JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Young-Tableaux und Schur-Polynome

Bachelorarbeit

im
Fachbereich 08 – Physik,
Mathematik und Informatik

---

vorgelegt von:

Nora Alina Henkeler

Matrikelnummer: 2672166

Mainz, den 28.Mai 2013

Erstgutachter	Jun.-Prof. Dr. Nikita Semenov
Zweitgutachter	Prof. Dr. Felix Leinen

Leitfaden

Die vorliegende Bachelorarbeit befasst sich hauptsächlich mit Tableaux und Schur-Polynomen. Ziel ist es, zu beweisen, dass es sich bei den Schur-Polynomen um symmetrische Polynome handelt. Um diese einzuführen, bedienen wir uns der Tableaux. Des Weiteren wird uns die Knuth-Äquivalenz von Worten und die Kostka-Zahl dabei helfen, eine eindeutige Darstellung der Schur-Polynome in elementar- beziehungsweise vollständig-symmetrischen Polynomen zu finden. Mit dieser Vorgehensweise orientieren wir uns grundlegend an William Fultons Buch *Young-Tableaux – With Application to Representation Theory and Geometry* [1].

Vorwort

Gegenstand dieser Bachelorarbeit sind die Schur-Polynome, die auf den weißrussischen Mathematiker Issai Schur zurückgehen. Er wurde am 10. Januar 1875 in Mogiljow geboren, besuchte aber bereits mit 13 Jahren ein deutsch-sprachiges Gymnasium. 1894 schrieb er sich an der Universität Berlin für Mathematik und Physik ein. Er wurde Schüler von Ferdinand Georg Frobenius, bei dem er 1901 *summa cum laude* promovierte. Frobenius war Mitbegründer der Gruppen- und Darstellungstheorie. Schur lernte die Grundlagen von ihm und entwickelte sie weiter. Es ist also nicht verwunderlich, dass auch die Schur-Polynome in der Darstellungstheorie Anwendung finden. Sein wohl bekanntester Beitrag zur Darstellungstheorie ist das *Lemma von Schur*.
Nach seiner Promotion lehrte Schur an den Universitäten in Berlin und Bonn. 1922 wurde er zudem in die Preußische Akademie der Wissenschaften aufgenommen.
Aufgrund seiner jüdischen Wurzeln wurde er jedoch nach der Machtübernahme der Nationalsozialisten stark unter Druck gesetzt und 1935 entlassen. Ein Jahr nach seinem Austritt aus der Akademie wanderte er im Jahr 1939 nach Palästina aus, wo er 1941 an seinem 66. Geburtstag starb.[1]

Quelle: I. Schur, Oberwolfach Photo Collection [8]

Das Ziel dieser Bachelorarbeit wird, wie bereits erwähnt, sein, die Symmetrie von Schur-Polynomen zu zeigen.
Das erste Kapitel werden wir damit beginnen, Young-Tableaux zu definieren. Wir werden den Bumping-Algorithmus von Schensted und den Sliding-Algorithmus von Schützenberger einführen, die es uns erlauben werden, zwei Produkte von Tableaux zu definieren. Mit diesen Produkten wird die Menge der Tableaux jeweils zu einem Monoid.
Im zweiten Kapitel werden wir Worte einführen. Dabei werden wir sehen, dass jedes Tableau ein Wort definiert, aber nicht jedes Wort von einem Tableau kommt. Parallel zu den ersten beiden Produkten werden wir ein weiteres über Knuth-Äquivalenz von Worten einführen und zeigen, dass alle drei Produkte äquivalent sind.

[1] Die Biographie wurde dem *MacTutor History of Mathematics archive citebio* entnommen.

Im dritten Kapitel wird das Theorem dieser Bachelorarbeit bewiesen, indem wir eine Darstellung der Schur-Polynome in Basen der symmetrischen Polynome finden. Abschließend soll der Zusammenhang zwischen Schur-Polynomen und anderen symmetrischen Polynomen an einem Beispiel veranschaulicht werden.

Schur-Polynome finden unter Anderem Anwendung in zwei wichtigen Gebieten der Mathematik. Zum einen ist dies die Darstellungstheorie, wo sie zur Beschreibung der symmetrischen Gruppe $\mathfrak{S}_n$ und der allgemeinen linearen Gruppe $\mathrm{GL}_n(\mathbb{C})$ dienen, zum anderen die Schnitttheorie, wie es Laurent Manivel in der Einleitung zu seinem Buch *Fonctions Symetriques, Polynomes De Schubert Et Lieux De Degenerescence* [3] erwähnt. Näheres kann zum Beispiel in den Werken von Fulton nachgelesen werden: *Young-Tableaux – With Application to Representation Theory and Geometry* [1] zeigte die Anwendung in der Darstellungstheorie, wohingegen *Intersection Theory* [2] die Schnitttheorie thematisiert.

Inhaltsverzeichnis

Kapitel 1

Young-Diagramme und Tableaux

Wir werden zunächst damit beginnen, eine geeignete Grundlage zu schaffen, um über Schur-Polynome reden zu können. Dazu benötigen wir Young-Tableaux. Wir werden den Schensted-Bumping-Algorithmus und den Schützenberger-Sliding-Algorithmus einführen. Diese werden es uns ermöglichen, eine Verknüpfung $\cdot$ von Tableaux zu definieren. Wir werden sehen, dass die Menge der Tableaux zusammen mit $\cdot$ ein Monoid ist.

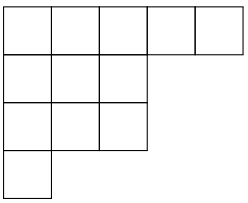
1.1 Notation

Definition 1.1. Ein *Young-Diagramm* sei eine Ansammlung von Kästchen,

1. die linksbündig angeordnet sind und
2. deren Anzahl pro Zeile von oben nach unten schwach monoton fällt.

Eine *Partition von n* sei eine Folge $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_l)$ in den natürlichen Zahlen[2] mit der Eigenschaft, dass $\lambda_1 + \lambda_2 + \ldots + \lambda_l = n$.
Wir wollen eine Partition immer als monoton fallende Folge angeben, da sie dann ein Young-Diagramm mit λ_i Kästchen in der i-ten Zeile ($i = 1, \ldots, l$) und insgesamt n Kästchen liefert.
Mit $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_l)$ assoziieren wir auch das resultierende Young-Tableau und schreiben $\lambda = (d_1^{a_1}, d_2^{a_2}, \ldots, d_s^{a_s})$, falls das Young-Diagramm jeweils a_i aufeinanderfolgende Zeilen der Länge d_i besitzt.

Beispiel 1.2. $\lambda = (5, 3, 3, 1)$ ist eine Partition von 12, das zugehörige Young-Diagramm sieht folgendermaßen aus:



[2]Wir wollen die natürlichen Zahlen in dieser Arbeit ohne die Null definieren, also $\mathbb{N} = \{1, 2, 3, \ldots\}$.

Die Kästchen eines solchen Young-Diagramms λ können auch mit Elementen aus einem beliebigen Alphabet gefüllt werden. Wir verwenden als Alphabet $[m] = \{1, 2, \ldots, m\} \subset \mathbb{N}$ und wollen dies eine *Belegung von* λ nennen. Eine spezielle Belegung kommt in folgender Definition vor.

Definition 1.3. Ein *(Young-)Tableau* T sei ein Young-Diagramm λ mit der folgenden Belegung:

1. In jeder Zeile seien die Einträge der Kästchen monoton steigend angeordnet.
2. Entlang jeder Spalte seien die Einträge streng monoton steigend.

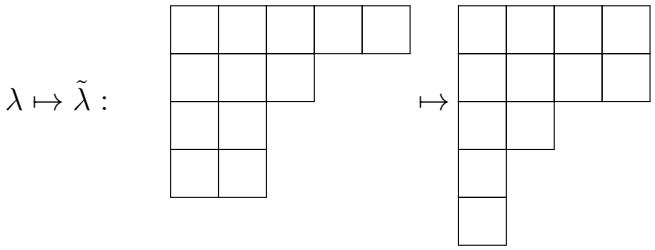
λ heiße *Umriss von* T.

Beispiel 1.4. Man erhält beispielsweise ein Tableau, wenn man das Young-Diagramm aus Beispiel 1.2 wie folgt füllt:

1	1	2	3	4
2	2	3		
3	3	4		
4				

Definition 1.5. Wir definieren das konjugierte Diagramm $\tilde{\lambda}$ zu einem gegebenen Diagramm λ durch Spiegelung von λ an seiner Hauptdiagonalen.

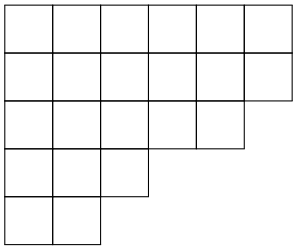
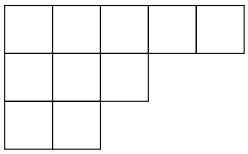
Beispiel 1.6. Beispiel für ein Young-Diagramm und sein konjugiertes Tableau:



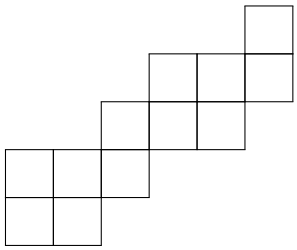
Definition 1.7. Seien λ und μ zwei Young-Diagramme. μ ist in λ *enthalten*, falls alle Kästchen von μ auch zu λ gehören. In diesem Fall schreiben wir $\mu \subset \lambda$.
Legt man ein Young-Diagramm $\mu \subset \lambda$ an die obere linke Ecke des Young-Diagramms λ an und entfernt aus λ diejenigen Kästchen, die auf diese Weise überdeckt werden, so heißt das resultierende Diagramm ein *Schiefdiagramm*. Wir bezeichnen dieses mit λ/μ.
Ein *Schieftableau* sei ein Schiefdiagramm mit demselben Belegungsschema wie ein Tableau:

1. In jeder Zeile seien die Einträge der Kästchen monoton steigend angeordnet.
2. Entlang jeder Spalte seien die Einträge streng monoton steigend.

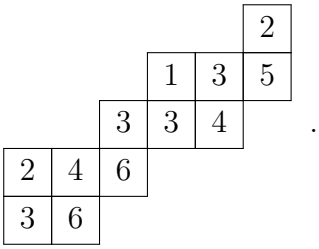
Beispiel 1.8. Haben wir zwei Partitionen $\lambda = (6, 6, 5, 3, 2)$ und $\mu = (5, 3, 2)$ gegeben, so asoziieren wir damit die Diagramme

$\lambda =$  und $\mu =$  .

