(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 111651905 B
(45)授权公告日 2024.11.05

(21)申请号 202010654910.7

(22)申请日 2020.07.09

(65)同一申请的已公布的文献号
申请公布号 CN 111651905 A

(43)申请公布日 2020.09.11

(73)专利权人 中国人民解放军国防科技大学
地址 410073 湖南省长沙市开福区德雅路109号

(72)发明人 刘晓路 彭观胜 何磊 沈大勇 吕济民 王术 王涛 张忠山 陈盈果 陈宇宁

(74)专利代理机构 长沙国科天河知识产权代理有限公司 43225
专利代理师 董惠文

(51)Int.Cl.
G06F 30/20(2020.01)
G06F 111/04(2020.01)
G06F 119/12(2020.01)

(56)对比文件
CN 104063748 A,2014.09.24
CN 107025363 A,2017.08.08

审查员 徐生芹

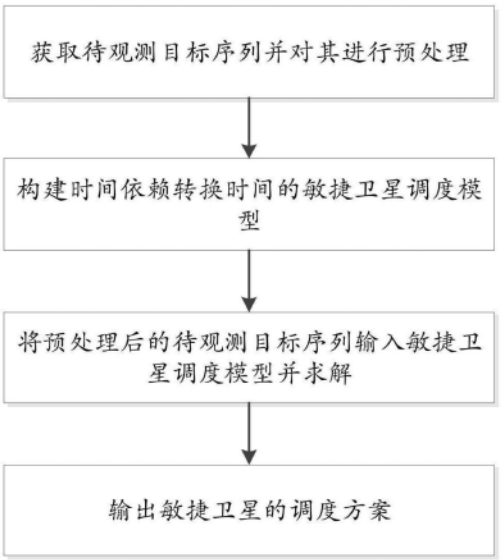
权利要求书4页 说明书15页 附图4页

(54)发明名称

考虑时间依赖转换时间的敏捷卫星调度方法

(57)摘要

本发明公开了一种考虑时间依赖转换时间的敏捷卫星调度方法,包括获取待观测目标序列并对其进行预处理;构建时间依赖转换时间的敏捷卫星调度模型;将预处理后的待观测目标序列输入敏捷卫星调度模型,并对敏捷调度模型进行求解;将步骤3中的求解结果输出,得到敏捷卫星的调度方案。由于考虑连续观测任务之间的姿态转换时间的时间依赖性,使所构建的敏捷卫星调度模型能更好地满足调度需求,所求解得到的调度方案其调度效率更高。在求解过程中,通过预计算每个时间窗口的不可达最早和最晚时间,以及同一圈次任意一对观测窗口间最早开始时间和最晚开始时间,在尝试进行插入操作时减少了多次重复计算,减少了计算时间,提高了调度效率。



CN 111651905 B

1.一种考虑时间依赖转换时间的敏捷卫星调度方法,其特征在于,包括以下步骤:

步骤1:获取待观测目标序列并对其进行预处理;

步骤2:构建时间依赖转换时间的敏捷卫星调度模型;

步骤3:将预处理后的待观测目标序列输入敏捷卫星调度模型,并对敏捷调度模型进行求解;

步骤4:将步骤3中的求解结果输出,得到敏捷卫星的调度方案;

所述敏捷卫星调度模型的构建方法是:

目标函数为:

$$Maximize \sum_{i\in T}\sum_{k\in O} p_i \cdot y_i^k \quad (1)$$

表示最大化调度任务的总收益,

其中:y_i^k为决策变量,为1时表示第i个待观测目标调度在圈次k上,否则为0,p_i表示第i个待观测目标的收益,T表示观测目标集合,O表示调度周期内的圈次集合;

约束条件为:

$$\sum_{k\in O} y_i^k \le 1, \forall i \in T \quad (2)$$

$$\sum_{\substack{j\in T\cup\{e\}\\ j\neq i}} x_{ij}^k = \sum_{\substack{j\in T\cup\{s\}\\ j\neq i}} x_{ji}^k = y_i^k, \forall i \in T, k \in O \quad (3)$$

$$\sum_{j\in T\cup\{e\}} x_{sj}^k = 1, \forall k \in O \quad (4)$$

$$\sum_{j\in T\cup\{s\}} x_{je}^k = 1, \forall k \in O \quad (5)$$

$$t_i^k + d_i + mintrans_{ij}^k(t_i^k) - t_j^k \le M\left(1 - x_{ij}^k\right), \forall i,j \in T,\ k \in O \quad (6)$$

$$st_i^k y_i^k \le t_i^k \le et_i^k y_i^k, \forall i,j \in T,\ k \in O \quad (7)$$

$$y_i^k \le b_i^k, \forall i,j \in T,\ k \in O \quad (8)$$

$$x_{ij}^k \in \{0,1\}, y_i^k \in \{0,1\}, \forall i,j \in T \cup \{s,e\},\ k \in O \quad (9)$$

约束(2)表示多圈唯一性约束,即每个观测目标在所有圈次内最多只能观测一次;

约束(3)代表流平衡约束,连接了决策变量x_{ij}^k和y_i^k;

x_{ij}^k为决策变量,为1时表示观测完i观测目标后接着观测j观测目标,否则为0,i,j为观

测目标索引,i,j∈T∪{s,e},其中{s,e}是指虚拟的起始目标和终止目标;

约束(4)和(5)规定了调度方案起始于虚拟目标s,终止于虚拟目标e;

约束(6)规定了连续两个观测之间的时间间隔不小于转换时间长度,i^k表示在圈次k对观测目标i的观测任务,t_i^k为整数决策变量,代表观测目标i在圈次k的观测开始时间,d_i表示观测目标i的观测持续时长,$mintrans_{ij}^k(t_i^k)$表示最小姿态转换时间,即在圈次k,给定观测目标i的观测开始时间t_i^k,下一个观测目标j的最早到达时刻对应的姿态转换时间,M表示最大正整数;

约束(7)代表了可见时间窗口约束,即每个任务的观测开始时间必须在其可见时间窗口范围内;st_i^k表示目标i在圈次k的可见时间窗口开始时间,et_i^k表示目标i在圈次k的可见时间窗口结束时间,$[st_i^k, et_i^k]$:目标i在圈次k的可见时间窗口时间范围;

约束(8)指观测目标只能调度到有其可见时间窗口的圈次上;b_i^k为二进制参数,为1时表示目标i在圈次k有可见时间窗,否则为0;

约束(9)代表了模型中决策变量的取值范围。

2.根据权利要求1所述的方法,其特征在于,步骤3中对敏捷卫星调度模型求解的方法是贪婪随机迭代局部搜索启发式算法。

3.根据权利要求2所述的方法,其特征在于,所述贪婪随机迭代局部搜索启发式算法的具体步骤是:

步骤3.1:预先计算同一圈次任意一对可见时间窗口VTW_{j-1}、VTW_j上,相对于前一个可见时间窗口上每一个观测开始时刻t_{j-1}^k后一个时间窗口上的最早开始时间es_j,以及相对于后一个可见时间窗口上每一个观测开始时刻t_j^k前一个时间窗口上的最晚开始时间ls_{j-1};

步骤3.2:设置贪婪随机迭代局部搜索算法中随机因子Greed的初始值为StartGreed,并初始化当前解为空;

步骤3.3:初始化内循环的迭代次数,初始化扰动因子参数向量S_d和R_d;

步骤3.4:调用扰动算子,在当前解位于圈次k的任务序列中从位置$S_d(k)$开始删除$R_d(k)$个连续的已调度任务,$S_d(k)$表示参数向量S_d的第k个元素,即移除子序列在圈次k的已调度任务序列中的起始位置,$R_d(k)$表示参数向量R_d的第k个元素,即圈次k的移除子序列规模大小;

步骤3.5:调用插入算子,依次将未调度的待观测目标尝试插入到当前解中,直至当前解无可行插入,对每个圈次k,计算该圈次上所有可见待观测目标i在当前圈次k的已调度任务序列所有可行插入位置,存入L_i^k中,若$|L_i^k| \geq 1$,则仅保留代价cost最小的插入位置,并将其存于L^k中,所述L^k用于存放圈次k上的所有未调度目标的可行插入,其中每个未调度目标最多只有一个可行插入;

步骤3.6:对L^k中所有插入按照对应观测目标的收益从大到小排序,仅保留前(1-Greed)·$|L^k|$个插入,采用轮盘赌的方式,随机选择一个插入并执行到当前解中,更新插入位置之后每个任务的最早开始时间,以及插入位置之前每个任务的最晚开始时间,然后返回步骤3.5,直至所有圈次找不到任何可行插入;

步骤3.7:计算新产生的解的总收益,如果新产生的解的总收益大于当前最好解,将新产生的解作为新的当前最好解,并将内循环迭代次数置为零,如果新产生的解比当前最好解差,则内循环迭代次数加1,若该次数超过最大连续迭代步数,则跳至步骤3.8,否则返回至步骤3.4,并令扰动因子参数向量$S_d=S_d+R_d$,$R_d=R_d+1$;

步骤3.8:令随机因子Greed=Greed-GreedDecrease,GreedDecrease为随机因子步长,若Greed仍大于或等于随机因子最小值StartGreed-GreedRange,GreedRange为随机因子波动范围,将当前最好解置为当前解,返回步骤3.2,否则,输出当前最好解。

4.根据权利要求3所述的方法,其特征在于:所述插入算子为:

