西安交通大學

自然计算

论文综述报告书

综述题目：遗传算法的原理、应用与挑战

撰 写 人：吴天阳

班　　级：强基数学 002

学　　号：2204210460

日　　期：2023 年 6 月 11 日

报告题目：遗传算法的原理、应用与挑战

学生姓名：吴天阳　**班级：**强基数学 002　**学号：**2204210460

中文摘要：遗传算法是一种随机全局搜索优化方法，它模拟了自然选择和遗传中发生的复制、交叉和变异等现象，从任一初始种群（Population）出发，通过随机选择、交叉和变异操作，产生一群更适合环境的个体，使群体进化到搜索空间中越来越好的区域，这样一代一代不断繁衍进化，最后收敛到一群最适应环境的个体（Individual），从而求得问题的优质解.

遗传算法可以用于解决各种优化问题，从神经网络架构搜索到策略游戏，以及模拟适应和学习现象. 它们能够探索大型和复杂的可能解空间，快速定位有前途的元素，并提供适当的建模工具来描述从游戏到经济学的进化系统. 但是，它们受到高计算成本、难以配置参数和关键解决方案表示等问题的困扰. 最近的发展，如 GPU 计算、并行和量子计算、强大的参数控制方法的概念以及表示策略中的新方法，可能是克服这些限制的关键.

关键词：进化算法，遗传算法.

目录

1 绪论

受达尔文进化论的启发，遗传算法 (Genetic Algorithm, GA) 是一种自适应搜索算法，它核心思想是模拟一些进化的过程：选择、适应性、繁殖、基因重组（也就是基因交叉）以及基因变异. 从而使具有遗传序列的群体（生物体、策略、个体、物体.....）在人工环境中的选择压力下进化，并从中选取适应压力的个体复制并传递其基因，并且使遗传物质进行突变和交换，从而探索新的个体特征.

在该算法由 1975 年 Holland 及 1994 年 Mitchel 和 Forreset 发明之后，该算法的应用不断被拓展并被多样化；其最初的研究的目标是自然系统的自适应性，后来该算法进一步被用于各种领域的优化问题：生物学、经济学、金融学、运筹学、博弈论、深度神经网络、医疗、数据科学等领域.

关于遗传算法的优缺点在不同领域中均有不同的体现，我们将其汇总并主要总结出以下几点并将分开展开讨论：高计算成本、超参数的设置问题以及如何将问题进行正确地表出. 在总结完这些方法后，我们将给出解决每种限制的有效方法.

这些问题一直以来是遗传算法的长期问题 [holland]，但是最近的研究为这些问题提供了新的思路. 例如计算机技术的快速发展：GPU[Cheng, Gen]，云计算 [Liu]，量子计算 [Malossini] 和并行计算 [Tang]，都是从硬件上降低计算成本的分那个发. 在参数调控上也出现了新的解决思路，动态参数配置 [C.Huang]，一定程度上解决了配置超参数困难的问题. 这些方法进一步提高了遗传算法对象的复杂性，从而以应对更复杂的任务.

2 遗传算法

2.1 核心概念

在介绍遗传算法的特点、局限和前景前，我们先介绍其核心概念和主要步骤. 我们首先定义种群、遗传表征以及作为遗传进化驱动力的适应度函数，然后我们描述算法的主要步骤，以及每个步骤涉及的参数.

2.1.1 种群和个体

遗传算法 (GA) 是进化算法 (Evolutionary algorithms, EAs) 中的一个，它们都是基于自然选择启发的搜索方法. 通过模拟达尔文进化论，研究个体如何随群体发生变化的，这些个体可以是任何东西，例如：人工生物体、比特串、计算机程序、金融策略等，随着进化影响个体，通过不断迭代从而最大化适应度，这与单细胞生于向智人发展的方向相同. 基于计算机算法，所有的进化都是将实体表征为个体开始，也就是**遗传表征** (the genetic representation).

2.1.2 遗传表征

遗传算法与其他进化算化或神经网络算法不同点在于它的研究个体不是数学函数或抽象的对象，而是形式化地将遗传学中 DNA 序列表示出来.

