第56卷第4期
2016年7月

大连理工大学学报
Journal of Dalian University of Technology

Vol. 56, No. 4
July 2016

文章编号：1000-8608(2016)04-0367-08

基于协同进化和并行计算的船舶管路布置方法

董宗然[1]，林　焰[*2]

（1.大连理工大学 电子信息与电气工程学部，辽宁 大连　116024；
2.大连理工大学 运载工程与力学学部，辽宁 大连　116024）

摘要： 为解决船舶管路协同布置问题，提出一种适合求解多管路或分支管路协同布置的算法框架.通过为每条管路或分支生成对应的进化种群，将管路间的协同布置转换为种群间的协同进化.基于提出的路径连接点概念，生成管路接口间的候选路径种群，并对种群进行交叉、变异操作.使用 A^* 算法作为寻路算子，提高了生成路径的质量，同时保证了路径的有效性.为了提高运算效率，引入并行计算策略对算法框架和 A^* 算法进行改进.最后，两个仿真实例验证了所提出方法的可行性和有效性.

关键词： 船舶管路；A^* 算法；协同进化；并行计算

中图分类号： U662.9　　**文献标识码：** A　　**doi：** 10.7511/dllgxb201604007

0 引　言

船舶管路设计是船舶详细设计阶段中最繁杂的工作之一.现代船舶需要大量的管路以完成油、气、水、电等的输送.管路需要连接指定的设备接口，同时需要满足诸多约束条件，包括管路长度、弯头数目等经济性约束，还包括易安装性、易维护性、美观性、安全性等高级需求.船舶设计具有非批量的特点，除了少量姐妹船以外，每条船都有各自的设计需求.因此，船舶管路设计的工作量巨大，难度较高.当前船舶管路设计主要由管路设计师完成，除了要考虑管路的连接性和相关约束外，当布置多条管路和带分支管路时，还要考虑管路间的协同，以使管路整体布置最优.研究船舶管路自动布置方法，可以减轻管路设计人员的工作量，实现管路设计方案的标准化，缩短船舶生产的交付周期.

近年来管路布置问题逐渐得到研究者的关注，出现在工业设计的不同领域，包括船舶管路设计、机械管路设计、航天器管路设计等.在使用的优化技术方面，可以分为确定性算法和非确定性算法.确定性算法在初始状态相同的情况下具有结果的可重复性和运算时间的确定性.基于Dijkstra[1]、最短路径快速算法[2]、A^* 算法[3]、直线探索法[4]的管路布置研究均归属于确定性算法.非确定性算法依赖控制参数设置，寻优过程依赖随机策略，结果不具有可重复性，而且算法收敛时间也不确定.采用群进化技术的管路布置方法可归属于非确定性算法[5-12].其中，Jiang 等[9]将遗传算法的交叉和变异操作嵌入蚁群算法的运算流程以提高布管算法性能；在文献[10]中又提出将机舱设备布置与船舶管路布置相结合的优化思路，并使用粒子群算法和蚁群算法分别求解设备布置和管路布置.王运龙等[11-12]将爬山算法引入遗传算法的迭代过程，用以改进算法的收敛性，并基于人机结合思想设计布管算法，通过在算法不同阶段加入人工个体提高布置效果.以上方法都使用了不止一种优化策略，但仍然属于非确定性方法.确定性算法和非确定性算法各具优势，将两者有机结合可以发挥确定性算法的高效率和非确定性算法的全局寻优能力，Asmara 等在文献[13]中提出利用确定性的 Dijkstra 算法完成管路生成，利用非确定性的粒子群算法优化布管序列，该方法从整体上考虑管路布置顺序对结果的影

收稿日期： 2015-11-30；　**修回日期：** 2016-01-30.
基金项目： 国家自然科学基金资助项目(51209034)；国家公益性行业科研专项(201003024)；辽宁省教育厅科研项目(L2012018).
作者简介： 董宗然(1981-)，男，博士生，E-mail：dongzongran@163.com；林　焰*(1963-)，男，教授，博士生导师，E-mail：linyanly@dlut.edu.cn.

响,但其顺序优化和管路布置分别由不同模块实现,不满足管路间的协同设计要求.范小宁等在先期工作中将每条管路的布置对应一个蚂蚁种群的路径寻优,利用多蚁群间的协作来实现管路间的协同敷设,该方法的主要不足是蚁群算法存在局部搜索能力较弱和收敛速度较慢等问题,且求得的路径存在较多折弯[5-6].本文对范小宁工作做出几点重要改进,包括提出路径中间连接点概念,并基于确定性 A* 算法生成候选路径,设计协同布置算法框架统一处理多管路和带分支管路的布置,以及对算法和框架做并行化改进以提升布管效率等.

