引导当下偏向型智能体的结构化方法

TatianaBelova Yuriy Dementiev Artur Ignatiev Danil SagunovITMO大学 ITMO大学 ITMO大学 ITMO大学

arXiv:2601.07763v1 [cs.GT] 12 Jan 2026

摘要

时间不一致行为，例如拖延或放弃长期目标，源于智能体对即时结果的重视程度远高于对未来结果的重视。这会导致全局次优行为，使计划频繁被修改或彻底放弃。在 Kleinberg and Oren（2014）提出的影响深远的模型中，这种行为被建模为一个当下偏向型智能体在任务图中朝目标前进，并在每一步依据折扣后的未来成本做出局部最优决策。因此，智能体可能会反复偏离最初计划。

Belova 等人（2024年）近期提出了该模型的双智能体扩展：一位完全知情的委托人试图引导当下偏向型智能体完成特定的关键任务集合，同时避免其放弃任务。这一模型刻画了行为场景中丰富多样的委托人—代理人动态。

本文对这一问题进行了全面的算法刻画。我们在参数化算法框架下分析其计算复杂性，重点关注在此情境中自然出现的图参数，例如树宽、顶点覆盖和反馈顶点集。我们的主要结果是：以任务图的树宽和不同的 $(v,\ t)$-路径成本数量为参数时，该问题存在固定参数可解算法。我们的算法涵盖了多种输入情形，例如有界边成本和受限任务图结构。我们证明，主要结果可为多种此类配置带来高效算法。

我们还给出了紧确的困难性结果，凸显了即使在节点数量有界、参数值为常数的最简单图上，该问题也极其困难，并说明了我们选择这些参数的理由。我们划定了问题中可解与不可解的区域，其中包括对 Belova 等人（2024）提出的开放问题的解答。

1 引言

当下偏向——即相较于未来结果，人们倾向于高估即时结果——是行为经济学中受到广泛研究的现象 [Laibson, 1997, Frederick等人， 2002]。这解释了为什么人们常常放弃长期有益的计划，转而选择短期轻松，进而在健康、金融、教育和生产力等领域产生次优结果。

[Kleinberg and Oren， 2014] 提出了一个图论模型，用于形式化序贯决策中的此类行为。在他们的模型中，一个具有当下偏向且时间不一致的智能体会在任务构成的有向无环图（DAG）中进行导航，并基于对未来成本的折扣评估来选择每一步。如果感知成本超过折扣后的奖励，智能体可能会在到达目标之前放弃计划。该模型为理解复杂系统中的拖延和动态决策提供了一个强大的抽象框架。

这一框架的一个自然扩展见于 [Belova 等人， 2024], 其中委托人（例如教师、系统设计者或平台）可以通过修改图来实施干预，引导智能体实现预期结果。这类干预措施可能包括移除干扰任务或添加有益捷径。我们的工作聚焦于这种情形：委托人希望确保当下偏向型智能体完成一系列重要任务并达到最终目标。

问题设定。 我们研究 Kleinberg–Oren 模型中的干预策略。在该模型中，智能体遵循一种时间不一致的规划模型 $M = (G,\ w,\ s,\ t,\ \beta,\ r)$:

- $G = (V,\ E)$ 是一个带有平行弧、用于表示任务结构的有向无环图,
- $w:\ E \to \mathbb{N}_0$ 为弧赋予成本,
- s 和 t分别是起点和目标顶点,
- $\beta \in (0, 1]$ 是当下偏差因子,
- r 是到达 t时获得的奖励。

当智能体位于顶点 v时，会评估所有 v-t 路径，并选择一条 $P = e_1 e_2 \dots e_k$ 使感知成本最小的路径

$$\zeta_M(P) = w(e_1) + \beta \cdot \sum_{i=2}^{k} w(e_i).$$

如果 $\zeta_M(P) > \beta \cdot r$，智能体就会完全放弃任务。否则，他会选择第一条弧 $e_1 = (v, u)$ 并移动到顶点 u，然后递归地重复这一决策过程。如果有多条路径具有最低感知成本，智能体会根据弧上的给定字典序，确定性地选择其中一条。

Belova 等人 2024] 近期开展的研究开启了委托人—智能体 问题的研究：委托人希望通过有限的图修改，确保智能体遵循一条从 s 到 t 的路径，并且该路径包含指定的关键弧集合 $T \subseteq E(G)$。研究考虑了两类干预措施:

- T-PATH-DELETION：我们能否从 G 中删除至多 k 条弧，使智能体遵循一条从 s 到 t 的路径，并且该路径包含 T中的所有弧?
- T-PATH-ADDITION：给定辅助加权弧集合 A，我们能否从 A 中添加至多 k 条弧（且不创建环），使智能体遵循一条从 s 到 t 的路径，并且该路径包含 T中的所有弧?

在这项工作中，我们将这两个问题统一为一个一般形式:

T-路径编辑

输入： 一个时间不一致规划模型 $M = (G,\ w,\ s,\ t,\ \beta,\ r)$，一个附加弧集合 A，满足 $G + A$ 为无环，且 w 将 w 映射到 $E(G) \cup A$ 中的 $\mathbb{N}_0$，以及一个关键弧集合 $T \subseteq E(G)$。**任务：** 计算所需弧编辑（从 $E(G)$ 中删除或从 A 中添加）的最小数量，使得智能体遵循一条 s–t 路径（G 中从 s 到 t 的路径），并遍历 T 中的所有弧。

在本文中，如果智能体按照 Kleinberg–Oren 模型行动，遍历 P 中的每条弧，且不会在任何中间顶点放弃任务，我们就称该智能体遵循路径 P。
值得注意的是，与先前固定预算 k 的形式不同，优化版本能够从更全局的角度考察问题的可处理性。

问题动机。 在涉及人类或自主智能体的新兴计算系统中，有效的任务委派与协调越来越依赖于对智能体行为的显式建模。现代多智能体架构通常遵循层级范式，由高层实体（委托人、中介方）与受时间或资源约束的智能体交互，以确保其与总体目标保持一致。

此类系统中的智能体可能表现出非理性行为，包括时间不一致性或局部短视行为。了解如何以及何时引导此类智能体采取期望的行为，尤其是在最小干预的情况下，是设计稳健、可解释且策略协调的智能体系统的一项重要基础。 T-路径编辑问题将此类情境形式化，通过受约束的结构修改来刻画目标对齐影响。

模型示例。 为说明我们的框架，设想这样一个场景：Bob 是一家大型科技公司的人工智能工程师，负责开发一个可用于生产环境、用于企业文档分析的基于大语言模型的智能体。他的最终目标是在真实任务上进行充分的基准测试后部署该智能体（顶点 t）。有多种策略可以实现这一目标，每种策略都由若干子任务构成，例如数据预处理、模型微调、提示工程、集成和评估。这些子任务构成一个有向无环图（见图1），其中每条弧代表一项行动，其权重对应于预估的工程投入或时间成本。

然而，Bob 表现出当下偏误行为：评估未来计划时，他会充分权衡紧接着要执行的任务，却会将剩余投入乘以折扣因子 $\beta = 1/2$。从节点 s 出发，他考虑了几条可能的执行路径：

- $P_1 = sbet$，$P_2 = scft$：严谨可靠的流水线，涵盖数据清洗、微调和完整评估：$\zeta_M(P_1) = \zeta_M(P_2) = 10 + \frac{10+10}{2} = 20$；
- $P_3 = sadt$：采用默认权重和部分测试的简化路径：$\zeta_M(P_3) = 10 + \frac{8+8}{2} = 18$；
- $P_4 = sbft$：基于仅提示适应、只进行最低限度验证的捷径：$\zeta_M(P_4) = 10 + \frac{2+10}{2} = 16$。

Bob选择了能使感知成本最低的路径 P_4，显而易见，在之后的每个顶点，他都会继续选择这条路径，并最终沿着 $sbft$ 前进。

现在来考虑Alice的角色。她是Bob的团队负责人和项目负责人。她的首要任务是确保完成关键评估步骤（弧（b，e）），因为这会直接影响模型的可靠性和监管合规。Alice承诺在成功部署后给予Bob奖励 $r = 36$。然而，由于Bob存在当下偏向，他在每个决策点都只将其感知为 $\beta \cdot r = 18$。

假设 Alice 只是通过移除弧 $(b,\ f)$ 来禁用仅提示捷径，希望阻断路径 P_4，那么在 s 处的感知成本会变为 $10 + \frac{8+8}{2} = 18 = \beta \cdot r$，这只能在路径 P_3 上实现。Bob 会沿着弧 $(s,\ a)$ 前进。相比之下，路径 $P_1 = sbet$ 只包含关键弧 $(b,\ e)$，其感知成本为 20——在当下偏向的影响下，这一成本太高，Bob 不会选择这条路径。

为了解决这个问题，Alice 采取了一项策略性编辑：她引入一条新的辅助弧 $(a,\ b)$，使 Bob 能够通往路径 $P = sabet$。这条新路径包含关键任务 $(b,\ e)$，得益于调整后的结构，它在关键决策点 a 的感知成本较低，为 $1 + \frac{10+10}{2} = 11$。因此，Bob 会沿着 $P = sabet$ 前进，完成所有必要阶段，使自己的行为符合 Alice 的目标。

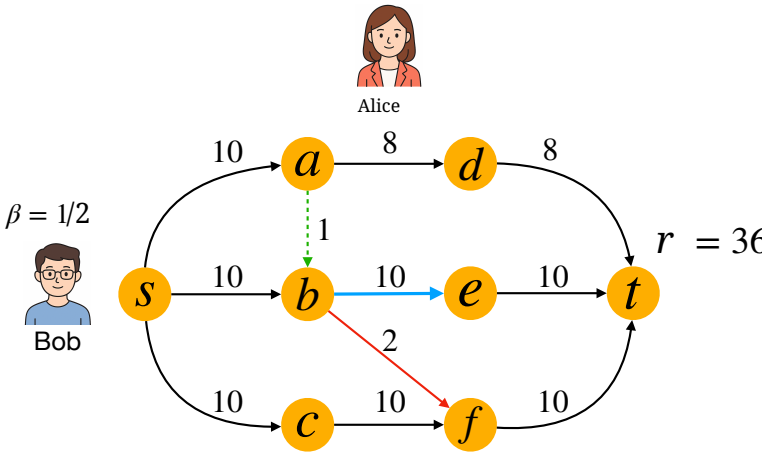


图1：示例：$T = \{(b,\ e)\}$，$A = \{(a,\ b)\}$。对于初始图 G，智能体沿着路径 $sbft$ 前进。删除 $(b,\ f)$ 后——路径 $sadt$。添加 $(a,\ b)$ 后——T-路径 $sabet$。

