Datenbank-Caching: Modellierung des Füllverhaltens von Cache-Groups

Diplomarbeit
von
Christian Bayerlein

Technische Universität Kaiserslautern
Fachbereich Informatik
DBIS Datenbanken und Informationssysteme

Übersicht

- Einführung
- Grundlagen der Modelle
- baumartige Cache-Groups
- RCC-zyklenfreie, nicht-baumartige Cache-Groups
- allgemeine Cache-Groups

Einführung

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
FACHBEREICH INFORMATIK
DBIS Datenbanken und Informationssysteme

Caching

Web-Caching (Caching von Web-Objekten, wie HTML-Fragmenten...)

Vorteile:

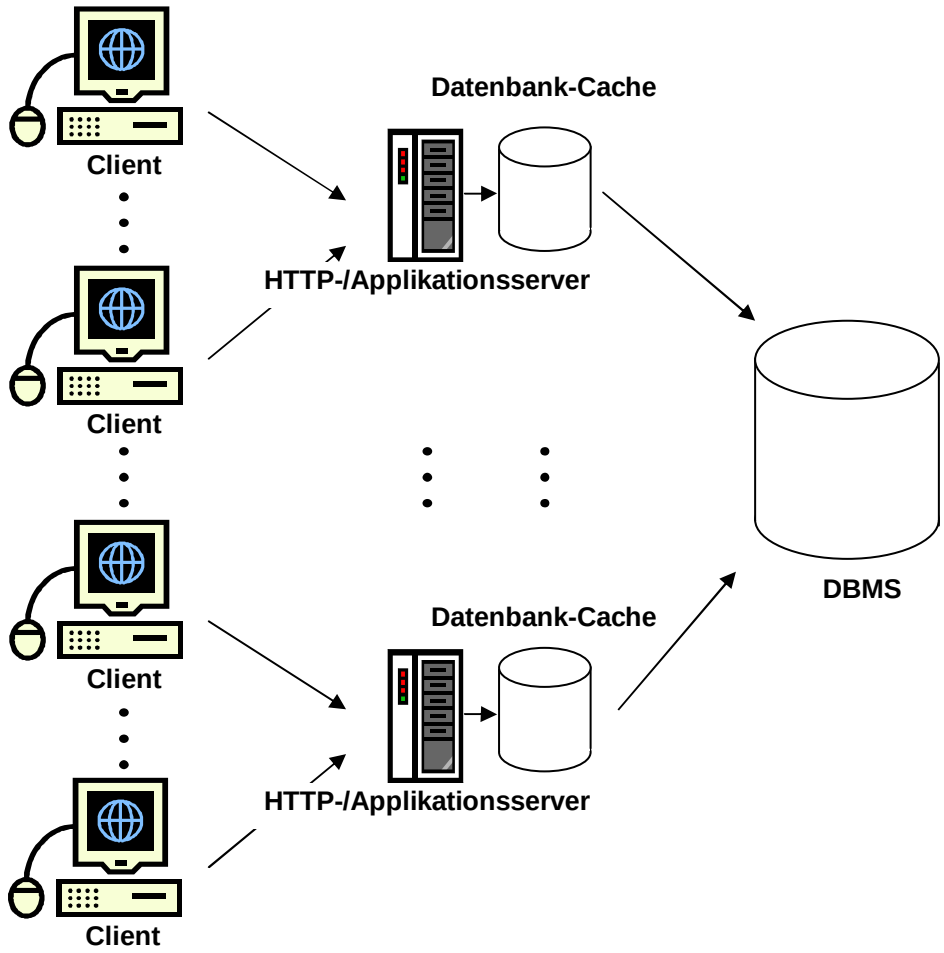
- Entlastung der Kommunikationsverbindungen
- Verkürzung der Antwortzeit
- Entlastung des Servers

Datenbank-Caching

- Caching von Objekten aus einer DB
- Backend/Frontend
- Daten: korrekt und aktuell
- Einfachster Ansatz: Volltabellen-Caching
- materialisierte Sichten

A cache is a collection of duplicate data, where the original data is expensive to fetch or compute (usually in terms of access time) relative to the cache. Future accesses to the data can be made by accessing the cached copy rather than refetching or recomputing the original data, so that the perceived average access time is lower.

Eine Web-Applikation mit DB-Caches



Constraint-basiertes Caching

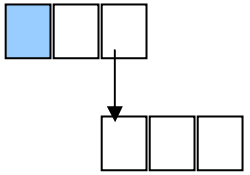
- Definition der zu speichernden Daten über Constraints
- Constraints regeln, wann der Cache einen gültigen Zustand hat.
- (Parametrisierte) Prädikate
- Extensionen
- Instanziierung
- Prädikatvollständigkeit

Cache-Groups für Gleichheitsprädikate

Gleichheitsprädikate

- Zwei Varianten:
  - Ein Attribut wird auf einen bestimmten Wert fixiert
  - Ein Attributwert wird mit einem zweiten verglichen
- Unterstützte Prädikate:

$$T_1.c = \varphi_{ck} \wedge T_1.e = T_2.k \wedge \ldots \wedge T_{n-1}.e = T_n.k$$

