

1 Einleitung

Übersicht

1.1	Spielen	2
1.2	Wählen	4
1.3	Teilen	7
1.4	Was ist Computational Social Choice?	8
1.5	Ein Exkurs in die Komplexitätstheorie	9

Spielen, Wählen und Teilen sind drei alltägliche Tätigkeiten unseres Lebens. Die Spieler, Wähler oder Teiler haben dabei ihre persönlichen Gewinnaussichten, Vorlieben, Meinungen oder Bewertungen im Blick und folgen daher individuellen Strategien. Zwar sind sie in erster Linie an ihrem eigenen Vorteil interessiert, doch in der Interaktion aller Akteure ergibt sich aus diesen Einzelinteressen und -strategien insgesamt ein Spielausgang, eine kollektive Entscheidung, ein gemeinsames Urteil oder eine Aufteilung von Gütern. Am Ende gibt es Gewinner und Verlierer.

Die individuelle Gewinnmaximierung ist jedoch nur ein Ziel. Kann man Mechanismen finden, die den *sozialen* oder *gesellschaftlichen* Nutzen erhöhen und somit allen dienen und nicht nur Einzelnen? Zum Beispiel damit beim Wochenendausflug ein Reiseziel gewählt werden kann, mit dem alle – sowohl die Eltern als auch die Kinder – zufrieden sind. Oder damit drei Geschwister, die sich bei einem Gesellschaftsspiel vergnügen, ihre Strategien so optimal wählen können, dass keines von ihnen seinen Gewinn durch einen Strategiewechsel erhöhen könnte, ohne gleichzeitig den Gewinn eines Mitspielers zu verringern. Oder damit es ihnen anschließend gelingt, einen Kuchen, dessen einzelne Stücke sie alle unterschiedlich bewerten, so aufzuteilen, dass kein Neid um die erhaltenen Portionen entsteht. Für alle nützlich ist auch die „Strategiesicherheit“ der verwendeten Verfahren. Kann man verhindern, dass sich jemand durch „unehrliche“ Strategien einen unlauteren Vorteil verschafft?

Dieses Buch führt in das junge, interdisziplinäre Gebiet *Computational Social Choice* ein, das sich seit einigen Jahren rasant an der Schnittstelle von Politik-, Sozial- und Wirtschaftswissenschaften einerseits und Informatik andererseits entwickelt und eng verwandt mit der algorithmischen Spieltheorie ist, die hier eben-

falls behandelt wird. Jedes der drei Themengebiete – Spielen, Wählen, Teilen – wird in zwei Kapiteln präsentiert, einem längeren und einem kürzeren. Während die längeren Kapitel (nämlich die Kapitel 2, 4 und 6) mit vielen Details, Beispielen und Abbildungen recht ausführlich in das jeweilige Thema einführen, sind die kürzeren (nämlich die Kapitel 3, 5 und 7) eher dazu gedacht, die thematische Breite des Gebiets *Computational Social Choice* zu umreißen. In diesen kürzeren Kapiteln werden wir uns demgemäß nur auf einige wesentliche Begriffe beschränken.

Ausgehend von den traditionellen Gebieten der klassischen Spieltheorie und der klassischen Social-Choice-Theorie werden insbesondere die algorithmischen Aspekte der auftretenden Probleme in den Vordergrund gestellt. Die nötigen grundlegenden Konzepte der Komplexitätstheorie und Algorithmik sowie der Aussagenlogik werden kurz und prägnant in Abschnitt 1.5 bereitgestellt. Elementare Grundlagen aus anderen Gebieten der Mathematik (z. B. Wahrscheinlichkeitstheorie, Topologie und Graphentheorie) werden jeweils dann präsentiert, wenn sie gebraucht werden, so knapp und informal wie möglich und mit so vielen Details wie nötig. Zum leichteren Verständnis werden alle wesentlichen Begriffe anhand von Beispielen und mit vielen Abbildungen illustriert. Auch werden manche Situationen zur besseren Motivation durch kleine Geschichten aus dem Alltag von Anna, Belle, Chris, David, Edgar, Felix, Georg, Helena und anderen veranschaulicht.

1.1 Spielen

Smith und Wesson, zwei Bankräuber, sind verhaftet worden. Da die Beweise für ihre Tat recht dünn sind, bietet man ihnen einen Handel an:

- Wenn einer von ihnen gesteht, kommt er auf Bewährung frei, sofern der andere weiter schweigt, und dieser kommt dann zehn Jahre hinter Gitter;
- gestehen beide, bekommen sie beide vier Jahre Knast;
- schweigen aber beide weiter beharrlich, kann man sie jeweils nur für zwei Jahre einbuchen.

Leider können sie sich nicht absprechen. Wie soll sich Smith verhalten, um mit einer möglichst geringen Gefängnisstrafe davonzukommen, deren Länge allerdings nicht nur von seiner, sondern auch von Wessons Entscheidung abhängt? Und wie soll Wesson sich entscheiden, dessen Strafe ebenfalls auch vom Verhalten des anderen abhängig ist? Das ist das Dilemma der beiden Gefangenen!

Georg und Helena möchten ihren ersten Hochzeitstag gemeinsam verbringen und irgendetwas Schönes machen. Georg möchte zusammen mit seiner Frau ein spannendes Fußballspiel ansehen. Helena dagegen würde lieber gemeinsam mit ihrem Mann in ein Konzert gehen. Sollten sie ihre unterschiedlichen Wünsche durchsetzen oder doch nachgeben? Wenn keiner der beiden nachgibt, werden sie ihren ersten Hochzeitstag nicht gemeinsam verbringen! Diese „Schlacht der Geschlechter“ hat wohl jedes Paar schon einmal geschlagen.

Solche Situationen, bei denen mehrere Akteure interagieren und Entscheidungen treffen müssen, wobei ihr Gewinn auch von den Entscheidungen der anderen Akteure abhängt, lassen sich durch strategische Spiele beschreiben. Borel (1921) und von Neumann (1928) verfassten bereits vor fast einem Jahrhundert die ersten mathematischen Abhandlungen zur Spieltheorie. Die Fundamente dieser Theorie als ein eigenständiges Gebiet legten von Neumann und Morgenstern (1944) etwa zwanzig Jahre später mit ihrem wegweisenden Werk. Seither hat sich die Spieltheorie zu einer reichen und zentralen Disziplin innerhalb der Wirtschaftswissenschaften entwickelt und eine Reihe von Nobelpreisträgern, wie John Forbes Nash, und viele grandiose Erkenntnisse hervorgebracht. Grundlegend unterscheidet man in dieser Theorie kooperative und nichtkooperative Spiele.

In der kooperativen Spieltheorie, die in Kapitel 3 vorgestellt wird, geht es u. a. um die Bildung von Koalitionen von Spielern, die zusammenarbeiten, um gemeinsame Ziele zu erreichen. Durch Kooperation kann der Einzelne seinen Gewinn möglicherweise erhöhen. Ob sich ein Spieler einer Koalition anschließt oder von ihr abfällt, entscheidet er danach, ob er selbst davon profitiert oder nicht.

Ein ganz zentrales Konzept ist dabei die Stabilität eines kooperativen Spiels. Hat ein Spieler einen Anreiz, von der großen Koalition (der Menge aller beteiligten Spieler) abzufallen, so ist das Spiel instabil und zerfällt in mehrere Koalitionen, die miteinander konkurrieren. In diesem Kapitel werden verschiedene Begriffe vorgestellt, die die Stabilität eines kooperativen Spiels in unterschiedlicher Weise erfassen. Auch kann in einem solchen Spiel der Einfluss – die Macht – eines Spielers in verschiedener Weise gemessen werden. Grob gesprochen ergibt sich der Machtindex eines Spielers daraus, wie oft seine Zugehörigkeit zu einer der möglichen Koalitionen entscheidend für deren Sieg ist. Stabilitätskonzepte und Machtindizes in kooperativen Spielen werden auch hinsichtlich ihrer algorithmischen Eigenschaften untersucht.

Die nichtkooperative Spieltheorie, der wir uns in Kapitel 2 zuwenden, befasst sich dagegen mit Spielen, bei denen die Spieler als Einzelkämpfer gegeneinander antreten, um ihren eigenen Gewinn zu maximieren. Kombinatorische Spiele wie Schach und Go, aber auch Kartenspiele wie Poker, bei denen verdeckte Information, der Zufall und die Psychologie des Bluffens eine Rolle spielen, gehören in diese Kategorie. Es lassen sich aber nicht nur Gesellschafts- oder Glücksspiele, sondern auch marktstrategische oder gar globalstrategische Konkurrenzsituationen zwischen Unternehmen oder Staaten durch solche Spiele ausdrücken.

Auch bei nichtkooperativen Spielen spielt der Begriff der Stabilität eine wichtige Rolle, um den Ausgang eines solchen Spiels vorherzusagen. Können die einzelnen Spieler ihre individuellen Strategien so wählen, dass alle im Gleichgewicht sind und keiner von seiner Strategie abweichen möchte (sofern alle anderen bei ihren Strategien bleiben)? Diese Frage war beispielsweise während des Kalten Krieges zwischen dem Ostblock und dem westlichen NATO-Bündnis von enormer Wichtigkeit für die ganze Menschheit. Auch in nichtkooperativen Spielen betrachten wir die algorithmischen Eigenschaften von Lösungskonzepten und suchen insbesonde-

re eine Antwort auf die Frage: Wie schwer ist es, solche Gleichgewichtsstrategien zu finden?

1.2 Wählen

Anna, Belle und Chris treffen sich zu einem gemeinsamen Spielabend. Doch zunächst ist zu klären, welches Spiel sie überhaupt spielen wollen. Zur Wahl stehen Schach, Poker und Kniffel.

„Lasst uns Schach spielen“, schlägt Anna vor. „Und wenn das nicht geht, dann könnten wir kniffeln. Am wenigsten bin ich für Poker.“

„Och nö!“, nörgelt Chris. „Schach ist doch total langweilig, und außerdem sind wir zu dritt.“

„Dann spielen wir eben ein Blitzschach-Turnier“, erwidert Anna. „Da kommt jeder mal dran.“

„Aber Kniffeln macht viel mehr Spaß!“, widerspricht Chris. „Und wenn ihr das nicht wollt, dann sollten wir wenigstens pokern.“

„Poker finde ich am besten“, schaltet sich nun Belle ein, „denn im Bluffen bin ich richtig gut. Aber Kniffel gefällt mir am wenigsten. Wenn schon kein Poker, dann Schach.“

„Hört mal!“, sagt Anna. „Wenn wir spielen wollen, müssen wir uns auf ein Spiel einigen. Aber da wir alle andere Vorlieben haben, sollten wir vielleicht einfach darüber abstimmen, was wir spielen.“

Die Vorlieben (oder Präferenzen) von Anna, Belle und Chris bezüglich dieser drei Alternativen sind in Abbildung 1.1 dargestellt, von links nach rechts geordnet. Das am meisten bevorzugte Spiel steht also am weitesten links.

Zur Abstimmung müssen sie ein Wahlsystem verwenden, eine Regel also, die sagt, wie man aus ihren individuellen Präferenzen eine *gesellschaftliche* Präferenz (oder Rangfolge) bzw. den oder die Sieger der Wahl bestimmen kann. Wahlsysteme gibt es in großer Zahl, denn bereits seit den Wurzeln der Demokratie in der Antike, spätestens aber seit der Französischen Revolution und der Unabhängigkeitserklärung der Vereinigten Staaten von Amerika sind demokratische Wahlen von zentraler Bedeutung in menschlichen Gesellschaften. Doch nach welcher Regel sollten Anna, Belle und Chris wählen?

„Gut“, stimmt Belle zu, „lasst uns so abstimmen: Je zwei Spiele treten gegeneinander zum Vergleich an ...“

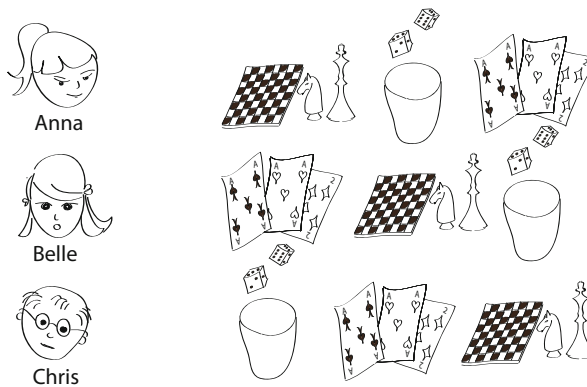


Abb. 1.1: Anna, Belle und Chris stimmen über ein Spiel ab

„Ich verstehe“, unterbricht Chris sie. „Welches Spiel in jedem Vergleich besser abschneidet als das andere, also in mehr Rangfolgen den Vorzug über das andere erhält, ...“

„... hat die Wahl gewonnen und wird gespielt!“, ergänzt Anna.

„Genau“, sagt Belle. „Denn es muss ja dann für uns alle besser sein als jedes andere Spiel. Schließlich zieht eine Mehrheit von uns es jeder Alternative vor.“

Diese Regel, die Belle vorgeschlagen hat, wurde schon vor über 225 Jahren von Marie Jean Antoine Nicolas de Caritat, dem Marquis de Condorcet (1743–1794), entdeckt (siehe Condorcet, 1785), einem französischen Philosophen, Mathematiker und Politikwissenschaftler. Condorcets Wahlsystem ist immer noch sehr beliebt, denn ein Condorcet-Gewinner muss notwendigerweise eindeutig sein und kann tatsächlich mit gutem Recht als die bestmögliche Alternative angesehen werden, da er jede andere Alternative im paarweisen Vergleich mit einer Mehrheit der Stimmen schlägt. Doch Condorcet-Wahlen haben auch ihre Tücken.

„Toll!“, ruft Anna. „Schach schlägt Kniffel! Dank meiner und Belles Stimme. Sorry, Chris, den Würfelbecher kannst du wieder einpacken.“

„Nicht so schnell!“, unterbricht Belle sie. „Wir spielen überhaupt nicht Schach! Poker schlägt Schach dank meiner und Chris’ Stimme, also werden wir wohl pokern.“ Sie überlegt. „Denn wenn Kniffel von Schach geschlagen wird und Schach von Poker, dann muss Poker doch auch Kniffel schlagen, oder?“

„Nicht so schnell!“, meldet sich nun Chris zu Wort. „Poker kannst du in den Skat drücken! Denn Kniffel schlägt Poker dank Annas und meiner Stimme.“

Die drei sehen sich ratlos an.

