COST ESTIMATION & QUERY OPTIMIZATION

CSE 444 – Section 4

Estimating Cost

We have 3 relations:

Student(sid, name, age, addr)

Book(bid, title, author)

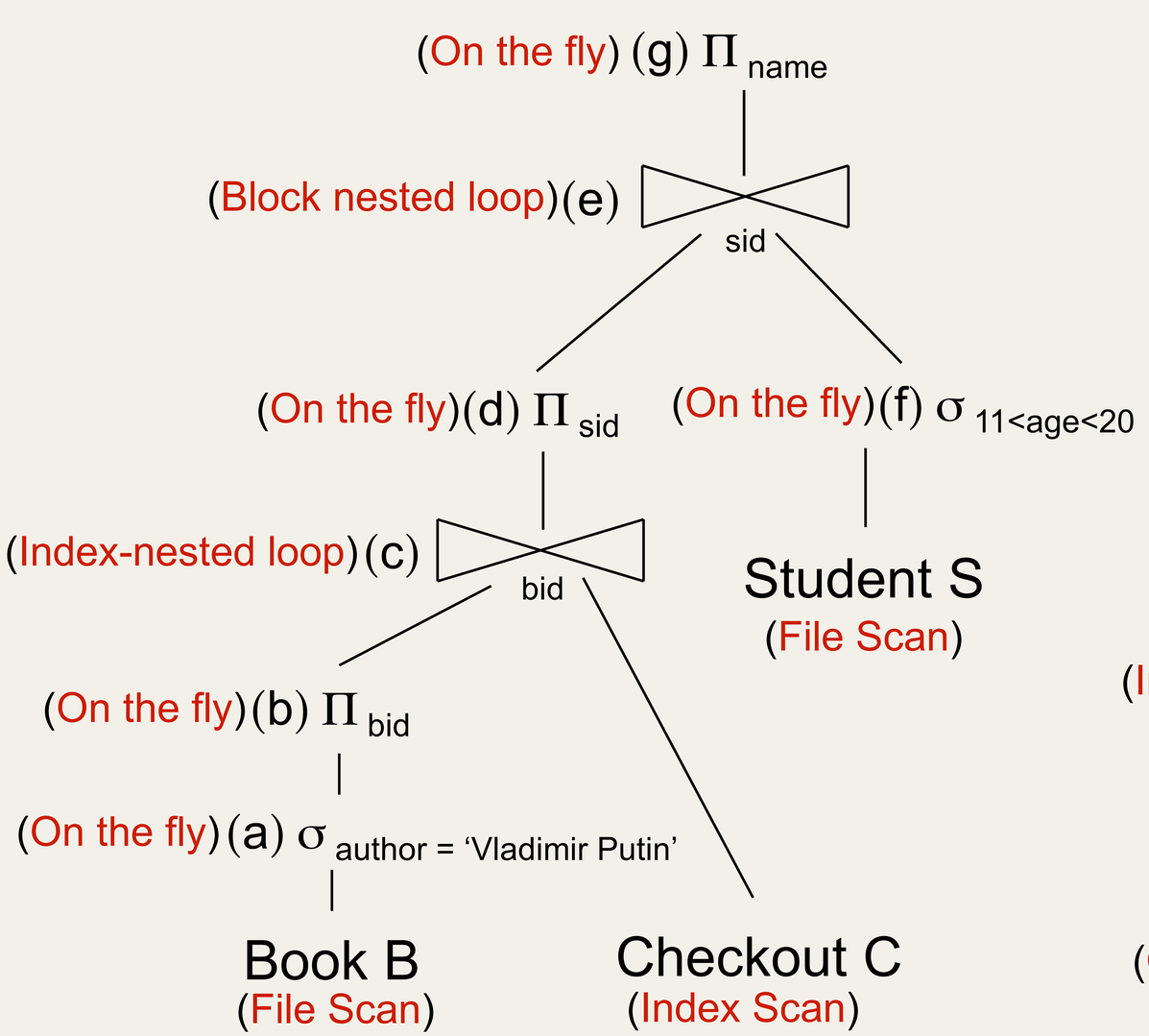
Checkout(sid, bid, date)