1 空间模型及布管约束

布管问题本质上类似于静态环境中的路径寻优,首先要对布置空间做预处理,采用的经典方法是网格分解法[1-3,5-8,11-13].网格分解法先将布置空间分解成网格集合,将设备、过道、维修空间等占据的网格标记为障碍网格,障碍网格禁止管路通过;将布管需要满足的几何约束转化为网格能量值,即将期望管路经过区域(舱壁附近、可依附设备周围、希望成束敷设的管路周围)的网格设置为低能量值,将希望管路远离布置区域(如锅炉周围、配电设备上方)的网格设置为高能量值,进而管路对应的能量值可描述其对布置空间几何约束的适应情况.网格分解法的优势在于方便寻路算法的实现和布置约束的表达;劣势在于网格数目随着布置空间增大而增多,需占用大量内存,同时因为搜索空间变大,导致寻路算法性能下降.船舶管路的布置空间往往很大,如果不对空间做预处理和选择高效算法,网格分解法难以应用于实船管路布置.空间预处理先根据船舶结构将舱内空间分成不同工作区(各工作区内的布管任务相对独立),再对独立工作区空间进行网格分解完成自动布管,为了提高布管效率,还需要设计高效的网格寻路算法.

实船设计中不同管路系统的布置要求往往不同,本文主要讨论船舶管路布置的共性需求,包括:①管路与管路之间或是管路与设备之间不发生干涉;②管路能够连接到指定的设备接口点;③管路走势尽量与钢结构件平行;④管路长度尽可能短;⑤管路弯头数目尽可能少;⑥最小管段长度尽量满足加工限制以降低生产成本;⑦管路满足便于安装和维护的几何约束.其中①～③作为必须满足的需求可以由布管算法和网格分解模型保证,④～⑦作为期望满足的目标可以通过评价机制从众多可行解中选择优解满足.

本文采用寻路算法中最为高效的 A* 算法进行布管,所得路径可避开设备和已敷设管路,因此评价管路时不必再考虑管路穿过障碍物的惩罚值.基于以上模型和布管约束,管路敷设质量的评价函数(F)需包括管路长度(P_l),它是在管径确定时,影响管路用料成本的主要因素;管路折弯数目(P_b),它表示布管过程中使用的弯头个数和对管路进行弯曲加工的次数;管路对应的能量值(P_{po}),它量化了对管路安装位置优劣的评价;管路不满足最小管段长度约束的惩罚值(P_{pu}),它代表管路布置中违反管路加工工艺限制的程度.评价函数的形式化描述如下式:

$$\min F(x)=a\times P_l+b\times P_b+c\times P_{po}+d\times P_{pu} \quad (1)$$

式中:x 代表一条管路,a、b、c、d 为各代价的权重系数.以上目标函数的设计源自实际工程需求,主要考虑布管的经济成本,属于可量化、非硬性的优化目标.在无法满足全部约束和目标时,可适当牺牲经济性目标(例如增加管路长度和弯头数目或改进加工工艺等)以求找到可行布管方案.

目标函数权重系数选择具有一定的随机性和经验性,不同的系数设置方案会产生不同的优化结果.因此选择权重系数时可遵循以下方法,包括:①结合船型特点分析管路布置需求,根据各目标的重要程度设置初始权重系数;②估算各优化目标的数值范围,进一步调整权重系数,使经过权重系数调整后的各目标值处在可比较的数量级;③根据仿真结果的反馈,微调权重系数;④将有效的系数设置方案存入系统,积累经验数据,为将来的权重系数选择提供参考.

2 船舶管路协同布置方法

协同进化由 Potter 在 1994 年提出,它是自然界物种进化的一种现象.生物进化往往不是单一生物种群的进化过程,而是要受到同一生态环境中其他种群的影响.根据其他种群对当前进化种群影响的不同,协同进化可分为协作式协同进化和竞争式协同进化两种类型.协作式是指各种群会从其他种群的进化中受益,种群的进化促使与其他种群配合较好的个体有更大的生存机会;竞争式是指其他种群的进化会抑制当前种群的进化,当前种群必须做出相应的反应以便获得生

存[14]. 船舶管路设计规范要求按照管子尺寸，由粗到细依次布管，但仍然存在一些管径相同或接近的管路，此时布管顺序会影响布局结果，需要协同布置. 管路设计的目标是使整个管路系统布局最优，各管路更多是一种合作关系，因此本文采用协作式进化算法框架解决此类管路的协同布置问题.

