给定路径上多机器人协调与有界交叉点复杂度

Mikkel Abrahamsen 哥本哈根大学，丹麦 miab@di.ku.dk

Tzvika Geft 特拉维夫大学，以色列 zvigreg@mail.tau.ac.il

Dan Halperin 特拉维夫大学，以色列 danha@tauex.tau.ac.il

Barak Ugav 特拉维夫大学，以色列 barakugav@gmail.com

arXiv:2303.00745v1 [cs.RO] 1 Mar 2023

摘要

我们研究多机器人/多智能体系统中的一个基本 NP-困难运动协调问题：给定一个图 G 和一组智能体，其中每个智能体在 G 中都有一条给定的有向路径。每个智能体初始位于其路径的第一个顶点。在每个时间步，只要下一个顶点未被其他智能体占据，智能体就可以沿其路径移动到该顶点。目标是寻找一系列沿给定路径执行的此类移动，使每个智能体都到达其目标；如果不存在这样的移动序列，则报告无解。该问题为基于导引路径的运输系统建模，这一抽象与当代多种应用中的交通密切相关，应用范围从铁路网络或工厂中的自动导引车（AGV），到电脑游戏动画，再到量子计算中的量子比特传输。它还作为更一般的多机器人运动规划问题的一个子问题出现。

我们通过考虑新的假设，并确定问题仍保持 NP-困难时关键参数的最小值，对该问题进行了细粒度的可处理性分析。我们的分析确定了一个称为顶点重数 （VM）的关键参数，其定义为经过同一顶点的路径数量的最大值。我们证明，该问题的一个常见变体——等价于顺序资源分配（涉及并发进程的死锁预防）——即使在 VM 为 3 时仍是 NP-困难的。另一方面，当 VM ≤ 2 时，我们给出了一种高效算法，能够迭代地解决智能体之间阻塞关系环。我们还提出了一个变体：即使 VM 为 2，当 G 是一个二维网格且每条路径都位于同一网格行或列中时，该变体仍是 NP-困难的。通过研究高度提炼但仍为 NP-困难的变体，我们加深了对问题难以处理之原因的理解，从而引导在实际假设下寻找高效解。

关键词

多机器人运动规划；多智能体寻路；预定义路径；顺序资源分配；调度；复杂性

ACM参考文献格式：

Mikkel Abrahamsen、Tzvika Geft、Dan Halperin 和 Barak Ugav。2023年。具有有界交叉口复杂性的给定路径上的多机器人协调。载于第*22*届自主智能体与多智能体系统国际会议论文集（*AAMAS 2023*），英国伦敦，*2023*年*5*月*29*日至*6*月*2*日，IFAAMAS，15页。

第*22*届国际自主智能体与多智能体系统会议论文集（*AAMAS 2023*），*A. Ricci*、*W. Yeoh*、*N. Agmon*、*B. An*（编），*2023*年*5*月*29*日至*6*月*2*日，伦敦，英国。© 2023年国际自主智能体与多智能体系统基金会（www.ifaamas.org）。版权所有。

1 引言

我们研究如何协调一组机器人/智能体的运动[1]，使其沿指定路径行进。

该问题源于基于导引路径的车辆，例如用于工业环境 [28] 中的自动导引车和高架单轨系统。这类环境通常具有高度结构化的特点，并通过集中控制将车辆限制在预定义路径上行驶。此类系统的一个关键组成部分是确保活性，即车辆能够完成分配给它们的任务，并执行类似的后续任务。车辆的运动可能不可逆，例如车辆无法在铁路上倒退，因此可能导致死锁，进而丧失活性。在进入新状态之前（例如为车辆分配一条新路径），最好先检查能否保持活性。一般来说，这项检查需要确定是否存在一系列运动，使所有智能体都能完成当前行程；而这归根结底就是要解决我们的问题。这种状态活性检查问题称为活性保持监督，离散事件系统（DES）领域对此已有广泛关注 [27]。

另一个研究动机来自智能交通领域。近年来的实践表明，在混合交通或城市区域运行自主车辆仍极具挑战，因此短期内不太可能普及。一种更可行的安全运行方式是采用专用基础设施（例如导向通道、轨道或专用车道），由于与人类驾驶员、行人和障碍物的交互有限，这种方式更为简单。不同于常规道路网络，这类基础设施将车辆限制在有限的路径集合上行驶，因而最终适用于我们的研究场景。预计这类受限自主系统将不再局限于简单的拓扑结构和固定时刻表（例如机场接驳车）[17]，因此需要更复杂的运动协调。例如，为满足不断增长且更加灵活的需求（例如按需服务带来的需求），允许驶向相反方向的车辆共用同一路径段，或许能够提高系统效率。事实上，针对

[1]本文中交替使用智能体 和机器人 这两个术语。按照运动规划文献中的术语习惯，在连续域中移动的对象通常称为机器人，而在图上移动的对象则称为智能体（或卵石等）。这种区分有时并不明确，因为连续运动规划问题通常会被转化为图上智能体的运动规划问题。

专用基础设施交通系统的特殊结构开发算法，近期已被列为城市交通和物流领域的一个研究方向 [17]。

我们的问题属于多机器人运动规划 （MRMP）这一更广泛的研究领域。在 MRMP 中，我们并未获得完整的路径，而只知道每个机器人/智能体的起点和目标位置；我们的目标是找到一种无碰撞运动，使所有机器人都能到达各自的目标。在各种平面场景中，一般问题已被证明是 PSPACE困难的 [4, 13, 15, 33]。即使所有机器人依次移动，该问题仍是 NP-困难的 [11]。为使问题变得易于处理，研究者提出了一些松弛条件，对机器人起点和目标位置之间的间距作出假设 [1, 32, 34]。MRMP 的离散对应问题（智能体在图上移动）可在多项式时间内求解 [18, 38]。然而，若要寻找最优解（例如针对时间或距离目标），该问题便会变为 NP-困难，即使是在二维网格图上也是如此 [2, 8, 10]。近期，研究者通过近似算法应对了这一难解性 [8, 36, 37]。我们指出，关于 MRMP 变体的研究成果极为丰富，在此无法一一详述。

尽管由于不预先约束路径，MRMP 比我们研究的问题有更大的自由度，但给定路径的MRMP（简称 MRMP-GP）可作为求解 MRMP 的子程序。求解 MRMP 的一种常见范式称为解耦规划：先在不考虑其他机器人的情况下，为每个机器人规划路径，然后再根据各自的路径协调机器人，例如调整机器人沿路径行进的速度。早期研究主要关注机器人数量固定时该问题的变体 [16]。在 [22], 中，协调图用于处理两个机器人；该图表示各机器人路径上可能发生相互碰撞的位置。之后，协调图被推广到多机器人情形 [19] ，并用于协调它们沿指定路径的运动 [23, 31]。然而，该图的规模在最坏情况下会随机器人数量呈指数级增长。另一类研究关注已规划路径（更准确地说，是轨迹）的执行 。在实践中，很难保证机器人不会偏离规划轨迹。当机器人因外部干扰等原因意外延误时，它们可能无法继续遵循最初有效的计划。因此，始终需要在机器人沿规划路径移动时对其运动进行验证和协调（必要时重新规划）。这类研究包括 [3, 5, 6, 14]。

尽管人们一直对这一问题保持浓厚兴趣，但据我们所知，首批解释为何缺乏高效算法的 NP困难性结果直到相对较近的时期才出现 [27, 28]。这些结果作为一系列关于资源分配问题中死锁预防的研究工作的一部分被提出 [24], , 并且与 DES领域相关；这类问题在各种自动化场景中具有广泛的适用性 [25]。此类问题涉及将有限的一组可复用资源分配给一组并发执行进程，其中每个进程只有按照特定顺序获取资源才能执行。我们的问题可以视为最简单的一类资源分配问题，即线性单单位资源分配系统（L-SU-RAS）；该问题是 NP-困难的 [24, 28]。按照 L-SU-RAS 的术语，每个机器人模拟一个进程，该进程需要一次“获取”序列中的一个位置，而每个位置一次只能容纳一个机器人。

我们观察到以往困难性构造存在以下不理想的性质：(i) 顶点重数无界，即存在一个“拥堵”位置，必须有无界数量的机器人经过。(ii) 有些路径段必须沿相反方向通行，即存在固有的正面碰撞风险。(iii) 机器人的路径并非其端点之间的最短路径。特别是，机器人必须沿路径多次经过同一顶点（平面情形的证明中如此）。我们在 *Table* 1中总结了以往的困难性结果。可以说，这些性质未必能代表真实系统，因此有必要对这些性质不成立的情形展开新的分析。

贡献。 本文对机器人沿固定路径运动的协调问题进行了精细化复杂性研究。我们考虑两个参数：顶点重数和路径形状复杂度，后者是机器人在网格上转弯次数的最大值（确切定义见第2节）。我们考虑两种主要问题变体：(1) 一种重要变体，对应于 L-SU-RAS资源分配问题；(2) 一种研究较少的变体，其中上述性质(ii)，即存在反向路径，不成立。对于每种变体，我们都确定了使问题仍为 NP-困难的参数临界值；低于该值时，问题可高效求解。由此，我们为否定与肯定结果之间确立了更精确的界限。

我们的主要正面结果针对变体（1）：我们提出了一种高效算法，可求解顶点重数为 2 的任意图上的问题。由此，我们扩大了可高效求解的实例类别。我们的解的核心是一种迭代过程，用于消解智能体之间的阻塞关系环。对于每个这样的环，我们构造一种特殊类型的有向图，称为路径构成的图。我们反复收缩该图，直至其达到一个问题等价的简化且不可约状态；在此状态下，可以轻松确定阻塞关系能否解开，否则便检测到死锁，且该实例无解。该算法的运行时间与给定路径的总长度呈线性关系。

在负面结果方面，我们证明顶点重数为 3 时，该问题是 NP-困难的。此外，对于变体（2），我们证明：当智能体沿二维网格图上的直线路径移动且顶点重数为 2 时，该问题仍是 NP-困难的；与此前的困难性结果相比，这一条件的限制严格得多。

我们的细粒度复杂性分析呼应了近期关于深入理解多智能体运动协调问题难点的呼吁 [9, 12, 29]。加深对这些问题的理解，有助于指导我们探索更优的算法和新的可高效求解的问题变体。我们将在结论中探讨这一方向上的潜在进展。

2 问题定义与假设

本节将正式定义我们研究的问题，以及所考虑的假设和参数。

给定路径的多机器人运动规划 *(MRMP-GP)*。 我们给定一组 R 个在工作空间 W 中运行的 n 个机器人，其中

表1：与先前困难性结果的比较。问题变体和参数的确切定义见第2节。最后一列（“最短路径？”）中的勾号表示构造中给定的每条路径都是其起点顶点与目标顶点之间的最短路径。我们用“min”表示某个参数的最小值；低于该值时，相应的问题变体可高效求解。

问题变体	论文	图 type	顶点重数	最大转弯次数 在路径中	最短路径?
NBT	[20]	一般	无界	n/a	✓
	[27]	平面	无界	n/a	
	[28]	二维网格	无界	>20	
	ours	**一般**	**3（最小值）**	**n/a**	✓
		二维网格	**4**	**1（最小值）**	✓
UNI	**我们的方法**	**二维网格**	**2（最小值）**	**0（最小值）**	✓

是一个有限无向图。W 的顶点，记作 $V(W)$，也可称为 位置。每个机器人 $r \in R$ 都有一个起始顶点 $s\ r\ \in V\ W$ $t\ r\ \in( )$ V $( )$（也称为 源点）和一个目标顶点 $( )(W)$，以及一条路径 $( ) = v_1, v_\ell$，其中 $v_1 = s$ $( )\ v_\ell = t( )$，且对于所有 $1 \le i < \ell$，$v_i v_{i+1}$ 是 W 的一条边。注意，由于 $\pi(r)$ 是一条路径，每个顶点在 $\pi(r)$ 中只出现一次。

