häufig verwendet, wenn die Ergebnisse von DNA-Experimenten realitätsnah vorhergesagt werden sollen. Im kTAM ist es möglich, dass fehlerhafte Bindungen auftreten, was das natürliche Verhalten von DNA widerspiegelt.

Es gibt drei Arten von Fehlern, welche in Experimenten mit DNA-Tiles regelmäßig auftreten. Das kTAM modelliert zwei dieser Fehlerarten und ist somit vergleichsweise realitätsgetreu [110].

DEFINITION 3.1 – Ein *Growth-Error* ist ein Fehler, bei dem ein Tile so zum Assembly hinzugefügt wird, dass wenigstens eine Seite des Tiles einen Nachbarn mit unpassendem Kleber hat.

Ein Growth-Error wird als *fest* bezeichnet, wenn er durch weitere hinzugefügte Tiles mit hoher Wahrscheinlichkeit nicht mehr gelöst werden kann. Dies ist der Fall, sobald eine korrekte absolute Bindungsstärke in Höhe der Temperatur erreicht wird. Abbildung 3.1 (a) zeigt ein Beispiel. Das grau hervorgehobene Tile bindet fälschlicherweise mit einem korrekten Kleber unten und einem falschen Kleber rechts. Abbildung 3.1 (b) zeigt dasselbe Assembly einen Zeitschritt später.

[1] Teile dieses Kapitels wurden bereits in [1, 3] vorgestellt.

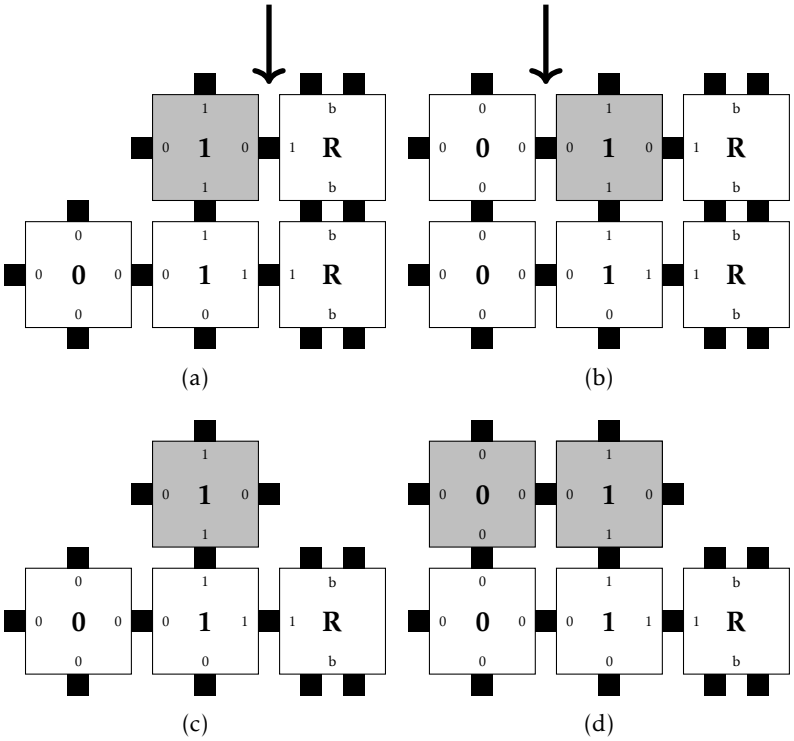


ABBILDUNG 3.1: (a) Growth-Error beim grauen Tile. (b) Der durch neu hinzugekommene Nachbarn festgesetzte Growth-Error. (c) Ein Facet-Error bei dem grauen Tile. (d) Der durch neu hinzugekommene Nachbarn festgesetzte Facet-Error.

Ein weiteres Tile wurde dem Assembly hinzugefügt, weswegen die Temperaturanforderung 2 für eine feste Bindung beim grau hervorgehobenen Tile erfüllt ist.

DEFINITION 3.2 – Ein *Facet-Error* ist ein Fehler, bei dem ein Tile mit unzureichender Stärke am Assembly bindet, aber keine fehlerhaften Bindungen aufweist.

Auch ein Facet-Error wird als *fest* bezeichnet, wenn eine oder mehrere hinzugefügte Tiles die absolute Bindungsstärke des Tiles so weit erhöhen, dass ein Ablösen des Tiles unwahrscheinlich wird. Abbildung 3.1 (c) und (d) zeigen ein Beispiel für diesen Prozess. Das grau hervorgehobene Tile bindet mit lediglich einem Kleber am Assembly. Im nächsten Zeitschritt bindet ein nun passendes Tile daneben, weswegen die notwendige absolute Bindungsstärke erreicht wird und ein Wiederablösen des falschen Tiles sehr unwahrscheinlich wird. Das beschriebene Verhalten kann zu Fehlern führen da es beispielsweise mehrere Tiles mit einem Kleber der Stärke 1 und dem Label 1 geben kann, von denen lediglich ein bestimmtes für eine Position vorgesehen ist. Dieses sollte erst eine feste Bindung eingehen sobald eine Anzahl benachbarter Kleber in Höhe der

Temperatur τ vorhanden sind. Bindet ein falsches Tile durch einen Facet-Error an einer nicht vorgesehenen Position, so wird das Assembly nicht mehr zur vorgesehenen Struktur erwachsen.

Die dritte Art Fehler wird Nucleation-Error genannt. Dieser Fehler tritt auf, wenn ein Assembly ohne Seed-Assembly σ erzeugt wird.

DEFINITION 3.3 – Ein *Nucleation-Error* tritt in Self-Assembly-Systemen auf, wenn ein Assembly gebildet wird, ohne dass ein Seed-Tile zugegen oder beteiligt ist.

Neben den drei genannten Fehlerquellen in Self-Assembly-Systemen existiert in der dreidimensionalen Version noch eine vierte Fehlerquelle. Es kann passieren, dass Teile der Wachstumsfront vollständig von Tiles eingeschlossen werden. Dieses Verhalten würde verhindern, dass unter realistischen Bedingungen eingeschlossene Orte mit Tiles gefüllt werden können.

Da die meisten Experimente mit DNA-Tiles in 2D durchgeführt wurden, ist dieser Fehler wenig erforscht, theoretisch jedoch zu erwarten. Für die in den folgenden Kapiteln betrachteten Strukturen ist diese Fehlerquelle jedoch nur von theoretischem Interesse.

3.2 FEHLERKORREKTURMECHANISMEN

In den folgenden Abschnitten werden verschiedene Techniken vorgestellt, mit denen die Anzahl der potentiellen Fehler in Self-Assembly-Systemen reduziert werden kann.

Die einfachste Möglichkeit, Fehler im kTAM zu verhindern, ist die Anpassung des Verhältnisses $\frac{G_{mc}}{G_{se}}$, welches mit der Temperatur τ eines aTAM verglichen werden kann. Wird die Umgebungstemperatur sehr präzise eingestellt, so können Fehler nahezu beliebig unwahrscheinlich werden. Tiles könnten beispielsweise ab einer Temperatur von 37 °C mit einer Kleberstärke von zwei stabil binden. Wird dieser Schwellwert genau eingestellt, so treten praktisch keine Fehler auf, da alle falschen Bindungen sofort wieder aufgelöst werden. Dies führt jedoch zu einer Verlangsamung des Assembly-Prozesses. Aus diesem Grund wird in Wet-Lab-Experimenten die Temperatur etwas weiter herab reguliert, um einen akzeptablen Kompromiss zwischen Fehlerrate und Geschwindigkeit zu erzielen.

Unabhängig voneinander haben verschiedene Wissenschaftler weitere Prozesse vorgeschlagen, mit denen man Fehler bei Assembly-Prozessen begegnen kann. Eine Möglichkeit besteht darin, Tiles durch $k \times k$-Blöcke zu ersetzen, sodass ihr semantisches Verhalten identisch bleibt [78]. Dies wird auch *kinetisches Fehlerlesen* oder *Blockersetzung* genannt. Blockersetzungsstrategien können zur Reduktion von Growth-Errors und Facet-Errors eingesetzt werden [49, 134]. Zwei Beispiele sind $k \times k$*-Proofreading* und *Snaked-Proofreading*. Diese werden in den folgenden Abschnitten genauer untersucht.

3.2.1 $k \times k$-PROOFREADING

$k \times k$-Proofreading ist die einfachste Blockersetzungsstrategie. Dabei wird jedes Tile eines ursprünglichen Tilesets durch einen $k \times k$-Block von Tiles (k-Block) ersetzt. Ein Beispiel für eine solche Ersetzung ist in Abbildung 3.2 (a) zu sehen. Das obere Tile wird durch den unten dargestellten Block ersetzt. Der Einfachheit halber wurde für k der Wert 2 gewählt. Die Kleber der neuen Tiles sind so gewählt, dass nach Außen dieselbe logische Kleberstruktur bereitgestellt wird. Ein Kleber a wird bei einer Kantenlänge $k = 2$ beispielsweise durch zwei Kleber a und $a^{'}$ ersetzt. Nach Innen werden bei $k = 2$ vier neue, eindeutige Kleber $x_1 \dots x_4$ bereitgestellt. Diese sind pro k-Block einzigartig.

Nach der Ersetzung aller Tiles durch k-Blöcke müssen mehrere Fehler auftreten, damit die ursprüngliche Semantik verfälscht werden kann. Im allgemeinen Fall kann die Anzahl der auftretenden Growth-Errors so um einen Faktor $1/k$ verringert werden [49, 134]. Bei einer Kantenlänge von k müssen k^2 Fehler auftreten, um die ursprüngliche Semantik zu verfälschen.

3.2.2 SNAKED-PROOFREADING

Eine weitere Verbesserung des $k \times k$-Proofreading ist das Snaked-Proofreading [144]. Snaked-Proofreading ist ebenfalls eine Blockersetzungsstrategie. Zusätzlich zu der Ersetzung eines Tiles durch einen k-Block wird die interne Kleberstruktur angepasst. Der Vorgang ist in Abbildung 3.2 (b) dargestellt. Im Gegensatz zu $k \times k$-Proofreading werden bei $k = 2$ nur drei interne Kleber benötigt. Diese werden so angeordnet, dass Facet-Errors verhindert werden. Beim unteren linken Tile des k-Blocks ist nach rechts intern kein Kleber vorgesehen. Dadurch werden Facet-Errors in diese Richtung verhindert. Die Tiles des k-Blocks können sich nur der Reihe nach zu einem k-Block vervollständigen.

Growth-Errors werden auf diese Weise genau wie beim $k \times k$-Proofreading reduziert. Zusätzlich werden Facet-Errors in der y-Dimension verhindert.

In zweidimensionalen Self-Assembly-Systemen können Growth-Errors durch Blockersetzungsstrategien beliebig unwahrscheinlich werden. Facet-Errors sind schwieriger zu reduzieren, da zwei Dimensionen eine Reduzierung nur in einer Dimension zulassen.

Da dreidimensionale Self-Assembly-Systeme ohnehin für Nanonetzwerke von Interesse sind, wird eine Blockersetzungsstrategie in dreidimensionalen Self-Assembly-Systemen gezeigt, welche Facet-Errors in allen Dimensionen verhindert und Growth-Errors beliebig unwahrscheinlich gestaltet.

3.2.3 3D-SNAKED-PROOFREADING

Im Folgenden wird ein Algorithmus vorgestellt, der das Snaked-Proofreading-Verfahren von zwei Dimensionen auf drei Dimensionen verallgemeinert. Das Verfahren [3] wurde von Philipp Bende Florian Lau und Stefan Fischer entwickelt. Der vorgestellte Beweis und die Verallgemeinerung des Snaked-Proofreading-Verfahrens auf beliebige Kantenlängen k wurden vornehmlich von Florian Lau entwickelt.

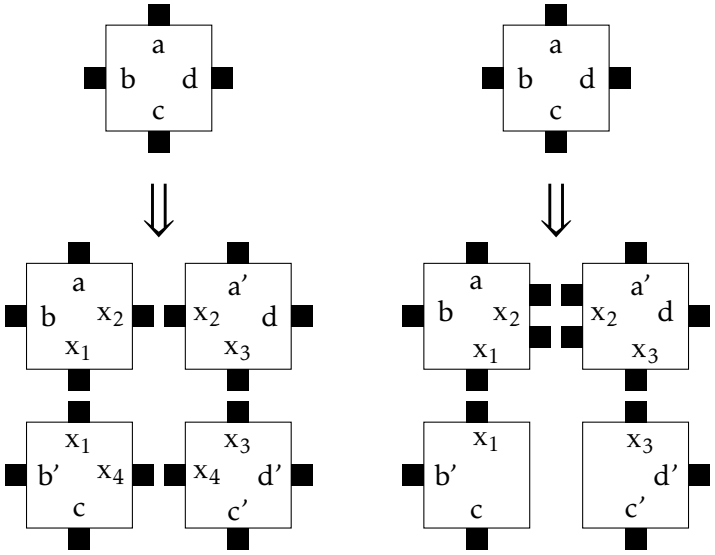


ABBILDUNG 3.2: (a) Der Tiletyp wird durch einen 2×2-Block von Proofreading-Tiletypen ersetzt. (b) Der Tiletyp wird durch einen 2×2-block von Snaked-Proofreading -Tiletypen ersetzt.

Zuerst wird die Vorgehensweise beschrieben, um dem Leser eine Intuition der Vorgehensweise zu vermitteln. Danach werden die notwendigen Lemmata für die Beweisführung vorgestellt. Im Anschluss folgt eine vollständige Induktion, um zu beweisen, dass das Vorgehen für beliebige, positive natürliche Kantenlängen der Blockersetzung funktioniert.

Der im Folgenden vorgestellte Algorithmus (Algorithmus 1) erzeugt für jedes Tile eines gegebenen Assemblies einen $k \times k \times k$-Snaked-Proofreading-Block. Dreidimensionale $k \times k \times k$-Snaked-Proofreading-Blöcke werden im Folgenden ebenfalls als *k-Block* abgekürzt.

Algorithmus 1 erstellt ein Snaked-Proofreading-Tileset, welches Theorem 3.1 genügt. Der Algorithmus wird aufgerufen, um einen k-Block für jedes Tile einer angestrebten Assemblysequenz zu erzeugen. Eine *angestrebte terminale Assemblysequenz* (Original- oder Ziel-Assemblysequenz) ist eine eindeutige kTAM Assemblysequenz. Für jede vorgesehene Position innerhalb eines k-Blocks wird ein Tiletyp generiert (Zeile 4). Für diesen Tiletyp werden alle Kleber definiert (Zeile 5). Zuerst werden die Seiten behandelt, die bereits über einen benachbarten Kleber verfügen. Es wird ein identischer Kleber definiert (Zeile 6–7). Danach wird geprüft, ob die betrachtete Seite extern ist (Zeile 8). *Externe* Seiten spiegeln die Logik des ersetzten Tiles wider. Wenn der besagte Kleber keinen Facet-Error zulässt (Algorithmus 2, Zeile 24–28) und eine eindeutige Assemblysequenz für den Block gewährt bleibt (Algorithmus 2, Zeile 18–23), so wird ein Kleber entsprechend der Ursprungslogik festgelegt. Entlang eines eindeutigen Pfades der Block-Assemblysequenz werden anschließend Kleber definiert (Zeile 10–13). Danach werden überall dort stabilisierende Kleber angebracht (Zeile 14–16), wo

keine Facet-Errors verursacht werden können (Algorithmus 2, Zeile 9–16).

Durch das Ersetzen jedes Tiles eines Assemblies durch einen k-Block werden Growth-Errors analog zur zweidimensionalen Variante reduziert. Um Facet-Errors zu verringern, wird die Kleberstruktur eines jeden k-Blocks angepasst, sodass in jede Richtung nur jedes zweite Tile einen Kleber in dieselbe Richtung besitzt. Kleber werden somit in einem dreidimensionalen Schachbrettmuster angebracht.

Damit Schranken für Fehler garantiert werden können, muss sicher gestellt sein, dass jeder k-Block sich mit hoher Wahrscheinlichkeit auf eine eindeutige Art und Weise zusammensetzt. Dafür muss eine Assemblysequenz des ursprünglichen Tilesets angenommen werden. Das ist notwendig, da die Zahl der bereits vorhandenen Nachbarn eines Tiles einen Einfluss auf die Block-intern benötigten Kleber hat. Um die Eindeutigkeit der Blockzusammensetzung zu gewährleisten, darf die Zahl der internen Kleber nicht zu hoch sein. Dies wird dadurch sichergestellt, dass jeder Block eine eindeutige Start- und Endposition zugewiesen bekommt.

ALGORITHMUS 1 Die Funktion CREATEKBLOCK erzeugt, gegeben ein Tile aus Assemblysequenz S, einen k-Block. Dies geschieht unter Zuhilfenahme der Subroutinen aus Algorithmus 2. Wird das Vorgehen auf alle Tiles der Zielassemblysequenz angewendet, kann so ein vollständiges Tileset für eine Nanostruktur erzeugt werden.

```
1:  procedure createKBlock(Tile t ∈ S, uint τ, uint k)
2:      set newTileSet = ∅;
3:      for Orte (x,y,z)^T ∈ k-Block do
4:          erzeuge Tiletyp t_xyz;
5:          for Seiten u_i von t_xyz do
6:              if (Seite_{-u_i}(t_xyz + u_i) ⊆ G_out(t_xyz + u_i)) then
7:                  Seite_{u_i}(t_xyz) = Seite_{-u_i}(t_xyz + u_i);
8:              else if (isExternal(S, t_xyz, u_i)) then
9:                  Seite_{u_i}(t_xyz) = setExt(S, (x,y,z)^T, u_i, k, τ);
10:             else if (not isExternal(u_i)) then
11:                 if (((x,y,z)^T + u_i) == nextTile()) then
12:                     Seite_{u_i}(t_xyz) = (str(g) = τ−
13:                         inputGlues(nextTile(S_b, (x,y,z)^T)));
14:                 else
15:                     if (stabilityGlue(S_b, (x,y,z)^T, u_i)) then
16:                         Seite_{u_i}(t_xyz) = (str(g) = ⌊τ/3⌋);
17:         newTileSet ∪ t_xyz;
18:     return newTileSet;
```

THEOREM 3.1 – Algorithmus 1 erzeugt ein $k \times k \times k$-Snaked-Proofreading-Tileset, für ein ursprüngliches Tileset mit angestrebter Assemblysequenz $S = \langle \alpha_0, \ldots, \alpha_\square \rangle$ unter Verwendung von $\mathcal{O}(nk^3)$ Tiletypen. Dabei werden Growth-Errors um einen Faktor von $1/k^3$ reduziert, wobei k die Seitenlänge des Blocks beschreibt und

ALGORITHMUS 2 Subroutinen von Algorithmus 1. Es werden Funktionen angeboten, welche das nächste Tile NEXTTILE einer Assemblysequenz zurückgeben, welche die Anzahl der vorhandenen Kleber INPUTGLUES bestimmen, welche eine Liste der externen Nachbarn EXTNEIGHBORS liefern, bestimmen, ob ein Ort extern ISEXTERNAL ist und wo weitere Kleber STABILITYGLUE gesetzt werden können, ohne die Prämisse zu verletzen. Außerdem gibt es eine Funktion zum Setzen der externen Kleber SETEXT, welche die Logik der Ursprungs-Tiles repräsentieren.

1: **procedure** NEXTTILE(Block Assemblysequenz S_b, Ort $(x,y,z)^T$)
2: **return** nächsten Ort $(x,y,z)^T$ aus S_b;
3: **procedure** INPUTGLUES(Assemblysequenz S, Block-Assemblysequenz S_b, Ort $(x,y,z)^T$)
4: **return** $|G_{in}|$;
5: **procedure** EXTNEIGHBORS(Block-Assemblysequenz S_b, Ort $(x,y,z)^T$)
6: **return** Liste⟨Ort⟩ l;
7: **procedure** ISEXTERNAL(Block-Assemblysequenz S, Seite $(x,y,z)^T$, Seite u)
8: **return** $(x,y,z)^T + u \subseteq$ EXTNEIGHBORS$(S(x,y,z)^T)$;
9: **procedure** STABILITYGLUE(Block-Assemblysequenz S_b, Ort $(x,y,z)^T$, Seite u)
10: **if** $(u == (1,0,0)^T$ AND ODD$(x+y+z))$ **then**
11: **return** true;
12: **else if** $(u == (0,1,0)^T$ AND EVEN$(x+y+z))$ **then**
13: **return** true;
14: **else if** $(u == (0,0,1)^T$ AND ODD$(x+y+z))$ **then**
15: **return** true;
16: **return** false;
17: **procedure** SETEXT(Assemblysequenz S, Ort $(x,y,z)^T$, Seite u_i, uint k, uint τ)
18: **if** $\mathrm{str}_{u_i}((x,y,z)^T) == \tau$ **then**
19: **if** $(t_{xyz} + u)$ nicht erstes Tile von Nachbarblock **then**
20: **if** (EVEN$(x+y+z)$ AND ODD(k)) **then**
21: **return** $\mathrm{Seite}_{u_i}(t) = (\mathrm{str}(g) = \tau - 1)$;
22: **else if** (ODD$(x+y+z)$ AND EVEN(k)) **then**
23: **return** $\mathrm{Seite}_{u_i}(t) = (\mathrm{str}(g) = \tau - 1)$;
24: **if** (EVEN$(x+y+z)$ AND ODD(k)) **then**
25: **return** $\mathrm{Seite}_{u_i}(t)$;
26: **else if** (ODD$(x+y+z)$ AND EVEN(k)) **then**
27: **return** $\mathrm{Seite}_{u_i}(t)$;
28: **return** $\mathrm{Seite}_{u_i} = (\mathrm{str}(g) = 0, \mathrm{Label}(g) = \mathrm{VOID})$;

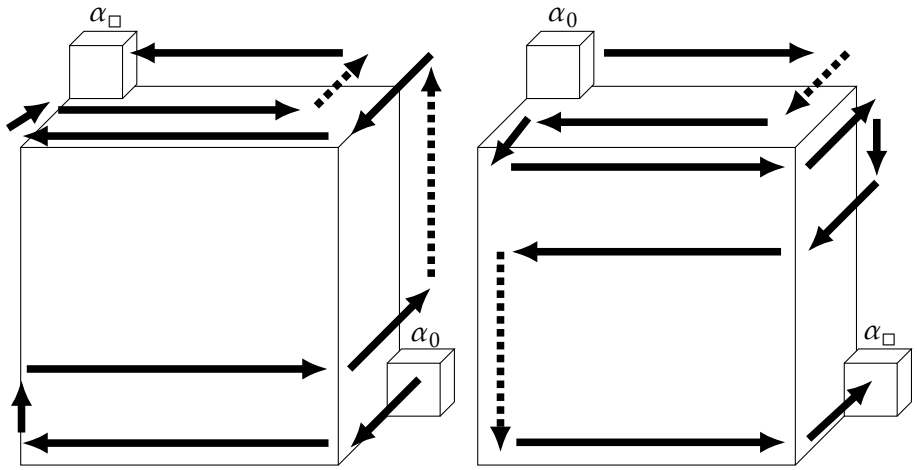


ABBILDUNG 3.3: Snaked-Pfad für zwei verschiedene Startpositionen. Ein $k+1$-Block kann aus einem k-Block durch Hinzufügen einer zusätzlichen Schicht in jeder Dimension generiert werden. Die äußeren Kleber müssen angepasst werden. (a) bezieht sich auf gerade Kantenlängen, (b) auf ungerade Kantenlängen.

n die Anzahl der Einträge der angestrebten Assemblysequenz darstellt. Des Weiteren werden Facet-Errors verhindert.

Beweis. Algorithmus 1 wird durch ein Konstruktionsschema für beliebig, aber feste Werte k bewiesen. Dieser erzeugt einen Snaked-Proofreading-k-Block für jedes Tile in der ursprünglichen Assemblysequenz S. Der k-Block erfüllt die Anforderungen von Lemma 3.1–3.3:

LEMMA 3.1 – Jeder k-Block, der durch Algorithmus 1 erzeugt wurde, hat einen eindeutigen Start- und Endpunkt und eine eindeutige k-Block-Assemblysequenz.

Beweis. Eine *k-Block-Assemblysequenz* (Snaked-Pfad) beschreibt die Reihenfolge, in der einzelne Tiles dem k-Block hinzugefügt werden (siehe Abbildung 3.3 (a) und (b)). Um eine Eindeutigkeit zu gewährleisten, adaptiert Algorithmus 1 die interne Kleberstruktur jedes k-Blocks. Für jeden bereits vorhandenen Nachbarn mit Klebern wird die Kleberstärke auf dem Snaked-Pfad um 1 reduziert. Das Ergebnis ist eine eindeutige k-Block-Assemblysequenz. □

LEMMA 3.2 – Ein k-Block, der Lemma 3.1 erfüllt, reduziert Growth-Errors um einen Faktor k^3.

Beweis. Da jedes Tile der ursprünglichen Assemblysequenz durch einen k-Block ersetzt wird, müssen k^3 Fehler auftreten, damit ein Block vollständig falsch ist. Da zusätzlich eine Blockersetzungsstrategie eingesetzt wird, sind die Garantien

gleichermaßen gut [134]. Dadurch werden Growth-Errors um einen Faktor k^3 unwahrscheinlicher. □

LEMMA 3.3 – k-Blöcke unterbinden Facet-Errors und stellen O(k) externe Kleber bereit.

Beweis. Um zu beweisen, dass k-Blöcke Facet-Errors effektiv verhindern, wird auf die Kleberstruktur aller jemals am Rand befindlichen Tiles verwiesen. Alle Tiles, die nicht Teil des Snaked-Pfades sind, haben keinen Nachbarn, der über Kleber in der gleichen Richtung verfügt. Dies schließt Tiles im k-Block und externe Tiles mit ein.

Da nur jedes zweite Tile Kleber nach außen hat, kann es nicht zu Facet-Errors kommen. Die Struktur des Snaked-Pfades lässt ebenfalls keine Facet-Errors zu.

Die Anzahl der Kleber, welche die ursprüngliche Logik repräsentieren, beläuft sich folglich auf $\frac{1}{2}k^2$ und entspricht der Klasse $\mathcal{O}(k)$. Die einzelnen Kleber werden entsprechend der ursprünglichen Stärke gesetzt. □

LEMMA 3.4 – Für jeden Eintrag der ursprünglichen Assemblysequenz S existiert eine eineindeutige k-Block-Assemblysequenz.

Um das zu beweisen wird gezeigt, dass eine 1:1 Korrespondenz zwischen ursprünglicher Assemblysequenz und der durch Algorithmus 1 erzeugten Assemblysequenz existiert.

Beweis. Sei $S = \langle \alpha_0, \ldots, \alpha_\square \rangle$ die ursprüngliche terminale Assemblysequenz des Tilesets T. Sei $S^{'} = \langle \alpha_0^{'}, \ldots, \alpha_\square^{'} \rangle$ die durch Algorithmus 1 erzeugte Assemblysequenz. Sei k die für die Blockersetzungsstrategie verwendete Kantenlänge und sei n die Anzahl der Einträge in S.

Algorithmus 1 sichert mit hoher Wahrscheinlichkeit, dass die Seite eines k-Blocks, welche als Wachstumsfront für einen benachbarten k-Block dient, zuerst vollständig sein muss, bevor der neue k-Block begonnen werden kann.

Dadurch kann jedes Assembly in S durch die k^3 Assemblies der Blockersetzungsstrategie substituiert werden, um so $S^{'}$ zu erzeugen. Durch Lemma 3.2 und Lemma 3.3 werden die Fehler reduziert. Lemma 3.1 garantiert die Eindeutigkeit.

Durch die Möglichkeit der Substitution der einzelnen Assemblies durch k-Block-Assemblysequenzen folgt, dass das neue Tileset mit hoher Wahrscheinlichkeit zur gewünschten Form heranwächst. □

Im Folgenden wird eine Vorgehensweise gezeigt, die Snaked-Proofreading-k-Blöcke erzeugt und Lemma 3.1, Lemma 3.2, Lemma 3.3 und Lemma 3.4 erfüllt.

Beweis. Sei $2 \leq k \in \mathbb{N}^+$ eine beliebige, aber feste Kantenlänge für die Blockersetzung. Um zu zeigen, dass die Strategie für beliebige k gilt, wird durch vollstän-

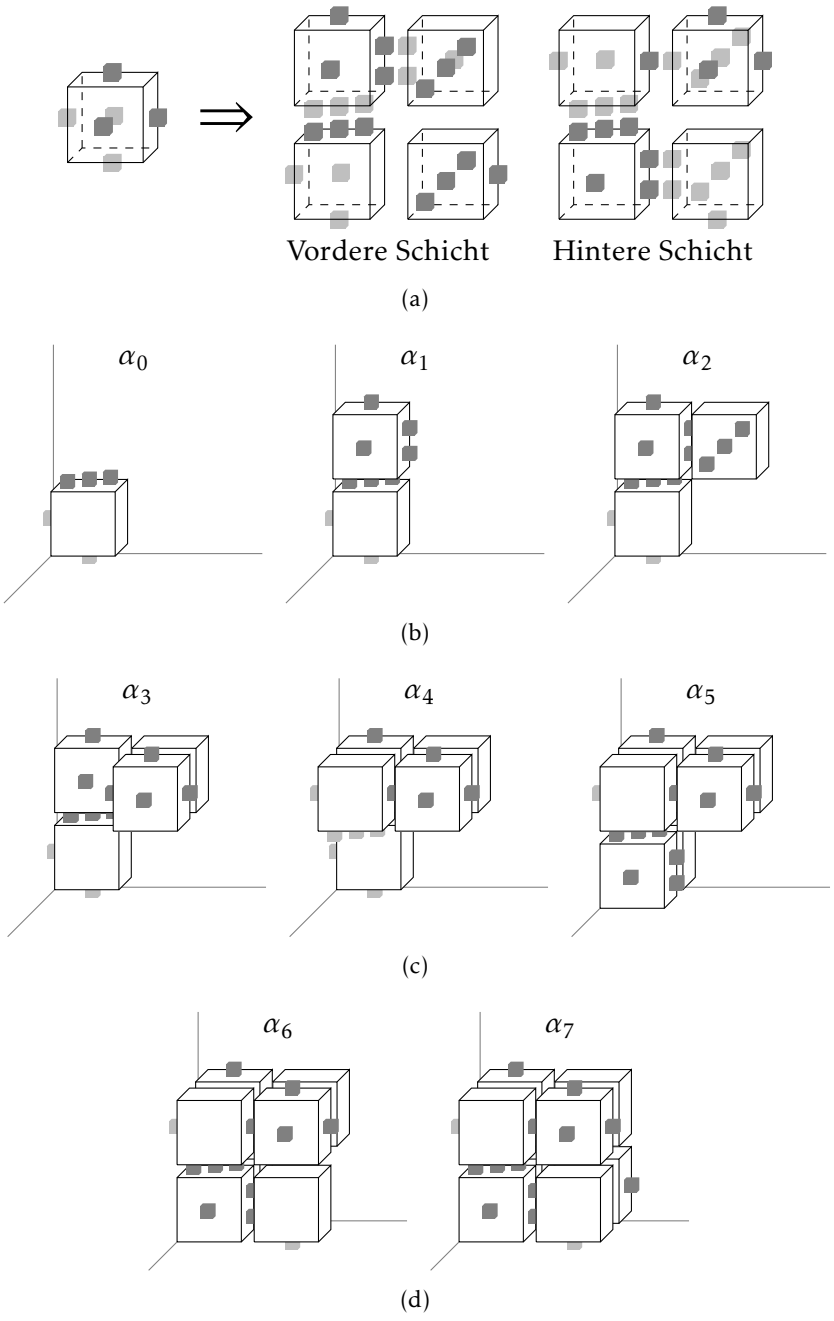


ABBILDUNG 3.4: (a) Ein $2 \times 2 \times 2$-Snaked-Proofreading-Tileset. (b) Eine Assemblysequenz eines Snaked-Proofreading-Block. Hinten, links und unten wird jeweils ein passender Kleber durch bereits vorhandene Nachbarn vorausgesetzt.

dige Induktion gezeigt, wie aus einem k-Block ein $k+1$-Block erzeugt werden kann. Es wird mit einem Wert $k \geq 2$ gestartet.

Induktionshypothese: Algorithmus 1 erzeugt korrekte k-Blöcke, die Lemmata 3.1–3.3 genügen.

Induktionsanfang: Ein 2-Block kann entsprechend Abbildung 3.4 (a)–(d)) leicht erzeugt werden:

Der 2-Block verhindert Facet-Errors, da nie zwei benachbarte Tiles t und $t^{'}$ externe Kleber haben, welche Teil der Wachstumsfront sind und in die gleiche Richtung zeigen. Growth-Errors werden um den Faktor $k^3 = 8$ reduziert.

Induktionsschritt: Sei $j \geq 3$ eine Blockersetzungskantenlänge.

Gemäß Induktionshypothese kann durch Algorithmus 1 ein $j+1$-Block durch Hinzufügen einer weiteren Schicht erzeugt werden. Der Prozess ist in Abbildung 3.3 dargestellt.

Abbildung 3.3 zeigt weiterhin mögliche Snaked-Pfade für gerade und ungerade k. Die zuvor nach außen zeigenden Kleber werden durch Kleber im Schachbrettmuster ersetzt, sodass ein einzigartiger Snaked-Pfad erhalten bleibt. Die jetzt externen Kleber erhalten die Kleber des ursprünglichen Tiles.

Da Algorithmus 1 die Nachbarschaften berücksichtigt, ist die Assemblysequenz des k-Blocks eindeutig. Durch das Schachbrettmuster werden Facet-Errors verhindert. Dadurch werden Lemma 3.3 und Lemma 3.1 eingehalten.

Um die Komplexität und Anzahl der Rechenschritte zu analysieren, wird wie folgt vorgegangen: Sei n die Anzahl der Einträge der ursprünglichen Assemblysequenz und k die Blocklänge. Algorithmus 1 verarbeitet jeden Eintrag der Sequenz exakt einmal und erzeugt maximal k^3 neue Tiletypen pro Eintrag. Daraus folgt, dass maximal $\mathcal{O}(nk^3)$ Tiletypen erzeugt werden und genauso viele Rechenschritte benötigt werden.

Da die Eindeutigkeit der Assemblysequenz jedes Tiles sichergestellt werden muss, kann ein zusätzlicher Faktor von bis zu 24 anfallen. Dies ist durch die verschiedenen Start- und Endpositionen des Snaked-Pfades für jeden k-Block bedingt. Es gibt acht verschiedene Startpunkte und für jeden Startpunkt drei mögliche Endpunkte. □

Durch Lemma 3.4 ist bekannt, dass die neue Form mit hoher Wahrscheinlichkeit eine skalierte Version der Ursprungsform ist. Dadurch gilt Theorem 3.1. □

3.3 FAZIT

Zusammenfassend wurde in Kapitel 3 gezeigt, wie man Fehler in Self-Assembly-Systemen, wie dem kTAM, verhindern kann. In zweidimensionalen Systemen existieren bereits Verfahren wie das Snaked-Proofreading, welches Facet-Errors jedoch nur in einer Dimension verhindert. Durch eine Verallgemeinerung des

Verfahrens auf drei Dimensionen können Facet-Errors in allen Dimensionen verhindert werden. Die übrigen Garantien des zweidimensionalen Verfahrens bleiben dabei unberührt.

Das neue Verfahren kann eingesetzt werden, um Fehler in DNA-Tile-basierten Nanonetzwerken zu vermeiden. Es lässt sich einsetzen, um Konstruktion, Kommunikation und Berechnungen robuster zu gestalten. Fehler sind in Self-Assembly-Systemen nie völlig vermeidbar. Durch Proofreading-Verfahren kann jedoch sichergestellt werden, dass DNA-basierte Technologien sinnvoll in Nanonetzwerken eingesetzt werden können.

KAPITEL 4

ANALYSE

Dieses Kapitel[1] befasst sich mit der Analyse und Kategorisierung von Anwendungsszenarien und Simulationswerkzeugen für Nanonetzwerke. Es wird analysiert, welche Mächtigkeit Nanogeräte mindestens haben müssen, damit die in der Fachliteratur vorgestellte Anwendungen sinnvoll unterstützt werden können.

In Abschnitt 4.1 werden Veröffentlichungen verschiedener Fachbereiche im Hinblick auf Anwendungsszenarien auf Nanoebene untersucht. In Abschnitt 4.1.1 werden die gesichteten Szenarien in ihre mathematischen Bestandteile zerlegt und in Abschnitt 4.1.2 entsprechend ihrer Schwierigkeit kategorisiert. In Abschnitt 4.1.3 werden Problemstellungen von besonderem Interesse für intensivere Untersuchungen in den folgenden Kapiteln ausgewählt. Zuletzt werden in Abschnitt 4.2 verschiedene Self-Assembly-Simulatoren und Netzwerksimulatoren auf ihre Tauglichkeit zur Simulation der beschriebenen Probleme untersucht.

4.1 ANFORDERUNGSANALYSE FÜR NANONETZWERKE

Dieser Abschnitt schildert die Vorgehensweise bei der Suche nach sinnvollen Szenarien für Nanonetzwerke, welche in späteren Kapiteln verwendet werden. Die Untersuchung von möglichen Anwendungen ermöglicht es, genauere Aussagen über die Anforderungen und Leistungsfähigkeit von einzelnen Nanogeräten zu tätigen. Die Analysen wurden von Florian Lau, Florian Büther und Bennet Gerlach durchgeführt [5]. Alle inhaltlichen Beträge gehen auf Florian Lau zurück.

Auf Basis der Analyse werden sinnvolle Berechnungsmodelle und Komplexitätsklassen für einzelne Nanogeräte aus bekannten Anwendungsszenarien abgeleitet. Der Fokus der untersuchten Szenarien liegt auf Nanomedizin und -kommunikation.

[1] Teile dieses Kapitels wurden bereits in [5, 7] vorgestellt.

Eines der prominentesten Szenarien ist die Erkennung bestimmter Krankheitsmarker mit anschließender Bekämpfung oder Meldung der Krankheit durch Nanogeräte oder Nanonetzwerke [125, 165].

In der Onkologie werden verschiedene Nanotechnologien bereits alltäglich eingesetzt und neue Ideen werden beständig aus der Forschung in die Wirtschaft integriert [124]. Dabei kommen vor allem Nanopartikel zum Einsatz. Einige Anwendungsideen schlagen z. B. Liposomen oder modifizierte Bakterien vor, um Medikamente an einen Anwendungsort zu transportieren [129, 119, 137].

Das Vorgehen lässt sich auf die gezielte Medikamentengabe durch Nanogeräte an einem Anwendungsort verallgemeinern. Bereits durchgeführte Forschung schließt unter anderem die Behandlung von Diabetes [71], die Erkennung und Bekämpfung von Entzündungen [125, 165, 7] und den Einsatz von molekularen Fabriken zur Erzeugung von Medikamenten mit ein [41, 35].

Sogar intrazelluläre Nanochirurgie wurde von einigen Forschern vorgeschlagen [124].

Des Weiteren wird oft vorgeschlagen, Nanonetzwerke zur Unterstützung des körpereigenen Immunsystems einzusetzen, um z. B. die Selbstheilung zu unterstützen oder eine vorübergehende Entlastung zu ermöglichen [66, 26, 71].

Ein beliebtes Szenario ist zudem die Überwachung von Gesundheitsparametern [27]. Die makroskopischen Kapazitäten reichen oft nicht aus, um über längere Zeit plausible Daten zu generieren [66]. Für eine effektive Dauerüberwachung von Gesundheitsparametern müssen Patienten oft stationär überwacht werden, was selten in frage kommt.

Viele vorgeschlagene Technologien teilen die Eigenschaft, dass ein hohes Maß an Präzision und Ortserkennung erforderlich ist, damit Aktuatoren zur richtigen Zeit und am richtigen Ort ihre Arbeit verrichten können [66].

