Sample, Align, Synthesize: Graph-Based Response Synthesis with CONGRS

Sayan Ghosh, Shahzaib Saqib Warraich, Dhruv Tarsadiya, Gregory Yauney, Swabha Swayamdipta
Thomas Lord Department of Computer Science
University of Southern California
Los Angeles, CA, USA
{ghoshsay,warraich,tarsadiy,yauney,swabhas}@usc.edu

Abstract

Inference-time scaling methods improve language model performance, but existing methods lack the efficiency and flexibility to synthesize information across multiple long-form generation samples. We introduce Consensus Graphs (CONGRS), a flexible DAG-based data structure that represents shared content and semantic variation across a set of LM responses to the same prompt. We construct CONGRS using an efficient lexical sequence alignment algorithm from bioinformatics, supplemented by the targeted usage of a secondary LM judge. We design and evaluate task-dependent decoding methods to synthesize final responses from CONGRS. Our experiments show that synthesizing responses from CONGRS improves factual precision on a biography generation task by up to 31% over an average response and reduces reliance on LM judges by more than 80% compared to other methods. We apply our approach to the MATH and AIME reasoning tasks and find an improvement over self-verification and majority vote baselines by up to 6 points of accuracy. CONGRS efficiently encode the variation among responses, which can then be used to improve downstream performance on various tasks.

1 Introduction and Related Work

Inference-time scaling methods improve language model performance by generating more tokens. This can occur serially within a model's chain-of-thought as with reasoning models [1, 2], or in parallel by sampling multiple response generations for a single query [3, 4]. Such parallel methods typically take the majority vote over final short answers [4] or select a single best response [5, 6]. However, selecting the single best parallel response can result in excluding valuable information, while serial methods can generate many redundant tokens [7]. Existing methods to aggregate information across multiple parallel long-form response generations often incur high costs by relying heavily on LMs to decompose responses into smaller units [8–11]. Thus, we ask: How can we efficiently synthesize information from multiple responses in long-form generation?

To answer this question, we exploit recent findings about the lack of textual diversity in generations from post-trained models. Specifically, RLHF alignment reduces the textual diversity of responses [12–14], resulting in *anchor spans*: sequences of words that occur in the same order across responses [15]. For example, instruction-tuned QWEN 2.5 72B responses to a query from a biography generation task share on average 7 anchor spans of 7.7 words each (details in §2).

We introduce Consensus Graphs (CONGRS), a data structure to capture similarities and differences within a set of sampled LM responses (Figure 1). Our work is inspired by the task of finding common regions in biological sequence data in bioinformatics, called Multiple Sequence Alignment (MSA). We use a two-step process to construct a directed acyclic graph of response text, where each path

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

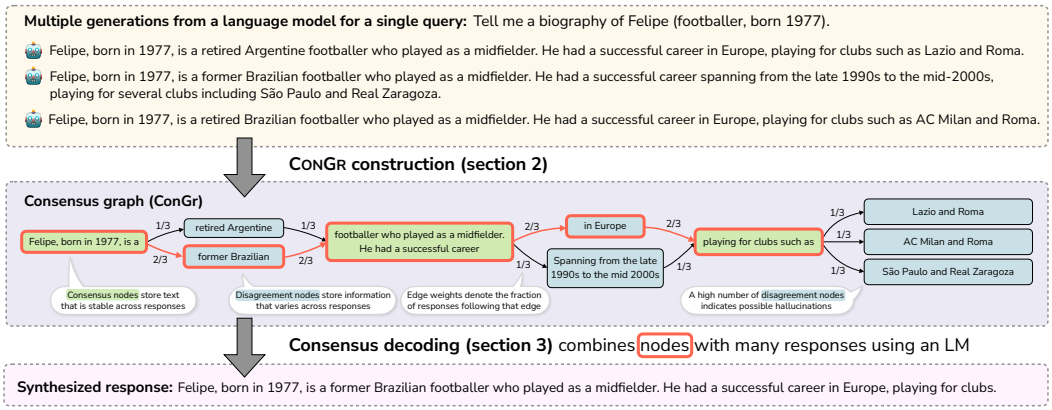

Figure 1: Consensus Graphs (CONGRS) capture the variation in a set of sampled LM responses. A CONGR is a weighted DAG that contains: consensus nodes that encode anchor spans of text present in all responses in the same order and disagreement nodes that encode differences between responses. A node's weighted-degree represents how many responses contain the information in that node. This information can be used differently for different tasks. For a task of improving factuality when generating biographies, disagreement nodes with lower weighted degree might be hallucinations.

through the graph corresponds to a single response. First, we adapt the Needleman-Wunsch algorithm [16, 17], a foundational MSA algorithm, to efficiently identify anchor spans shared by all responses (consensus nodes in Figure 1). We then use a secondary LM judge only to analyze whether the remaining parts of the responses, i.e., lexically different spans, are also semantically different, within the context of the graph (disagreement nodes in Figure 1). CONGRS capture agreement between model responses at various levels, from sentences to single words, going beyond approaches that only consider full-sentence agreement [18].

In real-world contexts, disagreements between traditional information sources can signal unreliability: information appearing in many sources may be more likely to be true [19, 20]. We study two tasks where high variation across responses might indicate model errors: biography factuality and mathematical reasoning. We devise two approaches to synthesize a response from a CONGR; each approach is tailored to the nature of its task. For biography factuality, we develop *consensus decoding*, which selectively aggregates text across multiple responses with the goal of removing hallucinations. A single biography can contain facts that need not logically rely on each other, so consensus decoding seeks to combine the most reliable facts from across responses. On the other hand, a reasoning task requires an argument that is logically consistent from start to end. We therefore develop *guided self-verification* to identify possible errors in existing reasoning responses instead of combining text from different responses. Standard self-verification, where a model evaluates its own generations for correctness, is a promising alternative to relying on a secondary LM judge, but current models struggle to self-verify on reasoning tasks [21] unless error locations are provided [22]. We use CONGRS to localize possible errors by finding high-variability regions across responses.

For the task of generating factual biographies, consensus decoding improves factual precision by up to 31% over an average response. It also achieves performance comparable to that of an alternative method that heavily uses LMs for claim analysis [8] at less than 20% of the cost in secondary LM API calls. For reasoning tasks, guided self-verification boosts performance on MATH by up to 6 points of accuracy beyond self-consistency and standard self-verification. CONGRS can also drastically improve refusal for questions without factual answers (Appendix I). Our approach efficiently identifies information that varies across model responses and shows that using this variation as a proxy for unreliability can improve performance on diverse downstream tasks. Future work will explore how variation can also encode alternate perspectives and creative possibilities for more open-ended tasks.

2 Capturing LM Response Variation with Consensus Graphs

Consensus Graphs (CONGRS) capture the lexical and semantic variation across a set of model responses to a prompt. We briefly describe how to construct CONGRS; a complete formal description

of the data structure and the construction process is provided in Appendix C. Given a set of responses sampled from an LM in response to a prompt, a CONGR is a weighted directed acyclic graph. Nodes in this graph are partitioned into *consensus nodes* and *disagreement nodes*. Each response will correspond to a path through the graph. First, we use the Needleman-Wunsch algorithm [16] to align words across responses and organize response text into a DAG, using the approach of Lee et al. [17]. Each text span present in all responses becomes a *consensus node*. For example, the first consensus node in Figure 1 contains the text that begins all responses: *"Felipe, born in 1977, is a"*. Second, we use a secondary LM as a judge to determine if the parts of responses between consensus nodes are semantically equivalent. In Figure 1, the text spans between the first two consensus nodes are 1) *"retired Argentine"*, 2) *"former Brazilian"*, and 3) *"retired Brazilian"*. Two disagreement nodes are constructed: the first for span 1 and the second for spans 2 and 3. This process is repeated for the spans between each pair of successive consensus nodes. Finally, each edge in the graph is given a weight equal to the fraction of responses that pass through that edge. The edge weight between the first consensus node and the first disagreement node is ⅓ because 1 out of 3 responses contain that text, and the edge weight between the first consensus node and the second disagreement node is ⅔.

Descriptive statistics of CONGRS. For the three datasets we consider, responses share enough text spans to construct consensus nodes (Table 1). For example, a CONGR for the biography generation task has, on average, 7 consensus nodes consisting of 7.7 words each. For the same task, 28.0% of consensus nodes contain only stopwords, and only 1.8% of disagreement nodes contain only stopwords. Only 6 out of 100 examples yield responses with so much variability that no consensus nodes are made. For the mathematical reasoning tasks, MATH and AIME, there are no degenerate graphs containing zero consensus nodes. While reasoning solutions can diverge, there is consensus in the form of explicit mentions of solution steps and common intermediate calculations.

Table 1: Descriptive statistics for CONGRS constructed from five QWEN 2.5 72B responses per evaluation example, averaged across examples.

	Biographies	MATH	AIME
# nodes	27.50	47.22	133.37
% consensus nodes	26.8%	30.7%	24.5%
% disagreement nodes	73.2%	69.3%	75.5%
# branches from consensus nodes	2.39	2.29	3.21
# words in consensus nodes	7.70	2.67	3.68
# words in disagreement nodes	25.71	36.06	38.30

3 Synthesizing Responses from Consensus Graphs

The structure of a CONGR provides data about the variability in a set of model responses. We devise two approaches that use CONGRS in different ways to synthesize a response: *consensus decoding* and *guided self-verification*. Details and prompts are in Appendix F.