2.1　管路协同布置算法框架

在布置实船管路时，对于接口点位置靠近、管路性质相似的多条管路，往往期望能够将其成束敷设，以方便管路的支撑和维护，且符合布置的整齐性[5]；对于一类等径带分支管路，其布置要使各分支接口点连接到指定设备，同时各分支管路的总长度最短，弯头数目最少，可以将此类管路的布置看作同时布置几条具有相同终止接口点的单管路，要求这些单管路间能够有尽可能多的重叠部分[6]. 文献[5]和[6]使用多蚁群协同进化算法求解多条管路和带分支管路的布置，验证了协同布置的有效性，主要不足是蚁群算法作为布管算法收敛速度较慢，得到管路折弯较多，且算法效果依赖较多的控制参数. 分析两类问题的区别和共性，提出一种求解两类问题的协作式进化算法框架，并将管路布置封装成可替换的模块，使其不再局限于特定算法. 同时，因为多核心处理器已经广泛应用，提出对算法做并行化改进以提升求解效率. 算法框架如图 1 所示.

以上框架统一了多管路和分支管路的算法布置流程，主要区别在于设置协同布置环境 W_i 时，需要根据管路类型采取不同策略. 对于多管路布置，为了引导管路并行成束敷设，需将其他种群的管路代表所在网格设置为障碍区，管路代表周围的网格设为低能量区；对于分支管路布置，为了能让管路分支间有更多的重合以使管路最短，需将其他种群的分支管路代表所在网格设置为低能区. 此外，在计算迭代解 S_{current} 的评价值时，多管路的长度计算包括各代表管路的长度总和，而分支管路的长度计算对分支间的重叠部分只统计一次.

2.2　管路种群的生成及进化

在两个连接点之间生成多条候选路径（即路径种群）是管路协同布置框架对管路生成算法的功能需求. 范小宁等[5]使用蚁群算法在构建路径时根据转移规则随机选择周围网格，因此每次找到的路径都会不同. 但 A* 算法属于确定性算法，当搜索方式确定之后，每次运行找到的两个连接点间的路径都相同. 因此提出路径中间连接点概念，将路径分成子段，把随机性引入路径生成过程，方法如下.

设 S 和 G 为某管路的两端网格连接点，假设在 S、G 之间生成的管路需要经过 T 个中间连接点，记为 $V_1, V_2, \cdots, V_{T-1}, V_T$. 则 S 到 G 的路径由 S-V_1, V_1-V_2, $\cdots$, V_{T-1}-V_T, V_T-G 构成的各子路径段组成. 如果中间连接点位置按启发式规则随机生成，连接点间的路径用 A* 算法生成，则可按需要生成 S 和 G 之间的多条候选路径.

