Wii: Dynamic Budget Reallocation in Index Tuning

- Xiaoying Wang (Simon Fraser University)
- **Wentao Wu (Microsoft Research)**
- Chi Wang (Microsoft Research)
- Vivek Narasayya (Microsoft Research)
- Surajit Chaudhuri (Microsoft Research)

The Architecture of Cost-based Index Tuning

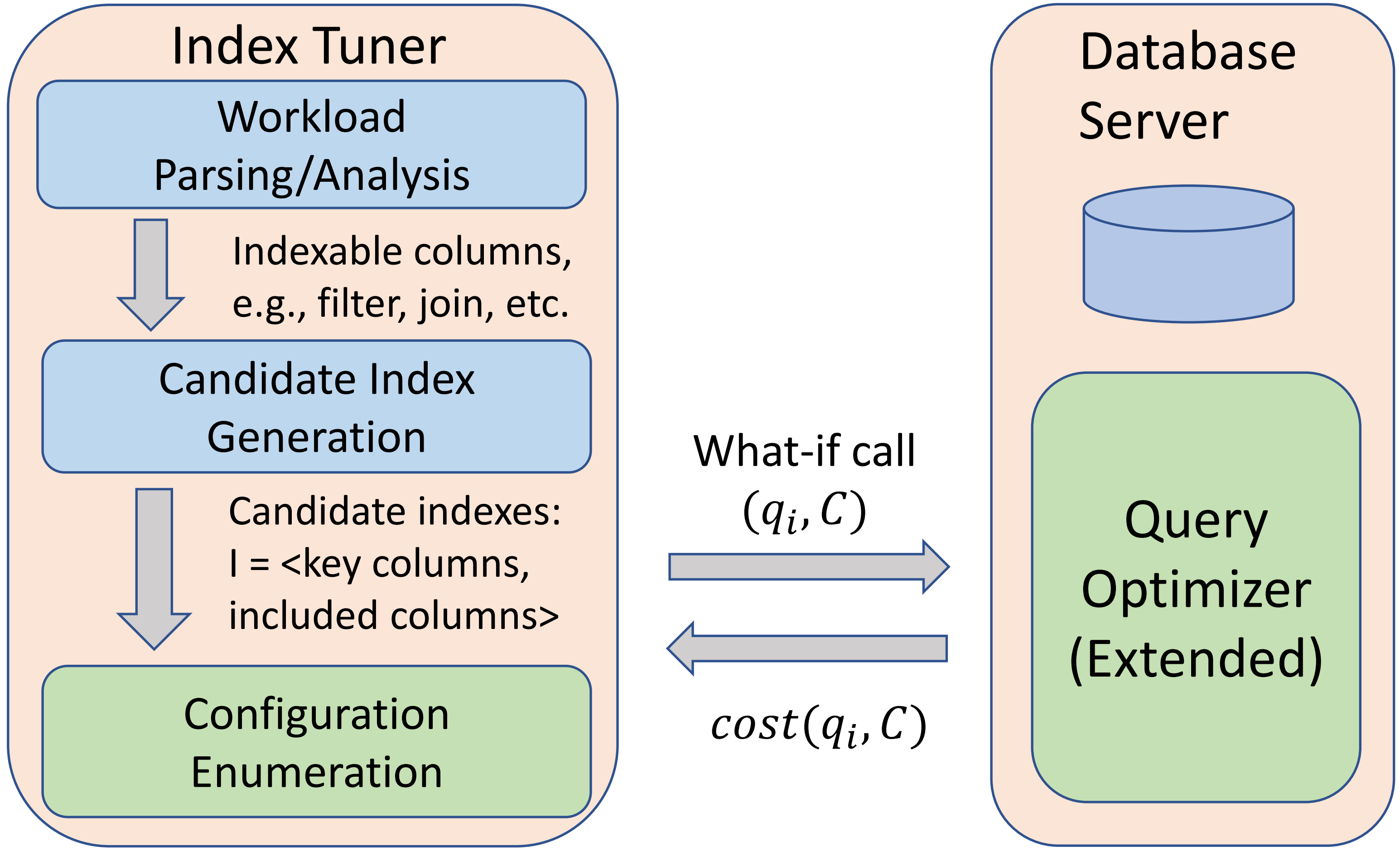
Input of Index Tuner:

(1) A workload, i.e., a set of SQL queries, $W = \{q_i\}$.
(2) A set of constraints Γ.

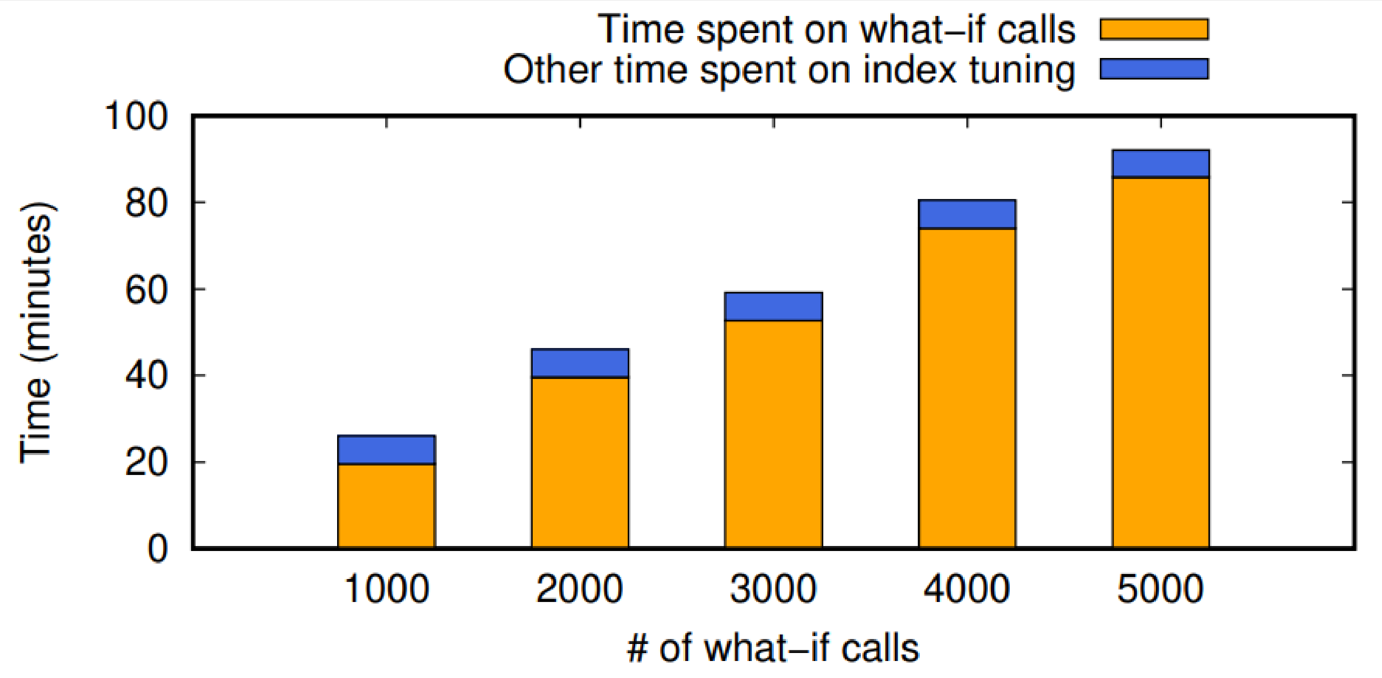
Output of Index Tuner:

