Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Access Path Selection in a Relational Database Management System

Gliederung

- Einführung
- Zugriffswege für eine Relation
  - Einsatz von Statistiken / Selektivitätsfaktoren
  - Kostenformeln für eine Relation
- Auswahl der Zugriffswege bei Joins
  - Wo kann man Kosten sparen - Optimierungen?
  - Berechnung der Kosten
- Zusammenfassung

Einführung

- System R
  - IBM, San Jose (Almaden) Research Center, Kalifornien
  - 1975, erstes relationales Datenbankmanagementsystem
  - reines Forschungsobjekt

- *Access Path Selection in a Relational Dabase Manament System*
  - 1979, International Conference on Management of Data

P. G. Sellinger

D. D. Chamberlin

R. A. Lorie

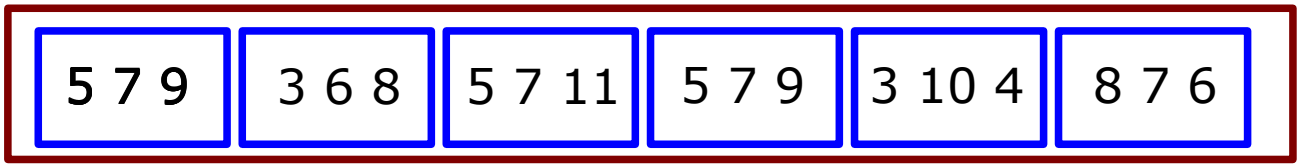
?

M. M. Astrahan

?

T. G. Price

Zugriffswege für eine Relation

- Wiederholung:
  - Abarbeitung einer Anfrage
    - Parsen, Optimierung, Codegenerierung, Ausführung
  - 

  - 2 grundsätzliche Zugriffe
    - Index Scan vs. Segment Scan

Zugriffswege für eine Relation - Statistiken

- Für jede Relation T
  - NCARD(T) – Kardinalität der Relation T
  - TCARD(T) – Anzahl pages im Segment die Tupel von T enthalten
  - P(T) – der Bruchteil pages im Segment mit Tupel von T
    P(T) = TCARD(T) / #nichtleere pages im Segment

- Für jeden Index I für die Relation T
  - ICARD(I) – Anzahl verschiedener Schlüssel im Index I
  - NINDX(I) – Anzahl pages im Index I

Zugriffswege für eine Relation - Selektivitätsfaktoren

- Feld = Wert
  - F = 1/ICARD(Feld Index) – wenn Index auf Feld (Annahme: Gleichverteilung der Werte)
  - Sonst: F = 1/10
- Feld1 = Feld2
  - F = 1/MAX(ICARD(Feld1 Index), ICARD(Feld2 index)) – wenn Indizes auf beiden Feldern
  - F = 1/ICARD(Feld-i Index), wenn Index nur auf Feld-i
  - Sonst: F = 1/10
- Feld in (Liste von Werten)
  - F = #Anzahl Werte in Liste * (Selektivitätsfaktor für Feld=Wert)

Zugriffswege für eine Relation - Kosten

- Interesting order: Ordnung der Tupel spezifiziert durch *order by* oder *group by*

Kosten = geladene Seiten + W * #Tupel

- I/O Kosten + CPU Kosten
- W ist Gewichtung zwischen beiden Kosten

- Berechnung der Kosten für jeden möglichen Zugriffsweg *(*Kosten + Ordnung der Tupel)
- Ermittlung der Kosten für:
  - günstigsten Zugriffsweg, unter Beachtung aller *Interesting orders*
  - günstigsten Zugriffsweg ohne *Interesting orders* + Sortierung

Zugriffswege für eine Relation – Kosten

Unique Index, Übereinstimmung in einem booleschen Faktor	1 + 1 + W * 1
Clustered Index, Übereinstimmung in mehreren booleschen Faktoren	F(preds) * (NINDX(I) + TCARD) + W * #Tupel
Nicht-Clustered Index, Übereinstimmung in mehreren booleschen Faktorenn	F(preds) * (NINDX(I) + NCARD) + W * #Tupel

Zugriffswege für eine Relation – Kosten

Segment Scan	#pages(Segment) + W * #Tupel
Clustered Index, keine Übereinstimmung mit booleschen Faktoren	(NINDX(I) + TCARD) + W * #Tupel
Nicht-Clustered Index, keine Übereinstimmung mit booleschen Faktoren	(NINDX(I) + NCARD) + W * #Tupel

