To Achieve Truly Generalist Models, We Need to Incentivize Collaboration Through Fair Revenue Sharing

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large language models (LLMs) are still developed and served as isolated, single-provider systems. While each excels on a set of benchmarks, real-world applications demand competence across many tasks and domains. In principle, an aggregate model that combines the strengths of multiple specialized checkpoints would Pareto-dominate today's monoliths—matching or exceeding every individual model on every objective. Realizing such a frontier, however, is impossible without collaboration among the diverse actors who control data, weights, compute, and user distribution. Collaboration raises a thorny question of who gets paid: each stakeholder contributes their distinct resources and will cooperate only if the additional revenue is shared in a way they perceive as fair. We argue that constructing truly generalist LLMs therefore hinges on mechanism design—specifically, revenue-sharing rules that are transparent, incentive-compatible, and robust to externalities. Drawing on cooperative game theory, we outline how Shapley-inspired allocations solution concepts can distribute the surplus revenue from such collaborations fairly. By embedding such mechanisms into model-hosting platforms and API brokers, the LLM community can move from siloed competition to productive cooperation, accelerating progress toward universally capable, socially beneficial language technologies.

1 Introduction

Large language models (LLMs) are still built and deployed largely as **monolithic artifacts**: a single organization curates the data, trains the weights, and serves the model behind an API. This siloed approach has delivered spectacular one-objective benchmarks—code generation here, medical QA there—but it struggles with the reality that **real-world users have many objectives at once**. No single checkpoint simultaneously tops the leaderboards for academic, health, finance, and programming queries (1).

From a social-welfare perspective this is wasteful. In theory, if we could *aggregate the specialized strengths of many models into a single composite agent*, the resulting system would *Pareto-dominate* every individual model: at least as good on each task, strictly better on some. Achieving that frontier, however, is impossible for any one lab acting alone; it requires *collaboration across data owners, model developers, compute providers, and service platforms*.

Collaboration introduces its own problem: who gets paid, and how much? Each stakeholder contributes a different scarce resource—high-quality domain data, proprietary weights, inference GPUs, user traffic—and each has the technical capability to release its own model outside of a coalition. Without a mechanism that *allocates the additional revenue created by collaboration in a way all*

Submitted to the NeurIPS 2025 Workshop on Algorithmic Collective Action. Do not distribute.

parties perceive as fair, rational actors will simply refuse to cooperate, leaving the status quo of narrow, duplicated models in place.

Our position is therefore straightforward: **to build truly generalist LLMs we must actively incentivize multi-stakeholder cooperation through principled, transparent revenue-sharing rules.** Mechanism-design tools such as cooperative game theory already offer a rich foundation. What is missing is their systematic application to the emerging LLM ecosystem.

After formalizing our setup and formulating our cost-benefit functions, we cast multi-provider LLM ecosystems as a cooperative game with *externalities* (i.e., where the revenue of a coalition of stakeholders is impacted by those outside of the coalition—a reality in the global LLM market). Our main technical contributions are:

A benchmark-aware revenue function. We couple model accuracy on public leaderboards with inference cost and market demand, creating a common yard-stick for heterogeneous agents.

Two cooperation paradigms. We analyze (i) weight-space collaboration—model merging, mixture-of-experts, and MoErging—and (ii) API-level routing, where a broker directs queries to the cheapest competent endpoint.

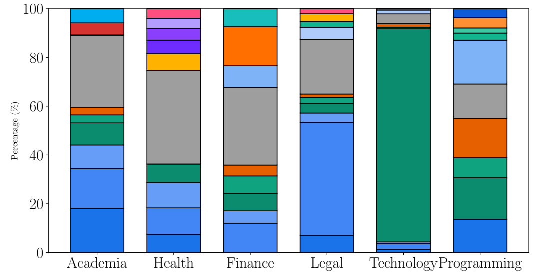


Figure 1: **LLM market share across various tasks.** No single model dominates in every domain: ■ Google: Gemini 2.5 Flash Preview 04-17, ■ Google: Gemini 2.0 Flash, ■ Google: Gemini 1.5 Flash, ■ Google: Gemini 2.5 Pro Preview, ■ Google: Gemini 2.0 Flash Lite, ■ Google: Gemini 2.5 Flash Preview 04-17 (thinking), ■ OpenAI: GPT-4o-mini, ■ OpenAI: GPT-4.1, ■ OpenAI: GPT-4o (2024-11-20), ■ OpenAI: GPT-4.1 Mini, ■ Anthropic: Claude 3.7 Sonnet, ■ Anthropic: Claude 3.5 Sonnet, ■ Anthropic: Claude 3.7 Sonnet (thinking), ■ Meta: Llama 3.2 3B Instruct, ■ Meta: Llama 4 Maverick, ■ Meta: Llama 3.3 70B Instruct, ■ Others, ■ DeepSeek: DeepSeek V3 0324, ■ Cohere: Command R7B (12-2024), ■ NousResearch: Hermes 2 Pro – Llama-3 8B, ■ xAI: Grok 3 Mini Beta, ■ Mistral: Mistral Small 3

A coalition-formation mechanism. To ensure a fair, efficient, and strategy-proof revenue sharing rule, we argue for and choose an appropriate extension of the Shapley value (known as Macho-Stadler value) that properly accounts for externalities. Our choice admits a compact representation that makes it viable for latency-sensitive systems such as LLM routers.