Consensus decoding. Consensus decoding generates a synthesized response from a CONGR by incorporating text present in many of the original responses. We start with a set R with $|R| = m$ responses and a hyperparameter consensus threshold $\tau \in [0, 1]$, where higher values mean that a text span must be present in more responses to make it into the final response. Consensus decoding traverses the CONGR in topological order and selects nodes with weighted degrees of at least τ, i.e., including text present in at least τm of the original responses. Then, the text in each selected node is concatenated to form a draft response. Since this can result in disfluencies, we prompt the same secondary LM that was used for CONGR construction to fix grammatical errors. This step also includes explicit instructions to abstain when the selected content is too fragmented to produce a coherent response. In practice, we find via manual inspection that this task is constrained enough that the secondary LM does not introduce (or remove) further hallucinations.

Guided self-verification. We design consensus decoding to combine text across responses since biographies can contain facts that need not logically rely on each other. For example, a person's birth city does not directly dictate that person's occupation. Reasoning chains require a globally coherent argument from start to end, so here we do not combine text spans across responses. Instead, we use the variation represented by a CONGR to mark locations of potential errors for self-verification. Given a set R with $|R| = m$ responses generated by model $\mathcal{M}$ and a pruning threshold hyperparameter $\kappa \in [0, 1]$, *guided self-verification* uses self-verification on partial responses to prune out incorrect responses. We initialize a set of candidate responses $\mathcal{C}$ with R and traverse the CONGR in a topological order, marking consensus nodes which are followed by at least κm disagreement nodes.

Table 2: Results for biography generation for larger models: FActScore, numbers of supported (#T) and unsupported (#F) facts, and response ratio (R, how often the model doesn't abstain). We report mean values over five replications. Standard deviation across runs is in small font.

	QWEN 2.5 72B				LLAMA 3.3 70B			
Method	**FActScore**	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**
Greedy	0.677 0.002	18.771 0.192	8.553 0.149	0.974 0.005	0.645 0.073	16.343 2.965	6.271 0.532	0.620 0.115
Mean of m	0.681 0.005	19.209 0.146	8.455 0.156	0.966 0.007	0.641 0.007	16.420 0.565	6.487 0.129	0.589 0.016
Shortest	0.689 0.009	19.018 0.338	8.121 0.258	0.952 0.004	0.692 0.023	16.989 0.551	4.931 0.427	0.482 0.027
LM consensus	0.693 0.010	20.925 0.330	8.979 0.448	0.974 0.005	0.687 0.014	19.966 0.299	6.587 0.318	0.618 0.019
MBR	0.696 0.006	19.469 0.097	7.984 0.333	0.962 0.007	0.676 0.014	17.505 0.359	5.779 0.592	0.566 0.017
QwQ 32B	0.562 0.003	19.830 0.283	15.073 0.196	0.974 0.006	0.562 0.003	19.830 0.283	15.073 0.196	0.974 0.006
ASC ($\Theta = 2$)	0.688 0.008	18.766 0.332	6.128 0.174	0.968 0.008	0.715 0.031	13.179 4.619	3.367 0.282	0.578 0.017
ASC ($\Theta = 3$)	0.733 0.013	12.880 0.602	2.988 0.366	0.882 0.015	0.796 0.018	9.483 3.662	1.933 0.241	0.486 0.014
CONGRS (τ=0.3)	0.715 0.014	20.784 0.379	6.007 0.509	0.924 0.008	0.827 0.005	22.226 1.120	2.722 0.264	0.462 0.023
CONGRS (τ=0.5)	0.795 0.021	17.462 0.358	2.970 0.270	0.846 0.021	0.845 0.004	18.395 1.157	1.814 0.148	0.430 0.014

These regions of high variability may correspond to different valid reasoning steps or to an error. To distinguish between these cases, we compare partial responses using self-verification: we provide $\mathcal{M}$ with each partial solution up to the next consensus node and ask if an error has occurred. If a partial response is said to contain an error, we prune that response from $\mathcal{C}$. After repeating this process at all uncertain nodes, we use the remaining responses in $\mathcal{C}$ as the context to prompt $\mathcal{M}$ to synthesize a final response.

4 Results

Consensus decoding improves factuality of long-form generations. We apply consensus decoding to the task of biography generation for 100 randomly sampled entities from FActScore [23]. We use the FActScore metric for evaluation, which measures the fraction of facts in a response deemed to be supported in a reference text by an LM judge [23]. We generate $m = 5$ responses each from the instruction-tuned models LLAMA 3.3 70B [24] and QWEN 2.5 72B [25] with temperature 0.9. We compare to: mean scores of the original responses, *Greedy decoding*, *MBR Decoding* [5], *Shortest response* [26], *Reasoning model response* from QwQ 32B [25], and an LM-generated consensus from the responses without a CONGR. We also compare to *Atomic Self-Consistency* (ASC, Thirukovalluru et al. 18), a synthesis method that uses sentence clustering. Additional models, method descriptions, and prompts are in Appendices G and H. We report mean results over five replications.

Table 2 shows that consensus decoding with CONGRS improves factuality over baselines. For instance, consensus decoding with a consensus threshold of $\tau = 0.3$ on Qwen increases mean FActScore over the original responses from 0.681 to 0.715 and increases the mean number of supported facts from 19.202 to 20.784. A threshold of $\tau = 0.5$ improves FActScore further at the expense of the number of supported facts. Consensus decoding has higher FActScores than the corresponding ASC setting (e.g., $\tau = 0.3$ corresponds to $\Theta = 2$), albeit with slightly lower response rates. When consensus decoding abstains, the original responses often had low factuality. For example, responses from QWEN 2.5 72B on average have a FActScore of 0.68, but the average FActScore of responses that lead consensus decoding (with $\tau = 0.3$) to abstain have a mean FActScore of only 0.31 (details in Appendix H, Table 6).

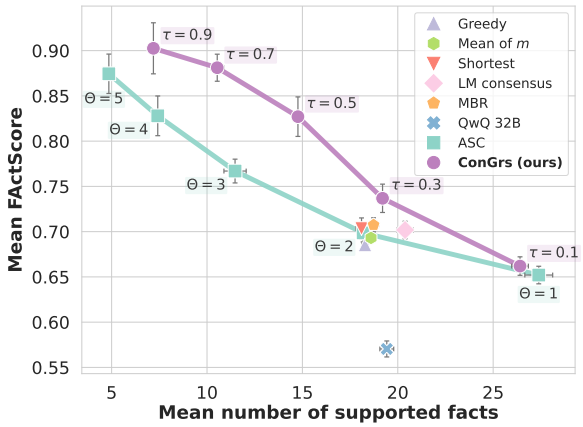

Figure 2: Consensus decoding with CONGRS achieves a better trade-off between FActScore (the fraction of a response's claims that are true) and the number of true claims. Up and right is better. Results for QWEN 2.5 72B. Error bars show standard deviation across runs.

Figure 2 shows that consensus decoding yields a better trade-off between FActScore and number of supported facts than baselines, even accounting for the differences in response rates (details in Appendix H). At nearly all settings of τ, consensus decoding is at the Pareto frontier of factual precision and factual recall. Choosing a selection threshold τ is a matter for the user to decide whether their use-case requires a) higher factuality and fewer facts or b) more facts at the expense of lower factuality.

Consensus decoding uses fewer tokens from secondary LMs than an alternative. Uncertainty-Aware Decoding (UAD) with Closeness Centrality is a successful approach to response synthesis for factuality that uses a secondary LM to split responses into claims [8]. Due to the cost of running UAD, we compare our method using a smaller evaluation set of 25 random entities. Consensus decoding uses less than 20% as many secondary LM API tokens (13,392 vs. 76,220 per entity on average) while achieving a slightly higher FActScore and the same response ratio. Details and full results are in Appendix H.

Guided self-verification from CONGRS improves reasoning performance. We evaluate our guided self-verification method using AIME 2024 [27] and the test split of MATH [28]. We generate $m = 5$ responses each from LLAMA 3.3 70B and QWEN 2.5 72B with a temperature of 0.9. We compare to: *Self-consistency* [4] and *Self-verification* [29, 30]. Appendix G contains details and prompts. We have only run one replicate for reasoning results due to computational constraints. Table 3 shows that for MATH, guided self-verification achieves 2 and 6 point accuracy gains over self-consistency for Qwen and Llama respectively, as well as larger gains against a standard self-verification baseline. On the more challenging AIME dataset, guided self-verification achieves a 4 point gain over self-consistency for Llama and no gain for Qwen. Consensus decoding in this setting does only as well as self-consistency on three out of four tasks, likely because it is also taking the majority vote of final answers. In all cases, guided self-verification outperforms or matches consensus decoding. Guided self-verification narrows the gap between self-consistency and Pass@m, demonstrating that comparing reasoning chains can improve performance.

Table 3: Guided self-verification with CONGRS gets higher accuracy than self-consistency and self-verification. Pass@m is an upper bound.

Method	QWEN 2.5 72B		LLAMA 3.3 70B	
	MATH	AIME	MATH	AIME
Self-verification	0.66	0.15	0.56	0.19
Self-consistency	0.68	0.2	0.59	0.23
Consensus decoding (τ=0.5)	0.68	0.2	0.61	0.23
Guided self-verification (κ=0.7)	0.70	0.2	0.65	0.27
Pass@m	0.74	0.2	0.7	0.33

Conclusion. We have introduced CONGRS, a DAG-based data structure that represents the variability in a set of LM responses. By combining fast lexical alignment with targeted use of secondary LMs, CONGRS can be constructed efficiently. We show how CONGRS can be used to synthesize information across responses and can be used to guide self-verification for reasoning problems. We plan to explore how CONGRS scale with more responses and the extent to which the structured information in CONGRS can be useful for more tasks.