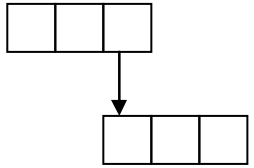


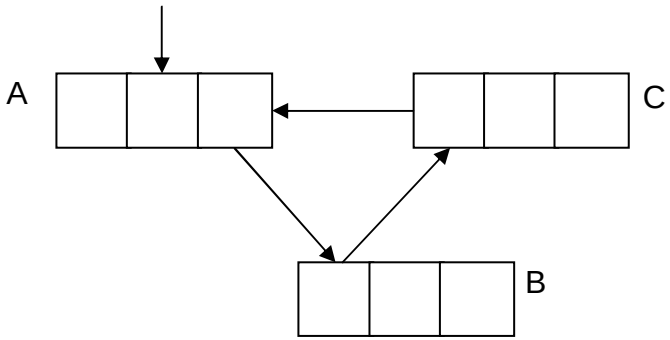
Cache-Groups für Gleichheitsprädikate

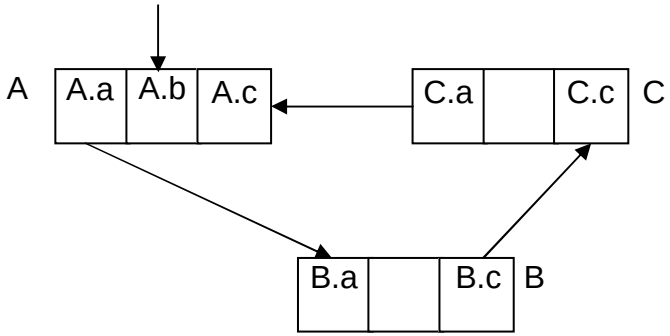
- Cache-Tabellen und Constraints
- RCCs – verbinden zwei Spalten über eine Werte-Beziehung (für Equi-Joins)
- Cache-Keys – Füllpunkteigenschaft, Einstiegspunkteigenschaft
- Ein paar Begriffe: Werteumgebung, Umgebungsvollständigkeit, Einstiegsfähigkeit, DBD...

Anfrageverarbeitung und Füllverhalten

- Sondierung, Cache-Hit, Cache-Miss
- Füllzyklus, Füllschritt
- induzieren von Werten, ungesättigte Spalten
- Schmuggeln/Schmuggler-Beziehungen
- stabile Cache-Group
- exzessives Füllverhalten: rekursives Laden; sichere Cache-Groups, homogene und heterogene Zyklen





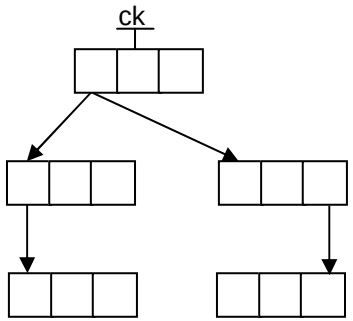


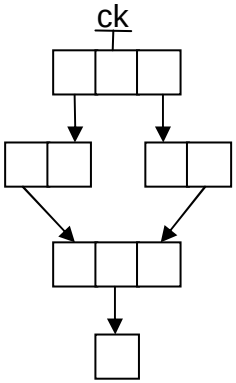
Quantitative Analyse

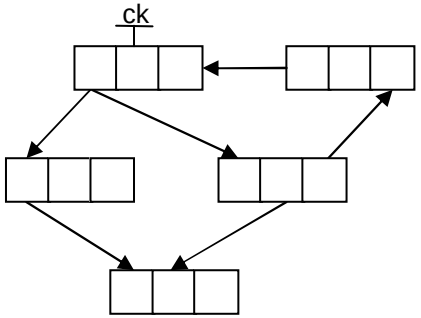
- Ziel: Cache-Advisor
- kostenbasierte Bewertung
- Granularität: auf der Ebene der Tabellen
- Kosten durch Bereitstellen von Speicherplatz
- Kosten durch steigende Netzwerk- und System-Last
- Gesamtkosten, Füllstände

Entwickelte Modelle, Klassifizierung der Cache-Groups

- Baumartige Cache-Groups
- nicht-baumartige, RCC-zyklenfreihe Cache-Groups
- allgemeine Cache-Groups