„Aber welches Spiel hat denn nun gewonnen?“

Der Gewinner ist . . . : Keines der drei Spiele! Diese Eigenschaft des Condorcet-Wahlsystems ist bekannt als das *Condorcet-Paradoxon*: Ein *Condorcet-Gewinner* (also ein Kandidat, der jeden anderen Kandidaten im paarweisen Vergleich schlägt) muss nicht immer existieren! Obwohl die individuellen Rangfolgen der drei Wähler in dem Sinn rational sind, dass es keine Zyklen gibt, ist die durch die Condorcet-Regel erzeugte gesellschaftliche Rangfolge irrational, also zyklisch: Schach schlägt Kniffel im paarweisen Vergleich, Kniffel schlägt Poker, aber Poker schlägt Schach. Dieser *Condorcet-Zyklus* ist in Abbildung 1.2 dargestellt.

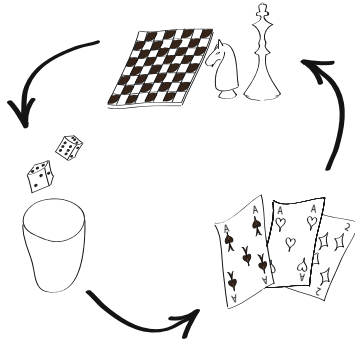


Abb. 1.2: Das Condorcet-Paradoxon

Seit der Arbeit von Condorcet (1785) befasst man sich in der Social-Choice-Theorie (auf Deutsch manchmal „Sozialwahltheorie“ genannt) mit der kollektiven Entscheidungsfindung durch Aggregation individueller Präferenzen mittels Wahlen. Kapitel 4 stellt die Grundlagen dieser Theorie und eine Vielzahl von Wahlsystemen und ihrer Eigenschaften vor. Auch in dieser Theorie wurden Nobelpreise für bahnbrechende Erkenntnisse verliehen, an Kenneth Arrow für sein berühmtes Unmöglichkeitstheorem (siehe Arrow, 1963) und an Amartya Sen für seine Forschungen zur Armut und Wohlfahrtsökonomie.

Auch Wähler können strategisch vorgehen und statt für ihre wahre Präferenz über die Kandidaten ihre Stimme so abgeben, wie sie ihnen für ihre Ziele nützlicher erscheint und – abhängig vom verwendeten Wahlsystem und von den Stimmen der anderen Wähler – ihrem Lieblingskandidaten zum Sieg verhilft. Nach dem Unmöglichkeitstheorem von Gibbard (1973) und Satterthwaite (1975) ist kein vernünftiges Wahlsystem sicher gegen eine solche Art der Manipulation. Wieder stehen dabei vor allem die algorithmischen Aspekte im Vordergrund: Wie schwer ist es, individuelle Präferenzen strategisch zu setzen, um die Wahl zu manipulieren?

Auch andere Arten der Einflussnahme auf den Ausgang von Wahlen, etwa durch Bestechung oder Wahlkontrolle, werden wir in Kapitel 4 vorstellen.

In Kapitel 5 wenden wir uns einem Gebiet zu, das man „*Judgment Aggregation*“ nennt und das deutlich jünger als das verwandte Gebiet der Präferenzaggregation durch Wahlen ist. Anders als bei diesem werden hier die individuellen Urteile von Experten (den „*judges*“) über logisch miteinander verknüpfte Aussagen zusammengeführt. Auch bei der gemeinsamen Urteilsfindung können sehr interessante paradoxe Situationen auftreten, wie z. B. das so genannte diskursive Dilemma, das auch unter dem Namen „*doctrinal paradox*“ bekannt ist (siehe Kornhauser und Sager, 1986; Pettit, 2001) und in diesem Kapitel genauer erläutert wird.

Wie bei Wahlen gibt es auch hier Möglichkeiten der Einflussnahme auf das Ergebnis einer Judgment-Aggregation-Prozedur. Externe Akteure könnten versuchen, Experten zu bestechen, damit deren gemeinsames Urteil in ihrem Sinn ausfällt. Die Experten selbst könnten versuchen, das Ergebnis zu manipulieren, indem sie nicht ihr ehrliches Urteil über die zur Disposition stehenden Aussagen verkünden. Die Untersuchung der algorithmischen und komplexitätstheoretischen Eigenschaften solcher Probleme wurde erst kürzlich von Endriss *et al.* (2010a,b) initiiert und steht noch am Anfang, eröffnet aber ein weites Forschungsfeld.

1.3 Teilen

„Ein Kompromiss“, sagte Ludwig Erhard, „das ist die Kunst, einen Kuchen so zu teilen, dass jeder meint, er habe das größte Stück bekommen.“

Um das gerechte Teilen von Kuchen (auf Englisch als „*cake-cutting*“ bezeichnet) geht es in Kapitel 6. Der „Kuchen“ ist dabei nur als eine Metapher zu verstehen, die für irgendeine teilbare Ressource oder ein teilbares Gut steht.

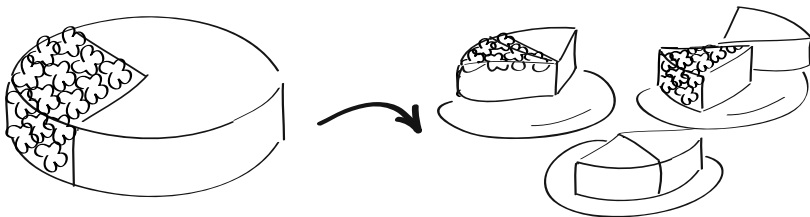


Abb. 1.3: Aufteilung eines Kuchens in drei Portionen

Abbildung 1.3 zeigt die Aufteilung eines Kuchens in drei Portionen. Doch wann ist eine Aufteilung „gerecht“? Wie viele Mütter und Väter sind schon kläglich bei dem Versuch gescheitert, einen Kuchen gerecht unter ihren Kindern aufzuteilen? Im schlimmsten Fall meinen alle Kinder, gerade das schlechteste Stück erhalten zu haben, und fühlen sich ungerecht behandelt. Denn jeder hat da seine eigene, ganz subjektive Bewertung der verschiedenen Stücke des Kuchens. In einem größeren politischen Maßstab könnte man auch fragen: Wie viele Unterhändler sind im

Nahen Osten schon kläglich bei dem Versuch gescheitert, die Gebiete zwischen dem Westjordanland und dem Gazastreifen – oder auch nur Jerusalem – gerecht unter Israelis und Palästinensern aufzuteilen?

Cake-cutting-Protokolle haben den Zweck, eine gerechte Aufteilung des Kuchens und damit Zufriedenheit unter den beteiligten Spielern zu erzeugen. „Gerecht“ kann dabei verschieden interpretiert werden. Es kann z. B. bedeuten, dass kein Spieler einen geringeren Anteil vom Kuchen erhält, als was ihm seiner Bewertung nach zusteht (sein „proportionaler“ Anteil am Kuchen). Oder es kann sogar bedeuten, dass kein Neid unter den Spielern entsteht. Diese Eigenschaften soll das Protokoll nach Möglichkeit sogar *garantieren*, d. h., die Proportionalität bzw. die Neidfreiheit der Aufteilung soll unabhängig davon gelten, welche individuellen Bewertungen die Spieler haben. Steinhaus (1948) hat als einer der Ersten das Problem der gerechten Kuchenaufteilung als eine überaus anspruchsvolle und schöne mathematische Aufgabe formuliert und erste Lösungen vorgeschlagen.

In Kapitel 7 schließlich wird das mit dem Cake-cutting verwandte Problem der gerechten Aufteilung von unteilbaren Gütern oder Ressourcen behandelt. Dieses Gebiet wird als „*Multiagent Resource Allocation*“ bezeichnet und steht auch im Zusammenhang mit Auktionen und E-Commerce. Auch hier haben die beteiligten Spieler individuelle, subjektive Bewertungen für die einzelnen Güter oder Ressourcen, aber auch für alle Bündel von Gütern oder Ressourcen. Neben dem individuellen Nutzen, den jeder Spieler für sich realisieren kann, interessieren wir uns dabei auch besonders für den optimalen *gesellschaftlichen* Nutzen, die „soziale Wohlfahrt“, die man in verschiedener Weise definieren kann. Eine optimale Aufteilung aller Güter, durch die die soziale Wohlfahrt maximiert wird, stellt ein schwieriges kombinatorisches Problem dar, dessen algorithmische und komplexitätstheoretische Eigenschaften ebenfalls untersucht werden.

1.4 Was ist Computational Social Choice?

In den einzelnen Kapiteln zum Spielen, Wählen und Teilen werden, wie gesagt, auch viele algorithmische Aspekte im Mittelpunkt stehen. In der *Computational Social Choice* (COMSOC) werden einerseits Methoden der Informatik (wie Algorithmenentwurf, Komplexitätsanalyse usw.) auf Mechanismen der Social-Choice-Theorie (wie Wahlsysteme und gerechte Aufteilungsverfahren) angewandt. Andererseits werden Konzepte und Ideen aus der Social-Choice-Theorie in die Informatik übertragen, etwa auf dem Gebiet der Multi-Agenten-Systeme, beim Netzwerkentwurf oder bei der Entwicklung von Ranking-Algorithmen.

Seit 2006 treffen sich Wissenschaftler aus aller Welt, die auf diesem Gebiet arbeiten, alle zwei Jahre auf dem „*International Workshop on Computational Social Choice*“, der bisher in Amsterdam, Liverpool und Düsseldorf stattfand. Die Tagungsbände dieser drei Treffen wurden herausgegeben von Endriss und Lang

(2006), Endriss und Goldberg (2008) und Conitzer und Rothe (2010). Wen das Lesen dieses Buches, das nicht mehr als einen Einstieg in dieses faszinierende Gebiet ermöglicht, zu einem tieferen Studium animiert, der wird in den COMSOC-Bänden eine anregende Fülle von Ideen finden und aktuellen spannenden Forschungsfragen begegnen, die noch auf eine Lösung warten.

Zu spezifischen COMSOC-Themen sind auch die Übersichtsartikel von Chevalleyre *et al.* (2006, 2007), Conitzer (2010), Daskalakis *et al.* (2009a), Faliszewski *et al.* (2010b) und Faliszewski und Procaccia (2010) sowie die Buchkapitel von Faliszewski *et al.* (2009c) und Baumeister *et al.* (2010) zu empfehlen.

Doch nun wollen wir spielen, wählen und teilen . . .

Halt! Bevor wir mit dem Spielen, Wählen und Teilen wirklich anfangen können, sind noch ein paar Vorbereitungen nötig. Wie gesagt, in allen Teilen dieses Buches wollen wir uns vor allem mit den algorithmischen und komplexitätstheoretischen Problemen des Spielens, Wählens und Teilens auseinandersetzen. Doch wie ermittelt man die Komplexität eines Problems?

Um diese Frage beantworten zu können, werden in dem nun folgenden Exkurs die Grundlagen der Komplexitätstheorie vorgestellt, so knapp und informal wie möglich und so ausführlich wie nötig. Wesentliche Konzepte dieser Theorie, die in nahezu allen folgenden Kapiteln eine zentrale Rolle spielen, werden an SAT illustriert, dem Erfüllbarkeitsproblem der Aussagenlogik, das einerseits eine prominente Stellung in der Komplexitätstheorie und Algorithmik einnimmt und andererseits in Kapitel 5 besonders wichtig sein wird, wenn wir aussagenlogische Formeln im Zusammenhang mit der Urteilsfindung verwenden.

Man kann diesen Exkurs auch erst einmal überspringen und gegebenenfalls später zurückkehren, um den einen oder anderen Begriff nachzuschlagen.

1.5 Ein Exkurs in die Komplexitätstheorie

1.5.1 Einige Grundlagen der Komplexitätstheorie

Die Berechnungskomplexität von Problemen wird seit über vierzig Jahren in der Komplexitätstheorie untersucht, einem Teilgebiet der Theoretischen Informatik, das sich aus der Berechenbarkeitstheorie (siehe z.B. Homer und Selman, 2001; Rogers, 1967) entwickelt hat. Letztere beschäftigt sich u. a. damit, nachzuweisen, dass bestimmte Probleme algorithmisch gar nicht lösbar sind. Die Komplexitätstheorie dagegen befasst sich nur mit algorithmisch lösbaren Problemen, fragt aber genauer nach dem Berechnungsaufwand, den ihre Lösung erfordert. Zur Komplexitätstheorie gibt es viele nützliche Lehrbücher und Monographien, etwa die von Bovet und Crescenzi (1993), Hemaspaandra und Ogihara (2002), Papadimitriou (1995), Rothe (2008), Wagner und Wechsung (1986), Wechsung (2000) und Wege-

ner (2003). Algorithmische und komplexitätstheoretische Konzepte und Methoden werden auch für die hier behandelten Probleme des Spielens, Wählens und Teilens eine zentrale Rolle spielen.

Komplexitätstheorie und Algorithmik verhalten sich zueinander wie Yin und Yang.¹ Während man in der Algorithmik möglichst effiziente Algorithmen zur Lösung von Problemen entwirft und somit möglichst gute *obere* Zeitschranken für die algorithmische Lösung dieser Probleme zeigt, versucht man in der Komplexitätstheorie nachzuweisen, dass sich manche Probleme überhaupt nicht effizient lösen lassen, es also gar keine effizienten Algorithmen für diese Probleme gibt. Man versucht also, eine möglichst gute *untere* Schranke für die Zeit zu beweisen, die zur algorithmischen Lösung eines Problems erforderlich ist.

Zeit ist dabei ein diskretes Komplexitätsmaß, definiert als die Anzahl der elementaren Rechenschritte, die ein Algorithmus in Abhängigkeit von der Größe der Eingabe ausführt. Die Eingabegröße hängt von der gewählten Codierung ab. In der Regel werden die Probleminstanzen binär dargestellt, also über dem Alphabet $\Sigma = \{0, 1\}$ codiert, damit sie durch einen Computer verarbeitet werden können, der diesen Algorithmus ausführt.