Decision guidelines. We characterize when actors should remain singletons, merge weights, or enter routing coalitions as a function of performance dispersion, demand elasticity, and compute prices.

2 Problem Setting

In this section, we formulate our problem and present our evaluation and cost metrics.

We consider a set of *LLM agents*, which abstractly represent **model providers**. These agents can range from large companies with extensive computational resources to smaller entities that commission models through fine-tuning or prompting on proprietary datasets. Formally, we denote the set of agents as $N = \{A_1, A_2, \ldots, A_n\}$, each characterized by their model parameters θ_i, trained on proprietary datasets D_i using private computational resources Φ_i. The performance of these agents is evaluated across benchmarks $B = \{B_1, B_2, \ldots, B_m\}$. Each agent A_i achieves a reward $r_{i,j} \in [0, 1]$ on benchmark B_j.

Assumptions. Our first assumption is that model performance is evaluated on transparent, public benchmarks. This assumption is consistent with the existing practice of public leaderboards and internal testing of models [1]. We also note that the granularity of the evaluation we require for our purposes is much coarser than what is needed for data attribution, as in Park, Georgiev, Ilyas, *et al.* [2] (see our discussion in Appendix A). This distinction allows us to avoid indeterminacy and the significant cost of such techniques, and provide algorithms that are applicable in latency-sensitive applications. Our second assumption is that demand for a model is driven by its utility (accuracy, cost, availability). We define the domain-level revenue function as follows:

Definition 1 (Revenue Function on Domain x)**.**

$$r(x) = q(x)\left[\alpha u(x) - (1-\alpha)c(x)\right], \tag{1}$$

where $q(x)$ is the user demand for domain x, dependent on price and model performance, and $u(x)$ is the model utility. $c(x)$ is the associated cost (including training cost, inference cost, or API calling cost).

In Definition 1, $\alpha \in [0, 1]$ is a tunable parameter reflecting the utility-cost trade-off. It captures the trade-off between model performance and cost in the objective function. For example, high-value financial analysis tasks may emphasize performance more ($\alpha \approx 0.8 \sim 0.9$); tasks involving a large amount of repetitive code generation or basic functions may focus more on cost ($\alpha \approx 0.4 \sim 0.6$).

Model Performance $p(x)$ **and Utility (Monetary Value)** $u(x)$**.** To normalize model performance, for a given model $a \in \mathcal{A}$, let $S(x)$ denote its performance score (e.g., accuracy or pass@k [3]) on publicly recognized benchmark (such as MMLU [4], BIG-bench [5], TruthfulQA [6], etc.) as an objective measure of model performance on task x.

Then the normalized performance $p(x)$ is defined as:

$$p(x) = \frac{s(x)}{\max_{a' \in \mathcal{A}} s(x)}, \quad 0 \leq p(x) \leq 1 \tag{2}$$

To directly convert the normalized model performance into monetary terms, we define utility as:

$$u(x) = p(x) \times m(x) \tag{3}$$

where $m(x)$ is the monetary value per unit performance.

Model Cost $c(x)$**.** If the model is white-box and the weights are available (i.e., checkpoints can be accessed and the provider deploys the model themselves), the costs include both training and inference. If the model is black-box (i.e., only accessible through an API), the main cost is the API usage fee.

Training, Fine-tuning, and Inference Cost Estimation. Typically determined by compute pricing (e.g., GPU or TPU cost):

$$c_{\text{train/infer}}(x) = t(x) \times p_{\text{GPU}}, \tag{4}$$

where $t(x)$ is the GPU hours consumed to perform training or inference on x, and p_{GPU} the cost per-GPU-hour.

API Calling Cost Estimation. Following established industry-wide pricing schemes [7], we model this cost based on the input token price and output token price:

$$c_{\text{api}}(x) = (\text{input\_tokens}(x) \times c_{\text{input}}) + (\text{output\_tokens}(x) \times c_{\text{output}}), \tag{5}$$

where $\text{input\_tokens}(x)$ is the number of input tokens for input x, $\text{output\_tokens}(x)$ is the number of output tokens generated from input x, c_{input} is the cost per input token and c_{output} is the cost per output token.

Definition 2 (Revenue Function Across All Domains)**.** We define the total coalition utility as the weighted sum across all domains:

$$r^{\text{total}} = \sum_{x \in \mathcal{X}} w(x) \cdot r(x), \tag{6}$$

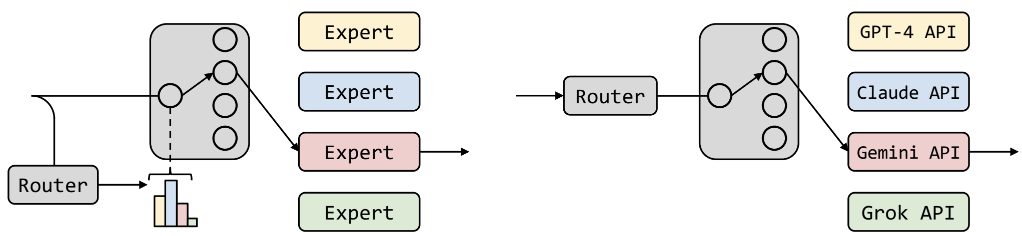
where $w(x)$ is a domain-specific weight representing the relative frequency or importance of domain x within the overall task distribution.

