Cooling-Aware Routing in Mixture-of-Experts Models: Optimizing Expert Assignment Under Routing-Deviation Constraints

Shreyasi Saxena

Abstract

Mixture-of-Experts (MoE) models reduce computational demand by only activating a subset of available experts for each input token, but uneven expert selection often produces imbalanced workloads when experts are distributed across physical accelerators. This study investigates whether token-level routing flexibility can be used to reduce modeled direct-to-chip liquid-cooling demand while limiting departure from the model's original router preferences. Routing data were collected from the Switch-Base-8 using five separately seeded 1,000-document samples from the C4 validation dataset, including 978,748 non-padding token positions and 5,872,488 token-layer routing decisions. Expert assignments were mapped only to an 8-GPU configuration and connected to a reduced-order thermal and hydraulic model. Next, a constrained optimization was used to redistribute selected token assignments while minimizing peak GPU workload and routing deviation. Across the five samples, baseline maximum GPU workload was approximately 21% above the equal-share reference. At a target corresponding to 75% of the maximum achievable workload balancing, an average on just 3.16% of token-layer assignments were rerouted, with the mean routing deviation coming to 0.009857, while the maximum GPU workload decreased by 12.96%. The result was reproducible across separately seeded samples and stayed present under alternative expert-to-GPU placements. Under modeled hydraulic assumptions, the corresponding mean pump-power reduction ranged from 12.96% to 24.18%. These results indicate that token-level MoE routing flexibility may provide a useful mechanism for reducing modeled peak routing-dependent cooling demand, even though hardware validation and direct evaluation of model-quantity effects remain necessary.

Keywords

Mixtures-of-Experts; MoE routing; Switch Transformer; expert routing; load balancing; GPU workload; direct-to-chip liquid-cooling; thermal management; cooling-aware optimization; router deviation; expert placement; pump power.

Notation

Symbol	Meaning
t	Token-position index
ℓ	MoE-layer index
e	Expert index
a	Expert originally selected by the model
b	Batch index
g	GPU index
G	Total number of GPUs in the modeled configuration (G=8)
$s_{t\ell e}$	Raw router score for token t, layer ℓ, and expert e
$p_{t\ell e}$	Softmax router probability corresponding to $s_{t\ell e}$

$d_{t\ell,a\to e}$	Routing-deviation cost of assigning a token to expert e instead of its originally selected expert
$m_{t\ell}$	Minimum routing-deviation cost among the alternative experts for a token-layer decision
$L_{\ell e}$	Expert-token workload of expert e in layer ℓ
$L_{b\ell e}$	Expert-token workload of expert e in layer ℓ for batch b
T_ℓ	Total number of non-padding token assignments in layer ℓ
T_b	Number of non-padding token positions in batch b
$r_{\ell e}$	Relative workload share of expert e in layer ℓ
$M_{\ell eg}$	Expert-to-GPU placement indicator; equal to 1 when expert e of layer ℓ is assigned to GPU g, and 0 otherwise
W_{bg}	Expert-token workload assigned to GPU g in batch b
$\dot{Q}_{bg}$	Modeled routing-dependent rate of heat generation associated with GPU g in batch b
α	Proportionality constant relating expert-token workload to routing-dependent heat generation
ρ	Coolant density
c_p	Coolant specific heat capacity
q_{bg}	Volumetric coolant flow rate through the branch serving GPU g in batch b
ΔT	Coolant temperature rise between inlet and outlet
ΔP	Hydraulic pressure drop
K	Hydraulic resistance coefficient in the reduced-order pressure-flow model
n	Hydraulic pressure-flow exponent
q_b^{tot}	Total coolant flow supplied during batch b
η	Pump efficiency
P_b^{pump}	Modeled pump power for batch b
A	Combined constant containing the fixed thermal and hydraulic parameters of our model
$x_{bt\ell e}$	Binary optimization variable equal to 1 when token t in batch b and layer ℓ is assigned to expert e, and 0 otherwise
$D_b(\mathrm{x})$	Total routing-deviation cost of routing configuration x for batch b
$\overline{D}_b(\mathrm{x})$	Mean routing-deviation cost per token-layer decision for batch b
B	Permitted mean routing-deviation budget
W_b^{orig}	Original maximum GPU workload for batch b
$W_b^{target}(f)$	Target maximum GPU workload used in the inverse optimization
f	Fraction of the maximum achievable workload-balancing improvement targeted by the optimization

1. Introduction

The rapid growth of large-scale artificial intelligence has increased computational and thermal demands placed on data-center infrastructure. Modern language models need large numbers of accelerator operations during training and inference, thus making energy consumption and heat removal increasingly important constraints (1-2).

One approach to improving computational efficiency is the Mixture-of-Experts (MoE) architecture, in which only a subset of the model's feed-forward networks (known as experts) are activated for each input token. A learned router is able to determine which expert or experts process each token, allowing the model to increase its total parameter count without having to activate the full network for each input (3).

However, sparse activation does not necessarily imply uniform computation demand. The router may consistently favor a particular expert, causing it to receive substantially more token assignments than others. If experts are distributed across physical accelerators, this routing imbalance can lead to unequal computational workloads across devices (4). Since accelerator power consumption dissipates primarily as heat, uneven workload distribution may also produce uneven thermal loads. This imbalance is a critical challenge for direct-to-chip liquid-cooling infrastructure, where coolant and pumping requirements depend on the magnitude and distribution of heat that must be removed (1-2).

A straightforward response to an uneven thermal workload would be to increase the cooling provisions for the heavily loaded hardware. An alternative is to consider whether part of the imbalance can be reduced before the heat is generated. This can be done by modifying the routing decisions themselves. This is not the same as simply balancing the number of tokens assigned to each expert. Tokens exhibit different degrees of router confidence: while some are directed to a primary expert with a strong preference, others have competing experts with nearly equivalent router scores. Consequently, different token assignments may have very different costs of rerouting depending on the margin between expert preferences, offering an opportunity to optimize thermal distribution while limiting departure from the router’s original preferences.

This study investigates whether token-to-expert assignments in a MoE model can be selectively modified to reduce liquid-cooling operational costs while limiting the deviation from the model's original routing preferences. The router's original expert scores are used to quantify the cost of assigning a token to an alternative expert, while a reduced-order thermal and hydraulic model is used to quantify the cooling consequence of the resulting workload redistribution. By pairing these frameworks, we formulate expert reassignment as a constrained optimization problem that reduces modeled cooling demand subject to a permitted routing-deviation budget.

2. Materials and Method

2.1 Model and Dataset

This study used the google/switch-base-8 model, an open Mixture-of-Experts transformer based on the Switch Transformer architecture (3). The model contains six Mixture-of-Experts layers in its encoder, with eight experts available at each MoE layer.

The input data were taken from the English validation split of the Colossal Clean Crawled Corpus (C4), a large collection of cleaned web text commonly used for language-model training and evaluation (5). Five separately seeded samples were generated from the validation split using random seeds 42, 123, 456, 789, and 2026. Each sample contained 1,000 documents, giving a total of 5,000 documents across all runs.

Documents were tokenised using the tokenizer associated with google/switch-base-8. Sequences were truncated to a maximum length of 256 token positions. Documents were processed in batches of 8, giving 125 batches per 1,000-document run. Padding was used within each batch so that all sequences had the same length, but padding positions were excluded from the routing analysis. The model was run in evaluation mode so that its routing behaviour stayed deterministic for the fixed inputs.

Run 1, using seed 42, contained 194,528 non-padding token positions. The same collection procedure was used on the other four samples so that the consistency of routing behaviour could later be assessed across separately seeded samples of C4.

For the final routing-data collection, the model was run using Transformers version 5.15.1 with bfloat16 model precision (6). For every non-padding token position at each of the six MoE layers, the selected expert and the router scores for all eight experts were recorded for workload and rerouting analysis.

