

面向装箱问题的量子遗传优化算法

郭 晶 陈贤富

(中国科学技术大学电子科学与技术系 合肥 230027)

摘 要 针对遗传算法系统的维持能力问题,提出一种量子演化算法(a Quantum-Inspired Evolutionary Algorithm)用于解决装箱问题的布局与优化。算法中采用量子比特编码、量子延伸变异操作。同时根据装箱问题具体情况,设计相应的量子旋转门更新策略,并在此基础上引入遗传操作,同时提出 MCBF 算法修复策略。最后,对 8 个测试数据集进行测试。实验测试结果显示,算法在维持遗传基因种群多样性与提高优化质量等方面效果明显。

关键词 量子演化,遗传操作,装箱问题,种群多样性

中图分类号 TP391.4 **文献标识码** A

Quantum Genetic Evolutionary Algorithm for Bin Packing Problem

GUO Jing CHEN Xian-fu

(Department of Electronic Science and Technology, University of Science and Technology of China, Hefei 230027, China)

Abstract Aiming at the problem of the genetic algorithm's system maintenance ability, the paper presented a quantum-inspired evolutionary algorithm for layout and optimization of bin packing problem. This algorithm adopts quantum bit coding and quantum improving mutation. According to the specific situation of bin packing problem, the corresponding rotation gate updating strategy was devised and genetic operation was introduced. Furthermore a new repair strategy called modified complete and best fit algorithm were proposed. Finally, the eight testing sets were tested. The experimental results show this algorithm has a higher performance in maintaining the population diversity of genetic gene and improving the quality of optimization.

Keywords Quantum-inspired, Genetic operation, Bin packing problem, Population diversity

装箱问题(bin packing problem, BPP)是典型的组合优化问题,同时也是典型的 NP 难问题,在现实生活中具有广泛的应用,如集装箱装载问题、内存分配问题、下料问题等^[12]。

装箱问题可以描述为:将 N 个容积不同的待装物品装入容积(C)固定的箱子中,如何用最少数目的箱子装载全部物品,并且每个箱子中物品的容积和不能超过箱子的固定容积 C ^[10]。

目前为止,很多学者设计了用于解决装箱问题的近似算法,例如 Next-Fit(NP)近似算法、First-Fit(FF)近似算法、Best-Fit(BF)近似算法、First-Fit-Decreasing(FFD)近似算法等^[5]。但是这些近似算法的求解结果与物品的容积等数据有较大的关系,甚至在某些特殊的情况下,其求解结果很不理想。1999 年, J. N. D. Gupta 和 Johnny G. HO 提出了最小化闲置空间(Minimum Bin Slack, MBS)算法^[14], 算法把待装物品按照容积递减排序后作为物品输入列表 π , 不断地重复以下步骤,直到物品列表 π 空为止:算法寻找能最小化当前箱子闲置空间的物品列表 s 且将该列表中的物品装入当前箱子中,然后把列表 s 中的物品从物品列表 π 中去掉。

尽作者所知,目前用于解决装箱问题优化质量较高的算法有 Falkenauer 提出的 a Hybrid Grouping Genetic Algorithm(HGGA)^[2]算法和 Bhatia 与 Basu 提出的 a Multi-chromosomal Grouping Genetic Algorithm(MGGA)^[11]算法。HGGA 算法把每个待装物品所装入的箱子编号编码为染色

体,交叉算子由贪婪分割交叉算子^[7]改进而来,选择策略采用 2-竞赛策略和 FF 近似算法。实验结果显示该算法有较高的优化质量,但是当待装物品数量较多时,算法的运行时间较长。MGGA 算法采用多染色体编码方法,即把每个待装物品作为基因,把每个被使用的箱子作为染色体。算法设计了新型的交叉算子和变异算子,同时引入了迁移操作,实验结果显示该算法也有较高的优化质量,且运行时间更短。HGGA 算法的编码方法要求计算所使用箱子数目的上限,增加了算法的复杂度,MGGA 算法很好地解决了 HGGA 算法的不足,由于使用了多染色体编码方法、Better-Fit 近似算法使得运行时间与 HGGA 算法相比更短。但是由于遗传算法易早熟收敛,系统维持能力较差,本文尝试引入量子演化算法 the Quantum-Inspired Evolutionary Algorithm(QIEA)解决此问题。

