GPTuner: A Manual-Reading Database Tuning System via GPT-Guided Bayesian Optimization

연세대학교 컴퓨터과학과 권세인

2024년 9월

SW STAR LAB
Software Technology Advanced Research

과제명: IoT 환경을 위한 고성능 플래시 메모리 스토리지 기반 인메모리 분산 DBMS 연구개발

과제번호: 2017-0-00477

과학기술정보통신부 Ministry of Science and ICT
연세대학교 YONSEI UNIVERSITY
IITP 정보통신기술진흥센터 Institute for Information & communications Technology Promotion

GPTuner: A Manual-Reading Database Tuning System via GPT-Guided Bayesian Optimization

Jiale Lao
Sichuan University
solidlao.jiale@gmail.com

Yibo Wang
Sichuan University
wangyibo.cs@gmail.com

Yufei Li
Sichuan University
liyufeievangeline@gmail.com

Jianping Wang
Northwest Normal University
2022222119@nwnu.edu.cn

Yunjia Zhang
University of Wisconsin-Madison
yunjia@cs.wisc.edu

Zhiyuan Cheng
Purdue University
cheng443@purdue.edu

Wanghu Chen
Northwest Normal University
chenwh@nwnu.edu.cn

Mingjie Tang*
Sichuan University
tangrock@gmail.com

Jianguo Wang
Purdue University
csjgwang@purdue.edu

International Conference on Very Large DataBases (VLDB 2024)

GPTuner

- Limitation

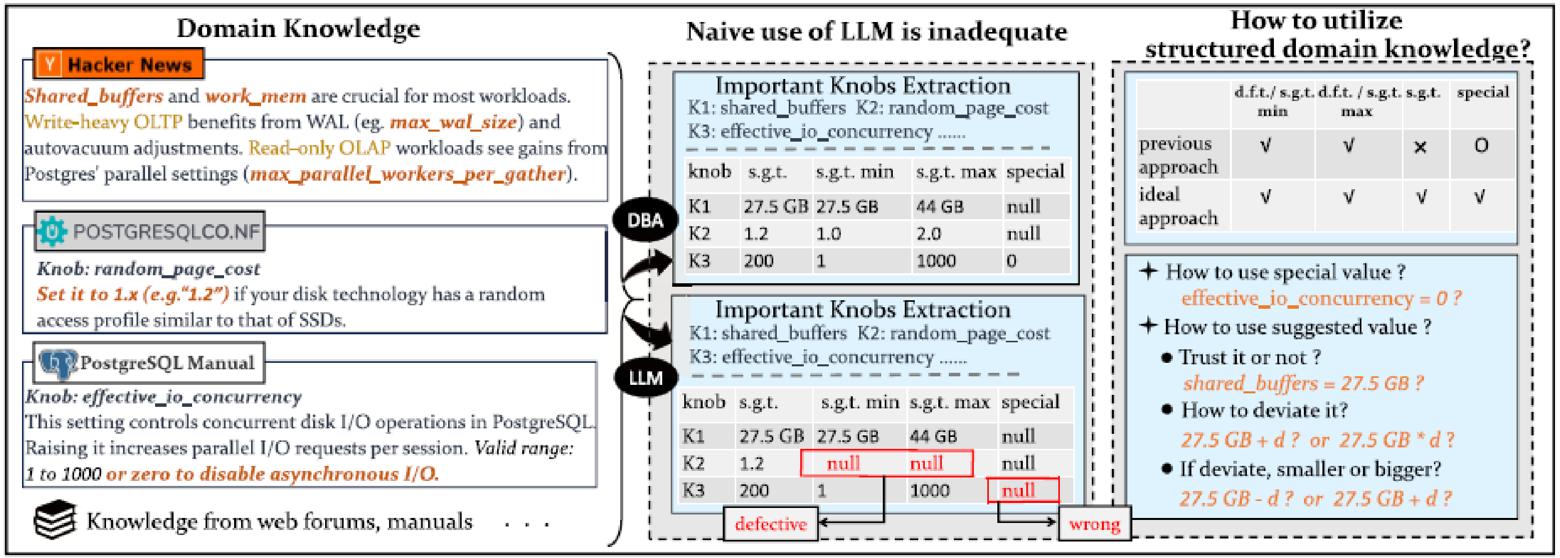
- Previous studies still **require hundreds to thousands iterations** to reach an ideal configuration, such high tuning costs stem form their inefficiency in handling.
- Fixed subset of parameters, sacrificing the flexibility to choose workload-relevant parameters, or execute workloads numerous times to identity important parameters.
- There are typical value ranges summarized for knobs.