References

[1] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[2] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

[3] Bradley Brown, Jordan Juravsky, Ryan Saul Ehrlich, Ronald Clark, Quoc V Le, Christopher Re, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2025. URL https://openreview.net/forum?id=0xUEBQV54B.

[4] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

[5] Amanda Bertsch, Alex Xie, Graham Neubig, and Matthew Gormley. It's MBR all the way down: Modern generation techniques through the lens of minimum Bayes risk. In *Big Picture Workshop at EMNLP*, December 2023. doi: 10.18653/v1/2023.bigpicture-1.9. URL https://aclanthology.org/2023.bigpicture-1.9/.

[6] Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal self-consistency for large language models. In *ICML 2024 Workshop on In-Context Learning*, 2024. URL https://openreview.net/forum?id=LjsjHF7nAN.

[7] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+3=? on the overthinking of o1-like LLMs. *arXiv preprint arXiv:2412.21187*, 2024.

[8] Mingjian Jiang, Yangjun Ruan, Prasanna Sattigeri, Salim Roukos, and Tatsunori Hashimoto. Graph-based uncertainty metrics for long-form language model generations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=YgJPQW0lkO.

[9] Ante Wang, Linfeng Song, Baolin Peng, Lifeng Jin, Ye Tian, Haitao Mi, Jinsong Su, and Dong Yu. Improving LLM generations via fine-grained self-endorsement. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 8424–8436, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.499. URL https://aclanthology.org/2024.findings-acl.499/.

[10] Xinglin Wang, Yiwei Li, Shaoxiong Feng, Peiwen Yuan, Boyuan Pan, Heda Wang, Yao Hu, and Kan Li. Integrate the essence and eliminate the dross: Fine-grained self-consistency for free-form language generation. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11782–11794, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.634. URL https://aclanthology.org/2024.acl-long.634/.

[11] Mohamed Elaraby, Mengyin Lu, Jacob Dunn, Xueying Zhang, Yu Wang, Shizhu Liu, Pingchuan Tian, Yuping Wang, and Yuxuan Wang. Halo: Estimation and reduction of hallucinations in open-source weak large language models. *arXiv preprint arXiv:2308.11764*, 2023. URL https://arxiv.org/abs/2308.11764.

[12] Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the effects of RLHF on LLM generalisation and diversity. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PXD3FAVHJT.

[13] Thom Lake, Eunsol Choi, and Greg Durrett. From distributional to overton pluralism: Investigating large language model alignment. In *Pluralistic Alignment Workshop at NeurIPS 2024*, 2024. URL https://openreview.net/forum?id=roe8GMahZL.

[14] Peter West and Christopher Potts. Base models beat aligned models at randomness and creativity. *arXiv preprint arXiv:2505.00047*, 2025. URL https://arxiv.org/abs/2505.00047.

[15] Margaret Li, Weijia Shi, Artidoro Pagnoni, Peter West, and Ari Holtzman. Predicting vs. acting: A trade-off between world modeling & agent modeling. *arXiv preprint arXiv:2407.02446*, 2024. URL https://arxiv.org/abs/2407.02446.

[16] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3): 443–453, 1970. URL https://www.sciencedirect.com/science/article/abs/pii/0022283670900574.

[17] Christopher Lee, Catherine Grasso, and Mark F. Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18(3):452–464, 03 2002. ISSN 1367-4803. doi: 10.1093/bioinformatics/18.3.452. URL https://doi.org/10.1093/bioinformatics/18.3.452.

[18] Raghuveer Thirukovalluru, Yukun Huang, and Bhuwan Dhingra. Atomic self-consistency for better long form generations. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12681–12694, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.706. URL https://aclanthology.org/2024.emnlp-main.706/.

[19] Don Fallis. Social epistemology and information science. *Annual review of information science and technology*, 40(1):475–519, 2006. URL https://asistdl.onlinelibrary.wiley.com/doi/10.1002/aris.1440400119.

[20] Alvin I Goldman. Social epistemology: Theory and applications. *Royal Institute of Philosophy Supplements*, 64:1–18, 2009.

[21] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024.

[22] Gladys Tyen, Hassan Mansoor, Victor Carbune, Peter Chen, and Tony Mak. LLMs cannot find reasoning errors, but can correct them given the error location. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13894–13908, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.826. URL https://aclanthology.org/2024.findings-acl.826/.

[23] Sewon Min, Kalpesh Krishna, Xinxi Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.741. URL https://aclanthology.org/2023.emnlp-main.741/.

[24] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

[25] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

[26] Alexandros Dimakis. Twitter post. https://x.com/AlexGDimakis/status/1885447830120362099, 2025. Accessed 10 May 2025.

[27] Mathematical Association of America MAA. AIME, February 2024. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions/.

[28] 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, 2021. URL https://arxiv.org/abs/2103.03874.

[29] Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.167. URL https://aclanthology.org/2023.findings-emnlp.167/.

[30] Eric Zhao, Pranjal Awasthi, and Sreenivas Gollapudi. Sample, scrutinize and scale: Effective inference-time search by scaling verification, 2025. URL https://arxiv.org/abs/2502.01839.

[31] Mirac Suzgun, Luke Melas-Kyriazi, and Dan Jurafsky. Follow the wisdom of the crowd: Effective text generation via minimum Bayes risk decoding. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4265–4293, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.262. URL https://aclanthology.org/2023.findings-acl.262/.

[32] Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=xbjSwwrQOe.

[33] Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

[34] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=5Xc1ecxO1h.

[35] Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. Branch-solve-merge improves large language model evaluation and generation. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8352–8370, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.462. URL https://aclanthology.org/2024.naacl-long.462/.

[36] Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, Min-Yen Kan, Junxian He, and Qizhe Xie. Self-evaluation guided beam search for reasoning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=Bw82hwg5Q3.

[37] Junlin Wang, Jue WANG, Ben Athiwaratkun, Ce Zhang, and James Zou. Mixture-of-agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=h0ZfDIrj7T.

[38] Lingjiao Chen, Jared Davis, Boris Hanin, Peter Bailis, Ion Stoica, Matei Zaharia, and James Zou. Are more llm calls all you need? towards the scaling properties of compound ai systems. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 45767–45790. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/51173cf34c5faac9796a47dc2fdd3a71-Paper-Conference.pdf.

[39] Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher Manning, and Chelsea Finn. Fine-tuning language models for factuality. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023. URL https://openreview.net/forum?id=kEK08VdSO5.

[40] Zorik Gekhman, Gal Yona, Roee Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. Does fine-tuning LLMs on new knowledge encourage hallucinations? In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7765–7784, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.444. URL https://aclanthology.org/2024.emnlp-main.444/.

[41] Katie Kang, Eric Wallace, Claire Tomlin, Aviral Kumar, and Sergey Levine. Unfamiliar finetuning examples control how language models hallucinate. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3600–3612, Albuquerque, New Mexico, April 2025. Association for

Computational Linguistics. ISBN 979-8-89176-189-6. URL https://aclanthology.org/2025.naacl-long.183/.

[42] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Th6NyL07na.

[43] Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale Fung, Mohammad Shoeybi, and Bryan Catanzaro. Factuality enhanced language models for open-ended text generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=LvyJX20Rll.

[44] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=zj7YuTE4t8.

[45] Roi Cohen, May Hamri, Mor Geva, and Amir Globerson. LM vs LM: Detecting factual errors via cross examination. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12621–12640, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.778. URL https://aclanthology.org/2023.emnlp-main.778/.

[46] Luke Yoffe, Alfonso Amayuelas, and William Yang Wang. Debunc: mitigating hallucinations in large language model agent communication with uncertainty estimations. *arXiv preprint arXiv:2407.06426*, 2024. URL https://arxiv.org/abs/2407.06426.

[47] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-verification reduces hallucination in large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3563–3578, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.212. URL https://aclanthology.org/2024.findings-acl.212/.

[48] Zizhong Wei, Dongsheng Guo, Dengrong Huang, Qilai Zhang, Sijia Zhang, Kai Jiang, and Rui Li. Detecting and mitigating the ungrounded hallucinations in text generation by llms. In *Proceedings of the 2023 International Conference on Artificial Intelligence, Systems and Network Security*, AISNS '23, page 77–81, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400716966. doi: 10.1145/3661638.3661653. URL https://doi.org/10.1145/3661638.3661653.

[49] Miaoran Li, Baolin Peng, Michel Galley, Jianfeng Gao, and Zhu Zhang. Self-checker: Plug-and-play modules for fact-checking with large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 163–181, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.12. URL https://aclanthology.org/2024.findings-naacl.12/.

[50] Loka Li, Zhenhao Chen, Guangyi Chen, Yixuan Zhang, Yusheng Su, Eric Xing, and Kun Zhang. Confidence matters: Revisiting intrinsic self-correction capabilities of large language models. *arXiv preprint arXiv:2402.12563*, 2024. URL https://arxiv.org/abs/2402.12563.

[51] Potsawee Manakul, Adian Liusie, and Mark Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.557. URL https://aclanthology.org/2023.emnlp-main.557/.

