文章编号: 1001-0920(2008)12-1338-05

多巷道固定货架拣选作业优化问题的研究

李梅娟[1,2], 陈雪波[3], 王 莉[3]
(1. 大连理工大学 信息与控制研究中心, 辽宁 大连 116024; 2. 鞍山师范学院 计算机系, 辽宁 鞍山 114005; 3. 辽宁科技大学 电子与信息工程学院, 辽宁 鞍山 114004)

摘 要: 拣选作业的效率直接影响自动化立体仓库系统的整体效益. 为满足客户货单动态变化的需求, 分析了自动化仓库单存/取机对多巷道固定货架拣选操作的工作特点, 构建了含装箱约束条件的多目标货物拣选路径问题的数学模型, 并提出一种带选择算子、插入点操作和动态自适应调整算法参数的改进蚁群算法. 实验表明, 该算法具有较好的全局寻优能力, 收敛速度快, 是解决货物拣选路径优化问题的有效算法.

关键词: 蚁群算法; 自动化立体仓库; 拣选作业; 优化
中图分类号: TP18 **文献标识码**: A

Research on order picking optimization problem for multiple aisles fixed storage racks

LI Mei-juan[1,2], *CHEN Xue-bo*[3], *WANG Li*[3]
(1. Research Center of Information and Control, Dalian University of Technology, Dalian 116024, China; 2. Department of Computer, Anshan Normal University, Anshan 114005, China; 3. School of Electronic and Information Engineering, Liaoning University of Science and Technology, Anshan 114004, China. Correspondent: LI Mei-juan, E-mail: asmeijuanli@126.com)

Abstract: The order picking efficiency directly affects the overall working efficiency of the automated warehouse. To satisfy the need of dynamic change of the items, we analyzed the working characteristics of single storage/retrieval machine serving multiple aisles fixed storage racks. A new mathematic model is constructed with capacity constraint and multiple objectives. An improved ant colony algorithm is presented to solve the order picking problem. Three improvements: Selection operator, interpolation operator and dynamic change on algorithm parameters, are introduced. Experimental results show the improved algorithm has better overall search ability and astringency, and it is an effective solution to the order picking problem.

Key words: Ant colony algorithm; Automated warehouse; Order picking problem; Optimization

1 引 言

自动存取系统 AS/RS(Automated storage and retrieval system)是自动化立体仓库(Automated warehouse)的重要组成部分,多任务拣选操作控制策略[1,2]是影响 AS/RS 性能及仓库系统吞吐效率的主要因素之一. 合理确定拣选路径,使整个拣选作业花费的总时间代价最小是一组合优化问题,称为固定货架拣选作业优化问题 OPP(Order picking problem),它类似于旅行售货商 TSP 问题.

Jeroen 和 Hu 等[1,2]提出了选择 S/R(Storage/Retrieval)机最佳驻留位置,以减少作业时间的方法;Wen 等[3]分析了 S/R 机速度对拣选时间的影响;Chang 等[4]用线性规划方法研究了单 S/R 机完成多巷道拣选任务 S/R 机的预先驻留位置问题. 蚁群算法 ACO(Ant colony algorithm)最初由 Dorigo 等[5]提出,对于求解 TSP 问题,曹国锐等[6]提出了建立信息素扩散模型的改进 ACO 算法,Tsai 等[7]研究了解决大规模 TSP 问题的新型 ACO 算法,吴斌等[8]在 ACO 的基础上提出了分段求解算法,Dorigo 等[9]综述了 ACO 的研究进展及成果. 这些

收稿日期: 2007-10-11; **修回日期**: 2007-12-23.
基金项目: 国家自然科学基金项目(60574010); 辽宁省高等学校优秀人才支持计划项目(R-06); 辽宁省教育厅项目(2008Z001)
作者简介: 李梅娟(1967—),女,河南洛阳人,教授,博士生,从事智能控制、先进控制理论及方法的研究; 陈雪波(1960—),男,辽宁鞍山人,教授,博士生导师,从事智能控制、复杂系统结构分析等研究.