每个机器人最初位于其起始顶点。MRMP-GP要求找到一个运动规划或解，即一系列移动，使每个机器人沿给定路径从源点到达目标，同时不与其他机器人发生碰撞。合法移动是指单个机器人移动到其路径上的下一个顶点 v，前提是没有其他机器人位于 v。不允许后退移动，即机器人不得移动到其路径上的前一个顶点。特别是，机器人一旦到达目标，就会停留在那里。

图 1 展示了一个实例。请注意，MRMP-GP 是一个可行性问题，而不是优化问题；因此，在一个解中，我们一次只移动一个机器人。

变体。现在我们定义所考虑问题的两种变体。设 r和 $r\prime$是两个不同的机器人。若 t（$r\prime$）位于 π（r）上，则称 t（$r\prime$）是 r的阻塞目标。如果某个实例不存在阻塞目标，则称其具有非阻塞目标属性。这种情况对应于 L-SU-RAS资源分配问题 [26]，因为在该问题中，进程完成后实际上就从系统中消失了。这等价于某个机器人的目标不是任何其他机器人的阻塞目标，因为我们可以认为机器人一到达目标就消失。

对于我们的第二种变体，若边 $(u,\ v)$ in$W$必须由不同的机器人沿两个方向通行，则称其为双向边；也就是说，存在两个机器人 r 和 $r\prime$，其中 $(u,\ v)$ 出现在 $\pi(r)$ 上，而 $(v,\ u)$ 出现在 $\pi(r\prime)$ 上。如果一个实例不包含双向边，则称其具有单向或单向运动。

参数。对于顶点 $v \in V(W)$，我们用 $N(v)$ 表示包含该顶点的路径数量，即 $N(R$ $M) := |\{$ $r \mid v \in \pi( )\}|$。我们将 MRMP-GP 实例的顶点重数定义为 VM(M)，即 max $_{v \in V(W)} N(v)$。对于 W 为二维网格的情况，我们考察路径形状复杂度。我们将转弯数定义为转弯次数的最大值，即最小

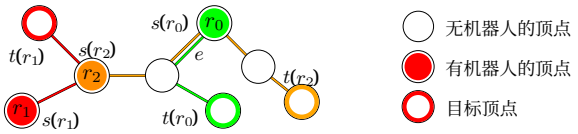

图 1：一个具有W 8 个顶点的 MRMP-GP实例。其中有3 个机器人，每个机器人都有自己的源点、目标和路径，并以该机器人的颜色显示。该实例具有非阻塞目标属性，但不具有单向运动（由于边e）。其顶点重数为 2。该实例的一种可行解是按照 r_0、 r_2、 r_1的顺序，将每个机器人一直移动到其目标位置。（虽然本例中不需要，但一般而言，我们允许机器人停留在其路径上的中间顶点，而其他机器人继续移动。）

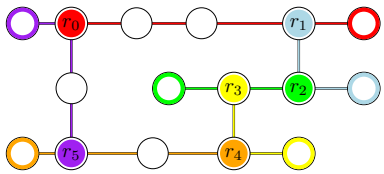

图 2：由6 个机器人组成的简单环，r_0、……、r_5，其中每个机器人都阻碍前一个机器人到达其目标。

由任意输入路径在网格上绘制该路径所需的线段数量减一。

简写。本文中我们使用以下简写。对于具有非阻塞目标的变体，我们使用 MRMP-GP-NBT(x)，其中 x表示顶点重数。类似地，对于具有单向运动的变体，我们使用 MRMP-GP-UNI(x)。当W为二维网格时，我们还将转弯数作为第二个参数，例如，MRMP-GP-UNI(2,0) 表示顶点重数为 2 且路径为直线路径（即 0 次转弯）。

3 算法

现在考虑 MRMP-GP-NBT(2)，即 MRMP-GP 的一种变体，具有非阻塞目标属性，且顶点重数至多为 2。我们提出一种多项式时间算法，可找到一个解，或报告无解。该算法分为两个阶段，下面将分别介绍。

阶段*1*。 位于顶点 $v \in \pi$（r）的机器人 r若其路径畅通，即其路径的剩余部分上没有其他机器人，也就是说，从 v到 t（r）的 π（r）子路径上没有其他机器人，则称其路径畅通。鉴于机器人的目标不在任何其他机器人的路径上，将路径畅通的机器人移至其目标后，未来不会阻挡其他机器人。因此，我们将每个路径畅通的机器人移至其目标，直至不存在此类机器人。请注意，移动一个机器人可能会使另一个机器人的路径畅通，因此我们会反复检查这些机器人。

阶段*2*。 接下来，算法通过迭代识别并解决环。每次迭代解决一个环，这意味着只移动该环中的机器人。每次迭代开始时，以下不变量成立：

每个机器人要么（i）已到达其目标顶点，要么（ii）仍停留在其源顶点，且没有畅通路径。

阶段1结束时，不变量显然仍然成立。如果从 r 当前位置到 t（r）的路径经过 $r_\prime$ 的当前位置，则称机器人 r 被另一机器人 $r_\prime$ 阻塞。在每次迭代开始时，任何尚未到达目标的机器人都会被另一机器人阻塞。设 r_0 是这样一台机器人，r_1 是阻塞它的机器人，因此 r_1 必定尚未到达其目标 t（r_1）。那么，机器人 r_1 尚未到达其目标，因此路径 π（r_1）上必定有另一台机器人 r_2，它阻塞了 r_1，以此类推。由阻塞关系可得到一个序列 r_0、r_1、r_2，……，r_{h-1}，其中 r_{h-1} 是第一个被序列中已出现的机器人 r_i 阻塞的机器人，即 $i < h-1$。图 2 展示了一个示例。为简化记号，从此处起，我们对机器人编号采用模 h 运算，即用 r_i 代替 $r_{(i \bmod h)}$。

引理 3.1。阻塞序列 $r_0, r_1, \ldots, r_{h-1}$ 在 r_{h-1} 被某个机器人 r_i 阻塞时结束（其中 $0 \le i < h-1$），并形成一个环；也就是说，每个机器人 r_i 都被 r_{i+1} 阻塞。

证明。根据定义，任何机器人 r_j，$j < h-1$ 都被 r_{j+1} 阻塞。因此，只需证明 r_{h-1} 被 r_0 阻塞。假设 r_{h-1} 被机器人 r_i 阻塞，且 $i > 0$，则会产生矛盾。这意味着机器人 r_{i-1}，r_i，r_{h-1} 的路径都经过起始顶点 s（r_i），这与每个顶点最多出现在两个机器人的路径中的假设相矛盾。□

因此，所有尚未到达目标的机器人都可以划分为互不相交的环。算法会独立地解决每个环（其含义将在下文定义），然后将环上的机器人移至各自的目标。这是可行的，因为正如下文的引理 3.2 所述，它们的路径必定畅通。

设定一个环 $C = r_0$，r_1，……，r_{h-1}。每个机器人 $r_i \in C$ 当前都位于自己的源顶点。求解环 C 是指将每个机器人 r_i 移动到下一个机器人的起始位置，即 $s(r_{i+1})$，而 $s(r_{i+1})$ 称为 r_i 的环目标。

引理 3.2。在 C 求解后，C 中的所有机器人都有一条畅通路径。

证明。反设在求解 C 后，C 中的某个机器人 r_i 被另一个机器人 r 阻挡。若 r 不属于 C，则 r 属于另一个环，并在那里阻挡另一个机器人 $r_\prime$。这意味着 r 在其当前顶点处阻挡了两个机器人，这与顶点重数为 2 的假设矛盾。若 r 也属于 C，则 r 此前被 C 中的机器人 $r_\prime$ 阻挡。求解 C 后，r 位于源顶点 $s(r_\prime)$，这意味着 $s(r_\prime)$ 出现在 3 个机器人的路径上，即 r、$r_\prime$ 和 r_i

，这再次与顶点重数为 2 的假设相矛盾。□

求解一个环。机器人 r_i 从 s（r_i）到 s（r_{i+1}）的子路径称为机器人 r_i 的环路径。如果机器人 r_j 的环路径上有一个顶点 p，且该顶点未出现在 C 中其他机器人的环路径上，那么我们称 r_j 为侦察者。例如，图 2 中的 r_0 就是一个侦察机器人。如果 C 中包含一个侦察机器人，那么可以先将该侦察机器人移动到 p，再按照从侦察机器人开始的逆序，沿各自的环路径移动其余机器人，从而求解 C。这个简单情形的完整细节见附录A.1。

如果 C 中没有侦察机器人，则解可能不存在；参见图3。为了确定一个环是否可解，我们构造一个有向图 G，其顶点为 $V(G) \subset V(W)$，边为 $E(G)$。顶点 $V(G)$ 是环路径上的顶点的子集，而

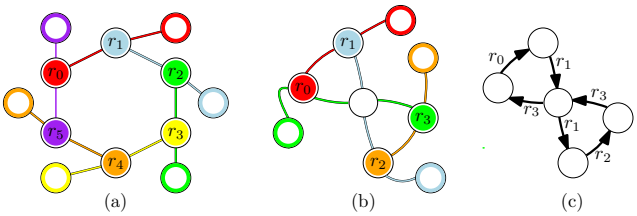

图3：不可解实例，每个实例都只包含一个环：(a) 一种没有机器人可以移动的死锁。(b) r_1 或 r_3 均可移动，但无论哪种移动，都会导致三个机器人陷入类似 (a) 的死锁。(c) 算法针对 (b) 中的环构造的图 G。

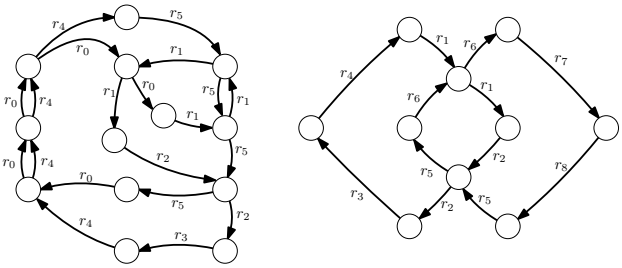

图4：左：由路径组成的图，根据引理 3.4，该图可解。右：该图并非由路径组成，因为这些边没有构成所需类型的欧拉回路。

C 中的机器人。对于每个 C 中的机器人 r 以及 r 的环路径上的每条有向边 e，我们将边 e 添加到 $E(G)$。我们为这条边标记 r，以记住是哪个机器人引入了它。因此，G 中可能存在从一个顶点到另一个顶点的两条边，但它们将分别标记为不同的机器人。我们称这种带标签的边组成的有向图为<样式 id='11'>由路径组成</样式>；参见图4。注意，图 G 具有以下性质：

- 对于每个不同的标签，带有该标签的边构成一条路径。
- 每个顶点恰好出现在两个标签对应的路径上。
- 一条路径的终点是另一标签对应路径的起始顶点。
- 一条路径的起始顶点是另一标签对应路径的终点。
- 在 G 中存在一条欧拉回路，使得对于每个标签，带有该标签的边在回路中连续经过。

任何具有这些性质的带标签边的有向图都称为由路径组成。此类图的起始顶点 G 是任意标签对应路径的起始顶点，我们将起始顶点的集合记为 $V_s(G)$。

由路径组成的图 G 表示一个等价的机器人循环，其构造方式如下：对于 G 中由所有标签为 r 的边组成的路径 π，我们将机器人 r 放置在 π 的起始顶点上，机器人 r 必须沿路径 π 移动。图 3(c)对此作了说明。如果这个 MRMP-GP 实例存在运动规划，我们就称 G 可解。

我们先证明一个关于由路径构成的图 G 的度的初等引理。令 $\delta\text{in}(v)$ 和 $\delta\text{out}(v)$ 分别表示图 G 中顶点 v 的入度和出度。

引理 3.3. 设 G 是一个由路径构成的图，v 是 G 中的一个顶点。若 $v \in V_s(G)$，则有 $(\delta$入度$(v)\ \delta$出度$(v)) = (1, 1)$；否则 $(\delta$入度$(v)\ \delta$出度$(v)) = (2, 2)$。