Die meisten der vorgestellten Szenarien erfordern eine Mindestmenge an Rechenkraft, damit sinnvoll bedingte Entscheidungen getroffen werden können [41]. Durch die eingeschränkte Rechenkraft pro Nanogerät und den Zusammenschluss zu Nanonetzwerken kann davon ausgegangen werden, dass Nanokommunikation und damit Routing und Adressierung ebenfalls wichtig sein werden. So wurden zum Beispiel Punkt-zu-Punkt-Protokolle oder Hop-Count-Ansätze vorgeschlagen, um trotz geringer Rechenkraft pro Gerät kollaborativ arbeiten zu können [160, 142].

Komplexere Vorgehensweisen erfordern deutlich höhere Rechenleistungen der einzelnen Nanogeräte [164, 37]. So benötigen Checksummenverfahren wie *zyklische Redundanzprüfung* (CRC) oder Verschlüsselungsverfahren wie der *Advanced Encryption Standard* (AES) zwar nur einfache Operationen wie XOR, diese jedoch in so hoher Anzahl, dass nicht klar ist, ob die Rechenkraft einzelner Nanogeräte dafür ausreicht [113, 116].

Leider werden selten so genaue Angaben gemacht, dass die Komplexität oder die Rechenanforderungen an einzelne Nanogeräte oder Nanonetzwerke bestimmt werden können. Oft werden zwar Formeln oder abstrakte Problemstellungen

erwähnt, die Komplexität oder mathematische Aspekte werden jedoch ausgelassen. Des Weiteren werden die daraus resultierenden Folgen für die Konzeption von Nanogeräten praktisch nie diskutiert. Dies führt zu der Frage, wie komplex Nanogeräte sein müssen, um die vorgestellten Szenarien sinnvoll bearbeiten zu können.

4.1.1 EXTRAKTION VON MATHEMATISCHEN PROBLEMSTELLUNGEN

Aus den vorgestellten Anwendungen werden mathematische Problemstellungen und Operationen oder Funktionen extrahiert. Diese mathematischen Problemstellungen entsprechen den an Nanogeräten gestellten Anforderungen.

Tabelle 4.1 zeigt und definiert alle extrahierten mathematischen Operationen. Im Verlauf dieses Abschnitts werden die einzelnen Probleme erklärt und in den Kontext von Nanonetzwerken gesetzt.

4.1.1.1 ARITHMETISCHE UND LOGISCHE OPERATOREN

Die meisten vorgestellten Szenarien benötigen wenigstens die Möglichkeit Booleans und Integer arithmetisch zu verrechnen. Dies schließt die Operationen Addition ADD, Subtraktion SUB und Multiplikation MULT sowie die der Multiplikation zugrundeliegende Operation IT-ADD ein. Des Weiteren werden grundlegende boolesche Funktionen wie AND, OR und XOR benötigt, damit einfache Entscheidungen getroffen werden können, die einen Wahrheitswert als Ausgabe haben. Gemeinsam bilden arithmetische und boolesche Operationen die Basis für Werteaggregation, die meisten Routingverfahren und komplexe Berechnungen, welche sich aus Basisoperationen zusammensetzen.

Darüber hinaus werden Komparatoren wie EQ benötigt. Ein Beispiel ist die Signum-Funktion SIGN, welche das Vorzeichen einer Zahl bestimmt. Analog bestimmen EVEN/ODD, ob eine Zahl gerade oder ungerade ist. Ebenso wird oft vergleichen, ob ein Schwellwert THRES überschritten wurde oder welcher Wert das Minimum MIN, das Maximum MAX oder das häufigste Element MAJOR einer Liste ist. Diese Operationen ermöglichen komplexes, bedingtes Verhalten und stellen die Grundlage für robuste, fehlertolerante Systeme dar.

Komplexere Szenarien benötigen aufwändigere Operationen wie Division DIV oder die Bestimmung des Modulos MOD einer Zahl. Es kann ebenfalls nötig sein den Durchschnittswert AVG einer Liste zu bestimmen oder eine Exponentiation EXP oder iterierte Multiplikation IT-MULT auszuführen. Letztere beiden Verfahren reduzieren sich aufeinander und sind äquivalent, da eine Exponentiation lediglich eine iterierte Multiplikation ist.

In der Binärdarstellung werden viele Operationen leichter. Beispielsweise ist eine Division durch den Wert 2 äquivalent zum Entfernen des letzten Bits einer Zahl. In Tabelle 4.1 werden Probleme, die als Eingabe eine Zweierpotenz erhalten, durch eine 2 im Subskript gekennzeichnet.

Problem	Signatur	Beschreibung
ADD	$\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$	Integer-Addition
EQ	$\mathbb{Z} \times \mathbb{Z} \to \{0,1\}$	Integer-Vergleich
LEQ	$\mathbb{Z} \times \mathbb{Z} \to \{0,1\}$	Integer $\leq$
GEQ	$\mathbb{Z} \times \mathbb{Z} \to \{0,1\}$	Integer $\geq$
SUB	$\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$	Integer-Subtraktion
MULT	$\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$	Integer-Multiplikation
DIV	$\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$	Integer-Division
SIGN	$\mathbb{Z} \to \{-1,0,1\}$	Signum-Funktion
INC	$\mathbb{Z} \to \mathbb{Z}$	Integer-Inkrement
AND	$\{0,1\} \to \{0,1\}$	Logisches und
OR	$\{0,1\} \to \{0,1\}$	Logisches oder
XOR	$\{0,1\} \to \{0,1\}$	Exklusives Oder
ODD, EVEN	$\mathbb{Z} \to \{0,1\}$	Gerade-/Ungerade-Test für Integer
DIV_2	$\mathbb{Z} \times \mathbb{B} \to \mathbb{Z}$	Division durch Zweierpotenz
MOD_2	$\mathbb{Z} \times \mathbb{B} \to \mathbb{Z}$	Modulo mit Zweierpotenz
INV_2	$\mathbb{Z} \to \mathbb{Z}$	Binäres Inverses
IT-MULT	$\mathbb{Z} \times \cdots \times \mathbb{Z} \to \mathbb{Z}$	Interative Multiplikation
MIN, MAX	$\mathbb{Z} \times \cdots \times \mathbb{Z} \to \mathbb{Z}$	Minimum/Maximum der Eingaben
MAJOR_2	$\mathbb{Z} \to \{0,1\}$	Binäre Mehrheit
EXP	$\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$	Exponentiation
THRES_2	$\mathbb{Z} \times \mathbb{N} \to \{0,1\}$	Prüft die letzten n positiven Bits
IT-ADD	$\mathbb{Z} \times \cdots \times \mathbb{Z} \to \mathbb{Z}$	Iterative Addition
REG_a	$\Sigma^* \to \{0,1\}$	Pattern Matching regulärer Ausdrücke
PARITY_2	$\mathbb{Z} \to \{0,1\}$	Binärer Paritäts-Test
MOD	$\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$	Modulo
AVG	$\mathbb{Z} \times \cdots \times \mathbb{Z} \to \mathbb{Z}$	Mittelwert von n Inputs
DFS, BFS	$(V,E) \times V \to \{0,1\}$	Tiefen-/Breitensuche
REACH	$(V,E) \times V \to \{0,1\}$	Graph-Erreichbarkeit
LOG_2	$\mathbb{Z} \times \mathbb{B} \to \mathbb{Z}$	Binärer Logarithmus von Integern
MEDIAN	$\mathbb{Z} \times \cdots \times \mathbb{Z} \to \mathbb{Z}$	Integer-Median

TABELLE 4.1: Formale Definitionen für Probleme, die für Nanonetzwerke von Interesse sind. Es wird angenommen, dass alle natürlichen Zahlen $\mathbb{N}$ und alle ganzen Zahlen $\mathbb{Z}$ binär repräsentiert sind – als $\{0,1\}^k, k \in \mathbb{N}^+$. Binäre Zweierpotenzen werden als $\mathbb{B}$ geschrieben. Eine 2 im Subskript entspricht einer Zweierpotenz als Eingabe, ein a einem regulärem Ausdruck.

4.1.1.2 KOMMUNIKATION

Viele der aufgezeigten Szenarien beziehen sich auf Nanonetzwerke, in denen Kommunikation eine notwendige Grundvoraussetzung ist. Ein leichtes Beispiel ist das Routingprotokoll aus [160]. Dort wird lediglich eine zweidimensionale Positionsinformation verrechnet, was lediglich ADD, GEQ, LEQ und EQ benötigt. Das Hop-Count-Routing aus [142] benötigt sogar noch weniger Voraussetzungen, um eine Nachricht zu einem Ziel zu befördern. Hier genügen die Operationen INC und ein Vergleich GEQ , LEQ oder EQ.

Komplexere Weiterleitungsmechanismen wie in [164] benötigen Division, Quadratwurzeln, Logarithmen und setzen eine große Zahl an Umgebungsinformationen voraus.

Generell kann davon ausgegangen werden, dass Nanogeräte in der Lage sein müssen, verschiedene Nachrichtentypen auseinanderzuhalten, damit sich ein Netzwerk bilden oder aufrecht halten kann. Hierfür kann es notwendig sein, dass Pattern Matching betrieben wird, bedingte Berechnungen durchgeführt werden und Adressierungsmechanismen zum Einsatz kommen.

4.1.1.3 KOMPLEXE OPERATIONEN

Neben den bereits genannten, vergleichsweise simplen Szenarien wurden ebenfalls künstliche neuronale Netze als eine Möglichkeit vorgeschlagen, Funktionalitäten zu implementieren [125]. Diese stellen ein alternatives Berechnungsmodell dar, welches auf einer Vielzahl von Neuronen basiert.

Neuronale Netze werden typischerweise als gerichteter oder ungerichteter Graph repräsentiert, weswegen Graph-Algorithmen von Interesse sind. Vor allem Suchalgorithmen wie die Tiefensuche DFS oder die Breitensuche BFS können verwendet werden, um Ausreißer in Sensordaten ausfindig zu machen. Die Funktion REACH zur Bestimmung der Erreichbarkeit kann zu Überwachungszwecken eingesetzt werden, um beispielsweise zu prüfen, ob ein Netzwerk noch verbunden ist oder eine Partitionierung eingetreten ist.

Der Einsatz von neuronalen Netzen auf Nanoebene ist noch weitestgehend unerforscht und dürfte unbekannte Herausforderungen mit sich bringen. Dennoch werden die für den Gebrauch von neuronalen Netzen benötigten mathematischen Operationen im Folgenden analysiert, um eine generelle Aussage über die Realisierbarkeit zu ermöglichen.

4.1.1.4 SPEICHER

Neben der benötigten Rechenkraft ist Speicher auf Nanoebene für viele Anwendungen unerlässlich. Für zahlreiche Szenarien ist es nötig, dass Nanogeräte ihre Position oder ihren Zustand speichern können.

Ein Speicherproblem von besonderem Interesse ist die Adressierung individueller Nanogeräte in einem Netzwerk. Um eine Adresse speichern zu können, wird wenigstens logarithmisch viel Speicher benötigt (in Relation zur Größe des Netzwerkes) und es ist nicht klar, ob Nanogeräte dies leisten können. Da für die

meisten Graphalgorithmen ebenfalls eindeutige Identifikatoren notwendig sind, benötigen diese dieselbe Menge an Speicher.

Auch für simple Szenarien wird Speicher benötigt. Um beispielsweise einen Durchschnittswert zu berechnen, müssen Werte paarweise miteinander verglichen und ein Zwischenergebnis gespeichert und aktualisiert werden. Selbiges gilt für Verfahren wie MAX, MIN, THRES und MAJOR.

4.1.1.5 PATTERN MATCHING UND PARITÄTEN

Pattern Matching durch reguläre Ausdrücke REG_a überprüft, ob eine Stringeingabe einem Muster genügt. Die zugrundeliegende Operation ist der Vergleich von Zahlen. Im einfachsten Fall muss eine Eingabe mit einem konstanten Muster verglichen werden. Dies kann zum Beispiel ein einfacher Befehlssatz für die Steuerung eines Nanogerätes sein.

Ein Spezialfall ist der Paritätscheck PARITY. Hier wird geprüft, ob eine Binärzahl ausschließlich aus Einsen besteht. Damit kann zum Beispiel die Integrität einer Nachricht geprüft werden.

4.1.1.6 SICHERHEIT

Da Nanonetzwerke häufig für den Einsatz in sicherheitskritischen Umgebungen vorgeschlagen werden, gewinnt der Aspekt Sicherheit an Bedeutung. Drei Eigenschaften sind dabei von besonderem Interesse:

- Integrität: Sicherstellung der Authentizität von Nachrichten in einem Netzwerk. Es muss erkennbar sein, ob eine Nachricht verändert wurde – z. B. durch Angreifer oder Fehler. Dies kann beispielsweise durch Signaturen, Paritäten oder Checksummenverfahren geschehen [113].
- Vertraulichkeit: Sicherstellung, dass der Inhalt einer Nachricht nur vom intendierten Empfänger gelesen werden kann. Dies geschieht in der Regel durch Verschlüsselungsverfahren wie AES [116].
- Authentizität: Eindeutige Bestimmbarkeit der Identität von Absender und Empfänger. Dies kann ebenfalls durch Verschlüsselungsverfahren in Form von Signaturen sichergestellt werden.

Die Sicherheitsverfahren haben gemein, dass vor allem die Operation XOR verwendet wird. Dasselbe gilt für das bewiesen sichere One-Time-Pad-Verfahren, bei dem eine Nachricht mit einem geheimen Binärstring durch Anwendung einer bitweisen XOR-Operation verschlüsselt wird [40]. Darüber hinaus kann zum Beispiel PARITY genutzt werden, um eine Abweichung von einem CRC-Kontrollstring aus Einsen festzustellen.

4.1.1.7 ZEIT

Neben Speicher und Rechenkraft ist eine weitere wichtige Komponente in herkömmlichen Rechnern ein Zeitgeber. Viele Algorithmen benötigen eine einfache Zeitinformation, um Ereignisse logisch anzuordnen, zu datieren oder zu sortieren. Vor allem bei sensorischen Aufgabenstellungen ist es oft wichtig zu wissen, wann eine Messung vorgenommen wurde.

In manchen Situationen genügt ein relatives Zeitempfinden. In solchen Fällen ist nur wichtig, ob ein Ereignis vor oder nach einem anderen Ereignis geschieht. So ist es beispielsweise bei der Ausschüttung von Medikamenten nur wichtig, dass dies "nach" einer Detektion von Symptomen oder Markern geschieht.

4.1.2 KLASSIFIKATION IN KOMPLEXITÄTSKLASSEN

Unter Zuhilfenahme von Reduktionen können nun die mathematischen Operationen in Komplexitätsklassen sortiert werden. Diese setzen die verschiedenen Probleme entsprechend ihrer Schwierigkeit miteinander in ein Verhältnis.

Tabelle 4.2 zeigt welche Probleme von Berechnungsmodellen gelöst werden können, die zu den Komplexitätsklassen AC^0, NC^1 und L korrespondieren. Wie in Abbildung 2.11 dargestellt, besteht ein Inklusionsverhältnis zwischen den einzelnen Klassen. Die Klasse NC^1 enthält beispielsweise auch alle Probleme der Klasse AC^0 und kann diese ebenfalls lösen.

Die hier präsentierte Einordnung muss nicht final sein. Es werden lediglich die momentan bekannten Verhältnisse aufgezeigt. Untere Schranken für Komplexität sind nur wenige bekannt. Probleme könnten folglich in Zukunft einer spezielleren Klasse zugeordnet werden.

Des Weiteren beziehen sich die hier aufgezählten Klassen auf bestimmte Schaltkreise und platzbeschränkte Turing-Maschinen und lassen sich nicht direkt auf Self-Assembly-Systeme übertragen. Dennoch hat sich gezeigt, dass für die Probleme der niedrigen Klassen wie zum Beispiel AC^0 oft einfache und kompakte Lösungen in Self-Assembly-Systemen existieren.

Ausgehend von der dargestellten Klassifikation, werden nun kurze Reduktionen gezeigt, welche die übrigen Probleme ohne Literaturangaben klassifizieren:

- INV: Das Inverter-Gatter ist eine elementare Komponente der AC^0-Schaltkreise und damit offensichtlich berechenbar.
- AND, OR: AND und OR sind ebenfalls elementare Komponenten der Schaltkreisklassen.
- XOR: Das exklusive OR lässt sich durch ein logisches OR über eine AND-Verknüpfung aller möglichen Eingaben, die genau eine „1" enthalten, bestimmen.
- SIGN: SIGN kann durch AC^0-Schaltkreise berechnet werden, indem ein vordefiniertes Vorzeichenbit durch ein Inverter-Gatter negiert wird.
- SUB: Das Subtraktionsproblem reduziert sich auf die Invertierung des Vorzeichens einer Zahl, gefolgt von einer Addition.
- INC: INC ist ein Spezialfall von ADD.
- ODD, EVEN: Das Problem ist lediglich eine Überprüfung, ob das letzte Bit der Eingabe eine „1" oder eine „0" ist.
- DIV_2: Die Division einer Binärzahl durch eine Zweierpotenz erfolgt durch Entfernen der k letzten Bits, wobei k die Potenz beschreibt.
- MOD_2: Das Modulo einer Binärzahl und einer Zweierpotenz berechnet sich durch die Ausgabe der k letzten Bits.

	Probleme aus Szenarien	Weitere lösbare Probleme
AC^0-Gerät:	ADD [15, 51]	ODD, EVEN
	SUB	DIV_2
	SIGN	MOD_2
	INC	INV
	AND	LOG_2
	OR	
	XOR	
NC^1-Gerät:	MULT [15, 51]	MIN, MAX
	DIV [51]	PARITY [15]
	EXP [80]	REG [176]
	MAJOR [15]	MOD
	THRES [15]	IT-MULT [80]
	IT-ADD [15, 51]	
	AVG	
	GEQ	
	LEQ	
	EQ	
L-Gerät:	Label	DFS [54]
	Log mem	BFS [54]
		REACH [162, 120]
		MEDIAN
Verschiedene:	*Adressierung*	
	Routing	
	Broadcasting	
	Weiterleitung	

TABELLE 4.2: Nach Komplexitätsklassen sortierte Liste von Problemen. Ein weiter unten dargestelltes Gerät ist in der Lage, alle darüber liegenden Probleme zu lösen. Kursiv geschriebene Operationen sind von unbekannter Komplexität.

- LOG_2: Der Logarithmus einer Binärzahl ist der Index der ersten „1".
- AVG: Der Durchschnitt berechnet sich durch die kumulative Addition einer Liste, gefolgt von einer Division durch die Anzahl der Einträge.
- MOD: n mod m kann durch SUB(n,MULT(DIV(n, m), m)) berechnet werden.
- MEDIAN: Der MEDIAN einer Liste kann bestimmt werden, indem alle Elemente paarweise verglichen werden und die größeren und kleineren gezählt werden.
- GEQ, LEQ, EQ: Vergleiche lassen sich über eine THRES-Berechnung realisieren.

4.1.2.1 PROBLEMKOMPENDIUM

Die präsentierten Komplexitätsklassen beinhalten weitere Probleme, welche ebenfalls durch ähnlich mächtige Geräte berechnet werden können. Einige davon werden in Tabelle 4.2 in der dritten Spalte dargestellt.

Des Weiteren können folgende weitere Probleme durch L-Geräte gelöst werden:

- ERREICHBARKEIT/PFAD/ZUSAMMENHANG [162, 120].
- BAUMISOMORPHIE [81].
- BIPATIT-Test[31].
- PARITÄTS-Test [30].

4.1.2.2 NICHT KATEGORISIERBARE PROBLEME

Das Problem einen eindeutigen Identifikator speichern zu können (Label) und die Nutzung von logarithmisch viel Speicher (Log mem) sind keine Probleme im klassischen Sinne. Sie stellen Aussagen über die Speicherkapazität von Nanogeräten dar. Beide Einträge werden von logarithmisch platzbeschränkten Turing-Maschinen abgedeckt.

Die kursiven Einträge aus Tabelle 4.2 sind von unbekannter Komplexität, da ihre Implementierungen stark variieren. Da sie jedoch für Netzwerke unerlässlich sind, sind sie in der Tabelle aufgeführt. Für viele dieser Algorithmen wird wenigstens linear viel Speicher benötigt, welcher in einem Nanonetzwerk mit Millionen Nanoknoten undenkbar ist.

- Adressierung: Für die Adressierung von Nachrichten wird in klassischen Rechensystemen wenigstens logarithmisch viel Speicher benötigt. Werden Self-Assembly-Systeme verwendet, so kann eingeschränkt über Klebersequenzen fester Länge adressiert werden. Allerdings wird dadurch nur eine vereinfachte Version des Problems gelöst, da die Länge der Sequenzen zwar beliebig, aber fest ist.
- Routing: Routing setzt oft a-priori-Wissen über die zugrundeliegende Netzwerkstruktur voraus, welches für Nanogeräte zugänglich gemacht werden muss. Das Thema wird in der Nanonetzwerkgemeinschaft intensiv erforscht. Besonders gute, verteilte Verfahren zur Bestimmung von Distanzen und Wegen (verteilter Bellman-Ford Algorithmus) benötigen $\mathcal{O}(\sqrt{n})$ große Routingtabellen [146].

- Broadcasting & Nachrichtenweiterleitung: Beide Operationen können Teil von Routingmechanismen sein. Dieses Verhalten wird nicht durch Schaltkreise oder Turing-Maschinen abgebildet.

4.1.3 OPERATIONSSELEKTION FÜR NANONETZWERKE

Um für möglichst viele mögliche Szenarien einen Machbarkeitsbeweis zu bieten, werden folgende Probleme in späteren Kapiteln analysiert, als Nanonetzwerke konzipiert und simuliert:

- THRES,
- ADD und
- AND.

Diese Probleme sind von besonderem Interesse, da viele weitere Probleme sich auf sie reduzieren und sie in den beliebtesten Anwendungsszenarien verwendet werden. THRES kann dabei eingesetzt werden, um die Signifikanz einer Messung zu ermitteln, was in vielen Szenarien von Nutzen ist. Außerdem lassen sich Vergleichsoperationen und MAJOR auf Basis eines THRES realisieren, was zusammen mit Invertern eine Turing-vollständige Logik ergibt. AND kann eingesetzt werden, um einen verteilten Konsens über eine Messung zu bilden. ADD bildet außerdem die Grundlage für Routingalgorithmen. Zum Beispiel ist es beim Hop-Count-Routing nötig, eine Zahl zur Entfernungsmessung zu erhöhen. ADD stellt für fast alle komplexeren Verfahren eine notwendige Grundvoraussetzung dar.

Im folgenden Abschnitt werden Simulatoren auf ihre Tauglichkeit zur Simulation der extrahierten Problemstellungen untersucht und bewertet.

4.2 SIMULATOREN

Für eine ganzheitliche Simulation ist es vorteilhaft, ein Simulationswerkzeug zu verwenden, welches sowohl Selbstzusammensetzungsprozesse als auch Netzwerkverhalten von Assemblies analysiert. Self-Assembly-Simulatoren sind für die Prüfung der Zielstellung unabdingbar, weswegen diese untersucht werden.

Dieser Abschnitt befasst sich mit bereits existierenden Simulationstechnologien. Es werden zuerst Tile-Assembly-Simulatoren und im Anschluss Netzwerksimulatoren für die Simulation der präsentierten Modelle gesichtet und auf ihre Eignung geprüft.

4.2.1 ISU TAS

Der Simulator ISU TAS von Matthew J. Patitz unterstützt die Evaluation von drei verschiedenen Modellen [111]. Das aTAM, das kTAM und das 2HAM können effizient simuliert werden. Da diese drei Modelle für die Evaluation verschiedener Aspekte von Nanonetzwerken eingesetzt werden können, ist das Programm ISU TAS ein geeigneter Kandidat. Die Simulation von zahlreichen Assemblies in einem Netzwerk wird nicht angeboten. Des Weiteren werden dreidimensionale Assemblies nicht unterstützt.

Das aTAM ist dabei die simpelste Form eines Self-Assembly-Systems. In jedem Zeitschritt wird im ISU TAS ein zufälliges, passendes Tile dem Assembly an einem zufälligen Ort der Wachstumsfront hinzugefügt. In jedem Zeitschritt können mehrere Tiles passend sein. Das aTAM kann im Zuge der Evaluation genutzt werden, um zu prüfen, ob die gewünschten Strukturen erwachsen. Es können mehrere Seed-Tiles auf einmal untersucht werden. Dies gilt sowohl für das aTAM-Modul als auch für das im folgenden Absatz vorgestellte kTAM-Modul.

Das aTAM-Modul wird genutzt, um besonders große Assemblies zu prüfen, da dies im kTAM-Modul zu viel Zeit in Anspruch nimmt.

Um Fehlerverhalten unter realistischen Bedingungen zu untersuchen, kann das kTAM-Modul des ISU TAS-Simulators eingesetzt werden. Dieses verwendet dasselbe Tileset wie das aTAM-Modul. Der Temperaturparameter τ wird durch Raten festgelegt, die beschreiben wie wahrscheinlich es ist, Tiles abzulösen oder hinzuzufügen. Geeignete Werte entsprechen den Temperaturen 1, 2 oder 3. Für größere Werte bietet die Software keine Unterstützung. Da jedoch maximal dreidimensionale Systeme im Zuge dieser Arbeit von Interesse sind, genügt die Auswahl.

Das 2HAM-Modul verwendet ebenfalls dasselbe Tileset. Es basiert jedoch auf einer anderen Simulationsgrundlage. Es wird kein einzelnes Assembly gebildet, sondern in jedem Zeitschritt untersucht, welche neuen Assemblies sich aus den zurzeit verfügbaren Assemblies neu bilden lassen. Sobald keine neuen Assemblies mehr möglich sind, bricht die Simulation ab. Das 2HAM kann nur bei Temperatur 1 oder 2 simuliert werden.

Das 2HAM-Modul wird verwendet, um Szenarien mit keinem dedizierten Seed-Tile zu untersuchen. Jedes Tile kann die Rolle eines Seed-Tiles einnehmen und potentiell einen Assemblyprozess beginnen. Es wird jedoch nicht der Assemblyprozess selbst untersucht, sondern lediglich der Raum der möglichen Ergebnisse und Zwischenergebnisse sowie deren Stabilität. Für eine gründlichere Analyse sind erweiterte Werkzeuge nötig. Das Modul reicht jedoch aus, um ungewollte Interaktionen von Zwischenprodukten eines Assemblyprozesses zu untersuchen.

In Kapitel 6 werden sowohl die kTAM-Implementierung als auch die 2HAM-Implementierung genutzt, um die konzipierten Tilesets zu evaluieren. In seltenen Fällen kommt das aTAM-Modul zum Einsatz. Die Syntax der Software wird ebenfalls in Kapitel 6 erläutert.

4.2.2 XGROW

Das Programm Xgrow von Eric Winfree ist für die Simulation von Berechnungen entwickelt worden [175, 171]. Der Simulator ist in der Programmiersprache C geschrieben und für Windowssysteme optimiert worden. Er wird vor allem eingesetzt, um Fehlerverhalten auf Nanoebene in Self-Assembly-Systemen zu untersuchen.

Der Simulator verfügt über einige Erweiterungen und darauf aufbauende Werkzeuge:

- Xtile: Ein Tool, welches Tilesets transformiert. Ein Beispiel sind Proofreading-Tilesets [180].
- R-TAM: Ein Tool zur Simulation der thermodynamischen Effekte von DNA [145].

Xgrow implementiert das aTAM und das kTAM. Xgrow kann zahlreiche Assemblies auf einmal simulieren, solange diese nicht miteinander interagieren. Der Simulator ist zudem um ein Vielfaches schneller als andere Simulatoren wie ISU TAS, da zahlreiche Berechnungen in einem Quad-Tree vorberechnet werden. Dies sorgt allerdings dafür, dass nur konstant große Assemblies simuliert werden können.

Tilesets der Xgrow-Software können ebenfalls im ISU TAS verwendet werden. Da die ISU TAS-Software moderner, und damit für viele Wissenschaftler besser zugänglich, ist, wird auf die Xgrow-Software verzichtet.

4.2.3 3D-ATAM-SIMULATOR

Da Nanosensoren und Nanobots ebenfalls Teil von Nanonetzwerken sind und diese mittels Self-Assembly-Systemen konzipiert werden können, bieten sich dreidimensionale Strukturen an. Da allerdings weder Xgrow noch ISU TAS Unterstützung für dreidimensionale Assemblies bieten, ist am Institut für Telematik der Universität zu Lübeck ein 3D aTAM-Simulator entwickelt worden [11].

Dieser ist in der Lage, dreidimensionale Tilesets zu simulieren und darzustellen und kann so verifizieren, dass Architekturen für Nanogeräte sich zur gewünschten Form zusammensetzen. Die so verifizierten Tilesets können als Eingabe für kTAM-Simulationen verwendet werden.

Der 3D-aTAM-Simulator kann beliebig große, dreidimensionale Assemblies simulieren und darstellen, da auf die Implementierung von D-Bäumen verzichtet wird.

4.2.4 3D-KTAM-SIMULATOR MIT FEHLERKORREKTUR

Da zudem keine dreidimensionale Variante des kTAM als Simulationswerkzeug existiert, wurde diese Variante ebenfalls am Institut für Telematik der Universität zu Lübeck entwickelt [8]. Das Werkzeug implementiert das von Winfree etablierte kTAM in drei Dimensionen unter realistischen Bedingungen.

Des Weiteren wird Unterstützung für die Generierung von Tilesets zur Fehlerkorrektur geboten. Es ist möglich, sich für eine gegebene Struktur k-Blöcke erzeugen zu lassen und diese ebenfalls zu simulieren. Dies schließt sowohl klassisches $k \times k \times k$-Proofreading als auch Snaked-Proofreading der Größe 2 mit ein.

Der Simulator verzichtet auf den Gebrauch von D-Bäumen und erlaubt somit die Simulation beliebig großer Strukturen. Zudem ist es möglich eine Vereinfachung einzustellen, welche Positionen mit starken potentiellen Bindungen um einen Faktor bevorzugt.

4.2.5 NETZWERKSIMULATOREN

In [20] wurde eine ausführliche Analyse bestehender Simulationswerkzeuge durchgeführt. Auf Basis dieser Ergebnisse werden ausgewählte Simulationssoftwares untersucht.

Die meisten Simulationswerkzeuge für Netzwerke auf Nanoebene beziehen sich auf *Body-Area-Networks* (BANs) oder *In-Body-Nanonetworks* (IBNs). Diese modellieren makroskopische Komponenten wie Kommunikationsprotokolle und Kanalmodelle sowie einige Einschränkungen durch das Umfeld. Um der Genauigkeit von in-vitro- oder in-vivo-Experimenten zu entsprechen, ist es notwendig das Umfeld sehr genau abzubilden [21].

Bei IBNs wird zwischen einer Vielzahl von Modellen unterschieden. Diese umfassen vor allem Eigenschaften der Kommunikation als solche. Beispielsweise gibt es Ansätze zur Simulation von elektromagnetischer Kommunikation im Terahertzband, chemischer Kommunikation oder Kommunikation durch spezielle Moleküle [73, 83, 68, 114, 43, 90]. In [68] wurden die vorgestellten Modelle um die akustische Kommunikation erweitert.

Der IEEE P1906.1/Draft 1.0 Standard empfiehlt generelle Praktiken für molekulare Kommunikation auf Nanoebene, ohne eine Schnittstelle für Self-Assembly-Systeme vorzusehen [46].

Der erste Simulator für Netzwerke auf Nanoebene war NanoNS [72]. Er basiert auf dem verbreiteten Netzwerksimulator ns-2, welcher um Module erweitert wurde, die die verschiedenen neuen Kommunikationsarten auf Nanoebene zulassen.

Ein weiteres Beispiel ist der BiNS2-Simulator [62]. Durch seine modulare Architektur können auch andere Arten der Kommunikation im selben Framework durchgeführt werden. Die Wahrheitstreue des BiNS2-Simulators wurde bereits in Vergleichen mit in-vitro-Experimenten bestätigt [63].

Der Netzwerksimulator ns-3 wurde um Module erweitert, welche Simulationen auf Nanoebene zulassen. Dies schließt molekulare Kommunikation mit ein. Zwei bekannte Komponenten sind nanoNS3 [148] für elektromagnetische Kommunikation und Nano-Sim [156] für molekulare Kommunikation.

Da DNA-basierte Nanonetzwerke mittels Molekülen kommunizieren, sind lediglich Simulatoren von Interesse, welche eine Schnittstelle hierfür anbieten.

Zusammenfassend bietet keines der genannten Simulationswerkzeuge Unterstützung für Self-Assembly-Systeme. Geeignete Schnittstellen in der zugrundeliegenden Architektur sind ebenfalls nicht vorgesehen. Folglich wird im weiteren Verlauf der Arbeit auf eine genaue Netzwerksimulation verzichtet und lediglich der Selbstzusammensetzungsprozess untersucht. Der generelle Nutzen von molekularer Kommunikation ließ sich bereits in Experimenten bestätigen und es ist nicht zu erwarten, dass sich durch Self-Assembly-Systeme erzeugte Moleküle andersartig verhalten.

KAPITEL 5

GENERIERUNG VON TILESETS FÜR NANONETZWERKE

DIESES Kapitel[1] definiert Nanonetzwerke und deren Referenzarchitektur. Dafür werden Tilesets vorgestellt, welche sich zu den benötigten Komponenten von Nanonetzwerken zusammensetzen können.

Zuerst werden in Abschnitt 5.1 die grundlegenden Definitionen für Nanonetzwerke erläutert. Danach werden in Abschnitt 5.2 Tile-basierte Algorithmen vorgestellt, welche für die Erschaffung von Nanosensoren und Nanobots verwendet werden können. In Abschnitt 5.2.3 werden mögliche Nachrichtenmoleküle vorgestellt, welche logische Operationen berechnen können. Zuletzt werden in Abschnitt 5.3 die einzeln modellierten Komponenten zu Nanonetzwerken zusammengefügt, welche ausgewählte Problemstellungen lösen.

5.1 DNA-BASIERTE NANONETZWERKE

In den folgenden Abschnitten werden unterschiedliche Komponenten definiert, welche zur Erschaffung von DNA-Tile-basierten Nanonetzwerken benötigt werden. Die Idee, DNA-Tiles als Baumaterial für Nanonetzwerke, deren Berechnungen und Kommunikationsmechanismen einzusetzen, geht auf [1] von Florian Lau, Florian Büther, Regine Geyer und Stefan Fischer zurück. Alle inhaltlichen Ideen und Konzepte wurden von Florian Lau entwickelt.

Zuerst wird ein Kommunikationsmechanismus auf Basis von Tiles erläutert. Im Anschluss daran wird erklärt, wie sich Rezeptoren und Liganden mittels DNA umsetzen lassen. Danach werden die einzelnen Komponenten zu einem DNA-Tile-basierten Nanonetzwerk zusammengefasst und formal definiert. Außerdem wird eine Referenzarchitektur für DNA-Tile-basierte Nanonetzwerke vorgestellt und auf mögliche Fehler des Prozesses hingewiesen.

[1] Teile dieses Kapitels wurden bereits in [1, 6, 2] vorgestellt.

5.1.1 NACHRICHTENMOLEKÜLE

Neben Terahertzkommunikation und akustischer Kommunikation ist die molekulare Kommunikation ein häufig vorgeschlagener Mechanismus zur Kommunikation auf Nanoebene [105]. Dabei handelt es sich um Hormon-analoge Verfahren, bei denen Partikelkonzentrationen gemessen werden, um Informationen zu übermitteln.

Diese Arbeit verfolgt einen ähnlichen Ansatz, bei dem molekulare Kommunikation durch Self-Assembly-Systeme umgesetzt wird. Spezielle von Self-Assembly-Systemen erzeugte Strukturen, welche in der Lage sind Informationen zu übertragen, werden Nachrichtenmoleküle genannt. Ein Nachrichtenmolekül kann entweder ein einzelnes Tile oder ein Assembly sein.

Da Self-Assembly-Systeme ebenfalls als Berechnungsmodell eingesetzt werden können, hat diese Vorgehensweise den Vorteil, dass Berechnungen in den Zusammensetzungsprozess eines Nachrichtenmoleküls integriert werden können. Dies modifiziert das vorherrschende Paradigma der Berechnung "innerhalb" von Nanogeräten. Berechnungen werden in den Übertragungskanal ausgelagert, welcher typischerweise weniger strikten Einschränkungen unterworfen ist, da beispielsweise Platz nahezu unbegrenzt verfügbar sein kann.

Ein Nachrichtenmolekül kann so entworfen werden, dass spezielle Tiles notwendig sind, damit es zu einer vollständigen Zusammensetzung kommen kann. Sorgt man dafür, dass diese Tiles nur unter bestimmten Bedingungen ausgeschüttet werden, können diese als Eingabe für eine Berechnung interpretiert werden. Andere, ebenfalls für die Berechnung benötigte Tiles können nach Belieben im Medium vorgehalten werden [1].

Auf diese Weise kann dafür gesorgt werden, dass sich zum Beispiel ein für eine Bindungsreaktion nötiger Ligand erst an einem Nachrichtenmolekül bildet, sobald eine Berechnung finalisiert wurde. Ein Ligand ist der Teil eines Moleküls, welcher eine Bindung mit einem Rezeptor eingehen kann.

Tiles und Assemblies unterliegen der Brownschen Molekularbewegung. Diese kann als Verteilungsmechanismus genutzt werden, um Tiles an die benötigten Stellen zu befördern. Der Prozess ist maßgeblich vom Zufall gesteuert, weswegen eine große Zahl von Nachrichtenmolekülen erforderlich ist.

Jedes Entscheidungsproblem [126] kann durch Self-Assembly-Systeme ab einer Temperatur von 2 gelöst werden [135]. Es ist jedoch unklar, ob Nanoroboter Berechnungen auf dieselbe Art und Weise wie makroskopische Computer durchführen werden und ob der Platz für komplexe Berechnungen auf Nanoebene ausreicht [5].

Da Self-Assembly-Systeme eine Vielzahl von Problemen auf einmal lösen und diese bereits im Labor erzeugt werden können, erscheint es sinnvoll diese bereits erprobte Technologie einzusetzen.

Abbildung 5.1(a) veranschaulicht die Idee anhand eines Tilesets für ein Nachrichtenmolekül, welches ein 4 Bit-AND berechnet. Als Eingabe für die Berechnung werden Input-Tiles (gelb) verwendet, welche unter bestimmten Bedingungen

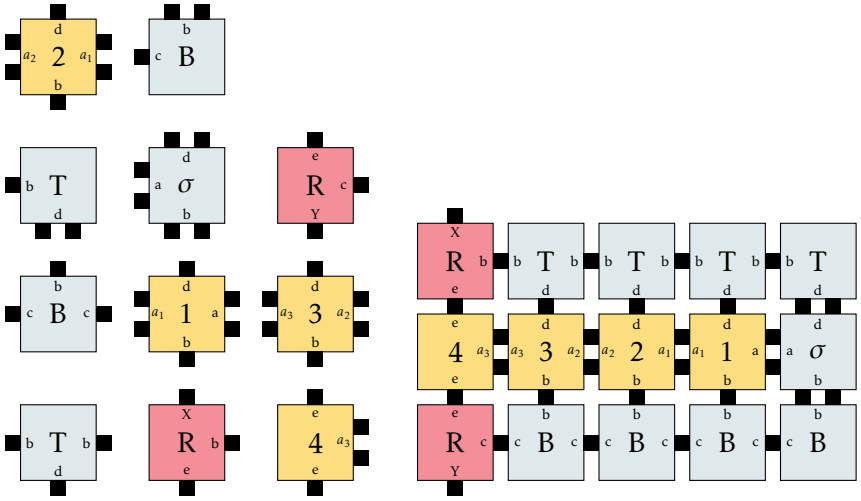


ABBILDUNG 5.1: (a) Ein Tileset, welches sich bei der Temperatur 2 zu einem 4 Bit-AND zusammensetzt. (b) Das resultierende Nachrichtenmolekül.

von Nanosensoren ausgeschüttet werden. *Input-Tiles* beschreiben Tile-basierte Instanziierungen von Algorithmen. Im Folgenden wird ein Tileset, welches ein algorithmisches Verhalten beschreibt, auch *Algorithmus* genannt, wenn keine Verwechslungsgefahr besteht.

