Foundations of Query Languages

Dr. Fang Wei

Lehrstuhl für Datenbanken und Informationssysteme
Universität Freiburg

SS 2011

BCQ Evaluation

Intuition: the problem with cycles in the query
Consider the queries:

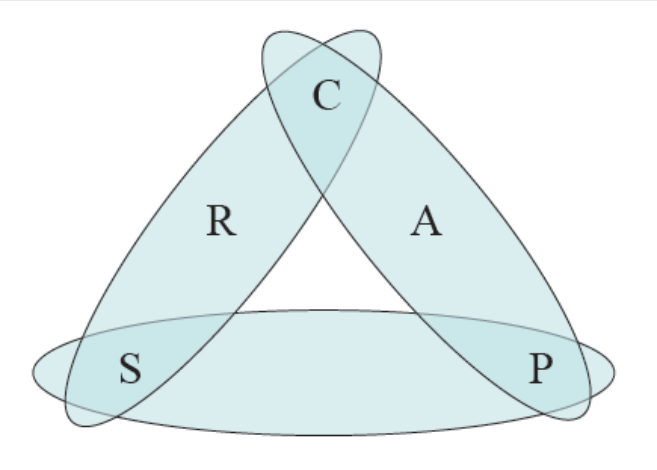
$$Q_1 : ans \leftarrow e(A, B), e(B, C), e(C, D), e(D, A)$$

$$Q_2 : ans \leftarrow e(A, B), e(B, C), e(C, D), e(D, E)$$

In processing Q_1, maintain intermediate join results $e'(A, B, C, D) \bowtie e(D, A)$.
In processing Q_2, all the join operations are only between two predicates (locally).

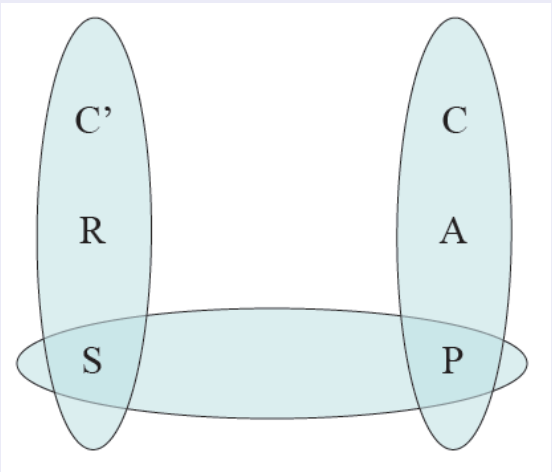
BCQ Evaluation

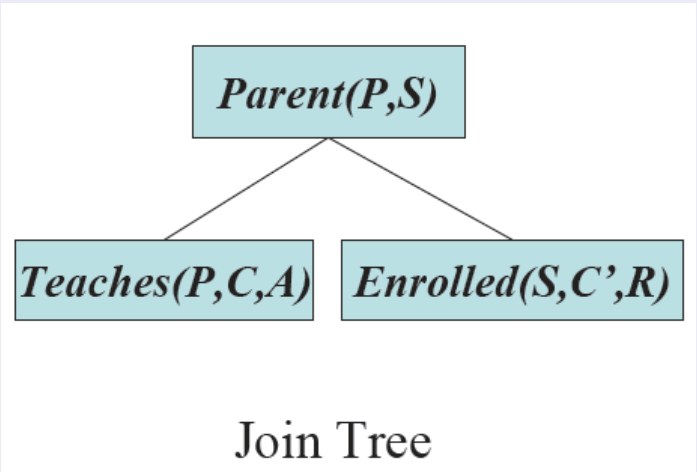
$$ans \leftarrow Enrolled(S, C, R), Teaches(P, C, A), Parent(P, S).$$



BCQ Evaluation

$$ans \leftarrow Enrolled(S, C', R), Teaches(P, C, A), Parent(P, S).$$



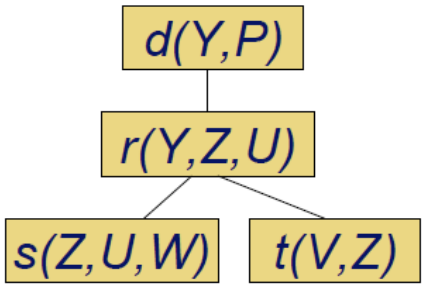


Join tree

Definition

A Join tree of a BCQ is an organisation of its atoms into a tree T such that for each variable X, the nodes in which X occurs span a connected subtreeof T. (Connectedness Condition)

A CQ is a tree query iff it has a join tree.
Example: $Q : ans \leftarrow R(Y, Z, U), s(Z, U, W), d(Y, P), t(V, Z)$

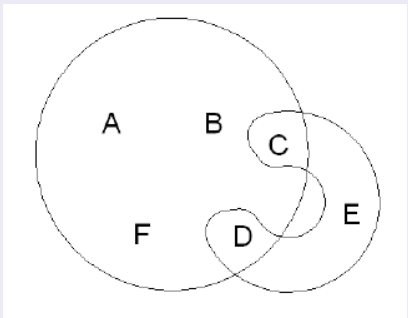


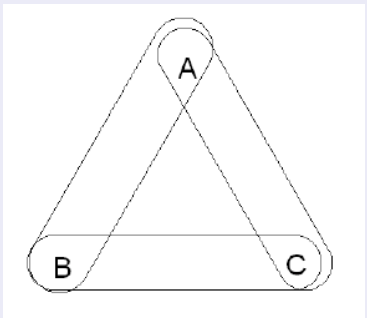
Acyclicity of Hypergraph

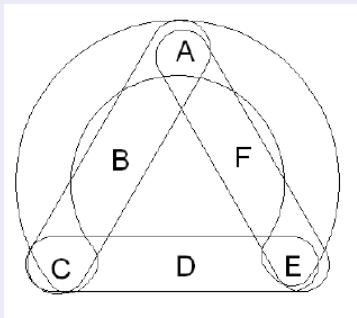
Definition

An *ear edge* (short: ear) of a hypergraph (V, E) is a hyperedge $e \in E$ such that one of the following conditions holds:

1. There is a witness $e' \in E$, such that $e' \neq e$ and each vertex from e is either only in e or in e', or
2. e has no intersection with any other hyperedge.







Acyclicity of Hypergraph: GYO algorithm

Algorithm (GYO Algorithm)
Input: Hypergraph $H = (V, E)$
Output: GYO-reduct of H
While H has ears:
    Choose any ear e of E.
    Set $H := (V', E - \{e\})$, where $V' = \mathit{vertices}(E - \{e\})$.
End While

Acyclicity Theorems

Theorem

A hypergraph is acyclic iff its GYO-reduct empty.

Theorem

A query has a join tree iff it is acyclic.

- The GYO algorithm takes linear time
- Easy algorithms exist fort checking whether a query is acyclic and for computing a join tree.

Acyclic query evaluation

$$ans \leftarrow d(Y, P), t(V, Z), s(Y, Z, U), s(Z, U, W)$$

The polynomial algorithm of acyclic BCQ evaluation:

- Bottom-up
- Semi-join

Semi Join

Semijoin Operation

Employee

Employee Name	Employee Salary	Department Name
Ackman	5000	Car
Baker	4500	Car
Carson	4800	Toy
Davis	5100	Toy

