arXiv:2301.12084v1 [cs.SD] 28 Jan 2023

为单声部乐器组自动编配多声部音乐

MATTHEW MCCLOSKEY, GABRIELLE CURCIO, AMULYA BADINENI, KEVIN MCGRATH, 和 DIM- ITRIS PAPAMICHAIL,新泽西学院，美国

为不同于原作的乐器组合编配音乐，传统上是一项繁琐且耗时的工作，通常由精通特定乐器的专家完成，并且需要大量试验。本文研究如何为原本创作给单音乐器或人声演奏的乐曲自动编配。我们设计并实现了一种算法，在可行的情况下，总能通过将乐曲移调至不同音阶、重新分配乐器/声部所演奏的声部，以及将各个声部移高或移低一个或多个八度，生成音乐编曲。我们还发布了用 Python 编写的开源软件，可处理 MusicXML 文件，让音乐家尝试不同的音乐编曲。我们希望该软件能够成为未来扩展的平台，包括音乐缩编以及加入复调乐器。

ACM 分类概念：• **应用计算** → **声音与音乐计算**。

附加关键词和短语：音乐编曲，音乐算法

ACM 引用格式：
Matthew McCloskey、GabrielleCurcio、Amulya Badineni、Kevin McGrath 和 Dimitris Papamichail。2023。为单声部乐器组自动编配多声部音乐。1，1（2023年1月），7页。https://doi.org/XXXXXXX.XXXXXXX

1 引言

音乐编曲是指将一首乐曲改编为适合不同乐器或合奏团演奏的形式。这使音乐能够在各种场合演奏，丰富音乐家的曲目库，也有助于为原本专为特定乐器或合奏团创作的乐曲注入新的活力 [16]。此外，编配还可以展现不同乐器的独特优势，甚至为乐曲带来全新的诠释。编配一首乐曲的过程本身也可以是一种创造性活动，让编曲家有机会为熟悉的作品融入自己的风格，从而大大提升听众的聆听体验 [1, 5, 12]。

Moses 和 Demaine 的研究考察了将为一组乐器创作的音乐编配给目标单一乐器所涉及的计算复杂度，其中通常会采用合理的缩减约束 [4]。处理钢琴和吉他等复调乐器的复杂性在于，除了缩减乐谱，还需要考虑可能的指法，并为确保可演奏性乃至可行性而删去某些音符。自动化音乐编曲的大多数研究都聚焦于钢琴，主要涉及管弦乐作品 [2, 8, 10, 11, 13, 14]。其中许多工作都通过缩减乐谱来确保可行性。该领域的其他研究则考察了吉他编配 [6, 7, 15], 、管乐合奏团编配 [9], 以及其他管弦乐器的编配 [3]。

作者地址： Matthew McCloskey；Gabrielle Curcio；Amulya Badineni；Kevin McGrath；Dimitris Papamichail，papamicd@tcnj.edu，新泽西学院 f
新泽西州，2000 彭宁顿路，尤因，新泽西州，美国，08618。

允许为个人或课堂使用而免费制作本作品的全部或部分数字或纸质副本，前提是不得为营利或商业利益制作或分发副本，并且副本首页须附有本声明及完整引文。本作品中由 ACM 以外的其他方拥有的组件，其版权必须得到尊重。在注明出处的情况下允许摘录。若要进行其他形式的复制、重新出版、发布到服务器或重新分发至列表，须事先获得具体许可和/或支付费用。请通过 permissions@acm.org 申请许可。
© 2023 计算机协会。稿件已提交至 ACM