Das zugehörige Schiefdiagramm ist

$\lambda/\mu =$  .

Dieses Schiefdiagramm könnten wir wie folgt füllen, um ein Schieftableau zu erhalten:



.

Definition 1.9. Ein (Schief-)Tableau habe das *Gewicht* $u = (u_1, u_2, \ldots, u_l)$, wenn es u_i-mal den Eintag i enthält $(i = 1, 2, \ldots, l)$.

Beispiel 1.10. Das Schieftableau aus Beispiel 1.8 hat das Gewicht $u = (1, 2, 4, 2, 1, 2)$.

Kommen wir nun zur Definition derjenigen Polynome, die Gegenstand dieser Arbeit sind:

Definition 1.11. Sei T ein Tableau mit Gewicht $u = (u_1, u_2, \ldots, u_m)$. Dann definiert T ein Monom

$$x^T = \prod_{k=1}^{m} {x_k}^{u_k}.$$

Das *Schur-Polynom* s_λ zu einem Young-Diagramm λ sei definiert via

$$s_\lambda(x_1, x_2, \ldots, x_m) = \sum_{T \text{ Tableau mit Umriss } \lambda} x^T.$$

Zwei Schur-Polynome sind dabei von besonderer Bedeutung:

- Das Schur-Polynom zu $\lambda = (1^n)$ heiße *n-tes elementar-symmetrisches Polynom* σ_n. In diesem Fall schreiben wir

$$s_\lambda(x_1, x_2, \ldots, x_m) = \sum_{1 \le i_1 < i_2 < \cdots < i_n \le m} x_{i_1} \cdot x_{i_2} \cdot \ldots \cdot x_{i_n} =: \sigma_n(x_1, x_2, \ldots, x_m),$$

  wobei $\{i_1, i_2, \ldots, i_n\} \subset [m]$.

- Das *n-te vollständig-symmetrische Polynom* h_n sei definiert als das Schur-Polynom zu $\lambda = (n)$. Wir schreiben $s_\lambda(x_1, x_2, \ldots, x_m) =: h_n(x_1, x_2, \ldots, x_m)$.

Nach Konstruktion ist klar, dass die elementar- und die vollständig-symmetrischen Polynome symmetrisch sind, d.h. sie sind invariant unter Permutation der Variablen: Ist τ eine Permutation in der symmetrischen Gruppe $\mathfrak{S}_m$, so gilt

$$s_\lambda(x_1, x_2, \ldots, x_m) = s_\lambda(x_{\tau(1)}, x_{\tau(2)}, \ldots, x_{\tau(m)}) \qquad \text{für } \lambda = (1^n) \text{ oder } \lambda = (n).$$

Beispiel 1.12.

- Für den Umriss $\lambda = (1^3)$ gibt es vier verschiedene Möglichkeiten, es mit Elementen aus $\{1, 2, 3, 4\}$ zu füllen:

1	1	2	1
2	2	3	3
3	4	4	4

  Das dritte elementar-symmetrische Polynom in den Variablen x_1, x_2, x_3, x_4 ist somit

$$\sigma_3(x_1, x_2, x_3, x_4) = x_1x_2x_3 + x_1x_2x_4 + x_2x_3x_4 + x_1x_3x_4.$$

- Den Umriss $\lambda = (3)$ können wir auf zehn verschiedene Weisen mit den Zahlen aus $\{1, 2, 3\}$ füllen:

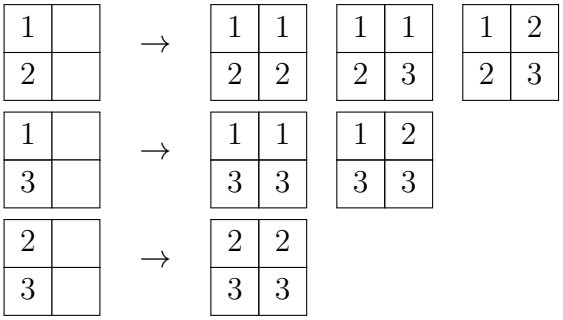
1 1 1	1 1 2	1 1 3	1 2 2	1 2 3
1 3 3	2 2 2	2 2 3	2 3 3	3 3 3

  Wir erhalten daraus

$$h_3(x_1, x_2, x_3) = x_1^3 + x_1^2x_2 + x_1^2x_3 + x_1x_2^2 + x_1x_2x_3 + x_1x_3^2 + x_2^3 + x_2^2x_3 + x_2x_3^2 + x_3^2.$$

- Da die beiden vorherigen Beispiele nur Spezialfälle von Schur-Polynomen waren, wollen wir zuletzt noch ein allgemeines Beispiel geben. Wir betrachten das Tableau $\lambda = (2, 2)$ und füllen dieses mit Elementen aus $\{1, 2, 3\}$. Es ergeben sich die folgen-

den Belegungen:



Das zugehörige Schur-Polynom ist also

$$s_\lambda(x_1, x_2, x_3) = x_1^2x_2^2 + x_1^2x_3^2 + x_2^2x_3^2 + x_1^2x_2x_3 + x_1x_2^2x_3 + x_1x_2x_3^2.$$

Zwei wichtige Eigenschaften der Schur-Polynome sind:

1. Sie sind symmetrisch.
2. Sie bilden eine additive Basis des Rings der symmetrischen Polynome.

Unser Ziel soll es sein, die hier an erster Stelle genannte Eigenschaft zu zeigen. Bevor wir jedoch die Symmetrie beweisen können, müssen wir zunächst noch ein wenig tiefer in die Materie der Tableaux eindringen.

1.2 Schensted-Bumping-Algorithmus

Wir beginnen mit dem Schensted-Bumping-Algorithmus, der es uns später erlauben wird, eine Verknüpfung zweier Tableaux zu definieren. Mit dessen Hilfe wird eine natürliche Zahl x in ein gegebenes Tableau T eingereiht. Das resultierende Tableau wollen wir mit $T \leftarrow x$ bezeichnen. Es wird wie folgt gebildet:

Algorithmus 1.13 (Bumping-Algorithmus)**.** Gegeben seien ein Tableau T und eine natürliche Zahl x. Man betrachte zunächst die erste Zeile des Tableaus. Wenn x größer als oder genauso groß wie der Eintrag des letzten Kästchens ist, wird x in einem neuen Kästchen an das Ende der ersten Zeile gesetzt.
Ist dies nicht der Fall, so suchen wir das am weitesten links gelegene Kästchen, für dessen Eintrag a gilt: $a > x$. Wir schreiben x in dieses Kästchen und verfahren in der nächsten Zeile mit a analog.
Der Algorithmus endet, wenn ein Eintrag an das Ende einer Zeile angefügt werden konnte. Wird das letzte Kästchen der letzten Zeile erreicht ohne dass der Eintrag an dieses angefügt werden kann, wird eine neue Zeile mit einem Kästchen eröffnet, welches Platz für diesen bietet.

Beispiel 1.14. Wir wollen die Zahl 2 in ein Tableau T einfügen. T sei gegeben durch:

1	3	3	4
2	4	4	
4	5	6	

Dazu gehen wir wie folgt vor:

1	**3**	3	4
2	4	4	
4	5	6	

$\longleftarrow 2$

1	2	3	4
2	**4**	4	
4	5	6	

$\longleftarrow 3$

1	2	3	4
2	3	4	
4	**5**	6	

$\longleftarrow 4$

1	2	3	4
2	3	4	
4	4	6	
5			

$T \leftarrow 2$

Blau markiert ist das *neue Kästchen*, die roten Felder zeigen den *Weg*, zu dem auch das neue Kästchen gehört (hier noch einmal deutlicher eingezeichnet):

1	**2**	3	4
2	**3**	4	
4	**4**	6	
5			

Bemerkung 1.15. *Der Output des Bumping-Algorithmus ist wieder ein Tableau:* Jede einzelne Zeile wird so konstruiert, dass ihre Einträge monoton steigen. Verdrängt der Eintrag x, den wir einreihen wollen, also einen Eintrag a, so ist x nach Konstruktion mindestens so groß wie sein Vorgänger und kleiner als sein Nachfolger. Der Eintrag a wird in die nächste Zeile gebracht und verdrängt dort einen Eintrag, der unterhalb oder links von der neuen Position von x liegt. Auf diese Weise bleibt jede Spalte streng monoton steigend, da a größer als x und somit auch größer als jeder Eintrag links von x ist.

Ist ein aus dem Bumping-Algorithmus entstandenes Tableau gegeben und das neue Kästchen bekannt, so kann der Algorithmus auf einfache Weise umgekehrt werden, sodass wir das ursprüngliche Tableau und die eingefügte Zahl zurückgewinnen.

Algorithmus 1.16 (Umkehrung des Bumping-Algorithmus)**.** Sei ein Tableau T und dessen neues Kästchen mit Eintrag y gegeben. (Bemerke: Die Umkehrung funktioniert nur für neue Kästchen. Diese sind immer äußere Ecken, d.h. solche Kästchen, wo rechts und unten keine weiteren Kästchen anschließen.)
Wir entfernen das neue Kästchen und suchen in der Zeile darüber den am weitesten rechts

gelegenen Eintrag b mit $b < y$, schreiben y in dieses Kästchen und verfahren analog mit b in der Zeile darüber.
Der Algorithmus endet, wenn der letzte Eintrag in die erste Zeile eingefügt wurde. Es ist das Tableau T' entstanden. Der verdrängte Eintrag x ist dann derjenige, für den gilt: $T' \leftarrow x = T$.

Im Folgenden betrachten wir zwei sukzessive Einreihungen durch den Bumping-Algorithmus. Wir wollen den Zusammenhang zwischen der Größe der einzufügenden Zahlen und den Positionen der neuen Kästchen herstellen. Aussage darüber macht das Row-Bumping-Lemma.
Wir stellen fest, dass ein Weg genau ein Kästchen in mehreren aufeinanderfolgenden Zeilen besitzt, beginnend in der ersten Zeile und endend in der Zeile, in der sich das neue Kästchen befindet.
Die folgende Definition setzt fest, wie wir die Position zweier Wege oder Kästchen vergleichen wollen.

Definition 1.17. Ein Weg R sei *strikt links* (beziehungsweise *schwach links*) von einem weiteren Weg R', wenn in jeder Zeile, die ein Kästchen von R' enthält, R ein Kästchen besitzt, das links von diesem (beziehungsweise links oder auf diesem) liegt.
Analog definieren wir *strikt/schwach rechts* für Wege und einzelne Kästchen und *strikt/schwach über* und *strikt/schwach unter* für Zeilen und Kästchen.