量子演化算法是基于量子计算机理的新算法,它以量子计算的一些概念和理论为基础,利用量子比特的概率幅编码染色体,以量子旋转门作为更新机制,指导搜索方向,从而实现目标的优化求解^[8]。

在量子演化算法中,利用一个量子比特表达一个基因,一个量子比特可以用式(1)来表示,因此该基因的状态可能是“0”态、“1”态或者“0”态和“1”态的任意叠加态,因此量子演化算法中每个基因表达的不再是一个确定的信息,而是所有可能的信息^[4]。

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \quad (1)$$

式中, (α, β) 是复数, 表示相应比特的概率幅, $|\alpha|^2$ 表示 $|0\rangle$ 的概率, $|\beta|^2$ 表示 $|1\rangle$ 的概率, 且它们满足 $|\alpha|^2 + |\beta|^2 = 1$, 那么一个量子比特可以用一对复数 (α, β) 来定义, 因此长度为 m 的染色体就可以表示为:

$$\begin{pmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{pmatrix} \quad (2)$$

式中, $|\alpha_i|^2 + |\beta_i|^2 = 1, i = 1, 2, \dots, m$ 。量子比特编码的染色体称为量子比特个体。

量子演化算法的更新操作是通过构造量子旋转门操作来实现的, 将该操作作用于量子比特的叠加态, 以更新种群。一般采用的量子旋转门操作如式(5)所示。

本文提出了用于解决装箱问题的量子演化算法, 从实验结果可以显示本文提出的量子演化算法在解决装箱问题上与 HGGA 算法、MGGA 算法相比有更高的优化质量, 同时编码方法更加简洁, 运行时间更短, 效率更高。

1 量子演化算法

1.1 编码方式

本文采用量子比特编码, 具体编码方法如下。

$$\begin{bmatrix} \alpha_{11} & \alpha_{21} & \cdots & \alpha_{N1} & \alpha_{12} & \cdots & \alpha_{ij} & \cdots & \alpha_{NN} \\ \beta_{11} & \beta_{21} & \cdots & \beta_{N1} & \beta_{12} & \cdots & \beta_{ij} & \cdots & \beta_{NN} \end{bmatrix} \quad (3)$$

式中, $|\alpha_{ij}^2|$ 表示物品 i 装入箱子 j 中的概率, $|\beta_{ij}^2|$ 表示物品 i 装入其他箱子的概率同时

$$|\alpha_{ij}^2| + |\beta_{ij}^2| = 1, i = 1, 2, \dots, N; j = 1, 2, \dots, N。$$

1.2 初始化种群

种群中全部量子比特个体的所有比特基因 $(1/\sqrt{2}, 1/\sqrt{2})$, $(\alpha_{ij}, \beta_{ij})$ 都被初始化为它表示一个个体是有可能状态的等概率叠加。

1.3 种群测量

对种群中的每个量子比特个体进行测量, 以获得确定解。主要方法为: 随机产生一个 $[0, 1]$ 数, 若它大于 $|\alpha_{ij}^2|$, 则测量结果为物品 i 装入箱子 j 中, 反之亦然。

1.4 适应度评估

本文采用 Falkenauer 提出的适应度评估函数^[2]。本文的目标是最大化该适应度评估函数值。

$$f_{chrom} = \frac{\sum_{i=1}^n (\frac{F_i}{C})^2}{N} \quad (4)$$

式中, N 表示此个体所使用的箱子数目总和, F_i 表示箱子 i 中所装物品的容积之和, C 表示箱子的固定容积。

1.5 量子操作

量子比特状态的更新是通过以下量子操作进行的。

1.5.1 量子旋转门

本文采用量子旋转门操作^[3]。

$$\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = U(\theta_i) \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \cdot \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix} \quad (5)$$

式中, θ_i 表示旋转角, 它决定旋转的大小和方向。旋转角是对个体中每个箱子的不同情况分别选取的。旋转角调整策略如表 1 所列。

表 1 中 x_i 和 b_i 表示当前解和当前最优解的第 i 个量子比特位即物品 i 的装载情况; $f(\cdot)$ 表示当前解和当前最优解的适应度函数值; 如果当前解和当前最优解的物品 i 分别装入箱子 j 和 k 中, $c(\cdot)$ 则相对应地表示箱子 j 和 k 中所装载