2.2 Routing Data Collection

For each non-padding token position, routing information was recorded at each of the six Mixture-of-Experts layers. At each MoE layer, the router produces eight raw scores, one corresponding to each available expert. These scores describe the router's relative preference for assigning the token to each expert.

For a token t at layer ℓ, the router scores were recorded as

$$s_{t\ell 1}, s_{t\ell 2}, \ldots, s_{t\ell 8,}$$

Where $s_{t\ell e}$ represents the router score associated with expert e. The expert ultimately selected by the model was also recorded for every token-layer combination.

In addition to the token-level data, expert workload was recorded at the batch level. For each batch, the number of token positions assigned to each of the eight experts was counted separately at all six MoE layers. This produced a workload array with the dimensions

$$125 \times 6 \times 8$$

Representing 125 batches, six MoE layers, and eight experts for each 1,000-document run.

For Run 1, the routing dataset contained 194,528 non-padding token positions. Therefore, the router-score data consisted of six sets of 194,528 token observations, with eight expert scores associated with each observation. The selected expert for each token was recorded alongside these scores. No non-padded token positions were dropped by the router during Run 1, meaning that every real token position was assigned to an expert at each MoE layer.

These measurements provide two forms of information needed for the following analysis: the selected experts determine the observed expert workload distribution, while the complete set of router scores allows the cost of alternative expert assignments to be quantified.

2.3 Quantifying Routing Deviation

To measure how strongly the router preferred its original expert assignment over an alternative, a routing-deviation cost was defined for every possible token reassignment.

For token t at MoE layer ℓ, let a denote the expert originally selected by the model and b denote a possible alternative expert. If the corresponding router scores are $s_{t\ell a}$ and $s_{t\ell b}$, the routing-deviation cost is defined as:

$$d_{t\ell,a\to b} = \max(0, s_{t\ell a} - s_{t\ell b}) \quad \text{(eq1)}$$

The maximum with zero was applied to prevent negligible negative deviation costs arising from finite-precision rounding in scores that were near to a tie. In Run 1 this affected only 349 of 1,167,168 routing decisions (0.030%).

A small value of $d_{t\ell,a\to b}$ indicates that the alternative expert received a router score close to that of the selected expert. A larger value indicates a stronger preference for the original expert and therefore a larger deviation from the model's natural routing decision.

For each token, this quantity was calculated for all seven alternative experts. This is important because the second-highest-scoring expert may not always be the best alternative for reducing thermal imbalance. A different expert may have a *slightly* larger routing-deviation cost but produce a greater reduction in the cooling imbalance.

Before the zero-clipping described above, the raw router-score difference can also be interpreted through the corresponding softmax probabilities. If $p_{t\ell a}$ and $p_{t\ell b}$ are the router probabilities of the selected and alternative experts, respectively, then

$$s_{t\ell a} - s_{t\ell b} = \ln\left(\frac{p_{t\ell a}}{p_{t\ell b}}\right) \quad \text{(eq2)}$$

This relationship shows how the unclipped router-score difference corresponds to the ratio between the selected and alternative expert probabilities. The routing-deviation cost used in the optimization remains the zero-clipped quantity defined above.

To examine how easily individual tokens could be reassigned, the minimum routing margin for each token was defined as:

$$m_{t\ell} = \min_{b\neq a} d_{t\ell,a\to b} \quad \text{(eq3)}$$

Which is equivalent to the difference between the selected expert's score and the highest-scoring alternative expert. Smaller values of $m_{t\ell}$ indicate that at least one alternative expert competed very closely with the selected expert, whereas larger values are indicative of a more decisive routing decision.

The full set of routing-deviation costs is retained for the later optimization procedure, while the minimum routing margin is used only to summarize the overall flexibility of the original routing distribution.

2.4 Baseline Expert Workload Distribution

The selected expert assignments were used to quantify the baseline workload distribution at each Mixture-of-Experts layer. For layer ℓ, the workload of expert e was defined as

$$L_{\ell e} = \sum_{t} 1(a_{t\ell} = e) \quad \text{(eq4)}$$

Where $a_{t\ell}$ is the expert selected for token t at layer ℓ, and $1(\cdot)$ is an indicator function equal to 1 when the condition is true, and 0 otherwise.

The workload distribution for layer ℓ can therefore be represented by the vector

$$L_{\ell} = (L_{\ell 1}, L_{\ell 2}, \ldots, L_{\ell 8}) \quad \text{(eq5)}$$

This vector describes how the model's token processing is distributed across the eight experts in that layer.

For comparison, an equal-share workload was defined as

$$L_{\ell}^{eq} = \frac{T_{\ell}}{8} \quad \text{(eq6)}$$

Where T_{ℓ} is the total number of non-padding token assignments at layer ℓ. This value represents the workload each expert would receive if assignments were distributed in a perfectly even manner.

The relative workload share of expert e was calculated as

$$r_{\ell e} = \frac{L_{\ell e}}{T_{\ell}} \quad \text{(eq7)}$$

These baseline workload measurements were initially used to identify experts receiving unusually high routing demand. However, a high workload alone did not make an expert a suitable target for rerouting, since its assigned tokens may be strongly preferred by the router and therefore costly to move. For this reason, expert workload was considered together with the routing deviation measures that were defined in Section 2.3.

Workload measurements were retained at both the full-run and batch levels for the later hardware analysis. The full-run totals show the overall routing pattern across each 1,000-document sample, while the batch-level values show how expert workload varies between individual groups of 8 documents. These batch-level measurements are later used to relate routing behavior to accelerator workload, heat generation, and cooling demand.

2.5 Expert-to-GPU Mapping and Hardware Workload

The expert workload measurements obtained from the MoE model cannot be treated directly as hardware thermal loads since an expert is a neural-network module instead of a physical processor. To

connect routing decisions to cooling demand, the experts must first be mapped upon a set of physical accelerator devices, consistent with expert-parallel MoE deployments In which experts are distributed across multiple accelerators (4).

Let $M_{\ell eg}$ represent the placement of expert e from MoE layer ℓ on GPU g, where

$$M_{\ell eg} = \begin{cases} 1, & \text{if expert } e \text{ of layer } \ell \text{ is alloted to GPU } g, \\ 0, & \text{otherwise.} \end{cases} \quad \text{(eq8)}$$

Each expert is assigned to one GPU, such that

$$\sum_{g} M_{\ell eg} = 1 \quad \text{(eq9)}$$

For batch b, let $L_{b\ell e}$ denote the number of token positions processed by expert e in layer ℓ. The workload assigned to GPU g during that batch can be calculated as

$$W_{bg} = \sum_{\ell} \sum_{e} M_{\ell eg} L_{b\ell e} \quad \text{(eq10)}$$

The quantity W_{bg} therefore represents the number of expert-token computations assigned to GPU g during batch b. This gives us the link between the model's routing behaviour and the physical hardware workload used in the thermal model.

Controlled expert-to-GPU placement is required because the Switch-Base-8 model does not specify a unique physical deployment configuration. For the primary hardware configuration, an eight-GPU expert-parallel arrangement was used. Experts were aligned by index across layers: GPU 1 hosted Expert 1 from each of the six MoE layers, GPU 2 hosted Expert 2 from each layer, and so on through GPU 8. Each GPU therefore contained six experts in total, one from each MoE layer. The configuration ensured that every GPU contained the same number of experts while allowing differences in routing behaviour to produce differences in hardware workload.

Under this index-aligned placement, the general workload expression simplifies to

$$W_{bg} = \sum_{\ell=1}^{6} L_{b\ell g} \quad \text{(eq11)}$$

Thus, the workload of GPU g during batch b is the total number of token computations assigned to Expert g across the six MoE layers. The same placement is retained before and after rerouting so that changes in modeled cooling demand come from changes in token assignment instead of changes in hardware configuration.

