arXiv:2303.15608v1 [cs.DS] 27 Mar 2023

克服迷失方向的线性搜索中的概率性故障 ⋆ ⋆⋆

Konstantinos Georgiou[1], Nikos Giachoudis[1], 和 Evangelos Kranakis[2][0000−0002−8959−4428]

[1] 数学系，多伦多都会大学，多伦多，安大略省，加拿大
konstantinos@torontomu.ca，ngiachou@gmail.com[2] 计算机科学学院，卡尔顿大学，渥太华，加拿大kranakis@scs.carleton.ca

摘要。 我们研究移动代理对位于无限直线上的隐藏静止目标进行搜索的问题。可行解是指所有智能体最终都能到达目标的智能体轨迹。我们的问题具有一个特殊特征：智能体存在p故障，即每次改变方向的尝试都是一次独立的伯努利试验，其已知概率为p，其中p是转向失败的概率。我们希望找到能够使最坏情况下的期望终止时间相对于隐藏目标到原点的距离最小的智能体轨迹（竞争分析）。因此，使用一个0-故障型智能体进行搜索就是著名的线性搜索（奶牛小径）问题；通过确定性算法和随机化算法，该问题分别可以达到最优的9和4.59112竞争比。

首先，我们研究一个确定性的 p-故障型智能体进行线性搜索的情形，即无法访问随机预言机，且 $p \in (0,1/2)$。对于这一问题，我们给出了能够将概率性故障转化为算法优势的轨迹。我们最强的结果涉及一种搜索算法（除对抗性概率故障外，该算法是确定性的）：当 $p \to 0$ 时，其性能达到最优值 $4.59112+\epsilon$，仅相差可任意小的加性项 ϵ。此外，当 $p \leq 0$ 时，其性能小于 9。390388。当 $p \to 1/2$ 时，我们的算法具有 $\Theta(1/(1-2p))$ 的性能；我们还证明，该结果在常数因子范围内是最优的。

其次，我们考虑两个 p-故障型智能体进行线性搜索的情形，其中 $p \in (0,1/2)$。对于这一问题，我们提供了三种各具优势的算法；即使当 $p \to 1/2$ 时，它们都具有有界的竞争比。事实上，对于这一问题，我们证明，无论底层通信模型如何，智能体都能够模拟任意 0-故障型智能体（确定性的或随机化的）的轨迹。因此，使用两个智能体进行搜索，可以实现竞争比为 $9+\epsilon$ 的解（我们证明，通过任意提高集中度即可达到该竞争比），或者实现竞争比为 $4.59112+\epsilon$ 的解。我们的最后一项贡献是一种用于两个 p-故障型智能体搜索的新算法，该算法实现了竞争比 $3+4\sqrt{p(1-p)}$，并具有任意高的集中度。

关键词： 线性搜索 · 概率性故障 · 移动代理

1 引言

线性搜索是指这样一个问题：在实数轴上某个未知位置放置了一个点目标，搜索者需要找到它。搜索者是一个移动代理，能够以最高 1 的速度移动，并从实数轴的原点开始搜索。目标是在最短时间内找到该目标。这个搜索问题为理解探索实数轴的极限提供了一种范式，并在数学和理论计算机科学中有重要应用。

本文研究故障智能体执行线性搜索的情形。该智能体会迷失方向：尝试改变方向时，它不仅可能无法成功转向，而且也无法察觉运动方向已经改变。更准确地说，对于某个 $0 \leq p \leq 1$，成功转向的概率为 $1-p$，但智能体必须到达一个锚点后才能察觉是否转向成功。锚点是实数轴上一个已知且预先指定的点。尽管智能体存在这种故障行为，令人颇感意外的是，我们仍能设计出优于著名之字形算法的算法；后者的竞争比为 9。

⋆ 研究部分得到了 NSERC 和数学科学研究菲尔兹研究所的支持。⋆⋆ 本文是同名论文的完整版本，该论文将发表于第 30 届结构信息与通信复杂性国际研讨会（SIROCCO'23）论文集。会议将于 2023 年 6 月 6 日至 9 日在西班牙阿尔卡拉-德埃纳雷斯举行。

1.1 相关工作

在实数轴上由单个智能体进行搜索的研究，近 50 年前由 Bellman [9] 和 Beck [6,7,8] 独立开创；作者证明了一个著名结论：最大速度为 1 的单个搜索者，无法在少于 $9d$ 的时间内找到距搜索者初始位置 d 的隐藏目标。这些论文催生了线性搜索的诸多变体。Baeza-Yates 等人 [3,4] 研究了智能体在其他环境中的搜索问题，例如在平面上，或从 w 条并发射线的原点出发的情形（也称为“迷失的小牛”问题）。群体搜索研究始于 [11]，其中研究了多个能够面对面交流的智能体如何进行疏散（这类问题类似于搜索，但目标是最小化最后一个智能体到达目标所需的时间）。文献 [17] 研究了该问题的一种扩展，即最小化疏散时间的加权平均值。关于这一主题已有大量文献，而 [12] 则简要综述了近期搜索研究中的一些课题。

存在多个智能体且其中部分可能出现故障（崩溃故障或拜占庭故障）的线性搜索，分别由 [15] 和 [13], 的工作开创。在这一研究主题中，人们利用通信的力量来克服故障的影响。对于三个智能体且其中一个为拜占庭智能体的情况，[21] 证明，可以分析 [15] 中提出的比例调度，从而得到 8.653055 的上界。最近，[14] 针对 n 个智能体提出了一类新的算法，适用于其中拜占庭故障智能体的数量接近多数的情况，并给出了在无限直线上三个智能体且其中一个为拜占庭智能体时已知的最佳上界 7.437011。

本文聚焦于概率搜索。前文提到的 Bellman [9] 和 Beck [6,7,8], 的工作也以概率为重点。此外，关于线性搜索概率模型的众多研究主题可见于搜索博弈与会合一书 [2]，以及 [1,20]。

考虑智能体转向成本的搜索是 [2][第 8 节4] 以及论文 [16]的研究重点。不确定检测下的搜索见 [2][第 8 节6]。根据这一模型，搜索者到达目标时并不一定能找到它；相反，假设搜索者第 k 次访问目标时找到它的概率为 p_k，其中 $\sum_{\mathrm{k}\geq 0} \mathrm{p}_k = 1$。其中一个特例是具有几何检测概率的搜索，见 [2][第 8.6 节2]；在该情形下，搜索者在第 k 次访问时找到目标的概率为 $(1-\mathrm{p})^{k-1}\mathrm{p}$。[19] 研究了对一维随机游走者的搜索，并且 [5]关注在起始点留下标记时的会合搜索。关于另一类概率性故障的研究结果中，[10] 研究了半直线（或 1-射线）上的问题，其中目标检测会发生故障，即每次访问都是一次成功概率已知为 p 的独立伯努利试验。回到无限直线上的搜索，[18] 给出了一种竞争比为 4.59112 的奶牛路径问题随机化算法，并证明该算法也是最优的。从严格意义上说，本文的结果是 [3] 中确定性搜索以及 [18] 中随机化搜索的最优解的直接延伸。据我们所知，本文所考虑的线性搜索问题此前从未被研究过。我们在第 2 节中正式定义问题。随后在第 3 节中，我们进一步阐述研究结果对于 [3] 和 [18]的相关性。

2 模型与问题定义（p-PDLS）

我们引入并研究所谓的概率性迷失方向线性搜索问题（简称 p-PDLS），该问题与某个概率p相关。我们对广为研究的线性搜索问题（也称为奶牛路径问题）进行推广，其中搜索者的轨迹决策会出现概率性故障。数值p 用于量化概率性失效（迷失方向）这一概念。

在p-PDLS 中，智能体（搜索者）可以在一条无限直线上以单位速度移动，改变方向不会产生额外代价。直线上有两个可与其他点区分的点：一个是原点，即智能体的起始位置；另一个是目标，即智能体要搜索的对象，当智能体走过该点时便可探测到它。

智能体存在故障行为。如果智能体尝试改变方向（即使是在停止之后），那么以已知概率 p，智能体将失败，并且仍会朝同一方向移动。后续改变方向的尝试是相互独立的伯努利试验，成功概率为 $1-p$。此外，智能体对每次伯努利试验的结果都是无感知的，也就是说，智能体并不知道自己是否成功改变了方向。我们将这种概率行为视为智能体的一个协程，该协程以概率 p执行失败。对于给定的 p，满足这一性质的智能体称为p故障代理。此外，为简单起见，我们假设在最开始时，p故障代理会无故障地开始朝某个特定方向移动。

智能体的故障行为可以通过利用原点和目标来恢复其方向感而得到补偿。事实上，假设智能体经过原点，并在时间 1 之后决定改变方向。再经过 1 个时间单位，智能体要么已经到达原点，此时它意识到自己成功转向；要么看不到原点，此时它检测到自己转向失败。我们将在后文进一步阐述这一想法。

p-PDLS 的解由智能体的轨迹（或多个智能体的轨迹）给出，即指示智能体在特定时间转向的指令，这些时间可能取决于先前的观测（例如，是否访问过原点以及访问时间）。可行轨迹是指能够以概率 1（迟早）访问无限直线上每个点的轨迹（因此，1-故障智能体不存在可行轨迹）。

对于直线上的一个点 $x \in \mathbb{R}$（目标），我们用竞争分析来定义可行轨迹的终止成本。具体而言，若 $E(x)$ 表示最后一个智能体到达目标 x 所需时间的期望值（若由单个智能体搜索，则即为该智能体到达目标 x 所需时间的期望值），其中期望是针对概率性故障，甚至也针对算法随机选择而言的，那么目标（输入）x 的终止成本定义为 $E(x)/|x|$。因此，可行轨迹的竞争比定义为 $\limsup_x E(x)/|x|$。

为简单起见，我们对线性搜索竞争比的定义偏离了标准定义。标准定义将性能定义为在绝对值不超过某个常数的 x 范围内取上确界 d，通常为 $d = 1$。不过，显然这两种度量的差异至多为 ϵ，对任意 $\epsilon > 0$ 皆如此。标准的重新缩放技巧（例如参见 [17]）说明了为什么 d 的取值并不重要，重要的是算法已知 d。

两个智能体搜索时的设定：当两个 p-故障智能体进行搜索时，两者的 p 值相同，并且假设它们所表现出的概率性故障是相互独立的伯努利试验。两个 p-故障智能体可以在无线模型或面对面模型中进行搜索。在前一种模型中，我们假设智能体能够即时交换消息；而在面对面模型中，只有智能体处于同一位置时才能交换消息。

在两种通信模型中，我们都假设两个智能体能够检测到彼此相遇（以及相遇的时刻）。因此，我们自然假设，在相遇时，一个 p 故障智能体也可以充当特殊点（与原点相同），从而帮助另一个智能体转向。下文中，我们将协助转向的智能体称为跟随者，将执行转向的智能体称为领导者。只要智能体能够确定这两个角色（这将在我们的算法中实现），我们还假设，领导者朝任意方向移动时，可以“带上”遇到的另一个故障智能体，使两个智能体继续朝该方向移动。这意味着，即使非领导者处于静止状态，在某一点相遇的两个智能体也能继续朝领导者的方向移动（概率为 1）。这一性质的依据是，实际上领导者并未改变方向，因此不存在出错的风险。最后，我们注意到，两个 p 故障智能体彼此独立地以至多 1 的速度移动；此外，它们中的任何一个都可以减速，甚至停在原地，同时仍遵循故障转向规范。

