Algorithmische Berechnung von Summen

Diplomarbeit

Gregor Stölting

Freie Universität Berlin

Fachbereich Mathematik und Informatik

Januar 1996

Inhaltsverzeichnis

Einleitung

In vielen Bereichen der Mathematik, wie z.B. in der Analysis, der Kombinatorik und der Wahrscheinlichkeitsrechnung treten Summen wie

$$\sum_{k=0}^{n} \binom{n}{k}^2 \text{ oder } \sum_{k=0}^{n} \binom{n}{k} \left(-\frac{1}{2}\right)^k \tag{1}$$

auf.
Ziel ist es, eine explizite **geschlossene Form** für die Summe zu finden, das heißt, einen Ausdruck als Funktion mit der Variablen n, in der das Summenzeichen und die Laufvariable k nicht mehr vorkommen. Die beiden obigen Beispiele entsprechen den geschlossenen Formen $(2\,n)!/n!^2$ bzw. $(1/2)^n$.
Auch kann man nach numerischer Auswertung vermuten, daß

$$\sum_{k=0}^{n} \binom{n}{k}^3 \stackrel{?}{=} \sum_{k=0}^{n} \binom{n}{k}^2 \binom{2\,k}{n} \tag{2}$$

gilt, und es stellt sich die Frage, ob der Beweis dieser These auch ohne Berechnung einer geschlossenen Form möglich ist.
In den obigen Fällen (1) und (2) sprechen wir von **definiter** Summation; "definit" deshalb, weil der Index k über ein festes Intervall aus $\mathbb{Z}$ läuft. Dieses Intervall kann durchaus auch $]-\infty, \infty[$ sein, das heißt, die Summe läuft über alle $k \in \mathbb{Z}$, und man sucht eine geschlossene Form $A\,(n)$ mit

$$A\,(n) := \sum_{k} F\,(n, k) \tag{3}$$

zu gegebenem $F\,(n, k)$.
Eine geschlossene Form (3) kann aber nur dann gefunden werden, wenn $\lim\limits_{k \to \pm\infty} F\,(n, k) = 0$ gilt. Sonst kann es aus konvergenzgründen keine Lösungen für (3) geben.
Der **Zeilbergeralgorithmus** (Kapitel 3) kann entscheidende Vorarbeit zum Finden einer geschlossenen Form (3) leisten, wenn $F\,(n, k)$ nicht nur für k nach $\pm\infty$ gegen 0 geht, sondern sogar einen endlichen Träger bezüglich k hat, es also Intervallgrenzen $c\,(n)\,, d\,(n)$ mit $F\,(n, k) = 0$ für alle $k \notin [c\,(n)\,, d\,(n)]$ gibt. Dann gilt nämlich

$$A\,(n) = \sum_{k} F\,(n, k) = \sum_{k=c(n)}^{d(n)} F\,(n, k)\,.$$

Mit dem Zeilbergeralgorithmus wird eine Rekursion

$$\sum_{i=0}^{l} \sigma_i(n) A(n-i) = 0, \tag{4}$$

berechnet, wobei die $\sigma_i(n)$ **rational** bezüglich n sind, es also zwei Polynome $a(n), b(n)$ mit $\sigma_i(n) = a(n)/b(n)$ (siehe Abschnitt 1.2) gibt. Dabei $\sigma_0(n)$ zu 1 normiert werden.
Wird bereits eine Rekursion der Ordnung $l = 1$ gefunden, so kann eine geschlossene Form $A(n)$ berechnet werden, sofern sämtliche Nullstellen des Zähler- und des Nennerpolynoms von $\sigma_1(n)$ gefunden werden.
Auch wenn es keine geschlossene Form gibt, kann man doch die vom Zeilbergeralgorithmus gelieferte Rekursion benutzen, um Identitäten wie (2) zu überprüfen.
In den Zeilbergeralgorithmus ist eine Version des **Gosperalgorithmus** (Kapitel 2) eingebettet. Der Gosperalgorithmus betreibt **indefinite** Summation. Er berechnet eine Stammfunktion $s(k)$ mit

$$a(k) = s(k) - s(k-1)$$

zu gegebenem $a(k)$. Durch sogenanntes **Teleskopieren**, d.h. derartiges Aufsummieren, daß sich die mittleren Terme jeweils paarweise zu 0 addieren, erhält man eine definite Summe, etwa

$$\sum_{k=0}^{n} a(k) = \sum_{k=0}^{n} s(k) - s(k-1) =$$

$$s(n) - s(n-1) + s(n-1) - s(n-2) + \ldots + s(0) - s(-1) = s(n) - s(-1).$$

Die bestehende Analogie zum unbestimmten Integral, die gleich zu Beginn dieser Arbeit in Abschnitt 1.1 diskutiert wird, wurde übrigens schon von Leonard Euler[1] thematisiert.
Notwendig für den Gosperalgorithmus ist, daß die Eingabe ein **hypergeometrischer Term** in k ist, das heißt, daß der Quotient $a(k)/a(k-1)$ rational bezüglich k ist, es also zwei Polynome $z(k), n(k)$ mit $a(k)/a(k-1) = z(k)/n(k)$ gibt. Der Gosperalgorithmus liefert immer ein **hypergeometrisches** $s(k)$, falls dieses existiert. Die Beschränkung auf hypergeometrische Terme ermöglicht, daß nur noch mit Polynomen gerechnet werden muß.
Ähnlich verfährt der Zeilbergeralgorithmus. Es wird mit Polynomen in k und n gerechnet. Deshalb werden Eingaben $F(n,k)$ gefordert, deren Quotienten $F(n,k)/F(n,k-1)$ und $F(n,k)/F(n-1,k)$ rational, sowohl in k als auch in n sind.
Ein Problem beim Zeilbergeralgorithmus ist, daß er nicht immer die Rekursion für $A(n)$ (gemäß (3)) niedrigster Ordnung findet. So gibt er für die Eingabe

$$F(n,k) := (-1)^k \binom{n}{k} \binom{4k}{n} \tag{5}$$

[1] "Quemadmodum ad differentiam denotandam usi sumus signo Δ, ita summam indicabimus signo Σ. ...ex quo æquatio $z = \Delta y$, si inventur dabit quoque $y = \Sigma z + C$"
Übersetzung: "Auf diese Weise benutzen wir das Zeichen Δ, um eine Differenz zu bezeichnen. Wir zeigen eine Summe mit dem Zeichen Σ an. ... Daraus folgt, daß die Gleichung $z = \Delta y$, wenn sie gefunden wird, auch $y = \Sigma z + C$ ergibt." aus L. Eulero "Institutiones Calculi Differentialis cum eius usu in Analysi Finitorum ac Doctrina Serium.", Berlin, Academiæ Imperialis Scientiarum Petropolitanæ, 1755. Zitiert nach [7], S.48.

die Rekursion

$$\frac{64}{3}\,\frac{n^2-3\,n+2}{(3\,n-5)\,(3\,n-1)}A(n-3)+\frac{16}{3}\,\frac{33\,n^3-106\,n^2+102\,n-29}{(3\,n-5)\,(9\,n^2-9\,n+2)}A(n-2)+$$

$$+\frac{4}{3}\,\frac{37\,n^2-42\,n+11}{9\,n^2-9\,n+2}\,A(n-1)+A(n)=0 \tag{6}$$

aus, obwohl die Rekursion

$$A(n+1)=(-4)\,A(n) \tag{7}$$

ebenfalls gültig ist, mit der man leicht die geschlossene Form $A(n)=(-4)^n$ findet.
Da für die Berechnung der geschlossenen Form eine Rekursion der Ordnung 1 notwendig ist, kommt es vor, daß der Algorithmus eine existierende geschlossene Form nicht findet.
Im Fall, daß der Algorithmus eine Rekursion der Ordnung $l>1$ liefert, kann man allerdings den Petkovšekalgorithmus (Kapitel 4) anwenden. Dieser findet zu gegebener Rekursion

$$\sum_{i=0}^{l}\sigma_i\,(n)\;A\,(n+i)=0 \tag{8}$$

mit **polynomiellen** $\sigma_i\,(n)$ und $l>1$ eine Rekursion

$$A\,(n+1)=S\,(n)\;A\,(n)\,,$$

sofern diese existiert.
Um von der Ausgabe (4) des Zeilbergeralgorithmus zur Eingabe (8) des Petkovšekalgorithmus zu kommen, braucht man nur mit dem Hauptnenner zu multiplizieren und n durch $n+l$ zu ersetzen. In unserem Beispiel etwa wandeln wir die Ausgabe (6) in

$$0=64\;(n+2)\,(n+1)\,(3\,n+6)\,A\,(n)+16\;(n+2)\left(33n^2+125\,n+107\right)A(n+1)+$$

$$4\;(3\,n+4)\left(-42\,n-115+37\;(n+3)^2\right)A(n+2)+3\;(3\,n-1)\,(3\,n+4)\,(3\,n+7)\,A(n+3)$$

um, und mit dieser Rekursion als Eingabe liefert der Petkovšekalgorithmus die Rekursion (7).
Das obige Beispiel (5) wird im Laufe der Arbeit auf alle drei vorgestellten Algorithmen angewendet, jeweils am Ende jedes Beispielabschnitts. Dieses soll zusätzlich den Zusammenhang als algorithmische Instrumentarium verdeutlichen, den die Algorithmen bieten, um hypergeometrischen Summen (mit beschränkten Trägern) auszurechnen. Trotzdem werden damit die traditionellen Methoden zur Bestimmung von Identitäten — Formelsammlungen, bereits bewiesenene Identitäten, Tricks, etc. — nicht überflüssig. Es gibt nämlich wenige generische Identitäten für Summen. Meistens werden Abhängigkeiten zwischen Eingabeparametern gefordert. So gibt es zwar eine Lösung für

$$\sum_k \binom{n}{k}\binom{n+1}{k},$$

aber keine Lösung für

$$\sum_k \binom{n}{k}\binom{m}{k}.$$

Für Abhängigkeiten wie $m = n + 1$ liefern die Algorithmen keine Anhaltspunkte. Sie sind in erster Linie ein Hilfsmittel zum **Überprüfen** von Vermutungen, und nicht zuletzt auch zum **Widerlegen** von Vermutungen.
Die Idee zu dieser Arbeit entstand während meiner Tätigkeit am Konrad-Zuse-Zentrum für Informationstechnik Berlin. Dort war ich von April 1994 bis März 1995 in der Abteilung Symbolik als Forschungstutor beschäftigt, und implementierte unter anderem zusammen mit Dr. Wolfram Koepf den Gosper- und den Zeilbergeralgorithmus in REDUCE [9] und in MAPLE . Durch die Implementierung habe ich ein Verständnis vom Zusammenhang der Verfahren untereinander entwickelt, und Einblick in die Möglichkeiten und Grenzen bei ihrer Anwendung auf Computer gewonnen. Ich hoffe davon möglichst viel an die Leserschaft weitergeben zu können. Ein besonderes Anliegen ist mir dabei die drei Algorithmen als **zusammenhängendes** Instrumentarium zur Berechnung von Summen darzustellen. Auch war es mir wichtig, explizite Darstellungen der Beweise dafür zu entwickeln, daß Petkovšeks und Gospers Verfahren Entscheidungsalgorithmen sind. Ich beweise, daß der Zeilbergeralgorithmus nicht immer terminiert und erläutere, daß er für eine große Klasse von Termen (3.19) ein Entscheidungsalgorithmus ist.

Danksagung

Ich danke Dr. Wolfram Koepf für gute und fruchtbare Zusammenarbeit während der Implementierungen und für freundliche und eingehende Betreuung dieser Arbeit.
Auch möchte ich betonen, daß mir das konstruktive, motivierende und herzliche Arbeitsklima in der Abteilung Symbolik des Konrad-Zuse-Zentrums während meiner dortigen Tätigkeit sehr gefallen hat.

Kapitel 1

Theoretische Grundlagen

1.1 Analogien zur Analysis

In diesem Kapitel wird die Analogie zwischen indefiniter Summation und uneigentlicher Integration erläutert. Die Argumentation lehnt sich dabei an [2], S.539-541 und [7], S.47-50 an.
In der Analysis wird der Differentialoperator D durch

$$D\,f\,(x) := \lim_{h \to 0} \frac{f\,(x+h) - f\,(x)}{h}$$

definiert.
In der diskreten Mathematik läßt sich für h der Grenzübergang nicht ausführen, so daß als beste Näherung für h nur ± 1 zur Verfügung stehen; also ergeben sich

$$\triangle s\,(k) := \frac{s\,(k+1) - s\,(k)}{1} = s\,(k+1) - s\,(k)$$

und

$$\bigtriangledown s\,(k) := \frac{s\,(k+(-1)) - s\,(k)}{-1} = s\,(k) - s\,(k-1) \tag{1.1}$$

für $h = 1$ bzw. $h = -1$.
Der Differentialoperator D, angewandt auf Monome, hat die praktische Eigenschaft

$$D\,(x^m) = m\,x^{m-1}. \tag{1.2}$$

Für $\triangle$ und $\bigtriangledown$ ergibt sich leider nichts Ähnliches, z.B.

$$\bigtriangledown \left(x^3\right) = x^3 - (x-1)^3 = 3x^2 - 3x + 1.$$

Dafür gibt es einen anderen Operator, der sich unter $\bigtriangledown$ elegant verändert. Wir führen ihn durch die folgende Definition ein.

Definition 1 (Pochhammersymbol)[1]

$$(a)_k := \begin{cases} 1 & \text{falls } k = 0 \\ a \cdot (a+1) \cdots (a+k-1) & \text{falls } k \in \mathbb{N}. \end{cases} \tag{1.3}$$

□

Es gilt

$$\bigtriangledown ((x)_m) = (x)_m - (x-1)_m = x \cdot (x+1) \cdots (x+m-1) - (x-1) \cdot x \cdots (x+m-2) =$$

$$((x+m-1) - (x-1))\ x \cdot (x+1) \cdots (x+m-2) = m\,(x)_{m-1}.$$

Somit haben wir das erwünschte Analogon zu (1.2)[2] .
In der Analysis lösen wir das bestimmte Integral $\int_a^b f(x)\,dx$ durch Finden einer Stammfunktion g zu f, d.h. $D\,g = f$. Wir erhalten dann

$$g(b) - g(a) = \int_a^b f(x)\,dx.$$

In der diskreten Mathematik berechnen wir die diskrete Stammfunktion $s(k)$ zu $a(k)$, d.h. $a(k) = \bigtriangledown (s(k))$ und erhalten dann

$$s(b) - s(a) = s(b) - s(b-1) + \ldots + s(a+1) - s(a) = \sum_{k=a+1}^{b} a(k) \tag{1.4}$$

durch Teleskopieren.
Allerdings ist auch im diskreten Fall die Berechnung einer Stammfunktion nicht trivial, es sei denn, man hat eine Summe $a(k) = \sum_{i=0}^{m} a_i\,(k)_i$, deren Stammfunktion $s(k) = \sum_{i=0}^{m} a_i \frac{(k)_{i+1}}{i+1} + C$, analog zur stetigen Stammfunktion eines Polynoms, unmittelbar angeben kann.
Der im nächsten Kapitel vorgestellte Gosperalgorithmus [6] findet die diskrete Stammfunktion für eine große Klasse von $a(k)$, nämlich die der hypergeometrischen Terme $a(k)$ mit hypergeometrischer diskreter Stammfunktion $s(k)$. Was ein hypergeometrischer Term ist, klärt der nächste Abschnitt.

1.2 Hypergeometrische Terme

Bevor wir die —für die Algorithmen in dieser Arbeit wichtigen— hypergeometrischen Terme definieren, sollten wir klären, in welchen Körpern wir uns bewegen.
Die in der Arbeit vorgestellten Algorithmen benutzen Faktorisierungsalgorithmen. Deren Funktionieren kann man nur auf $\mathbb{Q}$ garantieren. Deshalb sei der zugrunde gelegte Körper $\mathbb{K}$

[1] Es gibt auch eine erweiterte Definition des Pochhammersymbols für alle $k \in \mathbb{Z}$ (als steigende Faktorielle). Aus verschiedenen Gründen beschränken wir uns hier auf $\mathbb{N}$.

[2] Es gibt natürlich auch einen entsprechenden Operator für $\triangle$, den wir hier nicht erwähnen, da wir ihn nicht brauchen.

entweder $\mathbb{Q}$ oder $\mathbb{Q}$ adjungiert um eine feste Zahl von Konstanten (etwa $\mathbb{K} = \mathbb{Q} \cup \{a, b, c\}$).
Zunächst zwei wichtige **Notationen**:
Es sei $\mathbb{K}(k)$ der Körper von rationalen Funktionen bezüglich der Variablen k über einen Körper $\mathbb{K}$.
Es sei $\mathbb{K}[k]$ der Ring der Polynome in k über $\mathbb{K}$.

Definition 2 (Hypergeometrischer Term)

Ein Term $a(k)$ heißt *hypergeometrisch* in K, wenn

$$\frac{a(k)}{a(k-1)} \in \mathbb{K}(k)$$

gilt.
Die Klasse der hypergeometrischen Terme umfaßt in trivialer Weise Polynome und rationale Funktionen bezüglich k, aber auch Produkte und Summen von Binomialkoeffizienten, weil

$$\frac{\binom{n}{k}}{\binom{n}{k-1}} = \frac{n-k+1}{k} \in \mathbb{K}(k), \tag{1.5}$$

Fakultäten, weil

$$\frac{k!}{(k-1)!} = k \in \mathbb{K}(k) \tag{1.6}$$

und Pochhammersymbole, weil

$$\frac{(a)_k}{(a)_{k-1}} = a + k - 1 \in \mathbb{K}(k) \tag{1.7}$$

gilt.
Um hypergeometrische Terme als solche zu erkennen, ist eine Vereinheitlichung sicher nützlich. Eine Funktion, auf die sich alle der drei oben genannten Funktionen zurückführen lassen, ist die Eulersche Gamma-Funktion. Wir definieren sie wie bei [15], S. 143ff.

Definition 3 *Die Gamma - Funktion*

$$\Gamma(z) := \int_0^\infty e^{-t}\, t^{z-1}\, dt$$

Obwohl sich die Gamma-Funktion auch für komplexe Veränderliche definieren läßt, beschränken wir uns hier zunächst auf $z \in \mathbb{K}$, also auf rationale z.
Fakultäten, Binomialkoeffizienten und Pochhammersymbole lassen sich mit Hilfe der Gammafunktion darstellen. Die Darstellungen lassen sich aus der folgenden einfachen, aber bedeutungsvollen Rekursionsgleichung (1.8) herleiten:

$$\Gamma(z+1) = \int_0^\infty e^{-t}\, t^z\, dt \; - t^z\, e^{-t}\Big|_{t=0}^{t=\infty} = z\,\Gamma(z) \tag{1.8}$$

□

1. Fakultäten:
   Durch Induktion über $n \in \mathbb{N}$ auf (1.8) beweist man

$$\Gamma(z+n) = z \cdot (z+1) \cdots (z+n-1)\,\Gamma(z)\,. \tag{1.9}$$

   $\Gamma(1) = \int_0^\infty e^{-t}\,dt = 1 = 0!$ und für $n > 0$ ergibt sich aus (1.9) mit $z = 1$:

$$\Gamma(1+n) = 1 \cdot 2 \ldots n \cdot \Gamma(1) = n!$$

   Die Fakultätsfunktion läßt sich also für alle $n \in \mathbb{N}$ mit

$$n! = \Gamma(n+1) \tag{1.10}$$

   darstellen.

2. Binomialkoeffizienten sind durch

$$\binom{z}{n} := \frac{z \cdot (z-1) \cdots (z-n+1)}{n!}$$

   für alle $n \in \mathbb{N}_0$ definiert.
   Falls $z - n$ keine negative ganze Zahl ist, kann man mit $\Gamma(z-n+1)$ erweitern.

$$\binom{z}{n} = \frac{(z-n+1) \cdot (z-n+2) \cdots z \cdot \Gamma(z-n+1)}{n! \cdot \Gamma(z-n+1)} =$$

   wegen (1.9) mit $(z \mapsto z-n+1)$ und wegen (1.10)

$$\frac{\Gamma(z+1)}{\Gamma(n+1) \cdot \Gamma(z-n+1)}. \tag{1.11}$$

   Ist $z - n$ eine negative ganze Zahl, so gilt

$$z \cdot (z-1) \cdots (z-n+1) = z \cdot (z-1) \cdots (z-z) \cdots (z-n+1) = 0 \quad \Longrightarrow$$

$$\binom{z}{n} = 0. \tag{1.12}$$

3. Für Pochhammersymbole, deren zweites Argument 0 ist, gilt nach Definition

$$(z)_0 = 1.$$

   Sonst muß man eine Fallunterscheidung machen. Für $z \neq \{0, -1, -2, \ldots\}$ gilt

$$(z)_n \overset{(1.3)}{=} z \cdot (z+1) \cdots (z+n-1) \overset{(1.9)}{=} \frac{\Gamma(z+n)}{\Gamma(z)}\ . \tag{1.13}$$

   Für Pochhammersymbole mit $z \in \{0, -1, -2, \ldots\}$ multiplizieren wir alle Faktoren mit -1;

$$(z)_n = z \cdot (z+1) \cdots (z+n-1) = (-1)^n\ (-z)\ (-(z-1)) \cdots (-(z-n+1)) =$$

$$(-1)^n\ (-z)_n = (-1)^n\ \frac{\Gamma(-z+n)}{\Gamma(-z)}\ . \tag{1.14}$$

Wir kommen nun zu Reihen von hypergeometrischen Termen.

Definition 4 *Hypergeometrische Reihe*

Eine *hypergeometrische Reihe* ist eine Reihe

$$\sum_{k=0}^{\infty} a\left(k\right) \ ,$$

bei der $a\left(k\right)$ hypergeometrischer Term in $\mathbb{K}$ ist.
Hypergeometrische Reihen lassen sich auch als Potenzreihen

$$\sum_{k=0}^{\infty} a\left(k\right) \ x^k$$

mit $a\left(0\right) = 1$ und $x \in \mathbb{K}$ schreiben. □

Eine hypergeometrische Reihe ist vergleichbar mit einem Integral $\int_0^\infty f\left(t\right) dt$. Mit Methoden aus der Analysis besorgen wir uns nun Werkzeuge, um Potenzreihen auszurechnen. Dabei hat sich folgende Notation bewährt.
Notation für hypergeometrische Reihen:

$${}_p\mathrm{F}_q \left(\begin{array}{cccc} a_1 & a_2 & \cdots & a_p \\ b_1 & b_2 & \cdots & b_q \end{array} \middle| x \right) = \sum_{k=0}^{\infty} \frac{\left(a_1\right)_k \cdot \left(a_2\right)_k \cdots \left(a_p\right)_k}{\left(b_1\right)_k \cdot \left(b_2\right)_k \cdots \left(b_q\right)_k \, k!} x^k \tag{1.15}$$

Sie ist kompliziert, aber, wie sich im folgenden zeigen wird, außerordentlich nützlich für alle hypergeometrischen Reihen, deren Terme $a\left(k\right)$ den Voraussetzungen des folgenden Lemmas genügen.

Lemma 1 *Es sei $a\left(k\right)$ ein hypergeometrischer Term, für den $a_1, a_2, \ldots, a_p, b_1, b_2, \ldots, b_q, x \in \mathbb{K}$ und $p, q \in \mathbb{N}_0$ existieren, so daß es eine Darstellung*

$$\frac{a\left(k+1\right)}{a\left(k\right)} = \frac{\left(a_1 + k\right) \cdots \left(a_p + k\right) x}{\left(b_1 + k\right) \cdots \left(b_q + k\right)} \tag{1.16}$$

gibt.[3]
Dann existieren $A_1, A_2, \ldots, A_P, B_1, B_2, \ldots, B_Q, C, x \in \mathbb{K}$, so daß

$$\sum_{k=0}^{\infty} a\left(k\right) = C \ {}_P\mathrm{F}_Q \left(\begin{array}{cccc} A_1 & A_2 & \cdots & A_P \\ B_1 & B_2 & \cdots & B_Q \end{array} \middle| x \right) \tag{1.17}$$

gilt.
Falls ein minimales $b_j \in \mathbb{Z}$, $i \in \{1, \ldots, q\}$ existiert, und $a_i \leq -b_j + 1$ für alle $a_i \in \mathbb{Z}$, $i \in \{1, \ldots, p\}$ gilt, gibt es sogar eine Darstellung

$$\sum_{k} a\left(k\right) = C \ {}_P\mathrm{F}_Q \left(\begin{array}{cccc} A_1 & A_2 & \cdots & A_P \\ B_1 & B_2 & \cdots & B_Q \end{array} \middle| x \right) . \tag{1.18}$$

[3] $a_1, a_2, \ldots, a_p$ und $b_1, b_2, \ldots, b_q$ sind das Negative der Nullstellen von Zähler- bzw. Nennerpolynom von $a\left(k+1\right) / a\left(k\right)$.

