Selektivitätsabschätzung und Kostenmodelle

Ein Überblick über grundlegende Werkzeuge der Anfrageverarbeitung in relationalen Datenbanksystemen

Max Bechtold

Technische Universität Kaiserslautern
Lehrgebiet Informationssysteme

Zusammenfassung Die effiziente Ausführung von Anfragen ist das zentrale Maß für die Güte eines Datenbanksystems. Entscheidende Mittel zur Planung der Durchführung von Anfragen sind dabei die Selektivitätsabschätzung relationaler Operatoren und deren Kostenmodelle. Damit können Größen von Zwischen- und Endergebnissen verschiedener Varianten der Ausführung einer Anfrage vorberechnet und eine günstige ausgewählt werden. Neben weiteren Methoden werden *Histogramme* als herausragender Vertreter der Hilfsmittel zur Selektivitätsabschätzung vorgestellt. Einer einführenden Definition folgen Erläuterung und Klassifizierung einiger Arten von Histogrammen, Möglichkeiten der Erzeugung und Wartung und ein Ausblick auf mehrdimensionale Histogramme und spezielle Varianten. Kostenmodelle verwenden diese Abschätzungen um günstige Wege zur Beantwortung einer Anfrage zu finden. Der zweite Teil dieser Arbeit befasst sich mit Modellen für verschiedene Operatoren und geht dabei auch auf physikalische Aspekte der Modellierung ein.

1 Einleitung

Die kostengünstige Planung und Ausführung von Anfragen ist schon seit Jahrzehnten ein zentrales Thema in der Entwicklung von Datenbankverwaltungssystemen. Ohne Algorithmen zur Bestimmung einer möglichst effizienten Vorgehensweise kann ein Datenbanksystem die hohen Anforderungen im Mehrbenutzerbetrieb nicht erfüllen. Für die Güte eines solchen Systems ist die schnelle und günstige Anfrageverarbeitung also ein wichtiges Merkmal.

Die Ausführung einer Anfrage verläuft traditionell in mehreren Phasen. Am Anfang steht ein Anfragegraph, der die einzelnen Operatoren einer Anfrage ordnet und zusammen mit den zu Grunde liegenden Relationen darstellt. Nach syntaktischer und semantischer Prüfung wird der Anfragegraph einer Zugriffs- und Integritätskontrolle unterzogen. Besteht er diese, ist die Anfrage gültig und erlaubt und wird zum Optimierer weitergeleitet. Dieser wendet verschiedene Regeln und Umformungen an um die Anfrage zu vereinfachen und in eine standardisierte Form zu bringen. Danach werden die logischen Operatoren auf physische Operatoren abgebildet, wofür es generell mehrere Möglichkeiten gibt, deren Kombinationen in einer oft sehr großen Menge von möglichen Ausführungsplänen resultieren. Der Optimierer hat nun die Aufgabe, aus diesem Suchraum

einen günstigen Plan auszuwählen. Dazu benötigt er zum einen eine Methode, dem Suchraum effizient zu durchlaufen, was selbst ein sehr komplexes Thema und daher nicht im Fokus dieser Arbeit ist. Zum anderen muss er Pläne qualitativ vergleichen, das heißt, er muss ihre Kosten kennen. Diese ergeben sich aus Faktoren wie CPU-Zeit, E/A-Zugriffszeit usw. und werden in Abschnitt 3.3 erläutert. Sie zu berechnen ist Aufgabe der Kostenmodellierung, die, ausgehend von den Operatoren, aus denen sich ein Anfrageplan zusammensetzt und deren Selektivität Ausführungsplänen Kosten zuordnet [1].

Relationale Operatoren transformieren eine Eingabe (eine Relation oder ein Zwischenergebnis) in eine Ausgabe (ein Zwischen- oder Endergebnis). Die Größe der Eingabe ist dabei bekannt oder zumindest geschätzt, die Größe der Ausgabe nicht. Sie kann allerdings berechnet werden, in dem die Selektivität eines Operators mit der Eingabegröße multipliziert wird. Da die Selektivität meistens nur auf Basis einer Schätzung vorliegt, können die hier betrachteten Kostenmodelle für Operatoren im Allgemeinen keine exakten Aussagen treffen. Dennoch bieten sie brauchbare Unterstützung bei der Auswahl eines günstigen Ausführungsplans.

Ohne Wissen über die Verteilung eines Attributs einer Relation müssen einfache und oftmals fatale Annahmen darüber getroffen werden. Üblicherweise nimmt man dann eine Gleichverteilung an, wie zum Beispiel in *System R* [1]. Zwar ist diese Annahme besser als eine Schätzung durch einen Zufallsgenerator bestimmen zu lassen und liefert oft sinnvolle, wenn auch ungenaue Werte. Allerdings ist dieser primitive Ansatz nur der Ausgangspunkt für deutlich ambitioniertere und trotzdem effiziente Methoden. Einige davon sollen in dieser Arbeit vorgestellt und charakterisiert werden.

Verschiedene Operatoren haben im Allgemeinen unterschiedliche Auswirkung auf die Größe ihrer Ausgabe (in Form einer Relation oder eines Zwischenergebnisses). Daher weist man ihnen Kostenmodelle zu, die die Selektivitätsabschätzung nutzen um Ausgabegrößen zu quantifizieren. Vergleicht man verschiedene Ausführungspläne, zieht man solche mit kleineren Zwischenergebnissen anderen vor, weil sie zwar zum selben Ziel führen, insgesamt aber geringere Kosten verursachen.

Diese Arbeit ist wie folgt strukturiert: Nach der Einleitung werden Methoden der Selektivitätsabschätzung vorgestellt und in Klassen eingeteilt. Als einer der wichtigsten Vertreter dieser Methoden sollen Histogramme betrachtet werden, wobei chronologisch vorgegangen wird. Mit Kostenmodellen beschäftigt sich der zweite Teil dieser Arbeit. Nach der Präsentation von Kostenmodellen für die wichtigsten relationalen Operatoren sollen auch die physikalischen Aspekte beleuchtet werden.

2 Methoden der Selektivitätsabschätzung

Das Wissen über die Verteilung eines Attributs einer Relation (oder zumindest eine gute Schätzung davon) wird, wie bereits erwähnt, benutzt um Selektivitäten abzuschätzen und damit Kostenmodelle zu erstellen. Die *Selektivität* S eines Operators ist eine reelle Zahl $0 \leq S \leq 1$, die den Anteil der Ausgabe an der Ein-

gabe des Operators angibt, zumeist in der Anzahl der Tupel. Sie ist ein nützliches Maß bei der Größenabschätzung von Zwischenergebnissen, da die Kardinalität von Relationen (als Ausgangspunkt eines Operators oder Operatorsequenz) üblicherweise in der Statistik eines Datenbanksystems vorliegt [1].

Doch auch in anderen Aspekten von Datenbankverwaltungssystemen findet man Verwendung für Verteilungsannäherungen und Selektivitätsabschätzungen. So dienen diese zum Beispiel zur Beanwortung von approximativen Anfragen, die oft aus statistischem Interesse gestellt werden. Sie können auch als Information für den Benutzer vor beziehungsweise während der Bearbeitung einer Anfrage dienen, in dem beispielsweise die erwartete Anzahl der Ergebnistupel angezeigt wird. Damit kann dieser schnell erkennen, wenn ihm ein Fehler beim Stellen der Anfrage unterlaufen ist und diese neu formulieren. In verteilten Systemen helfen sie beim Lastausgleich von parallelen Verbünden. [2,3]

Um die Bedeutung einer guten Schätzung der Selektivität zu verdeutlichen soll hier ein Beispiel eingeführt werden, auf das im Rest dieses Kapitels immer wieder Bezug genommen werden wird um die Güte einer Abschätzung zu demonstrieren. Es sei das Attribut *Dienstjahre* einer Relation *Personal* gegeben. Die Verteilung wird in realen System meist ähnlich sein und Abb. 1 entsprechen. Die Relation besteht aus 420 Tupel, deren Wert für Dienstjahre zwischen 0 und 49 liegt. Der Nullwert ist für Dienstjahre verboten, d.h. jedes Tupel hat einen Wert für dieses Attribut.

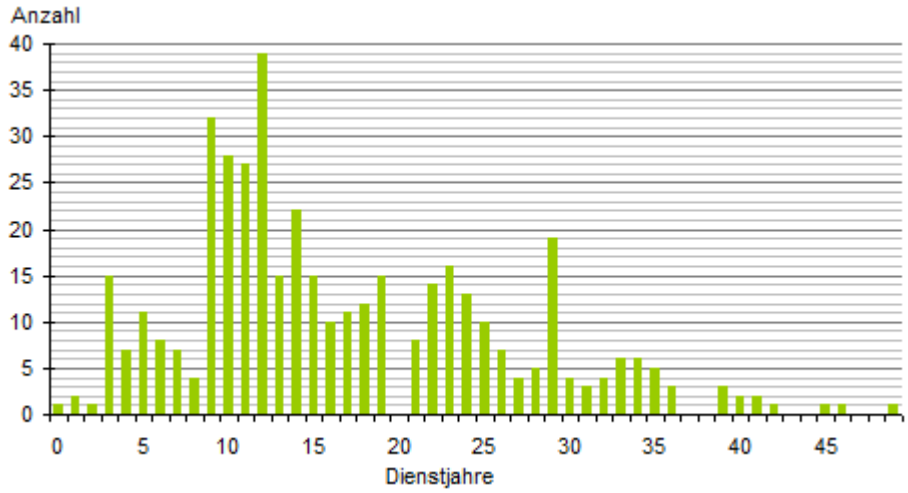

Abbildung 1. Verteilung des Attributs Dienstjahre

Um eine gegebene Annäherung zu testen, wird die folgende SQL-Abfrage benutzt, die 63 Tupel als Ergebnis liefert:

```
SELECT *
FROM Personal
WHERE Dienstjahre >= 15 AND Dienstjahre < 20
```

Zwar ist das Resultat einer einzigen Anfrage nicht repräsentativ und die Fehler der einzelnen Schätzmethoden werden auch nicht immer ihre tatsächliche Güte widerspiegeln. Trotzdem wird dieses Beispiel die verschiedenen Qualitäten und Probleme herausstellen können.

2.1 Einteilung

Obwohl Histogramme am weitesten verbreitet sind und auch hier im Besonderen betrachtet werden, gibt es noch andere nichtparametrische Methoden um die Ergebnisse von Anfragen abzuschätzen. So ist auch *Sampling*, also das Schließen auf die Häufigkeitsverteilung des gesamten Attributs anhand einiger Stichproben, ein möglicher Weg um Aussagen über Selektivität zu machen. Von anderer Natur sind parametrische Methoden, die allerdings nicht immer sinnvoll anwendbar sind.

Dieser Teil der Arbeit dient zur Abgrenzung der beiden Hauptklassen der Selektivitätsabschätzung, für die auch Beispiele genannt werden. Als Vertreter der zweiten Klasse werden Histogramme vorstellt und genau beleuchtet.

Parametrische Methoden. *Parametrische Methoden* basieren, wie der Name andeutet, auf Parametern, mit denen die Verteilung einer Variable beziehungsweise eines Attributs angenähert wird. Sie gehen dabei von der Annahme aus, dass die Art der Verteilung im Wesentlichen bekannt ist und nur durch ein paar „Stellknöpfe“ angepasst werden muss. Chen et al. [4] bezeichnen die Verteilungsart als *Modellfunktion*, die über einige Parameter konfiguriert wird. Ein prominentes Beispiel für eine solche Modellfunktion ist die Normalverteilung, der viele natürliche Gegebenheiten folgen, wie zum Beispiel die Größe von Studenten einer Universität. Darunter gibt es einige sehr kleine sowie ein paar sehr große Personen, die meisten aber sind in etwa so groß wie der Durchschnitt. Daraus ergibt sich die typische Glockenform einer Normalverteilung. Neben dieser begegnet man in Datenbanksystemen auch anderen Verteilungen, darunter die Gleichverteilung (z.B. Ergebnisse beim Würfeln) oder die Zipf-Verteilung (z.B. Nachnamen in einem Telefonbuch).

Ist über ein Attribut bekannt, dass seine Werte im Wesentlichen einer Normalverteilung unterliegen, kann man diese über zwei Parameter justieren. Zum einen über den Erwartungswert μ, der den Scheitelpunkt der Kurve der Verteilung definiert. Zum anderen über die Standardabweichung σ, die die „Breite“ der Glocke steuert: Im Bereich $[\mu-\sigma, \mu+\sigma]$ liegen etwa 68 % aller Werte, etwa 95 % fallen in $[\mu-2\sigma, \mu+2\sigma]$ und annähernd 100 % befinden sich in einem Radius von bis zu 3σ um den Mittelwert μ [5].

Eine Verteilungsannäherung des Attributs Dienstjahre aus obigem Beispiel ist denkbar, auch wenn es einige stark abweichende Werte gibt. In der ers-

ten Hälfte des Wertebereichs kann man grob den Scheitelpunkt einer Glockenform ausmachen. Für Mittelwert und Standardabweichung dieser Verteilung gilt $\mu = 16,77$ und $\sigma = 9,3$. Berechnet man mit diesen Parametern die Normalverteilung für Dienstjahre, erhält man eine Kurve wie in Abb. 2. Damit würde der Anfrageoptimierer die Selektivität der obigen Anfrage auf 21,2 % schätzen und 89 Tupel als Ergebnis erwarten.[1] Das ist ein relativer Fehler von 41,2 %, womit die parametrische Methode hier kein gutes Ergebnis liefert.

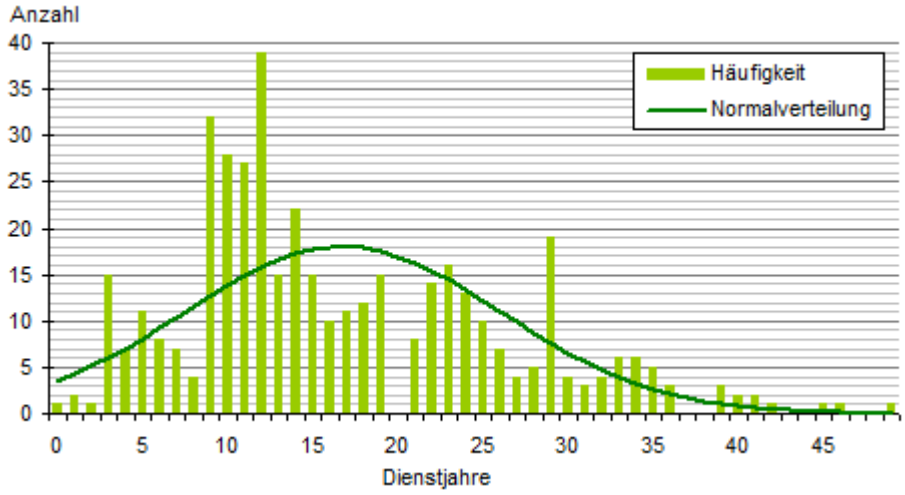

Abbildung 2. Annäherung von Dienstjahre durch Normalverteilung

Für die Bestimmung der Parameter wird zum Beispiel Sampling benutzt. Das ist ausreichend, vor allem weil die tatsächliche Verteilung wegen natürlicher Schwankungen durch eine einfache Funktion gar nicht ausdrückbar ist und die entstehende Kurve nie die genauen Werte treffen wird. Eine schnell zu berechnende Verteilungsfunktion aber ist Voraussetzung für die schnelle Selektivitätsabschätzung im laufenden Betrieb.

