基于遗传算法的计算机智能作曲模型①

陈洁群

(广东省培英职业技术学校, 广州 510663)

摘 要: 针对传统计算机智能作曲客观评价不准、实现难度过大问题, 本文对智能作曲进行探讨并给出一种遗传算法的智能作曲模型, 并提出一种简化的模型. 对该模型分别从算法设计和算法实现两方面进行了较深入的探讨, 同时给出了程序实现. 实验结果表明, 模型对局部进行改善而不影响到其他部件, 具有较高的可扩展性, 算法的时间和空间复杂度较好, 证明本文的算法是可行和有效的.

关键词: 人工智能; 遗传算法; 计算机作曲; 算法作曲; 作曲模型

Intelligent Music Composition Based on Genetic Algorithms

CHEN Jie-Qun

(Guangdong Province Pui Ying Occupation Technical School, Guangzhou 510663, China)

Abstract: According to the problem that traditional computer intelligent composing objective evaluation is not accurate, the difficulty is too big, this paper on intelligent composition discusses and gives a genetic algorithm intelligent composing model, and puts forward a simplified genetic algorithm based intelligent composing model. For the model from two aspects of algorithm design and realization of the algorithm is discussed, and the program is given out. The experimental results show that the model to improve the local without affecting other parts, has high scalability, the algorithm's time and space complexity is better, proves that this algorithm is feasible and effective.

Key words: artificial intelligence; genetic algorithm; computer music composition model; algorithm

1 引言

所谓计算机作曲, 是指在计算机的帮助下, 在人的参与下进行作曲. 按照人的参与程度可分为初阶智能和高阶智能. 初阶智能是指人直接参与到音乐创作中, 不断诱导计算机作曲; 而高阶智能是指人间接参与到音乐创作中, 通过制定规则或检验结果等手段影响最终生成的音乐[1]. 考虑到人工智能在高阶智能的计算机作曲中更能体现其优势, 本文将集中讨论高阶智能下的计算机作曲. 现阶段对人工智能在计算机作曲中应用的研究主要有四种模型: 分形音乐模型, 马可夫链(Markov chain)模型, 遗传算法(Genetic Algorithm)模型和人工神经网络(Artificial Neural Networks, ANN)模型[2]. 这四种模型各有特色, 但都不完善, 将在此分别对其进行探讨:

(1) 分形音乐. 分形音乐是分形几何学在智能作曲中的应用, 利用自相似原理来建构一些带有自相似小段的合成音乐, 主题在带有小调的三翻五次的返复循环中重复, 在节奏方面可以加上一些随机变化, 它所创造的效果, 但同时, 在实际应用中, 分形音乐的效果还有待考究[3].

(2) 马儿可夫链. 马儿可夫链的只能给出下一个音符而非旋律片断, 有失音乐的连贯性, 有失音乐的神韵.

(3) 遗传算法. 遗传算法有两大优势: 算法非常成熟; 和实现比较简单. 但是, 用遗传算法进行智能音乐生成, 选取合适的适应值函数(fitness function)是非常富于挑战性的工作, 现阶段并没有研究得出最优的适应值函数[4].

(4) 人工神经网络. 人工神经网络需要收集大量的作品来训练它. 因此, 人工神经网络技术更适合用于

① 收稿时间:2016-02-02;收到修改稿时间:2016-04-05 [doi:10.15888/j.cnki.csa.005395]

分析音乐作品而不是创作[12]. 同时, 与别的方法相比, 人工神经网络通常只能创造一些比较简单的旋律[5].

纵观上文介绍的四种模型, 他们都各有所长, 但各自都有缺憾. 对于计算机智能作曲, 国内学者对此也进行了一些研究, 具有代表性的是曹西征等发表的《基于遗传算法的智能作曲技术研究》对交互式遗传算法在计算机智能作曲中应用进行研究, 采取这种方式的优点是通过时值修正来解决进化过程中乐曲每小节各音符的时值之和的不稳定问题, 经过实验具有一定的效果, 但是此研究也有一定的缺陷, 如在计算机来自动生成音符序列需要评估人群一次次地不厌其烦地进行评价, 使得乐曲的产生具有较大的主观性[6].