Beweis:
Fall 1: $b_i \notin \mathbb{Z}$ für alle $i \in \{1, 2, \ldots, q\}$
Falls im Quotienten (1.16) kein $b_i \in \mathbb{Z}$ steht, erweitern wir um $(k+1)$.

$$\frac{a\,(k+1)}{a\,(k)} = \frac{(a_1+k)\cdots(a_p+k)\,(k+1)\,x}{(b_1+k)\cdots(b_q+k)\,(k+1)} \quad \Longrightarrow$$

$$a\,(k) = \frac{(a_1+k-1)\cdots(a_p+k-1)\,k\,x}{(b_1+k-1)\cdots(b_q+k-1)\,k}\,a\,(k-1)\,.$$

Diese Rekursion ist wohldefiniert, da für kein $k \in \mathbb{N}$ eine Null im Nenner steht.
Durch Umbenennen erhalten wir

$$a\,(k) = \frac{\prod\limits_{i=1}^{P}(A_i+k-1)x}{\prod\limits_{i=1}^{Q}(B_i+k-1)k}\,a\,(k-1) \tag{1.19}$$

mit

$$\begin{aligned} &P := p+1,\\ &Q := q,\\ &A_i := a_i \text{ für alle } i \in \{1,2,\ldots,p\}\,,\\ &A_P := 1 \text{ und}\\ &B_i := b_i \text{ für alle } i \in \{1,2,\ldots,q\}\,. \end{aligned} \tag{1.20}$$

Induktiv ergibt sich

$$a\,(k) = \frac{\prod\limits_{i=1}^{P}\left(\prod\limits_{j=1}^{k}(A_i+k-j)\right)x^k}{\prod\limits_{i=1}^{P}\left(\prod\limits_{j=1}^{k}(A_i+k-j)\right)k!}\,a\,(0) = \frac{(A_1)_k\cdot(A_2)_k\cdots(A_P)_k\,x^k}{(B_1)_k\cdot(B_2)_k\cdots(B_Q)_k\,k!}\,a\,(0)\,. \tag{1.21}$$

Wir erhalten (1.17) mit $C = a\,(0)$. □

Fall 2: Es gibt ein minimales $b_j \in \mathbb{Z}$, $j \in \{1, 2, \ldots, q\}$
Gibt es ein minimales $b_j \in \mathbb{Z}$, so erstzen wir k durch $k - b_j + 1$ in (1.17). Wir erhalten

$$\frac{a\,(k-b_j+2)}{a\,(k-b_j+1)} = \frac{\prod\limits_{i=1}^{p}(a_i-b_j+1+k)\;x}{\prod\limits_{i=1,i\neq j}^{q}(b_i-b_j+1+k)\;(k+1)\,.}$$

Analog zur Berechnung von (1.21) ergibt sich

$$a\,(k-b_j+1) = \frac{\prod\limits_{i=1}^{p}(a_i-b_j+1)_k x^k}{\prod\limits_{i=1,i\neq j}^{q}(b_i-b_j+1)_k k!}\,a\,(-b_j+1) =$$

$$\frac{(A_1)_k \cdot (A_2)_k \cdots (A_P)_k \, x^k}{(B_1)_k \cdot (B_2)_k \cdots (B_Q)_k \, k!} \, C$$

mit

$$\begin{array}{l} P := p, \\ Q := q - 1, \\ C := a(-b_j + 1), \\ A_i := a_i - b_j + 1 \,\text{für alle}\; i \in \{1, 2, \ldots, P\}\,, \\ B_i := b_i - b_j + 1 \;\text{für alle}\, i \in \{1, 2, \ldots, j-1\} \;\text{und} \\ B_i := b_{i+1} - b_j + 1 \,\text{für alle}\; i \in \{j, \ldots, Q\}\,. \end{array} \tag{1.22}$$

Für $A(k) := a(k - b_j + 1)$ gilt dann

$$A(k) = \frac{(b_1 - b_j + 1 + k) \cdots (b_q - b_j + 1 + k) \cdot (k+1)}{(a_1 - b_j + 1 + k) \cdots (a_p - b_j + 1 + k)\, x} \, A(k+1)\,. \tag{1.23}$$

□

Fall 2a: Es gilt $a_i \leq -b_j + 1$ für alle $i \in \{1, \ldots, p.\}$
Falls $a_i \leq -b_j + 1$ für alle $a_i \in \mathbb{Z}$, $i \in \{1, \ldots, p\}$ gilt, steht im Nenner nie Null für negative k.
Wegen

$$A(-1) = 0$$

folgt somit iterativ

$$A(k) = 0$$

für $k = -2, -3, \ldots$
Gemäß (1.22) gilt somit

$$C \, {}_P\mathrm{F}_Q \left(\begin{array}{cccc} A_1 & A_2 & \cdots & A_P \\ B_1 & B_2 & \cdots & b_Q \end{array} \middle| \, x \right) = \sum_{k=0}^{\infty} A(k) = \sum_k A(k) = a(-b_j + 1) \sum_k a(k)$$

mit $:= a(-b_j + 1)$. □

Fall 2b: Es gibt ein $a_i \in \mathbb{Z}$ mit $a_i > -b_j + 1$
Gibt es ein $a_i > -b_j + 1$, so ist die Rekursion (1.23) für $k = -a_i + b_j - 1 < 0$ nicht definiert, da eine Null im Nenner steht. Somit können wir nicht von $\sum_k A(k) = \sum_{k=0}^{\infty} A(k)$ ausgehen. Immerhin erhalten wir aber (1.17) mit Parametern gemäß (1.22) und $C := a(-b_j + 1)$. □

Beispiele:

1. Es sei

$$a(k) := \frac{\Gamma(a+k)\,\Gamma(b)\,\Gamma(c)}{\Gamma(a)\,\Gamma(b+k)\,\Gamma(c+k)} \;,$$

dann ist

$$a(0) = 1$$

und

$$\frac{a\,(k+1)}{a\,(k)} = \frac{(a+k)}{(b+k)\,(c+k)}$$

also

$$\sum_{k=0}^{\infty} a\,(k) = {}_2\mathrm{F}_2\left(\begin{array}{cc} a & 1 \\ b & c \end{array} \middle| 1 \right) \ .$$

2. (a) Es sei

$$a\,(k) := k \binom{2\,n}{k},$$

dann ist

$$\frac{a\,(k+1)}{a\,(k)} = -\frac{k-2\,n}{k},$$

$b_j = 0$ und $A\,(0) = a\,(1) = 2\,n$ und somit

$$\sum_k a\,(k) = 2\,n\ {}_1\mathrm{F}_0\left(\begin{array}{c} -2\,n+1 \\ - \end{array} \middle| -1 \right) \ .$$

(b) Es sei

$$a\,(k) := 3\,\frac{(\,2+k\,)!\,(-n)_k}{(\,3+k\,)!\,(\,k-2\,)!,}$$

dann ist

$$\frac{a\,(k+1)}{a\,(k)} = \frac{(\,3+k\,)\,(\,-n+k\,)}{(\,4+k\,)\,(\,k-1\,),}$$

$$b_j := \min\{4,-1\} = -1$$

und somit

$$A\,(k) = a\,(k+1+1) = 3\,\frac{(\,4+k\,)!\,(-n)_{2+k}}{(\,5+k\,)!\,k!} \ .$$

Nun gibt es aber ein $a_i = 3 > -b_j + 1 = 2$. Wie man auch deutlich sieht, ist die Rekursion

$$A\,(k) = \frac{(\,4+k\,)\,(\,-n+1+k\,)}{(\,5+k\,)\,k} A\,(k+1)$$

für $k = -a_i + b_j - 1 = -5$ nicht wohldefiniert, sie gilt aber für alle $k \in \mathbb{N}$.
Wir können nun immerhin die Gleichung

$$\sum_{k=0}^{\infty} a\,(k) = a\,(-b_j+1)\ {}_2\mathrm{F}_1\left(\begin{array}{cc} 3-b_j+1 & n-b_j+1 \\ 4-b_j+1 & \end{array} \middle| 1 \right) =$$

$$\frac{3}{5}\,n\ (n-1)\ {}_2\mathrm{F}_1\left(\begin{array}{cc} 5 & -n+2 \\ 6 & \end{array} \middle| 1 \right)$$

aufstellen.

□

Bemerkung 1 *zu Nullstellen in* $\mathbb{K} = \mathbb{C}, \mathbb{Q}$

Legt man $\mathbb{K} = \mathbb{C}$ zugrunde, gibt es zu jedem hypergeometrischen Term $a(k)$ eine Darstellung (1.16). Allerdings gibt es keine Faktorisierungsalgorithmen auf $\mathbb{C}(k)$ um die Nullstellen des Quotienten $a(k+1)/a(k)$ auch zu finden.
Wohl aber gibt es solche Algorithmen für $\mathbb{Q}(k)$ (siehe [20] Seiten 195-204). □

Terme von Potenzreihen, bei denen die Nullstellen des Zähler- und und des Nennerpolynoms von $a(k)/a(k-1)$ sämtlich in $\mathbb{K}$ liegen, sind nur ein Spezialfall des nun folgenden Gosperalgorithmus. Die Notation (1.15) verlangt sicherlich noch Gewöhnung. Sie wird sich aber in den folgenden Kapiteln als sehr nützlich erweisen. So kann man von einer Eingabe ${}_pF_q$ nicht nur erwarten, daß sie hypergeometrisch ist, sondern kann mit Hilfe von (1.16) den Quotienten $a(k)/a(k-1)$ beinahe unmittelbar hinschreiben.
Mit Hilfe der Notation (1.15) kann man auch Aussagen über die Konvergenz der Reihe direkt ablesen, wie die folgende Bemerkung zeigt.

Bemerkung 2 *Zur Konvergenz von* ${}_p\mathrm{F}_q$

Ist

1. $p \leq q$,
2. $p = q+1$ und $|x| < 1$ oder
3. $p = q+1$, $|x| = 1$ und $\sum_{i=1}^{q} b_i - \sum_{i=1}^{q+1} a_i > 0$,

so konvergiert die Reihe ${}_p\mathrm{F}_q(x)$.
Beweis: Für $p \leq q$ gilt

$$\lim_{k\to\infty} \frac{a(k+1)}{a(k)} = \lim_{k\to\infty} \frac{(b_1+k)\cdots(b_q+k)\,x}{(a_1+k)\cdots(a_p+k)\,(k+1)} = 0,$$

und für $p = q+1$ und $|x| < 1$ gilt

$$\lim_{k\to\infty} \frac{a(k+1)}{a(k)} = x.$$

In beiden Fällen konvergiert ${}_p\mathrm{F}_q$ offensichtlich nach dem Cauchyschen Konvergenzkriterium.
Interessanter ist der Fall 3:
Wir setzen

$$\delta := \frac{1}{2}\left(\sum_{i=1}^{q} b_i - \sum_{i=1}^{q+1} a_i\right)$$

und vergleichen die Terme der Reihe

$${}_{q+1}\mathrm{F}_q\left(\left.\begin{matrix} a_1 & a_2 & \cdots & a_{q+1} \\ b_1 & b_2 & \cdots & b_q \end{matrix}\right| x\right) = 1 + \sum_{k=1}^{\infty} \frac{\prod_{i=1}^{q+1}(a_i)_k}{\prod_{i=1}^{q}(b_i)_k\, k!} x$$

mit der Reihe

$$\sum_{k=1}^{\infty} \frac{1}{k^{1+\delta}} \ , \tag{1.24}$$

die bekanntermaßen konvergiert.

$$\lim_{k\to\infty} \left| \frac{\frac{\prod_{i=1}^{q+1} (a_i)_k}{\prod_{i=1}^{q} (b_i)_k \, k!} x^k}{\frac{1}{k^{1+\delta}}} \right|$$

erweitert um

$$(k-1)!^{q+1} \, k^{\left(\sum_{i=1}^{q} b_i - \sum_{i=1}^{q+1} a_i\right)} = (k-1)!^{q+1} \, k^{2\delta}$$

mit $|x| = 1$ gibt

$$\lim_{k\to\infty} \prod_{i=1}^{q+1} \frac{(a_i)_k}{(k-1)! \, k^{a_i}} \prod_{i=1}^{q} \frac{(k-1)! \, k^{b_i}}{(b_i)_k} \frac{(k-1)! \, k^{1+\delta}}{k!} =$$

$$\frac{\prod_{i=1}^{q} \Gamma(b_i)}{\prod_{i=1}^{q+1} \Gamma(a_i)} \lim_{k\to\infty} \frac{1}{k^{2\,\delta-\delta}} = 0, \tag{1.25}$$

wobei der Beweis von

$$\lim_{n\to\infty} \frac{(a)_n}{(k-1)! n^a} = \Gamma(a)$$

in [14], S.23 nachgelesen werden kann. Man beachte, daß (1.25) für $\mathbb{K} = \mathbb{Q}$ nicht notwendig folgt, da (1.24) nur für $\mathrm{Re}(\delta) > -1$ konvergiert. Ferner beachte man, daß der dritte Teil der Bemerkung nur dann brauchbar ist, wenn sich alle asoziierten Konstanten in der Summe wegheben. □

Im folgenden werden wir uns nicht mehr mit Konvergenzbetrachtungen dieser Art beschäftigen. Deswegen sind obige Ausführungen auch nur als “Bemerkung” ausgezeichnet.

Kapitel 2

Der Gosperalgorithmus

2.1 Kurzer Überblick über den Algorithmus

Gosper[6] findet zu gegebenem hypergeometrischem Term $a(k)$ eine hypergeometrische Stammfunktion $s(k)$ mit

$$a(k) = \bigtriangledown s(k) = s(k) - s(k-1), \tag{2.1}$$

falls eine solche Funktion $s(k)$ existiert. Er **entscheidet** somit auch, ob sie existiert.
Zunächst findet Gosper algorithmisch eine Darstellung von $a(k)/a(k-1)$ durch Polynome.
Die Idee dabei ist folgende:
Falls wir $a(k)$ aus $s(k)$ berechnen können, dann besteht $a(k)$ aus einem Produkt von einem Polynom $p(k)$ mit einem hypergeometrischen Term $b(k)$, z.B. $s(k) = (2k)!/k!$:

$$a(k) = s(k) - s(k-1) = \frac{(2k)!}{k!} - \frac{(2k-2)!}{(k-1)!} =$$

$$((2k)(2k-1) - k) \cdot \frac{(2k-2)!}{k!} = k(4k-3) \cdot \frac{(2k-2)!}{k!}$$

mit $p(k) = k(4k-3)$ und dem hypergeometrischen Term $b(k) = (2k-2)!/k!$, für den also $b(k)/b(k-1) = q(k)/r(k)$ rational ist. Gosper zeigt, daß es solche Polynome $p(k)$, $q(k)$ und $r(k)$ mit

$$\frac{a(k)}{a(k-1)} = \frac{p(k)}{p(k-1)} \frac{q(k)}{r(k)} \tag{2.2}$$

geben muß, die die Gleichung (2.7),

$$\operatorname{ggt}(q(k), r(k+j)) = 1$$

für alle $j \in \mathbb{N}_0$ erfüllen, und gibt einen Algorithmus um sie zu berechnen.
Gosper definiert die Funktion $f(k)$ durch

$$f(k) := \frac{s(k) \cdot p(k)}{a(k)\ q(k+1)} \tag{2.3}$$

und zeigt, daß $f(k)$ ein Polynom sein muß.
Nach (2.3) folgt dann

$$a(k) = s(k) - s(k-1) = \frac{q(k+1)}{p(k)} f(k)\ a(k) - \frac{q(k)}{p(k-1)} f(k-1)\ a(k-1), \qquad (2.4)$$

und (2.3) multipliziert mit $p(k)/a(k)$ gibt die Rekursionsgleichung

$$p(k) = q(k+1)\ f(k) - \frac{a(k-1)}{a(k)} \frac{p(k)}{p(k-1)} q(k)\ f(k-1) \overset{(2.2)}{=}$$

$$q(k+1)\ f(k) - r(k)\ f(k-1)\ . \qquad (2.5)$$

Gosper berechnet eine obere Schranke für den Grad von $f(k)$ (Lemma 8). Hat er diese, kann er $f(k)$ mit Hilfe der Methode der unbekannten Koeffizienten aus (2.5) bestimmen. $s(k)$ ergibt sich anschließend aus (2.3). □
Es bleiben noch folgende Fragen zu klären:

1. Wie erkennt man, ob $a(k)$ ein hypergeometrischer Term ist?
2. Wie berechnet man $p(k)$, $q(k)$ und $r(k)$?
3. Wie zeigt man, daß $f(k)$ ein Polynom sein muß?
4. Wie berechnet man eine obere Schranke für den Grad von $f(k)$?
5. Warum ist $s(k)$ eindeutig bis auf eine Konstante?

Diese Fragen behandelt der nächste Abschnitt.

2.2 Teilaspekte des Algorithmus

2.2.1 Erkennung von hypergeometrischen Termen

Für den Gosperalgorithmus wird gefordert, daß $a(k)$ hypergeometrisch ist. Gosper erreicht dies, indem er Zähler- und Nennerpolynom von $a(k)/a(k-1)$ als Eingabe fordert. Besser ist es natürlich, wenn $a(k)$ als Eingabe genügt. Koornwinder [10] fordert in seiner Weiterentwicklung von Zeilbergers MAPLE - Implementierung statt eines hypergeometrischen Terms $a(k)$ die Parameter der zugehörigen hypergeometrischen Reihe in der Notation (1.15). Das vereinfacht zwar die Berechnung von $a(k)/a(k-1)$ (siehe Lemma 1), schränkt aber die Eingabemöglichkeiten extrem ein. So ist es nicht möglich, mit Koornwinders Implementierung

$$\sum_k k \binom{n}{k} = n\,2^{n-1} = {}_1\mathrm{F}_0\left(\begin{matrix} -n+1 \\ - \end{matrix} \middle| -1 \right) n$$

auszurechnen.
Zeilberger selbst läßt als Eingabe für $a(k)$ Quotienten von Produkten von Polynomen und Fakultäten, wie z.B.

$$a(k) = \frac{(a+k)!\,(b+k)!}{k^2\,(c+k)!}$$

zu.
Das ist sicherlich sinnvoll, da sich Pochhammersymbole, Binomialkoeffizienten und Gammaterme ohne große Mühe in Fakultäten umwandeln lassen.
Zeilberger wendet anschließend die MAPLE - Funktion `expand` auf $a(k)/a(k-1)$ an. Die Funktion `expand` sorgt unter anderem für das Expandieren der Fakultäten. Z.B.:

```
> expand(factorial(n-1));
```

$$\frac{n!}{n}$$

Damit erkennt man natürlich, daß $a(k) = n!$ hypergeometrisch ist:

```
> factorial(n)/subs(n=n-1,factorial(n));
```

$$\frac{n!}{(n-1)!}$$

```
> expand(");
```

$$n$$

```
> type(",ratpoly(anything,n));
>
```

$$true$$

Der Nachteil dieser Strategie liegt allerdings bei Termen wie $a(k) = (n+1000000)!$. Offensichtlich ist $a(k)/a(k-1) = (n+1000000)$, aber `expand` expandiert zuerst, und vereinfacht dann. Daher liefert die obige MAPLE - Sitzung mit $a(k) = (n+1000000)!$ kein Ergebnis:

```
> factorial(n+1000000)/subs(n=n-1,factorial(n+1000000));
```

$$\frac{(n+1000000)!}{(n+999999)!}$$

```
> expand(");
Error, (in expand/factorial) object too large
```

Wolfram Koepf löst dieses Problem, indem er $a(k)/a(k-1)$ mit dem folgenden Algorithmus 1 zu einem rationalen Ausdruck vereinfacht , sofern dies möglich ist.
Er wandelt dabei nicht in Fakultäten um, sondern in Gammaterme.
Sicherlich ist der Definitionsbereich für Gammaterme größer als der für Fakultäten. Da wir aber mit Symbolen rechnen, spielt das kaum eine Rolle. Die entscheidende Neuerung bei Koepf ist, daß der Algorithmus ohne Expandieren von Termen auskommt. Auch berücksichtigt er die Division von Potenzen mit gleicher Basis.

Algorithmus 1 *Algorithmus zur kombinatorischen Vereinfachung eines Ausdrucks*

Eingabe: $b(k)$
Ausgabe: ein äquivalenter Ausdruck in $\mathbb{K}(k)$, falls dieser existiert (sonst irgendein äquivalenter Ausdruck).

1. Wandele alle Fakultäten, Binomialkoeffizienten und Pochhammer-Symbole in Γ-Ausdrücke um (gemäß (1.10) - (1.14)).

2. Kürze Γ-Ausdrücke mit ganzzahliger Differenz in Zähler- und Nennerpolynom, gemäß (1.9). Z.B.
$$\frac{\Gamma(k)}{\Gamma(k+2)} \to \frac{1}{(k+1)(k+2).}$$

3. Expandiere alle Terme b^{k+a} zu $b^k b^a$.

4. Kürze Potenzen mit ganzzahligem Abstand im Exponenten und gleicher Basis. Z.B.
$$b^a b^{3-a} \to b^{-3}.$$

□

Ein Beispiel:

$$b(k) = \frac{\binom{n}{k} a^k (k-1)! \Gamma(n-k+1) b^k}{(n+2)! a^{k+2} (b+1)^k} =$$

1.
$$\frac{\frac{\Gamma(n+1)}{\Gamma(k+1)\Gamma(n+1-k)} a^k \Gamma(k) \Gamma(n-k+1) b^k}{\Gamma(n+3) a^{k+2} (b+1)^k} =$$

2.
$$\frac{\Gamma(n+1)}{\Gamma(n+3)} \cdot \frac{\Gamma(k)}{\Gamma(k+1)} \cdot \frac{\Gamma(n-k+1)}{\Gamma(n-k+1)} \cdot \frac{b^k}{(b+1)^k} \cdot \frac{a^k}{a^{k+2}}$$

$$\frac{1}{(n+2)(n+3)} \cdot \frac{1}{k} \cdot 1 \cdot \frac{b^k}{(b+1)^k} \cdot \frac{a^k}{a^{k+2}}$$

3.
$$\frac{1}{(n+2)(n+3)} \cdot \frac{1}{k} \cdot \frac{b^k}{(b+1)^k} \cdot \frac{a^k}{a^k a^2}$$

4.

$$\frac{b^k}{(n+2)\ (n+3)\ k\ (b+1)^k\ a^2}$$

Bei Rechnung mit der Hand braucht man sicherlich nicht den Umweg über die Γ -Funktion zu gehen, sondern kann die einzelnen Teilausdrücke $b_i(k)$ von $a(k)$ direkt in $a_i(k)\, a_i(k-1)$ umwandeln.

Beispiel:

Es sei

$$a(k) := \binom{n}{k} (-1)^k,$$

dann ist

$$\frac{a(k)}{a(k-1)} = \frac{\binom{n}{k}}{\binom{n}{k-1}} \cdot \frac{(-1)^k}{(-1)^{k-1}} = \frac{n-k}{k+1}(-1).$$

2.2.2 Bestimmung von $p(k)$, $q(k)$ und $r(k)$

Der folgende Algorithmus 2 berechnet die Polynome $p(k)$, $q(k)$ und $r(k)$ für gegebenes $b(k) \in \mathbb{K}(k)$. Lemma 2 beweist, daß die mit dem Algorithmus gefundenen Polynome die Bedingungen (2.6) und (2.7) erfüllen. (2.7) wird gebraucht, um zu gewährleisten, daß $f(k)$ ein Polynom ist (Abschnitt 2.2.4).

Lemma 3 wird als Vorbereitung des Beweises der Eindeutigkeit der Polynome $p(k)$, $q(k)$ und $r(k)$ bis auf Konstante Vorfaktoren (Lemma 4) gebraucht. Diese Eindeutigkeit wird wiederum für den Beweis der Eindeutigkeit von $s(k)$ bis auf additive Konstanten in Lemma 9 in Abschnitt 2.2.6 benutzt. Letzteres ist für den Erfolg des Gosperalgorithmus zwar nicht notwendig, unterstreicht aber die Analogie zur uneigentlichen stetigen Integration.

Wirklich gebraucht werden alle drei Lemmata allerdings für den Petkovšekalgorithmus (Kapitel 4).

In den folgenden Kapiteln werden häufig **größte gemeinsame Teiler (ggt)** von je zwei Polynomen aus $\mathbb{K}[k]$ gebraucht. Eine Beschreibung ihrer Berechnung würde hier allerdings zu weit führen. Sie kann in [20] nachgelesen werden.

Zu bemerken ist allerdings, daß der ggt, wie in jedem Ring, nur bis zur Multiplikation mit einem Einselement eindeutig ist. Die Einselemente (oder auch invertierbare Elemente) des $\mathbb{K}[k]$ sind alle (konstanten) Elemente aus $\mathbb{K}$.

So ist ggt $(k^2, 2k+1) = -37$ keine falsche, aber eine ungewöhnliche Aussage. Im Ring $\mathbb{K}[k]$ ist es üblich, für den ggt immer das **monische** Polynom an- bzw. auszugeben. Monisch ist ein Polynom dann, wenn sein führender Koeffizient gleich 1 ist. In Computeralgebrasystemen wird diese Konvention allerdings nicht immer eingehalten.

Für den folgenden Algorithmus wird noch eine Definition gebraucht:

Definition 5 *Dispersionsmenge*

$$\mathrm{disp}_k\left(q\left(k\right), r\left(k\right)\right) := \{j \in \mathbb{N}_0 | \, \mathrm{ggt}\left(q\left(k\right), r\left(k+j\right)\right) \neq 1\}$$

heißt Dispersionsmenge von $q\left(k\right)$ und $r\left(k\right)$ bezüglich k. □

Man beachte, daß die Dispersionsmenge nicht symmetrisch bezüglich der Eingabe ist. So ist

$$\mathrm{disp}_k\left(k, k-1\right) = \{1\} \neq \{\} = \mathrm{disp}_k\left(k-1, k\right).$$

Algorithmus 2 *Algorithmus zur Bestimmung von $p\left(k\right), q\left(k\right)$ und $r\left(k\right)$:*

Eingabe: $b\left(k\right) \in \mathbb{K}(k)$

1. $q_0\left(k\right) :=$ Zählerpolynom von $b\left(k\right)$
   $r_0\left(k\right) :=$ Nennerpolynom von $b\left(k\right)$
   $p_0\left(k\right) := 1$
2. (a)
   $$N := \mathrm{disp}_k\left(\, q_0\left(k\right), r_0\left(k\right)\right)$$
   (Die Berechnung der Dispersionsmenge N wird nach dem Beweis in Abschnitt 2.2.3 erläutert.)
   (b) Sortiere N der Größe nach, und definiere $\{N_1, N_2, \ldots, N_l\} := N$ mit $N_1 < N_2 < \ldots N_l$.
3. Für alle $i = 1 \ldots l$ setze:
   (a) $\gamma_i\left(k\right) := \mathrm{ggt}\left(q_{i-1}\left(k\right), r_{i-1}\left(k+N_i\right)\right)$
   (b) $q_i\left(k\right) := \frac{q_{i-1}(k)}{\gamma_i(k)}$
   (c) $r_i\left(k\right) := \frac{r_{i-1}(k)}{\gamma_i(k-N_i)}$
   (d) $p_i\left(k\right) := \prod_{n=1}^{N_i} \gamma_i\left(k-n+1\right) \, p_{i-1}\left(k\right)$
4. $q\left(k\right) := q_l\left(k\right), r\left(k\right) := r_l\left(k\right), p\left(k\right) := p_l\left(k\right)$

□

Beispiel: :
Eingabe:

$$b\left(k\right) := \frac{\left(k+1\right)\left(k+b\right)\left(k+2\right)}{\left(k-1\right)\left(k+b-4\right)}$$

1.

$$q_0(k) := (k+1)(k+b)(k+2),$$

$$r_0(k) := (k-1)(k+b-4)$$

und

$$p_0(k) := 1.$$

2.

$$N := \{N_1, N_2, N_3\} := \{2, 3, 4\}$$

(Bestimmung: siehe Abschnitt 2.2.3)

3. $i = 1$, $N_1 = 2$

(a)

$$\gamma_1(k) := \operatorname{ggt}(q_0(k), r_0(k+2)) = \operatorname{ggt}((k+1)(k+b)(k+2), (k+1)(k+b-4+2)) = k+1$$

(b)

$$q_1(k) := \frac{(k+1)(k+b)(k+2)}{k+1} =$$

$$(k+b)(k+2)$$

(c)

$$r_1(k) := \frac{(k-1)(k+b-4)}{k+1-2} = k+b-4$$

$$p_1(k) := \prod_{n=1}^{2} \gamma_1(k-n+1)\, p_0(k) = \prod_{n=1}^{2} (k+1-n+1) = (k+1)\, k$$

$i = 2$, $N_2 = 3$

(a)

$$\gamma_2(k) := \operatorname{ggt}(q_1(k), r_1(k+3)) = \operatorname{ggt}((k+b)(k+2), (k+b-4+3)) = 1$$

(b)

$$q_2(k) := q_1(k) = (k+b)(k+2)$$

(c)

$$r_2(k) := r_1(k) = k+b-4$$

$$p_2(k) := p_1(k) = (k+1)\, k$$

$i = 3$, $N_3 = 4$

(a)

$$\gamma_3(k) := \operatorname{ggt}(q_2(k), r_2(k+4)) = \operatorname{ggt}((k+b)(k+2), (k+b-4+4)) = k+b$$

(b)

$$q_3(k) := \frac{(k+b)(k+2)}{k+b} = k+2$$

(c)

$$r_3(k) := \frac{k+b-4}{k+b-4} = 1$$

$$p_3(k) : - = \prod_{n=1}^{4} \gamma_3(k-n+1)\ p_2(k) =$$

$$\prod_{n=1}^{4} (k+b-n+1)\ (k+1)\ k$$

4.

$$q(k) := k+2; r(k) = 1;$$

$$p(k) := (k+b)(k+b-1)(k+b-2)(k+b-3)(k+1)\ k$$

□

Lemma 2 *Sei $b(k)$ ein von Null verschiedenes Element in $\mathbb{K}(k)$ also $b(k) \neq 0$, dann gibt es Polynome $p(k), q(k), r(k) \in \mathbb{K}[k]$ so, daß*

$$b(k) = \frac{p(k)}{p(k-1)} \cdot \frac{q(k)}{r(k)} \tag{2.6}$$

mit

$$\operatorname{ggt}(q(k), r(k+j)) = 1, \tag{2.7}$$

für alle $j \in \mathbb{N}_0$ gilt[1].

