



明智地赢得Wordle-- 或如何用数学毁掉一个有趣的互 联网小游戏

Martin B. Short

在这篇短文中，作者试图毁掉一个名为Wordle的在线猜词游戏，并试图将整个游戏数学化。另一方面，人们可能会认为，作者试图通过将数学应用于在线猜词游戏来调剂数学。另外，扰流警报：不要读过了

如果你想保留Wordle的魔力，可以使用 "解决方案是....."这句话。

如何玩转Wordle

现在是2022年1月，Wordle已经在互联网上掀起了风暴。6个星期后，即2月中旬，谷歌趋势的时间轴显示，Wordle在明显的指数式上升后，处于价值100的首位。

毫无疑问，鉴于互联网流行的一般性质，很快就会出现类似的快速下降。

你可能会问，什么是Wordle？它是一个有趣的基于互联网的猜词小游戏。其前提和玩法是Wordle的玩法很简单。Wordle游戏的基础是玩家试图通过一连串的猜测来找到一个秘密的解决方案。解答词总是英语中的一个五个字母的单词（至少根据一本字典；稍后会有更多关于这个的内容）。玩家最多有六次猜测的机会来尝试找到秘密的解决方案。玩家的每一个猜测也必须是英语中的五个字母的单词，这也是上文提到的关于字典的注意事项。我还应该指出，使Wordle特别有趣的一个方面是，每天都有一个独特的Wordle单词供全世界的玩家使用，这使这个相当简单的游戏具有一定的社会性。

Wordle游戏向玩家提供信息的方式是，将最近一次猜测中的每个字母用三种可能的颜色之一标记出来，每一种颜色都有特定的含义。

1. 绿色(G)：一个绿色的字母表明，秘密解决方案的单词在那个确切的位置上有那个特定的字母。如果你愿意的话，这是一个直接命中。所有字母都应该

首先要检查它们是否为绿色，在这种情况下，这些实例不需要检查其他两种颜色。

2. 黄色(Y)：黄色字母一般表示秘密解决方案的单词在某处有这个字母，但不是在那个特定的位置。这种颜色是一种比绿色复杂一点，因为这里有边缘情况要考虑，绿色是完全不含糊的。黄色字母的完整算法似乎是这样的。首先，从密解中删除任何与绿色字母对应的字母，因此

留下一个子串（这是为了使已经标记为绿色的字母不会同时产生黄色）。然后

从左到右，每次一个（非绿色）猜测字母，检查该字母是否位于子串内。如果是，那么该猜测字母将被

标记为黄色，与之匹配的子串中的字母将从子串中删除

在猜测词的下一个字母被检查之前（这是为了防止解决方案中的一个字母出现多个黄字）。最后，在

猜测词中出现的绿色和黄色字母的总数将是最小的

。猜测词中出现的次数和解决词中出现的次数的比值

3. 黑色(B)。灰色也许比黑色更能反映Wordle在这里使用的颜色，但我将在本文件中使用黑色。任何非绿色或黄色的字母都是黑色的。因此，黑色的字母通常表示不包含在秘密解决方案中的任何地方的字母。这方面的例外是，可能出现在解决方案中的字母，但已经有一个相应的猜测字母被标记为绿色或黄色。

下面是一些解决方案和猜测的例子，以及Wordle所指示的相应的字母颜色代码，以防止有任何混淆。

- 解决方案。WEARY，猜测：TRAWL。一个直接的例子，结果是BYGYB，即TRAVIL。

2当我写这篇文章时，我真的相信。然而，最近宣布，Wordle被《纽约时报》收购，收购金额未透露（但为七位数）。见"纽约时报购买Wordle"，可在<https://www.nytimes.com/2022/01/31/business/media/new-york-timeswordle.html>。2022年2月4日访问。

3编者注：本文于2022年6月中旬编辑，目前兴趣水平为50。

- 解决方案。DRAFT, 猜测WIST
。得到的颜色是BBBBBG。在这里, 虽然解决方案中包含了T, 但猜测中T的第一个实例被赋予了B的颜色, 因为解决方案中的T已经对应于一个G色的字母。
- 解决方案。GLEAN, 猜测NNY
。结果的颜色是YYBBB。在猜测中, 只有第一个字母N的实例, 从左到右读, 是Y的颜色。

为了使即将进行的讨论更有意义, 我应该在这里描述一些决定和假设, 这些决定和假设将成为我今后所写一切的基础。

首先, 让我们把Wordle允许玩家猜测的所有单词列表称为真正的Wordle词典。这个词典可以在网上找到, 事实上, 人们可以简单地下载Wordle的源代码, 其中就包含了这个词典。

然而, 在本文的大部分工作中, 除了特别指出的例外情况, 我不会使用真正的Wordle词典。这是有一些原因的。首先, 真正的词典可以(在被《纽约时报》购买后, 已经做到了)随时改变, 所以对特定的词典可能获得的确切结果其实并不那么有趣; 重要的是方法和想法。第二, 真正的Wordle词典是相当大的, 它包含了许多我根本拒绝承认英语单词的五个字母的组合, 并且我敢打赌, 99.99%的以英语为母语的人从来没有使用过, 也没有在任何情况下看到过, 甚至可以猜测他们的定义。所以我并不觉得把它们排除在外有什么不好, 而且为了计算的目的, 我宁愿用一个较小的词典来工作。第三, 我想在玩Wordle时为自己保留一些神秘感和挑战性, 花太多时间分析真正的词典可能会毁了我的想法。

因此, 我在这里介绍的大多数结果将基于一个特定的五字母英语单词词典, 该词典的内容是

我使用Mathematica

13.0.0.0获得。在从这本词典中删除专有名词以及少量含有一撇一捺或带有字母的单词之后

词典包含 $N=5170$ 个词(我在2月获得的"真正的"Wordle词典)。

4, 2022, 包含12,972个

"词")。此后, 变量 N 将被用来代表正在使用的Wordle词典的大小, 无论它是什么。在任何情况下, 无论Wordle游戏中使用的是哪个词典, 都将被称为完整的词典。

第二, 我将假定, 在未来, 秘密的

任何给定的Wordle游戏中的单词都是从上面提到的完整词典中提取的。此外, 我将假设词典中的每个词都同样有可能成为任何特定游戏的秘密词。这是不正确的。秘密的

在可预见的未来, 所有Wordle游戏的单词都可以通过Wordle的代码查看(但我自己绝对不敢看), 所以它们显然不是在每一天随机抽取的, 即使这个序列最初是随机确定的, 也很怀疑它是否包含任何单词超过一次, 所以随机过程也许有一些记忆。另外, 这个可预见的未来解决方案的列表要比完整的词典小得多。

在只有2315个字(对于我在2月4日获得的名单。

2022年, 我从来没有看过, 我发誓), 所以它不包含所有的字典里的词。所有这些在这里都被忽略了。要调整我在这里介绍的任何方法和算法, 使之只使用这套已知的可能解决方案, 是很容易的, 但如果你要这样做, 为什么不直接看那天的解决方案的源代码就完事了?