步骤3.5.1:对当前解的每个圈次k,若当前圈次k无调度任务时,只有一个插入位置时,设该插入位置对应的最早开始时间为待插入的未调度目标i的可见时间窗口开始时间,其最晚开始时间为其可见时间窗口结束时间;

步骤3.5.2:当试图往当前序列各个位置插入未调度目标i时,

若插入位置为序列头,则令该插入的最早开始时间es_i为待插入的未调度目标i的可见时间窗口开始时间;

若插入位置为序列尾,则令该插入的最晚开始时间ls_i为待插入的未调度目标i的可见时间窗口结束时间;

若插入位置为j和j+1之间时,根据当前序列中待插入位置之前任务j的最早开始时间es_j计算出待观测目标i插入到该位置时对应的最早开始时间es_i,根据当前序列中待插入位置之后任务j+1的最晚开始时间ls_{j+1}计算出待观测目标i插入到该位置时对应的最晚开始时间ls_i,如果$es_i<ls_i$,则插入可行,否则插入不可行。

5.根据权利要求4所述的方法,其特征在于:在步骤3.1之前,还包括

步骤3.1':预计算对任意圈次k上的任意一个时间窗口VTW_i^k的"不可达"最早时间向量ue_i^k和"不可达"最晚时间向量ul_i^k,其中$ue_i^k(j)$表示从VTW_i^k出发不能抵达VTW_j^k时对应的VTW_i^k的最早观测开始时间,$ul_i^k(j)$表示在VTW_i^k内观测目标i后,无法将j调度在VTW_j^k内的最晚观测开始时间;

则在步骤3.5.1之前还包括步骤3.5.1':

如果尝试插入观测目标i于已调度任务j和j+1之间,

若j的观测开始时间t_j不早于其对应于i的不可达最早时间,即$t_j<ue_j(i)$,则该位置不可插入,且j+1之后的位置也不可插入,无需再做插入尝试;

若j+1的观测开始时间不晚于其对应于i的不可达最晚时间,即$t_{j+1}<ul_{j+1}(i)$,则该位置不可插入,且j之前的位置也不可插入,无需再做插入尝试;

若该插入位置满足不可达最早和最晚时间的要求,则转步骤3.5.1。

6.根据权利要求4所述的方法,其特征在于,步骤3.5中所述代价是指:

$$cost_i=trans_{ji}(es_j,es_i)+trans_{i(j+1)}(es_i,es'_{j+1})-trans_{j(j+1)}(es_j,es_{j+1})$$

其中,es_j表示任务j的最早开始时间,es_i表示待观测目标i的最早开始时间,es'_{j+1}代表更新后的任务j+1的最早开始时间,i为待观测目标,$cost_i$表示待观测目标i插入到位置j与j+1之间的代价。

7.根据权利要求3所述的方法,其特征在于:步骤3.5还包括,对所有圈次都执行完一次

插入操作后,检查是否存在某个目标被调度到多个圈次的任务序列中,如果存在,则执行分配步骤,根据预先给定的分配策略将该目标分配到其中一个圈次的任务,并删除其余圈次对应的已调度任务。

8.根据权利要求7所述的方法,其特征在于:所述预先给定的分配策略是指将待观测目标优先分配到总时间窗口数较少的圈次上。

9.一种考虑时间依赖转换时间的敏捷卫星调度系统,其特征在于:包括处理器,以及与所述处理器连接的存储器,所述存储器存储有考虑时间依赖转换时间的敏捷卫星调度方法的程序,所述程序执行时实现上述权利要求1~8任一项所述方法的步骤。

考虑时间依赖转换时间的敏捷卫星调度方法

技术领域

[0001] 本发明涉及卫星调度技术领域,尤其涉及一种考虑时间依赖转换时间的敏捷卫星调度方法。

背景技术

[0002] 对地观测卫星(EOSs)是能够根据各种观测需求获取地球表面特定区域图像的空间平台系统。随着科技与经济的发展,EOSs已经被应用于许多经济社会的各方各面,诸如自然资源勘探、灾难预警、气候变化分析等等。考虑到发射卫星的不菲成本,高效可靠的EOSs的调度方法对提高卫星观测系统性能起着非常关键的作用。对地观测卫星的调度,是指给定一组不同收益值的待观测目标,从中选出部分目标进行调度,生成成像调度方案,从而最大化观测收益,同时满足卫星能力约束。成像调度方案,即是对于每个被调度的观测目标,确定其观测开始时间。为区别起见,将确定了观测开始时间的一次观测活动定义为任务。

[0003] 一般来说,传统的EOS只具备侧摆能力,卫星只能在待观测目标的上空拍摄图像,任意两个目标之间的姿态转换时间可提前计算,因此对EOS的调度问题相对简单,可高效完成调度任务。但这也意味着,对于分布紧密的多个观测目标,EOS系统的观测能力有限。相比而言,作为新一代对地观测系统,敏捷对地观测卫星(AEOSs)具备有侧摆、俯仰、偏航三个角度的机动能力,使得卫星在到达目标上空之前或之后都能拍摄图像,如图1所示,每个目标可在较长的时间窗口内进行观测,因此,卫星对于分布紧密的多个目标观测能力大幅提高,在给定时间内能观测到更多的任务。同时,这种敏捷机动能力造成了目标之间的转换时间以及目标的观测开始时间不固定,这极大地提高了其调度的问题空间和复杂程度。此外,两个连续观测任务之间需要的姿态转换时间取决于这两个任务的观测姿态角,也即是观测开始时间,也称为时间依赖的转换时间。这种时间依赖特性也对任务调度算法有了更高的性能要求。

[0004] 现有的对于敏捷卫星任务规划算法大多都忽视了转换时间的时间依赖特性,或者对时间依赖特性缺乏清晰的建模与分析,导致了调度效率不高的问题。

发明内容

[0005] 本发明所要解决的技术问题是怎样提高敏捷卫星的调度效率,提出了一种考虑时间依赖转换时间的敏捷卫星调度方法。

[0006] 为解决该问题,本发明所采用的技术方案是:

[0007] 一种考虑时间依赖转换时间的敏捷卫星调度方法,包括以下步骤:

[0008] 步骤1:获取待观测目标序列并对其进行预处理;

[0009] 步骤2:构建时间依赖转换时间的敏捷卫星调度模型;

[0010] 步骤3:将预处理后的待观测目标序列输入敏捷卫星调度模型,并对敏捷调度模型进行求解;

[0011] 步骤4:将步骤3中的求解结果输出,得到敏捷卫星的调度方案。

[0012] 进一步地,所述敏捷卫星调度模型的构建方法是:

[0013] 目标函数为:

[0014] $$Maximize \sum_{i\in T}\sum_{k\in O} p_i \cdot y_i^k \quad (1)$$

[0015] 表示最大化调度任务的总收益,

[0016] 其中:y_i^k为决策变量,为1时表示第i个待观测目标调度在圈次k上,否则为0,p_i表示第i个待观测目标的收益,T表示观测目标集合,O表示调度周期内的圈次集合;

[0017] 约束条件为:

[0018] $$\sum_{k\in O} y_i^k \le 1, \forall i \in T \quad (2)$$

[0019] $$\sum_{\substack{j\in T\cup\{e\}\\ j\neq i}} x_{ij}^k = \sum_{\substack{j\in T\cup\{s\}\\ j\neq i}} x_{ji}^k = y_i^k, \forall i \in T, k \in O \quad (3)$$

[0020] $$\sum_{j\in T\cup\{e\}} x_{sj}^k = 1, \forall k \in O \quad (4)$$

[0021] $$\sum_{j\in T\cup\{s\}} x_{je}^k = 1, \forall k \in O \quad (5)$$

[0022] $$t_i^k + d_i + mintrans_{ij}^k\left(t_i^k\right) - t_j^k \le M\left(1 - x_{ij}^k\right), \forall i,j \in T,\ k \in O \quad (6)$$

[0023] $$st_i^k y_i^k \le t_i^k \le et_i^k y_i^k, \forall i,j \in T,\ k \in O \quad (7)$$

[0024] $$y_i^k \le b_i^k, \forall i,j \in T,\ k \in O \quad (8)$$

[0025] $$x_{ij}^k \in \{0,1\}, y_i^k \in \{0,1\}, \forall i,j \in T \cup \{s,e\},\ k \in O \quad (9)$$

[0026] 约束(2)表示多圈唯一性约束,即每个观测目标在所有圈次内最多只能观测一次;

[0027] 约束(3)代表流平衡约束,连接了决策变量x_{ij}^k和y_i^k;

[0028] x_{ij}^k为决策变量,为1时表示观测完i观测目标后接着观测j观测目标,否则为0,i,j为观测目标索引,i,j∈T∪{s,e},其中{s,e}是指虚拟的起始目标和终止目标;

[0029] 约束(4)和(5)规定了调度方案起始于虚拟目标s,终止于虚拟目标e;

[0030] 约束(6)规定了连续两个观测之间的时间间隔不小于转换时间长度,i^k表示在圈次k对观测目标i的观测任务,t_i^k为整数决策变量,代表观测目标i在圈次k的观测开始时间,d_i表示观测目标i的观测持续时长,$mintrans_{ij}^k\left(t_i^k\right)$表示最小姿态转换时间,即在圈次k,给定观测目标i的观测开始时间t_i^k,下一个观测目标j的最早到达时刻对应的姿态转换时间,M表示最大正整数;

[0031] 约束(7)代表了可见时间窗口约束,即每个任务的观测开始时间必须在其可见时

间窗口范围内;st_i^k表示目标i在圈次k的可见时间窗口开始时间,et_i^k表示目标i在圈次k的可见时间窗口结束时间,$[st_i^k,et_i^k]$:目标i在圈次k的可见时间窗口时间范围;

