XLGoBench: Detecting cross-lingual skill gaps with algorithmic tasks

Purvam Jain[1] **Preethi Jyothi**[1,2] **Vihari Piratla**[1]* **Suvrat Raju**[1,3]
[1]Google DeepMind
[2]Indian Institute of Technology Bombay
[3]International Centre for Theoretical Sciences, Tata Institute of Fundamental Research

Abstract

We introduce a set of synthetic algorithmic tasks to detect cross-lingual gaps in the abilities of large language models. Our benchmark is commensurate across languages, since it requires models to perform the same underlying task in different languages; scalable, since each task can be generated at varying levels of complexity allowing it to be adapted to models with different capabilities; quantifiable, since every task admits an objective notion of correctness; and transparent, since tasks are generated from simple templates that can be readily audited for translation errors. Because our benchmark focuses on algorithmic tasks, differential performance is a sufficient—but not necessary—indicator of cross-lingual gaps. Nevertheless, we show through extensive experiments that our benchmark exposes persistent cross-lingual gaps in multiple state-of-the-art models.[1]

1 Introduction

Large language models (LLMs) have had a significant and increasing impact on society, including in fields such as education (Hausman et al., 2025), professional writing (Noy and Zhang, 2023), customer service (Brynjolfsson et al., 2025), programming (Handa et al., 2025), and others (Hausman et al., 2025). Nevertheless, the widespread adoption of these models has exacerbated concerns about global inequality (Skare et al., 2024; UNDP, 2025; Misra et al., 2025).

Cross-lingual gaps in the abilities of LLMs represent one facet of broader disparities faced by users. These gaps persist because capability evaluation benchmarks are largely in English (Ahuja et al., 2024), leaving model abilities across many non-English languages largely unmeasured (Wu et al., 2025). On the other hand, it is encouraging that LLMs show a measure of cross-lingual transfer, which allows them to perform well even in relatively low-resource languages (Nguyen et al., 2024). Moreover, improvements in the abilities of models can make LLMs more accessible to users (Microsoft AI Economy Institute, 2026) and potentially even empower linguistically marginalized communities (Ding et al., 2024).

Therefore, removing cross-lingual gaps is both important and potentially achievable. The first step towards this objective is to precisely measure cross-lingual gaps. Towards this goal, we introduce a new synthetic benchmark.

Our benchmark XLGoBench comprises a set of algorithmic puzzles. This offers some advantages over alternate methods of measuring cross-lingual gaps.

1. **Transparency in translation:** A recurrent challenge in multilingual evaluations is to ensure translation accuracy. For example, popular multilingual benchmarks such as MMLU (Hendrycks et al., 2021) and MGSM (Shi et al., 2023) have been shown to contain translation, annotation or evaluation errors (Gema et al., 2025; Singh et al., 2025; Peter et al., 2025). This makes it unclear whether, for example, the remaining 1.2% gaps between English and Telugu on MGSM (Peter et al., 2025) with Gemini-2.5-Flash arise from genuine capability differences or experimental errors. In contrast, our puzzles are formed by combining a small set of templates. The translation accuracy of these templates can be checked by hand, as we did for our empirical tests, using expert human raters. Therefore, we do not expect translation errors to affect our evaluations.

2. **Scalability:** Traditional multilingual evalua-

* Corresponding author: vpiratla@google.com
[1]Replication material for our benchmark is available on GitHub.

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 45660–45710
October 24-29, 2026 ©2026 Association for Computational Linguistics

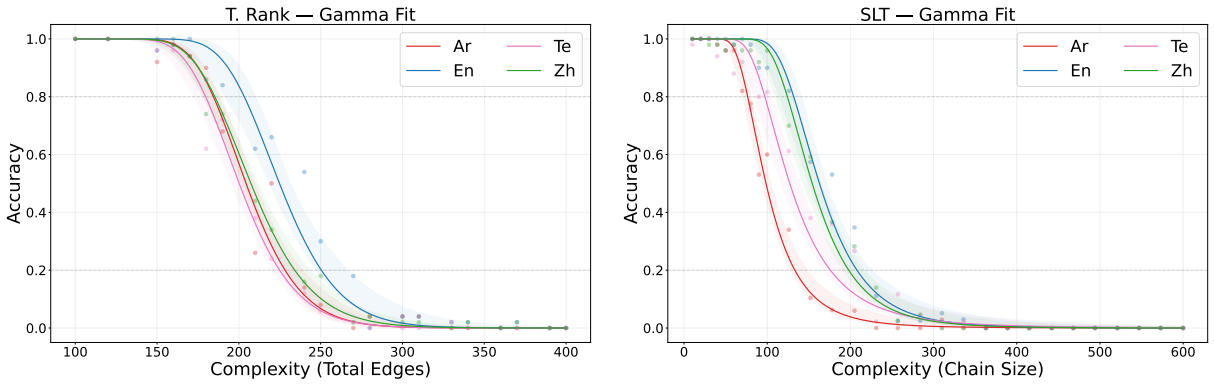


Figure 1: We show a graph of accuracy vs complexity for selected languages for (a) Gemini-3-Flash for the Tournament ranking task on the left and (b) Gemini-3.1-Flash-Lite for the Sequential Linear Transforms task on the right. Significant cross-lingual gaps are observed in the intermediate complexity region, as performance in English declines more slowly with complexity compared to other languages. The dots denote observed accuracy and the solid lines and uncertainty bands are obtained by fitting a mathematical model described in Section 3 to the data.

tions involve a multi-step process and manual collation. This has led to benchmarks that include long-context reasoning (Kim et al., 2025), expert knowledge (Xuan et al., 2025) or hard coding tasks (Huang et al., 2025). However, as models continue to improve, such benchmarks are expected to saturate, requiring the creation of new datasets.

In contrast, our puzzles can be scaled to arbitrarily high complexity. This reduces the accuracy of all models, including those that we might train in the foreseeable future, and exposes cross-lingual gaps where they exist. Similarly, the complexity can also be scaled down to test smaller models.

3. **Quantifiability:** Since the answers to our puzzles are easily verifiable, it is possible to perform a large number of experiments. This reduces potential statistical fluctuations in our measurements and enables the precise quantification of cross-lingual variation in model performance. Moreover, we do not need to use an LLM as a judge, removing another potential source of ambiguities.

4. **Commensurate Tasks:** In a multilingual test, it is subtle to ensure the task performed in one language is commensurate with a similar task in another language. For instance, differences in nuance, sentence structure and content might make it harder for models to retrieve information from a corpus in one language compared to another. Our tasks sidestep this issue since our puzzles are simple to express and, at an abstract level, encode a language-independent task that is the same regardless of the prompt language.[2]

The use of synthetic algorithm-based tasks is well-recognized for training and evaluating reasoning abilities of LLMs (Liu et al., 2025; Zhang, 2024; Tang et al., 2024; Wei et al., 2025; OpenAI, 2026; He et al., 2026). Nevertheless, it was not a priori guaranteed that cross-lingual gaps in the abilities of models on real-world tasks would be reflected in the simple synthetic tasks that we evaluate. A surprising empirical result of our study is that state-of-the-art models display differences in performance even on such synthetic puzzles.

We emphasize that a model that performs equally in two different languages on our simple algorithmic tasks, might show uneven performance when tested on tasks like creative writing, or awareness of nuance. So, our perspective is that gaps in performance in algorithmic tasks are *sufficient* but not necessary to demonstrate the existence of cross-lingual gaps. As a corollary, it is important to supplement the benchmark presented in this paper with measures of more subtle linguistic capabilities, and we leave this to future work.

Summary of results. We test six state-of-the-art models on a suite of five puzzles (described in more detail in Section 2) across seven different languages. The puzzles are based on templates, that are vetted by expert human raters as described in

[2]Here, commensurate refers to the underlying abstract puzzle rather than its surface realization in tokens. The potential implications of inefficient tokenization on cross-lingual gaps are discussed in Section 5.

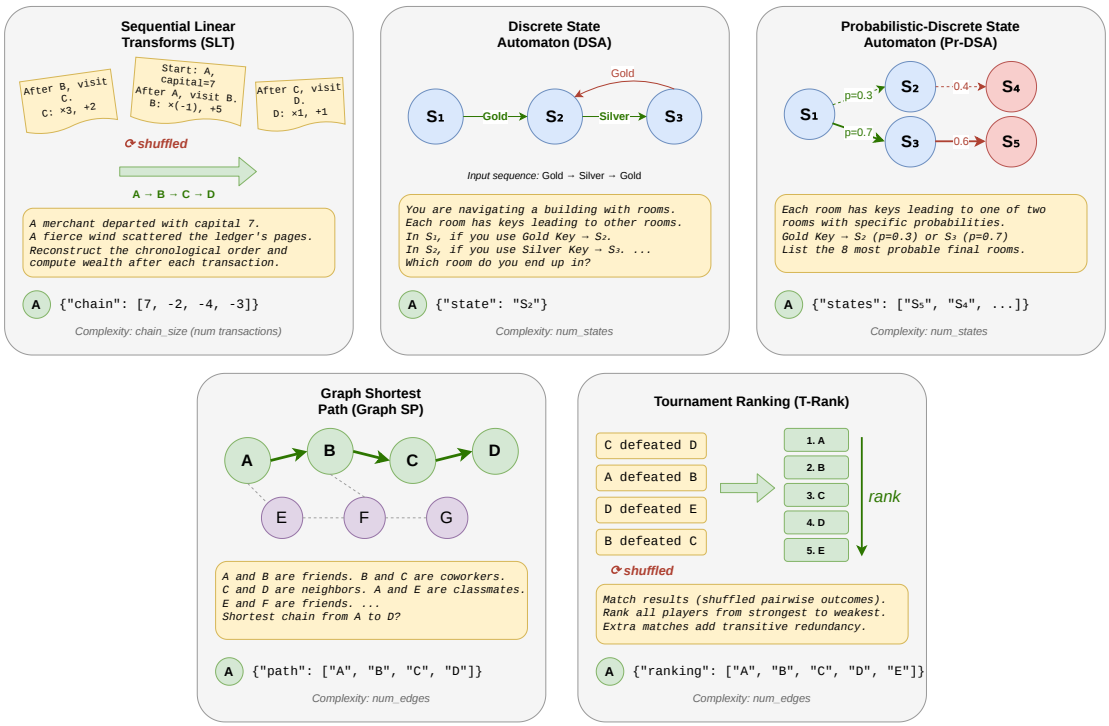


Figure 2: Five puzzle types in XLGoBench. We visually depict the basic schema of all puzzle types in our benchmark, along with their respective complexity measures. More detailed puzzle descriptions are in Section 2.

Section 2.3, and are scalable in complexity. We track the performance of the models from low complexity — where all models achieve perfect accuracy in all languages — to high complexity, where the accuracy collapses as described in Section 3. A statistically significant variation in the performance of models at a given complexity reflects a cross-lingual gap. We present a summary metric that measures the maximum possible cross-lingual gap utilizing a model of the variation of the accuracy with complexity. We show in Section 4 that this suite of puzzles, taken collectively, is sufficient to expose unambiguous cross-lingual gaps in all models. In Section 4.1, we show that these gaps cannot be attributed to translation errors. We also note that these gaps arise at intermediate complexity levels. We comment on possible underlying causes and future work in Section 5. The Appendices provide extended empirical results, including plots of accuracy vs complexity, and details on prompts and the construction of puzzles. The full set of results, code to reproduce our plots, and templates for the puzzles are included in the supplementary material for this paper. This supplementary material can be downloaded from zenodo.org/records/20423110.

1.1 Related Work

Benchmarks dedicated to measuring cross-lingual gaps have been of great interest since the era of encoder-only language models (Hu et al., 2020), and their scope has since expanded from classification tasks to more complex reasoning tasks for evaluating modern LLMs (Ghosh et al., 2025). Most closely related to our work, Dobler et al. (2026) previously introduced cross-lingual evaluations using synthetic tasks by translating templates from an English reasoning gym (Stojanovski et al., 2025). Bruun and Smart (2025) previously studied the variation in the accuracy of models on configurations of the Zebra puzzle across several European languages. Kim et al. (2025) also described synthetic tasks to test the ability of models to retrieve and aggregate data in different languages.

In contrast to prior work, our puzzles were selected explicitly for cross-lingual evaluation and are not translations of any existing reasoning benchmark. They also involve considerably more reasoning than the typical tasks presented in Dobler et al. (2026) and Kim et al. (2025), which allows our benchmark to expose cross-lingual gaps in some of the largest frontier LLMs. Moreover, our evaluation suite and results characterize cross-

lingual reasoning gaps with task complexity and expose a more severe gap at intermediate complexity. This contrasts with prior work, such as in Bruun and Smart (2025) on multilingual Zebra puzzle reasoning, that evaluates on fixed puzzle configurations and reports no significant gaps between English and Danish. We evaluate across varying task complexities, thus capturing cross-lingual disparities that may not be apparent when evaluation is confined to a small number of fixed puzzle configurations.

The structure of our puzzles, which can be solved by repetitively applying a simple algorithm, allows us to make contact with parallel quantitative studies of errors in LLMs (Raju and Netrapalli, 2026) leading to interesting novel observations about the variation of the cross-lingual gap with complexity.

Existing multilingual benchmarks are surveyed in Uppadhyay et al. (2026). Our benchmark is immune to concerns about data leakage described there. Each puzzle instance is randomly generated from a template, allowing for a very large number of unique realizations, making memorization unlikely.

2 The XLGoBench Benchmark

Our benchmark comprises a suite of algorithmic tasks. Each of these tasks can be solved by a repeated application of a small number of atomic operations. We propose to compare model correctness across the same problem translated in various languages. We leave the language of the thinking and response space to model default. In this paper, we consider six languages apart from English: Arabic (AR), Hindi (HI), Japanese (JA), Chinese (ZH), Tamil (TA), Telugu (TE). These languages represent large speaker populations and span diverse language families and scripts. The benchmark can be easily extended to other languages.

Now that several flagship language models offer context windows of one million or longer, reasoning over information presented in context is an important skill. This is a potential source of cross-lingual gaps, owing to the volume of non-English tokens. We picked three graph walk and two graph search tasks to align with the practical challenge of navigating information graphs (in context). We illustrate the structure of the five puzzles in our evaluation suite in Figure 2. Unique instances of the puzzle are generated using a base template that can be vetted for translation errors, and then scaled to any desired complexity.

While the actual content of the puzzles is always in the respective script, the identifiers used for people's names or states are always in alphanumeric form, such as A1234. This avoids potential ambiguities — for instance, in some cultures, "Banerjee" and "Bandopadhyaya" are understood to be equivalent whereas in others "Robert" and "Bob" are used interchangeably. The use of alphanumeric identifiers also allows us to scale the number of unique entities to any desired value. We provide example prompts for each task in Appendix C, both in English and other languages.

While the puzzle content is in the respective scripts, all formatting details, algorithmic constraints, and output constraints are provided in English. This is a design choice to ensure that the gaps we observe do not arise due to known deficiencies in multilingual instruction-following (Li et al., 2026, 2024). State-of-the-art LLMs commonly use English for tool-call instructions, search augmentations, and system prompts and our benchmark detects cross-lingual reasoning gaps within this common operational setting. Moreover, while code-switching in the input could be a cause for concern as a potential trigger of errors, frontier LLMs are generally robust to such inputs. In fact, in-context code-switching has even been shown to improve performance across multiple LLMs and tasks (Yoo et al., 2025).

2.1 Graph walk problems

Sequential Linear Transforms (SLT). A traveling merchant makes a sequence of transactions with T people starting with an initial wealth of w_0. Each t^{th} transaction is uniquely identified by the customer name, along with multiplicative (a_t) and additive (b_t) adjustment factors to the merchant's wealth, and mentions the $(t+1)^{\text{th}}$ customer. The transactions are chronologically jumbled to make the task harder. The problem requires computing the final wealth w_T through a sequence of transformations $w_{t+1} = a_t w_t + b_t$. The numbers are chosen to satisfy $a_t, b_t, w_t \in [-9, 9]$.

The total number of transactions T determines the complexity of the problem. Please see Appendix C.3.1 for a full prompt.

Discrete State Automaton (DSA). Each problem specifies a discrete state automaton, i.e., states transition depending on the action. We verbalize

the problem by defining states as different rooms and actions as keys to various other connecting rooms. We then prompt for the final state given the starting room (initial state) and a sequence of keys (actions). The state transitions are all deterministic. We verify the answer through exact match on the correct final state.

We use an equal number of states and actions, which defines the complexity. Please see an example prompt in Appendix C.3.2.

Probabilistic-Discrete State Automata (Pr DSA). The setup is similar to DSA except that the state transitions are probabilistic. For a given (state, action) pair, we specify the probabilities of two possible next states for a given action. Given the initial state and a sequence of actions, we prompt the model for the eight most likely final states, and validate the response based on whether the top-8 set (order-invariant) is correctly identified. We fixed the number of actions to a constant value of 15 and varied the number of states. Puzzle complexity, thereby, is defined by the number of states. Please refer to Appendix C.3.3 for an example prompt.

2.2 Graph search problems

Graph shortest path (Graph SP). Given a social acquaintance graph with connections described as *{Person A} and {Person B} are friends*, the problem requires finding the shortest path between any two people, which is ensured to be unique by construction. Please find graph construction details in Appendix C.1. The number of edges in the graph defines complexity. Please see Appendix C.3.4 for an example prompt.

Tournament Ranking (T. Rank). Given n players with unique strengths, the problem requires ordering players according to their strength given a directed graph of game outcomes. During construction, we ensure the problem is well-specified and the solution is unique through acyclic and connected graphs, as we first construct the backbone of a Hamiltonian path and then add extra valid edges. We maintain a ratio of $1 : 3$ between number of nodes (players) and number of edges (matches) in the graph such that the complexity is defined by a single variable: the number of edges. Please see Appendix C.3.5 for an example prompt.

2.3 Validation of questions

Traditional approaches demand considerable annotation effort to ensure questions remain valid in all languages. Our setup requires considerably lower annotation effort by construction. We only need to ensure that the few templates that constitute each problem are translated correctly to all languages. The average number of templates that make up any prompt across tasks is only 5.

We confirmed the validity of translations in English and six non-English languages with native speakers who performed the following checks. (a) Template translations in various languages are semantically equivalent to that in English, (b) spot-checking with five full prompts if they make a meaningful problem of intended meaning in each language. For English, we only need step (b) since there are no translations in English. We provide further details in Appendix C.2.

2.4 Validation of responses

We instructed models to produce JSON-formatted output that was parsed using an automated script. For Pr DSA, and the two graph search problems where the solution is a list of values, the answer was marked as correct only when the entire list matched the ground-truth solution. For Pr DSA, the order of the states was ignored for response matching.

This procedure sidesteps errors that could have been introduced by using an LLM judge.

2.5 Cross-lingual gaps

The templates described above can be used to generate unique puzzles with varying complexity. The accuracy of models is expected to drop from nearly 100% at very low complexity to nearly 0% at very high complexity (Shojaee et al., 2026; Raju and Netrapalli, 2026). However, the precise rate of decline of the accuracy with complexity might vary across languages. Our benchmark detects cross-lingual gaps by identifying differences in the accuracy vs complexity curve across languages.

3 Experiments

Models. Our empirical validation spans the following models: Gemini-3-Flash (Deepmind, 2025), Gemini 3.1 Flash-Lite (Deepmind, 2026a), Gemma-4-31B (Deepmind, 2026b), GLM-5.1 (Z.ai, 2026), Sarvam-105B (sarvam.ai, 2026), Deepseek-V4-Pro (DeepSeek-AI et al., 2026).

This selection was made to ensure that (a) we cover both open and proprietary models and (b) that we cover models developed in different countries, since this might influence the linguistic distribution of the training data and evaluation process. Due to infrastructural constraints, detailed further in Appendix A.1, token limits were imposed on Sarvam-105B and Deepseek-V4-Pro.

Complexity. Our setup naturally enables studying cross-lingual gaps at various complexity levels. For each problem type, we generate problems of varying complexities such that we see perfect accuracy at the lowest complexity level and zero accuracy at the highest complexity level for English language (as seen in Figure 1). Table 1 shows the default complexity ranges we used for different tasks and also describes the parameter used to measure the complexity. The complexity range for Gemini 3.1 Flash-Lite was chosen separately as described in Appendix A.1.[3]

Sample collection. We uniformly sample 20 complexity levels from the ranges specified in Table 1. For each complexity level, we randomly generate 50 questions per language (seeds are reused across languages). This protocol is followed across all puzzle types defined in Section 2. Therefore, in total, our experiments comprise about $4 \times 20 \times 50 \times 7 \times 5 = 140,000$ prompts. (Total prompts = models $\times$ configurations $\times$ questions per configuration $\times$ languages $\times$ tasks.)

Puzzle	Complexity Measure	c_{min}	c_{max}
SLT	Chain length	100	600
DSA	Number of states	32	2090
Pr DSA	Number of states	9	61
Graph SP	Number of edges	70	720
T. Rank	Number of edges	100	400

Table 1: Complexity measure and default range of complexity values for each puzzle.

Analyzing cross-lingual gaps. The procedure above yields a picture of the decline of accuracy with complexity in each language. At a fine-grained level, any variation in the accuracy of the model at a given complexity indicates a cross-lingual gap. However, although the cross-lingual gap is, strictly speaking, measured by a function worth of information, we find it convenient to compress this into a single numerical metric using the procedure below.

We utilize the quantitative model of the variation of the accuracy of LLMs with the complexity of simple algorithmic tasks described in Raju and Netrapalli (2026). In this model, the accuracy f_ℓ in language ℓ is related to the complexity via

$$f_\ell(c) = \frac{1}{\Gamma(q_\ell/2)} \gamma\left(\frac{q_\ell}{2}, \frac{q_\ell}{2\, r_\ell\, c^2}\right) \tag{1}$$

where $\gamma(x, y)$ denotes the lower incomplete gamma function, c is the task complexity and $r_\ell, q_\ell > 0$ are language-specific parameters that can be interpreted in terms of an elementary error rate, and the number of plausible erroneous tokens at each step. This functional form naturally captures the sigmoidal decay from near-perfect accuracy at low complexity to near-zero accuracy at high complexity, and it provides a good fit to our empirical data. We computed R^2 values for the fit and found $R^2 > 0.97$ across all models and tasks.

We fit the mean accuracy as measured above, with the complexity parameter as defined in Table 1, to define the *signed max divergence (SMD)* for a language ℓ.

$$\text{SMD}(\ell) = f_{\text{En}}(c^*) - f_\ell(c^*), \tag{2}$$

where

$$c^* = \arg\max |f_{\text{En}}(c) - f_\ell(c)|. \tag{3}$$

The SMD captures the worst possible cross-lingual gap for the respective model, task and language combination when compared to English.

