doi: 10.7690/bgzdh.2016.03.001

基于蚁群算法的空间目标地基监视重调度

鄢青青[1]，沈怀荣[2]，邵琼玲[2]
（1. 装备学院研究生管理大队，北京 101416；2. 装备学院航天装备系，北京 101416）

摘要：为解决空间目标地基监视中的资源故障或因天气、供电导致的资源失效等对调度的干扰问题，在扰动度量的基础上提出一种基于扰动邻域搜索的蚁群算法。建立以失败需求综合优先级最小和扰动最小为目标的重调度模型，算法中采用 2 个信息素矩阵来产生一个可行解，通过邻域搜索在可行解的扰动邻域内对其进行局部优化，再用蚁群算法进行全局寻优。仿真实验结果表明：该算法是有效、可行的，能在可接受时间内收敛，且其解的质量相对启发式方法有明显提升。

关键词：空间监视；重调度；蚁群算法；邻域搜索
中图分类号：TJ03　　**文献标志码：**A

Ground-based Space Surveillance Rescheduling Based on Ant Colony Optimization

Yan Qingqing[1], Shen Huairong[2], Shao Qiongling[2]
(1. *Administrant Brigade of Postgraduate*, *Equipment Academy*, *Beijing* 101416, *China*;
2. *Department of Spaceflight Equipment*, *Equipment Academy*, *Beijing* 101416, *China*)

Abstract: In order to solve the disruption problem caused by resource fault or resource failure due to weather or power supply in scheduling for ground-based space surveillance, a rescheduling model was constructed with the optimal objective which was a weighted sum of disturbances as well as integrated priority of failed demands. Through the disturbances metric in the rescheduling process, a rescheduling algorithm based on disturbance neighborhood searching and ant colony algorithm was proposed. Through the algorithm, two pheromone matrices were used to generate a feasible solution; local optimization was adopted to improve the feasible solution in its neighborhood; then ant colony algorithm was adopted for global optimization. Experimental results from simulation of resource failure showed that the proposed algorithm was useful and feasible.

Keywords: space surveillance; rescheduling; ant colony optimization; neighborhood search

0 引言

空间目标监视是感知大气层外空间态势的重要手段[1]。空间目标监视网规划调度是将有限的地基监视资源在时间上优化分配给不同的监视任务和关注的空间目标[2-3]。目前针对空间目标监视任务规划调度问题的研究主要集中在对静态环境下的全网或单个监视资源进行静态任务规划调度[3-5]；然而，在实际的空间目标监视过程中，静态调度方案常会因为监视资源网、运行环境或用户需求发生不确定的变化而部分或全部失效，特别是一些重要监视任务和空间目标可能会因此失去资源，需要在很短的时间内产生一个新调度方案，以使受到不确定事件影响的监视需求重新得到满足。

文献[6]在对生产系统的研究中将重调度(rescheduling)定义为：更新现有的生产计划以响应中断或其他变动。重调度按实施策略一般有全局重调度、部分重调度和右移重调度 3 类[7-8]，按驱动方式则主要有在线调度(实时调度)、周期性重调度、事件驱动重调度和混合型重调度等方法[9-12]。文献[13-14]分别在在线调度和不修改原调度的基础上，采用改进调度的方法，研究了空间目标监视的重调度问题，但这 2 种方法易造成局部最优或者让高优先级的任务无法获得监视窗口。

笔者考虑资源故障类不确定事件，采用事件触发的重调度方法，建立了以失败需求综合优先级最小和扰动最小为目标的重调度模型，并在扰动度量的基础上提出一种基于扰动邻域搜索的蚁群算法。结果表明，笔者所设计算法在空间目标地基监视重调度中是有效、可行的。

1 空间目标地基监视重调度模型

1.1 问题描述

在初始静态调度问题中：$\{M_m; m \in [1, N_M]\}$是初始调度方案中包含的一组计划监视任务，每个任务都包括指定的监视需求、获取数据类型

收稿日期：2015-11-18；修回日期：2015-12-26
作者简介：鄢青青(1986—)，男，重庆人，在读博士，从事空间目标监视网调度研究。

$\left\{\mathrm{Type}_1^{\mathrm{data}},\mathrm{Type}_2^{\mathrm{data}},\cdots,\mathrm{Type}_{t_1}^{\mathrm{data}}\right\}_m$、监视条件(最低观测仰角 $E_m^{\min}$、最短跟踪窗口时长 $WL_m^{\min}$、最大采样间隔 $TS_m^{\min}$)等约束；$\left\{S_j;\ j\in[1,N_S]\right\}$ 是地基监视网中的所有监视资源；$\left\{O_i;i\in[1,N_O]\right\}$是一组需要监视的空间目标。通过求解静态调度问题，可以得到一个监视任务、空间目标、监视资源和监视时间窗口对应的计划表，即静态调度方案P_0。