Unterschiedliche Codierungen haben in der Regel jedoch nur einen unbedeutenden Einfluss auf die Rechenzeit. Zum Beispiel werden konstante Faktoren bei der Laufzeitanalyse von Algorithmen üblicherweise vernachlässigt. Von Codierungsdetails werden wir daher hier absehen. Was ein „elementarer Rechenschritt“ eines Algorithmus ist, hängt von dem verwendeten Algorithmenmodell ab. In der Theoretischen Informatik und speziell in der Komplexitätstheorie ist das nach ihrem Erfinder Alan Turing (1936) benannte Modell der *Turingmaschine* gebräuchlich, da es besonders simpel und dennoch universell ist: Alles, was überhaupt berechenbar ist, lässt sich nach der These von Church (1936) auch auf einer Turingmaschine berechnen. Auch hier verzichten wir auf formale Details und verweisen stattdessen auf die oben angegebene Literatur zur Berechenbarkeits- und Komplexitätstheorie. Algorithmen werden wir stets informal beschreiben; ob sie dann in einem abstrakten Algorithmenmodell wie der Turingmaschine oder in einer konkreten Programmiersprache implementiert werden, ist für unsere Darstellung egal. Allerdings unterscheiden wir zwischen verschiedenen Typen von Turingmaschinen bzw. Algorithmen:

- Bei der Berechnung einer *deterministischen Turingmaschine* bei irgendeiner Eingabe ist jeder Rechenschritt eindeutig bestimmt, d. h., zu jedem Zeitpunkt der Berechnung ergibt sich aus der aktuellen Zustandsbeschreibung der Maschine (man nennt dies eine „*Konfiguration*“) eine eindeutige Folgekonfiguration,

¹In der chinesischen Philosophie bezeichnen Yin und Yang Gegensätze, die sich gegenseitig bedingen und ohneinander nicht sein können.

bis schließlich eine akzeptierende oder ablehnende Endkonfiguration erreicht wird.

- Die Berechnung einer *nichtdeterministischen Turingmaschine* bei irgendeiner Eingabe ist allgemeiner: Hier gibt es in jedem Rechenschritt die Möglichkeit einer nichtdeterministischen Verzweigung, d. h., zu jedem Zeitpunkt der Berechnung ergeben sich aus der aktuellen Konfiguration der Maschine möglicherweise mehrere Folgekonfigurationen, bis schließlich eine akzeptierende oder ablehnende Endkonfiguration erreicht wird. Demgemäß verläuft eine solche nichtdeterministische Berechnung nicht als eine deterministische Folge von Konfigurationen, sondern es ergibt sich ein *nichtdeterministischer Berechnungsbaum*,
 - dessen Wurzel die Startkonfiguration ist,
 - dessen innere Knoten die aus der Startkonfiguration erreichbaren Konfigurationen sind, bei denen die Berechnung noch nicht terminiert, und
 - dessen Blätter die akzeptierenden oder ablehnenden Endkonfigurationen sind.

Damit die Eingabe akzeptiert wird, genügt es, dass es mindestens einen akzeptierenden Berechnungspfad im Berechnungsbaum gibt. Nur wenn sämtliche Pfade in diesem Baum zu einer ablehnenden Endkonfiguration führen, wird die Eingabe verworfen.

Sowohl deterministische Berechnungen als auch einzelne oder sämtliche Pfade in einem nichtdeterministischen Berechnungsbaum können bei bestimmten Eingaben unendlich lang sein, also nie terminieren. Tatsächlich ist es algorithmisch nicht entscheidbar, ob irgendein gegebener Algorithmus (bzw. eine gegebene Turingmaschine) bei irgendeiner gegebenen Eingabe jemals anhält. Dies ist das berühmte *Halteproblem*, eines der Standardbeispiele für unentscheidbare Probleme in der Berechenbarkeitstheorie. Wie gesagt betrachten wir hier jedoch nur algorithmisch lösbare Probleme und interessieren uns für den Aufwand, den ihre Lösung erfordert.

Neben Determinismus und Nichtdeterminismus gibt es noch viele andere Berechnungsparadigmen, beispielsweise randomisierte Algorithmen, und neben der Rechenzeit gibt es auch andere Komplexitätsmaße, wie z. B. den bei einer Berechnung erforderlichen Speicherplatz. Diese werden wir hier jedoch nicht betrachten und uns im Wesentlichen auf die Analyse der Rechenzeit von deterministischen und nichtdeterministischen Algorithmen beschränken.

In der Komplexitätstheorie fasst man Probleme, deren Lösung ungefähr denselben Aufwand erfordert, in so genannten Komplexitätsklassen zusammen, und die wichtigsten Komplexitätsklassen bezüglich der Rechenzeit sind

- P („*deterministische Polynomialzeit*“) und
- NP („*nichtdeterministische Polynomialzeit*“).

P (bzw. NP) ist definiert als die Klasse aller Probleme, die sich mit einer deterministischen (bzw. nichtdeterministischen) Turingmaschine in Polynomialzeit lö-

sen lassen. Deterministische Polynomialzeit-Algorithmen gelten als effizient, weil das Wachstum eines Polynoms, wie z. B. $p(n) = n^2 + 13 \cdot n + 7$, relativ moderat ist, im Gegensatz zum explosiven Wachstum einer Exponentialfunktion, wie z. B. $e(n) = 2^n$. Für sehr kleine Eingabegrößen n mag die Funktion e zwar noch harmlose Werte haben (z. B. ist $e(n) = 2^n < n^2 + 13 \cdot n + 7 = p(n)$ für alle $n < 8$), aber schon für Eingabegrößen, die nur wenig größer sind, explodiert die Exponentialfunktion im Vergleich zum Polynom (für $n = 30$ z. B. ist $e(30) = 1073741824$ deutlich größer als $p(30) = 1297$). Für noch etwas größere Eingabegrößen wie z. B. $n = 100$, die in der Praxis nicht ungewöhnlich sind, erreichen Exponentialfunktionen geradezu astronomische Werte. So schätzt man die Anzahl der Atome im sichtbaren Universum auf etwa 10^{77} . Ein Algorithmus, der die Laufzeit 10^n hat und der – nur zur Veranschaulichung einmal angenommen – pro Rechenschritt ein Atom zerstört, hätte bei einer Eingabe der Größe 77 das gesamte sichtbare Universum ausgelöscht, wenn er am Ende seiner Berechnung ankommt, also auch sich selbst, weshalb die Annahme, er würde jemals mit einem Ergebnis terminieren, absurd ist. Doch auch wenn er bei seiner Berechnung keine Atome zerstört (was Algorithmen normalerweise ja nicht tun), wer immer ihn auf diese Eingabe angesetzt hätte, wäre am Ende dieser Berechnung sowieso schon seit Jahrmillionen verstorben.

Nichtdeterministische Polynomialzeit-Algorithmen gelten hingegen nicht als effizient. Würde man einen NP-Algorithmus deterministisch auszuführen versuchen (man sagt: „deterministisch simulieren“), also Pfad für Pfad den entsprechenden Berechnungsbaum nach einer akzeptierenden Endkonfiguration durchsuchen, so würde dieser deterministische Algorithmus wohl Exponentialzeit brauchen. Denn ist die Rechenzeit des NP-Algorithmus bei Eingaben der Größe n durch ein Polynom $p(n)$ beschränkt und verzweigt der NP-Algorithmus in jedem inneren Knoten des Berechnungsbaums in höchstens zwei Folgeknoten, so kann es bis zu $2^{p(n)}$ Pfade in diesem Baum geben, und erst wenn man den letzten dieser Pfade erfolglos untersucht hat, kann man sicher sein, dass es wirklich keinen akzeptierenden Pfad gibt. Dies ist natürlich kein Beweis dafür, dass P ungleich NP ist.

Tatsächlich ist die „P = NP?“-Frage seit etwa 40 Jahren ungelöst. Sie ist die vielleicht wichtigste offene Frage der Theoretischen Informatik überhaupt und sie ist eines der sieben *Millennium-Probleme*, deren Lösungen das Clay Mathematics Institute in Cambridge, Massachusetts, jeweils mit einem Preisgeld von einer Million US-Dollar honoriert. Natürlich gilt $P \subseteq NP$, doch es ist offen, ob diese Inklusion echt ist. Nach einer Abstimmung, die William Gasarch (2002) im Jahre 2002 unter Komplexitätstheoretikern durchführte, glaubt die überwiegende Mehrheit von ihnen, dass $P \neq NP$ gilt. Nur ein *Beweis* dieser Ungleichheit ist bisher niemandem gelungen, auch wenn immer einmal wieder ein „Beweis“ lanciert wird, der sich kurz darauf jedoch als fehlerhaft herausstellt. Das Preisgeld von einer Million US-Dollar für eine korrekte Lösung dieses faszinierenden Problems ist noch zu holen!

Um eine *obere Schranke* $t(n)$ für die Komplexität eines Problems zu zeigen, genügt es, einen *spezifischen* Algorithmus für das betrachtete Problem zu finden, der es in der durch t vorgegebenen Zeit löst, der also für Eingaben der Größe

n in der Zeit höchstens $t(n)$ arbeitet. Ein P-Algorithmus etwa zeigt, dass das entsprechende Problem in der Zeit $p(n)$ für ein Polynom p und somit effizient lösbar ist. Von konstanten Faktoren und endlich vielen Ausnahmen kann man, wie erwähnt, bei der Angabe oberer Schranken absehen, da man sich nur für das *asymptotische Wachstum* von Komplexitätsfunktionen interessiert. Die folgenden Notationen, die das asymptotische (d. h. größenordnungsmäßige) Wachstum von Funktionen beschreiben, gehen auf Bachmann (1894) und Landau (1909) zurück.

Definition 1.1 (Asymptotisches Wachstum)

Seien s und t Funktionen von \mathbb{N} in \mathbb{N} , wobei \mathbb{N} die Menge der natürlichen Zahlen bezeichnet.

1. $s \in \mathcal{O}(t)$ genau dann, wenn es eine reelle Konstante $c > 0$ und eine Zahl $n_0 \in \mathbb{N}$ gibt, sodass $s(n) \leq c \cdot t(n)$ für alle $n \in \mathbb{N}$ mit $n \geq n_0$ gilt.
Man sagt dann, dass s *asymptotisch höchstens so stark wie t wächst*.
2. $s \in \Omega(t)$ genau dann, wenn $t \in \mathcal{O}(s)$.
Man sagt dann, dass s *asymptotisch mindestens so stark wie t wächst*.
3. Die Klasse $\Theta(t) = \mathcal{O}(t) \cap \Omega(t)$ enthält alle Funktionen, die *asymptotisch genauso stark wie t wachsen*.
4. $s \in o(t)$ genau dann, wenn es für alle reelle Konstanten $c > 0$ eine Zahl $n_0 \in \mathbb{N}$ gibt, sodass $s(n) < c \cdot t(n)$ für alle $n \in \mathbb{N}$ mit $n \geq n_0$ gilt.
Man sagt dann, dass s *asymptotisch echt schwächer als t wächst*.
5. $s \in \omega(t)$ genau dann, wenn $t \in o(s)$.
Man sagt dann, dass s *asymptotisch echt stärker als t wächst*.



Beispielsweise gilt $p \in \mathcal{O}(e)$ bzw. $e \in \Omega(p)$ für die Exponentialfunktion $e(n) = 2^n$ und das Polynom $p(n) = n^2 + 13 \cdot n + 7$, die oben definiert wurden. Da jede Exponentialfunktion asymptotisch nicht nur mindestens so stark, sondern sogar echt stärker als jedes Polynom wächst, gilt sogar $p \in o(e)$ bzw. $e \in \omega(p)$. Auch ist klar, dass $\mathcal{O}(1)$ die Klasse aller konstanten Funktionen ist. Der wesentliche Unterschied zwischen den Definitionen der \mathcal{O} - und der o -Notation ist die Quantifizierung der reellen positiven Konstante c . Während der Existenzquantor vor c in der Definition von $s \in \mathcal{O}(t)$ dafür sorgt, dass man beliebig große konstante Faktoren vernachlässigen darf, bewirkt der Allquantor vor c in der Definition von $s \in o(t)$, dass $c \cdot t(n)$ selbst dann größer als $s(n)$ ist, wenn das Wachstum von $c \cdot t(n)$ durch einen beliebig kleinen konstanten Faktor c gebremst wird – so viel stärker wächst t im Vergleich zu s (siehe auch z. B. Rothe, 2008; Gurski *et al.*, 2010, für weitere Details, Eigenschaften und Beispiele).

Obere Schranken für die Komplexität eines Problems werden in der \mathcal{O} -Notation angegeben, und es genügt, einen geeigneten Algorithmus zu finden, der das Problem innerhalb dieser oberen Schranke löst. Im Gegensatz dazu spricht man von einer *unteren Schranke* $u(n)$ für die Komplexität eines Problems, falls *kein* Algorithmus für dieses Problem mit weniger als der durch $u(n)$ vorgegebenen Zeit

auskommen kann. Mindestens der Zeitaufwand $u(n)$ ist also nötig, um das Problem zu lösen, vielleicht zwar nicht für alle Eingaben der Größe n , aber für mindestens eine Eingabe dieser Größe, und egal, durch welchen Algorithmus des betrachteten Algorithmentyps. Auch hier vernachlässigen wir konstante Faktoren und lassen endlich viele Ausnahmen zu.

Offenbar sind beide Aspekte – die oberen und die unteren algorithmischen Zeitschranken zur Lösung eines Problems – eng miteinander verbunden, sie sind die zwei Seiten ein und derselben Medaille. Gelingt es, übereinstimmende obere und untere Schranken für ein Problem zu finden, so hat man dessen inhärente Komplexität bestimmt.² Leider ist dies jedoch oft nicht möglich.

1.5.2 Das Erfüllbarkeitsproblem der Aussagenlogik

Betrachten wir zum Beispiel das berühmte *Erfüllbarkeitsproblem* der Aussagenlogik, das auch mit dem englischen Begriff SATISFIABILITY (oder kurz mit SAT) bezeichnet wird. Entscheidungsprobleme wie dieses, bei denen eine Ja/Nein-Frage zu beantworten ist, stellen wir in der folgenden Form dar:

SATISFIABILITY (SAT)	
Gegeben:	Eine boolesche Formel φ .
Frage:	Gibt es eine Belegung der Variablen von φ , die φ wahr macht?