本文的最大亮点在于给出了切实可行的理论模型, 并就模型给出了非常高效而具体的算法,更分别从模型、算法和实现三个层面把本文的成果和现阶段学术界的研究成果作出了客观的比较和评价, 最终证明本文的观点的可行性.

2 模型和算法设计

2.1 方案选择

通过前面的分析, 并且经过反复权衡和折衷, 在此提出计算机作曲的模型. 考虑到现阶段人工神经网络尚处于理论研究阶段, 应用还不太成熟, 对很多程序员而言是一种较陌生的算法, 因此在此给出另一种更简单的实现方法(见图 1).

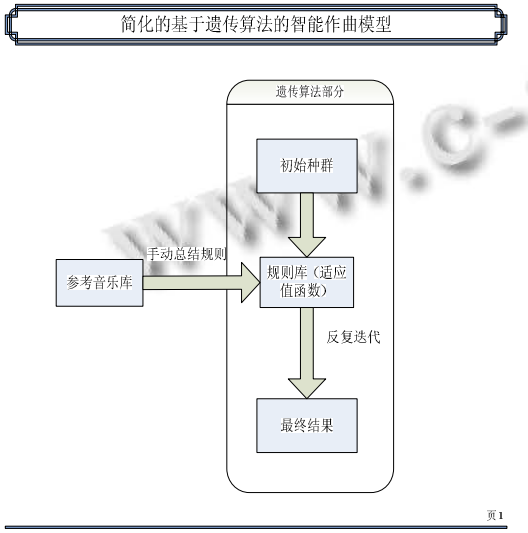

图 1 简化的基于遗传算法的智能作曲模型

3 算法设计

遗传算法将优化问题看作是自然界中生物的进化过程, 通过模拟大自然中生物进化过程中的遗传规律, 来达到寻优的目的. 图 2 为示意图.

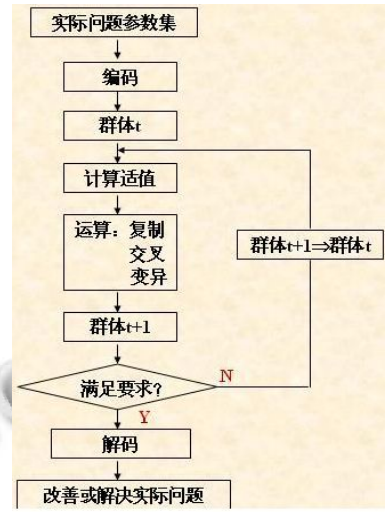

图 2 遗传算法示意图

使用遗传算法作为智能生成音乐的主算法, 首先需要设定好音乐到遗传算法种群的映射, 然后要定义好选择、交配和变异这三个主要操作. 将在下文逐一进行探讨.

3.1 实际问题参数集到遗传算法的映射

设计遗传算法, 首先要面对的是从实际问题参数集到遗传算法种群的映射. 经过仔细分析, 认为一个染色体对应一段音乐(问题的解)比较合适, 这样既可以用一个染色体表示一个可能解, 也可体现出面向对象的思想, 方便程序实现. 而种群则是所有可能解的集合, 表 1 给出清晰的对应关系.

表 1 遗传算法属性解释表

遗传算法参数	实际问题集参数
染色体	解的编码（一段音乐）
基因	编码的元素（例如音高，音长等）
群体	被选定的一组解
种群	根据适应函数选择的一组解
群体规模	群体中染色体的数量
交配概率	染色体互换基因片段的概率
变异概率	染色体内基因变异的概率

3.2 选择操作

要实现遗传算法的“进化”, 必须保证更优的基因有更大几率存活下来, 并产生后代, 即被“选择”. 基于这个思想, 必须保证适应值大的染色体可能会被多次从群体选出, 而适应值小的染色体可能会失去被选中

的机会. 因此, 建议采用“轮盘赌”的方法.

设群体的规模为 N, F(xi)(i=1,…N)是其中 N 个染色体的适应值, 则第 i 个染色体被选中的概率由下式给出:

$$p(x_i)=\frac{F(x_i)}{\sum_{j=1}^{N}F(x_j)} \quad (1)$$

即不妨假设有一个共有 N 个格子的转盘, 每一个 xi 在转盘上占有一格, 而格子的大小与 p(xi)成正比关系. 在选择一个染色体时, 先转动轮盘, 待转盘停下后, 指针指向的格子所对应的 xi 就是被选中的染色体.

