学术论文

基于候选方案排序的进化决策方法

荔建琦，陈火旺，王兵山
（国防科技大学计算机系，长沙 410073）

[摘要] 由于无法得到准确的期望效用函数，在信息不完全和结果不确定的环境下作出决策是困难的。提出基于候选方案排序的进化决策方法。通常通过分析得出一组与候选方案期望效用相关的指标，设计决策规则归结为寻找二者之间的相关关系。如果将所有候选方案按其对效用有影响的指标分为 n 类，并利用进化算法在 $n!$ 空间中搜索全部方案的期望效用排序，则根据此排序作出最佳决策。提出针对排序问题的遗传算法。该方法较少依赖专家知识，无须显式地构造期望效用函数，能有效处理非数值或非量化指标以及指标冲突和指标相关等问题，在带随机噪声环境下仍能获得稳健解。在仿真机器人控制器设计中的应用表明了该方法的有效性。

[关键词] 进化决策；进化机器人；排序问题遗传算法

1 引言

在实践中，常常需要在信息不完全和确定性因果模型难以构造或不存在的情况下作出决策。研究在结果不确定情况下获得最佳收益的决策方法。

传统决策分析方法的关键在于确定各种决策下后果的先验分布和效用值[1]，当决策动作与结果的因果关系不明晰或者由于多目标等因素导致效用函数难以通过分析构造时，传统方法难以适用。

进化算法由于具有较少依赖领域知识、适用面广、稳健性和全局搜索性好、易于并行等优点，在工程优化、机器学习等领域取得了大量成功的应用，被认为是未来人工智能发展最有生机与活力的领域之一。进化算法中最有代表性的是遗传算法[2]、进化策略[3]和进化规划[4]等。

针对随机性环境下有限个方案的决策，提出基于候选方案排序的进化决策方法（Evolutionary Decision Making Based on Candidates Ranking），将决策规则学习归结为排序问题，并用遗传算法进行求解。以仿真机器人控制器设计为例，介绍这种基于进化排序的决策规则学习方法。该应用实例将子行动规划按目标位置的局部环境分类，以所有可能情况的排序作为决策规则，并用遗传算法自适应学习该排序规则。通过对比试验确定了一组性能较好的遗传算子。

与传统方法相比，该方法无须知道决策后果的先验分布；可以处理收益无法立即观测的情况；所确立的优先关系为全序，不额外引入人为约束；采用进化算法自适应地构造最优规则，对专家知识依赖较少；可以有效处理方案未量化的情况。

2 基于候选方案排序的决策模型

基于排序的决策模型（DMBCR）对下面两类问题有效。**a.** 单次决策问题：候选方案总数有限，候选方案的优度存在全序关系，最佳决策通过在候选方案中选择最优者作出。**b.** 有限周期序贯决策问题：每一决策点有多个可行的候选方案，决策序列的优度主要取决于各候选方案在决策序列中的位置，决策序列的相对优劣可以相互比较。

以上两类问题的决策规则都可以表示为候选方

[收稿日期] 2000－02－26；修回日期 2000－05－20
[基金项目] 国家自然科学基金资助项目（69783007；69903010）
[作者简介] 荔建琦（1972－），男，陕西西安市人，国防科技大学博士研究生

案的排序。DMBCR 包含两个要素：**a**. 待排序的有穷候选方案集 C；**b**. 对候选方案排序的评估函数 E。通常以评估函数 E 作为适应值函数，当 E 反映决策者的主观偏好时，相当于以期望效用函数作为适应值函数。

应用 DMBCR 求解决策问题的一般步骤：

1) *确定待排序的候选方案*　对候选方案为无限集或数量过大的情况，将候选方案按特定指标分类，并以类作为候选方案。分类遵循两个准则：**a**. 分类指标选择恰当，所依据的分类指标与方案的重要性/执行顺序相关，将重要性/执行顺序相近的方案归于一类；**b**. 分类粒度适度，同类方案其重要性/执行顺序差别不应过大，不同类方案其差别不应过小。

2) *确定候选方案排序的性能评估函数 E*　通常 E 为决策规则的期望损益函数。

3) *确定决策问题候选方案（类）的最佳排序*　采用排序遗传算法自适应地学习该排序。

采用进化学习算法的 DMBCR 称为基于候选方案排序的进化决策。在决策点，基于 DMBCR 的决策过程为：**a**. 计算当前可行方案及相关属性，构成当前候选方案集；**b**. 根据完整方案排序为当前候选方案排序；**c**. 选择最高优先级方案执行。

3　仿真机器人控制器设计

进化机器人研究进化计算在机器人设计中的应用[5]，为复杂问题求解提供一种强有力的方法支持，又为探索进化计算的现实世界复杂问题求解能力的极限提供广阔的空间。其研究内容包括：机器人躯体设计、机器人规划、进化硬件、机器人结构与控制的联合设计、传感器设计与融合、机器人通信、多 agent 协同与合作设计等。

99′进化计算会议（CEC′99）机器人设计竞赛对推积木机器人控制器设计提出的要求[6]：设计的仿真机器人控制器能够在规定时间内、在初始地形未知的正方形网格积木世界中执行积木块收集任务，机器人性能用某种积木块排列的紧凑性得分衡量。我们将机器人任务分解为地形勘察和木块收集两个独立子任务，分别设计相应控制器。其中：前者以勘察完成时间最短为目标并且不改变地形，后者以得分最多为目标。勘察控制器的进化设计已另文论述[7,8]，用基于候选方案排序的进化决策方法设计积木收集机器人控制器是本文要介绍的内容。