空间目标地基监视中的不确定事件主要包括资源失效(资源故障及其他原因导致的资源不可用)、任务插入(紧急任务和普通任务)以及空间目标状态变化等；但对这些不确定事件的响应，经过预处理后均可表述为一组初始监视需求分配监视资源，故文中以资源故障类不确定事件作为触发重调度的典型事件进行研究。重调度初始时刻为启动重调度并做数据预处理后，此时所参照的初始调度方案不再是静态调度方案P_0，故将P_0中已执行部分和因不确定事件而失效部分删除后的初始方案记为 $\hat{P}_0$，相应的监视任务、空间目标、资源以及监视窗口需求集合记为 $\hat{N}_M$、$\hat{N}_O$、$\hat{N}_S$、$\hat{R}_0$。

空间目标地基监视重调度问题可以描述为：在初始调度方案 $\hat{P}_0$ 的基础上，考虑空间目标地基监视中的不确定事件在调度周期内发生，以对初始调度计划 $\hat{P}_0$ 的扰动最小和未成功调度的需求数目最少为调度目标，得到一个满足约束条件的调整后调度方案。问题中假设所有同类监视资源(光学望远镜、机械雷达或相控阵雷达)具有相同的状态转换时间，同一测站位置同时刻只有一个监视资源在用。

1.2 重调度模型

重调度是在已有初始调度计划的基础上进行再次调度，故在重调度时不但要考虑初始调度目标的最优化，还应考虑对初始调度计划的扰动尽量小。为便于识别，规定后文中 m、i、j 分别专指 $\hat{N}_M$、$\hat{N}_O$、$\hat{N}_S$ 中的任务 m、空间目标 i 和资源 j。空间目标地基监视的重调度模型如下：

$$\min F_X=\alpha\bullet\sum_{i=1}^{\hat{N}_O}p_i\sum_{m=1}^{\hat{N}_M}p_m\prod_{l=1}^{\hat{N}_{m,i}^r}\left(1-x_{m,i,l}\right)+\beta\bullet\left(\lambda_1\xi_1+\lambda_2\xi_2\right)\bullet\left(1+\lambda_3\xi_3\right);\tag{1}$$

$$\alpha+\beta=1;\tag{2}$$

$$\xi_1=\sum_{i=1}^{\hat{N}_O}p_i\sum_{m=1}^{\hat{N}_M}p_m\times\sum_{l=1}^{\hat{N}_{m,i}^r}\Gamma\left(w_{i,j_0},\ x_{m,i,l}\bullet w_{i,j}\right)\bullet x_{m,i,l}^0;\tag{3}$$

$$\xi_2=\sum_{i=1}^{\hat{N}_O}\sum_{m=1}^{\hat{N}_M}\sum_{l=1}^{\hat{N}_{m,i}^r}C_j^u\left(\tau_{i,j}^e-\tau_{i,j}^b\right)\times\Gamma\left(w_{i,j},\ x_{m,i,l}^0\bullet w_{i,j_0}\right)\bullet x_{m,i,l};\tag{4}$$

$$\xi_3=\sum_{m=1}^{\hat{N}_M}C_m^p q_m;\tag{5}$$

$$\Gamma(*,*)=\begin{cases}1, & \text{2个窗口不同时}\\ 0, & \text{2个窗口相同时}\end{cases},$$

$$q_m=\begin{cases}0 & Q_m^0\leqslant Q_m\\ Q_m^0-Q_m & Q_m^0>Q_m\end{cases},$$

$$Q_m=\frac{N_m^{\text{successful}}}{N_m^{\text{successful}}+N_m^{\text{failed}}}。\tag{6}$$