物品的容积之和。在计算 $c(\cdot)$ 之前, 需对当前个体中的箱子按照所装物品容积之和进行降序排序。

表 1 旋转角选择策略

x_i	b_i	$f(x) \geq f(b)$	$c(x_i) \geq c(b_i)$	θ_i
0	0	true	true/false	0.1π
0	0	false	true/false	0.2π
0	1	true	true/false	0.1π
0	1	false	true/false	0.2π
1	0	true	true/false	$\pm 0.4\pi$
1	0	false	true/false	0.1π
1	0	false	true/false	0.2π
1	1	true	true/false	$\pm 0.4\pi$
1	1	false	true/false	0.1π
1	1	false	true/false	0.2π

1.5.2 量子延伸变异

变异操作可以有效地防止早熟收敛并且能够提高局部搜索能力^[13], 算法中改进非门操作^[6]设计延伸变异算子。

1) 从种群中以概率 $p_{mutation}$ 随机选取若干个个体;

2) 对选中的个体随机产生变异位置 pos ;

3) 对位置 pos 的概率幅执行量子非门操作, 即交换位置 pos 的 α_i 和 β_i 。

1.6 遗传操作

本文主要采用 Flakenauer 提出的遗传操作^[2], 并针对量子比特编码进行修改, 遗传操作的主要目的是产生儿子个体 son , 对种群进行遗传更新。

交叉操作: 随机选取进行交叉操作的两个个体, 分别记为个体 p_1 和个体 p_2 , p_1 和 p_2 所使用的箱子数目分别记为 m_1 和 m_2 , 且 $m = \min(m_1, m_2) - K_{left}$, K_{left} 为一已知固定值。重复以下步骤 m 次: 随机生成概率 p , 若 $p \geq p_{cross}$, 则从个体 p_1 中选取闲置空间最少的箱子, 否则从个体 p_2 中选取闲置空间最少的箱子, 将选择的箱子放置到儿子个体 son 中, 且将已选择箱子的物品从个体 p_1 和 p_2 中去掉。

变异操作: 随机选取进行变异操作的个体, 从该个体中随机取出 k 个箱子且将箱子中的物品全部取出, 根据 MCBF (Modified Complete and Better Fit) 算法将取出的物品重新装入箱子中, 形成儿子个体 son 。

1.7 修复策略

在种群初始化和更新的过程中, 会产生非法解。非法解的情况主要分为两类: 箱子中物品的容积之和超出箱子自身容积 C 和有的物品未装入任何箱子中。

对于第一类非法解, 我们采用贪心策略。即每次取出箱子中容积最小的物品, 直到箱子中物品容积之和小于 C , 非法解变成合法解为止。

对于第二类非法解, 本文针对 Alok Singh 和 Ashok K. Gupta 提出的完全-更适合启发式 CBF (Complete-Better Fit) 算法^[2]进行改进, 设计 MCBF (Modified Complete and Better Fit) 算法。通过试验数据, 显示出 MCBF 算法的装箱效果更好。

MCBF 算法分为 3 个部分:

1) 固定部分: 对当前待装物品, 依次尝试每个箱子, 观察是否可以将某个箱子正好完全装满, 若能将某个箱子正好完全装满, 则装入该箱子, 算法结束; 若不能, 执行循环部分。

2) 循环部分: 对当前待装物品, 将每个箱子依次尝试 1-1 替换和 1-2 替换。1-1 替换即用当前待装入的物品替换当前箱子中的容积最小的物品且替换后必须满足箱子容积限制。1-2 替换即用当前待装物品替换当前箱子中的容积最小的两个物品且替换后必须满足箱子容积限制。当然, 替换物品的容积要

大于被替换物品的容积或者容积之和。被替换下的物品再尝试执行固定部分。最后未被装入的物品按照 BFD 算法装入箱子。若不能装入任何箱子,则打开新的箱子。

3)检测部分:对没有完全装满的箱子进行检测,用两个容积都大于等于 $\frac{1}{6}C$ 的待装物品替换该箱子中的容积最小的物品,且替换后满足箱子容积限制要求。