Dabei versteht man unter einer *booleschen* (oder *aussagenlogischen*) *Formel* eine Verknüpfung von *atomaren Aussagen* (auch als die *Variablen der Formel* bezeichnet) durch *boolesche Operationen* wie die

- *Konjunktion* (d. h. *und* bzw. \wedge),
- *Disjunktion* (d. h. *oder* bzw. \vee),
- *Negation* (d. h. *nicht* bzw. \neg),
- *Implikation* (d. h., *wenn ... , dann ...* bzw. \implies),
- *Äquivalenz* (d. h., *... genau dann, wenn ...* bzw. \iff)
- usw.

²Um Missverständnissen vorzubeugen: Übereinstimmende obere und untere Schranken für ein Problem zu finden, bedeutet nicht, dass man einen Algorithmus mit der Laufzeit z. B. $\Theta(t)$ findet, der das Problem löst. Dann hätte man lediglich die Laufzeit *dieses* Algorithmus (der *eine* obere Schranke zeigt) sehr gut analysiert, sodass keine wesentlichen Verbesserungen für ihn mehr möglich sind. Um eine mit $\Theta(t)$ übereinstimmende untere Schranke zu zeigen, muss man jedoch nachweisen, dass *jeder* Algorithmus für dieses Problem keine asymptotisch echt bessere Laufzeit hat.

Tab. 1.1: Wahrheitstabelle für einige boolesche Operationen

x	y	$x \wedge y$	$x \vee y$	$\neg x$	$x \implies y$	$x \iff y$
0	0	0	0	1	1	1
0	1	0	1	1	1	0
1	0	0	1	0	0	0
1	1	1	1	0	1	1

Die o. g. booleschen Operationen sind über ihre Wahrheitstabellen definiert (siehe Tabelle 1.1), wobei die Wahrheitswerte (auch als *boolesche Konstanten* bezeichnet) *wahr* und *falsch* kurz durch 1 und 0 dargestellt werden. Jede boolesche Formel φ mit n Variablen repräsentiert eine boolesche Funktion $f_\varphi : \{0,1\}^n \rightarrow \{0,1\}$. Es gibt genau zwei nullstellige boolesche Funktionen (nämlich die beiden booleschen Konstanten), vier einstellige boolesche Funktionen (z. B. die Identität und die Negation) und 16 zweistellige boolesche Funktionen (z. B. die Funktionen, die den booleschen Operationen \wedge , \vee , \implies und \iff aus Tabelle 1.1 entsprechen). Im Allgemeinen gibt es 2^{2^n} n -stellige boolesche Funktionen, weil je zwei davon durch eine der 2^n möglichen Wahrheitswertbelegungen festgelegt sind, nämlich die mit dem Wert 0 und die mit dem Wert 1 für die gegebene Belegung.

Den Wahrheitswert einer booleschen Formel – also den Wert der entsprechenden booleschen Funktion – kann man abhängig von den Wahrheitswerten bestimmen, mit denen ihre Variablen belegt sind. Die von drei Variablen abhängige boolesche Formel

$$\varphi(x, y, z) = (x \wedge \neg y \wedge z) \vee (\neg x \wedge \neg y \wedge \neg z) \quad (1.1)$$

z. B. lässt sich für die Wahrheitswertbelegung $(1, 1, 1)$ für (x, y, z) auswerten zu

$$\varphi(1, 1, 1) = (1 \wedge \neg 1 \wedge 1) \vee (\neg 1 \wedge \neg 1 \wedge \neg 1) = (1 \wedge 0 \wedge 1) \vee (0 \wedge 0 \wedge 0) = 0 \vee 0 = 0,$$

ist unter dieser Belegung also falsch. Mit der Belegung $(1, 0, 1)$ dagegen ergibt sich

$$\varphi(1, 0, 1) = (1 \wedge \neg 0 \wedge 1) \vee (\neg 1 \wedge \neg 0 \wedge \neg 1) = (1 \wedge 1 \wedge 1) \vee (0 \wedge 1 \wedge 0) = 1 \vee 0 = 1,$$

φ ist unter dieser Belegung also wahr. Eine boolesche Formel φ ist *erfüllbar*, falls es eine Belegung ihrer Variablen mit Wahrheitswerten gibt, die φ wahr macht. Die Formel φ aus (1.1) hat zwei erfüllende Belegungen, $(0, 0, 0)$ und die schon erwähnte Belegung $(1, 0, 1)$. Keine der anderen sechs Belegungen erfüllt φ .

Ein *Literal* ist eine Variable, wie z. B. x , oder ihre Negation, $\neg x$. Ein *Implikant* ist eine Konjunktion von Literalen, wie z. B. $(x \wedge \neg y \wedge z)$, und eine *Klausel* ist eine Disjunktion von Literalen, wie z. B. $(x \vee y \vee \neg z)$. Formeln wie die in (1.1) angegebene sind in *disjunktiver Normalform* (kurz DNF), d. h., sie sind eine Disjunktion von Implikanten. Eine Formel ist in *konjunktiver Normalform* (kurz KNF), falls sie eine Konjunktion von Klauseln ist.

Jede boolesche Formel lässt sich in eine äquivalente Formel in DNF bzw. in eine äquivalente Formel in KNF umformen, wobei die neue Formel allerdings exponentiell größer als die gegebene Formel sein kann. Zwei Formeln heißen dabei *äquivalent*, falls sie für alle Belegungen ihrer Variablen denselben Wahrheitswert haben. Beispielsweise kann man leicht überprüfen, dass

$$\begin{aligned}\varphi'(x, y, z) = & (x \vee y \vee \neg z) \wedge (x \vee \neg y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge \\ & (\neg x \vee y \vee z) \wedge (\neg x \vee \neg y \vee z) \wedge (\neg x \vee \neg y \vee \neg z)\end{aligned}\quad (1.2)$$

eine zu φ aus (1.1) äquivalente Formel in KNF ist. Ist die Anzahl der Literale in allen Klauseln einer Formel in KNF durch eine Konstante k beschränkt, so sagt man, die Formel ist in k -KNF. Zum Beispiel ist die Formel φ' aus (1.2) in 3-KNF. Die Einschränkung des oben definierten Problems SAT auf Formeln, die in k -KNF sind, wird als k -SAT bezeichnet.

Wie schwer ist SAT? Da es für eine Formel mit n Variablen 2^n Wahrheitswertbelegungen gibt, arbeitet der naive deterministische Algorithmus für SAT in der Zeit $\mathcal{O}(n^2 \cdot 2^n)$, denn bei Eingabe einer Formel φ mit n Variablen testet er nacheinander für alle möglichen Belegungen der Variablen von φ , ob sie φ erfüllen, und jeder einzelne Test kann in quadratischer Zeit erledigt werden. Wird eine erfüllende Belegung gefunden, hält der Algorithmus und akzeptiert, aber ablehnen kann er eine unerfüllbare Formel erst, wenn sämtliche Belegungen erfolglos getestet wurden. Dieser naive Algorithmus für SAT kann verbessert werden, wie wir unten sehen werden. Auch ist bekannt, dass bestimmte Einschränkungen des Problems SAT effizienter lösbar sind. Liegt die gegebene boolesche Formel beispielsweise in DNF vor, so lässt sich die Entscheidung, ob sie erfüllbar ist oder nicht, in Polynomialzeit treffen. Denn in diesem Fall kann man nacheinander die Erfüllbarkeit der Implikanten der Formel testen; hat man einen erfüllbaren Implikanten gefunden, akzeptiert man, andernfalls lehnt man ab, sobald man beim letzten unerfüllbaren Implikanten angekommen ist. Liegt die gegebene boolesche Formel dagegen in KNF vor, scheint das Problem viel schwieriger zu sein, nämlich so schwierig wie das uneingeschränkte Problem SAT. Doch auch in diesem Fall ist ein effizienter Algorithmus möglich, sofern keine Klausel der Formel mehr als zwei Literale hat: Jones *et al.* (1976) zeigten, dass 2-SAT in Polynomialzeit lösbar ist und sogar in einer vermutlich noch kleineren Komplexitätsklasse als P liegt, da es nämlich sogar mit „nichtdeterministischem logarithmischem Speicherplatz“ gelöst werden kann (siehe z. B. auch Rothe, 2008, für einen ausführlichen Beweis).

Die Laufzeit des naiven Algorithmus für SAT wurde oben als Funktion der Anzahl der Variablen der Eingabeformel angegeben. Warum? Es wäre ja naheliegend, die Größe einer Formel als die Anzahl der *Vorkommen* von positiven oder negierten Variablen zu definieren (wobei Codierungsdetails bezüglich der Klammern, \wedge -, \vee - und \neg -Symbole usw. bereits vernachlässigt sind), und in einer Formel kann jede Variable sehr oft vorkommen. Zum Beispiel kommen x , y und z in der Formel φ aus (1.1) je zweimal, aber in der Formel φ' aus (1.2) jeweils sechs Mal

vor. Allgemein könnte jede der Variablen exponentiell (in der Variablenanzahl) oft vorkommen. Dies liegt jedoch nur daran, dass die in (1.1) bzw. (1.2) verwendete Formelschreibweise verschwenderischer als nötig ist. Stellt man eine Formel dagegen durch einen *booleschen Schaltkreis* dar (siehe Abbildung 1.4), so erhält man für die gegebene SAT-Instanz eine kompakte Repräsentation, deren Größe (unter einer vernünftigen Codierung) polynomiell in der Variablenanzahl ist. Polynomielle Faktoren spielen bei der Abschätzung einer exponentiellen Laufzeit jedoch eine untergeordnete Rolle und können problemlos vernachlässigt werden.

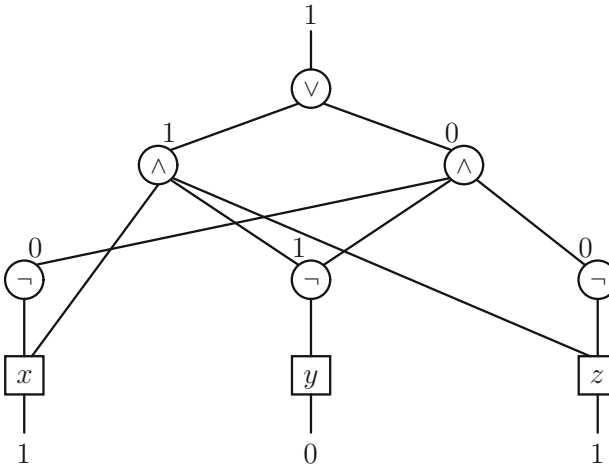


Abb. 1.4: Ein boolescher Schaltkreis für die Formel in (1.1) mit der erfüllenden Belegung $(1, 0, 1)$

Alternativ werden die Laufzeiten von SAT- oder k -SAT-Algorithmen auch in der Anzahl der Klauseln oder in beiden Parametern, also sowohl in der Anzahl der Variablen als auch in der Anzahl der Klauseln, angegeben.

Der naive SAT-Algorithmus lässt sich, wie bereits erwähnt, noch wesentlich verbessern. Seit Jahrzehnten wird dieses faszinierende, wichtige, in der Informatik überaus zentrale Problem untersucht, um immer bessere SAT-Solver zu entwickeln. Einer Sportart nicht unähnlich werden dabei immer neue Rekorde aufgestellt (siehe z.B. die Übersichtsartikel von Woeginger, 2003; Schöning, 2005; Riege und Rothe, 2006b, für das Erfüllbarkeitsproblem und andere harte Probleme).

Die Tabellen 1.2 und 1.3 listen einige dieser oberen Schranken für 3-SAT und für k -SAT bzw. SAT auf, wobei n die Anzahl der Variablen und m die Anzahl der Klauseln der Eingabeformel bezeichnet und polynomielle Faktoren weggelassen werden. Alle diese oberen Schranken beziehen sich auf *deterministische* Algorithmen (im Gegensatz zu *randomisierten* Algorithmen, die möglicherweise zwar effizienter sind, aber Fehler machen können) und auf den „worst case“, d. h., bei der Laufzeitanalyse geht man jeweils vom „schlimmsten“ Fall aus, also von den „hartnäckigsten“ Probleminstanzen.

Tab. 1.2: Einige obere Schranken für 3-SAT

obere Schranke	Quelle
2^n	naiver 3-SAT-Algorithmus
1.6181^n	Monien und Speckenmeyer (1985)
1.4970^n	Schiermeyer (1996)
1.4963^n	Kullmann (1999)
1.4802^n	Dantsin <i>et al.</i> (2002)
1.4726^n	Brueggemann und Kern (2004)

Wie man sieht, sind alle oberen Schranken für die verschiedenen Varianten des Erfüllbarkeitsproblems in den Tabellen 1.2 und 1.3 immer noch exponentiell in der Anzahl der Variablen. Was hat man davon, einen Exponentialzeit-Algorithmus, der z. B. in der Zeit 2^n läuft, auf einen solchen mit einer Laufzeit von c^n zu verbessern, wobei c eine Konstante mit $1 < c < 2$ ist? Bei der praktischen Anwendung kann eine solche verbesserte Exponentialzeit eine große Auswirkung haben, da man in derselben absoluten Zeit deutlich größere Eingaben verarbeiten kann. Denn weil das exponentielle Wachstum erst ab einer gewissen Eingabegröße zuschlägt, können für moderate Eingabegrößen, die unter dieser Schwelle liegen, auch Exponentialzeit-Algorithmen praktikabel sein. Nehmen wir etwa an, dass wir mit einem Algorithmus, der in der Zeit 2^n arbeitet, innerhalb einer Stunde Eingaben bis zur Größe 30 lösen können. Verbessern wir diesen Algorithmus, sodass wir einen neuen Algorithmus mit einer Laufzeit von z. B. $\sqrt{2}^n \approx 1.4142^n$ erhalten, so können wir nun innerhalb einer Stunde mit demselben Computer Eingaben bis zur Größe 60 lösen, denn es gilt $\sqrt{2}^{60} = 2^{(1/2) \cdot 60} = 2^{30}$. In der Praxis kann dies einen wesentlichen Unterschied ausmachen.

