第 35 卷第 10 期 系统工程理论与实践 Vol.35, No.10
2015 年 10 月 Systems Engineering — Theory & Practice Oct., 2015

文章编号: 1000-6788(2015)10-2504-09 中图分类号: C934; TP181 文献标志码: A

基于层次任务网络的应急资源协作规划方法

周 超[1,2], 王红卫[1,2], 祁 超[1,2]

(1. 华中科技大学 系统工程研究所, 武汉 430074; 2. 华中科技大学 图像信息处理与智能控制教育部重点实验室, 武汉 430074)

摘 要 应急响应一般涉及多个部门, 各参与部门需要通过合理的协调方法规划应对方案, 协同使用应急资源是不同部门间应急行动方案制定的关键. 为处理规划过程实时产生的资源冲突, 文章提出了基于层次任务网络 (hierarchical task network, HTN) 的多部门分布式协作任务规划框架, 将规划方案行动的生成过程作为需要考虑的协作问题, 并通过协作过程与规划过程的嵌套, 减少规划无效方案而提高协作效率. 其次, 探究并设计处理重用性资源的协调机制, 使用多种优先级判断规则消解资源使用冲突. 进一步, 在建立的协作框架和资源协调机制的基础上, 提出基于 HTN 的协作规划算法. 最后通过实验案例说明了应急资源协作规划方法的应用, 并对本方法的有效性和相对于原有方法的效率优势进行了验证.

关键词 应急决策; 层次任务网络; 多主体协调; 协作规划

Hierarchical task network based emergency resource coordinated planning approach

ZHOU Chao[1,2], WANG Hong-wei[1,2], QI Chao[1,2]

(1. Institute of Systems Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; 2. Key Laboratory of Image Processing and Intelligent Control of Education Ministry, Huazhong University of Science and Technology, Wuhan 430074, China)

Abstract The departments involved in emergency decision-making are required to perform coordinated planning for a global action plan. The coordination of using emergency resource is a key issue of coordinated planning for an valid emergency response plan. In order to handle the real-time generated emergency resource conflict along with the distributed planning process, an hierarchical task network (HTN) planning process based multi-agent distributed coordinated planning method is designed. The method embeds the coordination process into hierarchical task network (HTN) planning process with the consideration that the action plan generation process should also be coordinated. This method can reduce the generation of useless plan and improve the efficiency of cooperation. Additionally, the coordination mechanism dealing with the reusable resources is developed, and a variety of priority rules are used to resolve resource using conflicts. Next, based on these designs, an hierarchical task network (HTN) based coordinated planning algorithm is proposed and implemented. Finally, an experimental study was presented to demonstrate the application of emergency resource coordinated planning and the validity and the efficiency of the method.

Keywords emergency decision-making; hierarchical task network (HTN); multi-agent coordination; coordinated planning

0 引言

应急决策是应对突发事件的核心手段和重要内容 [1]. 突发事件一般涉及到多个地理、职能上分散的部门共同参与应急响应 [2], 通过规划制定应对方案是有效组织参与应急响应的多个部门协调决策和运作的关键 [3]. 因此, 多部门参与的应急行动方案制定是典型的分布式协作任务规划问题. 应急响应决策过程中各部

收稿日期: 2014-12-26
资助项目: 国家杰出青年基金 (71125001); 国家自然科学基金 (71371079)
作者简介: 王红卫 (1966–), 男, 浙江宁波人, 博士生导师, 教授, 研究方向: 公共安全与应急管理、物流与供应链系统, E-mail: hwwang@hust.edu.cn.

门分别制定行动方案以实现各自任务目标, 然而受外部环境的制约, 各部门规划出的行动方案可能在应急行动的时间、资源和执行效果等方面存在冲突关系. 协作是对不同部门行动间依赖关系的管理 [4], 是实现各部门应急行动协同配合的重要手段. 协调机制是协作过程中用于处理依赖关系的主要依据, 规定了何时进行信息交互、如何确定协调结果以及如何进行方案调整等. 在应急响应决策过程中, 应急资源调度对应急任务执行的关键, 特别是资源使用时引起的冲突, 可能不能保障优先级高的任务的资源需求, 甚至导致所有任务无法完成. 因此, 需要针对不同类型应急资源的冲突关系设计相应的协调机制.

