文档跨度的天际线算子

Antoine Amarilli LTCI，巴黎电信，巴黎综合理工学院，法国
BennyKimelfeld 以色列理工学院，海法，以色列
Sébastien Labbé巴黎高等师范学院
Stefan Mengel阿尔图瓦大学，法国国家科学研究中心，朗斯计算机研究中心（CRIL）

arXiv:2304.06155v2 [cs.DB] 4 Mar 2024

摘要

从文本文件中抽取跨度（区间）关系时，一种常见做法是过滤掉关系中被认为受其他元组支配的元组。支配规则被定义为一种偏序，因不同系统和任务而异。例如，我们可以规定，若某个元组可通过添加属性或赋予更大的区间来扩展，则该元组受扩展后的元组支配。根据这一偏序过滤关系后，得到的结果便是天际线 。由于这种过滤可能会移除大多数抽取出的元组，我们研究能否通过将支配规则编译进抽取器来提升抽取性能。

为此，我们为以文档跨度器表示的声明式信息抽取任务引入天际线算子 。我们证明，当支配偏序本身可表示为正则跨度器时，该算子可通过正则运算来表示，这涵盖了若干自然的支配规则。然而，我们也证明，天际线算子会带来计算代价（按组合复杂度衡量）。首先，在某些情况下，要将跨度器表示为序列变量集自动机，该算子会使所需状态数呈指数级膨胀。其次，求值可能变得计算困难。我们的分析更准确地确定了哪些类别的支配规则具有可处理或难处理的组合复杂度。

2012 年 ACM 学科分类 信息系统 → 信息抽取；计算理论 → 数据库查询处理与优化（理论）

关键词与短语 信息抽取、文档跨度、查询求值

资助 Amarilli、Labbé 和 Mengel 获得了法国国家研究署项目 EQUUS ANR-19-CE48-0019 的部分资助。

1 引言

文档跨度器 框架 [10]是一种成熟的形式体系，用于表达声明式信息抽取任务。一个跨度器规定了如何为文本文档中的变量赋值，从而生成所谓的映射，它们是抽取的结果：每个映射都将变量指派给文档的一个因子，称为跨度。跨度器形式体系通过若干算子来定义，尤其包括扩展了捕获变量的正则运算（对应于所谓的正则跨度器）、关系代数中的算子（有时可转换为正则表达式）、字符串相等（即所谓的核心跨度器）等。

现有算子无法以简单的方式抽取依据某些标准达到最大的映射。然而，在一些自然情形下，某些映射会优先于其他映射。事实上，传统的信息抽取声明式系统提供了明确机制，可根据不同的比较方式，将抽取跨度限制为最大跨度。IBM 的 SystemT [18] 具有合并规则，例如“包含”（跨度支配其子跨度）和“从左到右”（跨度支配

所有起始位置相同且更短的跨度）。类似地，GATE 系统 [9] 提供了诸如“Appelt”这样的控制方式（与 SystemT 的“从左到右”类似）。另一种情形是，在无模式的文档跨度器背景下，我们可以将跨度仅赋给一部分变量 [19], 此时，我们可能只想捕获那些赋值涉及最大变量子集、且无法通过赋予更多变量而扩展的跨度；例如，这与关系代数中的全析取 [14] 或 `OPTIONAL` SPARQL 的算子 [2]的理念相近。

为了探究控制器和整合器等算子的表达能力，Fagin 等人 [11] 提出了一种框架，将先前的优先修复 [27]概念融入文档跨度器。他们在那里定义了“否定偏好生成依赖”（否定 pgd）的概念，利用底层跨度器语言表达二元支配关系。当这种关系具有传递性时，应用否定 pgd 的结果恰好是极大元组的集合。然而，他们并未讨论该算子的计算复杂度，因此这一问题仍未解决。（此外，他们的研究不适用于无模式语境。）

在数据库研究的其他领域中，极大匹配这一概念已得到广泛研究，在这些领域中它被称为天际线算子 [5]。直观而言，偏序关系下一个结果集的天际线，就是其中极大的结果所组成的集合，也就是说，这些结果不会被其他结果支配。人们从许多不同角度研究了天际线计算的复杂性，例如输入/输出访问 [26], 并行计算 [1], 或含噪比较 [17]。然而，据我们所知，目前尚无研究探讨该算子在提取文档跨度器的极大匹配时的复杂性。本文聚焦于此。

贡献。 我们将本文的贡献与文章结构一并介绍。在介绍一些必要的预备知识（第 2 节）之后，我们首先在第 3 节介绍天际线算子。该算子定义为：根据映射上的偏序，从文档跨度器在某个文档上的映射中提取极大映射；我们将此偏序称为支配关系。具体而言，我们定义了跨距包含、跨距长度、变量包含以及从左到右支配关系，这些关系涵盖了前文介绍的示例。

为了能够统一研究这些算子，并且类似于 [11], ，我们提出一种通用模型，其中支配关系本身也表示为文档跨度器。更准确地说，支配规则是一个跨度器，它在每个文档上定义一种支配关系：它指出哪些映射支配其他映射，直观上通过捕捉成对的映射$(m,\ m')$ 来表示 m' 支配 m。我们还重点研究所谓的逐变量规则，其中映射上的支配关系可以定义为跨度上的关系的直积。换言之，逐变量规则是一种跨度器，用来表示哪些跨度支配哪些跨度，而映射上的支配关系则通过逐变量逐点计算得出，这类似于人工智能中偏好处理所采用的其他条件不变语义 [6], 也类似于多维数据天际线查询中的帕累托最优点 [17]。前文介绍的所有示例都可以用这种逐变量方式表达。

接下来，我们开始研究如何在文档跨度器上计算天际线算子，并在第 4 节从 表达能力 问题入手：该算子是否会严格增强跨度器形式体系的表达能力，还是可以用现有算子重写? 我们证明，正则跨度器 在天际线算子下封闭，并将 [11]的结果推广到了无模式语境。相比之下，我们证明，核心跨度器 在天际线运算下并不封闭，即使采用固定的变量包含或跨度包含支配关系也是如此；我们再次将 [11]的结果推广到了这一情形。

接下来，我们探讨是否可以将天际线算子改写为正则跨度器，以实现高效求值，例如，多项式时间的

基于模式的上下文中的连接算子的编译（见 [20], 引理4.4.7）。我们在第 5 节中提出了一个下界，证明情况并非如此：即使对于变量包含支配，将天际线算子应用于表示为顺序变量集自动机（变量自动机）的跨度器，也必然会产生指数级膨胀。该结果通过确定变量自动机与非确定性读一次分支程序（非确定性读一次分支程序）之间的联系而得到。这种通用方法可以应用于天际线之外的场景；事实上，我们还利用它证明了一个具有独立意义的结果：存在一些正则表达式公式，对这些公式而言，自然连接算子会产生不可避免的指数级膨胀（定理 5.5）。

接着，在第6节中，我们从状态复杂度转向正则跨度器天际线求值的计算复杂度。显然，该任务在数据复杂度意义下是可处理的，即对于固定的跨度器和支配规则，我们只需计算所有捕获映射，并筛除非极大映射。更有意思的是，假设P $\neq$NP，我们证明该任务在组合复杂度意义下是难处理的，即将输入跨度器视为变量时也是如此（定理6.3）；即使在变量包含关系的情形下也是如此。因此，即便不将天际线算子编译为显式的变量自动机，我们也无法在组合复杂度意义下对其进行可处理的求值。

最后，我们更详细地研究天际线计算的复杂性如何取决于固定的支配关系：是否存在一些非平凡的支配规则，使得天际线计算在组合复杂度意义下是可处理的？我们在第7 节给出支配规则的一个充分条件；我们提到的所有示例规则都满足该条件，且该条件意味着（条件性的）难处理性（定理7.5）。随后，我们证明，对于一类称为类变量包含规则的支配规则，该条件的一个变体可用于进行二分分类，从而判定其中哪些规则能够实现可处理的天际线计算（定理7.7）。最后，我们给出一般情形下可处理和难处理规则的示例。

我们在第 8节总结。由于篇幅所限，大多数证明推迟至附录。

2 预备知识

语言、跨度、映射和跨度器。 我们固定一个 字母表 Σ，它是一个由字母构成的有限集合。一个词 w 是由 Σ 中的字母组成的有限序列：我们用 Σ^* 表示所有词构成的集合。我们用 $|w|$ 表示 w 的长度，并用 ε 表示空词，其中 $|\varepsilon| = 0$。一个 语言 $L \subseteq \Sigma^*$ 是一个词的集合。两个语言 L_1 和 L_2 的连接 是语言 $L_1 \cdot L_2 = \{w_1 w_2 \mid w_1 \in L_1, w_2 \in L_2\}$。语言 L 的克莱尼星号 是语言 $L^* = \bigcup_{i \in \mathbb{N}} L^i$，其中我们递归定义 $L^0 = \{\varepsilon\}$，并且对所有 $i > 0$，有 $L_{i+1} = L \cdot L_i$。与文档跨度器领域中的通常做法一样，一个文档 就是 Σ^* 中的一个词。

跨度 $[i, j\rangle$ 是一个区间 $s = [i, j\rangle$，满足 $0 \leqslant i \leqslant j$。其长度为 $j - i$。我们用 Spans 表示所有跨度的集合。文档 d 的跨度是 Spans 中满足$j \leqslant |d|$的跨度 $[i, j\rangle$。我们用 $d_{[i,j\rangle}$ 表示 d 在跨度 $[i, j\rangle$处的连续子词，例如，“qwertyqwerty”$_{[2,5\rangle}$ = “qwertyqwerty”$_{[8,11\rangle}$ = “ert”。注意，对于所有$0 \leqslant i \leqslant |d|$，我们有$d_{[i,i\rangle} = \varepsilon$。如果 $i' \leqslant i$ 且 $j' \geqslant j$，则称跨度 $[i, j\rangle$包含于跨度 $[i'$, $j'\rangle$。如果两个跨度存在一个同时包含于二者的非空跨度，则称它们重叠；否则称它们不相交。

我们固定一个由变量名组成的无限集合 Variables。文档 $d \in \Sigma^*$ 的映射 m 是一个从变量的有限集合 $X \subseteq$Variables 到 d的跨度集合的函数，称为 m的定义域 dom(m)；dom(m) 中的变量称为由 m赋值。我们用 Maps 表示 Variables 中变量上的所有映射的集合。若对于所有 $x \in \text{dom}(m) \cap \text{dom}(m')$，都有 $m(x) = m'(x)$，则称映射 m 与映射 m' 相容，记作 $m \sim m'$。

一个跨度器是一个函数，它将每个文档 d 映射到由映射组成的有限集合，这些映射的跨度位于 d上，即包含于 $[0,\ |d|\rangle$中。对于跨度器 P，我们将至少出现在其某个映射定义域中的变量记为 SVars(P)，形式化地，SVars$(P) := \{x \in$

变量 | $\exists d \in \Sigma^*, \exists m \in P(d), x \in \mathsf{dom}(m)\}$。如果跨度器 P的所有输出映射都恰好为 $\mathsf{SVars}(P)$中的变量赋值，则称其为基于模式的，即对于每个 $d \in \Sigma^*$ 和 $m \in P(d)$，都有 $\mathsf{dom}(m) = \mathsf{SVars}(P)$。否则，将 P称为无模式的 [22], 或不完备的 [19]。如果跨度器 P在文档 $d \in \Sigma^*$ 上接受或捕获映射 $m \in \mathsf{Maps}$，即 $m \in P(d)$，则称其具有该性质。

变量集自动机。我们主要关注正则跨度器，它们可以使用变量集自动机（或变量自动机）来表示。直观地说，这些是非确定性自动机，其中每个转移要么标记为一个字母，要么标记为一个标记，指示哪个变量被打开或闭合。形式化地，对于变量集合 X，我们将标记 (X) 定义为 X上的标记集合：对于每个变量 $x \in X$，它包含打开标记 $x\vdash$ 和闭合标记 $\dashv x$。然后，在字母表 Σ 上的变量自动机是一个自动机$\mathsf{A} = (Q, q_0, F, \delta)$，其中 Q 是状态的有限集合，$q_0 \in Q$ 是初始状态，$F \subseteq Q$ 是终态集合，而 $\delta \subseteq Q \times (\Sigma \cup \text{标记}(X)) \times Q$ 是转移关系：我们写转移 $q \to^{\sigma} q'$ 来表示 $(q, \sigma, q') \in \delta$。注意，转移同时包含字母转移（其标记为 Σ中的字母）和标记转换（其标记为标记(X)中的标记）。

A 在文档 $d \in \Sigma^*$ 上的一次运行是一个序列 ρ：$q_0 \to^{\sigma_1} q_1 \cdots q_{n-1} \to^{\sigma_n} q_n$，其中 $\sigma_1 \ldots \sigma_n$ 限制在 Σ 的字母上后恰好为 d；若满足 $q_n \in F$，则称其为接受运行。若对于每个变量 $x \in X$，标记 $x\vdash$ 和 $\dashv x$ 要么不出现在 $\sigma_1 \cdots \sigma_n$中，要么恰好各出现一次，且 $x\vdash$ 出现在 $\dashv x$之前，则称 ρ 有效。若 A 的所有接受运行均有效，则称 A 为顺序的。本文始终假设变量自动机是顺序的，并且提到变量自动机时，均指顺序变量自动机。于是，运行 ρ 定义了一个从 d 到 m 的映射，其直观含义是：将读到标记的变量赋值为由这些标记界定的跨度。形式上，我们将运行中的每个索引 $0 \leqslant k \leqslant n$ 与 d 中的一个位置 $\pi(k)$ 相关联：初始化为 $\pi(0) := 0$；若转移 $q_k \to^{\sigma_{k+1}} q_{k+1}$ 读取一个标记，则令 $\pi(k+1) := \pi(k)$；若读取一个字母，则令 $\pi(k+1) := \pi(k) + 1$；注意，$\pi(n) = |d|$。然后，对于其标记在 ρ中被读取的每个变量 x，令 $\sigma_i = x\vdash$ 和 $\sigma_j = \dashv x$，其中 $i < j$（因为该运行有效），并令 $m(x) := [\pi(i), \pi(j)\rangle$。

因此，顺序变量自动机 $\mathcal{A}$ 定义了一个跨度器 $P_{\mathcal{A}}$，它将每个文档 d 映射到一个集合 $P_{\mathcal{A}}(d)$，其中包含如前所述由其接受运行得到的映射。请注意，不同的接受运行可能产生相同的映射。我们有时会滥用记法，将变量自动机与其定义的跨度器视为同一对象。正则跨度器 是指可以由变量自动机定义的跨度器，或者等价地说，可以由顺序变量自动机定义的跨度器。如果一个顺序变量自动机定义了一个基于模式的跨度器，即每个映射都为该变量自动机转移中出现的每个变量赋值，那么这个顺序变量自动机就是函数式的 。

正则表达式公式。本文中的跨度器示例不会以变量自动机的形式给出，而是采用更易于阅读的正则表达式公式形式。字母表 Σ上的正则表达式公式 是由空集 $\varnothing$、空词 ε和单个字母 $a \in \Sigma$通过三种正则运算符——析取（$e_1 \vee e_2$）、连接（$e_1 e_2$）和克莱尼星号（e^*）——以及形式为 $x\{e_1\}$ 的变量捕获 归纳定义的表达式，其中 x 是一个变量。文档 $d \in \Sigma^*$ 上的正则表达式公式 r 定义了一个作用于 r中出现的变量的跨度器。直观地说，d 上 r 的每个匹配都会产生一个映射，其中的变量按照捕获被赋值为良嵌套跨度；详情请见 [10] 。我们要求正则表达式公式在每个文档 $d \in \Sigma^*$上对每个变量至多赋值一次；但允许它们定义无模式跨度器，也就是说，它们可能只为变量的一个子集赋值。

众所周知，正则表达式公式捕获了正则跨度器的严格子集；对于基于模式的跨度器，请参见 [10] ，对于无模式张量，请参见 [19] 。

笛卡尔积。给定两个跨度器 P_1 和 P_2，其中 $X_1 =$ 变量(P_1) 和 $X_2 =$变量(P_2) 不相交，笛卡尔积 $P_1 \times P_2$ 是定义在变量$X_1 \cup X_2$ 上的跨度器，对于每个文档 d，它捕获映射 $(P_1 \times P_2)(d) := P_1(d) \times P_2(d)$。在这里，我们将一对 $(m_1, m_2) \in P_1(d) \times P_2(d)$ 解释为两个映射的合并，即在 X_1 上按照 m_1 定义、在 X_2上按照 m_2 定义的映射。如果给定 P_1 和 P_2 作为顺序变量自动机，则可以在多项式时间内计算出一个用于 $P_1 \times P_2$的顺序变量自动机。

3 天际线算子

本文定义并研究一种称为天际线算子的新算子。它旨在仅提取在某种意义上包含最多信息的映射。

支配关系。我们首先定义支配关系，用于描述如何比较给定文档 d 上两个映射所提供的信息。

▸ **定义 3.1**。对于文档 d，预支配关系 $\preccurlyeq$ 是 d 的映射集合映射集合上的一种关系。若该关系是（非严格）偏序，即自反、传递且反对称，则称其为支配关系。对于 m_1、$m_2 \in$ 映射集合，若 $m_1 \preccurlyeq m_2$，则称 m_2支配 m_1，否则记作 $m_1 \not\preccurlyeq m_2$。

支配关系旨在定义哪些映射优于其他映射，直观而言，是因为它们包含更多信息；这种关系可能取决于文档，不过我们将给出许多不取决于文档的例子。

我们介绍了若干支配关系，正如引言中所讨论的，它们是实际系统的一部分，也是我们在本文中始终考察的内容：

▸ **例 3.2.** 最简单的关系是平凡的自支配关系 $\preccurlyeq_{self}$，其中每个映射都只支配自身，即该关系中的配对为 (m, m) 对于 $m \in$ 映射集合。

▸ **示例 3.3.** 变量包含关系 $\preccurlyeq_{varInc}$ 包含对 (m_1, m_2) 使得对于所有 $x \in$ 变量，若 $m_1(x)$ 有定义，则 $m_2(x)$ 也有定义，并且 $m_1(x) = m_2(x)$。直观而言，当 $\mathsf{dom}(m_1) \subseteq \mathsf{dom}(m_2)$ 且 $m_1 \sim m_2$时，我们有 $m_1 \preccurlyeq_{varInc} m_2$，也就是说，当$m_2$ 是 m_1 的扩展，且可能赋予比 m_1更多变量时。

▸ **示例 3.4.** 跨距包含关系 $\preccurlyeq_{spanInc}$ 包含具有相同定义域（$\mathsf{dom}(m_1) = \mathsf{dom}(m_2)$）的映射对 (m_1, m_2)，且对于每个 $x \in \mathsf{dom}(m_1)$，跨度$m_1(x)$ 都包含于 $m_2(x)$。直观而言，m_1 和 m_2 在文档的相同部分匹配相同的变量，但 m_1 中变量的匹配结果是 m_2中对应匹配结果的子词。

▸ **例 3.5.** 从左到右关系 $\preccurlyeq_{ltr}$ 包含具有相同定义域的映射对 (m_1, m_2)；对于每个 x 上 m_1 和 m_2 都有定义的变量，跨度$m_1(x)$ 和 $m_2(x)$ 的起始位置相同，但 $m_2(x)$ 的长度不小于 $m_1(x)$。

▸ **例 3.6.** 跨度长度关系 $\preccurlyeq_{spanLen}$ 包含具有相同定义域的映射对 (m_1, m_2)，其中对于每个 $x \in$, $\mathsf{dom}(m_1)$ 跨度 $m_2(x)$ 的长度都不小于 $m_1(x)$。直观而言，$\preccurlyeq_{spanLen}$ 在文档中的任何位置都优先选择较长的跨度，而非较短的跨度。

支配规则。现在我们介绍支配规则，它为每个文档 d关联一个关于 d的支配关系。本文将支配规则表示为特定定义域上的跨度器。为此，给定变量集合 X，我们记 $X^\dagger$ 表示变量 X的带注释副本集合，形式化地，$X^\dagger := \{x^\dagger \mid x \in X\}$。我们通过如下方式将该记法扩展到映射：对于映射 m，定义 $m^\dagger$ 为一个映射，其定义域 $\mathsf{dom}(m^\dagger) =\mathsf{dom}(m)^\dagger$，并且对于所有 $x \in \mathsf{dom}(m)$，都有 $m^\dagger(x^\dagger) := m(x)$。接着我们定义：

▸ **定义 3.7**。 变量集合 $X \subseteq$变量上的预支配规则 D 是一个（无模式）跨度器，其 $\mathsf{SVars}(D) \subseteq X \cup X^{\dagger}$。对于每个文档 $d \in \Sigma^*$，我们将 $D(d)$ 视为定义在 d 上的预支配关系 $\preccurlyeq$，该关系由 D 在 d上捕获的映射定义；可比较对的左侧和右侧分别是这些映射在 X 和 $X^{\dagger}$ 上的限制。形式上，关系 $\preccurlyeq$ 为： $R := \{(m|_X, m') \mid m \in D(d), (m')^{\dagger} = m|_{X^{\dagger}}\}$。

我们称 D 为支配规则，如果对于每个文档 $d \in \Sigma^*$，上述定义的预支配关系 R 都是支配关系，即它正确地定义了一个偏序。

直观地说，对于每个文档 d，支配规则 D 定义了支配关系 $\preccurlyeq$，其中每个映射 $m \in D(d)$ 表示一对映射；也就是说，将 m 限制在 X 上的结果被将 m 限制在 $X^{\dagger}$ 上的结果支配（将变量从 $X^{\dagger}$ 重命名为 X）。注意，预支配规则只是一种过渡性概念；下文中我们只考虑支配规则。

▸ **例 3.8**。对于任意变量集合 X，例3.2–3.5 中引入的每个支配关系，都可以通过一个定义在 X上的正则跨度器所表达的支配规则来定义（对于例3.6中的跨距长度支配关系，参见引理3.13）。本节末尾，我们将说明如何以更简洁的逐变量方式表达它们，且不依赖于 X。

天际线算子。 我们已将支配规则引入为一种在任意文档上定义支配关系的方式。现在，我们可以引入天际线算子 来提取极大映射，即在支配关系中不受支配的映射：

▸**定义 3.9.** 给定一个支配规则 D，天际线算子 η_D 作用于跨度器 P，并按以下方式定义一个跨度器 $\eta_D P$：给定一个文档 d，记$\preccurlyeq$ 为由 D 在 d上给出的支配关系 $D(d)$，则 $\eta_D P$ 作用于 d 的结果是 $P(d)$ 在支配关系 $\preccurlyeq$下的极大映射集合。形式化地，我们有：$(\eta_D P)(d) := \{m \in P(d) \mid \forall m' \in P(d) \backslash \{m\}: \quad m \not\preccurlyeq m'\}$.