3.3 交配操作

交配发生在两个染色体之间(即两段音乐之间). 交配的作用在于使不同的音乐可以互相交换旋律片断, 促进进化的效果. 建议先选定要交换的两个染色体, 然后选定要交换的基因片断(音乐片段)的起点和终点, 然后进行交换, 过程如下: 设 a、b 是两个交配的染色体:

a: $a_1a_2a_3\ldots a_ia_{i+1}\ldots a_ja_{j+1}\ldots a_n$

b: $b_1b_2b_3\ldots b_ib_{i+1}\ldots b_jb_{j+1}\ldots b_n$

设要交换的基因片断的起点为 i, 终点为 j, 交换后的染色体为:

a: $a_1a_2a_3\ldots b_ib_{i+1}\ldots b_ja_{j+1}\ldots a_n$

b: $b_1b_2b_3\ldots a_ia_{i+1}\ldots a_jb_{j+1}\ldots b_n$

通过选择和交配, 可以保证整个群体像更优解进化, 同时, 在进化过程中, 可以通过交配概率来控制交配的频率和密度.

3.4 变异操作

变异可以用于保持群体的多样性, 避免种群早熟现象(过早的陷入举步最优解)的发生. 考虑到音乐的特殊性, 建议在音符级别进行变异, 例如音高从 do 变异为 mi.

同时, 因为变异有时对群体的进化有破坏作用, 必须小心控制变异的概率, 建议应该以一个很小的概率来控制变异的发生.

4 适应函数的选择

适应函数可以说是遗传算法的灵魂, 毫不夸张地说, 适应函数决定了遗传算法的表现. 特别在智能作曲这种没有客观评价标准的领域, 如何给出一个行之有效而又不失高效的适应函数, 将关系到整个系统的成败. 在简化的基于遗传算法的智能作曲模型中, 必须手动对音乐规则进行提取, 即告诉适应函数, 怎样的音乐是“好听”的. 众所周知, 一个音符的最基础属性包括两方面, 即音高和音长, 将就这两方面分别展开探讨.

4.1 音高的适应值计算

对一个音符, 人最直接的反应在于其音高, 哪怕一个完全没有受过音乐训练的人, 在听到 do 和 fa 的时候都可明确地指出他们是两个不同的音. 因此, 可以认为, 对音高适应值的计算在智能作曲中具有决定性的作用.

首先, 必须引入度和音程的概念. 度, 就是音与音之间距离的衡量单位. 音与音之间音高的距离, 就叫做音程. 十二个半音构成了有八个基本音级的音列, 大调音阶各音之音的关系是全全半全全全半. 著名的十二平均律就是在此理论基础上建立的. 在此, 把一个八度音阶划分成 12 个音调, 用 0 到 11 表示, 另外用一个属性表示音阶, 用 0 到 7 表示. 如果两个音符音阶相同, 一个音调为 5, 另一个音调为 1, 则它们之间的音程为 5-1=4; 如果两个音符一个音阶为 3, 音调为 7, 而另一个音阶为 4, 音调为 1, 则它们之间的音程为 4+12*1-7=9. 这样, 通过了音高到(音阶, 音调)的映射, 可以把现实中的音符映射到抽象的参数中去, 为后续算法的开展打下基础.

对音符的评分主要分两方面进行, 首先是段首和段尾的稳定, 然后是和谐音判断, 具体判断准则如下:

1) 检查段首是否为稳定音, 如果调式为大调, 则第一个音符的音调为 0,4,7 为稳定音; 如果调式为小调, 则第一个音符的音调为 0,4,9 为稳定音;

2) 检查段尾是否为稳定音, 规定最后一个音符的音调为 0 为稳定音;

3) 如果检查出该段音乐有稳定音, 则适应值加 10;

4) 遍历整段音乐, 进行和谐音检查;

5) 如果调式为大调, 遇到每个音调为 0,4,7 的音符, 适应值加 2; 如果调式为小调, 遇到每个音调为 2,5,9 的音符, 适应值加 2;

6) 依次计算出每个音符与前面音符的音程;

7) 如果音程为 0,12,则该音程为极完全协和音程, 适应值加 2;

8) 如果音程为 4,7,则该音程为完全协和音程, 适应值加 4;

9) 如果音程为 3,5,8,该音程为不完全协和音程, 适应值加 1;

10) 如果音程为 1, 2,9,10,11 则该音程为不协和音程, 适应值加 0;

经过上述准则的计算, 可以给出每一个染色体的适应值, 为后面的选择操作提供比较基础.