在分布式协作任务规划的相关研究中, Decker[5] 提出的 generalized partial global planning (GPGP) 为分布式计算机系统中的各个独立子系统构建了协作控制策略与协议 [6−14]. 基于 GPGP 的协作主要包括任务规划、通过共享形成全局任务视图和任务协调与调度等基本组成部分. 这类方法需要首先规划出完整本地方案, 再由协作模块利用协调机制处理依赖关系, 本质上是先规划后协作的过程. 但是在实际的应急决策中, 涉及的多个部门之间需要高度协同, 导致其规划的任务之间存在大量的实时依赖关系, 如果按照先规划后协作的方式进行处理, 会反复调用规划过程使得整体效率低下. 另一方面, 基于 HTN[15] 规划在应急响应决策中的使用日益广泛 [3], 考虑到多部门应急任务规划的分布式特征, 需要进一步引入分布式 HTN 规划技术. Dix 等 [16] 在多 Agent 系统中引入了 HTN 规划器, 由分布于系统中的 Agent 收集环境信息, 由规划 Agent 统计信息并利用 HTN 规划进行问题求解, 其实质仍是集中式规划. Kabanza 等 [17] 设计了 DSHOP, 对同一任务树的不同节点使用多个规划器进行规划, 从而减少了单个规划器的规划压力, 提高了系统规划速度, 该方法的本质是分布式计算而没有协作. Hayashi 等 [18] 将 HTN 分布在呈上下级关系的父 Agent 和子 Agent 中, 上层规划器的结果作为下层规划器的输入, 但未考虑同层次方案间行动的相互影响.

综上所述, 基于 GPGP 的协作任务规划先规划后协作的方式, 不符合应急任务规划中规划和协作紧密结合的特征. 另一方面, 分布式 HTN 规划的研究中, 较少有研究针对规划过程的协作问题, 不能有效处理应急规划中实时出现的资源冲突关系. 因此, 需要设计基于层次任务网络的协作规划方法, 着重针对应急资源产生的任务间的冲突关系设计相应的协调机制, 从而实现应急响应决策中的分布式协作任务规划.

本文提出了基于层次任务网络 (HTN) 的多部门分布式协作任务规划框架, 将协作与 HTN 规划相结合嵌套, 实现规划过程中的协作, 意在一定程度上减少无用方案的产生从而提高协作方案的生成效率. 另外重点设计了处理重用性资源冲突关系的协调机制, 并基于 HTN 规划和协作框架进一步设计了相应的协作任务规划算法. 最后通过一个应急运输案例说明了应急资源协作规划方法的应用过程及该方法的有效性和优势.

1 基于 HTN 的分布式协作框架和协调机制

1.1 基于 HTN 的分布式协作框架

分布式 HTN 规划需要多个独立的规划器完成各自的任务规划, 信息无法做到全局共享, 各规划器的内部推理对外不可见. 基于前文分析, 各规划器需要在规划过程中对存在依赖关系的行动进行信息交互, 处理依赖关系, 从而实现协作任务规划.

基于 HTN 的分布式协作任务规划框架如图 1 所示, 该框架通过共享任务视图, 在任务规划中嵌入任务协调过程, 并利用 HTN 规划器本身具有的节点返回与重规划功能, 及时展开行动方案的的调整和修改, 从而实现对规划生成的原子行动进行实时协调. 具体当规划获得原子行动时: 1) 进行潜在依赖关系判断, 判断该原子行动与其它主体规划的原子行动是否具有潜在依赖关系; 2) 若存在潜在依赖关系, 则进行信息交互, 向其它规划器发送该本地行动的信息; 3) 进行依赖关系确认, 通过上一步的交互确定一定会发生的依赖关系; 4) 进行依赖关系的协调, 利用协调机制进行行动的协调; 5) 获得行动协调结论后, 主体根据结果调整行动, 或继续进行下一步规划. 参与协调的主体重复上述步骤, 直到最终生成完整而互不影响的本地方案. 如图 1 所示, 本框架可以对两个主体的规划过程进行协调, 而且可进一步拓展为多个主体进行实时协调.

该协作任务规划框架, 通过将协调过程嵌入规划过程中, 使得方案的制定与协调同时进行. 由于每次生成原子行动即时而尽早地处理确认的依赖关系 (如资源冲突关系), 可以实时动态的对方案进行修改而不用等一个完整方案的生成, 使得规划器能够减少对无效的后续原子行动的规划和生成数量. 因此, 该框架可以减少无效行动规划的数量, 从而提高协作的效率.