2.1 关于算法随机性与对抗性随机性的注记

我们所考虑的算法（可行轨迹）要么是确定性的，要么是随机化的，并且独立于故障性所诱导的随机性。具体而言，效率度量的定义方式保持不变，其中任何期望都在基础概率空间上计算（该概率空间由概率性故障与可能的随机化算法选择共同诱导）。此外，如果算法是随机化的，则还会使用一种独立于故障行为的额外随机机制。例如，随机化智能体（算法）可以均匀随机地选择一个介于 0 和 1 之间的数。能够访问随机预言机的 p 故障智能体将被称为随机化的，否则称为确定性的 （但二者都会表现出概率性失败的回合）。

根据我们的定义，带有一个确定性智能体的 0-PDLS 就是著名的线性搜索问题（奶牛小径问题），该问题存在竞争比可证明最优的轨迹 9 [3]。在另一个极端情况下，1-PDLS 不存在可行解，因为智能体会沿一个方向无限移动。类似地，我们接下来在引理 1 中证明，只有当问题具有意义时，$p < 1/2$，参见第 A 节中的证明。

引理 1。 当 $p \geq 1/2$ 时，p-PDLS 的任何轨迹都没有有界的竞争比。

需要特别指出的是，在我们的模型中，智能体的概率性故障转向确实会妨碍对轨迹的控制，同时也会给对手的算法策略带来不确定性。因此，只要 $p > 0$，智能体的概率性移动即使不受算法控制，也可以被视为一种已确定的算法选择。因此， [18] 的否定性结果意味着我们的问题具有如下下界。

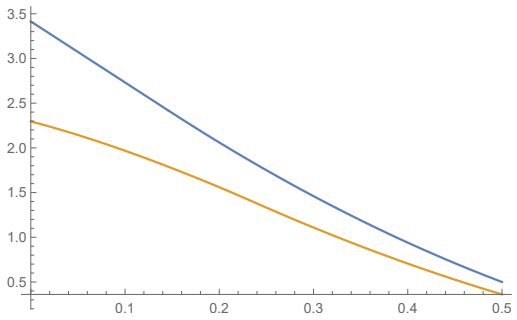
推论1。 对于任意 $p \in (0,1/2)$，不存在针对 p-PDLS 且包含单个智能体（确定性或随机化的）的解决方案，其竞争比低于 *4.59112*。

3 贡献概述与一些初步结果

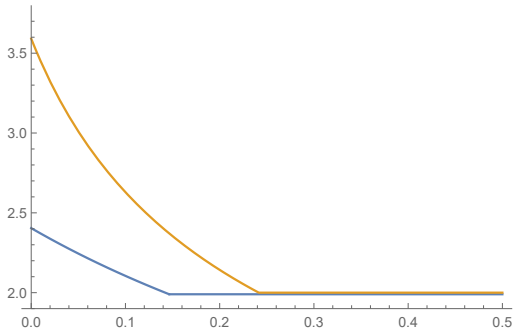
本文的主要结果涉及（一个或两个）故障智能体针对p-PDLS所能达到的竞争比上界。

3.1 使用一个故障智能体进行搜索的结果概述

我们首先介绍针对一个 p-故障智能体的搜索算法。第 4 节分析一种确定性搜索算法的性能，其性能总结于第 2 页的定理 6中。本节为使用一个随机化的 p-故障智能体进行搜索时所需的主要结果计算作铺垫。实际上，在第 5 节中，我们提出一种随机化的 算法，其性能总结于第 3 页的定理 7中。读者可以在定理的正式表述中看到相关的复杂公式，因此我们在此通过图 1以图形方式概述结果。这里有一些重要的观测。



(a) 定理2 中的确定性算法（蓝色）与定理3 中的随机化算法（黄色）的竞争比对比。竞争比按 $(1/2 - p)$ 缩放。



(b) 定理2 中的确定性算法（蓝色）与定理3 中的随机化算法（黄色）所用之字形算法的预期扩张因子。

图 1：正向结果的图示总结： 关于使用一个智能体进行搜 g 使用一个 p-故障代理，$p \in (0,1/2)$。

关于扩张因子的说明：我们强调，我们的两种算法都是对 [3] 和 [18] 提出的标准之字形算法的改编；当使用一个0-故障型智能体进行搜索时，它们分别适用于确定性模型和随机化模型，且均为最优。之字形算法由所谓的扩张因子 g 参数化，该因子刻画了搜索者每次改变方向时，搜索空间在每次迭代中的扩张速率。然而，在我们的情形中，搜索使用的是p-故障智能体，其中$p > 0$，因此算法无法完全控制搜索空间的扩张方式（因为转向会受到概率性故障的影响）。正因如此，在我们的情形下，对这些算法的分析技术性很强，这也是我们的贡献之一。

值得一提的是，最优确定性零故障智能体的最优扩张因子为 2，而我们为确定性p故障智能体（其中$p \in (0,1/2)$）采用的扩张因子$g = g(p)$会随着p增大而减小。当$p \to 0$时，

我们采用扩张因子 $1+\sqrt{2}$，且当$p \geq 0.146447$ 时，扩张因子降至 2。对于我们的随机化算法，所选扩张因子同样会随着p增大而减小；起始值与最优随机化零故障智能体的选取相同，为 [18],，并且当$p \geq 0.241516$ 时，扩张因子为 2。[3] 图 1b展示了这些扩张因子。

关于已确立竞争比的说明：根据引理 1 的证明可知，当 $p \to 1/2$ 时，p-PDLS 的最优竞争比的阶为 $\Omega(1/(1-2p))$。因此，为了更清晰地展示结果，我们在图 1a中绘出了按 $1/2-p$ 缩放后的已确立竞争比。此外，当$p \to 1/2$ 时，这些结果在常数因子范围内是最优的。

此外，值得注意的是，当 p 足够小时，已确立的竞争比优于 0-PDLS 著名的最优竞争比 9。这是因为我们的算法利用概率性故障来获取优势，从而削弱了对手的对抗性选择。事实上，代数计算表明，当 $p \leq 0.390388$ 时，确定性算法的竞争比小于 9；当 $p \leq 0.436185$ 时，随机化算法的竞争比小于 9。此外，在使用一个 p-故障代理进行搜索且 $p \to 0$ 时，我们的确定性算法的竞争比趋近于 $6.82843 < 9$，而随机化算法的竞争比趋近于可证明的最优竞争比 $\frac{1}{W\left(\frac{1}{e}\right)}+1 \approx 4.59112$，其中 $W(\cdot)$ 是朗伯 W 函数。还需要注意的是，仅对于确定性算法，在 $p=0$ 处不存在连续性，因为当智能体不出现任何故障时，对手能够确定所选择的轨迹，因此其能力严格更强。

当 p 接近 1/2 时，也会产生有趣的观测。事实上，从图 1a 中很容易看出，推导出的竞争比为 $\Theta\left(1/(1-2p)\right)$ 量级。更有趣的是，当 $\Theta\left(1/(1-2p)\right)$ 也成立时，确定性算法与随机化算法的竞争比之差同样为 $p \to 1/2$。因此，利用受控（算法性）随机性带来的改进十分显著。我们提出以下关键问题：

“对于只有一个智能体的 p-PDLS，访问随机预言机是否能带来优势？”

有些出人意料的是，我们给出的答案是 否定的！更具体地说，对于所有 $p \in (0,1/2)$，我们利用概率性故障所带来的优势，展示确定性 p-故障智能体（确定性算法）如何模拟随机化的 p-故障智能体（随机化算法）。因此，定理3中由随机化算法实现的改进上界，实际上可以由确定性 p-故障智能体模拟。该论断的证明基于这样一点：我们的随机化算法假设可以访问一个随机化预言机，该预言机仅从均匀分布中采样有限次（实际上只有 2 次）。其核心思想是，确定性智能体可以始终停留在任意接近原点的位置，并利用其故障回合采样任意多个随机比特，从而模拟对随机预言机的查询。下一定理的证明细节见第 B节。

定理 1。对于任意 $p \in (0,1/2)$，设随机故障智能体通过（有限次）访问从均匀分布中采样的预言机，为 p-PDLS 达到的竞争比为 c。则对于每个 $\epsilon > 0$，都存在一个针对同一问题的确定性 p-故障智能体，其竞争比至多为 $c+\epsilon$。

3.2 使用两个故障智能体进行搜索的结果概述

我们在第 6 节总结我们的贡献，其中我们研究 p-*PDLS* 以及两个智能体。首先，我们展示了两个（确定性）p-故障智能体（独立于底层通信模型）如何模拟任意一个（单个）0-故障型智能体的轨迹。作为直接推论，我们在第4 页的定理10 中推导出一种使用两个p-故障智能体寻找目标的方法，其竞争比为 $9+\epsilon$，对于每个 $\epsilon > 0$。最重要的是，只要 $p > 0$，该结果不仅在期望意义下成立，而且具有任意大的集中度。

受类似思想的启发，我们在第5 页的10 定理中展示了，两个确定性的p-故障智能体可以为 0-PDLS 模拟单个智能体所采用的著名最优随机化算法，使竞争比任意接近 4.59112。该结果同样与通信模型无关。然而，在这种情况下，我们无法保证像之前那样的集中度性质。

[3] 为简便起见，我们只给出关于p的数值界。本节中提到的所有p 的界都有明确的代数表示，稍后将对此进行讨论。

最后，我们研究使用两个无线 p故障智能体进行搜索的问题。在这里，我们能够在第6 页的定理11 中证明，两个智能体都能以竞争比 $3+4\sqrt{p}(1-p)$期望达到任意目标。性能随p $<1/2$ 增大，范围从 3（使用两个无线零故障智能体进行搜索时的最优竞争比）到 5。然而，与之前一样，我们可以控制性能的集中度，使其以任意置信度任意接近 $3+4\sqrt{p}(1-p)$。因此，~~对手较小~~的p 值，即所得竞争比小于 4.59112 时，这比第p5 定理更具优势。在这方面，我们证明当p >0.197063 时，竞争比超过 4.59112，因此上述各项结果本身都具有重要意义。

4 使用单个确定性p故障智能体进行搜索

我们首先介绍一种使用p故障智能体进行搜索的确定性算法，其中p $\in(0,1/2)$。本节的主要结果如下。

定理 2。使用单个智能体的 p-PDLS 存在一种确定性算法，其竞争比在 $0<p\leq\frac{1}{4}(2-\sqrt{2})$ 时等于$2(\sqrt{2}+2)$，并且在 $\frac{1}{4}(2-\sqrt{2})<p<1/2$ 时等于 $6-4p+\frac{1}{1-2p}$。

我们强调，在我们的算法中，拥有p >0 将是至关重要的。这是因为概率性故障给对手带来了不确定性。因此，事实上对于所有$p<\frac{1}{8}\left(\sqrt{17}-1\right)\approx 0.390388$，我们都能得到竞争比 $2(\sqrt{2}+2)\approx 6.82843<9$，这一点很有趣，但并不令人意外。

首先，我们对算法进行详细描述。算法以p $\in(0,1)$作为输入，并选择参数g $=g(p)$，作为搜索空间预期的扩张率。在算法的每次迭代中，智能体都会从原点出发，试图沿某个方向扩张至g_i。走过距离g^i后，智能体尝试返回原点。再经过g^i时间后，智能体便知道是否已到达原点；若已到达，她就朝相反方向扩张，预期扩张量为g^{i+1}。若未到达，智能体便知道自己未能转向，于是继续前进至点g^{i+1}（这就是我们要求g ≥ 2的原因）。随后，她再次尝试转向。这个过程持续进行，直到智能体成功在g_j处转向，其中$j>i$；之后，智能体尝试朝相反方向扩张，预期扩张长度为g^{j+1}。算法首先朝任意一个方向（例如向右）搜索，预期扩张量为g^0。稍后将会清楚，只有当g $<1/p$时，算法的终止时间才会收敛。