In den Fällen, in denen sich die Verteilung gut durch eine Modellfunktion beschreiben lässt, übertreffen parametrische Methoden ihre nichtparametrischen Pendants in punkto Speicheraufwand und auch Genauigkeit. Ersteres, weil auch für beliebig große Relationen im Wesentlichen nur eine Funktion mit wenigen Parametern gespeichert werden muss. Letzteres gilt, weil selbst für beliebig kleine Wertbereiche Informationen vorliegen (der Verlauf der Kurve), während bei parametrischen Methoden gröbere Informationen auf diese herunter gebrochen werden müssen (z.B. wird bei einem *Histogramm* aus der Information über einen umfassenden Bereich auf ein gefragtes Werteintervall geschlossen) und damit ein

[1] Zu beachten ist, dass dieser Anteil nicht von 420 genommen wird, sondern von 407 — der kumulierten Häufigkeit der Normalverteilung im Intervall [0, 49]

weiterer Ungenauigkeitsfaktor eingeführt wird. Zudem haben diese mehr Power[2] als nichtparametrische Methoden, da sie im Gegensatz zu diesen durch die Modellfunktion ein grundlegendes Wissen über die Verteilung haben [6].

Oft lässt sich die Verteilung eines Attributs aber nicht oder nur sehr schlecht durch eine Modellfunktion annähern. Dann würden parametrische Methoden zu schlechte Vorhersagen treffen, als dass sie sinnvoll zur Selektivitätsabschätzung herangezogen werden könnten, und man sollte stattdessen eine nichtparametrische Methode verwenden. Ein weiterer Nachteil ist, dass die Art der Verteilung im Voraus bekannt sein muss, was bei einer Relation, die gerade im Entstehen ist, nicht unbedingt der Fall ist [4].

Nichtparametrische Methoden. Im Gegensatz zu parametrischen Methoden wird bei *nichtparametrischen Methoden* im Allgemeinen keine Annahme über die Verteilung eines Attributs getroffen. Sie benötigen kein Vorwissen, daher finden sie auch Einsatz, wenn eingangs nichts über eine Verteilung bekannt ist.

Die prominentesten Vertreter der nichtparametrischen Methoden sind Histogramme, bei denen der Wertebereich in verschiedene Intervalle aufgeteilt wird und Verteilungsinformationen für jedes dieser Intervall vorliegen. Damit werden dann Selektivitäten geschätzt. Eine breite Diskussion von Histogrammen folgt im anschließenden Abschnitt.

Bevor Histogramme Einzug in Datenbanksysteme hielten benutzte man in *System R* eine ähnliche, aber einfachere Methode zur Selektivitätsabschätzung. Diese beruht im Wesentlichen auf der Anzahl unterschiedlicher Werte N_A eines Attributs A einer Relation R und geht von einer Gleichverteilung derselben sowie der Unabhängigkeit der Werte verschiedener Attribute aus. Wegen diesen Annahmen reicht N_A aus als grundlegender Faktor zur Berechnung von Selektivitäten verschiedener boolscher Ausdrücke. Diese teilen Mannino et al. [8] in verschiedene Klassen ein, von denen die wichtigsten sind:

1. Relational
   (a) Einfach
   (b) Parametrisch
   (c) Dyadisch
2. Bereich

Einfache relationale Ausdrücke sind von der Form *<Attribut> <Vergleichsoperator> <Wert>*, z.B. *Alter = 25*. Andere Ungleichheitszeichen wie „≤“ oder „>“ an Stelle der Gleichheit sind möglich, kommen aber weniger häufig vor. Parametrische relationale Ausdrücke haben dieselbe Form, allerdings mit einem erst zur Laufzeit instantiierten Parameter anstelle eines konkreten Werts auf der einen Seite des Operators.[3] Dyadische relationale Ausdrücke folgen dem Muster *<Attribut> <Vergleichsoperator> <Attribut>* und können sich auch auf

[2] Power bezeichnet in der Statistik die Aussagekraft einer Methode

[3] Sie sollten nicht verwechselt werden mit parametrischen Methoden zur Selektivitätsabschätzung

verschiedene Attribute einer Relation beziehen. Zu dieser Klasse gehört z.B. *Gehalt < Bonus*. Bereichsausdrücke folgen dem Schema *<Attribut> in [<Wert1>, <Wert2>]* und sind besonders gut mit Histogrammen abzuschätzen. Ein Beispiel ist *Alter in [20, 25]*.

Die Selektivität eines einfachen relationalen Ausdrucks $A = v_i$ wird in *System R* abgeschätzt mit

$$S(A = v_i) = \frac{1}{N_A},$$

also mit dem Kehrwert der Anzahl unterschiedlicher Attributwerte. Ist ein Attribut tatsächlich gleichverteilt, entspricht S genau dem Anteil der Tupel der Relation mit dem A-Wert v_i. Liegt N_A nicht in der DBS-Statistik vor, wählt man per Hand einen empirisch bestimmten Wert wie z.B. 10, d.h. $S = 1/10$.

Etwas komplizierter wird diese Formel, wenn man Verhältniszeichen wie „$<$“ oder „$>$“ verwendet. Dann hängt die Selektivität von der Lage von v_i im Wertebereich ab, wofür man die Werte v_{min} und v_{max}, also den minimalen und maximalen vorkommenden Wert in A, benötigt. Damit berechnet sich die Selektivität als

$$S(A < v_i) = \frac{v_i - v_{min}}{v_{max} - v_{min}} \text{ bzw. } S(A > v_i) = \frac{v_{max} - v_i}{v_{max} - v_{min}}.$$

Das ist sinnvoll, denn unter der Annahme der Gleichverteilung entspricht dieser Anteil S des selektierten Bereichs am Wertebereich von A dem Anteil der diese Bedingung erfüllenden Tupeln an der Kardinalität der Relation. $S = 1/3$ ist der empirische Standardwert, wenn keine ausreichende Statistik geführt wird. Für „$\leq$“ und „$\geq$“ wird dieselbe Formel verwendet, da der Einbezug eines zusätzlichen Werts bei einer Schätzung nicht ins Gewicht fällt.

Für parametrische Ausdrücke gelten, soweit anwendbar, ebenfalls diese Formeln. Allerdings ist dort v_i nicht bekannt, sodass man bei Ausdrücken mit Verhältniszeichen nur auf den Standardwert zurückgreifen kann.

Die Selektivität dyadischer Ausdrücke mit einem „$=$“ als Vergleichsoperator schätzt man mittels

$$S(A_i = A_k) = \frac{1}{\max\{N_{A_i}, N_{A_k}\}}.$$

Liefert die Statistik nur N_{A_i} oder N_{A_k}, bilden diese jeweils alleine den Nenner und die Formel entspricht der eines einfachen relationalen Ausdrucks. Für „$<$“ und „$\leq$“ bzw. „$>$“ und „$\geq$“ schätzt man die Selektivität durch

$$S(A_i > A_k) = \frac{1}{N_{A_i}} \text{ bzw. } S(A_i < A_k) = \frac{1}{N_{A_k}}.$$

Als Standardwert dient in allen Fällen $S = 1/10$.

Bereichsausdrücke lassen sich auf Verhältnisausdrücke zurückführen. *Alter in [20, 25]* beispielsweise entspricht mathematisch gesehen $20 \leq Alter \leq 25$ und die Selektivität lässt sich mit der Formel

$$S(A\ in\ [v_i, v_j]) = \frac{v_j - v_i}{v_{max} - v_{min}}$$

und dem Standardwert $S = 1/2$ abschätzen [1,9].

Die Annahme der Gleichverteilung und die Verwendung von N_A ist natürlich eine idealisierte Voraussetzung, was die Forderung nach realitätsnäheren und dennoch aufwandsarm zu berechnenden Faktoren rechtfertigt. Piatetsky-Shapiro et al. schlugen daher statt $1/N_A$ die *Attributdichte* als Maß für Selektivität vor. Diese berechnet sich nach

$$\frac{\sum_{i=1}^{N_A} N_i^2}{N_R^2},$$

wobei N_i für die Anzahl der Tupel mit dem Attributwert v_i steht und N_R die Kardinalität der Relation R ist. Die Bedeutung dieses Quotienten erschließt sich schnell, wenn man ihn für $N_A = 1$ und $N_A = N_R$ betrachtet, wenn also nur ein Attributwert vorkommt oder wenn jedes Tupel einen anderen Wert für A hat. Im ersten Fall gibt es nur einen Attributwert v_1 mit der Häufigkeit $N_1 = N_R$ und die Attributdichte ist 1. Im zweiten Fall gibt es N_A verschiedene, nur einmal vorkommende v_i, d.h. $N_i = 1$, und man erhält für die Attributdichte $N_R/N_R^2 = 1/N_R$. In Worte gefasst ist die Dichte eines Attributs der durchschnittliche Anteil der Relation mit demselben Wert dieses Attributs. Im Beispiel Dienstjahre sind das 4,3 % im Gegensatz zu 2 %, die sich aus $1/N_A$ ergeben. Piatetsky-Shapiro et al. sehen die Attributdichte als einen besseren Ansatz Verteilungen anzunähern, da für ungleichmäßig verteilte Attribute sinnvollere Selektivitäten geschätzt werden können als mit dem Faktor $1/N_A$ [9].

Um auch diese Methode zu untersuchen soll wieder das Beispiel von weiter oben dienen. Das Prädikat der Anfrage ist ein Bereichsausdruck *Dienstjahre in [15, 19]* und selektiert bei tatsächlicher Ausführung 63 Tupel. Die *System-R*-Formel dagegen würde $S = (19 - 15)/(49 - 0) \approx 1/12$ von 420 Tupel, also 35 Tupel erwarten und damit 44,4 % zu wenig.

Wie eingangs erwähnt, machen nichtparametrische Methoden keine Annahme über die Verteilung eines Attributs und sind deswegen öfter anwendbar als parametrische Methoden. Weil sie im Gegensatz zu diesen auch nicht von einer Modellfunktion ausgehen, sind sie im Allgemeinen robuster als parametrische Methoden. Auch kommt es vor, dass man sie parametrischen Methoden wegen ihrer Einfachheit vorzieht, obwohl man damit möglicherweise Genauigkeit einbüßt. Da sie kein Vorwissen benötigen, sind sie bestens geeignet für „junge“ Relationen, d.h. solche, die am Beginn ihres Lebenszyklus stehen und noch wenige Tupel halten, denn dort kann sich die Art der Verteilung schnell, oft und drastisch ändern. Eine bei einer parametrischen Methode gewählte Modellfunktion kann anfangs gute Ergebnisse liefern, sich dann aber beim Wachsen der Relation als falsch herausstellen, während nichtparametrische Methoden sich einfach anpassen.

Durch ihre generische Natur haben nichtparametrische Methoden zumeist weniger statistische Aussagekraft als parametrische Methoden. Dies bedeutet für Methoden wie beispielsweise Sampling, dass größere Stichproben erstellt werden müssen, um eine „sichere“ Information über die Verteilung eines Attributs zu erhalten, als z.B. beim Verwenden einer Modellfunktion. Bei Histogrammen

kann eine sehr große Zahl von Intervallen notwendig sein, um eine Aussagekraft ähnlich der von parametrischen Methoden zu erhalten [6].

2.2 Histogramme als nichtparametrische Methode

Histogramme sind keine Erfindung der Informatik. Es gibt sie und ähnlich aufgebaute Diagramme schon seit dem 19. Jahrhundert, in welchem der Begriff „Histogramm“ erstmals von dem britischen Mathematiker Karl Pearson geprägt wurde für eine „gebräuchliche Form der grafischen Repräsentation, in dem Säulen die Frequenz ihrer Basisbereiche darstellen“. Doch schon 1781 benutzte der schottische Volkswirt William Playfair ein Balkendiagramm, in welchem er die Im- und Exporte Schottlands aus und nach anderen Ländern vergleicht. Dieses Diagramm gilt als das erste dieser Art überhaupt und darf als Urahn aller Histogramme angesehen werden. Das Wort Histogramm selbst setzt sich aus den griechischen Wörtern „istos“, das sich zu „Mastbaum“ übersetzt, und „gramma“, zu deutsch „Geschriebenes“, zusammen, ist aber kein echtes Wort dieser Sprache [2]. Die Bedeutung destilliert der Duden auf die „grafische Darstellung einer Häufigkeitsverteilung“.

Mit Histogrammen wird also die Häufigkeitsverteilung einer bestimmten Variablen visualisiert. Da dies eine allgemeine Funktion ist, finden Histogramme Anwendung in vielen Bereichen der Informatik. Beispielsweise werden mit ihnen Verteilungen des Farbspektrums von Bildern repräsentiert, in dem einer bestimmten Farbe die Anzahl der Pixel mit dieser Farbe zugeordnet wird. Damit kann bei der Kompression eine bessere Qualität erreicht werden. In Datenbanksystemen wird neben dem Haupteinsatz in der Selektivitätsabschätzung die Lastverteilung von parallelen Verbünden optimiert. Dort ist nicht nur die Größe der Ausgabe eines Operators interessant, sondern auch die Verteilung der Ausgabe, wofür Histogramme verwendet werden [2].

Definition. Vor einer Definition von Histogrammen müssen zunächst einige notwendige Begriffe eingeführt werden. Der *Definitionsbereich* D eines Attributs A einer betrachteten Relation R ist die Menge aller möglichen Werte, die dieses Attribut annehmen kann. Der *Wertebereich* V beschreibt die Menge der tatsächlich angenommenen Attributewerte, d.h. $V \subseteq D$, wobei $D \setminus V$ nichtleer sein kann. V ist endlich, da es sich um eine physisch vorhandene Menge von Werten in einer Relation handelt. Deswegen lässt sich eine Aufzählung für V angeben, d.h. $V = \{v_i \mid 1 \leq i \leq N_A\}$, wobei $v_i < v_j$ für $i < j$. Weiterhin sei *Ausdehnung* s_i eines Werts v_i definiert als $s_i = v_{i+1} - v_i$ für $1 \leq i < N_A$ mit $s_0 = v_1$ und $s_{N_A} = 1$. Je dichter Werte also beieinander liegen, desto geringer ist jeweils ihre Ausdehnung. Diese kann dazu genutzt werden um Werte je nach ihrer Ausdehnung zu gewichten.

Die *Häufigkeit* oder *Frequenz* f_i von v_i ist gleich der Anzahl der Tupel $t \in R$, für die $t.A = v_i$ ist. Die *kumulative Häufigkeit* c_i entspricht der Anzahl der Tupel t mit $t.A \leq v_i$, d.h. $c_i = \sum_{k=1}^{i} f_k$. Damit kann man die *Verteilung* des Attributs A konkretisieren als eine Menge von Paaren $T = \{(v_i, f_i) \mid 1 \leq i \leq N_A\}$, in der jedem Attributwert aus V seine Häufigkeit zugeordnet ist.

Ein Histogramm entsteht nun durch Aufteilen von T in $\beta \geq 1$ paarweise disjunkte Partitionen, die *Intervalle* genannt werden. Diese werden so gewählt, dass eine möglichst genaue Annäherung an die tatsächliche Verteilung erreicht wird. Im pathologischen Fall $\beta = N_A$ ist die Approximation perfekt, dann allerdings wird jeder Wert einzeln gespeichert, was nicht sehr effizient ist. Dahingegen entspricht $\beta = 1$ der *System-R*-Methode, wenn innerhalb des Intervalls von einer Gleichverteilung der Werte und ihrer Häufigkeit ausgegangen wird. Darum wird diese Methode auch vereinzelt als *triviales* Histogramm bezeichnet. Neben der Anzahl der Intervalle gibt es noch einige weitere Dimensionen, die der im nächsten Abschnitt folgenden Klassifizierung zu Grunde liegen.