To evaluate whether the results depended strongly on this particular expert placement, five additional balanced expert-to-GPU configurations were generated for a placement sensitivity analysis. Within each MoE layer, the eight experts were randomly permuted across the eight GPUs, while preserving one expert from each layer on every GPU, and therefore six experts per GPU in total. The mappings were

generated using a fixed random seed of 20260830. The cooling-aware optimization was then repeated on Run 1 at the 75% balancing target for each alternative placement.

2.6 Thermal and Liquid-Cooling Model

The hardware workload defined in Section 2.5 was next connected to the thermal demand placed on the cooling system. The purpose of this model is not to predict the exact temperature or power consumption of a specific physical GPU, but to compare how different MoE routing configurations alter the distribution of heat-producing computational workload across devices.

Since each expert in Switch-Base-8 has the same underlying architecture, each expert-token assignment was treated as an equal unit of expert computation. The routing-dependent thermal load of GPU g during batch b was therefore modeled as proportional to its expert-token workload:

$$\dot{Q}_{bg} = \alpha W_{bg} \quad (\text{eq}12)$$

where $\dot{Q}_{bg}$ represents the rate of heat generation associated with GPU g, W_{bg} is the number of expert-token computations assigned to that GPU, and α is a proportionality constant relating computational workload to heat generation.

For a direct-to-chip liquid-cooling system, heat is transferred from high-power components through cold plates into a circulating coolant loop (2, 7). The rate of heat absorbed by the coolant was represented using the standard energy-balance relationship

$$\dot{Q}_{bg} = \rho c_p q_{bg} \Delta T \quad (\text{eq}13)$$

Where ρ is the coolant density, c_p is its specific heat capacity, q_{bg} is the volumetric coolant flow rate through the GPU cooling branch, and ΔT is the coolant temperature rise between the inlet and outlet.

This proportional relationship is a reduced-order modeling assumption rather than a direct heat measurement; it is based on empirical evidence that GPU load is strongly associated with server power consumption and GPU temperature (1)

Rearranging gives the coolant flow required to move a given thermal load:

$$q_{bg} = \frac{\dot{Q}_{bg}}{\rho c_p \Delta T} \quad (\text{eq}14)$$

Substituting the workload-based thermal model gives

$$q_{bg} = \frac{\alpha W_{bg}}{\rho c_p \Delta T} \quad (\text{eq}15)$$

Therefore, when coolant properties and the permitted coolant temperature rise are held constant, the coolant flow required by a GPU is proportional to its computational workload:

$$q_{bg} \propto W_{bg} \quad (\text{eq}16)$$

The hydraulic pressure required to drive coolant through a cold-plate branch increases with coolant flow rate. Experimental direct-to-chip cooling systems exhibit increasing pressure drop with increasing flow, while the precise pressure-flow relationship depends on the hydraulic characteristics and flow regime of the system (7-8). This relationship was represented using the reduced-order model

$$\Delta P = Kq^n \tag{eq17}$$

Where K represents the hydraulic resistance of the cooling path and $n > 0$ determines the degree of nonlinearity between flow rate and pressure drop. The exact value of n depends on the hydraulic characteristics of the cooling system. Rather than assuming that one value is universally applicable, the sensitivity analysis evaluated $n = 1$, $n = 1.5$, and $n = 2$.

The model cooling system consists of 8 parallel direct-to-chip cooling branches supplied by a shared variable-speed pump, with one branch corresponding to each GPU. CDU-based direct-to-chip systems commonly distribute coolant from a pumping unit through multiple server-level cooling paths (2, 7). However, the 8 identical branches used here make up a controlled reduced-order configuration instead of modeling a specific deployed cooling system. Since different GPUs may require different coolant flow rates, the pump must provide sufficient pressure to satisfy the branch with the greatest flow requirement. For batch b, the required pump pressure was therefore modeled as

$$\Delta P_b = K\left(\max_g q_{bg}\right)^n \tag{eq18}$$

This is an approximation that assumes identical branch resistance K and neglects additional manifold and header pressure losses.

The total coolant flow supplied by the shared pump is

$$q_b^{tot} = \sum_{g=1}^{G} q_{bg} \tag{eq19}$$

where $G = 8$ for the primary hardware configuration.

The hydraulic power required by the pump is

$$P_b^{pump} = \frac{\Delta P_b q_b^{tot}}{\eta} \tag{eq20}$$

Where η represents the efficiency of the pump. Pumping requirements depend on both the coolant flow rate and the pressure rise supplied by the pump (8).

Substituting the pressure-drop model gives

$$P_b^{pump} = \frac{K}{\eta}\left(\max_g q_{bg}\right)^n \sum_{g=1}^{G} q_{bg} \tag{eq21}$$

Since coolant flow is proportional to GPU workload, the expression can be rewritten in terms of the expert-token workload:

$$P_b^{pump} = A\left(\max_g W_{bg}\right)^n \sum_{g=1}^{G} W_{bg} \quad \text{(eq22)}$$

Where

$$A = \frac{K}{\eta}\left(\frac{\alpha}{\rho c_p \Delta T}\right)^{n+1} \quad \text{(eq23)}$$

contains the physical constants of the thermal and hydraulic system.

Token rerouting changes the distribution of expert-token computations between GPUs but does not change the total number of expert-token computations performed within the same batch. Therefore,

$$\sum_{g=1}^{G} W_{bg} = \text{constant for a fixed batch} \quad \text{(eq24)}$$

remains constant when comparing the original and rerouted configurations of a given batch.

Consequently, for $n > 0$, minimizing the modeled pump power is equivalent to minimizing the maximum GPU workload:

$$\boxed{\min P_b^{pump} \Leftrightarrow \min \max_g W_{bg}} \quad \text{(eq25)}$$

This provides the cooling objective used in the routing optimization. Instead of attempting to reduce the total amount of expert computation, which remains unchanged, the model attempts to redistribute that computation so that the most heavily loaded GPU requires less cooling.

The effect of a rerouting configuration can also be expressed without assigning arbitrary absolute values to the constants contained in A. For the same batch, the ratio between the modeling pumping requirement after rerouting and the original pumping requirement is

$$\boxed{\frac{P_{b,new}^{pump}}{P_{b,original}^{pump}} = \left(\frac{max_g\, W_{bg}^{new}}{max_g\, W_{bg}^{original}}\right)^n} \quad \text{(eq26)}$$

This relative formulation allows the thermal effect of alternative routing configurations to be compared without claiming an absolute pump-power measurement for a particular data center. Multiple values of the hydraulic exponent n are later evaluated to determine whether the estimated cooling benefit remains consistent under different pressure-flow relationships.

2.7 Cooling-aware Routing Optimization

The final stage of the methodology combines the routing-deviation measure introduced in Section 2.3 with the cooling objective derived in Section 2.6. The purpose of our optimization is to determine

whether selected token assignments can be redirected to alternative experts in a way that reduces peak GPU workload while limiting departure from the model's original routing preferences.

For token t in MoE layer ℓ, define the binary decision variable

$$x_{t\ell e} = \begin{cases} 1, & \text{if token } t \text{ is assigned to expert } e, \\ 0, & \text{otherwise.} \end{cases} \quad \text{(eq27)}$$

Represents the routing decision made by the optimizer.