让我们把当前的单词列表(取自完整的字典或你们作弊者可能想使用的任何子字典)称为

"可行集", 考虑到迄今为止所作的猜测和收到的相应颜色代码, 这些单词仍有可能是该游戏的秘密单词。然后, 每次玩家做出猜测并重新收到一个颜色代码, 可行的集合就会被更新, 无论是游戏还是玩家。如果我们把可行的

在第 m 轮(在进行第 m 次猜测之前), 由向量 \mathbf{v}^m

(所以这个向量的每个条目都是字典中的一个词)来设定, 在第 m 级=可行的解决方案列表是完整的词典(根据假设), 那么在第 m 轮所做的猜测和收到的颜色代码将决定 \mathbf{v}^{m+1} 是什么。

现在, 我已经想了很多关于不同方式的问题。

代表一个Wordle的游戏, 以及如何最有效地从先前的可行集、最后的猜测和收到的颜色代码中推断出新的可行集 \mathbf{v}^{m+1} 。我已经解决了

毫不奇怪, 这是一个矩阵。让我们把这个非常有用的矩阵称为颜色代码矩阵, 用 \mathbf{C}^m

表示, 其中 m 的上标再次指的是我们在进行猜测 m 之前所处的游戏回合。 \mathbf{C}^m

的每一列都对应于当前可行的不同的可能解字。

集 \mathbf{v}^m

, 而每一行对应于一个未猜测的词, 该词仍然可以从完整的词典中猜出;

行数为 $N+1$ -

m , 每一个未猜测的词有一行。矩阵条目 \mathbf{C}^m

是表示颜色的

人们可以用各种方式来表示这些颜色代码, 但为了我的目的, 我只用一个从1到 $M=3^5$ -

$5=238$ 的整数, 这就包含了所有可能的颜色代码(这里减去的五个可能的代码代表由四个G和一个Y组成的代码, 这在逻辑上不可能)。颜色代码在这些整数中的排序通常并不重要, 尽管我将决定颜色代码 M 对应的是GGG, 原因将在后面明确。

那么, 在作出猜测后更新可行集就很简单了。如果猜测的是单词 i , 收到的颜色代码是 c , 那么更新的可行集是

只是 \mathbf{v}^m 中具有索引 j 的所有词的集合, 以便 $\mathbf{C}^m = \mathbf{C}^m_{ij}$

当然, 要使用这种方法, 必须为给定的词典预先计算 \mathbf{C}^1

; 这只需要做一次, 然后可以把结果储存起来, 供以后使用。关于

我的(相当强大的)个人电脑运行所有12个内核在与Matlab并行的情况下, 使用大小为 $N=5170$ 的词典计算 \mathbf{C}^1

, 只需要16秒, 而对于大小为 $N=12972$ 的真正Wordle字典, 只需要100

秒。因此, 明显是可以做到的。

值得一提的是, Wordle也有一个可选的

"硬模式", 在这个模式下, 每个猜测都必须符合目前从先前的猜测中收集到的关于秘密解决方案的所有已知信息。也就是说, 在硬模式下, 任何在 m 级做出的猜

测都必须是属于

m 为了使我们的方法适应这种模式，从 C^m 到 C^{m+1} ，唯一必要的改变是只保留 C^m 中与 v^{m+1} 中的词相对应的那些行；这样， C 将始终是一个正方形的马。我们将在后面讨论更多关于硬模式的问题。

因此，在描述了这个游戏以及人们如何将其概念化或在计算机上实现它之后，我们现在来到了问题的核心。玩Wordle的"最佳

"方式是什么？在随后的章节中，我将为这个问题提供一些可能的解决方案，并提出一些可能很有意义的后续问题来解决。我还应该在这里指出，已经有很多很多的网上资料（根据Google4的数据，大约有3450万）在讨论这个问题。鉴于这些资料的数量

讨论这个问题的网站大大超过了Wordle词典的规模，我下面写的东西可能不是唯一的。然而，与传统的学术研究（本文肯定不是）相比，我认为在通过本说明将我自己的结果发送到世界之前，我必须不阅读任何这些文章。我认为这种感觉可以归因于我不想Wordle上过度"作弊"，毕竟这是一个我相当喜欢的游戏。

Wordle的意义何在？

一般来说，一个特定的Wordle游戏的唯一目标是在你分配的六个猜测中找到秘密的解决方案。如果你做到了，你就

"赢了"。那么，也许我们应该寻找一种方法，保证我们在任何Wordle游戏中都能获胜，无论该游戏的秘密解词是什么。这是一个有效的目标，但我不会过多地关注这个目标，主要原因是它似乎并没有

在Wordle游戏中获胜是如此困难；不是我自吹自擂，但我自己从未在Wordle游戏中失败过（谦虚的吹嘘）。

当然，由于Wordle只使用了完整字典的一个子集作为其秘密解决方案，而这个子集很可能被选择为人们最熟悉的单词，所以这并不是对我在使用完整字典的可能解决方案时可能遇到的情况的真正公平评估。但是，由于这项工作的其余部分将基于一个

熟悉某些算法可能是解决Wordle问题的最佳方法。

词或没有词并不真正相关，所以可能

即使使用完整的字典来寻找可能的解决方案，计算机也能很容易地在Wordle上获胜。在任何情况下。

这是我能够并将对这里描述的任何算法进行评估的东西。

我的立场是，Wordle的真正目标不仅是获胜，而且是以最少的猜测次数获胜。游戏本身也证明了这一点，当赢的次数越少时，游戏会给出更多令人印象深刻的褒奖--从第六次尝试时的"吁"到"伟大"，然后是

"辉煌"、"令人印象深刻"、"壮丽"，如果游戏在一开始的猜测中获胜，则会出现极其难得的

"天才"（这只是100%纯运气。

不是天才）。所以这是我在这里要关注的目标。如果我们实现了这一目标，并找到一种算法，当它获胜时，需要很少的猜测来实现，那么很有可能该算法将（几乎）总是获胜（但在没有进一步调查的情况下，这并不能严格保证）。

当然，Wordle的游戏也是一种机会。

除非你成功地构建了你的猜测以及相应的颜色代码，以至于只剩下一个可行的词（关于这一点，稍后会详细说明），否则你在下一次猜测中选择正确的词的概率小于1。因此，我们最多可以说，Wordle的目标是在最少的预期回合数中找到游戏的解决方案，这里的"预期"是指相对于某种概率分布的预期。这个概率当然会考虑到以下事实