式中：$x_{m,i,l}$是表示需求状态(满足与否)的 0～1 型决策变量，标志任务 m 中目标 i 的第 l 个需求是否被满足；当 $x_{m,i,l}$ 为 1 时，表示该需求必然已被一个时间窗口 $w_{i,j}\left(\tau_{i,j}^b,\tau_{i,j}^e\right)$(式中简写为 $w_{i,j}$)满足；$\tau_{i,j}^b$、$\tau_{i,j}^e$ 分别为时间窗口的开始和结束时间，每个时间窗口关联空间目标 i 和资源 j；$x_{m,i,l}^0$和 w_{i,j_0} 是重调度初始时刻的决策变量取值和对应的时间窗口，为常值；p_i 是空间目标 i 的优先级；p_m 为监视任务 m 的优先级；$N_{m,i}^r$为任务 m 对空间目标 i 监视需求的数目；α、β是 2 个优化目标的权重系数，根据用户对需求满足和扰动程度的偏重设置其相对大小；λ_1, λ_2, λ_3 为 3 个扰动系数；ξ_1、ξ_2 和ξ_3 分别为对初始调度计划的修改成本、监视资源的使用成本和监视任务完成度下降的惩罚 3 个扰动度量评价指标；$\Gamma(*,*)$用于判断重调度前后满足同一需求的 2 个窗口是否相同；C_j^u 是资源 j 的单位时间运行成本；C_m^p 是对任务 m 的完成度下降的惩罚系数；q_m 为任务 m 的完成度下降程度，是其初始完成度 Q_m^0 与当前完成度 Q_m 的差值，其中 $N_m^{\text{successful}}$ 和 N_m^{failed} 分别是任务 m 的需求完成和失败数目。

约束条件：

$$\sum_{i=1}^{\hat{N}_O}\sum_{m=1}^{\hat{N}_M}\sum_{l=1}^{\hat{N}_{m,i}^r}x_{m,i,l}^0-\sum_{i=1}^{\hat{N}_O}\sum_{m=1}^{\hat{N}_M}\sum_{l=1}^{\hat{N}_{m,i}^r}x_{m,i,l}=\sum_{i=1}^{\hat{N}_O}\sum_{m=1}^{\hat{N}_M}\sum_{l=1}^{\hat{N}_{m,i}^r}\Gamma\left(w_{i,j_0},x_{m,i,l}\bullet w_{i,j}\right)\bullet x_{m,i,l}^0+\sum_{i=1}^{\hat{N}_O}\sum_{m=1}^{\hat{N}_M}\sum_{l=1}^{\hat{N}_{m,i}^r}\Gamma\left(w_{i,j},x_{m,i,l}^0\bullet w_{i,j_0}\right)\bullet x_{m,i,l};\tag{7}$$

$$T_{i,j}^{\mathrm{bov}} \leqslant \tau_{i,j}^{b} < \tau_{i,j}^{e} \leqslant T_{i,j}^{\mathrm{eov}}; \tag{8}$$

$$\left[\tau_{i,j}^{b},\ \tau_{i,j}^{e}\right]_{\mathrm{optical}} \subseteq \left(\left[T_{j}^{\mathrm{bou}},\ T_{j}^{\mathrm{eou}}\right]_{\mathrm{optical}} \cap \left[T_{i}^{\mathrm{boi}},\ T_{i}^{\mathrm{eoi}}\right]\right); \tag{9}$$

$$\left[\tau_{i,j}^{b},\ \tau_{i,j}^{e}\right] \subseteq \left[T_{j}^{b},\ T_{j}^{e}\right]; \tag{10}$$

$$\left.\begin{aligned} \rho_{i,j} &\in \left[\rho_{j}^{\min},\ \rho_{j}^{\max}\right] \\ A_{i,j} &\in \left[A_{j}^{\min},\ A_{j}^{\max}\right] \\ E_{i,j} &\in \left[E_{j}^{\min},\ E_{j}^{\max}\right] \end{aligned}\right\}; \tag{11}$$

$$\sum_{i}\sum_{m}\sum_{l=1} x_{m,i,l}\Big|_{\substack{w_{i,j}=w_{i,j1} \\ t=t_1}} \leqslant \eta_{j1}; \tag{12}$$