图参数为何重要。 在对行为代理进行建模的现实世界规划图中——例如教育、研究、数字助理或财务规划领域中的工作流——底层结构很少是任意的。相反，它往往反映了由人设计或操作的系统所具有的有限复杂性。因此，结构图参数对于理解实践中的算法可处理性至关重要 [Maniu 等人， 2019]。

下面，我们总结本研究中考察的关键图参数。所有这些参数都是无向图参数。我们根据 $G + A$ 的底层无向图来考察这些参数。

图的树宽(tw) G 是所有可能的树分解中最小的宽度 G，其中树分解的宽度定义为其最大包的大小减一。直观而言，树宽量化了一个图与“树”的相似程度。许多现实世界规划图由于具有层次结构而呈现较低的树宽，而动态规划算法在有界树宽的图上通常效率较高 [Cygan 等人， 2015]。

反馈顶点集数 (fvs) 是移除后可使无向图骨架变为无环的最小顶点集的大小。该参数反映了循环依赖的程度，在无环或近似无环的规划系统中通常较小。

顶点覆盖数 (vc) 是满足以下条件的最小集合 $C \subseteq V(G)$ 的大小：每条边 $(u, v) \in E(G)$ 都至少与 C 中的一个顶点关联。在拥有少数高度连接的控制点或验证步骤的任务图中，顶点覆盖通常较小。

路径长度(p) 指 DAG 中从源点 s 到目标点 t 的最长有向路径上的弧数。它反映了智能体所需规划或推理的最大深度。

图 G的树深度(td) 是满足以下条件的根树 T 的最小高度： G 是 T闭包的子图；也就是说，对于每条边 $(u, v) \in E(G)$， u或 v必须是 F中另一个顶点的祖先。等价地说，树深度衡量图在祖先—后代关系意义上的“嵌套深度”，并反映其层次复杂性。

这些参数不仅从算法视角来看十分自然，也会有机地出现在结构化任务图中。在任务以分层或层次化方式分解的许多系统中，树宽和反馈顶点集都很小，例如课程图谱 [Stavrinides and Zuev, 2023、Yang 等人， 2024], 多阶段工作流或模块化设计流程。即使存在环，它们通常也局限于局部，范围有限。在依赖关系通过少量协调节点或瓶颈节点进行汇聚的图中，顶点覆盖往往较小——这在多智能体或监督式规划设置中很常见 [Cygan 等人， 2015]。在大多数实际系统中，路径长度天然受到限制：在达到目标之前，没有现实中的智能体会执行数百个连续步骤。在面向人类的系统中，这一限制还受到注意力持续时间、任务疲劳或实时处理限制的约束。

综合来看，这些结构属性促使我们采用参数化方法来处理 T-路径编辑问题。通过分离出在实际部署中仍保持较小的参数，我们确定了具有实际意义的可处理情形，在这些情形下，可以为当下偏向型智能体计算高效的干预策略。这种结构与可处理性之间的联系，架起了理论模型与现实应用之间的桥梁。

参数化复杂性。 为了严格分析 T-路径编辑问题的计算复杂性，我们采用参数化复杂性的框架。该框架通过将输入的特定方面（例如图的结构参数）单独作为形式化的参数，使我们能够区分困难性的来源。

形式上，参数化问题是一种语言 $Q \subseteq \Sigma^* \times \mathbb{N}$，其中实例表示为一对 (I, k)，由主要输入 I 和参数 k组成。如果某个问题能在时间 $f(k) \cdot |I|^{\mathcal{O}(1)}$ 内解决，其中 f 是仅依赖于 k的某个可计算函数，则称该问题是固定参数可解（FPT）的。这与更广泛的 XP 类不同，在 XP 类中，运行时间允许为 $|I|^{g(k)}$，其中 g是某个可计算函数。

为了对难解问题进行分类，W-层级提供了一系列困难度类：$\mathrm{FPT} = \mathrm{W}[0] \subseteq \mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots \subseteq \mathrm{W}[P]$. 人们普遍认为 $\mathrm{FPT} \neq \mathrm{W}[1]$,，因此，证明 W[1]-困难性是不存在固定参数可解算法的有力证据。这类困难性证明通常通过从已知的 W[1]-完全问题进行参数化归约来获得。

此外，我们使用 Para-NP-困难性的概念，它指的是即使将参数固定为常数，仍然是 NP-困难的参数化问题。这个概念有助于排除低参数范围内的可解性。
关于参数化复杂性理论与技术的一般性参考资料，请读者参阅 [Cygan 等人， 2015]。

我们的贡献。 我们针对 T-PATH-EDITING 给出了算法方面的结果，重点说明了在何种结构和数值条件下可以实施高效干预。我们全面描述了这些问题的复杂性图景。我们的主要结果是一个以图 $G + A$ 的树宽（tw）和路径代价多样性为参数的算法：

定理 1。 T-PATH-EDITING 存在一个运行时间为 $|L|^{\mathcal{O}(\mathrm{tw})} \cdot m^{\mathcal{O}(1)}$ 的算法，其中 tw 等于 $G + A$ 的树宽，且

$$L = \left| \bigcup_{v \in V(G)} \{w(P) \mid P \textit{ is a } (v,t)\textit{-path in } G + A\} \right|.$$

这表明 T-PA TH-EDITING 在图结构（通过树宽 tw）和 d
权重多样性 are b 界。我们还证明：

- 树宽为 2 的图上的 NP-困难性，

- 关于顶点数量 n 和组合参数（p，vc）的 [1]W-困难性，
- 关于反馈顶点集、顶点覆盖和树深度的 XP算法，
- 针对成本多样性有界的成本受限实例的 FPT算法 $|\{w(e)\}|$。

相关工作。 对当下偏误行为的研究源于经典经济学理论。Samuelson 的折现效用模型 [Samuelson, 1937] 为跨期选择的形式化模型奠定了基础。此后，这一模型被扩展为双曲线折扣和准双曲线折现框架 [Laibson, 1994, McClure et al., 2004],，能够更准确地捕捉行为随时间变化所产生的不一致性。我们采用的模型遵循 Kleinberg and Oren 引入的框架 [Kleinberg and Oren, 2014, Kleinberg and Oren, 2018], ，其中，当下偏向型智能体通过根据折现成本评估选择路径，在任务图中进行导航。该模型可以被视为准双曲线折现的离散类比，并融入了 Akerlof 关于显著性的概念 [Akerlof, 1991]。

尽管 Kleinberg–Oren 模型具有坚实的实证依据，但它并未涵盖时间不一致行为的所有已知心理学层面 [Frederick等人, 2002]。然而，该模型推动了大量算法研究，包括复杂性和近似分析 [Gravin et al., 2016, Kleinberg等人，2016, Kleinberg等人，2017, Halpern and Saraf, 2023, Meyer et al., 2022, Dementiev 等人，2022 年, Fomin和Strømme, 2020]。

近来有几项研究从互补的方向扩展了 Kleinberg–Oren 模型。 [Akagi et al., 2024, Akagi et al., 2025] 提出了一种连续时间变体，并给出了最优奖励放置和放弃条件的闭式解。 [Belova et al., 2024] 引入了 T-PATH-EDITING 框架，并分析了双代理设置下的结构性干预。

我们研究的委托—代理表述可视为对经典激励子图和 P-激励子图问题的一种自然且适时的推广。这些问题旨在找出一个子图，使具有当下偏误的智能体能够在不放弃的情况下到达目标（沿着特定路径 P）。 [Tang 等, 2017；Albers 和 Kraft，2017；Albers 和 Kraft，2019；Oren 和 Soker，2019；Fomin和Strømme， 2020]。

最后，我们的研究也置于图修改问题这一更广泛的背景之中；这是算法图论中一个发展成熟的领域。这类问题旨在通过删除或添加边或顶点来改变图的结构，以满足所需性质。如需全面概述，可参阅相关综述，例如 [Burzyn 等，2006；Natanzon 等，2001；Crespelle 等， 2023]。

2 有界顶点覆盖图上的多项式时间算法

本节证明，当 $G + A$ 的顶点覆盖大小为常数时， T-PATH-EDITING 可以在多项式时间内求解。

关于有界顶点覆盖图的第一个观察是，这类图的路径长度有界。

观察 1. 在有向无环图 G 中，每条路径最多由 $2 \cdot \mathrm{vc}(G) - 1$ 条弧组成。

证明. 令 S 为 G 的最小顶点覆盖，也就是说， G 中的任意一条弧，其起点或终点（也可能两者）都在 S S 中。我们有 $|S| = \mathrm{vc}(G)$。

采用反证法，假设 G 中存在一条路径 P，其中至少有 2 条 vc(G) 弧。由于 G 是无环的，这条路径是简单路径，并且至少包含 2 个 vc(G) 个不同顶点。根据鸽巢原理， P 上有两个相邻顶点不属于 S。也就是说，G 中存在一条弧，其起点在 S 之外，终点也在 S 之外。由此得到矛盾，证明完毕。 □

在继续介绍本节的主要结果之前，我们先给出一个辅助算法结果。我们将在主要算法中将其用作子程序。

命题 1. 存在一个多项式时间算法，给定两个不相交集合 A, B 和一个由 m 个数对组成的序列 $(a_1, b_1), (a_2, b_2) \ldots, (a_m, b_m) \in A \times B$，以及一个集合 $R \subset (A \cup B)$，可找到最小可能的 $X \subset (A \cup B)$，使得

- *for each* $i \in [m]$, $\{a_i, b_i\} \cap X \neq \emptyset$ *and*
- $X \cap R = \emptyset$,

或报告称 X 不存在。

证明。 我们给出一种多项式时间算法。首先，该算法判断 X 是否存在。如果存在一对 $(a_i,\ b_i)$，且两者均满足 $a_i,\ b_i \in R$，那么 X 无法同时满足这两个约束。在这种情况下，算法报告称 X 不存在，并停止。