The best configuration (i.e., subset) of indexes that minimizes the total workload cost:

$$cost(W, C) = \sum_{q_i \in W} cost(q_i, C)$$



What-if Calls are Expensive

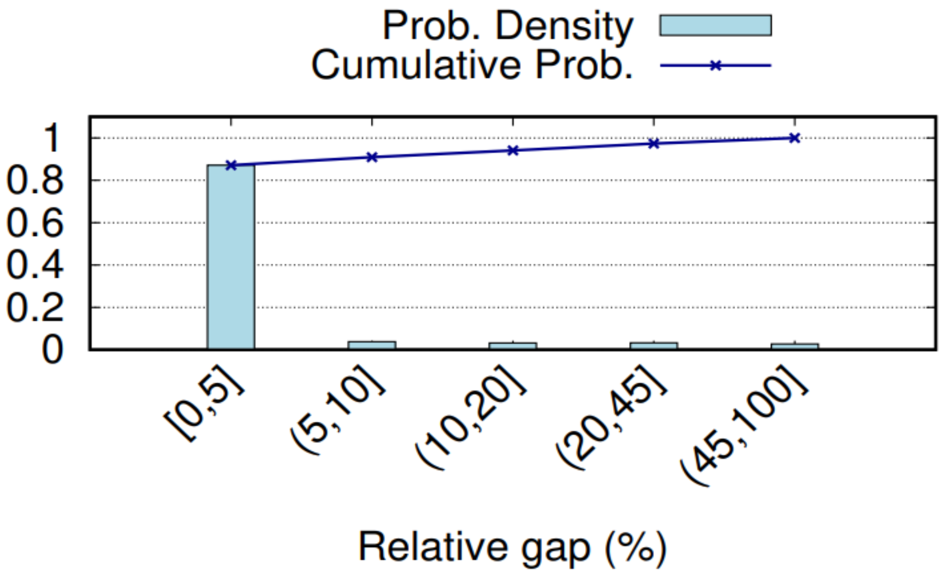


- A what-if call is as expensive as a regular query optimizer call.
- What-if calls dominate index tuning time.
  - TPC-DS, 99 queries, 20 recommended indexes
  - Greedy search (adopted by commercial index tuning software)

Many What-If Calls Are Inessential

Inessential What-if Call: The *relative gap* between *what-if cost* and *derived cost* is small.

Derived Cost: $d(q, C) = min_{S \subseteq C} cost(q, S)$, which is an upper-bound of $cost(q, C)$ if we assume that the cost function is *monotonic* (i.e., more indexes are better).



Distribution of Relative Gap for TPC-DS,

Relative Gap: $\frac{d(q,C) - cost(q,C)}{d(q,C)} \times 100\%$

Observations:

(1) 80% to 90% of the what-if calls made during index tuning are inessential.

(2) We can replace $cost(q, C)$ by $d(q, C)$ for inessential what-if calls to save the overhead of making these what-if calls.

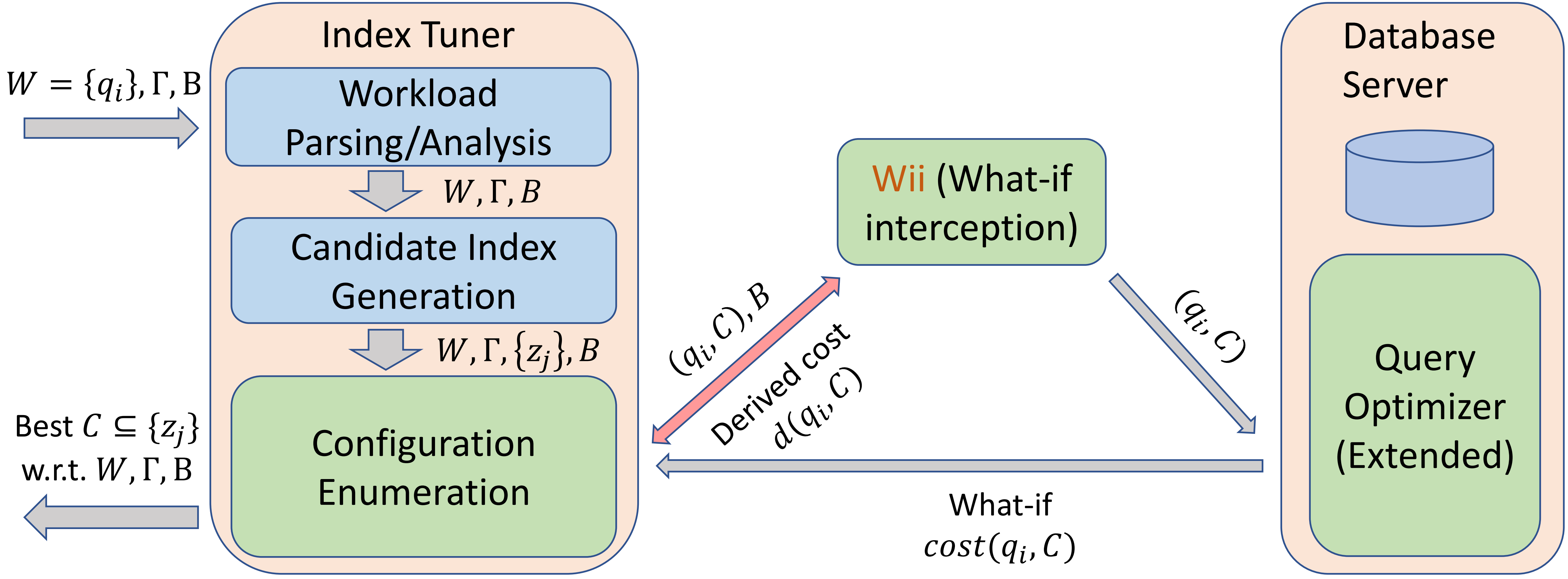
Can We Skip Inessential What-If Calls?