Die Farben der Tiles kodieren verschiedene semantische Eigenschaften. Rote Tiles sind Liganden und können an Nanoroboter binden. Ein Ligand besteht aus einem oder mehreren ungebundenen Klebern an einem terminalen Assembly. Ozeangrüne Tiles sind *Framework-Tiles*, welche bedingungslos vorhanden sind und weiteres Wachstum des Assemblies verhindern, indem keine Kleber nach außen bereitgestellt werden. Des Weiteren erhöhen sie die Stabilität des Nachrichtenmoleküls. Das gelbe Tile σ ist das Seed-Tile für das kTAM- oder das aTAM– im 2HAM ist jedes Tile als Seed-Tile zu betrachten. Die Labels in der Mitte der Tiles können für die Berechnung von Funktionsproblemen genutzt werden. In diesem Beispiel dienen sie allein der Unterscheidbarkeit. Der Wahrheitswert eines Tiles wird hier durch die Anwesenheit kodiert.

Abbildung 5.1(b) zeigt das resultierende Nachrichtenmolekül. Das Nachrichtenmolekül benötigt die Anwesenheit aller Tiles aus dem gezeigten Tileset, ansonsten kann weder die Berechnung noch das Nachrichtenmolekül fertiggestellt werden.

Das präsentierte Molekül kann auf k Bits erweitert werden. Werden Input-Tiles nur bei der Detektion eines Events wie z. B. Krankheitsmarkern ausgeschüttet, so kann ein verteilter Konsens über ein Ereignis in einem Netzwerk etabliert werden. Durch diese Strategie kann zudem die Anzahl falsch positiver Entdeckungen signifikant reduziert werden.

DEFINITION 5.1 – Ein Nachrichtenmolekül $\mathcal{M}_\Phi$ ist ein Tileset T, welches eine Formel Φ berechnet und bei Abschluss der Berechnung einen Liganden bildet.

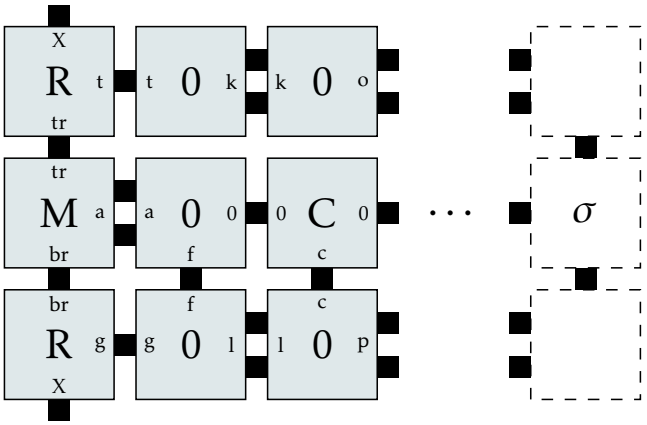


ABBILDUNG 5.2: Generischer Ligand eines Nachrichtenmoleküls.

5.1.2 LIGANDENBILDUNG

Als *Liganden* werden Bindungsstellen an Nachrichtenmolekülen bezeichnet, welche eine korrekte Bindung mit Nanorobotern ermöglichen. Diese werden durch die Kleber von Tiles modelliert.

Der Designprozess von Liganden kann schwierig sein. Die variable Größe und Form von Nachrichtenmolekülen erfordert beispielsweise zusätzliche Tiletypen, um immer einen uniformen Ansatzpunkt für die Zusammensetzung eines Liganden zu schaffen. Des Weiteren darf er sich nur dann bilden, wenn eine Berechnung erfolgreich abgeschlossen wurde. Abbildung 5.2 zeigt ein Beispiel für einen Liganden. Da es sich um ein System mit Temperatur 2 handelt, können die beiden Tiles mit den Labeln „R" erst binden, sobald das Tile mit dem marker „C" gebunden ist. Dieses hängt wiederum von der erfolgreichen Zusammensetzung eines angrenzenden, passenden Assemblies ab – hier durch weiße Tiles gekennzeichnet.

5.1.3 NACHRICHTENREZEPTOREN

Rezeptoren sind die Teile von Nanorobotern, welche Nachrichtenmoleküle binden können. Sie werden durch Tiles mit entsprechenden Klebern modelliert. Diese lassen sich einfach durch Assemblies erzeugen. Anders als Liganden sind sie nicht an die erfolgreiche Berechnung einer Funktion gebunden und somit statisch.

Rezeptoren können eine beliebige Form haben, solange sie überschneidungsfrei mit ihrem entsprechenden Liganden binden können. Dabei ist die Temperaturbeschränkung τ zu beachten.

Des Weiteren müssen die nach außen verfügbaren Kleber eines Rezeptors wenigstens ein Tile voneinander entfernt sein, damit es nicht zu einer vorzeitigen Bindung von Teilen des Liganden am Rezeptor kommen kann. Für alle in den folgenden Abschnitten vorgestellten Nachrichtenmoleküle kann leicht ein Rezeptor konzipiert werden.

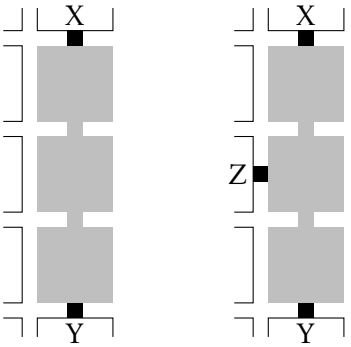


ABBILDUNG 5.3: Rezeptoren, welche Nachrichtenmoleküle stabil binden können (a) bei Temperatur 2 (b) bei Temperatur 3.

Abbildung 5.3 zeigt zwei Beispiele für mögliche Rezeptoren von Nachrichtenmoleküle. Die grauen Quadrate repräsentieren beliebige Bestandteile einer Nachricht mit Ligand. Die schwarzen Quadrate stellen einzelne Kleber der Stärke 1 dar. Die Label der Kleber des Rezeptors kodieren die Bindungsbedingung. Wenn der Rezeptor korrekt konzipiert wurde, kann er die äußersten drei Tiles (grau) des Nachrichtenmoleküls auf einmal "erkennen".

Sobald ein Nachrichtenmolekül am Rezeptor bindet, kann dies eine beliebige Reaktion auslösen. Zum Beispiel könnte ein Behältnis geöffnet werden, um Medikamente auszuschütten. Dies kann dadurch realisiert werden, dass das Nachrichtenmolekül am Rezeptor stärker bindet als der Verschlussmechanismus des Behältnisses und diesen so öffnet, wie in Abschnitt 2.3.3 gezeigt wurde.

5.1.4 MEDIZINISCHES BEISPIELSZENARIO

Eine Vision von DNA-Tile-basierten Nanonetzwerken ist deren Einsatz im Blutkreislauf des Menschen, um dort Aufgaben zu bearbeiten, die die Medizin zurzeit nicht lösen kann. Dabei könnte es sich beispielsweise um die Erkennung von Krankheitsmarkern oder die direkte Bekämpfung von Erregern durch Nanoroboter handeln. Da diese Szenarien jedoch durch eine Vielzahl unbekannter Parameter erschwert werden, wird das Vorgehen durch eine vereinfachte Vision verdeutlicht.

Um schnell und sicher Krankheitsmarker zu erkennen, kann ein DNA-Tile-basiertes Nanonetzwerk in einer Petrischale in-vitro eingesetzt werden. Die Komponenten des Netzwerks werden mit einer Probe, z. B. Blut oder Gewebe, vermengt und die nötigen Umweltbedingungen geschaffen, um die Zusammensetzung von Nachrichtenmolekülen zu ermöglichen. Nanosensoren und Nanoroboter müssen in einem früheren Schritt zusammengesetzt und mit Tiles und Medikamenten befüllt werden.

Sobald der Krankheitsmarker erkannt wurde, werden Input-Tiles ausgeschüttet und durch die Brownsche Molekularbewegung verteilt. Die Konzentration aller Komponenten kann in einer Petrischale beliebig hoch sein. Eine zu hohe Konzen-

tration der Komponenten könnte im menschlichen Körper allerdings Schaden anrichten. Es ist unklar, ob Nachrichtenmoleküle sich bei einer unschädlichen Konzentration im menschlichen Körper zusammensetzen können.

Sobald sich ein Nachrichtenmolekül zusammengesetzt hat, verhält es sich wie beliebige andere Moleküle, welche zur Kommunikation eingesetzt werden. Häufig werden Calcium Ionen zur Kommunikation vorgeschlagen, deren Konzentration gemessen wird [105]. Nach einer vollständigen Zusammensetzung kann die Nachricht von einem Nanobot empfangen werden, welcher darauf reagieren kann.

Das Szenario ist in Abbildung 5.4 dargestellt. Im Szenario kommt eine Vielzahl von Nanosensoren und ein Nanoroboter vor. Es gibt vier verschiedene Arten von Nanosensoren, welche vier verschiedene Input-Tiles ausschütten. Es wird das Tileset aus Abbildung 5.1 (a) verwendet.

In der ersten Phase messen Nanosenoren, ob Marker für ein definiertes Event vorhanden sind. Sollte dies der Fall sein, werden in Phase 2 Tiles ausgeschüttet. Diese finden sich in Phase 3 zu einem Nachrichtenmolekül zusammen. Dies kann nur geschehen, wenn wenigstens vier verschiedene Arten von Nanosensoren ein Event gemessen haben. Dabei wird ein 4 Bit-AND berechnet. Sobald ein Nachrichtenmolekül vollständig zusammengesetzt wurde, kann es in Phase 4 an einen Nanobot binden. Sobald diese Bindung stattgefunden hat, kann in Phase 5 ein Medikament ausgeschüttet werden.

Alle vorgestellten Komponenten lassen sich in kTAM-Simulatoren verifizieren [8, 111], wie in Kapitel 6 gezeigt wird.

5.1.5 SZENARIO-MODULARISIERUNG

Das in Abbildung 5.4 präsentierte Szenario verdeutlicht eine Vision von medizinischen Nanonetzwerken. Unter realistischen Bedingungen lassen sich einfachere Aufgabenstellungen im Labor umsetzen. Um Wet-Lab-Experimente zu erleichtern, kann das Szenario in fünf Module zerlegt werden, welche später zu einem Ganzen zusammengefügt werden könnten.

Das Szenario kann in folgende generische Phasen zerlegt werden:

1. Das Erkennen eines beliebigen Markers durch einen Nanosensor. Li et al. haben bereits gezeigt, dass Medikamente nach der Detektion von Markern ausgeschüttet werden können [95, 41].
2. Das Lagern von Tiles und Medikamenten in Nanosensoren und Nanobots. Eine Variation der Aufgabenstellung wurde bereits in [95, 34] gelöst.
3. Die Zusammensetzung eines Nachrichtenmoleküls unter Laborbedingungen. Viele Experimente mit DNA-Tiles haben bereits komplexe Strukturen erzeugen können [94].
4. Die Anbringung eines Nachrichtenmoleküls an einem Nanobot [50]. Dies wird generell durch das 2HAM beschrieben.
5. Die Freisetzung von Tiles oder Medikamenten, siehe 1. [95].

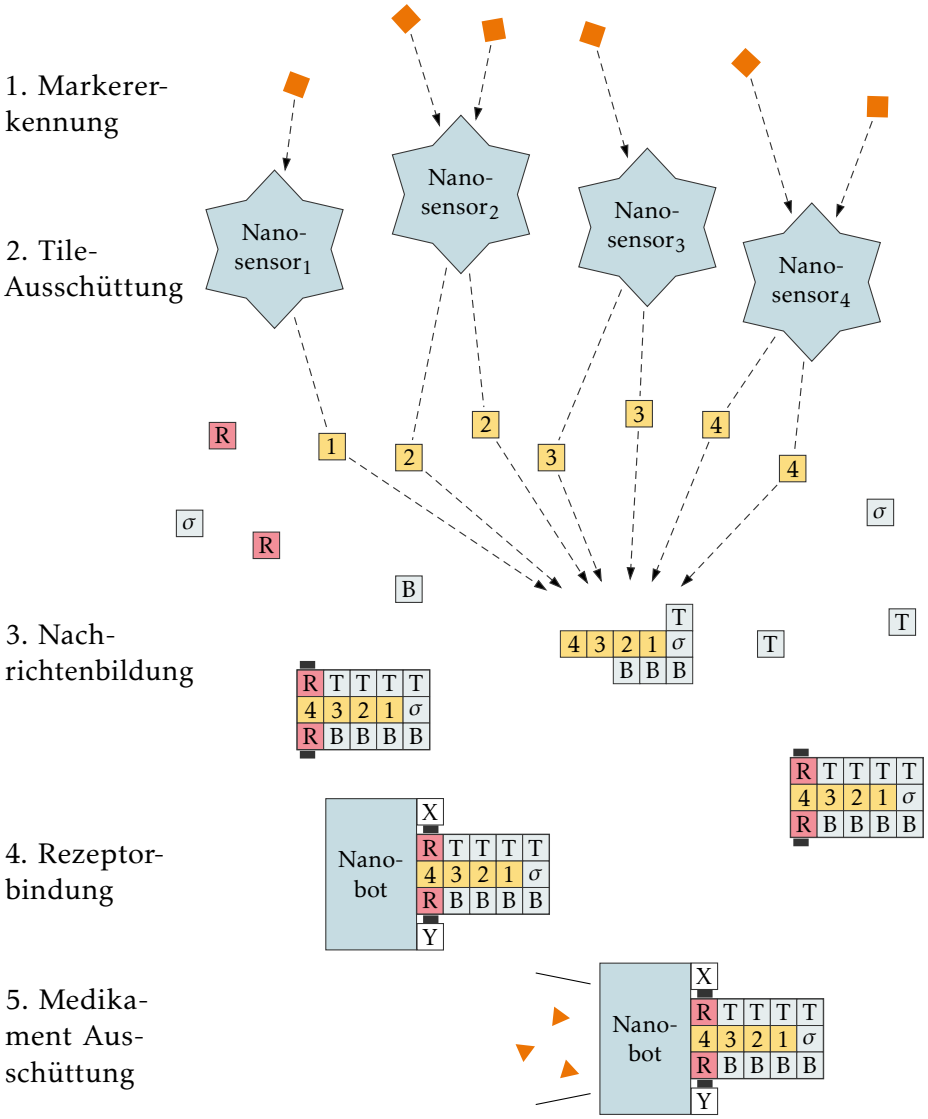


ABBILDUNG 5.4: Nanonetzwerk, welches Marker erkennt und daraufhin ein logisches 4 Bit-AND berechnet und so einen verteilten Konsens über eine Detektion schafft.

Variationen aller Module wurden bereits mittels DNA-basierter Methoden gelöst. Die Vermutung liegt nahe, dass auch das gesamte Szenario lösbar ist. Generell sind beliebige DNA-basierte Technologien miteinander kombinierbar. DNA-Origami und Tiles können im selben Anwendungsfall zum Einsatz kommen. Es ist denkbar, dass Nachrichtenmoleküle durch Tiles implementiert werden, wohingegen Nanosensoren und Nanoroboter durch DNA-Origami erzeugt werden.

5.1.6 DEFINITION VON NANONETZWERKEN AUS TILES

Folgender Abschnitt definiert DNA-Tile-basierte Nanonetzwerke auf Basis des präsentierten Szenarios.

DEFINITION 5.2 – Sei $\mathcal{T} = (T, \sigma, \tau)$ ein *Tile Assembly System*, wobei T ein endliches Tileset ist, σ ein Seed-Assembly und $\tau \in \mathbb{N}^+$ die Temperatur des TAS. Ein *DNA-Tile-basiertes Nanonetzwerk* $\mathcal{N}_\Phi$ ist ein Nanonetzwerk. Die Komponenten sind durch die Menge $\mathcal{N}_\Phi = \mathcal{N}_{Se} \cup \mathcal{N}_R \cup \mathcal{M}_\Phi$ gegeben, wobei $\mathcal{N}_{Se}$ eine Menge von DNA-basierten Nanosensoren und $\mathcal{N}_R$ eine Menge von DNA-basierten Nanorobotern ist. $\mathcal{M}_\Phi$ ist ein Tileset für eine Menge von Nachrichtenmolekülen. $\mathcal{N}_{Se}, \mathcal{N}_R$ und $\mathcal{M}_\Phi$ können korrekt aus $\mathcal{T}$ zusammengesetzt werden. Φ ist die vom Nachrichtenmolekül berechnete Funktion.

Die Mengen der Nanoroboter $\mathcal{N}_R$ und Nanosensoren $\mathcal{N}_{Se}$ können alternativ als DNA-Origami-Gerät zur Verfügung gestellt werden. In diesem Fall müssen die Komponenten $\mathcal{N}_{Se}$ und $\mathcal{N}_R$ nicht korrekt aus einem Tileset T entstehen, sondern entstehen aus einem langen Einzelstrang und DNA-Klammern. Da DNA-Origami bereits für die Konstruktion von Boxen erprobt wurde, ist es wahrscheinlich, dass dieser Ansatz schneller im Labor umsetzbar ist.

5.1.7 REFERENZARCHITEKTUR

Bei der weiteren Modellierung von Nachrichtenmolekülen und Nanonetzwerken wird von einer *Referenzarchitektur* ausgegangen. Dabei wird vorausgesetzt, dass die Topologie von DNA-Tile-basierten Nanonetzwerken gewissen Anforderungen genügt. Abbildung 5.5 stellt diese grafisch dar. Im Umfeld Γ des DNA-Tile-basierten Nanonetzwerkes sind $k \geq 0$ Marker für einen zu messenden Sachverhalt vorhanden.

Ein DNA-Tile-basiertes Nanonetzwerk beinhaltet immer eine sehr große Anzahl Nanosensoren, welche eingesetzt werden, um Marker zu messen. Eine *sehr große Anzahl* beschreibt eine ausreichend hohe Konzentration an Komponenten, damit es in überschaubarer Zeit zu einer Bindungsreaktion kommt. Ein sinnvolles Intervall sind 60 Minuten. Eine sehr große Anzahl an Komponenten ist notwendig, da sowohl die Zusammensetzung als auch der Transport von Nachrichtenmolekülen auf Brownscher Molekularbewegung basieren und somit maßgeblich vom Zufall gesteuert werden.

Jeder Nanosensor beinhaltet des Weiteren eine große Menge Input-Tiles A_n^i, welche im Falle der Detektion eines Markers ausgeschüttet werden.

Zusätzlich ist das Medium mit allen weiteren benötigten Tiles, welche nicht direkt an der Berechnung beteiligt sind, gefüllt. Es sind ausreichend Tiles jedes Typs vorhanden, um für jedes Input-Tile ein Nachrichtenmolekül zusammenzusetzen.

Abhängig von der Anzahl der Nanosensoren und der Tatsache, ob Marker vorhanden sind, kann eine sehr große Anzahl Nachrichtenmoleküle entstehen.

Eine sehr große Anzahl Nanobots ist immer Teil eines DNA-Tile-basierten Nanonetzwerkes. Dies ist notwendig, damit ausreichend Medikamente oder andere Nutzlasten transportiert werden können und genügend potentielle Bindungsorte für Nachrichtenmoleküle vorhanden sind.

5.1.8 FEHLER IN NACHRICHTENMOLEKÜLEN

Wie in Kapitel 3 beschrieben wurde, sind auch Nachrichtenmoleküle anfällig für Growth-, Facet- und Nucleation-Errors. Facet- und Growth-Errors können mittels Blockersetzungsstrategien, wie Snaked- oder $k \times k$-Proofreading, deutlich verringert werden [49, 134]. Growth-Errors werden durch beide Verfahren reduziert. Damit ein k-Block falsch wächst, müssen im Vergleich zum ursprünglichen Fall in zweidimensionalen Self-Assembly-Systemen k^2 Fehler auftreten. Snaked-Proofreading verhindert im Zweidimensionalen zudem Facet-Errors in einer Dimension – im Dreidimensionalen in allen Dimensionen, wie in Kapitel 3 gezeigt wurde.

Die ursprüngliche Logik bleibt dabei erhalten, weswegen beide Verfahren geeignet sind, um Nachrichtenmoleküle robuster zu gestalten.

Nucleation-Errors sind schwieriger zu handhaben. Da die Anwesenheit von zahlreichen Tiles zu jeder Zeit vorausgesetzt wird und der Self-Assembly-Prozess vom Zufall gesteuert wird, treten Nucleation-Errors häufig auf und müssen als Teil der Mechanik betrachtet werden. In aTAM und kTAM sind Nucleation-Errors per Definition ausgeschlossen, da der Self-Assembly-Prozess immer von einem Seed-Tile gestartet wird. In der Realität sind alle Tiles als potentielle Seed-Tiles zu betrachten.

Die beiden linken Spalten des Nachrichtenmoleküls aus Abbildung 5.1 (b) können sich so zum Beispiel ohne das Tile σ bilden, wenn eine ungünstige Verkettung von Fehlern auftritt. Dies würde zu der Bildung eines Liganden ohne erfolgreiche Berechnung führen. Damit es dazu kommt, müssen die Input-Tiles 3 und 4 zugegen sein und ober- und unterhalb jeweils einen Facet-Error erzeugen. Die Wahrscheinlichkeit dafür ist zwar gering, das Verhalten ist jedoch nicht auszuschließen.

Dies kann durch die in Kapitel 3 beschriebenen Blockersetzungsstrategien verhindert werden. Die Kleberstruktur der Liganden muss jedoch so angepasst werden, dass kein Teil eines unvollständigen Liganden am Rezeptor korrekt binden kann.

Eine weitere Möglichkeit stellt eine modifizierte Konzeption von Rezeptor und Nachrichtenmolekül dar, welche die Anwesenheit des Tiles σ erfordert. Abbildung 5.6 zeigt eine mögliche Anpassung. Ein passender Rezeptor kann leicht

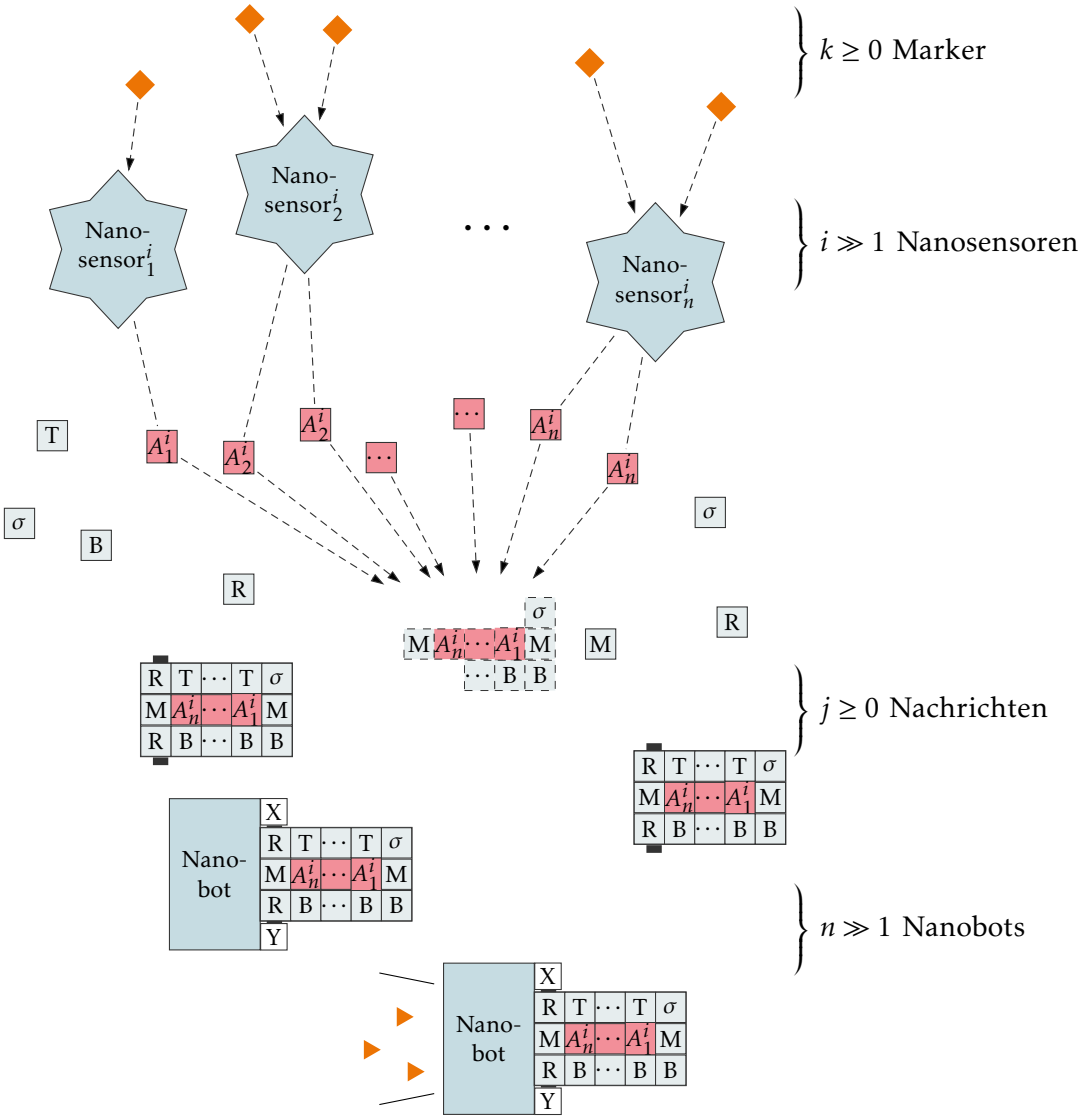


ABBILDUNG 5.5: Die allgemeine Referenzarchitektur für DNA-Tile-basierte Nanonetzwerke. Diese besteht aus $k \geq 0$ Markern, $i \gg 1$ Nanosensoren, $j \geq 0$ Nachrichtenmolekülen und $n \gg 1$ Nanobots.

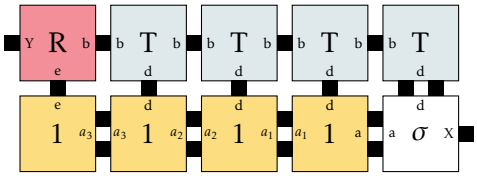


ABBILDUNG 5.6: Ein angepasstes Nachrichtenmolekül bei Temperatur 2, welches Nucleation-Errors umgeht.

konzipiert werden. Auf diese Weise wird die Anwesenheit des Tiles σ vorausgesetzt und somit Nucleation-Errors verhindert. Es ist somit schwieriger fehlerhaftes Verhalten zu erzeugen.

5.1.9 LOGISCHE KOMBINATION VON NACHRICHTENMOLEKÜLEN

Um beliebige Probleme als Berechnung innerhalb eines Nachrichtenmoleküls ausdrücken zu können, ist es notwendig die Wahrheitswerte von Nachrichtenmolekülen logisch miteinander kombinieren zu können. Dies kann durch zusätzliche, geschickt konzipierte Assemblies ermöglicht werden. Abbildung 5.7 zeigt ein Beispiel hierfür. Die hellblau hinterlegten Tiles stellen zwei Nachrichtenmoleküle N_1 und N_2 mit speziellen Liganden dar. Der lange weiße Strang aus Tiles F enthält verschiedene Rezeptoren, welche die beiden Nachrichtenmoleküle binden können. Der Ligand der Kombination beider Nachrichtenmoleküle kann sich nur bilden, wenn beide Nachrichtenmoleküle stabil binden. Der Ligand wird durch das Label „R" gekennzeichnet.

Eine Besonderheit der Konstruktion ist, dass das erste Nachrichtenmolekül N_1 einen Teil des Rezeptors x_2 für die stabile Bindung des zweiten Nachrichtenmoleküls N_2 bereitstellt. Auf diese Weise erfüllt das gesamte Assembly die Funktion eines AND-Gatters mit zwei Eingängen. Das Ergebnis kann durch den angebrachten Rezeptor an andere Nanobots kommuniziert werden und der Ligand bildet sich nur, wenn die erforderliche Anzahl Nachrichtenmoleküle an der Struktur gebunden ist.

Das Vorgehen lässt sich auf eine beliebige Anzahl von Eingabenachrichten erweitern. Natürliche Restriktionen der Stabilität sind dabei zu berücksichtigen.

5.2 TILESETS FÜR NACHRICHTENMOLEKÜLE, NANOBOTS UND NANOSENSOREN

Gemäß der vorgestellten Referenzarchitektur bestehen Nanonetzwerke aus Nanosensoren, Nanorobotern und Nachrichtenmolekülen. Sinnvolle Instanziierungen dieser Komponenten werden hier der Reihe nach vorgestellt und in Self-Assembly-Systemen modelliert. Nanosensoren und Nanoroboter werden dabei als Würfel variabler Größe konzipiert und es werden entsprechende Algorithmen für deren Erstellung vorgestellt.

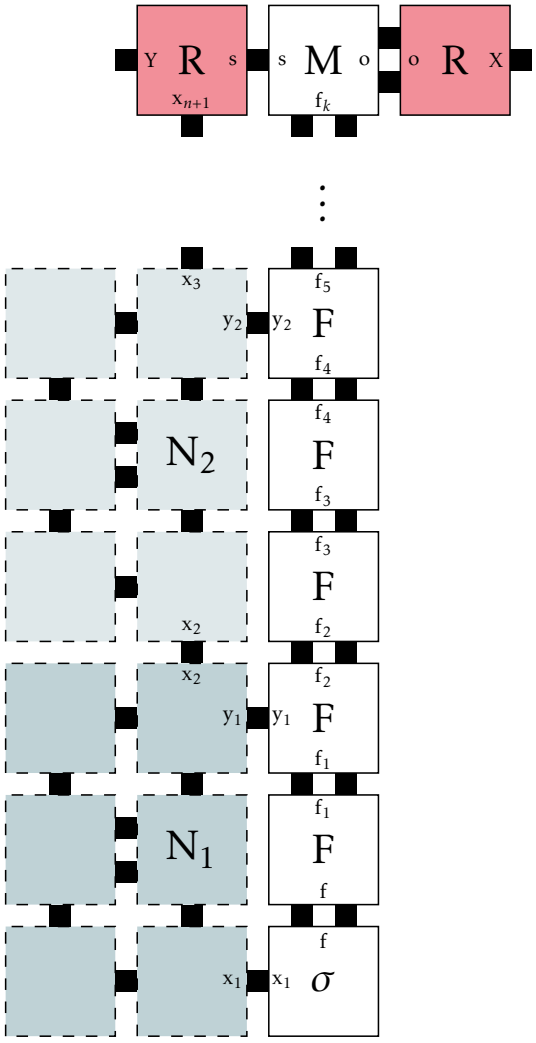


ABBILDUNG 5.7: Schematische Repräsentation eines Assemblies, welches n Nachrichtenmoleküle N_n bei Temperatur 2 miteinander kombiniert und selbst einen Liganden „R" bereitstellt.

Nachrichtenmoleküle sind, anders als Nanobots oder Nanosensoren, in der Lage Berechnungsprobleme zu lösen. Sie sind dementsprechend vielfältig und es werden zahlreiche verschiedene Architekturen für ausgewählte Probleme erörtert.

5.2.1 ALGORITHMEN FÜR NANOROBOTER UND NANOSENSOREN

In den folgenden Abschnitten werden Algorithmen präsentiert, die Tilesets für Quadrate und Würfel berechnen. Diese basieren zum Teil auf [157]. Quadrate und Würfel werden als Ausgangsstruktur für Nanosensoren und Nanoroboter verwendet. Hohle Strukturen sind dafür von besonderem Interesse, da diese mit Tiles oder Medikamenten gefüllt werden können, um diese später kontrolliert

freizugeben.

Zuerst werden relativ einfache Vorgehensweisen präsentiert, welche nach und nach verbessert werden. Des Weiteren wird ein generalisierter Algorithmus vorgestellt, der Tilesets für beliebige dreidimensionale Strukturen erzeugen kann.

5.2.1.1 NAIVER 2D-ALGORITHMUS

Dieser Abschnitt spezifiziert die naive Erzeugung von ausgefüllten Quadraten. Diese können beispielsweise als Seitenwand von Nanorobotern oder Nanosensoren eingesetzt werden. Des Weiteren besteht die Möglichkeit diese, analog zum Vorgehen von Andersen et al., zu einer dreidimensionalen Box zu falten [34].

```
1  Input: Kantenlänge n, Stärke s
2  Output: Tileset T
3
4  generateTiles(x,y,s){
5    label(t)= l;
6    col(t,(1,0)) = x+1,y; str(t, (1,0)) = s;
7    col(t, (-1,0)) = x,y; str(t, (-1,0)) = s;
8    col(t, (0,1)) = x, y+1; str(t, (0,1)) = s;
9    col(t, (0,-1)) = x, y; str(t, (0,-1)) = s;
10 }
11
12 makeTileset(n,s){
13   x,y,z = 0;
14
15   T = {};
16
17   while(x < n){
18     y = 0;
19     while(y < n){
20       l++;
21       T = T + generateTiles(x,y,s);
22       y++;
23       }
24     x++;
25     }
26   return T;
27 }
```

QUELLTEXT 5.1: Der intuitive Algorithmus MAKETILESET erzeugt für jede Position eines Quadrates einzigartige Tiletypen. [12]

Ein naiver Algorithmus erzeugt für jede Position in einem Quadrat einen einzigartigen Tiletyp, wofür $\mathcal{O}(n^2)$ verschiedene Tiletypen, im Verhältnis zur Kantenlänge n des Quadrates, benötigt werden [166, 157, 153]. Quelltext 5.1 spezifiziert die Vorgehensweise. Die Funktion `generateTiles` erzeugt für einen gegebenen Ort in einem spezifizierten Quadrat einen einzigartigen Tiletyp mit Stärke s

und Kleberlabels `col`, welche den benachbarten Orten im Quadrat entsprechen. Auf diese Weise wird sichergestellt, dass alle Tiletypen korrekt und nur auf genau eine Weise aneinander passen. Die Funktion `makeTileset` iteriert über alle Positionen des definierten Quadrates mit den Dimensionen $n \times n$ und ruft die vorherige Subroutine auf.

Durch dieses naive Vorgehen können zwar Tilesets erschaffen werden, welche sich zu beliebigen Strukturen zusammensetzen, diese benötigen jedoch die größtmögliche Anzahl unterschiedlicher Tiletypen. Dies kann bei komplexeren Nanostrukturen zu Problemen führen, da die Wahrscheinlichkeit für fehlerhafte Bindungen zunimmt, je größer die Anzahl unterschiedlicher Kleber wird.

5.2.1.2 NAIVER 3D-ALGORITHMUS

In diesem Abschnitt wird ein Algorithmus vorgestellt, der bei gegebener Kantenlänge n Tilesets für entweder einen hohlen Würfel, einen gefüllten Würfel oder einen Gitterboxwürfel erzeugt. Es wurde bereits eine Vielzahl verschiedener Würfel aus DNA erzeugt [33, 166, 153].

Die Grundidee des Algorithmus ist erneut die Erzeugung eines einzigartigen Tiletypen für jede Position der gewünschten Struktur. Auf diese Weise kann bei kleinen Strukturen die Anzahl möglicher Fehler reduziert werden, da die Klebersequenzen entsprechend unterschiedlich gewählt werden können. Da die Anzahl der realistisch unterscheidbaren Klebersequenzen der Länge k ausreichend groß ist, ist es bei kleineren Strukturen kein Problem so vorzugehen. In den meisten Fällen besteht jedoch Interesse an Strukturen, die aus möglichst wenig unterschiedlichen Tiletypen bestehen.

Quelltext 5.2 zeigt die notwendigen Schritte, um ein Tileset für die beschriebenen Würfel zu erzeugen.

In drei geschachtelten Schleifen wird jede Position eines Würfels genau einmal betrachtet. Zeile 9–12 speichern in der Variable c, ob die momentane Position Teil des inneren Würfels ($c = 0$), eine Wandposition ($c = 1$), eine Kantenposition ($c = 2$) oder eine Eckposition ($c = 3$) ist. Der Eingabeparameter p steuert auf dieser Basis, ob die jeweilige Position einen Tiletyp zugewiesen bekommen soll.

Bei Parameter $p = 0$ erzeugt Quelltext 5.2 einen gefüllten Würfel. Eine Erhöhung des Parameters erzeugt zuerst einen hohlen Würfel und zuletzt eine Gitterbox.

Zeile 16 fügt für jede Position, an der sich ein Tile befinden soll, dem Tileset einen neuen Tiletypen hinzu.

Abbildungen 5.8 (a)–(c) zeigen Zwischenschritte und ein Beispielergebnis eines simulierten Zusammensetzungsprozesses. Es wird ein Tileset für einen $5 \times 5 \times 5$ hohlen Würfel erzeugt und in drei verschiedenen Stadien dargestellt. Die unterschiedlichen Farben entsprechen jeweils einem einzigartigen Tiletyp. Die Kleber selbst wurden der Übersichtlichkeit halber weggelassen.

5.2.1.3 3D LINEARZEIT ALGORITHMUS

Die in [157] vorgestellte Vorgehensweise zur effizienteren Erzeugung von Quadraten lässt sich auf drei Dimensionen erweitern [12]. Die Idee ist, dass lediglich

```
1  Input: Kantenlänge n, Parameter p
2  Output: Tileset T
3  x,y,z = 0;
4
5  T = {};
6  while(x < n){
7   while(y < n){
8    while(z < n){
9     c = 0 ;
10    for(i in {x,y,z}){
11     if(i == 0 || i == n - 1){
12      c = c + 1;
13     }
14    }
15    if(c >= p){
16     T = T + createPositionTiletype(x,y,z,1);
17    }
18    z = z + 1;
19   }
20   y = y + 1;
21  }
22  x = x + 1;
23 }
24 return T;
```

QUELLTEXT 5.2: Naiver Algorithmus zur Erzeugung von Tilesets für dreidimensionale Würfel. Jede Position erhält einen einzigartigen Tiletyp. [12]

der untere Rand eines Quadrates unterschiedliche Tiletypen benötigt. Der Rest lässt sich mit identischen Stapeln von Tiles füllen, sodass ein Quadrat entsteht.

Noch effizienter wäre das Vorgehen, wenn zuerst eine Reihe aus einzelnen Tiletypen entstünde, dann aus Stapeln ein Quadrat und dann aus Stapeln ein Würfel.

Das Problem bei diesem Vorgehen ist, dass der Würfel nicht hohl und somit als Transportgefäß ungeeignet ist. Quelltext 5.3 verdeutlicht, wie das Vorgehen angepasst werden kann, damit es für hohle Würfel funktioniert. Zuerst wird, wie in Abbildung 5.9 (a) und (c) gezeigt, eine Kernstruktur aus Tile-Stapeln erzeugt. Diese haben an den Seiten spezielle Kleber, sodass Flächen aus Stapeln von Tiles daraus erwachsen können (Abbildung 5.9 (b)), um so einen hohlen Würfel zu formen (Abbildung 5.9 (d)).

Die Funktion `makeEdgeStackTiletype` erzeugt dabei die Tiletypen der verschiedenen Elemente des Kernstücks aus Abbildung 5.9 (c). Die Seitenflächen werden durch die korrespondierende Funktion `makeSquareSideTiletypes` erzeugt.

THEOREM 5.1 – Der Algorithmus für 3D Dreifachstapel erzeugt ein Tileset, welches korrekte $n \times n \times n$ Würfel unter Verwendung von $\mathcal{O}(n)$ Tiletypen erzeugt.