证明。考虑 $v \in V_s(G)$。根据 G 的性质，一个标签的路径从 v 开始，而另一个标签的路径以 v 结束。由于 v 只出现在两个标签的路径上，结论得证。

现在考虑顶点 $v \in V(G) \setminus V_s(G)$。根据 G 的性质，顶点 v 包含在两个标签的路径中，并且不是它们中任一者的目标顶点。这意味着，在这两条路径上，都有边通向 v，也有边从 v 引出，因此 $(\delta\text{in}(v)\ \delta\text{out}(v)) = (2, 2)$。□

现在我们转而对由路径构成的可解图进行分类。引理3.4和3.5给出了可解图与不可解图的**基本情形**：

引理 3.4。考虑由路径构成的图 G。如果 G 中的所有简单环都至少有两个不是起始顶点的顶点，那么 G 是可解的。

引理 3.5。如果由路径构成的图 G 包含一个所有顶点都是起始顶点的简单环，那么 G 不可解。

引理 3.5 的证明见附录 A.2。我们暂缓证明引理 3.4，先描述求解相应 MRMP-GP 实例的算法；图 5 展示了一个非平凡示例。

该算法首先按如下方式将机器人划分为若干块。考虑 G 中的一条极大有向路径 π，其至少包含 3 个顶点，且除首尾顶点外，每个顶点都是起始顶点。令 r_1、……、r_k、$k \geq 1$ 依次表示从这些起始顶点出发的相应机器人，并令 $B_1 = (r_1$、……、$r_k)$。遍历所有此类路径 π，即可得到机器人块 B_1、B_2……。

对于一个块 $B_i = (r_1$、……、$r_k)$，我们定义 头部$(B_i) = r_k$ 和 尾部$(B_i) = r_1$。机器人 r_k 的环目标 位于另一个块 B_j 中机器人 r_{1} 的起始顶点。接着，我们定义 后继$(B_i) = B_j$ 和 前驱$(B_j) = B_i$。由于路径构成的图的性质，这些关系对所有块 B_i 均有定义。

现在我们来描述算法；伪代码见算法 1。首先，任意选择一个区块 B_i。然后，将 B_i 中的所有机器人向前移动一条边，从 头部(B_i) 移至 尾部(B_i)。沿其环路径移动 头部(B_i)，直到无法再移动为止。令 $B_j =$ 下一个(B_i) 为包含阻挡 头部(B_i) 的机器人 尾部(B_j) 的区块。将机器人 B_j 向前移动一条边，然后将头部 (B_i) 移至其环目标，即 尾部 (B_j) 的原始位置。尽可能移动 头部(B_j)。此后，重复这一模式，直到遍历完所有区块。

请注意，该算法可能会移动区块的头部三次。或许有人会想，算法第 8 行可以省略（改为在第 7 行尽可能向前移动 头部(前一个(B))），但事实并非如此，如图 6 所示。

引理 3.4 的证明。我们证明，算法 1 求解 $I(C)$，即对应于环 C 的 MRMP-GP 实例 G。求解该环时，我们将每个机器人区块 B_i 从头部移至尾部。显然，只有移动 头部(B_i) 才可能是无效移动，因为 B_i 中其余机器人总是被移至一个刚被前一个移动的机器人腾空的顶点。

算法 1： 求解一个 MRMP-GP 实例，该实例对应于由满足引理 3.4 要求的路径组成的图 G

```
1  B_开始 ← B_1;
2  B ← B_开始;
3  循环
4      对于 r ∈ B，按从头部(B) 到尾部(B) 的顺序执行
5          将 r 向前移动一条边;
6      如果 B 不是B_起点，则
7          将头部(前驱(B))向前移动一条边;
8      尽可能向前移动头部(B);
9      B ←下一个(B);
10     如果 B 是B_起点 则
11         返回;
```

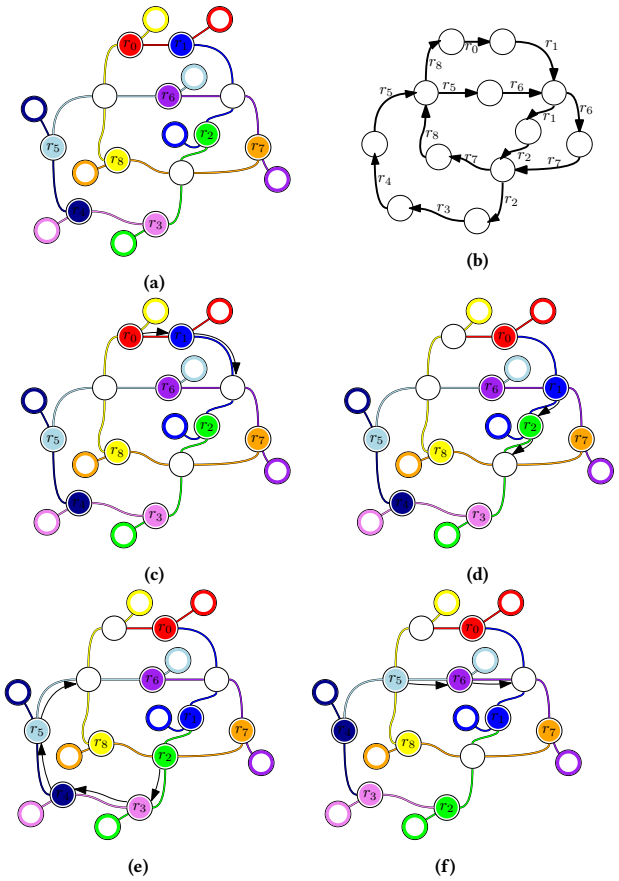

图 5：（a）一个包含单个阻塞环的实例。（b）为该单个环创建的图 G，其中所有简单环都包含两个空闲（非起点）顶点。最大连续机器人块为 $B_1 = (r_0, r_1)$ $B_2 = (r_2)$ $B_3 = (r_3, r_4, r_5)$ $B_4 = (r_6)$ $B_5 = (r_7)$ $B_6 = (r_8)$。我们展示通过算法 1 得到的解中的几个步骤：（c）$B_1 = (r_0, r_1)$ 是第一个选定的块，机器人从头部移动到尾部。（d）下一个(B_1)、$B_2 = (r_2)$ 中的机器人从头部移动到尾部，然后再次移动头部(B_1)。（e）下一个(B_2)、$B_3 = (r_3, r_4, r_5)$ 中的机器人从头部移动到尾部，然后移动头部(B_2)。（f）$B_4 = (r_6)$ 和头部(B_3) 被移动。

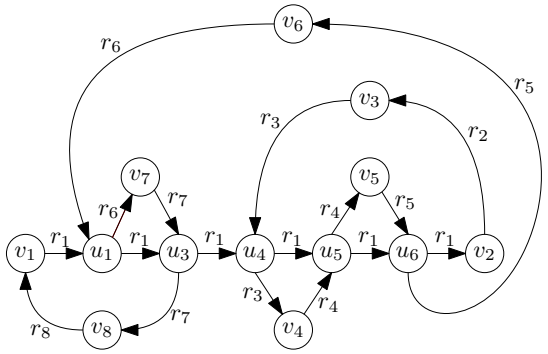

图 6：假设我们先将机器人 r_7 从 v_7 移动到 u_3。如果接下来按照算法 1 的指示，移动 r_1 和 r_8，那么就不可避免地要移动 r_1 三次。事实上，清空 u_3 后，r_1 必须移动到 u_6（或 u_5），并等待 v_2 被清空。否则，如果 r_1 留在 u_1，那么在腾出 v_2 时，顶点 u_4，u_5，u_6 中的一个会被占据，因此 r_1 无法在一次移动中移至 v_2。

将分别被尾部和头部占据的顶点记为 u，v（如果 B_i 仅由一个机器人组成，则 u，v 可能是同一个顶点）。u，v 均属于 $V_s(G)$，因此根据引理 3.3，u 只有一条入边，该边来自另一个顶点 w；而 v 只有一条出边，该边通向另一个顶点 z。w，z 均未被机器人占据，且不属于 $V_s(G)$，否则这些机器人就会被纳入 B_i。若 $w = z$，则 w 以及所有被 B_i 中机器人占据的顶点会构成一个仅含一个非起始顶点的简单环，这与引理的假设矛盾，因此 $w \neq z$。

在处理 B_i 的主循环（第 3 行）的一次迭代开始时，我们有如下不变量：$V(G) \setminus V_s(G)$ 中至多有一个顶点被机器人占据，且该被占据的顶点是指向顶点 s（尾部（B_i））的边的源点。换言之，该被占据的顶点就是上文定义的 w。现在我们来验证这一不变量得以保持。

初始时，所有机器人都位于 $V_s(G)$ 上，且不变量得以保持。现在假设在某次迭代开始时不变量成立。在移动 $B = B_i$ 的任何机器人之前，w 是唯一可能被占据的非起始顶点，因此 z 永远不会被占据；将头部(B_i)向前移动一步至 z 是有效的，因为 $w \neq z$。移动 B_i 的头部后，B_i 的其余机器人按从头部到尾部的顺序移动，占据 w 的机器人则移动到 B_i 尾部原先所在的位置。最后，尽可能移动 B_i 的头部，直到它到达 w，即 s 的入边（尾部(下一个(B_i))）的源顶点，并保持不变量。当处理完最后一个区块后，尽可能移动头部，使其到达 s(尾部(下一个(B_i)))。因此，该移动是有效的。

现在验证所有机器人都能到达各自的环目标位置。算法处理完单个区块 B_i 后，B_i中的所有机器人（B_i的头部除外）都能到达各自的环目标位置。作为处理下一个区块下一个(B_i)的一部分，B_i的头部会再次移动并到达其环目标位置，因此 B_i中的所有机器人都会沿各自的环路径行进。如果 B_i是最后处理的区块，那么其他区块中的所有机器人都已到达各自的环目标位置，包括 $tail(B_{起始})$；因此，当尽可能推进 $head(B_i)$ 时，它将到达其环目标位置，即 $tail(B_{起始})$的起始位置。所有区块 B_1，B_2，……都已检查，因此所有机器人都到达了各自的环目标位置，该环也就解决了。□

如果 G 不属于引理3.4和3.5的基本情形之一，我们就逐步将 G 化简为符合其中一种情形的图，具体如下。假设当前情况不属于基本情形，即 G有一个简单环 v_0，……，v_{k-1}，其中 v_0不是起始顶点，但其余所有顶点都是起始顶点。注意，根据引理 3.3，必有 (δ在(v_0) δ出(v_0)) =(2, 2)。由此可知，$E(G)$中与 v_0相连的边包括 (v_0, v_1)、(v_{k-1}, v_0)、(u, v_0) 和 (v_0, w)，其中 u、w是未出现在简单环 v_0，……，v_{k-1}中的顶点。此外，存在两个机器人 r和 r'，使得 (u, v_0) 和 (v_0, v_1) 标记为 r，而 (v_{k-1}, v_0) 和 (v_0, w) 标记为 r'，如图7(a)所示。如果四条边的标签都相同，均为 r，那么 r就会是一个侦察机器人。边 (v_{k-1}, v_0) 和 (v_0, v_1) 的标签不能相同，否则 v_0，……，v_{k-1}上的机器人将构成一个阻塞环，且独立于当前正在解决的其余阻塞环 C。

现在我们定义一个解缠操作，该操作会生成一个新图 $G_\prime$，其顶点为 $V(G_\prime) = V(G) \setminus \{v_0\}$；参见图7。在 $G_\prime$中，我们将标记为 r 的两条边替换为一条边（u, v_1），并保留标签 r。同样，我们将标记为 $r_\prime$ 的两条边替换为一条边（v_{k-1}, w），并将其标记为 $r_\prime$。