为了简化论述（并避免重复），我们先介绍子程序算法1，它是我们提出的两种单个智能体搜索算法的基线。接下来介绍算法 2，这是我们的第一个搜索算法。

算法 1 基线搜索

要求： g，p，i，d

重复
  重复
    以速度 1 朝点 dg_i移动
  直到 找到目标 $\vee$ g^i 时间已过
  当 原点为 ¬找到 $\wedge$ 目标为 ¬找到 **执行**
    设定 $d\leftarrow -d$ {这会改变方向，但失败的概率为 p}
    设置 $i\leftarrow i+1$
    重复
      以速度 1 朝点 dg_i移动
    直到 找到目标或 g^i-g_{i-1}时间已过，或找到原点
  结束循环
直到 找到目标

算法 2 有故障的确定性线性搜索

要求： $2 \le g \le 1/p$ 且 $0 < p < 1/21$：设置 $i \leftarrow 0$ 且 $d \leftarrow 1$ {$d \in \{1,-1\}$ 表示方向（1 表示向右，-1 表示向左）2：使用参数 g 、p 、i、d运行算法 *1*。

稍后我们将解释，定理 2 可直接由以下技术引理推出，其证明见第C节。

引理 2。 固定 $p \in (0,1/2)$ 且 $g \in [2,1/p)$。若目标位于 $(g^t, g^{t+1}]$，则算法*2* 的竞争比至多为

$$\frac{(1-p)g}{(1-gp)}\left(\frac{1+(1-2p)g}{g-1}+(1-2p)^{t+1}\right)+1.$$

取.....的极限　当 $t \to \infty$ 时，取引理 2 中表达式的极限，可知算法 2 is

$$f_p^{\mathrm{DET}}(g) := \frac{1-g(2g(p-2)p+g+2)}{(1-g)(1-gp)}$$

对所有符合前提的p、g均成立。[4]

接下来，我们优化函数$f_{\mathrm{DET}p}(g)$。当$p \le \frac{1}{4}(2-\sqrt{2})$时，最优扩张因子（即$f_{\mathrm{DET}p}(g)$的最优解）为

$$g_0(p) := \frac{-(\sqrt{2}+2)p+\sqrt{2}+1}{1-2p^2}.$$

容易看出，$2 \le g_0(p) < 1/p$，对所有$p \le \frac{1}{4}(2-\sqrt{2})$均成立（事实上，对所有 $0 < p < 1$，严格不等式都成立）。在这种情况下，得到的竞争比为 $2(\sqrt{2}+2) \approx 6.82843$。

当$p \ge \frac{1}{4}(2-\sqrt{2})$时，最优扩张因子（至少为 2）为$g = 2$，此时竞争比变为 $6-4p+\frac{1}{1-2p}$。有趣的是，当$p \ge \frac{1}{8}(\sqrt{17}-1) \approx 0.390388$ 时，竞争比至少为 9。换言之，0.390388 是与智能体故障性相关的概率阈值；至少对于所提出的算法而言，只有在故障概率低于该阈值时，这种概率性故障才有助于突破可证明的最优确定性界 9。相关地，回顾引理1，存在一个概率阈值p'，使得使用p-故障代理进行搜索时，任何算法（甚至是随机化算法）的竞争比都至少为 9，其中$p \ge p'$。

5 使用单个随机化（改进的确定性）p-故障智能体进行搜索

本节中，我们赋予p-故障智能体随机性的能力，并给出下一个正面结果。注意，由于定理 1，结果也可以由确定性p-故障智能体模拟，且精度任意。

定理 3. 令 $p_0 = \frac{1}{8}\left(5-\sqrt{1+\ln^2(2)+6\ln(2)-\ln(2)}\right) \approx 0.241516$。对于每个 $p < p_0$，，令 g^p 表示不小于 *2* 的唯一根，其满足

$$h_p(g) := (1-gp)(1+g(1-2p))-g(1-p)\ln g. \tag{1}$$

于是，对每个 $p \in (0,1/2)$，具有单个智能体的 p-PDLS 都存在一种随机化算法，其竞争比至多为

$$\begin{cases} g_p\left(\frac{1-p}{1-g_p p}\right)^2+1, & p \in (0,p_0] \\ (1-p)\frac{1+2(1-2p)}{(1-2p)\ln(2)}+1, & p \in (p_0,1/2). \end{cases}$$

[4]如果希望使用竞争比的原始定义，那么只需适当重新缩放搜索空间（即缩放预定转向点），对于任意 $\epsilon > 0$，都可以使竞争比与所得值的加性偏差至多为 ϵ。

元素 简单计算表明，定理 3 的竞争比仍至多为 9，只要 s

$$p \le \frac{1}{8}\left(7+\sqrt{1+256\ln^2(2)-96\ln(2)}-16\ln(2)\right)\approx 0.436185.$$

首先，我们证明定理 3 中关于 g_p 的前提是定义良好的。事实上，考虑式 (1) 中的 $h_p(g)$。我们有 $\partial h_p(g)/\partial p = 4g^2p - g^2 - 3g + g\ln g \le -3g + g\ln g < 0$，对所有 $p \le 1/4$ 均成立。因此，$h_p(2) = 8p^2 - 10p + 2p\ln(2) + 3 - 2$ 随 p 单调递减，因此根据 p_0 的定义，$h_p(2) > h_{p_0}(2) = 0$。此外，$h_p(4) = 32p^2 - 28p + 4p\ln(4) + 5 - 4\ln 4$ 也随 p 单调递减，因此 $h_p(4) \le h_0(4) = 5 - 4\ln 4 \approx -0.545177 < 0$。这意味着，对于所有 $p \in (0, p_0]$，$h_p(g)$ 关于 g 确实存在一个根，且该根位于 $[2,4)$。接下来我们证明这个根是唯一的。事实上，我们有 $\partial h_p(g)/\partial g = 4gp^2 - 2gp + p\ln g - \ln g - 2p \le -\ln g < 0$，当 $p \in [0, 1/2]$ 时成立。因此，对所有 $p \le p_0 < 1/2$，函数 $h_p(g)$ 关于 g 单调递减，所以根是唯一的；也就是说，对所有 $p \in (0, 1/2)$，g_p 确实定义良好，且位于区间 $[2, 1/p)$。

接下来，我们注意到，当 $p \to 0$ 时，定理 3 所保证的、使用 p 故障型智能体（即近乎无故障的代理）进行搜索时的竞争比为 g_0，其中 $g - g\ln g + 1$ 的根，即 $g_0 = \frac{1}{W(\frac{1}{e})} \approx 3.59112$，其中 $W(\cdot)$ 是朗伯 W 函数。[5] 此外，所得到的竞争比为 $1 + g_0 \approx 4.59112$，这恰好是 Kao 等人为原始线性搜索问题提出的最优随机化算法的竞争比 [18]。我们还将此视为对计算结果正确性进行合理性检查的一种方式。

不出所料，我们用于证明定理 3 的算法，是对使用 0-故障型智能体进行线性搜索的著名最优随机化算法的改编 [18]。与之前一样，搜索算法（见下方的算法 3 ）由某个扩张因子 g 决定，该因子表示每次运动方向改变时，搜索空间被扩展的（在我们这里因故障而预定的）倍数。随机化算法向随机预言机发出两次查询。首先，它选择一个随机比特，表示要遵循的初始随机方向。其次，算法对随机变量 ϵ 进行采样，该变量在 $[0,1]$ 中均匀随机地取值。变量 ϵ 量化了预定转向点所对应的随机尺度。值得注意的是，在算法 中将其 3 设为确定性值，并移除初始随机方向选择后，便得到之前的确定性算法 2。

算法 3 有缺陷的随机化线性搜索

要求： $g \ge 2$ 且 $0 < p < 1/2$

1: 均匀随机地选择 ϵ 从 $[0,1)$ 2: 设定 $i \leftarrow \epsilon$ 且 $d \leftarrow 1$ $\{d \in \{1,-1\}$ 表示方向（1 表示向右，-1 表示向左）3: 使用参数 1g 、p、i、d运行算法。

接下来我们证明 定理 3 如何由以下技术引理推出，其证明见第 D 节 .

引理 3。 对于任意 $p \in (0,1/2)$ 以及任意 $g \in [2, 1/p)$，若目标位于区间 $(g_t, g^{t+1}]$，则算法 3 的竞争比至多为

$$1 + \frac{1}{(1-gp)\ln g}\left(\begin{array}{c}(1-p)(1-g(-1+2p))(1+(-1+2p)^t)\\ +\frac{2(1-p)}{g^t} + 2g(1-p)(p+g^t(-1+p)(-1+2p)^t\end{array}\right).$$

回想一下，$p \in (0,1/2)$，因此令引理3 中的表达式在 $t \to \infty$ 时取极限，并对表达式进行代数化简，可得算法3 的竞争比至多为

$$f_p^{\mathrm{RAND}}(g) := 1 + \frac{(1-p)(1+g(1-2p))}{(1-gp)\ln g},$$

对于所有满足前提的 p，g。接下来，我们对所有参数 p 下的 $\mathrm{RAND}_p(g)$ 进行优化，其中 $p \in (0,1/2)$，并满足约束 $2 \le g < 1/p$。具体而言，我们证明，$f_{\mathrm{RAND}_p}(g)$ 的最优解为 g_p，若 $p \le p_0$，且 $g = 2$；否则，竞争比如定理3所述。

[5] 朗伯 W 函数是 $L(x) = xe^x$ 的反函数。

首先，我们计算 $\frac{\partial}{\partial g} f_{\mathrm{RAND}p}(g) = \frac{1-}{pg(1-gp)^2\ln^2 g} h^{p}(g)$，其中 $h^p(g)$ 与 (1) 相同。如定理 3陈述下方已证明的那样，我们有 $h^p(g)$ 在区间 $[2,4$ 中有唯一的根。接下来，我们证明 $f_{\mathrm{RAND}p}(g)$ 是凸函数。

事实上，我们有

$$\frac{\partial^2}{\partial g^2} f_p^{\mathrm{RAND}}(g) = \frac{1-p}{g^2(1-gp)^3\ln^3(g)} s_p(g),$$

其中 $s^p(g) = \sum^3{}_{i=0} \alpha_i p^i$ 是关于 p 的三次多项式，其系数为

$$\begin{aligned}
\alpha_3 &= 2g + g(-\ln g) + \ln(g) + 2 \\
\alpha_2 &= 2g\left(-2g + g\ln^2 g - \ln g - 4\right) \\
\alpha_1 &= g^2\left(2g - 2\ln^2 g + g\ln g + 3\ln g + 10\right) \\
\alpha_0 &= -2g^3(\ln g + 2)
\end{aligned}$$

因此，很容易验证，对于所有 $p \in (0,1/2)$，在满足 $g \in [2,4]$ 的条件下，$s^p(g)$ 始终为正（事实上，定理 3 中所描述的最优解确实满足这一性质）。我们得出，$f_{\mathrm{RAND}p}(g)$ 关于 g 是凸函数。

结合我们之前的观察，这意味着，在约束 $g \geq 2$ 下，当 $p \leq p_0$，时，函数 $f_{\mathrm{RAND}p}(g)$ 在 $\frac{\partial}{\partial g} f_{\mathrm{RAND}p}(g)$ 的唯一根处取得最小值；而当 $p \in [p_0, 1/2)$ 时，在 $g = 2$ 处取得最小值。这些正是定理 3 中所描述的最优解；特别地，竞争比 $f_{\mathrm{RAND}p}(g)$ 经过了简化，因为对于所选取的 g 值，我们有 $s^p(g) = 0$，对于所有 $p \leq p_0$ 均成立。