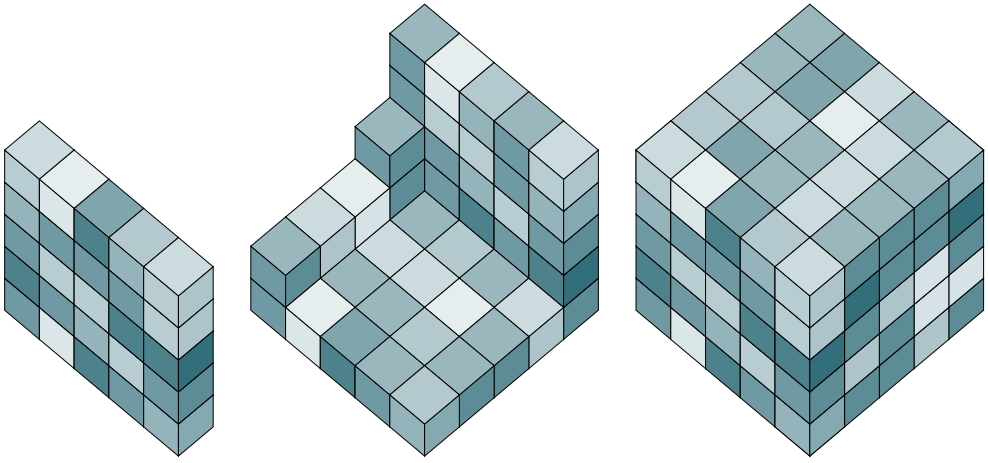


ABBILDUNG 5.8: Ein hohler Würfel mit Kantenlänge 5 in drei Stadien, welcher in einem 3D-aTAM-Simulator erzeugt wurde. An jeder Position befindet sich ein einzigartiger Tiletyp.

Beweis. Die Korrektheit des Ansatzes folgt aus Abbildung 5.9.

Die Größe des Tilesets ergibt sich aus der Summe des Kernstücks und der Flächen. Das Kernstück besteht aus fünf Stapeln mit einer Maximallänge n. Die Seiten bestehen aus Serien von insgesamt sechs verschiedenen Stapeln, die innerhalb einer Fläche wiederholt werden. Die Gesamtkomplexität entspricht folglich der Klasse $\mathcal{O}(n)$. □

Auf diese Weise genügen für einen dreidimensionalen Würfel linear viele verschiedene Tiletypen. Im nächsten Abschnitt wird ein sublineares Verfahren vorgestellt, welches allerdings eine zusätzliche konstante Anzahl Tiles benötigt.

5.2.1.4 LOGARITHMISCHER 3D-ALGORITHMUS

Durch den geschickten Einsatz von platzbeschränkten Binärzählern kann ein Quadrat aus Tiles bei Temperatur 2 durch logarithmisch viele Tiletypen erzeugt werden [157]. Dabei wird der Binärzähler in das Quadrat eingebettet. Dieser bildet sich bis zu einer vorgegebenen Zahl und schließt dann das Wachstum in y-Richtung ab. Auf diese Weise kann durch Verwendung von 22 zusätzlichen Binärzähler-Tiletypen ein Quadrat aus $\mathcal{O}(\log n)$ Tiletypen in Relation zur Kantenlänge n des Quadrates erzeugt werden.

Abbildung 5.10 zeigt ein Beispiel. Im Wesentlichen besteht das gebildete Quadrat aus einem integrierten, platzbeschränkten Binärzähler, welcher unter Einsatz von logarithmisch vielen Tiletypen ein Rechteck fester Höhe erzeugen kann. Die grauen Tiles im Binärzählers entsprechen einer logischen „1" und die weißen Tiles einer logischen „0". Es wird in Doppelreihen hochgezählt damit das Wachstum in der Höhe beschränkt werden kann. Am Binärzähler binden Tiles vom Typ A / Typ a und vom Typ B / Typ b und bilden eine Diagonale. Diese ist

```
1  Input: Kantenlänge n, Stärke s
2  Output: Tileset T
3
4  makeEdgeStackTiletype(s,n){
5   T = T + xxEdge(n); T = T + xyStack(n-1);
6   T = T + zEdge(n-1);
7   T = T + zxEdge(n-1); T = T + zyStack(n-1);
8   return T;
9  }
10
11 makeSquareSideTiletypes(s,n){
12  T = T + x1Side(n); T = T + z1Side(n-1);
13  T = T + y1Side(n); T = T + y2Side(n-1);
14  T = T + z1Side(n); T = T + z2Side(n-1);
15  return T;
16 }
17
18 makeTileset(n,s){
19
20  T = {};
21
22  T = T + makeEdgeStackTiletype(4,2);
23
24  T = T + makeSquareSideTiletypes(4,2);
25
26  return T;
27 }
```

QUELLTEXT 5.3: Pseudocode-Quelltext, der ein Tileset für einen hohlen $n \times n \times n$ Würfel in Linearzeit erzeugt.

durch die Dimensionen des Binärzählers begrenzt. Der übrige Platz wird von den Typ 1 (linke Hälfte) und Typ 0-Tiles (rechte Hälfte) gefüllt. Das Quadrat kann erst in x- oder y-Richtung weiter wachsen, wenn sich die vorherige Reihe gebildet hat.

Das Verfahren lässt sich auf drei Dimensionen verallgemeinern, indem für jede Seite eines angestrebten Würfels ein semantisch identisches Tileset angelegt wird, welches sich bilden kann, sobald das letzte Tile einer äußeren Seite des Quadrates gebunden wurde. Das Ursprungsquadrat erhält in der dritten Dimension Kleber auf der Ober- oder Unterseite. Der Self-Assembly-Vorgang wiederholt sich folglich sechs Mal, bis sich ein hohler Würfel mit der Kantenlänge n gebildet hat.

Die Tilekomplexität beläuft sich weiterhin auf $\mathcal{O}(\log n)$, da maximal die sechsfache Menge an Tiletypen verwendet werden muss. Für ein Quadrat der Kantenlänge 50 werden so 28 Tiles benötigt. Ein entsprechender hohler Würfel würde wenigstens 168 Tiletypen benötigen. Der zuvor vorgestellte Linearalgorithmus würde für dieselbe Struktur 1152 Tiletypen benötigen. Bereits ab einer Kantenlänge von 3–4 ist die zusätzliche konstante Anzahl Tiletypen für den Binärzähler ausgeglichen und der logarithmische 3D-Algorithmus ist effizienter.

Da der logarithmische 3D-Algorithmus die effizienteste der hier vorgestellten

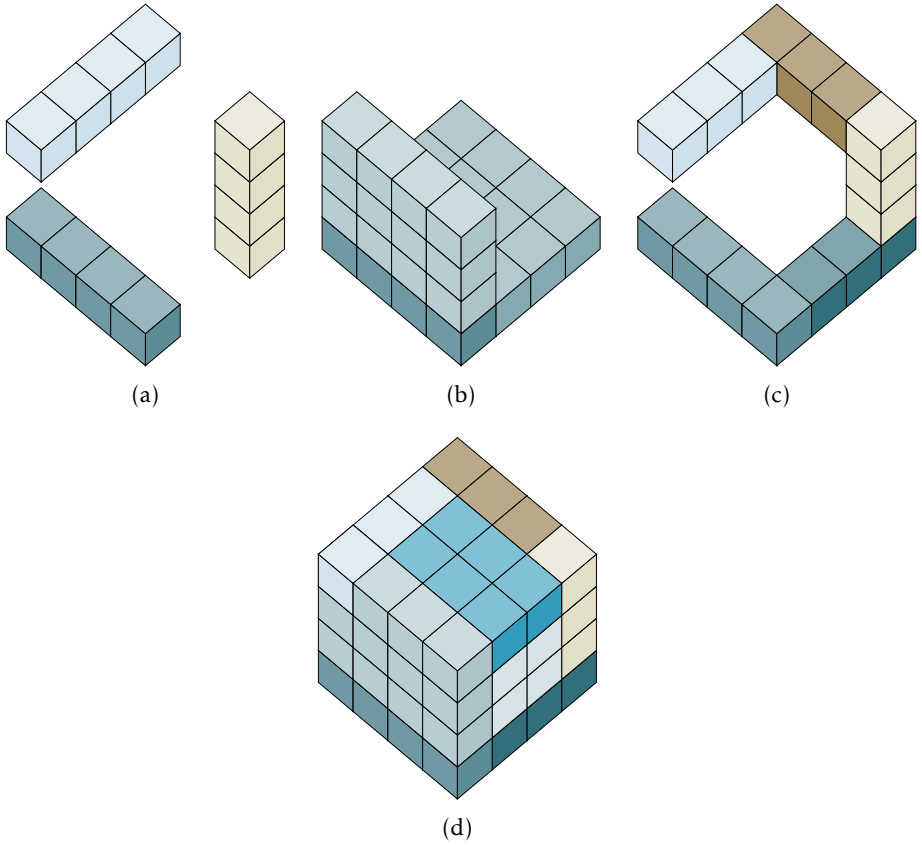


ABBILDUNG 5.9: Konzeptuelle Schritte eines Algorithmus zur Erzeugung von hohlen Würfeln in $\mathcal{O}(n)$.

Vorgehensweisen ist, wird dieser für die Modellierung von Nanosensoren $\mathcal{N}_{Se}$ und Nanorobotern $\mathcal{N}_R$ verwendet. Nanosensoren $\mathcal{N}_{Se}$ und Nanoroboter $\mathcal{N}_R$ werden im Folgenden als Tileset modelliert.

Eines der sechs Quadrate wird als Öffnungsmechanismus verwendet. Dazu wird dieses nur an einer Seite des Würfels befestigt. Die gegenüberliegende Seite erhält genau der Temperatur $\tau - 1$ entsprechend viele Kleber. Auf der gegenüberliegenden Seite existiert ein weiterer Kleber der Stärke $\tau - 1$. Auf diese Weise ist es möglich, dass ein anderes Molekül stärker bindet und so die Box öffnet.

DEFINITION 5.3 – Ein *Tile-basierter Nanosensor* $\mathcal{N}_{Se}^{i,k}$ ist ein Tileset, dass sich zu einem hohlen Würfel der Größe $2 < i \in \mathbb{N}^+$ zusammensetzt. $k \in \mathbb{N}^+$ beschreibt die Größe der Blockersetzungsstrategie, sofern eine verwendet wird. Sofern nicht anders angegeben, gilt $k = 1$. Ein Nanosensor $\mathcal{N}_{Se}^{i,k}$ verfügt über einen Rezeptor R, welcher mit bestimmten Markern korrekt bindet.

DEFINITION 5.4 – Ein *Tile-basierter Nanobot* $\mathcal{N}_R^{i,k}$ ist ein Tileset, dass sich zu einem hohlen Würfel der Größe $2 < i \in \mathbb{N}^+$ zusammensetzt. $k \in \mathbb{N}^+$ beschreibt die Größe der Blockersetzungsstrategie, sofern eine verwendet wird. Sofern nicht

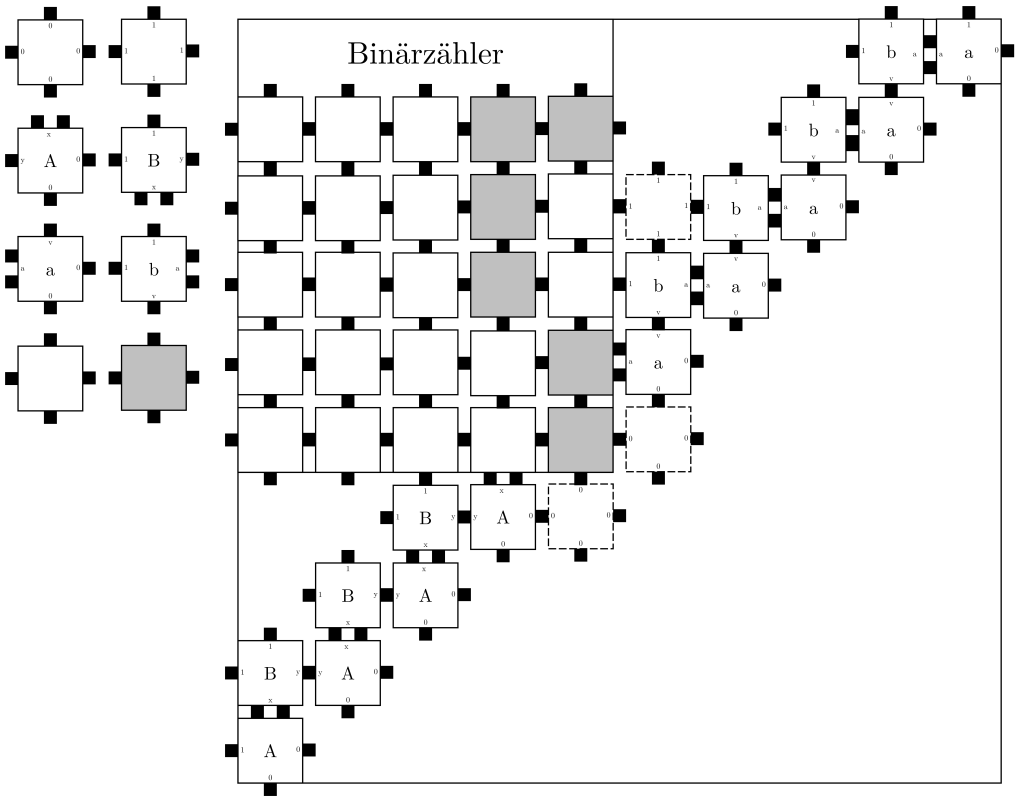


ABBILDUNG 5.10: Quadrat aus logarithmisch vielen Tiletypen auf Basis eines eingebetteten Binärzählers. Graue Tiles im Binärzähler entsprechen einer logischen „1", weiße Tiles einer „0". Es ist zu erkennen, dass in Doppelreihen hochgezählt wird, damit der Zähler höhenbeschränkt werden kann.

anders angegeben, gilt $k = 1$. Ein Nanobot $\mathcal{N}_R^{i,k}$ verfügt über einen Rezeptor R, welcher mit wenigstens einem Nachrichtenmolekül korrekt bindet.

Ein Tile-basierter Nanosensor oder Nanoroboter muss wenigstens eine Kantenlänge von drei haben, damit ein Hohlraum entstehen kann.

Ein Tile-basierter Nanosensor $\mathcal{N}_{Se}^{12,2}$ beschreibt somit einen Nanosensor der Größe 24, da jedes ursprüngliche Tile durch einen Block der Größe 2 ersetzt wurde, um Fehler zu vermeiden.

Nanosensoren und Nanoroboter unterscheiden sich gemäß der präsentierten Definition in der Sensorkomponente. Ein Nanoroboter kann Nachrichtenmoleküle durch eine einfache Bindungsreaktion erkennen. Bei Nanosensoren wird der Prozess anders realisiert und unterscheidet sich je nach Anwendungsfall. Sowohl bei Nanosensoren als auch bei Nanorobotern kann eine korrekte Bindung am Rezeptor einen Mechanismus auslösen. Je nach Anwendungsfall kann es sich dabei um das öffnen von Nanosensor oder Nanoroboter handelt, um Tiles oder Medikamente freizusetzen.

5.2.2 ZUSAMMENSETZUNG VON ALLGEMEINEN STRUKTUREN

Wie in den vorherigen Abschnitten beschrieben, sind Würfel effektiv nutzbare Strukturen in DNA-Tile-basierten Nanonetzwerken, welche im menschlichen Körper nicht vorkommen. Würfel sind zwar einfache und symmetrische Strukturen, es kann jedoch sein, dass andere Figuren vorteilhafter wären, um beispielsweise Medikamente zu transportieren. Daher wird im Folgenden ein Algorithmus vorgestellt, welcher für beliebige Strukturen ein Tileset erzeugen kann [2]. Das Verfahren geht auf Florian Lau, Kristof Stahl und Stefan Fischer zurück. Große Teile der Niederschrift, die algorithmische Idee und Formalisierung gehen auf Florian Lau zurück.

Um beliebige, dreidimensionale Strukturen erzeugen zu können, wird das Vorgehen aus Quelltext 5.2 verallgemeinert. Der resultierende Algorithmus kann Tilesets für beliebige, verbundene, endliche, dreidimensionale Strukturen erzeugen. Jede Position bekommt einen Tiletypen zugeordnet, der exakt einmal verwendet wird. Das Vorgehen funktioniert immer, benötigt jedoch die maximale Anzahl an verschiedenen Tiletypen.

Um große Strukturen erzeugen zu können, ist es jedoch notwendig auf komplexere Strategien auszuweichen, welche nur für bestimmte Klassen von Strukturen funktionieren. Ein Beispiel ist der in Abschnitt 5.2.1.4 vorgestellte Algorithmus.

Die geringe Zahl benötigter Tiletypen lässt sich bei Würfeln durch die Regelmäßigkeit der Struktur erklären. Weniger symmetrische Strukturen lassen sich vermutlich nicht durch ein kleines Tileset ausdrücken.

Quelltext 5.4 erhält ein dreidimensionales boolesches Array als Eingabe. Dieses repräsentiert eine dreidimensionale Struktur durch eine Menge an Punkten aus dem $\mathbb{Z}^3$. Die Eingabestruktur muss vollständig verbunden sein. Jedes Tile hat folglich wenigstens einen Nachbarn.

Der Algorithmus gibt ein Tileset aus, welches zur eingegebenen Struktur erwächst. Durch Blockersetzungsstrategien lässt sich die Größe der Struktur anpassen, wobei gleichzeitig Fehler reduziert werden.

Abbildungen 5.11 (a) und (b) zeigen ein Beispiel für ein Zwischenergebnis und das Ergebnis von der Anwendung des generalisierten Algorithmus. Eingegeben wurde eine Voxelapproximation einer Kugel als dreidimensionales boolesches Array.

5.2.3 MODELLIERUNG VON NACHRICHTENMOLEKÜLEN

Ein Nachrichtenmolekül für das Problem ADD mit vier Eingabebits wurde bereits in Abschnitt 5.1.1 vorgestellt. In den folgenden Abschnitten werden Nachrichtenmoleküle für weitere, teils kompliziertere Probleme konzipiert. Es wird zudem ein Verfahren vorgestellt, mit dem beliebige boolesche Formeln als Nachrichtenmolekül modelliert werden können. In den folgenden Abbildungen werden jeweils nur die algorithmisch relevanten Teile von Tilesets dargestellt. Framework-Tiles werden der Übersichtlichkeit halber ausgelassen.

```
1   Input: 3D boolesches Array: shape
2   Output: Tileset T
3   x,y,z = 0;
4
5   T = {};
6
7   while(x < n){
8     while(y < m){
9       while(z < o){
10        if(shape[x,y,z]){
11          T = T + createPositionTiletype(x,y,z,1)};
12        }
13        z = z + 1;
14      }
15      y = y + 1;
16    }
17    x = x + 1;
18  }
19  return T;
```

QUELLTEXT 5.4: Quelltext zur Erzeugung von Tiletypen für eine beliebige Struktur in einem Self-Assembly-System. [2]

5.2.3.1 BELIEBIGE ENTSCHEIDUNGSPROBLEME LÖSEN

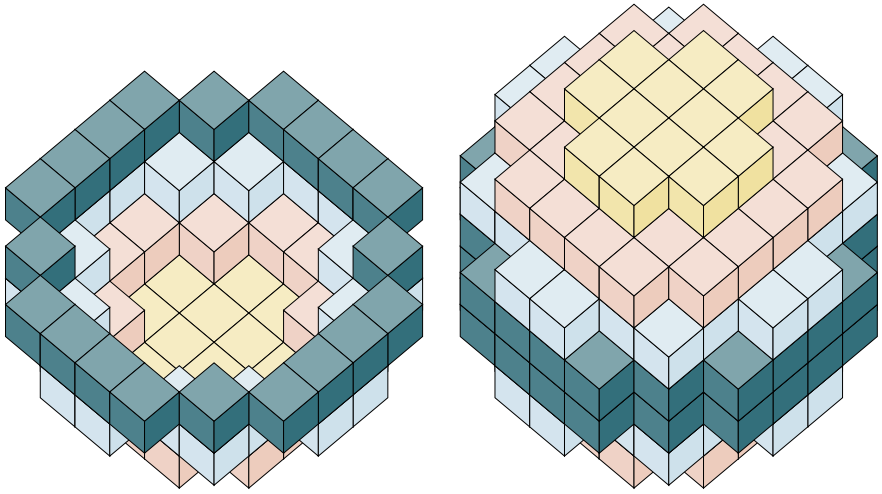
Es ist zwar bekannt, dass Self-Assembly-Systeme ab der Temperatur 2 Turingvollständig sind, dieses Ergebnis kann jedoch nicht direkt auf Nachrichtenmoleküle übertragen werden. In diesem Abschnitt wird ein Verfahren vorgestellt, mit dem sich beliebige mathematische Problemstellungen als Nachrichtenmoleküle ausdrucken lassen. Dies belegt die potenzielle Nützlichkeit von DNA-Tile-basierten Nanonetzwerken.

Aus der theoretischen Informatik ist bekannt, dass sich verschiedene Arten von Problemen ineinander umwandeln lassen. Da Entscheidungsprobleme leicht auf Nanoebene verarbeitet werden können, sind diese von besonderem Interesse.

THEOREM 5.2 – Für jedes als boolesche Formel Φ modellierte Entscheidungsproblem lässt sich ein Nachrichtenmolekül erzeugen, welches dieselbe Berechnung durchführt und bei erfolgreichem Abschluss einen Liganden bildet.

Beweis. Sei Φ eine aussagenlogische Formel. Nach Quine ist bekannt, dass sich jede aussagenlogische Formel Φ in eine disjunktive Normalform überführen lässt [115]. Dies geschieht, indem man die Wahrheitstabelle für besagte Formel aufstellt. Tabelle 5.1 zeigt ein Beispiel. Wahrheitstabellen sind im Allgemeinen exponentiell groß.

Für jede Zeile, welche in der letzten Spalte eine „1" enthält, wird ein Term angelegt. Alle Literale A_n einer Zeile werden mit einem logischen $\wedge$ verknüpft und geklammert – auch *Klausel* genannt. *Literale* sind boolesche Variablen und ihre Inversen. Alle so entstandenen Klauseln werden mit dem logischen $\vee$ verknüpft. Das Ergebnis ist eine Formel Φ', welche eine kanonische Form hat



(a) (b)

ABBILDUNG 5.11: Simulation einer hohlen Kugel im aTAM mit Radius vier – erzeugt durch Quelltext 5.4 im 3D-aTAM-Simulator [11]. Jede Position wird durch einen eindeutigen Tiletypen belegt. Es sind zwei unterschiedliche Wachstumsstadien aufgezeigt. [2]

ϕ_1	ϕ_2	$\dots$	Φ
1	1		1
1	0		1
		$\vdots$	
0	1		0
0	0		1

TABELLE 5.1: Beispiel für eine Wahrheitstabelle für die Formel Φ.

und dasselbe Ergebnis wie die Ursprungsformel liefert. Eine solche Formel wird auch *Distributive Normalform* (DNF) einer Formel Φ genannt. Diese lässt sich weiter mittels Verfahren wie McCluskey minimieren [101].

Für eine Formel $\Phi^{'}$ in der DNF kann folgendermaßen ein Tileset mit identischer Semantik erzeugt werden:

Für jede Klausel wird ein Nachrichtenmolekül angelegt. Die Länge des Nachrichtenmoleküls entspricht der Anzahl der verschiedenen Literale. Abbildung 5.12 zeigt das allgemeine Vorgehen. Die roten Tiles A_n^i entsprechen den Literalen der Formel $\Phi^{'}$.

Der Wahrheitswert jedes Literals wird sowohl im Marker als auch durch die Kleber widergespiegelt. Erst sobald alle Literale Teil des Nachrichtenmoleküls sind, kann ein Rezeptor angebracht werden.

Da für jede Klausel ein Nachrichtenmolekül angelegt wurde und diese mittels $\vee$ verknüpft sind, genügt es, wenn ein beliebiges Nachrichtenmolekül vollständig

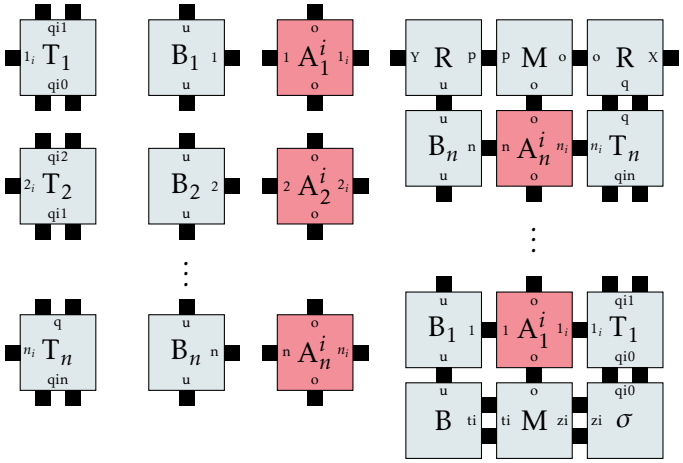


ABBILDUNG 5.12: Allgemeines Vorgehen zur Erzeugung von Tilesets aus Formeln in der DNF am Beispiel der Wahrheitstabelle 5.1.

zusammengesetzt wird, damit eine erfolgreiche Auswertung zu „1" kommuniziert werden kann. □

5.2.3.2 k-BIT-OR

Das logische OR lässt sich am einfachsten als Nanonetzwerk implementieren. Es ist prinzipiell keine Berechnung innerhalb einer Nachricht erforderlich. Ein einzelnes Tile genügt, um eine Information zu einem Nanobot zu transportieren.

Es wird die in Abschnitt 5.1.7 vorgestellte Referenzarchitektur angenommen. Die Nanosensoren beinhalten Tiles, welche unterschiedliche Label, aber identische Kleber haben. Diese Tiles erfüllen sowohl die Rolle eines Informationsträgers als auch die Rolle eines Liganden. Die Kleberstärke entspricht der Temperatur τ. Ein Rezeptor besteht nur aus einem Tile mit der Stärke τ.

Auf die präsentierte Weise genügt es, wenn ein beliebiger Nanosensor ein Ereignis detektiert und Tiles ausschüttet. Dieses Tile kann ungehindert und ohne weitere Bedingungen an den Rezeptor binden. Das Verhalten entspricht einem logischen OR in einem Schaltkreis. Sobald ein beliebiges Tile am Rezeptor bindet, gilt eine „1" als übermittelt. Der Wahrheitswert einer Berechnung entspricht in diesem Fall der Anwesenheit eines Tiles.

DEFINITION 5.5 – Ein Nachrichtenmolekül $\mathcal{M}_{Or}$ ist ein Tileset, welches das Entscheidungsproblem OR berechnet und bei einer erfolgreichen Auswertung einen Liganden bildet.

Durch die Einfachheit des Nachrichtenmoleküls ist die Berechnung bereits durch die Ausschüttung der Input-Tiles abgeschlossen.

5.2.3.3 k-BIT-THRES

Für die Berechnung eines THRES wird ebenfalls die in Abschnitt 5.1.7 vorgestellte Referenzarchitektur verwendet. Ein k-Bit-THRES stellt eine deutlich größere Herausforderung als die bislang vorgestellten Nachrichtenmoleküle für AND und OR dar. Der intuitive Gedanke, dass ein definierter Tile-Stapel der Höhe k im Kommunikationskanal vorhanden ist, scheitert, da ein Nanosensor eine sehr große Menge an Tiles ausschüttet. Self-Assembly-Prozesse werden vom Zufall gesteuert und ein Stapel könnte auf diese Weise bei der Detektion eines Markers durch einen einzelnen Nanosensor zur Höhe k zusammengesetzt werden. Dies entspricht nicht dem erwarteten Verhalten einer THRES-Berechnung über verschiedene Arten von Nanosensoren. Unter Verwendung eines einzelnen Stapels kann entweder dasselbe Tile k mal detektiert werden, was im klassischen Sinne nicht dem Verhalten eines THRES-Gatters entspricht oder es kann eine von vielen möglichen Konstellationen von Eingabebits erkannt werden.

Orientiert man sich an Schaltkreisen, so wird dort ein THRES berechnet, indem alle möglichen Kombinationen von k Eingabewerten, ohne Berücksichtigung der Reihenfolge, gebildet und mittels eines logischen AND verknüpft werden. Alle Ausgänge werden mittels eines logischen OR verbunden.

Dieses Verhalten lässt sich für Self-Assembly-Systeme adaptieren. Abbildung 5.13 zeigt ein Beispiel für die Berechnung eines 3-Bit-THRES über vier Eingabebits. Auch in diesem Fall entspricht die Anwesenheit eines Tiles einem Wahrheitswert von „1". Die unterschiedlich gefärbten Tiles entsprechen den verschiedenen binären Eingaben. Ozeangrüne Tiles bilden das Framework der verschiedenen Nachrichtenmoleküle. Im Tileset links sind lediglich Tiles dargestellt, welche mehr als ein Mal genutzt werden. Es werden insgesamt $\binom{n}{k}$ verschiedene AND-Nachrichtenmoleküle gebildet, die alle denselben Liganden ausbilden. Damit werden alle Möglichkeiten abgedeckt einen Schwellwert der Höhe 3 bei vier verschiedenen Eingaben zu erreichen. Es genügt, wenn ein einziges Nachrichtenmolekül an einem Nanobot bindet, um ein logisches OR zu implementieren. Kombiniert ergibt sich ein Verhalten, welches einem logischen THRES entspricht.

Ein Problem ist, dass die Anzahl der verschiedenen Nachrichtenmoleküle überexponentiell wachsen kann. Bei einer konstanten Anzahl Kleber der Länge k ist die Eingabegröße für THRES dadurch beschränkt.

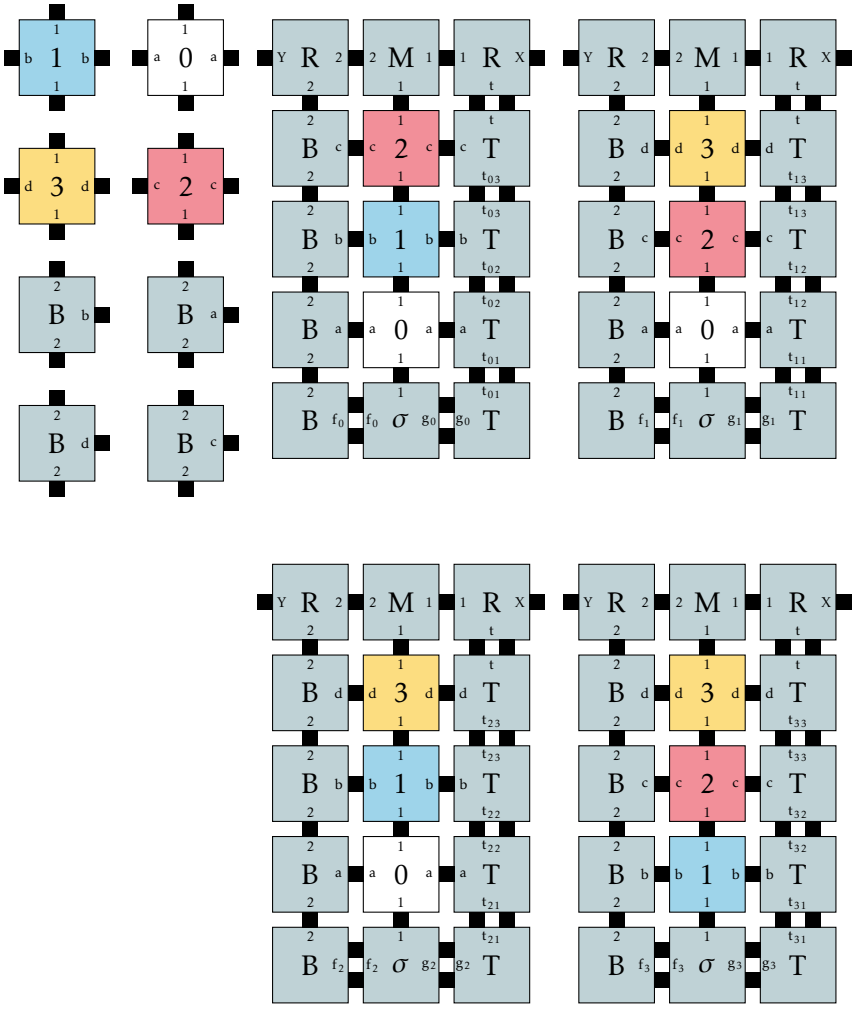


ABBILDUNG 5.13: Tileset (links) und vollständig zusammengesetzte Nachrichtenmoleküle (rechts) für das Entscheidungsproblem THRES. Das Tileset links beinhaltet nur die variablen Komponenten. Rechts sind alle vier Möglichkeiten einen Schwellwert von 3 bei vier möglichen Eingaben zu erreichen dargestellt.

DEFINITION 5.6 – Ein Nachrichtenmolekül $\mathcal{M}_{Thres}$ ist ein Tileset, welches das Entscheidungsproblem THRES berechnet und bei einer erfolgreichen Auswertung zu „1" einen Liganden bildet.

5.2.3.4 k-BIT-ADD

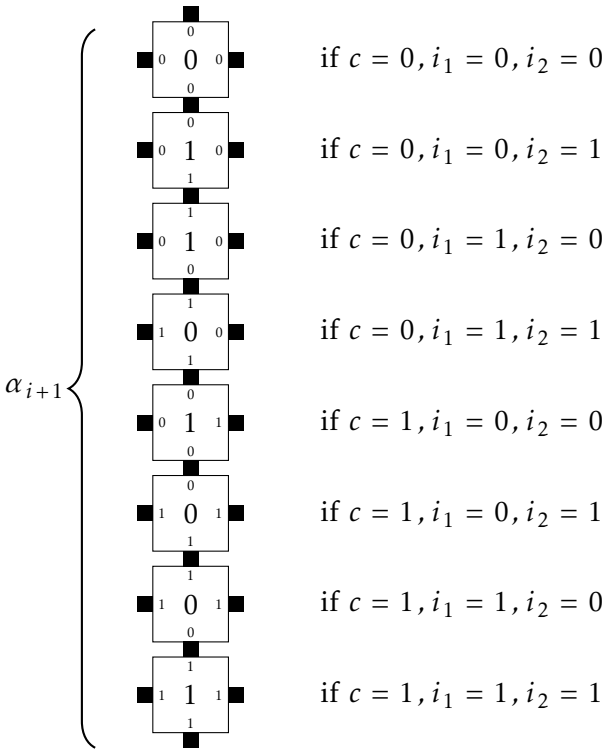
Für das Problem ADD wird die Referenzarchitektur aus Abschnitt 5.1.7 verwendet. Nanosensoren können in diesem Fall ganze Binärzahlen als Assemblies ausschütten, die vom Nanonetzwerk addiert werden sollen. Die Temperatur des Self-Assembly-Systems muss für die präsentierte Architektur wenigstens 3 sein. Es lässt sich ein Tile-basierter Addierer konzipieren, der bei der Temperatur 2 funktioniert und Bits reihenweise verrechnet [45]. Temperatur 2 Addierer sind deutlich komplizierter und der Einfachheit halber ist das Nachrichtenmolekül

für die Temperatur 3 konzipiert.

Die Wahrheitswerte werden so angepasst, dass sie den Labels/Klebern der Tiles entsprechen. Das ist notwendig, um zeitgleich Bits mit den Werten „0" und „1" korrekt darstellen zu können.

Abbildung 5.14 zeigt ein Beispiel für ein zusammengesetztes Nachrichtenmolekül [45]. Weiße Input-Tiles beschreiben den Additionsalgorithmus. Diese sind dauerhaft im Kommunikationskanal vorhanden. Ozeangrüne Tiles bilden Liganden und Framework. Rote, sowie grüne Tiles stellen die Eingabebits dar. Es kann notwendig sein, eine zusätzliche „0" anzufügen, weil das Ergebnis einer Binäraddition potentiell ein Bit länger ist als die Eingabezahlen in Binärrepräsentation. Der rechte Kleber bereits vorhandener Input-Tiles wird als *Carry-Bit* verstanden.

Der Algorithmus entspricht dem Vorgehen bei einer schriftlichen Addition von Binärzahlen. Die Tiles des Algorithmus sind so konzipiert, dass sie sich beim Bit-weisen Vergleich folgendermaßen organisieren:



Erst sobald sich das Ergebnis in der Mitte vollständig gebildet hat, kann es zur Bildung eines Liganden kommen. Da die Wahrheitswerte bei Funktionsproblemen den Labels/Klebern der Tiles entsprechen, kann das Ergebnis nicht direkt von einem Nanobot verwertet werden. Es sind zusätzliche Schritte zur Weiterverarbeitung des Ergebnisses notwendig.

Eine Möglichkeit besteht darin dreidimensionale Tiles zu verwenden. Auf diese Weise kann auf der Rückseite der Tiles ein weiterer Kleber angebracht werden,

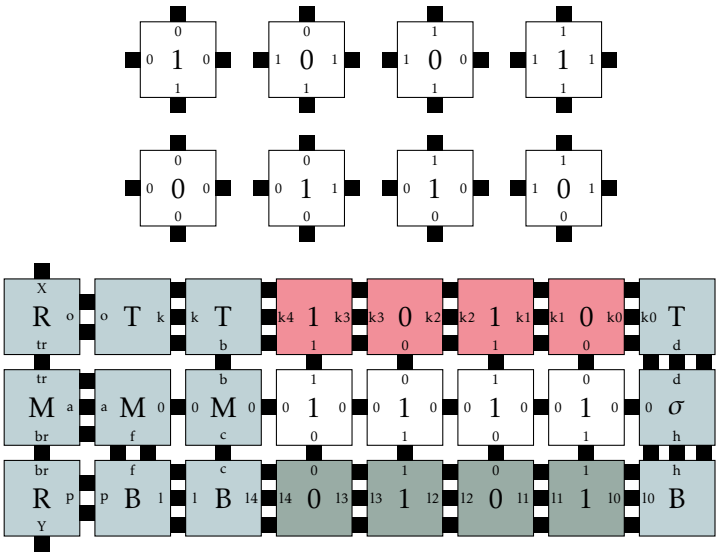


ABBILDUNG 5.14: Tileset und teilweise zusammengesetztes Nachrichtenmolekül für das Funktionsproblem ADD. Die weißen Tiles des Nachrichtenmoleküls (oben) kodieren einen Additionsalgorithmus. Unten ist ein in der Zusammensetzung befindliches Nachrichtenmolekül abgebildet.

welcher dem Wahrheitswert des Tiles entspricht. Die Binärinformation kann so propagiert werden.

DEFINITION 5.7 – Ein Nachrichtenmolekül $\mathcal{M}_{Add}$ ist ein Tileset, welches das Funktionsproblem ADD berechnet und bei einer erfolgreichen Berechnung einen Liganden bildet.

5.2.3.5 k-BIT-MULT

Für das Problem MULT wird ebenfalls die in Abschnitt 5.1.7 Referenzarchitektur verwendet. Nanosensoren können in diesem Fall ganze Binärzahlen in Form von Assemblies ausschütten, die vom Nanonetzwerk multipliziert werden sollen. Die Temperatur des Self-Assembly-Systems muss wenigstens 2 sein. Die Wahrheitswerte werden wie bei ADD angepasst.

Abbildung 5.15 zeigt ein Beispiel [45]. Oben wird das Tileset zur Berechnung dargestellt. Dieses ist dauerhaft im Kommunikationskanal vorhanden. Unten ist ein zusammengesetztes Nachrichtenmolekül mit Ligand „R" zu sehen. Dieser kann sich erst bilden, sobald das letzte Tile der Multiplikation zugegen ist. Es wird die Multiplikationen $7 \cdot 7$ durchgeführt. Der obere Teil des Markers der obersten Reihe entspricht den einzelnen Bits des Ergebnisses 110001_2. Es müssen verschiedene Tiles „M" und „R" existieren, damit für alle möglichen Ergebnisse ein Ligand gebildet werden kann.