- **Challenge:** We need to skip an inessential what-if call *before* it is made.

- Idea #1: Skip a what-if call *randomly*?
  - **Benefit:** Almost zero overhead.
  - We will show that this approach does not work.

- Idea #2: Bounding what-if cost (this work):
  - Develop lower/upper bounds of the true what-if cost (which is *unknown* without making the call).
  - Skip a what-if call if the gap between the lower and upper bounds is *small enough*.

Wii: What-if (Call) Interception



Applications of Wii in Index Tuning

- Budget-aware Index Tuning (SIGMOD'22)
  - A budget on the number of what-if calls is given.
  - **Examples:** Greedy search, Monte Carlo tree search (MCTS)
  - Wii can save on inessential what-if calls and reallocate budget to what-if calls that Wii cannot skip.
  - Budget reallocation can find better index configuration.

- Index Tuning with Unlimited Budget (traditional)
  - Wii cannot help improve the configuration found.
    - The best configuration can be found by always making what-if calls given that budget is unlimited.
  - Instead, Wii can reduce overall index tuning time by avoiding making inessential what-if calls.

Summary of Technical Contributions

- Developed a *lower bound* $L(q, C)$ and an *upper bound* $U(q, C)$ of $cost(q, C)$.
  - The upper bound $U(q, C) = d(q, C)$.
  - The lower bound $L(q, C) = cost(q, \emptyset) - \sum_{z \in C} u(q, z)$.
    - Intuition: Lower bound of $cost(q, C)$ means the *maximum possible improvement* $u(q, C)$ of C.
    - Assuming *submodularity* of the cost function, $u(q, C) \leq \sum_{z \in C} u(q, z)$.

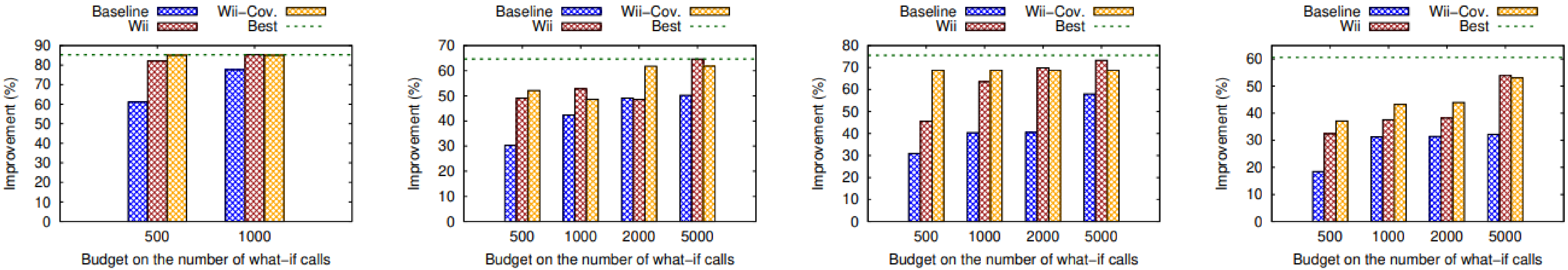
- Developed a *confidence-based mechanism* to skip a what-if call (q, C).
  - Define the *confidence* $\alpha(q, C) = \frac{L(q,C)}{U(q,C)}$.
  - Skip the what-if call (q, C) if $\alpha(q, C) \geq \tau$, where $0 \leq \tau \leq 1$ is some *threshold*.

- Developed a *coverage-based refinement* of the lower bound $L(q, C)$.
  - See the paper for the details.

Evaluation: Budget-aware Index Tuning (Index Quality)

Name	DB Size	# Queries	# Tables	Avg. # Joins	Avg. # Scans	# CandidateIndexes
TPC-H	sf=10	22	8	2.8	3.7	168
TPC-DS	sf=10	99	24	7.7	8.8	848
Real-D	587GB	32	7,912	15.6	17	417
Real-M	26GB	317	474	20.2	21.7	4,490

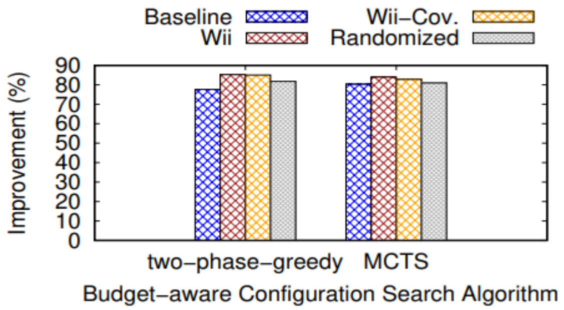
Set-up: *Greedy search*, with # of indexes allowed $K = 20$ and confidence threshold $\tau = 0.9$