算法的下一部分由两个阶段组成。在第一阶段，算法构造一个元素集合 $X' \subset A \cup B$，这些元素应属于任意有效解 X。
集合 X' 以迭代方式构造，直到 R 为空。当 R 不为空时，存在一个元素 $a \in R$。不失一般性，$a \in A$。然后，对于每一对满足 $a_i = a$ 的 $(a_i,\ b_i)$，算法将 b_i 纳入 X'，因为命中 $\{a_i, b_i\}$ 的唯一可能方式就是取 b_i。接着，算法从 A 和 R 中移除 a，并移除所有包含 a 的配对。如果 R 不为空，算法就用 R 中的下一个元素重复这一迭代。否则，算法进入第二阶段。情形 $a \in B$ 是对称的。
在第二阶段，算法首先移除所有满足 $|\{a_i, b_i\} \cap X'| > 1$ 的配对 $(a_i,\ b_i)$，因为这些配对都已经由 X' 命中。然后，算法从 A 和 B 中移除 X' 的所有顶点。
经过这样的预处理后，算法已拥有 $R = \emptyset$、$A \cap X' = B \cap X' = \emptyset$，以及一个由剩余配对组成的列表，其中 $(a_i,\ b_i) \in A \times B$。算法接下来的任务是，在没有任何额外限制的情况下，找到大小最小的 $X'' \subset (A \cup B)$，使其与每个剩余配对都相交。
这个问题等价于二分图 G 上的二分图顶点覆盖问题，其顶点集为 $A \cup B$，边集则等于剩余配对的集合。G 中的顶点覆盖与 $A \cup B$ 的一个子集之间存在一一对应关系，该子集与每个剩余配对都相交。二分图中的最小顶点覆盖可以通过最大二分图匹配算法，并结合 K˝onig 定理的构造性证明，在多项式时间内求得。关于这一主题的更多信息，可参阅经典算法教材 Kleinberg 和 Tardos [，以及 Cormen 等人 2006] [2009]。

该算法构造　　在多项式时间内 G，并找到最小顶点覆盖 X''，其等于最小
基数集合命中　　所有剩余的配对。最后，算法输出 $X = X' \cup X''$ 并停止。
该算法的正确性由上述讨论可知。证明完毕。 □

我们注意到，命题 1是民间常识，本质上是一个在二分图中寻找最小基数顶点覆盖的问题。为完整起见，我们在下文给出证明及必要的参考文献。
现在我们给出一个以 vc 为参数的 T-PATH-EDITING 的 XP 算法。在以下证明中，我们的算法会比较感知成本。

定理2. *T-PATH-EDITING 存在一个运行时间为 $m^{\mathcal{O}(\mathrm{vc}^2)}$ 的算法，其中 vc 是 $G + A$ 的顶点覆盖数。*

证明*T*-PATH-EDITING算法，如果开发者提供实例(这会正确地报告这一点)，则最优解，该实例属于

解存在，算法会正确地报告这一点。
不失一般性，我们假设算法还会额外获得一个顶点覆盖 C 或 $G + A$，满足 $|C| \leq 2 \cdot \mathrm{vc}(G + A)$；这是因为可以在多项式时间内找到这样的顶点覆盖（例如参见 [Papadimitriou 和 Steiglitz, 1982] 关于顶点覆盖的 2-近似算法）。

设 G^* 是通过最优弧编辑序列得到的图。通俗地说，我们的算法事先并不知道 G^*，因此其目标只是猜出 G^* 的相关结构：智能体的 s-t 路径，以及 G^* 中重要的最短路径。由于所有路径长度都受 $G + A$ 限制（因此也受 G^* 限制），这些猜测的搜索空间以 $m^{\mathcal{O}(\mathrm{vc}^2)}$ 为界。我们的算法概要如下。

1. 遍历 $G + A$ 中所有 s-t 路径 P（对智能体路径的猜测）；
2. 构造一个集合 S，其中包含 C 的所有顶点以及路径 P 的所有顶点。也就是说，P 的每条弧的两个端点都在 S 中，并且 G 的每条弧的起点位于 S、终点位于 S，或二者皆是；
3. 对每个 $v \in S$，遍历从 v 到 t 的一条路径 R_v，位于 $G + A$ 中（对最短路径的猜测）；
4. 找到一个尽可能小的弧删除集合，使得
   - 迫使智能体遵循 P，并且
   - 对于每个 $v \in S$，都会切断所有长度小于 R_v 的 v-t 路径。

接下来进入证明的形式化部分。我们将形式化描述算法的各个步骤，并在需要时为其提供正确性论证。

遍历智能体的 s-t路径和关键顶点。 在外层循环中，算法遍历弧集 $P \subset (E(G) \cup A)$，使得 $P \supset T$和 P在 $G + A$中构成一条简单的 s-t 路径。具体而言， $|P| \leq 2|C| - 1$。
稍微滥用一下记号，我们将 P既视为弧集，也视为 $G + A$中的一条路径。令 d为 P的长度，并令u_0， u_1、.....、 u_d 为 P上的顶点，按自然顺序排列，因此 $u_0 = s$和 $u_d = t$。对于每个 $i \in [d]$，算法用 e_i表示 P中连接 u_{i-1}和 u_i的边。

然后，算法构造一个关键顶点集合 $S := \{u_0, u_1, \ldots, u_d\} \cup C$。

遍历关键最短路径的边。 然后，算法遍历 $R \subset (E(G) \cup A)$，使得 $|R| \leq (|S| + 1) \cdot (2|C| - 1)$，并且 $R \supset P$和

(R1) for each $i \in [d]$, $w(e_i) + \beta \cdot \mathrm{dist}_R(u_i, t) \leq \beta \cdot r$.

在本证明中及下文中，我们稍微滥用记号，将 $\mathrm{dist}_R(x, y)$ 记为图 $(V(G), R)$ 中 x 与 y之间的距离。

R 背后的直觉是：当猜测正确时，它是智能体遵循 P所需的所有最短路径的并集。(R1) 特别确保智能体不会放弃 P。

突出显示局部障碍。 在固定 P和 R后，算法现在旨在寻找最小大小的子集$X \subset E(G)$（即必须删除的边），使得对于 $G' = (V(G), R \cup (E(G) \setminus X))$：(a) 我们不删除 R中的任何边；(b) G' 中的距离与 R中的距离一致；(c) 智能体遵循 G'中的每条边 $e_i \in P$。形式化地表示，这变为

(X1) $X \cap R = \emptyset$;

(X2) 对于每个 $v \in S$， $\mathrm{dist}_{G'}(v, t) = \mathrm{dist}_R(v, t)$;

(X3) 对于每个 $i \in [d]$、每个顶点 $v \in V(G)$ 以及每条从 u_{i-1} 到 v 和 $e' \neq e_i$ 的边 $e' \in E(G')$，均成立

$$w(e_i) + \beta \cdot \mathrm{dist}_{G'}(u_i, t) \prec w(e') + \beta \cdot \mathrm{dist}_{G'}(v, t).$$

以下我们用 $\prec$ 表示根据智能体的行为对感知成本进行比较。也就是说，如果 $\prec$ 左右两侧的部分相等，则比较结果等价于按照边的排序对 e_i和 e' 进行比较。

我们断言，这些约束本质上等价于智能体在 G' 中遵循 P。
断言 1. 设 $X \subset E(G)$ 满足 *(X2)*。那么， R 满足 *(R1)* 且 X 满足 *(X3)*，当且仅当智能体在G'中遵循 P。

断言 *1* 的证明。 由于 (X2) 成立，当 $v \in S$ 时， $\mathrm{dist}_R(v, t)$ 与 $\mathrm{dist}_{G'}(v, t)$ 等价。当且仅当满足以下条件时，智能体在 G' 中遵循 P：

- 智能体不会在 $u_0, u_1, \ldots, u_{d-1}$中的任何一个位置放弃任务，因此对于每个 $i \in [d]$，我们有$w(e_i) + \beta \cdot$距离$_{G'}(u_i, t) \leq \beta \cdot r$. 这等价于 (R1)。
- 对于每个 $i \in [d]$，智能体从 u_{i-1}出发时总是更倾向于选择边 e_i，这意味着选择从 u_{i-1} 出发的任何其他边e' 都会产生更高的成本估计（或者对 e' 和 e_i 的估计相同，但 e' 在边的排序中排在e_i 之后）。这等价于 (X3)。

这些等价关系证明了该结论。 ⌟

现在我们来说明算法如何以算法化的方式找到合适的 X。

关键顶点之间的障碍。 算法首先通过找出必须删除的、端点为关键顶点的弧来构造 X_1，以满足这些性质。 X_1 由 G中的所有弧 e' 组成，这些弧满足： e' 始于某个$x \in S$，并终于某个 $y \in S$，且满足以下条件之一：

(i) 距离$_R(x,\ t) > w(e') +$距离$_R(y,\ t)$，或

(ii) $x = u_{i-1}$ 对于某个 $i \in [d]$, $e' \neq e_i$ 和 $w(e_i) + \beta \cdot$距离$_R(u_i, t) \succ w(e') + \beta \cdot$距离$_R(y, t)$.

我们断言， X_1 中的所有弧都必须删除。

断言 2. 如果 X 满足 *(X2)*、*(X3)*，那么 $X_1 \subset X$。

断言 *2* 的证明。 设 $X \subset E(G)$ 满足 (X2)、(X3)。采用反证法，假设 $e' \in X_1 \setminus X$。记 $x, y \in S$ 为 e'的起点和终点，即 e' 从 $x \in S$ 到 $y \in S$。

如果 (i) 成立，那么距离$_R(x, t) >$ 距离$_R(y, t)$，则由 (X2) 可得距离$_{G'}(x, t) >$ 距离$_{G'}(y, t)$，这与最短距离的定义矛盾，因为 e' 从 x 到 y。

如果 (ii) 成立，那么根据 (X2)，我们有

$$w(e_i) + \beta \cdot \mathrm{dist}_{G'}(u_i, t) \succ w(e') + \beta \cdot \mathrm{dist}_{G'}(y, t),$$

而这显然违反了 (X3)，因为 e_i 和 e' 都从 u_{i-1} 开始。由此得到的矛盾完成了该命题的证明。」

稍后我们将证明，在任何最优解中， S 中应被删除但不属于 X_1 的顶点之间不存在弧。

与非关键顶点关联的障碍。 然后，对于每个非关键顶点 $v \in V$（G） $\setminus S$，算法构造一个弧对集合 $\mathcal{X}_v$。具体做法如下。考虑一条终止于 v 的弧 e' 和一条起始于 v 的弧 e''。令 x 和 y 分别为 e' 的起点和 e'' 的终点。注意， $x, y \in S$，因为两个非关键顶点之间不能存在弧。如果满足以下条件，则将弧对 $\{e', e''\}$ 添加到 $\mathcal{X}_v$ 中

(i) $\mathrm{dist}_R(x, t) > w(e') + w(e'') + \mathrm{dist}_R(y, t)$，或

(ii) $x = u_{i-1}$ for some $i \in [d]$ and $w(e_i) + \beta \mathrm{dist}_R(u_i, t) \succ w(e') + \beta(w(e'') + \mathrm{dist}_R(y, t))$.