Tab. 1.3: Einige obere Schranken für k -SAT und SAT

Problem	obere Schranke	Quelle
2-SAT	P	Jones <i>et al.</i> (1976)
3-SAT	1.4726^n	Brueggemann und Kern (2004)
k -SAT, $k \geq 4$	$(2 - 2/(k+1))^n$	Dantsin <i>et al.</i> (2002)
SAT	$2^{n(1-1/\log(2m))}$	Dantsin und Wolpert (2004)

Doch wie soll man eine *untere* Schranke für SAT beweisen? Tatsächlich ist für uneingeschränkte deterministische Algorithmen bisher keine bessere untere Schranke als Linearzeit für SAT bekannt (siehe z. B. Fortnow *et al.*, 2005). Diese untere Schranke ist aber trivial, denn Linearzeit ist schon nötig, um die gesamte Eingabe zu lesen. Aufgrund dieser Schwierigkeit, untere Schranken im Sinne der Ω - oder ω -Notation für Probleme zu beweisen, geht man folgendermaßen vor: Man vergleicht die Komplexität eines Problems mit der Komplexität anderer Probleme

einer Komplexitätsklasse und versucht zu zeigen, dass es mindestens so schwer zu lösen ist wie alle die anderen Probleme der Klasse. Gelingt dies, so ist das betrachtete Problem „hart“ für die ganze Komplexitätsklasse, und auch in diesem Sinn spricht man von einer *unteren Schranke*: Die entsprechende Komplexitätsklasse liefert eine untere Schranke für das betrachtete Problem, denn ein beliebiges Problem der Klasse zu lösen ist nicht schwieriger, als dieses eine Problem zu lösen. Gehört dieses Problem außerdem zur Klasse, dann ist es für sie „vollständig“.

Um die Komplexität zweier Probleme miteinander vergleichen zu können, führen wir nun den Begriff der Reduzierbarkeit ein, aus dem sich dann die o. g. Begriffe der Härte und der Vollständigkeit bezüglich dieser Reduzierbarkeit ergeben. Intuitiv versteht man unter einer Reduktion eines Entscheidungsproblems A auf ein anderes Entscheidungsproblem B , dass man sämtliche Instanzen von A in effizienter Weise in Instanzen von B transformieren kann, sodass die gegebenen Instanzen genau dann Ja-Instanzen von A sind, wenn ihre Transformationen Ja-Instanzen von B ergeben. Dieser Reduzierbarkeitsbegriff ist nur einer unter vielen, aber vielleicht der wichtigste, und er wird als *polynomialzeit-beschränkte Many-one-Reduzierbarkeit* bezeichnet, weil verschiedene Instanzen von A auf ein und dieselbe Instanz von B abgebildet werden können (siehe z. B. Rothe, 2008; Papadimitriou, 1995, für weitere Details und andere Reduzierbarkeiten).

Definition 1.2 (Reduzierbarkeit, Härte, Vollständigkeit und Abschluss)

Ein *Alphabet* ist eine endliche, nicht leere Menge von Buchstaben (oder Symbolen). Σ^* bezeichnet die Menge aller Wörter über dem Alphabet Σ . Eine totale Funktion $f : \Sigma^* \rightarrow \Sigma^*$ heißt *polynomialzeit-berechenbar*, falls es einen Algorithmus gibt, der bei Eingabe eines beliebigen Wortes $x \in \Sigma^*$ den Funktionswert $f(x)$ in Polynomialzeit berechnet. FP bezeichne die Menge aller polynomialzeit-berechenbaren Funktionen.

Seien A und B zwei Entscheidungsprobleme, die über demselben Alphabet Σ codiert sind, d. h., $A, B \subseteq \Sigma^*$. Sei \mathcal{C} eine Komplexitätsklasse.

1. $A \leq_m^P B$ genau dann, wenn es eine Funktion $f \in \text{FP}$ gibt, sodass für alle $x \in \Sigma^*$ gilt: $x \in A \iff f(x) \in B$. Man sagt dann, dass *sich A in Polynomialzeit mittels der Reduktion f auf B (many-one-)reduzieren lässt*.
2. B ist \leq_m^P -*hart* für \mathcal{C} (oder kurz \mathcal{C} -*hart*), falls $A \leq_m^P B$ für jede Menge $A \in \mathcal{C}$ gilt.
3. B ist \leq_m^P -*vollständig* für \mathcal{C} (oder kurz \mathcal{C} -*vollständig*), falls $B \in \mathcal{C}$ und \mathcal{C} -*hart* ist.
4. \mathcal{C} ist *abgeschlossen unter der \leq_m^P -Reduzierbarkeit*, falls für beliebige zwei Probleme A und B aus $A \leq_m^P B$ und $B \in \mathcal{C}$ folgt, dass $A \in \mathcal{C}$ ist.



Cook (1971) bewies, dass SAT NP-vollständig ist, indem er die Berechnung eines beliebigen NP-Algorithmus bei einer beliebigen Eingabe in eine boolesche Formel codierte, die genau dann erfüllbar ist, wenn der Algorithmus seine Eingabe

be akzeptiert. Diese \leq_m^P -Reduktion eines beliebigen NP-Problems auf SAT zeigt die NP-Härte von SAT und somit eine untere Schranke für SAT. Dass SAT zu NP gehört – und somit auch eine entsprechende obere Schranke für das Problem existiert – ist sehr einfach zu sehen. Bei Eingabe einer booleschen Formel φ mit n Variablen „rät“ ein NP-Algorithmus nichtdeterministisch eine Belegung für φ und testet anschließend deterministisch, ob die geratene Belegung die Formel erfüllt. Hier nutzt man die Macht des Nichtdeterminismus aus. In nichtdeterministischer Polynomialzeit lässt sich jede der 2^n möglichen Belegungen raten und deterministisch verifizieren, wobei der entsprechende Berechnungsbaum genau 2^n Pfade hat, deren Länge polynomiell in n ist. Gibt es mindestens einen akzeptierenden Pfad in diesem Baum (also mindestens eine erfüllende Belegung), so wird die Formel akzeptiert; andernfalls wird sie abgelehnt.

SAT ist das erste natürliche Problem, dessen NP-Vollständigkeit gezeigt werden konnte, und daher ist das o. g. Resultat von Cook (1971) ein Meilenstein der Komplexitätstheorie. Der Begriff der NP-Vollständigkeit hat einen unmittelbaren Bezug zum oben erwähnten „ $P = NP?$ “-Problem, wie das folgende Lemma zeigt, das auch einige andere grundlegenden Eigenschaften der \leq_m^P -Reduzierbarkeit sowie der Komplexitätsklassen P und NP auflistet.

Lemma 1.1

1. P und NP sind \leq_m^P -abgeschlossen.
2. Gilt $A \leq_m^P B$ und ist B in P, so ist auch A in P.
3. Gilt $A \leq_m^P B$ und ist A NP-hart, so ist auch B NP-hart.
4. $P = NP$ gilt genau dann, wenn SAT in P liegt.

P und NP sind, wie die meisten Komplexitätsklassen, nach der ersten Aussage von Lemma 1.1 unter der \leq_m^P -Reduzierbarkeit abgeschlossen. Das ist eine nützliche technische Eigenschaft, die in vielen Beweisen über diese Klassen ausgenutzt werden kann, u. a. im Beweis der vierten Aussage dieses Lemmas, nach der man das berühmte „ $P = NP?$ “-Problem einfach dadurch lösen könnte, dass man einen Polynomialzeit-Algorithmus für SAT findet. In diesem Sinn repräsentiert SAT – ebenso wie jedes andere NP-vollständige Problem – die ganze Klasse NP. Anstelle von SAT könnte man somit auch ein *beliebiges* anderes NP-vollständiges Problem in dieser Äquivalenz verwenden, denn wegen des \leq_m^P -Abschlusses von P würde die Zugehörigkeit irgendeines NP-vollständigen Problems zu P ganz NP in P hineinziehen, woraus die Gleichheit von P und NP unmittelbar folgen würde. Ähnlich einfach sind die Beweise der anderen Aussagen dieses Lemmas, die ebenfalls unmittelbar aus den Definitionen folgen.

Nach der zweiten Aussage des Lemmas vererben sich obere Schranken bezüglich der \leq_m^P -Reduzierbarkeit nach unten, und nach der dritten Aussage vererben sich untere Schranken bezüglich der \leq_m^P -Reduzierbarkeit nach oben. Deshalb ist die \leq_m^P -Reduzierbarkeit ein sehr nützliches Werkzeug, um einerseits neue obere Schranken (insbesondere neue Polynomialzeit-Algorithmen) und andererseits neue untere Schranken (insbesondere neue NP-Härte-Beweise) zu erhalten. Das

Problem SAT ist in dieser Hinsicht sehr geeignet, um die NP-Härte weiterer Probleme zu zeigen. Ausgehend von einer SAT-Instanz lassen sich nämlich Reduktionen auf viele andere Probleme sehr gut angeben. Noch geeigneter sind bestimmte Einschränkungen dieses Problems. Zum Beispiel liefert die Cook-Reduktion auf SAT sogar eine boolesche Formel in KNF. Somit ist auch die Einschränkung des Erfüllbarkeitsproblems auf Formeln in KNF NP-vollständig. Wenn man ausgehend von diesem Problem die NP-Härte anderer Probleme zeigen will, ist es manchmal sehr praktisch, wenn die gegebene Formel nicht nur in KNF, sondern in 3-KNF ist, wenn also jede Klausel der Formel höchstens drei Literale hat. Dafür muss zunächst aber erst die NP-Härte dieser Einschränkung des Problems nachgewiesen werden, d. h., es muss eine \leq_m^P -Reduktion von z. B. SAT, eingeschränkt auf KNF-Formeln, auf SAT, eingeschränkt auf 3-KNF-Formeln, angegeben werden.

Für eine beliebige gegebene Formel φ in KNF wollen wir also eine äquivalente Formel ψ in 3-KNF konstruieren. Dazu genügt es, jede Klausel von φ , die mehr als drei Literale besitzt, etwa $C = (\ell_1 \vee \ell_2 \vee \dots \vee \ell_k)$ mit $k \geq 4$ Literalen $\ell_1, \ell_2, \dots, \ell_k$, durch eine neue Teilformel C' zu ersetzen, die erstens in 3-KNF und zweitens genau dann erfüllbar ist, wenn C erfüllbar ist. Diese neue Teilformel hat $k - 3$ neue Variablen, y_1, y_2, \dots, y_{k-3} , und besteht aus $k - 2$ Klauseln der folgenden Art:

$$\begin{aligned} C' = & (\ell_1 \vee \ell_2 \vee y_1) \wedge (\neg y_1 \vee \ell_3 \vee y_2) \wedge \dots \wedge \\ & (\neg y_{k-4} \vee \ell_{k-2} \vee y_{k-3}) \wedge (\neg y_{k-3} \vee \ell_{k-1} \vee \ell_k). \end{aligned}$$

Es ist nicht schwer zu sehen, dass C' erfüllbar ist, wenn C erfüllbar ist, denn eine erfüllende Belegung für C kann folgendermaßen zu einer erfüllenden Belegung für C' erweitert werden:

- Ist die Klausel $(\ell_1 \vee \ell_2 \vee y_1)$ unter der erfüllenden Belegung für C wahr, weil diese ℓ_1 oder ℓ_2 erfüllt, so erweitern wir sie so, dass alle Variablen y_i mit 0 belegt werden. Offenbar ist dann jede Klausel von C' erfüllt.
- Ist die Klausel $(\neg y_{k-3} \vee \ell_{k-1} \vee \ell_k)$ unter der erfüllenden Belegung für C wahr, weil diese ℓ_{k-1} oder ℓ_k erfüllt, so erweitern wir sie so, dass alle Variablen y_i mit 1 belegt werden. Offenbar ist dann jede Klausel von C' erfüllt.
- Andernfalls muss irgendeine andere Klausel von C' unter der erfüllenden Belegung für C wahr sein, denn mindestens ein Literal ℓ_j wird ja erfüllt. Sei $(\neg y_{j-2} \vee \ell_j \vee y_{j-1})$ die erste solche Klausel von C' . Erweitern wir diese Belegung für C nun so, dass alle y_i , $1 \leq i \leq j-2$, mit 1 und alle y_i , $j-1 \leq i \leq k-3$, mit 0 belegt werden, so ist offenbar jede Klausel von C' erfüllt.

Andererseits kann jede erfüllende Belegung für C' auf die Variablen eingeschränkt werden, die zu den Literalen $\ell_1, \ell_2, \dots, \ell_k$ gehören, was eine erfüllende Belegung für C liefert. Deshalb ist C' nur dann erfüllbar, wenn auch C erfüllbar ist, die Klausel C ist also äquivalent zur Teilformel C' . Da für jede solche Klausel mit mehr als drei Literalen jeweils andere neue Variablen eingeführt werden, ist die ursprüngliche Formel φ zur so insgesamt konstruierten neuen Formel ψ äqui-

valent. Folglich ist nach der dritten Aussage von Lemma 1.1 auch das Problem 3-SAT NP-vollständig.

Manchmal ist es übrigens auch zweckmäßig, bei einer Reduktion von einer Formel auszugehen, die nicht nur in 3-KNF ist, sondern die zusätzlich die Eigenschaft hat, dass jede Klausel *genau* drei Literale besitzt. Sehen Sie, was man an der oben angegebenen Reduktion noch ändern müsste, um zu zeigen, dass auch diese Einschränkung des Erfüllbarkeitsproblems NP-vollständig ist?

Bereits vor mehr als dreißig Jahren sammelten Garey und Johnson (1979) mehrere hundert NP-vollständige Probleme aus den verschiedensten wissenschaftlichen Bereichen, inzwischen dürften es Tausende, wenn nicht Zehntausende sein (siehe auch Johnson, 1981, die erste einer Reihe von Kolumnen im *Journal of Algorithms*, in denen NP-vollständige Probleme vorgestellt werden).

Das Problem SAT wird uns in verschiedenen Varianten auch später in diesem Buch noch begegnen. Einerseits werden wir seine NP-Vollständigkeit ausnutzen, um die Komplexität anderer Probleme zu bestimmen. Andererseits werden wir uns in Kapitel 5 insbesondere mit booleschen Formeln im Zusammenhang mit der Urteilsfindung beschäftigen.

