Maximum Flow Retrieval Algorithm

To solve the maximum information flow problem defined over the semantic graph $\mathcal{H}$, we adopt a standard network flow algorithm, such as Edmonds-Karp. The core of this algorithm is to find a set of paths from the source entity e_q to the sink entity e_a that maximizes the total information flow.

We use the symbol ζ to denote an arbitrary triple in the graph $\mathcal{H}$. Each triple ζ has a head entity $h(\zeta)$, a tail entity $t(\zeta)$, and a non-negative capacity (weight) $w(\zeta)$. The information flow, denoted by $f(\zeta)$, represents the amount of information actually transmitted along the triple ζ.

The problem is formalized as an optimization problem subject to the following constraints:

1. **Capacity Constraint:** The flow through any given triple cannot exceed its capacity.

$$f(\zeta) \leq w(\zeta), \quad \forall \zeta \in \mathcal{H} \tag{1}$$

2. **Flow Conservation:** For any entity v that is not the source e_q or the sink e_a, the total incoming flow must equal the total outgoing flow.

$$\sum_{\zeta | t(\zeta)=v} f(\zeta) = \sum_{\zeta | h(\zeta)=v} f(\zeta), \quad \forall v \in \mathcal{E} \setminus \{e_q, e_a\} \tag{2}$$

3. **Skew Symmetry:** The flow in the reverse direction is defined as the negative of the forward flow.

$$f(\zeta) = -f(\zeta^{-1}) \tag{3}$$

The objective of the algorithm is to maximize the net flow $|f|$ out of the source entity e_q:

$$\text{Maximize } |f| = \sum_{\zeta | h(\zeta)=e_q} f(\zeta) - \sum_{\zeta | t(\zeta)=e_q} f(\zeta) \tag{4}$$

The algorithm operates by iteratively finding **augmenting paths** in a **residual network** $\mathcal{H}_f$. The residual network is defined as follows:

1. Residual Capacity: For a **forward triple** $\zeta \in \mathcal{H}$, its residual capacity is defined as:

$$w_f(\zeta) = w(\zeta) - f(\zeta) \tag{5}$$

For a **backward triple** ζ^{-1} (where $\zeta \in \mathcal{H}$), its residual capacity is the current forward flow:

$$w_f(\zeta^{-1}) = f(\zeta) \tag{6}$$

2. Residual Network Construction: The residual network $\mathcal{H}_f$ consists of all triples (both forward and backward) with a residual capacity greater than zero:

$$\mathcal{H}_f = \{\zeta' \mid w_f(\zeta') > 0\} \tag{7}$$

where ζ' can be either a triple ζ from the original graph or its reverse ζ^{-1}.

The iterative process of the algorithm is as follows:

Repeat:

1. In the residual network $\mathcal{H}_f$, find an augmenting path P from e_q to e_a using a search algorithm like Breadth-First Search (BFS).
2. **If no such path exists**, the algorithm terminates. The current flow f is the maximum flow.
3. **If a path P is found**, calculate its bottleneck capacity (the amount of flow that can be augmented):

$$w_f(P) = \min_{\zeta' \in P} w_f(\zeta') \tag{8}$$

4. Augment the flow by $w_f(P)$ along the path P. For each triple ζ' on the path P:
   - If ζ' is a **forward triple** ($\zeta' = \zeta$ and $\zeta \in \mathcal{H}$), update the flow:
   $$f(\zeta) \leftarrow f(\zeta) + w_f(P) \tag{9}$$
   - If ζ' is a **backward triple** ($\zeta' = \zeta^{-1}$ and $\zeta \in \mathcal{H}$), update the flow:
   $$f(\zeta) \leftarrow f(\zeta) - w_f(P) \tag{10}$$

Until no augmenting path can be found.

Upon termination, all feasible information-carrying channels are saturated. According to the max-flow min-cut theorem, this process yields a globally optimal, information-saturated subgraph $\mathcal{H}_m$ that cannot accommodate any additional semantically relevant information from e_q without violating the capacity constraints.

Algorithm 1: Lower-Upper Bound Path Selection

Require: Semantic Graph $\mathcal{H}_q$, density threshold η
Ensure: Optimal subgraph paths $\mathcal{P}$