Mit diesen Definitionen können wir nun das erwähnte Row-Bumping-Lemma formulieren:

Proposition 1.18 (Row-Bumping-Lemma)**.** *Gegeben seien zwei aufeinanderfolgende Einreihungen durch den Bumping-Algorithmus: Die Erste füge das Element x in das Tableau T, die Zweite x' in das resultierende Tableau $T \leftarrow x$ ein. Die auf diese Weise entstehenden Wege seien bezeichnet mit R beziehungsweise R', die neuen Kästchen mit B beziehungsweise B'. Es gilt:*

1. *$x \leq x'$ genau dann, wenn R strikt links von R' und B schwach unter B' liegt.*
2. *$x > x'$ genau dann, wenn R' schwach links von R und B' strikt unter B liegt.*

Beweis. zu (1): Sei $x \leq x'$ angenommen. Bei Anwendung des Algorithmus verdrängen x und x' die Einträge a und a'. Da $x \leq x' < a'$, liegt a' strikt rechts von dem Kästchen, das x eingenommen hat; es gilt also $a \leq a'$. Dieses Argument findet in jeder Zeile Anwendung.
Wir stellen fest, dass R nicht überhalb von R' enden kann: Endet R mit dem Eintrag $\tilde{a}$ in B, so ist es stets möglich, den Eintrag $\tilde{a}'$ an diese Reihe anzuhängen, da $\tilde{a} \leq \tilde{a}'$.
Falls R' weiter oben als R endet, bewegt sich R in seinem weiteren Verlauf nicht mehr nach rechts: B wurde an das Ende einer Zeile angefügt und da wir ein Tableau betrachten, haben alle folgenden Zeilen weniger Kästchen als diejenige, die B als letztes Kästchen enthält.
Sei nun umgekehrt R strikt links von R' gegeben. Dann ist insbesondere das oberste Kästchen von R strikt links von dem obersten Kästchen von R'. Die beiden Kästchen haben

die Einträge x beziehungsweise x'. Da wir ein Tableau betrachten, gilt $x \leq x'$.
zu (2): Seien a und a' analog definiert. In diesem Fall gilt $a > a'$, was sich wieder in allen folgenden Zeilen fortsetzt.
Für $x > x'$ ist R' immer mindestens eine Zeile länger als R: Endet R mit dem Eintrag $\tilde{a}$ in B, so kann $\tilde{a}'$ nicht mehr an diese Reihe angehängt werden, da $\tilde{a} > \tilde{a}'$ und es muss in jedem Fall die nächste Zeile noch miteinbezogen werden.
Um die Umkehrung zu zeigen, betrachten wir zwei Fälle.
Für den ersten Fall sei R' strikt links von R gegeben. Wie oben gilt $x' \leq x$ nach Konstruktion. Nehmen wir an, dass $x = x'$. Reihen wir x' in das Tableau ein, so suchen wir in der ersten Zeile nach dem am weitesten links gelegenen Eintrag, der größer als x' ist. Dieser müsste rechts von x liegen, was ein Widerspruch zur Voraussetzung ist, dass R' strikt links von R liegt. Es gilt also $x' < x$.
Im zweiten Fall betrachten wir R' schwach links von R, sodass die obersten Kästchen von R und R' an derselben Stelle liegen. Wir betrachten die Situation, in der x schon eingereiht wurde. Damit die Wege R und R' sich in der ersten Zeile überdecken, muss x' den Eintrag x verdrängen, was nur dann passiert, wenn $x' < x$. □

Korollar 1.19. *Sei T ein Tableau mit Umriss μ,*

$$U = (\dots((T \leftarrow x_1) \leftarrow x_2) \leftarrow \dots) \leftarrow x_p \qquad \textit{für gewisse } x_i \in \mathbb{N}.$$

Sei λ der Umriss von U. Ist $x_1 \leq x_2 \leq \dots \leq x_p$ (beziehungsweise $x_1 > x_2 > \dots > x_p$), so sind keine zwei Kästchen von λ/μ in derselben Spalte (beziehungsweise derselben Zeile).
Umgekehrt: Sei U ein Tableau mit Umriss λ und μ ein darin enthaltenes Young-Diagramm, sodass λ/μ aus p Kästchen besteht. Sind keine zwei Kästchen des Schiefdiagramms λ/μ in derselben Spalte (beziehungsweise Zeile), so gibt es genau ein Tableau T mit Umriss μ und eindeutige $x_1 \leq x_2 \leq \dots \leq x_p$ (beziehungsweise $x_1 > x_2 > \dots > x_p$), sodass

$$U = (\dots((T \leftarrow x_1) \leftarrow x_2) \leftarrow \dots) \leftarrow x_p.$$

Beweis. Die erste Behauptung ist eine direkte Folgerung aus dem Row-Bumping-Lemma; dieses muss lediglich sukzessive p-mal mit den Einträgen $x_1, x_2, \dots, x_p$ angewandt werden.
Um die zweite Behauptung zu zeigen, unterscheiden wir zwei Fälle:
Fall 1 (keine zwei Kästchen von λ/μ liegen in derselben Spalte): Wir wenden die Umkehrung des Bumping-Algorithmus auf U an, angefangen bei dem am weitesten rechts gelegenen Kästchen von λ/μ und sich vorwärtsarbeitend von rechts nach links. Auf diese Weise ensteht T. Die verdrängten Elemente sind $x_p, x_{p-1}, \dots, x_1$. Aus dem Row-Bumping-Lemma folgt dann, dass $x_1 \leq x_2 \leq \dots \leq x_p$.
Fall 2 (keine zwei Kästchen von λ/μ liegen in derselben Zeile): Wir gehen ähnlich vor wie in Fall 1. Wieder wenden wir die Umkehrung des Bumping-Algorithmus p-mal an, beginnen diesmal jedoch mit dem am weitesten unten gelegenen Kästchen und gehen weiter von unten nach oben. Auch hier erhalten wir so $x_p, x_{p-1}, \dots, x_1$. Das Row-Bumping-Lemma liefert uns, dass $x_1 > x_2 > \dots > x_p$. □

Wir sind nun so weit, ein (erstes) Produkt von Tableaux zu definieren:

Definition 1.20. Gegeben seien zwei Tableaux T und U, wobei U aus p Kästchen bestehe. Diese nummerieren wir von links nach rechts und von unten nach oben mit $x_1, x_2, \ldots, x_p$ durch. Wir definieren

$$T \cdot U = (\ldots((T \leftarrow x_1) \leftarrow x_2) \leftarrow \ldots) \leftarrow x_p.$$

Bemerkung 1.21.

- Im Spezialfall p=1 entspricht $T \cdot U$ der Einreihung des Kästcheninhaltes x in T durch den Bumping-Algorithmus: $T \cdot U = T \leftarrow x$.
- Die Anzahl der Kästchen von $T \cdot U$ entspricht der Summe der Kästchen von U und T.
- Die Einträge von $T \cdot U$ sind die Einträge von T zusammen mit denen von U.

Beispiel 1.22. Gegeben seien die Tableaux

$$T = \begin{array}{|c|c|c|c|c|}\hline 1&2&3&3&5\\\hline 3&4&4\\\cline{1-3} 5\\\cline{1-1}\end{array} \quad \text{und} \quad U = \begin{array}{|c|c|c|}\hline 1&2&5\\\hline 3&4\\\cline{1-2}\end{array}.$$

Wir berechnen $T \cdot U$ in folgenden Schritten:

$$T \cdot U = \begin{array}{|c|c|c|c|c|}\hline 1&2&3&3&5\\\hline 3&4&4\\\cline{1-3} 5\\\cline{1-1}\end{array} \cdot \begin{array}{|c|c|c|}\hline 1&2&5\\\hline 3&4\\\cline{1-2}\end{array} = \begin{array}{|c|c|c|c|c|}\hline 1&2&3&3&3\\\hline 3&4&4&5\\\cline{1-4} 5\\\cline{1-1}\end{array} \cdot \begin{array}{|c|c|c|}\hline 1&2&5\\\hline 4\\\cline{1-1}\end{array}$$

$$= \begin{array}{|c|c|c|c|c|c|}\hline 1&2&3&3&3&4\\\hline 3&4&4&5\\\cline{1-4} 5\\\cline{1-1}\end{array} \cdot \begin{array}{|c|c|c|}\hline 1&2&5\\\hline\end{array} = \begin{array}{|c|c|c|c|c|c|}\hline 1&1&3&3&3&4\\\hline 2&4&4&5\\\cline{1-4} 3\\\cline{1-1} 5\\\cline{1-1}\end{array} \cdot \begin{array}{|c|c|}\hline 2&5\\\hline\end{array}$$

$$= \begin{array}{|c|c|c|c|c|c|}\hline 1&1&2&3&3&4\\\hline 2&3&4&5\\\cline{1-4} 3&4\\\cline{1-2} 5\\\cline{1-1}\end{array} \cdot \begin{array}{|c|}\hline 5\\\hline\end{array} = \begin{array}{|c|c|c|c|c|c|c|}\hline 1&1&2&3&3&4&5\\\hline 2&3&4&5\\\cline{1-4} 3&4\\\cline{1-2} 5\\\cline{1-1}\end{array}$$

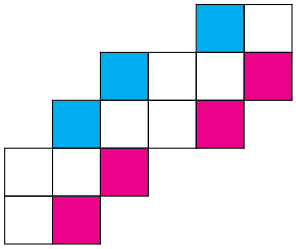
1.3 Schützenberger-Sliding-Algorithmus

Wir können ein Produkt von Tableaux auch über Schieftableaux definieren. Dazu führen wir zunächst Folgendes ein:

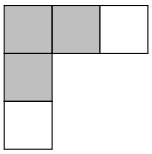
Definition 1.23. Sei λ/μ ein Schiefdiagramm.
Eine *innere Ecke* von λ/μ sei ein Kästchen des kleineren Tableaus μ, wo alle Kästchen unter und rechts von diesem nicht mehr zu μ gehören.
Eine *äußere Ecke* von λ/μ hingegen sei ein Kästchen von λ, wo alle Kästchen unter und rechts von diesem nicht mehr zu λ gehören.

Beispiel 1.24.

