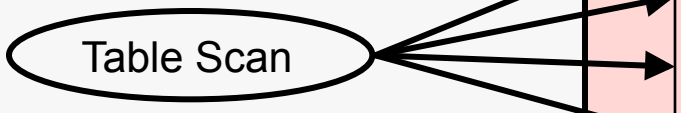
Physical Database Tuning with Interaction-Aware Index Selection

Karl Schnaitter
UC Santa Cruz
School of Engineering
Advisor: Neoklis Polyzotis
October 20, 2009

Physical Database Tuning

```
SELECT b FROM R WHERE a = 10
```

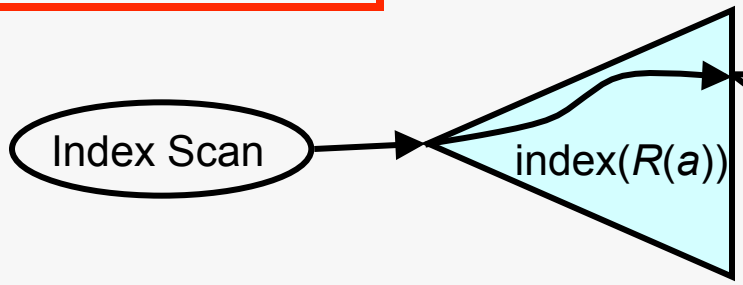
Execution Plan 1



R	*a*	*b*
	5	"blue"
	10	"red"
	97	"green"
	10	"blue"
	. . .	. . .

- Reads whole table
- Discards rows where a ≠ 10
- Returns *b* from remaining rows

Execution Plan 2



R	*a*	*b*
	5	"blue"
	10	"red"
	97	"green"
	10	"blue"
	. . .	. . .

- Only accesses rows where a = 10
- May be much faster than a table scan

Index Selection

- Indexes can reduce query execution time by orders of magnitude
  - Unfortunately, the best indexes are hard to choose
- The *index selection problem*:
  - Given a query workload
  - Choose indexes that improve workload performance
  - May have a limit on disk space

This Talk

- Will discuss two topics in index selection
  1. Understanding index "interactions"
     - Work published in the VLDB 2009 conference

       K. Schnaitter, N. Polyzotis, L. Getoor, "Index Interactions in Physical Design Tuning: Modeling, Analysis, and Applications"
     - More details will be in the poster session
  2. On-line index management
     - A variant on the index selection problem
     - We will see how index interactions play a role

This Talk

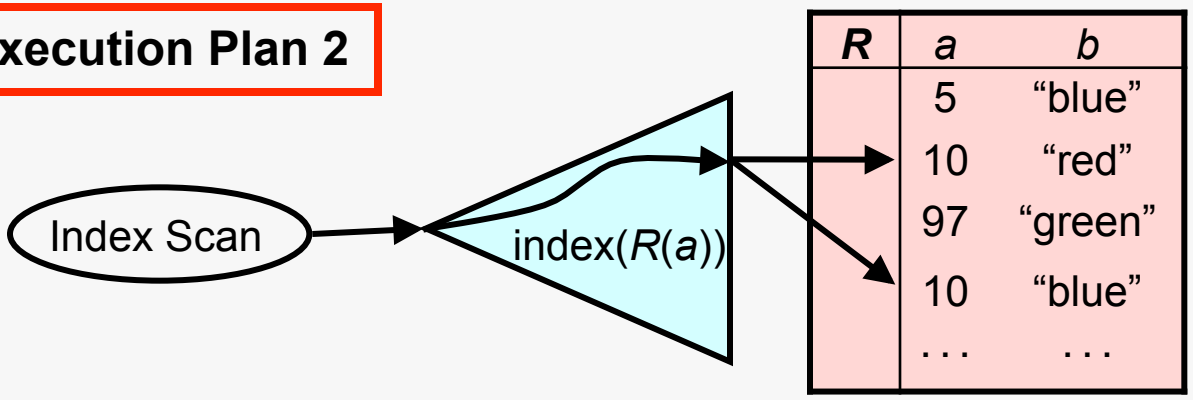
- Will discuss two topics in index selection
  1. Understanding index "interactions"
     - Work published in the VLDB 2009 conference