3.1　积木世界

积木世界用 10×10 矩阵描述，中心部分的 8×8 区域为活动区域，可以摆放积木和机器人，此外的周缘部分为墙壁。矩阵的一个元素称位置、网格或块，元素的值称为状态。全体积木世界的集合定义为 $BW=\{\langle S, CF, h\rangle\}$，其中：

1) $S=\{\text{Empt, Box, Wall, Rob}\}$ 为网格状态集，其中元素为状态常量。Empt 为当前位置空，Box 为放一块积木，Wall 为墙壁，Rob 为机器人。

2) 令 $P=\{0,\cdots,9\}$，函数 CF：$P^2\to S$（称格局）满足下列条件：**a**. $CF(x, y)=\text{Wall}$ 且仅当 $x\in\{0, 9\}$ 或 $y\in\{0, 9\}$；**b**. $\#(CF^{-1}(\{\text{Box}\}))=9$；**c**. $\#(CF^{-1}(\{\text{Rob}\}))=1$。其中 $\#(S)$ 表示集合 S 中的元素个数。

3) $h\in\{0, 1, 2, 3\}$ 代表机器人当前面向的方向。当把 CF 表示为按行排列的矩阵时，0、1、2、3 即分别代表上、右、下、左四个方向。

定义　块 $(x, y)\in P$ 的正交邻块集合为 $Nb(x, y)=\{(x', y')\mid|x'-x|+|y'-y|=1, (x', y')\in P^2\}$。$bw=\langle S, ICF, h\rangle$ 称为初始化积木世界，如果 ICF 还满足：若 $ICF(x, y)=\text{Box}$，则 $ICF(Nb(x, y))\cap\{\text{Wall, Box}\}=\varnothing$。即任何积木都不与其他积木或墙块正交邻接。图 1 是一个初始化积木世界的例子。左侧为 CF 的矩阵表示，机器人位于网格 (5, 5)，方向 h 为 3。称由 CF 确定的积木分布情况为地形或格局。

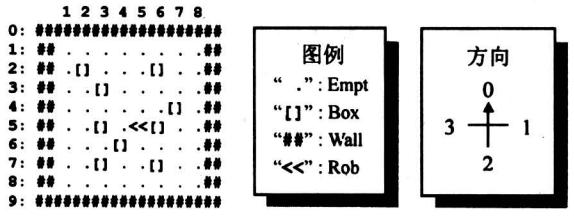


图 1　积木世界示例

Fig.1　An instance of boxes world

3.2　机器人

机器人只能在积木世界内部的 8×8 区域活动。它通过传感器感知周围地理环境信息，可在水平和垂直方向移动，改变积木世界格局。积木世界采用离散时间，一个时间步机器人最多执行一个动作。

1) *传感器*　传感器为一长度为 8 的向量，元素取值域为 {Empt, Box, Wall}。以机器人当前位置和向前方向为基准，返回上、下、左、右、和左

上、左下、右上、右下邻接块的状态，其布局方式见图2。

7	0	1
6		2
5	4	3

机器人位于中心，方向向上

图2 传感器布局

Fig.2 The configuration of sensors

2) *动作* 机器人每个时间步可以执行一个动作指令。动作指令集为 $Action = \{\text{Tlf}, \text{Trt}, \text{Fwd}\}$。Tlf 和 Trt 分别代表左转、右转，其执行只改变 h 为相应值。Fwd 代表前进一格：**a.** 当正前方为 Empt 时，Fwd 动作将机器人位置向前移动一格，方向 h 不变；**b.** 当正前方第一格为 Box 且正前方第二格为 Empt 时，Fwd 动作将正前方一格状态改变为 Rob，正前方第二格改变为 Box，方向 h 不变。此外，Fwd 指令不产生任何效果。

3.3 问题描述

目标（集木任务）：设计控制器，使机器人从任意地形及出发位置已知的初始化积木世界开始执行，在给定时间步内，将所有积木排列为尽可能紧凑的形式。由于总执行时间为150步，集木任务可用时间等于勘察完成后剩余时间，现有勘察控制器的子任务完成时间服从正态分布 $N(29, 6.0^2)$，故可用时间步定为 $Steps = 121 + \varepsilon$，其中 ε 为服从 $N(0, 6.0^2)$ 的随机变量（取整）。积木排列的紧凑性用 $SCORE: BW \to \mathbf{N}$ 来度量，其中 **N** 为自然数集，设 $bw = \langle S, CF, h \rangle \in BW$，定义

$$SCORE(bw) = \sum_{(x,y)\in P^2, CF(x,y)=\text{Box}} LS(x,y) \quad (1)$$

其中 $LS: P^2 \to \{0, \cdots, 4\}$ 定义为

$$LS(x,y) = \begin{cases} 0 & \text{if } x \in \{0,9\} \text{ or } y \in \{0,9\} \\ \#\{(x',y') \mid (x',y') \in Nb(x,y) \\ \quad \text{and } CF(x',y') \in \{\text{Box}, \text{Wall}\}\} \\ \text{otherwise} \end{cases} \quad (2)$$