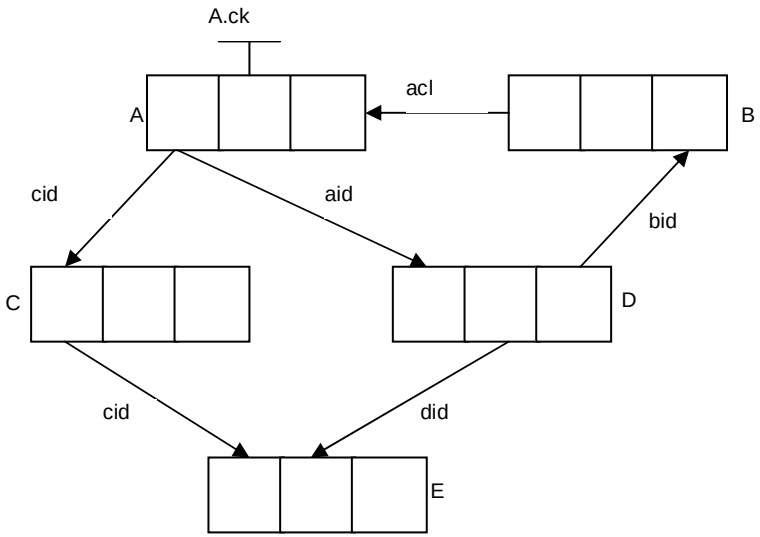


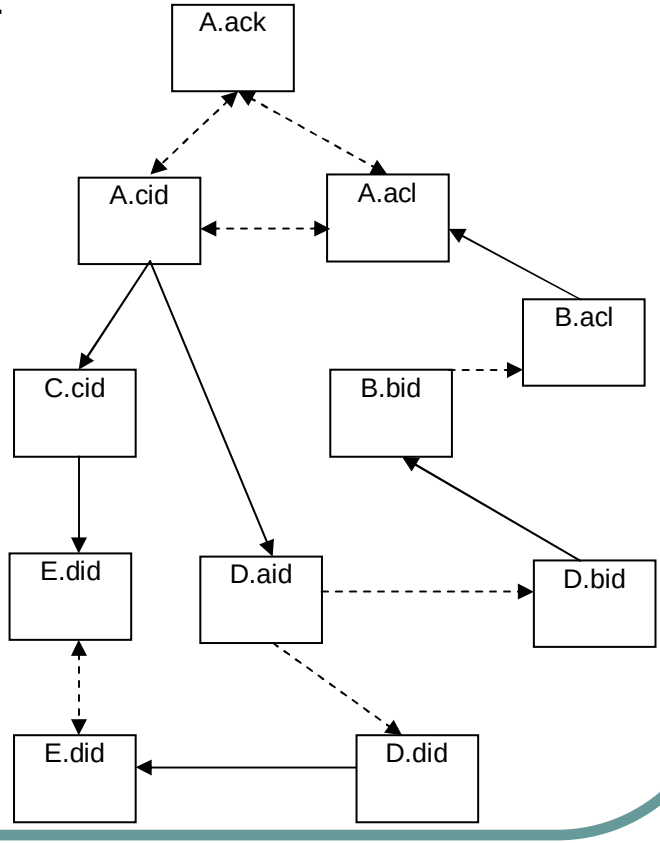
Grundlagen der Modelle

Technische Universität Kaiserslautern
Fachbereich Informatik
DBIS Datenbanken und Informationssysteme

Der Spalten-Beziehungsgraph

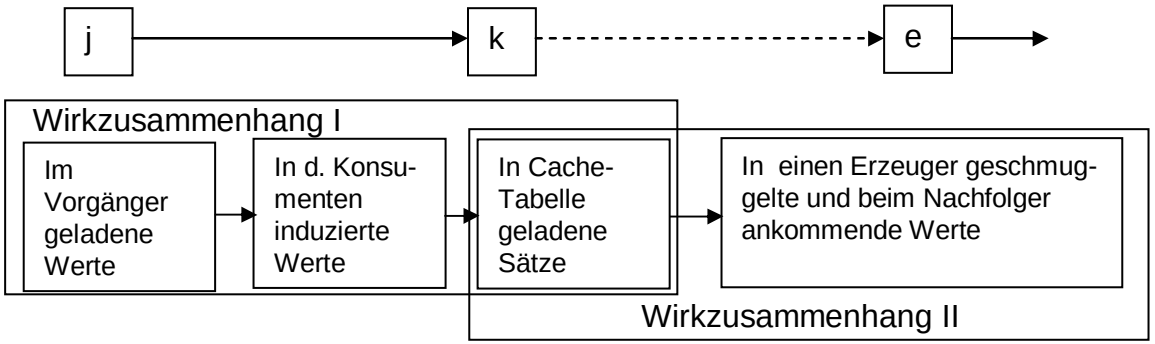
- Relevante Spalten
- RCC-Kanten und Schmuggler-Kanten
- Erzeuger, Konsument, Basiserzeuger
- Schmuggelquelle/Schmuggelziel



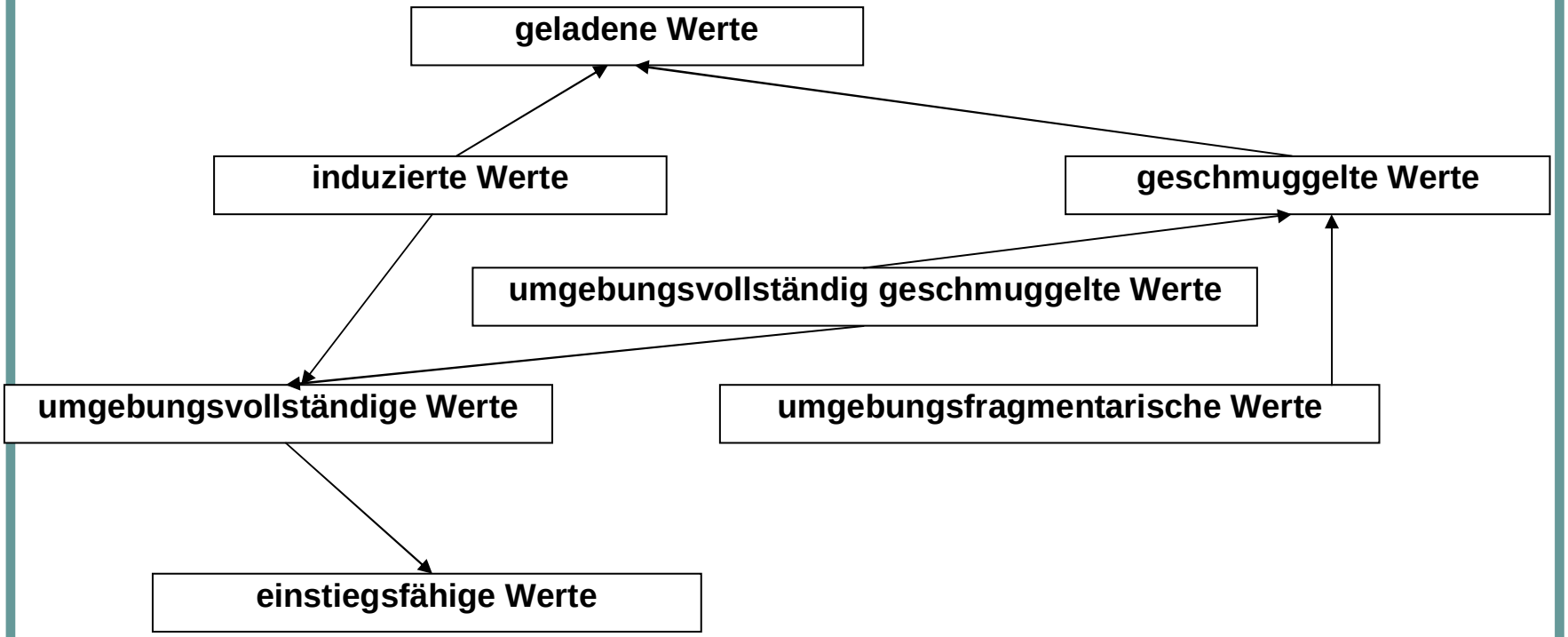


Füllschritte

- Ausgelöst durch ungesättigte Spalte
- Einzelwert-Füllschritte, direkte RCC-Füllschritte, zusammenfassende RCC-Füllschritte
- Zeitpunkte der Verarbeitung, Serialisierung



