Semantic Distance for Presburger Formula Generation

Masaya Taniguchi
RIKEN
masaya.taniguchi@riken.jp

Hitomi Yanaka
The University of Tokyo
RIKEN
Tohoku University
hyanaka@is.s.u-tokyo.ac.jp

Abstract

We study how to evaluate and train natural-language-to-formula generation when surface similarity is a poor proxy for semantic correctness. Focusing on translation into Presburger arithmetic, we introduce a geometric distance between formulas by viewing each formula as a set of integer lattice points and assigning an exponentially decaying weight to that set. The resulting distance yields finite comparisons even for infinite definable sets, can be computed through quantifier elimination, polyhedral decomposition, and lattice-point generating functions, and inherits the standard metric properties. We then use this distance in experiments on supervised fine-tuning (SFT) and Group Relative Policy Optimization (GRPO) for translating arithmetic statements into formal formulas. The results show that a distance-aware reward substantially improves parsability and adjusted semantic quality compared with a string-similarity-only reward, while also revealing the remaining challenge of preserving quantifier structure.

1 Introduction

Natural language models are increasingly asked to map descriptions into formal logical representations, from semantic parsing (Zettlemoyer and Collins, 2005; Liang et al., 2011; Dong and Lapata, 2016; Herzig et al., 2021) to program synthesis (Yin and Neubig, 2017). In these tasks, evaluating predictions against reference formulas is fundamentally difficult: string-based metrics such as BLEU, ROUGE, or exact match fail to capture semantic correctness. Two formulas can differ by a single character yet define entirely different solution sets, while a near-identical string can differ only in punctuation and preserve meaning. This gap motivates metrics that operate on the meaning of formulas rather than their surface form. Such metrics are especially important when small syntactic changes alter the space of valid numerical assignments.

Example 1 (Motivating example)**. Statement**: Alice has at least 6 apples, which is at least as many as Bob has oranges.
Reference: $x \geq 6 \wedge x \geq y$ (Alice has x apples, Bob has y oranges)
Prediction: $x \geq 6 \wedge x > y$
String similarity: ≈ 1 (single character difference)
Geometric distance: $d_\beta > 0$ (strict subset)
The prediction excludes valid boundary points $x = y$. Both surface similarity and semantic distance are needed to evaluate models faithfully.

While this paper situates itself within the broader context of semantic parsing, we explicitly restrict our focus to Presburger arithmetic; *i.e.*, linear integer arithmetic with addition and order. This domain serves as a well-defined and rigorous case study for evaluating semantics-aware rewards. We emphasize that Presburger arithmetic excludes multiplication between variables, polynomial equations, integrals, or higher-order logic commonly found in open-domain scientific texts. Generalizing our geometric distance metric to such richer formal languages remains highly non-trivial due to the lack of efficient quantifier elimination or tractable lattice-point counting methods; thus, this work represents a foundational first step rather than a universal solution.

The choice of Presburger arithmetic is deliberate. It is expressive enough to represent many elementary linear arithmetic statements, while remaining sufficiently well behaved for quantifier elimination and lattice-point counting. This makes it a suitable controlled testbed for studying semantics-aware rewards. At the same time, the restriction is essential: our method should be understood as a first step for linear integer arithmetic rather than as a general-purpose semantic distance for arbitrary mathematical language.

We address this by defining a **weighted geomet-**

Proceedings of the 6th Workshop on Natural Language Meets Logic and Machine Learning (NALOMA), pages 71–79

ric distance on Presburger-definable sets, interpreting each formula as the set of integer lattice points it entails. For instance, $x > 2$ and $x > 0$ share many lattice points, while $x > 1$ shares more with $x > 0$ than $0 > x$ does. The distance quantifies exactly this degree of overlap using an exponentially decaying weight. This yields a finite, semantics-aware comparison even for infinite sets, bridging the gap between strict symbolic equivalence (distance zero) and unconstrained string similarity (distance near one).

We show this distance is computable by combining *quantifier elimination* (Theorem 1), *polyhedral decomposition* (Section 3), and *lattice-point generating functions* (Section 3; Loera et al., 2004), classical tools from logic and discrete geometry. In experiments on natural-language-to-formula translation with supervised fine-tuning (SFT) and Group Relative Policy Optimization (GRPO), a distance-aware reward improves parsability from 76.5% to 98.6% and reduces semantic distance from 0.295 to 0.167 compared to a string-similarity-only baseline.