The SMD can potentially be subtle to interpret in cases where a language performs better than English for some complexity range and worse in another complexity range. Therefore, in Appendix A, we define a second measure called the reciprocal deviation (RD) that can be used to check for such a feature in the data. We do not find any statistically significant value for the RD in cases where we report a significant SMD.

Confidence intervals. We generate 300 samples of the accuracy at each value of the complexity using the posterior beta distribution described in Appendix C of (Raju and Netrapalli, 2026) to get multiple estimates for q_ℓ, r_ℓ. The distribution of gamma fit parameters defines the standard deviation for the SMD, which is reported below.

[3] Complexity ranges for each puzzle type were identified using one of our models (Gemini-3-Flash). c_{min} and c_{max} denote configurations with consistent near-100% and near-0% accuracy across multiple runs, enabling a full gamma fit.

4 Results

Our main results are in Table 2, which shows results for six models. We consider problems with insignificant cross-lingual gaps equally valuable and report observations on every task irrespective of their cross-lingual gaps. We report SMD values along with standard deviations per task for each model.

On average, Gemma-4-31B shows the smallest cross-lingual gaps, which is consistent with the stated multilingual emphasis of the model (Deepmind, 2026b). It is notable that GLM-5.1's performance on Chinese exceeds that of English on the Graph SP task and is on par with English on other tasks. This is consistent with the stated emphasis on training and evaluating the model on Chinese (Z.ai, 2026). On the whole, Tamil and Telugu saw the largest gaps which might arise due to their relatively low representation in the training data among the six non-English languages. Since, Sarvam-105B is the smallest model in our collection, and was tested with a limited token budget,its performance degrades at relatively-low complexity. It does not necessarily show smaller cross-lingual gaps on Indian languages than other models.

4.1 Observed cross-lingual gaps are not due to translation or other extraneous errors

As already mentioned, our puzzles are generated through templates that are human validated. However, beyond this, the following observations reinforce our confidence in the cause of gaps.

No gaps at low complexity. For lower complexity values, we observe that all languages achieve near 100% performance and gaps arise only for intermediate levels of complexity. This implies the correctness of the puzzle template and suggests that the reasoning gaps cannot be attributed to prompt ambiguities. This can also be seen in Figure 1 and in the figures of Appendix B.

Back-translation tests. We performed additional checks to rule out the possibility that gaps arise due to mild phrasing or syntactic errors between languages. For cases where a significant SMD is observed, we back-translated templates from the worst-performing non-English languages to English for Gemini-3-Flash and Gemini 3.1 Flash-Lite, and compared their performance with the original English prompt. Details of these experiments are provided in Appendix A.3. We did not find statistically significant deviations, which reinforces our confidence in the correctness of the templates.

Table 2: Signed Max Divergence results

Task	AR	HI	JA	TA	TE	ZH
SLT	9.49±4.12	3.51±4.69	8.56±4.92	10.77±4.26	18.56±4.36	−1.79±5.60
DSA	29.72±5.75	53.48±5.31	47.73±5.12	42.71±5.28	45.11±5.11	28.40±6.14
Pr DSA	6.05±4.34	5.39±5.23	−4.34±4.73	4.00±4.57	−1.50±5.33	−8.49±4.52
Graph SP	6.54±4.63	16.83±3.97	3.88±4.43	8.29±5.10	20.48±3.66	12.47±4.93
T. Rank	29.85±6.35	27.49±6.39	19.43±6.20	24.84±6.30	35.25±6.32	25.01±6.30

(a) **Gemini-3-Flash**

Task	AR	HI	JA	TA	TE	ZH
SLT	20.98±5.35	4.68±9.13	5.35±8.08	14.21±6.72	4.81±7.06	−5.37±7.31
DSA	13.48±4.99	16.40±4.97	12.94±5.41	4.31±6.65	12.65±5.13	1.06±6.14
Pr DSA	−6.13±5.98	9.15±6.03	4.53±6.00	12.50±5.80	−3.33±5.48	7.62±5.31
Graph SP	−1.88±5.06	−3.60±5.58	−4.47±5.16	6.97±4.30	7.99±4.00	5.33±4.64
T. Rank	14.06±5.23	9.83±5.27	−4.58±7.84	13.90±6.58	13.36±5.76	6.33±5.32

(b) **Gemma-4-31B-IT**

Task	AR	HI	JA	TA	TE	ZH
SLT	61.15±6.53	43.60±6.65	27.72±7.68	30.60±7.60	34.21±8.16	8.37±8.92
DSA	14.33±5.13	2.48±6.74	4.77±6.18	21.95±5.72	31.46±5.09	19.99±5.40
Pr DSA	4.46±5.53	14.10±4.16	12.20±4.56	−3.20±7.59	7.37±4.62	10.95±4.70
Graph SP	−3.68±8.58	−6.12±7.37	−4.46±7.56	−6.17±7.28	−10.50±7.10	−5.00±8.06
T. Rank	27.45±3.53	24.44±3.80	25.13±3.76	16.50±3.80	20.70±4.34	16.53±4.71

(c) **Gemini 3.1 Flash-Lite**

Task	AR	HI	JA	TA	TE	ZH
SLT	9.10±5.36	8.86±6.21	6.99±7.33	18.05±5.37	12.56±5.66	9.22±5.83
DSA	12.28±5.19	22.64±5.83	−6.79±6.30	43.89±6.09	42.59±5.45	6.15±6.46
Pr DSA	6.71±3.50	9.80±3.90	6.59±3.96	15.87±3.10	11.48±3.00	3.44±5.50
Graph SP	−6.83±6.29	−8.02±7.58	−5.95±6.97	−6.82±5.52	−9.53±6.20	−25.48±5.28
T. Rank	2.70±4.15	4.09±5.57	14.66±3.75	8.83±3.28	4.33±4.33	1.91±4.47

(d) **GLM-5-1**

Task	AR	HI	JA	TA	TE	ZH
SLT	-	57.82±7.32	-	28.86±8.37	16.70±9.31	-
DSA	-	3.41±6.84	-	5.56±7.22	−2.42±7.63	-
Pr DSA	-	6.86±5.84	-	−5.45±6.00	−5.48±5.35	-
Graph SP	-	34.26±7.09	-	24.07±8.46	24.85±12.28	-
T. Rank	-	−4.38±7.64	-	15.57±6.13	6.93±7.48	-

(e) **Sarvam-105B**

Task	AR	HI	JA	TA	TE	ZH
SLT	74.83±9.27	37.94±10.55	21.87±12.52	42.57±12.22	66.62±8.75	13.79±11.15
DSA	50.57±13.91	56.45±13.31	77.69±9.00	59.58±13.47	47.73±14.44	−27.07±18.64
Pr DSA	6.97±8.01	−6.97±11.28	16.37±6.78	12.86±6.34	−13.92±6.47	−23.21±6.34
Graph SP	20.07±11.93	−9.86±14.27	6.21±11.79	10.01±15.27	25.03±9.65	29.06±10.13
T. Rank	67.31±18.58	29.26±22.64	52.41±18.48	62.25±18.96	38.93±20.54	6.99±24.03

(f) **DeepSeek-V4-Pro**

Table 2: (contd.) Signed Max Divergence (w. En in %) $\pm\sigma$ per task and language across all evaluated models, shown along with its standard deviation. Positive values indicate superior performance by EN. We highlight cells with a significant gap (outside the reported $1.96*\sigma$) in red. Please see Section 4 for observations and Appendix A for experimental details, extended empirical results including additional measures of accuracy and cross-lingual gaps.

4.2 Gaps arise at intermediate complexity

We show accuracy with varying task complexity for different languages and models in Appendix B. We observe that all models achieve near perfect accuracy at low complexity. Similarly, the accuracy of all models collapses for sufficiently high complexity. Cross-lingual reasoning gaps are apparent only in the range of intermediate complexity. This highlights a critical drawback with static benchmarks for fast evolving models. In our benchmark,

we can easily vary the complexity range for each puzzle type and identify the ranges where cross-lingual gaps, if any, emerge.

4.3 Not all problems show gaps

Even though all our puzzles require extensive reasoning, we do not observe uniform cross-lingual gaps across tasks. Moreover, the tasks on which we observe cross-lingual gaps also vary across models. One notable exception is the T-Rank task, that exposed cross-lingual gaps for almost all models and languages.

5 Discussion

In this paper, we have introduced a method of detecting cross-lingual reasoning gaps. Unlike traditional benchmarks, generating problems with language templates enables us to scale problem complexity in any language while still preserving the accuracy of translation. Our extensive empirical tests demonstrate the utility of our benchmark, which serves to detect cross-lingual gaps in several state-of-the-art models.

One might naively have suspected that gaps can be ameliorated through longer thinking or traced back to the model expending less thinking effort in non-English languages.

However, our preliminary investigations do not indicate a clear correlation between thinking length and cross-lingual gaps in the results. This is discussed quantitatively in Appendix A.5. This suggests that cross-lingual gaps might reflect a fundamental issue that cannot be resolved merely by tweaking thinking in non-English languages. It might also be the case that the thinking process for multilingual prompts is sub-optimal.

A second observation is that most models (including GLM-5.1) tend to think in English even when the problem is given in another language. In our examination of thinking traces, we also observed the "quote and think" phenomenon described in Yong et al. (2025), where the model quotes from the original prompt but then proceeds to think in English. So, it is possible that performance in non-English languages degrades faster than English due to cross-attention between thinking tokens and the prompt tokens.

This suggests that the models could improve their accuracy by translating the question entirely into English and then ignoring the original prompt. However, we did not observe any model exploiting this procedure consistently.

Another possibility is that cross-lingual gaps might arise from inefficient tokenization in non-English languages. We have focused on the decline of accuracy with complexity, which is what matters from a user's point of view since tokenization is transparent to the user. But prompts at the same level of complexity might consume significantly different token budgets in different languages, depending on the tokenizer used by the model. From the results presented in Section A.4 of Appendix, we observe that input token length is roughly uniform across languages for Gemini models and languages with lower input token count such as Chinese do not necessarily have better accuracy. However, this remains an important issue for further investigation.

This paper focuses on the detection of cross-lingual gaps. It is important to follow this up by developing methods to ameliorate these gaps. We leave this to future work.

Acknowledgments. We are grateful to Kinshuk Vasisht for collaboration on related work, which was a precursor to this study. We would like to thank Jason Hickey and Katja Filippova for comments on a draft of this manuscript. We would like to thank Partha Talukdar, Markus Freitag, Vijai Mohan, Jan-Thorsten Peter, David Vilar Torres, Dan Deutsch, Mara Finkelstein for engaging in discussions regarding the benchmark. We thank Dinesh Tewari, Eric Baum for help in setting up the human verification of prompt templates. AI tools were used for assistance with coding, and with copyediting of the manuscript.

6 Limitations.

Our benchmark has two important limitations.

Real-world tasks. We expect that a model that displays differential performance on the synthetic tasks here will also show differential performance in real world tasks. However, the converse is not true. Our puzzles do not directly correspond to realistic applications and, for simplicity, we even removed names from the prompts.

Eventually, we would like models to be sensitive to subtleties and nuance in various languages, and also generate text as proficiently as they do in English (e.g., as evaluated in Vayani et al. (2025)). The removal of obvious cross-lingual gaps that can be detected by our benchmark is an important

step in this process. Our empirical results suggest that state-of-the-art models have not yet cleanly crossed this threshold.

Expensive evaluation. Since gaps emerge only on very long reasoning problems, the token count for our puzzles — both in terms of input and output — is high. For instance, we estimate incurring costs of approximately USD 2100 for Gemini-3-Flash and USD 6900 for GLM-5.1. These costs are expected to be higher for larger models.

It might be possible to reduce these costs by sampling thinly at extreme values of the complexity and focusing on the range of complexity where the SMD appears.

References

Sanchit Ahuja, Divyanshu Aggarwal, Varun Gumma, Ishaan Watts, Ashutosh Sathe, Millicent Ochieng, Rishav Hada, Prachi Jain, Mohamed Ahmed, Kalika Bali, and Sunayana Sitaram. 2024. MEGAVERSE: Benchmarking large language models across languages, modalities, models and tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2598–2637, Mexico City, Mexico. Association for Computational Linguistics.

Sofie Helene Bruun and Dan Saattrup Smart. 2025. Multizebralogic: A multilingual logical reasoning benchmark. *ArXiv preprint*, abs/2511.03553.

Erik Brynjolfsson, Danielle Li, and Lindsey Raymond. 2025. Generative ai at work. *The Quarterly Journal of Economics*, 140(2):889–942.

Google Deepmind. 2025. Gemini 3 flash: frontier intelligence built for speed. Google blog post.

Google Deepmind. 2026a. Gemini 3.1 flash-lite: Built for intelligence at scale. Google blog post.

Google Deepmind. 2026b. Gemma 4. Website.

DeepSeek-AI, Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chengyu Hou, Chenhao Xu, Chenze Shao, Chong Ruan, Conner Sun, and 300 others. 2026. Deepseek-v4: Towards highly efficient million-token context intelligence. *Preprint*, arXiv:2606.19348.

Zhaojun Ding, Zhengliang Liu, Hanqi Jiang, Yizhu Gao, Xiaoming Zhai, Tianming Liu, and Ninghao Liu. 2024. Foundation models for low-resource language education (vision paper). *ArXiv preprint*, abs/2412.04774.

Konstantin Dobler, Simon Lehnerer, Federico Scozzafava, Jonathan Janke, and Mohamed Ali. 2026. Multilingual reasoning gym: Multilingual scaling of procedural reasoning environments. *ArXiv preprint*, abs/2603.10793.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, and 1 others. 2025. Are we done with mmlu? In *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 5069–5096.

Akash Ghosh, Debayan Datta, Sriparna Saha, and Chirag Agarwal. 2025. A survey of multilingual reasoning in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 8920–8936, Suzhou, China. Association for Computational Linguistics.

Kunal Handa, Alex Tamkin, Miles McCain, Saffron Huang, Esin Durmus, Sarah Heck, Jared Mueller, Jerry Hong, Stuart Ritchie, Tim Belonax, and 1 others. 2025. Which economic tasks are performed with ai? evidence from millions of claude conversations. *ArXiv preprint*, abs/2503.04761.

Naomi Hausman, Oren Rigbi, and Sarit Weisburd. 2025. Generative ai's impact on student achievement and implications for worker productivity. *Available at SSRN 5393516*.

Andre He, Nathaniel Weir, Kaj Bostrom, Allen Nie, Darion Cassel, Sam Bayless, and Huzefa Rangwala. 2026. Resyn: Autonomously scaling synthetic environments for reasoning models. *ArXiv preprint*, abs/2602.20117.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Xu Huang, Wenhao Zhu, Hanxu Hu, Conghui He, Lei Li, Shujian Huang, and Fei Yuan. 2025. Benchmax: A comprehensive multilingual evaluation suite for large language models. *ArXiv preprint*, abs/2502.07346.

Yekyung Kim, Jenna Russell, Marzena Karpinska, and Mohit Iyyer. 2025. One ruler to measure them all: Benchmarking multilingual long-context language models. *ArXiv preprint*, abs/2503.01996.

Yizhi Li, Ge Zhang, Xingwei Qu, Jiali Li, Zhaoqun Li, Noah Wang, Hao Li, Ruibin Yuan, Yinghao Ma, Kai Zhang, and 1 others. 2024. Cif-bench: A chinese instruction-following benchmark for evaluating the generalizability of large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12431–12446.

Zhenyu Li, Kehai Chen, Yunfei Long, Xuefeng Bai, Yaoyin Zhang, Xuchen Wei, Juntao Li, and Min Zhang. 2026. Xifbench: Evaluating large language models on multilingual instruction following. *Advances in Neural Information Processing Systems*, 38.

Huanyu Liu, Jia Li, Hao Zhu, Kechi Zhang, Yihong Dong, and Ge Li. 2025. SATURN: SAT-based reinforcement learning to unleash LLMs reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Microsoft AI Economy Institute. 2026. Global ai adoption in 2025—a widening digital divide. https://www.microsoft.com/en-us/corporate-responsibility/topics/ai-economy-institute/reports/global-ai-adoption-2025/. See section: "South Korea's AI Surge: Policy, Models, and a Viral Cultural Moment".

Amit Misra, Jane Wang, Scott McCullers, Kevin White, and Juan Lavista Ferres. 2025. Measuring ai diffusion: A population-normalized metric for tracking global ai usage. *ArXiv preprint*, abs/2511.02781.

Xuan-Phi Nguyen, Mahani Aljunied, Shafiq Joty, and Lidong Bing. 2024. Democratizing llms for low-resource languages by leveraging their english dominant abilities with linguistically-diverse prompts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3501–3516.

Shakked Noy and Whitney Zhang. 2023. Experimental evidence on the productivity effects of generative artificial intelligence. *Science*, 381(6654):187–192.

OpenAI. 2026. Graphwalks. Huggingface.co.

Jan-Thorsten Peter, David Vilar, Tobias Domhan, Dan Malkin, and Markus Freitag. 2025. Mind the gap... or not? how translation errors and evaluation details skew multilingual results. *ArXiv preprint*, abs/2511.05162.

Suvrat Raju and Praneeth Netrapalli. 2026. A model of errors in transformers. In *Forty-third International Conference on Machine Learning*.

sarvam.ai. 2026. Open sourcing sarvam 30b and 105b. Sarvam..ai Blogpost.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, Dipanjan Das, and Jason Wei. 2023. Language models are multilingual chain-of-thought reasoners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Parshin Shojaee, Seyed Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. 2026. The illusion of thinking: Understanding the strengths and limitations

of reasoning models via the lens of problem complexity. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Shivalika Singh, Angelika Romanou, Clémentine Fourrier, David Ifeoluwa Adelani, Jian Gang Ngui, Daniel Vila-Suero, Peerat Limkonchotiwat, Kelly Marchisio, Wei Qi Leong, Yosephine Susanto, and 1 others. 2025. Global mmlu: Understanding and addressing cultural and linguistic biases in multilingual evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18761–18799.

Marinko Skare, Beata Gavurova, and Sanja Blažević Burić. 2024. Artificial intelligence and wealth inequality: A comprehensive empirical exploration of socioeconomic implications. *Technology in society*, 79:102719.

Zafir Stojanovski, Oliver Stanley, Joe Sharratt, Richard Jones, Abdulhakeem Adefioye, Jean Kaddour, and Andreas Köpf. 2025. Reasoning gym: Reasoning environments for reinforcement learning with verifiable rewards. *ArXiv preprint*, abs/2505.24760.

Jianheng Tang, Qifan Zhang, Yuhan Li, Nuo Chen, and Jia Li. 2024. Grapharena: Evaluating and exploring large language models on graph computation. *ArXiv preprint*, abs/2407.00379.

UNDP. 2025. The next great divergence. Technical report, United Nations Development Program.

Manan Uppadhyay, Himanshu Beniwal, Prashant Kodali, and Sunayana Sitaram. 2026. The state and fate of multilingual, contextual evaluation in the nlp world. Microsoft.com.

Ashmal Vayani, Dinura Dissanayake, Hasindri Watawana, Noor Ahsan, Nevasini Sasikumar, Omkar Thawakar, Henok Biadglign Ademtew, Yahya Hmaiti, Amandeep Kumar, Kartik Kukreja, and 1 others. 2025. All languages matter: Evaluating lmms on culturally diverse 100 languages. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19565–19575.

Anjiang Wei, Yuheng Wu, Yingjia Wan, Tarun Suresh, Huanmi Tan, Zhanke Zhou, Sanmi Koyejo, Ke Wang, and Alex Aiken. 2025. Satbench: Benchmarking llms' logical reasoning via automated puzzle generation from sat formulas. *ArXiv preprint*, abs/2505.14615.

Minghao Wu, Weixuan Wang, Sinuo Liu, Huifeng Yin, Xintong Wang, Yu Zhao, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. 2025. The bitter lesson learned from 2,000+ multilingual benchmarks. *ArXiv preprint*, abs/2504.15521.

Weihao Xuan, Rui Yang, Heli Qi, Qingcheng Zeng, Yunze Xiao, Aosong Feng, Dairui Liu, Yun Xing, Junjue Wang, Fan Gao, and 1 others. 2025. Mmlu-prox: A multilingual benchmark for advanced large language model evaluation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1513–1532.

Zheng-Xin Yong, M Farid Adilazuarda, Jonibek Mansurov, Ruochen Zhang, Niklas Muennighoff, Carsten Eickhoff, Genta Indra Winata, Julia Kreutzer, Stephen H Bach, and Alham Fikri Aji. 2025. Crosslingual reasoning through test-time scaling. *arXiv preprint arXiv:2505.05408*.

Haneul Yoo, Jiho Jin, Kyunghyun Cho, and Alice Oh. 2025. Code-switching in-context learning for cross-lingual transfer of large language models. *ArXiv preprint*, abs/2510.05678.

Z.ai. 2026. Glm-5.1: Towards long-horizon tasks. Z.ai.

Yifan Zhang. 2024. Training and evaluating language models with template-based data generation. *ArXiv preprint*, abs/2411.18104.

A Extended Empirical Results

In this section, we present the full extended results of our empirical experiments. This section is divided into several subsections. In Section A.1, we describe the complexity selection for Gemini-3.1-Flash-Lite, DeepSeek-V4-Pro and Sarvam-105B. In Section A.2, we describe some additional measures of accuracy and cross-lingual gaps and present detailed tables of results. In Section A.3, we present results for the back-translation experiments described in Section 4.1. In Section A.5, we present results showing the lack of clear correlations between thinking length and accuracy, as discussed in Section A.5.

A.1 Complexity selection

The complexity measures for Gemini-3-Flash, Gemma-4-31B, and GLM-5.1 were selected as described in section 3. But, since Gemini-3.1-Flash-Lite model is a smaller scale model than the other models in our evaluation set we chose to include some easier complexity levels for its evaluation to get the full accuracy curve starting from 100%. We present the complexity bounds in the Table 3. Note that DSA and Pr DSA had the same complexity levels as the default. Separate complexity levels were selected for SLT, Graph SP and T. Rank. For SLT we chose to include easier levels starting from chain length of 10; for Graph SP, we decided to sample more questions from the complexity range of 70-200(total edges); and for T. Rank we decided to sample questions only from the range of 25-150(total edges).

Puzzle	Complexity Measure	$c_{\min}$	$c_{\max}$
SLT*	Chain length	10	600
DSA	Number of states	32	2090
Pr DSA	Number of states	9	61
Graph SP*	Number of edges	70	200
T. Rank*	Number of edges	25	150

Table 3: Complexity measure and range of complexity values for each puzzle for Gemini 3.1 Flash-Lite.

