OPTIMALITY AND LOAD BALANCE OF GROUP-LIMITED ROUTING IN MIXTURE-OF-EXPERTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture-of-experts (MoE) language models activate only a few experts for each token, and large deployments place groups of experts on different devices. Group-limited routing bounds the number of groups per token, but not how much work each group receives. On single-domain batches of real text, the busiest group receives on average nearly twice its fair share; when groups work in parallel, it sets the pace for all of them, and a group with a buffer of fixed size must drop whatever does not fit. We give a novel analysis of group-limited routing that describes each route by how many experts it takes from each group. With this analysis, we determine exactly at which shapes the selection rule released with DeepSeek-V3-style models is score-optimal. We then prove what the existing rules cannot guarantee: with this rule, the routing score on some batches stays well below the best balanced score, whatever the group prices, and on bursty traffic every fixed bias either overloads a group or loses score. Prices set for each batch succeed there. When a linear programme sets one price per group and each token selects exactly its best route at these prices, the programme splits only a few tokens between routes. Rounding them jointly keeps every group within a fixed allowance of its target at any batch size, with a routing score at least that of the best routing that meets the target. On some batches, no routing reaches this score without an allowance. The resulting router, HYPERSPACE, needs no retraining and drops nothing. Our empirical results on $159{,}776$ recorded batches of user conversations, including prefill batches of up to $16{,}384$ tokens, show that HYPERSPACE meets both guarantees on every batch, whereas the baselines, group prices on the released rule and static replication, miss one of them on some batches. In every one of eight text domains on three MoE models, HYPERSPACE has a lower next-token loss than dropping the overflow. On GSM8K and HumanEval, its accuracy stays within four points of the model's own router; dropping loses 5.5 points on GSM8K.

1 INTRODUCTION

For each token, the router of a mixture-of-experts (MoE) layer activates only K of its many feed-forward networks, the experts (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022). Large MoE language models spread the experts over many devices. The router's choice for each token then decides how much data moves between devices and how much work each device does. To limit this traffic, models such as DeepSeek-V2 and DeepSeek-V3 keep each token's experts on at most a few devices or nodes (DeepSeek-AI, 2024a;b). This is called *group-limited routing* (DeepSeek-AI, 2025a): the experts are split into N groups, and each token picks K experts from at most M groups. Group-limited routing bounds the number of groups per token, but not how much work each group receives. When groups work in parallel, the busiest group of each batch sets the pace, and the other groups wait for it (He et al., 2026). A group with a buffer of fixed size must also drop the assignments that do not fit, and their tokens then skip experts that the router chose for them. Balance on average is therefore not enough; the load of every group must be bounded on every batch, without dropping work.

This imbalance appears in practice. On single-domain batches of real text, the busiest group of Ling-mini-2.0 (inclusionAI, 2025a) and Ring-mini-2.0 (inclusionAI, 2025b), two open models with group-limited routing, receives on average 1.84 and 1.89 times its *fair share* (the load under an equal split), respectively (Table 1). On recorded batches of user conversations, it receives 1.6 times its fair

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

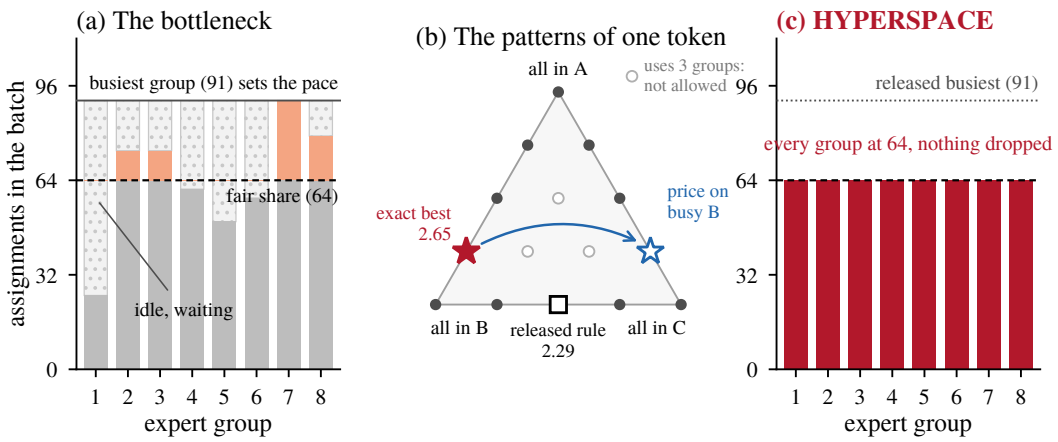

Figure 1: **The released router overloads some groups; HYPERSPACE gives every group its fair share.** (a) The released router of Ling-mini-2.0 at the first MoE layer, on the batch of 64 tokens of MATH solutions (Hendrycks et al., 2021b) whose busiest-group load is the median among this layer's batches. The batch makes 512 expert assignments, a fair share of 64 per group. The busiest group gets 91, and the other groups wait for it. (b) A toy token picks $K = 4$ experts from at most $M = 2$ of $N = 3$ groups. Each dot is a pattern, placed by how many experts come from groups A, B and C; hollow dots use all three groups and are not allowed. The released rule lands on the square and exact selection on the star. A price on the busy group B moves the choice to another allowed pattern (blue). Appendix A.1 lists the token's scores. (c) HYPERSPACE on the batch of panel (a): every group gets exactly 64 assignments, and nothing is dropped.

share (Zhao et al., 2024; Table 9). Figure 1a shows one such batch. Each MoE layer then takes as long as if every group had 1.6 to 1.9 times its fair share. Capacity limits drop the overflow (Lepikhin et al., 2021; Fedus et al., 2022), and biases tuned during training (DeepSeek-AI, 2024b) balance load only on average: they are frozen at inference (Kimi Team, 2026) and leave individual batches unbalanced.

To bound the load of every group on every batch, we study how each route loads the groups. A route is *legal* if it picks K experts from at most M groups, and its score is the sum of the router's scores for these experts. For load, all that matters about a route is how many experts land in each group, and we call this count vector the route's *pattern*, as in the cutting-stock problem (Gilmore & Gomory, 1961). The pattern of each legal route is therefore a point whose coordinates add up to K, with at most M of them nonzero (Figure 1b). A *price* on each group steers the tokens: each token takes its best pattern after paying the group price for each of its experts, and a high price moves tokens away from a crowded group (blue in Figure 1b).

Our analysis shows what the existing rules cannot guarantee (Section 4). We determine exactly at which shapes the *released rule* of DeepSeek-V3-style models (DeepSeek-AI, 2024b) is score-optimal. It can miss a token's best pattern (the square against the star in Figure 1b), and at the routing shape of DeepSeek-V3 and for nonnegative scores, it can lose $3/11$ of the best score (Theorem 4.1). With this rule, the routing score on some batches stays well below the best balanced score, whatever the group prices (Theorem 4.2). On bursty traffic, every fixed bias fails, whereas prices set for each batch succeed (Proposition 4.5).

We introduce HYPERSPACE, a routing algorithm that builds on patterns and prices (Figure 1b) and gives the guarantee that the existing rules cannot. It sets its prices for each batch by a linear programme (LP) over patterns, and at these prices it finds each token's best pattern exactly. It then rounds the few tokens that the LP splits between patterns (Algorithm 1). We adapt the iterative relaxation of Király et al. (2012) and show that every group stays within $2K - 2$ assignments of its fair share (rounded to an integer) at any batch size, and that the routing score is at least the LP value U, which no exactly balanced routing exceeds (Theorem 4.3). On some batches, no routing reaches U without an allowance, and finding the best exactly balanced routing is strongly NP-hard when the shape grows (Proposition 4.4). Nothing is dropped; on the batch of Figure 1a, every group

gets exactly its fair share (Figure 1c). HYPERSPACE needs no retraining or tuning, and every token keeps K experts within its group limit.

Our empirical results on $159{,}776$ recorded batches of user conversations, including prefill batches of up to $16{,}384$ tokens, show that HYPERSPACE never loads a group more than 4 assignments above its fair share (Tables 2, 8 and 12). Every other router that drops nothing, and static replication, misses the allowance or the score on some batches (Section 5.2). HYPERSPACE has a lower next-token negative log-likelihood (NLL) than dropping in every model and domain we test (Tables 3 and 7). On the two models with native group limits, this also holds when dropping caps each group at the largest load of HYPERSPACE. On GSM8K (Cobbe et al., 2021) and HumanEval (Chen et al., 2021), HYPERSPACE stays within four points of the released router's accuracy, whereas dropping loses 5.5 points on GSM8K (Table 4).

Our contributions are as follows. (i) We give a novel analysis of group-limited routing, in which each legal route is a pattern and group prices move tokens between patterns. (ii) With this analysis, we determine exactly at which shapes the released rule is score-optimal, and we prove that neither group prices on this rule nor any fixed bias can guarantee balanced loads with the best balanced score (Section 4). (iii) As an application, we introduce HYPERSPACE, a novel routing algorithm that keeps every group within $2K - 2$ of its target with a score of at least U, and on real traffic it meets both guarantees at a lower cost in quality than dropping (Section 5).

2 Setting and Related Work

We first state the routing problem and then discuss related work.

2.1 Group-Limited Routing

An MoE layer has E routed experts, split into N groups $G_1, \dots, G_N$ of L experts each, and each token picks K experts from at most M groups ($1 \le M \le N$, $1 \le K \le ML$). DeepSeek-V3, Ling-mini-2.0 and Ring-mini-2.0 use $(N, L, M, K) = (8, 32, 4, 8)$, which we call the *deployed shape*. The router gives every token one score per expert, a vector $p \in \mathbb{R}^E$. A route is a $0/1$ vector z over the experts. The set $\mathcal{F}$ of legal routes and the best score $H(p)$ that a legal route can reach are

$$\mathcal{F} = \Big\{ z \in \{0,1\}^E : \mathbf{1}^\top z = K,\ \#\{g : \textstyle\sum_{e \in G_g} z_e > 0\} \le M \Big\}, \qquad H(p) = \max_{z \in \mathcal{F}} p^\top z. \quad (1)$$

The *routing score* $p^\top z$ measures how well a route follows the router's preferences.

Released rule. The released rule of DeepSeek-V3-style models (DeepSeek-AI, 2024b; inclusionAI, 2025a;b) works in two steps. It scores each group by the sum of its $q = K/M \in \mathbb{N}$ highest expert scores and keeps the best M groups; it then takes the K highest-scoring experts in the kept groups. Every tie goes to the lower index. The scores used for selection include the *trained bias*, a per-expert bias tuned during training, which affects the choice of experts but not their output weights. The rule treats each token on its own, and nothing limits the load of a group.

Patterns. The pattern o of a route counts how many of its K experts fall in each group. The legal patterns and the best score $v(o)$ of a pattern are

$$\mathcal{O} = \big\{ o \in \{0, \dots, L\}^N :\ \mathbf{1}^\top o = K,\ |\{g : o_g > 0\}| \le M \big\}, \qquad v(o) = \textstyle\sum_g S_g(o_g), \quad (2)$$

where $S_g(j)$ is the sum of the j highest scores in group g. For a given pattern, the best route takes the o_g highest-scoring experts of each group g. Choosing a route therefore amounts to choosing a pattern, and $H(p) = \max_{o \in \mathcal{O}} v(o)$. The released rule ranks groups by $S_g(q)$, as if the token took q experts from each kept group. *Exact selection* instead maximises v over all of $\mathcal{O}$.

Load and the LP. A *routing step* is one MoE layer routing one batch of B tokens, and a *routing* gives every token of the batch a legal route. Each token makes K expert assignments, and the load Q_g of group g is the number of assignments it receives. The fair share is $\tau = BK/N$. Each group has a *target interval* $[\ell_g, h_g]$ with integer ends that contains τ; the *nearest-integer target* is $[\lfloor \tau \rfloor, \lceil \tau \rceil]$. Our goal is to find the optimal routing: the largest total routing score under the constraint that every

group's load stays in its interval. If each token may mix legal patterns, the best total score is the value U of the LP

$$U = \max_{w \geq 0} \Big\{ \sum_{t,o} w_{to} v_t(o) : \sum_{o} w_{to} = 1 \ \forall t, \quad \ell_g \leq \sum_{t,o} w_{to} o_g \leq h_g \ \forall g \Big\}, \tag{3}$$

where v_t is the pattern value (2) for the scores of token t, and w_{to} is the weight that token t puts on pattern $o \in \mathcal{O}$. The patterns of a legal routing that meets the intervals, each with weight 1, form a feasible point of (3) whose value is at least the routing's score. Hence U bounds it from above.

2.2 Related Work

Existing approaches fall into four families. (i) **Training-time methods** balance load with auxiliary losses (Lepikhin et al., 2021; Fedus et al., 2022), per-expert biases (DeepSeek-AI, 2024b; Han & Zhong, 2025; Kimi Team, 2026) or balanced assignments of tokens to experts (Lewis et al., 2021; Clark et al., 2022; Zhou et al., 2022; Dong et al., 2025). The biases balance load only on average, and the assignment methods change how the model is trained. (ii) **Capacity limits** drop the assignments that exceed an expert's capacity (Lepikhin et al., 2021; Fedus et al., 2022), or move them to local experts with room (He et al., 2026). (iii) **Inference-time rerouting** cuts the number of experts a batch activates, overall or per device (Vankov et al., 2026; Gupta et al., 2024; Wu et al., 2026; Oncescu et al., 2026). (iv) **Placement methods** replicate or move experts across devices (DeepSeek-AI, 2025a; Zhao et al., 2026; Yang et al., 2026; Nguyen et al., 2026; Wei et al., 2026; Huang et al., 2024), spread each batch over the replicas (DeepSeek-AI, 2025b; NVIDIA Team, 2026), schedule tokens across devices (Zhao et al., 2025) or co-locate experts used together (Luo et al., 2025). Architecture changes need training: Ada-K learns how many experts each token uses (Yue et al., 2025), and mixture of grouped experts fixes each token's number of experts in every group (Tang et al., 2025). HYPERSPACE works at inference time, as (ii) and (iii) do, but it drops nothing, bounds each group's load on every batch and changes token routes rather than expert placement.

3 The HYPERSPACE Router

Scores and outputs. HYPERSPACE takes the router scores of one batch, the shape (N, L, M, K) and a target interval for every group, and it returns a legal route for every token. The score a_{te} of expert e for token t is its gate without the trained bias, divided by the sum of the gates of the token's released route. This puts every token on the same scale (Appendix B). Only the choice of experts changes: the model weighs the selected experts by their gates, as it does for the released route.

Procedure. Algorithm 1 sets the released rule and HYPERSPACE side by side. Tokens compete for groups, and prices resolve this competition. *Step 1: pricing.* In each round, every token proposes its best legal pattern at the current group prices, where each chosen expert pays the price of its group. A dynamic programme (Baldassarre et al., 2016) finds this pattern exactly (Lemma A.3). The LP (3), restricted to the patterns proposed so far, then recomputes the prices as its dual variables: crowded groups become more expensive. This is column generation (Dantzig & Wolfe, 1960; Gilmore & Gomory, 1961). The rounds stop when no token can improve at the current prices. With slack variables of large enough cost in the load rows, the restricted LP has then reached U (Lemma A.2). *Step 2: rounding.* A corner (basic) solution of the LP splits at most $N - 1$ tokens between patterns, and every other token uses a single pattern. HYPERSPACE rounds the split tokens jointly: among the combinations whose total value is at least U, it takes the one closest to the intervals. This *min-deviation rounding* compares at most 2^{N-1} combinations, as Potts (1985) does for split jobs. *Step 3: placement.* Each token may use the groups of its pattern and, up to M groups in all, its other groups with the highest single expert score. One max-score network flow over the batch places the K experts of every token in these groups (Schrijver, 2003), and it moves no group's load further from its interval. The o_g highest-scoring experts of each group form a feasible placement. Hence the flow keeps the score at least U and every load within $2K - 2$ of its interval (Theorem 4.3).

Design choices. HYPERSPACE uses exact selection because, with the released rule, no group prices reach U on some batches (Theorem 4.2). It sets the prices for each batch because on bursty traffic every fixed bias fails (Proposition 4.5). It rounds only the split tokens, and its allowance therefore does not grow with the batch (Theorem 4.3).

Algorithm 1 HYPERSPACE replaces the per-token selection of the released rule with a joint step over the batch (shaded).

Require: scores a_t of the B tokens; shape (N, L, M, K), $q = K/M$; target intervals $[\ell_g, h_g]$
Released rule: each token on its own, no load control
R1: rank the groups by the sum of the token's top q scores and keep the best M
R2: take the token's top K experts inside the kept groups
HYPERSPACE: all tokens of the step jointly
start from zero group prices, $\mu = 0$
repeat
  each token picks its best legal pattern at the prices, $\max_o [v_t(o) - \mu^\top o]$ (exact)
  the LP over the patterns found so far recomputes μ (crowded groups cost more)
until no token improves at the prices; the LP value U is reached
round the at most $N - 1$ split tokens jointly: score at least U, loads within $2K - 2$
place each token's K experts in its groups by a max-score flow within the bound
Ensure: legal routes, nothing dropped, score at least U, every group within $2K - 2$ of its interval

Complexity. Each round solves one LP over the proposed patterns and, for every token, a dynamic programme with $O(NMK \min\{K, L\})$ operations. Placement solves one network-flow problem.