       K. Schnaitter, N. Polyzotis, L. Getoor, "Index Interactions in Physical Design Tuning: Modeling, Analysis, and Applications"
     - More details will be in the poster session
  2. On-line index management
     - A variant on the index selection problem
     - We will see how index interactions play a role

Example: Index Interaction

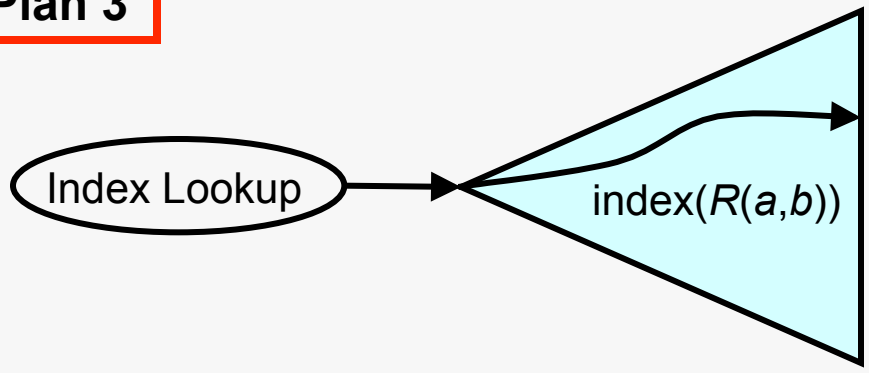
```
SELECT b FROM R WHERE a = 10
```

Execution Plan 2