Since Switch-Base-8 uses top-1 routing, each token must be assigned to exactly one expert in every MoE layer (3):

$$\sum_{e=1}^{8} x_{t\ell e} = 1 \quad \text{(eq28)}$$

The resulting workload of expert e in layer ℓ is

$$L_{\ell e}(\mathrm{x}) = \sum_{t} x_{t\ell e} \quad \text{(eq29)}$$

Using the expert-to-GPU placement matrix $M_{\ell eg}$ defined in Section 2.5, the resulting workload of GPU g is

$$W_g(\mathrm{x}) = \sum_{\ell=1}^{6} \sum_{e=1}^{8} M_{\ell eg} L_{\ell e}(\mathrm{x}) \quad \text{(eq30)}$$

For the primary index-alignment placement used in this study, GPU g hosts expert g from each of the six MoE layers. The expression therefore simplifies to

$$W_g(\mathrm{x}) = \sum_{\ell=1}^{6} L_{\ell g}(\mathrm{x}) \quad \text{(eq31)}$$

The routing-deviation cost associated with assigning a token to expert e is

$$d_{t\ell e} = \max(0, s_{t\ell a} - s_{t\ell e}) \quad \text{(eq32)}$$

where a is the expert originally selected by the model. Retaining the original assignment gives a deviation cost of zero, while rerouting to an alternative expert incurs a cost based on the difference between the corresponding router scores.

The total routing deviation of a candidate configuration is

$$D_b(\mathrm{x}) = \sum_{\ell=1}^{6} \sum_{t=1}^{T_b} \sum_{e=1}^{8} d_{bt\ell e} x_{bt\ell e} \quad \text{(eq33)}$$

Because batches contain different numbers of non-padding token positions, this quantity is normalized by the number of token-layer routing decisions. For batch b,

$$\overline{D}_b(\mathrm{x}) = \frac{D_b(\mathrm{x})}{6T_b} \tag{eq34}$$

where T_b is the number of non-padding token positions in that batch.

The general cooling-aware routing problem can therefore be expressed as

$$\boxed{\min_{\mathrm{x}} \max_{g} W_{bg}(\mathrm{x})} \tag{eq35}$$

subject to

$$\boxed{\overline{D}_b(\mathrm{x}) \leq B} \tag{eq36}$$

and

$$\sum_{e=1}^{8} x_{bt\ell e} = 1, \qquad x_{bt\ell e} \in \{0, 1\} \tag{eq37}$$

Here, B represents the permitted mean routing-deviation budget. A small B restricts the optimizer to routing changes that remain close to the model's original expert preferences, while a larger B permits progressively greater deviation.

To sample the complete routing-cooling trade-off systematically, the numerical experiments were performed using an equivalent inverse formulation. Rather than selecting arbitrary values of B in advance, a sequence of target peak GPU workloads was specified, and the optimizer determined the minimum routing deviation required to achieve each target.

For each batch, let W_b^{orig} denote the original maximum GPU workload and W_b^{best} denote the lowest achievable maximum workload. Target workloads were defined as

$$W_b^{target}(f) = W_b^{orig} - f\left(W_b^{orig} - W_b^{best}\right) \tag{eq38}$$

where

$$f \in \{0, 0.125, 0.25, 0.375, 0.50, 0.625, 0.75, 0.875, 1\}.$$

Thus, $f = 0$ represents the original routing configuration, while $f = 1$ represents the best achievable workload balance.

For each target, the inverse optimization was

$$\boxed{\min_{\mathrm{x}} \overline{D}_b(\mathrm{x})} \tag{eq39}$$

subject to

$$\boxed{W_{bg}(\mathrm{x}) \leq W_b^{target} \text{ for ever GPU } g} \quad (\mathrm{eq40})$$

together with the top-1 routing constraints defined in (eq37).

The resulting pairs of minimum routing deviation and peak GPU workload form the same routing-cooling trade-off frontier as the original budget-constrained formulation. The inverse form was used only to sample this frontier systematically; it does not change the underlying optimization objective.

Finally, reductions in peak GPU workload were converted into relative modeled pumping demand using the relationship derived in Section 2.6:

Run 1 was used to characterize the full routing-cooling trade-off frontier. To test reproducibility, the 75% balancing target was subsequently evaluated using the same optimization procedure on four additional separately seeded C4 samples.

2.7 Statistical Treatment

The optimization was deterministic for a fixed set of routing traces, so descriptive statistics were used to summarize variation in workload imbalance and optimization performance rather than inferential hypothesis testing. For Run 1, batch-level workload imbalance across the 125 batches was described using the mean, median, standard deviation, and range. Reproducibility at the 75% balancing target was assessed across five separately seeded runs, comprising 625 batches in total. Between-run variations were summarized using the range and standard deviation of the run-level means. Placement and hydraulic sensitivity were examined separately using the alternative configurations in Sections 2.5 and 2.6.

3. Results

3.1 Baseline Routing and GPU Workload

The original routing configuration was first evaluated without any token reassignment. For each of the 125 batches in Run 1, the expert-token workloads from the six MoE layers were converted into GPU workloads using the index-aligned expert placement defined in Section 2.5. Across the complete run, 194,528 non-padding token positions produced 1,167,168 expert-token computations across the six MoE layers.

Under the controlled eight-GPU placement, the aggregate workloads were

$$(141367, 111890, 139188, 159986,$$

$$152600, 145261, 174569, 142307)$$

for GPUs 1-8, respectively. An equal distribution of the total workload would correspond to 145,896 expert-token computations per GPU. GPU 7 received the greatest aggregate workload, with 174,569 expert-token computations, equivalent to 14.96% of the total and 19.65% above the equal-share workload. GPU 2 received the lowest workload, with 111,890 expert-token computations, equivalent to 9.59% of the total and 23.31% below the equal share.

However, the aggregate workload alone does not represent the instantaneous workload experienced by the modeled cooling system, since the 1,000 documents were processed over 125 separate batches. The workload imbalance was therefore also evaluated independently for each batch. GPU 7 was the maximum-workload device in 100 of the 125 batches (80.0%), while GPU 4 was the maximum in 24 batches (19.2%) and GPU 5 in one batch (0.8%). None of the remaining GPUs was the maximum-workload device in any batch.

For each batch, the maximum GPU workload was compared with the workload that would result from an equal distribution of that batch's expert-token computations across all eight GPUs. The maximum GPU workload was, on average, 20.76% above this equal-share reference, with a standard deviation of 4.31 percentage points and a median excess of 20.55%. Across individual batches, the excess ranged from 10.72% to 31.04%.

These results indicate that the original routing decisions produce a persistent workload imbalance under the controlled expert-to-GPU mapping. In particular, the concentration on GPU 7 occurred across the majority of batches rather than arising from a small number of unusually imbalanced inputs. This unoptimized workload distribution therefore provides the baseline against which the cooling-aware routing optimization is evaluated.

Table 1. Aggregate GPU workload under the original routing configuration for Run 1.

GPU	Expert-token workload	Share of total workload	Difference from equal share
GPU 1	141,367	12.11%	−3.10%
GPU 2	111,890	9.59%	−23.31%
GPU 3	139,188	11.93%	−4.60%
GPU 4	159,986	13.71%	+9.66%
GPU 5	152,600	13.07%	+4.59%
GPU 6	145,261	12.45%	−0.44%
GPU 7	174,569	14.96%	+19.65%
GPU 8	142,307	12.19%	−2.46%

3.2 Routing-Deviation Characteristics

The workload imbalance observed in Section 3.1 does not by itself indicate whether the corresponding assignments can be easily redistributed. The router may only weakly prefer the selected expert for some tokens, while showing a substantially stronger preference for others. The routing-deviation costs were thus examined before applying the cooling-aware optimization.

For each original token-layer assignment, the minimum routing-deviation cost among the seven alternative experts was calculated. Across all 1,167,168 routing decisions in Run 1, the median minimum deviation was 2.545. However, the magnitude of this deviation varied considerably between experts and layers.

The difference was particularly apparent when the highest-workload expert from each MoE layer was examined. Their routing loads and minimum-deviation characteristics are shown in Table 2.

Table 2. Routing-deviation characteristics of the highest-workload expert in each MoE layer.