直观而言，该算子根据支配规则在输入文档上定义的支配关系，筛除被其他映射支配的映射。

▸ **例 3.10**。 在图 1 中，我们展示了天际线算子在若干示例支配关系下的效果。假设给定一个变量集为 P 的跨度器$\{x, y\}$ 它在给定文档 d 上提取出图 1a 中给出的映射（此处短横线“–”表示该变量未被某个映射赋值）。我们展示了应用天际线算子的结果，这些算子分别采用定义变量包含支配关系的支配规则（可能不是正则的） $\preccurlyeq_{varInc}$ （图 1b）、定义跨度包含支配关系的支配规则 $\preccurlyeq_{spanInc}$（图 1c）以及定义跨度长度支配关系的支配规则 $\preccurlyeq_{spanLen}$ （图 1d）。请注意，对于变量包含支配规则，天际线仅对无模式跨度器有意义，因为两个赋值给相同变量的不同映射始终不可比较。

逐变量规则。我们定义的天际线算子是基于以显式变量集上的跨度器形式表示的支配规则。不过，通常更方便的做法是应用乘积算子，将规则定义为单个变量上的规则之乘积。这能确保规则具有“对称性”，即所有变量的行为都相同：

▸**定义 3.11.** 设 D 是单个变量 x 上的一条支配规则，即一个使用变量 $\{x, x^{\dagger}\}$ 的跨度器。对于 $y \in$ 变量，我们令 D^y 为将 x 和 $x^{\dagger}$ 分别替换为 y 和 $y^{\dagger}$ 后得到的支配规则，也就是说，对于每个文档 d，映射集合 $D^y(d)$ 由一个

$m(x)$	$m(y)$
$[1,2\rangle$	$[2,3\rangle$
–	$[2,3\rangle$
$[0,2\rangle$	$[2,3\rangle$
$[4,6\rangle$	$[4,10\rangle$

(a) 提取出的映射 $P(d)$。

$m(x)$	$m(y)$
$[1,2\rangle$	$[2,3\rangle$
$[0,2\rangle$	$[2,3\rangle$
$[4,6\rangle$	$[4,10\rangle$

(b) 变量包含关系下的天际线。

$m(x)$	$m(y)$
–	$[2,3\rangle$
$[0,2\rangle$	$[2,3\rangle$
$[4,6\rangle$	$[4,10\rangle$

(c) 跨度包含关系下的天际线。

$m(x)$	$m(y)$
–	$[2,3\rangle$
$[4,6\rangle$	$[4,10\rangle$

(d) 跨度长度关系下的天际线。

图 1 应用不同天际线算子前后的提取映射；见例3.10。

对于每个映射 m^y，满足 $m^y(y)$ 且 $m^y(y^\dagger)$ 的映射 $m \in D^y(d)$，定义方式如 $m(x)$ 和 $m(x^\dagger)$。

由 D 在变量集 X 上定义的逐变量支配规则就是 $\times_{y\in X} D^y$。如果一个支配规则可以用这种方式表示，就称它为逐变量支配规则。

我们通常会省略变量集 X，并且有时会滥用记号，将单个变量上的支配规则视为它们在任意变量集上所定义的逐变量支配规则。

▸ **例 3.12**。 自支配规则（例 3.2）是逐变量的，因为它可以由以下定义在单个变量上的平凡支配规则得到：

$$D_{self} = \Sigma^* x^\dagger \{x\{\Sigma^*\}\}\Sigma^* \vee \Sigma^*.$$

上面的 Σ^* 项用于确保自反性，并表示在未分配 x 的映射与未分配 $x^\dagger$ 的映射之间的空支配关系。

跨度包含支配规则、从左到右支配规则和变量包含支配规则（示例3.3–3.5）也具有逐变量性质，其单变量规则如下：

$$D_{spanInc} = \Sigma^* x^\dagger \{\Sigma^* x\{\Sigma^*\}\Sigma^*\}\Sigma^* \vee \Sigma^*.$$
$$D_{ltr} = \Sigma^* x^\dagger \{x\{\Sigma^*\}\Sigma^*\}\Sigma^* \vee \Sigma^*.$$
$$D_{varInc} = \Sigma^* x^\dagger \{\Sigma^*\}\Sigma^* \vee D_{self}.$$

这里，$\Sigma^* x^\dagger \{\Sigma^*\}\Sigma^*$ 表示分配变量优于不分配变量。

至于变量长度支配规则（示例 3.6），它同样是逐变量的，但一个标准的抽引论证表明，它无法由正则跨度器定义：

▸**引理 3.13.** 单变量跨度长度支配规则 $D_{spanLen}$ 无法表示为正则跨度器。

4 天际线算子下的闭包

我们已定义了相对于由正则跨度器表达的支配规则的天际线算子。于是，一个自然的问题是：此类规则下的天际线算子是否扩展了跨度器形式体系的表达能力，或者是否可以在现有模型中定义。我们将在本节对此进行研究。

正则跨度器。 我们首先关注正则跨度器，并证明它们对于表示为正则跨度器的支配规则在天际线算子下封闭。为此，我们将展示如何使用正则跨度器在其下封闭的运算来表示天际线算子，即连接、交集和差集（定义见附录A）。

▸**定理 4.1**。 存在一种算法，给定一个定义正则跨度器的顺序变量自动机P 和一个定义支配规则的顺序变量自动机 D，可计算出一个用于 $\eta_D P$。

定理 4.1 推广了 Fagin 等人关于传递“否定 pgd”的一项结果。 [11, 定理 5。3] 用我们的术语来说，他们的定理指出，由正则支配规则定义的、对极大答案的限制下，完备正则跨度器类是封闭的。因此，定理 4.1将他们的结果扩展到了无模式正则跨度器。

定理 4.1 意味着，相对于正则支配规则取天际线并不会增加正则跨度器的表达能力。然而，与 [11], 的结果类似，我们的构造可能会计算出规模远大于输入变量自动机的变量自动机，甚至呈指数级增长。在第 5 节中，我们将看到，对于任何表示天际线的顺序变量自动机而言，这种增长都不可避免。

作为定理 4.1 的一个应用，我们特别得出：对于前文给出的大多数示例，即示例 3.2–3.5，正则跨度器在天际线算子下封闭。

▸ **推论 4.2**。 存在一些算法，能够给定一个顺序变量自动机 P，分别计算出 $\eta_{self} P$、$\eta_{varInc} P$、 $\eta_{ltr} P$和 $\eta_{spanInc} P$的顺序变量自动机。

相比之下，定理 4.1 不适用于跨度长度支配关系，因为该关系无法表示为正则跨度器（引理 3.13 ）。事实上，我们可以证明，在此支配关系下取天际线通常无法表示为正则跨度器：

▸**命题 4.3**。 存在一个顺序变量自动机 P，使得 $\eta_{spanLen} P$ 不是正则的。

其他跨度器形式体系。 自然会有人问，定理 4.1中的闭包结果是否也适用于其他跨度器形式体系。特别是，我们可以考察核心跨度器的语言；它通过加入字符串相等约束扩展了正则跨度器；精确定义见 [10] ，无模式情形见 [25] 。我们可以证明，与正则跨度器不同，核心跨度器在天际线算子下不封闭：

▸**定理 4.4.** 关于跨度包含支配关系，核心跨度器在天际线算子下不封闭 $\preccurlyeq_{spanInc}$，即使对于基于模式的跨度器也是如此：存在一个基于模式的核心跨度器 P 使得 $\eta_{spanInc} P$ 无法表示为核心跨度器。对于采用变量包含支配规则的天际线 η_{varInc} ，结论也相同。

这一结果已在 [11] 中针对跨度包含支配关系得到证明，但该结果仅证明了其无法表示为基于模式的核心跨度器。我们的结果将这一结论推广到了无模式的情形，并且也证明了采用变量包含支配规则时的这一结论。形式化定义和证明见附录C.4 。

我们将采用天际线算子扩展其他形式体系的问题留待今后研究，例如，广义核心跨度器 通过差分算子扩展核心跨度器 [23], ，或 上下文无关跨度器 [21] ，后者通过上下文无关文法定义跨度器。请注意，与此相反，在 *RGXlog* 程序这一形式体系中，封闭性显然成立，因为其中的跨度器是用 Datalog 规则定义的 [23]。事实上，这一类恰好由多项式时间跨度器（按数据复杂度计）组成。因此，对于任何一种支配规则 D ，只要其极大答案可在数据复杂度下用多项式时间计算（特别是对于表示为正则跨度器的支配规则），那么，在 RGXlog 程序上对 D 应用天际线算子的结果就可以表示为一个 RGXlog 程序。

在本文其余部分，我们重点研究将天际线算子应用于正则跨度器，并通过正则支配规则定义支配关系。

5 天际线算子的状态复杂度

我们已经看到，从可以使用正则运算来表达这一点来看，天际线算子不会增加正则跨度器的表达能力。然而，这并未考虑这种转换所付出的代价。本节将证明，应用天际线算子时，顺序变量自动机的大小通常会呈指数级增长。具体而言，对于变量包含支配关系，我们将证明以下下界：

▸ **定理 5.1**。 对于每个 $n \in \mathbb{N}$，都存在一个具有 $O(n)$ 个状态的顺序变量自动机 A，使得令 $P_{\mathcal{A}}$ 为它定义的正则跨度器后，任何表示正则跨度器 $\eta_{varInc} P_{\mathcal{A}}$ 的顺序变量自动机都必须具有 $2^{\Omega(n)}$ 个状态。

我们将在后续章节中展示，如何用计算复杂度下界来补充天际线运算的状态复杂度下界。

证明技巧：将布尔函数表示为变量自动机。我们将按照下文所述，利用顺序变量自动机对布尔函数的表示来证明定理 5.1。设 SVars ⊆Variables 是一个有限变量集合（将用于定义跨度器），并设 $\text{Vars}_b := \{x_b \mid x \in \text{SVars}\}$ 是一个布尔变量集合。对于每个将跨度赋给 SVars 中部分变量的映射 m （即 $\text{dom}(m) \subseteq \text{SVars}$），我们定义一个布尔赋值$m_b$: $\text{Vars}_b \to \{0,1\}$，规定 $m_b(x_b) := 1$ 当且仅当 $x \in \text{dom}(m)$，也就是说， x 通过 m被赋予一个跨度。设 P 是一个变量集合为 SVars 的文档跨度器， d 是一个输入文档。那么，我们用 $\text{Bool}(P, d)$ 表示其模型为 $\{m_b \mid m \in P(d)\}$的布尔函数。

我们的直观想法是，如果函数 $\text{Bool}(P, d)$ 难以表示，那么跨度器 P 也应当如此。为了明确这一点，我们先介绍本文所使用的布尔函数表示：

▸ 定义 *5.2.* 变量集 Vars_b 上的非确定性读一次分支程序[1] *(NROBP)*是一个元组 $\Pi = (G, s, t, \mu)$，其中 $G = (V, E)$ 是一个有向无环图，$s \in V$ 和 $t \in V$ 分别是源节点和汇节点，而函数 μ 为部分边标记变量集 Vars_b中变量的文字，即变量及其否定；形式上， μ 是一个从 E 到 Vars_b上的文字的部分函数。我们要求，对于每条源汇路径$s = v_0, \ldots, v_n = t$，路径上各边标记的文字中，每个变量至多出现一次；也就是说，不存在两个下标 $0 \leqslant i < j \leqslant n-1$，使得 $\mu((v_i, v_{i+1}))$ 均有定义，且分别映射到同一变量的文字。

非确定性只读分支程序 Π 计算定义在变量$_b$ 上的布尔函数，其模型定义如下。赋值 m_b： 变量$_b \to \{0,1\}$ 是 Π 的一个模型，当且仅当 G 中存在一条源汇路径，使得该路径上的所有文字标签都被 m_b满足，即存在一个序列$s = v_0, \ldots, v_n = t$，使得对于每个 $0 \leqslant i < n$，如果 $\ell := \mu((v_i, v_{i+1}))$ 被定义，则文字 ℓ 根据 m_b的规定求值为真。

直观上，NROBP 与自动机类似。为形式化这种联系，我们展示如何根据顺序变量自动机和文档，高效计算一个描述捕获映射中哪些变量子集可以被赋值的 NROBP：

▸**引理 5.3**。 设 P 是定义在变量集 SVars 上的一个正则跨度器，由具有 $\mathcal{A}$ 的顺序变量自动机表示，其中 n 个状态。那么，对于每个文档 d，都存在一个非确定性只读分支程序 G ，用于表示 $\mathit{Bool}(P, d)$ 且具有 $O(|d| \times n \times |\text{SVars}|)$ 个节点。

[1] 我们指出，我们在此引入的对象有时也称为无环读一次切换网络和整流网络，但已知这些网络与更常见的 NROBPs 定义在常数因子范围内等价 [24], 因此我们不作区分。

证明概要。 直观地说，我们计算变量自动机与输入文档的乘积，从而得到一个表示该变量自动机在文档上运行过程的有向无环图。通过重新标记标记转换并进行其他一些修改，我们得到非确定性只读分支程序。◂

现在我们将使用这样一个事实：与其他布尔函数表示相比，NROBPs 的简洁性要低指数级。具体而言，我们将读取次数为 *3*的单调 *2-CNF* 公式定义为变量集合 X 上的一个子句合取，其中每个子句都是来自 X的 2 个变量的析取，并且每个变量总共至多出现3 次。我们利用这样一个事实：将此类公式转换为 NROBPs 可能会产生指数级膨胀。该结果是已知的（例如，参见 [7]），但为方便起见，我们在附录D.2中给出证明：

▸ **命题 5.4** ([7])。 对于任意 $n \in \mathbb{N}$，存在一个读取次数为 *3* 的单调 *2-CNF* 公式 Φ，它具有 n个变量，大小为 $O(n)$，使得 Φ 的每一种作为非确定性只读分支程序的表示，其大小均为 $2^{\Omega(n)}$。

现在我们完成定理 5.1 的证明，证明思路如下（详见附录 D.2 ）：

证明概要。 给定一个读取次数为 3 的单调 2-CNF 公式 Φ，我们将展示如何构造一个正则跨度器，使其上的天际线算子所捕获的映射恰好对应于 Φ的满足赋值。根据引理，表示该跨度器的顺序 VA 可以高效地转换为非确定性只读分支程序 5.3，因此可以得出结论：应用于命题 5.4中的公式族时，所有顺序 VA 表示的规模都是指数级的。◂

一项独立结果：无模式连接的状态复杂度下界。我们认为，用于证明定理 5.1的布尔函数关联可能作为一种通用技术，用于证明文档跨度器状态复杂度下界，具有独立的研究价值。事实上，除了天际线算子之外，我们也可以利用这一关联，证明表示两个正则表达式公式自然连接的顺序 VA 的大小下界。 自然连接算子是跨度器上的一种标准算子，用于合并相容的映射：形式化定义见附录 A 。我们有：

▸ **定理 5.5。** 对于每个 $n \in \mathbb{N}$，都存在大小为 $O(n)$ 的正则表达式公式 e_n 和 e'_n，使得每个等价于 $e_n \bowtie e'_n$ 的顺序变量自动机都至少有 $2^{\Omega(n)}$ 个状态。

这一结果是关于状态复杂度的对应结论，与计算两个正则表达式公式的连接这一问题的 NP-困难性相对应 [22]。它只适用于无模式情形；事实上，在基于模式的情形中，已知两个函数变元自动机的连接可以在多项式时间内计算为一个函数变元自动机 [13]。

6 天际线算子的复杂性

我们已经证明，对正则跨度器应用天际线算子后，无法在不造成规模指数级膨胀的情况下将其表示为正则跨度器，即使支配规则本身由正则跨度器表示也是如此（具体而言，这里指变量包含支配规则）。现在，我们研究能否在不将天际线算子编译进自动机的情况下，高效地对其求值。形式化地说，我们研究天际线提取的计算复杂度：

▸**定义 6.1。** 天际线提取问题 定义如下：给定一个文档 d、一个用于捕获正则跨度器的顺序变量自动机 A $P_{\mathcal{A}}$，以及一个由顺序变量自动机表示的支配规则 D，计算天际线算子 $(\eta_D P_{\mathcal{A}})(d)$结果中的映射集合。

数据复杂度。 首先注意，从数据复杂度的角度来看，天际线提取显然是可处理的；在这种情况下， d 是唯一的输入：

▸**命题 6.2**。 对于任意固定的顺序的变量自动机 A，以及表示为顺序变量自动机 D的支配规则， P_A 和 D 的天际线提取问题可以在数据复杂度下的多项式时间内求解，即在关于输入 d的多项式时间内求解。

证明。 我们只需具体化所有捕获映射的集合 $(P_A)(d)$，由于A 是固定的，这可以在多项式时间内完成。然后，对于任意一对映射，我们可以使用支配规则 D检查支配关系是否成立；这同样可以在多项式时间内完成。随后，我们在多项式时间内返回极大映射的集合。◂

注意，对于使用多个天际线算子以及正则跨度算子组合而成的固定表达式，该结果也可以轻松扩展，因为所有这些算子的运行时间都是多项式的。

组合复杂度。 现在我们转向组合复杂度设置，在这些设置中，支配规则 D 或跨度器 P 被视为输入的一部分。请记住，我们关注表示为顺序变量自动机的正则跨度器，因为已知对于这类跨度器，跨度器求值的组合复杂度是输出多项式的 [19]。

正如我们在第 4 节所见，就表达能力而言，正则跨度器对于所有可表示为正则跨度器的支配规则均封闭，尤其是示例 3.2–3.5 中的支配规则。然而，我们在第 5 节看到，将天际线编译为变量自动机通常可能导致指数级膨胀，即使支配规则是固定的也不例外。因此，无法指望通过对该变换的结果应用已知的求值算法（例如，来自 [13, 12, 3] 的算法）来证明天际线提取问题具有可处理性。

这引出了这样一个问题：是否存在其他方法可以在不具体化等价变量自动机的情况下，以高效的组合复杂度解决天际线提取问题。在本节中，我们将表明，在假设 P ≠NP 的情况下，事实并非如此。我们的下界对于一个固定的支配规则已经成立，即变量包含支配规则；事实上，它甚至对于查询复杂性也成立，即文档是固定的。

▸**定理 6.3.** 存在一个固定的文档 d，使得以下问题是 *NP* 困难的：给定一个编码正则跨度器 P_A 的顺序变量自动机 A 和一个数 $n \in \mathbb{N}$，判断 $(\eta_{varInc} P_A)(d)$ 是否包含多于 n 个映射。

这将意味着，有条件地说，天际线提取问题在组合复杂度方面是难处理的。我们用输出多项式算法的语言来表述这一点：对于问题 f： $\Sigma^* \to \Sigma^*$ 如果给定输入 x，其运行时间是关于 $|x| + |f(x)|$的多项式，则称其以输出多项式时间运行。具体而言，我们使用以下关于输出多项式时间与判定问题之间的民间联系，类似的构造见例如 [8] ：

▸ **引理 6.4**。设 f： $\Sigma^* \to \Sigma^*$，并设 p 为一个多项式。假设给定输入 x 和整数 $k \leqslant p(|x|)$，判定 $|f(x)| < k$是否成立是NP-困难的。那么，除非P = NP，否则不存在针对 f的输出多项式时间算法。

由定理6.3和引理6.4，我们可以直接得到我们的难处理性结果：

▸ **推论 6.5**。 除非P =NP，否则不存在一种针对变量包含支配规则的天际线提取问题的算法，该算法在组合复杂度（即输入顺序变量自动机）方面不是输出多项式的，即使输入文档是固定的。