$$\left\{\mathrm{Type}_{1}^{\mathrm{data}}, \mathrm{Type}_{2}^{\mathrm{data}}, \cdots, \mathrm{Type}_{k_1}^{\mathrm{data}}\right\}_{m,i} \subseteq \left\{\mathrm{Type}_{1}^{\mathrm{data}}, \mathrm{Type}_{2}^{\mathrm{data}}, \cdots, \mathrm{Type}_{k_2}^{\mathrm{data}}\right\}_{j}; \tag{13}$$

$$\left[\tau_{i,j}^{b},\ \tau_{i,j}^{e}\right] \subseteq \left[T_{m}^{b}, T_{m}^{e}\right]; \tag{14}$$

$$E_{i,j} \geqslant E_{m}^{\min}, \tau_{i,j}^{e} - \tau_{i,j}^{b} \geqslant WL_{m}^{\min}, TS_{i,j} \leqslant TS_{m}^{\min}。 \tag{15}$$

其中：式 (7) 表示重调度前后调度方案中的监视窗口总数关系；式 (8)～(9) 表示可见性约束，即监视窗口必须在几何可见时间窗口之内，且光学观测时光学设备必须在地影内、空间目标必须在阳光照射下；式 (10)～(12) 分别表示资源的工作时段、观测能力和多目标能力约束；式 (13)～(15) 分别表示监视任务的测量数据类型、执行时段和观测条件约束。

2　基于蚁群算法的重调度算法

蚁群算法(ant colony optimization，ACO)是意大利学者 M. Dorigo 等[15]提出的一种模拟蚂蚁群觅食时采用信息素进行正反馈信息交换的行为的群智能算法，目前已在多个领域得到了应用。算法中设置了一个具有一定规模的人工蚁群，其中每只蚂蚁都可以根据启发式信息和根据之前构建的可行解而更新的信息素矩阵来寻找新的可行解；而较优秀的解在接下来的信息素更新中在其经过的路径上释放更多的信息素；通过信息素的累积和挥发，在正反馈作用下控制蚂蚁协同找出高质量的解。笔者利用蚁群算法的并行性和全局寻优特点，来解决空间目标地基监视重调度的规模大、算法运行效率要求高的问题。

2.1　启发式信息

对每一个初始需求或中间需求(包含指定的空间目标、任务和相关约束条件，未指定监视资源)，可能存在多个长短不一的可用监视窗口，如何选择或截取一个合适的监视窗口，在满足监视需求的同时不再替换出初始方案中的窗口，或在替换窗口后使新产生的需求尽快实现无替换地满足，是启发式信息应该考虑的问题。文中采用如下的启发式信息评价备用窗口或长窗口的备选区段：

$$p_{z_1}^{\mathrm{RES}} = \frac{\sum_{l_1}\gamma_{z_1,l_1} + \sum_{l_1}\left(\sum_{k_1}\eta_{z_1,l_1,k_1} \Big/ \gamma_{z_1,l_1}\right)}{\varphi_{z_1}\xi_{z_1}}。 \tag{16}$$

式中：$p_{z_1}^{\mathrm{RES}}$ 表示当前需求的第 z_1 个备用窗口或当前需求的某个较长(大于最大窗口长度约束)的备用窗口中第 z_1 个可用区段的评价值；γ_{z_1,l_1} 表示与该备用窗口或区段冲突的第 l_1 个窗口所关联任务和目标对应的备用窗口数；η_{z_1,l_1,k_1} 表示第 l_1 个冲突窗口的第 k_1 个备用窗口或区段的资源多目标能力；φ_{z_1} 表示当前需求的第 z_1 个备用窗口或区段的冲突数；ξ_{z_1} 则为选择备用窗口 z_1 到形成下一个中间需求之前所产生的扰动量。

2.2　转移概率

蚂蚁在满足当前需求时面临多个备用窗口的选择和长窗口的截取问题，同时也是需求的转移或消除问题，因此转移概率为：

$$P_{z_1}^{\mathrm{Trans}}(t) = \frac{\left[\tau_{z_1}(t)\right]^{\alpha_{\mathrm{A}}} \cdot \left[p_{z_1}^{\mathrm{RES}}(t)\right]^{\beta_{\mathrm{A}}}}{\sum_{z_1 \in N_{rw}} \left[\tau_{z_1}(t)\right]^{\alpha_{\mathrm{A}}} \cdot \left[p_{z_1}^{\mathrm{RES}}(t)\right]^{\beta_{\mathrm{A}}}}; \tag{17}$$