Table 1: Tuning Knowledge Utilization

Knob	shared_buffers	random_page_cost
Default Range	[0.125MB, 8192 GB]	$[0,\ 1.79769 \times 10^{308}]$
Guidance	"shared_buffers" can be 25% of the RAM but no more than 40% . . . [39]	"random_page_cost" can be 1.x if disk has a speed similar to SSDs . . . [41]
DBA	The machine has a 16 GB RAM. Thus we can set "shared_buffers" from 16 GB × 25% = 4 GB to 16 GB × 40% = 6.4 GB.	The machine uses SSDs as disks. Thus we can set "random_page_cost" to a value from 1.0 to 2.0.
Improved Range	[4 GB, 6.4 GB]	[1.0, 2.0]

GPTuner

- Motivation



* "s.g.t." means "suggested" and "d.f.t." means "default", "o" means some approach handles this value manually

① Extensive tuning knowledge helps, but not well-exploited. (Left part)

② LLM is a notable step forward, but not adequate yet. (Middle part)

③ The lack of a knowledge-aware optimization framework. (Right part)

GPTuner

- Contribution

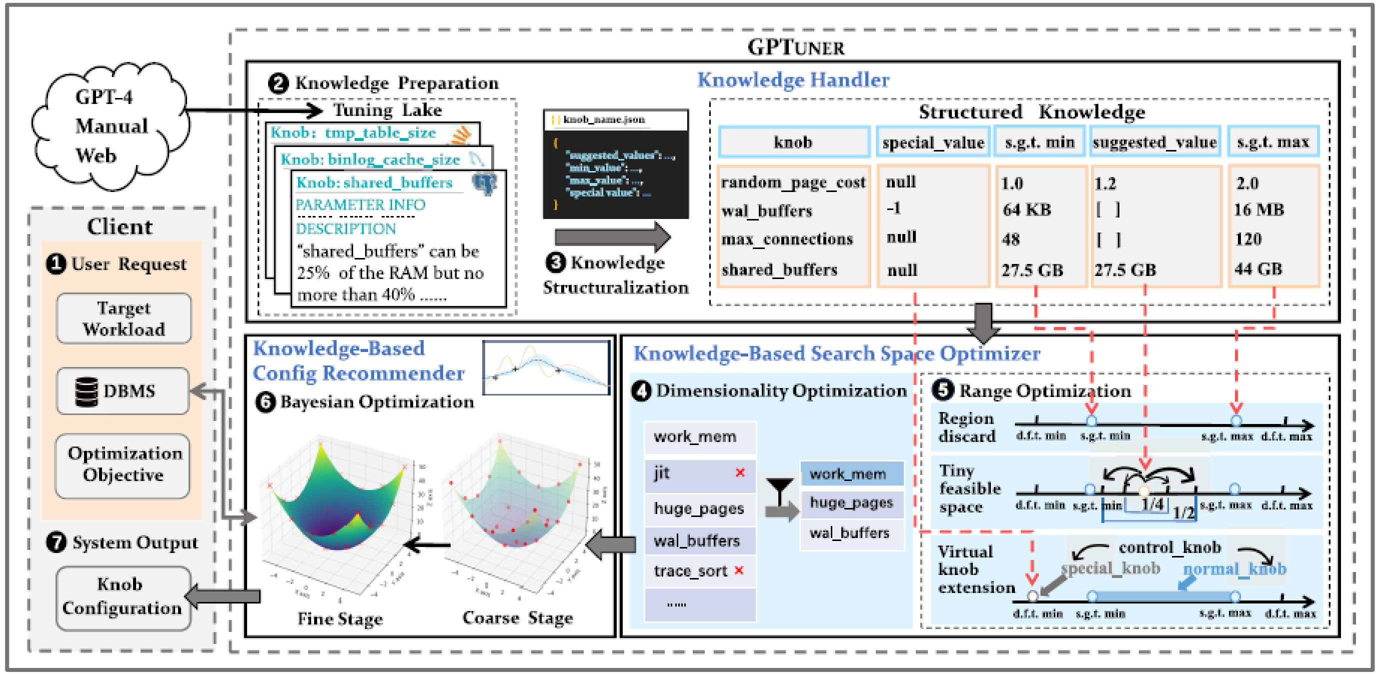
① GPTuner, a novel **manual-reading database tuning system** that leverages domain knowledge automatically and extensively to enhance the knob tuning process.

② Develop an LLM-based pipeline to **collect and refine domain knowledge**, and **propose a prompt ensemble algorithm** to unify a structured view of the refined knowledge.

③ **Workload-aware and trining-free knob selection strategy**, develop an optimization method for the value range of each knob, and propose a **Coase-to-Fine Bayesian Optimization** framework to explore the optimized space.

GPTuner

- Method



① User provides the DBMS to be tuned the target workload, and the optimization objective.