4.2 音长的设定

音符的音长, 在宏观来看, 即为一段音乐的节奏. 而经研究表明, 一段音乐的感情, 很大程度上取决于该音乐的节奏. 例如急促的节奏容易令人联想到激昂, 振奋的感情, 而舒缓的节奏容易让人放松. 因此, 经过仔细斟酌, 提出一种全新的节奏生成方法, 该方法可有效避免对音符的音长进行类似音高的迭代运算带来的巨大运行成本, 同时也能保证最终作品的水准, 认为可以建立一个精简而又包含最经典节奏的节奏库, 里面分门别类地存放表达各种感情的节奏, 例如欢快, 又例如悲伤等等, 当用户指定某种感情时, 从节奏库中相应类别中随机选择一个节奏, 然后匹配上经遗传算法得出的音高, 即为输出的音乐. 而这个节奏库, 可以手动地从各种经典而著名的音乐中提取, 当然, 算法中最好提供适当的变异, 以此达到多样性的目的. 这个方案, 可以看作是效率与效果的折衷. 如果单纯地使用遗传算法进行节奏生成, 必须首先从浩如烟海的音乐作品中提取出“动听”的节奏, 众所周知, 节奏远比音调复杂, 必将面对巨大的规则库; 然后, 必须付出庞大的运行成本, 规则库的增大, 必然导致运行时间的延长. 另一方面, 如果单纯地从几首音乐中抽取出节奏, 然后生硬地安插到生成的音乐上, 必然导致千篇一律和生硬. 因此, 在反复权衡后, 建议从更大的音乐库中提取更多的节奏, 同时进行在控制范围内的随机变异, 降低重复性.

5 仿真实验

5.1 模型层面比较

本文详细地就基于遗传算法的计算机智能作曲模型(见图 2)进行了讨论, 并在其基础上给出了高效的算法和具体的软件实现. 现在, 将会把这个模型和前言中介绍的 4 个最常用模型作横向比较, 并总结出结论.

一个模型的好坏, 通常可以从是否有坚实的理论基础, 是否易于实现, 是否容易扩展等方面进行衡量. 因此, 本文将会分别从 5 个方面对 5 个模型进行比较:

1) 理论成熟程度;

2) 算法设计难度;

3) 算法实现难度;

4) 模型效果

5) 可扩展性.

① 理论成熟程度

从这个角度考虑, 遗传算法模型显然是最成熟的, 也是应用最广泛的, 其次是马可夫链, 而分形音乐和人工神经网络在智能作曲方面还处于理论研究阶段, 并没有太多成功案例和应用. 本文提出的基于遗传算法的计算机智能作曲模型(见图 2)由于主要思想采用了遗传算法模型, 因此在理论成熟程度是毋庸置疑, 可以认为和遗传算法模型处于同等成熟程度.

② 算法设计难度

人工神经网络由于需要设计出一个算法模拟人类学习和归纳, 其算法设计将非常富有挑战性. 马可夫链模型和遗传算法模型的算法设计难度相仿, 但考虑到遗传算法模型已经有大量的成功案例可供参考, 可以认为遗传算法模型的算法设计难度较低.

③ 算法实现难度

本文提出的基于遗传算法的计算机智能作曲模型(见图 2)的算法实现难度也非常大, 也正是这个原因, 本文用了大量的篇幅对适应函数进行了讨论. 马可夫链模型和分形模型的实现难度都是最小的, 几乎可以把算法直接翻译成代码实现. 而人工神经网络则比较特殊, 其算法实现难度还要取决于其具体的算法设计.

④ 模型效果

分形音乐模型是得分最低的, 因为其效果还有待证实. 而马可夫链的排名仅高于分形音乐, 因为其只可以对某一特定风格的音乐进行模仿, 而非自己创作. 遗传算法模型和人工神经网络模型都有非常优秀的表现. 而本文提出的基于遗传算法的计算机智能作曲模型(见图 2), 由于主要思想采用了遗传算法模型, 其效果也相当优秀.