$\mathcal{P}_{\text{dense}} \leftarrow \{P \subseteq \mathcal{H}_q \mid D(P) \geq \eta\}$
for each $P \in \mathcal{P}_{\text{dense}}$ **do**
  Compute $\phi(P) = \min_{\zeta \in P} w(\zeta)$
  Compute $\psi(P) = \sum_{\zeta \in P} w(\zeta)$
end for
$\phi^\star \leftarrow \max_{P \in \mathcal{P}_{\text{dense}}} \phi(P)$
$\mathcal{P}_{\text{bottleneck}} \leftarrow \{P \in \mathcal{P}_{\text{dense}} \mid \phi(P) = \phi^\star\}$
$\mathcal{P}_{\text{lower}} \leftarrow \arg\max_{P \in \mathcal{P}_{\text{bottleneck}}} \psi(P)$
$D^\star \leftarrow \max_{P \in \mathcal{P}_{\text{dense}}} D(P)$
$\mathcal{P}_{\text{density}} \leftarrow \{P \in \mathcal{P}_{\text{dense}} \mid D(P) = D^\star\}$
$\mathcal{P}_{\text{upper}} \leftarrow \arg\max_{P \in \mathcal{P}_{\text{density}}} \psi(P)$
$\mathcal{P} \leftarrow \mathcal{P}_{\text{lower}} \cup \mathcal{P}_{\text{upper}}$
return $\mathcal{P}$

Error Propagation and Path Selection in Knowledge Graphs

This appendix provides a detailed mathematical formulation and analysis of error propagation in multi-hop reasoning over knowledge graphs, including the perturbation model, accumulated error bounds, the motivation for the widest path strategy, and supporting theoretical results.

Perturbation Model and Path Error Formulation

Definition 1 (Perturbation Mapping Model) *Let a multi-hop path be denoted as $p = (e_1, e_2, \ldots, e_n)$, where each edge e_i corresponds to a relation r_i linking entities. We model the semantic transformation along each edge as a latent function $\phi_i : \mathbb{R}^d \to \mathbb{R}^d$. We assume there exists a true,*

ideal transformation Φ_i, and our practical approximation ϕ_i is a perturbed version of it:

$$\phi_i(x) = \Phi_i(x) + \varepsilon_i(x) \tag{11}$$

where $\varepsilon_i(x) : \mathbb{R}^d \to \mathbb{R}^d$ denotes the step-wise error.

Definition 2 (Path Function and Path Error) *We define the full perturbed path function $f_p(x)$ and the ideal path function $F_p(x)$ as the composition of their respective step-wise functions:*

$$f_p(x) := \phi_n \circ \cdots \circ \phi_1(x), \quad F_p(x) := \Phi_n \circ \cdots \circ \Phi_1(x) \tag{12}$$

The total path error, $\delta_p(x)$, is the divergence between these two functions:

$$\delta_p(x) := f_p(x) - F_p(x) \tag{13}$$

Remark 1 *The validity of using composite perturbed mappings $\phi_i = \Phi_i + \varepsilon_i$ is formally guaranteed by Lemma 2, which ensures that the total composite function remains a good approximation to the ideal composite $\Phi_n \circ \cdots \circ \Phi_1$, under mild continuity and compactness conditions.*

Error Accumulation Bound

Lemma 1 (Accumulated Path Error Upper Bound) *Assume each ideal transformation Φ_i is differentiable and satisfies $\|\nabla \Phi_i(x)\| \leq \alpha_i < 1$, and each perturbation term is bounded by $\|\varepsilon_i(x)\| \leq \epsilon_i$. The total path error is then upper-bounded by:*

$$\|f_p(x) - F_p(x)\| \leq \sum_{i=1}^{n} \left(\prod_{j=i+1}^{n} \alpha_j \right) \cdot \epsilon_i \tag{14}$$

Proof 1 *We expand the composite mapping:*

$$f_p(x) = (\Phi_n + \varepsilon_n) \circ \cdots \circ (\Phi_1 + \varepsilon_1)(x) \tag{15}$$

Define intermediate variables:

$$x_0 := x, \quad x_i := \Phi_i(x_{i-1}) \quad \textit{for } i = 1, \ldots, n \tag{16}$$

Using first-order Taylor approximation, we obtain:

$$\delta_p(x) \approx \sum_{i=1}^{n} \left(J_n(x_n) \cdots J_{i+1}(x_{i+1}) \right) \varepsilon_i(x_i) \tag{17}$$

where $J_i = \nabla \Phi_i(x_i)$. Taking norms:

$$\|\delta_p(x)\| \leq \sum_{i=1}^{n} \left(\prod_{j=i+1}^{n} \|J_j(x_j)\| \right) \cdot \|\varepsilon_i(x_i)\| \tag{18}$$

By assumption, $\|J_j(x_j)\| \leq \alpha_j$, $\|\varepsilon_i(x_i)\| \leq \epsilon_i$, yielding Eq. (14)*.*

Simplified Bound under Uniform Conditions

Assuming uniform bounds $\epsilon_i \leq \epsilon_{\min}$, $\alpha_i \leq \alpha_{\max} < 1$, the error bound in Eq. (14) simplifies to:

$$\|\delta_p(x)\| \leq \epsilon_{\min} \cdot \sum_{i=1}^{n} \alpha_{\max}^{n-i} = \epsilon_{\min} \cdot \frac{1 - \alpha_{\max}^{n}}{1 - \alpha_{\max}} \tag{19}$$

Motivation for Widest Path Strategy

The *widest path strategy* aims to minimize global error by ensuring each transformation step is of high quality, i.e., all $\epsilon_i \leq \epsilon_{\min}$. This approach avoids steps with high approximation variance and ensures local transformations remain within a reliable bound.