Zugriffswege bei Joins

- Nested Loop Join vs. Merge Join
- Bottem up – Aufbau eines Baumes mit möglichen Joins
  - Ermittlung der günstigsten Zugriffe für jede Relation, für jede *Interesting Order* und für den ungeordneten Fall
  - Ermittlung der günstigsten Joins für jede Relation mit anderen Relationen
  - Ermittlung der günstigsten Joins für jedes Paar von Relationen mit anderen Relationen
- Kreuzprodukte möglichst vermeiden, nur Relationen joinen mit passendem Join-Attribut – Verkleinerung des Suchbaumes
- Lösung beinhaltet Reihenfolge der Relationen, die jeweilige Join-Methode, Zugriffsweg für jede Relation

Zugriffswege für Joins – Kosten für Joins

C-nested-loop(path1, path2) = C-outer(path1) +
N* C-inner(path2)

C-merge(path1, path2) = C-outer(path1) +
N* C-inner(path2)

C-inner(sorted list) = Temppages + N* W * RSICARD

- Prinzipiell gleiche Kosten, aber innerer merge-Scan meist günstiger

Zugriffswege für Joins - Beispiel

EMP

NAME	DNO (I)	JOB (I)	SAL
Smith	50	12	8500
Jones	50	5	15500
Doe	51	5	9500

DEPT

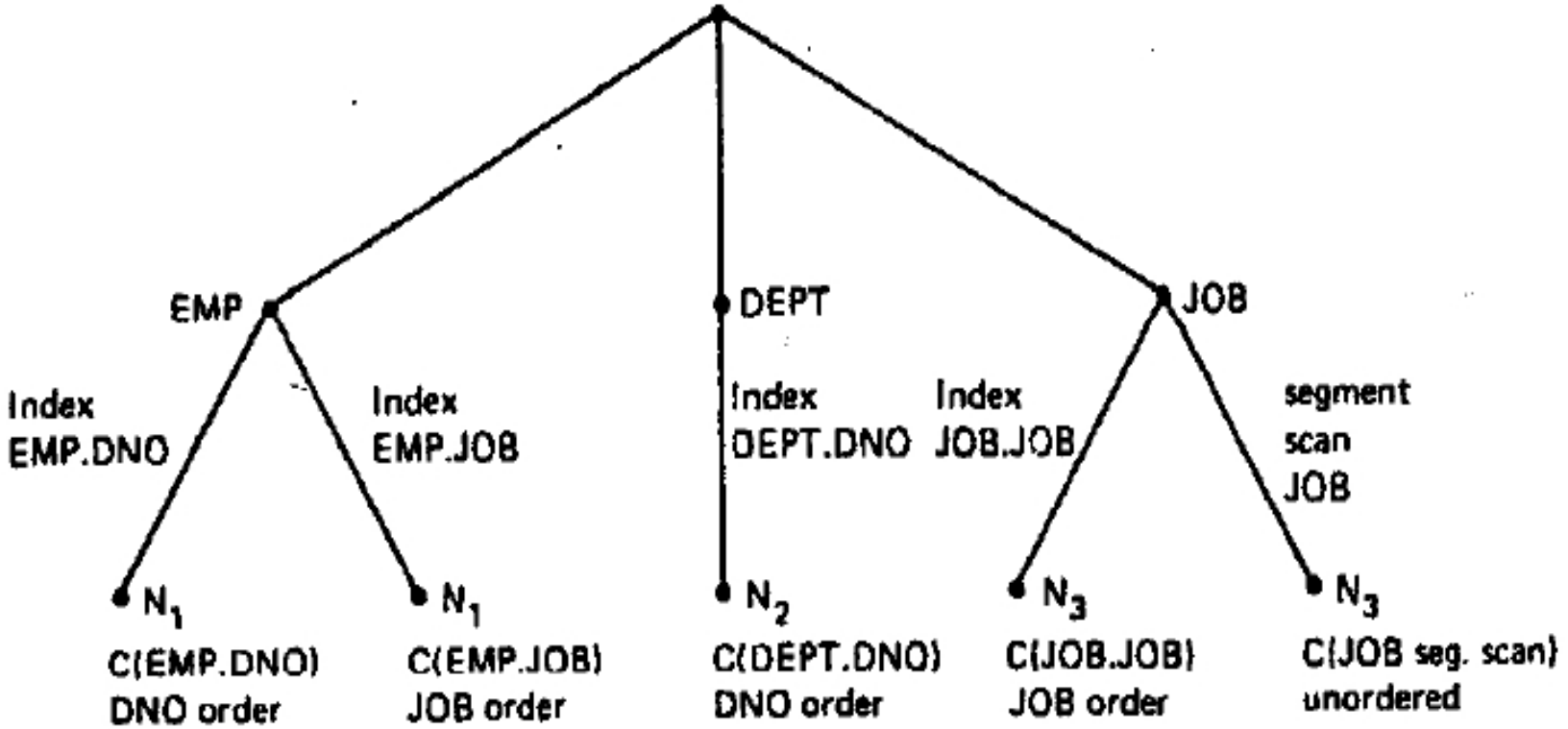
DNO (I)	DNAME	LOC
50	MFG	Denver
51	Billing	Boulder
52	Shipping	Denver

