Query Optimization: Exercise

Session 7

Bernhard Radke

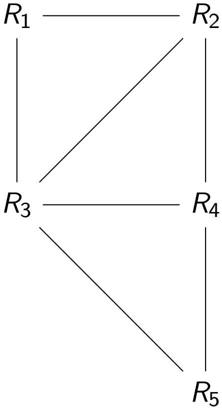
December 3, 2018

DPccp

- standard DP algorithms consider many pairs that are discarded due to disconnectedness or failing disjointness tests
- taking the query graph into account avoids this [2]

- Nodes in the query graph are ordered according to a BFS
- Start with the last node, all the nodes with smaller ID are forbidden
- At every step: compute neighborhood, get forbidden nodes, enumerate subsets of non-forbidden nodes N
- Recursive calls for subsets of N

- EnumerateCsg+EnumerateCmp produce all ccp
- resulting algorithm DPccp considers exactly #ccp pairs
- which is the lower bound for all DP enumeration algorithms


R_1
R_2
R_3
R_4
R_5

Hypergraphs

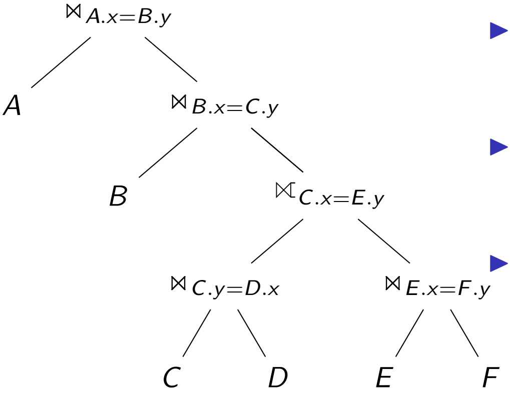
- Complex Join Predicates
- Reordering constraints due to non-inner Joins
- $\Rightarrow$ DPHyp [3]
- Edges connect sets of nodes
- Similar to DPccp but hyperedges 'lead' to the smallest edge within the set
- Subproblems may be disconnected

Query Simplification

Simplify the query graph if it is too complex [4].

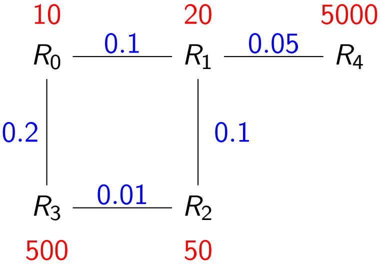
- GOO: greedily choose joins to perform
- Simplification: greedily choose joins that must be avoided (we can start with 'obvious' decisions)
- $\mathit{benefit}(X \bowtie R_1, X \bowtie R_2) = \frac{C((X \bowtie R_1) \bowtie R_2)}{C((X \bowtie R_2) \bowtie R_1)}$

Homework

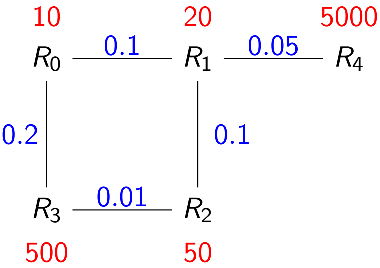


- Syntactic eligibility set - relations that have to be in the input
- Total eligibility set - captures also reordering restrictions, construct bottom-up
- Conflicts: $⟕_{C.x=E.y}$ and $\bowtie_{C.y=D.x}$, $⟕_{C.x=E.y}$ and $\bowtie_{B.x=C.y}$, $\bowtie_{A.x=B.y}$ and $⟕_{C.x=E.y}$

Important: consider all possible edge combinations, that is, $benefit(R_0 \bowtie R_1, R_0 \bowtie R_2)$ together with $benefit(R_0 \bowtie R_2, R_0 \bowtie R_1)$



- $benefit(R_0 \bowtie R_1, R_0 \bowtie R_3) = \frac{202}{300}$
- $b(R_0 \bowtie R_3, R_0 \bowtie R_1) = 300/202$
- $b(R_1 \bowtie R_2, R_1 \bowtie R_0) = 20/12$
- $b(R_3 \bowtie R_0, R_3 \bowtie R_2) = 2$
- $b(R_2 \bowtie R_3, R_2 \bowtie R_1) = 5/4$
- $b(R_1 \bowtie R_4, R_1 \bowtie R_0) = 500/251$
- $b(R_1 \bowtie R_4, R_1 \bowtie R_3) = 300/251$
- $R_3 \bowtie R_2$ before $R_3 \bowtie R_0$.
  Replace $R_0 - R_3$ by $\{R_0\} - \{R_2, R_3\}$



- $benefit(R_0 \bowtie R_1, R_0 \bowtie R_3) = \frac{202}{300}$
- $b(R_0 \bowtie R_3, R_0 \bowtie R_1) = 300/202$
- $b(R_1 \bowtie R_2, R_1 \bowtie R_0) = 20/12$
- $b(R_2 \bowtie R_3, R_2 \bowtie R_1) = 5/4$
- $b(R_1 \bowtie R_4, R_1 \bowtie R_0) = 500/251$
- $b(R_1 \bowtie R_4, R_1 \bowtie R_3) = 300/251$
- $b(R_0 \bowtie (R_3 \bowtie R_2), R_0 \bowtie R_1) = \frac{C((R_0 \bowtie (R_3 \bowtie R_2)) \bowtie R_1)}{C((R_0 \bowtie R_1) \bowtie (R_3 \bowtie R_2))} = 850/370$
- $b((R_2 \bowtie R_3) \bowtie R_0, R_2 \bowtie R_1) = \frac{C(((R_2 \bowtie R_3) \bowtie R_0) \bowtie R_1)}{C(((R_2 \bowtie R_3) \bowtie R_1) \bowtie R_0)} = 1$
- $R_0 \bowtie R_1$ before $R_0 \bowtie (R_3 \bowtie R_2)$. Replace $\{R_0\} - \{R_2, R_3\}$ by $\{R_0, R_1\} - \{R_2, R_3\}$

Next Homework

- implement DP (either DPsize or DPsub)

- Slides: db.in.tum.de/teaching/ws1819/queryopt
- Exercise task: gitlab
- Questions, Comments, etc:
  mattermost @ mattermost.db.in.tum.de/qo18
- Exercise due: 9 AM next monday

[1] G. Moerkotte, P. Fender, and M. Eich.
On the correct and complete enumeration of the core search space.
In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 493–504, 2013.

[2] G. Moerkotte and T. Neumann.
Analysis of two existing and one new dynamic programming algorithm for the generation of optimal bushy join trees without cross products.
In *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*, pages 930–941, 2006.

[3] G. Moerkotte and T. Neumann.
Dynamic programming strikes back.
In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 539–552, 2008.

[4] T. Neumann.
Query simplification: graceful degradation for join-order optimization.
In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*, pages 403–414, 2009.