This idea is analogous to layer-wise control in residual networks, where limiting per-layer deviation reduces the risk of vanishing/exploding error across a deep composite structure.

Supporting Lemma: Uniform Approximation

Lemma 2 *Let map $T = F_1 \circ \cdots \circ F_n$ be a composition of n continuous functions F_i defined on an open domain D_i, and let $\mathcal{F}$ be a continuous function class that can uniformly approximate each F_i on any compact domain $\mathcal{K}_i \subset D_i$. Then, for any compact domain $\mathcal{K} \subset D_1$ and $\varepsilon > 0$, there are n functions $\tilde{F}_1, \ldots, \tilde{F}_n \in \mathcal{F}$ such that*

$$\|T(x) - \tilde{F}_n \circ \cdots \circ \tilde{F}_1(x)\| \leq \varepsilon, \quad \forall x \in \mathcal{K} \tag{20}$$

Proof 2 (Proof for $n = 2$) *It is enough to prove the case of $n = 2$. (The case of $n > 2$ can be proven by induction, since $T = F_n \circ T_{n-1}$, with $T_{n-1} = F_{n-1} \circ \cdots \circ F_1$.) According to the definition, we have $F_1(D_1) \subset D_2$. Since D_2 is open and $F_1(\mathcal{K})$ is compact, we can choose a compact set $\mathcal{K}_2 \subset D_2$ such that*

$$\mathcal{K}_2 \supset \{F_1(x) + \delta_0 y : x \in \mathcal{K}, \|y\| < 1\} \tag{21}$$

for some $\delta_0 > 0$ that is sufficiently small.

According to the continuity of F_2, there is a $\delta \in (0, \delta_0)$ such that

$$\|F_2(y) - F_2(y')\| \leq \varepsilon/2, \quad \forall y, y' \in \mathcal{K}_2,\ \|y - y'\| \leq \delta \tag{22}$$

The approximation property of $\mathcal{F}$ allows us to choose $\tilde{F}_1, \tilde{F}_2 \in \mathcal{F}$ such that

$$\|\tilde{F}_1(x) - F_1(x)\| \leq \delta < \delta_0, \quad \forall x \in \mathcal{K} \tag{23}$$

$$\|\tilde{F}_2(y) - F_2(y)\| \leq \varepsilon/2, \quad \forall y \in \mathcal{K}_2 \tag{24}$$

As a consequence, for any $x \in \mathcal{K}$, we have $F_1(x), \tilde{F}_1(x) \in \mathcal{K}_2$, and

$$\begin{aligned} \|F_2 \circ F_1(x) - \tilde{F}_2 \circ \tilde{F}_1(x)\| &\leq \|F_2 \circ F_1(x) - F_2 \circ \tilde{F}_1(x)\| \\ &+ \|F_2 \circ \tilde{F}_1(x) - \tilde{F}_2 \circ \tilde{F}_1(x)\| \leq \varepsilon/2 + \varepsilon/2 = \varepsilon \end{aligned} \tag{25}$$

Maximize the posterior probability of the inference path

Premise

Definition 3 (Prior Probability of a Path $\Pr(P)$) *In the absence of any query information q, the existence of a path itself is governed by a prior probability. We assume that the prior probability of a path is inversely proportional to its length $|P| = m$. For example, we can adopt an exponential decay model:*

$$\Pr(P) \propto e^{-\lambda |P|} \tag{26}$$

where λ is a constant that controls the penalty for path length. This bias encourages the selection of shorter, more concise paths.

Definition 4 (Likelihood of a Query Given a Path $\Pr(q \mid P)$) *If a reasoning path P is correct, it should be highly relevant to the query q. This is where the triple weight $w(\zeta)$ plays a role. We interpret $w(\zeta)$ as the similarity between the triple ζ and the query q.*

Assuming each step ζ_i in the path contributes independently to the query, the total likelihood is the product over all steps:

$$\Pr(q \mid P) = \prod_{i=1}^{m} \Pr(q \mid \zeta_i) \tag{27}$$

For convenience, we use the log-likelihood. Let $w(\zeta_i) = \log \Pr(q \mid \zeta_i)$, then the path likelihood becomes:

$$\log \Pr(q \mid P) = \sum_{i=1}^{m} w(\zeta_i) = \psi(P) \tag{28}$$

Definition 5 (Posterior Probability of a Path $\Pr(P \mid q)$) *Based on Bayes' theorem, the posterior probability of path P given query q is:*

$$\Pr(P \mid q) = \frac{\Pr(q \mid P)\Pr(P)}{\Pr(q)} \tag{29}$$

Since $\Pr(q)$ is constant for all paths, maximizing $\Pr(P \mid q)$ is equivalent to maximizing the numerator:

$$\begin{aligned} \log \Pr(P \mid q) &\propto \log\left(\Pr(q \mid P)\Pr(P)\right) \\ &= \log \Pr(q \mid P) + \log \Pr(P) \end{aligned} \tag{30}$$

