arXiv:2304.01958v1 [cs.DS] 4 Apr 2023

带预测的在线时间窗旅行商问题

Shuchi Chawla* Dimitris Christou†

2023年4月5日

摘要

在时间窗旅行商问题（TW-TSP）中，给定网络上不同位置的请求；每个请求都附有奖励和一个时间区间；目标是在相应时间窗内访问尽可能多的奖励。对于该问题的在线版本，由于每个请求要到其时间窗开始时才会公布，因此无法获得有限的竞争比。我们考虑该问题的一个版本：算法会获得关于在线请求将在何时何地出现的预测，但并不知道这些辅助信息的质量。

车辆路径问题（例如 TW-TSP）对输入中的错误或变化可能非常敏感，这是由严格的时间窗约束造成的；而不准确的预测能否用于获得有限的竞争比，目前尚不明确。我们证明，通过在解中明确预留松弛量，可以取得良好的性能。我们的主要结果是一种在线算法，其竞争比关于底层网络的直径呈对数级，并且与最佳离线算法的性能相差不超过取决于所提供预测质量的因子。竞争比会随预测质量平滑退化；我们还证明，这种依赖关系在常数因子范围内是紧的。

*德克萨斯大学奥斯汀分校，shuchi@cs.utexas.edu

†德克萨斯大学奥斯汀分校，christou@cs.utexas.edu

1 引言

许多优化问题在离线优化与输入以在线方式到达时的优化效果之间存在巨大差距。为了在在线场景中获得有意义的算法结果，一个自然的研究方向是考虑“超越最坏情况”模型：这些模型要么限制对手的能力，要么增强算法的能力。近期，该方向的一系列研究探讨了利用预测 来弥合离线与在线之间的差距。在此语境下，预测仅指关于输入的辅助信息，在线算法可以据此做出决策；真实输入仍由对手选择，并以在线方式到达。我们的目标是证明：一方面，如果预测与输入相符，算法的表现几乎可以达到离线场景中的水平（这一性质称为一致性）；另一方面，如果预测与输入完全无关，算法的表现仍几乎可以达到最佳在线算法的水平（亦称鲁棒性）。简而言之，好的预测应当有所助益，而糟糕的预测不应造成损害；理想情况下，我们无需事先了解预测的质量，也能从中获益。

研究表明，预测能够有效突破多种在线决策问题的下界，例如缓存 [LV21、Roh20、JPS20]，调度 [BFM+21、IKQP21、ALT21]，在线图算法 [APT21]，负载均衡 [LLMV20、LMRX21]，在线集合覆盖 [BMS20]，匹配问题 [DIL+21]、k均值 [EFS+22]、秘书问题 [AGKK20、DLLV21]，网络设计 [EdLMS22、XM22、BBFL23] 等。[1]

本文研究一个目标函数值对输入变化高度敏感的问题，这为预测设定带来了新的挑战。在带时间窗的旅行商问题（简称 TW-TSP）中，给定带权无向图上不同位置的一系列服务请求。每个请求都带有奖励，以及一个应在其中得到服务的时间窗。算法的目标是生成一条路径，使在各自时间窗内访问的请求所获总奖励最大化。在线场景中，请求在各自时间窗开始时逐个到达，算法必须在不知道未来请求的位置或时间窗的情况下，逐步构建路径。

涉及 TW-TSP 等对子路径长度（例如访问某个位置的时间）施加硬约束的车辆路径问题，通常比仅需最小化路径长度的问题更具挑战性。尤其是，算法开始时一个微小的错误决策，例如选择一条略非最优的路径前往第一个请求，就可能迫使算法错失此后所有奖励，从而彻底损害其表现。在线下场景中，这意味着近似算法必须谨慎地通过有意放弃其他路段中可获取的奖励，来抵消某些路段中的路径规划低效。而在线场景中，这意味着不可能实现次线性的竞争比。

鉴于 *TW-TSP* 的目标对细微的路径规划低效十分敏感，能否利用不准确的预测，为这一问题设计有实际意义的在线算法？

我们考虑这样一种模型：算法在开始时会获得一个请求预测序列，其中每个请求都附有一个预测位置和一个预测时间窗。与之前一样，真实请求序列会随着时间推移逐步揭晓。当然，如果预测序列与真实请求序列完全相同，算法就能达到最佳离线算法的性能。但如果预测存在轻微偏差呢？这些小误差会导致在线算法遭受巨大损失吗？算法能否容忍较大的偏差？

[1]关于这一主题的全面综述文献可见于 LM [.]。

我们的主要结果是一种基于预测的 TW-TSP 在线算法，其性能会随着预测误差的增大而平稳退化。我们通过在解和基准中明确预留松弛量，得到了这一结果。与先前关于 TW-TSP 的研究略有不同，我们要求服务器在每个已服务的请求处花费一个单位的空闲“服务时间”。我们证明，即使有预测，要获得次线性近似，这一要求也是必要的（定理 3.7）。不过，在没有预测的情况下，包含服务时间的设定仍然存在竞争比的线性下界；参见定理 3.6。随后，我们在规划路线时审慎利用服务时间，并将预测误差造成的延迟纳入考量。

预测误差有三个来源：（1）请求的预测位置可能与真实位置相距甚远；（2）预测时间窗可能与真实时间窗不同；（3）预测奖励可能与真实奖励不同。我们证明，我们算法的竞争比对第一项和第三项误差呈多项式依赖，并证明这种依赖在常数因子范围内是紧的。此外，只要时间窗的偏差不超过最小时间窗长度的一半，这项误差造成的性能损失至多再增加一个常数因子。除对预测误差的这种依赖外，我们算法的竞争比还对底层网络的直径呈对数依赖，这与目前已知的 TW-TSP 最佳离线算法的性能相当。

重要的是，我们的算法几乎不需要了解预测与真实请求之间的匹配情况。在分析中，我们通过某种预测请求与真实请求之间的匹配来衡量预测误差——误差参数据此定义为任一预测请求与其匹配的真实请求之间的最大不匹配程度。算法无法获知这种匹配；事实上，算法的性能取决于预测请求与真实请求之间可能达到的最佳匹配质量。算法唯一需要了解的预测质量信息是位置误差，即任一预测与其匹配的真实请求之间的最大距离。即使对于这一参数，给出一个上界也已足够（再付出少量额外损失时，给出一个估计值也足够）。

我们的整体方法包含几个组成部分。首先，我们针对预测请求构造一个 TW-TSP 实例，要求服务器在每个请求处留出一段空闲时间，作为“服务延迟”。随后，我们将离线 TW-TSP 算法扩展到这种服务延迟场景，从而得到一个近似比关于直径呈对数级的算法。接着，我们在在线场景中跟随并调整这一离线解。每当离线解服务一个预测请求时，我们就将该请求与一个此前已揭示的真实请求进行匹配，偏离计算出的路径，绕行访问并服务该真实请求，然后继续沿预先计算的路径前进。综上，这种方法实现了所需的竞争比。

我们的结果还可推广到预测较为粗略的场景：每个预测位置对应多个彼此邻近的潜在真实请求。我们表明，只要适当定义预测误差，就能在这种“多对一匹配”场景中再次实现竞争比，该竞争比关于图的直径呈对数级，关于预测误差呈多项式级。

进一步相关工作

在在线算法设计领域中使用预测，最早是由 [LV21] 针对广为研究的缓存问题提出的。自那项工作以来，关于带预测的在线算法设计的文献迅速增长。感兴趣的读者可参阅 [LM] 汇编的参考资料。

带离群值的度量误差。 Azar 等人 [APT21] 开启了在在线图优化问题中研究预测的先河，并提出了一种用于量化预测误差的框架，称为带离群值的度量误差，我们对其进行了改编。该框架旨在捕捉两种误差来源：（1）某些真实请求可能未被预测涵盖，而某些预测可能不对应任何真实请求；（2）对于被预测涵盖的请求，预测可能并不完全忠实或准确。关键在于，可以设计出其性能取决于这两种误差来源的算法，而无需事先明确了解预测请求与真实请求之间的（部分）对应关系。

在这项工作中，我们主要关注第二种误差来源。为更细致地理解竞争比与不同误差类型之间的关系，我们将其进一步细分为三类误差。与 Azar 等人 [APT21], 的工作一样，我们假设算法不会获知预测请求与真实请求之间的对应关系。尽管如此，算法的性能仍取决于预测请求与真实请求之间最佳匹配的误差。在第3 节中，我们将说明如何把 Azar 等人框架中的第一种误差来源也纳入我们的界限。

带预测的旅行商问题。最近，有几篇论文 [BLMS+22、HWL+22、GLS22]研究了带预测的在线旅行商问题及相关路径规划问题。在线旅行商问题的输入与我们的问题类似：请求随时间在图上到达，且巡回路线必须在每个请求到达后访问它。然而，两者的目标不同。在我们的设定中，请求还有截止时间，算法不一定能访问所有请求，因此目标是尽可能多地访问请求。在线旅行商问题没有截止时间，所以目标是尽可能快地访问所有请求，换言之，最小化完工时间。与截止时间设定相比，这种完工时间最小化目标通常容易得多：在离线、在线和预测设定下，该问题已有常数因子近似算法，而后者问题只能达到对数级或更差的近似算法。

无预测的 TW-TSP。关于（离线）TW-TSP 问题已有大量研究，研究历史超过 20 年。已知即便在一些特殊情形下，该问题也是 NP 困难的，例如在线段 [Tsi92], 上，以及所有请求具有相同的释放时间和截止时间时（即定向越野问题） [BCK+03]。定向越野问题存在常数因子近似算法 [BCK+03, BBCM04, CKP08], ，而当请求位于固定维数的欧几里得空间中时，甚至存在多项式时间近似方案 [AMN98, CHP06]。对于一般时间窗，目前仅在某些特殊情形下已知存在常数因子近似算法：例如，不同时间窗的数量为常数时 [CK04]；以及在路径图上 [Tsi92, KN03, BYES05, GJMZ16]。对于一般图和时间窗，目前已知的最佳近似算法的近似比随输入参数呈对数增长 [BBCM04, CKP08]。

据我们所知，Azar 和 Vardi [AV15]仅研究过 TW-TSP 的在线场景。Azar 和 Vardi 假设服务时间非零，并在最短时间窗长度 $L_{\min}$ 与图的直径 D 相当的前提下，提出了竞争算法；若 $L_{\min} < D/2$，则无法实现次线性的竞争比。借助预测，我们能够突破这一下界。

论文结构

我们在第2节正式定义问题及我们的误差模型。第3 节介绍我们的结果，并概述我们的分析思路。我们所有的主要结果都在该节中给出。这些结果的证明见后续各节。第4、5和6 节详细阐述我们的上界，第7 节证明下界，第8 节分析我们的主要结果向以下内容的扩展：

上文所述的“多对一匹配”设置。正文中略去的证明见附录。

2 定义

2.1 带时间窗的旅行商问题

TW-TSP 的一个实例由一个网络 G 和一个（有限的）服务请求序列 I组成。$G=(V、E、\ell)$ 是一个边长为 $\{\ell_e\}_{e\in E}$ 的无向网络。将距离的概念扩展到 G 中的所有顶点对，我们定义，对于 $u、v\in V$，$\ell(u、v)$ 为从 u到 v 的最短路径长度。我们不失一般性地假设 G 是连通的，且边长 ℓ_e 为整数。一个服务请求 $\sigma=(v_\sigma、r_\sigma、d_\sigma、\pi_\sigma)$ 由以下部分组成：请求到达的顶点 $v_\sigma\in V$、释放时间 $r_\sigma\in\mathbb{Z}^+$、满足 $d_\sigma>r_\sigma$ 的截止时间 $d_\sigma\in\mathbb{Z}^+$，以及奖励 $\pi_\sigma\in\mathbb{Z}^+$。我们使用 $\Sigma\subseteq V\times\mathbb{Z}^+\times\mathbb{Z}^+\times\mathbb{Z}^+$ 表示所有可能的客户请求集合，并使用 $I\subset\Sigma$ 表示算法收到的请求集合。

TW-TSP 的解是一个连续的 游走 在 G 上，并且允许在图的顶点上停留不动。[2] 形式上，该游走从时间 $t=0$的某个顶点出发；在它占据顶点 $u\in V$的每个时间步，它可以选择在 u 上停留若干个时间步，或者 花费时间 $t=\ell_{(u,v)}$移动到某个 $v\in V$；需要说明的是，路径处于转移过程中时，不能再做出任何决策。请注意，这形成了离散时间范围的概念，这对于将问题的在线变体形式化至关重要。

我们用 $\mathcal{W}(G)$ 表示 G 上所有游走的集合。给定一个请求 $\sigma\in\Sigma$，如果游走 W 覆盖 该请求，即 W 从某个时间步 $\tau\in[r_\sigma,d_\sigma-1]$开始，在顶点 v_σ 上至少停留一个时间步，[3] 。对于请求序列 $I\subset\Sigma$，我们用 $\mathrm{Cov}(W,I)\subseteq I$表示 I 中被 W覆盖的请求集合。于是，W 获得的奖励记为 $\mathrm{Rew}(W,I):=\sum_{\sigma\in\text{覆盖 }(W,I)}\pi_\sigma$。TW-TSP 的目标是计算一个奖励最大的游走 $W\in\mathcal{W}(G)$。我们将其记为 $\mathrm{Opt}(G,I)$:

$$\textsc{Opt}(G,I):=\max_{W\in\mathcal{W}(G)}[\mathrm{Rew}(W,I)]$$

2.2 离线、在线和预测设定

我们假设，在所考虑的所有设定中，算法都预先知道网络 G。在问题的**离线** 版本中，请求序列 I 会提前提供给算法。在**在线** 版本中，请求 $\sigma\in I$ 以在线方式到达；具体而言，每个请求 $\sigma\in I$ 都会在其释放时间 r_σ告知算法。

在**预测** 场景中，真实请求序列 I 与在线场景一样在线到达。不过，算法还会提前获得预测序列 $I'\subset\Sigma$，其中每个请求 $\sigma'\in I'$ 都包含一个位置、一个时间窗和一项奖励。预测质量以预测与真实请求的接近程度来衡量。为此，我们定义三种不匹配度或误差。对于真实请求 σ 和预测请求 σ'，位置误差、时间窗

[2]为使叙述简洁，我们不指定游走的起始位置。不过，如第 3 节末尾所述，我们的所有算法都可以无损地调整，以适用于起始位置固定的情形。[3]正如我们在引言中提到的，即使有预测，要使在线 TW-TSP 达到任何次线性近似，也必须要求最短一单位服务时间。参见第 7 节中的定理 3.