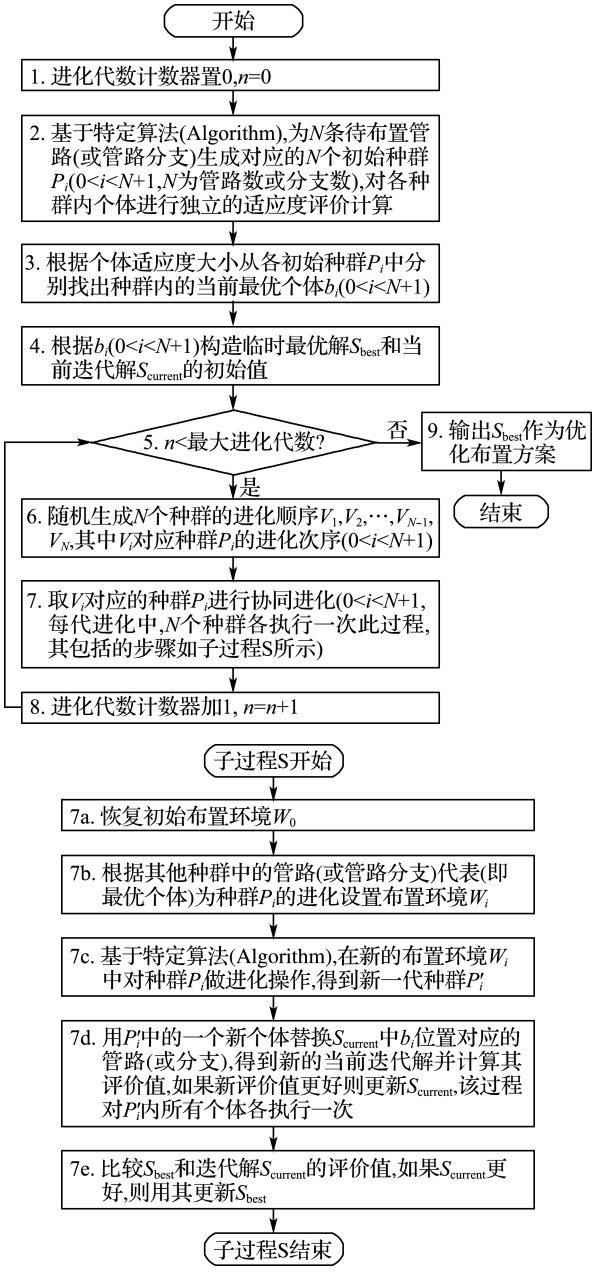

图 1　协同布管算法框架

Fig. 1　Algorithm framework for cooperative pipe routing

基于上述方法可生成各管路所对应的初始进化种群(步骤2),并在后续迭代进化过程中(步骤7c)对种群做进化操作,包括交叉和变异.交叉操作从种群中随机选择两条路径作为父代个体,随机产生一个1到T之间的序号,根据此序号交叉父代个体的中间连接点以形成两个新的子代个体.变异操作从种群中随机选择一条路径作为父代个体,随机产生一个1到T之间的序号,重新生成此序号位置上的连接点以形成一个新的子代个体.进行交叉和变异操作的个体是从父代中选出的,生成的子代个体将替换对应父代个体进入下一代进化,以此保证进化种群的稳定性和多样性.交叉和变异操作的示意如图2.子代路径中虚线表示的子段路径要用A*算法重新生成.

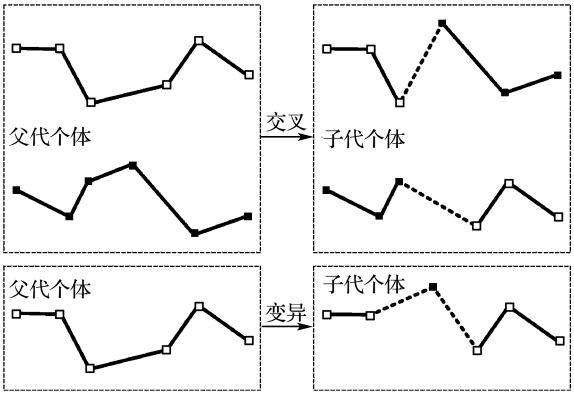

图2 交叉和变异

Fig. 2 Crossover and mutation

按启发式规则生成路径中间连接点可以提高初始路径种群的质量.提出如下启发规则:①连接点不能位于障碍物之内,以免产生非法路径;②在期望空间(包含路径两端连接点的外包络矩形空间)内产生连接点,避免路径向外过度延展;③协同布置多条管路时,按概率(R_{control})在已布置管路周围生成连接点,以引导管路成束布置;④协同布置分支管路时,按概率(R_{control})在已布置分支管路上生成连接点,以引导各分支路径更多重合.

2.3 种群进化的并行处理

从算法框架看出,步骤7c的任务是对种群做进化操作,这是计算代价最高的子任务.该任务需要在每代进化中为每一进化种群生成连接点间的候选网格路径,并按照评价函数计算每条路径的适应值.对该过程做并行化改进,可以提高算法效率.该策略简述为将总任务按CPU逻辑内核数目分解,使每个CPU内核都运转起来,完成部分计算任务,所有内核的计算任务完成对应总任务的完成.使用该并行策略的前提是子任务间不存在依赖关系,步骤7c中生成的尝试路径虽然有多条,但生成每条路径时基于的布置环境相同,生成各尝试路径的过程也不会改变布置环境,满足任务分解的前提条件.

当前主流CPU均为多核架构,要使各个逻辑内核并行工作,在实现上需借助多线程技术,例如,总的计算任务为生成两个管路连接点之间的M条候选路径,当前CPU逻辑内核数目为N,如果不使用多线程,那么所有计算将在1个内核上完成,而其他$N-1$个内核处在闲置状态,若将总任务分成N份,每份子任务为生成M/N条候选路径,创建N个计算线程对应N份任务,并分配到不同内核执行,则全部内核并行工作,完成总任务的时间会大为减少.随着多核并行编程技术的发展,可以借助OpenMp技术,它提供了对并行算法的高层抽象,通过在代码中加入pragma指明并行意图,编译器可以自动将程序并行化.借助OpenMp技术不需要人为创建线程、分配计算任务,优化了程序结构,基于OpenMp技术的候选路径生成及评价方法如下.

方法名:GenerateAndEvaluatePath_OMP

```
{
1.   #pragma omp parallel for
2.   for (i=0;i<P_size;i++)
     {
3.       t=::omp_get_thread_num( );
4.       GeneratePath(i,t);
5.       EvaluatePath(i);
     }
}
```

此方法根据当前CPU负载情况自动为每条路径的计算分配合适的内核资源.第1行为OpenMp指令,用于定义以下for循环为多线程并行处理代码块;第2行中P_{size}为要生成的候选路径条数,i为循环控制变量;第3行为OpenMp方法,从可用计算线程组中返回一个分配给此次计算任务的线程编号,如4核CPU,则t在0,1,2,3中取值,由系统根据各核计算负载状态分配;第4行为生成第i条路径的方法,该任务在第t号线程上执行,因为路径生成算法需要修改网格单元状态,而网格单元为线程间的共享资源,为了避免访问冲突,对算法用到的网格状态值按内核

数目生成副本，每个线程在自己对应的副本状态上做修改，进而提高了计算并行度，将线程号 t 传入方法，以决定路径生成时使用哪个副本；第 5 行为评价第 i 条路径的方法，该任务在第 t 号线程上执行.