$SCORE(bw)$ 称为积木世界 bw 的紧凑性得分。当积木在墙角排为 3×3 方阵时，得最高分30，但在有时间约束的情况下这未必是最佳排列。对机器人控制器优良性的评测，用统计其在样本初始化积木世界集合中运行所得到的平均分来衡量。

4 控制器模型与进化决策

机器人的任务可以分解为反复将某积木推至某位置，控制器的核心职能为序贯地选择下一个积木/目标位置子任务，并给出该子任务的路径规划。针对此类问题提出了规划/选择控制模型（Planning and Selecting Control Model）[7]，参见图3。

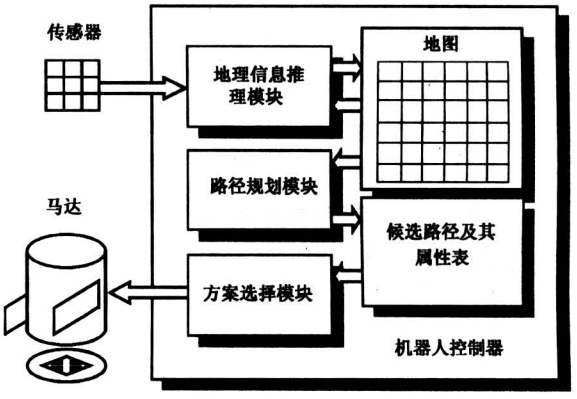


图3 PSCM 控制器模型

Fig.3 PSCM controller model

对地形已知的集木问题，除图3中传感器、地理信息推理模块和执行移动指令的马达外，其余各模块功能如下：

1) *路径规划* 根据地图，计算机器人从当前位置开始，将每个积木推到每个“有价值位置”的所有最短路径。如果 $LS(x, y) > 0$，(x, y) 是有价值位置。

2) *候选路径及其属性表* 列出所有有价值位置及其相关属性信息，用于目标分类和选择。

3) *方案选择* 根据候选目标属性及决策规则选择最佳目标，并向马达发出相应执行指令。

从 a 位置出发将 b 位置的积木推至 c 位置，其单次推积木子任务的最优路径规划可以通过确定性算法获得，因此关键在于如何确定积木最佳排列方式及实现该排列的行动次序。

该设计问题的难点在于，在初始世界不确定且有时间约束的情况下，缺乏有效且高效的确定性算法来对积木最优排列和实现该排列的最优步骤实施规划。对于特定形状和参数的积木世界，积木排列的最佳方式可以通过推理和计算得到。然而这种方法的脆弱性是明显的：**a.** 当时间约束使最优排列无法完成时，确定实际的最优规划非常困难，需要

设计者做出复杂的专家推理，且仍无法保证其最优性；**b**. 对仅仅是参数不同的问题，如世界大小和积木数目不同时，需要重新计算。

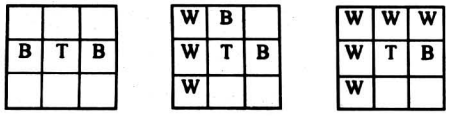
用基于排序的进化决策方法实现该控制器的决策模块。积木排列的过程，可以认为是顺次选择不同类型目标和相应被移动积木的过程。每次目标位置选定后，可选择与该目标实际距离最短的积木作为被移动积木。如果将选择的时间次序视为对不同类型目标的一种重要性排序，则目标选择就归结为：**a**. 将目标根据某种属性分类；**b**. 将不同类型的目标排序。任何积木排列方式都是一种地理形态，因此一种直观的分类方法是根据目标位置的局部地理信息分类，并根据各局部分类的排序作出决策选择。称这种方法为基于局部格局排序的选择方法（Local Case Ranking Based Selection）。要使这种方法有效，目标分类必须满足一定条件，即目标类型差别应反映目标在最优排列中次序的差别，且同类目标应在最优排列中位于同等或相近的次序。

根据 *SCORE* 函数的计算方法，积木排列的得分是该排列中各积木得分之和，而单个积木的得分与其正交相邻积木和墙块的数目有关。因此，提出直接根据目标位置的相邻块中积木和墙壁数目对行动方案分类。分类结果见表 1，部分局部格局类型示例见图 4。还可考虑按局部环境的分布方式细化分类，但排序的复杂度将明显增加。

表 1　局部格局分类

Table 1　Classification of local cases

类型	0	1	2	3	4	5	6	7	8	9
相邻墙块数（N_W）	0	0	0	0	1	1	1	2	2	$N_B+N_W=4$
相邻积木数（N_B）	0	1	2	3	0	1	2	0	1	
得分 LS（x，y）	0	2	4	6	1	3	5	2	4	



类型 2　　类型 6　　类型 8

图 4　局部格局类型示例

Fig.4　Examples of local cases

上述分类中第 9 类对应的中心位置不能作为积木块的目标位置，而第 0 类目标位置不得分，因此这两类的优先级最低。机器人决策模块的核心是确定各个局部格局的优先级排序，直接的表示方法是给各类格局赋予 {1，…，8} 中一个不同的整数，数值越大的优先级越高。机器人在每个决策点的决策过程包括以下三个步骤：**a**. 计算所有当前可行行动路径作为候选行动方案；**b**. 根据类型排序，按行动方案所属类型给各候选方案赋予优先级；**c**. 选择最高优先级方案执行。当有多个方案同时具有最高优先级时，选择耗时最短的方案执行。