Layer	Highest-workload expert	Assignments	Share of layer workload	Median minimum deviation	Alternative within $d \leq \ln 2$	Alternative within $d \leq \ln 5$
1	E8	27,916	14.35%	1.187	21.09%	73.65%
2	E7	27,428	14.10%	3.186	11.87%	22.74%
3	E5	31,379	16.13%	2.196	18.30%	37.17%
4	E6	34,197	17.58%	3.049	12.17%	25.71%
5	E7	58,176	29.91%	15.925	2.42%	5.76%
6	E5	38,757	19.92%	8.196	3.98%	8.96%

The largest contrast occurred in Layer 5. Expert 7 received 58,176 assignments, corresponding to 29.91% of all routing decisions in that layer and approximately 2.39 times the equal-share workload. However, these assignments were also associated with particularly large routing-deviation costs. The median cost of rerouting a Layer 5 Expert 7 assignment to its closest alternative was 15.925, compared with 2.545 across all routing decisions in Run 1.

Only 2.42% of Layer 5 Expert 7 assignments had an alternative within $d \leq \ln 2$, and only 5.76% had an alternative within $d \leq \ln 5$. In contrast, although Expert 8 was the highest-workload expert in Layer 1, its median minimum deviation was only 1.187, and 73.65% of its assignments had an alternative within $d \leq \ln 5$.

These results show that expert workload and routing flexibility are not equivalent. In particular, the most heavily used expert in the model was also among the most costly to reroute according to the router-deviation measure. Consequently, a strategy that attempts to balance hardware workload solely by rerouting assignments from the busiest experts may require substantially greater deviation from the model's original routing preferences than a strategy that considers the costs of alternative routing decisions across all layers.

3.3 Cooling-Aware Routing Optimization Results

The cooling-aware routing optimization was next applied independently to each of the 125 batches in Run 1. For each batch, the optimizer identified the minimum routing deviation required to achieve progressively greater reductions in maximum GPU workload, ranging from the original routing configuration to the best achievable workload balance.

The resulting trade-off is shown in Table 3. Small departures from the model's original routing preferences produced substantial reductions in peak GPU workload. Achieving 50% of the maximum possible workload balancing required a mean routing-deviation value of only 0.003642 and changed approximately 1.62% of token-layer assignments. This produced an average reduction of 8.51% in maximum GPU workload.

At 75% of the maximum achievable balancing improvement, the mean routing-deviation requirement remained below 0.01, at 0.009735. Only 3.20% of token-layer assignments were rerouted, while the mean maximum GPU workload was reduced by 12.77%.

The relationship became progressively less favorable as the workload approached perfect balance. Achieving the final 25% of the available balancing improvement increased the mean routing-deviation requirement from 0.009735 at the 75% level to 0.038202 at the fully balanced endpoint. At this endpoint, approximately 7.02% of routing decisions were modified and the mean reduction in maximum GPU workload was 17.06%.

These results indicate diminishing returns in the routing-cooling trade-off. A large proportion of the achievable reduction in peak workload could be obtained using relatively small deviations from the model's original routing preferences, whereas approaching the theoretical workload balance limit required increasingly larger routing deviations.

Table 3. Routing-deviation cost and workload reduction across the Run 1 optimization frontier.

Fraction of maximum achievable balancing	**Mean routing deviation $\overline{D}$**	**Token-layer assignments rerouted**	**Mean reduction in maximum GPU workload**
0%	0.000000	0.00%	0.00%
12.5%	0.000221	0.34%	2.10%
25%	0.000875	0.72%	4.24%
37.5%	0.001976	1.13%	6.37%
50%	0.003642	1.62%	8.51%
62.5%	0.006045	2.26%	10.63%
75%	0.009735	3.20%	12.77%
87.5%	0.016783	4.73%	14.90%
100%	0.038202	7.02%	17.06%

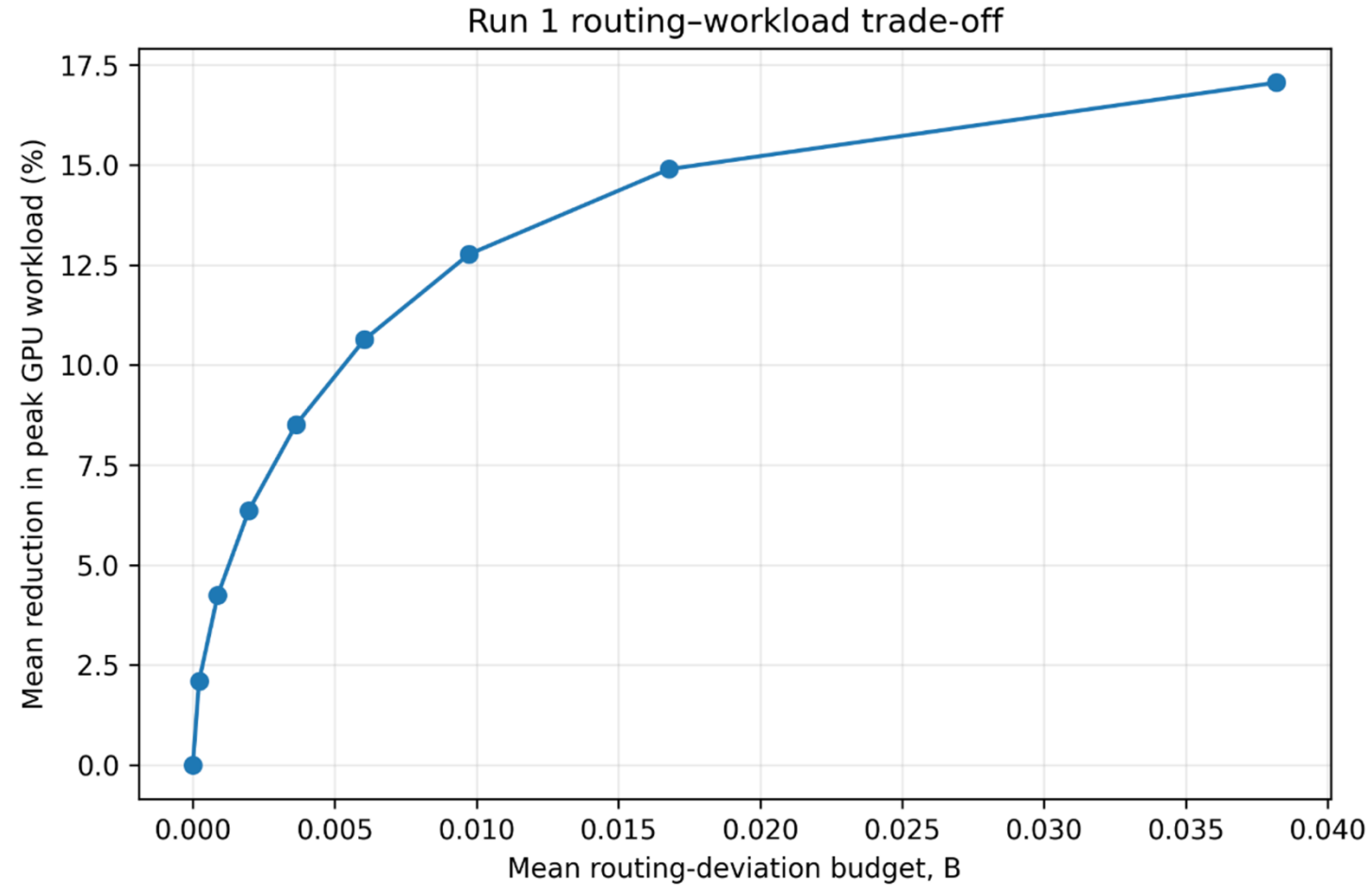

Figure 1. Mean reduction in maximum GPU workload as a function of the minimum mean routing deviation required across the 125 batches in Run 1. The flattening of the curve indicates diminishing returns as the workload approaches the best achievable balance

3.4 Modeled Cooling Reduction

The reductions in maximum GPU workload obtained through cooling-aware rerouting were then translated into relative thermal and cooling requirements using the model developed in Section 2.6. Since the routing-dependent thermal load was modeled as proportional to expert-token workload in (eq12), a given percentage reduction in maximum GPU workload corresponds to the same percentage reduction in modeled peak routing-dependent thermal load.