- Seien $\lambda = (6, 6, 5, 3, 2)$ und $\mu = (5, 3, 2)$ zwei partitionen. Dann hat das Schiefdiagramm λ/μ drei innere Ecken (blau), die nicht zu dem Schiefdiagramm gehören, und vier äußere Ecken (rot), die dazugehören:

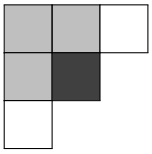


- Ein Schiefdiagramm λ/μ kann auf verschiedene Weisen durch die Wahlen von λ und μ enstehen. So haben beispielsweise



mit $\lambda/\mu = (3, 1, 1)/(2, 1)$

und



mit $\lambda/\mu = (3, 2, 1)/(2, 2)$

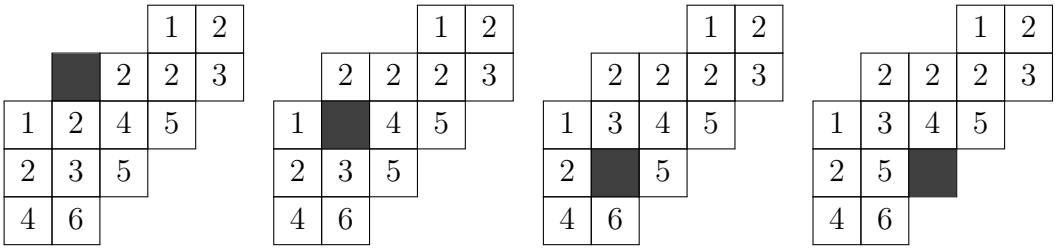
den gleichen Umriss. Bemerke, dass (3,2,1)/(2,2) ein Kästchen besitzt, das gleichzeitig innere und äußere Ecke ist (dunkelgrau).

Algorithmus 1.25 (Schützenberger Sliding-Algorithmus)**.** Gegeben sei ein Schieftableau S und eine innere Ecke von S. Es werden jeweils die beiden Kästchen unterhalb und rechts von der inneren Ecke betrachtet und deren Einträge verglichen. Die innere Ecke (meist gekennzeichnet durch eine schwarze Box, ein „Loch“) bewegt sich im Diagramm auf die folgende Weise:

- nach unten, wenn der Eintrag des unteren Kästchens kleiner als der des Kästchens zur Rechten ist oder wenn beide Einträge gleich groß sind
- nach rechts, wenn der Eintrag zur Rechten kleiner ist als der im unteren Kästchen.

Ist das Kästchen rechts von der schwarzen Box leer, so bewegt sich die Box automatisch nach unten. Ist umgekehrt unten ein leeres Kästchen, so wird sie sie nach rechts verschoben. Der Algorithmus endet, wenn das Kästchen der schwarzen Box zu einer äußeren Ecke geworden ist. Das schwarze Kästchen wird dann entfernt.

Beispiel 1.26.



Nun wird noch die schwarze Box entfernt und wir sind fertig.

Das Beispiel legt die Vermutung nahe, dass das Ergebnis wieder ein Schieftableau ist. In der Tat wurde der Algorithmus mit genau dieser Absicht konstruiert:

Bemerkung 1.27. *Wendet man den Sliding-Algorithmus auf ein Schieftableau an, so erhält man wieder ein Schieftableau:* Betrachten wir dazu die typische Konstellation

$$S = \begin{array}{|c|c|c|}\hline & b & v \\ \hline a & \blacksquare & y \\ \hline u & x & \\ \hline \end{array}$$

im Sliding-Algorithmus. Dabei können b, v, a und u auch situationsbedingt leer sein. Nehmen wir an, sie seien nicht leer, da dies den allgemeinen Fall zeigt.
Ist $x \leq y$, so führen wir die folgende Verschiebung durch:

$$S_1' = \begin{array}{|c|c|c|}\hline & b & v \\ \hline a & x & y \\ \hline u & \blacksquare & \\ \hline \end{array}$$

Damit es sich bei S' um ein Schieftableau handelt, muss gelten: $a \leq x \leq y$. Dabei gilt $x \leq y$ schon nach Voraussetzung. Da S selbst ein Schieftableau war, gilt $a < u \leq x$, also auch insbesondere $a \leq x$.
Ist $x > y$, so haben wir:

$$S_2' = \begin{array}{|c|c|c|}\hline & b & v \\ \hline a & y & \blacksquare \\ \hline u & x & \\ \hline \end{array}$$

Es muss gelten: $b \leq y \leq x$. Wieder gilt $y < x$ nach Voraussetzung und $b \leq v < y$, da S ein Schieftableau ist.

Algorithmus 1.28 (Umkehrung des Sliding-Algorithmus)**.** Um den Sliding-Algorithmus umzukehren, benötigen wir das Schieftableau, das sich ergeben hat. Zusätzlich müssen wir wissen, an welcher Stelle der Algorithmus geendet hat. Wir fügen an dieser Stelle das schwarze Kästchen wieder ein und beginnen nun dieses zu verschieben. Wir betrachten diesmal die benachbarten Kästchen oberhalb und links des Loches. Dieses bewegt sich

- nach oben, wenn beide Einträge gleich groß sind oder wenn der obere Eintrag größer als der zur Linken ist
- nach links, wenn der Eintag dort größer als derjenige oberhalb des Loches ist.

Bemerkung 1.29. Man macht sich leicht klar, dass diese Vorgehensweise auch wirklich den Sliding-Algorithmus umkehrt: Wir betrachten dazu noch einmal die typischen Konstellationen von oben. Im Fall von S_1' $(x \leq y)$ wissen wir, dass $u \leq v$, da S ein Schieftableau ist. Wir bewegen uns also laut Algorithmus 1.28 nach oben und stellen somit S wieder her. Im Schieftableau S_2' $(x > y)$ wissen wir, dass $v < y$ und gelangen durch eine Verschiebung nach links wieder zurück zu S.

Definition 1.30. Wendet man den Sliding-Algorithmus solange auf ein Schieftableau S an, bis das Resultat keine inneren Ecken mehr besitzt, so erhält man ein Tableau. Dieses Tableau nennen wir *Rektifizierung von* S und schreiben $\mathrm{rect}(S)$.
Der Prozess an sich wird oft mit *Jeu de Taquin*[3] bezeichnet.

Wie wir es zuvor auch schon mit dem Bumping-Algorithmus getan haben, definieren wir nun ein weiteres Produkt von Tableaux über den Sliding-Algorithmus.

Definition 1.31. Seien T und U zwei Tableaux. Wir konstruieren ein Schieftableau $T * U$, indem wir ein Rechteck mit genauso vielen Spalten wie T und Zeilen wie U bilden und dann T unten und U rechts anlegen. Wir definieren $T \cdot U = \mathrm{rect}(T * U)$.

Beispiel 1.32. Für

$$\mathrm{T} = \begin{array}{|c|c|}\hline 1 & 2 \\ \hline 4 & 4 \\ \hline \end{array} \quad \text{und} \quad \mathrm{U} = \begin{array}{|c|c|}\hline 2 & 3 \\ \hline 3 \\ \cline{1-1} \end{array}$$

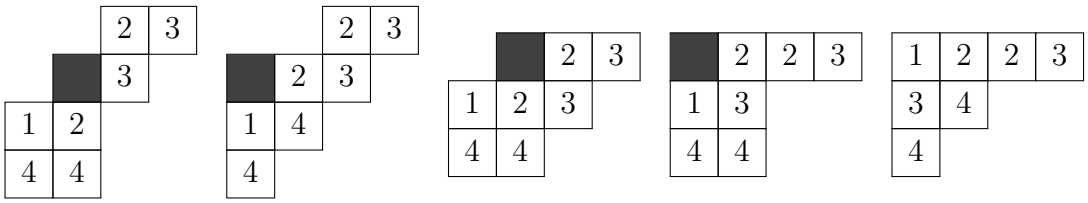
ist $T * U$ gegeben durch

$$\begin{array}{|c|c|}\hline 2 & 3 \\ \hline 3 \\ \cline{1-1} \end{array}\,.$$

$$\begin{array}{|c|c|}\hline 1 & 2 \\ \hline 4 & 4 \\ \hline \end{array}$$

[3]Beim Jeu de Taquin handelt es sich um ein bekanntes Schieberätsel, bei dem in einem 4x4-Quadrat die Zahlen 1 bis 15 in aufsteigender Reihenfolge angeordnet werden müssen. Es ist auch unter dem Namen 15-Puzzle bekannt.

Wir berechnen $T \cdot U = \mathrm{rect}(T * U)$ in folgenden Schritten:



Wir werden später sehen, dass diese Multiplikation wohldefiniert ist, d.h. das Ergebnis hängt nicht davon ab, in welcher Reihenfolge man die inneren Ecken wählt. Wir werden außerdem zeigen, dass die beiden Produkte, die wir definiert haben, sogar übereinstimmen. Dabei wird uns die Knuth-Äquivalenz von Worten helfen, die im zweiten Kapitel thematisiert wird.

Kapitel 2

Knuth-Äquivalenz von Worten

Zu Beginn dieses Kapitels werden wir den Begriff „Wort“ einführen, aber sehen, dass es keine Bijekion zwischen der Menge der Worte und der Menge der Tableaux gibt. In diesem Fall hilft uns die Knuth-Äquivalenz von Worten, sodass wir im letzten Kapitel zumindest einen Isomorphismus zwischen der Menge der Knuth-Äquivalenzklassen von Worten und der Menge der Tableaux finden können. Zudem werden wir zeigen, dass die Knuth-Äquivalenzklassen invariant unter den Algorithmen aus Kapitel 1 sind.

2.1 Worte

Ein *Wort* ist an sich eine Aneinanderreihung von natürlichen Zahlen (*Buchstaben*) wie zum Beispiel 156372863261. Um über Knuth-Äquivalenz reden zu können, wollen wir jedoch erst einmal definieren, was das Wort eines Tableaus sein soll.

Definition 2.1. Das *Wort eines (Schief-)Tableaus* T sei die Aneinanderreihung seiner Einträge, beginnend von links nach rechts und von unten nach oben. Wir bezeichnen dieses mit $w(T)$.
Die Verknüpfung $w \cdot w'$ sei definiert als die Nebeneinanderstellung der Worte w und w'.

Beispiel 2.2. Wir betrachten das Tableau

$$\mathrm{T} = \begin{array}{|c|c|c|c|c|c|} \hline 1 & 2 & 2 & 3 & 4 & 5 \\ \hline 2 & 3 & 4 & 4 \\ \cline{1-4} 4 & 4 & 5 & 5 \\ \cline{1-4} 5 \\ \cline{1-1} \end{array} .$$

Dann ist das Wort von T gegeben durch $w(T) = 544552344122345$.