Specifically, assume there is a set of benchmarks $B = \{B_1, B_2, \ldots, B_m\}$, each associated with a weight w_j indicating its importance or the frequency of user queries related to that benchmark, with $\sum_{j=1}^{m} w_j = 1$. Thus, total revenue on all domains in B is $\sum_{j=1}^{m} w_j r_j$ where w_j are benchmark weights.

3 Collaboration of Large Language Models

In this section, we identify several approaches for collaboration among models forming a coalition. These approaches fall into two categories: *weight-space collaboration*, applicable to white-box models whose weights (checkpoints) are directly accessible and can be deployed by the provider, and *API-level collaboration*, suitable for black-box models that are only accessible through an API.

Paradigm I: Weight-Space Collaboration Weight space coordination involves modifying model weights to create a unified or composite model that leverages the strengths of multiple models trained on different datasets or domains. This approach reduces training and fine-tuning costs compared to individual adaptation, particularly when addressing distribution shifts across tasks. It encompasses several techniques, including model merging, Mixture of Experts (MoE), and Model MoErging, each offering distinct mechanisms for collaboration.



(a) **Weight-space collaboration** (b) **API-Level Routing**

Figure 2: **Illustration of two model collaboration paradigms.** (a) a central router aggregates or merges multiple expert models into a single composite model that leverages diverse domain strengths; (b) a request router dispatches each inference query to the most suitable endpoint.

Paradigm II: API-level Collaboration API-level coordination, or API routing [8]–[11], involves distributing user queries during inference to the most suitable *black-box model* provider based on performance metrics from public benchmarks. Unlike weight space coordination, such as Mixture-of-Experts, API routing *does not modify model weights (no training process) or access model checkpoints when inference*; it optimizes query allocation across existing models to maximize utility and minimize inference costs. For example, a coalition might route a coding query to a provider whose model excels in generating functional code, as validated by benchmark pass rates.

3.1 Evaluating the Value of Collaboration

Having defined how collaborations can be formed in the context of LLMs, in this section we formalize a revenue function $r(\cdot)$ for a "coalition" of LLM agents. Note that *a priori*, any subset $S \subseteq N$ of the set of LLM agents N can potentially be a coalition. We dedicate this section to defining the revenue function for any *potential* coalition S. In Section 4 we use this revenue function to find viable coalitions.

Definition 3 (Coalition Revenue Function on Domain x)**.** The total revenue of a coalition $S \subseteq \{A_1, A_2, \ldots, A_n\}$ is defined as:

$$r_S(x) = q(x) \times [\alpha \cdot u_S(x) - (1 - \alpha) \cdot c_S(x)]$$

Coalition Utility Function $u_S(x)$

$$u_S(x) = p(x) \times m(x) \tag{7}$$

where $p(x)$ is the normalized performance score of the coalition (merged model, or mixture-of-experts):

$$p(x) = \frac{s(x)}{\max_{x' \in \mathcal{A}} s(x')}, \quad 0 \leq p(x) \leq 1. \tag{8}$$

$m(x)$ is the monetary value per unit performance.

Coalition Cost Function $c_S(x)$ is generally lower than the sum of individual costs, reflecting the cost-efficiency of collaboration due to economies of scale.

Definition 4 (Coalition Revenue Function Across All Domains)**.** We define the total coalition utility as the weighted sum across all domains:

$$r_S^{\text{total}}(x) = \sum_{x \in \mathcal{X}} w(x) \cdot r_S(x), \tag{9}$$

where $w(x)$ is a domain-specific weight representing the relative frequency, importance, or strategic value of domain x within the overall task distribution.

Coalition reward on benchmark B_j is:

$$R_{S,j} = \max_{A_i \in S} r_{i,j} \qquad \text{with total reward:} \qquad R_S = \sum_{j=1}^{m} w_j R_{S,j}, \tag{10}$$

where w_j are benchmark weights, and $r_{i,j}$ is the performance of agent A_i on benchmark B_j.

Table 1 compares different collaboration paradigms and their corresponding coalition revenues. API-level collaboration offers simple deployment with no training costs but may lead to model duplication or increased latency. In contrast, weight-space collaboration through coalition formation enables shared training or model merging, effectively reducing per-model adaptation costs.

4 Coalition Formation

In this section, we formulate a coalition formation game between the stakeholders. Coalition formation games are a class games where stakeholders arrive at an agreement—they cooperate to achieve a common goal. This is in contrast to non-cooperative games where stakeholders ultimately choose their strategy independently, even if they are allowed to collude, correlate their strategies, or share information with each other [12].

Consider the set of all stakeholders N. Our setting has the following key properties:

P1. Stakeholders seek to reach an agreement.

P2. The revenue of the stakeholder and any coalitions of stakeholders is ultimately monetary, and therefore transferable.

P3. Multiple coalitions can form as a result of strategic alliances between companies. The value that a coalition S brings to its members is not only a function of S's members but also of how non-members behave and the coalitions they form.