- Only accesses rows where a = 10
- May be much faster than a table scan

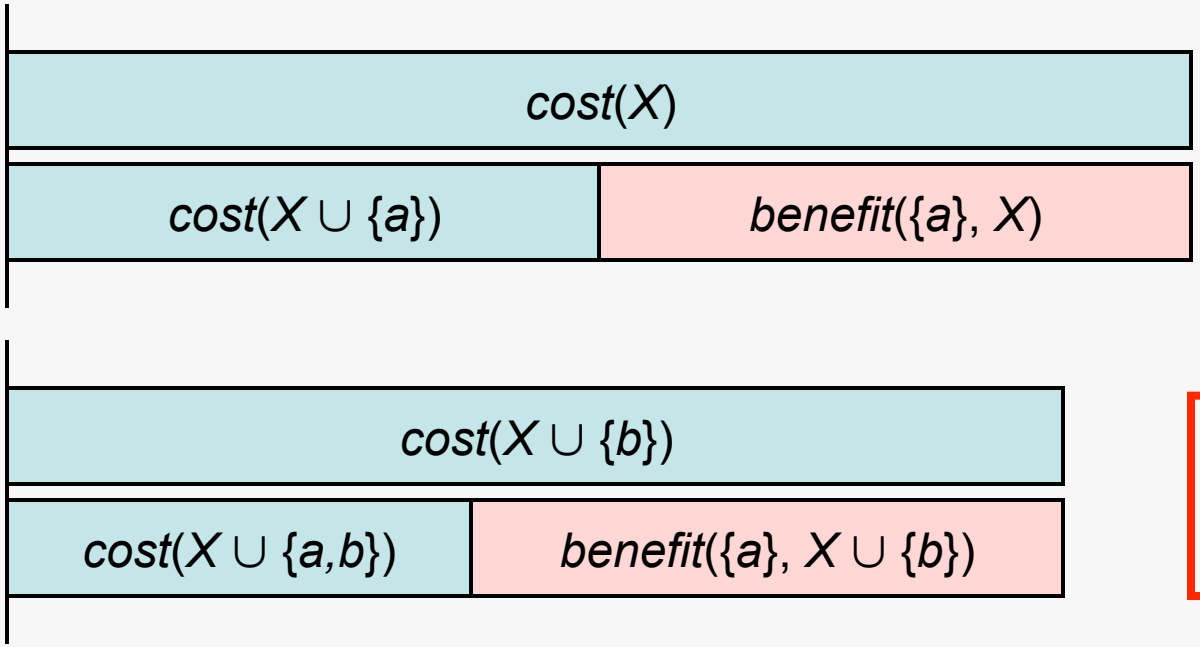
Execution Plan 3



- Only accesses index
- If plan 3 is cheaper, the index on R.a becomes obsolete

Index Interactions

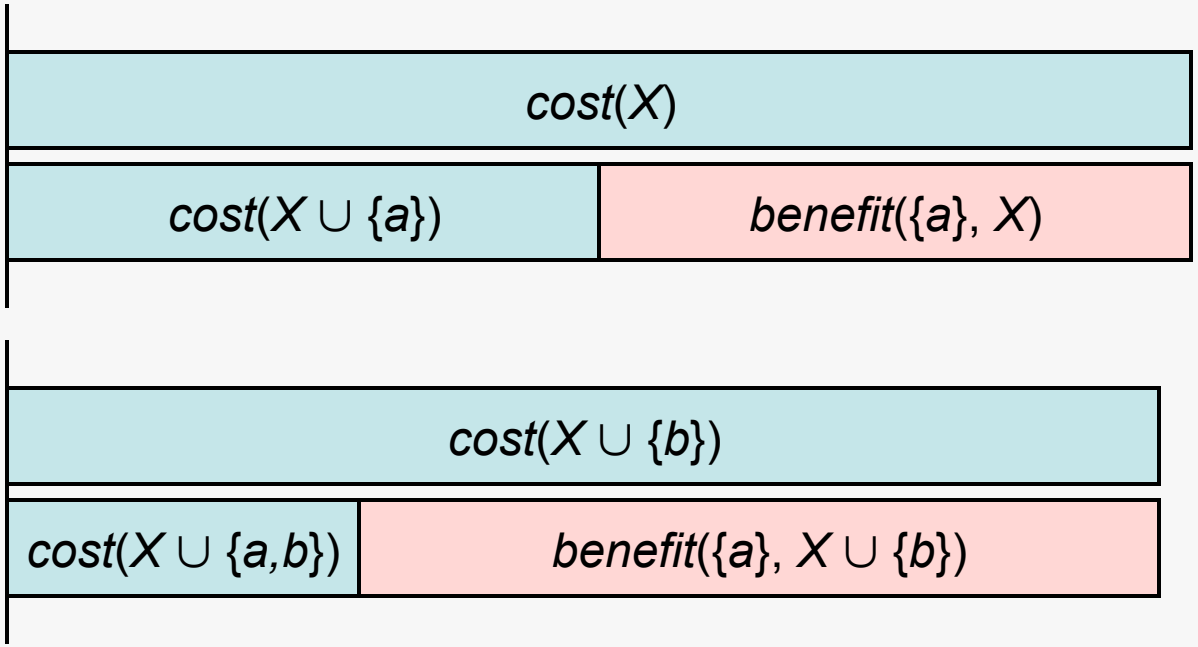
- Let **S** be a set of indexes relevant to a query **Q**
- $cost(X)$ = cost of **Q** if only $X \subseteq \mathbf{S}$ is available
- $benefit(Y,X) = cost(X) - cost(Y \cup X)$