1.2 处理资源冲突的协调机制

Decker 等 [19] 运用 GPGP 处理了资源冲突的协调问题, 在此基础上, 考虑到应急资源冲突关系在多部门应急任务规划中的重要性, 基于上述框架设计可重用性资源冲突的协调机制. 冲突关系是指一个行动导致

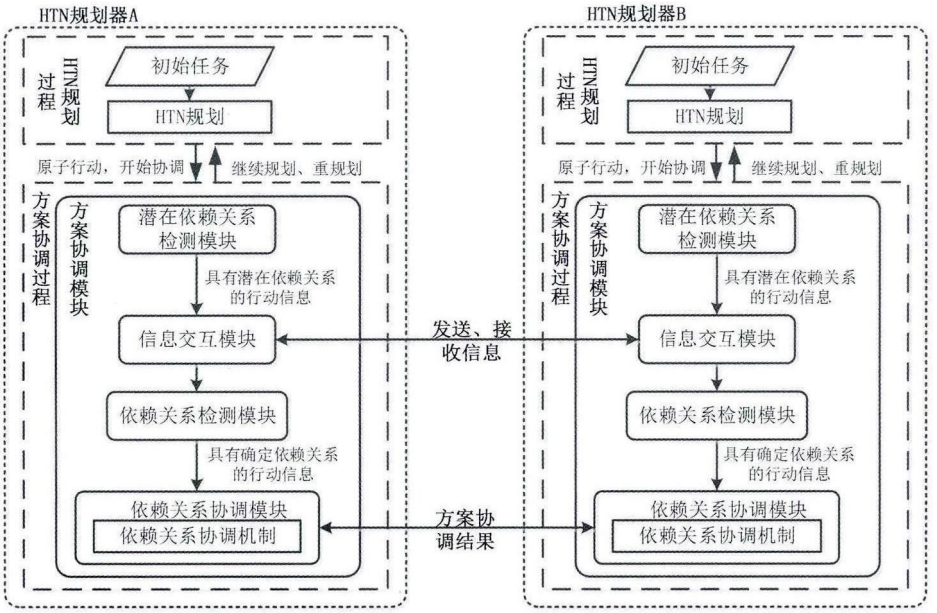

图 1 协作规划框架图

另一个行动无法执行 [9]. 可重用资源被多个部门同时使用时产生可重用资源冲突, 从而需要进行协调. 针对这类冲突, 可在比较方案行动优先级的基础上, 以保留优先级高的行动并调整优先级较低的行动的方式消解资源冲突. 以下三种情况下认为行动的优先级较高: 1) 行动使任务执行时间更少, 实际执行时更早使用资源; 2) 行动使任务执行质量更高, 完成收益高; 3) 行动使资源消耗更少, 单位时间使用资源较少.

2 基于 HTN 的应急资源协作规划方法

2.1 基于 HTN 的协作规划问题

基于 JSHOP2[20] 扩展的 HTN 应急资源协作规划问题可以表示为一个 3 元组 $P=(s_0,T,D)$, 其中 s_0 是初始状态, T 是初始任务网络, D 表示包括了操作集和方法集的序对的规划领域. 在分布式 HTN 规划问题中, 每个 HTN 规划器分别处理各自的 HTN 规划问题 $P_i=(s_{i0},T_i,D_i)$, 其中, i 代表 HTN 规划器的编号, s_{i0} 代表规划问题 P_i 的初始状态, T_i 代表规划问题 P_i 的初始任务网络, D_i 表示问题 P_i 的规划领域. 为了有效处理由应急资源产生的任务之间的冲突关系, 定义 $D_i=(O_i,M_i,Resource_i,Sendlist_i,Receivelist_i)$, 其中 O_i 是操作符集合, M_i 是方法集, 与传统 HTN 规划领域的方法集一致, $Resource_i$ 是规划问题所涉及的应急资源集合, $Sendlist_i$ 和 $Receivelist_i$ 分别表示具有潜在冲突关系的行动集合的发送列表和接收列表.

1) 操作符

为了描述行动间由资源占用产生的相互关系, 需要对每个规划器的操作符模型进行扩展, 增加资源使用和行动编号的变量, 第 i 个规划器的操作符描述为 $O_i=(:operatorhead(o),precond(o),del(o),add(o),Resourceused(o),st(o),et(o),costs(o),ActionID)$. 其中, $head(o)$、$precond(o)$、$del(o)$ 和 $add(o)$ 分别表示操作符的头、前提条件、删除效果和添加效果, 与传统定义相同; $Resourceused(o)$ 为操作所使用的资源编码, 后期用以确认不同操作行动是否使用了同一资源; $st(o)$ 表示操作占用资源的起始时间; $et(o)$ 表示操作占用资源的结束时间; $costs(o)$ 表示操作的效用值, 作为协调时的评价指标; $ActionID$ 表示此操作在本地规划器生成的规划方案中的行动序号. 扩展的操作符将操作与资源联系, 规划算法可以根据操作中使用的资源信息, 判断是否需要与其他规划器进行协调.

2) 应急资源

由于多个部门可能占用同一资源, 需要对这些资源状况进行描述. 第 i 个规划器的应急资源可描述为 6 元组 $Resourcei=(ResourceID,Resourcename,Resourcetype,Resourcestate,Resourcest,Resourceet)$. 其中, $ResourceID$ 代表资源编码; $Resourcename$ 表示资源名称; $Resourcetype$ 表示资源类型; $Resourcestate$ 表示资源的占用状态, 初始值为 0, 表示该资源没有被任何部门占用, 如果被某部门占用, 则被赋值为使用该资源的部门或规划器的编号 j; $Resourcest$ 表示资源开始被占用的时间; $Resourceet$ 表示资源被释放的时间.