$$\tau_{z_1}(t) = \frac{\sum_{k_0}\tau_{z_1}^{k_0}}{\Delta T_{z_1}}。 \tag{18}$$

式中：$P_{z_1}^{\mathrm{Trans}}(t)$ 表示当前需求的第 z_1 个备用窗口或某长窗第 z_1 个可用区段的转移概率；$\tau_{z_1}(t)$ 为该备用窗口或区段在信息素矩阵中的信息素，其中区段选择时采用整个区段的平均信息素(式 (17))；$\tau_{z_1}^{k_0}$ 是该区段中 k_0 时刻的信息素值；ΔT_{z_1} 是该区段时长；N_{rw} 是当前需求的备用窗口数或者长窗口的可用区段数；α_{A} 和 β_{A} 为信息素和启发函数重要程度因子。

2.3　信息素的更新

由于使用了 2 个信息素矩阵(以所属的空间目标 i、监视任务 m 和监视资源 j 为坐标的备用窗选择矩阵，和以超过最大窗口长度约束的较长备用窗口中的时间 k_0 为坐标的区段选择矩阵)，故更新信息阵的模型也分为 2 种，分别见式 (19)～式 (21) 和

式 (22)～式 (23)。

$$\tau_{i,m,j}(t+1)=(1-\rho)\tau_{i,m,j}(t)+\Delta\tau_{i,m,j}\ ; \quad (19)$$

$$\Delta\tau_{i,m,j}=\sum_{k_1=1}^{n}\sum_{k_2=1}^{n_{k_1}}\Delta\tau_{i,m,j}^{k_1,k_2}\ ; \quad (20)$$

$$\tau_{i,m,j}^{k_0}(t+1)=(1-\rho)\tau_{i,m,j}^{k_0}(t)+\Delta\tau_{i,m,j}^{k_0}\ ; \quad (21)$$

$$\Delta\tau_{i,m,j}^{k_0}=\sum_{k_1=1}^{n}\sum_{k_2=1}^{n_{k_1}}\Delta\tau_{i,m,j}^{k_0,k_1,k_2}\ ; \quad (22)$$

$$\Delta\tau_{i,m,j}^{k_1,k_2}=\Delta\tau_{i,m,j}^{k_0,k_1,k_2}=Q_{\mathrm{A}}/L_{k_1}\ 。 \quad (23)$$

式中：$\tau_{i,m,j}(t+1)$表示第 t+1 次迭代时目标 i、任务 m 及资源 j 所对应的信息素矩阵中的信息素值；ρ 为信息素挥发系数；$\Delta\tau_{i,m,j}$ 是信息素增量，由式 (20) 定义，其中 $\Delta\tau_{i,m,j}^{k_1,k_2}$ 表示第 k_1 只蚂蚁在为从属于目标 i 和任务 m 的第 k_2 个需求选择来自资源 j 的窗口时留下的信息素浓度；k_0 表示某长窗口中的时刻；Q_{A} 是信息素增强系数；L_{k_1} 为第 k_1 只蚂蚁的目标函数评价值。

2.4 算法步骤

基于上述分析，笔者设计了蚁群算法，其步骤如下：

1) 初始化。对重调度初始时刻的监视需求 R_1、初始方案 P_1 和蚁群算法参数 Q_{A}、ρ、α_{A}、β_{A} 以及蚁群规模 m_{A}、最大迭代次数 $T_{\mathrm{A}}^{\max}$、信息矩阵初值 $\tau_{i,m,j}(0)$、迭代次数初值 $T_{\mathrm{A}}^{\mathrm{iter}}=1$。

2) 构建解空间。每只蚂蚁采用邻域搜索方式，迭代搜索可行解。

① 迭代搜索可行解。从初始需求开始，对于每个需求的时长大于最大窗口长度约束的可用备用窗口，根据式 (17) 和式 (18) 选择一个区段作为该可用备用窗口的实际备用窗口。对于存在多个备用窗口的需求，根据式 (17) 的评价，概率选择一个窗口作为满足该需求的窗口，如果该窗口在初始方案中没有冲突窗口，则停止该路径或分支的搜索；如果有冲突窗口，则删除初始方案中与该窗口冲突的窗口，并将删除的窗口转为中间需求进入下一次迭代中。迭代中允许搜索路径交叉，如出现对本路径的重复则停止该路径搜索。

