CS-304 (DBMS)

Chapter (17)

Optimization

Introduction

- Optimization represents both a challenge and an opportunity for relational systems.
- Optimization means the process of choosing a suitable execution strategy for processing a query.
- Among all equivalent evaluation plans choose the one with lowest cost.
- The advantage of automatic optimization is not just that users do not have to worry about how best to state their queries.

- A good optimizer will have statistical information and keep in the system catalog.
- If the database statistics change over times, then a different choice of strategy might become desirable.
- Optimizer is a program. It is more patient than a typical human user.

A Motivating Example

S

S#	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

P

P#	PNAME	COLOR	WEIGHT	CITY
P1	Nut	Red	12	London
P2	Bolt	Green	17	Paris
P3	Screw	Blue	17	Rome
P4	Screw	Red	14	London
P5	Cam	Blue	12	Paris

SP

S#	P#	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300

S (S# , SNAME, STATUS, CITY)

P (P# , PNAME, COLOR, WEIGHT, CITY)

SP (S# , P*, QTY)

" Get names of suppliers who supply part P2."

Two equivalent relational algebra queries are:

1 ((SP JOIN S) WHERE P# = P# ('P2')) {SNAME}

2 ((SP WHERE P# = P# ('P2')) JOIN S) {SNAME}

Assume that:

- 100 tuples in Suppliers
- 10,000 tuples in Shipments, of which only 50 tuples are for part P2.

- **Relvars** : Supplier(**S**) and Shipment(**SP**) are represented directly on the disk as two separate stored files, with one stored record per tuple.

- 50 tuples at most, which can stay in main memory.

First Procedure

((SP JOIN S) WHERE P# =P# ('P2')) {SNAME}

step 1: JOIN SP tuples and S tuples with over S#

read SP tuples from disk: **10,000** tuple I/O's

JOIN, read S tuples from disk: **1,000,000** tuple I/O's

write result back to disk: **10,000** tuple I/O's

step 2: RESTRICT the result of Step1 to the tuples for part P2

read the Step1 result from disk: **10,000** tuple I/O's

produces a result consisting of only 50 tuples, enough to be kept in main memory.

First Procedure *(cont'd)*

step 3: PROJECT the result of Step2 over SNAME

produce the desired final result

(50 tuples at most, which can stay in main memory)

total of the first execution plan – 1,030,000 tuple I/O's

Second Procedure

((SP WHERE P# = P# ('P2')) JOIN S) {SNAME}

step 1: RESTRICT SP to the tuples for part P2

read SP tuples from disk: **10,000** tuple I/O's

produces a result consisting of only 50 tuples, which will be kept in main memory.

step 2: JOIN the result of Step1 to S (over S#)

read S tuples from disk: **100** tuple I/O's

produce a result of 50 tuples again (still in main memory)

Second Procedure *(cont'd)*

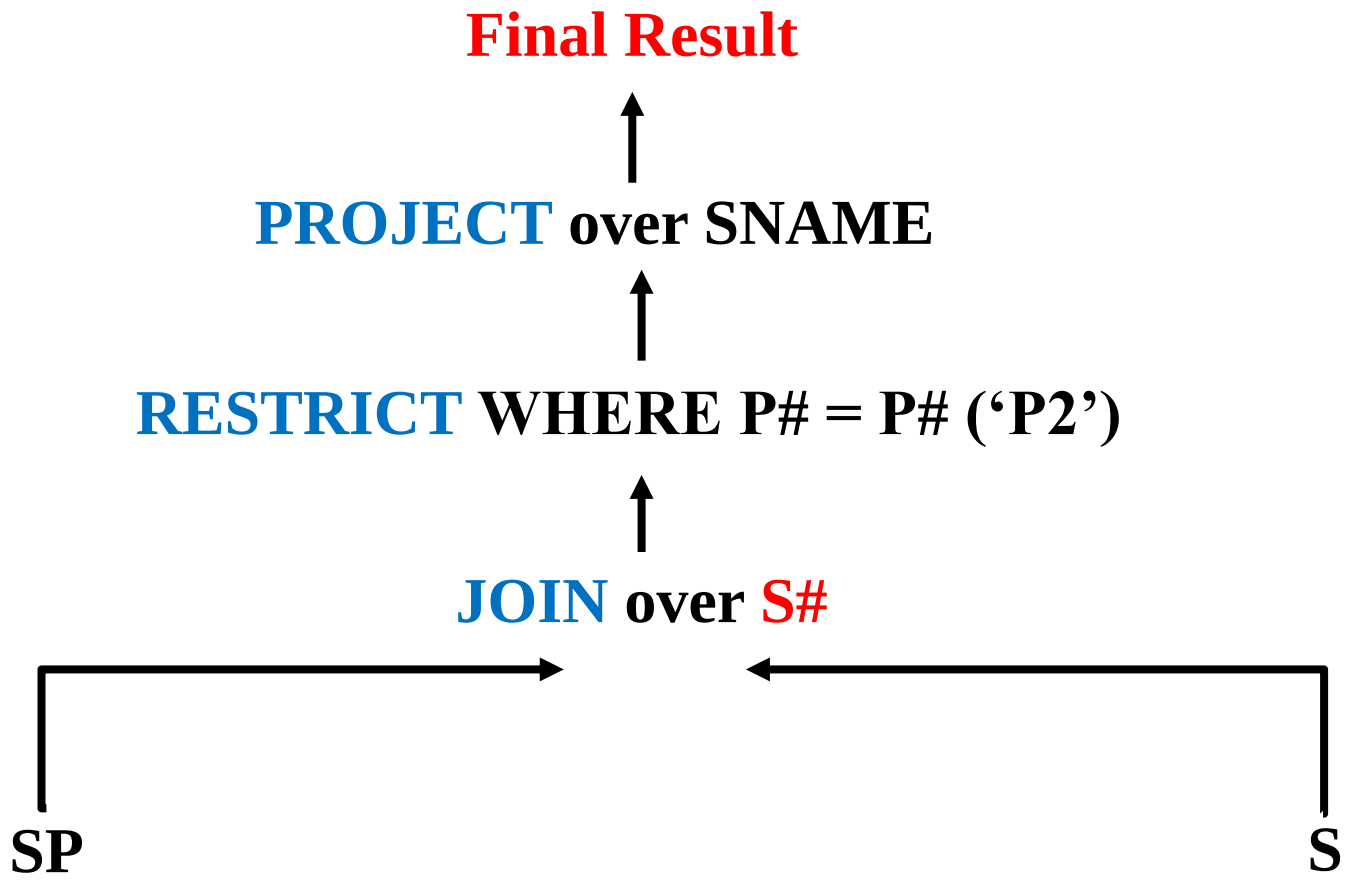
step 3: PROJECT the result of Step2 over SNAME