12 文档跨度的天际线算子

注意，该结果与定理5.1不可比较：等价变量自动机大小的下界通常并不排除存在其他在组合复杂度下可处理的算法；反过来，原则上也可能出现这样的情况：求值在组合复杂度下是难处理的，但存在规模较小且难以计算的等价变量自动机。此外，证明也有所不同。具体而言，定理 5.1 的证明使用了单调2-CNF 公式；对于这些公式，我们可以计算出能够精确表示满足赋值的跨度器，但其可满足性问题是可处理的。正如我们将看到的，定理 6.3 的证明利用了 CNF 公式上的SAT 难处理性，但并未使用满足赋值的精确表示。

定理6.3的证明。我们将在本节剩余部分给出定理6.3 的证明，并在结尾补充一项观察。下一节中，我们将研究如何将困难性推广到其他支配关系（尤其是第3 节中引入的所有支配关系，平凡的自支配关系除外），并探讨可处理情形是否存在。

定理6.3的证明。我们从可满足性问题 SAT 归约。设 F 是一个包含 $i \in [n_x]$ 个布尔变量 x_i （ n_x ）和 $j \in [n_c]$个子句 C_j （ n_c ）的合取范式公式。为方便起见，定义集合 $T_i = \{j \mid x_i$ 表示在 $C_j\}$ 中正向出现的变量，并定义集合$F_i = \{j \mid x_i$ 表示在 $C_j\}$中负向出现的变量。我们将在变量 $v_{i,j}$ 上构造一个正则跨度器，其中$i \in [n_x]$ 和 $j \in [n_c]$，并包含一个特殊变量 a。

我们将定义两个跨度器 r_{valid} 和 r_{mask}，二者均为正则表达式公式，并在空文档 $d = \varepsilon$ 上对其求值。先概述一下思路：跨度器 r_{valid} 将针对 F的变量的每种可能赋值，提取一个映射。每个这样的映射都会通过为相应的跨度器变量 $v_{i,j}$赋予跨度，编码该赋值中哪些变量满足了哪些子句。第二个跨度器 r_{mask} 将捕获 n_c个额外映射，这些映射是极大的（得益于额外变量 a），并且每个映射都支配由 r_{valid} 捕获的、其对应赋值不满足 F的某个特定子句的映射。这将确保 F 可满足，当且仅当天际线中 $r_{\text{valid}} \vee r_{\text{mask}}$ 在 d上的映射数量严格多于 n_c。

形式化地，我们将跨度器定义为正则表达式公式，其中的点表示连接：

$$r_{\text{valid}} = \cdot_{i\in[n_x]}((\cdot_{j\in T_i} v_{i,j}\{\varepsilon\}) \vee (\cdot_{j\in F_i} v_{i,j}\{\varepsilon\})) \qquad r_{\text{mask}} = a\{\varepsilon\} \cdot \bigvee_{k\in[n_c]} \cdot_{i\in[n_x],j\in[n_c]\setminus\{k\}} v_{i,j}\{\varepsilon\}.$$

此定义可在关于输入合取范式 F 的多项式时间内完成。

请注意， r_{mask} 捕获的映射永远不会被支配。首先，它们彼此之间不会互相支配：每个映射都不为某个 k赋值 $v_{i,\,k}$。此外， r_{mask}的所有映射都会为 a 赋值，而 r_{valid} 的所有映射都不会，因此后者不可能支配前者。

为了从 r_{valid}捕获的映射 m 构造合取范式变量赋值，我们采用以下编码：如果映射 m 将跨度 $[0,\ 0\rangle$ 赋给跨度器变量 $v_{i,\ j}$，那么这表示变量 x_i 出现在子句 C_j 中，并且 x_i 的赋值满足 C_j。 r_{valid} 的定义确保所有至少出现一次的变量都恰好被赋予真或假这两个真值之一。

我们断言，在 $d = \varepsilon$ 上，天际线 $(\eta_{varInc}(r_{\text{valid}} \vee r_{\text{mask}}))(d)$ 至少包含 $n_c + 1$ 个映射，当且仅当 F 可满足。首先假设 F 可满足，并令 v 为一个满足赋值。那么存在一个对应的映射 m，由编码 v 的 r_{valid} 捕获。事实上，由于 v 满足所有子句，对于每个子句索引 $j \in [n_c]$，都存在一个变量 x_i，其由 v 赋值的方式使 C_j 为真，也就是说，$v_{i,\ j}$ 被赋值。因此，任何由 r_{mask} 捕获的映射都不会支配 m。所以， $r_{\text{valid}} \vee r_{\text{mask}}$ 的天际线必定包含某个

由 r_{valid} 捕获的映射，即 m ，或由 r_{valid} 捕获并支配 m 的其他某个映射。无论哪种情况，天际线都必须至少包含 $n_c + 1$ 个映射。

现在假设 $r_{\text{valid}} \vee r_{\text{mask}}$ 的天际线至少有 $n_c + 1$ 个映射。根据构造，r_{mask} 恰好捕获 n_c 个极大映射，因此天际线中至少有一个映射 m 被 r_{valid} 捕获。该映射 m 编码了 F 的变量的一个赋值 v 。由于 m 不被 r_{mask} 捕获的任何映射支配，对于每个子句索引 $j \in [n_c]$ ，必定存在一个变量索引 $i \in [n_x]$ ，使得 $v_{i,j}$ 被赋值。因此， v 是 F 的一个满足赋值。总之，我们已经证明， F 可满足，当且仅当$\eta_{varInc}(r_{\text{valid}} \vee r_{\text{mask}})$ 至少有 $n_c + 1$ 个满足映射，由此完成证明。◂

最后，我们注意到可以对推论6.5稍作修改：该结果不仅适用于隐式表示乘积的固定单变量支配规则，也适用于在整个定义域上显式指定为正则跨度器的支配规则：

▸ **推论 6.6**。 假设 P ≠NP，即使支配关系由一个顺序变量自动机给出（而非通过隐式计算多个单变量序列 *VA* 的乘积），也不存在一种在组合复杂度下具有输出多项式时间的天际线提取问题算法。

7 难处理和可处理的支配规则

我们已经证明，对于正则跨度器而言，天际线提取问题在组合复杂度方面是难处理的；并且，在固定的逐变量支配规则（即变量包含规则）下，这种难处理性仍然存在。然而，对于其他支配规则，例如跨度包含规则，这一问题仍未得到解答——尤其是当我们将注意力限制在基于模式的跨度器上时；这类跨度器通常具有更好的性质（例如，就连接和差分算子的复杂性而言 [10]）。

在本节中，我们将证明，遗憾的是，在这种情境下，困难性依然成立。具体而言，我们为支配规则引入一个条件，称为具有无界多个互不相交的严格支配对(UMDSDP)。我们的示例支配规则（自支配除外）显然都满足此条件。接着，我们将证明，UMDSDP 是不可处理性的充分条件：这一结果重新确立了变量包含的困难性（定理6.3），并且还证明了跨距包含、从左到右和跨度长度支配规则的困难性。

接着，我们介绍一类受限的支配规则，称为变量包含类支配规则，并证明在这一类规则中，UMDSDP 条件的一个变体实际上刻画了难处理的情形。特别是，所有不满足该条件的此类支配规则都具有可处理的天际线提取。最后，我们研究一般支配规则的其他例子，并证明在 UMDSDP 未涵盖的规则中，有些容易处理，有些则难以处理。

UMDSDP 条件。为了介绍天际线提取难处理的一个充分条件，我们先定义不相交严格支配对。

▸ **定义 7.1**。对于文档 d 上的支配关系 $\preccurlyeq$ ， d 上 $\preccurlyeq$ 的严格支配对是由跨度 (s_1, s_2) 组成的一对，满足 $s_1 \preccurlyeq s_2$ 和 $s_1 \neq s_2$。若两对这样的支配对 $\{s_1, s_2\}$ 和 $\{s'_1, s'_2\}$ 满足以下条件，则称它们不相交：令 s 为包含 s_1 和 s_2 的最小跨度，令 s' 为包含 s'_1 和 s'_2 的最小跨度，则 s 和 s' 不相交。否则，它们重叠。

当支配对 (s_1, s_2) 中的一个 s_i 未被赋值时，我们用破折号表示：“–”。为了判断是否不相交，我们称“–”包含于每个跨度中。

▸示例 7.2。 $([1,3\rangle,[2,4\rangle)$ 和 $([9,10\rangle,[6,8\rangle)$ 不相交。 $([1,3\rangle,[7,9\rangle)$ 和 $([4,6\rangle,[10,12\rangle)$ 重叠（尽管所有组成的跨度都互不相交）。最后，$(-,[1,3\rangle)$ 和 $([4,6\rangle,[10,12\rangle)$ 也不相交。

现在我们可以定义 UMDSDP 条件，它足以证明问题的困难性：

▸**定义 7.3**。 单变量支配规则 D 具有无界多个互不相交的严格支配对（*UMDSDP*），如果给定 $n \in \mathbb{N}$，我们可以在关于 n 多项式时间内计算出一个文档 $d \in \Sigma^*$ 和 n 个严格支配对 S_1、……、S_n，它们来自 $D(d)$ 且两两不相交。

▸**例 7.4**。 D_{self} 不满足 UMDSDP，因为它没有严格支配对。

跨度长度支配规则满足 UMDSDP。实际上，对于 $n \in \mathbb{N}$，我们可以取词 a^n，以及不相交严格支配对 $\{([i,\ i\rangle,\ [i,\ i+1\rangle) \mid i \in [0,\ n-1]\}$。同样的支配对表明，跨度包含规则和从左到右规则也满足 UMDSDP。

最后，变量包含支配规则满足 UMDSDP，其配对集合为$\{(-,[i,i+1\rangle) \mid i \in [0,n-1]\}$（请记住约定：“–”表示某个变量未被赋值）。

考虑支配关系 D_{start} 定义支配关系 $\preccurlyeq_{start}$ 其中包含以下各对 $\{([1,\ i\rangle,\ [1,\ j\rangle) \mid i, j \in \mathbb{N},\ i \leqslant j\}$ 以及自反性的平凡对 $(-,\ -)$。那么 $\preccurlyeq_{start}$ 有无界多个严格支配对，但其中任意两对都不相交，因此不满足 UMDSDP 条件。（不过，我们仍然能够证明此规则是难解的；参见命题 7.9。）

我们指出，对于正则的单变量支配规则，只要存在 任意多个两两不相交的严格支配对，UMDSDP条件就成立（也就是说，在这种情况下，我们总能高效地计算它们）；详见附录 F.1 。

UMDSDP 蕴含困难性。 现在我们证明，UMDSDP 条件意味着天际线提取是困难的。证明是变量包含证明的一种变体：

▸**定理 7.5**. 设 D 是一个满足 *UMDSDP* 的单变量支配规则。对于给定的顺序变量自动机 D，$\mathcal{A}$ 和文档 $d \in \Sigma^*$，其天际线提取问题不是输出多项式可解的，除非 P =?= NP。

这意味着此前介绍的其他逐变量支配规则也具有困难性，从而完成推论 6.5。请注意，这些规则针对的是基于模式的跨度器；我们还可以注意到，即使输入跨度器是函数式的（即基于模式的），困难性依然成立：

▸**推论 7.6**。 除非 P = NP，否则不存在一种算法，能够在组合复杂度下以输出多项式时间，针对跨度包含支配规则、从左到右支配规则或跨度长度支配规则解决天际线提取问题。即使要求输入变量自动机是函数式的，这一结论仍然成立。

变量包含类规则。 我们已经看到，UMDSDP 条件是天际线提取问题具有困难性的充分条件，但这并未解决它是否也是必要条件的问题。现在我们将重点关注一类支配规则，称为变量包含型支配规则，对于这类规则，情况正是如此。形式化地说，如果对于所有严格支配对 (m_1, m_2)，以及所有变量 $x \in$，只要 $m_1(x)$ 有定义，$m_2(x)$ 就也有定义，并且 $m_1(x) = m_2(x)$，那么我们就称支配关系 $\preccurlyeq$ 是变量包含类的。

与包含所有此类 $(m_1$、$m_2)$对的变量包含规则不同，我们只要求其中的一个子集在 $\preccurlyeq$中成立。我们将以逐变量的方式定义变量包含型支配

规则：对于单变量变量包含型规则，严格支配对必然具有 $(-、\ s)$ 的形式，其中 s是一个跨度。换言之，逐变量的变量包含型支配规则通过指明在每个文档上哪些跨度 s 可以作为此类支配对的右侧来定义。此外，对于变量包含型规则，当且仅当两个严格支配对的右侧不相交时，它们才是不相交的。

我们可以证明，对于变量包含型支配规则，某个 UMDSDP 条件变体存在一个二分结果：

▸**定理 7.7**。 设 D 是一个变量包含型单变量支配规则。如果D 满足 *UMDSDP* 条件，或者在某个文档上接受形如 $(-, [i, i\rangle)$ 的变量对，那么对于给定顺序变量自动机和文档的 D 天际线提取问题在组合复杂度上不是输出多项式，除非 P =NP。否则，对于 D 的天际线提取问题在组合复杂度上是输出多项式。

该二分结果的下界由定理 7.5得出，此外还需注意，只要存在一对形如 $(-, [i, i\rangle)$ 的变量对，就足以证明其具有困难性：

▸**引理 7.8**。 设 D 是一个单变量支配规则，在某个文档上接受一对变量 $(-, [i, i\rangle)$。那么，对于给定顺序变量自动机和文档的 D天际线提取问题在组合复杂度上不是输出多项式，除非 P =NP。

因此，定理 7.7 中值得关注的结果是上界。我们在附录中证明了这一点：对于不满足 UMDSDP 的变量包含类规则，其严格支配对的右侧集合具有有界的击中集数，并且这意味着问题具有可处理性。

其他情况。定理 7.5 和 7.7并未解决非 UMDSDP 且非变量包含类的支配规则的复杂性问题。最后，我们给出一些可以证明是难处理的规则示例。我们首先证明早先引入的规则 $\preccurlyeq_{start}$ 是难处理的：

▸**命题 7.9**。 请回顾示例 D_{start} 中的逐变量支配规则 *7.4*。假设 P ≠NP，则该规则的天际线提取问题不存在输出多项式综合复杂度算法。

我们展示了另一条不满足UMDSDP 的规则的困难性，其中所有严格支配对共享同一个右侧：

▸**命题 7.10**。 考虑由正则表达式 $x\{a^*\}a^*x^\dagger\{b\} \vee D_{self}$表示的逐变量支配规则。假设 P ≠ NP，则该规则对应的天际线提取问题不存在输出多项式综合复杂度算法。

然而，我们注意到，该规则的逆规则（其中所有严格支配对共享同一个左侧）实际上是可处理的（而且也不满足 UMDSDP）。这说明，反直觉的是，逐变量支配规则的复杂性分类在左侧和右侧之间并不对称：

▸**命题 7.11**。 变量逐一支配规则$x^\dagger\{a^*\}a^*x\{b\} \vee D_{self}$对应的天际线提取问题具有输出多项式综合复杂度。

8 结论

我们介绍了支配规则的一般框架，用以表示文档跨度器的天际线算子，其中规则本身也表示为跨度器。我们已经

证明了该算子（采用正则规则）不会提升正则跨度器的表达能力，但会导致状态复杂度不可避免地发生指数级膨胀；对于许多自然的固定规则而言，在组合复杂性方面，该算子的求值也是难处理的。

我们的工作留下了几个有待未来研究的问题。最直接的问题是，一般情况下，天际线提取问题是否在逐变量正则支配规则下存在二分性，即将定理7.5 推广到任意此类规则。不过，这似乎颇具挑战性。另一个问题是，第7 节的困难性结果是否也能给出第5节所示类型的状态复杂度下界，尤其是在基于模式的上下文中；以及状态复杂度是否存在二分性。

最后，一个引人关注的问题是：从天际线中计算固定数量 k 映射的top-k 问题 在组合复杂性方面是否总是可处理。我们的困难性结果均未排除这种可能性，但我们尚未发现解决该问题的算法。

参考文献

1 Foto N. Afrati、Paraschos Koutris、Dan Suciu 和 JeffreyD. Ullman。并行天际线查询。载于 ICDT，2012。 2 Shqiponja Ahmetaj、Wolfgang Fischl、Markus Kröll、Reinhard Pichler、Mantas Šimkus和 SebastianSkritek。SPARQL 中可选匹配的挑战。载于 FoIKS，2016。 3Antoine Amarilli、Pierre Bourhis、StefanMengel和Matthias Niewerth。非确定性文档跨越器的常数延迟枚举。载于 ICDT，2019。 doi:10.4230/LIPIcs.ICDT.2019.22。 4 Antoine Amarilli、Florent Capelli、MikaëlMonet和 Pierre Senellart。连接知识编译类与宽度参数。TheoryComput. Syst.，64(5)，2020。 doi:10.1007/s00224-019-09930-2。 5 Stephan Börzsönyi、DonaldKossmann 和 KonradStocker。天际线算子。载于 ICDE。IEEE，2001。 6 Craig Boutilier、Ronen I. Brafman、Carmel Domshlak、Holger H.Hoos 和 DavidPoole。CP-net：用于表示和推理条件式其他条件相同偏好陈述的工具。J.Artif. Intell. Res.，21，2004。 doi:10.1613/jair.1234。 7 Simone Bova、Florent Capelli、Stefan Mengel和 Friedrich Slivovsky。DNNF 与合取范式公式之间的强指数分离。CoRR，abs/1411.1995，2014。URL：https://arxiv.org/abs/1411.1995， doi:10.48550/ARXIV.1411.1995。 8 FlorentCapelli 和 Yann Strozecki。增量延迟枚举：空间与时间。Discret. Appl. Math.，268，2019。 doi:10.1016/j.dam.2018.06.038。 9 Hamish Cunningham、Kevin Humphreys、Robert J. Gaizauskas和 Yorick Wilks。GATE——文本工程通用架构。载于ANLP。ACL，1997。10 Ronald Fagin、Benny Kimelfeld、Frederick Reiss 和 Stijn Vansummeren。文档跨度：信息抽取的形式化方法。J.ACM，62(2)：12，2015。URL： http://doi.acm.org/10.1145/2699442， doi:10.1145/2699442。11 Ronald Fagin、Benny Kimelfeld、FrederickReiss和Stijn Vansummeren。信息抽取中不一致的声明式清理。ACMTrans.DatabaseSyst.，41(1)，2016。URL:http://doi.acm.org/10.1145/2877202， doi:10.1145/2877202。 12 Fernando Florenzano、Cristian Riveros、Martín Ugarte、StijnVansummeren和 Domagoj Vrgoc。正则文档跨越器的常数延迟算法。载于 PODS，2018。 13 Dominik D. Freydenberger、Benny Kimelfeld和LiatPeterfreund。正则表达式抽取结果的连接。载于 PODS，2018。 14César A. Galindo-Legaria。将外连接视为析取。SIGMOD Rec.，23(2)，1994。 doi:10.1145/191843.191908。 15 Ronald L. Graham、BruceL.Rothschild 和 Joel H. Spencer。拉姆齐理论，第 20 卷。John Wiley & Sons，1991。

16 Martin Grohe 和 Dániel Marx。关于树宽、荆棘集大小和扩张率。《组合理论杂志 B 辑》，99(1)，2009。 doi:10.1016/j.jctb.2008.06.004。17 Benoit Groz 和 Tova Milo。具有含噪比较的天际线查询。载于 PODS，2015。18 Yunyao Li、Frederick Reiss 和 Laura Chiticariu。SystemT：一种声明式信息抽取系统。载于 ACL，2011。19 Francisco Maturana、Cristian Riveros 和 Domagoj Vrgoc。用于抽取不完整信息的文档跨度器：表达能力与复杂性。载于 PODS，2018。20 Liat Peterfreund。文本抽取结果上的关系查询复杂性。博士论文，Technion - 计算机科学系，2019。21 Liat Peterfreund。文档跨度器的文法。载于 ICDT，2021。22 Liat Peterfreund、Dominik D. Freydenberger、Benny Kimelfeld 和 Markus Kröll。文档跨度器上关系代数的复杂性界。载于 PODS，2019。 doi:10.1145/3294052.3319699。23 Liat Peterfreund、Balder ten Cate、Ronald Fagin 和 Benny Kimelfeld。用于文档跨度器的递归程序。载于 ICDT，2019。24 Igor Razgon。关于有界树宽分支程序和合取范式的一次读属性。Algorithmica，75(2)，2016。 doi:10.1007/s00453-015-0059-x。25 Markus L Schmid 和 Nicole Schweikardt。处理非正则核心跨度器的纯正则方法。载于 ICDT，2021。26 Cheng Sheng 和 Yufei Tao。最坏情况下 I/O 高效的天际线算法。ACM 数据库系统汇刊（TODS），37(4)，2012。27 Slawek Staworko、Jan Chomicki 和 Jerzy Marcinkowski。关系数据库中的优先修复与一致查询回答。《数学与人工智能年刊》，64(2-3)，2012。

A 其他预备知识

本节中，我们介绍一些我们在附录中使用的概念所需的额外预备知识。 .