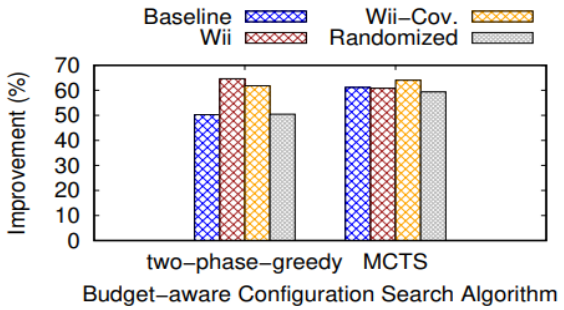
(a) TPC-H (b) TPC-DS (c) Real-D (d) Real-M

Evaluation: Budget-aware Index Tuning (vs. Random Skipping)

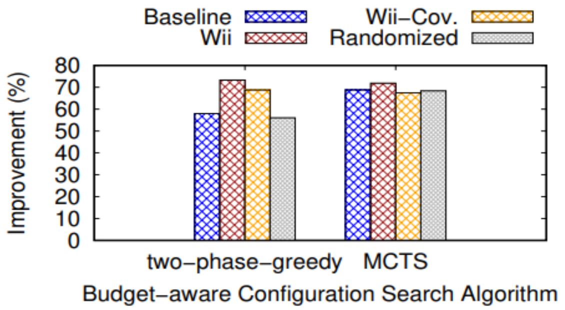
Set-up: # of indexes allowed $K = 20$, confidence threshold $\tau = 0.9$, random skipping probability $\mathrm{p} = \tau = 0.9$ (using the same skipping probability for fair comparison)



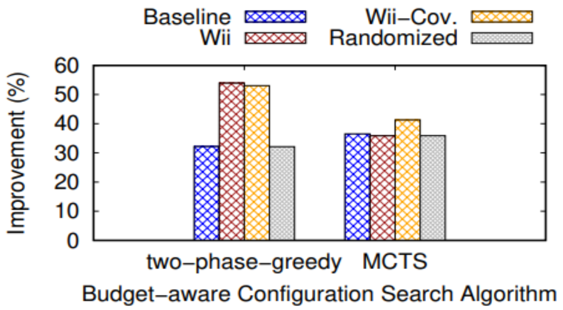
(a) TPC-H, B=1,000



(b) TPC-DS, B=5,000



(c) Real-D, B=5,000



(d) Real-M, B=5,000

Remark: The additional computation overhead introduced by Wii, while is higher than random skipping, remains negligible compared to the overall index tuning time.

Evaluation: Index Tuning with Unlimited Budget

- Wii cannot further improve the index quality achieved by *always making what-if calls*.
- However, Wii can significantly reduce tuning time by skipping inessential what-if calls.

TPC-H, $K = 20$				
Method	**Time** ($\alpha = 0.8$)	**Impr.** ($\alpha = 0.8$)	**Time** ($\alpha = 0.9$)	**Impr.** ($\alpha = 0.9$)
Baseline	8.22 min	85.22%	8.22 min	85.22%
Wii	1.62 min	84.74%	1.95 min	85.26%
Wii-Cov.	0.94 min	83.95%	1.67 min	85.02%

Real-D, $K = 5$				
Method	**Time** ($\alpha = 0.8$)	**Impr.** ($\alpha = 0.8$)	**Time** ($\alpha = 0.9$)	**Impr.** ($\alpha = 0.9$)
Baseline	380.63 min	62.32%	380.63 min	62.32%
Wii	118.95 min	64.10%	119.99 min	64.10%
Wii-Cov.	31.42 min	62.90%	53.38 min	59.63%

Conclusion

- We developed Wii, a lightweight what-if (call) interception mechanism that allows for skipping inessential what-if calls in index tuning.

- In budget-constrained index tuning, Wii can significantly improve index quality by reallocating budget to the what-if calls that cannot be skipped.

- For index tuning with unlimited budget, Wii cannot improve index quality but can significantly reduce index tuning time.

Thank you! If you have any questions, please contact:
wentao.wu@microsoft.com

Backup Slides

Why Can Budget Reallocation Help Improve Index Quality In Budget-aware Index Tuning?

Tables: R(a, b), S(c, d)

Query q:
SELECT a, d **FROM** R, S
WHERE R.b = S.c **AND** R.a = 40

Candidate Indexes:

$\boldsymbol{I_1}$ = [R.a; R.b] — For the filter predicate R.a = 40

$\boldsymbol{I_2}$ = [S.c; S.d] — For the join predicate R.b = S.c

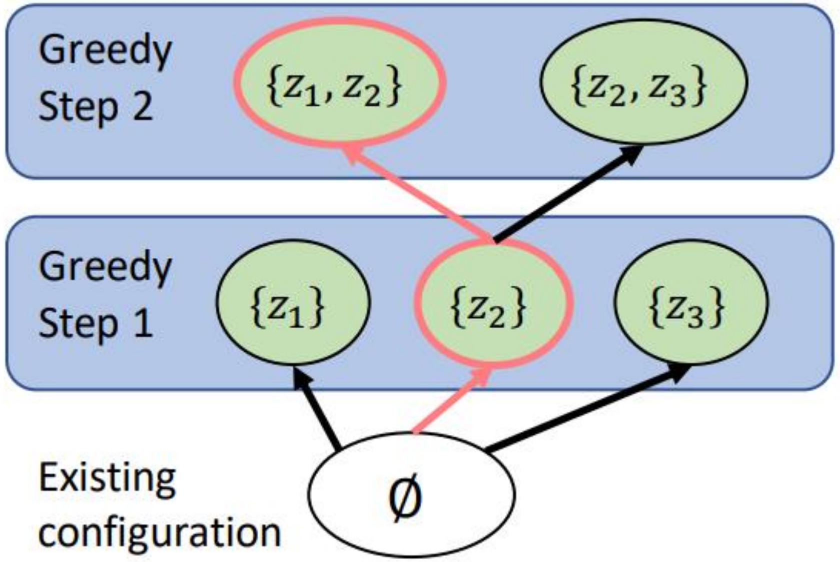
Budget Allocation: *Assume that the budget B = 1 (i.e., we can only make 1 what-if call) and K = 1 (i.e., we can only select one of the two indexes* $\boldsymbol{I_1}$ *and* $\boldsymbol{I_2}$ *for* **q***).*