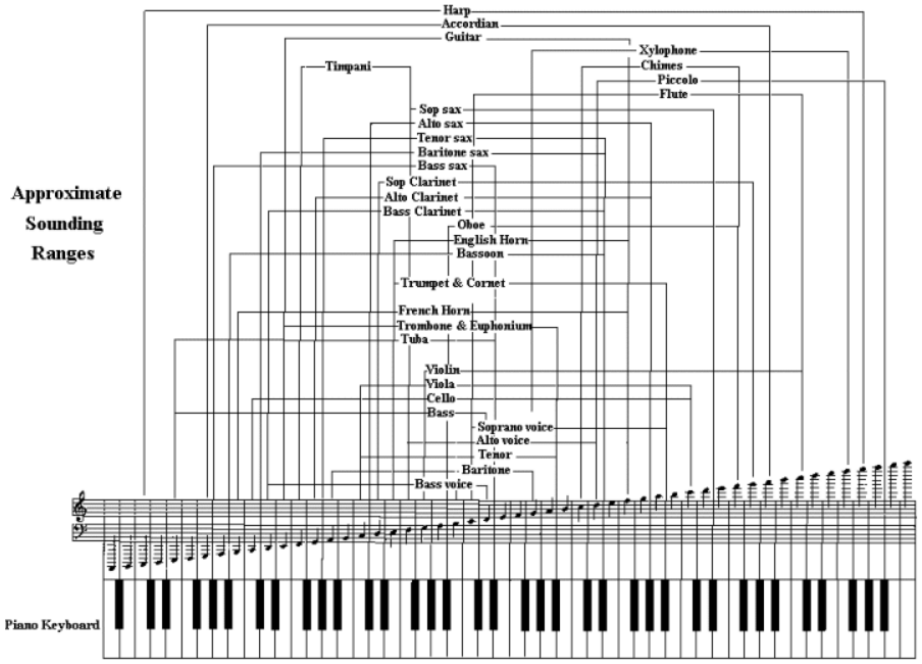


图1. 乐器和人声的大致音域。经 Brian Blood博士（dol-metch.com）许可转载。

尽管这显然大有裨益，但据我们所知，目前尚无已发表的算法或广泛可用的软件，能够在一般情况下将一首乐曲自动编配给一组不同于其原始创作对象的乐器。为满足这一需求，我们设计并实现了一种算法，用于编配为单音乐器创作的音乐；只要无需缩减乐谱便可完成编配，该算法就能保证成功。我们的递归回溯算法会穷尽考察将各声部分配给可用乐器的所有可行方案，以及乐曲的所有可能移调（包括对各个声部独立进行八度移调），从而确定一种能最大限度保留乐曲音乐性的成功编配方案。

2 方法

2.1 定义

在本研究中，乐曲采用半音阶记谱，音符之间以半音为间隔。我们假设所有音符都处于总跨度为88个半音的音域内，即传统钢琴从A0到C8的音域。我们将为该音域内的每个音符分配一个整数，使所有音符都能用1到88之间的整数表示。在本文讨论中，单音乐器是指一次只能演奏一个音高的乐器，例如长笛、双簧管或人声。复调乐器可以同时演奏多个音符，例如钢琴、吉他或竖琴。复调乐器始终能够在其音域内演奏单声部。

在本研究中，输入乐曲由 n 个声部组成，每个声部分配给一种单声部乐器或人声。乐曲的乐谱表示中，每个声部各对应一行谱表。提交至 ACM 的稿件

```
单簧管 = 1
次中音萨克斯管 = 2
中音萨克斯管 = 2
```

(a) 一个编配文件示例

```
[中音萨克斯管]
名称 = "中音萨克斯管"
最低音域 ="Db3"
最高音域 ="Bb5"
调性 = "Eb"
```

(b) 乐器元数据文件中的一条记录

图 2. 输入乐器集合和乐器信息文件示例

算法保留乐曲的节奏、音符和休止符的时值以及小节线。谱号、调号和变音记号会根据移调后的音乐的音阶以及声部所分配的乐器/声部进行调整。我们的算法不会控制乐曲中各声部可能预期的乐器音色；同样，也不一定会保持乐曲的织体厚度。

我们假设，输入乐曲最初是为 n 件乐器 $I_1, I_2, \cdots I_n$ 创作的，每件乐器负责演奏乐曲的一个声部 P_i，且 $1 \le i \le n$。我们希望将这首乐曲编配给 n 件输出乐器 $O_1, O_2, \cdots O_n$。每个声部 i 的音域是一个整数区间 $R_i = [\![a_i, b_i]\!]$，其中 a_i 是乐器 I_i 在声部 $P_i, 1 \le i \le n$ 中演奏的最低频率音符所对应的整数值，b_i 则是最高频率音符所对应的整数值。同样，每件输出乐器 O_i 的演奏音域记为 $OR_i, 1 \le i \le n$，表示该乐器能够演奏的音符所对应的整数值区间。一组乐器和人声的大致音域见图1。

2.2 单音乐器集合编配算法

我们的单声部音乐编配（MMS）算法会对声部可能的排列进行近乎穷尽的搜索。乐曲会移调到全部十二个调，并在每个调上运行算法；但如果此前已找到一个解决方案，使每个声部在所有调中的升号/降号总数更少，则不再继续搜索。这是为了避免在没有必要时采用调号复杂的“理想”移调。除此之外，搜索是完全穷尽的。对于每个声部，算法会找出源乐曲中该声部所有可由至少一件可用乐器演奏的移调方式。随后会检查这些可能移调方式的所有排列。如果所有声部都至少能由一件乐器演奏，算法便会检查是否存在一组有效的声部分配。该检查通过递归函数完成，并采用记忆化来提升性能。如果某个移调后的调产生了有效排列，则选择与原始乐曲总偏差最小的移调方式。检查完全部十二个调后，算法会使用选定的移调方式尝试所有排列；如果没有选定移调方式，则算法失败。算法会检查所有排列，并从在给定移调方式下有效的排列中，选出最佳编配：其依据是每个声部的平均音高与对应乐器音域的中位音高有多接近。

MMA 算法的实现由以下伪代码描述的四个主要函数组成 .