最后，还需论证 f RANDp (g) 的所有最优解确实至多为 1/p。为此，只需证明对于所有 p $\leq p_0$，h^p(g) 的唯一根 g^{p}至多为 1/p（因为对于更大的 p 值，我们使用扩张因子 g = 2）。由于 g ≥ 2，我们有 h^p(g) $\leq$ 2g ^{2}p^2 − g ^{2}p − 3g p + g p ln 2 + g − g ln 2 + 1。最后一个表达式是关于 p 的二次多项式，它只有一个正根，即

$$\frac{-\sqrt{(-3p + p\ln 2 + 1 - \ln 2)^2 - 4\left(2p^2 - p\right)} + 3p + p(-\ln 2) - 1 + \ln 2}{2p\left(2p - 1\right)}.$$

经过简单计算即可得出，对于所有 $p \in (0,1/2)$，后一个表达式至多为 $1/p$。事实上，可以证明，上述表达式乘以 p 后关于 p 严格递增，并且在 $p = 1/4 > p_0$ 时变为

$$\frac{1}{4}\left(1 - \ln 8 + \sqrt{9 + (\ln 8 - 2)\ln 8}\right) \approx 0.486991 < 1/2.$$

这表明，对于每个 $p < p_0$，对于 $h^p(g)$ 的唯一根 g^p，不等式 $g_p < 1/2p < 1/p$成立，正如所需。

6 使用两个 p 故障智能体进行搜索

本节针对两个故障智能体，介绍所有 $p \in (0,1/2)$ 情况下的 p-PDLS 算法。我们最初研究的核心是以下子程序：从较高层面来看，该子程序将由一个 p 故障智能体，即领导者，在跟随者的帮助下执行，以实现一次“强制”转向；其中跟随者可以是一个指定的不动点，例如目标或原点，也可以是另一个p故障智能体。在此过程中，承担跟随者角色的p故障智能体可能需要减速，甚至暂时停止一段时间，同时仍须遵守概率性故障转向规则（停止后，她可以继续沿之前的方向移动，但改变方向则会受到故障影响）。

从下文引理 4 的证明中可以明显看出，算法4 允许智能体在任意接近预定转向点的位置改变方向，并且可以具有任意集中度（两者均由参数 γ控制）。

下一个引理涉及一项任务：无论底层通信模型如何，两个p-故障智能体都能独立完成这项任务。概括来说，该引理证明，两个p-故障智能体可以绕过概率性故障，但代价是放弃搜索者移动的独立性。

算法 4 强制改变方向（领导者的指令）

要求： γ较小的实数，跟随者可以移动或静止。

1: **重复**
2: 改变方向{此操作失败的概率为p}
3: 移动 γ 时间（如果跟随者可以移动），如果跟随者静止则移动 $\gamma \leftarrow 2\gamma$ 时间。
4: **直到** 你与跟随者相遇
5: 向跟随者传达领导者的方向

引理4。 对于每个 $p \in [0,1/2)$，两个 p-故障智能体都能以任意精度（以及任意概率集中度）模拟一个确定性0-故障智能体的轨迹。

证明。 考虑两个最初同时位于原点的故障概率为p的智能体。我们将说明这些智能体如何模拟（精度任意）的确定性轨迹。为此，我们需要说明这两个智能体如何在任意点成功转向，同时在期望上不偏离该点。

事实上，考虑一个预定的转向点，以及正在接近该点的两个p-故障智能体。对于某个 $\gamma > 0$ 足够小的值，在时间 2γ 之前，两个智能体会在到达该点前承担两个不同的角色，分别是领导者和跟随者。在算法执行过程中，这些角色可以始终保持不变。跟随者瞬间减速，使得领导者与跟随者之间的距离变为 2γ时，跟随者位于转向点之前 $\gamma/(1-p$ 处（严格大于 γ 且严格小于2γ），因此跟随者越过转向点$\frac{1-2p}{1-p}\gamma$。此时，跟随者恢复全速，两个智能体继续沿之前相同的方向移动。随后，领导者运行算法4，并将移动的跟随者视为另一个p-故障智能体。

注意，如果两个智能体在任何时刻相遇，那是因为在一次成功的转向之后，二者朝彼此方向移动了时间 γ；而如果转向未成功，二者会保持其相对距离 γ。因此，自第一次转向尝试的时刻起，两个智能体在期望时间 $\sum_{i=0}^{\infty}(i+1)\gamma(1-p)p^i = \frac{\gamma}{1-p}$后相遇。我们得出结论：期望的相遇点（以及转向点）就是原始转向点。最重要的是，所得转向点偏离给定转向点的概率会随p呈指数下降，并且还与 γ 成正比，而 γ 可以独立地选择为任意小。□

引理 4 非常强大，因为它说明了如何以任意小的偏差来模拟确定性转向，使其偏差于实际转向点任意小。更重要的是，可以动态地选择使该偏差以任意快的速度下降，从而我们能够在算法执行的后期实现更小的期望偏差，以此补偿已经经过的时间。因此，我们得到以下定理。

定理 4. 对于所有 $p \in [0,1/2)$，两个确定性故障智能体都能解决 p-PDLS 问题，且竞争比为 $9+\epsilon$，对任意 $\epsilon > 0$ 均成立，与底层通信模型无关。此外，性能可以任意集中于 $9+\epsilon$ 附近。

值得注意的是，在定理 4 的底层算法中，由于存在概率性故障转向，智能体的移动仍然是随机的。不过，每次调用算法 4 时选择适当的参数，就能使算法的期望性能任意集中，因此，$9+\epsilon$的界在实践中可以视为确定性的。

相比之下，采用同样的技巧，我们可以获得好得多的竞争比，但只能在期望意义上达到 [18] 的竞争比（集中度不受控制）。要了解具体做法，请注意，根据定理 1的证明，两个 p- 故障智能体可以待在一起，以收集足够多的随机比特，并模拟对随机预言机的任意有限次查询。然后，利用引理 4，这些智能体可以模拟 [18]的最优随机化算法，其性能为 4.59112。该算法最初是为一个随机化的 0-故障型智能体设计的，该智能体只对均匀分布进行 2 次查询。换言之，两个确定性 p-故障智能体可以利用引理 4，并通过调用定理1，克服其故障转向以及缺少随机预言机的问题。总之，我们得到以下定理，其要求 $p > 0$。

定理 5。 对于任意 $\epsilon > 0$以及任意 $p \in (0,1/2)$，两个确定性故障智能体都能以竞争比 $459112+\epsilon$解决 p-PDLS，且不受底层通信模型的影响。

在最后一个主要结果中，我们证明，在无线模型中运行的两个p-故障智能体，对于所有$p < 1/2$，都能达到优于 9 的表现；对于范围很广的p取值，也能达到优于 4.59112 的表现。请注意，所达到的竞争比至少为 3；这是无线模型中两个零故障智能体进行搜索时的最优竞争比。此外，当$p \to 0$时，我们的结果与这一已知界相吻合。

定理 6. 无线模型中的两个确定性 p-故障智能体可以用竞争比为 $3+4\sqrt{p(1-p)}+\epsilon$ 的方法解决 p-PDLS 问题，对每个 $\epsilon > 0$ 以及每个 $p \in [0,1/2)$ 均成立。

算法 5 两个无线p-故障智能体的搜索**要求：** 具有不同角色（领导者和跟随者）的p-故障智能体，$s < 1$和 $\gamma > 0$。

1: 智能体朝相反方向搜索，直到找到目标并报告。2: 找到目标的智能体成为领导者，未找到目标的智能体成为跟随者。3: 未找到目标的智能体将速度改为s，尝试转向（该操作会以一定概率失败），并继续移动，直到与发现者相遇。4: 发现者沿同一方向移动 $\gamma > 0$，并运行算法4 （目标扮演跟随者的角色），直到再次到达目标。5: 发现者（领导者）继续移动，直到与未发现者相遇。6: 未发现者（跟随者）原地等待，直到领导者再次找到它。7: 领导者继续沿同一方向移动（远离目标和跟随者） γ 时间，然后以跟随者作为静止智能体运行算法4 以完成转向。8: 领导者成功转向后，接上跟随者，并继续沿同一方向一同移动至目标。

定理6的证明（（证明思路）；详情见第E节）。证明通过分析算法5 在适当选择速度$s < 1$时的性能给出。在这个简化的证明（思路）中，我们假设目标发现者（利用目标）以及结伴行走的两个智能体都能确定性地转向。事实上，在完整证明（第4节）中，我们利用算法E说明，实际中带有概率故障的转向对竞争比的影响很小。

我们假设发现者在时间 1 时报告目标，此时两个智能体之间的距离为 2。对于未发现者而言，她在收到无线消息时成功转向的概率为 $1-p$。由于发现者朝她移动，他们的相对速度为 $1+s$。这意味着他们会在额外经过 $2/(1+s)$ 的时间后相遇，在此期间，未发现者向目标靠近了 $2s/(1+s)$。因此，两个智能体相遇时，他们与目标的距离为 $2-2s/(1+s)$。

另一方面，以概率 p 未发现者未能转向，两个智能体继续朝同一方向移动，只是未发现者的新速度为 s。因此，在这种情况下，它们的相对速度为 $1-s$。这意味着它们将在额外经过 $2/(1-s)$ 的时间后相遇，在此期间，未发现者又沿远离目标的方向移动了 $2s/(1-s)$ 的距离。因此，当两个智能体相遇时，它们距目标的距离为 $2+sW+s(2+sW)/(1-s)$。还需注意的是，当它们相遇时，可以共同进行一次强制转向（这会使终止时间达到极小值），从而得到总终止时间（由于目标距离为 1，竞争比也为该值）

$$3+p\left(\frac{2}{1-s}+\frac{2s}{1-s}\right)+(1-p)\left(\frac{2}{1+s}-\frac{2s}{1+s}\right)=\frac{5-s(s+4-8p)}{1-s^2}.$$

我们选择 $s = s(p) = \frac{1-2\sqrt{p-p^2}}{1-2p}$，即后一表达式的极小值点。容易看出，对于所有 $p \in (0,1/2)$，该值都属于 $(0,1)$，因此这是对速度的有效选择。现在将其代回竞争比公式，并进行代数化简，可得表达式 $3+4\sqrt{(1-p)p}$。□

值得注意的是，定理 5 中的 4.59112 上界是在期望意义下成立的，无法控制偏差。然而，定理 6 中的上界同样是在期望意义下成立的，但由此得到的性能可以以任意精度集中在期望值附近。此外，所导出的竞争比随着 $p < 1/2$ 严格递增，范围从 3 到 5。因此，定理 6 的缺点在于，对于较高的

足够大的 p ($p > 0.197063$) 值，所诱导的竞争比会超过 4.59112。因此，我们有理由在 6所述算法与 $p \le 0.197063$ 时，以及其他情况下的定理 5 所述算法之间进行选择。研究一种结合这两种思想的混合算法能否取得更优结果，将会很有意义