4 THEORY

Group prices steer tokens only through the rule that selects their routes. For one token with scores p, $H(p)$ is the score of its best legal route (1), and $P(p)$ that of the route the released rule picks when it ranks the experts by p. Unless a sign is stated, the results hold for any scores. Appendix A gives the proofs.

Theorem 4.1 (When the released rule is exact, and its worst case)**.** *Let $q \in \{1, \dots, L\}$ be an integer and $K = Mq$. (a) $P(p) = H(p)$ for all scores p if and only if $M = 1$, $M = N$, $q = 1$ or $q = L$; at every other shape, including the deployed one, some scores make the rule pick a worse route. (b) If $1 < M < N$ and $q > 1$, then $P(p) \geq \rho\, H(p)$ for all nonnegative scores p, with the best possible constant $\rho = Mq/(Mq + D)$, where D depends only on the shape and is given in* (6)*, and $\rho \geq 1/(2 - 1/M)$. At the deployed shape, $\rho = 8/11$.*

The released rule values each group as if the token took exactly q experts from it, and a token whose best pattern is uneven can therefore lose score. In Figure 1b, the best pattern takes one expert from A and three from B, but A has the lowest top-two sum, and the rule keeps B and C. On recorded user traffic (Section 5.2), the rule picks a worse route than exact selection for 12.8% and 13.5% of the token-layer pairs of the two models (Table 11).

To see how prices balance a batch, fix a batch and its target intervals (Section 2.1). At group prices μ, the dual function of the LP (3) is

$$D_G(\mu) = \sum_t \max_{o \in \mathcal{O}} \left[v_t(o) - \mu^\top o \right] + \sum_g \left(h_g \mu_g^+ - \ell_g \mu_g^- \right),$$

where $\mu_g^\pm = \max(\pm\mu_g, 0)$. By LP duality, $\min_\mu D_G(\mu) = U$ (Lemma A.2). Let $Q(\mu)$ be the group loads when every token takes a best legal pattern at the prices μ. Auxiliary-loss-free balancing moves per-expert biases against the loads (DeepSeek-AI, 2024b), and we ask whether group prices moved in this way reach U.

Theorem 4.2 (Group prices need exact selection)**.** *Let every target interval contain τ. (a) With exact selection and a step $\eta > 0$, the update $\mu \leftarrow \Pi\big[\mu + \eta\,(Q(\mu) - \mathrm{target}(\mu))\big]$ is projected subgradient descent on D_G, where Π projects onto a closed convex set containing the initial prices and a minimiser of D_G, and $\mathrm{target}_g(\mu)$ is h_g, ℓ_g or any point of $[\ell_g, h_g]$ when μ_g is positive, negative or zero, respectively. (b) If $K = Mq$ with integers $1 < M < N$ and $1 < q < L$, then for every batch size B that is a multiple of $2N$ there are batches with scores in $[0, 1]$ and $U = BK$ on which, for every group-price vector μ, the routing that the released rule picks at the prices μ scores at most $U - B/8$; at the deployed shape, $B/8 = U/64$.*

Part (a) is the dual subgradient method (Nedić & Ozdaglar, 2009). Han & Zhong (2025) and Kimi Team (2026) read per-expert biases as dual prices, without group limits. With inexact subgradients, such methods reach the optimum within a tolerance set by the errors (Nedić & Bertsekas, 2010). The released rule is such an inexact best response: on the batches of (b), its routing stays well below the best balanced score, whatever the group prices. Hence HYPERSPACE uses exact selection.

Theorem 4.3 (Rounding guarantee and certificate)**.** *Let every target interval have integer ends and contain τ. (a) A corner (basic) optimum of the LP* (3) *splits at most $N-1$ tokens between patterns. (b) For a basic optimum, min-deviation rounding picks one of each split token's patterns such that the total pattern value stays at least U and, whatever the batch size, the group loads T_g of the chosen patterns satisfy*

$$\ell_g - (2K-2) \;\le\; T_g \;\le\; h_g + (2K-2), \tag{4}$$

with $2K-2 = 14$ at the deployed shape. (c) For any prices μ, $D_G(\mu)$ bounds from above the score of every legal routing that meets the intervals. From a basic optimum, the routing that HYPERSPACE returns scores at least $U = \min_\mu D_G(\mu)$, keeps every group load Q_g between $\min\{\ell_g, T_g\}$ and $\max\{h_g, T_g\}$, and therefore within the bound of (b), and is optimal whenever it meets the intervals.

By (b) and (c), a buffer of $h_g + 2K - 2$ slots per group never overflows, and nothing is dropped: at the deployed shape with $B = 64$ and $h_g = 64$, 78 slots suffice. Part (a) is the classical bound on split jobs in assignment-type LPs (Benders & van Nunen, 1983; Potts, 1985; Lenstra et al., 1990), and (c) is weak LP duality with integral flows (Schrijver, 2003). Part (b) adapts the iterative relaxation of Király et al. (2012) as in Bérczi et al. (2019, Algorithm 2), whose Theorem 1 gives $2K-1$ here; integer ends give $2K-2$ (Appendix A.4), and with upper bounds only the method gives $h_g + K - 1$ (Király et al., 2012; Bérczi et al., 2019). We then ask whether the allowance can be removed.

Proposition 4.4 (Some allowance is necessary, and exact balance is hard)**.** *Let $\ell_g = h_g = \tau$ for every group, with τ an integer. (a) If $N \ge K$, $L \ge K$ and $2 \le M < K$, then for every batch size B that is a multiple of N with $B \ge 2N$ there are batches with scores in $\{0,1,2\}$ on which every legal routing with score at least U puts at least $\tau + K - M$ assignments on some group. (b) Deciding whether some legal routing puts exactly τ assignments on every group and has score at least a given value is strongly NP-complete when N, L and K are part of the input, already for $M = 2$, $L = K$ and scores in $\{0,1,2\}$.*

As for degree-bounded matroids, where even feasibility is NP-complete (Király et al., 2012), some allowance is therefore the cost of reaching U: at the deployed shape and $B = 64$, (a) forces a load of 68 on some batches, the largest load HYPERSPACE reaches on the 64-token batches of Section 5. The reduction in (b) embeds 3-PARTITION (Garey & Johnson, 1979) and lets the shape grow; at a fixed shape, an exact dynamic programme is polynomial in B (Appendix A.5).

A *batch-blind* policy routes each token from its own scores alone, for example the released rule with any fixed bias. A batch can be dominated by one kind of text, such as all math, all code or all one language. We ask whether any fixed bias can balance such batches.

Proposition 4.5 (Bursty traffic defeats every fixed bias)**.** *Let $N \ge 2$, $K \le L$, N divide B and $0 \le \varepsilon < 1/2$. There are random traffic mixes of two document types, with a gap $\gamma > 0$ between high and low scores, on which every batch-blind policy, with probability at least $1/2$, either loads some group above $(1+\varepsilon)\,\tau$ or loses at least $\frac{\gamma}{2}(\frac{1}{2} - \varepsilon)\,\tau$ in score against the balanced optimum of the batch (the value U for $\ell_g = h_g = \tau$), whereas exact selection at prices set for each batch reaches that optimum with every group load equal to τ.*

Prices learned from past requests are near-optimal when requests arrive in random order (Devanur & Hayes, 2009), and Han & Zhong (2025) analyse the bias update in a stochastic setting; bursty batches break both (Appendix A.6). The prices must therefore be set for the batch at hand.

5 Experiments

We ask whether HYPERSPACE keeps its guarantees on real batches, and at what cost in quality.

5.1 Setup

Models and data. We use Ling-mini-2.0 and Ring-mini-2.0 (inclusionAI, 2025a;b), two openly released models with the deployed shape $(N, L, M, K) = (8, 32, 4, 8)$. We evaluate on eight text

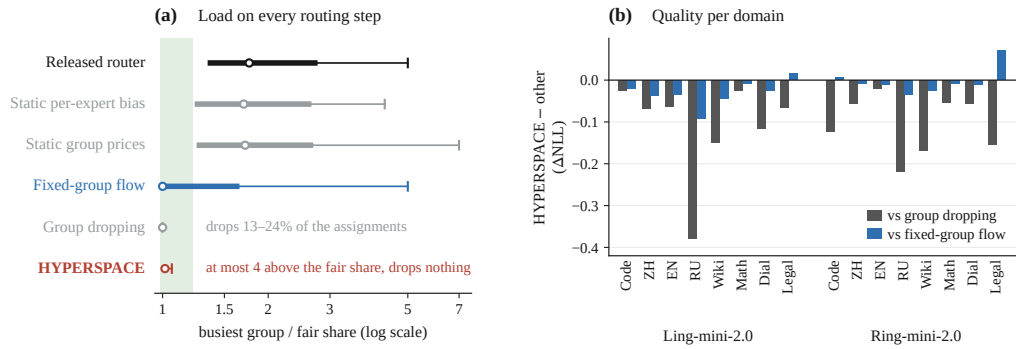

Figure 2: **Load and quality.** (a) For each router, the load of the busiest group over the fair share, on every routing step of both models. Thick bar: middle 90% of steps; circle: median; whisker: maximum. Green band: the guaranteed allowance of 14 assignments. (b) For each domain, the NLL of HYPERSPACE minus that of group dropping with a cap of 64 and of fixed-group flow. Below zero, HYPERSPACE has the lower NLL. ZH, EN, RU: Chinese, English and Russian conversation.

domains: code, Chinese, English and Russian conversation, Wikipedia, math, multi-turn dialogue and legal text (Appendix B). Each routing step routes $B = 64$ tokens, one from each of 64 sequences of one domain, as in decoding. We score the next-token predictions at 128 positions of 32 of these sequences, 4,096 tokens per domain. The fair share is then $\tau = 64$, and the nearest-integer target asks every group for exactly 64. The router under test runs at every MoE layer, and later layers therefore see the effect of earlier routing. All load-controlling routers use the nearest-integer target.

Routers. The released router, the model's own group-limited rule with its trained bias (Section 2.1), is the reference and does not control load. As a control, the released rule without the bias applies the same rule to the gates alone. HYPERSPACE also leaves the bias out of the scores it ranks (Section 3). Group dropping applies expert capacity (Lepikhin et al., 2021; Fedus et al., 2022) to groups, as DeepSeek-V2 does (DeepSeek-AI, 2024a): each keeps its highest-gate assignments up to a cap. Fixed-group flow keeps the groups that the released router chose and moves experts within them to meet the target. If it cannot, the whole step falls back to the released routes. The static per-expert bias and the static group prices are calibrated offline with the sign update of auxiliary-loss-free balancing (DeepSeek-AI, 2024b), then frozen, and are therefore batch-blind (Proposition 4.5).

Metric. We measure quality by the NLL of the next token. Each domain counts as one data point: we report the mean NLL difference over the eight domains with its standard error (SE), and the number of domains in which HYPERSPACE has the lower NLL.

5.2 Load on Every Batch

Single-domain batches. On every routing step of both models, HYPERSPACE drops nothing and never exceeds the target by more than 4 assignments, well inside the guaranteed 14 (Figure 2a and Table 1). The static biases and fixed-group flow miss the target: the biases do not see the batch (Proposition 4.5), and the flow falls back to the released routes on some steps. Group dropping with a cap of 64 meets the target by discarding 13 to 24% of the assignments.

Recorded user traffic. To test the bound on user traffic and on larger batches, we replay router scores recorded from both models on English WildChat conversations (Zhao et al., 2024) that appear in no evaluation domain. A *decoding batch* takes one token from each of 64 conversations, and a *prefill batch* takes the first 512 tokens of four conversations (Appendix B). On these batches, the busiest group of the released router receives on average 1.6 times its fair share (Table 9). The replay also includes two routers that drop nothing. *Capacity-aware rerouting*, a group-level heuristic modelled on the capacity-aware inference of He et al. (2026), moves the overflow of each full group to the best legal expert in a group with room. *Group prices on the released rule* set prices for each batch with the update of Theorem 4.2, but with the released rule in place of exact selection, and

Table 1: **Load over all** 24,320 **routing steps of each model** (19 MoE layers, 160 positions and eight domains). Busiest/fair: mean over the steps of the load of the busiest group over the fair share of 64 assignments. Excess: largest number of assignments by which the busiest group exceeds 64 (guaranteed allowance of HYPERSPACE: 14). Dropped: share of assignments discarded, range over domains. Fixed-group flow falls back to the released routes on 0.6 to 19.3% of a domain's steps.

	Ling-mini-2.0			Ring-mini-2.0		
Router	Busiest/fair	Excess	Dropped (%)	Busiest/fair	Excess	Dropped (%)
Released router	1.84	256	0	1.89	256	0
Released rule without the bias	1.84	196	0	1.87	201	0
Static per-expert bias	1.80	211	0	1.78	192	0
Static group prices	1.82	320	0	1.82	384	0
Group dropping	1.00	0	12.7–23.7	1.00	0	15.5–23.1
Fixed-group flow	1.07	256	0	1.08	256	0
HYPERSPACE	1.01	4	0	1.01	4	0

Table 2: **Replay of router scores recorded from both models on user conversations**: 155,648 decoding and 768 prefill batches. Over +14: batches whose busiest group exceeds the fair share by more than the allowance of 14. Score/U: mean over the batches of the routing score divided by U, the batch's LP value. It can exceed 1 only when the loads leave the target. Group dropping discards 13.8% (decoding) and 12.0% (prefill) of the assignments, and the other routers drop none.

	Decoding, $B = 64$		Prefill, $B = 2{,}048$	
Router	Over +14	Score/U	Over +14	Score/U
Released router	150,493	1.0065	768	1.0030
Group dropping	0	0.9232	0	0.9294
Fixed-group flow	293	0.9829	26	0.9864
Capacity-aware rerouting	37	0.9738	3	0.9751
Group prices on the released rule, 20 iterates	1,007	1.0058	760	1.0063
200 iterates	0	1.0005	102	0.9987
200 iterates, oracle step size	0	0.9990	0	0.9985
HYPERSPACE	0	1.0004	0	1.0000

return the best of the first 20 or 200 iterates. The oracle variant also picks the step size for each batch, in hindsight, from five values.

On every batch, including prefill batches of up to 16,384 tokens, HYPERSPACE keeps every group within 4 assignments of the target and scores at least U (Tables 2, 8 and 12). With 200 iterates, group prices on the released rule stay within the allowance on every decoding batch, with Score/U 1.0005 against 1.0004 there, but exceed it on 13% of the prefill batches at 2,048 tokens and 64% at 16,384. With the oracle step size they exceed it once, but score below U on average. The other routers that drop nothing exceed the allowance on both kinds of batch. Static replication does not bound the load of a batch either. Even with 8 redundant experts per GPU, EPLB placements leave the busiest GPU more than 14 assignments above its fair share on 7.6 to 14.4% of the batches (Table 9).

5.3 Quality

Two models with native group limits. HYPERSPACE has lower NLL than group dropping with a cap of 64 in all 16 model-domain pairs, by about 0.11 on average on both models (Table 3). With a cap of 68, the largest load HYPERSPACE reaches, group dropping still has the higher NLL in all 16 pairs, by 0.06–0.07 on average. HYPERSPACE also has lower NLL than fixed-group flow in 13 of the 16 pairs, and Figure 2b shows the differences for each domain. Against the released router, which ignores the target, load control has a cost that depends on the domain (Table 6). For HYPERSPACE it is at most 0.12 in seven of the eight domains, and 0.59 and 0.62 on Russian conversation, where group dropping and fixed-group flow lose more. The released rule without the bias, which leaves the bias out as HYPERSPACE does, changes the mean NLL by only -0.010 ± 0.003 and $+0.004 \pm 0.006$ on the two models, and it leaves the loads as uneven as the released router does.

Table 3: **NLL of HYPERSPACE minus that of another router**: mean $\pm$ SE over the eight domains and the number of domains ($k/8$) where HYPERSPACE is lower. Group dropping caps each group at 64, the fair share, or at 68, the largest load of HYPERSPACE (a separate paired run, Appendix B).

	vs. group dropping, 64		vs. group dropping, 68		vs. fixed-group flow	
Model	ΔNLL $\pm$ SE	$k/8$	ΔNLL $\pm$ SE	$k/8$	ΔNLL $\pm$ SE	$k/8$
Ling-mini-2.0	-0.1114 ± 0.0412	8/8	-0.0668 ± 0.0181	8/8	-0.0307 ± 0.0109	7/8
Ring-mini-2.0	-0.1061 ± 0.0248	8/8	-0.0605 ± 0.0206	8/8	-0.0028 ± 0.0113	6/8

Table 4: **Accuracy (%) with Ling-mini-2.0** and its change Δ in points from the released router, computed from solved counts. The router under test routes every prompt and generated token, with greedy decoding, on all 1,319 GSM8K test and 164 HumanEval problems (pass@1).