2.3 实现

MMA 算法使用 Python，借助 Music21库和 MuseScore软件实现。我们的程序需要两个输入文件，并生成一个包含音乐编编配的输出文件。所需的输入文件包括原始乐曲的 MusicXML格式文件，以及一个列出待编配乐器集合的 TOML文件；其中，为某件乐器指定值 k 表示应为该乐器编配 k 个声部。一个 TOML文件示例中，输入乐器集合包括一支单簧管、两支次中音萨克斯管和两支中音萨克斯管，示例如下：

算法1 查找移调选项

过程 查找移调选项(*originalStream*, *arrangementParts*, *semitones*)
  stream ← *originalStream* 按给定的 *semitones*移调
  parts ← 新列表
  对于 *part* 属于 *stream*，**执行**
    choices ← 新列表
    对每个 *transposition* **执行**
      set ← 能够在此移调下演奏的 *arrangementParts* 子集
      将（*semitones* + *transposition*， *set*）添加到 *choices*
    结束循环
    如果 *choices* 为空**则**
      返回 null
    结束条件判断
    将 *choices* 添加到 *parts*
  结束循环
  返回 *parts*
结束过程

算法 2 RunTransposed

过程 RunTransposed(*stream*, *parts*, *semitones*)
  selections ← 新列表
  对于 *option*，遍历 FindTransposedOptions(*stream*, *parts*, *semitones*) 返回的所有可能移调 **执行**
    partsCovered ← 新列表
    selection ← 新列表
    对于 *transposition* 在 *option* **中执行**
      将所覆盖声部的集合添加到 *partsCovered*
      将移调偏差添加到 *selection*
    结束循环
    allPartsCovered ← 所有集合在 *partsCovered*中的并集
    如果 *allPartsCovered*包含所有声部，且 *ValidateArrangement*(*parts*, *partsCovered*, *allPartsCovered*) **则**
      将 *selection* 加到 *selections*
    结束条件判断
  结束循环
  返回 *selections*
结束过程

如图 2a 所示。每件乐器的元数据包括其记谱调号和合理的音域，这些信息在单独的 TOML文件中定义；该文件由程序单独加载，并填入常见乐器的信息。图中展示了乐器元数据文件中中音萨克斯管条目的示例2b。

程序执行期间会检查输入乐器的数量是否与乐曲中的声部数量相符，然后尝试按照前文所述，为给定的乐器进行编配。如果找到编配方案，则会根据第2.2 节所述的标准，输出最佳编配方案的 MusicXML 文件。如果找不到可行的编配方案，或者乐器数量不匹配，则会显示错误消息，并且不会生成输出文件。

算法3查找最佳选择

过程 查找最佳选择(*stream*, *parts*)
bestChoice ←空
bestSharps ← ∞
对于 *semitones* 从 −6 到 5 **执行**
sharps ← 每个声部调号中将出现的升号/降号总数
如果 *sharps* ≤ *bestSharps* **则**
thisBestChoice ←从 RunTransposed(*stream*, *parts*, *semitones*) 中选取偏差最小的元素
如果 *thisBestChoice* ≠ 为空且 *sharps* < *bestSharps* 或偏差为 *thisBestChoice* < 的偏差 *bestChoice* **则**
bestChoice ← *thisBestChoice*
bestSharps ← *thisBestSharps*
结束如果
结束如果
结束循环
返回 *bestChoice*
结束过程

算法 4MMA 算法

过程 MMA(*stream*, *parts*)
bestChoice ← 查找最佳选项(*stream*, *parts*)
如果 *bestChoice* = null，**则**
返回 null
结束条件判断
将每个声部按所得移调进行移调
bestFit ← ∞
对于 *newParts* 的每种排列，**执行**
如果 所有声部在给定排列中均有效，**则**
fit ← 各声部的平均音高与中位音高之间的绝对差之和
if fit < *bestFit* **then**
bestFit ← fit
bestPermutation ← 此排列
结束 if
结束 if
结束 for
返回 *bestPermutation*
结束过程