引用词。在一些证明中，我们会使用引用词。与 [20], 中一样，我们将引用词定义为由 $(\Sigma \cup$ 标记$(X))$*构成的词，并将变量自动机的引用词语言定义为：将其视为字母表为$\Sigma \cup$ 标记(X)上的通常有限自动机时，它所接受的引用词集合。注意，对于顺序变量自动机，其引用词语言中的引用词都是有效的，也就是说，对于每个变量，要么它的标记完全不出现，要么恰好出现一个开标记，且其后在引用词的某个更晚位置恰好出现一个闭合标记。

修剪。 在证明中，通常假设变量自动机是修剪的，也就是说，我们移除所有不属于接受运行的状态。（自动机没有接受运行的情况是平凡的，因此我们通常隐式排除该情况，并假设得到的变量自动机定义良好。）我们可以在线性时间内修剪变量自动机，而且这不会影响它所定义的跨度器，也不会影响顺序性或函数性。此外，这只会使状态数量减少。拥有一个修剪的顺序变量自动机，可以确保对于变量自动机的所有部分 运行，我们都以有效方式分配标记，即：对于每个变量，我们首先尚未分配该变量的任何标记，然后可以为该变量分配一个开启标记，接着可以为该变量分配一个闭合标记。（相比之下，对于未修剪的顺序变量自动机，在无法完成为接受运行的部分运行中，该条件可能会被违反。）

ε-转移。在一些证明中，我们将考虑一种略作修改的变量自动机模型，其中还允许使用 ε-转移。形式上，变量自动机的转移包括字母转移、标记转换，以及标记为 ε 、可在运行过程中自由执行的转移。请注意，这一修改显然不会改变变量自动机形式体系的表达能力或简洁性，因为可以按照通常的方法，将含有 ε-转移的变量自动机改写为不含此类转移且状态数相同的变量自动机：将所有存在一条 ε-转移路径通向终态的状态设为终态；对于每个从状态 q 到状态 q'的字母转移或标记转换，以及每个存在一条 ε-转移路径通向 q的状态 q''，添加一条从 q'' 到 q' 、标签相同的转移。这种转换可以在多项式时间内完成，且不会改变所捕获的引用词，因此不会影响变量自动机是顺序的或函数式这一性质。

跨度器代数。 我们在此介绍一些跨越器上的算子及其性质。

对于每个跨度器 P 以及每个 $Y \subseteq \mathsf{SVars}(P)$子集，我们通过如下方式定义投影算子$\pi_Y$： $\pi_Y P$是这样一个跨度器，它对每个文档 d 提取集合 $(\pi_Y P)(d) :=\{m|_Y \mid m \in P(d)\}$，其中 $m|_Y$是 m 在 Y 上的限制。

自然连接 $P_1 \bowtie P_2$ 两个跨度器 P_1 和 P_2 是一个跨度器，它接受所有映射 m，这些映射是两个相容映射的并集，其中一个被 P_1 接受，另一个被 P_2接受。换言之， $(P_1 \bowtie P_2)(d) := \{m \in$ 映射集合 $\mid \exists m_1 \in P_1(d),\ \exists m_2 \in P_2(d),\ m_1 \sim m \wedge m_2 \sim m \wedge \mathsf{dom}(m_1) \cup \mathsf{dom}(m_2) = \mathsf{dom}(m)\}$。

我们指出，如果 $\mathsf{SVars}(P_1) \cap \mathsf{SVars}(P_2) = \varnothing$，那么连接算子就是前面定义的笛卡尔积。

交运算符 $\cap$ 定义为计算跨度器 $P_1 \cap P_2$，它在每个文档上计算集合 $(P_1 \cap P_2)(d)$: $= P_1(d) \cap P_2(d)$。注意，如果 $\mathsf{SVars}(P_1) =$

SVars(P_2)，且两个跨度器都基于模式的，那么连接算子就是交集：$(P_1 \bowtie P_2)(d) = P_1(d) \cap P_2(d)$。

并集 $P_1 \cup P_2$ 定义为这样的跨度器：它对每个文档都计算集合 $(P_1 \cup P_2)(d)$：$= P_1(d) \cup P_2(d)$。

差集，$P_1 - P_2$ 是一个二元算子，它接受所有被 P_1接受、但不被 P_2接受的映射。换句话说，$(P_1 - P_2)(d) := P_1(d) \backslash P_2(d)$。请注意，这是集合上的常见差集算子，而不是中定义的差集算子，后者 [20] 接受 P_1 中的映射，且不存在被 P_2接受的相容映射。

众所周知，投影、自然连接算子和并集算子不会提升正则跨度器的表达能力；关于基于模式的跨度器的情形，参见 [10] ；关于无模式跨度器的情形，参见 [19] 。由此可知，笛卡尔积算子也同样如此。我们将在第A.2 节稍后证明，交集也不会提升表达能力。至于差分算子，[10] 证明了基于模式的正则跨度器也有相同结果，但我们尚未发现关于无模式跨度器以及我们所采用的差集语义的相同结果。

A.1 无模式正则跨度器对差集封闭

我们展示如下结论：

▸**命题 A.1**。（无模式）正则跨度器对差集封闭：给定两个正则跨度器 P_1 和 P_2，差集 $P_1 - P_2$ 可以表示为一个正则跨度器。

注意，如果命题中给出的跨度器 P_1、P_2 是以顺序变量自动机的形式给出的，那么在命题 $P_1 - P_2$ 的证明中为 A.1 构造的变量自动机通常会大得多，规模呈指数增长。

证明。设 A_1 和 A_2 分别是关于 P_1 和 P_2 的两个顺序变量自动机，设 X 为它们的变量集合，并回忆一下，标记(X) 是形如 $x\vdash$ 或 $\dashv x$（关于 $x \in X$）的标记集合。我们使用引用词的概念（见附录 A）。由于变量自动机是顺序的，它们的接受运行只能捕获有效的引用词。此外，对于文档 $d \in \Sigma^*$，顺序变量自动机产生的映射集合可以从其引用词语言中的字符串得到：这些字符串擦除（即移除标记符号）后得到 d；对于每个这样的引用词w，我们得到一个映射，将其标记出现在 w中的变量映射到由唯一的开始标记和闭合标记所定义的跨度上。然而，同一个映射可能由多个引用词得到，因为标记之间的顺序可能不同。例如，引用词 $x\vdash y\vdash a \dashv x \dashv y$和 $x\vdash y\vdash a \dashv y \dashv x$ 定义了同一个映射。

为避免这种情况，我们将对自动机进行规范化（注意，在基于模式的情形中也进行了类似的规范化，见 [10] ，其名称为 字典序 变量自动机）。令 $<$ 为 标记(X)上的全序：我们规定所有开符号都先于所有闭符号，即 $x\vdash < \dashv y$ 对每个 x，$y \in X$，以确保引用词仍然有效。若一个顺序变量自动机中，其引用词语言里的任意引用词所含的每个连续标记子序列都按照 相对于 有序 $<$ ，则称该自动机相对于 $<$有序。

不难看出，我们可以对顺序变量自动机进行改写，使其成为有序的，代价是指数级膨胀。给定一个输入顺序变量自动机 $\mathcal{A} = (Q, q_0, F, \delta)$，构造有序变量自动机 $\mathcal{A}' = (Q \cup Q', q_0, F \cup F', \delta')$，其中 Q' 是 Q 的加撇副本（即 $Q' = \{q' \mid q \in Q\}$），F' 以相同方式定义，而 δ' 初始化为执行 δ 的字母转移，不过转移起点可以是不带撇状态或带撇状态，终点则为不带撇状态：形式化地说，对每个字母转移 $(q_1, a, q_2) \in \delta$，我们向 δ中加入两个字母转移 (q_1, a, q_2) 和 (q_1', a, q_2)。

现在，考虑$\mathcal{A}$ 中所有仅经过标记转换的路径，并令 q_1 和q_2 分别为初始状态和终态；在$\mathcal{A}'$ 中添加一条从 q_1 到 q_2 的路径，该路径经过新状态，读取相同的标记，但顺序按照 $<$给出。

经过此修改后，我们断言$\mathcal{A}'$接受与$\mathcal{A}$相同的引用词语言。事实上，$\mathcal{A}$ 中从初始状态到终态的任意路径，都可以在$\mathcal{A}'$中重现：每当我们在$\mathcal{A}$中经过标记转换时，接着就在$\mathcal{A}'$ 中沿一条路径前进，到达某个带撇状态；该状态随后与相应的不带撇状态无法区分。反过来，$\mathcal{A}'$ 中的任意路径也都可以在$\mathcal{A}$中重现。此外，$\mathcal{A}'$ 相对于 $<$是有序的：这是因为所有由连续标记符号组成的路径在构造时都已排序。（请注意，我们区分带撇状态与不带撇状态，是为了确保一旦沿某条路径前进，就不能在不经过字母转换的情况下继续沿另一条路径前进。）

回到输入的变量自动机 P_1 和 P_2，我们对它们应用这一变换，得到相对于同一顺序有序的变量自动机 P_1' 和 P_2'。现在，我们可以对 P_2 进行一些标准的自动机理论操作，最后构造一个乘积自动机。具体而言，首先修改 P_2，使其完备，即添加一个汇状态 $q_\perp$，并确保从每个状态（包括 $q_\perp$）出发，对于所有字母和标记都有一条转移到 $q_\perp$：由于新状态不会出现在任何接受运行中，因此得到的变量自动机仍然是顺序的。其次，通过应用有限自动机的标准确定化过程，使 P_2 成为关于 $\Sigma \cup$ 标记 (X) 的确定自动机。得到的变量自动机接受相同的引用词语言，因此它仍然是顺序的，且相对于同一顺序有序：不过，该变量自动机现在具有确定性，即其引用词语言中的每个引用词（因而每个映射）都恰好有一个接受运行，并且每个引用词都是自动机中从初始状态出发的一条路径的标签（该路径通常不接受，也就是说，它可能以汇状态结束）。

以这种方式修改 P_2' 后，让我们构造乘积自动机。记作$P_1' = (Q_1, q_0, \mathbf{1}, F_1, \delta_1)$和 $P_2' = (Q_2, q_0, 2, F_2, \delta_2)$。我们构造 $(Q_1 \times Q_2, (q_0, \mathbf{1}, q_{0,2}), F_1 \times (Q_2 \setminus F_2), \delta)$，其中定义 δ 在两个分量中都进行转移，即对于每个标记或字母 ℓ，以及每个 $(q_1, \ell, q_1') \in \delta_1$ 和 $(q_2, \ell, q_2') \in \delta_2$，将 $((q_1, q_2), \ell, (q_1', q_2'))$ 添加到 δ。

我们断言，所得变量自动机是顺序的，因为任意接受运行投影到第一个分量后，都会得到 P_1的一个接受运行，而它是顺序的。出于同样的原因，它也是有序的。现在，我们断言，该自动机接受的引用词语言是 P_1' 的引用词语言与 P_2'的引用词语言之差：这是因为我们对自动机进行了标准乘积构造，而且自动机 P_2' 是确定性的。形式化地说，乘积产生的任意引用词都会为该词给出 P_1' 的一个接受运行，以及 P_2' 在该引用词上的唯一非接受运行；因此，这证明了 P_2拒绝该引用词，因为它是确定性的。反过来，任何被 P_1' 接受而被 P_2' 拒绝的引用词，都会利用P_2' 完备这一事实，在乘积中为该引用词给出一个接受运行。

现在可以得出结论：所得变量自动机恰好接受被 P_1'接受且被 P_2'拒绝的引用词。由于 P_1'和 P_2'有序，其差集所接受的映射对应于引用词语言的差集，证明得证。◂

A.2 将变量自动机分解为函数变元自动机，以及交集运算下的封闭性

请记住，函数变元自动机$\mathcal{A}$ 是一种顺序变量自动机，其中$\mathcal{A}$ 在任意文档 d 上捕获的每个映射 m，都会为$\mathcal{A}$定义域中的每个变量指定一个跨度。给定一个跨度器 P 和一个变量集 X，我们用 $P^{[X]}$ 表示这样一个跨度器：它在每个文档 d上提取 $P^{[X]}(d) := \{m \in P(d) \mid \mathsf{dom}(m) = X\}$。

以下结果将在多个地方派上用场：

▸**引理 A.2**。 对于每个正则跨度器 P 和每个变量集 X，都存在一个定义 $P^{[X]}$ 的函数变元自动机。

证明。 假设 $X \subseteq \mathsf{SVars}(P)$；否则该命题显然成立，因为 $P^{[X]}$ 在任何文档上都不捕获任何映射。令 $\mathcal{A}$ 为定义 P的顺序变量自动机。我们说明如何构造一个用于 $P^{[X]}$的函数变元自动机。

首先，我们构造一个变量自动机 $\mathcal{A}^{\leqslant X}$，方法是从$\mathcal{A}$ 中删除所有针对不属于 X 的变量的标记转换。对于任意文档 d，变量自动机 $\mathcal{A}^{\leqslant X}$ 恰好捕获集合 $\{m \in P(d) \mid \mathrm{dom}(m) \subseteq X\}$。注意，$\mathcal{A}^{\leqslant X}$ 是顺序的，因为$\mathcal{A}$ 是顺序的。

同理，对于每个 $x \in X$，我们构造一个捕获$\{m \in P(d) \mid \mathrm{dom}(m) \subseteq X\backslash\{x\}\}$的顺序变量自动机 $\mathrm{A}^{\leqslant(X\backslash x)}$。然后，我们可以如下构造一个在每个文档上捕获$\{m \in P(d) \mid \mathrm{dom}(m) \subsetneq X\}$ 的顺序变量自动机 $\mathrm{A}^{<X}$：通过重命名，假设所有 $\mathrm{A}^{\leqslant(X\backslash x)}$ 的状态集合两两不相交。添加一个新的初始状态 s，并用一条 ε 边将其连接到所有 $\mathrm{A}^{\leqslant(X\backslash x)}$ 的初始状态（回忆一下附录 A 中 ε 转移的定义）。此外，添加一个新的终态 t，并用一条 ε 边将其连接到所有 $\mathrm{A}^{\leqslant(X\backslash x)}$ 的终态。令s 为 $\mathrm{A}^{<X}$ 的初始状态，并令 t 为其唯一的终态。容易看出，所得的变量自动机 $\mathrm{A}^{<X}$ 是顺序的，并接受所需的映射。

请记住，对于每个变量自动机 $\mathcal{A}'$, ，我们用 $P_{\mathcal{A}'}$ 表示由$\mathcal{A}'$定义的跨度器。那么

$$P^{[X]} = P_{\mathcal{A}^{\leqslant X}} - P_{\mathcal{A}^{<X}}.$$

由命题 A.1 我们得到一个计算 $P^{[X]}$ 的顺序变量自动机。此外，由于所有捕获映射都为 X 中的所有变量赋予跨度，因此该变量自动机是函数变元自动机。◂

我们得到以下直接推论，其证明是显然的：

▸ **引理 A.3**。 设 P 是一个正则跨度器。则对于每个 $X \subseteq \mathsf{SVars}$，存在一个定义 $P^{[X]}$ 的函数变元自动机，因此有 $P = \bigcup_{X \subseteq \mathsf{SVars}(P)} P^{[X]}$。

我们推知，正则跨度器在交集运算下封闭。（请注意，与基于模式的情形不同，在无模式情形下，集合上的交运算符不能立即用连接算子表示。）

▸**引理 A.4**。 （无模式的）正则跨度器在交集运算下封闭：给定两个正则跨度器 P_1 和 P_2，其差集 $P_1 \cap P_2$ 可以表示为一个正则跨度器。

证明。 我们使用引理 A.3 来分解 $P_1 = \bigcup_{X \subseteq \mathsf{SVars}(P)} P_1^{[X]}$ 和 $P_2 = \bigcup_{X \subseteq \mathsf{SVars}(P)} P_2^{[X]}$。每当 $X_1, X_2 \subseteq \mathsf{SVars}(P_1) \cup \mathsf{SVars}(P_2)$ 不同时，我们就有 $(P_1^{[X_1]} \cap P_2^{[X_2]})(d) = \varnothing$对每个文档都成立，因为它们 $P_i^{[X_i]}$ 提取的映射具有不同的定义域。因此

$$P_1 \cap P_2 = \bigcup_{X \subseteq \mathsf{SVars}(P_1) \cap \mathsf{SVars}(P_2)} (P_1^{[X]} \cap P_2^{[X]}).$$

由于对于每个 d，$P_1^{[X]}(d)$ 和 $P_2^{[X]}(d)$ 中的所有映射都恰好赋值给 X中的变量，因此在此特定上下文中，交运算符可以表示为自然连接，就像在基于模式的情形中一样：

$$P_1^{[X]} \cap P_2^{[X]} = P_1^{[X]} \bowtie P_2^{[X]}.$$

由于如前所述，正则跨度器在连接下封闭，因此对于每个$\subseteq \mathsf{SVars}(P_1) \cap \mathsf{SVars}(P_2)$，跨度器 $P_1^{[X]} \cap P_2^{[X]}$ 都是正则的。由正则跨度器在并集下封闭可得该引理。◂

B 第 3 节（天际线算子）的证明

▸**引理 3.13**。 单变量跨度长度支配规则 $D_{spanLen}$ 无法表示为正则跨度器。

证明。我们使用标准的抽引论证。反证，假设存在一个顺序变量自动机 P，它定义了 $D_{spanLen}$，变量为 x 和 $x^{\dagger}$。设 n 为 P 的状态数。那么，在输入 a^{2n+3} 上，跨度器 P 必须提取映射 m，其中 $m(x) = [0、\ n+1\rangle$ 和 $m(x^{\dagger}) = [n+1、\ 2n+3\rangle$。由于 $|m(x)| > n$，在针对该映射的一次接受运行中，从打开 x（位置为 $\vdash x$）到再次关闭它（位置为 $\dashv x$）之间，自动机必定两次访问同一个状态。设这两次访问之间的环长度为 i。那么在输入 $a^{2n+3+2i}$ 上，我们可以将该环运行三次而不是一次，因此 P 会提取 m'，其中 $m'(x) = [0、\ 2i+n+1\rangle$ 和 $m'(x^{\dagger}) = [2i+n+1、\ 2n+3+2i\rangle$。然而，由于 $i \geqslant 1$，我们有 $|m'(x)| = 2i+n+1 > 2n+2 = |m'(x^{\dagger})|$，所以本不应提取映射 m'。这与假设 P 定义了 $D_{spanLen}$ 相矛盾，证明完毕。◂

C 第 4 节（在天际线算子下的闭包）证明

C.1 定理证明 4.1

▸**定理 4.1**。 存在一种算法，给定一个定义正则跨度器的顺序变量自动机P 和一个定义支配规则的顺序变量自动机 D，计算一个用于 $\eta_D P$的顺序变量自动机。

证明。 在证明中，我们将使用第 A 节引入的跨越器上的算子，以及如该节所述，正则跨度器对所有这些算子都封闭这一事实。

设 P 是由顺序变量自动机构造的跨度器， D 是定义支配关系的跨度器。我们定义 $P_{\text{sky}} = P - \pi_X((P \times P^{\dagger}) \cap (D - D_{self}))$。直观而言，跨度器 $D - D_{self}$ 提取所有严格支配对（即包含两个不同映射的支配对），而跨度器 $P \times P^{\dagger}$ 提取所有两个映射均会被 P 提取的支配对。此处， $P^{\dagger}$ 表示从 P 构造得到的跨度器；对于 P 捕获的每个映射 m，它都会捕获映射 $m^{\dagger}$。随后，取这两个跨度器的交集并投影到 X，即可得到 P 的被支配映射。移除这些映射后，恰好剩下由 P 提取的非支配映射，因此 P_{sky} 恰好提取 $\eta_D P$ 中的映射。

还需证明 P_{sky} 是正则跨度器。这一点显然成立，因为它是通过应用跨度器运算由正则跨度器构造而成的，而正如我们上文所讨论的，正则跨度器在这些运算下是封闭的。◂

C.2 证明推论 4.2

▸ **推论 4.2**。 存在一些算法，给定一个顺序变量自动机 P，分别为 $\eta_{self} P$、 $\eta_{varInc} P$、$\eta_{ltr} P$和 $\eta_{spanInc} P$计算顺序变量自动机。

证明。 我们在例 3.12 中看到，支配关系 $\preccurlyeq_{self}$、 $\preccurlyeq_{varInc}$和$\preccurlyeq_{spanInc}$ 可以表示为逐变量规则，也就是由正则表达式公式定义的单变量支配规则，因此也可以由顺序变量自动机表示。我们可以计算 P中使用了哪些变量，并取笛卡尔积，从而得到一个在右侧变量集上定义该支配关系的跨度器。注意，顺序变量自动机的笛卡尔积可以由顺序变量自动机表示，因此我们由此得到了定义所需支配规则的顺序变量自动机。随后，我们可以直接根据定理 4.1得出结论。

请注意，在陈述中， $\eta_{self}P$ 的情形实际上是平凡的，因为对自支配规则取闭包不会产生任何影响，所以结果等于 P。我们仅为完备性而陈述关于 η_{self} 的结果。◂