Substituting in Definitions 1 and 2, we get:

$$\arg\max_P \log \Pr(P \mid q) = \arg\max_P \left(\psi(P) - \lambda|P|\right) \tag{31}$$

Theorem 1 (Connection to Maximum Path Density) *We aim to maximize the objective $\psi(P) - \lambda|P|$. By dividing by the path length $|P|$, we reformulate the objective as:*

$$\frac{f(P)}{|P|} = \frac{\psi(P)}{|P|} - \lambda = D(P) - \lambda \tag{32}$$

Since λ is a constant, maximizing $D(P) - \lambda$ is equivalent to maximizing $D(P)$.

Lemma 3 (Probabilistic Interpretation) ***(MaxDensity-InfoPath)*** *Assume the path prior $\Pr(P)$ decays exponentially with path length $|P|$, and the log-likelihood $\log \Pr(q \mid P)$ equals the total information $\psi(P)$. Then, maximizing the path density $D(P)$ is equivalent to selecting the path with the highest posterior probability:*

$$\arg\max_P D(P) \Longleftrightarrow \arg\max_P \left(\log \Pr(q \mid P) + \log \Pr(P)\right) \tag{33}$$

Definition of Positional Attention Weight

Given context sequence:

$$C = [x_1, x_2, \ldots, x_n], \tag{34}$$

suppose the key information lies at position r, define positional attention weight $\theta(r)$:

$$\begin{aligned} \theta(r) = \alpha \cdot \Big[&\exp(-\lambda \cdot |r-1|) \\ &+ \exp(-\lambda \cdot |r-n|)\Big], \quad r \in \{1, 2, \ldots, n\} \end{aligned} \tag{35}$$

where α is the normalization constant ensuring:

$$\max_r \theta(r) = 1. \tag{36}$$

Thus, we have:

$$\alpha = \frac{1}{\exp(0) + \exp(-\lambda \cdot (n-1))} = \frac{1}{1 + \exp(-\lambda(n-1))}. \tag{37}$$

Content Matching Score and Probability

Define content matching score as $f(x_r)$, and the probability of correct answer a conditioned on C:

$$P(a \mid C) \propto \theta(r) \cdot f(x_r). \tag{38}$$

Expected Probability Derivation

Assuming random distribution of key information, position r is uniformly distributed:

$$P(r) = \frac{1}{n}, \quad \forall r \in \{1, 2, \ldots, n\}. \tag{39}$$

Then the expected probability is:

$$\mathbb{E}[P(a \mid C)] = \sum_{r=1}^{n} P(a \mid C, r)P(r) = \frac{1}{n}\sum_{r=1}^{n} \theta(r) f(x_r). \tag{40}$$

Simplification under Uniform Matching Score

If the content matching score is approximately constant, i.e.:

$$f(x_r) \approx \bar{f}, \quad \forall r, \tag{41}$$

Then:

$$\mathbb{E}[P(a \mid C)] \approx \frac{\bar{f}}{n}\sum_{r=1}^{n} \theta(r). \tag{42}$$

The summation can be explicitly expanded:

$$\sum_{r=1}^{n} \theta(r) = \alpha \sum_{r=1}^{n} \left[\exp(-\lambda|r-1|) + \exp(-\lambda|r-n|)\right]. \tag{43}$$

Separate the summation:

$$\sum_{r=1}^{n} \theta(r) = \alpha \left[\sum_{r=1}^{n} \exp(-\lambda(r-1)) + \sum_{r=1}^{n} \exp(-\lambda(n-r))\right]. \tag{44}$$

Evaluate these geometric series:

$$\sum_{r=1}^{n} \exp(-\lambda(r-1)) = \frac{1-\exp(-\lambda n)}{1-\exp(-\lambda)}, \tag{45}$$

$$\sum_{r=1}^{n} \exp(-\lambda(n-r)) = \frac{1-\exp(-\lambda n)}{1-\exp(-\lambda)}. \tag{46}$$

Thus:

$$\sum_{r=1}^{n} \theta(r) = \alpha \cdot 2 \cdot \frac{1-\exp(-\lambda n)}{1-\exp(-\lambda)}. \tag{47}$$

Substituting α back, we have:

$$\sum_{r=1}^{n} \theta(r) = \frac{2}{1+\exp(-\lambda(n-1))} \cdot \frac{1-\exp(-\lambda n)}{1-\exp(-\lambda)}. \tag{48}$$

