根据混合物的单调分解对文件方言进行无监督聚类

Michael Robinson Tate Altman 数学与统计学系 数学与统计学系 美国大学 美国大学 华盛顿特区 华盛顿特区 电子邮件：michaelr@american.edu 电子邮件：ta8427a@american.edu

Denley Lam BAE Systems FAST Labs 弗吉尼亚州阿灵顿 电子邮件：
denley.lam@baesystems.com

Letitia W. Li BAE Systems FAST Labs 弗吉尼亚州阿灵顿
电子邮件：
letitia.li@baesystems.com

arXiv:2304.09082v1 [cs.PL] 9 Feb 2023

摘要——**本文提出了一种无监督分类方法，根据文件的行为将文件集划分为互不重叠的方言；这些行为由一组使用这些文件的程序所生成的消息来确定。消息模式可用作某类行为的特征标识，同时应认识到，某些消息可能倾向于共同出现，而另一些则不会共同出现。**

消息模式可用于将文件分类为不同方言。方言由一组消息的子集定义，这些消息称为必需消息。一旦以方言及其必需消息为条件，剩余消息在统计上相互独立。

基于上述方言定义，我们提出一种贪心算法，可从由文件-消息数据矩阵构成的数据集中推导出候选方言；我们通过多种文件格式展示该算法的性能，并证明其达到最优的条件。我们还表明，分析人员需要考虑的方言少于不同消息模式的数量，从而减轻了研究复杂格式时的认知负荷。

I. 引言

尽管人们已投入大量精力，试图将符合格式规范的含义形式化，但程序在读取文件时的行为才决定了特定格式的终端用户体验。以行为视角理解文件格式的优势在于，它适用于统计分析：在已观察到其他行为的前提下，我们可以为特定文件引发特定行为的条件概率赋值。

从行为角度来看，这与测试驱动设计的理念十分契合，因为能够引发不良行为的文件很容易被识别为测试用例。因此，整理符合格式规范的文件数据集是格式分析师的一项重要工作。按理说应符合某一临时格式规范的文件，实际上可能属于若干方言之一，其中可以观察到不同的行为模式。在尝试构建用于读取特定格式文件的程序时，如何处理不同方言之间的行为差异是一个棘手问题。为了获得足够的测试覆盖率，必须确保测试样本中包含所有方言，这意味着分析师必须先了解其数据集中哪些文件符合哪些方言。通过将文件格式划分为不同方言，遵循 LangSec 方法的解析器开发人员可以为每种方言编写语法，从而开发出更全面的解析器；也可以正式定义其语法应覆盖某种格式的哪些方言。

A. 贡献

本文提出一种无监督分类方法，根据文件的行为将文件集划分为互不重叠的方言；这些行为通过一组布尔特征的出现情况来衡量，这组特征称为消息。消息模式可用作特定行为类型的特征标识，其依据是某些消息可能共同出现，而另一些则不会。

我们的方法基于一种针对行为方言的新颖统计定义。方言由一个消息子集定义，称为 必需消息，该子集满足一项统计假设。一旦以某种方言及其必需消息为条件，剩余消息在统计上彼此独立。第 II 节详细阐述了这一定义的含义。

我们对方言的定义导出了一种贪心算法，该算法根据由文件-消息数据矩阵构成的数据集推断候选方言。该算法体现在第 IV 节命题 3 的构造性证明中。

只要有足够的消息，我们提出的方法就能处理任何格式的文件。为突出这一点，我们在第 III 节探讨了该方法在三种截然不同的格式上的表现：表格数据（CSV）、自由格式文档（PDF）和图像（NITF）。

最后，我们证明，在满足某些合理的统计假设后，我们的方法在产生最少多余方言这一意义下是最优的。

已批准向公众发布；不限制分发。根据 ES-FL-012723-0011，不受出口管制。

这些最优性证明见附录第 VI 节中的定理 1 和 2。

B. 局限性

我们的方法依赖于对所考虑格式的良好消息覆盖率。如果消息并非由我们关注的行为触发，那么表现出这些行为的文件就无法被检测到。幸运的是，我们在第 III 节中考虑的大多数现有即兴格式解析器都会向标准错误和标准输出生成大量输出。这些输出具有足够的标准化程度，因此可以使用正则表达式来收集输出并组成消息。

如果恶意行为者知道正在使用的消息集合，我们的方法可能会因其构造文件以避免生成某些消息而被规避。消息正则表达式可以通过半自动方式构建， [1],对此进行了详细说明；因此，获取足够多且具有充分多样性的消息并不困难，从而可以防止文件逃避正确分类。还可以构造与系统调用相对应的消息，以产生额外的行为多样性 [2]。据观察，似乎很难构造出能够避免某一类型的所有消息的文件。

我们的统计模型是非参数独立混合模型的一个特例。独立混合模型在相关文献中已得到充分研究，并且存在许多具有深厚理论基础的算法（例如 [3； 4] 以及许多其他算法）。遗憾的是，这些算法往往会作出不适用于文件消息数据语境的假设，例如假定混合成分服从某种已知分布，或假定混合成分（方言）的数量从一开始就是已知的。期望最大化是一种常见策略，可以避免作出这些假设，至少在特征（消息）数量较少时如此。

我们的问题涉及数百个特征（消息），这使得期望最大化在计算上不可行 [3, 1.3.1]。为管理复杂性，我们将独立性放宽为单调性条件。因此，我们无法获得最佳的混合分解，而只能获得其界限，参见定理 2。由于我们的实现使用贪心算法，候选方言会存在一定歧义，不过这种歧义影响相当有限。当多个消息在统计上相互依赖时，不必将它们全部选作某个方言所要求的消息，详情参见引理 8。

C. 相关工作

本文延续了 LangSec 社区过去几年开展的一系列工作，这些工作从统计角度审视格式规范 [1；5； 6]。与 [1], 中的假设不同，即在给定方言的条件下，特征彼此独立，本文还以每种方言都必须包含的一组消息为条件。当合并来自不同解析器的消息时，其中一些消息实际上是完全相同的特征。例如，两个解析器可能会输出同一种语法错误，从而产生两条不同但在统计上相关的消息。尽管这种行为直接违反了独立性假设，却被忽略了 [1] 。通过以其中一条、另一条或这两条消息为条件，我们可以恢复独立性。

统计格式分析似乎属于少数观点，因为大多数文件格式分析使用文件内容的结构，而不是解析器对这些内容的响应（例如，参见 [7；8；9；10； 11]）。然而，基于文件操作的统计特征也被用于识别某些恶意行为 [2]。

我们受到 [12], 的启发，其中发现了39种 CSV 文件方言。我们在第 III 节中将方法应用于同一数据集的一个简单随机样本，发现其中最常见的是由14种方言组成的一个相对更粗的集合，不过也存在许多不太常见的方言。

从数学角度来看，我们所使用的方法基于偏序集和 *Dowker* 复形[13]。近期研究已将 Dowker 复形与表格数据 [14] 以及形式概念分析 [15]联系起来。

II. 文件格式方言及其行为的统计模型

我们的数据由文件集 F 构成，这些文件由各种程序解析后，可能产生某个消息集合 K 中的任意消息。对于每个文件，每条消息 $k \in K$ 要么确实出现（此时我们说 $k = 1$），要么不出现（表示为 $k = 0$）。尽管解析给定文件通常是确定性的，但我们可以将给定消息出现的可能性建模为概率 $P(k = 1)$。概率模型避免了逐个处理特定文件，因此我们很少需要直接处理文件集 F 。

在 [1； 5],中，研究表明，考察一组消息是否发生的联合概率， $P(k_1 = 1, \ k_2 = 1, \ldots, \ k_n = 1, \ k_{n+1} = 0, \ldots)$，是识别某些感兴趣文件的一种有效方法。在这一概率框架下，事件由一个消息模式构成，即可能发生的消息的一个子集。

单独研究任意联合概率并非易事，不过数据往往支撑着有用的统计假设，为理论分析提供切入点。在 [1], 中，假设给定子集 $A \subseteq F$ 内文件的消息彼此独立。当消息在语义上互不相关时，这一假设是合理的；但若某些消息彼此相关，则不适用。本文采用一种更精细的方法。

定义 1。方言是文件的一个子集 $A \subseteq F$ 和消息的一个子集 $R_A \subseteq K$，满足在同时以 R_A和 A为条件后，其余消息彼此独立。子集R_A 称为方言 A的必需消息集。

具体而言，若我们将 $R_A = \{K_{i+1}, \ldots, K_n\}$ 写作消息模式 $K_1 = k_1, \ \ldots, \ K_n = k_n$ on

已获准公开发布；不限制分发。根据 ES-FL-012723-0011，不受出口管制。

文件 in A 中的形式为

$$P(k_1,k_2,\ldots,k_n|A)=\begin{cases}0 \text{ if } k_j=0 \text{ and } K_j\in R_A,\\ P(k_1|A)\cdots P(k_i|A)P(k_{i+1}=1\ldots|A) \\ \text{otherwise.}\end{cases} \tag{1}$$

如果两条消息实际上表达相同的含义，那么我们可以将其中一条、另一条或两条都视为必需消息。这意味着方言在有限程度上存在歧义，但正如我们在附录第 VI 节的定理 1 和 2 中所证明的，这种歧义相当良性。简而言之，方言数量的界以及方言本身都可以通过算法获得。