Indexes *a*,*b* are *independent* with respect to *X*

Index Interactions

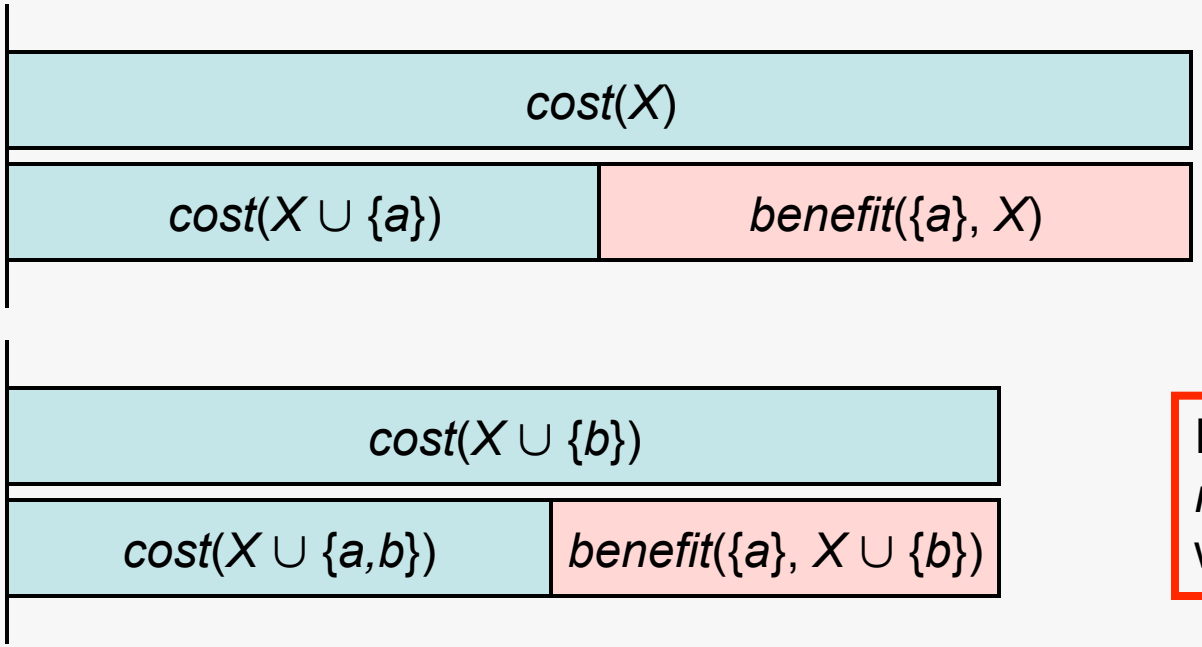
- Let **S** be a set of indexes relevant to a query **Q**
- $cost(X)$ = cost of **Q** if only $X \subseteq \mathbf{S}$ is available
- $benefit(Y,X) = cost(X) - cost(Y \cup X)$



Indexes *a*,*b* *positively interact* with respect to *X*

Index Interactions

- Let **S** be a set of indexes relevant to a query **Q**
- $cost(X)$ = cost of **Q** if only $X \subseteq \mathbf{S}$ is available
- $benefit(Y,X) = cost(X) - cost(Y \cup X)$