2.4　基于 A* 算法的子段路径生成

框架图中的特定算法（Algorithm）用于生成管路接口间的候选路径，2.2 节方法通过引入中间连接点把原始路径生成分解成各子段路径的生成，所以生成子段路径的过程会被反复调用，生成子段路径的效率和质量对布管结果影响很大，本文选择 A* 算法实现子段路径生成.

2.4.1　*A* 算法原理及复杂度*　A* 算法最早由 Hart 等提出，其不是单纯的深度优先或广度优先搜索，而是基于最优优先策略的搜索[3]. A* 算法在每一步迭代搜索过程中会从当前顶点的邻接候选顶点中选择评价函数值最小的顶点 n，评价函数表示为 $f(n)=g(n)+h(n)$，其中 $g(n)$ 代表从路径起始顶点到任意一个顶点 n 的实际代价，$h(n)$ 代表从任意一个顶点 n 到达目标顶点的估计代价. A* 在搜索过程中不断更新 $g(n)$ 和 $h(n)$，因此每个网格路径点的选择考虑了已形成路径代价和剩余路径代价，使搜索更具目的性.

A* 算法与 Dijkstra 算法都是寻找最短路径的经典方法，复杂度同为 $O(|V|^2)$，V 为顶点数，因为在最坏情况下 A* 仍然需要访问所有顶点，但启发函数 $h(n)$ 会引导搜索向目标顶点有效进行，使实际搜索的顶点数比 Dijkstra 算法少很多. 当使用 Fibonacci 堆作为算法优先队列时，A* 算法复杂度可降低到 $O(V\log V)$，这是目前求解有向图单源最短路径的最好算法.

2.4.2　*基于 A* 算法的子段网格路径生成*　结合三维网格空间的数据结构和布管约束，将使用 A* 算法生成两点间网格路径的过程说明如下.

（1）数据结构

空间网格的数据结构包括当前网格点标号 C_{id}，前继网格点标号 P_{id}，网格中心点坐标（x，y，z），网格边长 L，网格所在行号 r、列号 c、层号 l，网格所在位置的势能 E，A* 算法记录网格点路径代价的变量 f、g、h.

两个表结构：Open 表和 Close 表，Open 表用于保存有待考察的网格点（即可能构成路径的网格点）；Close 表用于记录所有已访问过的网格点.

（2）算法流程图

A* 算法生成管路子段的流程如图 3 所示.

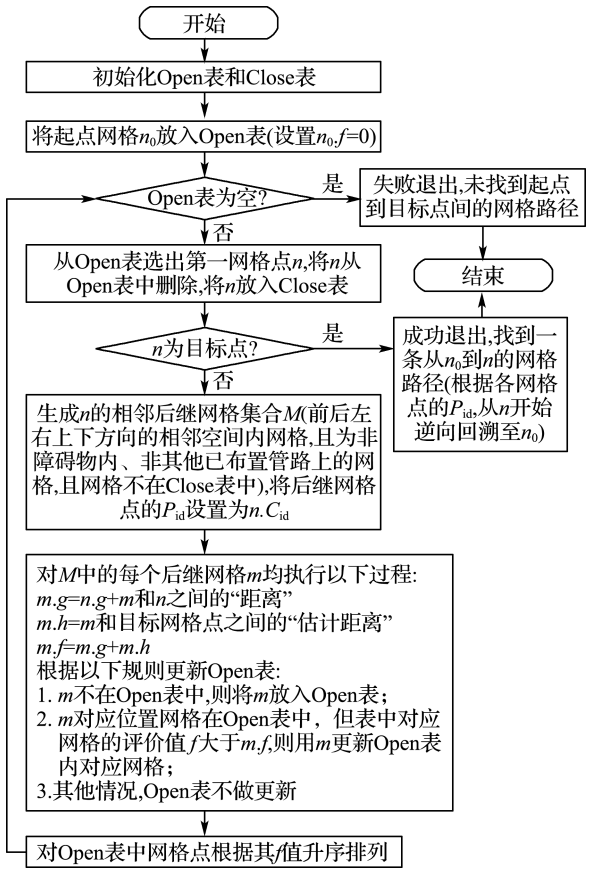

图 3　基于 A* 算法的子段路径生成

Fig. 3　Path segment generation based on A* algorithm

算法中的 Open 表为基于 Fibonacci 堆的优先队列. 其中，当前网格点 n 到 n_0 之间的已生成路径代价 g 不单纯表示管路长度，而是按路径评价函数（参见式(1)）计算出的综合代价，它考虑了 n 到 n_0 间的管路长度、弯头数目、管路对应的能量值和管路不满足最小管段长度约束的惩罚值；同理，m 和 n 之间的"距离"也代表了 m 与 n 之间的综合代价；m 和目标网格点（记为 t）之间的"估计距离"定义为网格点之间的 Manhattan 距离，该距离与从 m 到 t 需要沿行、列、层经过的网格数相关（只考虑水平或竖直方向，且忽略障碍物影响），m 和 t 之间的 Manhattan 距离计算如下式：