Doch jetzt wollen wir endlich spielen, wählen und teilen!

Teil I

Erfolgreiches Spielen

2 Nichtkooperative Spiele: Gegeneinander spielen

Übersicht

2.1	Grundlagen	26
2.2	Nash-Gleichgewichte in gemischten Strategien	43
2.3	Schachmatt: Spielbäume in Spielen mit perfekter Information	54
2.4	Full House: Spiele mit unvollkommener Information	65
2.5	Wie schwer ist es, ein Nash-Gleichgewicht zu finden?	83

Spielen ist etwas zutiefst Menschliches,¹ und die Fähigkeit zu spielen ist eng an die Intelligenz des Menschen gebunden,² an sein Vermögen, vorausschauend und strategisch zu denken, unter den gerade möglichen Spielzügen einen für sich besonders vorteilhaften auswählen zu können, mögliche Antwortzüge seiner Gegenspieler vorauszusehen und dabei insgesamt seinen Gewinn zu maximieren. Unter einem Spiel verstehen wir dabei ganz allgemein eine nach festgelegten Regeln verlaufende Interaktion zwischen mehreren Spielern, die jeweils an der Maximierung ihres Gewinns interessiert sind und zu diesem Zweck strategisch vorgehen. Spiele begegnen uns überall, ob als Gesellschafts-, Computer- oder Glücksspiel, ob als Einzel- oder Mannschaftssportart wie Fechten, Fußball oder Eishockey, ob bei der Ausrichtung von Unternehmensstrategien in einer Marktwirtschaft oder bei geopolitischen oder -strategischen Entscheidungen durch Staaten oder „*Global Players*“.

Dass alle Spieler gleichzeitig das Ziel der individuellen Gewinnmaximierung verfolgen, macht den besonderen Reiz des Spielens und die große intellektuelle Herausforderung der Spieltheorie aus, deren Fundamente von Neumann und Morgenstern

¹„Der Mensch spielt nur, wo er in voller Bedeutung des Wortes Mensch ist, und er ist nur da ganz Mensch, wo er spielt“, stellt Friedrich Schiller im 15. seiner *Briefe über die ästhetische Erziehung des Menschen* von 1795 fest.

²„Blödem Volke unverständlich treiben wir des Lebens Spiel“, meint Christian Morgenstern in seinen 1905 erstmals erschienenen *Galgenliedern*.

(1944) in ihrem bahnbrechenden Werk gelegt haben (siehe auch Borel, 1921; von Neumann, 1928).

Eine ganz grundlegende Einteilung von Spielen besteht darin, dass man kooperative Spiele von nichtkooperativen Spielen unterscheidet. In diesem Kapitel behandeln wir die nichtkooperative Spieltheorie. In einem nichtkooperativen Spiel konkurrieren die einzelnen Spieler miteinander: Jeder denkt eigennützig nur an seinen eigenen Gewinn. Dagegen geht es in der kooperativen Spieltheorie u. a. um die Bildung von Koalitionen von Spielern, die zusammenarbeiten, um gemeinsame Ziele umzusetzen. Dieser Theorie werden wir uns in Kapitel 3 zuwenden.

Um die ungeheure Vielfalt und Vielzahl nichtkooperativer Spiele klassifizieren und analysieren zu können, greifen wir im Folgenden unterschiedliche Aspekte und Kriterien (bzw. „Lösungskonzepte“) heraus, ohne dabei einen Anspruch auf Vollständigkeit zu erheben. Beispielsweise kann man Spiele mit perfekter Information (wie z. B. Schach) von solchen mit nur unvollkommener Information (dazu gehören z. B. viele Kartenspiele) unterscheiden. Weiter spielt bei manchen Spielen der Zufall eine Rolle, bei anderen nicht. Es gibt einzügige Spiele, bei denen die Spieler ihren Zug gleichzeitig machen, und es gibt mehrzügige (bzw. sequenzielle) Spiele, bei denen sie nacheinander oder abwechselnd ziehen. Die von den Regeln des Spiels vorgeschriebene Reihenfolge von Zügen (z. B. Gleichzeitigkeit versus Sequenzialität) hat natürlich Auswirkungen darauf, was jeder Spieler in einer bestimmten Spielsituation weiß, und beeinflusst somit seine Aktionen.

Es ist ganz klar, dass sich konkrete Spiele unter mehreren dieser Kriterien einordnen lassen. So ist etwa Schach ein sequenzielles Spiel, das nicht vom Zufall abhängt und bei dem, wie gerade erwähnt, jeder Spieler stets perfekte Information über die aktuelle Spielsituation hat. Poker hingegen ist ein sequenzielles Spiel, bei dem der Zufall die Blätter der Spieler bestimmt und bei dem kein (ehrlicher) Spieler vollkommene Kenntnis der Blätter seiner Mitspieler hat, auch wenn er vielleicht gewisse Schlüsse aus deren Spielverhalten ziehen kann (oder, besser gesagt, ziehen zu können glaubt), und daher nicht weiß, ob sie gerade bluffen oder nicht.

2.1 Grundlagen

Im Folgenden betrachten wir eine Menge $P = \{1, 2, \dots, n\}$ von Spielern; gelegentlich werden wir ihnen auch Namen statt Nummern geben. Wer die Spieler sind und wie viele es gibt, hängt von dem Spiel ab, das gespielt wird. Spieler können ebenso einzelne Personen sein wie Gruppen von Personen (z. B. in Mannschaftssportarten), sie können Computerprogramme, Staaten (bzw. ihre Regierungen), Wirtschaftsunternehmen, Volksgruppen oder Organisationen sein. Ein Spiel ist definiert durch seine *Regeln*, die beschreiben, wie das Spiel durchzuführen ist und was jeder Spieler in welcher Situation tun darf oder muss und was nicht. Auch

sollte stets klar sein, was die einzelnen Spieler über den jeweiligen Spielzustand wissen können, wann ein Spiel zu Ende ist und wer warum wie viel gewonnen hat.

Manche Spiele zeichnen sich durch ein ausgesprochen umfangreiches Regelwerk aus; beispielsweise hat die Internationale Skatordnung 65 Seiten im A5-Format (wobei allerdings viele Seiten dieses Büchleins keine Regeln, sondern andere wissenswerte Informationen rund um das Skatspiel beinhalten). Um die Analyse jedoch möglichst einfach und verständlich zu halten, werden wir uns hier auf relativ einfache Spiele konzentrieren, anhand derer die wesentlichen strategischen bzw. spieltheoretischen Aspekte dennoch gut erläutert werden können.

Im Unterschied zu den Spielregeln stellen die *Strategien der Spieler* vollständige und exakte Handlungsvorschriften für jede mögliche Situation dar, in die sie im Spielverlauf kommen können. Abhängig von der aktuellen Spielsituation hat ein Spieler demnach möglicherweise die Wahl zwischen alternativen Aktionen, die natürlich alle regelkonform sein müssen. Stehen einem Spieler keine Aktionen mehr zur Auswahl, ist das oft (aber nicht immer) gleichbedeutend mit dem Ende des Spiels und der Niederlage des Spielers (z. B. bei einem „Schachmatt!“).

2.1.1 Normalform, dominante Strategien und Gleichgewichte

Anhand eines konkreten Spiels stellen wir die Normalform sowie die Lösungskonzepte dominante Strategien, Pareto-Optima und Gleichgewichte in Spielen vor.

Das Gefangenendilemma

Ein besonders einfaches, aber auch sehr bekanntes und schönes Beispiel für ein Spiel ist das folgende Dilemma, in das zwei Gefangene geraten sind.

Zwei lang gesuchte Verbrecher, im Untergrund-Milieu bekannt als „Smith & Wesson“, sind verhaftet worden. Man wirft ihnen vor, gemeinsam eine Bank ausgeraubt zu haben, kann es aber leider nicht beweisen. Der für die Ermittlung zuständige Kriminalkommissar verhört die beiden einzeln nacheinander und lässt sich zunächst Smith vorführen. Trotz intensiver Befragung schweigt dieser jedoch beharrlich. Also bietet ihm der Kommissar einen Handel an.

„Sie kennen die Höchststrafe für diese Tat, Smith“, sagt der Kommissar, „zehn Jahre hinter Gittern! Wenn Sie aber gestehen, dass Sie und Wesson die Bank ausgeraubt haben, dann kommen Sie wegen guter Zusammenarbeit mit uns auf Bewährung frei, und Wesson muss die zehn Jahre allein absitzen, falls er weiter störrisch ist.“

Smith schweigt.

„Überlegen Sie sich’s bis morgen“, fährt der Kommissar fort. „Ich biete jetzt Wesson denselben Deal an.“

Als Smith abgeführt wird, dreht er sich noch einmal um und fragt: „Was ist, wenn Wesson ein Geständnis ablegt und mich belastet?“

„Das kommt darauf an“, antwortet der Kommissar. „Wenn nur er gesteht und Sie nicht, kommt er auf Bewährung frei und Sie gehen zehn Jahre in den Bau. Wenn Sie beide ein Geständnis ablegen, müssen Sie beide vier Jahre absitzen, weil dann zwar jeder von Ihnen mit uns kooperiert hat, das eine Geständnis für uns aber jeweils weniger wert ist, denn das andere Geständnis allein hätte uns ja auch gereicht.“

„Und wenn wir beide nichts sagen?“

„Ich will ehrlich sein“, erwidert der Kommissar. „Die Beweislage ist zu dünn, als dass wir in diesem Fall die Höchststrafe herausholen könnten. Wenn Sie sich beide weigern, ein Geständnis abzulegen, kriegen wir Sie nur wegen unerlaubten Waffenbesitzes und Widerstands gegen die Staatsgewalt dran. Das macht dann zwei Jahre Knast für Sie beide.“

Die ganze Nacht grübelt Smith in seiner Einzelzelle über dieses Angebot des Kommissars nach. Leider kann er sich nicht mit Wesson absprechen, der ebenfalls in seiner Einzelzelle sitzt und grübelt. Beide wollen ihre Entscheidung rational treffen und sind dabei – als skrupellose Verbrecher, die sie nun einmal sind – nur auf ihren eigenen Vorteil bedacht. Offensichtlich spielen sie ein strategisches Spiel, bei dem beide die Wahl zwischen zwei Strategien haben: ein Geständnis ablegen oder weiter schweigen. Das Ergebnis des Spiels hängt jedoch für jeden der beiden Spieler nicht nur von der eigenen, sondern auch von der Strategie des anderen Spielers ab, und sie müssen beide gleichzeitig ihren Zug machen, ohne zu wissen, wie sich der andere entscheidet.

Tab. 2.1: Das Gefangenendilemma

		Wesson	
		Geständnis	Schweigen
Smith	Geständnis	$(-4, -4)$	$(0, -10)$
	Schweigen	$(-10, 0)$	$(-2, -2)$

Tabelle 2.1 fasst die Spielregeln des Kommissars zusammen. Ein Eintrag $(-k, -\ell)$ bedeutet dabei, dass Smith zu einer Gefängnisstrafe von k Jahren und Wesson zu einer Gefängnisstrafe von ℓ Jahren verurteilt wird. Ihren Gewinn maximieren die Spieler also dann, wenn sie mit einer möglichst geringen Strafe davonkommen. Möchte man negative Gewinnbeträge vermeiden, so könnte man alle Gewinne gleichermaßen skalieren, ohne dadurch die strategischen Aspekte des

Spiele zu ändern. Halbiert man etwa die Beträge in Tabelle 2.1 und addiert 5, so erhält man die Werte in Tabelle 2.2, die in strategischer Hinsicht äquivalent sind.

Tab. 2.2: Das Gefangenendilemma ohne negative Einträge

		Wesson	
		Geständnis	Schweigen
Smith	Geständnis	(3, 3)	(5, 0)
	Schweigen	(0, 5)	(4, 4)

Allgemein betrachtet man für nichtkooperative Spiele mit einer beliebigen Anzahl von Spielern die *Normalform* (oder *strategische Form*), die für das Gefangenendilemma oben bereits angegeben wurde. Für mehr als zwei Spieler genügt allerdings die einfache zweidimensionale Tabellenform nicht mehr, um alle Gewinnvektoren für alle Tupel von Strategien der Spieler darzustellen. Die Repräsentation in Normalform ermöglicht für eine Vielzahl von Spielen – wenn auch nicht für die Gesamtheit aller Spiele – eine einheitliche Darstellung und Analyse. Die Normalform, die Borel (1921) und von Neumann (1928) zugeschrieben wird, ist am besten dafür geeignet, einzügige Spiele darzustellen, bei denen alle Spieler ihren Zug gleichzeitig und ohne Kenntnis der Züge der anderen Spieler machen und bei denen (externer) Zufall keine Rolle spielt. Für die Darstellung sequenzieller Spiele, bei denen die Spieler nacheinander oder abwechselnd ziehen, ist die so genannte *erweiterte Form* besser geeignet, die wir in Abschnitt 2.3.1 vorstellen. Ein- oder mehrzügige Spiele, bei denen auch noch der Zufall eine Rolle spielt, werden als so genannte *Bayes'sche Spiele* bezeichnet und in Abschnitt 2.4.2 näher erläutert.

Definition 2.1 (Normalform)

Ein Spiel mit n Spielern ist in *Normalform*, falls für alle i , $1 \leq i \leq n$, gilt:

1. Dem Spieler i steht eine (endliche oder unendliche) Menge S_i von (reinen) Strategien (bzw. Aktionen) zur Auswahl. Die Menge der *Profile (reiner) Strategien (oder Aktionen)* aller n Spieler wird dargestellt als das kartesische Produkt

$$\mathcal{S} = S_1 \times S_2 \times \cdots \times S_n.$$

2. Die Gewinnfunktion $g_i : \mathcal{S} \rightarrow \mathbb{R}$ gibt den Gewinn $g_i(\vec{s})$ des Spielers i für das Strategieprofil $\vec{s} = (s_1, s_2, \dots, s_n) \in \mathcal{S}$ an. Dabei bezeichne \mathbb{R} die Menge der reellen Zahlen und s_j , $1 \leq j \leq n$, die von Spieler j gewählte Strategie.