In summary, this paper makes three contributions:

1. We define a weighted geometric distance on Presburger-definable sets and show that it inherits the standard metric properties of weighted Jaccard distances.

2. We present a practical computation pipeline using quantifier elimination and polyhedral decomposition, establishing tractable cases.

3. We demonstrate through SFT and GRPO experiments that our distance reward substantially improves parsability and semantic quality, while critically discussing its limitations regarding synthetic datasets, quantifier preservation, and domain generalizability.

2 Preliminaries

Definition 1 (Presburger arithmetic)**.** *Presburger arithmetic* is the first-order theory of integers equipped with addition, order, constants, logical connectives, and quantifiers (Cooper, 1972; **?**). Multiplication between variables is excluded. Formulas are constructed from atomic constraints of the form

$$a_1x_1+\cdots+a_dx_d \leq b, \qquad a_1x_1+\cdots+a_dx_d = b,$$

where the coefficients a_i and the constant b are integers. Composite formulas are formed using the Boolean connectives $\wedge$, $\vee$, $\neg$, and the quantifiers $\forall$, $\exists$.

One of the key facts underlying our method is that Presburger arithmetic is *decidable* and, moreover, admits *quantifier elimination* (Cooper, 1972; **?**). Therefore, any logical formula can be transformed into a quantifier-free description. Geometrically, such definable sets can be understood as finite unions of regions cut out by linear constraints on the integer lattice. This observation allows one to pass from logical formulas to polyhedral pieces amenable to geometric analysis.

Theorem 1 (Quantifier elimination for Presburger arithmetic)**.** *For any Presburger formula $\varphi(\mathbf{x})$, there exists a quantifier-free formula $\psi(\mathbf{x})$ that defines the same subset of $\mathbf{Z}^d$ as φ. Moreover, ψ can be written as a finite Boolean combination of linear inequalities, linear equalities, and congruence conditions of the form $a_1x_1 + \cdots + a_dx_d \equiv c \pmod m$.*

This theorem is the bridge connecting logic and geometry in our study. Once all quantifiers are eliminated, the resulting formula can be decomposed into finitely many linear regions and periodic conditions.

Example 2 (Quantifier elimination)**.** Consider the formula

$$\exists z\,(x = 2z + 1 \wedge 0 \leq z \wedge z \leq 2).$$

The conjuncts $0 \leq z \wedge z \leq 2$ bound the auxiliary variable z, while $x = 2z + 1$ links x to z. Eliminating z removes the dependence on the internal variable and yields an equivalent quantifier-free formula:

$$x = 1 \vee x = 3 \vee x = 5.$$

The same solution set can additionally be expressed compactly as

$$1 \leq x \leq 5 \wedge x \equiv 1 \pmod 2.$$

This illustrates the general theorem: bounding constraints on quantified variables become explicit threshold conditions, while linear dependencies decompose into congruence conditions on the remaining free variables.

We identify each formula $\varphi(\mathbf{x})$ with its model set $S_\varphi = \{\mathbf{x} \in \mathbf{Z}^d \mid \varphi(\mathbf{x})\}$, so logical connectives correspond to set operations: $\wedge$ to $\cap$, $\vee$ to $\cup$, and $\neg$ to set complementation.

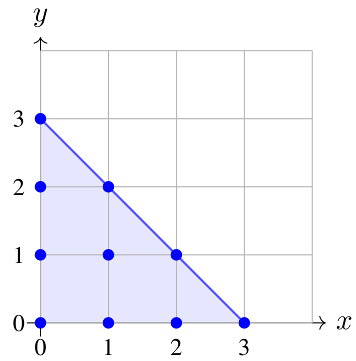

Figure 1: Integer solutions in the two-dimensional plane satisfying the Presburger formula $x + y \leq 3 \wedge x \geq 0 \wedge y \geq 0$.

Example 3 (Basic Presburger formulas)**.** The formula $x + y \leq 3 \wedge x \geq 0 \wedge y \geq 0$ defines the lattice points in a two-dimensional simplex (Figure 1). The formula $\exists z\,(x = 2z)$ defines the set of all even numbers.

On top of this representation, the distance in the next section compares two formulas through the difference in weighted mass of their model sets.

3 Geometric Distance and Metricity

In this section, we define a **weighted Jaccard distance** on Presburger-definable sets, viewing each formula as a set of integer lattice points with exponentially decaying weights. The construction generalizes the standard Jaccard index (union and intersection) to a weighted measure, yielding a finite-size-aware comparison even for infinite sets.