Jedem der entstandenen Intervalle ordnet man nun die Summe der Häufigkeiten der in ihnen enthaltenen Werte zu. Diese Zuordnung kann man sich als Tabelle vorstellen. In aufwändigeren Histogrammtypen werden unter Umständen noch zusätzliche Werte wie beispielsweise die Häufigkeiten der Intervallrandwerte gespeichert, wenn man nicht von einer Gleichverteilung derselben ausgehen will. Diese Sonderfälle werden an späterer Stelle erläutert, für den Moment soll das Verständnis eines Histogramms als Zuordnung *Intervall* $\rightarrow$ *Häufigkeitssumme der Intervallwerte* genügen. Neben der Repräsentation als Tabelle werden Histogramme häufig auch grafisch dargestellt. Dabei entstehen Diagramme in der Art von Abb. 3.

Einfache relationale Anfragen und Bereichsanfragen sind nichts anderes als ein Fenster $[a, b] \subseteq D$, wobei im ersten Fall $a = b$ ist. Die Anzahl an Tupeln, die damit selektiert wird, entspricht der Häufigkeitsumme der vollständig abgedeckten Intervalle und einem Anteil der Intervalle, die nur zu einem Teil im Anfragefenster liegen. Ist $a < v_0$ oder $b > v_{N_A}$, werden sie auf diese Grenzwerte gesetzt. Gilt gar $b < v_0$ oder $a > v_{N_A}$, erfüllt überhaupt kein Tupel die Anfrage und der Anfrageoptimierer kann, soweit minimaler und maximaler Wert des betroffenen Attributs bekannt sind, $S = 0$ setzen. Üblicherweise erwartet man von Histogrammen Schätzungen, die im Mittel nicht mehr als 10 % von der tatsächlichen Selektivität einer Anfrage abweichen.

Erste Histogramme. In seiner ursprünglichen Form wurde das Histogramm zu Beginn der 1980er eingeführt. Das klassische *breitengleiche Histogramm* (engl.: *equi-width histogram*)[4] teilt $V = [v_1, v_{N_A}]$ (hier: [0, 49]) in β gleich große Intervalle der Länge $(v_{N_A} - v_1 + 1)/\beta$ auf. Innerhalb dieser geht man von einer Gleichverteilung der Frequenzen aus sowie davon, dass alle möglichen Werte $v_1 \leq x \leq v_{N_A}$ vorkommen. Für $\beta = 5$ entstehen beispielsweise die Intervalle [0, 9], [10, 19], ..., [40, 49], denen die addierten Frequenzen ihrer enthaltenen Werte als Häufigkeit zugeordnet werden. Das entstehende Histogramm zeigen Tab 1 und Abb. 3. Zu beachten ist, dass in diesem und den weiteren Diagrammen die Häufigkeit pro Intervall gleich auf alle enthaltenen Attributwerte verteilt und die Höhe der Balken somit nicht unbedingt der einem Intervall zugeordneten

[4] Aus Stilgründen werden hier und im Weiteren eigene Übersetzungen an Stelle der englischen Begriffe verwendet, für die es keine gebräuchliche deutsche Pendants zu geben scheint

Gesamthäufigkeit entspricht. Damit erhält man einen besseren Vergleich mit der tatsächlichen Verteilung.

Tabelle 1. Breitengleiches Histogramm für Dienstjahre

Intervall	[0, 9]	[10, 19]	[20, 29]	[30, 39]	[40, 49]
Häufigkeit	88	194	96	34	8

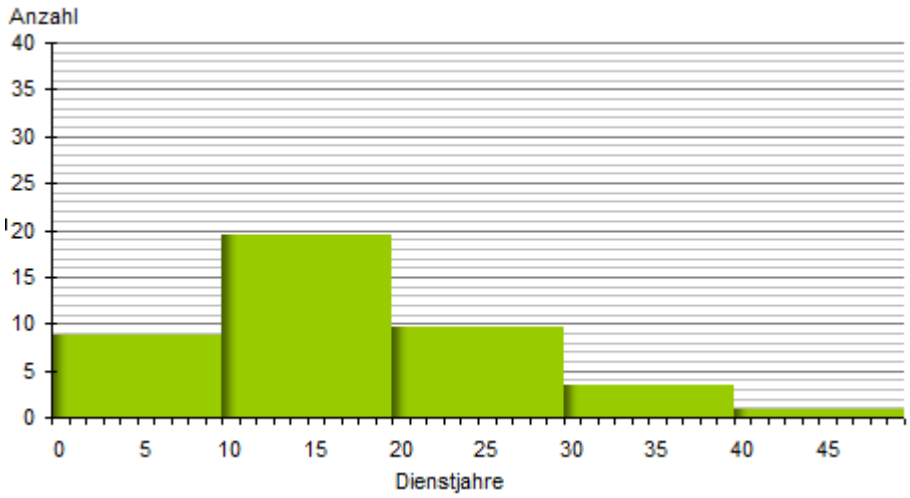

Abbildung 3. Breitengleiches Histogramm für Dienstjahre

Die Beispielanfrage von weiter oben würde das gegebene breitengleiche Histogramm folgendermamßen beantworten: Das Anfragefenster [15, 19] überschneidet sich mit genau einem Histogrammintervall, nämlich [10, 19] und ist genau halb so lang. Die Hälfte der Häufigkeit dieses Intervalls wird nun der Anfrage als Anzahl qualifizierender Tupel zugeordnet — das sind $0,5 \cdot 194 = 97$. Tatsächlich haben 63 Tupel Werte zwischen 15 und 19 für das Attribut Dienstjahre, in diesem Fall wäre die Schätzung also mit einem relativen Fehler von 54 % belastet.

Sieht man sich die tatsächliche Verteilung an, bemerkt man, dass die zweite Intervallhälfte, auf die sich diese Anfrage bezieht, deutlich weniger Tupel enthält als die erste Hälfte. Offensichtlich ließe sich damit leicht umgehen, wenn die Intervalle halbiert würden, indem man den Wertebereich in doppelt so viele davon aufgeteilt. Dann befände sich das Fenster [15, 19] sogar in einem eigenen Intervall mit ähnlichen Werten und man würde vermutlich eine bessere Schät-

zung erhalten — natürlich zu erhöhten Speicherkosten, da nun die Tabelle, die das Histogramm repräsentiert, doppelt so viele Einträge fassen muss. Weil man aber bei allen hier diskutierten Histogrammtypen derart vorgehen kann um bessere Schätzungen zu erhalten, werden weiterhin Histogramme mit 5 Intervallen betrachtet.

Klassifizierung. Untersucht man das breitengleiche Histogramm hinsichtlich Konstruktion und getroffener Annahmen, findet man Charakteristika, anhand derer sich diese Art von Histogrammen mit anderen vergleichen lässt bzw. sich von diesen unterscheidet. Damit lässt sich eine Klassifizierung aufstellen, deren wichtigste Dimensionen *Partitionsregel*, *Wertapproximation* und *Häufigkeitsapproximation* sind. Über die Werte in einem Intervall wird beim breitengleichen Histogramm die *Annahme kontinuierlicher Werte* getroffen, wie oben erläutert. Im gegebenen Beispiel erfasst das Histogramm also nicht, dass kein Tupel mit dem Dienstjahrewert 20 existiert. Für die Häufigkeitsapproximation wird eine Gleichverteilung angenommen.

Die Partitionsregel unterteilt sich in die Charakteristika Partitionsklasse, Sortierparameter, Quellparameter und Partitionsbeschränkung. Die *Partitionsklasse* gibt das Verhältnis der Intervalle zueinander an. Die Klasse der *seriellen* Histogramme, zu der das breitengleiche Histogramm gehört, ist die allgemeinste und stellt als wesentliche Anforderung, dass die Intervalle im Sinne des Sortierparameters angrenzend sind. Daneben wird paarweise Disjunktheit der Intervalle angestrebt, auch wenn das in manchen Fällen (siehe weiter unten) nicht erreicht wird. Das breitengleiche Histogramm erfüllt diese Anforderung, denn die Intervalle sind angrenzend und jeder Wert (Sortierparameter) befindet sich in genau einem Intervall. Grob gesagt werden bei seriellen Histogrammen alle Intervalle gleich behandelt und es gibt keine Einschränkung bezüglich der pro Intervall erfassten Werte. Weitere Klassen sind die *unregelmäßigen* Histogramme (engl.: *biased histograms*), bei denen mindestens ein Intervall nur aus einem Wert besteht und deren Sonderform, die *singulären* Histogramme (engl.: *end-biased histograms*), die bis auf ein Intervall nur einzelne Werte abdecken. Insbesondere letztere sind effizienter als serielle Histogramme, da für jedes Intervall, in dem nur ein Wert liegt, neben diesem nur die Häufigkeit gespeichert werden muss. Sammelintervalle müssen je nach Wertapproximation mehr Informationen bereit halten (siehe weiter unten).

Mit dem *Sortierparameter* legt man fest, in welcher Reihenfolge man die Attributwerte betrachtet, bevor sie in Intervalle aufgeteilt werden. Das breitengleiche Histogramm legt wie viele andere auch den Wertebereich selbst als Sortierung zu Grunde. Der *Quellparameter* wird zur Bestimmung der Intervallgrenzen benutzt — im Beispiel des breitengleichen Histogramms, bei dem alle Intervalle die gleiche Länge haben, ist das die Ausdehnung der Werte. Das erklärt sich, wenn man die *Partitionsbeschränkung* dieses Histogrammtyps betrachtet. Diese gibt an, welcher Beschränkung der Quellparameter innerhalb eines Intervalls unterliegt. Das breitengleiche Histogramm ist diesbezüglich *summengleich*, d.h. die Intervalle werden so gewählt, dass die Summe der Ausdehnungen der

enthaltenen Werte gleich ist. Dass daraus gleichlange Intervalle folgen, geht aus folgender Gleichung hervor, die die Summe der Ausdehnungen der Werte des Intervalls $[v_a, v_b]$ beschreibt:

$$\begin{aligned}\sum_{i=a}^{b} s_i &= \sum_{i=a}^{b} (v_{i+1} - v_i) \\ &= v_{a+1} - v_a + v_{a+2} - v_{a+1} + v_{a+3} - v_{a+2} + \ldots + v_b - v_{b-1} + v_{b+1} - v_b \\ &= v_{b+1} - v_a\end{aligned}$$

Diese Summe ist also gleich der Länge des Intervalls $[v_a, v_b]$, was zu zeigen war. Da der Wertebereich V aber nicht zwingend kontinuierlich ist und größere „Lücken“ aufweisen kann, sind die Intervalle in der Realität oft nicht genau gleich lang.

Zusammenfassen lässt sich die Partitionsregel mit der Notation *p(s, q)* von Poosala et al. [3], die Histogramme mit der Partitionsbeschränkung p, dem Sortierparameter s und dem Quellparameter q charakterisiert. Ist die Partitionsklasse nicht seriell, wird sie an p angehängt. Das breitengleiche Histogramm entspricht damit *summengleich(Wert, Ausdehnung)*. Es wurde bald nach seiner Vorstellung in der Doktorarbeit von R. P. Kooi vom DBMS *Ingres* übernommen und fand schnell auch in anderen Systemen Einsatz [2]. Weitere Arten der Wert- und Häufigkeitsapproximation sowie Partitionsbeschränkungen, Sortier- und Quellparameter sollen im Folgenden mit den dazugehörigen Histogrammtypen vorgestellt werden, bevor die vollständige Klassifizierung in einer Auflistung zusammengefasst wird.

Verbesserte Histogramme. Schon bald nach Einführung des klassischen breitengleichen Histogramms kam die Idee zu einem etwas adaptiveren Typ von Histogramm. Die schlichte Einteilung in gleich lange Intervalle schenkt „Hotspots“ wie im Bereich zwischen 9 und 12 des Eingangsbeispiels keine Beachtung. Außerdem fassen Intervalle teilweise stark unterschiedlich viele Werte — z.B. hat das zweite Intervall des breitengleichen Histogramms 24 mal so viele Einträge wie das letzte. Piatetsky-Shapiro et al. [9] schlugen daher das *tiefengleiche* Histogramm (engl.: *equi-depth* oder *equi-height histogram*) vor. Im Wesentlichen gibt dabei dieselbe Häufigkeit pro Intervall anstatt der Anzahl der Werte den Ausschlag für die Intervallgrenzen. In der *p(s, q)*-Notation entspricht das *summengleich(Wert, Frequenz)*, womit die Häufigkeit als Quellparameter benutzt wird. Obwohl sie ansonsten breitengleichen Histogrammen entsprechen, kann hier der Fall auftreten, dass Werte in mehr als ein Intervall fallen — dann nämlich, wenn ihre Häufigkeit größer ist als N_R/β, der jedem Intervall zugeordneten Häufigkeit,[5] oder wenn sie aufgrund der Sortierung nicht mehr ganz in ein Intervall passen.[6] Natürlich kann es auch hier durch die Gegebenheiten des Wertebereichs zu geringen Abweichungen kommen.

[5] Dies unter der Annahme, dass keine Nullwerte im betrachteten Attribut erlaubt sind — diese müssten dann von N_R abgezogen werden

[6] Hier wird die Disjunktheit der Intervalle serieller Histogramme nicht erreicht

In obigem Beispiel mit $N_R = 420$ Tupel ergibt sich die ideale Häufigkeit pro jedem der 5 Intervalle zu $420/5 = 84$. Das tiefengleiche Histogramm entspricht Tab. 2 bzw. Abb. 4.

Tabelle 2. Tiefengleiches Histogramm für Dienstjahre

Intervall	[0, 9]	[10, 12]	[13, 18]	[19, 26]	[27, 49]
Häufigkeit	88	94	85	83	70

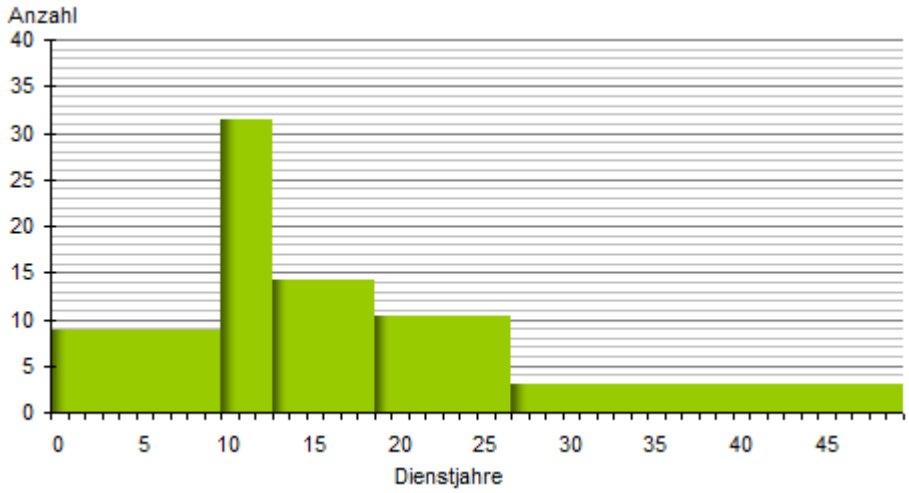

Abbildung 4. Tiefengleiches Histogramm für Dienstjahre

Zwar hat in diesem Fall kein einziges der 5 Intervalle die angestrebte Häufigkeit von 84, aufgrund der unterliegenden Daten ist dies aber die bestmögliche Aufteilung. Die erste Beobachtung ist, dass die Intervalle mit hochfrequenten Werten nun viel kleiner sind, weniger Werte zusammenfassen und damit genauere Abschätzungen ermöglichen. Dies geht zu Lasten der weniger häufig vorkommenden Werte, die nun in größeren Intervallen liegen. Die beispielhafte SQL-Anfrage, übersetzt zu einem Anfragefenster [15, 19], deckt sich nun mit zwei Dritteln des dritten und einem Achtel des vierten Intervalls. Unter Zuhilfenahme dieses Histogramms würde der Anfrageoptimierer also annehmen, dass sich $(2/3) \cdot 85 + (1/8) \cdot 83 \approx 67$ Tupel qualifizieren und liegt damit nur 6,4 % über der tatsächlichen Anzahl 63.