Beweis:
Wir zeigen, daß Algorithmus 2 Polynome $p(k), q(k)$ und $r(k)$ mit den gewünschten Eigenschaften liefert[2].

[1] Bedingung (2.7) ist äquivalent zur Bedingung $\operatorname{disp}_k(q(k), r(k)) = \{\}$.

[2] Die Beweisführung ist an Petkovšek [13] angelehnt. Allerdings wird in dessen Arbeit im zweiten Schritt des folgenden Algorithmus $N := \{0, 1, \ldots, \max \operatorname{disp}_k(q_0(k), r_0(k))\}$ statt der sortierten Menge $N := \operatorname{disp}_k(q_0(k), r_0(k))$ gesetzt, was die Beweisführung vereinfacht, aber auch redundante Schleifendurchläufe bewirkt.

Beweis von (2.6):
Wir zeigen durch vollständige Induktion, daß (2.6) für alle Tripel $(p_i(k), q_i(k), r_i(k))$ mit $0 \leq i \leq l$ gilt, also insbesondere auch für $(p_l(k), q_l(k), r_l(k)) = (p(k), q(k), r(k))$.
$i = 0$:

$$b(k) = \frac{1}{1} \cdot \frac{q_0(k)}{r_0(k)} = \frac{p_0(k)}{p_0(k-1)} \cdot \frac{q_0(k)}{r_0(k)} \tag{2.8}$$

$i - 1 \mapsto i$:

$$\frac{p_i(k)}{p_i(k-1)} \cdot \frac{q_i(k)}{r_i(k)} = \frac{p_{i-1}(k)}{p_{i-1}(k-1)} \cdot \frac{\prod_{n=1}^{N_i} \gamma_i(k-n+1)}{\prod_{n=1}^{N_i} \gamma_i(k-n)} \cdot \frac{q_{i-1}(k)}{r_{i-1}(k)} \cdot \frac{\gamma_i(k-N_i)}{\gamma_i(k)} =$$

$$\frac{p_{i-1}(k)}{p_{i-1}(k-1)} \cdot \frac{\gamma_i(k)}{\gamma_i(k-N_i)} \cdot \frac{\prod_{n=2}^{N_i} \gamma_i(k-n+1)}{\prod_{n=1}^{N_i-1} \gamma_i(k-n)} \cdot \frac{q_{i-1}(k)}{r_{i-1}(k)} \cdot \frac{\gamma_i(k-N_i)}{\gamma_i(k)} = \frac{p_{i-1}(k)}{p_{i-1}(k-1)} \cdot \frac{q_{i-1}(k)}{r_{i-1}(k)}$$

für $1 \leq i \leq l$ □

Beweis von (2.7):
Nach Konstruktion der Schleife (Schritt 3) folgt erstens, daß jedes $q_i(k)$ und jedes $r_i(k)$ Teiler seiner Vorgänger und sich selbst ist, also

$$q_i(k) \,|q_m(k) \text{ und } r_i(k) \,|r_m(k) \tag{2.9}$$

für $m \leq i \leq l$.
und zweitens

$$\operatorname{ggt}(q_i(k), r_i(k+N_i)) = 1. \tag{2.10}$$

Aus (2.9) und (2.10) folgt

$$\operatorname{ggt}(q_i(k), r_i(k+N_m)) = 1 \tag{2.11}$$

für $0 \leq m \leq i \leq l$.
Insbesondere für $q_l(k) = q(k)$ und $r_N(k) = r(k)$ gilt dann

$$\operatorname{ggt}(q(k), r(k+N_m)) = 1 \tag{2.12}$$

für $0 \leq m \leq l$.
Für $j \in \mathbb{N}_0 \backslash N$ gilt nach Konstruktion in Schritt 2

$$\operatorname{ggt}(q_0(k), r_0(k+j)) = 1$$

und wegen (2.9) mit $m = 0$ und $i = l$

$$\operatorname{ggt}(q(k), r(k+j)) = 1.$$

Wegen (2.12) folgt damit (2.7) für alle $j \in \mathbb{N}_0$. □

Nun kommen wir zum Beweis eines technischen Lemmas, das für den Beweis des nächsten Lemmas gebraucht wird.

Lemma 3 *Seien* $p(k), q(k), r(k) \in \mathbb{K}[k]; b(k) \in \mathbb{K}(k)$, *und gelten (2.6) und (2.7), dann gilt auch*

$$\operatorname{ggt}(q(k), p(k-1)) = 1 \tag{2.13}$$

und

$$\operatorname{ggt}(r(k), p(k)) = 1. \tag{2.14}$$

Beweis von (2.13):
Angenommen $q(k)$ und

$$p(k-1) = p_l(k-1) = \prod_{i=1}^{l} \prod_{n=1}^{N_i} \gamma_i(k-n)$$

haben einen gemeinsamen nichttrivialen Teiler, dann gibt es ein nichttriviales $\gamma_i(k-n)$ mit $1 \leq i \leq l$ und $1 \leq n \leq N_i$, so daß

$$\operatorname{ggt}(\gamma_i(k-n), q(k)) \neq 0 \quad \stackrel{(2.9)}{\Longrightarrow}$$

$$\operatorname{ggt}(\gamma_i(k-n), q_j(k)) \neq 0 \tag{2.15}$$

für alle $0 \leq j \leq l$, also insbesondere für

$$\operatorname{ggt}(\gamma_i(k-n), q_0(k)) \neq 0 \tag{2.16}$$

gilt.
Nach Konstruktion gilt außerdem

$$\gamma_i(k-N_i) \mid r_{i-1}(k) \quad \Longrightarrow$$

$$\gamma_i(k-n) \mid r_{i-1}(k+N_i-n) \quad \stackrel{(2.9)}{\Longrightarrow} \tag{2.17}$$

$$\gamma_i(k-n) \mid r_0(k+N_i-n). \tag{2.18}$$

Aus (2.16) und (2.18) folgt $\operatorname{ggt}(q_0(k), r_0(k+N_i-n)) \neq 1$ und somit
$N_i - n \in \operatorname{disp}_k(q_0(k), r_0(k))$. Also gibt es ein m mit $1 \leq m \leq i$ mit $N_m = N_i - n \in N$.
Aus (2.15) folgt

$$\operatorname{ggt}(\gamma_i(k-n), q_{i-1}(k)) \neq 0$$

und somit wegen (2.17) auch

$$1 \neq \operatorname{ggt}(q_{i-1}(k), r_{i-1}(k+N_i-n)) = \operatorname{ggt}(q_{i-1}(k), r_{i-1}(k+N_m)).$$

Dies ist jedoch ein Widerspruch zu (2.11). □

Völlig analog geht der
Beweis von (2.14).
Angenommen $r(k)$ und

$$p(k) = p_l(k) = \prod_{i=1}^{l} \prod_{n=1}^{N_i} \gamma_i(k-n+1)$$

haben einen gemeinsamen nichttrivialen Teiler, dann gibt es ein nichttriviales $\gamma_i(k-n+1)$ mit $1 \leq i \leq l$ und $1 \leq n \leq N_i$, so daß

$$\begin{aligned} &\operatorname{ggt}(\gamma_i(k-n+1), r(k)) \neq 0 \quad \Longrightarrow \\ &\operatorname{ggt}(\gamma_i(k), r(k+n-1)) \neq 0 \quad \stackrel{(2.9)}{\Longrightarrow} \\ &\operatorname{ggt}(\gamma_i(k), r_j(k+n-1)) \neq 0 \end{aligned} \tag{2.19}$$

für alle $0 \leq j \leq l$, also insbesondere für

$$\operatorname{ggt}(\gamma_i(k), r_0(k+n-1)) \neq 0 \tag{2.20}$$

gilt.
Nach Konstruktion gilt außerdem

$$\gamma_i(k) \mid q_{i-1}(k) \quad \stackrel{(2.9)}{\Longrightarrow} \tag{2.21}$$

$$\gamma_i(k) \mid q_0(k). \tag{2.22}$$

Aus (2.20) und (2.22) folgt $\operatorname{ggt}(q_0(k), r_0(k+n-1)) \neq 1$ und somit $n-1 \in \operatorname{disp}_k(q_0(k), r_0(k))$. Also gibt es ein m mit $1 \leq m \leq i$ mit $N_m = n-1 \in N$.
Aus (2.19) folgt

$$\operatorname{ggt}(\gamma_i(k), r_{i-1}(k+N_m)) \neq 0$$

und somit wegen (2.21) auch

$$1 \neq \operatorname{ggt}(q_{i-1}(k), r_{i-1}(k+n-1)) = \operatorname{ggt}(q_{i-1}(k), r_{i-1}(k+N_m)).$$

Dies ist jedoch ein Widerspruch zu (2.11). □

Lemma 4 *Sei $b(k) \in \mathbb{K}(k)$, dann sind Polynome $p(k)$, $q(k)$ und $r(k)$, die (2.6) und (2.7) erfüllen, eindeutig bis auf konstante Vorfaktoren.*

Beweis:
Es seien $q(k), r(k), p(k)$ und $Q(k), R(k), P(k)$ Tripel von Polynomen, für die jeweils (2.7) gilt und die außerdem derselben Eingabe $b(k)$ zugrunde liegen:

$$b(k) = \frac{p(k)}{p(k-1)} \cdot \frac{q(k)}{r(k)} = \frac{P(k)}{P(k-1)} \cdot \frac{Q(k)}{R(k)}. \tag{2.23}$$

Wir setzen

$$g(k) := \operatorname{ggt}(p(k), P(k)),$$

$$d(k) := \frac{p(k)}{g(k)}$$

und

$$D(k) := \frac{P(k)}{g(k)}.$$

Dann folgt

$$\operatorname{ggt}(d(k), D(k)) = 1. \tag{2.24}$$

Wegen (2.13) folgt

$$\operatorname{ggt}(q(k), d(k-1)) = 1 \tag{2.25}$$

und wegen (2.14)

$$\operatorname{ggt}(r(k), d(k)) = 1. \tag{2.26}$$

Wir multiplizieren (2.23) mit den Nennern beider Seiten

$$p(k)\ q(k)\ P(k-1)\ R(k) = P(k)\ Q(k)\ p(k-1)\ r(k),$$

und dann teilen wir durch $g(k-1)\ g(k)$. Wir erhalten

$$d(k)\ q(k)\ D(k-1)\ R(k) = D(k)\ Q(k)\ d(k-1)\ r(k). \tag{2.27}$$

Aus (2.27) folgt

1. $d(k-1) \mid (d(k)\ q(k)\ D(k-1)\ R(k)) \overset{(2.25)}{\Longrightarrow}$
   $d(k-1) \mid (d(k)\ R(k)\ D(k-1)) \overset{(2.24)}{\Longrightarrow}$
   $d(k-1) \mid (d(k)\ R(k)) \Longrightarrow$

   $$d(k) \mid (R(k+1)\ d(k+1)) \tag{2.28}$$

   und

2. $d(k) \mid (D(k)\ Q(k)\ d(k-1)\ r(k)) \overset{(2.26)}{\Longrightarrow}$
   $d(k) \mid (D(k)\ Q(k)\ d(k-1)) \overset{(2.24)}{\Longrightarrow}$

   $$d(k) \mid (Q(k)\ d(k-1)). \tag{2.29}$$

Wenden wir (2.28) und (2.29) rekursiv auf sich selbst an, so erhalten wir

$$d(k) \mid R(k+n-1)\ R(k+n-2) \cdots R(k)\ d(k+n)$$

und

$$d(k) \mid Q(k-n+1)\ Q(k-n+2) \cdots Q(k)\ d(k-n)$$

für alle $n \in \mathbb{N}$.
Da $d(k)$ nur endlich viele Nullstellen hat, folgt $\operatorname{ggt}(d(k), d(k+n)) = \operatorname{ggt}(d(k), d(k-n)) = 1$ für $n > \max(\operatorname{disp}_k(d(k), d(k)))$, und somit

$$d(k) \mid R(k+1)\ R(k+2) \cdots R(k+n)$$

und

$$d(k) \,|Q(k-1)\; Q(k-2) \cdots Q(k-n).$$

Da nach (2.7) $R(k+1)\; R(k+2) \cdots R(k+n)$ und $Q(k-1)\; Q(k-2) \cdots Q(k-n)$ teilerfremd sind, muß $d(k)$ konstant sein.
Also unterscheiden sich $P(k)$ und $p(k)$ nur um konstante Vorfaktoren, und es gilt nach (2.23)

$$\frac{q(k)}{r(k)} = \frac{Q(k)}{R(k)},$$

was äquivalent zur Gleichung

$$q(k) R(k) = Q(k) r(k)$$

ist.
Da nach (2.7) $\operatorname{ggt}(q(k), r(k)) = \operatorname{ggt}(Q(k), R(k)) = 1$ gilt, folgt $q(k) \,|Q(k)$, $Q(k) \,|q(k)$, $R(k) \,|r(k)$ und $r(k) \,|R(k)$, womit die Eindeutigkeit bis auf konstante Vorfaktoren für alle drei Polynome gezeigt ist. □

Bemerkung 3 *Das Sortieren der Dispersionsmenge N (Algorithmus 2, Schritt 2b) ist notwendig für die Eindeutigkeit der Lösung*

Beweis durch Gegenbeispiel:

$$b(k) := \frac{(k+2)\;(k+1)}{k}$$

$$q_0(k) := (k+2)\;(k+1)$$

$$r_0(k) := k$$

$$N := \operatorname{disp}_k(q_0(k), r_0(k)) = \{1, 2\}$$

Fall 1: $N_1 := 1$**,** $N_2 := 2$**:**
$i = 1, N_1 = 1$

$$\gamma_1(k) := \operatorname{ggt}(q_0(k), r_0(k+1)) = k+1$$

$$q_1(k) := \frac{(k+2)\;(k+1)}{k+1} = k+2$$

$$r_1(k) := \frac{k}{k} = 1$$

$$p_1(k) := k+1$$

$i = 2, N_2 = 2$

$$\gamma_2(k) := \operatorname{ggt}(q_1(k), r_1(k+2)) = 1$$

$$q(k) := q_2(k) := \frac{k+2}{1} = k+2$$

$$r(k) := r_2(k) := \frac{1}{1} = 1$$

$$p(k) := p_2(k) := k+1$$

Fall 2: $N_1 := 2$, $N_2 := 1$**:**
$i = 1, N_1 = 2$

$$\gamma_1(k) := \mathrm{ggt}(q_0(k), r_0(k+2)) = k+2$$

$$q_1(k) := \frac{(k+2)\ (k+1)}{k+2} = k+1$$

$$r_1(k) := \frac{k}{k} = 1$$

$$p_1(k) := (k+1)(k+2)$$

$i = 2, N_2 = 1$

$$\gamma_2(k) := \mathrm{ggt}(q_1(k), r_1(k+2)) = 1$$

$$q(k) := q_2(k) := \frac{k+1}{1} = k+1$$

$$r(k) := r_2(k) := \frac{1}{1} = 1$$

$$p(k) := p_2(k) := (k+1)(k+2)$$

□

Im nächsten Abschnitt klären wir, wie die Dispersionsmenge $N := \mathrm{disp}_k(q_0(k), r_0(k))$ berechnet wird.

2.2.3 Bestimmung der Dispersionsmenge

Wir werden im folgenden zeigen, daß die Dispersionsmenge der Menge der Nullstellen aus $\mathbb{N}_0$ der Resultante entspricht, und wie man diese Nullstellen berechnet. Die Resultante definieren wir wie folgt.

Definition 6 *Resultante*

Man schreibe

$$f(k) = \sum_{i=0}^{m} a_i k^i, g(k) = \sum_{j=0}^{n} b_j k^j,$$

und sei $a_i := 0$ für $i \notin \{0, \ldots, m\}$ und $b_j := 0$ für $j \notin \{0, \ldots, n\}$,
dann ist

$$\mathrm{Res}_k(f(k), g(k)) := \begin{cases} 1 & \text{falls } n = 0, m = 0 \\ \det(C) & \text{sonst} \end{cases}$$

die **Resultante** von $f(k), g(k) \in K[k]$ bzgl k, wobei die Koeffizienten der Matrix $C \in \mathbb{K}^{(m+n,m+n)}$ durch

$$c_{i,j} := \begin{cases} a_{m-j+i} & \text{für } 1 \leq i \leq n \\ b_{i-j} & \text{für } n+1 \leq i \leq m+n \end{cases}$$

definiert sind.
C heißt **Sylvestermatrix** von $f(k)$ und $g(k)$ bezüglich k. □
Für $n = m+1$ sieht die Sylvestermatrix beispielsweise so aus:

$$\left.\begin{pmatrix} a_m & a_{m-1} & \dots & a_0 & & & & \\ & a_m & a_{m-1} & \dots & a_0 & & & \\ & & & \dots & \dots & \dots & & \\ & & & a_m & a_{m-1} & \dots & a_0 & 0 \\ & & & & a_m & a_{m-1} & \dots & a_0 \\ b_n & b_{n-1} & \dots & b_1 & b_0 & & & \\ & b_n & b_{n-1} & \dots & \dots & b_0 & & \\ & & & & \dots & \dots & \dots & \\ & & & b_n & b_{n-1} & \dots & \dots & b_0 \end{pmatrix}\right\} \begin{matrix} \\ \\ \\ \\ \\ n \text{ Zeilen} \\ \\ \\ \\ m \text{ Zeilen} \end{matrix}$$

□

Das folgende Resultat aus der Algebra[3] ermöglicht uns die Bestimmung von $N := \text{disp}_k(q_0(k), r_0(k))$ mit Hilfe der Resultante.

Lemma 5

$$\text{Res}_k(f(k), g(k)) = 0 \iff \text{ggt}(f(k), g(k)) \neq 1. \tag{2.30}$$

Zunächst beweise ich drei Zwischenergebnisse, die später gebraucht werden:

1. $$\text{Res}_k(f(k), 0) = 0 \tag{2.31}$$
   gilt für $\text{Grad}(f(k)) = m > 0$, da
   $$\text{Res}_k(f(k), 0) = \det(0_m) = 0.$$

2. $$\text{Res}_k(f(k), g(k)) = 0 \iff \text{Res}_k(g(k), f(k)) = 0. \tag{2.32}$$
   Die Sylvestermatrix von $f(k)$ und $g(k)$ kann nämlich durch Zeilenvertauschungen in die Sylvestermatrix von $g(k)$ und $f(k)$ überführt werden. Dabei verändert sich höchstens das Vorzeichen der Determinante. □

   Bemerkung 4 *Genaugenommen gilt*
   $$\text{Res}_k(f(k), g(k)) = (-1)^{n\,m} \text{Res}_k(g(k), f(k)).$$

3. Ist $\text{Grad}(f(k)) = m \leq n = \text{Grad}(g(k))$,[4] so sei $q(k)$ das Vielfache von $f(k)$ und

[3] Siehe auch [12], S. 104-106 für eine kürzere aber fehlerhafte Fassung des Beweises.
[4] Wir können das wegen (2.32) ohne Beschränkung der Annahme (2.30) fordern.

$h(k)$ das Restpolynom der Euklidischen Division

$$g(k) =: q(k)\ f(k) + h(k) \tag{2.33}$$

mit $q(k), h(k) \in \mathbb{K}[k]$ und $\mathrm{Grad}(h) =: d < \mathrm{Grad}(f(k)) = m$.
Dann gilt

$$\mathrm{Res}_k(f(k), g(k)) = 0 \quad \Longleftrightarrow \quad \mathrm{Res}_k(f(k), h(k)) = 0 \tag{2.34}$$

Beweis:
Es sei C die Sylvestermatrix von $f(k)$ und $g(k)$. Wir substrahieren in C Vielfache der ersten n Zeilen von den letzten m Zeilen derart, daß am Schluß in den letzten m Zeilen anstelle der Koeffizienten von $g(k)$ die Koeffizienten von h stehen. Das geht gemäß der Euklidischen Division: Sei $q(k) =: \sum_{j=0}^{p} q_j\, k^j$ ($q(k)$ gemäß (2.33)), dann definieren sich die Zeilenvektoren der Matrix C' als Vielfache der Zeilenvektoren der Matrix C, und zwar

$$C'_i := \begin{cases} C_i & \text{für } i \in \{1, \ldots n\} \\ C_i - \sum_{j=0}^{p} q_j\, C_{i-n+p-j} & \text{für } i \in \{n+1, \ldots n+m\}. \end{cases}$$

C' hat die folgende Form:

$$C' = \begin{pmatrix} M_1 & M_2 \\ 0 & M_3 \end{pmatrix},$$

wobei $M_1 \in K^{(n-d)\,X\,(n-d)}$ eine obere Dreiecksmatrix mit a_m als Diagonalelement für sämtliche Zeilen und $M_3 \in K^{(m+d)X(m+d)}$ die Sylvestermatrix von $f(k)$ und $h(k)$ ist. Da sich die Determinante einer Matrix durch Addition von Vielfachen von Zeilen nicht ändert, gilt

$$\mathrm{Res}_k(f(k), g(k)) = \det(C) = \det(C') = (a_m)^{n-d}\ \mathrm{Res}_k(f(k), h(k)).$$

Da $a_m \neq 0$, folgt (2.34).
Für

$$g(k) := (k+1)(k+2) + 3 = k^2 + 3k + 5$$

und

$$f(k) := k + 1$$

ist

$$q(k) = k + 2,\ h(k) = 3,$$

und

$$C = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 3 & 5 \end{pmatrix}$$

$C'_1 = C_1 = (1,1,0)$, $C'_2 = C_2 = (0,1,1)$ und $C'_3 = C_3 - C_1 - 2\,C_2 = (1,3,5) - (1,1,0) - (0,2,2) = (0,0,3)$, also

$$C' = \left(\begin{array}{cc|c} 1 & 1 & 0 \\ 0 & 1 & 1 \\ \hline 0 & 0 & 3 \end{array}\right).$$

Beim nun folgenden eigentlichen **Beweis** von **Lemma 5** gehen wir algorithmisch vor:

- Setze
$$f_0 : (k) = f(k), g_0(k) := g(k), h_0(k) := h(k), i := 0$$
und berechne
- solange $\text{Grad}(h_i(k)) > 0$
$$i := i + 1, f_i(k) := h_{i-1}(k), g_i(k) := f_{i-1}(k)$$
und definiere h_i durch euklidische Division
$$g_i(k) =: q_i(k)\ f_i(k) + h_i(k). \tag{2.35}$$

Da $\text{Grad}(h_i(k)) < \text{Grad}(g_i(k)) = \text{Grad}(h_{i-1}(k))$ gilt, wird die Schleife des obigen Algorithmus spätestens nach dem $(m-1)$. Iterationsschritt, sagen wir nach dem $p.$, mit konstantem $h_p(k)$ verlassen.
Haben $g(k)$ und $f(k)$ einen nichttrivialen gemeinsamen Teiler $t(k)$, so pflanzt sich dieser in der Schleife fort:

$$t(k)\,|g_i(k) \text{ und } t(k)\,|f_i(k) \iff t(k)\,|h_i(k) \text{ und } t(k)\,|f_i(k) \iff$$
$$t(k)\,|f_{i+1}(k) \text{ und } t(k)\,|g_{i+1}(k). \tag{2.36}$$

Also gilt
$$t(k)\,|g(k) \text{ und } t(k)\,|f(k) \iff t(k)\,|h_p(k).$$
Da $\text{Grad}(g(k)) \geq 1 > 0 = \text{Grad}(h_p(k))$, folgt
$$t(k)\,|g(k) \text{ und } t(k)\,|f(k) \iff h_p(k) = 0. \tag{2.37}$$

Nach (2.34) ist
$$\text{Res}_k(f(k), g(k)) = 0 \iff \text{Res}_k(f_1(k), g_1(k)) = 0 \iff$$
$$\iff \ldots \iff \text{Res}_k(f_{p-1}(k), g_{p-1}(k)) = 0.$$
Wegen (2.32), wegen $f_{p-1}(k) = h_p(k)$ und wegen $g_{p-1}(k) = f_p(k)$ gilt
$$\text{Res}_k(f_{p-1}(k), g_{p-1}(k)) = 0 \iff \text{Res}_k(f_p(k), h_p(k)) = 0,$$
also
$$\text{Res}_k(f(k), g(k)) = 0 \iff \text{Res}_k(f_p(k), h_p(k)) = 0. \tag{2.38}$$
Die Sylvestermatrix von $f_p(k)$ und $h_p(k)$ ist eine Diagonalmatrix mit jeweils $h_p(k)$ als Diagonalelement, daher ist
$$\text{Res}_k(f_p(k), h_p(k)) = 0 \iff h_p(k) = 0 \overset{(2.38)}{\iff}$$
$$\text{Res}_k(f(k), g(k)) = 0 \overset{(2.37)}{\iff} t(k)\,|g(k) \text{ und } t(k)\,|f(k)$$
q.e.d. □

Bemerkung 5 *Zur Berechnung von* $\mathrm{Res}_k\left(f\left(k\right),g\left(k\right)\right)$

Der Beweis von Lemma 5 induziert einen Algorithmus zur rekursiven Berechnung der Resultante. Resultanten stehen in allen Computeralgebrasystemen zur Verfügung (mit zum Teil schnelleren Algorithmen). Daher kommt es uns an dieser Stelle ausschließlich auf die Aussage (2.30) an. □
Aufgrund von Lemma 5 kann man $\mathrm{Res}_k\left(q\left(k\right),r\left(k+i\right)\right)$ berechnen und erhält ein Polynom in i. Dessen Nullstellen aus $\mathbb{N}_0$ bilden N.
Beispiel:
Seien $q\left(k\right)=\left(k+1\right)\cdot\left(k+b\right)=k^2+\left(1+b\right)k+b$ und
$r\left(k\right)=k\Longrightarrow r\left(k+i\right)=\left(k+i\right)$,
dann ist

$$\mathrm{Res}_k\left(q\left(k\right),r\left(k+i\right)\right)=\begin{vmatrix}1 & 1+b & b\\ 1 & i & 0\\ 0 & 1 & i\end{vmatrix}=i^2+b-\left(i+b\right)=\left(i-1\right)\cdot\left(i-b\right)\Longrightarrow N=\left\{1\right\}.$$

□

Bei dem vorangegangenen Beispiel war $N=\{1\}$ schon durch scharfes Hinsehen sichtbar. Bei den meisten Beispielen ist das auch der Fall. Daher braucht man beim Rechnen per Hand sicherlich selten die Resultante auszurechnen. Dies ist auch sicherlich ein Grund, warum Gosper in seiner Arbeit [6] nicht auf dieses Problem einging.
Eine Implementierung muß aber auch auf unübersichtliche Resultenten vorbereitet sein. Da Resultantenrechnung in Computeralgebrasystemen sehr teuer ist (auch wenn Algorithmen angewandt werden, die ohne Matrizenrechnung auskommen), ist das oben beschriebene Verfahren leider wenig praktikabel. Man und Wright [11] erläutern dies Problem und präsentieren einen Algorithmus, der ohne Resultante auskommt. Dieser wird in Abschnitt 2.6 vorstellt.