我们证明，对于 X 而言，必须命中 $\mathcal{X}_v$中的每一对。

断言 3. 设 $X \subset E(G)$ 满足 *(X2)*、*(X3)*。那么，对于每个 $v \in V(G) \setminus S$ 以及每个 $Y \in \mathcal{X}_v$，都有 $X \cap Y \neq \emptyset$.

断言 *3* 的证明. 证明与断言 2 的证明类似，只是这里使用一对边 e', e'' ，而非单条边 e'。采用反证法，假设存在 $v \in V(G) \setminus S$ 和 $\{e', e''\} \in \mathcal{X}_v$ ，使得 $e', e'' \notin X$。

若 $\{e', e''\}$ 满足 (i)，则由 (X2) 得

$$\mathrm{dist}_{G'}(x, t) \succ w(e') + w(e'') + \mathrm{dist}_{G'}(y, t),$$

这与最短距离的定义矛盾，理由与断言 2 的证明相同。

剩下的情况是 $\{e', e''\}$ 满足 (ii)。由 (X2) 和 $u_i, y \in S$， 可得距离$_R(y, t) =$距离$_{G'}(y, t)$， 且距离$_R(u_i, t) =$距离$_{G'}(u_i, t)$。 由于 e'' 从 v 通向 y，根据距离的定义，有距离$_{G'}(v, t) \leq w(e'') +$距离$_{G'}(y, t)$。将这些与 (ii) 结合可得

$$w(e_i) + \beta \cdot \mathrm{dist}_{G'}(u_i, t) \succ w(e') + \beta \cdot \mathrm{dist}_{G'}(v, t),$$

与 (X3) 矛盾。该断言的证明至此完成。 」

算法的内部步骤。固定两个外层循环确定的 P 和 R，并针对每个 $v \notin S$ 构造出 X_1 和 $\mathcal{X}_v$ 后，算法会找到满足 (X1) 以及断言 2 和断言 3 右侧条件的最小基数集合 X，也就是说， X 与 R 不相交， X 包含 X_1，且 X 与 $\bigcup_{v \notin S} \mathcal{X}_v$ 中的每一对都相交。

算法也可能判定 X 不存在（这基本上意味着对 P 和 R 的选择有误，或者根本不存在解）。例如，当 $X_1 \cap R \neq \emptyset$ 时就会发生这种情况。

由于 $\mathcal{X}_v$ 和 $\mathcal{X}_{v'}$ 包含两两不同的弧，因此可以针对每个顶点 $v \notin S$，独立地求出与 $\mathcal{X}_v$ 中的每一对都相交的最小集合。对于每个 $v \notin S$，算法使用多项式时间子程序命题 1，找出能够命中 $\mathcal{X}_v$ 中的每一对且不与 R相交的最小集合 X_v。如果命题 1 的子程序报告称 X_v 不存在，算法便会正确判定 X 不存在，并进入外层循环的下一次迭代。

如果对每个 $v \notin S$都成功构造出 X_v，算法便通过 X 构造 X：$= X_1 \cup \bigcup_{v \in V(G) \setminus S} X_v$。算法得到一个大小为 $|R \cap A| + |X|$的解：应将 $(R \cap A)$中的边加入 G，并将 X 中的边从 G中移除。算法用此解更新当前最优解，并进入 P和 R的下一次选择。

算法的描述到此结束。下面我们来证明其正确性。

算法永远不会输出错误的解。我们先证明 $(R \cap A) \cup X$ 始终是一个正确的解。注意，如果我们将 $R \cap A$ 添加到 G 中，并从 G中移除 X，就会恰好得到 $G' = (V(G)\ R \cup (E(G) \setminus X)$。因此，只需证明智能体始终遵循 G'中的 P。

我们证明，在算法的内部步骤中构造的 X 满足 (X2) 和 (X3)。

命题 4。如果 X 满足 *(X1)*，X 包含 X_1，且 X 与 $\bigcup_{v \notin S} \mathcal{X}_v$的每一对都相交，那么 X 满足 *(X2)* 和 *(X3)*。

命题 *4* 的证明。该 用反证法证明。我们先证明 (X2)，再利用 X 满足 (X2) 来证明 (X3)。

X 满足 (X2)。假设 X 不满足 (X2)（G' 中的距离与 R 不一致）。这意味着存在 $x \in S$，使得 $\mathrm{dist}_{G'}(x,t) \neq \mathrm{dist}_R(x,t)$。在所有可能的此类选择 x 中，选择在拓扑上尽可能接近 t 的 x。由于 $R \subset E(G')$，从 R 出发的所有弧路径都存在于 G' 中。因此，我们有 $\mathrm{dist}_{G'}(x,t) < \mathrm{dist}_R(x,t)$。注意，从 x 到 t 的路径至少包含一条弧，因为 $\mathrm{dist}_{G'}(t,t) = \mathrm{dist}_R(t,t) = 0$。

设 v 为 G' 中最短 x-t 路径上位于 x 之后的第一个顶点，并设 e' 为该路径上的第一条弧（e' 从 x 指向 v）。如果 v 是关键的，即 $v \in S$，那么根据 x 的定义，$\mathrm{dist}_R(v,\ t) = \mathrm{dist}_{G'}(v,\ t)$。于是我们有

$$\mathrm{dist}_R(x,t) > \mathrm{dist}_{G'}(x,t) = w(e') + \mathrm{dist}_{G'}(v,t) = w(e') + \mathrm{dist}_R(v,t).$$

也就是说，e' 满足构造 X_1中的条件 (i)，因此 $e' \in X_1 \subset X$。但是 $e' \in E(G') = R \cup (E(G) \setminus X)$，所以 $e' \in R$。这与 (X1) 矛盾。

当 $v \notin S$时，存在第二条弧 e'' 和第三个顶点 y，它们位于 G'中的最短 x-t 路径上。顶点 y属于 S，因为 S 之外的顶点之间没有弧。因此，$\mathrm{dist}_R(y,\ t) = \mathrm{dist}_{G'}(y,\ t)$。像之前证明中那样继续推导，我们得到

$$\mathrm{dist}_R(x,t) > w(e') + w(e'') + \mathrm{dist}_R(y,t),$$

这意味着 $\{e', e''\} \in \mathcal{X}_v$，因此 e'，e'' 中至少有一个属于 X。但是，这两条边都属于 $E(G') = R \cup (E(G) \setminus X)$，这意味着 $R \supset (X \cap \{e', e''\})$，像之前一样与 (X1) 矛盾。

X **满足 (X3)**。假设 X 不满足 (X3)（智能体在 G'中转离 P）。也就是说，存在$i \in [d]$ 以及一条弧 $e' \in E(G')$，从 u_{i-1} 通向某个 $v \in V(G)$，且满足 $e' \neq e_i$ 和

$$w(e_i) + \beta \cdot \mathrm{dist}_R(u_i,t) \succ w(e') + \beta \cdot \mathrm{dist}_{G'}(v,t), \tag{1}$$

其中 $\mathrm{dist}_R(u_i,t) = \mathrm{dist}_{G'}(u_i,t)$，这是由 $u_i \in S$ 和 X 满足 (X2) 得出的。

若 $v \in S$，则我们也可以在 (1) 中将 $\mathrm{dist}_{G'}(y,t)$ 改写为 $\mathrm{dist}_R(y,t)$。因此，e' 满足 X_1的构造中的 (ii)。于是，$e' \in X_1 \subset X$。这与之前在该断言的证明中一样，和 (X1) 矛盾。

剩下的情况是 $v \notin S$。G' 中的最短 v-t 路径至少包含一个弧顶点。令 e'' 为这条路径上的第一条弧，令 y 为这条路径上的第二个顶点（e'' 从 v 通向 y）。我们有 $y \in S$。于是 $\mathrm{dist}_{G'}(v,t) = w(e'') + \mathrm{dist}_{G'}(y,t) = w(e'') + \mathrm{dist}_R(y,t)$。将此与 (1) 结合可得，$e', e''$ 满足 $\mathcal{X}_v$的构造中的 (ii)。因此，$\{e', e''\} \cap X \neq \emptyset$，这与 (X1) 矛盾。

该断言的证明完毕。 ⌟

结合命题 4 和断言 1，我们得到智能体在 P 中遵循 G'，并且根据构造，P 遍历所有弧 T。因此，该算法只输出正确的解。

该算法找到一个最小规模解。为了证明该算法输出的是最小规模解，我们展示一个有效的选择：P 和 R，它们能得到最优答案。

为了说明这一选择，令 G^* 为对 G 应用最优编辑序列后得到的图。令 P^* 为智能体在 G^* 中的 s-t 路径。我们有 $T \subset P^*$，并且 $|P^*| \leq 2\,\mathrm{vc}(G+A) - 1 \leq 2|C| - 1$。令 S 为 P^* 和 C 中所有顶点的并集。对于每个 $v \in S$，令 R_v^* 为 G^* 中连接 v 和 t 的最短路径的边集。将 R^* 构造为 P^*和 $\bigcup_{v \in S} R_v^*$ 的并集，显然

$$|R^*| \leq (2|C|-1) + |S| \cdot (2|C|-1) \leq (|S|+1) \cdot (2|C|-1),$$

并且 R^* 满足 (R1)，因为智能体不会在 G^*中放弃 P^*。

因此， P^* 和 R^* 是算法在某次迭代中会考虑的 P 和 R 的有效选择。注意，X^* （针对 G^*的一系列弧删除）满足针对 P 和 R这一选择的 (X1)、(X2)、(X3)。因此，算法将找到边删除集 X，其中 $|X| = |X^*|$，并将答案更新为大小为 $|R^* \cap A| + |X^*|$的弧编辑集。根据 G^*的定义，这是可能的最小值。