P1 and **P2** mean that a coalition formation game with transferable utility is a sensible choice. A common solution concept for such a game is the Shapley-value [13] which distributes the value of the coalition $v(S)$ according to the average marginal contributions each member provides on all possible sub-coalitions they can be a part of. A key condition that enables the use of Shapley values is that $v(S)$ is independent of the actions of non-members $N \setminus S$. This condition is known as *no-externalities*. **P3** shows that coalitions in our setup indeed have to contend with externalities.

We can characterize coalition games that satisfy **P1**, **P2** and no-externalities with a single function $v : 2^N \mapsto \mathbb{R}$ that assigns value to any of the 2^N subsets of the grand coalition. For this reason such games are known as Characteristic Function Form (CFF) [14].

Coalitional games with externalities (such as ours), on the other hand, are characterized by a value function $v : 2^N \times \Pi(N) \mapsto \mathbb{R}$. Provided that $v(S, P)$ is defined for a coalition $S \in 2^N$ and a partitioning $P \in \Pi(N)$ of agents, and $S \in P$; $v(S, P)$ assigns value to any coalition S under partitioning P. This extra dependency on a partitioning ensures that externalities are accounted for when assigning value. Such games are known as Partition Function Form (PFF) [15].

We provide an informal description of the algorithm in Algorithm 1 which explains the two stages of the coalition formation. In the insider stage, every agent will get an opportunity to become the

Algorithm 1 Coalition Formation Game (informal) (see Algorithm 2 for the formal version)

Initialize: Player set N, insiders $S = N$, outsiders $O = \emptyset$
Insiders Stage:
while $S \neq \emptyset$ **do**
    Select proposer $p \in S$ via multibidding
    Proposer p makes benefit-sharing proposal to S
    if proposal accepted **then**
        Coalition S forms and **terminate**
    else
        Move p to outsiders: $O \leftarrow O \cup \{p\}$, $S \leftarrow S \setminus \{p\}$
    end if
end while
Outsiders Stage:
if some proposals were rejected **then**
    Randomly select a partition that includes the insider coalition
    Identify which coalition contains the last rejected proposer
    For each multi-player coalition (except the insider coalition):
        Choose a random proposer within that coalition
        Exception: The last rejected proposer cannot be chosen as proposer
            in their own coalition (if it has multiple members)
    Play coalition formation games sequentially:
        Start with the coalition containing the last rejected proposer
        Then proceed with remaining coalitions in arbitrary order
    Within each coalition game:
        The chosen proposer makes payment offers to all other members
        Each member decides sequentially whether to accept or reject
        If all accept: The coalition forms and payments are made
        If someone rejects: The rejector becomes a singleton coalition
        The remaining members continue playing with the same proposer
end if
Result: A final partition consisting of the insider coalition, any coalitions successfully formed in the games, and singleton players

"proposer" of a fair split of the coalition revenue. The proposer is chosen via a process known as multibidding by Macho-Stadler, Pérez-Castrillo, and Wettstein [16] where all agents submit their bids (how much each agent should pay to every agent), and then the individual who has received the largest sum of bids becomes the proposer. If the first proposal is accepted, we have generated a coalition of all agents—also known as a grand coalition. If not, then the agent is put into the outsider circle and multibidding restarts once again, until a coalition with an accepted proposal is found. The outsider stage is discussed in more detail in Algorithm 1. A key point to remember is that bids in the insider stage are *binding* which means that regardless of the coalition formation that results, agents are committed to pay that initial bid. This fact coupled with the fact that the outsiders' game determines the outside option for the proposer, encourages the proposer in the insiders stage to make acceptable proposals. We defer the detailed description of the Macho-Stadler, Pérez-Castrillo, and Wettstein [17] value, and the coalition formation game that achieves it to Algorithm 2 in Appendix F and provide a table of the outcomes of the game in Table 2.

5 Conclusion

In conclusion, to build truly generalist large language models, it is essential to incentivize collaboration through fair and transparent revenue-sharing mechanisms. Leveraging cooperative game theory and specifically Shapley-inspired solutions, our proposed mechanism ensures that stakeholders—data owners, model developers, and compute providers—share surplus revenues in a fair and incentive-compatible manner. Implementing these collaborative frameworks into model-hosting platforms can transform the current siloed competition into productive cooperation, significantly advancing the development of universally capable and socially beneficial language technologies.

References

[1] W.-L. Chiang, L. Zheng, Y. Sheng, *et al.*, *Chatbot arena: An open platform for evaluating llms by human preference*, 2024. arXiv: `2403.04132 [cs.AI]`. [Online]. Available: `https://arxiv.org/abs/2403.04132`.

[2] S. M. Park, K. Georgiev, A. Ilyas, G. Leclerc, and A. Madry, "Trak: Attributing model behavior at scale," in *International Conference on Machine Learning (ICML)*, 2023.

[3] M. Chen, J. Tworek, H. Jun, *et al.*, "Evaluating large language models trained on code," *arXiv preprint arXiv:2107.03374*, 2021.

[4] G. Son, H. Lee, S. Kim, *et al.*, *Kmmlu: Measuring massive multitask language understanding in korean*, 2024. arXiv: `2402.11548 [cs.CL]`. [Online]. Available: `https://arxiv.org/abs/2402.11548`.

[5] R. Xu, Z. Wang, R.-Z. Fan, and P. Liu, *Benchmarking benchmark leakage in large language models*, 2024. arXiv: `2404.18824 [cs.CL]`. [Online]. Available: `https://arxiv.org/abs/2404.18824`.