Definition 2 (Weighted Presburger measure)**.** Fix $\beta > 0$, and let $\varphi(\mathbf{x})$ be a Presburger formula over $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbf{Z}^d$. Then the weighted measure of φ is defined by

$$\mu_\beta(\varphi) = \frac{1}{Z_{\beta,d}} \sum_{\mathbf{x} \in S_\varphi} e^{-\beta \|\mathbf{x}\|_1}$$

where the normalization constant is

$$Z_{\beta,d} = \left(\frac{1 + e^{-\beta}}{1 - e^{-\beta}} \right)^d .$$

These definitions yield a probability mass function on $\mathbf{Z}^d$, and the following theorem confirms it is well-defined.

Theorem 2 (Well-definedness and boundedness)**.** *For any Presburger formula φ and any $\beta > 0$, $\mu_\beta(\varphi)$ is well-defined and satisfies $0 \leq \mu_\beta(\varphi) \leq 1$.*

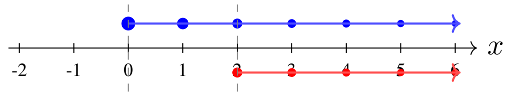

Figure 2: Threshold formulas $p(x)$ and $q(x)$. Points 0 and 1 dominate the distance.

Proof. Every weight $e^{-\beta\|\mathbf{x}\|_1}$ is nonnegative, so $\mu_\beta(\varphi) \geq 0$. The set S_φ is a subset of the full lattice $\mathbf{Z}^d$, so its weighted sum cannot exceed the total mass $Z_{\beta,d}$. Dividing by $Z_{\beta,d}$ therefore yields a value in $[0, 1]$. $\square$

The main computational challenge is to evaluate this weighted sum for an arbitrary Presburger formula. Quantifier elimination transforms the formula into a quantifier-free description, which decomposes into a finite family of rational polyhedra:

$$S_\varphi = \bigsqcup_{k=1}^{m} (P_k \cap \mathbf{Z}^d).$$

Each P_k is described by linear equalities and (strict or non-strict) inequalities. Since $\|\mathbf{x}\|_1$ is piecewise linear, we partition space into 2^d orthants indexed by sign vectors $\mathbf{s} \in \{\pm 1\}^d$ and apply a diagonal transformation $\mathbf{y} = \mathrm{diag}(\mathbf{s})\mathbf{x}$ to map each orthant into the nonnegative quadrant, where $\|\mathbf{x}\|_1$ becomes a linear sum. This yields

$$\mu_\beta(\varphi) = \frac{1}{Z_{\beta,d}} \sum_{k=1}^{m} \sum_{\mathbf{x} \in P_k \cap \mathbf{Z}^d} e^{-\beta\|\mathbf{x}\|_1}.$$

The exponential orthant partition 2^d is tractable for small d but may become prohibitive as dimension grows (see Section 6).

Example 4 (One-dimensional threshold formulas)**.** Let $d = 1$, and let $p(x)$ be the formula $x \geq 0$ and $q(x)$ the formula $x \geq 2$. Then $S_p = \{0, 1, 2, \ldots\}$ and $S_q = \{2, 3, 4, \ldots\}$. In one dimension, $Z_{\beta,1} = \frac{1+e^{-\beta}}{1-e^{-\beta}}$, so

$$\mu_\beta(p) = \frac{\sum_{n \geq 0} e^{-\beta n}}{Z_{\beta,1}}, \quad \mu_\beta(q) = \frac{\sum_{n \geq 2} e^{-\beta n}}{Z_{\beta,1}}.$$

Since $p \wedge q$ is equivalent to q and $p \vee q$ is equivalent to p, the distance is $d_\beta(p, q) = 1 - e^{-2\beta}$. In our experiments Section 4 we use $\beta = 1$.

After mapping into the nonnegative orthant, the sum can be computed with lattice-point generating functions (Ehrhart theory) (Woods, 2014). Let $Q \subseteq \mathbf{R}^d_{\geq 0}$ be a polyhedron and define $G_Q(\mathbf{t}) = \sum_{\mathbf{y} \in Q \cap \mathbf{Z}^d} t_1^{y_1} \cdots t_d^{y_d}$. Barvinok methods represent

G_Q as a rational function, evaluated at $t_i = e^{-\beta}$ to yield the required weighted count. The Barvinok representation is a compressed "weighted counting table": each monomial encodes a lattice point, and the rational-function form allows efficient comparison of $G_{p\wedge q}$ and $G_{p\vee q}$, making the distance a Jaccard-type overlap measure with emphasis near the origin.