Aufgrund besserer Schätzungen löste das tiefengleiche Histogramm bald seinen Vorgänger ab und entwickelte sich in vielen Systemen zum Standardwerk-

zeug der Selektivitätsschätzung. Tatsächlich dauerte es bis Anfang der 1990er, bis weitere Histogrammtypen vorgeschlagen wurden. Die erste Alternative zu den damals gängigen tiefengleichen Histogrammen schlug die Frequenz an Stelle der Werte als Sortierparameter vor. Angewandt auf obiges Beispiel erhält man ein Histogramm, dessen Intervalle nun im mathematischen Sinne keine Intervalle mehr sind, sondern Mengen. Tabelle 3 zeigt ein Histogramm des Typs *summengleich(Frequenz, Frequenz)*, hat also Intervalle gleicher Tiefe, die idealerweise 84 beträgt. Für die Beispielanfrage, dessen Auswahlfenster sich mit zwei Intervallen des Histogramms überschneidet, liefert dieses mit einer Schätzung von $S = (3/8) \cdot 83 + (2/6) \cdot 90 \approx 61$ fast die korrekten 63 (3 % Fehler). Abbildung 5 zeigt, wie nahe man mit diesem Histogramm der tatsächlichen Verteilung aus Abb. 1 kommt.

Tabelle 3. Tiefengleiches Histogramm für Dienstjahre (Sortierparameter Frequenz)

Intervall	⟨0–2, 4, 7, 8, 20, 26–28, 30–49⟩	⟨5, 6, 16–18, 21, 24, 25⟩
Häufigkeit	80	83

Intervall	⟨3, 13, 15, 19, 22, 23⟩	⟨10, 11, 14, 29⟩	⟨9, 12⟩
Häufigkeit	90	96	71

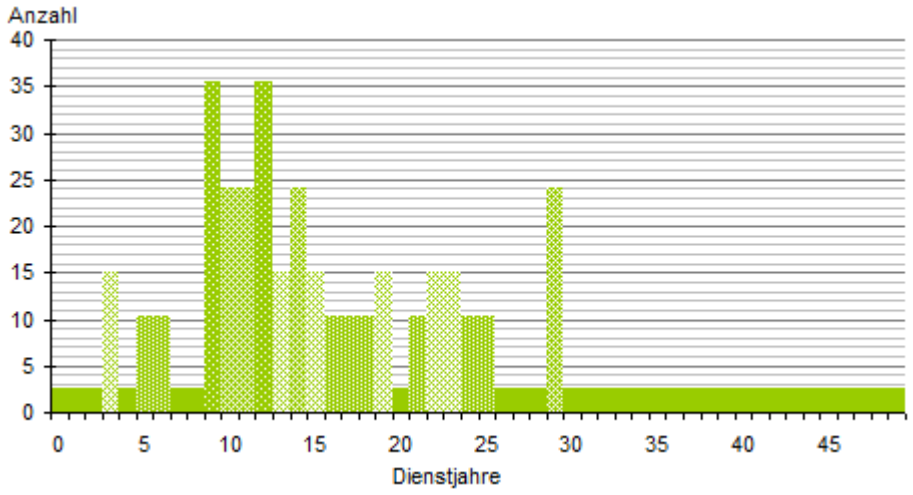

Abbildung 5. Tiefengleiches Histogramm für Dienstjahre (Sortierparameter Frequenz)

Weitere Arten. Die Frequenz erwies sich als ein sehr effektiver Sortierparameter, und so begann man davon ausgehend nach weiteren Histogrammarten zu suchen. Die Summengleichheit als bislang einzige Partitionsbeschränkung war relativ einfach und wurde deswegen Ansatzpunkt für elaboriertere Überlegungen. Man schlug vor, die gewichtete Varianz der Quellwerte über alle Intervalle zu minimieren — diese entspricht der Summe $\sum_{k=1}^{\beta} n_k V_k$, wobei n_k die Anzahl der Werte im k-ten Intervall ist. Die neue Partitionsbeschränkung nannte man kurz *V-optimal*. Weil dieser Ansatz deutlich komplizierter ist als der ursprüngliche (exponentieller Aufwand), adaptierte man den bekannten Algorithmus der *Iterativen Verbesserung* (engl.: *iterative improvement*) und stellte fest, dass man mit diesem heuristischen Mittel schnell effektive Histogramme generieren konnte. Histogramme der Klasse *v-optimal(Frequenz, Frequenz)* sind dabei von besonderer Bedeutung, denn für bestimmte Anfragen haben sie sich als optimal erwiesen [3].

In gegebenem Beispiel ist die Varianz für die Grenzen 10, 12, 15 und 29 mit 721,9 minimal. Das resultierende Histogramm beschreibt Tab. 4 und wird in Abb. 6 dargestellt. Die Testanfrage [15, 19] betrifft das vierte Intervall des Histogramms und würde unter Zuhilfenahme desselben vom Optimierer auf $(5/14) \cdot 140 = 50$ geschätzt werden. Das ist ein relativer Fehler von 20,6 %.

Tabelle 4. V-optimales Histogramm für Dienstjahre

Intervall	[0, 9]	[10, 11]	[12, 14]	[15, 28]	[29, 49]
Häufigkeit	88	55	76	140	61

Mitte der 1990er beschäftigten sich Poosala et al. [3] mit Bereichsanfragen und suchten nach weiteren Partitionsbeschränkungen mit dem Ziel große Unterschiede in Quellwerten eines Intervalls zu vermeiden. Die Beschränkung *MaxDiff* setzt Intervallgrenzen dort, wo sich die $\beta-1$ größten Differenzen zwischen in Sortierreihenfolge angrenzenden Quellwerten befinden. Das lässt sich am schnellsten exemplarisch am Attribut Dienstjahre begreifen. Ein Histogramm des Typs *maxdiff(Wert, Frequenz)* sortiert nach Werten und wählt die Intervallgrenzen nach den 4 größten Differenzen in der Frequenz (28, 24, 15 und 15), die zwischen angrenzenden Werten herrschen. Damit ergibt sich das Histogramm aus Tab. 5 und Abb. 7. Das Ergebnis der Anfrage [15, 19] würde der Optimierer damit auf $(5/7) \cdot 100 \approx 71$ schätzen und verfehlt die eigentliche Anzahl um 13,4 %.

Mit der Partitionsbeschränkung *komprimiert* werden die n größten Quellwerte in je einem Intervall gespeichert und die restlichen nach der Summengleichheit aufgeteilt. Poosala et al. wählten n als die Anzahl der Quellwerte, die a) größer sind als die Summe aller Quellwerte geteilt durch die Anzahl der Intervalle und b) kleiner ist als $\beta - 1$, denn die restlichen Werte müssen in mindestens einem weiteren Intervall erfasst werden. Wenig überraschend stellte sich heraus, dass die meisten komprimierten Histogramme unregelmäßig sind — nämlich dann,

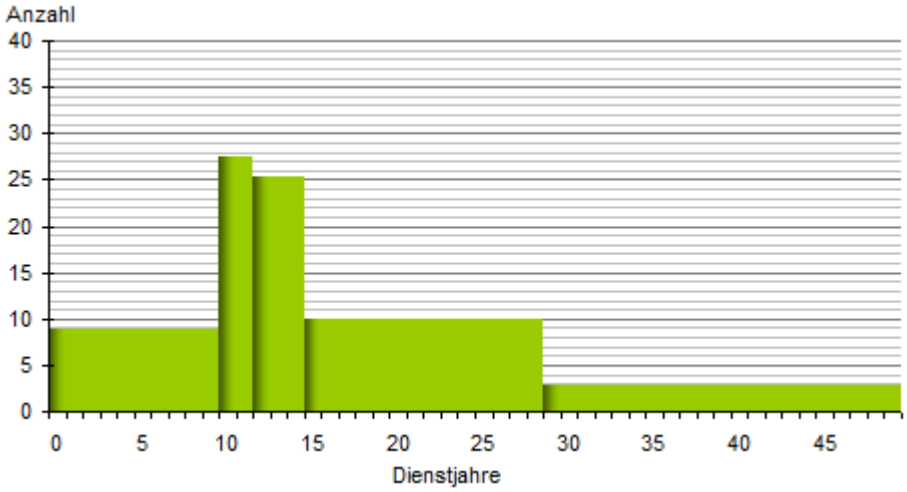

Abbildung 6. V-optimales Histogramm für Dienstjahre

Tabelle 5. MaxDiff-Histogramm für Dienstjahre

Intervall	[0, 8]	[9, 12]	[13, 19]	[20, 29]	[30, 49]
Häufigkeit	56	126	100	96	42

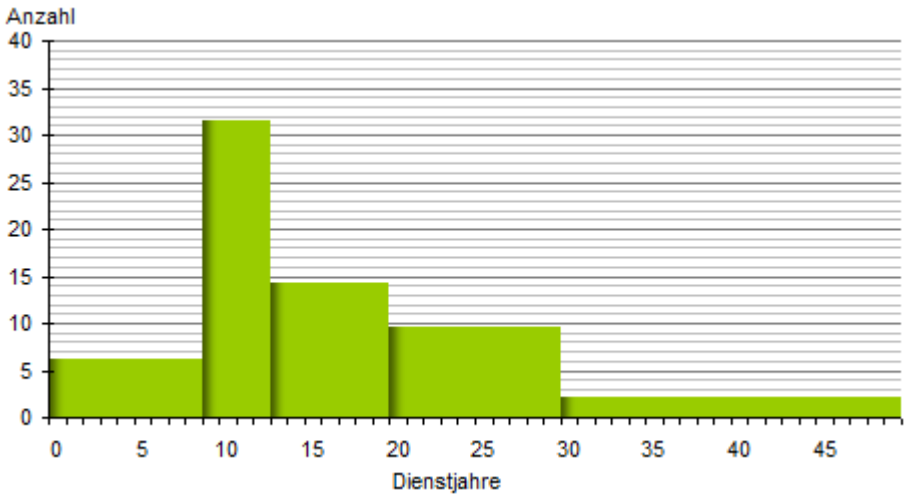

Abbildung 7. MaxDiff-Histogramm für Dienstjahre

wenn mindestens ein Quellwert Bedingung a) erfüllt und mehr als ein Intervall erzeugt wird, was normalerweise der Fall ist. Damit eignet sich diese Art von Histogrammen besonders für Attribute, deren Werte sehr ungleich verteilt sind und Ausreißer enthalten, die sonst in Sammelintervallen untergehen.

Für das Beispiel des Attributs Dienstjahre wird eine leicht veränderte Formel zur Berechnung von n benutzt. Es werden die Quellwerte in Einzelintervallen gespeichert, die größer sind als ein Drittel der Summe aller Quellwerte geteilt durch die Anzahl der Intervalle. Bedingung b) gilt weiterhin. Bei einem Histogramm des Typs *komprimiert(Wert, Frequenz)* berechnet sich n also als die Anzahl der Frequenzen, die größer sind als $1/3 \cdot 420/5 = 28$. Die beiden Werte 9 und 12 sind die einzigen, die häufiger als 28 mal vorkommen; die restlichen Werte werden in Intervallen mit der gleichen Gesamthäufigkeit von rechnerisch $(420 - 32 - 39)/3 \approx 116$ gespeichert. So erhält man ein unregelmäßiges Histogramm aus zwei Einzel- und drei Sammelintervallen, das nach oben definierter Notation ebenfalls der Klasse *komprimiert-unregelmäßig(Wert, Frequenz)* angehört und in Tab. 6 und Abb. 8 dargestellt ist.

Tabelle 6. Komprimiertes Histogramm für Dienstjahre

Intervall	⟨9⟩	⟨12⟩	⟨0–8, 10, 11⟩	⟨13–22⟩	⟨23–49⟩
Häufigkeit	32	39	111	122	116

Das Anfragefenster der Beispielanfrage deckt sich genau zur Hälfte mit dem vierten Intervall dieses Histogramms und würde schätzungsweise $1/2 \cdot 122 = 61$ Tupel zurückliefern. Dieses sehr gute Ergebnis liegt lediglich drei Prozent unter den tatsächlichen 63 Tupel, die bei der Ausführung der Anfrage zurück geliefert werden.

Weitere Überlegungen stellte man im Bereich der Sortier- und Quellparameter an, für die bisher nur Attributwerte und-frequenzen benutzt wurden. Damit erreichte man jeweils eine gute Annäherung an die Verteilung der Werte und ihre Häufigkeiten, aber nicht für beides gleichzeitig. Im *summengleich(Frequenz, Frequenz)*-Histogramm weiter oben fasst man ähnliche Frequenzen zusammen, vermischt dabei aber fast beliebig Attributwerte (nicht bei den letzten Intervallen, da die dort erfassten hohen Frequenzen allesamt in der Mitte des Wertebereichs liegen). Eine Lösung fand man in einem neuen Parameter, dem sogenannten *Bereich* a_i eines Werts v_i, definiert als das Produkt $s_i \cdot f_i$ aus Ausdehnung und Frequenz. Jeder Wert spannt damit ein Rechteck auf, dessen Kanten umso größer sind, wenn er alleinstehend ist und eine hohe Häufigkeit besitzt. Ein entsprechendes Histogramm auf obiges Beispiel anzuwenden macht wenig Sinn, da fast alle Werte die Ausdehnung 1 haben und deren Bereich somit gleich der Häufigkeit ist.