[6] S. Lin, J. Hilton, and O. Evans, *Truthfulqa: Measuring how models mimic human falsehoods*, 2022. arXiv: `2109.07958 [cs.CL]`.

[7] OpenAI. "Pricing." (), [Online]. Available: `https://openai.com/api/pricing/` (visited on 05/22/2025).

[8] I. Ong, A. Almahairi, V. Wu, *et al.*, "Routellm: Learning to route llms from preference data," in *The Thirteenth International Conference on Learning Representations*, 2024.

[9] Q. J. Hu, J. Bieker, X. Li, *et al.*, "Routerbench: A benchmark for multi-llm routing system," *arXiv preprint arXiv:2403.12031*, 2024.

[10] D. Stripelis, Z. Xu, Z. Hu, *et al.*, "Tensoropera router: A multi-model router for efficient llm inference," in *EMNLP (Industry Track)*, 2024.

[11] L. Chen, J. Q. Davis, B. Hanin, *et al.*, "Optimizing model selection for compound ai systems," *arXiv preprint arXiv:2502.14815*, 2025.

[12] M. Maschler, E. Solan, and S. Zamir, *Game Theory*, 1st ed. Cambridge University Press, Mar. 21, 2013, ISBN: 978-0-511-79421-6 978-1-107-00548-8. DOI: `10.1017/CBO9780511794216`. [Online]. Available: `https://www.cambridge.org/core/product/identifier/9780511794216/type/book` (visited on 08/10/2023).

[13] L. S. Shapley, "Notes on the n-person game—ii: The value of an n-person game," 1951.

[14] L. Á. Kóczy, *Partition Function Form Games* (Theory and Decision Library C). Cham: Springer International Publishing, 2018, vol. 48, ISBN: 978-3-319-69840-3 978-3-319-69841-0. DOI: `10.1007/978-3-319-69841-0`. [Online]. Available: `http://link.springer.com/10.1007/978-3-319-69841-0` (visited on 05/06/2025).

[15] R. M. Thrall and W. F. Lucas, "N-person games in partition function form," *Naval Research Logistics Quarterly*, vol. 10, no. 1, pp. 281–298, 1963. DOI: `https://doi.org/10.1002/nav.3800100126`. [Online]. Available: `https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800100126`.

[16] I. Macho-Stadler, D. Pérez-Castrillo, and D. Wettstein, "Efficient bidding with externalities," *Games and Economic Behavior*, vol. 57, no. 2, pp. 304–320, Nov. 1, 2006, ISSN: 0899-8256. DOI: `10.1016/j.geb.2005.12.002`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0899825605001636` (visited on 05/20/2025).

[17] I. Macho-Stadler, D. Pérez-Castrillo, and D. Wettstein, "Sharing the surplus: An extension of the shapley value for environments with externalities," *Journal of Economic Theory*, vol. 135, no. 1, pp. 339–356, Jul. 2007, ISSN: 00220531. DOI: `10.1016/j.jet.2006.05.001`. [Online]. Available: `https://linkinghub.elsevier.com/retrieve/pii/S0022053106000780` (visited on 05/20/2025).

[18] P. Duetting, V. Mirrokni, R. Paes Leme, H. Xu, and S. Zuo, "Mechanism design for large language models," in *Proceedings of the ACM Web Conference 2024*, 2024, pp. 144–155.

[19] H. Sun, Y. Chen, S. Wang, W. Chen, and X. Deng, "Mechanism design for llm fine-tuning with multiple reward models," in *Pluralistic Alignment Workshop in Neural Information Processing Systems (NeurIPS)*, Dec. 2024. [Online]. Available: `https://www.microsoft.com/en-us/research/publication/mechanism-design-for-llm-fine-tuning-with-multiple-reward-models/`.

[20] D. Cheng, J. Bae, J. Bullock, and D. Kristofferson, "Training data attribution (tda): Examining its adoption & use cases," *arXiv preprint arXiv:2501.12642*, 2025.

[21] S. K. Choe, H. Ahn, J. Bae, *et al.*, "What is your data worth to gpt? llm-scale data valuation with influence functions," *arXiv preprint arXiv:2405.13954*, 2024.

[22] J. Bae, W. Lin, J. Lorraine, and R. Grosse, "Training data attribution via approximate unrolling," in *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, *et al.*, Eds., vol. 37, Curran Associates, Inc., 2024, pp. 66 647–66 686. [Online]. Available: `https://proceedings.neurips.cc/paper_files/paper/2024/file/7af60ccb99c7a434a0d9d9c1fb00ca94-Paper-Conference.pdf`.

[23] Guardian, "OpenAI signs multi-year content partnership with condé nast," *The Guardian*, Aug. 20, 2024, ISSN: 0261-3077. [Online]. Available: `https://www.theguardian.com/technology/article/2024/aug/20/conde-nast-open-ai-deal` (visited on 05/22/2025).

[24] Reuters, "OpenAI signs content deals with the atlantic and vox media," *Reuters*, May 29, 2024. [Online]. Available: `https://www.reuters.com/business/media-telecom/openai-signs-content-deals-with-atlantic-vox-media-2024-05-29/` (visited on 05/22/2025).