1. There are two potential what-if calls to be made: (**q**, I_1), (**q**, I_2).
2. True what-if costs: c(**q**, I_1) = 180, c(**q**, I_2) = 50.
3. Derived costs: d(**q**, I_1) = 200, d(**q**, I_2) = 200.
4. If we make what-if call for (**q**, I_1), we will select I_1 and the cost of **q** is c(**q**, I_1) = 180.
5. If we make what-if call for (**q**, I_2), we will select I_2 and the cost of **q** is c(**q**, I_2) = 50.

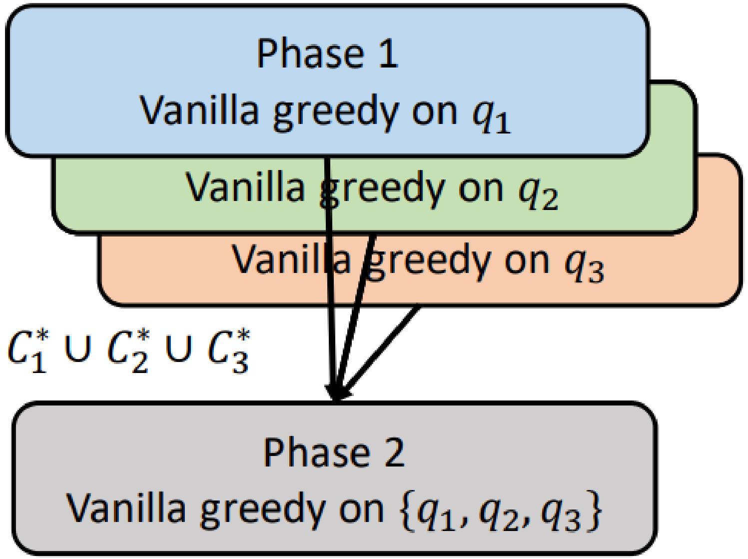
Budget-aware Index Tuning (SIGMOD’22)

- User may want to constrain the tuning time when face large/complex workloads.
  - Microsoft’s Database Tuning Advisor (DTA) allows user to specify a timeout.
- Under constrained tuning time, given the dominance of time on making what-if calls, we need to constrain the number of what-if calls made during configuration enumeration.
  - We call this constrained version of index tuning “budget-aware index tuning”.
  - **Budget allocation**: Decide which (q, C) pairs to make what-if calls.
- In previous work (SIGMOD’22), we studied the two types of budget-aware index tuning algorithms:
  - Greedy search (vanilla greedy search and two-phase greedy search)
  - Monte-Carlo tree search (MCTS)