3) 行动列表

不同部门各自规划生成的行动, 可能在交叠时间段内使用同一应急资源, 需要在规划领域中增加行动列表, 记录潜在的冲突行动, 表明相应的行动间存在潜在依赖关系. 行动列表分为发送行动列表和接收行动列表. 规划器 i 的发送行动列表 $Sendlist_i$ 是指主体 i 发送到其它主体的行动列表, 由一系列具有潜在依赖关系的本地行动 $opmessage()$ 构成. 接收行动列表 $Receivelist_i$ 是指本地规划器接收到来自其他规划器的行动列表, 由一系列具有潜在依赖关系的外部行动 $extopmessage()$ 构成. 本地行动和外部行动均可用 5 元组 $(ID, Resourceused(o), st(o), et(o), costs(o))$ 描述, 分别表示相应行动的编号、所使用的资源编码、资源开始占用时间、资源释放时间和行动的效用值. 发送行动列表和接收行动列表是规划器用来判断是否需要协作的重要依据, 如果为空则表明目前不需要考虑协作.

2.2　协作规划算法

基于 HTN 的协作规划具体的算法如图 2 所示. 考虑到不同主体规划过程的异步性, 在规划过程中, 较早检测到其分解产生的原子任务与另一规划器的方案存在潜在冲突的一方, 需要向对方发送潜在冲突信息, 待对方分解产生相应的具有冲突关系的原子任务后, 进行冲突协调. 根据协调机制判断哪一方应该对方案进行调整, 无需调整方案的一方可以继续规划过程, 需要调整的一方则获取冲突产生的任务节点, 进行重规划. 该规划算法在生成原子行动后, 对资源冲突关系进行检测并根据协调机制进行处理, 使用资源时间较早的行动具有更高的优先级而保持不变, 使用时间较晚的行动则需要进行调整或重新规划, 直至生成没有冲突的规划方案.

```
1:Procedure CP-HTN(s, T, D)
2: set s, initial state
3:   set T, initial task set
4:   set D, domain
5:   initialoperator ActionID= 0
6:   set P = the empty plan
7:   T0      ← {t belongs to T: no other task in T is constrained to precede t}
8:   loop
9:     if T is empty then return P
10:    nondeterministically choose any t in T0
11:    if t is a primitive task then
12:    choose a ground instance a forwith the smallest cost among the available resources
13:    ActionID = ActionID +1
14:    opmessage(a)=( ID, Resourceused(a), st(a),et(a),costs(a))
15:    sendlist←opmessage(a)
16:    send opmessage(a) to the other planners
17:      if receivelist ≠e:
18:        ResourceID = Resourceused(a)
19:        conflict actions a' ← { a' belongs to the extopmessages of receivelist :
20:             has Resourceused(a')= ResourceID, and has the smallest st(a')
21:             and duration(st,et) is overlapped with duration(st',et')}, then
22:        if st < st'
23:           delete the rest extopmessage in receivelist
24:           Resourcestate(r)=i, Resourcest(r)=st, Resourceet(r)=et
25:           delete T in T0 and add a into P
26:        else if st >st'
27:           delete opmessage(a) in sendlist
28:           Resourcestate(r)=i', Resourcest(r)=st', Resourceet(r)=et'
29:           Backtrack
30:      else delete T in T0 and add a into P
31:    if t is a non-primitive task then
32:      choose a method to decompose t into subtasks {t1, t2,... tn}
33:      delete T in T0 and add {t1, t2,... tn} into T0
34:    if receiving a new extopmessage(a') in receivelistof action a' and sendlist ≠e:
35:      ResourceID = Resourceused(a')
36:      conflict actions a ← {a belongs to the opmessages of sendlist :
37:           has Resourceused(a)= ResourceID, and has the smallest st(a)
38:           andduration(st,et)is overlapped with duration(st',et')}, then
39:      if st'< st
40:         delete the rest opmessage in sendlist
41:         Resourcestate(r)=i', Resourcest(r)=st', Resourceet(r)=et'
42:         Backtrack to task node at action a
43:      else if st' >st
44:         delete extopmessage(a') in receivelist
45:         Resourcestate(r)=i, Resourcest(r)=st, Resourceet(r)=et
46:         delete T in T0 and add a into P
47:  repeat
48: end CP-HTN
```

图 2　基于 HTN 的协作规划算法

具体来说, 当任务分解生成原子任务时 (第 12 行), 如果原子任务使用了资源, 则说明该任务存在潜在的冲突依赖关系, 需要与外部规划器进行交互, 更新发送列表并将其发送给其他规划器 (第 14~16 行). 查询接收列表, 判断是否存在资源冲突 (第 17~21 行). 如果存在冲突, 则继续判断冲突的任务在资源占用时间上的先后关系, 根据占用时间较早的任务优先级较高的原则, 对冲突进行协调 (第 22~29 行). 如果在任务分解过程中收到来自其他规划器的冲突消息 (第 31~34 行), 查询发送列表, 判断是否存在冲突 (第 35~38 行). 冲突的处理方式与生成原子任务时的冲突处理方式类似 (第 39~46 行).