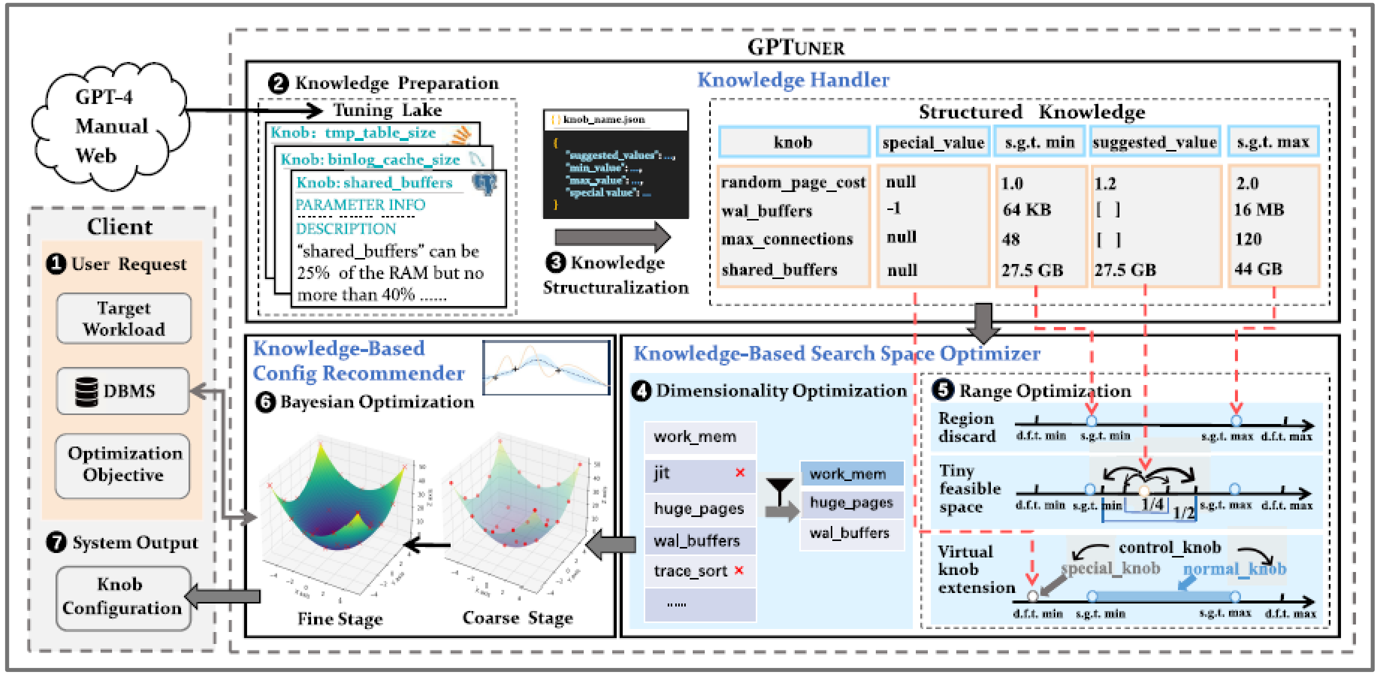
② GPTuner collects and refines the knowledge from different source to **construct Tuning Lake**.

③ Unifies the refined tuning knowledge from Tuning Lake into a **structured view** accessible to machines.

④ GPTuner reduces the search space dimensionality by **selecting important knobs** to tune.

GPTuner

- Method



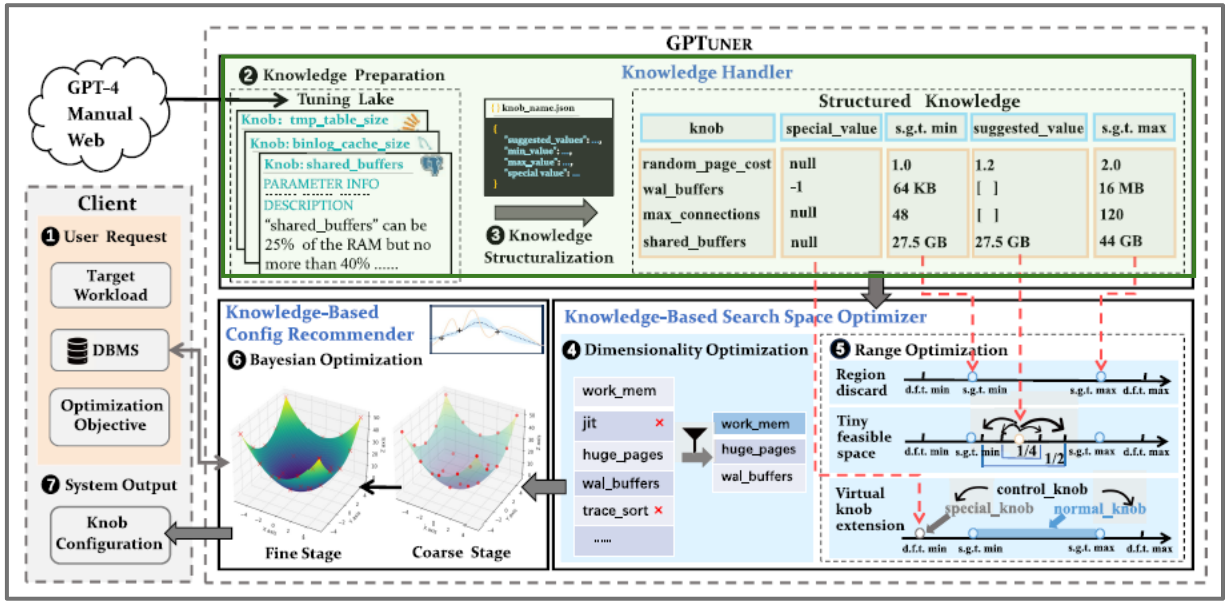
⑤ GPTuner **optimizes the search space** in terms of the value range for each knob based on structured knowledge.