现在我们证明，解缠操作能够保持可解性。

引理 3.6。设 G 是一个路径构成的图，$G_\prime$ 是在 G 中执行解缠过程后得到的图。那么，*(i)* $G_\prime$ 也是一个路径构成的图；*(ii)* $G_\prime$ 可解，当且仅当 G 可解。

证明。考虑上文所述的解缠操作及其涉及的顶点 v_{k-1}、v_0、v_1、u、w。首先，很容易验证该操作能够保持路径构成的图的性质。现在假设 G 可解。两个机器人需要经过 v_0：将包含 u、v_0、v_1 的环路径的机器人记为 r_u，将包含 v_{k-1}、v_0、w 的环路径的机器人记为 r_w。v_0 上没有机器人，因此算法必须决定 r_u 和 r_w 中哪个先经过 v_0。如果在 r_w 之前将 r_u 移至 v_0，机器人就会陷入死锁，因为顶点 v_0、……、v_{k-1} 都会有机器人（即与引理 3.5 中相同的情况）。因此，r_w 必须在 r_u 之前移至 v_0。于是，r_w 先移至 v_0，随后最终移至 w；然后 r_u 移至 v_0，再移至 v_1。由此可知，在 G' 中，存在一种运动规划：r_w 先经过合并边（v_{k-1}，w），接着路径 v_1、……、v_{k-2} 上的所有机器人移动，最后 r_u 经过合并边（u，v_1）。

现在假设 $G_\prime$可解。于是我们可以在 G中模拟相同的解：当 r_w在 $G_\prime$中经过边 (r_{k-1}, w) 时，我们让它在 G中依次经过边 (r_{k-1}, v_0) 和 (v_0, v_w)。同样，当r_u在 $G_\prime$中经过 (u, v_1) 时，我们让它在 G 中依次经过边 (u, v_0) 和 (v_0, v_1)。G和 $G_\prime$中的其他边均相同，因此我们直接照搬相应的移动。这样便得到了 G的一个解。□

我们的算法通过不断解缠环，直至无法继续为止。由此得到一系列图 G_0，G_1、……、G_m，其中$G_0 = G$是原图，且每个 G_i 都是在 G_{i-1}中执行解缠操作后得到的。根据引理 3.6，这些图均由路径组成，并且所得图 G_m 可解，当且仅当原图 G 可解。此外，G_m中已无可继续解缠之处，因此，要么存在一个所有顶点上都有机器人的简单环，要么至少有两个顶点在所有简单环上都没有

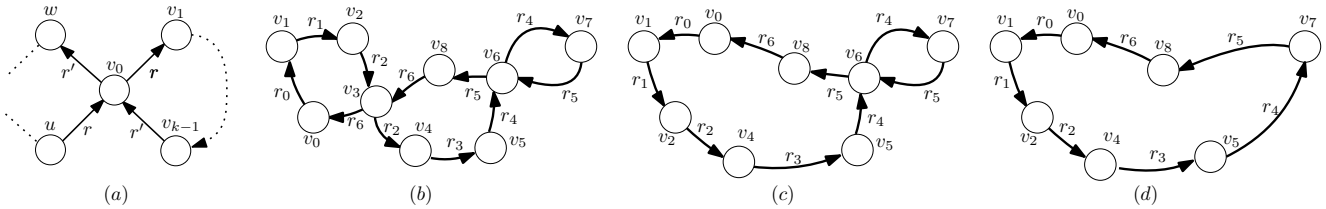

图7：（a）需要解缠的环 v_0、……、v_{k-1}。（b）需要进行两次解缠操作的路径构成的图，操作过程如（c）和（d）所示；完成后，我们得出结论：该图不可解。

机器人。由此可知，根据引理3.4和3.5，存在解，当且仅当属于后一种情况。

解开一个环后，环上的所有机器人都拥有通往其目标的畅通路径（见引理 3.2），算法会逐一将它们移至目标位置。阶段2的一次迭代至此结束，并且每个机器人都位于其起始位置或目标位置这一不变量得以保持，因为算法移动的机器人只有环上的机器人，且它们都已到达目标位置。

在算法的每一步中，我们只有在能够保证移动后存在解当且仅当移动前存在解时，才会移动机器人。因此，如果算法未能找到有效解，也就是说，某个环被解缠后形成了一个简单环，且所有顶点上都有机器人，那么就不存在解。否则，算法会为所有机器人找到有效的运动规划。

运行时间分析。 我们的算法可以在线性于所有机器人给定路径长度之和的时间内实现，我们将其记为 L。下面详细说明如何达到这样的运行时间。

为了高效实现阶段1，首先，对于每个机器人 r，我们找出第一个阻挡它的机器人。然后，将所有未被其他机器人阻挡的机器人移动到其目标位置。每当机器人 r 被移动到其目标位置时，如果 r 事先阻挡了机器人 r'，我们随后检查 r' 是否被其他机器人阻挡。我们只需检查 $\pi(r')$ 的后缀，从顶点 $s(r)$ 开始。因此，在阶段1中，给定路径中的每个顶点被检查的次数恰好等于它在给定路径中出现的次数。

为了高效实现阶段2，我们首先通过检查每个机器人被哪个机器人阻挡，将机器人划分为独立环，这需要 $O(L)$ 时间。然后，独立求解每个环 C。如果其中包含侦察机器人，那么显然所有计算都在线性于机器人路径长度之和的时间内完成。如果 C 不包含侦察机器人，我们将构造一个由路径 G 构成的图，其规模与机器人路径的长度呈线性关系。如果 G 属于引理3.4和3.5的基本情形之一，我们可以在线性于图规模的时间内轻松得到一个解，或判定不存在这样的解。

若 G 不属于这些基本情形之一，为了高效地执行所有解缠操作，首先计算最大连续机器人块 B_1，B_2，……。根据引理 3.3，任何被某个 B_i 中的机器人占据的顶点都只有一条入边和一条出边；而且，每个被机器人占据的顶点都包含在某个 B_i 中。因此，要找出仅有一个顶点未被机器人占据的简单环，只需检查是否存在某个 B_i，使得尾部(B_i)的入边的源顶点与头部(B_i)的出边的目标顶点相同。当一个块满足这一性质时，我们称其为结块。接下来，我们从任意一个块 B开始，利用 next(·) 关系迭代遍历这些块：如果 B 不是结块，就继续检查下一个块，即令 $B \leftarrow$ 为 next(B)。如果 B 是结块，则对尾部(B)的入边、头部(B)的出边以及二者共享的顶点 u 执行解缠操作，从而移除 u。如果 u 是它们之间唯一的顶点，则可能需要并合 B 与 prev(B)、next(B)；在这种情况下，下一个要检查的块是并合后的块，而不是 next(B)。遍历完所有块一次后即停止。每次迭代中，我们要么通过 next(·) 前进，要么至少并合 2 个块，因此迭代次数至多为 G_0（解缠前的原图）中的 $2|\{B_1, B_2, \ldots\}|$。每个非结块都会始终保持非结块状态。只有在并合因解缠操作而产生的 prev(B) B和 next(B)时，才可能生成新的结块；并且会在下一次迭代中检查这些新块。因此，完整遍历一次块 B_1, B_2,……后，不会再有更多结块。

解开环图 G后，它必然属于某一种基本情形，而这些情形决定了它是否可解。若可解，我们便使用算法 1计算机器人移动；该算法显然能在 $|V(G)|$的线性时间内运行。因此，阶段2中求解单个环 C的一次迭代，所需时间与 C中各机器人的路径长度之和成线性关系。由于每个环只求解一次，且每个机器人至多出现在一个环中，阶段2所需时间为 $O(L)$。

总体而言，该算法的两个阶段所需时间均与给定路径的总长度成线性关系。综上，我们得到以下结论：

定理 3.7. *MRMP-GP-NBT(2)* 存在一种算法，能够在线性于给定路径总长度的时间内找到一个解，或者报告不存在解。

4 困难性结果

本节将给出我们的困难性结果，具体如下：

定理 4.1。 以下 *MRMP-GP* 的受限变体均为 *NP* 完全：

(i) MRMP-GP-NBT(2, 0)，即工作空间是一个顶点重数为 *2* 的二维网格图。此外，每个智能体的路径都位于同一网格行或列中。*(ii) MRMP-GP-NBT(4, 1)*，即工作空间是一个顶点重数为 *4* 的二维网格图。此外，每个智能体的路径最多有一个转弯。*(iii) MRMP-GP-NBT(3)*，即顶点重数为 *3* 的一般图。

请注意，在 MRMP-GP-UNI 中，机器人只能单向运动，但允许存在阻塞目标（准确的定义见第2节）。与此相反，在 MRMP-GP-NBT 中，我们不允许存在阻塞目标，但允许双向运动，即机器人沿同一路径朝相反方向行进。因此，我们可以得出结论：仅双向运动或仅阻塞目标这两种因素中的任意一种，都足以使 MRMP-GP 变得难以求解。

由于我们在第3节提出的算法可以求解 MRMP-GP-NBT(2)，因此 MRMP-GP-NBT(3) 的困难性结果 (iii) 是紧的。也就是说，我们确立了基于顶点重数的 MRMP-GP-NBT 可处理性边界。我们还通过参数的最小性确定了其他可处理性边界，如表1所示；这很容易验证。要证明 MRMP-GP-NBT(4, 1) 的转弯数是最小的，我们可以将 MRMP-GP-NBT(4, 0) 归约到 MRMP-GP-NBT(2)。

我们分阶段证明定理 4.1，依次确立结果 (i)、(ii)，最后是 (iii)。这里我们概述证明过程，重点介绍 (i)；其余细节见附录 B。由于很容易验证 MRMP-GP 属于 NP，我们只讨论其 NP困难性。

首先，我们证明 MRMP-GP 的一个变体带优先约束的 *MRMP-GP*（*MRMP-GP-PC*）具有 NP困难性。MRMP-GP-PC 对机器人访问顶点的顺序施加了额外约束；我们借此抽象掉完整构造的细节，同时展示其构件的功能。作为概述的一部分，我们说明如何在二维网格上实现归约到 MRMP-GP-PC 所构造的实例 M_s。在完整证明中，我们逐步将 M_s 转换为同样难以求解的实例。也就是说，我们描述的每个后续实例 M' </style}当且仅当 M_s可解时才可解。

M_s 的转换步骤如下。首先，我们将 M_s 转换为 MRMP-GP-UNI(2,0) 的一个实例 M。为了实现M中的构件，我们使用阻塞目标。接下来，我们将 M 转换为 $M\prime$，并在其中消除所有阻塞目标。这些目标被方向相反的路径取代。此阶段需要进行以下大多属于局部的修改：设 r，$r\prime$为两个机器人，其中t（r）是位于顶点 v上的阻塞目标，且 $v \in \pi$（$r\prime$）。我们延长 π（r），使其沿着 π（$r\prime$）反向移动。此修改保留了这样的约束：在有效解中，$r\prime$必须先访问 v，再访问 $r\prime$；我们在构件中会用到这一点。这些修改将 $M\prime$的 VM 提高到 4，并使其成为 MRMP-GP-NBT(4,1) 的一个实例。最后，通过谨慎移除 $M\prime$ 中 VM 为=4 的顶点（这会使其不再是二维网格），我们得到 MRMP-GP- NBT(3) 的一个实例 $M\prime\prime$。

现在我们开始概述证明，重点讨论 MRMP-GP- UNI(2, 0) 的困难性。为了证明 MRMP-GP-PC 的困难性，我们引入一个名为枢轴调度的问题，并证明该问题是 NP-困难的。接下来，我们将枢轴调度归约到 MRMP-GP-PC。