3 算例分析

3.1 算例描述

考虑两个应急指挥部协同制定应急物资运输方案的情景: 假设在某区域内出现了两个较严重的堤防险情, 应急决策部门决定分别成立现场指挥部 A 和 B, 并设立一个抢险物资储备点 C 为两个指挥部提供抢险物资. 区域内有 R1、R2 等 6 条运输道路, 从物资储备点 C 到现场指挥部 A 和 B 之间各有 4 条可达道路, 其中有 2 条道路对 A 和 B 均可达, 运输队在各条道路上所需的运输时间如表 1 所示. 指挥部 A 和指挥部 B 分别拥有自己的运输队伍, 驻扎在地点 Loc1 的运输队伍 team1 和驻扎在 Loc2 的 team2 供指挥部 A 调遣, 而驻扎在 Loc3 的 team3 和驻扎在 Loc4 的 team4 则由指挥部 B 调遣. 运输队伍从驻扎地到达物资储备点或现场指挥部所需时间如表 2 所示. 每支运输队伍一次运输的容量上限、装载与卸载时间如表 3 所示. 两个现场指挥部的物资需求量如表 4 所示.

表 1 物资储备点和现场指挥部间各条道路的运输时间 小时

道路	R1	R2	R3	R4	R5	R6
C、A 之间	21	21.2	21.7	21.5	—	—
C、B 之间	20.5	20	—	—	20.9	22

表 2 运输队伍从驻扎地到达指定地点所需的时间 小时

驻扎点	A	B	C
Loc1	16.2	—	2.3
Loc2	17.7	—	2.7
Loc3	—	17.4	2.5
Loc4	—	16.5	2.6

表 3 物资运输队伍信息

队伍	容量上限/吨	装载时间/小时	卸载时间/小时
team1	35	1.2	1.2
team2	25	0.8	0.8
team3	25	0.6	0.6
team4	30	1	1

表 4 物资需求量

物资	A	B
需求量/吨	180	200

两个指挥部各自规划运输方案, 将 C 地物资运输至各自应急响应现场. 方案中的运输行动会涉及道路的占用, 道路在此视为可重用资源. 由于某些道路同时可以通往 A 地和 B 地, 很可能存在 2 个指挥部各自规划的方案在交叠时间段内占用同一条道路的情况, 因此需要对潜在的道路冲突进行处理. 这类冲突的协调以任务优先级作为判断依据, 保留优先级较高的任务, 并通过选择其他道路调整优先级较低的任务. 在本算例中, 规划过程中优先选择运输时间最短的道路, 而在协调道路资源冲突时较早占用道路的行动优先级较高.

3.2 实验结果及分析

前文设计的协作规划方法基于 JSHOP2[20] 实现, 仿真环境为 Windows XP 系统, Intel(R) Core(TM)2 Duo CPU E7500@2.93GHz, 2GB 内存. 基于案例设计和实际实验环境, 在单机上使用两个 HTN 协作规划器作为两个指挥部, 利用 Java 多线程技术对其逐一调用, 即每当规划器生成一个原子任务并完成需要的协作步骤后, 调用另一规划器进行同样的步骤. 如此反复此过程, 从而模拟出两个规划器的同步运行和协调过程.

由分布式协作任务规划方法获得的运输方案如表 5 所示. 结果表明, 通过资源冲突协调机制, 道路能在尽可能早的时间被使用, 并且各方案中的运输队伍持续进行往返行动时, 可以确保无冲突地使用道路资源. 分布式协作任务规划的内部推理过程如表 6 所示, 包括规划器进行深度优先搜索的内部推理步骤以及对本地方案进行调整的过程. 可见, 由于两个分布式协作任务规划器事先不知道对方的规划选择, 在每次使用道路资源时, 都会最优先选择对自身最有利的道路. 然而, 一旦发现使用道路存在资源冲突, 且应当由对方优先使用时, 将转而搜索使用次优道路的方案, 直到形成行动无冲突的最终方案.