In Definition 2.1 wird von „reinen“ Strategien gesprochen, da wir dort annehmen, dass sich jeder Spieler definitiv für genau eine Strategie entscheidet. Alternativ dazu kann man auch *gemischte* Strategien betrachten, bei denen eine Wahrscheinlichkeitsverteilung auf der Menge S_j der möglichen Strategien eines jeden Spielers j , $1 \leq j \leq n$, angenommen wird. Gemäß dieser Verteilung wählt jeder

Spieler dann seine Strategie mit einer bestimmten Wahrscheinlichkeit. Dies wird in Abschnitt 2.2 genauer erörtert.

Doch wie soll sich Smith nun in seinem Dilemma entscheiden? Könnte er sich darauf verlassen, dass Wesson schweigt, so wäre es für ihn, Smith, natürlich am besten, den Bankraub zu gestehen, denn eine Bewährungsstrafe ist besser als zwei Jahre Gefängnis. Natürlich kann er sich auf Wessons Schweigen nicht verlassen. Andererseits, auch wenn Wesson ein Geständnis ablegt und ihn mitbelastet, ist es für Smith besser, zu gestehen. Ins Gefängnis muss er dann zwar auf jeden Fall, aber vier Jahre sind besser als zehn. Aus Smith' Sicht ist ein Geständnis also für jede Strategie, die Wesson wählt, besser als weiter zu schweigen. Das heißt, zu gestehen ist für Smith eine dominante Strategie. Aus symmetrischen Gründen ist auch für Wesson ein Geständnis eine dominante Strategie. Allgemein ist dieser Begriff wie folgt definiert.

Definition 2.2 (dominante Strategie)

Sei $\mathcal{S} = S_1 \times S_2 \times \cdots \times S_n$ die Menge der Strategieprofile der n Spieler eines nicht-kooperativen Spiels in Normalform und sei g_i die Gewinnfunktion des Spielers i , $1 \leq i \leq n$. Eine Strategie $s_i \in S_i$ des Spielers i heißt *dominant* (oder *schwach dominant*), falls

$$g_i(s_1, \dots, s_{i-1}, s_i, s_{i+1}, \dots, s_n) \geq g_i(s_1, \dots, s_{i-1}, s'_i, s_{i+1}, \dots, s_n) \quad (2.1)$$

für alle Strategien $s'_i \in S_i$ und alle Strategien $s_j \in S_j$ mit $1 \leq j \leq n$ und $j \neq i$ gilt.

Ist die Ungleichung (2.1) echt für alle $s'_i \in S_i$ mit $s'_i \neq s_i$ und alle $s_j \in S_j$ mit $1 \leq j \leq n$ und $j \neq i$, so ist s_i für den Spieler i eine *echt dominante Strategie*. ♦

Die o. g. Strategie, zu gestehen, ist somit sowohl für Smith als auch für Wesson sogar echt dominant. Hat ein Spieler eine (echt) dominante Strategie, so ist dies für ihn natürlich sehr vorteilhaft: Er muss sich gar nicht darum kümmern, was die anderen Spieler tun, sondern besitzt unabhängig von ihnen eine beste Strategie.

Interessanterweise wäre es beim Gefangenendilemma aus *globaler* Sicht jedoch günstiger, wenn *beide* Spieler von ihrer echt dominanten Strategie abweichen, also schweigen würden. Könnten sie sich nämlich absprechen und auf gemeinsames Schweigen einigen, kämen beide für jeweils nur zwei statt für vier Jahre ins Gefängnis. Die dominante Strategie zu spielen und ein Geständnis abzulegen ist jedoch die sicherere Variante, denn selbst wenn sie sich auf gemeinsames Schweigen geeinigt hätten, wäre für beide Spieler die Gefahr einfach zu groß, dass der andere Spieler sein Wort bricht und aus Eigennutz gesteht. Der übers Ohr gehauene Spieler müsste dann die Höchststrafe abbrummen, während der Verräter draußen auf Bewährung herumspazieren könnte.

Wie oben erwähnt, ist die Variante, dass sich beide Spieler für das Schweigen entscheiden, also *kooperieren*, aus globaler Sicht durchaus vorteilhaft. Dies wird formal durch den Begriff der Pareto-Dominanz bzw. der Pareto-Optimalität (oder Pareto-Effizienz; vgl. auch Definition 6.4 auf Seite 248 in Abschnitt 6.3.2) ausge-

- Min-Cost-Flow-Theorem, 214
- Minimax-Theorem, 85
- Mirrlees, J., 330
- Monien, B., 18
- Monty-Hall-Dilemma, 65
- Monty-Hall-Problem, 65
- Morgenbesser, S., 154
- Morgenstern, C., 25
- Morgenstern, O., 3, 25, 72, 75, 100, 101
- Moulin, H., 163–165, 339
- Moving-Knife-Protokoll, 252
 - von Austin, 256
 - von Brams, Taylor und Zwicker, 289
 - von Dubins und Spanier, 262
 - von Stromquist, 287
- Mühle, 54
- Muller, E., 160
 - Unmöglichkeitstheorem von – und Satterthwaite, 160
- Multiagent Resource Allocation, 8, 323, 333
- Mundhe, M., 171
- N**
- N, 13
- Nagel, R., 43
- Naor, M., 171
- Narodytska, N., 180, 188
- Nasar, S., 32, 54
- Nash, J., 3, 32, 51, 53, 54, 83
- NASH EQUILIBRIUM, 90
- NASH PRODUCT SOCIAL WELFARE OPTIMIZATION_{form}, 339, 340
- Nash-Gleichgewicht
 - Bayes'sches, 76
 - für risikofreudige Spieler, 77
 - für risikoneutrale Spieler, 77
 - für risikoscheue Spieler, 77
 - Existenz eines –s, 50, 53
 - in gemischten Strategien, 44, 53
 - in reinen Strategien, 32
 - Komplexität von –en
 - in Normalform-Spielen, 86
 - in Nullsummenspielen, 84
 - symmetrisches, 48
 - unglaubliches, 63
- Negation, 14
- Neidfreiheit, 244, 245
 - starke, 245
- Nguyen, N., VI, 340
- Nguyen, T., VI
- Nicht-Diktatur, 152, 153, 169
- Nicolo, A., 255
- Niedermeier, R., 133, 171, 173, 180
- Nim, 57
 - Gewinnstrategie für, 59
- No-Show-Paradoxon, 163, 164
 - starke Variante des –s, 163
- Nonuniform Bribery, 214
- Nowak, M., 148, 202, 203
- NP, 11
- NP-Orakel, 174
- NP-Orakelmaschine, 229
- NP-vollständig, 19
- NPSWO_{form}, 339
- nucleolus, 100
- Nullsummenspiel, 79, 84, 85
 - Nash-Gleichgewicht in einem, 84
- Nutzfunktion, 333
- O**
- $o(\cdot)$, 13
- $\mathcal{O}(\cdot)$, 13
- Objekt, 323
- Ogihara, M., 9
- Online-Voting, 195
- open-cry auction, 327
- Operation
 - boolesche, 14
- OPTIMAL LOBBYING, 214
- Ordnung
 - lineare, 123
 - partielle, 176
- Orlin, J., 171, 180, 182
- Osborne, M., 99, 100
- Öztürk, M., 189
- P**
- $\mathfrak{P}(\cdot)$, 95, 122, 334
- P, 11
- P^{NP} , 174
- P_{\parallel} , 174
- $P = \text{NP}$?-Frage, 12
- packing, 196
- Padget, J., 9, 323, 334, 335, 339, 340
- Papadimitriou, C., 9, 18, 19, 86, 87, 90, 91, 117, 174, 227, 229, 340
- Papstwahl, 129
- parallel-universes tie-breaking, 175
- paralleler Zugriff auf NP, 174
- Pareto-Dominanz, 31
 - starke, 31
- Pareto-Effizienz, 30
- Pareto-Konsistenz, 153, 169
- Pareto-Optimalität, 31, 248
 - schwache, 31
- Pareto-Optimum, 31
- partielle Ordnung, 176
- PARTITION, 114, 185
- partizipation, 164
- Pasechnik, D., VII, 115, 116
- Paterson, M., 117, 118
- path-disruption game, 118
- Patty, J., 189
- Pauly, M., 223
- Paz, A., 271, 304, 309–311, 313, 314, 319
- Pěchouček, M., 118
- Peleg, B., 100

- Penn, E., 189
 Penrose, L., 111
 Permutation, 109
 Pettit, P., 7, 218, 219, 223
 Phelps, S., 9, 323, 334, 335, 339, 340
 PI-BENEFICIAL MERGE, 117
 PI-BENEFICIAL SPLIT, 118
 Piras, L., V, VI, 193, 203, 206, 207, 210
 pivotal player, 108
 PLS, 87
 Pluralität, 124
 Pluralitätsregel, 124, 148, 169, 183, 201
 Gewinner der, 124
 Poker, 72, 75, 76
 Pokerface, 83
 Politikverdrossenheit, 163
 Polynomialzeit
 deterministische, 11
 nichtdeterministische, 11
 Polynomialzeit-Hierarchie, 229
 Porat, E., 118
 Porello, D., 7, 226–229
 Portion, 235
 zusammenhängende, 265
 POSSIBLE WINNER, 177
 Potenzmenge, 122
 power index, 108
 PP, 117
 PPA, 87
 PPAD, 87
 PPAD-hart, 90
 PPAD-vollständig, 90
 PPP, 88
 Präferenz, 121, 122, 333
 kardinale, 334
 ordinale, 334
 Präferenzaggregation, 121
 Präferenzliste
 lineare, 123
 partielle, 176
 totale Erweiterung einer $-n$ –, 177
 Präferenzprofil, 122
 single-peaked, 188, 189
 Präferenzrelation, 123
 partielle, 176
 Prasad, K., 117
 Prinzip vom ausgeschlossenen Dritten, 220
 Prinzip vom ausgeschlossenen
 Widerspruch, 220
 Pritchard, G., 188
 PROBABILISTIC LOBBYING, 214
 probabilistic polynomial time, 117
 Problem
 Härte eines $-s$, 19
 Komplexität eines $-s$, 11
 obere Schranke der, 10, 12
 untere Schranke der, 10, 13, 19
 Reduzierbarkeit zwischen $-en$, 19
 Vollständigkeit eines $-s$, 19
 Procaccia, A., 9, 180, 187, 188, 209, 305, 306
 Profil reiner Strategien
 Menge der $-e$ –, 29
 Proportionalität, 240
 Pruhs, K., 305, 306, 315
 Puppe, C., 210, 224
- Q**
 \mathbb{Q} , 334
 Quote, 224
- R**
 \mathbb{R} , 29, 85
 \mathbb{R}^+ , 95
 Raghavan, P., 18
 Rebel Without a Cause, 37
 Recommender-System, 171
 Reichstagswahl, 163
 Relation
 asymmetrische, 123
 reflexive, 334
 totale, 123, 334
 transitive, 123, 334
 Ressource, 333
 Aufteilung einer, 233
 Aufteilung von $-n$, 323
 heterogene, 235
 homogene, 235
 teilbare, 233
 Aufteilung einer $-n$ –, 233
 unteilbare, 323
 Aufteilung von $-n$ $-n$, 323
 Rey, A., V, VI, 112, 118
 Rieck, C., 71
 Riege, T., 17, 340
 Robertson, J., 235, 239, 241, 253, 265, 281, 282, 287, 294, 300, 303–306, 314, 319, 321
 Rodríguez-Aguilar, J., 9, 323, 334, 335, 339, 340
 Rogers jr., H., 9
 Rognlie, M., 175
 Roos, M., V, VI, 187, 338, 340
 Rosamond, F., VI, 173, 214
 Rosenschein, J., VI, 112, 115, 116, 118, 171, 180, 187, 188, 209
 Roth, A., 108
 Rothe, E., VII
 Rothe, I., V, 13, 55
 Rothe, J., V, VI, 9, 13, 16, 17, 19, 55, 112, 115, 116, 118, 129, 132, 133, 148, 162, 173–175, 178, 182, 187–189, 191–193, 199–204, 206, 207, 210, 213, 214, 227, 229, 230, 269, 315, 318, 319, 321, 338, 340
 Rothe, P., VII
 Rubinstein, A., 99, 100