⑤ 可扩展性

在这一点上, 本文提出的基于遗传算法的计算机智能作曲模型(见图 2)脱颖而出, 在设计之初, 就是以高扩展性作为首要考虑的. 本文提及的基于遗传算法和人工神经网络的智能作曲模型(见图 1)就可以看作是其升级版. 而由于模型设计时采用了高内聚低耦合的思想, 本模型非常易于对局部进行改善而不影响到

其他部件. 人工神经网络模型也具有较高的可扩展性, 而分形音乐模型、马可夫链模型和单纯的遗传算法模型的可扩展性都比较低.

上文已经从 5 个方面分别对 5 个模型进行了比较, 在此给出一个量化的横向比较表(见表 2), 10 分为最高, 0 分为最低.

表 2　5 个模型的比较评分表

	理论成熟程度	算法设计难度	算法实现难度	模型效果	可扩展性	总分
本文算法模型	9	8	6	9	10	42
遗传算法模型	9	8	3	9	6	35

从表 2 可见, 本文提出的基于遗传算法的计算机智能作曲模型(模型 2)总分遥遥领先于现阶段使用最广泛的 4 个模型, 因此, 可以认为, 本文提出的这一模型是成功的. 分数对比见图 3.

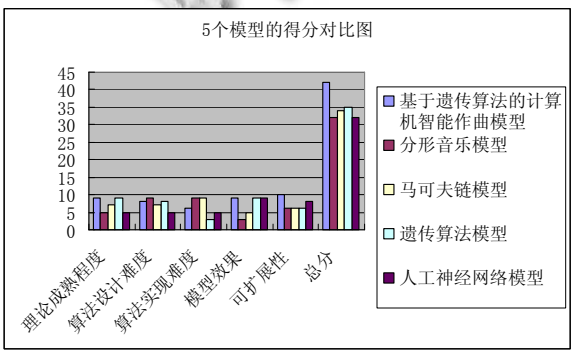

图 3　5 个模型的得分对比图

5.2 算法层面评价

模型的成功并不必然意味着算法的成功. 在得到一个经过验证的、确实可行的模型后, 更应该小心仔细地设计高效的算法.

基于遗传算法的计算机智能作曲模型的核心算法分四个主要部分: 适应函数、选择函数、交配函数、变异函数, 而正如第三章所示, 这四个函数分别为适应函数 Genome::judge(bool major)、选择函数 Genome::select()、交配函数 Genome::crossover()和变异函数 Genome::mutate(). 由于这几个函数都采用了 STL 作为底层数据结构和底层算法, 并进行了非常精心的设计, 因此不但可以完成模型设定的功能, 还能精确地给出每一个算法的时间复杂度和空间复杂度. 下文将分别从四个算法进行具体分析.

① 适应函数 Genome::judge(bool major)

该函数必须遍历整个种群(所有音乐), 对每一个染色体(一段音乐)作出评分. 评分的根据是和弦测试、稳定音测试、跳跃音过滤, 这 3 个操作的时间复杂度为 O(1),且无额外空间需求. 不妨设整个种群的大小为 N,染色体内基因的长度为 m(即该音乐具有的音符数量), 则该函数的时间复杂度为 O(Nm). 而由于所有操作都采用原地操作, 只有一个指针用于指向最优解(用于判断是否进化充分, 若连续几代最优解一致, 则认为进化充分从而停止算法, 给出最终结果), 因此空间复杂度为 O(1),隐藏在这一常量后的具体数量由具体的计算机硬件环境决定(即一个对象指针占用的存储空间).

② 选择函数 Genome::select()

选择函数首先必须先计算出每个染色体(一段音乐)被选中的概率, 该算法的实现原理为先遍历整个种群, 对适应值进行叠加, 然后再次遍历种群, 计算出每一个染色体(一段音乐)被选中的概率(该染色体的适应值 / 总适应值). 不妨设种群大小为 N, 这一操作的时间复杂度为 O(2N),可简化为 O(N),只需求一个额外的 i n t 用于存储总适应值, 因此空间复杂度为 O(1).

然后进行轮盘赌操作, 不妨设整个种群的数量为 N, 由于要选出 N 个染色体, 必须进行 N 各循环, 在每一个循环中, 使用 O(1)的命令, 因此时间复杂度为 O(N),必须采用一个 vector 存储选择出来的染色体, 不妨设每个染色体需要的存储空间为 k, 因此空间复杂度为 O(Nk).