⑥ GPTuner explores the optimized space via a novel **Coarse-to-Fine Bayesian Optimization** framework.

⑦ Identifies satisfactory knob configurations within resource limits.

GPTuner

- Method – Knowledge Handler



▪ Knowledge Preparation

① Extracting knowledge from LLM.

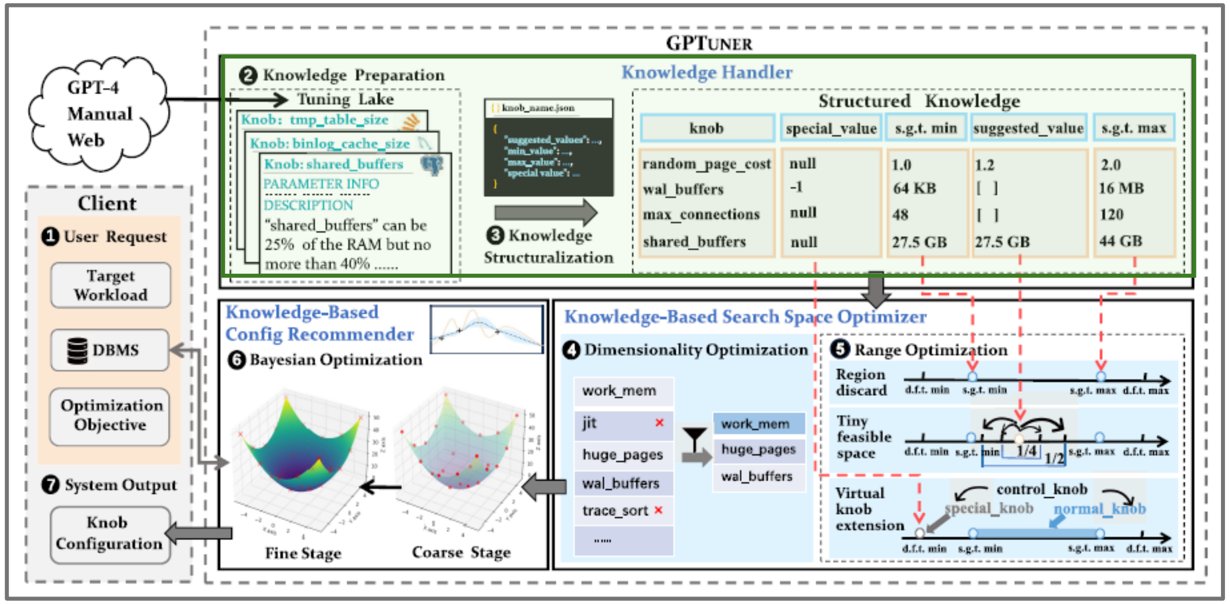
GPT is trained on a vast corpus related to database, GPT itself is an informative manual and allows to retrieve the knowledge through prompt.

② Filtering noisy knowledge.

With candidate tuning knowledge and an official system view, LLM evaluates whether the tuning knowledge conflicts with the system view and **discard any knowledge that does conflict.**