	GSM8K		HumanEval	
Router	Acc.	Δ	Acc.	Δ
Released router	78.9		73.8	
Group dropping	73.5	-5.5	73.2	-0.6
HYPERSPACE	76.8	-2.1	70.1	-3.7

A third model. On Qwen3-30B-A3B (Qwen Team, 2025), with 8 imposed groups and at most 4 per token, HYPERSPACE again has lower NLL than group dropping in all eight domains, by 0.017 ± 0.006 on average, and costs 0.018 against the released rule at this shape (Table 7).

5.4 Downstream Tasks and Computational Cost

Downstream tasks. With Ling-mini-2.0 on GSM8K, HYPERSPACE loses 2.1 points against the released router, and group dropping loses 5.5 (Table 4). On HumanEval, HYPERSPACE solves six fewer problems, and group dropping one fewer. On three multiple-choice tasks, routing the prompt with HYPERSPACE changes the accuracy by -1.1 to $+1.6$ points (Table 10).

Table 5: **Time in ms of one routing step of one MoE layer on 64 tokens**, from the router scores to the expert choices on the GPU, with one NVIDIA H200 GPU and two Intel Xeon Gold 6548Y+ host CPUs. HYPERSPACE: 192 recorded batches, without the placement flow; released router: 20 repetitions on one synthetic batch, in a separate run (Appendix B).

Router	Median	95th percentile
Released router	0.16	0.17
HYPERSPACE	26.9	30.6

Computational cost. On Ling-mini-2.0 batches of 64 tokens, the median routing step of HYPERSPACE without the placement flow takes 26.9 ms, against 0.16 ms for the released router on a synthetic batch (Table 5).

6 Conclusion

Group-limited routing does not limit how much work each group receives, and the released rule does not always pick the best legal route. HYPERSPACE fixes both with exact selection, group prices set for each batch and a joint rounding of the few split tokens. We gave an analysis in which each legal route is a pattern. In it, we proved that group prices on the released rule and fixed biases cannot guarantee balanced loads with the best balanced score, whereas HYPERSPACE keeps every group within $2K - 2$ of its target interval at any batch size, with a score of at least U. On real traffic, HYPERSPACE bounded the load of every group on every batch without dropping work, and it had a lower NLL than dropping on every model and domain we tested.

AI USE STATEMENT

Large language model assistants were used in three ways. *Writing:* to edit and polish the text, captions and tables. *Retrieval and discovery:* to search for and summarise related work; every cited work was read, checked against its source before citation. *Research ideation and execution:* to brainstorm and check derivations, to write and review training, evaluation and plotting code, and to manage cluster runs; every reported number comes from logged runs. All claims, proofs, numbers, and code were verified by the authors with independent checks before inclusion, and the authors take full responsibility for the final content.

ETHICS STATEMENT

This work studies routing in released MoE language models on public datasets, used under their licences, and involves no human subjects. HYPERSPACE routes each token using the whole batch, and a token's experts can therefore depend on other requests in the same batch. Deployments that need isolation between users should take this into account.

REPRODUCIBILITY STATEMENT

Appendix A gives complete proofs, and Appendix B gives the evaluation protocol. We will make the code and the data manifests public.

REFERENCES

Luca Baldassarre, Nirav Bhan, Volkan Cevher, Anastasios Kyrillidis, and Siddhartha Satpathi. Group-sparse model selection: Hardness and relaxations. *IEEE Transactions on Information Theory*, 62(11):6508–6534, 2016. doi: 10.1109/TIT.2016.2602222. URL https://doi.org/10.1109/TIT.2016.2602222.

J. F. Benders and J. A. E. E. van Nunen. A property of assignment type mixed integer linear programming problems. *Operations Research Letters*, 2(2):47–52, 1983. doi: 10.1016/0167-6377(83)90035-4.

Kristóf Bérczi, André Berger, Matthias Mnich, and Roland Vincze. Degree-Bounded Generalized Polymatroids and Approximating the Metric Many-Visits TSP. arXiv preprint arXiv:1911.09890, 2019. URL https://arxiv.org/abs/1911.09890.

Mark Chen et al. Evaluating Large Language Models Trained on Code. arXiv preprint arXiv:2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

Aidan Clark, Diego De Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, George Bm Van Den Driessche, Eliza Rutherford, Tom Hennigan, Matthew J Johnson, Albin Cassirer, Chris Jones, Elena Buchatskaya, David Budden, Laurent Sifre, Simon Osindero, Oriol Vinyals, Marc'Aurelio Ranzato, Jack Rae, Erich Elsen, Koray Kavukcuoglu, and Karen Simonyan. Unified Scaling Laws for Routed Language Models. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4057–4086. PMLR, 2022. URL https://proceedings.mlr.press/v162/clark22a.html.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge, 2018. URL https://arxiv.org/abs/1803.05457. preprint.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems. arXiv preprint arXiv:2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

CodeParrot. codeparrot-clean-valid. Hugging Face dataset, 2022. URL https://huggingface.co/datasets/codeparrot/codeparrot-clean-valid.

George B. Dantzig and Philip Wolfe. Decomposition principle for linear programs. *Operations Research*, 8(1):101–111, 1960. doi: 10.1287/opre.8.1.101.

DeepSeek-AI. DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model, 2024a. URL https://arxiv.org/abs/2405.04434. preprint.

DeepSeek-AI. DeepSeek-V3 Technical Report, 2024b. URL https://arxiv.org/abs/2412.19437. preprint.

DeepSeek-AI. Expert Parallelism Load Balancer (EPLB). GitHub repository, 2025a. URL https://github.com/deepseek-ai/EPLB.

DeepSeek-AI. Linear-Programming-Based Load Balancer (LPLB). GitHub repository, 2025b. URL https://github.com/deepseek-ai/LPLB.

Nikhil R. Devanur and Thomas P. Hayes. The Adwords Problem: Online Keyword Matching with Budgeted Bidders under Random Permutations. In *Proceedings of the 10th ACM Conference on Electronic Commerce*, pp. 71–78. ACM, 2009. doi: 10.1145/1566374.1566384. URL https://dl.acm.org/doi/10.1145/1566374.1566384.

Bowen Dong, Yilong Fan, Yutao Sun, Zhenyu Li, Tengyu Pan, Zhou Xun, and Jianyong Wang. Maximum Score Routing For Mixture-of-Experts. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12619–12632. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.653. URL https://aclanthology.org/2025.findings-acl.653/.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://jmlr.org/papers/v23/21-0998.html.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The Language Model Evaluation Harness. Zenodo, July 2024. URL https://zenodo.org/records/12608602.

Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, San Francisco, 1979.

Paul C. Gilmore and Ralph E. Gomory. A linear programming approach to the cutting-stock problem. *Operations Research*, 9(6):849–859, 1961. doi: 10.1287/opre.9.6.849.

Vima Gupta, Jae Hyung Ju, Kartik Sinha, Ada Gavrilovska, and Anand Padmanabha Iyer. Lynx: Enabling Efficient MoE Inference through Dynamic Batch-Aware Expert Selection, 2024. URL https://arxiv.org/abs/2411.08982. preprint.

S.-P. Han and O. L. Mangasarian. Exact Penalty Functions in Nonlinear Programming. *Mathematical Programming*, 17(1):251–269, 1979. doi: 10.1007/BF01588250. URL https://link.springer.com/article/10.1007/BF01588250.

X. Y. Han and Yuan Zhong. A Theoretical Framework for Auxiliary-Loss-Free Load-Balancing of Sparse Mixture-of-Experts in Large-Scale AI Models. In *ML x OR: Mathematical Foundations and Operational Integration of Machine Learning for Uncertainty-Aware Decision-Making, NeurIPS 2025 Workshop*, 2025. URL https://openreview.net/forum?id=sQHZ2w8UzP.

Shwai He, Weilin Cai, Jiayi Huang, and Ang Li. Capacity-Aware Inference: Mitigating the Straggler Effect in Mixture of Experts. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/94e845868a9ace4bc239d0c529d32f4c-Abstract-Conference.html.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*, 2021a. URL https://arxiv.org/abs/2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Haiyang Huang, Newsha Ardalani, Anna Sun, Liu Ke, Hsien-Hsin S. Lee, Shruti Bhosale, Carole-Jean Wu, and Benjamin Lee. Toward Efficient Inference for Mixture of Experts. In *Advances in Neural Information Processing Systems*, volume 37, pp. 84033–84059, 2024. doi: 10.52202/079017-2670. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/98bf3b8505c611ac21055dd9d355c66e-Abstract-Conference.html.

Qi Huangfu and J. A. Julian Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142, 2018. doi: 10.1007/s12532-017-0130-5.

inclusionAI. Ling-mini-2.0. Hugging Face model release, 2025a. URL https://huggingface.co/inclusionAI/Ling-mini-2.0.

inclusionAI. Ring-mini-2.0. Hugging Face model release, 2025b. URL https://huggingface.co/inclusionAI/Ring-mini-2.0.

Kimi Team. Kimi K3: Open Frontier Intelligence, 2026. URL https://arxiv.org/abs/2607.24653. preprint.

Tamás Király, Lap Chi Lau, and Mohit Singh. Degree Bounded Matroids and Submodular Flows. *Combinatorica*, 32(6):703–720, 2012. doi: 10.1007/s00493-012-2760-6. URL https://link.springer.com/article/10.1007/s00493-012-2760-6.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22199–22213, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

Jan Karel Lenstra, David B. Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46(1–3):259–271, 1990. doi: 10.1007/BF01585745. URL https://link.springer.com/article/10.1007/BF01585745.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qrwe7XHTmYb.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. BASE Layers: Simplifying Training of Large, Sparse Models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6265–6274. PMLR, 2021. URL https://proceedings.mlr.press/v139/lewis21a.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Shuqing Luo, Pingzhi Li, Jie Peng, Yang Zhao, Yu Cao, Yu Cheng, and Tianlong Chen. Occult: Optimizing collaborative communications across experts for accelerated parallel MoE training and inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 41235–41253. PMLR, 2025. URL https://proceedings.mlr.press/v267/luo25f.html.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer Sentinel Mixture Models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Angelia Nedić and Dimitri P. Bertsekas. The Effect of Deterministic Noise in Subgradient Methods. *Mathematical Programming*, 125(1):75–99, 2010. doi: 10.1007/s10107-008-0262-5. URL https://link.springer.com/article/10.1007/s10107-008-0262-5.

Angelia Nedić and Asuman Ozdaglar. Approximate Primal Solutions and Rate Analysis for Dual Subgradient Methods. *SIAM Journal on Optimization*, 19(4):1757–1780, 2009. doi: 10.1137/070708111. URL https://epubs.siam.org/doi/10.1137/070708111.

Xuan-Phi Nguyen, Shrey Pandit, Austin Xu, Caiming Xiong, and Shafiq Joty. Least-Loaded Expert Parallelism: Load Balancing An Imbalanced Mixture-of-Experts. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=DFi6hHce5t.

NVIDIA Team. Improving DeepEP MoE load balance in SGLang with Waterfill and LPLB. LMSYS technical article, June 2026. URL https://www.lmsys.org/blog/2026-06-26-waterfill-lplb.

Costin-Andrei Oncescu, Qingyang Wu, Wai Tong Chung, Tsai-chuan Wu, Bryan Dev Gopal, Junxiong Wang, Tri Dao, and Ben Athiwaratkun. Opportunistic Expert Activation: Batch-Aware Expert Routing for Faster Decode Without Retraining. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=bSLIDztFwx.

C. N. Potts. Analysis of a linear programming heuristic for scheduling unrelated parallel machines. *Discrete Applied Mathematics*, 10(2):155–164, 1985. doi: 10.1016/0166-218X(85)90009-5.

Qwen Team. Qwen3 Technical Report, 2025. URL https://arxiv.org/abs/2505.09388. preprint.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8583–8595, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/48237d9f2dea8c74c2a72126cf63d933-Abstract.html.

Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, Berlin, 2003. ISBN 978-3-540-44389-6. URL https://link.springer.com/book/9783540443896. Volume A.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations*, 2017.

Yehui Tang, Xiaosong Li, Fangcheng Liu, Wei Guo, Hang Zhou, Yaoyuan Wang, Kai Han, Xianzhi Yu, Jinpeng Li, Hui Zang, Fei Mi, Xiaojun Meng, Zhicheng Liu, Hanting Chen, Binfan Zheng, Can Chen, Youliang Yan, Ruiming Tang, Peifeng Qin, Xinghao Chen, Dacheng Tao, Yunhe Wang, et al. Pangu Pro MoE: Mixture of Grouped Experts for Efficient Sparsity, 2025. URL https://arxiv.org/abs/2505.21411. preprint.

Daniil Vankov, Nikita Ivkin, Jaime Campos Salas, Kyle R. Ulrich, Xiang Song, Ashish Khetan, and George Karypis. XShare: Collaborative in-Batch Expert Sharing for Faster MoE Inference. In *Resource-Adaptive Foundation Model Inference (AdaptFM), ICML 2026 Workshop*, 2026. URL https://openreview.net/forum?id=KmY5XUgEYe.

Xinming Wei, Chao Jin, Tuo Dai, Yinmin Zhong, Shan Yu, Chengxu Yang, Bingyang Wu, Zili Zhang, Jing Mai, Qianchao Zhu, Zhouyang Li, Yuliang Liu, and Guojie Luo. UltraEP: Unleash MoE Training and Inference on Rack-Scale Nodes with Near-Optimal Load Balancing. arXiv preprint arXiv:2606.04101, 2026. URL https://arxiv.org/abs/2606.04101.

Juntong Wu, Jialiang Cheng, Fuyu Lv, Ou Dan, and Li Yuan. SERE: Similarity-based Expert Re-routing for Efficient Batch Decoding in MoE Models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/5d07eb2335d88b1dfcefcc3d2da2a230-Abstract-Conference.html.

Jaehoon Yang, Yushin Kim, Seokwon Moon, Yeonhong Park, and Jae W. Lee. Libra: Effective yet Efficient Load Balancing for Large-scale MoE Inference. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=WhxNwgGkAS.

Tongtian Yue, Longteng Guo, Jie Cheng, Xuange Gao, Hua Huang, and Jing Liu. Ada-K routing: Boosting the efficiency of MoE-based LLMs. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/df22a19686a558e74f038e6277a51f68-Abstract-Conference.html.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Adrian Zhao, Zhenkun Cai, Zhenyu Song, Lingfan Yu, Haozheng Fan, Jun Wu, Yida Wang, and Nandita Vijaykumar. CRAFT: Fine-Grained Cost-Aware Expert Replication For Efficient Mixture-of-Experts Serving. In *Proceedings of Machine Learning and Systems*, volume 8, pp. 689–700, 2026. URL https://proceedings.mlsys.org/paper_files/paper/2026/hash/3a7f9e485845dac27423375c934cb4db-Abstract-Conference.html.

Chenqi Zhao, Wenfei Wu, Linhai Song, Yuchen Xu, and Yitao Yuan. Fine-grained MoE Load Balancing with Linear Programming. arXiv preprint arXiv:2511.16947, 2025. URL https://arxiv.org/abs/2511.16947.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. WildChat: 1M ChatGPT Interaction Logs in the Wild. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/9421261e06f1a63a352b068f1ac90609-Abstract-Conference.html.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Zhifeng Chen, Quoc V Le, and James Laudon. Mixture-of-Experts with Expert Choice Routing. In *Advances in Neural Information Processing Systems*, volume 35, pp. 7103–7114, 2022. doi: 10.52202/068431-0515. URL https://papers.nips.cc/paper_files/paper/2022/hash/2f00ecd787b432c1d36f3de9800728eb-Abstract-Conference.html.

A THEORY: SETTING AND PROOFS

A.1 SETTING, NOTATION AND EXACT PRICING

Routes and patterns. We use the notation of Section 2.1, with $E = NL$, $1 \le M \le N$ and $1 \le K \le ML$. The legal routes $\mathcal{F}$ and the best score $H(p)$ are those of (1), and the legal patterns $\mathcal{O}$ and their values $v(o)$ those of (2), with $S_g(0) = 0$. The pattern of a route z is $o(z)$, with $o_g(z) = \sum_{e \in G_g} z_e$. Groups and experts carry fixed indices, and every tie goes to the lower index. Lemmas A.1–A.4 adapt known tools to our LP: the reduction to patterns of the cutting-stock problem (Gilmore & Gomory, 1961), LP duality and column generation (Schrijver, 2003; Dantzig & Wolfe, 1960; Gilmore & Gomory, 1961), the dynamic programme of Baldassarre et al. (2016, Theorem 2) and the extreme-point argument of iterative relaxation (Király et al., 2012). Part (v) of Lemma A.2 is the exact-penalty argument for slack variables (Han & Mangasarian, 1979), proved here for completeness.