Department

Department Name	Department Budget
Car	450000000

Temp=
Employee semijoin (DepartmentName=DepartmentName)
Department

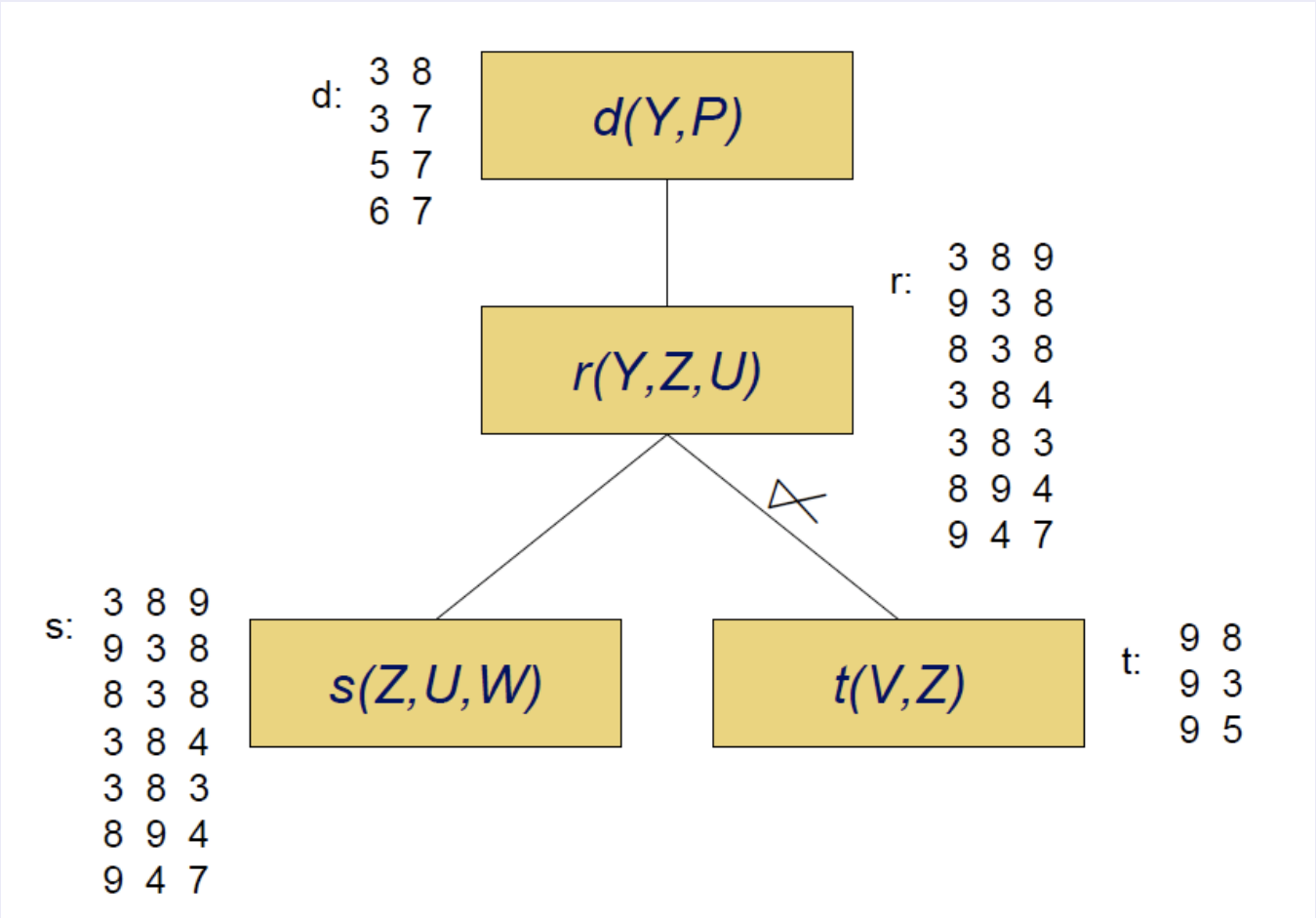
Employee

Employee Name	Employee Salary	Department Name
Ackman	5000	Car
Baker	4500	Car

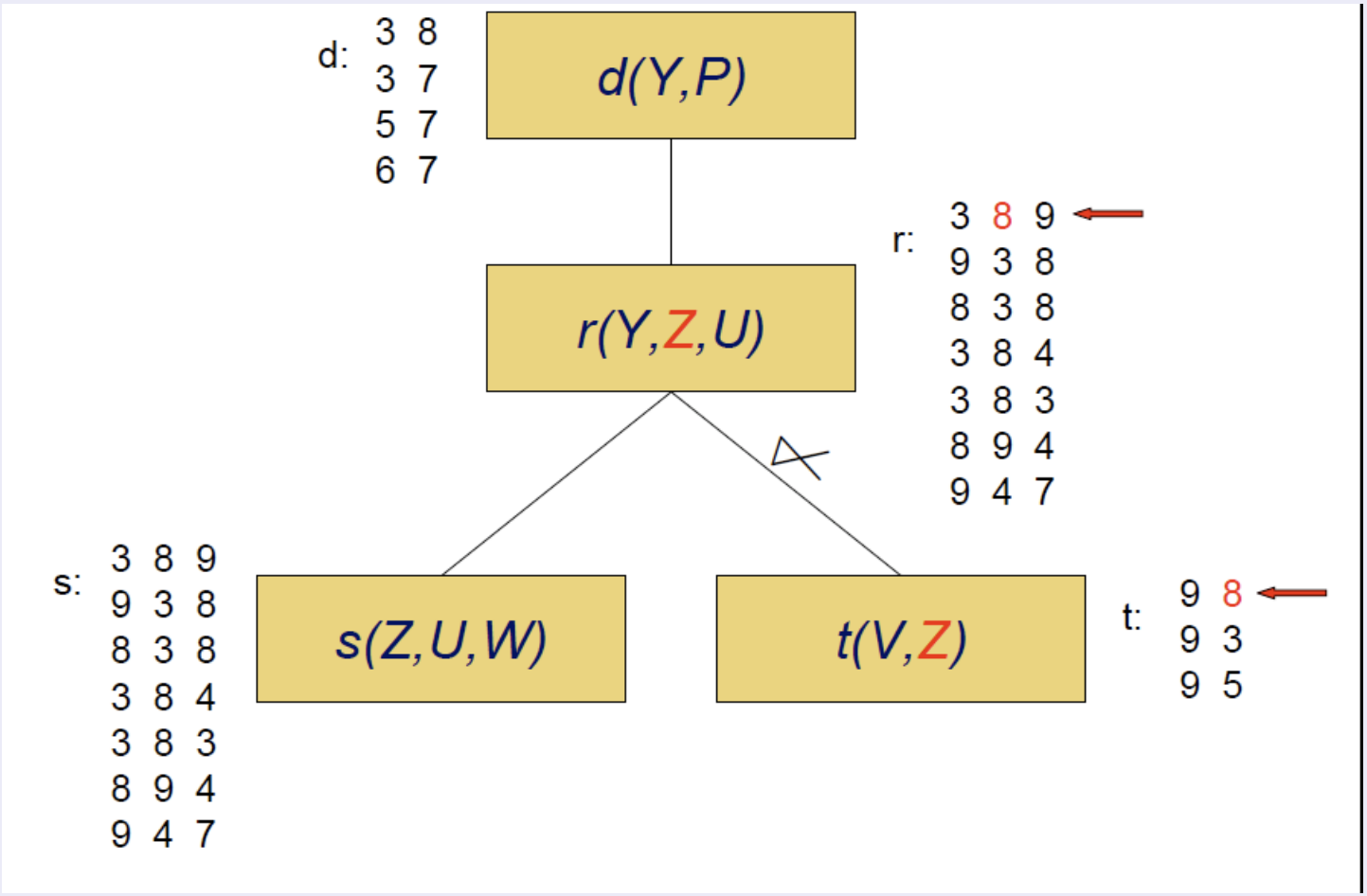