本文提出的协作规划方法将协调过程结合至 HTN 规划过程, 希望通过实时协调以减少方案的规划工作量, 进而提高协调方案的生成效率. 使用 GPGP 原有的协调方式 [5], 即在 HTN 规划获得本地方案视图后进

表 5　分布式协作任务规划生成的运输方案

指挥部 A						指挥部 B					
队伍	Act	ST	ET	Rd	Cap	队伍	Act	ST	ET	Rd	Cap
Team1	to C	0	2.3		0	Team3	to C	0	2.5		0
	load	2.3	3.5		35		load	2.5	3.1		25
	to A	3.5	24.5	R1	35		to B	3.1	23.1	R2	25
	upload	24.5	25.7		0		unload	23.1	23.7		0
	to C	25.7	47.2	R4	0		to C	23.7	43.7	R2	0
	load	47.2	48.4		35		load	43.7	44.3		25
	to A	48.4	69.9	R4	35		to B	44.3	64.3	R2	25
	unload	69.9	71.1		0		unload	64.3	64.9		0
	to C	71.1	92.8	R3	0		to C	64.9	84.9	R2	0
	load	92.8	94.0		35		load	84.9	85.5		25
	to A	94.0	115.7	R3	35		to B	85.5	105.5	R2	25
	unload	115.7	116.9		0		unload	105.5	106.1		0
	return	116.9	133.1		0		to C	106.1	126.1	R2	0
Team2	to C	0	2.7		0		load	126.1	126.7		25
	load	2.7	3.5		25		to B	126.7	146.7	R2	25
	to A	3.5	24.0	R4	25		unload	146.7	147.3		0
	unload	24.5	25.8		0		return	147.3	164.7		0
	to C	25.8	47.5	R3	0	Team4	to C	0	2.6		0
	load	47.5	48.3		25		load	2.6	3.6		30
	to A	48.3	70.0	R3	25		to B	3.6	24.5	R5	30
	unload	70.0	70.8		0		unload	24.5	25.5		0
	to C	70.8	92.3	R4	0		to C	25.5	46.0	R1	0
	load	92.3	93.1		35		load	46.0	47.0		30
	to A	93.1	114.6	R4	35		to B	47.0	67.5	R1	30
	unload	114.6	115.4		0		unload	67.5	68.5		0
	return	115.4	133.1		0		to C	68.5	89.0	R1	0
							load	89.0	90.0		30
							to B	90.0	110.5	R1	30
							unload	110.5	111.5		0
							to C	111.5	132.0	R1	0
							load	132.0	133.0		10
							to B	133.0	153.5	R1	10
							unload	153.5	154.5		0
							return	154.5	171.0		0

注: 表中 Act 表示 team 采取的行动, ST 为行动起始时间, ET 为行动结束时间, Rd 为当前行动使用的道路, Cap 为运输队伍的装载量 (吨).

行协调的方法, 也可以依据本文 1.2 节提出的资源冲突协调方法协调主体方案间的行动并获得互不冲突的方案, 但是每次对冲突的协调均需要完整的行动方案, 求解效率有限. 为验证本文新提出的协作规划方法在冲突协调中效率优势, 使用上述算例中道路冲突协调问题, 将本章提出的规划与协调结合的协作规划方法与基于 GPGP 方法 [5](获得全局方案视图后协调) 的协作规划方法进行对比. 对比和评价以其求解过程中的关键性能为评价指标 (具体为: 最终获得无冲突方案时总共规划的行动数量和总协调次数), 以判断和分析各方法协作规划求解的效率. 除实施协调过程设置不同, 两种方法的基本 HTN 规划功能和设置相一致, 比较结果如表 7 所示.

对比发现本文提出的规划与协调相结合的协作规划方法在成功的协调所需要尝试的方案行动数量上具有明显优势 (减少 77.5% 的行动规划数量), 显著减少了协作规划中的无效规划过程. 具体来说, 在表 6 所示的推理过程中, A 与 B 在规划的第 7 步均已出现与其他规划器方案冲突的行动. 如表中所示, 规划器 A 的 Team1 从 C 到 A 的运输任务占用 R1 的时间区间为 [3.5, 24.5](第 3~4 步), 规划器 B 的 Team4 从 C 到 B