Das Vorgehen entspricht, wie bei der Addition, dem Algorithmus zur schriftlichen Multiplikation von Binärzahlen. Das Vorgehen beschreibt eine iterative Addition. Jedes Bit wird einzeln mit einem passenden Bit der jeweils anderen

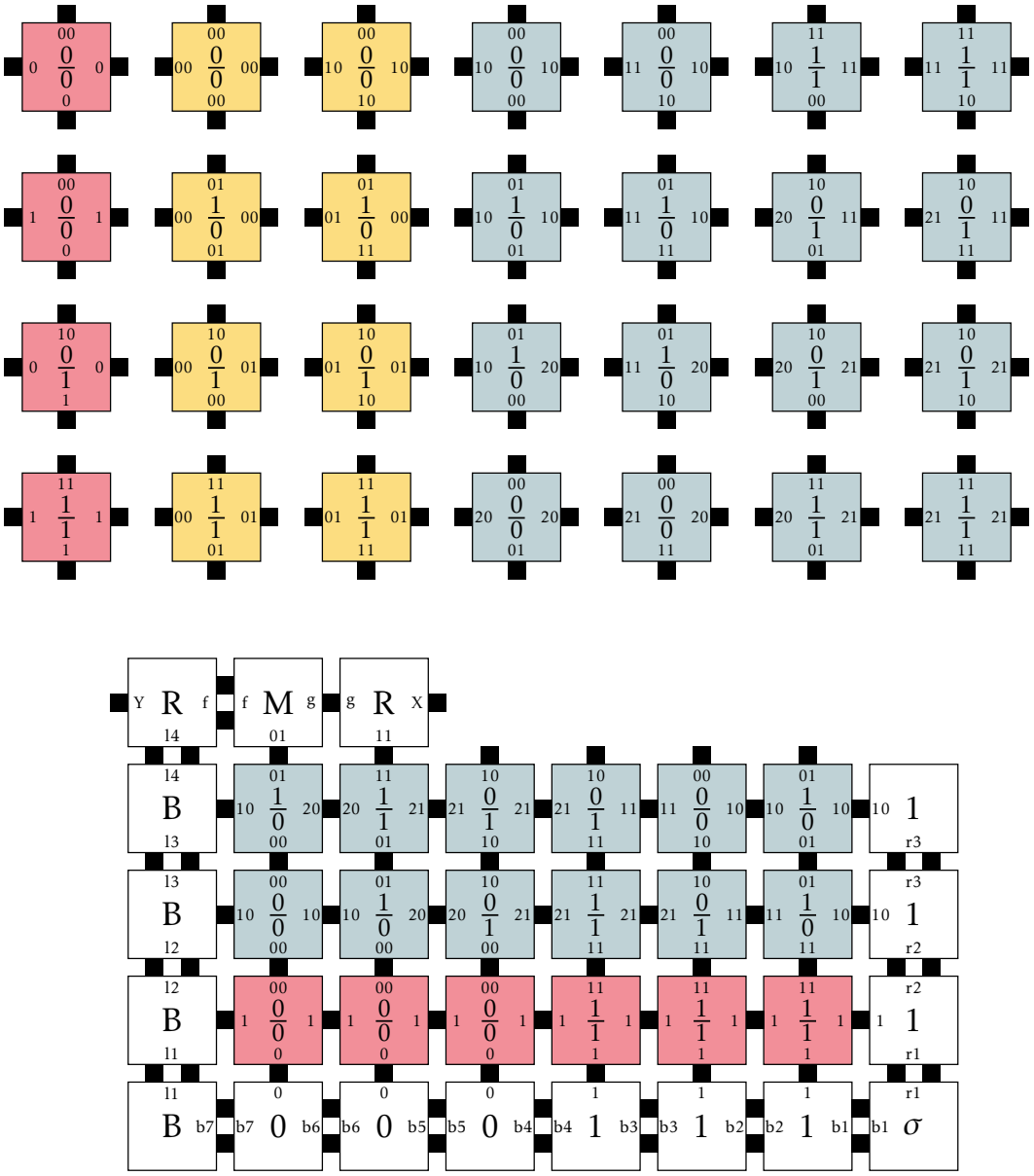


ABBILDUNG 5.15: Tileset und vollständig zusammengesetztes Nachrichtenmolekül für das Funktionsproblem MULT mit Rezeptor. Oben ist das Tileset des Algorithmus dargestellt, unten ein in der Zusammensetzung befindliches Nachrichtenmolekül.

Binärzahl verrechnet. Da das Ergebnis einer Multiplikation doppelt so lang sein kann wie die Eingabe, wird das Framework des Nachrichtenmoleküls in X-Richtung mit „0" entsprechend der Eingabelänge aufgefüllt.

DEFINITION 5.8 – Ein Nachrichtenmolekül $\mathcal{M}_{Mult}$ ist ein Tileset, welches das Funktionsproblem MULT berechnet und daraufhin einen Liganden bildet.

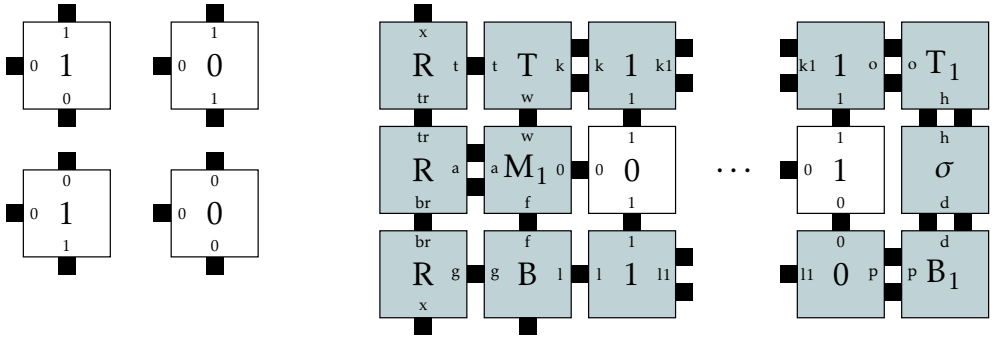


ABBILDUNG 5.16: Tileset (links) und vollständig zusammengesetztes Nachrichtenmolekül (rechts) für das Funktionsproblem XOR mit Ligand. Die weißen Input-Tiles besitzen rechts keinen Kleber, der Kleber links dient lediglich der Bindung eines Liganden.

5.2.3.6 k-BIT-XOR

Dem XOR-Problem liegt die in Abschnitt 5.1.7 vorgestellte Referenzarchitektur zugrunde. Nanosensoren schütten einzelne Bits oder Binärzahlen in Form von Assemblies aus, welche im Nachrichtenmolekül entsprechend einer XOR-Operation verrechnet werden. Der Wahrheitswert eines Tiles wird durch das Label/Kleber ausgedrückt, da nicht zeitgleich „0" und „1" durch Anwesenheit kodiert werden kann.

Abbildung 5.16 zeigt ein Nachrichtenmolekül samt Tileset zur Berechnung eines exklusiven OR. In Weiß ist der Algorithmus dargestellt, welcher immer im Kommunikationskanal vorhanden sein muss. Der Ligand und Eingabe sind ozeangrün dargestellt, die Eingabestruktur ist mit binären Markern versehen. Das Vorgehen funktioniert analog zu einem Bit-weisen ADD ohne Berücksichtigung eines Carry-Bits. Demnach verfügen die Input-Tiles auf der rechte Seite über *keinen* Kleber.

DEFINITION 5.9 – Ein Nachrichtenmolekül $\mathcal{M}_{Xor}$ ist ein Tileset, welches das Entscheidungsproblem XOR berechnet und bei einer erfolgreichen Auswertung zu „1" einen Liganden bildet.

5.2.3.7 k-BIT-INC

Das Zählproblem INC kann als Entscheidungsproblem modelliert werden, solange die Eingabe unär kodiert ist. Es wird dieselbe Referenzarchitektur wie in den übrigen Problemen verwendet. Abbildung 5.17 zeigt die Struktur des Nachrichtenmoleküls. Ozeangrüne Tiles„R" bilden den Rezeptor, ozeangrüne Tiles„F" das Framework des Moleküls. Weiße Tiles sind Teil der Eingabe. Da es sich um ein Zählproblem handelt, sind alle weißen Tiles identisch. Die Höhe des Frameworks kodiert dabei unär eine Zahl k. Erst wenn bis zur Frameworkhöhe k gezählt wurde, bildet sich ein Ligand.

DEFINITION 5.10 – Ein Nachrichtenmolekül $\mathcal{M}_{Inc}$ ist ein Tileset, welches das

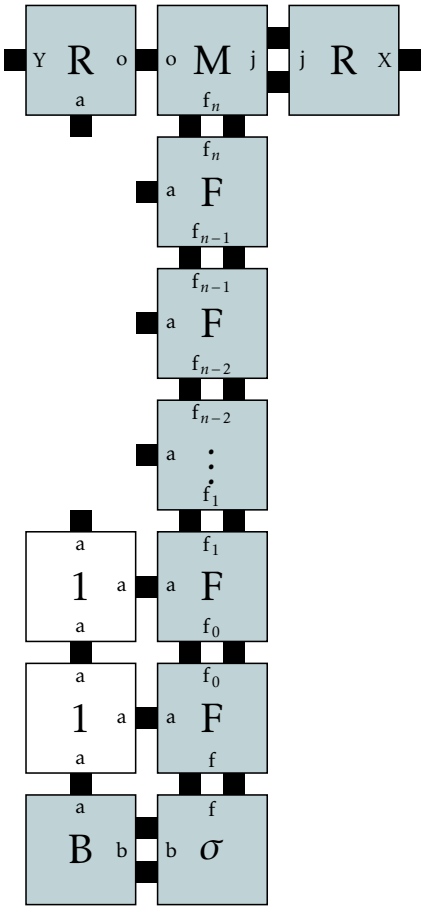


ABBILDUNG 5.17: Zusammensetzung eines Nachrichtenmoleküls für das Entscheidungsproblem INC mit Rezeptor.

Entscheidungsproblem INC berechnet und bei einer erfolgreichen Auswertung zu „1" einen Liganden bildet.

5.3 MODELLIERUNG VON NANONETZWERKEN FÜR PROBLEME

In den folgenden Abschnitten werden drei ausgewählte Szenarien als DNA-Tile-basierte Nanonetzwerke modelliert. Diese decken einen großen Anteil der durch die Literatur implizit gestellten Anforderungen an Nanonetzwerke ab. Zuerst wird ein verteilter Konsens durch AND implementiert, im Anschluss daran eine Ereigniserkennung anhand einer Schwellwertüberschreitung THRES und zum Schluss das Addieren ADD von Zahlen durch ein Nanonetzwerk.

5.3.1 VERTEILTE EINIGUNG ÜBER KRANKHEITSERKENNUNG

Abbildung 5.1 zeigt das verwendete Tileset für ein 4 Bit-AND. Abbildung 5.18 zeigt eine mögliche Implementierung eines 4 Bit-AND durch eine Nanonetzwerk.

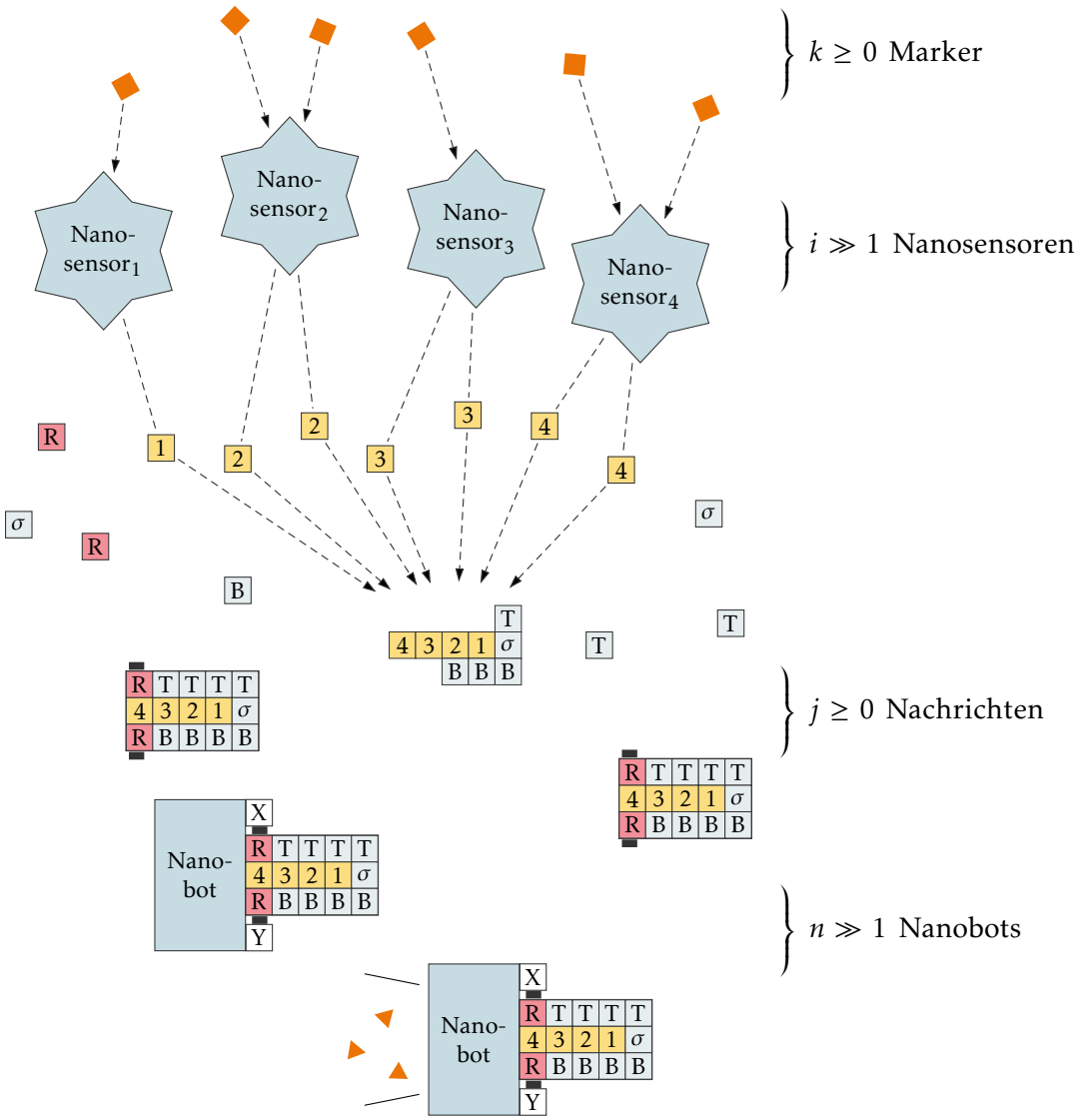


ABBILDUNG 5.18: Modellierung des Problems AND als DNA-Tile-basiertes Nanonetzwerk. Es wird ein verteilter Konsens über die Messungen von vier verschiedenen Sorten von Nanosensoren gebildet und kommuniziert.

Dabei wird die übliche Referenzarchitektur angenommen. Nanosensor 1–4 repräsentieren dabei vier sehr große Mengen an Nanosensoren, welche bei der Detektion eines Markers Tiles ausschütten. Diese setzen sich im Anschluss zu Nachrichtenmolekülen zusammen, wobei ein verteilter Konsens berechnet wird.

Sobald das Nachrichtenmolekül vollständig zusammengesetzt ist, gilt das Problem als berechnet und das Ergebnis kann von einem Nanobot in Empfang genommen werden.

Ein AND-Nanonetzwerk wird analog zu Nachrichtenmolekülen und Nanostrukturen als Tileset definiert. Für die Nanosensoren und Nanobots werden Tilesets

für hohle Würfel angenommen.

DEFINITION 5.11 – Ein Tile-basiertes *AND*-Nanonetzwerk $\mathcal{N}_{And}$ ist ein Tileset $\mathcal{N}_{Se}^{i,k} \cup \mathcal{N}_{R}^{i,k} \cup \mathcal{M}_{And}$. $\mathcal{N}_{Se}^{i,k}$ ist ein Tileset für Nanosensoren, $\mathcal{N}_{R}^{i,k}$ ist ein Tileset für Nanobots und $\mathcal{M}_{And}$ ein Tileset für Nachrichtenmoleküle, welche die Funktion AND berechnen. k ist ein Faktor für eine Blockersetzungsstrategie und i die Kantenlänge der Würfel.

5.3.2 KRANKHEITSSIGNIFIKANZ DURCH SCHWELLWERT-ÜBERSCHREITUNG

Abbildung 5.13 zeigt das verwendete Tileset für einen Schwellwert der Größe 3. Das dazugehörige Nanonetzwerk ist in Abbildung 5.19 dargestellt. Es wird die übliche Referenzarchitektur angenommen.

Es gilt die Besonderheit, dass Nachrichtenmoleküle verschiedene Formen annehmen können. Für jede mögliche Untermenge von drei Tiles sind spezifische Framework-Tiles im Medium vorhanden. In diesem Fall existieren vier verschiedene Tiles σ, T und B. Das übrige Szenario funktioniert analog zum 4 Bit-AND.

Ein THRES-Nanonetzwerk wird analog zu Nachrichtenmolekülen und Nanostrukturen als Tileset definiert. Nanobots und Nanosensoren werden durch hohle Würfel modelliert, welche sich entweder durch Bindung eines passenden Nachrichtenmoleküls oder Markers öffnen lassen.

DEFINITION 5.12 – Ein Tile-basiertes *THRES*-Nanonetzwerk $\mathcal{N}_{Thres}$ ist ein Tileset $\mathcal{N}_{Se}^{i,k} \cup \mathcal{N}_{R}^{i,k} \cup \mathcal{M}_{Thres}$. $\mathcal{N}_{Se}^{i,k}$ ist ein Tileset für Nanosensoren. $\mathcal{N}_{R}^{i,k}$ ist ein Tileset für Nanobots und $\mathcal{M}_{Thres}$ ein Tileset für Nachrichtenmoleküle, welche die Funktion THRES berechnen. k ist der Faktor für eine Blockersetzungsstrategie und i die Kantenlänge der Würfel.

5.3.3 ADDITION VON ZAHLEN BEI EREIGNISSEN

Abbildung 5.14 zeigt das verwendete Tileset für das ADD-Nanonetzwerk aus Abbildung 5.20. Nanosensoren schütten in diesem Szenario vollständige Binärzahlen als Assemblies oder deren Bestandteile in gleichen Mengen aus. Diese bilden anschließend einen Rahmen aus Framework-Tiles, in dem die Addition stattfinden kann.

Erst sobald das letzte Tile Teil des Nachrichtenmoleküls ist, kann sich ein Ligand bilden und das Ergebnis kommuniziert werden. Da es sich um ein Funktionsproblem handelt, kann das Ergebnis nicht direkt an einen Nanobot übermittelt werden. Es wird lediglich die Information einer erfolgreich durchgeführten Berechnung verbreitet. Im Falle dreidimensionaler Tiles kann die Rückseite der Input-Tiles genutzt werden, um weitere Kleber zu verwenden, an denen zusätzliche Bindungsprozesse oder Verarbeitungen stattfinden können.

Ein ADD-Nanonetzwerk wird analog zu Nachrichtenmolekülen und Nanostrukturen als Tileset definiert. Nanobots und Nanosensoren werden als hohle Würfel modelliert.

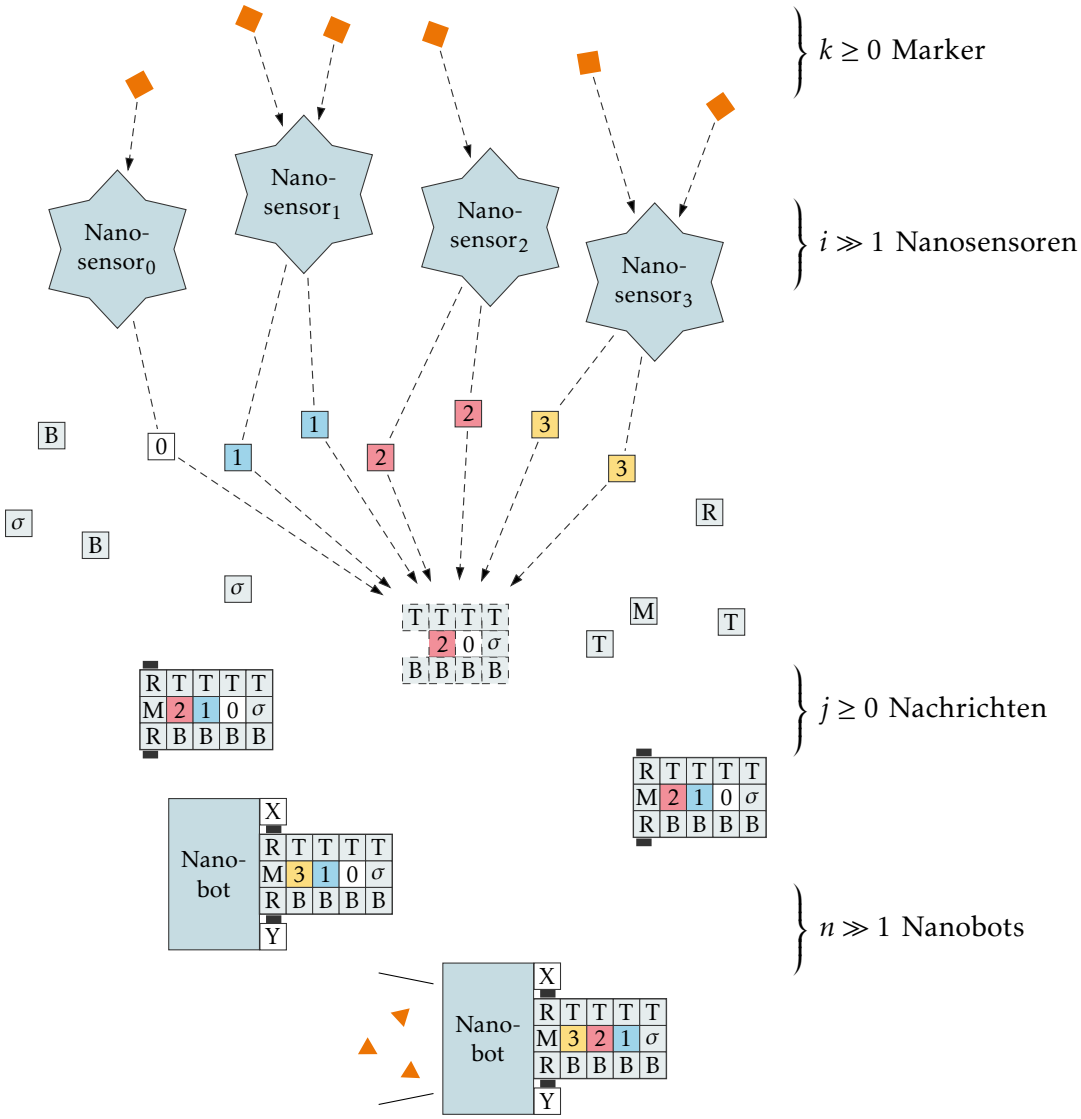


ABBILDUNG 5.19: Modellierung des Problems THRES als DNA-Tile-basiertes Nanonetzwerk. Im Gegensatz zum AND-Nanonetzwerk existieren verschiedene Nachrichtenmoleküle, von denen ein beliebiges ein positives Messergebnis kommuniziert.

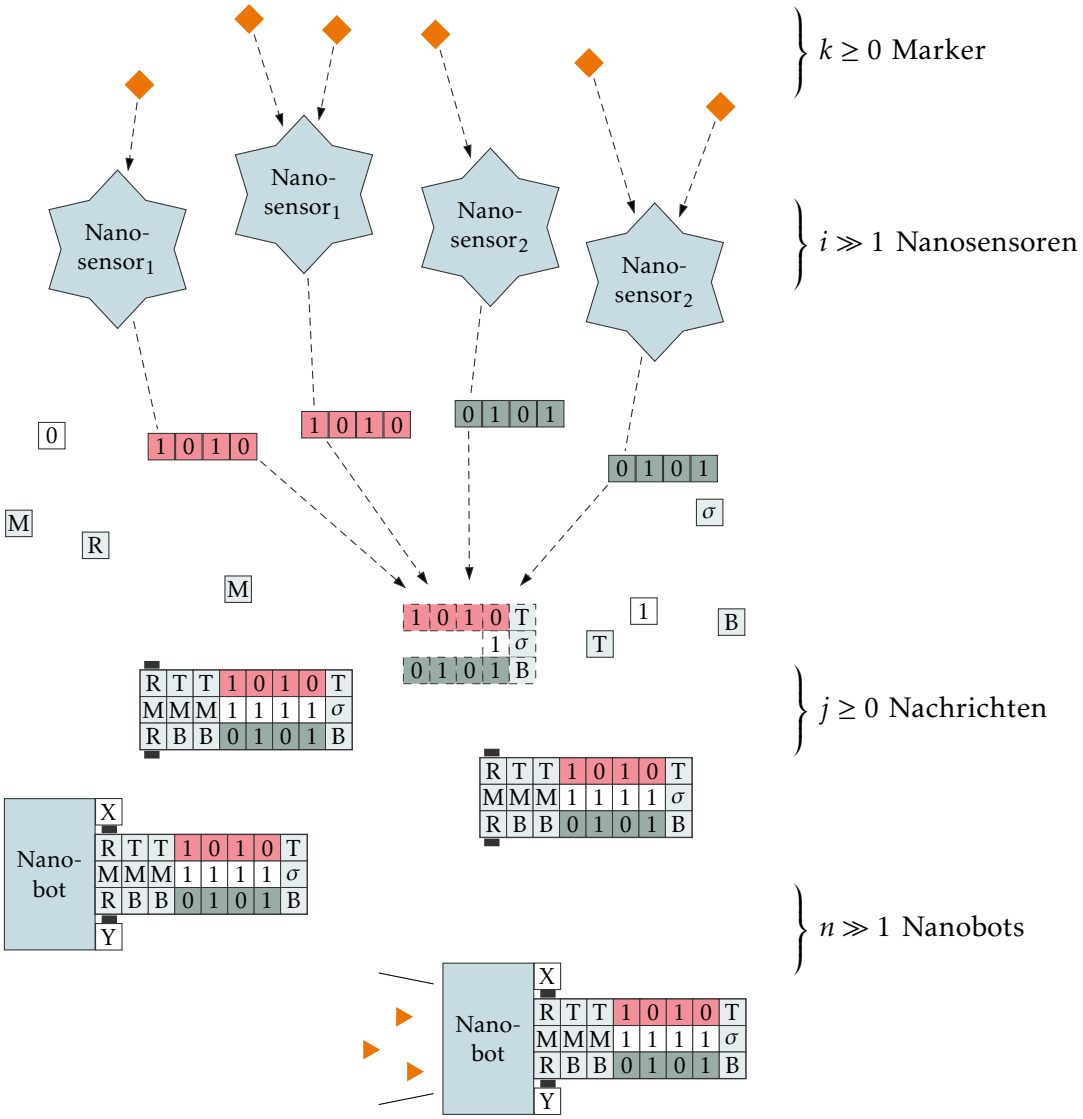


ABBILDUNG 5.20: Modellierung des Problems ADD als DNA-Tile-basiertes Nanonetzwerk. Es wird die Information einer erfolgreichen Addition kommuniziert. Die Nanosensoren schütten ganze Binärzahlen oder deren Bestandteile aus.

DEFINITION 5.13 – Ein Tile-basiertes *ADD*-Nanonetzwerk $\mathcal{N}_{Add}$ ist ein Tileset $\mathcal{N}_{Se}^{i,k} \cup \mathcal{N}_{R}^{i,k} \cup \mathcal{M}_{Add}$. $\mathcal{N}_{Se}^{i,k}$ ist ein Tileset für Nanosensoren. $\mathcal{N}_{R}^{i,k}$ ist ein Tileset für Nanobots und $\mathcal{M}_{Add}$ ein Tileset für Nachrichtenmoleküle, welche die Funktion ADD berechnen. k ist der Faktor für eine Blockersetzungsstrategie und i die Kantenlänge der Würfel.

KAPITEL 6

EVALUATION

In diesem Kapitel[1] werden die in Kapitel 5 vorgestellten Szenarien mittels Simulationen evaluiert. Hierfür wird die ISU TAS-Software von Patitz [111] verwendet. Die in Kapitel 5 gezeigten Tilesets werden in den entsprechenden Modulen für kTAM und 2HAM instanziiert.

In Abschnitt 6.1 wird das kTAM-Modul zur realitätsnahen Simulation von Assemblyprozessen verwendet. Das kTAM-Modul ist jedoch auf die Simulation von genau einem Assembly limitiert. Es sagt Wet-Lab-Experimente sinnvoll voraus, wie in [118] gezeigt wurde. In Abschnitt 6.2 werden die Tilesets mittels des 2HAM-Moduls der ISU TAS-Software untersucht. Das 2HAM wird genutzt, um die Interaktion zwischen Rezeptor und Nachrichtenmolekül zu bestätigen und um die Wechselwirkung der Zwischenprodukte eines Assemblyprozesses zu erforschen. Sobald Nachrichtenmoleküle vollständig zusammengesetzt sind, verhalten sich diese wie herkömmliche Moleküle bei der Kommunikation. Zuletzt wird in Abschnitt 6.3 auf Basis von bereits durchgeführten Wet-Lab-Experimenten extrapoliert, um Aufschlüsse über das zu erwartende Verhalten der Tilesets zu gewinnen.

6.1 REALISTISCHE SIMULATION IM KTAM

Für die kTAM-Simulation der Komponenten bei Temperatur 2 wurden die Parameter $G_{se} = 10,4$ und $G_{mc} = 17,0$ gewählt. Diese entsprechen den Werten aus [118]. Der Wert G_{se} beschreibt die nötigen Kosten, um eine Bindung aufzubrechen und G_{mc} modelliert die Monomerkonzentration. Das Verhältnis zwischen den beiden Werten $\frac{G_{mc}}{G_{se}}$ entspricht dem Temperaturparameter τ im aTAM.

Der Wert von $G_{se} = 10,4$ hat zu zahlreichen Fehlern bei der Simulation in [118] geführt. Übertragen auf die Simulation können so "schlechte" Self-Assembly Bedingungen simuliert werden. Ein G_{mc}-Wert von 17,0 korrespondiert mit einer Konzentration von 0,8 µM pro Tile. $G_{se} = 10,4$ korrespondiert mit einer

[1] Teile dieses Kapitels wurden bereits in [1] vorgestellt.

Umgebungstemperatur von 32,7 °C. Niedrigere Werte entsprechen höheren Temperaturen, was stabile Bindungen unwahrscheinlicher macht. Ein Wert von $G_{se} = 8,5$ korrespondiert z. B. mit einer Temperatur von 41,8 °C.

In der Simulation wird weiterhin angenommen, dass jeder Tiletyp gleich häufig und gleichverteilt im Medium vorkommt. In visionären Szenarien im menschlichen Körper können diese Einschränkungen nicht vorausgesetzt werden.

6.1.1 NACHRICHTENMOLEKÜLE

Die Nachrichtenmoleküle und die in ihnen kodierten Bedingungen bilden den Kern des intelligenten Verhaltens von Nanonetzwerken. Die Nachrichtenmoleküle der drei genauer betrachteten Szenarien werden in den folgenden Abschnitten im ISU TAS-Simulator untersucht.

Die Syntax der Ausgaben des Simulators weichen dabei von der in Kapitel 2 vorgestellten Syntax ab und ist weniger leserlich. Die Farbe eines Tiles deutet das semantische Verhalten an. Gelbe Tiles sind Teil eines Algorithmus. Da diese von Nanosensoren ausgeschüttet werde, werden sie auch *Eingabe* genannt. Rote Tiles bilden den Liganden. Ozeangrüne Tiles sind Framework-Tiles. Weiße Tiles sind Teil des Seed-Assembly und blaue Tiles stellen den Rezeptor dar.

Das fett gedruckte Label in der Mitte entspricht dem Marker, die Labels in den vier Himmelsrichtungen den Farben der Kleber. Eine gestrichelte Quadratseite entspricht keinem Kleber, eine durchgezogene Linie einem Kleber, zwei durchgezogene Linien einem Kleber der Stärke 2 und ein sehr dicker Quadratrand einem Kleber der Stärke 3. Die weniger prominenten Labels in der Mitte sind eindeutige Identifikatoren und dienen lediglich der Unterscheidbarkeit.

6.1.1.1 4 BIT-AND IM KTAM

In diesem Abschnitt wird der Self-Assembly-Prozess von Nachrichtenmolekülen, die ein 4 Bit-AND berechnen, im kTAM mittels Simulation evaluiert. Die Simulation hat gezeigt, dass ein Nachrichtenmolekül aus 15 Tiles im Erwartungswert nach ca. 3000 Bindungsreaktionen vollständig zusammengesetzt ist. Dies schließt sowohl die erfolgreichen Bindungen als auch jene Bindungen, die sich wieder abgelöst haben mit ein. Weiterhin werden Kleber der Stärke 0 in der Simulation berücksichtigt, was die Gesamtanzahl an benötigten Reaktionen weiter erhöht und realistischen Bedingungen entspricht. Abbildung 6.1 zeigt das verwendete Tileset. Abbildung 6.11 zeigt das Ergebnis der Simulation.

Abbildung 6.2 fasst das Ergebnis von 100 Simulationsdurchläufen in einem Histogramm zusammen. Die Anzahl der benötigten Bindungsreaktionen erscheint Log-normalverteilt. Es ist eine Verzerrung in Richtung höherer Werte erkennbar. Kein Nachrichtenmolekül wurde in unter 1500 Reaktionen fertiggestellt.

In der Realität finden unzählige Bindungsreaktionen nahezu zeitgleich statt. Generell korreliert die Anzahl der benötigten Bindungsreaktionen mit der benötigten Zeit, um ein Assembly zu vervollständigen.

ABBILDUNG 6.1: Tileset für ein 4 Bit-Nachrichtenmolekül $\mathcal{M}_{And}$ im kTAM mit Rezeptor. Notwendige Tiles für den Liganden sind rot dargestellt, der Algorithmus ist gelb und Framework-Tiles sind ozeangrün. Die hellblauen Tiles setzen sich zu einem Rezeptor zusammen.

Solange $2G_{se} > G_{mc} > G_{se}$ für Systeme mit der Temperatur 2 gilt, ist das Hinzufügen von Tiles zum Assembly wahrscheinlicher als das Ablösen und ein Nachrichtenmolekül wird in endlicher Zeit fertiggestellt [163].

Kein Simulationsdurchlauf hat eine fehlerhafte Interaktion zwischen den Tiles des Rezeptors und den Liganden der Nachrichtenmoleküle ergeben. Selbst nach 100 000 Bindungsreaktionen sind keine Fehler in den betrachteten Nachrichtenmolekülen zu erkennen und sie blieben stabil. Die Fehlerwahrscheinlichkeit für die Zusammensetzung eines einzelnen Nachrichtenmoleküls betrug 0 %.

6.1.1.2 3 BIT-THRES IM KTAM

Für die kTAM-Simulation bei Temperatur 2 wurden die Parameter $G_{se} = 10,40$ und $G_{mc} = 17,0$ gewählt. Die Simulation hat gezeigt, dass ein Nachrichtenmolekül der Größe 15 im Erwartungswert nach ca. 20 000 Bindungsreaktionen

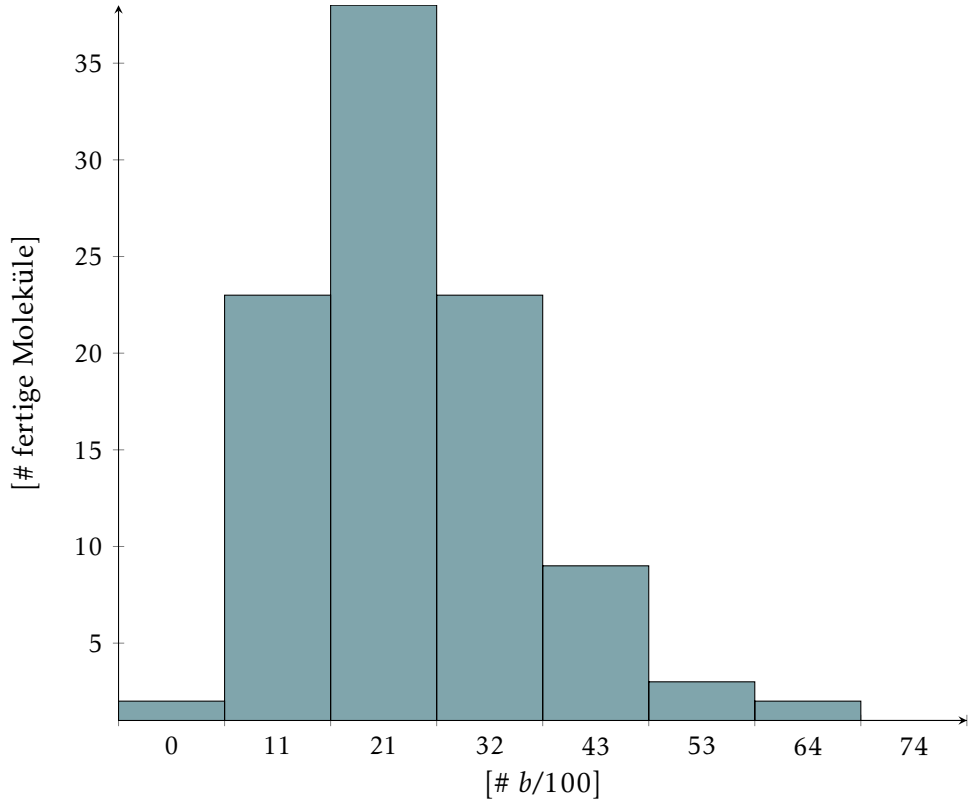


ABBILDUNG 6.2: Ergebnis von 100 kTAM-Simulationen des Nachrichtenmoleküls $\mathcal{M}_{And}$. Das Histogramm zeigt die Anzahl der vollständigen Moleküle in Bezug auf die benötigten Bindungsreaktionen b.

vollständig zusammengesetzt ist. Dies schließt sowohl das Hinzufügen als auch das Ablösen von Tiles mit ein. Die hohe Zahl an benötigten Bindungsreaktionen lässt sich durch die zeitgleiche Simulation von vier Assemblies erklären.

Abbildung 6.3 zeigt das verwendete Tileset. Abbildung 6.5 zeigt das Ergebnis der Simulation. Die Simulationen werden abgebrochen, sobald ein Nachrichtenmolekül vollständig zusammengesetzt wurde, da ein THRES ab diesem Zeitpunkt zu „1" auswertet.

Abbildung 6.4 fasst das Ergebnis von 50 Simulationsdurchläufen in einem Histogramm zusammen. Die Anzahl der benötigten Bindungsreaktionen erscheint ebenfalls Log-normalverteilt. Es ist eine Verzerrung in Richtung höherer Zahlen erkennbar, da eine vollständige Zusammensetzung in wenigen Schritten kaum möglich ist. Kein Nachrichtenmolekül wurde in unter 6500 Reaktionen fertiggestellt. In der Realität finden unzählige Bindungsreaktionen nahezu zeitgleich statt. Die Anzahl der benötigten Bindungsreaktionen korreliert mit der benötigten Zeit, um ein Assembly zu vervollständigen.