Figure by Dr. James A. Larson

$$Emp \ltimes_{A=B} Dept = \pi_{attr(Emp)}(Emp \bowtie_{A=B} Dept)$$

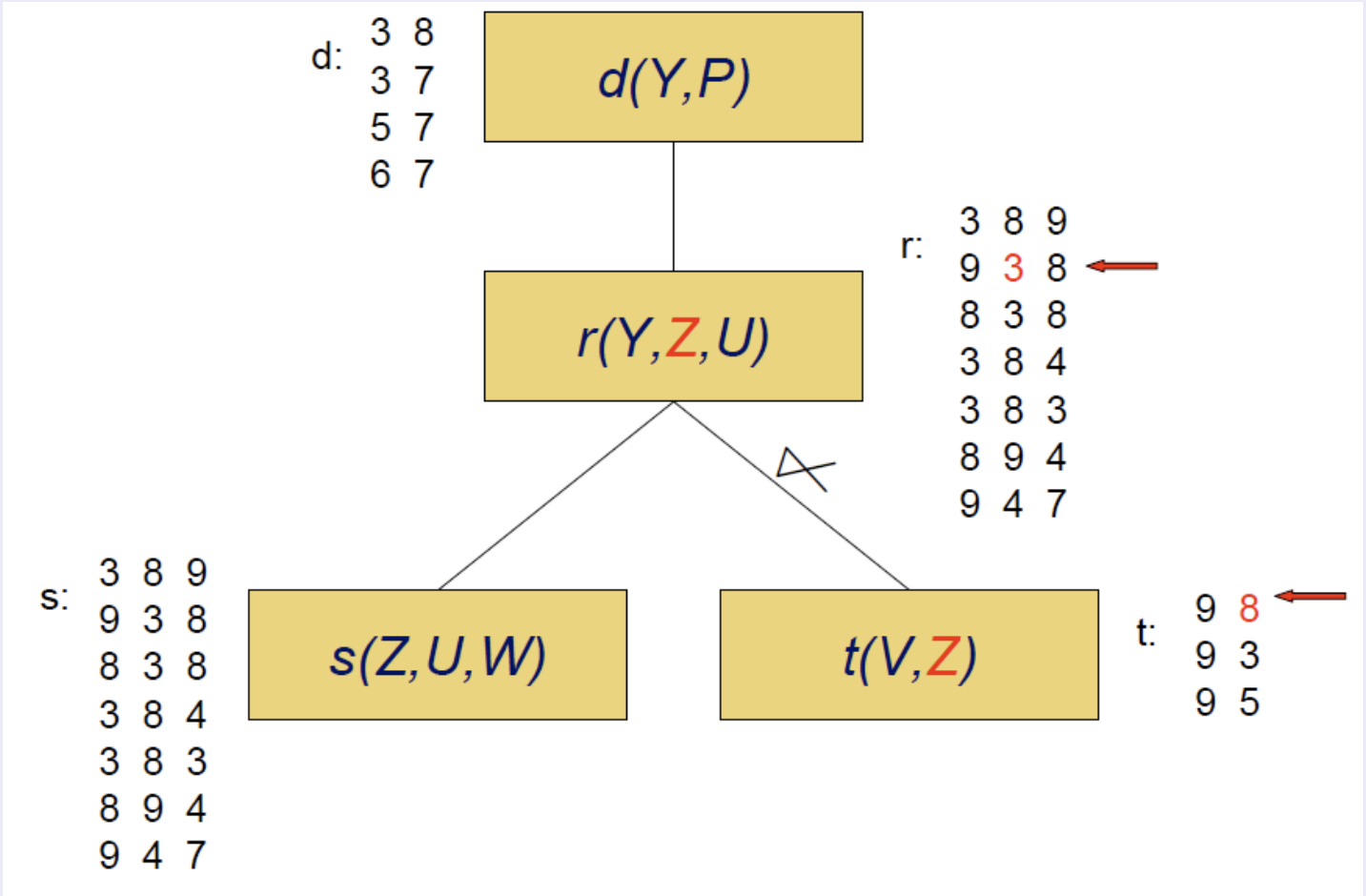
Acyclic Query Evaluation



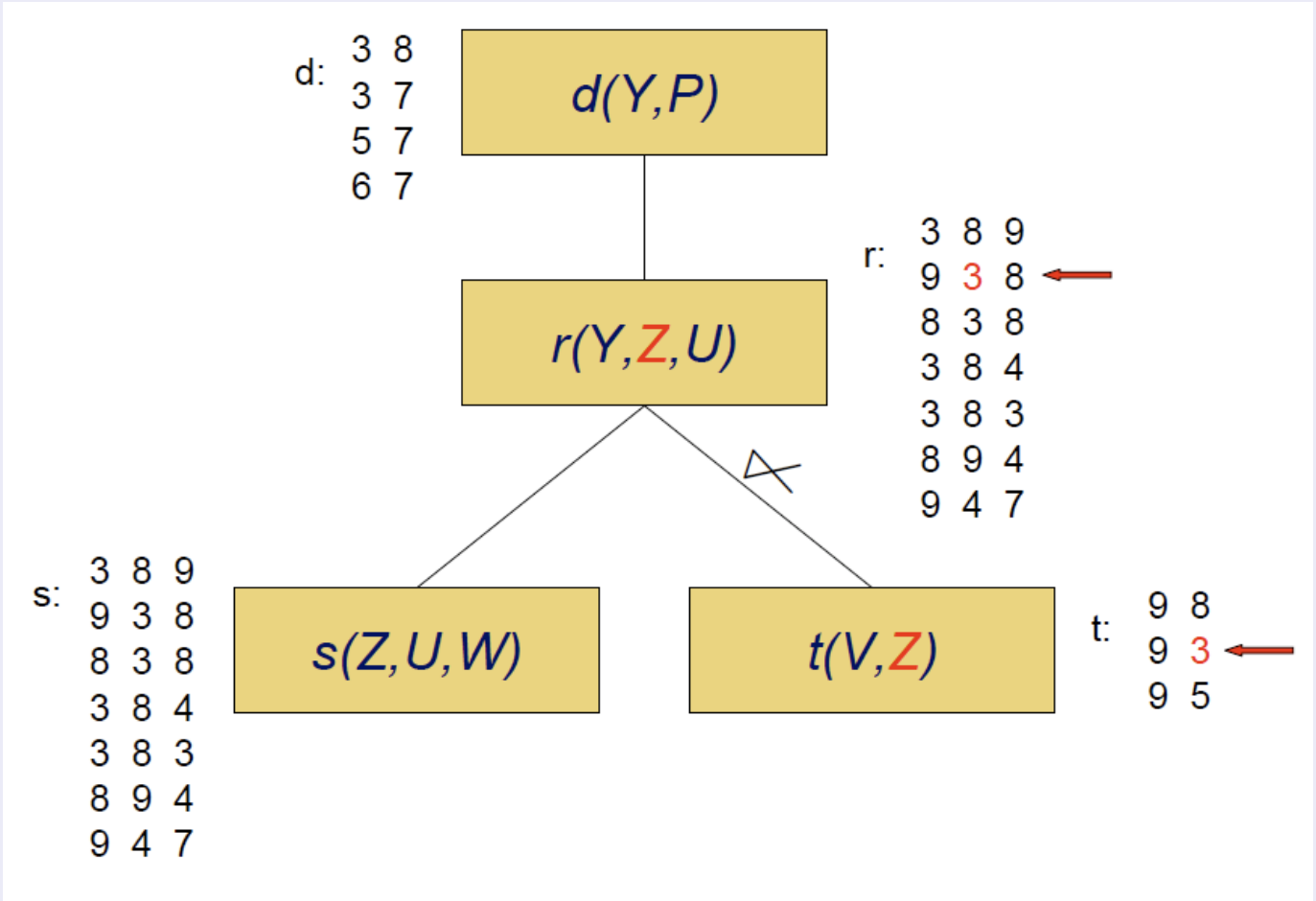
Acyclic Query Evaluation



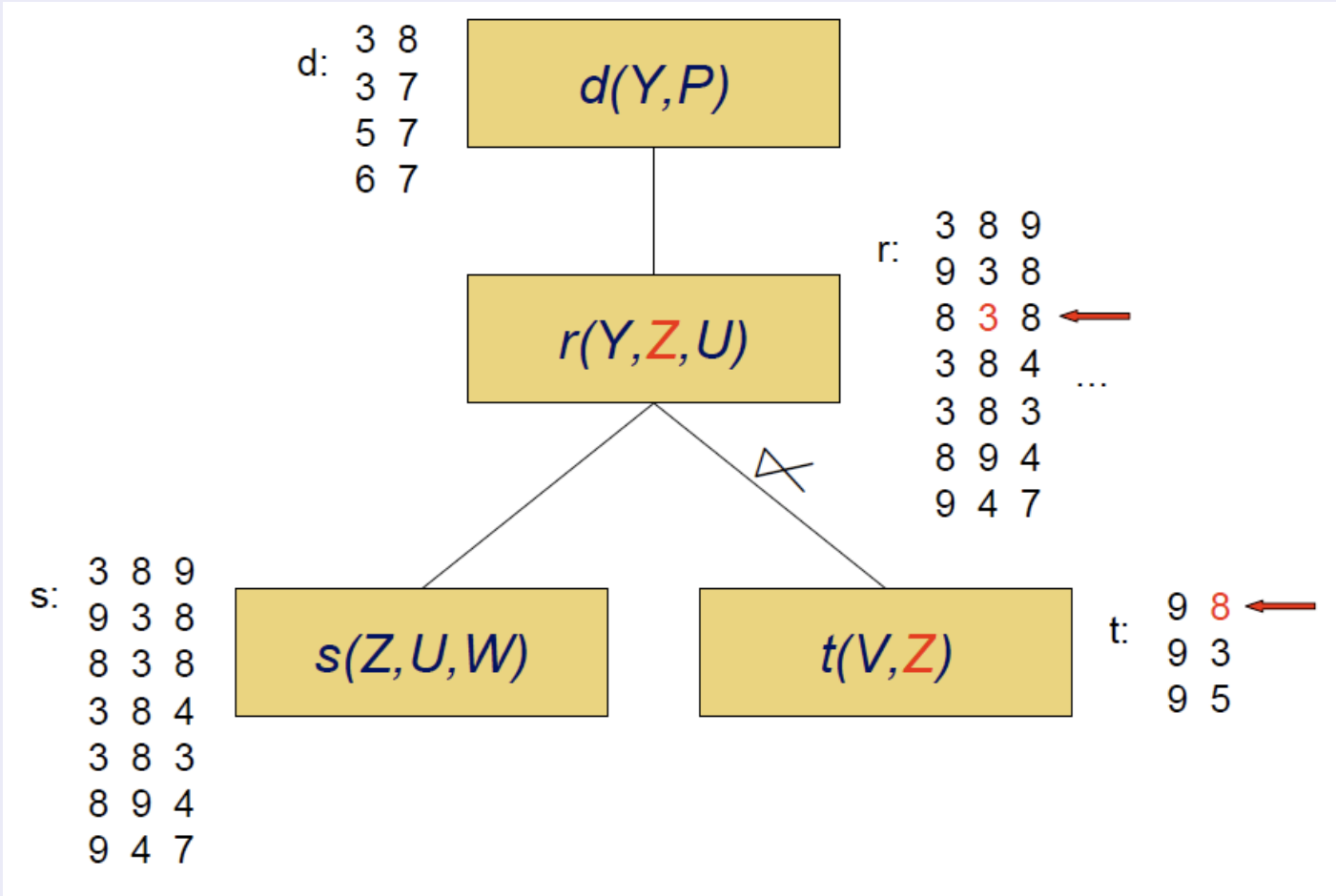