GPTuner

- Method – Knowledge Handler

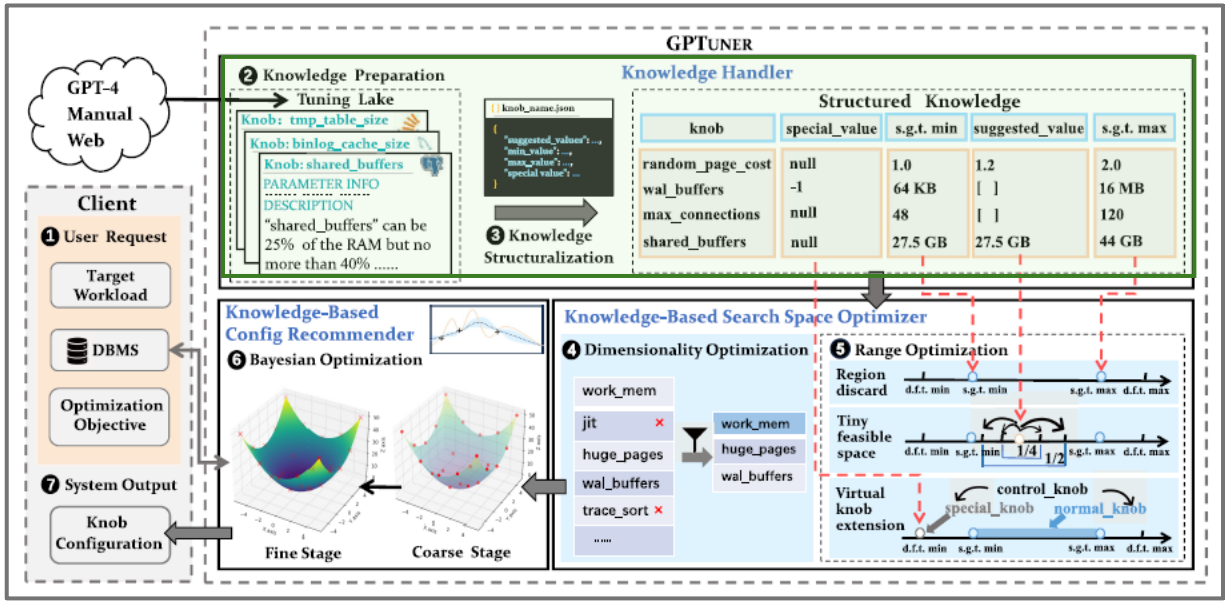


- Knowledge Preparation
  - ③ Summarizing knowledge from various resources.

    To handle conflict knowledge from different resource, **setting priority** for each information source based on its reliavbility. Then summarize the non-contradictory guidance and delete the content with low priority for the contradictory parts.

GPTuner

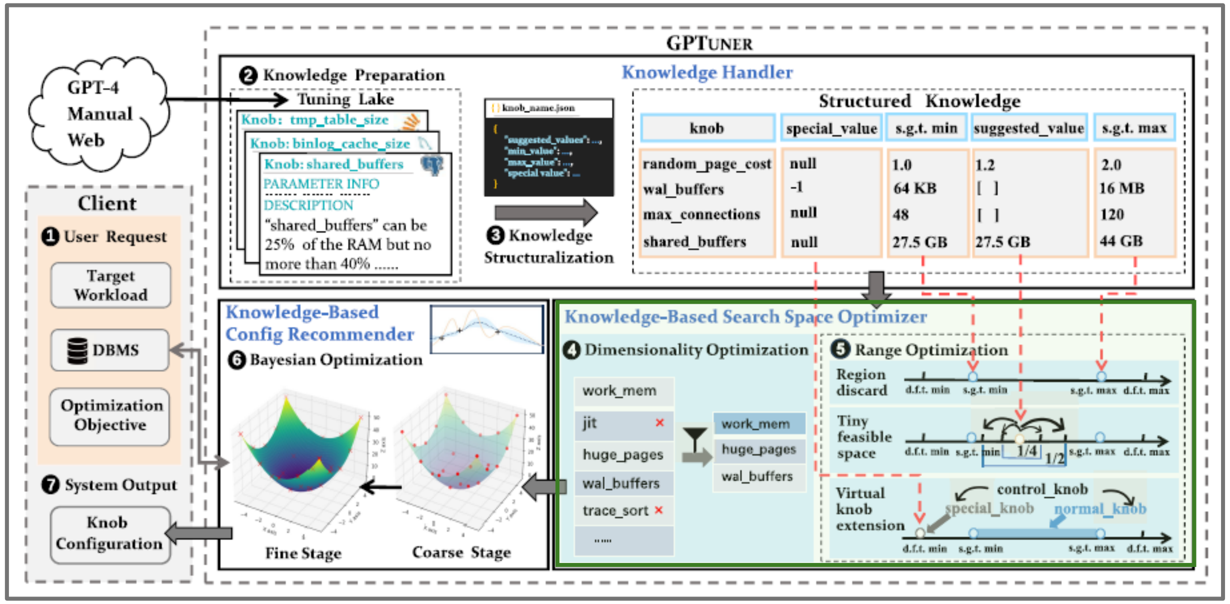
- Method – Knowledge Handler



- Knowledge Tranformation
  - ✓ Converts unstructured tuning knowledge into **structured knowledge** for machine learning models.
  - ✓ **Defines attributes** (e.g., suggested_values, min_value, max_value) for each parameter with few-shots learning.
  - ✓ Enhances tuning efficiency by narrowing search space and including special cases.

