arXiv:2305.02271v1 [cs.CC] 1 Apr 2023

L 在强指数时间假设下不等于NL

Reiner Czerwinsk

2023年5月4日

摘要

根据 Savitch 定理，我们知道$\mathbf{NL} \subseteq \mathbf{DSPACE}(\log^2(n))$。为了证明这一上界，Savitch 构造了一种算法，在工作带上使用 $O(\log^2(n))$ 空间。我们将证明，在强指数时间假设下，Savitch 算法也给出了一个下界。在这种情况下，任何用于连通性问题的算法都需要 $O(\log^2(n))$ 空间。

1 引言

在非确定性复杂度类与确定性复杂度类之间的关系中，存在一些众所周知的上界。显然，$\mathbf{NSPACE}(t) \subseteq \mathbf{DTIME}(2^{O(t)})$，并且根据 Savitch 定理[4]，$\mathbf{NSPACE}(s) \subseteq \mathbf{DTIME}(s^2)$。本文将分析，在强指数时间假设（SETH）下，这些界是否也是下界。

Savitch 定理的证明是构造性的。有一种算法可以在有向图中找到两个顶点之间的路径，该算法需要 $O(\log^2(n))$ 空间。为证明 Savitch 算法是最优的，我们采用以下方法：首先，我们将证明，对于每个算法，都存在一个含有 n 个顶点的图，使得该算法需要占用超过一个顶点的标记空间。利用这个图，我们将构造一个含有 n^l 个顶点的图，在这个图中我们必须标记 l个顶点。

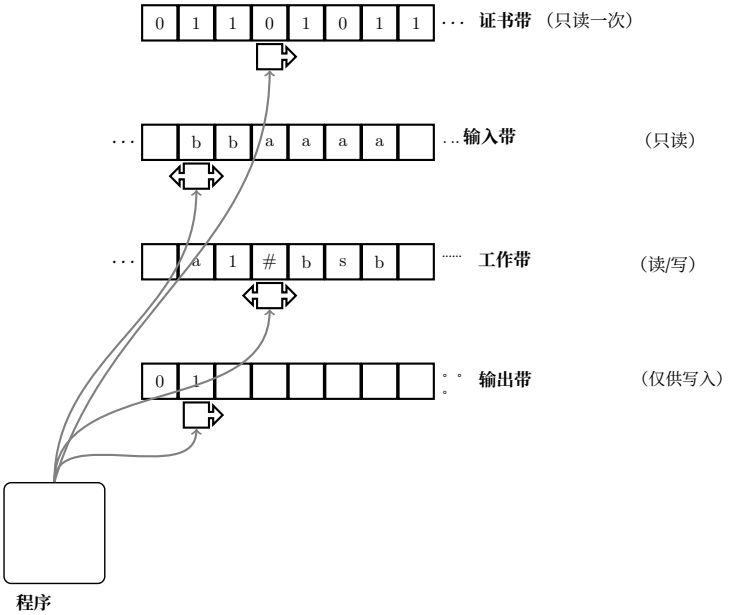
在第 2 节中，我们介绍非确定性图灵机（NTM）。我们需要使用非确定性图灵机来分析强指数时间假设对第 **NL** 节中 NL 完全问题的影响。3。

在第 4 节中，我们将证明，当强指数时间假设成立时，Savitch 定理中描述的算法具有最小空间复杂度。因此，强指数时间假设意味着 $\mathbf{L} \neq \mathbf{NL}$。

2 非确定性图灵机

在本文中，我们使用带有附加证明带[1, p.84-86]的非确定性图灵机（非确定性图灵机）。非确定性图灵机具有一条只读输入带、一条只读一次的证明带、一条工作带和一条输出带。不失一般性，

证明带的磁头在非确定性图灵机的每一步都向右移动一个单元。如果存在证明 $c \in \{0,1\}^*$ 使得 $M(x,\ c) = 1$，则输入 x 被非确定性图灵机 M 接受，即 $x \in L(M)$。



配置是指从某一特定时刻继续执行非确定性图灵机计算所需的信息。它由输入带和工作带上的内容、输入带和工作带的读写头位置，以及非确定性图灵机的状态确定。配置不依赖于证明带上的内容。