误差和奖励误差定义如下：

$$\begin{aligned}\mathrm{LocErr}(\sigma,\sigma') &:= \ell(v_\sigma, v_{\sigma'})\\ \mathrm{TWErr}(\sigma,\sigma') &:= \max\{|r_\sigma - r_{\sigma'}|, |d_\sigma - d_{\sigma'}|\}\\ \mathrm{RewErr}(\sigma,\sigma') &:= \max\{\pi_\sigma/\pi_{\sigma'}, \pi_{\sigma'}/\pi_\sigma\}\end{aligned}$$

我们将这些定义扩展到整个序列 I 和 I'，通过在两组列表中的请求之间建立一个底层匹配（但算法并不知道该匹配）：

定义 2.1。给定两个请求序列 $I, I' \subset \Sigma$，其中 $|I| = |I'|$，以及一个完美匹配M： $I \mapsto I'$，我们将匹配 M 的位置误差、时间窗误差和奖励误差定义如下：

$$\Lambda_M := \max_{\sigma\in I} \text{位置误差}(\sigma, M(\sigma)), \quad \tau_M := \max_{\sigma\in I} \mathrm{TWErr}(\sigma, M(\sigma)), \ \textit{and} \quad \rho_M := \max_{\sigma\in I} \mathrm{RewErr}(\sigma, M(\sigma))$$

我们用 $n = |V|$ 表示 G中的顶点数，用 D 表示图的直径，并用 $L_{\min}$ 和 $L_{\max}$ 分别表示给定实例中最小和最大的时间窗大小（针对真实请求或预测请求）： $L_{\min} = \text{最小值}_{\sigma\in I\cup I'} |d_\sigma - r_\sigma|$ 和 $L_{\max} = \text{最大值}_{\sigma\in I\cup I'} |d_\sigma - r_\sigma|$。我们所开发算法的竞争比取决于这些参数。

已知与未知。我们将一个带预测的 TW-TSP 实例表示为 (G, I, I', M)。实例的所有组成部分均由对手选择。如前所述，网络 G 和预测序列 I' 在算法开始时提供给算法。序列 I 在线到达。我们假设算法不会直接获知匹配 M，但会获得误差 Λ_M 的上界。我们还假设算法知道参数$L_{\min}$；这并不失一般性，因为可以根据预测以常数因子精度推断该参数。[4]

2.3 带服务时间的 TW-TSP

从整体上看，我们的算法由两个组件构成：一个离线组件，用于计算一条经过请求预测位置的高奖励游走；以及一个在线组件，它基本沿着这条游走前进，同时通过“绕行”覆盖到达的真实请求序列。具体而言，算法沿离线游走前进时，对于每个访问到的预测位置，只要附近有可处理的真实请求，算法就会“绕行”前往该真实请求，随后返回预测位置并继续剩余的游走。为了在计算离线游走时计入这些绕行所花费的时间，我们要求游走在其覆盖的每个预测位置停留一定的“服务时间”。据此，我们定义 TW-TSP 的一种推广：

定义 2.2。带服务时间的TW-TSP （*TW-TSP-S*）以网络 G、服务请求序列 I 和服务时间 $S \in \mathbb{Z}^+$ 为输入，并返回一条游走 $W \in \mathcal{W}(G)$。如果 W 从某个时间步 $t \in [r_\sigma, d_\sigma - S]$ 开始，在顶点 v_σ 上至少闲置 S 个时间步，则称其覆盖请求 $\sigma \in I$，记作 $\sigma \in \mathrm{Cov}(W, I, S)$。我们将 W 的奖励定义为$\mathrm{Rew}(W, I, S)$: $= \sum_{\sigma\in\mathrm{Cov}(W,I,S)}$ *{v47}*。该实例的最优值为：

$$\textsc{Opt}(G, I, S) := \max_{W\in\mathcal{W}(G)} [\mathrm{Rew}(W, I, S)]$$

请注意，如前所定义的原始版本 TW-TSP，只是服务时间为 $S = 1$ 的 TW-TSP-S 特例；特别地，我们有 Rew(W, I) = Rew(W, I, 1)，以及 Opt(G, I) = Opt(G, I, 1)。

[4]特别地，假设 $\tau_M \le L_{\min}/2$，这是我们的结果成立所必需的，那么任何真实请求的时间窗都不会短于任何预测请求的最小时间窗的一半。

3 我们的结果与方法概述

我们的主要结果如下。

定理 3.1。 给定任意一个(G, I, I', M)带预测的时间窗旅行商问题实例，其误差满足$\tau_M \le L_{\min}/2$和 $\Lambda_M \le (L_{\min}-1)/4$，则存在一个多项式时间在线算法，以元组 (G, I', Λ_M)作为离线输入，以 I作为在线输入，并构造游走 $W \in \mathcal{W}(G)$，使得

$$\mathbb{E}[\mathrm{Rew}(W, I)] \ge \frac{1}{O(\Lambda_M \cdot \rho_M^2 \cdot \log \min(D, L_{\max}))} \cdot \textsc{Opt}(G, I)$$

如前所述，我们的算法由两个组件构成。离线组件借助预测请求，在网络中构造一条潜在游走。然后，在线组件会调整这条游走，以覆盖逐一到达的真实请求。我们将算法的设计与分析分为四个步骤。前两个步骤将我们求解的离线实例与在线实例的事后最优解联系起来。第三步则将离线近似应用于具有适当服务时间的预测实例。最后一步处理如何在线调整游走，以应对到达的请求。

下面四个引理概括了这四个步骤。首先，我们证明（第 4 节）引入服务时间 S 对最优值造成的损失至多为 $2S-1$ 倍。正如我们在第 4.1 节的引理 4 中所证明的，这种对 S 的依赖是紧的。注意，我们要求 $S \le L_{\min}$，因为要使巡回路线能够覆盖某个请求，该请求的服务时间必须落在其时间窗内。

引理 3.2。 对于任意一个具有服务时间的 *TW-TSP* 实例 (G, I) 以及任意整数 $S \le L_{\min}$，有

$$\textsc{Opt}(G, I, S) \ge \frac{1}{2S-1} \cdot \textsc{Opt}(G, I, 1).$$

我们的第二个步骤（同样在第 4 节）将真实请求上的最优解值 I 与预测序列上的最优值 I' 联系起来。在这两种情况下，我们都会施加一些服务时间要求。请注意，这一论证需要考虑真实请求和预测请求在位置、时间窗以及奖励方面的差异。

引理 3.3。 设 (G, I, I', M)是一个带预测的时间窗旅行商问题实例，其中 Λ_M、 ρ_M 和τ_M 分别表示该实例的位置、奖励和时间窗的最大误差。定义$S := 4\Lambda_M + 1$ 以及 $S' := 2\Lambda_M + 1$。那么，若 $\tau_M \le L_{\min}/2$ 且 $\Lambda_M \le (L_{\min}-1)/4$，则有

$$\textsc{Opt}(G, I', S') \ge \frac{1}{3\rho_M} \cdot \textsc{Opt}(G, I, S).$$

我们的第三步（第 5 节）体现了算法中的离线组件：针对预测请求及其指定的服务时间，计算近似最优游走。为此，我们借鉴了关于不含服务时间的 TW-TSP 的已有研究，并说明如何对其进行调整，以满足服务时间要求。

引理 3.4。 对于任意一个带服务时间的 *TW-TSP* 实例 (G, I', S')，存在一个多项式时间算法，可返回一条游走 $W \in \mathcal{W}(G)$，其奖励为

$$\mathrm{Rew}(W, I', S') = \frac{1}{\mathcal{O}\left(\log \min(D, L_{\max})\right)} \cdot \textsc{Opt}(G, I', S').$$

最后，第四个组成部分（第 6 节）讨论了我们算法的在线部分。给定一条基于预测请求序列计算出的游走，该部分通过求解一个适当的在线匹配问题来构造绕行，以捕获真实请求。与引理 3.3 一样，这部分同样需要考虑真实请求与预测请求在位置、时间窗以及奖励方面的差异。

引理 3.5。 给定一个$(G,\ I,\ I',\ M)$的带预测的时间窗旅行商问题（*TW-TSP*）实例，满足 $\tau_M \le L_{\min}/2$且 $\Lambda_M \le (L_{\min}-1)/4$；一条游走 $W' \in \mathcal{W}(G)$；以及任意整数 $S' \ge 2\Lambda_M + 1$，存在一种在线算法（算法 *1*），返回一条游走 $W \in \mathcal{W}(G)$，其期望奖励为

$$\mathbb{E}\left[\mathrm{Rew}(W, I, 1)\right] \ge \frac{1}{6\rho_M} \cdot \mathrm{Rew}(W', I', S').$$

定理3.1 可由引理3.2、3.3、3.4 和3.5立即推出。

我们的结果的下界与紧确性。

我们证明，若 $L_{\min} < D$，则在没有预测的情况下，在线时间窗旅行商问题不存在次线性竞争算法，即使服务时间非零也如此。此外，如果所有服务时间都为0，即使使用除了请求位置之外各方面都准确的预测，也不可能实现次线性竞争比。因此，要实现非平凡的竞争比，必须使用预测，并对最优解施加非零服务时间的要求。证明见第 7 节。

定理 3.6。 对于在线时间窗旅行商问题中满足 $L_{\min} \le D$ 且所有服务时间均为1 的实例，任何随机在线算法的竞争比至多为 $1/n$。

定理 3.7。 对于任意 $S > 0$，存在一个(G, I, I', M) 形式的带预测的时间窗旅行商问题实例，其服务时间为0，且满足 $\tau_M = 0$、 $\rho_M = 1$和 $\Lambda_M = S$；对于任何将元组(G, I', Λ_M) 作为离线输入、将 I作为在线输入的随机在线算法，其获得的奖励不超过 $O(1/n)$ ·最优值$(G, I,0)$。这里， n 是 G中的顶点数。

如前所述，离线 TW-TSP 已知的最佳近似因子为 $O(\log L_{\max})$（我们证明可以将其略微改进为 $O(\log \min(D, L_{\max}))$）。在预测设定中，我们沿用了对 D 和 $L_{\max}$ 的这种对数依赖。此外，对离线近似的任何改进都会立即体现在我们的竞争比上。具体而言，给定一个竞争比为 $\alpha(D, L_{\max})$ 的离线 TW-TSP 算法，我们可以得到一个竞争比为 $O(\Lambda_M \cdot \rho_M^2 \cdot \alpha(D, L_{\max}))$ 的在线算法。

可以很容易地看出，我们的界对 ρ_M 的依赖是紧的——考虑一个星形图，请求位于叶节点上，并以一种特定方式设定边长和时间窗，使得任何可行行走至多覆盖一个请求。那么，即使预测在其他方面都完全准确，预测奖励中的 ρ_M 倍不确定性也可能迫使任何在线算法达到 $\Omega(\rho_M^2)$ 竞争比。最后，我们在第 7.2 节证明，我们的竞争比对Λ_M 的依赖同样是紧的：

定理 3.8. 对于任意 $S > 0$，都存在一个实例（G, I, I', M），它属于带预测的 *TW-TSP*，满足 $\tau_M = 0$、 $\rho_M = 1$和 $\Lambda_M = S$，使得任何将元组（G, I', Λ_M）作为离线输入、并将 I 作为在线输入的随机在线算法，其竞争比渐近趋于 $1/(S+1)$。

扩展与推广。

现在，我们介绍几种在保持其竞争比保证的同时，弱化定理 3.1 中的假设的方法：

- **缺乏对Λ_M的了解**。 如果提供的是参数 Λ_M 的上界，而非其精确值，我们的算法仍能按预期运行，其性能

会随上界线性下降，正如上述定理所示。一种这样的上界就是 $L_{\min}/4$。此外，通过在范围 $[0, L_{\min}/4]$ 内以 2 倍因子猜测 Λ_M，我们可以获得所声称的近似保证，代价是进一步损失 $\mathcal{O}(\log L_{\min})$。因此，即使在没有提供任何预测误差 Λ_M、τ_M、ρ_M 相关信息的情况下，我们的算法仍能实现随位置误差变化的非平凡保证。

- **关于 τ_M 和 Λ_M 的假设。** 显然，若要获得非平凡的竞争比，就必须假设 $\Lambda_M \le L_{\text{最小}}$，因为位置误差大于时间窗大小的预测对在线算法毫无价值。另一方面，不必假设 $\tau_M \le L_{\text{最小}}$。我们可以跟随离线游走的大约 $\tau_M/L_{\text{最小}}$ 种不同时间平移中的一种，从而应对更大的时间窗口错误。这会使我们的近似因子额外增加 $\tau_M/L_{\text{最小}}$ 倍。

- 随机奖励。我们的结果在随机奖励的情形下同样成立。具体而言，考虑这样一种情形：奖励 $\{\pi_\sigma\}_{\sigma\in I}$ 是从定义在 $\mathbb{R}_+^I$ 上的某个联合分布（不一定是乘积分布）D 中抽取的。在这种情况下，我们定义 $\mathrm{Rew}(W, I) := \sum_{\sigma\in \mathrm{Cov}(W,I)} \mathrm{E}[\pi_\sigma]$，并将 $\mathrm{Opt}(G, I)$ 定义为任意游走 $W \in \mathcal{W}(G)$ 所获得的最大奖励。[5] 最后，我们将 $\mathrm{RewErr}(\sigma, \sigma')$ 定义为 π'_σ 与 $\mathrm{E}[\pi_\sigma]$ 之间的不匹配。在此情形下，我们的分析仍能给出与之前相同的近似结果。形式化证明见附录 C。