[0032] 约束(8)指观测目标只能调度到有其可见时间窗口的圈次上;b_i^k为二进制参数,为1时表示目标i在圈次k有可见时间窗,否则为0;

[0033] 约束(9)代表了模型中决策变量的取值范围。

[0034] 进一步地,步骤3中对敏捷卫星调度模型求解的方法是贪婪随机迭代局部搜索启发式算法。

[0035] 进一步地,所述贪婪随机迭代局部搜索启发式算法的具体步骤是:

[0036] 步骤3.1:预先计算同一圈次任意一对可见时间窗口VTW_{j-1}、VTW_j上,相对于前一个可见时间窗口上每一个观测开始时刻t_{j-1}^k后一个时间窗口上的最早开始时间es_j,以及相对于后一个可见时间窗口上每一个观测开始时刻t_j^k前一个时间窗口上的最晚开始时间ls_{j-1};

[0037] 步骤3.2:设置贪婪随机自适应搜索算法中随机因子Greed的初始值为StartGreed,,并初始化当前解为空;

[0038] 步骤3.3:初始化内循环的迭代次数,初始化扰动因子参数向量S_d和R_d的各个元素为1;

[0039] 步骤3.4:调用扰动算子,从当前解位于圈次k的任务序列中从位置$S_d(k)$开始删除$R_d(k)$个连续的已调度任务,$S_d(k)$表示参数向量S_d的第k个元素,即移除子序列在圈次k的已调度任务序列中的起始位置,$R_d(k)$表示参数向量R_d的第k个元素,即圈次k的移除子序列规模大小;

[0040] 步骤3.5:调用插入算子,依次将未调度的待观测目标尝试插入到当前解中,直至当前解无可行插入,对每个圈次k,计算该圈次上所有可见待观测目标i在当前圈次k的已调度任务序列所有可行插入位置,存入L_i^k中,若$|L_i^k| \geq 1$,则仅保留代价cost最小的插入位置,并将其存于L^k中。所述L^k用于存放圈次k上的所有未调度目标的可行插入,其中每个未调度目标最多只有一个可行插入;

[0041] 步骤3.6:对L^k中所有插入按照对应观测目标的收益从大到小排序,仅保留前(1-Greed)·$|L^k|$个插入,采用轮盘赌的方式,随机选择一个插入并执行到当前解中,更新插入位置之后每个任务的最早开始时间,以及插入位置之前每个任务的最晚开始时间,然后返回步骤3.5,直至所有圈次找不到任何可行插入;

[0042] 步骤3.7:计算新产生的解的总收益,如果新产生的解的总收益大于当前最好解,将新产生的解作为新的当前最好解,并将内循环迭代次数置为零,如果新产生的解比当前最好解差,则内循环迭代次数加1,若该次数超过最大连续迭代步数,则跳至步骤3.8,否则返回至步骤3.4,并令扰动因子参数向量$S_d=S_d+R_d$,$R_d=R_d+1$;

[0043] 步骤3.8:令随机因子Greed=Greed-GreedDecrease,GreedDecrease为随机因子步长,若Greed仍大于或等于随机因子最小值StartGreed-GreedRange,GreedRange为随机因子波动范围,将当前最好解置为当前解,返回步骤3.3,否则,输出当前最好解。

[0044] 进一步地,所述插入算子为:

[0045] 步骤3.5.1:对当前解的每个圈次k,若当前圈次k无调度任务时,只有一个插入位

置时,设该插入位置对应的最早开始时间为待插入的未调度目标i的可见时间窗口开始时间,其最晚开始时间为其可见时间窗口结束时间;

[0046] 步骤3.5.2:当试图往当前序列各个位置插入未调度目标i时,

[0047] 若插入位置为序列头,则令该插入的最早开始时间es_i为待插入的未调度目标i的可见时间窗口开始时间;

[0048] 若插入位置为序列尾,则令该插入的最晚开始时间ls_i为待插入的未调度目标i的可见时间窗口结束时间;

[0049] 若插入位置为j和j+1之间时,根据当前序列中待插入位置之前任务j的最早开始时间es_j计算出待观测目标i插入到该位置时对应的最早开始时间es_i,根据当前序列中待插入位置之后任务j+1的最晚开始时间ls_{j+1}计算出待观测目标i插入到该位置时对应的最晚开始时间ls_i,如果$es_i < ls_i$,则插入可行,否则插入不可行。

[0050] 进一步地,在步骤3.1之前,还包括

[0051] 步骤3.1':预计算对任意圈次k上的任意一个时间窗口VTW_i^k的"不可达"最早时间向量ue_i^k和"不可达"最晚时间向量ul_i^k,其中$ue_i^k(j)$表示从VTW_i^k出发不能抵达VTW_j^k时对应的VTW_i^k的最早观测开始时间,$ul_i^k(j)$表示在VTW_i^k内观测目标i后,无法将j调度在VTW_j^k内的最晚观测开始时间。

[0052] 则在步骤3.5.1之前还包括步骤3.5.1':

[0053] 如果尝试插入观测目标i于已调度任务j和j+1之间,

[0054] 若j的观测开始时间t_j不早于其对应于i的不可达最早时间,即$t_j < ue_j(i)$,则该位置不可插入,且j+1之后的位置也不可插入,无需再做插入尝试;

[0055] 若j+1的观测开始时间不晚于其对应于i的不可达最晚时间,即$t_{j+1} < ul_{j+1}(i)$,则该位置不可插入,且j之前的位置也不可插入,无需再做插入尝试;

[0056] 若该插入位置满足不可达最早和最晚时间的要求,则转步骤3.5.1。

[0057] 进一步地,步骤3.3中所述代价是指:

[0058] $cost_i = trans_{ji}(es_j, es_i) + trans_{i(j+1)}(es_i, es'_{j+1}) - trans_{j(j+1)}(es_j, es_{j+1})$

[0059] 其中,es_j表示任务j的最早开始时间,es_i表示待观测目标i的最早开始时间,es'_{j+1}代表更新后的任务j+1的最早开始时间,i为待观测目标,$cost_i$表示待观测目标i插入到位置j与j+1之间的代价。

[0060] 进一步地,步骤3.5还包括,对所有圈次都执行完一次插入操作后,检查是否存在某个目标被调度到多个圈次的任务序列中,如果存在,则执行分配步骤,根据预先给定的分配策略将该目标分配到其中一个圈次的任务,并删除其余圈次对应的已调度任务。

[0061] 进一步地,所述预先给定的分配策略是指将待观测目标优先分配到总时间窗口数较少的圈次上。

[0062] 本发明还提供了一种考虑时间依赖转换时间的敏捷卫星调度系统,包括处理器,以及与所述处理器连接的存储器,所述存储器存储有考虑时间依赖转换时间的敏捷卫星调度方法的程序,所述程序执行时实现上面所述方法的步骤。

[0063] 与现有技术相比,本发明所取得的有益效果是:

[0064] 本发明考虑时间依赖转换时间的敏捷卫星调度方法,考虑到连续观测任务之间的

姿态转换时间的时间依赖性,使所构建的敏捷卫星调度模型能更好地满足调度需求,同时通过该模型所求解得到的调度方案其调度效率更高。

[0065] 在对模型求解时,通过证明转换时间满足先进先出规则和不等式规则后,在此基础上,通过预计算同一圈次任意一对观测窗口间相对于前一个可见时间窗口上每一个观测开始时刻t_{j-1}^{k}后一个时间窗口上的最早开始时间es_j,以及相对于后一个可见时间窗口上每一个观测开始时刻t_j^k,前一个时间窗口上的最晚开始时间ls_{j-1},在尝试进行插入操作时减少了调度求解过程中的多次重复计算,极大的减少了计算时间,提高了调度效率。

[0066] 通过预先计算每个时间窗口的不可达最早时间和不可达最晚时间,在待观测目标尝试插入时,将待观测目标的开始时间与不可达最早和最晚时间进行比较,可以提前筛选掉很多不可行的尝试插入操作,极大地减少计算时间,提高调度效率。

[0067] 通过使用扰动算子,可以放置算法陷入局部最优,从而更好地搜索到整个解空间。

附图说明

[0068] 图1为敏捷卫星调度的工作示意图;

[0069] 图2为本发明系统流程图;

[0070] 图3为转换时间先进先出规则示意图;

[0071] 图4为最小姿态转换时间示意图;

[0072] 图5为贪婪随机迭代局部搜索启发式算法流程框图;

[0073] 图6为插入算子示意图;

[0074] 图7为不可达最早时间和最晚时间示意图。

具体实施方式

[0075] 图1至图7示出了本发明考虑时间依赖转换时间的敏捷卫星调度方法的具体实施例,包括以下步骤,如图2所示:

[0076] 步骤1:获取待观测目标序列并对其进行预处理;

[0077] 步骤2:构建时间依赖转换时间的敏捷卫星调度模型;

[0078] 本实施例中所构建的时间依赖转换时间的敏捷卫星调度模型,首先是基于转换时间是满足先进先出规则和三角不等式规则的。

[0079] (1)先进先出规则(FIFO rule)

[0080] 先进先出规则是指,假设卫星对前一个观测任务有两个不同的观测开始时间t_i和t_j($t_i<t_j$),通过姿态转换之后,t_j对应的下一个任务的到达时间不早于t_i对应的到达时间。如图3中的(a)(b)所示,假设t_i和t_j分别为时间窗口i和j的两个观测开始时间,转换时间为$trans_{ij}(t_i,t_j)$,推迟前一个窗口观测时间$\Delta t\geq 0$,其对应的转换时间为$trans_{ij}(t_i+\Delta t,t_j)$,转换时间的计算公式为

