是多视角和多层次[1,2]. 多视角强调从多个不同的角度对问题进行描述和处理. 多层次强调从多个粒度层次对问题进行理解和描述, 在不同应用中, 不同的层次可被解释为不同的抽象度、不同的复杂度、不同的尺度、不同的细节. 人们在实际问题求解时, 通常是综合考虑多视角与多层次进行问题求解[1]. 下面以胃病诊断为例, 说明将多视角和多层次结合起来的重要性. 在胃病诊断中, 为了获得准确的诊断结果, 病人通常需要做不同类型的检查, 如常规检查, 通常包括: 是否腹痛、呕吐、出血等; 生化检查, 通常包括: 血常规、糖化血红蛋白、甲状腺功能等; 影像学检查, 通常包括: 腹部泌尿彩超、CT、胃肠镜检查等. 不同类型的检查从不同的角度给出病人的信息, 可以将每一种类型的检查作为一个视角. 在每个视角中, 随着具体检查项目的增加, 病人的信息会越来越详细, 从而构成多个层次. 首先, 如果医生仅通过单一类型的检查, 即单一视角往往很难得到准确的结论. 因此, 医生会将不同类型的检查结合起来进行诊断, 即综合多个视角的信息进行决策. 其次, 在某一视角中, 如生化检查, 医生一般不会让病人一次做完全部的检查项目, 然后给出诊断结果. 而是让病人先做血常规, 然后根据血常规的结果尝试得出诊断结论, 若无法确诊, 再进一步进行糖化血红蛋白等检查. 因此, 在每个视角中, 医生通常从多个层次逐步进行决策. 最后, 实际诊断过程中, 医生会根据实际情况灵活地让病人进行不同视角和不同层次的检查. 例如, 病人可以先进行腹痛、呕吐、血常规以及腹部泌尿彩超检查. 如果通过这些检查无法得出诊断结论, 则再进行糖化血红蛋白检查、CT 检查等. 也就是说, 医生在对病人进行诊断时通常是在不同视角和视角的不同层次之间灵活选择和变换, 会将多个视角、多个层次的检查结果结合起来考虑, 从而获得更加准确的诊断结果. 但是现有的粒计算模型大都是分别独立的研究多视角粒结构或者多层次粒结构, 没有将多视角和多层次结合起来. 多视角粒结构往往包含多个视角, 但每个视角仅包含一个层次. 基于多视角粒结构主要有以下研究工作. Qian 等人[3,4]对 Pawlak 粗糙集进行拓展, 采用一簇等价关系取代一个等价关系对论域进行分类和近似未知概念, 提出了多粒度粗糙集. 多粒度粗糙集中, 每个粒度相当于一个视角, 其本质上是多视角粒结构. 之后, Qian 等人[5]将多粒度粗糙集与三支决策相结合, 提出了多粒度三支决策. 利用多源近似空间表示多粒度空间, Khan 等人[6]提出了强弱上下近似集的概念. Sang 等人[7]提出了基于多源决策系统的决策粗糙集模型, 每个信息源均对应问题的一个视角. Chen 等人[8]基于多种形式的数据操作, 提出了智能数据分析的多视角框架. 多层次粒结构往往仅包含一个视角, 但视角由多个层次构成. 基于多层次粒结构主要有以下研究工作. Yao[9]基于一个嵌套的等价关系序列诱导出多层次粒结构, 提出层次粗糙集. Hong 等人[10]基于粗糙集, 提出了从具有层次属性值的数据中学习多层次确定规则和可能规则的算法. Feng 等人[11]研究了具有层次属性值的层次信息系统, 提出了从不同的属性概念层自上而下挖掘层次决策规则的策略. Wu 等人[12]提出多尺度决策信息系统, 将每个对象在每个属性下的属性值表示为多尺度值, 每个尺度对应一个层次. 在此基础上, 很多学者对多尺度决策信息系统进行了深入研究[13−22].

划分序乘积空间作为一种新的粒计算模型[23], 将多层次粒结构和多视角粒结构相结合, 能够从多个视角和多个层次对问题进行描述和求解, 符合人类的认知习惯. 划分序乘积空间的解空间是格结构, 格结构中的每个节点是一个问题求解层. 在实际问题求解中, 如何在划分序乘积空间中找到合适的问题求解层进行问题求解, 是一个非常重要的研究问题. 由于划分序乘积空间的解空间较大, 如何在划分序乘积空间中找到合适的问题求解层是一个 NP 难问题. 而遗传算法是处理 NP 难问题的常用方法, 因此本文采用遗传算法在划分序乘积空间中选择问题求解层. 考虑到经典遗传算法存在收敛速度慢、容易陷入局部最优等问题. 我们对经典遗传算法进行改进, 提出一种两阶段自适应遗传算法 TSAGA (two stage adaptive genetic algorithm), 使其能够有效地从划分序乘积空间中选择问题求解层. 首先, 在算法第 1 阶段利用具有较少迭代次数的经典遗传算法对划分序乘积空间中问题求解层进行预选取, 并将其作为第 2 阶段初始种群的一部分, 从而使第 2 阶段获得较优的初始种群, 缩小解空间范围优化解空间. 然后, 在第 2 阶段中, 考虑种群的多样性会随着种群进化迭代次数的增加而变化, 我们定义随当前种群进化迭代次数动态变化的自适应选择算子、自适应交叉算子和自适应大变异算子, 从而在优化的解空间中进一步选择问题求解层, 实验结果证明了所提方法的有效性. 相比于其他遗传算法, 论文所提出的两阶段自适应遗传算法的优势主要体现在以下 4 个方面.

(1) 通过两个阶段对解空间进行优化, 缩小了解空间的范围.

(2) 所提的自适应选择算子, 在种群进化的前期, 能够保证最优个体得以保留的同时加快算法收敛速度. 在种群进化的后期, 能够在保留最优个体的同时降低对种群多样性的破坏.

(3) 所提的自适应交叉算子, 能够保证在种群进化的前期, 采用较大的交叉概率, 提高算法的收敛速度. 在种群进化的后期, 采用较小的交叉概率, 降低适应度值高的染色体被破坏的可能性. 同时适应度值较小的染色体能够取较高的交叉概率来优化其基因型, 适应度值较大的染色体能够取较低的交叉概率来保留其基因型.

(4) 所提的自适应大变异算子, 基于大变异操作, 能够保证在种群进化的前期, 采用较小的变异概率来防止优良基因被破坏. 在种群进化的后期, 采用较大的变异概率来增强算法跳出局部最优解的能力.

本文第 1 节介绍划分序乘积空间的相关知识. 第 2 节介绍本文提出的两阶段自适应遗传算法. 第 3 节通过仿真实验验证所提算法的有效性. 第 4 节总结全文.

1 划分序乘积空间

划分序乘积空间同时考虑多层次和多视角[23], 因此可以更加全面地理解和描述问题, 从而更加合理地解决问题. 给定一个论域, 首先, 划分序乘积空间将论域上的一个划分定义为一个层次. 其次, 使用一个嵌套的划分序定义一个层次结构, 表示为一个视角. 最后, 给定多个视角, 则定义了多个线性序关系, 基于多个线性序关系的笛卡尔乘积定义划分序乘积空间. 划分序乘积空间是一个格结构, 格结构中的每个节点都是问题求解的一个解决方案, 称为问题求解层, 每个问题求解层均是多个单层次视角的集合.

定义 1[24]. 一个决策系统定义为一个四元组 $DS=(U,AT=C\cup D,V,f)$, 其中, $U=\{x_1,x_2,\ldots,x_n\}$ 为一个非空有限对象集, 称为论域. AT 为一个非空有限属性集, 其中 C 为条件属性集, D 为决策属性, $C\cap D=\varnothing$, V 是属性集 AT 的值域, $V=\bigcup_{a\in AT}V_a$. $f:U\times AT\to V$ 为信息函数, $f(x,a)\in V_a$ 表示对象 $x\in U$ 在属性 $a\in AT$ 上的取值.

定义 2[23]. 给定一个决策系统 DS, $\pi=\{A_1,A_2,\ldots,A_k\}$ 是论域 U 上的一个非空子集簇. 若 $\bigcup_{i=1}^{k}A_i=U$ 且 $A_i\cap A_j=\varnothing(i\neq j)$, 则称 π 为 U 上的一个划分.

定义 3[23]. 给定一个决策系统 DS, $\pi_1=\{A_1,A_2,\ldots,A_k\}$ 和 $\pi_2=\{B_1,B_2,\ldots,B_t\}$ 是论域 U 上的两个划分. 如果 $\forall B_i\in\pi_2$, $A_j\in\pi_1$ 满足 $B_i\subseteq A_j$, 则称 π_2 比 π_1 细, 或称 π_1 比 π_2 粗, 记为 $\pi_2\preccurlyeq\pi_1$. 若 $\pi_2\preccurlyeq\pi_1$ 且 $\pi_2\neq\pi_1$, 则称 π_1 严格粗于 π_2, 记为 $\pi_2\prec\pi_1$.

定义 4[23]. 给定一个决策系统 DS, 划分序定义为论域 U 上的一簇划分 $P=\{\pi_1,\pi_2,\ldots,\pi_n\}$, 满足 $\pi_n\preccurlyeq\pi_{n-1}\preccurlyeq\ldots\preccurlyeq\pi_1$.

定义 5[23]. 给定一个决策系统 DS, $P=\{\pi_1,\pi_2,\ldots,\pi_n\}$ 是论域 U 上的一个划分序. 全序集合 $(P,\preccurlyeq)$ 定义为一个视角, 记为 v. 其中任一划分 π_i 定义为一个层.

一个划分序可由一个嵌套的等价关系序来构造, 假设 $R=\{E_1,E_2,\ldots,E_n\}$ 是论域 U 上的一个嵌套的等价关系序, 满足 $E_n\subseteq E_{n-1}\subseteq\ldots\subseteq E_1$. $U/E_i=\{[x]_{E_i}|x\in U\}$ 是一个划分, 其中 $[x]_{E_i}=\{y\in U|(x,y)\in E_i\}$ 是 x 关于 E_i 的等价类, $1\leqslant i\leqslant n$. 如果我们将 U/E_i 作为 π_i, 就可以得到一个划分序 $P=\{\pi_1,\pi_2,\ldots,\pi_n\}$.