Based on the substituted expression:

$$\mathbb{E}[P(a \mid C)] \approx \frac{\bar{f}}{n} \cdot \frac{2}{1+\exp(-\lambda(n-1))} \cdot \frac{1-\exp(-\lambda n)}{1-\exp(-\lambda)} \tag{49}$$

D.5: Effect of Parameter λ on Positional Sensitivity

Limit Analysis ($\lambda \to 0^+$) When $\lambda \to 0^+$, we have:

- $\exp(-\lambda) \approx 1-\lambda$, so:

$$1-\exp(-\lambda) \approx \lambda \tag{50}$$

- Similarly:

$$1-\exp(-\lambda n) \approx \lambda n \tag{51}$$

Substituting into the expression, we get the approximation:

$$\mathbb{E}[P(a \mid C)] \approx \frac{\bar{f}}{n} \cdot \frac{2}{1+1} \cdot \frac{\lambda n}{\lambda} = \bar{f} \tag{52}$$

Conclusion: When positional sensitivity is extremely low ($\lambda \to 0$), the positional attention tends to be uniformly distributed, thus the average prediction probability reaches its maximum, and the model exhibits almost no positional bias.

Limit Analysis ($\lambda \to +\infty$) When $\lambda \to +\infty$, exponential terms decay rapidly, and we have:

- For $r=1$ or $r=n$, $\theta(r) \approx 1$
- For all other positions, $\theta(r) \approx 0$

So the total summation approximates:

$$\sum_{r=1}^{n} \theta(r) \approx 2 \tag{53}$$

Then:

$$\mathbb{E}[P(a \mid C)] \approx \frac{\bar{f}}{n} \cdot 2 = \frac{2\bar{f}}{n} \tag{54}$$

Conclusion: When positional sensitivity is extremely high ($\lambda \to +\infty$), the expected prediction probability significantly decreases, and it declines rapidly as the sequence length n increases. This shows that only boundary positions can be effectively captured under such bias.

Effect of Sequence Length n on Expected Probability For a fixed λ, we analyze the effect of varying n:

- When n is small, the exponential term $\exp(-\lambda(n-1))$ approaches 1, indicating weak positional bias.
- As n increases, we have $1+\exp(-\lambda(n-1)) \approx 1$, and $1-\exp(-\lambda n) \approx 1$, so:

$$\mathbb{E}[P(a \mid C)] \approx \frac{2\bar{f}}{n(1-\exp(-\lambda))}, \quad n \gg 1 \tag{55}$$

Conclusion: As sequence length increases, the expected prediction probability significantly declines, especially when positional sensitivity is relatively high.

Effect of Positional Bias on Probability Distribution (Minimum Probability Position) From the definition of $\theta(r)$, we have:

- $\theta(r)$ reaches the maximum at edge positions ($r=1$ or $r=n$)
- $\theta(r)$ reaches the minimum at the middle position ($r = \frac{n+1}{2}$)

Specifically, for odd n, the minimum position is $r = \frac{n+1}{2}$, and:

$$\theta\left(\frac{n+1}{2}\right) = 2\alpha \exp\left(-\lambda \cdot \frac{n-1}{2}\right) = \frac{2\exp\left(-\lambda \cdot \frac{n-1}{2}\right)}{1+\exp(-\lambda(n-1))} \tag{56}$$

As λ increases, this value tends to 0, and the prediction probability at this position is extremely low.

Conclusion: The model's probability of correctly predicting the answer is minimized when the key information is located at the middle of the input sequence, consistent with the U-shaped positional attention distribution.

Practical Implications and Strategy Choices

From theoretical derivation and analysis, the above formulas and results indicate:

- When key information lies in the middle of the sequence, it is the hardest for the model to capture effectively.
- When designing tasks or training models, we should consider using position-sensitive mechanisms or randomized positional strategies (e.g., DSRS mechanism) to mitigate the negative effects of positional bias.

Final Summary

To summarize, we conclude the following key points:

- When $\lambda \to 0^+$, the model has almost no positional bias and achieves the highest prediction probability.
- When $\lambda \to +\infty$, the model heavily relies on boundary information, and the expected prediction probability sharply drops with sequence length.
- For a fixed λ, the expected probability gradually decreases as sequence length increases.
- Information located in the middle of the sequence is hardest to model due to the minimal attention weight at those positions.

Analysis of Probability Minimization

The positional attention weight $\theta(r)$ has a U-shaped distribution, achieving maximum values at positions 1 and n, and minimum in the middle positions. Specifically, for midpoint $r = \frac{n+1}{2}$ when n is odd (similar logic applies for even n), $\theta(r)$ achieves minimum due to symmetric exponential decay from both ends.