[25] O. Skibski, T. Michalak, Y. Sakurai, M. Wooldridge, and M. Yokoo, "A graphical representation for games in partition function form," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, no. 1, Feb. 16, 2015, ISSN: 2374-3468, 2159-5399. DOI: `10.1609/aaai.v29i1.9306`. [Online]. Available: `https://ojs.aaai.org/index.php/AAAI/article/view/9306` (visited on 05/20/2025).

[26] O. Skibski, T. Michalak, and M. Wooldridge, *Marginality approach to shapley value in games with externalities*, Rochester, NY, Aug. 1, 2013. DOI: `10.2139/ssrn.2311112`. [Online]. Available: `https://papers.ssrn.com/abstract=2311112` (visited on 05/22/2025).

A Related Work

Mechanism Design for LLMs. Mechanism design has emerged as a powerful framework for aligning incentives in AI systems [18]. A notable example is the work by Sun, Chen, Wang, *et al.* [19], which proposes a "Token Auction Model" for LLM fine-tuning with multiple reward models. Their approach treats LLMs as resources to be allocated via an auction, where bidders (users or entities) compete to access model responses based on their valuation. This zero-sum game aims to maximize revenue for the LLM operator by selling tokens to the highest bidder. In contrast, our work pursues a cooperative game-theoretic approach, focusing on revenue sharing to unlock superior model performance through collaboration among LLM operators (in API routing) or domain experts (in model MoErging). While their auction model emphasizes competitive bidding, our revenue-sharing mechanism encourages stakeholders to pool resources—data, weights, or compute—to create a composite model that Pareto-dominates individual models, with fair remuneration ensuring sustained cooperation. Furthermore, their agents are bidders and a single bid collector, whereas our agents include diverse stakeholders with complementary resources, fostering a non-zero-sum outcome where collaboration enhances overall performance.

Data Attribution in LLMs. Data attribution techniques aim to quantify the contribution of individual data sources to a model's performance [20]–[22]. For instance, Park, Georgiev, Ilyas, *et al.* [2] proposes TRAK, a method to attribute model behavior to specific training data points, which is computationally intensive and primarily used for post-hoc analysis. While data attribution shares our goal of fairly recognizing contributions, it differs significantly in scope and application. Data attribution focuses on tracing model outputs to specific data points, often requiring fine-grained analysis that is infeasible in latency-sensitive settings. Our work, conversely, operates at a coarser level, leveraging public benchmarks to evaluate model performance and distribute revenue without needing to trace individual data contributions. This distinction allows our framework to be practical for real-time applications like API routing or model merging, avoiding the computational overhead of data attribution.

B Discussion

B.1 Compression's Perspective

Models or experts can be regarded as compressed representations of proprietary training data and computational resources. The process of forming coalitions can be understood as a way to integrate this information effectively. Rather than viewing models as isolated data points, we should see them as valuable clusters that aggregate data and compute into a meaningful asset. This perspective highlights their collective worth, enabling revenue sharing based on model performance, which reflects the combined value of the underlying resources.

B.2 Challenges and Open Questions

Several significant challenges remain unresolved in designing mechanisms for collaboration among Large Language Model (LLM) providers.

One critical issue is how to effectively account for the dynamic entry and exit of agents over time. As agents enter or leave coalitions, maintaining fair and incentive-compatible revenue-sharing agreements becomes increasingly complex. This dynamism necessitates mechanisms that can adapt efficiently to changing coalition compositions without extensive renegotiation or instability. Future research must develop robust yet flexible solutions to ensure fairness and continued participation despite evolving agent networks.

Another major concern is the risk of centralization due to coalition formation. Coalitions, while beneficial for leveraging complementary resources, could inadvertently foster monopolistic behaviors or create vulnerabilities due to coordinator defection. Such centralization risks not only affect market fairness but could also degrade overall system resilience. It is crucial to design governance structures and regulatory frameworks that mitigate these risks, ensuring decentralized power distribution and safeguarding against potential abuses by dominant actors.

Moreover, current evaluation frameworks like Chatbot Arena [1] have faced criticism for implicitly encouraging leaderboard gaming, potentially overshadowing genuine innovation. Hidden dynamics and undisclosed evaluation criteria may distort rankings, misleading stakeholders about true model performance and capability. To address this issue, the community needs more transparent, fair, and meaningful benchmarks. Alternative evaluation strategies could include openly shared performance metrics, continuous and dynamic evaluation processes, and incentive structures that genuinely reflect and reward innovative contributions rather than superficial performance gains.

Addressing these challenges requires an integrated approach combining principles from cooperative game theory, adaptive mechanism design, and transparent evaluation practices, thus ensuring sustainable, equitable, and productive collaborations in the evolving LLM ecosystem.