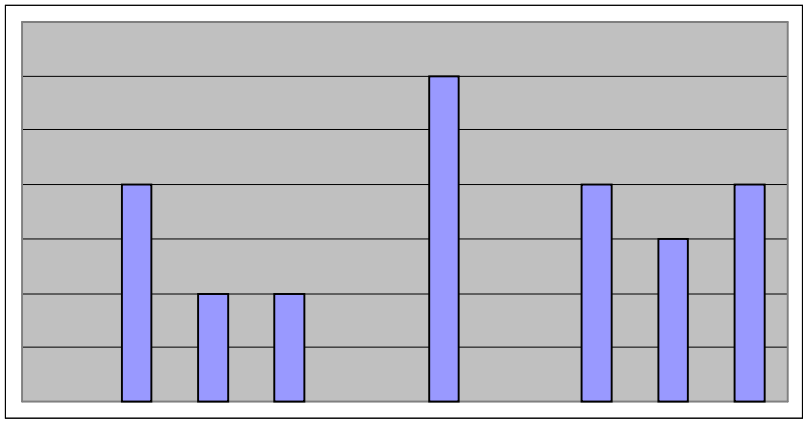
Werte und Wertemengen



Histogramme und Zugehörigkeitsfunktion

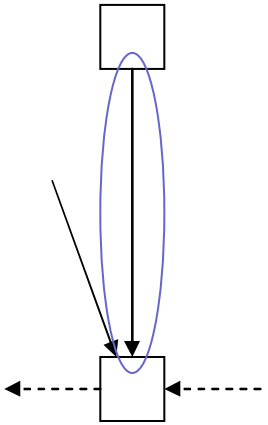
- Ein Histogramm gibt an, wie häufig ein Wert in einer Spalte vorkommt
- Histogramme können zur Steigerung der Genauigkeit der Modelle verwendet werden; Einsatz hauptsächlich bei baumartigen Cache-Groups

- Die Zugehörigkeitsfunktion gibt an, ob ein Wert in der DBD einer Spalte liegt.
- Die gemeinsame Zugehörigkeitsfunktion gibt an, ob ein Wert in mehreren DBDs liegt



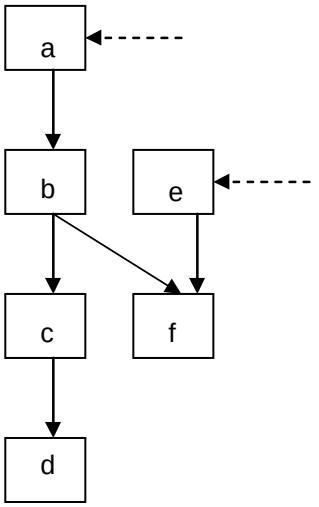
RCCs

- RCC-Implikation: Werte, die im Vorgänger geladen sind, und in der DBD des Nachfolgers vorhanden sind, sind im Nachfolger einstiegsfähig
- isolierte RCCs
- Die RCC-Teilmengen-Annahme
- Überdeckung zwischen zwei Spalten



Wertemengen-Pfade

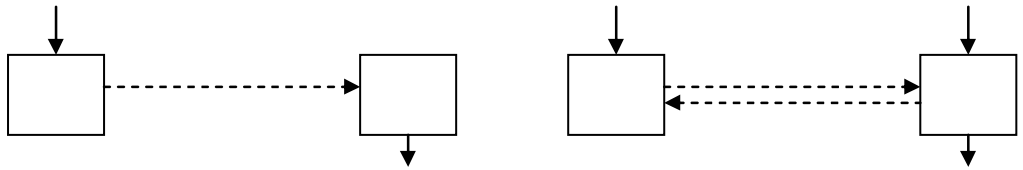
- Wertemengen-Pfade erweitern die RCC-Eigenschaft für einzelne Werte transitiv
- Über einen WMP induzierte Werte
- maximale WMPe und Basiserzeuger
- isolierte Wertemengen-Pfade, Basiserzeuger-Zuordnungsfunktion



- Ladewahrscheinlichkeit im Konsumenten ist abhängig von:
  - Ladewahrscheinlichkeit im Basiserzeuger
  - kombinierte Zugehörigkeitsfunktion der Spalten zwischen Basiserzeuger und Konsument

- Ladewahrscheinlichkeit im Basiserzeuger ist:
  - Schmuggelwahrscheinlichkeit oder
  - Referenzierungswahrscheinlichkeit

Schmuggler-Beziehungen



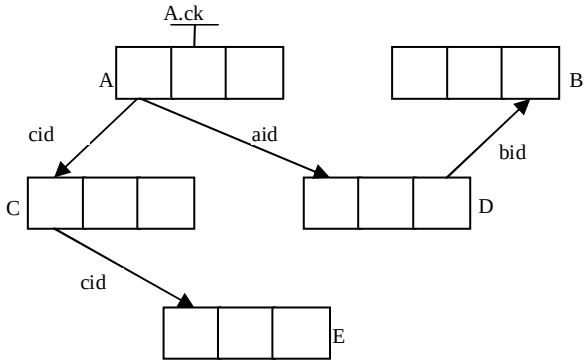
- Schmuggelwahrscheinlichkeit, Einflußgrößen:
  - Mächtigkeit der Schmuggelumgebung
  - Häufigkeit (Histogrammwert)
  - Kardinalität der Spalte
- Berechnung der Schmuggelwahrscheinlichkeit, Probleme; Annäherung mit Hilfe von:
  - Stirling-Formel
  - rekursive Darstellung (iterativ berechnen), Interpolation