3 结果

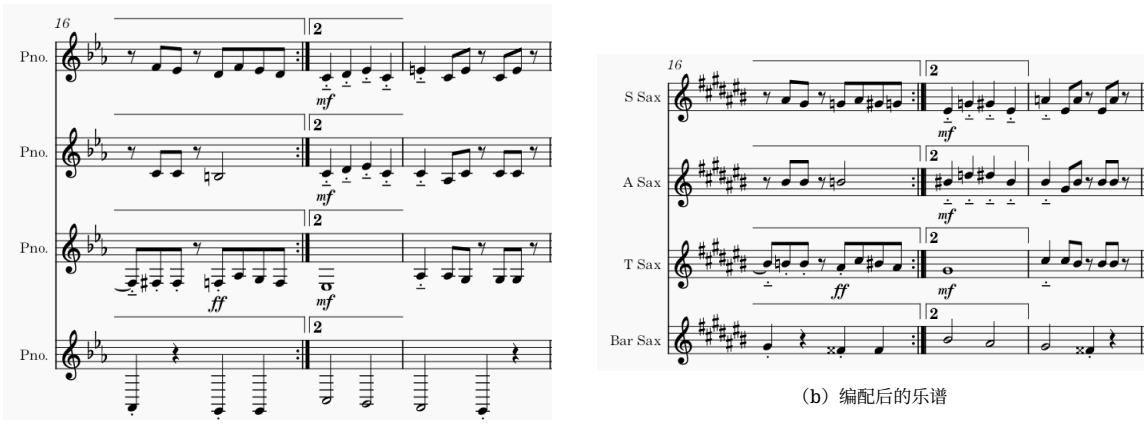
我们在多首为单音乐器创作的乐曲上测试了该软件。图3 展示了《Puttin' on the Ritz》第 16小节开始的三个小节。该曲由 Irving Berlin 创作。（a）部分展示了由四个单声部组成的输入乐谱。（b）部分展示了编配后的萨克斯四重奏乐曲，由高音、中音、次中音和上低音萨克斯管组成。类似地，图 4 展示了《*Carol of the Bells*》的三个小节。该曲由 Pentatonix 人声组合编配并演唱，选取自乐曲第 18 小节开始的部分。

我们软件三个测试用例的完整输入/输出文件（包括上文提到的《*Puttin' on the Ritz*》 和《铃儿响叮当》）可在以下网址查看：https://owd.tcnj.edu/~papamicd/music/mma/examples/

该项目的代码仓库位于：https://github.com/spazzylemons/music-arrangemen t/

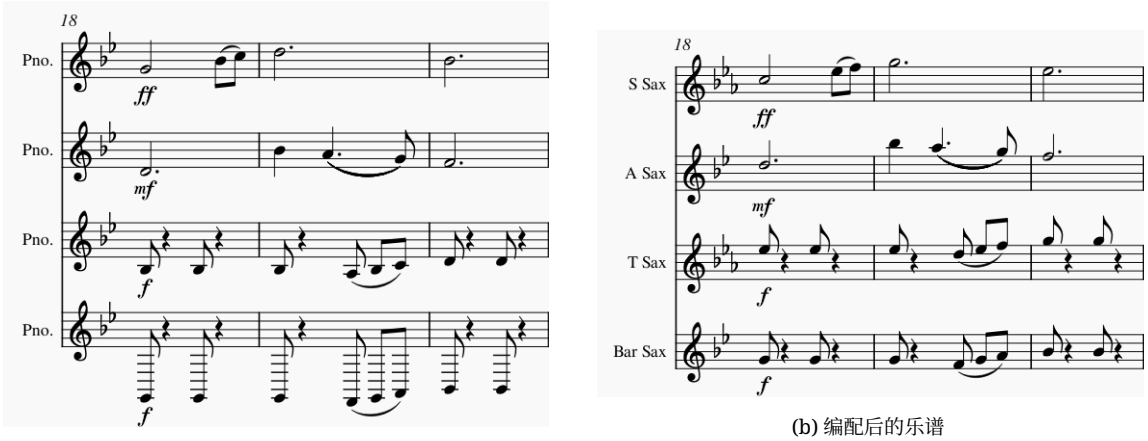
4 结论与未来工作

我们的单声部音乐编曲算法及其软件实现，打造了一个平台，可通过最少的用户输入自动完成音乐编曲。尽管目前功能还比较基础，但它很容易沿多个不同方向扩展。若要为乐曲编配少于声部数量的乐器集合，可应用乐谱简化技术，删除某些声部，或至少减少整首乐曲中同时发出的音符数量，同时保持对原作的忠实。若要在编配方案中加入复调乐器，还需要进一步研究如何分析和分解