A. 带有必需消息的条件独立混合物

考虑所有消息的集合 K。幂集 2^K of$K$ 由 K的所有子集组成。幂集按子集 $\subseteq$ 进行偏序，因此（2^K，$\subseteq$）是一个偏序集。等价地，我们可以将 K 的每个子集视为长度为 #K的二进制序列。公式（1）可以视为为每个方言 A定义一个函数 $P(\cdot|A)$：$2^K\to[0,1]$。顺带一提，由文件数据集中的K 子集构成的偏序，是 Dowker 复形 [13] 的面序的一个子序。

根据 [1, 引理 1],中已证明的内容，我们可以预期，一旦必需消息出现，由公式（1）定义的概率会随着更多消息的出现而降低。也就是说，随着更多的 k_1，…，k_i取值为1，概率会降低。

引理1。假设在某个方言中 A, $P(k_j=1|A)<1/2$ 对每条消息 K_j均成立。由公式（1）定义的概率函数也可以写成

$$P(k_1,\ldots,k_n|A)=1_{U_R}(k_1,\ldots,k_n)g(k_1,\ldots,k_n),$$

其中

1） R 是 A的必需消息集合，2） $U_R=\{B\in 2^K: R\subseteq B\}$ 是包含 R的所有消息模式的集合，3） 1_{U_R} 是定义在 U_R上的指示函数，并且4） g：$2^K\to[0,1]$ 是单调递减的。

证明。公式（1）规定，如果对于某个 $j\in\{i+1\ldots,n\}$有 $k_j=0$，那么 $P(k_1,\ldots,k_n|A)=0$。这意味着，$P(\cdot|A)$ 的支撑集包含于定义在该集合上的指示函数的支撑集内

$$R=\{K_{i+1},\ldots,K_n\}$$

必需消息。

另一方面，由于对于每条消息 K_j，$P(k_j=1|A)<1/2$，这意味着如果我们省略一条非必需消息，概率就会降低

$$\begin{aligned}P(k_1,\ldots,k_n|A)&=P(k_1|A)\cdots P(k_i|A)P(k_{i+1}=1,\ldots|A)\\&<P(k_1|A)\cdots P(k_{j-1}|A)P(k_{j+1}|A)\cdots\\&\quad P(k_i|A)P(k_{i+1}=1,\ldots|A)\\&<P(k_1,\ldots,k_{j-1},k_{j+1}\ldots|A).\end{aligned}$$

换句话说，在 U_R 内，概率是一个单调递减函数。 □

如果存在多个方言，消息模式出现的概率具有相当明确的形式。

推论1. 假设在某个方言中 A, $P(k_j=1|A)<1/2$ 对所有消息 k_j均成立。假设各方言互不相交，则所有文件中消息的联合分布为

$$\begin{aligned}P(k_1,k_2,\ldots,k_n)&=\sum_A P(k_1,k_2,\ldots,k_n|A)P(A)\\&=\sum_A 1_{U_{R_A}}(k_1,\ldots,k_n)g_A(k_1,\ldots,k_n),\end{aligned}$$

其中，$1_{U_{R_A}}$ 和 g_A 是引理*1*的语句中定义的、与方言 A相关联的函数。

命题1. 假设在某个方言中 A, $P(k_j=1|A)<1/2$ 对所有消息 k_j均成立。在公式(1)所给的模型下，一个方言的支撑集（即其概率非零的消息模式集合）具有唯一的最小出现消息数，即必需消息。

证明。根据引理1，对于一个方言 A,

$$P(k_1,\ldots,k_n|A)=1_{U_R}(k_1,\ldots,k_n)g(k_1,\ldots,k_n)$$

对于单调递减的 g。这确保了该方言的支撑集包含于 U_R之内。

反证法。假设存在两个出现的极小消息集合。这等价于说，在必需消息 R 中至少存在两个真子集 $S_1\subset R$ 和 $S_2\subset R$，使得 $P(S_1|A)$和$P(S_2|A)$均非零，但 $P(R|A)=0$。注意，根据构造，

$$P(S_1|A)=g(S_1),$$

and

$$P(R|A)=g(R).$$

我们刚刚证明了 $g(S_1)>g(R)$，然而这违反了单调性，因此产生矛盾。 □

III. 实验结果

命题1给出了联合消息概率的一种分解，将其分解为由必需消息的极小集合所确定的方言。假设这些分解可以获得——第四节描述了构建这些分解的一种算法——本节讨论如何将给定的文件集划分为语义上有用的方言，以便通过其他方式进行进一步探索。

A. CSV

乍看之下，朴素的逗号分隔值（CSV）文件格式似乎完全可以从其名称推断出来。它是一种用于表示表格数据的文本文件格式，由按行和列分组的单元格组成。每一行对应文件中的一行，并由少数几种行结束符之一分隔。每一列则由逗号字符分隔。这种简单的描述很快就会遇到问题，因为

已获准公开发布；不限制分发。根据 ES-FL-012723-0011，不受出口管制。

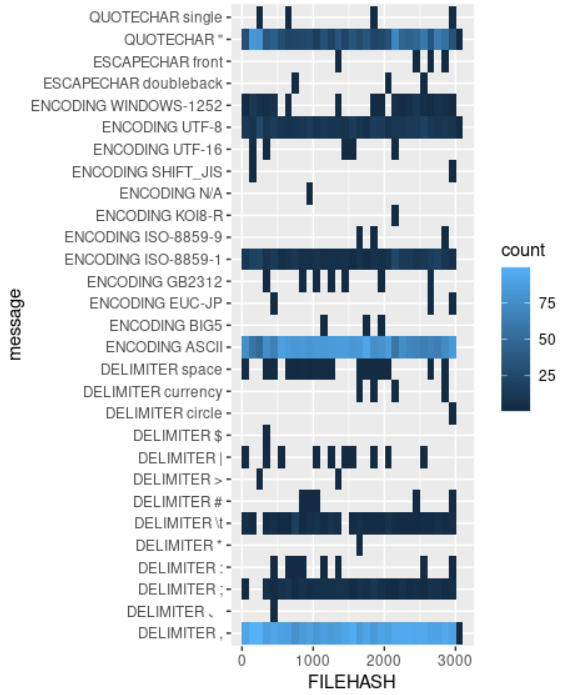


图 1。CSV 数据摘要：行对应可能的消息，而列对应样本中的文件。灰色单元格表示相应文件中未出现该消息，阴影单元格则表示该消息已出现。由于水平分辨率的限制，文件被汇总到分箱中，并在其中显示计数。

“逗号”的具体含义会因语言和文本文件编码而异 [12]。此外，由于单元格中可能包含行尾或逗号等分隔符，因此需要某种引号处理方式。引号字符同样会因编码而异。最后，由于 CSV 文件通常由电子表格应用程序读取，因此其中可能包含公式或可执行代码片段，并以出人意料的方式产生影响 [16]。

为展示 CSV 文件语料库中存在的方言，我们从 [12] 中所述的数据集中简单随机抽取了 3005 个文件。对于其中每个文件，我们提取了由同一篇文章中所述的 CleverCSV 工具获得的共 33 条消息。这些消息总共包括

- 14个分隔符，
- 3个引号字符，
- 3个转义字符，以及
- 13种不同的文本编码。

图 1 展示了通过此过程获得的数据。每行对应一条不同的消息，每列对应一个不同的文件。图 1 中共有 29 行，因为在我们的样本中，有 4 条消息（1 个引号字符、2 个转义字符和 1 个分隔符）未出现在任何文件中。

图 1 中的水平条纹表明，有些消息非常常见（例如 ASCII 编码和 ASCII 逗号）。表 I 记录了

表 I CSV 数据中最常见的消息模式的文件数量

文件数量	消息模式
1417	, ASCII
682	, " ASCII
196	, " UTF-8
156	, " ISO-8859-1
119	, UTF-8
64	, " WINDOWS-1252
55	; ISO-8859-1
51	TAB ASCII
48	TAB ISO-8859-1
45	; ASCII
29	ASCII
18	space ASCII
18	, ISO-8859-1
11	; " ASCII
10	; " UTF-8
8	| ASCII
8	, GB2312
8	; " ISO-8859-1
7	: ASCII
6	, WINDOWS-1252

呈现每种消息模式的文件数量。这里只显示出现在 5 个或更多文件中的消息模式。大多数 CSV 文件（约 89%）都是以逗号分隔的文本文件，这一点应该不太令人意外。以 ASCII 逗号分隔的 ASCII 文本文件约占样本总数的 70%。这些 ASCII 文件中约有一半使用 ASCII " 作为引号字符。值得注意的是，Microsoft Excel 中常使用 ASCII 制表符作为字段分隔符。

表 I 所呈现的视角很有用，但不擅长发现异常方言。如 [1], 所述，文件-消息数据的另一种概括方式是一个偏序集，其中每个节点对应一种消息模式，而排序关系连接的是通过添加额外消息得到的消息模式对。图 2 展示了我们数据对应的这一偏序的哈斯图。在图中，每个节点的大小由呈现其对应消息模式的文件数量决定。由于各消息模式中的文件数量差异极大，节点大小按对数进行缩放。