Similarly for Sarvam-105B and DeepSeek-V4-Pro we provide in Tables 4 and 5 the complexity bounds chosen to evaluate respective models. For these models we followed a slightly different experimental setup, requiring us to choose easier set of complexity levels. For Sarvam-105B , owing to API limits (max new tokens set to 4096) we chose to perform the evaluation in 2-stages. In Stage-1 we prompted the model with thinking level set to low to allow it to reason over the question. Following that if the model was unable to provide the response in Stage-1 we would prompt it again in Stage-2 with this thinking trace appended to the question and elicit response with thinking level set to None. For DeepSeek-V4-Pro, we hosted the model on a dedicated endpoint which had a lower token throughput and so we set the max new tokens to 16,384 to prevent overthinking and to limit costs. In this setup, we marked all responses which failed to produce the response within this budget as incorrect.

Puzzle	Complexity Measure	$c_{\min}$	$c_{\max}$
SLT*	Chain length	2	20
DSA*	Number of states	4	48
Pr DSA*	Number of states	4	20
Graph SP*	Number of edges	16	666
T. Rank*	Number of edges	12	50

Table 4: Complexity measure and range of complexity values for each puzzle for Sarvam-105B.

Puzzle	Complexity Measure	$c_{\min}$	$c_{\max}$
SLT	Chain length	10	600
DSA	Number of states	32	2090
Pr DSA*	Number of states	6	61
Graph SP*	Number of edges	13	936
T. Rank*	Number of edges	15	340

Table 5: Complexity measure and range of complexity values for each puzzle for DeepSeek-V4-Pro.

A.2 Additional Accuracy Measures

The SMD reported in the main text is a sufficient indicator of the presence of a cross-lingual gap. However, one might be concerned that it obscures cases where some language displays superior performance to English at one complexity and inferior performance at another.

Therefore, it is convenient to define two supplementary measures. We define the average accuracy by

$$\text{Avg Acc}(\ell) = \frac{1}{c_{\max} - c_{\min}} \int_{c_{\min}}^{c_{\max}} f_\ell(c)\, dc. \quad (4)$$

We report the Average Accuracy in Table 6.

We also define the *Reciprocal Divergence* (RD) by

$$\mathrm{RD}(\ell) = \min_c \mathrm{sign}(\mathrm{SMD}(\ell))\,(f_{\mathrm{En}}(c) - f_\ell(c))\,. \tag{5}$$

If the SMD is positive, which means that at the point of the largest cross-lingual gap English displays superior performance, the RD tells us the best relative-performance of the non-English language. If the SMD is negative (as in the case of GLM-5.1), the RD tells us the best relative-performance of English.

The RD is reported in Table 7. This Table shows the SMD is a good representative of the holistic performance, since the RD is insignificant in all cases. Therefore the peak difference in the accuracy is also an indicator of the difference in accuracy at other complexity values.

Finally, Tables 10, 11, 12 and 13 expand the SMD results displayed in Table 2 by presenting an estimate of the accuracy of English and the non-English languages at the point of maximum cross-lingual gap. The values in this table are generated as follows. For all the metrics reported, we sample $N = 300$ times from the observed distribution of accuracies using the posterior distribution of (Raju and Netrapalli, 2026) to generate the accuracy curve, both for English and the other language. Each sample, i, yields a value c_i^* as defined in (3). Across these values of c_i^*, we now compute

$$\begin{aligned}\langle f_{\mathrm{En}}\rangle &= \frac{1}{N}\sum_i f_{\mathrm{En}}(c_i^*);\\ \langle f_\ell\rangle &= \frac{1}{N}\sum_i f_\ell(c_i^*);\\ \langle c^*\rangle &= \frac{1}{N}\sum_i c_i^*;\end{aligned} \tag{6}$$

together with their corresponding standard deviations.

A.3 Back-Translation Experiments

As described in section 4.1, we performed some back-translation experiments to rule out translation errors. For Gemini-3-Flash and Gemini 3.1 Flash Lite, in cases where a significant SMD was observed, we picked the worst-performing language and back-translated its prompt template to English.

In all cases, the back-translated prompts were semantically identical to the original prompt and, in addition had $\geq$ 95% word-level overlap with the original prompt in English.

Nevertheless, to rule out the possibility that small variations in the prompt lead to variations in accuracy, we evaluated the accuracy of the model using this back-translated prompt on the two original complexity values c_1, c_2 that were closest to the value of c^* defined in equation (3) with $c_1 < c^* < c_2$. This yielded accuracy values $f^b_{\mathrm{En}}(c_1)$ and $f^b_{\mathrm{En}}(c_2)$. This allows us to compute

$$\delta_i = f^b_{\mathrm{En}}(c_i) - f_{\mathrm{En}}(c_i), \tag{7}$$

where $f_{\mathrm{En}}(c_i)$ are the original accuracies and c_i runs over c_1 and c_2. With the original standard deviations, σ_1 and σ_2, we can then finally compute

$$\delta = \frac{1}{(c_2 - c_1)}\left(|c_2 - c^*|\frac{\delta_1^2}{\sigma_1^2} + |c_1 - c^*|\frac{\delta_2^2}{\sigma_2^2}\right) \tag{8}$$

This measure δ yields a weighted χ^2 value, and values of $\delta \gg 1$ would indicate that back-translation has induced a significant difference in accuracy.

We present the observed value of δ in Table 16. These show that there are no significant gaps between the original prompt performance and back-translated prompt performance.

We note that for Gemini-3.1-Flash-Lite, the tournament ranking prompt was an exact match to the original prompt. The Pr DSA task with Gemini-3-Flash and the Graph SP tasks with Gemini 3.1 Flash Lite did not display any significant SMD values at all. For these reasons, the corresponding entries in Table 16 have been left blank.

A.4 Input Token length Comparision

In this section we present the input token lengths recorded per language per task in table 17 for Gemini-3-Flash. We note that across all languages the input token lengths are within the same range, isolating cross lingual reasoning gaps from input token budget.

A.5 Correlation between Thinking Budget and Cross-Lingual gaps

To verify if the thought length is correlated with the cross-lingual gaps, we did the following analysis. We ranked the language based on SMD and then computed the Spearman's rank correlation with the rankings observed on the average thought usage. This average was computed over all the data points for that specific model x language x

Task	EN	AR	HI	JA	TA	TE	ZH
SLT	45.18±1.13	41.18±1.17	43.70±1.13	43.58±1.31	42.45±1.30	40.30±1.33	45.65±1.26
DSA	52.88±1.74	40.52±1.95	29.61±1.48	33.08±1.54	34.81±1.90	33.32±2.13	41.75±2.03
Pr DSA	31.74±1.16	29.97±1.05	31.02±1.21	32.93±1.25	30.28±0.98	32.27±1.09	33.64±1.00
Graph SP	26.01±1.40	22.40±1.13	17.74±1.27	23.93±1.34	21.40±1.26	16.32±0.92	19.46±1.36
T. Rank	42.69±1.15	35.83±0.81	36.39±0.84	38.33±0.83	37.01±0.86	34.51±0.75	37.03±0.81

(a) **Gemini-3-Flash**

Task	EN	AR	HI	JA	TA	TE	ZH
SLT	28.88±0.87	24.32±0.75	29.09±1.07	27.97±1.03	27.41±0.95	28.88±0.97	30.53±0.95
DSA	24.81±1.20	21.59±1.13	20.33±0.91	22.53±1.21	24.32±1.37	21.77±1.17	24.99±1.34
Pr DSA	30.14±1.21	32.84±1.31	29.51±1.21	30.30±1.05	29.45±1.12	31.55±1.15	28.27±1.25
Graph SP	17.03±0.92	17.92±1.07	18.81±1.04	19.29±1.07	16.11±1.07	14.99±1.16	16.00±1.06
T. Rank	23.39±1.11	19.09±0.94	20.31±0.89	23.19±0.89	19.13±0.96	19.29±1.06	21.34±0.92

(b) **Gemma-4-31B-IT**

Task	EN	AR	HI	JA	TA	TE	ZH
SLT*	27.84±0.95	16.84±0.60	20.77±0.71	24.48±0.85	23.13±0.76	22.53±0.96	26.35±0.89
DSA	18.83±0.93	16.10±0.93	17.96±1.02	17.36±1.07	13.91±0.83	13.33±0.79	16.34±0.84
Pr DSA	21.04±0.91	19.56±0.83	17.12±0.84	17.65±0.86	20.32±0.86	18.67±0.77	18.48±0.91
Graph SP*	14.71±1.47	15.59±1.15	16.25±1.00	15.82±0.89	16.26±0.90	17.50±1.08	15.96±1.20
T. Rank*	52.93±0.90	45.21±0.70	45.73±0.76	45.45±0.80	47.93±0.81	47.09±0.92	47.91±1.02

(c) **Gemini 3.1 Flash-Lite**

Task	EN	AR	HI	JA	TA	TE	ZH
SLT	32.28±0.93	30.06±0.87	30.13±0.94	31.44±1.07	27.73±0.86	28.99±0.76	30.20±0.90
DSA	24.95±1.07	21.68±0.99	19.12±0.91	25.89±0.97	15.05±0.79	15.37±0.76	24.48±1.14
Pr DSA	44.15±1.61	39.36±1.49	37.25±1.58	39.52±1.51	33.30±1.25	36.13±1.23	42.26±1.57
Graph SP	32.88±1.14	34.91±1.58	37.40±1.49	32.99±1.52	35.62±1.47	36.70±1.59	45.27±1.52
T. Rank	40.75±1.06	40.07±1.00	40.68±1.32	34.81±1.16	36.78±0.94	38.74±1.02	39.93±1.19

(d) **GLM-5-1**

Table 6: Average Accuracy (%) as calculated from the gamma fits. Higher is better. * denotes custom complexity levels used as described in Appendix A.1.

Task	AR	HI	JA	TA	TE	ZH
SLT	$-0.01_{\pm 1.42}$	$0.00_{\pm 1.22}$	$-1.49_{\pm 1.46}$	$-0.83_{\pm 1.30}$	$-1.44_{\pm 1.42}$	$-1.66_{\pm 1.39}$
DSA	$0.00_{\pm 2.25}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.43}$	$0.00_{\pm 1.64}$	$0.00_{\pm 0.59}$	$0.00_{\pm 2.27}$
Pr DSA	$0.00_{\pm 1.06}$	$-0.87_{\pm 1.45}$	$-0.04_{\pm 1.24}$	$0.00_{\pm 1.08}$	$-1.22_{\pm 1.36}$	$-0.35_{\pm 1.27}$
Graph SP	$0.04_{\pm 1.11}$	$0.19_{\pm 0.71}$	$0.08_{\pm 0.92}$	$-0.62_{\pm 1.48}$	$0.23_{\pm 0.73}$	$-0.55_{\pm 1.66}$
T. Rank	$0.00_{\pm 0.13}$	$0.00_{\pm 0.20}$	$0.00_{\pm 0.37}$	$0.00_{\pm 0.20}$	$0.00_{\pm 0.03}$	$-0.00_{\pm 0.26}$

(a) **Gemini-3-Flash**

Task	AR	HI	JA	TA	TE	ZH
SLT	$0.00_{\pm 0.93}$	$-2.14_{\pm 2.32}$	$-0.02_{\pm 1.68}$	$-1.37_{\pm 1.74}$	$-1.62_{\pm 2.12}$	$-0.40_{\pm 2.38}$
DSA	$0.00_{\pm 0.97}$	$0.00_{\pm 0.75}$	$-0.10_{\pm 1.36}$	$-0.30_{\pm 1.80}$	$0.00_{\pm 0.78}$	$-0.69_{\pm 1.63}$
Pr DSA	$-0.01_{\pm 1.41}$	$-2.35_{\pm 1.52}$	$-2.02_{\pm 1.22}$	$-3.50_{\pm 1.57}$	$-0.00_{\pm 1.12}$	$-0.11_{\pm 0.96}$
Graph SP	$-0.12_{\pm 0.86}$	$-1.71_{\pm 1.40}$	$-1.02_{\pm 1.52}$	$-0.37_{\pm 0.86}$	$0.12_{\pm 0.67}$	$-0.05_{\pm 0.66}$
T. Rank	$0.05_{\pm 0.89}$	$0.05_{\pm 1.79}$	$-2.85_{\pm 2.34}$	$0.05_{\pm 3.12}$	$0.05_{\pm 1.96}$	$0.04_{\pm 1.45}$

(b) **Gemma-4-31B-IT**

Task	AR	HI	JA	TA	TE	ZH
SLT*	$0.00_{\pm 0.15}$	$-0.29_{\pm 0.88}$	$-1.58_{\pm 1.66}$	$-0.28_{\pm 0.94}$	$-0.31_{\pm 1.09}$	$0.00_{\pm 1.50}$
DSA	$0.00_{\pm 0.81}$	$-0.35_{\pm 1.57}$	$-0.00_{\pm 1.53}$	$0.00_{\pm 0.49}$	$0.00_{\pm 0.50}$	$-0.43_{\pm 1.11}$
Pr DSA	$-0.00_{\pm 1.51}$	$0.11_{\pm 0.47}$	$0.09_{\pm 1.01}$	$-3.16_{\pm 2.02}$	$-0.00_{\pm 1.53}$	$-0.00_{\pm 0.59}$
Graph SP*	$0.00_{\pm 2.64}$	$-0.17_{\pm 2.13}$	$0.00_{\pm 2.40}$	$0.00_{\pm 2.05}$	$-0.48_{\pm 2.05}$	$0.00_{\pm 2.25}$
T. Rank*	$-0.00_{\pm 0.44}$	$0.00_{\pm 0.07}$	$0.00_{\pm 0.15}$	$0.00_{\pm 0.27}$	$-0.04_{\pm 0.83}$	$0.00_{\pm 0.62}$

(c) **Gemini 3.1 Flash-Lite**

Task	AR	HI	JA	TA	TE	ZH
SLT	$0.00_{\pm 0.64}$	$0.00_{\pm 1.30}$	$-0.64_{\pm 0.87}$	$0.00_{\pm 0.28}$	$-0.84_{\pm 2.50}$	$0.00_{\pm 0.64}$
DSA	$0.00_{\pm 0.77}$	$0.00_{\pm 0.68}$	$-0.28_{\pm 0.96}$	$0.00_{\pm 0.38}$	$0.00_{\pm 0.00}$	$-0.77_{\pm 1.53}$
Pr DSA	$0.01_{\pm 1.22}$	$0.02_{\pm 1.45}$	$0.06_{\pm 1.18}$	$0.02_{\pm 0.67}$	$0.05_{\pm 0.94}$	$-1.48_{\pm 1.85}$
Graph SP	$-0.06_{\pm 1.05}$	$-2.97_{\pm 2.80}$	$-1.92_{\pm 1.33}$	$0.00_{\pm 1.05}$	$0.00_{\pm 1.03}$	$0.00_{\pm 0.05}$
T. Rank	$-0.26_{\pm 1.18}$	$-1.99_{\pm 1.71}$	$0.03_{\pm 0.49}$	$0.00_{\pm 0.63}$	$-0.00_{\pm 0.94}$	$0.00_{\pm 1.10}$

(d) **GLM-5-1**

Table 7: Reciprocal Divergence(%).

Task	EN	HI	TA	TE
SLT	$34.81_{\pm 2.01}$	$13.75_{\pm 1.72}$	$25.73_{\pm 1.88}$	$30.75_{\pm 2.15}$
DSA	$60.91_{\pm 1.50}$	$59.58_{\pm 1.30}$	$61.04_{\pm 1.33}$	$61.53_{\pm 1.64}$
Pr DSA	$55.15_{\pm 1.62}$	$52.07_{\pm 1.56}$	$56.19_{\pm 1.53}$	$56.96_{\pm 1.65}$
Graph SP	$51.70_{\pm 1.63}$	$42.77_{\pm 2.35}$	$46.28_{\pm 2.34}$	$45.32_{\pm 3.14}$
T. Rank	$46.65_{\pm 1.02}$	$47.69_{\pm 1.07}$	$44.25_{\pm 1.18}$	$45.27_{\pm 1.13}$

(a) **Average Accuracy (%) from Gamma Fits**

Task	HI	TA	TE
SLT	$2.20_{\pm 2.47}$	$-0.26_{\pm 2.20}$	$-1.32_{\pm 3.17}$
DSA	$-0.00_{\pm 1.80}$	$-3.89_{\pm 2.42}$	$-1.35_{\pm 2.07}$
Pr DSA	$-1.81_{\pm 2.39}$	$-2.10_{\pm 2.26}$	$-0.88_{\pm 1.89}$
Graph SP	$-5.34_{\pm 3.23}$	$-4.59_{\pm 3.20}$	$-3.43_{\pm 3.52}$
T. Rank	$-0.00_{\pm 2.14}$	$-1.99_{\pm 2.59}$	$-0.03_{\pm 1.97}$

(b) **Reciprocal Divergence (%)**

Table 8: Sarvam-105B results across all tasks and Indic languages (HI, TA, TE). (a) Average Accuracy (%) as computed from gamma fits (AUC Normalized). (b) Reciprocal Divergence (%).

Task	EN	AR	HI	JA	TA	TE	ZH
SLT	$31.11_{\pm 0.75}$	$21.82_{\pm 0.44}$	$26.86_{\pm 0.79}$	$29.23_{\pm 0.84}$	$26.20_{\pm 0.67}$	$21.96_{\pm 0.65}$	$29.54_{\pm 0.72}$
DSA	$18.47_{\pm 0.64}$	$15.48_{\pm 0.53}$	$14.93_{\pm 0.53}$	$12.14_{\pm 0.51}$	$14.74_{\pm 0.47}$	$15.72_{\pm 0.43}$	$20.48_{\pm 0.76}$
Pr DSA	$20.01_{\pm 0.85}$	$18.79_{\pm 0.71}$	$19.84_{\pm 0.60}$	$17.38_{\pm 0.65}$	$18.10_{\pm 0.69}$	$22.81_{\pm 0.94}$	$23.30_{\pm 0.80}$
Graph SP	$53.54_{\pm 2.59}$	$45.92_{\pm 1.87}$	$54.13_{\pm 1.62}$	$51.05_{\pm 1.41}$	$51.03_{\pm 1.02}$	$44.19_{\pm 1.23}$	$42.93_{\pm 1.43}$
T. Rank	$43.43_{\pm 1.88}$	$34.04_{\pm 0.76}$	$41.49_{\pm 1.27}$	$36.88_{\pm 0.80}$	$34.75_{\pm 0.79}$	$39.43_{\pm 0.72}$	$43.25_{\pm 1.40}$

(a) **Average Accuracy (%) from Gamma Fits**

Task	AR	HI	JA	TA	TE	ZH
SLT	$0.00_{\pm 0.04}$	$-0.48_{\pm 2.51}$	$-1.87_{\pm 3.45}$	$0.00_{\pm 0.68}$	$-0.10_{\pm 0.78}$	$0.00_{\pm 2.65}$
DSA	$-0.80_{\pm 2.07}$	$-0.75_{\pm 1.39}$	$-0.50_{\pm 1.12}$	$-0.09_{\pm 0.87}$	$-0.35_{\pm 1.55}$	$-9.55_{\pm 8.17}$
Pr DSA	$0.00_{\pm 3.31}$	$-5.56_{\pm 3.70}$	$-0.02_{\pm 2.82}$	$-0.06_{\pm 2.35}$	$-0.00_{\pm 2.07}$	$-0.07_{\pm 1.82}$
Graph SP	$0.00_{\pm 2.19}$	$-4.05_{\pm 2.98}$	$-0.25_{\pm 2.67}$	$-6.34_{\pm 3.95}$	$0.00_{\pm 1.42}$	$0.00_{\pm 0.59}$
T. Rank	$-0.07_{\pm 0.64}$	$-7.97_{\pm 4.91}$	$-3.35_{\pm 2.78}$	$-0.71_{\pm 1.22}$	$-3.70_{\pm 2.90}$	$-3.45_{\pm 3.42}$

(b) **Reciprocal Divergence (%)**

Table 9: DeepSeek-V4-Pro results across all tasks and languages. (a) Average Accuracy (%) as computed from gamma fits (AUC Normalized). (b) Reciprocal Divergence (%).

Task	Metric	AR	HI	JA	TA	TE	ZH
SLT	c^*	326±35	303±52	254±36	258±25	251±12	410±54
	f_{EN}	41.4±15.7	52.6±22.0	78.9±16.3	76.9±12.0	80.4±5.9	15.6±23.6
	f_ℓ	31.9±15.4	49.1±23.0	70.3±14.3	66.1±10.7	61.8±5.2	17.4±21.7
DSA	c^*	721±73	667±32	651±30	680±45	692±47	704±80
	f_{EN}	79.8±7.5	85.6±4.8	87.3±4.0	84.3±4.7	83.0±5.9	81.6±8.2
	f_ℓ	50.1±5.1	32.1±3.9	39.5±3.3	41.6±4.2	37.9±5.3	53.2±5.9
Pr DSA	c^*	21±4	19±5	21±5	22±4	33±5	21±3
	f_{EN}	67.8±18.5	77.5±21.9	64.5±21.1	57.4±20.9	16.6±25.6	65.8±15.7
	f_ℓ	61.7±16.8	72.1±19.2	68.9±22.4	53.5±19.7	18.1±22.6	74.3±17.1
Graph SP	c^*	211±53	192±22	195±60	245±51	189±18	230±36
	f_{EN}	42.5±17.4	49.2±9.2	48.2±19.9	32.8±16.2	50.7±8.2	36.5±11.4
	f_ℓ	36.0±19.2	32.3±8.2	44.3±20.0	24.5±17.8	30.3±6.4	24.1±12.7
T. Rank	c^*	212±6	210±6	212±8	211±7	208±4	209±6
	f_{EN}	68.9±9.1	71.4±9.2	68.9±11.6	69.7±10.1	73.8±7.6	72.2±9.3
	f_ℓ	39.0±7.0	43.9±7.2	49.4±10.2	44.9±8.5	38.6±5.6	47.2±7.9

Table 10: **Gemini-3-Flash.** Here we present the $\langle c^* \rangle$ at which SMD values were reported along with $\langle f_{EN} \rangle$ and $\langle f_\ell \rangle$ at the respective complexities over 300 bootstraps.