推论 3.9. 给定一个 (G, I, I', M) 实例，其中带预测的时间窗旅行商问题的请求具有随机抽取的奖励，且预测误差满足 $\tau_M \le L_{\min}/2$ 且 $\Lambda_M \le (L_{\min} - 1)/4$，则存在一种多项式时间在线算法，将元组 (G, I', Λ_M) 作为离线输入，并将 I 作为在线输入，构造一条游走 $W \in \mathcal{W}(G)$，使得

$$\mathbb{E}[\mathrm{Rew}(W, I)] \ge \frac{1}{O(\Lambda_M \cdot \rho_M^2 \cdot \log \min(D, L_{\max}))} \cdot \mathrm{OPT}(G, I)$$

- 有根实例。接下来，我们考虑也指定起始顶点 v_0 的情形，解空间 $\mathcal{W}(G)$ 包含所有从 $t = 0$ 时刻起始于顶点 v_0、在 G 上进行的游走。我们很容易看出，在额外假设每个请求 $\sigma = (v_\sigma, r_\sigma, d_\sigma, \pi_\sigma)$ 满足条件 $\ell(v_0, v_\sigma) \le r_\sigma$ 时，这种设定与其无根对应情形基本等价。这一假设是合理的，因为无论如何，任何算法都无法在时间 $\ell(v_0, v_\sigma)$ 之前访问请求 σ。显然，对于任意有根实例 (G, I, v_0)，无根最优值 $\mathrm{Opt}(G, I)$ 是有根最优值 $\mathrm{Opt}(G, I, v_0)$ 的上界。另一方面，我们的算法计算出的无根路径可以转换为一条从 v_0 出发、奖励相同的路径：直接从 v_0 前往最先服务的预测请求即可，因为这段距离至多等于该请求的释放时间。

- **部分匹配。** 接下来，我们考虑这样一种情况：并非所有真实请求都包含在预测请求中；反过来，有些预测请求也完全不对应任何真实请求。遵循 [APT21], 的框架，我们考虑 I 与 I' 之间的部分匹配，并将 Δ_1^M 定义为所有未匹配真实请求的总奖励，将 Δ_2^M 定义为所有未匹配预测请求的预测总奖励。于是，很容易看出，我们的分析对 I 与 I' 中彼此匹配的子集仍然成立，额外代价不超过 $\Delta_1^M + \Delta_2^M$。形式化证明见附录 C。

[5] 请注意，我们不允许最优游走根据奖励实例化进行调整。即使有预测，在在线场景中也无法与自适应游走竞争。

推论 3.10。 给定一个带预测的时间窗旅行商问题的实例 (G, I, I')，令 M 为 I 与 I' 之间的任意（不完整）匹配，并令误差参数 Λ_M、ρ_M、τ_M、Δ_1^M和Δ_2^M如上所定义。则存在一个在线算法，以 (G, I', Λ_M) 作为离线输入，以 I 作为在线输入，并返回一个游走 $W \in \mathcal{W}(G)$，使得

$$\mathbb{E}\left[\mathrm{Rew}(W, I)\right] \geq \Omega\left(\frac{1}{\Lambda_M \cdot \rho_M^2 \cdot \log \min(D, L_{\max})}\right) \cdot \left(\textsc{Opt}(G, I) - \Delta_1^M\right) - \frac{\Delta_2^M}{\rho_M}.$$

- 多对一匹配。考虑这样一种情形：预测较为粗略，每个预测位置都对应多个潜在真实请求。我们可以在预测框架中对此类情形进行建模，并获得与定理 3.1 几乎相同的保证。具体而言，在此情形下，设 M 是从 I 到 I'的多对一匹配。我们将预测请求 $\sigma' \in I'$ 的位置误差定义为这样一条最短路径的长度：路径从σ'出发，访问 M中所有作为 σ' 原像的真实请求的位置，在每处停留一个时间单位，最后返回 σ'。注意，该位置误差等于以 σ'为起点的定向越野问题的最优解长度。相应地，我们希望与 σ' 相关联的奖励能够反映与 σ'匹配的所有真实请求的总奖励，并据此定义其奖励误差。最后，时间窗误差仍按之前的方式定义，即取彼此匹配的所有请求对 σ 和 σ' 中的最大值。对于定理 3.1 所述情形，我们的算法以在线方式构造 I' 与 I 之间的匹配。在这种一对多情形下，我们会为访问到的每个预测请求求解一个以该请求为起点的定向越野问题。因此，算法的性能会因一个较小的常数因子而有所下降，但与之前一样，我们仍可达到 $O(\Lambda_M \rho_M^2 \log \min(D, L\max))$ 的竞争比。更多细节见第 8 节。

定理 3.11。 给定一个带预测的时间窗旅行商问题实例 (G, I, I', M)，其中 M 是如上所定义、带有误差的多对一匹配，并且满足 $\tau_M \leq L_{\min}/2$ 和 $\Lambda_M \leq L_{\min}/2$，存在一种多项式时间在线算法，以元组 (G, I', Λ_M) 作为离线输入，并以 I 作为在线输入，构造一条游走 $W \in \mathcal{W}(G)$，使得

$$\mathbb{E}[\mathrm{Rew}(W, I)] \geq \frac{1}{O(\Lambda_M \cdot \rho_M^2 \cdot \log \min(D, L_{\max}))} \cdot \textsc{Opt}(G, I)$$

4 关联最优值

本节给出引理 3.2 和3.3 的证明。这些引理通过使用服务时间来刻画预测误差，建立真实请求序列与预测请求序列的最优值之间的关系。我们首先证明，服务时间 S 对最优值造成的影响至多为 $2S - 1$ 倍。

引理 3.2。 对于任意带服务时间的 *TW-TSP* 实例(G, I) 以及任意整数 $S \leq L_{\min}$，我们有

$$\textsc{Opt}(G, I, S) \geq \frac{1}{2S - 1} \cdot \textsc{Opt}(G, I, 1).$$

证明。 令 $W \in \mathcal{W}(G)$ 为实现最优值 $\mathrm{Opt}(G, I, 1)$ 的游走，并将 I 中由 W 覆盖的请求记为 $\sigma_i = (v_i, r_i, d_i, \pi_i)$，并按照它们被 W 覆盖的顺序排列。该引理直接来自一个简单的观察：如果我们不服务某个请求 σ_i 之前 和 之后 的 $(S - 1)$ 个请求，那么我们就能节省足够的时间，在该请求的时间窗内用 S 个时间步服务 σ_i。

形式化地说，若 $t_i \in [r_i, d_i - 1]$ 是 W 开始服务请求 σ_i 的时间步，那么通过跳过前后 $(S-1)$ 个请求的空闲时间，我们可以从时间步 t_i–$(S-1)$ 到时间步 $t_i + S$ 一直让 v_i 保持空闲（因为 W 已经让 v_i 空闲了 1 个时间步），同时仍能跟上游走 W。由于 $S \le L_{min}$，容易验证这些时间步中至少有 S 个落在时间窗 $[r_i, d_i]$ 内。

现在，我们将请求 $\sigma_i = (v_i,\ r_i,\ d_i,\ \pi_i)$ 划分为 $2S-1$ 个子序列，每个子序列都从某个请求 $i \in [S]$ 开始，并覆盖请求 σ_i，$\sigma_{i+(2S-1)}$，$\sigma_{i+2(2S-1)}$，依此类推。每个这样的序列都可以像上文那样通过纳入空闲时间的游走来覆盖，从而满足实例 $(G,\ I,\ S)$ 的要求。显然，其中一条游走获得的奖励至少为 $\mathrm{Opt}(G,\ I,\ 1)/(2S-1)$，证明完毕。 □

我们在附录 A 中证明，上述引理给出了不同服务时间下最优值之间的紧确差距。

引理 4.1。对于任意一对整数 (L, S)，只要满足 $L \ge 2S - 2 \ge 1$，就存在一个以根节点为起点的实例(G, I)，其属于带服务成本的 *TW-TSP*，且满足 $L_{min} = L$ 和

$$\textsc{Opt}(G, I, S) = \frac{1}{2S-1} \cdot \textsc{Opt}(G, I, 1).$$

接下来，我们证明引理 3.3，通过妥善处理三种可能的预测误差，建立预测请求序列与真实请求序列的最优解之间的关系。

引理 3.3。设 (G, I, I', M) 是一个带预测的 *TW-TSP* 实例，其中 Λ_M、ρ_M 和 τ_M 分别表示该实例的位置误差、奖励误差和时间窗误差的最大值。定义$S' := 4\Lambda_M + 1$ 和 $S' := 2\Lambda_M + 1$。那么，如果 $\tau_M \le L_{\min}/2$ 且 $\Lambda_M \le (L_{\min} - 1)/4$，则有

$$\textsc{Opt}(G, I', S') \ge \frac{1}{3\rho_M} \cdot \textsc{Opt}(G, I, S).$$

证明。设 W是实现最优值 $\mathrm{Opt}(G, I, S)$ 的游走，并将 I中被 W覆盖的请求记为 $\sigma_i = (v_i, r_i, d_i, \pi_i)$，并按照 W访问它们的顺序排列。设 $\sigma'_i = (v'_i, r'_i, d'_i, \pi'_i)$表示与 σ_i匹配的预测请求，即 $\sigma'_i = M(\sigma_i)$。注意，对应于 $\sigma_i \in \mathrm{Cov}(W, I, S)$ 的所有请求 $\{\sigma'_i\}$ 的总奖励至少为 $\mathrm{Rew}(W, I, S)/\rho_M$。

我们将考虑如下定义的游走 W'： G。游走 W'沿着 W前进，并按序访问请求σ_i 。一旦 W 开始服务 σ_i，W' 就会绕行去访问 σ'_i；在 σ'_i处等待 S' 个时间步；返回 σ_i；在 σ_i 处等待 $S - 2\ell$ 个 $(v_i,\ v'_i)\ - S' \ge 0$ 时间步；然后继续游走 W。注意，除了绕行访问各个 σ'_i 之外，W'与 W完全相同。

我们的目标是切实获取 σ'_i中包含的全部奖励。由于真实请求与预测请求的时间窗不匹配，游走W' 可能会错过其中一部分奖励。为此，我们将考虑游走 W'的两种变体。令 $K := L_{\min}/2 \ge \tau_M$。游走 W'_1与 W'完全相同，只是它在 W'开始后 K 步才出发，因此访问每个位置的时间都恰好比 W' 访问该位置晚 K 步。游走 W'_2 与 W' 完全相同，只是它在 W' 开始之前 K 步出发，[6] 因此访问每个位置的时间都恰好比 W' 访问该位置早 K 步。

现在考虑实例（$G,\ I,\ S$）中由 W 覆盖的请求 σ_i 所对应的某个 σ'_i。我们声称，游走 W'、W'_1 和 W'_2 中至少有一个会在（$G,\ I',\ S'$）中覆盖 σ'_i。令 t 为 W' 到达 v'_i 的时间；回想一下，W' 会停留在该节点，直到至少 $t + S'$。注意，由于 $\sigma_i \in$ 覆盖（$W,\ I,\ S$），$t \ge r_i$ 且 $t + S' \le d_i$。

[6] 具体而言，这段游走从步骤 K 时 W' 所在的位置开始。

首先，假设 $r'_i \le t$ 和 $d'_i \ge t + S'$，那么在（G， I'， S'）中， σ' 会被 W' 覆盖。接下来，假设 $r'_i > t$。那么， W'_1 在时间 $t + K \ge r_i + K \ge r_i + \tau_M \ge r'_i$ 到达 v'_i。另一方面，它会一直停留在 v'_i，直到时间 $t + K + S' < r'_i + K + S' \le r'_i + L_{\min} \le d'_i$。因此， σ'_i 会被 W'_1 覆盖。最后，假设 $d'_i < t + S'$。那么， W'_2 在时间 $t - K > d'_i - S' - K \ge d'_i - L_{\min} \ge r'_i$ 到达 v'_i。另一方面，它会一直停留在 v'_i，直到时间 $t - K + S' \le d_i - K \le d_i - \tau_M \le d'_i$。因此， σ'_i 会被 W'_2 覆盖。

我们得到， W'、 W'_1或 W'_2中至少有一个能取得 Opt(G， I， S) 的至少 $1/3\rho_M$ 份额；由于预测奖励不匹配，损失了 ρ_M 这一因子。引理由此直接得证。

□

5 离线近似

在本节中，我们设计一种 $O(\log \min(D, L_{\max}))$ 确定性多项式时间近似算法，用于求解具有服务时间的 TW-TSP，并证明引理 3.4。我们的证明依赖于不同离线问题之间的一系列归约、对现有算法的应用，以及新算法组件的设计。我们将论证拆分为一系列引理，这些引理的证明见附录 B。

首先，我们设计一种 $O(\log \min(D, L_{\max}))$-近似算法，用于求解 TW-TSP（不含服务时间）。由于 [CKP08] 的工作已经为这一设定提供了一种 $O(\log L_{max})$ 近似算法，因此只需证明以下结论：

引理 5.1。 给定一个具有 $L_{min} \ge 4D$ 的 *TW-TSP* 实例（不含服务时间），存在一种多项式时间算法，可实现 $O(1)$ 近似。

我们的证明基于这样一个观察：当时间窗相较于图的直径足够大时，该问题本质上可归约为一个广为研究的定向越野问题实例，而该问题已有已知的常数近似算法。我们指出，类似的思路已用于 [AV15]。然后，将该算法与 [CKP08] 的算法相结合，即可得到 TW-TSP 的 $O(\log \min(L_{\max}, D))$近似。

引理 5.2。 存在一种用于 *TW-TSP* 的 $O(\log \min(D, L_{\max}))$ 近似算法。

2. 接下来，我们设计一种简单的保持近似比的归约，将含服务时间的 TW-TSP 归约为不含服务时间的 TW-TSP。该归约的核心思路是：将服务时间视为一个增广图中的边长，而该图的直径约为 $D + S$。对于满足 $S \le D$ 的实例，这一增幅可以忽略不计，因此将我们的归约与引理 5.2相结合，即可立即得到以下结论：

引理 5.3. 给定一个具有服务时间的 *TW-TSP* 实例(G, I, S)，满足 $S \le D$，则存在一种多项式时间算法，可实现 $O(\log \min(D, L_{\max}))$ 近似。