枢轴调度。一个实例的形式为（V，C），其中 V 是成对出现的作业集合，C 是排序约束集合：设 $V = \{x_1, y_1, \ldots\ldots, x_n, y_n\}$ 为由 $2n$ 个互不相同的作业组成的集合。设 C $= \{C_1, \ldots\ldots, C_m\}$，其中每个 $C_j \in V^3$ 都是一个三元组。该问题[2]要确定是否可以将 V 划分为之前集合 B和之后集合 A（即 $A \cap B = \emptyset$和 $A \cup B = V$），使得满足以下约束：(i) 对于每一对 x_i，y_i，要么 $x_i \in B$ 要么 $y_i \in B$，且 (ii) 对于每个 $C_j \in$C，C_j 中必须有一个作业属于 A，即 $C_j \cap A \neq \emptyset$。

我们将前一类约束称为之前约束，将后一类约束称为之后约束。直观而言，之前/之后约束隐含地要求存在一个特殊的枢轴作业，输入作业必须相对于该作业进行排序。具体来说，B和 A 分别对应于排在枢轴作业之前和之后的作业。

引理4.2。枢轴调度是 *NP*-困难的。

证明。我们从 3SAT 归约而来。3SAT 是判定一个合取范式公式是否可满足的问题，其中每个子句包含三个文字。给定一个 3SAT 公式 ϕ，我们定义一个对应的枢轴调度实例 $S(\phi) = (V, C)$，如下所示：每个变量及其否定构成一对作业，属于V，即 $V = \bigcup_{x \in \phi}\{x, x\}$。至于之后约束集合 C，$\phi$ 中的每个子句都定义一个约束，该约束包含该子句中的文字。为简便起见，我们对作业及其对应的文字使用相同的记号。

为了完成归约，我们定义满足赋值 A 与 V 的有效划分之间的对应关系：当且仅当作业 x 属于之后集合（即 $x \in A$）时，变量 x 被赋值为“真”。换言之，给定 ϕ 的满足赋值，B 和 A 分别对应赋值为“假”和“真”的文字。很容易验证，所得划分满足约束，当且仅当 A 满足 ϕ。□

MRMP-GP-PC 的困难性。现在我们来定义 MRMP-GP-PC。输入与 MRMP-GP 相同，但还包含一些特殊顶点，我们将把它们用作构件。每个装置顶点都必须由经过它的机器人按照特定的遍历顺序遍历（即访问；归约中将具体说明所用的约束）。MRMP-GP-PC 的解与 MRMP-GP 的解相同，但还需满足一项附加要求：必须按照各自的遍历顺序访问构件顶点。

我们接下来进行归约。给定一个枢轴调度 $I = (V, C)$ 的实例，我们构造一个对应的 MRMP-GP-PC 实例 M_s。我们用 V 中对应的作业机器人 M_s表示每项作业。我们还用枢轴机器人 r^*表示隐式枢轴作业。为简化记号，我们对作业及其对应的机器人使用相同的名称。另一个机器人是 β，可以将其视为继续执行 r^* 行程的机器人。最后，对于每个约束$C_j = \{z_1, z_2, z_3\} \in V^3$，都有三个对应的检查机器人 c_j^1，c_j^2，c_j^3，我们用它们来模拟后约束。

现在我们讨论 M_s 中的构件。对于每个 $i \in [n]$，我们都有一个前约束构件，记为 B_i，它出现在 r^*、x_i 和 y_i 的路径上。B_i 的遍历顺序定义为：x_i 或 y_i 必须先于 r^* 遍历 B_i。对于每个 $j \in [m]$，我们都有一个后约束小工具，记为 A_j，它出现在 β 以及检查机器人 c_j^1，c_j^2，c_j^3 的路径上。A_j 的遍历顺序定义为：c_j^1，c_j^2，c_j^3 中的一个必须在 β 遍历之后遍历 A_j。我们使用的最后一种构件是优先约束装置，记为 P(r，$r\prime$)；必须先由 r 遍历，再由 $r\prime$ 遍历，其中 r 和 $r\prime$是任意机器人。我们使用的这一构件的具体实例将在下文说明。

现在我们来描述各构件沿路径的顺序。路径 $\pi(r^*)$ 首先经过所有前约束构件（顺序任意），然后经过优先约束装置 G πP (r^*,)。

[2]枢轴调度可视为广义与/或调度的一个特例，而后者是 NP-困难的 [21]。

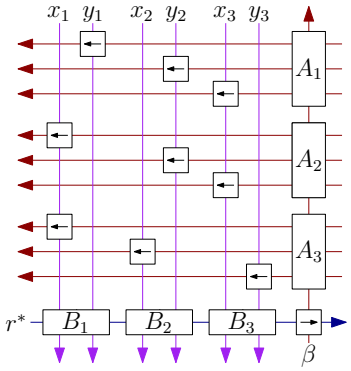

图8：与枢轴调度对应的一个 MRMP-GP-PC 实例，其中 $V = \{x_1, y_1, \ldots\ldots, x_3, y_3\}$ 且 $C = \{\{y_1, y_2, x_3\}, \{x_1, y_2, x_3\}, \{x_1, x_2, y_3\}\}$。机器人的路径以长箭头表示。优先级构件以包含短箭头的方框表示；短箭头的方向与需要先穿过该构件的机器人的路径方向一致。

路径 $\pi(\beta)$ 首先经过 G，然后经过所有后约束小工具（顺序任意）。由于 G 是沿 $\pi(r^*)$ 的最后一个构件，也是沿 $\pi(\beta)$ 的第一个构件，因此 β 本质上只能在 r^* 到达其目标后才开始移动。现在设 c_j^ℓ 是一台检查机器人，它对应于约束 C_j 中的作业 z。机器人 c_j^ℓ 首先经过 A_j，然后经过优先约束装置 P (c_j^ℓ, z)。这意味着，在一台作业机器人进入其对应的 B_i 构件之前，对应于该作业的所有检查机器人都必须先经过各自的 A_j 构件。

机器人与路径布局。图8展示了 M_s 的一个示例，该图既说明了如何将 M_s 转换为网格实例 M，也起到了示例说明的作用。该图应作如下理解：每个（长的、有颜色的）箭头表示一条位于同一网格行或列中的路径，矩形则表示构件。对于 M，每个矩形表示一个构件的位置，其中包含数量为常数的额外机器人（图中未示出）。对于 M_s，每个构件都是一个顶点，因此可以认为机器人的路径离开网格，前往某个顶点以访问该构件。注意，构件之外的顶点重数确实为 2。

我们按如下方式在二维网格上实现 M。M 的顶行最初包含作业机器人 x_1，y_1、……、x_n，y_n，它们按从左到右的顺序排列，必须向下移动到 M 的底行。M 的最右列最初包含检查机器人 c_1^1，c_1^2，c_1^3、……、c_m^1，c_m^2，c_m^3，它们按从上到下的顺序排列，必须向左移动到 M 的最左列。枢轴机器人 r^* 最初位于底行附近，必须从 M 的最左列移动到最右列。机器人 β 最初位于最右列附近，必须从 M 的底行移动到顶行。

现在我们来证明归约的正确性。对于 I 的一个划分 (B, A) 与 M_s 的一个解之间，存在如下对应关系：当且仅当对应的作业机器人在 r^* 之前经过 B_i 时，一个 x_i，y_i 作业对中的作业才属于之前集合 B。现在，令 (B', A') 为根据刚才定义的、由 M_s 的一个解得到的划分。显然，根据 B_i 的定义，B' 满足 I 的之前约束。现在我们证明，A' 满足之后约束：

引理 4.3。设 C_j 是 I 的一个后约束，并设 $R(C_j)$ 表示 M_s 中对应的作业机器人。那么，$R(C_j)$ 中的一个机器人会在 r^* 之后穿过其对应的前约束构件。

证明。假设每个机器人 $R(C_j)$都在 r^* 之前通过各自的前约束构件，这会导致矛盾。令r为 $R(C_j)$中最后一个这样做的机器人，并令 B_i为它通过的构件。现在考察 r位于 B_i的时间步。此时，r^* 位于 B_i左侧，而 C_j的所有检查机器人都已通过构件 A_j，这是由优先级构件所致。因此，β 必定已经通过 A_j，因为根据定义，它不可能是最后一个通过 A_j的机器人。特别地，β已经通过优先约束装置 G = P(r^*, β)（图8右下方）。这就产生了矛盾，因为 r^* 必须在 β之前通过 G。□

给定一个有效的作业划分，很容易验证，上述对应关系可直接导出 M_s 中机器人的一个排序，使它们能够逐个移动到各自的目标。因此，我们得到以下结论：

引理 4.4。*MRMP-GP-PC* 是 *NP* 完全的。

在二维网格上对 M 的完整实现，以及证明定理 4.1 的后续步骤，见附录 B。

5 结论

我们对 MRMP-GP 给出了更精细的复杂性分析，从而进一步揭示了该问题困难性的来源。以往 MRMP-GP 困难性构造的一个关键要素，是存在沿相反方向遍历同一组顶点的路径（例如，一条给定路径可能包含序列 v_1，v_2，v_3，而另一条路径可能包含 v_3，v_2，v_1）。我们证明，即使不存在此类路径，只要存在另一种要素，即阻塞目标，同样可能产生困难性。这一观察引出了一个耐人寻味的问题：当这两种要素都不存在时，MRMP-GP 是否仍然是困难的。肯定的答案可能会对固定路径机器人/运输系统产生影响，因为这类系统可以设计为避开上述要素。

从参数化复杂性的角度来看，[7], 对于困难的运动规划问题而言，这是一个研究方向，[29],我们的困难性结果排除了若干候选参数。具体而言，通过证明即使顶点重数（VM）为常数，问题仍然具有困难性，我们证明了以 VM 为参数时，MRMP-GP 不太可能是固定参数可处理（FPT）的。对于路径形状复杂度，同样的结论也成立。因此，我们将参数化算法的探索引向其他参数。我们认为，由于这些结果适用于高度提炼的 MRMP-GP 表述，我们的困难性构造更加清晰地揭示了潜在参数。

MRMP-GP 的一种自然扩展是优化解，例如优化其最大完成时间。这类优化变体与多智能体路径寻找密切相关 [35], , 后者可在其中作为子程序发挥潜在作用。例如，在常用的基于冲突的搜索算法 [30] 的每个高层节点中，都会为每个智能体找到一条路径，并且可以将其视为该节点中的固定路径。因此，MRMP-GP 优化变体的快速算法或许能够改进下界

，从而更好地引导搜索。我们相信，我们在判定可行性方面的新见解为此类未来研究方向提供了更好的基础。

致谢

M. Abrahamsen 获得丹麦独立研究基金通过 Sapere Aude研究职业计划提供的 Starting Grant 1054-00032B 资助，并参与哥本哈根基础算法研究（BARC）；该项目由 VILLUM基金会资助项目16582 支持。T. Geft 和 D. Halperin 关于本文的研究部分获得了以色列科学基金会（资助编号1736/19）、NSF/美国-以色列双边科学基金会（资助编号2019754）、以色列科学技术部（资助编号103129）、Blavatnik 计算机科学研究基金以及特拉维夫大学机器学习 Yandex机器学习倡议的支持。T. Geft 还获得了特拉维夫大学 Shlomo Shmeltzer智能交通研究所和以色列智能交通研究中心提供的奖学金。