Für die Wertapproximation wurde bisher von kontinuierlichen Werten ausgegangen, wobei pro Intervall der kleinste und größte Wert gespeichert werden

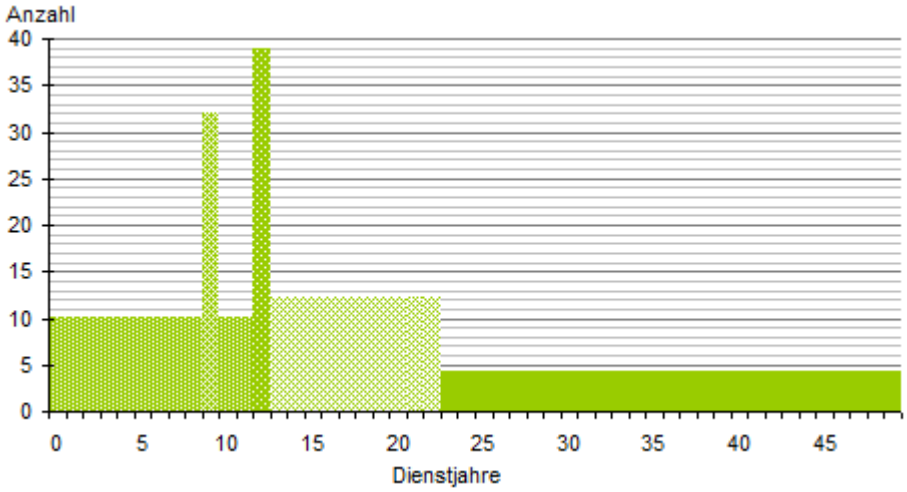

Abbildung 8. Komprimiertes Histogramm für Dienstjahre

muss und angenommen wird, dass alle Werte dazwischen vorkommen. Einen anderen Ansatz wählten Piatetsky-Shapiro et al. mit der *Punktwertannahme*. Pro Intervall weist man die Summe der Häufigkeiten aller enthaltenen Werte einem einzelnen Wert zu, zumeist dem kleinsten des Intervalls. Werden die Intervallgrenzen gut positioniert, können mit dieser scheinbar unsinnigen Methode trotzdem gute Ergebnisse erzielt werden [9]. Genauer, aber auch aufwändiger ist es alle vorkommenden Werte eines Intervalls zu speichern und die Gesamthäufigkeit auf diese aufzuteilen. Ein gutes Mittelmaß scheint die *Annahme gleichmäßiger Ausdehnung*, die zusätzlich zum Minimum und Maximum der Intervallwerte lediglich die Anzahl darin vorkommender Werte speichert und davon ausgeht, dass sie alle im gleichen Abstand voneinander liegen. Ob nun kontinuierliche Werte oder gleichmäßige Ausdehnung die bessere Annahme ist, konnten mehrere Experimente nicht abschließend feststellen, sodass von Fall zu Fall entschieden werden muss [2].

Weniger Alternativen neben der Gleichverteilung kennt man für die Häufigkeitsapproximation innerhalb eines Intervalls. Das diese so verbreitet ist liegt auch daran, dass gerade hier Kosten gespart und durch die Einteilung des Wertebereichs in Intervalle weniger Informationen vorgehalten werden sollen. Zudem wurde bereits viel Aufwand in Methoden zur geschickten Partitionierung gesteckt, damit die Häufigkeit möglichst billig angenähert wird.

Zusammenfassung und Bewertung. Damit wurden nun alle möglichen Eigenschaften eingeführt, die zu einer Klassifizierung beitragen. Zusammengefasst stellt sich diese folgendermaßen dar:

Partitionsregel zusammengesetzt aus:
Partitionsklasse Seriell, unregelmäßig, singulär

Partitionsbeschränkung Summengleich, V-optimal, MaxDiff, komprimiert
Sortierparameter Wert, Frequenz, Bereich
Quellparameter Ausdehnung, Frequenz, Bereich

Wertapproximation Annahme kontinuierlicher Werte, Annahme gleichmäßiger Ausdehnung, Punktwertannahme

Häufigkeitsapproximation Gleichverteilung

Es wurde bereits angesprochen, dass mehr Intervalle eine feinere Annäherung ermöglichen. Natürlich profitieren davon alle Arten von Histogrammen. Unterschiedliche Partitionsbeschränkungen gehen allerdings unterschiedlich „intelligent“ bei der Verteilung der Intervallgrenzen vor. So ist die gleichmäßige Positionierung der Intervalle der breitengleichen Histogramme weniger adaptiv als die aufwändige Berechnung der minimalen Varianz bei V-optimalen Histogrammen. Diese sowie MaxDiff- und komprimierte Histogramme haben die ausgefeiltesten der hier betrachteten Mechanismen zur Bestimmung der Intervalle. Die Beschränkung MaxDiff setzt Grenzen zwischen stark abweichende Quellwerte und verhindert damit bei genügend großem β allzu große Unterschiede in einzelnen Intervallen. Die komprimierten Histogramme eignen sich besonders, um starke Ausreißer getrennt zu speichern und für diese bessere Schätzungen zu erreichen. V-optimale Histogramme orientieren sich stattdessen am mittleren Quellwert und haben damit eine globalere Sicht auf die Verteilung.

Um die vorgestellten Histogrammtypen nach ihrer Güte zu ordnen, führten Poosala et al. ausgiebige Experimente durch. Sie betrachteten verschiedene Anfragetypen auf synthetischen Daten zwischen 100000 und 500000 Tupel und zwischen 200 und 1000 unterschiedlichen Werten des betrachteten Attributs. Dieses konstruierte man nach verschiedenen Verteilungen, darunter die Gleichverteilung und Zipf-Verteilung. Pro Histogramm gab es zwischen 10 und 20 Intervalle, denn der Speicherplatz war jeweils auf 160 Byte beschränkt. Weil vor allem singuläre Histogramme sehr speichereffizient sind, konnte man damit mehr Intervalle erzeugen als mit seriellen.

Den durchschnittlichen relativen Fehler verschiedener Histogrammarten, ermittelt über eine große Zahl von Anfragen, gibt Tab. 7 in absteigender Folge an.[7] Weit abgeschlagen an oberster Stelle findet sich dort die *System-R*-Methode als triviales Histogramm. Besser platzieren sich die breitengleichen Histogramme mit knapp unter 15 % Fehler. Tiefengleiche Histogramme positionieren sich an verschiedenen Stellen, je nach Konstruktion — dazu mehr im nächsten Abschnitt. Ebenfalls im Bereich zwischen 10 % und 15 % liegen singuläre Histogramme, deren Fokus eben nicht Genauigkeit, sondern Speichereffizienz ist. Danach folgt ein Bruch in der Rangfolge, die weiteren Histogramme liegen allesamt unter 4 % Fehler und sind entweder vom Typ komprimiert, V-optimal oder maxDiff, mit den beiden Spitzenreitern maxdiff(Wert, Bereich) und v-optimal(Wert, Bereich) mit weniger als 1 % Fehler.

[7] Zur Übersichtlichkeit werden folgende Abkürzungen benutzt: W = Wert, F = Frequenz, B = Bereich, K = kumulierte Häufigkeit

Tabelle 7. Histogrammrangliste nach durchschnittlichem Fehler

Histogrammtyp	**Fehler** (%)
Trivial	60,84
Tiefengleich (P^2)	17,87
V-optimal(B, B)	15,28
V-optimal(W, K)	14,62
Breitengleich	14,01
V-optimal(F, F)	13,40
V-optimal-singulär(B, B)	12,84
V-optimal-singulär(F, F)	11,67
Tiefengleich (Genau)	10,92
Komprimiert(W, B)	3,76
Komprimiert(W, F)	3,45
MaxDiff(W, F)	3,26
V-optimal(W, F)	3,26
MaxDiff(W, B)	0,77
V-optimal(W, B)	0,77

Zu beobachten ist, dass die Histogramme in diesem unteren Bereich nach Wert sortiert sind. Also scheint es zwischen nah beeinander liegenden Attributwerten genügend Korrelation zu geben, dass sich deren gemeinsame Behandlung in einem Intervall eher lohnt als die von z.B. ähnlichen Frequenzen, wenn nach dieser Eigenschaft sortiert wird. Auch ist Bereich ein besserer Quellparameter als Frequenz, denn v-optimal(Wert, Bereich) und maxdiff(Wert, Bereich) übertreffen ihre Pendants, bei denen die Häufigkeit als Quellparameter genutzt wird. Das tritt besonders bei stark unterschiedlichen Ausdehnungen von Attributwerten zu Tage, denen mit dem Bereich mehr Achtung geschenkt wird. Interessant ist die Feststellung, dass auch erhöhter Speicherplatz und damit mehr Intervalle die Fehlergrößen der Top-Histogramme nur unwesentlich verringern kann. Das liegt aber nicht zuletzt daran, dass diese bereits so gut sind, dass der verbleibende Fehler größtenteils durch minimale Varianzen in der Verteilung verursacht wird, denen nur durch eine genaue Speicherung aller Werte und Frequenzen beizukommen ist. Eine weitere Erkenntnis aus diesen Experimenten ist, dass die Annahme gleichmäßiger Ausdehnung bessere Resultate zur Folge hat als die anderen Wertapproximationen. In anderen, späteren Experimenten stellte sich dagegen die Annahme kontinuierlicher Werte als besser heraus, so dass eine endgültige Bewertung ausbleiben muss [3].

Konstruktion. Im vorherigen Abschnitt wurden Histogramme hinsichtlich ihrer Eigenschaften betrachtet und in Klassen eingeteilt. Allerdings wurde nicht auf praktische Aspekte eingegangen. So stellt sich die Frage, wie man bei einer gegebenen Relation ein Histogramm zu einem Attribut erstellt. Die Beispiele oben wurden alle durch genaue Betrachtung der Relation *Personal* erstellt.

Durch Analyse ihrer Tupel waren alle Werte und deren Häufigkeiten bekannt. Das erscheint im ersten Moment selbstverständlich, in der Praxis setzt das aber einen kompletten Tabellendurchlauf (*table scan*) voraus, während dem alle unterschiedlichen Attributwerte erfasst und ihre Häufigkeit durch Zähler bestimmt wird. Dieser vor allem bei großen Relationen nicht zu vernachlässigende Aufwand ist zum Zwecke der Selektivitätsabschätzung, bei der kein exaktes Wissen verlangt wird, kaum zu rechtfertigen. Es stellt sich also die Frage, wie man die Konstruktion von Histogrammen (sehr viel) günstiger gestalten kann.

Wie bereits festgestellt, ist der Aufwand eines Tabellendurchlaufs zur Konstruktion eines Histogramms allenfalls bei kleinen Relationen vertretbar. *Sampling*, also das Sammeln einer Stichprobe, ist eine übliche Alternative und wird in verschiedenen Variationen eingesetzt. Allen ist gemein, dass dabei nur ein geringer Anteil der Tupel einer Relation betrachtet wird, was im physikalischen Sinne bedeutet, dass diese Tupel die Speicherhierarchie von Externspeicher über Arbeitsspeicher und DB-Puffer durchlaufen müssen. Daraus ergeben sich zum einen große Kostenersparnisse, zum anderen durch den Verzicht auf genaue Informationen die Notwendigkeit für Methoden zur Bestimmung bzw. Annäherung von Attributwerten, Häufigkeiten und Ausdehnungen.

In der Rangliste der Histogrammtypen im vorherigen Abschnitt waren die einzelnen Ausprägungen nicht, wie die Beispiele in der Chronologie davor, mit Hilfe eines Tabellendurchlaufs erzeugt, sondern durch Sampling von 2000 Tupel (also 0,8–2 % der betrachteten Relation). Eine Ausnahme bilden die beiden tiefengleich-Histogramme, die sich in ihrer Konstruktion von den anderen unterscheiden. Während man bei breitengleichen Histogrammen bereits aus den Informationen über den Wertebereich $[v_1, v_{N_A}]$ diesen in Intervalle aufteilen kann, ist dies bei tiefengleichen schwieriger. Hier sind die Intervallgrenzen $q_1, \ldots, q_\beta$ nicht einfach aus Informationen über den Wertebereich zu bestimmen. Für sie gilt $q_i = min\{v_j \mid c_j/N_R \geq i/\beta\}$, d.h. man wählt die $i$te Grenze als den kleinsten vorkommenden Attributwert, für den der Anteil seiner kumulierten Frequenz an der Gesamtanzahl der Tupel mindestens i/β ist. Man nennt q_i das i/β-*Quantil* der Verteilung der Attributwerte. Die genaue Berechnung, wie sie bei der Konstruktion der Histogramme des Typs tiefengleich (genau) durchgeführt wurde, erfordert eine Sortierung der Relation nach dem betrachteten Attribut und der Extraktion des jeweils *r(i)*ten Tupel, wobei $r(i) = \lceil Ni/\beta \rceil$. Da hier neben des Tabellendurchlaufs die Relation noch sortiert wird, ist das in den meisten Fällen nicht akzeptabel.

Eine Alternative ist der P^2-*Algorithmus*, eine Form des Samplings. Dieser erfordert einen einzigen Tabellendurchlauf, während dem fünf Werte (m_{min}, m_{left}, m_q, m_{right}, m_{max}), genannt Markierungen, berechnet werden. Diese sind Annäherungen für den kleinsten vorkommenden Wert v_1 (m_{min}) bzw. den größten v_{N_A} (m_{max}), das zu bestimmende Quantil (m_q) und zwei Werte mittig zwischen den Randwerten und dem Quantil (m_{left} bzw. m_{right}). Mit jedem neuen Tupel werden diese Markierungen durch eine numerische Methode neu berechnet (stückweise parabolische Ausgleichsrechnung), am Ende wird m_q als das gesuchte Quantil benutzt. Damit erklärt sich, wieso das P^2-berechnete Histogramm we-

niger gute Schätzungen liefert als das durch Sortierung und Durchlauf erzeugte. Trotzdem gehört der P^2-Algorithmus zu den aufwändigeren Samplingmethoden. Die mehr oder weniger komplexen Berechnungen, die für jedes einzelne Tupel einer Relation durchgeführt werden, begründen Forderungen nach schnelleren Methoden.

Eine davon heißt *Zufälliges Sampling*. Diese betrachtet nur $n \ll N_R$ zufällig gewählte Tupel der ganzen Relation als Stichprobe und bestimmt auf dieser die Quantile. Um die Stichprobe zu erlangen, benutzt man die Technik *Reservoir-Sampling*. Dieser Algorithmus benötigt nur einen einzigen Durchlauf und arbeitet mit einem *Reservoir*, einer Sammlung von n Tupel. Dieses wird mit den ersten n Tupel der Relation gefüllt. Danach wird eine zufällige Anzahl von Tupel übersprungen. Das nächste Tupel ersetzt dann ein zufälliges der bereits gesammelten Proben im Reservoir und es wird eine neue Länge für den nächsten Sprung bestimmt. So wird die komplette Relation durchlaufen, deren Kardinalität nicht von Interesse ist. Weil die Verteilungsfunktion der Sprunglängen abhängt von der bisher durchlaufenen Tupelzahl, ist sicher gestellt, dass jedes Tupel die gleiche Chance hat sich am Ende im Reservoir zu befinden. Die darin enthaltenen Tupel bilden dann die Stichprobe.

Da P^2-Algorithmus und Zufälliges Sampling beide auf einem Tabellendurchlauf beruhen, haben sie gleiche E/A-Kosten. Allerdings betrachtet letzteres nur einen Teil der Tupel und führt auch keine Berechnungen darauf aus, darum verursacht Zufälliges Sampling geringere CPU-Kosten und ist somit die effizientere Methode [3].

Während Quantile nur für tiefengleiche Histogramme berechnet werden müssen, benötigt man in allen Histogrammtypen Informationen über die Häufigkeiten von Attributwerten. Bei einem Tabellendurchlauf kann man diese durch Zählvariablen für jeden vorkommenden Attributwert ermitteln, die immer dann erhöht werden, wenn ein Tupel den jeweiligen Wert annimmt. Das verursacht hohe CPU-Kosten, womit sich auch hier die Frage nach schnelleren Methoden stellt. Poosala et al. benutzten eine durch Reservoir-Sampling erlangte Stichprobe um die Häufigkeit eines Werts v_i durch $n_i N_R/n$ abzuschätzen. Dabei ist n_i die Häufigkeit von v_i in der Stichprobe der Größe n, welche dann auf die Größe der Relation extrapoliert wird [3].