3. 最后，我们处理服务时间较大的情形，具体而言，即 $S \ge D$。在这种情况下，我们可以将该实例归约为一个均匀完全图上的实例。此时，TW-TSP-S 本质上等价于研究充分的作业调度 问题，而该问题已有常数近似算法。

引理 5.4。 给定一个满足 $S \le D$ 的、服务成本已知的 *TW-TSP* 实例(G, I, S)，存在一种多项式时间算法，可实现 $O(1)$ 近似。

引理 3.4 的证明可由引理 5.3 和引理 5.4 立即推出。我们指出，TW-TSP 已知最佳近似算法的任何改进，都将立即带来本文所有结果的改进。引理 5.4 基本上使我们可以假设，在所有关注的实例中，$S \le D$。在这一假设下，我们在引理 5.3 的证明中使用的归约基本上表明，TW-TSP-S 等价于直径为 $\mathcal{O}(D)$、最大窗口大小为 $\mathcal{O}(L_{\max})$ 的图上的 TW-TSP。作为直接推论，给定一个竞争比为 $\alpha(D, L_{\max})$ 的离线 TW-TSP 算法，我们可立即得到一个针对 TW-TSP-S 的离线$O(\alpha(D, L_{\max}))$ 近似算法，可在我们的分析中用来替代引理 3.4，并改进定理 3.1 的竞争比。

6 在线算法

本节介绍一种在线算法：它以预测请求序列上的预计算游走为输入，并求解适当的在线匹配问题，以构造能够捕获真实请求的绕行，同时考虑预测中可能存在的误差。该算法的形式化保证见引理 3.5，现为方便读者重述如下：

引理 3.5。给定一个 G，I，I'，M 实例，其属于带预测的时间窗旅行商问题，且满足 $\tau_M \le L_{\min}/2$和 $\Lambda_M \le (L_{\min}-1)/4$；一条游走 $W' \in \mathcal{W}(G)$；以及任意整数 $S' \ge 2\Lambda_M + 1$，存在一种在线算法（算法 *1*），可返回一条游走 $W \in \mathcal{W}(G)$，其期望奖励为

$$\mathbb{E}\left[\mathrm{Rew}(W, I, 1)\right] \ge \frac{1}{6\rho_M} \cdot \mathrm{Rew}(W', I', S').$$

我们先确立一些记号。设 $W' \in \mathcal{W}(G)$ 是任意一条游走，它在 I' 中以 S' 的服务时间服务若干预测请求。我们用 $\sigma'_i = (v'_i, r'_i, d'_i, \pi'_i)$ 表示 I' 中被 W' 覆盖的预测请求，并按照它们沿 W' 被访问的顺序排列。同样，我们用 $\sigma_i = (v_i, r_i, d_i, \pi_i) \in I$ 表示与预测请求 σ'_i 匹配的真实请求，即 $\sigma'_i = M(\sigma_i)$。

总体而言，我们的算法沿着游走 W' 前进，但到达预测请求 σ'_i 时，会考虑绕行，服务此时可用的真实请求。为此，我们将“可达”真实请求集合定义如下。

定义 6.1. 给定一个在时间 t 位于请求 $\sigma'_i \in I'$ 处的部分游走 W，我们将可达请求集合 $R_i(W, t)$ 定义为所有满足以下条件的 $\sigma \in I$ 的集合：

1. $r_\sigma \le t \le d_\sigma - \ell(v'_i, v_\sigma) - 1$.
2. $2\ell(v'_i, v_\sigma) + 1 \le S'$。

我们的算法会考虑所有尚未被该游走覆盖的可达请求，选取奖励最高的请求，绕行前往并覆盖该请求，然后返回到 σ'_i 并继续游走。为处理时间窗口错误，我们的算法会稍早、准时或稍晚开始游走，如引理 3.3 的证明所示。下面正式描述该算法。

我们首先指出，W 算法构造的游走始终能够在所需时间 v'_i 可行地访问与请求 $\sigma'_i \in \mathrm{Cov}(W', I', S')$ 对应的顶点 t_i。这是因为，根据构造，游走 W 在步骤12 中采取的绕行长度始终至多为 S'——即空闲时间的长度 W' 在 v'_i 所花费的时间——这是因为$\hat{}\sigma \in R_i(W, t_i)$ 以及

算法 1 带预测的 TSP-TW 在线算法 离线输入：图 G，预测请求 I'，游走 $W' \in \mathcal{W}(G)$，服务时间 S'。在线输入：真实请求 I。输出：游走 $W \in \mathcal{W}(G)$。1：令 $K = L_{\min}/2$。从 $\{-1, 0, 1\}$ 中均匀随机选择 ϵ。2：定义已覆盖请求集合 $C = \emptyset$。3：对于 $i \leftarrow 1$ 到 $|\mathsf{Cov}(W', I', S')|$，执行以下操作：4：令 t'_i 表示 W' 访问 σ'_i 的时间。5：设置 $t_i \leftarrow t'_i + \epsilon K$。6：在时间 t_i 访问 v'_i。7：构造集合 $R_i(W, t_i)$，其中包含在时间 t_i 可从 I 到达的请求。8：如果 $R_i(W, t_i) \setminus C = \emptyset$，则：9：不执行任何操作。10：否则：11：令 $\sigma\hat{}$ 为 $R_i(W, t_i) \setminus C$ 中奖励最高的请求。12：访问 $v_{\hat{\sigma}}$；在 $v_{\hat{\sigma}}$ 处花费一个单位的空闲时间；返回 v'_i。13：设置 $C \leftarrow C \cup \{\hat{\sigma}\}$。

因此，$2\ell(v'_i, v_{\hat{\sigma}}) + 1 \leq S'$。因此，所有请求$\hat{\sigma}$在步骤12中访问的请求，确实都由游走 W访问。

现在，我们将 W 覆盖的总奖励与对应于 $\sigma'_i \in \mathsf{Cov}(W', I', S')$ 的真实请求 σ_i 中包含的奖励联系起来。为此，我们首先注意到，每个这样的请求以恒定概率可由 W 到达。

声明1。对于每个 i，$\sigma_i \in R_i(W, t_i)$ 以至少 $1/3$ 的概率成立。

证明。回顾定义可知，我们有 $\sigma'_i = M(\sigma_i)$，因此 $2\ell(v'_i, v_i) + 1 \leq 2\Lambda_M + 1 \leq S'$。所以，请求 σ 始终满足可达集合 $R_i(W, t_i)$ 定义中的第二个要求。现在让我们考虑第一个要求，并回顾 $t_i = t'_i + \epsilon K$，其中 $\epsilon \in \{-1, 0, 1\}$。接下来我们将论证，对于 ϵ 的三种选择中至少一种，$t_i \in [r_i, d_i - 1 - \ell(v'_i, v_i)]$。根据 ϵ 的均匀随机选择，即可得出该结论。

1. 若 $t'_i \in [r_i, d_i - 1 - \ell(v'_i, v_i)],$，则该结论对 $\epsilon = 0$ 和 $t_i = t'_i$ 成立。

2. 假设 $t'_i < r_i$。那么，对于 $\epsilon = 1$，有 $t_i = t'_i + K \geq r'_i + \tau_M \geq r_i$，并且还有 $t_i = t'_i + K < r_i + K < d_i - L_{min} + L_{min}/2$，因此 $t_i = t'_i + K \leq d_i - 1 - \ell(v'_i, v_i)$，因为$\ell(v'_i, v_i) \leq \Lambda_M \leq L_{min}/2$。因此，在这种情况下，取 $\epsilon = 1$，我们有 $t_i = t'_i + K \in [r_i, d_i - 1 - \ell(v'_i, v_i)]$。

3. 最后，假设 $t'_i > d_i - 1 - \ell(v'_i, v_i)$。那么，对于 $\epsilon = -1$，我们有 $t_i = t'_i - K \leq d'_i - S' - \tau_M \leq d_i - S'$，因此 $t'_i - K \leq d_i - 1 - \ell(v'_i, v_i)$，因为 $S' \geq 2\Lambda_M + 1 \geq \ell(v'_i, v_i) + 1$。此外，$t_i = t'_i - K > d_i - 1 - \ell(v_i, v'_i) - L_{min}/2 > r_i + L_{min}/2 - \ell(v_i, v'_i) - 1$，因此 $t'_i - K \geq r_i$，因为 $\ell(v_i, v'_i) \leq \Lambda_M \leq L_{min}/2$。因此，在这种情况下，我们有 $t_i = t'_i - K \in [r_i, d_i - 1 - \ell(v'_i, v_i)]$，并选择 $\epsilon = -1$。

□

现在，我们准备通过匹配型论证来证明引理 3.5 。为了计算算法构造的游走 W 所覆盖的奖励，我们将采用标准的收费方案。

每当算法在步骤 12中，从预测请求 σ'_i绕行以覆盖某个真实请求 $\hat{\sigma}$ 时，我们会将获得的奖励 $\pi_{\hat{\sigma}}$ 的一半计入 $\hat{\sigma}$ 本身，另一半计入请求 σ_i。形式化地，令 $\mathrm{Cr}(\sigma)$ 表示 $\sigma \in I$收到的总计入额。那么在步骤 12 中，我们将把 $\mathrm{Cr}(\hat{\sigma})$ 和 $\mathrm{Cr}(\sigma_i)$ 都增加 $\pi_{\hat{\sigma}}/2$。

现在考虑一个 $\sigma_i \in I$，对应于 $\sigma'_i \in$ 覆盖(W', I', S')。根据声明1，该请求以至少 $1/3$ 的概率位于$R_i(W, t_i)$中。如果在时间 t_i，该请求已被 W 覆盖，那么我们得到 $\mathbf{Cr}(\sigma_i) \geq \pi_i/2$。否则，我们选取一个 $\hat{\sigma} \in R_i(W, t_i)$，使得 $\pi_{\hat{\sigma}} \geq \pi_i$，因此，我们再次得到 $\mathbf{Cr}(\sigma_i) \geq \pi_i/2$。

综合以上各部分，我们得到

$$\begin{aligned}
\mathrm{E}[\mathrm{Rew}(W, I, S)] = \mathrm{E}\left[\sum_{\sigma \in I} \mathrm{Cr}(\sigma)\right] &\geq \sum_{i:\sigma'_i \in \mathrm{Cov}(W', I', S')} \mathrm{E}\left[\frac{\pi_i}{2} \mathbb{1}[\sigma_i \in R_i(W, t_i)]\right] \\
&\geq \frac{1}{3} \cdot \sum_{i:\sigma'_i \in \mathrm{Cov}(W', I', S')} \frac{\pi_i}{2} \\
&\geq \frac{1}{6} \cdot \frac{1}{\rho_M} \cdot \sum_{i:\sigma'_i \in \mathrm{Cov}(W', I', S')} \pi'_i \\
&= \frac{1}{6\rho_M} \cdot \mathrm{Rew}(W', I', S')
\end{aligned}$$

引理的证明至此完成。

7 下界

在本节中，我们给出补充我们结果的下界。首先，我们将在第 7.1 节说明预测的必要性。然后，在第 7.2 节中，我们将证明，带预测的时间窗旅行商问题的竞争比必须随位置误差线性增长。最后，在第 7.3 节中，我们将论证在带预测的时间窗旅行商问题的定义中采用非零服务时间的必要性。

7.1 无预测的在线时间窗旅行商问题下界

我们认为，在没有预测的情况下，在线时间窗旅行商问题不存在任何合理的竞争比。对于行为完全可预测的确定性算法，只需构造仅含 2 个顶点且时间窗足够小的简单实例，即可说明其近似比无法获得任何有界保证。

引理 7.1. 对于包含 $L_{\min} \leq D$ 的实例，任何用于在线时间窗旅行商问题的确定性在线算法，其竞争比均无界。

证明. 设 Det 是任意确定性算法，G 是仅含两个顶点v_1, v_2 的路径图，两顶点之间由一条长度为 D 的边连接。由于 Det 是确定性的，只要指定所有释放时间为 $\leq t$ 的请求，我们就可以假定知道它在任意步骤 t 的位置。现在，我们将构造一个利用这一信息的请求序列 I。

在最初的 D 个时间步中，我们不发布任何请求。然后，在 $t = D$，令 $v_D \in \{v_1, v_2\}$ 为 Det 当前游走所在的位置，并相应地令 v'_D 为 G 的另一个顶点。我们将第一个请求设为任意 $L \leq D$ 的 $\sigma =$(${v'_D}$, D, $D + L$, 1)。显然，Det 无法处理此请求，因为即使对于 $L = D$，它也会在截止时间到达 v'_D，因此无法在一个时间步内处理该请求。接下来，我们在之后的 $2D$ 个时间步中不发布任何新请求，并在 $t = 3D$ 重复相同的过程，

请求 Det 当前没有占据的顶点。同样，我们在 $t = 5D$、 $t = 7D$ 等时刻重复相同的过程。无论请求序列的长度如何，Det 获得的总奖励均为 0。

另一方面，不难看出，如果我们的请求序列 I 总共包含 N 个请求，那么 Opt(G, I, 1) $= N$。这是因为请求之间相隔 $2D$步，因此，若最优离线算法提前知道整个序列，就总能及时到达每个请求，并在其对应的时间窗内完成服务。

□

对于随机化算法，可以略微改善这一结果。具体而言，随机化算法先均匀随机地选取一个顶点，然后在整个序列期间停留在该顶点上，其竞争比为 $1/n$。事实证明，这实际上是随机化算法在在线时间窗旅行商问题中能够达到的最佳竞争比：

定理 3.6。 对于实例中含有 $L_{\min} \leq D$ 的在线时间窗旅行商问题，任意随机在线算法的竞争比至多为 $1/n$。

证明。 设 $G(V, E, \ell)$ 是一个含有 $n = |V|$ 个顶点的均匀完全图，其中所有边的长度均为 D。固定任意整数 N，令顶点 v_1, v_2, ..., $v_N \in V$独立且均匀随机地选取。接下来，固定任意时间窗长度 $L \leq D$，并考虑这些顶点上的（随机）请求序列 $I = \{\sigma_i\}_{i=1}^{N}$，其形式为 $\sigma_i =$(v_i, ($2i-1$)D, ($2i-1$)$D+L$, 1)。根据姚氏极小极大原理，确定性算法在该随机实例上的（期望）竞争比下界，同样也是随机化算法的竞争比下界。