Lemma A.1 (Pattern reduction)**.** *Let $p \in \mathbb{R}^E$ and $o \in \mathcal{O}$. The best score $p^\top z$ of a legal route z with pattern o is $v(o)$, and the route that takes the o_g highest-scoring experts of every group g attains it. Hence $H(p) = \max_{o \in \mathcal{O}} v(o)$.*

Proof. A route with pattern o selects o_g distinct experts of G_g for every g. Their scores sum to at most $S_g(o_g)$, which gives $p^\top z \le v(o)$. The route that takes the o_g highest-scoring experts of every group has pattern o and score $v(o)$. It selects $\mathbf{1}^\top o = K$ experts from the at most M groups with $o_g > 0$ and is therefore legal. The pattern of a legal route has integer entries in $[0, L]$ that sum to K, at most M of them positive, and therefore lies in $\mathcal{O}$. Taking the maximum over $o \in \mathcal{O}$ gives $H(p) = \max_{o \in \mathcal{O}} v(o)$. □

The released rule. Let $K = Mq$ with $q \in \{1, \dots, L\}$. The *group score* of g is $\pi_g(p) = S_g(q)$. The released rule keeps the set $A(p)$ of M groups with the largest group scores. It returns the route $n(p)$ formed by the K highest-scoring experts in the groups of $A(p)$. Because the kept groups hold $ML \ge K$ experts, $n(p)$ is legal. Its score is $P(p) = p^\top n(p)$.

Example (Figure 1b). Let $(N, L, M, K) = (3, 4, 2, 4)$, with $q = 2$, and let the scores in groups A, B and C be $(1.00, 0.10, 0.05, 0.02)$, $(0.60, 0.55, 0.50, 0.10)$ and $(0.58, 0.56, 0.30, 0.05)$. The group scores are 1.10, 1.15 and 1.14. The released rule keeps B and C and takes 0.60, 0.58, 0.56 and 0.55, with $P(p) = 2.29$. The pattern $(1, 3, 0)$ scores $1.00 + 0.60 + 0.55 + 0.50 = 2.65 = H(p)$. If each expert taken from group B pays 0.2, the pattern $(1, 0, 3)$ has the largest value after the prices, 2.44.

Batches, prices and the LP. A batch has tokens $t = 1, \dots, B$ with scores $a_t \in \mathbb{R}^E$, and S_{tg} and v_t are S_g and v computed from a_t. A routing (z_t) has score $\sum_t a_t^\top z_t$ and loads $Q_g = \sum_t o_g(z_t)$. For weights $w = (w_{to})_{t \le B,\, o \in \mathcal{O}}$ put $\bar{Q}(w) = \sum_{t,o} w_{to}\, o$. The LP (3) maximises $\sum_{t,o} w_{to} v_t(o)$ over the set $\mathcal{W}$ of weights $w \ge 0$ with $\sum_o w_{to} = 1$ for every t and $\ell_g \le \bar{Q}_g(w) \le h_g$ for every g, and U is its value. A point $w \in \mathcal{W}$ *splits* token t if $w_{to} > 0$ for two or more patterns o. At prices $\mu \in \mathbb{R}^N$, the *priced scores* $a_t - \mu$ of token t are its scores with μ_g subtracted from every score in G_g. The dual function of Section 4 is $D_G(\mu) = \sum_t H_t(\mu) + \sum_g (h_g \mu_g^+ - \ell_g \mu_g^-)$, where $H_t(\mu) = \max_{o \in \mathcal{O}} [v_t(o) - \mu^\top o]$ and $x^\pm = \max\{\pm x, 0\}$ for real x. Subtracting μ_g keeps the order of the scores inside G_g and lowers the sum of any j of them by $j\mu_g$. By Lemma A.1 applied to $a_t - \mu$, the best priced score of a legal route with pattern o is $v_t(o) - \mu^\top o$, and $H_t(\mu) = H(a_t - \mu)$. Exact selection at prices μ gives token t a pattern $o_t(\mu)$ that attains $H_t(\mu)$ and the route that takes the $o_{tg}(\mu)$ highest-scoring experts of every group g. Its loads are $Q(\mu) = \sum_t o_t(\mu)$.

Lemma A.2 (LP duality)**.** *Let every target interval contain τ. (i) $\mathcal{W}$ is a nonempty polytope, and the LP* (3) *has an optimal solution at a vertex of $\mathcal{W}$ (a* basic optimum*).*

(ii) $\sum_{t,o} w_{to} v_t(o) \le D_G(\mu)$ for all $w \in \mathcal{W}$ and $\mu \in \mathbb{R}^N$.

(iii) D_G is convex and piecewise linear, and $\min_\mu D_G(\mu) = U$, with the minimum attained.

(iv) Fix $w_{to} = 0$ for all pairs (t, o) outside a given set of pairs, the pool, and assume the restricted LP is feasible, with value U'. Let (y', α', β') be an optimal solution of its dual, where y'_t, $\alpha'_g \ge 0$

and $\beta'_g \geq 0$ are the multipliers of the rows $\sum_o w_{to} = 1$, $\bar{Q}_g \leq h_g$ and $\bar{Q}_g \geq \ell_g$. Put $\mu' = \alpha' - \beta'$. If $H_t(\mu') \leq y'_t$ for every t, or $D_G(\mu) \leq U'$ for some $\mu \in \mathbb{R}^N$, then $U' = U$, and every basic optimum of the restricted LP is a basic optimum of the LP (3).

(v) Let the pool hold a pattern of every token, and give the restricted LP of (iv) slack variables $s^+, s^- \in \mathbb{R}^N_{\geq 0}$, with the rows $\bar{Q}_g(w) - s^+_g \leq h_g$ and $\bar{Q}_g(w) + s^-_g \geq \ell_g$ and the objective $\sum_{t,o} w_{to} v_t(o) - C\,\mathbf{1}^\top(s^+ + s^-)$ for a cost $C > 0$. This slack LP has an optimal solution. If one of its basic optima has zero slacks, then the restricted LP of (iv) is feasible and has the same value, this optimum without its slacks is a basic optimum of it, and every optimal dual solution of the slack LP is an optimal dual solution of it. If $C > \max_g |\mu^*_g|$ for some minimiser μ^* of D_G, and the value and an optimal dual solution of the slack LP meet one of the conditions of (iv), then every optimum of the slack LP has zero slacks.

Proof. (i) Fix $o \in \mathcal{O}$ and let $o^{(k)}$, $k = 0, \ldots, N-1$, be its cyclic shifts, $o^{(k)}_g = o_{g+k}$ with indices modulo N. They lie in $\mathcal{O}$, and $\frac{1}{N}\sum_k o^{(k)} = \frac{K}{N}\mathbf{1}$. Giving every token weight $1/N$ on each shift (adding the weights of equal shifts) yields a point w with $\bar{Q}(w) = \tau\mathbf{1}$, which lies in $\mathcal{W}$ because every interval contains τ. The set $\mathcal{W}$ is defined by finitely many linear constraints and lies in $[0,1]^{B\times\mathcal{O}}$, and it is therefore a nonempty polytope. A linear function attains its maximum over a nonempty polytope at a vertex (Schrijver, 2003, Section 5.6).

(ii) Let $w \in \mathcal{W}$. Because $w \geq 0$ and $\sum_o w_{to} = 1$, $\sum_o w_{to}[v_t(o) - \mu^\top o] \leq H_t(\mu)$ for every t. Because $\ell_g \leq \bar{Q}_g(w) \leq h_g$, $\mu_g \bar{Q}_g(w) \leq h_g\mu^+_g - \ell_g\mu^-_g$. Summing the first bound over t and the second over g gives $\sum_{t,o} w_{to} v_t(o) = \sum_{t,o} w_{to}[v_t(o) - \mu^\top o] + \mu^\top \bar{Q}(w) \leq D_G(\mu)$.

(iii) Each H_t is a maximum of finitely many affine functions of μ, as is $\mu_g \mapsto h_g\mu^+_g - \ell_g\mu^-_g = \max\{h_g\mu_g, \ell_g\mu_g\}$ because $h_g \geq \ell_g$. Hence D_G is convex and piecewise linear. The dual of the LP (3) is

$$\min\Big\{\sum_t y_t + \sum_g (h_g\alpha_g - \ell_g\beta_g) :\ \alpha, \beta \geq 0,\ \ y_t \geq v_t(o) - (\alpha - \beta)^\top o\ \ \forall t,\ \forall o \in \mathcal{O}\Big\}.$$

By (i) and the duality theorem of linear programming (Schrijver, 2003, Theorem 5.4), it has an optimal solution (y^*, α^*, β^*) of value U. Put $\mu^* = \alpha^* - \beta^*$. Because $\alpha^*_g - (\mu^*_g)^+ = \beta^*_g - (\mu^*_g)^- = \min\{\alpha^*_g, \beta^*_g\} \geq 0$ and $h_g \geq \ell_g$, we get $h_g\alpha^*_g - \ell_g\beta^*_g \geq h_g(\mu^*_g)^+ - \ell_g(\mu^*_g)^-$. The constraints give $y^*_t \geq H_t(\mu^*)$. Summing both bounds over t and g gives $D_G(\mu^*) \leq U$. By (i) and (ii), $U \leq D_G(\mu)$ for every μ. Hence $\min_\mu D_G(\mu) = D_G(\mu^*) = U$.

(iv) Every feasible point of the restricted LP lies in $\mathcal{W}$, and therefore $U' \leq U$. The first condition says that $y'_t \geq v_t(o) - \mu'^\top o$ for all t and $o \in \mathcal{O}$. For $w \in \mathcal{W}$, multiplying by w_{to} and summing gives $\sum_{t,o} w_{to} v_t(o) \leq \sum_t y'_t + \alpha'^\top \bar{Q}(w) - \beta'^\top \bar{Q}(w) \leq \sum_t y'_t + h^\top\alpha' - \ell^\top\beta' = U'$, where the last equality is the duality theorem for the restricted LP. Under the second condition, (ii) gives $\sum_{t,o} w_{to} v_t(o) \leq D_G(\mu) \leq U'$ for every $w \in \mathcal{W}$. In both cases $U \leq U'$, and therefore $U = U'$. Fixing $w_{to} = 0$ outside the pool selects a face of $\mathcal{W}$, and every vertex of this face is a vertex of $\mathcal{W}$. A basic optimum of the restricted LP is therefore a vertex of $\mathcal{W}$ with value U.

(v) One pattern of the pool for every token, with $s^+_g = \max\{\bar{Q}_g(w) - h_g, 0\}$ and $s^-_g = \max\{\ell_g - \bar{Q}_g(w), 0\}$, is a feasible point. The objective is at most $\sum_t \max_o v_t(o)$, and the slack LP therefore has a basic optimal solution and an optimal dual solution. Its dual is the dual of the restricted LP of (iv) with the additional bounds $\alpha_g \leq C$ and $\beta_g \leq C$, which come from the slack columns, and the two duals have the same objective $\sum_t y_t + h^\top\alpha - \ell^\top\beta$. The feasible points with zero slacks form a face of the feasible set of the slack LP, and this face is the feasible set of the restricted LP of (iv). If a basic optimum has zero slacks, it is therefore a vertex of this face and an optimum of the restricted LP of (iv), whose value is at most that of the slack LP. The two values are then equal, and an optimal dual solution of the slack LP is feasible for the dual of the restricted LP with this value and is therefore optimal. Now let $C > m = \max_g |\mu^*_g|$ and take a feasible point (w, s^+, s^-) of the slack LP. Then $\sum_{t,o} w_{to} v_t(o) = \sum_{t,o} w_{to}[v_t(o) - \mu^{*\top} o] + \mu^{*\top}\bar{Q}(w) \leq \sum_t H_t(\mu^*) + \mu^{*\top}\bar{Q}(w)$. Because $\ell_g - s^-_g \leq \bar{Q}_g(w) \leq h_g + s^+_g$, we have $\mu^*_g\bar{Q}_g(w) \leq h_g(\mu^*_g)^+ - \ell_g(\mu^*_g)^- + m\,(s^+_g + s^-_g)$. By (iii), the objective at this point is therefore at most $D_G(\mu^*) - (C-m)\mathbf{1}^\top(s^+ + s^-) = U - (C - m)\mathbf{1}^\top(s^+ + s^-)$. Let U' be the value of the slack LP. Under the first condition of (iv), the argument of

(iv) applies unchanged, because the dual objective is the same, and gives $U \le U'$; under the second, (ii) gives $U \le D_G(\mu) \le U'$. Every optimum therefore has $(C-m)\mathbf{1}^\top(s^+ + s^-) \le U - U' \le 0$, and its slacks are zero. □

The first condition in part (iv) is the stopping rule of column generation (Dantzig & Wolfe, 1960; Gilmore & Gomory, 1961). Column generation stops after finitely many rounds: by dual feasibility, every pattern o of token t in the pool has $v_t(o) - \mu'^\top o \le y'_t$, every round that does not stop therefore adds a pattern that is not yet in the pool, and $\mathcal{O}$ is finite. The next lemma computes each $H_t(\mu')$ exactly.

Lemma A.3 (Exact pricing). *Fix a token with scores p and prices μ. Put $J = \min\{K, L\}$ and $c_g(j) = S_g(j) - j\mu_g$ for $1 \le g \le N$ and $0 \le j \le J$. Let $F_0(m,0) = 0$ and $F_0(m,k) = -\infty$ for $k \ge 1$. For $g = 1, \dots, N$, $0 \le m \le M$ and $0 \le k \le K$ let*

$$F_g(m,k) = \max\Big\{F_{g-1}(m,k),\ \max_{1 \le j \le \min\{k,J\}} \big[F_{g-1}(m-1, k-j) + c_g(j)\big]\Big\}, \tag{5}$$

where the inner maximum is omitted if $m = 0$ or $k = 0$. Then $F_N(M,K) = \max_{o \in \mathcal{O}} \big[v(o) - \mu^\top o\big]$, and recording a maximising j at every state yields a maximising pattern. Given the scores of every group in decreasing order, the computation takes $O(NMK \min\{K,L\})$ operations.

Proof. For $0 \le g \le N$ let $\mathcal{O}_g(m,k)$ be the set of tuples $(o_1, \dots, o_g) \in \{0, \dots, J\}^g$ with $\sum_h o_h = k$ and at most m positive entries. We show by induction on g that $F_g(m,k)$ is the largest value of $\sum_{h \le g} c_h(o_h)$ over $\mathcal{O}_g(m,k)$, with $\max \emptyset = -\infty$. For $g = 0$ the only tuple is the empty one, which has $k = 0$ and value 0. For $g \ge 1$, a tuple of $\mathcal{O}_g(m,k)$ with $o_g = 0$ is a tuple of $\mathcal{O}_{g-1}(m,k)$ followed by 0. A tuple with $o_g = j \ge 1$ is a tuple of $\mathcal{O}_{g-1}(m-1, k-j)$ followed by j, which requires $m \ge 1$ and $j \le \min\{k, J\}$. Conversely, each such extension lies in $\mathcal{O}_g(m,k)$. This gives (5), because $c_g(0) = 0$. Every $o \in \mathcal{O}$ has $o_g \le \min\{K,L\} = J$, and therefore $\mathcal{O} = \mathcal{O}_N(M,K)$ and $\sum_g c_g(o_g) = v(o) - \mu^\top o$. The prefix sums $S_g(j)$, $j \le J$, cost $O(NJ)$ operations. There are $N(M+1)(K+1)$ states, and each costs $O(J)$ operations. □

Because prices keep the order of the scores inside each group, each token's scores are sorted once, in $O(E \log L)$ operations, and reused at every price vector. Lemma A.3 adapts the dynamic programme of Baldassarre et al. (2016, Theorem 2), which selects at most a given number of entries with nonnegative weights from a bounded number of groups, to exactly K entries and to priced scores of any sign.

A.2 Proof of Theorem 4.1

For $1 < M < N$ put $a = q - 1$, $b = L - q$, $\bar{r} = \min\{N - M, M - 1\} \ge 1$ and

$$D = \max_{1 \le r \le \bar{r}} \max\big\{\min\{ra, (M-r)b\},\ \min\{rb, (M-r)a\}\big\} \ge 0. \tag{6}$$

Theorem 4.1 (restated). *Let $K = Mq$ with an integer $q \in \{1, \dots, L\}$. (a) $P(p) = H(p)$ for all $p \in \mathbb{R}^E$ if and only if $M = 1$, $M = N$, $q = 1$ or $q = L$; at every other shape, including the deployed one, some p has $P(p) < H(p)$. (b) If $1 < M < N$ and $q > 1$, then $P(p) \ge \rho H(p)$ for all $p \in [0, \infty)^E$, with $\rho = Mq/(Mq + D)$ and D as in* (6)*. With ties to the lower index, some such p with $H(p) > 0$ has $P(p) = \rho H(p)$; among scores with distinct entries, $P(p)/H(p)$ comes arbitrarily close to ρ. Also $\rho \ge 1/(2 - 1/M)$, and $\rho = 8/11$ at the deployed shape. Part (a) and the bound $P(p) \ge \rho H(p)$ hold for every order of breaking ties.*

Proof of (a). Because $n(p)$ is legal, $P \le H$. We show $P \ge H$ in the four cases, and then give scores with $P < H$ at every other shape.