因此, 视角由划分序定义, 划分序可由嵌套的等价关系序来构造. 现有很多方法可以用来构建嵌套的等价关系序[25−27]. 例如给定一个决策系统, 基于嵌套的属性集序列 $C_1\subset C_2\subset\ldots\subset C_n\subseteq C$ 可以构建一个嵌套的等价关系序. 给定一个多尺度信息系统 $S=(U,A)=(U,\{a_j^k|k=1,2,\ldots,I,j=1,2,\ldots,m\})$[26], 基于嵌套的属性值序列 $V^I\prec V^{I-1}\prec\ldots\prec V^1$, 也可以构建一个嵌套的等价关系序.

给定 m 个视角, 由 m 个视角构成的划分序乘积空间定义如下.

定义 6[23]. 给定一个决策系统 DS, m 个视角 $v_i=(P_i,\preccurlyeq)$, $1\leqslant i\leqslant m$, 其中 $P_i=\{\pi_i^1,\pi_i^2,\ldots,\pi_i^{n_i}\}$, π_i^j 表示第 i 个视角的第 j 个层次, n_i 表示视角 v_i 的层次数, $1\leqslant j\leqslant n_i$. 由 m 个视角构成的划分序乘积空间 $POPS_m$ 定义为 $(P_i,\preccurlyeq)$ 的笛卡尔乘积:

$$POPS_m=(\overset{m}{\underset{i=1}{\times}}P_i,\preccurlyeq_P) \tag{1}$$

其中, $\overset{m}{\underset{i=1}{\times}}P_i=\{(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})|\pi_i^{j_i}\in P_i,1\leqslant i\leqslant m,1\leqslant j_i\leqslant n_i\}$. 偏序关系 $\preccurlyeq_P$ 定义为: $\forall(\pi_1^{j_{11}},\pi_2^{j_{21}},\ldots,\pi_m^{j_{m1}})$, $(\pi_1^{j_{12}},\pi_2^{j_{22}},\ldots,\pi_m^{j_{m2}})\in\overset{m}{\underset{i=1}{\times}}P_i$, $(\pi_1^{j_{11}},\pi_2^{j_{21}},\ldots,\pi_m^{j_{m1}})\preccurlyeq_P(\pi_1^{j_{12}},\pi_2^{j_{22}},\ldots,\pi_m^{j_{m2}})$, 当且仅当 $\forall 1\leqslant i\leqslant m$, $\pi_i^{j_{i1}}\preccurlyeq\pi_i^{j_{i2}}$.

通常, 划分序乘积空间 $POPS_m=(\overset{m}{\underset{i=1}{\times}}P_i,\preccurlyeq_P)$ 可简记为 $POPS_m=(\times P_i,\preccurlyeq_P)$.

定理 1[23]. 划分序乘积空间 $POPS_m=(\times P_i,\preccurlyeq_P)$ 是一个格结构.

详细证明见文献 [23], 由定义 6 和定理 1 可以看出划分序乘积空间是通过多个视角的笛卡尔乘积定义的. 多

视角强调从多个不同的角度对问题进行理解与描述, 在每个视角中又可以从多个层次对问题进行理解与描述. 因此, 划分序乘积空间提供了对问题的多视角和多层次的理解和描述.

划分序乘积空间中的每一个节点都是一个问题求解层, 一个问题求解层是多个单层次视角的集合.

定义 7[23]. 给定一个决策系统 DS, $POPS_m=(\times P_i,\preccurlyeq_P)$ 是由 m 个视角 v_i 构成的划分序乘积空间. $\forall(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})\in\underset{i=1}{\overset{m}{\times}}P_i$, $(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})$ 定义为一个问题求解层, 记为 l.

下面通过一个例子来说明划分序乘积空间.

例 1: 表 1 从 3 个视角给出了 8 位病人的描述信息, 论域 $U=\{x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8\}$, 3 个视角分别为血压视角 v_1、血糖视角 v_2、年龄视角 v_3. 为了简化讨论, 每个视角仅包含一个属性. 在血压视角 v_1 中, a_1, a_2 对应两个尺度的属性值序列. a_1 对应第 1 个尺度值{Normal, Abnormal}. a_2 对应第 2 个尺度值{<90, 90–120, >120}. 基于 a_1, a_2, 可以得到划分序 $P_1=\{\pi_1^1,\pi_1^2\}$. 因此 v_1 可以表示为 $(P_1,\preccurlyeq)$. 同理, 对于血糖视角 v_2, b_1, b_2 对应两个尺度的属性值序列. 基于 b_1, b_2, 可以得到划分序 $P_2=\{\pi_2^1,\pi_2^2\}$. 因此 v_2 可以表示为 $(P_2,\preccurlyeq)$. 对于年龄视角 v_3, c_1, c_2 对应两个尺度的属性值序列. 基于 c_1, c_2, 可以得到划分序 $P_3=\{\pi_3^1,\pi_3^2\}$. 因此 v_3 可以表示为 $(P_3,\preccurlyeq)$.

表 1 病人信息

U	血压 (v_1)		血糖 (v_2)		年龄 (v_3)		心脏病
	a_1	a_2	b_1	b_2	c_1	c_2	
x_1	Normal	90–120	N	<6	≤50	18–30	No
x_2	Normal	90–120	AN	>7.5	≤50	18–30	Yes
x_3	Abnormal	>120	AN	6–7.5	≤50	31–40	Yes
x_4	Normal	90–120	AN	>7.5	≤50	31–40	No
x_5	Abnormal	>120	N	<6	≤50	41–50	Yes
x_6	Abnormal	>120	AN	>7.5	>50	51–65	Yes
x_7	Abnormal	<90	AN	6–7.5	>50	66–100	Yes
x_8	Abnormal	<90	N	<6	>50	66–100	No

对于 v_1, 其 2 个层描述为:

$$\pi_1^1=\{\{x_1,x_2,x_4\},\{x_3,x_5,x_6,x_7,x_8\}\},\ \pi_1^2=\{\{x_1,x_2,x_4\},\{x_3,x_5,x_6\},\{x_7,x_8\}\}.$$

对于 v_2, 其 2 个层描述为:

$$\pi_2^1=\{\{x_1,x_5,x_8\},\{x_2,x_3,x_4,x_6,x_7\}\},\ \pi_2^2=\{\{x_1,x_5,x_8\},\{x_2,x_4,x_6\},\{x_3,x_7\}\}.$$

对于 v_3, 其 2 个层描述为:

$$\pi_3^1=\{\{x_1,x_2,x_3,x_4,x_5\},\{x_1,x_5,x_8\}\},\ \pi_3^2=\{\{x_1,x_2\}\{x_3,x_4\},\{x_5\},\{x_6\},\{x_7,x_8\}\}.$$

则由 3 个视角构成的划分序乘积空间 $POPS_m=(\times P_i,\preccurlyeq_P)$ 如图 1 所示.

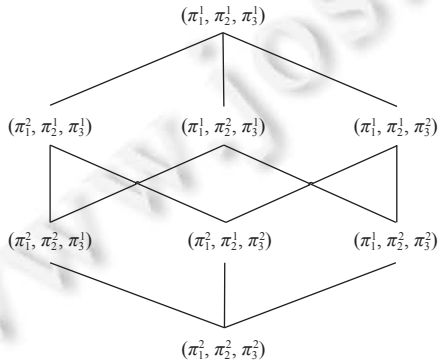

图 1 由 3 个视角构成的划分序乘积空间

从图 1 可以看出, 划分序乘积空间中有 8 个问题求解层, 不仅可以从多个视角解决问题, 还可以从多个层次解决问题. 根据实际诊断需要, 医生可以在不同视角和视角的不同层次之间灵活选择和变换, 从而获得更加准确的诊断结果.

2 两阶段自适应遗传算法

由于划分序乘积空间是格结构, 因此解空间较大, 在划分序乘积空间中找到合适的问题求解层是一个 NP 难问题. 遗传算法是处理 NP 难问题的常用方法[28,29], 但经典遗传算法存在收敛速度慢、容易陷入局部最优等问题. Mathias 等人[30]提出了一种平行两阶段多种群遗传算法, 其在第 1 阶段使用多种群遗传算法来寻找优良的个体, 并将这些个体作为第 2 阶段的初始种群来缩小解空间范围. 但是该方法基于多种群遗传算法, 因此算法复杂度较高. 此外, 该算法没有考虑种群的多样性会随着种群进化迭代次数的增加而变化, 因此其选择算子、交叉算子以及变异算子在整个种群进化迭代期间是固定不变的. 为此, 本文在上述算法的基础上, 提出一种两阶段自适应遗传算法 TSAGA. 算法第 1 阶段利用具有较少迭代次数的经典遗传算法, 从划分序乘积空间中选择问题求解层, 每一次迭代都将当前种群中最好的个体保存在个体集合 *POP* 中. 算法第 2 阶段, 首先, 将 *POP* 中保存的个体作为第 2 阶段初始种群的一部分, 其余染色体仍然按照传统方法随机产生, 从而使第 2 阶段获得较优的初始种群, 优化解空间. 然后, 在第 2 阶段中, 考虑种群的多样性会随着种群进化迭代次数的增加而变化, 定义随当前种群进化迭代次数动态变化的自适应选择算子和自适应交叉算子. 最后, 为了增强算法跳出局部最优解的能力, 定义自适应大变异算子. 从而在优化的解空间中进一步选择问题求解层. 算法详细流程图如图 2 所示.