图中显然由许多彼此不连通的组件组成，每个组件都对应至少一种方言。其中许多组件对应不同的文本编码。正如预期，ASCII 编码文件构成了最大的组件。

第 II 节中描述的统计模型表明，消息模式之间的关系可能解释它们的普遍程度。例如，考虑消息模式 ASCII 制表符。鉴于有 51 个文件出现了这种模式，因此还有一些文件同时呈现这两条消息以及引号或转义字符，也就不足为奇了。此外，统计模型表明，如果这些其他文件并不常见，那么我们应将它们视为同一方言的一部分。

批准公开发布；不限制分发。根据 ES-FL-012723-0011，不受出口管制。

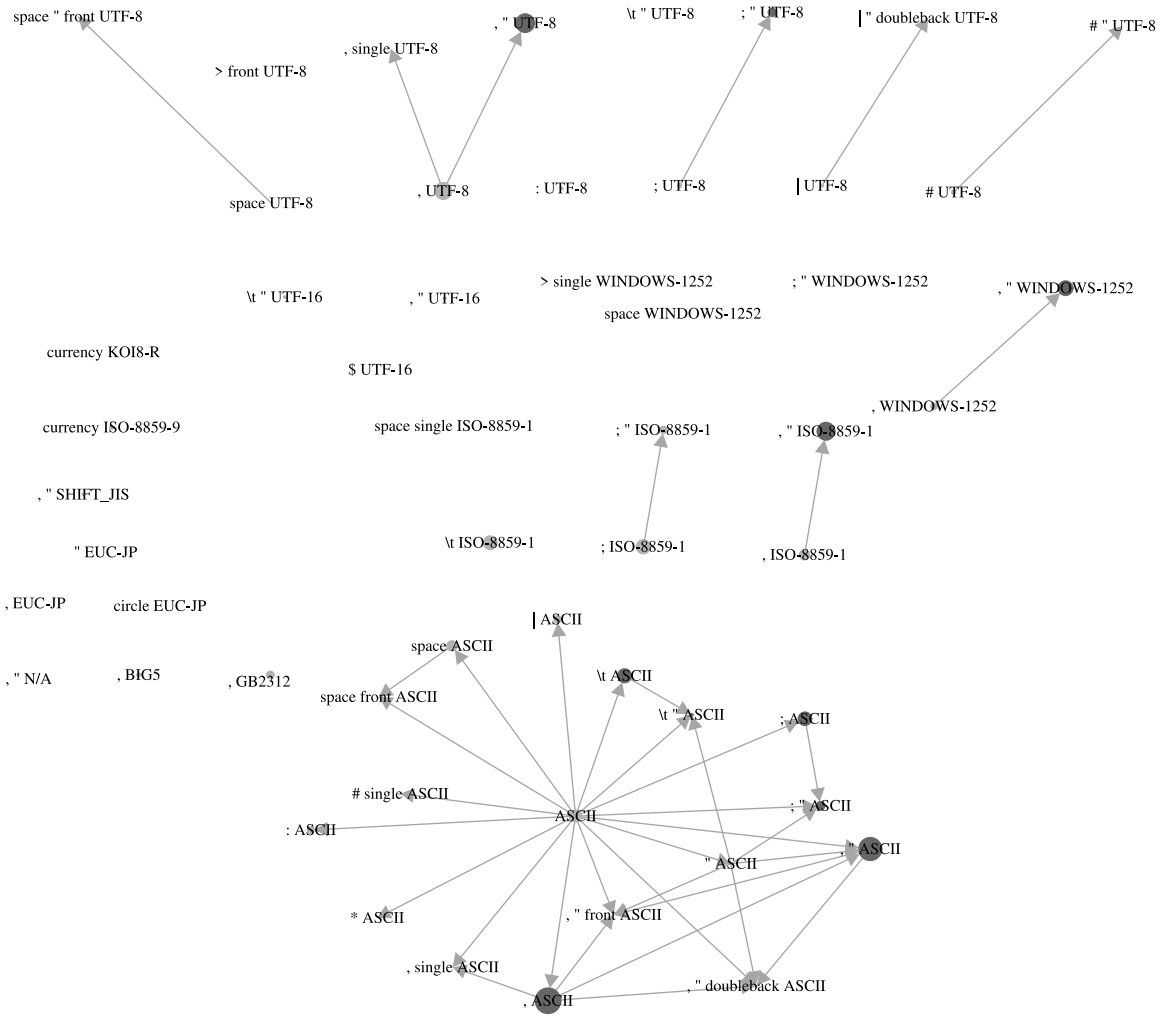


图2。消息模式的偏序。图中仅显示数据中存在的消息模式。每个节点的大小与呈现相应消息模式的文件数量的对数成正比。

方言的形成方式存在一定的歧义，不过第 VI-B 节中的定理 2 证明了存在一组唯一的最大方言。以此为前提，我们得到的方言如表 II 所示。

“根节点文件数”列报告了呈现必需消息且不呈现其他消息的文件数量。根据引理1，与任何其他消息模式相关联的方言中的文件数量都不会超过此值。按文件数量计算，表II中的前几项是采用各种文本编码的逗号分隔文本文件。至少包含5个文件的方言数量为14，小于至少包含5个文件的20种消息模式，因此数据已发生了一些合并。例如，消息模式 , " ASCII 已被并入 , ASCII。由于统计模型预计会有少量呈现消息模式 , ASCII 的文件作为 ASCII 方言的一部分出现，因此每种方言中的文件数量都有所减少。简而言之，该模型预计可能有少量呈现消息模式 , ASCII 的文件实际上并不是 CSV 文件——

表II 按发现顺序排列的最大CSV候选方言

文件数量 位于根目录	必需消息
1388	, ASCII
119	, UTF-8
77	, " UTF-8
18	, ISO-8859-1
7	; " UTF-8
22	ASCII 制表符
138	, " ISO-8859-1
48	ISO-8859-1 制表符
58	, " WINDOWS-1252
29	ASCII
55	; ISO-8859-1
16	; ASCII
8	, GB2312
6	, WINDOWS-1252

表格数据——而是带有逗号的非结构化 ASCII 文本文件。

Approved for public release; distribution unlimited. Not export controlled per ES-FL-012723-0011.

TABLE III
MESSAGES ASSOCIATED TO EACH PDF PARSER

Parser	Message count
caradoc	253
hammer	65
mutool	796
origami	39
pdfium	21
pdfminer	62
pdftk	29
pdftools	10
poppler	995
qpdf	136
xpdf	598
总计	3004

B. PDF

便携式文档文件（PDF）格式由 ISO 32000-2 标准定义。在本次练习中，我们使用了测试与评估团队为 DARPA SafeDocs 评估练习 4 精选的 10000 个文件样本。

每个文件都经过了 11 个不同的解析器处理，并使用了各种选项运行。共收集到 3004 条布尔消息，如表 III 所示。每个解析器的一条消息是一个退出代码，对应于错误的存在。其余消息对应于针对标准错误和标准输出运行的特定正则表达式，如前文 [1]所述。其中一些消息被发现对于识别方言起着重要作用，并列于表 IV 中。

处理完成后，我们发现了 1658 种不同的消息模式，其中文件数量至少为 100 的消息模式列于表 V 中。需要注意的是，有效文件通常仍会产生大量警告和其他输出，因此它们并不对应于少量的消息模式。

我们提出的方言分解算法将消息模式压缩为 4 种根目录文件数量超过 100 的方言，如表 VI 所示。这四种方言具有相当明确的解释，如最后一列所示。后两种方言似乎对应于不同类型的语法错误。

作为比较，如果我们考虑文件数量至少为25的情况，则发现有10种方言的根目录文件数量大于25，而消息模式有43种。在这两种情况下，格式分析师只需考虑大约为全部消息模式四分之一数量的方言。

C. NITF

国家影像传输格式（NITF）被许多美国政府机构用于共享地理编码影像。这些文件包含复杂的头部和数据载荷。实际应用中存在若干种NITF变体，这导致格式出现了一定程度的分化。

作为 DARPA SafeDocs 黑客马拉松练习的一部分，测试与评估团队提供了一组 2626 个 NITF 文件。针对每个文件，运行了 6 个解析器。收集了标准错误、标准输出以及解析器的返回代码。采用 [1] 中针对 PDF 描述的相同流程，并根据 NITF 进行了适当调整，对这些输出运行了一组正则表达式，生成了 136 条可能的消息。解析器和消息的细分情况如图 VII 所示。

最常见的消息如表 VIII 所示。有几种正则表达式的匹配率超过了50%。由于这违背了第II-A节中关于消息概率的假设，因此使用不匹配的情况反而更有信息量。

在我们的数据中，有103条消息仍然存在。有20种不同的消息模式，其文件数量至少为25，如表 IX 所示。

数据按方言进行的分解总结于表 X 中。“根节点文件数”列报告了包含必需消息且不包含其他消息的文件数量。根据引理1，与任何其他消息模式相关的方言中文件数量都不会超过此值。

如表 X 所示，有 12 种方言的必需消息对应的文件数量至少为 25。这比表 IX 所列的 20 种消息模式有所改进。回到表 VIII，除所示的两种最常见方言外，其他方言都包含某种解析器错误 消息。

虽然并非每条可能的消息都对应长度字段违规，但表 X 所示的大多数方言都对应于 NITF 头部中的损坏长度字段。由于 NITF规范允许对数据载荷进行随机访问，长度字段损坏解释了消息 37 和 94 的出现，这些消息表示读取超出了文件末尾。最终区分这些方言的是哪些长度字段遭到了损坏。由于解析器的运行方式存在差异，不同解析器会在解析过程中的不同位置收集不同字段。

进一步检查表X中倒数第二个方言里的文件（即包含消息 2 的那个方言），发现这些文件属于 afrl 解析器不支持的一个版本。

IV. 详细方法

我们的目标是根据每种消息模式的概率 $P(k_1, k_2, \ldots\ldots, k_n)$ 找出候选方言。我们在命题 3 中证明，这些数据可以分解为推论1所设想形式的互不相交方言的混合。此外，还存在一种用于寻找此类分解的贪心算法。

尽管我们的概率模型定义在按子集关系偏序排列的消息幂集上（$2^K, \subseteq$），但我们将改为针对任意偏序集P建立结果。这样可以大幅减少处理数据的任何算法的内存使用量和运行时间，因为我们只需考虑数据中实际存在的消息模式。此外，在这里采用的概括性层次下，我们的数据既可以表示为取值介于 0 和 1 之间的概率，也可以表示为文件计数。

本节建立了一种根据数据构造候选方言分解的算法。如第II-A节所述，数据被表示为一个函数 f ：

Approved for public release; distribution unlimited. Not export controlled per ES-FL-012723-0011.