We also require that basic order relations and triangle-inequality-type estimates hold in this weighted setting. Since every lattice-point weight is strictly positive, $A \subseteq B$ implies $\mu_\beta(A) \leq \mu_\beta(B)$, and $A \subseteq B \cup C$ implies $\mu_\beta(A) \leq \mu_\beta(B) + \mu_\beta(C)$. These monotonicity and subadditivity properties underpin the triangle inequality for $\delta_\beta(p, q) = \mu_\beta(p\triangle q)$.

Finally, the distance between two formulas p and q is defined as the Jaccard-type ratio with respect to the normalized weighted measure:

$$d_\beta(p, q) = 1 - \frac{\mu_\beta(p \wedge q)}{\mu_\beta(p \vee q)} = \frac{\mu_\beta(p\triangle q)}{\mu_\beta(p \vee q)},$$

where $p\triangle q = (p \wedge \neg q) \vee (q \wedge \neg p)$. When $\mu_\beta(p \vee q) = 0$ (both formulas define the empty set), we set $d_\beta(p, q) = 0$. This is a discrete weighted version of the classical Jaccard distance (Jaccard, 1901): larger shared probability mass implies closer formulas. The parameter β controls locality: large β concentrates mass near the origin, while small β gives more weight to distant points.

To verify that d_β defines a metric, we use symmetric difference.

Lemma 1 (Basic identities for μ_β)**.** *For Presburger formulas p and q, the measure μ_β satisfies:*

1. $\mu_\beta(p) \geq 0$.
2. $\mu_\beta(p \vee q) = \mu_\beta(p) + \mu_\beta(q) - \mu_\beta(p \wedge q)$.
3. $\mu_\beta(p\triangle q) = \mu_\beta(p \vee q) - \mu_\beta(p \wedge q)$.

Proof. Point (1) holds because all weights are nonnegative. For (2), adding the weights of p and q counts the overlap $p \wedge q$ twice, so subtracting it once gives the correct weight of $p \vee q$. For (3), the symmetric difference contains exactly those points that belong to p or q but not both, i.e., the union with the overlap removed, so its weight equals $\mu_\beta(p \vee q) - \mu_\beta(p \wedge q)$. □

Lemma 2 (Positivity and symmetric-difference representation)**.** *For Presburger formulas p and q, if $\mu_\beta(p \vee q) > 0$, then*

$$d_\beta(p, q) = \frac{\mu_\beta(p\triangle q)}{\mu_\beta(p \vee q)}$$

holds. Moreover, if p and q define different subsets of $\mathbf{Z}^d$, then $d_\beta(p, q) > 0$.

Proof. The first claim follows directly from the previous lemma: $\mu_\beta(p \vee q) - \mu_\beta(p \wedge q) = \mu_\beta(p\triangle q)$, so dividing by $\mu_\beta(p \vee q)$ gives the claimed representation.

For positivity: if p and q define different sets, then their symmetric difference contains at least one lattice point $\mathbf{a}$. Since every lattice point has strictly positive weight, a single such contribution already implies $\mu_\beta(p\triangle q) > 0$, and hence $d_\beta(p, q) > 0$. □

Theorem 3 (Metricity of the Presburger distance)**.** *Fix $\beta > 0$. Then the function d_β defines a metric on the family of Presburger-definable subsets of $\mathbf{Z}^d$.*

Proof. The distance d_β is the weighted Jaccard distance induced by the finite positive measure μ_β on $\mathbb{Z}^d$. Hence non-negativity, symmetry, and the triangle inequality follow from the standard measure-theoretic Jaccard metric argument.[1] The identity axiom also holds because every lattice point has strictly positive weight: $\mu_\beta(A\triangle B) = 0$ implies $A = B$. Therefore d_β is a metric on Presburger-definable subsets of $\mathbb{Z}^d$. □

4 Presburger Formula Translation Experiments

This section evaluates GRPO fine-tuning for translating natural-language Presburger arithmetic problems into formal formulas. We follow the GRPO-style reinforcement-learning setup introduced in recent mathematical reasoning work on open language models (Shao et al., 2024). All runs use `Qwen/Qwen3-4B` (Yang et al., 2025) and a fixed split: `data/train.jsonl` for training, `data/dev.jsonl` for validation during SFT, and `data/test.jsonl` for final evaluation. The test set contains 2,000 examples. Code for the experiments is available at https://github.com/tani/presburger-distance-naloma2026.