2.2.4 $f\left(k\right)$ ist ein Polynom

Wir zeigen nun zunächst, daß $f\left(k\right)$ rational in k, und dann, daß f sogar ein Polynom in k sein muß. Ersteres geht sehr schnell.

Lemma 6 *Seien (2.1),(2.3) und (2.5) gültig für alle* $k\in\mathbb{N}$*,*
dann ist

$$\frac{s\left(k\right)}{s\left(k-1\right)}\in\mathbb{K}\left(k\right)\Longrightarrow f\left(k\right)\in\mathbb{K}(k).$$

Beweis

$$f\left(k\right)\overset{(2.3)}{=}\frac{s\left(k\right)\cdot p\left(k\right)}{a\left(k\right)\cdot q\left(k+1\right)}=\frac{s\left(k\right)\cdot p\left(k\right)}{\left(s\left(k\right)-s\left(k-1\right)\right)\cdot q\left(k+1\right)}=\frac{p\left(k\right)}{\left(1-\frac{s\left(k-1\right)}{s\left(k\right)}\right)\cdot q\left(k+1\right)}$$

. □

Der Beweis dafür, daß $f(k)$ ein Polynom in k sein muß, ist umfangreicher:

Lemma 7 Unter den früheren Voraussetzungen sei $f(k) \in \mathbb{K}(k)$, so daß (2.5) erfüllt ist. Dann ist $f(k) \in \mathbb{K}[k]$.

Angenommen, $f(k)$ wäre kein Polynom in k,
dann gäbe es nach Lemma 6 trotzdem teilerfremde Polynome $c(k)$ und $d(k)$ mit $f(k) = \frac{c(k)}{d(k)}$.
Es müßte $Grad(d(k)) > 0$ gelten, da $f(k)$ sonst ein Polynom wäre.
Durch Multiplikation von (2.5) mit $d(k) \cdot d(k-1)$ erhält man

$$q(k+1) \cdot c(k) \cdot d(k-1) - r(k) \cdot c(k-1) \cdot d(k) = d(k) \cdot d(k-1) \cdot p(k). \tag{2.39}$$

Es sei

$$j := \max \operatorname{disp}_k (d(k), d(k))$$

und

$$g(k) := \operatorname{ggt}(d(k), d(k+j)). \tag{2.40}$$

j existiert, da $0 \in \operatorname{disp}_k (d(k), d(k))$.
Aus der Maximalität von j folgt

$$\operatorname{ggt}(d(k+j), d(k-1)) = 1. \tag{2.41}$$

Da $g(k) | d(k+j) \Longrightarrow$

$$\operatorname{ggt}(g(k), d(k-1)) = 1. \tag{2.42}$$

Man ersetze k durch $k-j$ in (2.41) und erhalte

$$\operatorname{ggt}(d(k-j-1), d(k)) = 1.$$

Da $g(k-j-1) | d(k-j-1) \Longrightarrow$

$$\operatorname{ggt}(g(k-j-1), d(k)) = 1. \tag{2.43}$$

Aus (2.40) folgt:

$$g(k) | d(k). \tag{2.44}$$

Da nach Voraussetzung $\operatorname{ggt}(d(k), c(k)) = 1$ gilt, folgt

$$\operatorname{ggt}(g(k), c(k)) = 1 \tag{2.45}$$

Wegen (2.44) teilt $g(k)$ die rechte Seite von (2.39) also
$g(k) | q(k+1) \cdot c(k) \cdot d(k-1) - r(k) \cdot c(k-1) \cdot d(k) \overset{(2.44)}{\Longrightarrow}$
$g(k) | (q(k+1) \cdot c(k) \cdot d(k-1)) \overset{(2.42)}{\Longrightarrow}$
$g(k) | (q(k+1) \cdot c(k)) \overset{(2.45)}{\Longrightarrow}$

$g(k)\,|q(k+1) \Longrightarrow$

$$g(k-1)\,|q(k)\,. \tag{2.46}$$

Völlig analog zeigt man (2.49).
Aus (2.40) folgt nämlich außerdem

$$g(k-j-1)\,|d(k-1)\,. \tag{2.47}$$

Da nach Voraussetzung $d(k-1)$ relativ prim zu $c(k-1)$ ist, folgt

$$g(k-j-1) \text{ ist relativ prim zu } c(k-1)\,. \tag{2.48}$$

Wegen (2.47) teilt $g(k-j-1)$ die rechte Seite von (2.39), also
$g(k-j-1)\,|d(k)\cdot d(k-1)\cdot p(k) \overset{(2.39)}{\Longrightarrow}$
$g(k-j-1)\,||q(k+1)\cdot c(k)\cdot d(k-1) - r(k)\cdot c(k-1)\cdot d(k) \overset{(2.47)}{\Longrightarrow}$
$g(k-j-1)\,|(r(k)\cdot c(k-1)\cdot d(k)) \overset{(2.43)}{\Longrightarrow}$
$g(k-j-1)\,|(r(k)\cdot c(k-1)) \overset{(2.48)}{\Longrightarrow}$
$g(k-j-1)\,|r(k) \Longrightarrow$

$$g(k-1)\,|r(k+j)\,. \tag{2.49}$$

Nach (2.46) und (2.49) haben $q(k)$ und $r(k+j)$ also den gemeinsamen Faktor $g(k-1)$.
Das ist aber ein Widerspruch zur geforderten Teilerfreiheit (2.7). □

2.2.5 Bestimmung einer oberen Schranke des Grades von $f(k)$

Wir nehmen im Folgenden an, daß der Grad des Nullpolynoms gleich -1 ist. Diese Konvention wird für (2.51) gebraucht.

Lemma 8 Unter den vorhergehenden Voraussetzungen sei $f(k)$ ein von Null verschiedenes Element von $\mathbb{K}[k]$ und eine Lösung von (2.5), also von

$$q(k+1)\cdot f(k) - r(k)\cdot f(k-1) = p(k)$$

1. *Ist*
$$\text{Grad}\,(q(k+1)+r(k)) \leq \text{Grad}\,(q(k+1)-r(k))\,,$$
dann folgt
$$\text{Grad}\,(f(k)) = \text{Grad}\,(p(k)) - \text{Grad}\,(q(k+1)-r(k))\,.$$

2. *Ist*
$$L := \text{Grad}\,(q(k+1)+r(k)) > \text{Grad}\,(q(k+1)-r(k))\,,$$
dann sei e_L der Koeffizient vor k^L in $(q(k+1)+r(k))$ und d_{L-1} der Koeffizient vor k^{L-1} in $(q(k+1)-r(k))$, (wobei $d_{L-1}=0$ sein kann).

(a) Ist $\frac{-2\,d_{L-1}}{e_L} < 0$ *oder* $\frac{-2\,d_{L-1}}{e_L} \notin \mathbb{N}$*, so ist*

$$\mathrm{Grad}\,(f\,(k)) = \mathrm{Grad}\,(p\,(k)) - L + 1.$$

(b) Ist $\frac{-2\,d_{L-1}}{e_L} \in \mathbb{N}$ *und* $\frac{-2\,d_{L-1}}{e_L} \geq 0$*, so ist*

$$\mathrm{Grad}\,(f\,(k)) \leq \max\left\{\frac{-2\,d_{L-1}}{e_L}, \mathrm{Grad}\,(p\,(k)) - L + 1\right\}.$$

Beweis:
(2.5) läßt sich umformen zu

$$p\,(k) = (q\,(k+1) - r\,(k)) \cdot \frac{f\,(k) + f\,(k-1)}{2} + (q\,(k+1) + r\,(k)) \cdot \frac{f\,(k) - f\,(k-1)}{2} \quad (2.50)$$

Sei $d := \mathrm{Grad}\,(f\,(k))$

$$f\,(k) =: \lambda_d k^d + \lambda_{d-1} k^{d-1} + \ldots + \lambda_0$$

mit $\lambda_i \in F$ für alle $i \in \{0, \ldots, d\}$,
dann folgt

$$f\,(k) + f\,(k-1) =$$

$$\left(\lambda_d k^d + \lambda_{d-1} k^{d-1} + \ldots\right) + \left(\lambda_d\,(k-1)^d + \lambda_{d-1}\,(k-1)^{d-1} + \ldots\right) =$$

$$2\lambda_d k^d + \ldots$$

$$f\,(k) - f\,(k-1) =$$

$$\lambda_d \cdot d\,k^{d-1} + \ldots \qquad .$$

Das heißt
$\mathrm{Grad}\,(f\,(k) + f\,(k-1)) = d$
und
$\mathrm{Grad}\,(f\,(k) - f\,(k-1)) = d - 1$ für $d > 0$.
Eingesetzt in (2.50) ergibt[5] sich so

$$\mathrm{Grad}\,(p\,(k)) = \max\left\{\mathrm{Grad}\,(q\,(k+1) - r\,(k)) + d, \mathrm{Grad}\,(\,q\,(k+1) + r\,(k)\,) + d - 1\right\}. \quad (2.51)$$

Im**Fall 1** ergibt sich durch Einsetzen in (2.51)

$$\mathrm{Grad}\,(p\,(k)) = \mathrm{Grad}\,(q\,(k+1) - r\,(k)) + \mathrm{Grad}\,(f\,(k)) \quad \Longrightarrow$$

$$\mathrm{Grad}\,(\,f\,(k)) = \mathrm{Grad}\,(p\,(k)) - \mathrm{Grad}\,(q\,(k+1) - r\,(k)\,).$$

Im Fall 1) kann man (2.50) ausschreiben zu

$$p\,(k) = \left(d_{L-1} + \frac{1}{2} d\,e_L\right)\,\lambda_d\,k^{L+d-1} + \ldots \quad (2.52)$$

[5]Hier wird die Festlegung $\mathrm{Grad}\,(0) = -1$ gebraucht. Ist nämlich etwa $r\,(k) = q\,(k+1) = 1$, so ist nach (2.50) $\mathrm{Grad}\,(p\,(k)) = d - 1$

Fall 2a):
Ist $-2\,d_{L-1}$ keine ganze Zahl ≥ 0, so ist:

$$d_{L-1} + \frac{1}{2} d\, e_L \neq 0 \quad \Longrightarrow$$

$$\mathrm{Grad}\,(p\,(k)) = L + d - 1$$

$$d = \mathrm{Grad}\,(f\,(k)) = \mathrm{Grad}\,(p\,(k)) + d - 1 \quad \Longrightarrow$$

Sonst, im Fall 2b), ist entweder $d = \frac{-2\,d_{L-1}}{e_L}$ oder $d_{L-1} + \frac{1}{2} d\, e_L \neq 0$, und somit

$$\mathrm{Grad}\,(p\,(k)) = \max\left\{\mathrm{Grad}\,(q\,(k+1) - r\,(k)) + d, \mathrm{Grad}\,(q\,(k+1) + r\,(k)) + d - 1\right\}.$$

□

Daß alle Fälle in Lemma 8 wirklich auftreten, zeigen folgende Beispiele:

1. Bei $a\,(k) = k\,x^k$ gilt $q\,(k) = x$ und $r\,(k) = 1$ und somit

$$\mathrm{Grad}\,(q\,(k+1) + r\,(k)) = \mathrm{Grad}\,(x+1) = 0 \leq$$

$$\mathrm{Grad}\,(q\,(k+1) - r\,(k)) = \mathrm{Grad}\,(x-1) = 0.$$

2. (a) Bei dem trivialsten Beispiel für die Anwendung des Gosperalgorithmus, $a\,(k) = k$, kommt die Festlegung, daß der Grad des Nullpolynoms gleich -1 ist. Es ergibt sich nämlich $q\,(k) = 1$ und $r\,(k) = 1$ und somit

$$L := \mathrm{Grad}\,(q\,(k+1) + r\,(k)) = \mathrm{Grad}\,(2) = 0 >$$

$$\mathrm{Grad}\,(q\,(k+1) - r\,(k)) = \mathrm{Grad}\,(0) = -1.$$

Es gibt aber auch schwerere Beispiele, wo Fall 2a eintritt.
Für $a\,(k) = \frac{6\,k+3}{4\,k^4+8\,k^3+8\,k^2+4\,k+3}$ etwa gilt $q\,(k) = 2\,k^2 - 4\,k + 3$ und $r\,(k) = 2\,k^2 + 4\,k + 3$ und somit

$$L := \mathrm{Grad}\,(q\,(k+1) + r\,(k)) = \mathrm{Grad}\left(4\,k^2 + 6\right) = 2 >$$

$$\mathrm{Grad}\,(q\,(k+1) - r\,(k)) = \mathrm{Grad}\,(-12\,k + 6) = 1.$$

(b) Schließlich gilt bei $a\,(k) = (k-n)_n$ gilt $q\,(k) = -k + 1$ und $r\,(k) = -k + 1 + n$ und somit

$$\mathrm{Grad}\,(q\,(k+1) + r\,(k)) = \mathrm{Grad}\,(1+n) = 0 <$$

$$\mathrm{Grad}\,(q\,(k+1) - r\,(k)) = \mathrm{Grad}\,(-2\,k - n) = 1.$$

2.2.6 s ist eindeutig bis auf eine Konstante

Die Gleichung (2.5),

$$q(k+1) \cdot f(k) - r(k) \cdot f(k-1) = p(k)$$

kann man als Rekursionsformel für ein $f(k) \in \mathbb{K}[k]$ auffassen. Der Lösungsraum dieser Rekursionsformel hat dann höchstens Dimension 1, wie im folgenden Lemma 9 bewiesen wird. Daraus ergibt sich durch Einsetzen in (2.3) ebenfalls ein Lösungsraum der Dimension 1 für die Berechnung von $s(k)$, was dann genau die Verschiebung um eine Konstante bedeutet.

Lemma 9 *$s(k)$ ist eindeutig bis auf einen konstanten Faktor*

Beweis: Wir zeigen, daß $f(k)$ eindeutig bis auf einen konstanten Faktor ist.
Seien $\tilde{f}(k), f_0(k)$ zwei verschiedene Lösungen von (2.5), dann gilt

$$q(k+1) \cdot \tilde{f}(k) - r(k) \cdot \tilde{f}(k-1) = q(k+1) \cdot f_0(k) - r(k) \cdot f_0(k-1) \quad \Longrightarrow$$

$$q(k+1) \cdot \tilde{f}(k) - q(k+1) \cdot f_0(k) - \Big(r(k) \cdot \tilde{f}(k-1) - r(k) \cdot f_0(k-1)\Big) = 0 \quad \Longrightarrow$$

$$q(k+1) \cdot f_1(k) - r(k) \cdot f_1(k-1) = 0$$

mit $f_1(k) := \tilde{f}(k) - f_0(k)$.
Somit folgt

$$b(k) := \frac{r(k)}{q(k+1)} = \frac{f_1(k)}{f_1(k-1)}.$$

Da $b(k)$, $\tilde{p}(k) := f_1(k)$, $\tilde{q}(k) := 1$, und $\tilde{r}(k) := 1$ die Bedingungen (2.6) und (2.7) erfüllen, folgt nach Lemma 2, daß $f_1(k)$ eindeutig bis auf Konstanten ist.
Also gibt es für alle Lösungen $f(k)$ von (2.5) eine Konstante c mit

$$f(k) - f_0(k) = c\, f_1(k)$$

oder

$$f(k) = f_0(k) + c\, f_1(k).$$

Der Lösungsraum des Gleichungssystems der Koeffizienten von $f(k)$ hat höchstens Dimension 1, da c die einzige freie Variable ist.
Also hat nach (2.3) der Lösungsraum von $s(k)$ ebenfalls die Dimension 1. □

2.3 Zusammenfassung in Pseudocode

Nun sind wir soweit, den Gosperalgorithmus in allen seinen Schritten aufzeigen zu können:

Algorithmus 3 *Gosper*

Eingabe: $a(k)$
Ausgabe: $s(k)$ mit $s(k) - s(k-1) = a(k)$

1. Berechne $a(k)/a(k-1)$, und verifiziere, daß $a(k)$ hypergeometrisch ist.

2. Bestimme $p(k), q(k), r(k)$ mit dem Algorithmus aus Lemma 2.

3. Finde mit Lemma 8 eine obere Schranke d für den Grad des Polynoms $f(k)$. Ist $d < 0$, dann gibt es keine Lösung von $s(k)$ mit den gewünschten Eigenschaften.

4. Setze
$$f(k) := \sum_{i=0}^{d} f_i k^i, \tag{2.53}$$
wobei die f_i bisher unbekannte Elemente von $\mathbb{K}$ sind. Finde eine Lösung für das lineare Gleichungssystem in f_i, das man erhält, indem man alle Koeffizienten des Polynoms
$$q(k+1)\ f(k) - r(k)\ f(k-1) - p(k) \tag{2.54}$$
(Umformung von (2.5))
gleich Null setzt und dann nach den f_i auflöst.
Gibt es keine Lösung, so gibt es keine Lösung für $s(k)$ in (2.1) mit den gewünschten Eigenschaften. Sonst hat der Lösungsraum nach Lemma 9 Dimension 0 oder 1.

5. Finde die gewünschte(n) Lösung(en) von $s(k)$ in (2.1) durch Einsetzen in (2.3), dann ist (2.1)
$$s(k) - s(k-1) = a(k)$$
und gebe $s(k)$ aus.

2.4 Der Gosperalgorithmus ist ein Entscheidungsalgorithmus

Der folgende Satz ergibt sich aus den in Abschnitt 2.2 bewiesenen Lemmata.

Satz 1 *Der Gosperalgorithmus ist ein Entscheidungsalgorithmus*

Beweis:
Ist $a(k)$ ein hypergeometrischer Term und findet man (etwa mit Hilfe von Algorithmus 1) eine rationale Darstellung
$$\frac{a(k)}{a(k-1)} = \frac{q_0(k)}{r_0(k)}$$

mit $q_0(k), r_0(k) \in \mathbb{K}[k]$, so findet man nach Lemma 2 Polynome $p(k), q(k)$ und $r(k)$, die die Bedingungen (2.6) und (2.7) erfüllen.
Existiert eine Lösung $s(k)$ gemäß (2.1), so ist nach Lemma 7 die Funktion

$$f(k) := \frac{s(k) \cdot p(k)}{a(k) \cdot q(k+1)}$$

ein Polynom, für dessen Grad sich mit Hilfe von Lemma 8 eine obere Schranke berechnen läßt.
Da (2.54)

$$q(k+1)\ f(k) - r(k)\ f(k-1) - p(k)$$

ein lineares Gleichungssystem in den Koeffizienten von $f(k)$ ist, finden wir $f(k)$, sofern es existiert, und somit auch $s(k)$. □

2.5 Beispiele

1. Zunächst ein einfaches Beispiel:

$$a(k) := \frac{(b)_k}{k!}$$

$$\frac{a(k)}{a(k-1)} = \frac{b+k-1}{k}$$

Wir sparen uns an dieser Stelle die Anwendung des Algorithmus 2 aus Lemma 2, da offensichtlich $p(k) = 1$, $q(k) = b - k + 1$ und $r(k) = k$ gilt.

$$\text{Grad}\,(q(k+1) + r(k)) = \text{Grad}\,(2k + b - 1) = 1 >$$

$$0 = \text{Grad}\,(b-1) = \text{Grad}\,(q(k+1) - r(k))$$

Es ergibt sich $e_L = e_1 = 2$ und $d_{L-1} = d_0 = b - 1$, und da $\frac{-2\,d_0}{e_L} = -b + 1 \notin \mathbb{N}$, muß Fall 2a von Lemma 8 gelten:

$$\text{Grad}\,(f(k)) = 0 - 1 + 1 = 0.$$

Durch Einsetzen in (2.5) ergibt sich

$$(b+k)\,f_0 - k\,f_0 - 1 = 0$$

und somit

$$f(k) = f_0 = \frac{1}{b.}$$

Nach (2.3) ist

$$s(k) = \frac{a(k) \cdot q(k+1)}{p(k)} \cdot f(k) = \frac{(b)_k}{k!} \cdot \frac{b+k}{1}\,\frac{1}{b} = \frac{(b+1)_k}{k!} \quad .$$

□

2. Ein schwierigeres Beispiel ist

$$a(k) := \frac{\binom{n}{k}\binom{n+1}{k}}{\binom{2n}{2k}} \quad .$$

Wir vereinfachen den Quotienten

$$\frac{a(k)}{a(k-1)} = \frac{\binom{n}{k}}{\binom{n}{k-1}} \cdot \frac{\binom{n+1}{k}}{\binom{n+1}{k-1}} \cdot \frac{\binom{2n}{2n-2k}}{\binom{2n}{2k}} = \frac{(2k-1)(n+2-k)}{(2n-2k+1)\,k} \quad .$$

Wir initialisieren

$$p_0(k) := 1, q_0(k) := (2k-1)(n+2-k) \text{ und } r_0(k) := (2n-2k+1)\,k$$

und berechnen die Resultante von $q(k)$ und $r(k+j)$ mit Hilfe der MAPLE -Funktion `resultant`:

```
> resultant((2*k-1) * (n + 2 - k),subs(k=k+j,2*n+2*k+1)*k,k);
```

$$(4+4n+4j)(-4n-5-2j)(-n-2).$$

Da die Nullstellen $-1-n$ und $-2-n-\frac{5}{2}$ nicht unabhängig von n sind und auch nicht in $\mathbb{N}_0$ liegen, brauchen wir den Algorithmus 2 nicht anzuwenden, sondern können die Polynome verwenden, wie sie initialisiert sind.
Nun kommen wir zur Berechnung des maximalen Grades von $f(k)$:

$$q(k+1) + r(k) = -4k^2 + (4n+2)k + (n+1)$$

und

$$q(k+1) - r(k) = n+1.$$

Also ist $\text{Grad}\,(q(k+1)+r(k)) < \text{Grad}\,(q(k+1)-r(k))$, und da $d_{L-1} = d_1 = 0$ ist, folgt nach Fall 2b, daß $\text{Grad}\, f(k) \leq \max\{0, 0-2+1\} = 0$ ist.
Wir lösen nun (2.54) mit konstantem $f(k) = f_0$:

$$0 = f_0 \cdot (q(k+1) - r(k)) - p(k) = f_0 \cdot (n+1) - 1 \quad \Longrightarrow$$

$$f_0 = \frac{1}{n+1.}$$

Eingesetzt in (2.3) ergibt sich

$$s(k) = \frac{a(k) \cdot q(k+1)}{p(k)} \cdot f(k) = \frac{\binom{n}{k}\binom{n+1}{k}(2k+1)(n+1-k)}{\binom{2n}{2k}(2n+2k+1)(n+1)}$$

□

3. Nun kommen wir zum Beispiel (5) aus der Einleitung:

$$a\,(k) := (-1)^k \binom{n}{k} \binom{4\,k}{n}.$$

Der Algorithmus 1 wird in unserem Paket durch die Prozedur `simplify_combinatorial` ausgeführt:

```
> ak:= binomial(n,k) * (-1)^k*binomial(4*k,n);
```

$$ak := \mathrm{binomial}(n, k)\,(-1)^k\,\mathrm{binomial}(4\,k, n)$$

```
> quotient:=simplify_combinatorial(ak/subs(k=k-1,ak));
```

$$quotient := -8\,\frac{(n-k+1)\,(4\,k-1)\,(2\,k-1)\,(4\,k-3)}{(-4\,k+n)\,(-4\,k+n+1)\,(-4\,k+n+2)\,(-4\,k+3+n)}$$

Wir initialisieren $p\,(k)$, $q\,(k)$ und $r\,(k)$ gemäß Schritt 1 von Algorithmus 2:

```
> q0:= numer(quotient);r0:=denom(quotient);p0:=1;
```

$$q0 := -8\,(n-k+1)\,(4\,k-1)\,(2\,k-1)\,(4\,k-3)$$

$$r0 := (-4\,k+n)\,(-4\,k+n+1)\,(-4\,k+n+2)\,(-4\,k+3+n)$$

$$p0 := 1$$

und da die Resultante

```
> resultant(q0,subs(k=k+j,r0),k);
```

$$\begin{aligned}
&4096(\,3\,n+4+4\,j\,)\,(\,3\,n+3+4\,j\,)\,(\,3\,n+2+4\,j\,)\,(\,3\,n+1+4\,j\,)\\
&\quad(\,-4-16\,j+4\,n\,)^2\,(\,-16\,j+4\,n\,)^2\,(\,-16\,j+4\,n+4\,)\,(\,-16\,j+4\,n+8\,)\\
&\quad(\,-4-8\,j+2\,n\,)\,(\,-2-8\,j+2\,n\,)\,(\,-8\,j+2\,n\,)\,(\,-8\,j+2\,n+2\,)\\
&\quad(\,-12-16\,j+4\,n\,)\,(\,-8-16\,j+4\,n\,)
\end{aligned}$$

keine natürlichen Nullstellen in j hat, können wir wieder direkt mit der Initialisierung den Höchstgrad von $f\,(k)$ berechnen.

```
> pol1:=collect(subs(k=k+1,q0) + r0,k);
```

$$\begin{aligned}
&512\,k^4-512\,n\,k^3+(16\,n\,(n+1)-4\,(-8\,n-4)\,(n+2)\\
&\quad+(48\,n+48)\,(3+n)-384\,n+176)k^2+(-4\,n\,(n+1)\,(n+2)\\
&\quad+(-4\,n\,(n+1)+(-8\,n-4)\,(n+2))\,(3+n)-176\,n+24)k\\
&\quad-24\,n+n\,(n+1)\,(n+2)\,(3+n)
\end{aligned}$$

```
> degree(pol1,k);
```

4

```
> pol2:=collect(subs(k=k+1,q0)- r0,k);
```

$$\begin{aligned}
&768\,k^3+(-16\,n\,(n+1)+4\,(-8\,n-4)\,(n+2)-(48\,n+48)\,(3+n)\\
&\quad-384\,n+176)k^2+(4\,n\,(n+1)\,(n+2)\\
&\quad-(-4\,n\,(n+1)+(-8\,n-4)\,(n+2))\,(3+n)-176\,n+24)k\\
&\quad-24\,n-n\,(n+1)\,(n+2)\,(3+n)
\end{aligned}$$

```
> degree(pol2,k);
```

$$3 \quad .$$

Somit gilt Fall 2 von Lemma 8, und wegen

$$\frac{-2\,d_{L-1}}{e_L} = -2\,\frac{512}{768} = -3 \notin \mathbb{N}$$

folgt nach Fall 2a

$$\mathrm{Grad}\,(f\,(k)) = \mathrm{Grad}\,(p\,(k)) - 4 + 1 = -3.$$

Also kann es keinen hypergeometrischen Term $s\,(k)$ geben.

2.6 Implementierung des Gosperalgorithmus

2.6.1 Behandlung spezieller Eingaben

- Ist $a\,(k) = \sum_{i=0}^{p} a_i k^i$ ein Polynom, so liefern die Koeffizienten von (2.1)

$$\sum_{i=0}^{p} a_i k^i - \sum_{i=0}^{p+1} s_i k^i + \sum_{i=0}^{p+1} s_i\,(k-1)^i = 0 \tag{2.55}$$

ein lineares Gleichungssystem mit $p+1$ Koeffizientengleichungen in $p+1$ Unbekannten. Der Koeffizient bezüglich k^{p+1} hebt sich nämlich weg, die Unbekannte s_0 ist beliebig. Beispiel:

$$a\,(k) = k + 2$$

(2.55) ergibt

$$0 = k + 2 - \left(s_2\,k^2 + s_1\,k + s_0\right) + \left(s_2\,(k-1)^2 + s_1\,k - 1 + s_0\right) =$$

$$k\,(1 - 2s_2) + s_2 + 2 - s_1$$

$$\Longrightarrow \quad s_2 = \frac{1}{2} \quad \Longrightarrow \quad s_1 = \frac{5}{2} \quad ,$$

also

$$s\,(k) = \frac{1}{2}k^2 + \frac{5}{2}k + s_0.$$

In Computeralgebrasystemen kommen Algorithmen zur Anwendung, die (2.55) lösen.