基因型 (Genotype) 在遗传算法中经常将基因型表示为二进制编码形式，即由 0 和 1 构成，这与人类的 DNA 类似，遗传信息也可由字母编码存储，例如 A、C、T、G，更多地还有序列、树或数学运算符构成的基因型. 这些字符串通常成为染色体 (chromosomes)，表示基因的元素序列，类似遗传信息在生物中的存储方式. J.Holland[1] 使用术语块 (schema block) 或构建块 (building block) 来描述一组特定基因组，它们就是实体基因型的构建块. 例如，在二进制编码中，每个基因有等位基因 0 或 1，二进制染色体变为一个比特串. 但我们需注意遗传算法通常仅适用于单倍体，即它们的基因型的特征只有一条染色体，

表现型 (Phenotype) 表现型将遗传基因型中的编码对应到实体的特征：形状、大小、颜色、能力等，表现型将信息映射到具体的行为上. 通常认为环境和基因型共同影响其表现型，而非基因性的单独影响. 但过去很多遗传算法认为无论环境如何改变，基因型和表现型之间的对应关系都是固定的.

经过对个体在遗传算法中的表示方式，下面我们将介绍其进化的驱动力，也就是适应性函数.

2.1.3 适应度函数

在自然系统中，进化会将更有繁殖能力的个体保留，它们具有的优势包括：更强大的实力、更具吸引力或更适应环境. 在遗传算法中，适应度 (Fitness) 用于衡量群体中的个体在给定环境或执行某种任务上的能力. 在真实的自然界中，适应性是隐式存在的，个体通过一切可能的方法进行繁殖与生存. 然而，在遗传算法中，我们通过引入适应度函数使得定义变得更加明确，并且常常将表现型与适应度严格关联.

在强化学习中，我们曾学过对与智能体每个行动的奖励值，奖励值的评估与适应度的评估二者十分类似，而在机器学习中，则是最小化损失函数或最大化收益函数.

这些适应度评估函数可以非常简单，例如比特串中的 1 的数目；也有其他非常复杂的例子，例如优化复杂交易策略中的回报、解决多目标优化问题等等. 这些适应度函数隐式或显式地定义了**适应度地貌** (Fitness landscape, [2])，

这种适应度地貌是将表现型与适应度进行关联的物理量. 某些表现型具有较高的适应度，对应于适应度地貌中的山峰或山脉；而某些适应度的表现型较低，对应于适应度地貌中的山谷或洞穴. 遗传算法与机器学习之间的重要区别就是，遗传算法通过维护一个种群的每一个个体的基因型，通过启发式引导的适应度地貌，向更高的山峰移动；而机器学习则是以一个独立的个体进行梯度下降，向着上升速度最快的方向移动.

遗传算法可以被视为一群寻找最高山峰的登山者（类似群智能算法），初始时他们被随机地放置在适应度地貌中，并只能看到眼前的一小部分，每个人设定自己的爬山策略，这些登山者会相互进行交流，结合他们自身的地理位置，在适应度地貌中进行移动，试图攀爬上更高的山峰. 尽管每个人都是以自己利益最大化而前进，但是他们通过其他登山者的信息和策略来提高自己的能力. 通过这样的分散式群体智能，整个登山者群体获得了强大的探索该适应度地貌的能力，从而可以快速识别区域，并达到最高点.

2.2 算法步骤

遗传算法总共执行五个主要步骤，尽管每个遗传算法具体实现方式不同. 首先，生成初始种群；然后，重复以下步骤进行多次迭代：对当前的每个个体进行适应度评估，选择一些个体进行繁衍后代，后代的基因型受到突变的影响从而提高多样性，从而我们获得了新的一代群体. 我们重复进行迭代，直到进化达到某个预期值，或达到迭代上限.

2.2.1 初始化种群

遗传算法通常是从随机染色体开始，这样的采样方法可以保证在搜索空间中均匀的分别初始解，使其能经可能均匀地覆盖个体空间，从而快速找到适应度地貌中更优的区域 [3]，并最大程度地减少搜索偏差.

2.2.2 适应度计算

评估当前一代中所有的个体的适应度，通常使用模拟或函数的方法实现，从而对适应度进行定义. 适应度评估对每个个体分配一个得分，适应度的评估通常分为四种，在 Koza 的论文 [4] 中提到，记**原始适应度** (Raw fitness) 为 r，表示适应度的精确度量，例如：误差率或者比特串中 1 的数量等等；**标准化适应度** (Standardised fitness) $s = r_{max} - r$，其中 r_{max} 表示 r 的上界，标准化适应度将最大化问题转化为最小化问题. **调整适应度** (Adjusted fitness) $a = \dfrac{1}{1+s}$定义了一个 $(0, 1)$ 中的适应度度量，调整适应度可以放大两个个体间微小的适应度差别. **规范化适应度** (Normalised fitness) $n = \dfrac{a}{\sum_{i=1}^{N} a_i}$表示当前个体的适应度相对于整个种群的适应度，我们将在下面分别介绍这四种适应度的选择方法.