Besonders häufig vorkommende Attributwerte haben große Auswirkung auf Anfrageergebnisse, darum ist dort eine möglichst genaue Schätzung wünschenswert. Komprimierte Histogramme speichern diese sogar in Einzelintervallen. Eine genaue Bestimmung der m am häufigsten vorkommenden Werte funktioniert nur nach obiger Methode der Zählvariablen. Da häufige Werte auch eine höhere Wahrscheinlichkeit haben in einer zufälligen, schon vorhandenen Stichprobe zu liegen, kann man die darin enthaltenen Attributwerte als die vermutlich häufigsten betrachten. Beim anschließenden Sampling werden für diese Werte Zähler geführt, sodass man deren Frequenz genau kennt. Die anderen Häufigkeiten werden durch die Samplingmethode bestimmt. Damit verbessert man besonders bei sehr ungleich verteilten Attributen die Schätzungen hochfrequenter Werte.

Bei Wertapproximation durch die Annahme gleichmäßiger Ausdehnung benötigt man die Anzahl unterschiedlicher Werte in einem Intervall $[v_a, v_b]$. Auch hier kann eine Stichprobe als Alternative zu einem Tabellendurchlauf dienen, indem die Anzahl unterschiedlicher Werte zwischen v_a und v_b in dieser Stichprobe als Schätzung für die tatsächliche Anzahl verwendet wird. Werte, die nicht in dieser Probe liegen, haben wahrscheinlich eine geringe Häufigkeit (siehe obige Argumentation) und fallen deswegen nicht ins Gewicht.

Benutzt man den Bereich eines Werts als Sortier- oder Quellparameter, ist neben der Frequenz auch die Ausdehnung zur Berechnung notwendig. Diese kann man natürlich wie bei der Bestimmung der vorkommenden Werte durch einen Tabellendurchlauf herausfinden. Doch auch hier kann man eine Stichprobe benutzen und für die darin vorkommenden Werte die Ausdehnung setzen, die sie in dieser Probe haben. Durch Experimente hat sich diese Methode als adäquat herausgestellt [3].

Wartung. Die Methoden im vorherigen Abschnitt ermöglichen es ein Histogramm entweder akkurat oder effizient und dennoch hinreichend genau zu erzeugen. Damit ergibt sich direkt ein weiteres Problem. Die Konstruktion steht am Beginn der Lebenszeit eines Histogramms. Da Relationen selten fix sind und Einfügungen, Änderungen sowie Löschungen von Tupel unterliegen, basiert ein Histogramm nur zum Zeitpunkt der Erstellung auf aktuellen Daten. Hier wirkt sich die approximative Natur dieser nichtparametrischen Methode positiv aus, denn ein gegebenes Histogramm nähert eine Relation R ähnlich gut an wie eine Relation R', die aus R durch Einfügen, Löschen oder Ändern von einigen n Tupel entsteht. Hat man festgestellt, dass sich die unterliegende Relation zu stark verändert hat (was durch n spezifiziert ist), ist die Neukonstruktion die sicherste Lösung. Eventuell erhält man durch eine Anpassung des Histogramm aber weiterhin gute Abschätzungen der Selektivität. Wie dabei vorgegangen werden kann und wann ein Histogramm neu erstellt oder aktualisiert werden sollte sind zwei Fragen, mit deren Beantwortung sich dieser Abschnitt befasst.

Wie oben geschildert, degeneriert ein Histogramm mit der Zeit durch Manipulation der Relation, zu dem es gehört. Eine offensichtliche Möglichkeit ist, das Histogramm neu zu erstellen, nachdem zum $n$ten Mal ein Tupel eingefügt, geändert oder gelöscht wurde. Das ist eine sehr inflexible Lösung, denn zwei Mengen von je n Manipulationen können das Histogramm verschieden stark verschlechtern. Alternativ lässt man es durch den Benutzer steuern, wann ein Histogramm neu aufgebaut werden soll, wie es zum Beispiel *DB2* durch den RUNSTATS-Befehl ermöglicht.

Ein anderer, interessanter Ansatz ist die Konstruktion nicht als statisches Ereignis zu Beginn der Lebenszeit eines Histogramms zu begreifen, sondern als einen ständigen, dynamischen Prozess, der Manipulationen in das Histogramm einpflegt, sodass dieses zu jedem Zeitpunkt äquivalent ist zu einem frisch konstruierten. Dazu verwenden Gibbons et al. eine spezielle Stichprobe, die durch Reservoir-Sampling erzeugt wird. Diese Stichprobe, genannt *Backing-Sample*, wird bei Einfügungen aktualisiert, in dem die neuen Einträge als weitere Tupel

im Tabellendurchlauf des Reservoir-Samplings betrachtet werden. D.h., man bestimmt am Ende des initialen Aufbaus der Stichprobe eine weitere Sprunglänge d und ersetzt dann eines der Tupel im Backing-Sample durch das $d$te einzufügende Tupel. Änderungen behandelt man, in dem der Attributwert des jeweiligen Tupel angepasst wird, falls es in der Stichprobe liegt. Ein zu löschendes Tupel wird aus dem Backing-Sample entfernt, wenn es darin liegt. Dadurch wird die Stichprobe kleiner und muss, sobald sie eine vorher festgelegte untere Grenze L erreicht, in einem Durchlauf der Relation neu aufgebaut werden. Gibbons et al. haben festgestellt, dass das bei selten vorkommenden Löschungen kaum passiert. Treten diese häufiger auf, werden die Kosten des Neu-Samplings durch die Kosten des Löschens amortisiert.

Das Backing-Sample wird nun an Stelle eines Tabellendurchlaufs zur Neukonstruktion eines Histogramms benutzt. Während bei letzterem durch Sprünge zufällige Tupel aus zumeist unterschiedlichen Blöcken betrachtet werden, kann das Backing-Sample in fortlaufenden Blöcken des Haupt- oder Externspeichers liegen. Dabei müssen pro Tupel nur das/die betrachtete(n) Attribut(e) gespeichert werden. Das verringert die E/A-Kosten und liefert damit einen Geschwindigkeitsvorteil gegenüber obigen Methoden, bei denen die tatsächliche Relation durchlaufen wird.

Allerdings erfordert nicht jede Manipulation an einer Relation eine Neukonstruktion des Histogramms. Für die einzelnen Operationen Einfügen, Ändern und Löschen haben Gibbons et al. verschiedene Regeln für Histogrammtypen aufgestellt, nach denen ein Histogramm entweder nur angepasst oder neu erstellt wird. Im Folgenden werden die Regeln für tiefengleiche Histogramme betrachtet. Bei diesem Typ von Histogramm versucht man den Wertebereich in Intervalle mit der Gesamthäufigkeit N_A/β aufzuteilen. Wie bereits festgestellt, wird diese Aufteilung meistens nur annähernd erreicht, insbesondere weil N_A/β oft keine ganze Zahl ist. Durch neu hinzukommende Tupel verändert sich diese Aufteilung nur leicht und man hat weiterhin annähernd die ideale Partitionierung. Ab einem gewissen Zeitpunkt driftet man aber zu weit davon weg, z.B. wenn Einfügungen wiederholt in das selbe Intervall fallen. Dieser Zeitpunkt wird als das Überschreiten eines Schwellwerts T der tatsächlichen Häufigkeit spezifiziert. Dieser ist im Wesentlichen ein Vielfaches der idealen Gesamthäufigkeit, z.B. $T = 1,2N_A/\beta$.

Ein solcher Überlauf löst ein Splitten des betroffenen Intervalls in zwei Hälften auf. Weil das Histogramm nun $\beta+1$ Intervalle hat, müssen zwei angrenzende Intervalle, deren addierte Gesamthäufigkeit höchstens T ist, zu einem einzigen verschmolzen werden. Während dieser Vorgang aus einfachem Aufsummieren der Gesamtfrequenz und Wegfallen der Intervallgrenze zwischen den beiden Intervallen besteht, erfordert das Splitten eines Intervalls etwas mehr Aufwand. Dabei muss ein ungefährer Mittelwert der darin liegenden Attributwerte berechnet werden, welcher dann als Intervallgrenze zwischen den beiden neuen Intervallen dient [10].

Mehrdimensionale Histogramme. Die bislang betrachteten Histogramme sind Annäherungen an die Verteilungen einzelner Attribute. Das ist für viele

Anfragen auch ausreichend. Bei dyadischen Ausdrücken, die sich auf zwei Attribute beziehen, benötigt man Informationen über die Verteilung von mehreren Attributen. Die meisten realen Systeme lösen dies schlichtweg, indem Abschätzungen pro Attribut getrennt betrachtet werden und von einer Unabhängigkeit der Attribute ausgegangen wird. In vielen Fällen ist das eine brauchbare Methode, korrelieren die Werte dieser Attribute allerdings, können daraus schlechte bis völlig falsche Abschätzungen resultieren.

Eine Abhilfe schaffen mehrdimensionale Histogramme, die den durch die Wertebereiche der Attribute aufgespannten Körper in mehrdimensionale Intervalle einteilen. Diesen werden Tupel[8] zugeteilt, deren Komponenten aus den jeweiligen Attributbereichen stammen und denen, wie auch bei den bisher betrachteten eindimensionalen Histogrammen, Häufigkeiten zugeordnet werden.

Das erste mehrdimensionale Histogramm *MHIST* wurde Ende der 1980er eingeführt und war im Wesentlichen ein zweidimensionales tiefengleiches Histogramm. Der Raum wurde schrittweise aufgeteilt durch Setzen einer Grenze in einem der Wertebereiche. So entstanden disjunkte Rechtecke mit gleicher Gesamthäufigkeit. Jahre später machte man sich die von Poosala et al. [3] erarbeitete Klassifizierung zu Nutze und verwendete auch in mehrdimensionalen Histogrammen neue Partitionsbeschränkungen wie MaxDiff und V-optimal und Quellparameter wie Frequenz und Bereich. Der Konstruktionsansatz blieb derselbe. In einer Weiterentwicklung namens *GENHIST* erlaubte man auch überlappende Intervalle, wobei man Anfragen, die solche Überschneidungen betrafen, mit Informationen aus mehreren davon abschätzte. Dadurch erhielt man genauere Ergebnisse als mit disjunkten Intervallen, obwohl deren Anzahl nicht erhöht wurde. *STHoles*-Histogramme wiederum sind von der Natur her mit komprimierten Histogrammen vergleichbar, denn sie entfernen einzelne Werte aus Intervallen und stecken sie in ein einzelnes, womit Löcher (engl.: *holes*) in Intervallen entstehen. Damit eignen sie sich gut für sehr unregelmäßig verteilte Attributkombinationen [2].

Kamel et al. wählten eine andere Herangehensweise für die Approximation mehrdimensionaler Verteilungen. Sie sahen eine Relation mit n Attributen als eine n-dimensionale Bitmap, in der jeder Punkt ein mögliches Tupel beschreibt. Ist das Tupel vorhanden, ist das entsprechende Bit gesetzt, wenn nicht, ist das Bit 0. Anfragen lassen sich damit auf n-dimensionale Körper abbilden, deren Ergebnis sich aus der Anzahl gesetzter Bits berechnet, die in diesem Anfragefenster liegen.

Hier wirkt sich der bekannte „Fluch der Dimensionalität“ aus, da sich die Größe der Bitmap aus dem Produkt der Kardinalität der einzelnen Attribute berechnet. Eine Relation aus 12 Attributen mit jeweils 100 möglichen Werten benötigt damit bespielsweise 12,5 TB Speicherplatz. Für damalige Verhältnisse praktisch nicht zu speichern, ist das auch heute nicht sinnvoll handhabbar. Da eine solche Bitmap aber nur sehr spärlich besetzt ist (da sie alle möglichen Tupel beschreibt), kann man mit Kompression sehr viel Speicherplatz einsparen. Kamel et al. wandten Methoden aus dem Gebiet der Mustererkennung an um homogene

[8] Hier im mathematischen Sinne

Bereiche einer Bitmap zu identifizieren und für diese die Anzahl darin gesetzter Bits als eine Zahl zu speichern — das Analogon zu den Intervallen der oben erläuterten mehrdimensionalen Histogramme. „Homogen“ definierte man als die Summe der Varianzen der Anzahl gesetzter Bits über alle Zeilen und Spalten. Je gleichmäßiger also die Einsen in einer solchen Zelle verteilt sind, desto homogener ist diese, und desto besser sind Schätzungen, da innerhalb von Zellen von einer Gleichverteilung der Einsen ausgegangen werden muss.

Da es ein zu großer Aufwand ist die Varianz für alle möglichen Zellen zu berechnen, begann man mit einer gleichmäßigen Aufteilung der Bitmap in Bereiche und berechnete lediglich für diese die Homogenität. Dann fasste man Zellen mit ähnlicher Homogenität zusammen. Am Ende erhielt man einige kleinere Zellen mit stark vom Rest abweichender Homogenität, der Rest war in größere Zellen aufgeteilt. Damit konnte man platzsparend gute Schätzungen ermöglichen [11].

Obwohl einige Ansätze vorgeschlagen wurden, haben sich mehrdimensionale Histogramme nicht in realen Systemen durchsetzen können. Stattdessen verwendet man in *Oracle Database*, *DB2* von IBM und Microsofts *SQL Server* Informationen über Attributkorrelationen aus Indizes und passt so die Schätzungen aus eindimensionalen Histogrammen an. Fehlen derartige Informationen, geht man von einer Unabhängigkeit der Attribute aus [2].

Varianten. Dieser Abschnitt beschreibt zwei Techniken, die bei Histogrammen benutzte Konzepte und Methoden weiter führen, wobei sie dabei gegenläufig vorgehen. Der erste Ansatz ist Sampling nicht zur Konstruktion von Histogrammen zu benutzen, sondern als einziges Hilfsmittel zur Selektivitätsabschätzung. Dabei werden im Zuge der Optimierung Anfragen auf einer Stichprobe ausgeführt und das Ergebnis dann zur Abschätzung der tatsächlichen Selektivität verwendet. Ist die Stichprobe gut gewählt und aktuell, erreicht man damit sehr genaue Annäherungen. Allerdings ist diese Methode nicht besonders schnell, da die Anfragen ausgeführt werden (wenn auch auf einem kleinen Teil der betroffenen Relation) und nicht mit vorberechneten Histogrammen geschätzt werden. Für einfügungsintensive Umgebungen wie Data Warehouses eignet sie sich noch am ehesten, da man so immer auf aktuellen Daten arbeitet. Ansonsten überwiegen die Laufzeitkosten den Aktualitätsvorteil [12].

Während pures Sampling ein sehr „datennahes“ Verfahren ist, ist die Philosophie von Aboulnaga et al. genau entgegengesetzt. Diese verfolgten die Idee Histogramme zu konstruieren ohne auf die unterliegende Relation zuzugreifen. Stattdessen nutzte man reale Anfragen im laufenden Betrieb um „kostenlos“ Informationen über Selektivitäten und die Verteilung der Attribute einer Relation zu erhalten und diese sogenannten *ST-Histogramme* (*Self-tuning histograms*) fortlaufend anzupassen. Dieser Ansatz ist vergleichbar mit der dynamischen Aktualisierung von Gibbons et al.