C.3 命题 证明4.3

▸**命题 4.3**。 存在一个顺序变量自动机 P 使得 $\eta_{spanLen}P$ 不是正则的。

证明。 我们证明 $\eta_{spanLen}P$ 不能表示为变量自动机，并使用与引理 3.13 类似的抽引论证。令 P 为变量 x 和字母表$\Sigma = \{a$上的跨度器 $b\}$，其由以下正则表达式公式定义：

$$x\{a^*\}ba^* \vee a^*bx\{a^*\}.$$

那么，在输入 $a^ib j^j$（其中 i、$j \in \mathbb{N}$）上，会提取出两个跨度 $[0$、$i\rangle$ 和 $[i+1$、$i+j+1\rangle$（应用天际线算子之前）；天际线算子应当移除其中一个映射（如果 $i \neq j$），或者不移除任何映射（如果 $i = j$）。我们将证明 $\eta_{spanLen}P$ 不能表示为变量自动机。反设存在一个变量自动机 P' 提取天际线；不失一般性，我们假设它是顺序的。于是，对于任何形如 a^iba^j的文档， P'会在 $i > j$时提取跨度 $[0$、$i\rangle$，并在 $j > i$时提取跨度 $[i+1$、$i+j+1\rangle$。令 n为 P'的状态数。那么，在 $a^{n+1}ba^{n+2}$上，该变量自动机必须提取 $[n+2$、$2n+4\rangle$。与引理 3.13 的证明类似， P' 在提取该跨度的 $a^{n+1}ba^{n+2}$ 运行过程中，必定经过一个长度为 k 的环。然后，经过该环 4 次后， P' 在输入 $a^{n+3k+1}ba^{n+2}$ 上提取跨度 $[n+3k+2$、$2n+3k+4\rangle$。然而，由于 $n+3k+1 > n+2$，本不应提取该跨度，因为它被 $[0$、$n+3k+1\rangle$支配。因此，我们得到了矛盾，从而证明了结论。◂

C.4 核心跨度器在天际线下的非闭包性

现在我们展示核心跨度器在天际线算子下不闭合。为此，我们引入核心跨度器中使用的字符串相等选择算子。对于任意两个变量 x和 y，我们将 $\zeta^{=}_{x,y}$ 写作变量 x 和 y，以表示定义于跨度器上的一元算子，其定义如下：给定文档 d上的跨度器 P，将该算子应用$\zeta^{=}_{x,y}P$于文档 d 上的 P，所得到的是由文档 d上的 P捕获的映射的一个子集，其中的映射满足： x 或 y 未定义，或者二者都被赋值为包含相同词的跨度。形式化地，$(\zeta^{=}_{x,y}P)(d) = \{m \in P(d) \mid x \notin \mathsf{dom}(m)$ 或 $y \notin \mathsf{dom}(m)$ 或 $d_{m(x)} = d_{m(y)}\}$。如 [25], 中一样，我们将字符串相等选择的记号扩展如下：对于每个变量集合 X，令 $(\zeta^{=}_{Y}P)(d) = \{m \in P(d) \mid \forall x, y \in Y$： $x \notin \mathsf{dom}(m)$ 或 $y \notin \mathsf{dom}(m)$ 或 $d_{m(x)} = d_{m(y)}\}$。然后，对于子集 $\mathbf{E} = \{Y_1, \ldots, Y_r\} \subseteq \mathbf{P}(\mathbf{SVars}(P))$，我们定义 $\zeta^{=}_{\mathcal{E}}(P) := \zeta^{=}_{Y_1}(\zeta^{=}_{Y_2}(\ldots\zeta^{=}_{Y_r}(P)\ldots))$（注意，应用字符串相等选择的顺序不会改变结果，因此这是定义良好的）。

我们定义字符串相等选择的方式遵循通常的语义：未赋值的变量代表缺失的信息，因此可能与任意值相容；另见 [25]，该文献也在无模式情形下采用此定义来处理字符串相等。然后，核心跨度器是指可以将正则跨度器作为基本情况，并迭代应用投影、并集、连接和字符串相等选择算子来定义的跨度器。

正如我们上文所定义的，核心跨度器是无模式的，因为它们提取的映射可能会赋值给不同的变量集合。相比之下， [10] 中核心跨度器的原始定义是基于模式的。为了明确这一差异，我们定义基于模式的核心跨度这一概念：

▸ **定义 C.1**。 一个基于模式的核心跨度器 是满足以下条件的核心跨度器：核心跨度器定义中的正则跨度器 P 都是基于模式的，即对于每次应用，它们的所有输出映射的定义域始终是该跨度器的定义域 $\mathsf{SVars}(P)$，对于每次应用 $\zeta^{=}_{\mathcal{E}}(P)$，我们都有 $Y \in \mathcal{E}$ 满足 $Y \subseteq \mathsf{SVars}(P)$，并且每当进行并集 $P_1 \cup P_2$ 时，都有 $\mathsf{SVars}(P_1) \cup \mathsf{SVars}(P_2)$。

容易通过归纳证明，按照此定义的基于模式的核心跨度器确实是基于模式的核心跨度器，即确保所有捕获映射都为跨度器定义域中的所有变量赋值。我们稍后将证明其逆命题（引理 C.4）。我们还指出，我们对基于模式的核心跨度器的定义与核心跨度器的传统定义一致，也就是 [10] 中的定义。

我们将证明以下结果：

▸**定理 4.4**。 对于跨度包含支配关系，核心跨度器在天际线算子下不封闭 $\preccurlyeq_{spanInc}$，即使对于基于模式的跨度器也是如此：存在一个基于模式的核心跨度器 P，使得 $\eta_{spanInc}P$ 无法表示为核心跨度器。对于采用变量包含支配规则的天际线 η_{varInc}，结论也相同。

我们将在本附录余下部分证明定理 4.4 。首先，我们做一些额外的准备工作。首先，我们将证明一个结果，它本质上是引理 A.2 针对核心跨度器的变体。请记住，对于跨度器 P 和变量集 X，跨度器 $P^{[X]}$ 在每个文档 d 上恰好捕获映射 $\{m \in P(d) \mid \mathsf{dom}(m) = X\}$。

▸ **引理 C.2**。 设 P 是一个核心跨度器， X 是一个变量集。那么 $P^{[X]}$ 可以表示为一个基于模式的核心跨度器。

证明中，我们将使用以下 核心简化引理，它出自 [25]，是针对基于模式的核心跨度器的类似结果的一个变体，见 [10]。

▸ **引理 C.3** ([25], 引理 2.2)。 对于每个核心跨度器 P，都存在一个正则跨度器 P_R，$\mathcal{E} \subseteq \mathcal{P}(\mathsf{SVars}(P_R))$ 以及 $Y \subseteq \mathsf{SVars}(P_R)$ 使得 $P = \pi_Y(\zeta^{=}_{\mathcal{E}}(P_R))$。

引理 C.2 的证明。 根据核心简化引理（引理C.3），我们可以假设 $P = \pi_Y(\zeta^{=}_{\mathcal{E}}(P_R))$，其中 P_R 是正则的。由引理 A.3，我们可以将其写作 $P_R = \bigcup_{X' \subseteq \mathsf{SVars}(P_R)} P_R^{[X']}$，其中所有 $P_r^{[X']}$都可以假设由函数变元自动机定义。根据定义，字符串相等选择与并集可交换，投影也与并集可交换，因此

$$
\begin{aligned}
P &= \pi_Y\left(\zeta^{=}_{\mathcal{E}}\left(\bigcup_{X' \subseteq \mathsf{SVars}(P_R)} P_R^{[X']}\right)\right) \\
&= \pi_Y\left(\bigcup_{X' \subseteq \mathsf{SVars}(P_R)} \zeta^{=}_{\mathcal{E}}(P_R^{[X']})\right) \\
&= \bigcup_{X' \subseteq \mathsf{SVars}(P_R)} \pi_Y(\zeta^{=}_{\mathcal{E}}(P_R^{[X']})).
\end{aligned}
$$

对于每个 X'，跨度器 $\pi_Y(\zeta^{=}_{\mathcal{E}}(P_R^{[X']}))$ 都是基于模式的，因为 $P_R^{[X']}$ 是基于模式的，而应用字符串相等选择和投影不会改变这一点。此外，$\mathsf{SVars}(\pi_Y(\zeta^{=}_{\mathcal{E}}(P_R^{[X']}))) = Y \cap X'$。因此，当且仅当 $X' \cap Y = X$ 时， $\pi_Y(\zeta^{=}_{\mathcal{E}}(P_R^{[X']}))$ 才会向 $P^{[X]}$ 贡献元组。由此可知

$$
P^{[X]} = \bigcup_{X' \subseteq \mathsf{SVars}(P_R):\ X' \cap Y = X} \pi_Y(\zeta^{=}_{\mathcal{E}}(P_R^{[X']}))
$$

现在，对每个 X' 定义集合 $\mathcal{E}_{X'} := \{Z \cap X' \mid Z \in \mathcal{E}\}$。那么，由于字符串相等选择会忽略未定义的值，并且 $P^{[X']}$ 捕获的映射恰好赋值为 X'，我们有 $\zeta^{=}_{\mathcal{E}}(P_R^{[X']}) = \zeta^{=}_{\mathcal{E}_{X'}}(P_R^{[X']})$，因此

$$P^{[X]} = \bigcup_{X' \subseteq \mathsf{SVars}(P_R):\ X' \cap Y = X} \pi_Y(\zeta^{=}_{\mathcal{E}_{X'}}(P_R^{[X']})).$$

我们断言，这是 $P^{[X]}$ 作为基于模式的核心跨度器的一种表示。事实上，

- 这些 $P^{[X']}$ 根据引理 A.3，对于每个应用 $\zeta^{=}_{\mathcal{E}_{X'}}(P^{[X']})$，我们对所有 $Z \in \mathcal{E}_{X'}$都有
- $Z \subseteq X' = \mathsf{SVars}(P^{[X']})$根据 $E_{X'}$的构造，且

- 在并集中，所有项都是定义域为 X 的跨度器。

因此，我们已将 $P^{[X]}$ 表示为正则核心跨度器，满足所需要求。 ◂

我们主要关注引理 C.2的以下推论。

▸**引理 C.4.** 设 P 是一个基于模式的核心跨度器。那么 P 可以按照定义 *C.1* 的含义，表示为一个基于模式的核心跨度器。

引理 C.4 直观地说，如果一个核心跨度器是基于模式的，即它的所有捕获映射都恰好赋值给同一个变量集，那么允许跨度器中出现无模式子项并不会提升该模型的表达能力。换句话说，如果一个基于模式的跨度器无法按照 [10]，所述的传统基于模式的含义表示为核心跨度器，那么在无模式设定下（即使整体跨度器是基于模式的，也允许其中出现无模式子项），它同样无法表示为核心跨度器。

现在，我们可以给出关于变量包含支配关系的定理4.4 的证明:

▸**命题 C.5.** 核心跨度器在变量包含支配关系 $\preccurlyeq_{varInc}$ 下不关于天际线算子封闭。

证明。我们固定字母表 $\Sigma = \{0, 1\}$。我们定义词 Σ^* 中的一个0-块为仅由0组成的极大子词，其中极大性是相对于子词包含而言的。跨度器 $P_{0\text{chunk}}$ 是布尔跨度器，即$\mathsf{SVars}(P_{0\text{chunk}}) = \varnothing$，当且仅当输入 d 以一个0-块结尾，且该块严格长于0中所有其他 d-块时，它提取空映射。已知 $P_{0\text{chunk}}$ 无法表示为基于模式的核心跨度器 [10]，因此根据引理C.4，它也无法表示为无模式核心跨度器。我们将证明，如果核心跨度器在变量包含意义下对天际线封闭，那么我们就可以将 $P_{0\text{chunk}}$ 写成一个核心跨度器，命题便可直接得出。

因此，设 $r_1 = z\{\}\Sigma^*$、$r_2 = z\{\Sigma\} \cdot \Sigma^*$ 和 $r_{\text{end}} = \Sigma^* \cdot x\{0^*\}$。最后，令 $\zeta^{\sqsubseteq}_{\{x,y\}}$ 为跨度算子，使得对于每个跨度器 P，$\zeta^{\sqsubseteq}_{\{x,y\}}(P)$ 都是一个跨度器，它在每个文档 d 上捕获映射 $\{m \in P(d) \mid d_{m(x)}$，其中 $\{m \in P(d) \mid d_{m(x)}$ 是 $d_{m(y)}\}$的子词。众所周知，基于模式的核心跨度器在 $\zeta^{\sqsubseteq}_{\{x,y\}}$ 下封闭，因此特别地，$r_3 = \zeta^{\sqsubseteq}_{\{x,y\}}(\Sigma^* y\{\Sigma^*\} \cdot \Sigma^* \cdot x\{0^+\})$ 是一个核心跨度器。（此处 0^+ 是 00^* 的缩写。）接着我们断言，

$$P_{0\text{chunk}} = \pi_{\varnothing}(r_2 \bowtie \eta_{varInc}((\pi_x(r_3) \bowtie r_1)) \cup r_{\text{end}}). \tag{1}$$

如果 d 是空文档，那么 $r_2(d) = \varnothing$，因此右侧不会捕获任何映射。这对于 $P_{0\text{chunk}}$也成立，因为0-块在 d 中不存在，因此根据定义，$P_{0\text{chunk}}$ 不会捕获任何内容。所以对于空文档，(1)成立。

26 用于文档跨度的天际线算子

现在设 d 非空，那么 r_1 将 z 映射到 $[0,\ 0\rangle$ 而 r_2 将它映射到 $[0,\ 1\rangle$。此外，r_3 将 x，y 映射到跨度 $m(x)$，$m(y)$，其中 $m(x)$ 是 d 末尾的一个跨度，仅由0组成；$m(y)$ 则使得 $d_{m(y)}$ 包含一个长度至少为 $|m(x)|$的0-块。因此，$\pi_x(r_3)$包含 d 中所有仅由 d 组成的后缀，且在 d的其他位置存在一个同样仅由0组成、长度至少相同的子词。特别地，如果 d 末尾有一个0-块 c，其长度超过 d中的所有其他0-块，那么 c 的跨度属于 $r_{\text{end}}(d)$，但不属于 $\pi_x(r_3(d))$。反之，如果不存在这样的块 c，那么 $r_{\text{end}}(d) \subseteq \pi_x(r_3)(d)$。集合 $(\pi_x(r_3) \bowtie r_1)) \cup r_{\text{end}}$ 包含 $r_1(d)$ 中的所有内容，以及 $(\pi_x(r_3))(d)$ 中的那些映射；此外，这些映射还将 z 映射到 $[0,\ 0\rangle$。对其应用天际线算子时，$\pi_x(r_3)(d) \cap r_{\text{end}}(d)$ 中的映射会被消除。因此，正如我们之前所说，当且仅当 d 末尾有一个严格长于文档中所有其他0-块的0-块时，天际线中才包含不对 z 赋值的映射。现在，由于 r_2 和 r_1 中对 x 的赋值不相容，与 r_2 连接后所得映射集合非空，当且仅当天际线中存在一个不对 z赋值的映射。因此，当且仅当 d 末尾有一个严格长于任何其他0-块的0-块时，(1)右侧才会捕获空映射。这就证明了(1)。

跨度器 r_1、r_2、r_3 和 r_{end} 都是核心跨度器。此外，核心跨度器在投影、连接和并集运算下封闭。因此，如果它们在变量包含意义下对天际线运算封闭，那么跨度器 $P_{0\text{chunk}}$ 就会是核心跨度器。这证明了该命题。◂

其次，我们证明跨度包含支配关系的结果：

▸命题*C.6*。*</style}*核心跨度器在跨度包含支配关系下不关于天际线算子封闭 $\preccurlyeq_{spanInc}$。

证明。为了证明这一结果，我们再次使用命题 $P_{0\text{chunk}}$ 的证明中的跨度器 C.5。我们将证明跨度器 $r_{0\text{chunk}}$ 属于核心跨度器在算子 $\zeta^{=}, \pi, \bowtie, \eta_{spanInc}$下的闭包。这足以证明该结果：由于核心跨度器在字符串相等、投影和连接下是封闭的，如果它们在跨度包含支配关系下也对天际线封闭，那么 $P_{0\text{chunk}}$ 就会是一个核心跨度器。但根据引理 C.4，它也可以表示为一个基于模式的核心跨度器，而这已在 [10]中证明是不成立的。

为了表达 $P_{0\text{chunk}}$，令 $r_1 = \eta_{spanInc}\pi_{\{x\}}\zeta^{=}_{x,\ y}(\Sigma^* y\{0^*\}\Sigma^+ \bowtie \Sigma^* x\{0^*\})$。（此处 Σ^+ 是 $\Sigma\Sigma^*$ 的缩写。）直观地说，r_1 选择一个映射，将 x 赋给词末尾的 0-块；该 0-块也会在词中的其他某处出现。换句话说，如果文档应被 $P_{0\text{chunk}}$ 接受，那么它将选择文档末尾最长 0-块的一个真后缀；否则，它将选择文档末尾的 0-块。为了区分这两种情况，令 $r_2 = \Sigma^* 0 z\{0^*\}$ 选择输入中仅包含 0 的一个非极大后缀，并令 $r_3 = \pi_{\varnothing}\zeta^{=}_{x,\ z}(r_1 \bowtie r_2)$。现在可以看出，当且仅当 r_1 选中了文档末尾 0-块的一个真后缀时，也就是当且仅当文档被 $P_{0\text{chunk}}$ 接受时，r_3 才会接受该输入词。因此，r_3 确实等价于 $P_{0\text{chunk}}$；并且根据构造，r_3 属于在字符串相等、投影、连接以及跨距包含支配关系下，对核心跨度器施加天际线算子后所得的闭包。这就完成了证明。◂

定理 4.4 由命题C.5 和 C.6得出。

D 第 5 节的证明（天际线算子的状态复杂度）

D.1 引理 5.3 的证明

▸**引理 5.3**。 设 P 是定义在变量集 SVars 上的一个正则跨度器，由一个具有 $\mathcal{A}$ 个 n 状态的顺序变量自动机表示。那么，对于每个文档 d，都存在一个非确定性只读分支程序 G 表示 $\mathit{Bool}(P, d)$ 具有 $O(|d| \times n \times |\mathrm{SVars}|)$ 个节点。

证明。 在此证明中，我们使用带有 ε-转移的变量自动机（参见附录 A）。此外，我们假定该变量自动机是 ε-无环的，即不存在 ε-转移构成的环或自环。可以在线性时间内对自动机施加这一条件。具体而言，考虑ε-转移，计算所得有向图的强连通分量。接着，对于每个强连通分量（状态集合），将相应状态合并为一个状态：若被合并的状态中有初始状态，则将新状态设为初始状态；若其中有终态，则将新状态设为终态；并添加转移，以反映从任一被合并状态出发能够执行的所有转移。最后，删除所有 ε-转移的自环。显然，这种变换不会影响自动机的语义，因为原自动机中的任何接受运行都能导出新自动机中的一个接受运行，该运行经过合并后的状态，不需要使用已删除的 ε-转移，并且以相同顺序读取相同的字母和标记；反过来，新自动机中的任何接受运行都可以通过沿已删除的 ε-转移移动而改写。注意，这种变换也不会影响自动机的顺序性，并且只会使状态数减少。

得益于添加 ε-转移，我们可以不失一般性地假设$\mathcal{A}$ 恰好有一个终态，并且该状态没有出边。事实上，只需添加一个新的终态，使其通过一条 ε-边从每个旧终态均可到达，并将所有旧终态设为非终态，即可在线性时间内实现这一点。显然，这不会改变抽取跨度，也不会影响顺序性或之前的变换。此外，状态数量仅增加 1：这不会妨碍我们建立大小界。

最后，我们假设该自动机是修剪的（参见附录A）。

在下文中，令 s 为 A 的初始状态，令 t 为 A 的终态。我们分几步构造 G。第一步的目的是去掉输入词 d，其核心做法是构造 A 和 d 的乘积：对于 G 的每个状态 x 以及 d 的每个前缀 p，我们创建一个新状态 (x, p)。我们按如下方式连接这些状态：如果 x 和 y 是 A 中的状态，且存在一条从 x 到 y 的 ε-边，那么对于 d 的每个前缀 p，我们添加边 ε $(x, p)(y, p)$。如果 A 中存在一条边 xy，其上读取了字母 a，那么对于每个前缀 p，只要 $p \cdot a$ 也是 d 的前缀，我们就添加边 $(x, p)(y, pa)$。在这两种情况下，如果边 xy 上有变量标记，我们也在新边上添加该操作。将所得有向图称为 G'。显然， G' 至多有 $|d|n$ 个顶点。令 $s' := (s, d)$ 和 $t' = (t, \varepsilon)$。那么， G' 中存在一条 s'-t'-路径，其变量标记对应变量集 $X \subseteq \mathrm{SVars}$，当且仅当 A 存在一个接受运行，为 X 中的变量赋予跨度。

我们现在注意到，由于初始变量自动机是 ε-环自由的，因此图 G' 是无环的。事实上，反设 G'中存在一个环。根据构造，这个环必定对应于变量自动机中的一个环。 G' 中的环只能涉及第二个分量为 d的某个给定前缀 p 的顶点，因为不存在从较长前缀指向较短前缀的 G' 边。因此，变量自动机中的环不可能包含任何字母转移。由于变量自动机是顺序的且经过修剪，我们知道该环不可能涉及标记；否则，我们就能从该环构造出一条接受运行，使同一个标记被赋值两次。因此，该环只由 ε-转移组成，但这与变量自动机是环自由的矛盾。