2.2.3 个体选择

在我们确定了每个个体的适应度之后，我们将再选择一些个体作为父母，通过两两之间的基因重组，交换遗传信息，并产生两个后代，这些后代将继承父母的信息. 重复上述操作直到我们有足够的后代来构建下一代，由于产生下一代的选择方式是基于适应

度为标准，基于上面给出的四种不同的适应度，我们可以给出很多不同的父母选择方法：

轮盘赌选择 (roulette-wheel selection) 是使用最多且最简单的方法 [3]，每个个体都有成为父母的概率，概率大小就是规范化适应度，这个概率大小与原始适应度大小成正比关系，这种选择方式还包括在下一代中保存当前的最佳个体而不进行修改 [5]，以便保留搜索中得到的最优个体.

排名选择 (Rank selection)[6] 需要指定选择参数，通常可以将参数设为调整适应度或标准化适应度，然后根据排名先后进行父母的选择.

锦标赛选择 (tournament selection)[7] 是通过不同的方法建立个体之间的 Pareto 优势（这是博弈论中的一个名词，表示在多目标优化问题中，一个解在所有目标上至少与另一个解相等，而在某个目标函数上优于另一个解，则称该解 Pareto 优于另一个解），也就是建立序数比较：选择种群的一个子集，将该子集中的个体与子集中的其他个体进行比较，以确定最优个体，这种锦标赛的获胜者将成为父母.

这里我们希望选择最优的个体以改进下一代，而非仅仅关注短期回报或过早地缩小搜索范围，解决这两个问题的分那个发可以通过调节选择方法或者修改相关参数（例如锦标赛中锦标赛的大小）. 如果将整个种群作为锦标赛，适应度较低的个体可能完全没有被选到的可能，但是如果只针对其中的子集进行竞争，我们仍有可能得到一些竞争力相对较弱的个体，这用可以保证我们的模型不容易陷入局部最优解，同时能保持种群的多样性.

2.2.4 基因重组

在基于适应度或排名选择完父母之后，我们需要使用基因重组方法将他们的遗传物质进行结合，以创建种群的新个体. 假设已经选择了两个表现良好的父母，基因重组将他们的基因型进行混合，进而可能得到更优秀的个体

通常的基因重组是基于单点或双点交叉，通过随机选择交换位点，并将左右两段的基因序列进行互换即可，而双点交叉只是在单点的基础上，加入额外的一个交换位点进行基因交换. 这种交叉机制是最常用的重组方式，它借鉴了遗传学中的基因重组，并对应于生殖细胞的减数分裂中的联会的过程，

2.2.5 基因突变

另一种维持物种多样性的方法是基因突变过程，在遗传算法中，两代之间的多样性非常重要 [8]，遗传算法对基因型进行一些随机扰动，从而得到一些有益或有害的突变. 对于有益突变在下一代中会有更高的适应度从而保留下来，而有害的突变会导致较低的适应度从而在下一代中被淘汰掉. 同时，突变还能避免模型过早的收敛，保持模型的均匀搜索 [1].

在遗传算法中，常用的操作是翻转突变和点突变，前者就是随机选择翻转为点，然后将位点前后的串进行翻转；后者就是单点翻转，例如将 0 转化为 1 或者 1 转化为 0. 突变率的选择非常重要，过小的突变概率通过降低多样性来减少损害搜索，而过大的突变率可能会破坏正在进化的良好基因型. 在自然界中，这些确切的突变概率很难估计，范围大概在 10^{-8} 量级上.

突变算子还有很多不同的实现方式，为了动态处理适应度地貌，其最优解会随时间发生变化，**部分超级突变** (partial hyper-mutation)[9] 其方法就是用随机基因替换给定比

例的种群大小，从而使得遗传算法保持对搜索空间的连续变化性；另一种突变是**扩展基因型长度** (duplication mutation)[10] 的重复性突变，这种方法将基因型进行随机翻转，某些部分可能被删除，或将某些部分进行交换或扩展.

