Logical Approach to Physical Data Independence and Query Compilation

Future Directions

David Toman

D.R. Cheriton School of Computer Science

University of Waterloo
Research Directions and Open Problems

- computationally better query operators
  - ⇒ merge-join via *finger* data structures
  - ⇒ intermediate results via additional physical design (incremental compilation?)
  - ⇒ general *physical design* adviser (ala index adviser)
Research Directions and Open Problems

1. Computationally better query operators
   ⇒ merge-join via *finger* data structures (√)
   ⇒ intermediate results via additional physical design
     (incremental compilation?)
   ⇒ general physical design adviser (ala index adviser)

2. Computationally better schema language (e.g., DLs)
   ⇒ what to do about queries? (parametric analysis?)

3. High performance query compilation

4. Diagnostic features/explanations (why doesn’t this query compile?)

5. Additional features
   ⇒ (better) updates
   ⇒ transaction management (locking, recovery)

6. Beyond FOL: synthesis of iterators for APs (e.g., inductive types)
   ⇒ interaction with interpolation/query compilation?
Research Directions and Open Problems

1. Computationally better query operators
   ⇒ merge-join via *finger* data structures (√)
   ⇒ intermediate results via additional physical design
   (incremental compilation?)
   ⇒ general *physical design* adviser (ala index adviser)

2. Computationally better schema language (e.g., DLs)
   ⇒ what to do about queries? (parametric analysis?)
Research Directions and Open Problems

1. Computationally better query operators
   ⇒ merge-join via *finger* data structures (√)
   ⇒ intermediate results via additional physical design
      (incremental compilation?)
   ⇒ general *physical design* adviser (ala index adviser)

2. Computationally better schema language (e.g., DLs)
   ⇒ what to do about queries? (parametric analysis?)

3. High(re) performance query compilation

4. Diagnostic features/explanations (why doesn’t this query compile?)

5. Additional features
   ⇒ (better) updates
   ⇒ transaction management (locking, recovery)

6. Beyond FOL: synthesis of iterators for APs (e.g., inductive types)
   ⇒ interaction with interpolation/query compilation?
1. Computationally better query operators
   ⇒ merge-join via *finger* data structures (√)
   ⇒ intermediate results via additional physical design (incremental compilation?)
   ⇒ general *physical design* adviser (ala index adviser)

2. Computationally better schema language (e.g., DLs)
   ⇒ what to do about queries? (parametric analysis?)

3. High(re) performance query compilation

4. Diagnostic features/explanations (why doesn’t this query compile?)
Research Directions and Open Problems

1. Computationally better query operators
   - Merge-join via 
     "finger" data structures (√)
   - Intermediate results via additional physical design
     (incremental compilation?)
   - General physical design adviser (ala index adviser)

2. Computationally better schema language (e.g., DLs)
   - What to do about queries? (parametric analysis?)

3. High(re) performance query compilation

4. Diagnostic features/explanations (why doesn’t this query compile?)

5. Additional features
   - (Better) updates
   - Transaction management (locking, recovery)
Research Directions and Open Problems

1. Computationally better query operators
   ⇒ merge-join via finger data structures (√)
   ⇒ intermediate results via additional physical design (incremental compilation?)
   ⇒ general physical design adviser (ala index adviser)

2. Computationally better schema language (e.g., DLs)
   ⇒ what to do about queries? (parametric analysis?)

3. High(re) performance query compilation

4. Diagnostic features/explanations (why doesn’t this query compile?)

5. Additional features
   ⇒ (better) updates
   ⇒ transaction management (locking, recovery)

6. Beyond FOL: synthesis of iterators for APs (e.g., inductive types)
   ⇒ interaction with interpolation/query compilation?