表 6 协作任务规划内部推理过程

指挥部 A										指挥部 B									
Team1					Team2					Team3					Team4				
Act	Time	Rd	Cap	Amnt	Act	Time	Rd	Cap	Amnt	Act	Time	Rd	Cap	Amnt	Act	Time	Rd	Cap	Amnt
to C	2.3	-	0	-						to C	2.5	-	0	-					
load	3.5	-	35	-						load	3.1	-	25	-					
to A	24.5	R1	35	-						to B	23.1	R2	25	-					
unload	25.7	-	0	35						unload	23.7	-	0	25					
					to C	2.7	-	0	-						to C	2.6	-	0	-
					load	3.5	-	25	-						load	3.6	-	30	-
					to A	24.7	R2	25	-						to B	24.1	R1	30	-
					to A	25.0	R4	25	-						to B	24.5	R5	30	-
					unload	25.8	-	0	60						unload	25.5	-	0	55
to C	46.7	R1	0	-						to C	43.7	R2	0	-					
load	47.9	-	35	-						load	44.3	-	25	-					
to A	68.9	R1	35	-						to B	64.3	R2	25	-					
unload	70.1	-	0	95						unload	64.9	-	0	80					
					to C	47.0	R2	0	-						to C	46.0	R1	0	-
to C	46.9	R2	0	-											load	47.0	-	30	-
to C	47.2	R4	0	-											to B	67.5	R1	30	-
load	48.4	-	35	-											unload	68.5	-	0	110
to A	69.4	R1	35	-						to C	84.9	R2	0	-					
to A	69.6	R2	35	-						load	85.5	-	25	-					
to A	69.9	R4	35	-						to B	105.5	R2	25	-					
unload	71.1	-	0	95						unload	106.1	-	0	135					
					to C	47.5	R3	0	-						to C	89.0	R1	0	-
					load	48.3	-	25	-						load	90.0	-	30	-
					to A	70.0	R3	25	-						to B	110.5	R1	30	-
					unload	70.8	-	0	120						unload	111.5	-	0	165
					to C	91.8	R1	0	-	to C	126.1	R2	0	-					
					to C	92.0	R2	0	-	load	126.7	-	25	-					
					to C	92.3	R4	0	-	to B	146.7	R2	25	-					
					load	93.1	-	25	-	unload	147.3	-	0	190					
					to A	114.1	R1	25	-						to C	132.0	R1	0	-
					to A	114.3	R2	25	-						load	133.0	-	10	-
					to A	114.6	R4	25	-						to B	153.5	R1	10	-
					unload	115.4	-	0	145						unload	154.5	-	0	200
to C	92.8	R3	0	-											return	171.0	-	0	200
load	94.0	-	35	-						return	164.7	-	0	200					
to A	115.7	R3	35	-															
unload	116.9	-	0	180															
return	133.1	-	0	180															
					return	133.1	-	0	180										
Task Finished										Task Finished									

注: 表中 Act 表示 team 采取的行动, Time 为各行动结束时间点 (或下一步行动起始时间点), Rd 为当前行动使用的道路, Cap 为运输队伍的装载量 (吨), Amnt 表示已完成的任务量, 灰色标记的行动为因协作过程而舍弃的行动.

表 7 资源冲突协调过程对比

方法	GPGP 协调	规划与协调结合	减少
总行动数量	328	74	77.50%
冲突协调次数	10	10	—

的运输任务占用 R1 的时间区间为 [3.6, 24.1](第 6~7 步). 对于这一冲突, 根据使用时间较早优先级较高的

原则, 保持规划器 A 的 Team1 的运输任务不变, 相应地修改规划器 B 的 Team4 的方案, 重新选取 R5 完成该运输任务 (第 8 步). 后续的规划推理中存在多次类似的冲突处理过程 (表中灰色标记处). 如果运用传统方法进行处理, 各个规划器将先各自独立生成一个完整的方案, 而后再进行协调, 不会在规划过程中修改方案. 对于第 7 步的冲突, 若不立刻修改方案, 很可能在之后的协调中将 A 的 19 步和 B 的 25 步作为前提条件失效的行动被舍弃. 此外, 传统的处理方法在完成一轮规划与调整后, 还要针对后续所有的依赖关系完成多轮规划和调整才能得到没有冲突的最终行动方案. 另外, 两种方法下主体规划时具有同样的本地搜索偏好, 本地搜索方案的总体趋势不变, 需要协调的关键行动相同, 因此两种方法下的协调次数相同. 本章设计的方法在没有减少协调次数的情况下, 能够明显减少规划过程, 也从侧面体现其相较于传统协调方式的优越性.

综上所述, 本章设计的在规划过程中进行协调的协作规划方法: 1) 通过有效的协调机制和协作规划, 能够处理主体规划方案间的资源冲突, 并获得互不影响的行动方案; 2) 通过及时协调可以显著改善规划过程, 减少无效方案的规划, 能够在一定程度上减少整体求解过程中的规划工作量, 提高协作任务规划的效率.

4 结论

应急响应决策中存在地理上分散的多个部门单独制定各自的应急行动方案的情况, 尤其各单位对应急资源的使用冲突问题需要有效解决. 本文提出了基于层次任务网络的分布式协作任务规划框架, 实现了协调与规划的实时嵌套, 在规划过程中实时发现并应对依赖关系, 回避了本地行动方案完整生成后协调效率较低的问题, 在一定程度上减少了无效行动生成数量、提高了协作的效率. 另外针对应急响应中较为突出的任务间的资源冲突关系设计了协调机制, 并以此为基础提出了 HTN 协作任务规划算法.