Similarly, (eq15) shows that coolant flow is proportional to GPU workload when coolant properties and the permitted coolant temperature rise are held constant. Therefore, the same percentage reduction applies to the modeled peak branch-flow requirement.

At the 50% balancing level identified in Section 3.3, the maximum GPU workload was reduced by an average of 8.51%. This thus corresponds to an 8.51% reduction in both modeled peak routing-dependent thermal load and peak coolant-flow requirement. At the 75% balancing level, where only 3.20% of token-layer assignments were rerouted and the mean routing deviation remained below 0.01,

the corresponding reduction was 12.77%. At the best achievable workload balance, the mean reduction reached 17.06%.

The effect on modeled pump power additionally depends on the nonlinear relationship between coolant flow and hydraulic pressure. The relative modeled pump-power requirement was calculated using (eq26). Therefore, the same reduction in peak workload can produce different predicted reductions in pumping demand depending on the hydraulic exponent n. Rather than selecting a single value of n, this dependence is evaluated across multiple hydraulic assumptions in the sensitivity analysis in Section 3.6.

These results show that the routing improvements obtained in Section 3.3 translate directly into reductions in the peak routing-dependent thermal and coolant-flow requirements of the modeled system. The analysis represents relative cooling demand rather than absolute GPU temperature, heat generation, or pump power, since the experiment did not include physical hardware measurements or calibration of the proportionality constant α.

Table 4. Modeled thermal and coolant-flow effects at selected points on the Run 1 optimization frontier.

Balancing level	Assignments rerouted	Peak workload reduction	Modeled peak thermal-load reduction	Modeled peak coolant-flow reduction
0%	0.00%	0.00%	0.00%	0.00%
50%	1.62%	8.51%	8.51%	8.51%
75%	3.20%	12.77%	12.77%	12.77%
100%	7.02%	17.06%	17.06%	17.06%

3.5 Replication Across Separately Seeded C4 Samples

To determine whether the workload imbalance and optimization performance observed in Run 1 were specific to a particular input sample, the analysis was repeated using four additional 1,000-document C4 samples generated with different random seeds. Across the five runs, 978,748 non-padding token positions produced a total of 5,872,488 token-layer routing decisions.

The baseline workload imbalance remained highly consistent between samples. The mean excess of the maximum GPU workload above the equal-share reference ranged from 20.62% to 21.67% across the five runs. GPU 7 remained the maximum-workload device in between 80.0% and 93.6% of batches in every run. Across all 625 batches, GPU 7 was the maximum-workload device in 545 batches (87.2%).

The routing behavior responsible for this imbalance was also stable across samples. Layer 5 Expert 7 received between 29.75% and 30.39% of all Layer 5 assignments in each run. Despite this consistently high workload, its median minimum routing-deviation cost remained between 15.84 and 16.19, compared with an overall median minimum deviation between 2.545 and 2.558 across all routing decisions in the individual runs.

The cooling-aware optimization was then evaluated at the 75% balancing target identified from the Run 1 trade-off frontier. The results were highly consistent across all five samples. The mean routing-deviation requirement ranged only from 0.009695 to 0.010062, while the proportion of token-layer

assignments rerouted ranged from 3.13% to 3.20%. The corresponding mean reduction in maximum GPU workload ranged from 12.67% to 13.22%.

Across the five run-level means, achieving the 75% balancing target required a mean routing deviation of 0.009857 and rerouting of 3.16% of token-layer assignments. This resulted in a mean reduction of 12.96% in maximum GPU workload. The standard deviation between run-level mean workload reductions was only 0.24 percentage points, meaning that the optimization benefit was not strongly dependent on the particular C4 sample used.

These results demonstrate that both the baseline workload imbalance and the low-deviation region of the routing-cooling trade-off were reproducible across separately seeded input samples. The similar rerouting costs, rerouting fractions, and peak-workload reductions suggest that the Run 1 result was not produced by an unusually favorable set of inputs.

Table 5. Replication of baseline imbalance and the 75% balancing target across five C4 samples.

Run	Non-padding tokens	Mean baseline peak excess	GPU 7 busiest batches	Mean routing deviation at 75% target	Assignments rerouted	Mean peak-workload reduction
Run 1	194,528	20.76%	80.0%	0.009735	3.20%	12.77%
Run 2	194,124	21.67%	93.6%	0.009997	3.14%	13.22%
Run 3	197,288	21.53%	88.8%	0.009796	3.13%	13.17%
Run 4	197,437	20.62%	87.2%	0.009695	3.17%	12.67%
Run 5	195,371	21.18%	86.4%	0.010062	3.17%	12.96%
Mean	-	**21.15%**	**87.2%**	**0.009857**	**3.16%**	**12.96%**

3.6 Sensitivity Analysis

The sensitivity of the results was evaluated with respect to two assumptions of the modeled system: the hydraulic pressure-flow exponent n and the expert-to-GPU placement. These analyses were performed to determine whether the estimated cooling benefit was dependent on a single hydraulic relationship or on the index-aligned hardware configuration used in the primary analysis.

3.6.1 *Sensitivity to the Hydraulic Exponent*

The routing optimization itself is independent of the hydraulic exponent n, since it minimized peak GPU workload before the resulting workload reduction is converted into modeled pump demand. However, the magnitude of the predicted pump-power reduction depends on n through (eq26).

At the 75% balancing target, the mean reduction in maximum GPU workload across all 625 batches from the five separately seeded runs was 12.96%. For $n = 1$, this corresponded to a mean modeled pump-power reduction of 12.96%. Increasing the exponent to $n = 1.5$ increased the predicted reduction to 18.77%, while $n = 2$ produced a mean predicted reduction of 24.18%.

The direction of the cooling benefit therefore remained unchanged across all three hydraulic assumptions, although its estimated magnitude increased with the nonlinearity of the pressure-flow relationship.

Table 6. Sensitivity of modeled pump-power reduction to the hydraulic exponent at the 75% balancing target.

Hydraulic exponent n	**Mean modeled pump-power reduction**	**Range of run-level means**
1.0	12.96%	12.67–13.22%
1.5	18.77%	18.36–19.14%
2.0	24.18%	23.66–24.64%

3.6.2 *Sensitivity to Expert-To-GPU Placement*

The primary index-aligned mapping intentionally provides a controlled hardware configuration, but the absolute workload imbalance produced by an MoE model can depend on which experts are collocated per GPU. The Run 1 analysis was therefore repeated using 5 additional balanced expert-to-GPU placements.

Changing the placement substantially altered the original degree of hardware imbalance. Across the 5 alternative configurations, the mean excess of the maximum GPU workload above the equal-share reference ranged from 14.94% to 27.98%, compared with 20.76% under the primary index-aligned configuration.

Despite these differences, the cooling-aware optimizer retained a low-deviation balancing region under every tested placement. At the 75% balancing target, the minimum mean routing deviation ranged from 0.006439 to 0.020723 across the 5 alternative mappings. The proportion of token-layer assignments that required rerouting ranged from 2.79% to 4.78%.

The resulting mean reduction in maximum GPU workload ranged from 9.63% to 16.19%. This variation primarily reflected differences in the amount of baseline imbalance available to remove: placements that were already more balanced provided less absolute potential for peak-workload reduction, whereas more imbalanced placements provided a larger potential reduction.

Table 7. Sensitivity of the 75% balancing result to expert-to-GPU placement in Run 1.