研究表明,ACO 具有分布式计算、正反馈和鲁棒性强等优点,在动态环境下也表现出了高度的灵活性和健壮性. 本文提出一种改进的蚁群算法,研究单 S/R 机服务多巷道货架拣选任务的路径优化问题.

2　系统描述

2.1　问题的提出

图 1 是 AS/RS 一个存储区域的俯视图,货物分类存储,一台 S/R 机负责对存储区域的多排货架拣选,相邻两排货架之间有一条巷道,S/R 机在巷道内运行存取货物. 存储区左端设置一出入库 I/O (Input/Output)台,根据客户货单,拣选作业过程为:计算机发送货单后,S/R 机开始运动,对第一条目所对应的货位进行存取;然后访问下一条目,直到顺序执行完货单中的所有条目;拣选完毕,S/R 机携带货箱返回至 I/O 台,将货箱传送至输送系统,完成一次作业.

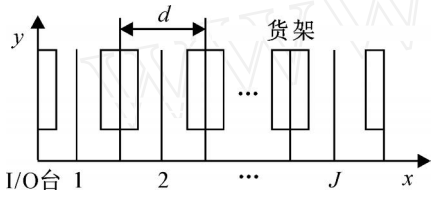

图 1　多巷道 AS/RS 存储区俯视图

若改变货单中各个条目的顺序,则将有不同的存取路径与之对应,所以 OPP 问题可归结为多旅行售货商 MTSP 问题,是典型的 NP 难题.

2.2　基本假设及基本约束条件

2.2.1　基本设定条件

设定 1　坐标(X, Y, Z)表示货位点,巷道号用$\{k_1, k_2 \in (1,2,\cdots,J)\}$表示,$(0,0,0)$视为 I/O 台,作为整个拣选作业的附加货位点. 货位点及货物体积用(X, Y, Z, Q)表示. 每排货架之间的间距为 d,货格长度为 a,宽度为 b,高度为 h,且 d, a, b, h均为常数.

设定 2　货架上存储的货物类型已知,货架当前的存放状态(空或满)是确定的;存取时,S/R 机恒速运行,启动和制动过程忽略不计.

设定 3　S/R 机存取货物时能同时沿 x 轴和 z 轴方向运行,或同时沿 y 轴和 z 轴方向运行,但不能同时沿 x 轴和 y 轴方向运行.

设定 4　拣选时,S/R 机对任何货物的存取速度恒定,不因存取顺序的改变而变化,计算拣选时间代价时,忽略货物存取时间.

2.2.2　基本约束条件

1) S/R 机运行限制:S/R 机同一时刻只能在一条巷道内(x 轴或 y 轴)运行.

2) S/R 机的容量限制:每次作业拣选货物时,货物装箱总容量不大于 S/R 机的最大容量 C.

3) 路线的单向性:S/R 机从 I/O 台出发访问货位点的行驶方向一致连续,不能发生折返,最后返回 I/O 台.

2.3　数学模型

当 S/R 机接收到拣选作业命令时从 I/O 台出发,存/取完 k_1 巷道货位 i 再对 k_2 巷道货位 j 操作,运行距离为

$$d_{i_{k_1} j_{k_2}} = \begin{cases} \max\{(|Y_i - Y_j|), (|Z_i - Z_j|)\}, & k_1 = k_2; \\ \max\{(|X_i - X_j|) + (|Y_i + Y_j|), (|Z_i - Z_j|)\}, & k_1 \neq k_2. \end{cases}$$

花费的时间代价为

$$T_{i_{k_1} j_{k_2}} = \begin{cases} \max\left\{\dfrac{|Y_i - Y_j|}{V_y}, \dfrac{|Z_i - Z_j|}{V_z}\right\}, & k_1 = k_2; \\ \max\left\{\dfrac{|X_i - X_j|}{V_x}, \dfrac{|Y_i - Y_j|}{V_y}, \dfrac{|Z_i - Z_j|}{V_z}\right\}, & k_1 \neq k_2. \end{cases} \tag{1}$$

其中:(X_i, Y_i, Z_i)为货位点 i 的坐标;(X_j, Y_j, Z_j)为货位点 j 的坐标.