Task	Metric	AR	HI	JA	TA	TE	ZH
SLT	c^*	155±15	131±35	146±30	133±23	132±36	204±40
	f_{EN}	59.6±13.1	81.8±24.0	68.2±20.3	79.8±15.6	80.8±24.0	25.8±25.5
	f_ℓ	38.6±10.2	77.1±18.4	62.8±20.0	65.6±12.1	76.0±20.1	31.1±22.2
DSA	c^*	309±57	346±47	285±53	285±102	312±54	260±103
	f_{EN}	65.5±15.3	53.6±13.4	74.0±14.2	74.0±23.9	64.5±14.8	82.3±23.2
	f_ℓ	52.1±14.0	37.2±12.0	61.1±12.4	69.7±21.7	51.8±13.8	81.2±23.2
Pr DSA	c^*	26±5	18±4	18±4	18±3	25±4	20±4
	f_{EN}	33.2±21.2	80.9±18.1	82.1±20.6	83.4±14.4	36.0±21.2	71.5±18.4
	f_ℓ	39.3±19.6	71.7±14.7	77.6±18.3	70.8±11.5	39.3±19.7	63.9±16.9
Graph SP	c^*	199±47	234±60	220±58	114±27	125±24	119±31
	f_{EN}	27.8±22.3	18.9±27.2	22.0±25.6	74.7±13.8	66.9±13.3	70.8±16.0
	f_ℓ	29.7±20.5	22.4±23.0	26.5±22.2	67.7±12.3	58.9±12.5	65.4±15.0
T. Rank	c^*	160±13	163±17	138±25	161±15	161±14	169±21
	f_{EN}	53.5±14.4	50.0±17.4	79.0±22.5	52.8±15.3	52.8±15.2	44.1±20.2
	f_ℓ	39.4±12.2	40.2±17.4	83.6±27.6	38.9±16.1	39.4±14.6	37.7±20.3

Table 11: **Gemma-4-31B-IT.** Here we present the $\langle c^* \rangle$ at which SMD values were reported along with $\langle f_{EN} \rangle$ and $\langle f_\ell \rangle$ at the respective complexities over 300 bootstraps.

Task	Metric	AR	HI	JA	TA	TE	ZH
SLT	c^*	122±4	126±5	128±5	131±6	129±6	147±26
	f_{EN}	88.0±3.7	85.2±3.8	83.2±4.4	81.2±6.5	82.2±6.3	65.1±20.1
	f_ℓ	26.8±3.9	41.6±3.9	55.5±5.1	50.6±4.9	48.0±5.2	56.8±19.4
DSA	c^*	271±55	461±104	370±105	288±40	242±11	242±24
	f_{EN}	70.0±14.0	27.3±21.1	42.8±22.0	64.8±11.6	79.4±4.2	79.4±6.7
	f_ℓ	55.7±11.6	24.8±22.6	38.0±22.5	42.8±9.0	47.9±3.8	59.4±5.5
Pr DSA	c^*	20±4	17±2	17±2	14±4	20±3	16±2
	f_{EN}	39.4±20.4	61.1±12.2	61.1±15.4	82.5±22.7	41.3±16.8	69.5±14.8
	f_ℓ	34.9±22.4	47.0±10.2	48.9±13.3	85.7±28.0	34.0±17.9	58.6±13.0
Graph SP	c^*	89±9	95±9	91±10	92±10	96±9	94±9
	f_{EN}	45.9±23.5	27.7±24.1	38.5±24.4	36.9±24.1	25.2±22.6	30.4±23.7
	f_ℓ	49.6±25.3	33.9±22.4	42.9±25.1	43.1±24.4	35.7±20.4	35.4±24.3
T. Rank	c^*	87±2	86±3	84±3	85±4	90±3	87±4
	f_{EN}	53.9±6.8	57.3±8.0	62.1±7.7	59.4±10.2	47.3±9.2	53.9±10.6
	f_ℓ	26.5±6.4	32.9±7.0	37.0±7.5	42.9±9.2	26.6±9.0	37.4±10.8

Table 12: **Gemini 3.1 Flash-Lite.** Here we present the $\langle c^* \rangle$ at which SMD values were reported along with $\langle f_{EN} \rangle$ and $\langle f_\ell \rangle$ at the respective complexities over 300 bootstraps.

Task	Metric	AR	HI	JA	TA	TE	ZH
SLT	c^*	167±24	167±28	151±25	170±15	207±21	163±27
	f_{EN}	62.7±15.5	62.7±16.8	75.9±15.8	60.9±11.1	35.2±12.5	66.5±16.5
	f_ℓ	53.6±13.8	53.9±16.2	68.9±13.7	42.8±8.1	22.6±13.8	57.3±14.8
DSA	c^*	302±51	296±33	282±72	259±24	256±10	256±83
	f_{EN}	60.4±14.9	62.4±11.5	67.5±17.0	75.8±9.0	76.8±5.3	76.8±20.7
	f_ℓ	48.1±13.7	39.7±9.1	74.3±18.9	31.9±7.1	34.2±4.5	70.7±17.5
Pr DSA	c^*	30±8	30±7	27±8	29±4	28±5	43±11
	f_{EN}	42.5±18.3	44.7±16.1	50.2±19.0	46.7±10.4	47.2±12.4	22.9±23.6
	f_ℓ	35.8±18.1	34.9±16.0	43.6±18.5	30.8±9.1	35.7±11.3	19.5±27.1
Graph SP	c^*	212±54	377±90	190±65	226±63	229±54	259±20
	f_{EN}	64.7±17.4	18.0±27.5	75.2±19.9	58.1±18.9	57.0±16.0	45.1±5.4
	f_ℓ	71.5±18.4	26.0±21.1	81.2±23.1	64.9±19.2	66.5±17.7	70.6±8.8
T. Rank	c^*	183±34	172±41	192±12	211±21	227±30	204±33
	f_{EN}	73.8±23.4	82.2±27.4	66.3±9.8	50.4±14.9	39.0±20.7	56.2±22.3
	f_ℓ	71.1±21.6	78.1±23.2	51.6±8.9	41.6±14.0	34.6±21.1	54.3±22.6

Table 13: **GLM-5-1.** Here we present the $\langle c^* \rangle$ at which SMD values were reported along with $\langle f_{EN} \rangle$ and $\langle f_\ell \rangle$ at the respective complexities over 300 bootstraps.

Task	Metric	HI	TA	TE
SLT	c^*	$4_{\pm 0}$	$5_{\pm 1}$	$5_{\pm 1}$
	f_{EN}	$90.4_{\pm 5.4}$	$82.9_{\pm 12.1}$	$80.7_{\pm 14.2}$
	f_ℓ	$32.6_{\pm 4.1}$	$54.0_{\pm 8.3}$	$64.0_{\pm 11.9}$
DSA	c^*	$31_{\pm 4}$	$24_{\pm 5}$	$37_{\pm 5}$
	f_{EN}	$43.7_{\pm 22.3}$	$84.8_{\pm 28.2}$	$18.1_{\pm 24.8}$
	f_ℓ	$40.3_{\pm 24.1}$	$79.3_{\pm 22.6}$	$20.5_{\pm 22.1}$
Pr DSA	c^*	$15_{\pm 2}$	$10_{\pm 3}$	$10_{\pm 3}$
	f_{EN}	$26.9_{\pm 20.4}$	$76.1_{\pm 23.0}$	$72.2_{\pm 22.4}$
	f_ℓ	$20.1_{\pm 24.2}$	$81.6_{\pm 27.0}$	$77.7_{\pm 24.7}$
Graph SP	c^*	$206_{\pm 17}$	$217_{\pm 25}$	$219_{\pm 48}$
	f_{EN}	$90.1_{\pm 4.3}$	$86.9_{\pm 8.1}$	$86.0_{\pm 13.4}$
	f_ℓ	$55.8_{\pm 4.6}$	$62.8_{\pm 6.9}$	$61.1_{\pm 10.0}$
T. Rank	c^*	$30_{\pm 2}$	$27_{\pm 1}$	$28_{\pm 2}$
	f_{EN}	$45.4_{\pm 22.7}$	$80.6_{\pm 12.2}$	$68.6_{\pm 22.6}$
	f_ℓ	$49.8_{\pm 23.6}$	$65.0_{\pm 10.2}$	$61.6_{\pm 20.8}$

Table 14: **Sarvam-105B.** Here we present the $\langle c^* \rangle$ at which SMD values were reported along with $\langle f_{\text{EN}} \rangle$ and $\langle f_\ell \rangle$ at the respective complexities over 300 bootstraps.

Task	Metric	AR	HI	JA	TA	TE	ZH
SLT	c^*	$106_{\pm 3}$	$109_{\pm 4}$	$111_{\pm 7}$	$115_{\pm 3}$	$104_{\pm 4}$	$119_{\pm 8}$
	f_{EN}	$89.0_{\pm 5.5}$	$84.3_{\pm 7.8}$	$81.6_{\pm 13.6}$	$74.1_{\pm 10.5}$	$91.9_{\pm 3.8}$	$63.9_{\pm 20.2}$
	f_ℓ	$14.2_{\pm 4.9}$	$46.4_{\pm 7.0}$	$59.7_{\pm 11.1}$	$31.5_{\pm 7.4}$	$25.3_{\pm 5.9}$	$50.1_{\pm 14.7}$
DSA	c^*	$181_{\pm 7}$	$181_{\pm 7}$	$173_{\pm 8}$	$181_{\pm 6}$	$182_{\pm 7}$	$229_{\pm 18}$
	f_{EN}	$89.5_{\pm 9.0}$	$89.5_{\pm 7.8}$	$95.9_{\pm 3.3}$	$89.5_{\pm 9.1}$	$87.2_{\pm 11.7}$	$10.9_{\pm 26.3}$
	f_ℓ	$38.9_{\pm 8.6}$	$33.0_{\pm 7.8}$	$18.2_{\pm 6.2}$	$29.9_{\pm 7.5}$	$39.5_{\pm 9.2}$	$37.9_{\pm 14.5}$
Pr DSA	c^*	$8_{\pm 0}$	$8_{\pm 1}$	$8_{\pm 0}$	$8_{\pm 0}$	$8_{\pm 0}$	$8_{\pm 1}$
	f_{EN}	$98.1_{\pm 3.9}$	$92.0_{\pm 4.3}$	$92.0_{\pm 3.6}$	$98.9_{\pm 4.1}$	$92.0_{\pm 4.5}$	$92.0_{\pm 4.5}$
	f_ℓ	$89.2_{\pm 6.1}$	$96.0_{\pm 3.1}$	$86.0_{\pm 6.4}$	$92.5_{\pm 5.1}$	$96.0_{\pm 3.7}$	$96.0_{\pm 3.1}$
Graph SP	c^*	$527_{\pm 90}$	$363_{\pm 55}$	$348_{\pm 77}$	$342_{\pm 85}$	$416_{\pm 46}$	$359_{\pm 30}$
	f_{EN}	$19.8_{\pm 23.0}$	$69.4_{\pm 13.8}$	$75.2_{\pm 19.8}$	$77.3_{\pm 21.7}$	$48.8_{\pm 14.9}$	$70.9_{\pm 11.7}$
	f_ℓ	$12.6_{\pm 33.5}$	$83.9_{\pm 20.6}$	$83.7_{\pm 27.6}$	$88.2_{\pm 34.6}$	$35.4_{\pm 14.6}$	$45.8_{\pm 7.8}$
T. Rank	c^*	$137_{\pm 4}$	$139_{\pm 11}$	$137_{\pm 8}$	$137_{\pm 6}$	$139_{\pm 10}$	$142_{\pm 10}$
	f_{EN}	$91.5_{\pm 15.8}$	$89.6_{\pm 23.0}$	$92.3_{\pm 14.5}$	$92.3_{\pm 14.1}$	$88.5_{\pm 19.9}$	$83.7_{\pm 20.8}$
	f_ℓ	$24.2_{\pm 6.6}$	$60.3_{\pm 15.3}$	$39.9_{\pm 9.6}$	$30.1_{\pm 8.1}$	$49.6_{\pm 14.5}$	$76.7_{\pm 19.1}$

Table 15: **DeepSeek-V4-Pro.** Here we present the $\langle c^* \rangle$ at which SMD values were reported along with $\langle f_{\text{EN}} \rangle$ and $\langle f_\ell \rangle$ at the respective complexities over 300 bootstraps.

Task	Gemini-3-Flash	Gemini 3.1 Flash-Lite
SLT	0.064 (TE)	0.368 (AR)
DSA	0.683 (HI)	1.195 (TE)
Pr DSA	—	0.761 (HI)
Graph SP	2.903 (TE)	—
T. Rank	2.003 (TE)	— (AR)

Table 16: The variance δ between the back-translated prompt and the original prompt.

Task	EN	AR	HI	JA	TA	TE	ZH
DSA	43,618	48,528	47,779	41,312	44,193	52,149	38,962
SLT	21,722	23,291	26,058	23,333	28,493	29,769	20,918
Graph SP	5,994	6,242	6,188	6,625	6,435	6,884	6,108
Pr DSA	3,563	3,842	4,162	3,489	4,136	4,745	3,418
T. Rank	5,028	5,289	5,530	6,020	6,523	6,788	6,010

Table 17: **Average Input Tokens (Task $\times$ Language) for Gemini-3-Flash.** Gemini's tokenizer is relatively uniform across languages (ratios $0.89\times$–$1.20\times$ vs EN), indicating token inflation is not a meaningful confounder.

task triplet (approximately 1000 points). The number of thought tokens was accessible for Gemini-3-Flash, whereas for other models we estimated the average thought token length using the thought character lengths.[4] We present the results of this analysis in Table 18. These results indicate that thought length is not strongly correlated with language performance.

[4]This is an imperfect measure since character-based measurement does not account for script density differences. For instance, a single CJK character encodes more information than an ASCII character. This introduces an estimated 10–20% measurement error in cross-script comparisons.

Model	SLT	DSA	Pr DSA	Graph SP	T. Rank
Gemini-3-Flash	-0.371	-0.600	-0.086	-0.429	-0.657
Gemma-4-31B-IT	$+0.429$	-0.314	-0.257	$+0.257$	$+0.029$
Gemini 3.1 Flash-Lite	-0.257	$+0.200$	-0.314	-0.143	$+0.143$
GLM-5-1	$\mathbf{-0.886}^*$	-0.657	$\mathbf{-0.886}^*$	$+0.029$	-0.257

Table 18: Spearman rank correlation (ρ) between language ranking by thinking effort and language ranking by Signed Max Divergence (SMD) across tasks and models. Thinking effort is measured by $\mathrm{avg_thinking_tokens}$ for Gemini-3-Flash (script-agnostic) and by $\mathrm{avg_thought_len}$ (character count) for all other models. *Significant at $p < 0.05$ (highlighted in bold and green). A negative ρ would support the hypothesis that languages with lower SMD (better cross-lingual parity) achieve this by eliciting greater thinking effort from the model. Only 2/20 combinations show a significant negative correlation (both GLM-5-1), while 18/20 are statistically indistinguishable from zero, confirming that **cross-lingual gaps cannot be explained merely by differential thinking effort**.

B Gamma Fits

In this section we present graphs for the incomplete gamma function fit described in Section 3 for all the model, task pairs, and for all the languages. These are generated using a Bayesian bootstrap approach: for each bootstrap iteration, we sample the accuracy at each complexity level from the posterior Beta distribution Beta(k+1, n−k+1), where k is the number of correct responses and n is the total, and refit the gamma CDF to obtain a new curve. We repeat this 300 times and shade the region between the 2.5th and 97.5th percentiles of the predicted curves, yielding a 95% credible interval for the fitted curve at each complexity. The plots for models *Gemini-3-Flash, Gemini 3.1 Flash-Lite, Gemma-4-31B, GLM-5.1, Sarvam-105B and DeeepSeek-V4-Pro* are shown in Figures 3, 5, 4, 6, 7 and 8 respectively.

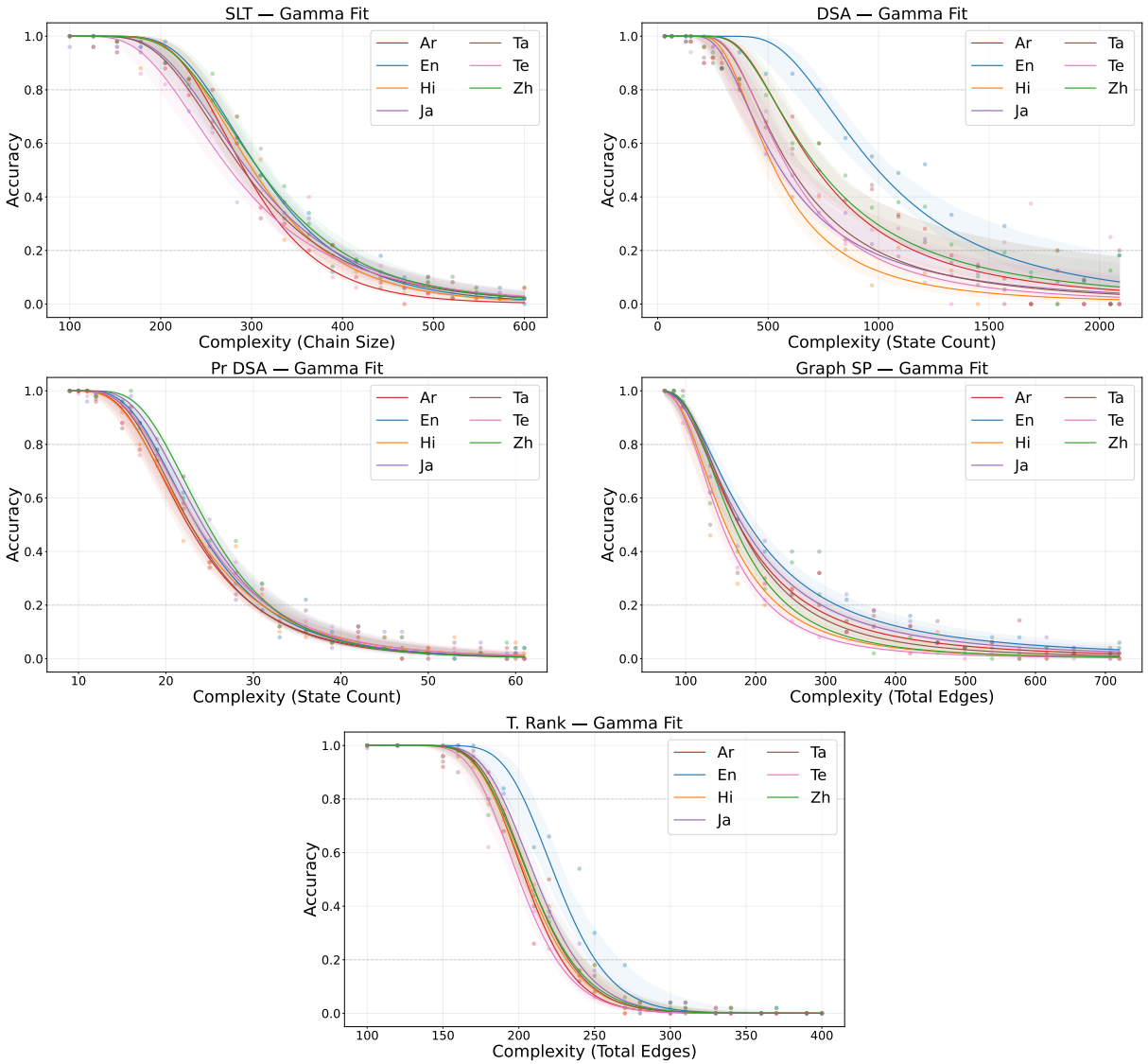


Figure 3: Accuracy vs Complexity plots for different tasks. In each sub-plot, we compare between English and non-English with Gemini-3-Flash model.

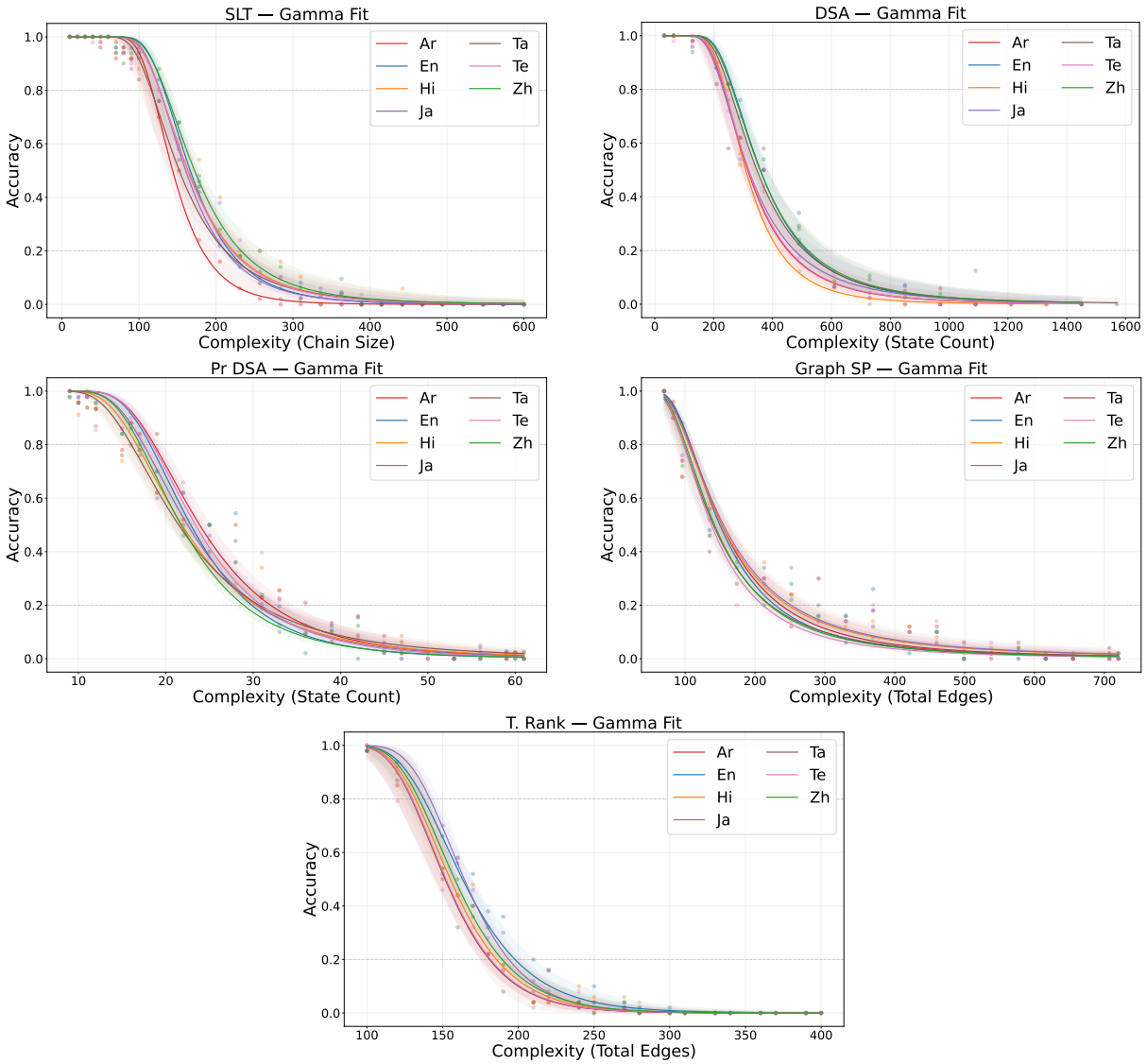


Figure 4: Accuracy vs Complexity plots for different tasks. In each sub-plot, we compare between English and non-English with Gemma-4-31B-IT model.