Greedy Search

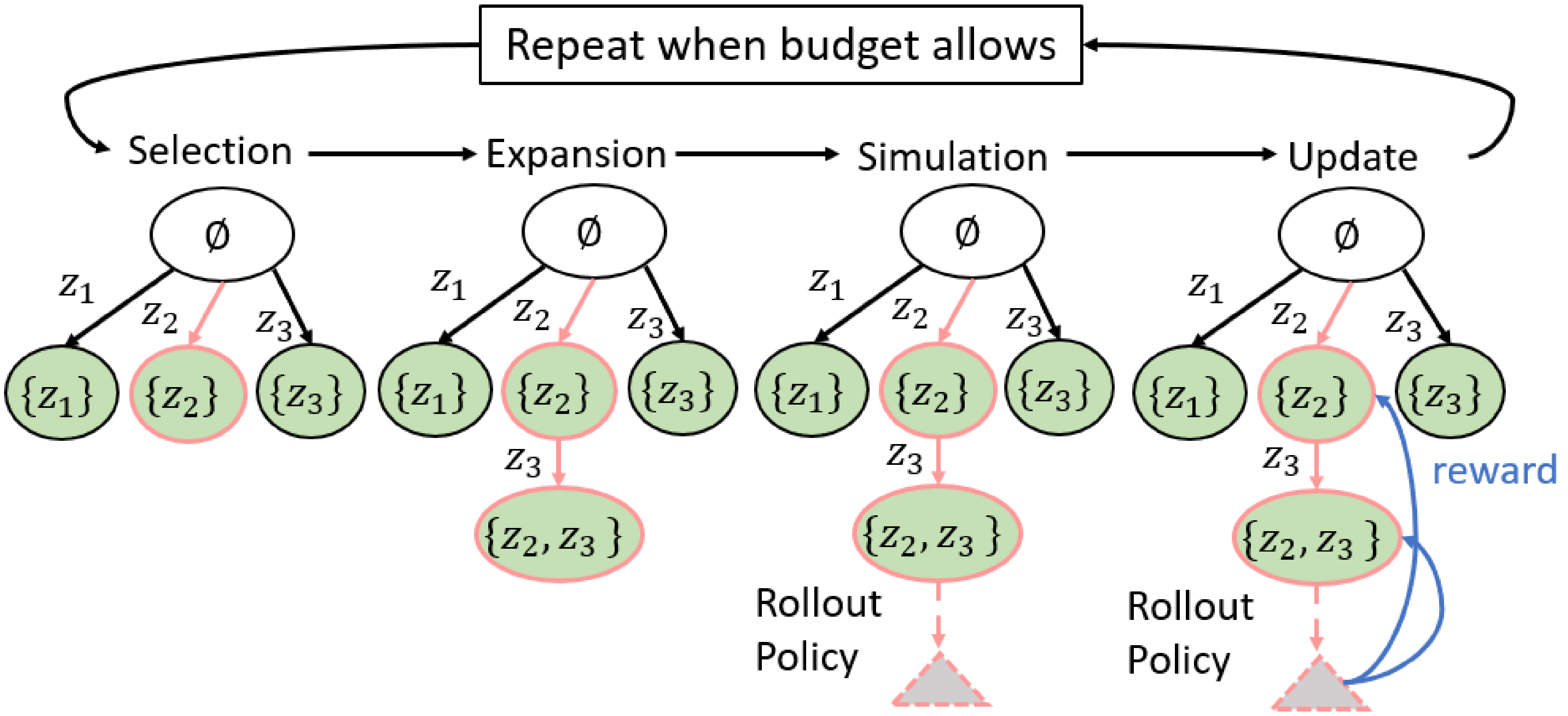


(a) Vanilla greedy search



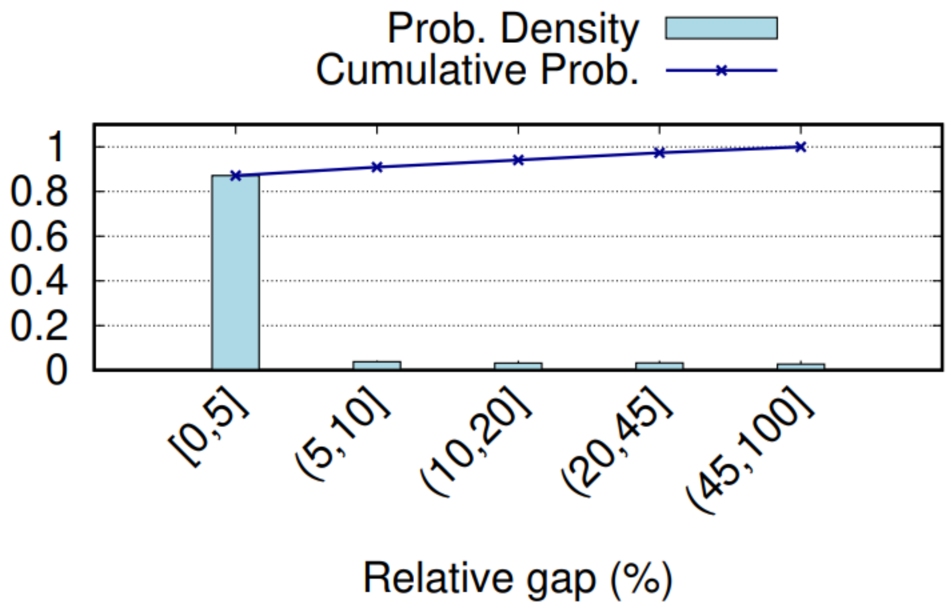
(b) Two-phase greedy search

Monte Carlo Tree Search (MCTS)

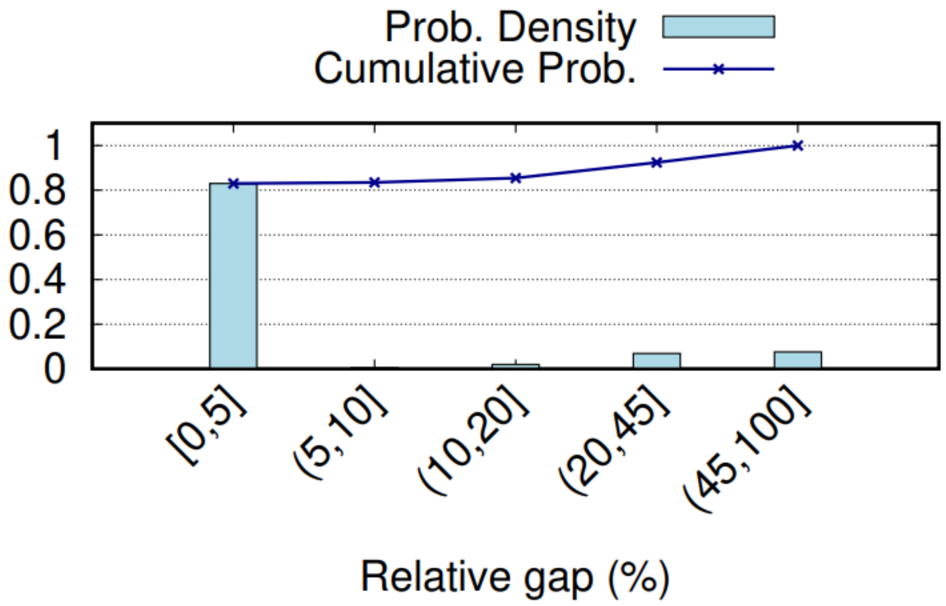


Spurious What-If Calls In Budget Allocation

Distribution of the *relative gap* between *what-if cost* and *derived cost*



(a) Two-phase greedy search



(b) MCTS

Derived Cost: $d(q, C) = min_{S \subseteq C} c(q, S)$

Lower and Upper Bounds

Assumption 1 (**Monotonicity**): Let C_1 and C_2 be two index configurations where $C_1 \subseteq C_2$. Then $c(q, C_2) \leq c(q, C_1)$.

Assumption 2 (**Submodularity**): Given two configurations $X \subseteq Y$ and an index $z \notin Y$, we have $c(q, Y) - c(q, Y \cup \{z\}) \leq c(q, X) - c(X \cup \{z\})$.

- Upper bound is set as the derived cost $U(q, C) = d(q, C) = min_{S \subseteq C} c(q, S)$.
- Lower bound is set as $L(q, C) = c(q, \emptyset) - \sum_{z \in C} u(q, z)$
    - $u(q, z)$ is the *upper bound* of the <u>marginal cost improvement</u> (MCI) of z.
    - Example 1: $u(q, z) = c(q, \emptyset) - c(q, \{z\}) = \Delta(q, \{z\})$
    - Example 2: $u(q, z) = c(q, \emptyset) - c\left(q, \Omega_{\mathrm{q}}\right) = \Delta(q, \Omega_q)$

Ω_{q} represents the "best possible configuration" with **ALL** candidate indexes of q included.