对于 n 个依次编号为$\{1,2,\cdots,n\}$的待拣选货位,所求目标是寻找一条最佳的货物拣选顺序,使 S/R 机完成一个货单拣选作业花费的时间最短,且作业次数 m 最少.

设 $x^l_{i_{k_1} j_{k_2}}$ 是决策变量,取值范围是 $x^l_{i_{k_1} j_{k_2}} \in \{0, 1\}$, $i_{k_1}, j_{k_2} = 1,2,\cdots,n$, $l = 1,2,\cdots,m$. $x^l_{i_{k_1} j_{k_2}} = 1$ 表示 S/R 机第 l 次行走的路线是访问完 k_1 巷道货位 i 后接着访问 k_2 巷道货位 j; $x^l_{i_{k_1} j_{k_2}} = 0$ 表示 S/R 机第 l 次行走时未选择这条路线.

构造数学模型如下:

$$\min \sum_{k_1, k_2 = 1}^{J} \sum_{i,j=1}^{n} T_{i_{k_1} j_{k_2}} \sum_{l=1}^{m} x^l_{i_{k_1} j_{k_2}}, \tag{2}$$

$$\min\ m. \tag{3}$$

约束条件为

$$\sum_{i,j=1}^{n} \sum_{k_1, k_2 = 1}^{J} \sum_{l=1}^{m} x^l_{i_{k_1} j_{k_2}} = \begin{cases} 1, & j_{k_2} = 1,2,\cdots,n; \\ m, & j_{k_2} = 0; \end{cases} \quad k_1, k_2 = 1,2,\cdots,J. \tag{4}$$

$$W_t + \sum_{k=1}^{J} U_{kt} = 1, \quad t = 1,2,\cdots, \quad k = 1,2,\cdots,J. \tag{5}$$

$$\sum_{k_1,k_2=1}^{J}\sum_{i,j=1}^{n} c_{i_{k_1}j_{k_2}} \le C,\ l = 1,2,\cdots m,\quad k_1,\ k_2 = 1,2,\cdots,J. \tag{6}$$

其中：$T_{i_{k_1}j_{k_2}}$ 为 S/R 机从货位 i 到货位 j 所需要的时间；m 为作业次数；n 为待拣选货位总数；目标(2) 确定 S/R 机进行货物拣选的运行时间最短；目标(3) 确定作业次数最少；约束(4) 保证每个货位只被访问一次；约束(5) 要求同一时刻 S/R 机只能在一条巷道内运行，W_t，U_{kt} 分别表示 t 时刻 S/R 机位于 x 轴和 y 轴，$\{W_t, U_{kt} \in \{0,1\},\ t = 1,2,\cdots,\ k = 1,2,\cdots,J\}$，而且

$$W_t = \begin{cases} 1, & \text{S/R 机在 } t \text{ 时刻位于 } x \text{ 轴}(Y_t = 0); \\ 0, & \text{否则}; \end{cases}$$

$$U_{kt} = \begin{cases} 1, & \text{S/R 机在 } t \text{ 时刻位于第 } k \text{ 条巷道}; \\ 0, & \text{否则}; \end{cases}$$

约束(6) 要求每次作业拣选货物容量不能大于 S/R 机的最大容量.

3 改进的蚁群算法设计

OPP 是多目标组合优化问题，与 TSP 问题既具有一定的相似性，又存在许多差异，其复杂程度远高于 TSP. 该问题变量多，求解过程复杂. 基于群集智能的 ACO，在求解可用分布图表示的组合优化问题方面[9] 具有很强的发现较好解的能力，并在求解 TSP 中取得了成果[6-9]. 文献[10,11] 证明了改进的 ACO 算法在求解 TSP 问题时能收敛到全局最优解. 因此，本文根据所研究问题的特点，针对蚁群算法在解决组合优化问题时存在收敛速度与全局搜索能力之间的矛盾，设计了一种改进的蚁群算法.