表 IV 我们的 PDF 数据中与候选方言相关的示例消息

消息	解析器	正则表达式
69	caradoc	PDF 错误：文件中偏移量 \d+ \[0x[A-Fa-f\d]+\] 处存在语法错误！
163	caradoc	PDF 错误：在文件中的偏移量 .* 处出现语法错误！
217	caradoc	PDF 错误：词法分析错误：意外字符：0x[A-Fa-f\d]+ ，位于偏移量...
220	caradoc	PDF 错误：词法分析错误：偏移量 \d+ \[0x[A-Fa-f\d]+\ 处出现意外单词...
250	caradoc	警告：对象 .* 中附加换行符的 Flate\/Zlib 流
96,188,251	caradoc	退出代码表示错误
255	hammer	.*：无法解析
258	hammer	(?:/[a-zA-Z\d \-]+)+/[A-Fa-f\d]+：在位置 \d+ \(0x[A-Fa-f\d 后发生错误...
297	hammer	违规 ... 'endstream' 前没有换行符 ...
308	hammer	VIOLATION ... Missing endobj token \(seve...
313	hammer	VIOLATION ... No linefeed after 'stream' \...
314	hammer	VIOLATION ... Nonconformant WS at end of x...
316	hammer	Exit code meaning error
482	mutool	警告：流开始标记后缺少换行符 \(\d+ \d+ R\)
720	mutool	警告：流开始标记后缺少换行符 \(\d+ \d+ R\)
899	mutool	页面 (?:/[a-zA-Z\d]+)+/[A-Fa-f\]+ \d +
978	mutool	警告：流开始标记后缺少换行符 \(\d+ \d+ R\)
1143	origami	.*对象应以 'endobj' 结束 语句.*
1153	origami	退出代码表示错误
2346	qpdf	警告：.*：应为 endobj
2384	qpdf	警告：.*：流关键字后仅跟回车符
2889	xpdf	语法警告.*：正在替换字体 '.*' 用于 '.*'
3015	xpdf	未嵌入字体_ _

表 V PDF 数据中最常见消息模式的文件数量

文件数量	消息模式
3761	250 251 899 1153
347	251 297 899 1153
319	250 251 899 1153 2888
186	217 251 899 1153
117	92 96 184 188 247 251 899 1153
116	200 250 251 899 1153
114	69 96 163 188 220 251 255 258 297 308 313 ...
107	234 251 899 1153
102	228 251 899 1153
101	7 96 104 188 217 251 899 1153

TABLE VI
LARGEST PDF CANDIDATE DIALECTS

File count at root	Required messages	Interpretation
3684	250 251 899 1153	压缩流错误
270	251 297 899 1153	缺失/位置错误
111	69 96 163 188 220 251 255 258 297 308 313 ...	endstream分隔符语法错误
109	217 251 899 1153	语法错误

表VII 用于处理NITF数据的解析器

解析器	消息数量
afrl	42
codice	28
gdal	36
hammer	15
nitf_	6
kaitai	9
nitro	136
总计	

$P \to [0,\infty)$，其定义域为偏序集（P, $\leq$），值域为非负实数。关键洞见是定义 2，它将推论1推广到处理所有可能的候选方言分解。

起点是将独立混合（推论1）放宽为单调函数的分解。

定义 2。 假设 f：$P \to [0,\infty)$ 是一个从有限偏序集（P, $\leq$) 到非负实数的函数。单调分解 将 f 表示为一组函数之和

$$f(x) = \sum_{k=1}^{N} 1_{U_{y_k}}(x) g_k(x) \tag{2}$$

其中 g_k：$P \to [0,\infty)$ 是单调递减的，且 $U_{y_k} = \{x \in P$: $x \geq y_k\}$ 是一个上闭集。
单调分解中允许 y_k 和/或 g_k 重复。

单调分解推广了第 II 节中提出的概率模型。

命题 2。 假设存在一组消息，使得每个方言中的每种消息模式的概率均由公式 （1）给出。假设各方言互不相交，因此推论*1*适用。则公式 （1） 是所有消息模式的联合概率分布的单调分解。

证明。 根据引理1，每种方言都对应于 $1_{U_y} g(x)$ 形式的项，其中 g 是单调递减函数。根据推论 1，所有消息模式的联合概率分布公式恰好具有定义 2 中公式（2）所要求的相同形式。 □

Approved for public release; distribution unlimited. Not export controlled per ES-FL-012723-0011.

TABLE VIII
MOST COMMON MESSAGES IN THE NITF DATA

Message	File count	parser	regex
59	1051	codice	不存在解析错误\n
102	1039	gdal	不存在 gdalinfo 失败 \- 无法打开 '.*'\。
107	1038	hammer _nitf	退出代码中不存在错误
71	1029	gdal	退出代码中不存在错误
1	825	afrl	退出代码中不存在错误
37	812	afrl	Error reading, read returned .*\. \(start = .*, …
94	527	gdal	ERROR \d+: Not enough bytes to read segment info
108	470	hammer_nitf	/[a-zA-Z\d _\\.\-\(\):/, +]+\.[a-zA-Z\d]+: 无法解析
113	420	hammer nitf_	违规 ... 头部中的文件长度无效 \(严重性=\d+\)
21	394	afrl	读取头部时出错.*
12	308	afrl	用户定义数据长度 = \d+
103	241	gdal	GDAL 错误 .*: NITF 头部长度 \(.*\) 似乎.....
119	241	hammer nitf_	违规 .....图形段数量无效 \(严重性=\d+\)
99	227	gdal	警告 \d+: .....似乎是一个 NITF 文件, 但没有图像 ...

表 IX NITF 数据中最常见消息模式的文件数量

文件数量	消息模式
357	1 59 71 102 107
151	1 12 59 71 102 107
100	1 59 71 99 102 107 122
70	14 23 94
70	94
66	21 37 81 113
59	1 59 71 99 102 107
56	103 113
50	15 37 86
48	21 37 94 113
47	33 37 103 113
44	22 37 76 108 119
43	94 111
42	21 37 103 113
30	21 37 94 119
29	17 59 71 102 107
29	21 37 76 113
29	7 59 71 102 107
28	12 34 37 40 94
26	2 12 59 79 82 107

表X 最大的 NITF 候选方言

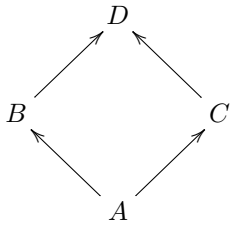
File count at root	Required messages	Interpretation
352	1 59 71 102 107	Valid files
93	1 59 71 99 102 107 122	损坏的数据载荷
70	94	读取访问错误
60	14 23 94	读取访问错误
54	103 113	损坏的标头长度
49	15 37 86	读取访问错误
43	21 37 81 113	损坏的标头长度
41	21 37 94 113	损坏的标头长度
27	22 37 76 108 119	读取访问错误
26	21 37 94 119	损坏的标头
26	2 12 59 79 82 107	有效但不受支持的版本
25	21 37 76 113	损坏的标头长度

我们的实验结果源于以下内容的构造性证明。

命题 3. 从有限偏序集$(P, \leq)$ 到非负实数的每个函数 $f : P \to [0, \infty)$ 都有单调分解。

命题 3 中提出的分解并非唯一，这一点尤其显而易见，因为其原本具有构造性的证明中包含若干任意选择。

例 1. 考虑由下式给出的偏序集



其中函数 f 由下式给出

$$f(A) := 0,\ f(B) := 4,\ f(C) := 4,\ \text{and}\ f(D) := 5.$$

该函数可以写成以下和式

$$f = 1_{U_B} g_1 + 1_{U_C} g_2,$$

其中

$$g_1(A) := 5,\ g_1(B) := 4,\ g_1(C) := 4,\ \text{and}\ g_1(D) := 2,$$

and

$$g_2(A) := 5,\ g_2(B) := 4,\ g_2(C) := 4,\ \text{and}\ g_2(D) := 3.$$

显然， g_1 和 g_2 都是单调递减的。然而，下式同样成立：

$$f = 1_{U_B} g_2 + 1_{U_C} g_1,$$

这与分解的唯一性相矛盾，因为使用的正是同样的两个开集。

我们将在本节稍后构造性地（在作出某些选择后算法地）证明命题3，此前先建立两个引理作为工具。

Approved for public release; distribution unlimited. Not export controlled per ES-FL-012723-0011.