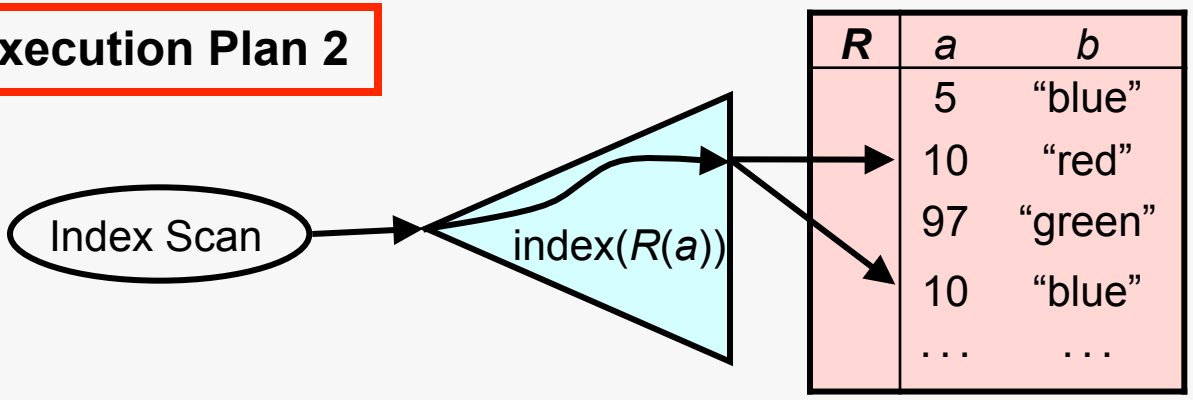


Indexes *a*,*b* *negatively interact* with respect to *X*

Example: Index Interaction

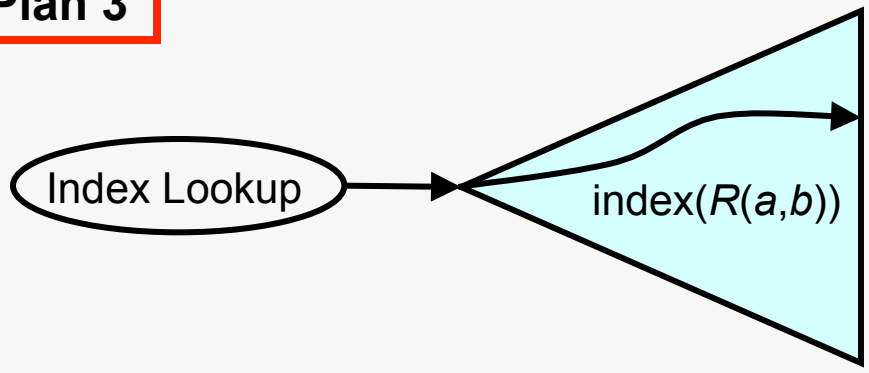
```
SELECT b FROM R WHERE a = 10
```

Execution Plan 2



- Only accesses rows where a = 10
- May be much faster than a table scan

Execution Plan 3



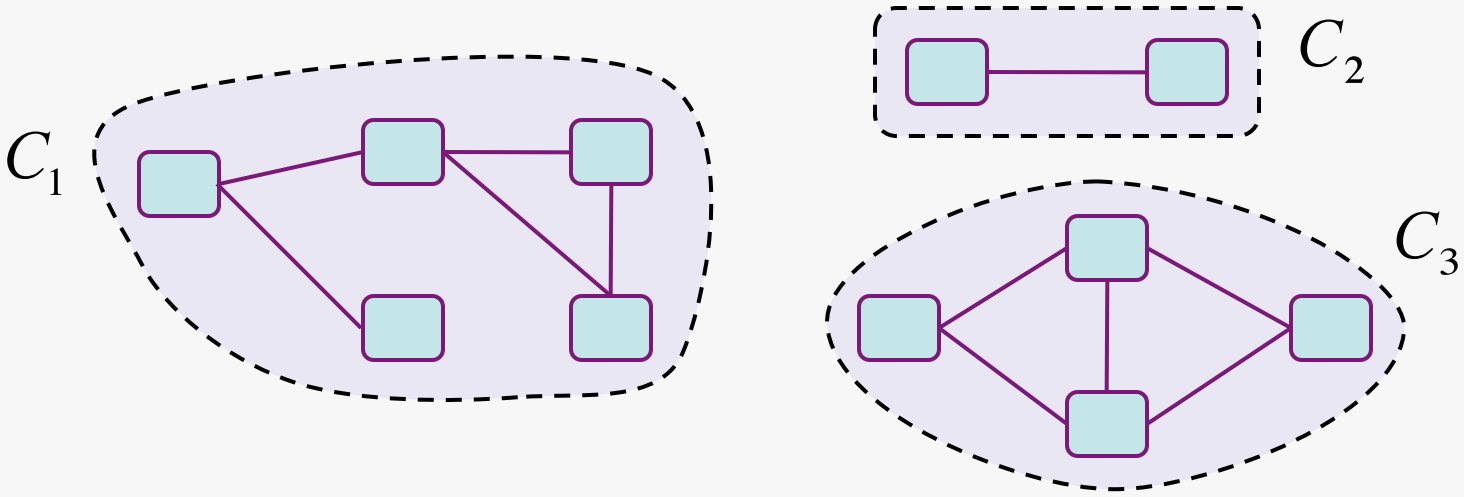
- Only accesses index
- If plan 3 is cheaper, the index on R.a becomes obsolete