3.1 符号定义

从人工智能的角度，称蚁群算法中的蚂蚁为“人工蚂蚁”. 在此每个人工蚂蚁都是一个 agent，它根据当前路段上信息素的强弱随机选取运动方向. 设任意一只蚂蚁为 $A_k = \langle F_k, E_k, V_k' \rangle$. 其中：$F_k$ 为该蚂蚁一次完全迭代后根据式(2) 求得的该方案的目标函数值，即拣选时间；E_k 为该蚂蚁走过的路段留下的信息素的量，$E_k = Q_L / F_k$，Q_L 是常数；V_k' 为该蚂蚁走过的顶点集合.

在货物拣选问题中，设货单条目中各货位点组成有向图[12]，用图 $G = (V, R)$ 表示. 其中：$V = \{v_i \mid i = 0,1,\cdots,n\}$ 为顶点集合，v_0 代表 I/O 台(附加货位点)，其余点代表被存取的货位点；$R = \{r_{ij} \mid i \in [0,n], j \in [0,n]\}$ 为顶点间连接路段的集合.

假设共有 M 只蚂蚁在图中运动，与基本蚁群算法相同，搜索过程中蚂蚁 k 在 t 时刻从 v_i 向 v_j 转移的概率为

$$p_{ij}^k = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[1/d_{i_{k_1}j_{k_2}}]^{\beta}}{\sum_{s \in \text{allowed}_k}[\tau_{is}(t)]^{\alpha}[1/d_{i_{k_1}s_{k_2}}]^{\beta}}, \\ \quad j \in \text{allowed}_k; \\ 0,\ j \notin \text{allowed}_k. \end{cases} \tag{7}$$

其中：$\text{allowed}_k = V - V_k'$ 表示蚂蚁 k 下一步允许选择的顶点，α 为路段的重要度，β 为路段的可见度，$\tau_{ij}(t)$ 表示在循环 t 路段 (v_i, v_j) 的信息素. 经过 n 个时刻蚂蚁完成一次循环，对于各个路径上的信息素，可根据 $\tau_{ij}(t+n) = \rho\tau_{ij}(t) + (1-\rho)\Delta\tau_{ij}(t)$ 进行调整，ρ 为信息素挥发速度系数，$\Delta\tau_{ij}(t)$ 表示本次循环中路段 (v_i, v_j) 的信息素增量.

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{M}\Delta\tau_{ij}^k(t);$$

$$\Delta\tau_{ij}^k(t) = \begin{cases} E_k, & \text{第 } k \text{ 只蚂蚁在循环} \\ & t \text{ 经过路段 } r_{ij}; \\ 0, & \text{否则}. \end{cases} \tag{8}$$

其中 $\Delta\tau_{ij}^k(t)$ 表示第 k 只蚂蚁本次循环中留在路段 (v_i, v_j) 的信息素增量.

3.2 算法改进操作

3.2.1 动态插入点操作

在蚂蚁行走轨迹的基础上设计插入点操作. 插入点操作是动态的，而且是必须的. 蚂蚁运行的轨迹称为逻辑轨迹，在插入点以后才成为实际轨迹，即问题真正的解. 例如，一只蚂蚁的行走轨迹为 $(v_0, v_1, v_3, v_5, v_4, v_2)$，对该轨迹进行插入点操作，如果最后插入点结果为 $(v_0, v_1, v_3, v_0, v_5, v_4, v_2, v_0)$，则表示此次拣选作业方案由 S/R 机分两次完成拣选任务，第 1 次作业路径是 (v_0, v_1, v_3, v_0)，第 2 次作业路径是 $(v_0, v_5, v_4, v_2, v_0)$. 进行插入点操作时，应考虑 S/R 机容量约束条件.

拣选时不允许超过 S/R 机最大容量，进行插入点操作时，从行走轨迹的第 2 个点开始，$i' = 1$，依次将各货位点的容量进行累加，当 $\sum_{i=1}^{k} c_i \le C$ 而 $\sum_{i=1}^{k+1} c_i > C$ 时，在 k 与 $k+1$ 之间插入 v_0 点，然后从 $k+1$ 点开始 $(i' = k+1)$，重新计算插入点位置. 计算时可能出现 r_{ij} 趋向于无穷大，这表示路段 v_i 与 v_j 没连通，而是在 v_i 和 v_j 之间插入了 v_0，v_0 与 v_i 和 v_j 是连通的，是问题的可行解.