Acyclic Query Evaluation



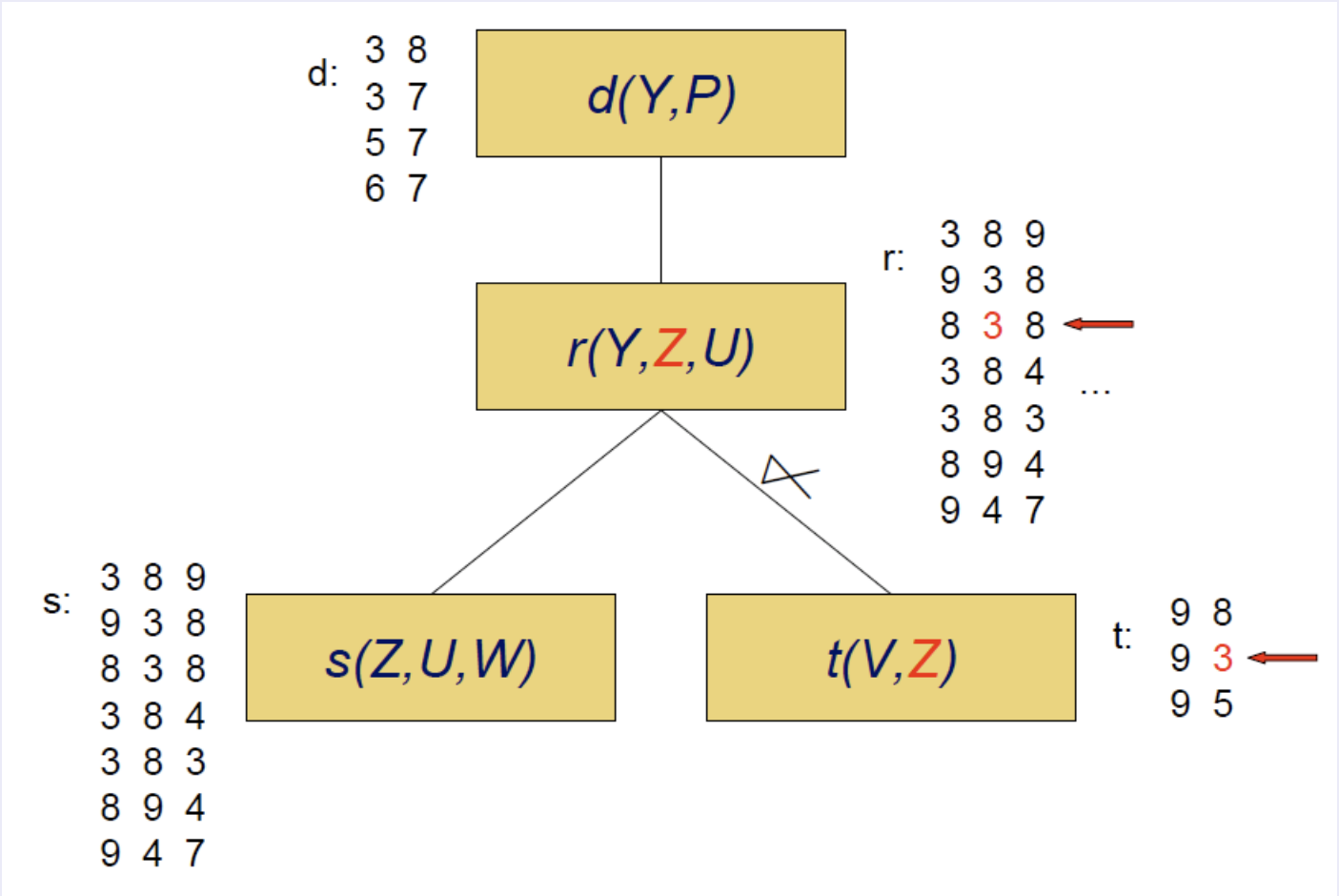
Acyclic Query Evaluation



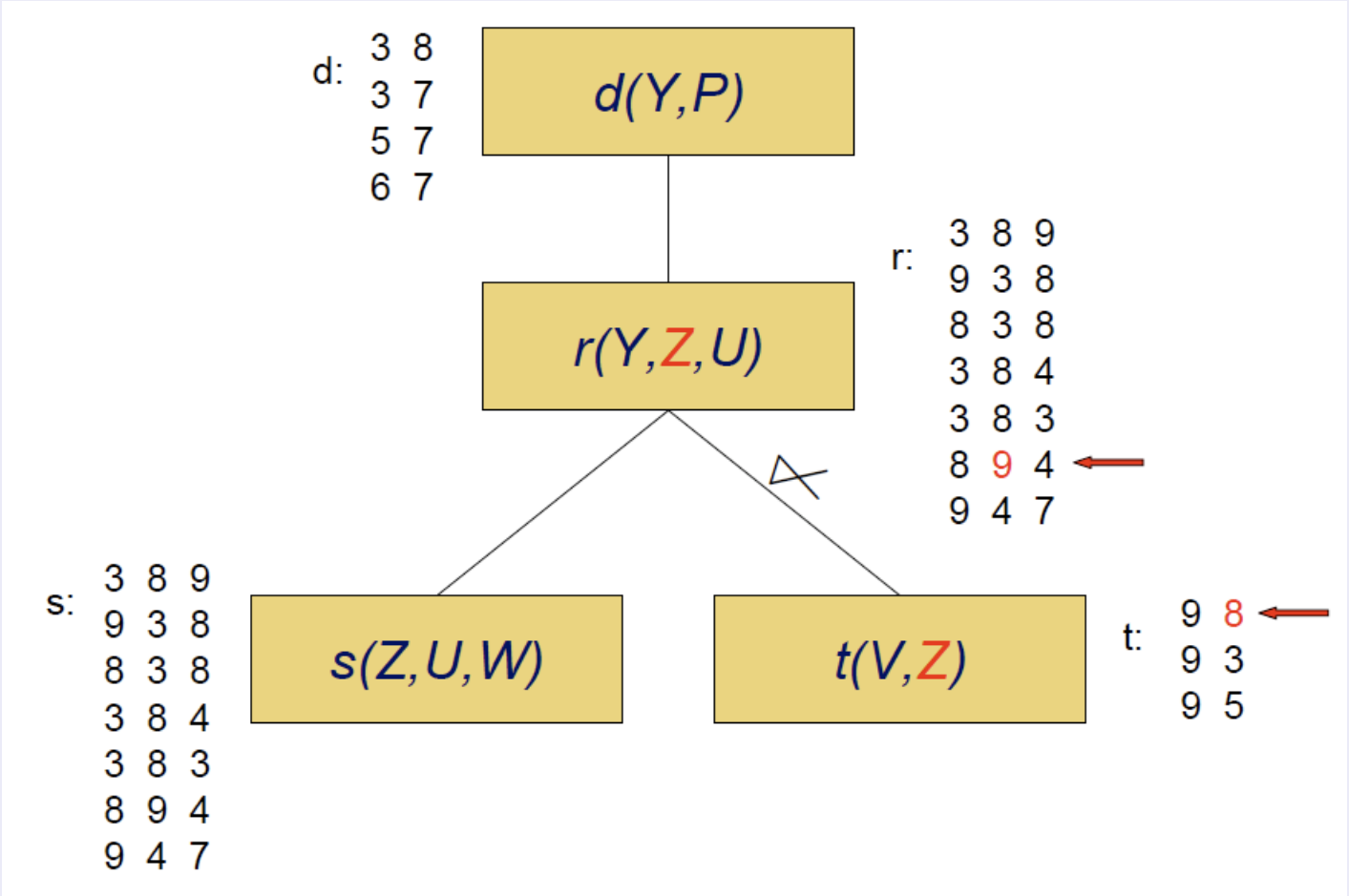
Acyclic Query Evaluation



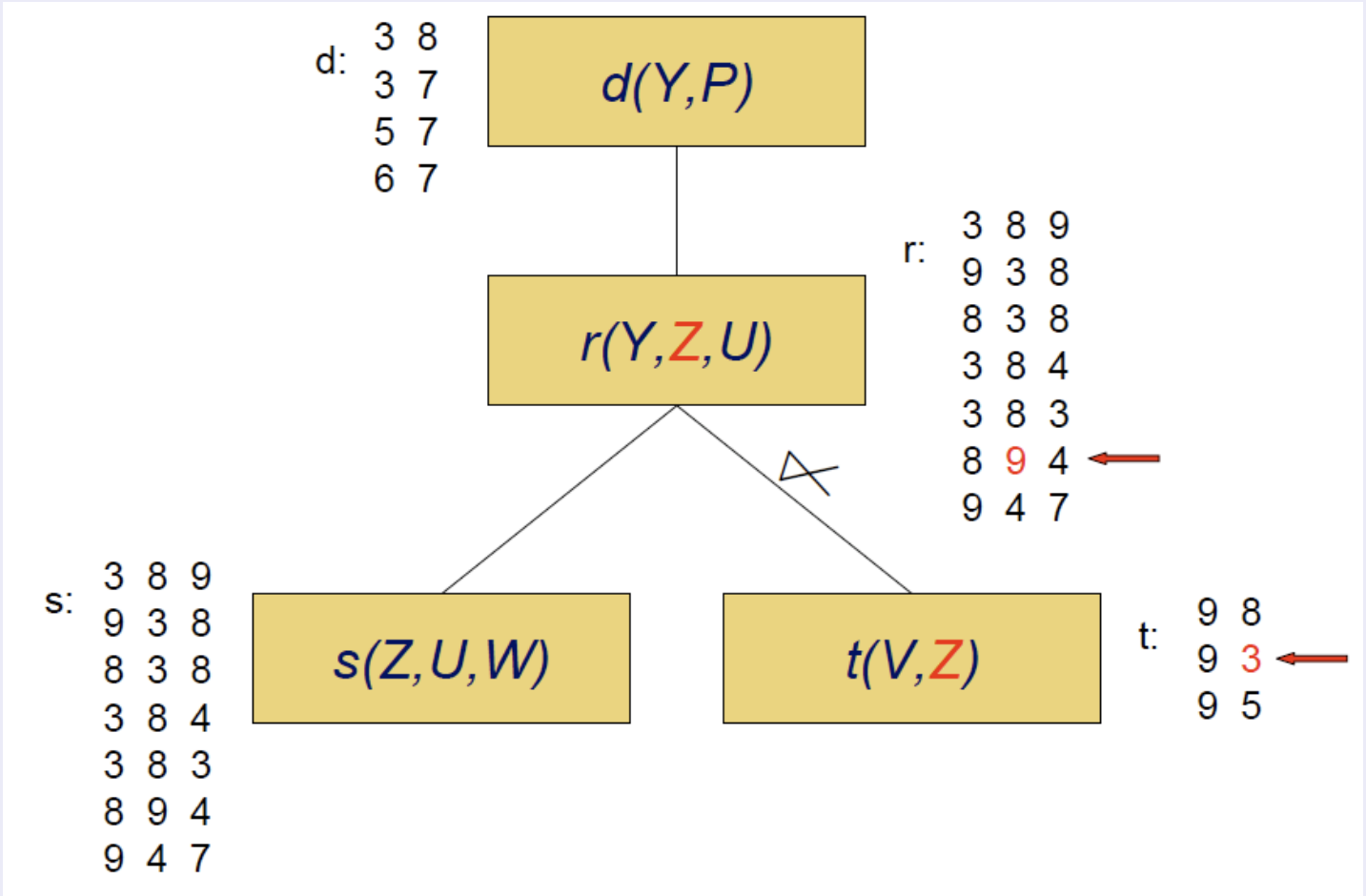
Acyclic Query Evaluation



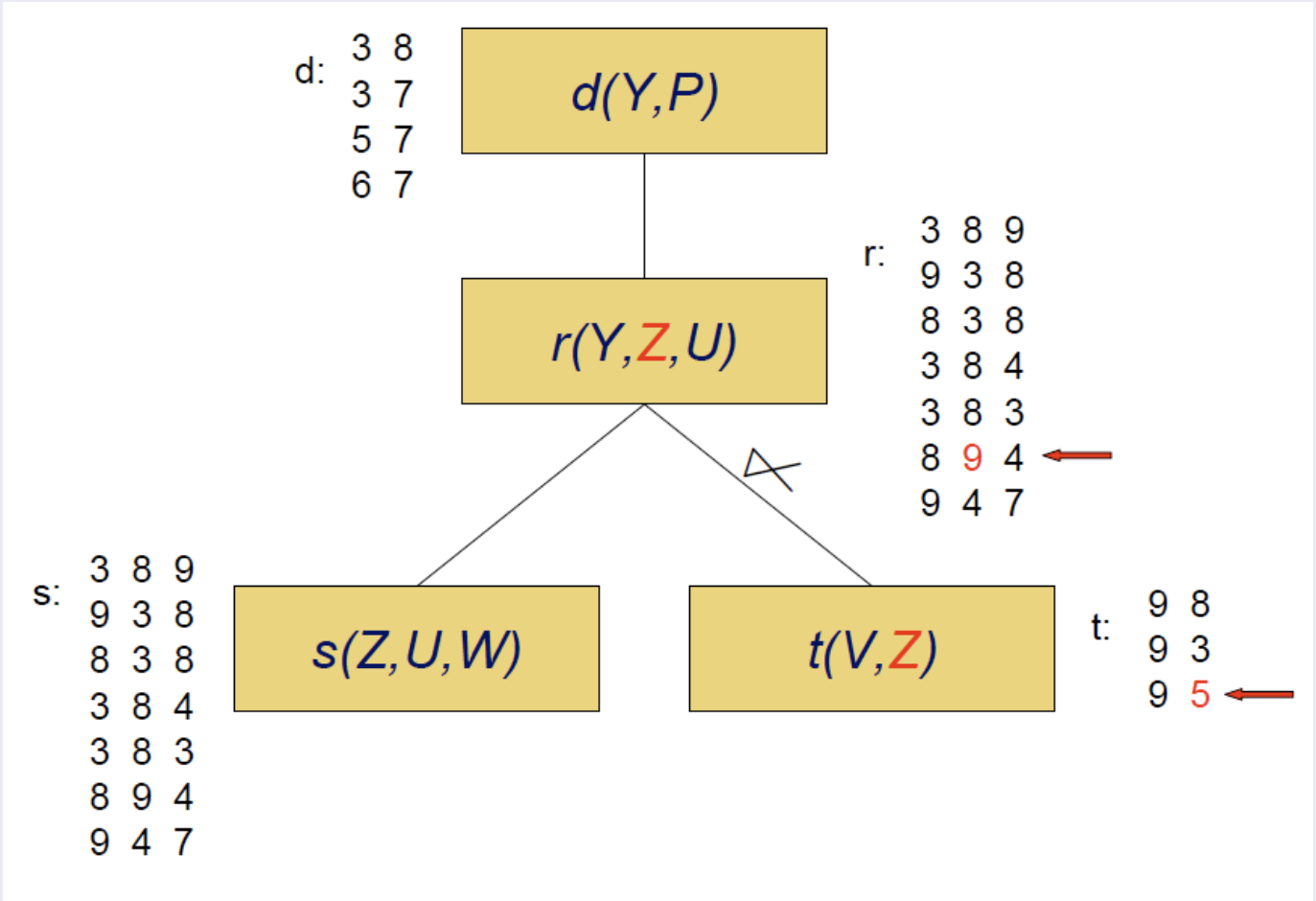
Acyclic Query Evaluation