综上, 选择函数的时间复杂度为 O(2N+N),空间复杂度为 O(1+Nk),化简后得到时间复杂度为 O(N),空间复杂度为 O(Nk), 其中 N 为种群大小, k 为每个染色体占用的空间.

③ 交配函数 Genome::crossover()

交配函数遍历整个种群, 然后在相邻的两个染色体之间进行交配进化. 首先必须通过一个概率判定函数, 该函数产生一个随机数, 如果随机数小于交配概率, 则两个染色体进行交配, 反之跳过这两个染色体. 然后用两个随机函数求出交换的基因片断(音乐片断)的起点和终点. 最后把这两段基因片断进行交换. 由于整个函数比较简单, 不妨设整个种群大小为 N, 则时间复杂度为 O(N),空间复杂度为 O(1).

④ 变异函数 Genome::mutate()

对整个种群(即音乐库)进行遍历, 对每一个染色体(一段音乐)进行下列操作:

1) 进行是否变异判定, 原理同交配判定, 时间复杂度为 O(1);

2) 对于通过变异判定的进行音调变异, 首先随机得出产生变异的位置, 然后随机得出变异后的音调, 时间复杂度为 O(1), 由于采用原地操作, 并没有额外空间要求;

3) 进行音阶变异, 首先进行音阶变异判定, 通过变异则继续通过随机得出产生变异的位置, 然后随机得出变异后的音阶, 并对其进行音阶越界判定, 若通过判定则进行音阶变异, 时间复杂度为 O(1), 由于采用原地操作, 并没有额外空间要求.

因此, 变异操作的时间复杂度为 N*[O(1)+ O(1)+ O(1)],无额外空间要求,化简得时间复杂度为 O(N),空间复杂度为 O(1).

综上, 不妨设种群大小为 N, 经过 n 代后得到最优解, 染色体内基因的长度为 m(即该音乐具有的音符数量), 每个染色体占用的空间为 k, 本文设计的遗传算法的时间复杂度为 n*[O(Nm)+ O(N)+ O(N)+ O(N)],空间复杂度为 n*[O(1)+O(Nk)+ O(1)+ O(1)], 化简得时间复杂度为 O(nNm), 空间复杂度为 O(nNk), 需要指出的是, 如果算法运行环境提供良好的垃圾回收(Garbage Collection)机制, 空间复杂度可以压缩到 O(Nk), 因为每次循环中使用的临时存储空间可被及时地回收. 在该算法的时间复杂度中, 并没有出现可能会导致运行时间急剧增加的因子, 而且, 不难看到, 这样的时间和空间复杂度已经达到了遗传算法的下界, 因此, 可以认为, 算法层面的设计是相当优秀的.

为了验证系统的有效性, 选出了 5 人参与了评价实验. 这些人按照要求通过自己主观的感知用当前的系统做出欢快的乐曲. 每产生一代, 都从进化结果中随机选取一个乐曲片段, 经过 10 次进化后, 把选出的 10 个片段再排列起来由 5 人对其进行共同评价, 每一片段都要给出分数, 评价值从 1 到 10, 评价界面如图 4 所示.

图 5 显示出随着进化的发展, 平均评价值的变化趋势. 从图 5 中都能看出随着进化的向前发展, 平均评价值越来越高. 这表明虽然评价者本身存在着个体的差异, 但是随着遗传操作的进行, 系统所产生的乐曲片段已能逐渐符合人的欣赏习惯.

图 4 乐曲评价界面

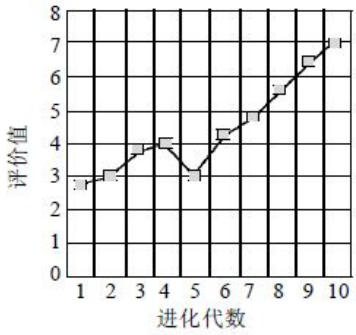

图 5 进化乐曲评价

6 具体实现

本文从模型-算法-实现三个方面对遗传算法在计算机智能作曲中的应用进行了深入探讨, 并最终给出了其软件实现 MyMelody. 在这一部分, 将对市面上流行的几款算法作曲软件进行比较. 参与比较的软件有 MyMelody(本文的模型及算法实现成果)、Punk 音乐生成器、TT 作曲家、恒乐交响曲、Sony Cinescore 这五款. 现在, 将分别从 1)简单易用度, 2)可选自由度, 3)音乐多样化, 4)生成速度和 5)音乐效果对这 5 款软件进行比较, 比较示意图见图 6 所示.