2 算法流程

算法采用针对装箱问题的量子比特编码方法,首先按照 1.2 节的方法随机生成初始种群,然后对初始种群进行测量以获得量子比特个体的单一态,检测测量后的个体是否出现非法解,若出现,则利用修复策略进行修复,接下来计算每个个体的适应度函数值,记录最优个体,然后对当前种群进行量子旋转门更新操作和量子延伸变异操作,随后种群中的每个个体概率性地选择进行交叉操作或者变异操作,生成儿子个体 son,替换种群中的最差个体,判断是否满足终止条件,若满足,则算法结束,否则,算法继续执行。

装箱问题的量子演化算法流程如图 1 所示。

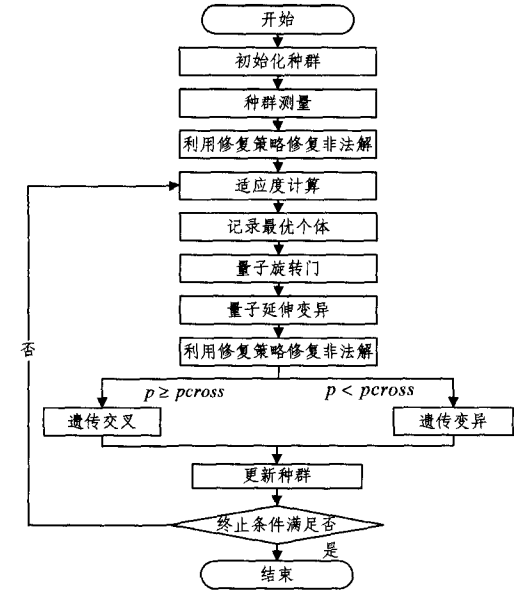


图 1 QIEA 算法流程图

3 试验研究

本文采用 OR-Library 的 u120, u250, u500, u1000, T60, T120, T249, T501 的 8 个测试数据集^[1]。具体数据如表 2 所列。

表 2 测试数据集^[1]

数据集	实例数目	箱子容积	物品数目
u120	20	150	120
u250	20	150	250
u500	20	150	500
u1000	20	150	1000
T60	20	1000	60
T120	20	1000	120
T249	20	1000	249
T501	20	1000	501

表 3 是关于多种修复启发式算法解决表 2 装箱测试数据集的结果比较。其中 NFD, FFD, BFD, MFFD 算法试验结果数据来自参考文献[9]。

表 3 修复策略算法结果比较(使用箱子数目的平均值)

	u120	u250	u500	u1000	T60	T120	T249	T501
NFD	68.6	142.6	282.4	562.7	24.6	48.6	100.1	201.2
FFD	49.8	103.1	203.9	405.4	23.2	45.8	95	190.1
BFD	49.8	103.1	203.9	405.4	23.2	45.8	95	190.1
MFFD	49.7	103	203.6	404.9	23.2	45.8	95	190.1
CBF	49.8	103.1	203.5	405.1	23	45.7	94.9	190.4
MCBF	50.8	103.4	204.5	404.4	21.8	42.3	87.2	173.9

表 3 的实验结果显示, MCBF 算法除了数据集 u120, u500 以外, 都能得到比其他算法更优的解, 且在数据集 T60, T120, T249, T501 上, 解的质量有优。所以本文采用 MCBF 算法作为 QIEA 算法的修复策略。

表 4 是 HGGA 算法和 MGGA 算法与本文提出的 QIEA 算法的实验结果比较。表中 HGGA 算法和 MGGA 算法分别运行 100 次。QIEA 算法运行 1 次, 且交叉概率 $pcross=0.8$, 量子变异概率 $pmutation=0.1$ 。