$$F_{\text{Manhattan}}(m,t)=(\text{abs}(t.r-m.r)+\text{abs}(t.c-m.c)+\text{abs}(t.l-m.l))\times L \tag{2}$$

式中：abs()为绝对值计算函数，L 为网格边长.

（3）算法并行化修改

2.3 节提到种群进化过程中路径生成需要支

持并行计算，其实质就是指此处的 A* 算法在同一时刻会运行在 N（N 为内核数）个线程上，网格作为线程间的共享资源，需要支持多线程并发访问，因此对算法中用到的网格状态值需按内核数目生成副本，使各线程在自己对应的副本状态上做修改，以提高运算并行度. 从流程图 3 看出，网格状态中的 P_{id}、f、g、h 需要支持并发修改，当算法支持多线程并发计算时，需对以上状态生成副本，使用 $P_{id}[t]$、$f[t]$、$g[t]$、$h[t]$（$0\leqslant t<N$）替换 P_{id}、f、g、h 参与运算，其中 t 为此次执行该算法的线程号.

2.5 算法成本分析

2.5.1 空间成本 算法内存需求主要包括：①存储布置空间的网格状态，A* 算法的寻路及路径评价需要在网格空间进行；②存储表示管路的各种群内个体，协同算法需要基于种群个体进行迭代. 因为网格空间划分及种群个体生成均在算法初始化阶段完成，所以算法运行时内存（M_{cost}）随时间变化波动不大，可基于下式估算：

$$M_{cost}=O(S_{grid}\times N_{grid}+S_{path}\times P_{size}\times N_{pipe}) \quad (3)$$

式中：S_{grid} 为网格结构大小，N_{grid} 为网格数目，S_{path} 为路径个体结构大小，P_{size} 为一个种群内的个体数目，N_{pipe} 为管路或分支对应的种群数目.

2.5.2 时间成本 群进化算法的时间复杂度可通过对种群的更新和评价次数来估计，因此本文算法的时间复杂度（T_{cost}）可以用下式估算：

$$T_{cost}=O((C_{evolve}+C_{evaluate})\times P_{size}\times N_{pipe}\times N_{iter}) \quad (4)$$

式中：C_{evolve} 为对路径个体执行进化操作的代价，$C_{evaluate}$ 为对协同解方案的评价代价，N_{iter} 为迭代次数.

3 仿真实验

通过两个实例验证提出方法的有效性. 实验机器配置：操作系统为 Windows 7 32 位，内存为 2 GB，CPU 为 Intel(R) Core(TM) i3 M390 2.67 GHz、支持双核 4 线程. 算法参数设置：最大迭代次数 $G_{max}=200$，种群规模 $P_{size}=80$，种群进化的交叉概率 $R_c=0.7$，变异概率 $R_m=0.3$，连接点控制概率 $R_{control}=0.5$，连接点个数 $N=3$，管路评价函数调节系数 $a=10$，$b=10$，$c=20$，$d=100$，最小管段长度为 2 倍网格边长.

3.1 多管路协同布置

3.1.1 模型空间 模型空间为一立方体，其中包括设备 $M_1\sim M_4$、虚拟障碍 $R_1\sim R_5$（R_1 用来控制管路避开工作空间中心位置，R_2、R_3 为设备 M_3 提供维修空间，R_4 为防止管路从 M_4 上方通过，R_5 为一禁止区）. 待布置管路有 3 条，管路 1、2 连接设备 M_2 和 M_3，管路 3 连接设备 M_1 和 M_3. 关于设备、障碍物、管路的更多信息参见文献[5].

3.1.2 布置结果 图 4 为本文方法迭代 15 次得到的一个布置结果. 协同算法使 3 条管路更好地成束敷设，且路径不与障碍物相交，管路之间不干涉，在满足管路安装约束和管段最小折弯长度约束的情况下，使管路长度尽可能短，折弯数目尽可能少. 未采用并行算法时，全部计算任务在一个内核上进行，进程的 CPU 利用率仅为 25%；采用并行算法时，线程在不同逻辑内核同时工作，进程的 CPU 利用率约为 75%. 统计 10 次独立计算结果，非并行算法平均耗时 315 s，并行算法平均耗时 156 s，采用并行布管算法可使效率提升约 50.5%，10 次计算中有 5 次收敛到实验最优解（实验最优解指实验过程中获得的满足布管约束的最好解）.