在任何给定的游戏中，我们不知道秘密的解决方案是什么，所以我们必须对所有可能的（剩余的）解决方案进行平均。

现在我们有了一个目标，我们可以详细说明如何在原则上实现它。一个完整的Wordle策略将涉及到指定，对于任何

潜在的（可实现的）颜色代码矩阵 C^m

下一个词将选择 g^{m+1} 。我们将标志着一个给定的策略的 S_k ，这样， $g^{m+1} = S(C^m, m)_k$ 。当然了。

所有可能的 C 的全部空间是巨大的，而

所有可能的策略的全部空间仍然更大。不过，就目前而言，考虑到理论上有能力划定

在所有可能的策略 S_k 中

，我们可以通过以下方式找到最优的此类策略。对于每个这样的 S_k ，

在 $j=1, 2, \dots, N$ 的情况下，对每一个可能的第 j 个隐藏解词使用该策略，在每一种情况下都注意到其中猜测得到的解决方案。称这个数字为

R_{kj} 。在许多情况下，结果将是 $R_{kj} > 6$ ，表明该特定策略在对抗第 j 个秘密解决方案时将失败，但这没有关系，因为策略

在任何相当数量的解词上显示出这一特性的策略，无论如何都不可能是我们衡量的最优策略。那么，对于策略 S_k ，其平均数量

在所有可能的解决方案中需要猜测的字数。

再次提醒大家，我们假设所有的解字都是同样的概率，就是

$$E_k = \frac{1}{N} \sum_{j=1}^N R_{kj}$$

那么，最优策略就是一个策略 S_{k_i} ，据此 $E_{k_i} \leq E_k$ ，适用于所有 k ；也就是说，一个具有最小的相应 E_k 的策略。

当然，虽然上述情况在概念上可能有帮助

在决定最优策略必须满足什么条件时，我无法想象它是可以通过这种方式计算的，因为所涉及的状态空间极其庞大。因此，为了尝试找到这样的最优策略，我们需要构建一些近似的算法，看看我们是否能用这些算法找到最优策略。

4见<https://www.google.com/search?q=wordle+最佳+第一+字>。

近似的最佳策略

在这里，我概述了一些近似上述真正的最优性问题的方法。

基本方法

再考虑一下，我们的目标是用最少的猜测次数来解决Wordle问题，在所有可能的秘密解决方案中平均。

解释。还要注意的，在Wordle中每猜中一个词，所得到的颜色代码将导致当前可行的可能解决方案集的减少，从完整的字典开始，因为获得的信息排除了一些词（列被删除）。

猜测/颜色代码可能不会导致减少，但可行集的大小绝不会因猜测而增加）。

最快速减少（MRD）算法

那么，也许达到我们目标的一个近似方法是尝试如下：在每次猜测 m 时，给定当前的颜色代码矩阵 C^m

，猜测那个词/行（或者如果有并列的话，猜测其中的一个）将导致更新的可行的最小尺寸（以成员数计算）。

设 v^{m+1} ，期望在当前所有可行的词中，从 v^m 。

在这种方法中，我们并不真正担心获得在我们达到可行的解决方案列表被缩减到一个（或者也许两个，这时我们只需猜测其中一个）剩余的解决方案之前，我们的目标是尽可能快地达到这个结果。然而，我们可以在这个方法中建立一个简单的理智检查：如果在任何一个猜测 m ，当前的最佳猜测集有一些与可行集 v^m

重叠的词，那么我们肯定会选择这些重叠的词中的一个。这就是--

因为这些重叠的词不仅和非重叠的词一样实现了可行集大小的减少，而且它们本身也有一些（也许是很小的）概率成为正确的解决方案，因此可能在它们被选中后立即结束游戏。

对于MRD方法，需要记住的一点是，它是一种贪婪的算法。这意味着它并不展望下一步；它只是试图在下一步使可行的集合尽可能小。这显然会导致整体上的次优，因为当只考虑下一步时，一个没有达到MRD最优的词有可能通过考虑下一步和后面的步骤而达到更小的预期可行集大小。

在描述了MRD方法之后，我们现在描述一下如何在计算上实现它。幸运的是，鉴于我们用颜色矩阵 C^m 来表示游戏，这样做是很容易的。首先，请注意，对于 C^m 的某一行 i ，我们将收到一个给定颜色的概率通过简单地计算，就可以找到猜测单词 i 的代码 c 。 c 在该行出现的次数，然后按列的总数进行归一化。这是因为我们假设可行集合中的每个词作为秘密解决方案的可能性相同。让我们用 $L(c, i, C^m)$

表示颜色代码 c 在 C^m 的第 i 行出现的次数，并让

我们用 N_v 表示 C 的列数 m ，也就是当前可行集的大小。但也要注意，大小

v^{m+1}

，在猜中单词 i 和收到颜色代码 c 时，更新的可行集的长度也是 $L(c, i, C^m)$ 。那么，如果我们猜中了单词 i ，更新的可行集 v^{m+1} 的预期长度，我们将用 L_i 来表示，简单地说就是

$$L_i = \frac{1}{N_v - 1M \in C_m} \sum_{c=1}^{M-1} L^2(c, i, C^m)。$$

其中 $1M \in C_m$ 是一个指标函数，告诉我们是否

第 i 行包含颜色代码 M ，也就是GGGG，根据定义，它在一行中最多只能出现一次，而且只能出现在可行集的那些词中。因此，我们也在总和中排除了颜色代码GGGG

。我之所以选择在这种情况下剔除GGGGGG，是因为如果我们收到的颜色代码是GGGGGG，那么游戏就已经结束了，我们已经赢了，

，所以计算 L 是毫无意义的。另外，我们的理智检查将倾向于选择属于可行集的最佳词语而不是那些不属于可行集的词语，将明确涵盖这种情况，以打破僵局。

在计算了 C^m

的所有行的 L_i 之后，MRD最优的下一个猜测就是从GIF这个词的列表中抽取的，这样 $L_i \leq L_j$ 对所有 i 来说都是如此。

如果这个MRD最优列表和 v^m 之间有任何重叠，那么这个重叠的子集就是MRD最优列表。

因为找到MRD最优猜测所需要的只是计算每个颜色代码在矩阵 C 的每一行中出现的次数 m

，然后将每个计数平方，并将它们全部相加，这就可以做到非常

在计算机上快速地进行。我发现这个算法可以在我的个人电脑上实时应用，几乎可以立即确定在Wordle游戏中任何时候的MRD最佳下一步猜测。

在这一点上，扰乱警报开始了：如果你想保持你的Wordle的清白，就不要再读下去了。考虑到MRD算法，人们自然会想，Wordle的最佳开局猜测是什么？这是唯一一个在不同的游戏中没有变化的猜测，因为

viable list v^1

始终是完整的字典。鉴于我所使用的是上述的完整字典，MRD最优的

开头的猜测是.....请击鼓传花.....。

TARES

是的，TARES。就像

"每天早上，他在制作浓缩咖啡时都会用秤来测量"。

现在，如果你改用真正的Wordle词典来回答这个同样的问题，你会得到LARES这个词。是的，LARE

S。就像，"我不知道什么是LARES

的意思，但它与TARES只相差一个字母。请注意，LARES并不存在于较小的字典中。另外，按照这个标准，TARES是真正的Wordle词典中第三好的词（第二名是RALES）。