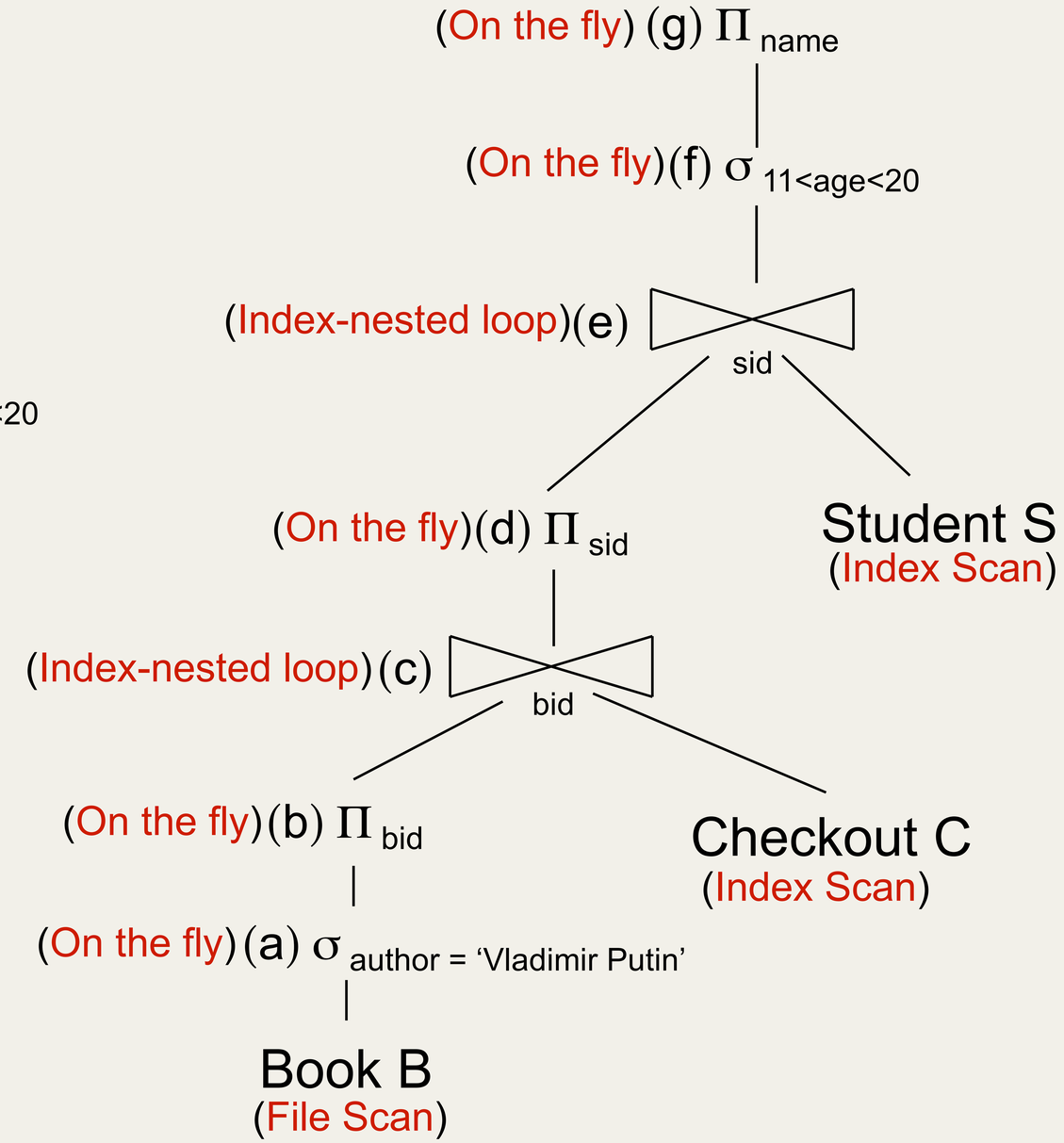
We want to run this query:

```
SELECT S.name
FROM Student S, Book B, Checkout C
WHERE S.sid = C.sid
AND B.bid = C.bid
AND B.author = 'Vladimir Putin'
AND S.age > 11
AND S.age < 20
```

Draw a possible logical query plan

Which physical plan is better?