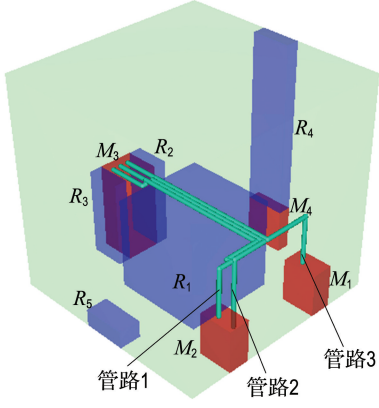

图 4 多管路协同布置

Fig. 4 The cooperative layout of multiple pipes

3.2 分支管路协同布置

3.2.1 模型空间 模型空间为一立方体，在其内部有 7 个立方体障碍物，关于布置空间和障碍物的详细信息请参见文献[6]，要求敷设一条具有 4 个连接点的分支管路，分支管路起点坐标为（4，10，10），终点坐标为（15，30，20），（32，38，15），（45，30，30）.

3.2.2 布置结果 本文方法迭代 15 次时的一个优化结果如图 5 所示. 由结果可以看出分支管路在绕开障碍物，弯头尽可能少的情况下，可以与其他分支管路重合敷设，使分支管路的总路径较短，

且分支管路能较多地沿物体表面敷设. 统计 10 次独立计算结果，非并行算法平均耗时 214 s，并行算法平均耗时 97 s，效率提升约 54.7%，10 次计算中有 6 次收敛到实验最优解.

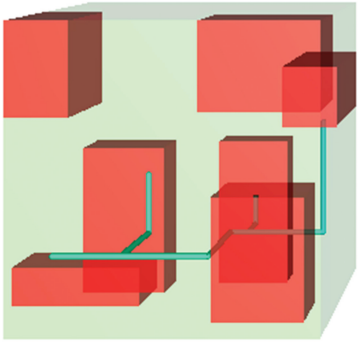

图 5　分支管路协同布置

Fig. 5　The cooperative layout of branch pipe

按分支路径重合思想敷设多分支管路，虽然不必在敷设前确定分支点的位置，但该方法可能形成实际布管中不允许出现的分支点形状，如图 6(a)所示，实际布管中要求使用 T 型分支点，如图 6(b)所示. 因此在评价带分支管路质量时，对存在不可行分支点的布置方案要加以惩罚(在式(1)中引入惩罚项)，使其不被选入迭代优解.

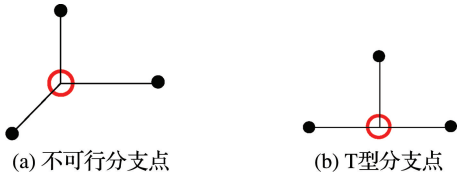

图 6　分支点

Fig. 6　Branch point

从实验结果看出，应用本文方法可以得到满足协同布置多管路或分支管路需求的布管方案. 统计多次实验，算法收敛到实验最优解的概率较高，使用并行策略改进算法能有效缩短布管时间. 验证了提出方法的可行性和有效性.

4　结　语

本文提出一种适合求解管路协同布置的算法框架，统一解决了多管路和带分支管路的协同敷设问题. 使用确定性 A* 算法生成管路路径，相比之前采用非确定性蚁群算法生成管路，布管质量更好，且算法参数更易配置. 结合协同布管问题特性，提出使用并行计算技术对协同进化过程和路径生成算法做并行化改进，使求解效率提升 50% 以上，说明将并行计算引入布管系统具有实用性. 仿真实例验证了应用本文方法可生成更为合理的管路协同布置方案.

后续研究包括：(1)将更高级的专家经验形成布管规则，引入自动化布管算法；(2)研究协同进化算法的调优理论，为改进算法框架和设置算法参数提供指导；(3)探索基于多主机网格系统的复杂管路并行优化.

参考文献：

[1] Ando Y, Kimura H. An automatic piping algorithm including elbows and bends [C] // **Proceedings of the 15th International Conference on Computer Applications in Shipbuilding**. Trieste: RINA, 2011: 153-158.

[2] 董宗然，林　焰. 基于最短路径快速算法的船舶管路自动敷设方法[J]. 计算机集成制造系统，2014，**20**(12)：2962-2972.
DONG Zong-ran, LIN Yan. Automatic ship pipe routing method based on shortest path faster algorithm [J]. **Computer Integrated Manufacturing Systems**, 2014, **20**(12): 2962-2972. (in Chinese)

[3] 李纯军. 基于 A* 算法的多管线通道化自动敷设方法研究[D]. 武汉：华中科技大学，2011.
LI Chun-jun. Study on the channelized auto-routing method of multi-pipeline based on A* algorithm [D]. Wuhan: Huazhong University of Science and Technology, 2011. (in Chinese)

[4] Kim S H, Ruy W S, Jang B S. The development of a practical pipe auto-routing system in a shipbuilding CAD environment using network optimization [J]. **International Journal of Naval Architecture and Ocean Engineering**, 2013, **5**(3): 468-477.