运行时间分析。 算法的内部步骤关于 m 是多项式时间，因此我们需要分析其两个循环所决定的迭代次数。在第一个循环中，算法遍历大小至多为 $\mathcal{O}(\mathrm{vc})$ 的弧集；在第二个循环中，它遍历大小为 $\mathcal{O}(\mathrm{vc}^2)$ 的弧集。两者相乘，得到总计 $m^{\mathcal{O}(\mathrm{vc}^2)}$ 次迭代。 □

3 参数图景

我们的主要算法结果定理 1 的证明，与定理2 的证明共享基本思路：猜测从 v 到 t 的每个距离，确保添加必要的弧，并删除必要的弧，使所得距离与猜测一致，且所得智能体路径在 T 中遍历每条弧。为了达到定理 1 的运行时间，我们不再像定理2 中那样猜测智能体路径 P，因为这种方法需要按字典序比较 $G + A$ 中任意的弧。在定理 1 的证明中，我们充分利用了这样一点：只需将$G + A$ 中的弧与来自 T 的弧进行比较即可。

除此之外，定理 1 是一种基于树分解的技术性动态规划算法，在参数化复杂性领域很常见（例如，见 [Cygan 等人， 2015]）。定理 1 的完整证明见附录 A。

现在我们展示，定理 1 涵盖了 T-PATH-EDITING 的若干结构情形，包括有界顶点覆盖情形和常数弧成本。

引理1. 设 (G, w) 是一个弧成本为非负整数的有向无环图 $w\colon E(G) \to W$。设 t 是 G中的一个顶点。设 p 是 G中一条路径所含弧数的最大值。那么

$$|L| \leq \min\{(p+1)^{|W|}, 1 + p \cdot \max W\} \textit{ and}$$
$$|L| \leq \min\{m^{p+1}, m^{2\cdot \mathrm{vc}}, m^{2^{\mathrm{td}+1}}, m^{2\cdot \mathrm{fvs}+1}\}.$$

证明。 要证明引理的第一部分，注意到 (G, w) 中的任意路径至多由 p 条弧组成。因此，(G, w) 中一条路径的成本是取自 W 的 p 个整数之和。路径 P的成本可通过计算其中成本为 w 的弧的数量来确定，对每个 $w \in W$ 均如此（至多有 $(p+1)^{|W|}$ 种组合）。这个和是一个整数，其最小值可以是 0（空路径），最大值可以是 $p \cdot \max W$（p 条弧，每条弧的成本都达到可能的最大值）。 $|L| \leq \min\{(p+1) \cdot W,\ 1 + p \cdot \max W\}$由此得证。

要看出第二部分，请注意， P中的任意路径都是由至多 p 条弧组成的集合，而此类集合的数量至多为$\sum_{i=0}^{p} \binom{m}{i} \leq m^{p+1}$。对于顶点覆盖，我们有 $p \leq 2 \cdot \mathrm{vc} - 1$ （观察 1）。对于树深度，我们有$p \leq 2^{\mathrm{td}+1} - 2$ （例如参见 [Demaine 等人， 2019]）。
反馈顶点集大小有界的图，其路径长度可能无界。但 G 中的任意（v， t）-路径，都由其起始弧（m 种选择）以及反馈顶点集中的每个顶点对应的0条或2条弧（$(1 + \binom{m}{2})^{\mathrm{fvs}}$种选择）唯一确定，因此路径总数至多为 $m^{2\,\mathrm{fvs}+1}$。 □

引理1有一些有趣的推论。由于三个参数 vc、fvs、td 均不小于 tw，定理 1 中算法的运行时间上界为 $m^{\mathcal{O}(\mathrm{vc}^2)}$、$m^{\mathcal{O}(\mathrm{fvs}^2)}$、 $m^{\mathcal{O}(\mathrm{td}\cdot 2^{\mathrm{td}})}$。也就是说，特别地，定理 1 推广了定理2。

当不同弧权值的数量为常数时，定理 1 的算法效率更高，运行时间为 $p^{\mathcal{O}(\mathrm{tw}\cdot|W|)} m^{\mathcal{O}(1)}$。特别地， T-路径编辑关于 fvs $+|W|$ 和 td $+|W|$都是固定参数可解的。

4 T-路径编辑的困难性

本节证明 T-路径编辑的困难性，说明若不破坏 FPT $\neq$ W[1]，就无法将我们的 XP 算法转化为 FPT 算法。

定理 3. T-路径-编辑 关于以下参数是 W[1]-困难的：

- n，当允许 G中存在平行弧时，

- p +vc，当 G 中禁止使用平行弧时。

证明。我们给出一种针对修正后的 k-和问题（MkS）的参数化间隙归约；该问题是 W[1]-难的[Dementiev 等人， 2022]。在 MkS 中，给定 k 个整数集合 X_1，.. .， X_k 以及目标和 S。目标是确定是否存在一种选择：从每个 X_1， X_2，.. .， X_k中各选取一个整数，使它们的总和为 S。
给定 MkS 的一个实例（X_1，.....， X_k， S），我们构造一个相应的 T-路径-编辑实例，如图 2 所示。对于每个集合 X_i，我们构造一个由两个顶点 v_i 和 v_{i+1} 组成的构件，并用多条平行弧将它们连接起来，每条弧分别表示一个元素 $x \in X_i$，其权重为 $w = x$。目标是确保智能体通过包含关键弧（b， t）的唯一 T-路径 $sabt$ 到达顶点 t。

该构造使用以下参数：对于任意 $0 < \varepsilon < 1$， $\beta = \frac{1}{4} + \varepsilon$； $x = 10 \cdot S$； $A = \emptyset$； $T = \{(b,\ t)\}$； $x = S + 2$。我们将（s， c）、（c， t）中的每条弧复制为 $z := |X_1| + \ldots + |X_k|$ 个具有相同弧成本的相同副本。破坏 $s \to c \to t$ 中的路径 G 至少需要 z 次编辑。
我们将证明，当且仅当存在一个大小不超过 $z - k$ 的T-路径编辑问题的解时，修改后的 k-SUM 问题的答案为“是”。

假设该 T-路径编辑实例存在一个大小至多为 $z - k$ 的可行解。那么，唯一被移除的弧是构件中顶点 v_1、...、...、v_k、 t 之间的弧；从图的下部移除任何弧的代价都会过高，或者会阻断所需的边（b、 t）。令 G' 表示修改后的图，并令 S' 表示 G' 中最短 v_1-t 路径的成本。

现在我们分析 S' 的可能取值。如果 $S' > x - 2$，则 $S' \geq x - 1$，从 $s \to a \to \ldots$ 开始的任意路径的感知成本至少为 $\beta \cdot (1 + x - 1)$，这高于下部路径 $s \to c \to t$ 的感知成本，即 $\beta \cdot (x - \varepsilon)$。因此，智能体不会选择上方的 $s \to a \to b \to t$ 路径。如果 $S' < x - 2$，则 $S' \leq x - 3$，并且当智能体在顶点 a 做出决策时，它会比较

$$1 + \beta \cdot S' < 0 + \beta \cdot (x + 1),$$

并且更愿意经过 v_1，而不是直接前往 b。
因此，智能体遍历目标弧（b， t）的唯一方式是 $S' = S = x - 2$。这意味着存在一组弧权重序列，对应于原始 MODIFIED k-SUM 实例的一个有效解。反过来，任何此类解都会在 T-PATH-EDITING 中产生一种编辑模式，编辑次数不超过 $z - k$，从而迫使智能体按预期经过弧（b， t）。

与此同时，正如证明中显而易见的那样，对于此类 T-PATH，另一种类型的唯一解
-编辑实例— 用于打断路径 $s \to c \to t$—其大小至少为 z。由此完成我们的归约。

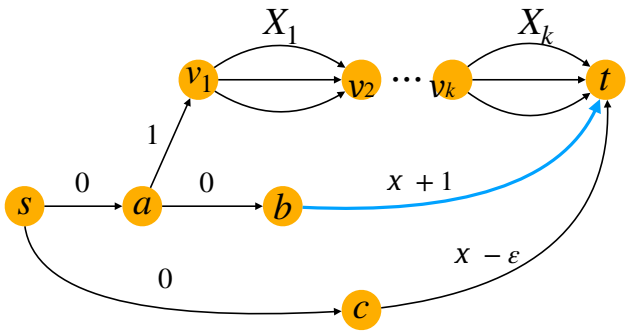


图 2：定理 3 中图 G的构造。

请注意，图 G 的构造仅涉及 $k + 5$ 个顶点。与此同时，这个例子很容易通过在每条弧的中间添加一个顶点，改造成一个不含平行弧的例子。在这样的图中，顶点覆盖数不会超过 $k + 5$，而参数 p（即图 G中一条路径的最大弧长）等于 $2k + 2$。因此，我们得到一个参数化归约：在一种情况下归约到参数为 $n = k + 5$ 的 T-PATH-EDITING，另一种情况下归约到 p +vc $\leq 3k + 7$。
至此，归约完成，证明了这两种情形均具有 W[1]-困难性。 □

定理 3 的归约中的图是串并联图，众所周知，这类图的树宽至多为二 [Brandst¨adtetal., 1999]。

推论 1。 T-路径-编辑 是 NP-困难的，对于 tw $= 2$。

请注意，该结果是紧的：当 tw($G + A$) $= 1$ 时，该图是一棵树，而 T-PATH-EDITING 问题变得微不足道。在这种情况下，智能体有一条唯一的 s–t 路径，决策归结为检查智能体是否沿着这条路径行动，以及该路径是否包含 T 中的所有弧，无需进行任何修改。

另一方面， [Belova 等人 2024] 证明了， T-路径-删除问题——这是我们的 T-路径-编辑表述的一个特例——即使 p 任意 s–t 路径在 G 中的最大长度被限制为 8，仍然是 NP-困难的。这一结果立即意味着，相对于参数 p， T-路径-编辑是 Para-NP-困难的。

5 结论

我们提出并研究了 T-路径-编辑问题，这是一个用于设计结构性干预措施的通用框架，旨在引导时间不一致智能体完成关键任务。我们的模型扩展了 Kleinberg–Oren 框架，通过图编辑的最小化形式，明确形式化了委托人在影响智能体行为方面所发挥的作用。

我们的结果刻画了 T-路径-编辑在一系列结构参数下的可处理性边界，包括树宽、顶点覆盖、反馈顶点集和路径长度。具体而言，我们证明了：对于同时刻画图结构和成本多样性的组合参数，该问题具有固定参数可处理性；同时还通过 Para-NP-困难性和 W[1]-困难性建立了下界。

