Ottimizzazione Fisica (Parte 2)

Lorenzo Vaccarecci

3 Maggio 2024

1 Operatori Fisici

Sono implementazioni specifiche di un operatore logico (ad esempio del join), basate su un certo algoritmo di realizzazione.
Esistono diversi algoritmi di realizzazione per ogni operatore logico. Algoritmi diversi possono utilizzare diverse informazioni contenute a livello fisico.
La possibilità di usare un certo operatore fisico e quindi generare un certo piano fisico dipendono dallo schema fisico esistente (org primaria e indici).
Operatori fisici:

- Accesso alle relazioni di base
- Operazioni di relazione
- Operazioni di Join

1.1 Accesso alle relazioni di base

Il nostro piano sarà un albero (PQP). Le foglie sono le relazioni, l'accesso alle foglie avviene tramite cammini di accesso.

1.1.1 Cammini di accesso

Descrive come accedere al file dei dati di una relazione.

- Scansione sequenziale
- Indice più una condizione di selezione (Predicato di ricerca): utilizzabile solo se esiste una condizione di selezione che può essere eseguita usando l'indice

Esempio:

```
SELECT * FROM R WHERE R.A=5
```

$$\sigma_{A=5}$$
$$|$$
$$\text{R}$$

Può essere implementato in due modi:

- Accesso sequenziale

- alla scansione sequenziale si aggiunge $(I_A(R), A = 5)$

Ogni cammino di accesso diventa un nodo foglia.

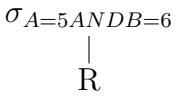
```
SEQ SCAN R Filter (A=5)
INDEX SCAN (I_{A}(R),A=5)
```

Esempio:

```
SELECT * FROM R WHERE R.A <> 5
```

Non posso usare l'indice, quindi ho solo la scansione sequenziale. Perchè io devo trovare tutte le tuple che soddisfano la condizione, ma non posso sapere a priori quali sono. **Esempio:**

```
SELECT * FROM R WHERE R.A=5 AND R.B=6
```

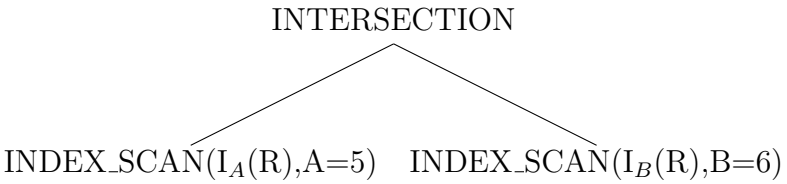



```
SEQ SCAN R Filter (A=5 AND B=6)
INDEX SCAN (I_{A}(R),A=5) Filter (B=6)
INDEX SCAN (I_{B}(R),B=6) Filter (A=5)
```

Dopo avere acceduto la relazione, è necessario filtrare (scartare) le tuple che non soddisfano l'altro congiunto. Per accedere una singola relazione, può anche essere scelto più di un cammino di accesso, combinando poi i risultati con un ulteriore operatore fisico. **Esempio:**

```
SELECT * FROM R WHERE R.A=5 AND R.B=6
INDEX SCAN (I_{A}(R),A=5)
INDEX SCAN (I_{B}(R),B=6)
```

E poi i risultati delle due scan vengono intersecati.



Con l'**OR** è simile ma si usa l'unione.

1.1.2 Costi

- Costo di accesso: numero di pagine di blocco lette
- Costo di selezione: tipo di indice e numero di tuple che soddisfano il predicato

Costo totale = (costo di determinare i riferimenti ai dati che soddisfano la condizione, in organizzazione secondaria (1 per hash, $h(\log n)$ per albero)) + (costo per accedere ai corrispondenti blocchi dei dati in organizzazione primaria, la clusterizzazione influisce).

- Indice ordinato clusterizzato: Ogni blocco dati viene letto al più una volta
- Indice ordinato non clusterizzato: Un blocco può essere acceduto più volte
- Indice non ordinato clusterizzato: Un blocco può essere acceduto più volte, i puntatori ai blocchi vengono ordinati. (Operatore BITMAP)

1.2 Operazioni di relazione

1.2.1 Selezione

Cosa succede se eseguo una selezione su un risultato intermedio?
La selezione può essere implementata solo in modo sequenziale perchè non è possibile utilizzare un indice su un risultato intermedio.

1.2.2 Join

E' un'operazione costosa: richiede $T(R) \cdot T(S)$ confronti (dove $T(R)$ e $T(S)$ sono il numero di tuple delle due relazioni). Per evitare tutti i possibili confronti:

- Nested Loop Join: Si accede ad una tupla di R (outer relation) e si confronta con ogni tupla di S (inner relation). L'ordine corrisponde all'ordine delle tuple della relazione esterna. Costi: $B(R) + T(R) \cdot B(S)$. Conviene considerare come relazione outer la relazione più grande.
- Index Nested Join: Come il nested loop join, ma si utilizza un indice su S per accedere alle tuple e si usa solo se esiste l'indice. Costi: $B(R) + T(R)$·costo di accesso a S.
- Merge Join: applicabile quando le relazioni in input sono ordinate rispetto all'attributo di join. L'algoritmo sfrutta il fatto che entrambi gli input sono ordinati rispetto all'attributo di join. Prima avviene l'ordinamento delle relazioni, poi si scansionano le relazioni in parallelo e si fanno i confronti tupla per tupla.
- Hash Join: applico la funzione hash alle tuple delle due relazioni e le metto in un bucket. Se due tuple hanno lo stesso hash, vengono messe nello stesso bucket. Si calcola il join tra i bucket. Sicuramente due tuple in due bucket diversi non possono essere in join, se sono nello stesso bucket **potrebbero** essere in join.

$B(\cdot)$ è il numero di blocchi letti.

Operatore fisico	Vincolo su schema fisico	Costo
Iterazione semplice	Nessuno	B(R) + T(R) * B(S)
Iterazione orientata ai blocchi	Nessuno	B(R) + B(R) * B(S)
Index nested loop	Indice su attributo di join della relazione inner	B(R) + T(R)* Costo di accesso alla relazione interna S con cammino di accesso (I_S(A), S.A θ tr.A)
Merge join	Relazioni devono essere ordinate rispetto all'attributo di join	B(R) + B(S)
Hash join	Nessuno	Nell'ordine di B(R) + B(S)

1.3 Condizioni più generali

- Uguaglianza su più di un attributo
- Theta-join con disuguaglianza