Bemerkung 2.3. Ein Tableau kann auch aus seinem Wort rekonstruiert werden. Hat man zum Beispiel das Wort $w(T) = 656345623445$ gegeben, so erhält man T, wenn man einen Trennstrich an den Stellen zieht, an denen die Folge nicht mehr monoton steigt und die so

entstandenen Teile nacheinander aufeinandersetzt. In unserem Fall würden wir wie folgt trennen: 6|56|3456|23445. Wir erhalten dann das Tableau

$$\mathrm{T} = \begin{array}{|c|c|c|c|c|}\hline 2 & 3 & 4 & 4 & 5 \\ \hline 3 & 4 & 5 & 6 \\ \cline{1-4} 5 & 6 \\ \cline{1-2} 6 \\ \cline{1-1} \end{array} \quad .$$

Mit verschiedenen Beispielen macht man sich schnell klar, dass nicht jedes Wort von einem Tableau kommt. Unterteilt man ein Wort in seine monoton steigenden Teilstücke und setzt diese aufeinander, so müssen dafür die Eigenschaften eines Tableaus gegeben sein. Im Allgemeinen ist dies nicht der Fall.

In der Tat kann man jedoch zu jedem Wort ein *Schief*tableau finden: Dazu teilt man das Wort wieder wie oben und setzt die Teilstücke an deren äußeren Ende von unten nach oben übereinander. Wegen der gewählten Unterteilung des Wortes ist klar, dass die Einträge jeder Spalte dann streng monoton steigen. Ein Beispiel wird diesen Vorgang veranschaulichen:
Ist das Wort $w = 1223|125|345|133$ gegeben, so kann man ihm das folgende Schieftableau $S(w)$ zuordnen:

$$\mathrm{S(w)} = \begin{array}{|c|c|c|c|c|c|c|}\hline \multicolumn{4}{c}{} & \multicolumn{3}{|c|}{1} & 3 & 3 \\ \cline{5-7} \multicolumn{3}{c}{} & \multicolumn{3}{|c|}{3} & 4 & 5 \\ \cline{4-6} \multicolumn{2}{c}{} & \multicolumn{2}{|c|}{1} & 2 & 5 \\ \cline{3-5} \multicolumn{1}{c}{} & 1 & 2 & 2 & 3 \\ \cline{2-5} \end{array}$$

2.2 Auswirkungen des Sliding- und Bumping-Algorithmus auf Worte

Es stellt sich uns nun die Frage: Wenn wir den Bumping- oder den Sliding-Algorithmus auf ein (Schief-)Tableau anwenden, welchen Einfluss haben die Algorithmen dann auf dessen Wort? Beginnen wir mit dem Bumping-Algorithmus:

Die Zahl x soll in ein Tableau eingereiht werden. Wir betrachten dazu zunächst die erste Zeile. Wir wollen diese schreiben als $u \cdot x' \cdot v$, wobei

- u und v Worte sind, deren Buchstaben monoton steigend angeordnet sind
- x' ein Buchstabe ist
- jeder Buchstabe in u kleiner als x ist (kurz: $u < x$)
- $x < x'$.

In diesem Fall nimmt x die Stelle von x' ein: $u \cdot x' \cdot v \mapsto u \cdot x \cdot v$. Mit x' wird in der nächsten Zeile analog verfahren.
Nehmen wir an, dass das betrachtete Tableau nur aus einer Zeile besteht, so wäre das Wort des neuen Tableaus $x' \cdot u \cdot x \cdot v$, da für x' eine neue Zeile angelegt würde. Der Grundschritt des Algorithmus besteht also immer aus

$$(u \cdot x' \cdot v) \cdot x \longmapsto x' \cdot u \cdot x \cdot v, \quad \text{falls } u \leq x < x' \leq v.$$

Beispiel 2.4. Analog zu Beispiel 1.14 könnten wir so verfahren:

$$\begin{aligned}(456) \cdot (244) \cdot (1334) \cdot 2 &\longmapsto (456) \cdot (244) \cdot 3 \cdot (1234) \\ &\longmapsto (456) \cdot 4 \cdot (234) \cdot (1234) \\ &\longmapsto 5 \cdot (446) \cdot (234) \cdot (1234)\end{aligned}$$

Aus diesem Wort erhalten wir dasselbe Tableau wie schon zu Beginn.

Wir setzen nun $v = (v_1\ v_2 \dots v_q)$. Dann können wir die Vorgehensweise des Bumping-Algorithmus auch beschreiben durch:

$$\begin{aligned}u \cdot x' \cdot v_1 \cdots v_q \cdot x &\longmapsto u \cdot x' \cdot v_1 \cdots v_{q-1} \cdot x \cdot v_q && (x < v_{q-1} \leq v_q) \\ &\longmapsto u \cdot x' \cdot v_1 v_{q-2} \cdot x \cdot v_{q-1} \cdot v_q && (x < v_{q-2} \leq v_{q-1}) \\ &\longmapsto \quad \dots \\ &\longmapsto u \cdot x' \cdot v_1 \cdot x \cdot v_2 \cdots v_q && (x < v_1 \leq v_2) \\ &\longmapsto u \cdot x' \cdot x \cdot v_1 \cdots v_q && (x < x' \leq v_1)\end{aligned}$$

Wir sehen, dass in jedem Schritt jeweils nur drei aufeinanderfolgende Buchstaben involviert sind. Grundlegend für jeden Schritt ist

$$yzx \longmapsto yxz, \qquad \text{falls } x < y \leq z. \tag{K1}$$

Angelangt an der Stelle, an welcher x' von x verdrängt wird, beschreiben wir den weiteren Weg von x'. Wir setzen nun $u = (u_1\ u_2 \dots u_p)$:

$$\begin{aligned}u_1 \cdots u_p \cdot x' \cdot x \cdot v &\longmapsto u_1 \cdots u_{p-1} \cdot x' \cdot u_p \cdot x \cdot v && (u_p \leq x < x') \\ &\longmapsto u_1 \cdots u_{p-2} \cdot x' \cdot u_{p-1} \cdot u_p \cdot x \cdot v && (u_{p-1} \leq u_p < x') \\ &\longmapsto \quad \dots \\ &\longmapsto u_1 \cdot x' \cdot u_2 \cdots u_p \cdot x \cdot v && (u_2 \leq u_3 < x') \\ &\longmapsto x' \cdot u_1 \cdot u_2 \cdots u_p \cdot x \cdot v && (u_1 \leq u_2 < x')\end{aligned}$$

Die Grundlage hier ist also:

$$xzy \longmapsto zxy, \qquad \text{falls } x \leq y < z. \tag{K2}$$

Wir können (K1) und (K2) auch als Multiplikation von Tableaux auffassen:

$$\begin{array}{|c|c|}\hline y & z \\ \hline \end{array} \cdot \begin{array}{|c|}\hline x \\ \hline \end{array} = \begin{array}{|c|c|}\hline x & z \\ \hline y \\ \cline{1-1} \end{array}, \qquad \text{falls } x < y \leq z \text{ und} \tag{K1$'$}$$

$$\begin{array}{|c|c|}\hline x & z \\ \hline \end{array} \cdot \begin{array}{|c|}\hline y \\ \hline \end{array} = \begin{array}{|c|c|}\hline x & y \\ \hline z \\ \cline{1-1} \end{array}, \qquad \text{falls } x \leq y < z. \tag{K2$'$}$$

Wir haben also gesehen, dass sich das Wort $w(T) \cdot x$ durch elementare Umformungen der oben beschriebenen Weise in das Wort zu $T \leftarrow x$ überführen lässt.

Definition 2.5. Die Anwendung von (K1), (K2) oder deren Umkehrungen heißt eine *elementare Knuth-Transformation*. Zwei Worte w und w' seien *Knuth-äquivalent*, wenn sie durch elementare Knuth-Transformationen ineinander überführt werden können. Wir schreiben dann $w \equiv w'$.

In der neuen Notation formuliert haben wir gezeigt:

Proposition 2.6. *Sei T ein Tableau und $x \in \mathbb{N}$. Dann gilt:*

$$w(T \leftarrow x) \equiv w(T) \cdot x.$$

Korollar 2.7. *Ist $T \cdot U$ das Produkt zweier Tableaux T und U (definiert durch den Bumping-Algorithmus), so gilt:* $w(T \cdot U) \equiv w(T) \cdot w(U)$.

Beweis. Folgt aus Proposition 2.6, da die Buchstaben des Wortes U sukzessive in T eingereiht werden. □

Kommen wir nun zum Einfluss des Sliding-Algorithmus. Man kann die elementaren Knuth-Transformationen auch in Form von Schieftableaux formulieren:

$$\begin{array}{|c|c|}\hline \blacksquare & x \\ \hline y & z \\ \cline{1-2} \end{array} \rightarrow \begin{array}{|c|c|}\hline \blacksquare & x \\ \hline y & z \\ \cline{1-2} \end{array} \rightarrow \begin{array}{|c|c|}\hline x & z \\ \hline y \\ \cline{1-1} \end{array}, \qquad \text{falls } x < y \leq z. \tag{K1$''$}$$

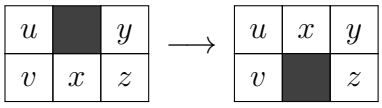
$$\begin{array}{|c|c|}\hline \blacksquare & y \\ \hline x & z \\ \cline{1-2} \end{array} \rightarrow \begin{array}{|c|c|}\hline \blacksquare & y \\ \hline x & z \\ \cline{1-2} \end{array} \rightarrow \begin{array}{|c|c|}\hline x & y \\ \hline z \\ \cline{1-1} \end{array}, \qquad \text{falls } x \leq y < z. \tag{K2$''$}$$

Wir sehen also, dass die Knuth-Äquivalenzklassen invariant unter diesen Sliding-Operationen sind. Weniger offensichtlich ist, was mit Worten passiert, wenn eine beliebige Sliding-Operation durchgeführt wird.

Proposition 2.8. *Können zwei (Schief-)Tableaux durch Sliding-Operationen ineinander überführt werden, so sind ihre Worte Knuth-äquivalent.*

Vorbemerkung. Wendet man den Sliding-Algorithmus auf ein Schieftableau an, so erhält man zwar nicht in jedem Schritt ein Schieftableau, jedoch zumindest ein Schieftableau mit einem Loch. Für diese sei der Begriff *Wort* analog zu oben definiert.