5　进化排序算法

采用排序问题遗传算法 OPGA 自适应地学习最优排序，算法的基本框架如下。

1）种群初始化 *Initialize* (*Pop* (0))；

2）初始种群适应值计算 *FitnessEvaluate* (*Pop* (0))；

3）$t=0$；

4）进化下一代：

4.1）如果满足终止条件转 6；

4.2）选择待繁殖个体 *Pop′* (= *Select* (*Pop* (*t*))；

4.3）遗传操作，生成下一代种群 *Pop* (*t*+1) = *Genetic-Operation* (*Pop′*)；

4. 4）适应值计算 *FitnessEvaluate* (*Pop* (*t*+1))；

4. 5）$t=t+1$；

5）转 4；

6）结束。

用 OPGA 进行问题求解必须对几个基本环节作出设计抉择。

5.1　编码方式与遗传操作

排序问题解的形式为候选方案编号的全排列。第 0 类和第 9 类优先级最低，编码时将它们排除，因此解为 {1，…，8} 的全排列。

组合优化问题的遗传操作与编码方式密切相关，针对排序问题的特点设计专用遗传算法。本文实现了三种编码方式：排序编码、序数编码和隐序编码。其中后两种为本文提出的。下面介绍这几种编码方式及其相应的遗传算子，并用正整数编号代表待排序对象，简称对象，排列长度设为 n。

1）*排序编码* OE　即直接采用待排序对象的排序作为解的染色体表示，这相当于 TSP 的路径表示。对这种编码必须设计相应交叉算子以保证其封闭性，已提出的交叉算子有部分映射杂交（PMX），次序杂交（OX），循环杂交（CX），基于位置杂交（PX），基于次序交叉（OBX）等[9]。

双端对齐交叉（ACX）。设父本为 $V^1=(v_1^1, v_2^1, \cdots, v_n^1)$ 和 $V^2=(v_1^2, v_2^2, \cdots, v_n^2)$，$j$、$k$ 满足 $v_i^1=v_i^2$ ($i=1, \cdots, j-1, k+1, \cdots, n$)，$v_j^1 \neq v_j^2$，$v_k^1 \neq v_k^2$。设 $v_{j_2}^1=v_j^2$，$v_{j_1}^2=v_j^1$，$v_{k_2}^1=v_k^2$，$v_{k_1}^2=v_k^1$，定义：

$ACX_l(V^1, V^2) = \langle (v_1^1, \cdots, v_{j-1}^1, v_j^2, v_j^1, \cdots, v_{j_2-1}^1, v_{j_2+1}^1, \cdots, v_n^1), (v_1^2, \cdots, v_{j-1}^2, v_j^1, v_j^2, \cdots, v_{j_1-1}^2, v_{j_1+1}^2, \cdots, v_n^2)\rangle$。

$ACX_r(V^1, V^2) = \langle (v_1^1, \cdots, v_{k_2-1}^1, v_{k_2+1}^1, \cdots, v_k^1, v_k^2, v_{k+1}^1, \cdots, v_n^1), (v_1^2, \cdots, v_{k_1-1}^2, v_{k_1+1}^2, \cdots, v_k^2, v_k^1, v_{k+1}^2, \cdots, v_n^2)\rangle$。

双端对齐交叉 *ACX* 每次以相等概率随机选择 ACX_l 和 ACX_r 之一执行。

此外，分别实现了倒位、插入、互换[9]和相邻互换变异算子。倒位算子随机选择一对位置参数 $(x, y) \in \{1, \cdots, n\}^2$，将位于 x 位置和 y 位置间的连续子串（含 x、y 位置）逆转后放回原位置。如 (2, 8, 5, 3, 4, 6, 7, 1) 以 (4, 7) 为参数做倒位，得到 (2, 8, 5, 7, 6, 4, 3, 1)。插入算子随机选择一对原位置和目标置 $(x, y) \in \{1, \cdots, n\} \times \{1, \cdots, n+1\}$，将原位置中的对象插入目标位置对象之前，其余对象相对次序不变，其中目标位置为 $n+1$ 时插入排列尾。如 (2, 8, 5, 3, 4, 6, 7, 1) 以 (2, 6) 为参数做插入，得到 (2, 5, 3, 4, 8, 6, 7, 1)。互换随机选择一对位置 $(x, y) \in \{1, \cdots, n\}^2$，将该两位置中的元素互换。如 (2, 8, 5, 3, 4, 6, 7, 1) 以 (1, 3) 为参数做互换得到 (5, 8, 2, 3, 4, 6, 7, 1)。相邻互换随机选择一个位置 $x \in \{1, \cdots, n-1\}$，将该位置中对象与其右邻对象互换。如 (2, 8, 5, 3, 4, 6, 7, 1) 以 2 为参数做相邻互换得到 (2, 5, 8, 3, 4, 6, 7, 1)。

2) *序数编码* (ONE)　即按对象的编号顺序列出对象在最终排序中的序数作为编码。当排列顺序的重要性由小到大时，序数可看作是对象的重要性级别，且序数越大重要性级别越高，这种序数编码又称为基于重要性评分的编码。

严格定义如下，若 $\varphi: \{1, \cdots, n\} \to \{1, \cdots, n\}$ 为一一映射，则称 φ 为置换。如将顺序排列 $(1, \cdots, n)$ 看作特殊排列，则每个置换 φ 都对应唯一的排列 $(\varphi(1), \cdots, \varphi(n))$，由于二者间具有一一对应关系，在含义明确时可将置换 φ 及其对应排列 $(\varphi(1), \cdots, \varphi(n))$ 随意互换使用。定义排序 φ 对应的序数编码为 φ^{-1}。