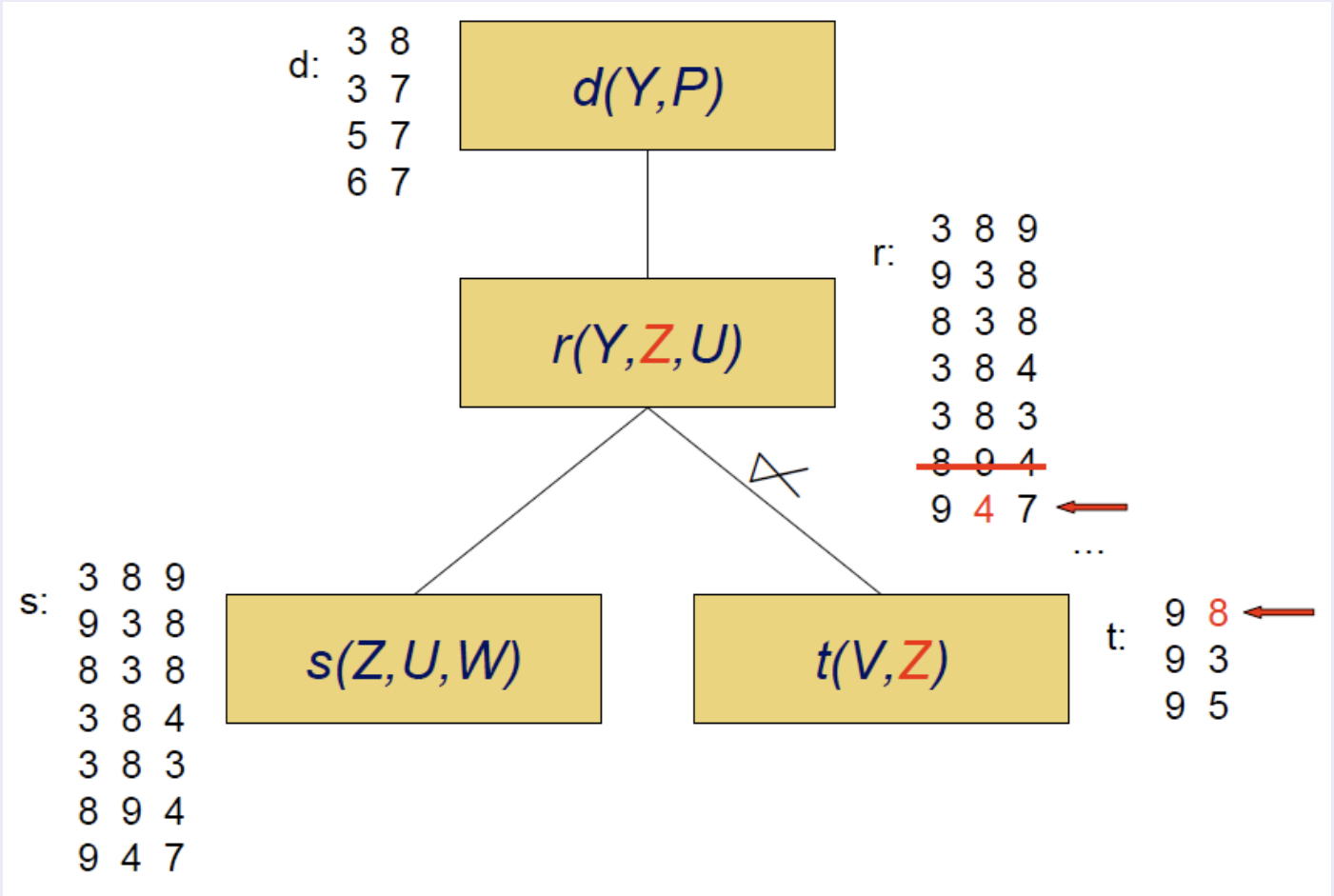
Acyclic Query Evaluation



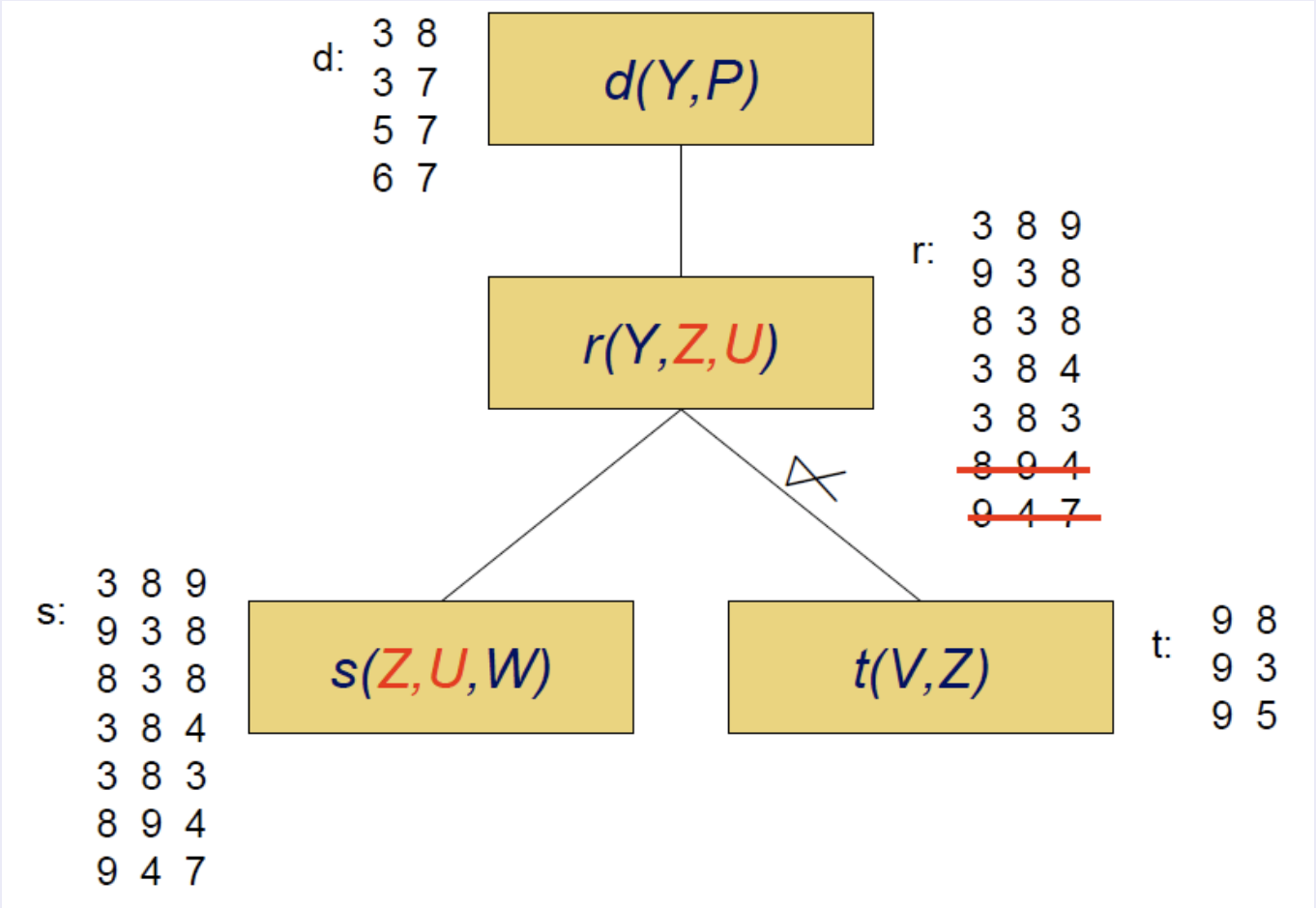
Acyclic Query Evaluation



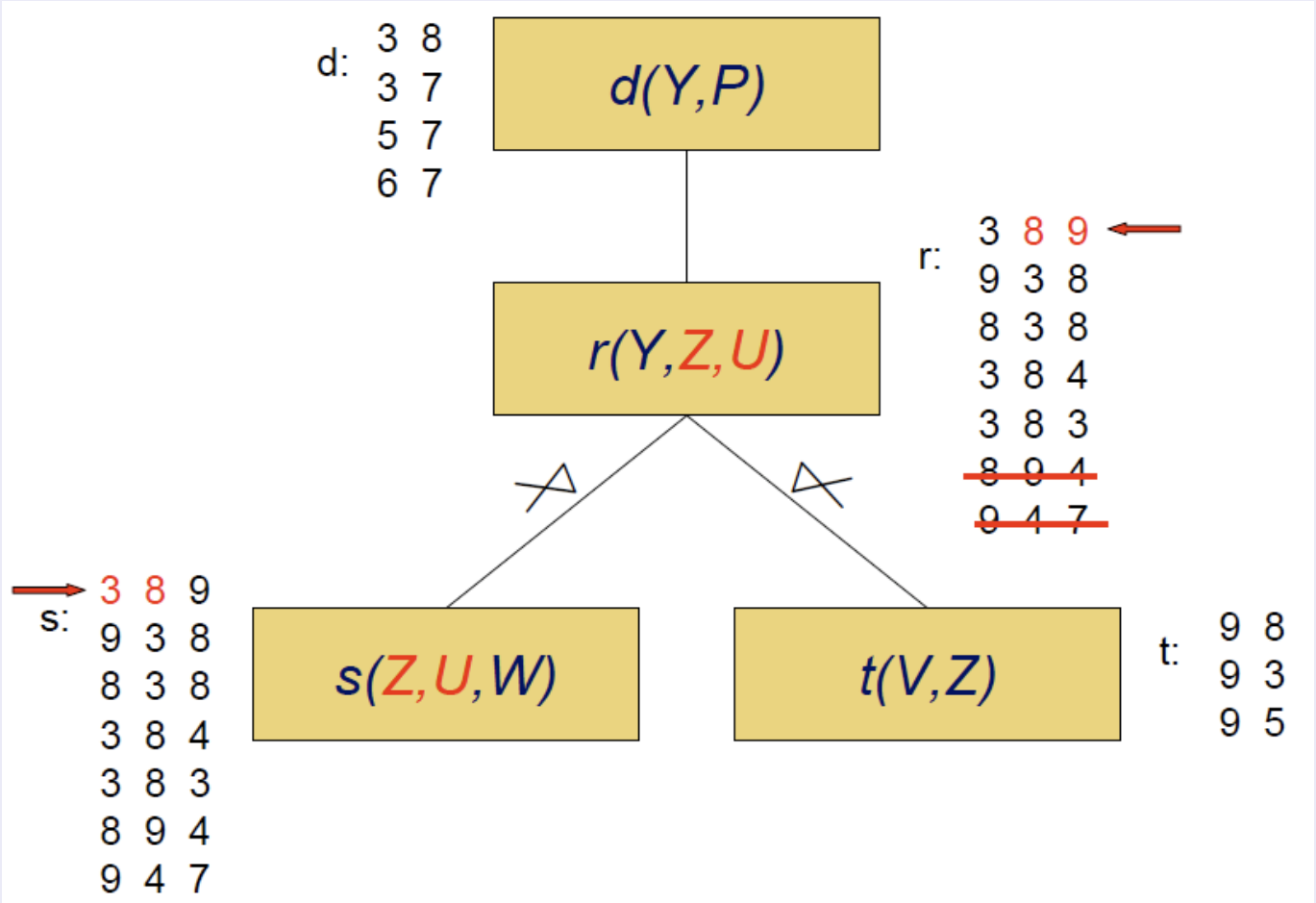
Acyclic Query Evaluation



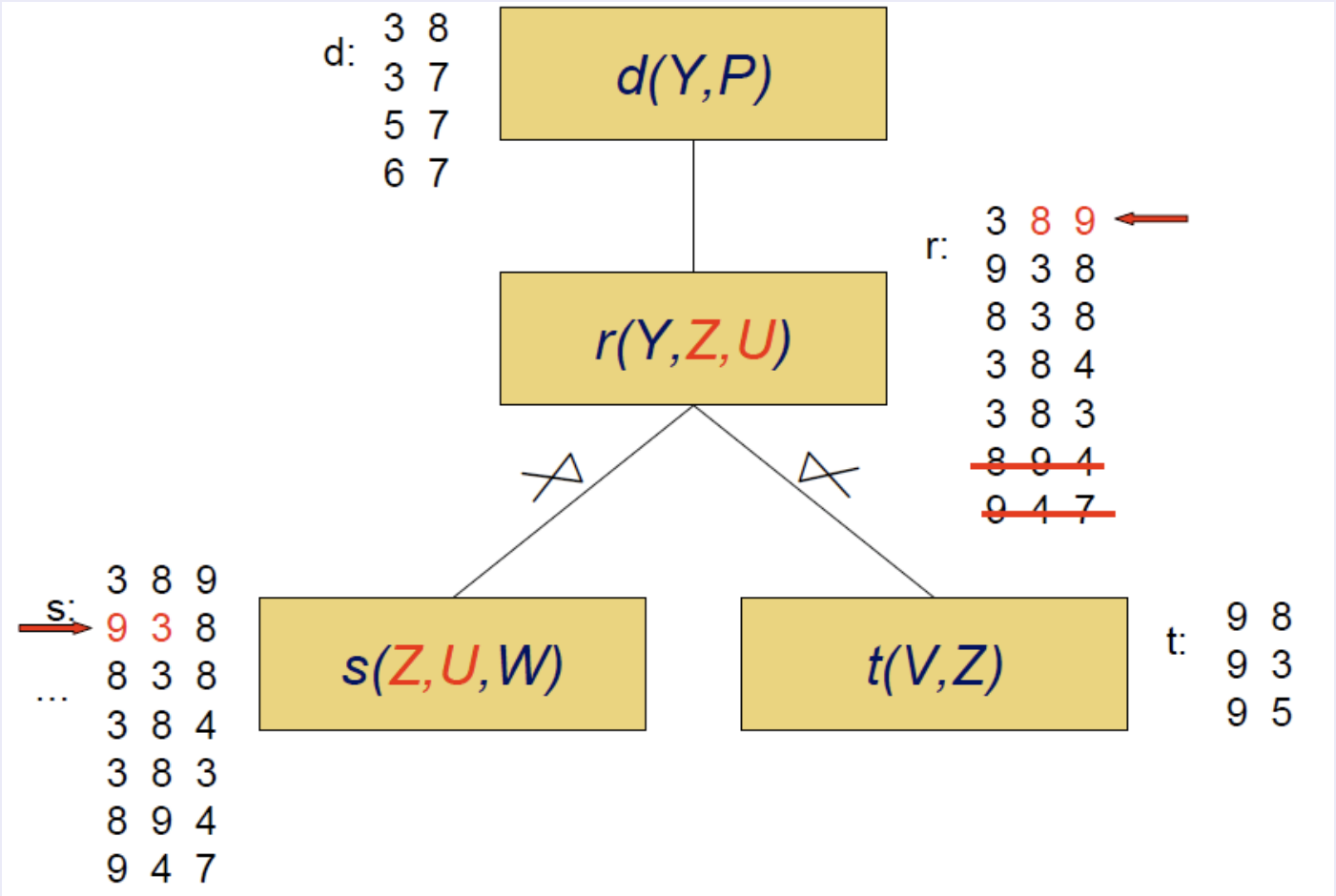
Acyclic Query Evaluation