Beweis. Wird das Loch in einem Schritt des Algorithmus horizontal verschoben, so ist nach Definition klar, dass das Wort dadurch nicht verändert wird.
Von Interesse ist vielmehr, was eine vertikale Verschiebung verursacht. Betrachten wir zunächst den Spezialfall

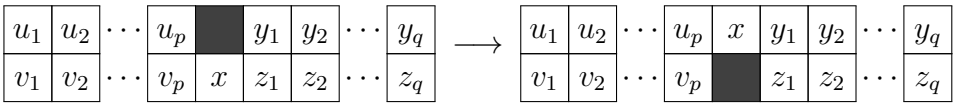


mit $u < v \leq x \leq y < z$,

um die Vorgänge zu verstehen. Das Wort $vxzuy$ geht über in $vzuxy$ durch die folgenden Knuth-Transformationen:

$$\begin{aligned} v\underline{xz}uy &\overset{(K1)}{\equiv} \underline{vxu}zy && (\text{da } u < x < z, \text{ insb. } x \leq z)\\ &\overset{(K1)}{\equiv} vu\underline{xz}y && (\text{da } u < v \leq x)\\ &\overset{(K2)}{\equiv} v\underline{uz}xy && (\text{da } x \leq y < z)\\ &\overset{(K2)}{\equiv} vzuxy && (\text{da } u < x < z, \text{ insb. } u \leq x) \end{aligned}$$

Aus diesem Spezialfall erhalten wir den allgemeinen Fall, wenn wir die Ecken u,v,y und z durch Zeilen $u = (u_i)_{i=1,\dots,p}$, $v = (v_i)_{i=1,\dots,p}$, $y = (y_j)_{j=1,\dots,q}$ und $z = (z_j)_{j=1,\dots,q}$ ersetzen, deren Einträge jeweils eine schwach monoton steigende Folge bilden. Zudem wollen wir voraussetzen, dass $u_i < v_i$ für alle i, $y_j < z_j$ für alle j und $v_p \leq x \leq y_1$ gilt, damit es sich überhaupt um ein Tableau handelt. Wir betrachten also den Übergang



Behauptung: $vxzuy \equiv vzuxy$
Wir zeigen dies durch Induktion nach p:
Für den Induktionsanfang $p = 0$ müssen wir zeigen, dass $xzy \equiv zxy$ gilt, also

$$xz_1z_2 \dots z_qy_1y_2 \dots y_q \equiv z_1z_2 \dots z_qxy_1y_2 \dots y_q.$$

Reiht man y_1 in das Wort $xz_1z_2 \dots z_q$ ein, so wird z_1 von y_1 verdrängt, da $x \leq y_1 < z_1$ und z_1 somit der am weitesten links gelegene Buchstabe ist, der größer als y_1 ist. Laut Proposition 2.6 ist das Einreihen durch den Bumping-Algorithmus invariant unter Knuth-Transformationen, sodass das neue Wort Knuth-äquivalent zu dem Ausgangswort ist und es gilt:

$$(xz_1z_2 \dots z_q) \cdot y_1 \cdot (y_2y_3 \dots y_q) \equiv (z_1xy_1z_2 \dots z_q) \cdot (y_2y_3 \dots y_q).$$

Führen wir dies weiter für alle y_i, reihen also y_k in $xy_1z_2 \dots z_{k-1}z_kz_{k+1} \dots z_q$ ein, so erhalten wir die Behauptung.
Sei die Behauptung nun schon für alle $k < p$ gezeigt.
Um den Beweis übersichtlicher zu gestalten, setzen wir abkürzend $u' = u_2u_3 \dots u_p$ und $v' = v_2v_3 \dots v_p$ und betrachten $vxzuy = v_1v'xzu_1u'y$. Wir reihen nun u_1 in $v_1v'xz$ ein. Es wird v_1 verdrängt und Proposition 2.6 lässt uns wieder folgern, dass $v_1v'xz_1 \equiv v_1u_1v'xz$, also

$$vxzuy = v_1v'xzu_1u'y \equiv v_1u_1v'xzu'y.$$

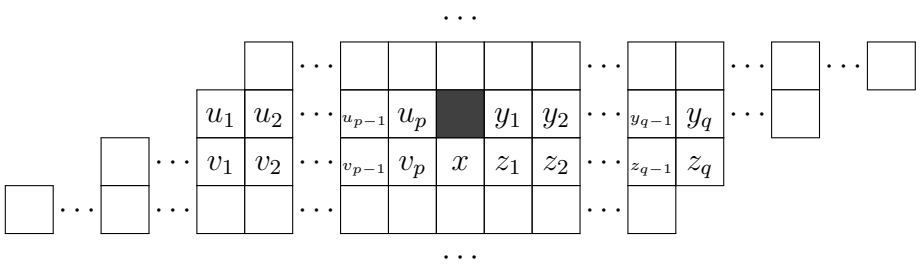
Laut Induktionsvoraussetzung ist bereits $v'xzu'y \equiv v'zu'xy$ und wir wissen daher, dass

$$v_1u_1v'xzu'y \equiv v_1u_1v'zu'xy.$$

Zuletzt reihen wir noch u_1 in $v_1v'z$ ein, wobei v_1 verdrängt wird und wir $v_1v'zu_1 \equiv v_1u_1v'z$ erhalten.
Sukzessives Anwenden dieser Vorgehensweise auf alle u_i's zeigt die Behauptung. Schlussendlich folgt dann, dass

$$v_1u_1v'zu'xy \equiv v_1v'zu_1u'xy = vzuxy.$$

Eine beliebige Verschiebung in einem *Schief*tableau bekommt man, indem man das Tableau auf die folgende Weise erweitert:



□

Proposition 2.9. *Jedes Wort ist Knuth-äquivalent zu dem Wort eines Tableaus.*

Beweis. Wir betrachten das Wort $w = x_1x_2 \dots x_p$. Mit Proposition 2.6 folgt, dass

$$w \;\equiv\; w((\dots((\boxed{x_1} \leftarrow x_2) \leftarrow x_3) \leftarrow \dots) \leftarrow x_p).$$

□

Wir werden noch sehen, dass jedes Wort zu dem Wort *genau eines* Tableaus Knuth-äquivalent ist. Nehmen wir die Eindeutigkeit also im Weiteren an und beweisen sie später. Dann folgt aus Proposition 2.8 und 2.9:

Korollar 2.10. *Die Rektifizierung eines Schieftableaus S ist das (eindeutige) Tableau, dessen Wort Knuth-äquivalent zu dem von S ist.*
Sind S und S' zwei Schieftableaux, so gilt:

$$\mathrm{rect}(S) = \mathrm{rect}(S') \;\Leftrightarrow\; w(S) \equiv w(S').$$

Diese Beobachtung legt ein dritte Möglichkeit nahe, das Produkt zu definieren:

Definition 2.11. Seien T und U zwei Tableaux. $T \cdot U$ sei das (eindeutige) Tableau, dessen Wort Knuth-äquivalent zu $w(T) \cdot w(U)$ ist.

Proposition 2.12. *Die drei gegebenen Konstruktionen des Produkts (Definition 1.20, 1.30 und 2.9) stimmen überein.*

Beweis. Dazu genügt es, zu zeigen: Jede der ersten beiden Konstruktionen liefert ein Produkt $T \cdot U$ mit $w(T \cdot U) \equiv w(T) \cdot w(U)$.
Konstruktion aus Definition 1.20: Die Einträge von U werden nacheinander in das Tableau T eingereiht. Proposition 2.6 zeigt uns hier die Behauptung.
Konstruktion aus Definition 1.30: Bei diesem Produkt überführen wir ein Schieftableau mit dem Wort $w(T) \cdot w(U)$ in ein Tableau $T \cdot U$ mittels *Jeu de Taquin*. In diesem Fall zeigt uns Proposition 2.8, dass die beiden Worte Knuth-äquivalent sind. □

Aus dieser Proposition folgt nun, dass die Rektifizierung eines beliebigen Schieftableaus eindeutig ist. Dies ist auch die behauptete Eindeutigkeit ab Proposition 2.9.

Kapitel 3

Schur-Polynome

Im letzten Kapitel werden wir das zentrale Theorem dieser Bachelorarbeit beweisen, nämlich dass die Schur-Polynome symmetrisch sind. Dazu werden wir die Kostka-Zahl einführen, die uns eine Darstellung der Schur-Polynome in einer Basis der symmetrischen Polynome liefert. Abschließend werden wir noch kurz auf weitere symmetrische Polynome und deren Verbindung zu den Schur-Polynomen eingehen.

3.1 Symmetrie von Schur-Polynomen

Sei $M = M_m$ die Menge aller Knuth-Äquivalenzklassen von Worten im Alphabet $[m]$. Die Nebeneinanderstellung von Worten stellt eine assoziative Multiplikation auf M dar. M ist abgeschlossen bezüglich Multiplikation, da

$$w \equiv w', v \equiv v' \Rightarrow w \cdot v \equiv w' \cdot v \equiv w' \cdot v'.$$

Dadurch wird M ein Monoid mit dem leeren Wort $\emptyset$ als neutrales Element.

Sei F die Menge aller Worte. F ist ein freier Monoid, da die Buchstaben als Teilmenge von F eine Basis bilden, mit Nebeneinanderstellung als Produkt und dem leeren Wort als neutralem Element. Die Abbildung

$$k : F \longrightarrow M, \qquad w \longmapsto [w]$$

ist also ein Homomorphismus zwischen Monoiden.

Wir können M auch definieren, indem wir $M = {}^{F}/_{\equiv}$ bilden. Auf diese Weise erhalten wir ebenfalls die Menge aller Knuth-Äquivalenzklassen von Worten.

Auch die Menge der Tableaux mit Verknüpfung $\cdot$ und dem leeren Tableau $\emptyset$ als neutrales Element ist ein Monoid. Wir haben gesehen, dass es über Knuth-Äquivalenzen eine lineare Eins-zu-Eins-Zuordnung zwischen den Äquivalenzklassen der Worte und den Tableaux mit

Einträgen aus $[m]$ gibt, nämlich

$$T \longmapsto [w(T)] \quad \text{mit Umkehrung} \quad [w] \longmapsto \text{rect}(S(w)).$$

Daher ist M isomorph zum Monoid der Tableaux. Wenn wir nun von M reden, können wir dessen Elemente auch als Tableaux identifizieren.

Zu dem gegebenen Monoid M führen wir den zugehörigen *Tableau-Ring* $R_{[m]}$ als freien $\mathbb{Z}$-Modul mit Basis M ein. Das Produkt sei wie bisher das Produkt von Tableaux. Zu einem gegebenen Umriss λ definieren wir $S_\lambda = S_\lambda[m]$ als die formale Summe aller Tableaux T mit Umriss λ und Einträgen in $[m]$.
Wir definieren auf dem Tableau-Ring den Homomorphismus

$$g : R_{[m]} \ni T \overset{g}{\longmapsto} x^T \in \mathbb{Z}[x_1, x_2, \ldots, x_m].$$

Das Bild von S_λ unter g ist $s_\lambda(x_1, x_2, \ldots, x_m)$.