（a）原始乐谱 （b）编配后的乐谱

图 3. 从钢琴改编为萨克斯四重奏的《Puttin’ on the Ritz》编配方案中的三个小节



(a) 原始乐谱 (b) 编配后的乐谱

图 4. 《Carol of the Bells》编配为萨克斯四重奏的三个小节

将多声部声部转换为单声部声部，反之亦然，同时遵守与指法及其他乐器和演奏者限制相关的约束。

致谢

作者感谢使用新泽西学院的 ELSA 高性能计算集群开展本文所报告的研究。该集群部分由美国国家科学基金会通过资助编号 OAC-1826915 和 OAC-1828163 提供经费支持。

参考文献

[1] D. Baker. 1988.《David Baker 的编曲与作曲：小型合奏、爵士、节奏布鲁斯与爵士摇滚》。Alfred 出版公司。https: // books.google.com/books?id=Le0EnwEACAAJ[2] Shih Chuan Chiu, Man Kwan Shan, and Jiun Long Huang. 2009. 钢琴缩编自动编曲系统。载于 ISM 2009 - 第 11 届 IEEE 多媒体国际研讨会。https://doi.org/10.1109/ISM.2009.105[3] Léopold Crestel and Philippe Esling. 2019. 实时管弦乐钢琴：实时管弦乐生成系统。载于第 14 届声音与音乐计算会议 2017 论文集，SMC 2017。arXiv:1609.01203[4] Erik D. Demaine and William S. Moses. 2017. 364音乐编配的计算复杂性。载于《各种娱乐主题的数学：游戏、图论、计数与复杂性研究》，第 2 卷。普林斯顿大学出版社。https://doi.org/10.23943/princeton/9780691171920.003. 0019[5] J.B. Elder. 2018.《编曲与配器艺术》。独立出版。https://books.google.com/books?id=mOQKugEACAAJ[6] Gen Hori, Hirokazu Kameoka, and Shigeki Sagayama. 2013. 将输入输出隐马尔可夫模型用于吉他自动编配。《信息处理学报》(2013)。https://doi.org/10.2197/ipsjjip.21.264[7] Gen Hori, Yuma Yoshinaga, Satoru Fukayama, Hirokazu Kameoka, and Shigeki Sagayama. 2012. 使用隐马尔可夫模型为吉他自动编配。第 9 届声音与音乐计算会议论文集 (SMC2012) (7 2012), 450–455.[8] Jiun-Long Huang, Shih-Chuan Chiu, and Man-Kwan Shan. 2012. 构建基于乐谱缩编的自动音乐编配框架。《ACM 多媒体计算、通信与应用汇刊》8, 1, Article 8 (feb 2012), 23 页。 https://doi.org/10.1145/2071396.2071404[9] Hiroshi Maekawa, Norio Emura, Masanobu Miura, and Masuzo Yanagida. 2006. 基于标准管乐团乐谱为小型管乐团进行机器编配。载于音乐感知与认知国际会议，ICMPC 2006。268–273.[10] Eita Nakamura and Kazuyoshi Yoshii. 2018. 控制演奏难度的统计式钢琴缩编。《亚太信号与信息处理协会信号与信息处理汇刊》(2018)。 https://doi.org/10.1017/ATSIP.2018.18 arXiv:1808.05006[11] Sho Onuma and Masatoshi Hamanaka. 2010. 基于作曲家编配过程的钢琴编配系统。载于国际计算机音乐会议，ICMC 2010。[12] T.H. Stefan Kostka, T.H. Dorothy Payne, and B. Almén. 2017.《调性和声》。麦格劳-希尔教育。https://books.google.com/books?id=Cs2UAQAACAAJ[13] Hirofumi Takamori, Haruki Sato, Takayuki Nakatsuka, and Shigeo Morishima. 2019. 使用重要音乐元素为钢琴自动编配乐谱。载于第 14 届声音与音乐计算会议 2017 论文集，SMC 2017。[14] Moyu Terao, Yuki Hiramatsu, Ryoto Ishizuka, Yiming Wu, and Kazuyoshi Yoshii. 2022. 基于音符级与统计级标准的难度感知神经网络乐队谱转钢琴谱编配。载于 ICASSP 2022 - 2022 IEEE 声学、语音与信号处理国际会议 (ICASSP)。196–200。 https://doi.org/10.1109/ICASSP43922.2022.9747615[15] D.R. Tuohy and W.D. Potter. 2006. 基于遗传算法的吉他音乐编配。载于 2006 IEEE 进化计算国际会议。1065–1070。https://doi.org/10.1109/CEC.2006.1688427[16] G.C. White. 1992.《器乐编配》。麦格劳-希尔。 https://books.google.com/books?id=4c2KMwEACAAJ