[1] The standard argument uses $A\triangle C \subseteq (A\triangle B) \cup (B\triangle C)$ together with the usual normalization by union mass; equivalently, this is the classical Jaccard metric proof for finite measures.

Parameter	Value
base model	`Qwen/Qwen3-4B`
SFT epochs	1
SFT max steps	120
SFT batch size	32
SFT gradient accumulation	2
SFT learning rate	`1.5e-5`
max sequence length	512
SFT save steps	20
SFT logging steps	10

Table 1: Shared SFT configuration.

Run	Reward	Steps	Batch	Gens
`string_similarity`	sim-only	100	32	16
`presburger_distance`	dist-pen.	100	32	16

Table 2: GRPO configurations. Both runs use learning rate `1.0e-5`.

4.1 Experimental Setup

All runs first perform supervised fine-tuning (SFT), following the now-standard instruction-tuning stage used before preference optimization and reinforcement-learning-based alignment, and then GRPO. The shared SFT configuration is shown in Table 1. We compare two GRPO reward functions under the same training budget: `string_similarity` with `similarity_only` and `presburger_distance` with `distance_penalized`. Both GRPO runs use 100 steps, batch size 32, 16 generations per prompt, learning rate `1.0e-5`, `temperature = 0.8`, `max_completion_length = 128`, `distance_beta = 1.0`, and vLLM inference.

The full pipeline is straightforward: we first train an SFT model, merge the resulting LoRA adapter into the base model, run GRPO from that SFT-initialized checkpoint, and then evaluate the base, SFT, and GRPO models on the same test set. For each output, we compute parsability, exact match, equivalence when available, string similarity, Presburger distance, and adjusted distance.

4.2 Dataset Format and Pipeline

Each dataset item contains a natural-language arithmetic problem and a target Presburger formula. The model is trained to map the `problem` field to the `formula` field. A typical JSONL entry has the form shown below.

```
{"uuid7": "...",
 "problem": "...",
 "formula": "..."}
```

The training data are synthetic. We first generate logical formulas and propositions by a random-walk-like procedure over tree-structured formula templates, including atomic linear constraints, Boolean connectives, and quantifier prefixes. We then ask a generative language model to produce the corresponding natural-language mathematical problem for each target formula. The prompt used for this problem-generation step is included in `src/dataset.py` in the anonymous repository: `https://anonymous.4open.science/r/presburger-distance-1348/src/dataset.py`.

Starting from the shared train/dev/test split, we first perform SFT, merge the adapter into the base model, run GRPO with one of the two reward definitions, and then evaluate the base, SFT, and GRPO models on the same held-out test set. This pipeline separates format learning, reward-based optimization, and semantic evaluation.

Illustrative training examples. The task itself ranges from direct linear inequalities to quantified formulas. For example, a simple atomic case maps the sentence "the sum $8x + 6z$ is less than -2" to the target formula `8 x + 6 z < -2`. A logical-equivalence case maps "true exactly when" to `<==>`, for example `[-5 x + 8 z = 8 <==> -6 x + 9 y + 8 z < 6]`. A quantified case requires preserving a prefix such as `(A x)(E y)` in addition to the bracketed matrix formula. These examples already suggest why exact string accuracy, syntactic validity, and semantic distance capture different aspects of model behavior.

4.3 Reward and Evaluation Metrics

The `similarity_only` reward is the normalized string similarity between the generated formula g and the reference formula r:

$$R_{\text{sim}}(g, r) = \text{SeqMatch}(g, r).$$

The `distance_penalized` reward gives invalid outputs a reward of -1.0 and otherwise combines semantic distance with string similarity:

$$\begin{aligned} R_{\text{dist}}(g, r) = 0.5 \cdot (1 - D(g, r)) \\ + 0.5 \cdot \text{SeqMatch}(g, r). \end{aligned}$$

If distance computation times out, the distance-derived component is set to 0.0.

We use a mixed reward rather than a distance-only reward because the semantic distance component is informative only after the output is parsable and the distance computation succeeds. In early

GRPO stages, a pure-distance reward would assign little useful signal to invalid strings. The string-similarity term therefore acts as a syntactic shaping signal, while the distance term adds semantic pressure once formal outputs become well formed. A pure-distance ablation remains an important direction for future work.