[5] 范小宁，林　焰，纪卓尚. 多蚁群协进化的船舶多管路并行布局优化[J]. 上海交通大学学报，2009，**43**(2)：193-197.
FAN Xiao-ning, LIN Yan, JI Zhuo-shang. Multi ant colony cooperative coevolution for optimization of ship multi pipe parallel routing [J]. **Journal of Shanghai Jiaotong University**, 2009, **43**(2): 193-197. (in Chinese)

[6] 邬　君，林　焰，纪卓尚，等. 基于协同进化的舰船分支管路系统路径优化研究[J]. 船海工程，2008，**37**(4)：135-138.
WU Jun, LIN Yan, JI Zhuo-shang, *et al*. Optimal

approach of ship branch pipe routing optimization based on co-evolutionary algorithm [J]. **Ship & Ocean Engineering**, 2008, **37** (4): 135-138. (in Chinese)

[7] LIU Qiang, WANG Cheng-en. Pipe-assembly approach for aero-engines by modified particle swarm optimization [J]. **Assembly Automation**, 2010, **30**(4):365-377.

[8] Kimura H. Automatic designing system for piping and instruments arrangement including branches of pipes [C] // **Proceedings of the 15th International Conference on Computer Applications in Shipbuilding**. Trieste:RINA, 2011:93-99.

[9] JIANG Wen-ying, LIN Yan, CHEN Ming, *et al*. An ant colony optimization-genetic algorithm approach for ship pipe route design [J]. **International Shipbuilding Progress**, 2014, **61**(3-4): 163-183.

[10] 姜文英,林 焰,陈 明,等. 基于粒子群和蚁群算法的船舶机舱规划方法[J]. 上海交通大学学报, 2014, **48**(4):502-507.
JIANG Wen-ying, LIN Yan, CHEN Ming, *et al*. An optimization approach based on particle swarm optimization and ant colony optimization for arrangement of marine engine room [J]. **Journal of Shanghai Jiaotong University**, 2014, **48** (4): 502-507. (in Chinese)

[11] 王运龙,王 晨,韩 洋,等. 船舶管路智能布局优化设计[J]. 上海交通大学学报, 2015, **49**(4):513-518.
WANG Yun-long, WAN Chen, HAN Yang, *et al*. Intelligent layout optimization design of ship pipe [J]. **Journal of Shanghai Jiaotong University**, 2015, **49**(4):513-518. (in Chinese)

[12] 王运龙,王 晨,彭 飞,等. 基于人机结合遗传算法的船舶管路布局优化设计[J]. 中国造船, 2015, **56**(1):196-202.
WANG Yun-long, WANG Chen, PENG Fei, *et al*. Intelligent layout design of ship pipes based on genetic algorithm with human-computer cooperation[J]. **Shipbuilding of China**, 2015, **56**(1):196-202. (in Chinese)

[13] Asmara A, Nienhuis U. Automatic piping system in ship [C] // **Proceedings of the 5th International Conference on Computer and IT Application**. Delft: COMPIT, 2006:269-279.

[14] Nielsen S S, Dorronsoro B, Danoy G, *et al*. Novel efficient asynchronous cooperative co-evolutionary multi-objective algorithms [C] // **2012 IEEE Congress on Evolutionary Computation**. Brisbane: IEEE Press, 2012:1-7.

Method of ship pipe routing based on co-evolution and parallel computing

DONG Zong-ran[1], LIN Yan[*2]

(1. Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116024, China;
2. Faculty of Vehicle Engineering and Mechanics, Dalian University of Technology, Dalian 116024, China)

Abstract: In order to solve the problem of cooperative pipe optimization in ship, an algorithm framework which is suitable for multiple pipes routing or branch pipe routing is presented. In this framework, evolutionary populations are created for the corresponding pipes or branches. Therefore, the cooperative arrangement of pipes or branches is transformed to the co-evolution of these populations. Based on the concept of connection points, the populations that represent the candidate paths between interface points are generated and gradually evolved through the operations like crossover and mutation. A^* algorithm is used as the path finding operator, and the quality and validity of the paths are assured. To speed up the pipe routing, parallel computing technology is adopted to improve the proposed framework and A^* algorithm. In the end, the feasibility and effectiveness of the proposed approach are verified by two simulation cases.

Key words: ship pipe; A^* algorithm; co-evolution; parallel computing