- Ist der Zähler von $a\,(k) = 1$ und $a\,(k)$ ein Produkt aus polynomiellen und hypergeometrischen Termen, so initialisiere man die Polynome entsprechend.

2.6.2 Algorithmus zur Berechnung der Dispersionsmenge

Die in Abschnitt 2.2.3 vorgestellte Berechnung der Dispersionsmenge $N = \mathrm{disp}_k\left(q_0\left(k\right), r_0\left(k\right)\right)$ ist besonders dann teuer, wenn die Polynome $q_0\left(k\right)$ und $r_0\left(k\right)$ einen hohen Grad haben. Der Hauptgrund dafür ist, daß der Grad j in der $\mathrm{Res}_k\left(q_0\left(k\right), r_0\left(k+j\right)\right)$ genau das Quadrat des Grades J von $r_0\left(k+j\right)$ (bezüglich k oder j) ist. Um die Dispersionsmenge N zu berechnen, muß man also ein Polynom vom Grad J^2 faktorisieren.
Der von Man und Wright [11] vorgestellte Algorithmus macht die Resultantenberechnung überflüssig und gibt eine recht effiziente Methode zur Berechnung der Dispersionsmenge. Allerdings kommen die Autoren auch nicht ohne die Einführung einer zusätzlichen Variablen j aus.
Am Ende dieses Kapitels wird in Form einer MAPLE -Session ein Beispiel für die Überlegenheit des jetzt folgenden Algorithmus gegeben.

Algorithmus 4 *Algorithmus zur Berechnung der Dispersionsmenge*

Eingabe: zwei Polynome $q\left(k\right)$ und $r\left(k\right)$
Ausgabe: $\mathrm{disp}_k\left(q\left(k\right), r\left(k\right)\right)$

1. Berechne jeweils die Menge der monischen Primfaktoren S von $q\left(k\right)$ und T von $r\left(k\right)$.
2. Setze $J := \{\}$
3. Für jedes Paar $\left(s\left(k\right), t\left(k\right)\right) \in \left(S, T\right)$ berechne $D := \mathrm{disp}_k\left(s\left(k\right), t\left(k\right)\right)$:
   (a) Ist $\mathrm{Grad}\left(s\left(k\right)\right) =: n \neq \mathrm{Grad}\left(t\left(k\right)\right)$, so setze $D := \{\}$ und springe zu (3d).
   (b) Setze: $b :=$ Koeffizient von $s\left(k\right)$ vor k^{n-1},
   $d :=$ Koeffizient von $t\left(k\right)$ vor k^{n-1}
   und
   $$j := \frac{b-d}{n} \quad . \tag{2.56}$$
   Falls $j \notin \mathbb{N}_0$, so setze $D := \{\}$, und springe zu (3d).
   (c) Ist $s\left(k\right) - t\left(k+j\right) = 0$, dann setze $D := \{j\}$, sonst setze $D := \{\}$.
   (d) $J := J \cup D$.
4. Gebe J aus.

Um zu beweisen, daß der Algorithmus $\mathrm{disp}_k\left(q\left(k\right), r\left(k\right)\right)$ liefert, genügt es zu beweisen, daß der Schleifendurchlauf in Schritt 3 die Dispersionsmenge $\mathrm{disp}_k\left(s\left(k\right), t\left(k\right)\right)$ liefert. Gilt nämlich $\mathrm{ggt}\left(q\left(k\right), r\left(k+j\right)\right) \neq 1$, so muß es auch monische Primfaktoren $s\left(k\right)$ und $t\left(k\right)$ von $q\left(k\right)$ bzw. von $r\left(k\right)$ geben mit $\mathrm{ggt}\left(s\left(k\right), t\left(k+j\right)\right) \neq 1$.
Existiert also ein $j \in \mathrm{disp}_k\left(s\left(k\right), t\left(k\right)\right) \subset \mathbb{N}_0$, also $\mathrm{ggt}\left(s\left(k\right), t\left(k+j\right)\right) =: g\left(k\right) \neq 1$, dann folgt, da $s\left(k\right)$ und $t\left(k+j\right)$ nicht faktorisierbar sind, $\mathrm{Grad}\left(k\right) = n$. Also sind $s\left(k\right)$ und $t\left(k+j\right)$ Vielfache voneinander, und da sie monisch sind, gilt

$$s\left(k\right) = t\left(k+j\right). \tag{2.57}$$

Die beiden Polynome haben die Darstellung

$$s\left(k\right)=k^{n}+b\,k^{n-1}+\ldots$$

und

$$t\left(k\right)=k^{n}+d\,k^{n-1}+\ldots$$

Wenn man mithilfe des Binomialsatzes expandiert, erhält man

$$k^{n}+b\,k^{n-1}+\ldots=k^{n}+\left(j\,n+d\right)\,k^{n-1}+\ldots\quad\Longrightarrow$$

$$b=j\,n+d\quad\Longleftrightarrow$$

$$j=\frac{b-d}{n}\quad. \tag{2.58}$$

Ist $j=\frac{b-d}{n}\in\mathbb{N}_{0}$, kann man (2.57) überprüfen. □

Beispiel:

$$q\left(k\right):=4k^{10}\,\left(k^{2}+k+3\right)$$

$$r\left(k\right):=\left(k-5\right)\left(k^{2}+k+1\right)$$

1.

$$S=\left\{k,k^{2}+k+3\right\},T=\left\{k-5,k^{2}+k+1\right\}$$

2.

$$J:=\{\}$$

3. $s\left(k\right):=k,t\left(k\right):=k-5$

   (a)

   $$\mathrm{Grad}\left(s\left(k\right)\right)=1=\mathrm{Grad}\left(t\left(k\right)\right)$$

   (b)

   $$b:=0,d:=-5\quad\Longrightarrow\quad j=\frac{0-\left(-5\right)}{1}=5$$

   (c)

   $$s\left(k\right)-t\left(k+5\right)=k-k=0\quad\Longrightarrow\quad D:=\{5\}$$

   (d)

   $$J:=\{5\}$$

$s(k) := k, t(k) := k^2 + k + 1$

(a)

$$\text{Grad}(s(k)) = 1 \neq 2 = \text{Grad}(t(k))$$

$s(k) := k^2 + k + 3, t(k) := k - 5$

(a)

$$\text{Grad}(s(k)) = 2 \neq 1 = \text{Grad}(t(k))$$

$s(k) := k^2 + k + 3, t(k) := k^2 + k + 1$

(a)

$$\text{Grad}(s(k)) = 2 = \text{Grad}(t(k))$$

(b)

$$b := 1, d := 1 \quad \Longrightarrow \quad j = \frac{1-1}{2} = 0$$

(c) $s(k) - t(k+5) = 2 \neq 0 \quad \Longrightarrow \quad D := \{\}$

Ausgabe:

$$\text{disp}_k(q(k), r(k)) = \{5\}$$

□

Nach dem diesem didaktischen Beispiel kommen wir nun zur versprochenen MAPLE - Sesssion.
Der Algorithmus 4 wurde von Wolfram Koepf und mir in der Prozedur `dispersionset` implementiert. Wir zeigen das Polynom `q` zuerst in faktoriesierter Form, damit die Dispersionsmenge auf den ersten Blick sichtbar wird.

```
> q:= k^2 * (k+ a^2) *(k- 1000) *(k^3 +b);
```

$$q := k^2 (k + a^2)(k - 1000)(k^3 + b)$$

```
> r:=(subs(k=k-22,q));
```

$$r := (k - 22)^2 (k - 22 + a^2)(k - 1022)\left((k - 22)^3 + b\right)$$

In expandierter Form sieht `r` schon etwas unübersichtlicher aus.

```
> r:=expand(r);
```

$$\begin{aligned} r := {} & 31715451328\,k + 10882256\,b + 221158960\,k^3 - 1066\,k^2\,a^2\,b \\ & + 45452\,k\,a^2\,b + k^3\,a^2\,b - 494648\,a^2\,b - 1494592\,k\,b \\ & - 115874261888 + k^7 + k^4\,b - 1088\,k^3\,b + 68904\,k^2\,b + k^6\,a^2 \\ & - 1132\,k^5\,a^2 + 117260\,k^4\,a^2 - 1154\,k^6 + 142164\,k^5 - 7632680\,k^4 \\ & - 5052960\,k^3\,a^2 - 3622066272\,k^2 + 109993840\,k^2\,a^2 \\ & - 1202201792\,k\,a^2 + 5267011904\,a^2 \end{aligned}$$

Wir berechnen die Dispersionsmenge mit unserer Prozedur und nehmen die Zeit.

```
> bisher:=time():dispersionset(q,r,k); time()-bisher;
```

$$\{\, 22, 1022 \,\}$$

.267

Nun berechnen wir die Resultante

```
> bisher:=time(): res:= resultant(q, subs(k=k+j,r),k):  time()-bisher;
```

1.000

Das dauert viermal solange wie die gesamte Rechnung mit Man und Wrights Prozedur. Doch wir müssen noch die Nullstellen in $\mathbb{N}_0$ bezüglich j finden, und das dauert über 200 mal solange, wie die Ausführung von `dispersionset`.

```
> bisher:=time(): isolve(res,j): time()-bisher;
```

58.433

Das ist auch nicht erstaunlich, da ein Polynom vom Grad 49 faktorisiert werden muß.

```
> degree(res,j);
```

49

□

Kapitel 3

Der Zeilbergeralgorithmus

Gosper entwickelte nicht nur das im letzten Kapitel beschriebene Verfahren zur Berechnung von diskreten Stammfunktionen, sondern entwickelte auch Heuristiken zur Berechnung von definiten Summen (3.1)

$$A(n) := \sum_k F(n,k). \tag{3.1}$$

Zeilberger ([18],[19]) zeigte vor kurzem, wie Gosper unter Benutzung seines eigenen Algorithmus hätte vorgehen können.

3.1 Kurzer Überblick über den Algorithmus

Der Zeilbergeralgorithmus sucht iterativ für $l = 1, 2, \ldots$ eine Rekursion

$$\sum_{j=0}^{l} \sigma_j(n)\ A(n-j) = 0, \tag{3.2}$$

wobei $\sigma_j(n) \in \mathbb{K}(n)$ für alle $j \in \{0, \ldots, l\}$ gilt.
Er setzt dafür

$$a(k) := \sum_{j=0}^{l} \sigma_j(n)\ F(n-j,k), \tag{3.3}$$

wobei die $\sigma_j(n)$ bisher nur als unbekannte Konstanten in k betrachtet werden.
Es gilt dann

$$\sum_k a(k) = \sum_k \sum_{j=0}^{l} \sigma_j(n)\ F(n-j,k) =$$

$$\sum_{j=0}^{l} \sigma_j(n) \sum_k F(n-j,k) \overset{(3.1)}{=} \sum_{j=0}^{l} \sigma_j(n)\ A(n-j). \tag{3.4}$$

Zeilberger berechnet dann mit einer Version des Gosperalgorithmus eine Funktion $G(n,k) := s(k)$ mit (2.1)

$$a(k) = s(k) - s(k-1) = G(n,k) - G(n,k-1).$$

Die Polynome $\sigma_j(n)$ werden dabei beim Berechnen der Koeffizienten von $f(k)$ durch Lösen der Gleichung (2.5)

$$q(k+1)\ f(k) - r(k)\ f(k-1) - p(k) = 0$$

gleich mitbestimmt. Dieses ist möglich, da (2.5) nicht nur in den Koeffizienten f_i von $f(k)$ linear ist, sondern auch in den $\sigma_j(n)$, das heißt, es gibt in der Gleichung keine Produkte von Unbekannten (siehe Abschnitt 3.2.2).
Hat $G(n,k)$ einen endlichen Träger, d.h. gibt es nur endlich viele $k \in \mathbb{N}_0$ mit $G(n,k) \neq 0$, so folgt

$$\sum_k a(k) = \sum_k G(n,k) - G(n,k-1) = \left(\lim_{k\to\infty} G(n,k)\right) - \left(\lim_{k\to-\infty} G(n,k)\right) = 0$$

durch Teleskopieren.
Wegen (3.4) gilt dann (3.2).
Im nächsten Abschnitt klären wir, welche Voraussetzungen für den Zeilbergeralgorithmus erforderlich sind, erläutern einige Teilaspekte und beschreiben, wie und wann man mit Hilfe der Rekursion (3.2) eine geschlossene Form für $A(n)$ gemäß (3.1) findet.
Im einzelnen werden folgende Fragen behandelt:

1. Wie kann man sicherstellen, daß $a(k)$ gemäß (3.3) hypergeometrisch in k ist?
2. Warum ist (2.5) linear bezüglich der $\sigma_j(n)$?
3. Wie kann man sicherstellen, daß der Träger von $G(n,k)$ beschränkt ist?
4. Wie und wann kann man mit Hilfe der Rekursion eine geschlossene Form finden?

3.2 Voraussetzungen und Teilaspekte

3.2.1 Voraussetzungen, damit $a(k)$ hypergeometrisch wird

Der Gosperalgorithmus verlangt hypergeometrische Eingaben. Damit wir ihn auf $a(k)$ anwenden können, fordern wir

1.
$$\frac{F(n,k)}{F(n,k-1)} \in \mathbb{K}(n,k) \tag{3.5}$$

und

2.
$$\frac{F(n,k)}{F(n-1,k)} \in \mathbb{K}(n,k). \tag{3.6}$$

Wie im folgenden Lemma 10 gezeigt wird, folgt dann, daß $a(k)$ hypergeometrisch ist. Vorher dürfen wir allerdings noch die Normierung $\sigma_0(k) := 1$ vornehmen. Wir setzen also

$$a(k) := F(n,k) + \sum_{j=1}^{l} \sigma_j(n)\ F(n-j,k). \tag{3.7}$$

Gibt es nämlich eine Lösung $\{\sigma_0(n), \sigma_1(n), \ldots, \sigma_l(n)\}$ für (3.2) mit $\sigma_0(n) \neq 0$, so gilt

$$0 = \sum_{i=0}^{l} \frac{\alpha_i(n)}{\alpha_0(n)} A(n-i) = A(n) + \sum_{i=1}^{l} \frac{\alpha_i(n)}{\alpha_0(n)} A(n-i).$$

Findet man eine Lösung $\{0, \sigma_1(n), \ldots, \sigma_l(n)\}$, so war die Ordnung l zu hoch gewählt. □

Nun kommen wir zum angekündigten Beweis.

Lemma 10 *Gelten die Voraussetzungen (3.5) und (3.6) und ist $a(k)$ gemäß (3.7) definiert, so folgt $a(k)/a(k-1) \in \mathbb{K}(n,k)$.*

Beweis:
Es gilt

$$\frac{a(k)}{a(k-1)} = \frac{1 + \sum_{j=1}^{l} \sigma_j(n)\ \frac{F(n-j,k)}{F(n,k)}}{1 + \sum_{j=1}^{l} \sigma_j(n)\ \frac{F(n-j,k-1)}{F(n,k-1)}} \cdot \frac{F(n,k)}{F(n,k-1)}. \tag{3.8}$$

Da wegen Voraussetzung (3.5)

$$\frac{F(n,k)}{F(n,k-1)} \in \mathbb{K}(n,k)$$

und wegen Voraussetzung (3.6)

$$\frac{F(n-j,k)}{F(n,k)} = \frac{F(n-j,k)}{F(n-j+1,k)} \cdot \frac{F(n-j+1,k)}{F(n-j+2,k)} \cdots \frac{F(n-1,k)}{F(n,k)} =$$

$$\prod_{i=0}^{j-1} \frac{F(n-i-1,k)}{F(n-i,k)} \in \mathbb{K}(n,k) \tag{3.9}$$

gilt, folgt die Behauptung. □

3.2.2 Das zu lösende Gleichungssystem ist linear

Bei der Einbettung des Gosperalgorithmus im Zeilbergeralgorithmus löst man die Gleichung (2.5) nicht nur nach den bisher unbekannten Koeffizienten von $f(k)$, sondern auch nach den $\sigma_j(n)$. Daß man trotzdem nur ein **lineares** Gleichungssystem zu lösen hat, wird in folgendem Lemma bewiesen:

Lemma 11 *Gelten für die Eingabe $F(n,k)$ die Vorraussetzungen (3.15) und (3.5), und $a(k)$ sei gemäß (3.7) definiert, dann bilden die Koeffizienten (bezüglich k) von*

$$p(k) - q(k+1)\ f(k) + r(k)\ f(k-1) = 0$$

ein lineares Gleichungssystem in den Variablen f_i, und $\sigma_j(n)$ mit $i = 0, 1, \ldots, d$, [1] *und $j = 1, 2, \ldots, l$.*

Beweis:
Es seien $B(n,k)$ und $C(n,k)$ bzw. $D(n,k)$, $E(n,k)$ jeweils teilerfremde Zähler und Nennerpolynome der Quotienten (3.5) und (3.6), nämlich

$$\frac{F(n,k)}{F(n-1,k)} =: \frac{B(n,k)}{C(n,k)} \tag{3.10}$$

und

$$\frac{F(n,k)}{F(n,k-1)} =: \frac{D(n,k)}{E(n,k)}. \tag{3.11}$$

Wir initialisieren

$$P_0(k) := \prod_{i=0}^{l-1} B(n-i,k) + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} C(n-i,k) \prod_{i=j}^{l-1} B(n-i,k), \tag{3.12}$$

$$Q_0(k) := D(n,k) \cdot \prod_{i=0}^{l-1} B(n-i,k-1) \tag{3.13}$$

und

$$R_0(k) := E(n,k) \cdot \prod_{i=0}^{l-1} B(n-i,k). \tag{3.14}$$

Wir zeigen zunächst, daß diese Initialisierung die Gleichung (2.2) erfüllt.
Dazu setzen wir (3.9) in (3.8) ein. Es ergibt sich

$$\frac{a(k)}{a(k-1)} = \frac{1 + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} \frac{F(n-i-1,k)}{F(n-i,k)}}{1 + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} \frac{F(n-i-1,k-1)}{F(n-i,k-1)}} \cdot \frac{F(n,k)}{F(n,k-1)} =$$

$$\frac{1 + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} \frac{C(n-i,k)}{B(n-i,k)}}{1 + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} \frac{C(n-i,k-1)}{B(n-i,k-1)}} \cdot \frac{D(n,k)}{E(n,k)}.$$

Wir erweitern die Ausdrücke im Nenner und im Zähler jeweils um ihre Hauptnenner:

$$\frac{a(k)}{a(k-1)} = \frac{\prod_{i=0}^{l-1} B(n-i,k) + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} C(n-i,k) \prod_{i=j}^{l-1} B(n-i,k)}{\prod_{i=0}^{l-1} B(n-i,k-1) + \sum_{j=1}^{l} \sigma_j(n) \prod_{i=0}^{j-1} C(n-i,k-1) \prod_{i=j}^{l-1} B(n-i,k-1)} \cdot$$

$$\frac{\prod_{i=0}^{l-1} B(n-i,k-1)}{\prod_{i=0}^{l-1} B(n-i,k)} \cdot \frac{D(n,k)}{E(n,k)} = \frac{P_0(k)}{P_0(k-1)} \frac{Q_0(k)}{R_0(k)}.$$

[1] d ist der Maximalgrad von $f(k)$.

Die Initialisierung (3.12) - (3.14) nimmt lediglich einen Teil des Schleifendurchlaufs mit $N_i = 1$ in Algorithmus 2 vorweg. Mit der Eingabe $a(k)/a(k-1)$ liefert Algorithmus 2 eine Lösung $(p(k), q(k), r(k))$. Wendet man den Algorithmus 2 stattdessen auf $Q_0(k)/R_0(k)$ an, erhält man eine Lösung $(P(k), Q(k), R(k))$ mit $Q(k) = c_1\, q(k)$, $R(k) = c_2\, r(k)$ und $p(k) = c_3\, P_0(k)\; P(k)$, mit Konstanten $c_1, c_2, c_3 \in \mathbb{K}\backslash\{0\}$. Nun enthält aber die Initialisierung (3.12) - (3.14) die Variablen $\sigma_j(n)$ nur noch im Polynom $P_0(k)$, und auch dort nur linear (d.h. es gibt keine Produkte der Unbekannten). Auch durch Iteration der Schleife in Algorithmus 2 können keine $\sigma_j(n)$ zurück nach $q(k)$ und $r(k)$ gelangen, und das Gleichungssystem der Koeffizienten von (2.5) bezüglich k hat die Form

$$c_3 P_0(k)\; P(k) - c_1 q(k+1)\; f(k) + c_2\, r(k)\; f(k-1) = 0.$$

In diesem Gleichungssystem tauchen die $\sigma_j(n)$ nur im ersten Term, und auch dort nur linear, und die f_i nur im zweiten und dritten Term, und dort ebenfalls nur linear auf. □

Die geänderte Initialisierung ändert an der Struktur des Algorithmus fast nichts. Läßt man sie weg, nimmt man nur eine Chance zur Beschleunigung nicht wahr. Der Algorithmus 2 mit Initialisierung gemäß Gosper kommt zum gleichen Ergebnis. Seine Dispersionsmenge $N = \mathrm{disp}_k(q_0(k), r_0(k+j))$ enthält immer die Zahl Eins, weil im Iterationsschritt für $N_i = 1$ die $\sigma_j(n)$ gemäß der oberen Initialisierung zu $p(k)$ hinzumultipliziert werden. □

3.2.3 Wie stellt man sicher, daß $G(n,k)$ einen endlichen Träger hat?

Eine hinreichende Bedingung dafür, daß $G(n,k)$ einen endlichen Träger hat, ist, daß $F(n,k)$ einen endlichen Träger bezüglich k hat, das heißt, es muß Intervallgrenzen $c(n), d(n) \in \mathbb{K}[n]$ geben mit

$$F(n,k) = 0 \tag{3.15}$$

für $F(n,k) \notin [c(n), d(n)]$.
Dieses wird im folgenden **Lemma 12** bewiesen.

Lemma 12 *Gelten (3.15), (3.5) und (3.6) für ein $F(n,k)$, und findet der Gosperalgorithmus ein $G(n,k) = s(k)$ gemäß (2.1), so hat dieses $G(n,k)$ einen einen endlichen Träger bezüglich k.*

Beweis:
Da

$$F(n,k) = 0$$

für alle $k \notin [c(n), d(n)]$ gilt, folgt

$$F(n-j,k) = 0$$

für alle $k \notin [c(n-j), d(n-j)]$ für alle $j \in \{1, 2, \ldots, l\}$ und somit

$$a(k) \overset{(3.7)}{=} F(n,k) + \sum_{j=1}^{l} \sigma_j(n)\ F(n-j,k) = 0$$

für alle $k \notin \bigcup_{j=0}^{l} [c(n-j), d(n-j)]$.

$$G(n,k) = s(k) = \frac{q(k+1)\ f(k)}{p(k)}\, a(k)$$

ist ein rationales Vielfaches (bezüglich k) von $a(k)$, hat also nur endlich viele Pole, nämlich die Nullstellen von $p(k)$.
Also ist

$$G(n,k) = 0$$

für alle $k \notin \bigcup_{j=0}^{l} [c(n-j), d(n-j)] \cup \{k | \, p(k) = 0\}$. □

Die Bedingung (3.6), also daß $F(n,k)$ einen endlichen Träger hat, wird in der Praxis meist vom Benutzer vorgegeben.
Man kann die Bedingung aber auch für bestimmte Terme algorithmisch überprüfen. Dabei geht man ähnlich wie in **Lemma 1** vor. Man stellt Rekursionen der Form (1.23) auf und findet so Schranken für den Träger. □

3.2.4 Anwendungen

Liefert der Zeilbergeralgorithmus eine Rekursion erster Ordnung, so gilt

$$-\sigma_1(n+1) = \frac{A(n+1)}{A(n)} \quad .$$

Findet man für diesen Ausdruck eine Darstellung (1.16), so kann man sofort eine geschlossene Form

$$A(n) = C \frac{(A_1)_n \cdot (A_2)_n \cdots (A_P)_n\, x^n}{(B_1)_n \cdot (B_2)_n \cdots (B_Q)_n\, n!}$$

gemäß (1.20) oder (1.22) in **Lemma 1** berechnen. □

Selbst wenn sich keine geschlossene Form finden läßt, kann man den Zeilbergeralgorithmus zum Überprüfen von Äquivalenzen von Summen benutzen. Liefert der Algorithmus nämlich die gleiche Rekursion für die beiden zu vergleichenden Ausdrücke, so braucht man nur noch die Gleichheit der Anfangswerte zu überprüfen.
Man kann zum Beispiel mit dem Petkovšekalgorithmus (Kapitel **4**) zeigen, daß es für keine der beiden Seiten von

$$\sum_{k=0}^{n} \binom{n}{k}^3 \overset{?}{=} \sum_{k=0}^{n} \binom{n}{k}^2 \binom{2k}{n} \tag{3.16}$$

eine Rekursion (3.2) erster Ordnung gibt.
Wendet man aber den Zeilbergeralgorithmus auf beide Seiten der Gleichung an, erhält man nach Multiplikation mit dem Hauptnenner jeweils die Rekursion

$$-8\,(n-1)^2\,\mathrm{A}(n-2) - (7\,n^2 - 7\,n + 2)\,\mathrm{A}(n-1) + \mathrm{A}(n)\,n^2 = 0.$$

Jetzt braucht man nur noch die Anfangswerte zu überprüfen. Wir erhalten

$$\binom{0}{0}^3 = 1 = \binom{0}{0}^2 \binom{0}{0}$$

für $n = 0$ und

$$\binom{1}{0}^3 + \binom{1}{1}^3 = 1 + 1 = 2 = 0 + 2 = \binom{1}{0}^2 \binom{0}{1} + \binom{1}{1}^2 \binom{2}{1}$$

für $n = 1$. □

3.3 Zusammenfassung in Pseudocode

Algorithmus 5 *Der Zeilbergeralgorithmus*

Eingabe: $F(n,k)$
Ausgabe: $\sum_{i=0}^{l} \sigma_i(n)\,A(n-i)$ (wie in (3.2))

1. Überprüfe, ob die Grundvorausetzungen (3.5), (3.6) und (3.15) gegeben sind, und definiere dabei die Polynome $B(k,n), C(k,n), D(k,n), E(k,n)$ gemäß (3.10) und (3.11)
2. Für $l = 1, 2, \ldots$
   (a)
   - Berechne die Polynome $P_0(k), Q_0(k), R_0(k)$ gemäß (3.12) - (3.14)
   - Berechne die Polynome $P(k), q(k), r(k)$ durch Anwendung des Algorithmus 2 auf $Q_0(k)/R_0(k)$ und setze $p(k) := P(k)\,P_0(k)$

   (b) Finde mit Lemma 8 eine obere Schranke d für den Grad einer Lösung $f(k)$ von (2.5) bezüglich k. Ist $d < 0$, dann findet der Algorithmus keine Lösung von $s(k)$ mit den gewünschten Eigenschaften und bricht ab.