This directly implies:

$$\min_r P(a \mid C, r) \quad \text{at} \quad r = \frac{n+1}{2}. \tag{57}$$

Thus, from a theoretical expectation viewpoint, the expected probability $\mathbb{E}[P(a \mid C)]$ is minimized when key information is more likely located in the middle positions of the input sequence.

Final Conclusion

Hence, the detailed derivation mathematically validates the claim that the positional bias inherent in attention mechanisms leads to lower prediction accuracy when the key information is located at the middle positions of the context sequence.

Explanation via Transformer Attention

In Transformer models, attention is defined as:

$$\text{Attention}(i, j) = \frac{\exp(q_i \cdot k_j/\sqrt{d})}{\sum_j \exp(q_i \cdot k_j/\sqrt{d})}. \tag{58}$$

Tokens in the middle suffer from softmax competition and thus receive lower attention weights when q_i is from the head or tail. Even when the query vector q_i corresponds to a middle token, it tends to assign moderate attention to both ends of the sequence, leading to a more uniform and dispersed attention distribution. Due to the softmax normalization, this dispersion reduces the peak attention values, making it harder for middle positions to dominate. In contrast, when q_i is near the edge, the attention naturally concentrates on nearby tokens, leading to stronger localized weights and a higher semantic impact.

Derivation of the Differentiable Ranking Loss

This appendix details the derivation of our Differentiable Subgraph Ranking Supervision (DSRS) loss, starting from the standard Kendall's Tau correlation and arriving at our final objective function.

The Kendall's Tau Coefficient

Definition 6 (Concordant and Discordant Pairs) *Given two rankings (score vectors) $\mathbf{x}$ and $\mathbf{y}$ of length n, for any pair of distinct indices (i, j), the pair is:*

- ***Concordant*** *if the relative order is the same, i.e., $(x_i - x_j)(y_i - y_j) > 0$.*
- ***Discordant*** *if the relative order is opposite, i.e., $(x_i - x_j)(y_i - y_j) < 0$.*

Definition 7 (Kendall's Tau) *The Kendall's Tau rank correlation coefficient $\tau(\mathbf{x}, \mathbf{y})$ is defined as the difference between the number of concordant pairs (N_{con}) and discordant pairs (N_{dis}), normalized by the total number of pairs ($N_0 = \frac{n(n-1)}{2}$):*

$$\tau(\mathbf{x}, \mathbf{y}) = \frac{N_{con} - N_{dis}}{N_0} \tag{59}$$

As noted in the main text, τ is non-differentiable due to its reliance on discrete counts.

Differentiable Approximation of Kendall's Tau

To enable gradient-based optimization, we introduce a continuous and differentiable surrogate for τ.

Definition 8 (Differentiable Indicator Function) *We replace the discrete counting of concordant/discordant pairs with a smooth sigmoid-based function. We define a differentiable indicator $I_\beta(i, j)$ for a pair (i, j) as:*

$$I_\beta(i, j) = \frac{e^{\beta(x_i - x_j)(y_i - y_j)} - 1}{e^{\beta(x_i - x_j)(y_i - y_j)} + 1} \tag{60}$$

where $\beta > 0$ is a temperature hyperparameter.

Lemma 4 (Properties of the Differentiable Indicator) *The differentiable indicator $I_\beta(i, j)$ approximates the discrete indicator function ($+1$ for concordant, -1 for discordant) as $\beta \to +\infty$:*

- *For concordant pairs, where $(x_i - x_j)(y_i - y_j) > 0$:*

$$\lim_{\beta \to +\infty} I_\beta(i, j) = 1 \tag{61}$$

- *For discordant pairs, where $(x_i - x_j)(y_i - y_j) < 0$:*

$$\lim_{\beta \to +\infty} I_\beta(i, j) = -1 \tag{62}$$

Definition 9 (Differentiable Approximate Kendall's Tau) *By summing the differentiable indicators over all pairs, we define the approximate Kendall's Tau $\tilde{\tau}_\beta(\mathbf{x}, \mathbf{y})$ as:*

$$\tilde{\tau}_\beta(\mathbf{x}, \mathbf{y}) = \frac{1}{N_0} \sum_{i<j} I_\beta(i, j) = \frac{1}{N_0} \sum_{i<j} \frac{e^{\beta(x_i - x_j)(y_i - y_j)} - 1}{e^{\beta(x_i - x_j)(y_i - y_j)} + 1} \tag{63}$$

From Lemma 4, it follows directly that:

$$\lim_{\beta \to +\infty} \tilde{\tau}_\beta(\mathbf{x}, \mathbf{y}) = \tau(\mathbf{x}, \mathbf{y}) \tag{64}$$

From Approximate Kendall's Tau to DSRS

Proposition 1 (Sigmoid Equivalence) *The term $(I_\beta(i, j) + 1)/2$, which maps the $[-1, 1]$ range of the indicator to $[0, 1]$, is equivalent to the sigmoid function applied to the pairwise agreement score:*

$$\begin{aligned} \frac{I_\beta(i, j) + 1}{2} &= \frac{1}{2}\left(\frac{e^{\beta(x_i - x_j)(y_i - y_j)} - 1}{e^{\beta(x_i - x_j)(y_i - y_j)} + 1} + 1\right) \\ &= \frac{e^{\beta(x_i - x_j)(y_i - y_j)}}{e^{\beta(x_i - x_j)(y_i - y_j)} + 1} \end{aligned} \tag{65}$$

This term can be interpreted as the "probability of concordance" for a pair (i, j).