[52] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024. URL https://www.nature.com/articles/s41586-024-07421-0.

[53] Zhengping Jiang, Anqi Liu, and Benjamin Van Durme. Conformal linguistic calibration: Trading-off between factuality and specificity. *arXiv preprint arXiv:2502.19110*, 2025.

[54] Christopher Mohri and Tatsunori Hashimoto. Language models with conformal factuality guarantees. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[55] Abhilasha Ravichander, Shrusti Ghela, David Wadden, and Yejin Choi. HALoGEN: Fantastic LLM hallucinations and where to find them. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1402–1425, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.71. URL https://aclanthology.org/2025.acl-long.71/.

[56] Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

[57] Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, et al. Deepseek-r1 thoughtology: Let's< think> about llm reasoning. *arXiv preprint arXiv:2504.07128*, 2025. URL https://arxiv.org/abs/2504.07128.

A *n*-gram Overlap Experiment

Figure 3 shows that independently generated responses from aligned LMs have a high amount of n-gram overlap. To construct Figure 1, we randomly sample 50 named entities from the FActScore Biographies dataset and generate $m \in [5, 10, 20, 50]$ biographies from LLAMA 3.3 70B. The y-axis is averaged over all 50 entities.

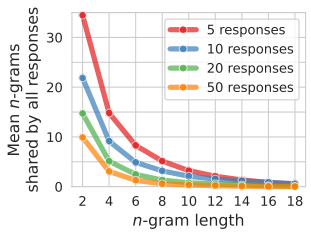

Figure 3: Independently generated preference-tuned LM responses to the same query share many n-grams.

B Additional Related Work

Inference-Time Scaling. Inference-time scaling strategies vary widely, with most selecting one response and only operating on short responses. Selection methods include Minimum Bayes Risk (MBR) decoding [5, 31], Universal Self-Consistency [6], and rejection sampling/Best-of-N sampling [32, 3, 33]. Other methods aggregate one model's responses using sophisticated intermediate generation structures [34–36] that decode partial sequences in parallel. Some approaches also aggregate across responses from multiple LMs [37, 38] through multi-agent debates. In comparison, our methods sample full independent responses from one model and then synthesize novel responses.

LM Response Factuality. Several methods improve long-form response factuality from LMs. Many involve additional fine-tuning [39], which has mixed results in terms of consistent improvement [40, 41]. Some methods involve only sampling one generation with novel decoding algorithms [42, 43]. Others rely on discussions between multiple LLMs [44–46] or self-correction strategies [47–51]. Some methods create claim-level uncertainty estimates based on multiple sampled generations and use them to filter out low-confidence claims [8, 51–54, 18, 9, 10].

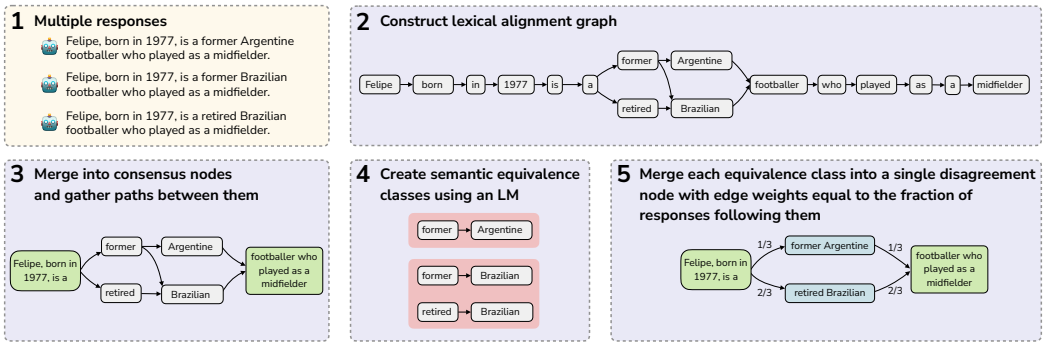

Figure 4: From a set of responses, we construct a CONGR by 1) Using Needleman-Wunsch [17] to construct a lexical DAG where each node's text is a single token, 2) Merging consecutive sequences of nodes that are present in all responses to create consensus nodes, 3) Extracting paths between consecutive pairs of consecutive nodes and using a LM to create semantic equivalence classes, 4) Creating a disagreement node for each semantic equivalence class.

C Detailed Consensus Graph Construction

Formal Definition. Given a set of responses $R = \{y_1, \ldots, y_m\}$ sampled from an LM $\mathcal{M}$ in response to a prompt x, a CONGR is a weighted directed acyclic graph $g(R) = (V, E, W)$. Each response corresponds to some path through the graph.

A CONGR consists of the following elements:

- $V = V_C \cup V_D \cup \{v_{\text{START}}, v_{\text{END}}\}$ is the set of nodes, where V_C represents consensus nodes, V_D represents disagreement nodes, and v_{START}, v_{END} are special nodes marking the beginning and end of all paths. We use v_{text} to refer to a node's text label.
- $E \subseteq V \times V$ is the set of directed edges
- $W : E \to [0, 1]$ is a weight function that indicates what fraction of original responses follow each edge. We use $d^w(v)$ to refer to a node's weighted degree and $d_{in}(v), d_{out}(v)$ to refer to unweighted in-degree and out-degree respectively, normalized by the number of responses m. While CONGRS are directed graphs, each node's weighted in and out-degree are enforced to be equal, with the exception being $\{v_{\text{START}}, v_{\text{END}}\}$.

Consensus nodes $c \in V_C$ contain the text that is consistent across all model responses (as shown in the green nodes in Figure 1). For all $c \in V_C$, $d^w(c) = 1$. These represent the anchor spans present across the set of responses. Disagreement nodes $v \in V_D$ contain information that varies across responses (as shown in the blue nodes in Figure 1). That is, they contain what is in between each anchor span. With these elements, CONGRS simply represent the regions of similarity and diversity across a range of responses. Because DAGs naturally induce partial orders on nodes, we can sort each node set by topological order and index so that we can represent each node as an ordered sequence: $V_C = c_1, \ldots, c_k$ and $V_D = v_1, \ldots, v_l$, which facilitates the decoding algorithms explained in §3.

Step 1: Generating lexical partial order graphs. The first step in constructing a consensus graph is finding common lexical subsequences (anchor spans) among a set of m model responses.

To do so, we adapt the Needleman-Wunsch algorithm [16], a dynamic programming-based algorithm originally developed to align biological sequence data. While Li et al. [15] also use MSA to perform sequence alignments, we perform several modifications to the standard versions of these algorithms. In particular, we use the approach of Lee et al. [17] to create a lexical DAG from the alignments produced by running Needleman-Wunsch. In addition, we adapt Needleman-Wunsch to process lexical token sequences as opposed to single-character sequences.[1] Further details are in Appendix D.

This step results in a weighted lexical DAG, with a node's weighted degree corresponding to the proportion of sequences containing that node. As seen Panel 2 of Figure 4, there is no distinction yet

[1] In this context, *tokens* are full words as opposed to the subwords commonly used by LM tokenizers for ease of decoding responses.

between consensus and disagreement nodes, and each node contains only one token. Nonetheless, this structure already begins to show where responses converge and diverge.

Step 2: Creating consensus nodes. To create consensus nodes, we merge sequences of nodes that are present in each generation. For each such sequence $s = v_1, v_2, ..., v_k$, we: 1) Create a consensus node c with c_{text} equal to the concatenation of all tokens in the sequence, 2) Replace each sequence with its corresponding consensus node, and 3) Preserve edge connections at the beginning and end of each sequence. Specifically, all edges that entered v_1 now enter c, and all edges that exited v_k now exit c. Panel 3 of Figure 4 shows how the graph appears at the end of this step.

Step 3: Creating disagreement nodes. Now, the graph contains consensus nodes $c_1, \ldots, c_k$ that include the anchor spans that are shared by all m responses. There are multiple directed paths between each pair of consecutive consensus nodes (c_i, c_{i+1}). Concatenating the text corresponding to each node in such a path $(v_{i_1})_{\text{text}} \circ \cdots \circ (v_{i_t})_{\text{text}}$ recovers a substring of a response. As demonstrated in Panel 3 of Figure 4, these paths represent how the original responses diverge between anchor spans. The final step in creating a CONGR is determining whether these paths between consensus nodes are semantically equivalent. We consider each pair of consecutive consensus nodes (c_i, c_{i+1}) and all paths between them. We use a secondary LM to perform pairwise comparisons between each path to create semantic equivalence classes, which are groups of paths judged to convey the same meaning despite different wording. The complete prompt templates and judgment criteria are provided in Appendix E. Panel 4 of Figure 4 shows a simple example of this process. For each equivalence class, we create a disagreement node $v \in V_d$ with one path's text chosen as v_{text}. Importantly, we preserve alternative phrasings as additional metadata in the node, in order to ensure none of original response information is lost. We then replace nodes that are in between each pair of consecutive consensus nodes with the newly created disagreement nodes, as shown in Panel 5 of Figure 4. This process results in an alternating structure of consensus and disagreement nodes.