7 结论

本文研究了一种新的移动智能体搜索问题，其中智能体在搜索空间中的导航能力存在概率性故障，即智能体每次尝试改变方向都是一次独立的伯努利试验（智能体以概率 $p < 1/2$ 转向失败）。在使用单个智能体进行搜索时，我们性能最佳的算法在 $p \to 0$ 时具有最优性能 4.59112，在 $p \le 0.436185$ 时性能低于 9，并且当 $p \to 1/2$ 时，其性能在常数因子意义下最优且无界。在使用两个故障智能体进行搜索时，我们提供了 3 种具有不同特性的算法。其中一种算法具有（期望）性能 9 和任意集中度，另一种算法的性能为 4.59112，最后一种算法的性能为 $3 + 4\sqrt{p(1-p)}$（介于 3 和 5 之间），同样具有任意集中度。

即使在这种概率性设置下，对于单个搜索者而言，我们仍能设计出性能优于著名之字形算法的算法，这相当令人惊讶；后者用于线性搜索时的竞争比为 9。同样令人惊讶的是，对于所有 $p \in (0,1/2)$，双搜索者问题都具有有界竞争比，这一点不同于单个搜索者问题。未来研究中无疑可以进一步探讨类似或相关的“概率性导航”故障，既可以单独研究这些故障，也可以将其与更一般搜索领域中的“通信”故障结合起来（例如在平面中，或在具有 w 条射线的更一般的奶牛小径中），并研究多个（可能相互协作的）智能体。

参考文献

1. R. Ahlswede 和 I. Wegener。《搜索问题》。John Wiley & Sons, Inc., 1987。2. S. Alpern 和 S. Gal。《搜索博弈与会合理论》，第 55 卷。Springer Science & Business Media, 2006。3. R. Baeza-Yates、J. Culberson 和 G. Rawlins。平面搜索。《Information and Computation》，106(2):234–252, 1993。4. R. Baeza-Yates 和 R. Schott。平面并行搜索。《Computational Geometry》，5(3):143–154, 1995。5. V. Baston 和 S. Gal。当起点留下标记时的会合搜索。《Naval Research Logistics (NRL)》，48(8):722–731, 2001。6. A. Beck。关于线性搜索问题。《Israel J. of Mathematics》，2(4):221–228, 1964。7. A. Beck。关于线性搜索问题的进一步研究。《Israel J. of Mathematics》，3(2):61–70, 1965。8. A. Beck 和 D. Newman。关于线性搜索问题的更多研究。《Israel J. of Mathematics》，8(4):419–429, 1970。9. R. Bellman。一种最优搜索。《SIAM Review》，5(3):274–274, 1963。10. A. Bonato、K. Georgiou、C. MacRury 和 P. Prałat。半直线上的 p 故障搜索算法。《Algorithmica》，2022。11. M. Chrobak、L. G ąasieniec、G. T. 和 R. Martin。直线上的群体搜索。载于《SOFSEM 2015》，第 164–176 页，捷克共和国 Sn˘ežkou，2015。Springer。12. J. Czyzowicz、K. Georgiou 和 E. Kranakis。群体搜索与疏散。载于《移动实体分布式计算》，第 335–370 页。Springer，2019。13. J. Czyzowicz、K. Georgiou、E. Kranakis、D. Krizanc、L. Narayanan、J. Opatrny 和 S. Shende。拜占庭机器人在线上的搜索。《International Journal of Foundations of Computer Science》，第 1–19 页，2021。14. J. Czyzowicz、R. Killick、E. Kranakis 和 G. Stachowiak。近多数故障智能体条件下的搜索与疏散。载于《SIAM 应用与计算离散算法会议 (ACDA21)》，第 217–227 页。SIAM，2021。15. J. Czyzowicz、E. Kranakis、D. Krizanc、L. Narayanan 和 J. Opatrny。存在故障机器人的直线搜索。《Distributed Com- puting》，32(6):493–504, 2019。16. E. Demaine、S. Fekete 和 S. Gal。带转向成本的在线搜索。《Theoretical Computer Science》，361(2):342–355, 2006。17. K. Georgiou 和 J. Lucier。直线上的加权群体搜索及其对优先疏散问题的影响。《Theoretical Computer Science》，939:1–17, 2023。18. M.-Y. Kao、J. H. Reif 和 S. R. Tate。在未知环境中搜索：奶牛小径问题的最优随机化算法。《Information and Computation》，131(1):63–79, 1996。19. B. J. McCabe。搜索一维随机游走者。《Journal of Applied Probability》，11(1):86–93, 1974。20. D. L. Stone。《最优搜索理论》。Academic Press, Incorporated, 1975。21. X. Sun、Y. Sun 和 J. Zhang。拜占庭机器人在线上搜索的更优上界。载于《Complexity and Approx- imation》，第 151–171 页。Springer，2020。

引理1 的证明

证明（引理 *1* 的概要）。考虑一条轨迹，以及智能体在某一时刻远离原点移动并转向，此时它与原点的距离为 $d > 0$。我们计算智能体返回原点所需时间的一个下界；这也构成了向相反方向扩展搜索空间所需时间的下界。

注意，以概率 $1-p$ 转向成功，智能体可以在至少d的时间内返回原点。以概率p 转向也可能失败，而智能体最早要到时间d 才能察觉自己偏离了预定轨迹（因为它至少需要d 的时间才能到达原点）。在后一种情况下，智能体必须到达距原点 $2d$的位置。类似地，尝试下一次转向时，以概率 $1-p$，智能体至少需要 $2d$ 的时间才能到达原点；在它到达距原点 $4d$ 的位置之前，都无法得知转向可能失败（概率为p），依此类推。因此，智能体到达原点的期望时间至少为 $\sum_{i\geq 0}(1-p)p^i 2^i d$。当$p \geq 1/2$ 时，该级数发散，由此得出我们的结论。□

B 定理1的证明

为了证明定理1，我们需要以下引理，它说明了如何提高故障概率。

引理 5。对于任意 $p \in (0,1)$，以及任意 $\epsilon > 0$，存在 $\delta \in (0,\epsilon)$，使得确定性 p-故障智能体能够模拟确定性 q 故障智能体，其中 $|q - 1/2| \leq \delta$。

证明。考虑一个p-故障型智能体，其中$p \in (0,1)$。假设它当前的方向朝右，并试图改变方向。方向保持不变的概率为p。将此称为q_1,，其中索引表示尝试转向的次数。

现在考虑$k \geq 2$ 次连续的伯努利试验，在此过程中，智能体连续k次尝试改变方向，且各次尝试之间不移动。我们计算方向仍然朝右的概率q_k。这一事件发生的情况有两种：在第$k-1$ 次尝试后，方向朝左（发生概率为$1-q_{k-1}$），而最后一次试验中方向按尝试的结果发生改变（概率为 $1-p$）；或者在第$k-1$次尝试后，方向朝右（发生概率为q_{k-1}），而最后一次试验中方向未按尝试的结果发生改变（概率为p）。因此，我们得到递推关系

$$q_k = (1-p)(1-q_{k-1}) + p\, q_{k-1}$$

初始条件为 $q_1 = p$。该递推关系的解为

$$q_k = \frac{1}{2}\left((2p-1)^k + 1\right).$$

由于 $p \in (0,1)$，因此对于任意 $\epsilon > 0$，都存在一个足够大的 $k \in \mathbb{N}$，使得 $1/2 - \epsilon < q_k < 1/2 + \epsilon$。 □

我们现在可以证明定理 1。

证明（定理*1*）。根据引理 5，不妨设我们有一个确定性q-故障智能体，其中 q可以任意接近 1/2。因此，只需说明确定性q-故障智能体如何模拟随机化的p-故障智能体（算法）的随机选择。

回想一下，p-故障智能体（算法）的随机性源于其能够访问一个从均匀分布中采样的预言机。因此，只需说明如何利用（其他方面均为）确定性q-故障智能体的随机故障，从均匀分布中采样。为此，只需说明如何生成任意长度的（近似）均匀分布二进制字符串。

确实，固定任意整数 k（可以任意大；目标是生成长度为 k 的字符串）。对于任意 $\gamma > 0$，我们将说明确定性 q 故障智能体如何生成长度为 k的近乎均匀的二进制字符串，同时始终与原点保持不超过 γ 的距离。[6] 为此，智能体沿任意方向移动时间 $\zeta > 0$，其中某个 $\zeta < \gamma/2^k$ 很小

[6] 由于我们将出口任意放置在远离原点的位置所得到的竞争比定义为，γ 不需要很小，但为了便于直观理解，我们仍将其设为较小值。

足够长。然后，它尝试转向，并继续移动一段额外的时间 ζ，以便要么返回原点（概率为$1-q$），要么不返回（概率为 q）。这样，智能体就收集到了一个随机比特，该比特以概率 q 为 1。如果智能体位于原点，它会重复这一过程以收集更多随机比特。如果不在原点，那么可以证明它已 2ζ 远离原点。随后，它再次尝试转向，并继续移动总计额外的时间 2ζ，在这种情况下，它要么到达原点（概率为 $1-q$），要么不到达（概率为 q），从而收集到另一个随机比特。注意，这里概率性故障必须是独立伯努利试验，这一点非常重要。智能体以同样的方式继续，直到收集到 k 个随机比特；此时，它至多已移动了 $2^k\zeta < \gamma$，远离原点。

总而言之，我们上文展示了确定性 p-故障智能体如何以临时方式充当确定性 q-故障智能体（q 可以任意接近 1/2），从而收集任意长、近乎服从均匀分布的二进制字符串；最重要的是，它始终能与原点保持任意近的距离。利用这个随机二进制字符串，智能体可以以任意精度模拟对均匀分布的任意有限次查询。因此，接下来只需让确定性 p-故障智能体返回原点，以执行随机化 p-故障智能体的轨迹（即模拟算法的随机选择）。根据上一步，确定性智能体知道自己距原点有多远（如果它还没有到达原点）。将这个距离记为 $\mathrm{w} \le \gamma$。回想一下，γ 可以选得任意小，其取值也可能取决于 p。智能体可以反复尝试返回原点，每次失败后便将其到原点的距离加倍。事实上，经过 l 次尝试后，智能体仍未返回原点的概率为 p^l；此时，它距原点至多为 $2^l\mathrm{w} < 2^l\gamma$。在这种情况下，p-故障智能体可以运行定理 2 的确定性算法，其竞争比记为 $\mathrm{c}^{\mathrm{DET}}$；由于智能体的初始点与原点任意接近，所得到的竞争比也任意接近 $\mathrm{c}^{\mathrm{DET}}$。反之，智能体已返回原点，概率为 $1-\mathrm{p}^l$。因此，竞争比为 $(1-\mathrm{q}^l)\mathrm{c} + \mathrm{q}^l\mathrm{c}^{\mathrm{DET}}$，这正如所需，任意接近 c。□

引理2的证明

本节将证明引理2。我们强调，下面（以及所有章节中）对移动方向的表述都是相对的。由于智能体能够识别原点，我们将移动方向表示为向左或向右，并且不失一般性地假设右侧为正点、左侧为负点。然而，智能体实际上并不知道这一事实，因为它只是计算已经经过的时间，以确定其下一步移动。

对于目标的任意位置n，存在$t \in \mathbb{N}$使得$n \in (g^t, g^{t+1}]$。对于任意初始预定扩展，我们令T为一个随机变量，其值等于目标位置n对应算法的终止时间。当智能体经过原点时，我们用X表示等于预定扩展的随机变量，因此，给定算法的期望终止时间为$E[T \mid X = g^0]$。我们设

$$R_k := E[T \mid X = g^k]$$
$$L_k := E[T \mid X = -g^k],$$

即R_k、L_k分别表示在智能体最初向右扩展至g^k以及向左扩展至$-g^k$条件下，算法的期望终止时间。因此，我们希望计算$R_0 = E[T \mid X = g^0]$。现在假设智能体位于原点，并打算扩展至g^k，其中$k \le t$。我们有如下

$$\begin{aligned} R_k &= (1-p)\left(2g^k + L_{k+1}\right) + (1-p)p\left(2g^{k+1} + L_{k+2}\right) + \dots \\ &\quad + (1-p)p^{t-k}\left(2g^t + L_{t+1}\right) + (1-p)\sum_{i=t-k+1}^{\infty} p^i n \\ &= (1-p)\sum_{i=0}^{t-k-1} p^i L_{k+i+1} + (1-p)p^{t-k}L_{t+1} \\ &\quad + (1-p)2g^k\sum_{i=0}^{t-k}(pg)^i + (1-p)\sum_{i=t-k+1}^{\infty} p^i n. \end{aligned}$$