由于时间窗之间相隔 $2D$，不难看出，对于 I 的任意一种实现，最优值 Opt(G, I, 1) $= N$。另一方面，由于 $D \geq L$，服务请求的唯一方式是在释放时间位于其对应的顶点上。由于顶点是随机的，对于任何确定性算法，这种情况发生的概率恰好为 $1/n$，因此任何确定性算法在此实例上的期望奖励为 N/n，从而证明了该结论。

□

7.2 对位置误差的紧线性依赖

本节将证明，即使假设计算能力呈指数级，对于带预测的 TW-TSP，任何随机在线算法都无法避免对位置误差的线性依赖。换言之，我们将正式证明定理 3.8，并为方便读者在此重述：

定理 3.8。 对于任意 $S > 0$，都存在一个带预测的 *TW-TSP* 实例 (G, I, I', M)，满足 $\tau_M = 0$、$\rho_M = 1$ 和 $\Lambda_M = S$，使得任何将元组 (G, I', Λ_M) 作为离线输入并将 I 作为在线输入的随机在线算法，其竞争比渐近地趋近于$1/(S+1)$。

证明。 任取 $S > 0$ 并令 K, C 和 N 为稍后指定的整数参数。我们构造一个图 $G = \cup_{i=0}^{N-1} G_i$，它由 N个 $G_0, \ldots, G_{N-1}$ 个完全图的副本组成，每个完全图有 C个顶点，且所有边长均为 S，排列方式使得 G_i 中的每个顶点都与 G_{i+1} 中的每个顶点通过一条边长为 KS的边相连。图 C 和 N 取较小值时的示意图见图 1。

接下来，我们从每个子图 G_i中独立且均匀地随机选择一个顶点 v_i，并构造（随机化的）请求序列 $I = \{\sigma_i\}_{i=0}^{N-1}$，其中 $\sigma_i =$ (v_i, r_i, $r_i + KS$, 1)，对于$r_i = i \cdot (KS+1)$。至于预测，我们只需以相同方式构造第二个实例 I'，并将其作为 I 的离线预测提供。请注意，对于 I 和I'的所有可能构造，都存在一个匹配 M，使得 $\tau_M = 0$、 $\rho_M = 1$ 和 $\Lambda_M = S$ 成立；具体来说，就是将来自同一子图的顶点配对的匹配。该预测没有提供

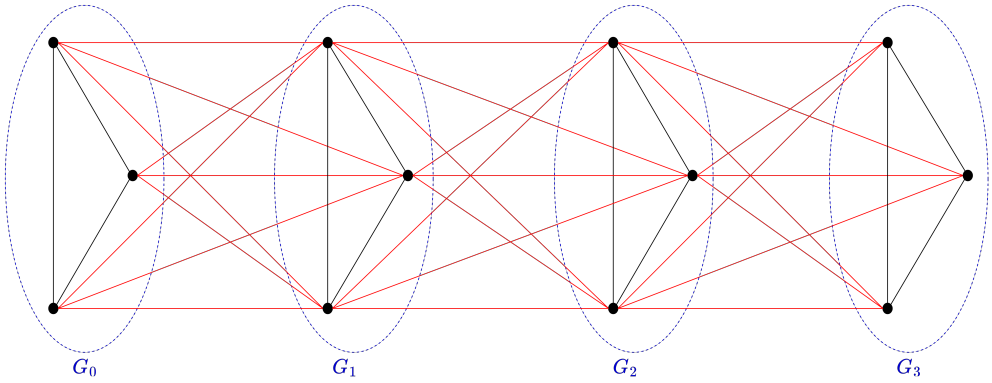

图 1： G 关于 $C = 3$ 和 $N = 4$的一个示例。黑色边的长度为 S，红色边的长度为 KS。

除真实实例 I 是通过上述随机化方法构造这一事实之外的任何信息给算法。此外，请注意，通过适当增大 K，位置误差 Λ_M 相对于时间窗长度 $L = KS$ 可以任意小。

容易看出，对于 I的所有可能实现，都有 Opt(G， I，1) $= N$。事实上，考虑这样一条游走：从（随机）顶点 v_0出发，原地等待 1 步，然后访问顶点 v_1，原地等待 1 步，再访问 v_2，依此类推。这样的游走会在第$t = i \cdot KS + i = r_i$ 步访问每个顶点 v_i，因此能够处理 I中的所有请求，从而获得 N的总奖励。接下来，我们将证明，任何确定性算法在随机序列 I上的期望奖励都趋近于 N/S。利用姚氏极小极大原理，这将立即推出随机化算法的下界趋近于 $1/S$，从而完成定理的证明。

考虑任意一个针对实例 (G， I， I'， M) 的带预测时间窗旅行商问题确定性算法。注意，由于预测 I' 不提供任何信息，只需将该算法作为实例 I上的在线 TW-TSP 确定性在线算法进行分析。关键观察是，由于请求的时间窗以及连接不同子图的边的长度均为 KS，算法在访问同一子图中某个可能不同的顶点之前，不可能知道请求将到达子图 G_i的哪个顶点 v_i。具体而言，如果算法在时刻 r_i位于子图 G_j 中以服务 $j < i$，那么它无法在请求 i 的时间窗结束之前到达顶点 v_i。因此，任何确定性算法若要处理子图 G_i上的请求，就必须先访问 G_i中的某个顶点 v'_i。如果恰好 $v'_i = v_i$，那么它可以立即处理该请求；否则，它必须行进 S 的距离才能到达 v_i。

我们根据确定性算法是否恰好到达子图中的正确顶点，将请求集划分为两组 N_+ 和 N_-。N_+中的所有请求都可以像最优游走那样，在没有任何额外延迟的情况下完成服务。另一方面，处理 N_- 中的一个请求需要算法花费额外的 S 时间，才能转移到正确的顶点。由于时间窗的长度为 KS，算法在松弛量耗尽之前最多可以这样做 K 次。发生这种情况时，算法必须跳过接下来的 S 个请求，以恢复足够的松弛量来弥补下一次失误。

形式化地，考虑算法在 N_- 中能够可行地服务的最后一个请求，将其记为 l。令 A 表示算法在 N_- 中通过额外延迟服务的请求数量（截至 l之前），令 B 表示算法在 N_+ 或 N_- 中跳过的请求数量（截至 l之前）。那么，为了使算法能在 l 的时间窗结束之前到达该处，算法产生的额外延迟总量必须

不超过 KS；该总量即为 $AS-B$。整理后可得 :

$$A(S+1)-(A+B)\le KS,\quad \text{or},\quad A\le \frac{A+B}{S+1}+\frac{KS}{S+1}<\frac{N}{S+1}+K$$

因此，算法获得的总期望奖励至多为

$$\mathbb{E}[|N_+|]+A+1\le \mathbb{E}[|N_+|]+\frac{N}{S+1}+K+1$$

最后，利用 $\mathbb{E}[|N_+|]=N/C$ 这一事实，我们得到，任何确定性算法在随机实例 I 上的竞争比至多为

$$\frac{1}{C}+\frac{K+1}{N}+\frac{1}{S+1}$$

将 $N>>K$ 和 C 取足够大，我们就可以使该竞争比如期渐近趋近于 $1/(S+1)$。

□

7.3 比较有无服务时间时的最优值

正如引理 3.2 所示，$\textsc{Opt}(G,I,1)$ 与 $\textsc{Opt}(G,I,S)$ 之间的差距随服务时间 S线性变化。本节研究 $\textsc{Opt}(G,I,1)$ 与 $\textsc{Opt}(G,I,0)$之间的差距，并表明二者之间存在大得多的差异。

定理7.2. 对于任意整数 L 和 D，$L\le D$，存在一个 *TW-TSP* 的离线实例 (G,I)，其中 D 是网络的直径，且每个请求的时间窗长度均为 L，使得 $\textsc{Opt}(G,I,1)\le \frac{L}{D+1}\cdot\textsc{Opt}(G,I,0)$。

证明。 考虑路径图 G，其顶点 v_0 到 v_D 依次通过长度为 1 的边相连。显然， G 的直径为 D。接下来，考虑由 $(D+1)$ 个请求$I=\{\sigma_i\}_{i=0}^{D}$组成的序列，其中 $\sigma_i=(v_i,\ i,\ L+i,\ 1)$。注意到，当服务成本为 0 时，只需沿着从 v_0 到 v_D 的游走即可覆盖所有请求，因此 $\textsc{Opt}(G,\ I,\ 0)=D+1$。

另一方面，不难看出 $\textsc{Opt}(G,\ I,\ 1)=L$。首先注意，不失一般性，可以假设最优游走从 v_0 开始。否则，设 σ_i 为最优游走服务的第一个 请求。由于 σ_i 无法在第 $r_i=i$ 步之前得到服务，我们可以改为从 v_0 出发，走 i 步到达 v_i，然后沿着原游走继续前进，从而获得完全相同的奖励。

我们的论证只需注意到：任何从 v_0 出发并已服务 x 个请求、服务成本为 1 的游走，在第 $j+x$步之前都无法到达顶点v_j。因此，一旦 x变为 L，此后所有时间窗都将错过。

□

定理7.2 指出，最优值$(G,\ I,\ 0)$ 是一个比最优值$(G,\ I,\ I1)$强得多的基准。然而，这并不排除设计一种算法，使其能够针对这一更强的基准最优值$(G,\ I,\ 0)$具有竞争力的可能性。接下来我们将证明，针对这一基准，任何带预测的随机在线算法都无法达到优于线性的竞争比。直观而言，要求算法在每个请求处花费非零时间，也使算法能够通过跳过最优解延迟1才服务的请求，从预测误差可能造成的失误中恢复。对于0 服务时间的情形，则无法采用类似的方法。

定理 3.7。 对于任意 $S>0$，都存在一个实例 (G,I,I',M)，它属于带预测的时间窗旅行商问题，且服务时间为 0，满足 $\tau_M=0$、$\rho_M=1$和 $\Lambda_M=S$；对于任何将元组 (G,I',Λ_M) 作为离线输入、将 I 作为在线输入的随机在线算法，其获得的奖励至多为 $O(1/n)\cdot$最优值$(G,I,0)$。这里，n 是 G 中顶点的数量。

证明。 我们构造与定理 $G = \cup_{i=0}^{N-1} G_i$ 中相同的图 3.8，该图由 N 个边长为 S 的完全图副本组成，这些副本通过边长为 KS 的边依次连接（见图 1）。对于请求序列，我们再次从每个子图中独立且均匀随机地选取一个顶点v_i ，并构造（随机化的）请求序列 $I = \{\sigma_i\}_{i=0}^{N-1}$，其中$\sigma_i = (v_i, iKS, (i+1)KS - 1, 1)$；请注意，释放时间与定理 3.8 略有不同，以适应不存在服务成本的情况。

对于预测，我们以相同方式构造第二个实例 I' ，并将其作为针对 I 的离线预测。通过将同一子图上的请求进行匹配，我们得到$\tau_M = 0$、 $\rho_M = 1$ 和 $\Lambda_M = S$。显然，对于 I 的任意一种实现，都有 $\mathrm{Opt}(G, I, 0) = N$。因此，要证明我们的定理，只需论证：对于实例 (G, I, I', M)，任何针对带预测的 TW-TSP 的确定性算法都能获得 $\mathcal{O}(N/n)$ 的期望奖励，然后应用姚氏极小极大原理。此外，由于预测 I' 不提供关于 I 的任何信息，只需界定针对（在线）实例 I 的在线 TW-TSP 确定性算法所获的奖励。

固定任意一个针对实例 I 的在线 TW-TSP 确定性算法。注意，由于不同子图之间的边长为 KS ，且时间窗长度为 $KS - 1$，除非算法在 $t = r_i$ 时已经位于子图 G_i中的某个顶点，否则它不可能服务请求 σ_i 。同理，在服务请求 σ_i 后，算法不可能再服务任何满足 $i > j$ 的请求 σ_j 。最后，由于算法若从 G_0 中的某个顶点出发，就能在请求的释放时间之前到达所有请求，因此不失一般性，我们可以假设确实如此。

我们根据确定性算法是否 在请求的释放时间之前到达子图中的正确顶点，将 N 请求划分为两组： N_+ 和 N_- 。根据实例的随机构造，我们有 $\mathbb{E}[|N_+|] \le N/C$。对于 N_-中的请求，如果算法希望为其提供服务，就必须绕行长度为 S 的路程才能到达正确的顶点。我们的证明依赖于这样一个事实：为 N_-中的 K 个请求提供服务后，算法便无法再为任何其他请求提供服务。为说明这一点，令v_j 为确定性算法所服务的 N_- 中的第 K个请求。正如我们已经证明的，任何满足 $i < j$ 的请求 v_i 都无法再得到服务。另一方面，不失一般性地假设算法从G_0中的一个顶点出发；要在进行长度为 S 的 K 次绕行后到达 G_i 中的一个顶点，至少需要 $iKS + K > d_i$个时间步。综上，通过令 $K = \mathcal{O}(1)$ 且 $N = 2K$，我们得到算法的期望奖励至多为$N/C + K = \mathcal{O}(N/n)$，因为 $n = NC$。 □

8 多对一匹配

本节将考虑第3节中引入的针对预测误差的多对一匹配模型。我们将证明定理 3.11：

定理 3.11。给定一个(G, I, I', M) 的带预测的时间窗旅行商问题实例，其中 M是如上所定义的带误差的多对一匹配，并满足 $\tau_M \le L_{\min}/2$ 和 $\Lambda_M \le L_{\min}/2$，则存在一个多项式时间在线算法，以元组 (G, I', Λ_M) 作为离线输入，以 I作为在线输入，并构造一条游走 $W \in \mathcal{W}(G)$ ，使得

$$\mathbb{E}[\mathrm{Rew}(W, I)] \ge \frac{1}{O(\Lambda_M \cdot \rho_M^2 \cdot \log \min(D, L_{\max}))} \cdot \mathrm{OPT}(G, I)$$