最大期望概率（GEP）算法

MRD方法的一个缺点是，它通常放弃了在每一轮中从完整的字典中随机选择解字的可能性（以前猜测的字不包括在内）。如果可行的解决方案列表在某一时刻仍然包含几十个词，这可能不会放弃什么，但在以后

，当可行的单词列表变得更小的时候，这可能会产生很大的差异。它也不直接适用于当一个人是

在硬模式下播放。人们可以通过简单地调整 C^m ，使MRD适应硬模式，如上文所述的硬模式。然而，鉴于硬模式下的每个猜测词都有可能是正确的解决方案，我们可以考虑另一种近似的优化算法是否可以在此情况下，“MDR”比“MRD”更合适。

在这里，我们考虑这样一种替代方法，我将其称为最大预期概率（GEP）算法。在这个近似的算法中，目标不是做出一个会导致最短的预期更新可行集的猜测，而是挑选一个会导致你在下一轮游戏中随机选择正确解决方案的最大预期概率的词。当然，这些概率与上面讨论的潜在更新可行集 $L(c, i, C^m$

)的长度之间存在着联系。具体来说，现在让我们定义 $P(c, i, C^m) := \frac{1}{M} \sum_{c \in C^m} L(c, i, C^m)$ 。这个数量的解释是，如果我们选择了猜测词 i ，并且收到了颜色代码 c （我们假设它不是GGGG），那么我们的下一次猜测将是正确的解决方案的概率（假设我们只从更新的可行解决方案列表中进行猜测）是 $P(c, i, C^m)$ 。当然，这个定义只对实际出现在 C^m 的*i*行的颜色代码*c*有效。那么我们可以简单地通过以下方式找到每个猜测词*i*的预期这种概率

$$\bar{p}_i = \frac{1}{N_V - \sum_{c \in C^m} L(c, i, C^m)}$$

或等价的。

$$\bar{p}_i = M_i(C^m) / (N_V - \sum_{c \in C^m} M_i(C^m))$$

其中 $M_i(C^m)$ 是出现在矩阵 C^m 的第*i*行中的唯一颜色代码的数量（不包括GGGG）。那么，GEP的最佳猜测就是猜测集合中的一个成员，即 $\arg \max_i \bar{p}_i$ 。如果有多个成员具有相同的最大 \bar{p}_i ，那么任何一个是最佳猜测。如果有任意GEP最优词也是当前可行集的成员，我们肯定会选择其中的一个。

我们现在可以问，GEP对Wordle的最佳开局猜测是什么。解决方案是

TARES
是的，TARES。如：“Wordle的GEP最佳开篇词是TARES，就像在MRD中一样！”在这里，我不会通过指出这与哪本词典有关来限定这个解决方案，因为它在两本词典中都是GEP的最佳选择

当然，一般来说， $\bar{p}_i = 1$ 的情况并不存在，所以最小的 L 不一定要对应于最大的 L 。而在真正的Wordle词典中，它确实没有。为了了解两种算法的最佳词汇之间的重叠情况，我在表1中列出了MRD和GEP的前十个最佳词汇（对于较小的词典），除了TEARS出现在两个列表中之外，其他地方没有重叠。

表1.根据MRD和GEP的最优性，前10个最好的Wordle开场词。为了好玩，我还包括了每种情况下最差的两个开场词。不，我也不知道PZAZZ是一个词。

排名	MRD	L	GEP	P
1	TARES	117.54	TARES	0.0354
2	费率	120.07	泪水	0.0350
3	故事	122.34	轮胎	0.0344
4	AAA	122.37	实验	0.0340
5	萨内尔	125.87	PARES	0.0340
6	角色	128.50	故事	0.0337
7	路径	131.13	CARES	0.0337
8	RILES	133.90	梨子	0.0337
9	泪水	135.12	PORES	0.0337
10	循环经济	136.32	板块	0.0335
5169	福兹	2148.3	JAZZY	0.0074
5170	YUKKY	2161.1	PZAZZ	0.0056

从那里归纳

虽然MRD和GEP算法是出于对我们在Wordle猜测中可能想要的东西的具体可解释性考虑，但这种限制从来没有阻止了一位数学家。因此，我们现在要把考虑一种更普遍的最佳算法，其中包括

我将把它称为*p-optimal-ity*，由数量定义的MRD和GEP。

$$f(p)_i = \frac{1}{N_V - \sum_{c \in C^m} L(c, i, C^m)^p}$$

这里，同名参数*p*一般可以是喜欢的任何实数。由于一般来说，我们可能更喜欢那些导致新的可行集规模较小的猜测，如果*p*>0，我们将选择*p*最佳猜测，即在所有*i*上使*f*(*p*)最小的词*i*，而如果*p*<0，我们将选择使*f*(*p*)最大化的词。像往常一样，我们总是会检查这个最优集合中的任何一个词是否是当前可行集中的成员，如果是的话，我们会优先选择那些就是这样的情况。那么很明显，如果我们选择*p*=1，我们就会恢复MRD，如果我们选择*p*=-1，我们就会恢复GEP。

鉴于*p*优化方法的一般性质，天空是无限的，我们可以尝试许多不同的*p*值，看看哪个是最好的。当然，对于每个*p*来说，可能有不同的最佳开局Wordle猜测。但是，我们可以研究一些极限，告诉我们在哪里可以期待什么。首先，考虑*p*=∞

的情况。在这里，唯一对(1)中的总和有实质性贡献的项是那些其中 $L(c, i, C^m) = 1$ 。由于我们在这种情况下试图使*f*最大化，这种特殊的*p*-优化只是试图找到

5作为一个有趣的数学旁观者，对于 $p \geq 0$ ，这些 $f(p)$ 与潜在可行集长度向量的向量 $(p+1)$ -norms直接相关。

有最大数量的大小为1的潜在可行集合的词，由它产生。在小词典中，这个词是PLATS。是的，PLATS。就像，“我想我知道如何在句子中使用PLATS，但现在我在现场，我发现我不能自信地这样做。”⁶

在光谱的另一端是 $p=\infty$

，在这种情况下，到目前为止，每个和中最重要的术语是 $L(c, i, C^m)$

值最大的条目。在这种情况下，我们试图使 f 最小化，所以这种特殊的 p -

优化只是试图找到其最大可能的结果可行集最小的词，基本上使最坏情况下的损害最小化。在较小的字典中，这个词是ALOES。是的，ALOES。就像，“芦荟的复数是ALOES”。