S(sid, name, age, addr)
B(bid, title, author)
C(sid, bid, date)

Assumptions

- Student: S, Book: B, Checkout: C
- Sid, bid foreign key in C referencing S and B resp.
- Clustered index on C(bid, sid)
- There are 10,000 Student records stored on 1,000 pages.
- There are 50,000 Book records stored on 5,000 pages.
- There are 300,000 Checkout records stored on 15,000 pages.
- There are 8,000 unique students who have an entry in Checkout
- There are 10,000 unique books that are referenced in Checkout
- There are 500 different authors.
- 8 <= student age <= 23

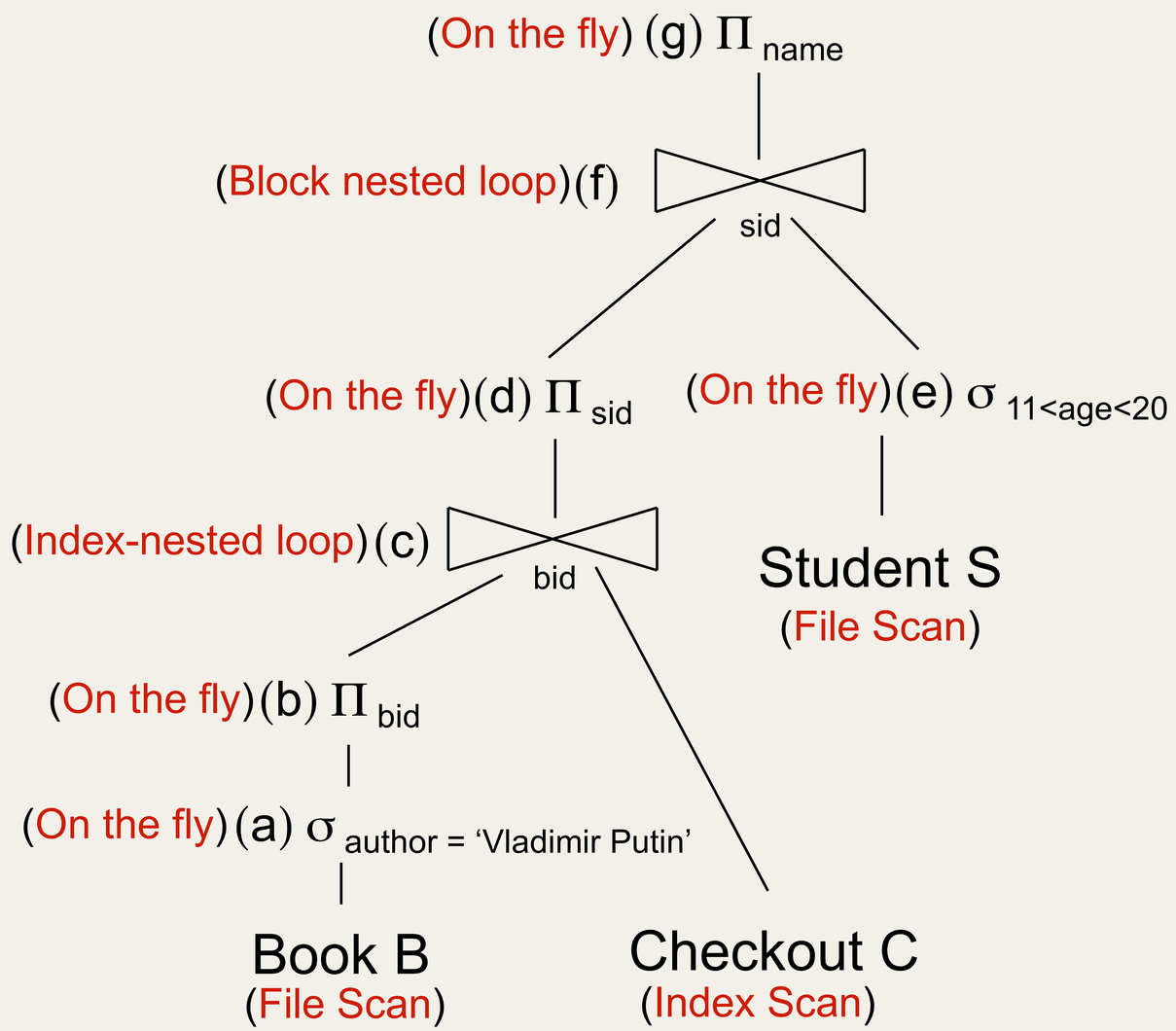
V(B, author) = 500
V(C, sid) = 8000
V(C, bid) = 10000
8 <= age <= 23