② 邻域生成。上述迭代完成后，最终形成一个存在部分需求未满足的可行解，假设每条搜索路径的每个节点(增加的窗口)为该路径终点，生成一个该可行解的扰动相对较小的可行解邻域。

③ 邻域搜索。对上述邻域，采用贪婪方法，根据式 (1) 搜索该邻域内的最优解，并以之作为当前蚂蚁得到的可行解。

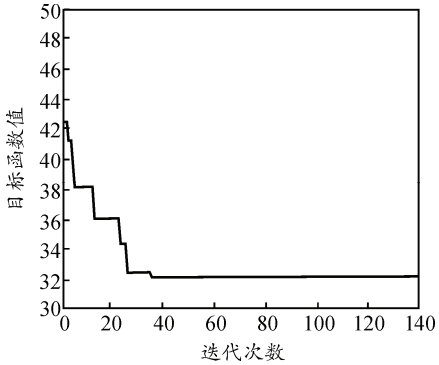
3) 更新信息素。计算各蚂蚁获得的可行解的目标函数值，记录本次迭代的最优解，并根据式 (19)～式 (23) 对信息素矩阵进行更新。

4) 如果 $T_{\mathrm{A}}^{\mathrm{iter}}<T_{\mathrm{A}}^{\max}$，则令 $T_{\mathrm{A}}^{\mathrm{iter}}=T_{\mathrm{A}}^{\mathrm{iter}}+1$，清空各蚂蚁获得的可行解，返回步骤 2)；否则停止迭代，输出最优解。

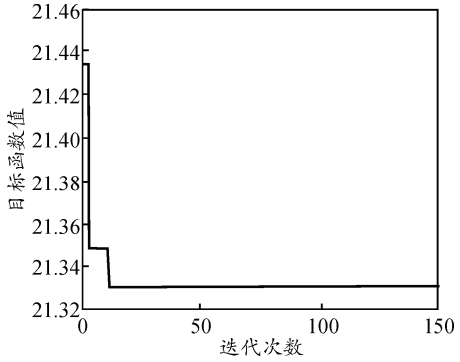
3 仿真实验与分析

为了验证重调度算法的有效性，设计一个包含 23 个观测资源、8 146 个独立空间目标和 6 个计划监视任务(其中支持监视任务 3 个、重点监视任务 1 个、编目监视任务 1 个、空域搜索任务 1 个)，调度周期为 24 h 的场景，通过运行静态调度算法得到一个静态调度方案。假设检测到监视资源故障发生后直接启动重调度算法，经过预处理得到初始需求和相关备用窗口集，并设置 2 个算例，分别为光学望远镜和机械雷达故障，时长为 12、5.5 h，初始需求数为 56、24。基于蚁群算法的重调度算法(ACO-RA)在配置为 Intel(R) Core i5-4440 CPU @ 3.10 GHz，内存 8 GB，操作系统为 Windows 7(64 位)的多个平台上，采用 Matlab 软件进行 Monte Carlo 并行仿真实现。相关参数设置为 α=0.8、ρ=0.1、Q_{A}=100、α_{A}=1、α_{B}=5、$T_{\mathrm{A}}^{\max}$=150、m_{A}=25，实验结果如图 1 所示，并将最终结果和基于邻域搜索的启发式方法(HA，使用式 (17) 的启发式和邻域搜索方法，计算 50 次求平均值)获得的结果进行比较，如表 1 所示。

从图 1 可知：ACO-RA 能快速收敛，且在算例 1(光学望远镜故障)中优化效果较为明显。由于光学监视资源主要用于监视深空目标，资源相对紧张，故仿真结果表明，ACO-RA 在资源紧张时可快速获得优化重调度方案。上述 2 个算例中，ACO-RA 的 100 次迭代平均运行时间分别为 15.48 min 和 8.59 s。



(a) 算例 1



(b) 算例 2

图 1　仿真结果

通过表 1 的对比结果可知：ACO-RA 在算例 1 运行的结果在目标函数、失败需求数和扰动 3 个方面均优于 HA 获得的结果。在算例 2 中，由于资源相对富余和初始需求较少，故 ACO-RA 相对 HA 运行结果在目标函数和失败需求数上差距不大；但 ACO-RA 比 HA 多满足的一个监视需求所产生的扰动值较大，这是因为目标函数中的 $\alpha = 0.8$，失败需求的综合优先级占比较高，扰动考虑得相对较少，故出现了上述情况。另外，ACO-RA 的运行时间基本是 HA 的运行时间与迭代次数的乘积，但仍在可接受范围内，故文中所设计算法可行。