现在，真正有趣的是测试不同的 p 值，看看哪一个能在实际的Wordle游戏中带来最好的整体结果。

测试算法

为了全面评估这些算法作为Wordle的潜在最佳算法，我对每一种可能出现的Wordle对局进行了测试，在每种情况下都使用适当的最佳开局词。也就是说，对于一些特定的 p 值，当然包括GEP和MRD，我首先确定了最佳开局词，然后模拟了所有可能的秘密解法的Wordle游戏（包括正常模式和困难模式），使用 p 优化来选择每个游戏中每个回合的“最佳”猜测。然后我测量了多少轮 R_{pj} 每一个密码 j 都需要赢，这听起来可能是好像要花很长时间才能完成，但每个 P 值只需要30秒左右（硬模式更少）。

在透露结果之前，我应该提供一些更细微的澄清。首先，我应该描述一下我是如何处理在某种情况下有多个最优猜测的情况的。

圆。例如，假设我使用 $p=0.25$ ，有两个词具有相同的最小 $f(0.25)$ ，而我

必须选择其中之一作为我的下一个猜测。作为第一步，由于同时计算许多不同 p 值的各种 $f(p)_i$ 是很容易的，我将总是计算至少使用 $p=-1$ 、 1 、

10 和 10 ，并且我将打破任何平局

用这些结果来计算实际的 p 值。

其他 P 值的顺序，我在这里提出。因此，在这个关于 $f(0.25)$ 的平局的具体例子中，我会在平局的词中选择那些有最佳 $f(-$

$1)$ 的词，然后用最佳 $f(1)$ 来打破这些词中剩下的平局，然后是 $f(-$

$10)$ ，再是 $f(10)$ 。无可否认，这四个具体的 P 值（以及它们在这里的排序）在某种程度上是任意的，但它们都被选来代表容易解释的最优属性，正如上面所讨论的（因为这些选项包括MRD和GEP，而且就我而言， $\pm 10 \approx \pm \infty$ ）。在所有这些仍然不能解决平局的情况下，我就偏离了 p -

optimality，而只是选择在所产生的可行集中具有最高的 G 和 Y 颜色预期数量的词。这种选择一方面是因为我必须选择一些东西，另一方面是

因为这些是我在玩Wordle时喜欢看到的解决方案的类型（后面会有更多介绍）。如果这还是不行，那么出于类似的原因，我只尝试最高的预期 G 色数。最后，如果所有这些都不能打破平局，我就简单地按字母顺序选择第一个词；这种情况确实发生了，而且在困难模式下相当频繁（在正常模式下要少得多），因为所有可能的选择根据定义都是相似的。

第二，我选择对这些模拟游戏中的每一个进行反复猜测，直到发生三种情况中的一种。第一种情况是我碰巧猜中了正确的词，在这种情况下，我当然会停下来，并记下我刚赢的那一轮。第二种情况是我把可行的集合缩小到只有一个可能的词，在这种情况下，最优解是显而易见的，我可以停止迭代，但我仍然注意到我需要多花一个回合才能获胜。最后一种情况是，我把可行的词组缩小到两个词。这里很明显，最好的方法是随机选择这两个词中的一个（因为这两个词都不可能比另一个好），在这种情况下，我要么立即获胜，要么将可行的词组缩小到一个词，再花一个回合获胜。结果是，在实现了大小为两个的可行集合之后，我的期望是在1.5个回合中获胜。沿着这些思路，我没有把我的模拟限制在最多六轮，因为我想跟踪每个人需要多少轮，即使一个人有无限次的尝试。当然，任何回合数 $R_j > 6.5$ 对算法来说都是肯定的损失，而 $R_j = 6.5$ 的值表明，在第五次猜测之后，它有选择的余地。

划到了两组，所以至少还有可能赢。

我在表2中展示了比较几种不同 P 值的性能的结果，该表以两种方式展示了不同的 P 值：一种是获胜所需的平均回合数，另一种是测量多长时间的算法根本无法（或有可能无法）获胜。有一个特别突出的 P 值，在这两方面都相当糟糕

是 $p=-$

1.25 ，它有一些最多（和最坏）的损失，而且也有一个奇怪的高预期回合数。

至少与它相邻的 P 值相比是这样。这里值得注意的是，这个值也是唯一一个使用了一个似乎是突然出现的开头词的值。泪水。

是的，泪水。正如“算法的性能在

$p=-$

1.25 让我眼泪汪汪”。所有其他数值都使用神奇的TAR ES，否则就是PLATS（用于上文讨论的 $p=-\infty$

）或ALOES（用于上文讨论的 p

∞ ）。所以我把这里的奇怪和糟糕的表现归咎于开头的词。

现在让我们把讨论的重点放在正常模式上。这里一个有趣的发现是，对于中间的 p 值，在损失和平均回合数之间似乎有一个权衡。例如，最低的平均回合数是 $p=-0.75$ ，但该算法也明确地输了两轮，可能输了六次。在另一个方面， $p=0.75$ ，从未确定输过，只可能输过两次，但它的平均数量要高一些。

圆的。GEP和MRD之间的比较也显示出

6 尽管《韦伯斯特大学词典》给出了该词的十个不同含义。

表2.在正常模式（用上标表示）和硬模式（用上标表示h）下，不同 p 值的 p 优化算法的性能。标有 R_{pj} = X的列给出了观察到该特定 R_{pj} 的频率。请注意标有星号的在 $R^h = 7$ 一栏中， $p = -1.25$ 。这表明实际上有六个 $R^h = 7$ 和两个 $R^h = 7.5$ 的实例（都是其他行的最大值为 R^h $p_j = 7$).

p	开瓶器	平均 R^h_{pj}	$R^h_{pj} = 6.5$	$R^h_{pj} = 7$	平均 R^h_{pj}	$R^h_{pj} \geq 6.5$
-2.00	计划	3.8097	6	3	3.9162	207
-1.75	计划	3.8070	4	3	3.9137	208
-1.50	计划	3.8039	4	3	3.9083	203
-1.25	泪水	3.8236	16	8*	3.9431	249
-1.00	TARES	3.7752	6	2	3.8781	183
-0.75	TARES	3.7712	6	2	3.8737	183
-0.50	TARES	3.7739	2	2	3.8760	189
-0.25	TARES	3.7733	4	2	3.8731	190
0.25	TARES	3.7843	4	0	3.8903	200
0.50	TARES	3.7901	4	0	3.8959	196
0.75	TARES	3.7983	2	0	3.9056	200
1.00	TARES	3.8008	2	0	3.9124	208
1.25	TARES	3.8056	2	0	3.9186	208
1.50	ÄÄÄ	3.8851	2	3	4.0451	286
1.75	ÄÄÄ	3.8905	2	3	4.0576	306
2.00	ÄÄÄ	3.8940	2	3	4.0578	305