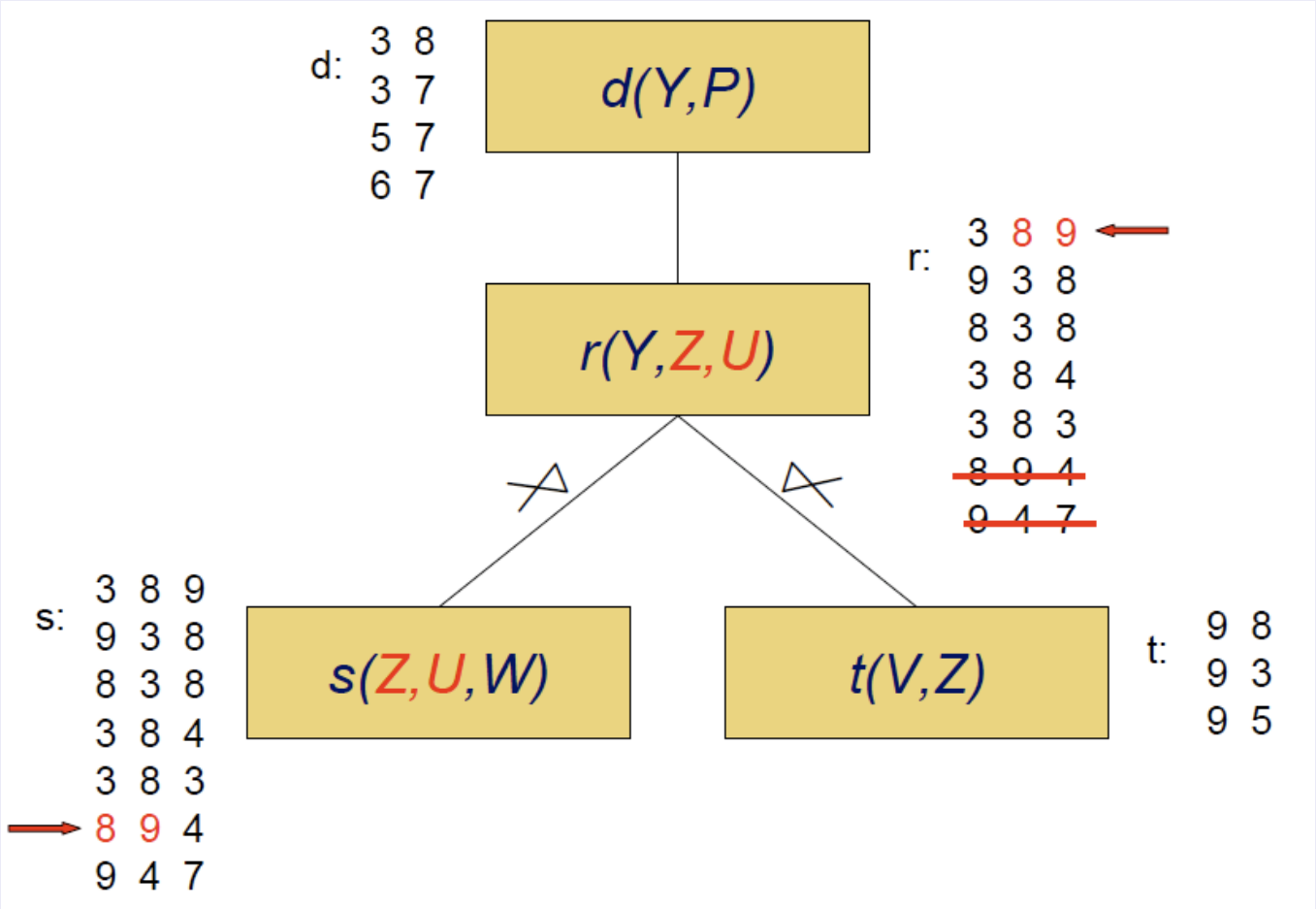
Acyclic Query Evaluation



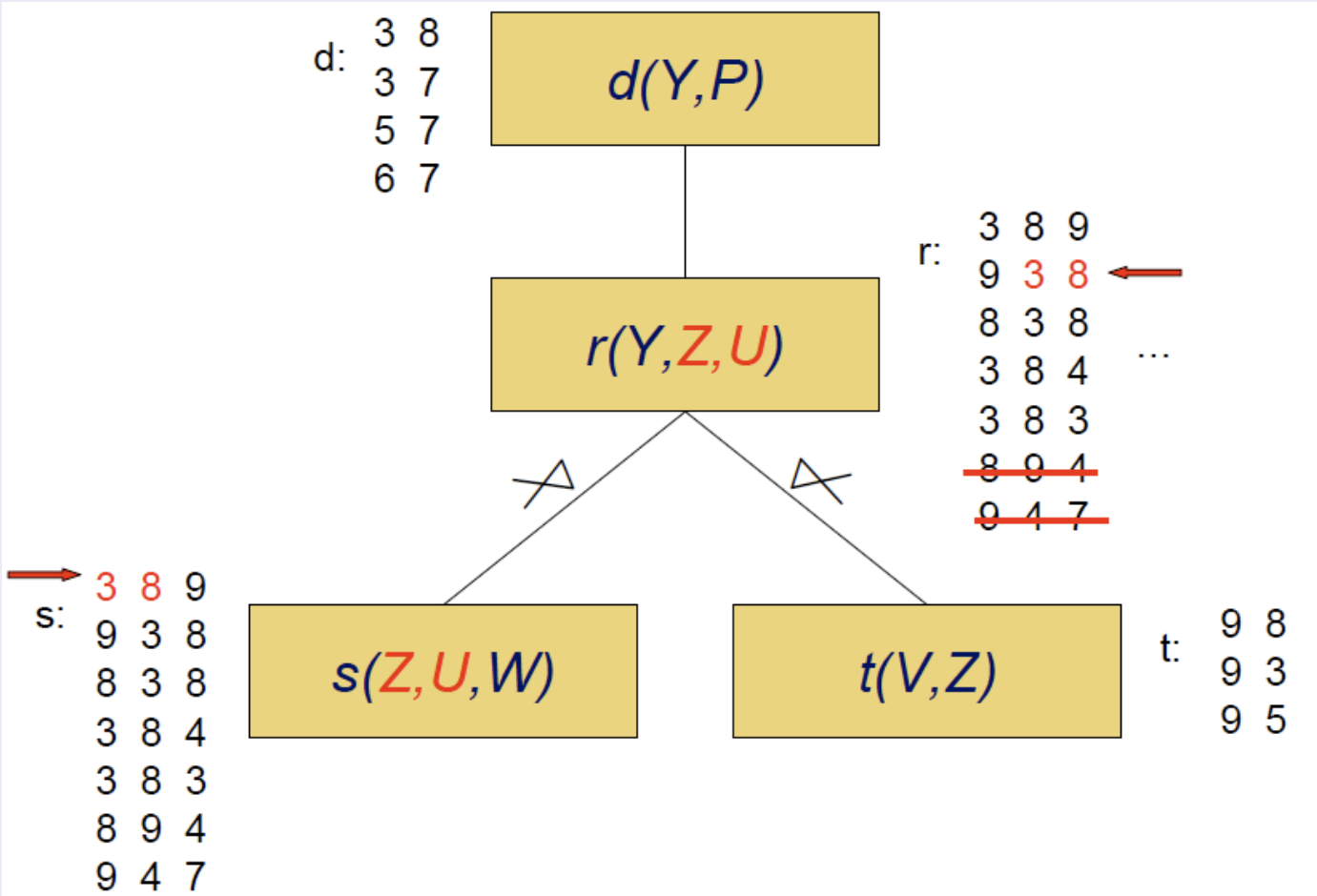
Acyclic Query Evaluation



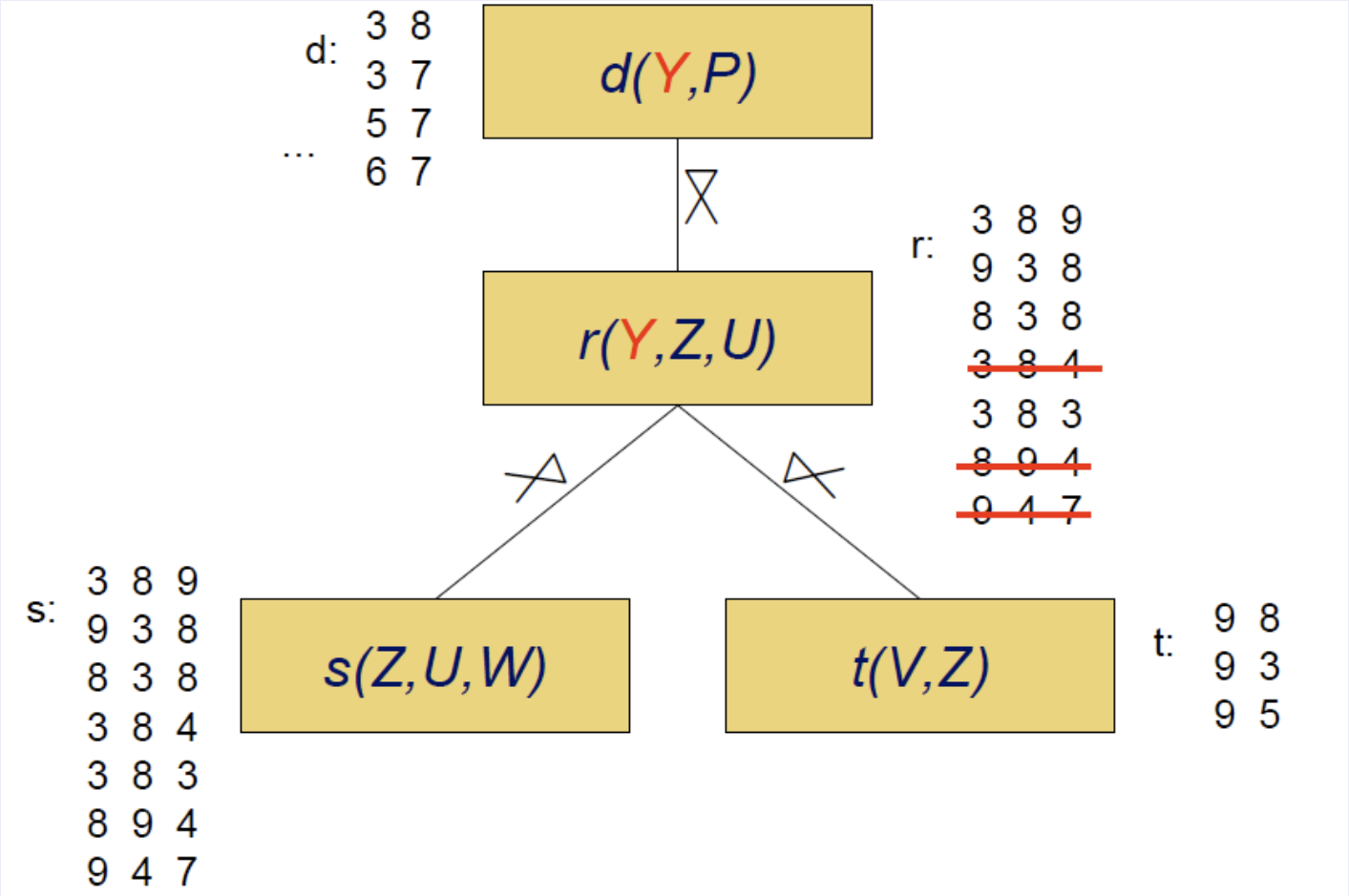
Acyclic Query Evaluation



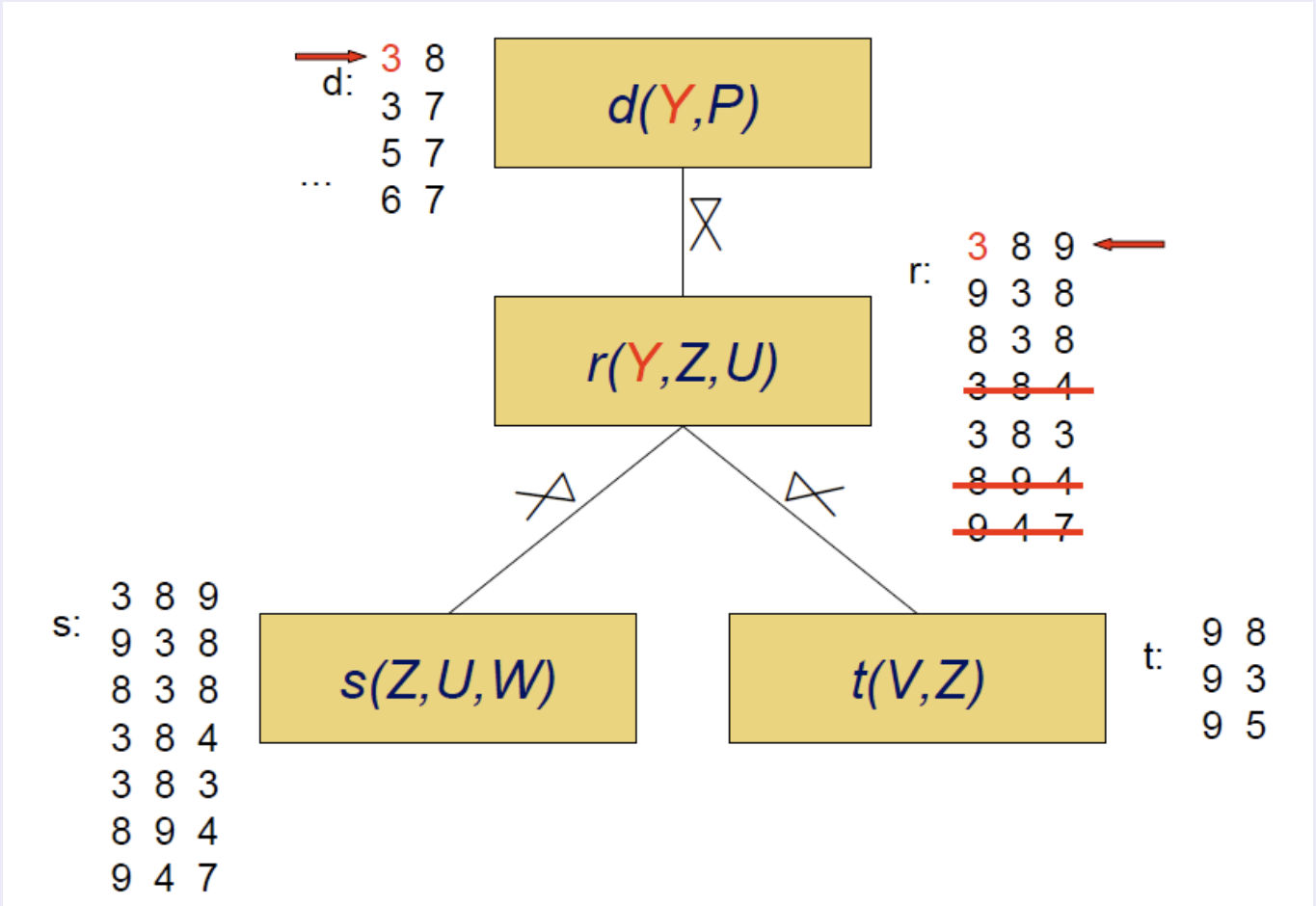
Acyclic Query Evaluation



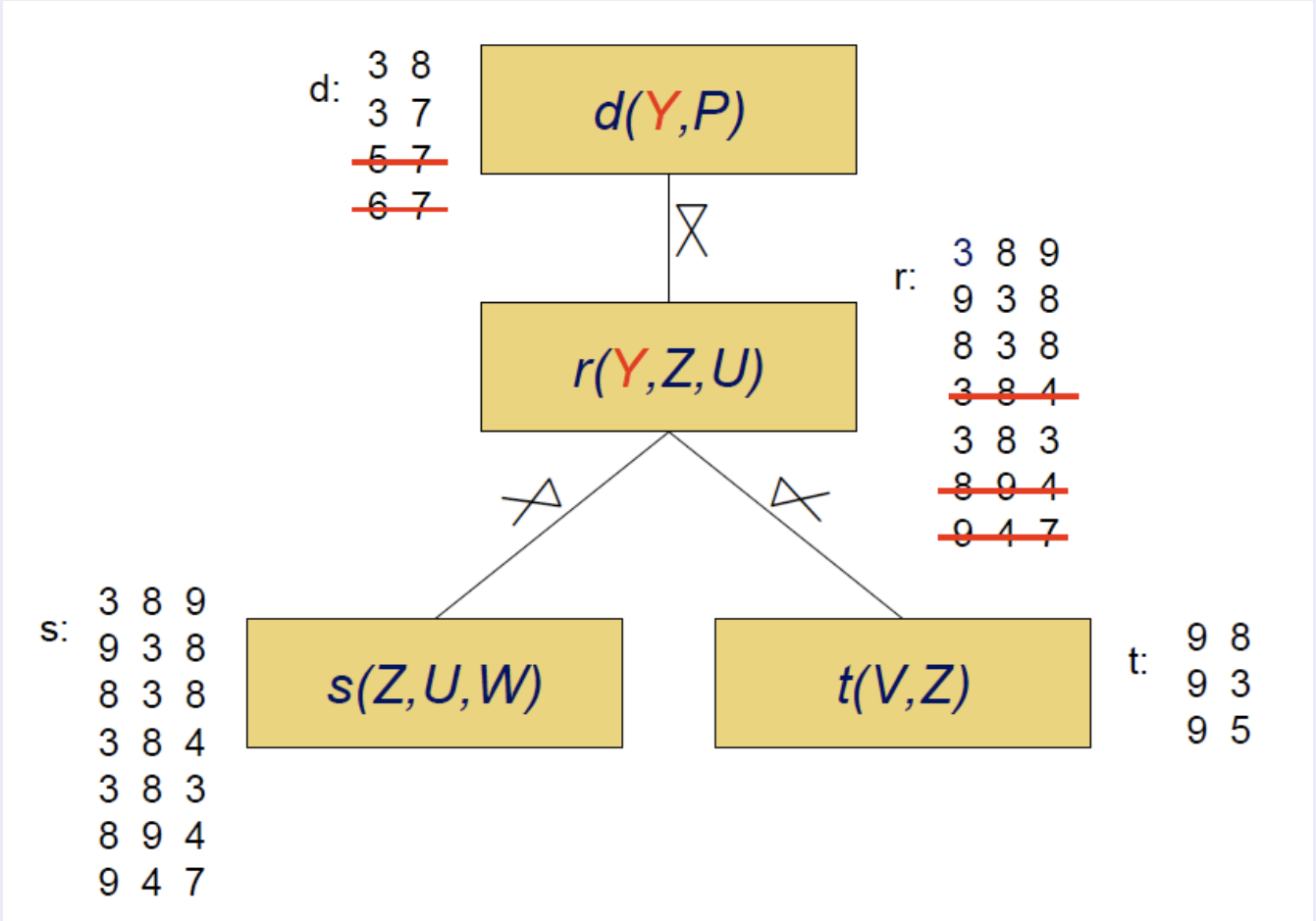