Case $M = 1$. Then $K = q$. A legal route lies in one group, and the best route inside G_g scores $S_g(K)$. Hence $H = \max_g S_g(K)$. The released rule keeps a group with the largest $S_g(q) = S_g(K)$ and takes its K highest scores, which gives $P = H$.

Case $M = N$. Every group is kept, and $n(p)$ takes the K highest-scoring experts of all groups. No K experts score more, and $P = H$.

Case $q = L$. Then $K = ML$, and a legal route consists of all experts of M groups. Hence H is the largest value of $\sum_{g\in A'} S_g(L)$ over sets A' of M groups. The released rule keeps M groups with the largest $\pi_g = S_g(L)$ and takes all their experts. Again $P = H$.

Case $q = 1$. Then $K = M$, and any K experts lie in at most M groups. Every set of K experts is therefore a legal route, and H is the sum of the K largest scores. Let s be the K-th largest score, X the set of experts scoring more than s, and m the number of groups that meet X. Then $m \le |X| \le K-1$. Because $q = 1$, the group score is the group maximum. The m groups that meet X have maxima above s, and every other group has maximum at most s. Because $m < M$, these m groups are kept. If each of the other $M - m$ kept groups contains an expert scoring s, the kept groups contain at least $|X| + M - m \ge K$ experts scoring at least s. Otherwise some kept group has maximum below s. Then every group whose maximum is at least s ranks above that group and is therefore kept, and the kept groups contain all experts scoring at least s, which number at least K. In both cases the kept groups contain X, at least $K - |X|$ of the other kept experts score exactly s, and the rest score at most s. Hence $P = \sum_{e\in X} p_e + (K - |X|)s = H$.

The other shapes. Let $1 < M < N$ and $1 < q < L$, and pick $M + 1$ distinct groups $g_1, \dots, g_{M-1}, g_B, g_C$. Define p by the scores 3 on $q + 1$ experts of g_1 and on q experts of each of $g_2, \dots, g_{M-1}$ (possible because $q+1 \le L$); 2 on $q-1$ experts of g_B; $2-1/q$ on q experts of g_C; and 0 elsewhere. There are $(q+1)+q(M-2) = K-q+1$ scores equal to 3. The group scores are $3q$ for $g_1, \dots, g_{M-1}$, $2q-1$ for g_C, $2q-2 > 0$ for g_B and 0 for all other groups. The released rule therefore keeps $g_1, \dots, g_{M-1}, g_C$. Its K highest kept scores are the $K-q+1$ threes and $q-1$ scores $2-1/q$ of g_C, and $P(p) = 3(K-q+1)+(q-1)(2-1/q)$. The $K-q+1$ threes and the $q-1$ twos of g_B form a route with K experts in M groups. It is legal and scores $P(p) + (q-1)/q > P(p)$. The deployed shape has $1 < M = 4 < N = 8$ and $1 < q = 2 < L = 32$. □

Proof of (b). Fix $p \in [0,\infty)^E$ and write $A = A(p)$. Because $M < N$, some group is not kept. Let $\bar\pi = \max_{g\notin A} \pi_g(p)$. Every kept group has $\pi_g \ge \bar\pi$.

Step 1: two facts about P. Together, the q highest-scoring experts of the kept groups are K kept experts with total score $\sum_{g\in A} \pi_g \ge M\bar\pi$. Hence

$$P(p) \ge M\bar\pi. \tag{7}$$

Let $\lambda \ge 0$ be the K-th largest score among the $ML \ge K$ experts of the kept groups. The K highest kept scores are at least λ, and the other kept scores are at most λ. Hence

$$P(p) = K\lambda + \sum_{g\in A}\sum_{e\in G_g} (p_e - \lambda)^+. \tag{8}$$

Step 2: a bound through one pattern. By Lemma A.1 some $o \in \mathcal{O}$ has $v(o) = H(p)$. Let m_A count the kept groups with $o_g > 0$, and put $k_A = \sum_{g\in A} o_g$, $c = K - k_A$ and $d = \sum_{g\notin A,\, o_g>0} (q - o_g)^+$. We bound the groups one by one.

(i) An excluded group with $o_g = k \ge 1$ has $S_g(k) \le \frac{\bar\pi}{q}\big(k + (q-k)^+\big)$. If $k \le q$, then $S_g(k) \le S_g(q) = \pi_g \le \bar\pi$, because scores are nonnegative. If $k > q$, each of the scores ranked $q+1, \dots, k$ is at most the q-th largest score, which is at most π_g/q. Hence $S_g(k) \le \pi_g + (k-q)\pi_g/q = k\pi_g/q \le k\bar\pi/q$.

(ii) A kept group with $o_g = k \ge 1$ has $S_g(k) - \sum_{e\in G_g}(p_e - \lambda)^+ \le k\lambda$, because each of its k highest scores is at most $\lambda + (p_e - \lambda)^+$ and every term $(p_e - \lambda)^+$ is nonnegative.

(iii) A kept group with $o_g = 0$ has $\sum_{e\in G_g}(p_e - \lambda)^+ \ge (\pi_g - q\lambda)^+ \ge (\bar\pi - q\lambda)^+$. For the first inequality, restrict the sum to the q highest scores and use $\sum_i x_i^+ \ge (\sum_i x_i)^+$.

By (8),

$$H(p) - P(p) = \sum_{g\notin A} S_g(o_g) + \sum_{g\in A}\Big[S_g(o_g) - \sum_{e\in G_g}(p_e - \lambda)^+\Big] - K\lambda.$$

The excluded groups contribute at most $\frac{\bar{\pi}}{q}(c+d)$ by (i). The kept groups contribute at most $k_A\lambda - (M-m_A)(\bar{\pi}-q\lambda)^+$ by (ii) and (iii). Writing $K\lambda = c\lambda + k_A\lambda$ gives

$$H(p) - P(p) \le c\Big(\frac{\bar{\pi}}{q} - \lambda\Big) + \frac{d\bar{\pi}}{q} - (M-m_A)(\bar{\pi}-q\lambda)^+. \tag{9}$$

If $\lambda \ge \bar{\pi}/q$, the right side is at most $d\bar{\pi}/q$, because $c \ge 0$. On $[0, \bar{\pi}/q]$ it is affine in λ. Its value at $\lambda = 0$ is $\bar{d}\bar{\pi}/q$, where $\bar{d} = c + d - q(M-m_A)$, and its value at $\lambda = \bar{\pi}/q$ is $d\bar{\pi}/q$. Because $\lambda \ge 0$, in all cases

$$H(p) - P(p) \le \frac{\bar{\pi}}{q}\max\{d, \bar{d}\}. \tag{10}$$

Step 3: $d \le D$ and $\bar{d} \le D$. Let r count the excluded groups with $0 < o_g < q$. Only these groups contribute to d, each at most $q-1 = a$. This gives $d \le ra$, and $d = 0 \le D$ if $r = 0$. These groups hold $rq - d$ experts of o. The other $K - rq + d$ experts lie in at most $M - r$ groups of L experts, which gives $K - rq + d \le (M-r)L$, that is, $d \le (M-r)b$. Also $r \le N - M$, and $r \le M - 1$ because M such groups would hold fewer than $Mq = K$ experts. Hence, if $r \ge 1$, then $d \le \min\{ra, (M-r)b\} \le D$.

For $\bar{d}$, let m_X count the excluded groups with $o_g > 0$. Because $c - q(M-m_A) = qm_A - k_A = \sum_{g\in A,\, o_g>0}(q-o_g)$,

$$\bar{d} = \sum_{g\in A,\, o_g>0}(q-o_g) + \sum_{g\notin A,\, o_g>0}(q-o_g)^+. \tag{11}$$

Because $\sum_{g:\, o_g>0}(q-o_g) = q(m_A+m_X) - K = -q(M-m_A-m_X)$, this equals

$$\bar{d} = \sum_{g\notin A,\, o_g>q}(o_g - q) - q(M-m_A-m_X). \tag{12}$$

Now let r count the excluded groups with $o_g > q$. Because $m_A + m_X \le M$ and $o_g \le L$, (12) gives $\bar{d} \le rb$. In particular, $\bar{d} \le 0 \le D$ if $r = 0$. In (11) these r groups contribute 0, and each of the at most $M - r$ other groups with $o_g > 0$ contributes at most $q - 1 = a$. This gives $\bar{d} \le (M-r)a$. Also $r \le N - M$, and $r \le M - 1$ because M such groups would hold more than K experts. Hence, if $r \ge 1$, then $\bar{d} \le \min\{rb, (M-r)a\} \le D$.

Step 4: the bound. By (10), Step 3 and (7), $H(p) - P(p) \le \bar{\pi}D/q \le P(p)D/(Mq)$. This rearranges to $(Mq + D)P(p) \ge MqH(p)$, that is, $P(p) \ge \rho H(p)$.

Step 5: attainment. Let $r \in \{1, \dots, \bar{r}\}$ attain the maximum in (6). Let G' be the M groups of lowest index and G'' a set of r other groups, which exists because $r \le N - M$. All scores not specified below are 0.

If $D = \min\{ra, (M-r)b\}$, every expert of G' scores $1/q$, and each group of G'' has one expert scoring 1. Every group of $G' \cup G''$ has group score 1, and every other group has group score 0. With ties to the lower index, the released rule keeps G', and $P = K/q = M$. Consider the route formed by the r experts scoring 1, by $u = \min\{(M-r)L, K-r\}$ experts scoring $1/q$ from $M - r$ groups of G', and by $K - r - u$ experts scoring 0 from G''. It uses M groups, and $K - r - u \le r(L-1)$ because $K \le ML$. The route is therefore legal. Because $(M-r)L = (M-r)q + (M-r)b$ and $K - r = (M-r)q + ra$, we get $u = (M-r)q + D$, and the route scores $r + u/q = M + D/q$.

If $D = \min\{rb, (M-r)a\}$, each group of G' has one expert scoring 1, and every expert of G'' scores $1/q$. Again every group of $G' \cup G''$ has group score 1, the released rule keeps G', and $P = M$. Consider the route formed by $M - r$ experts scoring 1 from $M - r$ groups of G', by $u' = \min\{rL, K - M + r\}$ experts of G'', and by $K - M + r - u'$ experts scoring 0 from those $M - r$ groups of G'. It uses M groups, and $K - M + r - u' \le (M-r)(L-1)$ because $K \le ML$; the route is therefore legal. Because $rL = rq + rb$ and $K - M + r = rq + (M-r)a$, we get $u' = rq + D$, and the route scores $M - r + u'/q = M + D/q$.

In both cases $H(p) \ge M + D/q = P(p)/\rho$, and Step 4 gives $P(p) = \rho H(p)$. These vectors tie the group scores of G' and G'', and the tie order decides for G'. With distinct scores, the ratio ρ is approached arbitrarily closely. Let p be one of the two vectors, let $\zeta > 0$, and let $\mathbf{1}_{G'} \in \{0,1\}^E$ indicate the experts of G'. Put $p^\zeta = p + 2\zeta\mathbf{1}_{G'} + \xi$, with $0 \le \xi_e < \zeta$ such that all entries of p^ζ are distinct (each ξ_e avoids finitely many values). The groups of G' have group scores at least $1 + 2q\zeta$

and all other groups less than $1+q\zeta$, and the released rule therefore keeps G'. Every kept score rises by less than 3ζ, which gives $P(p^\zeta) < M + 3K\zeta$. No score falls, and $H(p^\zeta) \geq H(p) \geq M + D/q$. Hence $P(p^\zeta)/H(p^\zeta) < \rho + 3K\zeta/(M + D/q)$.

Step 6: the constants. In (6), $\min\{ra, (M-r)b\} \leq ra \leq (M-1)a$ and $\min\{rb, (M-r)a\} \leq (M-r)a \leq (M-1)a$. Hence $D \leq (M-1)(q-1)$, and

$$\rho \;\geq\; \frac{Mq}{Mq + (M-1)(q-1)} \;\geq\; \frac{Mq}{(2M-1)q} \;=\; \frac{1}{2-1/M}.$$

At the deployed shape $(N, L, M, K) = (8, 32, 4, 8)$ we have $q = 2$, $a = 1$, $b = 30$ and $\bar{r} = 3$. The inner maxima in (6) are 3, 2 and 3 for $r = 1, 2, 3$. Hence $D = 3$ and $\rho = 8/11$. By Step 5, the released rule loses a fraction $3/11$ of H when four kept groups have all scores $1/2$ and three further groups have one score 1: then $P = 4$ and $H = 11/2$. □

A.3 Proof of Theorem 4.2

Theorem 4.2 (restated). *Let every target interval contain τ. (a) With exact selection and a step $\eta > 0$, the update $\mu \leftarrow \Pi\big[\mu + \eta\,(Q(\mu) - \mathrm{target}(\mu))\big]$ is projected subgradient descent on D_G, where Π projects onto a closed convex set containing the initial prices and a minimiser of D_G, and $\mathrm{target}_g(\mu)$ is h_g, ℓ_g or any point of $[\ell_g, h_g]$ when μ_g is positive, negative or zero, respectively. (b) If $K = Mq$ with integers $1 < M < N$ and $1 < q < L$, the shapes outside the four exact cases, then for every batch size B that is a multiple of $2N$ there are batches with scores in $[0, 1]$ and $U = BK$ on which, for every $\mu \in \mathbb{R}^N$, the routing $(n(a_t - \mu))_t$ that the released rule picks at the prices μ, with any order of breaking ties, scores at most $U - B(q-1)/(4q) \leq U - B/8$; at the deployed shape, $B/8 = U/64$.*

Proof of (a). By Lemma A.2(iii), D_G is convex and $\min_\mu D_G(\mu) = U$. Let $o_t = o_t(\mu)$ attain $H_t(\mu)$. For every $\nu \in \mathbb{R}^N$,

$$H_t(\nu) \;\geq\; v_t(o_t) - \nu^\top o_t \;=\; H_t(\mu) - o_t^\top(\nu - \mu);$$

the vector $-o_t$ is therefore a subgradient of H_t at μ. The convex function $\mu_g \mapsto h_g\mu_g^+ - \ell_g\mu_g^-$ has derivative h_g where $\mu_g > 0$ and ℓ_g where $\mu_g < 0$, and its subdifferential at 0 is $[\ell_g, h_g]$. Hence $\mathrm{target}(\mu)$ is a subgradient of $\mu \mapsto \sum_g (h_g\mu_g^+ - \ell_g\mu_g^-)$ at μ. Adding the subgradient inequalities shows that $\mathrm{target}(\mu) - Q(\mu)$ is a subgradient of D_G at μ. The update equals $\mu \leftarrow \Pi\big[\mu - \eta\,(\mathrm{target}(\mu) - Q(\mu))\big]$, a step against a subgradient of D_G followed by the projection Π. This is projected subgradient descent on D_G, the classical dual subgradient method (see Nedić & Ozdaglar, 2009). □

Proof of (b). Put $\theta = 1 - \frac{1}{2q}$ and $\delta = \frac{q-1}{2q}$, and index the groups by $\mathbb{Z}_N$. For each $j \in \mathbb{Z}_N$ there are two tokens t_j^+ and t_j^-. Each scores 1 on $q+1$ experts of group j and on q experts of each of the groups $j+1, \ldots, j+M-2$. These $K - q + 1$ experts are its *mandatory* experts, and these $M-1$ groups its mandatory groups. In the groups $j+M-1$ and $j+M$ it has a *unit block* of $q-1$ experts scoring 1 and a *θ-block* of q experts scoring θ. Token t_j^+ has its unit block in group $j+M-1$ and its θ-block in group $j+M$; token t_j^- has them the other way round. All other scores of both tokens are 0. The groups $j, \ldots, j+M$ are distinct because $M < N$, and the blocks fit because $q+1 \leq L$. The batch consists of m copies of these $2N$ tokens, which gives $B = 2mN$ and $\tau = 2mK$. Let $z_j^\pm$ be the route of $t_j^\pm$ formed by its mandatory experts and unit block; it has K experts in M groups.

Step 1: every legal route other than $z_j^\pm$ scores at most $K - \delta$ for $t_j^\pm$. Consider t_j^+; the case of t_j^- is symmetric. Let g_u and g_θ be the groups of its unit block and its θ-block, and let $z \neq z_j^+$ be legal. The token has exactly K scores equal to 1, those of z_j^+; its other scores are θ or 0. Suppose first that z uses every mandatory group, and therefore at most one further group. If that group is g_u, or there is none, the only positive scores available to z are the 1s of z_j^+; because $z \neq z_j^+$, z contains a 0 and scores at most $K - 1$. If that group is g_θ and z contains all mandatory experts, its other $q-1$ experts score at most θ, and z scores at most $K - q + 1 + (q-1)\theta = K - \delta$. If that group is g_θ and z omits $k \geq 1$ mandatory experts, it contains at most q experts scoring θ and

scores at most $K - q + 1 - k + q\theta = K + \frac{1}{2} - k \le K - \frac{1}{2}$. If that group is any other group, z scores at most $K - q + 1 \le K - 1$. Suppose next that z misses a mandatory group. Then it misses at least q mandatory experts and gains at most the two blocks, and therefore scores at most $(K - 2q + 1) + (q - 1) + q\theta = K - \frac{1}{2}$. Because $\delta < \frac{1}{2} < 1$, every such z scores at most $K - \delta$.