The Presburger distance is

$$D(p, q) = 1 - \frac{\mu(p \wedge q)}{\mu(p \vee q)}.$$

Lower values are better. Because raw mean distance is available only for parsable outputs whose distance can be computed, we also report the adjusted distance

$$D_{\text{adj}}(g, r) = \begin{cases} D(g, r) & \text{if computable,} \\ 1.0 & \text{otherwise.} \end{cases}$$

This metric prevents models with many invalid outputs from appearing artificially strong.

4.4 Main Results

Table 3 shows that semantic feedback changes GRPO behavior: the strongest overall result is obtained by the distance-based GRPO run `presburger_distance`, which achieves the best adjusted distance (0.1667) and the highest parsability (98.55%). Its raw distance is worse than that of `string_similarity`, but this is misleading because `string_similarity` leaves many outputs unparsable and therefore excludes them from the raw-distance average.

Relative to SFT, GRPO with `similarity_only` substantially degrades performance: parsability drops from 87.70% to 76.50%, exact/equivalent accuracy drops from 68.45% to 47.90%, and adjusted distance worsens from 0.1778 to 0.2953. In contrast, GRPO with `distance_penalized` improves parsability from 87.45% to 98.55% and improves adjusted distance from 0.1821 to 0.1667, although it still reduces exact/equivalent accuracy from 67.30% to 54.05%. This shows that the distance-based reward mainly improves formal validity rather than exact reproduction of the reference string.

4.5 Direct Comparison and Error Analysis

A direct comparison over the same 2,000 test cases shows where this feedback takes effect. The `presburger_distance` model produces 191 exact outputs that `string_similarity` misses, whereas `string_similarity` uniquely gets only 68 exact. Likewise, `presburger_distance` is uniquely parsable on 448 examples, while `string_similarity` is uniquely parsable on only 7. Under adjusted distance, `presburger_distance` is better on 474 examples, `string_similarity` is better on 62, and 1,464 examples are tied. These results are summarized in Table 4.

The dominant failure mode is quantifier handling, but the two rewards fail differently. Among the 611 quantified references, `string_similarity` often preserves quantifier tokens in invalid bracket syntax: 336 outputs contain fragments such as `[A x]` or `[E y]`, despite high mean string similarity (0.902). By contrast, `presburger_distance` almost eliminates this syntax error, but usually by deleting the prefix: it drops all quantifiers on 597 quantified-reference examples. Thus the distance reward improves parsability but still loses exact formula structure. Even when SFT is exactly correct, `string_similarity` creates 187 invalid bracketed-quantifier outputs, whereas `presburger_distance` creates 309 parsable but quantifier-free outputs.

4.6 Representative Examples

Individual errors make the semantic-feedback effect visible. In one representative case, the SFT model outputs `(3 x - 4 y - 5 z >= 9 \/ 9 x + 5 y - 5 z /= -7)`, which is close to the reference string but unparsable because the grammar expects brackets rather than parentheses. The GRPO model with `distance_penalized` instead produces the exact reference `[3 x - 4 y - 5 z >= 9 \/ 9 x + 5 y - 5 z /= -7]`, so its adjusted distance becomes 0.0 while the SFT output receives the worst-case adjusted distance.

However, the opposite phenomenon also appears. For `(A x)(E y)[-x + 5 z <= 7 <==> 2 y - 3 z = -6]`, `similarity_only` produces the unparsable `[A x][E y][5 z - x <= 7 <==> 2 y - 3 z = -6]`, while `distance_penalized` produces the parsable but quantifier-free `[5 z - x <= 7 <==> 2 y - 3 z = -6]`. These examples show why both parsability and semantic distance are necessary for evaluation.

Presburger arithmetic admits quantifier elimination, so some formulas with different surface quantifier structure may nevertheless induce closely related or even equivalent arithmetic constraints after

Run	Model	Parsable	Exact / Equivalent	String Similarity	Raw Distance	Adjusted Distance
`string_similarity`	Base	9.90%	0.50%	0.3814	0.2115	0.9223
`string_similarity`	SFT	87.70%	68.45%	0.9561	0.0533	0.1778
`string_similarity`	GRPO	76.50%	47.90%	0.9410	0.0740	0.2953
`presburger_distance`	Base	9.90%	0.50%	0.3814	0.2111	0.9223
`presburger_distance`	SFT	87.45%	67.30%	0.9544	0.0555	0.1821
`presburger_distance`	GRPO	98.55%	54.05%	0.9382	0.1462	0.1667

Table 3: Main evaluation results.