3.2.2 自适应调整信息素挥发系数

问题的规模较大时，由于信息素挥发速度系数 ρ 存在，ρ 过大，算法的全局搜索能力会降低；减小 ρ，算法的全局搜索能力会随之提高，但收敛速度变慢. 文中对 ρ 值采取自适应控制策略，求解初始阶段 ρ 值取稍大一些以增加全局搜索能力，后期增加收敛速

度.随着循环次数的不断增加,若每次的最优值相差不大,则说明过程陷入了某个极值点,不一定是全局最优解.此时,将 ρ 改为阈值函数

$$\rho(n) = \begin{cases} \xi \cdot \rho(n-1), & \xi \cdot \rho(n-1) > \rho_{\min}; \\ \rho_{\min}, & \text{否则}. \end{cases} \tag{9}$$

其中 $\xi \in (0,1)$ 为挥发约束系数.

3.2.3　选择算子操作

设计选择算子.通过两个解的比较,选择好的解作为新的起点进行下一次迭代.设有一个全局变量 A^*,代表目前为止所找到的目标函数值最好的蚂蚁轨迹.每次迭代完成时,将 A^* 与当前最优方案的蚂蚁 A' 的轨迹相比较,若 A' 的目标值优于 A^*,则 $A^* = A'$,在计算信息素强度时,将 A' 替换掉目标函数值差的蚂蚁的轨迹.

求解过程中,经过多次循环解的最优值没有明显改进时,为避免陷入局部最优,按式(9)动态调整 ρ,使算法既能探索新解又能快速收敛到近优解.文中 ξ 取 0.95,$\rho_{\min}$ 取 0.1.

3.3　算法描述

改进算法流程如下:

Step1:初始化算法参数.$t = 0$,$n = 0$,$\tau_{ij}(0) = \tau_0$,$\Delta\tau_{ij} = 0$,$V_k' = \Phi$(禁忌表为空),设定 ρ_0,$\rho_{\min}$,α,β,Q_L.

Step2:将 M 只蚂蚁放置在 I/O 台初始位置,开始循环搜索路径,根据式(7)计算 A_k 的概率 p_{ij}^k,选择下一个拣选的货位点 v_j 并记录该点,将 v_j 加入到 V_k 中.

Step3:判断是否满足条件 $\sum_{i=1}^{k} c_i \le C$ 且 $\sum_{i=1}^{k+1} c_i > C$,若满足,则继续;否则,转 Step2.

Step4:插入点计算.在 V_k 中插入 v_0,计算 G_k 和 E_k,修正 V_k,清空 V_k'.

Step5:计算信息素挥发速度系数 ρ.

Step6:选择算子操作.记录目前为止蚂蚁的最优运动轨迹 A^*.

Step7:更新路段上的信息量 $\tau_{ij}(t+n)$.

Step8:判断所有蚂蚁走过的路径轨迹是否完全相同.若不同,则将 v_0 插入 V_k' 中,转 Step2;否则,终止程序.

4　实验与分析

为了验证改进蚁群算法的有效性,测试数据如下:货架系统由 10 排、72 列组成,每排 10 层共 720 个货位.货架系统及 S/R 机各参数如下:$V_x = V_y = 3\,\text{m/s}$,$V_z = 1\,\text{m/s}$,$d = 4\,\text{m}$,$a = 1\,\text{m}$,$b = 1\,\text{m}$,$h = 1\,\text{m}$,$C = 70\,\text{dm}^3$.算法中所取参数如下:蚂蚁数目为 $M = 50$,$\rho_0 = 1$,$\rho_{\min} = 0.1$,$\alpha = 1$,$\beta = 2$,$Q_L = 1000$,最大迭代步数为 2000.用基本 ACO 算法和本文算法分别对 10,30,50 的货单拣选作业进行试验,并与优化前的性能指标(按货单中的顺序进行拣选作业求得的性能指标)进行比较.