Step 2: $U = BK$. The patterns of z_j^+ and z_j^- add up to $2q + 2$ on group j, to $2q$ on each of the groups $j + 1, \dots, j + M - 2$, and to $q - 1$ on each of the groups $j + M - 1$ and $j + M$. Over all $j \in \mathbb{Z}_N$, every group receives $(2q + 2) + 2q(M - 2) + 2(q - 1) = 2K$ assignments per copy, which gives exactly τ in total. The routing $(z_j^{\pm})$ therefore meets every interval and scores BK. By Lemma A.1 and Step 1, $\max_o v_t(o) = K$ for every token, and every $w \in \mathcal{W}$ has value at most $\sum_t \max_o v_t(o) = BK$. Hence $U = BK$, and exact selection at $\mu = 0$ returns the routing $(z_j^{\pm})$.

Step 3: each pair loses at least δ. Fix μ, an index j and one copy of t_j^+ and t_j^-. At the priced scores, the group score of g is lowered by $q\mu_g$. For both tokens, the group of the unit block has group score $q - 1$ before pricing, because its q-th largest score is 0 ($q < L$). The group of the θ-block has group score $q\theta = q - \frac{1}{2}$. If the released rule gives t_j^+ the route z_j^+, it keeps the M groups of z_j^+. It then keeps group $j + M - 1$ but not group $j + M$, and whatever the tie order,

$$(q - 1) - q\mu_{j+M-1} \;\ge\; \big(q - \tfrac{1}{2}\big) - q\mu_{j+M}.$$

If it gives t_j^- the route z_j^-, the same argument with the two groups exchanged gives $(q - 1) - q\mu_{j+M} \ge (q - \frac{1}{2}) - q\mu_{j+M-1}$. The two inequalities add up to $0 \ge 1$. At least one token of the pair therefore does not receive its route $z_j^{\pm}$, and by Step 1 its route scores at most $K - \delta$.

Step 4: the deficit. There are $mN = B/2$ pairs, and every token scores at most K. The routing picked at prices μ therefore scores at most $BK - \frac{B}{2}\delta = U - \frac{B(q-1)}{4q}$. Because $q \ge 2$, $\frac{q-1}{4q} \ge \frac{1}{8}$. At the deployed shape, $U = 8B$ and $B/8 = U/64$. □

A.4 Proof of Theorem 4.3

HYPERSPACE rounds the basic optimum of the restricted LP at which column generation stops, and it returns a routing only if the slacks of this optimum are zero (Appendix B). This optimum is then a basic optimum of the LP (3) by Lemma A.2(iv) and (v), and a slack cost above $\max_g |\mu_g^*|$ for a minimiser μ^* of D_G makes the slacks zero. For $w \in \mathcal{W}$, let $\Omega_t = \{o : w_{to} > 0\}$ be the support of token t, and $\sigma(w) = \sum_t (|\Omega_t| - 1)$. A *selection* r from w picks $r_t \in \Omega_t$ for every token. Its value is $V(r) = \sum_t v_t(r_t)$, its loads are $T(r) = \sum_t r_t$, and its deviation is $\mathrm{dev}(r) = \max_g \max\{0,\, T_g(r) - h_g,\, \ell_g - T_g(r)\}$. For a basic optimum w, min-deviation rounding returns a selection r^{MD} with $V(r^{\mathrm{MD}}) \ge U$ and the least deviation among such selections. Such selections exist: taking for every token a pattern of largest value in Ω_t gives $V(r) \ge \sum_{t,o} w_{to} v_t(o) = U$. There are $\prod_t |\Omega_t| \le \prod_t 2^{|\Omega_t|-1} = 2^{\sigma(w)}$ selections, at most 2^{N-1} by part (a) below. Enumerating them follows the linear programming heuristic of Potts (1985), which enumerates the assignments of the at most $m - 1$ jobs that a basic solution splits over m machines.

Placement. Write $r = r^{\mathrm{MD}}$ and $T = T(r)$. For every token t, let A_t be the groups with $r_{tg} > 0$, completed to M groups by the other groups g with the largest top scores $S_{tg}(1)$. Placement solves

$$\max_x \sum_{t,e} a_{te} x_{te} \quad \text{s.t.} \quad \sum_e x_{te} = K \;\; \forall t, \qquad \min\{\ell_g, T_g\} \le \sum_t \sum_{e \in G_g} x_{te} \le \max\{h_g, T_g\} \;\; \forall g, \tag{13}$$

over $0 \le x_{te} \le 1$ with $x_{te} = 0$ unless e lies in a group of A_t, and returns an integral optimum. Such an optimum exists. Every variable x_{te} appears in one token row and one group row of (13). The constraint matrix is therefore a bipartite incidence matrix, which makes it totally unimodular, and all bounds are integers. Hence the feasible set, which contains the routing that gives each token t the r_{tg} highest-scoring experts of every group g, has only integral vertices (Schrijver, 2003, Theorems 5.20 and 18.2).

Theorem 4.3 (restated). *Let every target interval have integer ends and contain τ. (a) A basic optimum w of the LP* (3) *has $\sigma(w) \le N - 1$; in particular it splits at most $N - 1$ tokens. (b)*

For a basic optimum w, min-deviation rounding picks one pattern of Ω_t for every token, keeps $V(r^{\mathrm{MD}}) \geq U$ and, whatever the batch size, gives loads $T_g = T_g(r^{\mathrm{MD}})$ that satisfy (4) *for every group g, with $2K - 2 = 14$ at the deployed shape. (c) For any prices μ, $D_G(\mu)$ bounds from above the score of every legal routing that meets the intervals. From a basic optimum w, the routing that HYPERSPACE returns is legal, scores at least $V(r^{\mathrm{MD}}) \geq U = \min_\mu D_G(\mu)$, keeps every group load Q_g between $\min\{\ell_g, T_g\}$ and $\max\{h_g, T_g\}$, and therefore within the bound of (b), and is optimal among the legal routings that meet the intervals whenever it meets them.*

We prove (a) and (b) with the following lemma, which is the extreme-point argument of iterative relaxation (Király et al., 2012).

Lemma A.4 (Independence at a vertex)**.** *For every token let $\mathcal{C}_t \subseteq \mathcal{O}$, and let $R \subseteq \{1, \dots, N\}$. Let $\mathcal{W}'$ be the set of $w \geq 0$ with $w_{to} = 0$ for $o \notin \mathcal{C}_t$, $\sum_o w_{to} = 1$ for every t, and $\ell_g \leq \bar{Q}_g(w) \leq h_g$ for every $g \in R$. Let $\bar{w}$ be a vertex of $\mathcal{W}'$, let $\Omega_t^{\bar{w}} = \{o : \bar{w}_{to} > 0\}$, fix $\bar{o}_t \in \Omega_t^{\bar{w}}$ for every t, and let $I \subseteq R$ be the groups with $\bar{Q}_g(\bar{w}) \in \{\ell_g, h_g\}$. Then the $\sum_t(|\Omega_t^{\bar{w}}| - 1)$ vectors $\bar{o}_t - o$, $o \in \Omega_t^{\bar{w}} \setminus \{\bar{o}_t\}$, restricted to the coordinates in I, are linearly independent.*

Proof. Suppose that $\sum_{t,o} \chi_{to}(\bar{o}_t - o)_g = 0$ for all $g \in I$, where o runs over $\Omega_t^{\bar{w}} \setminus \{\bar{o}_t\}$ and $\chi \neq 0$. Define y by $y_{to} = -\chi_{to}$ for these pairs, $y_{t\bar{o}_t} = \sum_o \chi_{to}$, and $y = 0$ elsewhere. Then $y \neq 0$, y vanishes outside the support of $\bar{w}$, $\sum_o y_{to} = 0$ for every t, and $\bar{Q}(y) = \sum_{t,o} \chi_{to}(\bar{o}_t - o)$ vanishes on I. For small $\zeta > 0$, both $\bar{w} + \zeta y$ and $\bar{w} - \zeta y$ are nonnegative and satisfy the rows $\sum_o w_{to} = 1$. They keep the loads of the groups in I, and keep those of the groups in $R \setminus I$ strictly inside their intervals. Both therefore lie in $\mathcal{W}'$, and $\bar{w}$ is their midpoint, which contradicts that $\bar{w}$ is a vertex. □

Proof of (a). Apply Lemma A.4 to $\bar{w} = w$, with $\mathcal{C}_t = \mathcal{O}$, $R = \{1, \dots, N\}$ and any $\bar{o}_t \in \Omega_t$; then $\mathcal{W}' = \mathcal{W}$ and $\Omega_t^{\bar{w}} = \Omega_t$. The $\sigma(w)$ vectors $\bar{o}_t - o$ are linearly independent, because their restrictions to I are. Each has coordinate sum $K - K = 0$, and they therefore lie in an $(N-1)$-dimensional subspace of $\mathbb{R}^N$. Hence $\sigma(w) \leq N - 1$. Every split token contributes at least one vector, and at most $N - 1$ tokens are therefore split. This is the classical argument for split jobs in assignment-type LPs (Benders & van Nunen, 1983; Potts, 1985; Lenstra et al., 1990), applied to patterns. □

Proof of (b). By definition, r^{MD} picks $r_t^{\mathrm{MD}} \in \Omega_t$ and has $V(r^{\mathrm{MD}}) \geq U$. It remains to exhibit a selection $\hat{r}$ from w with $V(\hat{r}) \geq U$ and $\mathrm{dev}(\hat{r}) \leq 2K - 2$, because then $\mathrm{dev}(r^{\mathrm{MD}}) \leq \mathrm{dev}(\hat{r})$. We construct $\hat{r}$ by iterative relaxation.

The procedure. Start with $\Omega_t^0 = \Omega_t$ and $R_0 = \{1, \dots, N\}$. In round $k = 0, 1, \dots$, let $\mathcal{W}_k$ be the set $\mathcal{W}'$ of Lemma A.4 with $\mathcal{C}_t = \Omega_t^k$ and $R = R_k$. Let w^k be a vertex of $\mathcal{W}_k$ that maximises $\sum_{t,o} w_{to} v_t(o)$, and put $\Omega_t^{k+1} = \{o : w_{to}^k > 0\}$. If every Ω_t^{k+1} has one element, stop. Otherwise, remove from R_k the group g chosen below, giving R_{k+1}.

Value. We have $w \in \mathcal{W}_0$, and $w^k \in \mathcal{W}_{k+1}$ because Ω_t^{k+1} is the support of w_t^k and $R_{k+1} \subseteq R_k$. Every $\mathcal{W}_k$ is therefore a nonempty polytope, the optimal values do not decrease with k, and all are at least $\sum_{t,o} w_{to} v_t(o) = U$.

Choice of g. For a round k that does not stop, write $\bar{w} = w^k$ and $\Omega_t' = \Omega_t^{k+1}$ for its supports. Let $f \geq 1$ be the number of tokens that $\bar{w}$ splits, and $s = \sum_t(|\Omega_t'| - 1)$. Then $f \leq s$, and the supports of the split tokens hold $f + s \leq 2s$ patterns. Fix any $\bar{o}_t \in \Omega_t'$ for every t. By Lemma A.4, the s vectors $\bar{o}_t - o$ of the split tokens, restricted to the groups $I \subseteq R_k$ with tight rows, are linearly independent. Choose $I' \subseteq I$ with $|I'| = s$ such that their restriction to I' is a nonsingular $s \times s$ matrix. For every group put

$$\omega_g = \sum_{t \text{ split}} \Big(\max_{o \in \Omega_t'} o_g - \min_{o \in \Omega_t'} o_g \Big) \in \mathbb{Z}_{\geq 0}.$$

Because patterns are nonnegative, $\max_{o \in \Omega_t'} o_g - \min_{o \in \Omega_t'} o_g \leq \sum_{o \in \Omega_t'} o_g$. Because every pattern has coordinate sum K, this gives $\sum_g \omega_g \leq \sum_{t \text{ split}} \sum_{o \in \Omega_t'} \mathbf{1}^\top o = (f + s)K \leq 2sK$. Every vector $\bar{o}_t - o$ vanishes on the groups with $\omega_g = 0$ and has coordinate sum 0. Vectors with coordinate sum 0 that vanish outside a set of j groups span at most $j - 1$ dimensions. At least $s + 1$ groups therefore have $\omega_g \geq 1$, and one of them lies outside I'. Every $g \in I'$ has $\omega_g \geq 1$, because

row g of the nonsingular matrix is nonzero. Hence $\sum_{g\in I'}\omega_g \le 2sK-1$, and some $g\in I'$ has $1\le\omega_g\le 2K-1$. Remove this g.

Load of the removed group. Let $\underline{T}_g$ and $\overline{T}_g$ be the least and largest load of g over all selections from the supports Ω'_t. Tokens with one pattern contribute a constant, which gives $\overline{T}_g-\underline{T}_g=\omega_g$. Because $g\in I$, $\bar{Q}_g(\bar{w})$ is an end of $[\ell_g,h_g]$ and therefore an integer. For a split token, $\sum_o \bar{w}_{to}o_g$ is a mean of the values o_g, $o\in\Omega'_t$, with positive weights. It lies strictly between their minimum and maximum when these differ. Because $\omega_g\ge 1$, some token has differing values, and $\underline{T}_g<\bar{Q}_g(\bar{w})<\overline{T}_g$. For integers this gives $\bar{Q}_g(\bar{w})-\underline{T}_g\le\omega_g-1$ and $\overline{T}_g-\bar{Q}_g(\bar{w})\le\omega_g-1$, and $\omega_g-1\le 2K-2$. Later supports are subsets of Ω'_t. The load of g in the final selection therefore lies in $[\underline{T}_g,\overline{T}_g]\subseteq[\ell_g-(2K-2),\,h_g+(2K-2)]$.

Termination. Every round that does not stop removes a group. If $R_k=\emptyset$, then $\mathcal{W}_k$ is a product of simplices, whose vertices put weight 1 on one pattern per token, and round k stops. In the final round, w^k is a selection $\hat{r}$ from w with $V(\hat{r})\ge U$. It meets the intervals of the groups in R_k, and every removed group has load within $2K-2$ of its interval. Hence $\mathrm{dev}(\hat{r})\le 2K-2$. □

This is the iterative relaxation of Király et al. (2012) for degree-bounded matroids, in the form with multiplicities of Bérczi et al. (2019, Algorithm 2). There the matroid is the partition matroid whose bases pick one pair (t,o) with $w_{to}>0$ for every token, and group g counts the pair (t,o) with multiplicity o_g. Every pair has total multiplicity K. By Theorem 1 of Bérczi et al. (2019) and its proof, their Algorithm 2 then returns a selection with value at least U that violates every interval by at most $2K-1$. Integer interval ends lower this bound to $2K-2$ (paragraph *Load of the removed group* above). With upper bounds only, the same method keeps every load at most h_g+K-1, with value at least that of the LP with these bounds, which is at least U (Király et al., 2012; Bérczi et al., 2019). Min-deviation rounding bounds both sides of every interval.

Proof of (c). Let (z_t) be a legal routing that meets the intervals, and put $o_t=o(z_t)$. Setting $w_{to_t}=1$ and $w_{to}=0$ otherwise gives $w\in\mathcal{W}$. By Lemmas A.1 and A.2(ii), $\sum_t a_t^\top z_t\le\sum_t v_t(o_t)\le D_G(\mu)$ for every μ.

Let z^0 be the routing that gives each token t the r_{tg} highest-scoring experts of every group g. By Lemma A.1, z^0_t is legal, has pattern r_t and scores $v_t(r_t)$. Hence z^0 has loads T and score $V(r)$, which is at least U by (b), and $U=\min_\mu D_G(\mu)$ by Lemma A.2(iii). Because z^0 uses only groups of A_t, it is an integral feasible point of (13), and the integral optimum that placement returns therefore scores at least $V(r)\ge U$. An integral feasible point gives each token K distinct experts from the at most M groups of A_t. The returned routing is therefore legal, and its loads lie between $\min\{\ell_g,T_g\}$ and $\max\{h_g,T_g\}$, where $\ell_g-(2K-2)\le T_g\le h_g+(2K-2)$ by (b). If it meets the intervals, then, by the first claim at a minimiser μ^*, every legal routing that meets the intervals scores at most $D_G(\mu^*)=U$, and therefore at most the score of the returned routing. □

A.5 Proof of Proposition 4.4

Proposition 4.4 (restated). *Let $\ell_g=h_g=\tau$ for every group, with τ an integer. (a) If $N\ge K$, $L\ge K$ and $2\le M<K$, then for every batch size B that is a multiple of N with $B\ge 2N$ there are batches with scores in $\{0,1,2\}$ on which every legal routing with score at least U puts at least $\tau+K-M$ assignments on some group. (b) Deciding whether some legal routing puts exactly τ assignments on every group and has score at least a given value is strongly NP-complete when N, L and K are part of the input, already for $M=2$, $L=K$ and scores in $\{0,1,2\}$.*

Proof of (a). We build the batches and then prove the bound. Put $n_A=M-1$ and $d=K-M+1$. Then $n_A\ge 1$, $d\ge 2$ and $n_A+d=K$.