Rückwärtsauktion, 329

Rückwärtsinduktion, 63

Rüdinger, A., V, VII

S

Saari, D., 143

Sachs, R., 66

Sachverständiger, 219

Sager, L., 7, 218

Sandholm, T., 91, 180, 181, 183, 186, 323, 332, 335

Santi, P., 335

Sanver, R., VI, 143, 146–148, 151, 202, 203

SAT, 14

k -SAT, 16

SAT-Solver, 17

SATISFIABILITY, 14

Satterthwaite, M., 6, 156, 160

Unmöglichkeitstheorem von Muller und –, 160

Scarf, H., 91

Algorithmus von, 91

Schach, 54

Schachcomputer, 56

Schadrack, H., VI

Schaltkreis

boolescher, 17

Scheidungsformel, V, 325

Scheuermann, B., VII

Schiermeyer, I., 18

Schiller, F., 25

Schlacht der Geschlechter, 2, 35, 48

Schnoor, H., 180, 183

Schöning, U., 18, 174

Schwartz, T., 156

Scoring-Protokoll, 124, 148, 212

Gewinner in einem, 124

Scoring-Regel, 124

Scoring-Vektor, 124

sealed-bid auction, 328

second-price auction, 328

Selfridge, J., 282, 284, 285

Selfridge-Conway-Protokoll, 282

Selman, A., 9, 87, 182

Selten, R., 32, 43, 64

Sen, A., 6

Sen, S., 171

Sewelson, V., 340

Sgall, J., 305, 306

Shapley, L., 100–102, 108, 110–112

Shapley-Shubik(\cdot, \cdot), 109

Shapley-Shubik $^*(\cdot, \cdot)$, 109

SHAPLEY-SHUBIK, 117

SHAPLEY-SHUBIK-POWER-COMPARE, 117

Shoham, Y., 323, 337

Shubik, M., 100, 108

Simplexmethode, 84

Simpson, P., 132

Wahlssystem von, 132, 148, 183

Simpson-Gewinner, 132

Simpson-Score, 132

Sincere-Strategy Preference-Based

Approval Voting, 146

Single Transferable Vote, 138

single-item auction, 327

Sivakumar, D., 171

Slinko, A., VI, 172, 188, 210, 213, 214

Social-Choice-Funktion, 123

Surjektivität einer, 155

Social-Choice-Korrespondenz, 122

Social-Choice-Theorie, 6

Social-Welfare-Funktion, 123

Sotheby's, 329

Sousa, P., 9, 323, 334, 335, 339, 340

Souveränität der Bürger, 155, 169

soziale Wohlfahrt

egalitaristische, 338

nach dem Nash-Produkt, 338

Unabhängigkeit von unbeteiligten

Agenten für die, 339

utilitaristische, 338

Sozialwahltheorie, 6

SP-AV, 146, 148, 201

Spakowski, H., 133, 135, 162, 173–175, 187, 188

Spanier, E., 262, 263, 276, 296, 307, 320

Speckenmeyer, E., 18

Sperner, E., 90

Lemma von, 90

Spiel

Bayes'sches, 72, 74

kombinatorisches, 56

kooperatives, 93

ϵ -Kern eines $-n$ –s, 100

angepasstes, 115

Auszahlungsvektor eines $-n$ –s, 97, 99

Banzhaf-Index in einem einfachen $-n$ –, 111, 112

charakteristische Funktion eines $-n$ –s, 95

CS-Kern eines $-n$ –s, 116

effizientes, 99

einfaches, 103

Ergebnis eines $-n$ –s, 97

externe Stabilität in einem $-n$ –, 101

große Koalition eines $-n$ –s, 96

individuell rationales, 99

interne Stabilität in einem $-n$ –, 101

Kern eines $-n$ –s, 98, 99, 102

kleinster Kern eines $-n$ –s, 100

Koalitionsstruktur eines $-n$ –s, 96

konvexes, 101, 102

Kosten der Stabilität in einem $-n$ –, 115

kostengünstigste stabile

Koalitionsstruktur in einem $-n$ –, 116

- Machtindex in einem einfachen $-n$ –, 108
 - mit nicht übertragbarem Gewinn, 97
 - mit übertragbarem Gewinn, 95
 - monotone charakteristische Funktion eines $-n$ $-s$, 95
 - normalisierte charakteristische Funktion eines $-n$ $-s$, 95
 - Penrose-Banzhaf-Index in einem einfachen $-n$ –, 111
 - probabilistischer Banzhaf-Index in einem einfachen $-n$ –, 112
 - roher Banzhaf-Index in einem einfachen $-n$ –, 111
 - roher Shapley-Shubik-Index in einem einfachen $-n$ –, 109
 - Scheinspieler in einem einfachen $-n$ –, 108
 - Schlüsselspieler in einem einfachen $-n$ –, 108
 - Shapley-Shubik-Index in einem einfachen $-n$ –, 108, 109
 - Shapley-Wert eines $-n$ $-s$, 100
 - Shapley-Wert in einem $-n$ –, 110, 111
 - stabile Menge eines $-n$ $-s$, 100, 101
 - superadditives, 98
 - supermodulare charakteristische Funktion eines $-n$ $-s$, 101
 - Teilspiel eines $-n$ $-s$, 103
 - Veto-Spieler in einem einfachen $-n$ –, 103
 - nichtkooperatives, 25, 29
 - in erweiterter Form, 54
 - mit perfekter Information, 55, 74
 - mit unvollkommener Information, 65
 - mit vollkommener Information, 74
 - mit vollständiger Information, 73
 - Normalform eines $-n$ $-s$, 27, 29
 - Regel eines $-s$, 26
 - Struktur eines, 73
 - Spielbaum, 55
 - Blatt eines $-s$, 55
 - Entscheidungsknoten eines $-s$, 55
 - Gleichgewicht in einem, 61
 - Wurzel eines $-s$, 55
 - Spieler
 - Bewertungsfunktion eines $-s$, 235
 - Additivität der, 236
 - Boxendarstellung der, 255
 - Nichtnegativität der, 235
 - Normalisierung der, 235
 - Positivität der, 235
 - stetige, 257
 - Teilbarkeitseigenschaft der, 236
 - Gewinn eines $-s$, 29
 - Koalition von $-n$, 93, 95
 - marginaler Beitrag eines $-s$, 102
 - Portion eines $-s$, 235
 - risikofreudiger, 74
 - risikoneutraler, 74
 - risikoscheuer, 74, 251
 - Strategie eines $-s$, 27, 238
 - symmetrische, 110
 - Typ eines $-s$, 73, 74
 - unaufhebbarer Vorteil eines $-s$, 284
 - Zug eines $-s$, 54
 - Spieltheorie
 - kooperative, 26, 93
 - nichtkooperative, 26, 93
 - SScore*(\cdot), 132
 - Stabilität einer Lösung, 31
 - Stein-Schere-Papier-Spiel, 40, 47
 - Steinberg, R., 337
 - Steinhaus, H., 8, 234, 258, 265, 268, 269, 275, 282, 296, 319
 - Still, M., 66
 - Stockmeyer, L., 229
 - Strategie
 - dominante, 27, 30
 - echt dominante, 30
 - gemischte, 29, 44
 - reine, 29
 - schwach dominante, 30
 - Stromquist, W., 265, 282, 287–289, 294, 320
 - STV, 138, 148, 169, 183
 - Suchproblem, 86
 - SUPER-IMPUTATION STABILITY, 116
 - Super-Neidfreiheit, 244, 245
 - Sviridenko, M., 340
 - SWAP-BRIBERY, 213
 - Swap-Bribery-Preisfunktion, 213
 - systematicity, 222
- T**
- Tambe, M., 118
 - Tasnádi, A., 210
 - Tautologie, 219
 - TAUTOLOGY, 227
 - Taylor, A., 154, 235, 238, 239, 253, 266, 279, 281, 282, 287, 289, 290, 305, 321, 324–326
 - TE, 193
 - Teile-und-herrsche-Prinzip, 271, 302, 306
 - Teilnahme-Kriterium, 163, 164, 169
 - Thapa, M., 84, 85
 - threshold network flow game, 118
 - Tic-Tac-Toe, 56
 - Spielbaum für, 58
 - Tideman, N., 131, 157
 - tie-breaking rule, 137
 - Tierney, J., 67
 - ties eliminate, 193
 - ties promote, 194
 - Tovey, C., 171–173, 178, 179, 190–192, 196, 200, 201
 - TP, 194

Trick, M., 171–173, 178, 179, 190–192, 196, 200, 201
 Trimming Algorithm, 300
 Turing, A., 10
 Turingmaschine, 10
 deterministische, 10
 Endkonfiguration einer, 11
 ablehnende, 11
 akzeptierende, 11
 Konfiguration einer, 10
 nichtdeterministische, 11
 Berechnungsbaum einer $-n-$, 11
 Rechenzeit einer, 10, 11

U

Überproportionalität, 240, 241
 Übertragbare Einzelstimmgebung, 138
 Gewinner bei der $-n-$, 138
 UCO, 188
 UEFA Champions League, 138
 Uhlmann, J., 173
 UN-Sicherheitsrat, 106
 Unabhängigkeit von irrelevanten
 Alternativen, 153, 169
 Unabhängigkeit von Klonen, 157
 Unique Weak Axiom of Revealed
 Preference, 200
 Unique-WARP, 199, 200
 UNWEIGHTED COALITIONAL
 OPTIMIZATION, 188
 Urteilsmenge, 220
 individuelle, 220
 kollektive, 220
 Komplementfreiheit einer, 220
 Konsistenz einer, 220
 Vollständigkeit einer, 220
 US-Präsidentenwahl 2000, 191, 192
 USWO_{form}, 339
 utilitarian social welfare, 338
 UTILITARIAN SOCIAL WELFARE
 OPTIMIZATION_{form}, 339

V

Valiant, L., 117
 van Hees, M., 223
 van Melkebeek, D., 18
 Vaněk, O., 118
 Veto-Regel, 124, 148, 183
 Gewinner der, 124
 Vickrey, W., 330, 337
 Vickrey–Clarke–Groves-Mechanismus, 337
 Vickrey-Auktion, 330
 Vier gewinnt, 56
 Viertel-Protokoll
 für drei Spieler, 308
 für vier Spieler, 311
 Viglas, A., 18
 Vogel, J., 133, 135, 162, 173–175

Volkskammer-Wahl, 163
 von Neumann, J., 3, 25, 29, 32, 54, 72, 75–77, 82, 84, 100, 101
 vereinfachte Poker-Variante von, 75
 Simplifizierung der $-n-$, 76
 von Randow, G., 65, 67
 Vorzugsregel, 137
 randomisierte, 170
 vos Savant, M., 66, 67
 voting tree, 138

W

Wagner, K., 9, 174, 340
 Wahl, 122
 Bestechung in einer, 210
 Gewinner einer
 eindeutiger, 122
 möglicher, 175, 176, 186
 nicht eindeutiger, 122
 notwendiger, 175, 178
 Manipulation einer, 178, 180, 183, 185–187
 destruktive, 181
 durch eine Koalition von
 Manipulatoren, 180
 für single-peaked Präferenzprofile, 188
 konstruktive, 181
 mit gewichteten Wählern, 180
 Manipulation einer, 180
 mit irrationalen Wählern, 214
 Wahlbaum, 138
 Wahlbeteiligung, 163
 Wahlkontrolle, 190, 200
 Anfälligkeit für, 198
 destruktive, 191
 durch Entfernen von Kandidaten, 191
 durch Entfernen von Wählern, 194
 durch Hinzufügen von Kandidaten, 191
 durch Hinzufügen von Wählern, 194
 durch Partitionieren der
 Kandidatenmenge, 193
 durch Partitionieren der Wählerliste, 196
 für single-peaked Präferenzprofile, 209
 Immunität gegen, 198
 konstruktive, 191
 Resistenz gegen, 198
 Übersicht über Probleme der, 197
 Verletzbarkeit durch, 198
 zertifizierbare, 206
 Wahlleiter, 190
 Wahlsystem, 121, 122
 Anfälligkeit eines $-s$ für einen
 Kontrolltyp, 198
 Anonymität eines $-s$, 157, 158, 169
 auf paarweisen Vergleichen beruhendes, 125, 148
 diktatorisches, 152

Gewinnerbestimmung in einem, 172
 Homogenität eines –s, 161, 169
 hybrides, 142, 148
 Immunität eines –s gegen einen
 Kontrolltyp, 198
 Konsistenz eines –s, 166, 167, 169
 Manipulierbarkeit eines –s, 155
 mehrstufiges, 136, 148
 Monotonie eines –s, 158, 159, 169
 strenge, 159
 Neutralität eines –s, 158, 169
 reines, 142
 Resistenz eines –s gegen einen
 Kontrolltyp, 198
 Resolutheit eines –s, 155
 Strategiesicherheit eines –s, 155, 169
 Verletzbarkeit eines –s durch einen
 Kontrolltyp, 198
 zertifizierbare, 206
 voiced, 199
 Wahrheitswert, 15
 Wahrscheinlichkeit, 69
 bedingte, 69
 totale, 69
 Gesetz der –n –, 68, 70
 Wahrscheinlichkeitsraum, 69
 Wahrscheinlichkeitsverteilung, 69
 Walsh, T., VI, 178, 180, 188, 189, 209
 Wang, J., 187
 Wanke, E., VII, 13, 55
 WARP, 200
 Waters, J., 37
 Weak Axiom of Revealed Preference, 200
 Webb, W., 235, 239, 241, 246, 253, 265,
 281, 282, 287, 294, 300, 303–306,
 314, 319, 319
 Wechsung, G., 9, 340
 Wegener, I., 9
 weighted majority game, 104
 weighted threshold game, 104
 weighted voting game, 104
 unanimous, 118
 Weller, D., 249
 Werbekampagnenspiel, 62
 Spielbaum für das, 63
 Widerspruch, 220
 WINNER, 172
 winner's curse, 329
 winner-turns-loser paradox, 159
 Woeginger, G., 17, 180, 305, 306
 Wolpert, A., 18
 Woodall, D., 163, 278, 279, 282
 Wooldridge, M., 91, 113, 114, 323, 329,
 332, 335, 337, 340

X

X3C, 206
 XESWO_{form}, 340
 Xia, L., 175, 178, 180

XNPSWO_{form}, 340
 XOR bidding language, 334
 XUSWO_{form}, 340

Y

Yannakakis, M., 340
 Young, H., 133, 162, 173, 175
 homogene Variante des Wahlsystems
 von, 162, 175
 Wahlsystem von, 133, 148, 169, 173
 Young-Gewinner, 133
 Young-Score, 133
 YScore(·), 133
 Yu, Y., 255

Z

Zachos, S., 174
 Zahlen
 Menge der ganzen, 85
 Menge der natürlichen, 13
 Menge der nicht negativen reellen, 95
 Menge der rationalen, 334
 Menge der reellen, 29
 Zahlenwahlspiel, 41
 Zankó, V., 89
 Ziegenproblem, 65
 intuitive Lösungen des –s, 67
 Lösung des –s mit dem Gesetz der
 totalen Wahrscheinlichkeit, 68
 Lösung des –s mit der Formel von
 Bayes, 71
 Zuckerman, M., VII, 115, 116, 118, 180,
 188, 209
 Zufallsvariable
 diskrete, 77
 Erwartungswert einer –n –n, 77
 Zwei-Personen-Spiel, 28, 34
 mehrzügiges, 54
 sequenzielles, 54
 Zwei-Wege-Neidfreiheit, 317
 Zwei-Wege-Neid-Relation, 318
 Zweitpreis-Auktion, 328
 mit verdeckten Geboten, 330
 Zwicker, W., 289
 Zwilling, 165
 Zwillinge-willkommen-Kriterium, 165
 Zwillingsparadoxon, 165

Einführung in Computational Social Choice
Individuelle Strategien und kollektive Entscheidungen
beim Spielen, Wählen und Teilen
Rothe, J.; Baumeister, D.; Lindner, C.; Rothe, I.
2012, XII, 375 S. 89 Abb., Softcover
ISBN: 978-3-8274-2570-6