实际上排序编码是通过描述位置 i 对应的对象 $\varphi(i)$ 定义排序，而序数编码则通过描述对象 j 所在位置 $\varphi^{-1}(j)$ 定义排序。以排序 (2, 8, 5, 3, 4, 6, 7, 1) 为例，1 号对象排在第 8 位，2 号对象排在第 1 位，…，8 号对象排在第 2 位，按对象编号序列出各对象对应位置，得到对应序数编码 (8, 1, 4, 5, 3, 6, 7, 2)。这种编码方式可用的遗传操作与排序编码完全相同。针对序数编码提出混合排序交叉和算术排序交叉。

a. *单点混合排序交叉* (MRCX)　*MRCX* 先执行单点交叉，其结果按序数值大小重新排序，其中冲突值随机排序。设 V^1 = (8, 4, 5, 6, 7, 1, 3, 2)，V^2 = (8, 7, 1, 2, 3, 5, 4, 6)，在位置 4 执行单点交叉，得 $V^{1'}$ = (8, 4, 5, 6, 3, 5, 4, 6)，$V^{2'}$ = (8, 7, 1, 2, 7, 1, 3, 2)。重排序后得到

$$MRCX(V^1, V^2) = \langle (8, 2, 5, 6, 1, 4, 3, 7), (8, 6, 1, 4, 7, 2, 5, 3)\rangle$$

MRCX 能保持交叉前基因在父本中的相对排序，使 *MRCX* 后代中的基因序数与某个父本的该基因的值接近。类似地可定义均匀混合排序交叉 (UMRC)，即将 MRCX 的单点交叉换为均匀交叉。

b. *算术排序交叉* (ARCX)　参数为 e 的算术排序交叉 $ARCX_e$ = 参数为 e 的算术交叉 + 排序。设两父本为 $V^1 = (v_1^1, v_2^1, \cdots, v_n^1)$、$V^2 = (v_1^2, v_2^2, \cdots, v_n^2)$，$e \in [0, 1]$，$\alpha_i$ 以概率 e 取自 [0, 1] 上的均匀分布随机数，以概率 $(1-e)$ 取 1 $(i=1, \cdots, n)$。定义 $AMCX_e(V^1, V^2) = \langle (\alpha_i v_i^1 + (1-\alpha_i) v_i^2) |_i, (\alpha_i v_i^2 + (1-\alpha_i) v_i^1) |_i \rangle$，$ARCX_e(V^1, V^2) = RankIndex(AMCX_e(V^1, V^2))$。其中 $RankIndex_e(\boldsymbol{V})$ 表示将向量 $\boldsymbol{V}$ 的各分量由小到大排序后用对应序数值取代原分量。

针对序数编码实现了在表现型空间执行插入、互换和相邻互换的变异算子。

针对以上两种编码设计的算子称组合遗传算子。由于序数编码和排序编码的染色体形式完全相同，因此所有组合遗传算子均可不加区别地用于这两种编码。

3) *隐序编码* (IOE) 是一种实数编码，通过列出各基因座对应基因的由大到小排名，可以获得一个排列。例如编码 (0.85, 0.09, 0.60, 0.83, 0.69, 0.56, 0.43, 0.33, 0.97) 对应排列为 (2, 8, 5, 3, 4, 6, 7, 1)。有两种解码方式：**a.** 排序方式，将对应排列解释为排序编码；**b.** 序数方式，

将对应排列解释为序数编码。这种编码的优点是可以使用所有实数编码遗传算子，如离散杂交、算术杂交和均匀变异、正态变异等，另外也可通过变换使用组合遗传算子。

5.2 适应值计算

编码实现了采用 PSCM 的仿真机器人，染色体形式的决策规则可以嵌入仿真机器人的控制器运行。用随机生成的初始化积木世界构成测试集，用统计得分对集木控制器性能进行测试。测试集太小影响性能估计的精度，因此要根据精度要求和计算量的折衷确定其大小，通常选择 20～100。

固定测试集或测试时刻生成测试集选择。采用每次重新生成测试集的方法，相当于对搜索曲面引入随机噪声，尽管对有些问题有助于跳出局部最优[10]，但实验却发现噪声带来了明显不利。在使用最优保留选择策略的情况下，如果最优个体适应值不重新计算，最优个体往往是随机偏差产生的“虚假”最优，最终解的性能较差；如果每一代都重新计算适应值，种群长时间不收敛。定性分析，其原因主要在于噪声的方差明显大于适应值图景上临近区域的方差，使得搜索过程更多地表现出随机游走的特点。因此本文算法采用固定测试集的方法，尽管仍然存在偏差，但实验结果表明，估计性能已经比较接近大样本的测试结果。

确定单次测试时间的方法有两种：**a**. 直接嵌入完整控制器，执行完整任务，测试时间 150 步；**b**. 只采用积木控制器执行集木子任务，此时测试时间选择有三种方案：① 测试时刻用 Steps = 121 + NormRand (μ, σ^2) 生成测试步，其中 *NormRand* (μ, σ^2) 表示均值为 $\mu=0$、方差为 $\sigma^2=6.0^2$ 的正态分布随机数生成函数；②用 *Steps* 公式一次性生成测试步，对不同测试集的同一编号测试用相同的测试步；③ 固定测试步为 121。为减少适应值计算时间，通常采用 **b** 方法。为避免搜索算法陷入局部极小，采用固定测试步。实验表明通过这种测试方法获得的最终解的性能确实较好而且稳定。