现在，我们通过将每个变量 $x \in \in$SVars对应的所有起始变量标记 $\vdash x$ 替换为文字 x_b，并删除其余所有标记，将 G' 转化为非确定性只读分支程序。将所得结果称为G''。同样，由于A是顺序的且经过修剪，对于 G' 中的每条 s'-t'路径，路径上的每个标记$\vdash x$ 后面都有一个标记 $\dashv x$ 。由此可知， G'' 恰好接受这样的赋值 m_b ： 存在A在 d 上的一条接受运行，使得对于所有被赋予跨度的变量 x ，均有 $m_b(x_b) = 1$。不过请注意，可能存在变量 x_b ，使得 $m_b(x_b) = 1$ 成立，但在该运行中 x 未被赋予跨度。为避免这种情况，我们必须在相应模型中，将 G'' 路径上未出现的所有变量强制设为0。为此，我们采用有序读取一次分支程序的常用补全技术的一种变体：对于每个节点 u，我们计算一个集合 $V(u)$ ，其中包含从源点 s 到 v的路径上的边所对应的所有变量。接下来，我们迭代处理每条边 uv：当存在一个变量 $y \in V(v) \backslash V(u)$ 不是 uv的标签时，就将 uv 替换为一条长度为二的路径 $uv_y v$ ，其中 v_y 是一条新边；若 uv_y 有标签，则为其赋予 uv 的标签，并为 $v_y v$ 赋予标签 $\neg y$。如此反复，最终得到一个新的非确定性只读分支程序 G ，其中，对于每条 s'-t'路径及每个变量 $x \in \in$ SVars，恰有一条边的边标签为 x_b 或 $\neg x_b$。此外， G'' 中的 π- s'路径与 G 之间存在双射，并且对于 G'' 中的每条路径 p ，满足：所有出现在 p上的变量也出现在 $\Pi(p)$ 上，且均为正向出现；所有其他变量均为负向出现。由此可知， G 计算Bool(P, d)。注意到 G''中的每条边都会产生一条长度至多为 $|\text{SVars}| + 1$ 的路径，即可得到规模上界并完成证明。◂

D.2 命题5.4 的证明

▸ **命题 5.4** ([7])。 对于任意 $n \in \mathbb{N}$，存在一个读取次数为 *3* 的单调 *2-CNF* 公式 Φ ，它包含 n个变量且大小为 $O(n)$ ，使得 Φ 的每一种非确定性只读分支程序表示的大小均为 $2^{\Omega(n)}$。

证明。 给定一个图 $G = (V, E)$ 其最大度为 3，我们构造一个读取次数为 3 的单调 2-CNF 公式 F_G ，其变量为 $\{x_v \mid v \in V\}$ ，如下

$$F_G := \bigwedge_{uv \in E} x_u \vee x_v.$$

在 [4, 定理 85], 中证明了，若 F_G 的最大度有常数上界，则计算 $2^{\Omega(\text{tw}(G))}$ 的任何非确定性只读分支程序的大小为 G 。这里 $\text{tw}(G)$ 表示 G的树宽。利用这样一个事实：存在最大度为 3、大小为 $O(n)$且树宽为 $\Omega(n)$ [16,的图，命题 1 和定理 5] 即可推出该断言。◂

D.3 定理5.5的证明

我们先在本节给出定理5.5的证明，然后在下一节给出定理5.1的证明。

该证明的灵感来自NP困难性证明，该证明针对的是对由两个正则表达式公式连接而成的跨度器进行求值 [22, 定理 3。1]。我们定义两个正则表达式 r_c 和 $r_=$ ，它们的连接将编码一个合取范式公式。设 $F = C_1 \wedge \ldots \wedge C_m$ 是一个变量取自 x_1、 $\ldots x_n$的 2-CNF 公式。我们使用命题5.4中的公式。提醒读者，这些公式是单调2-CNF，即所有子句都只包含两个变量，且所有变量都只以正形式出现。此外，每个变量只出现在三个子句中。我们首先定义 $r_=$ 为

$$r_= := r_{x_1} \cdot \cdots \cdot r_{x_n} \cdot a$$

其中 a 是一个字母，并且

$$r_{x_i} := \left(x_i^1\{\varepsilon\}x_i^2\{\varepsilon\}x_i^3\{\varepsilon\}\right) \vee \varepsilon.$$

注意，考虑在单字母文档 a 上对 $r_=$ 进行一次成功求值，对于每个 $i \in [n]$，求值要么将三个变量 x_i^1，x_i^2，x_i^3 映射到跨度 $[0,\ 0\rangle$，要么不映射这三个变量中的任何一个。这三个变量必须以相同的方式匹配，这一事实将使我们稍后能够一致地模拟对 x_i 的变量赋值。

接下来，我们通过如下设置来构造 r_C

$$r_C = a \cdot r_{C_1} \cdot \ldots \cdot r_{C_m}$$

其中

$$r_{C_j} := x_{j_1}^{p(j,j_1)}\{\varepsilon\} \vee x_{j_2}^{p(j,j_2)}\{\varepsilon\}$$

其中，x_{j_1} 和 x_{j_2} 是 C_j 中的两个变量，而 $p(j,\ j_2)$ 是数字 $s \in \{1,\ 2,\ 3\}$，使得从左向右读取时，x_j 在 C_j 中的出现是 x_j 在 F 中的第 s 次出现。注意，这一定义良好，因为每个变量至多出现在三个子句中。

现在考虑跨度器 $r := r_= \bowtie r_C$ 在输入 $d = a$ 上的结果。当且仅当对于 F 的每个子句 C_j，都存在一个变量 x_b 属于 C_j，使得 $x^{p(j,}_{\ \ b)}$ 未被 m 映射到 $[0,\ 0\rangle$，映射 m 才属于 $r(a)$。

设$\mathcal{A}$ 是一个序列变量集自动机，对于 r，不失一般性地假设它是修剪的，并再次考虑输入 $d = a$。那么，对于$\mathcal{A}$的每个状态 s，我们都能确定：在$\mathcal{A}$ 的一次接受运行中，是否可以在读入字母 a 之前或之后到达状态 s（注意，在不同的运行中，同一状态 s 不可能既在读入该字母之前又在读入之后到达：由于该变量自动机是顺序的且经过修剪，这将意味着该自动机可以接受包含多个 a 的文档，而 r 不允许这种情况）。现在，对于所有从此前已读入 a 的状态出发的边，删除所有变量标记。根据前述内容，所得自动机$\mathcal{A}$ 恰好接受这样的映射 m：存在 F 的一个满足赋值 m_b，使得 m 将变量 $\{x_i^j \mid m_b(x_i) = 0\}$ 赋值为.....。现在，删除变量 x_i^2 和 x_i^3 中的所有变量标记，并将所有 x_i^1 替换为 x_i，即可得到一个变量自动机$\mathcal{A}'$，它表示一个跨度器 P'，使得 $\mathrm{Bool}(P', a) = \{1 - m_b \mid m_b$是 $F\}$ 的一个模型；也就是说，它是 F 的模型，至多相差对满足赋值中的所有比特进行翻转。注意，这个变量自动机$\mathcal{A}'$的大小与$\mathcal{A}$呈线性关系。

现在应用引理 5.3 于 $\mathcal{A}'$，我们得到一个计算 Bool G 的非确定性只读分支程序$(P',\ a)$ 因此，通过翻转 G中所有变量的符号，我们得到一个计算 G' 的非确定性只读分支程序 F。此外，G' 的大小为 $O(|\mathcal{A}|n)$。但是，根据 F这一合取范式的选取，我们由命题 5.4 知，任何编码 F 的非确定性只读分支程序大小均为 $2^{\Omega(n)}$，证明完毕。

D.4 定理的证明 5.1

接下来我们证明定理5.1，该定理说明，在变量包含支配规则下，天际线算子通常会导致指数级膨胀。直观而言，这一结果可视为一种证明变体，用以证明计算该算子在组合复杂度上是 NP-困难的（定理 6.3），该定理将在本文后文给出；此证明也采用了与前文定理 5.5 证明相似的思路。

我们再次固定一个单调 2-CNF公式 F，它有 n 个变量 $\{x_1,\ \ldots,\ x_n\}$ 并且我们假设每个变量只出现在三个子句中。令 $C_1,\ \ldots,\ C_m$ 为F的子句。我们选择 F，使得任何针对 F 的非确定性只读分支程序大小为 $2^{\Omega(n)}$，根据命题 5.4这是可能的。

30 用于文档跨度的天际线算子

现在考虑通过略微修改定理 6.3 的证明中的两个跨度器所定义的两个跨度器。为此，令 T_i 表示 x_i 出现的子句的索引；请记住， F 是单调的，因此变量总是以正形式出现。此外，为每个 $i \in [n]$ 引入三个跨度器变量 $v_{i,j}$，每个 $j \in T_i$ 对应一个，并额外引入一个变量 $\bar{x}_i$。然后我们设：

$$r_{\text{valid}} := \cdot_{i\in[n]}((x_i\{\varepsilon\} \cdot \cdot_{j\in T_i} v_{i,j}\{\varepsilon\}) \vee \bar{x}_i\{\varepsilon\}).$$

直观而言，在每次匹配中，跨度器都会将跨度 $[0,0\rangle$赋给某些 x_i ，我们将其解释为在合取范式中把变量 x_i 设为真。此外，将 x_i 设为真后成立的子句，通过将它们赋值为 $[0,0\rangle$也会成立。最后，对 $\bar{x}_i$ 的赋值编码了对 x_i 的赋值取反。

对于 r_{mask} ，我们设：

$$r_{\text{mask}} = \bigvee_{k\in[m]} (\cdot_{i\in[n]}(x_i\{\bar{x}_i\{\varepsilon\}\}(\cdot_{j\in[T_i]\setminus\{k\}} v_{i,j}\{\varepsilon\})).$$

注意， r_{mask} 在输入 ε 中恰好有 m 种匹配：在这些匹配中的每一个里，所有变量$x_i, \bar{x}_i$ 都被赋值 $[0,0\rangle$。此外，在第 k个匹配中，所有$v_{i,j}$ 除了形如$v_{i,k}$ 的那些之外，都被赋值。

那么，完备的正则跨度器为 $r := r_{\text{mask}} \vee r_{\text{valid}}$，我们在 r 上应用天际线算子η_{varInc}。我们考虑该跨度器在空文档 ε上的求值。

我们首先研究这种跨度器所捕获的映射之间可能存在何种支配关系。首先注意， r_{valid} 的匹配不会彼此支配，因为在每个匹配中，对于每个 $i \in [n]$ ，要么赋值了 x_i ，要么赋值了 $\bar{x}_i$ ，而这些赋值各不相同。同样， r_{mask} 的匹配也无法彼此支配，因为对于 k 的不同取值，它们不会赋值$v_{i,k}$ ，因此两两不可比较。最后，由于 r_{mask} 的所有匹配都会赋值所有 x_i 和 $\bar{x}_i$，它们都不会被 r_{valid}的任何匹配支配。因此，唯一可能发生的支配关系是 r_{mask} 的匹配可以支配 r_{valid}的匹配。此外，当且仅当 r_{valid} 的匹配 m 满足如下条件时，这种情况才会发生：存在一个索引$k \in [m]$ ，使得没有形如$v_{i,k}$ 的变量被赋值；也就是说，存在 F 的一个子句 $C_k = x_{i_1} \vee x_{i_2}$，使得在相应的赋值中，两个变量 x_{i_1} 和 x_{i_2} 都没有被赋为真。换言之，当且仅当 r_{valid} 的映射所描述的 x_i的赋值不满足合取范式 F时，它才会被 r_{mask} 的映射支配。

现在考虑函数 f，即 $\text{Bool}(\eta_{varInc} r, \varepsilon)$ 投影到变量 $x_1, \ldots, x_n$。该函数描述了 $x_1, \ldots, x_n$ 的哪些子集在映射中被赋予一个跨度（必然是 $[0,0\rangle$）。我们断言，这些恰好是满足合取范式 F的赋值。事实上，被捕获的赋值包括全为 1 的赋值（由 r_{mask}捕获），而它确实是 F的满足赋值；此外还包括 r_{valid} 的所有不被 r_{mask}中的某个赋值支配的赋值，根据我们前面的讨论，这些赋值满足 F。反过来，对于 F 的每个满足赋值，我们都能得到一个不被 r_{mask} 中的映射支配、且属于输出的 r_{valid} 映射（注意，全为 1 的赋值既由 r_{mask} 捕获，也由 r_{valid}捕获）。因此，函数 f 实际上恰好就是 F。

最后，固定一个变量集自动机 $\mathcal{A}$ ，它表示 $\eta_{varInc} r$。然后利用引理 5.3 以及投影不会增大非确定性只读分支程序的规模这一事实，我们得到一个非确定性只读分支程序 G ，用于表示 F ，其规模为 $O(|\mathcal{A}|n)$。由于我们选择了 F ，使得非确定性只读分支程序G 的规模至少为 $2^{\Omega(n)}$，因此可得 $|\mathcal{A}| = 2^{\Omega(n)}$ ，由此完成定理 5.1的证明。

E 证明（天际线算子的6 复杂性）

▸ **引理6.4**。设 f： $\Sigma^* \to \Sigma^*$ 并设 p 为一个多项式。假设给定输入NP-难的，给定一个输入 x 和整数 $k \leqslant p(|x|)$，判断是否 $|f(x)| < k$。那么，除非 f，否则不存在P $=$ NP。

证明。 反证，假设存在一个输出多项式时间算法，用于f计算 $f(x)$ 其运行时间为 $r(|x| + |f(x)|)$ 对于某个多项式 r。不失一般性地，假设 r 是单调的。于是我们证明存在一个用于判定问题的多项式时间算法，引理即可直接得证。

算法如下：给定 x 和 k，我们对 $r(|x| + k + 1)$ 模拟输出多项式时间算法的执行步数。如果模拟在这么多步内终止，我们只需检查 $|f(x)| \geqslant k$ 是否成立。如果模拟未终止，我们就拒绝该输入。

我们断言，上述算法是引理陈述中的判定问题的多项式时间算法。首先注意到，对于运行时间而言，只需证明我们仅对输出多项式时间算法进行多项式数量步骤的模拟即可。但这是正确的，因为 $r(|x| + k + 1) \leqslant r(|x| + p(|x|) + 1)$ 在 $|x|$中是多项式有界的。

还需证明其正确性。首先，如果模拟在允许的步数内终止，那么输出显然正确。因此，假设模拟不会终止。如果 $f(|x|) < k$，那么根据假设，输出多项式时间算法会在 $r(|x| + |f(x)|) \leqslant r(|x| + k)$ 步内终止，因此模拟必定终止。因此，从模拟不会终止这一事实可以推断出 $|f(x)| \geqslant k$，所以拒绝是正确的。◂

▸ **推论 6.6**。 假设 P $\neq$NP，即使支配关系由一个顺序变量自动机给出（而不是通过隐式地取单变量顺序 *VA* 的乘积），也不存在一个在组合复杂度方面具有输出多项式时间的天际线提取问题算法。

证明。 检查定理6.3的证明后，我们注意到，即使在空文档上，变量包含支配关系也已具有困难性。幸运的是，我们可以在关于变量的定义域 X 内，用多项式时间构造一个支配规则，使其在空文档上表示这一支配关系。直观地说，捕获映射应当是这样的映射：将左侧变量的某个子集 $X' \subseteq X$ 赋给一组值，并将 X' 的一个超集作为右侧变量。形式化地，我们可以取如下支配规则：

$$\cdot_{x \in X} \left(\varepsilon \vee x^{\dagger}\{\varepsilon\} \vee x\{x^{\dagger}\{\varepsilon\}\}\right)$$

（请注意，这只适用于空文档，而且关键在于：每个位置上的标记都可以按任意顺序排列，只要对每个变量而言，开标记都在闭合标记之前。如果文档非空，我们就无法在连续变量之间进行连接。）

因此，给定一个合取范式公式，我们按照定理 P 的证明中的方法构造一个跨度器6.3，如上所述，在多项式时间内计算所需变量集合上的支配规则，并像先前的证明一样，利用引理6.4得出结论。◂

F 第7 节证明（难处理和可处理的支配规则）

F.1 用于以顺序变量自动机表示的支配规则的 UMDSDP

本节将证明，对于以顺序变量自动机表示的支配规则，UMDSDP 条件无需要求这些支配对能够被高效计算，只需要求它们存在即可。我们将定义一个比 UMDSDP 更弱的条件，其中不设效率要求：

32 文档跨度的天际线算子

▸**定义 F.1**。如果对于任意 $n \in \mathbb{N}$，都存在一个文档 d 和 n 个 $D(d)$ 的严格支配对 S_1、...、S_n，且这些支配对两两不相交，则称单变量支配规则 D 弱 UMDSDP 。

因此，UMDSDP 单变量支配规则 D 尤其是弱 UMDSDP。反过来，如果 D 是弱 UMDSDP，且我们能够高效计算文档和严格支配对，那么它就是 UMDSDP。

我们的精确断言是，对于以变量自动机表达的支配规则，弱 UMDSDP 蕴含 UMDSDP：

▸**命题 F.2**。令 D 为由顺序变量自动机A 表示的单变量支配规则。如果 D 是弱 *UMDSDP*，则它是 *UMDSDP*。

为此，我们还需要关于有限自动机（不带标记转换）的一个附加断言；这个断言有些晦涩，也有些难以证明。我们将 Σ 称为字母表，并称在 Σ 上、其转移函数为全函数的确定性有限自动机是完备的，即对于每个状态和字母 $a \in \Sigma$，只有一个出转移。通过添加一个汇状态，可以在不失一般性的情况下在线性时间内实现这一点。于是我们有：

▸**引理 F.3.** 对于任意确定性的完备有限自动机 A，若其状态集合 Q 的大小为 k，则存在一个仅取决于 k 的数值 K，使得以下结论成立：设 $\delta\colon Q \times \Sigma \to Q$是 A 的（全）转移函数，并以预期的方式将其扩展为定义在词上的函数 $\delta^*\colon Q \times \Sigma^* \to Q$。令 $w_1, \ldots, w_{K+1}$ 为任意字符串，$q_1, \ldots, q_K \in Q$ 为任意状态。那么存在指标 $1 \leqslant s < t \leqslant K+2$，满足 $s < t$；令$q := \delta^*(q_s, w_{s+1} \cdots w_{t-1})$，则有 $\delta^*(q, w_s \cdots w_{t-1}) = q$。

为了证明引理 F.3，我们将使用拉姆齐定理的以下变体，证明见例如 [15]。

▸**定理 F.4.** 对于任意一对 $k, r \in \mathbb{N}$，都存在整数 $R(k,r)$，使得对于一个至少有 $R(k,r)$ 个顶点的完全图的边的任意 r-着色，都存在一个大小为 k 的单色团，即一个所有边颜色都相同的团。

引理的证明 F.3。令 C 为从 Q 到 Q的函数集合。注意到 C 是有限的，并且 $|C| = k^k$ 仅取决于 k。对于每个词 $w \in \Sigma^*$，我们将其关联到一个函数 f_w，该函数从 C 定义如下：对于每个状态 $q \in Q$，令 $f_w(q) := \delta^*(q, w)$，也就是说，从状态 q 启动自动机并读取 w[2]后到达的状态。根据定理 $K > R(k+2, |C|)$我们选取 F.4。

现在考虑引理陈述中的输入词 w_1、.....、w_{K+1}。我们定义一个以 $K+1$ 个顶点为顶点的完全图，其中每条边 $\{i, j\}$ 对应于词 $w_i \cdots w_{j-1}$在 C中的元素 $f_{w_i \cdots w_{j-1}}$。根据 K 的选择以及定理 F.4的陈述，该图包含一个单色 $(k+2)$-团，因此存在一种颜色 $f \in C$ 以及位置 $i_1 < \cdots < i_{k+2}$，使得对于每个 $1 \leqslant a < b \leqslant k+2$，令 $w_{a,b} = w_{i_a} \cdots w_{i_b - 1}$，则有 $f_{w_{a,b}} = f$。为证明该断言，我们取 $s := i_1$ 和 $t := i_{k+2}$，并相应地定义$q := \delta^*(q_s, w_{s+1} \cdots w_{t-1})$。

我们来证明引理陈述中的等式。考虑运行过程 ρ：根据定义，自动机从状态 q_s 开始，读取词 $w_{s+1} \cdots w_{t-1}$，并到达状态 q。考虑此运行中，在读取因子 $k+1$ w_{i_2}、.....、$w_{i_{k+2}}$ 之前到达的状态。根据鸽巢原理，同一状态 q' 重复出现了两次，这意味着存在某个

[2] 我们指出，这是代数自动机理论中的 变换幺半群。

状态 q'，使得读取 $w_{a,\,b} = w_{i_a} \cdots w_{i_b-1}$时，我们从状态 q' 转移到状态 q'，也就是说，$\delta^*(q', w_{a,\,b}) = q'$ 对于某个 $2 \leqslant a < b \leqslant k+2$成立。现在，定理 F.4 的应用保证，对于任意 $1 \leqslant a' < b' \leqslant k+2$，都有 $\delta^*(q', w_{a',\,b'}) = q'$。特别地，考虑因子$w_{b,\,k+2} = w_{i_b} \cdots w_{i_{k+2}-1}$，我们得到如下结论（*）：$\delta^*(q', w_{b,\,k+2}) = q'$。