T(S)=10,000
T(B)=50,000
T(C)=300,000

B(S)=1,000
B(B)=5,000
B(C)=15,000

S(<u>sid</u>, name, age, addr)
B(<u>bid</u>, title, author)
C(<u>sid</u>, <u>bid</u>, date)

Selectivity



- (a) σ author = 'Vladimir Putin'
- (c) Join predicate bid
- (e) σ 11<age<20
- (f) Join predicate sid

- *(a) 1 / V(B, author)*
  = 1 / 500
- *(c) 1 / max(V(B, bid), V(C, bid))*
  = 1 / max(50000, 10000)
  = 1 / 50000
- *(e) (# ages covered) / (# possible ages)*
  = 8 / 16
  = 1 / 2
- *(f) 1 / max(V(C, sid), V(S, sid))*
  = 1 / max(8000, 10000)
  = 1 / 10000

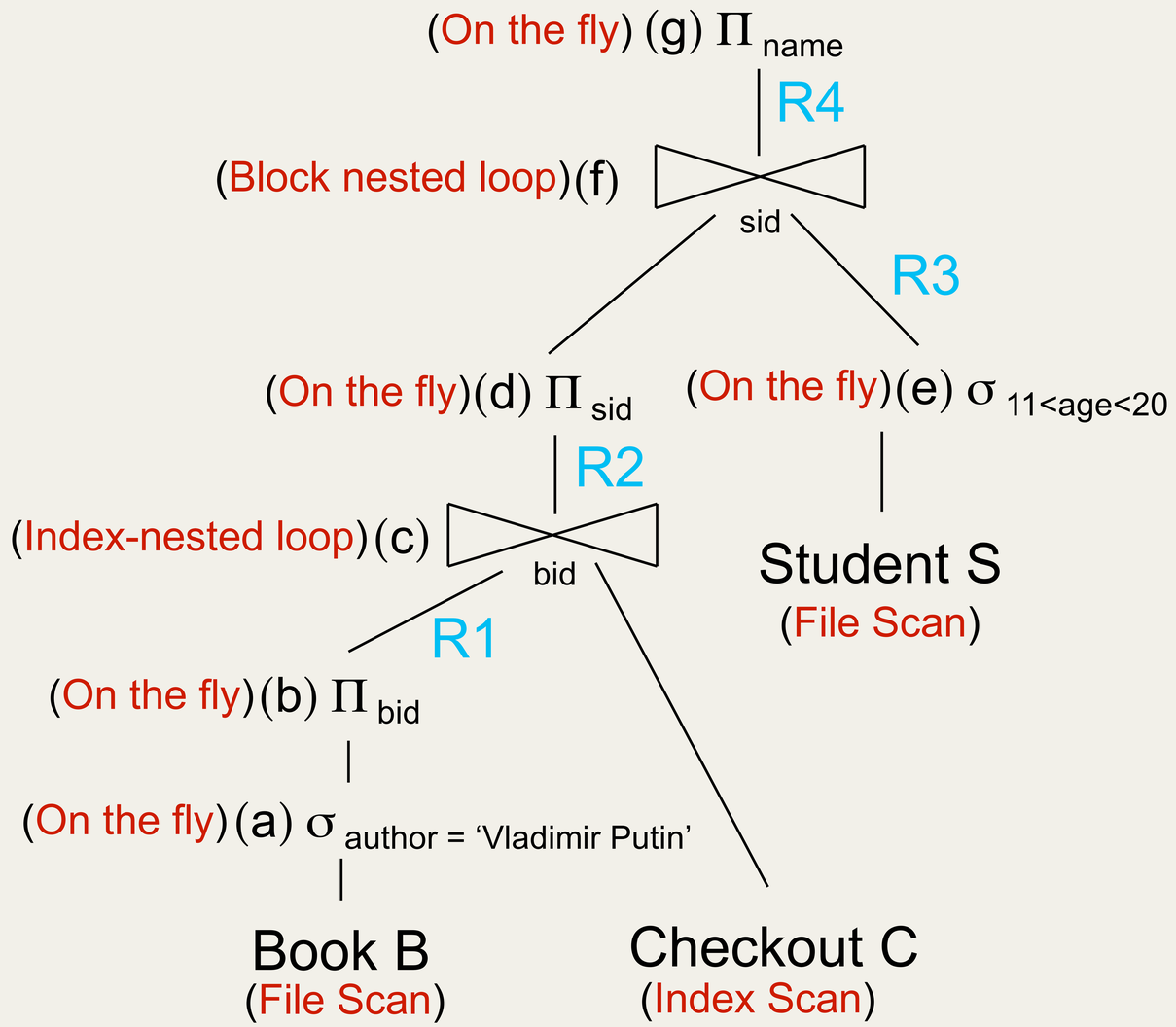
V(B, author) = 500
V(C, sid) = 8000
V(C, bid) = 10000
8 <= age <= 23