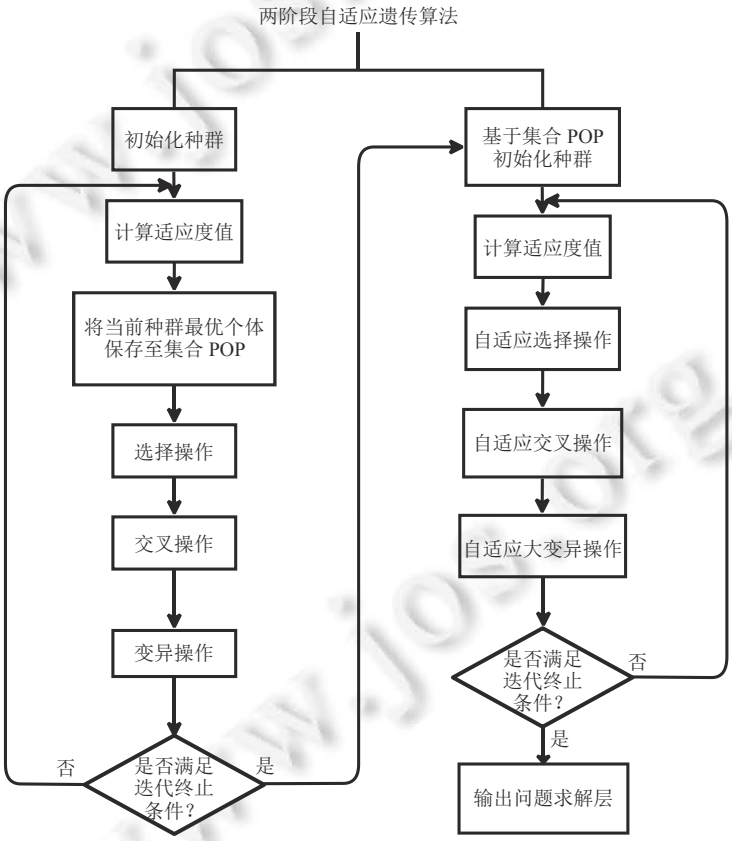

图 2 两阶段自适应遗传算法流程图

接下来将按顺序介绍两阶段自适应遗传算法的实数编码、适应度函数、自适应选择算子、自适应交叉算子以及自适应大变异算子.

2.1 实数编码

给定一个决策系统 DS, $POPS_m=(\times P_i,\preccurlyeq_P)$ 是由 m 个视角 v_i 构成的划分序乘积空间, $1\leqslant i\leqslant m$, $(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})$ 是一个问题求解层, $1\leqslant j_i\leqslant n_i$, 其中 n_i 表示视角 v_i 的层次数. 考虑到对问题求解层使用二进制编码会导致染色体过长, 本文采用实数编码. 设置染色体长度为 m, 每一位对应一个视角, 每一位上的基因值对应该视角下的层次序号. 例如给定问题求解层 $(\pi_1^1,\pi_2^3,\pi_3^1,\pi_4^5,\pi_5^4)$, 其对应的染色体如图 3 所示.

染色体	1	3	1	5	4
	v_1	v_2	v_3	v_4	v_5

图 3 问题求解层的实数编码

2.2 适应度函数

基于粒计算进行问题求解时, 在粗粒度层上可以通过省略细节信息来更抽象和更简略地描述问题, 从而可以以较高的效率进行问题求解, 但求解结果的精度不高. 而在细粒度层上, 通过更具体和更精细的问题描述, 问题求解的效率会降低, 但求解结果的精度会增加[31]. 因此我们希望在划分序乘积空间中选出的问题求解层在满足分类精度要求的同时粒度尽可能粗. 为此, 根据问题求解层的分类精度和粒度定义两阶段自适应遗传算法的适应度函数. 首先给出问题求解层的分类精度和粒度的定义.

划分序乘积空间中每个问题求解层是多个单层次视角的集合, 当基于问题求解层进行问题求解时, 需要对多个视角的层次进行融合. 根据多粒度粗糙集模型[3,4], 具体的融合策略很多, 典型的有乐观融合策略, 悲观融合策略, 可变融合策略等. 在本文中, 我们主要以乐观融合策略为例进行算法设计, 所以问题求解层的分类精度是基于乐观多粒度粗糙集模型定义的. 当基于悲观多粒度粗糙集模型或者可变多粒度粗糙集模型定义问题求解层的分类精度时, 算法设计过程是类似的. 接下来我们基于乐观多粒度粗糙集模型[3,4], 定义问题求解层的分类精度.

定义 8. 给定一个决策系统 DS, $POPS_m=(\times P_i,\preccurlyeq_P)$ 是由 m 个视角 v_i 构成的划分序乘积空间, n_i 表示视角 v_i 的层次数. $l=(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})$ 是一个问题求解层, $\pi_i^{j_i}$ 表示第 i 个视角的第 j_i 层, $1\leqslant i\leqslant m$, $1\leqslant j_i\leqslant n_i$. 假设 $U/D=\{X_1,X_2,\ldots,X_r\}$. l 关于 D 的分类精度定义为:

$$r^O(l,D)=\frac{\sum_{j=1}^{r}\left\{\left|\underline{\sum_{i=1}^{m}l^O_{\pi_i^{j_i}}}(X_j)\right|\right\}}{|U|} \tag{2}$$

其中, $\underline{\sum_{i=1}^{m}l^O_{\pi_i^{j_i}}}(X_j)=\left\{x:[x]_{\pi_1^{j_1}}\subseteq X_j\vee\ldots\vee[x]_{\pi_m^{j_m}}\subseteq X_j\right\}$, "$|\cdot|$"表示集合的基数.

定义 9. 给定一个决策系统 DS, $POPS_m=(\times P_i,\preccurlyeq_P)$ 是由 m 个视角 v_i 构成的划分序乘积空间, n_i 表示视角 v_i 的层次数. $l=(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})$ 是一个问题求解层, $\pi_i^{j_i}$ 表示第 i 个视角的第 j_i 层, $1\leqslant i\leqslant m$, $1\leqslant j_i\leqslant n_i$. 则问题求解层 l 的粒度定义为:

$$g(l)=\sum_{i=1}^{i=m}j_i \tag{3}$$

$g(l)$ 值越大我们认为问题求解层 l 的粒度越细, 反之越粗.

根据问题求解层的分类精度和粒度, 两阶段自适应遗传算法的适应度函数定义如下.

定义 10. 给定一个决策系统 DS, $POPS_m=(\times P_i,\preccurlyeq_P)$ 是由 m 个视角 v_i 构成的划分序乘积空间, n_i 表示视角 v_i 的层次数. $l=(\pi_1^{j_1},\pi_2^{j_2},\ldots,\pi_m^{j_m})$ 是一个问题求解层, $\pi_i^{j_i}$ 表示第 i 个视角的第 j_i 层, $1\leqslant i\leqslant m$, $1\leqslant j_i\leqslant n_i$. $r^O(l,D)$ 和 $g(l)$ 分别是问题求解层 l 的分类精度和粒度. 适应度函数 $f(l)$ 定义为:

$$f(l)=[1-abs(r^O(l,D)-r_a)]\times\alpha+\left(1-\frac{g(l)}{g_{\text{Thinnest}}}\right)\times\beta \tag{4}$$

其中, α 和 β 分别为分类精度与粒度的权重, $0\leqslant\alpha\leqslant1$, $0\leqslant\beta\leqslant1$, $\alpha+\beta=1$, r_a 是问题求解层需要达到的分类精度, g_{Thinnest} 是划分序乘积空间 $POPS_m$ 中最细问题求解层的粒度, $g_{\text{Thinnest}}=\sum_{i=1}^{i=m}n_i$, $abs()$ 返回给定参数的绝对值.

2.3 自适应选择算子

选择算子用于对群体进行优胜劣汰操作, 轮盘赌选择[28]和精英保留策略[32]是两种常用的选择算子. 轮盘赌选择能增大优秀个体被选中的概率, 但无法保证最优秀个体一定能被保留到下一代. 精英保留策略能保证最优秀个体能被保留到下一代, 但无法对种群其他个体进行控制. 由于轮盘赌选择和精英保留策略都是通过牺牲种群的多样性来获得更快的收敛速度, 因此在种群进化的前期, 种群多样性较高, 为了使算法尽快收敛, 将轮盘赌选择和精英保留策略结合使用, 在保证最优个体得以保留的同时加快收敛速度[33]. 在种群进化的后期, 种群的多样性逐渐降低, 仅使用精英保留策略, 在保留最优个体的同时降低对种群多样性的破坏. 为此, 本文提出一种随种群进化迭代次数动态变化的自适应选择算子.

首先通过设置阈值将种群进化过程分为前期与后期. 假设遗传算法的最大迭代次数为 T, 当前迭代次数为 t, $0 \leqslant t \leqslant T$. 给定阈值 $0 \leqslant \gamma \leqslant 1$, 则当 $\frac{t}{T} \leqslant \gamma$ 时, 第 t 次迭代属于进化前期, 当 $\frac{t}{T} > \gamma$ 时, 第 t 次迭代属于进化后期. 通常 γ 的值可以设置为 0.5.

自适应选择算子的伪代码描述如下.

伪代码 1. 自适应选择算子.

1. 初始化最大迭代次数 T, 当前迭代次数 t, 阈值 γ;
2. **if** $\frac{t}{T} \leqslant \gamma$ **then**
3. 选择算子同时使用轮盘赌选择和精英保留策略;
4. **else**
5. 选择算子仅使用精英保留策略;

自适应选择算子, 在种群进化的前期, 能够保证最优个体得以保留的同时加快算法收敛速度. 在种群进化的后期, 能够在保留最优个体的同时降低对种群多样性的破坏.

2.4 自适应交叉算子

交叉算子用于维持种群的多样性[34], 交叉算子以交叉概率为基础, 较高的交叉概率能增加交叉操作发生频率, 加快搜索过程, 但可能会破坏适应度值高的染色体. 较低的交叉概率能降低适应度值高的染色体被破坏的可能性, 但会使得进化速度变慢[35]. 因此, 在种群进化的前期, 我们希望设置较大的交叉概率, 在种群进化的后期, 设置较小的交叉概率. 同时, 在进化过程中, 适应度值较小的个体应该获得较大的交叉概率来优化其基因型, 适应度值较大的个体应该获得较小的交叉概率来保留其基因型. 为此, 本文提出一种交叉概率随当前种群进化迭代次数以及交叉个体适应度值动态变化的自适应交叉算子. 和第 2.3 节一样, 通过设置阈值 γ 将种群进化过程分为前期与后期. 自适应交叉概率定义如下.