GPTuner

- Method – Knowledge-Based Search Space Optimizer



Algorithm 1: LLM-based Knob Selection

Input: Knob Set $\mathcal{K}$; LLM $\mathcal{F}$; DBMS $\mathcal{D}$; Workload $\mathcal{W}$; Tuning Lake $\mathcal{L}$.

Output: Target Knob Set $\mathcal{T}$.

Configurable Knob Set $C \leftarrow FILTER(\mathcal{K})$;

`// Filter out knobs that are related to debugging, security and path-setting`

System Level Selection:

$C_s \leftarrow \mathcal{F}(C, \mathcal{D})$;

Workload Level Selection:

$C_w \leftarrow \mathcal{F}(C, \mathcal{W})$;

Query Level Selection:

$C_q \leftarrow \emptyset$;

for *query* q_i *in* $\mathcal{W}$ **do**

  $\mathcal{E}_i \leftarrow EXECUTE(\mathcal{D}, q_i)$;

  `// Get execution plan for query` q_i `from` $\mathcal{D}$

  $C_{q_i} \leftarrow \mathcal{F}(C, \mathcal{E})$;

  $C_q \leftarrow C_q \cup C_{q_i}$;

end

Knob Level Selection:

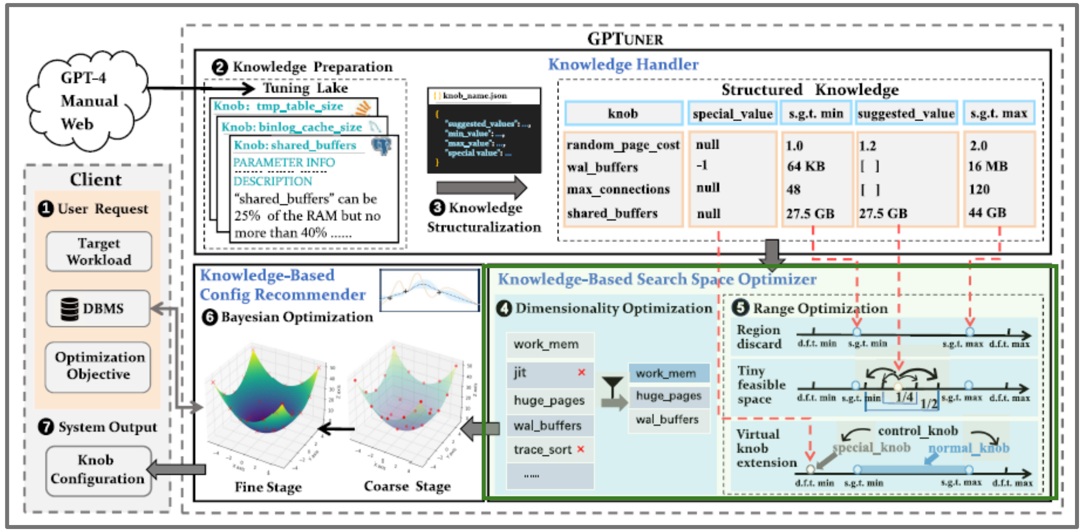
Target Knob Set $\mathcal{T} \leftarrow \mathcal{F}(\mathcal{L}, C_s \cup C_w \cup C_q)$;

return $\mathcal{T}$;

- **Dimensionality Optimization**
  - ✓ System-Level : Optimizes global DBMS settings (e.g., memory, caching policies).
  - ✓ Workload Level: Parameters based on workload type (e.g., OLTP vs. OLAP).
  - ✓ Query Level: Adjusts parameters based on query execution plans for fine-grained optimization.