[0081] $$trans_{ij}(t_i,t_j)=\begin{cases}11.66, & |\Delta g(t_i,t_j)|\le 10\\ 5+\frac{|\Delta g(t_i,t_j)|}{v_1}, & 10<|\Delta g(t_i,t_j)|\le 30\\ 10+\frac{|\Delta g(t_i,t_j)|}{v_2}, & 30<|\Delta g(t_i,t_j)|\le 60\\ 15+\frac{|\Delta g(t_i,t_j)|}{v_3}, & 60<|\Delta g(t_i,t_j)|\le 90\\ 20+\frac{|\Delta g(t_i,t_j)|}{v_4}, & 90<|\Delta g(t_i,t_j)|\end{cases} \quad (a)$$

[0082] 其中,姿态变化量

[0083] $\Delta g(t_i,t_j)=|\Delta\gamma_i(t_i)-\Delta\gamma_j(t_j)|+|\Delta\pi_i(t_i)-\Delta\pi_j(t_j)|+|\Delta\varphi_i(t_i)-\Delta\varphi_j(t_j)|$. $\gamma_i(t_i),\Delta\pi_i(t_i)$, $\Delta\varphi_i(t_i)$分别代表t_i时刻窗口i的侧摆角,俯仰角和偏航角。

[0084] 那么存在如下命题:

[0085] 命题1:时间依赖转换时间满足先进先出规则,当且仅当

[0086] $$\frac{d\,trans_{ij}(t_i,t_j)}{d\,t_i}\ge -1. \quad (b)$$

[0087] 证明:(→必要性)

[0088] 如果$trans_{ij}(t_i,t_j)$满足先进先出规则,那么

[0089] $$t_i+trans_{ij}(t_i,t_j)\le t_i+\Delta t+trans_{ij}(t_i+\Delta t,t_j),\forall\Delta t\ge 0$$

[0090] 进而,有

[0091] $$\frac{trans_{ij}(t_i+\Delta t,t_j)-trans_{ij}(t_i,t_j)}{\Delta t}\ge -1,$$

[0092] $$\lim_{\Delta t\to 0}\frac{trans_{ij}(t_i+\Delta t,t_j)-trans_{ij}(t_i,t_j)}{\Delta t}\ge \lim_{\Delta t\to 0}-1\frac{d\,trans_{ij}(t_i,t_j)}{d\,t_i}\ge -1.$$

[0093] (→充分性)

[0094] 根据上述转换时间和姿态变化角度计算公式,且已知姿态角度随时间连续变化,可知$trans_{ij}(t_i,t_j)$在范围$[t_i,t_i+\Delta t]$之间是连续变化的。由中值定理,可知在$[t_i,t_i+\Delta t]$至少存在一个点t_i^*使得

[0095] $$\frac{d\,trans_{ij}(t_i^*,t_j)}{d\,t_i^*}=\frac{trans_{ij}(t_i+\Delta t,t_j)-trans_{ij}(t_i,t_j)}{\Delta t}$$

[0096] 若$\frac{d\,trans_{ij}(t_i^*,t_j)}{d\,t_i^*}\ge -1$对所有$[t_i,t_i+\Delta t]$中的$t_i^*$都成立,则有

[0097] $$\frac{trans_{ij}(t_i+\Delta t,t_j)-trans_{ij}(t_i,t_j)}{\Delta t}\ge -1,$$

[0098] $t_i+trans_{ij}(t_i,t_j)\le t_i+\Delta t+trans_{ij}(t_i+\Delta t,t_j)$

[0099] 即满足先进先出规则。

[0100] 根据以上命题,只需验证$\frac{d\,trans_{ij}(t_i,t_j)}{d\,t_i}\ge -1$,即可证明转换时间符合先进先出规则。

不妨设$\kappa = \frac{d\,trans_{ij}(t_i,t_j)}{d\,t_i}$,其代表固定$t_j$,时刻$t_i$对应的窗口i的单位时间内转换时间变化量。而由转换时间和姿态变化量计算公式,且对于目前使用的半敏捷卫星来说,执行观测时侧摆角不变,偏航角接近于零,可得:

[0101] $$|\kappa| = \lim_{\Delta t\to 0}\frac{|g_i(t_i+\Delta t)-g_i(t_i)|}{v} \approx \lim_{\Delta t\to 0}\frac{|\pi(t_i+\Delta t)-\pi(t_i)|}{v} \qquad (c)$$

[0102] 其中,$g_i(t_i)$代表窗口i在t_i时刻的瞬时姿态角,v代表瞬时角速度。由于作为调度输入,姿态角度是以秒为单位的离散形式给出,上述式子可转化为:

[0103] $$|\kappa| \approx \frac{|\pi(t_i+1)-\pi(t_i)|}{v} \le \frac{|\pi(t_i+1)-\pi(t_i)|}{v_{min}} = |\kappa_{upper}| \qquad (d)$$

[0104] 其中v_{min}代表了$\{v_1,v_2,v_3,v_4\}$中的最小角速度,$\kappa|_{upper}|$代表了$|\kappa|$的上界。实际应用中,$|\kappa_{upper}|$的值取决于卫星的各项参数以及观测目标的地理位置。我们发现,给定任意实际卫星参数和算例,$|\kappa_{upper}|$总是严格小于1。因此,转换时间总是满足先进先出规则。

[0105] 此外,固定t_i,变化t_j,同样可以得出结论,推迟t_j,时间窗口i的最晚开始时间必然更晚。

[0106] (2)三角不等式规则

[0107] 三角不等式规则是指,对于时间窗口i和j来说,固定其观测开始时间,其转换时间必然小于或等于从i经过另一“迂回”时间窗口m到j的总转换时间。当姿态转换角速度为固定值(非时间依赖),由于$|\Delta g_{ij}|\le|\Delta g_{im}|+|\Delta g_{mj}|$($\Delta g_{ij}$代表从i到j的姿态角变化量),因此必然满足三角不等式规则。然而在实际应用中,姿态转换角速度取决于姿态角度的变化量,变化量越大,角速度越大,这是因为通常来说,越大的姿态角,系统会采用更快的机动引擎实现姿态转换动作。因此,一般来说,$v_1\le v_2\le v_3\le v_4$。

[0108] 假设忽略迂回时间窗口m的服务时间,如果下列不等式满足,则满足三角不等式规则:

[0109] $$trans(\Delta g_{im})+trans(\Delta g_{mj})\ge trans(\Delta g_{ij}) \qquad (e)$$

[0110] 假设,v_{im},v_{mj},v_{ij}代表对应于$|\Delta g_{ij}|$的角速度。已知$\Delta g_{ij}=\Delta g_{im}+\Delta g_{mj}$,必有$|\Delta g_{ij}|\le|\Delta g_{im}|+|\Delta g_{mj}|$,下面分为两种取值情况:

[0111] (1)若$|\Delta g_{ij}|>|\Delta g_{mj}|$,则有

[0112] $v_{im},v_{mj}\le v_{ij}$,

[0113] 代入公式(a),必有

[0114] $$trans(\Delta g_{im})+trans(\Delta g_{mj})\ge trans(\Delta g_{ij})$$

[0115] (2)若$|\Delta g_{ij}|\le|\Delta g_{mj}|$,考虑到转换时间函数必须为单调递增函数,因此满足

[0116] $$\frac{|\Delta g_{mj}|}{v_{mj}}\ge\frac{|\Delta g_{ij}|}{v_{ij}}$$

[0117] 满足不等式(e)。

[0118] 基于先进先出规则和三角不等式规则,所构建敏捷卫星调度模型的具体为:

[0119] 目标函数为:

[0120] $$Maximize \sum_{i\in T}\sum_{k\in O} p_i \cdot y_i^k \quad (1)$$

[0121] 表示最大化调度任务的总收益，

[0122] 其中：y_i^k为决策变量，为1时表示第i个待观测目标调度在圈次k上，否则为0，p_i表示第i个目标的收益，T表示观测目标集合，O表示调度周期内的圈次集合；

[0123] 约束条件为：

[0124] $$\sum_{k\in O} y_i^k \le 1, \forall i \in T \quad (2)$$

[0125] $$\sum_{\substack{j\in T\cup\{e\}\\ j\neq i}} x_{ij}^k = \sum_{\substack{j\in T\cup\{s\}\\ j\neq i}} x_{ji}^k = y_i^k, \forall i \in T, k \in O \quad (3)$$

[0126] $$\sum_{j\in T\cup\{e\}} x_{sj}^k = 1, \forall k \in O \quad (4)$$

[0127] $$\sum_{j\in T\cup\{s\}} x_{je}^k = 1, \forall k \in O \quad (5)$$

[0128] $$t_i^k + d_i + mintrans_{ij}^k\left(t_i^k\right) - t_j^k \le M\left(1 - x_{ij}^k\right), \forall i,j \in T,\ k \in O \quad (6)$$

[0129] $$st_i^k y_i^k \le t_i^k \le et_i^k y_i^k, \forall i,j \in T,\ k \in O \quad (7)$$

[0130] $$y_i^k \le b_i^k, \forall i,j \in T,\ k \in O \quad (8)$$

[0131] $$x_{ij}^k \in \{0,1\}, y_i^k \in \{0,1\}, \forall i,j \in T \cup \{s,e\},\ k \in O \quad (9)$$