Expert placement	**Mean baseline peak excess**	**Mean routing deviation**	**Assignments rerouted**	**Mean peak-workload reduction**
Primary index-aligned	20.76%	0.009735	3.20%	12.77%
Random balanced 1	14.94%	0.006439	2.79%	9.63%

Random balanced 2	26.36%	0.020723	4.78%	15.52%
Random balanced 3	24.93%	0.017781	4.24%	14.85%
Random balanced 4	19.48%	0.008214	2.96%	12.06%
Random balanced 5	27.98%	0.017385	3.63%	16.19%

Across all 5 alternative placements, 75% of the available workload-balancing improvement was therefore achieved while rerouting fewer than 4.8% of the token-layer assignments and maintaining a mean routing deviation below 0.021. The identity and magnitude of the hardware bottleneck changed with expert placement, but the optimization procedure itself required no modification, since the objective minimizes the maximum workload across all GPUs rather than targeting a specific device.

4. Discussions

4.1 Main Findings

The results indicate that the original MoE routing decisions produced a persistent imbalance in hardware workload under the controlled expert-to-GPU mapping. Across the 5 separately seeded C4 samples, the maximum GPU workload was on average approximately 21% above the equal share reference. This imbalance was not limited to isolated batches, as GPU 7 was the maximum-workload device in 87.2% of the 625 batches evaluated across all 5 runs.

However, the optimization results suggest that a large portion of this imbalance can be reduced without extensively altering the model's original routing decisions. At the 75% balancing target, only 3.16% of token-layer assignments were rerouted on average across the 5 samples, with a mean routing-deviation value of 0.009857. These changes reduced the maximum GPU workload by an average of 12.96%.

The trade-off frontier obtained from Run 1 further showed that the relationship between routing deviation and workload reduction was nonlinear. Early reductions in peak workload required relatively small increases in routing deviations, whereas approaching the best achievable workload balance became progressively more expensive. For example, 75% of the available balancing improvement was reached with a mean routing deviation below 0.01, while achieving the fully balanced endpoint required a mean value of 0.038202.

This behaviour suggests the existence of a low-deviation region in which relatively small modifications to token routing can produce disproportionately large changes in the distribution of hardware workload. Beyond this region, additional balancing requires increasingly strong departures from the router's original preferences.

Routing-induced workload imbalance has previously been recognized as an important systems challenge in distributed MoE models. Existing approaches have primarily addressed this imbalance to

improve computational efficiency, throughput, communication, or inference latency, for example through adaptive parallelism and expert-placement optimization (9-10). The present study considers a different physical consequence of the same routing imbalance by connecting expert-token workload distribution to modeled direct-to-chip cooling demand.

4.2 Interaction Between Expert Workload and Router Preference

A key finding of the analysis was that expert workload and router flexibility were not equivalent. The most heavily used expert was not necessarily the most suitable source of rerouted assignments.

This was most clearly demonstrated by Layer 5 Expert 7. Across the 5 samples, this expert consistently received approximately 30% of all Layer 5 routing decisions, substantially exceeding the equal-share value of 12.5%. Under the primary hardware mapping, this concentration contributed strongly to the high workload observed on GPU 7.

Despite its high workload, Layer 5 Expert 7 also exhibited unusually strong router preferences. In Run 1, the median minimum routing-deviation cost for these assignments was 15.925, compared with 2.545 across all routing decisions. Only 2.42% of its assignments had an alternative expert within a deviation of $\ln 2$.

Consequently, directly redistributing large numbers of assignments from the most heavily used expert would have incurred a comparatively large routing-deviation cost. The optimization instead exploited lower-cost opportunities across other experts and layers that contributed to the same physical GPU workload.

At the 75% balancing target in Run 1, the optimizer modified only a small fraction of Layer 5 Expert 7 assignments despite GPU 7 being the dominant hardware bottleneck. A larger proportion of rerouting occurred in Layers 1, 3, and 4, where alternative expert assignments were available at lower router-deviation costs.

This behaviour illustrated an important distinction between the proposed approach and simple load balancing. A strategy based on only expert utilization would preferentially target the most heavily loaded experts. In contrast, the cooling-aware optimization considers both the physical effect of a routing decision and the cost of deviating from the router's original preference. It can therefore reduce the workload of an overloaded device without necessarily modifying the routing decisions that contributed most strongly to the original imbalance.

This distinction is important to note because prior MoE systems have demonstrated that uneven expert activation can reduce hardware efficiency and have developed mechanisms for balancing or adapting execution to these dynamic workloads (9-10). The present optimization also incorporates the router's relative expert preferences, allowing hardware balancing opportunities to be distinguished according to their routing-deviation cost.

4.3 Generality and Sensitivity of the Optimization

The replication results suggest that the observed trade-off was not specific to a single input sample. At the 75% balancing target, the mean peak-workload reduction varied only from 12.67% to 13.22% across

the 5 separately seeded C4 samples. The corresponding proportion of rerouted token-layer assignments remained between 3.13% and 3.20%, while the mean routing-deviation requirement remained close to 0.01 in every run.

The similarity of these values indicates that the low-deviation balancing region observed in Run 1 was reproducible across different input samples. The persistence of the Layer 5 Expert 7 routing pattern across the 5 runs further suggests that the underlying imbalance was not produced by a small number of strange documents.

The placement sensitivity analysis also demonstrated that the optimization was not restricted to the index-aligned expert placement used in the primary experiment. Randomizing the expert-to-GPU mapping changed both the identity and magnitude of the hardware bottleneck. Across the 5 alternative balanced placements, the baseline peak-workload excess ranged from 14.94% to 27.98%.

Nevertheless, the optimization continued to achieve 75% of the available balancing improvement under every tested mapping. This required rerouting between 2.79% and 4.78% of token-layer assignments, with mean routing deviations between 0.006439 and 0.020723. The resulting reductions in maximum GPU workload ranged from 9.63% to 16.19%

This variation is expected because the amount of workload that can be removed depends on the imbalance created by the expert placement itself. A configuration that is already relatively balanced provides less potential reduction than one containing a stronger bottleneck. More importantly, the optimization formula did not need to be changed when the expert clacement changed. The general workload expression defined in (eq10) allows the same routing framework to operate with different expert-to-device mappings.

The hydraulic sensitivity analysis produced a similar distinction. Changing the pressure-flow exponent n did not alter which routing decisions were selected by the optimizer or the reduction in peak GPU workload. Instead, it changed only the magnitude of the cooling benefit predicted from that workload reduction. At the 75% balancing target, the 5-run mean workload reduction of 12.96% corresponded to the modeled pump-power reductions ranging from 12.96% for $n = 1$ to 24.18% for $n = 2$.

Therefore, the primary optimization result – the redistribution of hardware workload at low routing-deviation cost – does not depend on the assumed hydraulic exponent. The physical interpretation of this reduction in terms of pump power remains dependent on the characteristics of the cooling system.

Being able to retain the same optimization formulation across different expert placements is important because expert placement itself is an active optimization problem in distributed MoE serving. Previous work has shown that changing the distribution of experts across GPUs can alter processing imbalance and communication costs (10). In our present framework, these placement differences are represented explicitly through the matrix $M_{\ell eg}$, allowing the routing optimization to be applied without assuming a particular expert-to-device arrangement.

4.4 Limitations and Future Work

Several limitations affect the interpretations of the results.

First, the thermal and cooling quantities in this study were modeled rather than physically measured. Expert-token workload was treated as proportional to the routing-dependent component of GPU heat generation, as defined in (eq12).

This provides a useful reduced-order comparison between routing configurations, but it does not account for all components of GPU or server power consumption. Attention layers, memory access, communication, idle power, and other hardware operations also contribute to GPU and server power consumption and thus to the heat that we are trying to remove (1). The reported reductions should thus be interpreted as reductions in modeled routing-dependent thermal load instead of equivalent reductions in total GPU power, temperature, or data-center cooling energy.