Definition 10 (DSRS Function) *To maximize the concordance probability, we can equivalently minimize its complement,* $1 - \sigma(\cdot)$*. Summing this pairwise error over all pairs gives us our final DSRS loss function,* $\mathcal{L}_{ken}$*:*

$$\mathcal{L}_{ken} = \frac{1}{|N|} \sum_{(i,j)\in N} [1 - \sigma(\beta \Delta s_{ij} \Delta q_{ij})] \tag{66}$$

This formulation standardizes the pairwise loss to the $[0, 1]$ *range, where a loss of 0 indicates perfect agreement and a loss of 1 indicates perfect disagreement.*

Experiments

Method	WebQSP		CWQ	
	Macro-F1	Hit	Macro-F1	Hit
UniKGQA	70.2	–	49.0	–
KD-CoT	52.5	68.6	–	55.7
SR+NSM w E2E	64.1	–	46.3	–
ToG (Llama2-70B-chat)	–	68.9	–	57.6
Retrieve-Rewrite-Answer	–	79.36	–	–
G-Retriever	53.41	73.46	–	–
RoG	66.45	82.19	53.87	60.55
EtD (Llama2-13B-chat)	–	77.4	–	57.7
GNN-RAG	71.3	85.7	**59.4**	**66.8**
SubgraphRAG (8B)	70.49	86.57	47.08	56.85
SubgraphRAG (70B)	74.52	86.21	51.56	57.45
DCTR + 8B	72.13	**88.05**	50.12	59.72
DCTR + 8B($\leftrightarrow$)	67.52	84.62	38.92	49.19
DCTR + 70B	**75.72**	87.89	53.17	60.17

Table 1: Generate performance. GNN-RAG, RoG, KD-CoT, and G-Retriever use 7B fine-tuned Llama2 models.

Hyperparameter	**Path**	**GPT-4o**	**Answer**
0.1	0.863	0.845	0.923
0.2	0.872	0.852	0.935
0.3	0.880	0.858	0.942
0.4	0.884	0.860	0.950
0.5	0.890	0.867	0.956
0.6	0.883	0.852	0.932
0.7	0.880	0.861	0.940
0.8	0.876	0.856	0.942
0.9	0.886	0.865	0.946
1.0	0.878	0.850	0.941

Table 2: Performance under different hyperparameter values.

Hyperparameter Sensitivity Analysis. We investigate the impact of the DSRS loss hyperparameter, which balances the binary supervision and ranking supervision signals. As shown in Table 2, we evaluate its effect on the WebQSP dataset over a range of $[0.1, 1.0]$. The results indicate a clear trend: performance across all metrics generally improves as the hyperparameter increases from 0.1, **peaks at a value of 0.5**, and then begins to decline. This demonstrates the importance of the ranking signal, but also suggests that an over-reliance on it can be suboptimal. Based on this empirical evidence, we set the hyperparameter to 0.5 for all our main experiments.

β	**Path**	**GPT-4o**	**Answer**
0.1	0.806	0.783	0.894
0.5	0.829	0.801	0.911
1	0.890	0.867	0.956
2	0.832	0.803	0.900
4	0.825	0.798	0.900
8	0.829	0.801	0.904

Table 3: Performance under different β values.

Threshold	**Path**	**GPT-4o**	**Answer**
0.3	0.866	0.832	0.912
0.4	0.863	0.838	0.920
0.5	0.870	0.845	0.933
0.6	0.886	0.864	0.950
0.7	0.890	0.867	0.956
0.8	0.875	0.852	0.932
0.9	0.863	0.831	0.924

Table 4: Performance under different threshold values.

We investigate the effect of the DSRS temperature hyperparameter β, which controls the smoothness of the contrastive distribution and regulates the sharpness of the similarity weighting. As shown in Table 3, we evaluate its impact on the WebQSP dataset over a range of $[0.1, 8]$. The results show that performance across all metrics **peaks at** $\beta = 1$, indicating that moderate temperature scaling effectively balances gradient stability and discrimination. When β is either too small or too large, the model exhibits degraded performance, suggesting that extreme temperature values may either over-sharpen or over-smooth the contrastive signal. Consequently, we set $\beta = 1$ for all main experiments.