5.3 选择算子

标准遗传算法的轮赌选择算子，尽管有着简单的优点，但对克服早熟收敛和后期的收敛速度变慢十分不利。本文采用 Michalewicz 提出的基于排名的选择算子[11]和最优保存策略。这样，除了排名选择本身可以维持稳定的选择压力外，还使得选择压力参数化，便于设计选择压力可调控的适应性遗传算法。

本文对 Michalewicz 的排名选择算子作了修改。在文献 [11] 中，种群按适应值从好到坏排序得到 $\{C_i \mid i=1, \cdots, m\}$，并按下面的等比递降级数分配个体的选择概率

$$P_r(C_i)=\begin{cases} q(1-q)^{i-1}, i=1,\cdots,m-1 \\ (1-q)^{m-1}, i=m。\end{cases} \tag{3}$$

其中：最优个体的选择概率 q 为常数，它的大小决定选择压力的大小。但 $P_r(C_m)$ 作为补足项不服从等比关系，根据上式

$$P_r(C_m)=q(1-q)^{m-2}(1-q)/q= P_r(C_{m-1})(1-q)/q \tag{4}$$

可知，当 $q<0.5$ 时，$(1-q)/q>1$；当 q 较小时，$(1-q)/q>>1$，这意味着给最差个体分配异常高的选择概率。本文提出两种措施消除原分配公式在排名尾部的异常。

令 $P_r''(C_i)=q(1-q)^{i-1}$, $i=1, \cdots, m$, $z=\sum_{i=1}^{m} P'_r(C_i)$。第一种方法采用比例补偿，分配方案定义为：

$$P_r^P(C_i)=P_r'(C_i)/z, i=1,\cdots,m \tag{5}$$

第二种方案采用均匀补偿，分配方案为：

$$P_r^U(C_i)=P_r'(C_i)+(1-z)/m, i=1,\cdots, m。 \tag{6}$$

关于选择压力设置。用参数 q 作为选择压力调控参数，q 越大压力越大。选择压力直观上是指个体适应值差异对繁殖概率的影响，压力大时种群基因分布迅速趋同于群体最优个体。选择压力的设定应同时兼顾群体多样性和收敛速度。

5.4 群体形态及规模

标准遗传算法采用单种群模型，但生物学发现生殖隔离有利于缓解选择压力，保护物种多样性。受此启发的多种群模型采用多个平行进化的同构种群，并按一定小概率在种群间交换个体，在许多情况下使算法的全局最优性得到明显改进[12]。但对适应值图景较简单的问题，多种群算法的优势并不明显。OPGA 算法可以选择单种群或 2 种群进化模式。在 2 种群模式下，平均每 5 代进行一次全局选择交配。

5.5 终止条件

根据试验观察，大部分情况下算法在 30 代以后就基本收敛，固定进化代数为此收敛代数的 1～2 倍。

5.6 其他控制参数

包括交叉和变异算子的使用概率等。对排列编码和序数编码，组合变异算子的操作对象是整个染色体，变异率指染色体执行变异的概率。对隐序编码，需确定基因的变异概率和正态变异的标准差。

6 进化机器人实验结果

本文编程实现了基于 PSCM 控制模型的仿真机器人及其测试环境，以及用于进化决策的排序遗传算法 OPGA。在此基础上建立了完整的机器人进化学习环境，见图 5。

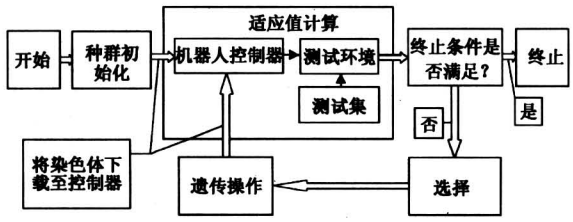


图 5 进化机器人学习环境

Fig.5 Evolutionary robotics learning environment

上述学习环境的主框架为 OPGA，该算法采用基于重要性评分的编码，每个基因占一个字节，编码长 8，且基因值取自 {0，…，7}，并表示相应类型的优先级，数值越大的优先级越高。在为计算个体适应值而将其下载至控制器时，为与优先级预设为 0 的两种类型兼容，各基因值应再加上 1。群体为单种群，群体大小为 21。选择操作采用“修正的非线性排名选择 + 最佳保留”模型，其中修正的排名选择采用“等比分配概率 + 比例补偿”方式，参数 $q = 0.6$；固定测试集，容量取为 100。最大代 30 停机。适应值计算所采用的测试条件为：测试世界对控制器已知，只执行集木任务，测试时间固定为 121 步。每代作 2 100 个测试，在 Intel Pentium MMX166 微机上大约耗时 4 min。

上述算法中大部分可调控参数都通过对比实验作了优选，上面给出的为优选后的配置。对编码、交叉和变异算子的优化过程分两个阶段：**a.** 首先作粗选。为减少计算代价，人工设计排序问题适应值函数，该函数在人工指定一个最优排列的基础上，按任意排列与最优排列的距离（偏移距离）定义该排列的适应值。用该人工排序问题比较所有“编码 + 交叉 + 变异”配置下的算法性能。**b.** 用粗选性能最佳的 7 种算法求解集木控制器设计问题，得到实际性能最佳的算子配置为“序数编码 + 基于位置交叉 + 序数相邻互换变异”，交叉率和变异率均为 1。