4.1　N = 10

随机产生一组含 10 个货位点的拣选单,各货位点如下:{(2,63,4,9)(5,6,9,2)(2,22,8,9)(6,26,5,7)(9,39,4,8)(4,47,7,8)(1,54,4,5)(3,48,8,8)(8,24,1,9)(3,52,10,4)}.用货位点次序表示最短路径,基本蚁群算法优化后求得的最短拣选路径为{0,1,3,7,8,2,6,10,4,9,5,0};本文算法优化后求得的最短路径为{0,7,3,1,8,10,6,2,4,9,5,0}.两种算法求解结果如图 2 所示,与优化前性能指标对比如表 1 所示.可以看出,基本 ACO 拣选时间比优化前减少了 79.73 s,改进的 ACO 拣选时间比优化前减少了 83.35 s.实验结果表明,改进蚁群算法的优化性能和收敛速度都比基本 ACO 好.

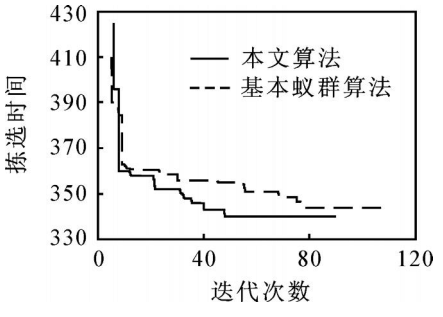

图 2　N = 10 时本文算法与基本蚁群算法求解结果

4.2　N = 30

随机产生包含 30 个待拣选货位点的拣选单,各货位点如下:{(9,67,2,8)(3,16,1,9)(6,43,7,2)(1,35,4,9)(9,54,2,5)(5,54,4,2)(2,32,4,15)(2,1,8,8)(4,58,5,8)(3,32,2,2)(7,44,6,5)(5,56,8,8)(3,65,0,9)(8,52,6,9)(4,13,3,15)(2,29,7,5)

表 1　拣选作业优化前后性能指标比较

货位规模	优化前		用基本蚁群算法求解				用改进蚁群算法求解			
	拣选时间 t/s	作业次数	拣选时间 t/s	优化率/%	作业次数	迭代数	拣选时间 t/s	优化率/%	作业次数	迭代数
10	423.54	1	343.81	18.82	1	110	340.19	19.51	1	91
30	876.72	3	630.12	28.13	3	278	627.77	28.40	3	160
50	2367.47	9	1473.86	37.75	7	496	1472.59	37.80	7	237

(1 ,66 ,5 ,2) (1 ,65 ,6 ,2) (10 ,29 ,4 ,2) (4 ,63 ,3 ,2) (9 ,4 ,2 ,5) (7 ,4 ,2 ,8) (4 ,58 ,6 ,5) (3 ,1 ,3 ,9) (2 ,10 ,5 ,8) (6 ,14 ,5 ,8) (1 ,14 ,6 ,2) (10 ,43 ,3 ,5) (4 ,19 ,8 ,8) (3 ,36 ,8 ,2) }. 改进的蚁群算法优化后求得的一个可行解为{0 ,4 ,8 ,7 ,2 ,10 ,9 ,6 ,3 ,5 ,1 ,0} , {0 ,17 ,16 ,13 ,15 ,12 ,11 ,14 ,0} ,{0 ,27 ,18 ,25 ,24 ,30 ,29 ,3 ,20 ,26 ,22 ,21 ,19 ,28 ,0} ,两种算法的比较结果见表 1. 可以看出, 改进的蚁群算法优化率比优化前提高了 28. 40 % ,比基本 ACO 优化率提高了 0. 27 %. 图 3 结果表明 ,本文算法求解的迭代次数比基本 ACO 减少了 118 次 ,收敛速度明显提高.