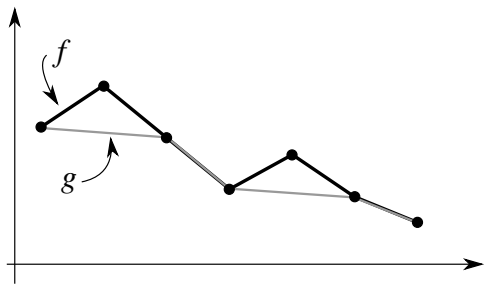


图3. 引理 3 所保证的函数 g 示例

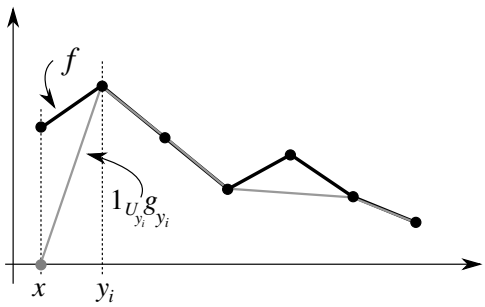


图 4. 命题 3 证明中函数 g_{y_i} 的构造

引理 2. 假设 $f\colon\ P \to [0,\infty)$ 是从偏序集 $(P,\leq)$ 到非负实数的函数。如果 $g_1, g_2\colon\ P \to [0,\infty)$ 是满足以下条件的两个函数

1) 两者 g_i 均单调递减：如果 $x \leq y$ 是 P 的两个元素，那么 $g_i(x) \geq g_i(y)$，并且2) $g_i(x) \leq f(x)$ 对所有 $x \in P$ 以及两者 g_i 均成立。

然后，函数

$$h(x) := \max\{g_1(x), g_2(x)\}$$

满足相同的两个条件。

证明。对于所有 $x \in P$， $h(x) \leq f(x)$ 这一事实直接由以下事实得出：$h(x)$ 等于 $g_1(x)$、$g_2(x)$中的任一个，或两者兼有。现在假设 $x \leq y$ 是P的两个元素。不失一般性，假设 $h(x) = g_1(x)$。这意味着 $g_1(x) \geq g_2(x)$，并且根据假设， $g_2(x) \geq g_2(y)$。根据传递性，这意味着 $h(x) = g_1(x) \geq g_2(y)$。根据假设，我们还有 $h(x) = g_1(x) \geq g_1(y)$。由于$h(y)$ 等于 $g_1(y)$ 和 $g_2(y)$ 中较大的一个，这意味着$h(x)$ 大于 $h(y)$。 □

引理 3。设 $f\colon\ P \to [0,\infty)$ 是从有限偏序集 $(P,\leq)$ 到非负实数的函数。满足 $g\colon\ P \to [0,\infty)$ 且 $g(x) \leq f(x)$ 对所有 $x \in P$成立的单调递减函数集合具有唯一极大元。（见图 *3*。）

证明。我们只需证明所讨论的集合非空，因为引理 2 已经完成了其余部分。这很容易，因为零函数属于该集合。 □

存在一种贪心递归算法，可以构造出由引理 3 保证存在的函数 g 。

1）从 P 的极小元集合 M_0 开始。2）我们可以毫无困难地为每个 $m \in M$ 定义 $g(m) := f(m)$，因为在所有情况下它显然都是极大的。3）为递归步骤做准备，令 $L_0 := M_0$。 4）对于递归步骤，假设 g 已经定义在 P 的某个下闭子集 L_k 上。

a) 考虑 M_k 的极小元集合 $P \backslash L_k$。 b) 对每个 $m \in M_k$，定义 $g(m) := \min\{f(m)\} \cup \{g(x)\colon\ x < m\}$，注意到后一个集合中的每个 x 都是 L_k 的元素，因此 $g(x)$ 有良好定义。以这种方式定义 g 可确保它具有上界为 f，是单调递减的，但除此之外是极大的。

c) 为下一递归步骤做准备，令 $L_{k+1} := L_k \cup M_k$。

这些极大单调递减函数可用于将任意函数分解为若干单调递减函数之和，并适当地限制这些函数的定义域。这使我们能够证明命题 3。

证明。（命题 3）对 f 不满足单调递减的那些位置的数量进行归纳。

- 基例：f 是单调递减的。若 P中只有一个极小元 y，则只需令 $y_1 := y$，并令 $g_1 := f$。然而，若极小元不止一个，情况就会变得令人烦恼地不唯一。对此可以有多种解决方法，例如采用以下平局打破程序。设 $y_1, \ldots, y_m$ 是 P中极小元的任意排序。由于它们都是极小元，因此它们是 P中的两两不可比元素。定义

$$g_i(x) := 1_{A_i}(x) f(x),$$

其中

$$A_i = U_{y_i} \setminus \bigcup_{j=1}^{i-1} U_{y_j}.$$

注意，由于每个 1_{A_i} 都是单调递减的——它在 y_i上取值为 1，但在 P中足够大的元素上最终降至 0——因此得到的 g_i 函数也都是单调递减的。此外，根据构造， P 中的每个元素 x 都恰好属于一个 A_i 集合。因此， f 的分解公式成立。

- 归纳情形：假设 f 在 k个元素 $\{y_1, \ldots, y_k\}$上不是单调递减的，这些元素属于 P。对于这些元素中的每一个 y_i，存在一个 x 属于 P，满足 $x \leq y_i$ 但$f(x) < f(y_i)$。其中至少有一个 y_i在集合 $\{y_1, \ldots, y_k\}$中是极小的，这意味着对于每个 $j \neq i$，要么 $y_i < y_j$，要么 y_i，且 y_j 彼此不可比较。利用引理 3，在 U_{y_i} 上构造一个受 f上界约束的极大单调递减函数g_{y_i}。我们将证明，余函数 $f - 1_{U_{y_i}} g_{y_i}$ 至多在 P中的 $k-1$ 个元素处不满足单调性。

Approved for public release; distribution unlimited. Not export controlled per ES-FL-012723-0011.

我们假设存在一个 x，使得 $f(x) < f(y_i)$。由于假设 g_{y_i} 受 f的上界约束，这意味着 $g_{y_i}(y_i) \leq f(y_i)$。另一方面，鉴于引理 3断言在 U_{y_i}上存在这样的极大 g_{y_i}，我们必须得出结论：$g_{y_i}(y_i) = f(y_i)$，如图 4 所示，因为该值不会进一步影响 g_{y_i}的单调性。因此，余函数 $f - 1_{U_{y_i}} g_{y_i}$ 在 y_i上取值为 0，因而自动满足

$$\begin{aligned} f(x) &= f(x) - 1_{U_{y_i}}(x) g_{y_i}(x) \\ &\geq 0 = f(y_i) - 1_{U_{y_i}}(y_i) g_{y_i}(y_i). \end{aligned}$$

因此，f 中至少有一个单调性违规不存在于残差中。

我们来证明，残差中不会出现新的单调性违规。假设 $x \leq z$ 是 P 中的两个元素，且 $f(x) \geq f(z)$。如果这两个元素都在 U_{y_i}之外，则残差在这两个元素上的值与 f 相同。如果 x 在 U_{y_i} 之外，但 $z \in U_{y_i}$，那么

$$\begin{aligned} f(x) - 1_{U_{y_i}}(x) g_{y_i}(x) &= f(x) \\ &\geq f(z) \\ &\geq f(z) - 1_{U_{y_i}}(z) g_{y_i}(z), \end{aligned}$$

因为根据构造，g_{y_i} 是非负的。最后，假设 x（因此 z也）是U_{y_i} 中的元素，并且$f(x) \geq f(z)$。尽管任意一个以上界 f 为界的单调递减函数 h 可能不会导致 $f(x)–h(x) \geq f(z) - h(z)$，但由于 g_{y_i} 的极大性，这种情况不可能发生。我们通过反证法证明这一点；假设

$$f(x) - g_{y_i}(x) < f(z) - g_{y_i}(z).$$

重新整理这个不等式可得

$$0 \leq f(x) - f(z) < g_{y_i}(x) - g_{y_i}(z),$$

这意味着存在一个单调递减 h，使得$h(x) = g_{y_i}(x)$，且 $g_{y_i}(z) < h(z) \leq f(z)$，这与 g_{y_i}的极大性相矛盾。因此，我们已经证明，残差$f - 1_{U_{y_i}} g_{y_i}$ 的单调性违规数严格少于 f。

□

遗憾的是，命题 3 的证明中提出的分解并不唯一。这意味着，有时可以选择不同的方言，却得到相同的消息概率。在附录第 VI 节中，定理 1 和 2 表明，该算法构造的分解仍能为方言的数量和结构提供一个界限。具体而言，该过程找到的必需消息集合对应于某些真实方言的必需消息集合，尽管可能还有其他尚待发现的方言。

V. 结论

本文提出了一种新颖的基于统计的方法，根据文件被解析时的行为，将文件集划分为格式方言。从理论上讲，我们的方法能够得到与统计模型相符的最粗划分。这意味着格式分析师可以从最少数量的方言开始分析。在实践中，分析师需要考虑的方言数量约为不同消息模式数量的一半。直观地说，这大大减轻了他们研究复杂格式时的认知负荷。