D Needleman-Wunsch Hyperparameters

We apply the Needleman-Wunsch multiple sequence alignment algorithm from Lee et al. [17]. We create partial order graphs in an iterative fashion by aligning all sequences one by one. In our case; each node in the graph represents a token (full word). In order to align an incoming response to the intermediate partial order graph; we visit graph nodes in the order of the topological sort and consider all valid positions for insertion through a dynamic programming approach. In particular, we select an insertion which minimizes the total cost of aligning an incoming sequence to an intermediate lexical partial order graph. The cost of two aligned positions between the graph and the incoming sequence is determined by string similarity if both positions correspond to nodes; and by a gap penalty if one or both positions correspond to gaps. We use an affine gap penalty scheme as it is better suited for aligning our lexical sequences. We use the following hyper-parameters for our alignment: *gap_open_penalty* $= -1$, *gap_extend_penalty* $= -1$, *match_penalty* $= 1$, *mismatch_penalty* $= -2$.

E CONGRS Construction Prompts

Pseudocode for creating disagreement nodes is in Algorithm 1. Prompts for pairwise comparison to create equivalent classes with a secondary LM are below.

Algorithm 1: Creating Disagreement Nodes

Input: Set of m responses R, Graph $g(R) = (V, E, W)$ with consensus nodes $V_C = \{c_1, c_2, ..., c_k\}$
Output: Complete CONGR with disagreement nodes
$V_D \leftarrow \emptyset$
for *Each consecutive pair* (c_i, c_{i+1}) **do**
  $P_{i,i+1} \leftarrow \{p_1, p_2, ..., p_l\}$; `// Paths between` c_i `and` c_{i+1}
  for *each path* $p_j \in P_{i,i+1}$ **do**
    $p_{j_{\text{text}}} \leftarrow$ ""
    for *each node* $v \in p_j$ **do**
      $p_{j_{\text{text}}} \leftarrow p_{j_{\text{text}}} \circ v_{\text{text}}$
    end
  end
  $\{E_1, E_2, ..., E_n\} \leftarrow$ SemanticEquivalenceClasses$(P_{i,i+1})$; `// Using LM`
  for *each equivalence class* $E_q \in \{E_1, E_2, ..., E_n\}$ **do**
    Create disagreement node v_q with $v_{q_{\text{text}}} \leftarrow p_{j_{\text{text}}}$ for some $p_j \in E_q$;
    Store alternative phrasings $S(v_q) \leftarrow \{p_{j_{\text{text}}} | p_j \in E_q\} \setminus \{v_{q_{\text{text}}}\}$;
    Add edges (c_i, v_q) and (v_q, c_{i+1}) to E, preserving original weights;
  end
  Add v_q to V_D;
end
return *Complete* CONGR g

Comparison prompt for disagreement node construction for text

You are given two pieces of text. Your task is to determine whether they are semantically equivalent based solely on their factual content.
Here are the specific guidelines:
- Texts are equivalent if they convey the same core information or concept, regardless of wording or structure
- If one text has information that is a subset of the other text, then the texts are equivalent
- Focus ONLY on the essential claims, not on:
  * Stylistic differences or tone
  * Level of detail (if the core facts remain the same)
  * Connotative differences between words
  * Implied significance or emphasis
  * Presentation order (if all key information is present in both)
- Minor additions of non-contradictory information should not make texts non-equivalent
- For ambiguous cases, prioritize the central claim or purpose of the text

Examples of equivalent pairs:
- "The meeting starts at 3pm" and "The 3 o'clock meeting will begin on time"
- "Research indicates a 15
- "was influential in the field" and "had a significant impact on the community"

Examples of non-equivalent pairs:
- "The project might be completed by Friday" and "The project will be finished by Friday"
- "Most experts agree on the approach" and "All experts support the approach"

Strictly follow these guidelines and return ONLY:
- equivalent
- not equivalent

Comparison prompt for disagreement node construction for math

You are given two pieces of text from mathematical solutions. Your task is to determine whether the two solution segments are mathematically equivalent in their content, while allowing for stylistic variations.

Here are some important guidelines:

- Solutions should be considered equivalent if:
  1. They communicate the same mathematical content/approach, even if word choice or phrasing differs
  2. They contain the same key mathematical ideas, even if expressed differently
  3. The same mathematical steps are described, even if using different words
  4. They present the same final answer, regardless of wording style or formatting
- Allow for these variations while still considering solutions equivalent:
  1. Stylistic differences ("we will" vs. "we'll" or "I'll")
  2. Different levels of formality in the explanation
  3. Minor rephrasing that preserves the core mathematical content
  4. Use of synonyms or alternative mathematical terminology for the same concept
- Solutions are NOT equivalent if:
  1. They use fundamentally different mathematical approaches
  2. They work with different formulas or equations
  3. They present different mathematical steps or operations
  4. They reach different conclusions or answers
  5. One contains substantial mathematical content that the other lacks
- When examining final answers, focus on mathematical equivalence rather than stylistic presentation
- For solution steps, maintain the core mathematical approach while allowing for rephrasing

Examples of solutions that SHOULD be considered equivalent:

- "We will systematically evaluate each possible grouping" and "We'll evaluate each grouping"
- "The answer is x = 5" and "Therefore, x equals 5"
- "Using the quadratic formula" and "Applying the quadratic formula"

Strictly follow the guidelines above.

Return your judgment in the following format. Do not include any other text:

- equivalent
- not equivalent

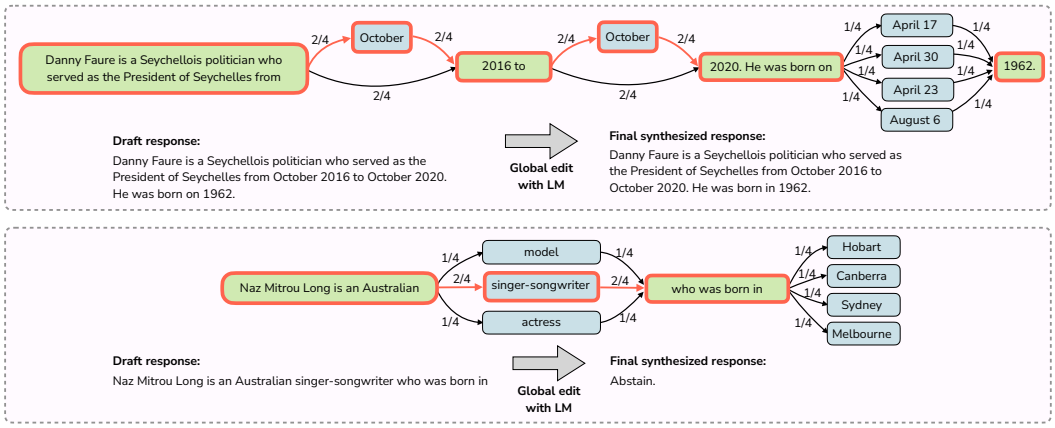

Figure 5: Consensus decoding synthesizes responses by traversing a ConGr and selecting nodes that are present in at least a τ fraction of responses, where $\tau \in [0, 1]$ is a hyperparameter ($\tau = 0.5$ in this example). The selected nodes' text labels are concatenated and processed by a secondary LM to produce a coherent response (top). When this is not possible, the result is an abstention (bottom).

F Decoding Algorithm Details

Figure 5 gives examples for applying consensus decoding in two different scenarios.

We present the pseudocode for our consensus decoding algorithm in Algorithm 2. We present the pseudocode for our guided self-verification algorithm in Algorithm 3.

Algorithm 2: Consensus Decoding from ConGrs

Input: ConGr $g = (V, E, W)$ with $V = V_C \cup V_V \cup \{v_{\text{START}}, v_{\text{END}}\}$, Consensus Threshold $\tau \in [0, 1]$, Edit Function f

Output: Consensus Response $y_{\text{consensus}}$

$y_{\text{consensus}} \leftarrow$ "" ; `// Initialize empty string`

$V_{\text{ordered}} \leftarrow \text{TopologicalSort}(g)$; `// Order nodes`

for *each node* $v \in V_{ordered}$ *in order* **do**

  if $d^w(v) \geq \tau$ **then**

    $y_{\text{consensus}} \leftarrow y_{\text{consensus}} \circ v_{\text{text}}$; `// Concatenate text`

  end

end

$y_{\text{consensus}} \leftarrow f(y_{\text{consensus}})$; `// Apply edits`

return $y_{consensus}$

Algorithm 3: Guided Self-Verification using ConGrs

Input: ConGr $g(R) = (V, E, W)$ with $V = V_C \cup V_V \cup \{v_{\text{START}}, v_{\text{END}}\}$ for a response set R of size m, LM $\mathcal{M}$ which generated R, Pruning Threshold $\kappa \in [0, 1]$, Candidate Solution set $\mathcal{C} \leftarrow R$

Output: Decoded Response

$V \leftarrow \text{TopologicalSort}(g)$; `// Order all nodes`

for *each consensus node* $u \in V_C$ **do**

  if $d_{out}(u) \geq \kappa$ **then**

    for *each following variable nodes* v_a *and* v_b *with their corresponding candidate solutions* C_a *and* C_b, *such that* $(u, v_a), (u, v_b) \in E$ **do**

      Prune C_a, C_b from $\mathcal{C}$ based on verification scores from $\mathcal{M}(partial(C_a, v_a), partial(C_b, v_b))$

    end

  end

end

Synthesize final decoded response: $y_{\text{decoded}} = \mathcal{M}(\mathcal{C})$

return $y_{decoded}$

The prompt for removing disfluencies with a secondary LM and hence synthesizing the final response from the draft response is given below:

Consensus Decoding: Final synthesis prompt

You are given a piece of text that is a part of a {task}. This text may contain some minor errors that make it incoherent as well as potentially redundant information. Your task is to fix the errors and make the text coherent. Then, remove any redundant information. Text: {text}
If this is not possible because the text is just a fragment of a sentence, return "Abstain". If the text already claims a lack of knowledge about the topic, return "Abstain". Only return the cleaned up text. Do not include any other text:

The prompt for pairwise self-verification of partial solutions with the same LM $\mathcal{M}$ is given on the next page:

Guided Self-Verification: Pairwise self-verification prompt

You will be given a problem and 2 partial solutions. Your task is to use comparison as an EFFICIENCY TOOL to quickly identify potential errors. You will be given guidelines to follow, and you will be penalized if you do not follow them.
Problem: {problem}
Partial Solution 1: {partial_solution_1} Partial Solution 2: {partial_solution_2}

- CRITICAL GUIDELINES:
  * DO NOT penalize a solution for being incomplete or having missing steps
  * DO NOT make a comparison of which solution is better
  * DO NOT consider steps incorrect just because they differ between solutions
  * DO NOT prematurely evaluate based on final answers or future steps
  * DO NOT expect both solutions to be at the same stage of completion
  * DO NOT consider a step incorrect just because it lacks sufficient detail or justification
- KEY EFFICIENCY PRINCIPLE:
  * Use agreement between solutions as evidence of correctness
  * Use disagreement as a signal to investigate more deeply
  * Only label a step as an error if it contains a specific mathematical mistake
  * Incompleteness is not a mathematical error.
- EFFICIENT VERIFICATION APPROACH:
- 1. QUICK COMPARISON (Use this to focus your attention):
  * Immediately identify where the solutions differ in approach or results
  * Use these differences as "error hotspots" to prioritize your verification
  * When solutions agree, you can generally assume that part is correct
  * When solutions disagree, investigate those specific points deeply
- 2. TARGETED VERIFICATION (Only where needed):
  * Most important: Do not consider any incomplete steps as errors
  * Focus your mathematical verification on the "hotspots" identified above
  * Check mathematical validity only at points of difference or uncertainty
  * Avoid line-by-line checking of steps where solutions agree
  * For each potential error spot, verify if the mathematical reasoning is valid
  * If an intermediate step is later corrected, do not penalize the solution for having the incorrect intermediate step
- After your targeted verification, propose a score tuple (score_1, score_2):
  * Score (1,1) if both partial solutions are valid
  * Score (1,0) if only the first solution is valid
  * Score (0,1) if only the second solution is valid
  * Score (0,0) if both solutions are invalid
- In case you score a solution as 0, you must give an explanation for each check below:
  * If you score a solution as 0, you MUST identify the specific mathematical error.
  * You must also double check the problem statement. Reconsider your score and determine if you have misinterpreted the problem statement.
  * You must also check whether you have penalized a solution for being incomplete or having missing steps.
- Before outputting your final score, you must answer these questions:
  * STOP! Did you give a score of 0 to a solution that was incomplete?
  * STOP! Did you penalize a solution for being incomplete or having missing steps?
  * STOP! Did you make a comparison of which solution is better?
  * STOP! Did you consider steps incorrect just because they differ between solutions?
  * STOP! Did you prematurely evaluate based on final answers?
  * STOP! Did you consider a step incorrect just because it lacks sufficient detail or justification?

Now give your final score: Final score:

The prompt for synthesizing a final response from the pruned candidate response set using the same LM $\mathcal{M}$ is given below:

Final synthesis prompt for guided self-verification

Solve the following math problem with mathematical precision and clarity.
Problem: {problem}
Below are potential solution approaches with sections marked as uncertain (between `*START_UNCERTAIN_REGION*` and `*END_UNCERTAIN_REGION*`). These sections may contain conceptual or computational errors.
There are also sections marked as `*START_POSSIBLE_ERROR*` and `*END_POSSIBLE_ERROR*`. A verification step indicated that these steps are highly likely to contain errors.
Potential Approaches: {masked_candidate_responses}

- Your task:
  1. Analyze all potential approaches critically, identifying their mathematical strengths and weaknesses If the approaches contain different answers, think carefully about why they are different, and use this to identify potential errors.
  2. Using the sections with special markers, identify potential errors.
  3. Develop a rigorous, step-by-step solution based on sound mathematical principles
  4. For uncertain regions:
     * Verify each step using algebraic or numerical validation
     * If correct, incorporate these steps with appropriate justification
     * If incorrect, provide clear corrections with mathematical reasoning for your changes
  5. Follow a comparative approach, using the differences between approaches to identify potential errors.
  6. Do not blindly follow the approaches, but rather use them to identify potential errors.
- Guidelines for your solution:
  * Begin with a strategic overview of your chosen approach
  * Present each mathematical step with clear notation and justification
  * Pay special attention to areas that were previously marked uncertain

Conclude your solution with: Therefore, the final answer is: `$\boxed{{answer}}$`. Solution:

G Experiment Prompts and Details

G.1 Factuality and Refusal-based Experiments

Generation hyperparameters. We first generate five samples per test example that our method will synthesize for the factuality and refusal-based tasks. For the factuality tasks, we use the same prompts as [8] while for the refusal-based tasks, we use the same prompts as [55]. Table 4 gives our hyperparameters.

	QWEN 2.5 72B LLAMA 3.3 70B OLMO 2 32B	QWEN 2.5 7B LLAMA 3.1 8B OLMO 2 7B
Quantization	8-bit	n/a
Temperature	0.9	0.9
samples_per_prompt	5	5
max_new_tokens	500	500
Random seed	42	42

Table 4: Hyperparameters for response generation for factuality and refusal experiments.

Runtime and infrastructure. For CONGRS construction and decoding, experiments using a single model's responses took on average: 33 seconds per entity (55 minutes total for 100 prompts) for Biographies, 38 seconds per entity (63 minutes total for 100 prompts) for PopQA, 31 seconds per prompt (129 minutes total for 250 prompts) for Scientific Attributions, 3 seconds per prompt (13 minutes total for 250 prompts) for False Presuppositions, 28 seconds per prompt (116 minutes total for 250 prompts) for Historical Events. We used a cluster compute node with 5 NVIDIA RTX A6000 GPUs and a Macbook Pro with an Apple M1 Pro with 16 GB of RAM for analysis.

Methods. We compare consensus the following methods and baselines:

- *Consensus decoding with* CONGRS: We apply consensus decoding with selection thresholds of $\tau = 0.3, 0.5$. We use `gpt-4.1-mini` as the secondary LM. We also perform additional analyses for biography-based tasks with different selection thresholds in Appendix J.
- *Minimum Bayes Risk Decoding* [5]: performs pairwise comparisons among responses to choose the response with lowest expected risk. We use BERTScore [56] as the risk function.
- *LM consensus*: We prompt an LM to synthesize a consensus response, with an option to abstain if there is too much variation. Exact prompts are below This method is similar to Universal Self-Consistency [6], except that we synthesize a new response instead of selecting one response. For the LM Consensus baseline, we use gpt-4o-mini for generating the consensus generation with an added option to abstain. We use the following prompt:

LM consensus baseline prompt

You are given 5 texts. Your task is to form/generate a consensus/agreement text using the given texts. Consensus or agreement would mean producing a new text that uses the given 5 texts to find a coherent text that includes words and information that is consistent across all the given texts. Text 1: text[0] Text 2: text[1] Text 3: text[2] Text 4: text[3] Text 5: text[4]
Here are some important guidelines:

- If the texts differ at a certain point/word, the consensus text should select the most frequent word from among the given texts at the point of difference.
- If the texts differ at a certain point/word and there is no most frequent word, the consensus text should select the word that is most similar to the other words in the text.
- Abstain if the texts are too different and no consensus can be reached.

Strictly follow the guidelines above, especially regarding abstaining if the texts are too different. Return your generation in the following format. Do not include any other text:
consensus text: [your consensus text here]

- *Greedy decoding*: We generate a single greedy response using a temperature of zero and the following hyperparameters: *do_sample*: False, *max_new_tokens*: 500.
- *Shortest response*: We prompt an LM to select the shortest response from the candidate set of multiple responses based on number of words in the response. It has been observed that correct responses are often shorter than incorrect responses for some tasks [26, 57]. We hypothesize that shorter responses may contain fewer hallucinations. We use gpt-4o-mini with the following prompt:

Shortest response baseline prompt

You are given 5 texts. Text 1: text[0] Text 2: text[1] Text 3: text[2] Text 4: text[3] Text 5: text[4]
Your task is to output the shortest text amongst the given texts in terms of total words in a text. Compare the texts and select the one that is the shortest. If there are multiple texts with the same length, select the first one.
Strictly follow the guidelines above.
Return your generation in the following format. Do not include any other text:
shortest text: [your shortest text here]

- *Reasoning model response*: We use QwQ-32B [25], a state-of-the-art reasoning model with thinking and reflection capabilities. We generate only one response, since we are using QwQ-32B to compare our method against *serial* inference-time scaling as opposed to parallel. Due to this, we use a larger context window of 4096 tokens. Note that this method does not utilize the original set of model responses. We use the following standard hyperparameters for QwQ 32B: *quantization*: 4-bit, *temperature*: 0.6, *top_k*: 40, *top_p*: 0.95, *do_sample*: True, *random seed*: 42, *samples_per_prompt*: 1, *max_new_tokens*: 4096.
- *Atomic self-consistency* [18]: ASC first clusters all sentences in all responses with hierarchical clustering. It then discards any cluster with fewer than Θ responses. Θ is a hyperparameter similar to τ in consensus decoding: $\Theta = 2$ corresponds to $\tau = 0.3$, and $\Theta = 3$ corresponds to $\tau = 0.5$. Finally, a secondary LM is given a list of the longest sentences from each remaining cluster and prompted to produce a synthesized response. We use the settings from Thirukovalluru et al. [18]: `sup-simcse-bert-base-uncased` is the sentence embedding model, and $d = 0.15$ for agglomerative clustering. For fair comparison, we use `gpt-4.1-mini` as the secondary LM and the same prompt as consensus decoding to produce the synthesized response.