参考文献

[1] Aviv Adler、Mark de Berg、Dan Halperin 和 Kiril Solovey。2015。《简单多边形中无标签圆盘的高效多机器人运动规划》。*IEEE*自动化科学与工程汇刊12，4（2015），1309–1317。[2] Jacopo Banfi、Nicola Basilico 和 Francesco Amigoni。2017。《具有孔洞的二维网格图上的时间最优多机器人路径规划的不可处理性》。*IEEE*机器人与自动化快报2，4（2017），1941–1947。[3] Alexander Berndt、Niels van Duijkeren、Luigi Palmieri 和 Tamás Keviczky。2020。《通过持续优化可切换动作依赖图来重新排序多智能体执行调度的反馈方案》。*CoRR* abs/2010.05254（2020）。[4] Thomas Brocken、G. Wessel van der Heijden、Irina Kostitsyna、Lloyd E. Lo-Wong 和 Remco J. A. Surtel。2021。《k色圆盘的多机器人运动规划是 PSPACE困难的》。载于*FUN*（*LIPIcs*，第*157*卷）。Schloss Dagstuhl－莱布尼茨信息学中心，15:1–15:16。[5] Michal Cáp、Jean Gregoire 和 Emilio Frazzoli。2016。《在延迟扰动下安全且无死锁地执行多机器人规划，可得到理论保证》。载于*IROS*。IEEE，5113–5118。[6] Adem Coskun、Jason M. O'Kane 和 Marco Valtorta。2021。《扰动下多机器人协调中的无死锁在线规划修复》。载于*FLAIRS*会议。[7] Marek Cygan、Fedor V. Fomin、Lukasz Kowalik、Daniel Lokshtanov、Dániel Marx、Marcin Pilipczuk、Michal Pilipczuk 和 Saket Saurabh。2015。参数化算法。第5卷。施普林格。[8] Erik D. Demaine、Sándor P. Fekete、Phillip Keldenich、Henk Meijer 和 Christian Scheffer。2019。《协同运动规划：以有界拉伸重构一群有标签机器人》。*SIAM*计算杂志48，6（2019），1727–1762。[9] Eric Ewing、Jingyao Ren、Dhvani Kansara、Vikraman Sathiyanarayanan 和 Nora Ayanian。2022。《多智能体路径寻找中的介数中心性》。载于*AA- MAS*。国际自主智能体与多智能体系统基金会（IFAAMAS），400–408。[10] Tzvika Geft 和 Dan Halperin。2022。《距离最优多智能体路径寻找的精细复杂性下界》。载于*AAMAS*。国际自主智能体与多智能体系统基金会（IFAAMAS），481–488。[11] Tzvika Geft、Dan Halperin 和 Yonatan Nakar。2021。《多机器人协调与几何重构的可解性前沿》。*arXiv*预印本 *arXiv:2104.07011*（2021）。[12] Ofir Gordon、Yuval Filmus 和 Oren Salzman。2021。《重新审视基于冲突的搜索的复杂性分析：新的计算技术与改进界》。载于*SOCS*。AAAI出版社，64–72。[13] Robert A. Hearn 和 Erik D. Demaine。2005。《通过非确定性约束逻辑计算模型证明滑块拼图及其他问题的PSPACE完全性》。理论计算机科学343，1-2（2005），72–96。[14] Wolfgang Hönig、Scott Kiesel、Andrew Tinka、Joseph W. Durham 和 Nora Ayanian。2019。《仓库中多智能体路径寻找调度的持续稳健执行》。*IEEE*机器人与自动化快报4，2（2019），1125–1131。[15] John E. Hopcroft、Jacob Theodore Schwartz 和 Micha Sharir。1984。《多个独立对象运动规划的复杂性；"仓库管理员问题"的 PSPACE困难性》。国际机器人研究杂志3，4（1984），76–88。[16] Kamal Kant 和 Steven W Zucker。1986。《迈向高效轨迹规划：路径-速度分解》。国际机器人研究杂志5，3 (1986), 72–89. [17] Mor Kaspi、Tal Raviv 和 Marlin W. Ulmer。2022。城市交通与城市物流未来研究方向。Networks 79, 3 (2022), 253–263. [18] Daniel Kornhauser、Gary L. Miller 和 Paul G. Spirakis。1984。图上的卵石移动、置换群直径及其应用。载于 FOCS。IEEE计算机学会，241–250. [19] Steven M. LaValle 和 Seth Hutchinson。1996。具有独立目标的多机器人的最优运动规划。载于 ICRA。IEEE，2847–2852. [20] Mark Lawley 和 Spyros Reveliotis。2001。顺序资源分配系统的死锁避免：难解与易解情形。《国际柔性制造系统杂志》13, 4 (2001), 385–404. [21] Rolf H. Möhring、Martin Skutella 和 Frederik Stork。2004。带与/或优先约束的调度。SIAM计算杂志 33, 2 (2004), 393–415. [22] Patrick A. O'Donnell 和 Tomás Lozano-Pérez。1989。两个机器人操作臂的无死锁、无碰撞协调。载于 ICRA。IEEE计算机学会，484–489.[23] Roberto Olmi、Cristian Secchi 和 Cesare Fantuzzi。2010。具有指定路径的多机器人的协调。IFAC会议论文集 43, 16 (2010), 312–317. [24] Spyros A. Reveliotis。2006。资源分配系统的实时管理：一种离散事件系统方法。第79卷。Springer科学与商业传媒。[25] Spyros A. Reveliotis。2015。协调自主性：用于自动化的顺序资源分配系统。IEEE机器人与自动化杂志 22, 2 (2015), 77–94. [26] Spyros A. Reveliotis。2020。关于某些基于导引路径的运输系统的状态活性及其计算复杂性。Autom. 113 (2020), 108777.[27] Spyros A. Reveliotis 和 Tomás Masopust。2019。关于开放式基于导引路径的交通系统状态活性的一些新结果。载于 MED。IEEE，398–404. [28] Spyros A. Reveliotis 和 Elzbieta Roszkowska。2010。多车辆交通系统中最大许可死锁避免的复杂性。IEEE自动控制汇刊 55, 7 (2010), 1646–1651. [29] Oren Salzman 和 Roni Stern。2020。多智能体路径寻找和多智能体取送问题中的研究挑战与机遇。载于 AAMAS。国际自主智能体与多智能体系统基金会，1711–1715. [30] Guni Sharon、Roni Stern、Ariel Felner 和 Nathan R. Sturtevant。2015。用于最优多智能体寻路的基于冲突的搜索。人工智能 219 (2015), 40–66. [31] Thierry Siméon、Stéphane Leroy 和 Jean-Paul Laumond。2002。多台移动机器人的路径协调：一种完备求解算法。IEEE机器人与自动化汇刊 18, 1 (2002), 42–49. [32] Israela Solomon 和 Dan Halperin。2018。多台单位球机器人运动规划 $\mathbb{R}^d$。载于机器人算法基础研讨会，WAFR。799–816.[33] Kiril Solovey 和 Dan Halperin。2016。无标签多机器人运动规划的困难性。国际机器人研究杂志 35, 14 (2016), 1750–1759. [34] Kiril Solovey、Jingjin Yu、Or Zamir 和 Dan Halperin。2015。具有最优性保证的无标签圆盘运动规划。载于机器人学：科学与系统。[35] Roni Stern、Nathan R. Sturtevant、Ariel Felner、Sven Koenig、Hang Ma、Thayne T. Walker、Jiaoyang Li、Dor Atzmon、Liron Cohen、T. K. Satish Kumar、Roman Barták 和 Eli Boyarski。2019。多智能体寻路：定义、变体与基准。载于 SOCS。AAAI出版社，151–159. [36] Hanlin Wang 和 Michael Rubenstein。2020。行走、停止、计数与交换：具有理论保证的去中心化多智能体路径搜索。IEEE机器人与自动化快报 5, 2 (2020), 1119–1126. [37] Jingjin Yu。2020。在连通性良好的环境中，平均情形下具有常数因子时间与距离最优性的多机器人路径规划。自主机器人 44, 3-4 (2020), 469–483.[38] Jingjin Yu 和 Daniela Rus。2014。带旋转的图上卵石移动：高效的可行性测试与规划算法。载于机器人算法基础研讨会，WAFR。729–746.

A 我们算法的缺失细节

A.1 使用侦察机器人求解环

令 r_j 为阻塞环 C中的一台侦察机器人。也就是说，在 r_j的环路径上存在一个顶点 p，该顶点不位于 C中任何其他机器人的环路径上。我们通过将 r_j移动到 p来解决 C，然后依次将 r_{j-1}，r_{j-2}、……、r_0，r_{h-1}，$r_{h-2}\ldots$，r_j移动到下一台机器人的起始顶点。伪代码见算法2，示例见图9。

算法2： 使用侦察机器人求解环 r_j

1 将 r_j移动到 p； 2 **对于** $i = j-1, j-2, \ldots, 0, h-1, h-2, \ldots, j$ **执行** 3 将 r_i移动到 $s(r_{i+1})$；

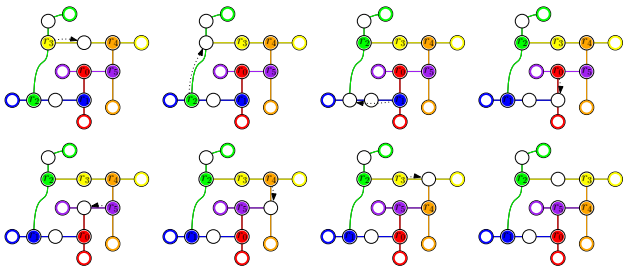

图9：含有侦察机器人的阻塞环 r_3，其求解方式如正文所述。初始状态位于左上方的配置中，右下方的状态则展示了已求解的环。首先，将 r_3移动到一个不出现在任何其他机器人路径上的顶点。然后依次移动所有其他机器人。

引理A.1。 设 C是一个包含侦察机器人的环r_j，也就是说，r_j的环路径上有一个顶点 p，该顶点不在 C中任何其他机器人的环路径上。则算法*2*可以求解 C。

证明。在移动任何机器人之前，每个机器人 r_i在其环路径上只有另一个机器人，即 r_{i+1}，这是由环的定义可知的。侦察机器人r_j移动到 p是有效的，因为r_j的环路径上唯一的机器人是r_{j+1}，它位于路径上p之后的最后一个顶点。在移动任何机器人 r_i之前，任意机器人 r_k，$i+1 \le k \le j-1$都已移动到 s（r_{k+1}）；由于顶点重数为 2，顶点s（r_{k+1}）最多包含在两个机器人的路径中，即 r_k，r_{k+1}，且不包含在 r_i的路径中，因此 r_k不会阻挡 r_i。侦察机器人不会阻挡 r_i，因为 p只包含在r_j的环路径中。任意机器人 r_k，$j+1 \le k \le i-1$都位于其初始位置，只阻挡 r_{k-1}，而不阻挡 r_i。因此，r_i移动到 s（r_{i+1}）不会受到任何机器人的阻挡。遍历所有机器人后，算法将每个机器人 r_i移动到 s（r_{i+1}），并根据定义完成该环。□

A.2 不可解基础情形的证明

引理 3.5。 如果由路径组成的图 G包含一个所有顶点均为起始顶点的简单环，那么 G不可解。

证明。考虑这样的环 $C = v_0, \ldots\ldots, v_{k-1}$。在对应的 MRMP-GP 实例中，$I(C)$，每个顶点 v_i上都有一个机器人 r_i。我们断言，对于所有 i，r_i的路径均以一条从 v_i到 v_{i+1}的边开始。否则，利用由路径组成的图的性质，很容易得出顶点 v_i至少出现在三条不同的路径上，这是不可能的。因此，r_i无法移动，因为它只能沿路径移动到 v_{i+1}，而该位置已被 r_{i+1}占据。换言之，这个环构成了一个死锁，因此不存在解。□

A.3 二维网格图上直线路径的高效实现

在求解更具体的 MRMP-GP-NBT(2) 变体时，机器人在二维网格上移动，且每个机器人的路径都包含在单独的一行或一列中；此时，我们实现的运行时间仅取决于机器人的数量。

定理 A.2。 *MRMP-GP-NBT(2,0)*，即W为二维网格且每个机器人的路径都包含在W的单独一行或一列中的情形，存在一种算法，可在时间 $O(n^2)$内找到一个解，或报告不存在解；其中 n为机器人的数量。

如果工作空间是这样的网格，则可以采用更紧凑的方式给出输入：用端点表示每条行/列路径，而不是列出路径上的所有顶点。在这种情况下，算法的运行时间仅取决于机器人的数量，而非路径长度。