Index Interactions

- Formally, we say *a interacts with b* if:

  $\exists X \subseteq \mathbf{S}$ such that $benefit(\{a\}, X) \neq benefit(\{a\}, X \cup \{b\})$

- This is a symmetric binary relation on indexes
  - Yields an undirected graph



The benefit of any $X \subseteq C_i$ does not depend on $\mathbf{S} - C_i$

Discovering Index Interactions

- We would like to know which edges exist
  - I.e., which pairs of indexes interact?
  - This is very hard in general
- Our algorithm
  - We impose some abstract assumptions on the use of indexes in query plans
  - These assumptions allow an efficient algorithm to find index interactions
  - See our poster for more information

This Talk

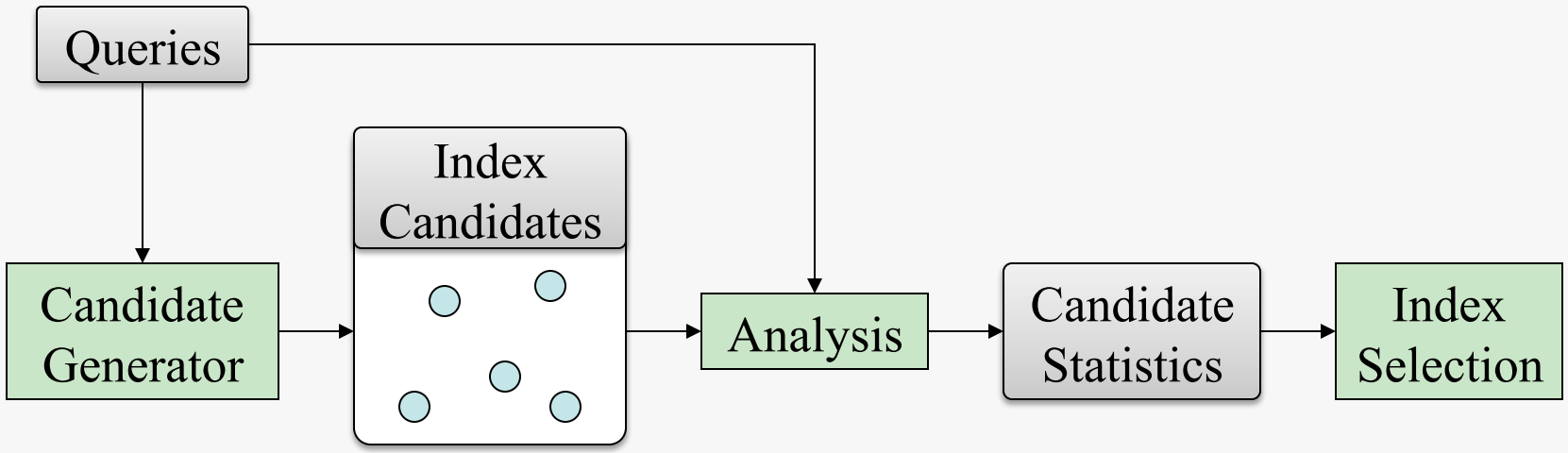
- Will discuss two topics in index selection
  1. Understanding index "interactions"
     - Work published in the VLDB 2009 conference

       K. Schnaitter, N. Polyzotis, L. Getoor, "Index Interactions in Physical Design Tuning: Modeling, Analysis, and Applications"
     - More details will be in the poster session
  2. On-line index management
     - A variant on the index selection problem
     - We will see how index interactions play a role