Evaluation. We use FActScore to decompose the factuality task baseline responses into atomic units for verification against the Wikipedia database. In addition to FActScore, we also compute the number of supported, unsupported, and total facts for the response.

For the refusal-based tasks, we use the HALoGEN code for evaluation of response ratio and hallucination scores of the baseline responses. We use `gpt-4.1-mini` as an additional LM judge for assessing abstention in a given response with the following prompt:

Abstention prompt

You are given a prompt and its response . Your task is to judge whether the response to the prompt is an abstention from answering or not? Just answer with 'yes' or 'no'. 'yes' if it is an abstention, 'no' if it is not an abstention and it seems like an answer.
prompt: prompt, response: response
Return your generation in the following format. Do not include any other text:
abstention: [your judgment here]

G.2 Reasoning Experiments

For both MATH and AIME, we use the following configuration for both models: *quantization*: 8-bit, *temperature*: 0.9, *random seed*: 42, *samples_per_prompt*: 5

For MATH, we use *max_new_tokens* = 1024, whereas for the more difficult AIME dataset, we use *max_new_tokens* = 8192.

For both MATH and AIME, we use the following prompt to generate samples:

Prompt for generating samples for MATH and AIME

Solve the following math problem efficiently and clearly:

- For simple problems (2 steps or fewer): Provide a concise solution with minimal explanation.
- For complex problems (3 steps or more): Use this step-by-step format:

  ## Step 1: [Concise description] [Brief explanation and calculations]
  ## Step 2: [Concise description] [Brief explanation and calculations]
  ...

Regardless of the approach, always conclude with:
Therefore, the final answer is: `$\boxed{{answer}}$`. I hope it is correct.
Where {answer} is just the final number or expression that solves the problem.
Problem: {problem}

We evaluate the following methods:

- *Guided Self-Verification with* CONGRS: We apply guided self-verification with pruning threshold of $\kappa = 0.7$.
- *Self-consistency* [4]: We take a majority vote of the final answer over all m responses.
- *Self-verification* [30, 22, 29]: We ask the same LM to score all m responses based on correctness and select the best scoring response. For fairness, the verification prompt is kept the same as our guided self-verification method.
- *Pass@*m: We measure whether at least one of the m responses contains a correct answer. This represents an upper bound for methods that aggregate over responses.

Runtime. For CONGRS construction and guided self-verification, experiments using a single model's responses took on average: 150 seconds per question (21 hours total) for MATH and 360 seconds per question (3 hours total) for AIME.

H Additional Results

Table 5 gives our full results for comparing consensus decoding to UAD [8]. Table 6 shows that when consensus decoding does abstain, it does so for entities where the original responses have low FActScore on average.

We also evaluate consensus decoding with CONGRS on long-form PopQA, as in Jiang et al. [8] (Tables 8, 9). We also benchmark our approaches using three smaller models: QWEN 2.5 7B, LLAMA 3.1 8B, and OLMO 2 7B (Tables 7, 9, 11).

Figure 2 in the main paper shows that consensus decoding at various selection thresholds τ achieves a better tradeoff between FActScore and the number of true claims provided. It accounts for methods' different response ratios using the approach as Jiang et al. [8]: if a method abstains for an entity, then that entity gets a FActScore of 1 and a number of supported facts of 0.

Table 5: Consensus decoding with $\tau = 0.3$ uses 82% fewer tokens from secondary LMs compared to UAD [8] without sacrificing performance when synthesizing across $m = 5$ responses from QWEN 2.5 72B.

Method	FActScore	#T	#F	RR	Mean # Tokens Used
UAD	0.59	27.32	20.84	0.96	76,220.48
CONGRS	0.62	19.00	8.63	0.96	13,391.88

Table 6: Mean FActScore of original responses for entities that consensus decoding abstains on, compared to FActScore on all entities. Consensus decoding abstains on responses that are highly likely to contain hallucinations.

τ	Model	FActScore Abstained Entities	FActScore All entities
0.3	QWEN 2.5 72B	0.31	0.68
	QWEN 2.5 7B	0.34	0.68
0.5	QWEN 2.5 72B	0.20	0.64
	QWEN 2.5 7B	0.28	0.64

Table 7: Results for biography generation for smaller models: FActScore, numbers of supported (#T) and unsupported (#F) facts, and response ratio (R, how often the model doesn't abstain).

	QWEN 2.5 7B				LLAMA 3.1 8B				OLMO 2 7B			
Method	**FActScore**	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**
Greedy	0.68	18.40	8.27	0.98	0.85	15.24	2.28	0.72	0.60	22.71	14.03	1.00
Mean of m	0.56	16.02	11.69	0.99	0.78	19.96	4.77	0.76	0.59	24.23	14.56	0.99
Shortest	0.57	15.81	11.13	0.99	0.81	20.21	4.10	0.71	0.58	22.35	13.80	0.98
LM Consensus	0.58	18.43	13.03	1.00	0.67	19.64	7.00	0.64	0.64	24.69	13.33	1.00
MBR	0.56	15.76	11.35	1.00	0.81	20.60	4.24	0.75	0.60	22.08	13.42	1.00
QWQ 32B	0.55	19.26	14.05	0.98	0.55	19.26	14.05	0.98	0.55	19.26	14.05	0.98
CONGRS (τ=0.3)	0.64	16.75	6.21	0.87	0.80	21.26	3.86	0.74	0.78	26.29	4.86	0.70
CONGRS (τ=0.5)	0.76	11.71	2.68	0.75	0.85	20.32	1.78	0.41	0.85	19.46	2.52	0.61

Table 8: Results for larger models on **PopQA**. We report FActScore [23], number of supported (#T) and unsupported (#F) facts, and response ratio (R, how often the model responds and doesn't abstain). Like in Table 7, consensus decoding with $\tau = 0.3$ consistently improves FActScore by decreasing the number of unsupported facts.

Method	QWEN 2.5 72B				LLAMA 3.3 70B				OLMO 2 32B			
	FActScore	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**
Greedy	0.68	18.24	8.08	1.00	0.62	15.54	8.24	0.84	0.74	22.96	7.47	0.99
Mean of m	0.69	18.94	8.04	0.97	0.62	15.83	8.50	0.81	0.75	24.11	7.77	0.99
Shortest	0.71	18.96	7.18	0.96	0.64	14.51	7.04	0.71	0.76	23.87	7.21	0.97
LM Consensus	0.72	21.40	8.10	0.99	0.68	19.08	8.31	0.88	0.78	25.14	7.03	1.00
MBR	0.72	19.49	7.24	0.97	0.65	16.01	7.87	0.83	0.77	25.03	7.06	0.98
QWQ 32B	0.57	22.37	15.81	0.97	0.57	22.37	15.81	0.97	0.57	22.37	15.81	0.97
CONGRS (τ=0.3)	0.74	19.06	5.79	0.97	0.76	16.80	4.51	0.69	0.79	23.63	5.24	0.88
CONGRS (τ=0.5)	0.79	14.67	3.68	0.88	0.81	11.48	2.55	0.58	0.85	16.28	2.34	0.76

Table 9: Results for smaller models on **PopQA**. We report FActScore [23], number of supported (#T) and unsupported (#F) facts, and response ratio (R, how often the model responds and doesn't abstain) for smaller models. Like in Table 7, consensus decoding with $\tau = 0.3$ consistently improves FActScore by decreasing the number of unsupported facts.

Method	QWEN 2.5 7B				LLAMA 3.1 8B				OLMO 2 7B			
	FActScore	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**	**FActScore**	**#T**	**#F**	**R**
Greedy	0.55	15.11	11.74	0.99	0.66	16.06	7.92	0.86	0.60	19.76	12.59	0.98
Mean of m	0.52	14.33	12.69	0.99	0.65	16.11	8.27	0.82	0.59	20.36	13.82	0.98
Shortest	0.56	14.86	11.87	0.99	0.65	15.93	7.60	0.73	0.58	19.67	13.93	0.97
LM Consensus	0.61	17.67	10.97	0.96	0.72	20.25	7.77	0.87	0.70	23.47	9.84	0.99
MBR	0.51	13.91	12.47	0.96	0.69	17.06	7.19	0.83	0.62	20.33	12.30	0.98
QWQ 32B	0.57	22.37	15.81	0.97	0.57	22.37	15.81	0.97	0.57	22.37	15.81	0.97
CONGRS (τ=0.3)	0.66	11.33	4.78	0.83	0.74	13.11	4.04	0.71	0.75	18.18	4.86	0.83
CONGRS (τ=0.5)	0.76	7.74	2.12	0.73	0.82	10.83	2.47	0.60	0.86	10.09	1.85	0.54

I Consensus Decoding Improves Abstention Ability

We evaluate consensus decoding on 250 instances from each of the three abstention-based task in the HALoGEN benchmark [55]: False Presuppositions, Scientific Attribution, and Historical Events. The HALoGEN metrics are Response Ratio, which measures how often a model responds to the initial prompt, and a Hallucination Score. Lower is better for both. We use the same models and baselines as for biography generation.