致谢

作者谨向 SafeDocs 测试与评估团队致谢，其中包括美国国家航空航天局（NASA）喷气推进实验室、加州理工学院和 PDF 协会，感谢他们提供测试数据。作者还要感谢 Cory Anderson 最初将文件处理为消息集合。

本材料基于美国国防高级研究计划局（DARPA）SafeDocs 计划在合同 HR001119C0072 项下资助的工作。本文所表达的任何意见、研究结果、结论或建议均属于作者观点，不一定反映 DARPA 的立场。

利益冲突

作者声明不存在利益冲突。

参考文献

[1] M. Robinson、L. W. Li、C. Anderson 和 S. Huntsman，《使用加权 Dowker 复形进行格式方言的统计检测》，载于 *2022* 年 *IEEE* 安全与隐私研讨会（*SPW*），2022 年，第 98–112 页。 [在线]。可访问：http://arxiv.org/pdf/2201.08267[2] D. Scofield、C. Miles 和 S. Kuhn，《用于检测恶意数字文档的快速模型学习》，载于*SSPREW-7*，2017 年 12 月。[3] J. Marin、K. Mengersen 和 C. P. Robert，《分布混合的贝叶斯建模与推断》，载于 贝叶斯基本模型。统计学手册：贝叶斯思维——建模与计算。第 *25* 卷，D. Dey 和 C. Rao 编。爱思唯尔，2011 年）。[4]G. McLachlan 和 D. Peel，《有限混合模型》。威利，2000 年。[5]M. Robinson，《使用来自多个解析器的错误消息查找不合规文档》，载于 *2021* 年 *IEEE* 安全与隐私研讨会（*SPW*），2021 年，第 184–193 页。 [在线]。可访问：https://doi.org/10.1109/ SPW53761.2021.00032[6]K. Ambrose、S. Huntsman、M. Robinson 和 M. Yutin，《拓扑微分测试，arxiv:2003.00976》，2020 年。[在线]。可访问：https://arxiv.org/abs/2003.00976[7] M. Belaoued 和 S. Mazouzi，《一种基于卡方检验和 PE 文件的实时 PE 恶意软件检测系统

特征》，载于*IFIP*计算机科学及其应用国际会议。施普林格，2015 年，第 416–425 页。

[8] B. A. S. Al-rimy、M. A. Maarof 和 S. Z. M. Shaid，“勒索软件威胁成功因素、分类体系与对策：综述及研究方向”，《计算机与安全》，第 74 卷，第 144–166 页，2018 年。

[9] S. D. S.Land J. CD，“基于可视化图像的卷积神经网络 Windows 恶意软件检测器”，《*IEEE*新兴计算主题汇刊》，第 1–1 页，2019 年。 [在线]。网址：https: // doi.org/10.1109/TETC.2019.2910086

[10]M. Alazab，“恶意代码行为分析与分类”，《系统与软件杂志》，第 100 卷，第 91–102 页，2015 年。 [在线]。网址：http://www.sciencedirect.com/science/article/ pii/S0164121214002283[11] J. Demme、M. Maycock、J. Schmitz、A. Tang、A. Waks- man、S. Sethu madhavan 和 S. Stolfo，“利用性能计数器进行在线恶意软件检测的可行性”，*ACM SIGARCH*《计算机体系结构新闻》，第 41 卷，第 3 期，第 559–570 页，2013 年。[12]G. J. J. van den Burg、A. Naz´abal 和 C. Sutton，“通过检测行与类型模式整理杂乱的 CSV 文件”，《数据挖掘与知识发现》，第 33 卷，第 6 期，第 1799–1820 页，2019 年。 [在线]。网址：https://doi.org/10.1007/s10618-019-00646-y[13] A. Bj¨orner，“拓扑方法”，《组合学手册》，第 2 卷，第 1819–1872 页，1995 年。[14]M. Robinson，“关系与道克复形的余层表示”，《应用与计算拓扑杂志》，2021 年。 [在线]。网址：https://doi.org/10.1007/s41468-021-00078-y[15]M. Brun 和 L. M. Salbu，“关系的矩形复形”，《地中海数学杂志》，第 20 卷，第 1 期，第 1–8 页，2023 年。[16]J. Dileo，“ CSV注入，RFC 5322”，2019 年 3 月。[在线]。网址：https://unpack.debug.su/pocorgtfo/ pocorgtfo19.pdf[17] C. R. Kime、R. P. Batni 和 J. D. Russell，“一种寻找无冗余集合覆盖的高效算法”，《*ACM*杂志》，第 21 卷，第 3 期，第 351–355 页，1974 年 7 月。 [在线]。网址：https://doi.org/10.1145/321832.321833[18] S.-i. Minato，“从二元决策图快速生成素数无冗余覆盖”，《*IEICE*电子、通信与计算机科学基础汇刊》，第 76 卷，第 6 期，第 967–973 页，1993 年。

VI. 附录

尽管命题 3 构造了函数的单调分解，但其非唯一性可能妨碍实际应用。第 VI-A 节中的定理 1 断言，命题 3 的证明中构造的分解是极小的，即无法进一步分解。此外，定理 2 断言，如此构造的分解可为真实方言数量找到一个明确的下界。

A. 方言结构的界定

处理可能的方言分解之间歧义的一种有效方法，就是坦然接受这种歧义。有些分解显然更精细，因为它们会将文件集拆分成更小的方言。精细化这一概念可以轻松地描述这种情况。

定义 3. 设函数 $f\colon P \to [0,\infty)$ 来自有限偏序集 $(P,\ \leq)$，且具有多个单调分解。我们将称单调分解

$$f(x) = \sum_{k=1}^{N} 1_{U_{z_k}}(x) h_k(x)$$

精细化该单调分解

$$f(x) = \sum_{j=1}^{M} 1_{U_{y_j}}(x) g_j(x)$$

若对于每个 $k = 1,\ \cdots,\ N$，存在一个 j 使得

1) $U_{z_k} \subseteq U_{y_j}$, and
2) $h_k(x) \leq g_j(x)$ for every $x \in U_{z_k}$.

引理 4. 函数$f\colon\ P \to [0,\infty)$ 从有限偏序集 $(P,\ \leq)$ 映射得到的单调分解集合，在精细化关系下本身构成一个预序集。

证明。 这只是对这些公理的直接验证。假设我们有 f 的三个单调分解，

$$\begin{aligned} f(x) &= \sum_{k=1}^{N} 1_{U_{z_k}}(x) h_k(x) \\ &= \sum_{j=1}^{M} 1_{U_{y_j}}(x) g_j(x) \\ &= \sum_{\ell=1}^{P} 1_{U_{w_\ell}}(x) p_\ell(x). \end{aligned}$$

- 自反性显而易见；集合和函数都满足自反性。

- 对于传递性，假设第一个单调分解细化第二个，且第二个细化第三个。对于每个 $k = 1,\ \cdots,\ N$，都存在一个 j，使得$U_{z_k} \subseteq U_{y_j}$。但还存在一个 ℓ，使得 $U_{y_j} \subseteq U_{w_\ell}$。因此 $U_{z_k} \subseteq U_{w_\ell}$。对于完全相同的这些指标，我们有

$$h_k(x) \leq g_j(x) \leq p_\ell(x)$$

对于所有 $x \in U_{z_k}$ 都成立。因此，第一个单调分解细化第三个。

□

我们寻求的是最小精化的单调分解，即不存在由其精化得到的其他单调分解。由于存在类似例 1 的情况，最小精化单调分解并不唯一。在例 1 中，两个分解都是最小精化的，且彼此都不是对方的精化。然而，我们有如下结论。

已获准向公众发布；分发不受限制。根据 ES-FL-012723-0011，不受出口管制。

定理 1。 证明命题 *3* 时定义的过程，会为有限偏序集上的任意非负函数 f 生成一个最小精化单调分解。

推论 2. 由于根据命题 *2*，每个方言分解都对应一个单调分解，因此定理 *1* 表明，对于方程 *1* 所表达的概率分布，方言数量的下界是最小精化单调分解中项数的最小值。

在尝试证明定理 1 之前，先考虑单调函数的情形，再考虑一般情况，会有所帮助。

引理 5. 设 $f\colon\ P \to [0,\infty)$ 是从有限偏序集 $(P,\leq)$ 到 $P(\leq)$ 的单调函数。若该偏序集有唯一极小元 p，则 f 的每个单调分解都精化

$$f = 1_{U_p} f. \tag{3}$$

此外，该单调分解不精化 f 的任何其他单调分解。

证明。 对于第一个语句，假设

$$f(x) = \sum_{k=1}^{N} 1_{U_{z_k}}(x) h_k(x).$$

显然，由于 p 是 $(P,\leq)$ 的极小元，因此可得 $U_{z_k} \subseteq 1_{U_p} = P$。此外，由于 $h_k(x)$ 非负，且这些项的和为 f，因此可得 $h_k \leq f$。

反过来，由方程 3 定义的单调分解能够精细化任何其他分解的唯一方式是，对于某个 k， $z_k = p$ 。如果情况确实如此，则精细化要求 $f \leq h_k$。然而，由于 h_k 必须加总为 f，仍然有$h_k \leq f$。因此 $h_k = f$。 □