JOB

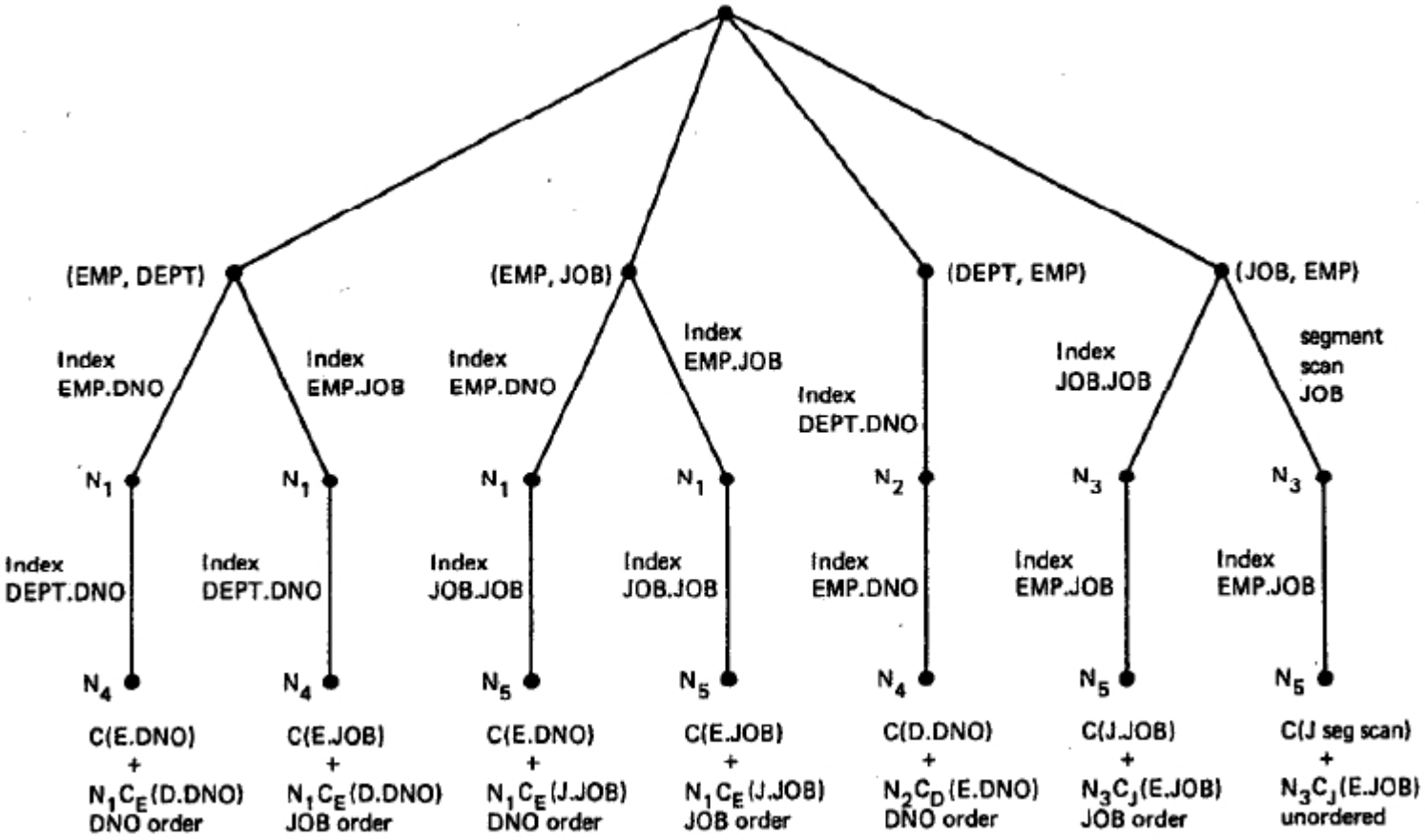
JOB (I)	TITLE
5	CLERK
6	TYPIST
9	SALES
12	MECHANIC

```
SELECT NAME, TITLE, SAL, DNAME FROM EMP, DEPT, JOB WHERE
TITLE='CLERK' AND LOC='DENVER' AND EMP.DNO=DEPT.DNO AND
EMP.JOB=JOB.JOB
```