非确定性图灵机的配置由工作带和输出带上的内容，以及输入带、工作带和输出带的读写头位置表示。

配置图是一个有向图，其顶点是非确定性图灵机可能的配置。如果一个配置可以在非确定性图灵机的一步操作内转换为另一个配置，那么这两个配置之间就有一条有向边。如果配置图中某个顶点的出度大于 1，则根据证明带当前单元格的内容区分可能的边。因此，证明描述了配置图中的一条路径，但它本身并不属于配置。

3 强指数时间假设

指数时间假设意味着，解决**可满足性问题** 的确定性算法需要指数级运行时间。如果强指数时间假设（SETH）成立，那么对于一个**可满足性问题**公式，若其为 k –**合取范式**，且有n 个变量，那么在最坏情况下，运行时间需要 $O(2^n$，当 $k \to \infty$[3]时。

根据 Cook–Levin 定理，SETH 与非确定性图灵机有关[2]。我们定义

$$U_{\mathbf{NP}} = \{\langle N, x, 1^k\rangle \mid \text{ NTM } N \text{ accepts } x \text{ within } k \text{ steps}\} \tag{1}$$

$$U_{\mathbf{NL}} = \{\langle N, x, 1^k\rangle \mid \text{ NTM } N \text{ accepts } x \text{ within } \log(k) \text{ working space}\}. \tag{2}$$

显然，集合 $U_{\mathbf{NP}}$ 是**NP**-完全的，而集合 $U_{\mathbf{NL}}$ 是**NL**-完全的。Cook–Levin 定理使用从 $U_{\mathbf{NP}}$ 到**可满足性问题**的多项式时间多对一归约。借助这种证明技巧，我们可以在强指数时间假设下分析 $U_{\mathbf{NL}}$ 。

引理 1. 假设强指数时间假设成立。那么我们必须检查以配置图的初始配置为起点的每一条可能路径。

证明。 我们使用 Cook-Levin 定理中的归约。非确定性图灵机运行期间每一步的配置都由布尔变量描述。如果工作空间磁带在第 p 步的 s 位置上有符号 s，则变量 $T_W(t, p, s)$ 为真。如果工作空间磁带的读写头在第 t 步位于 p，则变量 $H_W(t, p)$ 为真。其他读写头也有类似的变量。

此外，我们还根据空间限制得到一些子句。为此，我们引入布尔变量 $S_W(t, p)$；当工作空间磁带的读写头在第 t 步之前经过位置 p 时，该变量为真。它满足 $S_W(t+1, p) = S_W(t, p) \vee H_W(t, p)$。如果 k 是磁带空间的最大值，那么对于所有 t、p，我们都添加子句 $(\neg S_W(t, p) \vee \neg S_W(t, p + k))$。

假设强指数时间假设成立。如果非确定性图灵机尚未用完整个磁带空间，那么算法就必须检查所有路径。 □

虽然我们必须检查所有路径，但可以通过记住一些配置来加速针对 $U_{\mathbf{NL}}$ 的算法。

4 Savitch 的算法是最优的

一个 **NL**-完全问题是 **st** $-$**CON**。给定有向图中的两个点之间是否存在一条路径？这个问题属于 **NL**，因为非确定性图灵机只需要存储一个点所需的工作空间大小，即 $\log(n)$，其中 n 是顶点数。该问题是 **NL**-困难的，因为存在从 $U_{\mathbf{NL}}$ 到 $\mathbf{st} - \mathbf{CON}$ 的归约，方法是写出配置图。

一个 $\mathbf{st} - \mathbf{CON}$ 算法必须在工作空间中写入有关有向图的一些信息。不失一般性，它可以通过标记某些顶点来存储信息。有向图由顶点和有向边组成。每条边都是由两个顶点构成的元组。如果通过将某个顶点写在工作带上来标记它，则需要 $O(\log(n))$ 的空间。

引理 2。 假设强指数时间假设成立。对于每个能够解决 **st** $-$**CON** 的算法，都存在一个有向图，其中至少有两个顶点会在同一时刻被标记。

证明。 问题 $\mathbf{st}-\mathbf{CON}$ 是 **NL**完全的。因此，存在一个从$U_{\mathbf{NL}}$ 到 $\mathbf{st}$ $-\mathbf{CON}$的归约。我们用对数空间转换器构造配置图。根据定理 1，在最坏情况下，确定性算法必须测试从初始配置到接受状态的所有可能路径。如果算法每次只标记一个顶点，就必须遍历每条路径。在这种情况下，运行时间将呈指数级增长。因此，工作空间的增长速度会超过对数级。