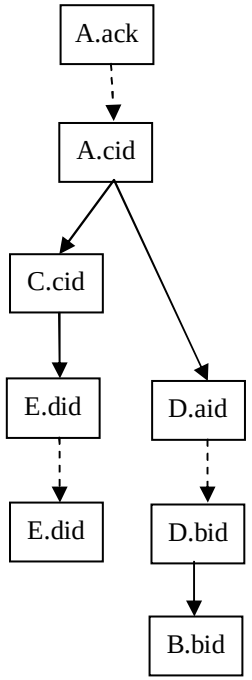
Baumartige Cache-Groups

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
FACHBEREICH INFORMATIK
DBIS Datenbanken und Informationssysteme

Eigenschaften

- Spalten-Beziehungs**baum**
- in jeder Cache-Table maximal ein Konsument, in den werden Werte induziert, in die anderen Spalten geschmuggelt
- maximal ein zusammenfassenden RCC-Füllschritt pro Cache-Tabelle
- Jeder Wertemengen-Pfad ist isoliert
- Die Spalten einer Cache-Table bilden einen Teilbaum





Erster Ansatz (ohne Histogramme)

- Füllstände werden in TSO der Cache-Tabellen berechnet, der Füllstand einer Cache-Tabelle ergibt sich aus:
  - Anzahl der im Vorgänger des Konsumenten geladenen Werte
  - Überdeckung des Konsumenten zu seinem Vorgänger
  - durchschnittliche Häufigkeit eines Werts
- Für jeden Erzeuger wird berechnet, wie viele Werte in ihn geschmuggelt wurden

```
void berechne_Füllstand(Konsument, Anzahl induzierter Werte) {
        berechne die Anzahl der zu ladenden Sätze in T(c).
        Für jeden Erzeuger der Cache-Tabelle {
                berechne Anzahl geschmuggelter Werte (falls Schmuggelziel);
                Für jeden Nachfolger des Erzeugers {
                        berechne Anzahl induzierter Werte (mit Hilfe von
                        Überdeckung);
                        berechne Füllstand seiner Cache-Tabelle rekursiv;
                }
        }
}
berechne_Füllstand(Cache-Key, 1);
```

Einfacher Algorithmus (mit Histogrammen)

Idee: betrachte jeden Wert aus der DBD jedes Konsumenten und berechne mit Hilfe der Ladewahrscheinlichkeit den durchschnittlichen Füllstand

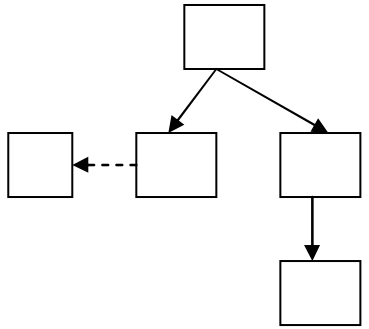
```
Berechne zunächst den Füllstand der Wurzel-Tabelle (mit Hilfe der
Referenzierungswahrscheinlichkeit und des Histogramms des Cache-Keys

Berechne dann die Füllstände der restlichen Cache-Tabellen, betrachte
hierfür jede Cache-Tabelle außer der Wurzel-Tabelle in TSO:

Für jeden Wert aus der DBD des Konsumenten {
          berechne Ladewahrscheinlichkeit
          erhöhe Füllstand der Cache-Tabelle
}
```

Steigerung der Effizienz

- Problem: jeder Wert in der DBD jedes Erzeugers muss einmal „angefaßt“ werden.
- Aber: jeder Wert „stammt“ aus einem Basiserzeuger und verbreitet sich über Wertemengen-Pfade
- Idee: betrachte die Ladewahrscheinlichkeit jedes Werts aus der DBD jedes Basiserzeugers. Dieser Wert „verbreitet“ sich über Wertemengen-Pfade, man kann also die Füllstände der Cache-Tabellen auf einem „RCC-Teilbaum“ betrachten.

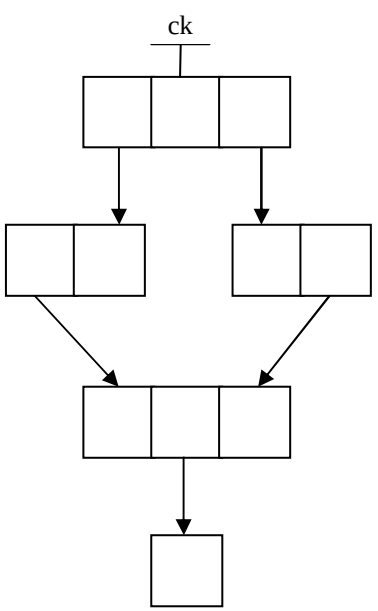


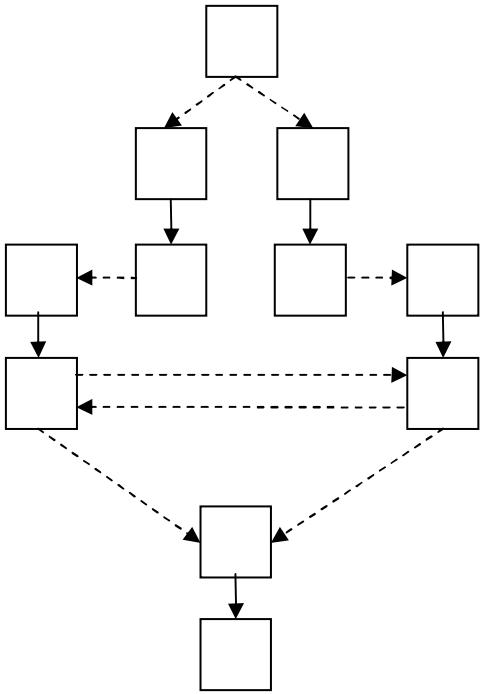
nicht-baumartige, RCC-zyklenfreie Cache-Groups