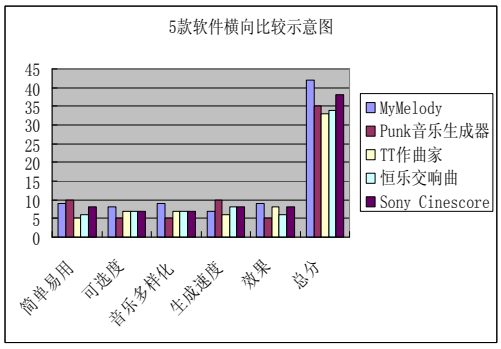

图 6 5 款软件横向比较示意图

从上述比较可以得出, 本软件在同类软件的比较

中脱颖而出, 比较成功地达到了预期目标, 软件的成功意味着模型和算法设计均达到了期望的效果. 简化的基于遗传算法的智能作曲模型主要有 3 个优点:

1) 简单, 不需要设计复杂而庞大的人工神经网络, 并搜集海量的音乐对人工神经网络进行训练, 只需要从各种音乐教材中搜集动听的音乐的共同规律, 并把这些规律转化成程序语句;

2) 便于维护, 如果使用人工神经网络, 一旦感觉创作出的音乐并不能让人满意, 必然要同时对人工神经网络和音乐库进行检查(因为既可能是人工神经网络有瑕疵, 也有可能是因为音乐库里的音乐不够经典), 但若采用模型 2, 只需要修改几条用于表示规则的数据;

3) 模型 2 可以看作是模型 1 的基础, 在模型 2 成功运作后, 可以在不影响整个模型的基础上, 用人工神经网络替代手工总结的规则, 从而进行下一步研究.

7 结语

本文对人工智能在计算机作曲中的方法进行了分析, 提出一种基于遗传算法的智能作曲模型. 对该模型分别从算法设计和算法实现两方面进行了较深入的探讨. 最后, 分别从模型、算法、软件三个层面对本文提出的理论和现有的理论作出了客观比较, 给出评价. 模型是在没有客观判断准则的人工智能应用中的一次大胆尝试, 本身的成功也能给以后的进一步应用打下坚实的基础和提供很好的参考价值.

参考文献

1 张英俐,刘弘,马金刚.遗传算法作曲系统研究.信息技术与信息化,2015,(5):69–72.

2 崔嘉,刘弘.遗传算法在计算机辅助创新作曲中的应用.计算机工程与应用,2014,(3):44–45.

3 冯寅,周昌乐.算法作曲的研究进展.软件学报,2015,(2):187–189.

4 Unehara M, Onisawa T. Construction of music composition, system with interactive genetic algorithm. Proc. of the 6th Asian Design International Conference, 2014

5 廖大强,邹杜,印鉴.一种基于优先级的网格调度算法.计算机工程,2014,40(10):11–16.

6 曹西征,张爱丽,徐久成.基于遗传算法的智能作曲技术研究,计算机工程与应用,2014,(32):79–84.

7 陈智鹏.基于勋伯格十二音体系的作曲算法.计算机工程与设计,2015,(15):59–62.

8 马冬青,王蔚.基于改进遗传算法的星地任务优化调度研究.计算机工程与应用,2014,27(6):49–52.

9 廖大强.面向多目标的云计算资源调度算法.计算机系统应用,2016,25(2):180–189.

10 廖大强,印鉴,邬依林,邹杜.基于兴趣传播的用户相似性计算方法研究.计算机应用与软件,2015,32(10):95–100,104.

11 张方舟,郝庆辉,周勃,刘庆,韩东洋.遗传算法的 RBF 神经网路在线损计算中的应用.计算机技术与发展,2014,(6):69–73.

12 欧阳红祥,陈伟伟,李欣.基于间隔率和遗传算法的多资源均衡优化研究.武汉理工大学学报(信息与管理工程版),2014,(1):129–132.