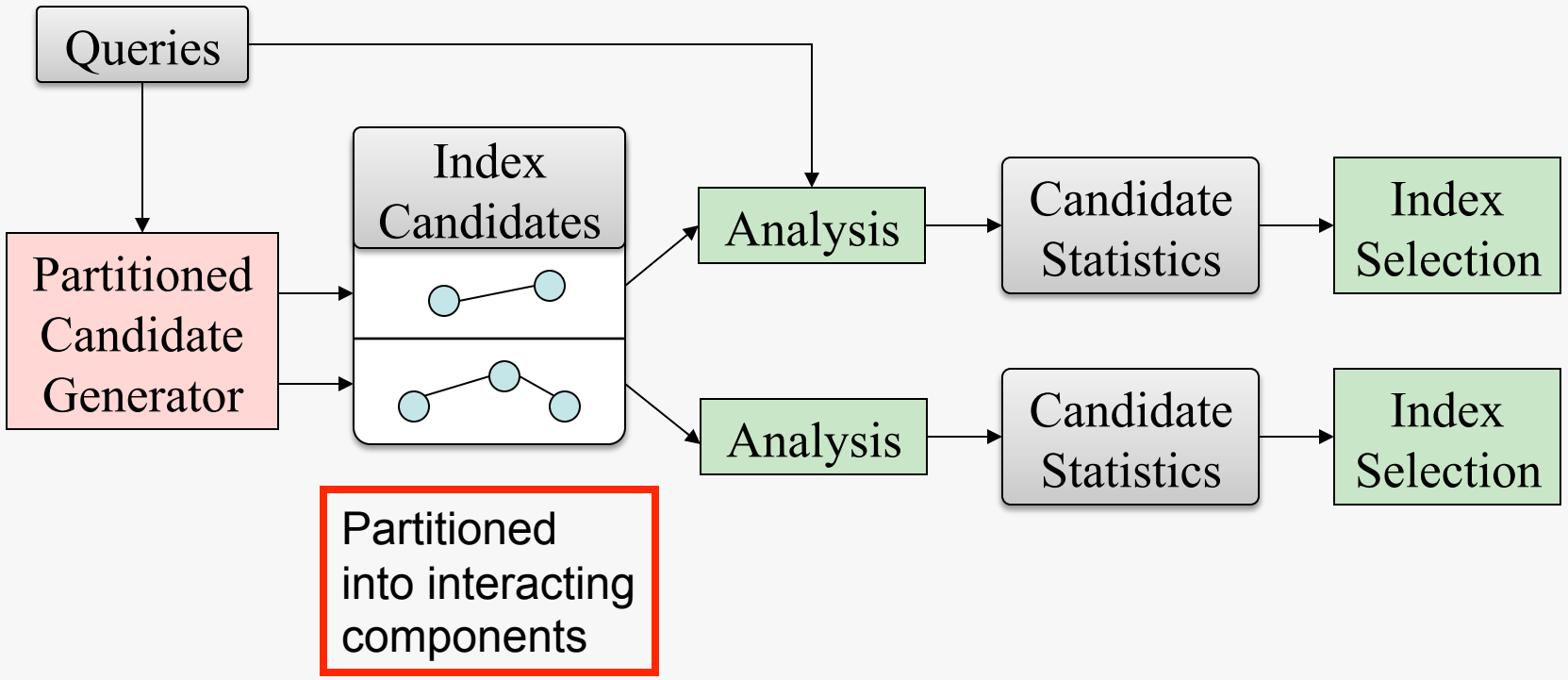
On-line Index Selection

- Recall the *index selection* problem:
  - Given a query workload
  - Choose indexes that improve workload performance
- This approach fails when the query workload is unknown or changing significantly over time
- Alternative approach: *on-line index selection*
  - Given continuous stream of queries
  - Choose indexes automatically to improve throughput

On-line Index Selection



On-line Index Selection



Summary

- Index selection is a challenging problem for database administrators
- On-line index selection is a promising approach
  - No advance knowledge of workload required
  - Can adapt to an evolving workload
- Knowledge of index interactions allows on-line tuning to be more intelligent at a large scale

Thank You