T(S)=10,000
T(B)=50,000
T(C)=300,000

B(S)=1,000
B(B)=5,000
B(C)=15,000

S(sid, name, age, addr)
B(bid, title, author)
C(sid, bid, date)

Cardinality



- T(R1) = ?
- T(R2) = ?
- T(R3) = ?
- T(R4) = ?

- T(R1) = T(B) / 500
  = 100
- T(R2) = T(R1) * T(C) / 50000
  = 100 * 300000 / 50000
  = 600
- T(R3) = T(S) / 2
  = 10000 / 2
  = 5000
- T(R4) = T(R2) * T(R3) / 10000
  = 600 * 5000 / 10000
  = 300

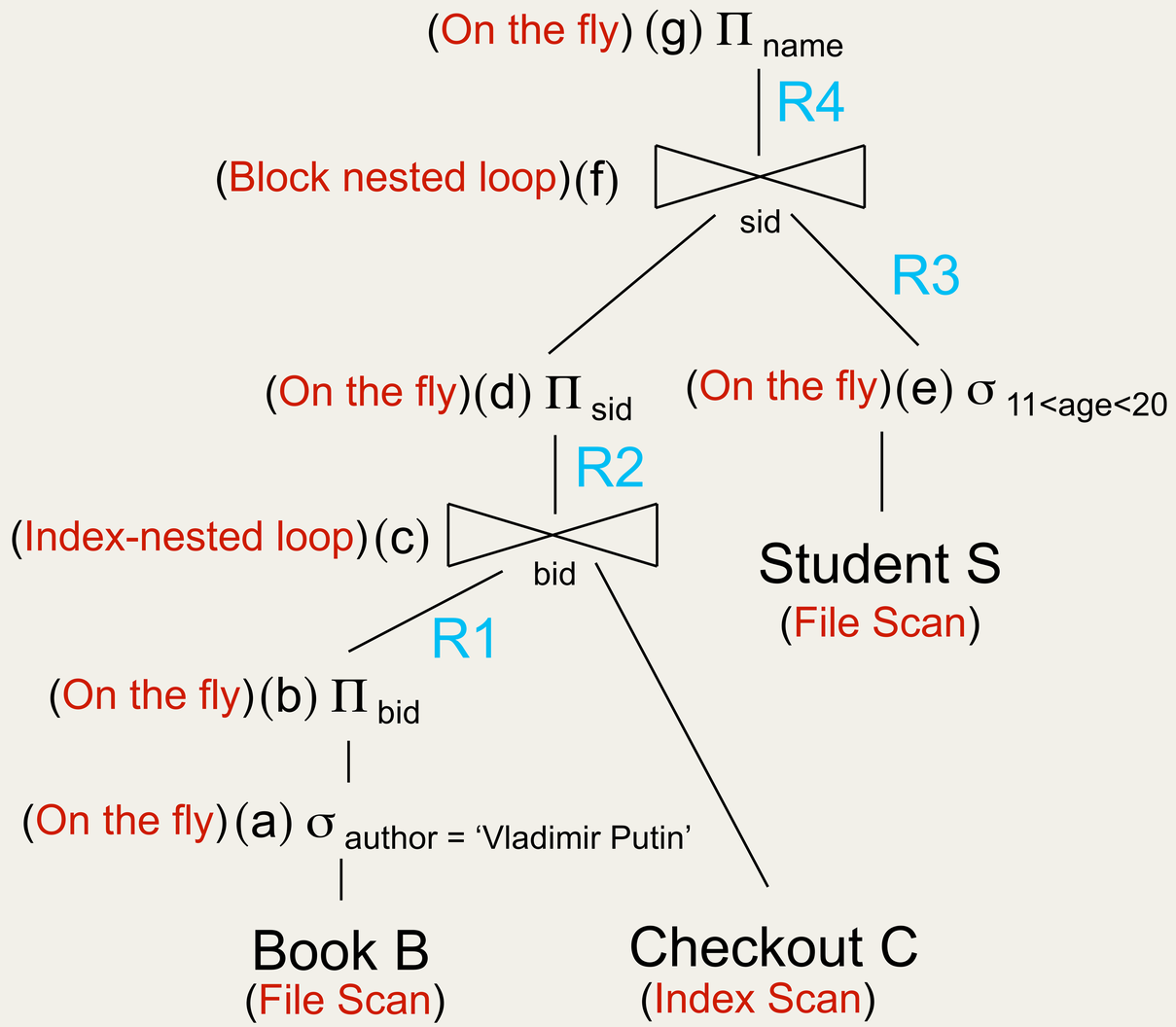
V(B, author) = 500
V(C, sid) = 8000
V(C, bid) = 10000
8 <= age <= 23