表 4 QIEA 算法、MGGA 算法和 HGGA 算法的测试比较结果表

数据集 ^[1]	实例数 目总和	HGGA		MGGA		QIEA	
		最优值	运行时间	最优值	运行时间	最优值	运行时间
u120	20	18	1.701292	18	0.635905	20	0.958405
u250	20	17	39.610823	18	13.803	19	6.937677
u500	20	18	752.04794	18	108.812	19	57371164
u1000	20	16	6572.9562	17	588.88075	18	585.845125
T60	20	6	20.789	7	10.6953	18	1.216115
T120	20	0	0	5	3.7691	12	22.20842
Total	120	75	7387.105	83	726.5961	106	684.5369

测试硬件条件为: 3.10GHz, 3.49G 的内存, WIN-XP 操作系统, VC++ 环境。

表 4 中“最优值”表示该算法成功解决的实例数目, “运行时间”为每个实例每次的平均运行时间。

表 4 的实验数据显示, QIEA 算法在解决数据集 u120, u250, u500, u1000 时, 数据集中每个实例的平均运行时间都小于 HGGA 算法的平均运行时间, 特别是当解决数据集 u1000 时, QIEA 的平均运行时间更是远远小于 HGGA 算法。

MGGA 算法有两个数据集的运行时间要优于 QIEA 算法, 分析可能原因如下: 由于 QIEA 算法引入了量子旋转门更新操作, 提高了算法的空间复杂性和计算复杂性, 特别是计算时涉及到了浮点运算, 因此可能延长算法的执行时间。

由于 QIEA 算法利用了量子比特编码, 因此提高了种群的多样性和系统维持能力, 克服了遗传算法的早熟收敛的情况。同时, QIEA 算法与 HGGA 算法相比较, 其不需要提前计算所使用箱子数目的上限值, 使算法更加简洁。

在解决数据集 T60 和 T120 时, QIEA 算法的效果更加明显, 不仅能成功解决更多的实例数目, 而且运行时间更短。

结束语 理论分析与部分实验结果显示, 在系统维持能力和种群多样性方面, QIEA 算法优于 HGGA 算法和 MGGA 算法, 面向特种装箱的实验应用方面可以显示良好的优化效果。需要指出的是装箱问题是一个强约束多目标优化经典 NP 难题, 有关一般三维强约束装箱应用问题有待进一步扩展研究。

参考文献

[1] Beasley J E. OR-Library: Distributing test problems by electronic mail [J]. Journal of Operational Research Society, 1990, 41 (11): 1069-1072

模和效率。关于这方面,我们也通过小的实验给予了验证。

虽然本文给出的验证正确的 ACTL 性质的限界模型检测方法完全基于完备上界的方法相比要好很多,同时也更利于实现,但是,该方法还是不完备的,这也与限界模型检测本身的特点有一定的关系。在今后的工作当中,我们可以结合多种不同的方法,如规约法^[15]、内插法^[16]等来有效地解决这一问题。同样,还可以将该方法应用在文献^[17,18]的工作基础上,以利于基于 SAT 的限界模型检测方法在更广泛的领域得到应用,如安全策略模型的验证、无线传感网中路由协议的安全性验证等。