另请注意 对于所有 $k \ge t+1$，我们都有 $R_k = n$。由此可知，对于所有 $k \le t +$ 1,

$$
\begin{aligned}
L_k &= (1-p)\left(2g^k + R_{k+1}\right) + (1-p)p\left(2g^{k+1} + R_{k+2}\right) + (1-p)p^2\left(2g^{k+2} + R_{k+3}\right) + \dots \\
&= (1-p)2g^k \sum_{i=0}^{\infty} (pg)^i + (1-p)\sum_{i=0}^{\infty} p^i R_{k+1+i} \\
&= (1-p)2g^k \sum_{i=0}^{\infty} (pg)^i + (1-p)\sum_{i=0}^{t-k-1} p^i R_{k+1+i} + (1-p)\sum_{i=t-k}^{\infty} p^i n
\end{aligned}
$$

这尤其意味着

$$
L_{t+1} = (1-p)2g^{t+1}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=0}^{\infty} p^i n = \frac{2(1-p)g^{t+1}}{1-gp} + n.
$$

接下来，对于$k = 0$、...、t，我们定义

$$
\begin{aligned}
\alpha_k :=& (1-p)p^{t-k}L_{t+1} + (1-p)2g^k\sum_{i=0}^{t-k}(pg)^i + (1-p)\sum_{i=t-k+1}^{\infty} p^i n \\
=& (1-p)p^{t-k}\left(\frac{2(1-p)g^{t+1}}{1-gp} + n\right) + (1-p)2g^k\left(\frac{1-(gp)^{t-k+1}}{1-gp}\right) + np^{t-k+1} \\
=& p^{t-k}\frac{2(1-p)^2 g^{t+1}}{1-gp} + (1-p)2g^k\left(\frac{1-(gp)^{t-k+1}}{1-gp}\right) + np^{t-k} \\
=& \frac{2(1-p)g^k}{1-gp}\left(1 + (1-2p)g(pg)^{t-k}\right) + np^{t-k} \qquad (2) \\
\beta_k :=& (1-p)2g^k\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=t-k}^{\infty} p^i n \\
=& \frac{2(1-p)g^k}{1-gp} + np^{t-k}, \qquad (3)
\end{aligned}
$$

同时，我们也定义 $\alpha^T = (\alpha_0,\dots,\alpha_t)$、$\beta^T = (\beta_0,\dots,\beta_t)$。

根据上述记号，R_0,...、R_t、L_0,...、L_t 满足以下包含 $2(t+1)$ 个约束的线性系统

$$
R_k - (1-p)\sum_{i=0}^{t-k-1} p^i L_{k+i+1} = \alpha_k,\ \ k = 0,\dots,t, \qquad (4)
$$

$$
L_k - (1-p)\sum_{i=0}^{t-k-1} p^i R_{k+1+i} = \beta_k,\ \ k = 0,\dots,t. \qquad (5)
$$

Nex我们定义矩阵$P \in \mathbb{R}^{(t+1)\times(t+1)}$，其中$P_{i,\ j} = -(1-p)p_{j-i-1}$（若$j > i$），否则为 0，即

$$
P = -(1-p)\begin{pmatrix}
0 & p^0 & p^1 & p^2 & \dots & p^{t-2} & p^{t-1} \\
0 & 0 & p^0 & p^1 & \dots & p^{t-3} & p^{t-2} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \dots & p^0 & p^1 \\
0 & 0 & 0 & 0 & \dots & 0 & p^0 \\
0 & 0 & 0 & 0 & \dots & 0 & 0
\end{pmatrix},
$$

使线性系统（4）、（5）可以写成如下矩阵形式：$Ax = b$，其中

$$
A := \begin{pmatrix} I_{t+1} & P \\ P & I_{t+1} \end{pmatrix}, \qquad (6)
$$

I_{t+1}表示（$t+1$）×（$t+1$）单位矩阵，x^T =（R_0,R_1,...，R_t，L_0,L_1,...，L_t），而b^T =（α^T，β^T）。为简便起见，我们将I_{t+1} 简写为I。

引理 6. 若矩阵 I $-P^2$ 可逆，则分块矩阵 A 可逆，且

$$A^{-1} = \begin{pmatrix} (I-P^2)^{-1} & -(I-P^2)^{-1}P \\ P(I-P^2)^{-1} & I+P(I-P^2)^{-1}P \end{pmatrix}$$

证明。 通过直接进行矩阵乘法即可得出。 □

接下来，我们的目标是计算 A^{-1}。我们以分块形式求其逆矩阵。

引理 7. 设 $\delta_1 = 1$ 且

$$\delta_m = \frac{1-p}{2}\left(1-(-1+2p)^{m-2}\right), \text{ for all } m \geq 2.$$

则矩阵 $B := I$ $-P^2$ 可逆，且

$$B^{-1}_{i,j} = \begin{cases} 0 & \text{if } i > j \\ \delta_{j-i+1} & \text{if } j \geq i \end{cases}$$

证明（略）。 容易看出，$\left(I-P^2\right)_{i,i} = 1$，并且 $\left(I-P^2\right)_{i,i+1} = 0$。对于所有 $j \geq i+2$，都有

$$\begin{aligned} \left(I-P^2\right)_{i,j} &= -\left(P^2\right)_{i,j} = -\sum_{r=1}^{t+1} P_{i,r}P_{r,j} = -\sum_{r=i+1}^{j-1} P_{i,r}P_{r,j} \\ &= -(1-p)^2 \sum_{r=i+1}^{j-1} p^{r-i-1}p^{j-r-1} = -(j-i-1)(1-p)^2 p^{j-i-2}. \end{aligned}$$

接下来我们证明 $\left(I-P^2\right)B = B\left(I-P^2\right) = I$。证明采用关于 t 的归纳法（回忆一下，$B \in \mathbb{R}^{(t+1)\times(t+1)}$）。

当 $t=1$ 时，情形显然，因为此时有 $B = I - P^2 = I$。对于 $t \geq 2$ 的情形（当 $B \in \mathbb{R}^{(t+1)\times(t+1)}$ 时），注意到 <style id='B 的主子式与取 $t \leftarrow t-1$ 所得到的矩阵相同。

因此，为了证明 $B\left(I-P^2\right) = I$，只需证明对所有 $r = 0,..., t$，有

$$\left(B\left(I-P^2\right)\right)_{r,t+1} = \left(B\left(I-P^2\right)\right)_{t+1,r} = \left(\left(I-P^2\right)B\right)_{r,t+1} = \left(\left(I-P^2\right)B\right)_{t+1,r} = 0,$$

以及那个

$$\left(B\left(I-P^2\right)\right)_{t+1,t+1} = \left(\left(I-P^2\right)B\right)_{t+1,t+1} = 1.$$

我们证明第一个 等式，其中 $r=1$，其余等式的计算也非常相似。事实上，对于 $r=1$，我们有 e

$$\begin{aligned} \left(B\left(I-P^2\right)\right)_{1,t+1} &= \sum_{r=1}^{t+1} B_{1,r}(I-P^2)_{r,t+1} \\ &= -\sum_{r=1}^{t-1} B_{1,r}(t-r)(1-p)^2 p^{t-r-1} + B_{1,t+1} \\ &= -(t-1)(1-p)^2 p^{t-1-1} \\ &\quad -\sum_{r=3}^{t-1} \frac{1-p}{2}\left(1-(-1+2p)^{r-2}\right)(t-r)(1-p)^2 p^{t-r-1} \\ &\quad +\frac{1-p}{2}\left(1-(-1+2p)^{t-1}\right) \\ &= 0. \end{aligned}$$

□

以下是引理7的直接推论。

推论2. $\left((I-P^2)^{-1}P\right)_{1,1}=0$对于$r=2,\ldots,t+1$，有$\left((I-P^2)^{-1}P\right)_{1,r}=-\frac{1}{2}(1-p)\left((2p-1)^{r-2}+1\right)$。证明。利用引理 7，我们计算

$$\begin{aligned}\left((I-P^2)^{-1}P\right)_{1,r}&=\sum_{i=1}^{t+1}\left((I-P^2)^{-1}\right)_{1,i}P_{i,r}\\&=P_{1,r}-\frac{1}{2}(1-p)^2\sum_{i=3}^{r-1}\left(1-(-1+2p)^{i-2}\right)p^{r-i-1}\end{aligned}$$

由此可得

$$\begin{aligned}\left((I-P^2)^{-1}P\right)_{1,1}&=P_{1,1}=0\\\left((I-P^2)^{-1}P\right)_{1,2}&=P_{1,2}=-(1-p)\\\left((I-P^2)^{-1}P\right)_{1,3}&=P_{1,3}=-p(1-p),\end{aligned}$$

且对所有$i\geq 4$,

$$\begin{aligned}\left((I-P^2)^{-1}P\right)_{1,i}&=P_{1,r}-\frac{1}{2}(1-p)^2\sum_{i=3}^{r-1}\left(1-(-1+2p)^{i-2}\right)p^{r-i-1}\\&=-\frac{1}{2}(1-p)\left((2p-1)^{r-2}+1\right).\end{aligned}$$

□

现在我们可以计算R_0。由式（6）和引理6，可得

$$\begin{aligned}R_0&=A^{-1}\begin{pmatrix}\alpha\\\beta\end{pmatrix}\\&=\sum_{j=1}^{t+1}\left((I-P^2)^{-1}\right)_{1,j}\alpha_{j-1}-\sum_{j=1}^{t+1}\left((I-P^2)^{-1}P\right)_{1,j}\beta_{j-1}\\&=\alpha_0+\sum_{j=2}^{t+1}\left((I-P^2)^{-1}\right)_{1,j}\alpha_{j-1}-\sum_{j=2}^{t+1}\left((I-P^2)^{-1}P\right)_{1,j}\beta_{j-1}.\end{aligned}$$

现在，利用引理 7和推论 2，我们继续计算上一个表达式 as

$$\begin{aligned}R_0&=\alpha_0+\frac{1-p}{2}\left(\sum_{j=2}^{t+1}\left(1-(2p-1)^{j-2}\right)\alpha_{j-1}+\sum_{j=2}^{t+1}\left(1+(2p-1)^{j-2}\right)\beta_{j-1}.\right)\\&=\alpha_0+\frac{1-p}{2}\left(\sum_{j=2}^{t+1}\left(\alpha_{j-1}+\beta_{j-1}\right)+\sum_{j=2}^{t+1}(2p-1)^{j-2}\left(\beta_{j-1}-\alpha_{j-1}\right)\right)\end{aligned}$$

现在利用式（2）和式（3），并展开前面的公式，可得

$$R_0=\frac{(1-p)g^{t+1}\left(1+(1-2p)\left(g+(g-1)(-1+2p)^t\right)\right)}{(g-1)(1-gp)}-\frac{2(1-p)}{g-1}+n. \tag{7}$$