Zwei Spezialfälle zur Multiplikation von Schur-Polynomen folgen aus dem Row-Bumping-Lemma. Es gilt:

$$S_\lambda \cdot S_{(p)} = \sum_\mu S_\mu, \tag{1}$$

wobei über alle μ summiert wird, die entstehen, indem man p Kästchen zu λ hinzufügt, jedoch keine zwei in derselben *Spalte*, und

$$S_\lambda \cdot S_{(1^p)} = \sum_\mu S_\mu, \tag{2}$$

wobei über alle μ summiert wird, die entstehen, indem man zu λ genau p Kästchen hinzufügt, jedoch keine zwei in derselben *Zeile*.

Wendet man den Homomorphismus $T \mapsto x^T$ an, so folgt aus (1) und (2)

$$s_\lambda(x_1, x_2, \ldots, x_m) \cdot h_p(x_1, x_2, \ldots, x_m) = \sum_\mu s_\mu(x_1, x_2, \ldots, x_m) \tag{3}$$

und

$$s_\lambda(x_1, x_2, \ldots, x_m) \cdot \sigma_p(x_1, x_2, \ldots, x_m) = \sum_\mu s_\mu(x_1, x_2, \ldots, x_m) \tag{4}$$

mit μ jeweils wie oben.

Definition 3.1. Seien λ und u Partitionen. $K_{\lambda u}$ sei die Anzahl der Tableaux mit Umriss λ mit Gewicht u. Genauer: $K_{\lambda u}$ sei die Anzahl der Folgen von Partitionen

$$\lambda^{(1)} \subset \lambda^{(2)} \subset \ldots \subset \lambda^{(l)} = \lambda,$$

wo das Schiefdiagramm $\lambda^{(i)}/\lambda^{(i-1)}$ genau u_i Kästchen hat, jedoch keine zwei in derselben Spalte.
$K_{\lambda u}$ heißt *Kostka-Zahl*.

Beispiel 3.2. Es seien $\lambda = (3,3,1)$ und $u = (2,2,2,1)$ zwei gegebene Partitionen. Dann ist die Kostka-Zahl $K_{\lambda u} = 3$. Dieses kann man auf zwei verschiedene Arten sehen:

1. Als erste Möglichkeit können wir uns überlegen, wie viele Tableaux mit Umriss λ und Gewicht u es gibt. Wir finden dazu drei verschiedene Belegungen:

1	1	2
2	3	4
3		

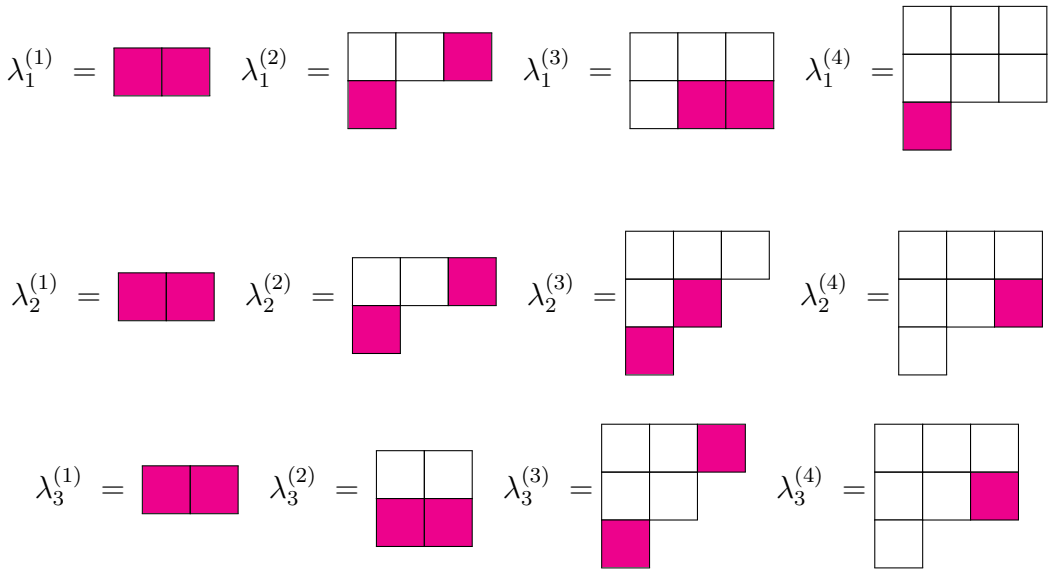
1	1	3
2	2	4
3		

1	1	2
2	3	3
4		

2. Die zweite Möglichkeit besteht darin, sich die Anzahl der Folgen

$$\lambda^{(1)} \subset \lambda^{(2)} \subset \lambda^{(3)} \subset \lambda^{(4)} = (3,3,1)$$

zu überlegen, wobei $\lambda^{(i)}/\lambda^{(i-1)}$ genau u_i Kästchen besitzt (keine zwei in derselben Spalte). Auch von diesen Folgen gibt es drei:



Lemma 3.3 (vgl [4])**.** *Mit den Bezeichnungen von oben gilt*

$$S_{(u_1)} \cdot S_{(u_2)} \cdot \ldots \cdot S_{(u_l)} = \sum_{\lambda \text{ Partition}} K_{\lambda u} S_\lambda \tag{5}$$

und

$$S_{(1^{u_1})} \cdot S_{(1^{u_2})} \cdot \ldots \cdot S_{(1^{u_l})} = \sum_{\lambda \text{ Partition}} K_{\tilde{\lambda}u} S_\lambda = \sum_\lambda K_{\lambda u} S_{\tilde{\lambda}}, \tag{6}$$

wobei $\tilde{\lambda}$ die Konjugation von λ bezeichnet.

Beweis. Wir beweisen (5) durch vollständige Induktion nach l. Für $l = 1$ müssen wir zeigen, dass

$$S_{(u_1)} = \sum_\lambda K_{\lambda(u_1)} S_\lambda.$$

Bei $S_{(u_1)}$ handelt es sich um die Summe zeilenförmiger Tableaux mit u_1 Einsen. Von diesen gibt es jedoch nur eines, also ist $S_{u_1} = (u_1)$. Desweiteren ist $K_{\lambda(u_1)} = 0$ für alle λ, die aus mehr als einer Zeile bestehen, da in einem Tableau niemals zwei Einsen untereinander stehen dürfen. Für $\lambda = (u_1)$ ist $K_{\lambda(u_1)} = 1$ und wir haben gezeigt, dass $S_{(u_1)} = (u_1) = \sum_\lambda K_{\lambda(u_1)} S_\lambda$.
Sei die Behauptung nun für $k < l$ bereits gezeigt. Sei $u = (u_1, u_2, \ldots, u_l)$ und
$u' = (u_1, u_2, \ldots, u_{l-1})$ mit $|u| = u_1 + u_2 + \ldots + u_l$ und $|u'| = u_1 + u_2 + \ldots + u_{l-1}$. Dann können wir schreiben

$$\begin{aligned} S_{(u_1)} \cdot S_{(u_2)} \cdot \ldots \cdot S_{(u_{l-1})} \cdot S_{(u_l)} &\overset{IV}{=} \left(\sum_{\lambda \text{ Partition von } |u'|} K_{\lambda u'} S_\lambda \right) S_{(u_l)} \\ &\overset{(1)}{=} \sum_{\lambda \text{ Partition von } |u'|} K_{\lambda u'} \sum_\mu S_\mu \\ &= \sum_{\lambda \text{ Partition von } |u|} K_{\lambda u} S_\lambda, \end{aligned}$$

wobei in der Summe von μ über alle Young-Diagramme summiert wird, die entstehen, indem man zu λ genau u_l Kästchen hinzufügt, jedoch keine zwei in derselben Spalte.
Gleichung (6) wird auf ähnliche Weise durch vollständige Induktion gezeigt. □

Wendet man auf (5) beziehungsweise (6) wie oben den Homomorphismus $T \mapsto x^T$ an, so erhält man

$$h_{u_1} \cdot h_{u_2} \cdot \ldots \cdot h_{u_l} = \sum_{\lambda \text{ Partition}} K_{\lambda u} s_\lambda \tag{7}$$

und

$$\sigma_{u_1} \cdot \sigma_{u_2} \cdot \ldots \cdot \sigma_{u_l} = \sum_{\lambda \text{ Partition}} K_{\tilde{\lambda}u} s_\lambda = \sum_\lambda K_{\lambda u} s_{\tilde{\lambda}}. \tag{8}$$

Theorem 3.4. *Schur-Polynome sind symmetrisch.*

Beweis. Sind u und λ Partitionen, so gilt

- $u = \lambda \Rightarrow K_{\lambda u} = 1$ und
- $u \leq \lambda$ $(u \neq \lambda) \Rightarrow K_{\lambda u} = 0$,

wobei $u \leq \lambda$, falls $u_i = \lambda_i$ für alle i oder $u_1 = \lambda_1$, $u_2 = \lambda_2$, ..., $u_{i-1} = \lambda_{i-1}$ und $u_i < \lambda_i$ (*lexikographische Ordnung*).
Für lexikographisch geordnete Partitionen ist $(K_{u\lambda})$ also eine linke untere Dreiecksmatrix mit Einsen auf der Hauptdiagonalen. Um die Schur-Polynome in vollständig- beziehungsweise elementar-symmetrische Polynome umzuschreiben, können daher (7) und (8) benutzt werden. Es folgt, dass die Schur-Polynome symmetrisch sind. □

Beispiel 3.5. Gegeben seien drei Partitionen $\lambda_1 = u_1 = (3,0,0)$, $\lambda_2 = u_2 = (2,1,0)$ und $\lambda_3 = u_3 = (1,1,1)$. Mit Hilfe von (7) wollen wir die zugehörigen Schur-Polynome in h_1, h_2 und h_3 schreiben. Es sind

$$\begin{aligned} s_{\lambda_1} &= x_1^3 + x_1^2x_2 + x_1^2x_3 + x_1x_2^2 + x_1x_2x_3 + x_1x_3^2 + x_2^3 + x_2^2x_3 + x_2x_3^2 + x_3^3, \\ s_{\lambda_2} &= x_1^2x_2 + x_1^2x_3 + x_1x_2^2 + x_1x_2x_3 + x_1x_3^2 + x_2^2x_3 + x_2x_3^2 + x_3^3, \\ s_{\lambda_3} &= x_1x_2x_3. \end{aligned}$$