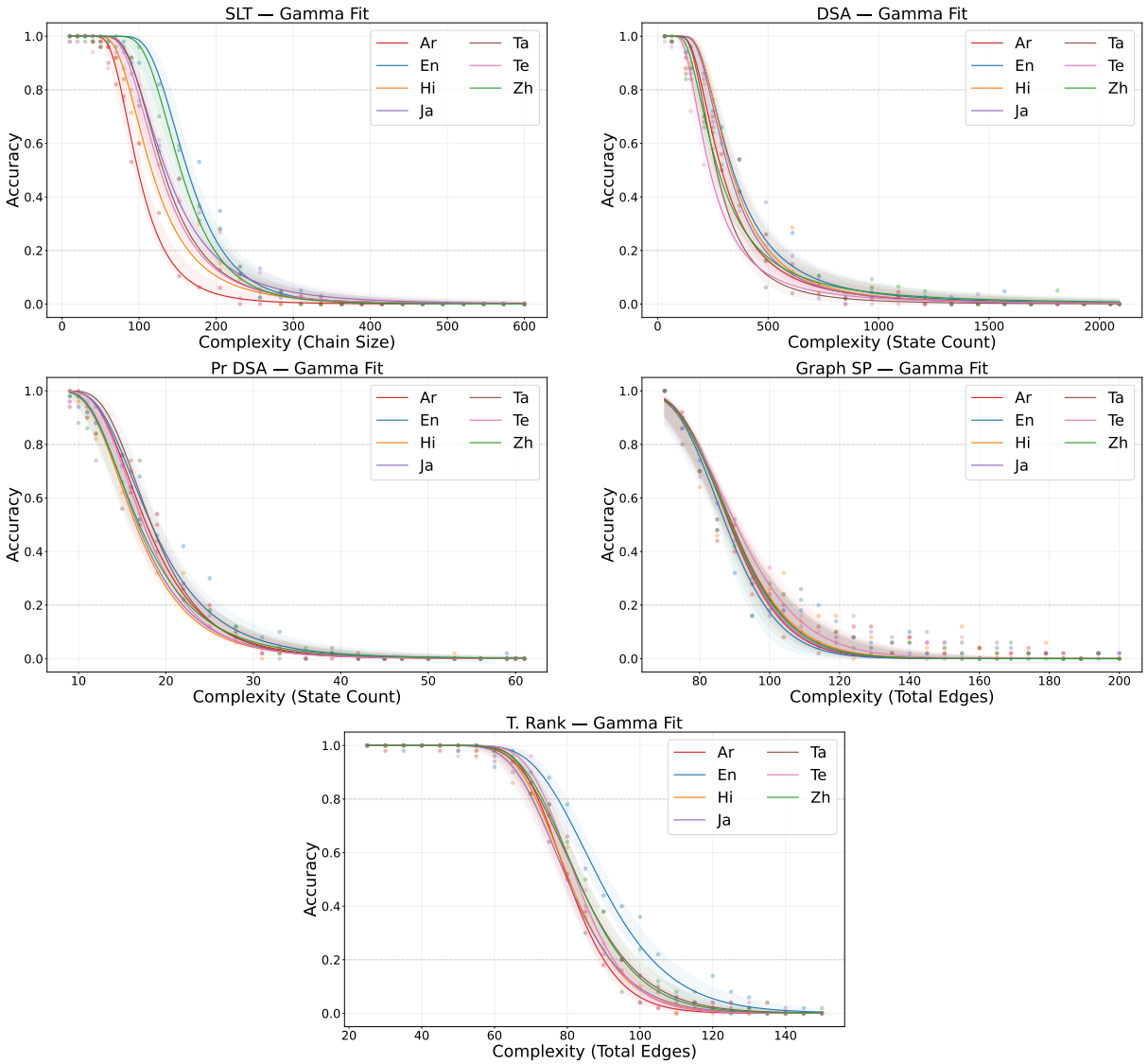


Figure 5: Accuracy vs Complexity plots for different tasks. In each sub-plot, we compare between English and non-English with Gemini 3.1 Flash-Lite model.

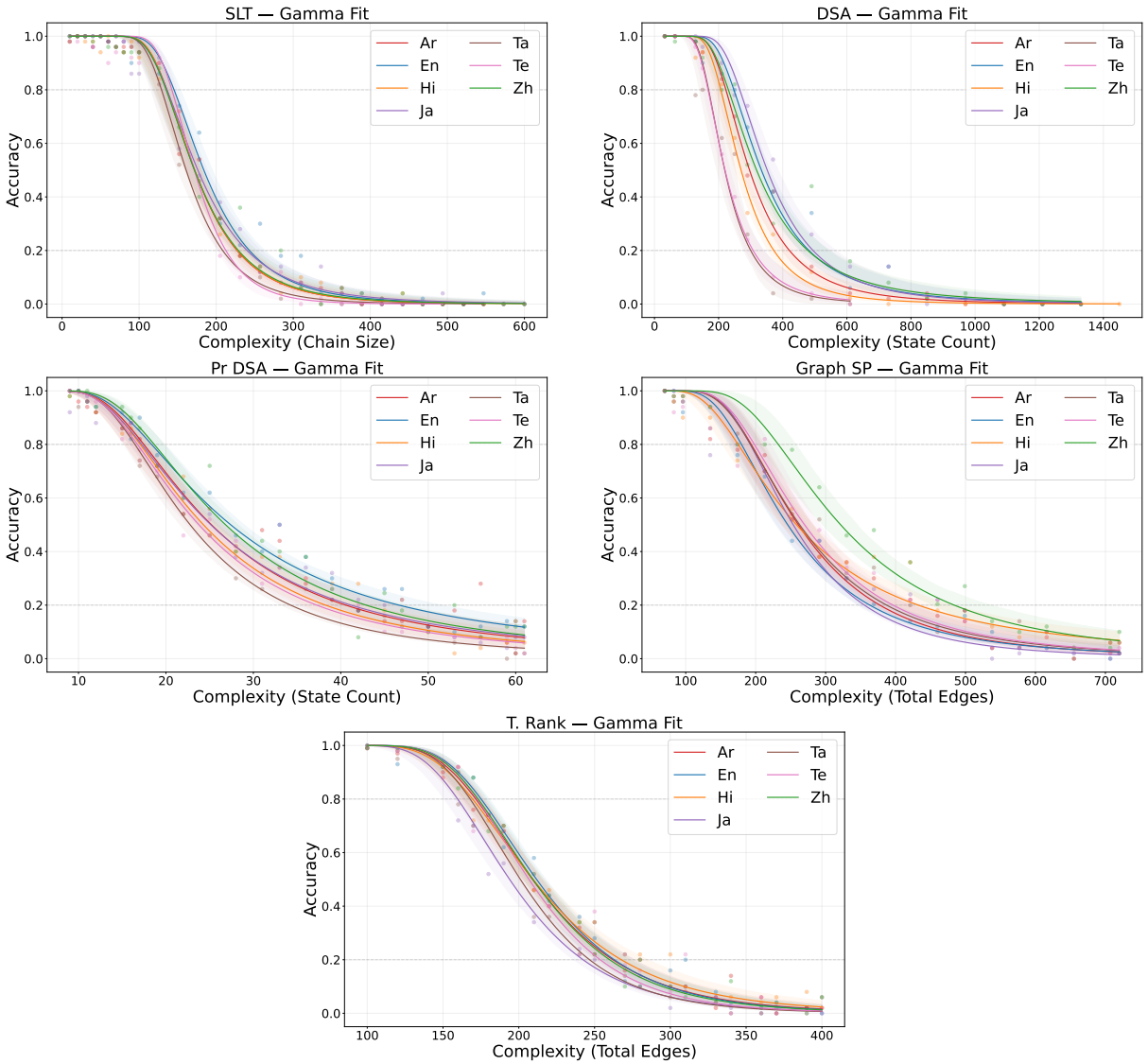


Figure 6: Accuracy vs Complexity plots for different tasks. In each sub-plot, we compare between English and non-English with GLM-5.1 model.

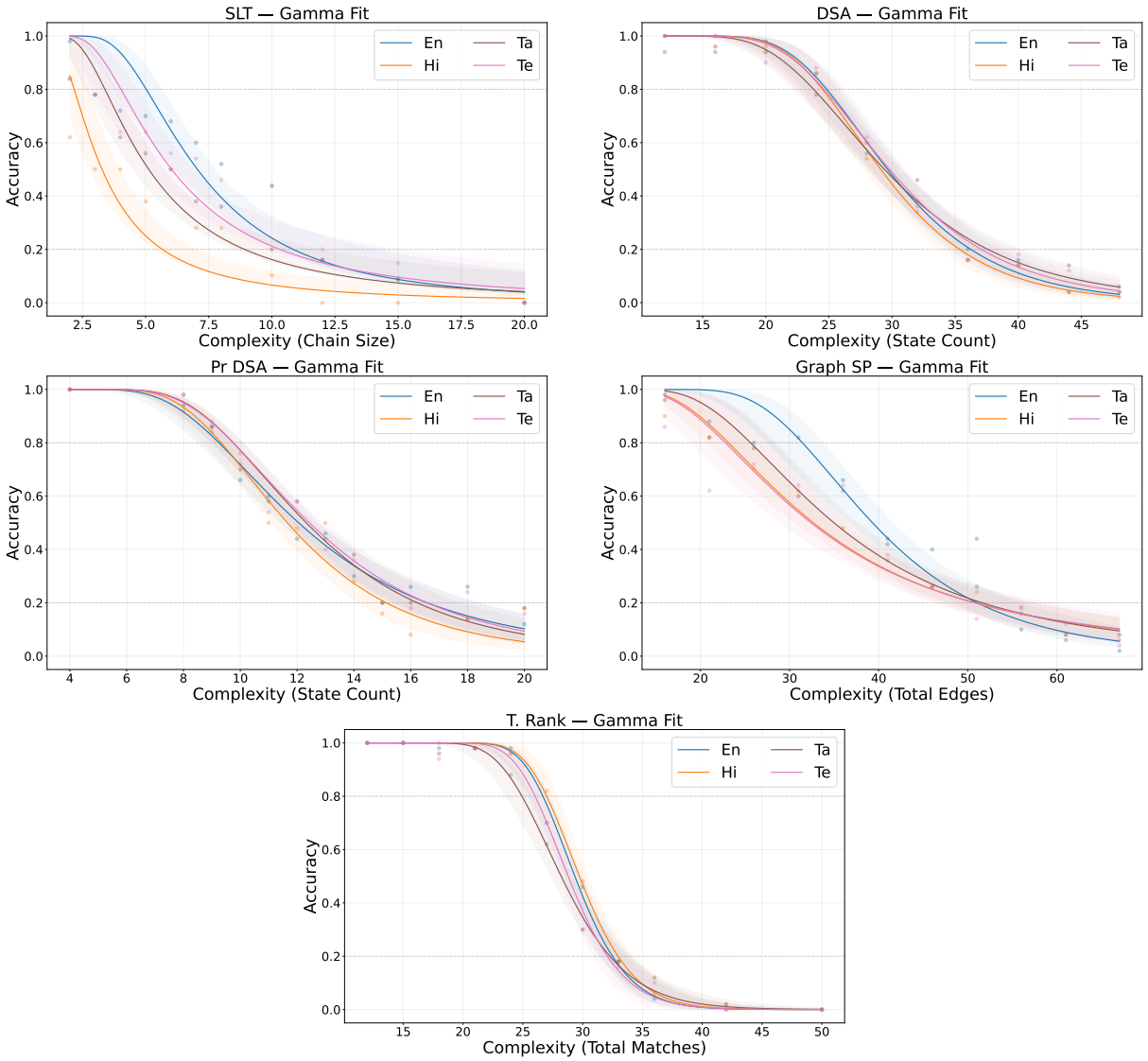


Figure 7: Accuracy vs Complexity plots for different tasks. In each sub-plot, we compare between English and non-English with Sarvam-105B model.

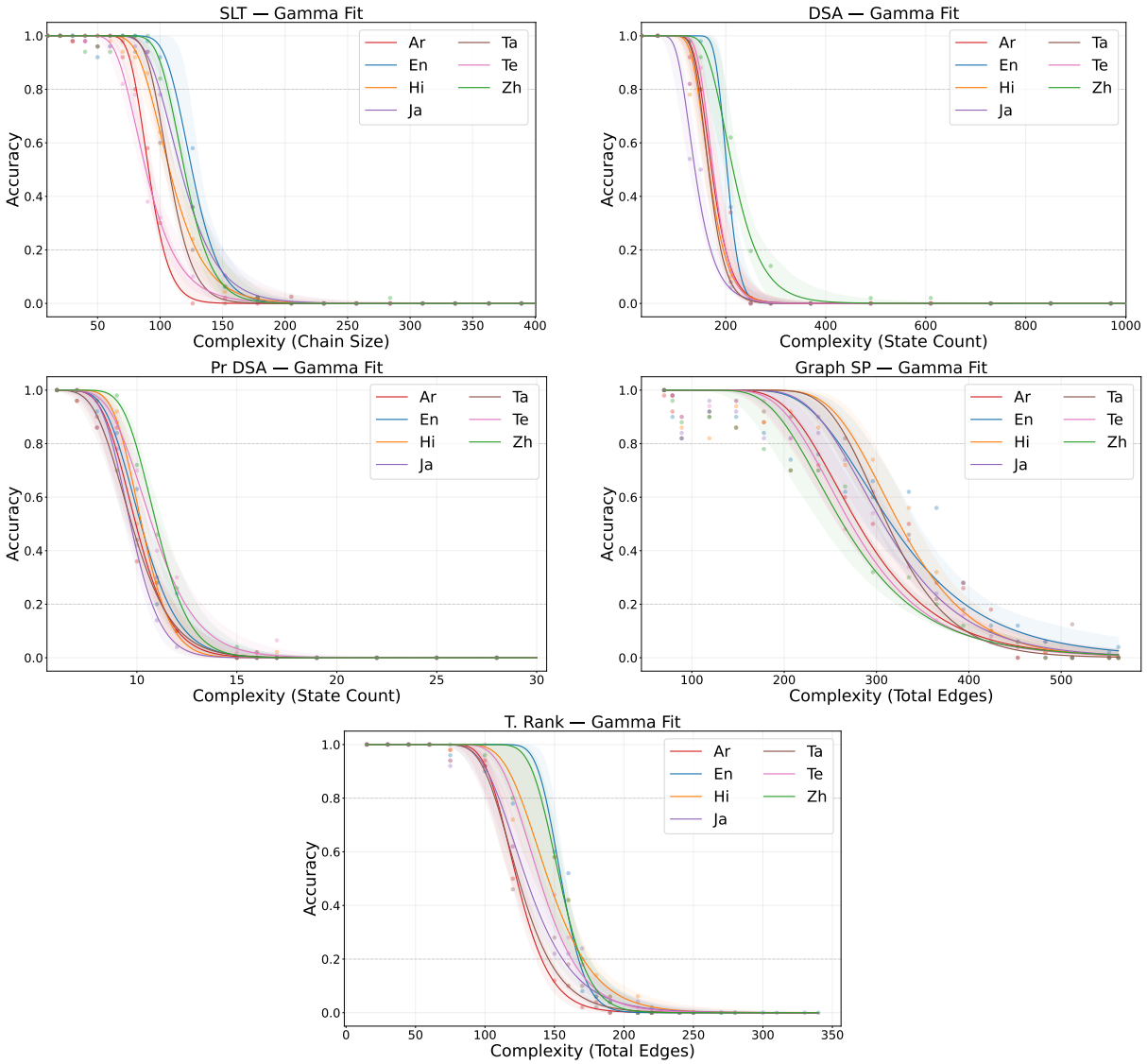


Figure 8: Accuracy vs Complexity plots for different tasks. In each sub-plot, we compare between English and non-English with DeepSeek-V4-Pro model.

C Construction of Prompts and Puzzles

In this section, we explain the construction of prompts and puzzles and provide sample prompts for each puzzle. We start by explaining the construction of the Graph Shortest Path Puzzle. The other puzzles in our suite can be straightforwardly constructed given their description and the prompts given in section C.3. We explain the procedure used to translate the puzzles in section C.2.

C.1 Graph Shortest Path Construction

The Graph Shortest Path puzzle presents a social network as an undirected graph and asks the model to find the shortest path between a source and a target node. The puzzle is constructed so that a *unique* shortest path of length d exists, embedded within a densely cross-wired graph of bypass chains. The generation procedure is parameterized by:

- d: the shortest path length (backbone length),
- n_c: the number of parallel bypass chains,
- c_e: the chain excess (each bypass chain has length $d + c_e$, making it strictly longer than the backbone),
- $\rho \in [0, 1]$: noise density controlling the fraction of safe cross-edges added.

The total number of nodes and edges are given by:

$$N = (d + 1) + n_c \cdot (d + c_e - 1)$$

$$E_{\text{structural}} = d + n_c \cdot (d + c_e)$$

$$E_{\text{estimated}} = E_{\text{structural}} + \rho \cdot |C|$$

where $|C|$ is the number of candidate noise edges.

Algorithm: Graph Shortest Path Puzzle Generation

Input: Parameters (d, n_c, c_e, ρ)

Step 1: Backbone Construction.

1. Compute total nodes $N = (d + 1) + n_c \cdot (d + c_e - 1)$.
2. Generate N unique alphanumeric node IDs (e.g. A1234).
3. Assign the first $d+1$ nodes as the *backbone*: $b_0 \to b_1 \to \cdots \to b_d$.
4. Set *source* $= b_0$, *target* $= b_d$.
5. Add edges (b_i, b_{i+1}) for $i = 0, \ldots, d-1$. This creates the unique shortest path of length d.

Step 2: Parallel Bypass Chains.

1. For each chain $j = 1, \ldots, n_c$:
   - Allocate $d + c_e - 1$ fresh nodes: $c_1^j, c_2^j, \ldots, c_{d+c_e-1}^j$.
   - Add edges: $b_0 \to c_1^j \to c_2^j \to \cdots \to c_{d+c_e-1}^j \to b_d$.
2. Each bypass chain has total length $d + c_e > d$, so they cannot create a shorter path than the backbone.

Step 3: Cross-Wiring Noise Edges.

1. **Enumerate candidate edges.** For each chain node u, consider two types of partners:
   - **Chain–chain pairs:** For each other chain node v:
     - Same chain: allow if depth gap > 1 and $\leq c_e$.
     - Different chains: allow if $\mathit{depth}(v) - \mathit{depth}(u) \leq \mathit{excess}(v)$ and $\mathit{depth}(u) - \mathit{depth}(v) \leq \mathit{excess}(u)$.
   - **Chain–backbone pairs:** For each backbone node b_i: allow only if b_i is at the same depth or behind u (i.e. $i \leq \mathit{depth}(u)$ and $\mathit{depth}(u) - i \leq \mathit{excess}(u)$). This prevents shortcut edges.

   No backbone–backbone pairs are ever considered.
2. **Add edges with distance safety check.** Shuffle candidates, then for each (u, v) sampled with probability ρ:
   - Compute $\mathit{dist}_s(u) + 1 + \mathit{dist}_t(v)$ and $\mathit{dist}_s(v) + 1 + \mathit{dist}_t(u)$.
   - **Reject** if either value $\leq d$ (would create a path $\leq d$ from source to target).
   - Otherwise, add edge (u, v) and propagate updated distances via incremental BFS.

Step 4: Verification.

1. Run BFS from source to target. Verify shortest path length $= d$.
2. Count shortest paths. Verify exactly 1 unique shortest path exists.

When scaling complexity levels we keep the parameters d, n_c, c_e constant, and vary the noise den-

sity ρ to meet the desired number of edges. Once the maximum number of edges are achieved with the set parameters, we increment d by 1 and continue from appropriate p based on required total edges.

C.2 Translation and Annotation procedure

For translating and auditing our puzzles, we employed native speakers through a third-party aggregator to verify and check the templates. Each translation was validated and audited by two separate annotators. No issues were found for 70% of the translations, while minor stylistic changes were made for 30% of the prompts. The annotators were appropriately compensated. Below we present the guidelines provided to the annotators.

Task: This is a Translation Verification task.

Metrics: Check for semantic equivalence with English ground truth and naturalness of the words used in translated versions.

Rater Process:

1. For each locale, we provide a separate tab within the same google sheet labeled with the locale name.
2. Column A: "ENGLISH_GROUNDTRUTH" to check against.
3. Column B: "{lang_id}_TRANSLATION" (e.g., AR_TRANSLATION) to verify.
4. Column C: "Translation OK" (Yes/No).
5. Column D: Edited Translation to add the fixed translations if required.
6. Column E: Comments.

Instructions:

- Compare the English ground truth with translation based on metrics defined above.
- Before you go ahead with translation please look at the rows marked as full example to see the full context of the task which will help with any ambiguities.
- If translation has semantic equivalence with English ground truth and naturalness of the words, Column C: "Translation OK" = "Yes".
- If any translation edit is required:
  - Column C: "Translation OK" = "No".
  - Fill out Column D: Edited Translation.
  - Fill out Column E: Comments with comments on what the issue was and what changes were made.

Audit Process: To be done by a separate set of annotators.

- **Column F**: "Audit Done?" [Data Validation: Yes, No].
- **Column G**: "Issues Found?" [Data Validation: True, False].
- **Column H**: "Comments" [If issue found? = True, provide a justification].

C.3 Prompts

We now provide example prompts from all five puzzle types in our benchmark, shown across all seven evaluation languages. Each prompt is generated from the actual puzzle templates with small configuration parameters for readability. The underlying puzzle structure (states, edges, players, etc.) is identical across all languages—only the surface-level rendering changes.

C.3.1 Sequential Linear Transforms (SLT)

English.

A merchant departed from R4257 with an initial capital of 7. He traveled far and wide, met many people, and carefully recorded every transaction in his ledger. However, a sudden fierce wind scattered the ledger's pages, completely disrupting his itinerary. Your task is to help this merchant piece the ledger back together. Starting from R4257, infer whom he met in order, reconstruct the timeline, and calculate his exact wealth after each encounter.

- After ending business with Z1106, the merchant set off to visit Y3615. Y3615 reviewed the accounts and multiplied the merchant's balance by a factor of -2. As a token of goodwill, Y3615 also gifted the merchant 9.

- After ending business with N4611, the merchant set off to visit O5557. O5557 reviewed the accounts and multiplied the merchant's balance by a factor of -4. Before parting ways, O5557 demanded a fee of 2.
- After ending business with W9928, the merchant set off to visit N4611. N4611 reviewed the accounts and multiplied the merchant's balance by a multiple of -1. Before parting ways, N4611 demanded a fee of 1.
- After ending business with K5552, the merchant set off to visit E4527. E4527 reviewed the accounts and multiplied the merchant's balance by a factor of -1. Before parting ways, E4527 demanded a fee of 2.
- After ending business with O5557, the merchant set off to visit Z1106. Z1106 reviewed the accounts and multiplied the merchant's balance by a factor of 2. Before parting ways, Z1106 demanded a fee of 6.
- After ending business with W7924, the merchant set off to visit K5552. K5552 reviewed the accounts and multiplied the merchant's balance by a factor of 2. Before parting ways, K5552 demanded a fee of 7.
- After ending business with R4257, the merchant set off to visit W9928. W9928 reviewed the accounts and multiplied the merchant's balance by a factor of 1. Before parting ways, W9928 demanded a fee of 6.
- After ending business with Y3615, the merchant set off to visit W7924. W7924 reviewed the accounts and multiplied the merchant's balance by a factor of -5. Before parting ways, W7924 demanded a fee of 8.