我们观察到，对于所有t和$g\geq 2$，都有$g+(g-1)(-1+2p)^t\geq 0$。因此，

$$\begin{aligned}\sup_{n\in(g^t,g^{t+1}]}\frac{R_0}{n}&=\frac{R_0}{g^t}\\&\leq\frac{(1-p)g}{(g-1)(1-gp)}\left(1+(1-2p)\left(g+(g-1)(-1+2p)^t\right)\right)+1\\&=\frac{(1-p)g}{(1-gp)}\left(\frac{1+(1-2p)g}{g-1}-\left(-1+2p\right)^{t+1}\right)+1\\&\leq\frac{(1-p)g}{(1-gp)}\left(\frac{1+(1-2p)g}{g-1}+\left(1-2p\right)^{t+1}\right)+1\end{aligned}$$

如前所述。

D 引理证明3

与引理的证明2 中一样，我们引入以下记号（例如，使用$R(x)$，而不是R_x 以便区分各个证明）。

定义1. 令 $R(x)$ 表示向右移动且初始预定转向点为 $g^{x+\epsilon}$ 时的期望终止时间，其中 x 为非负整数。

定义2. 令 $L(x)$ 表示向左移动且初始预定转向点为 $g^{x+\epsilon}$ 时的期望终止时间，其中 x 为非负整数。

根据上述定义，且由于智能体最初以 1/2 的概率向左或向右移动，我们可以计算总期望终止时间：

$$\frac{1}{2}L(0)+\frac{1}{2}R(0).$$

然后，我们可以用总期望终止时间除以（位置最不利的）目标与原点之间的距离（该距离可任意大），从而求出期望竞争比。

现在假设目标位于原点[7] 右侧的$n=g^{t+\delta}$处，其中t是一个非负整数，且 $\delta\in(0,1]$。由此可知，智能体到达目标有两种可能情况。一种是 $\epsilon<\delta$，另一种是 $\epsilon\geq\delta$，其中 ϵ 是从 $[0,1]$ 中均匀随机地抽取的随机算法选择。

因此，我们自然地将 $L(x)$ 与随机变量 $L_{\epsilon<\delta}$ 和 $L_{\epsilon\geq\delta}$相关联，同时将 $R(x)$ 与 $R_{\epsilon<\delta}$ 和$L_{\epsilon\geq\delta}$相关联，分别对应于两种情况 $\epsilon<\delta$ （见第 D.1节）和 $\epsilon\geq\delta$ （见第 D.2节）。在这种情况下，我们有

$$L(0)=\int_0^\delta L_{\epsilon<\delta}(0)+\int_\delta^1 L_{\epsilon\geq\delta}(0),$$
$$R(0)=\int_0^\delta R_{\epsilon<\delta}(0)+\int_\delta^1 R_{\epsilon\geq\delta}(0).$$

D.1 第一种情况（$\epsilon<\delta$）

本节中的函数L、R特指 $\epsilon<\delta$ 的情形，即它们分别是$L_{\epsilon<\delta}$ 和$R_{\epsilon<\delta}$的缩写。接下来，我们递归地表示$R(x)$、$L(x)$。事实上，我们有

$$\begin{aligned}
R(t+1)&=n,\quad \textit{(Since intended turning point is } g^{t+1+\epsilon}>g^{t+\delta}=n)\\
R(t)&=pR(t+1)+(1-p)(2g^{t+\epsilon}+L(t+1))\\
&=pn+(1-p)(2g^{t+\epsilon}+L(t+1))\\
R(t-1)&=pR(t)+(1-p)(2g^{t-1+\epsilon}+L(t))\\
&=p^2n+(1-p)p(2g^{t+\epsilon}+L(t+1))+(1-p)(2g^{t-1+\epsilon}+L(t))
\end{aligned}$$

此外，对于任意$x\geq t+1$，有$R(x)=n$，因为预定转向点到起点的距离大于目标到起点的距离，因此在到达预定转向点之前就会找到目标。

对于某个 k 的 x，且 k 小于 t，我们区分两种情况。一种情况是改变方向成功（这发生的概率为 p）。在这种情况下，智能体移动至 g，然后返回原点；从那里开始，我们需要加上 L(k) 的期望终止时间。另一种情况是改变方向失败（这发生的概率为 p）。此时，智能体移动到下一个预定转向点（即 g）。这种情况

[7] 如果目标在左侧，则分析过程类似。

与智能体移动到 g 的情形相同，而 g 实际上就是 R(k)。因此，在一般情况下，对于某个 k 的 x，且 k 小于 t，向右移动的期望终止时间为

$$R(k) = pR(k+1) + (1-p)(2g^{k+\epsilon} + L(k+1)).$$

通过展开递归表达式 我们得到以下仅依赖于$L(x)$的递归表达式 .

$$R(k) = np^{t+1-k} + (1-p)2g^{k+\epsilon}\sum_{i=0}^{t-k}(pg)^i + (1-p)\sum_{i=0}^{t-k}p^i L(k+1+i) \tag{8}$$

按照类似的论证，我们计算$L(x)$。不同之处在于，智能体向左移动时没有目标，因此智能体会一直移动，直到实际改变方向。智能体改变方向时会到达原点，然后我们加上向右移动的期望时间，即$R(x)$；其中的x取决于从左侧覆盖的距离。

$$\begin{aligned}
L(t+1) &= (1-p)(2g^{t+1+\epsilon} + R(t+2)) + (1-p)p(2g^{t+2+\epsilon} + R(t+3)) + \dots \\
&= (1-p)2g^{t+1+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=0}^{\infty}p^i R(t+2+i) \\
&= \frac{(1-p)2g^{t+1+\epsilon}}{1-pg} + (1-p)\sum_{i=0}^{\infty}p^i n \\
&= \frac{(1-p)2g^{t+1+\epsilon}}{1-pg} + n
\end{aligned}$$

同理，我们有

$$\begin{aligned}
L(t) &= (1-p)(2g^{t+\epsilon} + R(t+1)) + (1-p)p(2g^{t+1+\epsilon} + R(t+2)) + \dots \\
&= (1-p)2g^{t+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=0}^{\infty}p^i R(t+1+i) \\
L(t-1) &= (1-p)(2g^{t-1+\epsilon} + R(t)) + (1-p)p(2g^{t+\epsilon} + R(t+1)) + \dots \\
&= (1-p)2g^{t-1+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=0}^{\infty}p^i R(t+i).
\end{aligned}$$

由此我们推导出$L(k)$的一般表达式，其中$k < t+1$，如下

$$\begin{aligned}
L(k) &= (1-p)(2g^{k+\epsilon} + R(k+1)) + (1-p)p(2g^{k+1+\epsilon} + R(k+2)) + \dots \text{ up to infinity} \\
&= (1-p)\sum_{i=0}^{t-1-k}(2g^{k+\epsilon+i} + R(k+1+i)) + (1-p)\sum_{i=t-k}^{\infty}p^i(2g^{k+\epsilon+i} + R(k+1+i)) \\
&= (1-p)2g^{k+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=0}^{t-k-1}R(k+1+i) + (1-p)\sum_{i=t-k}^{\infty}p^i n.
\end{aligned}$$

因此，我们有

$$L(k) = (1-p)2g^{k+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=0}^{t-k-1}R(k+1+i) + (1-p)\sum_{i=t-k}^{\infty}p^i n. \tag{9}$$

接下来，对于$k=0$、...、t，我们设定

$$\begin{aligned}\alpha_k :=& np^{t+1-k} + (1-p)2g^{k+\epsilon}\sum_{i=0}^{t-k}(pg)^i + (1-p)p^{t-k}L(t+1)\\ =& np^{t+1-k} - np^{t+1-k} + np^{t-k} + \frac{2g^{t+1+\epsilon}}{1-pg}(1-p)^2p^{t-k} + \frac{2(1-p)g^{k+\epsilon}(1-(pg)^{t-k+1})}{1-pg}\\ =& \frac{2g^{t+1+\epsilon}(1-p)^2p^{t-k}}{1-pg} + \frac{2g^{k+\epsilon}(1-p)}{1-pg} - \frac{2g^{k+\epsilon}(1-p)(pg)^{t-k+1}}{1-pg} + np^{t-k}\\ =& \frac{2(1-p)g^{k+\epsilon}}{1-pg}\left((1-p)p^{t-k}g^{t+1-k} + 1 - (pg)^{t-k+1}\right) + np^{t-k}\\ =& \frac{2(1-p)g^{k+\epsilon}}{1-pg}\left(1 + g(pg)^{t-k} - 2(pg)^{t-k+1}\right) + np^{t-k}\\ =& \frac{2(1-p)g^{k+\epsilon}}{1-pg}\left(1 + (1-2p)g(pg)^{t-k}\right) + np^{t-k} \end{aligned} \tag{10}$$

$$\begin{aligned}\beta_k :=& (1-p)2g^{k+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=t-k}^{\infty}p^i n\\ =& \frac{2(1-p)g^{k+\epsilon}}{1-pg} + np^{t-k}, \end{aligned} \tag{11}$$

此外，我们还设置 $\alpha^T = (\alpha_0,...,\alpha_t), \beta^T = (\beta_0,...,\beta_t)$。

记住上述记号，并结合方程（8）、（9）、（10）和（11），我们得到$R(0),...,R(t)$ 和 $L(0),...,L(t)$ 满足以下由$2(+1$个约束构成的线性系统

$$R(k) - (1-p)\sum_{i=0}^{t-k-1}p^i L(k+i+1) = \alpha_k,\ \ k=0,\dots,t, \tag{12}$$

$$L(k) - (1-p)\sum_{i=0}^{t-k-1}p^i R(k+1+i) = \beta_k,\ \ k=0,\dots,t. \tag{13}$$

该系统具有与第 15 页上引理 2 在 C 节证明中所考虑的系统相同的变量系数。由于我们通过求逆系数矩阵求解了前一个系统，因此这里使用相同的逆矩阵，并通过初等代数计算（矩阵乘法）得到

$$R(0) = n + \frac{g^{\epsilon}(-1+p)(-2+g(2p+g^t(1+g-2gp-(-1+g)(-1+2p)^{1+t})))}{(-1+g)(-1+gp)} \tag{14}$$

如果我们在（$\epsilon=0$ 中设定 14），实际上会得到与（7）相同的结果。由于这是 $\epsilon<\delta$ 的情况，为了计算给定 $\epsilon<\delta$ 时的期望终止时间，我们必须计算

$$\int_0^{\delta} R_{\epsilon<\delta}(0)\,d\epsilon$$

这给出了

$$n\delta + \frac{(-1+g^{\delta})(-1+p)(-2+g(2p+g^t(1+g-2gp-(-1+g)(-1+2p)^{(1+t)})))}{(-1+g)(-1+gp)\ln g}. \tag{15}$$

D.2 第二种情况（$\epsilon \geq 0$）

接下来我们分析 $\epsilon \geq \delta$ 的情况。同样为方便记号起见，仅在本节中，我们分别将 $L_{\epsilon\geq\delta}$和 $R_{\epsilon\geq\delta}$ 简写为 L、R。我们通过像之前一样建立递推关系，计算 $L(x)$、$R(x)$。

$$
\begin{aligned}
R(t) &= n, \quad \textit{(Since intended turning point is } g^{t+\epsilon} > g^{t+\delta} = n\textit{)} \\
R(t-1) &= pR(t) + (1-p)(2g^{t-1+\epsilon} + L(t)) \\
&= pn + (1-p)(2g^{t-1+\epsilon} + L(t)) \\
R(t-2) &= pR(t-1) + (1-p)(2g^{t-2+\epsilon} + L(t-1)) \\
&= p^2 n + (1-p)p(2g^{t-1+\epsilon} + L(t)) + (1-p)(2g^{t-2+\epsilon} + L(t-1))
\end{aligned}
$$