一个引人入胜的开放问题是，能否为可计算函数 f and g 确定一种 $\cdot\, m^{g(p)}$ 时间算法来解决 T-路径编辑问题，其运行时间为 f(tw)。

参考文献

[Akagi 等人， 2025] Akagi, Y.、Kim, H. 和 Kurashima, T.（2025）。一种适用于当下偏向型智能体的连续时间可处理模型。载于*AAAI-25，由人工智能促进协会赞助，2025*年*2*月*25*日至*3*月*4*日，美国宾夕法尼亚州费城，第 13510–13519 页。AAAI 出版社。[Akagi 等人， 2024] Akagi, Y.、Marumo, N. 和 Kurashima, T.（2024）。当下偏向下决策的解析可处理模型。载于第三十八届 *AAAI* 人工智能大会、*AAAI 2024*、第三十六届人工智能创新应用大会、*IAAI 2024*、第十四届人工智能教育进展研讨会、*EAAI 2014，2024*年*2*月*20*日至*27*日，加拿大温哥华，第 9441–9450 页。AAAI 出版社。[Akerlof， 1991] Akerlof, G. A.（1991）。拖延与服从。《美国经济评论：论文与会议录》，81(2):1–19。[Albers 和 Kraft， 2017] Albers, S. 和 Kraft, D.（2017）。时间不一致规划中惩罚的价值。载于第*44*届国际自动机、语言与编程研讨会（*ICALP*），第 10:1–10:12 页。[Albers 和 Kraft， 2019] Albers, S. 和 Kraft, D.（2019）。激励时间不一致智能体：一种计算方法。*Theory Comput. Syst.*，63(3):466–487。[Belova 等人， 2024] Belova, T.、Dementiev, Y.、Fomin, F. V.、Golovach, P. A. 和 Ignatiev, A.（2024）。如何引导当下偏向型智能体完成指定任务？载于*ECAI 2024*——第*27*届欧洲人工智能大会，*2024*年*10*月*19*日至*24*日，西班牙圣地亚哥-德孔波斯特拉——含第*13*届智能系统杰出应用大会（*PAIS 2024*），第392卷《人工智能与应用前沿》，第 3461–3468 页。IOS 出版社。[Brandst¨adt 等人， 1999] Brandst¨adt, A.、Le, V. B. 和 Spinrad, J. P.（1999）。《图类：综述》。工业与应用数学学会。[Burzyn 等人， 2006] Burzyn, P.、Bonomo, F. 和 Dur´an, G.（2006）。边修改问题的 NP-完全性结果。《离散应用数学》，154(13):1824–1844。[Cormen 等人， 2009] Cormen, T. H.、Leiserson, C. E.、Rivest, R. L. 和 Stein, C.（2009）。《算法导论》第*3*版。麻省理工学院出版社。[Crespelle 等人， 2023] Crespelle, C.、Drange, P. G.、Fomin, F. V. 和 Golovach, P. A.（2023）。参数化算法与边修改复杂性的综述。*Comput. Sci. Rev.*，48:100556。[Cygan 等人， 2015] Cygan, M.、Fomin, F. V.、Kowalik, L.、Lokshtanov, D.、Marx, D.、Pilipczuk, M.、Pilipczuk, M. 和 Saurabh, S.（2015）。《参数化算法》。施普林格。

[Demaine 等人， 2019] Demaine, E. D.、Eppstein, D.、Hesterberg, A.、Jain, K.、Lubiw, A.、Uehara, R. 和 Uno, Y.（2019）。无向路径重配置，第 353–365 页。施普林格国际出版。[Dementiev 等人， 2022] Dementiev, Y.、Fomin, F. 和 Ignatiev, A.（2022）。不一致规划：犹豫不决时，掷个硬币！《第36届AAAI人工智能会议论文集》（AAAI），36(9):9724–9731。[Fomin 和 Strømme， 2020] Fomin, F. V. 和 Strømme, T. J. F.（2020）。时间不一致规划：难以找到简单的激励方式。载于《第34届AAAI人工智能会议论文集》（AAAI），第 9843–9850 页。AAAI 出版社。[Frederick等人， 2002] Frederick, S.、Loewenstein, G. 和 O'Donoghue, T.（2002）。时间折扣与时间偏好：批判性综述。《经济文献杂志》，40(2):351–401。[Gravin 等人， 2016] Gravin, N.、Immorlica, N.、Lucier, B. 和 Pountourakis, E.（2016）。可变现在偏差下的拖延。载于 ACM经济学与计算会议（EC），第 361 页。[Halpern 和 Saraf， 2023] Halpern, J. Y. 和 Saraf, A.（2023）。为当下偏向型智能体进行任务分块。载于第24届 ACM经济学与计算会议（EC）论文集，第 853–884 页。[Kleinberg and Oren， 2014] Kleinberg, J. M. 和 Oren, S.（2014）。时间不一致规划：行为经济学中的一个计算问题。载于 ACM经济学与计算会议（EC），第 547–564 页。[Kleinberg and Oren， 2018] Kleinberg, J. M. 和 Oren, S.（2018）。时间不一致规划：行为经济学中的一个计算问题。Commun. ACM，61(3):99–107。[Kleinberg 等人， 2016] Kleinberg, J. M.、Oren, S. 和 Raghavan, M.（2016）。当下偏向下复杂代理的规划问题。载于 ACM经济学与计算会议（EC），第 343–360 页。[Kleinberg等人， 2017] Kleinberg, J. M.、Oren, S. 和 Raghavan, M.（2017）。具有多重偏差的规划。载于 ACM经济学与计算会议（EC），第 567–584 页。[Kleinberg 和 Tardos， 2006] Kleinberg, J. M. 和 Tardos, E.（2006）。算法设计。Addison-Wesley。[Laibson， 1997] Laibson, D.（1997）。金蛋与双曲线折扣*。《经济学季刊》，112(2):443–478。[Laibson， 1994] Laibson, D. I.（1994）。双曲线折扣与消费。博士论文，麻省理工学院经济学系。[Maniu 等人， 2019] Maniu, S.、Senellart, P. 和 Jog, S.（2019）。真实世界图数据树宽的实验研究。载于第22届国际数据库理论会议（ICDT 2019），2019年3月26日至28日，葡萄牙里斯本，LIPIcs 第127卷，第 12:1–12:18 页。Schloss Dagstuhl - Leibniz-Zentrum f¨ur Informatik。[McClure 等人， 2004] McClure, S. M.、Laibson, D. I.、Loewenstein, G. 和 Cohen, J. D.（2004）。不同的神经系统对即时与延迟货币奖励进行估值。《科学》，306(5695):503–507。[Meyer 等人， 2022] Meyer, S. A.、Pomplun, J. 和 Schill, J.（2022）。部分复杂代理和辅助代理中的当下偏向。《数学社会科学》，118:36–47。[Natanzon 等人， 2001] Natanzon, A.、Shamir, R. 和 Sharan, R.（2001）。若干边修改问题的复杂性分类。《离散应用数学》，113(1):109–128。[Oren 和 Soker， 2019] Oren, S. 和 Soker, D.（2019）。针对当下偏向型智能体的委托—代理问题。载于第12届国际算法博弈论研讨会（SAGT）论文集，第 237–251 页。施普林格。[Papadimitriou 和 Steiglitz， 1982] Papadimitriou, C. H. 和 Steiglitz, K.（1982 年）。组合优化：算法与复杂性。Prentice-Hall。[Samuelson， 1937] Samuelson, P. A.（1937）。效用测量札记。《经济研究评论》，4(2):155–161。[Stavrinides and Zuev， 2023] Stavrinides, P. 和 Zuev, K.（2023）。用于分析和理解学术课程体系的课程先修关系网络。《应用网络科学》，8。[Tang 等人， 2017] Tang, P.、Teng, Y.、Wang, Z.、Xiao, S. 和 Xu, Y.（2017）。时间不一致规划中的计算问题。载于第31届人工智能会议（AAAI）论文集。AAAI 出版社。[Yang 等人， 2024] Yang, B.、Gharebhaygloo, M.、Rondi, H.、Hortis, E.、Lostalo, E.、Huang, X. 和 Ercal, G.（2024）。五所中西部公立机构课程先修关系网络的比较分析。《应用网络科学》，9。

A 附录. 主要结果

本节中，我们证明定理 1。我们需要引入规范树分解的概念。关于树分解以及基于树分解的动态规划的全面介绍，请参阅 Cygan 等人的著作[2015]。

定义 1（优美树分解）。 图 $G = (V, E)$ 的一个优美树分解 是一种特殊形式的树分解 $(\mathcal{T}, \{(B_a, E_a)\}_{a \in V(\mathcal{T})})$，其满足：

- 树 $\mathcal{T}$有根，即我们指定 $\mathcal{T}$中的一个顶点 a_0 作为 $\mathcal{T}$的根。它的包为空：$B_{a_0} = \emptyset$；并且它的边集合是完整的：$E_{a_0} = E(G)$。每个节点 a 都属于以下五种类型之一：
  - 叶节点：没有子节点，且其包和边集均为空： $B_a = E_a = \emptyset$。 - 引入顶点节点：有一个子节点 b，其包增加一个顶点： $B_a = B_b \cup v$，其中 $v \notin B_b$，且$E_a = E_b$。 - 遗忘节点：有一个子节点 b，其包移除一个顶点： $B_a = B_b \setminus v$，其中 $v \in B_b$，且$E_a = E_b$。 - 连接节点：有两个子节点 b_1 和 b_2，满足 $B_a = B_{b_1} = B_{b_2}$ 和 $E_a = E_{b_1} \cup E_{b_2}$ - 引入边节点：有一个子节点 b， $B_a = B_b$，且存在一条边 $uv \in E \setminus E_b$，满足 $E_a = E_b \cup \{uv\}$。

- 这些包 B_a 仍满足标准的树分解性质：
  - 对于每个顶点 $v \in V(G)$，节点集合 $\{a \in V(\mathcal{T}) \mid v \in B_a\}$ 构成 $\mathcal{T}$的一棵连通子树。 - 对于每条边 $(u, v) \in E(G)$，存在一个包 B_a 使得 $u, v \in B_a$。