Acyclic Query Evaluation



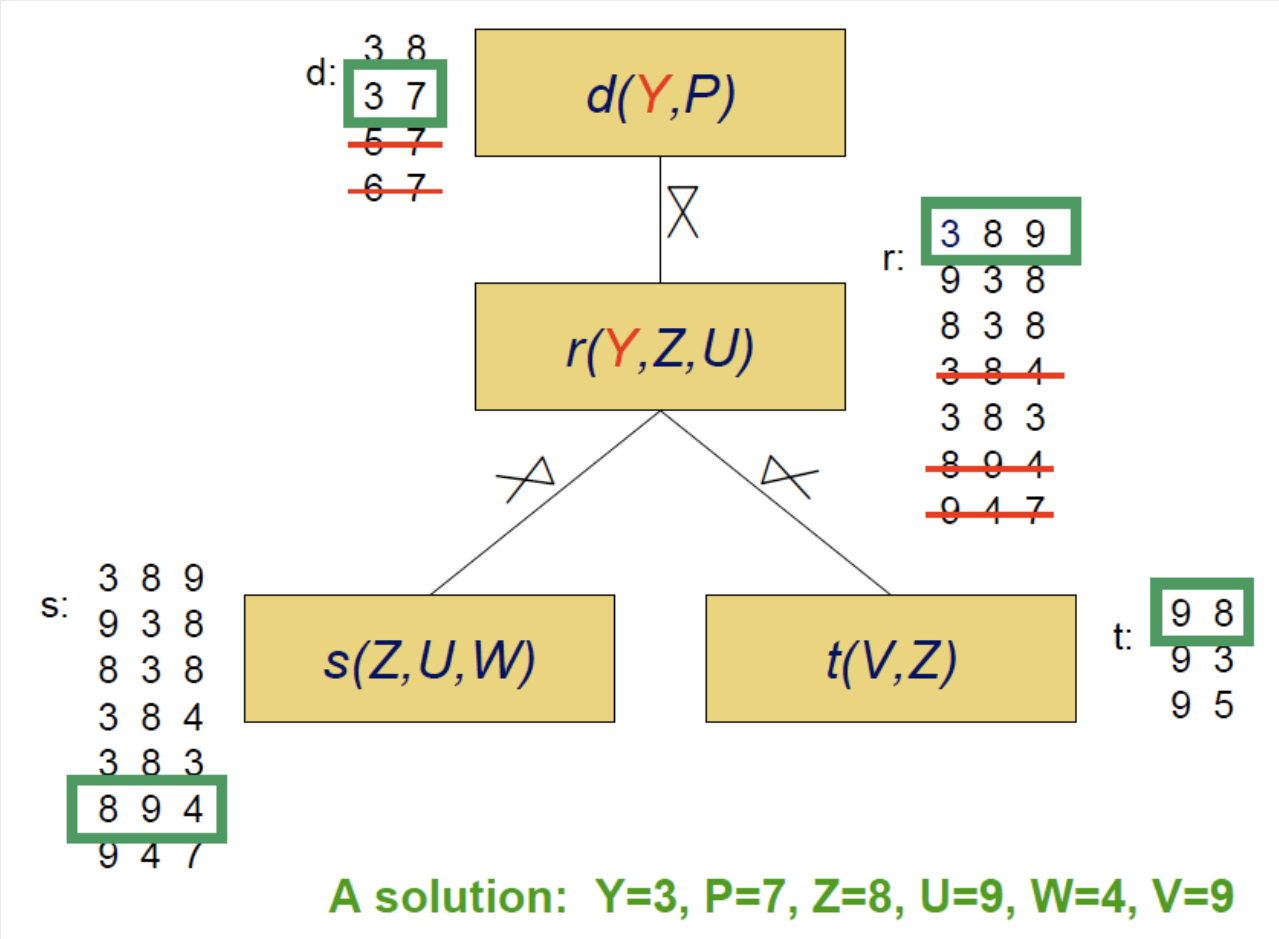
Acyclic Query Evaluation



Acyclic Query Evaluation



Acyclic Query Evaluation



Computing the Results

The result size can be exponential (even in case of ACQs).

- Even when the result is of polynomial size, it is in general hard to compute.
- In case of acyclic queries, the result can be computed in time polynomial in the result size (i.e., in output-polynomial time).
- The result of ACQs can be computed by adding a top-down phase to Yannakakis' algorithm for ABCQs and by joining the partial results.