类似地，对于任意 $x \geq t$，我们有 $R(x) = n$。

注意到，当前的分析与前一种情况几乎完全相同；事实上，区别在于现在目标提前一步到达。因此，在一般情况下向右移动到 $x = k$（对于某个 $k < t$）时，预期的终止时间为

$$
R(k) = pR(k+1) + (1-p)(2g^{k+\epsilon} + L(k+1)).
$$

展开 t his 递归表达式，我们得到如下仅依赖于 $L(x)$ 的递归表达式 .

$$
R(k) = np^{t-k} + (1-p)2g^{k+\epsilon} \sum_{i=0}^{t-1-k} (pg)^i + (1-p) \sum_{i=0}^{t-1-k} p^i L(k+1+i) \tag{16}
$$

类似地，我们计算某些有用 x 值对应的 $L(x)$。得到

$$
\begin{aligned}
L(t) &= (1-p)(2g^{t+\epsilon} + R(t+1)) + (1-p)p(2g^{t+1+\epsilon} + R(t+2)) + \ldots \\
&= (1-p)2g^{t+\epsilon} \sum_{i=0}^{\infty} (pg)^i + (1-p) \sum_{i=0}^{\infty} p^i R(t+1+i) \\
&= \frac{(1-p)2g^{t+\epsilon}}{1-pg} + (1-p) \sum_{i=0}^{\infty} p^i n \\
&= \frac{(1-p)2g^{t+\epsilon}}{1-pg} + n.
\end{aligned}
$$

类似地，我们有

$$
\begin{aligned}
L(t-1) &= (1-p)(2g^{t-1+\epsilon} + R(t)) + (1-p)p(2g^{t+\epsilon} + R(t+1)) + \\
&= (1-p)2g^{t-1+\epsilon} \sum_{i=0}^{\infty} (pg)^i + (1-p) \sum_{i=0}^{\infty} p^i R(t+i).
\end{aligned}
$$

由上式可推导出某个 $k < t+1$ 对应的 $L(k)$ 的一般表达式如下

$$
\begin{aligned}
L(k) &= (1-p)(2g^{k+\epsilon} + R(k+1)) + (1-p)p(2g^{k+1+\epsilon} + R(k+2)) + \ldots \\
&= (1-p) \sum_{i=0}^{t-2-k} (2g^{k+\epsilon+i} + R(k+1+i)) + (1-p) \sum_{i=t-1-k}^{\infty} p^i (2g^{k+\epsilon+i} + R(k+1+i)) \\
&= (1-p)2g^{k+\epsilon} \sum_{i=0}^{\infty} (pg)^i + (1-p) \sum_{i=0}^{t-2-k} R(k+1+i) + (1-p) \sum_{i=t-1-k}^{\infty} p^i n.
\end{aligned}
$$

因此我们得出结论：

$$
L(k) = (1-p)2g^{k+\epsilon} \sum_{i=0}^{\infty} (pg)^i + (1-p) \sum_{i=0}^{t-2-k} R(k+1+i) + (1-p) \sum_{i=t-1-k}^{\infty} p^i n. \tag{17}
$$

接下来，对于 $k=0$、.....、t，我们设定：

$$\begin{aligned}
\alpha_k :=& np^{t-k} + (1-p)2g^{k+\epsilon}\sum_{i=0}^{t-1-k}(pg)^i + (1-p)p^{t-1-k}L(t) \\
=& np^{t-k} - np^{t-k} + np^{t-1-k} + \frac{(1-p)2g^{k+\epsilon}(1-(pg)^{t-k})}{1-pg} + \frac{2g^{t+\epsilon}(1-p)^2p^{t-1-k}}{1-pg} \\
=& \frac{2g^{t+\epsilon}(1-p)^2p^{t-1-k}}{1-pg} + \frac{(1-p)2g^{k+\epsilon}}{1-pg}(1-(pg)^{t-k}) + np^{t-1-k} \\
=& \frac{2g^{k+\epsilon}(1-p)}{1-pg}\left((1-p)p^{t-1-k}g^{t-k} + 1 - (pg)^{t-k}\right) + np^{t-1-k} \\
=& \frac{2g^{k+\epsilon}(1-p)}{1-pg}\left(g(pg)^{t-1-k} - (pg)^{t-k} + 1 - (pg)^{t-k}\right) + np^{t-1-k} \\
=& \frac{2(1-p)g^{k+\epsilon}}{1-pg}\left(1 + (1-2p)g(pg)^{t-1-k}\right) + np^{t-1-k} \qquad (18) \\
\beta_k :=& (1-p)2g^{k+\epsilon}\sum_{i=0}^{\infty}(pg)^i + (1-p)\sum_{i=t-1-k}^{\infty}p^i n \\
=& \frac{2(1-p)g^{k+\epsilon}}{1-pg} + np^{t-1-k}, \qquad (19)
\end{aligned}$$

此外，我们还设定 $\alpha^T = (\alpha_0,...,\alpha_t), \beta^T = (\beta_0,...,\beta_t)$。

与前面一样，根据方程（16）、17、18和19，可知$R(0),...,R(t-1)$ 和 $L(0),...,L(t-1)$ 满足以下含有 $2t$ 个约束

$$R(k) - (1-p)\sum_{i=0}^{t-2-k} p^i L(k+1+i) = \alpha_k, \quad k = 0,\dots,t-1, \qquad (20)$$

$$L(k) - (1-p)\sum_{i=0}^{t-2-k} p^i R(k+1+i) = \beta_k, \quad k = 0,\dots,t-1. \qquad (21)$$

求解该系统，可得 $R(0)$ 如下。

$$R(0) = n + \frac{g^{\epsilon}(-1+p)(-2+g(2p+g^{t-1}(1+g-2gp-(-1+g)(-1+2p)^t)))}{(-1+g)(-1+gp)} \qquad (22)$$

若在（*14*）中将$t-1$代入，可推出（22）。此外，由于这里属于$\epsilon \geq \delta$的情形，我们计算得

$$\int_{\delta}^{1} R(0)\, d\epsilon$$

化简后得

$$n - n\delta + \frac{(g-g^{\delta})(-1+p)(-2+g^t+g^{1+t}+2gp-2g^{1+t}p-(-1+g)g^t(-1+2p)^t)}{(-1+g)(-1+gp)\ln g}. \qquad (23)$$

D.3 合并两种情形

现在我们已准备好结合这两种情况，计算总的期望终止时间。更具体地说，我们有：

$$R(0) = \int_0^{\delta} R_{\epsilon<\delta}(0) + \int_{\delta}^{1} R_{\epsilon\geq\delta}(0).$$

利用（15）和（23），我们得到

$$n\left(1+\frac{(1-p)(-1+g(-1+2p))(1+(-1+2p)^t)}{(-1+gp)\ln g}\right)-\frac{2(-1+p)}{(-1+gp)\ln g}+\frac{2g(-1+p)(p+g^t(-1+p)(-1+2p)^t)}{(-1+gp)\ln g}. \quad (24)$$

按目标距离 n 进行缩放，即可得到所述竞争比

$$1+\frac{(1-p)(-1+g(-1+2p))(1+(-1+2p)^t)}{(-1+gp)\ln g}-\frac{2(-1+p)}{n(-1+gp)\ln g}+\frac{2g(-1+p)(p+g^t(-1+p)(-1+2p)^t)}{n(-1+gp)\ln g} \quad (25)$$

其中 $n\in(g_t,g_{t+1}]$。

E 定理6的证明

这里给出定理6证明的全部细节。

证明（定理6）。

为便于说明，我们再次给出算法5 并对其进行分析。

阶段 *1*：首先，两个p-故障智能体以全速朝相反方向开始搜索，直到找到目标（由于这是初始移动，选择方向时不可能出错）。不失一般性，我们假设发现者在时间 1 报告了目标，此时目标也会立即传达给未发现者。

阶段 *2*：此时，未发现者尝试转向（可能不会成功），并将其速度切换为$s=s(p)<1$，其值将在之后确定。现在我们把注意力转向目标发现者。发现者成为领导者，并选择 $\gamma=\gamma(p)>0$。因此，发现者会越过目标，持续时间为 γ，然后借助目标（此时扮演静止跟随者的角色）运行算法4，直到再次到达目标。

阶段 *3*：发现者继续朝同一方向前进，直到遇到未发现者。请注意，无论未发现者是否成功转向，这次相遇都以概率 1 实现，因为未发现者的速度现在是$s<1$。

阶段 *4*：一旦会面实现，未发现者便担任跟随者，发现者仍为领导者。跟随者停留在原地，领导者沿相同方向（远离目标和跟随者）继续移动 γ 时间，然后运行算法 4，其中跟随者作为不移动的智能体，以完成转向。

阶段 *5*：一旦转向完成，两个p故障智能体便向目标移动 .

接下来，我们计算期望终止时间。记W为一个随机变量，表示发现者在阶段 2 中两次连续访问之间的时间。再记T为一个随机变量，表示两个智能体在阶段 4 中第二次会面所需的时间。

回到阶段 2，我们考察与未发现者有关的独立事件。一方面，以概率$1-p$，未发现者成功转向，因此当发现者第二次到达目标并开始向未发现者移动时，两者相距 $2-sW$，相对速度为 $1+s$。这意味着它们将在额外时间 $(2-sW)/(1+s)$ 后相遇，在此期间，未发现者向目标靠近了 $s(2-sW)/(1+s)$ 的距离。因此，当两个智能体相遇时，它们与目标的距离为 $2-sW-s(2-sW)/(1+s)$。

另一方面，以概率 p，未发现者未能转向，因此当发现者第二次到达目标并开始朝未发现者移动时，两者相距 $2+sW$，且相对速度为 $1-s$（因为它们朝同一方向移动）。这意味着它们会在额外的时间 $(2+sW)/(1-s)$ 后相遇，在此期间，未发现者远离目标又移动了 $s(2+sW)/(1-s)$。因此，当两个智能体相遇时，它们距离目标的距离为 $2+sW+s(2+sW)/(1-s)$。

但是，根据期望的线性性，终止时间（等于竞争比）为

$$\begin{aligned}&1+\mathrm{E}[W]+\mathrm{E}[T]+p\left(\mathrm{E}[(2+sW)/(1-s)+2+sW+s(2+sW)/(1-s)]\right)\\&\quad+(1-p)\left(\mathrm{E}[(2-sW)/(1+s)+2-sW-s(2-sW)/(1+s)]\right)\\&=\frac{5-s(s+4-8p)}{1-s^2}+\mathrm{E}[T]+\frac{s(4p+s-2)+1}{1-s^2}\mathrm{E}[W]\end{aligned}$$

我们选择 $s = s(p) = \frac{1-2\sqrt{p-p^2}}{1-2}p$，显然对于所有 $p \in (0,1/2)$，其取值都在 $(0,1)$ 内，因此这是一个有效的速度选择。[8] 现在我们将其代回竞争比公式，并进行代数化简，得到

$$3 + 4\sqrt{(1-p)p} + \mathrm{E}[T] + 2\sqrt{(1-p)p}\mathrm{E}[W].$$

主要结论由以下事实可得：记住 $\mathrm{E}[T]$,$\mathrm{E}[W]$ 都与 γ成正比，因此可以将它们变得任意小。

□

[8] s的取值选择为使 $5-s(s+4-8p)$ $1-s^2$ 在所有参数 $0 < p < 1/2$ 上达到极小值点的取值。