定理的证明 3.11。 显然，引理 3.2 和 3.4 在多对一匹配设置下仍然成立，因为它们与匹配定义无关，只是陈述了带服务时间的（离线）TW-TSP 问题的性质。此外，引理 3.3 仅考虑从真实请求到预测的匹配，即从真实请求 到 预测。由于每个请求仍然只与一个预测匹配，我们的

分析过程无需任何改动即可继续。事实上，请注意，在这种情况下，我们对位置误差 Λ_M 的定义略强一些，因为它还包括服务时间和返回时间。因此，我们可以放宽假设为 $\Lambda_M \le L_{\min}/2$ ，并采用$S' = \Lambda_M$ 的服务时间，而非 $2\Lambda_M + 1$。

我们结合所有这些组件，预先计算一条游走 $W' \in \mathcal{W}(G)$，使得

$$\mathrm{Rew}(W', I', S') \ge \frac{1}{O(\Lambda_M \cdot \rho_M \cdot \log \min(D, L_{\max}))} \cdot \mathrm{OPT}(G, I)$$

对于 $S' = \Lambda_M$，假设 $\Lambda_M \le L_{\min}/2$ 和 $\tau_M \le L_{\min}/2$。接下来需要解决分析中的在线组件，即论证如何构造一条在线游走 $W \in \mathcal{W}(G)$，使得

$$\mathbb{E}\left[\mathrm{Rew}(W, I, 1)\right] \ge \frac{1}{\mathcal{O}(\rho_M)} \cdot \mathrm{Rew}(W', I', S').$$

回想一下，在（标准的）一对一匹配设置中，我们通过算法 1来实现这一点，其形式化保证见引理 3.5 。算法1 以在线方式构造 I' 与 I 之间的匹配。对于这种多对一设置，我们需要为每个预测覆盖多个真实请求。为此，我们针对访问的每个预测请求，求解以该请求为根的定向越野问题实例。

总体而言，我们的算法沿着游走 W' 前进，但到达预测请求 σ'_i 时，会考虑绕行长度为 S' 的路线，以尽可能多地获取当时可服务的真实请求所带来的奖励。为此，我们更新“可达”真实请求的定义，以仅处理预测时间窗之间的差异。

定义 8.1. 给定一段在时间 t位于请求 $\sigma'_i \in I'$ 处的部分游走 W ，我们将可达请求集合 $R_i(W, t)$ 定义为所有满足 $r_\sigma \le t \le d_\sigma - \ell(v'_i, v_\sigma) - 1$ 的 $\sigma \in I$。

请注意，此定义弱于一对一设置中的定义，因为在近似求解相应的定向越野问题时，将通过限制绕行长度来施加对可达请求的接近性约束。我们的算法会考虑所有尚未被游走覆盖的可达请求，并近似求解长度为 S' 的最大奖励环：该环从预测请求出发，覆盖一部分尚未被覆盖的可达真实请求，然后返回预测顶点。同样，为了应对预测时间窗中可能存在的误差，我们的算法会稍早开始、准时开始或稍晚开始游走。下面将正式描述该算法。

我们的算法使用$(2+\epsilon)$-近似算法来求解有根定向越野问题，该算法见 [CKP08] ，并将其作为子程序，以决定在服务时间内服务哪些真实请求。我们将此算法称为ORIEN。需要说明的是，定向越野问题的实例通常不指定终点顶点，但我们仍可尝试所有可能的终点v_j ，并将游走长度从 S' 缩短为 $S' - \ell(v'_i, v_j)$，整个过程均可在多项式时间内完成。此外，定向越野问题通常将服务时间定义为0，但我们仍可通过在图中添加额外的边，将 1/2 长度的边连接到新的请求顶点，并将请求移至这些顶点，使其服务时间为0。这样，服务时间便可表示为边长。

由于 τ_M 定义为所有匹配对中的最大时间窗口误差，并且$\tau_M \le K = L_{\min}/2$ 和 $\Lambda_M \ge 1 + 2\ell(v_{\sigma'}, v_\sigma)$ 对所有满足 σ 的 $M(\sigma) = \sigma'$，我们可以直接应用命题1 的证明，得到以下结论：

断言 2. 对于每个 i，只要 $\sigma'_i \in$ 覆盖(W', I', S')，以及每个 j，只要 $\sigma'_i = M(\sigma_j)$，则以至少 1/3 的概率有$\sigma_j \in R_i(W, t_i)$ 。

算法 2 一对多匹配设置下带预测的带时间窗旅行商问题在线算法。离线输入：图 G、预测请求 I'、游走 $W' \in \mathcal{W}(G)$、服务时间 S'。在线输入：真实请求 I。输出：游走 $W \in \mathcal{W}(G)$。1：令 $K = L_{\min}/2$。从 $\{-1, 0, 1\}$ 中均匀随机选择 ϵ。2：定义已覆盖请求集合 $C = \emptyset$。3：对于 $i \leftarrow 1$ 至 $|Cov(W', I', S')|$，执行。4：令 t'_i 表示 W' 访问 σ'_i 的时刻。5：设定 $t_i \leftarrow t'_i + \epsilon K$。6：在时刻 t_i 访问 v'_i。7：构造集合 $R_i(W, t_i)$，其中包含在时刻 t_i 于 I 可达的请求。8：使用 ORIEN 计算长度为 $|W_i| \le S'$ 的环 W_i，该环以顶点 v'_i 为根，并从集合 $C_i \subseteq R_i(W, t_i)$ 中尽可能多地收集奖励，服务时间为 1。9：令 $C_i \subseteq R_i(W, t_i) \setminus C$ 表示 ORIEN 覆盖的请求集合。10：沿着 W_i 绕行并服务 C_i 中的请求；返回后，空闲 $S' - |W_i| \ge 0$ 步。11：设定 $C \leftarrow C \cup C_i$。

接下来，令 $N_i \subseteq I$ 为与预测 $\sigma'_i \in \mathrm{Cov}(W', I', S')$ 匹配的请求集合。令 $U_i \subseteq N_i$ 为 I 中映射到预测 σ'_i 且在算法针对预测 σ'_1 至 σ'_{i-1} 进行绕行期间未被算法覆盖的请求集合；即 $U_i = \{\sigma_j : \sigma'_i = M(\sigma_j)\} \setminus \cup_{j<i} C_j$。根据奖励误差的定义，我们有 $\pi'_i \le \rho_M \cdot \sum_{j:\ \sigma_j \in N_i} \pi_j$，因此得到

$$\mathrm{Rew}(W', I', S') \le \rho_M \cdot \sum_{i:\sigma'_i \in \mathrm{Cov}(W', I', S')} \left(\sum_{j:\sigma_j \in N_i} \pi_j \right)$$

为了计算算法构造的游走 W 所覆盖的奖励，我们采用以下收费方案。每当算法在某个预测 σ'_i 的服务时间内进行绕行，并服务某个与预测 $\hat{}\sigma'$ 匹配的请求 $\sigma \in C_i$（即 $\hat{}\sigma' = M(\sigma)$）时，我们将一半奖励记入预测 σ'_i，另一半记入预测 $\hat{}\sigma'$。我们用 $\mathrm{Cr}(\sigma')$ 表示预测 $\sigma' \in I'$ 获得的总奖励记账。

断言 3. 对于每个 $\sigma'_i \in$ 覆盖(W', I', S')，都有 $\mathbb{E}\left[\mathrm{Cr}(\sigma'_i)\right] \ge \Omega(1) \cdot \sum_{j:\ \sigma_j \in N_i} \pi_j$.

证明。任取 $\sigma'_i \in \mathrm{Cov}(W', I', S')$。若 $\sigma_j \in N_i \setminus U_i$，则 σ_j 被 W 覆盖；又因为 $\sigma'_i = M(\sigma_j)$，可知 $\mathrm{Cr}(\sigma'_i)$ 在 σ_j 的覆盖时间被 $\pi_j/2$ 增加。因此，

$$\mathrm{Cr}(\sigma'_i) \ge \frac{1}{2} \cdot \sum_{j \in N_i \setminus U_i} \pi_j \tag{1}$$

现在我们分析在 $\sigma_j \in U_i$ 的绕行之前尚未覆盖的请求 σ'_i。注意：(i) 由于断言 $1/3$，这些请求各自以至少2的概率可达；(ii) U_i 中的可达请求显然可以通过一条长度为 S' 的环覆盖（根据 $S' = \Lambda_M$ 和 Λ_M 的定义）；(iii) ORIEN 保证了对最大奖励环的 ()-近似，因此我们得到，在 σ'_i 的绕行期间收集到的期望总奖励至少为 $1\,(2+\epsilon)$ ·$(1/3) \cdot \sum_{j:\ \sigma_j \in U_i} \pi_j$。由于其中一半奖励将存入 σ'_i，因此我们有

$$\mathbb{E}\left[\mathrm{Cr}(\sigma'_i)\right] \ge \frac{1}{6(2+\epsilon)} \cdot \sum_{j \in U_i} \pi_j \tag{2}$$

该断言由公式 (1) 和 (2) 推得。 □

该方　　定理 3.11 可通过上述主张完成。特别地，我们得到

$$\mathrm{Rew}(W, I, S) = \sum_{\sigma' \in I'} \mathrm{Cr}(\sigma') \geq \frac{\Omega(1)}{\rho_M} \cdot \mathrm{Rew}(W', I', S').$$

参考文献

[AGKK20] Antonios Antoniadis、Themis Gouleakis、Pieter Kleer 和 Pavel Kolev。秘书问题与带机器学习建议的在线匹配问题。载于神经信息处理系统年会（NeurIPS），2020。1[ALT21] Yossi Azar、Stefano Leonardi 和 Noam Touitou。不确定处理时间下的流动时间调度。载于计算理论研讨会（STOC），2021。1[AMN98] Esther M. Arkin、Joseph B. M. Mitchell 和 Giri Narasimhan。资源受限几何网络优化。载于计算几何研讨会（SoCG），1998。3[APT21] Yossi Azar、Debmalya Panigrahi 和 Noam Touitou。带预测的在线图算法。载于ACM-SIAM离散算法研讨会（SODA），2021。1, 3, 8[AV15] Yossi Azar 和 Adi Vardi。带时间窗和服务时间的旅行商问题。CoRR，abs/1501.06158，2015。3, 11, 26[BBCM04] Nikhil Bansal、Avrim Blum、Shuchi Chawla 和 Adam Meyerson。截止期限旅行商问题与带时间窗的车辆路径规划的近似算法。载于计算理论研讨会（STOC），2004。3[BBFL23] Magnus Berg、Joan Boyar、Lene M. Favrholdt 和 Kim S. Larsen。带权重预测的在线最小生成树。CoRR，abs/2302.12029，2023。1[BCK+03] Avrim Blum、Shuchi Chawla、David Karger、Terran Lane 和 Maria Minkoff。定向越野问题与折扣奖励旅行商问题的近似算法。载于计算机科学基础会议（FOCS），2003。3[BFM+21] Niv Buchbinder、Yaron Fairstein、Konstantina Mellou、Ishai Menache 和 Joseph (Seffi) Naor。具有生命周期和负载预测的在线虚拟机分配。载于计算机系统测量与建模国际会议（SIGMETRICS），2021。1[BLMS+22] Giulia Bernardini、Alexander Lindermayr、Alberto Marchetti-Spaccamela、Nicole Megow、Leen Stougie 和 Michelle Sweering。应用于在线图问题的输入预测通用误差度量。CoRR，abs/2205.12850，2022。3[BMS20] Étienne Bamas、Andreas Maggiori 和 Ola Svensson。学习增强算法的原始-对偶方法。载于神经信息处理系统年会（NeurIPS），2020。1[BNBYF+00] Amotz Bar-Noy、Reuven Bar-Yehuda、Ari Freund、Joseph Naor 和 Baruch Schieber。近似求解资源分配与调度问题的统一方法。载于计算理论研讨会（STOC），2000。29[BNGNS99] Amotz Bar-Noy、Sudipto Guha、Joseph Naor 和 Baruch Schieber。实时调度下多台机器吞吐量的近似。载于计算理论研讨会（STOC），1999。29

[BYES05] Reuven Bar-Yehuda、Guy Even 和 Shimon Shahar。关于带时间窗的几何奖品收集旅行商问题的近似。《算法杂志》，55:76– 92，2005年。3[CHP06] Ke Chen 和 Sariel Har-Peled。重新考察平面上的定向越野问题。载于计算几何研讨会（SoCG），2006年。3[CK04] Chandra Chekuri 和 Amit Kumar。带群组预算约束的最大覆盖问题及其应用。载于国际近似、随机化与组合优化研讨会（APPROX），2004年。3[CKP08] Chandra Chekuri、Nitish Korula 和 Martin Pál。定向越野及相关问题的改进算法。载于ACM-SIAM离散算法研讨会（SODA），2008年。3, 11, 19, 26, 27[DIL+21] Michael Dinitz、Sungjin Im、Thomas Lavastida、Benjamin Moseley 和 Sergei Vassilvitskii。通过学习得到的对偶解加速匹配。载于神经信息处理系统年会（NeurIPS），2021年。1[DLLV21] Paul Dütting、Silvio Lattanzi、Renato Paes Leme 和 Sergei Vassilvitskii。带建议的秘书问题。载于ACM经济学与计算会议（EC），2021年。1[EdLMS22] Thomas Wilhelm Erlebach、Murilo Santos de Lima、Nicole Megow 和 Jens Schloter。带不确定性的最小生成树学习增强型查询策略。载于欧洲算法研讨会（ESA），2022年。1[EFS+22] Jon C. Ergun、Zhili Feng、Sandeep Silwal、David Woodruff 和 Samson Zhou。学习增强型 k均值聚类。载于国际学习表征会议（ICLR），2022年。1[GJMZ16] Jie Gao、Su Jia、Joseph S. B. Mitchell 和 Lu Zhao。时间窗旅行商问题和奖品收集旅行商问题的近似算法。载于机器人算法基础研讨会（WAFR），2016年。3[GLS22] Themis Gouleakis、Konstantinos Lakis 和 Golnoosh Shahkarami。用于在线直线旅行商问题的学习增强型算法。CoRR，abs/2206.00655，2022年。3[HWL+22] Hsiao-Yu Hu、Hao-Ting Wei、Meng-Hsi Li、Kai-Min Chung 和 Chung-Shou Liao。带预测的在线旅行商问题。CoRR，abs/2206.15364，2022年。3[IKQP21] Sungjin Im、Ravi Kumar、Mahshid Montazer Qaem 和 Manish Purohit。带预测的非先知调度。载于ACM算法与体系结构并行研讨会（SPAA），2021年。1[JPS20] Zhihao Jiang、Debmalya Panigrahi 和 Kevin Sun。带预测的加权分页在线算法。载于自动机、语言与编程国际研讨会（ICALP），2020年。1[KN03] Yoshiyuki Karuno 和 Hiroshi Nagamochi。具有释放时间和处理时间的路径上多车辆调度问题的2-近似算法。《离散应用数学》，129:433–447，2003年。3