[0132] 约束(2)表示多圈唯一性约束，即每个观测目标在所有圈次内最多只能观测一次；

[0133] 约束(3)代表流平衡约束，连接了决策变量x_{ij}^k和y_i^k；

[0134] x_{ij}^k为决策变量，为1时表示观测完i观测目标后接着观测j观测目标，否则为0，i，j为观测目标索引，i，j∈T∪{s，e}，其中{s，e}是指虚拟的起始目标和终止目标；

[0135] 约束(4)和(5)规定了调度方案起始于虚拟目标s，终止于虚拟目标e；

[0136] 约束(6)规定了连续两个观测之间的时间间隔不小于转换时间长度，i^k表示在圈次k对观测目标i的观测任务，t_i^k为整数决策变量，代表观测目标i在圈次k的观测开始时间，d_i表示观测目标i的观测持续时长，$mintrans_{ij}^k\left(t_i^k\right)$表示最小姿态转换时间，即在圈次k，给定观测目标i的观测开始时间t_i^k，下一个观测目标j的最早到达时刻对应的姿态转换时间，M表示最大正整数；

[0137] 本实施例基于时间依赖转换时间满足先进先出规则的情况下，提出“最小姿态转换时间”，用来替代模型求解中实际的转换时间。如图4所示，给定两个时间窗口i和j，固定i

的观测开始时间t_i,而j的观测开始时间t_j可变,根据先进先出规则,总能找到窗口j的最早到达时间,此时对应的转换时间即为最小姿态转换时间,标记为$mintrans_{ij}(t_i)$。由此可知,最小姿态转换时间仅依赖于前一个观测任务的开始时间,而与后一个任务的观测开始时间无关。采用最小姿态转换时间替代实际转换时间,有利于降低问题维度,简化计算复杂程度。此外,为了减少调度算法重复计算最小姿态转换时间,对任意两个时间窗口,相对于前一窗口的每一秒观测开始时间的最小姿态转换时间可预处理计算出来。

[0138] 约束(7)代表了可见时间窗口约束,即每个任务的观测开始时间必须在其可见时间窗口范围内;st_i^k表示目标i在圈次k的可见时间窗口开始时间,et_i^k表示目标i在圈次k的可见时间窗口结束时间,$[st_i^k,et_i^k]$:目标i在圈次k的可见时间窗口时间范围;

[0139] 约束(8)指观测目标只能调度到有其可见时间窗口的圈次上;b_i^k为二进制参数,为1时表示目标i在圈次k有可见时间窗,否则为0;

[0140] 约束(9)代表了模型中决策变量的取值范围。

[0141] 步骤3:将预处理后的待观测任务序列输入敏捷卫星调度模型,并对敏捷调度模型进行求解;

[0142] 本实施例中对敏捷卫星调度模型求解的方法是贪婪随机迭代局部搜索启发式算法(Greedy Randomized Iterated Local Search,GRILS)。是贪婪随机自适应搜索算法(Greedy Randomized Adaptive Search Procedure,GRASP)和迭代局部搜索算法(Iterated Local Search,ILS)的结合。它被证明在多资源约束的带时间窗的团队定向问题上能取得很好的求解效果,而该问题与敏捷卫星调度问题的模型具有很高的相似性。在采用该算法的同时,本算法也进行了一定的修改,用于适应敏捷卫星调度的问题特点。具体如图5所示,:

[0143] 算法可分为两个层级的循环。内层循环对应于一个ILS算法,用于具体安排观测任务的调度访问,是整体算法的核心部分。外层循环控制一个随机因子参数Greed(0<Greed<1)的值,该参数作为内层循环的输入,用于控制ILS算法的随机性。在内层循环中,ILS由两个算子组成:插入算子(INSERT)和扰动算子(SHAKE)。每一步ILS的迭代,INSERT算子依次插入未调度的任务到当前解(即调度任务序列)中,直至无法再插入为止。该算子受Greed参数的控制,Greed越大,INSERT算子的随机性越强。例如,假设当前解有10个可行插入(选定待插入任务和插入位置),这些插入按收益大至小地排序,那么只考虑前(1-Greed)·10的插入,再从这些插入中以轮盘赌的方式随机选择一个执行插入。SHAKE算子从当前解中移除一部分的一调度任务,从而避免搜索陷于局部最优,提高搜索的多样性。执行完INSERT算子后,如果新产生的解比当前最好解更好,则将其记录为新的当前最好解。如果在一定的连续迭代步数内都,当前最好解都得不到提升,则结束内层ILS循环。在外层的循环中,随机因子Greed从参数StartGreed开始,以GreedDecrease作为步长,逐步递减至(StartGreed-GreedRange)。在每次外层循环迭代步,ILS算法都是以当前最好解作为起点开始求解。详细的步骤如下:

[0144] 步骤3.1':预计算对任意圈次k上的任意一个时间窗口VTW_i^k的“不可达”最早时间向量ue_i^k和“不可达”最晚时间向量ul_i^k,其中$ue_i^k(j)$表示从VTW_i^k出发不能抵达VTW_j^k时对应

的VTW_i^k的最早观测开始时间，$ul_i^k(j)$表示在VTW_i^k内观测目标i后，无法将j观测目标调度在VTW_j^k内的最晚观测开始时间；

[0145] 为了减少计算时间，如图7所示，对某圈次k上的某个时间窗口VTW_i^k，计算出“不可达”最早时间向量ue_i^k，和“不可达”最晚时间向量$ul_i^k(j)$，通过采用这样的预处理，当往已调度任务i^k后(前)尝试插入任务j^k时，必须满足$t_i^k < ue_i^k(j)(t_i^k > ul_i^k(j))$。从而很多不可行的尝试插入操作就可以提前筛选掉，极大地减少了计算时间。VTW_i^k表示圈次k上第i个观测目标的可见时间窗口，VTW_j^k表示圈次k上第j个观测目标的可见时间窗口。

[0146] 步骤3.1：预先计算并保存同一圈次任意一对可见时间窗口VTW_{j-1}、VTW_j上，相对于前一个可见时间窗口上每一个观测开始时刻t_{j-1}^k后一个时间窗口上的最早开始时间es_j，以及相对于后一个可见时间窗口上每一个观测开始时刻t_j^k，前一个时间窗口上的最晚开始时间ls_{j-1}。本实施例中，是预先计算并保存相对于可见前一个时间窗口上每一秒都作为观测开始时刻时，在满足最小姿态转换时间的情况下，后一个可见时间窗口上的最早开始时间向量。通过这样的预处理，在对调度模型求解时，只要给定前一个可见时间窗口上任务的观测开始时间，通过查询即可得到与该观测开始时间对应的下一个可见时间窗口上的任务观测最早开始时间。

[0147] 在前面基于先进先出规则得到，最小姿态转换时间仅依赖于前一个观测任务的开始时间，最小姿态转换时间的计算方法是通过二分搜索法求解得到的，具体为，固定前一个时间窗口VTW_i^k的观测时间t_i^k，检测后一个时间窗口VTW_j^k的可见时间窗口开始时间st_j^k，如果$trans_{ij}^k\left(t_i^k, st_j^k\right)$满足转换时间约束，则令$mintrans_{ij}^k\left(t_i^k\right) = trans_{ij}^k\left(t_i^k, st_j^k\right)$。否则检测其时间窗口结束时间$et_j^k$，若不满足转换时间约束，则代表$t_i^k$时刻观测目标i后，无法再观测目标j。若满足约束，则VTW_j^k的最早观测时间必然在可行区间$\left[st_j^k, et_j^k\right]$之内，使用二分搜索检查$\left[st_j^k, et_j^k\right]$的中点是否能满足约束，并将可行区间分成原来的一半，继续二分搜索直至找到最早开始时间，此时对应的转换时间即为最小姿态转换时间。这个计算过程标记为$EarliestStartTime_{ij}^k(t_i^k)$。同样地，固定后一个观测$VTW_j^k$的开始时间，也可以采用类似的方法，计算出前一个观测的最晚开始时间，标记为$LatestStartTime_{ij}^k(t_j^k)$。具体为：固定后一个观测$VTW_j^k$的开始时间，检测前一个时间窗口$VTW_i^k$的可见时间窗口开始时间$st_i^k$，如果$trans_{ij}^k(st_i^k, t_j^k)$满足转换时间约束，则令$LatestStartTime_{ij}^k = st_i^k$，否则检测其时间窗口结束时间$et_i^k$，若不满足转换时间约束，则代表$t_i^k$时刻观测目标i后，无法再观测目标j，若满足约束，则VTW_i^k的最晚观测时间必然在$\left[st_i^k, et_i^k\right]$之内，继续检查$\left[st_i^k, et_i^k\right]$的中点是否能满足转换时

间约束,并将可行区间分成原来的一半,继续二分搜索直至找到最晚开始时间。

[0148] 步骤3.2:设置贪婪随机迭代局部搜索算法中随机因子Greed的初始值为StartGreed,并初始化当前解为空;

[0149] 步骤3.3:初始化内循环的迭代次数,初始化扰动因子参数向量S_d和R_d;

[0150] 步骤3.4:调用扰动算子,从当前解位于圈次k的任务序列中从位置$S_d(k)$开始删除$R_d(k)$个连续的已调度任务,$S_d(k)$表示参数向量S_d的第k个元素,即移除子序列在圈次k的已调度任务序列中的起始位置,$R_d(k)$表示参数向量R_d的第k个元素,即圈次k的移除子序列规模大小;