Bei ST-Histogrammen ist der Wertebereich in Intervalle gleicher Gesamthäufigkeit aufgeteilt. Da man zu Beginn noch keine Information über die Verteilung eines Attributs hat, geht man von einer Gleichverteilung der Attributwerte und ihrer Häufigkeit aus. Wird eine Anfrage gestellt, die das von einem ST-

Histogramm repräsentierte Attribut betrifft, kann man anhand des tatsächlichen Ergebnisses den Fehler bestimmen, den das Histogramm durch Abschätzung der Selektivität produziert hat. Dieser Fehler wird auf alle vom Anfragefenster überdeckten Intervalle aufgeteilt und deren Gesamthäufigkeit um einen Prozentsatz des Fehlers angepasst. Dieser berechnet sich im Wesentlichen aus dem Anteil, den die Frequenz des jeweiligen Intervall zur Gesamtschätzung beigetragen hat. Entsprechend muss ein vollständig abgedecktes Intervall mit hoher Gesamthäufigkeit stärker angepasst werden als ein nur teilweise abgedecktes mit geringerer Frequenz. So erreicht man schrittweise durch Anpassung der Frequenzen der Intervalle eine Verbesserung des Histogramms und nähert sich unweigerlich der tatsächlichen Verteilung an.

Diese Frequenzanpassung ist aber nur der eine Teil der Restrukturierung. Der andere betrifft die Intervalle selbst, deren Grenzen zuerst gleichmäßig verteilt werden. Dadurch passiert es leicht, dass Werte hoher Frequenz in den selben Intervallen landen wie solche mit geringer Häufigkeit. Durch die Wertapproximation der Gleichverteilung bedeutet dies, dass für Werte aus solchen Intervallen potentiell schlechtere Schätzungen resultieren. Also konstruierte man einen *Split&Merge*-Algorithmus ähnlich dem von Gibbons et al. [10], der die Partitionierung des Wertebereichs in Intervalle nach r Verwendungen des ST-Histogramms anpasste. In einer ersten Phase werden angrenzende Intervalle mit ähnlich häufig vorkommenden Werten zu einem *Run* zusammengefasst (*Merge*). Die Intervalle dieser Runs werden dann jeweils aggregiert, wodurch eine Kapazität an freien Intervallen entsteht. Diese wird benutzt um in der zweiten Phase nicht vom Merge-Prozess betroffene Intervalle mit besonders hoher Gesamthäufigkeit zu splitten. Je größer diese jeweils ist, desto mehr der freien Intervalle können zur Aufteilung eines Intervalls benutzt werden. Die Kriterien „ähnlich häufig“ und „besonders hoch“ werden dabei von Parametern gesteuert.

Während beim Ansatz von Gibbons et al. immer nur Paare von Intervallen gesplittet oder verschmolzen werden, fassen Aboulnaga et al. auch mehr als zwei Intervalle zusammen. Beim Splitten kann dagegen nicht der Mittelwert der in einem Intervall liegenden Attribute berechnet werden, denn es wird nicht auf die Daten zugegriffen. Stattdessen werden die neuen Grenzen gleichverteilt. Durch diese Repartitionierung und die anteilsmäßige Zuordnung der Frequenz entfernt man sich möglicherweise wieder von der tatsächlichen Verteilung des Attributs, was aber durch folgende Anfragen und Nachbesserungen korrigiert wird.

Gibbons et al. konnten in Experimenten zeigen, dass ST-Histogramme bei nicht allzu unregelmäßig verteilten Attributen Resultate mit einer Genauigkeit im Bereich von maxDiff(Wert, Bereich)-Histogrammen liefern. Für besonders ungleichmäßige Verteilungen ist die Idee nicht einmal auf eine Stichprobe der Daten zurückzugreifen ungeeignet. Dafür schlugen die Autoren Optimierungen vor, wie zum Beispiel, dass die Verfeinerung statt im laufenden Betrieb (online) offline durchgeführt wird. Anfrage- und Schätzergebnisse werden in einem Log gespeichert und dann später zur Anpassung benutzt. Zusätzlich kann man diese nur für Anfragen mit einem hohen Schätzfehler durchführen. Eine wichtige Eigenschaft der ST-Histogramme ist jedoch, dass sie so genau sind wie nötig: Je

öfter ein ST-Histogramm benutzt wird, desto öfter wird es angepasst und desto genauer nähert es die Verteilung an. Insgesamt erreicht man mit ihnen eine erhebliche Reduktion der Konstruktionskosten (insbesondere E/A), da gänzlich auf Sampling oder gar komplette Tabellendurchläufe verzichtet wird.

Auch für mehrdimensionale Verteilungen können ST-Histogramme benutzt werden. Hier erweisen sie sich sogar als besonders günstig, denn traditionelle mehrdimensionale Histogramme sind teuer. Ihre Genauigkeit bewegt sich dabei auf dem selben Niveau [13].

Einsatz in realen Systemen. Heute werden Histogramme in nahezu allen Datenbankverwaltungssystemen eingesetzt. Wie beschrieben, setzte sich das klassische breitengleiche Histogramm schnell gegen die einfachen Mechanismen, die man in *System R* anwandte, durch und fand in tiefengleichen Histogramme bald eine sinnvolle Verbesserung. Tatsächlich verwendete *Oracle Database* noch Anfang des Jahrtausends tiefengleiche Histogramme. Bei Microsofts *SQL Server* setzte man dagegen in dieser Zeit MaxDiff-Histogramme mit bis zu 199 Intervallen ein. *DB2* verwendete komprimierte Histogramme, standardmäßig mit 10 Einzel- und 20 Sammelintervallen. In der aktuellen Version 9.7 benutzt man beim *DB2*-Workload-Manager unregelmäßige Histogramme mit 41 Intervallen. Dabei gibt der Benutzer einen Grenzwert für das jeweilige Attribut an. Über diesem liegende Werte werden im 41. Intervall, einem Sammelintervall, erfasst, die Grenzen der anderen 40 Intervalle werden in exponentiell ansteigendem Abstand gesetzt, sodass die ersten Intervalle Einzelwerte speichern [2,14].

3 Kostenmodelle

Ziel der Anfrageoptimierung ist es aus einer Vielzahl möglicher Ausführungspläne für eine Anfrage einen im Sinne der Ressourcenbeanspruchung möglichst günstigen auszuwählen. Dazu müssen den Ausführungsplänen Kosten zugewiesen werden. Das ist ein iterativer Vorgang, denn die Pläne sind Bäume von physischen Operatoren, deren Kosten einzeln modelliert und am Ende zusammengefasst werden.

Im nächsten Abschnitt werden Modelle für einzelne Operatoren und für Verknüpfungen von Operatoren präsentiert. Dabei werden die Kosten zuerst aus logischer Sicht betrachtet. Danach werden physikalische Aspekte der Kostenmodellierung betrachtet und insbesondere Faktoren vorgestellt, die darauf Einfluss nehmen. In einer Formel werden diese dann zu einem konkreten Mittel der Kostenberechnung zusammengeführt.

3.1 Grundlagen

Einem Operator A eines Ausführungsplans, der eine Relation oder ein Zwischenergebnis als Eingabe nimmt und diese in eine Ausgabe transformiert (z.B. $A =$ Selektion), werden in der Kostenmodellierung Ausführungskosten und Statistiken über die Ausgabe zugewiesen. Letztere beinhalten u.a. Eigenschaften wie

die Kardinalität der Ausgabe in Tupel, die Sortierung der Ausgabe usw. und werden von Mannino et al. als *statistisches Profil* bezeichnet. Die Ausführungskosten werden mit Hilfe des statistischen Profils des vorangehenden[9] Operators B berechnet und können sich aus verschiedenen Faktoren zusammensetzen, die im Abschnitt 3.3 erläutert werden. Bis dahin seien die Kosten eines Operators äquivalent zu der Anzahl der Tupel, die er ausgibt. Ebenso konstruiert man das statistische Profil eines Operators mit demjenigen des vorangehenden Operators. So wird der Baum in einem Bottom-Up-Verfahren durchlaufen, an dessen Ende man die Kosten des gesamten Ausführungsplans als die Summe der Ausführungskosten all seiner Operatoren definiert.

Bei den statistischen Profilen unterscheiden Mannino et al. zwischen Basisprofilen und Zwischenprofilen. Basisprofile beziehen sich auf Relationen oder Sichten (d.h. Blätter eines Operatorbaums) und können im Allgemeinen genau berechnet werden. Bei Zwischenprofilen ist das nicht möglich, da sie Zwischenergebnissen zugeordnet sind, d.h. Ausgaben von Operatoren. Diese können vor der Ausführung der Anfrage nur geschätzt werden. Hier wird die Wichtigkeit einer guten Selektivitätsabschätzung deutlich, denn diese Zwischenprofile beeinflussen alle folgenden Profile und Operatoren bis hin zur Wurzel des Baums [8].

3.2 Kosten von Operatoren

Wie bereits erläutert, werden Kosten in diesem Abschnitt auf einer logischen Ebene betrachtet. Die Kosten eines Operators werden also bestimmt durch die Kardinalität seiner Ausgabe. In diesem Abschnitt werden Kostenmodelle für die gängigsten Operatoren und häufig vorkommende Kombinationen von Operatoren, d.h. Teilen eines Operatorbaums, aufgestellt. Anzumerken ist, dass sich die folgenden Modelle auf Operatoren der relationalen Algebra beziehen und deswegen nicht immer auf ihre Entsprechungen in SQL anwendbar sind.

Elementare Operatoren. Die Selektion gehört zu den wichtigsten Operatoren der relationalen Algebra. Sie gibt die Tupel der Eingabe aus, die ihr Prädikat erfüllen. Die Berechnung der Kosten dieses Operators basiert im Wesentlichen auf der Selektivität S des Prädikats, die mit den bereits diskutierten Methoden geschätzt und durch folgende Formel konkretisiert wird:

$$N_{Ausgabe} = S \cdot N_{Eingabe}$$

Dabei bezeichnen $N_{Eingabe}$ bzw. $N_{Ausgabe}$ die Anzahl der Tupel der Ein- bzw. Ausgabe [1].

Mannino et al. [8] unterscheiden verschiedene Klassen von Prädikaten, für die sich manche Methoden der Selektivitätsabschätzung besser eignen als andere. Die Methoden aus *System R* sind zwar simpel, aber auf alle Klassen anwendbar. Für einfache relationale Anfragen sowie Bereichsanfragen werden häufig

[9] B geht A voran, wenn die Ausgabe von B Eingabe von A ist

Histogramme eingesetzt, weil sie genauere Schätzungen ermöglichen. Bei dyadischen Ausdrücken mit Gleichheit als Verknüpfung geht man generell von einer Unabhängigkeit der Attribute aus. Dann lassen sich auf die einzelnen Attribute wiederum die bekannten Methoden anwenden. Bei anderen Vergleichszeichen wendet man üblicherweise den *System-R*-Ansatz an. Dasselbe gilt für relationale Ausdrücke mit Parameter. Eine weitere Klasse sind komplexe Ausdrücke. Dabei besteht das Prädikat einer Selektion aus einer Konjunktion, Disjunktion oder Negation von Ausdrücken, die wiederum komplex sein können. Generell wird von einer Unabhängigkeit der Attribute ausgegangen und eine der folgenden Formeln angewandt:

$$S(P_1 \wedge P_2) = S(P_1) \cdot S(P_2)$$

$$S(P_1 \vee P_2) = S(P_1) + S(P_2) - S(P_1) \cdot S(P_2)$$

$$S(\neg P_1) = 1 - S(P_1)$$

Angenommen jedoch, die Prädikate P_1 und P_2 aus der ersten Formel sind nicht komplex. Liegt Information über die Korrelation dieser beiden Attribute vor (z.B. in Form eines mehrdimensionalen Histogramms), ist es sinnvoller auf die Vereinfachung durch die Multiplikation der Einzelselektivitäten zu verzichten und das konjunktive Prädikat $P_1 \wedge P_2$ mit diesem Wissen abzuschätzen [1].

Die Projektion ist ein weiterer Operator der relationalen Algebra, der Tupel auf ihren Teil ihrer Attribute einschränkt. Er ist in gewisser Weise orthogonal zur Selektion, denn in der Vorstellung von Relationen als Tabellen selektiert letztere Reihen, also Tupel, die Projektion dagegen in erster Linie Spalten, also Attribute. Da in der relationalen Algebra Duplikate, d.h. identische Tupel, verboten sind, kann es auch bei einer Projektion passieren, dass die Ausgabe weniger Tupel enthält als die Eingabe, nämlich wenn sich mindestens zwei Tupel der Eingabe nur in einem oder mehreren der auszuschließenden Attribute unterscheiden.

Da Projektionen nur selten als Blätter eines Operatorbaums vorkommen, sondern viel häufiger in Kombination mit einer Selektion oder anderen Operatoren, gibt es hier nur wenige Vorschläge aus der Forschung. Der einfachste Fall ist, wenn nur ein Attribut beibehalten wird. Dann ist die Kardinalität der Ausgabe gleich der Anzahl unterschiedlicher Werte dieses Attributs, weil hier durch mehrfaches Vorkommen eines Werts direkt Duplikate entstehen würden. Werden weitere Spalten beibehalten, lässt sich eine genaue Aussage über die Ausgabegröße nur treffen, wenn Informationen über die Wertkombinationen dieser Spalten existieren. Da dies selten der Fall ist, wird zumeist von einer Unabhängigkeit der Attribute ausgegangen sowie davon, dass jede Kombination vorkommt. Damit lässt sich die Ausgabegröße nach oben abschätzen durch das Produkt der unterschiedlichen Attributgrößen oder, falls diese kleiner ist, die Größe der Ausgangsrelation. Die Anzahl unterschiedlicher Attributwerte kann wieder durch Sampling bestimmt und danach von der Stichprobe auf die gesamte Relation extrapoliert werden. Diese Abschätzung ist eine untere Schranke der tatsächlichen

Anzahl, denn in einer Stichprobe können niemals mehr Attributwerte enthalten sein als in der Relation selbst[10].

Der Verbundoperator ist eine Verknüpfung des kartesischen Produkts zweier Relationen mit einer Selektion und eventuell einer Projektion. Dabei entstehen Zwischenergebnisse, die um ein Vielfaches größer sein können als die Ausgangstabellen, deswegen ist eine gute Schätzung umso kritischer und wichtiger. Es wird hier nur der Gleichverbund betrachtet, da andere Verhältniszeichen nie im Fokus von Untersuchungen standen. Leider muss man auch hier bei den meisten Methoden von einer Unabhängigkeit der zu verbindenden Attribute ausgehen, denn es können nicht für alle möglichen Attributkombinationen Informationen über deren Korrelation vorgehalten werden.

Die Kardinalität der Ausgabe einer Verbundoperation schätzt man mit der Formel

$$N_{R_1 \bowtie R_2} = S_J \cdot N_{R_1} \cdot N_{R_2}.$$

Dabei bezeichnen N_{R_1} und N_{R_2} die Kardinalitäten der Eingaberelationen und S_J den sogenannten *Verbundselektivitätsfaktor*. Es ist schwierig diesen genau zu bestimmen. Besteht zwischen den zu verknüpfenden Attributen eine verlustfreie (n:1)-Beziehung, so dass jedes Tupel einen „Partner“ in der anderen Relation findet, ergibt sich die Größe der Ausgabe als die Kardinalität der mächtigeren Eingaberelation, d.h. es gilt $S_J = 1/min\{N_{R_1}, N_{R_2}\}$ bzw.

$$N_{R_1 \bowtie R_2} = max\{N_{R_1}, N_{R_2}\}.$$

Diese Formel lässt sich häufig anwenden, denn die Kardinalitäten liegen, wie bereits erwähnt, meistens in der Statistik eines Datenbanksystems vor.