定义 11. 给定当前进行交叉的父本染色体 p_1 和母本染色体 p_2, 假设 T 是遗传算法的最大迭代次数, t 是当前迭代次数, $0 \leqslant t \leqslant T$, $f(p_1)$ 和 $f(p_2)$ 是 p_1 和 p_2 的适应度值, $f_{pavg} = avg(f(p_1), f(p_2))$ 是 $f(p_1)$ 和 $f(p_2)$ 的平均值, P_b 是给定的交叉概率分界值, $0.5 \leqslant P_b \leqslant 1$. 则 p_1 和 p_2 的自适应交叉概率 P_c 定义为:

$$P_c = \begin{cases} P_b + (1-P_b) \times (1-f_{pavg}), & \frac{t}{T} \leqslant \gamma \\ P_b - (1-P_b) \times f_{pavg}, & \frac{t}{T} > \gamma \end{cases} \tag{5}$$

其中, $\frac{t}{T} \leqslant \gamma$ 表示第 t 次迭代属于进化前期, $\frac{t}{T} > \gamma$ 表示第 t 次迭代属于进化后期.

由定义 11 可以看出, 进化前期的交叉概率大于进化后期的交叉概率. 此外, 不论在进化前期还是进化后期, 适应度值较小的染色体会获得较高的交叉概率, 适应度值较大的染色体则会获得较低的交叉概率.

本文采用多点交叉的方式进行交叉操作[35]. 自适应交叉算子的伪代码描述如下.

伪代码 2. 自适应交叉算子.

1. 初始化最大迭代次数 T, 当前迭代次数 t, 阈值 γ, 交叉概率分界值为 P_b, 父本染色体 p_1 和母本染色体 p_2;
2. 计算父本染色体 p_1 和母本染色体 p_2 适应度平均值 f_{pavg};
3. **if** $\frac{t}{T} \leqslant \gamma$ **then**
4. $\quad P_c = P_b + (1 - P_b) \times (1 - f_{pavg})$;
5. **else**
6. $\quad P_c = P_b - (1 - P_b) \times f_{pavg}$;
7. 随机生成 0 到 1 之间的一个随机数 r;
8. **if** $P_c > r$ **then**
9. $\quad$ 采用多点交叉的方式产生子代 p_1' 和 p_2' 并用子代 p_1' 和 p_2' 替换父本 p_1 和母本 p_2;
10. **else**
11. $\quad$ 父本 p_1 和母本 p_2 之间不进行交叉操作;

自适应交叉算子, 能够保证在种群进化的前期, 采用较大的交叉概率, 提高算法的收敛速度. 在种群进化的后期, 采用较小的交叉概率, 降低适应度值高的染色体被破坏的可能性. 同时适应度值较小的染色体能够取较高的交叉概率来优化其基因型, 适应度值较大的染色体能够取较低的交叉概率来保留其基因型.

2.5 自适应大变异算子

变异操作用于帮助遗传算法跳出局部最优解. 但是, 传统变异算子会将变异概率设置的极小, 这样就会造成变异操作被执行的可能性很小, 再加上变异操作并不一定能生成更优秀的个体, 所以单靠传统的变异算子很难帮助算法跳出局部最优解. 因此本文基于大变异操作, 同时考虑到在进化前期, 种群多样性较高且种群内优秀个体较少, 此时需要减小变异概率来防止优良基因被破坏. 在进化后期, 种群多样性较低, 此时需要增大变异概率来跳出局部最优解. 为此, 本文提出一种随当前种群进化迭代次数动态变化的自适应大变异算子. 大变异操作[36]是当某一代的最大适应度 $F_{\max}$ 与平均适应度 F_{avg} 满足:

$$\frac{F_{\text{avg}}}{F_{\max}} > \delta \tag{6}$$

则认为当前种群的适应度值较为集中, 将该代中所有个体的变异概率设置为比通常变异概率大 4 倍以上的概率 $P_{m\text{big}}$. 其中 $0.5 \leqslant \delta < 1$, 被称为密集因子, 决定大变异操作被执行的概率, δ 越接近 0.5, 种群被判断为适应度集中的可能性越高, 大变异操作被执行的概率就越大.

同第 2.3 节一样, 通过设置阈值将种群进化过程分为前期与后期. 在进化前期将 δ 设置为一个较大的值, 在进化后期将 δ 设置为一个较小的值.

由于当变异概率过大时算法接近随机搜索, 为了防止种群发散, 本文引入一个条件来判断大变异操作是否可行. 若某个体变异后的适应度值小于变异前个体的适应度值, 将本次变异取消, 种群中仍然保留变异前个体; 否则我们用变异后个体替换种群中的原个体. 本文采用随机单点变异的方式进行变异操作, 即随机将染色体中的一个基因位从当前值变为值域内的一个可行值.

自适应大变异算子的伪代码描述如下.

伪代码 3. 自适应大变异算子.

1. 初始化最大迭代次数 T, 当前迭代次数 t, 阈值 γ, 种群染色体 p, 常规变异概率 $P_{m\text{small}}$, 大变异概率 $P_{m\text{big}}$;
2. **if** $\frac{t}{T} \leqslant \gamma$ **then**
3. $\quad$ 设置密集因子 δ 为较大值;
4. **else**

5. 设置密集因子 δ 为较小值;
6. 计算当前种群最大适应度 $F_{\max}$ 与平均适应度 F_{avg};
7. **if** $\frac{F_{\text{avg}}}{F_{\max}} > \delta$ **then**
8. 变异概率 $P_m = P_{m\text{big}}$;
9. **else**
10. 变异概率 $P_m = P_{m\text{small}}$;
11. 随机生成 0 到 1 之间的一个随机数 r;
12. **if** $P_m > r$ **then**
13. 采用随机单点变异的方式对染色体 p 进行变异, 产生新个体 p';
14. **if** $f(p) > f(p')$ **then**
15. 种群中保留 p, 将 p' 舍弃;
16. **else**
17. 用 p' 替换掉种群中的 p;
18. **else**
19. 不进行变异操作;

自适应大变异算子, 基于大变异操作, 能够保证在种群进化的前期, 采用较小的变异概率来防止优良基因被破坏. 在种群进化的后期, 采用较大的变异概率来增强算法跳出局部最优解的能力.

2.6 算法流程

基于上述实数编码、适应度函数、自适应选择算子、自适应交叉算子以及自适应大变异算子的介绍, 本节给出两阶段自适应遗传算法, 如算法 1 所示.

算法 1. 两阶段自适应遗传算法 (TSAGA).

输入: 划分序乘积空间 $POPS_m = (\times P_i, \preccurlyeq_P)$, 阈值 γ, 第 1 阶段遗传算法最大迭代次数 T_1, 第 1 阶段遗传算法当前迭代次数 t_1, 第 2 阶段遗传算法最大迭代次数 T_2, 第 2 阶段遗传算法最大迭代次数 t_2, 染色体 p, 交叉概率分界值 P_b, 常规变异概率 $P_{m\text{small}}$, 大变异概率 $P_{m\text{big}}$;
输出: 问题求解层.

1. 随机生成实数编码的染色体表示问题求解层, 构成第 1 阶段初始种群 //开始第 1 阶段遗传算法
2. 生成一个空集合 POP;
3. $t_1 = 1$;
4. **while** $t_1 \leqslant T_1$ **do**
5. 根据定义 10 计算种群中染色体的适应度值;
6. 将当前种群中最优染色体保存到集合 POP 中;
7. 按照经典的选择、交叉、变异算子进行选择、交叉、变异操作;
8. $t_1 = t_1 + 1$;
9. **end while**
10. 将 POP 作为第 2 阶段初始种群的一部分, 种群中其余染色体随机产生; //开始第 2 阶段遗传算法
11. $t_2 = 1$;
12. **while** $t_2 \leqslant T_2$ **do**
13. 根据定义 10 计算种群中染色体的适应度值;

14. **if** $\frac{t_2}{T_2} \leqslant \gamma$ **then** //自适应选择算子
15. 同时使用轮盘赌选择和精英保留策略进行选择操作;
16. **else**
17. 仅使用精英保留策略进行选择操作;
18. 将种群中所有染色体任意两个随机组合构成一对父本染色体 p_1 和母本染色体 p_2; //自适应交叉算子
19. 计算父本染色体 p_1 和母本染色体 p_2 适应度平均值 $f_{p\text{avg}}$;
20. **if** $\frac{t_2}{T_2} \leqslant \gamma$ **then**
21. $P_c = P_b + (1 - P_b) \times (1 - f_{p\text{avg}})$;
22. **else**
23. $P_c = P_b - (1 - P_b) \times f_{p\text{avg}}$;
24. 随机生成 0 到 1 之间的一个随机数 r;
25. **if** $P_c > r$ **then**
26. 采用多点交叉的方式产生子代 p_1' 和 p_2' 并用子代 p_1' 和 p_2' 替换父本 p_1 和母本 p_2;
27. **else**
28. 父本 p_1 和母本 p_2 之间不进行交叉操作;
29. **if** $\frac{t_2}{T_2} \leqslant \gamma$ **then** //自适应大变异算子
30. 设置密集因子 δ 为较大值;
31. **else**
32. 设置密集因子 δ 为较小值;
33. 计算当前种群最大适应度 $F_{\max}$ 与平均适应度 F_{avg};
34. **if** $\frac{F_{\text{avg}}}{F_{\max}} > \delta$ **then**
35. 变异概率 $P_m = P_{m\text{big}}$;
36. **else**
37. 变异概率 $P_m = P_{m\text{small}}$;
38. 对种群中染色体随机生成 0 到 1 之间的一个随机数 r;
39. **if** $P_m > r$ **then**
40. 采用随机单点变异的方式对染色体 p 进行变异, 产生新个体 p';
41. **if** $f(p) > f(p')$ **then**
42. 种群中保留 p, 将 p' 舍弃;
43. **else**
44. 用 p' 替换掉种群中的 p;
45. **else**
46. 不进行变异操作;
47. $t_2 = t_2 + 1$;
48. 输出当前种群中最优染色体即为所求的问题求解层;
49. **end while**

对于两阶段自适应遗传算法, 在算法的一次迭代过程中, 对于每个染色体在计算其适应度函数时需要计算分类精度, 时间复杂度为 $O(m \times |U|^2 \times |C|)$, 其中 m 为视角的个数, 即染色体的长度, $|U|$ 为视角中对象个数, $|C|$ 为视角中属性个数, 所以计算种群中所有染色体适应度函数的时间复杂度为 $O(m \times |U|^2 \times |C| \times N)$, 其中 N 为种群规模. 选择算子的时间复杂度均为 $O(N)$. 交叉算子采用多点交叉, 时间复杂度为 $O(N \times m)$. 变异算子采用单点变异, 时间

复杂度为 O(N). 因此, 两阶段遗传算法的第 1 阶段, 迭代次数为 T_1, 其时间复杂度为 O($m\times|U|^2\times|C|\times N\times T_1$). 两阶段遗传算法的第 2 阶段, 迭代次数为 T_2, 其时间复杂度为 O($m\times|U|^2\times|C|\times N\times T_2$). 通常 $T_1\leqslant T_2$, 所以两阶段自适应遗传算法的时间复杂度为 O($m\times|U|^2\times|C|\times N\times T_2$).