因此，存在一个图，算法必须标记多个顶点才能缩短计算时间。

□

定义 1（st-替换）。 设有一个包含顶点 v 的有向图 G，以及一个包含顶点 s_H 和 t_H 的有向图 H。当每个与 v 相邻的顶点都将与 s_H 相邻，并且每个从 v 可达的顶点都将从 s_H 可达时，将用 H 替换顶点 v。

如果一个用于 $\mathbf{st}$ $-\mathbf{CON}$ 的算法将在原有向图中标记被替换的顶点，那么它就能在替换该顶点的子图中解决 $\mathbf{st}$ $-\mathbf{CON}$ 问题。因此，我们可以将引理 2推广为:

引理 3。 假设强指数时间假设成立。对于任意用于 $\mathbf{st}-\mathbf{CON}$ 的确定性算法，都存在一个具有以下性质的有向图：如果用一个子有向图对每个顶点进行 *st*-替换，那么在运行过程中存在某个时刻，算法正在其中一个子有向图上求解 $\mathbf{st}$ $-\mathbf{CON}$，并且该子有向图之外的某个顶点已被标记。

证明。 如引理 2所述，算法需要检查的路径数量可能会随着替换前顶点数量的增加而呈指数增长。因此，在运行过程中必然存在某个时刻，算法正在一个替换后的子有向图上进行计算，却标记了该子有向图之外的一个顶点。

□

现在我们使用 st-替换来构造一个有向图，在某个时刻必须标记多于两个顶点。

算法 1图 G^l 的生成

$K \leftarrow G$
如果 $l > 1$ **则**
  对于 $v \in V(K)$ **执行**
    将 v 替换为 G^{l-1}
$G^l \leftarrow K$

如果图 G 有 n 个顶点，那么 G^l 有 n^l 个顶点。

引理 4. 如果强指数时间假设成立，那么对于任意求解$\mathbf{st}-\mathbf{CON}$ 的算法，都存在一个有向图 G，使得 A 在输入G^l上某个时刻至少标记 $l+1$ 个顶点。

证明。 我们用归纳法证明这个引理，归纳对象为 l。根据引理 2，存在一个有向图 G，在该图上，算法 A 必须在某个时刻标记至少两个顶点。如果 $l = 1$，那么 $G^l = G$。因此，在 A的运行过程中，存在某个时刻它已标记至少两个顶点。

对于图 G^{l+1}，存在一个与 G 同构的有向图 K，其中每个顶点都将被 G^l 替代。如果 v 是 K 的一个顶点，那么 $G^l(v)$ 就是将替代 v 的子有向图。因此，st − CON 算法必须在子有向图 $G^l(v)$ 上求解 st − CON，而不是标记 v。根据归纳假设，必须标记该子有向图中的 $l + 1$ 个顶点。但是， K 与 G 同构。根据定理 3，在该子有向图上计算 st − CON 时，存在一个顶点 v，还必须标记 $G^l(v)$ 之外的一个顶点。 □

图 G^l 有 $N = n^l$ 个顶点，算法 A 需要工作空间来标记 $l + 1$ 个顶点。每个顶点需要 $\Theta(\log(N))$ 空间。因此，我们需要$\Theta(\log^2(N))$ 工作空间。强指数时间假设意味着 Savitch 算法是最优的。

参考文献

[1] Arora, S., Barak, B.: 《计算复杂性：现代方法》。剑桥大学出版社（2009），https://books.google.de/books?id=nGvI7cOuOOQC[2] Cook, S. A.: 定理证明过程的复杂性。载于：第三届年度 ACM 计算理论研讨会论文集。第 151–158 页。STOC '71，计算机协会，美国纽约州纽约市（1971）。https://doi.org/10.1145/800157.805047,
https://doi.org/10.1145/800157.805047[3] Impagliazzo, R., Paturi, R.: k-SAT 的复杂性。载于：第十四届年度 IEEE 计算复杂性会议论文集（原名：复杂性理论中的结构会议）（目录编号：99CB36317）。第 237–240 页（1999）。https://doi.org/10.1109/CCC.1999.766282[4]Savitch, W.J.: 非确定性与确定性磁带复杂性之间的关系。《计算机与系统科学杂志》4（2），177–192（1970）。
https://doi.org/https://doi.org/10.1016/S0022-0000(70)80006-X,
h
t
tp://www.sciencedirect.com/science/article/pii/S002200007080006X