3 遗传算法在复杂问题上应用

3.1 最优化问题的鲁棒性解法

遗传算法通常被认为是一种“弱”(weak) 或“鲁棒”(robust) 的方法 [11]，因为其可以用于各种的问题的求解，并在较少假设的条件下给出较好的解. 遗传算法的个体可以抽象为：最大化函数、分类器系统、用于调度的序列，编码函数或算法的参数表格. 遗传算法适用于解决以上这些问题：它是一种广泛适用的通用方法. 在优化问题上，以前的数学研究已经建立了收敛性保证. 将遗传算法视为有限状态 Markov 链，Bhandari 等人的论文 [12] 表明，当迭代次数趋近于无穷时，具有非零突变概率的遗传算法将得到最优解. 同样地，Del Moral 和 Miclo[13] 将遗传算法作为具有竞争选择和突变的粒子群系统，并表明种群渐近地收敛到对应于适应度函数全局最优解，尤其在非线性约束优化下 [14]，发现遗传算法结果与梯度下降方法结果非常接近. 这表明，遗传算法可以演化出与传统方法相似的解决方案，并且可以在更广泛的范围上进行应用.

3.2 通过探索学习架构来发展神经网络

遗传算法已经成功地将其和其他算法混合的方法，例如在降维中使用 k 最近邻分类 [15]. 早期，遗传算法已成功应用于特征子集选择 [16] 和降维 [15]. 这先于与神经网络的进一步组合，并取得了成功. 20 世纪 90 年代首次出现了关于进化网络架构、权重甚至是神经网络使用的学习规则的架构 [17, 18]. 多年后，遗传算法辅助神经网络在建模日常外汇汇率 [19]、日常比特币价格 [20] 方面表现出优异的性能，这样的结构比固定几何神经网络和传统非线性时间序列技术更好. [21] 使用了混合遗传算法和卷积神经网络（CNN），其表现优于其他模型. 他们使用遗传算法来识别最佳 CNN 架构，这是这些程序性能的重要决定因素. 最近的大量研究将遗传算法方法与机器学习技术相结合：传统神经网络也称为进化人工神经网络（银行业绩预测 [22]；棉纱质量 [23]；时间序列预测 [24]；机器生产率 [25]；余热发电过程建模 [26]；空气爆炸预测 [27]；控制器设计 [28]）和基于案例的推理（公司债券评级 [29]；破产预测 [30]）. 最近强化学习研究中的一些突破包括了进化算法的一些关键要素. Deepmind 的开创性 AlphaStar 算法 [31] 使用了 AlphaStar 联赛，可以描述为 AlphaStar 版本的人口与其他版本对抗以改进并找到新的利基策略. 因此，这些成就与进化计算的关键概念 [32] 相关：Larmarckian 进化（由获得的特征继承驱动）、竞争共同进化质量多样性，遗传算法自然地融合了这些概念. 最近，AutoML-zero 算法仅通过进化运算符从头开始演变多层机器学习程序 [33]. 这些程序中所做的计算机模拟以及这些机制包含的遗传算法既有广阔的研究前景.

参考文献

[1] Holland, J. (1992). Genetic algorithms. Scientific American, 267 (1), 66 73.

[2] Wright, S. (1931). Evolution in mendelian populations. Genetics, 16 (2), 97.

[3] Mirjalili, S. (2019). Genetic algorithm. Evolutionary algorithms and neural networks (pp. 43 55). Springer.

[4] Koza, J. R. (1992). Genetic programming: On the programming of computers by means of natural selection (Vol. 1). MIT press.

[5] Ahn, C., & Ramakrishna, R. S. (2003). Elitism-based compact genetic algorithms. IEEE Transactions on Evolutionary Computation, 7 (4), 367 385.

[6] Razali, N. M., Geraghty, J. et al. (2011). Genetic algorithm performance with different selection strategies in solving tsp. Proceedings of the world congress on engineering, 2 (1), 1 6.

[7] Miller, B. L., Goldberg, D. E. et al. (1995). Genetic algorithms, tournament selection, and the effects of noise. Complex systems, 9 (3), 193 212.

[8] Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A niched pareto genetic algorithm for multiobjective optimization. Proceedings of the first IEEE conference on evolutionary computation. IEEE world congress on computational intelligence, 82 87.

[9] Grefenstette, J. J. et al. (1992). Genetic algorithms for changing environments. Ppsn, 2, 137 144.

[10] Lindgren, K. (1992). Evolutionary phenomena in simple dynamics. Artificial life II, 295 312.

[11] Whitley, D. (1994). A genetic algorithm tutorial. Statistics and computing, 4 (2), 65 85.

[12] Bhandari, D., Murthy, C., & Pal, S. K. (1996). Genetic algorithm with elitist model and its convergence. International journal of pattern recognition and artificial intelligence, 10 (06), 731 747.