表 1　仿真结果对比

算例	目标函数值			失败需求数			扰动 $(\lambda_1\xi_1+\lambda_2\xi_2)(1+\lambda_3\xi_3)$	
	初始	HA	ACO-RA	初始	HA	ACO-RA	HA	ACO-RA
1	260.73	59.66	32.21	56	10	5	222.15	75.08
2	294.25	21.47	21.33	24	1	0	90.14	106.66

4　结束语

资源故障或资源因天气、供电而失效是空间目标地基监视中的一类主要不确定因素。笔者用邻域搜索对每只蚂蚁获得的可行解进行局部最优化，再通过蚁群的全局寻优能力进行全局最优化。并同时更新 2 个信息阵，分别用于对长窗口的截取和备用窗口的选择，以充分利用蚁群算法的正反馈特性，增强全局寻优能力。仿真实验结果表明：基于蚁群算法的重调度算法能够在可接受的时间内收敛，且其解的质量相对启发式方法有明显提升，证明了算法的有效性和可行性。

参考文献：

[1] Hobson T A, Clarkson I V L. Collaborative sensor scheduling for space situational awareness[C]//Statistical Signal Processing Workshop (SSP), 2012 IEEE, Ann Arbor, MI: IEEE, 2012: 640-643.

[2] Berger J M, Moles J B, Wilsey D G. An Analysis of USSPACECOM's: Space Surveillance Network (SSN) Sensor Tasking Methodology[R]. Dayton, Ohio, US: Air Force Inst Of Tech Wright-Patterson AFB Oh School Of Engineering, 1992.

[3] Miller J G G. A new sensor allocation algorithm for the Space Surveillance Network[J]. Military Operations Research, 2007, 12(1): 57-70.

[4] Cherry P R. Sensor Tasking by the Space Defense Operations Center[C]//Proceedings of the Space Surveillance Workshop. Lexington, MA: Massachusetts Inst. of Technology Lincoln Lab., 1993: 93-98.

[5] Petrick B L. Weighting Scheme for the Space Surveillance Network Automated Tasker[R]. Dayton, Ohio, US: AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH, 1994.

[6] Vieira G E, Herrmann J W, Lin E. Rescheduling manufacturing systems: a framework of strategies, policies, and methods[J]. Journal of scheduling, 2003, 6(1): 39-62.

[7] Abumaizar R J, Svestka J A. Rescheduling job shops under random disruptions[J]. International Journal of Production Research, 1997, 35(7): 2065-2082.

[8] 刘乐, 周泓. 单一机器干扰下的开放式车间重调度[J]. 计算机集成制造系统, 2013, 19(10): 2467-2480.

[9] Vieira G E, Herrmann J W, Lin E. Rescheduling manufacturing systems: a framework of strategies, policies, and methods[J]. Journal of scheduling, 2003, 6(1): 39-62.

[10] Aytug H, Lawley M A, McKay K, et al. Executing production schedules in the face of uncertainties: A review and some future directions[J]. European Journal of Operational Research, 2005, 161(1): 86-110.

[11] Ouelhadj D, Petrovic S. A survey of dynamic scheduling in manufacturing systems[J]. Journal of Scheduling, 2009, 12(4): 417-431.

[12] 聂黎. 基于基因表达式编程的车间动态调度方法研究[D]. 武汉: 华中科技大学, 2011: 8-13.

[13] Erwin R S, Albuquerque P, Jayaweera S K, et al. Dynamic sensor tasking for space situational awareness[C]//2010 American Control Conference (ACC), Baltimore: IEEE, 2010: 1153-1158.

[14] Herz A F, Stoner F, Hall R, et al. SSA Sensor Tasking Approach for Improved Orbit Determination Accuracies and More Efficient Use of Ground Assets[C]//The 2013 Advanced Maui Optical and Space Surveillance Technologies (AMOS) Conference, Maui: The Maui Economic Development Board, 2013: 1-10.

[15] Dorigo M, Caro G D. The ant colony optimization metaheuristic. New ideas in optimization[J]. Ideas in Optimization, 1999: 11-32.