The first $d+1$ tokens. Take K of the groups, and call n_A of them *anchor* groups. Each of $B_0=d+1$ tokens has the same scores: in each anchor group one expert scores 2 and the others score 0, every expert of the other d of the K groups scores 1, and every expert of the remaining $N-K$ groups scores 0. A legal route scores at most $2n_A+d=K+n_A$ on such a token. It attains $K+n_A$ exactly

when it takes the n_A experts of score 2 and d experts of one other group among the K. Leaving out an expert of score 2, or taking an expert of score 0, loses at least 1. A route with all n_A anchor experts and experts of two other groups would use $n_A + 2 = M + 1$ groups. If every such token puts weight $1/d$ on each of the d patterns attaining $K + n_A$, each of the K groups receives B_0 assignments from these tokens.

Fillers for every $B = cN$ with an integer $c \geq 2$. Then $\tau = BK/N = cK$; put $\tau_0 = \tau - B_0 = (c-1)K + M - 2 \geq K$. Add *fillers* that supply τ_0 assignments to each of the K groups and τ to each of the other $N - K$ groups. To build them, list each group's label once for every assignment the group should receive, with equal labels next to each other, and cut the list into consecutive blocks of K labels. Every label is listed at least K times because $\tau_0 \geq K$ and $\tau \geq K$. A block of K consecutive labels therefore uses at most two groups, and it is a legal pattern because $M \geq 2$ and $K \leq L$. Each block becomes one filler, which scores 1 on the experts that the block assigns to it and 0 elsewhere. A filler scores K on its own route and less on every other route. The fillers add $K\tau_0 + (N - K)\tau$ assignments, and the batch has $B = B_0 + \tau_0 + c(N - K) = cN$ tokens, with $BK/N = cK = \tau$.

The bound. The weights $1/d$ above, with every filler on its own route, give a point of the LP (3) that puts exactly τ assignments on every group and gives every token its best score. Because no point of the LP gives a token more than its best score, U is the sum of these scores. A legal routing with score at least U therefore gives every filler its own route and each of the B_0 tokens a route that scores $K + n_A$, with its d assignments outside the anchor groups on one group. Because the $d + 1$ tokens choose among d groups, two of them choose the same group. That group receives at least $\tau - B_0 + 2d = \tau + d - 1 = \tau + K - M$ assignments. All scores lie in $\{0, 1, 2\}$. □

Proof of (b). The problem is in NP because a routing is a certificate that we can check in polynomial time. We reduce from 3-PARTITION, which is strongly NP-complete (Garey & Johnson, 1979, Section 4.2 and problem SP15): given $3m$ positive integers b_i with $T/4 < b_i < T/2$ and $\sum_i b_i = mT$, decide whether they form m triples of sum T each. Set $K = L = T$, $M = 2$ and $N = m + 1$: one anchor group and m bin groups. Item token i scores 2 on $K - b_i$ experts of the anchor group, 0 on the other anchor experts and 1 on every expert of every bin group. Its best score, $2K - b_i$, is reached only by routes that take its $K - b_i$ experts of score 2 and b_i experts of a single bin group, because a second bin group would make three groups. Add $m + 2$ fillers on the anchor group and $3m + 1$ fillers on every bin group. Each filler scores 1 on every expert of its group and 0 elsewhere, and its best score, K, is reached only by taking its whole group. Then $B = 3m + (m + 2) + m(3m + 1) = (m + 1)(3m + 2)$ and $\tau = BK/N = (3m + 2)K$. Let the threshold be the sum of the best scores of all tokens. A routing that meets the target with this score gives every token its best score. The fillers of a bin group then put $(3m + 1)K$ assignments on it, and the item tokens that choose this group supply the remaining $K = T$. Their numbers b_i therefore sum to T, and $T/4 < b_i < T/2$ forces exactly three item tokens per bin group: a 3-partition. Conversely, a 3-partition gives such a routing, and the anchor group then receives $(m + 2)K + \sum_i (K - b_i) = (3m + 2)K = \tau$ assignments. The instance has $(m + 1)(3m + 2)$ tokens and $(m + 1)T$ experts with scores in $\{0, 1, 2\}$, and its size is polynomial in m and T. 3-PARTITION is NP-complete already on instances with T polynomial in m. On such instances the reduction runs in polynomial time, and every number of the routing instance (scores in $\{0, 1, 2\}$ and a threshold of at most $2BK$) is bounded by a polynomial in its size. Hence the problem, which is in NP, is strongly NP-complete. □

For a fixed shape, a dynamic programme solves the problem in time polynomial in B: it routes the tokens one at a time and keeps, for every vector of partial group loads, the best total score. With loads of at most τ and a fixed total, there are at most $(\tau + 1)^{N-1}$ such vectors, about $4.9 \cdot 10^{12}$ at the deployed shape with $B = 64$.

A.6 Proof of Proposition 4.5

A batch-blind policy gives each token a legal route that is a function of the token's own scores and a random seed. The seeds are independent of everything else. The released rule with a fixed bias $\phi \in \mathbb{R}^E$, which gives the route $n(a_t + \phi)$, is such a policy. Prices learned from past requests and biases updated after each batch are also such policies, because they do not depend on the batch at hand. Because N divides B, τ is an integer.

Proposition 4.5 (restated). *Let $N \geq 2$, $K \leq L$, N divide B and $0 \leq \varepsilon < 1/2$. There are random traffic mixes of two document types, with a gap $\gamma > 0$ between high and low scores, on which every batch-blind policy, with probability at least $1/2$, either loads some group above $(1+\varepsilon)\,\tau$ or loses at least $\frac{\gamma}{2}(\frac{1}{2}-\varepsilon)\,\tau$ in score against the balanced optimum of the batch (the value U for $\ell_g = h_g = \tau$), whereas exact selection at prices set for each batch reaches that optimum with every group load equal to τ.*

Proof. The mix. Fix $0 < s_{\mathrm{lo}} < s_{\mathrm{hi}}$, put $\gamma = s_{\mathrm{hi}} - s_{\mathrm{lo}}$, and fix a noise level $0 \leq \sigma \leq \gamma(\frac{1}{2}-\varepsilon)/(4N)$. Then $\sigma < \gamma/4$. The document type $\kappa \in \{1,2\}$ is uniform. A batch consists of N blocks of B/N tokens; each block makes τ assignments. The *common* block scores s_{hi} on all experts of groups 1 and 2 and s_{lo} elsewhere. The other blocks are *dedicated*: one to group κ and one to each group $g \in \{3,\dots,N\}$ (none if $N = 2$). A block dedicated to g scores s_{hi} on the experts of G_g and s_{lo} elsewhere. Every score then receives noise in $[-\sigma,\sigma]$, independent of κ and the seeds. An assignment is *high* if its score before noise is s_{hi}, and *low* otherwise. The common block and its seeds are independent of κ.

Step 1: low assignments. Fix a batch-blind policy, and let Q and V be its loads and score. Let X_1 and X_2 be the numbers of assignments of the common block to groups 1 and 2, and $X_0 = \tau - X_1 - X_2$ the number of its other assignments, all low. Let $j = 1$ if $X_1 \geq X_2$ and $j = 2$ otherwise. Then $X_j \geq (\tau - X_0)/2$. The policy routes the common block from its scores and seeds alone; j is therefore independent of κ, and $\Pr\{\kappa = j\} = \frac{1}{2}$. Assume $\kappa = j$, and let Y be the number of assignments of the block dedicated to j outside G_j, all low. Group j receives X_j assignments from the common block and $\tau - Y$ from the block dedicated to j, which gives $Q_j \geq X_j + \tau - Y$. If $\max_g Q_g \leq (1+\varepsilon)\tau$, then $Y \geq X_j - \varepsilon\tau$, and the number F of low assignments satisfies

$$F \;\geq\; X_0 + Y \;\geq\; X_0 + \frac{\tau - X_0}{2} - \varepsilon\tau \;\geq\; \Big(\frac{1}{2} - \varepsilon\Big)\tau.$$

Step 2: the score gap. A high assignment scores at most $s_{\mathrm{hi}} + \sigma$, and a low one at most $s_{\mathrm{hi}} - \gamma + \sigma$. Hence $V \leq BK(s_{\mathrm{hi}} + \sigma) - \gamma F$. On $\{\kappa = j\}$, send each dedicated block to its own group and the common block to group $3 - j$, every token taking K experts of that group, which is legal because $K \leq L$. Every group then receives exactly τ assignments, all high, and the routing scores at least $BK(s_{\mathrm{hi}} - \sigma)$. Its patterns give a point of $\mathcal{W}$ whose value is at least this score (Lemma A.1). Hence $U \geq BK(s_{\mathrm{hi}} - \sigma)$. On $\{\kappa = j\} \cap \{\max_g Q_g \leq (1+\varepsilon)\tau\}$, Step 1 gives

$$U - V \;\geq\; \gamma F - 2\sigma BK \;\geq\; \gamma\Big(\frac{1}{2} - \varepsilon\Big)\tau - \frac{\gamma}{2}\Big(\frac{1}{2} - \varepsilon\Big)\tau \;=\; \frac{\gamma}{2}\Big(\frac{1}{2} - \varepsilon\Big)\tau,$$

because $2\sigma BK \leq \gamma(\frac{1}{2}-\varepsilon)BK/(2N) = \frac{\gamma}{2}(\frac{1}{2}-\varepsilon)\tau$. The event of the proposition therefore contains $\{\kappa = j\}$, which has probability $\frac{1}{2}$.

Step 3: prices set for the batch. The batch determines κ: it is the group $g \in \{1,2\}$ such that some block scores high on G_g and low on the other of the groups 1 and 2. Set the prices $\mu_\kappa = \gamma/2$ and $\mu_g = 0$ for $g \neq \kappa$. At the priced scores, a token dedicated to $g \neq \kappa$ scores at least $s_{\mathrm{hi}} - \sigma$ on G_g and at most $s_{\mathrm{lo}} + \sigma$ elsewhere. A token dedicated to κ scores at least $s_{\mathrm{hi}} - \sigma - \gamma/2$ on G_κ and at most $s_{\mathrm{hi}} - \gamma + \sigma$ elsewhere. A common token scores at least $s_{\mathrm{hi}} - \sigma$ on $G_{3-\kappa}$ and at most $s_{\mathrm{hi}} + \sigma - \gamma/2$ elsewhere. Because $\sigma < \gamma/4$, in each case every expert of one group G_{g^*} has a higher priced score than every expert outside it. A route with $k < K$ experts in G_{g^*} therefore has a lower priced score than the K highest priced scores of G_{g^*}, which form a legal route because $K \leq L$. Hence exact selection gives every token K experts of its group G_{g^*}: each dedicated block its own group, and the common block group $3 - \kappa$. Every group receives exactly τ assignments. Each token's priced score is $H_t(\mu)$, and the score V^* of this routing is

$$V^* = \sum_t \big[H_t(\mu) + \mu^\top o(z_t)\big] = \sum_t H_t(\mu) + \tau\,\mathbf{1}^\top\mu = D_G(\mu) \;\geq\; U \;\geq\; V^*.$$

The inequality $D_G(\mu) \geq U$ is Lemma A.2(iii). The routing's patterns give a point of $\mathcal{W}$ whose value is at least V^* (Lemma A.1), and therefore $U \geq V^*$. Hence $V^* = U$, with loads $Q(\mu) = \tau\mathbf{1}$. □

B EXPERIMENTAL DETAILS

Models. Ling-mini-2.0 (inclusionAI, 2025a) and Ring-mini-2.0 (inclusionAI, 2025b) share one architecture with 19 MoE layers of 256 routed experts at the deployed shape $(N, L, M, K) = (8, 32, 4, 8)$. The router computes sigmoid gates, and the trained bias enters the selection but not the output weights. All model weights stay frozen, and only the routing changes.

Domains. Code takes Python files from the validation split of codeparrot-clean (CodeParrot, 2022). ZH, EN and RU take Chinese, English and Russian conversations from WildChat-1M (Zhao et al., 2024). The dialogue domain takes English WildChat conversations with at least six turns that are not in EN. Wikipedia takes articles from WikiText-103 (Merity et al., 2017), the scored sequences from its test split and the others from its training split. Math takes the worked solutions of problems from the test set of MATH (Hendrycks et al., 2021b) that are not in MATH-500 (Lightman et al., 2024). Legal takes questions from the professional-law part of the MMLU test set (Hendrycks et al., 2021a), excluding the questions of our MMLU evaluation. ZH, EN, RU and the dialogue domain are rendered with each model's own chat template, whereas the other four domains are plain text.

Batches and scoring. Each domain supplies 64 sequences, each a window of 161 tokens at a seeded random offset, without an inserted beginning-of-sequence token. A batch holds the tokens at one window position of all 64 sequences ($B = 64$), as in decoding. A window gives 160 forward steps. The first 32 positions are warm-up. At the other 128 positions, the next-token predictions of the first 32 sequences are scored, and the other 32 sequences only fill the batch. With 19 MoE layers, each router and domain has 3,040 routing steps. In a domain, the NLL difference between two routers is the mean over the 32 scored sequences of the difference between their mean NLLs over the 128 scored tokens. Because the 32 sequences share every batch and the routing couples them, we treat each domain as one data point and report the mean $\pm$ SE over the eight domains.

Scores and output weights. Let s_{te} be the gate of expert e for token t, without the trained bias, and let R_t be the token's released route. HYPERSPACE and fixed-group flow maximise sums of the scores $a_{te} = s_{te}/Z_t$, where $Z_t = \sum_{j \in R_t} s_{tj}$. Every released route therefore has score exactly 1. Every router sets the output weights as the model does. In Ling-mini-2.0 and Ring-mini-2.0, the selected gates are divided by their sum and multiplied by 2.5.

Routers. The static per-expert bias adds one frozen value per layer and expert to the selection scores of the released rule. The static group prices subtract one frozen value per layer and group from the selection scores of all experts in the group. We calibrate both with the sign update of auxiliary-loss-free balancing (DeepSeek-AI, 2024b) on router scores recorded from each model on English WildChat conversations that appear in no evaluation domain. After each recorded batch, every value moves by 0.001 towards balance, and after ten passes over the batches, we freeze the values.

Group dropping starts from the released routes and keeps, in each group, the assignments with the highest gates up to its cap, $\tau = 64$ unless stated otherwise. Fixed-group flow keeps the released groups of each token and reassigns its experts within them by a max-score flow on a that meets the target. HYPERSPACE runs Algorithm 1 on a, with each LP solved by the dual simplex method of HiGHS (Huangfu & Hall, 2018). To make the restricted LP feasible from the first round, every load row carries a slack variable with cost $C = 10K \max\{1, \max_{t,e} |a_{te}|\}$, as in Lemma A.2(v). HYPERSPACE returns a routing only if the slacks sum to at most 10^{-7} when the rounds stop, which held on every batch. The rounds also stop once D_G at the prices of some round is at most the value of the restricted LP, the second condition of Lemma A.2(iv).

Replay on recorded router scores. The replay of Section 5.2 uses the router scores recorded for the static calibration. We ran each model unmodified on the first 512 tokens of eight sets of 64 English WildChat conversations, which appear in no evaluation domain. We recorded the gates of all 19 MoE layers at every token, together with the trained bias. A decoding batch takes one token from each of the 64 conversations of a set, token $(b + 8i) \bmod 512$ from conversation i in batch b; we replay all 512 batches of every set at every MoE layer. A prefill batch takes four conversations,

2,048 tokens; we replay all 16 batches of every set at layers 1, 10 and 19. Each model gives 77,824 decoding and 384 prefill batches. Because $K = N$, the fair share is $\tau = B$, and the target interval is $[\tau, \tau]$. Every router sees the same scores, and the routing score is the sum of a over the kept assignments. HYPERSPACE, group dropping with cap τ and fixed-group flow are those of the quality runs.

Capacity-aware rerouting takes the released routes and moves assignments out of the groups above τ, those with the lowest score first, each to the best legal unused expert in a group below τ. It drops nothing and keeps the overflow it cannot move. Unlike the Expanded Drop of He et al. (2026), which adds only local experts, it can move an assignment to another group. Batch-prioritised routing at the group level, adapted from Riquelme et al. (2021), takes the tokens in decreasing order of their highest gate. It keeps a token's released route if the route fits under the group capacity τ, otherwise takes the best legal route that fits, and drops the token's assignments if none does.

Group prices on the released rule start from zero. At every iterate, each token takes the route that the released rule picks at the current prices. The prices then move by $\mu \leftarrow \mu + \eta_k(Q - \tau\mathbf{1})$ and are shifted to mean zero, with $\eta_k = m\,(0.01/\tau)/\sqrt{k+1}$ and the step multiplier $m = 1$. The rule ranks either the scores a minus the group prices, or the model's own selection scores (the gates plus the trained bias) minus Z_t times the group prices. The first iterate of the second variant is the released route. Among its first 20 or 200 iterates, the router returns the one with the highest score among those whose busiest group is at most $\tau + 2$, or else the one whose busiest group is least loaded. The oracle variant runs the same number of iterates for each step multiplier $m \in \{0.1, 0.3, 1, 3, 10\}$ and keeps, for each batch, the iterate with the smallest excess over τ and, among these, the highest score. Table 8 gives these routers with their largest excess. HYPERSPACE returns a legal route for every token of every batch, and its score is at least U up to a relative $2 \cdot 10^{-9}$.