对于两阶段自适应遗传算法, 在算法的一次迭代过程中, 对于适应度函数需要计算分类精度, 空间复杂度为 O($m\times|U|\times|C|$), 其中 m 为视角的个数, 即染色体的长度, $|U|$ 为视角中对象个数, $|C|$ 为视角中属性个数. 选择算子的空间复杂度均为 O($N\times m$), 其中 N 为种群规模. 交叉算子的空间复杂度为 O($N\times m$). 变异算子的空间复杂度为 O($N\times m$). 对于两阶段遗传算法的两个阶段, 每次迭代之后空间可以释放, 所以空间复杂度和迭代次数无关, 因此, 两阶段自适应遗传算法的空间复杂度为 O($m\times|U|\times|C|+N\times m$).

3 实验分析

3.1 实验数据

为验证本文提出的两阶段自适应遗传算法 (TSAGA) 的有效性, 我们分别选取了来自 UCI 机器学习数据库[37]和 Kaggle 数据库[38]的 9 个数据集进行实验验证, 数据集的具体信息如表 2 所示. 其中 Sonar 和 Urban 是连续数据集, 对于连续数据集, 我们基于文献 [39] 的等宽分箱法进行离散化处理. 在每个数据集上, 首先, 将条件属性集划分成多个子集 AT_i, $AT_i\subseteq C$, $1\leqslant i\leqslant n$, 每个属性子集 AT_i 定义为不同的视角. 然后, 将每个视角中的属性再划分为多个子集 AT_i^k, $1\leqslant k\leqslant m$. 通过将属性子集组合起来生成嵌套的属性集序列 $C_1\subset C_2\subset\ldots\subset C_n$, 其中 $C_j=\bigcup_{1\leqslant k\leqslant j}AT_i^k$, 构成多个层次. 最后, 基于多个视角和多个层次构建划分序乘积空间. 每个数据集的多视角多层次结构如表 3 所示. 需要注意的是, 在构造划分序乘积空间时, 通过不同的方法, 例如选取属性个数不同, 可以构造出不同的视角和层次, 进而构造不同的划分序乘积空间. 实验中, 我们对于所有的数据集都是先构造出划分序乘积空间, 然后所有实验都是基于相同的划分序乘积空间进行, 从而保证实验的公平性. 但是我们没有考虑不同划分序乘积空间的构造方法对算法实验性能的影响, 我们将在以后的工作中对此问题进行深入研究.

表 2 数据集

序号	数据集	对象数量	属性数量	类别数量
1	Kr-vs-kp	1 056	36	2
2	Dermatology	358	34	6
3	Student-por	649	32	2
4	Divorce	170	54	2
5	Urban	168	147	9
6	Sonar	208	60	2
7	Phishing	11 055	30	2
8	Mushroom	8 124	22	2
9	SCADI	70	205	7

表 3 数据集的多视角多层次结构

数据集	每个视角和层次下的属性				
Kr-vs-kp	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角8	层次1 $\{a_{36}\}$	—	—	—

表 3 数据集的多视角多层次结构 (续)

数据集	每个视角和层次下的属性				
Dermatology	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角7	层次1 $\{a_{31}\}$	层次2 $\{a_{31},a_{32}\}$	层次3 $\{a_{31},a_{32},a_{33}\}$	层次4 $\{a_{31},a_{32},a_{33},a_{34}\}$
Student-por	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角7	层次1 $\{a_{31}\}$	层次2 $\{a_{31},a_{32}\}$	—	—
Divorce	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角11	层次1 $\{a_{51}\}$	层次2 $\{a_{51},a_{52}\}$	层次3 $\{a_{51},a_{52},a_{53}\}$	层次4 $\{a_{51},a_{52},a_{53},a_{54}\}$
Urban	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角30	层次1 $\{a_{146}\}$	层次2 $\{a_{146},a_{147}\}$	—	—
Sonar	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角12	层次1 $\{a_{56}\}$	层次2 $\{a_{56},a_{57}\}$	…	层次5 $\{a_{56},a_{57},a_{58},a_{59},a_{60}\}$
Phishing	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮

表 3 数据集的多视角多层次结构 (续)

数据集	每个视角和层次下的属性				
	视角6	层次1 $\{a_{26}\}$	层次2 $\{a_{26},a_{27}\}$	…	层次5 $\{a_{26},a_{27},a_{28},a_{29},a_{30}\}$
Mushroom	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角5	层次1 $\{a_{21}\}$	层次2 $\{a_{21},a_{22}\}$	—	—
SCADI	视角1	层次1 $\{a_1\}$	层次2 $\{a_1,a_2\}$	…	层次5 $\{a_1,a_2,a_3,a_4,a_5\}$
	视角2	层次1 $\{a_6\}$	层次2 $\{a_6,a_7\}$	…	层次5 $\{a_6,a_7,a_8,a_9,a_{10}\}$
	⋮	⋮	⋮	⋮	⋮
	视角41	层次1 $\{a_{200}\}$	层次2 $\{a_{201},a_{202}\}$	…	层次5 $\{a_{201},a_{202},a_{203},a_{204},a_{205}\}$

3.2 实验方法

划分序乘积空间作为一种新的粒计算模型, 目前还没有在划分序乘积空间中选择问题求解层的方法, 因此本文所提两阶段自适应遗传算法无法直接与其他现有方法进行对比. 但针对多层次粒结构中最优层次选择问题现有的研究相对较多, 代表性工作是吴伟志及其工作团队提出的多尺度决策系统以及针对多尺度决策系统的最优尺度选择[12−20]. 例如针对不协调广义多尺度决策系统中的最优尺度选择问题[13]. 但是文献 [13] 提出的方法本质上是在单视角下进行最优层次选择, 无法直接用于在划分序乘积空间中选择问题求解层. 所以我们先使用文献 [13] 的方法在每个视角下选出一个层, 然后将这些层组合起来构成一个问题求解层. 将该问题求解层与使用所提两阶段自适应遗传算法选出的问题求解层进行比较. 此外, 为了说明两阶段自适应遗传算法中各个改进部分对算法性能的影响, 我们进行了消融实验.

我们共进行 3 组实验. 在第 1 组实验中, 首先, 使用文献 [13] 的方法在每个视角下选出一个广义决策最优尺度, 该尺度能够保持决策系统的广义决策和最细尺度的广义决策完全一致. 然后, 将所有广义决策最优尺度组合起来构成一个问题求解层, 称之为最优问题求解层. 最后, 以该最优问题求解层的分类精度为标准, 在划分序乘积空间中基于两阶段自适应遗传算法选出和最优问题求解层分类精度相同的问题求解层, 比较这两个问题求解层的粒度. 考虑到实际问题求解中, 由于时间和成本等约束, 人们往往不需要选出广义决策最优尺度, 只需要选出广义决策满意尺度, 该尺度只要保持决策系统的广义决策和最细尺度的广义决策在一定程度上一致即可. 在第 2 组实验中, 首先, 使用 Wu 等人的方法在每个视角下选出一个广义决策满意尺度. 然后将所有广义决策满意尺度组合起来构成一个问题求解层, 称之为满意问题求解层. 以该满意问题求解层的分类精度为标准, 在划分序乘积空间中基于两阶段自适应遗传算法选出和满意问题求解层分类精度相同的问题求解层, 然后比较这两个问题求解层的粒度. 在第 3 组实验中, 对两阶段自适应遗传算法进行消融实验, 考虑到篇幅原因, 我们仅基于第 2 组实验进行消融实验. 为了简化讨论, 把使用文献 [13] 的广义决策最优尺度选择问题求解层的方法, 记为 GOS; 把使用文献 [13] 的广义决策满意尺度选择问题求解层的方法, 记为 GSS.

广义决策最优尺度[13]和广义决策满意尺度[13]定义如下.

定义 12[13]. 给定一个广义多尺度决策系统 $S=(U,C\cup\{d\})=(U,\{a_j^k|k=1,2,\ldots,I_j,j=1,2,\ldots,m\}\cup\{d\})$, 记尺度全体为 L. $\forall K\in L$, $\forall x\in U$, 记 $\varepsilon_{C^K}(x)=\{d(y)|y\in[x]_{C^K}\}$ 为对象 x 在决策系统 $S^K=(U,C^K\cup\{d\})$ 中的广义决策. 设 $K_0\in L$

为最细尺度, 若 $\forall x \in U$, 有:

$$\varepsilon_{C^K}(x) = \varepsilon_{C^{K_0}}(x) \tag{7}$$

且 $\forall H \in L$, $K \prec H$, $\varepsilon_{C^K}(x) \neq \varepsilon_{C^{K_0}}(x)$ 我们就称 K 为广义决策最优尺度.