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
FACHBEREICH INFORMATIK
DBIS Datenbanken und Informationssysteme

Eigenschaften

- Konsumenten/Basiserzeuger nicht mehr eindeutig bestimmt
- mehr als ein RCC-Füllschritt pro Cache-Tabelle möglich
- Cache-Tabellen topologisch sortierbar





Berechnung der Füllstände

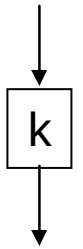
- Füllstände werden in TSO der Cache-Tabellen berechnet.
- Cache-Tabelle führt für jede eingehende RCC einen zusammenfassenden RCC-Füllschritt aus.Modelliere diese Füllschritte als unabhängige Ereignisse.

```
Für jede Cache-Tabelle in topologischer Suchordnung {
        Wenn es in T keine Konsumenten gibt (es handelt sich um die
        Wurzel-Tabelle):
                Berechne die Anzahl der Sätze, wenn ein Cache-Key-Wert in
                den Cache-Key ck geladen wird;
        Ansonsten:
                Mache für jede eingehende RCC einen RCC-Füllschritt
                (betrachte diese als unabhängige Ereignisse) und berechne
                entsprechend den Füllstand;
}
```

Probleme und Ungenauigkeiten

- Abhängigkeiten zwischen ankommenden Wertemengen werden vernachlässigt
- Wir verwenden Gleichverteilungsannahme (keine Histogramme)
- Zurückrechnen von der Anzahl der geladenen Sätze auf die Anzahl der geladenen Werte in den Erzeugern (auch wenn dieser ein Konsument ist)

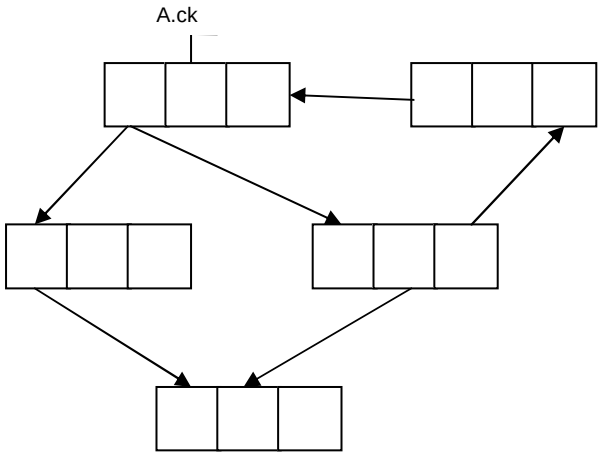
- Beispiel: betrachte Cache-Tabelle mit einem Konsumenten k und einer eingehenden RCC. Diese Cache-Tabelle führt im Prinzip drei Schritte aus:
  - Abschätzung der in k induzierten Werte
  - Abschätzung der geladenen Sätze
  - Abschätzung der in k geschmuggelten Werte (wir betrachten k als Schmuggelziel)