produce the desired final result (50 tuples at most) stays in main memory

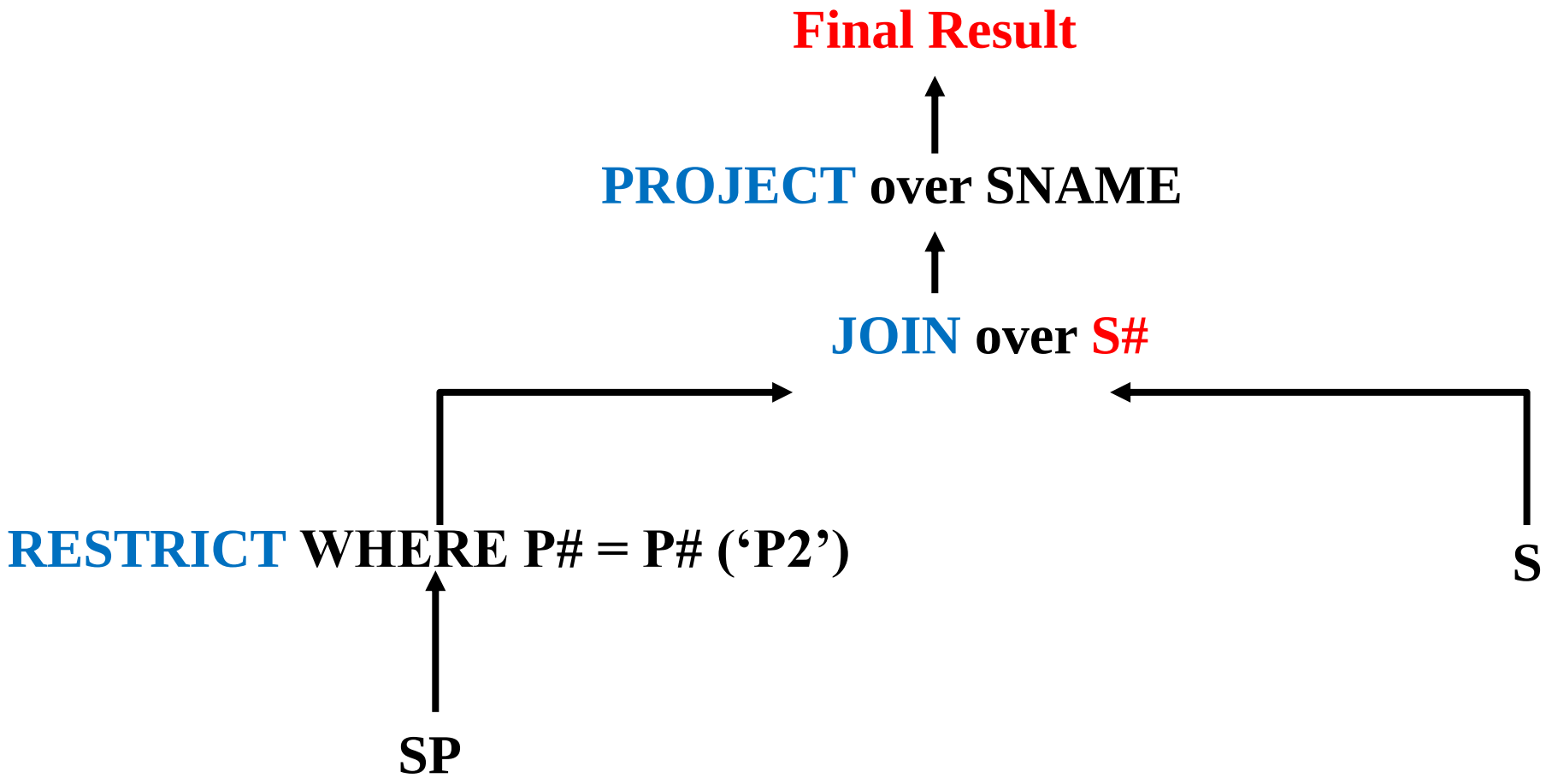
total of the second execution plan – 10,100 tuple I/O's

The second procedure is better than the first that take number of tuple I/O's as performance measure, so second execution plan is optimized.

Query Tree : First Procedure



Query Tree : Second Procedure



Next Lecture…

An Overview of Query Processing