IMPORTANT: Output ONLY the answer in this exact format with no other text: {"chain": [v0, v1, v2, ...]} where each value is the merchant's wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

Arabic.

جاب 7. قدره أولي مال برأس R4257 من تاجر انطلق صفقة كل وسجل الأشخاص، من بالعديد والتقى الآفاق، ومفاجئة عاتية رياح بعثرت ذلك، ومع الدفتر. في بعناية مهمتك تماماً. رحلته مسار أربك مما الدفتر، صفحات بدءاً جديد. من الدفتر تجميع في التاجر هذا مساعدة هي بناء وأعد بالترتيب، بهم التقى من استنتج ،R4257 من لقاء. كل بعد الدقيقة ثروته واحسب الزمني، التسلسل

- لزيارة التاجر انطلق ،Z1106 مع أعماله إنهاء بعد وضرب الحسابات Y3615 راجع.Y3615 أهدى مودة، .كعربون2− في التاجر رصيد 9. أيضاً التاجر Y3615
- لزيارة التاجر انطلق ،N4611 مع أعماله إنهاء بعد وضرب الحسابات O5557 راجع.O5557 طالب الافتراق، .قبل4− في التاجر رصيد 2. قدرها برسوم O5557
- لزيارة التاجر انطلق ،W9928 مع أعماله إنهاء بعد وضرب الحسابات N4611 راجع.N4611 طالب الافتراق، .قبل1− في التاجر رصيد 1. قدرها برسوم N4611
- لزيارة التاجر انطلق ،K5552 مع أعماله إنهاء بعد وضرب الحسابات E4527 راجع.E4527 طالب الافتراق، .قبل1− في التاجر رصيد 2. قدرها برسوم E4527
- لزيارة التاجر انطلق ،O5557 مع أعماله إنهاء بعد وضرب الحسابات Z1106 راجع.Z1106 Z1106 طالب الافتراق، 2.قبل في التاجر رصيد 6. قدرها برسوم
- لزيارة التاجر انطلق ،W7924 مع أعماله إنهاء بعد وضرب الحسابات K5552 راجع.K5552 K5552 طالب الافتراق، 2.قبل في التاجر رصيد 7. قدرها برسوم
- لزيارة التاجر انطلق ،R4257 مع أعماله إنهاء بعد وضرب الحسابات W9928 راجع.W9928 W9928 طالب الافتراق، 1.قبل في التاجر رصيد 6. قدرها برسوم
- لزيارة التاجر انطلق ،Y3615 مع أعماله إنهاء بعد وضرب الحسابات W7924 راجع.W7924 طالب الافتراق، .قبل5− في التاجر رصيد 8. قدرها برسوم W7924

IMPORTANT: Output ONLY the answer in this exact format with no other text:

{“chain”: [v0, v1, v2, ...]} where each value is the merchant’s wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

Hindi.

एक व्यापारी ने R4257 से 7 की पूंजी के साथ अपनी व्यापारिक यात्रा शुरू की। उसने दूर-दूर तक यात्रा की, कई लोगों से व्यापार किया और अपनी हर लेन-देन को सावधानीपूर्वक अपने बही-खाते में दर्ज किया। लेकिन अचानक आई एक भयानक आंधी ने उसके बही-खाते के पन्ने बिखेर दिए, जिससे उसकी यात्रा का पूरा क्रम बिगड़ गया। अब आपको इस व्यापारी की मदद करनी है। R4257 से शुरू करते हुए, यह पता लगाएं कि वह किसके बाद किससे मिला, और हर मुलाकात के बाद उसकी सटीक संपत्ति की गणना करें।

- Z1106 के साथ अपना काम खत्म करने के बाद, व्यापारी Y3615 से मिलने निकल पड़ा। Y3615 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को -2 से गुणा किया। सद्भावना के प्रतीक के रूप में, Y3615 ने व्यापारी को 9 का उपहार भी दिया।
- N4611 के साथ अपना काम खत्म करने के बाद, व्यापारी O5557 से मिलने निकल पड़ा। O5557 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को -4 से गुणा किया। वहां से विदा लेने से पहले, O5557 ने 2 का शुल्क मांगा।
- W9928 के साथ अपना काम खत्म करने के बाद, व्यापारी N4611 से मिलने निकल पड़ा। N4611 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को -1 से गुणा किया। वहां से विदा लेने से पहले, N4611 ने 1 का शुल्क मांगा।
- K5552 के साथ अपना काम खत्म करने के बाद, व्यापारी E4527 से मिलने निकल पड़ा। E4527 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को -1 से गुणा किया। वहां से विदा लेने से पहले, E4527 ने 2 का शुल्क मांगा।
- O5557 के साथ अपना काम खत्म करने के बाद, व्यापारी Z1106 से मिलने निकल पड़ा। Z1106 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को 2 से गुणा किया। वहां से विदा लेने से पहले, Z1106 ने 6 का शुल्क मांगा।
- W7924 के साथ अपना काम खत्म करने के बाद, व्यापारी K5552 से मिलने निकल पड़ा। K5552 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को 2 से गुणा किया। वहां से विदा लेने से पहले, K5552 ने 7 का शुल्क मांगा।
- R4257 के साथ अपना काम खत्म करने के बाद, व्यापारी W9928 से मिलने निकल पड़ा। W9928 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को 1 से गुणा किया। वहां से विदा लेने से पहले, W9928 ने 6 का शुल्क मांगा।
- Y3615 के साथ अपना काम खत्म करने के बाद, व्यापारी W7924 से मिलने निकल पड़ा। W7924 ने बही-खाते की जांच की, और व्यापारी की शेष राशि को -5 से गुणा किया। वहां से विदा लेने से पहले, W7924 ने 8 का शुल्क मांगा।

IMPORTANT: Output ONLY the answer in this exact format with no other text: {“chain”: [v0, v1, v2, ...]} where each value is the merchant’s wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

Japanese.

ある商人が初期資本7を持ってR4257を出発しました。彼はあちこちを旅し、多くの人と出会い、すべての取引を帳簿に注意深く記録しました。しかし、突然の強風で帳簿のページが散乱し、彼の旅程は完全に分からなくなってしまいました。あなたの任務は、この商人が帳簿を元通りに繋ぎ合わせるのを手伝うことです。R4257から出発し、彼が順番に誰と出会ったかを推測してタイムラインを再構築し、それぞれの出会いの後の彼の正確な所持金を計算してください。

- Z1106との取引を終えた後、商人はY3615を訪ねるために出発しました。Y3615は帳簿を確認し、商人の残高に -2 を掛けました。親善の印として、Y3615は商人に9を贈りました。
- N4611との取引を終えた後、商人はO5557を訪ねるために出発しました。O5557は帳簿を確認し、商人の残高に -4 を掛けました。別れる前に、O5557は2の料金を要求しました。
- W9928との取引を終えた後、商人はN4611を訪ねるために出発しま

した。N4611は帳簿を確認し、商人の残高に −1 を掛けました。別れる前に、N4611は1の料金を要求しました。
- K5552との取引を終えた後、商人は E4527を訪ねるために出発しました。E4527は帳簿を確認し、商人の残高に −1 を掛けました。別れる前に、E4527は2の料金を要求しました。
- O5557との取引を終えた後、商人は Z1106を訪ねるために出発しました。Z1106は帳簿を確認し、商人の残高に2を掛けました。別れる前に、Z1106は6の料金を要求しました。
- W7924との取引を終えた後、商人は K5552を訪ねるために出発しました。K5552は帳簿を確認し、商人の残高に2を掛けました。別れる前に、K5552は7の料金を要求しました。
- R4257との取引を終えた後、商人はW9928を訪ねるために出発しました。W9928は帳簿を確認し、商人の残高に1を掛けました。別れる前に、W9928は6の料金を要求しました。
- Y3615との取引を終えた後、商人はW7924を訪ねるために出発しました。W7924は帳簿を確認し、商人の残高に −5 を掛けました。別れる前に、W7924は8の料金を要求しました。

IMPORTANT: Output ONLY the answer in this exact format with no other text: {“chain”: [v0, v1, v2, ...]} where each value is the merchant’s wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

Tamil.

ஒரு வியாபாரி R4257 -லிருந்து 7 என்ற ஆரம்ப மூலதனத்துடன் புறப்பட்டார். அவர் பல இடங்களுக்குப் பயணம் செய்து, பலரைச் சந்தித்து, ஒவ்வொரு பரிவர்த்தனையையும் தனது கணக்குப் புத்தகத்தில் கவனமாகப் பதிவு செய்தார். இருப்பினும், திடீரென வீசிய பலத்த காற்று கணக்குப் புத்தகத்தின் பக்கங்களைச் சிதறடித்து, அவரது பயணத் திட்டத்தை முற்றிலுமாக சீர்குலைத்தது. இந்த வியாபாரிக்கு தனது கணக்குப் புத்தகத்தை மீண்டும் ஒன்றிணைக்க உதவுவதே உங்கள் பணியாகும். R4257 -லிருந்து தொடங்கி, அவர் யாரை, எந்த வரிசையில் சந்தித்தார் என்பதை ஊகித்து, காலவரிசையை மறுகட்டமைப்பு செய்து, ஒவ்வொரு சந்திப்பிற்கும் பிறகு அவரது சரியான செல்வத்தைக் கணக்கிடுங்கள்.

- Z1106 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி Y3615 ஐச் சந்திக்கப் புறப்பட்டார். Y3615 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை −2 ஆல் பெருக்கினார். நல்லெண்ணத்தின் அடையாளமாக, Y3615 வியாபாரிக்கு 9 -ஐப் பரிசளித்தார்.
- N4611 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி O5557 ஐச் சந்திக்கப் புறப்பட்டார். O5557 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை −4 ஆல் பெருக்கினார். பிரியும் முன், O5557 2 -ஐக் கட்டணமாகக் கோரினார்.
- W9928 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி N4611 ஐச் சந்திக்கப் புறப்பட்டார். N4611 கணக்குகளைச்

சரிபார்த்து, வியாபாரியின் இருப்பை −1 ஆல் பெருக்கினார். பிரியும் முன், N4611 1 -ஐக் கட்டணமாகக் கோரினார்.

- K5552 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி E4527 ஐச் சந்திக்கப் புறப்பட்டார். E4527 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை −1 ஆல் பெருக்கினார். பிரியும் முன், E4527 2 -ஐக் கட்டணமாகக் கோரினார்.
- O5557 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி Z1106 ஐச் சந்திக்கப் புறப்பட்டார். Z1106 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை 2 ஆல் பெருக்கினார். பிரியும் முன், Z1106 6 -ஐக் கட்டணமாகக் கோரினார்.
- W7924 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி K5552 ஐச் சந்திக்கப் புறப்பட்டார். K5552 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை 2 ஆல் பெருக்கினார். பிரியும் முன், K5552 7 -ஐக் கட்டணமாகக் கோரினார்.
- R4257 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி W9928 ஐச் சந்திக்கப் புறப்பட்டார். W9928 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை 1 ஆல் பெருக்கினார். பிரியும் முன், W9928 6 -ஐக் கட்டணமாகக் கோரினார்.
- Y3615 உடனான வியாபாரத்தை முடித்த பிறகு, வியாபாரி W7924 ஐச் சந்திக்கப் புறப்பட்டார். W7924 கணக்குகளைச் சரிபார்த்து, வியாபாரியின் இருப்பை −5 ஆல் பெருக்கினார். பிரியும் முன், W7924 8 -ஐக் கட்டணமாகக் கோரினார்.

IMPORTANT: Output ONLY the answer in this exact format with no other text: {“chain”: [v0, v1, v2, ...]} where each value is the merchant’s wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

Telugu.

ఒక వ్యాపారి 7 ప్రారంభ పెట్టుబడితో R4257 నుండి బయలుదేరాడు. అతను దేశదేశాలు తిరిగి, ఎంతో మందిని కలిసి, ప్రతి లావాదేవీని తన ఖాతా పుస్తకంలో జాగ్రత్తగా నమోదు చేసుకున్నాడు. అయితే, అకస్మాత్తుగా వీచిన బలమైన గాలి ఖాతా పుస్తకం పేజీలను చెల్లాచెదురు చేసింది, దీని వల్ల అతని ప్రయాణ వివరాలు పూర్తిగా తారుమారయ్యాయి. ఈ వ్యాపారి తన ఖాతా పుస్తకాన్ని మళ్లీ ఒక క్రమంలో పెట్టడానికి సహాయం చేయడమే మీ పని. R4257 నుండి ప్రారంభించి, అతను వరుసగా ఎవరిని కలిశాడో ఊహించండి, కాలక్రమాన్ని పునర్నిర్మించండి మరియు ప్రతి ఒక్కరినీ కలిసిన తర్వాత అతని వద్ద ఉన్న ఖచ్చితమైన సంపదను లెక్కించండి.

- Z1106 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి Y3615 ని కలవడానికి బయలుదేరాడు.Y3615 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను −2 తో గుణించారు. సద్భావనకు చిహ్నంగా, Y3615 వ్యాపారికి 9 కూడా బహుమతిగా ఇచ్చారు.
- N4611 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి O5557 ని కలవడానికి బయలుదేరాడు.O5557 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను −4 తో గుణించారు. విడిపోయే ముందు, O5557 2 రుసుముగా అడిగారు.
- W9928 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి N4611 ని కలవడానికి బయలుదేరాడు.N4611 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను −1 తో గుణించారు. విడిపోయే ముందు, N4611 1 రుసుముగా అడిగారు.
- K5552 తో వ్యాపారం ముగిసిన తర్వాత,

వ్యాపారి E4527 ని కలవడానికి బయలుదేరాడు.E4527 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను -1 తో గుణించారు. విడిపోయే ముందు, E4527 2 రుసుముగా అడిగారు.

- O5557 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి Z1106 ని కలవడానికి బయలుదేరాడు.Z1106 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను 2 తో గుణించారు. విడిపోయే ముందు, Z1106 6 రుసుముగా అడిగారు.
- W7924 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి K5552 ని కలవడానికి బయలుదేరాడు.K5552 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను 2 తో గుణించారు. విడిపోయే ముందు, K5552 7 రుసుముగా అడిగారు.
- R4257 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి W9928 ని కలవడానికి బయలుదేరాడు.W9928 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను 1 తో గుణించారు. విడిపోయే ముందు, W9928 6 రుసుముగా అడిగారు.
- Y3615 తో వ్యాపారం ముగిసిన తర్వాత, వ్యాపారి W7924 ని కలవడానికి బయలుదేరాడు.W7924 ఖాతాలను పరిశీలించి, వ్యాపారి బ్యాలెన్స్‌ను -5 తో గుణించారు. విడిపోయే ముందు, W7924 8 రుసుముగా అడిగారు.

IMPORTANT: Output ONLY the answer in this exact format with no other text: {“chain”: [v0, v1, v2, ...]} where each value is the merchant’s wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

Chinese.

一位商人带着7的初始本金从 R4257启程。他游历四方，结识了许多人，并将每一笔交易都仔细记录在账本中。然而，一阵突如其来的狂风吹散了账本的书页，将他的行程彻底打乱。你的任务是帮助这位商人重新拼凑账本。从 R4257出发，推断出他先后遇见了谁，重建时间线，并计算出他每次遭遇后的确切财富。

- 在与 Z1106结束生意后，商人启程去拜访 Y3615。Y3615审核了账目，将商人的余额乘以了 -2。作为善意的象征，Y3615还赠予了商人9。
- 在与 N4611结束生意后，商人启程去拜访 O5557。O5557审核了账目，将商人的余额乘以了 -4。在分道扬镳之前，O5557索要了2的费用。
- 在与 W9928结束生意后，商人启程去拜访 N4611。N4611审核了账目，将商人的余额乘以了 -1。在分道扬镳之前，N4611索要了1的费用。
- 在与 K5552结束生意后，商人启程去拜访 E4527。E4527审核了账目，将商人的余额乘以了 -1。在分道扬镳之前，E4527索要了2的费用。
- 在与 O5557结束生意后，商人启程去拜访 Z1106。Z1106审核了账目，将商人的余额乘以了2。在分道扬镳之前，Z1106索要了6的费用。
- 在与 W7924结束生意后，商人启程去拜访 K5552。K5552审核了账目，将商人的余额乘以了2。在分道扬镳之前，K5552索要了7的费用。
- 在与 R4257结束生意后，商人启程去拜访 W9928。W9928审核了账目，将商人的余额乘以了1。在分道扬镳之前，W9928索要了6的费用。
- 在与 Y3615结束生意后，商人启程去拜访 W7924。W7924审核了账目，将商人的余额乘以了 -5。在分道扬镳之前，W7924索要了8的费用。

IMPORTANT: Output ONLY the answer in this exact format with no other text: {“chain”: [v0, v1, v2, ...]} where each value is the merchant’s wealth after each transaction, in chronological order. Do not show any intermediate steps or explanations. Your response must be a single valid JSON object and nothing else.

C.3.2 Discrete State Automata (DSA)

English.

You are navigating a building with different rooms. Each room has keys that take you to other rooms.

You start in Room E2679.

- In Room V9935, if you use the Gold

Key, you reach Room H4657.
- In Room V9935, if you use the Silver Key, you reach Room A5506.
- In Room U2824, if you use the Silver Key, you reach Room V9935.
- In Room A5506, if you use the Gold Key, you reach Room V9935.
- In Room A5506, if you use the Silver Key, you reach Room V9935.
- In Room E2679, if you use the Silver Key, you reach Room A5506.
- In Room H4657, if you use the Gold Key, you reach Room A5506.
- In Room E2679, if you use the Gold Key, you reach Room A5506.
- In Room U2824, if you use the Gold Key, you reach Room E2679.
- In Room H4657, if you use the Silver Key, you reach Room V9935.

If you use the following sequence of keys:

1. Gold Key
2. Silver Key
3. Gold Key
4. Gold Key
5. Silver Key, which room do you end up in?

Example response: {"state": "Room D6881"}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is:

1. Gold Key
2. Silver Key
3. Gold Key

you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Arabic.

بها غرفة كل مختلفة. غرف به مبنى في تتنقل أنت أخرى. غرف إلى تأخذك مفاتيح

غرفة في تبدأ E2679.

- الذهبي، المفتاح استخدمت إذا، V9935 غرفة في H4657. غرفة في الأمر بك ينتهي فسوف
- الفضي، المفتاح استخدمت إذا، V9935 غرفة في A5506. غرفة في الأمر بك ينتهي فسوف
- المفتاح استخدمت إذا، U2824 غرفة في غرفة في الأمر بك ينتهي فسوف الفضي، V9935.
- الذهبي، المفتاح استخدمت إذا، A5506 غرفة في V9935. غرفة في الأمر بك ينتهي فسوف
- الفضي، المفتاح استخدمت إذا، A5506 غرفة في V9935. غرفة في الأمر بك ينتهي فسوف
- الفضي، المفتاح استخدمت إذا، E2679 غرفة في A5506. غرفة في الأمر بك ينتهي فسوف
- المفتاح استخدمت إذا، H4657 غرفة في غرفة في الأمر بك ينتهي فسوف الذهبي، A5506.
- الذهبي، المفتاح استخدمت إذا، E2679 غرفة في A5506. غرفة في الأمر بك ينتهي فسوف
- المفتاح استخدمت إذا، U2824 غرفة في غرفة في الأمر بك ينتهي فسوف الذهبي، E2679.
- المفتاح استخدمت إذا، H4657 غرفة في غرفة في الأمر بك ينتهي فسوف الفضي، V9935.

التالي: المفاتيح تسلسل استخدمت إذا

1. الذهبي المفتاح.
2. الفضي المفتاح.
3. الذهبي المفتاح.
4. الذهبي المفتاح.
5. بها؟ ستنتهي التي الغرفة هي فما الفضي، المفتاح.

الاستجابة: على مثال {``state": غرفة `` D6881``}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is:

1. الذهبي المفتاح
2. الفضي المفتاح
3. الذهبي المفتاح

you apply trigger 1 (الذهبي المفتاح), then trigger 2 (الفضي المفتاح), then trigger 3 (الذهبي المفتاح) and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Hindi.

आप एक इमारत में अलग-अलग कमरों से गुज़र रहे हैं। हर कमरे में चाबियाँ हैं जो आपको दूसरे कमरों में ले जाती हैं।

आप कमरा E2679 में शुरू करते हैं।

- कमरा V9935 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप कमरा H4657 में पहुँचते हैं।
- कमरा V9935 में, यदि आप चांदी की चाबी का उपयोग करते हैं, तो आप कमरा A5506 में पहुँचते हैं।
- कमरा U2824 में, यदि आप चांदी की चाबी का उपयोग करते हैं, तो आप कमरा V9935 में पहुँचते हैं।
- कमरा A5506 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप कमरा V9935 में पहुँचते हैं।
- कमरा A5506 में, यदि आप चांदी की चाबी का उपयोग करते हैं, तो आप कमरा V9935 में पहुँचते हैं।
- कमरा E2679 में, यदि आप चांदी की चाबी का उपयोग करते हैं, तो आप कमरा A5506 में पहुँचते हैं।
- कमरा H4657 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप कमरा A5506 में पहुँचते हैं।
- कमरा E2679 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप कमरा A5506 में पहुँचते हैं।
- कमरा U2824 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप कमरा E2679 में पहुँचते हैं।
- कमरा H4657 में, यदि आप चांदी की चाबी का उपयोग करते हैं, तो आप कमरा V9935 में पहुँचते हैं।

यदि आप चाबियों के निम्नलिखित क्रम का उपयोग करते हैं:

1. सोने की चाबी
2. चांदी की चाबी
3. सोने की चाबी
4. सोने की चाबी
5. चांदी की चाबी, तो आप अंत में किस कमरे में पहुँचेंगे?

उदाहरण प्रतिक्रिया: {"state": "कमरा D6881"}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is:

1. सोने की चाबी
2. चांदी की चाबी
3. सोने की चाबी

you apply trigger 1 (सोने की चाबी), then trigger 2 (चांदी की चाबी), then trigger 3 (सोने की चाबी), and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Japanese.