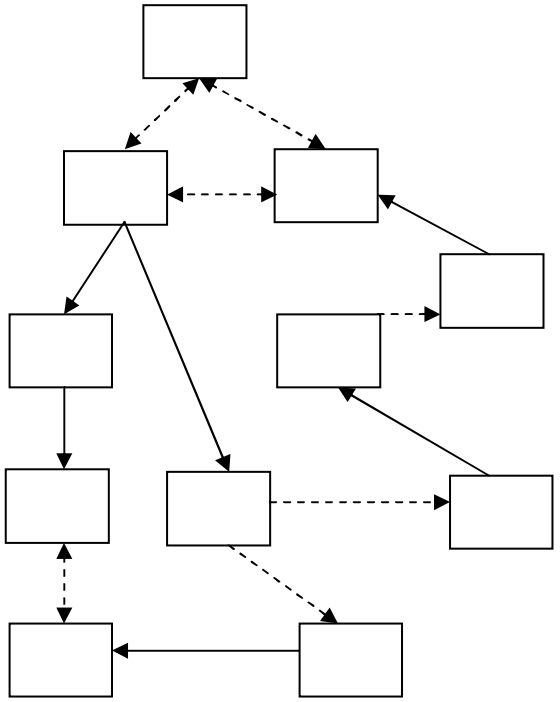


allgemeine Cache-Groups

Eigenschaften

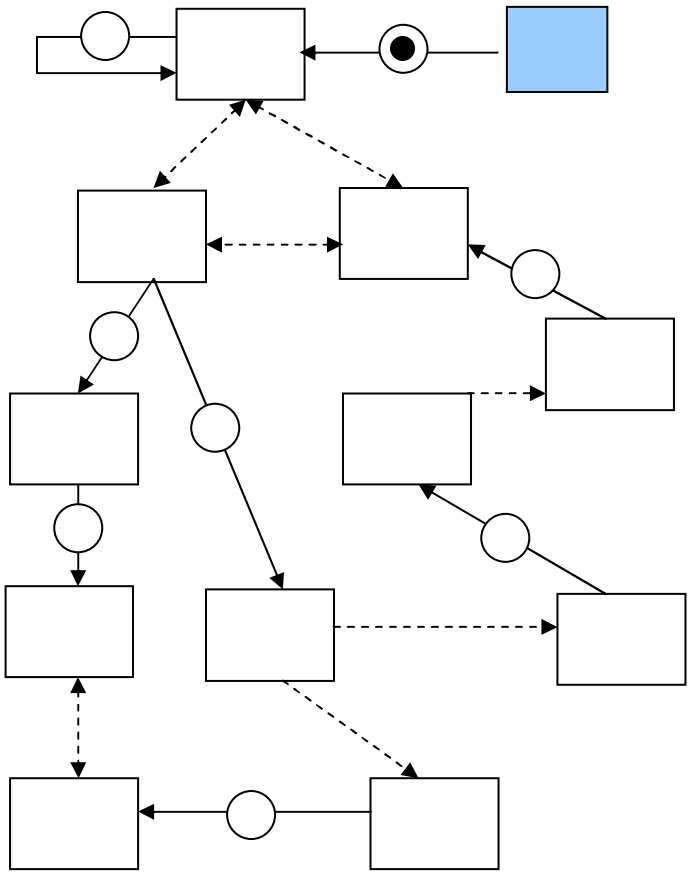
- Konsumenten/Basiserzeuger nicht mehr eindeutig bestimmt
- mehr als ein RCC-Füllschritt pro Cache-Tabelle möglich
- Cache-Tabellen nicht mehr topologisch sortierbar