Confidence-based Skipping

- **Intuition:**
  - We are more confident to skip a what-if call if its lower and upper bounds are closer.

- Definition of confidence $\alpha(q, C)$:
  - $\alpha(q, C) = 1 - \frac{U(q,C)-L(q,C)}{U(q,C)} = \frac{L(q,C)}{U(q,C)}$
  - $0 \leq \alpha(q, C) \leq 1$

- Confidence-based skipping:
  - Define some threshold α s.t. $0 \leq \alpha \leq 1$.
  - Skip the what-if call (q, C) if $\alpha(q, C) \geq \alpha$.

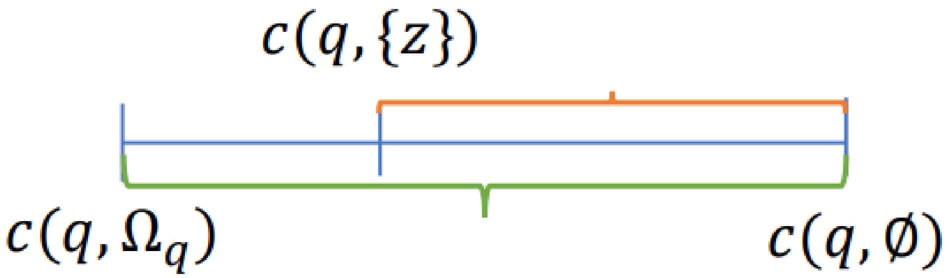
Coverage-based Refinement

- **Problem:** The MCI upper bound $u(q, z)$ used by the lower bound $L(q, C)$ heavily depends on the knowledge of singleton what-if cost $c(q, \{z\})$.
  - $u(q, z) = \Delta(q, \{z\}) = c(q, \emptyset) - c(q, \{z\})$, if we know $c(q, \{z\})$; otherwise, we can only use the "naïve bound" $u(q, z) = \Delta(q, \Omega_q) = c(q, \emptyset) - c(q, \Omega_q)$.

- **Solution:** Use "coverage" to *estimate* $c(q, \{z\})$, if it is not known.
  - Definition of "coverage": $\rho(q, z) = \frac{c(q,\emptyset) - c(q,\{z\})}{c(q,\emptyset) - c(q,\Omega_q)} = \frac{\Delta(q,\{z\})}{\Delta(q,\Omega_q)}$.
  - Estimation of "coverage" using similarity of $\{z\}$ and Ω_q: $\hat{\rho}(q, z) = Sim(\{z\}, \Omega_q)$.
  - See paper for more details.

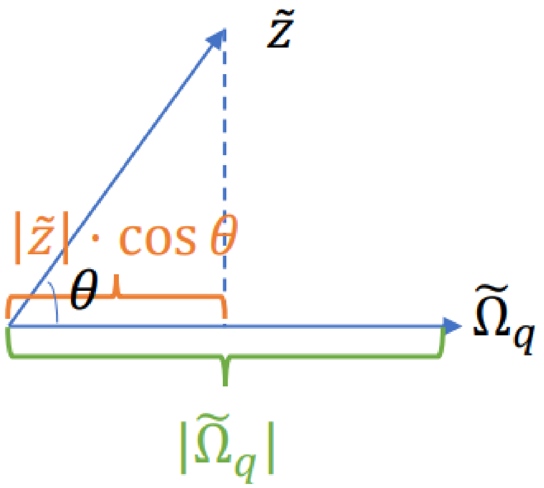
Geometric Interpretation of Coverage

$$\mathrm{Sim}(\{z\}, \Omega_q) = \frac{\langle \tilde{\mathbf{z}}, \tilde{\Omega}_q \rangle}{|\tilde{\Omega}_q|^2} = \frac{|\tilde{\mathbf{z}}| \cdot |\tilde{\Omega}_q| \cdot \cos\theta}{|\tilde{\Omega}_q|^2} = \frac{|\tilde{\mathbf{z}}| \cdot \cos\theta}{|\tilde{\Omega}_q|}$$



$$\rho = \frac{c(q, \emptyset) - c(q, \{z\})}{c(q, \emptyset) - c(q, \Omega_q)}$$

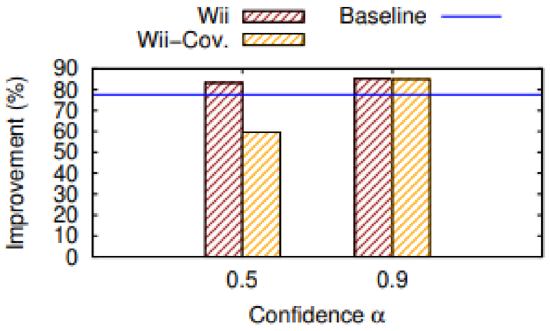
(a) Definition of “Coverage”



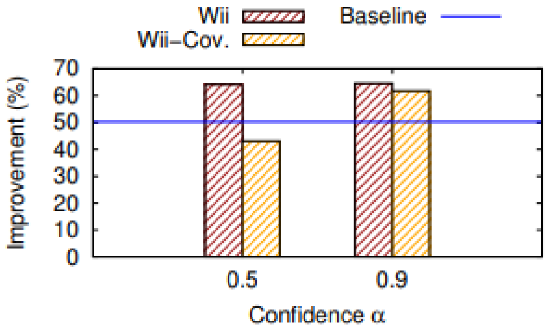
(b) Estimation of “Coverage”

Evaluation: Impact of Confidence Threshold α

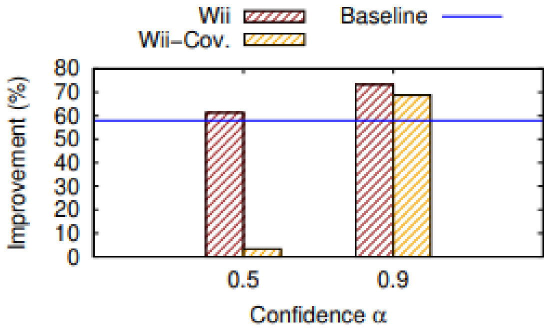
1. Results for ***two-phase greedy search***, with the number of indexes allowed $K = 20$