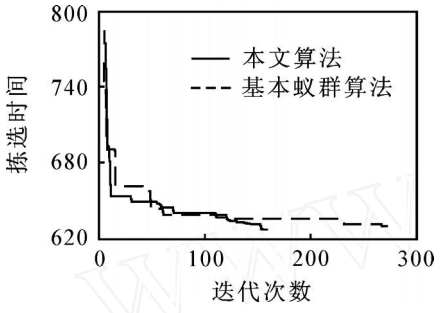

图 3 N = 30 时本文算法与基本蚁群算法求解结果

货位规模增大到 $N = 50$ 时 ,本文算法迭代到 237 次便能搜索到近优解 ,而基本 ACO 则需迭代到 496 次. 说明改进算法收敛速度显著提高 ,而且能求出全局近优解.

5 结 论

本文设计了一种改进的蚁群算法 ,以求解单 S/R 机完成对多条巷道的固定货架系统拣选作业问题. 仿真实验表明 ,该算法收敛速度快 ,能求得全局近优解 ,不仅可以求解货物拣选路径问题 ,而且可应用于其他归纳为图结构的组合优化问题 ,具有实用价值.

参考文献(References)

[1] Jeroen P , Van den Berg. Analytic expressions for the optimal dwell point in an automated storage/ retrieval system[J]. Int Production Economics , 2002 , 76 (1) : 13-25.

[2] Hu Ya-hong , Huang Shell-ying , Chen Chuan-yu , et al. Travel time analysis of a new automated storage and retrieval system [J]. Computers & Operations Research , 2005 , 32(6) : 1514-1544.

[3] Wen U P , Chang D T , Chen S P. The impact of acceleration/ deceleration on travel-time models in class-based automated S/ R system[J]. IEEE Trans , 2001 , 33(7) : 599-608.

[4] Chang S H , Egbelu P J. Relative pre-positioning of storage/ retrieval machines in automated storage/ retrieval system successful response time [J]. IIE Trans , 1997 , 29(4) : 302-312.

[5] Dorigo M , Vitorio M , Alberto C. The ant system: Optimization by a colony of cooperating agents [J]. IEEE Trans on Systems , Man and Cybernetics — Part B , 1996 , 26(1) : 1-13.

[6] 黄国锐 , 曹先彬 , 王煦法. 基于信息素扩散的蚁群算法[J]. 电子学报 , 2004 , 32(5) : 865-868.
(Huang G R , Cao X B , Wang X F. An ant colony optimization algorithm on pheromone diffusion[J]. Acta Electronica Sinica , 2004 , 32(5) : 865-868.)

[7] Tsai C F , Tsai C W. A new approach for solving large traveling salesmam problem using evolution ant rules [C]. Proc of the 2002 Int'1 Joint Conf. Honolulu: IEEE Press , 2002: 1540-1545.

[8] 吴斌 , 史忠植. 一种基于蚁群算法的 TSP 问题分段求解算法[J]. 计算机学报 , 2001 , 24(12) : 1328-1333.
(Wu B , Shi Z Z. An ant colony algorithm based partition algorithm for TSP[J]. Chinese J Computer , 2001 , 24(12) : 1328-1333.)

[9] Dorigo M , Brirattari M , Stützle T. Ant colony optimization: Artificial ants as a computational intelligence technique [J]. IEEE Computational Intelligence Magazine , 2006 , 1(4) : 28-39.

[10] 冯远静 , 冯祖仁 , 彭勤科. 一类自适应蚁群算法及其收敛性分析[J]. 控制理论与应用 , 2005 , 22 (5) : 713-717.
(Feng Y J , Feng Z R , Peng Q K. Adaptive ant colony optimization algorithm and its convergence[J]. Control Theory and Applications , 2005 , 22 (5) : 713-717.)

[11] Thomas S , Marco D A. A short convergence for a class of ant colony optimization algorithm[J]. IEEE Trans on Evolutionary Computation , 2002 , 6(4) : 358-365.

[12] Nenad S R , Stejepan B , Zdenko K , et al. String algebra-based approach to dynamic routing in multi-LGV automated warehouse systems[C]. Proc of the 2006 IEEE Int Conf on Control Applications. Munich: IEEE , 2006: 1872-1878.