图的树宽是其优美树分解可能达到的最小宽度。

定义 2. 优美树分解 $(\mathcal{T}, \{(B_a, E_a)\}_{A \in \mathcal{T}})$ 的宽度 定义为

$$\max_{a \in \mathcal{T}} |B_a| - 1,$$

也就是说，比最大包的大小少 1。

尽管计算图的树宽是 NP-困难的，但我们能够高效地找到宽度较小的规范树分解。

命题 2 ([Cygan 等人， 2015])。宽度为 $\mathcal{O}(\mathrm{tw})$ 的优美树分解最多包含 $m^{\mathcal{O}(1)}$ 个节点，并可在 $2^{\mathcal{O}(\mathrm{tw})} \cdot m^{\mathcal{O}(1)}$ 时间内计算出来。

为了让智能体遍历 T 的所有弧，我们将迫使它不沿任何 T-避让弧前进。

定义 3（T-避让弧）。如果存在一条弧 $x \to y$ 属于 T，且以下条件全部成立，那么我们称 $G + A$ 中一条从 u 到 v 的弧 e 是 T-避让的：

1. $\{u, v\} \neq \{x, y\}$；2. x 在 $G + A$ 中从 u 可达。3. x 在 $G + A$ 中从 v 不可达。

显然，如果一条 s-t 路径不包含任何 T-避让弧，那么它就会遍历 T 的所有弧。

我们即将证明定理 1。在给出其余辅助结果和证明本身之前，为方便起见，我们先在此重述定理 1。

定理 1. T-路径-编辑 可在 $|L|^{\mathcal{O}(\mathrm{tw})} \cdot m^{\mathcal{O}(1)}$ 时间内运行的算法，其中 tw 等于 $G + A$ 的树宽，并且

$$L = \left| \bigcup_{v \in V(G)} \{w(P) \mid P \textit{ is a } (v, t)\textit{-path in } G + A\} \right|.$$

我们注意到，我们的算法会进行所有必要的弧删除和添加，以强制具有任意弧优先级的智能体遍历 T 的所有弧。也就是说，如果智能体在某个顶点有两条感知成本相同的弧，它可能会选择其中任意一条，因此我们应确保任何 智能体可能走的路径都会遍历 T 的所有弧。

我们要求 在我们的算法中进行显式构造。我们证明， L 可以在多项式时间内计算 e.

引理 2. 给定一个 T-PATH-EDITING 实例，可以在 $(|L|+m)^{\mathcal{O}(1)}$ 时间内计算 L .

证明。 我们通过以下递归关系计算 $L(v)$，它等于每个 $v \in V(G)$的所有可能 v-t 路径成本的集合:

$$L(u) = \bigcup_{e \in E(G+A), e:u \to v} \{w(e) + \ell \mid \ell \in L(v)\},$$

以及 $L(t) = \{0\}$. 。我们可以按照 $G + A$ 的拓扑序进行计算，其中 t 是第一个顶点，而 s 是最后一个顶点。

对于每个顶点，计算所需的运行时间至多为 $m \cdot |L| \cdot |L|$ 。总运行时间关于 $m + |L|$ 为多项式时间。

□

我们现在可以证明主要结果了。

定理 *1* 的证明。 我们描述一种具有所需运行时间的算法。

我们假设给定一个 T-PATH-EDITING 实例，其中 $|L| > 1$；否则 $|L| = \{0\}$，也就是说，每条弧的成本均为 0，且该问题可以在多项式时间内求解。我们还假设每个节点都能从 s 在 $G + A$ 中到达，并且 t 能从 $G + A$ 中的每个节点到达。

然后，在 $2^{\mathcal{O}(\mathrm{tw})} \cdot m^{\mathcal{O}(1)}$ 运行时间内计算出宽度为 $\mathcal{O}$(tw) 的 $\mathcal{T}$ $G + A$ 的优美树分解。我们假设 $G + A$ 的每条弧都对应一个引入边节点。也就是说，如果 $G + A$ 中 u 和 v 之间有 z 条平行弧，那么 $\mathcal{T}$ 中就有 z 个对应的引入边节点。

此外，我们还修改 $\mathcal{T}$，将 s，$t \in B_a$ 放入每个节点 $a \in V$（$\mathcal{T}$）的包中，并使根节点 a_0 的包B_{a_0} 恰好由 s 和 t 构成，也就是说，$B_{a_0} = \{s, t\}$。我们还强制要求优美树分解不引入或遗忘 s 或 t。也就是说，$\mathcal{T}$ 的叶节点也具有 $B_a = \{s, t\}$。$\mathcal{T}$ 的宽度仍然有界于 $\mathcal{O}$(tw)。

有了优美树分解 $\mathcal{T}$后，我们计算

$$\mathrm{OPT}(a, D, d, R, F_0, F_1, f),$$

其中

- a 是 $\mathcal{T}$的一个节点。针对$\mathrm{OPT}(a,\ldots)$ 的弧编辑搜索空间为 E_a。
- D 是一个从 B_a 到 $L \cup \{+\infty\}$的映射。</style}直观地说，(v)存储了解图中从 v 到 t 的距离。
- d 是一个从 B_a 到 $\{0, 1\}$的映射。直观地说，如果我们确信最短 v-t 路径的第一条弧属于解图，那么 $d(v) = 1$ 。
- R 是 B_a的子集。 R （可达顶点的集合）的直观含义是，它由智能体在解图中能够到达的所有顶点组成。
- F_0和 F_1 是从 R 到 L的映射。这些具有以下含义：如果智能体位于解图中的 $v \in R$，那么解图中存在一条 v-t 路径（其第一条弧为某个 $x \in V(G)$），智能体可以沿着这条路径前进；该路径的第一条弧成本为$F_0(v) - F_1(v)$，且从 x 到 t 的距离等于 $F_1(v)$。
- f 是一个从 R 到 $\{0, 1\}$的映射。对于 F_0、F_1，它发挥着与 d 对于 D相似的作用。直观地说，如果我们确定从 v 出发、与 $F_0(v)$ $F_1(v)$对应的弧存在于解图中，那么 $f(v) = 1$ 。

形式化地说，

$$\mathrm{OPT}(a, D, d, R, F_0, F_1, f)$$

等于初始弧集 $E_a \setminus A$ 中所需的最少弧编辑次数（如果不存在这样的编辑序列，则为 $+\infty$）；其中允许添加来自 $E_a \cap A$ 的弧，使得最终得到的集合 $E'_a \subset E_a$ 满足以下八个条件。下文中我们使用

$$\mathrm{dist}'(u,t) = \min_{v \in B_a \setminus \{u\}} \left\{ \mathrm{dist}_{E'_a}(u,v) + D(v) \right\},$$

若 $d(u) = 0$，且

$$\mathrm{dist}'(u,t) = D(u),$$

若 $d(u) = 1$。

1. （距离为最短距离）。

   对于每个 $u \in B_a$,

   $$D(u) \leq \min_{v \in B_a \setminus \{u\}} \left\{ \mathrm{dist}_{E'_a}(u,v) + D(v) \right\}.$$

2. （距离为实数）。

   对于每个 $u \in B_a$，若 $d(u) = 0$，则

   $$D(u) = \mathrm{dist}'(u,t).$$

3. （跟随弧由智能体跟随）。

   对于每个带有 $u \neq t$ 的 $u \in R$，对于从 u 出发并指向某个 $v \in V$（G）的每条边 $e \in E'_a$，

   $$(F_0(u) - F_1(u)) + \beta \cdot F_1(u) \leq w(e) + \beta \cdot \mathrm{dist}'(v,t).$$

4. （R 未被智能体放弃）。

   对于满足 $u \neq t$ 的每个 $u \in R$，

   $$(F_0(u) - F_1(u)) + \beta \cdot F_1(u) \leq \beta \cdot r.$$

5. （跟随弧是真实的）。

   对于满足 f（u）$= 0$ 的每个 $u \in R$，存在一条从 u 出发并指向某个 $v \in V$（G）的 $e \in E'_a$，使得

   $$F_1(u) = \mathrm{dist}'(v,t) \text{ and } F_0(u) = w(e) + F_1(u).$$

6. （R 可从 R到达）。

   对于每个 $u \in R$，使得 $f(u) = 0$，都存在 $v \in R$和一条 u-v路径 $P' \subset E'_a$，其弧为 e_1，e_2、.....、e_p，顶点为 $u = u_0$，u_1、.....、$u_p = v$，使得对于每个 $i \in [p]$,

   $$w(e_i) + \beta \cdot \mathrm{dist}'(u_i,t) \leq w(e') + \beta \cdot \mathrm{dist}'(x,t),$$

   对于每条从 u_{i-1} 出发、通向某个 $x \in V(G)$的弧 $e' \in E'_a$，并且

   $$w(e_i) + \beta \cdot \mathrm{dist}'(u_i,t) \leq \beta \cdot r.$$

7. （任意智能体路径都会经过 T）。

   对于每个 u，$v \in R$，以及满足前一点两个条件的每条 u-v路径 $P' \subset E'_a$，对于每个 $i \in [p]$，e_i 都不是一条避开 T的弧。

8. （智能体无法沿着一条从 R出发的弧前进）。

   对于每个 $u \in R$ 和 $v \notin R$，对于每条具有弧 $P' \subset E'_a$、e_1，e_2、.....、e_p 以及顶点 $u = u_0$，u_1、.....、$u_p = v$ 的 u-v 路径，都存在 $i \in [p]$，使得

   $$w(e_i) + \beta \cdot \mathrm{dist}'(u_i,t) > w(e') + \beta \cdot \mathrm{dist}'(x,t),$$

   其中，$e' \in E'_a$ 是一条从 u_{i-1} 出发、通向某个 $x \in V$（G）的弧。

如果最优解图 G^* 存在，且智能体在 G^* 中的一条路径为 P，该路径以 e_1 从 s 到某个$u_1 \in V(G)$的弧开始，那么对于 $\ell^* =$距离$_{G^*}(s,\ t)$、$\ell_1^* =$距离$_{G^*}(u_1,\ t)$、$\ell_0^* = w(e_1) + \ell_1^*$和

- $R^* = \{s,t\}$;

- $D^* = \{\ell^*, 0\}$;
- $d^* = f^* = \{s \to 0, t \to 1\}$;
- $F_0^* = \{s \to \ell_0^*, t \to 0\}$;
- $F_1^* = \{s \to \ell_1^*, t \to 0\}$;