GPTuner

- Method – Knowledge-Based Search Space Optimizer



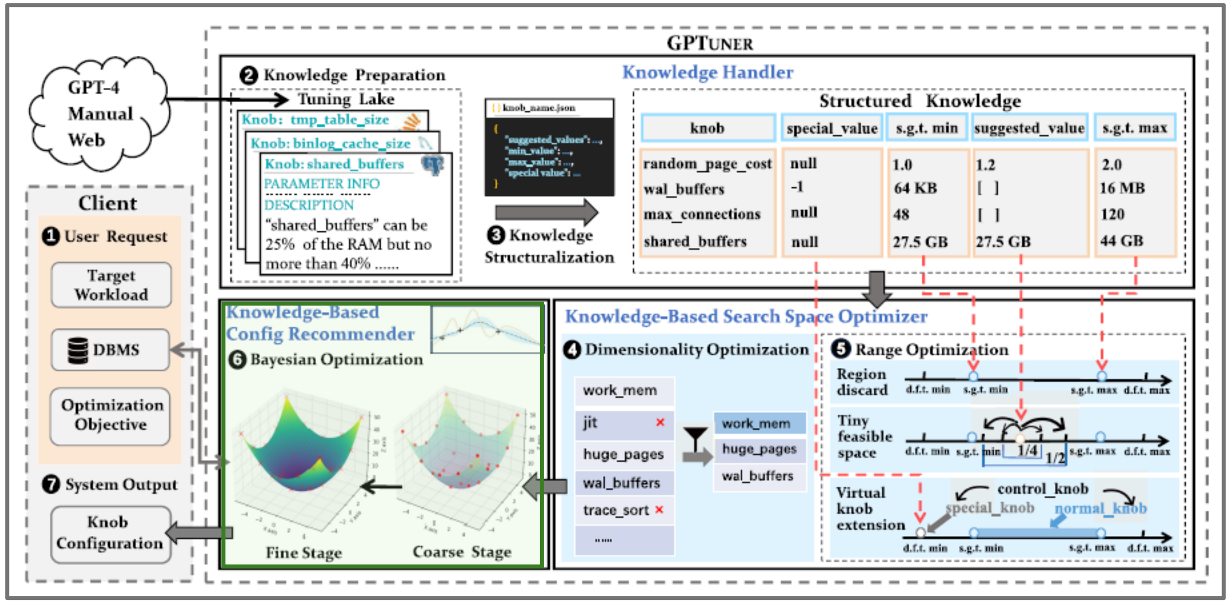
- Range Optimization
  ① Region Discard with 'structured knowledge' to **refine the value range** of each parameter to improve tuning efficiency.
  ② Tiny Feasible Space (*U: max or min value, V: optimized value, β : Scale factor*)

  $\alpha = 1 + \frac{\beta}{V}(U - V), \beta \in \{r_1, r_2, \ldots, r_n \mid r_i \in [0, 1]\}$

  ③ Virtual Knob Extension about the special parameters value.

GPTuner

- Method – Configuration Recommender



- Coarse-to-Fine Bayesian Optimization
  ① **Coarse-grained Stage** : Explore part of the whole space (*Tiny Feasible Space*) and train surrogate model. This output is non-optimal but promising results in practice, owing to the guidance of domain knowledge.
  ② **Fine-grained Stage** : Explore the space thoroughly just apply *Region Discard* and *Virtual Knob Extension*.

GPTuner

- Experiments – Performance Comparison (PostgreSQL, MySQL)