Table 10: Performance on the **False Presuppositions** and **Scientific Attribution** tasks from HALoGEN. R: Response Ratio, H: Hallucination Score. Lower is better. Consensus decoding with ConGrs consistently achieves low R and low H.

	False Presuppositions				Scientific Attribution			
	Qwen 2.5 72B		Llama 3.3 70B		Qwen 2.5 72B		Llama 3.3 70B	
Method	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**
Greedy	0.22	0.16	0.66	0.49	0.78	0.67	0.32	0.30
Mean of m	0.25	0.17	0.62	0.46	0.76	0.66	0.28	0.26
Shortest	0.16	0.11	0.68	0.49	0.61	0.55	0.16	0.16
LM Consensus	0.33	0.23	0.70	0.51	0.33	0.29	0.29	0.26
MBR	0.22	0.15	0.62	0.45	0.75	0.66	0.24	0.22
QwQ 32B	0.04	0.01	0.04	0.01	0.83	0.74	0.83	0.74
ConGrs (τ=0.3)	0.14	0.09	0.33	0.21	0.19	0.16	0.16	0.15
ConGrs (τ=0.5)	0.11	0.07	0.27	0.17	0.11	0.08	0.08	0.07

Tables 10, 11, and 12 show that consensus decoding with ConGrs consistently reduces both Response Ratio and Hallucination Score compared to baselines. QwQ 32B is a strong baseline for False Presuppositions, but confidently produces hallucinated references for Scientific Attribution even after extended reflection.

Table 11: Performance on the **False Presuppositions** and **Scientific Attribution** tasks from HALoGEN when synthesizing responses from small model sizes. We report Response Ratio (R↓) and Hallucination Score (H↓), both lower is better. Consensus decoding with ConGrs consistently achieves low Response Ratio and Hallucination Score.

	False Presuppositions						Scientific Attribution					
	Qwen 2.5 7B		Llama 3.1 8B		OLMo 2 7B		Qwen 2.5 7B		Llama 3.1 8B		OLMo 2 7B	
Method	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**
Greedy	0.40	0.28	0.60	0.45	0.51	0.41	0.56	0.49	0.42	0.33	0.89	0.81
Mean of m	0.41	0.30	0.61	0.47	0.51	0.42	0.67	0.62	0.49	0.43	0.88	0.82
Shortest	0.36	0.25	0.50	0.34	0.54	0.46	0.49	0.45	0.29	0.25	0.77	0.71
LM Consensus	0.48	0.36	0.69	0.55	0.76	0.63	0.05	0.04	0.15	0.13	0.05	0.04
MBR	0.39	0.28	0.58	0.43	0.48	0.39	0.65	0.61	0.46	0.39	0.91	0.84
QwQ 32B	0.04	0.01	0.04	0.01	0.04	0.01	0.83	0.74	0.83	0.74	0.83	0.74
ConGrs (τ=0.3)	0.10	0.06	0.09	0.05	0.13	0.10	0.07	0.06	0.30	0.26	0.15	0.13
ConGrs (τ=0.5)	0.08	0.06	0.07	0.04	0.11	0.08	0.02	0.02	0.18	0.16	0.09	0.08

Table 12: Performance on the **Historical Events** task from HALoGEN [55]. We report Response Ratio (R↓) and Hallucination Score (H↓), both lower is better. Consensus decoding with ConGrs consistently achieves low Response Ratio and Hallucination Score for the OLMo model family.

Method	Qwen 2.5 72B		Qwen 2.5 7B		Llama 3.3 70B		Llama 3.1 8B		OLMo 2 32B		OLMo 2 7B	
	R↓	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**	**R↓**	**H↓**
Greedy	0.008	0.008	0.064	0.064	0	0	0	0	0.280	0.280	0.004	0.004
Mean of m	0.008	0.008	0.056	0.056	0.0008	0.0008	0.0032	0.0032	0.252	0.252	0.0064	0.0064
Shortest	0.008	0.008	0.024	0.024	0	0	0	0	0.148	0.148	0.004	0.004
LM Consensus	0.012	0.012	0.056	0.056	0.004	0.004	0	0	0.312	0.312	0.004	0.004
MBR	0.008	0.008	0.040	0.040	0.004	0.004	0	0	0.200	0.200	0.004	0.004
QwQ 32B	0.232	0.232	0.232	0.232	0.232	0.232	0.232	0.232	0.232	0.232	0.232	0.232
ConGrs (τ=0.3)	0.008	0.008	0.028	0.028	0.004	0.004	0.016	0.016	0.096	0.096	0	0
ConGrs (τ=0.5)	0.008	0.008	0.012	0.012	0	0	0.008	0.008	0.060	0.060	0	0

J Ablation Experiments

J.1 Temperature

For a set of 25 randomly sampled entities for the biography generation setting with Qwen 2.5 72B, we generate responses with different temperatures. ConGrs are effective at synthesizing information across responses even when the responses were generated with various temperatures.

Table 13: Ablation results for Qwen 2.5 72B Inst at different temperatures and CONGRS thresholds τ = 0.3, 0.5. We report FActScore, number of supported (#T) and unsupported facts (#F), and response ratio (R) averaged across the 25 sampled entities.

Temperature	Threshold τ	FActScore	#T	#F	R
0.1	CONGRS (0.3)	0.61	17.83	10.25	0.96
	CONGRS (0.5)	0.64	16.04	7.29	0.96
0.3	CONGRS (0.3)	0.62	18.83	10.83	0.96
	CONGRS (0.5)	0.68	14.65	5.13	0.92
0.5	CONGRS (0.3)	0.64	19.50	9.67	0.96
	CONGRS (0.5)	0.66	15.09	5.35	0.92
0.7	CONGRS (0.3)	0.67	19.22	7.35	0.92
	CONGRS (0.5)	0.71	17.41	4.55	0.88

J.2 Number of responses

We generate $m = 10$ responses per entity, for a set of 25 randomly sampled entities for the biography generation setting with QWEN 2.5 72B; with temperatures 0.7 and 0.9. We report results for our consensus decoding method with different threshold τ values. CONGRS are effective at synthesizing information across 10 responses.

Table 14: Ablation results for Qwen 2.5 72B Inst ($m = 10$) at different temperatures and CONGRS thresholds τ. We report FActScore, number of supported (#T) and unsupported facts (#F), and response ratio (R) averaged across the 25 sampled entities.

Temperature	Threshold τ	FActScore	#T	#F	R
0.9	CONGRS (0.1)	0.47	20.80	30.36	1.00
	CONGRS (0.2)	0.49	16.72	14.52	1.00
	CONGRS (0.3)	0.67	13.86	4.81	0.84
	CONGRS (0.4)	0.75	11.90	2.90	0.84
	CONGRS (0.5)	0.79	11.85	2.20	0.80
	CONGRS (0.7)	0.81	9.58	1.74	0.76
0.7	CONGRS (0.1)	0.59	25.44	22.44	1.00
	CONGRS (0.2)	0.59	21.56	12.56	1.00
	CONGRS (0.3)	0.67	20.41	7.68	0.88
	CONGRS (0.4)	0.70	18.45	5.23	0.88
	CONGRS (0.5)	0.74	16.45	4.05	0.88
	CONGRS (0.7)	0.78	13.70	2.30	0.80

J.3 Selection Threshold

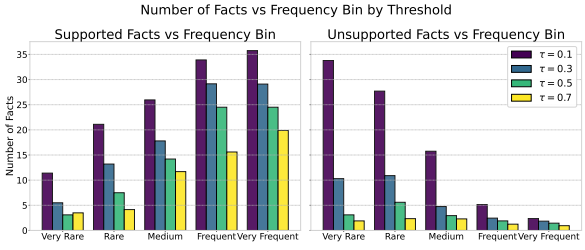

Figure 6: Different selection thresholds for Consensus Decoding are better for different kinds of entities. Generated biographies for very frequent entities are often very factual and benefit from a permissive, low value of τ. On the other hand, generated biographies for very rare entities benefit from a high value of τ, which only aggregates spans of text that occur in the vast majority of responses.

Selection threshold τ trades off informativeness and factuality. The frequency of each entity in pre-training data varies across our sets of entities. As a result, the factuality of the models' original responses varies as well. In the case of rare entities, variation between model responses can be indicative of hallucinations. However, for very common entities, variation may not be a result of hallucinations. In general, the threshold τ that we choose when performing consensus decoding controls whether the final result is more of an intersection between the original set of responses or whether it is a union of the responses.

To study this trade-off between informativeness and factuality, we first estimate the frequency of each entity. Since we do not have access to each model's pretraining data, we use the monthly page views of each entity's Wikipedia page as a proxy measure of its frequency, as in Min et al. [23]. We then partition the entities into 5 equal-sized bins and perform consensus decoding with thresholds of $\tau = 0.1, 0.3, 0.5, 0.7$. We then analyze FActScores and the number of supported/unsupported claims for entities in each bin in Figure 6.

For very frequent entities, we see that all thresholds result in high FActScores. Moreover, even using the most permissive threshold $\tau = 0.1$ results in a negligible increase in the number of unsupported claims, while greatly increasing the number of supported claims.