[0151] 扰动算子就是从当前解中移除一部分的已调度任务,本实施例中定义两个参数向量S_d和R_d,在内层循环开始时,初始化S_d和R_d的各个元素为1。当调用扰动算子时,对每个圈次k,从其调度序列的位置$S_d(k)$开始删除$R_d(k)$个连续调度任务,若删除时超过末位的任务,则从第一个任务继续删。令$S_d(k)=S_d(k)+R_d(k)$,$R_d(k)=R_d(k)+1$,下一次调用该算子时,采用新的参数值。如果$S_d(k)$大于圈次k的序列的任务数,则令其减去该任务数,使其回到序列靠前的位置。如果R_d大于任务数的$\frac{1}{3}$,则令$R_d=1$。该算子能确保在整个搜索过程中,每个已调度任务都至少能被删除一次。本实施例中使用扰动算子主要是为了防止算法陷入局部最优,从而更好地搜索到整个解空间。每次调用该算子,算法分别从每个圈次的已调度任务序列中移除由若干个连续任务组成的移除子序列。算子定义了两个参数:S_d指示了移除子序列在当前已调度任务序列中的起始位置;R_d代表了移除子序列的规模大小。由于每个圈次已调度序列的规模不一,不同圈次上,这两个参数的值不同。为了保证搜索的多样性,规模较大的已调度任务序列,对应于更大的R_d。

[0152] 步骤3.5:调用插入算子,依次将未调度的待观测任务插入到当前最好解在圈次k上的已调度任务序列中,记录每个未调度任务i在圈次k上的所有可行插入存入L_i^k中,若$|L_i^k| \geq 1$,则仅保留代价最小的插入,并将其存于L^k中,所述L^k用于存放圈次k上的的所有可行插入,其中每个未调度任务最多只有一个可行插入。本实施例中的可行插入就是记录了对应一个待观测目标在某一圈次上插入该观测目标的可行位置。本实施例中,代价的计算方法是:

[0153] $cost_i=trans_{ji}(es_j,es_i)+trans_{i(j+1)}(es_i,es'_{j+1})-trans_{j(j+1)}(es_j,es_{j+1})$

[0154] 其中,es_j表示任务j的最早开始时间,es_i表示待观测目标i的最早开始时间,es'_{j+1}代表更新后的任务j+1的最早开始时间,i为待观测目标,$cost_i$表示待观测目标i插入到位置j与j+1之间的代价。

[0155] 本实施例中插入算子的具体插入方法:

[0156] 步骤3.5.1':

[0157] 如果尝试插入观测目标i于已调度任务j和j+1之间,

[0158] 若j的观测开始时间t_j不早于其对应于i的不可达最早时间,即$t_j<ue_j(i)$,则该位置不可插入,且j+1之后的位置也不可插入,无需再做插入尝试;

[0159] 若j+1的观测开始时间不晚于其对应于i的不可达最晚时间,即$t_{j+1}<ul_{j+1}(i)$,则该位置不可插入,且j之前的位置也不可插入,无需再做插入尝试;

[0160] 若该插入位置满足不可达最早和最晚时间的要求,则转步骤3.5.1。

[0161] 步骤3.5.1:对当前解的每个圈次k,若当前圈次k无调度任务时,只有一个插入位置时,设该插入位置对应的最早开始时间为待插入的未调度目标i的可见时间窗口开始时间,其最晚开始时间为其可见时间窗口结束时间;

[0162] 步骤3.5.2:当试图往当前序列各个位置插入未调度目标i时,

[0163] 若插入位置为序列头,则令该插入的最早开始时间es_i为待插入的未调度目标i的可见时间窗口开始时间;

[0164] 若插入位置为序列尾,则令该插入的最晚开始时间ls_i为待插入的未调度目标i的可见时间窗口结束时间;

[0165] 若插入位置为j和j+1之间时,根据当前序列中待插入位置之前任务j的最早开始时间es_j计算出待观测目标i插入到该位置时对应的最早开始时间es_i,根据当前序列中待插入位置之后任务j+1的最晚开始时间ls_{j+1}计算出待观测目标i插入到该位置时对应的最晚开始时间ls_i,如果$es_i<ls_i$,则插入可行,否则插入不可行。

[0166] 所述最早开始时间es_j的计算方法为:

[0167] 给定某圈次上的一个已调度任务序列(j-1)-j-(j+1)-(j+2),忽略上标k,通过$es_j=EarliestStartTime_{(j-1)j}(es_{j-1})$从前往后依次计算每个任务j的最早开始时间,其中最靠前安排的任务j-1的最早开始时间置为其窗口开始时间st_{j-1},在步骤3.1已经预计算出前一个可见时间窗口上每一秒作为观测开始时间时,下一个可见时间窗口上观测任务的最早开始时间,因此在尝试插入时,根据给定的前一个可见窗口任务的开始时间es_{j-1},得到后一个可见时间窗口上任务的最早开始时间es_j。同样,所述最晚开始时间ls_j:在预计算的基础上,通过$ls_j=LastestStartTime_{j(j+1)}(ls_{j+1})$从后往前依次计算每个任务的最晚开始时间,最靠后的任务(j+2)的最晚开始时间设为$(et_{j+2}-d_{j+2})$,et_{j+2}为任务j+2的窗口结束时间,d_{j+2}为任务j+2的观测持续时长,因此给出后一个可见时间窗口的最晚开始时间,就可以得到前一个可见时间窗口的最晚开始时间。

[0168] 本实施例中,插入算子选取一部分未调度目标,根据其可调度窗口插入到当前解的指定位置,同时保证插入后当前解的可行性。本实施例中所定义的解是指由若干个已调度任务的序列构成,每个序列对应于一个圈次上的调度方案。当往当前解中插入某个观测任务时,由于转换时间约束的影响,插入位置之后的后续任务有可能不得不重新调度或者推迟观测时间,从而保证解的可行性。因此这些后续任务之间的转换时间需要逐个更新,来检测其是否违反可见时间窗约束。由于可见时间窗口约束的存在,大量的插入尝试都是不可行的,并且已调度序列规模越大,每次插入需要更新和检查转换时间的次数越多,这就占用了大量的计算资源,因此,这种“全”可行性检测不能满足高效求解的要求。为了减少这种大量的计算资源,本实施例所设计的插入算子是一种针对于时间窗约束和转换时间约束的快速插入方法。通过计算每个任务的最早开始时间和最晚开始时间,根据先进先出规则,对序列中任意一个任务j来说,$[es_j,ls_j]$内任意一个时间点都可作为其观测开始时间,且不会影响整个序列的可行性。当序列中任意一个任务的观测开始时间确定,都可以通过回溯法确定序列中其余任务的观测开始时间。图6中,当试图往当前序列的j和(j+1)之间的位置插入新任务i时,由$es_i=EarliestStartTime_{ji}(es_j)$和$ls_i=LastestStartTime_{i(j+1)}(ls_{j+1})$分别计算出任务i插入到该位置时对应的最早开始时间$es_i$和最晚开始时间$ls_i$。若$es_i\leqslant ls_i$,则该插入可行,否则插入该位置后,会致使其余已调度任务违反时间窗约束。采用这种快速

插入方法,不需要重新计算当前序列其余已调度任务的转换时间,因此大幅减少了检测插入可行性所花费的时间。但需注意的是,每次成功插入任务后,需要更新插入位置之后的所有后续任务的最早开始时间,以及插入位置之前的所有前置任务的最晚开始时间。

[0169] 通过上述算子的快速插入方法,可以高效评估所有未调度任务在当前解中任意插入位置的可行性,筛选出所有可行插入。通过对所有圈次执行这样的插入操作,当前解中可能会存在某个目标被调度到多个圈次上,也就是违反了多圈唯一性约束。为了确保方案的可行性,定义了一个分配操作(标记为Assignment()),根据预设的规则,对于重复调度的观测目标,仅将其分配到其中一个可行圈次上。根据初步实验发现,采用柔性分配策略,往往能取得较好的分配效果。柔性分配策略是指,将待观测目标优先分配到总时间窗口数较少的圈次上。这是因为把任务优先分配到时间窗口数少的圈次,那么拥有窗口数较多的圈次,就有足够的空间容纳更多的观测任务。执行完分配操作后,重新尝试插入新任务,进入下一个循环。当不存在任何可行的可行插入时,插入算子终止。

[0170] 步骤3.6:对L^k中所有插入按照对应观测目标的收益从大到小排序,仅保留前(1-Greed)·$|L^k|$个插入,采用轮盘赌的方式,随机选择一个插入并执行到当前解中,更新插入位置之后每个任务的最早开始时间,以及插入位置之前每个任务的最晚开始时间,然后返回步骤3.5,直至所有圈次找不到任何可行插入;

[0171] 外循环的随机因子参数Greed(0<Greed<1)越大时,插入算子的随机性越强,假设当前解有10个可行插入(选定待插入任务和插入位置),这些插入按收益大至小地排序,那么只考虑前(1-Greed)·10的插入,再从这些插入中以轮盘赌的方式随机选择一个执行插入。每次成功插入任务后,需要更新插入位置之后的所有后续任务的最早开始时间,以及插入位置之前的所有前置任务的最晚开始时间。

[0172] 步骤3.7:计算新产生的解的总收益,如果新产生的解的总收益大于当前最好解,将新产生的解作为新的当前最好解,并将内循环迭代次数置为零,如果新产生的解比当前最好解差,则内循环迭代次数加1,若该次数超过最大连续迭代步数,则跳至步骤3.8,否则返回至步骤3.4,并令扰动因子参数向量$S_d=S_d+R_d$,$R_d=R_d+1$;