Comparison	Count
both exact	890
only `string_similarity` exact	68
only `presburger_distance` exact	191
only `string_similarity` parsable	7
only `presburger_distance` parsable	448
`string_similarity` better by adjusted distance	62
`presburger_distance` better by adjusted distance	474
tied by adjusted distance	1464

Table 4: Head-to-head comparison between the two GRPO rewards.

Pattern	sim	dist
invalid bracketed quantifier after quantified reference	336	0
all quantifiers dropped after quantified reference	158	597
other invalid output after quantified reference	117	14
invalid GRPO output	470	29

Table 5: Most important error patterns for the two GRPO rewards.

elimination. Accordingly, the present metric prioritizes overlap of induced solution sets over exact preservation of the original logical surface form.

4.7 Additional Case Studies

It is also useful to contrast cases in which semantic correctness and syntactic correctness diverge. In one example, the reference is `-9 x + 2 z <= -4`. The `similarity_only` GRPO model outputs `[-9 x + 2 z <= -4]`, which remains parsable and semantically equivalent under the distance computation, but fails exact match because of the added outer brackets. The `distance_penalized` model instead reproduces the exact target string. This illustrates that the two rewards can produce semantically similar outputs while differing in how closely they match the annotation convention.

A second case shows the opposite failure mode. For the reference `[-10 x - 5 y = -5 <==> -9 x - 9 y >= -5]`, the `distance_penalized` model may produce `[-10 x + (-5 y) = -5 <==> -9 x + (-9 y) >= -5]`. Although this looks close to the target as a string, the inserted parenthesized negative terms are rejected by the grammar in the current evaluation pipeline. The example highlights that a reward which strongly discourages invalid outputs can still generate local syntactic forms that are outside the accepted grammar.

Taken together, these examples reinforce a central point of the experiments: the best reward is not simply the one that maximizes local string similarity, but the one that most reliably preserves formal well-formedness while staying semantically close to the target formula.

5 Discussion

5.1 Scope in the NLP Landscape

This work should be read as a controlled case study in semantics-aware formula generation, not as a general solution to semantic similarity for formal mathematical language. Presburger arithmetic is restricted to linear integer arithmetic with addition, order, Boolean connectives, and first-order quantifiers. It excludes multiplication between variables, polynomial equations, integrals, gradients, differential operators, and higher-order or dependent type-theoretic constructions.

This restriction is also what makes the proposed distance computationally meaningful in the present setting. Quantifier elimination, polyhedral decomposition, and lattice-point generating functions provide a principled route from formulas to weighted model sets. For richer formal languages, comparable semantic distances would require different computational tools, and in many cases no similarly tractable pipeline is available. Thus, the present metric is best understood as a foundational step toward semantics-aware rewards, rather than as a universal metric for mathematical NLP.

5.2 Synthetic Data and Pre-training Dependencies

A primary criticism of our evaluation might focus on its reliance on a synthetically generated dataset rather than noisy, real-world human text. However, this choice reflects a deliberate methodological decoupling rather than an oversight. The core objective of this study is not to benchmark the well-documented linguistic robustness of LLMs against typos, colloquialisms, or stylistic variants—confounding factors that are better evaluated in orthogonal studies. Instead, our goal is to rigorously isolate and quantify a single, precise causal relationship: whether mapping discrete arithmetic propositions into a continuous space via a geometric distance reward inherently guides policy optimization more effectively.

Introducing open-domain human noise at this foundational stage would introduce unnecessary variance, blurring the semantic signal of the reward and obscuring whether our geometric metric is genuinely driving the observed performance gains. Furthermore, given that Presburger arithmetic is restricted to linear transformations, solving open-domain offline natural language tasks would not immediately yield an end-to-end industrial application. Therefore, we maintain a cautious and principled stance, restricting our benchmark to controlled environments to ensure the precision and clarity of our theoretical contributions.

We also do not claim that the observed performance is independent of pre-training. The experiments start from a pretrained language model that already has substantial knowledge of mathematical notation and instruction-following conventions. Our claim is therefore narrower: given such a pretrained model, a semantic distance component provides a more effective optimization signal than string similarity alone. Disentangling the contribution of pre-training would require additional ablations, such as randomly initialized models, in-domain language-model pre-training, held-out vocabulary splits, or held-out lattice-region splits. We leave these evaluations to future work.