Larger prefill batches. We also replay the recorded scores in prefill batches of 512, 8,192 and 16,384 tokens, which take one, 16 and 32 whole conversations of a set in request order, at layers 1, 10 and 19 of every set. The routers are those of Table 2 except fixed-group flow, which we did not run at these sizes, and each is called as in the replay above. Table 12 adds the batches of Section 5.2 at the same layers. At every size, HYPERSPACE keeps every group within 4 assignments of the target, splits at most $N - 1 = 7$ tokens and scores at least U up to a relative $2 \cdot 10^{-9}$. Its largest excess falls from 4 at 64 tokens to 2 at 16,384, whereas that of group prices on the released rule with 200 iterates grows from 5 to 150.

Released rule against exact selection. On the same recorded scores, we compare the released rule with exact selection for every token at every MoE layer: 8 sets, 512 tokens, 64 conversations and 19 layers, 4,980,736 token-layer pairs per model. For each pair, P is the score of the route that the released rule picks and H that of the best legal route, found by the dynamic programme of Lemma A.3, both on the same scores. We use the selection scores of the model (the gates plus the trained bias), and also the gates alone, which are nonnegative, where Theorem 4.1(b) gives $P \geq \frac{8}{11}H$. A pair counts as a miss when $H - P > 10^{-9}\max\{1, |H|\}$. Table 11 gives the results.

Expert placements. We ported EPLB (DeepSeek-AI, 2025a), the expert-parallel load balancer of DeepSeek, line by line to NumPy; the port reproduces the example of its documentation exactly. For each model and MoE layer, EPLB places the 256 experts on the 8 GPUs of one node with 8 groups (its hierarchical policy), with $R \in \{0, 1, 2, 4, 8\}$ redundant experts per GPU. EPLB fits a placement to the expert loads of the released router. The loads are counted on the four conversation sets of the half that does not contain the evaluated set (OUT), on all eight sets (ALL), or, as a reference with exact look-ahead, on the evaluated set itself (WORLD). The released routes of each batch then run on the placement, and the load of a replicated expert is split evenly over its replicas, as in EPLB's own model of balance. The even split favours EPLB. HYPERSPACE keeps group g on GPU g, without replicas (Table 9).

Hardware and the run with a cap of 68. The NLL runs of Ling-mini-2.0 and Ring-mini-2.0 used NVIDIA RTX A6000 GPUs, and those of Qwen3-30B-A3B NVIDIA H200 GPUs. The comparison with a cap of 68 in Table 3 comes from a separate run on NVIDIA H200 GPUs, which reran the released router, HYPERSPACE and group dropping with caps 64 and 68 on every domain of both models. Because small numerical differences between the GPUs change some discrete routing deci-

sions, the table compares the cap of 68 with HYPERSPACE from this run. In this run, the mean over the domains of HYPERSPACE minus group dropping with a cap of 64 is -0.112 on Ling-mini-2.0 and -0.106 on Ring-mini-2.0, against -0.111 and -0.106 in the main runs.

Qwen3-30B-A3B. Qwen3-30B-A3B (Qwen Team, 2025) has 48 MoE layers of 128 experts, with softmax gates, $K = 8$ and no group limit. We impose one with the shape $(N, L, M, K) = (8, 16, 4, 8)$, with groups of 16 contiguous experts. The released rule, applied to the gates at this shape, stands in for the released router, and Z_t is the gate sum of the route it selects. We use the domains as raw text without a chat template, encoded with the Qwen3 tokeniser. Windows, batches and scoring are as above.

Downstream tasks. All downstream runs use Ling-mini-2.0. On GSM8K (Cobbe et al., 2021) and HumanEval (Chen et al., 2021), the router under test routes every prompt and generated token. Prompts run in consecutive batches of up to 64, in dataset order. Each sequence feeds its prompt one token per step and then generates, and a finished sequence leaves the batch; every layer computes the fair share $\tau = BK/N$ from the current batch size B. A GSM8K prompt is "Question:" and the problem, followed on a new line by "Answer: Let's think step by step." (Kojima et al., 2022); HumanEval uses its standard prompts. Decoding is greedy, with at most 512 new tokens on GSM8K and 384 on HumanEval. The GSM8K output is cut at the next "Question:". The prediction is the first number that directly follows "the answer is" or "the final answer is" (ignoring case; a colon or a dollar sign may come in between). If there is none, it is the last number in the output. HumanEval is scored by pass@1 through execution. The first 500 GSM8K problems and HumanEval ran on one NVIDIA RTX A6000, and the other 819 GSM8K problems on NVIDIA H200 GPUs, with the same prompts, decoding and answer rule.

On ARC-Challenge (Clark et al., 2018) and HellaSwag (Zellers et al., 2019), zero-shot, and on MMLU (Hendrycks et al., 2021a), 5-shot, each task has 800 prompts, drawn at random with a fixed seed from the ARC-Challenge test set, the HellaSwag validation set and the MMLU test set. The MMLU draw is stratified by subject, and each prompt's five demonstrations are the development examples of its subject. ARC-Challenge and HellaSwag ran on one NVIDIA RTX A6000 and MMLU on NVIDIA H200 GPUs. The router under test routes each prompt, demonstrations included, as one batch at every layer, and the answer choices are then scored with the released router. As in the LM Evaluation Harness (Gao et al., 2024), the prediction is the choice with the highest total log-probability, raw or divided by the length of the choice in characters. Ties go to the first choice. For MMLU, the bootstrap for the intervals of Table 10 resamples within subjects.

Timing. Each timing run uses one NVIDIA H200 GPU on a node with two Intel Xeon Gold 6548Y+ CPUs of 32 cores each, and the LP runs on the CPU. The timer starts with the router scores on the GPU and stops when the expert indices and output weights are ready on the GPU. It covers the transfers, the LP and the rounding, but not the expert computation. The inputs are 192 batches of 64 router-score vectors recorded from Ling-mini-2.0 on WildChat conversations at layers 1, 10 and 19. The timed implementation keeps one HiGHS restricted LP across the pricing rounds and runs the pricing dynamic programme on the GPU. It skips the placement flow (Step 3 in Section 3) and gives each token the o_g highest-scoring experts of each group. On every batch, its LP value agrees with that of the implementation used in the quality runs to a relative 10^{-9}. We time the released router separately, 20 times on one synthetic batch of 64 tokens.

C ADDITIONAL RESULTS

Table 6: **NLL change from the released router in each domain** (mean over the 32 scored sequences of the domain), and its mean ± SE over the eight domains. Group dropping (cap 64), fixed-group flow and HYPERSPACE use the nearest-integer target. ZH, EN, RU: Chinese, English and Russian conversation.

Router	Code	ZH	EN	RU	Wikipedia	Math	Dialogue	Legal	Mean ± SE
Ling-mini-2.0									
Released rule without the bias	+0.001	−0.007	−0.004	−0.003	−0.013	−0.020	−0.014	−0.021	−0.010 ± 0.003
Static per-expert bias	+0.029	+0.109	+0.209	+0.068	+0.115	−0.019	+0.075	+0.080	+0.083 ± 0.024
Static group prices	−0.002	+0.022	+0.015	+0.003	+0.005	−0.007	+0.016	−0.007	+0.006 ± 0.004
Group dropping	+0.050	+0.127	+0.063	+0.970	+0.140	+0.025	+0.102	+0.105	+0.198 ± 0.111
Fixed-group flow	+0.046	+0.096	+0.035	+0.680	+0.034	+0.008	+0.013	+0.024	+0.117 ± 0.081
HYPERSPACE	+0.026	+0.058	+0.001	+0.589	−0.010	0.000	−0.013	+0.040	+0.086 ± 0.072
Ring-mini-2.0									
Released rule without the bias	+0.012	+0.001	−0.018	+0.025	−0.023	0.000	+0.028	+0.009	+0.004 ± 0.006
Static per-expert bias	+0.029	+0.092	+0.163	+0.073	+0.154	+0.018	+0.075	+0.181	+0.098 ± 0.022
Static group prices	+0.013	+0.012	−0.012	+0.001	+0.004	+0.022	+0.011	−0.001	+0.006 ± 0.004
Group dropping	+0.163	+0.095	+0.084	+0.841	+0.185	+0.055	+0.089	+0.269	+0.223 ± 0.092
Fixed-group flow	+0.035	+0.047	+0.074	+0.656	+0.043	+0.012	+0.044	+0.045	+0.119 ± 0.077
HYPERSPACE	+0.041	+0.038	+0.064	+0.622	+0.017	+0.002	+0.033	+0.115	+0.117 ± 0.073

Table 7: **NLL of HYPERSPACE minus that of another router on Qwen3-30B-A3B** (with 8 imposed groups of 16 experts and at most 4 groups per token, every MoE layer routed by the router under test): mean ± SE over the eight domains, and the number of domains ($k/8$) in which HYPERSPACE is lower. HYPERSPACE exceeds the fair share of 64 by at most 4 assignments on every routing step. Group dropping discards 18.1% of the assignments.

HYPERSPACE minus	ΔNLL ± SE	$k/8$
group dropping	−0.0166 ± 0.0056	8/8
released rule at this shape	+0.0182 ± 0.0055	0/8

Table 8: **All routers of the replay in Section 5.2** (155,648 decoding and 768 prefill batches). Over +14: batches whose busiest group exceeds the fair share by more than 14. Max: largest excess over the fair share. Score/U: mean over the batches of the routing score divided by U. Group dropping discards 13.8% and 12.0% of the assignments of the decoding and prefill batches, and batch-prioritised routing fewer than 0.01%; the other routers drop nothing.

	Decoding, $B = 64$			Prefill, $B = 2{,}048$		
Router	Over +14	Max	Score/U	Over +14	Max	Score/U
Released router	150,493	118	1.0065	768	3,393	1.0030
Group dropping	0	0	0.9232	0	0	0.9294
Batch-prioritised routing, group level	0	0	0.9048	0	0	0.9067
Fixed-group flow	293	106	0.9829	26	3,145	0.9864
Capacity-aware rerouting	37	23	0.9738	3	413	0.9751
Group prices on the released rule, ranking the scores a minus the prices						
20 iterates	1,007	22	1.0058	760	700	1.0063
200 iterates	0	5	1.0005	102	65	0.9987
20 iterates, oracle step size	0	5	0.9992	6	82	0.9985
200 iterates, oracle step size	0	2	0.9990	0	3	0.9985
Group prices on the released rule, ranking the gates plus the trained bias minus Z_t times the prices						
20 iterates	1,010	24	0.9912	756	736	0.9886
200 iterates	0	4	0.9863	80	46	0.9817
20 iterates, oracle step size	0	6	0.9850	33	58	0.9816
200 iterates, oracle step size	0	2	0.9848	0	5	0.9816
HYPERSPACE	0	4	1.0004	0	3	1.0000

Table 9: **Busiest GPU under HYPERSPACE and under EPLB placements** (DeepSeek-AI, 2025a) on the batches of Section 5.2, both batch kinds pooled (156,416 batches). Over +14: share of batches whose busiest GPU exceeds its fair share by more than 14 assignments. Max excess: largest excess over the fair share (fractional, because a replicated expert's load is split evenly over its replicas). Busiest/fair: mean and 99th percentile of the busiest GPU's load over the fair share. R: redundant experts per GPU. Extra memory: $R/32$ of the routed-expert memory. EPLB is fitted on the four conversation sets of the half without the evaluated set (OUT), on all of them (ALL), or, as a reference with exact look-ahead, on the evaluated set itself (WORLD). HYPERSPACE and the released router keep group g on GPU g, without replicas.

Placement	R	Extra memory	Over +14 (%)	Max excess	Mean busiest/fair	p99 busiest/fair
EPLB, OUT	0	0	26.6	1,522.0	1.189	1.391
	1	3.1%	24.5	1,319.3	1.182	1.375
	2	6.3%	22.4	1,484.8	1.177	1.367
	4	12.5%	20.4	1,241.5	1.171	1.352
	8	25%	14.4	1,150.8	1.158	1.326
EPLB, ALL	0	0	21.7	1,394.0	1.179	1.359
	1	3.1%	22.1	1,489.5	1.177	1.359
	2	6.3%	18.9	1,215.0	1.169	1.348
	4	12.5%	17.0	1,093.2	1.164	1.339
	8	25%	11.6	982.8	1.151	1.309
EPLB, WORLD (reference)	0	0	14.6	1,448.0	1.164	1.328
	1	3.1%	13.7	1,519.0	1.158	1.328
	2	6.3%	12.7	1,456.5	1.155	1.320
	4	12.5%	10.7	1,294.1	1.149	1.305
	8	25%	7.6	1,309.9	1.139	1.286
HYPERSPACE	0	0	0.0	4	1.011	1.031
Released router	0	0	96.7	3,393	1.597	2.344

Table 10: **Multiple-choice accuracy with Ling-mini-2.0**: change from the released router in points, with paired 95% bootstrap intervals below (800 prompts per task; the router under test routes only the prompt). ARC-Challenge and HellaSwag are zero-shot, scored by the total log-probability of a choice (raw) or by that total per character of the choice (char.); MMLU is 5-shot.

	ARC-Challenge		HellaSwag		MMLU
Router	raw	char.	raw	char.	5-shot
Group dropping	-1.375 $[-3.625, 0.875]$	-3.625 $[-5.875, -1.375]$	-0.250 $[-1.375, 0.875]$	-0.125 $[-1.500, 1.375]$	$+0.375$ $[-1.750, 2.500]$
HYPERSPACE	-0.375 $[-2.500, 1.750]$	-1.125 $[-3.375, 1.125]$	$+0.250$ $[-0.750, 1.250]$	$+1.625$ $[0.250, 3.000]$	$+0.250$ $[-1.750, 2.250]$

Table 11: **The released rule against exact selection on the recorded scores of Section 5.2** (4,980,736 token-layer pairs per model). Miss: share of the pairs on which the released rule picks a route with a lower score than the best legal route, with the selection scores (the gates plus the trained bias) and with the gates alone. Layer range: smallest and largest share over the 19 layers, with the selection scores. Loss: $(H-P)/H$ on the gates alone, mean over the missed pairs and largest value, set against the largest loss $3/11 \approx 27.3\%$ that Theorem 4.1(b) allows. Tokens: share of the tokens with a miss at one or more of the 19 layers, with the selection scores.

	Miss (%)			Loss on the gates (%)		
Model	Gates plus bias	Layer range	Gates alone	Mean	Largest	Tokens (%)
Ling-mini-2.0	12.8	8.3–15.4	13.2	0.98	8.9	91.7
Ring-mini-2.0	13.5	8.8–16.9	13.6	1.00	9.3	93.1

Table 12: **Replay at five batch sizes**, pooled over both models at layers 1, 10 and 19 (Appendix B): 24,576, 3,072, 768, 192 and 96 batches of 64, 512, 2,048, 8,192 and 16,384 tokens. The batches of 64 and 2,048 tokens are those of Table 2 at these layers. Over $+14$: batches whose busiest group exceeds the fair share by more than 14. Max: largest excess over the fair share. Score$/U$: mean over the batches of the routing score divided by U. Group dropping discards 9.9 to 17.3% of the assignments, and the other routers drop nothing.

	Tokens per batch, B				
Router	64	512	2,048	8,192	16,384
Over $+14$ (batches)					
Released router	22,131	3,072	768	192	96
Group dropping	0	0	0	0	0
Capacity-aware rerouting	0	141	3	0	0
Group prices on the released rule, 20 iterates	171	2,765	760	192	96
200 iterates	0	33	102	101	61
200 iterates, oracle step size	0	0	0	1	0
HYPERSPACE	0	0	0	0	0
Max					
Released router	84	1,474	3,393	8,367	14,835
Group dropping	0	0	0	0	0
Capacity-aware rerouting	14	246	413	0	0
Group prices on the released rule, 20 iterates	20	343	700	1,525	2,666
200 iterates	5	41	65	114	150
200 iterates, oracle step size	2	3	3	16	11
HYPERSPACE	4	3	3	3	2
Score$/U$					
Released router	1.0000	1.0244	1.0030	0.9970	0.9961
Group dropping	0.9293	0.9074	0.9294	0.9369	0.9383
Capacity-aware rerouting	0.9763	0.9668	0.9751	0.9774	0.9776
Group prices on the released rule, 20 iterates	1.0062	1.0119	1.0063	1.0048	1.0045
200 iterates	1.0007	0.9984	0.9987	0.9988	0.9988
200 iterates, oracle step size	0.9991	0.9980	0.9985	0.9986	0.9986
HYPERSPACE	1.0003	1.0001	1.0000	1.0000	1.0000