定义 13[13]. 给定一个广义多尺度决策系统 $S = (U, C \cup \{d\}) = (U, \{a_j^k | k = 1,2,\ldots,I_j, j = 1,2,\ldots,m\} \cup \{d\})$, 记尺度全体为 L. 设 $K_0 \in L$ 为最细尺度, 给定阈值 $0 \leqslant \zeta \leqslant 1$, $\forall K \in L$, $\forall x \in U$, 有:

$$\frac{|\{x | \varepsilon_{C^K}(x) = \varepsilon_{C^{K_0}}(x), x \in U\}|}{|U|} \geqslant \zeta \tag{8}$$

且 $\forall H \in L$, $K \prec H$, $\dfrac{|\{x | \varepsilon_{C^K}(x) = \varepsilon_{C^{K_0}}(x), x \in U\}|}{|U|} < \zeta$, 我们就称 K 为广义决策满意尺度.

在两组实验中, 我们设置两阶段遗传算法初始种群大小均为 60. 第 1 阶段遗传算法最大迭代次数为 5 次, 交叉概率为 0.8, 变异概率为 0.1. 第 2 阶段遗传算法最大迭代次数为 40 次, 阈值 γ 为 0.5, 交叉概率分界值 P_b 为 0.8, 常规变异概率 $P_{m\text{small}}$ 为 0.1, 大变异概率 $P_{m\text{big}}$ 为 0.4.

3.3 实验结果

在第 1 组实验中, 我们在每个数据集上均进行 10 次实验, 取 10 次实验结果的平均值作为最终结果. TSAGA 和 GOS 两种方法在每个数据集上所选问题求解层的粒度如表 4 所示.

表 4 TSAGA 和 GOS 所选问题求解层的粒度

数据集	最细粒度	GOS	TSAGA
Kr-vs-kp	36.0	29.0	**23.2**
Dermatology	34.0	34.0	**26.0**
Student-por	32.0	32.0	**28.0**
Divorce	54.0	53.0	**14.9**
Urban	147.0	147.0	**66.8**
Sonar	60.0	60.0	**40.0**
Phishing	30.0	28.0	**24.0**
Mushroom	22.0	22.0	**16.0**
SCADI	205.0	172.0	**82.5**

由表 4 可以看出, 所提两阶段自适应遗传算法 (TSAGA) 在保持和 GOS 分类精度相同的前提下, 在所有数据集上选出的问题求解层都比 GOS 方法选出的问题求解层粒度更粗.

在第 2 组实验中, 设置阈值 ζ 分别取 0.2、0.4、0.6、0.8、1.0, 在每个阈值下均进行 10 次实验, 取 10 次实验结果的平均值作为最终结果. TSAGA 和 GSS 两种方法在不同阈值下所选问题求解层的粒度如表 5 所示. 为了使结果更加直观, 我们将表 5 中的实验数据以图的形式展现出来, 如图 4 所示. 图 4 中横坐标为阈值 ζ, 纵坐标为问题求解层的粒度.

表 5 TSAGA 和 GSS 在不同阈值 ζ 下所选问题求解层的粒度

数据集	最细粒度	$\zeta = 0.2$		$\zeta = 0.4$		$\zeta = 0.6$		$\zeta = 0.8$		$\zeta = 1.0$	
		GSS	TSAGA	GSS	TSAGA	GSS	TSAGA	GSS	TSAGA	GSS	TSAGA
Kr-vs-kp	36.0	19.0	**16.1**	19.0	**16.0**	20.0	**18.0**	23.0	**18.0**	29.0	**23.0**
Dermatology	34.0	21.0	**15.0**	26.0	**21.3**	31.0	**21.3**	34.0	**26.0**	34.0	**26.0**
Student-por	32.0	24.0	**19.1**	26.0	**16.5**	28.0	**26.0**	31.0	**28.0**	32.0	**28.0**
Divorce	54.0	17.0	**14.9**	20.0	**15.0**	35.0	**15.1**	41.0	**14.8**	53.0	**14.6**
Urban	147.0	121.0	**64.1**	132.0	**65.9**	134.0	**67.4**	146.0	**65.4**	147.0	**66.8**
Sonar	60.0	43.0	**26.7**	49.0	**29.7**	53.0	**32.5**	60.0	**40.3**	60.0	**39.8**
Phishing	30.0	15.0	**13.0**	16.0	**13.1**	16.0	**13.0**	21.0	**15.0**	28.0	**24.0**
Mushroom	22.0	18.0	**15.0**	19.0	**16.0**	21.0	**16.0**	21.0	**16.0**	22.0	**16.0**
SCADI	205.0	103.0	**78.5**	135.0	**83.4**	157.0	**82.5**	169.0	**80.6**	172.0	**84.0**

由表 5 和图 4 可以看出, 随着阈值 ζ 的增大, TSAGA 和 GSS 两种方法在大多数数据集上选出的问题求解层的粒度都逐渐变细. 但是, 所提两阶段自适应遗传算法 (TSAGA) 在保持和 GSS 分类精度相同的前提下, 在所有数据集上选出的问题求解层都比 GSS 方法选出的问题求解层粒度更粗.

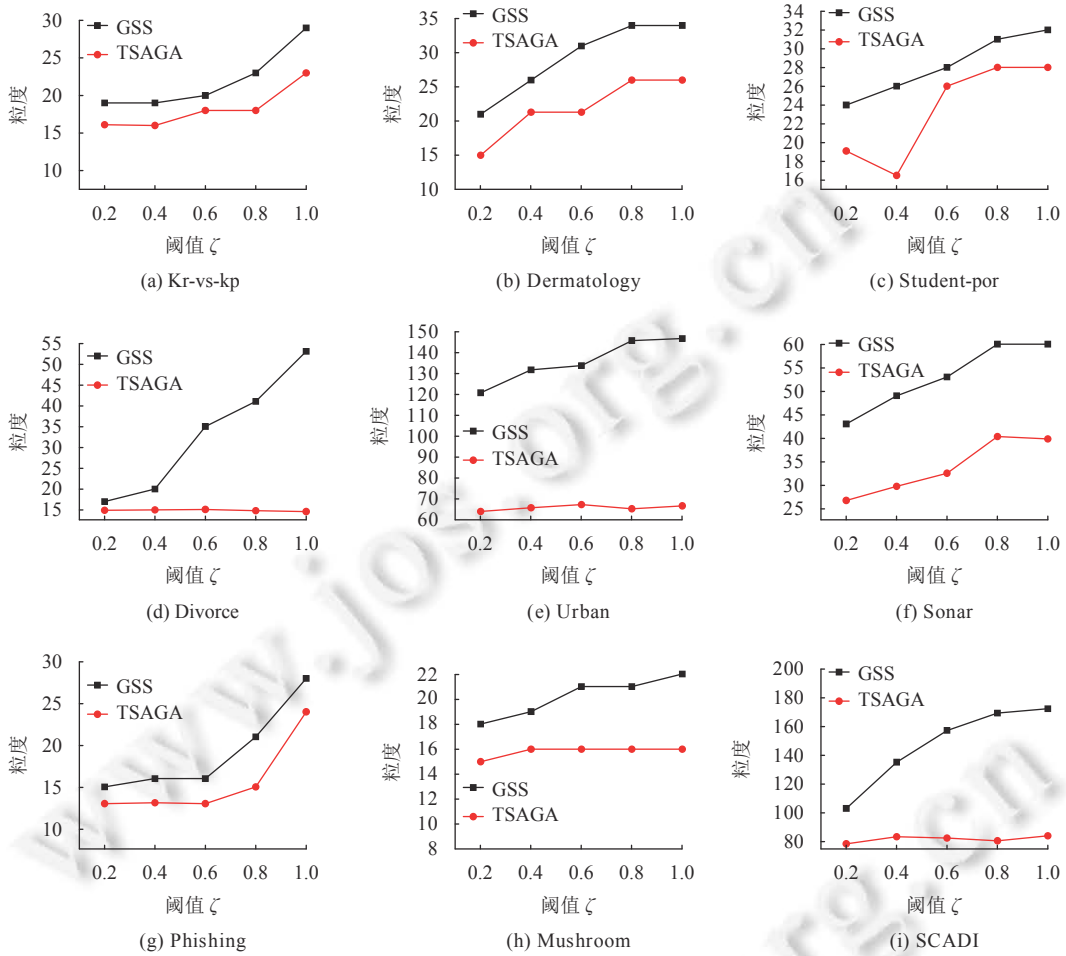

图 4 TSAGA 和 GSS 在不同阈值下所选问题求解层的粒度

在第 3 组实验中, 依然以 GSS 方法选出的满意问题求解层的分类精度为标准, 设置阈值 ζ 分别取 0.2、0.4、0.6、0.8、1.0, 在每个阈值下均进行 10 次实验, 取 10 次实验结果的平均值作为最终结果. 消融实验结果如表 6 所示. 为了使结果更加直观, 我们将表 6 中的实验数据以图的形式展现出来, 如图 5 所示. 图 5 中横坐标为阈值 ζ, 纵坐标为问题求解层的粒度.