[LLMV20] Silvio Lattanzi、Thomas Lavastida、Benjamin Moseley 和 Sergei Vassilvitskii。通过学习得到的权重进行在线调度。载于ACM-SIAM离散算法研讨会（SODA），2020年。1[LM] Alexander Lindermayr 和 Nicole Megow。带预测的算法。https://algorithms-with-predictions.github.io/。1, 2[LMRX21] Thomas Lavastida、Benjamin Moseley、R. Ravi 和 Chenyang Xu。面向在线匹配、流和负载均衡的可学习且实例鲁棒的预测。载于欧洲算法研讨会（ESA），2021年。1[LV21] Thodoris Lykouris 和 Sergei Vassilvitskii。利用机器学习建议的竞争性缓存。《ACM杂志》，68(4):24:1–24:25，2021年。1, 2[Roh20] Dhruv Rohatgi。利用机器学习建议实现在线缓存的近最优界。载于ACM-SIAM离散算法研讨会（SODA），2020年。1[Tsi92] John N. Tsitsiklis。带时间窗的旅行商和维修工问题的特殊情形。Networks，22(3):263–282，1992年。3[XM22] Chenyang Xu 和 Benjamin Moseley。在线斯坦纳树的学习增强型算法。AAAI人工智能会议论文集（AAAI），36:8744–8752，2022年。1

附录

A 第 4 节中省略的证明4

引理 4.1. 对于满足 $L \geq 2S-2 \geq 1$ 的任意一对整数(L,S)，存在一个 *TW-TSP*(G,I) 以根节点为起点的实例，其服务成本满足 $L_{min}=L$ *and*

$$\textsc{Opt}(G,I,S)=\frac{1}{2S-1}\cdot\textsc{Opt}(G,I,1).$$

证明。考虑一个路径图 G，它有 $n=2S-1$ 个顶点，标记为 v_0 到 v_{n-1}。每个顶点 v_i 都通过长度为 $\alpha>0$ 的边与顶点 v_{i-1} 和 v_{i+1} 相连。接下来，考虑请求序列 $I=\{\sigma_i\}_{i=0}^{n-1}$，其中对于所有 $i=1,\ldots\ldots,\ n-1,\ \sigma_0=(v_0,0,L,1)$ 和 $\sigma_i=(v_i,d_i-L,d_i,1)$，且$d_i=i\alpha+2S-1$。因此，图 G 的每个顶点对应一个单位奖励请求，其时间窗长度为 L。边长 α 与分析无关，可以设为任何能保证释放时间非负的值，即 $\alpha=(L+1-2S)$。

注意，任何从 v_0 在 $t=0$ 出发、以S 的服务成本提供请求服务的游走，最多只能覆盖一个请求。事实上，在已服务一个请求的情况下，到达顶点 v_j 不会早于第 $\alpha\cdot j+S>d_j-S$ 步。因此，当实例 (G,I) 以顶点 v_0 为根时，$OPT(G,\ I,\ S)=1$。

另一方面，考虑这样一条游走：从 v_0 出发，依次访问 G 的所有顶点，并在每个顶点停留一个时间步。这样的游走会在每个顶点v_j 从 $t_j=j+\alpha\cdot j$ 一直停留到第 (t_j+1) 步。由于 $n=2S-1$，所以总有 $t_j\leq d_j-1$。此外，假设 $L\geq 2S-2$，则总有 $t_j\geq r_j$。因此，这条游走在单位服务成本模型中成功服务了所有请求，这意味着 $OPT(G,\ I,\ 1)=2S-1$。 □

B 第 5 节中省略的证明

引理 5.1。给定一个满足 $L_{min}\geq 4D$ 的 *TW-TSP*（不含服务时间）实例，存在一种多项式时间算法，可实现 $O(1)$ 近似。

证明。固定任意一个满足 $L_{min}\geq 4D$ 的 TW-TSP 实例（G,I）。我们首先说明，由于时间窗较大，我们可以假设（在常数因子的范围内）所有释放时间和截止时间都是 $K:=2D$ 的倍数。事实上，令 I' 为从 I 得到的请求序列，其中每个 $\sigma_i=(v_i,r_i,d_i,\pi_i)\in I$ 都替换为一个新请求 $\sigma_i'=(v_i,r_i',d_i',\pi_i)$，其中每个释放时间都向上取整到最接近的 K（$r_i'=K\cdot\lceil\frac{r_i}{K}\rceil$）的倍数，每个截止时间都向下取整到最接近的 K（$d_i'=K\cdot\lfloor\frac{d_i}{K}\rfloor$）的倍数。我们将这样的实例称为 K-对齐实例。注意，由于 $L_{min}\geq 4D=2K$，此变换不会使任何请求消失，这意味着 $d_i'\geq r_i'+K$。显然，由于 I' 除了时间窗更小之外，与 I 完全相同，因此我们有

$$\text{For any walk } W\in\mathcal{W}(G):\quad \mathrm{Rew}(W,I,0)\geq\mathrm{Rew}(W,I',0) \tag{3}$$

接下来，考虑实现最优奖励值 $\mathsf{Opt}(G,I,0)$ 的游走 $W^*\in\mathcal{W}(G)$。固定任意一个由 W^* 服务的请求 $\sigma_i\in I$，并令 $t_i\in[r_i,d_i]$ 为该请求开始被服务的步骤。我们区分以下三种情况：

1. 如果 $t_i\in[r_i',d_i']$，那么显然，游走 W^* 也服务请求 $\sigma_i'\in I'$。

2. 如果 $t_i < r'_i = K \cdot \lceil \frac{r_i}{K} \rceil$，那么考虑游走 W^*_+，它与 W^* 完全相同，不同之处在于它最初空闲了 K 步。这样的游走会在 $t'_i = t_i + K$ 访问顶点 v_i。不难看出，根据 r'_i 和 d'_i 的定义以及 $d'_i \geq r'_i + K$ 这一事实，我们很容易得到 $t'_i \in [r'_i, d'_i]$，因此游走 W^*_+ 服务请求 $\sigma'_i \in I'$。

3. 同样，如果 $t_i > d'_i = K \cdot \lfloor \frac{d_i}{K} \rfloor$，我们考虑游走 W^*_-，它与 W^* 完全相同，唯一的区别是，它在 $t = K$ 时 W^* 所处的位置提前开始。这样的游走会在 $t'_i = t_i - K$ 访问顶点 v_i。同样，我们很容易验证 $t'_i \in [r'_i, d'_i]$，因此游走 W^*_- 服务请求 $\sigma'_i \in I'$。注意，如果 $t_i < K$，这一论证就不成立，因为我们跳过了 W^* 的前 K 个时间步；不过，根据 $t_i > d'_i \geq K$，我们知道这种情况绝不会发生。

因此，我们已经论证，对于最优游走 W^* 覆盖的每个请求 $\sigma_i \in I$，都存在三种游走之一能够服务请求 $\sigma'_i \in I'$。这意味着

$$\textsc{Opt}(G, I', 0) \geq \frac{1}{3} \cdot \textsc{Opt}(G, I, 0) \tag{4}$$

由 (3) 和 (4) 可立即得出，任何针对 G 对齐实例 (G, I') 的 α-近似算法，都可以直接转化为针对实例 (G, I) 的 3α-近似算法。下一步，我们通过将问题非标准地归约为定向越野问题，来获得适用于 K 对齐实例上 TW-TSP 的常数近似算法。我们指出，[AV15] 中采用了类似的方法。我们的算法见算法 3；正如我们将很快证明的，它是适用于 K 对齐实例的 6-近似算法。综合以上结果，当 $L_{min} \geq 4D$ 时，我们设计出了一个适用于 TW-TSP 的 18-近似算法，如引理所述。

算法 3 适用于 $2D$ 对齐实例的 TW-TSP 常数近似算法。

1: 将时间范围划分为长度为 K 的阶段，其中阶段 i 覆盖步骤 $[(i-1)K, iK]$。2: 设定 $C \leftarrow \emptyset$。

3: 在阶段 i 开始时，执行：

4: $Rem(i) \leftarrow \{\sigma_j \in I : r_j \leq (i-1)K$ 和 $d_j \geq i \cdot K\} \setminus C$。5: $P_i \leftarrow$ 在图 G 上运行定向越野问题子程序的输出，其中包含请求 $Rem(i)$，预算为 $K/2$。6: 从当前位置 v_0 移动到路径 P_i 的根节点 v_i，并执行该路径 P_i。7: 在该阶段剩余的 $K - K/2 - \ell(v_0, v_i) \geq 0$ 个时间步内保持空闲。

算法 3按长度为 K的阶段运行。注意，由于对齐假设，在每个阶段开始时，我们都知道该阶段内可用的全部请求，因为没有请求会在不是 K倍数的时间步到达。我们用 $Rem(i)$ 表示阶段 i 中尚未被算法服务的这些请求集合。显然，这正是算法在阶段i内能够服务的最大请求集合。此外，所有截止时间都不会早于阶段结束时到期，因此阶段内请求的服务顺序并不重要。

因此，我们的算法需要决定：通过服务 $Rem(i)$ 中的请求，并使用长度至多为 $K = 2D$ 的路径，最多能获得多少奖励。这正是广为研究的 定向越野问题 的定义，它是 $r_i = 0$ 且$d_i = D$ 对所有 i 均成立时 TW-TSP 的特例。由于算法在阶段开始时所处的位置可能不便，我们转而寻找长度为 $K/2$ 的最佳路径；其思路是，无论返回游走的根位于图上的何处，其余 $K/2 = D$ 个时间步都可用于前往该根。对于定向越野问题的子程序，我们采用 2.5-近似确定性算法，该算法由 [CKP 08] 提出。

分析。现在我们可以分析算法 3 的性能了。我们用 Alg(G, I) 表示算法获得的总奖励，用 Opt(G, I) 表示最优奖励。此外，我们将 Adv 定义为在 (G, I) 上运行的最优离线算法，并限制其不能从服务 Alg 已覆盖的请求中获得任何奖励。因此，若用 Adv(G, I) 表示这一辅助算法的总奖励，则有 Opt$(G, I) \leq$ Alg$(G, I) +$ Adv(G, I)。

现在固定某个阶段 i，并令 $A(i) \subseteq I$ 表示 Alg 覆盖的请求集合，令 $B(i) \subseteq I$ 表示 Adv 覆盖的请求集合。关键观察是，$B(i)$ 中的所有请求都必须满足：(i) 在阶段 i 开始时发布；(ii) 在阶段 i 结束后过期；(iii) 在 Alg 的整个执行过程中都未被覆盖。因此，我们得到 $B(i) \subseteq Rem(i)$。然而，这意味着存在一条长度为 K 的路径（即 Adv 在阶段 i 中所走的路径），它从 $Rem(i)$ 中的请求获得总奖励$\pi(B(i))$（稍微滥用一下记号）；因此，也存在一条长度为 $K/2$ 、总奖励为 $\pi(B(i))/2$ 的路径，其请求完全来自 $Rem(i)$。

我们的论证基于以下事实得以完成：$A(i)$ 中的请求是通过一个针对定向越野问题的 2.5-近似算法计算得到的。根据所用子程序的近似保证，我们知道，任何长度为 $K/2$ 、覆盖了 $Rem(i)$ 中请求的路径，其总奖励至多为 $2.5 \cdot \pi(A(i))$。因此，我们得到 $\pi(B(i)) \leq 5 \cdot \pi(A(i))$，进而 Alg$(G, I) \geq 0.2 \cdot$ Adv(G, I)。结合 Opt$(G, I) \leq$ Alg$(G, I) +$ Adv(G, I)，可知算法 3 对具有 $K = 2D$ 的 K-aligned 实例是一个 6- 近似算法，证明完毕。 □

引理 5.　2. 存在一个针对 *TW-TSP* 的 $O(\log \min(D, L_{\max}))$ 近似算法 .