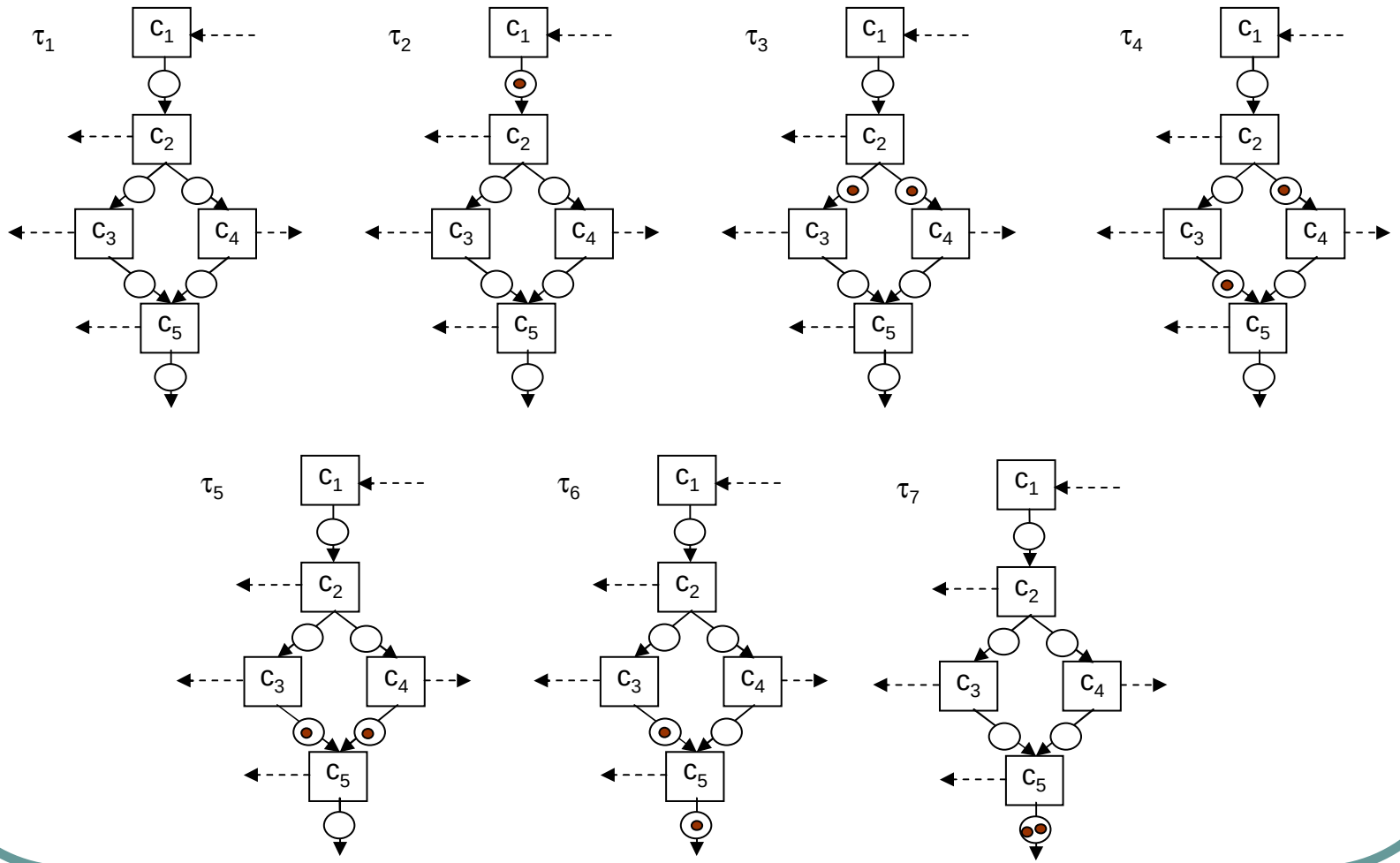


Token und Stellen

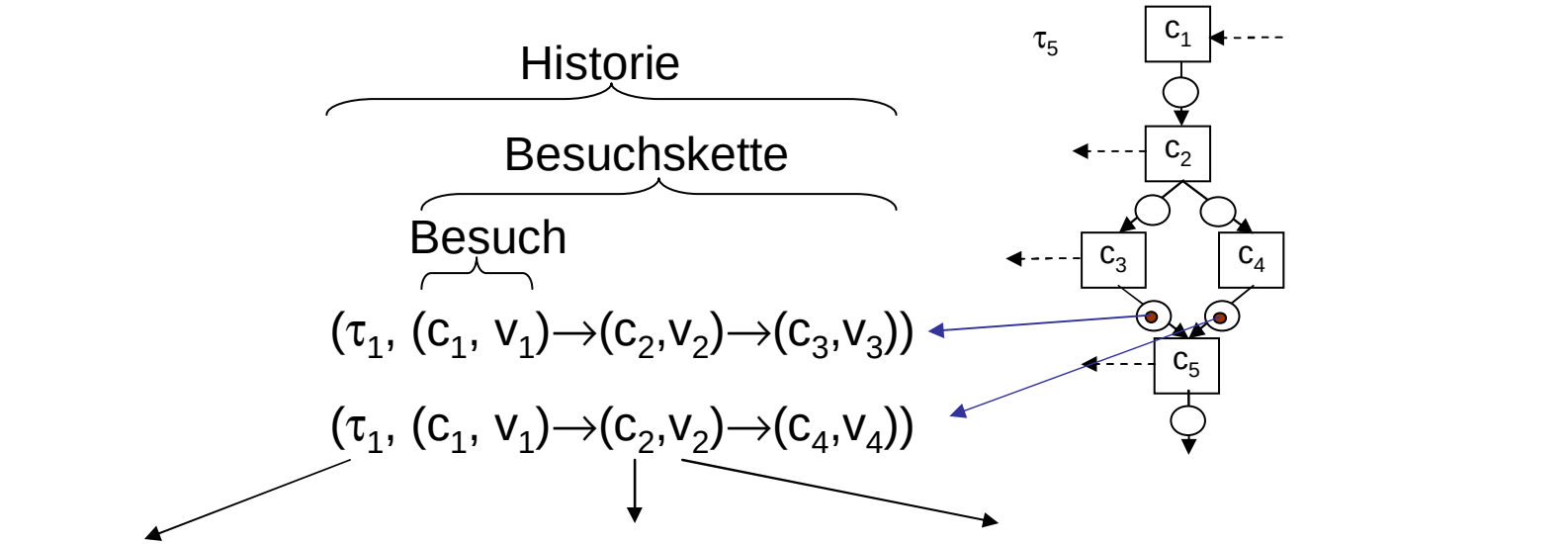
- Idee: Modellierung durch Netze, Inspiration durch Petrinetze
- „fließende“ Werte werden durch Token repräsentiert, die in Stellen gespeichert werden
- Auswertung des Modells entspricht Situation des Token-Stellen-Systems

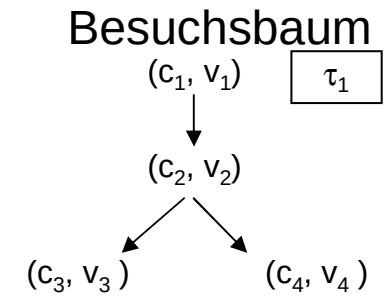


Fluß eines Tokens



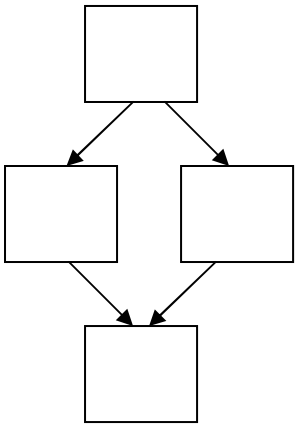
Semantik der Token – Besuche und Historien

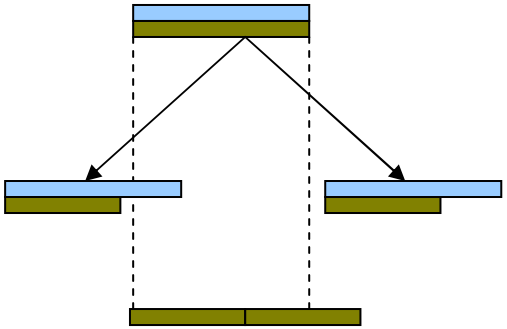




Kontexte und Kontext-Bäume

- Wie viele der „dort“ geladenen Werte wurden „hier“ induziert?
- Zur Identifizierung von Abhängigkeiten
- Kontext-Baum: Besuchsbäumen mit „Füllständen“
- Kontext-Bäume entstehen dadurch, dass Token-Historien „integriert“ werden





Ablauf eines Füllschritts

- „Konsumieren“ des Tokens
- Prüfung, ob Erzeugungsspalte aktiv ist (homogener Zyklus)
- Berechnung der zusätzlich zu ladenden/zusätzlich zu induzierenden Werte
- Berechnung der Anzahl der zu ladenden Sätze
- Bei ausgehenden RCCs: Verlängerung der Besuchskette, Duplikation und Weitergabe des Tokens
- Integration des Tokens und Update der „Statistik“
- Berechnung der geschmuggelten Werte in die Schmuggelziele
- Bei ausgehenden RCCs am Schmuggelziel: Generierung jeweils eines neuen Tokens

Ein Füllzyklus

- Zurücksetzen der Statistiken
- Einführung des initialen Tokens in die Stelle zwischen Kontrollspalte und Cache-Key
- es werden solange Token verarbeitet und Füllschritte modelliert, bis keine Token mehr im System sind
- Auswertung der Statistiken (hauptsächlich Füllstände der Cache-Tabellen)
- eventuell Wiederholung für einen anderen Cache-Key/Bildung von Mittelwerten

Vielen Dank für Ihre Aufmerksamkeit !