参考文献

- [1] Clarke E M, Emerson A. Design and synthesis of synchronization skeletons using branching-time temporal logic [C]// Lecture Notes in Computer Science, 1981, 131: 52-71
 - [2] Clarke E M, Grunberg O, Peled D A. Model Checking [M]. MIT Press, 1999
 - [3] Biere A, Cimatti A, Clarke E, et al. Symbolic model checking without BDDs [C]// Proc. of TACAS 99, 1999: 193-207
 - [4] Biere A, Cimatti A, Clarke E, et al. Symbolic Model Checking using SAT procedures instead of BDDs [C]// Proc. of DAC 99, 1999: 317-320
 - [5] Bryant R E. Graph-based algorithms for Boolean function manipulation [J]. IEEE Transactions on Computers, 1986, 35(12): 1035-1044
 - [6] McMillan K L. Symbolic Model Checking [M]. Kluwer Academic Publishers, Boston, 1993
 - [7] Coudert O, Madre J C. A unified framework for the formal verification of sequential circuits [C]// Proc. ICCAD 90, 1990: 126-129
 - [8] Emerson E A, Clarke E M. Using branching-time temporal logics to synthesize synchronization skeletons [J]. Science of Computer Programming, 1982, 2(3): 241-266
 - [9] Penczek W, Wozna B, Zbrzezny A. Bounded Model Checking for the Universal Fragment of CTL [J]. Fundamenta Informaticae, 2002, 51(1/2): 135-156
 - [10] Zhang Wen-hui. Verification of ACTL Properties by Bounded Model Checking [C]// EUROCAST 2007. Lecture Notes in Computer Science, 2007, 4739: 556-563
 - [11] Xu Yan-yan, Chen Wei, Xu Liang, et al. Evaluation of SAT-based Bounded Model Checking of ACTL Properties [C]// Proceedings of the 1st Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering, 2007: 339-348
 - [12] Xu Liang, Chen Wei, Xu Yan-yan, et al. Improved Bounded Model Checking for Universal Fragment of CTL [J]. Journal of Computer Science and Technology, 2009, 24(1): 96-109
 - [13] 徐亮. 限界模型检测方法及其应用[D]. 北京: 中国科学院研究生院, 2009
 - [14] Pieprzyk J, Qu Cheng-xin. Rotation-Symmetric Functions and Fast Hashing [C]// Proc. of ACISP 98, 1998: 169-180
 - [15] de Moura L, Ruess H, Sorea M. Bounded Model Checking and Induction: From Refutation to Verification [C]// Proc. of CAV 2003, 2003: 14-26
 - [16] Jhala R, McMillan K L. Interpolation and SAT-Based Model Checking [C]// Proc. of CAV 2003, 2003: 1-13
 - [17] Xu Liang. SMT-based Bounded Model Checking for Real-time Systems [C]// Proceedings of the Eighth International Conference on Quality Software, 2008: 120-125
 - [18] Xu Liang. Improved SMT-based Bounded Model Checking for Real-time Systems [J]. Journal of Software, 2010, 21(7): 1491-1502
-
- (上接第 69 页)
- [2] Singh A, Gupta A K. Two heuristics for the one-dimensional bin-packing problem [J]. OR Spectrum, 2007, 29(4): 765-781
 - [3] Kuk-Hyun, Kim J-H. Quantum-Inspired Evolutionary Algorithm for a Class of Combinatorial Optimization [J]. Evolutionary Computation, 2002, 6(6): 580-593
 - [4] Gu Jin-wei, Gu Man-zhan. Anovel competitive co-evolutionary quantum genetic algorithm for stochastic job shop scheduling problem[J]. Computers and Operations Research, 2010, 37(5): 927-937
 - [5] Conffman E G, Galambos J G. Bin packing approximation algorithms; combinatorial analysis [M]. the Netherlands: Kluwer Academic Publishers, 1998: 151-207
 - [6] 杨俊安, 庄镇全. 量子遗传算法研究现状[J]. 计算机科学, 2003, 30(11): 13-15
 - [7] Galinier P, Hao J K. Hybrid evolutionary algorithms for graph coloring[J]. Journal of Combinatorial Optimization, 1999, 3(4): 381-383
 - [8] Han K-H, Kim J-H. Genetic Quantum Algorithm and its Application to Combinatorial Optimization Problem [J]. Evolutionary Computation, 2000, 2: 1354-1360
 - [9] Kim B-I, Wy J. Last two fit augmentation to the well-known construction heuristics for one-dimensional bin-packing problem: an empirical study [J]. The International Journal of Advanced Manufacturing Technology, 2010, 50(9-12): 1145-1152
 - [10] Stawowy A. Evolutionary based heuristic for bin packing problem[J]. Computer and Industrial Engineering, 2008, 55(2): 465-474
 - [11] Bhatia A K, Basu S K. Packing Bins Using Multi-chromosomal Genetic Representation and Better-Fit Heuristic [J]. Computer Science, 2004, 3316: 181-186
 - [12] Chan F T S, Au K C, Chan L Y, et al. Using genetic algorithms to solve quality-related bin packing problem[J]. Robotics and Computer-Integrated Manufacturing, 2007, 23(1): 71-72
 - [13] 陈国良, 等. 遗传算法及其应用[M]. 北京: 人民邮电出版社, 2001: 101-161
 - [14] Gupta J N D, Ho J C. A new heuristic algorithm for the one-dimensional bin-packing problem[J]. Production planning & control, 1999, 10(6): 598-603