证明。 固定任意一个 TW-TSP 实例 (G, I) 。我们将请求序列 I 划分为两个子序列 I_1 和 I_2，使得 I_1 包含所有时间窗为 $\sigma_j \in I \quad d_j - r_j \geq 4D$ 的请求，而 I_2包含其余所有请求。由于 I_1 中的所有请求的窗口长度至少为 $4D$，我们可以使用引理 5.1 中的算法，得到一个游走 W_1 ，使得

$$\mathrm{Rew}(W_1, I_1) \geq \frac{1}{\mathcal{O}(1)} \cdot \mathrm{OPT}(G, I_1)$$

此外，由于 I_2 中的所有请求的窗口长度至多为 $4D$，我们可以使用$\mathcal{O}(\log L_{max})$-近似算法 [CKP08] ，得到一个游走 W_2 ，使得

$$\mathrm{Rew}(W_2, I_2) \geq \frac{1}{\mathcal{O}(\log D)} \cdot \mathrm{OPT}(G, I_2)$$

显然，由于 I_1和 I_2构成了 I的一个划分，因此 Opt$(G, I_1) +$ Opt$(G, I_2) \geq$ Opt(G, I)。此外，对于每个 $j \in \{1, 2\}$，都有 $I_j \subseteq I$，从而 Rew$(W_j, I) \geq$ Rew(W_j, I_j)。结合这些不等式可立即得出：若输出 $\{W_1, W_2\}$ 中奖励最高的游走，则可以保证获得 Opt(G, I) 的 $\mathcal{O}(\log D)$-近似。再结合 [CKP08], 的 $\mathcal{O}(\log L_{\max})$ 近似，即得引理。 □

引理 5.3。 给定一个具有服务时间且满足 $S \leq D$ 的 *TW-TSP* 实例(G, I, S)，存在一个多项式时间算法，可实现 $O(\log \min(L_{\max}, D))$ 近似。

证明。 固定任意一个服务成本为 S 的 TW-TSP-S 实例 (G, I)。我们按以下方式从 G 构造一个新图 G_0:

- 对于每个请求 $\sigma_i = (v_i, r_i, d_i, \pi_i) \in I$，我们在 G 上添加一个新顶点 v_{σ_i}。因此，我们有$V(G_0) = V(G) \cup \{v_{\sigma_i} | \sigma_i \in I\}$。
- 对于每个添加的顶点 v_{σ_i}，我们添加一条将其连接到顶点 v_i 的新边。因此，我们有 $E(G_0) = E(G) \cup \{(v_i, v_{\sigma_i}) | \sigma_i \in I\}$。
- 我们将每条已有边的边长加倍，并将每条新添加边的边长设为 S，即 $\ell'_e = 2\ell_e$ 对所有 $e \in E$ 和 $\ell'(v_i, v_{\sigma_i}) = S$ 对所有 $\sigma_i \in I$。注意，图 G_0 的直径为 $D_0 \leq 2(D + S)$。

接下来，我们从 I 构造请求序列 I_0，将每个 $\sigma_i = (v_i, r_i, d_i, \pi_i) \in I$替换为一个新的请求 $\sigma'_i = (v_{\sigma_i}, 2r_i + S, 2d_i - S, \pi_i) \in I_0$。注意，$I_0$ 的最大时间窗长度为$L^0_{\max} \leq 2L_{\max}$。如我们将看到的，带服务成本 S 的 TW-TSP-S 实例 (G, I) 与不带服务成本的 TW-TSP 实例 (G_0, I_0) 在本质上是等价的。确立这一点后，我们可以使用引理 5.2 中针对 TW-TSP 的近似算法，以 $\mathcal{O}(\log \min(L^0_{\max}, D_0)) = \mathcal{O}(\log \min(L_{\max}, D + S))$ 的近似因子求解实例 (G_0, I_0)，从而立即得到一个针对带服务成本 S 的 TW-TSP-SC 的 $\mathcal{O}(\log \min(L_{\max}, D + S))$-近似算法。最后，在假设 $S \leq D$ 下即可得到引理 5.3 的证明。

在证明的其余部分，我们将证明，(G, I, S) 的任意解都可以转换为 $(G_0, I_0, 0)$ 的一个奖励相同的解，反之亦然。根据我们对图 G_0 和序列 I_0 的定义，这几乎可以直接得出。设 $W_0 \in \mathcal{W}(G_0)$ 是任意一条游走，它服务若干服务成本为零的请求$\{\sigma'_i\} \in I_0$。如果 W_0 服务某个请求 $\sigma'_i \in I_0$，那么我们知道它会在 $[2r_i + S, 2d_i - S]$ 的某个时刻访问顶点 v_{σ_i}。由于到达顶点 v_{σ_i} 只有一种方式，我们可以推断，游走 W_0 不会早于 $t = 2r_i$ 访问顶点 v_i，并且会在 $2S$ 个时间步之后、最迟不晚于 $t = 2d_i$ 返回该顶点。

设 $W \in \mathcal{W}(G)$ 是一条游走，其访问顶点的顺序与 W_0 相同，但它不会进行形如 (v_i, v_{σ_i}, v_i) 的绕行，而是停留在 v_i 上。此外，由于边被加倍，所有时间都减半。由于 W_0 在 $t = 2r_i$ 之后访问每个顶点 v_i，并进行长度为 $2S$ 的绕行，在 $t = 2d_i$ 时返回，我们立即得到，这条游走 W 将以 S 的服务成本服务请求 $\sigma_i \in I$。因此，W 在实例 (G, I, S) 中获得的奖励将与 W_0 在实例 $(G_0, I_0, 0)$ 中获得的奖励相同。

另一个方向的证明（从 G 中服务成本为 S 的游走出发，并得到 G_0 中奖励相同、服务成本为 0 的游走）完全类似，因此省略。 □

引理 5.4. 给定一个 *TW-TSP* 实例 (G, I, S)，其服务成本满足 $S \leq D$，则存在一种多项式时间算法，可实现 $O(1)$ 近似。

证明。 任取一个服务成本为 $S \geq D$ 的 TW-TSP-SC 实例 (G, I)。我们首先将该实例归约为均匀图的情形。令 G_U 为一个新图，它由 G 构造而成：连接所有顶点，并将所有边长增大到 G 的直径；也就是说，G_U 是一个边长为 D 的完全图，其中 $V(G_U) = V(G)$。注意，由于我们只增大了距离，实例 (G_U, I) 至少与实例 (G, I) 一样难，这意味着对于任意游走 $W_U \in \mathcal{W}(G_U)$，都存在另一个游走 $W \in \mathcal{W}(G)$，使得

$$\mathrm{Rew}(W, G, I, S) \geq \mathrm{Rew}(W_U, G_U, I, S), \tag{5}$$

也就是说，游走 W 遵循 W_U，并在 G_U 中的距离大于 G中的距离时，加入适当的空闲时间。

接下来，我们将证明这些实例的最优值最多相差一个常数因子。为此，设 $W^* \in \mathcal{W}(G)$ 为实现最优奖励 Opt(G， I， S) 的游走，并设 $\{\sigma_i\} \subseteq I$为 W^* 所覆盖的请求序列，按它们被 W^*访问的先后顺序排列。现在，我们将构造两个游走 W_o，$W_e \in \mathcal{W}(G_U)$，分别对应实例 (G_U， I)，分别服务所有奇数编号请求 σ_{2i+1} 和所有偶数编号请求 σ_{2i}。

由于任意两个顶点之间的距离最多增加 D G 到 G_U，且服务成本满足 $S \geq D$，因此，在序列 $\{\sigma_i\}$ 中跳过每隔一个请求的服务时间并遵循 W^*，就能获得足够的时间来弥补距离的增加。形式上， W_e 由 W^* 推导而来：在每个奇数位请求 σ_{2i+1} 上跳过空闲时间 $S' - \ell\left(v_{2i+1}, v_{2i+2}\right) \geq 0$；同样， W_o 由 W^* 推导而来：在每个偶数位请求 σ_{2i}上跳过空闲时间 $S' - \ell\left(v_{2i}, v_{2i+1}\right) \geq 0$。需要指出的是，对于 W_o，为了服务第一个请求（因为无法在它之前跳过时间），我们只需将游走的起点改为顶点 v_1，并添加适当的空闲时间。因此，我们得到

$$\textsc{Opt}(G_U, I, S) \geq \frac{1}{2} \cdot \textsc{Opt}(G, I, S) \tag{6}$$

由（5）和（6），我们立即得到：对于均匀图上的 TW-TSP-SC，任何 α-近似算法都可直接推出一个适用于一般图的 2α-近似算法，其 $D \leq S$。证明还基于这样一个事实：均匀图上的 TW-TSP-SC 与 [BNGNS99] 和 [BNBYF+00]中研究的作业调度问题本质上等价，而该问题已有常数近似算法。

注意到所有距离均设为 D，因此可以从问题的表述中去掉旅行这一部分。具体而言，我们不再要求每个请求 σ_i 在 $t \in [r_i, d_i - S]$的某个时间区间 $[t, t+S]$ 内得到服务，而是要求它在某个时间区间$[t, t+S+D]$ 内得到服务，其中 $t \in [r_i, d_i - S]$ 还包含前往下一个请求所需的旅行时间 D。因此，我们可以将每个请求 σ_i 视为一个作业 J_i，其释放时间为 r_i，截止时间为$d_i + D$，处理时间为 $S + D$，奖励为 π_i。于是，我们的目标是在单机上调度这些作业，使它们彼此不重叠，并最大化在各自时间窗内处理的作业所获得的总奖励。这正是 [BNGNS99], 中研究的问题表述；该文给出了一个 3-近似确定性算法，后来 [BNBYF+00] 将其改进为 $(2+\epsilon)$-近似确定性算法。无论如何，使用这些算法中的任意一种，我们都能得到一个$\mathcal{O}$(1)-近似算法来解决均匀图上的 TW-TSP-SC，从而完成引理的证明。 □

C 第 3 节中省略的证明3

推论 3.9。 给定一个(G, I, I', M) 实例，该实例是带预测的时间窗旅行商问题，其中请求的奖励是随机抽取的，且预测误差满足 $\tau_M \leq L_{\min}/2$ 以及 $\Lambda_M \leq (L_{\min} - 1)/4$，则存在一个多项式时间在线算法，它将元组 (G, I', Λ_M) 作为离线输入，并将 I 作为在线输入，从而构造出游走 $W \in \mathcal{W}(G)$，使得

$$\mathbb{E}[\mathrm{Rew}(W, I)] \geq \frac{1}{O(\Lambda_M \cdot \rho_M^2 \cdot \log \min(D, L_{\max}))} \cdot \textsc{Opt}(G, I)$$

证明。 回顾一下，根据定义，在随机奖励设定中，我们有

$$\textsc{Opt}(G, I) = \arg\max_{W \in \mathcal{W}(G)} \left(\mathbb{E}\left[\sum_{\sigma \in \mathrm{Cov}(W, I)} \pi_\sigma \right] \right) = \arg\max_{W \in \mathcal{W}(G)} \left(\sum_{\sigma \in \mathrm{Cov}(W, I)} \mathbb{E}\left[\pi_\sigma\right] \right) = \textsc{Opt}(G, I_{det}) \tag{7}$$

其中 I_{det} 是确定性请求序列，其中每个请求 $\sigma \in I$ 都具有确定性奖励 $\mathbb{E}[\pi_\sigma]$，而非随机奖励 π_σ。需要说明的是，在交换求和与期望的顺序时，我们利用了这样一个事实：只有 I 中的奖励是随机的，因此对于任意游走 $W \in \mathcal{W}(G)$，集合 $\mathsf{Cov}(W,\ I)$ 是确定性的。

回顾一下，在这个随机设定中，ρ_M 刻画了期望奖励 $\mathbb{E}[\pi_\sigma]$与预测奖励 $\pi_{\sigma'}$ 之间的最大差异，因此等于 I' 与I_{det}之间的奖励误差。此外，位置误差和时间窗口误差与奖励无关，因此在(I, I') 和(I_{det}, I')之间完全相同。因此，我们可以直接应用定理3.1，得到一个在线游走 $W \in \mathcal{W}(G)$ ，使得

$$\mathbb{E}\left[rew(W, I_{det})\right] \geq \frac{1}{O(\Lambda_M \cdot \rho_M^2 \cdot \log\min(D, L_{\max}))} \cdot \textsc{Opt}(G, I_{det}), \tag{8}$$

其中，期望是对算法的随机选择取的。

最后，对于任意游走 $W \in \mathcal{W}(G)$，有

$$\mathrm{Rew}(W, I_{det}) = \sum_{\sigma \in \mathrm{Cov}(W, I_{det})} \mathbb{E}\left[\pi_\sigma\right] = \mathbb{E}\left[\sum_{\sigma \in \mathrm{Cov}(W, I)} \pi_\sigma\right] = \mathbb{E}\left[\mathrm{Rew}(W, I)\right] \tag{9}$$

因为 覆盖(W, I_{det}) =覆盖(W, I)。在这种情况下，期望仅针对I的随机性取。证明由等式(7)至(9)可得。 □

推论 3.10. 给定一个带预测的时间窗旅行商问题实例(G, I, I')，令 M为 I与 I'之间任意一个（不完全）匹配，并令误差参数 $\Lambda_M, \rho_M, \tau_M, \Delta_1^M$和 Δ_2^M如上定义。那么，存在一个在线算法，以(G, I', Λ_M)作为离线输入，以 I作为在线输入，并返回一个游走 $W \in \mathcal{W}(G)$ ，使得

$$\mathbb{E}\left[\mathrm{Rew}(W, I)\right] \geq \Omega\left(\frac{1}{\Lambda_M \cdot \rho_M^2 \cdot \log\min(D, L_{\max})}\right) \cdot \left(\textsc{Opt}(G, I) - \Delta_1^M\right) - \frac{\Delta_2^M}{\rho_M}.$$

证明。令 $I_1 \subseteq I$ 为 I 中仅由与 I'中的某个预测请求相匹配的真实请求组成的请求序列。注意，引理 3.2 和引理 3.4 与 M 和 I 之间的匹配 I'无关。此外，引理 3.3只要求I 中的所有请求在 I'中都有一个匹配的预测，因此可以直接应用于 I_1。综合以上结果，我们可以预先计算一条游走 $W' \in \mathcal{W}(G)$，使得

$$\mathrm{Rew}(W', I', S') = \Omega\left(\frac{1}{\Lambda_M \cdot \rho_M^2 \cdot \log\min(D, L_{\max})}\right) \cdot OPT(G, I_1)$$

对于 $S' \geq 2\Lambda_M + 1$。接下来，令 $I_2 \subseteq I'$ 为仅由与某个真实请求匹配的预测组成的预测请求序列。由于 I_2 中的所有预测都与 I中的某个真实请求匹配，因此我们可以使用引理3.5 以游走 W' 为输入来构造一个在线游走 $W \in \mathcal{W}(G)$ 使得

$$\mathbb{E}\left[\mathrm{Rew}(W, I)\right] \geq \frac{1}{6\rho_M} \cdot \mathrm{Rew}(W', I_2, S')$$

该推论由未匹配请求的奖励 Δ_1^M和 Δ_2^M的定义可得，这意味着 $\mathsf{Opt}(G, I_1, S) \geq \mathsf{Opt}(G, I, S) - \Delta_1^M$ 且 $\mathsf{Rew}(W', I_2, S') \geq \mathsf{Rew}(W', I', S') - \Delta_2^M$。 □