Die Simulationsdurchläufe haben gezeigt, dass die präsentierte Architektur wenig fehleranfällig ist – sieben von 100 Assemblies wiesen bleibende Growth-Errors auf, welche sich allerdings nicht ausbreiten konnten. Die Wahrscheinlichkeit für eine fehlerhafte Zusammensetzung eines Nachrichtenmoleküls beträgt

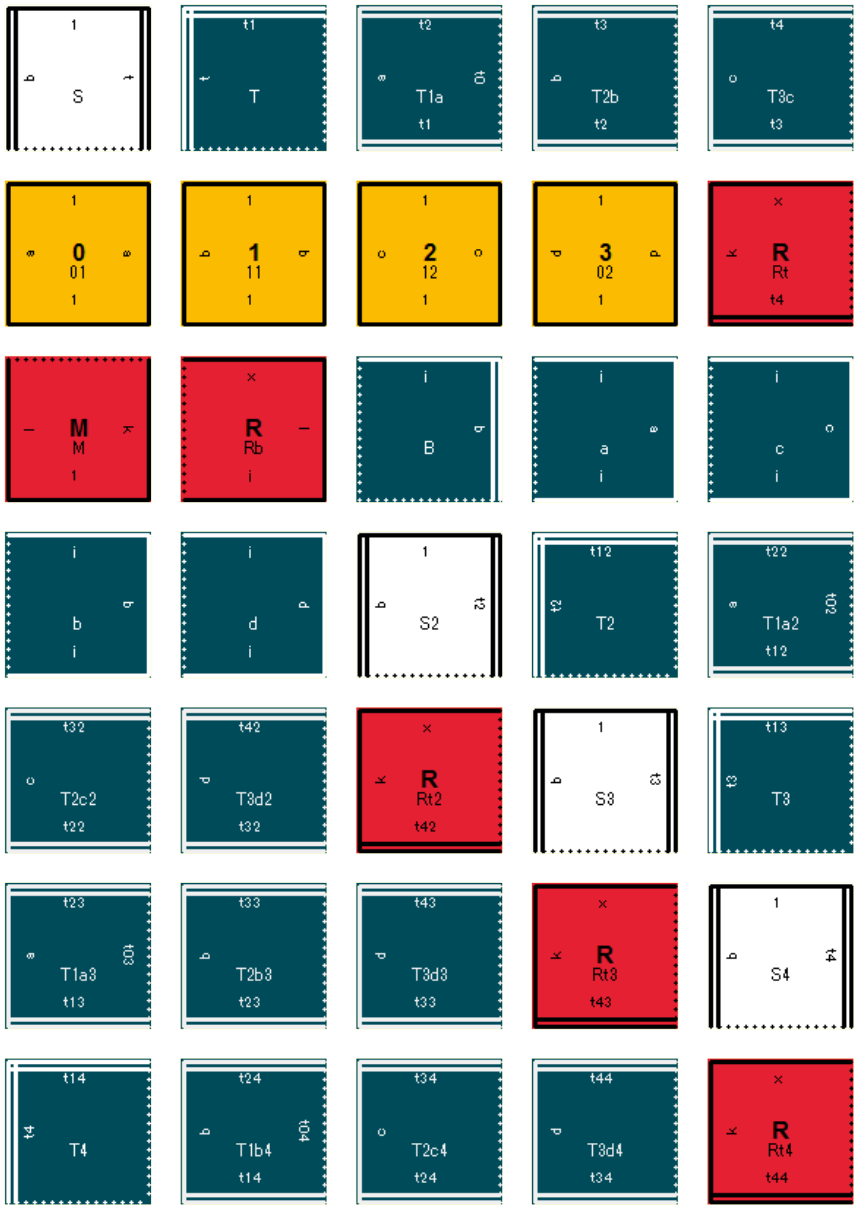


ABBILDUNG 6.3: Tileset für ein 3 Bit-Nachrichtenmolekül $\mathcal{M}_{Thres}$ im kTAM. Rote Tiles sind Teil des Liganden. Weiße Tiles sind Seed-Tiles. Ozeangrüne Tiles sind Framework-Tiles und gelbe Tiles beschreiben den Algorithmus.

folglich 3,5 %. Die einzelnen Reihen setzen sich nacheinander zusammen, sodass stets nur ca. zwei Teile der Wachstumsfront benachbart sein können. Die Simulationszeit wächst multiplikativ mit der Anzahl der Tiletypen und der möglichen Bindungspositionen.

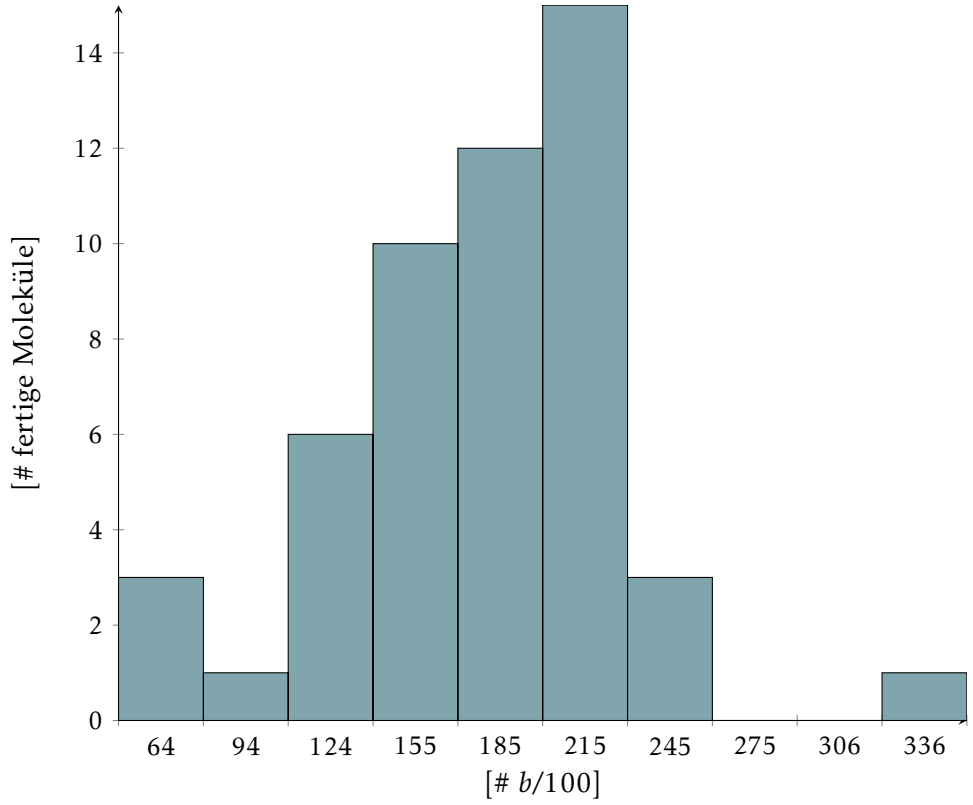


ABBILDUNG 6.4: Ergebnis von 50 kTAM-Simulationen des Nachrichtenmoleküls $\mathcal{M}_{Thres}$. Das Histogramm zeigt die Anzahl der vollständigen Moleküle in Bezug auf die benötigten Bindungsreaktionen b.

6.1.1.3 4 BIT-ADD IM KTAM

Für die kTAM-Simulation bei Temperatur 3 wurden die Parameter $G_{se} = 5,95$ und $G_{mc} = 16,0$ gewählt. Die Simulation hat gezeigt, dass ein Nachrichtenmolekül der Größe 24 im Erwartungswert nach ca. 14 000 Bindungsreaktionen vollständig zusammengesetzt wurde. Dies schließt sowohl das Hinzufügen als auch das Ablösen von Tiles mit ein. Die Wahrscheinlichkeit für eine fehlerhafte Zusammensetzung eines Nachrichtenmoleküls beträgt 14 %. Abbildung 6.7 zeigt das Ergebnis der Simulation. Abbildung 6.6 zeigt das verwendete Tileset.

Abbildung 6.8 fasst das Ergebnis von 50 Simulationsdurchläufen in einem Histogramm zusammen. Die Anzahl der benötigten Bindungsreaktionen erscheint ebenfalls Log-normalverteilt. Es ist eine Verzerrung in Richtung höherer Zahlen erkennbar. Kein Nachrichtenmolekül wurde in unter 8500 Reaktionen fertiggestellt.

Die Simulationsdurchläufe haben gezeigt, dass die präsentierte Architektur fehleranfällig ist. Die obere und untere Reihe setzen sich mit hoher Wahrscheinlichkeit früh zusammen. Bindungen in der Mitte sind somit sehr anfällig für Growth-Errors. Nur wenn die Parameter G_{se} und G_{mc} sehr präzise gewählt werden kann mit hoher Wahrscheinlichkeit eine korrekte Zusammensetzung gewährleistet werden. Dies kann in Wet-Lab-Experimenten problematisch sein.

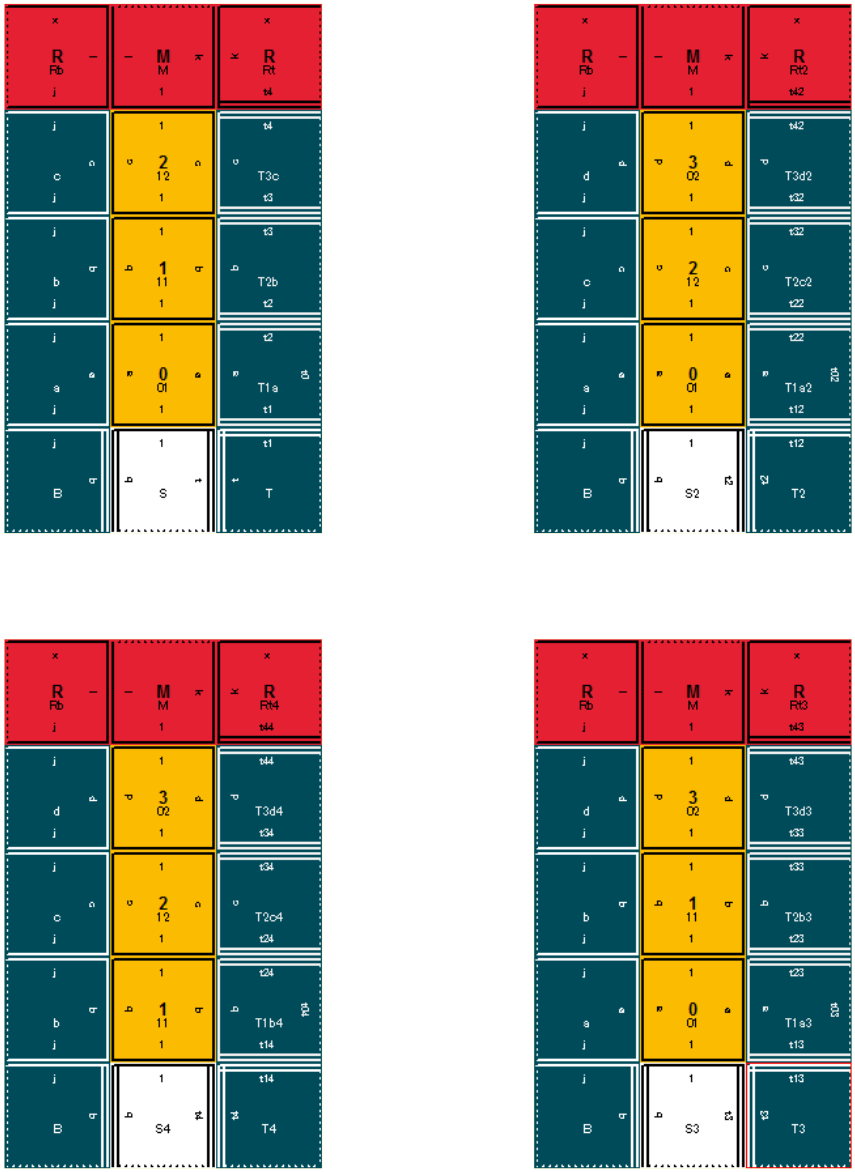


ABBILDUNG 6.5: 3 Bit-Nachrichtenmolekül $\mathcal{M}_{Thres}$ im kTAM. Es wurden vier Nachrichtenmoleküle zeitgleich simuliert. Die Funktion THRES wertet bereits bei der Zusammensetzung eines Nachrichtenmoleküls zu „1" aus. Die Nachrichtenmoleküle unterscheiden sich sowohl im Aufbau der rechten Framework-Tiles als auch in der daraus resultierenden Reihenfolge der gelben Input-Tiles.

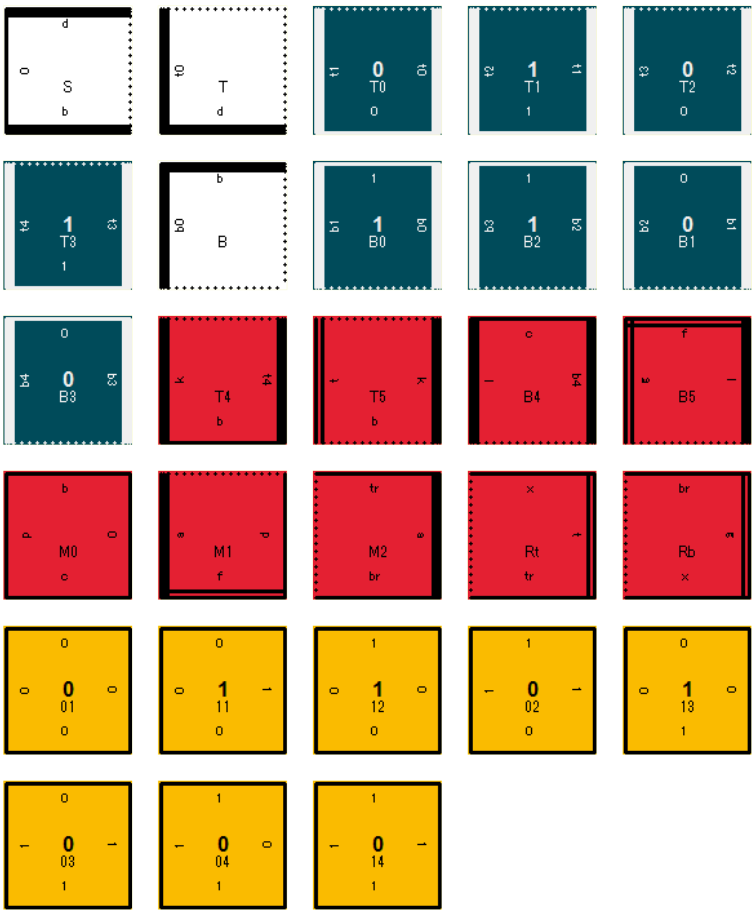


ABBILDUNG 6.6: Tileset für ein 4 Bit-Nachrichtenmolekül $\mathcal{M}_{Add}$ im kTAM. Die roten Tiles bilden den Liganden. Gelbe Tiles beschreiben den Algorithmus und ozeangrüne Tiles beschreiben die Eingaben für den Algorithmus, welche von Nanosensoren ausgeschüttet wird. Die weißen Tiles bilden die Seed-Struktur.

Durch das schmale Intervall an möglichen Werten für G_{se} und G_{mc} wird die Assemblygeschwindigkeit reduziert, da es häufiger zu Tile-Ablösungen kommen muss – so denn Fehlerreduktion ein Ziel ist. Nach Anpassung der Parameter auf einen Wert der im aTAM $\tau = 3$ entspräche, kam es zu sieben dauerhaften Fehlern. Diese Werte gelten jedoch nur für genau ein simuliertes Assembly zur Zeit. Die 2HAM-Simulation zeigt weitere Schwächen der Architektur, wie in späteren Abschnitten gezeigt wird.

6.1.2 NANOSTRUKTUREN UND NANOGERÄTE

Es gibt zwei grundlegende Techniken hohle Würfel herzustellen, welche als Nanogeräte oder Nanosensoren genutzt werden können. DNA-Origami und Self-Assembly-Systeme. Die folgenden Simulationen basieren auf Self-Assembly-Systemen. Dabei können Würfel entweder aus zweidimensionalen Quadraten gefaltet oder direkt aus dreidimensionalen Tiles zusammengesetzt werden.

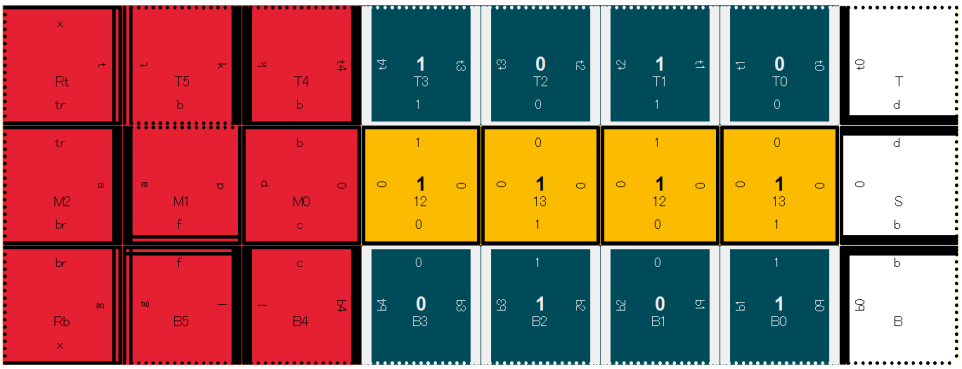


ABBILDUNG 6.7: 4 Bit-Nachrichtenmolekül $\mathcal{M}_{Add}$ im kTAM. Die weißen Tiles bilden die Seed-Struktur. Ozeangrüne Tiles sind Framework-Tiles und rote Tiles bilden den Liganden. Gelbe Tiles beschreiben den Algorithmus.

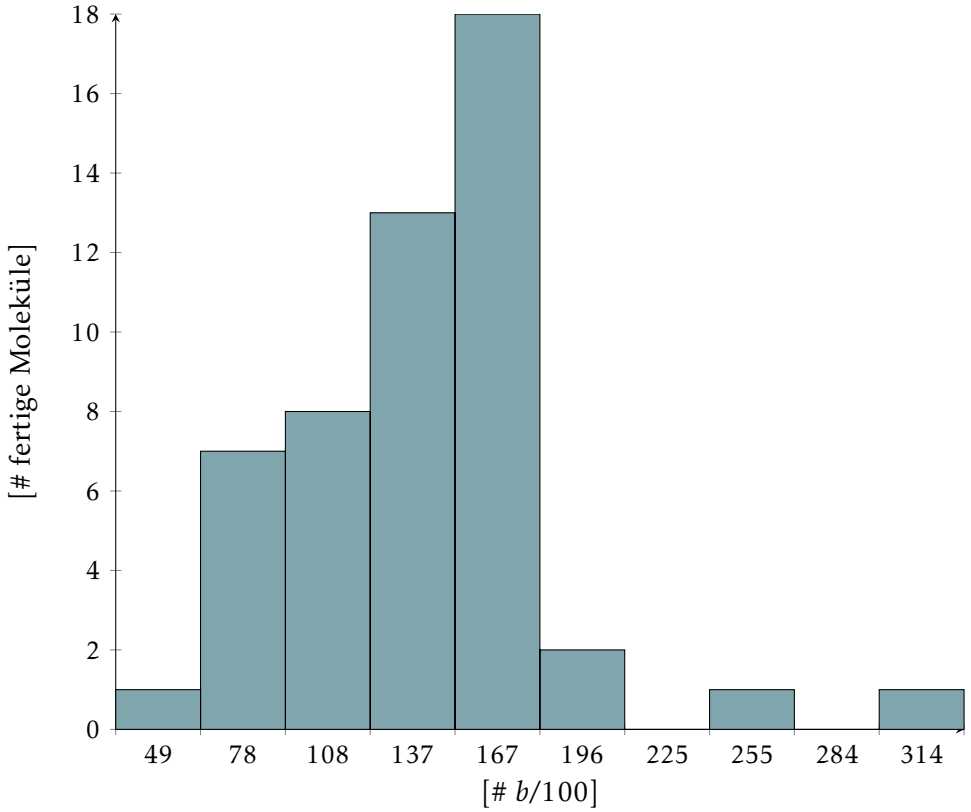


ABBILDUNG 6.8: Ergebnis von 50 kTAM-Simulationen des Nachrichtenmoleküls $\mathcal{M}_{Add}$. Das Histogramm zeigt die Anzahl der vollständigen Moleküle in Bezug auf die benötigten Bindungsreaktionen b.

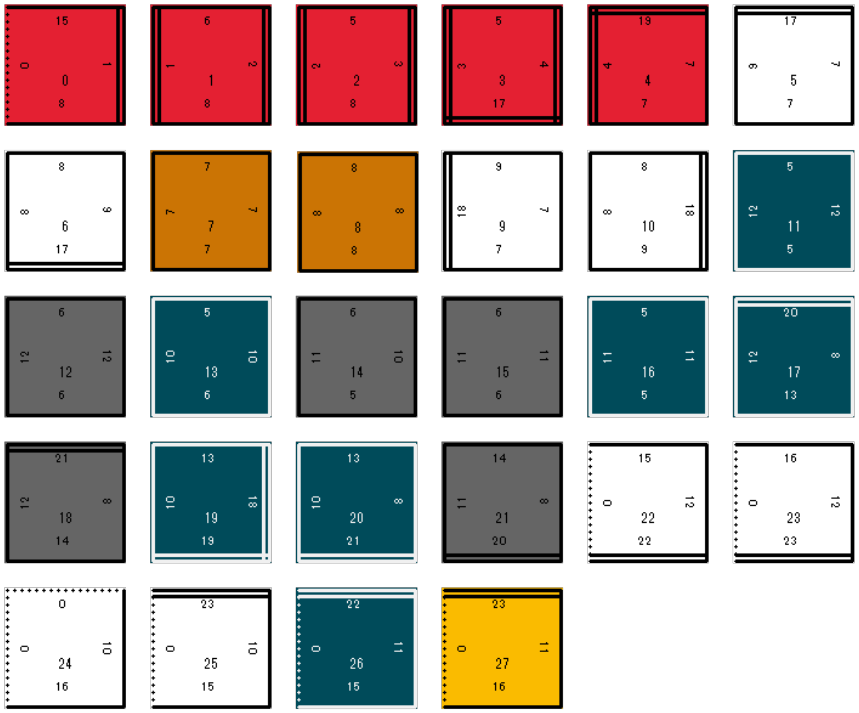


ABBILDUNG 6.9: Tileset logarithmischer Größe (28) im Verhältnis zu Kantenlänge (50) des resultierenden Quadrates.

Abbildung 6.9 zeigt ein Tileset für ein Quadrat der Kantenlänge 50 [157]. Rote Tiles bilden die initiale Reihe des größenbeschränkten Binärzählers. Weiße und gelbe Tiles beschränken das Wachstum des Binärzählers nach links/oben und bilden die diagonale Rahmenstruktur des Quadrates. Orange Tiles füllen den Rahmen. Es wird jeweils ein Tiletyp links und rechts der Diagonalen verwendet. Graue und ozeangrüne Tiles bilden die Logik des Binärzählers ab und stehen für logische „1" und „0".

Abbildung 6.10 zeigt das Ergebnis. Die Simulation wurde im aTAM durchgeführt, da ein Assembly der Größe 2500 eine enorme Simulationsdauer bedeutet. Das aTAM lässt sich deutlich schneller simulieren als kTAM oder 2HAM. Das Tileset ist mit einer Größe von 28 genau so groß wie das Tileset des Nachrichtenmoleküls ADD, lässt jedoch deutlich größere Assemblies entstehen.

Instanziiert man das Tileset weitere fünf Mal in einem dreidimensionalen Self-Assembly-System und passt die äußeren Kleber entsprechend an, lässt sich leicht ein Würfel erzeugen. Dafür sind mindestens 168 verschiedene Tiletypen nötig, da Tiles aus DNA nicht rotiert werden können.

Da es sich bei Quadraten und Würfeln um beliebte Vergleichsstrukturen zur Effizienzmessung von Self-Assembly-Systemen handelt, sind diese bereits gut erforscht und eine genauere Analyse wird ausgelassen [111]

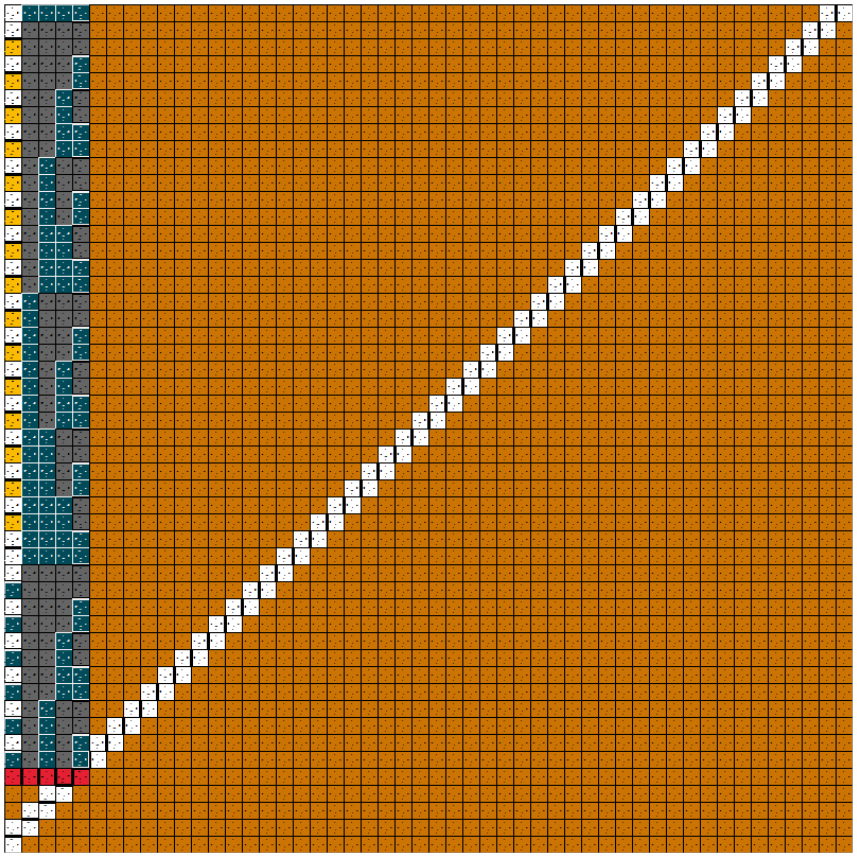


ABBILDUNG 6.10: Quadrat mit Kantenlänge 50 aus logarithmisch großem Tileset im ISU TAS-aTAM-Modul.

6.1.3 ANALYSE DER KTAM-SIMULATIONEN

Die kTAM-Simulationen haben gezeigt, dass die benötigte Anzahl an Bindungsreaktionen pro Ort bis zur stabilen Bindung eines Tiles im Erwartungswert gleich bleibt. Für jede Position der Wachstumsfront kommt potentiell jeder Tiletyp in Frage.

Analytisch lässt sich der Erwartungswert der benötigten Bindungsreaktionen für ein Nachrichtenmolekül bestimmen, indem für jedes Assembly a_i einer gegebenen Assemblysequenz S die Wahrscheinlichkeit einer korrekten Bindung aller Tiles an passenden Orten der Wachstumsfront bestimmt wird. Aufsummiert ergibt sich die Wahrscheinlichkeit, dass eine Bindungsreaktion korrekt ist. Nun kann für eine gewünschte Wahrscheinlichkeit Δ berechnet werden, nach wie vielen Bindungsreaktionen eine korrekte Bindung im Erwartungswert stattgefunden hat. Bildet man die Summe über die Erwartungswerte aller Assemblies a_i, ergibt sich die erwartete Anzahl an benötigten Bindungsreaktionen für ein Nachrichtenmolekül mit gegebener Assemblysequenz S. Der Erwartungswert lässt sich unter der Voraussetzung abschätzen, dass für alle Assemblies a_i aus S gilt, dass $|a_i| \leq |a_{i+1}|$. Der Erwartungswert der benötigten Bindungsreaktionen lässt sich durch eine Worst Case-Betrachtung nach oben abschätzen. Dafür wird

angenommen, dass zu jedem Zeitpunkt nur ein Tile an einem bestimmten Ort der Wachstumsfront w korrekt binden kann. Die Wahrscheinlichkeit hierfür beträgt $1/(|T|\cdot|w|)$. Löst man nun die Ungleichung $1-((|T|\cdot|w|-1)/(|T|\cdot|w|))^p > \Delta$ nach p auf, ergibt sich die benötigte Anzahl an Bindungsreaktionen für ein Assembly a_i aus S. Bei 15 verschiedenen Tiletypen, $\Delta = 0,95$ und einer Wachstumsfront der Größe 4 entspräche dies 179 erwarteten Bindungsreaktionen. Multipliziert man p mit $|S|$, so erhält man den Worst Case-Erwartungswert der benötigten Bindungsreaktionen eines Nachrichtenmoleküls bei gegebener Assemblysequenzlänge $|S|$.

Da eine unendlich große Anzahl potenzieller Assemblysequenzen für jedes Nachrichtenmolekül existiert und die tatsächliche Assemblysequenz aufgrund des Nichtdeterminismus von Self-Assembly-Systemen unbekannt ist, ist eine Simulation einer analytischen Betrachtung vorzuziehen.

Assemblies sind im Allgemeinen fehleranfälliger, wenn mehrere identische Kleber an unterschiedlichen Tiletypen vorkommen. Baut sich ein Assembly reihenweise auf, so treten kaum Fehler auf. Die Wahl der Parameter G_{se} und G_{mc} kann dabei so getroffen werden, dass beliebig wenig Fehler auftreten. Die Assemblydauer nimmt jedoch sehr stark im Verhältnis zu.

Generell lassen sich alle Probleme auf ähnliche Weise wie THRES lösen (siehe Theorem 5.2). Dementsprechend kann angenommen werden, dass die Fehlerraten gering sein werden und die wenigen auftretenden Fehler lediglich einige Nachrichtenmoleküle unbrauchbar machen, sich aber nicht ausbreiten. Als Vergleichswerte dienen die Simulationsergebnisse des THRES-Nachrichtenmoleküls.

6.2 UNTERASSEMBLY-INTERAKTIONEN

Die kTAM-Simulation hat gezeigt, dass Nachrichtenmoleküle sich unter realistischen Bedingungen vollständig zusammensetzen. In diesem Abschnitt wird die Interaktion von Zwischenprodukten sowie Rezeptoren und Nachrichtenmolekülen im 2HAM untersucht. Dafür wird die ISU TAS-Software verwendet.

6.2.1 ANALYSE VON AND-NACHRICHTENMOLEKÜLEN IM 2HAM

Das Tileset für das simulierte Nachrichtenmolekül $\mathcal{M}_{And}$ ist in Abbildung 6.1 dargestellt. Die blauen Tiles zeigen einen Rezeptor. Die übrigen Tiles setzen sich zum Nachrichtenmolekül zusammen. Wie in Tabelle 6.1 dargestellt, existieren zahlreiche Unterstrukturen zeitgleich, da 2HAM ohne Seed-Assembly auskommen. Dies ermöglicht Interaktionen zwischen zwei beliebigen Tiles oder Unterassemblies, welche die Temperaturanforderung τ erfüllen. Bei geschickt gewählten Klebern kommt es zu keinen ungewollten Interaktionen zwischen Unterassemblies. Um fehlerhafte Bindungen zu erzeugen, muss es zu mehreren zeitgleichen Fehlern kommen.

Nur wenn Nachrichtenmolekül und Rezeptor vollständig zusammengesetzt sind, kann es zu einer Interaktion zwischen beiden kommen. Abbildung 6.11 zeigt das gebundene Nachrichtenmolekül für das Problem AND. Die Bindung eines

Größe	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	22
Anzahl	18	12	11	11	11	12	11	11	10	9	7	5	3	2	1	1

TABELLE 6.1: Anzahl von Unterassemblies der Größe k des Nachrichtenmoleküls $\mathcal{M}_{And}$. Es ist zu erkennen, dass viele Möglichkeiten existieren, kleinere Assemblies zu erzeugen. Große Assemblies werden selten und eine Bindung am Rezeptor der ganzen Nachricht existiert nur ein Mal (Größe 22).

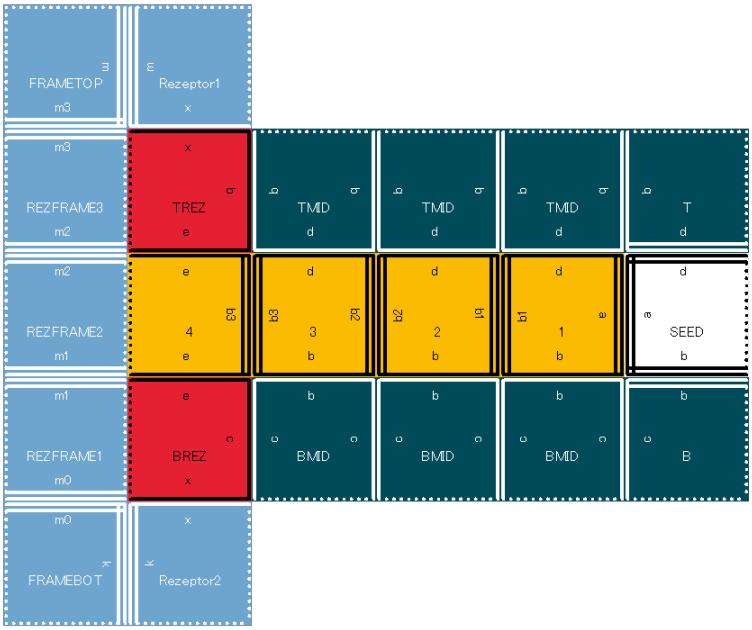


ABBILDUNG 6.11: Bindung eines 4 Bit-Nachrichtenmoleküls $\mathcal{M}_{And}$ im 2HAM an einen Rezeptor (hellblau).

Nachrichtenmoleküls wird lediglich für das Nachrichtenmolekül AND bildlich dargestellt, da der Prozess bei allen Nachrichtenmolekülen identisch ist.

6.2.2 ANALYSE VON THRES-NACHRICHTENMOLEKÜLEN IM 2HAM

Für die 2HAM-Simulation wird das in Abbildung 6.3 dargestellte Tileset der Größe 35 verwendet. Lediglich die gelb dargestellten Tiles sind an der Berechnung des Schwellwerts beteiligt und werden bedingt von Nanosensoren ausgeschüttet. Die vier unterschiedlichen Möglichkeiten einen Schwellwert der Höhe 3 zu erreichen werden durch die Framework-Tiles kodiert.

Tabelle 6.2 zeigt, dass zahlreiche Unterstrukturen zeitgleich existieren. Die 2HAM-Simulation zeigt weiterhin, dass keine wahrscheinlichen Interaktionen der Unterassemblies existieren, welche zu Fehlverhalten führen können. Es existieren keine Assemblies, die aus mehr als 15 Tiles bestehen – der Simulationsprozess gelang nach 15 Schritten zu einem Fixpunkt an dem keine neuen

Größe	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Anzahl	35	24	20	20	20	24	24	24	24	20	16	12	8	4	4

TABELLE 6.2: Anzahl von Unterassemblies der Größe k des Nachrichtenmoleküls $\mathcal{M}_{Thres}$. Es ist zu erkennen, dass viele Möglichkeiten existieren, kleinere Assemblies zu erzeugen. Große Assemblies werden seltener und Assemblies der Größe 15 existieren nur vier Mal. Diese korrespondieren mit den vier vollständigen Nachrichtenmolekülen.

Unterassemblies entstehen. Mit jedem Assembly der Größe 15 korreliert ein THRES-Nachrichtenmolekül.

6.2.3 ANALYSE VON ADD-NACHRICHTENMOLEKÜLEN IM 2HAM

Das in Abbildung 6.6 dargestellte Tileset für ein ADD-Nachrichtenmolekül lässt sich in der ISU TAS-Software nicht sinnvoll simulieren, da Kleber der Stärke 3 oder mehr im 2HAM-Modul zu Programmfehlern führen.

Versuche der Simulationen bei Temperatur 2 haben jedoch gezeigt, dass die Wahrheitswerte der Kleber der Eingabe-Tiles unterschiedlich kodiert werden müssen, damit verhindert werden kann dass diese direkt aneinander binden. Dies gilt folglich ebenso für die Kleber der Input-Tiles.

Abbildung 6.12 zeigt eine mögliche Anpassung für Systeme ohne dediziertes Seed-Assembly. Die Bits der Kleber der unteren Reihe der Eingabebits wurden um ein weiteres identisches Bit erweitert. Die Bits der Kleber der Input-Tiles wurden um zwei weitere identische Bits erweitert.

Insgesamt gilt es die Anzahl der Kleber, welche in verschiedenen Tiletypen verwendet werden zu minimieren, um Fehlern vorzubeugen. Ein fehlerfreier Assembly-Prozess wird dadurch wahrscheinlicher. Außerdem sollten die Assemblies nur in möglichst wenig Richtungen gleichzeitig eine Wachstumsfront bereitstellen. Im besten Fall ist jeder Schritt vom vorherigen abhängig. Je größer die Zielstruktur, desto wahrscheinlicher werden unerwünschte Interaktionen von Zwischenprodukten.

6.2.4 FAZIT ZUR 2HAM-SIMULATION

Die Simulationen im 2HAM haben gezeigt, dass sich Nachrichtenmoleküle genau dann möglichst fehlerfrei aufbauen, wenn zu jedem Zeitpunkt nur eine geringe Menge potenzieller Orte der Wachstumsfront eine korrekte Bindung zulassen. Ergebnisse mit besonders wenig Fehlern lassen sich erzielen, wenn zu jedem Zeitpunkt genau ein Ort der Wachstumsfront eine korrekte Bindung zulässt. Auf diese Weise kann sichergestellt werden, dass die Platzierung jedes Tiles von einer Anzahl Nachbarn entsprechend der Temperatur τ abhängt. Die auf diese Weise entstandenen Unterassemblies können nicht korrekt miteinander interagieren.

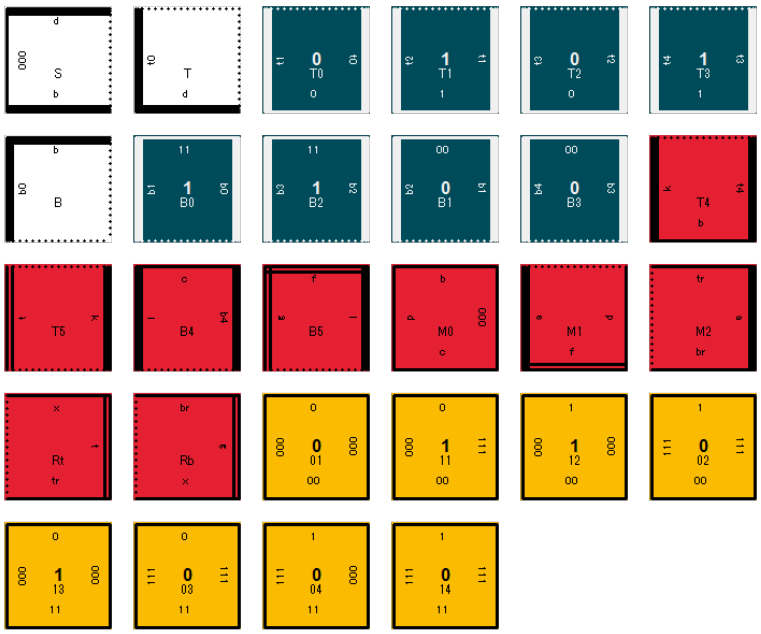


ABBILDUNG 6.12: Das Tileset für das Nachrichtenmolekül ADD wurde so angepasst, dass möglichst wenig Kleber in mehreren Tiletypen verwendet werden.

Die geringen Fehlerraten bedingen jedoch eine höhere Assemblydauer. Ist zu jedem Zeitpunkt nur ein Ort korrekt, so sind effektiv keine parallelen Bindungsreaktionen möglich.

Eine Voraussetzung für problematische Interaktionen ist zudem die Anwesenheit von identischen Klebern in verschiedenen Tiletypen. Treten unter diese Bedingung lange oder große Unterassemblies auf, welche mehrere korrekte Wachstumsfrontorte haben, kann kaum vermieden werden, dass eine passende Untersequenz von Klebern in anderen Unterassemblies existiert.

Wird kein Kleber an mehr als zwei Tiletypen verwendet, so sollten parallele Bindungsreaktionen an verschiedenen Orten der Wachstumsfront kein Problem darstellen. Bei Berechnungen ist diese Bedingung jedoch sehr selten einhaltbar, weswegen mit anderen Mitteln der Fehlervermeidung vorlieb genommen werden muss.

Da bereits bei einigen Nanonetzwerkbestandteilen unerwünschte Interaktionen von Zwischenprodukten aufgetreten sind, empfiehlt es sich Komponenten in getrennten Medien zu erzeugen und zu einem späteren Zeitpunkt zu einem Nanonetzwerk zusammenzuführen.

6.3 WET-LAB-EXPERIMENT EXTRAPOLATION

[118] stellt die Ergebnisse eines Wet-Lab-Experiments mit DNA-Tiles vor. In besagtem Experiment wurde der Versuch unternommen, ein Sierpinski-Dreieck

aus Tiles zu erschaffen. Das Sierpinski-Dreieck entsteht bei der Berechnung eines XOR in Self-Assembly-Systemen.