   (c) Löse dieses lineare Gleichungssystem (2.54) in den Variablen $\sigma_0(n), \sigma_1(n), \ldots, \sigma_l(n)$, $f_0, f_1, \ldots, f_d$.
   Wenn es keine Lösung gibt, gehe zum nächsten Schleifendurchlauf, sonst verlasse die Schleife und gehe zu Schritt 3.
3. Gebe $\sum_{i=0}^{l} \sigma_i(n)\,A(n-i)$ aus.

3.4 Beispiele

1. Wir suchen eine geschlossene Form für $A(n)$ gemäß (3.1) mit

$$F(n,k) := \binom{n}{k}$$

oder in der Notation (1.15) für

$$A(n) = {}_1\mathrm{F}_0\left(\begin{matrix}-n\\-\end{matrix}\,\middle|\,-1\right).$$

Es gilt

$$\frac{F(n,k)}{F(n-1,k)} = \frac{n}{(n-k)} =: \frac{B(n,k)}{C(n,k)}$$

und

$$\frac{F(n,k)}{F(n,k-1)} = \frac{(n-k+1)}{k} =: \frac{D(n,k)}{E(n,k)}.$$

Da auch $F(n,k) = 0$ für alle $k \notin \{0,n\}$ gilt, sind alle Voraussetzungen nämlich (3.15), (3.5) und (3.6) erfüllt.
Wir setzen

$$a(k) := F(n,k) + \sigma_1(n)\, F(n-1,k) = \binom{n}{k} + \sigma_1(n)\binom{n-1}{k},$$

und initialisieren $p(k)$, $q(k)$ und $r(k)$ mit $P_0(k)$, $Q_0(k)$ und $R_0(k)$ gemäß Lemma 11.

$$P_0(k) := B(n,k-1) + \sigma_1(n)\, C(n,k) =$$

$$n + \sigma_1(n)\,(n-k)$$

$$Q_0(k) := D(n,k)\; B(n,k-1) = (n-k+1)\; n$$

$$R_0(k) := E(n,k)\; B(n,k) = k\,n$$

Algorithmus 2 angewandt auf

$$\frac{Q_0(n,k)}{R_0(n,k)} = \frac{n-k+1}{k}$$

liefert offensichtlich

$$P(k) = 1 \quad \Longrightarrow \quad p(k) = P_0(k) = n + \sigma_1(n)\,(n-k)$$

$$q(k) = n-k+1$$

und

$$r(k) = k.$$

Es gilt

$$\operatorname*{Grad}_k (\, q\,(k+1) + r\,(k)\,) = \operatorname*{Grad}_k (\,(n-k)+k\,) = \operatorname*{Grad}_k (n) = 0 <$$
$$1 = \operatorname*{Grad}_k (-2\,k+n) = \operatorname*{Grad}_k (\, q\,(k+1) - r\,(k)\,).$$

Also folgt Fall 1 aus Lemma 8:

$$\operatorname{Grad}(f\,(k)) = \operatorname{Grad}(p\,(k)) - \operatorname{Grad}(q\,(k+1) - r\,(k)) = 1 - 1 = 0.$$

$f\,(k) = f_0$ ist also konstant, und somit gilt nach (2.5):

$$0 = p\,(k) - (q\,(k+1) - r\,(k))\ f_0 =$$

$$n + \sigma_1\,(n)\,(n-k) - (-2\,k+n)\ f_0 =$$

$$k\ (-\sigma_1\,(n) + 2\,f_0) + (n + n\,\sigma_1\,(n) - n\,f_0) = 0$$

Wir lösen die Koeffizienten:

$$0 = -\sigma_1\,(n) + 2\,f_0 \quad \Longrightarrow \quad \sigma_1\,(n) = 2 f_0$$

und

$$0 = n\,(1 + \sigma_1\,(n) - f_0) \quad \Longrightarrow \quad 1 + 2\,f_0 - f_0 = 0 \quad \Longrightarrow \quad f_0 = -1 \quad \Longrightarrow \quad \sigma_1\,(n) = -2$$

und somit

$$A\,(n) - 2A\,(n-1) = 0.$$

Hier ist der Zeilbergeralgorithmus fertig, und wir haben ein Ergebnis, mit dem wir sogar eine geschlossene Form finden können:

$$\frac{A\,(n+1)}{A\,(n)} = 2,$$

also

$$A\,(n) = 2^n\,A\,(0) = 2^n.$$

2. Nun kommen wir wieder zum Beispiel (5)

$$F\,(n,k) := (-1)^k \binom{n}{k} \binom{4\,k}{n}$$

aus der Einleitung zurück.
Es gilt

$$\frac{F\,(n,k)}{F\,(n-1,k)} = -\,\frac{4\,k-n+1}{-n+k} =: \frac{B\,(n,k)}{C\,(n,k)}$$

und

$$\frac{F\,(n,k)}{F\,(n,k-1)} = \frac{(4\,k-1)\,(4\,k-3)\,(4\,k-3)\,(-4\,n+4\,k-4)}{(4\,k-n)\,(4\,k-n-1)\,(4\,k-n-2)\,(4\,k-3-n)} =: \frac{D\,(n,k)}{E\,(n,k)}.$$

Da auch $F(n,k)=0$ für alle $k \notin \{0,n\}$ gilt, sind alle Voraussetzungen nämlich (3.15), (3.5) und (3.6) erfüllt.
Für dieses Beispiel findet Zeilberger nur eine Rekursion dritter Ordnung, obwohl es auch eine erster Ordnung gibt. Wir ersparen uns das Scheitern im ersten und zweiten Schritt und gehen gleich zum dritten über:

$$a(k) := F(n,k) + \sum_{i=1}^{3} \sigma_j(k)\, F(n-j,k)$$

Wir initialisieren die Polynome $p(k)$, $q(k)$ und $r(k)$ gemäß Lemma 11 und erhalten

$$P_0(k) := \prod_{i=0}^{2} B(n-i,k) + \sum_{j=1}^{3} \sigma_j(n) \prod_{i=0}^{j-1} C(n-i,k) \prod_{i=j}^{2} B(n-i,k) =$$

$$\prod_{i=0}^{2} (4k-(n-i)+1) + \sum_{j=1}^{3} \sigma_j(n) \prod_{i=0}^{j-1} (-(n-i)+k) \prod_{i=j}^{2} (4k-(n-i)+1)$$

$$Q_0(k) := (4k-1)(4k-2)(4k-3)(-4n+4k-4)$$

und

$$R_0(k) := (4k-n+3)(4k-n+2)(-4k-n+1)(4k-n).$$

Mit Faktorisierungsalgorithmen in Computeralgebrasystemen kann man überprüfen, daß $P_0(k)$ irreduzibel ist. Also erfüllt die obige Initialisierung offensichtlich (2.7) und wir setzen $p(k) := P_0(k)$, $q(k) := Q_0(k)$ und $r(k) := R_0(k)$.
Um den Höchstgrad für $f(k)$ zu bestimmen, müssen wir allerdings den Grad von $p(k)$ ausrechnen. $p(k)$ ist die Summe von vier Polynomen dritten Grades, deren führende Koeffizienten sich nicht wegheben:

$$\begin{aligned}
p(k) :=\ & (64 - 16\,\sigma_1(n) + 4\,\sigma_2(n) - \sigma_3(n))\ k^3 \\
& + (-48\,n + 96 + 24\,\sigma_1(n)\,n - 20\,\sigma_1(n) - 9\,\sigma_2(n)\,n + 7\,\sigma_2(n) + 3\,\sigma_3(n)\,n - 3\,\sigma_3(n))\ k^2 \\
& + \Big(12\,n^2 - 48\,n + 44 - 9\,\sigma_1(n)\,n^2 + 25\,\sigma_1(n)\,n - 6\,\sigma_1(n) + 6\,\sigma_2(n)\,n^2 - 11\,\sigma_2(n)\,n \\
& + 3\,\sigma_2(n) - 3\,\sigma_3(n)\,n^2 + 6\,\sigma_3(n)\,n - 2\,\sigma_3(n)\Big)\,k + 6 - 11\,n + 4\,\sigma_2(n)\,n^2 - n^3 \\
& - 5\,\sigma_1(n)\,n^2 + 6\,n^2 - \sigma_2(n)\,n^3 + \sigma_3(n)\,n^3 + 2\,\sigma_3\,n - 3\,\sigma_2(n)\,n + 6\,\sigma_1(n)\,n - 3\,\sigma_3(n)\,n^2 \\
& + \sigma_1(n)\,n^3.
\end{aligned}$$

Also ist $\mathrm{Grad}(p(k)) = 3$.

$$q(k+1) = 256\,k^4 + (-256\,n + 384)\,k^3 + (176 - 384\,n)\,k^2 + (24 - 176\,n)\,k - 24\,n$$

$$\begin{aligned}
r(k) = {} & 256\,k^4 + (-256\,n + 384)\,k^3 + (176 + 96\,n^2 - 288\,n)\,k^2 \\
& + (-88\,n - 16\,n^3 + 72\,n^2 + 24)\,k + n^4 - 6\,n^3 + 11\,n^2 - 6\,n
\end{aligned}$$

Also gilt $\mathrm{Grad}(q(k+1)+r(k)) = 5 > 3 = \mathrm{Grad}(q(k+1)-r(k))$ und, nach Fall 1 von Lemma 8,

$$\mathrm{Grad}(f(k)) = \mathrm{Grad}(p(k)) - \mathrm{Grad}(q(k+1)-r(k)) = 3-3 = 0.$$

Also muß $f(k)$ konstant in k sein, und eingesetzt in (2.5) ergibt sich:

$$\begin{aligned}
&(-64+16\,\sigma_1(n)-4\,\sigma_2+\sigma_3(n))\,k^3+\Big(-96\,f_0\,n^2-96\,f_0\,n+48\,n-96-24\,\sigma_1(n)\,n\\
&\quad+20\,\sigma_1(n)+9\,\sigma_2(n)\,n-7\,\sigma_2(n)-3\,\sigma_3(n)\,n+3\,\sigma_3(n)\Big)\,k^2+\Big(-88\,f_0\,n+16\,f_0\,n^3\\
&\quad-72\,f_0\,n^2-12\,n^2+48\,n-44+9\,\sigma_1(n)\,n^2-25\,\sigma_1(n)\,n+6\,\sigma_1(n)-6\,\sigma_2(n)\,n^2\\
&\quad+11\,\sigma_2(n)\,n-3\,\sigma_2(n)+3\,\sigma_3(n)\,n^2-6\,\sigma_3(n)\,n+2\,\sigma_3(n)\Big)\,k-6+11\,n-f_0\,n^4\\
&\quad-\sigma_3(n)\,n^3-\sigma_1(n)\,n^3+n^3-6\,n^2-2\,\sigma_3(n)\,n+5\,\sigma_1\,n^2-4\,\sigma_2(n)\,n^2+3\,\sigma_3(n)\,n^2\\
&\quad-6\,\sigma_1(n)\,n-18\,f_0\,n-11\,f_0\,n^2+6\,f_0\,n^3+3\,\sigma_2(n)\,n+\sigma_2(n)\,n^3.
\end{aligned}$$

Gemäß Schritt 2c lösen wir nach der Methode der unbekannten Koeffizienten, und erhalten

$$\begin{aligned}
f_0&=-\frac{1}{3}\,\frac{1}{n}, & \sigma_1(n)&=\frac{4}{3}\,\frac{37\,n^2-42\,n+11}{9\,n^2-9\,n+2},\\
\sigma_2(n)&=\frac{16}{3}\,\frac{33\,n^3-106\,n^2+102\,n-29}{(3\,n-5)\,(9\,n^2-9\,n+2)}\ \text{und} & \sigma_3(n)&=\frac{64}{3}\,\frac{n^2-3\,n+2}{(3\,n-5)\,(3\,n-1)}
\end{aligned}$$

und somit die Rekursion

$$\begin{array}{lcl}
\frac{64}{3}\,\frac{n^2-3\,n+2}{(3\,n-5)\,(3\,n-1)}A(n-3) & + & \frac{16}{3}\,\frac{33\,n^3-106\,n^2+102\,n-29}{(3\,n-5)\,(9\,n^2-9\,n+2)}A(n-2)+\\
+\frac{4}{3}\,\frac{37\,n^2-42\,n+11}{9\,n^2-9\,n+2}\,A(n-1) & + & A(n)=0.
\end{array}\tag{3.17}$$

□

3.5 Wann terminiert der Zeilbergeralgorithmus?

Das Beispiel (5) zeigt, daß der Zeilbergeralgorithmus nicht immer die Rekursion niedrigster Ordnung findet. Nun stellt sich die Frage, ob man garantieren kann, daß der Algorithmus überhaupt eine Rekursion findet. Wie wir gleich in Lemma 13 beweisen, ist die Antwort zwar grundsätzlich "Nein". Allerdings gibt es die große Klasse der zulässigen hypergeometrischen Terme (3.19), für die man den Erfolg des Zeilbergeralgorithmus garantieren kann. Dieses wird im Anschluß in Satz 2 bewiesen.

Lemma 13 *Der Zeilbergeralgorithmus terminiert nicht für alle Eingaben*

Beweis durch Gegenbeispiel
Wir zeigen, daß der Zeilbergeralgorithmus bei der Eingabe von

$$F(n,k):=\frac{1}{n^2+k^2+1}\binom{n}{k}$$

für kein $l\in\mathbb{N}$ eine Rekursion (3.2) findet.
Es seien $B(n,k)$, $C(n,k)$, $D(n,k)$ und $E(n,k)$ gemäß (3.10) und (3.11) in Lemma 11 definiert, also

$$\frac{F(n,k)}{F(n-1,k)}=\frac{(n-1)^2+k^2+1}{n^2+k^2+1}\cdot\frac{n}{n-k}=:\frac{B(n,k)}{C(n,k)}$$

und

$$\frac{F(n,k)}{F(n,k-1)} = \frac{n^2+k^2+1}{n^2+(k-1)^2+1} \cdot \frac{n-k+1}{k} =: \frac{D(n,k)}{E(n,k)}.$$

Da auch $F(n,k) = 0$ für alle $k \notin \{0,n\}$ gilt, sind alle Voraussetzungen nämlich (3.15), (3.5) und (3.6) erfüllt.
Wir definieren

$$H(n,k) := n^2+k^2+1$$

und erhalten

$$B(n,k) = n\,H(n-1,k),$$
$$C(n,k) = (n-k)\;H(n,k),$$
$$D(n,k) = (n-k+1)\;H(n,k-1)$$

und

$$E(n,k) = k\,H(n,k).$$

Mit $\alpha_0(n) = 1$ gilt für $a(k)$ gemäß (3.7) die Gleichung (2.2)

$$\frac{a(k)}{a(k-1)} = \frac{P_0(k)}{P_0(k-1)} \cdot \frac{Q_0(k)}{R_0(k)}$$

mit $P_0(k)$, $Q_0(k)$und $R_0(k)$ gemäß (3.12) bis (3.14), insbesondere[2]

$$Q_0(k) := D(n,k) \cdot \prod_{i=0}^{l-1} B(n-i,k-1) =$$

$$(n-k+1)\,H(n,k-1) \cdot \prod_{i=0}^{l-1} (n-i)\;H(n-i-1,k-1) =$$

$$(n-k+1) \prod_{i=0}^{l-1} (n-i) \prod_{i=0}^{l} H(n-i,k-1) = (n-k+1)\;L(n,k-1)$$

mit

$$L(n,k) := \prod_{i=0}^{l-1} (n-i) \prod_{i=0}^{l} H(n-i,k) \tag{3.18}$$

und

$$R_0(k) := E(n,k) \cdot \prod_{i=0}^{l-1} B(n-i,k) =$$

$$k\,H(n,k) \cdot \prod_{i=0}^{l-1} (n-i)\,H(n-i-1,k) = k \prod_{i=0}^{l-1} (n-i) \prod_{i=0}^{l} H(n-i,k) = k\,L(n,k).$$

Wir zerlegen $Q_0(k)$ und $R_0(k)$ in jeweils zwei Teiler $\{n-k+1, L(n,k-1)\}$ bzw. $\{k, L(n,k)\}$ und zeigen, daß alle Paare von Teilern eine leere Dispersionsmenge haben:
Es gilt

$$H(n,k) = H(n+i,n+j) \quad \Longleftrightarrow \quad (i = 0 \,\text{und}\, j = 0)$$

[2] $P_0(k)$ braucht für den Beweis nicht explizit ausgeschrieben zu werden.

also auch $\mathrm{disp}_k\left(L\left(n,k-1\right),L\left(n,k\right)\right)=\{\}$.
Ferner gilt $\mathrm{disp}_k\left(L\left(n,k-1\right),k\right)=\{\}$, $\mathrm{disp}_k\left(n-k,L\left(n,k\right)\right)=\{\}$ und
$\mathrm{disp}_k\left(n-k,k\right)=\{\}$.
Somit ergibt sich

$$\mathrm{disp}_k\left(Q_0\left(k\right),R_0\left(k\right)\right)=\{\}\,.$$

Damit ist (2.7) erfüllt, und der Zeilbergeralgorithmus liefert Polynome $p\left(k\right)=c_p\,P_0\left(k\right)$, $q\left(k\right)=c_q\,Q_0\left(k\right)$ und $r\left(k\right)=c_r\,R_0\left(k\right)$, wobei c_p,c_q und c_r Konstanten sind.
Eingesetzt in (2.5) ergibt sich

$$p\left(k\right)=q\left(k+1\right)f\left(k\right)-r\left(k\right)\;f\left(k-1\right)=$$

$$c_pP_0\left(k\right)=c_q\left(n-k\right)\;L\left(n,k\right)\;f\left(k\right)-c_r\,k\,L\left(n,k\right)\;f\left(k-1\right)=$$

$$c_pP_0\left(k\right)=L\left(n,k\right)\left(\left(n-k\right)\;c_qf\left(k\right)-c_r\,k\,f\left(k-1\right)\right).$$

Also ist $L\left(n,k\right)$ ein Teiler von $P_0\left(k\right)$. Da aber $L\left(n,k\right)$ ein Teiler von $R_0\left(k\right)$ ist und nach (2.14)

$$1=\mathrm{ggt}\left(p\left(k\right),r\left(k\right)\right)=\mathrm{ggt}\left(P_0\left(k\right),R_0\left(k\right)\right)$$

gilt, folgt $L\left(n,k\right)=1$. Dies ist aber ein Widerspruch zur Definition (3.18) von $L\left(n,k\right)$ für alle $l\in\mathbb{N}$.
Also kann es keine Polynomlösung für (2.5) geben, und der Zeilbergeralgorithmus findet keine Rekursion (3.2). □

Der am Ende dieses Kapitels folgende Satz 2 gibt eine obere Schranke J für die Ordnung l der Rekursion (3.2) an, falls die Eingabe $F\left(n,k\right)$ ein zulässiger hypergeometrischer Term ist[3]. Zulässige hypergeometrische Terme definieren wir folgt:

Definition 7 *Zulässiger hypergeometrischer Term*

Ein zulässiger hypergeometrischer Term ist ein Term der Form

$$F\left(n,k\right):=P\left(n,k\right)\;\frac{\Gamma\left(\alpha_1k+\beta_1n+c_1\right)\cdots\Gamma\left(\alpha_pk+\beta_pn+c_p\right)}{\Gamma\left(\gamma_1k+\delta_1n+d_1\right)\cdots\Gamma\left(\gamma_qk+\delta_qn+d_q\right)}w^nx^k \tag{3.19}$$

mit $P\left(n,k\right)\in\mathbb{K}\left[n,k\right];\alpha_i,\beta_i,\gamma_i,\delta_i\in\mathbb{Z};c_i,d_i,w,x\in\mathbb{K}$.

Die folgenden Lemmata sind Vorbereitungen zum Beweis von Satz 2. Im folgenden wird viel mit Translationsoperatoren in k und n gerechnet. Diese sind folgendermaßen definiert.

$$N(F(n,k)):=F(n+1,k)$$

$$K(F(n,k)):=F(n,k+1)$$

[3]Die Beweisführung lehnt sich an [7], Seiten 239-241 an.

Lemma 14 *Sei $F(k,n)$ ein zulässiger hypergeometrischer Term gemäß (3.19) und sei der Operator $H(N,K,n)$ mit*

$$H(N,K,n) := \sum_{i=0}^{I} \sum_{j=0}^{J} a_{ij}(n) K^i N^j \tag{3.20}$$

definiert, dann ist $H(N,K,n) F(k,n)$ ebenfalls ein zulässiger hypergeometrischer Term.

Beweis: Seien nun $H(N,K,n)$ und $F(n,k)$ gemäß (3.20) bzw. (3.19) definiert, dann sei

$$\tilde{F}(n,k) = \frac{\prod_{l=1}^{p} \Gamma(\alpha_l k + \beta_l n + \alpha_l Is(\alpha_l) + \beta_l Js(\beta_l) + c_l)}{\prod_{l=1}^{q} \Gamma(\gamma_l k + \delta_l n + \gamma_l Is(-\gamma_l) + \delta_l Js(-\delta_l) d_l)} w^n x^k, \tag{3.21}$$

wobei

$$s(x) := \begin{cases} 1 & \text{falls } x < 0 \\ 0 & \text{sonst.} \end{cases}$$

Wir zeigen, daß für alle $i = 1, 2, \ldots, I$ und für alle $j = 1, 2, \ldots, J$ der Ausdruck $a_{ij}(n) K^i N^j F(n,k)$ ein polynomielles Vielfaches vom offensichtlich zulässigen hypergeometrischen Term $\tilde{F}(n,k)$ ist.
Um dies zu zeigen, betrachten wir jeweils das Argument des l. oberen Gammaterms von $F(n+j, k+i)$ und $\tilde{F}(n,k)$.
Wir setzen

$$A_l := \alpha_l (k+i) + \beta_l (n+j) + c_l$$

und

$$B_l := \alpha_l k + \beta_l n + \alpha_l Is(\alpha_l) + \beta_l Js(\beta_l) + c_l.$$

Wir zeigen, daß $A_l - B_l \in \mathbb{N}_0$ liegt und es somit ein Polynom $p_{ijl}(n,k)$ mit $\Gamma(A_l) = p_{ijl}(n,k)\, \Gamma(B_l)$ gibt.
Da $i \leq I$, folgt

$$\alpha_l i \geq 0 = I \alpha_l s(\alpha_l) \tag{3.22}$$

für $\alpha_l \geq 0$ und

$$\alpha_l i \geq I \alpha_l = I \alpha_l s(\alpha_l) \tag{3.23}$$

für $\alpha_l < 0$.
Analog zeigt man auch $\beta_l j \geq J \beta_l s(\beta_l)$, und es ergibt sich

$$A_l - B_l = \alpha_l (i - Is(\alpha_l)) + \beta_l (j - Js(\beta_l)) \geq 0. \tag{3.24}$$

Die

$$p_{ijl}(n,k) := \prod_{m=1}^{A_l - B_l} (A_l - m) \tag{3.25}$$

sind Polynome in k vom Grad $A_l - B_l$.

Für die unteren Gammaargumente

$$C_l := \gamma_l (k+i) + \delta_l (n+j) + c_l$$

und

$$D_l := \gamma_l k + \delta_l n + \gamma_l \, Is(-\gamma_l) + \delta_l Js(-\beta_l) + c_l$$

beweist man völlig analog $C_l - D_l \in \mathbb{N}_0$. Also gibt es Polynome

$$q_{ijl}(n,k) := \prod_{m=1}^{C_l - D_l} (D_l - m). \tag{3.26}$$

Wir wissen nun, daß

$$p_{ij}(n,k) := a_{ij}(n) \; P(n+j, k+i) \prod_{l=1}^{p} p_{ijl}(n,k) \prod_{l=1}^{q} q_{ijl}(n,k) \tag{3.27}$$

die gewünschten Polynome sind, um

$$a_{ij}(n) \, K^i N^j F(n,k) = p_{ij}(n,k) \, \tilde{F}(n,k) \tag{3.28}$$

zu erfüllen. □

Lemma 15 *Für zulässige hypergeometrische Terme $F(n,k)$ der Form (3.19) gibt es eine Rekursion*

$$f(n,k) := \sum_{i=0}^{I} \sum_{j=0}^{J} a_{ij}(n) \, F(n+j, k+i) = 0 \tag{3.29}$$

für genügend große J und I, genauer

$$I \geq 2B + 2D + \operatorname*{Grad}_k P(k,n) \text{ und } J \geq 2A + 2C + 1 \tag{3.30}$$

mit $A := \sum_{l=1}^{p} |\alpha_l|$, $B := \sum_{l=1}^{p} |\beta_l|$, $C := \sum_{l=1}^{p} |\gamma_l|$, und $D := \sum_{l=1}^{p} |\delta_l|$.

Beweis: In der Operatorschreibweise besagt der Satz, daß es einen Operator $H(N,K,n)$ gemäß (3.20) mit $H(N,K,n) \; F(n,k) = 0$ geben muß.
Wir definieren die $p_{ijl}(n,k)$, $q_{ijl}(n,k)$ und $p_{ij}(n,k)$ gemäß (3.25), (3.26) und (3.27).
Nach (3.28) folgt dann

$$\operatorname*{Grad}_k p_{ij}(n,k) = \operatorname*{Grad}_k a_{ij}(n) + \operatorname*{Grad}_k P(n+j, k+i) + \sum_{l=1}^{p} \operatorname*{Grad}_k p_{ijl}(n,k) + \sum_{l=1}^{q} \operatorname*{Grad}_k q_{ijl}(n,k). \tag{3.31}$$

Nun ist

$$\operatorname*{Grad}_k p_{ijl}(n,k) = \sum_{m=1}^{A_l - B_l} \operatorname*{Grad}_k (A_l - m) = \sum_{m=1}^{A_l - B_l} 1 = A_l - B_l. \tag{3.32}$$

Wir zeigen, daß

$$A_l - B_l \leq |\alpha_l| I + |\beta_l| J \tag{3.33}$$

gilt.
Für $\alpha_l \geq 0$ folgt

$$\alpha_l \left(i - I\, s\left(\alpha_l\right)\right) = -|\alpha_l| \left(i - I\right) \leq |\alpha_l| I, \tag{3.34}$$

und für $\alpha_l < 0$ folgt

$$\alpha_l \left(i - I\, s\left(\alpha_l\right)\right) = |\alpha_l|\, i \leq |\alpha_l| I. \tag{3.35}$$

Analog folgert man

$$\beta_l \left(j - J\, s\left(\beta_l\right) J\right) \leq |\beta_l|\, J. \tag{3.36}$$

Also gilt (3.33) und es folgt

$$\sum_{l=1}^{p} \operatorname*{Grad}_k p_{ijl}\left(n,k\right) = \sum_{l=1}^{p} \left(A_l - B_l\right) \leq I \sum_{l=1}^{p} |\alpha_l| + J \sum_{l=1}^{p} |\beta_l| = I\, A + J\, B. \tag{3.37}$$

Analog folgert man

$$\sum_{l=1}^{q} \operatorname*{Grad}_k q_{ijl}\left(n,k\right) \leq I\, C + J\, D. \tag{3.38}$$

Also folgt nach (3.31),(3.37) und (3.38)

$$\operatorname*{Grad}_k p_{ij} \leq 0 + \operatorname*{Grad}_k P\left(n,k\right) + I\left(A + C\right) + J\left(B + D\right) \tag{3.39}$$

und somit, da $f\left(n,k\right)$ über die p_{ij} summiert wird,

$$E := \operatorname*{Grad}_k f\left(n,k\right) \leq \operatorname*{Grad}_k P\left(n,k\right) + I\left(A + C\right) + J\left(B + D\right). \tag{3.40}$$

Da (3.29) für alle $k \in \mathbb{Z}$ gelten soll, führt die Gleichung zu einem System von $(E+1)$ homogenen Gleichungen in $(I+1)\,(J+1)$ Variablen.
Wählt man I und J so, daß $(I+1)\,(J+1) > (E+1)$, so ist das Gleichungssystem überbestimmt, und es gibt eine nichttriviale Lösung.
Dieses erreicht man durch Wahl von I und J mit

$$I \geq 2\,\left(B + D\right) + \operatorname*{Grad}_k P\left(n,k\right)$$

und

$$J \geq 2\,\left(A + C\right) + 1.$$

Dann ist nämlich

$$\left(I+1\right)\left(J+1\right) = \frac{1}{2} I\, J + \frac{1}{2} I\, J + I + J + 1 \leq$$

$$I\left(A + C\right) + J\left(B + D\right) + \left(2\,\left(B + D\right) + \operatorname*{Grad}_k P\left(n,k\right)\right) + + J + 1 \leq$$

$$I\left(A + C\right) + J\left(B + D\right) + \operatorname*{Grad}_k P\left(n,k\right) + 1 < E + 1$$

nach (3.40) □

Lemma 16 *Sei $F(n,k)$ ein zulässiger hypergeometrischen Term gemäß (3.19), dann gibt es für ein $J \in \mathbb{N}$ Polynome $\sigma_0(n), \sigma_1(n), \ldots, \sigma_J(n)$ mit mindestens einem $\sigma_j(n) \neq 0$, so daß*

$$\sum_{j=0}^{J} \sigma_j(n) F(n+j,k) = G(n,k+1) - G(n,k) \tag{3.41}$$

gilt.