表 6 消融实验结果

数据集	最细粒度	实验方法	ζ				
			0.2	0.4	0.6	0.8	1.0
Kr-vs-kp	36	TSAGA不采用自适应选择算子	16.5	16.4	18.0	18.0	23.0
		TSAGA不采用自适应交叉算子	16.2	16.4	18.0	18.0	23.0
		TSAGA不采用自适应大变异算子	16.9	17.7	18.2	18.2	23.2
		TSAGA不采用两阶段	16.5	16.4	18.0	18.0	23.0
		TSAGA	16.1	16.0	18.0	18.0	23.0

表 6 消融实验结果 (续)

数据集	最细粒度	实验方法	ζ				
			0.2	0.4	0.6	0.8	1.0
Dermatology	34	TSAGA不采用自适应选择算子	15.3	21.2	21.5	26.0	26.0
		TSAGA不采用自适应交叉算子	15.4	21.3	21.7	26.0	26.0
		TSAGA不采用自适应大变异算子	15.9	21.3	21.5	26.0	26.2
		TSAGA不采用两阶段	15.0	21.3	21.7	26.0	26.0
		TSAGA	15.0	21.3	21.3	26.0	26.0
Student-por	32	TSAGA不采用自适应选择算子	19.4	16.6	26.0	28.0	28.0
		TSAGA不采用自适应交叉算子	19.9	16.5	26.0	28.0	28.0
		TSAGA不采用自适应大变异算子	19.4	17.0	26.2	28.3	28.3
		TSAGA不采用两阶段	20.2	16.6	26.3	28.0	28.0
		TSAGA	19.1	16.5	26.0	28.0	28.0
Divorce	54	TSAGA不采用自适应选择算子	14.9	15.1	15.1	15.5	15.4
		TSAGA不采用自适应交叉算子	15.8	15.0	15.1	15.5	15.7
		TSAGA不采用自适应大变异算子	15.8	16.7	16.5	15.9	16.2
		TSAGA不采用两阶段	15.2	15.1	15.2	14.8	15.1
		TSAGA	14.9	15.0	15.1	14.8	14.6
Urban	147	TSAGA不采用自适应选择算子	64.2	66.3	69.2	65.9	67.2
		TSAGA不采用自适应交叉算子	66.2	67.9	69.6	68.1	67.7
		TSAGA不采用自适应大变异算子	68.8	69.2	70.1	70.5	72.5
		TSAGA不采用两阶段	64.4	67.8	68.5	68.7	68.4
		TSAGA	64.1	65.9	67.4	65.4	66.8
Sonar	60	TSAGA不采用自适应选择算子	27.0	30.1	32.5	40.3	40.6
		TSAGA不采用自适应交叉算子	26.9	30.4	32.6	42.6	41.0
		TSAGA不采用自适应大变异算子	28.3	30.6	33.1	43	43.8
		TSAGA不采用两阶段	26.8	29.8	32.6	40.5	40.3
		TSAGA	26.7	29.7	32.5	40.3	39.8
Phishing	30	TSAGA不采用自适应选择算子	13.0	13.1	13.0	15.0	24.0
		TSAGA不采用自适应交叉算子	13.1	13.1	13.0	15.0	24.0
		TSAGA不采用自适应大变异算子	13.0	13.1	13.1	15.0	24.0
		TSAGA不采用两阶段	13.0	13.1	13.0	15.0	24.0
		TSAGA	13.0	13.1	13.0	15.0	24.0
Mushroom	22	TSAGA不采用自适应选择算子	15.0	16.0	16.0	16.0	16.0
		TSAGA不采用自适应交叉算子	15.0	16.0	16.0	16.0	16.0
		TSAGA不采用自适应大变异算子	15.0	16.0	16.0	16.0	16.0
		TSAGA不采用两阶段	15.0	16.0	16.0	16.0	16.0
		TSAGA	15.0	16.0	16.0	16.0	16.0
SCADI	205	TSAGA不采用自适应选择算子	80.1	85.6	82.7	87.8	85.3
		TSAGA不采用自适应交叉算子	86.1	92.6	90.9	89.1	89.7
		TSAGA不采用自适应大变异算子	85.1	94.0	89.2	87.4	90.5
		TSAGA不采用两阶段	78.5	83.5	86.4	83.4	84.0
		TSAGA	78.5	83.4	82.5	80.6	84.0

由表 6 和图 5 可以看出, 当划分序乘积空间较小时, 两阶段自适应遗传算法的各个改进部分对算法性能的影响区别不大, 如对于数据集 Dermatology、Student-por、Phishing 和 Mushroom. 但是随着划分序乘积空间的增大, 两阶段自适应遗传算法的各个改进部分对算法性能的影响区别越来越大, 如对于数据集 Kr-vs-kp、Divorce、Urban、Sonar 和 SCADI. 很容易看出自适应大变异算子对算法性能的影响程度最大, 自适应交叉算子对算法性能

的影响程度次之, 两阶段划分和自适应选择算子对算法性能的影响程度相对较小.

通过上述实验结果, 说明了所提两阶段自适应遗传算法在选择问题求解层时的有效性.

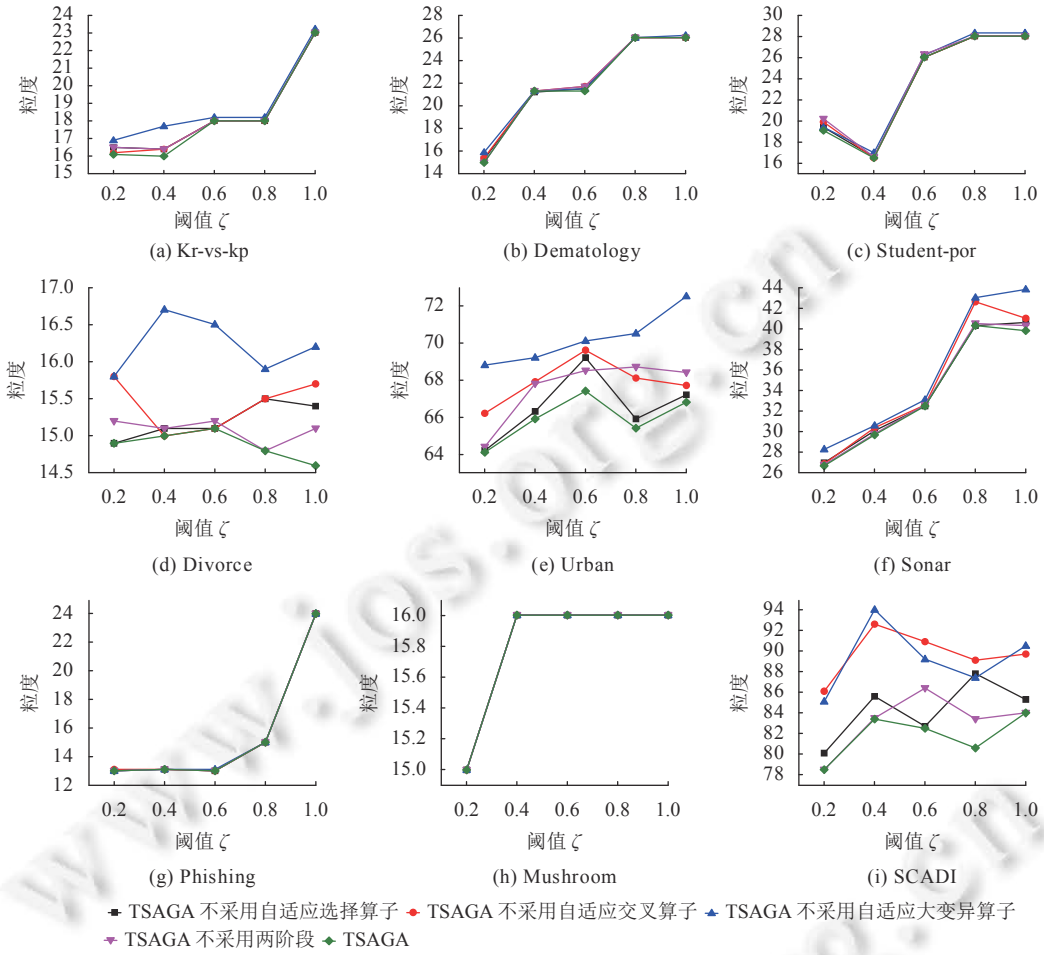

图 5 消融实验结果

4 结论及未来展望

划分序乘积空间是一种新的粒计算模型, 提供了对问题多视角和多层次的描述, 从而可以从多视角和多层次进行问题求解, 符合人类的认知规律. 其解空间是由多个问题求解层构成的格结构, 如何在划分序乘积空间中找到合适的问题求解层进行问题求解是一个重要且值得研究的问题. 本文提出一种两阶段自适应遗传算法在划分序乘积空间中选择问题求解层. 算法第 1 阶段基于经典遗传算法, 预选出一些优秀问题求解层作为第 2 阶段初始种群的一部分, 从而优化解空间. 算法第 2 阶段, 提出随当前种群进化迭代次数动态变化的自适应选择算子、自适应交叉算子以及自适应大变异算子, 从而在优化的解空间中进一步选择问题求解层. 通过实验证明了本文所提方法能在保证分类精度的前提下选出较粗的问题求解层.

未来的工作可以从以下几个方面展开: (1) 进一步研究以商空间为代表的粒计算理论中有效的粒化方法, 构造不同的划分序乘积空间, 研究不同划分序乘积空间对问题求解层选择算法性能的影响. (2) 针对构造划分序乘积空间时视角和层次可能存在冗余的问题, 进一步研究以粗糙集为代表的粒计算理论中有效的约简方法, 对划分序乘积空间进行约简, 在此基础上, 研究更高效的问题求解层选择算法. (3) 针对实际应用中广泛存在的多模态数据, 如智慧医疗和智慧物流领域, 构建多模态数据的划分序乘积空间, 利用选择出的问题求解层对多模态数据进行多视

角多层次的分析和处理, 研究划分序乘积空间的实际应用.