本文主要考虑应急任务之间存在的资源冲突关系, 针对应急管理实践, 需要将任务间的依赖关系进一步细分, 并设计相应的协调机制. 另一方面, 本文主要按照资源占用的时间先后确定任务优先级的方式进行冲突协调, 今后的研究还需要考虑其他因素, 设计更合理的优先级计算规则, 以提高冲突处理的有效性, 改善协作规划的整体效率.

参考文献

[1] 钟永光, 毛中根, 翁文国, 等. 非常规突发事件应急管理研究进展 [J]. 系统工程理论与实践, 2012, 32(5): 911–918.
Zhong Yongguang, Mao Zhonggen, Weng Wenguo, et al. Progress of "study on unconventional emergencies management"[J]. Systems Engineering — Theory & Practice, 2012, 32(5): 911–918.

[2] Salmon P, Stanton N, Jenkins D, et al. Coordination during multi-agency emergency response: Issues and solutions[J]. Disaster Prevention and Management, 2011, 20(2): 140–158.

[3] Tang P, Wang Z, Qi C, et al. Anytime heuristic search in temporal HTN planning for developing incident action plans[J]. AI Communications, 2012, 25(4): 321–342.

[4] Malone T W, Crowston K. The interdisciplinary study of coordination[J]. ACM Computing Surveys (CSUR), 1994, 26(1): 87–119.

[5] Decker K S. Environment centered analysis and design of coordination mechanisms[D]. University of Massachusetts, 1995.

[6] Decker K S. TAEMS: A framework for environment centered analysis and design of coordination mechanisms[J]. Foundations of Distributed Artificial Intelligence, 1996: 429–448.

[7] Durfee E H, Lesser V R. Partial global planning: A coordination framework for distributed hypothesis formation[J]. IEEE Transactions on Systems, Man and Cybernetics, 1991, 21(5): 1167–1183.

[8] Lesser V R, Decker K S, Carver N, et al. Evolution of the GPGP domain independent coordination framework[R]. University of Massachusetts Computer Science Technical Report 1998-05, 1998.

[9] Lesser V R. Reflections on the nature of multi-agent coordination and its implications for an agent architecture[J]. Autonomous Agents and Multi-agent Systems, 1998, 1(1): 89–111.

[10] Wagner T, Lesser V. Toward generalized organizationally contexted agent control[R]. AAAI Technical Report WS-99-14, 1999.

[11] Lesser V R, Decker K S, Wagner T, et al. Evolution of the GPGP/TEAMs domain-independent coordination framework[J]. Autonomous Agents and Multi-Agent Systems, 2004, 9(1): 87–143.

[12] Wagner T, Phelps J, Guralnik V, et al. COORDINATORS: Coordination managers for first responders[C]// Proceedings of the 3rd International Joint Conference on Autonomous Agents and Multiagent Systems, NY, USA, IEEE Computer Society, 2004, 3: 1140–1147.

[13] Musliner D J, Durfee E H, Wu J, et al. Coordinated plan management using multiagent MDPs[R]. Technical Report SS-06-04, AAAI Spring Symposium: Distributed Plan and Schedule Management, 2006: 73–80.
[14] Chen W. Designing an extended set of coordination mechanisms for multi-agent systems[D]. University of Delaware, 2005.
[15] Erol K, Hendler J, Nau D. Complexity results for hierarchical task network planning[J]. Annals of Mathematics and Artificial Intelligence, 1996, 18(1): 69–93.
[16] Dix J, Muñoz-Avila H, Nau D S, et al. IMPACTing SHOP: Putting an AI planner into a multi-agent environment[J]. Annals of Mathematics and Artificial Intelligence, 2003, 37(4): 381–407.
[17] Kabanza F, Lu S, Goodwin S. Distributed hierarchical task planning on a network of clusters[C]// Proceedings of International Conference on Parallel and Distributed Computing and Systems, Calgary, Canada, ACTA Press, 2004: 439–444.
[18] Hayashi H, TokuraS, Ozaki F. Towards real-world HTN planning agents[J]. Knowledge Processing and Decision Making in Agent-based Systems, 2009, 170: 13–41.
[19] Decker K S, Li J. Coordinating mutually exclusive resources using GPGP[J]. Autonomous Agents and Multi-Agent Systems, 2000, 3(2): 1.
[20] Nau D, Tsz-Chiu A, Iighami O, et al. SHOP2: An HTN planning system[J]. Journal of Artificial Intelligence Research, 2003, 20: 379–404.