C Collaboration Paradigms

Table 1: Comparison of Model Merging, Mixture of Experts (MoE), and API Routing

Paradigm	Aspect	Details	Mathematical Formulation
Model Merging	Marginal Utility	*Compositional Generalization*: Merged model excels on unseen tasks when models complement each other, yielding superlinear utility.	$p_{\text{merge}}(t) > \max(p_1(t), p_2(t)) + \epsilon, \quad \epsilon > 0$
		Interference: Knowledge conflicts may degrade performance, resulting in negative utility.	$p_{\text{merge}}(t) < \min(p_1(t), p_2(t))$
		Independent: Orthogonal capabilities yield positive but linear utility, approximating a simple combination.	$p_{\text{merge}}(t) \approx \max(p_1(t), p_2(t))$
	Marginal Cost	*Training Cost*: Merging involves fine-tuning or weighted averaging, reducing costs compared to training a new model from scratch.	$c_{\text{train, merge}} < c_{\text{train, new}}$
		Inference Cost: Merged model size aligns with a single model, keeping inference costs comparable.	$c_{\text{infer, merge}} \approx c_{\text{infer, single}}$
Mixture of Experts (MoE)	Marginal Utility	Integrates multiple expert models via a gating mechanism, dynamically selecting the best expert per input, approximating maximum performance across experts.	$p_{\text{MoE}}(t) \approx \max_k p_k(t)$
	Marginal Cost	*Training Cost*: Training multiple experts and a gating network increases costs compared to a single model.	$c_{\text{train, MoE}} > c_{\text{train, single}}$
		Inference Cost: Only a subset of experts is activated, reducing costs proportional to the number of activated experts m out of K.	$c_{\text{infer, MoE}} \approx \frac{m}{K} c_{\text{infer, single}}$
API Routing	Marginal Utility	Distributes queries based on strategies (e.g., average performance or task fit), yielding a weighted average performance across K models.	$P_{\text{routing}}(t) = \sum_k w_k p_k(t), \quad \sum w_k = 1$
	Marginal Cost	*API Call Cost*: Costs reflect a weighted sum of individual model API costs; providers can minimize costs by favoring lower-cost models when performance is comparable.	$c_{\text{API, routing}} = \sum_k w_k c_{\text{API},k}$

D Evaluating the Value of Collaboration with Different Strategies

Coalitions can compute their revenue through various strategies:

Max Pooling: Select the best performing model per domain,

Averaging: Use an ensemble or arithmetic mean of member model predictions,

Model Merging: Integrate models to reduce redundant fine-tuning costs.

E Solution Concepts

Solution concepts for Characteristic Function Form (CFF) games. Two broad categories of solution concepts used for coalition games are based on notions of stability and (value allocation) fairness.

For CFFs, the canonical stability-based solution concept is the *"core"* which is the set of all value assignments that are i) *efficient* (no value left unassigned), ii) *individually rational* (the value assigned to the agent is at least as big as the value it could have achieved on its own), and iii) *coalitionally rational* (the value assigned to any (sub-)coalitions of agents is at least as big as what the sub-coalition can achieve on its own).

The canonical fairness-based solution concept is the *Shapley-value* which we introduced earlier. We note that Shapely arrived at this solution concept through a number of reasonable axioms: i) *anonymity* (re-labeling of the agents should not change their value assignments), ii) *efficiency* (as discussed earlier), iii) *null agent property* (if a player does not change the value of *any* coalition it joins, it should not get any value), iv) *additivity* (the sum of the values given by two games defined for the same N agents, should be equal to value of the game that combines the two value functions), v) *symmetry* (if two agents provide the same value to every coalition they join, their assigned value should be the same).

Choosing an appropriate solution concept for our setting. The stability notions take a more non-cooperative stance and try to avoid members leaving the coalitions; the fairness-based notions on the other hand assume all agents have the desire to stay within the coalition provided that the value of the coalition is fairly distributed.

In our setting, the coalitions are often formed as strategic alliances between various stakeholders such data providers (New York Times, Vox Media, etc.), model producers (OpenAI, Anthropic, etc.), and others. These agreements are often officiated in the form of *binding contracts* [23], [24] where the threat of leaving the coalition (as understood in stability-based notions) is not very credible. Furthermore, these contracts that implement the coalition formation are often made with the goal of fairness in revenue sharing to begin with. Therefore, we choose to use an extension to the fairness-based Shapley value, known as the Macho-Stadler value [17].

Macho-Stadler, Pérez-Castrillo, and Wettstein [17] value has a few useful properties that makes it suitable to be our coalition formation game. First, Macho-Stadler, Pérez-Castrillo, and Wettstein [17] value is one of the few Shapley value extensions that admit a compact graphical representation introduced by Skibski, Michalak, Sakurai, *et al.* [25], which has time complexity of $O(|N| \times |\mathcal{T}|)$ where $|N|$ is the number of agents and $|\mathcal{T}|$ is the size of the representation.

More importantly, Macho-Stadler, Pérez-Castrillo, and Wettstein [17] value admits an *implementation* is subgame perfect equilibria [16]. Given our choice of a fairness-based solution concept rather than a stability-based one, this property is very important because it ensures that the values assigned by the method are achievable at the equilibrium (i.e. steady state) of a non-cooperative game where coalitions are formed *over time* by rational agents seeking to optimize their own outcomes. Subgame perfectness ensures that the values are also achievable in any truncated history of the game (i.e., a subgame)—meaning, agents have no incentive to deviate from the agreement at any point in time.