References:

[1] Yao YY. A triarchic theory of granular computing. Granular Computing, 2016, 1(2): 145–157. [doi: 10.1007/s41066-015-0011-0]

[2] Pedrycz W, Skowron A, Kreinovich V. Handbook of Granular Computing. West Sussex: John Wiley & Sons, 2008. [doi: 10.1002/9780470724163]

[3] Qian YH, Liang JY, Yao YY, Dang CY. MGRS: A multi-granulation rough set. Information Sciences, 2010, 180(6): 949–970. [doi: 10.1016/j.ins.2009.11.023]

[4] Qian YH, Liang JY, Dang CY. Incomplete multigranulation rough set. IEEE Trans. on Systems, Man, and Cybernetics—Part A: Systems and Humans, 2010, 40(2): 420–431. [doi: 10.1109/TSMCA.2009.2035436]

[5] Qian YH, Zhang H, Sang YL, Liang JY. Multigranulation decision-theoretic rough sets. Int'l Journal of Approximate Reasoning, 2014, 55(1): 225–237. [doi: 10.1016/j.ijar.2013.03.004]

[6] Khan MA, Banerjee M. Formal reasoning with rough sets in multiple-source approximation systems. Int'l Journal of Approximate Reasoning, 2008, 49(2): 466–477. [doi: 10.1016/j.ijar.2008.04.005]

[7] Sang BB, Guo YT, Shi DR, Xu WH. Decision-theoretic rough set model of multi-source decision systems. Int'l Journal of Machine Learning and Cybernetics, 2018, 9(11): 1941–1954. [doi: 10.1007/s13042-017-0729-x]

[8] Chen YH, Yao YY. A multiview approach for intelligent data analysis based on data operators. Information Sciences, 2008, 178(1): 1–20. [doi: 10.1016/j.ins.2007.08.011]

[9] Yao YY. Information granulation and rough set approximation. Int'l Journal of Intelligent Systems, 2001, 16(1): 87–104. [doi: 10.1002/1098-111X(200101)16:1<87::AID-INT7>3.0.CO;2-S]

[10] Hong TP, Lin CE, Lin JH, Wang SL. Learning cross-level certain and possible rules by rough sets. Expert Systems with Applications, 2008, 34(3): 1698–1706. [doi: 10.1016/j.eswa.2007.01.038]

[11] Feng QR, Miao DQ, Cheng Y. Hierarchical decision rules mining. Expert Systems with Applications, 2010, 37(3): 2081–2091. [doi: 10.1016/j.eswa.2009.06.065]

[12] Wu WZ, Leung Y. Theory and applications of granular labelled partitions in multi-scale decision tables. Information Sciences, 2011, 181(18): 3878–3897. [doi: 10.1016/j.ins.2011.04.047]

[13] Wu WZ, Zhuang YB, Tan AH, Xu YH. Scale combinations in inconsistent generalized multi-scale decision systems. Pattern Recognition and Artificial Intelligence, 2018, 31(6): 485–494 (in Chinese with English abstract). [doi: 10.16451/j.cnki.issn1003-6059.201806001]

[14] Zheng JW, Wu WZ, Bao H, Tan AH. Evidence theory based optimal scale selection for multi-scale ordered decision systems. Int'l Journal of Machine Learning and Cybernetics, 2022, 13(4): 1115–1129. [doi: 10.1007/s13042-021-01438-x]

[15] Wu WZ, Qian YH, Li TJ, Gu SM. On rule acquisition in incomplete multi-scale decision tables. Information Sciences, 2017, 378: 282–302. [doi: 10.1016/j.ins.2016.03.041]

[16] Bao H, Wu WZ, Zheng JW, Li TJ. Entropy based optimal scale combination selection for generalized multi-scale information tables. Int'l Journal of Machine Learning and Cybernetics, 2021, 12(5): 1427–1437. [doi: 10.1007/s13042-020-01243-y]

[17] Niu DR, Wu WZ, Li TJ. Variable precision based optimal scale combinations in generalized multi-scale decision systems. Pattern Recognition and Artificial Intelligence, 2019, 32(11): 965–974 (in Chinese with English abstract). [doi: 10.16451/j.cnki.issn1003-6059.201911001]

[18] Wu WZ, Leung Y. A comparison study of optimal scale combination selection in generalized multi-scale decision tables. Int'l Journal of Machine Learning and Cybernetics, 2020, 11(5): 961–972. [doi: 10.1007/s13042-019-00954-1]

[19] Wu WZ, Sun Y, Wang X, Zheng JW. Local optimal scale combination selections in inconsistent generalized multi-scale decision systems. Pattern Recognition and Artificial Intelligence, 2021, 34(8): 689–700 (in Chinese with English abstract). [doi: 10.16451/j.cnki.issn1003-6059.202108002]

[20] Deng J, Zhan JM, Wu WZ. A three-way decision methodology to multi-attribute decision-making in multi-scale decision information systems. Information Sciences, 2021, 568: 175–198. [doi: 10.1016/j.ins.2021.03.058]

[21] Liu MM, Zhao SL, Han YH, Su DH, Li XC, Chen M. Research on multi-scale data mining method. Ruan Jian Xue Bao/Journal of Software, 2016, 27(12): 3030–3050 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4924.htm [doi: 10.13328/j.cnki.jos.004924]

[22] Li F, Hu BQ, Wang J. Stepwise optimal scale selection for multi-scale decision tables via attribute significance. Knowledge-based Systems, 2017, 129: 4–16. [doi: 10.1016/j.knosys.2017.04.005]

[23] Xu Y, Yao YY. Partition order product space: Partition based granular computing model. Journal of Computer Research and Development, 2019, 56(4): 836–843 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2019.20180325]

[24] Yang X, Li TR. , Liu D, Chen HM, Luo C. A unified framework of dynamic three-way probabilistic rough sets. Information Sciences, 2017, 420: 126–147. [doi: 10.1016/j.ins.2017.08.053]

[25] Catanzariti F, Chiaselotti G, Infusino FG, Marino G. Object similarity measures and Pawlak's indiscernibility on decision tables. Information

Sciences, 2020, 539: 104–135. [doi: 10.1016/j.ins.2020.05.030]
[26] Chen H, Li JH, Min F, Liu WQ, Tsang ECC. Optimal scale selection in dynamic multi-scale decision tables based on sequential three-way decisions. Information Sciences, 2017, 415–416: 213–232. [doi: 10.1016/j.ins.2017.06.032]
[27] Liu FL, Zhang BW, Ciucci D, Wu WZ, Min F. A comparison study of similarity measures for covering-based neighborhood classifiers. Information Sciences, 2018, 448–449: 1–17. [doi: 10.1016/j.ins.2018.03.030]
[28] Wang HZ, He CQ, Li ZP. A new ensemble feature selection approach based on genetic algorithm. Soft Computing, 2020, 24(20): 15811–15820. [doi: 10.1007/s00500-020-04911-x]
[29] Li SJ, Zhang KX, Chen QR, Wang SQ, Zhang SQ. Feature selection for high dimensional data using weighted K-nearest neighbors and genetic algorithm. IEEE Access, 2020, 8: 139512–139528. [doi: 10.1109/ACCESS.2020.3012768]
[30] Mathias HD, Foley SS. A parallel two-stage genetic algorithm for route planning. In: Proc. of the 2020 Genetic and Evolutionary Computation Conf. Companion. Cancún: ACM, 2020. 1739–1746. [doi: 10.1145/3377929.3398116]
[31] Xu Y, Li BF. Multiview sequential three-way decisions based on partition order product space. Information Sciences, 2022, 600: 401–430. [doi: 10.1016/j.ins.2022.04.007]
[32] Garg H. A hybrid GSA-GA algorithm for constrained optimization problems. Information Sciences, 2019, 478: 499–523. [doi: 10.1016/j.ins.2018.11.041]
[33] An LP, Liu S. Two-phase genetic algorithm for attributes reduction. Systems Engineering-theory & Practice, 2014, 34(11): 2892–2899 (in Chinese with English abstract).
[34] Xue Y, Zhu HK, Liang JY, Słowik A. Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification. Knowledge-based Systems, 2021, 227: 107218. [doi: 10.1016/j.knosys.2021.107218]
[35] Li SQ, Sun X, Sun DH, Bian WP. Summary of crossover operator of genetic algorithm. Computer Engineering and Applications, 2012, 48(1): 36–39 (in Chinese with English abstract). [doi: 10.3778/j.issn.1002-8331.2012.01.011]
[36] Chen ZQ, Huang ZY, Sun MW, Sun QL. Active disturbance rejection control of load frequency based on big probability variation's genetic algorithm for parameter optimization. CAAI Trans. on Intelligent Systems, 2020, 15(1): 41–49 (in Chinese with English abstract). [doi: 10.11992/tis.201906026]
[37] Dua D, Graff C. UCI machine learning repository. 2019. http://archive.ics.uci.edu/ml
[38] Kaggle. Your machine learning and data science community. 2022. https://www.kaggle.com/datasets
[39] Xie JJ, Hu BQ, Jiang HB. A novel method to attribute reduction based on weighted neighborhood probabilistic rough sets. Int'l Journal of Approximate Reasoning, 2022, 144: 1–17. [doi: 10.1016/j.ijar.2022.01.010]

附中文参考文献:

[13] 吴伟志, 庄宇斌, 谭安辉, 徐优红. 不协调广义多尺度决策系统的尺度组合. 模式识别与人工智能, 2018, 31(6): 485–494. [doi: 10.16451/j.cnki.issn1003-6059.201806001]
[17] 牛东苒, 吴伟志, 李同军. 广义多尺度决策系统中基于可变精度的最优尺度组合. 模式识别与人工智能, 2019, 32(11): 965–974. [doi: 10.16451/j.cnki.issn1003-6059.201911001]
[19] 吴伟志, 孙钰, 王霞, 郑嘉文. 不协调广义多尺度决策系统的局部最优尺度组合选择. 模式识别与人工智能, 2021, 34(8): 689–700. [doi: 10.16451/j.cnki.issn1003-6059.202108002]
[21] 柳萌萌, 赵书良, 韩玉辉, 苏东海, 李晓超, 陈敏. 多尺度数据挖掘方法. 软件学报, 2016, 27(12): 3030–3050. http://www.jos.org.cn/1000-9825/4924.htm [doi: 10.13328/j.cnki.jos.004924]
[23] 徐怡, 姚一豫. 划分序乘积空间: 基于划分的粒计算模型. 计算机研究与发展, 2019, 56(4): 836–843. [doi: 10.7544/issn1000-1239.2019.20180325]
[33] 安利平, 刘森. 属性约简的两阶段遗传算法. 系统工程理论与实践, 2014, 34(11): 2892–2899.
[35] 李书全, 孙雪, 孙德辉, 边伟朋. 遗传算法中的交叉算子的述评. 计算机工程与应用, 2012, 48(1): 36–39. [doi: 10.3778/j.issn.1002-8331.2012.01.011]
[36] 陈增强, 黄朝阳, 孙明玮, 孙青林. 基于大变异遗传算法进行参数优化整定的负荷频率自抗扰控制. 智能系统学报, 2020, 15(1): 41–49. [doi: 10.11992/tis.201906026]

徐怡(1981－), 女, 博士, 教授, 博士生导师, CCF高级会员, 主要研究领域为智能信息处理, 粒计算, 边缘计算.

邱紫恒(1998－), 男, 硕士, 主要研究领域为粒计算.