采用两个不同的随机测试集分别运行优化配置后的算法各 5 次，得到各终止代的最佳结果和次最佳结果见表 2。终止代种群可能包含多个具有相同适应值的不同最佳结果，对不同运行中得到的相同最佳结果统计其出现次数。测试 T_{100} 即为其适应值，测试 T_{1000} 是在算法结束后，将其嵌入集木控制器，并与进化学习得到的勘察控制器组合构成完整控制器，用另外生成的 1 000 个积木世界作完整任务测试的结果，测试时间为 150 步。

10 次实验均找到了对应组的最高适应值个体。从表 2 可以看出，染色体中存在中性基因及其漂移现象。中性基因是指自然选择对其在种群中的分布不起作用的基因。例如表 2 中第 1.1、1.2、1.3、1.4 染色体的第 2、7 基因座取等位基因 2 或 3 对适应值无明显影响。另外也发现中性基因是相对于特定环境而言的，例如上述 2.3、2.4 染色体的第 4、7 基因座取 0 或 1 在上述容量 100 测试集下都有相同适应值，但在容量 1 000 测试下却存在差别。中性基因的数量从某种程度上反映了分类的有效性。实验同时表明，进化算法具有同时提供多个可行解的群体搜索优势，能够减小测试集容量限制带来的偏差。

表 2 进化机器人学习算法运行结果（10 次）

Table 2 Experimental results of evolutionary roboticslearning algorithm (10 runs)

编号	次数	平均分		局部格局的重要性级别							
		$T_{1\,000}$	T_{100}	1	2	3	4	5	6	7	8
1.1	5	23.600	23.58	1	2	4	0	5	6	3	7
1.2	5	23.600	23.58	1	3	4	0	5	6	2	7
1.3	1	23.604	23.56	1	2	4	0	5	7	3	6
1.4	3	23.604	23.56	1	3	4	0	5	7	2	6
1.5	2	23.618	23.56	1	4	3	0	5	6	2	7
1.6	3	23.622	23.56	2	3	4	0	5	6	1	7
1.7	1	23.613	23.56	2	4	3	1	5	6	0	7
2.1	4	23.622	23.47	2	4	3	0	5	6	1	7
2.2	5	23.622	23.47	2	4	3	0	5	7	1	6
2.3	3	23.622	23.46	2	3	4	0	5	7	1	6
2.4	1	23.613	23.46	2	3	4	1	5	7	0	6
2.5	2	23.613	23.46	2	4	3	1	5	6	0	7
2.6	2	23.613	23.46	2	4	3	1	5	7	0	6

为评价进化算法的性能，对 3 种基于局部格局

的启发式方案选择方法进行了测试，并与进化机器人的结果进行了对比。选测试集容量为 1 000，比较结果见表 3。

3 种作为对照的基于排序的选择方法：**a.** 随机选择：按公式（2）计算将积木从当前位置（x_0，y_0）移到目标位置（x，y）的得分为 $LS(x, y) - LS(x_0, y_0)$，随机选择得分不为 0 的方案。这是一种粗粒度分类的选择策略，即仅将局部格局分为得分和不得分两种；**b.** 按得分选择：计算行动方案的得分 $LS(x, y) - LS(x_0, y_0)$，选择最高得分方案执行。当有多个最优方案时，选择耗时最少的执行。这种方法将局部格局按得分分为 7 类，并直接按照得分值大小排序；**c.** 人工排序：采用表 1 分类法，从直观出发确定局部格局的重要性依次为高分＞位于墙角＞一般得分＞位于墙边。据此得到一个排序（1，5，7，0，3，6，2，4）。

当样本容量为 1 000、样本标准差为 1.601 时，只要样本均值差大于 0.167，显著性水平为 0.01，即可以认为两样本来自不同总体。表 3 表明进化算法结果明显优于一些基于直观的选择方法，即算法是有效的。

表 3　不同排序方法性能比较*

Table 3　Performance comparison of different ranking methods

排序方法	进化排序	随机选择	按得分选择	人工排序
平均得分	23.622	18.860	21.834	21.868

*测试集容量为 1 000

另外可以从统计的角度评价算法有效性。两组实验的初始种群平均适应值的平均值分别为16.37 和 16.81，而进化所得最优结果平均为 23.58 和 23.47，说明学习结果明显优于平均性能。

根据 1999 年 5 月公布的该问题最佳结果：在容量 8 000 测试集下，平均得分为 17.3186[6]。相比之下 EDMBCR 方法性能优势明显。

7　讨论

对于候选方案可按重要性排序的决策问题，提出用分类后的候选方案排序作为决策规则，并采用遗传算法自适应学习排序的进化决策方法 EDMBCR。给出这种方法的应用原则、决策模型、决策分析过程、决策规则进化学习方法和应用该模型的决策过程。提出排序问题遗传算法的相关设计选择。将该进化决策方法应用于 CEC'99 机器人控制器设计开放问题的推积木子问题，结合 PSCM 控制模型、机器人推理机制和遗传算法建立完整进化机器人设计平台。试验表明，基于排序的进化决策较好地解决了该问题。