[0173] 步骤3.8:令随机因子Greed=Greed-GreedDecrease,GreedDecrease为随机因子步长,若Greed仍大于或等于随机因子最小值StartGreed-GreedRange,GreedRange为随机因子波动范围,将当前最好解置为当前解,返回步骤3.2,否则,输出当前最好解。

[0174] 步骤4:将步骤3中的求解结果输出,得到敏捷卫星的调度方案。

[0175] 本发明还提供了一种考虑时间依赖转换时间的敏捷卫星调度系统,包括处理器,以及与所述处理器连接的存储器,所述存储器存储有考虑时间依赖转换时间的敏捷卫星调度方法的程序,所述程序执行时实现上面所述方法的步骤。

[0176] 下面通过实验验证本发明所述方法的有效性:

[0177] (1)实验算例与算法参数

[0178] 实验主要对比了最新研究文献中的单星算法(ALNS)和多星算法(A-ALNS),这两个算法主要关注两个区域内的侦察活动:中国区域(3°N-53°N)和全球区域(74°E-133°E)。算例假设观测目标均匀随机生成于这些区域中。由于初步的实验发现,对于全球实验,无论是对比算法,还是本算法,均能调度近乎所有的观测目标,因此全球区域的算例过于简单,不能反映出算法差异,不在对比实验考虑范围内。算例中,所有观测目标的收益和观测持续时

间服从[1,10]和[15,30]的均匀分布随机生成(以秒为单位),调度周期设为24小时。

[0179] 算法涉及了4个输入参数,分别是StartGreed,GreedRange,GreedDecrease,以及最大连续不提高迭代步数。初步实验证明,当StartGreed＝1,GreedRange＝0.2,GreedDecrease＝0.02时,算法能取得较好的结果。对内层ILS循环来说,当最大连续不提高迭代步数设为300时,内层ILS已经能平稳收敛到某个稳定值。该实验运行于IntelCore22.5GHz处理器和8GB内存的机器上。对每一个算例,本算法求解结果取自5次独立重复试验的平均值

[0180] (2)单星算例对比

[0181] 表1中国区域单星算例实验对比

[0182]

目标数	P_t	ALNS		本发明 GRILS		Outper(%)
		P_s/P_t(%)	Cpu(s)	P_s/P_t(%)	Cpu(s)	
100	571	91.13	22.06	99.75	7.99	9.45

[0183]

200	1053	60.32	89.69	88.37	21.37	46.51
300	1474	43.45	175.65	72.41	38.13	66.63
400	2065	31.68	303.85	60.93	62.75	92.36
500	2683	26.42	420.61	53.45	94.02	102.28
600	3231	25.17	504.08	48.16	126.48	91.32
Avg		46.36		70.51		52.09

[0184] 表1展示了在单星调度的中国区域算例上,本算法与文献中最好算法ALNS之间的对比结果。其中,P_t表示了当前算例所有待观测目标的总收益,P_s代表了该算法调度方案的收益值,Outper代表了在当前算例中,本算法求解质量优于ALNS的百分比。表1的结果显示,我们的算法GRILS的调度方案要比ALNS算法的平均高出52%。此外,这种求解质量的差距,随着算例规模的增大而增大。例如,在目标数为400,500,600的算例上,GRILS的收益值都近乎是ALNS的两倍。这充分证明了,本算法要优于ALNS算法,特别是在大规模算例上。此外,本算法花费的求解时间要比ALNS少好几倍。

[0185] 表2中国区域多星算例实验对比

[0186]

卫星数	目标数	A-ALNS		本发明 GRILS		Outper(%)
		P_s/P_t(%)	Cpu(s)	P_s/P_t(%)	Cpu(s)	
2	100-500	68.29	406.12	78.63	8.49	15.14
2	600-1000	30.09	1133.03	51.60	34.15	71.50
3	100-500	76.98	172.47	93.92	18.41	22.01
3	600-1000	37.49	763.92	73.18	58.56	95.19
4	100-500	88.57	172.08	99.06	27.10	11.85
4	600-1000	49.72	640.25	84.14	74.05	69.23
5	100-500	92.34	128.25	99.11	32.85	7.33
5	600-1000	55.54	589.94	86.59	86.89	55.91
6	600-1000	95.87	87.79	99.80	39.64	4.10
6	100-500	62.79	433.86	90.58	101.99	44.26
Avg		65.77		85.66		30.25

[0187] 表2展示了在中国区域的多星调度算例上,本发明与文献最好算法A-ALNS之间的对比实验。算例根据规模大小分为两个部分:100-500的中等规模组和600-1000的大规模组。对比结果表明,在多星算例上,本发明GRILS表现远好于A-ALNS算法,调度收益值平均高出30%,而本算法的计算时间远少于A-ALNS。

[0188] 以上仅是本发明的优选实施方式,本发明的保护范围并不仅局限于上述实施例,凡属于本发明思路下的技术方案均属于本发明的保护范围。应当指出,对于本技术领域的普通技术人员来说,在不脱离本发明原理前提下的若干改进和润饰,应视为本发明的保护范围。