F Coalition Formation Algorithm

Macho-Stadler, Pérez-Castrillo, and Wettstein [17] introduce an "average method" in which they construct a new game in which the contributions of each coalition to other coalitions are averaged; and then they define their value as the Shapley value of this constructed game. Concretely, under additional axiom of strong symmetry (i.e. exchanging the names of agents that are external to coalition S should not change the value received by members of S), they construct an associated average game $(N, \hat{v})$ to the PFF (N, v) assigning each coalition $S \subseteq N$ an average worth $\hat{v}(S) \equiv \sum_{P \ni S, P \in \mathcal{P}} \alpha(S, P) v(S, P)$ where $\alpha(S, P)$ are the 'weights' of the partition P in the computation of value of the coalition $S \in P$ and $\sum_{P \ni S, P \in \Pi(N)} \alpha(S, P) = 1$. Skibski, Michalak, and Wooldridge [26] later argued that if weights αs are taken as probabilities, they admit a Chinese restaurant process interpretation: an agent is more likely to transfer to a bigger coalition than a smaller one.

In the rest of this section, we present the detailed version of Algorithm 1. Table 2 discusses the five different outcomes of the game for the agents involved.

Algorithm 2 Coalition Formation using Multibidding [16]

Initialize: Set of agents N , insiders $S \leftarrow N$, round index $t \leftarrow 1$
while $|S| > 1$ **do**
  Insiders Stage $I(S)$:
  Each agent $j \in S$ submits a bid $b_j^i \in \mathbb{R}$ for each $i \in S$ such that $\sum_j b_j^i = 0$
  For each $i \in S$, compute $B_i = \sum_j b_i^j$
  Choose proposer $\gamma_S = \arg\max_i B_i$ (tie-break arbitrarily)
  γ_S pays $b_j^{\gamma_S}$ to each $j \in S$ and receives B_{γ_S}
  γ_S proposes offer $x_i^{\gamma_S} \in \mathbb{R}$ to each $i \in S \setminus \{\gamma_S\}$
  for each $i \in S \setminus \{\gamma_S\}$ (sequentially) **do**
    if i rejects the offer **then**
      $S \leftarrow S \setminus \{\gamma_S\}$, γ_S becomes outsider
      Continue to next round with updated S
    end if
  end for
  All agents accept, proposer γ_S pays $x_i^{\gamma_S}$ to each $i \in S \setminus \{\gamma_S\}$
  Terminate with final coalition S
end while
if $S = \emptyset$ or only one agent remains **then**
  Outsiders Stage $O(S)$:
  Define $O = N \setminus S$, initialize γ_{S+1} as the last rejected proposer
  Draw partition P of N including S with probability $\alpha(S, P)$
  Let $T_{S+1} \in P$ be the coalition containing γ_{S+1}
  Select proposer $\beta(T_{S+1}) \neq \gamma_{S+1}$
  Play Game $G(T_{S+1})$:
  $\beta(T_{S+1})$ proposes $x_i^{\beta(T_{S+1})}$ to all $i \in T_{S+1} \setminus \{\beta(T_{S+1})\}$
  for each $i \in T_{S+1} \setminus \{\beta(T_{S+1})\}$ (sequentially) **do**
    if agent i rejects **then**
      All agents in $T_{S+1} \setminus \{i\}$ play again with $\beta(T_{S+1})$ as singleton
      Continue to next $G(T)$
    end if
  end for
  All accept, proposer pays $x_i^{\beta(T_{S+1})}$ to each i
  Coalition T_{S+1} forms
  Repeat $G(T)$ for other $T \in P \setminus \{T_{S+1}\}$ in arbitrary order
  Partition $P(S)$ is finalized
end if

Agent Type	Conditions	Final Outcome/Payoff
Agent i	$i \in S^* \setminus \{\gamma_{s^*}\}$	$\boldsymbol{x}_i^{\gamma_{s^*}} + \sum_{k=s^*}^n \left(-b_{\gamma_k}^i + B_{\gamma_k}/k\right)$
Agent γ_{s^*}	γ_{s^*} is the proposer of S^*	$v\left(S^*, P^*\right) - \sum_{i \in S^* \setminus \{\gamma_{s^*}\}} x_i^{\gamma_{s^*}} + \sum_{k=s^*}^n \left(-b_{\gamma_k}^{\gamma_{s^*}} + B_{\gamma_k}/k\right)$
Outsider γ_m	$\{\gamma_m\} \in P^*$	$v\left(\{\gamma_m\}, P^*\right) + \sum_{k=m}^n \left(-b_{\gamma_k}^{\gamma_m} + B_{\gamma_k}/k\right)$
Outsider γ_m	$\gamma_m \neq \beta\left(T_m\right)$	$x_{\gamma_m}^{\beta(T_m)} + \sum_{k=m}^n \left(-b_{\gamma_k}^{\gamma_m} + B_{\gamma_k}/k\right)$
Outsider γ_m	$\gamma_m = \beta\left(T_m\right)$	$v\left(T_m, P^*\right) - \sum_{i \in T_m \setminus \{\gamma_m\}} x_i^{\gamma_m} + \sum_{k=m}^n \left(-b_{\gamma_k}^{\gamma_m} + B_{\gamma_k}/k\right)$

Table 2: **Coalition Game Outcomes by Agent Type.** S^* is the coalition of insiders. $P^* = P(S^*)$ is the final partition formed. The outsiders are $N \setminus S^* = \{\gamma_m\}_{m=s^*+1,\ldots,n}$. T_m is the coalition in P^* containing agent γ_m, $\beta(T_m)$ is the proposer in that coalition.