あなたは異なる部屋がある建物を移動しています。各部屋には他の部屋に移動できる鍵があります。

あなたは部屋 E2679 から始めます。

- 部屋 V9935 で金の鍵を使うと、部屋 H4657 にたどり着きます。
- 部屋 V9935 で銀の鍵を使うと、部屋 A5506 にたどり着きます。
- 部屋 U2824 で銀の鍵を使うと、部屋 V9935 にたどり着きます。
- 部屋 A5506 で金の鍵を使うと、部屋 V9935 にたどり着きます。
- 部屋 A5506 で銀の鍵を使うと、部屋 V9935 にたどり着きます。
- 部屋 E2679 で銀の鍵を使うと、部屋 A5506 にたどり着きます。
- 部屋 H4657 で金の鍵を使うと、部屋 A5506 にたどり着きます。
- 部屋 E2679 で金の鍵を使うと、部屋 A5506 にたどり着きます。

- 部屋 U2824 で金の鍵を使うと、部屋 E2679 にたどり着きます。
- 部屋 H4657 で銀の鍵を使うと、部屋 V9935 にたどり着きます。

次の鍵のシーケンスを使用した場合:

1. 金の鍵
2. 銀の鍵
3. 金の鍵
4. 金の鍵
5. 銀の鍵、どの部屋にたどり着きますか？

回答の例: {“state”: “部屋 D6881”}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is:

1. 金の鍵
2. 銀の鍵
3. 金の鍵

you apply trigger 1 (金の鍵), then trigger 2 (銀の鍵), then trigger 3 (金の鍵), and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Tamil.

நீங்கள் வெவ்வேறு அறைகள் கொண்ட ஒரு கட்டிடத்திற்குள் செல்கிறீர்கள். ஒவ்வொரு அறையிலும் மற்ற அறைகளுக்கு அழைத்துச் செல்லும் சாவிகள் உள்ளன.

நீங்கள் அறை E2679 இல் தொடங்குகிறீர்கள்.

- அறை V9935 இல், நீங்கள் தங்கச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை H4657-ஐ அடைவீர்கள்.
- அறை V9935 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை A5506-ஐ அடைவீர்கள்.
- அறை U2824 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை V9935-ஐ அடைவீர்கள்.
- அறை A5506 இல், நீங்கள் தங்கச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை V9935-ஐ அடைவீர்கள்.
- அறை A5506 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை V9935-ஐ அடைவீர்கள்.
- அறை E2679 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை A5506-ஐ அடைவீர்கள்.
- அறை H4657 இல், நீங்கள் தங்கச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை A5506-ஐ அடைவீர்கள்.
- அறை E2679 இல், நீங்கள் தங்கச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை A5506-ஐ அடைவீர்கள்.
- அறை U2824 இல், நீங்கள் தங்கச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை E2679-ஐ அடைவீர்கள்.
- அறை H4657 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் அறை V9935-ஐ அடைவீர்கள்.

பின்வரும் சாவிகளின் வரிசையை நீங்கள் பயன்படுத்தினால்:

1. தங்கச் சாவி
2. வெள்ளிச் சாவி
3. தங்கச் சாவி

4. தங்கச் சாவி
5. வெள்ளிச் சாவி, இறுதியில் நீங்கள் எந்த அறையை அடைவீர்கள்?

எடுத்துக்காட்டு பதில்: {“state”: “அறை D6881”}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is:

1. தங்கச் சாவி
2. வெள்ளிச் சாவி
3. தங்கச் சாவி

you apply trigger 1 (தங்கச் சாவி), then trigger 2 (வெள்ளிச் சாவி), then trigger 3 (தங்கச் சாவி), and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Telugu.

మీరు వివిధ గదులు ఉన్న భవనంలో నడుస్తున్నారు. ప్రతి గదిలో మిమ్మల్ని ఇతర గదులకు తీసుకెళ్ళే తాళంచెవులు ఉన్నాయి.

మీరు గది E2679 లో ప్రారంభించారు

- గది V9935లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు గది H4657 కి చేరుకుంటారు.
- గది V9935లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు గది A5506 కి చేరుకుంటారు.
- గది U2824లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు గది V9935 కి చేరుకుంటారు.
- గది A5506లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు గది V9935 కి చేరుకుంటారు.
- గది A5506లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు గది V9935 కి చేరుకుంటారు.
- గది E2679లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు గది A5506 కి చేరుకుంటారు.
- గది H4657లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు గది A5506 కి చేరుకుంటారు.
- గది E2679లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు గది A5506 కి చేరుకుంటారు.
- గది U2824లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు గది E2679 కి చేరుకుంటారు.
- గది H4657లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు గది V9935 కి చేరుకుంటారు.

మీరు క్రింది తాళంచెవుల క్రమాన్ని ఉపయోగిస్తే:

1. బంగారు తాళంచెవి
2. వెండి తాళంచెవి
3. బంగారు తాళంచెవి
4. బంగారు తాళంచెవి
5. వెండి తాళంచెవి, చివరకు మీరు ఏ గదికి చేరుకుంటారు?

ఉదాహరణ ప్రతిస్పందన: {“state”: “గది D6881”}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is: 1. బంగారు తాళంచెవి 2. వెండి తాళంచెవి 3. బంగారు తాళంచెవి you apply trigger 1 (బంగారు తాళంచెవి), then trigger 2 (వెండి తాళంచెవి), then trigger 3 (బంగారు తాళంచెవి), and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Chinese.

你正在一栋建筑物中穿行，里面有不同的房间，每个房间里都有通往其他房间的钥匙。

你从房间 E2679 开始。

- 在房间 V9935，如果使用金钥匙，你会到达房间 H4657。
- 在房间 V9935，如果使用银钥匙，你

会到达房间 A5506。
- 在房间 U2824, 如果使用银钥匙, 你会到达房间 V9935。
- 在房间 A5506, 如果使用金钥匙, 你会到达房间 V9935。
- 在房间 A5506, 如果使用银钥匙, 你会到达房间 V9935。
- 在房间 E2679, 如果使用银钥匙, 你会到达房间 A5506。
- 在房间 H4657, 如果使用金钥匙, 你会到达房间 A5506。
- 在房间 E2679, 如果使用金钥匙, 你会到达房间 A5506。
- 在房间 U2824, 如果使用金钥匙, 你会到达房间 E2679。
- 在房间 H4657, 如果使用银钥匙, 你会到达房间 V9935。

如果你按以下顺序使用钥匙：1. 金钥匙

1. 银钥匙
2. 金钥匙
3. 金钥匙
4. 银钥匙，你最终会在哪个房间?

示例回答：{“state”：“房间 D6881”}

IMPORTANT: The sequence has exactly 5 numbered triggers. Process them step by step: for each numbered trigger, find the matching transition rule for your current room and that trigger, then move to the next room. After applying all 5 triggers, report the final room. For example, if the sequence is:

1. 金钥匙
2. 银钥匙
3. 金钥匙

you apply trigger 1 (金钥匙), then trigger 2 (银钥匙), then trigger 3 (金钥匙), and report the room after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

C.3.3 Probabilistic Discrete State Automata (Pr DSA)

English.

You navigate through a building with different rooms. Each room has keys, which can lead you into one of two possible rooms, each with a specific probability.

You begin in Room U5506.

- In Room D5012, if you use the Silver Key, you reach Room D5506 with probability 0.3 or Room A5012 with probability 0.7.
- In Room U5506, if you use the Silver Key, you reach Room D5012 with probability 0.1 or Room U5506 with probability 0.9.
- In Room A5012, if you use the Gold Key, you reach Room D5012 with probability 0.2 or Room D5506 with probability 0.8.
- In Room D5506, if you use the Gold Key, you reach Room U5506 with probability 0.2 or Room U5012 with probability 0.8.
- In Room D5012, if you use the Gold Key, you reach Room A5012 with probability 0.1 or Room D5012 with probability 0.9.
- In Room U5012, if you use the Gold Key, you reach Room A5012 with probability 0.1 or Room U5012 with probability 0.9.
- In Room D5506, if you use the Silver Key, you reach Room U5506 with probability 0.2 or Room A5012 with probability 0.8.
- In Room U5012, if you use the Silver Key, you reach Room U5012 with probability 0.2 or Room D5506 with probability 0.8.
- In Room U5506, if you use the Gold Key, you reach Room A5012 with probability 0.3 or Room U5012 with probability 0.7.
- In Room A5012, if you use the Silver Key, you reach Room D5012 with probability 0.1 or Room U5012 with probability 0.9.

If you use the following sequence of keys: Gold Key, Gold Key, Gold Key, Gold Key, Silver Key, Gold Key, Gold Key, Gold Key, Silver Key, Silver Key, Silver Key, Silver Key, Gold Key, Silver Key, Silver Key, which are the 3 most probable rooms you could reach? List these in the order from the highest to the lowest probability. Ex-

ample answer: {“states”: [“Room G5374”, “Room W2169”, “Room T3803”, “Room R5010”, “Room F8573”, “Room M5422”, “Room U4598”, “Room V6313”]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 8 most probable rooms at that point. For example, if the sequence

is “Gold Key, Silver Key, Gold Key“ (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response

must be a single valid JSON object and nothing else.

Arabic.

بها غرفة كل مختلفة. غرف به مبنى في تتنقل أنت كل محتملتين، غرفتين من واحدة إلى تأخذك قد مفاتيح محددة. باحتمالية منهما

U5506. غرفة في تبدأ

- المفتاح استخدمت إذا ،D5012 غرفة في باحتمالية D5506 غرفة إلى فستنتقل الفضي، 0.7. باحتمالية A5012 غرفة إلى أو 0.3
- المفتاح استخدمت إذا ،U5506 غرفة في باحتمالية D5012 غرفة إلى فستنتقل الفضي، 0.9. باحتمالية U5506 غرفة إلى أو 0.1
- الذهبي، المفتاح استخدمت إذا ،A5012 غرفة في إلى أو 0.2 باحتمالية D5012 غرفة إلى فستنتقل 0.8. باحتمالية D5506 غرفة
- المفتاح استخدمت إذا ،D5506 غرفة في باحتمالية U5506 غرفة إلى فستنتقل الذهبي، 0.8. باحتمالية U5012 غرفة إلى أو 0.2
- المفتاح استخدمت إذا ،D5012 غرفة في باحتمالية A5012 غرفة إلى فستنتقل الذهبي، 0.9. باحتمالية D5012 غرفة إلى أو 0.1
- المفتاح استخدمت إذا ،U5012 غرفة في باحتمالية A5012 غرفة إلى فستنتقل الذهبي، 0.9. باحتمالية U5012 غرفة إلى أو 0.1
- المفتاح استخدمت إذا ،D5506 غرفة في باحتمالية U5506 غرفة إلى فستنتقل الفضي، 0.8. باحتمالية A5012 غرفة إلى أو 0.2
- المفتاح استخدمت إذا ،U5012 غرفة في باحتمالية U5012 غرفة إلى فستنتقل الفضي، 0.8. باحتمالية D5506 غرفة إلى أو 0.2
- المفتاح استخدمت إذا ،U5506 غرفة في باحتمالية A5012 غرفة إلى فستنتقل الذهبي، 0.7. باحتمالية U5012 غرفة إلى أو 0.3
- الفضي، المفتاح استخدمت إذا ،A5012 غرفة في إلى أو 0.1 باحتمالية D5012 غرفة إلى فستنتقل 0.9. باحتمالية U5012 غرفة

الذهبي, المفتاح التالي: المفاتيح تسلسل استخدمت إذا المفتاح الذهبي, المفتاح الذهبي, المفتاح الذهبي, المفتاح الذهبي, المفتاح الذهبي, المفتاح الذهبي, المفتاح الفضي, المفتاح الفضي, المفتاح الفضي, المفتاح الفضي, المفتاح الفضي, المفتاح الفضي, المفتاح الذهبي, المفتاح الفضي, الأمر بك ينتهي أن يمكن محتملة غرف 8 أفضل هي فما احتمالية. الأقل إلى الأكثر من بترتيب اذكرها فيها؟

الاستجابة: على مثال {``states": [``غرفة G5374``, ``غرفة W2169``, ``غرفة T3803``, ``غرفة R5010``, ``غرفة F8573``, ``غرفة M5422``, ``غرفة U4598``, ``غرفة V6313``]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 3 most probable rooms at that point. For example, if the sequence

is “Gold Key, Silver Key, Gold Key“ (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response

must be a single valid JSON object and nothing else.

Hindi.

आप अलग-अलग कमरों वाली एक इमारत में घूम रहे हैं। प्रत्येक कमरे में ऐसी चाबियां हैं जो आपको दो संभावित कमरों में से किसी एक में ले जा सकती हैं, और हर एक की अपनी एक निश्चित प्रायिकता है।

आप कमरा U5506 में शुरू करते हैं।

- कमरा D5012 में, यदि आप चाँदी की चाबी का उपयोग करते हैं, तो आप 0.3 प्रायिकता से कमरा D5506 में या 0.7 प्रायिकता से कमरा A5012 में पहुँचते हैं।

- कमरा U5506 में, यदि आप चाँदी की चाबी का उपयोग करते हैं, तो आप 0.1 प्रायिकता से कमरा D5012 में या 0.9 प्रायिकता से कमरा U5506 में पहुँचते हैं।
- कमरा A5012 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप 0.2 प्रायिकता से कमरा D5012 में या 0.8 प्रायिकता से कमरा D5506 में पहुँचते हैं।
- कमरा D5506 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप 0.2 प्रायिकता से कमरा U5506 में या 0.8 प्रायिकता से कमरा U5012 में पहुँचते हैं।
- कमरा D5012 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप 0.1 प्रायिकता से कमरा A5012 में या 0.9 प्रायिकता से कमरा D5012 में पहुँचते हैं।
- कमरा U5012 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप 0.1 प्रायिकता से कमरा A5012 में या 0.9 प्रायिकता से कमरा U5012 में पहुँचते हैं।
- कमरा D5506 में, यदि आप चाँदी की चाबी का उपयोग करते हैं, तो आप 0.2 प्रायिकता से कमरा U5506 में या 0.8 प्रायिकता से कमरा A5012 में पहुँचते हैं।
- कमरा U5012 में, यदि आप चाँदी की चाबी का उपयोग करते हैं, तो आप 0.2 प्रायिकता से कमरा U5012 में या 0.8 प्रायिकता से कमरा D5506 में पहुँचते हैं।
- कमरा U5506 में, यदि आप सोने की चाबी का उपयोग करते हैं, तो आप 0.3 प्रायिकता से कमरा A5012 में या 0.7 प्रायिकता से कमरा U5012 में पहुँचते हैं।
- कमरा A5012 में, यदि आप चाँदी की चाबी का उपयोग करते हैं, तो आप 0.1 प्रायिकता से कमरा D5012 में या 0.9 प्रायिकता से कमरा U5012 में पहुँचते हैं।

यदि आप चाबियों के निम्नलिखित क्रम का उपयोग करते हैं: सोने की चाबी, सोने की चाबी, सोने की चाबी, सोने की चाबी, चाँदी की चाबी, सोने की चाबी, सोने की चाबी, सोने की चाबी, चाँदी की चाबी, चाँदी की चाबी, चाँदी की चाबी, चाँदी की चाबी, सोने की चाबी, चाँदी की चाबी, चाँदी की चाबी, तो वे टॉप 8 कमरे कौनसे हैं जिनमें आपके पहुँचने की प्रायिकता सबसे ज़्यादा है? प्रायिकता के घटते क्रम में सूचीबद्ध करें।

उदाहरण प्रतिक्रिया: {“states”: [“कमरा G5374”, “कमरा W2169”, “कमरा T3803”, “कमरा R5010”, “कमरा F8573”, “कमरा M5422”, “कमरा U4598”, “कमरा V6313”]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 3 most probable rooms at that point. For example, if the sequence is “Gold Key, Silver Key, Gold Key“ (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Japanese.

あなたは、さまざまな部屋がある建物の中を移動しています。各部屋には鍵があり、鍵を使うと、候補となる2つの部屋のうちの1つに一定の確率で入れます。

あなたは部屋 U5506 から始めます。

- 部屋 D5012 で銀の鍵を使うと、0.3 の確率で部屋 D5506、または 0.7 の確率で部屋 A5012 に入ります。
- 部屋 U5506 で銀の鍵を使うと、0.1 の確率で部屋 D5012、または 0.9 の確率で部屋 U5506 に入ります。
- 部屋 A5012 で金の鍵を使うと、0.2 の確率で部屋 D5012、または 0.8 の確率で部屋 D5506 に入ります。
- 部屋 D5506 で金の鍵を使うと、0.2 の確率で部屋 U5506、または 0.8 の確率で部屋 U5012 に入ります。
- 部屋 D5012 で金の鍵を使うと、0.1 の確率で部屋 A5012、または 0.9 の確率で部屋 D5012 に入ります。
- 部屋 U5012 で金の鍵を使うと、0.1 の確率で部屋 A5012、または 0.9 の確率で部屋 U5012 に入ります。
- 部屋 D5506 で銀の鍵を使うと、0.2 の確率で部屋 U5506、または 0.8 の確率で部屋 A5012 に入ります。
- 部屋 U5012 で銀の鍵を使うと、0.2 の確率で部屋 U5012、または 0.8 の確率で部屋 D5506 に入ります。
- 部屋 U5506 で金の鍵を使うと、0.3 の確率で部屋 A5012、または 0.7 の確率で部屋 U5012 に入ります。

- 部屋 A5012 で銀の鍵を使うと、0.1 の確率で部屋 D5012、または 0.9 の確率で部屋 U5012 に入ります。

次の鍵のシーケンスを使用した場合: 金の鍵, 金の鍵, 金の鍵, 金の鍵, 銀の鍵, 金の鍵, 金の鍵, 金の鍵, 銀の鍵, 銀の鍵, 銀の鍵, 銀の鍵, 金の鍵, 銀の鍵, 銀の鍵、最終的に到達する可能性が最も高い上位8つの部屋は何ですか？最も可能性の高いものから低いものの順にリストしてください。

回答の例: {“states”: [“部屋 G5374”, “部屋 W2169”, “部屋 T3803”, “部屋 R5010”, “部屋 F8573”, “部屋 M5422”, “部屋 U4598”, “部屋 V6313”]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 3 most probable rooms at that point. For example, if the sequence

is “Gold Key, Silver Key, Gold Key“ (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response

must be a single valid JSON object and nothing else.

Tamil.

நீங்கள் வெவ்வேறு அறைகள் கொண்ட ஒரு கட்டிடத்திற்குள் செல்கிறீர்கள். ஒவ்வொரு அறையிலும் சாவிகள் உள்ளன. அவை உங்களை இரண்டு சாத்தியமான அறைகளில் ஒன்றுக்கு அழைத்துச் செல்லக்கூடும், ஒவ்வொன்றும் ஒரு குறிப்பிட்ட சாத்தியக்கூறைக் கொண்டுள்ளது.

நீங்கள் அறை U5506 இல் தொடங்குகிறீர்கள்.

- அறை D5012 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.3 சாத்தியக்கூறுடன் அறை D5506 க்கு அல்லது 0.7 சாத்தியக்கூறுடன் அறை A5012 க்குச் செல்வீர்கள்.
- அறை U5506 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.1 சாத்தியக்கூறுடன் அறை D5012 க்கு அல்லது 0.9 சாத்தியக்கூறுடன் அறை U5506 க்குச் செல்வீர்கள்.
- அறை A5012 இல், நீங்கள் தங்க சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.2 சாத்தியக்கூறுடன் அறை D5012 க்கு அல்லது 0.8 சாத்தியக்கூறுடன் அறை D5506 க்குச் செல்வீர்கள்.
- அறை D5506 இல், நீங்கள் தங்க சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.2 சாத்தியக்கூறுடன் அறை U5506 க்கு அல்லது 0.8 சாத்தியக்கூறுடன் அறை U5012 க்குச் செல்வீர்கள்.
- அறை D5012 இல், நீங்கள் தங்க சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.1 சாத்தியக்கூறுடன் அறை A5012 க்கு அல்லது 0.9 சாத்தியக்கூறுடன் அறை D5012 க்குச் செல்வீர்கள்.
- அறை U5012 இல், நீங்கள் தங்க சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.1 சாத்தியக்கூறுடன் அறை A5012 க்கு அல்லது 0.9 சாத்தியக்கூறுடன் அறை U5012 க்குச் செல்வீர்கள்.
- அறை D5506 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.2 சாத்தியக்கூறுடன் அறை U5506 க்கு அல்லது 0.8 சாத்தியக்கூறுடன் அறை A5012 க்குச் செல்வீர்கள்.
- அறை U5012 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.2 சாத்தியக்கூறுடன் அறை

U5012 க்கு அல்லது 0.8 சாத்தியக்கூறுடன் அறை D5506 க்குச் செல்வீர்கள்.

- அறை U5506 இல், நீங்கள் தங்க சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.3 சாத்தியக்கூறுடன் அறை A5012 க்கு அல்லது 0.7 சாத்தியக்கூறுடன் அறை U5012 க்குச் செல்வீர்கள்.
- அறை A5012 இல், நீங்கள் வெள்ளிச் சாவி ஐப் பயன்படுத்தினால், நீங்கள் 0.1 சாத்தியக்கூறுடன் அறை D5012 க்கு அல்லது 0.9 சாத்தியக்கூறுடன் அறை U5012 க்குச் செல்வீர்கள்.

பின்வரும் சாவிகளின் வரிசையை நீங்கள் பயன்படுத்தினால்: தங்க சாவி, தங்க சாவி, தங்க சாவி, தங்க சாவி, வெள்ளிச் சாவி, தங்க சாவி, தங்க சாவி, தங்க சாவி, வெள்ளிச் சாவி, வெள்ளிச் சாவி, வெள்ளிச் சாவி, வெள்ளிச் சாவி, தங்க சாவி, வெள்ளிச் சாவி, வெள்ளிச் சாவி, நீங்கள் முடிவடையக்கூடிய அதிக நிகழ்தகவு 8 அறைகள் எவை? அதிக நிகழ்தகவிலிருந்து குறைவான நிகழ்தகவு வரை வரிசையாக பட்டியலிடுங்கள். எடுத்துக்காட்டு பதில்: {“states”: [“அறை G5374”, “அறை W2169”, “அறை T3803”, “அறை R5010”, “அறை F8573”, “அறை M5422”, “அறை U4598”, “அறை V6313”]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 3 most probable rooms at that point. For example, if the sequence is “Gold Key, Silver Key, Gold Key” (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Telugu.

మీరు వివిధ గదులు ఉన్న భవనంలో నడుస్తున్నారు. ప్రతి గదిలో తాళంచెవులు ఉన్నాయి, అవి మిమ్మల్ని రెండు సాధ్యమైన గదులలో ఒకదానికి తీసుకెళ్ళవచ్చు, ప్రతిదానికీ ఒక నిర్దిష్ట సంభావ్యత ఉంటుంది.

మీరు గది U5506 లో ప్రారంభిస్తారు.