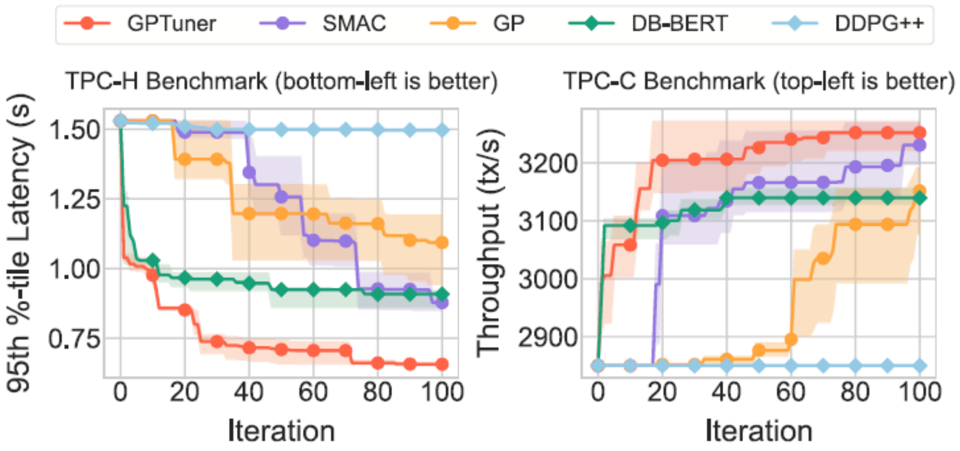


Figure 4: Best performance over iterations on PostgreSQL

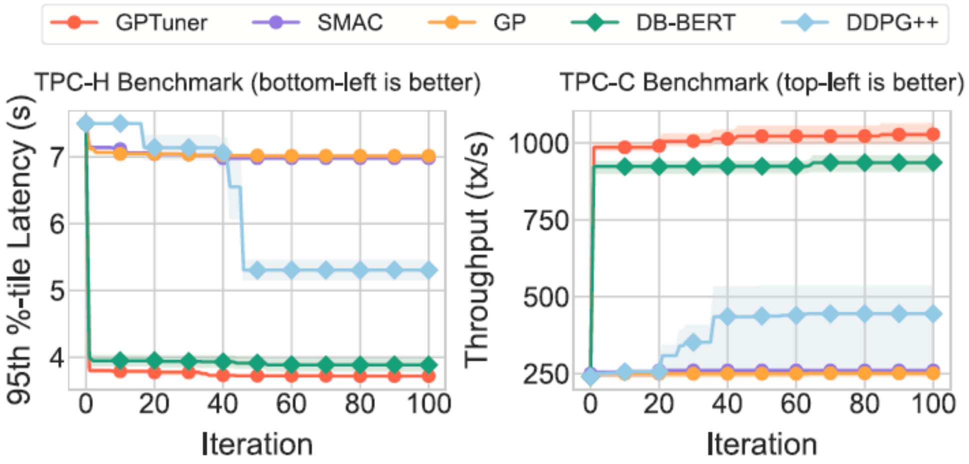


Figure 5: Best performance over iterations on MySQL

- GPTuner **rapidly achieves** significant performance improvement and reaches near optimal latency with only 20 iterations in terms of TPC-H benchmark.
- GPTuner significantly reduces the latency at the **very beginning**, surpassing the best performance achieved by all other baselines within 100 iterations.
- GP and SMAC fail to have considerable performance improvement, because the **default value ranges are excessively broad**, making the optimizers struggle to explore the vast search space.

GPTuner

- Experiments – Scalability Study

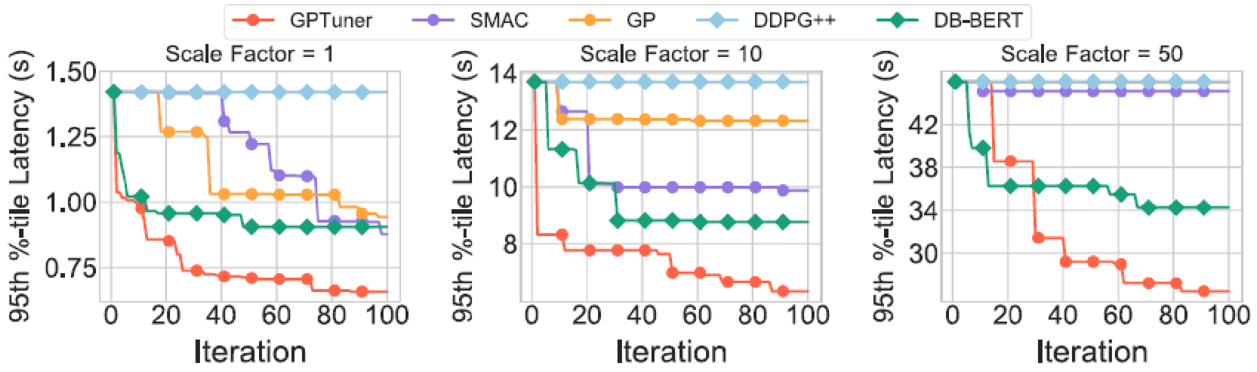


Figure 6: Effect of Database Size on Tuning Performance (bottom-left is better)

- GPTuner finds better configurations in much **fewer iterations in all sizes.**
- GPTuner learns such experience directly from **domain knowledge** rather than through iterative trial and error.

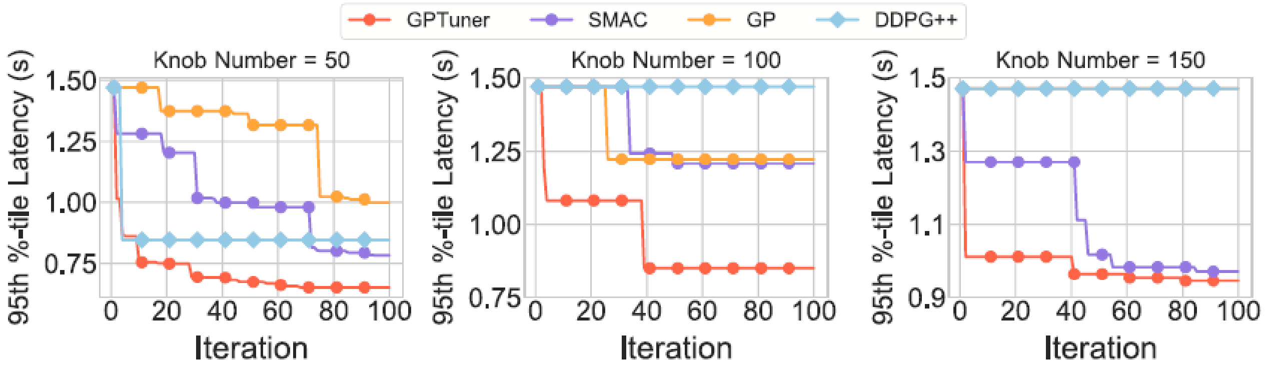


Figure 8: Effect of Space Dimensionality on Tuning Performance (bottom-left is better)

- GPTuner consistently showcases the **best performance in all space sizes.**
- Other baselines perform well in low-dimensional case, their performance deteriorated in high-dimensional cases.

YONSEI UNIVERSITY 연세대학교

Thank You for Listening