Wir müssen dazu die Matrix $K_{\lambda u}$ aufstellen und wissen dann, dass

$$\begin{pmatrix} K_{\lambda_1 u_1} & K_{\lambda_2 u_1} & K_{\lambda_3 u_1} \\ K_{\lambda_1 u_2} & K_{\lambda_2 u_2} & K_{\lambda_3 u_2} \\ K_{\lambda_1 u_3} & K_{\lambda_2 u_3} & K_{\lambda_3 u_3} \end{pmatrix} \begin{pmatrix} s_{\lambda_1} \\ s_{\lambda_2} \\ s_{\lambda_3} \end{pmatrix} = \begin{pmatrix} h_3 \\ h_1h_2 \\ h_1^3 \end{pmatrix}.$$

$K_{\lambda u}$ ist eine linke untere Dreiecksmatrix mit Einsen auf der Diagonalen. Wir müssen also nur noch $K_{\lambda_1 u_2}$, $K_{\lambda_1 u_3}$ und $K_{\lambda_3 u_3}$ bestimmen. λ_1 ist ein Zeilendiagramm mit drei Kästchen und es gibt nur eine Belegung mit Gewicht $u = (2,1,0)$; daher ist $K_{\lambda_1 u_2} = 1$. Auch für $K_{\lambda_1 u_3}$ gibt es nur eine Möglichkeit. Weiter ist $K_{\lambda_3 u_3} = 2$, da folgende Belegungen möglich sind:

1	2
3	

und

1	3
2	

Lösen wir nun das lineare Gleichungssystem

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & 1 \end{pmatrix} \begin{pmatrix} s_{\lambda_1} \\ s_{\lambda_2} \\ s_{\lambda_3} \end{pmatrix} = \begin{pmatrix} h_3 \\ h_1h_2 \\ h_1^3 \end{pmatrix},$$

so erhalten wir

$$s_{\lambda_1} = h_3 \quad s_{\lambda_2} = h_1 \cdot h_2 \quad s_{\lambda_3} = h_1h_2 - h_3.$$

Ausmultiplizieren zeigt, dass es sich dabei tatsächlich um die drei Schur-Polynome handelt, die wir zu Beginn aufgestellt haben.

3.2 Schur-Polynome und andere Klassen von symmetrischen Polynomen

Neben den Schur-Polynomen gibt es noch weitere Klassen von Polynomen, die eine $\mathbb{Z}$-Basis der symmetrischen Polynome bilden. Dieser Abschnitt soll den Zusammenhang zwischen diesen kurz darlegen, jedoch ohne die Zwischenergebnisse zu beweisen. Die Beweise können in [1] nachgelesen werden.

Die Formeln (7) und (8) kann man auch in Determinantenform schreiben. Dies zeigt uns für Partitionen $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k)$:

$$s_\lambda(x_1, x_2, \ldots, x_m) = \det(h_{\lambda_i+j-i})_{ij} \qquad 1 \leq i, j \leq k$$

mit $h_\lambda = h_{\lambda_1} \cdot h_{\lambda_2} \cdots h_{\lambda_k}$ und

$$s_\lambda(x_1, x_2, \ldots, x_m) = \det(\sigma_{\tilde{\lambda}_i+j-i})_{ij} \qquad 1 \leq i, j \leq l,$$

wobei $\tilde{\lambda}$ die Konjugation von λ mit l Zeilen ist und $\sigma_{\tilde{\lambda}} = \sigma_{\tilde{\lambda}_1} \cdot \sigma_{\tilde{\lambda}_2} \cdots \sigma_{\tilde{\lambda}_l}$.

Bisher noch nicht erwähnt haben wir die folgenden Polynome:

Definition 3.6. Sei $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ eine Partition. Das *monomial-symmetrische Polynom* m_λ sei die Summe aller Monome, die durch Permutation der Variablen x_1 bis x_m aus dem Monom $x_1^{\lambda_1} x_2^{\lambda_2} \cdots x_m^{\lambda_m}$ entstehen:

$$m_\lambda(x_1, x_2, \ldots, x_m) = \sum_{\tau \in \mathfrak{S}_m} x_{\tau(1)}^{\lambda_1} x_{\tau(2)}^{\lambda_2} \cdots x_{\tau(m)}^{\lambda_m}.$$

Proposition 3.7. *Folgende Mengen bilden jeweils eine $\mathbb{Z}$-Basis der symmetrischen Polynome vom Grad n in m Variablen:*

- $\{m_\lambda(x_1, x_2, \ldots, x_m) | \lambda$ *Partition von n mit mindestens m Zeilen*$\}$
- $\{s_\lambda(x_1, x_2, \ldots, x_m) | \lambda$ *Partition von n mit mindestens m Zeilen*$\}$
- $\{\sigma_\lambda(x_1, x_2, \ldots, x_m) | \lambda$ *Partition von n mit mindestens m Spalten*$\}$
- $\{h_\lambda(x_1, x_2, \ldots, x_m) | \lambda$ *Partition von n mit mindestens m Zeilen*$\}$
- $\{h_\lambda(x_1, x_2, \ldots, x_m) | \lambda$ *Partition von n mit mindestens m Spalten*$\}$

Ebenfalls ohne Beweis sei die folgende Identität gegeben (mit $x = (x_1, x_2, \ldots, x_m)$):

$$h_k(x) - \sigma_1(x) h_{k-1}(x) + \sigma_2(x) h_{k-2}(x) - \cdots + (-1)^k \sigma_k(x) = 0. \tag{9}$$

Abschließend wollen wir nun ein symmetrisches Polynom in den Basen aus Proposition 3.7 darstellen und somit den Zusammenhang von Schur-Polynomen und den anderen Klassen veranschaulichen.

Beispiel 3.8. Gegeben sei das symmetrische Polynom

$$f(x_1, x_2) = 2x_1^3 + x_1^2x_2 + 3x_1x_2 + x_1x_2^2 + 2x_2^3.$$

Wir wollen dieses in den oben genannten Basen schreiben.

- Für die Darstellung in *monomial-symmetrischen Polynomen* brauchen wir das Leitmonom von f bezüglich der lexikographischen Ordnung. Dieses ist $\mathrm{LM}(f) = x_1^3$. Nun suchen wir dasjenige monomial-symmetrische Polynom, welches das gleiche Leitmonom wie f hat: $m_{(3,0)} = x_1^3 + x_2^3$. Da unser Leitkoeffizient 2 ist, ziehen wir dieses Polynom zweimal von f ab. Wir wiederholen dieses Verfahren, bis kein Rest mehr vorhanden ist. Auf diese Weise erhalten wir dann

$$f(x_1, x_2) = 2m_{(3,0)} + m_{(2,1)} + 3m_{(1,1)}.$$

- Um f als Summe von *Schur-Polynomen* zu schreiben, verfahren wir prinzipiell analog zu den monomial-symmetrischen Polynomen. Das Schur-Polynom, welches das gleiche Leitmonom wie f hat, ist hier $s_{(3,0)} = x_1^3 + x_1^2x_2 + x_1x_2^2 + x_2^3$. Ziehen wir dieses zweimal von f ab, so ist unser neues Leitmonom $x_1^2x_2$; wir müssen also das Polynom $s_{(2,1)} = x_1^2x_2 + x_1x_2^2$ dazuaddieren. Übrig bleibt $3s_{(1,1)}$ und wir haben die folgende Darstellung gefunden:

$$f(x_1, x_2) = 2s_{(3,0)} - s_{(2,1)} + 3s_{(1,1)}.$$

- Wir gehen nun von der gerade gefundenen Darstellung aus, wenn wir f in *vollständig-symmetrische Polynome* umschreiben wollen. Wir wissen, dass $s_{(3,0)} = h_3$ ist und dass $s_{(1,1)} = \sigma_2 \overset{(9)}{=} h_1^2 - h_2 = h_{(1,1)} - h_{(2,0)}$. Um $1s_{(2,1)}$ umzuschreiben, benutzen wir die Determinantendarstellung. Es ist

$$s_{(2,1)} = \det \begin{pmatrix} h_2 & h_3 \\ h_0 & h_1 \end{pmatrix} = h_2h_1 - h_3 = h_{(2,1)} - h_{(3,0)},$$

  wobei $h_0 = 1$. Insgesamt bekommen wir also

$$\begin{aligned} f(x_1, x_2) &= 2h_{(3,0)} - \left(h_{(2,1)} - h_{(3,0)}\right) + 3\left(h_{(1,1)} - h_{(2,0)}\right) \\ &= 3h_{(3,0)} - h_{(2,1)} - 3h_{(2,0)} + 3h_{(1,1)}. \end{aligned}$$

- Als letztes wollen wir f noch mit Hilfe der *elementar-symmetrischen Polynome* darstellen. Dazu können wir zum Beispiel (9) benutzen. Für $k = 1, 2, 3$ und mit

$\sigma_3(x_1, x_2) = 0$ erhalten wir in $\mathbb{Z}[x_1, x_2]$

$$h_1 = \sigma_1, \qquad h_2 = \sigma_1^2 - \sigma_2, \qquad \text{und} \qquad h_3 = \sigma_1^3 - \sigma_1\sigma_2.$$

Einsetzen der h_i in die oben gefundene Darstellung liefert dann

$$f(x_1, x_2) = 2\sigma_1^3 - 5\sigma_1\sigma_2 + 3\sigma_2.$$

Literaturverzeichnis

[1] William Fulton, *Young Tableaux - With Application to Representation Theory and Geometry*, Cambridge University Press, Cambridge, 1997

[2] William Fulton, *Intersection Theory*, Springer, 1984

[3] Laurent Manivel, *Fonctions symétriques, polynômes de Schubert et lieux de dégénérescence*, Société Mathématique de France, 1998

[4] Michael Walter, *Representation Theory of the Symmetric Group*, http://www.leetspeak.org/math/notes/repr_sym.pdf, S.6, zuletzt aufgerufen am 21.05.2013

[5] Axel Kohnert, *Weintrauben, Polynome, Tableaux*, http://opus4.kobv.de/opus4-ubbayreuth/frontdoor/deliver/index/docId/210/file/27857.0.diss_kohnert.pdf, zuletzt aufgerufen am 14.05.2013

[6] MacTutor History of Mathematics archive, *Issai Schur*, http://www-history.mcs.st-and.ac.uk/Biographies/Schur.html, zuletzt aufgerufen am 14.05.2013

[8] Oberwolfach Photo Collection, *Issai Schur*, http://owpdb.mfo.de/detail?photo_id=12209, zuletzt aufgerufen am 21.05.2013