[13] Del Moral, P., & Miclo, L. (2001). Asymptotic results for genetic algorithms with applications to nonlinear estimation. Theoretical aspects of evolutionary computing (pp. 439 493). Springer.

[14] Homaifar, A., Qi, C. X., & Lai, S. H. (1994). Constrained optimization via genetic algorithms. Simulation, 62 (4), 242 253.

[15] Raymer, M. L., Punch, W. F., Goodman, E. D., Kuhn, L. A., & Jain, A. K. (2000). Dimensionality reduction using genetic algorithms. IEEE transactions on evolutionary computation, 4 (2), 164 171.

[16] Yang, J., & Honavar, V. (1998). Feature subset selection using a genetic algorithm. Feature extraction, construction and selection (pp. 117 136). Springer.

[17] Kitano, H. (1990). Designing neural networks using genetic algorithms with graph generation system. Complex systems, 4, 461 476.

[18] Kitano, H. (1994). Neurogenetic learning: An integrated method of designing and training neural networks using genetic algorithms. Physica D: Nonlinear Phenomena, 75 (1-3), 225 238.

[19] Waheeb, W., & Ghazali, R. (2019). A new genetically optimized tensor product functional link neural network: An application to the daily exchange rate forecasting. Evolutionary Intelligence, 12 (4), 593 608.

[20] Han, J.-B., Kim, S.-H., Jang, M.-H., & Ri, K.-S. (2019). Using genetic algorithm and narx neural network to forecast daily bitcoin price. Computational Economics, 1 17.

[21] Chung, H., & Shin, K.-s. (2020). Genetic algorithm-optimized multi-channel convolutional neural network for stock market prediction. Neural Com- puting and Applications, 32 (12), 7897 7914.

[22] Ravi, V., Kurniawan, H., Thai, P. N. K., & Kumar, P. R. (2008). Soft computing system for bank performance prediction. Applied soft computing, 8 (1), 305 315.

[23] Amin, A. (2013). A novel classification model for cotton yarn quality based on trained neural network using genetic algorithm. Knowledge-Based Systems, 39, 124 132.

[24] Donate, J. P., Li, X., Sánchez, G. G., & de Miguel, A. S. (2013). Time series fore casting by evolving artificial neural networks with genetic algorithms, differential evolution and estimation of distribution algorithm. Neural Computing and Applications, 22 (1), 11 20.

[25] Azadeh, A., Mianaei, H. S., Asadzadeh, S., Saberi, M., & Sheikhalishahi, M. (2015). A flexible ann-ga-multivariate algorithm for assessment and optimization of machinery productivity in complex production units. Journal of Manufacturing Systems, 35, 46 75.

[26] Braun, M. A., Seijo, S., Echanobe, J., Shukla, P. K., del Campo, I., Garcia Sedano, J., & Schmeck, H. (2016). A neuro-genetic approach for model ing and optimizing a complex cogeneration process. Applied Soft Com puting, 48, 347 358.

[27] Armaghani, D. J., Hasanipanah, M., Mahdiyar, A., Abd Majid, M. Z., Amnieh, H. B., & Tahir, M. M. (2018). Airblast prediction through a hybrid ge netic algorithm-ann model. Neural Computing and Applications, 29 (9), 619 629.

[28] Abd-Elazim, S., & Ali, E. (2018). Load frequency controller design of a two-area system composing of pv grid and thermal generator via firefly algorithm. Neural Computing and Applications, 30 (2), 607 616.

[29] Shin, K.-s., & Han, I. (1999). Case-based reasoning supported by genetic al gorithms for corporate bond rating. Expert Systems with applications, 16 (2), 85 95.

[30] Ahn, H., & Kim, K. (2009). Bankruptcy prediction modeling with hybrid case based reasoning and genetic algorithms approach. Applied soft comput ing, 9 (2), 599 607.

[31] Vinyals, O., Babuschkin, I., Chung, J., Mathieu, M., Jaderberg, M., Czarnecki, W. M., Dudzik, A., Huang, A., Georgiev, P., Powell, R., et al. (2019). Alphastar: Mastering the real-time strategy game starcraft ii. DeepMind blog, 2.

[32] Arulkumaran, K., Cully, A., & Togelius, J. (2019). Alphastar: An evolutionary computation perspective. Proceedings of the Genetic and Evolutionary Computation Conference Companion, 314 315.

[33] Real, E., Liang, C., So, D., & Le, Q. (2020). Automl-zero: Evolving machine learning algorithms from scratch. International Conference on Machine Learning, 8007 8019.