这种权衡，MRD需要更多的回合才能获胜，但输的次数较少。 $p=-$ 0.5这个值似乎提供了一个很好的中间地带，平均回合数几乎是最小的，最多只输四次。为了提供更详细一点，我已经挑出了具体的价值。 $p=-0.5$ 为强势表现，并绘制了一个直方图图1（a）中这种情况下的 R_{pj} 。如果你有兴趣，你为什么感兴趣呢。在导致算法失败状态的词方面有一个明显的趋势。有40个词以失败告终，所有 p 值中最常见的失败词是FAZES，其次是FAXES和HAZES。一般来说，许多（14/40）失败的词都遵循_A_E S的规律，包括那些在损失最少的情况下仅有的两个失败的词（BABES和HAZES）。困难模式呈现出某种程度上的不同情况。在这里， $p=-$ 0.75几乎是一个明确的赢家，因为它是损失最少的（与GEP并列），并且有几乎最小的预期回合数，只是稍稍落后于GEP。 $p=-0.25$.图1(b)显示了 $p=-0.75$ 的结果柱状图。也许毫不奇怪，GEP在硬模式下令人信服地击败了MRD；毕竟GEP是专门为硬模式设计的。除此之外，关于硬模式没有太多可说的，至少相对于正常模式而言，它确实是硬的。为了向真正的Wordle词典表示敬意，我计算了相当于表2中的正常模式的结果，用于这个更大的词典。结果显示在表3中。值得注意的一点是，现在TARES在更少的情况下是最佳的开局者，神秘的LARES在大多数使用的正 P 值中取代了它。然而，在 $p=-0.5$ 的情况下，TARES仍然获得了预期获胜回合数最少的奖项，这也是在损失最少的 p 值方面表现最好的一个。有趣的是

尽管在这种情况下，字典要大得多，而且对于许多 p 值来说，给出的最佳开局明显不同，但这里仍然是 $p=-0.5$ 成为总冠军。最后，显然我们对这本字典有 $\pm 2 \neq \pm \infty$ ，因为渐进的最佳开局是VENAL和SERAI，因为 $p \rightarrow \pm \infty$ ，分别。

一个更详细的方法

虽然上面的 p 优化方法在计算上非常快，而且对于 p 的明智选择来说似乎效果很好，但该方法至少有一个方面是有点欠缺的。也就是说，该方法只关注下一个可行集的可能长度，而没有注意下一个可行集可能具有的任何其他品质。虽然这显然是一个弱点，但值得探讨的是，什么样的品质潜在的可行集可能是有意义的，以帮助指导更精细的技术的发展。为了开始讨论，我想定义一下我所说的完全可辨识集（FDS）。一个完全可辨识的集合是任何可行的集合，其中至少存在一个"关键词"，以该关键词作为猜测词，可行集合中的所有词都会返回不同的颜色代码。等价地，在矩阵C中存在一行 m ，使得该行中对应于可行词的所有条目都是不同的。我称这是一个完全可辨的集合，因为如果你的猜测词是关键词，所产生的颜色代码将立即让你完全辨别出正确的答案是什么（该行中包含该颜色代码的唯一列所对应的词）。 m 使用上面的一些术语，当前可行的集合是一个FDS，当且仅当矩阵C的第 i 行 i ，这样的 $\mathbb{N} \times (C^m) =_{M_N}$ ，在这种情况下，词的对应---ing to row i 是FDS的一个关键词。鉴于这个定义。

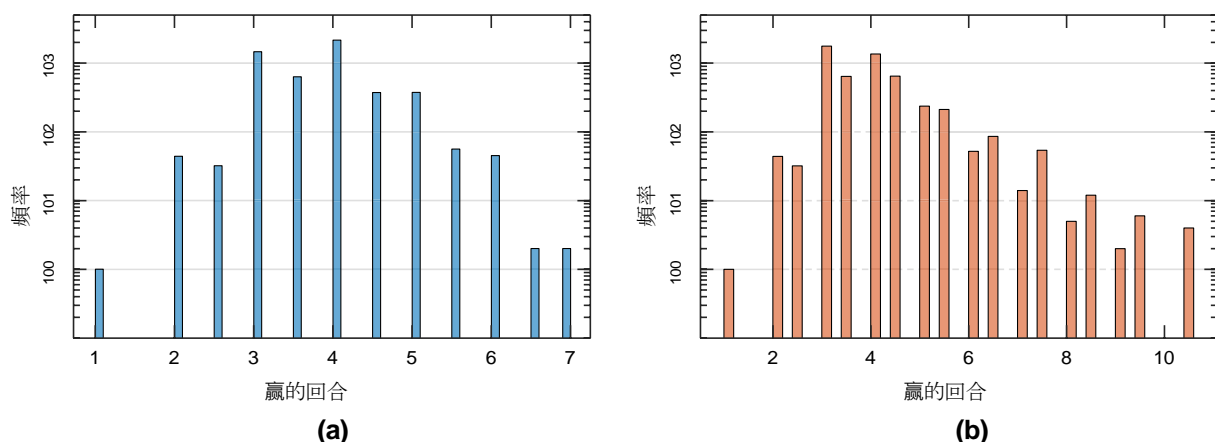


图1.在(a)正常模式下使用 $p=-0.5$ 的最优性和(b)困难模式下使用 $p=-0.75$ 的最优性，在每一种情况下都从最优开局词TARES开始，使用较小的字典赢得Wordle的每一局所需的回合数。

表

3.在正常模式下，各种 p 值的 p 优化算法的性能（用上标表示），当用于真实的Wordle词典时。最后两栏给出了观察到 R_{pj} 的指定值的频率。

p	开瓶器平均值 R^a	$R^a = 6.5$ $R^a \geq 7$		
		p_j	p_j	p_j
-2.00	PLEAT	4.1856	62	58
-1.75	PLEAT	4.1691	62	39
-1.50	PEATS	4.1600	50	39
-1.25	PELAS	4.1464	48	61
-1.00	TARES	4.0889	36	16
-0.75	TARES	4.0859	36	15
-0.50	TARES	4.0807	26	10
-0.25	TARES	4.0827	30	13
0.25	TARES	4.0910	42	6
0.50	LARES	4.1156	22	13
0.75	LARES	4.1240	24	13
1.00	LARES	4.1263	26	10
1.25	LARES	4.1327	24	10
1.50	LARES	4.1378	26	12
1.75	LARES	4.1452	28	12
2.00	LARES	4.1479	28	12

请注意，一个FDS的最大可能大小是 M ，因为这是 $M_i(C^m)$ 的最大可能值。

在所有FDS中，一般有两个品种。内部-FDS (I-FDS) 和External-FDS (E-FDS)。对于一个可行的集合来说，是I-FDS仅仅意味着存在一个该FDS的关键词，它是FDS的成员。典型的情况是：所有大小为2的可行集合都是I-FDS，集合中的两个成员都是关键词，因为选择任何一个成员都会导致立即胜利或完全知道正确的解决方案。反之，E-FDS是一个FDS，其中没有一个关键词是FDS的成员。