现在，运行 ρ从 q_s开始并读取 $w_{s+1} \cdots w_{t-1}$时，会执行以下操作：首先读取$w_{s+1} \cdots w_{i_b-1}$，最终到达状态 q'；然后读取 $w_{b,\,k+2}$，并根据上文的 **(*)** 到达状态 q'。但根据定义，在 $w_{s+1} \cdots w_{t-1}$上从 q_s开始时，自动机会到达状态 q，因此可得 $q = q'$。现在考虑词 $w_s \cdots w_{t-1} = w_{1,\,k+2}$，我们再次知道$\delta^*(q', w_{1,\,k+2}) = q'$，因此实际上 $\delta^*(q, w_{1,\,k+2}) = q$，这正是我们需要证明的。◂

现在我们证明本节的主要结果。

命题的证明F.2. 不失一般性，我们假设$\mathcal{A}$ 是输入输出确定的，也就是说，对于输出中的每个映射，都恰好存在一个见证该映射的接受运行，且在该运行中，每个状态都只有一种可能的后续转移。文献 [12] 的定理 3.1 说明了如何从一般的变量自动机构造此类变量自动机[3]。需要指出的是，这种转换可能会使自动机的规模呈指数级增长，因此，特别是其运行时间相对于 $|\mathcal{A}|$而言是指数级的。不过，这并不构成问题：断言该支配规则是 UMDSDP，指的是能够根据输入数 n，以易处理的方式计算出一个见证文档和严格支配对，而定义该支配规则的变量自动机规模是常数。类似地，下面有些论证并非构造性的，这也不会造成问题：我们只证明 d 存在，并且存在一个多项式时间算法来计算它，但并未实际说明如何从$\mathcal{A}$推导出该算法。

令 Q 为$\mathcal{A}$ 的状态集合，并令 k 为 Q的大小。令 K 为引理 F.3 给出的数（该引理适用于自动机而非变量自动机，但我们稍后会解释如何在当前语境中使用它）。由于 $\mathcal{A}$ 定义的规则是弱 UMDSDP，因此存在一个文档 d包含 $(k+1) \times (K+2) + 3$ 个两两不相交的严格支配对：我们将它们记为 $S_1, \ldots, S_{(k+1)\times(K+2)+3}$，并按照覆盖它们的最小跨度的左端点递增排列；注意，这些跨度两两不相交。

考虑 d 的索引 $i_1, \ldots, i_{(k+1)\times(K+2)}$，它们是覆盖各对 $S_1, \ldots, S_{(k+1)\times(K+2)}$ 的跨度的右端点。也就是说，对于每个 $1 \leqslant j \leqslant (k+1) \times (K+2)$，索引 i_j 是 d 中字母的位置；在读完对 S_j 的最后一个标记后，自动机读取的第一个字母转换所读取的就是该字母。接着定义 q_j 为见证运行 ρ 在读取位置 i_j 处的字母之前、针对最后一对 $S_{(K+2)\times(k+1)+3}$ 到达的状态。由于 $\mathcal{A}$ 是输入输出确定的，我们知道，在到达读取位置 $i_{(k+1)\times(K+2)}$ 处的字母之前，运行不会读取任何变量标记；事实上，ρ 的所有标记都是在读取严格支配对 $S_{(K+2)\times(k+1)+3}$ 时赋值的，而我们知道，该严格支配对的覆盖跨度起点位于支配对 $S_{(k+1)\times(K+2)}$ 的右端点 $i_{(k+1)\times(K+2)}$ 之后。这里，我们利用这样一个事实：三个支配对 $S_{(K+2)\times(k+1)+1}$、$S_{(K+2)\times(k+1)+2}$、$S_{(K+2)\times(k+1)+3}$ 中，至多有两个严格支配对赋予空跨度，因此最后一对的起点确实必定位于 $i_{(k+1)\times(K+2)}$ 之后。于是，借助输入输出确定性，q_j 实际上是 $\mathcal{A}$ 读取文档 d 直到位置 i_j（不含该位置）且不赋予任何标记时所能到达的唯一状态。

根据鸽巢原理，存在状态 q 和索引 $K+2$ $j_1, \ldots, j_{K+2}$，使得对于每个 $1 \leqslant a \leqslant K+2$，状态 q_{j_a} 都是 q。

[3] 我们指出，严格来说 [12] 展示的是一种转换，转换后的形式称为确定性扩展变量自动机，但这些自动机可以轻易地转换为本文意义下的变量自动机，同时保持输入输出确定性。

34 文档跨度的天际线算子

现在相应地分解 $d = d_1 \dots d_{K+3}$，即对于每个 $2 \leqslant a \leqslant K+2$， d_1 是 d 从位置 j_1（不包括）之前的前缀， d_a 是 d 从位置 j_{a-1}（包括）到位置 j_a（不包括）的因子，而 d_{K+3} 是 d 从位置 j_{K+2}（包括）开始的后缀。我们对因子的选择确保，当不分配任何标记时，$\mathcal{A}$ 在读取 d_1 时会从初始状态转移到 q；对于任意 $2 \leqslant a \leqslant K+2$，**(*)** 在读取 d_a 时会从 q 转移到 q。（它也可以通过读取 d_{K+3} 从 q 完成运行，为这一对 $S_{(K+2)\times(k+1)+3}$ 分配标记；但我们不会使用这种方式。）

现在，对于每个 $2 \leqslant a \leqslant K+2$，考虑将支配对 S_{j_a-1} 指派给 ρ_a 的运行。运行 ρ_a（1.）从初始状态出发，在读取 $d_1 \cdots d_{a-1}$ 时到达状态 q；随后（2.）读取 d_a，其最后一对为 S_{j_a-1}，并在读取该对时指派标记。令 q'_a 为后续字母转移之前到达的状态，也就是说，该字母转移读取位置 i_{j_a} 处的字母。我们知道（3.）自动机可以完成该运行，即 ρ_a 从状态 q'_a 继续读取文档 $d_{a+1} \cdots d_{K+3}$，且不指派任何标记，最终到达一个接受状态。

因此，定义 $\mathcal{A}'$ 为通过删除所有标记转移而得到的确定性完备有限自动机$\mathcal{A}$，并对 A′ 应用引理 F.3 （将所有索引统一偏移 *1′* 即可），并使用引理 F.3。我们知道，对于某个 $2 \leqslant s < t \leqslant K+2$，考虑（**）状态$q''$（在引理 qF.3的陈述中称为） $d_{s+1} \cdots d_{t-1}$，它是从状态 q'_s 读取 $d_s \cdots d_{t-1}$ 后到达的$\mathcal{A}'$（直观地说，就是刚指派完标记之后），则（***）在 A′ 中读取 $d_s \cdots d_{t-1}$ 会从 q'' 到达 q''。因此，回到 $\mathcal{A}$，我们知道如下结论：

1. 运行 ρ_s 从初始状态转到状态 q，同时读取 $d_1 \cdots d_{s-1}$（根据上文 (1.)）；2. 运行 ρ_s 随后在读取 $d_s \cdots d_{t-1}$ 时为配对 S_{j_s-1} 赋予标记，并转到状态 q''（根据上文 (2.) 中关于 d_s 的内容，以及上文 (**) 中关于 $d_{s+1} \cdots d_{t-1}$ 的内容）；3. 运行 ρ_s 随后结束，且不赋予任何变量（根据上文 (3.)）。4. 变量自动机 A 从状态 q'' 开始，在读取 $d_s \cdots d_{t-1}$ 且不赋予任何标记时，转到状态 q''（根据上文 (***)）；5. 变量自动机 A 从状态 q 开始，在读取 $d_s \cdots d_{t-1}$ 且不赋予任何标记时，转到状态 q（根据上文 (*)）。

现在我们终于可以将 d 定义为 $d = uvw$，其中 v 由子词 $d_s \cdots d_{t-1}$ 构成，而 u 和 w 分别是位于其前后的前缀和后缀。

我们断言，对于任意 $n \in \mathbb{N}$，文档 $uv^n w$ 上的变量自动机 $\mathcal{A}$ 具有 n 个彼此不相交且易于计算的严格支配对。更具体地说，在 v 的每个副本中，因子 v^n 内都存在一个支配对，其位置对应于支配对 S_{j_s-1} 的端点。如果这点成立，那么它就证明了 $\mathcal{A}$ 的支配规则是 UMDSDP：给定一个整数 $\mathbb{N}$，我们可以在关于 n 的多项式时间内计算文档 $uv^n w$。此外，通过对 u 和 v 的长度以及所考虑的支配对在 v 中的位置进行一些算术运算，我们可以在多项式时间内轻松计算出 n 个支配对。

为了解释为什么我们有 n 对，任选 $n \in \mathbb{N}$，再选取一个索引 $1 \leqslant i \leqslant n$，并说明我们为何可以得到支配对。根据构造，当 $\mathcal{A}$ 读取前缀 u 且不分配任何标记时，它会转到状态 q（上文第 1 点）。此外，当它读取 v且不分配任何标记时，也会转到 q（上文第 5 点）。因此，对于任意 $\ell \in [n]$，读取 $v^{\ell-1}$ 时也同样如此。现在我们知道，可以读取 v，并分配与该因子内某个严格支配对相对应的标记，然后转到状态 q''（上文第 2 点）。此外，读取 v 且不分配任何标记，会使其从状态 q'' 转到状态 q''（上文第 4 点），从而让我们能够继续运行，期间不进行任何标记转换，直到

我们已经读完 v的最后一份副本；最后，我们无需分配任何标记即可读取 w 并完备运行，使其成为接受运行（上面的第 3 点）。证明至此结束。◂

F.2 UMDSDP 条件下的困难性证明

▸**定理7.5**。设 D 是满足 *UMDSDP* 的单变量支配规则。对于 D，给定一个顺序变量自动机$\mathcal{A}$ 和一个文档 $d \in \Sigma^*$，除非P == NP，否则天际线提取问题不是输出多项式的。

证明。与定理6.3的证明一样，我们再次从合取范式公式的可满足性问题进行归约。

设 D 是满足 UMDSDP 的逐变量支配规则。令 F 为一个合取范式公式，其中包含 n_v 个布尔变量 x_i 和 n_c 个子句 C_j。我们将说明如何将 F 的可满足性归约为：计算某个适当构造的正则表达式公式相对于规则 D 的天际线。为此，我们将在变量 $v_{i,\ j}$ 上构造一个跨度器，用于 $i \in [n_v]$ 和$j \in [n_c]$。我们定义集合 $T_i := \{j \mid x_i \text{ 在 } C_j\}$ 中正向出现，以及 $F_i := \{j \mid x_i \text{ 在 } C_j\}$ 中负向出现。

该证明的思路与定理 6.3 类似，我们先在此再次概述：定义两个跨度器 r_{valid} 和 r_{mask}。跨度器 r_{valid} 将再次为每个可能的合取范式变量赋值m 捕获一个映射 a。每个映射 m 都会编码在赋值 a下哪些变量 x_i 使哪些子句C_j 得到满足。第二个跨度器 r_{mask}将再次捕获 n_c 其他映射 m_j，其中每个映射都支配由r_{valid} 捕获的所有映射 m，这些映射编码了不满足子句 C_j的赋值。这些映射 m_j共同支配由 r_{valid} 捕获的、编码不满足赋值的所有映射。因此，当且仅当合取范式公式 F 可满足时，$r_{\text{valid}} \vee r_{\text{mask}}$的天际线中才存在由 r_{valid} 捕获的映射 m。

现在我们给出构造的细节。首先计算将要使用的输入文档和跨度。由于 D 满足 UMDSDP，我们可以在 n_v 上的多项式时间内计算出文档 $d \in \Sigma^*$ 以及两两不相交的严格支配对 (s_1^1, s_2^1)、(s_1^2, s_2^2)、$\ldots$、$(s_1^{n_v}, s_2^{n_v})$。我们将文档 d 切分为 n_v 个不相交的部分 d_1、$\ldots$、d_{n_v}，使得对于每个 $i \in [n_v]$，部分 d_i 都包含严格支配对 (s_1^i, s_2^i)。这是可行的，因为根据定义，(s_1^i, s_2^i) 彼此不重叠。设 s^i 为给出 d_i边界的跨度，即 $d_{s^i} = d_i$。注意，对于所有 $i \in [n_v]$，若跨度s_1^i 和s_2^i有定义，则它们包含在跨度 s^i 中。

现在我们构造这些跨度器，从 r_{valid} 开始。给定一个变量索引 $i \in [n_v]$、一组子句索引 $I \subseteq [n_c]$和一个索引 $b \in \{1, 2\}$，我们首先定义跨度器 $r^i(I, b)$。该跨度器将读取词 d_i，并根据支配对中的值 s_b^i 为 $j \in I$ 到 $v_{i,\ j}$ 的变量赋值。形式化地说，如果未定义 s_b^i，那么我们读取 d_i 而不进行任何赋值：

$$r^i(I, b) := d_i$$

并且如果 s_b^i 有定义，那么对于定义 d_i 的跨度，记为 $s_b^i = [p, q\rangle$，记为 $[p', q'\rangle$，我们有：

$$r^i(I, b) := d_{[p', p\rangle} \left(\bowtie_{j \in I} v_{i,j}\{d_{[p,q\rangle}\}\right) d_{[q, q'\rangle})$$

注意，此处的连接很容易针对每个具体的 s_b^i 表达，因为所有变量$v_{i,j}$ 与$j \in I$ 都映射到同一跨度。

现在，为了构造 r_{valid}，我们定义用于 $i \in [n_v]$ 的跨度器 r^i_{valid}，它们在读取 d_i 时分别捕获两个映射 m_t，m_f。映射 m_t（分别为 m_f）编码子句 C_j

当 x_i 被赋值为真（分别为假）时满足，如下所示：

$$r^i_{\text{valid}} := (r^i(T_i, 2) \bowtie r^i([n_c]\backslash T_i, 1)) \vee (r^i(F_i, 2) \bowtie r^i([n_c]\backslash F_i, 1)).$$

同样，对于每种具体情况，构造都很简单，因为所有相关变量都只映射到两个不同的跨度 s^i_1 和 s^i_2，因此可以轻松地将 r^i_{valid} 表示为 VA，并且也能在多项式时间内计算出来。我们将跨度器 r^i_{valid} 连接起来，得到 r_{valid}，它读取整个文档 d 并编码布尔变量的所有可能赋值，因此

$$r_{\text{valid}} := r^1_{\text{valid}} \cdot \ldots \cdot r^{[n_v]}_{\text{valid}}.$$

由于所有 r^i_{valid} 都能在多项式时间内构造出来，而且 VA 很容易表示连接，因此 r_{valid} 是一个正则跨度器，其 VA 可以在多项式时间内计算出来。

为了从映射 m 构造合取范式变量赋值，该映射由 r_{valid} 捕获，我们将使用以下编码：如果映射 m 将变量 $v_{i,\,j}$ 赋给后续的 s^i_2（即，如果这是一个跨度，则赋给该跨度；如果未定义，则完全不赋值），那么变量 x_i 就会出现在子句 C_j 中，并且 x_i 的赋值会使该子句 C_j 为真。得益于 r^i_{valid} 的构造方式，该映射给出了对任意给定变量的所有出现位置进行赋值的一致选择。

现在来构造 r_{mask}。我们定义用于 $j \in [n_c]$ 的跨度器 $r^{j\ \text{mask}}$，它们读取 d，并且各自捕获一个映射 m^j。映射 m^j 几乎处处都将按照 s^i_2 赋值，只有变量 $v_{i,k}$ 除外；对于这些变量 $k = j$，将按照 s^i_1 赋值。回忆一下，s^i_2 支配 s^i_1。

$$r^j_{\text{mask}} := \cdot_{i\in[n_v]}(r^1(j, s^1_1) \bowtie r^1([n_c]\backslash j, s^1_2)) \cdot \ldots \cdot (r^{[n_v]}(j, s^{[n_v]}_1) \bowtie r^{[n_v]}([n_c]\backslash j, s^{[n_v]}_2))$$

其中，与之前一样，$\cdot$ 表示连接。如上所述，$r^i(j, s^i_1)$ 和 $r^i([n_c], s^i_2)$ 可以在多项式时间内转换为变量自动机，因此也可以在多项式时间内构造一个用于 $r^{j\ \text{mask}}$ 的变量自动机。我们对跨度器 $r^{j\ \text{mask}}$ 取并集，得到

$$r_{\text{mask}} := \bigcup_{j\in[n_c]} r^j_{\text{mask}}$$

它读取整个文档 d。r_{mask} 捕获的映射 n_c 共同支配所有由 r_{valid} 捕获的映射，这些映射编码了一个不满足赋值。同样，这可以高效地编码为变量自动机，因此 $r_{\text{valid}} \vee r_{\text{mask}}$ 也可以在多项式时间内编码为变量自动机。

现在来证明该归约正确，也就是说，F 可满足，当且仅当天际线 $r_{\text{valid}} \vee r_{\text{mask}}$ 中存在一个由 r_{valid} 捕获的映射 m。首先假设 F 可满足。令 v 为 x_i 的一个满足赋值。那么存在一个对应的映射，由 r_{valid} 捕获，用以编码这个满足赋值。由于 v 满足所有子句，对于每个子句索引 $j \in [n_c]$，都存在一个合取范式变量索引 $i \in [n_v]$，使得 $v_{i,j}$ 按照 s^i_2 赋值。因此，m 不受任何由 r_{mask} 捕获的映射支配。这意味着天际线中包含一个由 r_{valid} 捕获的映射：或者是 m，或者是支配 m 的其他某个映射；根据传递性，该映射也必定由 r_{valid} 捕获。

现在假设一个被 r_{valid} 捕获的映射 m 位于 $r_{\text{valid}} \vee r_{\text{mask}}$的天际线中。由于 m不受任何被 r_{mask}捕获的映射支配，对于所有子句索引 $j \in [n_c]$，必然存在一个变量索引 $i \in [n_v]$，使得 $v_{i,\,j}$ 按照 s^i_2被赋值。因此，m 编码的合取范式赋值满足所有子句，从而证明 F 是可满足的。总之，F 是可满足的，当且仅当一个被 r_{valid} 捕获的映射 m 位于 $r_{\text{valid}} \vee r_{\text{mask}}$的天际线中。

要检查天际线中 $r_{\text{valid}} \vee r_{\text{mask}}$ 的一个映射 m 是否被 r_{valid} 捕获，只需检查对于所有子句索引 $j \in [n_c]$，是否存在某个变量索引 $i \in [n_v]$，使得 $v_{i,\ j}$ 按照 s_2^i被赋值；这可以在线性时间内完成。

我们通过证明以下结论完成证明：如果我们能够在 D 的顺序变量自动机和归约中定义的文档上，在输出多项式时间内解决天际线提取问题（正如定理陈述所假设的那样），那么 P = NP，从而完成证明。这是引理 6.4的一个变体，不过现在我们不知道捕获映射的确切数量（例如，被 r_{mask} 捕获的一些映射可能彼此支配）。

假设存在这样的算法，且它能在 $p(|P|+|d|+|out|)$时间内解决问题，其中 out 是输出。我们按如下方式在多项式时间内求解 SAT：给定一个合取范式 F，构造 P：$= r_{\text{valid}} \vee r_{\text{mask}}$ 以及上述的 d。运行天际线计算算法 $p(|P| + |d| + n_c|m|)$ 步，其中 $|m|$ 是单个映射的编码大小。如果算法在分配的时间内终止，就检查输出是否包含一个 m 编码了满足赋值的映射，并据此判断 F 的可满足性。如果算法未能终止，就回答 F 可满足。这样做是正确的，因为与引理 6.4的推理类似，在这种情况下，天际线至少包含 $n_c + 1$ 个映射，其中只有n_c 个能被 r_{mask}捕获。因此，天际线中至少有一个映射 m 被 r_{valid} 捕获，由此可知 F 可满足。◂

▸**推论 7.6**。 除非 P = NP，否则不存在一种关于跨度包含支配规则、从左到右支配规则或跨度长度支配规则的天际线提取问题算法，能够在组合复杂度上达到输出多项式时间。即使要求输入变量自动机是函数式的，这一结论仍然成立。

证明。 我们已在示例 7.4 中证明，跨度包含和跨度长度支配规则满足 UMDSDP，因此该结果可由定理 7.5 直接得出。

对于断言的第二部分，如果要求输入变量自动机是函数式的，那么我们可以通过检查该定理的证明得出结论。具体而言，我们知道，跨度包含和跨度长度支配关系的严格支配对总是同时指定左侧跨度和右侧跨度。现在，证明中定义的跨度器总会为其捕获的映射中的所有形如 $v_{i,\ j}$ 的变量赋值：这一点可从每个 i 值对应的 r^i_{valid} 以及每个 j 对应的 r^j 有效看出。因此，所构造的变量自动机都是函数式的，所以即使在这种情况下，困难性仍然成立。◂

F.3 变量包含类规则的证明

▸**定理 7.7**。 设 D 是一个变量包含类的单变量支配规则。如果D 满足 *UMDSDP* 条件或接受形如 $(-,\ [i,\ i\rangle)$ 的对（在某个文档上），那么对于 D，给定一个顺序变量自动机和文档，其天际线提取问题在组合复杂度上不是输出多项式的，除非 P =NP。否则，对于 D，其天际线提取问题在组合复杂度上是输出多项式的。