Ein anderer Ansatz multipliziert die Anzahl der Vorkommen der Verbundattributwerte und summiert darüber:

$$\sum_{i=1}^{N_A} (N_{i_1} \cdot N_{i_2})$$

Dabei ist N_{i_1} bzw. N_{i_2} die Anzahl der Tupel mit dem Attributwert v_i in R_1 bzw. R_2. Hier sind allerdings genaue Informationen über die Vorkommen einzelner Werte von Nöten, deswegen ist es fraglich, in wie weit Histogramme bei dieser Formel nützlich sind. Diese halten Informationen über Wertebereiche vor und eignen sich daher besser für Bereichsanfragen [1,8].

Verknüpfungen von Operatoren. Die eben vorgestellten Rechenmethoden für elementare Operatoren sind nur in der sogenannten *unabhängigen* Modellierung ausreichend. Dabei werden zur Berechnung der Kosten eines Operators nur die Ausgabegrößen des vorangehenden Operators benutzt. In der *integrierten* Modellierung dagegen betrachtet man Verknüpfungen von Operatoren, wie sie in

[10] Unter der Annahme, dass Löschungen in der Relation in die Stichprobe propagiert werden

Ausführungsplänen vorkommen, die aus mehr als nur einer Operation bestehen. Zwei solche Verknüpfungen sollen hier diskutiert werden.

Beim Verbund zweier Relationen, auf denen zuvor bereits Selektionen ausgeführt wurden, kann die oben vorgestellte Formel $\sum_{i=1}^{N_A}(N_{i_1} \cdot N_{i_2})$ nicht mehr angewandt werden, da sich durch die Selektion die Größen der Eingaberelationen geändert haben können. Stattdessen wurde eine Anpassung vorgeschlagen, die die Selektivitäten der Selektionsoperationen (S_1 und S_2) einbezieht:

$$S_1 S_2 \sum_{i=1}^{N_A}(N_{i_1} \cdot N_{i_2})$$

Die Summe selbst blieb unverändert, nun aber wendet man die kombinierten Selektivitäten auf die Ausgabegröße des Verbunds an, denn dieser arbeitet mit Tupeln, die durch die Selektion bereits herausgefiltert wurden.

Eine weitere Verknüpfung von Operatoren ist eine Projektion nach einem Verbund, wofür wieder die Anzahl verschiedener Attributwerte interessiert. Bei Nichtverbundattributen ändert sich diese durch den Verbund nicht. Beim Verbundattribut jedoch kann sie sich verringern, wenn ein Wert in einem Attributbereich vorkommt, im anderen aber nicht. Man schätzt die Anzahl unterschiedlicher Werte des Verbundattributs nach dem Verbund ab durch

$$\sum_{i=1}^{N_A}(p_{i_1} \cdot p_{i_2}).$$

Hier sind p_{i_1} und p_{i_2} die Wahrscheinlichkeiten, dass ein Attributwert v_i in R_1 bzw. R_2 vorkommen. Kennt man diese nicht, kann man ein Histogramm verwenden, das die Anzahl unterschiedlicher Werte pro Intervall speichert. Dann ergibt sich p_i als der Quotient dieser Anzahl und der Länge des Intervalls.

Auch für andere Verknüpfungen wurden Berechnungsmethoden vorgeschlagen, die hier aufgeführten Beispiele sollen jedoch genügen um die Problematik der integrierten Kostenmodellierung aufzuzeigen. Sie ist oft aufwändig, da Operatoren nicht mehr getrennt voneinander betrachtet werden können, erlaubt dafür aber genauere Abschätzungen als mit der unabhängigen Modellierung zu erreichen sind [8].

3.3 Physikalische Kostenmodellierung

Im letzten Abschnitt wurden Kosten von Operatoren durch die Größe ihrer Ausgabe in Tupel modelliert. Dieser Ansatz findet zwar durchaus Verwendung, schenkt aber den physikalischen Aspekten der Datenspeicherung keine Beachtung. Tupel liegen auf Externspeichern und müssen von dort erst in den Hauptspeicher geladen werden. Dabei macht es einen Unterschied, ob man die Tupel blockweise laden kann oder ob sie verstreut gespeichert sind. Derartige physikalische Eigenschaften werden im folgenden Abschnitt angesprochen.

Hintergrund. Relationen sind aus logischer Sicht eine unsortierte Menge von Tupeln. Tatsächlich liegen diese in Seiten des Externspeichers, von wo aus sie bei Referenzierung in den DB-Puffer, einen Teil des Hauptspeichers, geladen werden müssen. Daraus folgen physische Seitenzugriffe, deren Anzahl ein entscheidender Faktor für die Kosten eines Ausführungsplans ist.

Um die physikalischen Aspekte zu verdeutlichen sei eine Selektion mit dem Bereichsausdruck $[v_a, v_b]$ gegeben, der sich auf ein Attribut A bezieht. Es wird davon ausgegangen, dass keine referenzierte Seite bereits im DB-Puffer liegt. Die Relation oder das Zwischenergebnis, die/das Eingabe für einen Operator ist, liegt gespeichert in einer Anzahl von Seiten, die aus dem statistischen Profil der Eingabe zu entnehmen ist. Wird als Zugriffspfad ein Tabellendurchlauf benutzt, müssen diese Seiten alle referenziert, d.h. zugegriffen und geladen werden.

Liegt ein Index auf dem Attribut vor, ergibt sich die Anzahl der Seitenzugriffe als die Anzahl der selektierten Tupel, denn diese liegen potentiell alle in verschiedenen Seiten. Hinzu kommen Zugriffe auf Indexseiten, deren Anzahl allerdings im Allgemeinen gering ist und die meistens im DB-Puffer vorgehalten werden. Hier würde man der Selektion geringere Kosten zuweisen, da in den meisten Fällen weniger Seitenzugriffe stattfinden. Qualifizieren sich allerdings sehr viele Tupel für die Selektion, hat die sequentielle Vorgehensweise weniger Seitenzugriffe zur Folge als der wahlfreie Zugriff über den Index. Bei diesem werden im Gegensatz zum Tabellendurchlauf Seiten u.U. mehrfach referenziert und müssen erneut vom Externspeicher geladen werden, wenn sie zwischenzeitlich aus dem DB-Puffer verdrängt wurden. In diesem Fall würde man einen Tabellendurchlauf günstiger bewerten.

Ist die Eingabe zusätzlich bezüglich A aufsteigend sortiert, d.h. es liegt ein *Clustering* bezüglich A vor, spart man den Zugriff auf die Seiten, die ausschließlich Tupel mit einem A-Wert kleiner als v_a enthalten. Außerdem müssen nur die Seiten bis zu den Tupeln mit dem Wert v_b geladen werden.[11] Zusätzlich wirkt sich hier der *Clusterfaktor* aus, denn da nun in einer Seite aufeinanderfolgende Tupel des Ergebnisses liegen, reduziert sich die Anzahl zugegriffener Seiten um die durchschnittliche Anzahl Tupel pro Seite (d.h., sie wird durch diese geteilt). Auch hier ist der Weg über den Index potentiell günstiger als ein Tabellendurchlauf, wenn sich nicht zu viele Tupel für das Ergebnis qualifizieren.

Kostenfaktoren. Die Beachtung der physikalischen Seitenzugriffe ist ein wesentlicher Schritt hin zur Berechnung konkreter Kosten für Ausführungspläne. Je nach der Systemumgebung, auf der sich ein Datenbanksystem befindet, können allerdings weitere Faktoren eine Rolle spielen.

Dazu gehören an erster Stelle E/A-Kosten, die oft durch die Anzahl physischer Seitenzugriffe quantifiziert werden. Demnach sind Ausführungspläne günstiger, wenn für sie weniger Tupel vom Externspeicher geladen werden müssen. Ein weiterer Aspekt sind die Berechnungskosten. Die Verarbeitung von Tupeln

[11] Analoges gilt bei absteigender Sortierung

hat Aufrufe des Zugriffssystems[12] zur Folge und macht einen großen Teil der Rechenzeit in einem Datenbanksystem aus. Die Anzahl dieser Aufrufe ist daher ein repräsentatives Maß für die CPU-Beanspruchung. Eine dritte Größe sind die Kommunikationskosten, die bei verteilten Datenbanksystemen eine große Rolle spielen. Diese werden durch das Zugreifen bzw. Verschicken entfernter Daten verursacht und sind ein eigenes Thema der Anfrageverarbeitung, weswegen sie im Weiteren außen vor gelassen werden.

Datenbankserver lassen sich, je nach aktueller Last, als *CPU-beschränkt* oder *E/A-beschränkt* ansehen (d.h. der limitierende Faktor ist CPU bzw. E/A). Entsprechend möchte man Ausführungspläne bevorzugen, die im ersten Fall weniger Rechenaufwand bzw. im zweiten Fall möglichst wenig Externspeicherzugriffe erfordern. Diese adaptive Modellierung spiegelt sich in folgender Formel wider, die eine konkrete Berechnung der Kosten C eines Ausführungsplans erlaubt:

$$C = \langle \text{Anzahl physischer Seitenzugriffe} \rangle + W \cdot \langle \text{Anzahl Zugriffssystemaufrufe} \rangle$$

W ist dabei ein nach momentaner Last gewähltes Gewichtungsmaß. Ist ein System E/A-beschränkt, sollte ein kleines W gewählt werden, womit eine hohe Anzahl physischer Seitenzugriffe teuer bewertet wird. Bei Beschränkung durch die CPU empfiehlt sich dagegen ein großes Gewichtungsmaß, wodurch Aufrufe des Zugriffssystems größeren Einfluss auf die Kosten eines Ausführungsplans besitzen. Statt nach diesem dynamischen Ansatz kann man W auch nach Einschätzung oder empirischem Wissen über den DB-Server statisch festlegen.

Die Bedeutung der oben genannten Faktoren hat sich — ebenso wie die Technik — im Laufe der Jahrzehnte verändert. Die klassische „Zugriffslücke“ befindet sich zwischen Externspeicher und Hauptspeicher, wo sich Zugriffszeiten um einen Faktor im Bereich 10^5 unterscheiden. Im Laufe der Jahre wurden Prozessoren immer schneller und verbesserten sich mit einer Geschwindigkeit, mit der nun der Hauptspeicher nicht mehr Schritt halten konnte. Heute erreicht man zudem durch Mehrkernprozessoren einen hohen Grad an Parallelität, mit der häufigere Seitenreferenzen einhergehen. Damit ergibt sich zwischen Prozessor und Hauptspeicher eine neue Zugriffslücke.

Mit zunehmender Kapazität des Hauptspeichers verlor die Zugriffslücke zum Externspeicher an Bedeutung, weil einmal gepufferte Seiten potentiell länger im Hauptspeicher verblieben. Durch *Solid State Drives* (SSDs), einer neuen Alternative zu Magnetfestplatten mit z.T. enorm kürzeren Zugriffszeiten, schrumpft diese Lücke weiter. Insgesamt muss man heutige Systeme damit eher als CPU-beschränkt ansehen, da die E/A-Kosten von geringer Bedeutung sind und gleichzeitig Zugriffssystemaufrufe unter dem neuen „Flaschenhals“ Hauptspeicher zu leiden haben [1,16,17].

[12] Das Zugriffssystem ist verantwortlich für die Verwaltung von physischen Sätzen und Zugriffspfaden

4 Fazit

In dieser Arbeit wurde ein Überblick über zentrale Arbeitsmittel der Anfrageverarbeitung gegeben. Wichtige Methoden der Selektivitätsabschätzung wurden vorgestellt und ihre Vor- und Nachteile sowie Einsatzmöglichkeiten diskutiert. Dabei wurden insbesondere verschiedene Arten und Varianten von Histogrammen im Laufe der Zeit betrachtet, klassifiziert und in eine Rangfolge gebracht. Des Weiteren wurde die Modellierung der Kosten von Ausführungsplänen behandelt, wobei Kostenmodelle für Operatoren und Operatorsequenzen aufgestellt wurden. Im Anschluss wurden diese auf physikalischer Ebene untersucht sowie wichtige Faktoren vorgestellt und in Zusammenhang gesetzt.

Trotz mittlerweile gut drei Jahrzehnten Forschungsarbeit ist das Thema Anfrageverarbeitung und insbesondere deren Optimierung noch nicht ausgeschöpft. Offene Aspekte sind beispielsweise die effiziente Erfassung mehrdimensionaler Verteilungen, bei denen oft stark vereinfachende Annahmen zu schlechten Abschätzungen führen. Die Gewichtung der verschiedenen physikalischen Kostenfaktoren befindet sich im Wandel und ist keinesfalls ein abgeschlossenes Thema, denn durch innovative Technologien wie SSDs und Mehrkernprozessoren müssen traditionelle Annahmen neu bewertet werden.

Literatur

1. Härder, T., Rahm, E.: Datenbanksysteme: Konzepte und Techniken der Implementierung (zweite, überarbeitete Auflage). Springer (2001)
2. Ioannidis, Y.: The History of Histograms (abridged). Proceedings of the 29th VLDB Conference, 19–30 (2003)
3. Poosala, V., Ioannidis, Y., Haas, P., Shekita, E.: Improved Histograms for Selectivity Estimation of Range Predicates. Proceedings of the 1996 ACM SIGMOD international conference on Management of data, 294–305 (1996)
4. Chen, C. M., Roussopoulos, N.: Adaptive selectivity estimation using query feedback. Proceedings of the 1994 ACM SIGMOD international conference on Management of data, 161–172 (1994)
5. Normalverteilung — Wikipedia, `http://de.wikipedia.org/w/index.php?title=Normalverteilung&stableid=67880735`
6. Parametric statistics — Wikipedia, `http://en.wikipedia.org/w/index.php?title=Parametric_statistics&oldid=329635372`
7. Non-parametric statistics — Wikipedia, `http://en.wikipedia.org/w/index.php?title=Non-parametric_statistics&oldid=322307531`
8. Mannino, M. V., Chu, P., Sager, T.: Statistical profile estimation in database systems. ACM Computing Surveys (CSUR), Volume 20, Issue 3, 191–221 (1988)
9. Piatetsky-Shapiro, G., Connell, C.: Accurate estimation of the number of tuples satisfying a condition. Proceedings of the 1984 ACM SIGMOD international conference on Management of data, 256–276 (1984)
10. Gibbons, P. B., Matias, Y, Poosala, V.: Fast incremental maintenance of approximate histograms. Proceedings of the 23rd VLDB Conference, 466–475 (1997)
11. Kamel, N., King, R.: A model of data distribution based on texture analysis. Proceedings of the 1985 ACM SIGMOD international conference on Management of data, 319–325 (1985)

12. Ioannidis, Y., Poosala, V.: Balancing histogram optimality and practicality for query result size estimation. Proceedings of the 1995 ACM SIGMOD international conference on Management of data, 233–244 (1995)
13. Aboulnaga, A., Chaudhuri, S. Self-tuning histograms: building histograms without looking at data. Proceedings of the 1999 ACM SIGMOD international conference on Management of data, 181–192 (1999)
14. IBM DB2 9.7 Information Center, http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.admin.wlm.doc/doc/c0052789.html
15. Chaudhuri, S.: An overview of query optimization in relational systems. Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems, 34–43 (1998)
16. Ramanathan, R. M., Thomas, V.: Platform 2015: Intel ® Processor and Platform Evolution for the Next Decade. http://www.ifi.uio.no/~inf3410/docs/Intel_Corp_Platform_2015.pdf (2005)
17. Lee, S., Moon, B., Park, C.: Advances in flash memory SSD technology for enterprise database applications. Proceedings of the 35th SIGMOD international conference on Management of data, 863–870 (2009)