Beweis:
Nach Lemma 15 gibt es ein $H(N,K,n)$ gemäß (3.20) mit $H(N,K,n)\, F(n,k) = 0$ mit minimalem I.
Da die Funktion

$$H(N,1,n) - H(N,K,n)$$

an der Stelle $K=1$ eine Nullstelle hat, können wir mit Hilfe des Euklidischen Algorithmus durch den Faktor $(K-1)$ dividieren und erhalten ein Polynom $h(N,K,n)$ mit

$$H(N,1,n) - H(N,K,n) = (K-1)\, h(N,K,n)$$

und somit

$$H(N,k,n) = H(N,1,n) - (K-1)\, h(N,K,n). \tag{3.42}$$

Da $H(N,1,n)$ unabhängig von K und ein Polynom vom Grad J bezüglich N ist, können wir es als

$$H(N,1,n) = \sum_{j=0}^{J} \sigma_j(n)\, N^j$$

schreiben.
Da $h(N,K,n)$ die Form (3.20) hat, ist nach Lemma 15 der Term

$$G(n,k) := h(N,K,n) F(n,k)$$

ein zulässiger hypergeometrischer Term.
Wenden wir (3.42) auf $F(n,k)$ an, erhalten wir

$$0 = H(N,K,n)\; F(n,k) = (H(N,1,n) - (K-1)\, h(N,K,n))\; F(n,k) =$$

$$\sum_{j=0}^{J} \sigma_j(n)\, N^j\, F(n,k) - (K-1)\, G(n,k).$$

Also gilt (3.41).
Wir müssen allerdings noch zeigen, daß es ein $\sigma_j(n) \neq 0$ gibt.
Angenommen, es würde $\sigma_j(n) = 0$ für alle $j \in \{0,1,\ldots,J\}$ gelten, dann wäre $G(n,k)$ unabhängig von k. Da $G(n,k)$ hypergeometrisch in n ist, müßte es Polynome $\rho_0(n), \rho_1(n)$ mit

$$\frac{G(n+1,k)}{G(n,k)} = -\frac{\rho_0(n)}{\rho_1(n)}$$

bzw. $(\rho_0(n) + \rho_1(n)N)\, G\,(n,k) = 0$ geben.
Also ist

$$(\rho_0(n) + \rho_1(n)N)\, h\,(N, K, n)$$

ein Differenzoperator, der wenn er auf $F\,(n,k)$ angewandt wird, Null liefert, nach (3.42) aber nur Grad $(I-1)$ hat. Das ist ein Widerspruch zur Minimalität von I. □

Satz 2 *Für Summen mit zulässigen hypergeometrischen Termen terminiert der Zeilbergeralgorithmus.*

Beweis:
In Lemma 16 wird bewiesen, daß es für zulässige hypergeometrischen Terme $F\,(n,k)$ gemäß (3.19) eine Stammfunktion $G\,(n,k)$ mit (3.41)

$$\sum_{j=0}^{J} \sigma_j\,(n)\, F\,(n+j,k) = G\,(n,k+1) - G\,(n,k)$$

für $J > 2\,A + 2\,C + 1$ gemäß (3.30) gibt.
Mit der in dieser Arbeit gewählten Schreibweise mit absteigenden Rekursionen müssen wir noch Umformungen vornehmen. Wir ersetzen n durch $n-J$ in (3.41) und erhalten

$$G\,(n-J,k+1) - G\,(n-J,k) = \sum_{j=J}^{0} \sigma_j\,(n-J)\, F\,(n+j-J,k) = \sum_{j=0}^{J} \sigma_{J-j}\,(n-J)\, F\,(n-j,k)\,.$$

Dann gilt

$$a\,(k) := \sum_{j=0}^{l} \tilde{\sigma}_j\,(n)\; F\,(n-j,k) = s\,(k) - s\,(k-1)$$

mit $\tilde{\sigma}_j\,(n) := \sigma_{J-j}\,(n-J)$ und
$s\,(k) := G\,(n-J,k+1)$.
Für das Finden einer Stammfunktion $s\,(k)$ ist der Gosperalgorithmus nach Satz 1 ein Entscheidungsalgorithmus. □

Bemerkung 6 *Zur Schranke J*

Die gelieferte obere Schranke J ist bei den meisten Beispielen deutlich höher als die von Zeilberger berechnete Ordnug l.
So ist für (5)

$$F\,(n,k) := (-1)^k \binom{n}{k} \binom{4\,k}{n}$$

aus der Einleitung

$$\frac{F\,(n,k)}{F\,(n,k-1)} = \frac{(\,4\,k-1\,)\,(\,4\,k-2\,)\,(\,4\,k-3\,)\,(\,-4n+4k-4\,)}{(\,4\,k-n\,)\,(\,4\,k-n-1\,)\,(\,4\,k-n-2\,)\,(\,4\,k-3-n\,).}$$

Gemäß (3.30) berechnen sich $A = 4 \cdot 4 = 16$, $C = 4 \cdot 4 = 16$ und somit $J = 33$. Der Zeilbergeralgorithmus findet zwar nicht die vorhandene Rekursion erster Ordnung (4.28), ist aber immerhin schon bei $l = 3$ erfolgreich. □

Bemerkung 7 *Zulässige hypergeometrische Terme erfüllen die Bedingungen (3.5) und (3.6).*

Beweis:
Es sei $F(n,k)$ ein zulässiger hypergeometrischer Term gemäß (3.19).
Nach (1.9) gilt

$$\frac{\Gamma(\alpha k + \beta n + c)}{\Gamma(\alpha k - \alpha + \beta n + c)} = \prod_{j=1}^{\alpha} (\alpha k + \beta n + c - j)$$

für $\alpha \in \mathbb{N}$ und somit

$$\frac{F(n,k)}{F(n,k-1)} = \frac{P(n,k)}{P(n,k-1)} \cdot \frac{\prod_{i=1}^{p} \prod_{j=1}^{\alpha_i} (\alpha_i k + \beta_i n + c_i - j)}{\prod_{i=1}^{q} \prod_{j=1}^{\gamma_i} (\delta_i k + \beta_i n + d_i - j)} \in \mathbb{K}(k,n).$$

Analog folgt

$$\frac{F(n,k)}{F(n-1,k)} = \frac{P(n,k)}{P(n,k-1)} \cdot \frac{\prod_{i=1}^{p} \prod_{j=1}^{\beta_i} (\alpha_i k + \beta_i n + c_i - j)}{\prod_{i=1}^{q} \prod_{j=1}^{\delta_i} (\delta_i k + \beta_i n + d_i - j)} \in \mathbb{K}(k,n).$$

□

Kapitel 4

Der Petkovšekalgorithmus

Marko Petkovšeks Algorithmus `hyper` findet alle hypergeometrischen Lösungen von Rekursionen

$$\sum_{i=0}^{l} \sigma_i(n)\ A(n+i) = 0 \tag{4.1}$$

mit polynomiellen Koeffizienten $\sigma_i(n) \in \mathbb{K}[n]$, das heißt, er führt (4.1) auf eine Rekursion 1. Ordnung

$$A(n+1) = S(n)\, A(n) \tag{4.2}$$

mit $S(n) \in \mathbb{K}(n)$ zurück, falls dies möglich ist. Insbesondere kann der Algorithmus überprüfen, ob es für Lösungen vom Zeilbergeralgorithmus mit $l > 0$ nicht doch eine Rekursion der Ordnung $l = 1$ existiert. Da die vom Zeilbergeralgorithmus gelieferten Koeffizienten $\sigma_i(n)$ allerdings rationale Funktionen sind, muß allerdings die Rekursion (3.2) noch mit ihrem Hauptnenner $\sigma_*(n)$ multipliziert werden. Man erhält

$$0 = \sigma_*(n) \sum_{j=0}^{l} \sigma_j(n)\ A(n-j) = \sum_{j=0}^{l} \sigma_*(n)\, \sigma_j(n)\ A(n-j)$$

mit $\sigma_*(n)\, \sigma_j(n) \in \mathbb{K}[n]$ für alle $j \in \{0, \ldots, l\}$.
eimem Ausdruck der Form (4.1) zu kommen, muß man nur noch eine Indexverschiebung von n auf $n+l$ durchführen[1] .

$$0 = \sum_{j=0}^{l} \sigma_*(n+l)\, \sigma_j(n+l)\ A(n+l-j) = \sum_{j=0}^{l} \sigma_*(n)\, \sigma_{l-j}(n+l)\ A(n+j)$$

4.1 Kurzer Überblick über den Algorithmus

Den Spezialfall, daß $A(n)$ ein Polynom ist, betrachtet Petkovšek in seinem Algorithmus `poly`. Dieser berechnet sogar eine geschlossene Form $A(n)$ und keine Rekursion. Der Algo-

[1] Der Petkovšekalgorithmus könnte ohne weiteres auch mit absteigenden Rekursionen beschrieben werden, die gewählte Darstellung ermöglicht aber eine erheblich elegantere Beweisführung.

rithmus besteht im wesentlichen aus zwei Schritten:

1. Finde eine obere Schranke $N \in \mathbb{N}_0$ für den Grad von $A(n)$ (siehe Abschnitt 4.2.1).

2. Setze
$$A(n) := \sum_{k=0}^{N} f_k \, n^k \tag{4.3}$$
mit zunächst unbekannten f_k, und da $T(n) = 0$ mit
$$T(n) := \sum_{i=0}^{l} \sigma_i(n) \; A(n+i) = \sum_{i=0}^{l} \sigma_i(n) \sum_{k=0}^{N} f_k \; (n+i)^k \tag{4.4}$$
für alle $n \in \mathbb{Z}$ gelten muß, können wir (4.4) als homogenes Gleichungssystem der Koeffizienten von $T(n)$ mit den f_i als Unbekannten lösen.

Im hypergeometrischen Fall nutzen wir —ähnlich wie im Gosperalgorithmus— die Eigenschaft von hypergeometrischen Lösungen, daß es nach Lemma 2 Polynome $p(n), q(n)$ und $r(n)$ mit
$$S(n) = \frac{p(n)}{p(n-1)} \cdot \frac{q(n)}{r(n)}$$
gibt.
Zu diesen Polynomen $p(n), q(n)$ und $r(n)$ und ihren führenden Koeffizienten $\tilde{p}, \tilde{q}$ und $\tilde{r}$ gibt es monische Polynome $P(n), Q(n)$ und $R(n)$, so daß
$$Q(n) = \frac{q(n)}{\tilde{q}},$$
$$R(n) = \frac{r(n)}{\tilde{r}}$$
und
$$P(n) = \frac{p(n-1)}{\tilde{p}}$$
gilt, und es eine Konstante
$$Z = \frac{\tilde{r}}{\tilde{q}},$$
mit
$$S(n) = Z \, \frac{Q(n)}{R(n)} \, \frac{P(n+1)}{P(n)} \tag{4.5}$$
gibt.
Die Eigenschaften (2.7), (2.13) und (2.14) gelten dann auch für $P(n+1), Q(n)$ und $R(n)$, da sie sich von $p(n)$, $q(n)$ und $r(n)$ nur durch konstante Vorfaktoren unterscheiden.
$P(n+1), Q(n)$ und $R(n)$ sind eindeutig, da sie monisch sind und da $p(n+1), q(n)$ und $r(n)$ eindeutig bis auf konstante Vorfaktoren sind. Als einziger freier Bestandteil der Gleichung (4.5) ist Z somit auch eindeutig.
Wir suchen statt nach der rationalen Funktion $S(n)$ nach drei Polynomen $Q(n)$, $R(n)$ und

$P(n)$ und nach einer Konstante Z. Dabei hilft uns, daß die Beschaffenheit der Polynome $\sigma_i(n)$ die Anzahl der möglichen Lösungen auf eine endliche Menge einschränkt.
Um das zu zeigen, gehen wir in vier Schritten vor:

1. Zunächst setzen wir (4.2) geeignet in die Rekursion höherer Ordnung (4.1) ein:
   Wenden wir (4.2) rekursiv an, ergibt sich

$$A(n+i) = A(n) \prod_{j=0}^{i-1} S(n+j),$$

   das eingesetzt in (4.1) gibt

$$A(n) \sum_{i=0}^{l} \sigma_i(n) \prod_{j=0}^{i-1} S(n+j) = 0. \tag{4.6}$$

2. Um zur polynomiellen Rekursion (4.8) zu kommen, formen wir geeignet um:
   Wir können (4.6) durch $A(n)$ teilen und (4.5) einsetzen:

$$0 = \sum_{i=0}^{l} \sigma_i(n) \prod_{j=0}^{i-1} \left(Z \frac{Q(n+j)}{R(n+j)} \frac{P(n+j+1)}{P(n+j)} \right) =$$

$$\sum_{i=0}^{l} \sigma_i(n)\ Z^i \frac{P(n+i)}{P(n)} \prod_{j=0}^{i-1} \left(\frac{Q(n+j)}{R(n+j)} \right) \tag{4.7}$$

   Wir multiplizieren mit dem Hauptnenner $P(n) \prod_{j=0}^{l-1} R(n+j)$ und erhalten

$$\sum_{i=0}^{l} Z^i \beta_i(n) P(n+i) = 0 \tag{4.8}$$

   mit

$$\beta_i(n) := \sigma_i(n) \prod_{j=0}^{i-1} Q(n+j) \prod_{j=i}^{l-1} R(n+j). \tag{4.9}$$

3. Wir machen uns die Eigenschaften (2.13) und (2.14) und die Eindeutigkeit der Lösung (4.5) zunutze, um $Q(n)$ und $R(n)$ als monische Teiler von $\sigma_0(n)$ bzw. von $\sigma_l(n-l+1)$ zu identifizieren. Somit schränken wir die Anzahl der möglichen $Q(n)$ und $R(n)$ auf eine endliche Zahl ein:
   Alle Terme $Z^i \beta_i(n) P(n+i)$ der Summe (4.8) mit $i < l$ enthalten $R(n+l-1)$ als Faktor, also ist $R(n+l-1)$ auch ein Teiler vom Term mit $i = l$:

$$R(n+l-1) \,|\, Z^l \sigma_l(n) \prod_{j=0}^{l-1} Q(n+j)\ P(n+l).$$

   Nach (2.14) gilt $1 = \operatorname{ggt}(r(k+l-1), p(k+l-1)) = \operatorname{ggt}(R(k+l+1), P(k+l))$ und somit

$$R(n+l-1) \,|\, Z^l \sigma_l(n) \prod_{j=0}^{l-1} Q(n+j).$$

Da Z^l konstant ist, und nach (2.7) $\operatorname{ggt}\left(R\left(n+l-1\right), Q\left(n+j\right)\right)$ für $j \leq l-1$, folgt

$$R\left(n+l-1\right) | \sigma_l\left(n\right) \quad \Longrightarrow \quad R\left(n\right) | \sigma_l\left(n-l+1\right). \tag{4.10}$$

Analog zeigt man

$$Q\left(n\right) | \sigma_0\left(n\right). \tag{4.11}$$

Und zwar enthalten alle Terme $Z^i \beta_i\left(n\right) P\left(n+i\right)$ der Summe (4.8) mit $i > 0$ das Polynom $Q\left(n\right)$ als Faktor, also ist $Q\left(n\right)$ auch ein Teiler vom Term mit $i = 0$:

$$Q\left(n\right) | \sigma_0\left(n\right) \prod_{j=0}^{l-1} R\left(n+j\right)\ P\left(n\right) \quad \stackrel{(2.13)}{\Longrightarrow}$$

$$Q\left(n\right) | \sigma_0\left(n\right) \prod_{j=0}^{l-1} R\left(n+j\right) \quad \stackrel{(2.7)}{\Longrightarrow} \quad Q\left(n\right) | \sigma_0\left(n\right).$$

4. Für die Auswahl der möglichen Z setzen wir zunächst

$$m := \max_i \operatorname{Grad}_n\left(\beta_i\left(n\right)\right) \tag{4.12}$$

und

$$a_i := \text{der Koeffizient vor } n^m \text{ in jedem } \beta_i\left(n\right) \tag{4.13}$$

mit $i = 0 \ldots l$. Da (4.8) für alle $n \in \mathbb{Z}$ gelten soll, kann es als homogenes Gleichungssystem der Koeffizienten von $P\left(n\right)$ aufgefaßt werden. Die Gleichung des führenden, $\left(\, m + \operatorname{Grad}\left(P\left(n\right)\right)\,\right)$.Koeffizienten sieht folgendermaßen aus:

$$\sum_{i=0}^{l} a_i Z^i = 0. \tag{4.14}$$

Somit reduziert sich die Anzahl der möglichen Z auf die nichttrivialen, rationalen Lösungen von (4.14).

Diese Lösungen kann man in Computeralgebrasystemen mit Hilfe der Standardfunktionen `solve` in MAPLE und REDUCE bzw. `Solve` in MATHEMATICA berechnen. Diese Funktionen benutzen Faktorisierungsalgorithmen, die unter anderem in [20], Seiten 321-327, beschrieben werden. Um das noch fehlende $P\left(n\right)$ zu berechnen, wenden wir mit allen möglichen Lösungen von $Q\left(n\right)$, $R\left(n\right)$ und Z den Algorithmus `poly` auf die Rekursion (4.8) an. Gibt es eine Lösung $P\left(n\right)$, so läßt sich $S\left(n\right)$ leicht aus (4.5) berechnen. □

Bemerkung 8 *zur Laufzeit von* `hyper`

`hyper` hat den Nachteil, keine polynomielle Laufzeit zu haben. Es muß hier beachtet werden, daß nicht nur die irreduziblen monischen Teiler der Polynome betrachtet werden müssen, sondern alle deren mögliche Produkte. Ein Polynom $p(n)$ vom Grad l hat höchstens l nichttriviale monische Teiler $t_1, \ldots, t_l$.
Die Menge der möglichen Teiler berechnet sich gemäß der Berechnung der Potenzmenge wobei der triviale Teiler 1 den Platz der leeren Menge einnimmt. Es ergibt sich:

$$|\text{monische Teiler von } \sigma_0(n)| \cdot |\text{monische Teiler von } \sigma_l(n+l-1)| \cdot$$

$$|\text{Lösungen von (4.14) bzgl. } Z| \leq$$

$$2^{\text{Grad}(\sigma_0(n))} \cdot 2^{\text{Grad}(\sigma_l(n+l-1))} \cdot l.$$

Diese Schranke läßt sich auch nicht durch geschickte Programmierung entscheidend verringern. Man beachte außerdem, daß sich die obige Schranke auf die Anzahl der Aufrufe des Algorithmus `poly` bezieht. Diesen werden wir jetzt näher betrachten.

4.2 Teilaspekte des Algorithmus

Bisher wurde in diesem Kapitel noch nicht erklärt, wie man Schritt 1 von `poly` vollzieht, also eine obere Schranke N für den Grad von $A(n)$ findet. Das wird jetzt nachgeholt. Die Bestimmung der Schranke ist leider recht aufwendig. Auch muß zwischendurch eine weitere Zahl s_0 (4.22) berechnet werden. Diese wird im Abschnitt 4.2.1 als gegeben angenommen, dann aber in Abschnitt 4.2.2 explizit berechnet.

4.2.1 Finden einer obere Schranke N für den Grad von $A(n)$

Um $T(n) = 0$ zu erfüllen, muß jeder Koeffizient von $T(n)$ zu Null werden. Wir gehen in drei Schritten vor:

1. Wir isolieren die Koeffizienten von $T(n)$ durch geschickte Umformungen.
2. Wir wählen einen bestimmten Koeffizienten aus.
3. Wir zeigen, daß die Gleichung des gewählten Koeffizienten nur für bestimmte N erfüllt werden kann. Das größte dieser N ist dann die gewünschte Schranke.

Nehmen wir also an, es gäbe ein $N := N_0$, so daß

$$A(n) = \sum_{j=0}^{N_0} f_k \, n^k \tag{4.15}$$

mit $f_k \in \mathbb{Q}$ für alle $k \in \{1, 2, \ldots, N_0\}$ und $f_{N_0} \neq 0$ gilt, so daß $T(n) = 0$ erfüllt ist.
Für die Umformungen müssen wir uns Polynome $\sigma_i(n)$ allerdings genauer betrachten.
Es seien

$$m := \max_{i=1,2,\ldots,l} \left\{ \operatorname*{Grad}_n \sigma_i(n) \right\}. \tag{4.16}$$

Es sei ferner $a_{i,j}$ die Koeffizienten der $\sigma_i(n)$ in der Form, daß

$$\sigma_i(n) =: \sum_{j=0}^{m} a_{i,j}\, n^{m-j} \tag{4.17}$$

ist. Wir setzen nun (4.17) und (4.15) in $T(n) = 0$ (gemäß 4.4) ein und erhalten:

$$T(n) = \sum_{i=0}^{l} \sum_{j=0}^{m} a_{i,j}\, n^{m-j} \sum_{k=0}^{N_0} f_k\, (n+i)^k = 0.$$

Nach dem Binomialsatz folgt

$$T(n) = \sum_{i=0}^{l} \sum_{j=0}^{m} a_{i,j}\, n^{m-j} \sum_{k=0}^{N_0} f_k \sum_{p=0}^{k} \binom{k}{p} i^{k-p}\, n^p.$$

Wir setzen

$$a_{i,j} := 0 \tag{4.18}$$

für alle $i \notin \{1, 2, \ldots, l\}$ und für alle $j \notin \{1, 2, \ldots, m\}$ und können so wegen $\binom{k}{p} = 0$ für $p \notin \{1, 2, \ldots, k\}$ drei der vier Summen über ganz $\mathbb{Z}$ laufen lassen:

$$T(n) = \sum_i \sum_j a_{i,j} n^{m-j} \sum_{k=0}^{N_0} f_k \sum_p \binom{k}{p} i^{k-p}\, n^p = \sum_p n^{m-j+p} \left(\sum_{k=0}^{N_0} f_k \sum_p \sum_i \binom{k}{p} i^{k-p} a_{i,j} \right).$$

Wir ersetzen j durch $r := m - j + p$ und erhalten

$$T(n) = \sum_r n^r \left(\sum_{k=0}^{N_0} f_k \sum_p \sum_i \binom{k}{p} i^{k-p} a_{i,m+p-r} \right).$$

Wir setzen $j := k - p$ und $s := k + m - r$ und erhalten

$$T(n) = \sum_r n^r \left(\sum_{k=0}^{N_0} f_k \sum_p \sum_i \binom{k}{p} i^j a_{i,s-j} \right) = \sum_r n^r \left(\sum_{k=0}^{N_0} f_k \sum_p \binom{k}{p} b_j^{(s)} \right) \tag{4.19}$$

mit

$$b_j^{(s)} := \sum_{i=0}^{l} i^j\, a_{i,s-j}. \tag{4.20}$$

Damit sind die Koeffizienten

$$\sum_{k=0}^{N_0} f_k \sum_p \binom{k}{p} b_j^{(s)} \tag{4.21}$$

von $T(n)$ bereits isoliert. Bevor wir jedoch zum zweiten Schritt kommen, sehen wir uns die Koeffizienten noch etwas genauer an. Zu diesem Zweck müssen wir noch eine weitere Zahl

$$s_0 := \min\left\{ s \in \mathbb{N}_0 \middle| \exists j \in \mathbb{N}_0 \text{ mit } b_j^{(s)} \neq 0 \right\} \tag{4.22}$$

berechnen.
Für alle $j > s$ gilt $a_{i,j-s} = 0$ und somit auch $b_j^{(s)} = 0$. Deshalb können wir s_0 iterativ berechnen.
In Abschnitt 4.2.2 werden wir sehen, wie man s_0 findet. Nach Definition von s_0 haben wir $b_j^{(s)} = 0$, falls $s = k + m - r < s_0$, also falls $k < s_0 - m + r$ ist. Somit können wir (4.21) als

$$\sum_{k=s_0-m+r}^{N_0} f_k \sum_p \binom{k}{p} b_{k-p}^{(k+m-r)}$$

schreiben.
Wir ersetzen r durch $t = s_0 - m + r$ und erhalten

$$\sum_{k=t}^{N_0} f_k \sum_p \binom{k}{p} b_{k-p}^{(k-t+s_0)} \quad . \tag{4.23}$$

Jetzt sind wir endlich soweit, den zweiten Schritt auszuführen:
Wir wählen den (N_0)-ten Koeffizienten von $T(n)$ durch Einsetzen von $t = N_0$ in (4.23):

$$f_{N_0} \sum_p \binom{N_0}{p} b_{N_0-p}^{(s_0)} = 0. \tag{4.24}$$

Wegen $f_{N_0} \neq 0$ und durch Ersetzen von p durch $j = N_0 - p$ erhalten wir

$$\sum_{j=0}^{s_0} \binom{N_0}{j} b_j^{(s_0)} = 0. \tag{4.25}$$

Dieses kann als Polynom bezüglich N_0 vom Grad s_0 aufgefaßt werden und wiederum mit Hilfe der Solve-Funktionen der Computeralgebrasysteme gelöst werden. Die Lösungen aus $\mathbb{N}_0$ sind dann alle möglichen Grade von $A(n)$, und es gilt das folgende Korollar.