T(S)=10,000
T(B)=50,000
T(C)=300,000

B(S)=1,000
B(B)=5,000
B(C)=15,000

S(<u>sid</u>, name, age, addr)
B(<u>bid</u>, title, author)
C(<u>sid</u>, <u>bid</u>, date)

Cost



- Data not sorted in any way
- Relations can fit in memory
- Compute the cost of each step (a) through (g)

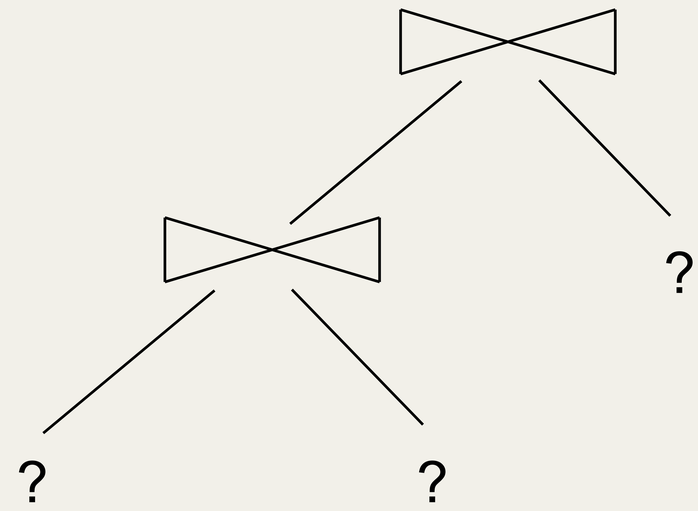
- (a) B(B) = 5000
- (b) 0
- (c) T(R1) * B(C) / V(C, bid)
  = 100 * 15000 / 10000 = 150
- (d) 0
- (e) B(S) = 1000
- (f) 0
- (g) 0
- Total: 6150

Selinger Optmization

We want to run this query:

```
SELECT *
FROM R, S, T
WHERE R.a = S.a
AND S.b = T.b
```

What is the best join order?



Search space heuristics:

- Push selections down
- Avoid cartestian products
- Restrict to left-linear trees

Example OPT Table

```
SELECT *
FROM R, S, T
WHERE R.a = S.a
AND S.b = T.b
```

Subquery	Cost	Output Size	Plan	Prune or Keep
R	5	…	Seq. scan	Keep
S	6	…	Seq. scan	Keep
T	20	…	Seq. scan	Keep
RS	40	…	Index Join	Keep
SR	220	…	Nested loop	Prune
ST	60	…	…	Keep
TS	120	…	…	Prune
(RS)T	600	…	…	Prune
(ST)R	480	…	…	Keep