Reale Experimente entsprechen einer Mischung aus kTAM und 2HAM. Es wurde eine Tilekonzentration von 0,2 µM für jedes Tile verwendet. Der Wert entspricht einem Viertel der simulierten Konzentration, was in etwa der Salzkonzentration im menschlichen Blut (0,14 µM) entspricht. Die Ergebnisse deuten an, dass Strukturen aus 100 Tiles mit niedrigen Fehlerraten in 60–96 min erwachsen können.

Da die vorgestellten Nachrichtenmoleküle nur aus 15–24 Tiles bestehen, ist zu erwarten, dass die Fehlerrate und die benötigte Assemblydauer geringer sind. In Wet-Lab-Experimenten würden sich demnach wenigstens 0,2 µM pro Tile empfehlen.

Im Folgenden werden die Ergebnisse von den Simulationen mit dem Wet-Lab-Experiment verglichen und auf Basis dessen wird extrapoliert.

6.3.1 SIMULATION UND EXPERIMENT IM VERGLEICH

Ein Vergleich zwischen Simulation und Wet-Lab-Experiment verfügt nur über eine eingeschränkte Aussagekraft. Bei einer Tilekonzentration von 0,8 µM pro Tile wurde ein Nachrichtenmolekül AND in 3000 Bindungsreaktionen fertiggestellt. Im Wet-Lab-Experiment waren die gewünschten Strukturen bei einer Konzentration von 0,2 µM nach einer Stunde fertig.

Da die Nachrichtenmoleküle – im Gegensatz zu den 100 Tiles des Wet-Lab-Experiments – nur aus 15–24 Tiles bestehen, ist damit zu rechnen, dass sie schneller finalisiert wären. Extrapoliert man, so kommt man auf ca. 10–16 Minuten bis zur Vervollständigung der Assemblies. Generell wachsen größere Assemblies im Labor schneller als kleine, da die Wachstumsfront mehr Orte für Bindungen enthält. Allerdings ist zu berücksichtigen, dass die Wachstumsfront des Wet-Lab-Experimentes zwar sehr groß war, die Anzahl der unterschiedlichen Tiletypen mit 15–35 jedoch gering. In Simulationen führt diese Ausgangssituation zu einer geringen Anzahl benötigter Bindungsreaktionen. In der Realität ist ebenfalls mit einem solchen Effekt zu rechnen. Allerdings ist davon auszugehen, dass die Gesamtzeit bei vielen Tiletypen kaum verändert wird, wenn die absolute Tilekonzentration erhöht wird. Das ist beim Hinzufügen von weiteren Tiletypen gängige Praxis. Durch die erhöhte Gesamtkonzentration werden lediglich mehr Bindungsreaktionen pro Zeiteinheit stattfinden.

Die Menge benötigter Nanoroboter und Nanosensoren lässt sich nach unten beschränken. Damit sichergestellt werden kann, dass für jedes Seed-Tile ein Nachrichtenmolekül entstehen kann, müssen ebenso viele Input-Tiles und andere Eingaben ausgeschüttet werden können. Wenn die Tiles in einer DNA-Box mit den Dimensionen $42 \cdot 36 \cdot 36$ nm^3 platziert werden [34], so kann mit 472 Tiles der Größe $14,4 \cdot 4 \cdot 2$ nm^3 [118] gerechnet werden. Wenn die vorgestellten Quadrate der Kantenlänge 50 zu einer Box gefaltet werden, könnten 8 680 555 Tiles darin untergebracht werden.

Bei angenommener Gleichverteilung würden so $1,275 \cdot 10^{10}$ DNA-Boxen ausreichen, um die benötigten Tiles bei einem Medienvolumen von 50 µl zu fassen [118]. Wird hingegen eine Box mit Kantenlänge 50 verwendet, wären es 694 008 Boxen. Wie die Tiles auf die Nanosensoren verteilt werden, kann nach Belieben entschieden werden.

In [118] wurde die Anzahl aller benötigten Komponenten großzügig nach oben abgeschätzt, weswegen sich dieses Vorgehen für Wet-Lab-Experimente mit Nachrichtenmolekülen anbietet.

Vollständig zusammengesetzte Nachrichtenmoleküle verhalten sich wie reguläre Moleküle, die zur Kommunikation eingesetzt werden. Sie unterliegen ebenfalls der Brownschen Molekularbewegung. Für eine genaue Analyse des Verhaltens von Molekülen, die zur Kommunikation auf Nanoebene eingesetzt werden, kann [64] konsultiert werden.

Zusammenfassend ist zu erwarten, dass Nachrichtenmoleküle und Rezeptoren sich korrekt zusammensetzen und in einem kontrollierten Umfeld wie einer Petrischale sinnvoll interagieren können. Eine hohe Dichte ist außerhalb von lebenden Organismen unbedenklich.

6.4 KOMPLEXITÄT VON NACHRICHTENMOLEKÜLEN

Dieser Abschnitt untersucht verschiedene Nachrichtenmoleküle in Hinblick auf ihre Komplexität und setzt diese mit Schaltkreisen in Beziehung.

Die Komplexität von verschiedenen Operationen unterscheidet sich von Berechnungsmodell zu Berechnungsmodell. Die komplexitätstheoretischen Betrachtungen aus Kapitel 2 beziehen sich lediglich auf Schaltkreise – bei Self-Assembly-Systemen sind andere Parameter von Interesse. Tabelle 6.3 zeigt eine Gegenüberstellung von Schaltkreisimplementierungen und Nachrichtenmolekülen. Für die gezeigten Probleme sind häufig nur obere Schranken bekannt. Diese beziehen sich auf die beste bekannte Implementierung. Für das logische OR kann in Self-Assembly-Systemen eine untere Schranke angegeben werden, da hierfür lediglich ein Tile benötigt wird.

In Kapitel 5 wurden die Operationen AND, ADD und THRES genauer untersucht und als Nanonetzwerke modelliert. Diese Probleme entsprechen den Schaltkreisklassen AC^0 (AND und ADD) und NC^1 (THRES), es handelt sich folglich um eher einfache Operationen. Dieselben Operationen benötigen in Self-Assembly-Systemen für AND ein Tileset linearer Größe, für ADD ein Tileset konstanter Größe und für THRES ein Tileset überexponentieller Größe. Dies zeigt, dass eine einfache Implementierbarkeit durch Schaltkreise nicht notwendigerweise bedeutet, dass diese Probleme ebenfalls leicht durch Self-Assembly-Systeme umsetzbar sind.

Die meisten der präsentierten Nachrichtenmoleküle fallen in die Klasse NC^1, haben jedoch teilweise eine stark abweichende Tile-Komplexität. Im Allgemeinen scheint die Tile-Komplexität kein geeignetes Komplexitätsmaß für Nachrichten-

Problem	Platzklasse	Assemblygröße	Tile-Komplexität
ADD	AC^0	$\mathcal{O}(n)$	$\mathcal{O}(1)$
GEQ, LEQ, EQ	NC^1	$\mathcal{O}(n)$	$\mathcal{O}(n)$
SUB	AC^0	$\mathcal{O}(n)$	$\mathcal{O}(1)$
MULT	NC^1	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
DIV	NC^1	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
INC	AC^0	$\mathcal{O}(n)$	$\mathcal{O}(\log(n))$
AND	AC^0	$\mathcal{O}(n)$	$\mathcal{O}(n)$
OR	AC^0	$\mathcal{O}(1)$	$\mathcal{O}(1)$
XOR	AC^0	$\mathcal{O}(n)$	$\mathcal{O}(1)$
ODD, EVEN	AC^0	$\mathcal{O}(1)$	$\mathcal{O}(1)$
MIN, MAX	NC^1	–	–
THRES	NC^1	$\mathcal{O}\, n$	$\mathcal{O}\binom{n}{k}$
IT-ADD	NC^1	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
MOD	NC^1	$\mathcal{O}(n^2)$	$\mathcal{O}(1)$
AVG	NC^1	–	–
MEDIAN	L	–	–
CUBE	–	$\mathcal{O}(n^3)$	$\mathcal{O}(\log n)$

TABELLE 6.3: Komplexität von Problemen in Relation zur Eingabegröße n, welche von Interesse für Nanonetzwerke sind, im Vergleich zu Schaltkreisen. Es sind jeweils obere Schranken angegeben. Die Tile-Komplexität berücksichtigt dabei nicht die Eingabe. Die Größe eines Assemblies entspricht der Anzahl der verwendeten Tiles.

moleküle zu sein. Die Tile-Komplexität wird als Standardmaß für die Komplexität von Self-Assembly-Systemen eingesetzt. Ein Großteil der benötigten Tiles bei AND und THRES geht auf die benötigte Rahmenstruktur zurück, welche der Eingabe der Berechnung eine Bindungsmöglichkeit liefert – die Berechnung selbst ist weiterhin einfach.

Ein sinnvoller, weiterführender Ansatz ist eine Bestimmung der Anzahl der algorithmisch relevanten Tiletypen oder unterschiedlichen Kleber. Beim Problem ADD ist dies bereits der Fall, da lediglich das Tileset konstanter Größe gezählt wird, welches an der Bildung der Ausgabe beteiligt ist. Framework-Tiletypen und Eingabe sind identisch. Beim Problem AND gleicht sich die Komplexität der Schaltkreisklasse an, da lediglich eine konstante Tile-Komplexität nach Abzug von Eingabe- und Framework-Tiletypen übrig bleibt. Auch das Problem THRES reduziert sich bei identischem Vorgehen bezüglich seiner Tile-Komplexität. Lediglich eine lineare Anzahl algorithmisch relevanter Tiletypen bleibt.

KAPITEL 7

ZUSAMMENFASSUNG UND AUSBLICK

In diesem Kapitel werden die Ergebnisse der Arbeit im Detail diskutiert und dargelegt, welche der gesetzten Zielstellungen erreicht wurden. Außerdem werden offene Punkte und Fragestellungen erläutert. Themen, welche in Zukunft von Interesse sein können, werden diskutiert. Es wird zudem eine philosophische, weiterführende Fragestellung untersucht.

7.1 ZUSAMMENFASSUNG

In der vorliegenden Arbeit wurde das vorherrschende Paradigma der Kommunikation und Berechnung auf Nanoebene grundlegend erweitert. Es wurde ein Konzept vorgestellt, welches Berechnungen in den Zusammensetzungsprozess von Nachrichten im Kommunikationskanal integriert. Dadurch werden bislang angenommene Platzrestriktionen innerhalb von Nanogeräten verringert.

Um dies in einem ganzheitlichen Modell zu ermöglichen, wurde eine Modellierung von Nanonetzwerken und deren Bestandteilen auf Basis von DNA-Tiles unternommen. Nanosensoren, Nanobots und Nachrichten wurden als einfache Tilesets betrachtet und modelliert. Diese finden sich im geteilten Kommunikationskanal zu den gewünschten Nanostrukturen zusammen.

Für die physikalische Umsetzung werden ausschließlich DNA-basierte Technologien benötigt. Diese Technologien sind allesamt bereits im Labor herstellbar, weswegen die präsentierte Technologie als leicht realisierbarer Ansatz, primitive Nanonetzwerke herzustellen, interpretiert werden kann.

Die folgenden Paragraphen verdeutlichen die wichtigsten Schritte und Erkenntnisse kapitelweise.

In Kapitel 1 wurde eine Intuition zum Thema Nanotechnologien vermittelt. Es wurde erklärt, wie die historische Entwicklung durch Feynman zunehmendes

Interesse erlangten. Aufbauend darauf wurde verständlich gemacht, warum zu erwarten ist, dass gerade medizinische Nanotechnologien sich in naher Zukunft weiterentwickeln werden. Nach der Zukunftsprognose wurden allgemeine, medizinische Anwendungsszenarien für Nanotechnologien aufgezeigt, um den Sinn und Nutzen des Forschungszweiges zu untermauern. Um einen zu starken Fokus zu vermeiden, wurden zusätzlich eine Reihe nicht-medizinischer Szenarien vorgestellt. Zuletzt erfolgten Auflistungen der Beiträge der Arbeit und eine kurze Strukturierung.

Kapitel 2 definiert die mathematischen und biologischen Grundlagen der Arbeit. Es wurde dabei mit einer historischen Einordnung und Darstellung der relevanten Technologien begonnen. Danach erfolgte eine detaillierte Erklärung der biologischen Grundlagen von DNA und den aus ihr gefertigten, komplexeren Technologien. Dabei lag der Fokus auf DNA-Tiles, welche als physikalische Basis für die folgenden mathematischen Modellierungen dienten.

Im Anschluss daran wurden zunächst diverse Klassen von Geräten sowie ihre wichtigsten Komponenten definiert. Diese stellen eine notwendige Basis für die späteren Maschinenmodelle dar. Außerdem wurde erklärt, was unter Nanonetzwerken verstanden wird und wie deren Bestimmungsort beschaffen ist.

Es folgte eine Kurzeinführung in die Komplexitätstheorie, welche erforderlich ist, um präsentierte Lösungsansätze bewerten zu können. Außerdem wurden die notwendigen Grundlagen geschaffen, um zu verstehen, was im Kontext der Arbeit unter einer Berechnung verstanden wird.

Aufbauend darauf konnten verschiedene Berechnungsmodelle voneinander abgegrenzt werden. Ein Vergleich und eine Einschätzung haben ergeben, dass Self-Assembly-Systeme für die gegebene Problemstellung das vielversprechendste Berechnungsmodell auf Nanoebene darstellen. Self-Assembly-Systeme können zur Beantwortung von drei unterschiedlichen Fragestellungen auf einmal verwendet werden. Dabei handelt es sich um Nanoberechnung, Nanokonstruktion und Nanokommunikation.

Nachdem die Wahl der Technologie getroffen und begründet wurde, konnte der zugrundeliegende, mathematische Definitionsapparat von Self-Assembly-Systemen verständlich erklärt werden. Der Fokus lag auf dem kTAM und dem 2HAM, welche wichtige Evaluationswerkzeuge für die später vorgestellten Konzepte darstellen.

Kapitel 3 stellt die verschiedenen Arten von Fehlern vor, welche in Self-Assembly-Systemen wie dem kTAM oder dem 2HAM auftreten können. Dabei wurden Definitionen für Growth-Errors, Facet-Errors und Nukleations-Errors präsentiert. Im Anschluss wurde ein neues Verfahren vorgestellt, welches zeigt, wie Fehler in dreidimensionalen Self-Assembly-Systemen vermieden oder reduziert werden können. Dieses Verfahren wird 3D-Snaked-Proofreading genannt und verhindert Facet-Errors in allen Dimensionen. Growth-Errors werden um einen Faktor, welcher kubisch mit der Kantenlänge der Blockersetzungsstrategie wächst, reduziert.

Kapitel 4 analysiert eine Vielzahl medizinischer und nicht-medizinischer Anwendungsszenarien. Es wurde untersucht, welche Anforderungen in der For-

schungsgemeinschaft an Nanonetzwerke und Nanogeräte gestellt werden. Im Anschluss daran wurden die Anforderungen formalisiert und in ihre elementaren Operationen zerlegt. Es hat sich gezeigt, dass die an Nanogeräte gestellten Anforderungen stark variieren. Das Spektrum reicht von wenigen arithmetischen Operationen bis hin zu der Mächtigkeit makroskopischer CPUs.

Aufbauend auf der Dekomposition konnte eine Einteilung in Komplexitätsklassen erfolgen. Der Fokus lag auf dem für eine Berechnung benötigten Platz. Es wurden die drei Klassen AC^0, NC^1 und L gewählt und verwendet, um verschiedene Abstufungen der Mächtigkeit von Geräten zu beschreiben. Die gefundenen Probleme wurden in drei korrespondierende Komplexitätsklassen aufgeteilt. Das Ergebnis der Kategorisierung ist, dass ein Gerät einer Klasse mit hoher Wahrscheinlichkeit – unter geringem Aufwand – Probleme der gleichen und eingeschlossenen Klassen lösen kann.

In einem weiterführenden Schritt wurden häufig geforderte Operationen für eine erweiterte Analyse und Konzeption als Nanonetzwerk ausgewählt.

Um eine möglichst aussagekräftige Evaluation zu ermöglichen, wurden zudem diverse Simulationswerkzeuge gesichtet und analysiert. Das Ergebnis der Analyse war, dass der Simulator ISU TAS die meisten Anforderungen erfüllt. aTAM, kTAM und 2HAM können mit demselben Tileset im ISU TAS simuliert werden. Lediglich eine Netzwerksimulation wurde nicht angeboten. Generell wurde festgestellt, dass sich Nachrichtenmoleküle, nach ihrer vollständigen Zusammensetzung, analog zu allgemeinen zur molekularen Kommunikation eingesetzten Molekülen verhalten, welche bereits gut erforscht sind.

Kapitel 5 stellt den innovativsten Teil des wissenschaftlichen Beitrags dieser Arbeit dar. Es befasst sich mit der Erzeugung von Tilesets für Nanonetzwerke und deren Komponenten. Dafür wurde zuerst definiert, was mathematisch unter Nachrichtenmolekülen verstanden wird und wie sich Liganden sowie Rezeptoren aufbauen können. Anhand eines Beispielszenarios für die Berechnung eines 4 Bit-AND wurde eine Referenzarchitektur für allgemeine Nanonetzwerke abgeleitet und eine generelle Tileset-Definition von Nanonetzwerken und Nachrichtenmolekülen vorgestellt.

Im Anschluss daran wurden Algorithmen steigender Effizienz vorgestellt, welche zur Erzeugung von beliebig großen Quadraten oder Würfeln verwendet werden können. Diese dienen als grundlegende Nanostruktur für die Erzeugung von Nanosensoren und Nanobots. Der logarithmische Algorithmus benötigt dabei lediglich 168 Tiletypen, um einen hohlen Würfel der Kantenlänge 50 zu erzeugen. Dies entspricht einer logarithmischen Skalierung im Verhältnis zur Kantenlänge des Würfels oder Quadrates.

Danach wurden Nachrichtenmoleküle für eine große Auswahl von mathematischen Funktionen konzipiert und erklärt. Des Weiteren wurde ein allgemeines Vorgehen bewiesen, welches für beliebige aussagenlogische Formeln ein korrespondierendes Nachrichtenmolekül ermitteln kann, welches dasselbe Problem löst. Für Funktionsprobleme lässt sich das Vorgehen jedoch nicht anwenden, weswegen einfache arithmetische Operationen in Form eines Nachrichtenmoleküls als Proof-of-Concept umgesetzt wurden.

Nanosensoren, Nanobots und Nachrichtenmoleküle wurden für die in Kapitel 4 ermittelten Probleme als vollständige Nanonetzwerke modelliert. Es handelt sich dabei um eine verteilte Einigung über Ereignisse, die Erkennung einer signifikanten Konzentration von Markern anhand von Thresholds und die Addition von Zahlen, ausgelöst durch Ereignisse.

In Kapitel 6 wurden die Nanonetzwerke aus Kapitel 5 durch Simulationen auf ihrer Korrektheit überprüft. Zusätzlich wurde Extrapolation von bereits durchgeführten Wet-Lab-Experimenten verwendet, um sinnvolle Simulationsparameter zu gewinnen und generelle Schwachstellen ausfindig zu machen. Außerdem konnte so die benötigte Zeit für die Zusammensetzung eines Assemblies approximiert werden.

Durch kTAM-Simulationen hat sich herausgestellt, dass die Nachrichtenmoleküle AND und THRES sich mit einer Wahrscheinlichkeit von weniger als 3,5 % fehlerhaft zusammensetzen. Bei der Addition ADD kam es zu vergleichsweise vielen Problemen. Das entsprechende Nachrichtenmolekül müsste bei Temperatur 3 simuliert werden, was die ISU TAS-Software theoretisch zulässt. Praktisch kam es jedoch zu Fehlern. In einer Ersatzsimulation bei Temperatur 2 kam es in 14 % aller Fälle zu Fehlern, welche die weitere Zusammensetzung stoppten. Die Simulation wurde testweise nach der vollständigen Zusammensetzung der Nachrichtenmoleküle fortgeführt, um die Stabilität von Nachrichtenmolekülen über die Zeit zu ermitteln. Selbst nach 100 000 Bindungsreaktionen kam es nach einer vollständigen Zusammensetzung eines Nachrichtenmoleküls nicht zu fehlerhafter Hinzufügung von ungewollten Tiles oder Ablösung von erforderlichen Teilen eines Nachrichtenmoleküls.

In 2HAM-Simulationen hat sich gezeigt, dass die Zwischenprodukte eines Selbstzusammensetzungsprozesses fehlerhaft miteinander interagieren können. In der Realität existieren viele Zwischenprodukte zeitgleich und kein dediziertes Seed-Assembly startet den Prozess – die Zusammensetzung beginnt bei beliebigen passenden Komponenten. Vor allem beim Nachrichtenmolekül ADD kam es zu Schwierigkeiten, da viele Kleber an unterschiedlichen Tiletypen wiederverwendet wurden. Außerdem hat sich gezeigt, dass sich ein Nachrichtenmolekül im Optimalfall reihenweise aufbaut, damit die Anzahl der verschiedenen, potentiell kompatiblen Unterassemblies möglichst gering bleibt.

Zusammenfassend ergab sich, dass die ausgewählten Nanonetzwerke unter realistischen Bedingungen und Voraussetzungen ihre definierte Aufgabe erfüllen können. Alle dafür benötigten Komponenten wurden durch Simulation überprüft.

7.2 AUSBLICK

In der Nanogeräteforschung existieren bislang keine mit verfügbaren Technologien realisierbaren Konzepte für Nanonetzwerke, welche Konstruktion, Kommunikation und Berechnung in einem einheitlichen Modell zusammenführen. Es existieren lediglich Teillösungen deren Kompatibilität ungeklärt ist. Diese Arbeit präsentiert damit einen revolutionären Lösungsansatz für eines der

fundamentalsten, ungelösten Probleme in der Nanogeräteforschung auf Basis des Moleküls DNA.

Bei der Bearbeitung der Problemstellungen haben sich einige zusätzliche Fragestellungen aufgetan. In manchen Fällen hat weitere Forschung dazu geführt, dass eine erneute Analyse notwendig wird.

2017 wurde die Analyse der medizinischen Szenarien durchgeführt. Seitdem wurden zahlreiche weitere Vorschläge gemacht, welche einen sinnvollen Beitrag zur Motivation leisten. Darüber hinaus kann es sein, dass zusätzliche mathematische Operationen im Zuge der Weiterentwicklungen interessant geworden sind. Eine erneute Analyse erscheint unumgänglich, wenn für viele Szenarien Lösungen in Form von DNA-Tile-basierten Nanonetzwerken bereitgestellt werden sollen.

Des Weiteren können mehr Funktions- und Entscheidungsprobleme als Nachrichtenmoleküle modelliert werden, um eine Art Bibliothek zur Lösung von Problemen in Form von Nanonetzwerken bereitzustellen.

In dieser Arbeit wurde vom Prinzip der Modularisierung Gebrauch gemacht – wie es auch in weiten Teilen der empirischen Wissenschaften gängige Praxis ist. Dabei wird ein Problem in seine Bestandteile zerlegt, gelöst und im Anschluss wieder zusammengesetzt. Dies geschieht in der Hoffnung, dass sich die Teillösungen zu einer Lösung des ursprünglichen Problems zusammenfügen lassen. Das muss nicht der Fall sein. Deswegen bietet sich eine realitätsnahe Netzwerksimulation von Selbstzusammensetzungsprozessen an. Dadurch könnte die generelle Tauglichkeit des DNA-Tile-basierten Nanonetzwerke-Paradigmas weiter untermauert werden. Dabei ist es nötig, dass Zusammensetzung und Netzwerkinteraktion gemeinsam und kompatibel simuliert werden. Werkzeuge hierfür stehen zurzeit nicht zur Verfügung und müssen entwickelt werden.

Ein weiterer wichtiger Punkt sind Wet-Lab-Experimente. Nichts belegt die Funktionsfähigkeit einer neuartigen Technologie so gut wie ein positiver, empirischer Test. Dafür kann von der Problemmodularisierung aus Kapitel 5 Gebrauch gemacht werden. Dies geschieht allerdings ebenfalls unter der Annahme, dass sich die Teillösungen sinnvoll rekombinieren lassen. Dies ist zwar nicht garantiert, aber oft die beste bekannte Strategie. Da alle Einzelprobleme ebenfalls durch DNA-Technologien lösbar erscheinen, kann jedoch generell davon ausgegangen werden.

Die vorgestellte Technologie der DNA-Tile-basierten Nanonetzwerke verdeutlicht weiterhin, wie mit Partikelsystemen gerechnet werden kann. Dies führt zu der Fragestellung, ob es in lebenden, ausreichend komplexen Organismen ähnliche, natürlich entstandene Systeme gibt. Sind Hormone oder andere Moleküle im menschlichen Körper eventuell an Berechnungen beteiligt? Und was ist die Bedeutung für die Entwicklung künftiger Medikamente? Nach intensiver Auseinandersetzung mit rechnenden Partikelsystemen bin ich zu der persönlichen Vermutung gekommen, dass ein natürliches Vorkommen fast unumgänglich ist.

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

QUELLTEXTVERZEICHNIS

PUBLIKATIONSVERZEICHNIS

ARTIKEL AUS FACHZEITSCHRIFTEN

[1] Florian Lau, Florian Büther, Regine Geyer und Stefan Fischer: „Computation of decision problems within messages in DNA-tile-based molecular nanonetworks“ · *Nano Communication Networks* (2019) · ISSN: 1878-7789 · DOI: https://doi.org/10.1016/j.nancom.2019.05.002 · URL: http://www.sciencedirect.com/science/article/pii/S1878778919300018

[2] Florian Lau, Kristof Stahl und Stefan Fischer: „Techniques for the Generation of Arbitrary Three-Dimensional Shapes in Tile-Based Self-Assembly Systems“ · *Open Journal of Internet Of Things (OJIOT)* Bd. 4, Nr. 1 (2018) · Special Issue: Proceedings of the International Workshop on Very Large Internet of Things (VLIoT 2018) in conjunction with the VLDB 2018 Conference in Rio de Janeiro, Brazil., S. 126–134 · ISSN: 2364-7108 · URL: https://www.ronpub.com/ojiot/OJIOT_2018v4i1n10_Lau.html

KONFERENZBEITRÄGE

[3] Philipp Bende, Florian Lau und Stefan Fischer: „Error-Resistant Scaling of Three-Dimensional Nanoscale Shapes on the Basis of DNA-Tiles“ · *6th ACM International Conference on Nanoscale Computing and Communication 2019 (ACM NanoCom'19)* · Dublin, Ireland, Sep. 2019

[4] Florian Büther, Florian Lau, Marc Stelzner und Sebastian Ebers: „A Formal Definition for Nanorobots and Nanonetworks“ · *The 17th International Conference on Next Generation Wired/Wireless Advanced Networks and Systems + The 10th conference on Internet of Things and Smart Spaces* · St.Petersburg, Russia: Springer, Sep. 2017

[5] Florian Lau, Florian Büther und Bennet Gerlach: „Computational Requirements for Nano-Machines: There is Limited Space at the Bottom“ · *4th ACM International Conference on Nanoscale Computing and Communication* · ACM NanoCom'17 · in press · Washington DC, USA, Sep. 2017

[6] Florian Lau und Stefan Fischer: „Embedding Space-Constrained Quantum-Dot Cellular Automata in Three-Dimensional Tile-Based Self-Assembly Systems“ · *4th ACM International Conference on Nanoscale Computing and Communication 2017* · Washington DC, USA: ACM, Aug. 2017

[7] Marc Stelzner, Florian Lau, Katja Freundt, Florian Büther, Mai Linh Nguyen, Cordula Stamme und Sebastian Ebers: „Precise Detection and Treatment of Human Diseases Based on Nano Networking“ · *11th International Conference on Body Area Networks* · EAI · Turin, Italy, Dez. 2016

ABSCHLUSSARBEITEN

UNTERSTÜTZTE ABSCHLUSSARBEITEN

[8] Philipp Bende: „Error Correction in Three-Dimensional Tile Assembly Systems“ · Masterarbeit · Universität zu Lübeck, 2018

[9] Tobias Braun: „Circuit Minimization in Quantum-Dot Cellular Automata“ · Masterarbeit · Universität zu Lübeck, 2017

[10] Kaspar David Buss: „Circuit Minimization in Quantum-Dot Cellular Automata With Genetic Algorithms“ · Masterarbeit · Universität zu Lübeck, 2017

[11] Kristof Stahl: „Algorithmic Complexity of Simple Structures in Three-Dimensional Tile Assembly Models“ · Masterarbeit · Universität zu Lübeck, 2017

[12] Rico Wysocki: „Efficient Algorithms for Creating Cubes in DNA Self-Assembly Systems“ · Bachelorarbeit · Universität zu Lübeck, 2019

LITERATURVERZEICHNIS

DISSERTATIONEN, FACHBÜCHER UND ABSCHLUSSARBEITEN

[13] Monique Augustine Vigu Axelos und Marcel Van de Voorde: *Nanotechnology in Agriculture and Food Science* · Applications of Nanotechnology · Wiley, 2017 · ISBN: 9783527697731 · URL: https://books.google.de/books?id=RtZRDgAAQBAJ

[14] Arthur Charles Clarke: *The Next Tenants* · Englisch · Tales from the White Hart · 1957 · URL: http://www.isfdb.org/cgi-bin/title.cgi?56730

[15] Prof. Dr. Peter Clote und Prof. Dr. Evangelos Kranakis (auth.): *Boolean Functions and Computation Models* · 1. Aufl. · Texts in Theoretical Computer Science. An EATCS Series · Berlin, Heidelberg: Springer, 2002

[16] Ismo Hänninen und Jarmo Takala: *Arithmetic Design on Quantum-Dot Cellular Automata Nanotechnology* · hrsg. von Mladen Bereković, Nikitas Dimopoulos und Stephan Wong · Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, S. 43–52 · ISBN: 978-3-540-70550-5 · DOI: 10.1007/978-3-540-70550-5_6 · URL: http://dx.doi.org/10.1007/978-3-540-70550-5_6

[17] Robert Freitas Jr.: *Nanomedicine, Volume I: Basic Capabilities* · Georgetown, Texas: Landes Bioscience, Apr. 1999

[18] Stanislaw Lem: *Pokój na Ziemi* · Polish · Harcourt Brace, 1984

[19] Stanislaw Lem: *The Invincible* · Polish · Wydawnictwo MON, 1964

[20] Mai Linh Edith Nguyen: „Analyse und Evaluation von Simulatoren für die Kommunikation in Nanonetzwerken“ · Masterarbeit · Universität zu Lübeck, 2017

[21] Bernd Page: *Diskrete Simulation: Eine Einführung mit Modula-2* · Springer-Lehrbuch · Springer Berlin Heidelberg, 1991 · ISBN: 978-3-642-76862-0

[22] Kim Kristin Scharringhausen: „Nanosensorik im menschlichen Körper: Entwicklung eines Systemkonzepts für Nanogeräte im Blutkreislauf“ · Masterarbeit · Universität zu Lübeck, 2018

[23] Ming Xie: *Fundamentals of Robotics: Linking Perception to Action* · Series in machine perception and artificial intelligence · World Scientific Pub., 2003 · ISBN: 9789812383358

[24] John XJ Zhang und Kazunori Hoshino: *Molecular Sensors and Nanodevices: Principles, Designs and Applications in Biomedical Engineering* · Academic Press, 2018

ARTIKEL AUS FACHZEITSCHRIFTEN

[25] Leonard Max Adleman: „Molecular computation of solutions to combinatorial problems“ · *Nature* Bd. 369 (1994), S. 40

[26] Ian Fuat Akyildiz, Fernando Brunetti und Cristina Blázquez: „Nanonetworks: A new communication paradigm“ · *Computer Networks* Bd. 52, Nr. 12 (2008), S. 2260–2279 · ISSN: 1389-1286

[27] Ian Fuat Akyildiz, Josep Miquel Jornet und Massimiliano Pierobon: „Nanonetworks: A New Frontier in Communications“ · *Commun. ACM* Bd. 54, Nr. 11 (Nov. 2011), S. 84–89 · ISSN: 0001-0782

[28] Ian Fuat Akyildiz, Massimiliano Pierobon, S. Balasubramaniam und Y. Koucheryavy: „The internet of Bio-Nano Things“ · *IEEE Communications Magazine* Bd. 53, Nr. 3 (März 2015), S. 32–40 · ISSN: 0163-6804

[29] Faisal Aldaye und Hanadi Sleiman: „Modular access to structurally switchable 3D discrete DNA assemblies“ · *Journal of the American Chemical Society* Bd. 129, Nr. 44 (2007), S. 13376–13377

[30] Eric Allender, David A. Mix Barrington, Tanmoy Chakraborty, Samir Datta und Sambuddha Roy: „Planar and Grid Graph Reachability Problems“ · *Theory of Computing Systems* Bd. 45, Nr. 4 (2009), S. 675–723

[31] Carme Alvarez und Raymond Greenlaw: „A compendium of problems complete for symmetric logarithmic space“ · *computational complexity* Bd. 9, Nr. 2 (2000), S. 123–145 · ISSN: 1420-8954

[32] Islamshah Amlani, Alexei Orlov, Gregory Snider, Craig Lent und Gary Bernstein: „Realization of a Functional Cell for Quantum-Dot Cellular Automaton“ · *Science* Bd. 277 (Aug. 1997), S. 928–930

[33] Ebbe Andersen, Mingdong Dong, Morten M. Nielsen, Kasper Jahn, Ramesh Subramani, Wael Mamdouh, Monika M. Golas, Bjoern Sander, Holger Stark, Cristiano L. P. Oliveira, Jan Skov Pedersen, Victoria Birkedal, Flemming Besenbacher, Kurt V. Gothelf und Jorgen Kjems: „Self-assembly of a nanoscale DNA box with a controllable lid“ · *Nature* Bd. 459, Nr. 7243 (2009), S. 73–76 · ISSN: 0028-0836 · DOI: 10.1038/nature07971 · URL: http://dx.doi.org/10.1038/nature07971

[34] Ebbe Andersen, Mingdong Dong, Morten M Nielsen, Kasper Jahn, Ramesh Subramani, Wael Mamdouh, Monika M Golas, Bjoern Sander, Holger Stark, Cristiano LP Oliveira u. a.: „Self-assembly of a nanoscale DNA box with a controllable lid“ · *Nature* Bd. 459, Nr. 7243 (2009), S. 73–76

[35] Christopher Anderson, Elizabeth J Clarke, Adam P Arkin und Christopher A Voigt: „Environmentally controlled invasion of cancer cells by engineered bacteria“ · *Journal of molecular biology* Bd. 355, Nr. 4 (2006), S. 619–627

[36] David Angeli: „A Tutorial on Chemical Reaction Network Dynamics“ · *European Journal of Control - EUR J CONTROL* Bd. 15 (Mai 2009), S. 398–406 · DOI: 10.3166/ejc.15.398-406

[37] Baris Atakan, Ozgur Akan und Sasitharan Balasubramaniam: „Body area nanonetworks with molecular communications in nanomedicine“ · *IEEE Communications Magazine* Bd. 50, Nr. 1 (Jan. 2012), S. 28–34 · ISSN: 0163-6804

[38] Sasitharan Balasubramaniam und Pietro Lio': „Multi-hop conjugation based bacteria nanonetworks“ · *IEEE Transactions on Nanobioscience* Bd. 12, Nr. 1 (2013), S. 47–59 · ISSN: 15361241 · DOI: 10.1109/TNB.2013.2239657

[39] Ray Baughman, Anvar A Zakhidov und Walt A De Heer: „Carbon nanotubes – the route toward applications“ · *science* Bd. 297, Nr. 5582 (2002), S. 787–792

[40] Steven Bellovin: „Frank Miller: Inventor of the one-time pad“ · *Cryptologia* Bd. 35, Nr. 3 (2011), S. 203–222

[41] Yaakov Benenson, Binyamin Gil, Uri Ben-Dor, Rivka Adar und Ehud Shapiro: „An autonomous molecular computer for logical control of gene expression“ · *Nature* Bd. 429, Nr. 6990 (Mai 2004), S. 423–429 · ISSN: 0028-0836

[42] Erik Benson, Abdulmelik Mohammed, Johan Gardell, Sergej Masich, Eugen Czeizler, Pekka Orponen und Björn Högberg: „DNA rendering of polyhedral meshes at the nanoscale“ · *Nature* Bd. 523, Nr. 7561 (2015), S. 441–444

[43] A. Bicen und Ian Fuat Akyildiz: „End-to-end Propagation Noise and Memory Analysis for Molecular Communication over Microfluidic Channels“ · *IEEE Transactions on Communications* Bd. 62, Nr. 7 (Juli 2014), S. 2432–2443 · DOI: 10.1109/TCOMM.2014.2323293

[44] Robert Brijder: „Computing with chemical reaction networks: a tutorial“ · *Natural Computing* Bd. 18, Nr. 1 (März 2019), S. 119–137 · ISSN: 1572-9796 · DOI: 10.1007/s11047-018-9723-9 · URL: https://doi.org/10.1007/s11047-018-9723-9

[45] Yuriy Brun: „Arithmetic computation in the tile assembly model: Addition and multiplication“ · *Theoretical computer science* Bd. 378, Nr. 1 (2007), S. 17–31

[46] Stephen Bush, Janet Paluh, Guiseppe Piro, Vijay Rao, Venkatesha Prasad und Andrew Eckford: „Defining Communication at the Bottom“ · *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* Bd. 1, Nr. 1 (März 2015), S. 90–96

[47] Jie Chao, Yunfeng Lin, Huajie Liu, Lianhui Wang und Chunhai Fan: „DNA-based plasmonic nanostructures“ · *Materials Today* Bd. 18, Nr. 6 (2015), S. 326–335

[48] Junghuei Chen und Nadrian Charles Seeman: „Synthesis from DNA of a molecule with the connectivity of a cube“ · *Nature* Bd. 350, Nr. 6319 (1991), S. 631–633

[49] Ho-Lin Chen und Ashish Goel: „Error free self-assembly using error prone tiles“ · *Proceedings of the 10th International Conference on DNA Computing, DNA'04* (2005)

[50] Qi Cheng, Gagan Aggarwal, Michael H. Goldwasser, Ming-Yang Kao, Robert Schweller und Pablo Moisset de Espanés: „Complexities for Generalized Models of Self-Assembly“ · *SIAM Journal on Computing* Bd. 34 (2005), S. 1493–1515

[51] Chiu, Andrew, Davida, George und Litow, Bruce: „Division in logspace-uniform NC1“ · *RAIRO-Theor. Inf. Appl.* Bd. 35, Nr. 3 (2001), S. 259–275

[52] Arkadiusz Chworos, Isil Severcan, Alexey Koyfman, Patrick Weinkam, Emin Oroudjev, Helen Hansma und Luc Jaeger: „Building programmable jigsaw puzzles with RNA“ · *Science* Bd. 306, Nr. 5704 (2004), S. 2068–2072

[53] Luis Cobo und Ian F Akyildiz: „Bacteria-based communication in nanonetworks“ · *Nano Communication Networks* Bd. 1, Nr. 4 (2010), S. 244–256

[54] Stephen Cook und Pierre McKenzie: „Problems complete for deterministic logarithmic space“ · *Journal of Algorithms* Bd. 8, Nr. 3 (1987), S. 385–394 · ISSN: 0196-6774

[55] Hendrik Dietz, Shawn Douglas und William Shih: „Folding DNA into twisted and curved nanoscale shapes“ · *Science* Bd. 325, Nr. 5941 (2009), S. 725–730

[56] Shawn Douglas, Ido Bachelet und George Church: „A logic-gated nanorobot for targeted transport of molecular payloads“ · *Science* Bd. 335, Nr. 6070 (2012), S. 831–834

[57] Shawn Douglas, Hendrik Dietz, Tim Liedl, Björn Högberg, Franziska Graf und William Shih: „Self-assembly of DNA into nanoscale three-dimensional shapes“ · *Nature* Bd. 459, Nr. 7245 (2009), S. 414–418