为了高效实现阶段1，可以使用平面扫描在 $O(n\log n)$ 时间内计算每个机器人可见的 4 个（或更少）机器人，并能在 $O(1)$ 时间内判断某个机器人是否有一条畅通路径通往其目标。一旦算法移动一个机器人 r（不失一般性，设该机器人沿单行移动），我们就能在常数时间内更新与 r位于同一列的两个可见机器人的可见机器人集合。为了高效实现阶段2，我们可以在 $O(n^2)$ 时间内计算任意两个机器人之间的所有交点。计算出所有交点后，图构造以及对图的所有操作都可以在线性时间内完成。最坏情况下，总运行时间为 $O(n^2)$。

A.4 更多示例

本节中，我们再给出一些示例，以展示该算法的运行过程。

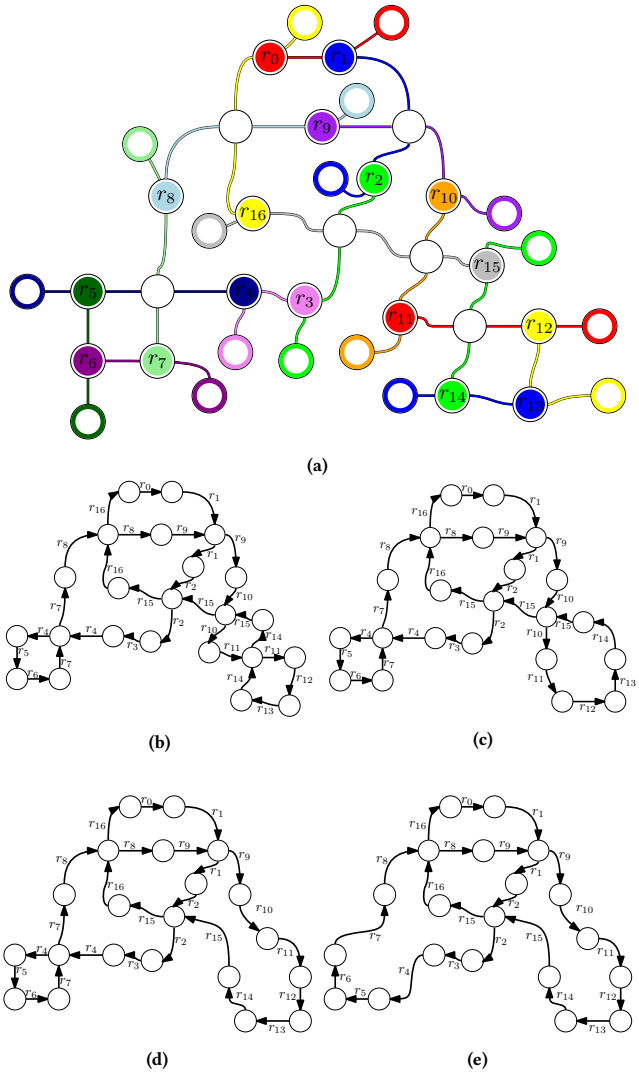

图10：(a) 一个稍复杂的 MRMP-GP-NBT(2) 实例，其中有一个阻塞环，需要进行几次解缠操作才能解决。(b) 对应的图 G。(c) 进行一次解缠操作后的 G，G_1。(d) 进行两次解缠操作后的 G，G_2。(e) 进行三次解缠操作后的 G；此后无法再进行此类操作，G_3。

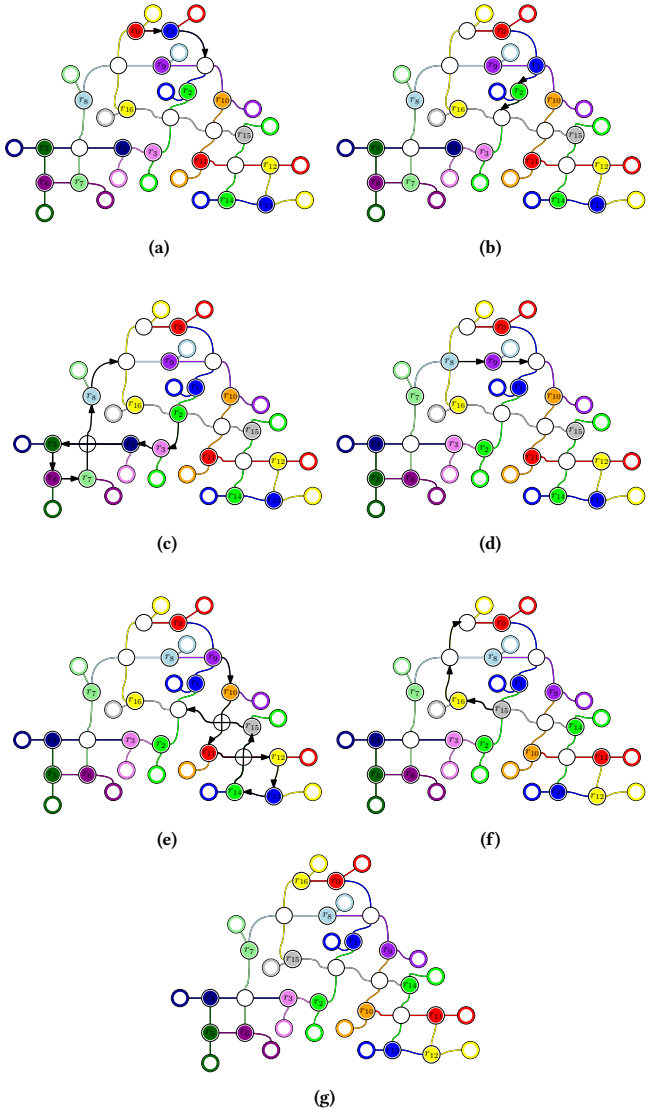

图11：如图10e所示，尽可能多地计算图10中环的 G_3 后，可以使用算法 1 计算运动。机器人块为 $B_1 = (r_0, r_1)$ $B_2 = (r_2)$ $B_3 = (r_3, r_4, r_5, r_6, r_7, r_8)$ $B_4 = (r_9)$ $B_5 = (r_{10}, r_{11}, r_{12}, r_{13}, r_{14}, r_{15})$ $B_6 = (r_{16})$。(a) 选取第一个块，$B = B_1$，并移动 head(B_1)。(b) 将 next(B_1) 中的机器人 B_2 从头部到尾部依次移动，然后再次移动 head(B_1)。(c) 将 next(B_2) 中的机器人 B_3 从头部到尾部依次移动，然后移动 head(B_2)。(d、e、f) 对其余块重复此过程。(g) 环已解决，所有机器人都拥有通往其目标的畅通路径。

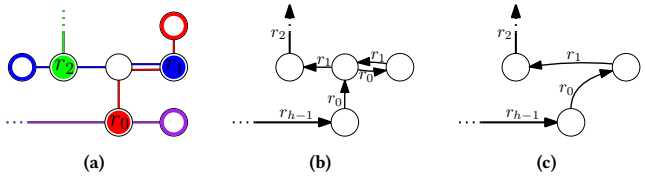

图12：展示算法如何处理包含必须沿相反方向遍历的边的环的示例。(a) 阻塞环的局部视图，其中两个机器人 r_0，r_1需要沿相反方向遍历同一条边。(b) 算法构造的相应图 G 的局部视图（假设环中没有侦察机器人）。(c) 解开后一个环之后的 G 局部视图，这强制施加了 r_1必须先于 r_0移动的约束。

B 定理4.1的更多证明细节

在本节中，我们继续第 n 4.

B.1 MRMP-GP-UNI(2,0)的困难性

我们通过将 MRMP-GP-PC 实例 M_s 转换为 MRMP-GP-UNI(2, 0) 实例 M，证明 MRMP-GP-UNI(2, 0) 的 NP困难性。具体做法是，为 M_s 中的每个构件 G 提供一个在网格上的对应实现，记为 M(G)，位于 M 中。M 与 M_s 包含相同的一组机器人，我们称之为（两个实例中的）主机器人。M 还包含次要机器人，它们最初位于已实现的构件中。请注意，对于这些机器人，我们会在不同构件中重复使用机器人名称（尽管这些机器人彼此不同），但同一构件中的名称始终唯一。完整实例 M 的示意图见图 16。请注意，每个构件都需要与穿过它的路径对齐，如各构件单独的图所示。

为了证明其困难性，我们将证明 M_s 可解，当且仅当 M 可解。为此，令 S_s 和 S 分别表示 M_s 和 M 的解。基于 S_s（分别基于 S），我们描述 M（分别是 M_s）的转换后的解 C(S_s)（分别是 C(S)）。

现在，我们指出实例 M_s 和 M 的一个共同性质，这有助于描述转换后的解。固定一条机器人的路径 π。在 M_s 和 M 中，每个构件 π 的正前方和正后方都有一个不会出现在其他路径上的顶点。我们称这样的顶点为 非阻塞顶点。通过在网格中插入一行或一列，并相应地“拉伸”路径，可以轻松创建这样的顶点。

在说明每个构件的实现之前，我们先概述如何在两个实例的解之间进行转换。转换后的解 C(S_s) 和 C(S) 都由 移动步骤组成；每一步之后，所有主要机器人都位于非阻塞顶点，这一不变量始终成立。每一步都只需让一个主要机器人 r 穿过一个构件并到达一个非阻塞顶点，即可维持这一不变量。由于该不变量，

r 在该步骤中不会与任何其他主机器人发生碰撞。因此，为了证明转换后的解有效，只需分别考察每个构件，具体如下。对于 C(S)，证明有效性只需验证每个构件是否按正确顺序被遍历。对于 C(S_s)，只需证明构件内部不会发生碰撞（因为其他地方不会发生碰撞）。我们注意到，很容易验证在任何机器人移动之前，不变量都成立。

转换后的解中的移动步骤按以下方式启动。对于 C(S_s)，每当机器人 r 离开S_s中的一个装置顶点 G 时，我们就启动一个移动步骤，在该步骤中，r沿 M穿过对应的构件 M(G)，到达最近的非阻塞顶点。作为该步骤的一部分，也可以移动次级机器人。对于 C(S)，由于构件并非单个顶点，我们定义遍历边，机器人遍历这些边即表示其已穿过该构件。更确切地说，当机器人 r 穿过构件 M(G)中的这类边时，我们就让 r 穿过 M_s 中对应的构件 G（同样到达 G 之后最近的非阻塞顶点）。在每个已实现的构件中，每个主机器人都有自己的遍历边，我们用锯齿线标记这些边。

现在我们可以介绍这些构件了。根据上述讨论，对于每种构件类型，我们首先给出 C(S_s) 中的移动步骤。然后，我们证明 C(S) 在 M_s 中产生正确的遍历顺序。在这一方向的转换中，当我们说机器人 r 在 M中遍历一个构件时，指的是 r 穿过其标记的遍历边。

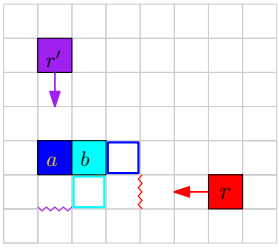

图13：网格上的优先约束装置实现。

优先约束装置。 设 G =P(r, r') 是一个优先约束装置。图13展示了 M(G)，即 G 在网格上的实现。该构件可以轻松旋转，以适配向左和向上延伸的路径。回想一下，在 S_s中，r必须先于 r'遍历 G。

在 C(S_s) 中，对应的步骤为：(1) r只需遍历M(G)；(2) b 和 a依次移动到各自的目标，随后由 r'遍历 M(G)。

对于另一个方向，注意到在解 S 中，b 必须先到达其目标，之后 r' 才能遍历 M(G)。因此，r'不能在 r 之前遍历 M(G)，否则 $\pi(r)$ 将在 $t(b)$ 处被 b 阻挡。因此，我们在 C(S) 中得到了 G 的一个有效遍历顺序。