OPT(a_0, D^*, d^*, R^*, F_0^*, F_1^*, f^*) 等于获得 G^* 所需的弧编辑次数。这直接由 OPT 的定义得出。

接下来，我们通过在 $\mathcal{T}$ 上进行动态规划来计算 OPT(a, D, d, R, F_0, F_1, f)（其中 a, D, d, R, F_0, F_1, f 的取值任意）。请注意，对于违反条件 4 的参数选择，OPT 的对应值为 $+\infty$。我们假设 OPT 的自变量选择始终满足条件 4。同样，我们也假设 s, $t \in R$、$D(t) = 0$ 和 $D(s) < +\infty$ 始终成立。

计算过程按照 $\mathcal{T}$ 的结构采用递归的自顶向下方法进行。根据树分解节点的类型，通过其子节点处理一个节点时会有几种情况。

遗忘节点。节点 a 恰好有一个子节点 b。于是，对于某个 $u \in V(G)$，有 $E_a = E_b$ 和 $B_a = B_b \setminus \{u\}$。根据定义，OPT($a$, D, d, R, F_0, F_1, f) 等于以下各项中的最小值

$\min_{\ell, \ell_0, \ell_1 \in L}$ OPT(b, $D \cup \{u \to \ell\}, d \cup \{u \to 0\}$, $R \cup \{u\}$, $F_0 \cup \{u \to \ell_0\}, F_1 \cup \{u \to \ell_1\}$, $f \cup \{u \to 0\}$),

对应于 u 可达的解，并且

$$\min_{\ell \in L \cup \{+\infty\}} \mathrm{OPT}(b, D \cup \{u \to \ell\}, d \cup \{u \to 0\}, R, F_0, F_1, f),$$

这对应于 u 不在任何智能体路径上的情况。

注意，第一部分中的 ℓ_0, ℓ_1 仅取满足 $(\ell_0 - \ell_1) + \beta \cdot \ell_1 \leq \beta \cdot r$（条件 4）的值。

引入顶点节点。节点 a 恰有一个子节点 b。那么，对于某个 $u \in V$（G），$E_a = E_b$ 和 $B_a = B_b \cup \{u\}$。此外，在 E_a 中，没有弧与 u 相连。根据定义，如果 d（u）$= 0$，则 OPT(a, D, d, R, F_0, F_1, f) 等于 $+\infty$；但由于对于任意选择的 E'_a 和 $v \in V$（G），dist′(u, v) $= +\infty$，因此 D（u）$\neq +\infty$。类似地，如果 $u \in R$，则它等于 $+\infty$，但 f（u）$= 0$。否则，它等于

$$\mathrm{OPT}(a, D \setminus \{u\}, d \setminus \{u\}, R, F_0, F_1, f),$$

如果 $u \notin R$，或者

$$\mathrm{OPT}(a, D \setminus \{u\}, d \setminus \{u\}, R \setminus \{u\}, F_0 \setminus \{u\}, F_1 \setminus \{u\}, f \setminus \{u\}),$$

如果 $u \in R$。

引入边节点。节点 a 恰好有一个子节点 b。那么，对于连接某对顶点 u, $v \in V(G)$ 的某条弧$e \in E(G) \cup A$，$B_a = B_b$ 和 $E_a = E_b \cup \{e\}$。它要么是 $e \in E(G)$，此时我们考虑在图中保留或删除e；要么是 $e \in A$，此时我们分别选择添加或跳过它。这些情况实际上相同，区别仅在于编辑代价：若解中包含 $e \in E(G)$，我们付出 0；若解中不包含 $e \in E(G)$，则付出 1。对于 $e \in A$，则分别付出 1 和 0。下文中，我们用 c_1 和 c_0分别表示解中包含或不包含 e 或 e 的成本。

如果 e 不属于 OPT(a, D, d, R, F_0, F_1, f) 的最优解，那么它等于

$$\mathrm{OPT}(b, D, d, R, F_0, F_1, f) + c_0, \tag{2}$$

由于解配置对于 b是相同的，但由于解中没有该边，我们需要付出一定代价。

另一种可能的情况是，e 属于该解。只有在以下条件下才可能：

- $D(u) \leq w(e) + D(v)$, and,

- $(F_0(u) - F_1(u)) + \beta \cdot F_1(u) \le w(e) + \beta \cdot D(v)$, and

若 $u \in R$但如果 $v \notin R$，或者 e 是 T-规避的，则最后一个不等式应为严格不等式（智能体不应遵循 e ）.
若将 e 纳入解中，则会对配置产生一些影响。若 e 满足 $D(u) = w(e) + D(v)$（条件 2），则我们可以为 b的配置传播 $d(u) = 1$ 。若 e 满足 $F_0(u) - F_1(u) = w(e)$以及 $F_1(v) = D(v)$（条件 5），则我们传播 $f(u) = 1$。将得到的映射记为 d'和 f'。那么（若应将 e纳入当前配置下的最优解），$\mathrm{OPT}(a, D, d, R, F_0, F_1, f)$等于

$$\mathrm{OPT}(b, D, d', R, F_0, F_1, f') + c_1. \tag{3}$$

因此，为了计算 $\mathrm{OPT}(a, D, d, R, F_0, F_1, f)$，我们只需取（2）和（3）中的较小值，其中仅当第二项被纳入考虑 当存在这条弧与配置相容时，才予以考虑，正如上一段所述。

叶节点。那么，a 在 $\mathcal{T}$、$E_a = \emptyset$ 和 $B_a = \{s, t\}$中没有子节点。要计算 $\mathrm{OPT}(a, D, d, R, F_0, F_1, f)$，我们只需检查 $d(s) = 1$ 和 $f(s) = 1$。如果两者都成立，我们就将值设为 0；否则，将 OPT 设为 $+\infty$ 。这与 OPT 的定义一致。

连接节点。这是 a 有两个子节点 b_1、b_2的唯一情况。此时，$B_a = B_{b_1} = B_{b_2}$，且 E_{b_1} 和 E_{b_2}构成 E_a的一个划分。要计算 $\mathrm{OPT}(a, D, d, R, F_0, F_1, f)$，我们必须将 d和 f拆分为两个独立的配置，并将 E_{b_1} 和 E_{b_2}中的编辑数相加。这是因为距离（D）或跟随弧（F_0、F_1）可以在 E_{b_1} 或 E_{b_2}中实现（OPT 定义中的条件 2 和 5），因此我们必须考虑所有可能的情况。于是，$\mathrm{OPT}(a, D, d, R, F_0, F_1, f)$等于

$$\min_{\substack{d^1 \cdot d^2 = d \\ f^1 \cdot f^2 = f}} \left[\sum_{j=1}^{2} \mathrm{OPT}(b_j, D, d^j, R, F_0, F_1, f^j) \right],$$

其中，d^1，d^2和 f^1，f^2分别遍历从 B_a和 R到 $\{0, 1\}$的所有映射。对于每个 $u \in B_a$，这些配对都应满足 $d^1(u) \cdot d^2(u) = d(u)$；对于每个 $u \in R$，还应满足 $f^1(u) \cdot f^2(u) = f(u)$。
树分解节点类型的各种情况已全部列举完毕。至此，算法的动态规划过程描述完毕。

根据动态规划值还原答案。为了求出弧编辑的最小数量，我们遍历所有 ℓ^*，ℓ_0^*，$\ell_1^* \in L$，并按照本证明开头的方式构造 D^*，d^*，R^*，F_0^*，F_1^*，f^*。弧编辑的最小数量等于

$$\min_{\ell^*, \ell_0^*, \ell_1^* \in L} \mathrm{OPT}(a_0, D^*, d^*, R^*, F_0^*, F_1^*, f^*).$$

按照常规方法，可以根据动态规划的状态转移还原编辑序列。

正确性。首先，$\mathrm{OPT}(a_0, D^*, d^*, R^*, F_0^*, F_1^*, f^*) \le k$，其中 k是获得解图 G^*所需的最少弧编辑次数。要说明这一点，可以从根状态 $\mathrm{OPT}(a_0, D^*, d^*, R^*, F_0^*, F_1^*, f^*)$ 开始执行自顶向下计算直到叶节点，并在每个节点作出与解图G^*相对应的选择。

其次，为证明 $\mathrm{OPT}(a_0, D, d, \{s, t\}, F_0, F_1, f) \ge k$，可以按照动态规划中自然的方式恢复弧编辑序列 $\mathrm{OPT}(a_0, D, d, \{s, t\}, F_0, F_1, f)$，并证明它构成了 T-PATH-EDITING 初始实例的一个解。事实上，恢复出的动态规划转移序列保证 $R \subset V(G)$ 构成智能体可达的顶点集合。不存在从 R到 $V(G) \setminus R$ 的弧，使得智能体能够通过遵循具有最小感知成本的弧来行动。另一方面，智能体从 v 到$u \in R$ 有一条弧，在每个顶点 $v \in R \setminus \{t\}$处都可以沿该弧行动。最后，不存在智能体可以遵循的 T-规避弧。

运行时间。要计算 OPT 的状态数量，请注意

- a 有 $|V(\mathcal{T})| = m^{\mathcal{O}}$ 个（1）种选项；
- R，d，f有 $2^{\mathcal{O}(\mathrm{tw})}$ 种选项；
- D 具有 $(|L| + 1)^{\mathcal{O}(\mathrm{tw})}$ 个选项；

- F_0，F_1 有 $|L|^{\mathcal{O}(\mathrm{tw})}$ 个选项。

OPT 的状态总数至多为 $(2|L|+2)^{\mathcal{O}(\mathrm{tw})} \cdot m^{\mathcal{O}(1)}$，其上界为 $|L|^{\mathcal{O}(\mathrm{tw})}$，因为 $|L| \geq 2$。接下来只需给出计算 OPT 各个值所需时间的上界。注意，对于每种节点类型，除连接节点外，计算所需时间均为关于 m 的多项式时间。在连接节点中计算 OPT$(a,\ D,\ d,\ R,\ F_0,\ F_1,\ f)$ 时，我们至多考虑 $(d^1,\ d^2)$ 的 $3^{|R|}$ 种选项，以及 $(f^1,\ f^2)$ 的 $3^{|R|}$ 种选项。其上界也为 $|L|^{\mathcal{O}(\mathrm{tw})}$，因为 $|L| \geq 2$。恢复动态规划转移所需的时间相同。

因此，总运行时间至多为 $|L|^{\mathcal{O}(\mathrm{tw})}$。证明完毕。 □