我们将在本附录的其余部分证明这一结果。首先，我们证明该引理，以排除包含一个带有空跨度的对的情况：

▸**引理 7.8**。 设 D 是一个单变量支配规则，该规则在某个文档上接受一对 $(-,\ [i,\ i\rangle)$。那么，对于给定的顺序变量自动机和文档， D的天际线提取问题在组合复杂度下不是输出多项式的，除非 P =NP。

证明。 该结果的证明是定理 6.3 的一个简单变体。令 d 为一个文档，且$(-,\ [i,\ i\rangle)$ 为所接受的对：令 d_1 和 d_2 分别为 d之前的前缀和后缀

位置 i，使得 $d = d_1 d_2$。我们采用定理 6.3 中相同的归约，但将其应用于文档 d，并通过添加前缀 d_1 和后缀 d_2来定义 r_{valid} 和 r_{mask}。鉴于该归约只能将跨度 $[i,\ i\rangle$ 赋给变量，其余论证保持不变。◂

因此，在本节余下部分，我们证明定理 7.7在不存在此类被接受的变量对时成立。

我们先介绍关于顺序变量自动机的一些简单观察。设 $\mathcal{A}$ 是变量为 X 的变量自动机。设 d 是 $\mathcal{A}$ 的一个输入。在 $\mathcal{A}$ 对 d 的一次运行中的任意时刻，如果已经读取了 $x\vdash$，但尚未读取 $\dashv x$，我们就称 $x \in X$ 处于打开状态。

我们指出，在所谓函数变元自动机的枚举算法背景下，也使用了一个类似概念，称为变量配置，见 [13]。这里我们只需要以下简单观察：

▸**观察 F.5**。设 s 是 A 的一个状态。如果 A 在某个输入 d 上存在一条接受运行，使得在自动机 A 处于状态s时，变量 x 在运行中的某一时刻处于打开状态，那么对于任意文档上的任意接受运行，每当自动机到达状态 s 时，变量 x 都处于打开状态。

证明。反设结论不成立。设 ρ_1 是在输入 d 上的一次接受运行，在到达状态 s时的某个时刻，x 处于开放状态；令 ρ_1' 为 ρ_1 在该时刻之后的后缀。设 ρ_2 是在文档 d_2 上的一次接受运行，在某个时刻到达状态 s 时，x 并未处于开放状态；令 ρ_2' 为 ρ_2 截至该时刻的前缀。我们构造 $\rho' = \rho_2'\rho_1'$：这给出了 A 在某个文档上的一次接受运行。现在，由于 A 是顺序的，我们知道 ρ' 将包含 x 的闭合标记，且不包含 x的开放标记。由于 ρ_2' 要么包含一个开放标记和一个闭合标记，要么不包含任何标记，因此 ρ'无效，这与 A 是顺序的这一假设相矛盾。证明至此完成。◂

我们给出一些额外的定义。设 S 为一个不包含任何长度为 0 的跨度的跨度集合。若对于每个 $s \in S$，都存在一个 $i \in H$，使得 $[i,\ i+1\rangle$ 包含于 s，则称集合 $H \subseteq \mathbb{N}$ 为 S 的击中集。我们称 S 的击中集数为 S的击中集中的最小基数。

现在考虑一个文档 d，以及单变量变量包含类支配规则 D 在文档 d 上的映射集合 $D(d)$，以及它所定义的支配关系 $\preccurlyeq$。考虑按以下方式定义的严格支配跨度集合：

$$\pi_2(\preccurlyeq, d) := \{m_2 \mid (m_1, m_2) \in \preccurlyeq, m_1(x) \neq m_2(x)\}$$

在上述情形中，m_1 在 x 上必然未定义，只要 $m_1(x) \neq m_2(x)$，因为该关系属于变量包含类。换言之，$\pi_2(\preccurlyeq,\ d)$ 不过是支配“不分配变量”这一选择“–”的跨度集合。根据先前的假设，我们知道 $\pi_2(\preccurlyeq,\ d)$从不包含空跨度，否则我们就可以根据引理7.8得出结论。现在，我们将 D 在 d 上的击中集数定义为 $\pi_2(\preccurlyeq,\ d)$的击中集数。最后，我们将 D 的击中集数定义为对所有文档 d 取值时，d上的击中集数的上确界。

击中集数之所以重要，是因为它们为抽取结果的大小提供了上界 o f 跨度器与其天际线的比较方式如下。

▸**引理 F.6.** 设 D 为一个变量包含型支配规则，其右侧从不包含空跨度，且其击中集数为 $k \in \mathbb{N}$。设 $P_{\mathcal{A}}$ 为一个跨度器

由顺序变量自动机 A 定义。那么，对于每个文档 d

$$|P_{\mathcal{A}}(d)| \leqslant |\mathcal{A}|^k |(\eta_D P_{\mathcal{A}})(d)|.$$

证明。 我们将证明，$(\eta_D P_{\mathcal{A}})(d)$ 中的每个映射只能支配 $|\mathcal{A}|^k$ 中$P_{\mathcal{A}}(d)$的映射。引理由此直接得出。

我们知道，$\pi_2(D, d)$ 有一个大小为 k 的击中集 H，因此令 $K = \{i_1, \ldots, i_k\}$ 为这样的击中集。

考虑 $m \in (\eta_D P_{\mathcal{A}})(d)$ 和 $m' \in P_{\mathcal{A}}(d)$，使得 $m' \preccurlyeq m$。令 X 为 m 上有定义的变量。于是我们可以构造 X 的子集 $X_1, \ldots, X_k$，使得对每个 $j \in [k]$，当且仅当 $[i_j, i_j + 1\rangle$ 包含于 $m(x)$ 中时，$x \in X_j$。由于 H 是一个击中集，我们得到 $X = \bigcup_{j \in [k]} X_j$。由于 $m' \preccurlyeq m$，且 $\preccurlyeq$ 属于变量包含类，映射 m' 定义在 X 的一个子集上；对于其定义域中的所有变量 x，都有 $m'(x) = m(x)$。因此，m 和 m' 之间唯一的差别是它们定义在哪些变量上。

现在考虑 $j \in [k]$，并令 $X_j^{m'} := \{x \in X_j \mid m' \text{ 定义在 } x\}$上。考虑 A 在接受运行中读取 d 的第 i_j 个字母的字母转换时所处的状态 s，该运行产生映射 m'。在状态 s 中，开放变量必须恰好是 $X_j^{m'}$ 中的变量，因为 $[i_j, i_j + 1\rangle$ 包含在这些跨度中。根据观察 F.5，由于每组开放变量都必须对应一个独立的状态，因此 m' 可从中选择的子集 $X_j^{m'}$ 的数量不超过 A 的状态数，所以 $|A|$。对所有$j \in [k]$作同样的推理可知，A 只能编码 $|A|^k$ 个不同的映射 m'，这些映射受 m支配，因此断言得证。◂

▸**推论 F.7.** 设 D 是一个变量包含型支配规则，其中严格支配对的右侧不包含任何空跨度，并假设击中集数有常数上界。那么，给定一个由顺序变量自动机 A 定义的跨度器 $P_{\mathcal{A}}$ 和一份文档 d，天际线 $(\eta_D P_{\mathcal{A}})(d)$ 可以在输出多项式时间内计算出来。

证明。 我们只需计算 $P_{\mathcal{A}}(d)$ 对于顺序变量自动机，这可以在输出多项式时间内完成，见 [3, 22]。然后，我们显式比较 $P_{\mathcal{A}}(d)$ 中的所有映射对，以筛除那些被支配的映射，从而计算出 $(\eta_D P_{\mathcal{A}})(d)$。根据引理 F.6，中间结果 $P_{\mathcal{A}}(d)$ 仅比结果 $(\eta_D P_{\mathcal{A}})(d)$大多项式倍。此外，支配关系可在多项式时间内检验：只需针对每个变量，判断两个映射中所作的赋值是否满足单变量支配规则；这只需检验引用词是否被定义该规则的变量自动机接受（见附录 A）（注意，由于标记可能有多种顺序，我们可能需要检验常数多个引用词是否被接受）。因此，整个过程可在多项式时间内完成。◂

现在我们将证明，推论 F.7 中的上界对应于 UMDSDP 性质不成立的情形。为此，我们证明以下对应关系。

▸**引理 F.8.** 设 S 为不包含长度为 0 的跨度的有限集合，并设$\preccurlyeq_S := \{(-, s) \mid s \in S\}$ 为由 S诱导的变量包含式支配关系。设 k_h为 S 的击中集数，设 k_p 为 $\preccurlyeq_S$的严格支配对的最大数量。那么我们有：

$$k_p \leqslant k_h \leqslant 2k_p.$$

证明。 注意，$\preccurlyeq_S$ <style id='6}>中的两个有序对不相交，当且仅当它们的第二坐标不相交，因此，与其讨论不相交的有序对，不如讨论 S中的不相交跨度。

对于第一个不等式，请注意，每当两个跨度包含相同的跨度 $[i$时，$i+1\rangle$，根据定义，它们就不相交。该断言可直接得出：给定一个包含多于 k_h 个跨度的集合，由于每个跨度都包含击中集中的某个元素，因此必有两个跨度包含相同的元素，于是它们不相交，从而该集合不是不相交的。

对于第二个不等式，令 $K=\{i_1,\ldots,i_k\}$ 为 S 的一个大小最小的击中集。对于每个 $j\in k$，令 S_j 为 S 的跨度中包含 $[i_j,i_j+1\rangle$ 的子集。我们断言，对于每个 j，在 S_j 中都有一个跨度既不属于 S_{j-1}，也不属于 S_{j+1}（这里我们稍微滥用了记号，对于指标 $j'\notin[k]$，称 $S_{j'}$ 为空）。反证法地，假设情况并非如此，那么可以从 K 中删除 i_j，从而得到 S的一个更小的击中集，这与 K 是最小的这一假设相矛盾。由此我们可以直接选取一个大小为 $\lfloor k_h/2\rfloor$ 的跨度集合 S'，其中的跨度两两不相交：取一个属于 S_1 但不属于 S_2 的跨度，再取一个属于 S_3 但不属于 S_2（因此不会与第一个跨度相交）或 S_4，以此类推。由此可得 $k_h\leqslant 2k_p$。◂

现在我们终于可以证明定理7.7。

定理 7.7的证明。 对于上界，我们知道，如果该规则不满足 UMDSDP，那么它也不满足弱 UMDSDP（根据命题 F.2的逆否命题），因此文档上的严格支配对的最大数量是有界的。根据引理 F.8，并且由于该规则不会将空跨度捕获为严格支配对的右侧，文档上的击中集的最大大小也被某个常数所界定，记为 k。（由于该数值只取决于固定规则，因此我们是否能够高效地计算它并不重要。）

现在，给定一个变量自动机$\mathcal{A}$ 和文档 d，我们只需使用推论 F.7，即可在输出多项式时间内计算结果。注意，为此我们不需要计算基数为 k的击中集，因为它的作用只是界定结果数量与最大结果数量之间的因子。

对于困难性部分，有两种情况：如果存在一个文档 d 使得 $D(d)$包含一对形如 $(\varepsilon,[i,i+1\rangle)$的项，那么结论由引理 7.8得出。如果不存在这样的文档，那么由于该规则不满足 UMDSDP，根据定理 7.5 可知，求值不是输出多项式的。◂

F.4 其他情形的证明

▸**命题 7.9.** 回顾示例 7.4中的逐变量支配规则 D_{start}。假设 P $\neq$NP，则该规则对应的天际线提取问题不存在输出多项式综合复杂度算法。

证明。 我们沿用先前困难性证明中的类似方法。我们从合取范式的可满足性问题进行归约。给定一个含有 n_x 个布尔变量和 n_c 个子句的合取范式 Φ，令我们构造一个作用于变量 $v_{i,j}$ 的跨度器，适用于 $i\in[n_x]$ 和 $j\in[n_c]$。对于每个变量i，令 T_i 和 F_i 分别为当将 x_i 设为真（分别为假）时会得到满足的子句子集。

所用文档为 $d=a^{2n_x}$。跨度器 r_{valid} 直接定义为一个变量自动机：首先打开所有变量。立即关闭形如 $x_{i,j}$ 的变量，其中变量 i 不出现在子句 j 中。然后，对于每个变量 $1\leqslant i\leqslant n_x$，在以下两种操作之间进行析取：关闭所有变量 $x_{i,j}$（对所有 $j\in F_i$），然后读取 aa；以及读取 a，关闭所有变量$x_{i,j}$（对所有 $j\in T_j$），然后读取 a。不出所料，r_{valid} 捕获的映射与变量的赋值相对应：如果变量 i 捕获跨度 $[0,2(i-1)\rangle$，则其值设为 false；如果捕获跨度 $[0,2(i-1)+1\rangle$，则其值设为 true。

跨度器 r_{mask} 是 n_c 个跨度器之间的析取，每个跨度器都捕获一个映射：对于 $1 \leqslant k \leqslant n_c$，第 k 个跨度器捕获如下映射：将变量$x_{i,j}$ 对所有 $i \in [n_x]$ 和 $j \in [n_c]\backslash\{k\}$ 赋值为整个文档（跨度 $[0, 2n_x\rangle$），并将变量$x_{i,k}$ 对所有 $i \in [n_x]$ 赋值为空跨度 $[0, 0\rangle$。

和通常一样，如果对赋值的选择中没有任何变量使子句 k 成立，那么 r_{valid} 的映射就被 r_{mask} 的第 k 个映射严格支配；也就是说，它没有对任何形如 $x_{i,k}$ 的变量赋值。因此，如果 r_{valid} 的映射表示 Φ 的满足赋值，那么它就不会被 r_{mask} 的任何映射严格支配。

因此我们断言，如果 $r_{\text{valid}} \vee r_{\text{mask}}$ 的天际线包含一个由 r_{valid} 捕获的映射，那么 Φ 可满足。确实，对于正向推导，天际线中的一个 r_{valid} 映射不可能被任何 r_{mask} 映射严格支配，因此它证明了 Φ 可满足。对于反向推导，我们已经论证，公式的一个满足赋值描述了一个 r_{valid} 映射，该映射不被任何 r_{valid} 映射支配，这意味着该映射或 r_{valid} 的某个其他映射必定在天际线中。

然后，我们以定理 7.5证明结尾处的论证得出结论：对于该规则，天际线提取问题不可能具有输出多项式组合复杂度，除非 P = NP。 ◂

▸**命题 7.10**。 考虑由正则表达式 $x\{a^*\}a^*x^\dagger\{b\} \vee D_{self}$表示的逐变量支配规则。假设 P ≠ NP，则不存在适用于该规则的天际线提取问题的输出多项式综合复杂度算法。

证明。 证明再次是前述证明的一种变体。我们再次将一个合取范式的可满足性问题归约为 Φ，其中有 n_x 个变量 x_i 和 n_c 个子句 C_j，再次记 T_i和 F_i 分别表示令 x_i 为真或假时所满足的子句。

文档为 $d = a^{2n_x}b$。直观上，形式为 $[0, 2i\rangle$ 的跨度对应于我们做出的选择，而形式为 $[0, 2i + 1\rangle$ 的跨度对应于我们未做出的选择。

跨度器 r_{valid} 首先通过打开所有变量$x_{i,\ j}$来定义为一个变量自动机。然后，对于所有 $1 \leqslant i \leqslant n_x$，考虑 i-th 因子 aa，我们在两种可能性之间进行析取：关闭变量 $x_{i,\ j}$ 用 $j \in T_i$，然后读取 a，再关闭变量 $x_{i,\ j}$ 用$j \in [n_c]\backslash T_i$，然后读取 a；或者关闭变量 $x_{i,\ j}$ 用 $j \in F_i$，然后读取 a，再关闭变量 $x_{i,\ j}$ 用$j \in [n_c]\backslash F_i$，然后读取 a。

最后，我们读取 b（注意，所有变量都已关闭）。每个捕获映射都对应一个赋值；在该赋值中，如果我们对变量 $x_{i,\ j}$ 赋值为 $[0,\ 2i\rangle$，使子句 j 为真，则将变量赋值为 x_i；如果我们对 x_i 的赋值未能使子句 j为真（包括 x_i 未出现在子句 j中的情况），则将变量 x_i赋值为 $[0,\ 2i + 1\rangle$。

跨度器 r_{mask} 定义为以下各跨度器的析取，每个$1 \leqslant k \leqslant [n_c]$对应一个跨度器。首先打开所有形如 $x_{i,\ k}$的变量。然后，对于第 i个因子 aa，读取一个 a，接着关闭 $x_{i,\ k}$，再读取一个 a。最后，在最终的 b 上，为所有$i \in [n_x]$ 和所有 $j \in [n_c]\backslash\{k\}$将变量 $x_{i,\ j}$ 赋值。

我们再次断言， r_{valid} 的映射严格劣于r_{mask} 的第 k 个映射 m_k，当且仅当它没有为使子句 k 成立而赋值变量。这是因为所有形如 $x_{i,j}$ 且满足 $j \neq k$ 的变量，都被赋值到了 d 中的最终 b，而这些变量严格优于所有其他可能的选择；因此，要想不被严格劣化，唯一的办法就是将形如 $x_{i,k}$ 的变量赋值到不同于 m_k 为其赋值的位置，也就是说，使该子句成立。因此， r_{valid} 的映射不被任何 r_{mask} 的映射严格劣化，当且仅当它编码了 Φ 的一个满足赋值。

42 用于文档跨度器的天际线算子

我们再次考虑 $r_{\text{valid}} \vee r_{\text{mask}}$，并由前文（如定理7.5）得出，对于该规则，天际线提取问题不可能具有输出多项式组合复杂度，除非P = NP。◂

▸**命题 7.11**。逐变量支配规则下的天际线提取问题$x^{\dagger}\{a^*\}a^*x\{b\} \vee D_{self}$在组合复杂性方面是输出多项式的。

证明。对于镜像规则，给出一个算法会更直观：$x\{b\}a^*x^{\dagger}\{a^*\}$。我们这样做是因为，取镜像操作可以保持可处理性，只需对输入变量自动机和文档分别取镜像即可。更准确地说，要对输入变量自动机取镜像，我们只需反转所有转移（对于每个从 q 到 q'的转移，我们创建一个从 q' 到 q的转移），交换初始状态和终态，最后修改变量自动机以满足我们的定义，因为该定义只允许一个初始状态：可以通过向模型中添加 ε-转移来实现（见附录 A），添加一个带有 ε-转移、指向所有原初始状态的初始状态，然后移除 ε 转移。这个镜像过程在多项式时间内完成，因此我们可以证明上述镜像规则具有可处理性。

考虑一个输入变量自动机 $\mathcal{A}$（不失一般性地，我们假设它是修剪的），其状态空间为 Q，以及一个输入文档 d。我们假设输入文档 d 属于支配规则的语言 ba^*；否则，支配关系就是平凡的，我们只需在多项式时间内计算 $\mathcal{A}$ 在 d 上的全部输出。现在，我们提出如下断言（*）：$\mathcal{A}(d)$ 的每个映射至多能支配 $|Q|^2$ 个其他映射。如果此断言成立，我们便可立即得出一个输出多项式算法，如证明推论 F.7中的算法。

要了解断言（*）为何成立，我们来考察自动机可以将哪些变量集合赋给初始 b。设 $q_1, \ldots, q_n$（其中 $n \leqslant |Q|$）为从初始状态 q_0 出发、仅读取标记后接一个 b- 转移即可到达的状态。根据观察 F.5，每个满足 q_i 的状态 $1 \leqslant i \leqslant n$ 都必须对应一个确定的集合 X_i，即在该阶段仍处于开放状态的变量集合，因为$\mathcal{A}$ 是顺序的且经过修剪。现在，考虑集合 $q'_1, \ldots, q'_m$（其中 $m \leqslant |Q|$）—— 这些是可通过标记转换从 q_i 到达的状态。同样，这些状态中的每一个 q'_i 都必须对应一个确定的集合 X'_i，即仍有开放标记的变量集合。

现在，$\mathcal{A}$ 的一个接受运行必须先转移到状态 q_i，同时读取初始 b 并开启一组精确的变量 X_i；然后通过标记转换继续到状态 q'_j，再进行下一次字母转换（如果 d 是单字母文档 b，则接受），此时仍处于开启状态的变量集合为 X'_j。在该运行中，分配给初始 b 的变量集合恰好是 $X_i \backslash X'_j$，也就是说，这些变量在读取 b后立即开启，并在下一次转移前关闭。现在我们可以看到，由于 n，$m \leqslant |Q|$，这样的集合至多有$|Q|^2$ 个。

这一论证表明，如果映射 m 被另一个映射 m'严格支配，那么根据变量支配规则，m 与 m' 之间的差别必定是：m 将某些变量分配给初始 b，而 m' 将这些变量分配给形式为 a^*的后缀；其他变量的分配则完全相同。但分配给第一个 b的变量集合只有 $|Q|^2$ 种可能，因此这样的映射 m至多有 $|Q|^2$ 个。这就证明了我们的断言（*），并使我们得以结束证明。◂