证明。（定理 1）假设 $f\colon\ P \to [0,\infty)$ 是从有限偏序集（P，$\leq$）到某个集合的函数。

引理 5 可与引理 3 一起使用，以排除由极小元支撑的某些分解所进行的精细化。暂且假设 $p \in P$ 是（P，$\leq$）的一个极小元。如果 $g_p\colon U_p \to [0,\infty)$ 是满足 $g \leq f$ 且由引理 3 保证存在的唯一极大单调递减函数，那么

$$f = 1_{U_p} g_p + \sum_{k=1}^{N} 1_{U_{z_k}}(x) h_k(x)$$

无法精细化任何不同时包含项 $1_{U_p} g_p$的单调分解。

处理完这种情况后，现在考虑（P，$\leq$）具有多个极小元的情形。设 p_1、.....、p_m是有限偏序集（P，$\leq$）的全部极小元。

对每个 $i = 1,\ldots,m$，通过以下方式归纳定义单调递减函数 $g_i\colon P \to [0,\infty)$

- 基例： g_1 是 U_{p_1} 上唯一的极大单调递减函数，满足 $g \leq f$ ，这由引理 3 保证，
- 归纳情形： g_i 是 U_{p_1} $g \leq \left(f - \sum_{j=1}^{i} 1_{U_{p_i}} g_i\right)$ 3 上由引理保证的唯一极大单调递减函数。

任何形式为

$$f(x) = \sum_{j=1}^{m} 1_{U_{p_i}} g_i(x) + \sum_{k=1}^{N} 1_{U_{y_k}} h_k(x)$$

都无法精化不包含第一项总和中所有项的任何单调分解。

需要提醒读者的是，在上述构造中对 p_i 的排序通常会产生不同的相应g_i 函数。因此，由此产生的单调分解无法彼此精化。命题 3 的证明中所使用的平局打破程序为排序提供了一种选择。

定理的完整语句可通过仿照命题 3 证明中的归纳情形得到。也就是说，我们用 $\left(f - \sum_{j=1}^{m} 1_{U_{p_i}} g_i\right)$ 代替 f重复上述程序，并将定义域限制为 $P \setminus \{p_1,\ldots,p_m\}$。在每次迭代中，我们都会获得最小精化单调分解中的更多项。

□

B. 精化预序的结构

在上一节中，我们证明了（引理 4）单调分解按精化关系构成预序。现在我们将证明，如果消除某类冗余，则该预序可以加强为偏序（命题4）。

引理6。 假设函数 $f\colon\ P \to [0,\infty)$ 存在两个彼此精化的单调分解，并且这两个分解可以写成

$$f(x) = \sum_{k=1}^{N} 1_{U_{z_k}}(x) h_k(x) = \sum_{j=1}^{M} 1_{U_{y_j}}(x) g_j(x).$$

如果假设集合 $\{z_k\}$ 和 $\{y_j\}$ 是 P中的反链，那么这两个分解至多只相差项的重新排序。

证明。 由于第一个单调分解精化第二个分解，这意味着每个 z_k 都大于 $(P,\leq)$ 中至少一个 $\{y_j\}$ 的元素。事实上，这意味着存在一个（不唯一的）保序函数 $r\colon\ \{z_k\} \to \{y_j\}$ ，使得对于每个 k，都有 $r(z_k) \leq z_k$ 。另一方面，第二个单调分解精化第一个分解这一事实意味着每个 y_j 都大于 $(P,\leq)$ 中至少一个 $\{z_k\}$的元素。同样，存在一个保序函数 s: $\{y_j\} \to \{z_k\}$ ，使得对于每个 j，都有 $s(y_j) \leq y_j$ 。

使用这一记号，

$$z_m = s(r(z_k)) \leq r(z_k) = y_j \leq z_k,$$

Approved for public release; distribution unlimited. Not export controlled per ES-FL-012723-0011.

但成为反链意味着 $z_m \leq z_k$ 蕴含 $m = k$。因此， $y_j = z_k$ 也成立。于是，这两个分解的支撑集的集合仅相差一个指标置换。因此，我们可以在 $U_{z_k} = U_{y_j}$ 上比较相应的单调函数 h_k 和 g_j。由于两个分解彼此细化，我们有 $h_k \leq g_j$ 和 $h_k \geq g_j$。函数上的 $\leq$ 的反对称性完成了论证。

□

引理6的假设只能得出一个充分条件。这个条件限制过于严格，因为我们常常希望表示具有较大方言的子集行为的方言。这样的情况可以用一种单调分解来表示，其中 $z_m < z_k$ 同时存在。

引理7。 假设

$$f(x) = 1_{U_y}(x)g_1(x) + 1_{U_z}(x)g_2(x) = 1_{U_y}(x)h(x)$$

是 $f\colon\ P \to [0,\infty)$ 的两个彼此细化的单调分解，并且 $y \leq z$ 属于 P。那么 g_2 恒等于零。

证明。 由于左侧分解细化右侧分解，因此有 $g_1 \leq h$。另一方面，由于右侧分解细化左侧分解，我们也有 $h \leq g_1$。因此 $g_1 = h$。由于 $y \leq z$ 属于 P，这意味着 $U_z \subseteq U_y$。所以，

$$\begin{aligned} f(z) &= 1_{U_y}(z)h(z) \\ &= h(z) \\ &= 1_{U_y}(z)g_1(z) + 1_{U_z}(z)g_2(z) \\ &= g_1(z) + g_2(z) \\ &= h(z) + g_2(z), \end{aligned}$$

由此得出 $g_2(z) = 0$。由于假设 g_2 是单调的，这意味着 g_2 恒等于零。 □

一个令人头疼的问题是，可能存在冗余，从而使极小性变得复杂。

定义4。 单调分解称为无冗余的

$$f(x) = \sum_{k=1}^{N} 1_{U_{y_k}}(x)g_k(x)$$

如果每个 g_k 函数至少对某个 $x \in P$取非零值。

只需排除所有零项，每个单调分解都能精细化为唯一的无冗余单调分解。

命题4。 来自有限偏序集$(P, \leq)$ 的函数 $f\colon\ P \to [0,\infty)$ 的无冗余单调分解的集合是一个偏序集。

证明。 引理 4 之后剩下的只有反对称性。假设

$$f(x) = \sum_{j=1}^{M} 1_{U_{y_j}}(x)g_j(x) = \sum_{k=1}^{N} 1_{U_{z_k}}(x)h_k(x)$$

存在两个彼此细化的无冗余单调分解。我们要证明，这两个分解实际上相同，只是项的顺序可能不同。

我们来证明集合 $\{y_j\}$ 和 $\{z_k\}$ 是相同的。为说明所需结果由该语句推出，假设对于某个 j， $y_j = z_j$。根据细化假设， $g_j \leq h_j$ 且 $h_j \leq g_j$ ，因此 $g_j = h_j$。

不妨设存在一个 $y \in \{y_j\}$，它不等于任何 z_k。由于第一个分解精细化第二个分解，这意味着仍然必定存在一个 z_k，使得 $z_k \leq y$。

区分两种情况： z_k 要么等于 y_{j_1}中的一个元素，要么不存在这样的元素。第一种情况下，引理7断言 $g_{j_1} = 0$ ，这与第一个分解的不可冗余性矛盾。

在第二种情况下，尽管 z_k 不等于y_m中的任何元素，但精细化仍要求存在一个 y_{j_1}，使得$y_{j_1} \leq z_k \leq y$。假设引理7不能直接应用于这一新情形，我们可以继续重复这一过程，从而得到一个序列 $y \geq y_{j_1} \geq y_{j_2} \geq \cdots$。由于 P是有限集，该序列必定会在某个 $y' \in \{y_j\}$处终止。同样，由于两个分解彼此精细化，我们必须得出结论： $y' \in \{z_k\}$。引理7适用于这一情形，因此与至少一个分解的不可冗余性矛盾。 □

定义5。 （标准定义，参见例如 [17]）设$(X,\ \mathcal{T})$是一个拓扑空间，其中 $\mathcal{T}$是有限的，且$\mathcal{U} \subseteq \mathcal{T}$是 X的一个覆盖。

若存在子集 $\mathcal{V} \subset \mathcal{U}$，使得 $U \notin \mathcal{V}$，但 $U \subseteq \cup\mathcal{V}$，则称开集 $U \in \mathcal{U}$ 在 $\mathcal{U}$ 中冗余。

不含冗余开集的覆盖称为不可冗余覆盖。

用于寻找不可冗余覆盖的高效算法早已为人所知 [17]。不可冗余单调分解对应于具有非零概率的方言，因此有助于识别候选方言分解。了解给定数据集中可能存在的方言数量很有意义，推论 2 将其与最小单调分解联系起来。从实用角度看，不可冗余单调分解尤其有用，因为尽管它们并非唯一，但对于所涉及方言的数量却没有歧义。

定理 2。 方言的数量不小于任意最小不可冗余单调分解所细化的联合概率分布函数的不可冗余覆盖中的支撑集数量。

定理 2 是以下两个引理的直接推论。

引理 8。 给定函数的所有最小无冗余单调分解都具有相同的支撑集，因此具有相同数量的项。

已获准公开发布；可无限制分发。根据 ES-FL-012723-0011，不受出口管制。

这不同于寻找逻辑函数的最小无冗余分解这一相关情形。已知某些逻辑函数具有分解为项数不同的和的最小无冗余分解 [18]。

证明。假设

$$f(x) = \sum_{y \in R \subseteq P} 1_{U_y}(x) g_y(x)$$

是任意函数 f 的最小不可约单调分解。要证明的语句是，方程中的R 集合对于 f 的所有最小不可约单调分解都相同。更明确地说，若 $y \in R$，使得 $1_{U_y}(x)g_y(x)$ 是上述最小不可约单调分解中的一项，那么任何其他不可冗余单调分解也必须包含一项，形式为$1_{U_y}(x)h_y(x)$。