Korollar 1 *Ist $A(n)$ ein Polynom in n, so gilt*

$$\operatorname*{Grad}_n (A(n)) \leq \{N_0 \in \mathbb{N}_0 | N_0 \text{ ist eine Lösung für } (4.25)\}.$$

□

4.2.2 Die Bestimmung von s_0

Zur Berechnung von s_0 gemäß (4.22) benutzen wir folgenden Algorithmus:

Algorithmus 6 *zur Berechnung von s_0 gemäß (4.22)*

Eingabe: Die Koeffizienten der σ_i gemäß (4.17)
Ausgabe: $s_0 := \min\left\{s \in \mathbb{N}_0 | \exists j \in \mathbb{N}_0 \text{ mit } b_j^{(s)} \neq 0\right\}$ (4.22)

1. Initialisiere $s := -1$;
2. Wiederhole
   (a) $s := s + 1$
   (b) Für $j := 0, 1, \ldots, s$ berechne

$$b_j^{(s)} := \sum_{i=0}^{l} i^j \, a_{i,s-j}$$

(gemäß (4.20)) bis ein $j \in \{0, 1, \ldots, s\}$ mit $b_j^{(s)} \neq 0$ gefunden ist, oder $s = l$.
War der Algorithmus bis $s = l$ nicht erfolgreich, breche ab.

□

Es genügt, die $j \leq s$ zu betrachten, da für alle $j > s$ wegen $a_{i,s-j} = 0$ auch $b_j^{(s)} = 0$ gilt.
Ansonsten ist der Algorithmus selbsterklärend. Wir müssen allerdings noch beweisen, daß der Algorithmus eine Lösung liefert, sofern es eine gibt.

Lemma 17 $s_0 \leq l$

Beweis:
Sei der Algorithmus bei $s = s_0$ angelangt,
dann ist $b_j^{(s)} = 0$ für $0 \leq j \leq s < s_0$.
Insbesondere gilt $b_s(s) = 0$ für $0 \leq s < s_0$.
Wäre $s_0 > d$, so wäre

$$b_s(s) = \sum_{i=0}^{d} i^s \, a_{i,0}$$

für $0 \leq s < d$.
Da $\det(i^s)_{i,s=0}^{l}$ die Vandermonde-Determinante $V(0, 1, \ldots, d)$ ist, folgt $a_{i,0} = 0$ für alle $i \in \{0, 1, \ldots, d\}$. Das ist aber ein Widerspruch zur Voraussetzung der Existenz eines $c_{i,0} \neq 0$.
□
Ein Beispiel für die Funktionsweise des Algorithmus 6 wird im Rahmen eines Beispiels für den Petkovšekalgorithmus in Abschnitt 4.5 gegeben.

4.3 Die Algorithmen `poly` und `hyper` in Pseudocode

Algorithmus 7 `poly`

Eingabe:
Polynome

$$\sigma_i(n) = \sum_{j=0}^{m} a_{i,j}\, n^{m-j}$$

gemäß (4.17) mit $i = 0, 1, \ldots, l$, es gelte $a_{i,j} \in \mathbb{K}$ und es gebe mindestens ein $a_{i,0} \neq 0$.
Wir nehmen außerdem an, daß $a_{i,j} = 0$, falls $j \notin \{0, 1, \ldots, m\}$.
Ausgabe:
Eine Basis B für den Raum der polynomiellen Lösungen von

$$0 = \sum_{i=0}^{l} \sigma_i(n)\, A\,(n+i) \tag{4.26}$$

1. Bestimme s_0, und berechne $b_j^{(s_0)}$ für alle $j \in \{0, \ldots, s_0\}$ mit Hilfe von Algorithmus 6.

2. Berechne die Menge der D der Wurzeln in N_0 von

$$D(N) := \sum_{j=0}^{s_0} \left(\begin{array}{c} N_0 \\ j \end{array} \right) b_j^{(s_0)}. \tag{4.27}$$

3. Ist $D = \{\}$, dann setze $B := 0$,
   sonst:

   (a) Setze $k :=$max$D$;

   (b) Finde eine Basis B für den Raum von polynomiellen Lösungen von (4.26) über $\mathbb{K}$ mit Hilfe der Lösung des folgenden linearen Gleichungssystems:
   Setze

$$A(n) := \sum_{k=0}^{N} f_k\, n^k$$

   und berechne die f_i mit der Methode der unbekannten Koeffizienten.

4. Gebe B zurück.

□

Algorithmus 8 `hyper`

Eingabe: Polynome $\sigma_i(n)$ gemäß (4.17)
Ausgabe: Eine hypergeometrische Lösung $S(n)$ von (4.2)

1. Finde alle monischen Faktoren $Q(n)$ von $\sigma_0(n)$ und $R(n)$ von $\sigma_l(n-l+1)$ mit Hilfe der Solve -Funktionen und Faktorisierungsalgorithmen in Computeralgebrasystemen.
2. Für alle Paare von $Q(n)$ und $R(n)$:
   (a) Stelle die Gleichung
   $$\sum_{i=0}^{l} Z^i \beta_i(n) P(n+i)$$
   gemäß (4.9), (4.12) und (4.13) auf, und wende darauf `poly` an.
   (b) Falls es eine Lösung $P(n)$ gibt, setze
   $$S(n) := \frac{Q(n)}{R(n)} \cdot \frac{P(n+1)}{P(n)}$$
3. Gebe $S(n)$ aus.

□

4.4 Der Petkovšekalgorithmus ist ein Entscheidungsalgorithmus

In diesem Abschnitt fasse ich noch einmal die wichtigste Errungenschaft des Petkovšekalgorithmus zusammen. Diese ist, daß er für für eine Rekursion (4.1) eine hypergeometrische Lösung (4.2) liefert, sofern sie existiert.

Satz 3 *Der Petkovšekalgorithmus ist ein Entscheidungsalgorithmus.*

Sei $A(n)$ ein Ausdruck, für den eine Rekursionsformel (4.1)

$$\sum_{i=0}^{l} \sigma_i(n)\ A(n+i) = 0$$

gilt. Existiert eine hypergeometrische Lösung $S(n)$ gemäß (4.2), so gibt es eindeutige $P(n)$, $Q(n)$ und $R(n) \in \mathbb{K}[n]$ und $Z \in \mathbb{K}$, so daß

$$\frac{A(n+1)}{A(n)} = S(n) \overset{(4.5)}{=} Z \frac{Q(n)}{R(n)} \frac{P(n+1)}{P(n)}$$

gilt.
Für $Q(n)$, $R(n)$, und Z muß (4.10) (4.11) und (4.14) gelten. `hyper` sucht also für jedes mögliche Tripel ($Q(n)$, $R(n)$,Z) nach einer Lösung.
Korollar 1 liefert eine Schranke für den Grad der Lösungen von `poly`. In Schritt 3a von `poly` werden sämtliche Lösungen gefunden die innerhalb dieser Beschränkung liegen. Somit kann `poly` entscheiden, ob $P(n)$ existiert.
Also entscheidet `hyper`, ob die rechte Seite der Gleichung (4.5) existiert, was gleichbedeutend zu der Existenz von $S(n)$ ist. □

4.5 Beispiele

Wir betrachten wieder das Beispiel (5)

$$F(n,k) := (-1)^k \binom{n}{k} \binom{4k}{n},$$

an dem der Gosperalgorithmus scheiterte und für das der Zeilbergeralgorithmus die Rekursion (3.17)

$$\frac{64}{3}\,\frac{n^2-3n+2}{(3n-5)(3n-1)}A(n-3)+\frac{16}{3}\,\frac{33n^3-106n^2+102n-29}{(3n-5)(9n^2-9n+2)}A(n-2)+$$

$$+\frac{4}{3}\,\frac{37n^2-42n+11}{9n^2-9n+2}\,A(n-1)+A(n)=0$$

liefert.
Multipliziert man mit dem Hauptnenner erhält man

$$0=64\,(n-1)(n-2)(3n-2)\,A(n-3)+16\,(n-1)\left(33n^2-73n+29\right)A(n-2)+$$

$$4\,(3n-5)\left(-42n+11+37n^2\right)A(n-1)+3\,(3n-1)(3n-5)(3n-2)\,A(n).$$

Wir ersetzen n durch $n+3$, um eine aufsteigende Rekursion zu erhalten:

$$0=64\,(n+2)(n+1)(3n+6)\,A(n)+16\,(n+2)\left(33n^2+125n+107\right)A(n+1)+$$

$$4\,(3n+4)\left(-42n-115+37\,(n+3)^2\right)A(n+2)+3\,(3n-1)(3n+4)(3n+7)\,A(n+3)$$

Wir durchlaufen jetzt die Schleife für alle monischen Faktoren $Q(n)$ und $R(n)$ von

$$\sigma_0(n)=64\,(n+2)(n+1)(3n+6)=192\,(n+2)(n+1)\left(n+\frac{1}{3}\right)$$

bzw.

$$\sigma_3(n-2)=3\,(-2+3n)\,(3n+2)\,(1+3n)=81\,\left(n-\frac{2}{3}\right)\left(n+\frac{2}{3}\right)\left(n+\frac{1}{3}\right).$$

Jedes der Polynome hat drei nichttriviale monische Teiler; also $2^3 \cdot 2^3 = 64$ mögliche Kombinationen von $Q(n)$ und $R(n)$.
Nun zur Bestimmung der möglichen Z. Um die Koeffizienten der Polynome sichtbar zu machen, expandieren wir sie:

$$\sigma_0(n) = 192\,n^3 + 1024\,n^2 + 1728\,n + 896$$

$$\sigma_1(n) = 528\,n^3 + 3056\,n^2 + 5712\,n + 3424$$

$$\sigma_2(n) = 444\,n^3 + 2752\,n^2 + 5496\,n + 3488$$

$$\sigma_3(n) = 81\,n^3 + 513\,n^2 + 1044\,n + 672$$

Also $m = \max_i \operatorname{Grad} \sigma_i(n) = 3$, und somit bezeichen die a_i jeweils die Koeffizienten vor n^3 in $\sigma_i(n)$.
Wir suchen nun die nichttrivialen, rationalen Lösungen von (4.14):

$$0 = \sum_{i=0}^{3} a_i Z^i = 192 + 3424\,Z + 444\,Z^2 + 81\,Z^3 = 3\,(Z+4)\,(27\,Z^2 + 40\,Z + 16)$$

Also kann nur $Z = -4$ gelten.
Wir ersparen uns die Betrachtung aller möglichen 64 Paare von Teilern, untersuchen ein Paar das kein Ergebnis liefert, und kommen dann gleich zum richtige Paar. Für Paar $Q(n) := n+1$ und $R(n) := n + 1/3$ eröffnen wir eine MAPLE -Sitzung:

```
> q:= (n+1);
```

$$q := n + 1$$

```
> r:=n+1/3;
```

$$r := n + \frac{1}{3}$$

Wir berechnen die $\beta_i(n)$

```
> bet:= sigma[x] *product(subs(n=n+j,q),j=0..x-1)
> *product(subs(n=n+j,r),j=x..3):
> beta[0]:=simplify_combinatorial(eval(subs(x=0,bet)));
```

$$\beta_0 := \frac{64}{81}\,(n+2)\,(n+1)\,(7+3\,n)^2\,(3\,n+10)\,(3\,n+4)\,(3\,n+1)$$

```
> beta[1]:=simplify_combinatorial(eval(subs(x=1,bet)));
```

$$\beta_1 := \frac{16}{27}(n+2)\,(33\,n^2 + 125\,n + 107)\,(n+1)\,(3\,n+10)\,(7+3\,n)\,(3\,n+4)$$

```
> beta[2]:=simplify_combinatorial(eval(subs(x=2,bet)));
```

$$\beta_2 := \frac{4}{9}(218 + 37\,n^2 + 180\,n)\,(3\,n+4)\,(n+2)\,(n+1)\,(3\,n+10)\,(7+3\,n)$$

```
> beta[3]:=simplify_combinatorial(eval(subs(x=3,bet)));
```

$$\beta_3 := (3n+8)(7+3n)(3n+4)(n+3)(n+2)(n+1)(3n+10)$$

Eingesetzt in (4.8) ergibt sich

```
> eval(sum((-4)^i*beta[i]*P[n+i],i=0..3));
```

$$\frac{64}{81}(n+2)(n+1)(7+3n)^2(3n+10)(3n+4)(3n+1)P_n - \frac{64}{27}(n+2)(33n^2+125n+107)(n+1)(3n+10)(7+3n)(3n+4)P_{n+1} + \frac{64}{9}(218+37n^2+180n)(3n+4)(n+2)(n+1)(3n+10)(7+3n)P_{n+2} - 64(3n+8)(7+3n)(3n+4)(n+3)(n+2)(n+1)(3n+10)P_{n+3}$$

Um die Koeffizienten der Polynome in der Rekursion sichtbar zu machen, expandieren wir[2].

```
> convert([seq(expand((-4)^i*beta[i])*P[n+i], i=0..3)], `+`);
```

$$\left(\frac{37888}{3}n^5 + \frac{311680}{9}n^4 + \frac{1447616}{27}n^3 + \frac{3735808}{81}n^2 + 2432n^6 + 192n^7 + \frac{535808}{27}n + \frac{250880}{81}\right)P_n + \left(-29120n^6 - 167808n^5 - 2112n^7 - \frac{4713920}{9}n^4 - \frac{25810496}{27}n^3 - \frac{27537920}{27}n^2 - \frac{15902464}{27}n - \frac{3834880}{27}\right)P_{n+1} + \left(\frac{20189440}{9}n^4 + 659776n^5 + \frac{13426048}{3}n^3 + \frac{47050240}{9}n^2 + \frac{9907712}{3}n + \frac{7813120}{9} + 105600n^6 + 7104n^7\right)P_{n+2} + (-81216n^6 - 533952n^5 - 1905600n^4 - 3977216n^3 - 4840704n^2 - 3171328n - 5184n^7 - 860160)P_{n+3}$$

Offensichtlich ist der Maxilmalgrad der Eingabepolynome für `poly` $m = 7$.

```
> m:= max(seq(degree((-4)^i*beta[i],n),i=0..3));
```

$$m := 7$$

Wir berechnen die $b_j^{(s)}$ gemäß (4.20).

```
> btemp:= 0:
> for i from 0 to 3 do tmp:= expand((-4)^i*beta[i]):
> btemp:= btemp+coeff(tmp,n,7): od:
> b[0,0]:= btemp;
```

$$b_{0,0} := 0$$

```
> btemp:=0:
> for i from 0 to 3 do tmp:= expand((-4)^i*beta[i]):
> btemp:= btemp+coeff(tmp,n,6): od:
> b[1,0]: =btemp;
```

$$b_{1,0} := -2304$$

[2]Dieser Schritt ist nicht erforderlich, erleichtert aber dem Leser das Nachvollziehen der Rechnungen.

Somit ist $s_0 = 1$. Es fehlt aber noch

```
> btemp:=0:
> for i from 0 to 3 do tmp:= expand((-4)^i*beta[i]):
> btemp:= btemp+i *coeff(tmp,n,6): od:
> b[1,1]:=btemp;
```

$$b_{1,1} := -61568$$

Wir berechnen jetzt die Wurzeln in N_0 von (4.27)

```
> binomial(N[0],0) * b[1,0] + binomial(N[0],1) * b[1,1];
```

$$-2304 - 61568\, N_0$$

```
> solve(",N[0]);
```

$$\frac{-18}{481}$$

Da $-18/481 \notin \mathbb{N}_0$ gilt, kommt das gewählte Paar $(Q(n), R(n))$ nicht als Lösung infrage. Nun kommen wir zum erfolgreichen Paar, das aus den trivialen monischen Teilern

$$R(n) := Q(n) := 1$$

besteht. Wir brauchen hier keine MAPLE -Sitzung, und müssen die $\beta_i(n)$ auch nicht gesondert berechnen. Wir setzen direkt in (4.8) ein, und es ergibt sich

$$0 = \sum_{i=0}^{3} Z^i \beta_i(n)\, P(n+i) = (-4)^i\, \sigma_i(n)\; P(n+i) =$$

$$\begin{aligned}
&(\,1024\,n^2 + 192\,n^3 + 1728\,n + 896\,)\, P(n) \\
&+ (-4)\;(\,528\,n^3 + 3056\,n^2 + 5712\,n + 3424\,)\, P(n+1) \\
&+ (-4)^2\;(\,5496\,n + 3488 + 2752\,n^2 + 444\,n^3\,)\, P(n+2) \\
&+ (-4)^3\;(\,81\,n^3 + 513\,n^2 + 1044\,n + 672\,)\, P(n+3) =
\end{aligned}$$

$$\begin{aligned}
&(\,192\,n^3 + 1024\,n^2 + 1728\,n + 896\,)\, P(n) \\
&+ (\,-2112\,n^3 - 12224\,n^2 - 22848\,n - 13696\,)\, P(n+1) \\
&+ (7104\,n^3 + 44032\,n^2 + 87936\,n + 55808 + 44032\,)\, P(n+2) \\
&+ (\,-5184\,n^3 - 32832\,n^2 - 66816\,n - 43008\,)\, P(n+3).
\end{aligned}$$

Wir müssen zunachst s_0 mit Hilfe von Algorithmus 6 bestimmen.
Wir beginnen mit $s := 0, j := 0$.

$$b_0^{(0)} := \sum_{i=0}^{3} a_{i,0} = 192 - 2112 + 7104 - 5184 = 0$$

$s := 1, j := 0$

$$b_0^{(1)} := \sum_{i=0}^{3} a_{i,1} = 1024 - 12224 + 44032 - 32832 = 0$$

$s := 1, j := 1$

$$b_1^{(1)} := \sum_{i=0}^{3} i\, a_{i,1} = -2112 + 14208 - 15552 = -3456 \neq 0$$

Somit ist $s_0 = 1$ und

$$D(N_0) = \binom{N_0}{1} - 3456 = -3456 N_0 \quad \Longrightarrow \quad N_0 = 0 \in \mathbb{N}_0.$$

$P(n) := f_0$ ist also eine Konstante, und wir setzten

$$\begin{aligned} & 192\,n^3 + (\,1024\,n^2 + 1728\,n + 896\,)\,f_0 \\ & \quad + (\,-2112\,n^3 - 12224\,n^2 - 22848\,n - 13696\,)\,f_0 \\ & \quad + (7104\,n^3 + 44032\,n^2 + \ 87936\,n + 55808 + 44032\,)\,f_0 \\ & \quad + (\,-5184\,n^3 - 32832\,n^2 - 66816\,n - 43008\,)\,f_0 = \end{aligned}$$

$$0\,f_0.$$

Also ist $P(n) = f_0$ eine beliebige Konstante, und es gilt nach (4.5)

$$S(n) = Z\,\frac{Q(n)}{R(n)}\,\frac{P(n+1)}{P(n)} = -4\frac{1}{1}\cdot\frac{f_0}{f_0} = -4$$

und somit

$$\begin{aligned} A(n+1) &= (-4)\,A(n) \quad \Longrightarrow \\ A(n) &= (-4)^n A(0) = (-4)^n. \end{aligned} \tag{4.28}$$

□

Kapitel 5

Ausblick

Mit den vorgestellten Algorithmen haben wir nun ein mächtiges Instrumentarium für algorithmische Summation. Der Gosperalgorithmus ist dabei — obwohl er in einigen Fällen geschlossene Formen liefert — hauptsächlich als Einbettung in den Zeilbergeralgorithmus interessant.
Der Zeilbergeralgorithmus bietet sicherlich die meisten Anwendungsmöglichkeiten von allen drei Algorithmen. Es gibt nur wenige Beispiele, bei denen er eine Rekursion höherer Ordnung liefert, obwohl es eine der Ordnung eins gibt. Daher sind die Anwendungen des Petkovšekalgorithmus als erfolgreicher Anschluß beim Scheitern des Zeilbergeralgorithmus eher selten.
Der Petkovšekalgorithmus schließt vor allem eine theoretische Lücke. Wenn wir mit ihm keine geschlossene Form finden, gibt es keine. Er läßt sich also gut einsetzen um Vermutungen zu widerlegen.
Das vorgestellte Instrumentarium ist zwar mächtig, aber durchaus noch ausbaufähig. Das wird an einigen neueren Facharbeiten deutlich, die hier zum Abschluß kurz vorgestellt wurden.

1. Im Abschnitt 1.1 werden Analogien zwischen der stetigen Integration und dem Finden einer diskreten Stammfunktion $s(k)$ mit Hilfe des Gosperalgorithmus behandelt.
   Ebenso wie die Stammfunktion $s(k)$ mit (1.4)

   $$s(b) - s(a) = \sum_{k=a+1}^{b} a(k)$$

   in Analogie zur stetigen Stammfunktion $F(x)$ mit

   $$F(b) - F(a) = \int_a^b f(x)\,dx$$

   gesehen werden kann, stellt die geschlossene Form (3.1)

   $$A(n) = \sum_k F(n,k)$$

eine Analogie zum bestimmten Integral

$$F(y) := \int_{-\infty}^{\infty} f(x,y)\ dx \tag{5.1}$$

dar.
Die **stetige Version des Zeilbergeralgorithmus** von Almkvist und Zeilberger [3] kann einige Integrale der Form (5.1) lösen. Eine genauere Beschreibung des Verfahrens kann in [4] nachgelesen werden.

2. Wilf und Zeilberger [16] präsentieren mit der **Multisummenerweiterung des Zeilbergeralgorithmus** eine Verallgemeinerung der diskreten **und** der stetigen Version des Zeilbergeralgorithmus. Das von den Autoren beschriebene Verfahren findet (im Prinzip) geschlossene Formen für Summen $S(n)$ der Form

$$S(n) := \int_{a_1} \int_{a_2} \cdots \int_{a_p} \sum_{x_1} \sum_{x_2} \cdots \sum_{x_q} F(a_1, a_2, \ldots, a_p, x_1, x_2, \ldots, x_q) d(x_1, x_2, \ldots, x_p)$$

mit festen $p, q \in \mathbb{N}_0$.
Allerdings haben die Autoren bisher[1] noch keine ausgereifte Implementierung vorgelegt, und der Algorithmus hat keine gute Komplexität, das heißt er ist im allgemeinen sehr langsam und hat einen immensen Speicherbedarf.

3. Wolfram Koepf [8] erweitert den Zeilbergeralgorithmus auf sogenannte (m,l)-fach hypergeometrische Eingaben $F(n,k)$, das heißt Eingaben für die es feste natürliche m und l gibt, so daß

$$\frac{F(n,k)}{F(n-m,k)},\ \frac{F(n,k)}{F(n,k-l)} \in \mathbb{K}(n,k) \tag{5.2}$$

gilt.
Koepf findet die geeigneten m und l, so daß (5.2) gilt, und kann so den Zeilbergeralgorithmus auf

$$\tilde{F}(n,k) := F(m\,n, l\,k)$$

anwenden.
Findet er eine geschlossene Form für $\tilde{F}(n,k)$, so gilt diese auch für $F(n,k)$ mit $n \bmod m = 0$. Für die Fälle $n \bmod m = 1, 2, \ldots, n-1$ müssen allerdings gesondert geschlossene Formen mit den entsprechenden Anfangswerten berechnet werden. So ergeben sich dann Lösungen wie

$${}_2\mathrm{F}_1\left(\begin{matrix} -n & n/2+1 \\ 4/3 & \end{matrix} \,\middle|\, \frac{8}{9} \right) = \begin{cases} 0 & \text{falls } n \bmod 2 = 0 \\ \frac{(1/2)_{n/2}\,(3/2)_{n/2}}{(5/6)_{n/2}\,(7/6)_{n/2}}\,(-3)^{-(n/2)} & \text{falls } n \bmod 2 = 1. \end{cases} \tag{5.3}$$

Viele Gleichungen dieser Form wurden von Gessel und Stanton [5] aufgestellt, und konnten mit Hilfe Koepfs **erweitertem Zeilbergeralgorithmus** erstmals bewiesen werden.

[1] Stand Januar 1996

4. Der rechnerisch aufwendigste Teil bei allen drei Algorithmen ist das Lösen einer Rekursion durch ein Polynom (Gosper Schritt 4, Zeilberger Schritt 2c, `poly` 3b). Inbesondere Petkovšeks `hyper` ist sehr langsam, weil er `poly` exponentiell (bezüglich des Grades der Eingabepolynome) oft aufruft.
   Ungünstig wirkt sich dabei aus, daß die von den in Computeralgebrasystemen benutzten `solve`-Funktionen alle Polynomgleichungen nach der Methode der unbekannten Koeffizienten lösen. Die Rekursionsgleichung wie ein Polynom behandelt, und es geht Information über die Beschaffenheit der Rekursion verloren. Vor allem ist die Anzahl der zu lösenden Unbekannten gleich dem Maximalgrad des zu bestimmenden Polynoms. Dabei ist die Dimension des Lösungsraums häufig nicht viel höher als die Ordnung der Rekursion. So kann beim Gosperalgorithmus ein Maximalgrad $d = 100$ für das Polynom $f(k)$ auftreten, der Grad des Lösungsraums nicht höher als eins sein kann.
   Abramov, Bronstein, und Petkovšek [1] geben verbesserte Algorithmen zur Berechnung von homogenen und inhomogenen polynomiellen Rekursionen.

Literaturverzeichnis

[1] Abramov, S.A., Bronstein, M. und Petkovšek, M.: *On polynomial solutions of linear operator equations*, Proceedings of ISSAC'95, ACM, 1995, 290-296.

[2] Aigner, M.: *Diskrete Mathematik*, Vieweg, Braunschweig, 1993.

[3] Almkvist, G. und Zeilberger, D.: *The method of differentiating under the integral sign*, J. Symb. Comp. **10**, 1990, 571–591.

[4] Cartier, P.: *Démonstration "automatique" d'identités et fonctions hypergéometriques [d'après D.Zeilberger]*, Astérisque **206** (1992), 41–91.

[5] Gessel, I. und Stanton, D.: *Strange evaluations of hypergeometric series*, SIAM J. of Math. Anal. **13**(1982), 295–308.

[6] Gosper, R. W., Jr.: *Descision procedure for indefinite hypergeometric summation*, Proc. Nat. Acad. Sci. USA **75** (1978), 40–42.

[7] Graham, R. L., Knuth, D. E. und Patashnik, O.: *Concrete Mathematics. A Foundation for Computer Science*. Addison-Wesley, Reading, Massachussets, second edition 1994.

[8] Koepf, W.: *Algorithms for indefinite and definite summation*, Preprint, ZiB-Berlin, TR 94-33.

[9] Koepf, W.: REDUCE *Package for Indefinite and Definite Summation*,ZiB-Berlin, TR 94-9.

[10] Koornwinder, T.H.: *On Zeilberger's algorithm and its q-analogue: a rigorous description.*, J. of Comput. and Appl. Math. **48**, 1993, 91–111

[11] Man, Y.-K. und Wright, F. J.: *Fast polynomial dispersion computation and its application to indefinite summation*, Proceedings of ISSAC'94, ACM, 1994, 175-180.

[12] Mignotte, M.: *Mathematics for Computer Algebra*, New York 1992.

[13] Petkovšek, M.: *Hypergeometric Solutions of Linear Recurrences with Polynomial Coefficients*, J. Symb. Comp. **14**, 1992, 243–264.

[14] Rainville, E.D.: *Special Functions*, The Macmillan Company, New Yorck 1960.

[15] Tricomi, G.: *Vorlesungen über Orthogonalreihen*, Springer, Berlin 1970.

[16] Wilf, H. S. und Zeilberger, D.: *An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities*, Invent. Math. **108** (1992), 575–633.

[17] Zeilberger D.: *A holonomic systems approach to special functions identities*, J. Comput. Appl. Math. **32** (1990), 321–368.

[18] Zeilberger D.: *A fast algorithm for proving terminating hypergeometric identities*, Discrete Math. **80** (1990), 207–211.

[19] Zeilberger D.: *The method of creative telescoping*, J. Symb. Comput. **11** (1991), 195–204.

[20] Zippel, R.: *Effective polynomial computation*, Kluwer Academic Publishers, Norwell, Massachussetts USA (1993)