[58] Falko Dressler und Frank Kargl: „Towards Security in Nano Communication: Challenges and Opportunities“ · *Nano Communication Networks* Bd. 3, Nr. 3 (Sep. 2012), S. 151–160

[59] Masayuki Endo, Kumi Hidaka, Takayuki Kato, Keiichi Namba und Hiroshi Sugiyama: „DNA prism structures constructed by folding of multiple rectangular arms“ · *Journal of the American Chemical Society* Bd. 131, Nr. 43 (2009), S. 15570–15571

[60] Masayuki Endo, Yangyang Yang und Hiroshi Sugiyama: „DNA origami technology for biomaterials applications“ · *Biomaterials Science* Bd. 1, Nr. 4 (2013), S. 347–360

[61] Nariman Farsad, Birkan Yilmaz, Andrew Eckford, Chan-Byoung Chae und Weisi Guo: „A comprehensive survey of recent advancements in molecular communication“ · *IEEE Communications Surveys & Tutorials* Bd. 18, Nr. 3 (2016), S. 1887–1919

[62] Luca Felicetti, Mauro Femminella und GianLuca Reali: „A simulation tool for nanoscale biological networks“ · *Nano Communication Networks* Bd. 3, Nr. 1 (2012), S. 2–18 · ISSN: 1878-7789 · DOI: https://doi.org/10.1016/j.nancom.2011.09.002 · URL: http://www.sciencedirect.com/science/article/pii/S1878778911000524

[63] Luca Felicetti, Mauro Femminella, Gianluca Reali, Paolo Gresele und Marco Malvestiti: „Simulating an in vitro experiment on nanoscale communications by using BiNS2“ · *Nano Communication Networks* Bd. 4, Nr. 4 (2013), S. 172–180 · ISSN: 1878-7789 · DOI: https://doi.org/10.1016/j.nancom.2013.08.003 · URL: http://www.sciencedirect.com/science/article/pii/S1878778913000471

[64] Luca Felicetti, Mauro Femminella, Gianluca Reali, Paolo Gresele, Marco Malvestiti und John Daigle: „Modeling CD40-Based Molecular Communications in Blood Vessels“ · *IEEE Transactions on NanoBioscience*, Nr. 3 (Sep. 2014), S. 230–243 · ISSN: 1536-1241 · DOI: 10.1109/TNB.2014.2340134

[65] Richard Philips Feynman: „There's Plenty of Room at the Bottom“ · *Engineering and science* Bd. 23, Nr. 5 (Dez. 1959), S. 22–36

[66] Robert Freitas: „Current Status of Nanomedicine and Medical Nanorobotics“ · *Journal of Computational and Theoretical Nanoscience* Bd. 2, Nr. 1 (2005), S. 1–25 · ISSN: 1546-1955

[67] Tsu Ju Fu und Nadrian Charles Seeman: „DNA double-crossover molecules“ · *Biochemistry* Bd. 32, Nr. 13 (1993), S. 3211–3220

[68] Laura Galluccio, Tommaso Melodia, Sergio Palazzo und Enrico Santagati: „Medium Access Control and Rate Adaptation for Ultrasonic Intrabody Sensor Networks“ · *IEEE/ACM Transactions on Networking* Bd. 23, Nr. 4 (Aug. 2015), S. 1121–1134 · ISSN: 1063-6692 · DOI: 10.1109/TNET.2014.2316675

[69] Cody Geary, Paul Wilhelm Karl Rothemund und Ebbe Andersen: „A single-stranded architecture for cotranscriptional folding of RNA nanostructures“ · *Science* Bd. 345, Nr. 6198 (2014), S. 799–804

[70] Siavash Ghavami, Farshad Lahouti und Ali Masoudi-Nejad: „Modeling and analysis of abnormality detection in biomolecular nano-networks“ · *Nano Communication Networks* Bd. 3, Nr. 4 (2012), S. 229–241

[71] Zhen Gu, Alex Aimetti, Qun Wang, Tram Dang, Yunlong Zhang, Omid Veiseh, Hao Cheng, Robert Langer und Daniel Anderson: „Injectable nano-network for glucose-mediated insulin delivery“ · *ACS nano* Bd. 7, Nr. 5 (2013), S. 4194–4201

[72] Ertan Gul, Baris Atakan und Ozgur B. Akan: „NanoNS: A nanoscale network simulator framework for molecular communications“ · *Elsevier Nano Communication Networks* Bd. 1, Nr. 2 (Juni 2010), S. 138–156 · DOI: 10.1016/j.nancom.2010.08.003

[73] Aydin Guney, Baris Atakan und Özgür Akan: „Mobile Ad Hoc Nanonetworks with Collision-based Molecular Communication“ · *IEEE Transactions on Mobile Computing* Bd. 11, Nr. 3 (2012), S. 353–366 · DOI: 10.1109/TMC.2011.53

[74] Dongran Han, Suchetan Pal, Jeanette Nangreave, Zhengtao Deng, Yan Liu und Hao Yan: „DNA origami with complex curvatures in three-dimensional space“ · *Science* Bd. 332, Nr. 6027 (2011), S. 342–346

[75] Yudong Hao, Martin Kristiansen, Ruojie Sha, Jens Birktoft, Carina Hernandez, Chengde Mao und Nadrian Charles Seeman: „A device that operates within a self-assembled 3D DNA crystal“ · *Nature Chemistry* (2017)

[76] Robin Holliday: „Molecular aspects of genetic exchange and gene conversion“ · *Genetics* Bd. 78, Nr. 1 (1974), S. 273

[77] Fan Hong, Shuoxing Jiang, Tong Wang, Yan Liu und Hao Yan: „3D Framework DNA Origami with Layered Crossovers“ · *Angewandte Chemie* (2016)

[78] John Joseph Hopfield: „Kinetic Proofreading: A New Mechanism for Reducing Errors in Biosynthetic Processes Requiring High Specificity“ · *Proc Natl Acad Sci U S A* Bd. 71, Nr. 10 (Okt. 1974) · 4530290[pmid], S. 4135–4139 · ISSN: 0027-8424 · URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC434344/

[79] Hen-Wei Huang, Mahmut Selman Sakar, Andrew J. Petruska, Salvador Pane und Bradley J. Nelson: „Soft micromachines with programmable motility and morphology“ · *Nat Commun* Bd. 7 (Juli 2016)

[80] Neil Immerman und Susan Landau: „The Complexity of Iterated Multiplication“ · *Information and Computation* Bd. 116, Nr. 1 (1995), S. 103–116 · ISSN: 0890-5401

[81] Birgit Jenner, Johannes Köbler, Pierre McKenzie und Jacobo Torán: „Completeness results for graph isomorphism“ · *Journal of Computer and System Sciences* Bd. 66, Nr. 3 (2003), S. 549–566 · ISSN: 0022-0000

[82] Liuyi Jin, Lihua Zuo, Zhipei Yan und Radu Stoleru: „NanoCommunication-based Impermeable Region Mapping for Oil Reservoir Exploration“ (2019)

[83] Josep Miquel Jornet und Ian Fuat Akyildiz: „Graphene-based Plasmonic Nano-Antenna for Terahertz Band Communication in Nanonetworks“ · *IEEE Journal on Selected Areas in Communications* Bd. 31, Nr. 12 (Dez. 2013), S. 685–694 · DOI: 10.1109/JSAC.2013.SUP2.1213001

[84] Neville Kallenbach, Rong-Ine Ma und Nadrian Charles Seeman: „An immobile nucleic acid junction constructed from oligonucleotides“ · *Nature* Bd. 305, Nr. 5937 (1983), S. 829

[85] Yonggang Ke, Shawn Douglas, Minghui Liu, Jaswinder Sharma, Anchi Cheng, Albert Leung, Yan Liu, William Shih und Hao Yan: „Multilayer DNA origami packed on a square lattice“ · *Journal of the American Chemical Society* Bd. 131, Nr. 43 (2009), S. 15903–15908

[86] Yonggang Ke, Luvena Ong, William Shih und Peng Yin: „3D structures self-assembled from DNA bricks“ · *science* Bd. 338, Nr. 6111 (2012), S. 1177–1183

[87] Yonggang Ke, Luvena Ong, Wei Sun, Jie Song, Mingdong Dong, William Shih und Peng Yin: „DNA brick crystals with prescribed depths“ · *Nature chemistry* Bd. 6, Nr. 11 (2014), S. 994

[88] Yonggang Ke, Jaswinder Sharma, Minghui Liu, Kasper Jahn, Yan Liu und Hao Yan: „Scaffolded DNA origami of a DNA tetrahedron molecular container“ · *Nano letters* Bd. 9, Nr. 6 (2009), S. 2445–2447

[89] Donald Ervin Knuth: „Big omicron and big omega and big theta“ · *ACM Sigact News* Bd. 8, Nr. 2 (1976), S. 18–24

[90] Mehmet Şükrü Kuran, H. Birkan Yilmaz, Tuna Tugcu und Ian Fuat Akyildiz: „Interference effects on modulation techniques in diffusion based nanonetworks“ · *Nano Communication Networks* Bd. 3, Nr. 1 (2012), S. 65–73 · DOI: 10.1016/j.nancom.2012.01.005

[91] Thomas LaBean, Erik Winfree und John H Reif: „Experimental progress in computation by self-assembly of DNA tilings“ (2000)

[92] Leslie Lamport: „Time, Clocks, and the Ordering of Events in a Distributed System“ · *Commun. ACM* Bd. 21, Nr. 7 (Juli 1978), S. 558–565 · ISSN: 0001-0782

[93] Bryden Le Bailly: „Computing: Nothing more than DNA“ · *Nat Nano* (Sep. 2016) · Research Highlights · ISSN: 1748-3387 · URL: http://dx.doi.org/10.1038/nnano.2016.173

[94] Ji Youn Lee, Soo-Yong Shin, Tai Hyun Park und Byoung-Tak Zhang: „Solving traveling salesman problems with DNA molecules encoding numerical values“ · *Biosystems* Bd. 78, Nr. 1 (2004), S. 39–47 · ISSN: 0303-2647 · DOI: https://doi.org/10.1016/j.biosystems.2004.06.005 · URL: http://www.sciencedirect.com/science/article/pii/S0303264704001157

[95] Suping Li, Qiao Jiang, Shaoli Liu, Yinlong Zhang, Yanhua Tian, Chen Song, Jing Wang, Yiguo Zou, Gregory J. Anderson, Jing-Yan Han, Yung Chang, Yan Liu, Chen Zhang, Liang Chen, Guangbiao Zhou, Guangjun Nie, Hao Yan, Baoquan Ding und Yuliang Zhao: „A DNA nanorobot functions as a cancer therapeutic in response to a molecular trigger in vivo“ · *Nature Biotechnology* Bd. 36 (Feb. 2018), · URL: http://dx.doi.org/10.1038/nbt.4071

[96] Xiaojun Li, Xiaoping Yang, Jing Qi und Nadrian Charles Seeman: „Antiparallel DNA double crossover molecules as components for nanoconstruction“ · *Journal of the American Chemical Society* Bd. 118, Nr. 26 (1996), S. 6131–6140

[97] Chenxiang Lin, Yan Liu, Sherri Rinker und Hao Yan: „DNA tile based self-assembly: building complex nanoarchitectures“ · *ChemPhysChem* Bd. 7, Nr. 8 (2006), S. 1641–1647

[98] Pik Kwan Lo, Pierre Karam, Faisal Aldaye, Christopher McLaughlin, Graham Hamblin, Gonzalo Cosa und Hanadi Sleiman: „Loading and selective release of cargo in DNA nanotubes with longitudinal variation“ · *Nature chemistry* Bd. 2, Nr. 4 (2010), S. 319–328

[99] Angela Mammana, Gregory Carroll, Jetsuda Areephong und Ben Feringa: „A Chiroptical Photoswitchable DNA Complex“ · *The Journal of Physical Chemistry B* Bd. 115, Nr. 40 (2011) · PMID: 21879715, S. 11581–11587 · DOI: 10.1021/jp205893y · eprint: https://doi.org/10.1021/jp205893y · URL: https://doi.org/10.1021/jp205893y

[100] Abraham Maslow: „A theory of human motivation.“ · *Psychological review* Bd. 50, Nr. 4 (1943), S. 370

[101] Edward McCluskey Jr: „Minimization of Boolean functions“ · *Bell system technical Journal* Bd. 35, Nr. 6 (1956), S. 1417–1444

[102] Deyev Sergei Mikhailovich und E.N. Lebedenko: „Modern technologies for creating synthetic antibodies for clinical application“ · *Acta Naturae* Bd. 1, Nr. 1 (1) (2009)

[103] George Edward Moore: „Cramming more components onto integrated circuits“ · *Electronics* Bd. 38, Nr. 8 (1965), S. 114–117

[104] Kary Mullis: „The polymerase chain reaction (Nobel lecture)“ · *Angewandte Chemie International Edition in English* Bd. 33, Nr. 12 (1994), S. 1209–1213

[105] Tadashi Nakano: „Molecular Communication: A 10 Year Retrospective“ · *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* Bd. 3, Nr. 2 (Juni 2017), S. 71–78 · DOI: 10.1109/TMBMC.2017.2750148

[106] Alec Nielsen, Bryan S. Der, Jonghyeon Shin, Prashant Vaidyanathan, Vanya Paralanov, Elizabeth A. Strychalski, David Ross, Douglas Densmore und Christopher A. Voigt: „Genetic circuit design automation“ · *Science* Bd. 352, Nr. 6281 (2016)

[107] Dmytro Nykypanchuk, Mathew Maye, Daniel Van Der Lelie und Oleg Gang: „DNA-guided crystallization of colloidal nanoparticles“ · *Nature* Bd. 451, Nr. 7178 (2008), S. 549–552

[108] Norman Packard und Stephen Wolfram: „Two-dimensional cellular automata“ · *Journal of Statistical Physics* Bd. 38, Nr. 5 (März 1985), S. 901–946 · ISSN: 1572-9613 · DOI: 10.1007/BF01010423 · URL: https://doi.org/10.1007/BF01010423

[109] Sung Yong Park, Abigail Lytton-Jean, Byeongdu Lee, Steven Weigand, George Schatz und Chad Mirkin: „DNA-programmable nanoparticle crystallization“ · *Nature* Bd. 451, Nr. 7178 (2008), S. 553–556

[110] Matthew Patitz: „An introduction to tile-based self-assembly and a survey of recent results“ · *Natural Computing* Bd. 13, Nr. 2 (2014), S. 195–224 · ISSN: 1572-9796 · DOI: 10.1007/s11047-013-9379-4 · URL: http://dx.doi.org/10.1007/s11047-013-9379-4

[111] Matthew Patitz: „Simulation of Self-Assembly in the Abstract Tile Assembly Model with ISUTAS“ · *CoRR* Bd. abs/1101.5151 (2009)

[112] Dieter Perl, Uwe Mueller, Udo Heinemann und Franz Schmid: „Two exposed amino acid residues confer thermostability on a cold shock protein“ · *Nature Structural & Molecular Biology* Bd. 7, Nr. 5 (2000), S. 380

[113] William Wesley Peterson und Daniel Brown: „Cyclic codes for error detection“ · *Proceedings of the IRE* Bd. 49, Nr. 1 (1961), S. 228–235

[114] M. Pierobon und Ian Fuat Akyildiz: „A physical end-to-end model for molecular communication in nanonetworks“ · *IEEE Journal on Selected Areas in Communications* Bd. 28, Nr. 4 (Mai 2010), S. 602–611 · ISSN: 0733-8716

[115] Willard Quine: „The problem of simplifying truth functions“ · *The American mathematical monthly* Bd. 59, Nr. 8 (1952), S. 521–531

[116] Vincent Rijmen und Joan Daemen: „Advanced encryption standard“ · *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology* (2001), S. 19–22

[117] Paul Wilhelm Karl Rothemund: „Folding DNA to create nanoscale shapes and patterns“ · *Nature* Bd. 440, Nr. 7082 (März 2006), S. 297–302 · ISSN: 0028-0836 · DOI: 10.1038/nature04586 · URL: http://dx.doi.org/10.1038/nature04586

[118] Paul Wilhelm Karl Rothemund, Nick Papadakis und Erik Winfree: „Algorithmic Self-Assembly of DNA Sierpinski Triangles“ · *PLOS Biology* Bd. 2, Nr. 12 (Dez. 2004) · DOI: 10.1371/journal.pbio.0020424 · URL: https://doi.org/10.1371/journal.pbio.0020424

[119] Rudnitzki, Feineis, Rahmanzadeh, Endl, Lutz, Groll und Huettmann: „siRNA release from gold nanoparticles by nanosecond pulsed laser irradiation and analysis of the involved temperature increase“ · *Journal of BIOPHOTONICS* () · DOI: 10.1002/jbio.201700329

[120] Walter John Savitch: „Relationships between nondeterministic and deterministic tape complexities“ · *Journal of Computer and System Sciences* Bd. 4, Nr. 2 (1970), S. 177–192 · ISSN: 0022-0000

[121] Nadrian Charles Seeman: „DNA in a material world“ · *Nature* Bd. 421, Nr. 6921 (2003), S. 427

[122] Nadrian Charles Seeman: „Nucleic acid junctions and lattices“ · *Journal of Theoretical Biology* Bd. 99, Nr. 2 (1982), S. 237–247 · ISSN: 0022-5193 · DOI: https://doi.org/10.1016/0022-5193(82)90002-9 · URL: http://www.sciencedirect.com/science/article/pii/0022519382900029

[123] Nadrian Charles Seeman und Hanadi Sleiman: „DNA nanotechnology“ · *Nature Reviews Materials* Bd. 3, Nr. 1 (2018), S. 17068

[124] Sunil Singhal, Shuming Nie und May Wang: „Nanotechnology applications in surgical oncology“ · *Annual review of medicine* Bd. 61 (2010), S. 359–373

[125] Mark Staples, Karen Daniel, Michael Cima und Robert Langer: „Application of Micro- and Nano-Electromechanical Devices to Drug Delivery“ · *Pharmaceutical Research* Bd. 23, Nr. 5 (2006), S. 847–863 · ISSN: 1573-904X

[126] Alan Mathison Turing: „On computable numbers, with an application to the Entscheidungsproblem“ · *Proceedings of the London mathematical society* Bd. 2, Nr. 1 (1937), S. 230–265

[127] Frank Walsh und Sasitharan Balasubramaniam: „Reliability and delay analysis of multihop virus-based nanonetworks“ · *IEEE Transactions on Nanotechnology* Bd. 12, Nr. 5 (2013), S. 674–684 · ISSN: 1536125X · DOI: 10.1109/TNANO.2013.2268389

[128] Joseph Wang und Wei Gao: „Nano/Microscale Motors: Biomedical Opportunities and Challenges“ · *ACS Nano* Bd. 6, Nr. 7 (2012) · PMID: 22770233, S. 5745–5751 · DOI: 10.1021/nn3028997 · eprint: http://dx.doi.org/10.1021/nn3028997 · URL: http://dx.doi.org/10.1021/nn3028997

[129] Silja Wang, Gereon Huttmann, Zhenxi Zhang, Alfred Vogel, Reginald Birngruber, Shifalika Tangutoori, Tayyaba Hasan und Ramtin Rahmanzadeh: „Light-Controlled Delivery of Monoclonal Antibodies for Targeted Photoinactivation of Ki-67“ · *Mol Pharm* (2015) · ISSN: 1543-8384 · DOI: 10.1021/acs.molpharmaceut.5b00260

[130] Xiao Wang, Ruojie Sha, Martin Kristiansen, Carina Hernandez, Yudong Hao, Chengde Mao, James Canary und Nadrian Charles Seeman: „An Organic Semiconductor Organized into 3D DNA Arrays by “Bottom-up” Rational Design“ · *Angewandte Chemie* Bd. 129, Nr. 23 (2017), S. 6545–6548

[131] Xudong Wang: „Piezoelectric nanogenerators—Harvesting ambient mechanical energy at the nanometer scale“ · *Nano Energy* Bd. 1, Nr. 1 (2012) · ISSN: 2211-2855

[132] James Dewey Watson und Francis Harray Crick: „A Structure for Deoxyribose Nucleic Acid“ · *Nature* Bd. 171 (Apr. 1953), S. 737–738 · URL: http://www.nature.com/nature/dna50/watsoncrick.pdf

[133] Bryan Wei, Mingjie Dai und Peng Yin: „Complex shapes self-assembled from single-stranded DNA tiles“ · *Nature* Bd. 485, Nr. 7400 (2012), S. 623–626

[134] Erik Winfree und Renat Bekbolatov: „Proofreading Tile Sets: Error Correction for Algorithmic Self-Assembly“ · Lecture Notes in Computer Science Bd. 2943 (2003), S. 126–144

[135] Erik Winfree, Furong Liu, Lisa Wenzler und Nadrian Charles Seeman: „Design and self-assembly of two-dimensional DNA crystals“ · *Nature* Bd. 394, Nr. 6693 (1998), S. 539–544

[136] Stephen Wolfram: „Computation theory of cellular automata“ · *Communications in Mathematical Physics* Bd. 96, Nr. 1 (März 1984), S. 15–57 · ISSN: 1432-0916 · DOI: 10.1007/BF01217347 · URL: http://dx.doi.org/10.1007/BF01217347

[137] Cuiping Yao, Florian Rudnitzki, Gereon Hüttmann, Zhenxi Zhang und Ramtin Rahmanzadeh: „Important factors for cell-membrane permeabilization by gold nanoparticles activated by nanosecond-laser irradiation“ · *International Journal of Nanomedicine* Bd. Volume 12 (2017), S. 5659–5672 · DOI: 10.2147/IJN.S140620

[138] Chuan Zhang: „DNA self-assembly: from 2D to 3D“ · *Faraday Discuss (143)* (2009), S. 221–233

[139] Jianping Zheng, Jens Birktoft, Yi Chen, Tong Wang, Ruojie Sha, Pamela E Constantinou, Stephan L Ginell, Chengde Mao und Nadrian Charles Seeman: „From molecular to macroscopic via the rational design of a self-assembled 3D DNA crystal“ · *Nature* Bd. 461, Nr. 7260 (2009), S. 74

[140] Victor Zhirnov und Daniel Herr: „New Frontiers: Self-Assembly and Nanoelectronics“ · *Computer* Bd. 34, Nr. 1 (Jan. 2001), S. 34–43 · ISSN: 0018-9162 · DOI: 10.1109/2.895116 · URL: http://dx.doi.org/10.1109/2.895116

KONFERENZBEITRÄGE

[141] Apostolos Almpanis, Christophe Corre und Adam Noel: „Agent Based Modeling of the Rhizobiome with Molecular Communication and Game Theory“ · *Proceedings of the Sixth Annual ACM International Conference on Nanoscale Computing and Communication, NANOCOM 2019, Dublin, Ireland, September 25-27, 2019.* · 2019, 20:1–20:7 · DOI: 10.1145/3345312.3345476 · URL: https://doi.org/10.1145/3345312.3345476

[142] Florian Büther, Immo Traupe und Sebastian Ebers: „Hop Count Routing: A Routing Algorithm for Resource Constrained, Identity-Free Medical Nanonetworks“ · *5th ACM International Conference on Nanoscale Computing and Communication 2018 (ACM NanoCom'18)* · Reykjavik, Iceland, Sep. 2018 · ISBN: 978-1-4503-5711-1/18/09 · DOI: 10.1145/3233188.3233193

[143] Vincenzo Catania, Andrea Mineo, Salvatore Monteleone und Davide Patti: „A Low-resource and Scalable Strategy for Segment Partitioning of Many-core Nano Networks“ · *Proceedings of International Workshop on Manycore Embedded Systems* · MES '14 · Minneapolis, MN, USA: ACM, 2014, 17:17–17:24 · ISBN: 978-1-4503-2822-7 · DOI: 10.1145/2613908.2613915 · URL: http://doi.acm.org/10.1145/2613908.2613915

[144] Ho-Lin Chen und Ashish Goel: „Error Free Self-Assembly using Error Prone Tiles“ · *Proceedings of the 10th International Meeting on DNA Based Computers* · 2004, S. 274–283

[145] Kenichi Fujibayashi und Satoshi Murata: „Thermodynamic Simulations of DNA Tile Self-assembly“ · *Proceedings of the 2nd International Conference on Nano-Networks* · Nano-Net '07 · Catania, Italy: ICST (Institute for Computer Sciences, Social-Informatics und Telecommunications Engineering), 2007, 20:1–20:5 · ISBN: 978-963-9799-10-3 · URL: http://dl.acm.org/citation.cfm?id=1459290.1459317

[146] Cyril Gavoille, Christian Glacet, Nicolas Hanusse und David Ilcinkas: „On the communication complexity of distributed name-independent routing schemes“ · *International Symposium on Distributed Computing* · Springer · 2013, S. 418–432

[147] Nor Zaidi Haron und Said Hamdioui: „Why is CMOS scaling coming to an END?“ · *2008 3rd International Design and Test Workshop* · Dez. 2008, S. 98–103 · DOI: 10.1109/IDT.2008.4802475

[148] Yubing Jian, Bhuvana Krishnaswamy, Caitlin M. Austin, Ozan Bicen, Jorge Perdomo, Sagar Patel, Ian Fuat Akyildiz, Craig Forest und Raghupathy Sivakumar: „nanoNS3: Simulating Bacterial Molecular Communication Based Nanonetworks in Network Simulator 3“ · *Proceedings of the 3rd ACM International Conference on Nanoscale Computing and Communication* · NANOCOM'16 · New York, NY, USA: ACM, Sep. 2016, 17:1–17:7 · ISBN: 978-1-4503-4061-8 · DOI: 10.1145/2967446.2967464 · URL: http://doi.acm.org/10.1145/2967446.2967464

[149] Edmund Landau: „Handbuch der Lehre von der Verteilung der Primzahlen“ · Bd. 01 · Leipzig B.G. Teubner, 1909

[150] James Lathrop, Jack Lutz und Scott Summers: „Strict Self-assembly of Discrete Sierpinski Triangles“ · *Computation and Logic in the Real World: Third Conference on Computability in Europe, CiE 2007, Siena, Italy, June 18-23, 2007. Proceedings* · hrsg. von S. Barry Cooper, Benedikt Löwe und Andrea Sorbi · DOI: 10.1007/978-3-540-73001-9_47 · Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, S. 455–464 · ISBN: 978-3-540-73001-9 · URL: http://dx.doi.org/10.1007/978-3-540-73001-9_47

[151] Anthony Liekens und Chrisantha Fernando: „Turing Complete Catalytic Particle Computers“ · *Advances in Artificial Life* · hrsg. von Fernando Almeida e Costa, Luis Mateus Rocha, Ernesto Costa, Inman Harvey und Antonio Coutinho · Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, S. 1202–1211 · ISBN: 978-3-540-74913-4

[152] Xiaojun Ma, Masoud Hashempour, Lei Wang und Fabrizio Lombardi: „Manufacturing Yield of QCA Circuits by Synthesized DNA Self-assembled Templates“ · *Proceedings of the 20th Symposium on Great Lakes Symposium on VLSI* · GLSVLSI '10 · Providence, Rhode Island, USA: ACM, 2010, S. 275–280 · ISBN: 978-1-4503-0012-4 · DOI: 10.1145/1785481.1785546 · URL: http://doi.acm.org/10.1145/1785481.1785546

[153] Kao Ming-Yang und Vijay Ramachandran: „DNA Self-Assembly For Constructing 3D Boxes“ · *Algorithms and Computation: 12th International Symposium, ISAAC 2001 Christchurch, New Zealand, December 19–21, 2001 Proceedings* · hrsg. von Peter Eades und Tadao Takaoka · Berlin,

Heidelberg: Springer Berlin Heidelberg, 2001, S. 429–441 · ISBN: 978-3-540-45678-0 · URL: https://doi.org/10.1007/3-540-45678-3_37

[154] Seungbin Moon und Soon-Geul Lee: „Revision of Vocabulary Standard for Robots and Robotic Devices in ISO" · *15th International Conference on Climbing and Walking Robots* · World Scientific, 2012, S. 849–854

[155] Tadashi Nakano, Kazufumi Hosoda, Yutaka Nakamura und Kojiro Ishii: „A Biologically-inspired Intrabody Nanonetwork: Design Considerations" · *8th International Conference on Body Area Networks* · BodyNets '13 · Boston, Massachusetts: ICST, 2013, S. 484–487 · ISBN: 978-1-936968-89-3

[156] Giuseppe Piro, Luigi Alfredo Grieco, Gennaro Boggia und Pietro Camarda: „Nano-Sim: Simulating Electromagnetic-based Nanonetworks in the Network Simulator 3" · *Proceedings of the 6th International ICST Conference on Simulation Tools and Techniques* · SimuTools '13 · Cannes, France: ICST (Institute for Computer Sciences, Social-Informatics und Telecommunications Engineering), März 2013, S. 203–210 · ISBN: 978-1-4503-2464-9 · DOI: 10.4108/icst.simutools.2013.251699 · URL: http://dl.acm.org/citation.cfm?id=2512734.2512762

[157] Paul Wilhelm Karl Rothemund und Erik Winfree: „The Program-size Complexity of Self-assembled Squares (Extended Abstract)" · *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing* · STOC '00 · New York, NY: ACM, 2000, S. 459–468 · ISBN: 1-58113-184-4 · DOI: 10.1145/335305.335358 · URL: http://doi.acm.org/10.1145/335305.335358

[158] Marc Stelzner, Falko Dressler und Stefan Fischer: „Function Centric Networking: An Approach for Addressing in In-Body Nano Networks" · *3rd ACM International Conference on Nanoscale Computing and Communication* · NANOCOM'16 · New York, NY, USA: ACM, Sep. 2016 · ISBN: 978-1-4503-4061-8

[159] Abdelmajid Taibi, Antoine Durant, Valeria Loscri, Anna Vegni und Luigi Spada: „Controlling Light by Curvilinear MetaSurfaces" · 2019

[160] Ageliki Tsioliaridou, Christos Liaskos, Sotiris Ioannidis und Andreas Pitsillides: „CORONA: A Coordinate and Routing System for Nanonetworks" · *Second Annual International Conference on Nanoscale Computing and Communication* · NANOCOM' 15 · Boston, MA, USA: ACM, 2015, 18:1–18:6 · ISBN: 978-1-4503-3674-1

[161] Vamsi Vankamamidi, Marco Ottavi und Fabrizi Lombardi: „Tile-based Design of a Serial Memory in QCA" · *15th ACM Great Lakes Symposium on VLSI* · GLSVLSI '05 · Chicago, Illinois, USA: ACM, 2005, S. 201–206 · ISBN: 1-59593-057-4

[162] Avi Wigderson: „The complexity of graph connectivity" · *International Symposium on Mathematical Foundations of Computer Science* · Springer · 1992, S. 112–132

[163] Erik Winfree und Renat Bekbolatov: „Proofreading Tile Sets: Error Correction for Algorithmic Self-Assembly“ · *DNA* · Bd. 2943 · Lecture Notes in Computer Science · Springer, 2003, S. 126–144

[164] Hang Yu, Bryan Ng und Winston K. G. Seah: „Forwarding Schemes for EM-based Wireless Nanosensor Networks in the Terahertz Band“ · *Second Annual Int. Conf. on Nanoscale Computing and Communication* · NANOCOM' 15 · Boston, MA, USA: ACM, 2015, 17:1–17:6 · ISBN: 978-1-4503-3674-1

SONSTIGE

[165] Paolo Amato, Massimo Masserini, Giancarlo Mauri und Gianfranco Cerofolini: „Early-Stage Diagnosis of Endogenous Diseases by Swarms of Nanobots: An Applicative Scenario“ · *Swarm Intelligence: 7th International Conference, ANTS 2010, Brussels, Belgium, September 8-10, 2010. Proceedings* · hrsg. von Marco Dorigo, Mauro Birattari, Gianni A. Di Caro, René Doursat, Andries P. Engelbrecht, Dario Floreano, Luca Maria Gambardella, Roderich Groß, Erol Şahin, Hiroki Sayama und Thomas Stützle · Berlin, Heidelberg: Springer, 2010, S. 408–415 · ISBN: 978-3-642-15461-4

[166] Florent Becker, Éric Rémila und Nicolas Schabanel: „Time Optimal Self-assembly for 2D and 3D Shapes: The Case of Squares and Cubes“ · *DNA Computing: 14th International Meeting on DNA Computing, DNA 14, Prague, Czech Republic, June 2-9, 2008. Revised Selected Papers* · hrsg. von Ashish Goel, Friedrich C. Simmel und Petr Sosík · DOI: 10.1007/978-3-642-03076-5_12 · Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, S. 144–155 · ISBN: 978-3-642-03076-5 · URL: http://dx.doi.org/10.1007/978-3-642-03076-5_12

[167] LLC Clever Prototypes: *Storyboard Creator | Comic Strip Maker | Online Storyboarding* · 28. Juni 2019 · URL: https://www.storyboardthat.com/storyboard-creator

[168] Wikimedia Commons: *File:Feste Struktur Holliday (dt.).svg — Wikimedia Commons, the free media repository* · [Online; accessed 20-May-2019] · 2019 · URL: https://commons.wikimedia.org/w/index.php?title=File:%D0%9D%D0%B5%D0%BF%D0%BE%D0%B4%D0%B2%D0%B8%D0%B6%D0%BD%D0%B0%D1%8F_%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%A5%D0%BE%D0%BB%D0%BB%D0%B8%D0%B4%D0%B5%D1%8F_(%D0%B0%D0%BD%D0%B3%D0%BB.).svg&oldid=346086830

[169] Wikimedia Commons: *File:Holliday junction coloured.png — Wikimedia Commons, the free media repository* · [Online; accessed 17-May-2019] · 2019 · URL: https://commons.wikimedia.org/w/index.php?title=File:Holliday_junction_coloured.png&oldid=346086825

[170] Wikimedia Commons: *File:Mao-DX-schematic-2.svg — Wikimedia Commons, the free media repository* · [Online; accessed 17-May-2019] · 2019 · URL: https://commons.wikimedia.org/w/index.php?title=File:Mao-DX-schematic-2.svg&oldid=346087234

[171] *DNA Lab: Xgrow Tile Assembly Simulator* · 27. Feb. 2014 · URL: http://www.dna.caltech.edu/Xgrow/ *(zuletzt abgerufen am 27. 05. 2019)*

[172] Shahram Mohrehkesh, Michele Weigle und Sajal Das: „Energy Harvesting in Nanonetworks“ · *Modeling, Methodologies and Tools for Molecular and Nano-scale Communications: Modeling, Methodologies and Tools* · Springer International Publishing, 2017, S. 319–347 · ISBN: 978-3-319-50688-3

[173] Maria Staiano, Angela Pennacchio, Antonio Varriale, Alessandro Capo, Adelia Majoli, Clotilde Capacchione und Sabato D'Auria: „Enzymes as sensors“ · *Methods in enzymology* · Bd. 589 · Elsevier, 2017, S. 115–131

[174] *The Biggest Industries In The United States - WorldAtlas.com* · 12. Juni 2019 · URL: https://www.worldatlas.com/articles/which-are-the-biggest-industries-in-the-united-states.html *(zuletzt abgerufen am 12. 06. 2019)*

[175] *The Xgrow Simulator* · http://www.dna.caltech.edu/Xgrow/ · Accessed: 2018-05-04

[176] Heribert Vollmer: „The NC Hierarchy“ · *Introduction to Circuit Complexity: A Uniform Approach* · Berlin, Heidelberg: Springer, 1999, S. 107–171 · ISBN: 978-3-662-03927-4

[177] Walus Group: *QCADesigner* · 2009

[178] Hao Wang: „Dominoes and the Aea Case of the Decision Problem“ · *Computation, Logic, Philosophy: A Collection of Essays* · Dordrecht: Springer Netherlands, 1990, S. 218–245 · ISBN: 978-94-009-2356-0 · DOI: 10.1007/978-94-009-2356-0_11 · URL: https://doi.org/10.1007/978-94-009-2356-0_11

[179] Christof Windeck: *AMD Ryzen 3000: Am 7.7. kommt die 7-nm-CPU | heise online* · 5. Juni 2019 · URL: https://www.heise.de/newsticker/meldung/AMD-Ryzen-3000-Am-7-7-kommt-die-7-nm-CPU-4312622.html *(zuletzt abgerufen am 05. 06. 2019)*

[180] *XTile index* · 27. Mai 2019 · URL: http://www.guptalab.org/xtile/ *(zuletzt abgerufen am 27. 05. 2019)*

DANKSAGUNG

An dieser Stelle möchte ich einigen Personen danken, welche aktiv oder passiv zum Gelingen dieser Arbeit beigetragen haben – zum Teil könnte es den besagten Personen nicht einmal bewusst sein. Folgenden Personen gilt mein Dank:

Ich Danke meinen Eltern, die mir stets den nötigen Freiraum gelassen haben, um zu tun, was ich wollte. Sie brachten mir so viel Selbstständigkeit bei, wie nötig war, dass ich eigenständig Fortschritte machen und einen gewissen Stolz auf meine eigene Leistung entwickeln konnte – dieser ist die Basis meines Ehrgeizes und meines Antriebs.

Ich danke meiner Großmutter dafür, dass sie eine herzensgute Person ist.

Ich danke meinen einstigen gymnasialen Lehrkräften Peter Koerting und Wolfgang Malm, welche mich ein inniges Bedürfnis nach formaler Korrektheit und eine Zuneigung zu den Naturwissenschaften lehrten.

Ich danke Rüdiger Reischuk dafür, dass er sich die Zeit und Geduld genommen hat, mein im Studium angeeignetes Wissen auf die Probe zu stellen.

Ich danke Bennet Gerlach dafür, dass er mir praktisch täglich für Diskussion und Reflexion zur Verfügung stand und damit viele Lücken in meinem Schulwissen schloss, welche sonst heute noch klaffen würden. Ohne ihn wäre mein Modell der Welt fundamental anders – und ich glaube schlechter.

Ich danke meiner Freundin Christin Grill, welche das ausführlichste Lektorat aller Zeiten walten ließ. Ich danke ihr ebenfalls, dass sie mir jederzeit mit Diskussion, Rat und Tat zur Seite stand.

Weiterhin danke ich Regine Wendt, welche die letzten verbliebenen Fehler fand und alle biologischen Inhalte auf Korrektheit geprüft hat.

Ich danke meinen Brüdern Finn und Frederic für die freizeitlichen Aktivitäten, wenn die Last von Studium und Dissertation erdrückend wurde.

Und nicht zuletzt danke ich meinem Doktorvater Stefan Fischer für die langjährige Betreuung und die Einführung in die akademische Welt mit all ihren Feinheiten sowie die finanzielle Unterstützung, welche mir an seinem Institut zugutekam.

Der Rest meines Dankes gilt allen Mitgliedern der Nanogruppe und des Institutes für Telematik, welche über die letzten Jahre einen Teil meines Lebensweges mit mir gegangen sind und mich stets unterstützt haben, wenn ich darum bat.

LEBENSLAUF DES AUTORS

PERSÖNLICHES

Name:	Florian-Lennert Adrian Lau
Geburtsdatum:	21. März 1990
Geburtsort:	Lübeck

BERUFLICHE STATIONEN

HOCHSCHULE

seit Juni 2016	Wissenschaftlicher Mitarbeiter *Universität zu Lübeck – Institut für Telematik*
2015/16	Hilfswissenschaftler *Universität zu Lübeck – Institut für Telematik*
2013	Studentische Hilfskraft *Universität zu Lübeck – Institut für Softwaretechnik und Programmiersprachen*

INDUSTRIE

2014 – 2015	Einjährige Fallstudie *Capgemini GmbH*

AUSBILDUNG

HOCHSCHULE

2013 – 2016	Master of Science Informatik *Universität zu Lübeck*
2010 – 2013	Bachelor of Science Informatik *Universität zu Lübeck*

SCHULE

2000 – 2009	Abitur *Gymnasium am Mühlenberg, Bad Schwartau*