对于上述决策问题，本文方法与传统方法相比具有无须显式地构造效用函数、能有效处理非数值或非量化指标以及指标冲突和指标相关等问题、较少依赖领域先验知识、自适应学习决策规则、在随机性环境下的稳健性等优点。

排序决策方法具有普遍的应用价值，而进化算法是解决排序问题的有效方法，其优点在于对领域知识有较弱的依赖性，能在有随机噪声的环境下获得可行解，且群体搜索策略能减小训练集容量限制带来的偏差。但在应用基于排序的进化决策方法时必须满足下列 3 个前提条件：

1) 排序是原问题解的有效表示形式。该假设可以在建立决策问题模型之后、选择决策模型之前通过模型分析进行确认。在决策规则确定后，可通过与其他非排序决策方法的性能进行比较来验证。

2) 分类指标正确反映了问题本质。如果分类准则错误或不尽合理将无法找到真正的最优解。分类准则是否合理可通过对比同一分类准则下不同排序之间性能差异是否明显来验证。

3) 分类粒度选择恰当。分类粒度过大时决策规则性能变劣，过小时增加规则学习的难度或浪费计算资源。分类粒度过大和过小的情况可能会同时存在，并可通过检测验证。

将进化算法用于机器人设计时，遵循了两个重要原则。**a.** 仅将进化算法用于解决传统方法难以求解或求解性能不高的问题，对确定性问题尽可能使用确定性方法。**b.** 尽可能充分利用领域知识降低问题复杂度。上述原则对进化算法用于工程实践具有一定的普遍意义。

初步工作表明，进化决策在智能控制等领域有着广阔的应用前景[7]。下一步将继续探索该方法在其他现实决策问题中的应用。

致谢　王正志教授、谢涛博士后审阅了本文的初稿并提出了中肯的建议，陈渝博士为本文部分工作的完成提供了计算资源，特此鸣谢！

参考文献

[1] 陈 珽. 决策分析 [M], 北京: 科学出版社, 1987

[2] Holland J H. Adaptation in natural and artificial system [M], Ann Arbor, MI: University of Michigan Press, 1975

[3] Rechenberg I. Cybernetic solution path of an experimental problem [R]. [Tech. Report], Roy Aircraft Establishment, Library Translation No 1222, Farnborough, Hants, U K, 1965

[4] Fogel L J. Autonomous automata [J]. Industry Research, 1962, 4: 14～19

[5] Gomi T, Griffith A. Evolutionary robotics-an overview [A]. In Proceedings of 1996 IEEE International Conference on Evolutionary Computation (ICEC, '96) [C], Piscataway, NJ: IEEE Press, 1996. 40～49

[6] Welcome homepages of congress on evolutionary computation 1999 [DB/OL], Internet URL: http: // garage. cps. msu. edu/cec99/, 1999

[7] Li Jianqi, Chen Huowang, Wang Bingshan. Evolutionary learning function approximator as robot controller [A]. Eighth International Symposium on Robotics with Applications, World Automation Congress (WAC2000), to be published in Series on Intelligent Automation and Soft Computing [C], Albuquerque, NM: TSI Press, 2000

[8] 荔建琦, 陈火旺, 王兵山. 多维函数的进化逼近 [J]. 计算机学报, 2000, 23 (6): 593～601

[9] 潘正君, 康立山, 陈毓屏. 演化计算 [M], 北京: 清华大学出版社, 南宁: 广西科学技术出版社, 1998

[10] Rana S, Whitley D, Cogswell R. Searching in the presence of noise [A]. Parallel Problem Solving from Nature IV [C]. Voigt H M, Ebeling M, Rechenberg I, et al. Heidelberg: Springer Verlag, 1996. 2～11

[11] Michalewicz Z. Genetic algorithms + data structures = evolution programs [M], Berlin: Springer-Verlag, 1992

[12] Tanese R. Distributed genetic algorithms [A]. In: Proceeding of the Third International Conference on Genetic Algorithms [C], Los Altos: Morgan Kaufmann, 1989. 434～439

Evolutionary Decision Making Based on Candidates Ranking

Li Jianqi, Chen Huowang, Wang Bingshan

(*Department of Computer Science, National University of Defense Technology, Changsha* 410073, *China*)

[**Abstract**] Since the exact Expected Utility Functions (EUF) are not available in many circumstances, it is hard to make decision in environment characterized by incomplete information and uncertain decision results. Being aware of the defects of traditional decision analysis techniques, a new decision making method named Evolutionary Decision Making Based on Candidates Ranking is proposed. By selecting a set of indexes relevant to the expected utility of the candidates, the construction of decision rules can be reduced to finding the quantitative relationship between them. If all of the candidates are classified according to the indexes relevant to their expected utility, then Evolutionary Algorithms can be used to search for the expected utility ranking of the whole set of candidate classes, thus the optimal decision can be made based on the ranking. Some special considerations for Genetic Algorithms for ordering problem are also highlighted. The new method enjoys the advantages of weak dependence on expert knowledge, robustness in environment with random noise, no dependence on explicit EUF, effectively treating non-numerical, non-quantificational indexes and conflicts or correlation among indexes. The effectiveness of the proposed method is validated by its successful application in the controller design of certain simulated robot.

[**Key words**] evolutionary decision making; evolutionary robotics; genetic algorithms for ordering problem