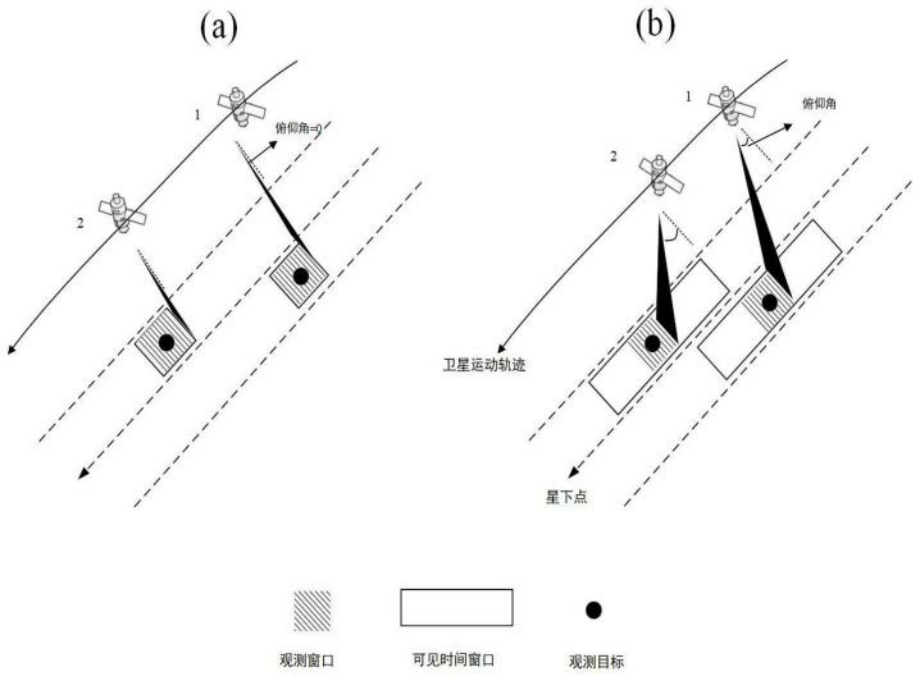

(a)
(b)
1
2
俯仰角
卫星运动轨迹
星下点
观测窗口
可见时间窗口
观测目标

图1

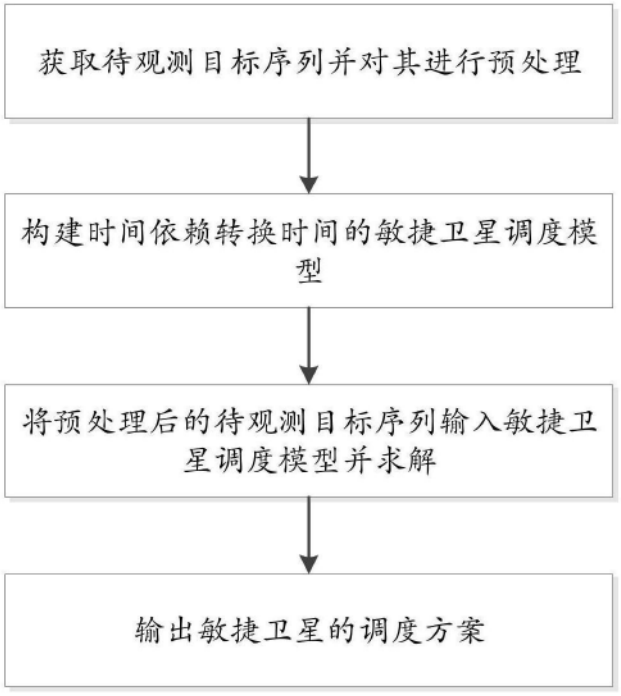

获取待观测目标序列并对其进行预处理
构建时间依赖转换时间的敏捷卫星调度模型
将预处理后的待观测目标序列输入敏捷卫星调度模型并求解
输出敏捷卫星的调度方案

图2

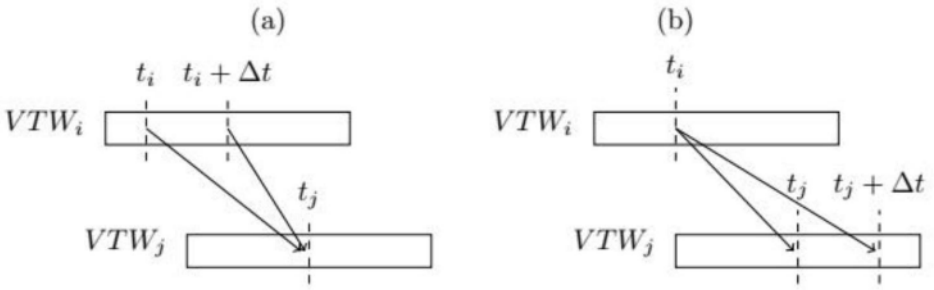

(a)
t_i $t_i+\Delta t$
VTW_i
t_j
VTW_j
(b)
t_i
VTW_i
t_j $t_j+\Delta t$
VTW_j

图3

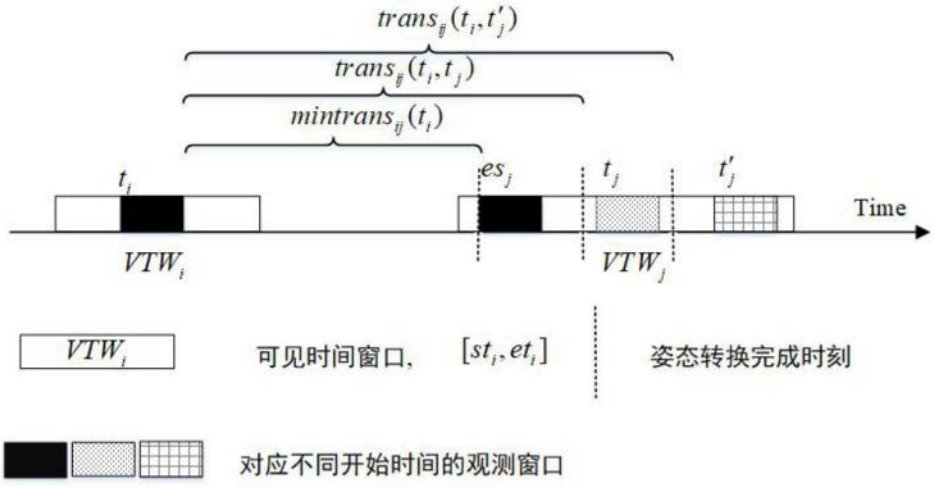

$trans_{ij}(t_i,t'_j)$
$trans_{ij}(t_i,t_j)$
$mintrans_{ij}(t_i)$
t_i
es_j
t_j
t'_j
Time
VTW_i
VTW_j
VTW_i
可见时间窗口，
$[st_i,et_i]$
姿态转换完成时刻
对应不同开始时间的观测窗口

图4

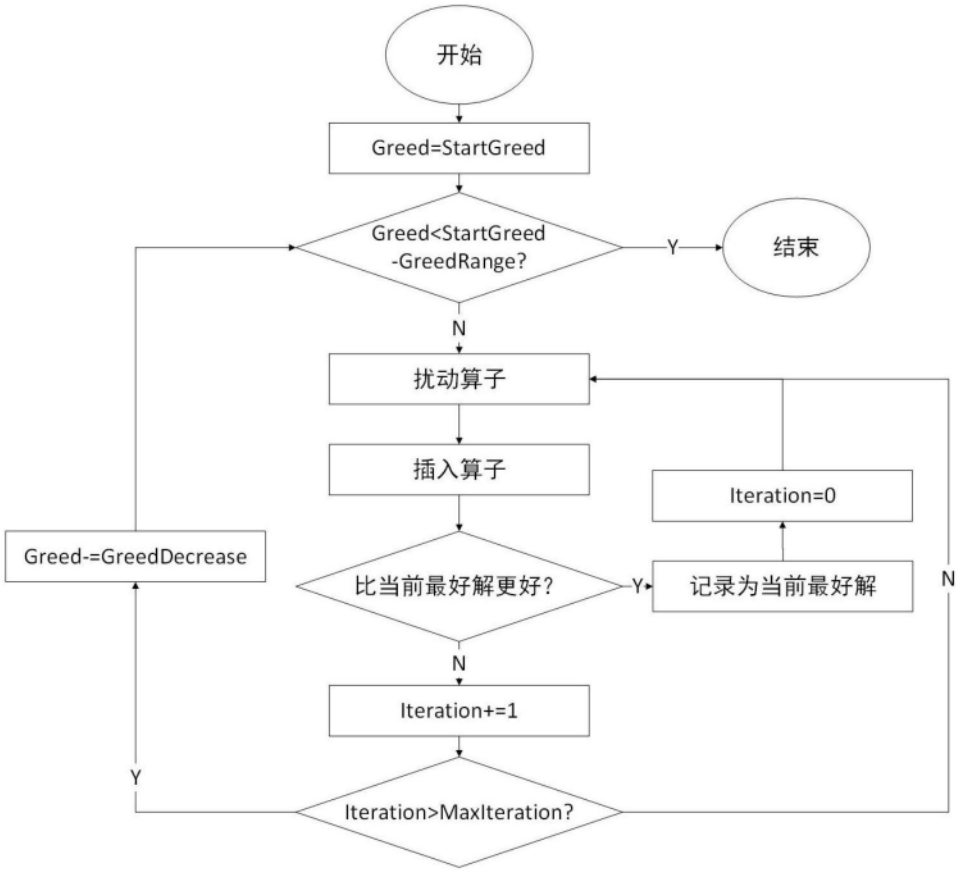

开始
Greed=StartGreed
Greed<StartGreed
-GreedRange?
Y
结束
N
扰动算子
插入算子
Iteration=0
Greed-=GreedDecrease
比当前最好解更好?
Y
记录为当前最好解
N
N
Iteration+=1
Y
Iteration>MaxIteration?

图5

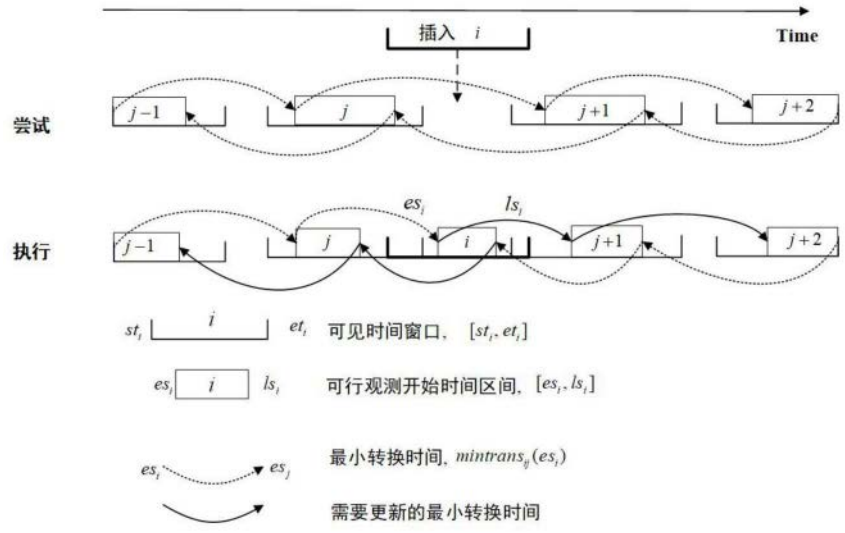

插入 i
Time
尝试
j−1
j
j+1
j+2
执行
es_i
ls_i
j−1
j
i
j+1
j+2
st_i
i
et_i
可见时间窗口，[st_i, et_i]
es_i
i
ls_i
可行观测开始时间区间，[es_i, ls_i]
es_i
es_j
最小转换时间，mintrans_ij(es_i)
需要更新的最小转换时间

图6

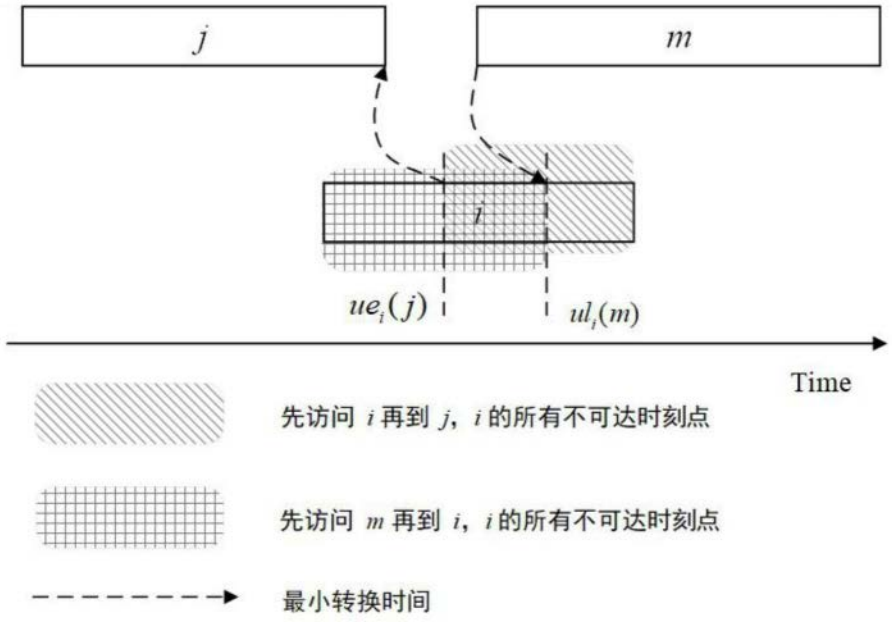

j
m
i
$ue_i(j)$
$ul_i(m)$
Time
先访问 i 再到 j，i 的所有不可达时刻点
先访问 m 再到 i，i 的所有不可达时刻点
最小转换时间

图7