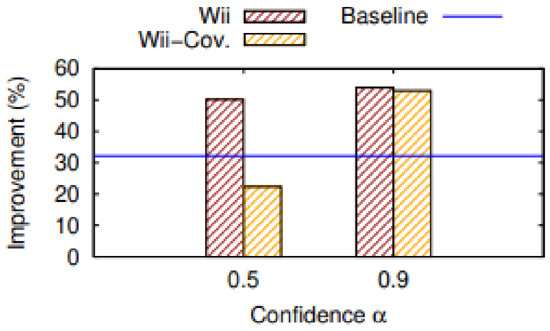
(a) TPC-H, B=1,000



(b) TPC-DS, B=5,000

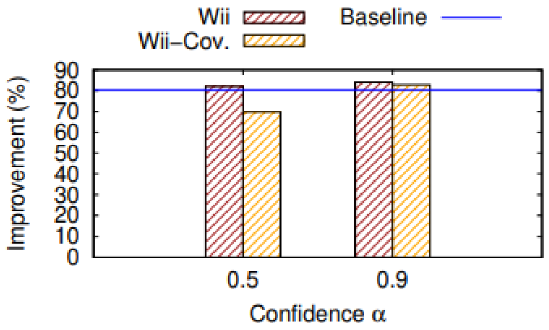


(c) Real-D, B=5,000

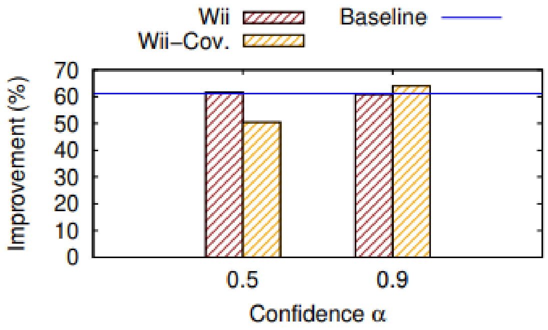


(d) Real-M, B=5,000

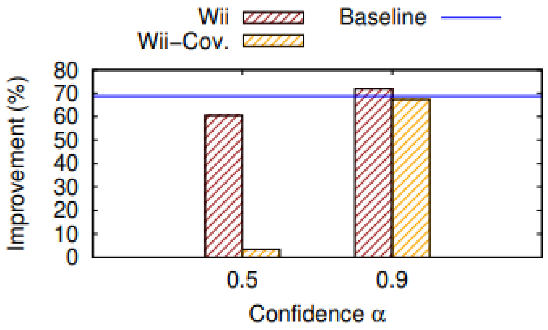
2. Results for ***MCTS***, with the number of indexes allowed $K = 20$



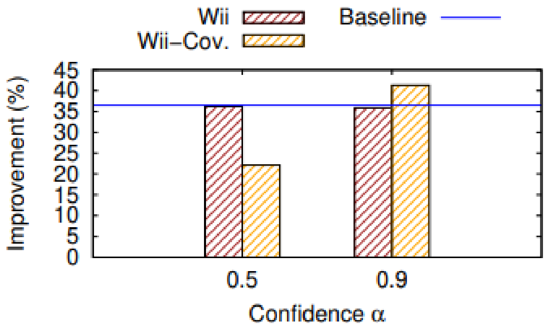
(a) TPC-H, B=1,000



(b) TPC-DS, B=5,000

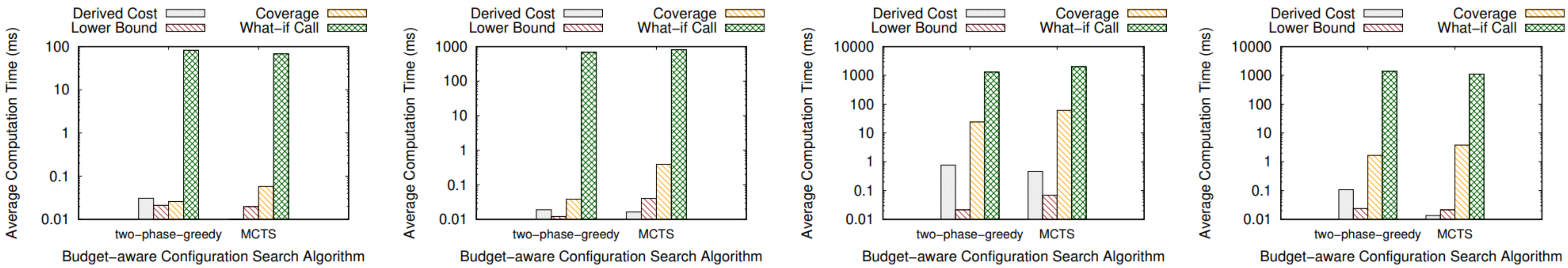


(c) Real-D, B=5,000



(d) Real-M, B=5,000

Evaluation: Budget-aware Index Tuning (Overhead)



(a) TPC-H, B=1,000 **(b) TPC-DS,** B=5,000 **(c) Real-D,** B=5,000 **(d) Real-M,** B=5,000

Additional overhead of Wii and Wii-Coverage, measured as percentage of the execution time of the baseline configuration search algorithm ($K = 20, \alpha = 0.9$).

Wii (Wii-Cov.)	*two-phase greedy*	*MCTS*
TPC-H ($B = 1,000$)	0.199% (0.273%)	0.064% (0.106%)
TPC-DS ($B = 5,000$)	0.016% (0.345%)	0.015% (0.164%)
Real-D ($B = 5,000$)	0.087% (2.354%)	0.029% (3.165%)
Real-M ($B = 5,000$)	0.055% (2.861%)	0.003% (2.544%)