Zugriffswege für Joins - Beispiel

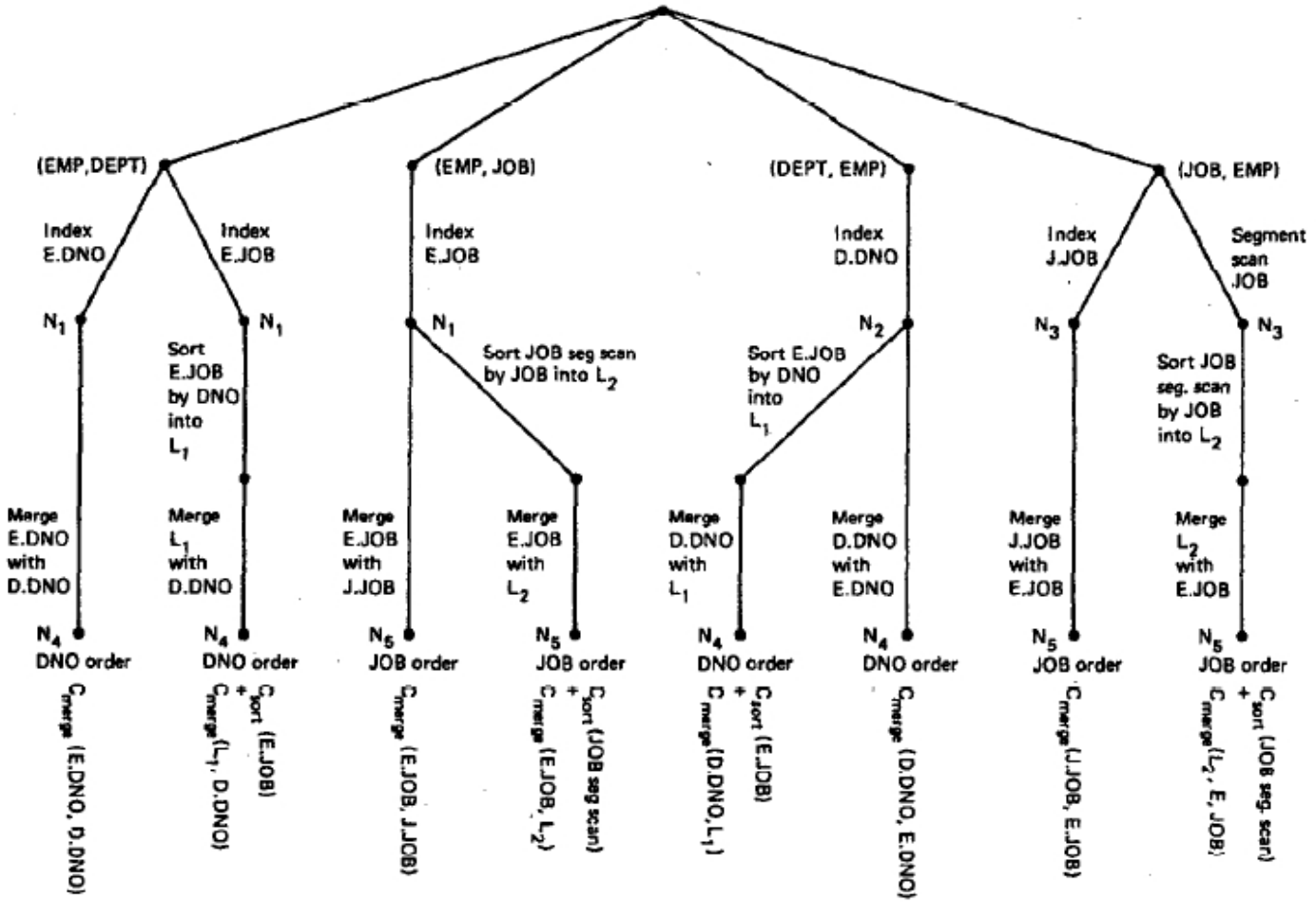


Zugriffswege für Joins - Beispiel



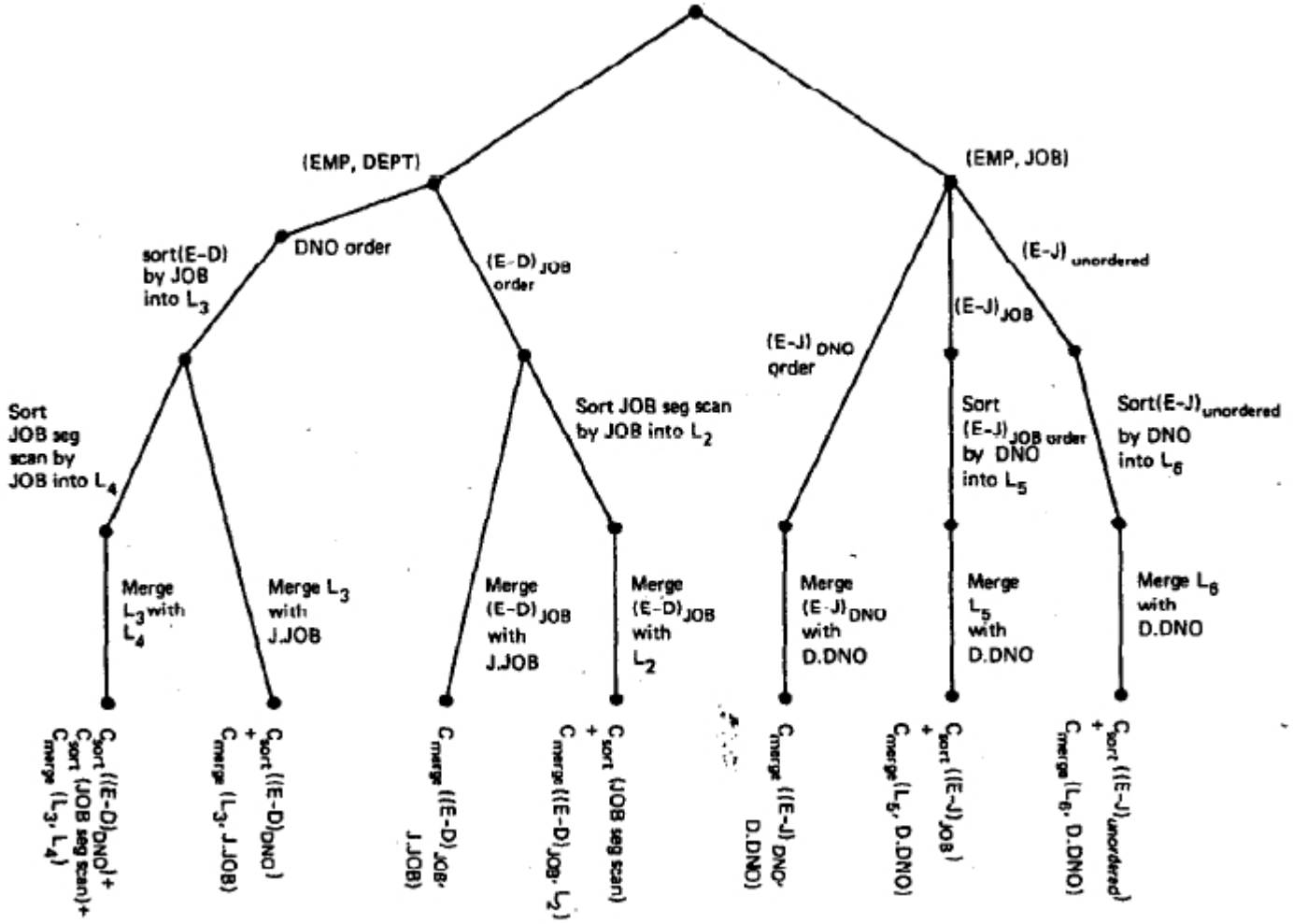
- Erweiterter Suchbaum - nested loop

Zugriffswege für Joins - Beispiel



- Erweiterter Suchbaum – merge

Zugriffswege für Joins - Beispiel



Zusammenfassung

- Einsatz von Selektivitätsfaktoren und Statistiken um Schätzungen treffen zu können
  - daraus resultierende Kostenformulare
- Bottem up Strategie – Aufbau eines Suchbaumes
  - Heuristiken um Suchbaum zu verkleinern
- Beachtung aller *Interesting Orders,* daher evt. für einen Join mehr als nur eine Zwischenlösung
- Ermittlung der günstigsten Lösung:
  - Reihenfolge,Join-Methoden, Zugriffswege,
- → Datenbanksysteme mit deskriptiven Anfragesprachen können performant arbeiten (8 Tabellen Joinoptimierung in wenigen Sek.)