- గది D5012 లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.3 సంభావ్యతతో గది D5506 కి లేదా 0.7 సంభావ్యతతో గది A5012 కి చేరుకుంటారు.
- గది U5506 లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.1 సంభావ్యతతో గది D5012 కి లేదా 0.9 సంభావ్యతతో గది U5506 కి చేరుకుంటారు.
- గది A5012 లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.2 సంభావ్యతతో గది D5012 కి లేదా 0.8 సంభావ్యతతో గది D5506 కి చేరుకుంటారు.
- గది D5506 లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.2 సంభావ్యతతో గది U5506 కి లేదా 0.8 సంభావ్యతతో గది U5012 కి చేరుకుంటారు.
- గది D5012 లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.1 సంభావ్యతతో గది A5012 కి లేదా 0.9 సంభావ్యతతో గది D5012 కి చేరుకుంటారు.
- గది U5012 లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.1 సంభావ్యతతో గది A5012 కి లేదా 0.9 సంభావ్యతతో గది U5012 కి చేరుకుంటారు.
- గది D5506 లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.2 సంభావ్యతతో గది U5506 కి లేదా 0.8 సంభావ్యతతో గది A5012 కి చేరుకుంటారు.
- గది U5012 లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.2 సంభావ్యతతో గది U5012 కి లేదా 0.8 సంభావ్యతతో గది D5506 కి చేరుకుంటారు.
- గది U5506 లో, మీరు బంగారు తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.3 సంభావ్యతతో గది A5012 కి లేదా 0.7 సంభావ్యతతో గది U5012 కి చేరుకుంటారు.
- గది A5012 లో, మీరు వెండి తాళంచెవి ని ఉపయోగిస్తే, మీరు 0.1 సంభావ్యతతో

గది D5012 కి లేదా 0.9 సంభావ్యతతో గది U5012 కి చేరుకుంటారు.

మీరు క్రింది తాళంచెవుల క్రమాన్ని ఉపయోగిస్తే: బంగారు తాళంచెవి, బంగారు తాళంచెవి, బంగారు తాళంచెవి, బంగారు తాళంచెవి, వెండి తాళంచెవి, బంగారు తాళంచెవి, బంగారు తాళంచెవి, బంగారు తాళంచెవి, వెండి తాళంచెవి, వెండి తాళంచెవి, వెండి తాళంచెవి, వెండి తాళంచెవి, బంగారు తాళంచెవి, వెండి తాళంచెవి, వెండి తాళంచెవి, మీరు చేరుకోగల అత్యంత సంభావ్య 8 గదులు ఏవి? అత్యంత సంభావ్యం నుండి తక్కువ సంభావ్యం వరకు క్రమంలో జాబితా చేయండి.

ఉదాహరణ ప్రతిస్పందన: {“states”: [“గది G5374”, “గది W2169”, “గది T3803”, “గది R5010”, “గది F8573”, “గది M5422”, “గది U4598”, “గది V6313”]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 3 most probable rooms at that point. For example, if the sequence is “Gold Key, Silver Key, Gold Key“ (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

Chinese.

你正在一栋建筑物中穿行，里面有不同的房间，每个房间里都有一些钥匙。使用钥匙后，你可以前往两个特定房间中的一个，且前往每个房间的概率是预设好的。

你从房间 U5506 开始。

- 在房间 D5012，如果使用银钥匙，你有 0.3 的概率会到达房间 D5506，有 0.7 的概率会到达房间 A5012。
- 在房间 U5506，如果使用银钥匙，你有 0.1 的概率会到达房间 D5012，有 0.9 的概率会到达房间 U5506。
- 在房间 A5012，如果使用金钥匙，你有 0.2 的概率会到达房间 D5012，有 0.8 的概率会到达房间 D5506。
- 在房间 D5506，如果使用金钥匙，你有 0.2 的概率会到达房间 U5506，有 0.8 的概率会到达房间 U5012。
- 在房间 D5012，如果使用金钥匙，你有 0.1 的概率会到达房间 A5012，有 0.9 的概率会到达房间 D5012。
- 在房间 U5012，如果使用金钥匙，你有 0.1 的概率会到达房间 A5012，有 0.9 的概率会到达房间 U5012。
- 在房间 D5506，如果使用银钥匙，你有 0.2 的概率会到达房间 U5506，有 0.8 的概率会到达房间 A5012。
- 在房间 U5012，如果使用银钥匙，你有 0.2 的概率会到达房间 U5012，有 0.8 的概率会到达房间 D5506。
- 在房间 U5506，如果使用金钥匙，你有 0.3 的概率会到达房间 A5012，有 0.7 的概率会到达房间 U5012。
- 在房间 A5012，如果使用银钥匙，你有 0.1 的概率会到达房间 D5012，有 0.9 的概率会到达房间 U5012。

假设按以下顺序使用钥匙：金钥匙, 金钥匙, 金钥匙, 金钥匙, 银钥匙, 金钥匙, 金钥匙, 金钥匙, 银钥匙, 银钥匙, 银钥匙, 银钥匙, 金钥匙, 银钥匙, 银钥匙。请问你最终所处位置概率最高的 8 个房间分别是？请按概率从高到低依次列出。

响应示例：{“states”: [“房间 G5374”, “房间 W2169”, “房间 T3803”, “房间 R5010”, “房间 F8573”, “房间 M5422”, “房间 U4598”, “房间 V6313”]}

IMPORTANT: Solve the puzzle step by step. Apply each trigger in the sequence exactly once, in order. The sequence has exactly 15 triggers. After applying all 15 triggers, report the top 3 most probable rooms at that point. For example, if the sequence is “Gold Key, Silver Key, Gold Key“ (3 triggers), you apply trigger 1 (Gold Key), then trigger 2 (Silver Key), then trigger 3 (Gold Key), and report the top 3 rooms after exactly 3 steps. IMPORTANT: Your response must be a single valid JSON object and nothing else.

C.3.4 Graph Shortest Path

English.

Social network connections:

1. W9928 and C4582 are neighbors.
2. C7912 and R4257 are classmates.
3. H4657 and C7912 are neighbors.
4. H9279 and U2824 are classmates.
5. U2824 and H4657 are coworkers.
6. A5506 and B1488 are coworkers.
7. W9928 and V9935 are relatives.
8. T1434 and V9935 are relatives.
9. B1488 and R4257 are relatives.
10. E2679 and T1434 are relatives.
11. E2679 and A5506 are classmates.
12. H9279 and C4582 are friends.

Question: What is the shortest chain of connections from H4657 to V9935? List the people in order from H4657 to V9935, including both H4657 and V9935.

Solve the puzzle step by step using BFS algorithm. At the very end, only output your final answer as a single JSON object on its own line in this exact format: {"path": ["A1234", "B5678", "C9012"]} The path should list all nodes from the source to the target in order, including both the source and the target. Do not include any other text after the JSON.

Arabic.

الاجتماعية: الشبكة روابط

1. W9928 وC4582 جيران.
2. C7912 R4257 وزملاء دراسة.
3. H4657 C7912 وجيران.
4. H9279 U2824 وزملاء دراسة.
5. U2824 H4657 وزملاء عمل.
6. A5506 B1488 وزملاء عمل.
7. W9928 V9935 وأقارب.
8. T1434 V9935 وأقارب.
9. B1488 R4257 وأقارب.
10. E2679 T1434 وأقارب.
11. E2679 A5506 وزملاء دراسة.
12. H9279 C4582 وأصدقاء.

السؤال: ما هي أقصر سلسلة اتصالات من H4657 إلى V9935؟ اذكر الأشخاص بالترتيب من H4657 إلى V9935، بما في ذلك H4657 وV9935.

Hindi.

सामाजिक नेटवर्क के संबंध:

1. W9928 और C4582 पड़ोसी हैं।
2. C7912 और R4257 सहपाठी हैं।
3. H4657 और C7912 पड़ोसी हैं।
4. H9279 और U2824 सहपाठी हैं।
5. U2824 और H4657 सहकर्मी हैं।
6. A5506 और B1488 सहकर्मी हैं।
7. W9928 और V9935 रिश्तेदार हैं।
8. T1434 और V9935 रिश्तेदार हैं।
9. B1488 और R4257 रिश्तेदार हैं।
10. E2679 और T1434 रिश्तेदार हैं।
11. E2679 और A5506 सहपाठी हैं।
12. H9279 और C4582 मित्र हैं।

प्रश्न: H4657 से V9935 तक पहुँचने का सबसे छोटा संपर्क मार्ग क्या है? H4657 से V9935 तक के सभी लोगों को क्रम में बताइए, H4657 और V9935 दोनों को शामिल करते हुए।

Solve the puzzle step by step using BFS algorithm. At the very end, only output your final answer as a single JSON object on its own line in this exact format: {"path": ["A1234", "B5678", "C9012"]} The path should list all nodes from the source to the target in order, including both the source and the target. Do not include any other text after the JSON.

Japanese.

ソーシャルネットワークのつながり：

1. W9928と C4582は隣人です。
2. C7912と R4257は同級生です。
3. H4657と C7912は隣人です。
4. H9279と U2824は同級生です。
5. U2824と H4657は同僚です。
6. A5506と B1488は同僚です。
7. W9928と V9935は親戚です。
8. T1434と V9935は親戚です。
9. B1488と R4257は親戚です。
10. E2679と T1434は親戚です。
11. E2679と A5506は同級生です。
12. H9279と C4582は友達です。

質問：H4657から V9935までの最も短いつながりは何ですか？H4657から V9935 までの人物を、H4657 および V9935を含めて順番に挙げてください。

Solve the puzzle step by step using BFS algorithm. At the very end, only output your

final answer as a single JSON object on its own line in this exact format: {"path": ["A1234", "B5678", "C9012"]} The path should list all nodes from the source to the target in order, including both the source and the target. Do not include any other text after the JSON.

Tamil.

சமூக வலைப்பின்னல் இணைப்புகள்:

1. W9928 மற்றும் C4582 அண்டை வீட்டார்.
2. C7912 மற்றும் R4257 வகுப்புத் தோழர்கள்.
3. H4657 மற்றும் C7912 அண்டை வீட்டார்.
4. H9279 மற்றும் U2824 வகுப்புத் தோழர்கள்.
5. U2824 மற்றும் H4657 சக ஊழியர்கள்.
6. A5506 மற்றும் B1488 சக ஊழியர்கள்.
7. W9928 மற்றும் V9935 உறவினர்கள்.
8. T1434 மற்றும் V9935 உறவினர்கள்.
9. B1488 மற்றும் R4257 உறவினர்கள்.
10. E2679 மற்றும் T1434 உறவினர்கள்.
11. E2679 மற்றும் A5506 வகுப்புத் தோழர்கள்.
12. H9279 மற்றும் C4582 நண்பர்கள்.

கேள்வி: H4657 இலிருந்து V9935 வரையிலான மிகக் குறுகிய தொடர்பு சங்கிலி என்ன? H4657 இலிருந்து V9935 வரை வரிசையாக நபர்களைப் பட்டியலிடுங்கள், H4657 மற்றும் V9935 இருவரையும் சேர்த்து.

Solve the puzzle step by step using BFS algorithm. At the very end, only output your final answer as a single JSON object on its own line in this exact format: {"path": ["A1234", "B5678", "C9012"]} The path should list all nodes from the source to the target in order, including both the source and the target. Do not include any other text after the JSON.

Telugu.

సామాజిక నెట్‌వర్క్ సంబంధాలు:

1. W9928 మరియు C4582 ఇరుగుపొరుగువారు.
2. C7912 మరియు R4257 సహవిద్యార్థులు.
3. H4657 మరియు C7912 ఇరుగుపొరుగువారు.
4. H9279 మరియు U2824 సహవిద్యార్థులు.
5. U2824 మరియు H4657 సహోద్యోగులు.
6. A5506 మరియు B1488 సహోద్యోగులు.
7. W9928 మరియు V9935 బంధువులు.
8. T1434 మరియు V9935 బంధువులు.
9. B1488 మరియు R4257 బంధువులు.
10. E2679 మరియు T1434 బంధువులు.
11. E2679 మరియు A5506 సహవిద్యార్థులు.
12. H9279 మరియు C4582 స్నేహితులు.

ప్రశ్న: H4657 మరియు V9935 మధ్య ఉన్న అతి చిన్న పరిచయ సంబంధం ఏమిటి?H4657 నుండి V9935 వరకు వ్యక్తులను వరుసగా తెలపండి,H4657 మరియు V9935 ఇద్దరినీ చేర్చండి.

Solve the puzzle step by step using BFS algorithm. At the very end, only output your final answer as a single JSON object on its own line in this exact format: {"path": ["A1234", "B5678", "C9012"]} The path should list all nodes from the source to the target in order, including both the source and the target. Do not include any other text after the JSON.

Chinese.

社交网络关系：

1. W9928和 C4582是邻居。
2. C7912和 R4257是同学。
3. H4657和 C7912是邻居。
4. H9279和 U2824是同学。
5. U2824和 H4657是同事。
6. A5506和 B1488是同事。
7. W9928和 V9935是亲戚。
8. T1434和 V9935是亲戚。
9. B1488和 R4257是亲戚。
10. E2679和 T1434是亲戚。

11. E2679和 A5506是同学。
12. H9279和 C4582是朋友。

问题：从 H4657到 V9935，最短的人脉关系链是什么？请按顺序列出所有将两人联系起来的人，包括 H4657和 V9935本人。

Solve the puzzle step by step using BFS algorithm. At the very end, only output your final answer as a single JSON object on its own line in this exact format: {"path": ["A1234", "B5678", "C9012"]} The path should list all nodes from the source to the target in order, including both the source and the target. Do not include any other text after the JSON.

C.3.5 Tournament Ranking

English.

Match results:

1. Player V9935 defeated Player A5506.
2. Player U2824 defeated Player A5506.
3. Player E2679 defeated Player H4657.
4. Player E2679 defeated Player A5506.
5. Player A5506 defeated Player H4657.
6. Player U2824 defeated Player E2679.
7. Player V9935 defeated Player E2679.
8. Player V9935 defeated Player U2824.
9. Player V9935 defeated Player H4657.
10. Player U2824 defeated Player H4657.

Question: Based on the results given above, rank all players from strongest to weakest.

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

There are exactly 5 players in total. Your final ranking must include all of them.

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like "D5506", not the full names) in this exact format: {"ranking": ["<best_id>", "<2nd_id>", ..., "<worst_id>"]} For example if Player A1234 is best and Player B5678 is worst: {"ranking": ["A1234", "B5678"]} Do not include any other text after the JSON.

Arabic.

البطولة: نتائج

1. لاعب هزم V9935 لاعب A5506.
2. لاعب هزم U2824 لاعب A5506.
3. لاعب هزم E2679 لاعب H4657.
4. لاعب هزم E2679 لاعب A5506.
5. لاعب هزم A5506 لاعب H4657.
6. لاعب هزم U2824 لاعب E2679.
7. لاعب هزم V9935 لاعب E2679.
8. لاعب هزم V9935 لاعب U2824.
9. لاعب هزم V9935 لاعب H4657.
10. لاعب هزم U2824 لاعب H4657.

السؤال: بناءً على النتائج المذكورة، رتب جميع اللاعبين من الأقوى إلى الأضعف.

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

هناك بالضبط 5 لاعبين في المجموع. يجب أن يتضمن تصنيفك النهائي جميعهم.

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like "D5506", not the full names) in this exact format: {"ranking": ["<best_id>", "<2nd_id>", ..., "<worst_id>"]} For example if Player A1234 is best and Player B5678 is worst: {"ranking": ["A1234", "B5678"]} Do not include any other text after the JSON.

Hindi.

प्रतियोगिता के परिणाम:

1. खिलाड़ी V9935 ने खिलाड़ी A5506 को हराया।
2. खिलाड़ी U2824 ने खिलाड़ी A5506 को

हराया।
3. खिलाड़ी E2679 ने खिलाड़ी H4657 को हराया।
4. खिलाड़ी E2679 ने खिलाड़ी A5506 को हराया।
5. खिलाड़ी A5506 ने खिलाड़ी H4657 को हराया।
6. खिलाड़ी U2824 ने खिलाड़ी E2679 को हराया।
7. खिलाड़ी V9935 ने खिलाड़ी E2679 को हराया।
8. खिलाड़ी V9935 ने खिलाड़ी U2824 को हराया।
9. खिलाड़ी V9935 ने खिलाड़ी H4657 को हराया।
10. खिलाड़ी U2824 ने खिलाड़ी H4657 को हराया।

प्रश्न: ऊपर दिए गए परिणामों के आधार पर, सभी खिलाड़ियों को सर्वश्रेष्ठ से सबसे कमज़ोर तक क्रमित करें।

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

इसमें कुल मिलाकर 5 खिलाड़ी हैं। आपकी रैंकिंग में सभी खिलाड़ी शामिल होने चाहिए।

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like “D5506”, not the full names) in this exact format: {“ranking”: [“<best_id>“, “<2nd_id>“, ..., “<worst_id>“]} For example if Player A1234 is best and Player B5678 is worst: {“ranking”: [“A1234”, “B5678”]} Do not include any other text after the JSON.

Japanese.

大会結果:
1. プレイヤー V9935 はプレイヤー A5506 に勝利した。
2. プレイヤー U2824 はプレイヤー A5506 に勝利した。
3. プレイヤー E2679 はプレイヤー H4657 に勝利した。
4. プレイヤー E2679 はプレイヤー A5506 に勝利した。
5. プレイヤー A5506 はプレイヤー H4657 に勝利した。
6. プレイヤー U2824 はプレイヤー E2679 に勝利した。
7. プレイヤー V9935 はプレイヤー E2679 に勝利した。
8. プレイヤー V9935 はプレイヤー U2824 に勝利した。
9. プレイヤー V9935 はプレイヤー H4657 に勝利した。
10. プレイヤー U2824 はプレイヤー H4657 に勝利した。

質問: 上記の結果に基づいて、すべてのプレイヤーを強い人から弱い人へ順に並べてください。

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

合計でちょうど 5 人のプレイヤーがいます。最終的なランキングには全員を含める必要があります。

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like “D5506”, not the full names) in this exact format: {“ranking”: [“<best_id>“, “<2nd_id>“, ..., “<worst_id>“]} For example if Player A1234 is best and Player B5678 is worst: {“ranking”: [“A1234”, “B5678”]} Do not include any other text after the JSON.

Tamil.

ரூர்னமென்ட் முடிவுகள்
1. வீரர் V9935 வீரர் A5506ஐ தோற்கடித்தார்.
2. வீரர் U2824 வீரர் A5506ஐ தோற்கடித்தார்.
3. வீரர் E2679 வீரர் H4657ஐ தோற்கடித்தார்.

4. வீரர் E2679 வீரர் A5506ஐ தோற்கடித்தார்.
5. வீரர் A5506 வீரர் H4657ஐ தோற்கடித்தார்.
6. வீரர் U2824 வீரர் E2679ஐ தோற்கடித்தார்.
7. வீரர் V9935 வீரர் E2679ஐ தோற்கடித்தார்.
8. வீரர் V9935 வீரர் U2824ஐ தோற்கடித்தார்.
9. வீரர் V9935 வீரர் H4657ஐ தோற்கடித்தார்.
10. வீரர் U2824 வீரர் H4657ஐ தோற்கடித்தார்.

கேள்வி: மேலே உள்ள முடிவுகளின் அடிப்படையில், அனைத்து வீரர்களையும் வலிமையானவரிலிருந்து பலவீனமானவர் வரை வரிசைப்படுத்துங்கள்.

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

மொத்தம் சரியாக 5 வீரர்கள் உள்ளனர். உங்கள் தரவரிசையில் அந்த அனைத்து வீரர்களும் இடம்பெற வேண்டும்.

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like “D5506”, not the full names) in this exact format: {“ranking”: [“<best_id>“, “<2nd_id>“, ..., “<worst_id>“]} For example if Player A1234 is best and Player B5678 is worst: {“ranking”: [“A1234”, “B5678”]} Do not include any other text after the JSON.

Telugu.

టోర్నమెంట్ ఫలితాలు:

1. ప్లేయర్ V9935 ప్లేయర్ A5506ను ఓడించారు.
2. ప్లేయర్ U2824 ప్లేయర్ A5506ను ఓడించారు.
3. ప్లేయర్ E2679 ప్లేయర్ H4657ను ఓడించారు.
4. ప్లేయర్ E2679 ప్లేయర్ A5506ను ఓడించారు.
5. ప్లేయర్ A5506 ప్లేయర్ H4657ను ఓడించారు.
6. ప్లేయర్ U2824 ప్లేయర్ E2679ను ఓడించారు.
7. ప్లేయర్ V9935 ప్లేయర్ E2679ను ఓడించారు.
8. ప్లేయర్ V9935 ప్లేయర్ U2824ను ఓడించారు.
9. ప్లేయర్ V9935 ప్లేయర్ H4657ను ఓడించారు.
10. ప్లేయర్ U2824 ప్లేయర్ H4657ను ఓడించారు.

ప్రశ్న: పైన ఇచ్చిన ఫలితాల ఆధారంగా, అందరు ప్లేయర్ బలమైన వారి నుండి బలహీనమైన వారి వరకు ర్యాంక్ చేయండి.

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

మొత్తం 5 మంది ప్లేయర్స్ ఉన్నారు. మీ ఫైనల్ ర్యాంకింగ్‌లో అందరినీ చేర్చాలి.

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like “D5506”, not the full names) in this exact format: {“ranking”: [“<best_id>“, “<2nd_id>“, ..., “<worst_id>“]} For example if Player A1234 is best and Player B5678 is worst: {“ranking”: [“A1234”, “B5678”]} Do not include any other text after the JSON.

Chinese.

比赛结果：

1. 选手 V9935 击败了选手 A5506。
2. 选手 U2824 击败了选手 A5506。
3. 选手 E2679 击败了选手 H4657。
4. 选手 E2679 击败了选手 A5506。

5. 选手 A5506 击败了选手 H4657。
6. 选手 U2824 击败了选手 E2679。
7. 选手 V9935 击败了选手 E2679。
8. 选手 V9935 击败了选手 U2824。
9. 选手 V9935 击败了选手 H4657。
10. 选手 U2824 击败了选手 H4657。

问题：根据上述结果，将所有选手按实力从强到弱进行排序。

This is a topological ranking puzzle. The match results define a directed acyclic graph where each edge represents one player defeating another. Your task is to find the unique total ordering consistent with all results.

共有 5 名选手。您的最终排名必须包括所有选手。

Solve the puzzle step by step. At the very end, only output your final answer as a single JSON object on its own line. Use only the player IDs (the alphanumeric codes like “D5506”, not the full names) in this exact format: {“ranking”: [“<best_id>“, “<2nd_id>“, ..., “<worst_id>“]} For example if Player A1234 is best and Player B5678 is worst: {“ranking”: [“A1234”, “B5678”]} Do not include any other text after the JSON.