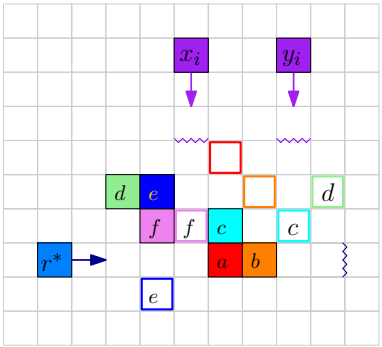

图14：在网格上实现 B_i。

前约束构件。图14展示了 $M(B_i)$，即 B_i 在网格上的实现。回想一下，在 M_s 中，x_i 或 y_i 会先于 r^* 遍历 B_i。我们假设 M_s 存在一个解 S_s，并分析两种情况，具体取决于 x_i 或 y_i 是否先于 r^* 遍历 B_i（两者都先遍历的情况较简单）。

情况1：假设在 S_s中穿过 B_i的遍历顺序为x_i, r, y_i。在 C(S_s) 中，相应的步骤为：(1) x_i先直接穿过 $M(B_i)$。(2) 随后，我们按顺序将以下机器人移至各自的目标：f, e, d, b。然后，将 c移动一条边（让 $t(c)$保持畅通，以便 y_i通过），并将 a移至其目标。这些移动清空了 r^*在 $M(B_i)$内的路径，因此接下来可以让它穿过 $M(B_i)$。(3) y_i穿过 $M(B_i)$。

情况2：另一方面，假设在 S_s中穿过 B_i的遍历顺序为 y_i, r^*, x_i。在 C(S_s) 中，相应的步骤为：(1) 在 y_i完成遍历后，(2) 我们将 c和 a依次移至各自的目标。然后，将 b向前移动一条边，这样r^*就可以穿过 $M(B_i)$。(3) x_i穿过 $M(B_i)$。

在所有情况下，在 r，x_i，y_i遍历 $M(B_i)$之后，任何属于 $M(B_i)$且尚未到达各自目标的次级机器人都可以轻松移动到目标位置。

反过来，假设 r^*在 x_i和 y_i之前遍历 $M(B_i)$，并由此导出矛盾。通过分类讨论不难验证，在 r^*遍历$M(B_i)$之前，b或 c必定会先到达其目标。在前一种情况下，进一步观察可知，f也必定会到达其目标（在 r^*遍历 $M(B_i)$之前）。由于 $t(f)$位于$\pi(x_i)$上，而 $t(c)$位于 $\pi(y_i)$上，后面提到的路径中必有一条会被阻塞。因此，x_i或 y_i将无法到达其目标，这就产生了矛盾。因此，对 S进行转换，可得到C(S)中 B_i的一个有效遍历顺序。

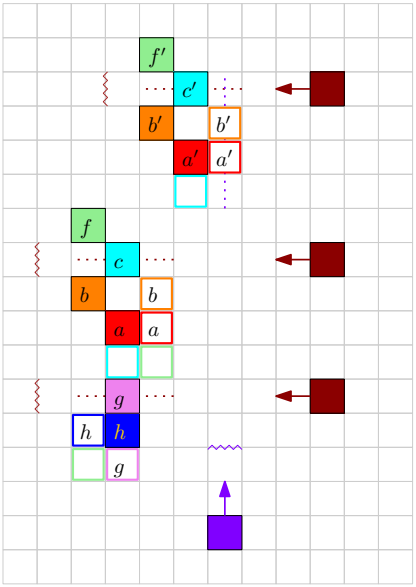

图15：在网格上实现 A_j。

后约束小工具。图15展示了 $M(A_j)$，即 A_j在网格上的实现。回想一下，在 M_s中，经过 A_j的检查机器人必须有一台在 β之后遍历 A_j。假设M_s 有一个解，其中只有一台检查机器人 r 会在 β之后遍历 A_j。我们这样假设，是因为多台机器人在 β之后遍历 A_j的情况更简单。注意，在下面的情形中，我们可以轻松调换最先遍历A_j的两台检查机器人的顺序，因此前两个移动步骤合并描述。

情况1：S_s中经过 A_j的遍历顺序为 c_j^1、c_j^2、β、c_j^3。在 C(S_s)中，对应的步骤如下：(1-2) 将c和 c'都移动一条边。这些移动为 c_j^2和 c_j^1经过$M(A_j)$清出路径，使它们能够遍历 $M(A_j)$。(3) 然后让 β遍历 $M(A_j)$。(4) 最后，为 c_j^3清出路径，之后 c_j^3遍历 $M(A_j)$；为此，按以下顺序将这些机器人移动到各自的目标位置：$a'\ c'\ b'\ f'\ a$、c、b、f、h、g。

情况2：在 S_s中穿过 A_j的遍历顺序为 $c_j^1, c_j^3, \beta, c_j^2$。在C($S_s$)中，我们采取以下相应步骤：(1-2) 将 c'和 b沿一条边移动。然后按指定顺序将以下机器人移至其目标：f, h, g。这些移动清空了 c_j^3和c_j^1穿过 $M(A_j)$的路径，使它们能够穿过 $M(A_j)$。(3) 然后我们让 β穿过 $M(A_j)$。(4) 最后，我们为 c_j^2清出路径，之后 c_j^2穿过 $M(A_j)$；为此，按以下顺序将机器人移至其目标：$a'\ c'\ b'\ f'\ a, b, c$。

情况3：在 S_s中穿过 A_j的遍历顺序为 $c_j^2, c_j^3, \beta, c_j^1$。在C($S_s$)中，我们采取以下相应步骤：(1-2) 将 b'沿一条边移动。然后按指定顺序将以下机器人移至其目标：$f'\ a, b, c, f, h, g$。这些移动清空了 c_j^3和 c_j^2穿过 $M(A_j)$的路径，使它们能够穿过 $M(A_j)$。(3) 然后我们让 β穿过 $M(A_j)$。(4) 最后，我们为 c_j^1清出路径，之后 c_j^1穿过 $M(A_j)$；为此，按指定顺序将 $a'b'\ c'$移至其目标。

反过来，为了导出矛盾，假设 A_j的所有检查机器人都在 β之前穿过 $M(A_j)$。我们考察检查机器人穿过 $M(A_j)$之前必须发生的移动。

引理 B.1. b'必须先移动一条边，然后 c_j^2和 c_j^3才能在 $M(A_j)$中穿过各自的遍历边。

证明。在 c_j^3穿过 $M(A_j)$之前，g必须移动一条边，因此 h也必须如此。在这些移动发生之前，f必须到达其目标，否则它会被 h堵在 $t(h)$处。因此，为了让 f到达其目标，b必须在 c_j^3穿过 $M(A_j)$之前移动一条边。显然，在 c_j^2穿过 $M(A_j)$之前，c也必须移动一条边。结合最后两点，通过简单的分类讨论可知，a或 b必须在 c_j^2和 c_j^3穿过 $M(A_j)$之前到达其目标。因此，更早之前，f'必须越过 $s(b')$，以免被 a或 b堵住。由此即可得到所需结果，因为 b'必须移动一条边，才能让 f'完成后一移动。□

现在注意到，c'必须先移动一条边，随后 c_j^1才能穿过$M(A_j)$。现在将这一事实与引理 B.1结合起来，可得（通过与上文针对 a和 b所作的类似分类讨论），a'或 b'必须在所有检查机器人穿过 $M(A_j)$之前到达其目标。这就产生了矛盾，因为这样一来，β经过 $M(A_j)$的路径就会被到达目标的 a',b机器人阻挡。因此，转换 S可得 A_j在C(S)中的一个有效遍历顺序。

通过观察到，在 M 的每个构件中，顶点重数均为 2，我们完成了从 M_s 到 M 的转换。由于

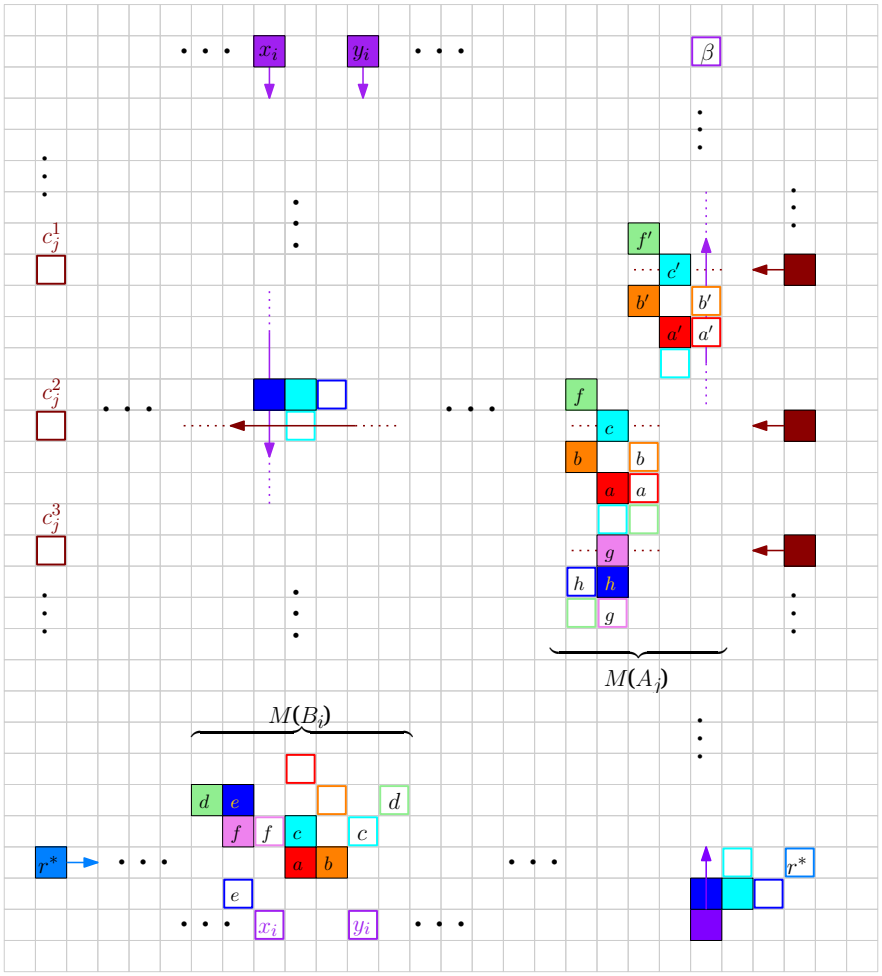

图 16： *M* **的示意图，展示了各构件的相对位置。彩色实心方块表示起点，空心方块表示目标。**

构件之外也同样如此，因此 M 确实是 MRMP-GP-UNI(2,0) 的一个实例，符合要求。最后，可以验证该归约可在多项式时间内完成。因此，我们得到以下结论。

引理 B.2。 *MRMP-GP-UNI(2,0)* 是 *NP* 完全的。

B.2 MRMP-GP-NBT(3) 的困难性

我们进行归约的下一阶段旨在证明 MRMP-GP-NBT(3) 的 NP-困难性，即将 M转换为 MRMP-GP-NBT(4,1) 的一个实例 M'，其中 W仍为二维网格。这一阶段需要进行以下大多为局部的修改：设 r, r'为两个机器人，其中 $t(r)$ 是位于顶点 v处的阻塞目标，且 $v \in \pi(r')$。我们延长 $\pi(r)$，使其沿着 $\pi(r')$ 反方向延伸，直到刚刚经过 $s(r')$。图 17 展示了这一主要修改，说明如何在示例 (a) 中执行该操作，从而得到示例 (b)。这一修改保持了机器人之间的优先约束，即关于带叉号标记顶点处的遍历顺序的优先约束。

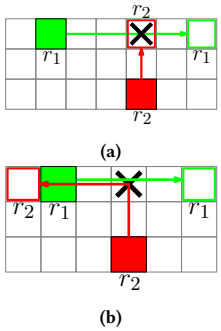

图 17：两个场景，其中 r_1 必须先于 r_2访问标有叉号的顶点。

证明的最后一步是将 M'转换为 MRMP-GP-NBT(3) 的一个实例 M''。具体做法是从 M'中移除带有 VM=4 的顶点。可以验证，在进行此更改的同时，仍能保持 M' 可解当且仅当 M'' 可解这一性质，正如所需。