We further investigate the impact of the DSRS threshold hyperparameter, which determines the filtering strength of candidate subgraphs before reasoning. As shown in Table 4, we evaluate its effect on the WebQSP dataset across a range of $[0.3, 0.9]$. The results reveal that performance steadily improves as the threshold increases, **peaking at 0.7**, beyond which further increases lead to a slight decline. This indicates that a moderate threshold effectively balances noise reduction and information retention, while overly strict filtering may remove semantically useful subgraph connections. Consequently, we set the threshold to 0.7 for all subsequent experiments.

To ensure the full reproducibility of our work, this appendix provides a comprehensive list of all key hyperparameters. We detail the configuration for our proposed DCTR model, including both general training parameters

Hyperparameter	Value / Setting
DCTR Model (Ours)	
Optimizer	Adam
Learning rate	1×10^{-3}
Training epochs	25
DCTR Method-Specific Parameters	
DCFGR density threshold (η)	0.7
DSRS loss weight (α)	0.5
DSRS temperature (β)	1.0
LLM Generation (LLaMA 3.1)	
Sampling temperature	0.5
Top-p	1.0
Max new tokens	1024

Table 5: Hyperparameter settings for our experiments.

and method-specific ones. For reproduced baselines, we specify the configurations used. The settings for the Llama 3.1 generation module are also included for completeness. All settings are summarized in Table 5.

Prompt

Prompt for Dynamic Triple Scoring (F_{LLM})

You are an expert in knowledge graph reasoning. Your task is to evaluate the relevance of a "Candidate Triple" for answering a "Query", given the "Current Reasoning Path".
Instructions:

1. Analyze the Query, the Current Reasoning Path, and the Candidate Triple.
2. The Current Reasoning Path shows the steps already taken to answer the query.
3. The Candidate Triple is the next potential step.
4. Determine how relevant and logical the Candidate Triple is as the *next step* in the path.
5. Output a single floating-point relevance score between 0.0 (completely irrelevant) and 1.0 (highly relevant and logical).
6. Your response **MUST** contain **ONLY** the numerical score and nothing else.

Input:

Query: `"What is the capital of the country where the Eiffel Tower is located?"`

Current Reasoning Path:

- (Eiffel Tower, located in, France)

Candidate Triple: `(France, capital is, Paris)`

Relevance Score:

Figure 1: The prompt structure used for the dynamic triple scoring function, $F_{\text{LLM}}(query, P_{\text{cur}}, \zeta_{\text{cand}})$. The LLM is tasked to evaluate the relevance of a candidate triple given the context of the query and the reasoning path constructed so far. The expected output is a single float number (e.g., `0.95`).

Prompt of DCTR

Instruction:
Given a set of entity-relation triples from a structured knowledge base, determine the correct response to the query. Please provide each answer on a new line beginning with "output:".
User Input: `// Example for LLMs`
Knowledge Triples:

- (Darth Vader, film.character.portrayed_by, James Earl Jones)
- (Star Wars: A New Hope, film.film.character, Darth Vader)
- (Star Wars: A New Hope, film.film.initial_release_date, 1977)
- (Star Wars: The Empire Strikes Back, film.film.character, Darth Vader)
- (Star Wars: The Empire Strikes Back, film.film.initial_release_date, 1980)
- (Star Wars: Return of the Jedi, film.film.character, Darth Vader)
- (Star Wars: Return of the Jedi, film.film.initial_release_date, 1983)
- (Darth Vader, film.character.appeared_in, Star Wars: A New Hope)
- (Darth Vader, film.character.appeared_in, Star Wars: The Empire Strikes Back)
- (Darth Vader, film.character.appeared_in, Star Wars: Return of the Jedi)
- (James Earl Jones, film.actor.film, Star Wars: A New Hope)
- (James Earl Jones, film.actor.film, Star Wars: The Empire Strikes Back)
- (James Earl Jones, film.actor.film, Star Wars: Return of the Jedi)
- (Star Wars: A New Hope, film.film.production_company, Lucasfilm)

Query: In which years did the character portrayed by James Earl Jones appear in Star Wars films?
Response: `// Example explanation`
To answer this question, we first identify that James Earl Jones portrayed Darth Vader. Next, we find the movies in which Darth Vader appeared. According to the triples, he appeared in "Star Wars: A New Hope", "The Empire Strikes Back", and "Return of the Jedi". Finally, we extract the release years of those films. Therefore, the years are:

- output: 1977 (Star Wars: A New Hope)
- output: 1980 (Star Wars: The Empire Strikes Back)
- output: 1983 (Star Wars: Return of the Jedi)

User:
Triplets:

- (e_a, r_{ab}, e_b)
- (e_c, r_{cd}, e_d)

Question: what's . . . ?

Figure 2: Prompt for DCTR.