在玩Wordle游戏时，假设你发现当前可行的集合是一个I-

解决方案，很明显，在这一点上你能做的最好的事情就是选择（内部）关键词。

这将导致你立即获胜，概率为 $\frac{1}{N}$ 。如果不是，你在下一轮肯定会赢。但如果是E-FDS呢？那么你一般有两个貌似合理的选择：挑选关键词（不能导致

你会立即获胜，因为它是可行集合的外部），然后在下一轮肯定会获胜；或者选择集合中的一个成员，抓住你现在获胜的机会，即使你没有，你也会保证在下一轮仍然处于FDS中，因为FDS的每个子集显然也是FDS。如果E-FDS的规模越小越好，即3，这也许是最诱人的，从而使你立即获胜的机会最大化。但即使在这种情况下，在预期中，这种偶然的选择也不会比简单地选择外部关键词更好。通过选择外部关键词，你肯定会在两轮中获胜。通过选择内部集合的成员，你在一轮中获胜的概率为三分之一；否则你就只能选择I-FDS（一个大小为2的集合），意味着你在两轮中获胜的概率为三分之一，在三轮的的概率为三分之一，导致预期两轮获胜。没有一个 $N_v > 3$ 的E-FDS能比这更好，所以在这些情况下你也不可能比选择外部密钥做得更好。因此，我们只需将选择，当遇到E-FDS时，我们采取

FDS。同样，在假设所有的词都同样可能是秘密的情况下

的肯定，而只是选择关键词，在这种情况下，不管它的大小，它大致上是与一套可行的长度三相称的。

当然，我们可以从FDS的概念推导出更高阶的概念。例如，假设我目前的可行集不是一个FDS，但存在 C 的第 i 行 m ，从而

所有可能出现的 v^{m+1} 集，都应
我选择 i 是自己的FDS。那么，人们可以把
当前可行的集合是一个阶-

1可辨识集合（1DS），它可能是内部的或外部的。同样地，人们可以设想一个阶-

k 可辨识集（ kDS ），这样它就不是一个FDS，但存在一个词，使所有可能的

由该词产生的可行集合 v^{m+1} ，本身就是阶数为 $(k-1)$ 或更少的可辨识集合。那么，这就是开始

摒弃贪婪的算法，因为我们在讨论 k 阶可辨识集合时，往往要提前多步考虑。出于这个原因，随着 k 的增加，计算一个集合是否是 k DS很快就变得不可行了。这里有一个不过，有一个赠品：一个长度为 $N_V=3$ 的可行集合，如果本身不是FDS，也自动是I-IDS。这是因为通过挑选该集合中的任何成员，你要么立即得到解决方案，要么就只剩下一个长度为2的可行集合，这就是I-FDS。

我要指出的是，在FDS的概念和 p 最优策略之间存在一种很好的关系。具体来说，如果一个集合是一个FDS，那么就会自动出现这样的情况，即关键词 i 会对每一个人产生 p 最优解 $p \neq 0$ ，因为该词的每个 $L(c, i, C^m)$ 都将等于1，这将导致 $f(p)$ 是最小的（或最大的。取决于 p 的符号），它可能是。更进一步。如果该集合是一个I-FDS，那么内部关键词将是最佳选择，因为我们总是检查在当前可行的集合中是否有一个最小化器，如果有，就选这个。

在定义了一堆东西之后，现在让我描述一下如何利用它们来尝试获得一个修正的最佳Wordle策略。考虑到上面的讨论，我们注意到E-FDS本质上等同于一个大小为3的集合，而I-FDS类似于一个大小为2的集合（但也许不那么好，取决于它的大小），我们可以修改我们的 p -optimal算法以考虑到这些事实。具体来说，我们首先像往常一样，计算对于 C 的每一行 i^m

，所有潜在可行集的长度按颜色代码返回， $L(c, i, C^m)$ 。然后对于出现在第 i 行的每个这样的颜色代码，让我们定义一个新的值 $L \sim(c, i, C^m)$ ，定义如下。如果 $L < 3$ 或 $L > M$ ，则 $L \sim = L$ 。如果 $3 \leq L(c, i, C^m) \leq M$ ，则检查该集合以确定它是否是某种FDS。如果该集合是一个E-FDS，那么让 $L \sim = L$ 。 $L \sim = 3$ ，以捕捉到这样一个事实，即就获胜的回合数而言，它基本上与长度为3的集合相同。最后，如果该集合是一个I-FDS，那么让

$$L \sim = \frac{2}{L} + \left(1 - \frac{2}{L}\right) 3,$$

来捕捉这样一个事实：这个I-FDS的预期获胜回合数等于大小为2和3的集合的获胜回合数的这个特定加权平均数。然后我们可以简单地修改我们的 p 最优算法，使用 $L \sim$ 而不是 L^p ，从而得到我们的 p -FDS算法。

$$FDS(p)_i = \frac{1}{M-1} \sum_{c=L, c \in C^m} L(c, i, C^m) L \sim(c, i, C^m) \quad (2)$$

在这里，我们将再次使 $FDS(p)_i$ 在 i 上达到最小或最大。取决于 p 是正还是负。

所有这些在纸面上都很容易说明，但在计算上开始变得有点慢。幸运的是，至少对于较小的字典来说，使用 p -FDS算法的计算是可以承受的，⁷所以我可以提供一些结果。对于 $p=-0.5$ 这个值，选择它是因为它的好如上所述， p -FDS的最佳开局词是，你猜对了，TARES。真是个好词。此外，我在正常模式下模拟了这个特定 p 值的所有可能的Wordle游戏。在这种情况下，获胜的平均回合数为3.7696，比 $p=-0.5$ 时从标准 p 优化中得到的3.7739略好。 p -FDS方法在以下情况下没有任何损失7轮的水平，与 p -最优方法不同，它有两个。但是 p -FDS在6.5轮水平上确实有6个词失败，而 p -optimal值为2个词；但是再次注意，6.5轮水平上的失败只是可能的失败，而不是保证的失败。对于那些好奇的人来说，六个可能的失败词是BAKES、BASES、FAKES、HAKES、WAKES和WASES，都遵循上面提到的_A _ E S模式。如果我们直接比较单词， p -FDS获胜在742个词中， p -优化比 p -优化严格地以更少的回合获胜，而在702个词中， p -FDS严格地以更少的回合获胜；其余的是并列。因此，在我看来， p -FDS显然在这里夺得了桂冠，但有一点需要注意的是有一个潜在的失败，即有四个以上的词比 p -optimality（但不保证失败）。

所以，我们应该总是先使用TARES，对吗？

也许吧。我认为这个问题的答案取决于你的大脑与电脑的相似程度。可能它并不那么相似，所以它不会同样好地使用所有种类的信息。例如，在我自己玩的Wordle中，我发现最有用的猜测是那些返回带有G和Y的颜色代码的猜测。这是因为对我来说，列出我知道包含某些字母的单词比列出不包含某些字母的单词要容易。算法