5.3 Computational Complexity and Scalability

Overall, GRPO responds to the semantic signal in the reward. A string-similarity-only reward does not reliably preserve syntactic validity and performs poorly under adjusted distance. The distance-based reward penalizes invalid formulas and shifts outputs toward the formal grammar, yielding much higher parsability and the best adjusted semantic distance.

At the same time, neither GRPO reward surpasses the SFT baseline on exact string accuracy. The remaining bottleneck is structural fidelity, especially quantifier preservation. Because the current distance computation is most useful for quantifier-free formulas, the reward does not strongly penalize missing prefixes such as `(A ...)` and `(E ...)`.

There is also an important computational trade-off behind the semantic evaluation itself. Computing Presburger distances through QEPCAD and counting lattice points with LattE is computationally expensive from a computer-science perspective, and the running time can grow substantially as the number of variables increases or the coefficients become larger, consistent with the broader difficulty of integer-programming-style reasoning and related optimization problems (Papadimitriou, 1981). In that sense, the present benchmark benefits from the fact that the test formulas are comparatively simple. This makes the distance-based pipeline practical in our experiments, but it also means that scalability to harder instances remains an important limitation and a topic for future work.

6 Limitations

This study has several limitations typical of semantics-aware evaluation in NLP. First, the benchmark formulas are comparatively simple, so the current results do not establish that the proposed distance remains practical for larger formulas with more variables, larger coefficients, or more deeply nested quantifier structure.

Second, our semantic pipeline depends on quantifier elimination and lattice-point computation, which are substantially more expensive than string-based metrics and may limit scalability during training.

Third, although the distance-aware reward improves parsability and adjusted semantic quality, it still does not adequately preserve quantifier prefixes. Thus, semantic similarity alone is not sufficient to guarantee full structural fidelity.

Fourth, the experiments focus on a single formal translation setting, and it remains to be tested how well the same reward design transfers to other symbolic generation tasks in NLP.

Fifth, the experiments do not isolate the contribution of the pretrained base model. The results therefore show the effect of reward design under a pretrained-model regime, not the intrinsic learnability of the task from scratch.

Finally, we did not include a pure-distance GRPO reward. The present comparison evaluates a mixed syntactic-semantic reward against a syntactic baseline, leaving a pure semantic-reward ablation for future work.

7 Conclusion

Among the two GRPO reward functions, `distance_penalized` is preferable: its errors suggest that the semantic reward is being optimized, moving outputs away from invalid string-like forms toward parsable formulas. It substantially improves parsability, improves exact match relative to `similarity_only`, and achieves the best adjusted semantic distance. Nevertheless, SFT remains stronger for exact formula reproduction, so the next step is to design rewards that preserve quantifier structure as well as formal validity.

Acknowledgments

This work was supported by JSPS KAKENHI Grant-in-Aid for JSPS Fellows (Grant Number JP26KJ0414), JST CREST (Grant Number JPMJCR2565), and the JST BOOST Program (Grant Number JPMJBY24H5), Japan.

We used ChatGPT for limited coding assistance, LaTeX snippet drafting, and language-editing support. We also used it to help check the wording of mathematical definitions, which were verified by the authors against standard formulations. The authors take full responsibility for all code, mathematical definitions, experimental design, analyses, scientific claims, and conclusions.

References

D. C. Cooper. 1972. Theorem proving in arithmetic without multiplication. In Bernard Meltzer and Donald Michie, editors, *Machine Intelligence 7*, pages 91–99. Edinburgh University Press.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43.

Jonathan Herzig, Peter Shaw, Ming-Wei Chang, Kelvin Guu, Panupong Pasupat, and Yuan Zhang. 2021. Span-based semantic parsing for compositional generalization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 908–921.

Paul Jaccard. 1901. Étude comparative de la distribution florale dans une portion des alpes et du jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37(142):547–579.

Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 590–599.

Jesús A. De Loera, Raymond Hemmecke, Jörg Tauzer, and Ruriko Yoshida. 2004. Effective lattice point counting in rational convex polytopes. *Journal of Symbolic Computation*, 38(4):1273–1302.

Christos H. Papadimitriou. 1981. On the complexity of integer programming. *Journal of the ACM*, 28(4):765–768.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Kevin Woods. 2014. The unreasonable ubiquitousness of quasi-polynomials. *The Electronic Journal of Combinatorics*, 21(1):Paper 1.44.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 658–666.