Similarly, the shared-pump model simplifies the hydraulic behaviour of a real direct-to-chip liquid-cooling system. Actual systems may contain unequal branch resistances, manifold losses, control valves, nonlinear pump-efficiency curves, and transient thermal effects (2, 7-8). The sensitivity analysis over n reduces dependence on a single pressure-flow relationship but does not replace physical calibration of a cooling system.

Second, the routing-deviation measure represents the router's preference instead of a direct measurement of model-quality degradation. A small router-score difference indicates that an alternative expert received a score close to that of the originally selected expert, but it does not guarantee that substituting the alternative produces an equally accurate model output. The terms routing deviation and router preference should therefore not be interpreted as direct measures of task accuracy or language-model quality.

This distinction is integral in relation to other work that focuses on modifying expert selection for systems objectives. For example, recent energy-aware MoE research has considered replacing or skipping experts to reduce communication energy while separately constraining the resulting accuracy degradation (11). In contrast, our present study constrains deviation from the original router scores but does not directly evaluate the effect of alternative expert assignments on model output quality.

Third, the optimization was performed using routing traces recorded from the model's original forward pass. Alternative expert assignments were evaluated counterfactually using these fixed router scores. The rerouted tokens were not executed through their alternative experts and propagated through subsequent MoE layers. In a deployed system, changing an expert at an earlier layer could alter the token representation and therefore change later routing decisions.

The present results thus establish the optimization potential within observed routing traces instead of demonstrating a fully implemented online routing policy.

Future work should therefore execute the optimized routing decisions within the model itself and recompute downstream router decisions after each intervention. This would allow the relationship between routing-deviation cost and actual model quality to be evaluated using task accuracy, output loss, or other relevant performance.

A hardware implementation would provide a second important extension. Measuring GPU power, device temperature, coolant inlet and outlet temperatures, branch flow rates, pressure drop, and pump power

would allow the proportionality constant α and the hydraulic model to be calibrated experimentally. The resulting measurement could determine how much of the modeled routing-dependent cooling reduction translates into actual system-level energy savings.

Finally, the present case study uses a top-1 Switch-Base-8 model. Applying the framework to larger MoE architectures, different numbers of experts and devices, top-k routing, heterogeneous experts, and different deployment mappings would give a stronger test of its generality.

More broadly, the present study connects 2 optimization problems that have been investigated separately in great detail. MoE systems research has addressed dynamic expert workload and expert placement primarily from the perspective of computation, communication, and latency (9-10), while thermal-aware LLM serving research has optimized workload placement and request routing under power and cooling constraints (1). Recent work has also considered expert selection for communication-energy reduction (11). This study instead examines whether token-level MoE routing flexibility can serve as a control variable for modeled direct-to-chip cooling demand while explicitly accounting for the router's original expert preferences. More hardware and model-quality validation would be needed to determine a practical benefit of this approach in a deployed system.

5. Conclusion

This study investigated whether token-level routing flexibility in a mixture-of-experts model could be used to reduce modeled liquid-cooling demand while limiting departure from the model's original router preferences. Using Switch-Base-8 as a case study, expert-token assignments were mapped onto a controlled eight-GPU configuration and linked to a reduced-order direct-to-chip cooling model. Routing changes were then selected through a constrained optimization that minimized peak GPU workload while accounting for the router-deviation cost of assigning tokens to alternative experts.

The original routing configuration produced a persistent hardware workload imbalance; across 5 separately seeded C4 samples, the maximum GPU workload was, on average, approximately 21% above the equal-share reference. However, a substantial proportion of this imbalance could be reduced using relatively few routing changes. At the 75% balancing target, only 3.16% of token-layer assignments were rerouted on average, with a mean routing deviation of 0.009857, while the maximum GPU workload decreased by 12.96%.

The results also showed that expert workload alone was insufficient for identifying useful rerouting opportunities. The most heavily used expert, Layer 5 Expert 7, exhibited particularly large routing-deviation costs, meaning that directly redistributing its assignments would often require strong departures from the router's original preferences. The optimization instead exploited lower-cost routing opportunities across other layers and experts contributing to the same physical GPU workload. This demonstrated the value of jointly considering hardware workload and router preference rather than balancing expert utilization alone.

The observable trade-off was reproducible across the 5 input samples and remained present under alternative balanced expert-to-GPU mappings. The reduction in workload also translated directly into an equivalent reduction in modeled peak routing-dependent thermal load and coolant-flow requirement.

Under the tested hydraulic assumptions, the corresponding modeled pump-power reduction at the 75% balancing target ranged from 12.96% to 24.18%.

These findings suggest that MoE rerouting can provide a useful control variable for workload-aware cooling optimization. However, the results represent a trace-based counterfactual analysis instead of a deployed routing system. Future work should execute the alternative expert assignments within the model, recompute downstream routing decisions, measure any resulting change in model quality, and validate the cooling model using physical GPU and liquid-cooling measurements.

6. Supplementary Information

This research received no external funding.

The author declares no conflicts of interest.

Acknowledgements

An AI assistant (ChatGPT) was used during the preparation of this manuscript for editorial and technical support, including feedback on structure, clarity, and language; grammar and spelling editing; image generation; and processing author-generated .npz data files.

1. Stojkovic J, Zhang C, Goiri Í, Choukse E, Qiu H, Fonseca R, et al. TAPAS: Thermal- and Power-Aware Scheduling for LLM Inference in Cloud Platforms. *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 1266-1281, 2025. https://doi.org/10.1145/3676641.3716025

2. Heydari A, Soud A, Tradat M, Soud Q, Eslami B, Shahi P, et al. Experimental Evaluation of Cooling Loop Configurations for Server-Level Direct-to-Chip Liquid Cooling Systems. *J Electron Packag*, 1-25, 2026. Paper No. EP-26-1034. https://doi.org/10.1115/1.4072567

3. Fedus W, Zoph B, Shazeer N. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *J Mach Learn Res*, 23(120): 1-39, 2022. https://jmlr.org/papers/v23/21-0998.html

4. Liu J, Tang P, Wang W, Ren Y, Hou X, Heng PA, et al. A Survey on Inference Optimization Techniques for Mixture of Experts Models. *ACM Comput Surv*, 58(10): Article 247, 1-37, 2026. https://doi.org/10.1145/3794845

5. Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J Mach Learn Res*, 21(140): 1-67, 2020. https://doi.org/10.48550/arXiv.1910.10683

6. Wolf T, Debut L, Sanh V, Chaumond J, Delangue C, Moi A, et al. Transformers: State-of-the-Art Natural Language Processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38-45, 2020. https://doi.org/10.18653/v1/2020.emnlp-demos.6

7. Heydari A, Gharaibeh AR, Tradat M, Soud Q, Manaserh Y, Radmard V, et al. Experimental evaluation of direct-to-chip cold plate liquid cooling for high-heat-density data centers. *Appl Therm Eng*, 239: 122122, 2024. https://doi.org/10.1016/j.applthermaleng.2023.122122

8. Rennels D. *Pipe Flow: A Practical and Comprehensive Guide*. 2nd ed., John Wiley & Sons, Hoboken, NJ, USA, 2022. https://doi.org/10.1002/9781119756460

9. Hwang C, Cui W, Xiong Y, Yang Z, Liu Z, Hu H, et al. Tutel: Adaptive Mixture-of-Experts at Scale. *Proc Mach Learn Syst*, 5: 269-287, 2023. https://doi.org/10.48550/arXiv.2206.03382

10. Go S, Mahajan D. MoETuner: Optimized Mixture of Expert Serving with Balanced Expert Placement and Token Routing. *arXiv preprint* arXiv:2502.06643, 2025. https://doi.org/10.48550/arXiv.2502.06643

11. Chen Q, Chen X, Huang K. SiftMoE: Similarity-Aware Energy-Efficient Expert Selection for Wireless Distributed MoE Inference. *arXiv preprint* arXiv:2603.23888, 2026. https://doi.org/10.48550/arXiv.2603.23888