当然，在这里不做任何区分（除了从我的打破平局的方法来看，这是在考虑到这种偏好的情况下选择的），所以他们可能会建议一些对实际玩游戏的人来说不那么有用的词（假设人在整个过程中不会继续使用解题工具）。

有鉴于此，选择一个开场白，在返回的颜色代码中平均包含最多数量的G和Y颜色，也不是没有道理的。这很容易。给出预先计算的 C^1 ，以确定哪个词是这样的。

解决方案是AROSE，平均为1.924个G加Y。该神奇的TARES与其他五个词并列第四位通过这种核算，平均为1.894G加Y。这与AROSE相当接近，所以考虑到上述所有结果，也许TA

RES8仍然是最好的。

7在我的代码的早期版本中，速度相当慢，预计使用 p -

*FDS*计算Wordle的所有可能的游戏需要9天左右。我把它设置为在我工作的电脑上后台运行，然后继续我的生活。不幸的是，在计算的一周后，我的办公大楼遭遇了一次非常罕见的停电，所有的结果都丢失了。我把这个悲剧理解为一个信号，，并继续优化我的代码，现在可以在大约三个小时内完成同样的计算。

8嘿，我刚刚注意到你可以只用一只手来打TARES。

为了完整起见，我还将在此提及平均G色数最高的词，以及平均Y色数最高的词；它们是SANES（平均有0.9255个G）和RESAT（平均有1.5712个Y）。在这些名单上，TARES分别出现在第19位和第849位。哇，849，是吧？

另一件需要考虑的事情是我们在本文开始时所作的一些假设。正如已经指出的，Wordle肯定不会每天从完整的字典中随机选择一个秘密解决方案。我也很确定，Wordle从子词典中选择可能的解决方案的词，一般都包含“常见

”的词，这样就不会让玩家太难过。在我玩Wordle的过程中，我见过的最

“晦涩”（但肯定没有PZAZZ那么晦涩）的词是ABBEY和KNOLL。当然，我希望大多数说英语的人在玩Wordle时都熟悉这两个词，即使它们不是人们在考虑五个字母的单词时首先想到的。但是，如果有一天Wordle的解决方案是YUKKY，也许会有场反抗。是的，YUKKY。就像“Fozzie Bear的喜剧非常YUKKY”⁹。

当然，如果在某一时刻，人们知道了Wordle实际上是通过某种概率分布 q_j 生成或产生其子字典的解决方案，其中 q_j 是字典中的单词 j 被选为的概率。对于一个给定的游戏，上面的算法可以进行调整。在每个猜测水平 m 所需要做的就是计算当前（或推测，如果你是潜在的下一个可行集做的）相关可行集中每个词的概率 q^m ，只需将已知的 q_j 用于每个词，然后将每个词归一化，使其总和为1。因此

$$q^m = \sum_{j \in \mathcal{C}^m} \frac{q_j}{\sum_{j \in \mathcal{C}^m} q_j}.$$

那么我们上面的公式将使用，代替当前的概率 $L(c, i, \mathcal{C}^m)$ ，值

$$W(c, i, \mathcal{C}^m) = \sum_{j, C_j^m = c} q_j^m. \quad (3)$$

当然，在这种情况下，其他修改也可能有很大的意义。例如，如果两个大小相同的集合所包含的字的概率分布非常不同，那么它们可能不再是几乎等同的。作为一个例子，假设一个由四个词组成的潜在可行的集合的概率分布是非常

而另一个由四个词组成的潜在可行的集合对所有四个词都有相同的概率能力。在这种情况下，你肯定会更喜欢概率分布不等的那一组。

因为这将使它更像一个只有一个词的集合，而不是四个。因此，在调整算法以适应这些更普遍的每个字的概率时，可能比 L^p

更好的选择会有意义。不幸的是，这样的探索将不得不等到下一天，否则我将永远无法完成这篇笔记。这一讨论自然引出了我们的结论部分。

可能的扩展和未来的工作

所有的学术论文都以这样的章节结束，所以这篇也会，尽管它不是一篇正常的学术论文。

从这项工作中直接得出的一个自然的潜在扩展是构建一个在每轮游戏中都不恒定的时变策略。也就是说，人们可以清楚地（和有效地）计算出在每个猜测水平上对许多不同的 p 值的最佳话语，然后尝试使用某种启发式方法来决定对该猜测使用哪一个。目前我还没有对这种可能性进行足够详细的探讨，因此无法对它说些特别聪明的话，但我认为这是一个明显的下一步行动。

更大的困难，但也可能有更大的回报，就是研究比上述方法更“全局

”的方法，因为它们在一个时间内考虑多个步骤，而不是只有一个步骤。 p -

FDS算法是朝着这个方向迈出的第一小步，因为它有效地展望了某类可行的集合，其结果很容易事先预测，但更多的是

在这里，我们仍要做的。同样，虽然我还没有深入思考过这个问题，但提前两步看问题可能也不是那么可怕的。正如上面的结果所表明的那样， p -

optimality似乎是解决Wordle的典型方法（从技术上讲，我自己的Wordle实现肯定能解决这个问题）。

符合我们在本文中所作的假设）在三个或四个步骤中，提前两步看可以让你很好地走向解决方案，而且可能相当有用。

另一个扩展是将这里的方法应用于使用其他长度的单词的替代Wordles。

比五个。当然，鉴于所选择的长度，这些方法从计算的角度来看，上述做法可能是完全不可行的，因为可能的词的数量可能要多得多。例如，鉴于我的字典来源

词，有8,459个6个字母的词，11,934个7个字母的词，13,057个8个字母的词，12,009个9个字母的词，并且从那里开始似乎有下降的趋势。关于

在光谱的另一端，有2,593个四字母词，667个三字母词和77个二字母词。（最后一个数字在我看来很高。）这些字母数较少的词可以证明是一个有趣的测试平台，可以看到某些算法能找到真正的最优，因为在这些更短的列表（字母数较少的词）上用蛮力找到真正的全局最优会更有说服力（但可能还是不那么容易）。

⁹至少我是这样使用它的。在线词典通常将其列为YUCKY的替代拼法，我认为这是在错过一个黄金语言学机会。

最后，再考虑一下上面讨论的可能性，即任何给定的Wordle的解决方案都是为了
是从一个给定的概率分布中抽取的。如果Wordle的设计者/维护者有此意愿，他们可以尝试设计这个概率分布
在这样一种方式下，以最佳方式挫败我的算法

解算器（或其他可能存在的解算器）。
这将使Wordle的设计者/维护者与像我这样的人进行
全面的游戏理论对决。
战斗，这可能是非常有趣的尝试解决，在两端。

马丁-B-肖特，佐治亚理工学院数学学院，686 Cherry
Street, Atlanta, GA 30332-
0160，美国。电子邮件：mbshort@math.gatech.edu

出版商提示 《斯普林格-
自然》杂志对出版的地图和机构隶属关系中的管辖权要求保持中立。