假设 $y \in P$ 满足：对于所有 $x \leq y$，都有 $f(x) = 0$。那么，f 的每个单调分解（无论是否冗余）都必须包含一项，形式为 $1_{U_y}(x)h_y(x)$。

假设 $y \in P$ 满足：存在一个 $x \in P$，使得 $x \leq y$ 且 $f(x) \neq 0$。我们可以将单调分解改写为

$$f(x) = \sum_{v \in R: v < y} 1_{U_v}(x) g_v(x) + \sum_{w \in R: y \leq w} 1_{U_w}(x) g_w(x) + \sum_{z \in R: z \not\leq y, y \not\leq z} 1_{U_z}(x) g_z(x).$$

由于 $g_y(y) \neq 0$ 根据不可冗余性，中间项必为正数。这意味着上述分解导出了不等式

$$f(y) > \sum_{v \in R: v < y} 1_{U_v}(y) g_v(y).$$

我们可以将这个不等式进一步推进。如果存在一个子集 R'，其中的 $v \in R$ 满足 $v < y$ 以及

$$f(y) > \sum_{v \in R: v < y} 1_{U_v}(y) g_v(y) = \sum_{u \in R'} f(u). \tag{4}$$

这个断言可以通过反证法证明；假设存在一个子集 $R' \subseteq \{v \in R : \ v < y\}$，使得

$$f(y) = \sum_{u \in R'} f(u)。$$

如果情况确实如此，利用 g_v 函数在每个 $u \in R'$ 上的和等于 $f(u)$ 这一事实，我们可以令 g_v 函数在 y 处取相同的值，而不违反单调性。因此， g_y 必须为零，因为所有 $g_v(y)$ 的和等于 $f(y)$，这与不可冗余性矛盾。显然，一个更小的 $f(y)$ 也会迫使 $g_y(y) = 0$。无论如何，这同样与不可冗余性矛盾。因此， $f(y)$ 严格大于该值，正如公式（4）所述。

请注意，公式（4）中的最后一个求和项不依赖于 g_v。因此，不可约单调分解中涉及 U_y 的项由 f 的值直接确定。因此，任何其他单调分解也会面临相同的情况，因此也需要包含一个涉及 y 的项。 □

引理 9. 假设$(X, \mathcal{T})$ 是一个拓扑空间，其中 $\mathcal{T}$是有限的，且 $\mathcal{U}$ 是一个覆盖，并细化了一个不可冗余的覆盖 $\mathcal{V}$。那么$\#\mathcal{V} \leq \#\mathcal{U}$。

证明。令 $V \in \mathcal{V}$。这些假设意味着存在一个 $U \in \mathcal{U}$，使得 $U \subseteq V$。下面用反证法证明这一断言。假设没有任何 $U \in \mathcal{U}$ 是 V的子集。由于 $\mathcal{U}$ 是一个覆盖，因此存在一个集合族 $\mathcal{U}' \subseteq \mathcal{U}$，使得 $V \subseteq \cup\mathcal{U}'$。由于$\mathcal{U}$细化了 $\mathcal{V}$，每个 $U' \in \mathcal{U}'$ 都是某个 $V' \in \mathcal{V}$的子集。考虑子集

$$\mathcal{V}' := \{V' \in \mathcal{V} : U' \subseteq V' \text{ for some } U' \in \mathcal{U}'\} \subseteq \mathcal{V}.$$

显然 $V \subseteq \cup\mathcal{V}'$。回想一下，我们假设没有任何 $U \in \mathcal{U}$是 V的子集，因此必须得出 $V \neq \cup\mathcal{V}'$，这与 $\mathcal{V}$的不可冗余性矛盾。

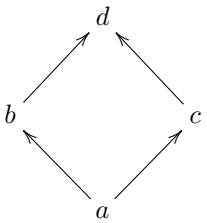
我们通过对 $\#\mathcal{V}$ 进行归纳来完成论证。

- 基例：假设 $\#\mathcal{V} = 1$。由于 $\mathcal{V}$ 和 $\mathcal{U}$都覆盖 X，且 $\mathcal{V}$ 显然非空，因此 $\mathcal{U}$ 也必定非空。
- 归纳情形：假设对于某个整数n，所有满足 $\#\mathcal{V} \leq n$ 的 $\mathcal{V}$，引理均已得到证明。假设 $\mathcal{V}$ 是一个包含$n+1$个元素的不可冗余覆盖，元素为 $V_0, \ldots, V_n$。考虑由 $V_0, \ldots, V_{n-1}$覆盖的 $(X, \mathcal{T})$ 的子空间。根据上面证明的断言，存在一个子集 $\mathcal{U}' \subseteq \mathcal{U}$，它既覆盖$V_0 \cup \cdots \cup V_{n-1}$，又细化覆盖 $\{V_0, \ldots, V_{n-1}\}$。将归纳假设应用于这一情形可得 $\#\mathcal{U}' \geq n$。根据 $\mathcal{V}$ 的不可冗余性，必有 $V \not\subseteq (V_0 \cup \cdots \cup V_{n-1})$。由于 $\mathcal{U}'$的每个元素都是至少一个 $V_0, \ldots, V_{n-1}$ 的子集，因此我们同样有 $V \not\subseteq \cup\mathcal{U}'$。因此，为了覆盖 X， $\mathcal{U}$必须比 $\mathcal{U}'$至少多一个元素。因此，

$$\#\mathcal{U} \geq \#\mathcal{U}' + 1 \geq n + 1 = \#\mathcal{V}.$$ □

证明。（定理 2 的）这是引理 8 和 9 的直接推论。 □

例 2。 假设 $P = \{a, b, c, d\}$ 是由哈斯图定义的偏序



根据定理 2，任意函数 $f : P \to [0, \infty)$ 的所有极小精化的不可冗余单调分解都具有相同的项数。我们可以直接通过分析单调分解来证明这一事实。

如果 $f(a) \neq 0$，那么 f 的每个无冗余单调分解都必须包含一个形如 $1_{U_a} g_a$ 的项，其中

已批准公开发布；分发不受限制。根据 ES-FL-012723-0011，不受出口管制。

$g_a(a) = f(a)$。因此，不失一般性，我们可以假设 $f(a) = 0$。此外，不失一般性，我们可以假设 $f(b) \le f(c)$。

如果 $f(b) = 0$，那么在对 f 进行分解时无需作出任何选择：

- 如果 $f(c) = 0$，则该分解至多包含一个形如 $1_{U_d} f(d)$ 的项。
- 如果 $f(d) > f(c) > 0$，则该分解包含一个形如 $1_{U_d}(f(d) - f(c))$ 的项。
- 否则，该分解仅包含一个项。

反之，如果 $f(b) \neq 0$，那么无论我们使用 b 还是 c 开始分解，都不会改变分解中最终的项数。这是因为，移除由 b 或 c 支撑的项的贡献后，会得到一个可按上述方式分解的新函数。

虽然定理 2 处理的是最小精细化无冗余单调分解的情形，这类分解对于确定方言数量具有重要意义，但也存在最大精细化无冗余单调分解。

命题 5. 函数 $f: P \to \mathbb{Z}^+$从有限偏序集 $(P, \le)$ 映射到非负整数，存在唯一的极大精细化无冗余单调分解，即

$$f(x) = \sum_{y \in P} \sum_{i=1}^{f(y)} 1_{U_y}(x) 1_{\{y\}}(x). \tag{5}$$

极大精细化的单调分解并没有太大意义，因为这意味着每个方言恰好包含一个文件。

证明。 由于假设 P 是有限的，且单调分解中的每个单调函数都取非负整数值，因此单调分解的集合是有限的。因此，引理 4 蕴含在精细化关系下存在极大和极小的单调分解。

假设我们有 f 的任意一个单调分解，

$$f(x) = \sum_{k=1}^{N} 1_{U_{y_k}}(x) g_k(x). \tag{6}$$

我们必须证明方程（5）是该分解的精细化。

仔细检查方程（5）中的求和可知，仅当 $f(y) > 0$ 时，其中才包含涉及 U_y 的项。虽然外层求和似乎意味着对于所有 $y \in P$，都会出现涉及 U_y 的项，但内层求和会阻止任何满足 $f(y) = 0$ 的项被纳入。因此，方程（5）定义了一个无冗余的单调分解。

鉴于这一观察，考虑一个 $y \in P$，使得 $f(y) > 0$。必然存在方程（6）中的一项，使得$y \in U_{y_k}$ 且 $g_k(y) > 0$。因此， $U_y \subseteq U_{y_k}$。此外，由于方程（5）中的每个单调函数恰好在 P 的一个元素上取值 1，我们有 $1_{\{y\}} \le g_k(y)$。 □

注释1。 如果我们反过来允许命题 5 中存在冗余，那么可以任意添加带有零函数的项。虽然这些单调分解彼此都构成细化，但这会使此类分解不再具有唯一性。

注释2。 如果我们改为考虑命题5中的 $f: P \to [0, \infty)$，那么方程（5）中的内层求和将变为无穷大。在这种情况下，不存在最大精细化单调分解。

已批准公开发布；发行不受限制。根据 ES-FL-012723-0011，不受出口管制。